HITACHI SINGLE-CHIP MICROCOMPUTER

H8/330

HD6473308, HD6433308, HD6413308

HARDWARE MANUAL

Preface

The H8/330 is a high-performance single-chip microcomputer ideally suited for embedded control of industrial equipment. Its core is the H8/300 CPU: a high-speed processor. On-chip supporting modules provide memory, I/O, and timer functions, including:

- 16K bytes of on-chip ROM
- 512 bytes of on-chip RAM
- 15 bytes of dual-port RAM for a master-slave interface
- Serial I/O
- General-purpose I/O ports
- A/D converter
- Timers

Compact, high-performance control systems can be built using the H8/330.

The H8/330 is available with either electrically programmable or mask programmable ROM. Manufacturers can use the electrically programmable ZTAT™* (Zero Turn-Around Time) version to get production off to a fast start and make software changes quickly, then switch over to the masked version for full-scale production runs.

This manual describes the H8/330 hardware. Refer to the *H8/300 Series Programming Manual* for a detailed description of the instruction set.

* ZTAT is a registered trademark of Hitachi, Ltd.

CONTENTS

Table

Figure

Section 1. Overview

The H8/330 is a single-chip microcomputer with an H8/300 CPU core and a complement of on-chip supporting modules. A variety of system functions are integrated onto the H8/330 chip.

The H8/300 CPU is a high-speed Hitachi-original processor with an architecture featuring powerful bit-manipulation instructions, ideally suited for realtime control applications. The on-chip supporting modules include 16K bytes of ROM, 512 bytes of RAM, a 16-bit free-running timer, two 8-bit timers, two PWM timers, a serial communication interface, an A/D converter, dual-port RAM, and I/O ports.

The H8/330 can operate in single-chip mode or in two expanded modes, depending on the memory requirements of the application. The operating mode is referred to in this manual as the MCU mode (MCU: MicroComputer Unit).

The H8/330 is available in a masked ROM version, in an electrically programmable ROM version that can be programmed at the user site, or in a version with no ROM. The no-ROM version can be used only in mode 1.

Section 1.1 shows a block diagram of the H8/330. Section 1.2 describes the main features of the blocks. Section 1.3 shows the pin layout and describes the pin functions.

1.1 Block Diagram

Figure 1-1 shows a block diagram of the H8/330 chip.

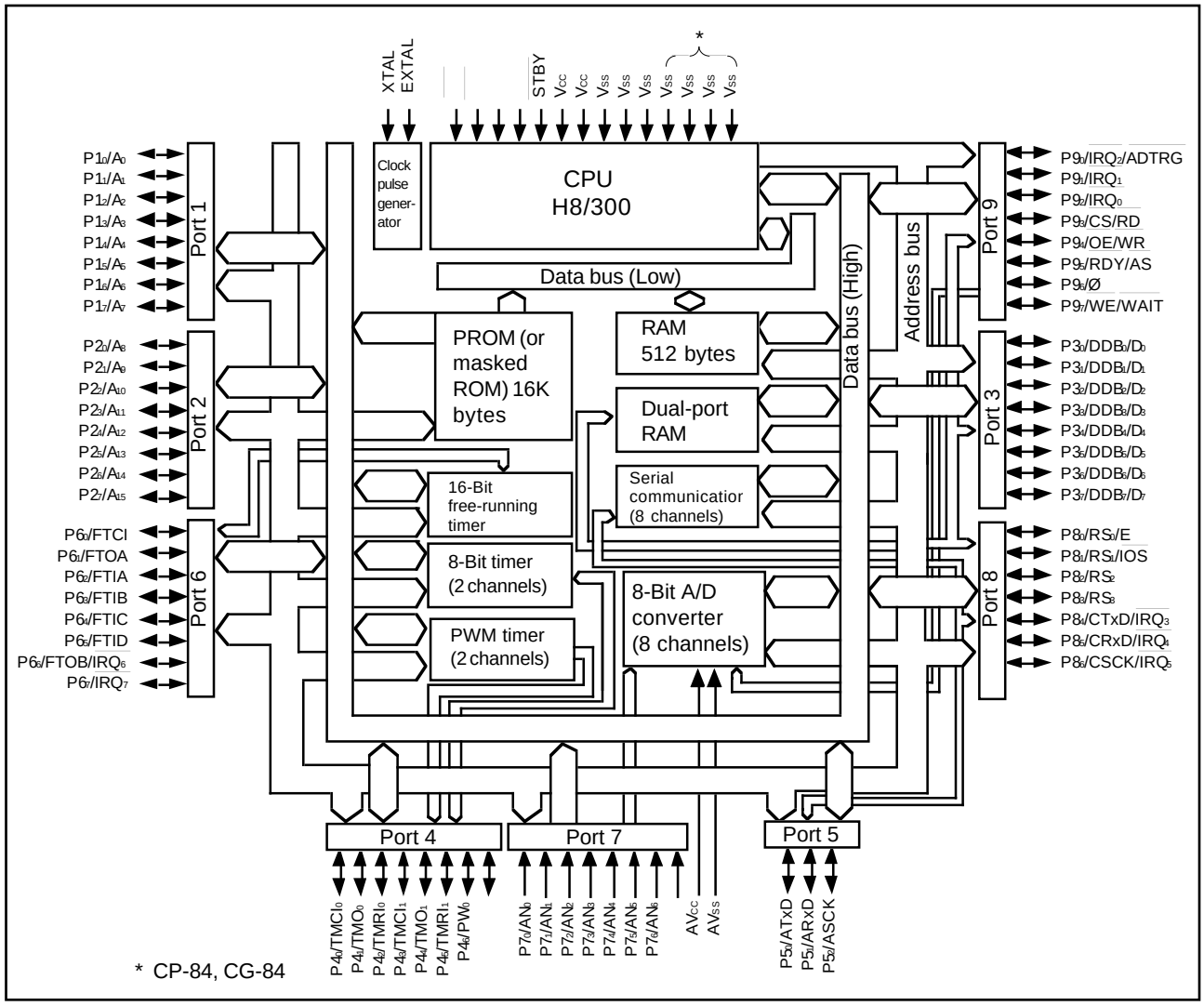


Figure 1-1. Block Diagram

1.2 Descriptions of Blocks

CPU: The CPU has a high-speed-oriented architecture in which operands are located in general registers.

- Two-way general register configuration
  - Eight 16-bit registers, or
  - Sixteen 8-bit registers

- Streamlined instruction set
  - Instruction length: 2 or 4 bytes
  - Register-register arithmetic, logic, and shift operations, including:
    - 8×8-bit multiply
    - $16 \div 8$-bit divide
  - Extensive bit-manipulation instructions, featuring:
    - Bit accumulator
    - Register-indirect specification of bit positions
  - Maximum clock rate: 10MHz
    - Register-register add or subtract: 0.2µs
    - Register-register multiply or divide: 1.4µs

ROM: The 16K-byte on-chip ROM is accessed in two states via a 16-bit bus. Three versions are available:

- Masked ROM
- Electrically programmable ROM, programmable with a standard PROM writer
- No ROM

RAM: The 512-byte on-chip RAM is accessed in two states via a 16-bit bus. RAM contents are held in the power-down state.

Dual-Port RAM: In single-chip mode, the 15 bytes of dual-port memory can be accessed by both the on-chip CPU and an external CPU for convenient parallel data transfer in master-slave systems.

Serial Communication Interface: The single serial I/O channel offers:

- Synchronous or asynchronous communication
- Separate input/output pins for the synchronous and asynchronous modes
- An on-chip baud rate generator supporting up to megabit-per-second speeds
- Serial clock input or output

A/D Converter: A/D conversion can be performed in single or scan mode.

- Eight-bit resolution
- Eight input channels; selection of single mode or scan mode
- Conversion can be started by an external trigger signal
- Sample-and-hold

I/O Ports: Pins not used for other functions are available for general-purpose input and output. I/O is memory-mapped, with the CPU reading and writing the port registers in three states via an 8-bit internal bus.

- 58 Input/output pins (including 16 pins with LED driving capability)
- 8 Input-only pins

Interrupts: With a 10MHz clock rate, interrupt response times are on the order of 2 or 3µs (when the vector table and stack are located in on-chip memory).

- 9 External interrupts: $\overline{\text{NMI}}$ and $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_7$
- 19 Internal interrupts

Free-Running Timer: The time base is a 16-bit free-running counter that can be internally or externally clocked. Applications range from programmable pulse output to counting or timing of external events.

- Two independent, comparator-controlled outputs
- Four input capture channels
- Input capture buffering

8-Bit Timers: Two independent 8-bit timers support applications such as programmable pulse output and external event counting.

- Internal or external clocking
- Output controlled by values in two compare registers

PWM Timers: Two independent timers are provided for pulse-width modulated output. Duty cycles from 0 to 100% can be selected with 1/250 resolution.

Power-Down State: In the three power-down modes some or all chip functions are halted but memory contents are retained.

- Sleep mode: CPU halts to save power while waiting for an interrupt
- Software standby mode: entire chip halts to save power while waiting for an external interrupt
- Hardware standby mode: totally shut down, but on-chip RAM contents are held

Clock Pulse Generator: The H8/330 can generate its system clock from a crystal oscillator, or can input an external clock signal.

E-Clock Interface: An E clock can be output for interfacing to peripheral devices.

MCU Modes: The H8/330 has three operating modes:

- Mode 1: Expanded mode, on-chip ROM disabled
- Mode 2: Expanded mode, on-chip ROM enabled
- Mode 3: Single-chip mode

Product Lineup: A selection is offered of 80- or 84-pin packages with PROM or masked ROM. See table 1-1. The windowed PROM version is UV-erasable.

Table 1-1. Product Lineup

Product code	Package	On-chip ROM
HD6473308F	80-Pin QFP (FP-80A)	PROM
HD6473308CP	84-Pin PLCC (CP-84)	
HD6473308CG	84-Pin windowed LCC (CG-84)	
HD6433308F	80-Pin QFP (FP-80A)	Masked ROM
HD6433308CP	84-Pin PLCC (CP-84)	
HD6413308CP	84-Pin PLCC (CP-84)	No ROM
HD6413308F	80-Pin QFP (FP-80A)	

1.3 Pin Assignments and Functions

1.3.1 Pin Arrangement

Figure 1-2 shows the pin arrangement of the FP-80A package. Figure 1-3 shows the pin arrangement of the CP-84 package. Figure 1-4 shows the pin arrangement of the CG-84 package.

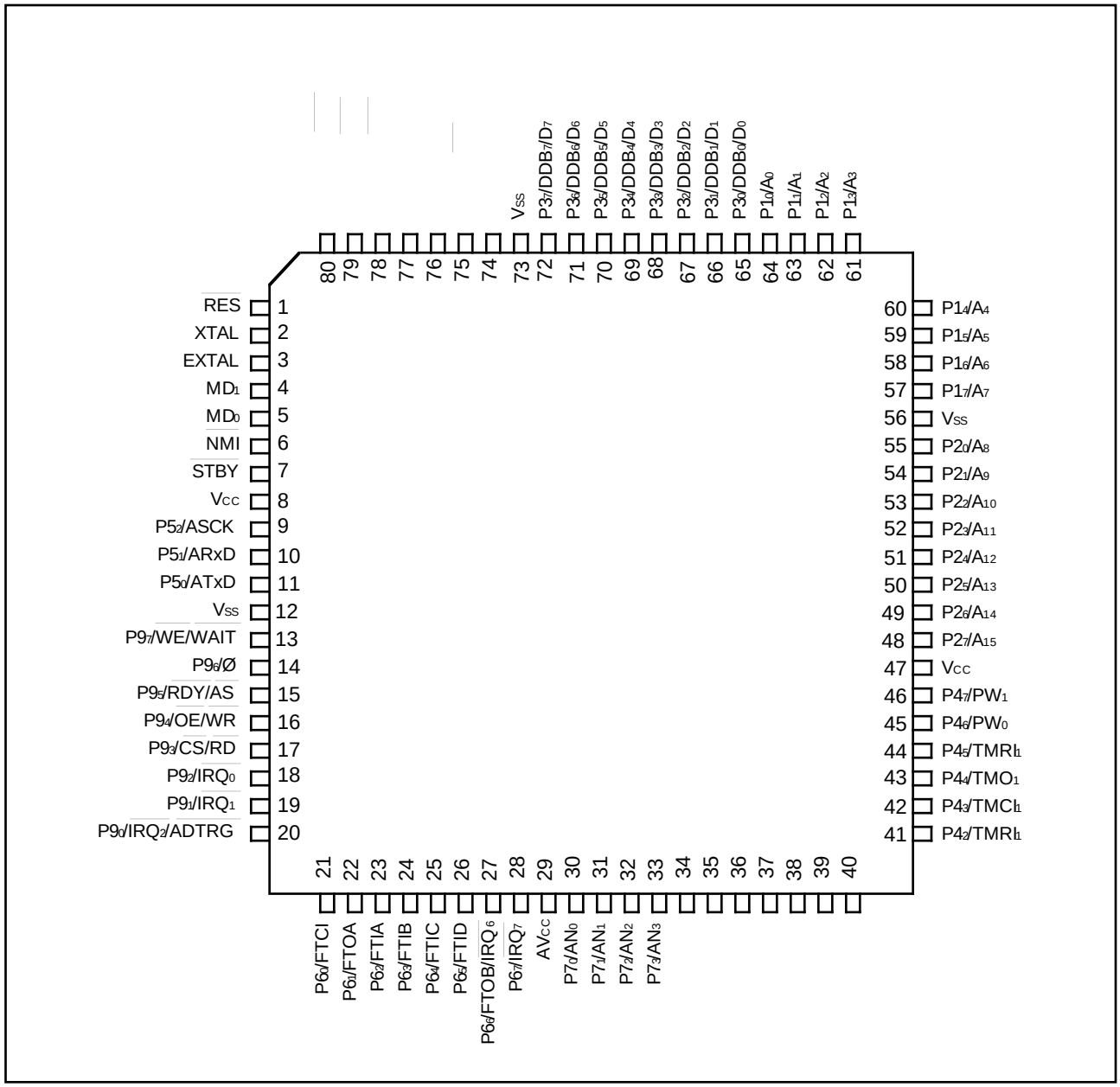


Figure 1-2. Pin Arrangement (FP-80A, Top View)

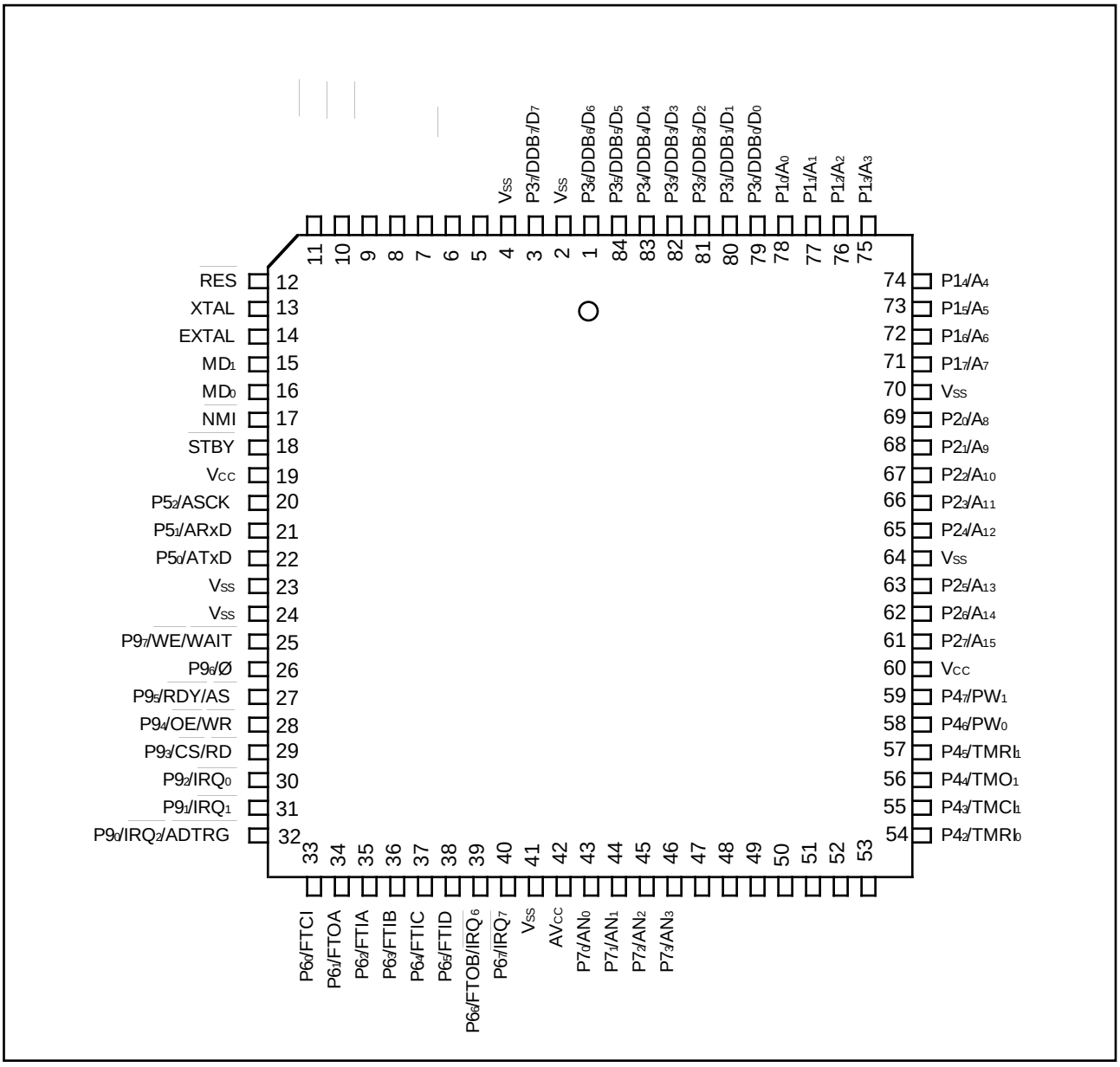


Figure 1-3. Pin Arrangement (CP-84, Top View)

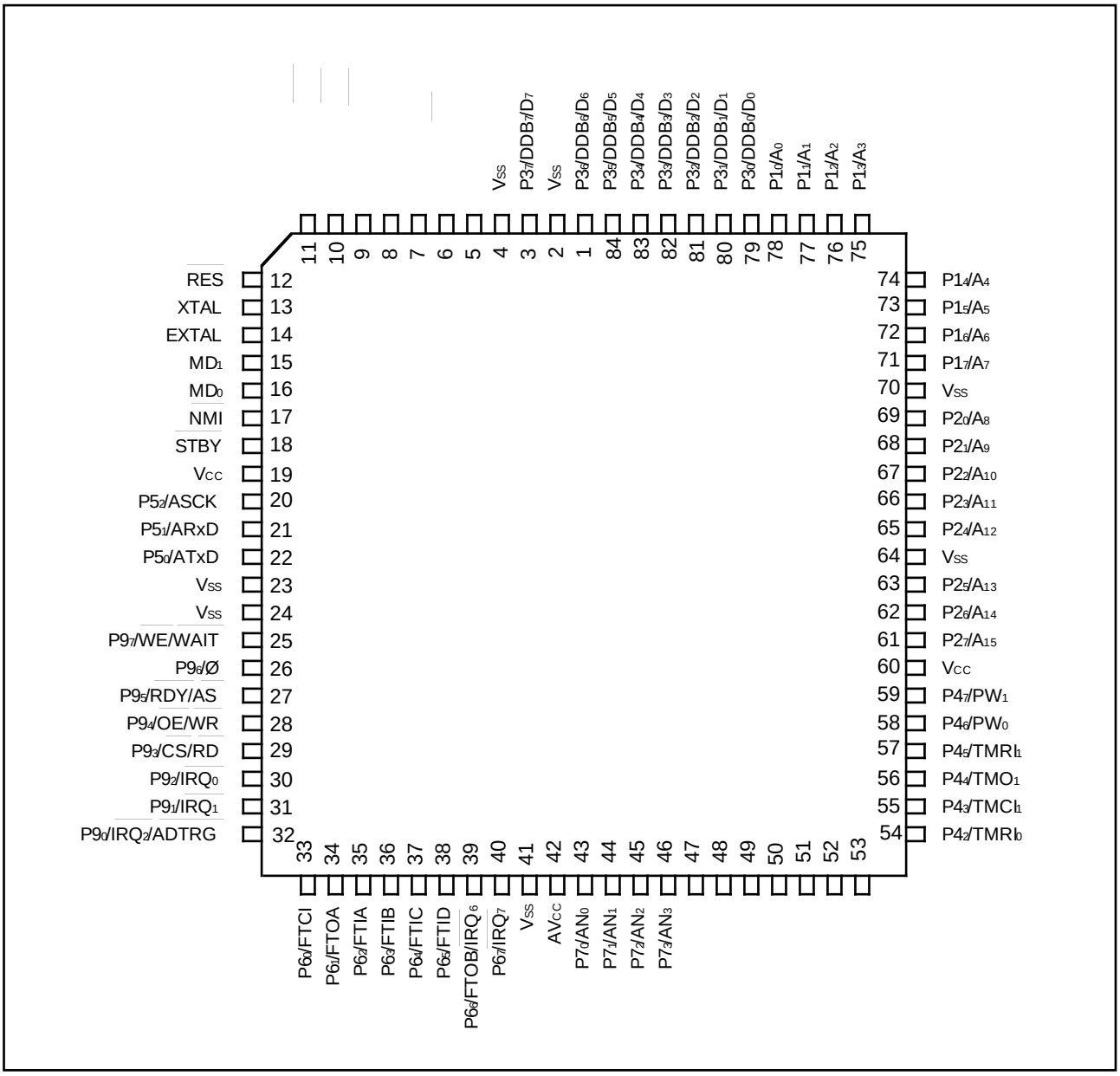


Figure 1-4. Pin Arrangement (CG-84, Top View)

1.3.2 Pin Functions

(1) Pin Assignments in Each Operating Mode: Table 1-2 lists the assignments of the pins of the FP-80A, CP-84, and CG-84 packages in each operating mode.

The PROM mode is a non-operating mode used for programming the on-chip ROM. See section 13, “ROM” for details.

Table 1-2. Pin Assignments in Each Operating Mode (1)

Pin No.		Single-chip mode (mode 3)		Expanded modes	
CP-84 CG-84	**FP -80A**	**DPRAM disabled**	**DPRAM enabled**	**(modes 1 and 2)**	**PROM mode**
1	71	$P3_6$	DDB_6	D_6	EO_6
2	—	V_{SS}	V_{SS}	V_{SS}	V_{SS}
3	72	$P3_7$	DDB_7	D_7	EO_7
4	73	V_{SS}	V_{SS}	V_{SS}	V_{SS}
5	74	$P8_0$	RS_0	$P8_{0*}$ / E	V_{CC}
6	75	$P8_1$	RS_1	$P8_{1*}$ /$\overline{IOS}$	V_{CC}
7	76	$P8_2$	RS_2	$P8_2$	NC
8	77	$P8_3$	RS_3	$P8_3$	NC
9	78	$P8_4$ / CTxD /$\overline{IRQ_3}$	$P8_4$ / CTxD /$\overline{IRQ_3}$	$P8_4$ / CTxD /$\overline{IRQ_3}$	NC
10	79	$P8_5$ / CRxD /$\overline{IRQ_4}$	$P8_5$ / CRxD /$\overline{IRQ_4}$	$P8_5$ / CRxD /$\overline{IRQ_4}$	NC
11	80	$P8_6$ / CSCK /$\overline{IRQ_5}$	$P8_6$ / CSCK /$\overline{IRQ_5}$	$P8_6$ / CSCK /$\overline{IRQ_5}$	NC
12	1	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	V_{PP}
13	2	XTAL	XTAL	XTAL	NC
14	3	EXTAL	EXTAL	EXTAL	NC
15	4	MD_1	MD_1	MD_1	V_{SS}
16	5	MD_0	MD_0	MD_0	V_{SS}
17	6	$\overline{NMI}$	$\overline{NMI}$	$\overline{NMI}$	EA_9
18	7	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	V_{SS}
19	8	V_{CC}	V_{CC}	V_{CC}	V_{CC}
20	9	$P5_2$ / ASCK	$P5_2$ / ASCK	$P5_2$ / ASCK	NC
21	10	$P5_1$ / ARxD	$P5_1$ / ARxD	$P5_1$ / ARxD	NC
22	11	$P5_0$ / ATxD	$P5_0$ / ATxD	$P5_0$ / ATxD	NC

Note: Pins marked NC should be left unconnected.

*Input port only

Table 1-2. Pin Assignments in Each Operating Mode (2)

Pin No.		Single-chip mode (mode 3)		Expanded modes	
CP-84 CG-84	**FP -80A**	**DPRAM disabled**	**DPRAM enabled**	**(modes 1 and 2)**	**PROM mode**
23	12	V_{SS}	V_{SS}	V_{SS}	V_{SS}
24	—	V_{SS}	V_{SS}	V_{SS}	V_{SS}
25	13	$P9_7$	$\overline{WE}$	$\overline{WAIT}$	NC
26	14	$P9_6$ * / Ø	$P9_6$ * / Ø	Ø	NC
27	15	$P9_5$	$\overline{RDY}$	$\overline{AS}$	NC
28	16	$P9_4$	$\overline{OE}$	$\overline{WR}$	NC
29	17	$P9_3$	$\overline{CS}$	$\overline{RD}$	NC
30	18	$P9_2$ / $\overline{IRQ_0}$	$P9_2$ / $\overline{IRQ_0}$	$P9_2$ / $\overline{IRQ_0}$	NC
31	19	$P9_1$ / $\overline{IRQ_1}$	$P9_1$ / $\overline{IRQ_1}$	$P9_1$ / $\overline{IRQ_1}$	NC
32	20	$P9_0$ / $\overline{ADTRG}$ / $\overline{IRQ_2}$	$P9_0$ / $\overline{ADTRG}$ / $\overline{IRQ_2}$	$P9_0$ / $\overline{ADTRG}$ / $\overline{IRQ_2}$	NC
33	21	$P6_0$ / FTCI	$P6_0$ / FTCI	$P6_0$ / FTCI	NC
34	22	$P6_1$ / FTOA	$P6_1$ / FTOA	$P6_1$ / FTOA	NC
35	23	$P6_2$ / FTIA	$P6_2$ / FTIA	$P6_2$ / FTIA	NC
36	24	$P6_3$ / FTIB	$P6_3$ / FTIB	$P6_3$ / FTIB	NC
37	25	$P6_4$ / FTIC	$P6_4$ / FTIC	$P6_4$ / FTIC	NC
38	26	$P6_5$ / FTID	$P6_5$ / FTID	$P6_5$ / FTID	NC
39	27	$P6_6$ / FTOB /$\overline{IRQ_6}$	$P6_6$ / FTOB /$\overline{IRQ_6}$	$P6_6$ / FTOB /$\overline{IRQ_6}$	NC
40	28	$P6_7$ /$\overline{IRQ_7}$	$P6_7$ /$\overline{IRQ_7}$	$P6_7$ /$\overline{IRQ_7}$	NC
41	—	V_{SS}	V_{SS}	V_{SS}	V_{SS}
42	29	AV_{CC}	AV_{CC}	AV_{CC}	V_{CC}
43	30	$P7_0$ / AN_0	$P7_0$ / AN_0	$P7_0$ / AN_0	NC
44	31	$P7_1$ / AN_1	$P7_1$ / AN_1	$P7_1$ / AN_1	NC
45	32	$P7_2$ / AN_2	$P7_2$ / AN_2	$P7_2$ / AN_2	NC
46	33	$P7_3$ / AN_3	$P7_3$ / AN_3	$P7_3$ / AN_3	NC
47	34	$P7_4$ / AN_4	$P7_4$ / AN_4	$P7_4$ / AN_4	NC
48	35	$P7_5$ / AN_5	$P7_5$ / AN_5	$P7_5$ / AN_5	NC
49	36	$P7_6$ / AN_6	$P7_6$ / AN_6	$P7_6$ / AN_6	NC
50	37	$P7_7$ / AN_7	$P7_7$ / AN_7	$P7_7$ / AN_7	NC
51	38	AV_{SS}	AV_{SS}	AV_{SS}	V_{SS}
52	39	$P4_0$ / $TMCI_0$	$P4_0$ / $TMCI_0$	$P4_0$ / $TMCI_0$	NC
53	40	$P4_1$ / TMO_0	$P4_1$ / TMO_0	$P4_1$ / TMO_0	NC

Note: Pins marked NC should be left unconnected.
*Input port only

Table 1-2. Pin Assignments in Each Operating Mode (3)

Pin No.		Single-chip mode (mode 3)		Expanded modes		
CP-84	FP	DPRAM	DPRAM	(modes 1 and 2)		PROM
CG-84	-80A	disabled	enabled			mode
54	41	$P4_2$ / $TMRI_0$	$P4_2$ / $TMRI_0$	$P4_2$ / $TMRI_0$		NC
55	42	$P4_3$ / $TMCI_1$	$P4_3$ / $TMCI_1$	$P4_3$ / $TMCI_1$		NC
56	43	$P4_4$ / TMO_1	$P4_4$ / TMO_1	$P4_4$ / TMO_1		NC
57	44	$P4_5$ / $TMRI_1$	$P4_5$ / $TMRI_1$	$P4_5$ / $TMRI_1$		NC
58	45	$P4_6$ / PW_0	$P4_6$ / PW_0	$P4_6$ / PW_0		NC
59	46	$P4_7$ / PW_1	$P4_7$ / PW_1	$P4_7$ / PW_1		NC
60	47	V_{CC}	V_{CC}	V_{CC}		V_{CC}
61	48	$P2_7$	$P2_7$	A_{15}	$P2_7$* / A_{15}	$\overline{CE}$
62	49	$P2_6$	$P2_6$	A_{14}	$P2_6$* / A_{14}	EA_{14}
63	50	$P2_5$	$P2_5$	A_{13}	$P2_5$* / A_{13}	EA_{13}
64	—	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
65	51	$P2_4$	$P2_4$	A_{12}	$P2_4$* / A_{12}	EA_{12}
66	52	$P2_3$	$P2_3$	A_{11}	$P2_3$* / A_{11}	EA_{11}
67	53	$P2_2$	$P2_2$	A_{10}	$P2_2$* / A_{10}	$\overline{EA}_{10}$
68	54	$P2_1$	$P2_1$	A_9	$P2_1$* / A_9	$\overline{OE}$
69	55	$P2_0$	$P2_0$	A_8	$P2_0$* / A_8	EA_8
70	56	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
71	57	$P1_7$	$P1_7$	A_7	$P1_7$* / A_7	EA_7
72	58	$P1_6$	$P1_6$	A_6	$P1_6$* / A_6	EA_6
73	59	$P1_5$	$P1_5$	A_5	$P1_5$* / A_5	EA_5
74	60	$P1_4$	$P1_4$	A_4	$P1_4$* / A_4	EA_4
75	61	$P1_3$	$P1_3$	A_3	$P1_3$* / A_3	EA_3
76	62	$P1_2$	$P1_2$	A_2	$P1_2$* / A_2	EA_2
77	63	$P1_1$	$P1_1$	A_1	$P1_1$* / A_1	EA_1
78	64	$P1_0$	$P1_0$	A_0	$P1_0$* / A_0	EA_0
79	65	$P3_0$	DDB_0	D_0	D_0	EO_0
80	66	$P3_1$	DDB_1	D_1	D_1	EO_1
81	67	$P3_2$	DDB_2	D_2	D_2	EO_2
82	68	$P3_3$	DDB_3	D_3	D_3	EO_3
83	69	$P3_4$	DDB_4	D_4	D_4	EO_4
84	70	$P3_5$	DDB_5	D_5	D_5	EO_5

Note: Pins marked NC should be left unconnected.

*Input port only

(2) Pin Functions: Table 1-3 gives a concise description of the function of each pin.

Table 1-3. Pin Functions (1)

Type	Symbol	I/O	Name and function
Power	V_{CC}	I	**Power:** Connected to the power supply (+5V). Connect both V_{CC} pins to the system power supply (+5V).
	V_{SS}	I	**Ground:** Connected to ground (0V). Connect all V_{SS} pins to the system power supply (0V).
Clock	XTAL	I	**Crystal:** Connected to a crystal oscillator. The crystal frequency should be double the desired system clock frequency. If an external clock is input at the EXTAL pin, a reverse-phase clock should be input at the XTAL pin.
	EXTAL	I	**External crystal:** Connected to a crystal oscillator or external clock. The frequency of the external clock should be double the desired system clock frequency. See Section 16.2, “Oscillator Circuit” for examples of connections to a crystal and external clock.
	Ø	O	**System clock:** Supplies the system clock to peripheral devices.
	E	O	**Enable clock:** Supplies an E clock to E clock based peripheral devices.
System control	$\overline{RES}$	I	**Reset:** A Low input causes the H8/330 chip to reset.
	$\overline{STBY}$	I	**Standby:** A transition to the hardware standby mode (a power-down state) occurs when a Low input is received at the $\overline{STBY}$ pin.
Address bus	A_{15} to A_0	O	**Address bus:** Address output pins.

Table 1-3. Pin Functions (2)

Type	Symbol	I/O	Name and function
Data bus	D_7 to D_0	I/O	**Data bus:** 8-Bit bidirectional data bus.
Bus control	$\overline{WAIT}$	I	**Wait:** Requests the CPU to insert Tw states into the bus cycle when an off-chip address is accessed.
	$\overline{RD}$	O	**Read:** Goes Low to indicate that the CPU is reading an external address.
	$\overline{WR}$	O	**Write:** Goes Low to indicate that the CPU is writing to an external address.
	$\overline{AS}$	O	**Address Strobe:** Goes Low to indicate that there is a valid address on the address bus.
	$\overline{IOS}$	O	**I/O Select:** Goes Low when the CPU accesses addresses H'FF00 to H'FFFF. Can be used as a chip select signal replacing the upper 8 bits of the address bus when external devices are mapped onto addresses H'FF80 to H'FF8F and H'FFA8 to H'FFAF.
Interrupt signals	$\overline{NMI}$	I	**NonMaskable Interrupt:** Highest-priority interrupt request. The NMIEG bit in the system control register determines whether the interrupt is requested on the rising or falling edge of the NMI input.
	$\overline{IRQ_0}$ to $\overline{IRQ_7}$	I	**Interrupt Request 0 to 7:** Maskable interrupt request pins.
Operating mode control	MD_1, MD_0	I	**Mode:** Input pins for setting the MCU operating mode according to the table below.

MD_1	MD_0	Mode	Description
0	1	Mode 1	Expanded mode with on-chip ROM disabled
1	0	Mode 2	Expanded mode with on-chip ROM enabled
1	1	Mode 3	Single-chip mode

The inputs at these pins are latched in mode select bits 1 to 0 (MDS1 and MDS0) of the mode register (MDCR) on the rising edge of the $\overline{RES}$ signal.

Table 1-3. Pin Functions (3)

Type	Symbol	I/O	Name and function
Serial communication interface	ATxD	O	**Asynchronous Transmit Data:** Asynchronous data output pin for the serial communication interface.
	ARxD	I	**Asynchronous Receive Data:** Asynchronous data input pin for the serial communication interface.
	ASCK	I/O	**Asynchronous Serial ClocK:** Input/output pin for the asynchronous serial clock.
	CTxD	O	**Clock-synchronized Transmit Data:** Synchronous data output pin for the serial communication interface.
	CRxD	I	**Clock-synchronized Receive Data:** Synchronous data input pin for the serial communication interface.
	CSCK	I/O	**Clock-synchronized Serial ClocK:** Input/output pin for the synchronous serial clock.
16-Bit free-running timer	FTOA, FTOB	O	**FRT Output compare A and B:** Output pins controlled by comparators A and B of the free-running timer.
	FTCI	I	**FRT counter Clock Input:** Input pin for an external clock signal for the free-running timer.
	FTIA to FTID	I	**FRT Input capture A to D:** Input capture pins for the free-running timer.
8-Bit timer	TMO_0, TMO_1	O	**8-bit TiMer Output:** Compare-match output pins for the 8-bit timers.
	$TMCI_0$, $TMCI_1$	I	**8-bit TiMer counter Clock Input:** External clock input pins for the 8-bit timer counters.
	$TMRI_0$, $TMRI_1$	I	**8-bit TiMer counter Reset Input:** A High input at these pins resets the 8-bit timer counters.
PWM timer	PW_0, PW_1	O	**PWM timer output (channels 0 and 1):** Pulse-width modulation timer output pins.
A/D converter	AN_7 to AN_0	I	**ANalog input:** Analog signal input pins.
	$\overline{ADTRG}$	I	**A/D Trigger:** External trigger input for starting the A/D converter.
	AV_{CC}	I	**Analog reference Voltage:** Reference voltage pin for the A/D converter.
	AV_{SS}	I	**Analog ground:** Ground pin for the A/D converter.

Table 1-3. Pin Functions (4)

Type	Symbol	I/O	Name and function
Dual-port RAM	DDB_7 to DDB_0	I/O	**Dual-port RAM Data Bus:** Bidirectional 8-bit bus by which an external CPU can access the dual-port RAM.
	$\overline{CS}$	I	**Chip Select:** Input pin for selecting the dual-port RAM.
	RS_3 to RS_0	I	**Register Select:** Input pins for addressing the dual-port RAM.
	$\overline{OE}$	I	**Output Enable:** Enables output on DDB_7 to DDB_0.
	$\overline{WE}$	I	**Write Enable:** Enables writing to the dual-port RAM.
	$\overline{RDY}$	O	**ReaDY:** For sending an interrupt request signal to an external CPU. NMOS open-drain output.
General-purpose I/O	$P1_7$ to $P1_0$	I/O	**Port 1:** An 8-bit input/output port with programmable MOS input pull-ups and LED driving capability. The direction of each bit can be selected in the port 1 data direction register (P1DDR).
	$P2_7$ to $P2_0$	I/O	**Port 2:** An 8-bit input/output port with programmable MOS input pull-ups and LED driving capability. The direction of each bit can be selected in the port 2 data direction register (P2DDR).
	$P3_7$ to $P3_0$	I/O	**Port 3:** An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 3 data direction register (P3DDR).
	$P4_7$ to $P4_0$	I/O	**Port 4:** An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 4 data direction register (P4DDR).
	$P5_2$ to $P5_0$	I/O	**Port 5:** A 3-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 5 data direction register (P5DDR).
	$P6_7$ to $P6_0$	I/O	**Port 6:** An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 6 data direction register (P6DDR).
	$P7_7$ to $P7_0$	I	**Port 7:** An 8-bit input port.
	$P8_6$ to $P8_0$	I/O	**Port 8:** A 7-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 8 data direction register (P8DDR).
	$P9_7$ to $P9_0$	I/O	**Port 9:** An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 9 data direction register (P9DDR).

Section 2. MCU Operating Modes and Address Space

2.1 Overview

The H8/330 operates in three modes numbered 1, 2, and 3. An additional non-operating mode (mode 0) is used for programming the PROM version of the chip. The mode is selected by the inputs at the mode pins (MD_1 and MD_0) at the instant when the chip comes out of a reset. As indicated in table 2-1, the mode determines the size of the address space and the usage of on-chip ROM and on-chip RAM.

The no-ROM version (HD6413308) can operate only in mode 1 (expanded mode with on-chip ROM disabled).

Table 2-1. Operating Modes

MD_1	MD_0	Mode	Address space	On-chip ROM	On-chip RAM
Low	Low	Mode 0	—	—	—
Low	High	Mode 1	Expanded: 64KB	Disabled	Enabled*
High	Low	Mode 2	Expanded: 64KB	Enabled	Enabled*
High	High	Mode 3	Single-chip: about 17KB	Enabled	Enabled

* In modes 1 and 2, external memory can be accessed instead of on-chip RAM by clearing the RAME bit in the system control register (SYSCR) to "0."

Modes 1 and 2 are referred to as “expanded” because they permit access to off-chip memory addresses.

2.2 Mode Descriptions

Mode 1 (Expanded Mode without On-Chip ROM): Mode 1 supports a 64K-byte address space most of which is off-chip. In particular, the interrupt vector table is located in off-chip memory. The on-chip ROM and dual-port RAM are not used. Software can select whether to use the on-chip RAM. Ports 1 to 3, 8, and 9 are used for the address and data bus lines and control signals as follows:

Ports 1 and 2:	Address bus
Port 3:	Data bus
Port 8 (pin 1), port 9 (pins 7, 5, 4, 3):	Bus control signals

Mode 2 (Expanded Mode with On-Chip ROM): Mode 2 supports a 64K-byte address space of which the first 16K bytes are in on-chip ROM. Software can select whether or not to use the on-chip RAM, and can select the usage of pins in ports 1 and 2.

Ports 1 and 2:	Address bus (see note)
Port 3:	Data bus
Port 8 (pin 1), port 9 (pins 7, 5, 4, 3):	Bus control signals

Note: In mode 2, ports 1 and 2 are initially general-purpose input ports. Software must change the desired pins to output before using them for the address bus. See section 5, “I/O Ports” for details.

Mode 3 (Single-Chip Mode): In this mode all memory is on-chip, in 16K bytes of ROM, 512 bytes of RAM, and internal I/O registers. If enabled by software, the dual-port RAM can be accessed by an external CPU.

Since no off-chip memory is accessed, there is no address bus; ports 1 and 2 are available for general-purpose input and output. When the dual-port RAM is enabled, ports 3, 8, and 9 are used as follows:

Port 3:	Dual-port RAM data bus
Port 8 (pins 0 to 3):	Dual-port RAM register select
Port 9 (pins 7, 5, 4, 3):	Dual-port RAM interface signals

The mode in which the dual-port RAM is enabled is also called the slave mode.

2.3 Address Space Map

Figure 2-1 shows a memory map in each of the three operating modes. The on-chip register field consists of control, status, and data registers for the on-chip supporting modules, I/O ports, and dual-port RAM.

Off-chip addresses can be accessed only in the expanded modes. Access to an off-chip address in the single-chip mode does not cause an address error, but all “1” data are returned.

2.3.1 Access Speed

On-chip ROM and RAM are accessed a word (16 bits) at a time in two states. (A “state” is one system clock period.) The on-chip register field is accessed a byte at a time in three states.

External memory is accessed a byte at a time in three or more states. The basic bus cycle is three states, but additional wait states can be inserted on request.

2.3.2 $\overline{\text{IOS}}$

There are two small gaps in the on-chip address space above the on-chip RAM. Addresses H’FF80 to H’FF8F, situated between the on-chip RAM and register field, are off-chip. Addresses H’FFA8 to H’FFAF are also off-chip. These 24 addresses can be conveniently assigned to external I/O devices.

To simplify the addressing of devices at these addresses, an $\overline{\text{IOS}}$ signal is provided that goes Low when the CPU accesses addresses H’FF00 to H’FFFF. The $\overline{\text{IOS}}$ signal can be used in place of the upper 8 bits of the address bus.

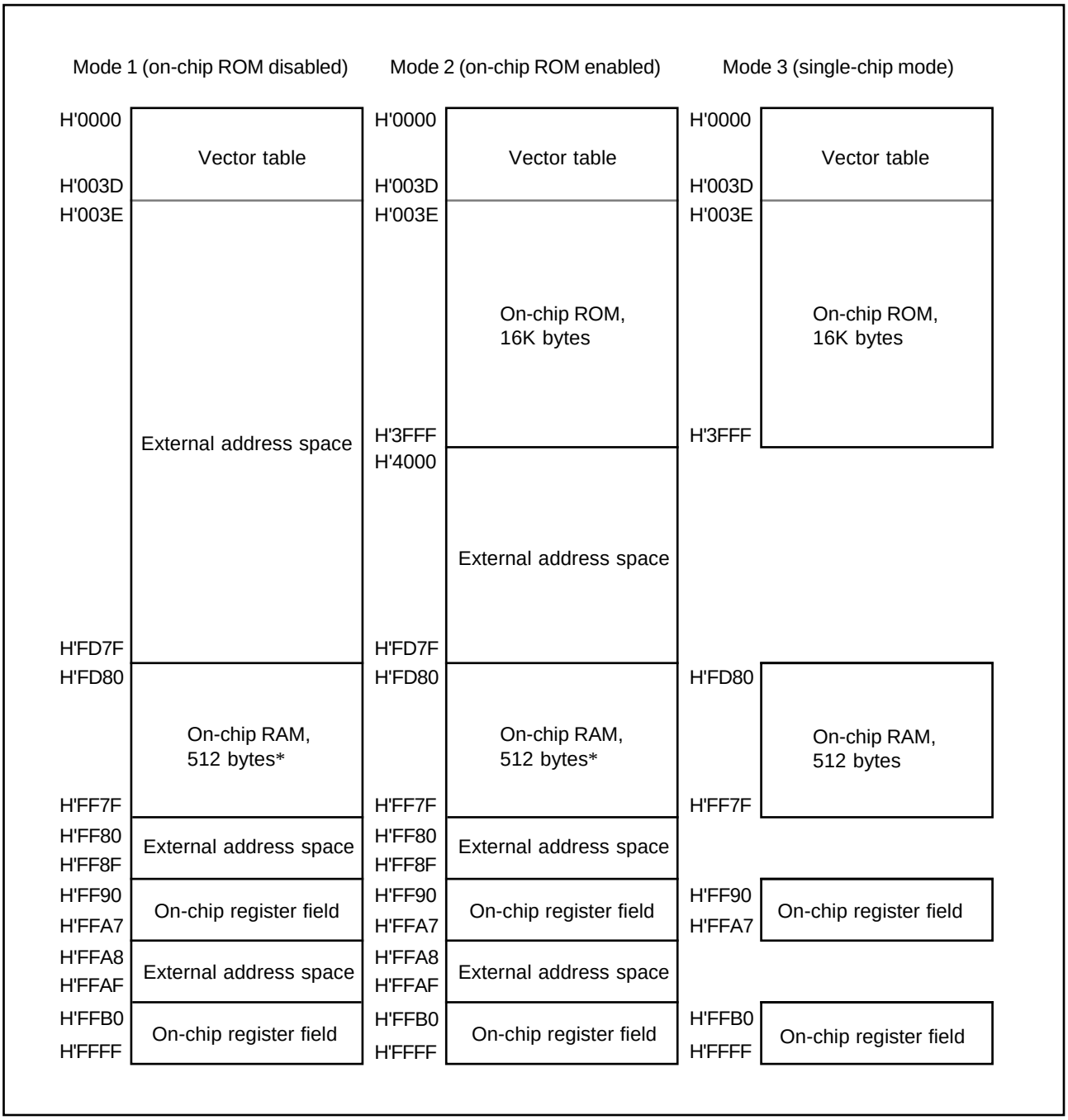


Figure 2-1. Address Space Map

* External memory can be accessed at these addresses when the RAME bit in the system control register (SYSCR) is cleared to "0".

2.4 Mode and System Control Registers (MDCR and SYSCR)

Two of the control registers in the register field are the mode control register (MDCR) and system control register (SYSCR). The mode control register controls the MCU mode: the operating mode of the H8/330 chip. The system control register has bits that enable or disable the on-chip RAM and dual-port RAM. Table 2-2 lists the attributes of these registers.

Table 2-2. Mode and System Control Registers

Name	Abbreviation	Read/Write	Address
Mode control register	MDCR	R	H'FFC5
System control register	SYSCR	R/W	H'FFC4

2.4.1 Mode Control Register (MDCR) – H'FFC5

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	MDS1	MDS0
Initial value	1	1	1	0	0	1	*	*
Read/Write	—	—	—	—	—	—	R	R

* Initialized according to MD1 and MD0 inputs.

Bits 7 to 5 and 2—Reserved: These bits cannot be modified and are always read as "1."

Bits 4 and 3—Reserved: These bits cannot be modified and are always read as "0."

Bits 1 and 0—Mode Select 1 and 0 (MDS1 and MDS0): These bits indicate the values of the mode pins (MD1 and MD0) latched on the rising edge of the $\overline{\text{RES}}$ signal. These bits can be read but not written.

Coding example: To test whether the MCU is operating in mode 1:

```
MOV.B @H'FFC5, R0L
CMP.B #H'E5, R0L
```

The comparison is with H'E5 instead of H'01 because bits 7, 6, 5, and 2 are always read as "1."

2.4.2 System Control Register (SYSCR) – H'FFC4

By setting or clearing the lower two bits of the system control register, software can enable or disable the on-chip RAM and dual-port RAM.

The other bits in the system control register concern the software standby mode and the valid edge of the $\overline{\text{NMI}}$ signal. These bits will be described in section 4, "Exception Handling" and section 14, "Power-Down State."

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W

Bit 1—Dual-Port RAM Enable (DPME): In the single-chip mode, this bit enables or disables the dual-port RAM. When enabled, the dual-port RAM can be accessed by both an external (master) CPU and the on-chip (slave) CPU. When disabled, the dual-port RAM can be accessed only by the on-chip CPU.

This bit affects the usage of ports 3, 8, and 9.

Bit 1

DPME	Description
0	The dual-port RAM is disabled. (Initial state)
1	Single-chip mode: The dual-port RAM is enabled (slave mode).
	Expanded modes: The dual-port RAM is disabled (but can be accessed by the on-chip CPU).

Bit 0—RAM Enable (RAME): This bit enables or disables the 512-byte on-chip RAM. When enabled, the on-chip RAM occupies addresses H'FD80 to H'FF7F of the address space. When the on-chip RAM is disabled, accesses to these addresses are directed off-chip.

The RAME bit is initialized to "1" by a reset, enabling the on-chip RAM. The setting of the RAME bit is not altered in the sleep mode or software standby mode. It should be cleared to "0" before entering the hardware standby mode. See section 14, "Power-Down State."

Bit 0

RAME	Description
0	The on-chip RAM is disabled.
1	The on-chip RAM is enabled. (Initial state)

Coding Examples: To disable the on-chip RAM (in expanded modes):

```
BCLR #0, @H'FFC4
```

To enable the dual-port RAM (in single-chip mode):

```
BSET #1, @H'FFC4
```

Section 3. CPU

3.1 Overview

The H8/330 chip has the generic H8/300 CPU: an 8-bit central processing unit with a speed-oriented architecture featuring sixteen general registers. This section describes the CPU features and functions, including a concise description of the addressing modes and instruction set. For further details on the instructions, see the *H8/300 Series Programming Manual*.

3.1.1 Features

The main features of the H8/300 CPU are listed below.

- Two-way register configuration
  - — Sixteen 8-bit general registers, or
  - — Eight 16-bit general registers
- Instruction set with 57 basic instructions, including:
  - — Multiply and divide instructions
  - — Powerful bit-manipulation instructions
- Eight addressing modes
  - — Register direct (Rn)
  - — Register indirect (@Rn)
  - — Register indirect with displacement (@(d:16, Rn))
  - — Register indirect with post-increment or pre-decrement (@Rn+ or @–Rn)
  - — Absolute address (@aa:8 or @aa:16)
  - — Immediate (#xx:8 or #xx:16)
  - — PC-relative (@(d:8, PC))
  - — Memory indirect (@@aa:8)
- Maximum 64K-byte address space
- High-speed operation
  - — All frequently-used instructions are executed two to four states
  - — The maximum clock rate is 10MHz
    - — 8- or 16-bit register-register add or subtract: 0.2µs
    - — 8 × 8-bit multiply: 1.4µs
    - — 16 ÷ 8-bit divide: 1.4µs
- Power-down mode
  - — SLEEP instruction

3.2 Register Configuration

Figure 3-1 shows the register structure of the CPU. There are two groups of registers: the general registers and control registers.

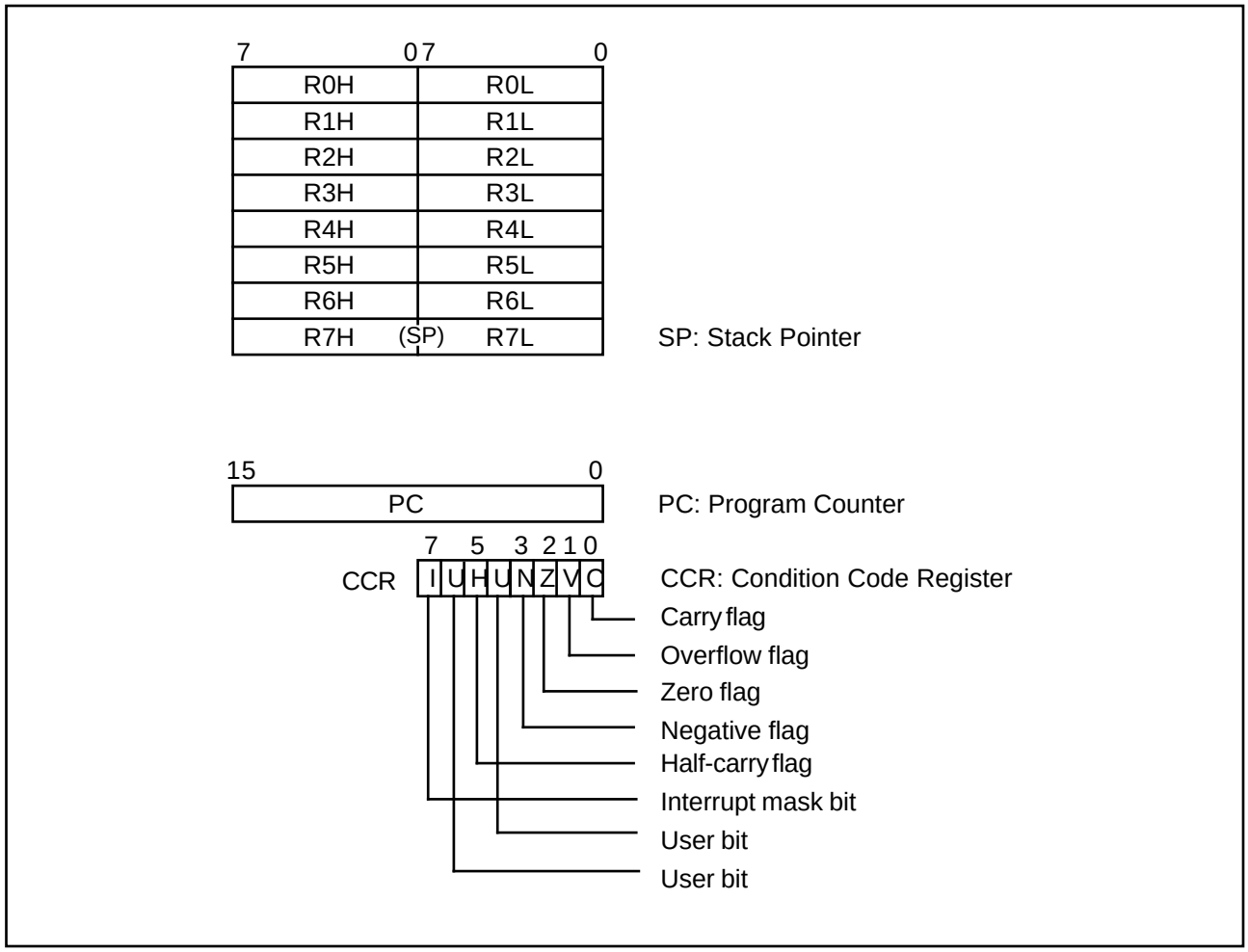


Figure 3-1. CPU Registers

3.2.1 General Registers

All the general registers can be used as both data registers and address registers. When used as address registers, the general registers are accessed as 16-bit registers (R0 to R7). When used as data registers, they can be accessed as 16-bit registers, or the high and low bytes can be accessed separately as 8-bit registers.

R7 also functions as the stack pointer, used implicitly by hardware in processing interrupts and subroutine calls. In assembly-language coding, R7 can also be denoted by the letters SP. As indicated in figure 3-2, R7 (SP) points to the top of the stack.

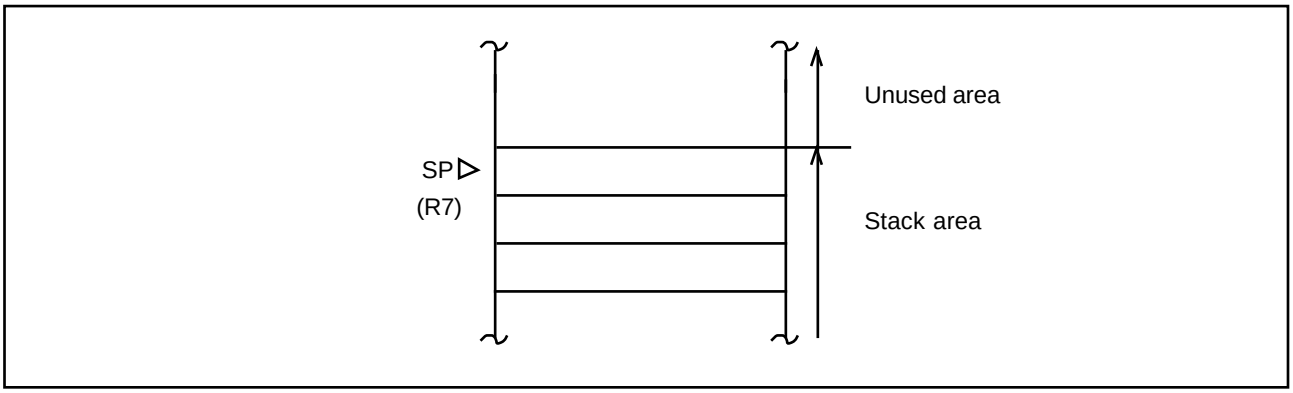


Figure 3-2. Stack Pointer

3.2.2 Control Registers

The CPU control registers include a 16-bit program counter (PC) and an 8-bit condition code register (CCR).

(1) Program Counter (PC): This 16-bit register indicates the address of the next instruction the CPU will execute. Each instruction is accessed in 16 bits (1 word), so the least significant bit of the PC is ignored (always regarded as 0).

(2) Condition Code Register (CCR): This 8-bit register contains internal status information, including carry (C), overflow (V), zero (Z), negative (N), and half-carry (H) flags and the interrupt mask bit (I).

Bit 7—Interrupt Mask Bit (I): When this bit is set to "1," all interrupts except NMI are masked. This bit is set to "1" automatically by a reset and at the start of interrupt handling.

Bit 6—User Bit (U): This bit can be written and read by software for its own purposes.

Bit 5—Half-Carry (H): This bit is set to "1" when the ADD.B, ADDX.B, SUB.B, SUBX.B, NEG.B, or CMP.B instruction causes a carry or borrow out of bit 3, and is cleared to "0" otherwise. Similarly, it is set to "1" when the ADD.W, SUB.W, or CMP.W instruction causes a carry or borrow out of bit 11, and cleared to "0" otherwise. It is used implicitly in the DAA and DAS instructions.

Bit 4—User Bit (U): This bit can be written and read by software for its own purposes.

Bit 3—Negative (N): This bit indicates the most significant bit (sign bit) of the result of an instruction.

Bit 2—Zero (Z): This bit is set to “1” to indicate a zero result and cleared to “0” to indicate a nonzero result.

Bit 1—Overflow (V): This bit is set to “1” when an arithmetic overflow occurs, and cleared to “0” at other times.

Bit 0—Carry (C): This bit is used by:

- Add and subtract instructions, to indicate a carry or borrow at the most significant bit of the result
- Shift and rotate instructions, to store the value shifted out of the most significant or least significant bit
- Bit manipulation and bit load instructions, as a bit accumulator

The LDC, STC, ANDC, ORC, and XORC instructions enable the CPU to load and store the CCR, and to set or clear selected bits by logic operations.

Some instructions leave some or all of the flag bits unchanged. The action of each instruction on the flag bits is shown in Appendix A-1, “Instruction Set List.” See the *H8/300 Series Programming Manual* for further details.

3.2.3 Initial Register Values

When the CPU is reset, the program counter (PC) is loaded from the vector table and the interrupt mask bit (I) in the CCR is set to “1.” The other CCR bits and the general registers are not initialized.

In particular, the stack pointer (R7) is not initialized. To prevent program crashes the stack pointer should be initialized by software, by the first instruction executed after a reset.

3.3 Addressing Modes

The H8/330 supports eight addressing modes. Each instruction uses a subset of these addressing modes.

(1) Register Direct—Rn: The register field of the instruction specifies an 8- or 16-bit general register containing the operand. In most cases the general register is accessed as an 8-bit register. Only the MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (8 bits × 8 bits), and DIVXU (16 bits ÷ 8 bits) instructions have 16-bit operands.

(2) Register indirect—@Rn: The register field of the instruction specifies a 16-bit general register containing the address of the operand.

(3) Register Indirect with Displacement—@(d:16, Rn): This mode, which is used only in MOV instructions, is similar to register indirect but the instruction has a second word (bytes 3 and 4) which is added to the contents of the specified general register to obtain the operand address. For the MOV.W instruction, the resulting address must be even.

(4) Register Indirect with Post-Increment or Pre-Decrement—@Rn+ or @–Rn:

- Register indirect with Post-Increment—@Rn+
  The @Rn+ mode is used with MOV instructions that load registers from memory.
  It is similar to the register indirect mode, but the 16-bit general register specified in the register field of the instruction is incremented after the operand is accessed. The size of the increment is 1 or 2 depending on the size of the operand: 1 for MOV.B; 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

- Register Indirect with Pre-Decrement—@–Rn
  The @–Rn mode is used with MOV instructions that store register contents to memory.
  It is similar to the register indirect mode, but the 16-bit general register specified in the register field of the instruction is decremented before the operand is accessed. The size of the decrement is 1 or 2 depending on the size of the operand: 1 for MOV.B; 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

(5) Absolute Address—@aa:8 or @aa:16: The instruction specifies the absolute address of the operand in memory. The MOV.B instruction uses an 8-bit absolute address of the form H'FFxx. The upper 8 bits are assumed to be 1, so the possible address range is H'FF00 to H'FFFF (65280 to 65535). The MOV.B, MOV.W, JMP, and JSR instructions can use 16-bit absolute addresses.

(6) Immediate—#xx:8 or #xx:16: The instruction contains an 8-bit operand in its second byte, or a 16-bit operand in its third and fourth bytes. Only MOV.W instructions can contain 16-bit immediate values.
The ADDS and SUBS instructions implicitly contain the value 1 or 2 as immediate data. Some bit manipulation instructions contain 3-bit immediate data (#xx:3) in the second or fourth byte of the instruction, specifying a bit number.

(7) PC-Relative—@(d:8, PC): This mode is used to generate branch addresses in the Bcc and BSR instructions. An 8-bit value in byte 2 of the instruction code is added as a sign-extended value to the program counter contents. The result must be an even number. The possible branching range is –126 to +128 bytes (–63 to +64 words) from the current address.

(8) Memory Indirect—@@aa:8: This mode can be used by the JMP and JSR instructions. The second byte of the instruction code specifies an 8-bit absolute address from H’0000 to H’00FF (0 to 255). The word located at this address contains the branch address. Note that addresses H’0000 to H’003D (0 to 61) are located in the vector table.

If an odd address is specified as a branch destination or as the operand address of a MOV.W instruction, the least significant bit is regarded as “0,” causing word access to be performed at the address preceding the specified address. See section 3.4.2, “Memory Data Formats” for further information.

3.4 Data Formats

The H8/300 CPU can process 1-bit data, 4-bit (BCD) data, 8-bit (byte) data, and 16-bit (word) data.

- Bit manipulation instructions operate on 1-bit data specified as bit n (n = 0, 1, 2, ..., 7) in a byte operand.
- All arithmetic and logic instructions except ADDS and SUBS can operate on byte data.
- The DAA and DAS instruction perform decimal arithmetic adjustments on byte data in packed BCD form. Each nibble of the byte is treated as a decimal digit.
- The MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (8 bits × 8 bits), and DIVXU (16 bits ÷ 8 bits) instructions operate on word data.

3.4.1 Data Formats in General Registers

Data of all the sizes above can be stored in general registers as shown in figure 3-3.

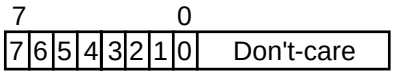
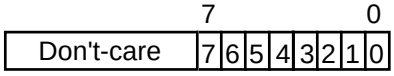
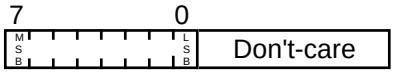
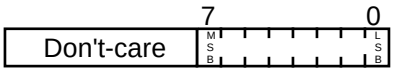
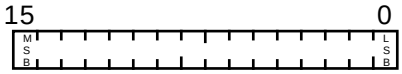
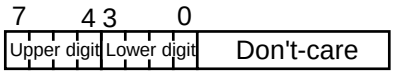
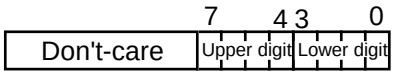
Data type	Register No.	Data format
1-Bit data	RnH	
1-Bit data	RnL	
Byte data	RnH	
Byte data	RnL	
Word data	Rn	
4-Bit BCD data	RnH	
4-Bit BCD data	RnL	

Figure 3-3. Register Data Formats

Note:

RnH: Upper digit of general register

RnL: Lower digit of general register

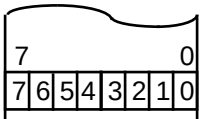
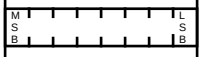
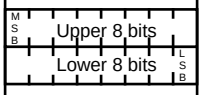
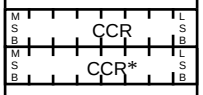
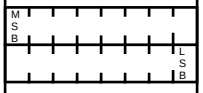
MSB: Most significant Bit

LSB: Least significant Bit

3.4.2 Memory Data Formats

Figure 3-4 indicates the data formats in memory.

Word data stored in memory must always begin at an even address. In word access the least significant bit of the address is regarded as “0.” If an odd address is specified, no address error occurs but the access is performed at the preceding even address. This rule affects MOV.W instructions and branching instructions, and implies that only even addresses should be stored in the vector table.

Data type	Address	Data format
1-Bit data	Address n	
Byte data	Address n	
Word data	Even address Odd address	
Byte data (CCR) on stack	Even address Odd address	
Word data on stack	Even address Odd address	

CCR: Condition Code Register

*: Ignored when return

Figure 3-4. Memory Data Formats

The stack must always be accessed a word at a time. When the CCR is pushed on the stack, two identical copies of the CCR are pushed to make a complete word. When they are returned, the lower byte is ignored.

3.5 Instruction Set

Table 3-1 lists the H8/330 instruction set.

Table 3-1. Instruction Classification

Function	Instructions	Types
Data transfer	MOV, MOVTPE, MOVFPE, PUSH*1, POP*1	3
Arithmetic operations	ADD, SUB, ADDX, SUBX, INC, DEC, ADDS, SUBS, DAA, DAS, MULXU, DIVXU, CMP, NEG	14
Logic operations	AND, OR, XOR, NOT	4
Shift	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	14
Branch	Bcc*2, JMP, BSR, JSR, RTS	5
System control	RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	8
Block data transfer	EEPMOV	1
	Total	57

*1 PUSH Rn is equivalent to MOV.W Rn, @–SP.
POP Rn is equivalent to MOV.W @SP+, Rn.

*2 Bcc is a conditional branch instruction in which cc represents a condition code.

The following sections give a concise summary of the instructions in each category, and indicate the bit patterns of their object code. The notation used is defined next.

Operation Notation

Rd	General register (destination)
Rs	General register (source)
Rn, Rm	General register
r_n, r_m	General register field
<EAs>	Effective address: general register or memory location
(EAd)	Destination operand
(EAs)	Source operand
SP	Stack pointer
PC	Program counter
CCR	Condition code register
N	N (negative) bit of CCR
Z	Z (zero) bit of CCR
V	V (overflow) bit of CCR
C	C (carry) bit of CCR
#imm	Immediate data
#xx:3	3-Bit immediate data
#xx:8	8-Bit immediate data

#xx:16	16-Bit immediate data
op	Operation field
disp	Displacement
abs	Absolute address
B	Byte
W	Word
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	AND logical
∨	OR logical
⊕	Exclusive OR logical
→	Move
↔	Exchange
¬	Not
cc	Condition field

3.5.1 Data Transfer Instructions

Table 3-2 describes the data transfer instructions. Figure 3-5 shows their object code formats.

Table 3-2. Data Transfer Instructions

Instruction	Size*	Function
MOV	B/W	(EAs) → Rd, Rs → (EAd) Moves data between two general registers or between a general register and memory, or moves immediate data to a general register. The Rn, @Rn, @(d:16, Rn), @aa:16, #xx:8 or #xx:16, @–Rn, and @Rn+ addressing modes are available for byte or word data. The @aa:8 addressing mode is available for byte data only. The @–R7 and @R7+ modes require word operands. Do not specify byte size for these two modes.
MOVTPE	B	Rs → (EAd) Transfers data from a general register to memory in synchronization with the E clock.
MOVFPE	B	(EAs) → Rd Transfers data from memory to a general register in synchronization with the E clock.
PUSH	W	Rn → @–SP Pushes a 16-bit general register onto the stack. Equivalent to MOV.W Rn, @–SP.
POP	W	@SP+ → Rn Pops a 16-bit general register from the stack. Equivalent to MOV.W @SP+, Rn.

* Size: operand size
B: Byte
W: Word

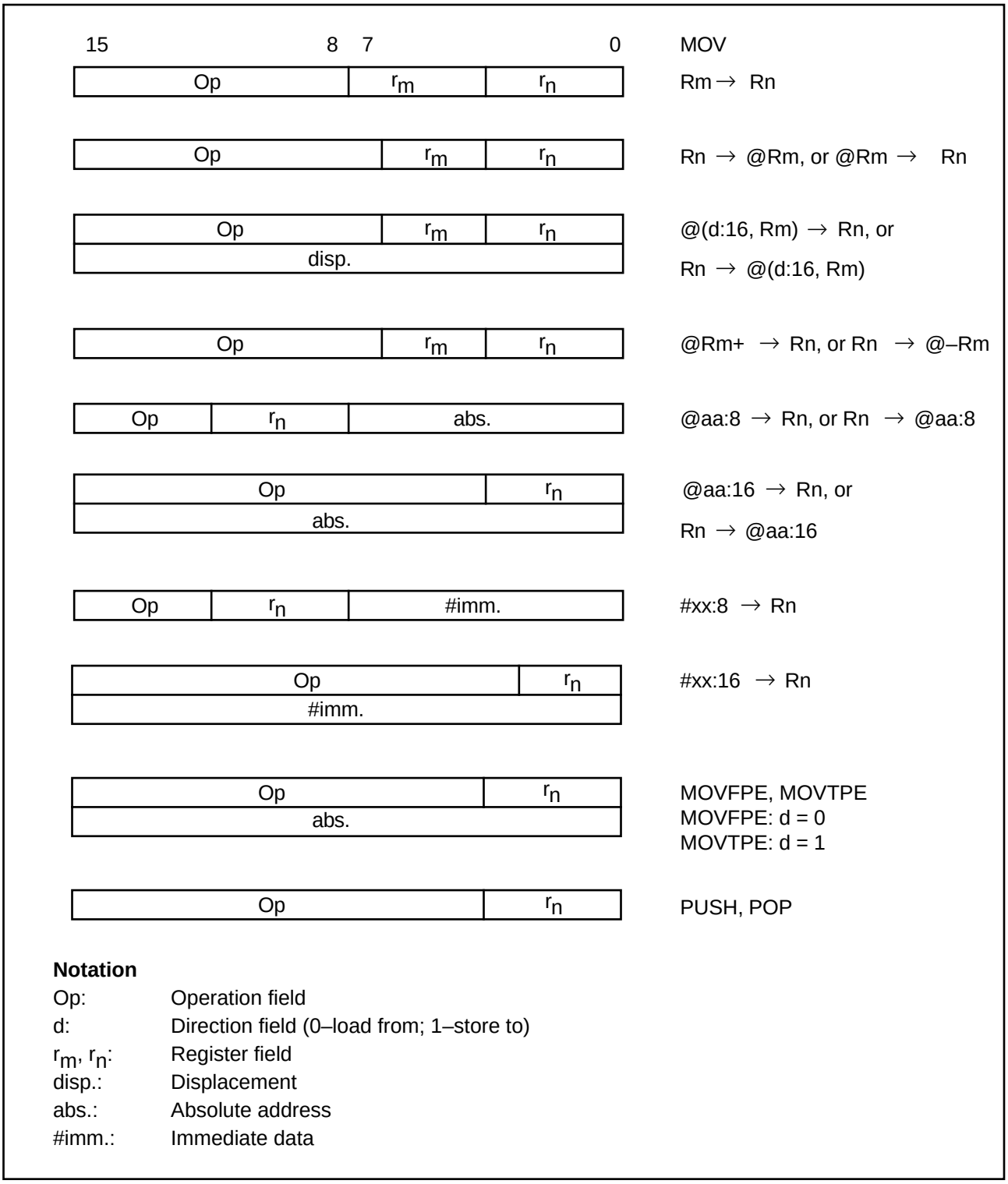


Figure 3-5. Data Transfer Instruction Codes

3.5.2 Arithmetic Operations

Table 3-3 describes the arithmetic instructions. See figure 3-6 in section 3.5.4, “Shift Operations” for their object codes.

Table 3-3. Arithmetic Instructions

Instruction	Size*	Function
ADD SUB	B/W	Rd ± Rs → Rd, Rd + #imm → Rd Performs addition or subtraction on data in two general registers, or addition on immediate data and data in a general register. Immediate data cannot be subtracted from data in a general register. Word data can be added or subtracted only when both words are in general registers.
ADDX SUBX	B	Rd ± Rs ± C → Rd, Rd ± #imm ± C → Rd Performs addition or subtraction with carry or borrow on byte data in two general registers, or addition or subtraction on immediate data and data in a general register.
INC DEC	B	Rd ± #1 → Rd Increments or decrements a general register.
ADDS SUBS	W	Rd ± #imm → Rd Adds or subtracts immediate data to or from data in a general register. The immediate data must be 1 or 2.
DAA DAS	B	Rd decimal adjust → Rd Decimal-adjusts (adjusts to packed BCD) an addition or subtraction result in a general register by referring to the CCR.
MULXU	B	Rd × Rs → Rd Performs 8-bit × 8-bit unsigned multiplication on data in two general registers, providing a 16-bit result.
DIVXU	B	Rd ÷ Rs → Rd Performs 16-bit ÷ 8-bit unsigned division on data in two general registers, providing an 8-bit quotient and 8-bit remainder.
CMP	B/W	Rd – Rs, Rd – #imm Compares data in a general register with data in another general register or with immediate data. Word data can be compared only between two general registers.
NEG	B	0 – Rd → Rd Obtains the two’s complement (arithmetic complement) of data in a general register.

* Size: operand size
B: Byte
W: Word

3.5.3 Logic Operations

Table 3-4 describes the four instructions that perform logic operations. See figure 3-6 in section 3.5.4, “Shift Operations” for their object codes.

Table 3-4. Logic Operation Instructions

Instruction	Size*	Function
AND	B	Rd $\wedge$ Rs $\rightarrow$ Rd, Rd $\wedge$ #imm $\rightarrow$ Rd Performs a logical AND operation on a general register and another general register or immediate data.
OR	B	Rd $\vee$ Rs $\rightarrow$ Rd, Rd $\vee$ #imm $\rightarrow$ Rd Performs a logical OR operation on a general register and another general register or immediate data.
XOR	B	Rd $\oplus$ Rs $\rightarrow$ Rd, Rd $\oplus$ #imm $\rightarrow$ Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data.
NOT	B	$\neg$ (Rd) $\rightarrow$ (Rd) Obtains the one’s complement (logical complement) of general register contents.

3.5.4 Shift Operations

Table 3-5 describes the eight shift instructions. Figure 3-6 shows the object code formats of the arithmetic, logic, and shift instructions.

Table 3-5. Shift Instructions

Instruction	Size*	Function
SHAL SHAR	B	Rd shift $\rightarrow$ Rd Performs an arithmetic shift operation on general register contents.
SHLL SHLR	B	Rd shift $\rightarrow$ Rd Performs a logical shift operation on general register contents.
ROTL ROTR	B	Rd rotate $\rightarrow$ Rd Rotates general register contents.
ROTXL ROTXR	B	Rd rotate through carry $\rightarrow$ Rd Rotates general register contents through the C (carry) bit.

* Size: operand size
B: Byte

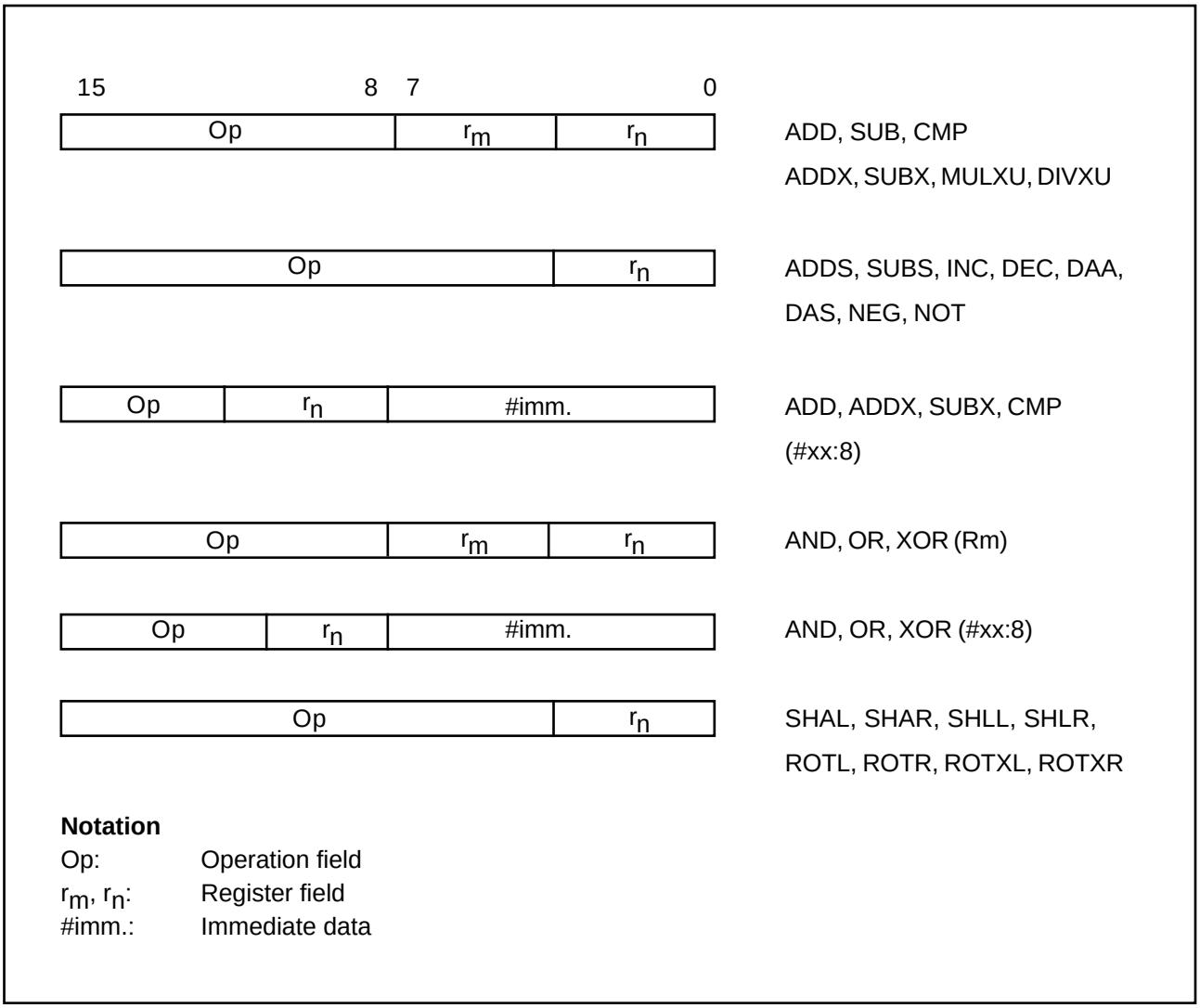


Figure 3-6. Arithmetic, Logic, and Shift Instruction Codes

3.5.5 Bit Manipulations

Table 3-6 describes the bit-manipulation instructions. Figure 3-7 shows their object code formats.

Table 3-6. Bit-Manipulation Instructions (1)

Instruction	Size*	Function
BSET	B	1 → (<bit-No.> of <EAd>) Sets a specified bit in a general register or memory to "1." The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BCLR	B	0 → (<bit-No.> of <EAd>) Clears a specified bit in a general register or memory to "0." The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BNOT	B	¬ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>) Inverts a specified bit in a general register or memory. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register
BTST	B	¬ (<bit-No.> of <EAd>) → Z Tests a specified bit in a general register or memory and sets or clears the Z flag accordingly. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BAND	B	C ∧ (<bit-No.> of <EAd>) → C ANDs the C flag with a specified bit in a general register or memory.
BIAND		C ∧ [¬ (<bit-No.> of <EAd>)] → C ANDs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BOR	B	C ∨ (<bit-No.> of <EAd>) → C ORs the C flag with a specified bit in a general register or memory.
BIOR		C ∨ [¬ (<bit-No.> of <EAd>)] → C ORs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → C XORs the C flag with a specified bit in a general register or memory.

* Size: operand size
B: Byte

Table 3-6. Bit-Manipulation Instructions (2)

Instruction	Size*	Function
BIXOR	B	C ⊕ ¬ [(<bit-No.> of <EAd>)] → C XORs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BLD	B	(<bit-No.> of <EAd>) → C Copies a specified bit in a general register or memory to the C flag.
BILD		¬ (<bit-No.> of <EAd>) → C Copies the inverse of a specified bit in a general register or memory to the C flag. The bit number is specified by 3-bit immediate data.
BST	B	C → (<bit-No.> of <EAd>) Copies the C flag to a specified bit in a general register or memory.
BIST		¬ C → (<bit-No.> of <EAd>) Copies the inverse of the C flag to a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.

* Size: operand size
B: Byte

Notes on Bit Manipulation Instructions: BSET, BCLR, BNOT, BST, and BIST are read-modify-write instructions. They read a byte of data, modify one bit in the byte, then write the byte back. Care is required when these instructions are applied to registers with write-only bits and to the I/O port registers.

Read	Read one data byte at the specified address
Modify	Modify one bit in the data byte
Write	Write the modified data byte back to the specified address

Example 1: BCLR is executed to clear bit 0 in the port 4 data direction register (P4DDR) under the following conditions.

$P4_7$: Input pin, Low, MOS pull-up transistor on
$P4_6$: Input pin, High, MOS pull-up transistor off
$P4_5$ – $P4_0$: Output pins, Low

The intended purpose of this BCLR instruction is to switch $P4_0$ from output to input.

Before Execution of BCLR Instruction

	$P4_7$	**$P4_6$**	**$P4_5$**	**$P4_4$**	**$P4_3$**	**$P4_2$**	**$P4_1$**	**$P4_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low	High	Low	Low	Low	Low	Low	Low
DDR	0	0	1	1	1	1	1	1
DR	1	0	0	0	0	0	0	0
Pull-up Mos	On	Off	Off	Off	Off	Off	Off	Off

Execution of BCLR Instruction

```
BCLR.B  #0, @P4DDR        ;clear bit 0 in data direction register
```

After Execution of BCLR Instruction

	$P4_7$	**$P4_6$**	**$P4_5$**	**$P4_4$**	**$P4_3$**	**$P4_2$**	**$P4_1$**	**$P4_0$**
Input/output	Output	Output	Output	Output	Output	Output	Output	Input
Pin state	Low	High	Low	Low	Low	Low	Low	High
DDR	1	1	1	1	1	1	1	0
DR	1	0	0	0	0	0	0	0
Pull-up Mos	Off	Off	Off	Off	Off	Off	Off	Off

Explanation: To execute the BCLR instruction, the CPU begins by reading P4DDR. Since P4DDR is a write-only register, it is read as H'FF, even though its true value is H'3F.

Next the CPU clears bit 0 of the read data, changing the value to H'FE.

Finally, the CPU writes this value (H'FE) back to P4DDR to complete the BCLR instruction.

As a result, $P4_0$DDR is cleared to "0," making $P4_0$ an input pin. In addition, $P4_7$DDR and $P4_6$DDR are set to "1," making $P4_7$ and $P4_6$ output pins.

Example 2: BSET is executed to set bit 0 in the port 4 data register (P4DR) under the following conditions.

$P4_7$: Input pin, Low, MOS pull-up transistor on

$P4_6$: Input pin, High, MOS pull-up transistor off

$P4_5$ – $P4_0$: Output pins, Low

The intended purpose of this BSET instruction is to switch the output level at $P4_0$ from Low to High.

Before Execution of BSET Instruction

	$P4_7$	**$P4_6$**	**$P4_5$**	**$P4_4$**	**$P4_3$**	**$P4_2$**	**$P4_1$**	**$P4_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low	High	Low	Low	Low	Low	Low	Low
DDR	0	0	1	1	1	1	1	1
DR	1	0	0	0	0	0	0	0
Pull-up Mos	On	Off	Off	Off	Off	Off	Off	Off

Execution of BSET Instruction

```
BSET.B  #0, @PORT4        ;set bit 0 in data register
```

After Execution of BSET Instruction

	$P4_7$	**$P4_6$**	**$P4_5$**	**$P4_4$**	**$P4_3$**	**$P4_2$**	**$P4_1$**	**$P4_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low	High	Low	Low	Low	Low	Low	High
DDR	0	0	1	1	1	1	1	1
DR	0	1	0	0	0	0	0	1
Pull-up	Off	On	Off	Off	Off	Off	Off	Off

Explanation: To execute the BSET instruction, the CPU begins by reading port 4. Since $P4_7$ and $P4_6$ are input pins, the CPU reads the level of these pins directly, not the value in the data register. It reads $P4_7$ as Low ("0") and $P4_6$ as High ("1").

Since $P4_5$ to $P4_0$ are output pins, for these pins the CPU reads the value in the data register ("0"). The CPU therefore reads the value of port 4 as H'40, although the actual value in P4DR is H'80.

Next the CPU sets bit 0 of the read data to "1," changing the value to H'41.

Finally, the CPU writes this value (H'41) back to P4DR to complete the BSET instruction.

As a result, bit $P4_0$ is set to "1," switching pin $P4_0$ to High output. In addition, bits $P4_7$ and $P4_6$ are both modified, changing the on/off settings of the MOS pull-up transistors of pins $P4_7$ and $P4_6$.

Programming Solution: The switching of the pull-ups for $P4_7$ and $P4_6$ in example 2 can be avoided by reserving a byte in RAM as a temporary register for P4DR and using it as follows. RAM0 is a symbol for the user-selected address of the temporary register.

Before Execution of BSET Instruction

```
MOV.B  #80, R0L          ;write data (H'80) for data register
MOV.B  R0L, @RAM0        ;write to DR temporary register (RAM0)
MOV.B  R0L, @PORT4       ;write to DR
```

	P4$_7$	**P4$_6$**	**P4$_5$**	**P4$_4$**	**P4$_3$**	**P4$_2$**	**P4$_1$**	**P4$_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low	High	Low	Low	Low	Low	Low	Low
DDR	0	0	1	1	1	1	1	1
DR	1	0	0	0	0	0	0	0
Pull-up Mos	On	Off	Off	Off	Off	Off	Off	Off
RAM0	1	0	0	0	0	0	0	0

Execution of BSET Instruction

```
BSET.B  #0, @RAM0        ;set bit 0 in DR temporary register (RAM0)
```

After Execution of BSET Instruction

```
MOV.B  @RAM0, R0L        ;obtain value of temporary register RAM0
MOV.B  R0L,   @PORT4     ;write value to DR
```

	P4$_7$	**P4$_6$**	**P4$_5$**	**P4$_4$**	**P4$_3$**	**P4$_2$**	**P4$_1$**	**P4$_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low	High	Low	Low	Low	Low	Low	High
DDR	0	0	1	1	1	1	1	1
DR	1	0	0	0	0	0	0	1
Pull-up Mos	On	Off	Off	Off	Off	Off	Off	Off
RAM0	1	0	0	0	0	0	0	1

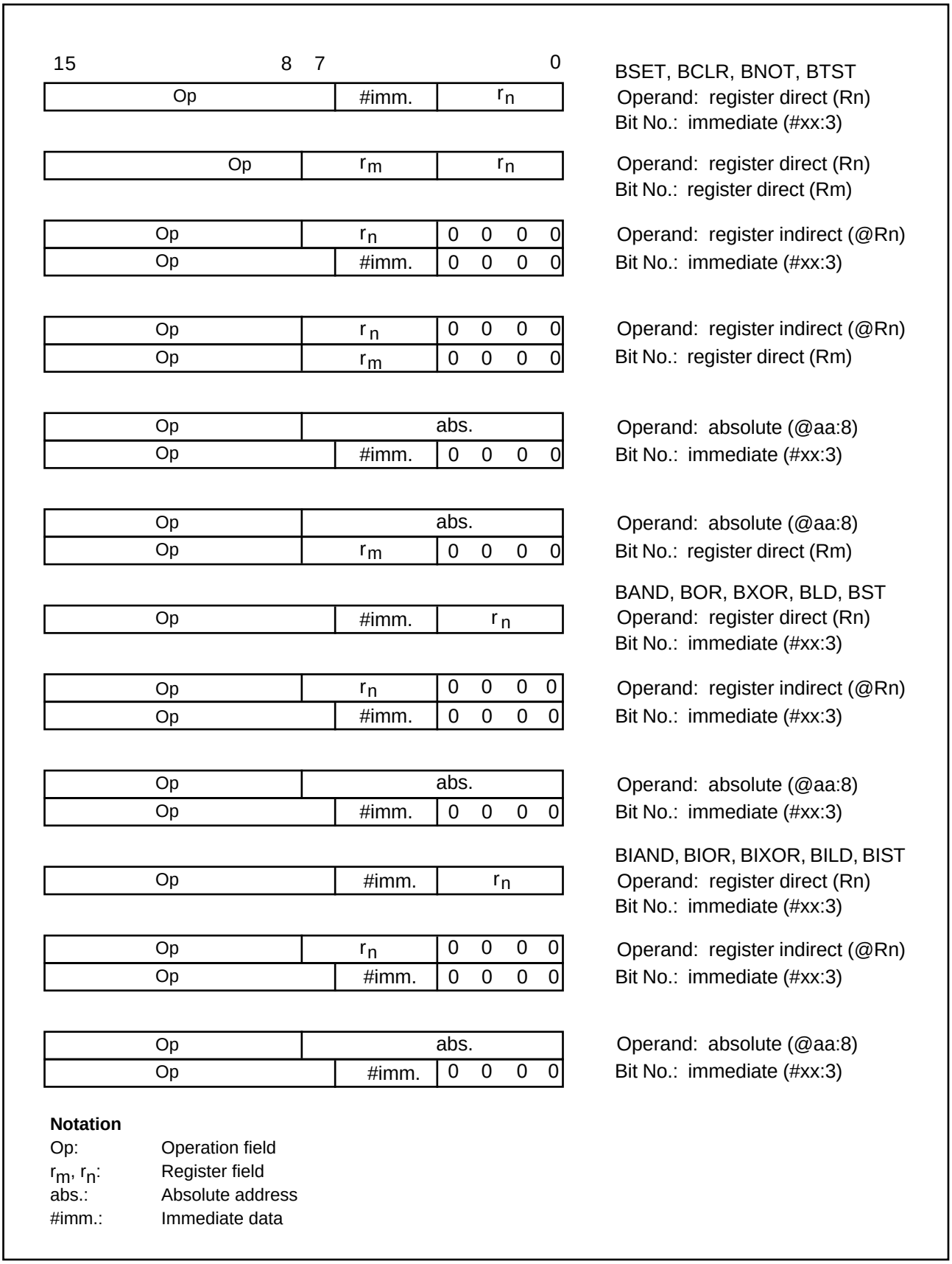


Figure 3-7. Bit Manipulation Instruction Codes

3.5.6 Branching Instructions

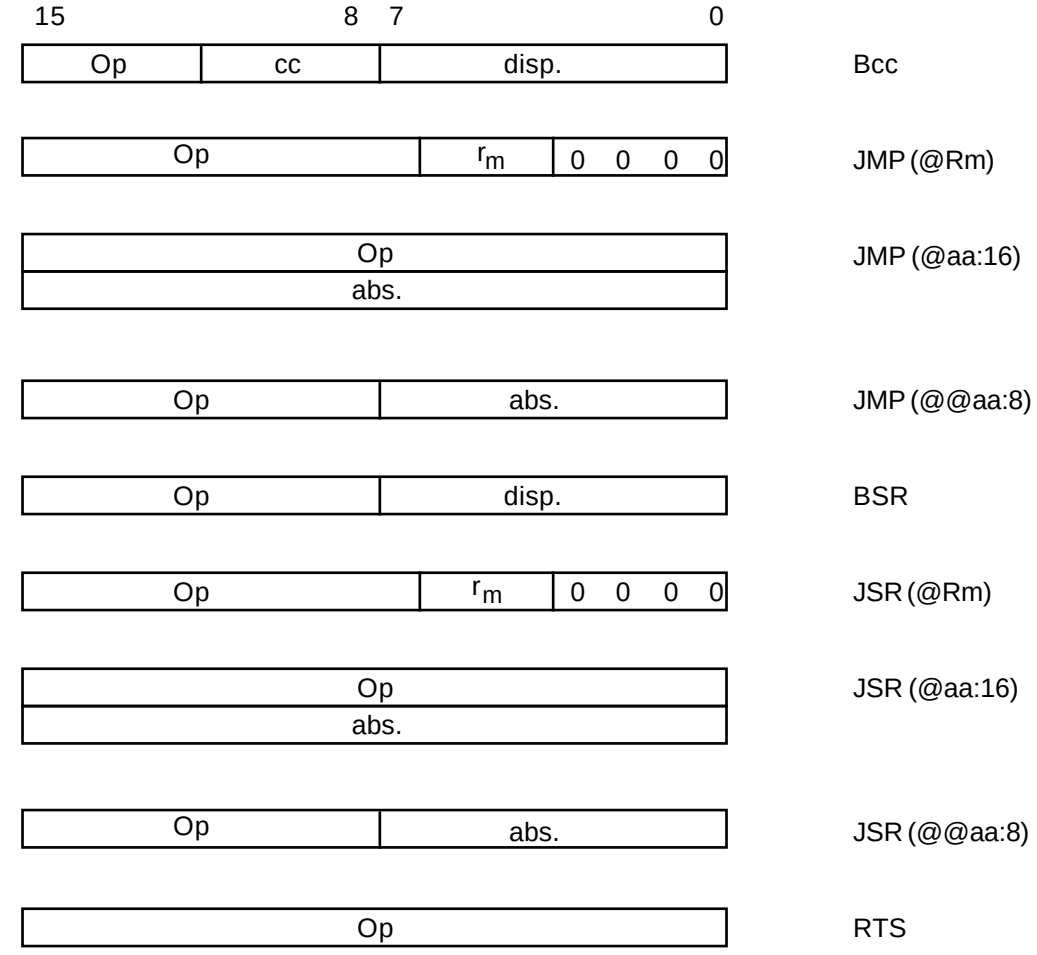
Table 3-7 describes the branching instructions. Figure 3-8 shows their object code formats.

Table 3-7. Branching Instructions

Instruction	Size	Function
Bcc	—	Branches if condition cc is true.

Mnemonic	cc Field	Description	Condition
BRA (BT)	0 0 0 0	Always (True)	Always
BRN (BF)	0 0 0 1	Never (False)	Never
BHI	0 0 1 0	High	$C \vee Z = 0$
BLS	0 0 1 1	Low or Same	$C \vee Z = 1$
BCC (BHS)	0 1 0 0	Carry Clear (High or Same)	$C = 0$
BCS (BLO)	0 1 0 1	Carry Set (Low)	$C = 1$
BNE	0 1 1 0	Not Equal	$Z = 0$
BEQ	0 1 1 1	Equal	$Z = 1$
BVC	1 0 0 0	Overflow Clear	$V = 0$
BVS	1 0 0 1	Overflow Set	$V = 1$
BPL	1 0 1 0	Plus	$N = 0$
BMI	1 0 1 1	Minus	$N = 1$
BGE	1 1 0 0	Greater or Equal	$N \oplus V = 0$
BLT	1 1 0 1	Less Than	$N \oplus V = 1$
BGT	1 1 1 0	Greater Than	$Z \vee (N \oplus V) = 0$
BLE	1 1 1 1	Less or Equal	$Z \vee (N \oplus V) = 1$

JMP	—	Branches unconditionally to a specified address.
JSR	—	Branches to a subroutine at a specified address.
BSR	—	Branches to a subroutine at a specified displacement from the current address.
RTS	—	Returns from a subroutine



Notation

Op:	Operation field
cc:	Condition field
r_m:	Register field
disp.:	Displacement
abs.:	Absolute address

Figure 3-8. Branching Instruction Codes

3.5.7 System Control Instructions

Table 3-8 describes the system control instructions. Figure 3-9 shows their object code formats.

Table 3-8. System Control Instructions

Instruction	Size	Function
RTE	—	Returns from an exception-handling routine.
SLEEP	—	Causes a transition to the power-down state.
LDC	B	Rs → CCR, #imm → CCR Moves immediate data or general register contents to the condition code register.
STC	B	CCR → Rd Copies the condition code register to a specified general register.
ANDC	B	CCR ∧ #imm → CCR Logically ANDs the condition code register with immediate data.
ORC	B	CCR ∨ #imm → CCR Logically ORs the condition code register with immediate data.
XORC	B	CCR ⊕ #imm → CCR Logically exclusive-ORs the condition code register with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

* Size: operand size
B: Byte

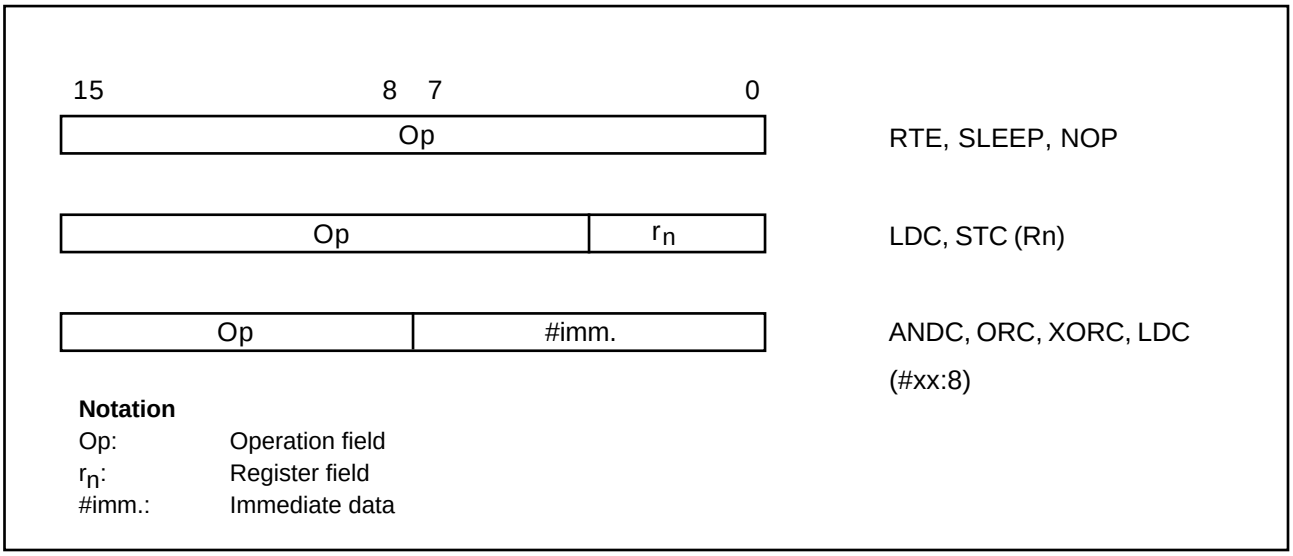


Figure 3-9. System Control Instruction Codes

3.5.8 Block Data Transfer Instruction

In the H8/330 the EEPMOV instruction is a block data transfer instruction. It does not have the EEPROM write function it has in some other chips.

Table 3-9 describes the EEPMOV instruction. Figure 3-10 shows its object code format.

Table 3-9. Block Data Transfer Instruction/EEPROM Write Operation

Instruction	Size	Function
EEPMOV	—	if R4L ≠ 0 then repeat @R5+ → @R6+ R4L – 1 → R4L until R4L = 0 else next; Moves a data block according to parameters set in general registers R4L, R5, and R6. R4L: size of block (bytes) R5: starting source address R6: starting destination address Execution of the next instruction starts as soon as the block transfer is completed.

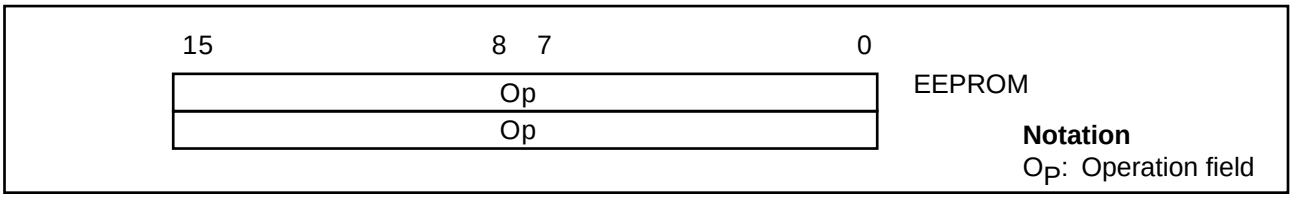
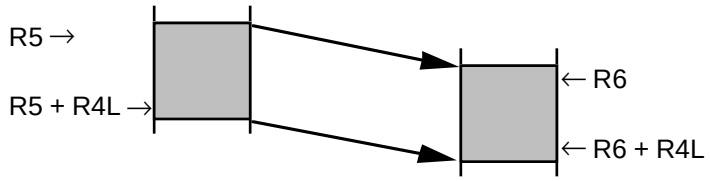


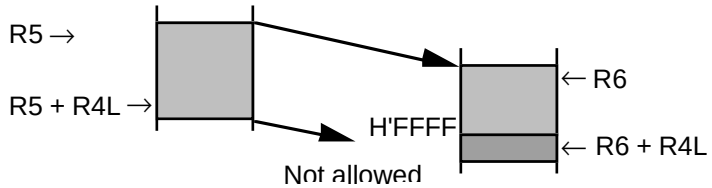
Figure 3-10. Block Data Transfer Instruction/EEPROM Write Operation Code

Notes on EEPMOV Instruction

1. The EEPMOV instruction is a block data transfer instruction. It moves the number of bytes specified by R4L from the address specified by R5 to the address specified by R6.



2. When setting R4L and R6, make sure that the final destination address (R6 + R4L) does not exceed H'FFFF. The value in R6 must not change from H'FFFF to H'0000 during execution of the instruction.



3.6 CPU States

The CPU has three states: the program execution state, exception-handling state, and power-down state. The power-down state is further divided into three modes: the sleep mode, software standby mode, and hardware standby mode. Figure 3-11 summarizes these states, and figure 3-12 shows a map of the state transitions.

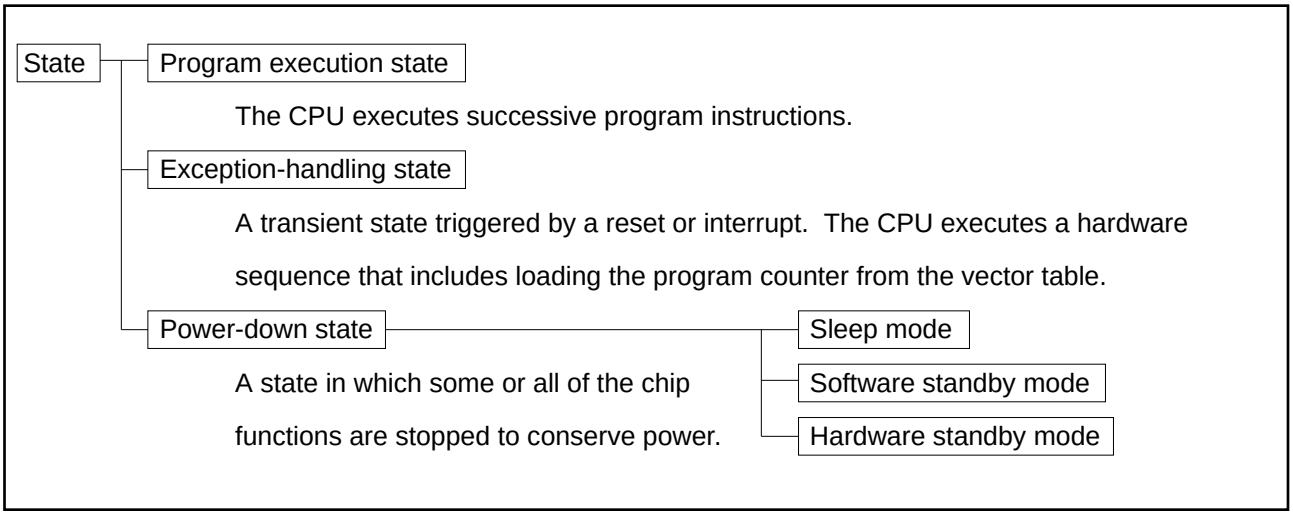
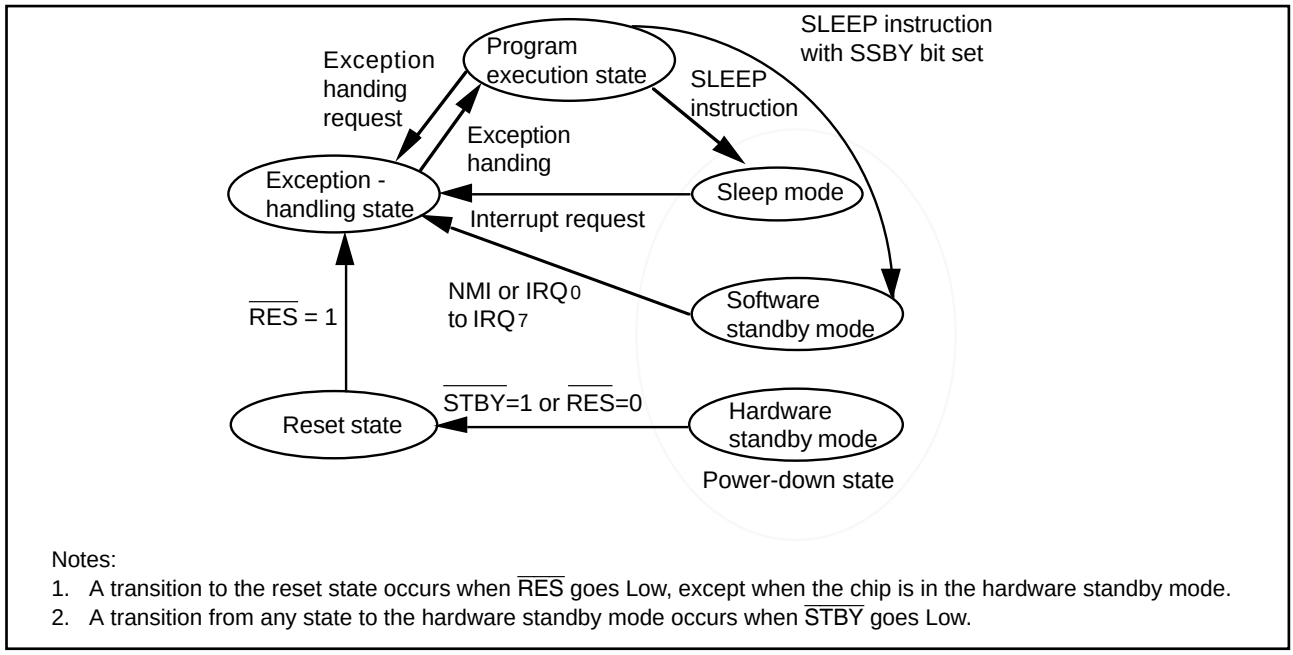


Figure 3-11. Operating States



Notes:
1. A transition to the reset state occurs when $\overline{RES}$ goes Low, except when the chip is in the hardware standby mode.
2. A transition from any state to the hardware standby mode occurs when $\overline{STBY}$ goes Low.

Figure 3-12. State Transitions

3.6.1 Program Execution State

In this state the CPU executes program instructions in sequence. The main program, subroutines, and interrupt-handling routines are all executed in this state.

3.6.2 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU is reset or accepts an interrupt. In this state the CPU carries out a hardware-controlled sequence that prepares it to execute a user-coded exception-handling routine.

In the hardware exception-handling sequence the CPU does the following:

(1) Saves the program counter and condition code register to the stack (except in the case of a reset).
(2) Sets the interrupt mask (I) bit in the condition code register to “1.”
(3) Fetches the start address of the exception-handling routine from the vector table.
(4) Branches to that address, returning to the program execution state.

See section 4, “Exception Handling,” for further information on the exception-handling state.

3.6.3 Power-Down State

The power-down state includes three modes: the sleep mode, the software standby mode, and the hardware standby mode.

(1) Sleep Mode: The sleep mode is entered when a SLEEP instruction is executed. The CPU halts, but CPU register contents remain unchanged and the on-chip supporting modules continue to function.

When an interrupt or reset signal is received, the CPU returns through the exception-handling state to the program execution state.

(2) Software Standby Mode: The software standby mode is entered if the SLEEP instruction is executed while the SSBY (Software Standby) bit in the system control register (SYSCR) is set. The CPU and all on-chip supporting modules halt. The on-chip supporting modules are initialized, but the contents of the on-chip RAM and CPU registers remain unchanged. I/O port outputs also remain unchanged.

(3) Hardware Standby Mode: The hardware standby mode is entered when the input at the $\overline{\text{STBY}}$ pin goes Low. All chip functions halt, including I/O port output. The on-chip supporting modules are initialized, but on-chip RAM contents are held.

See section 14, “Power-Down State” for further information.

3.7 Access Timing and Bus Cycle

The CPU is driven by the system clock (Ø). The period from one rising edge of the system clock to the next is referred to as a “state.”

Memory access is performed in a two-or three-state bus cycle as described below. For more detailed timing diagrams of the bus cycles, see section 17, “Electrical Specifications.”

3.7.1 Access to On-Chip Memory (RAM and ROM)

On-chip ROM and RAM are accessed in a cycle of two states designated T_1 and T_2. Either byte or word data can be accessed, via a 16-bit data bus. Figure 3-13 shows the on-chip memory access cycle. Figure 3-14 shows the associated pin states.

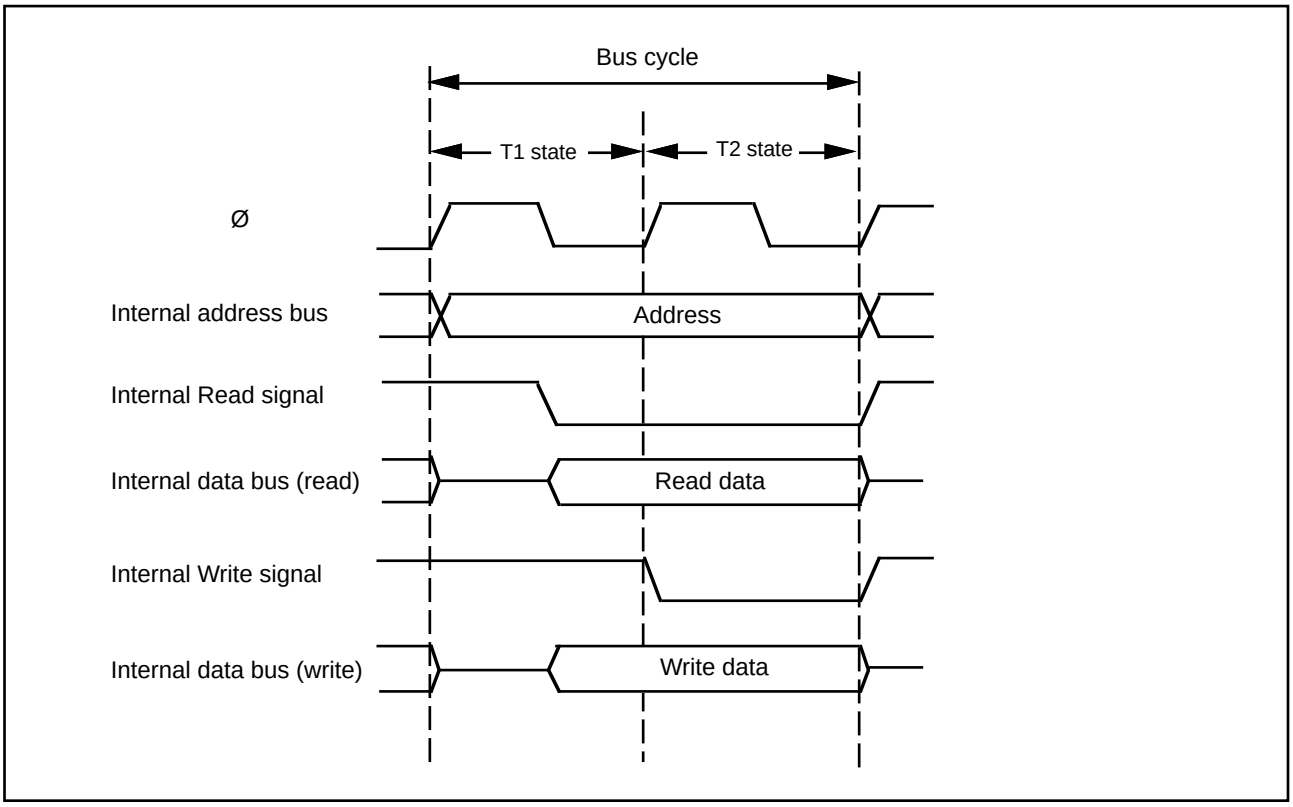


Figure 3-13. On-Chip Memory Access Cycle

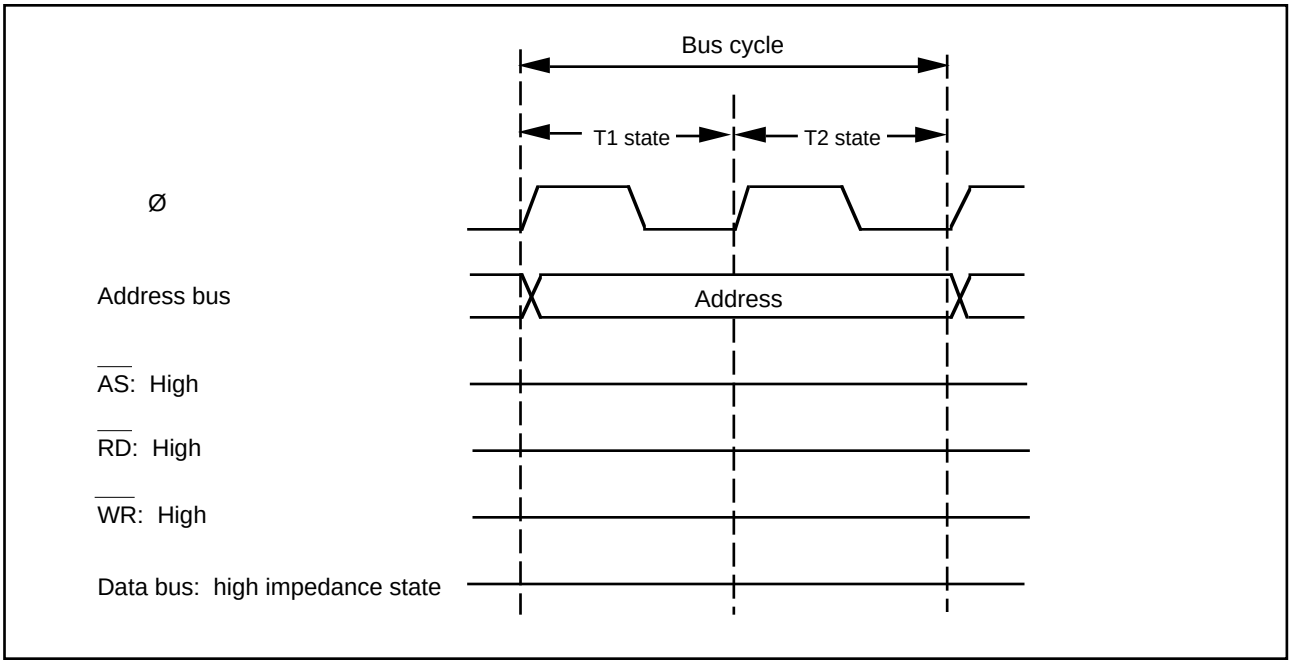


Figure 3-14. Pin States during On-Chip Memory Access Cycle

3.7.2 Access to On-Chip Register Field and External Devices

The on-chip register field (I/O ports, dual-port RAM, on-chip supporting module registers, etc.) and external devices are accessed in a cycle consisting of three states: T_1, T_2, and T_3. Only one byte of data can be accessed per cycle, via an 8-bit data bus. Access to word data or instruction codes requires two consecutive cycles (six states).

Wait States: If requested, additional wait states (T_W) are inserted between T_2 and T_3. The $\overline{\text{WAIT}}$ pin is sampled at the center of state T_2. If it is Low, a wait state is inserted after T_2. The $\overline{\text{WAIT}}$ pin is also sampled at the center of each wait state and if it is still Low, another wait state is inserted. An external device can have any number of wait states inserted by holding $\overline{\text{WAIT}}$ Low for the necessary duration.

The bus cycle for the MOVTPE and MOVFPE instructions will be described in section 15, "E-Clock Interface."

Figure 3-15 shows the access cycle for the on-chip register field. Figure 3-16 shows the associated pin states. Figures 3-17 (a) and (b) show the read and write access timing for external devices.

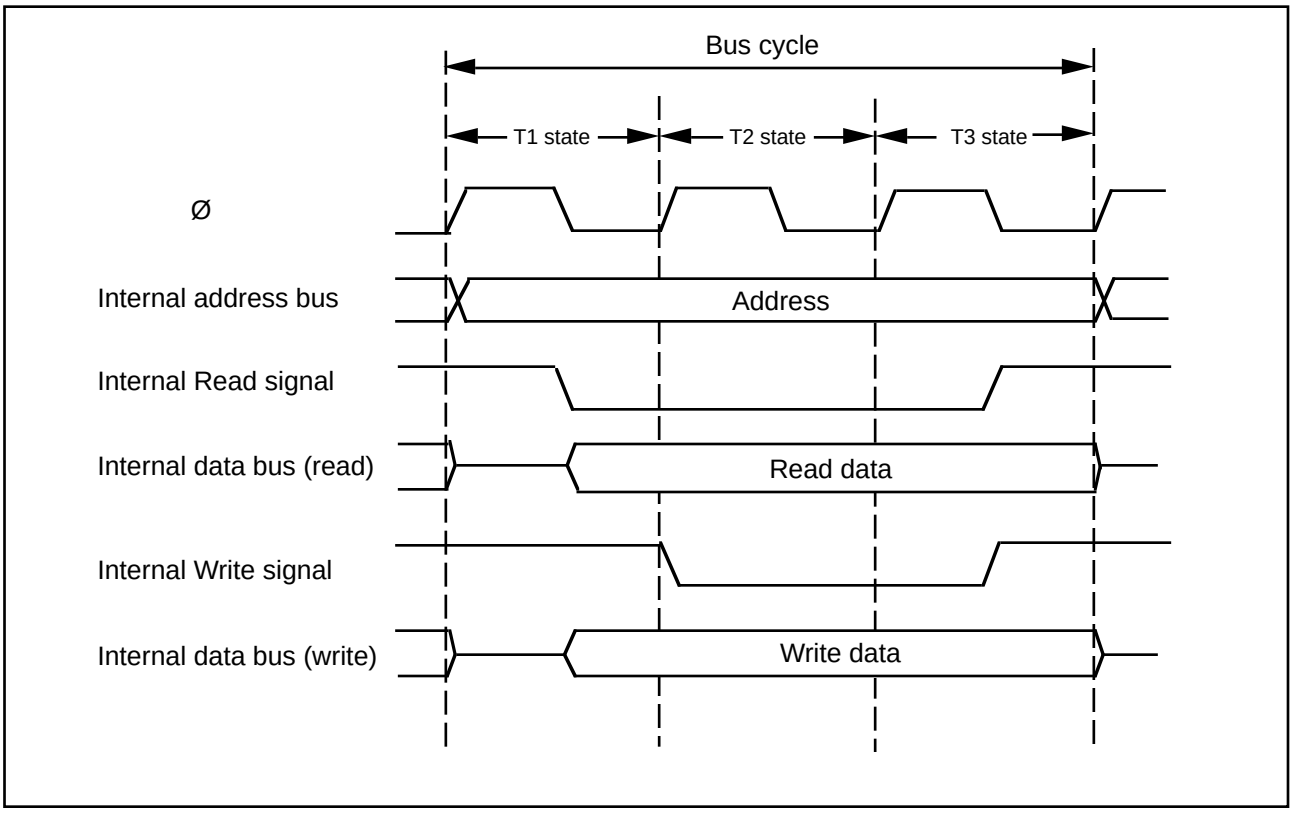


Figure 3-15. On-Chip Register Field Access Cycle

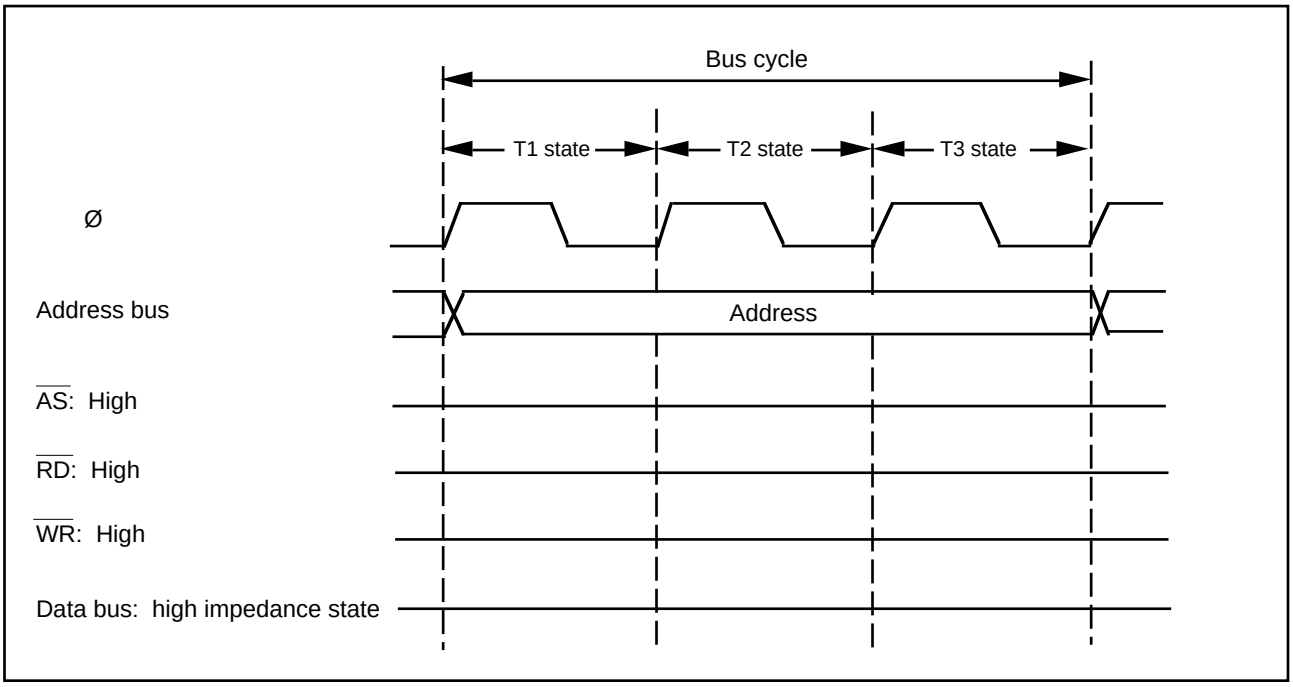


Figure 3-16. Pin States during On-Chip Register Field Access Cycle

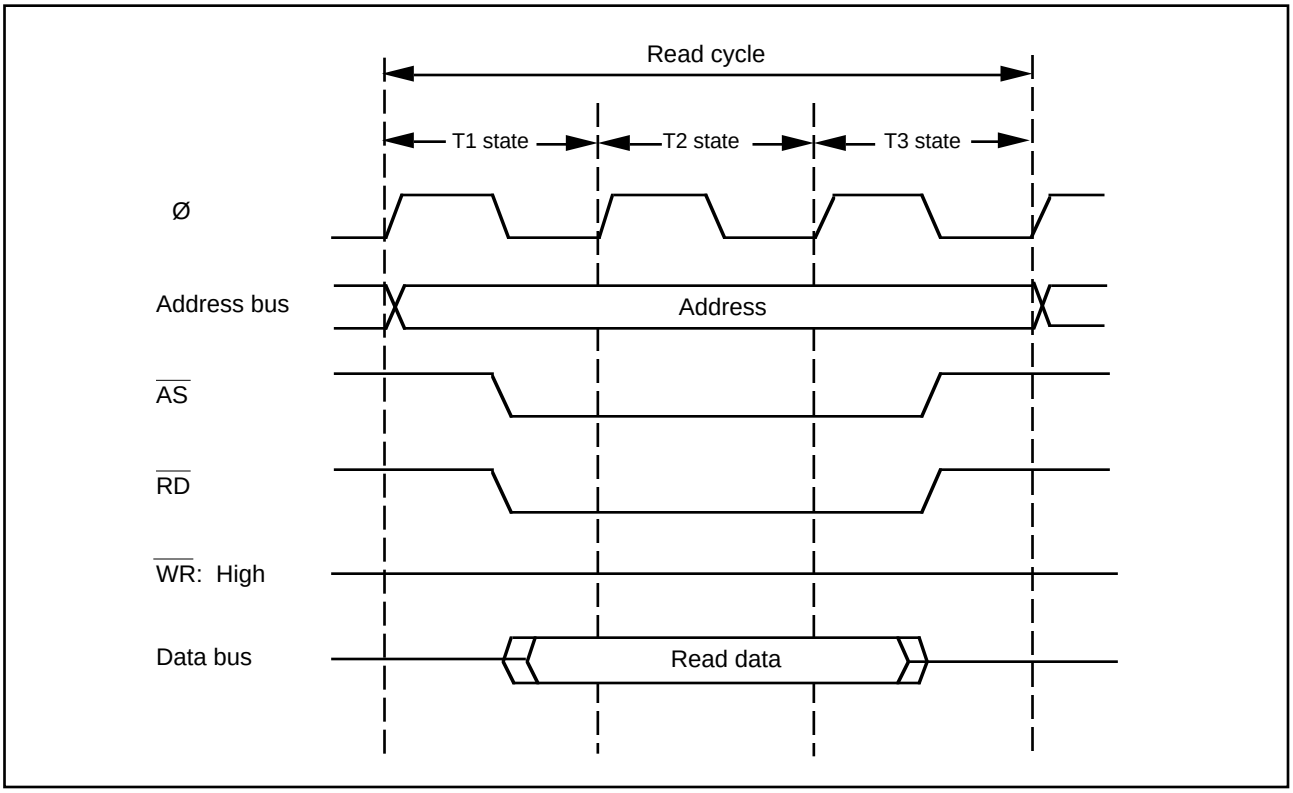


Figure 3-17 (a). External Device Access Timing (read)

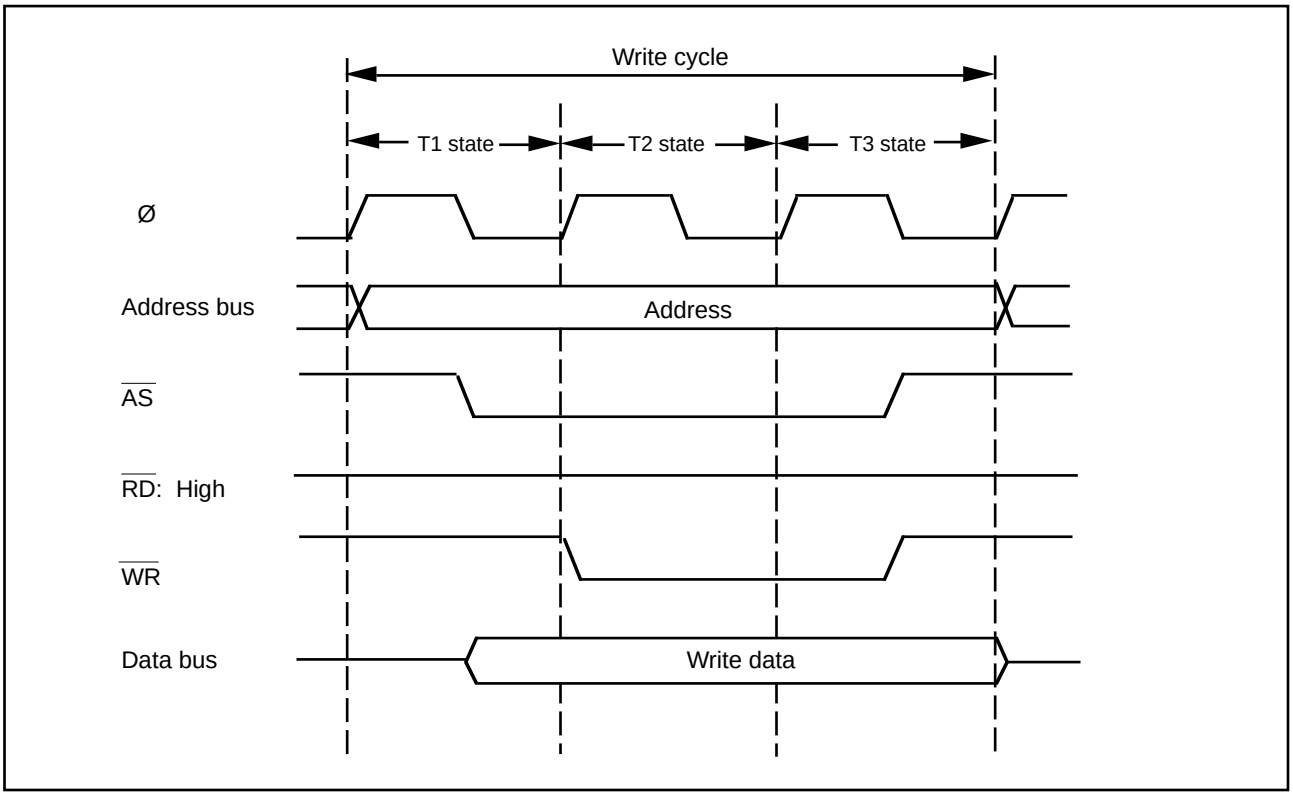


Figure 3-17 (b). External Device Access Timing (write)

Section 4. Exception Handling

As indicated in table 4-1, the H8/330 recognizes only two kinds of exceptions: interrupts (28 sources) and the reset. There are no error or trap exceptions.

When an exception occurs the CPU enters the exception-handling state and performs a hardware exception-handling sequence. There are two exception-handling sequences: one for the reset and one for interrupts. In both sequences the CPU:

- Sets the interrupt mask (I) bit in the CCR to “1,” and
- Loads the program counter (PC) from the vector table.

After the program counter is loaded, the CPU returns to the program execution state and program execution starts from the new PC address.

The vector table occupies addresses H’0000 to H’003D in memory. It consists of word entries giving the addresses of software interrupt-handling routines and the reset routine. The entries are indexed by a vector number associated with the particular exception.

For an interrupt, before the PC and CCR are altered as described above, the old PC and CCR contents are pushed on the stack, so that they can be restored when an RTE (Return from Exception) instruction is executed.

If a reset and interrupt occur simultaneously, the reset has priority. There is also a priority order among different types of interrupts. Table 4-1 compares the reset and interrupt exceptions.

Table 4-1. Reset and Interrupt Exceptions

Item	Reset	Interrupt
Priority	Highest	Lower
Cause	Low $\overline{\text{RES}}$ input	Internal or external interrupt signal
When detected	Any clock period	At end of current instruction, unless current instruction is ANDC, ORC, XORC, or LDC, or at end of hardware interrupt-handling sequence.
When handled	Immediately	At end of current instruction.
Vector numbers	0	3 to 30
Vector table	H’0000 – H’0001	H’0006 – H’003D

4.1 Reset

A reset has the highest exception-handling priority. When the $\overline{\text{RES}}$ pin goes Low, all current processing by the CPU and on-chip supporting modules halts. When $\overline{\text{RES}}$ returns from Low to High, the following hardware reset sequence is executed.

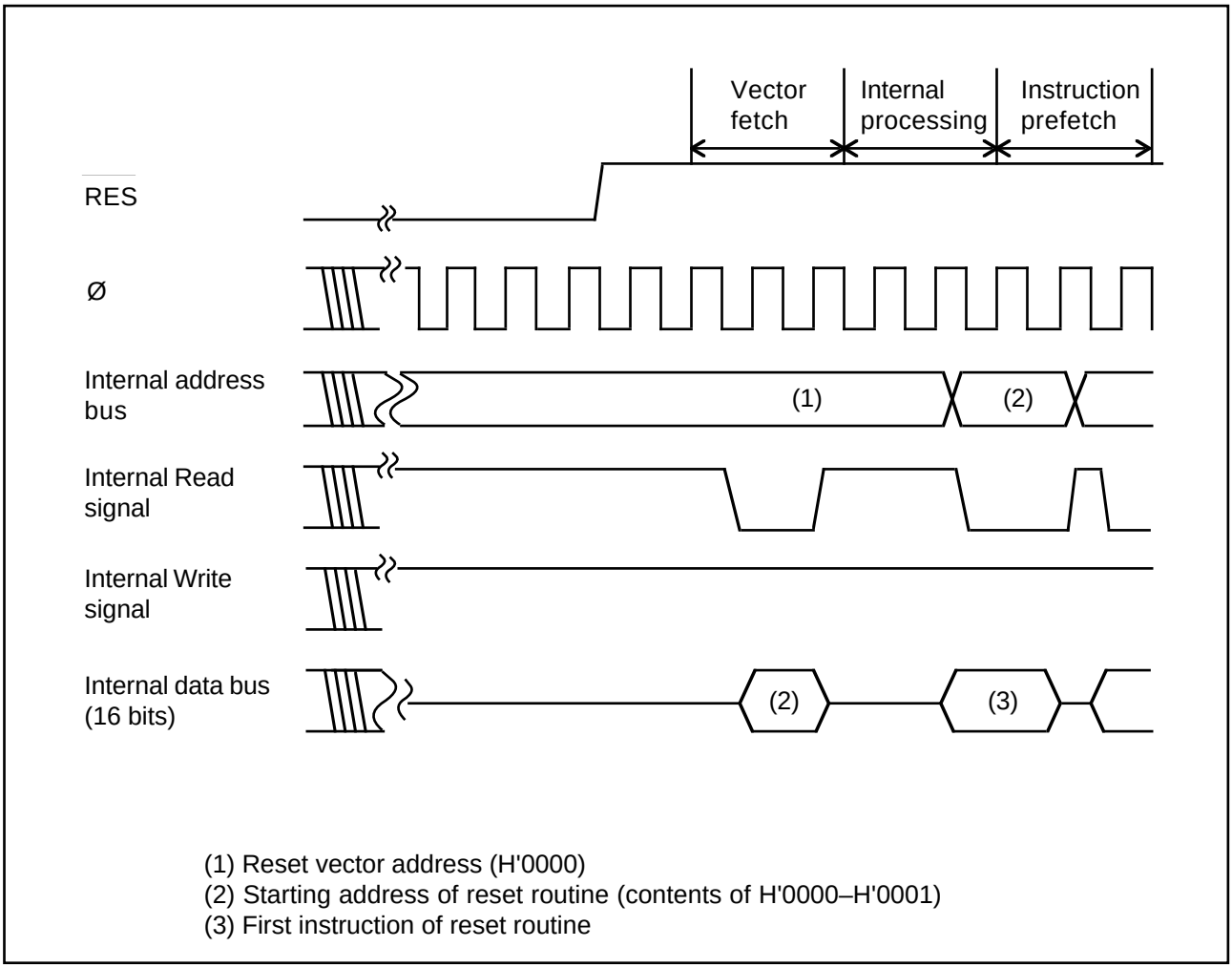
(1) The value at the mode pins (MD_1 and MD_0) is latched in bits MDS1 and MDS0 of the mode register (MDCR).
(2) In the condition code register (CCR), the I bit is set to “1” to mask interrupts.
(3) The registers of the I/O ports and on-chip supporting modules are initialized.
(4) The CPU loads the program counter with the first word in the vector table (stored at addresses H’0000 and H’0001) and starts program execution.

A reset does not initialize the general registers or on-chip RAM.

All interrupts, including NMI, are disabled immediately after a reset. The first program instruction, located at the address specified at the top of the vector table, is therefore always executed. This instruction should be a MOV.W instruction initializing the stack pointer (R7). After execution of this instruction, the NMI interrupt is enabled. Other interrupts remain disabled until their enable bits are set to “1” and the interrupt mask is cleared.

To ensure correct resetting, at power-on the $\overline{\text{RES}}$ pin should be held Low for at least 20ms. In a reset during operation, the $\overline{\text{RES}}$ pin should be held Low for at least 10 system clock periods. The $\overline{\text{RES}}$ pin should also be held Low when power is switched off.

Figure 4-1 indicates the timing of the reset sequence when the vector table and reset routine are located in on-chip ROM. Figure 4-2 indicates the timing when they are in off-chip memory.



(1) Reset vector address (H'0000)
(2) Starting address of reset routine (contents of H'0000–H'0001)
(3) First instruction of reset routine

Figure 4-1. Reset Sequence (Mode 2 or 3, Reset Routine in On-Chip ROM)

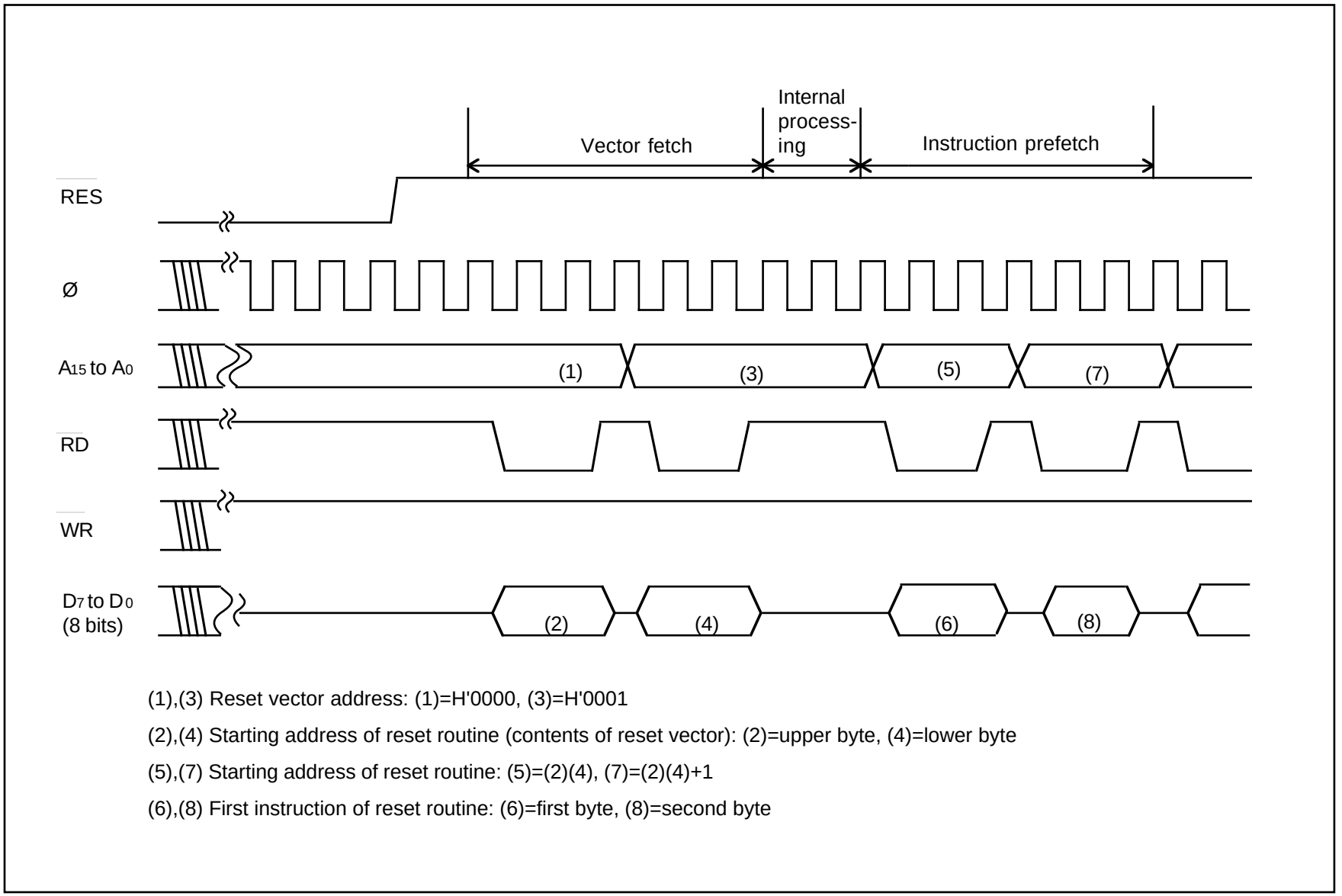


Figure 4-2. Reset Sequence (Mode 1)

4.2 Interrupts

There are nine input pins for external interrupts (NMI, IRQ_0 to IRQ_7). There are also 19 internal interrupts originating in the 16-bit free-running timer (FRT), 8-bit timers (TMR0 and TMR1), serial communication interface (SCI), and A/D converter. The features of these interrupts are:

- All internal and external interrupts except NMI can be masked by the I bit in the CCR.
- IRQ_0 to IRQ_7 can be edge-sensed or level-sensed. (The falling edge or Low level is active.) The type of sensing can be selected for each interrupt individually. NMI is edge-sensed, and either the rising or falling edge can be selected.
- Interrupts are individually vectored. The software interrupt-handling routine does not have to determine what type of interrupt has occurred.

Table 4-2 lists all the interrupts in their order of priority and gives their vector numbers and the addresses of their entries in the vector table.

Table 4-2. Interrupts

Priority	Source	Interrupt	Vector No.	Address of vector
High	External interrupts	NMI	3	H’0006
		IRQ_0	4	H’0008
		IRQ_1	5	H’000A
		IRQ_2	6	H’000C
		IRQ_3	7	H’000E
		IRQ_4	8	H’0010
		IRQ_5	9	H’0012
		IRQ_6	10	H’0014
		IRQ_7	11	H’0016
	Free-running timer	ICIA (Input capture A)	12	H’0018
		ICIB (Input capture B)	13	H’001A
		ICIC (Input capture C)	14	H’001C
		ICID (Input capture D)	15	H’001E
		OCIA (Output compare A)	16	H’0020
		OCIB (Output compare B)	17	H’0022
		FOVI (Overflow)	18	H’0024
	8-Bit timer 0	CMI0A (Compare-match A)	19	H’0026
		CMI0B (Compare-match B)	20	H’0028
		OVI0 (Overflow)	21	H’002A
	8-Bit timer 1	CMI1A (Compare-match A)	22	H’002C
		CMI1B (Compare-match B)	23	H’002E
		OVI1 (Overflow)	24	H’0030
	Dual-port RAM	MREI (Master read end)	25	H’0032
		MWEI (Master write end)	26	H’0034
	Serial communication interface	ERI (Receive error)	27	H’0036
		RXI (Receive end)	28	H’0038
		TXI (Transmit end)	29	H’003A
Low	A/D converter	ADI (Conversion end)	30	H’003C

Notes:

1. H’0000 and H'0001 contain the reset vector.
2. H’0002 to H’0005 are reserved by the H8/330 and are not available to the user.

Figure 4-3 shows a block diagram of the interrupt controller. Figure 4-4 is a flowchart showing the operation of the interrupt controller and the sequence by which an interrupt is accepted. This sequence is outlined below.

(1) The interrupt controller receives an interrupt request signal. Interrupt request signals can be generated by:
   — A High-to-Low (or Low-to-High) transition of the $\overline{\text{NMI}}$ signal
   — A Low input (or High-to-Low transition) of one of the $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_7$ signals
   — An on-chip supporting module

   All interrupts except $\overline{\text{NMI}}$ have enable bits. The interrupt can be requested only when its enable bit is set to "1."

(2) When notified of an interrupt, the interrupt controller scans the interrupt signals in priority order and selects the one with the highest priority. (See table 4-2 for the priority order.) Other requested interrupts remain pending.

(3) The interrupt controller accepts the interrupt if it is an NMI, or if it is another interrupt and the I bit in the CCR is cleared to “0.” If the interrupt is not an NMI and the I bit is set to “1,” the interrupt is held pending.

(4) When an interrupt is accepted, after completion of the current instruction, the CPU pushes first the PC then the CCR onto the stack. The stacked PC indicates the address of the first instruction that will be executed after the return. The stack pointer (R7) must indicate an even address. See section 4.2.5, “Note on Stack Handling” for details.

(5) The CPU sets the I bit in the CCR to “1,” masking all further interrupts except NMI during the interrupt-handling routine.

(6) The CPU generates the vector address of the interrupt and loads the word at this address into the program counter.

(7) Execution of the software interrupt-handling routine starts from the address now in the program counter.

(8) On the return from the interrupt-handling routine (RTE instruction), the CCR and PC are popped from the stack and execution of the interrupted program resumes.

The timing of this sequence is shown in Figure 4-5 for the case in which the program and vector table are in on-chip ROM and the stack is in on-chip RAM.

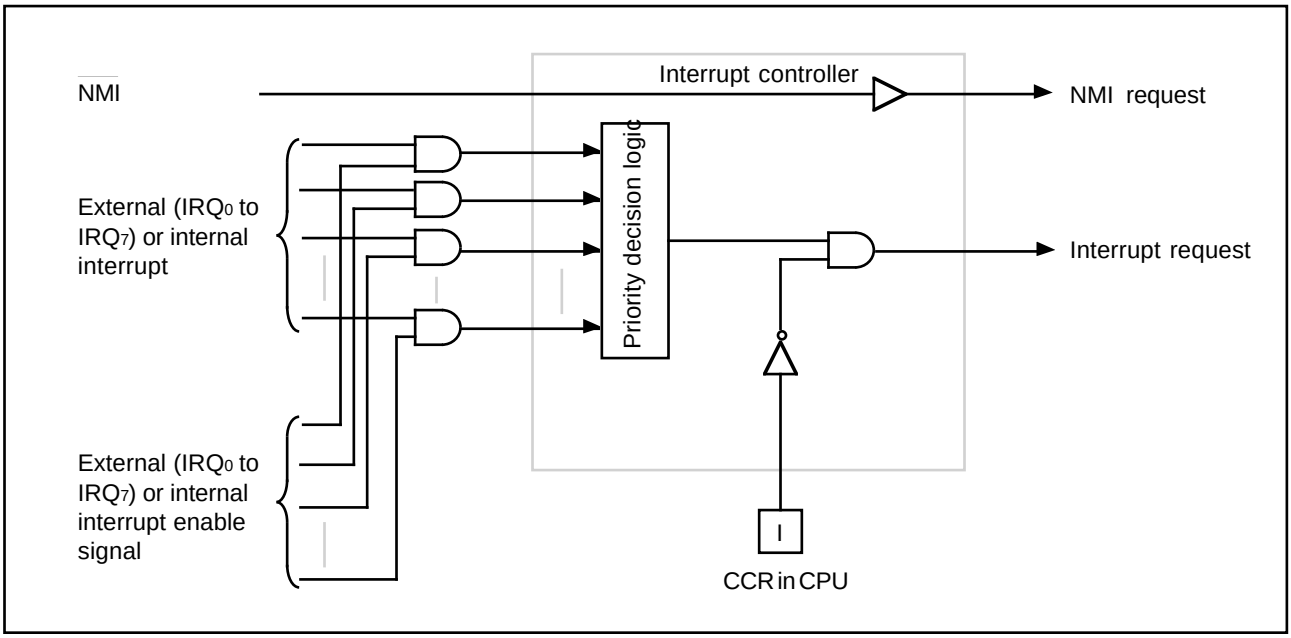


Figure 4-3. Block Diagram of Interrupt Controller

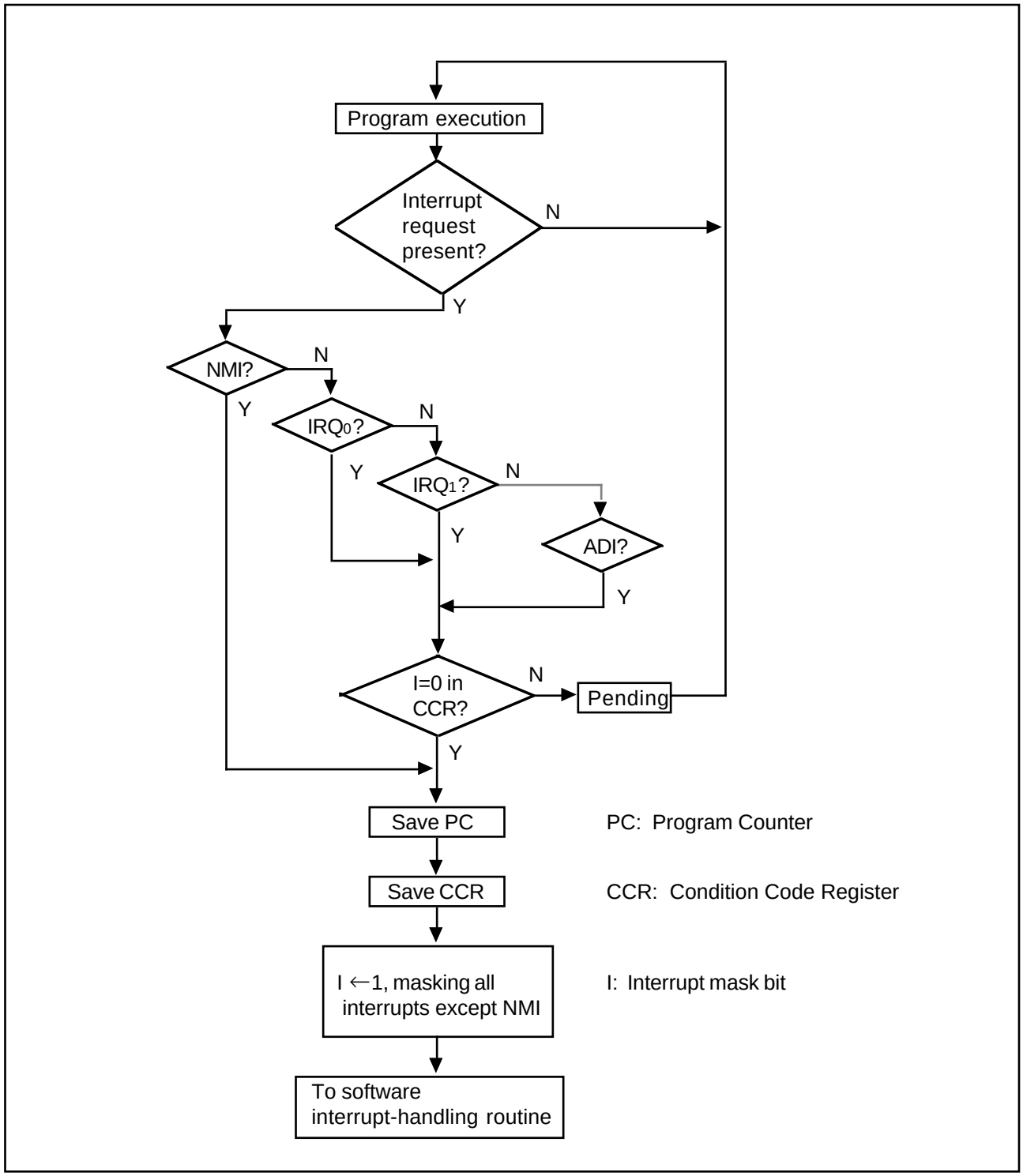
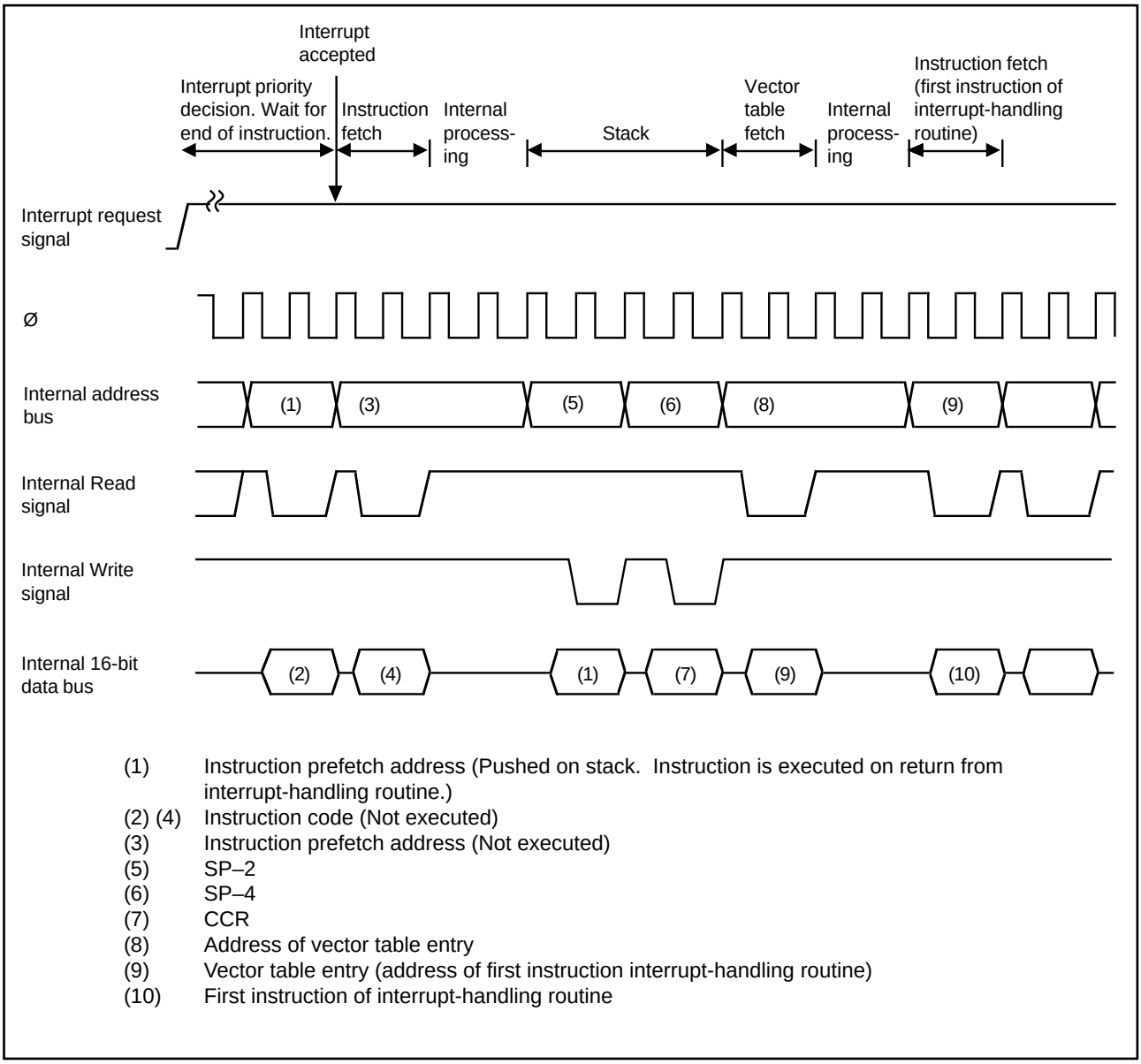


Figure 4-4. Hardware Interrupt-Handling Sequence



(1) Instruction prefetch address (Pushed on stack. Instruction is executed on return from interrupt-handling routine.)
(2) (4) Instruction code (Not executed)
(3) Instruction prefetch address (Not executed)
(5) SP–2
(6) SP–4
(7) CCR
(8) Address of vector table entry
(9) Vector table entry (address of first instruction interrupt-handling routine)
(10) First instruction of interrupt-handling routine

Figure 4-5. Timing of Interrupt Sequence

4.2.1 Interrupt-Related Registers

The interrupt controller refers to three registers in addition to the CCR. The names and attributes of these registers are listed in Table 4-3.

Table 4-3. Registers Read by Interrupt Controller

Name	Abbreviation	Read/write	Address
System control register	SYSCR	R/W	H'FFC4
IRQ sense control register	ISCR	R/W	H'FFC6
IRQ enable register	IER	R/W	H'FFC7

(1) System Control Register (SYSCR)—H'FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W

The first four bits of the system control register concern the software standby mode, and the last two bits enable the on-chip RAM and dual-port RAM. Bit 2 is the only bit read by the interrupt controller.

Bit 2—Nonmaskable Interrupt Edge (NMIEG): This bit determines whether a nonmaskable interrupt is generated on the falling or rising edge of the $\overline{\text{NMI}}$ input signal.

Bit 2

NMIEG	Description	
0	An interrupt is generated on the falling edge of $\overline{\text{NMI}}$.	(Initial state)
1	An interrupt is generated on the rising edge of $\overline{\text{NMI}}$.	

(2) IRQ Sense Control Register (ISCR)—H'FFC6

Bit	7	6	5	4	3	2	1	0
	IRQ_7SC	IRQ_6SC	IRQ_5SC	IRQ_4SC	IRQ_3SC	IRQ_2SC	IRQ_1SC	IRQ_0SC
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 0 to 7 – IRQ_0 to IRQ_7 Sense Control (IRQ_0SC to IRQ_7SC): These bits determine whether the $\overline{IRQ}_0$ to $\overline{IRQ}_7$ inputs are edge-sensed or level-sensed.

Bit i IRQiSC	Description	
0	$\overline{IRQ}$i is level-sensed.	(Initial state)
1	$\overline{IRQ}$i is sensed on the falling edge.	

Edge-sensed interrupt signals are latched (if enabled) until the interrupt is serviced. They are latched even if the interrupt mask bit (I) is set in the CCR, and remain latched even if the enable bit (IRQ_0E to IRQ_7E) is later cleared to 0.

(3) IRQ Enable Register (IER)—H'FFC7

Bit	7	6	5	4	3	2	1	0
	IRQ_7E	IRQ_6E	IRQ_5E	IRQ_4E	IRQ_3E	IRQ_2E	IRQ_1E	IRQ_0E
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 0 to 7 – IRQ_0 to IRQ_7 Enable (IRQ_0E to IRQ_7E): These bits enable or disable the IRQi signals individually.

After a reset, all IRQi interrupts are disabled (as well as masked).

Bit i IRQiE	Description	
0	IRQi is disabled.	(Initial state)
1	IRQi is enabled.	

4.2.2 External Interrupts

The external interrupts are NMI and IRQ_0 to IRQ_7.

(1) NMI: A nonmaskable interrupt is generated on the rising or falling edge of the $\overline{NMI}$ input signal regardless of whether the I (interrupt mask) bit is set in the CCR. The valid edge is selected by the NMIEG bit in the system control register.

An NMI has highest priority and is always accepted as soon as the current instruction ends, unless the current instruction is an ANDC, ORC, XORC, or LDC instruction. When an NMI interrupt is

accepted the interrupt mask (I bit) is set, so the NMI handling routine cannot be interrupted except by another NMI.
The NMI vector number is 3. Its entry is located at address H'0006 in the vector table.

(2) IRQ_0 to IRQ_7: These interrupt signals are level-sensed or sensed on the falling edge of the input, as selected by the bits in the ISCR. These interrupts can be masked collectively by the I bit in the CCR, and can be enabled and disabled individually by setting and clearing the bits in the IER. When one of these interrupts is accepted, the I bit is set to "1" to mask further interrupts (except $\overline{NMI}$).

The interrupt controller reads level-sensed signals directly from the input pin, so the signal must be held Low until the interrupt is accepted.

Edge-sensed signals are latched in a flip-flop in the interrupt controller. The signal is latched only if the interrupt is enabled in the IRQ enable register. However, the signal is latched even if the interrupt is masked (I bit set to "1" in the CCR).

These interrupts are second in priority to NMI. Among them, IRQ_0 has the highest priority and IRQ_7 the lowest priority.

Interrupts IRQ_0 to IRQ_7 occur regardless of whether the $\overline{IRQ}_0$ to $\overline{IRQ}_7$ lines are used for input or output. When IRQ_0 to IRQ_7 are requested by external signals, clear the corresponding bits in the port data direction register (DDR) to 0, and do not use the same pins for timer or serial communication interface input or output.

4.2.3 Internal Interrupts

Nineteen internal interrupts can be requested by the on-chip supporting modules. All of them are masked when the I bit in the CCR is set. In addition, they can all be enabled or disabled by bits in the control registers of the on-chip supporting modules. When one of these interrupts is accepted, the I bit is set to "1" to mask further interrupts (except $\overline{NMI}$).

Power can be conserved by waiting for an internal interrupt in the sleep mode, in which the CPU halts but the on-chip supporting modules continue to run. When the interrupt arrives, the CPU returns to the program-execution state, services the interrupt, then resumes execution of the main program. See section 14, "Power-Down State" for further information on the sleep mode.

The internal interrupt signals received by the interrupt controller are generated from flag bits in the

registers of the on-chip supporting modules. The interrupt controller does not reset these flag bits when accepting the interrupt. The flag bit must be reset by the software interrupt-handling routine. To reset an interrupt flag, software must read the relevant bit or register, then clear the flag bit to "0." The flag bit cannot be cleared unless it is first read. The following is a coding example that clears the A/D interrupt flag (ADF bit) in the A/D control/status register.

```
BCLR #7, @H'FFE8
```

Note: When disabling internal interrupts, note the following points.

1. Set the interrupt mask (I) to "1" before disabling an internal interrupt (an interrupt from an on-chip supporting module) or clearing an interrupt flag.
2. If an instruction that disables an internal interrupt is executed while the interrupt mask (I) is cleared to "0", and the interrupt is requested during execution of the instruction, the CPU resolves this conflict as follows:
   (1) If one or more other interrupts are also requested, the other interrupt with the highest priority is serviced.
   (2) If no other interrupt is requested, the CPU branches to the reset address.

Example: The following coding disables the output compare A interrupt from the free-running timer module in the H8/330 by clearing the OCIAE bit. The I bit is first set to "1."

```
ORC     #80, CCR        ; Set I bit
BCLR    #3, @TIER       ; Disable output compare A interrupt
ANDC    #7F, CCR        ; Clear I bit
```

Note: Interrupt requests are not detected immediately after the ANDC, ORC, XORC, and LDC instructions.

For the priority order of these interrupts, see table 4-2.

4.2.4 Interrupt Response Time

Table 4-4 indicates the time that elapses from an interrupt request signal until the first instruction of the software interrupt-handling routine is executed. Since the H8/330 accesses its on-chip memory 16 bits at a time, very fast interrupt service can be obtained by placing interrupt-handling routines in on-chip ROM and the stack in on-chip RAM.

Table 4-4. Number of States before Interrupt Service

		Number of states	
No.	**Reason for wait**	**On-chip memory**	**External memory**
1	Interrupt priority decision	2 (note 3)	2 (note 3)
2	Wait for completion of current instruction (note 1)	1 to 13	5 to 17 (note 2)
3	Save PC and CCR	4	12 (note 2)
4	Fetch vector	2	6 (note 2)
5	Fetch instruction	4	12 (note 2)
6	Internal processing	4	4
	Total	17 to 29	41 to 53 (note 2)

Notes:
1. These values do not apply if the current instruction is an EEPMOV, MOVFPE, or MOVTPE instruction.
2. If wait states are inserted in external memory access, these values may be longer.
3. 1 for internal interrupts.

4.2.5 Note on Stack Handling

When the H8/330 performs word access, the least significant bit of the address is always assumed to be “0.” If an odd address is specified, no address error occurs, but the intended address is not accessed.

The stack is always accessed by word access. Care should be taken to keep an even value in the stack pointer (general register R7). The PUSH and POP (or MOV.W Rn, @–SP and MOV.W @SP+, Rn) instructions should be used for pushing and popping registers on the stack. The MOV.B Rn, @–SP and MOV.B @SP+, Rn instructions should never be used; they can easily cause programs to crash.

Figure 4-6 shows how the PC and CCR are pushed on the stack during the hardware interrupt-handling sequence. The CCR is saved as a word consisting of two identical bytes, both containing the CCR value. On return from the interrupt-handling routine, the CCR is popped from the upper of these two bytes. The lower byte is ignored.

Figure 4-7 shows an example of damage caused when the stack pointer contains an odd address.

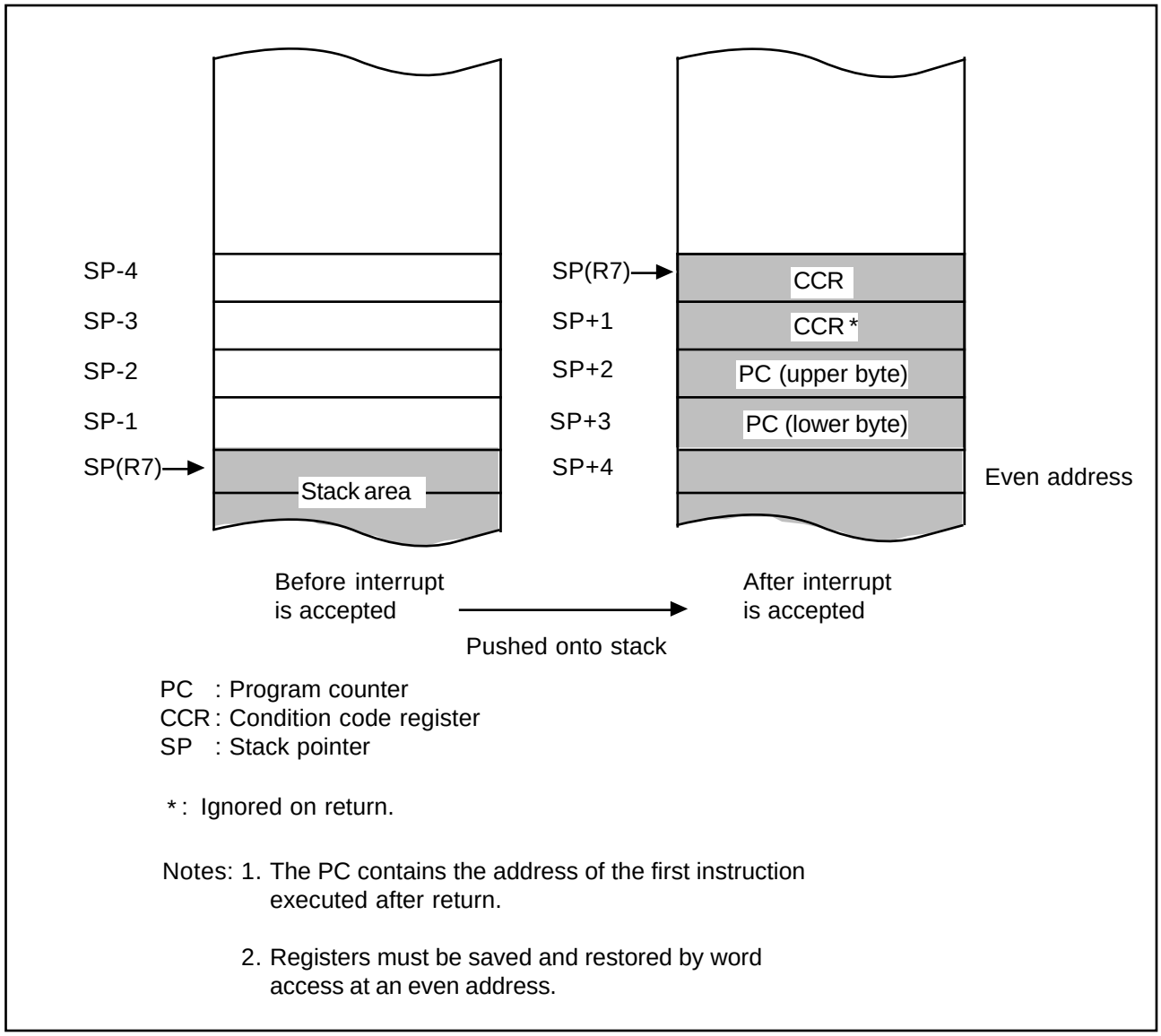


Figure 4-6. Usage of Stack in Interrupt Handling

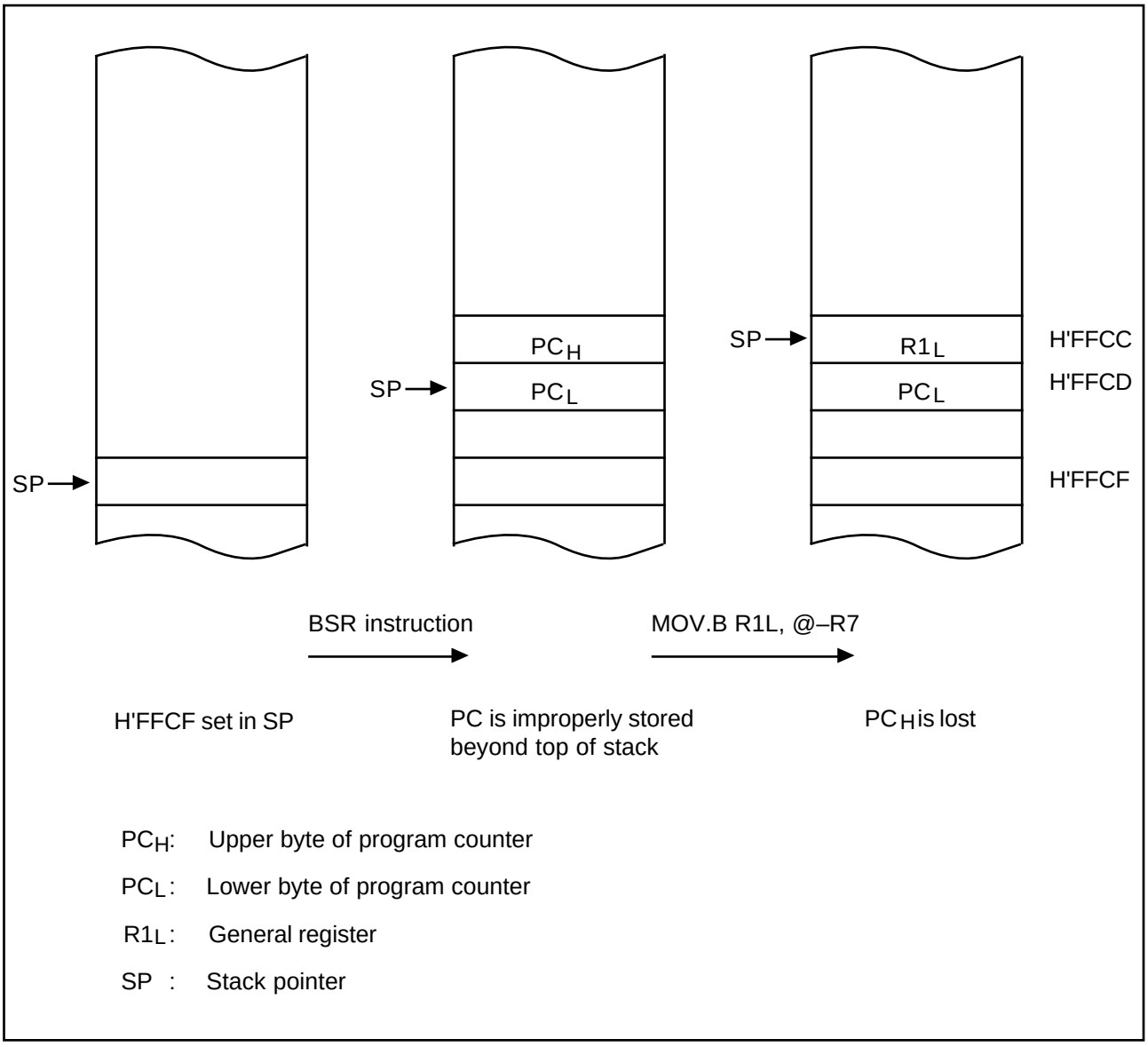


Figure 4-7. Example of Damage Caused by Setting an Odd Address in R7

4.2.6 Deferring of Interrupts

As noted previously, no interrupt is accepted immediately after a reset. System control instructions that rewrite the CCR have a similar effect. Interrupts requests received during one of these instructions are deferred until at least one more instruction has been executed.

The instructions that defer interrupts in this way are XORC, ORC, ANDC, and LDC. At the completion of these instructions the interrupt controller does not check the interrupt signals. The CPU therefore always proceeds to the next instruction. (And if the next instruction is one of these four, the CPU also proceeds to the next instruction after that.) The interrupt-handling sequence starts after the next instruction that is not one of these four has been executed. Figure 4-8 shows an example.

NMI and other edge-sensed interrupt request signals that arrive during the execution of an ANDC, ORC, XORC, or LDC instruction are not lost. The request is latched in the interrupt controller and detected after another instruction has been executed.

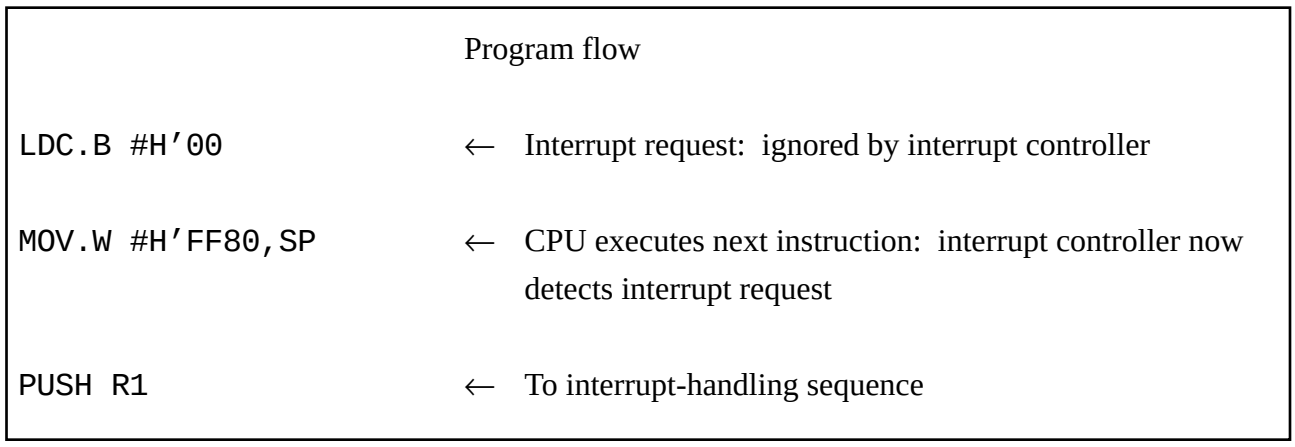


Figure 4-8. Example of Deferred Interrupt

Section 5. I/O Ports

5.1 Overview

The H8/330 has nine parallel I/O ports, including:

- Six 8-bit input/output ports—ports 1, 2, 3, 4, 6, and 9
- One 8-bit input port—port 7
- One 7-bit input/output port—port 8
- One 3-bit input/output port—port 5

All ports except port 7 have programmable MOS input pull-ups. Ports 1 and 2 can drive LEDs.

Input and output are memory-mapped. The CPU views each port as a data register (DR) located in the register field at the high end of the address space. Each port (except port 7) also has a data direction register (DDR) which determines which pins are used for input and which for output.

Output: To send data to an output port, the CPU selects output in the data direction register and writes the desired data in the data register, causing the data to be held in a latch. The latch output drives the pin through a buffer amplifier. If the CPU reads the data register of an output port, it obtains the data held in the latch rather than the actual level of the pin.

Input: To read data from an I/O port, the CPU selects input in the data direction register and reads the data register. This causes the input logic level at the pin to be placed directly on the internal data bus. There is no intervening input latch.

MOS Pull-Up: The MOS pull-ups for input pins are controlled as follows. To turn on the pull-up transistor for a pin, software must first clear its data direction bit to “0” to make the pin an input pin, then write a “1” in the data bit for that pin. The pull-up can be turned off by writing a “0” in the data bit, or a “1” in the data direction bit. The pull-ups are also turned off by a reset and by entry to the hardware standby mode.

The data direction registers are write-only registers; their contents are invisible to the CPU. If the CPU reads a data direction register all bits are read as “1,” regardless of their true values. Care is required if bit manipulation instructions are used to set and clear the data direction bits. See the note on bit manipulation instructions in Section 3.5.5, "Bit Manipulations."

Auxiliary Functions: In addition to their general-purpose input/output functions, all of the I/O ports have auxiliary functions. Most of the auxiliary functions are software-selectable and must be enabled by setting bits in control registers. When selected, an auxiliary function usually replaces

the general-purpose input/output function, but in some cases both functions can operate simultaneously. Table 5-1 summarizes the auxiliary functions of the ports.

Table 5-1. Auxiliary Functions of Input/Output Ports

I/O port	Expanded modes	Single-chip mode
Port 1	Address bus (Low)*1	—
Port 2	Address bus (High)*1	—
Port 3	Data bus*2	Dual-port RAM data bus
Port 4	8-Bit and PWM timer input and output	
Port 5	Serial communication (asynchronous mode)	
Port 6	Free-running timer input/output, IRQ_6 and IRQ_7	
Port 7	Analog input	
Port 8	Serial communication (synchronous mode) E clock and $\overline{IOS}$ output IRQ_3 to IRQ_5	Serial communication (synchronous mode) Dual-port RAM address select input IRQ_3 to IRQ_5
Port 9	Bus control and Ø output*2 IRQ_0 to IRQ_2 and ADTRG	Dual-port RAM interface control, Ø output IRQ_0 to IRQ_2 and ADTRG

Notes:

*1 Selected automatically in mode 1; software-selectable in mode 2

*2 Selected automatically in modes 1 and 2

5.2 Port 1

Port 1 is an 8-bit input/output port that also provides the low bits of the address bus. The function of port 1 depends on the MCU mode as indicated in table 5-2.

Table 5-2. Functions of Port 1

Mode 1	Mode 2	Mode 3
Address bus (Low) (A_7 to A_0)	Input port or Address bus (Low) (A_7 to A_0)*	Input/output port

* Depending on the bit settings in the data direction register: 0—input pin; 1—address pin
Pins of port 1 can drive a single TTL load and a 90pF capacitive load when they are used as output pins. They can also drive light-emitting diodes and a Darlington pair. When they are used as input pins, they have programmable MOS pull-ups.

Table 5-3 details the port 1 registers.

Table 5-3. Port 1 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 1 data direction register	P1DDR	W	H'FF (mode 1) H'00 (modes 2 and 3)	H'FFB0
Port 1 data register	P1DR	R/W	H'00	H'FFB2

Port 1 Data Direction Register (P1DDR)—H'FFB0

Bit	7	6	5	4	3	2	1	0
	$P1_7$DDR	$P1_6$DDR	$P1_5$DDR	$P1_4$DDR	$P1_3$DDR	$P1_2$DDR	$P1_1$DDR	$P1_0$DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P1DDR is an 8-bit register that selects the direction of each pin in port 1. A pin functions as an output pin if the corresponding bit in P1DDR is set to "1," and as an input pin if the bit is cleared to "0."

Port 1 Data Register (P1DR)—H'FFB2

Bit	7	6	5	4	3	2	1	0
	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P1DR is an 8-bit register containing the data for pins $P1_7$ to $P1_0$. When the CPU reads P1DR, for output pins it reads the value in the P1DR latch, but for input pins, it obtains the logic level directly from the pin, bypassing the P1DR latch.

MOS Pull-Ups: Are available for input pins in modes 2 and 3. Software can turn on the MOS pull-up by writing a "1" in P1DR, and turn it off by writing a "0." The pull-ups are automatically turned off for output pins in modes 2 and 3, and for all pins in mode 1.

Mode 1: In mode 1 (expanded mode without on-chip ROM), port 1 is automatically used for address output. The port 1 data direction register is unwritable. All bits in P1DDR are automatically set to "1" and cannot be cleared to "0."

Mode 2: In mode 2 (expanded mode with on-chip ROM), the usage of port 1 can be selected on a pin-by-pin basis. A pin is used for general-purpose input if its data direction bit is cleared to "0," or for address output if its data direction bit is set to “1.”

Mode 3: In the single-chip mode port 1 is a general-purpose input/output port.

Reset: A reset clears P1DDR and P1DR to all “0,” placing all pins in the input state with the MOS pull-ups off. In mode 1, when the chip comes out of reset, P1DDR is set to all "1."

Hardware Standby Mode: All pins are placed in the high-impedance state with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, both P1DDR and P1DR remain in their previous state. Address output pins are Low. General-purpose output pins continue to output the data in P1DR.

Figure 5-1 shows a schematic diagram of port 1.

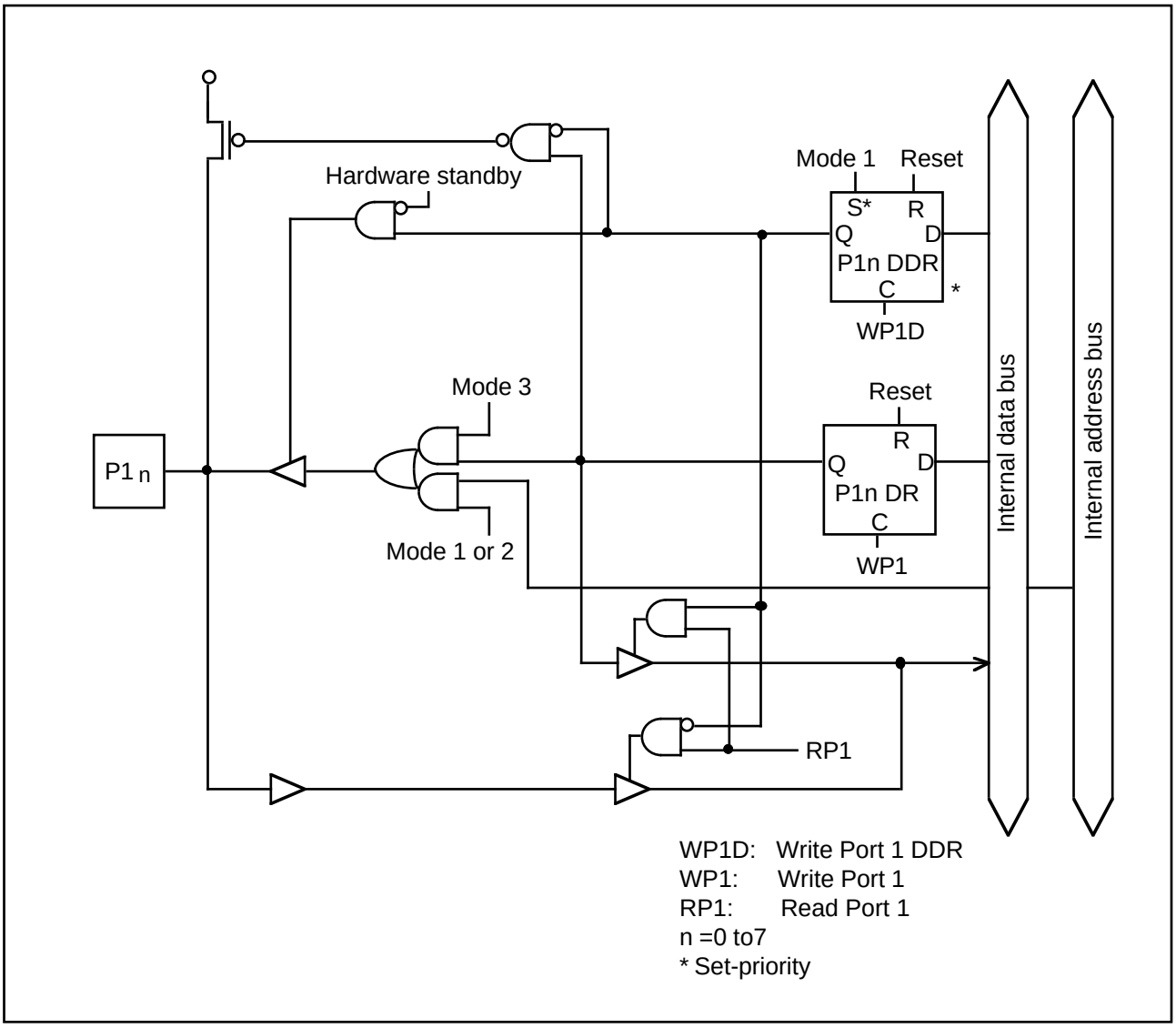


Figure 5-1. Port 1 Schematic Diagram

5.3 Port 2

Port 2 is an 8-bit input/output port that also provides the high bits of the address bus. The function of port 2 depends on the MCU mode as indicated in Table 5-4.

Table 5-4. Functions of Port 2

Mode 1	Mode 2	Mode 3
Address bus (High) (A_{15} to A_8)	Input port or Address bus (High) (A_{15} to A_8)*	Input/output port

* Depending on the bit settings in the data direction register: 0—input pin; 1—address pin

Pins of port 2 can drive a single TTL load and a 90pF capacitive load when they are used as output pins. They can also drive light-emitting diodes and a Darlington pair. When they are used as input pins, they have programmable MOS pull-ups.

Table 5-5 details the port 2 registers.

Table 5-5. Port 2 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 2 data direction register	P2DDR	W	H'FF (mode 1) H'00 (modes 2 and 3)	H'FFB1
Port 2 data register	P2DR	R/W	H'00	H'FFB3

Port 2 Data Direction Register (P2DDR)—H'FFB1

Bit	7	6	5	4	3	2	1	0
	$P2_7DDR$	$P2_6DDR$	$P2_5DDR$	$P2_4DDR$	$P2_3DDR$	$P2_2DDR$	$P2_1DDR$	$P2_0DDR$
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P2DDR is an 8-bit register that selects the direction of each pin in port 2. A pin functions as an output pin if the corresponding bit in P2DDR is set to "1," and as an input pin if the bit is cleared to "0."

Port 2 Data Register (P2DR)—H'FFB3

Bit	7	6	5	4	3	2	1	0
	$P2_7$	$P2_6$	$P2_5$	$P2_4$	$P2_3$	$P2_2$	$P2_1$	$P2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P2DR is an 8-bit register containing the data for pins $P2_7$ to $P2_0$. When the CPU reads P2DR, for output pins it reads the value in the P2DR latch, but for input pins, it obtains the logic level directly from the pin, bypassing the P2DR latch.

MOS Pull-Ups: Are available for input pins in modes 2 and 3. Software can turn on the MOS pull-up by writing a “1” in P2DR, and turn it off by writing a “0.” The pull-ups are automatically turned off for output pins in modes 2 and 3, and for all pins in mode 1.

Mode 1: In mode 1 (expanded mode without on-chip ROM), port 2 is automatically used for address output. The port 2 data direction register is unwritable. All bits in P2DDR are automatically set to "1" and cannot be cleared to "0."

Mode 2: In mode 2 (expanded mode with on-chip ROM), the usage of port 2 can be selected on a pin-by-pin basis. A pin is used for general-purpose input if its data direction bit is cleared to "0," or for address output if its data direction bit is set to “1.”

Mode 3: In the single-chip mode port 2 is a general-purpose input/output port.

Reset: A reset clears P2DDR and P2DR to all “0,” placing all pins in the input state with the MOS pull-ups off. In mode 1, when the chip comes out of reset, P2DDR is set to all "1."

Hardware Standby Mode: All pins are placed in the high-impedance state with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, both P2DDR and P2DR remain in their previous state. Address output pins are Low. General-purpose output pins continue to output the data in P2DR.

Figure 5-2 shows a schematic diagram of port 2.

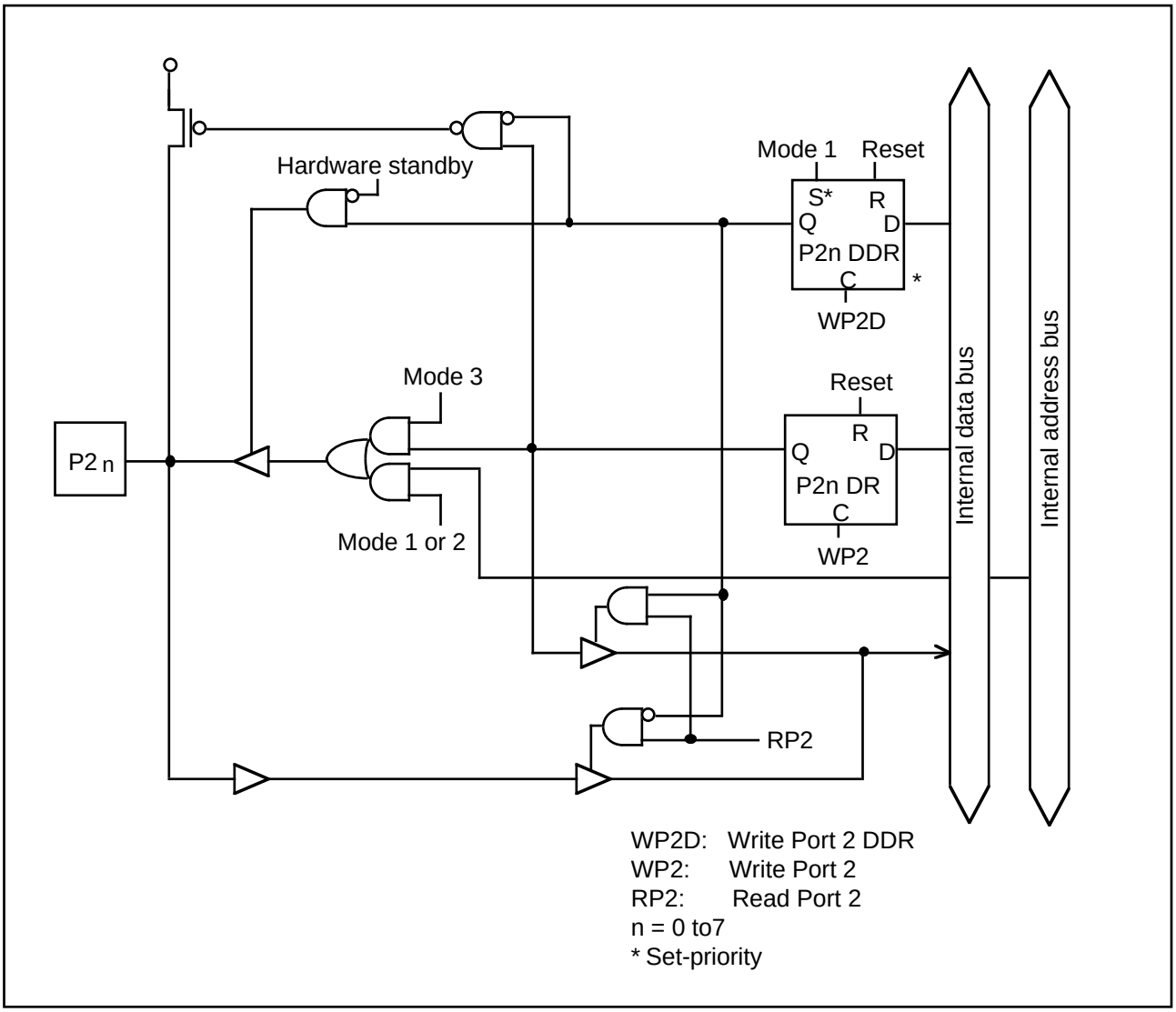


Figure 5-2. Port 2 Schematic Diagram

5.4 Port 3

Port 3 is an 8-bit input/output port that also provides the external data bus and dual-port RAM (master-slave) data bus. The function of port 3 depends on the MCU mode as indicated in table 5-6.

Table 5-6. Functions of Port 3

		Mode 3	
Mode 1	**Mode 2**	**DPME = "0"**	**DPME = "1"**
Data bus	Data bus	Input/output port	Dual-port RAM data bus

Pins of port 3 can drive a single TTL load and a 90pF capacitive load when they are used as output

pins. They can also drive a Darlington pair. When they are used as input pins, they have programmable MOS pull-ups.

Table 5-7 details the port 3 registers.

Table 5-7. Port 3 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 3 data direction register	P3DDR	W	H'00	H'FFB4
Port 3 data register	P3DR	R/W	H'00	H'FFB6

Port 3 Data Direction Register (P3DDR)—H'FFB4

Bit	7	6	5	4	3	2	1	0
	$P3_7$DDR	$P3_6$DDR	$P3_5$DDR	$P3_4$DDR	$P3_3$DDR	$P3_2$DDR	$P3_1$DDR	$P3_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P3DDR is an 8-bit register that selects the direction of each pin in port 3. A pin functions as an output pin if the corresponding bit in P3DDR is set to "1," and as an input pin if the bit is cleared to "0."

Port 3 Data Register (P3DR)—H'FFB6

Bit	7	6	5	4	3	2	1	0
	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3DR is an 8-bit register containing the data for pins $P3_7$ to $P3_0$. When the CPU reads P3DR, for output pins it reads the value in the P3DR latch, but for input pins, it obtains the logic level directly from the pin, bypassing the P3DR latch.

MOS Pull-Ups: Are available for input pins in mode 3 when the dual-port RAM is disabled. Software can turn on the MOS pull-up on by writing a "1" in P3DR, and turn it off by writing a "0."

The MOS pull-ups cannot be used in slave mode (when the dual-port RAM is enabled). P3DR should be cleared to H'00 (its initial value) in slave mode.

Modes 1 and 2: In the expanded modes, port 3 is automatically used as the data bus. The values in P3DDR and P3DR are ignored.

Mode 3: In the single-chip mode, when the dual-port RAM enable (DPME) bit in the system control register is cleared to “0,” port 3 can be used as a general-purpose input/output port.

When DPME is set to “1,” entering the slave mode, port 3 is used as the dual-port RAM data bus (DDB_7 to DDB_0). P3DR should also be cleared to H'00 in slave mode.

See section 12, “Dual-Port RAM” for further information.

Reset and Hardware Standby Mode: A reset or entry to the hardware standby mode clears P3DDR and P3DR to all “0,” and clears the DPME bit to "0." In modes 1 and 2, all pins are placed in the data input (high-impedance) state. In mode 3 (single-chip mode), all pins are in the input state with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, P3DDR, P3DR, and the DPME bit remain in their previous state. In modes 1 and 2 and slave mode, all pins are placed in the data input (high-impedance) state. In mode 3 with the dual-port RAM disabled, all pins remain in their previous input or output state.

Figure 5-3 shows a schematic diagram of port 3.

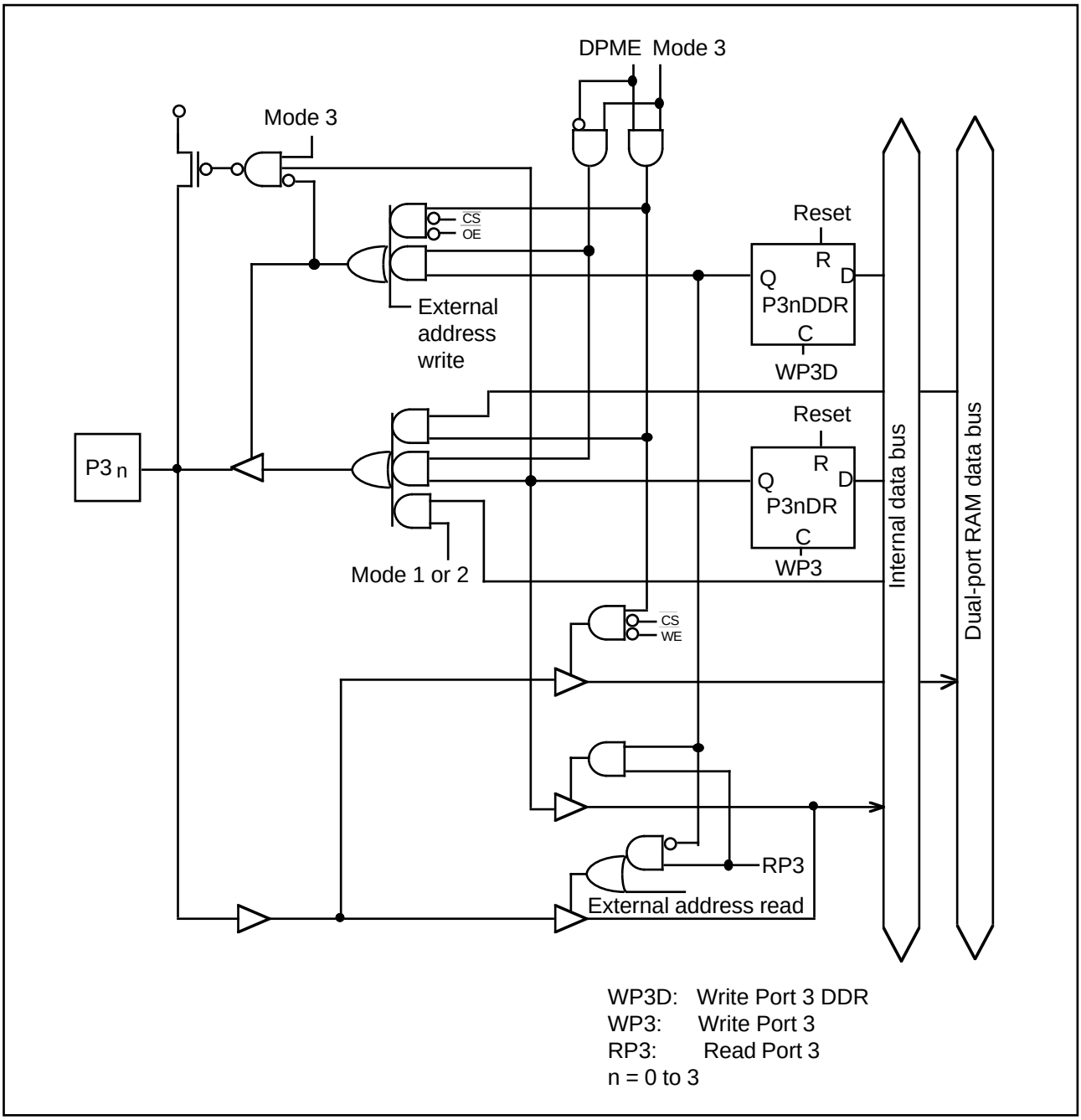


Figure 5-3. Port 3 Schematic Diagram

5.5 Port 4

Port 4 is an 8-bit input/output port that also provides the input and output pins for the 8-bit timers and the output pins for the PWM timers. The pin functions depend on control bits in the control registers of the timers. Pins not used by the timers are available for general-purpose input/output. Table 5-8 lists the pin functions, which are the same in both the expanded and single-chip modes.

Table 5-8. Port 4 Pin Functions (Modes 1 to 3)

Usage	Pin functions							
I/O port	$P4_0$	$P4_1$	$P4_2$	$P4_3$	$P4_4$	$P4_5$	$P4_6$	$P4_7$
Timer	$TMCI_0$	TMO_0	$TMRI_0$	$TMCI_1$	TMO_1	$TMRI_1$	PW_0	PW_1

See section 7, “8-Bit Timer Module” and section 8, “PWM Timer Module” for details of the timer control bits.

Pins of port 4 can drive a single TTL load and a 90pF capacitive load when they are used as output pins. They can also drive a Darlington pair. When used as input pins, they have programmable MOS pull-ups.

Table 5-9 details the port 4 registers.

Table 5-9. Port 4 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 4 data direction register	P4DDR	W	H’00	H’FFB5
Port 4 data register	P4DR	R/W	H’00	H’FFB7

Port 4 Data Direction Register (P4DDR)—H’FFB5

Bit	7	6	5	4	3	2	1	0
	$P4_7$DDR	$P4_6$DDR	$P4_5$DDR	$P4_4$DDR	$P4_3$DDR	$P4_2$DDR	$P4_1$DDR	$P4_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P4DDR is an 8-bit register that selects the direction of each pin in port 4. A pin functions as an output pin if the corresponding bit in P4DDR is set to “1,” and as an input pin if the bit is cleared to “0.”

Port 4 Data Register (P4DR)—H'FFB7

Bit	7	6	5	4	3	2	1	0
	$P4_7$	$P4_6$	$P4_5$	$P4_4$	$P4_3$	$P4_2$	$P4_1$	$P4_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P4DR is an 8-bit register containing the data for pins $P4_7$ to $P4_0$. When the CPU reads P4DR, for output pins (P4DDR = "1") it reads the value in the P4DR latch, but for input pins (P4DDR = "0"), it obtains the logic level directly from the pin, bypassing the P4DR latch. This also applies to pins used for timer input or output.

MOS Pull-Ups: Are available for input pins, including timer input pins, in all modes. Software can turn the MOS pull-up on by writing a "1" in P4DR, and turn it off by writing a "0."

Pins $P4_0$, $P4_2$, $P4_3$, and $P4_5$: As indicated in Table 5-8, these pins can be used for general-purpose input or output, or input of 8-bit timer clock and reset signals. When a pin is used for timer signal input, its P4DDR bit should normally be cleared to "0;" otherwise the timer will receive the value in P4DR. If input pull-up is not desired, the P4DR bit should also be cleared to "0."

Pins $P4_1$, $P4_4$, $P4_6$, and $P4_7$: As indicated in Table 5-8, these pins can be used for general-purpose input or output, or for 8-bit timer output ($P4_1$ and $P4_4$) or PWM timer output ($P4_6$ and $P4_7$). Pins used for timer output are unaffected by the values in P4DDR and P4DR, and their MOS pull-ups are automatically turned off.

Reset and Hardware Standby Mode: A reset or entry to the hardware standby mode clears P4DDR and P4DR to all "0" and makes all pins into input port pins with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, the control registers of the 8-bit and PWM timers are initialized but P4DDR and P4DR remain in their previous states. All pins become input or output port pins depending on the setting of P4DDR. Output pins output the values in P4DR. The MOS pull-ups of input pins are on or off depending on the values in P4DR.

Figures 5-4 and 5-5 show schematic diagrams of port 4.

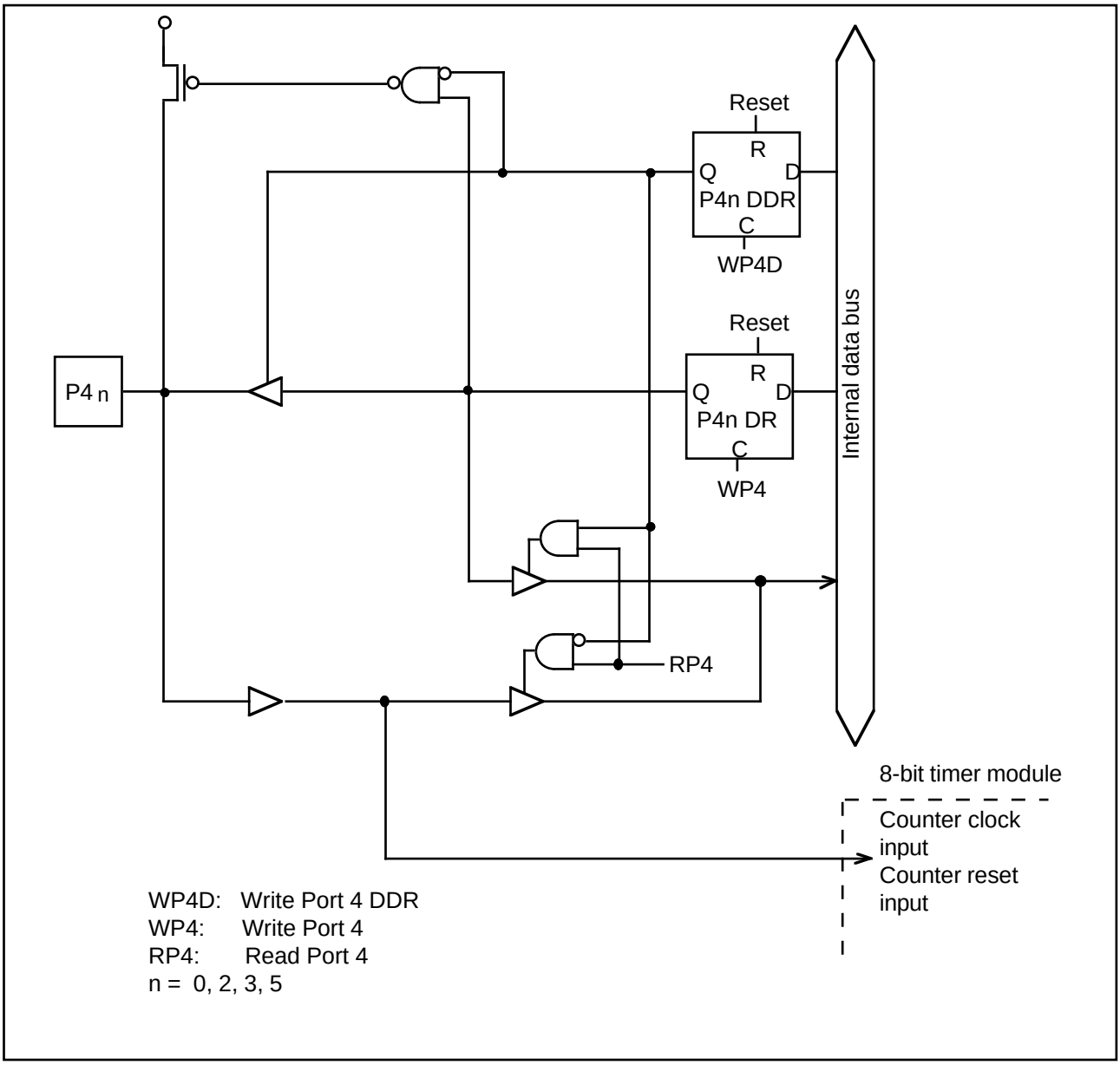


Figure 5-4. Port 4 Schematic Diagram (Pins $P4_0$, $P4_2$, $P4_3$, and $P4_5$)

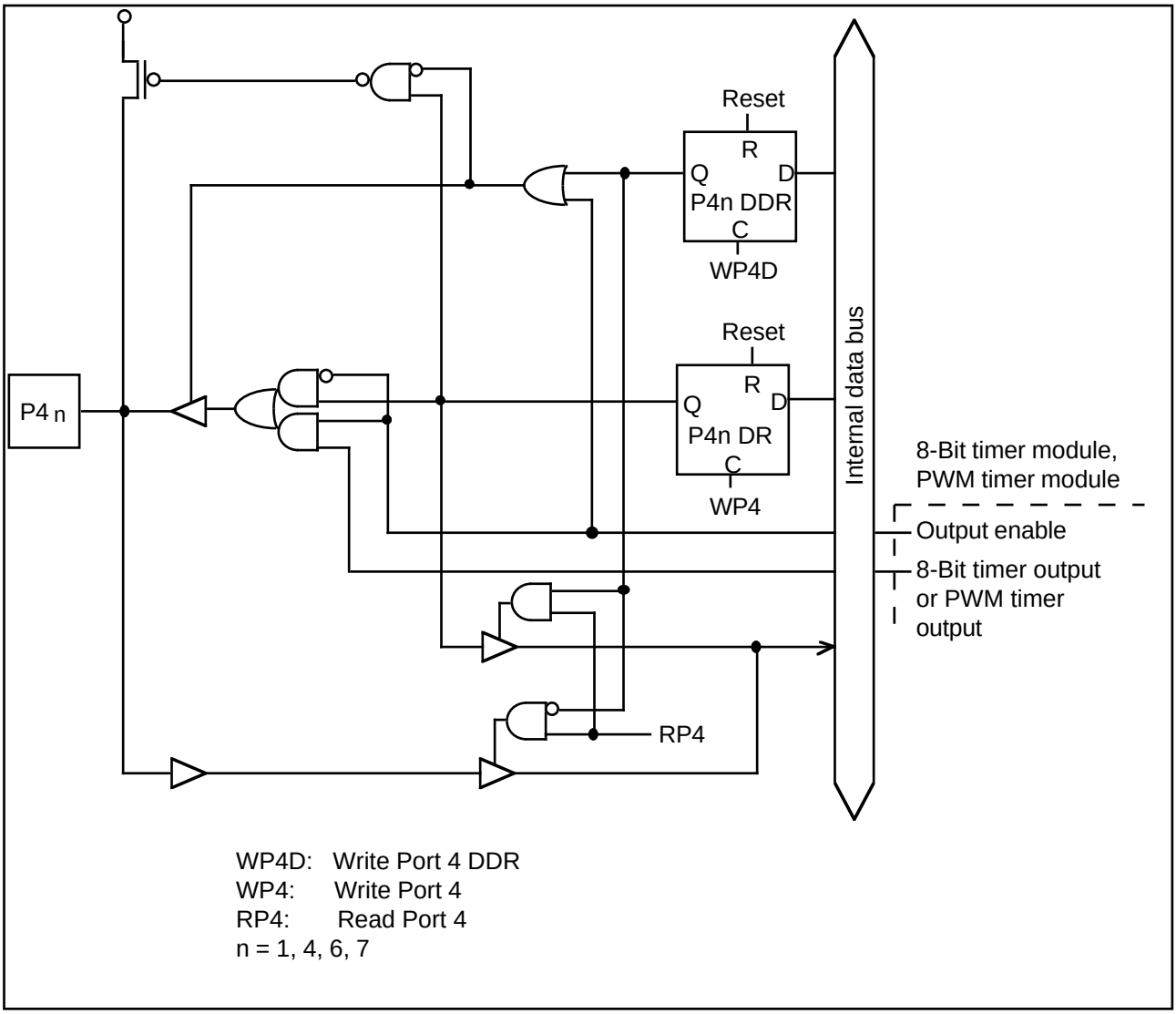


Figure 5-5. Port 4 Schematic Diagram (Pins P4$_1$, P4$_4$, P4$_6$, and P4$_7$)

5.6 Port 5

Port 5 is a 3-bit input/output port that also provides the input and output pins for asynchronous serial communication. The pin functions depend on control bits in the serial control register (SCR). Pins not used for serial communication are available for general-purpose input/output. Table 5-10 lists the pin functions, which are the same in both the expanded and single-chip modes.

Table 5-10. Port 5 Pin Functions (Modes 1 to 3)

Usage	Pin functions		
I/O port	P5$_0$	P5$_1$	P5$_2$
Timer	ATxD	ARxD	ASCK

See section 9, “Serial Communication Interface” for details of the serial control bits. Pins used by the serial communication interface are switched between input and output without regard to the values in the data direction register.

Pins of port 5 can drive a single TTL load and a 30pF capacitive load when they are used as output pins. They can also drive a Darlington pair. When used as input pins, they have programmable MOS pull-ups.

Table 5-11 details the port 5 registers.

Table 5-11. Port 5 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 5 data direction register	P5DDR	W	H’F8	H’FFB8
Port 5 data register	P5DR	R/W	H’F8	H’FFBA

Port 5 Data Direction Register (P5DDR)—H’FFB8

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$P5_2$DDR	$P5_1$DDR	$P5_0$DDR
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	W	W	W

P5DDR is an 8-bit register that selects the direction of each pin in port 5. A pin functions as an output pin if the corresponding bit in P5DDR is set to “1,” and as an input pin if the bit is cleared to “0.”

Port 5 Data Register (P5DR)—H’FFBA

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$P5_2$	$P5_1$	$P5_0$
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

P5DR is an 8-bit register containing the data for pins $P5_2$ to $P5_0$. When the CPU reads P5DR, for output pins (P5DDR = "1") it reads the value in the P5DR latch, but for input pins (P5DDR = "0"), it obtains the logic level directly from the pin, bypassing the P5DR latch. This also applies to pins

used for serial communication.

MOS Pull-Ups: Are available for input pins, including serial communication input pins. Software can turn the MOS pull-up on by writing a “1” in P5DR, and turn it off by writing a “0.”

Pin P5$_0$: This pin can be used for general-purpose input or output, or for output of asynchronous serial transmit data (ATxD). When used for ATxD output, this pin is unaffected by the values in P5DDR and P5DR, and its MOS pull-up is automatically turned off.

Pin P5$_1$: This pin can be used for general-purpose input or output, or for input of asynchronous serial receive data (ARxD). When used for ARxD input, this pin is unaffected by P5DDR and P5DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1."

Pin P5$_2$: This pin can be used for general-purpose input or output, or for asynchronous serial clock input or output (ASCK). When used for ASCK input or output, this pin is unaffected by P5DDR and P5DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1." For ASCK usage, the MOS pull-up should be turned off.

Reset and Hardware Standby Mode: A reset or entry to the hardware standby mode makes all pins of port 5 into input port pins with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, the serial control register is initialized but P5DDR and P5DR remain in their previous states. All pins become input or output port pins depending on the setting of P5DDR. Output pins output the values in P5DR. The MOS pull-ups of input pins are on or off depending on the values in P5DR.

Figures 5-6 to 5-8 show schematic diagrams of port 5.

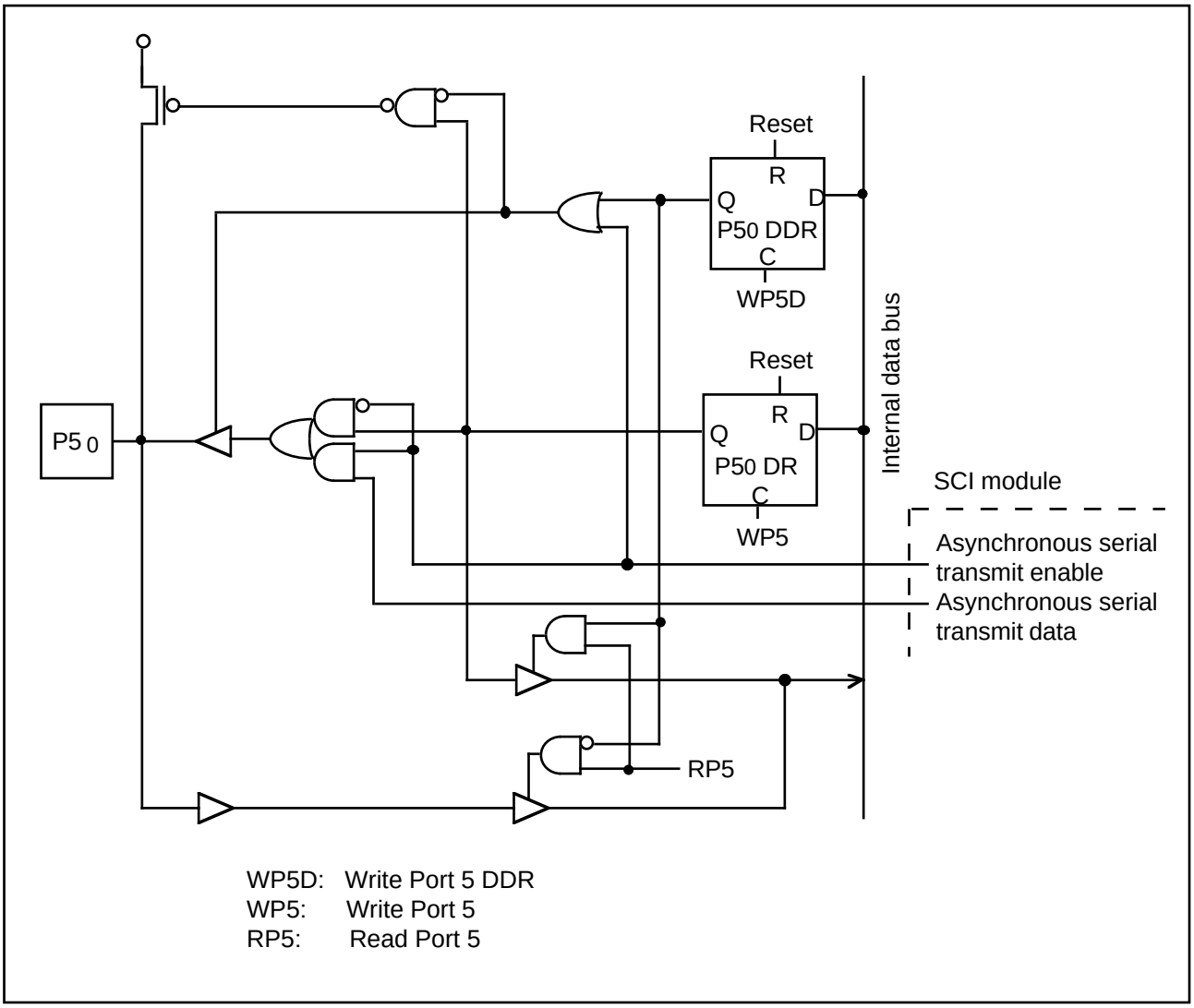


Figure 5-6. Port 5 Schematic Diagram (Pin P5$_0$)

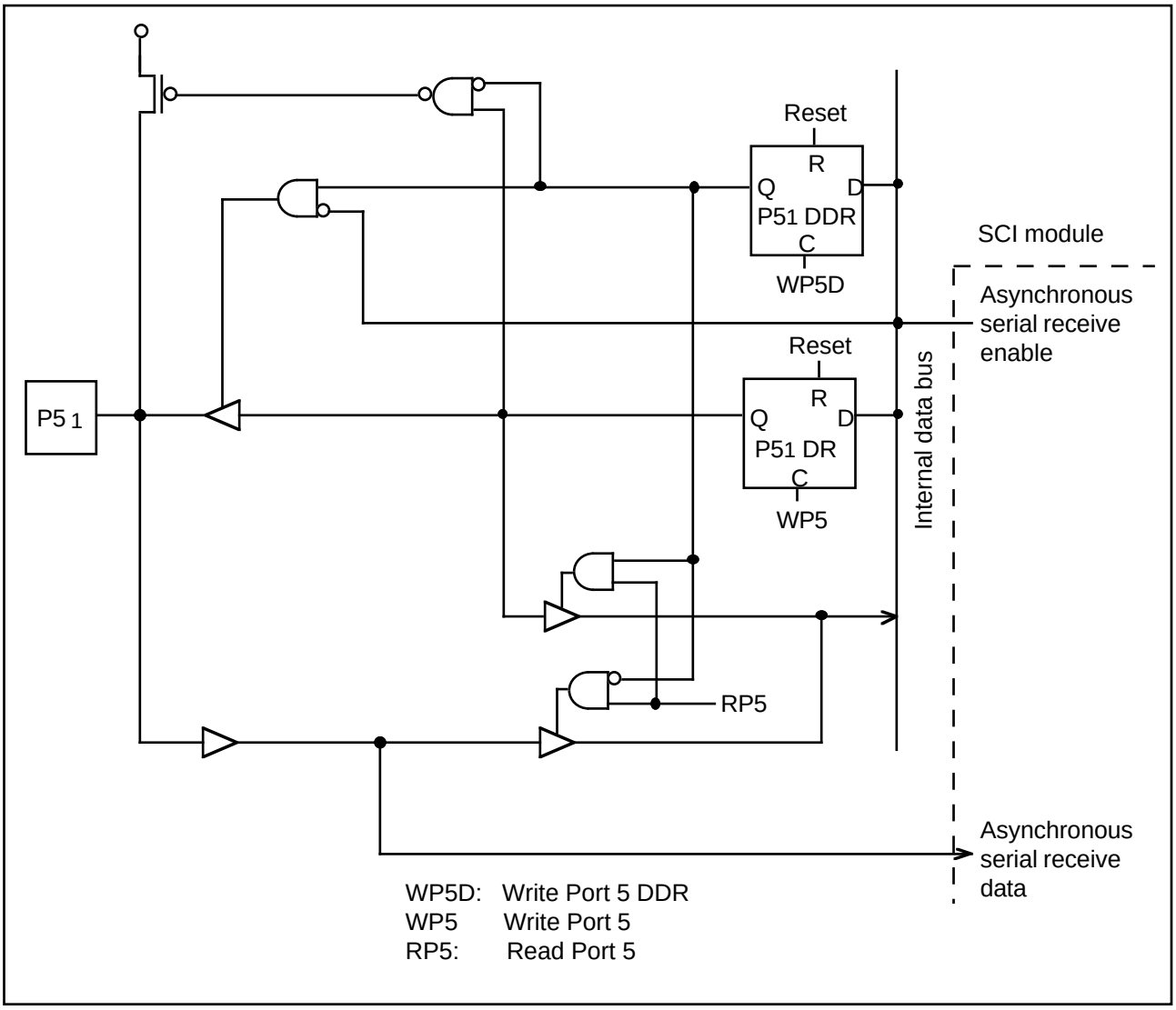


Figure 5-7. Port 5 Schematic Diagram (Pin P5$_1$)

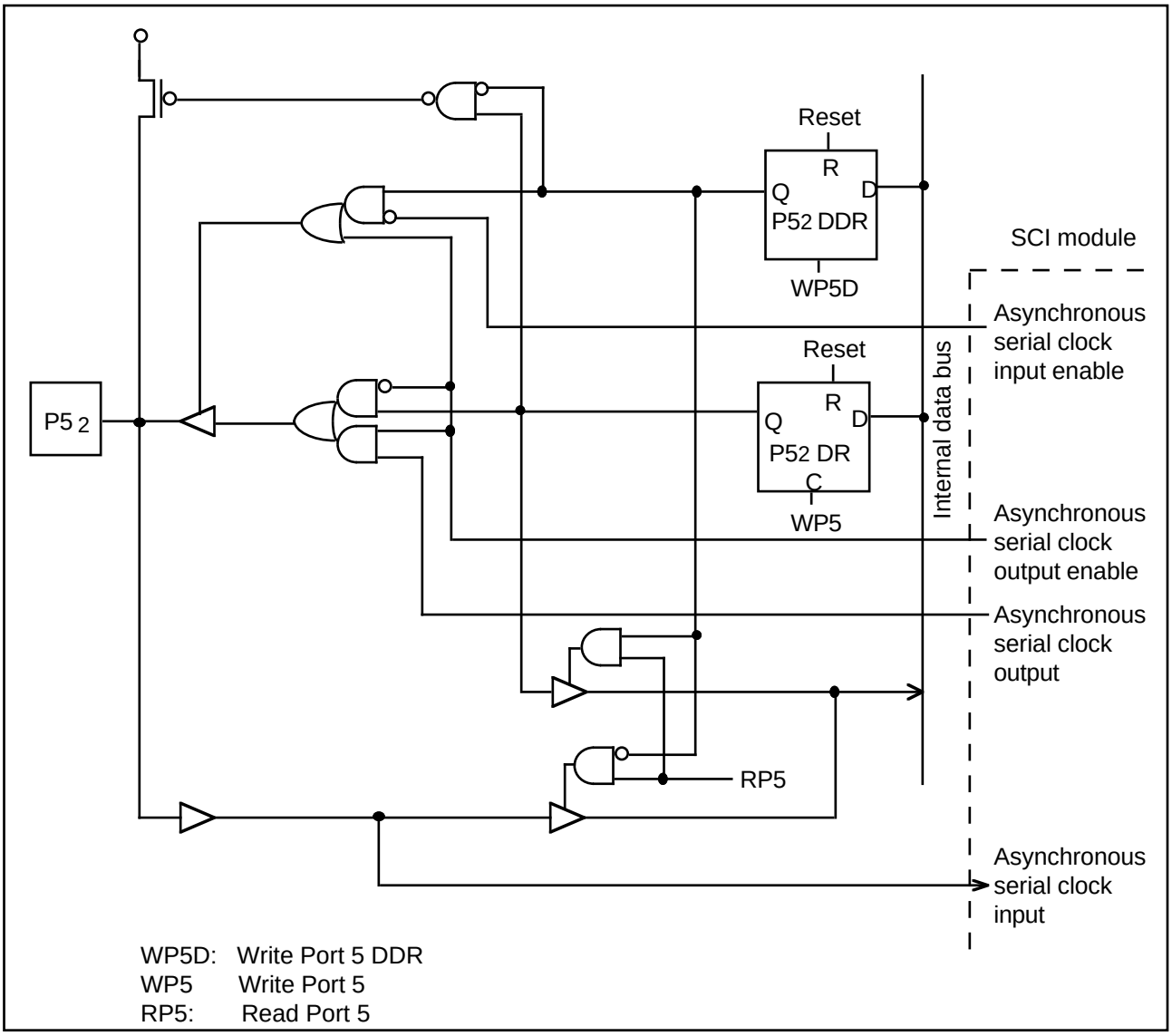


Figure 5-8. Port 5 Schematic Diagram (Pin P5$_2$)

5.7 Port 6

Port 6 is an 8-bit input/output port that also provides the input and output pins for the free-running timer and the IRQ$_6$ and IRQ$_7$ input/output pins. The pin functions depend on control bits in the free-running timer control registers, and on bit 6 or 7 of the interrupt enable register. Pins not used for timer or interrupt functions are available for general-purpose input/output. Table 5-12 lists the pin functions, which are the same in both the expanded and single-chip modes.

Table 5-12. Port 6 Pin Functions

Usage Pin functions (Modes 1 to 3)

I/O port	P6$_0$	P6$_1$	P6$_2$	P6$_3$	P6$_4$	P6$_5$	P6$_6$	P6$_7$
Timer/interrupt	FTCI	FTOA	FTIA	FTIB	FTIC	FTID	FTOB/IRQ$_6$	IRQ$_7$

See section 4 “Exception Handling” and section 6, “Free-Running Timer Module” for details of the free-running timer and interrupts.

Pins of port 6 can drive a single TTL load and a 90pF capacitive load when they are used as output pins. They can also drive a Darlington pair. When they are used as input pins, they have programmable MOS pull-ups.

Table 5-13 details the port 6 registers.

Table 5-13. Port 6 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 6 data direction register	P6DDR	W	H’00	H’FFB9
Port 6 data register	P6DR	R/W	H’00	H’FFBB

Port 6 Data Direction Register (P6DDR)—H’FFB9

Bit	7	6	5	4	3	2	1	0
	$P6_7$DDR	$P6_6$DDR	$P6_5$DDR	$P6_4$DDR	$P6_3$DDR	$P6_2$DDR	$P6_1$DDR	$P6_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P6DDR is an 8-bit register that selects the direction of each pin in port 6. A pin functions as an output pin if the corresponding bit in P6DDR is set to “1,” and as an input pin if the bit is cleared to “0.”

Port 6 Data Register (P6DR)—H’FFBB

Bit	7	6	5	4	3	2	1	0
	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P6DR is an 8-bit register containing the data for pins $P6_7$ to $P6_0$. When the CPU reads P6DR, for output pins (P6DDR = "1") it reads the value in the P6DR latch, but for input pins (P6DDR = "0"), it obtains the logic level directly from the pin, bypassing the P6DR latch. This also applies to pins used for input and output of timer and interrupt signals.

MOS Pull-Ups: Are available for input pins, including pins used for input of timer or interrupt signals. Software can turn the MOS pull-up on by writing a “1” in P6DR, and turn it off by writing a “0.”

Pins $P6_0$, $P6_2$, $P6_3$, $P6_4$ and $P6_5$: As indicated in Table 5-12, these pins can be used for general-purpose input or output, or for input of free-running timer clock and input capture signals. When a pin is used for free-running timer input, its P6DDR bit should be cleared to "0;" otherwise the free-running timer will receive the value in P6DR. If input pull-up is not desired, the P6DR bit should also be cleared to "0."

Pin $P6_1$: This pin can be used for general-purpose input or output, or for the output compare A signal (FTOA) of the free-running timer. When used for FTOA output, this pin is unaffected by the values in P6DDR and P6DR, and its MOS pull-up is automatically turned off.

Pin $P6_6$: This pin can be used for general-purpose input or output, for the output compare B signal (FTOB) of the free-running timer, or for $\overline{IRQ}_6$ input. When used for FTOB output, this pin is unaffected by the values in P6DDR and P6DR, and its MOS pull-up is automatically turned off. When this pin is used for $\overline{IRQ}_6$ input, $P6_6$DDR should normally be cleared to "0," so that the value in P6DR will not generate interrupts.

Pin $P6_7$: This pin can be used for general-purpose input or output, or $\overline{IRQ}_7$ input. When it is used for $\overline{IRQ}_7$ input, $P6_7$DDR should normally be cleared to "0," so that the value in P6DR will not generate interrupts.

Reset and Hardware Standby Mode: A reset or entry to the hardware standby mode clears P6DDR and P6DR to all “0” and makes all pins into input port pins with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, the free-running timer control registers are initialized but P6DDR and P6DR remain in their previous states. All pins become input or output port pins depending on the setting of P6DDR. Output pins output the values in P6DR. The MOS pull-ups of input pins are on or off depending on the values in P6DR.

Figures 5-9 to 5-11 shows schematic diagrams of port 6.

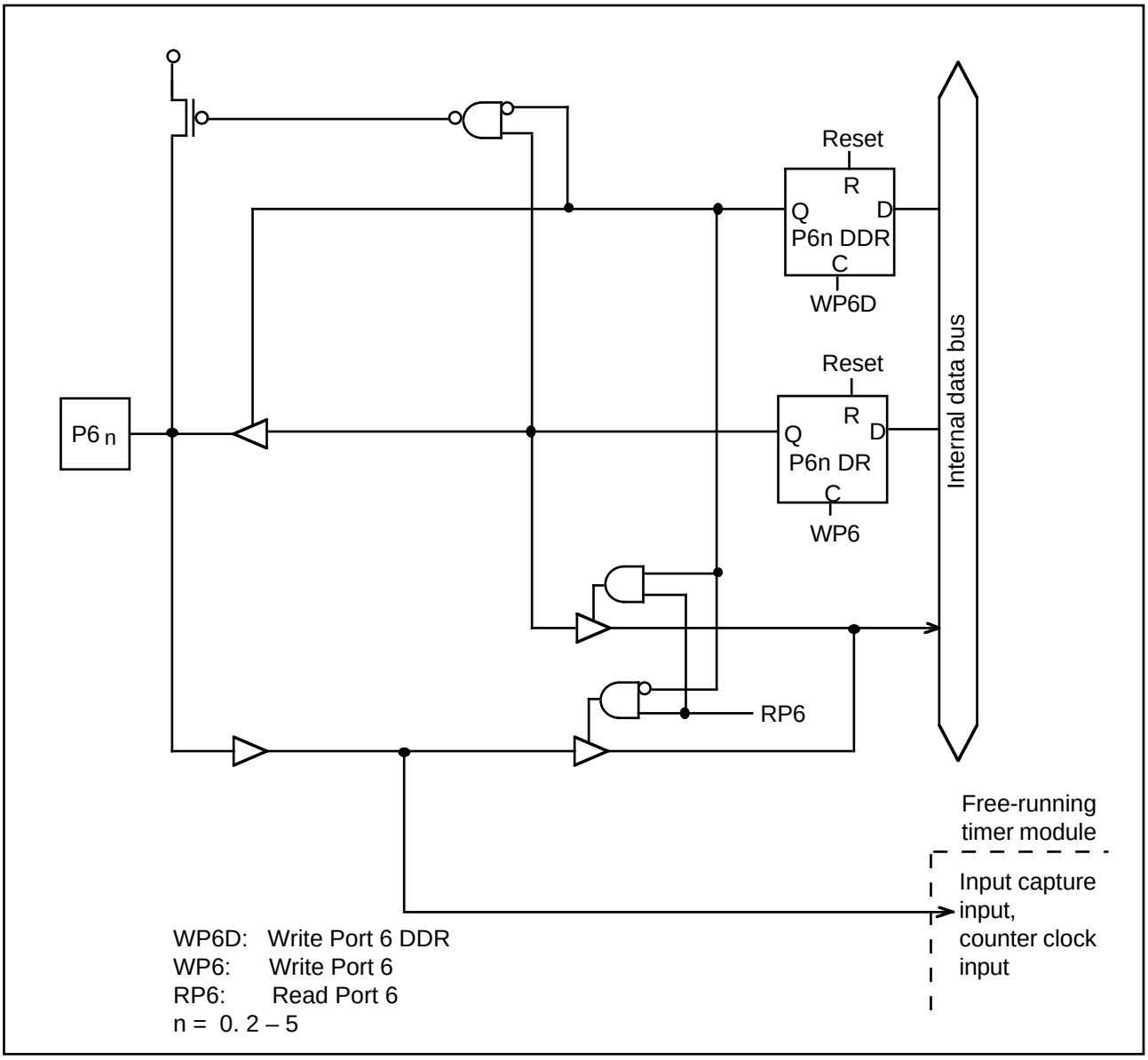


Figure 5-9. Port 6 Schematic Diagram (Pins P6$_0$, P6$_2$, P6$_3$, P6$_4$, and P6$_5$)

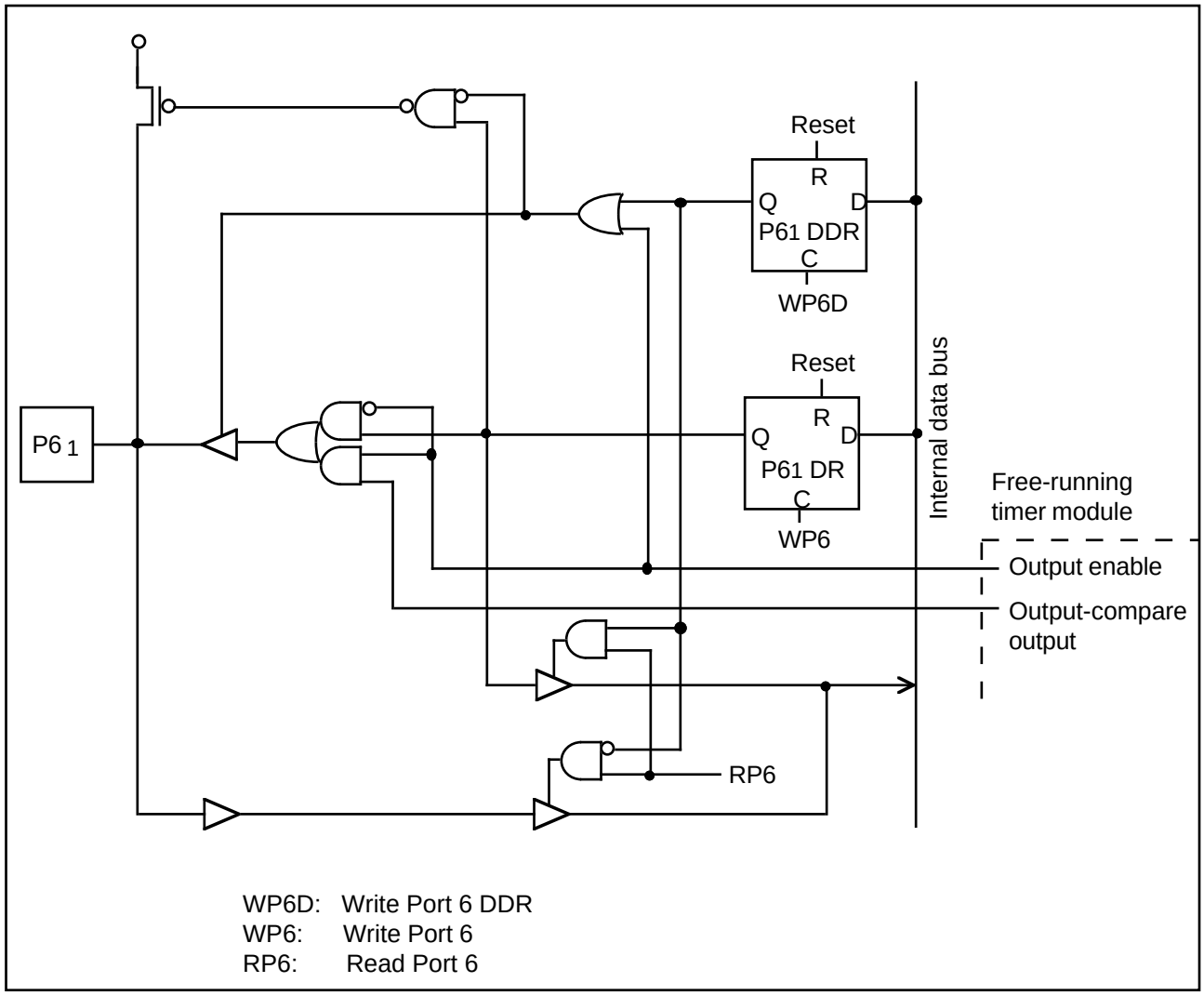


Figure 5-10. Port 6 Schematic Diagram (Pin P6$_1$)

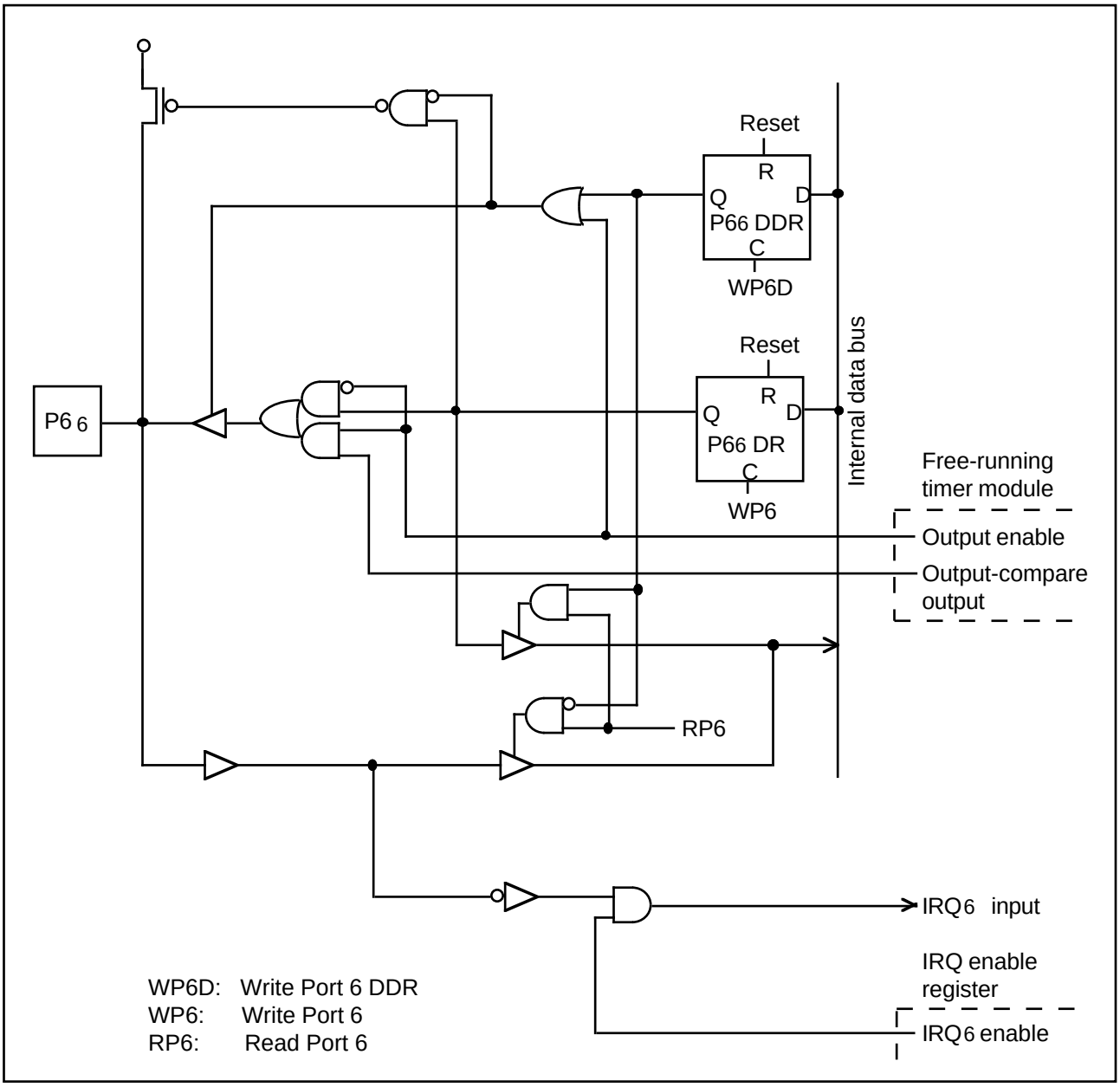


Figure 5-11. Port 6 Schematic Diagram (Pin P6$_6$)

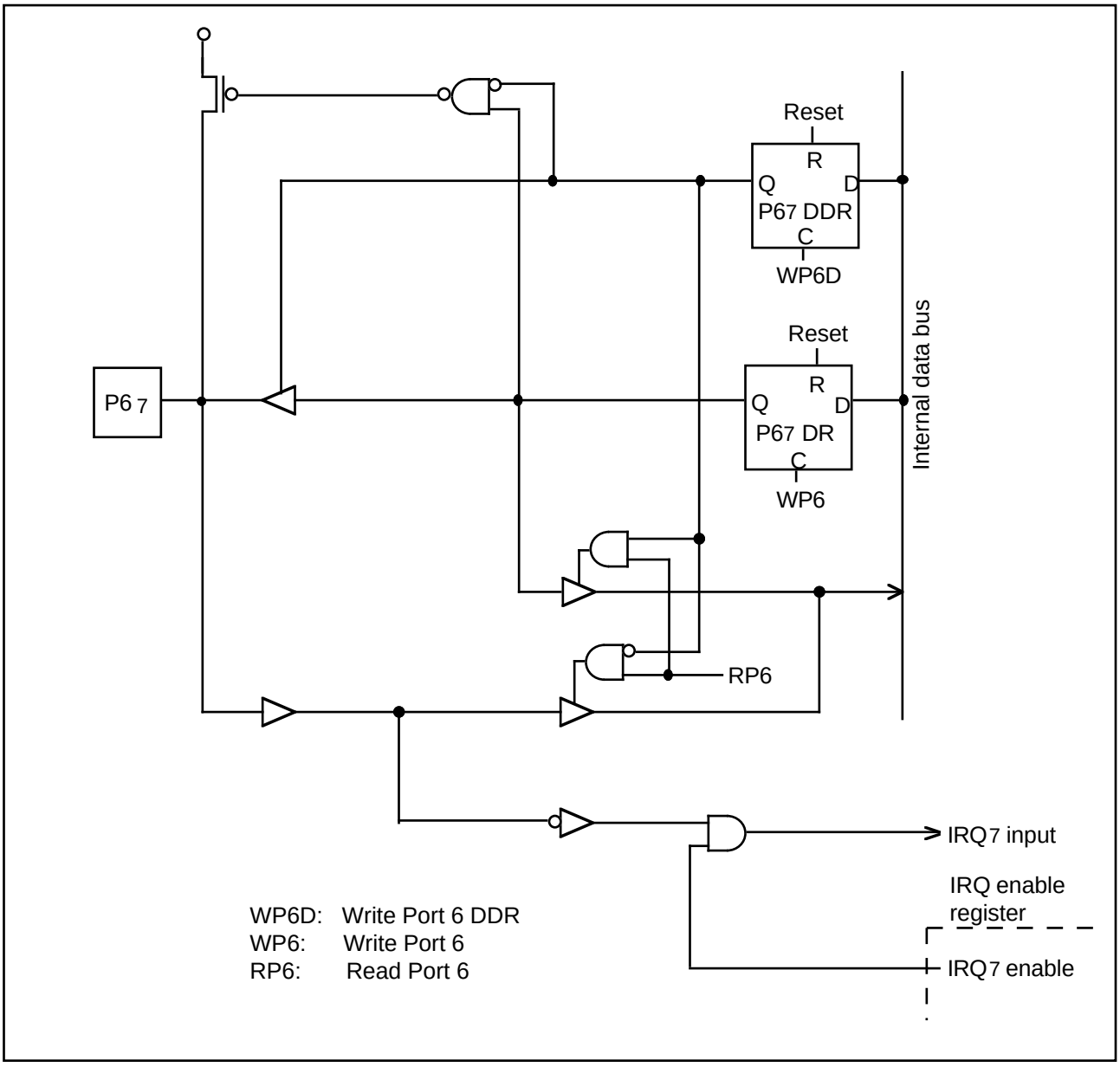


Figure 5-12. Port 6 Schematic Diagram (Pin P6$_7$)

5.8 Port 7

Port 7 is an 8-bit input port that also provides the analog input pins for the A/D converter module. The pin functions are the same in both the expanded and single-chip modes.

Table 5-14 lists the pin functions. Table 5-15 describes the port 7 data register, which simply consists of connections of the port 7 pins to the internal data bus. Figure 5-13 shows a schematic diagram of port 7.

Table 5-14. Port 7 Pin Functions (Modes 1 to 3)

Usage	Pin functions							
I/O port	$P7_0$	$P7_1$	$P7_2$	$P7_3$	$P7_4$	$P7_5$	$P7_6$	$P7_7$
Analog input	AN_0	AN_1	AN_2	AN_3	AN_4	AN_5	AN_6	AN_7

Table 5-15. Port 7 Register

Name	Abbreviation	Read/Write	Initial value	Address
Port 7 data register	P7DR	R	Undetermined	H'FFBE

Port 7 Data Register (P7DR)—H'FFBE

Bit	7	6	5	4	3	2	1	0
	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$
Initial value	*	*	*	*	*	*	*	*
Read/Write	R	R	R	R	R	R	R	R

* Depends on the levels of pins $P7_7$ to $P7_0$.

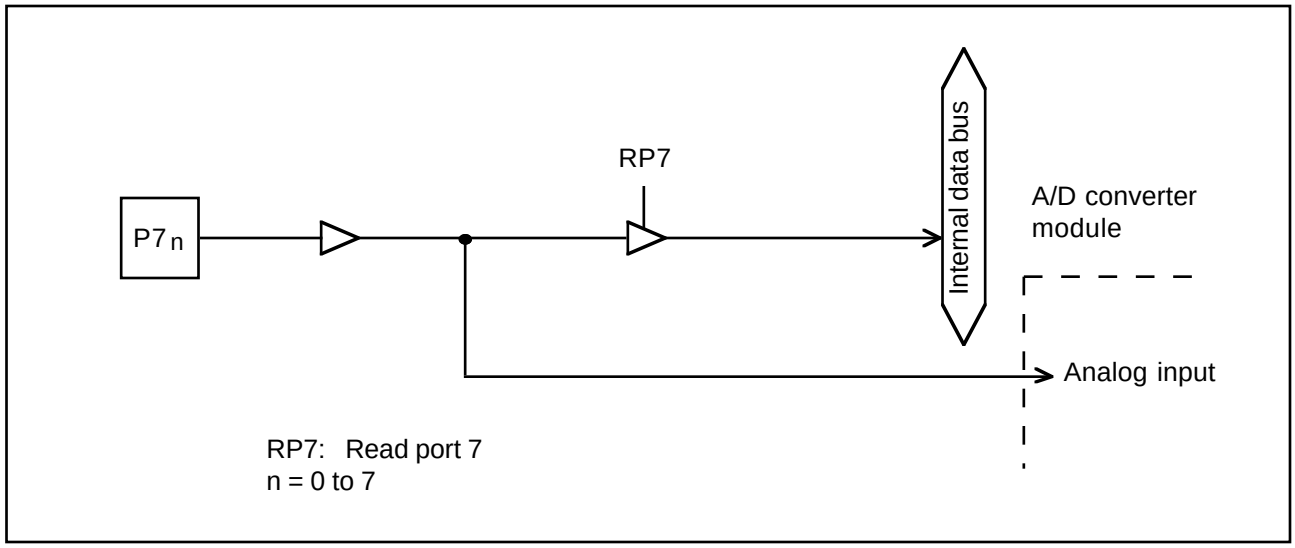


Figure 5-13. Port 7 Schematic Diagram

5.9 Port 8

Port 8 is a 7-bit input/output port that also provides pins for E clock output, dual-port RAM register select input, interrupt input, and clock-synchronized serial communication. Table 5-16 lists the pin functions.

Table 5-16. Port 8 Pin Functions

	Auxiliary functions	
I/O port	**Expanded modes**	**Single-chip mode**
$P8_0$ input/output	E clock output	RS0 input
$P8_1$ input/output	$\overline{IOS}$ output	RS1 input
$P8_2$ input/output	—	RS2 input
$P8_3$ input/output	—	RS3 input
$P8_4$ input/output	CTxD output /IRQ_3 input	
$P8_5$ input/output	CRxD input /IRQ_4 input	
$P8_6$ input/output	CSCK input/output /IRQ_5 input	

Pins of port 8 can drive a single TTL load and a 30pF capacitive load when they are used as output pins. They can also drive a Darlington pair. When used as input pins, they have programmable MOS pull-ups.

Table 5-17 details the port 8 registers.

Table 5-17. Port 8 Registers

Name	Abbreviation	Read/Write	Initial value	Address
Port 8 data direction register	P8DDR	W	H’81 (modes 1 and 2) H’80 (mode 3)	H’FFBD
Port 8 data register	P8DR	R/W	H'80	H’FFBF

Port 8 Data Direction Register (P8DDR)—H’FFBD

Bit	7	6	5	4	3	2	1	0
	—	$P8_6$DDR	$P8_5$DDR	$P8_4$DDR	$P8_3$DDR	$P8_2$DDR	$P8_1$DDR	$P8_0$DDR
Modes 1 and 2								
Initial value	1	0	0	0	0	0	0	1
Read/Write	—	W	W	W	W	W	W	W
Mode 3								
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

P8DDR is an 8-bit register that selects the direction of each pin in port 8. A pin functions as an output pin if the corresponding bit in P8DDR is set to “1,” and as in input pin if the bit is cleared to “0.”

Bit 7 is reserved. It cannot be modified, and is always read as "1."

Port 8 Data Register (P8DR)—H’FFBF

Bit	7	6	5	4	3	2	1	0
	—	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P8DR is an 8-bit register containing the data for pins $P8_6$ to $P8_0$. When the CPU reads P8DR, for output pins (P8DDR = "1") it reads the value in the P8DR latch, but for input pins (P8DDR = "0"), it obtains the logic level directly from the pin, bypassing the P8DR latch. This also applies to pins used for dual-port RAM register select input, interrupt input, serial communication, and E clock or $\overline{\text{IOS}}$ output.

Bit 7 is reserved. It cannot be modified, and is always read as "1."

MOS Pull-Ups: Are available for input pins in all modes, including pins used for dual-port RAM register select input, interrupt input, or serial communication input. Software can turn the MOS pull-up on by writing a “1” in P8DR, and turn it off by writing a “0.”

Pin P8$_0$: In modes 1 and 2 (expanded modes), pin P8$_0$ is used for E clock output if P8$_0$DDR is set to "1," and for general-purpose input if P8$_0$DDR is cleared “0.” It cannot be used for general-purpose output.

In mode 3 (single-chip mode), when the dual-port RAM is disabled (DPME = "0"), pin P8$_0$ can be used for general-purpose input or output. In the slave mode (DPME = "1"), this pin is used for register select input (RS0). In slave mode this pin is unaffected by the values in P8DDR and P8DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1."

Pin P8$_1$: In modes 1 and 2 (expanded modes), pin P8$_1$ is used for $\overline{\text{IOS}}$ output if P8$_1$DDR is set to "1," and for general-purpose input if P8$_1$DDR is cleared “0.” It cannot be used for general-purpose output.

In mode 3, when the dual-port RAM is disabled (DPME = "0"), pin P8$_1$ can be used for general-purpose input or output. In the slave mode (DPME = "1"), this pin is used for register select input (RS1). In slave mode this pin is unaffected by the values in P8DDR and P8DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1."

Pins P8$_2$ and P8$_3$: These pins are available for general-purpose input or output in modes 1 and 2, and in mode 3 if the dual-port RAM is disabled (DPME = "0").

In the slave mode (mode 3 with DPME = "1"), these pins are used for register select input (RS2 and RS3). They are unaffected by the bits in P8DDR and P8DR, except that software can turn on their MOS pull-ups by clearing the P8DDR bit to "0" and setting the P8DR bit to "1."

Pin P8$_4$: This pin has the same functions in all modes. It can be used for general-purpose input or output, for output of clock-synchronized serial transmit data (CTxD), or for $\overline{\text{IRQ}}_3$ input. When used for CTxD output, this pin is unaffected by the values in P8DDR and P8DR, and its MOS pull-up is automatically turned off. When this pin is used for $\overline{\text{IRQ}}_3$ input, P8$_4$DDR should normally be cleared to "0," so that the value in P8DR will not generate interrupts.

Pin P8$_5$: This pin has the same functions in all modes. It can be used for general-purpose input or output, for input of clock-synchronized serial receive data (CRxD), or for $\overline{\text{IRQ}}_4$ input. When used for CRxD input, this pin is unaffected by the values in P8DDR and P8DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1." When this pin is used for $\overline{\text{IRQ}}_4$ input, P8$_5$DDR should normally be cleared to "0," so that the value in P8DR will not generate interrupts.

Pin P8$_6$: This pin has the same functions in all modes. It can be used for general-purpose input or output, for serial clock input or output (CSCK), or for $\overline{\text{IRQ}}_5$ input. When this pin is used for $\overline{\text{IRQ}}_5$ input, P8$_6$DDR should normally be cleared to "0," so that the value in P8DR will not generate interrupts.

When used for CSCK input or output, this pin is unaffected by the values in P8DDR and P8DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1." For CSCK usage, the MOS pull-up should be turned off.

Reset: A reset clears bits P8$_6$DDR to P8$_1$DDR to “0” and clears the DPME bit, serial control bits, and interrupt enable bits to “0,” making P8$_6$ to P8$_1$ into input port pins with the MOS pull-ups off.

In the expanded modes (modes 1 and 2), P8$_0$DDR is initialized to “1” and the P8$_0$ pin is used for E clock output. In the single-chip mode (mode 3), P8$_0$DDR is initialized to “0” and the P8$_0$ pin is used for port input.

Hardware Standby Mode: All pins are placed in the high-impedance state with the MOS pull-ups off.

Software Standby Mode: In the software standby mode, the serial control register is initialized, but the DPME bit, the interrupt enable register, P8DDR, and P8DR remain in their previous states. Pins that were being used for serial communication revert to general-purpose input or output, depending on the value in P8DDR. Other pins remain in their previous state. Output pins output the values in P8DR. E clock output is Low.

Figures 5-14 to 5-19 show schematic diagrams of port 8.

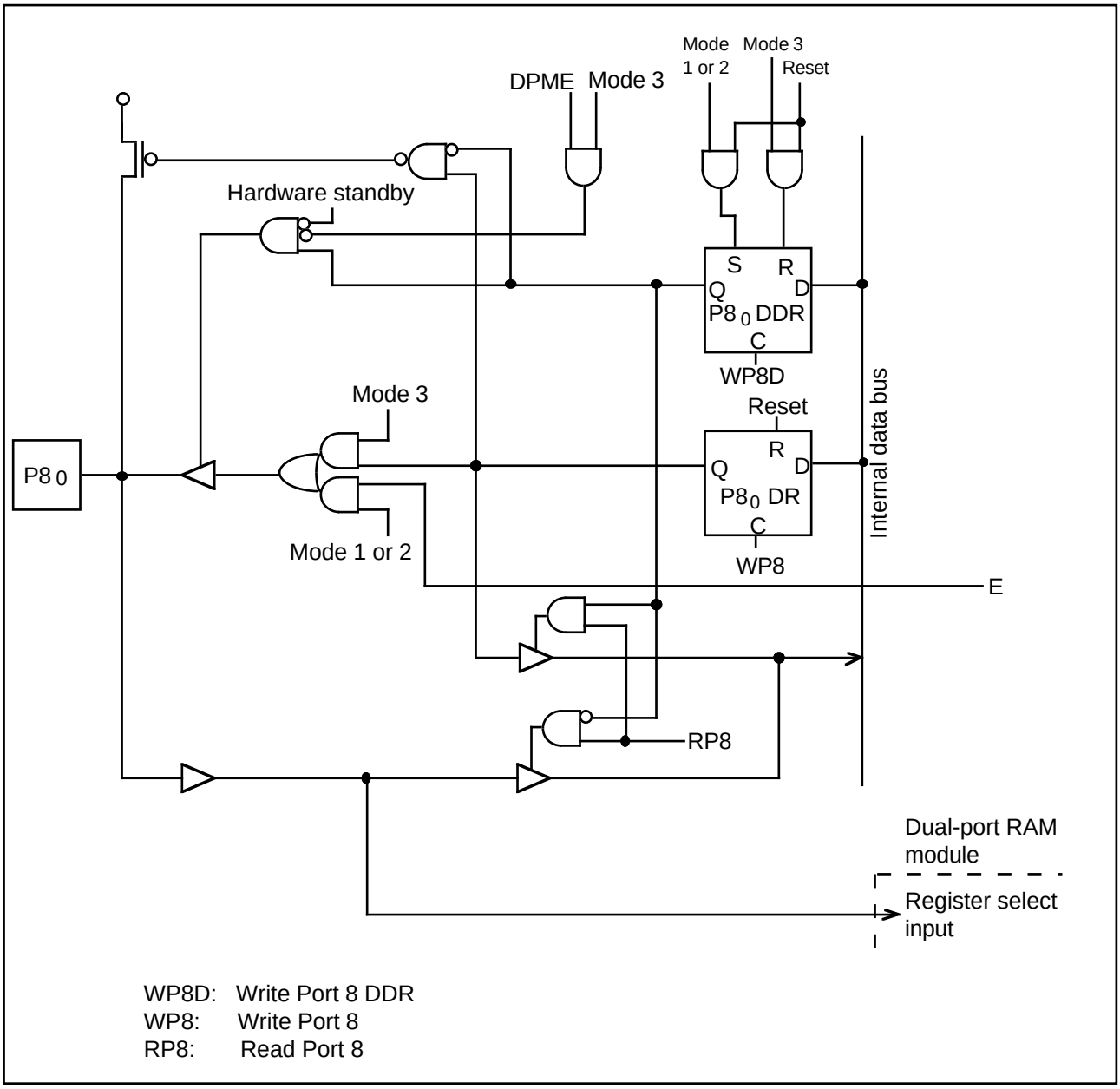


Figure 5-14. Port 8 Schematic Diagram (Pin P8$_0$)

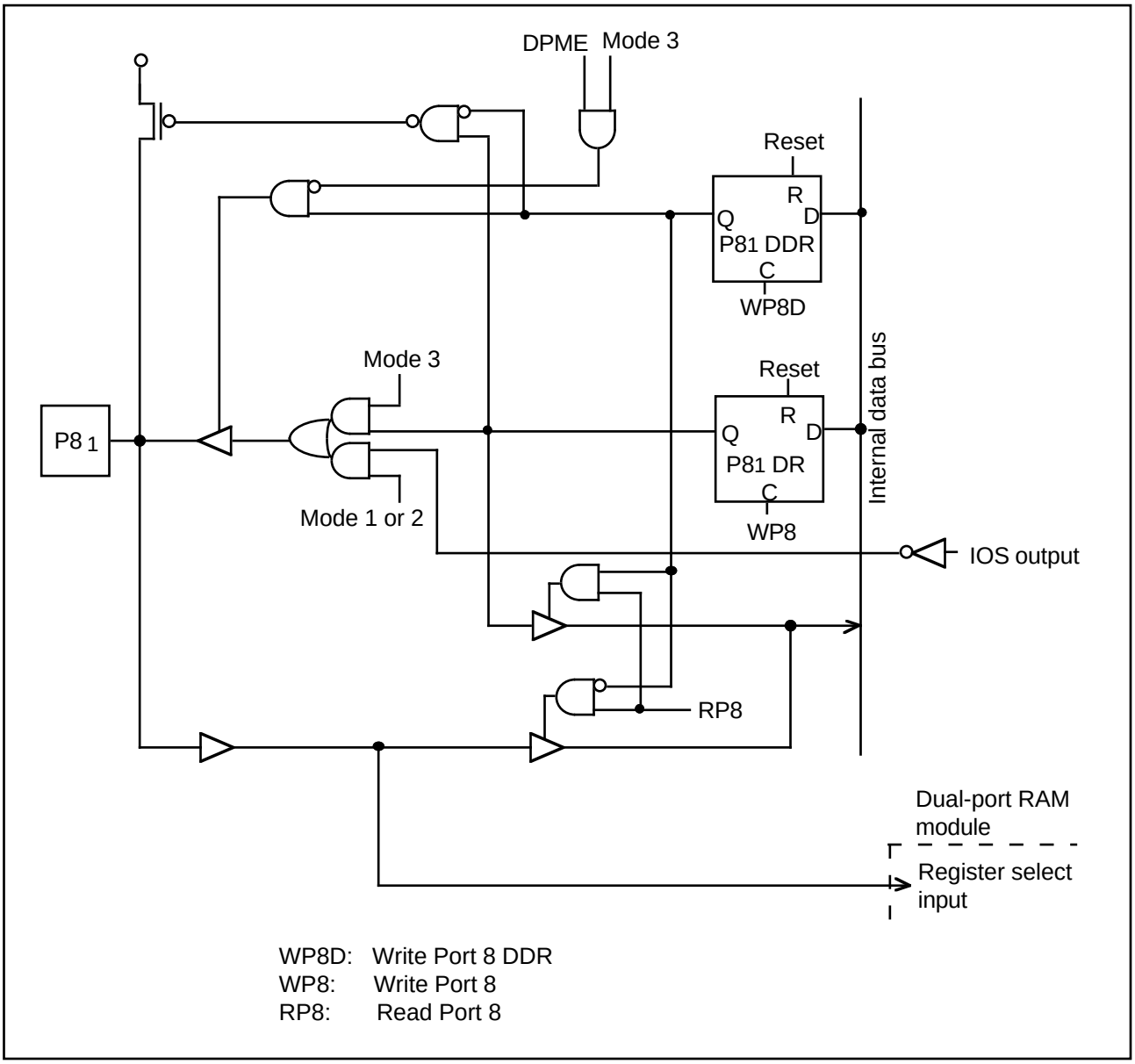


Figure 5-15. Port 8 Schematic Diagram (Pin P81)

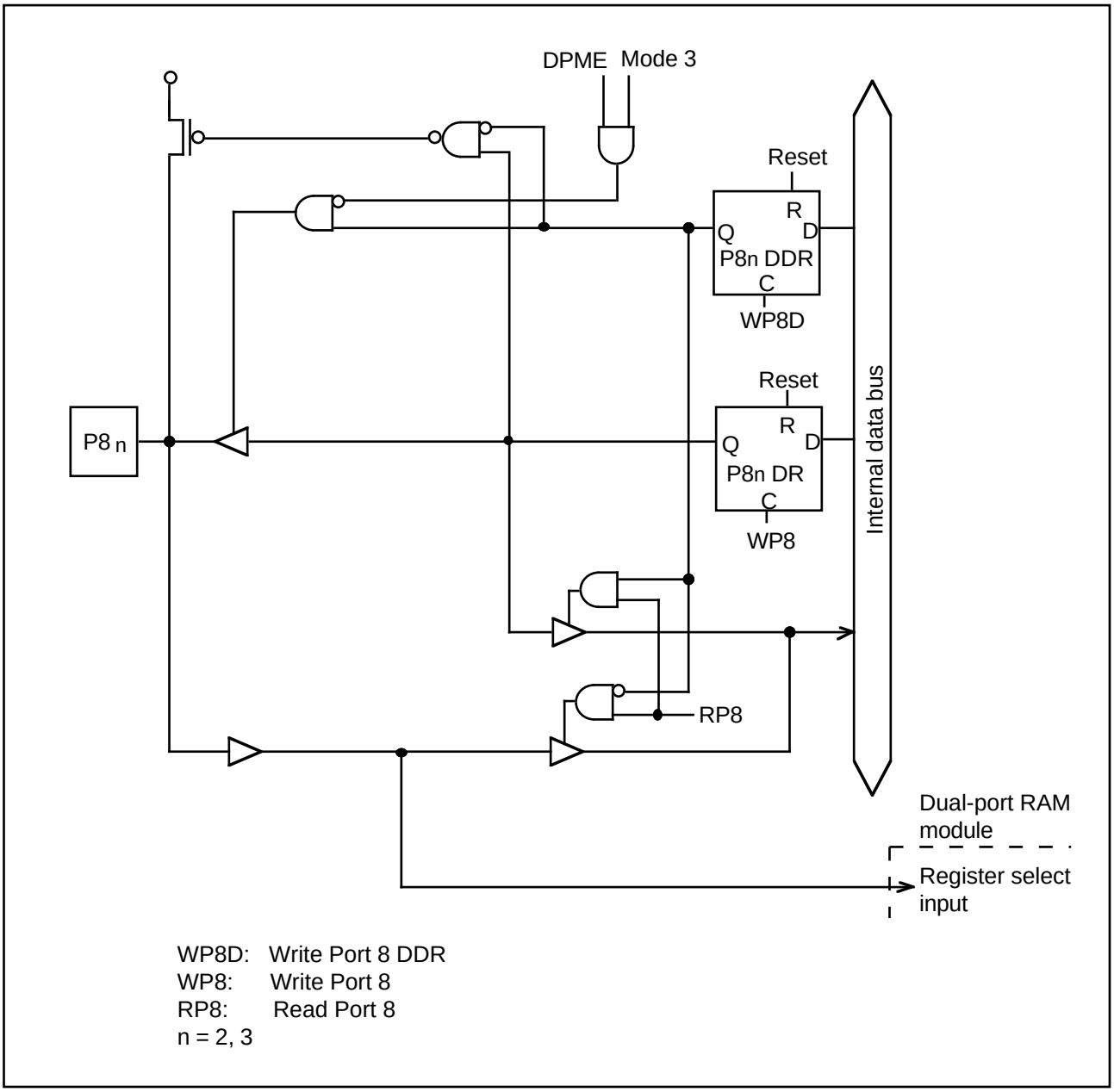


Figure 5-16. Port 8 Schematic Diagram (Pins P8$_2$ and P8$_3$)

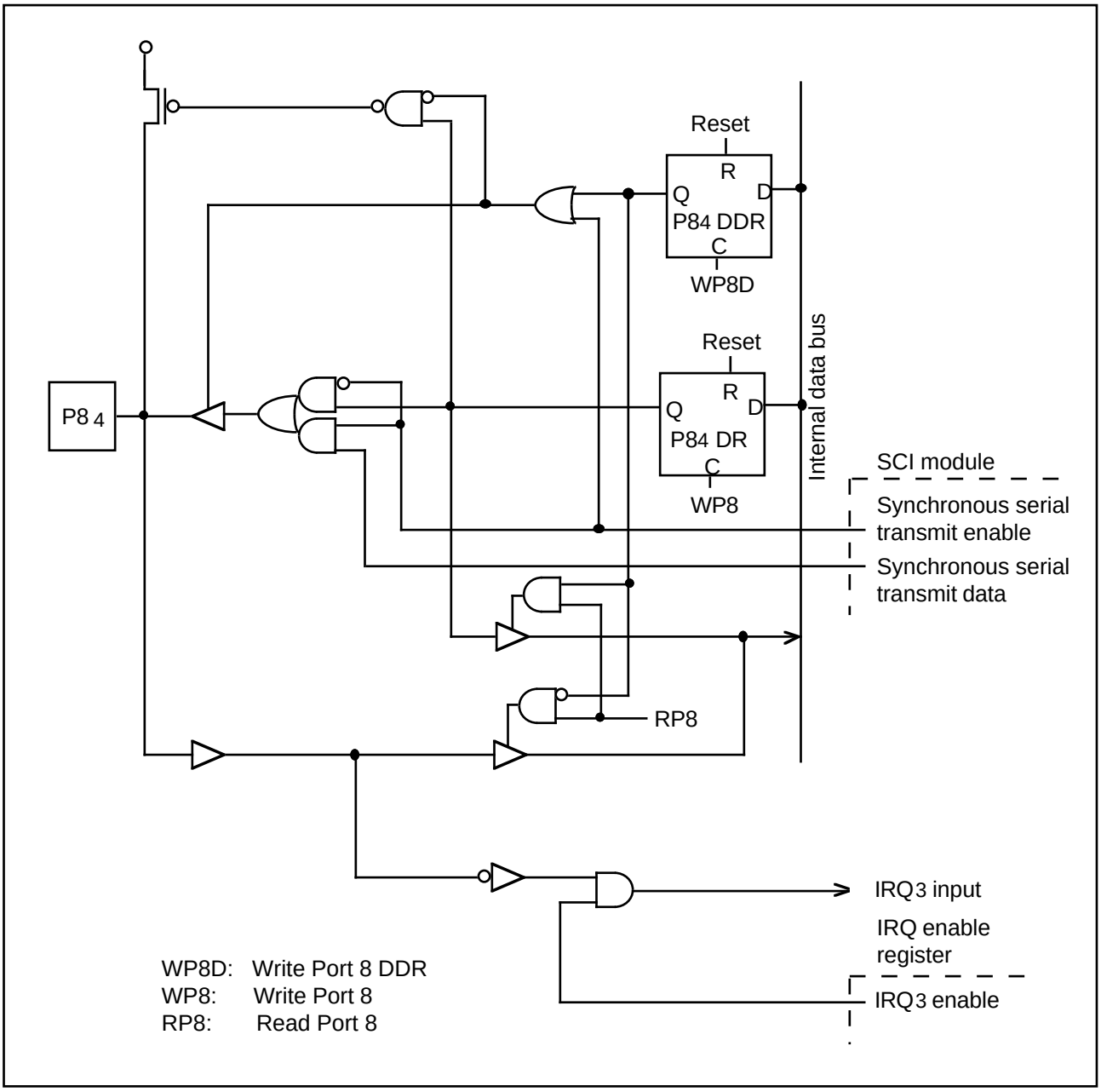


Figure 5-17. Port 8 Schematic Diagram (Pin P84)

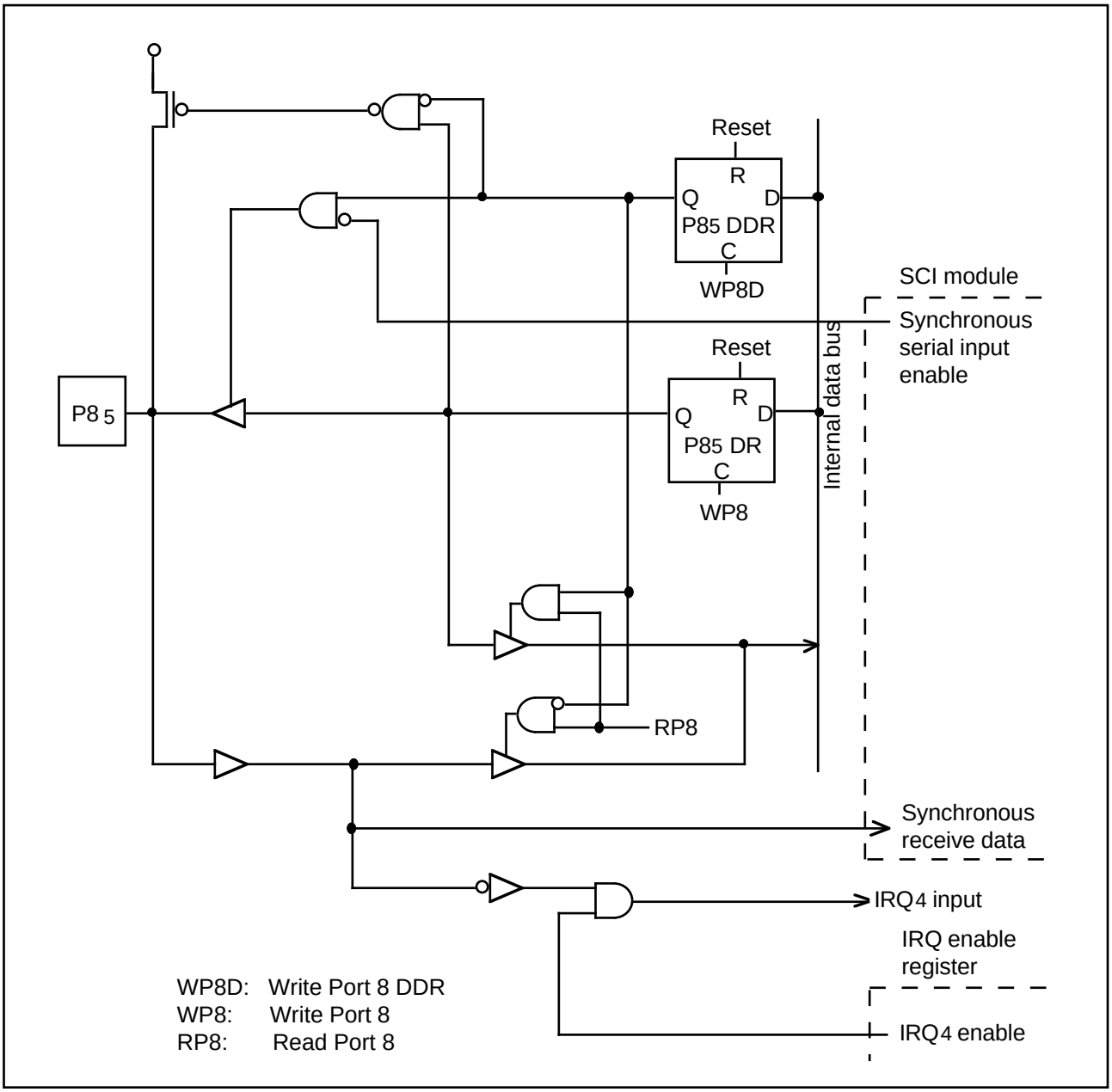


Figure 5-18. Port 8 Schematic Diagram (Pin P85)

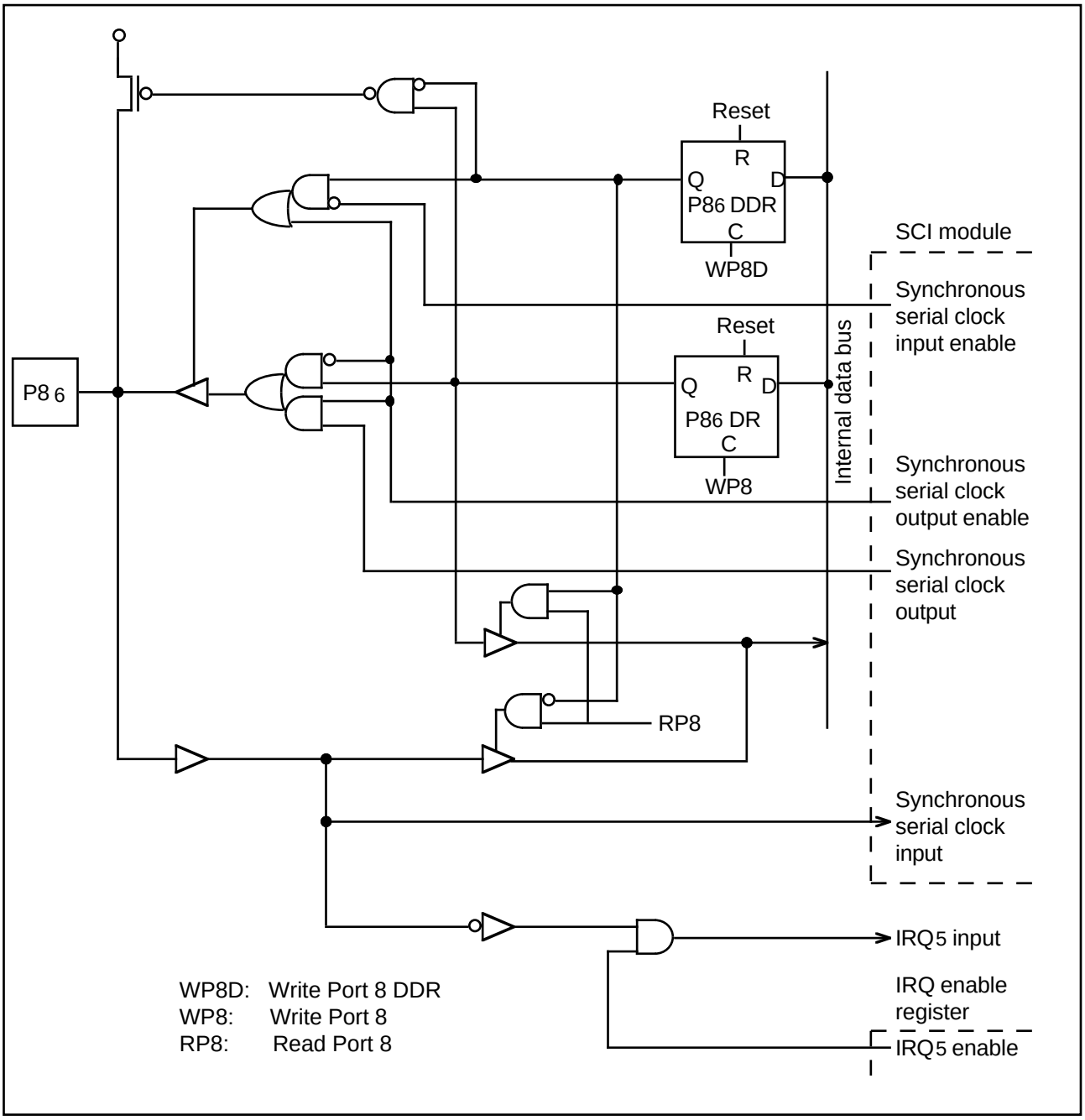


Figure 5-19. Port 8 Schematic Diagram (Pin P86)

5.10 Port 9

Port 9 is an 8-bit input/output port that also provides pins for interrupt input ($\overline{IRQ_0}$ to $\overline{IRQ_2}$), A/D trigger input, system clock (Ø) output, bus control signals (in the expanded modes), and dual-port RAM interface control signals (in the single-chip mode).

Pins $P9_7$ to $P9_3$ have different functions in different modes. Pins $P9_2$ to $P9_0$ have the same functions in all modes. Table 5-18 lists the pin functions.

Table 5-18. Port 9 Pin Functions

		Single-chip mode	
Pin	**Expanded modes**	**DPME = 0**	**DPME = 1**
$P9_0$	$P9_0$ input/output , $\overline{IRQ_2}$ input, and $\overline{ADTRG}$ input (simultaneously)		
$P9_1$	$P9_1$ input/output and $\overline{IRQ_1}$ input (simultaneously)		
$P9_2$	$P9_2$ input/output and $\overline{IRQ_0}$ input (simultaneously)		
$P9_3$	$\overline{RD}$ output	$P9_3$ input/output	$\overline{CS}$ input
$P9_4$	$\overline{WR}$ output	$P9_4$ input/output	$\overline{OE}$ input
$P9_5$	$\overline{AS}$ output	$P9_5$ input/output	$\overline{RDY}$ output
$P9_6$	Ø output	$P9_6$ input or Ø output	$P9_6$ input or Ø output
$P9_7$	$\overline{WAIT}$ input	$P9_7$ input/output	$\overline{WE}$ input

Pins of port 9 can drive a single TTL load and a 90pF capacitive load when they are used as output pins. When used as input pins, they have programmable MOS pull-ups.

Table 5-19 details the port 9 registers.

Table 5-19. Port 9 Registers

Name	**Abbreviation**	**Read/Write**	**Initial value**	**Address**
Port 9 data direction register	P9DDR	W	H’40 (modes 1 and 2) H'00 (mode 3	H’FFC0
Port 9 data register	P9DR	R/W	H’00	H’FFC1

Port 9 Data Direction Register (P9DDR)—H'FFC0

Bit	7	6	5	4	3	2	1	0
	$P9_7$DDR	$P9_6$DDR	$P9_5$DDR	$P9_4$DDR	$P9_3$DDR	$P9_2$DDR	$P9_1$DDR	$P9_0$DDR
Modes 1 and 2								
Initial value	0	1	0	0	0	0	0	0
Read/Write	W	—	W	W	W	W	W	W
Mode 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P9DDR is an 8-bit register that selects the direction of each pin in port 9. A pin functions as an output pin if the corresponding bit in P9DDR is set to "1," and as in input pin if the bit is cleared to "0."

Port 9 Data Register (P9DR)—H'FFC1

Bit	7	6	5	4	3	2	1	0
	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	$P9_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P9DR is an 8-bit register containing the data for pins $P9_7$ to $P9_0$. When the CPU reads P9DR, for output pins (P9DDR = "1") it reads the value in the P9DR latch, but for input pins (P9DDR = "0"), it obtains the logic level directly from the pin, bypassing the P9DR latch. This also applies to pins used for interrupt input, A/D trigger input, clock output, and control signal input or output.

MOS Pull-Ups: Are available for input pins, including pins used for input of interrupt request signals, the A/D trigger signal, and control signals. Software can turn the MOS pull-up on by writing a "1" in P9DR, and turn it off by writing a "0."

Pins $P9_0$, $P9_1$, and $P9_2$: Can be used for general-purpose input or output, interrupt request input, or A/D trigger input. See Table 5-18. If a pin is used for interrupt or A/D trigger input, its data direction bit should be cleared to "0," so that the output from P9DR will not generate an interrupt request or A/D trigger signal.

Pins $P9_3$ and $P9_4$: In modes 1 and 2 (the expanded modes), these pins are used for output of the $\overline{RD}$ and $\overline{WR}$ bus control signals. They are unaffected by the values in P9DDR and P9DR, and their

MOS pull-ups are automatically turned off.

In mode 3 (single-chip mode) with the dual-port RAM disabled (DPME = "0"), these pins can be used for general-purpose input or output.

In slave mode (mode 3 with DPME = "1"), these pins are used for input of the $\overline{CS}$ and $\overline{OE}$ dual-port RAM interface control signals. They are unaffected by the values in P9DDR and P9DR, except that software can turn on their MOS pull-ups by clearing their data direction bits to "0" and setting their data bits to "1."

Pin P9$_5$: In modes 1 and 2 and slave mode, this pin is used for output of the $\overline{AS}$ bus control signal or $\overline{RDY}$ dual-port RAM interface control signal. It is unaffected by the values in P9DDR and P9DR, and its MOS pull-up is automatically turned off.

In mode 3 with the dual-port RAM disabled (DPME = "0"), this pin can be used for general-purpose input or output.

Pin P9$_6$: In modes 1 and 2, this pin is used for system clock (Ø) output. Its MOS pull-up is automatically turned off.

In mode 3, this pin is used for general-purpose input if P9$_6$DDR is cleared to "0," or system clock output if P9$_6$DDR is set to "1."

Pin P9$_7$: In modes 1 and 2 and slave mode, this pin is used for input of the $\overline{WAIT}$ bus control signal or $\overline{WE}$ dual-port RAM interface control signal. It is unaffected by the values in P9DDR and P9DR, except that software can turn on its MOS pull-up by clearing its data direction bit to "0" and setting its data bit to "1."

In mode 3 (single-chip mode) with the dual-port RAM disabled (DPME = "0"), this pin can be used for general-purpose input or output.

Reset: In the single-chip mode (mode 3), a reset initializes all pins of port 9 to the general-purpose input function with the MOS pull-ups off. In the expanded modes (modes 1 and 2), P9$_0$ to P9$_2$ are initialized as input port pins, and P9$_3$ to P9$_7$ are initialized to their bus control and system clock output functions.

Hardware Standby Mode: All pins are placed in the high-impedance state with their MOS pull-ups off.

Software Standby Mode: All pins remain in their previous state. For $\overline{RD}$, $\overline{WR}$, $\overline{AS}$, and Ø this means the High output state.

Figures 5-20 to 5-25 show schematic diagrams of port 9.

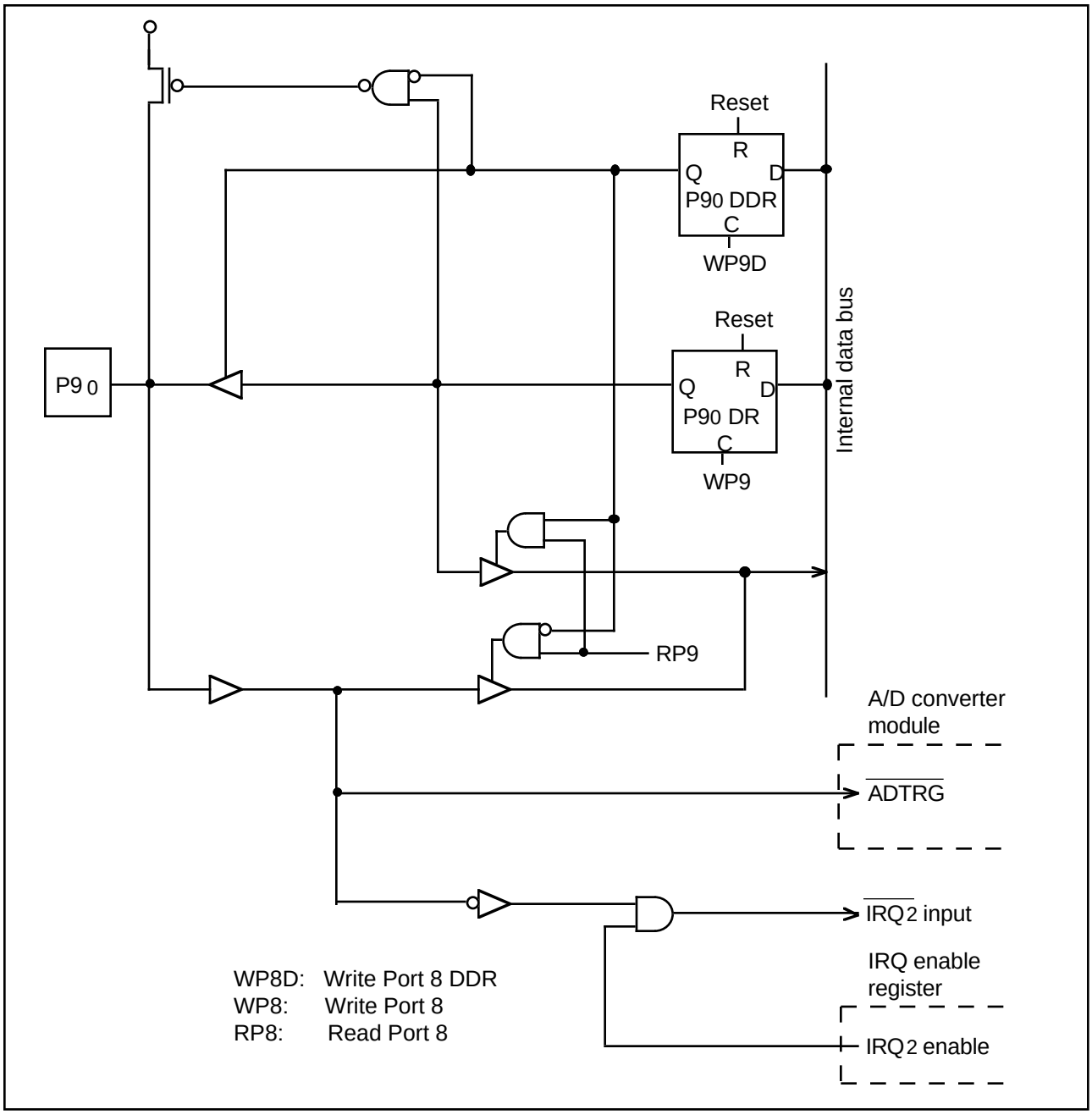


Figure 5-20. Port 9 Schematic Diagram (Pin P9$_0$)

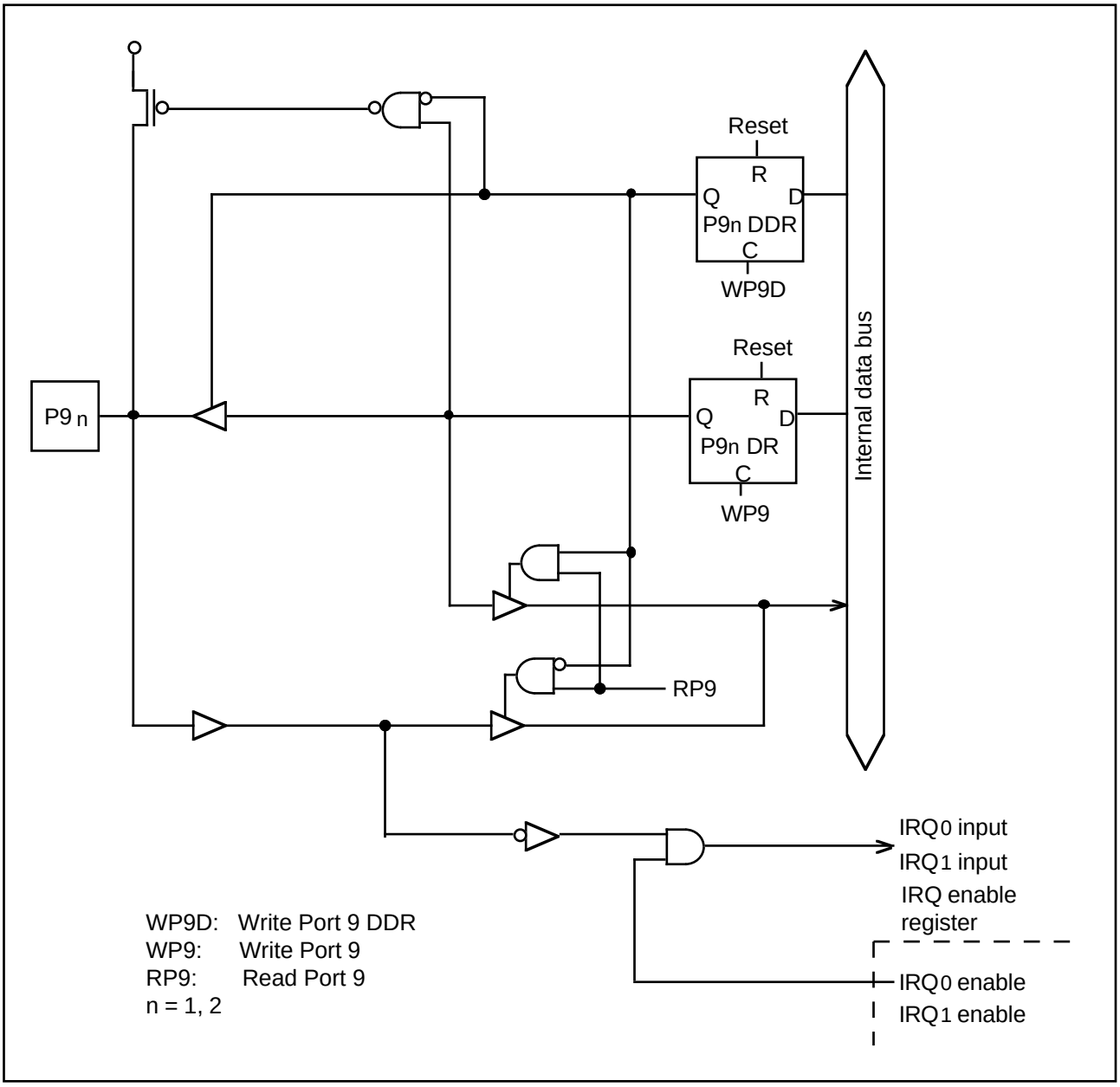


Figure 5-21. Port 9 Schematic Diagram (Pins P9$_1$ to P9$_2$)

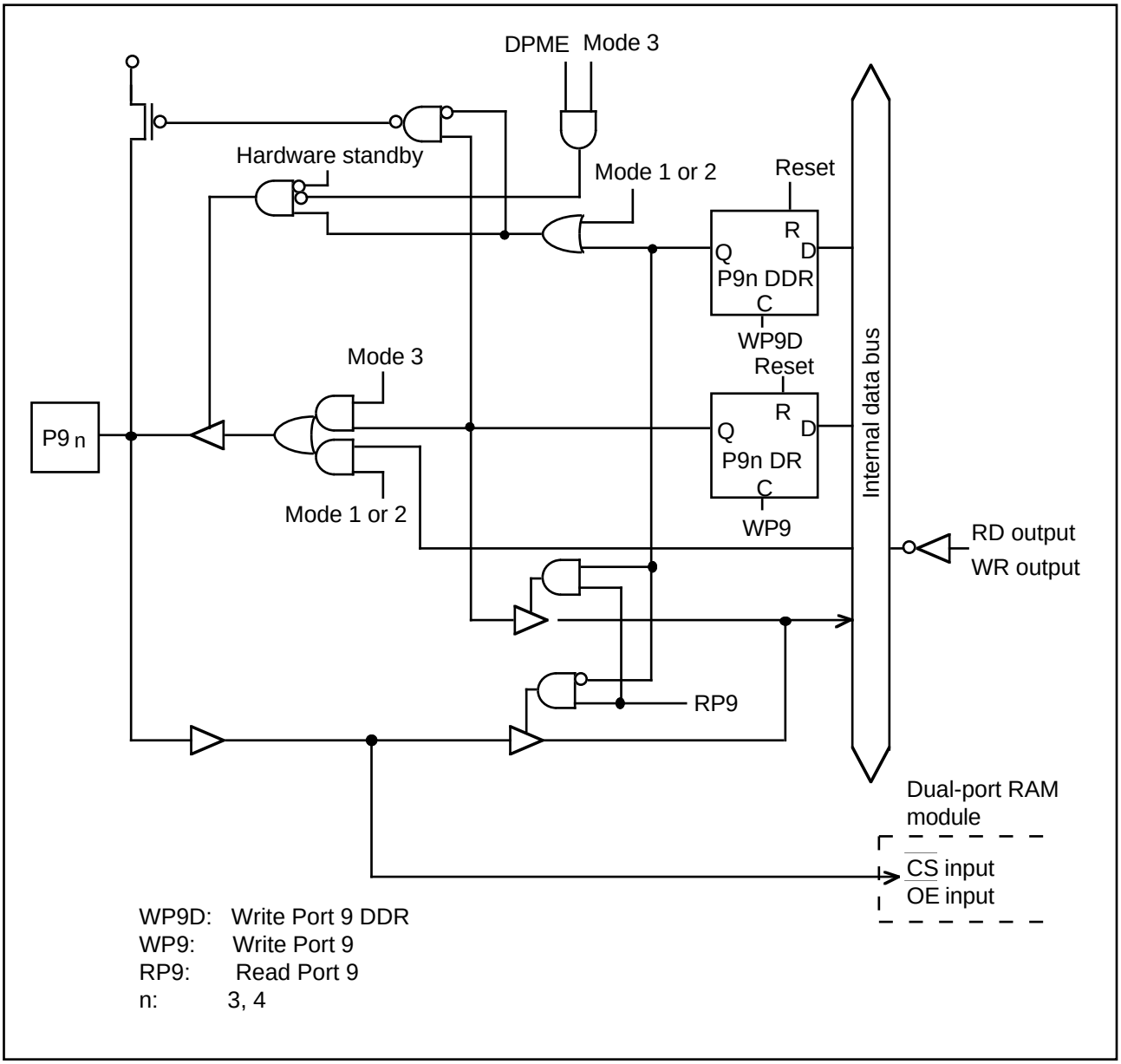


Figure 5-22. Port 9 Schematic Diagram (Pins P9$_3$ and P9$_4$)

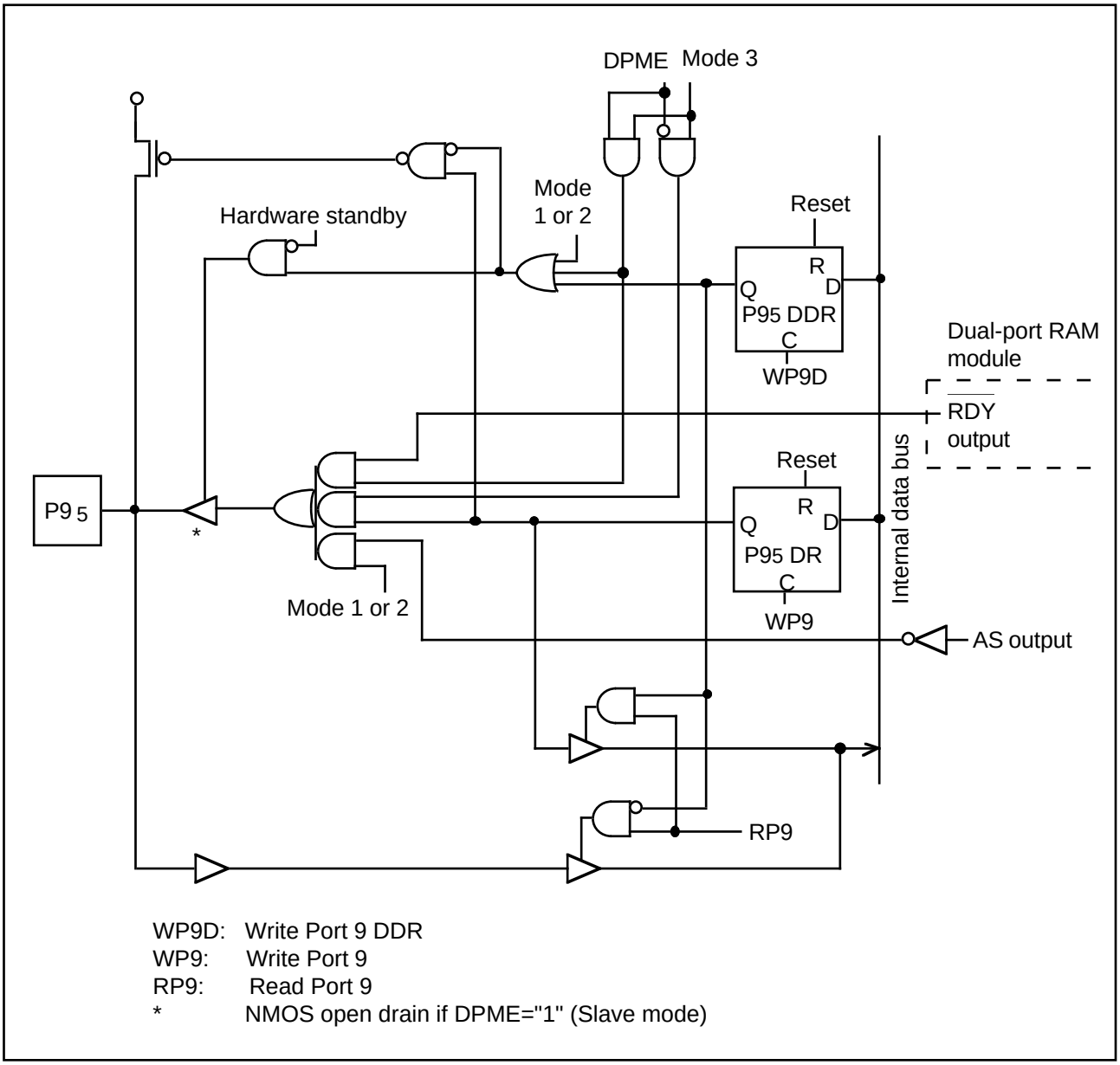


Figure 5-23. Port 9 Schematic Diagram (Pin P9$_5$)

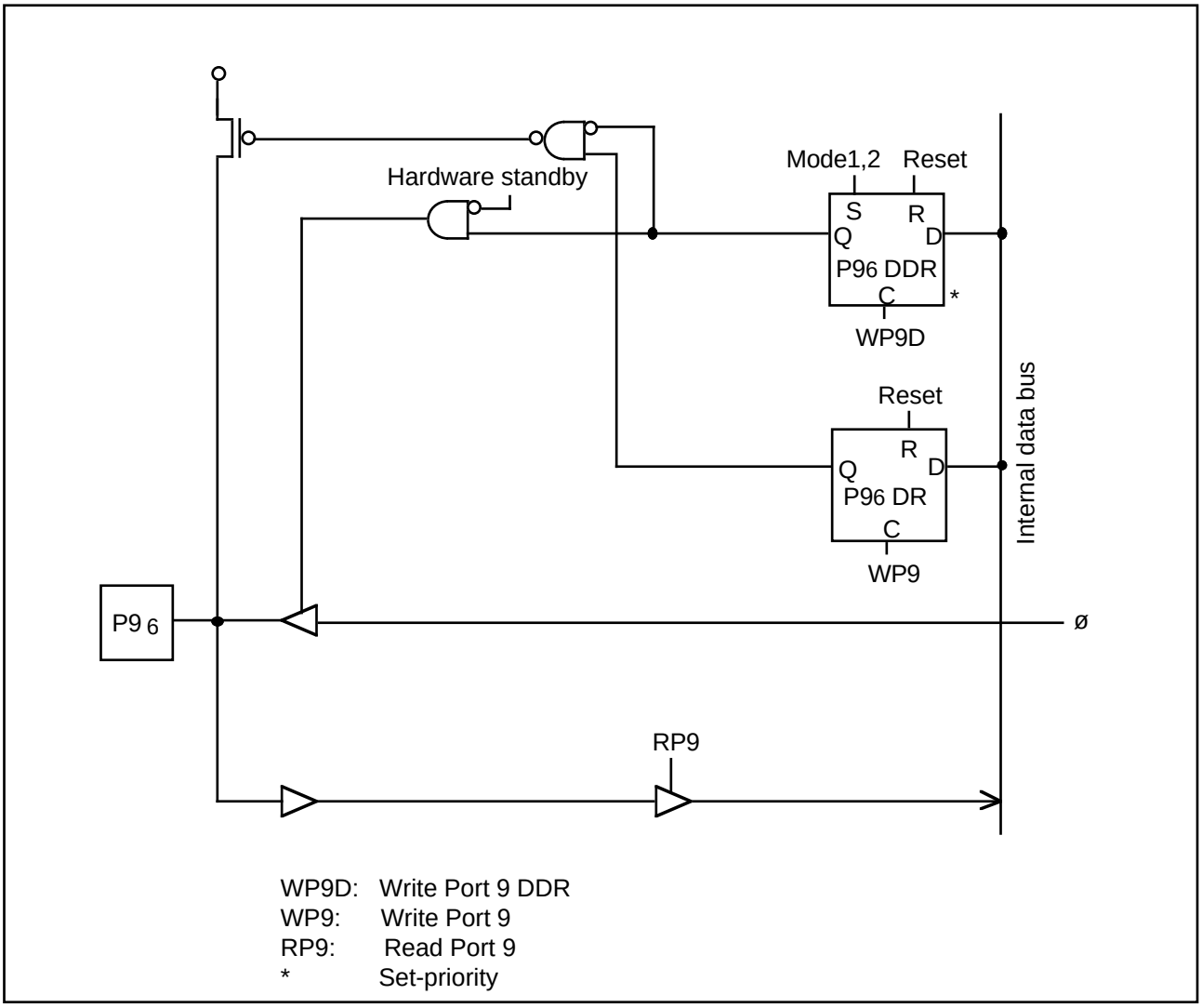


Figure 5-24. Port 9 Schematic Diagram (Pin P96)

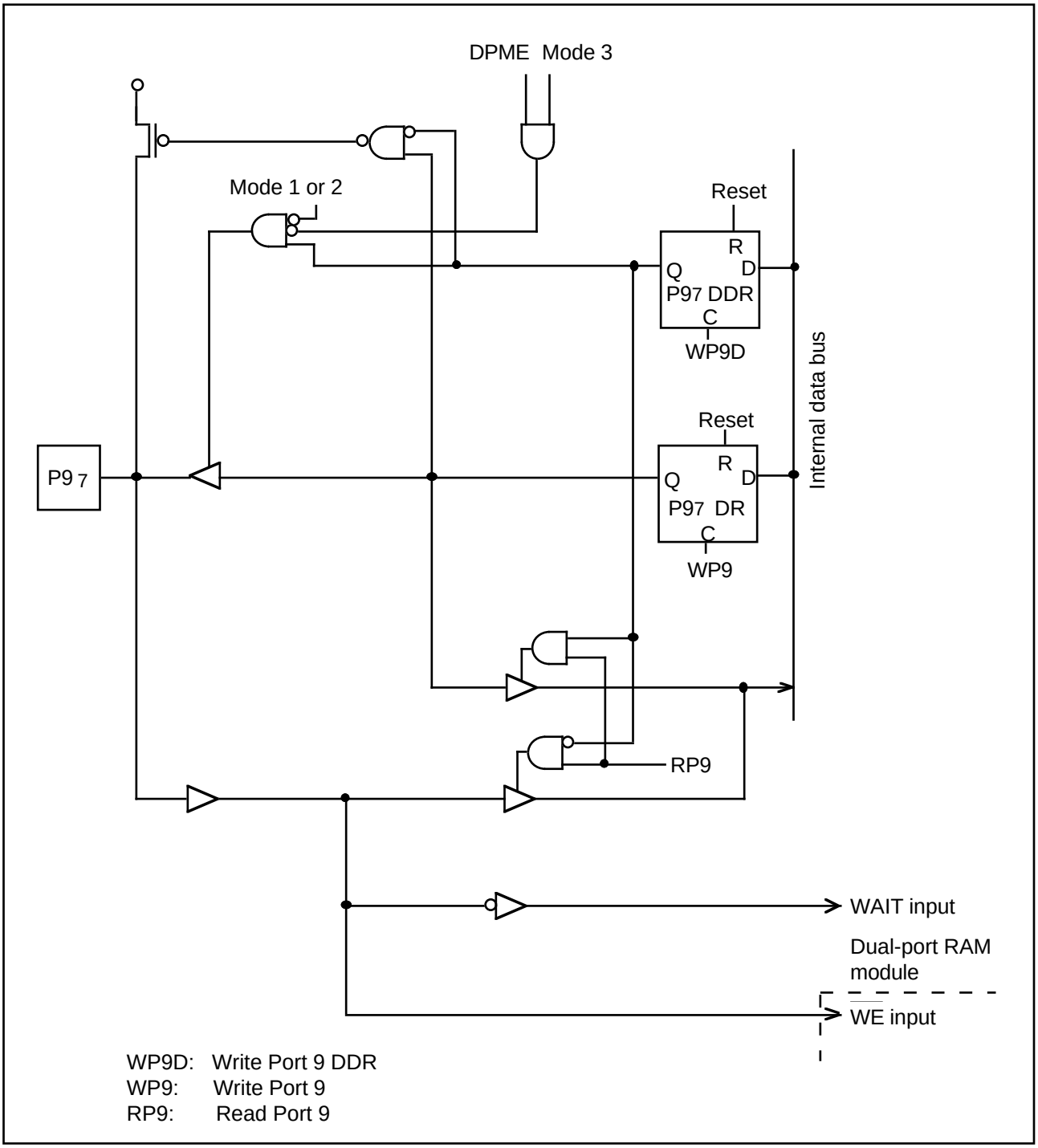


Figure 5-25. Port 9 Schematic Diagram (Pin P9$_7$)

Section 6. 16-Bit Free-Running Timer

6.1 Overview

The H8/330 has an on-chip 16-bit free-running timer (FRT) module that uses a 16-bit free-running counter as a time base. Applications of the FRT module include rectangular-wave output (up to two independent waveforms), input pulse width measurement, and measurement of external clock periods.

6.1.1 Features

The features of the free-running timer module are listed below.

- Selection of four clock sources
  The free-running counter can be driven by an internal clock source (Ø/2, Ø/8, or Ø/32), or an external clock input (enabling use as an external event counter).
- Two independent comparators
  Each comparator can generate an independent waveform.
- Four input capture channels
  The current count can be captured on the rising or falling edge (selectable) of an input signal.
  The four input capture registers can be used separately, or in a buffer mode.
- Counter can be cleared under program control
  The free-running counters can be cleared on compare-match A.
- Seven independent interrupts
  Compare-match A and B, input capture A to D, and overflow interrupts are requested independently.

6.1.2 Block Diagram

Figure 6-1 shows a block diagram of the free-running timer.

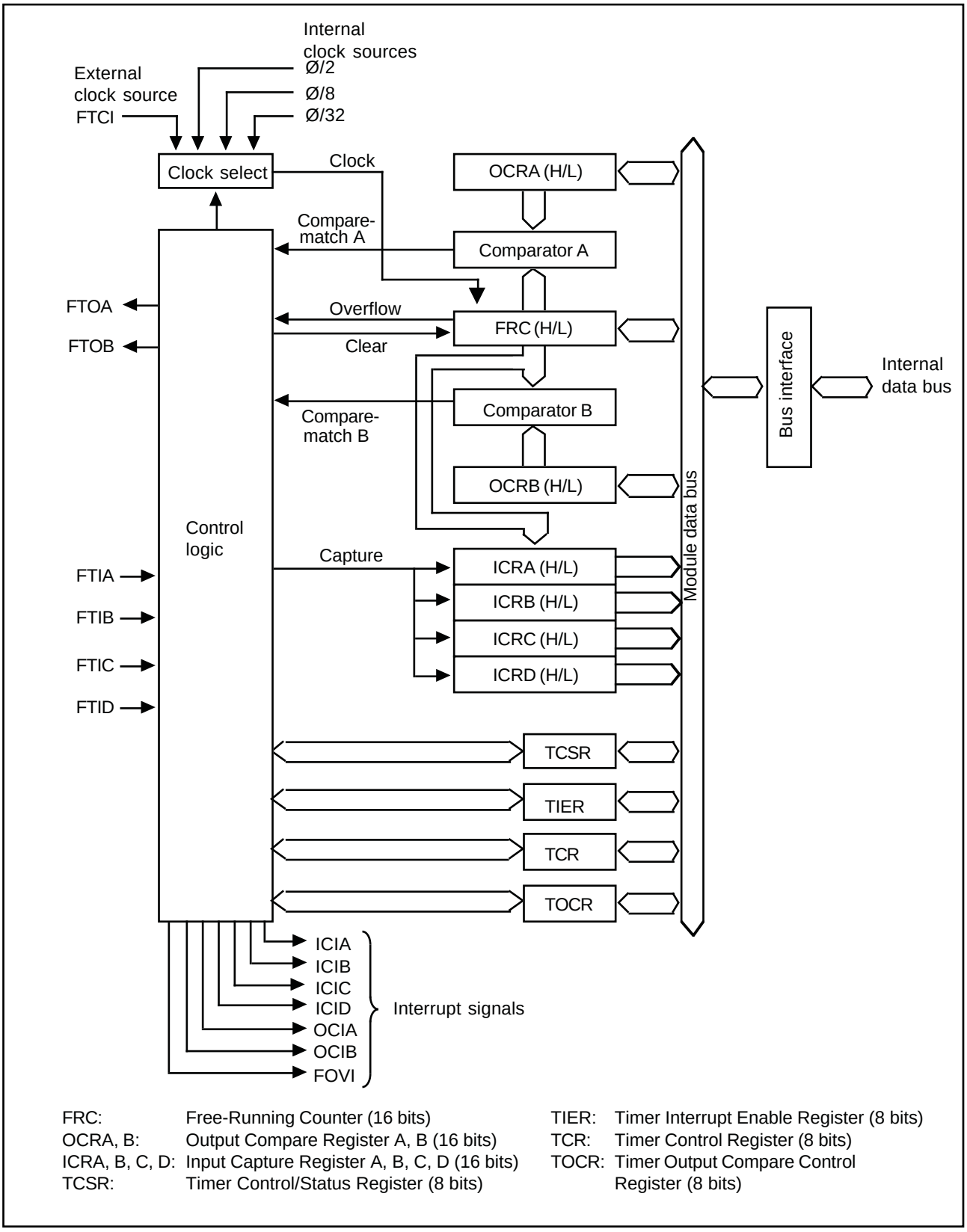


Figure 6-1. Block Diagram of 16-Bit Free-Running Timer

6.1.3 Input and Output Pins

Table 6-1 lists the input and output pins of the free-running timer module.

Table 6-1. Input and Output Pins of Free-Running Timer Module

Name	Abbreviation	I/O	Function
Counter clock input	FTCI	Input	Input of external free-running counter clock signal
Output compare A	FTOA	Output	Output controlled by comparator A
Output compare B	FTOB	Output	Output controlled by comparator B
Input capture A	FTIA	Input	Trigger for capturing current count into input capture register A
Input capture B	FTIB	Input	Trigger for capturing current count into input capture register B
Input capture C	FTIC	Input	Trigger for capturing current count into input capture register C
Input capture D	FTID	Input	Trigger for capturing current count into input capture register D

6.1.4 Register Configuration

Table 6-2 lists the registers of the free-running timer module.

Table 6-2. Register Configuration

Name	Abbreviation	R/W	Initial value	Address
Timer interrupt enable register	TIER	R/W	H’01	H’FF90
Timer control/status register	TCSR	R/(W)*1	H’00	H’FF91
Free-running counter (High)	FRC (H)	R/W	H’00	H’FF92
Free-running counter (Low)	FRC (L)	R/W	H’00	H’FF93
Output compare register A/B (High)*2	OCRA/B (H)	R/W	H’FF	H’FF94
Output compare register A/B (Low)*2	OCRA/B (L)	R/W	H’FF	H’FF95
Timer control register	TCR	R/W	H’00	H’FF96
Timer output compare control register	TOCR	R/W	H’E0	H’FF97
Input capture register A (High)	ICRA (H)	R	H’00	H’FF98
Input capture register A (Low)	ICRA (L)	R	H’00	H’FF99

Notes:

*1 Software can write a “0” to clear bits 7 to 1, but cannot write a “1” in these bits.

*2 OCRA and OCRB share the same addresses. Access is controlled by the OCRS bit in TOCR.

Table 6-2. Register Configuration (cont.)

Name	Abbreviation	R/W	Initial value	Address
Input capture register B (High)	ICRB (H)	R	H’00	H’FF9A
Input capture register B (Low)	ICRB (L)	R	H’00	H’FF9B
Input capture register C (High)	ICRC (H)	R	H’00	H’FF9C
Input capture register C (Low)	ICRC (L)	R	H’00	H’FF9D
Input capture register D (High)	ICRD (H)	R	H’00	H’FF9E
Input capture register D (Low)	ICRD (L)	R	H’00	H’FF9F

6.2 Register Descriptions

6.2.1 Free-Running Counter (FRC) – H’FF92

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/ Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The FRC is a 16-bit readable/writable up-counter that increments on an internal pulse generated from a clock source. The clock source is selected by the clock select 1 and 0 bits (CKS1 and CKS0) of the timer control register (TCR).

When the FRC overflows from H’FFFF to H’0000, the overflow flag (OVF) in the timer control/status register (TCSR) is set to “1.”

Because the FRC is a 16-bit register, a temporary register (TEMP) is used when the FRC is written or read. See section 6.3, “CPU Interface” for details.

The FRC is initialized to H’0000 at a reset and in the standby modes. It can also be cleared by compare-match A.

6.2.2 Output Compare Registers A and B (OCRA and OCRB) – H’FF94

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Read/ Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

OCRA and OCRB are 16-bit readable/writable registers, the contents of which are continually compared with the value in the FRC. When a match is detected, the corresponding output compare flag (OCFA or OCFB) is set in the timer control/status register (TCSR).

In addition, if the output enable bit (OEA or OEB) in the timer output compare control register (TOCR) is set to “1,” when the output compare register and FRC values match, the logic level selected by the output level bit (OLVLA or OLVLB) in the TOCR is output at the output compare pin (FTOA or FTOB).

OCRA and OCRB share the same address. They are differentiated by the OCRS bit in the TOCR. A temporary register (TEMP) is used for write access, as explained in section 6.3, "CPU Interface."

OCRA and OCRB are initialized to H’FFFF at a reset and in the standby modes.

6.2.3 Input Capture Registers A to D (ICRA to ICRD) – H’FF98, H’FF9A, H’FF9C, H’FF9E

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/ Write	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Each input capture register is a 16-bit read-only register.

When the rising or falling edge of the signal at an input capture pin (FTIA to FTID) is detected, the current value of the FRC is copied to the corresponding input capture register (ICRA to ICRD). At the same time, the corresponding input capture flag (ICFA to ICFD) in the timer control/status register (TCSR) is set to “1.” The input capture edge is selected by the input edge select bits (IEDGA to IEDGD) in the timer interrupt enable register (TIER).

Input capture can be buffered by using the input capture registers in pairs. When the BUFEA bit in the timer control register (TCR) is set to "1," ICRC is used as a buffer register for ICRA as shown in Figure 6-2. When an FTIA input is received, the old ICRA contents are moved into ICRC, and the new FRC count is copied into ICRA.

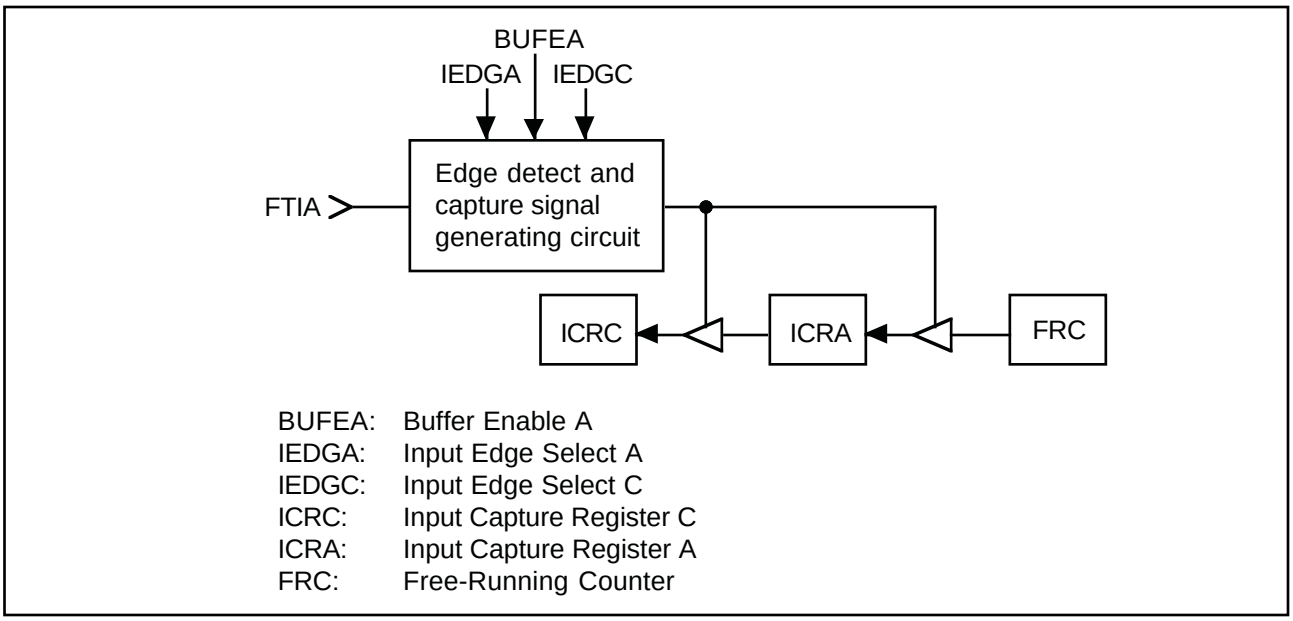


Figure 6-2. Input Capture Buffering

Similarly, when the BUFEB bit in TIER is set to "1," ICRD is used as a buffer register for ICRB.

When input capture is buffered, if the two input edge bits are set to different values (IEDGA ≠ IEDGC or IEDGB ≠ IEDGD), then input capture is triggered on both the rising and falling edges of the FTIA or FTIB input signal. If the two input edge bits are set to the same value (IEDGA = IEDGC or IEDGB = IEDGD), then input capture is triggered on only one edge.

Because the input capture registers are 16-bit registers, a temporary register (TEMP) is used when they are read. See Section 6.3, "CPU Interface" for details.

To ensure input capture, the width of the input capture pulse (FTIA, FTIB, FTIC, FTID) should be at least 1.5 system clock periods (1.5·Ø). When triggering is enabled on both edges, the input capture pulse width should be at least 2.5 system clock periods.

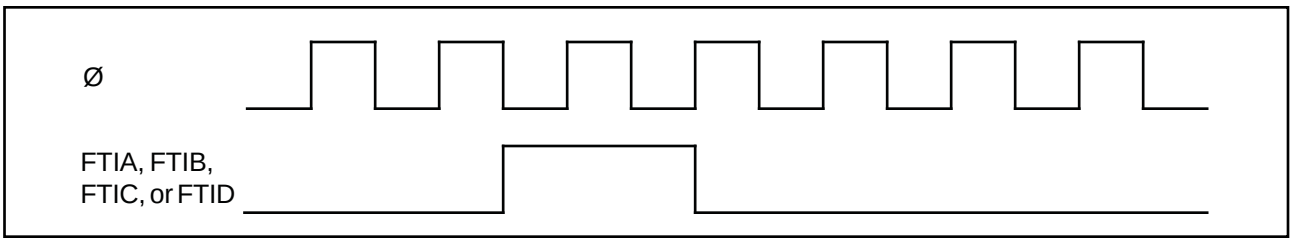


Figure 6-3. Minimum Input Capture Pulse Width

The input capture registers are initialized to H’0000 at a reset and in the standby modes.

Note: When input capture is detected, the FRC value is transferred to the input capture register even if the input capture flag is already set.

6.2.4 Timer Interrupt Enable Register (TIER) – H’FF90

Bit	7	6	5	4	3	2	1	0
	ICIAE	ICIBE	ICICE	ICIDE	OCIAE	OCIBE	OVIE	—
Initial value	0	0	0	0	0	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—

The TIER is an 8-bit readable/writable register that enables and disables interrupts.

The TIER is initialized to H’01 (all interrupts disabled) at a reset and in the standby modes.

Bit 7 – Input Capture Interrupt A Enable (ICIAE): This bit selects whether to request input capture interrupt A (ICIA) when input capture flag A (ICFA) in the timer status/control register (TCSR) is set to “1.”

Bit 7

ICIAE	Description	
0	Input capture interrupt request A (ICIA) is disabled.	(Initial value)
1	Input capture interrupt request A (ICIA) is enabled.	

Bit 6 – Input Capture Interrupt B Enable (ICIBE): This bit selects whether to request input capture interrupt B (ICIB) when input capture flag B (ICFB) in the timer status/control register (TCSR) is set to “1.”

Bit 6

ICIBE	Description	
0	Input capture interrupt request B (ICIB) is disabled.	(Initial value)
1	Input capture interrupt request B (ICIB) is enabled.	

Bit 5 – Input Capture Interrupt C Enable (ICICE): This bit selects whether to request input capture interrupt C (ICIC) when input capture flag C (ICFC) in the timer status/control register (TCSR) is set to “1.”

Bit 5

ICICE	Description	
0	Input capture interrupt request C (ICIC) is disabled.	(Initial value)
1	Input capture interrupt request C (ICIC) is enabled.	

Bit 4 – Input Capture Interrupt D Enable (ICIDE): This bit selects whether to request input capture interrupt D (ICID) when input capture flag D (ICFD) in the timer status/control register (TCSR) is set to “1.”

Bit 4

ICIDE	Description	
0	Input capture interrupt request D (ICID) is disabled.	(Initial value)
1	Input capture interrupt request D (ICID) is enabled.	

Bit 3 – Output Compare Interrupt A Enable (OCIAE): This bit selects whether to request output compare interrupt A (OCIA) when output compare flag A (OCFA) in the timer status/control register (TCSR) is set to “1.”

Bit 3

OCIAE	Description	
0	Output compare interrupt request A (OCIA) is disabled.	(Initial value)
1	Output compare interrupt request A (OCIA) is enabled.	

Bit 2 – Output Compare Interrupt B Enable (OCIBE): This bit selects whether to request output compare interrupt B (OCIB) when output compare flag B (OCFB) in the timer status/control register (TCSR) is set to “1.”

Bit 2

OCIBE	Description	
0	Output compare interrupt request B (OCIB) is disabled.	(Initial value)
1	Output compare interrupt request B (OCIB) is enabled.	

Bit 1 – Timer overflow Interrupt Enable (OVIE): This bit selects whether to request a free-running timer overflow interrupt (FOVI) when the timer overflow flag (OVF) in the timer status/control register (TCSR) is set to “1.”

Bit 1

OVIE	Description	
0	Timer overflow interrupt request (FOVI) is disabled.	(Initial value)
1	Timer overflow interrupt request (FOVI) is enabled.	

Bit 0 – Reserved: This bit cannot be modified and is always read as “1.”

6.2.5 Timer Control/Status Register (TCSR) – H’FF91

Bit	7	6	5	4	3	2	1	0
	ICFA	ICFB	ICFC	ICFD	OCFA	OCFB	OVF	CCLRA
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/W

The TCSR is an 8-bit readable and partially writable* register contains the seven interrupt flags and specifies whether to clear the counter on compare-match A (when the FRC and OCRA values match).

* Software can write a “0” in bits 7 to 1 to clear the flags, but cannot write a “1” in these bits.

The TCSR is initialized to H’00 at a reset and in the standby modes.

Bit 7 – Input Capture Flag A (ICFA): This status bit is set to “1” to flag an input capture A event. If BUFEA = “0,” ICFA indicates that the FRC value has been copied to ICRA. If BUFEA = “1,” ICFA indicates that the old ICRA value has been moved into ICRC and the new FRC value has been copied to ICRA.

ICFA must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 7

ICFA	Description	
0	To clear ICFA, the CPU must read ICFA after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when an FTIA input signal causes the FRC value to be copied to ICRA.	

Bit 6 – Input Capture Flag B (ICFB): This status bit is set to “1” to flag an input capture B event. If BUFEB = “0,” ICFB indicates that the FRC value has been copied to ICRB. If BUFEB = “1,” ICFB indicates that the old ICRB value has been moved into ICRD and the new FRC value has been copied to ICRB.

ICFB must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 6

ICFB	**Description**	
0	To clear ICFB, the CPU must read ICFB after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when an FTIB input signal causes the FRC value to be copied to ICRB.	

Bit 5 – Input Capture Flag C (ICFC): This status bit is set to “1” to flag input of a rising or falling edge of FTIC as selected by the IEDGC bit. When BUFEA = “0,” this indicates capture of the FRC count in ICRC. When BUFEA = “1,” however, the FRC count is not captured, so ICFC becomes simply an external interrupt flag. In other words, the buffer mode frees FTIC for use as a general-purpose interrupt signal (which can be enabled or disabled by the ICICE bit).

ICFC must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 5

ICFC	**Description**	
0	To clear ICFC, the CPU must read ICFC after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when an FTIC input signal is received.	

Bit 4 – Input Capture Flag D (ICFD): This status bit is set to “1” to flag input of a rising or falling edge of FTID as selected by the IEDGD bit. When BUFEB = “0,” this indicates capture of the FRC count in ICRD. When BUFEB = “1,” however, the FRC count is not captured, so ICFD becomes simply an external interrupt flag. In other words, the buffer mode frees FTID for use as a general-purpose interrupt signal (which can be enabled or disabled by the ICIDE bit).

ICFD must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 4

ICFD	Description	
0	To clear ICFD, the CPU must read ICFD after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when an FTID input signal is received.	

Bit 3 – Output Compare Flag A (OCFA): This status flag is set to “1” when the FRC value matches the OCRA value. This flag must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 3

OCFA	Description	
0	To clear OCFA, the CPU must read OCFA after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when FRC = OCRA.	

Bit 2 – Output Compare Flag B (OCFB): This status flag is set to “1” when the FRC value matches the OCRB value. This flag must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 2

OCFB	Description	
0	To clear OCFB, the CPU must read OCFB after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when FRC = OCRB.	

Bit 1 – Timer Overflow Flag (OVF): This status flag is set to “1” when the FRC overflows (changes from H’FFFF to H’0000). This flag must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 1

OVF	Description	
0	To clear OVF, the CPU must read OVF after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when FRC changes from H’FFFF to H’0000.	

Bit 0 – Counter Clear A (CCLRA): This bit selects whether to clear the FRC at compare-match A (when the FRC and OCRA values match).

Bit 0

CCLRA	Description	
0	The FRC is not cleared.	(Initial value)
1	The FRC is cleared at compare-match A.	

6.2.6 Timer Control Register (TCR) – H'FF96

Bit	7	6	5	4	3	2	1	0
	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The TCR is an 8-bit readable/writable register that selects the rising or falling edge of the input capture signals, enables the input capture buffer mode, and selects the FRC clock source.

The TCR is initialized to H'00 at a reset and in the standby modes.

Bit 7 – Input Edge Select A (IEDGA): This bit causes input capture A events to be recognized on the selected edge of the input capture A signal (FTIA).

Bit 7

IEDGA	Description	
0	Input capture A events are recognized on the falling edge of FTIA.	(Initial value)
1	Input capture A events are recognized on the rising edge of FTIA.	

Bit 6 – Input Edge Select B (IEDGB): This bit causes input capture B events to be recognized on the selected edge of the input capture B signal (FTIB).

Bit 6

IEDGB	Description	
0	Input capture B events are recognized on the falling edge of FTIB.	(Initial value)
1	Input capture B events are recognized on the rising edge of FTIB.	

Bit 5 – Input Edge Select C (IEDGC): This bit causes input capture C events to be recognized on the selected edge of the input capture C signal (FTIC). In buffer mode (when BUFEA = "1"), it also causes input capture A events to be recognized on the selected edge of FTIA.

Bit 5

IEDGC	Description	
0	Input capture C events are recognized on the falling edge of FTIC.	(Initial value)
1	Input capture C events are recognized on the rising edge of FTIC.	

Bit 4 – Input Edge Select D (IEDGD): This bit causes input capture D events to be recognized on the selected edge of the input capture D signal (FTID). In the buffer mode (when BUFEB = "1"), it also causes input capture B events to be recognized on the selected edge of FTIB.

Bit 4

IEDGD	Description	
0	Input capture D events are recognized on the falling edge of FTID.	(Initial value)
1	Input capture D events are recognized on the rising edge of FTID.	

Bit 3 – Buffer Enable A (BUFEA): This bit selects whether to use ICRC as a buffer register for ICRA.

Bit 3

BUFEA	Description	
0	ICRC is used for input capture C.	(Initial value)
1	ICRC is used as a buffer register for input capture A. Input C is not captured.	

Bit 2 – Buffer Enable B (BUFEB): This bit selects whether to use ICRD as a buffer register for ICRB.

Bit 2

BUFEB	Description	
0	ICRD is used for input capture D.	(Initial value)
1	ICRD is used as a buffer register for input capture B. Input D is not captured.	

Bits 1 and 0 – Clock Select (CKS1 and CKS0): These bits select external clock input or one of three internal clock sources for the FRC. External clock pulses are counted on the rising edge.

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	Ø/2 Internal clock source	(Initial value)
0	1	Ø/8 Internal clock source	
1	0	Ø/32 Internal clock source	
1	1	External clock source (rising edge)	

6.2.7 Timer Output Compare Control Register (TOCR) – H'FF97

Bit	7	6	5	4	3	2	1	0
	—	—	—	OCRS	OEA	OEB	OLVLA	OLVLB
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

The TOCR is an 8-bit readable/writable register that controls the output compare function.

The TOCR is initialized to H'E0 at a reset and in the standby modes.

Bits 7 to 5 – Reserved: These bits cannot be modified and are always read as "1."

Bit 4 – Output Compare Register Select (OCRS): When the CPU accesses addresses H'FF94 and H'FF95, this bit directs the access to either OCRA or OCRB. These two registers share the same addresses as follows:
Upper byte of OCRA and upper byte of OCRB: H'FF94
Lower byte of OCRA and lower byte of OCRB: H'FF95

Bit 4 OCRS	Description	
0	The CPU can access OCRA.	(Initial value)
1	The CPU can access OCRB.	

Bit 3 – Output Enable A (OEA): This bit enables or disables output of the output compare A signal (FTOA). When output compare A is disabled, the corresponding pin is used as a general-purpose input/output port.

Bit 3 OEA	Description	
0	Output compare A output is disabled.	(Initial value)
1	Output compare A output is enabled.	

Bit 2 – Output Enable B (OEB): This bit enables or disables output of the output compare B signal (FTOB). When output compare B is disabled, the corresponding pin is used as a general-purpose input/output or interrupt port.

Bit 2

OEB	Description	
0	Output compare B output is disabled.	(Initial value)
1	Output compare B output is enabled.	

Bit 1 – Output Level A (OLVLA): This bit selects the logic level to be output at the FTOA pin when the FRC and OCRA values match.

Bit 1

OLVLA	Description	
0	A “0” logic level (Low) is output for compare-match A.	(Initial value)
1	A “1” logic level (High) is output for compare-match A.	

Bit 0 – Output Level B (OLVLB): This bit selects the logic level to be output at the FTOB pin when the FRC and OCRB values match.

Bit 0

OLVLB	Description	
0	A “0” logic level (Low) is output for compare-match B.	(Initial value)
1	A “1” logic level (High) is output for compare-match B.	

6.3 CPU Interface

The free-running counter (FRC), output compare registers (OCRA and OCRB), and input capture registers (ICRA to ICRD) are 16-bit registers, but they are connected to an 8-bit data bus. When the CPU accesses these registers, to ensure that both bytes are written or read simultaneously, the access is performed using an 8-bit temporary register (TEMP).

These registers are written and read as follows:

- **Register Write**
  When the CPU writes to the upper byte, the byte of write data is placed in TEMP. Next, when the CPU writes to the lower byte, this byte of data is combined with the byte in TEMP and all 16 bits are written in the register simultaneously.

- **Register Read**
  When the CPU reads the upper byte, the upper byte of data is sent to the CPU and the lower byte is placed in TEMP. When the CPU reads the lower byte, it receives the value in TEMP.
  (As an exception, when the CPU reads OCRA or OCRB, it reads both the upper and lower bytes directly, without using TEMP.)

Programs that access these registers should normally use word access. Equivalently, they may access first the upper byte, then the lower byte by two consecutive byte accesses. Data will not be transferred correctly if the bytes are accessed in reverse order, if only one byte is accessed, or if the upper and lower bytes are accessed separately and another register is accessed in between, altering the value in TEMP.

Coding Examples

To write the contents of general register R0 to OCRA: `MOV.W R0, @OCRA`
To transfer the contents of ICRA to general register R0: `MOV.W @ICRA, R0`

Figure 6-4 shows the data flow when the FRC is accessed. The other registers are accessed in the same way.

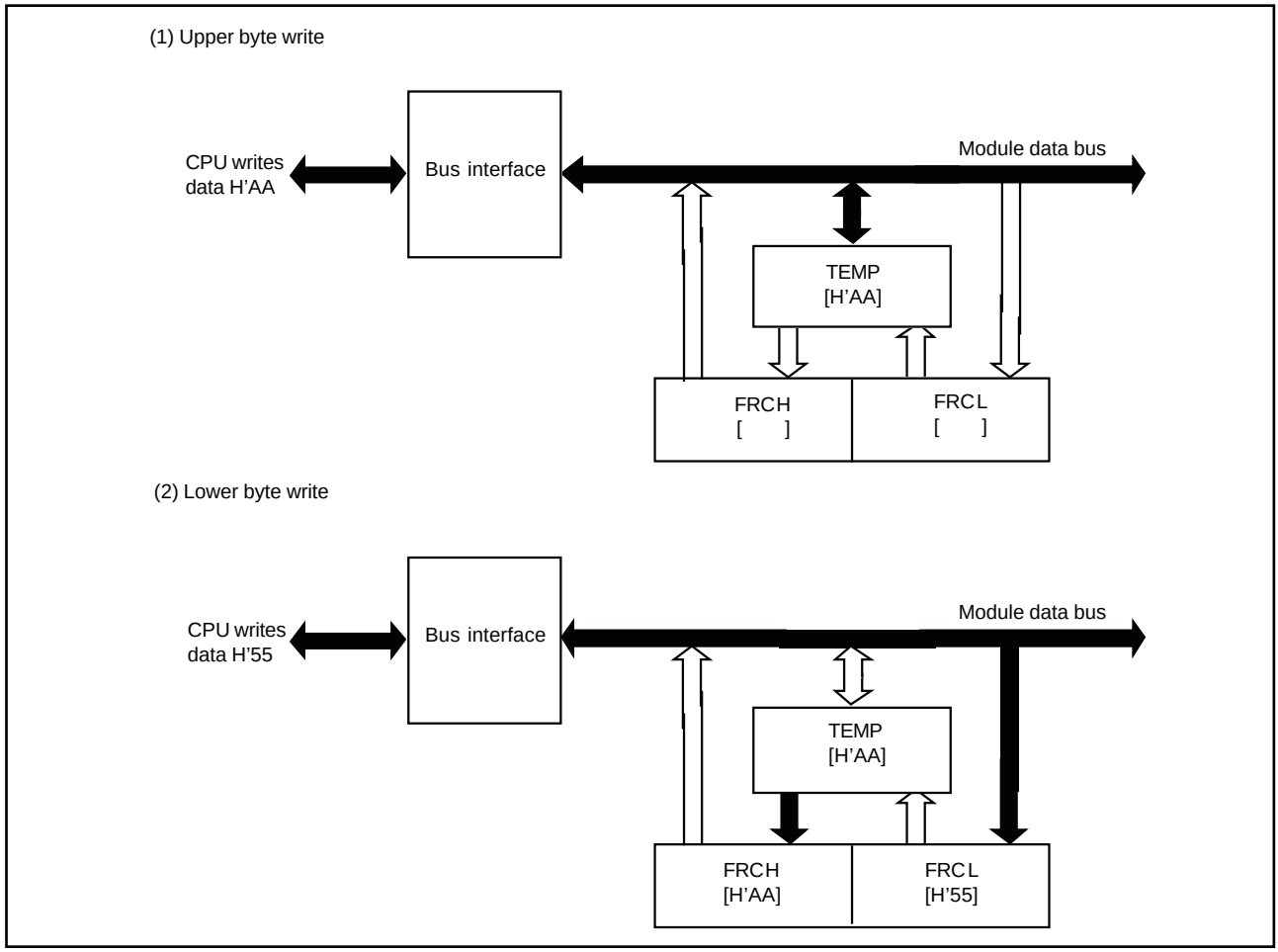


Figure 6-4 (a). Write Access to FRC (When CPU Writes H'AA55)

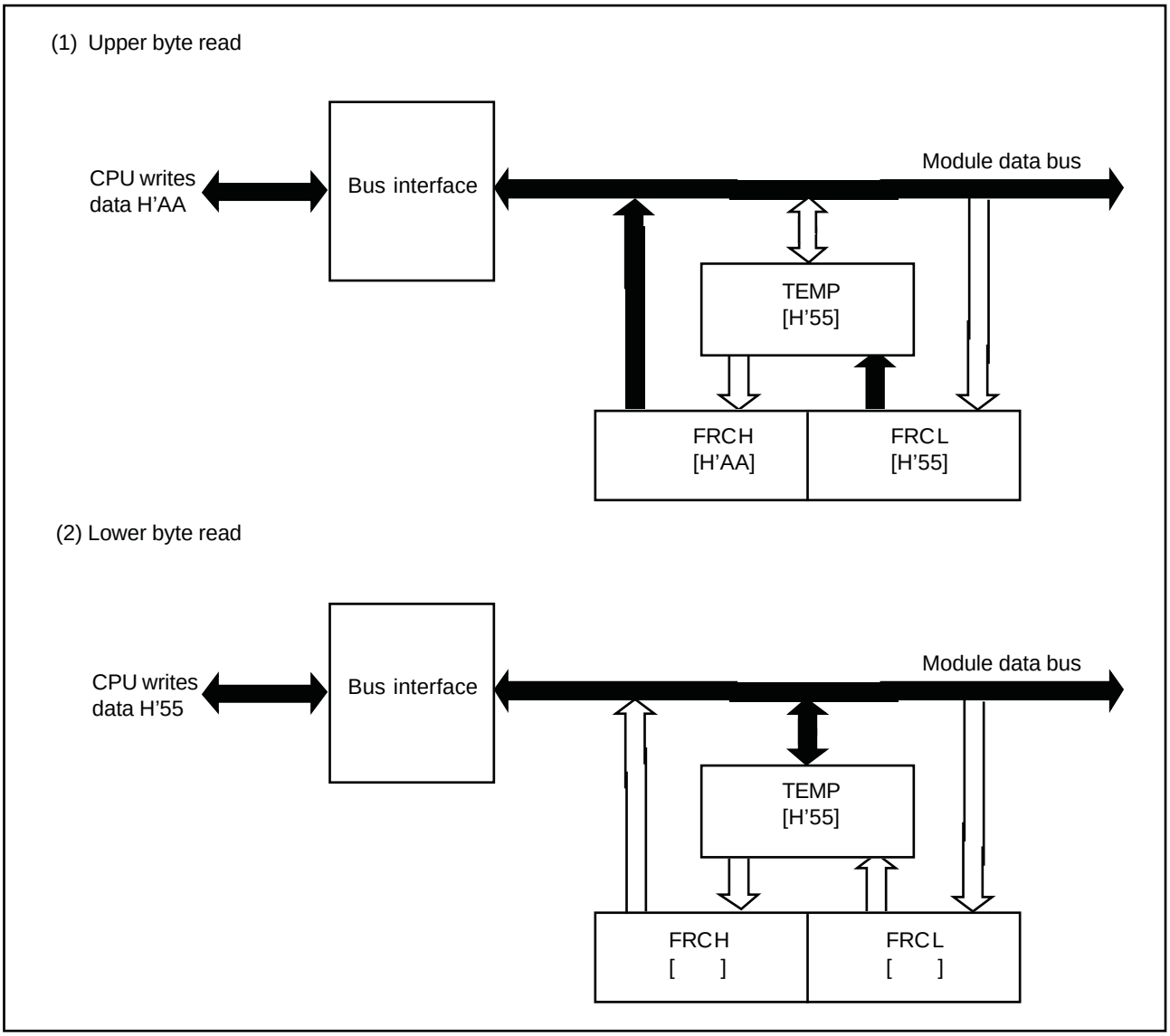


Figure 6-4 (b). Read Access to FRC (When FRC Contains H'AA55)

6.4 Operation

6.4.1 FRC Incrementation Timing

The FRC increments on a pulse generated once for each period of the selected (internal or external) clock source.

The internal clock sources are created from the system clock (Ø) by a prescaler. The FRC increments on a pulse generated from the falling edge of the prescaler output. See Figure 6-5.

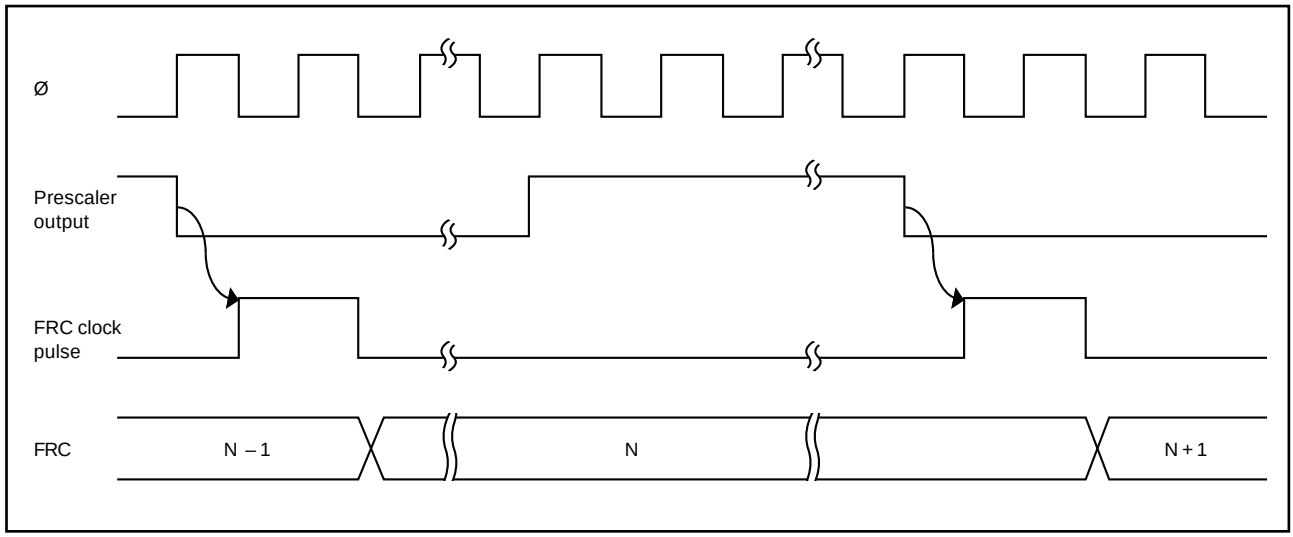


Figure 6-5. Increment Timing for Internal Clock Source

If external clock input is selected, the FRC increments on the rising edge of the FTCI clock signal. Figure 6-6 shows the increment timing.

The pulse width of the external clock signal must be at least 1.5 system clock (Ø) periods. The counter will not increment correctly if the pulse width is shorter than one Ø period.

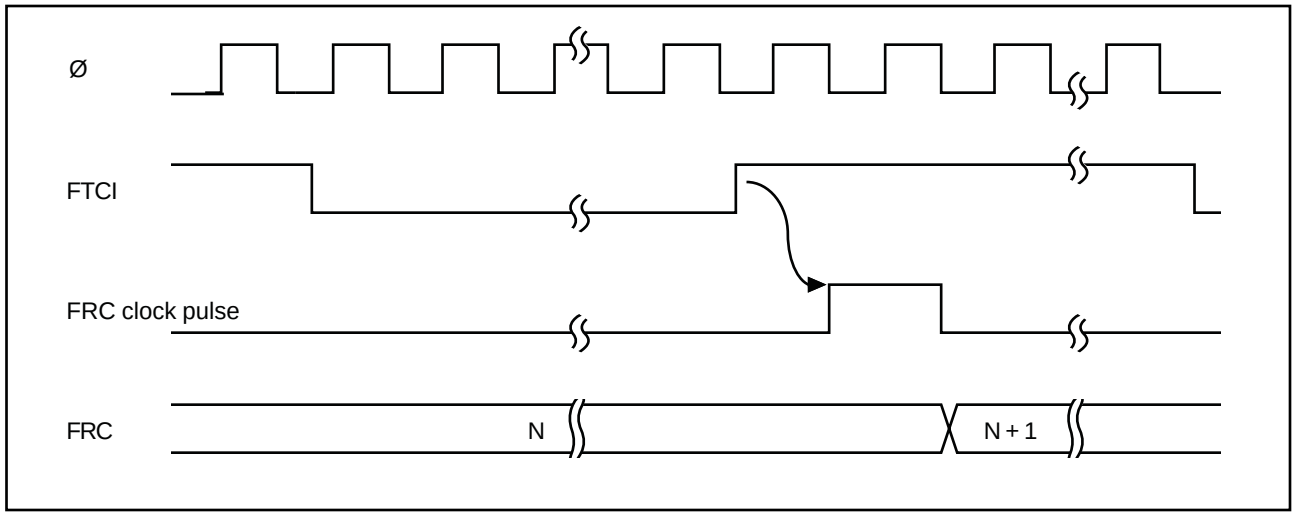


Figure 6-6. Increment Timing for External Clock Source

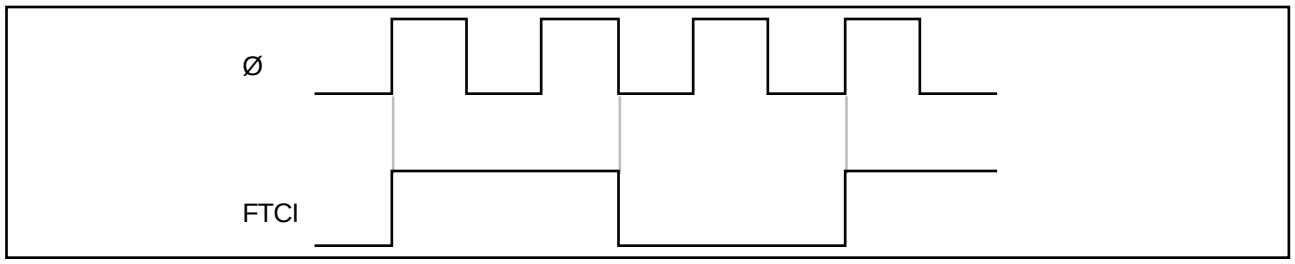


Figure 6-7. Minimum External Clock Pulse Width

6.4.2 Output Compare Timing

(1) Setting of Output Compare Flags A and B (OCFA and OCFB): The output compare flags are set to “1” by an internal compare-match signal generated when the FRC value matches the OCRA or OCRB value. This compare-match signal is generated at the last state in which the two values match, just before the FRC increments to a new value.

Accordingly, when the FRC and OCR values match, the compare-match signal is not generated until the next period of the clock source. Figure 6-8 shows the timing of the setting of the output compare flags.

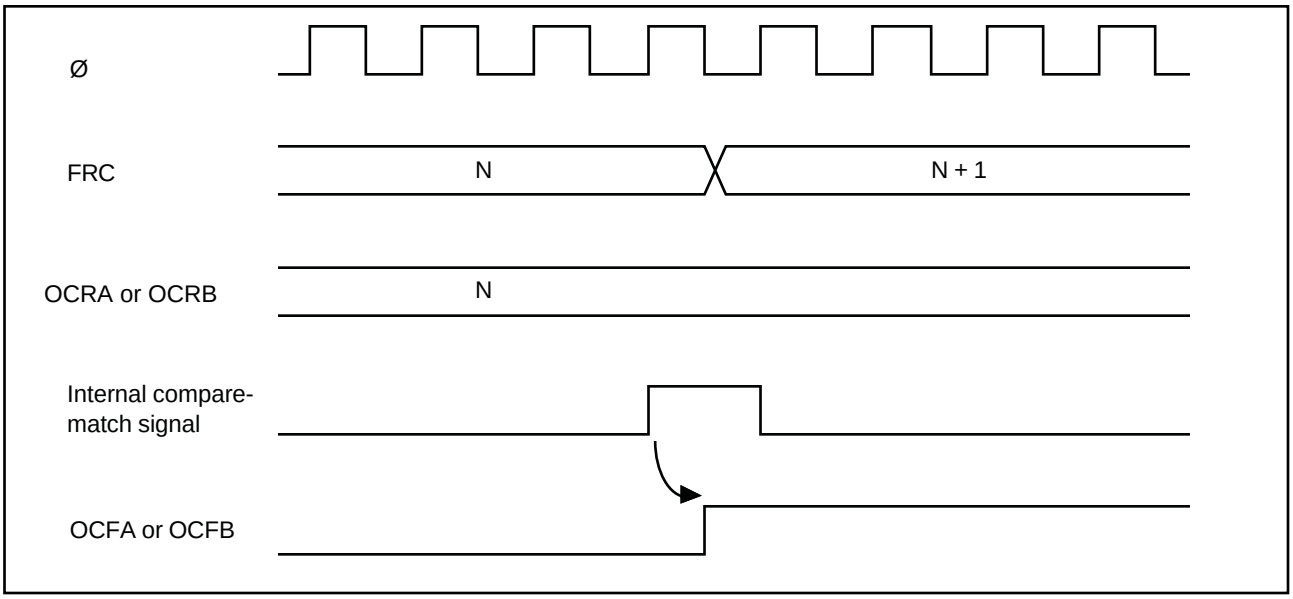


Figure 6-8. Setting of Output Compare Flags

(2) Timing of Output Compare Flag (OCFA or OCFB) Clearing: The output compare flag OCFA or OCFB is cleared when the CPU writes a “0” in this bit.

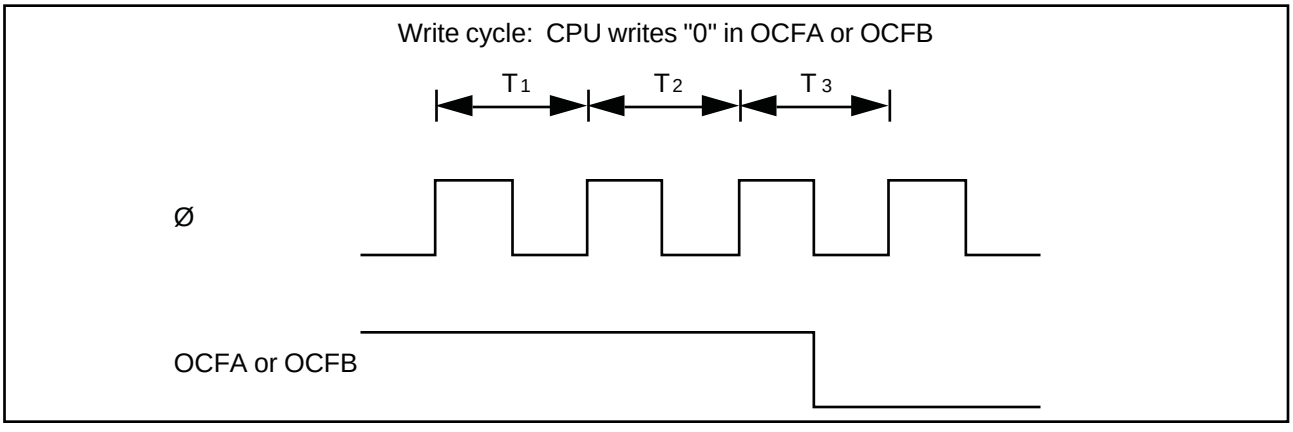


Figure 6-9. Clearing of Output Compare Flag

(3) Output Timing: When a compare-match occurs, the logic level selected by the output level bit (OLVLA or OLVLB) in TOCR is output at the output compare pin (FTOA or FTOB). Figure 6-10 shows the timing of this operation for compare-match A.

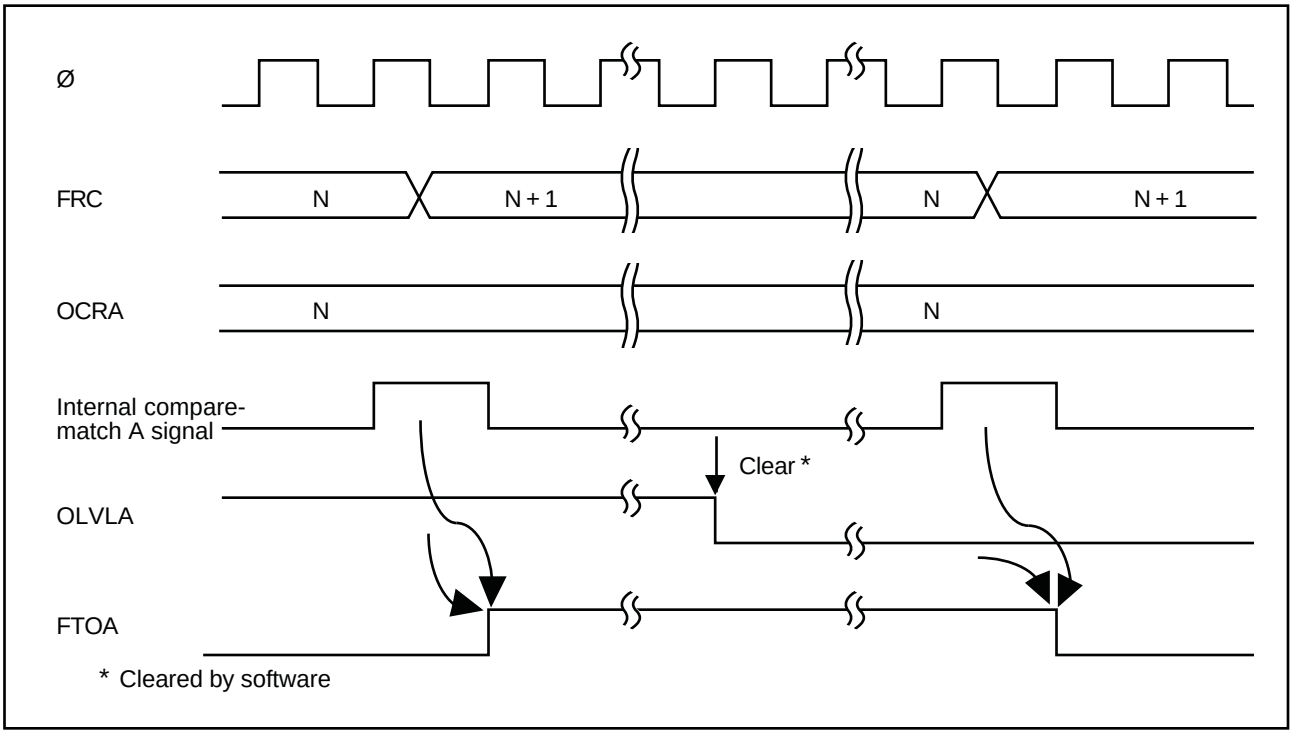


Figure 6-10. Timing of Output Compare A

(4) FRC Clear Timing: If the CCLRA bit in the TCSR is set to "1," the FRC is cleared when compare-match A occurs. Figure 6-11 shows the timing of this operation.

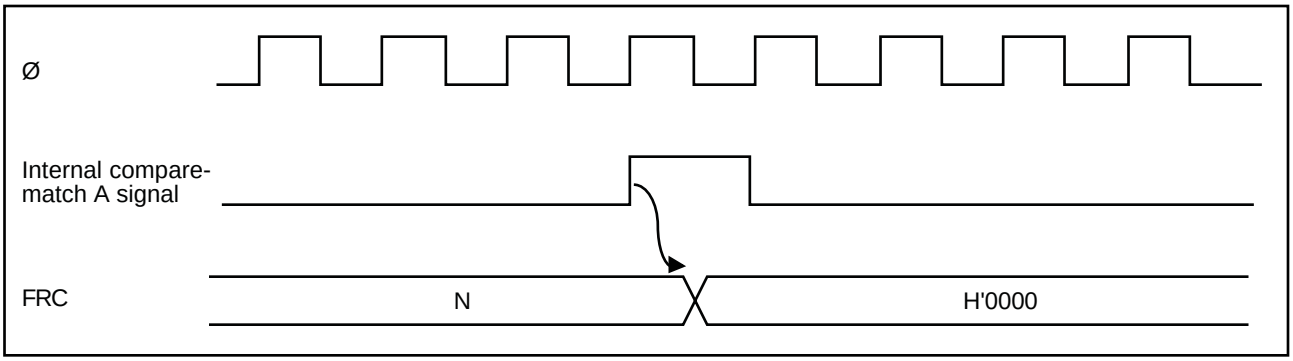


Figure 6-11. Clearing of FRC by Compare-Match A

6.4.3 Input Capture Timing

(1) Input Capture Timing: An internal input capture signal is generated from the rising or falling edge of the signal at the input capture pin FTIx (x = A, B, C, D), as selected by the corresponding IEDGx bit in TCR. Figure 6-12 shows the usual input capture timing when the rising edge is selected (IEDGx = "1").

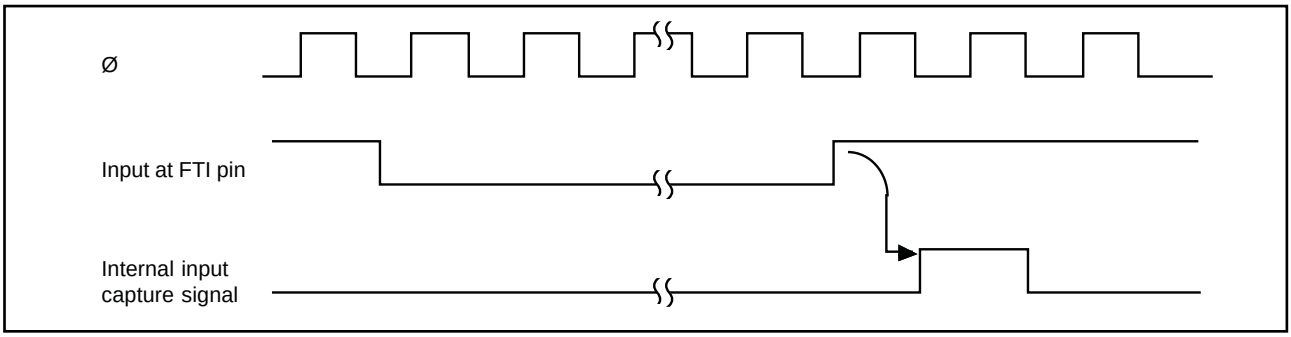


Figure 6-12. Input Capture Timing (Usual Case)

If the upper byte of ICRx is being read when the input capture signal arrives, the internal input capture signal is delayed by one state. Figure 6-13 shows the timing for this case.

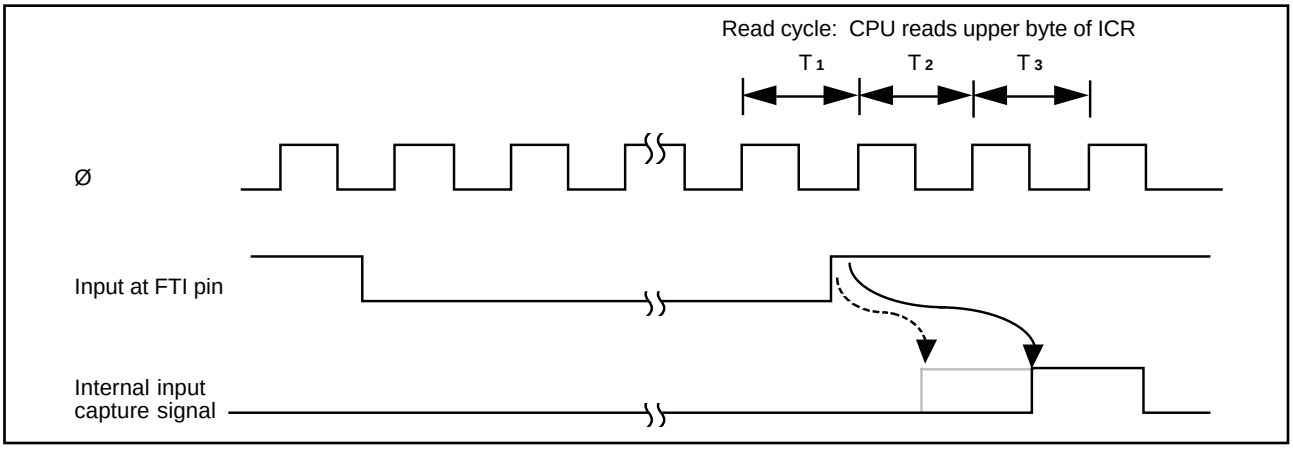


Figure 6-13. Input Capture Timing (1-State Delay)

In buffer mode, this delay occurs if the CPU is reading either of the two registers concerned. When ICRA and ICRC are used in buffer mode, for example, if the upper byte of either ICRA or ICRC is being read when the FTIA input arrives, the internal input capture signal is delayed by one state. Figure 6-14 shows the timing for this case. The case of ICRB and ICRD is similar.

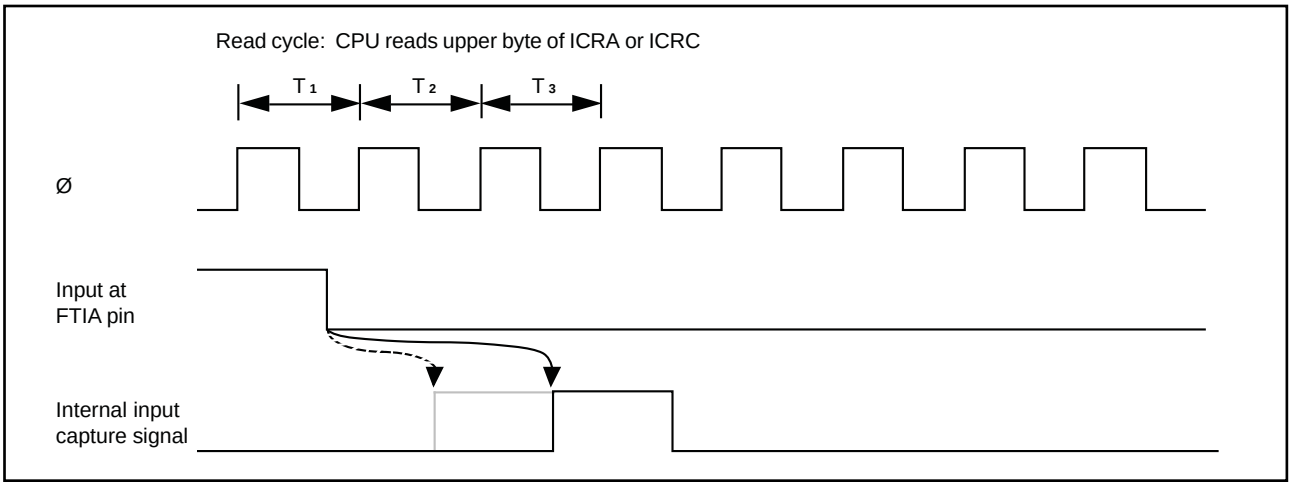


Figure 6-14. Input Capture Timing (1-State Delay, Buffer Mode)

Figure 6-15 shows how input capture operates when ICRA and ICRC are used in buffer mode and IEDGA and IEDGC are set to different values (IEDGA = 0 and IEDGC = 1, or IEDGA = 1 and IEDGC = 0), so that input capture is performed on both the rising and falling edges of FTIA.

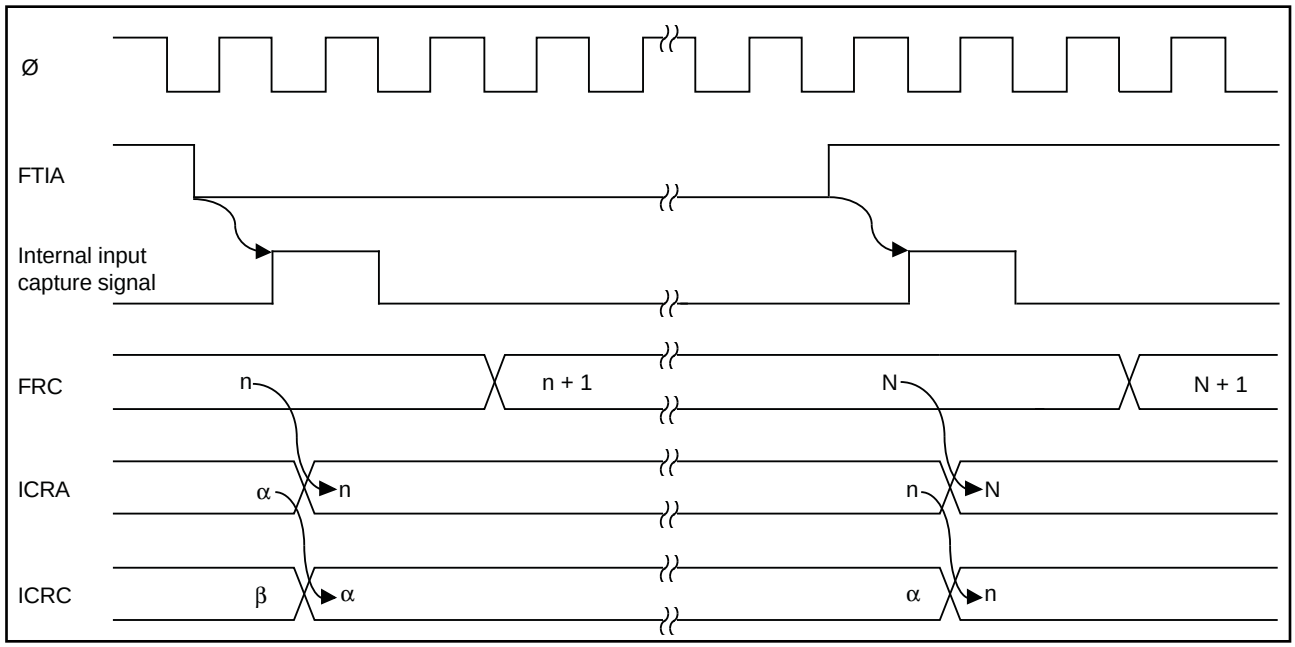


Figure 6-15. Buffered Input Capture with Both Edges Selected

In this mode, FTIC does not cause the FRC contents to be copied to ICRC. However, input capture flag C still sets on the edge of FTIC selected by IEDGC, and if the interrupt enable bit (ICICE) is set, a CPU interrupt is requested.

The situation when ICRB and ICRD are used in buffer mode is similar.

(2) Timing of Input Capture Flag (ICF) Setting: The input capture flag ICFx (x = A, B, C, D) is set to "1" by the internal input capture signal. Figure 6-16 shows the timing of this operation.

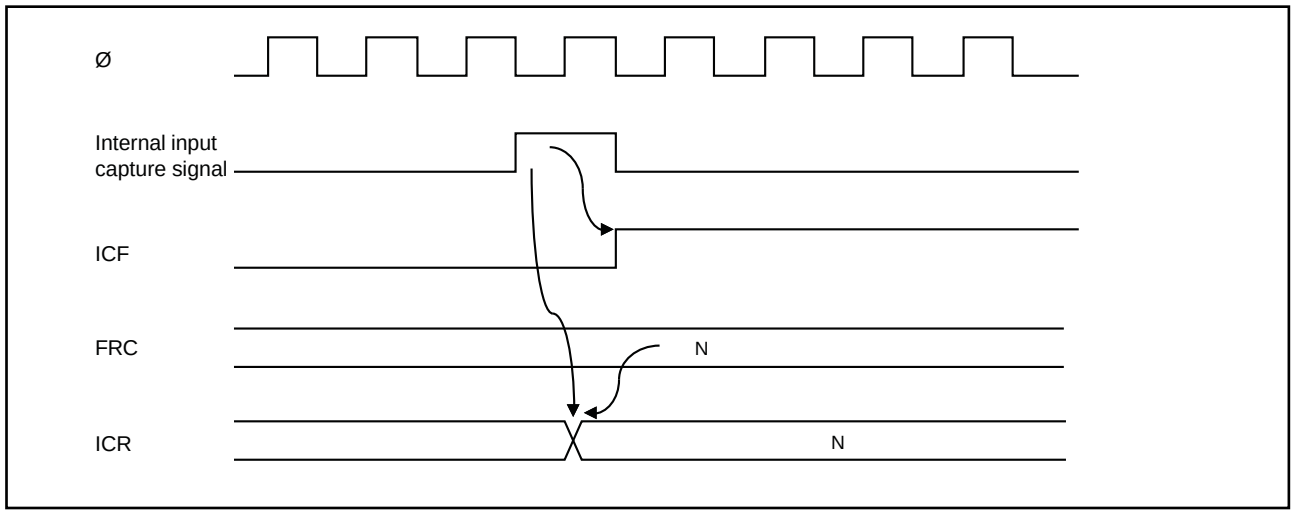


Figure 6-16. Setting of Input Capture Flag

(3) Timing of Input Capture Flag (ICF) Clearing: The input capture flag ICFx (x = A, B, C, D) is cleared when the CPU writes a “0” in this bit.

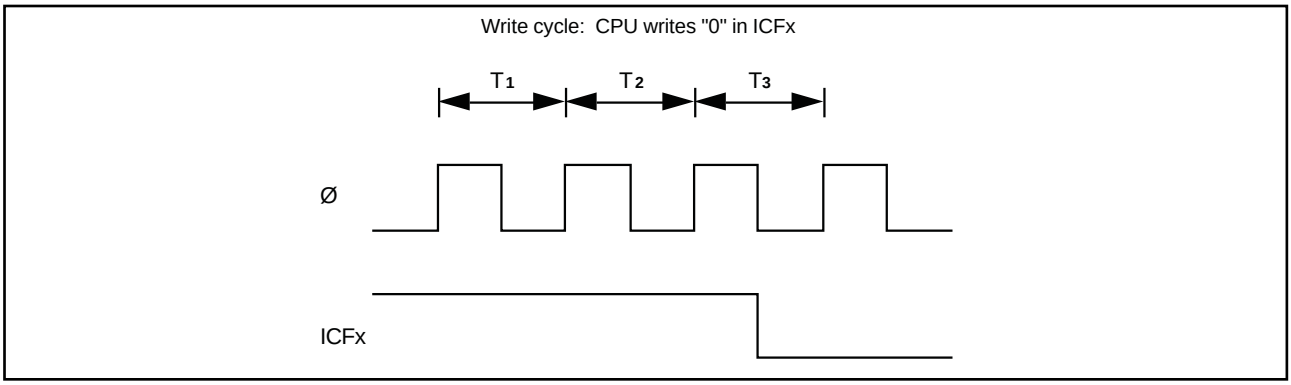


Figure 6-17. Clearing of Input Capture Flag

6.4.4 Setting of FRC Overflow Flag (OVF)

The FRC overflow flag (OVF) is set to “1” when the FRC overflows (changes from H’FFFF to H’0000). Figure 6-18 shows the timing of this operation.

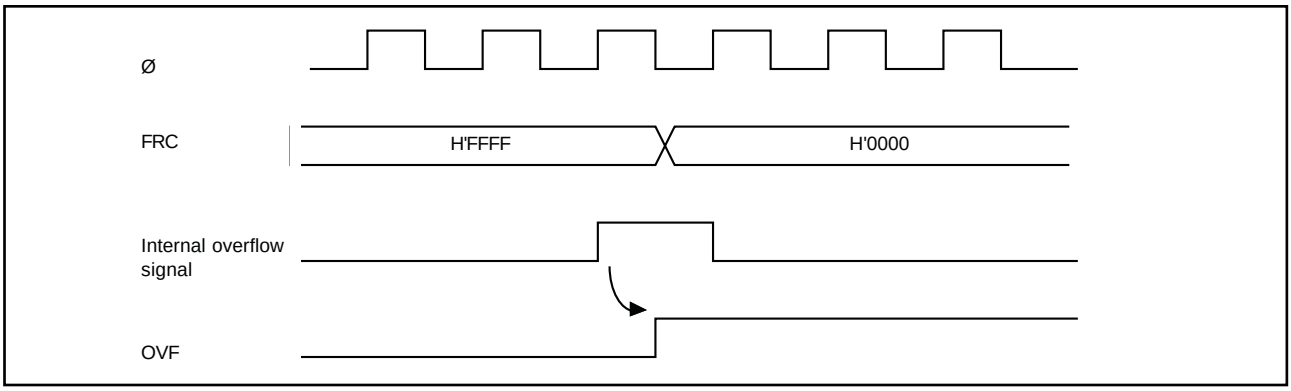


Figure 6-18. Setting of Overflow Flag (OVF)

(2) Timing of Overflow Flag (OVF) Clearing: The overflow flag is cleared when the CPU writes a “0” in this bit.

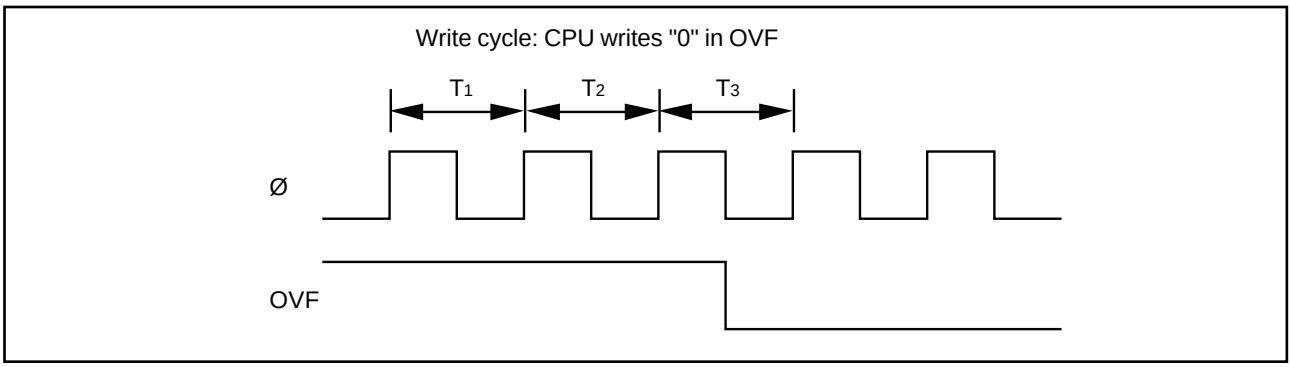


Figure 6-19. Clearing of Overflow Flag

6.5 Interrupts

The free-running timer channel can request seven types of interrupts: input capture A to D (ICIA, ICIB, ICIC, ICID), output compare A and B (OCIA and OCIB), and overflow (FOVI). Each interrupt is requested when the corresponding enable and flag bits are set. Independent signals are sent to the interrupt controller for each type of interrupt. Table 6-3 lists information about these interrupts.

Table 6-3. Free-Running Timer Interrupts

Interrupt	Description	Priority
ICIA	Requested when ICFA and ICIAE are set	High
ICIB	Requested when ICFB and ICIBE are set	
ICIC	Requested when ICFC and ICICE are set	
ICID	Requested when ICFD and ICIDE are set	
OCIA	Requested when OCFA and OCIAE are set	
OCIB	Requested when OCFB and OCIBE are set	
FOVI	Requested when OVF and OVIE are set	Low

6.6 Sample Application

In the example below, the free-running timer channel is used to generate two square-wave outputs with a 50% duty factor and arbitrary phase relationship. The programming is as follows:

(1) The CCLRA bit in the TCSR is set to “1.”

(2) Each time a compare-match interrupt occurs, software inverts the corresponding output level bit in TOCR (OLVLA or OLVLB).

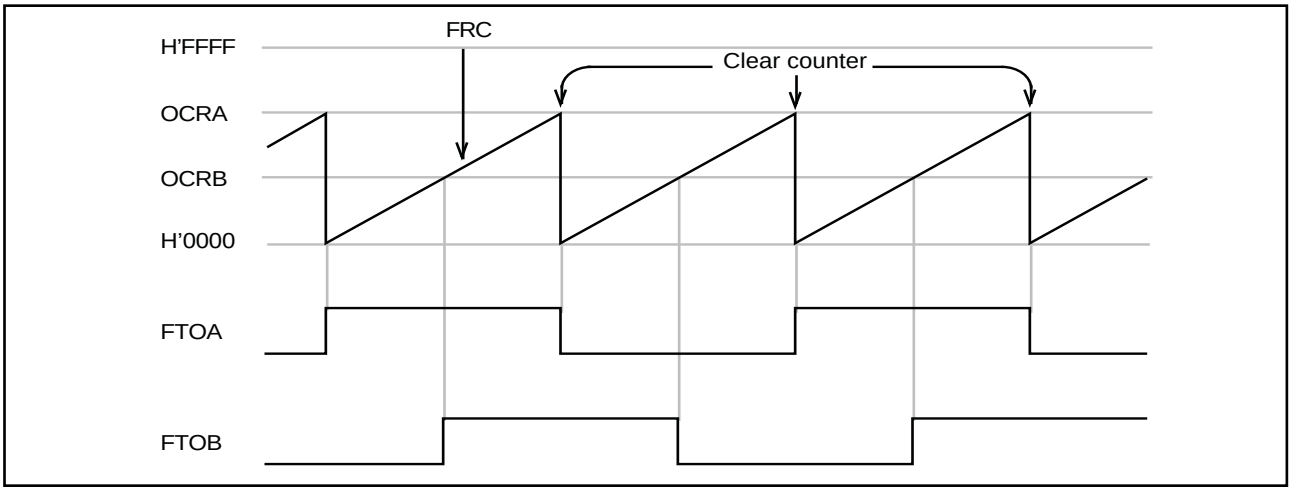


Figure 6-20. Square-Wave Output (Example)

6.7 Application Notes

Application programmers should note that the following types of contention can occur in the free-running timers.

(1) Contention between FRC Write and Clear: If an internal counter clear signal is generated during the T_3 state of a write cycle to the lower byte of the free-running counter, the clear signal takes priority and the write is not performed.

Figure 6-21 shows this type of contention.

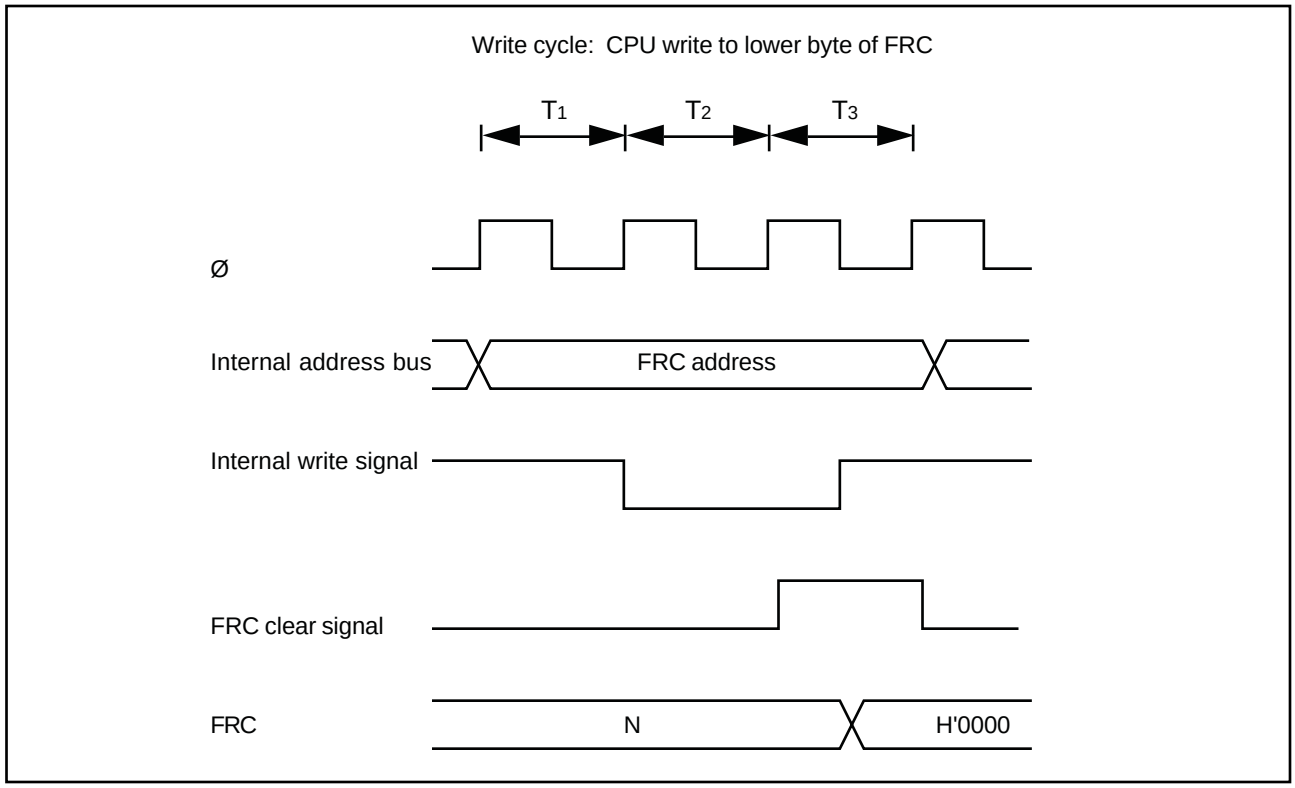


Figure 6-21. FRC Write-Clear Contention

(2) Contention between FRC Write and Increment: If an FRC increment pulse is generated during the T_3 state of a write cycle to the lower byte of the free-running counter, the write takes priority and the FRC is not incremented.

Figure 6-22 shows this type of contention.

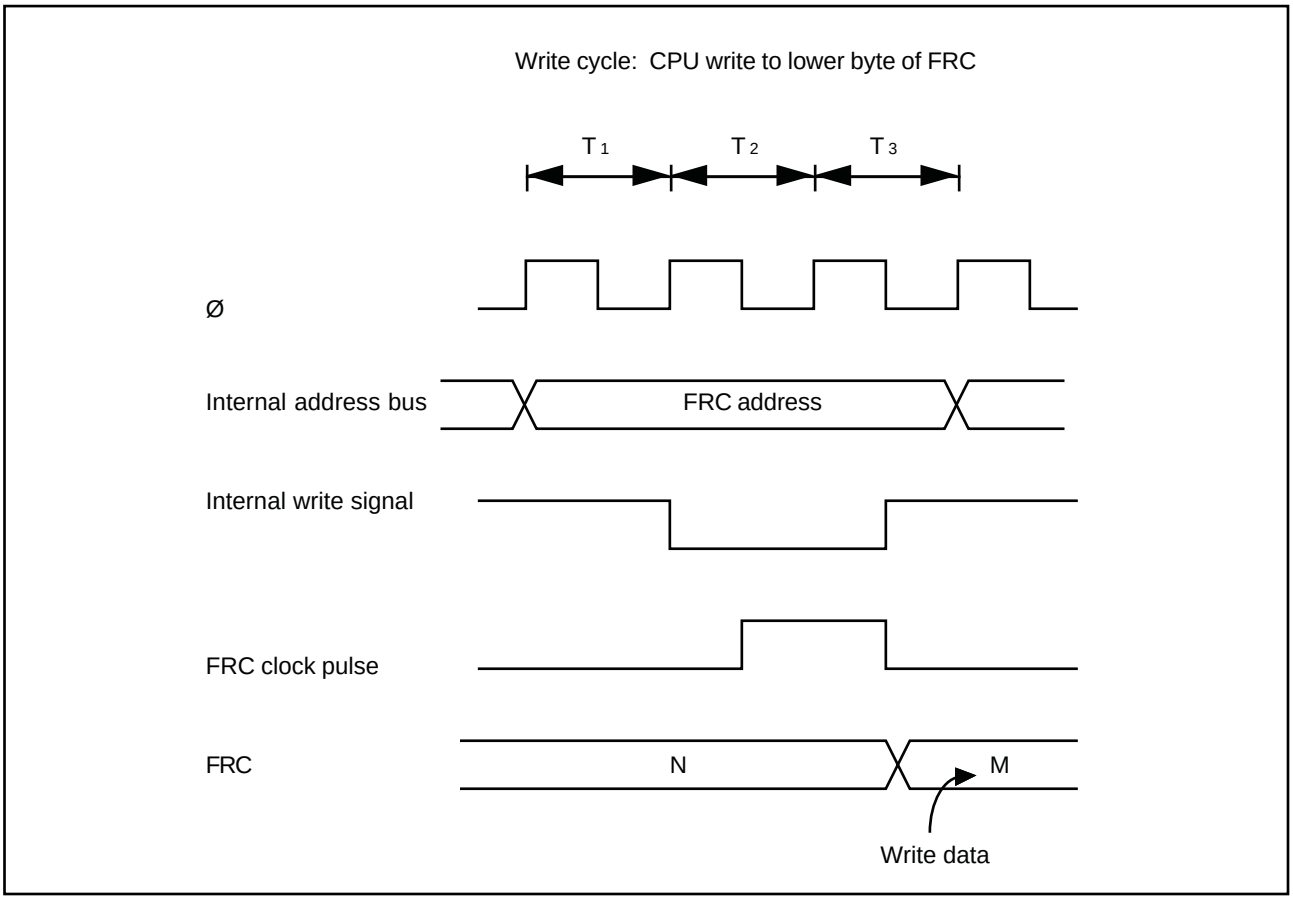


Figure 6-22. FRC Write-Increment Contention

(3) Contention between OCR Write and Compare-Match: If a compare-match occurs during the T3 state of a write cycle to the lower byte of OCRA or OCRB, the write takes precedence and the compare-match signal is inhibited.

Figure 6-23 shows this type of contention.

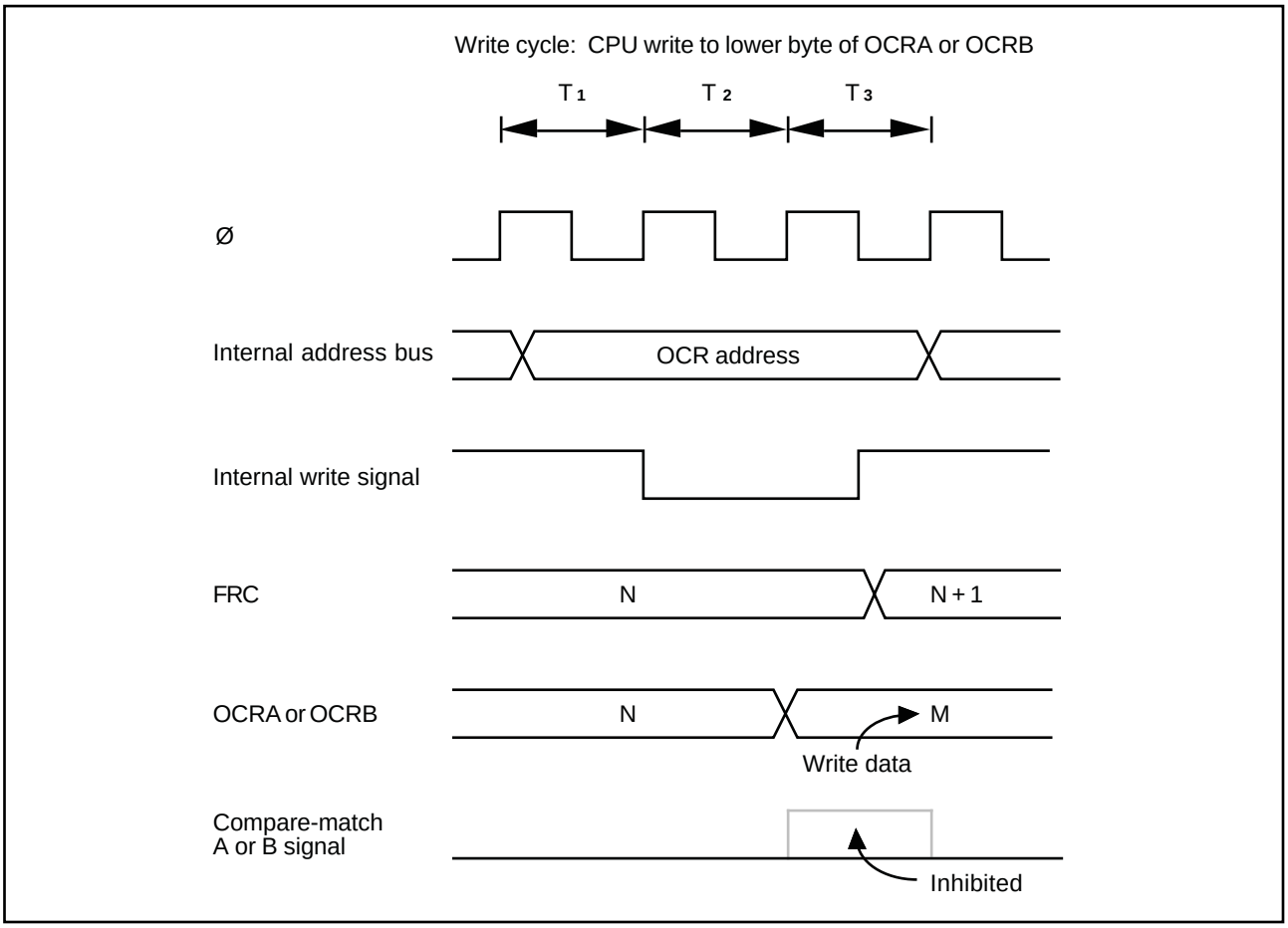


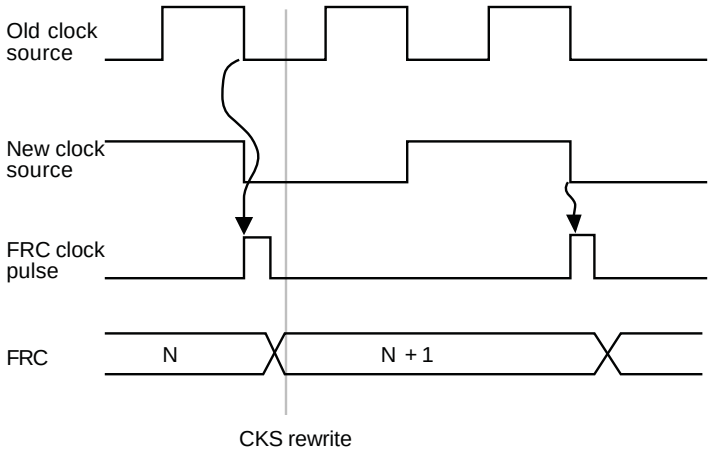
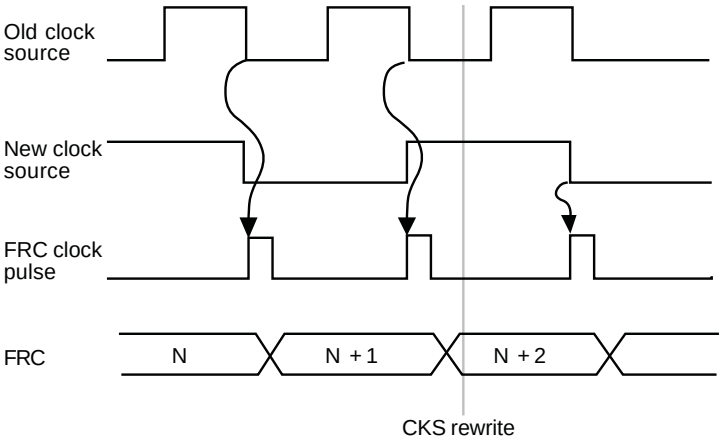
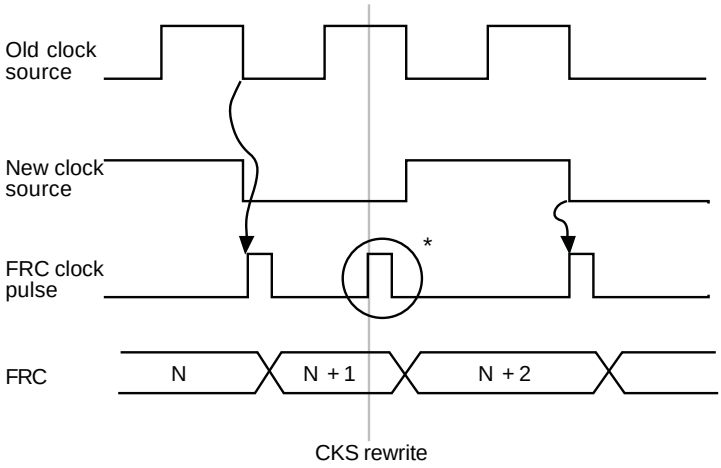
Figure 6-23. Contention between OCR Write and Compare-Match

(4) Incrementation Caused by Changing of Internal Clock Source: When an internal clock source is changed, the changeover may cause the FRC to increment. This depends on the time at which the clock select bits (CKS1 and CKS0) are rewritten, as shown in Table 6-4.

The pulse that increments the FRC is generated at the falling edge of the internal clock source. If clock sources are changed when the old source is High and the new source is Low, as in case No. 3 in Table 6-5, the changeover generates a falling edge that triggers the FRC increment clock pulse.

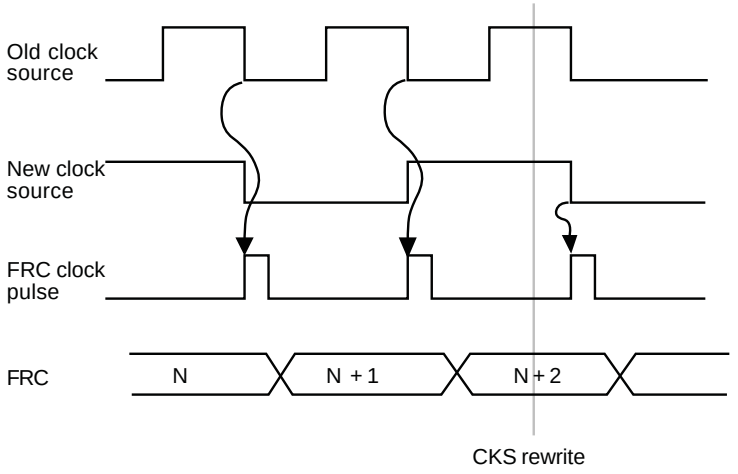
Switching between an internal and external clock source can also cause the FRC to increment.

Table 6-4. Effect of Changing Internal Clock Sources

No.	Description	Timing chart
1	Low → Low: CKS1 and CKS0 are rewritten while both clock sources are Low.	
2	Low → High: CKS1 and CKS0 are rewritten while old clock source is Low and new clock source is High.	
3	High → Low: CKS1 and CKS0 are rewritten while old clock source is High and new clock source is Low.	

* The switching of clock sources is regarded as a falling edge that increments the FRC.

Table 6-4. Effect of Changing Internal Clock Sources (cont.)

No.	Description	Timing chart
4	High → High: CKS1 and CKS0 are rewritten while both clock sources are High.	

Section 7. 8-Bit Timers

7.1 Overview

The H8/330 chip includes an 8-bit timer module with two channels (TMR0 and TMR1). Each channel has an 8-bit counter (TCNT) and two time constant registers (TCORA and TCORB) that are constantly compared with the TCNT value to detect compare-match events. One application of the 8-bit timer module is to generate a rectangular-wave output with an arbitrary duty factor.

7.1.1 Features

The features of the 8-bit timer module are listed below.

- Selection of four clock sources
  The counters can be driven by an internal clock signal (Ø/8, Ø/64, or Ø/1024) or an external clock input (enabling use as an external event counter).

- Selection of three ways to clear the counters
  The counters can be cleared on compare-match A or B, or by an external reset signal.

- Timer output controlled by two time constants
  The timer output signal in each channel is controlled by two independent time constants, enabling the timer to generate output waveforms with an arbitrary duty factor.

- Three independent interrupts
  Compare-match A and B and overflow interrupts can be requested independently.

7.1.2 Block Diagram

Figure 7-1 shows a block diagram of one channel in the 8-bit timer module. The other channel is identical.

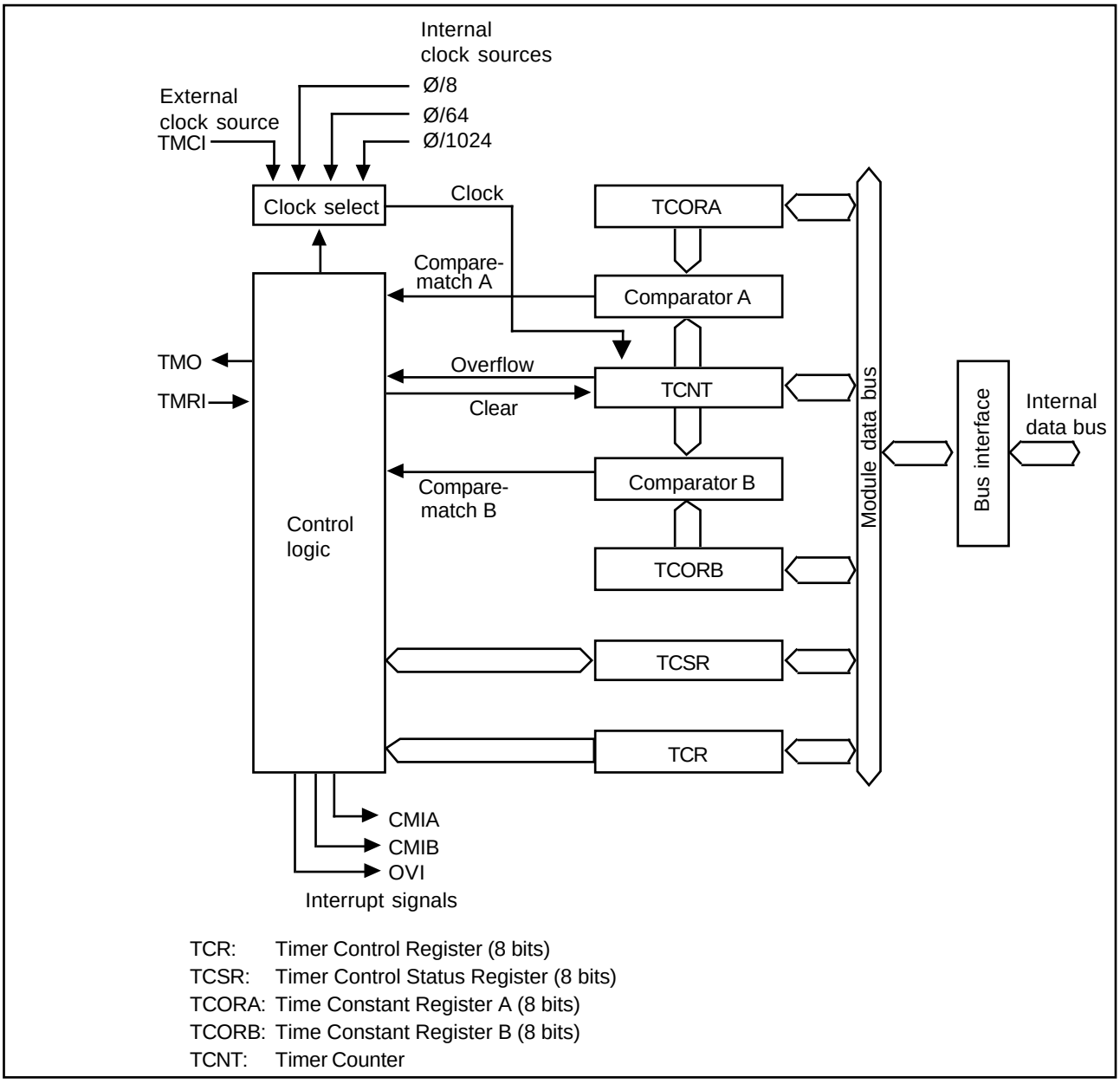


Figure 7-1. Block Diagram of 8-Bit Timer

7.1.3 Input and Output Pins

Table 7-1 lists the input and output pins of the 8-bit timer.

Table 7-1. Input and Output Pins of 8-Bit Timer

	Abbreviation			
Name	**TMR0**	**TMR1**	**I/O**	**Function**
Timer output	TMO_0	TMO_1	Output	Output controlled by compare-match
Timer clock input	$TMCI_0$	$TMCI_1$	Input	External clock source for the counter
Timer reset input	$TMRI_0$	$TMRI_1$	Input	External reset signal for the counter

7.1.4 Register Configuration

Table 7-2 lists the registers of the 8-bit timer module. Each channel has an independent set of registers.

Table 7-2. 8-Bit Timer Registers

				Address	
Name	**Abbreviation**	**R/W**	**Initial value**	**TMR0**	**TMR1**
Timer control register	TCR	R/W	H'00	H'FFC8	H'FFD0
Timer control/status register	TCSR	R/(W)*	H'10	H'FFC9	H'FFD1
Timer constant register A	TCORA	R/W	H'FF	H'FFCA	H'FFD2
Timer constant register B	TCORB	R/W	H'FF	H'FFCB	H'FFD3
Timer counter	TCNT	R/W	H'00	H'FFCC	H'FFD4

* Software can write a "0" to clear bits 7 to 5, but cannot write a "1" in these bits.

7.2 Register Descriptions

7.2.1 Timer Counter (TCNT) – H'FFC8 (TMR0), H'FFD0 (TMR1)

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

Each timer counter (TCNT) is an 8-bit up-counter that increments on a pulse generated from one of four clock sources. The clock source is selected by clock select bits 2 to 0 (CKS2 to CKS0) of the timer control register (TCR). The CPU can always read or write the timer counter.

The timer counter can be cleared by an external reset input or by an internal compare-match signal generated at a compare-match event. Clock clear bits 1 and 0 (CCLR1 and CCLR0) of the timer control register select the method of clearing.

When a timer counter overflows from H'FF to H'00, the overflow flag (OVF) in the timer control/status register (TCSR) is set to "1."

The timer counters are initialized to H'00 at a reset and in the standby modes.

7.2.2 Time Constant Registers A and B (TCORA and TCORB) – H’FFCA and H’FFCB (TMR0), H’FFD2 and H’FFD3 (TMR1)

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

TCORA and TCORB are 8-bit readable/writable registers. The timer count is continually compared with the constants written in these registers. When a match is detected, the corresponding compare-match flag (CMFA or CMFB) is set in the timer control/status register (TCSR).

The timer output signal (TMO0 or TMO1) is controlled by these compare-match signals as specified by output select bits 3 to 0 (OS3 to OS0) in the timer control/status register (TCSR).

TCORA and TCORB are initialized to H’FF at a reset and in the standby modes.

Compare-match is not detected during the T3 state of a write cycle to TCORA or TCORB. See item (3) in section 7.6, "Application Notes."

7.2.3 Timer Control Register (TCR) – H’FFC8 (TMR0), H’FFD0 (TMR1)

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Each TCR is an 8-bit readable/writable register that selects the clock source and the time at which the timer counter is cleared, and enables interrupts.

The TCRs are initialized to H’00 at a reset and in the standby modes.

Bit 7 – Compare-match Interrupt Enable B (CMIEB): This bit selects whether to request compare-match interrupt B (CMIB) when compare-match flag B (CMFB) in the timer control/status register (TCSR) is set to “1.”

Bit 7

CMIEB	Description	
0	Compare-match interrupt request B (CMIB) is disabled.	(Initial value)
1	Compare-match interrupt request B (CMIB) is enabled.	

Bit 6 – Compare-match Interrupt Enable A (CMIEA): This bit selects whether to request compare-match interrupt A (CMIA) when compare-match flag A (CMFA) in the timer control/status register (TCSR) is set to "1."

Bit 6

CMIEA	Description	
0	Compare-match interrupt request A (CMIA) is disabled.	(Initial value)
1	Compare-match interrupt request A (CMIA) is enabled.	

Bit 5 – Timer Overflow Interrupt Enable (OVIE): This bit selects whether to request a timer overflow interrupt (OVI) when the overflow flag (OVF) in the timer control/status register (TCSR) is set to "1."

Bit 5

OVIE	Description	
0	The timer overflow interrupt request (OVI) is disabled.	(Initial value)
1	The timer overflow interrupt request (OVI) is enabled.	

Bits 4 and 3 – Counter Clear 1 and 0 (CCLR1 and CCLR0): These bits select how the timer counter is cleared: by compare-match A or B or by an external reset input.

Bit 4	Bit 3		
CCLR1	CCLR0	Description	
0	0	Not cleared.	(Initial value)
0	1	Cleared on compare-match A.	
1	0	Cleared on compare-match B.	
1	1	Cleared on rising edge of external reset input signal.	

Bits 2, 1, and 0 – Clock Select (CKS2, CKS1, and CKS0): These bits select the internal or external clock source for the timer counter. For the external clock source they select whether to increment the count on the rising or falling edge of the clock input, or on both edges. For the internal clock sources the count is incremented on the falling edge of the clock input.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	0	No clock source (timer stopped)	(Initial value)
0	0	1	Ø/8 Internal clock source, counted on the falling edge	
0	1	0	Ø/64 Internal clock source, counted on the falling edge	
0	1	1	Ø/1024 Internal clock source, counted on the falling edge	
1	0	0	No clock source (timer stopped)	
1	0	1	External clock source, counted on the rising edge	
1	1	0	External clock source, counted on the falling edge	
1	1	1	External clock source, counted on both the rising and falling edges	

7.2.4 Timer Control/Status Register (TCSR) – H'FFC9 (TMR0), H'FFD1 (TMR1)

Bit	7	6	5	4	3	2	1	0
	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	—	R/W	R/W	R/W	R/W

* Software can write a "0" in bits 7 to 5 to clear the flags, but cannot write a "1" in these bits.

The TCSR is an 8-bit readable and partially writable register that indicates compare-match and overflow status and selects the effect of compare-match events on the timer output signal.

The TCSR is initialized to H'10 at a reset and in the standby modes.

Bit 7 – Compare-Match Flag B (CMFB): This status flag is set to "1" when the timer count matches the time constant set in TCORB. CMFB must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 7 CMFB	Description	
0	To clear CMFB, the CPU must read CMFB after it has been set to "1," then write a "0" in this bit.	(Initial value)
1	This bit is set to 1 when TCNT = TCORB.	

Bit 6 – Compare-Match Flag A (CMFA): This status flag is set to "1" when the timer count matches the time constant set in TCORA. CMFA must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 6 CMFA	Description	
0	To clear CMFA, the CPU must read CMFA after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when TCNT = TCORA.	

Bit 5 – Timer Overflow Flag (OVF): This status flag is set to “1” when the timer count overflows (changes from H’FF to H’00). OVF must be cleared by software. It is set by hardware, however, and cannot be set by software.

Bit 5 OVF	Description	
0	To clear OVF, the CPU must read OVF after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when TCNT changes from H’FF to H’00.	

Bit 4 – Reserved: This bit is always read as “1.” It cannot be written.

Bits 3 to 0 – Output Select 3 to 0 (OS3 to OS0): These bits specify the effect of compare-match events on the timer output signal (TCOR or TCNT). Bits OS3 and OS2 control the effect of compare-match B on the output level. Bits OS1 and OS0 control the effect of compare-match A on the output level.

If compare-match A and B occur simultaneously, any conflict is resolved as explained in item (4) in section 7.6, "Application Notes."

After a reset, the timer output is "0" until the first compare-match event.

When all four output select bits are cleared to “0” the timer output signal is disabled.

Bit 3 OS3	Bit 2 OS2	Description	
0	0	No change when compare-match B occurs.	(Initial value)
0	1	Output changes to “0” when compare-match B occurs.	
1	0	Output changes to “1” when compare-match B occurs.	
1	1	Output inverts (toggles) when compare-match B occurs.	

Bit 1 OS1	Bit 0 OS0	Description	
0	0	No change when compare-match A occurs.	(Initial value)
0	1	Output changes to “0” when compare-match A occurs.	
1	0	Output changes to “1” when compare-match A occurs.	
1	1	Output inverts (toggles) when compare-match A occurs.	

7.3 Operation

7.3.1 TCNT Incrementation Timing

The timer counter increments on a pulse generated once for each period of the clock source selected by bits CKS2 to CKS0 of the TCR.

Internal Clock: Internal clock sources are created from the system clock by a prescaler. The counter increments on an internal TCNT clock pulse generated from the falling edge of the prescaler output, as shown in figure 7-2. Bits CKS2 to CKS0 of the TCR can select one of the three internal clocks (Ø/8, Ø/64, or Ø/1024).

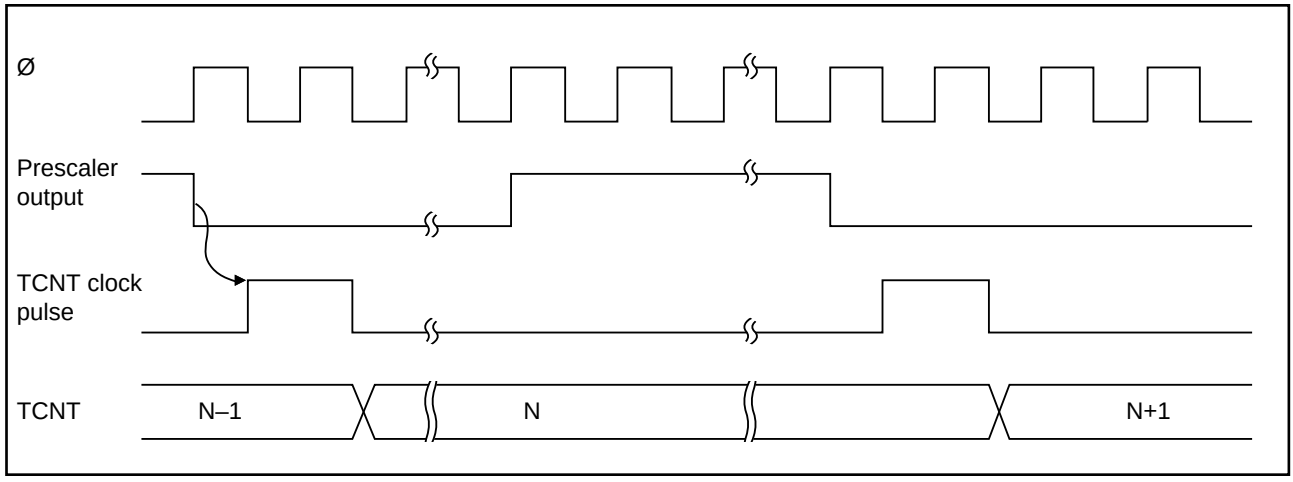


Figure 7-2. Count Timing for Internal Clock Input

External Clock: If external clock input (TMCI) is selected, the timer counter can increment on the rising edge, the falling edge, or both edges of the external clock signal. Figure 7-3 shows incrementation on both edges of the external clock signal.

The external clock pulse width must be at least 1.5 system clock periods for incrementation on a single edge, and at least 2.5 system clock periods for incrementation on both edges. See figure 7.4. The counter will not increment correctly if the pulse width is shorter than these values.

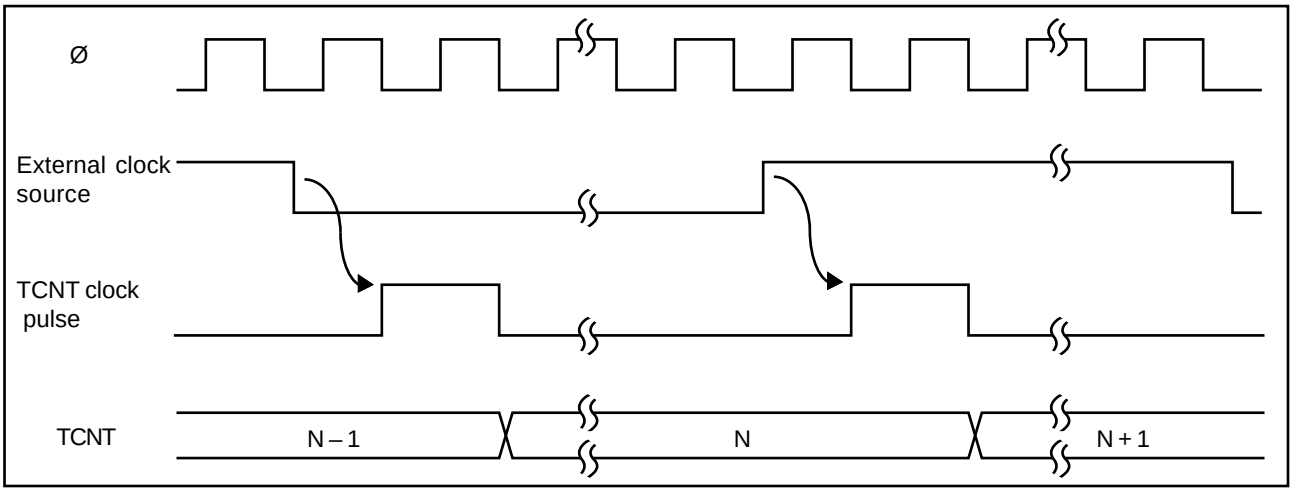


Figure 7-3. Count Timing for External Clock Input

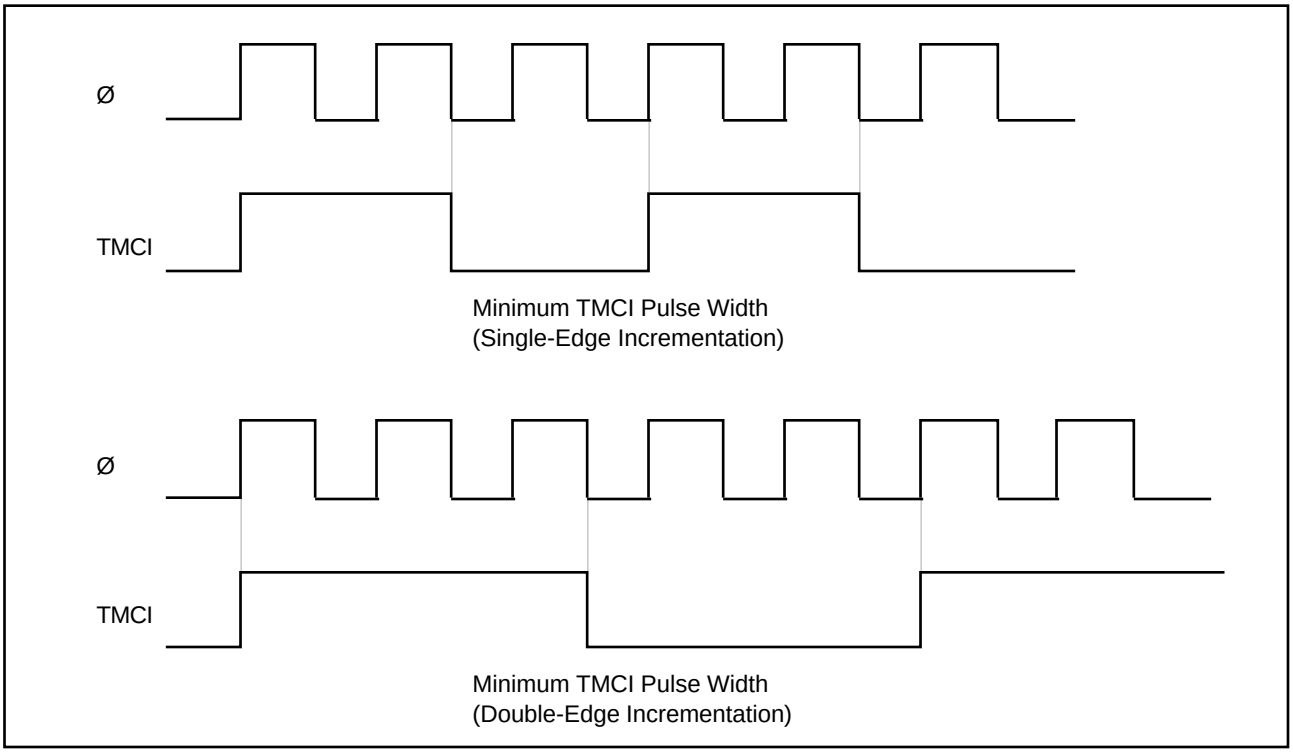


Figure 7-4. Minimum External Clock Pulse Widths (Example)

7.3.2 Compare Match Timing

(1) Setting of Compare-Match Flags A and B (CMFA and CMFB): The compare-match flags are set to "1" by an internal compare-match signal generated when the timer count matches the time constant in TCNT or TCOR. The compare-match signal is generated at the last state in which the match is true, just before the timer counter increments to a new value.

Accordingly, when the timer count matches one of the time constants, the compare-match signal is not generated until the next period of the clock source. Figure 7-5 shows the timing of the setting of the compare-match flags.

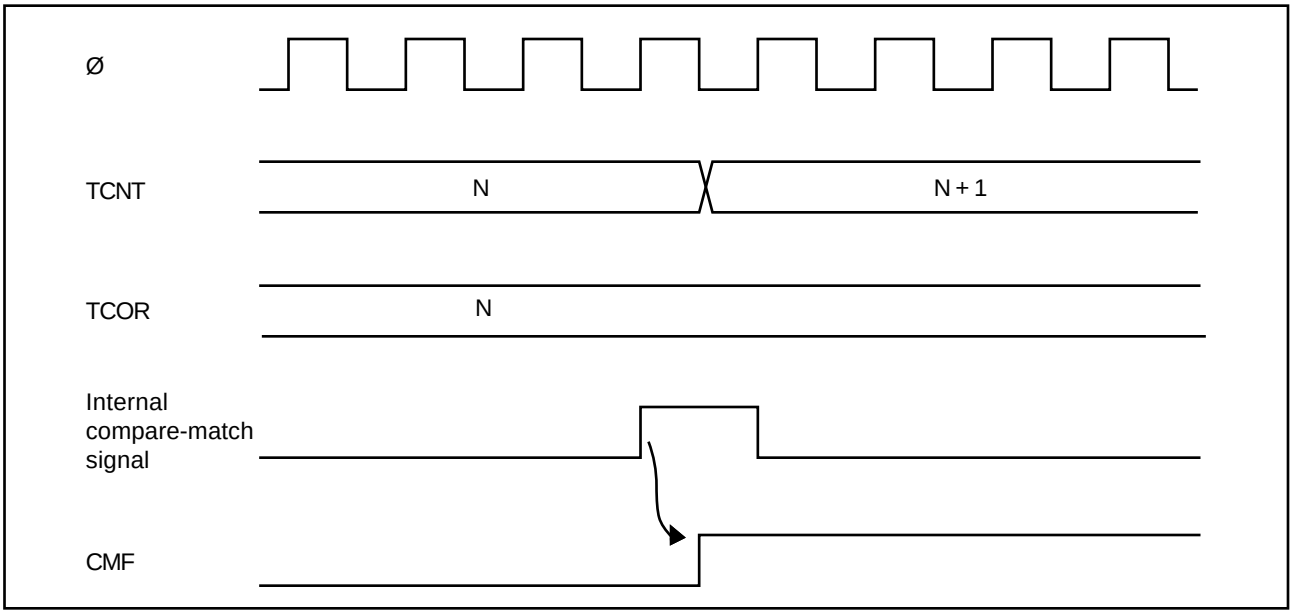


Figure 7-5. Setting of Compare-Match Flags

(2) Timing of Compare-Match Flag (CMFA or CMFB) Clearing: The compare-match flag CMFA or CMFB is cleared when the CPU writes a "0" in this bit.

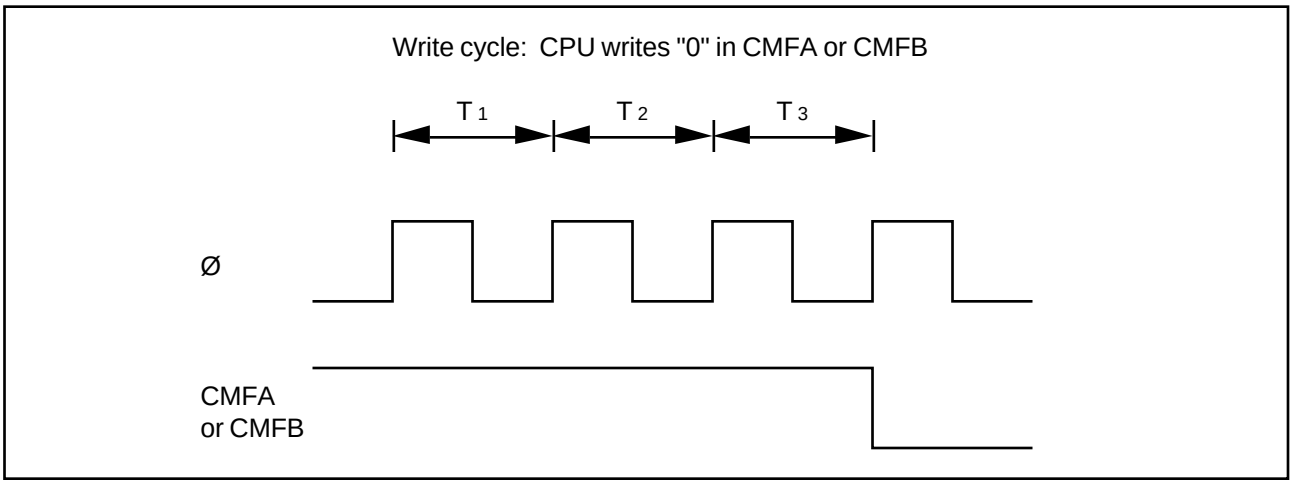


Figure 7-6. Clearing of Compare-Match Flags

(3) Output Timing: When a compare-match event occurs, the timer output (TMO0 or TMO1) changes as specified by the output select bits (OS3 to OS0) in the TCSR. Depending on these bits, the output can remain the same, change to "0," change to "1," or toggle. If compare-match A and B occur simultaneously, the higher priority compare-match determines the output level. See item (4) in section 7.6, "Application Notes" for details.

Figure 7-7 shows the timing when the output is set to toggle on compare-match A.

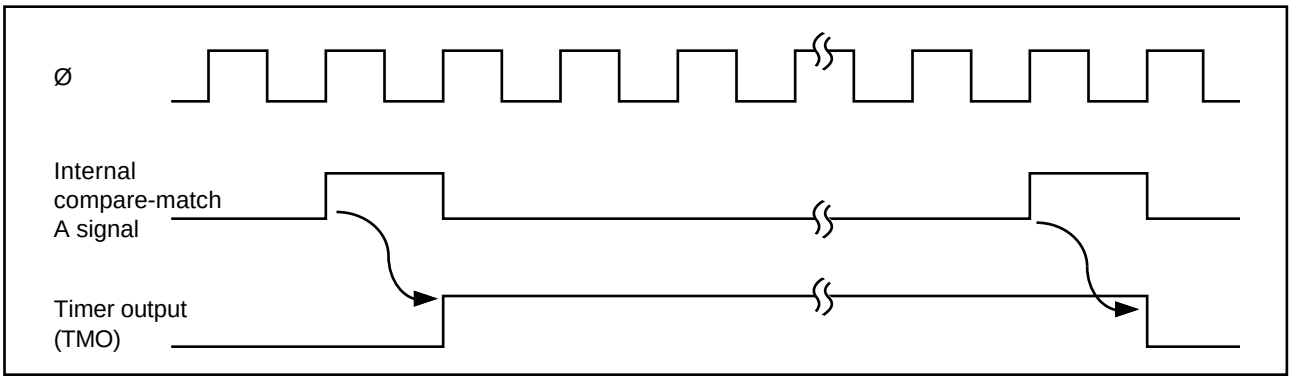


Figure 7-7. Timing of Timer Output

(4) Timing of Compare-Match Clear: Depending on the CCLR1 and CCLR0 bits in the TCR, the timer counter can be cleared when compare-match A or B occurs. Figure 7-8 shows the timing of this operation.

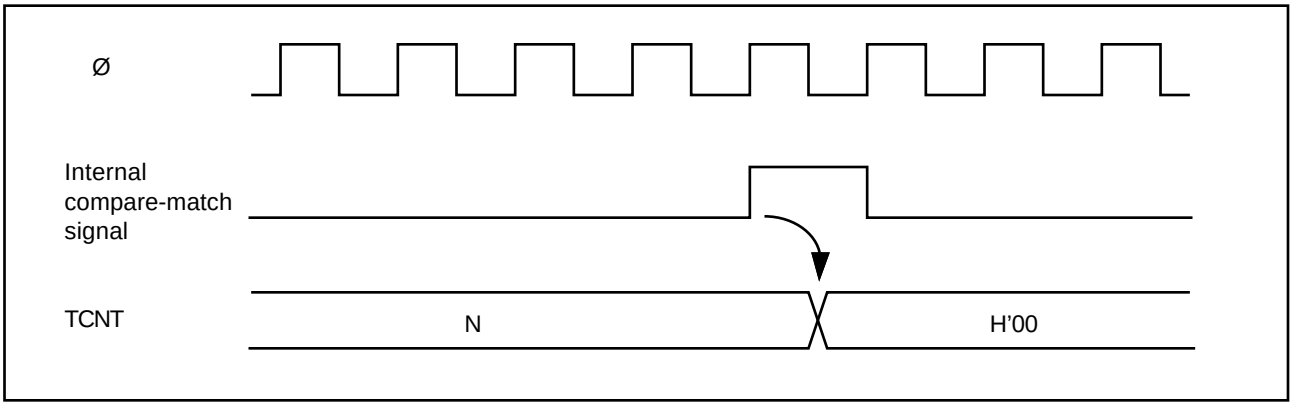


Figure 7-8. Timing of Compare-Match Clear

7.3.3 External Reset of TCNT

When the CCLR1 and CCLR0 bits in the TCR are both set to "1," the timer counter is cleared on the rising edge of an external reset input. Figure 7-9 shows the timing of this operation. The timer reset pulse width must be at least 1.5 system clock periods.

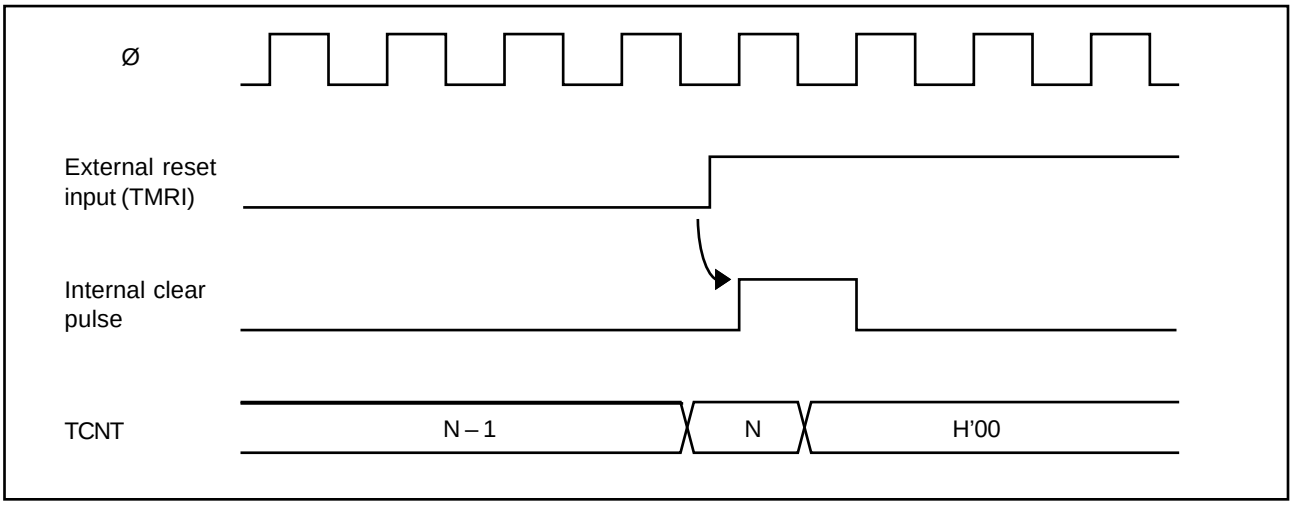


Figure 7-9. Timing of External Reset

7.3.4 Setting of TCSR Overflow Flag

(1) Setting of TCSR Overflow Flag (OVF): The overflow flag (OVF) is set to "1" when the timer count overflows (changes from H'FF to H'00). Figure 7-10 shows the timing of this operation.

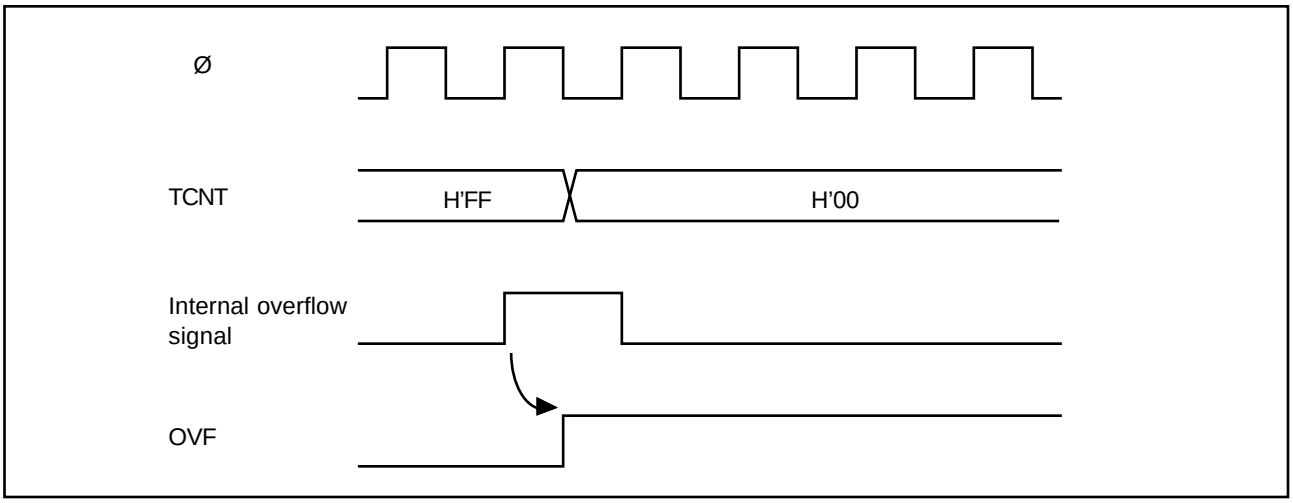


Figure 7-10. Setting of Overflow Flag (OVF)

(2) Timing of TCSR Overflow Flag (OVF) Clearing: The overflow flag (OVF) is cleared when the CPU writes a "0" in this bit.

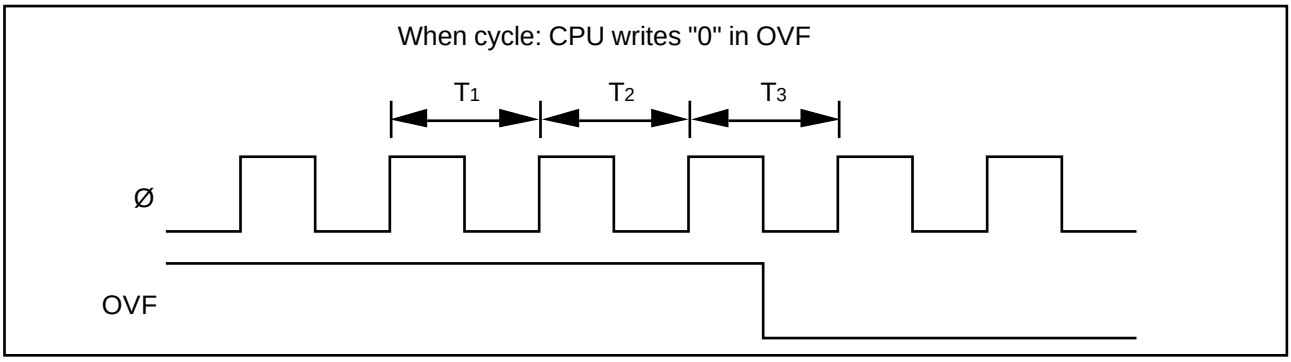


Figure 7-11. Clearing of Overflow Flag

7.4 Interrupts

Each channel in the 8-bit timer can generate three types of interrupts: compare-match A and B (CMIA and CMIB), and overflow (OVI). Each interrupt is requested when the corresponding enable bits are set in the TCR and TCSR. Independent signals are sent to the interrupt controller for each interrupt. Table 7-3 lists information about these interrupts.

Table 7-3. 8-Bit Timer Interrupts

Interrupt	Description	Priority
CMIA	Requested when CMFA and CMIEA are set	High
CMIB	Requested when CMFB and CMIEB are set	↕
OVI	Requested when OVF and OVIE are set	Low

7.5 Sample Application

In the example below, the 8-bit timer is used to generate a pulse output with a selected duty factor. The control bits are set as follows:

(1) In the TCR, CCLR1 is cleared to “0” and CCLR0 is set to “1” so that the timer counter is cleared when its value matches the constant in TCORA.

(2) In the TCSR, bits OS3 to OS0 are set to “0110,” causing the output to change to “1” on compare-match A and to “0” on compare-match B.

With these settings, the 8-bit timer provides output of pulses at a rate determined by TCORA with a pulse width determined by TCORB. No software intervention is required.

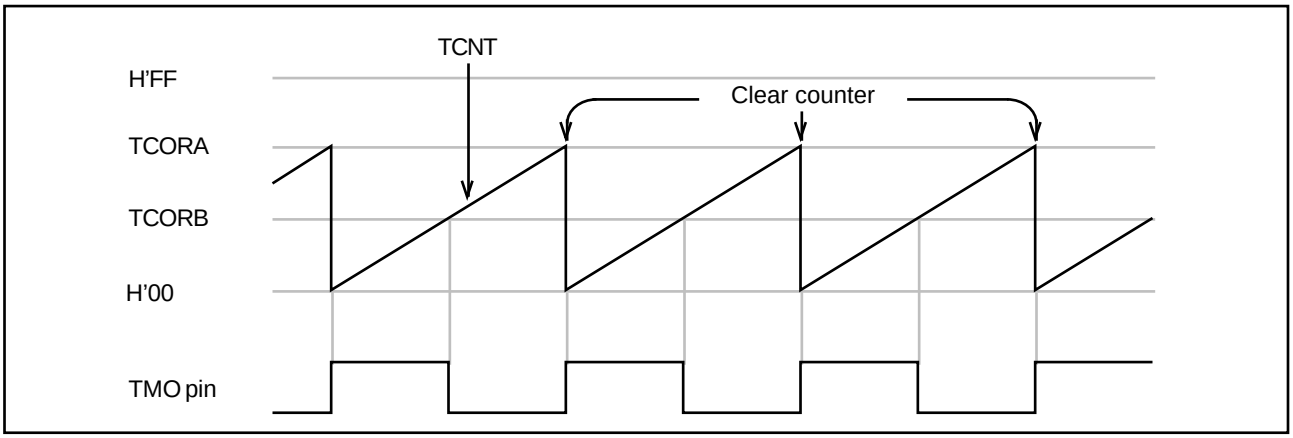


Figure 7-12. Example of Pulse Output

7.6 Application Notes

Application programmers should note that the following types of contention can occur in the 8-bit timer.

(1) Contention between TCNT Write and Clear: If an internal counter clear signal is generated during the T_3 state of a write cycle to the timer counter, the clear signal takes priority and the write is not performed.

Figure 7-13 shows this type of contention.

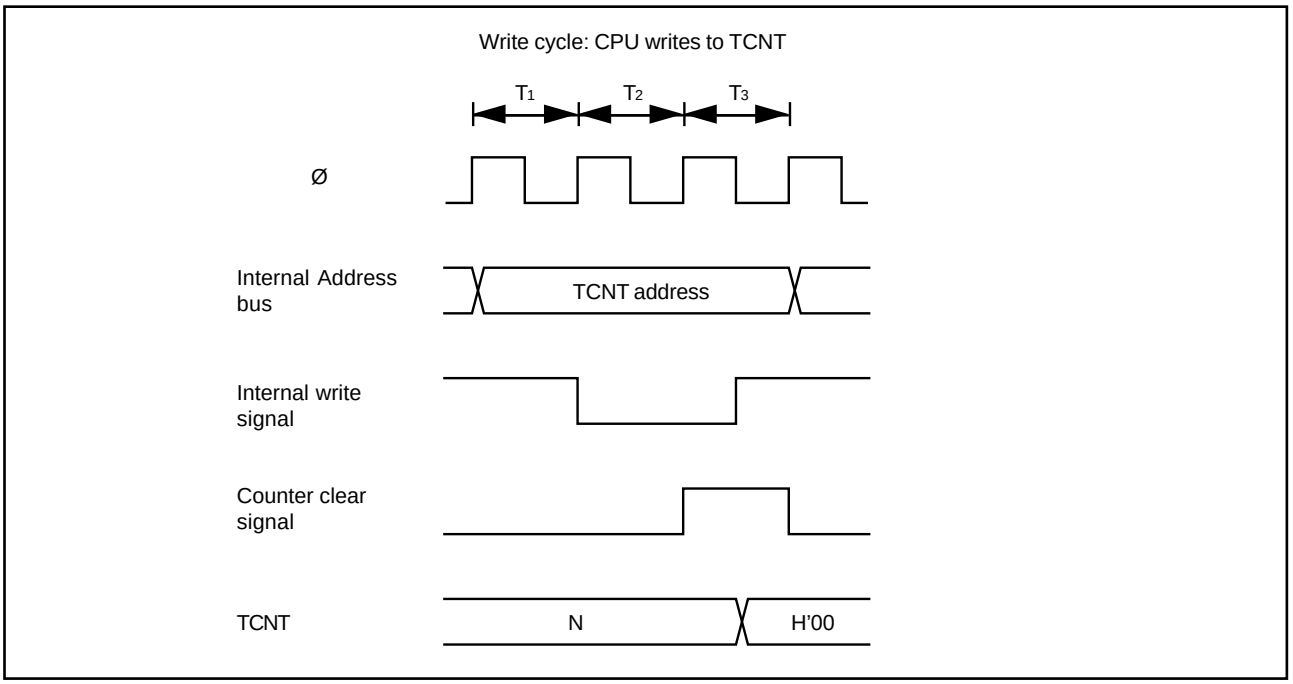


Figure 7-13. TCNT Write-Clear Contention

(2) Contention between TCNT Write and Increment: If a timer counter increment pulse is generated during the T_3 state of a write cycle to the timer counter, the write takes priority and the timer counter is not incremented.

Figure 7-14 shows this type of contention.

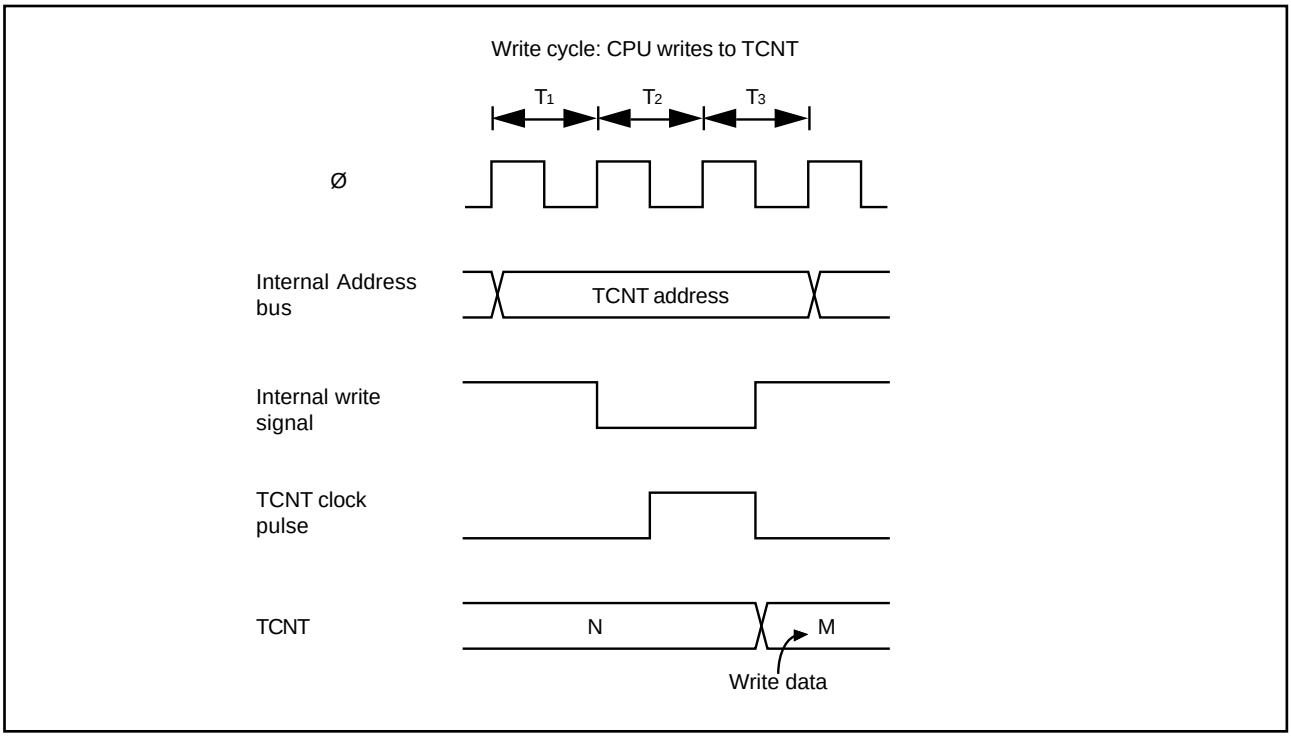


Figure 7-14. TCNT Write-Increment Contention

(3) Contention between TCOR Write and Compare-Match: If a compare-match occurs during the T_3 state of a write cycle to TCORA or TCORB, the write takes precedence and the compare-match signal is inhibited.

Figure 7-15 shows this type of contention.

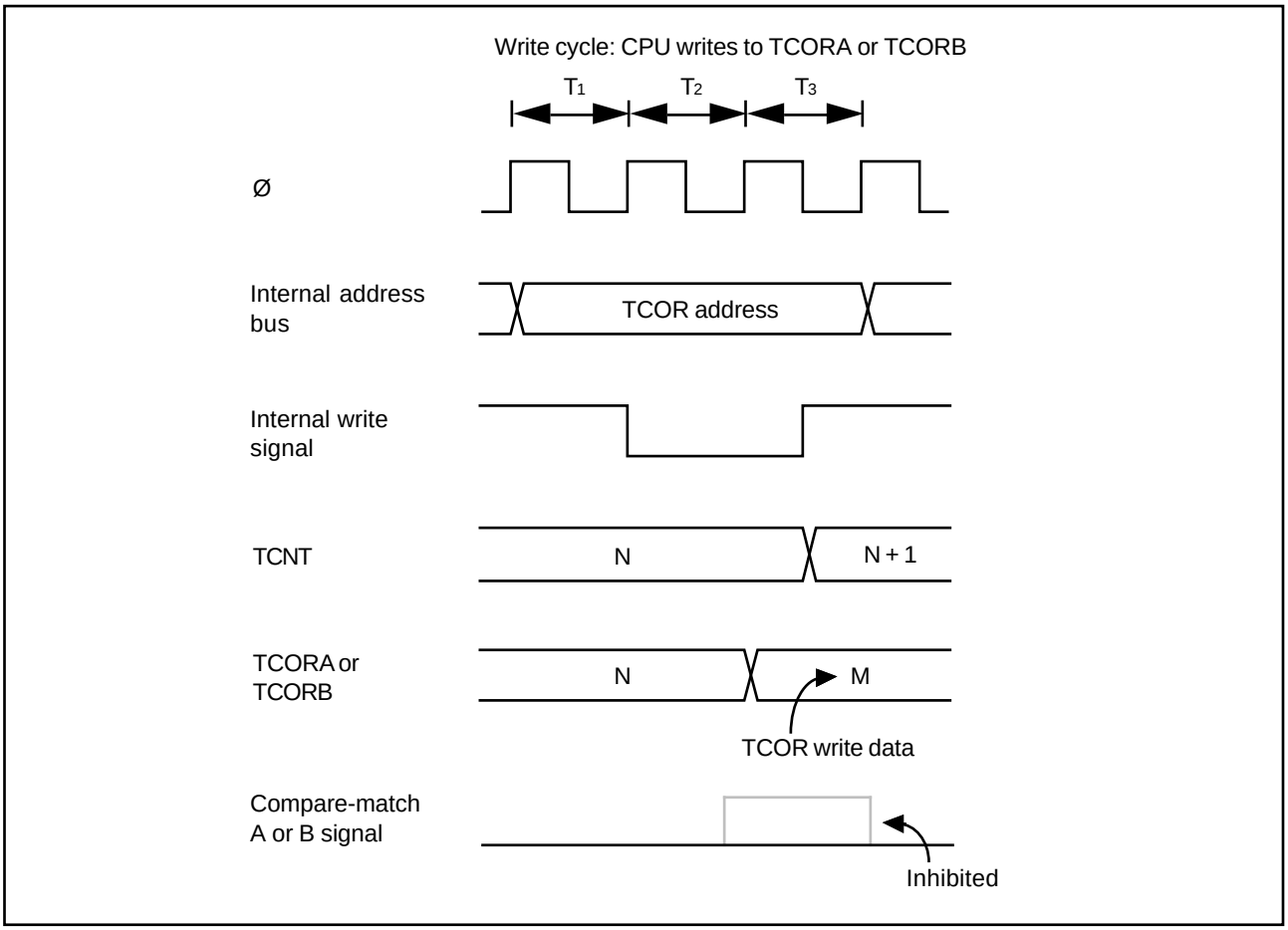


Figure 7-15. Contention between TCOR Write and Compare-Match

(4) Contention between Compare-Match A and Compare-Match B: If identical time constants are written in TCORA and TCORB, causing compare-match A and B to occur simultaneously, any conflict between the output selections for compare-match A and B is resolved by following the priority order in Table 7-4.

Table 7-4. Priority of Timer Output

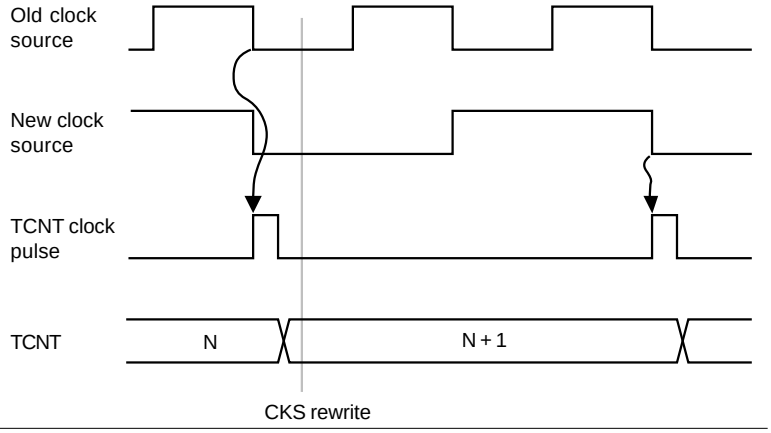
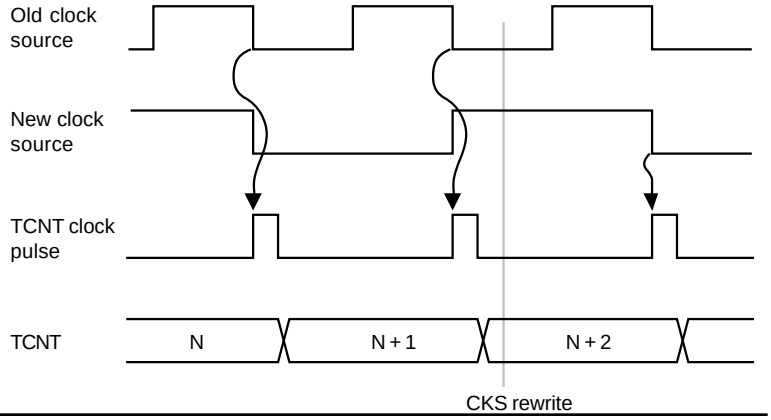
Output selection	Priority
Toggle	High
“1” Output	↑
“0” Output	
No change	Low

(5) Incrementation Caused by Changing of Internal Clock Source: When an internal clock source is changed, the changeover may cause the timer counter to increment. This depends on the time at which the clock select bits (CKS2 to CKS0) are rewritten, as shown in Table 7-5.

The pulse that increments the timer counter is generated at the falling edge of the internal clock source signal. If clock sources are changed when the old source is High and the new source is Low, as in case No. 3 in Table 7-5, the changeover generates a falling edge that triggers the TCNT clock pulse and increments the timer counter.

Switching between an internal and external clock source can also cause the timer counter to increment. This type of switching should be avoided at external clock edges.

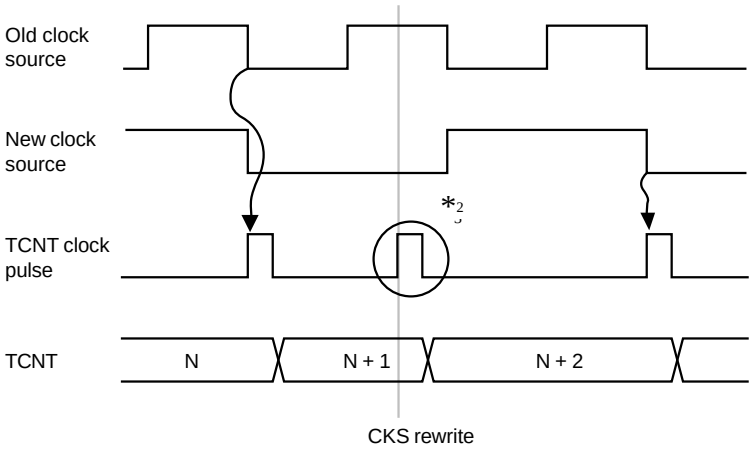
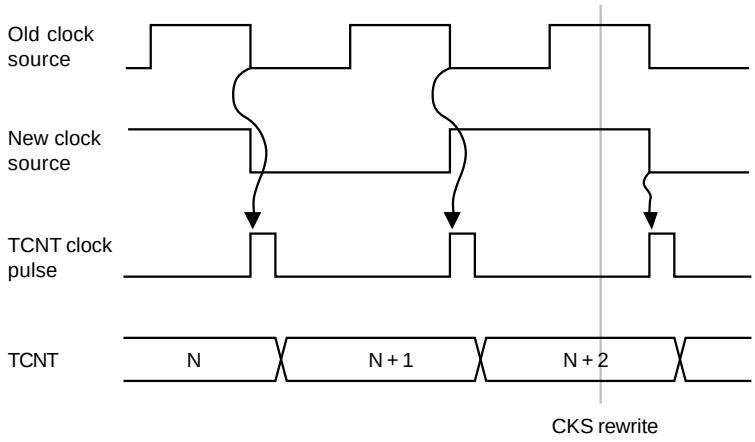
Table 7-5. Effect of Changing Internal Clock Sources

No.	Description	Timing chart
1	Low → Low*1: CKS1 and CKS0 are rewritten while both clock sources are Low.	
2	Low → High*2: CKS1 and CKS0 are rewritten while old clock source is Low and new clock source is High.	

*1 Including a transition from Low to the stopped state (CKS1 = 0, CKS0 = 0), or a transition from the stopped state to Low.

*2 Including a transition from the stopped state to High.

Table 7-5. Effect of Changing Internal Clock Sources (cont.)

No.	Description	Timing chart
3	High → Low*1: CKS1 and CKS0 are rewritten while old clock source is High and new clock source is Low.	
4	High → High: CKS1 and CKS0 are rewritten while both clock sources are High.	

*1 Including a transition from High to the stopped state.

*2 The switching of clock sources is regarded as a falling edge that increments the TCNT.

Section 8. PWM Timers

8.1 Overview

The H8/330 has an on-chip pulse-width modulation (PWM) timer module with two independent channels (PWM0 and PWM1). Both channels are functionally identical. Each PWM channel generates a rectangular output pulse with a duty factor of 0 to 100%. The duty factor is specified in an 8-bit duty register (DTR).

8.1.1 Features

The PWM timer module has the following features:

- Selection of eight clock sources
- Duty factors from 0 to 100% with 1/250 resolution
- Output with positive or negative logic and software enable/disable control

8.1.2 Block Diagram

Figure 8-1 shows a block diagram of one PWM timer channel.

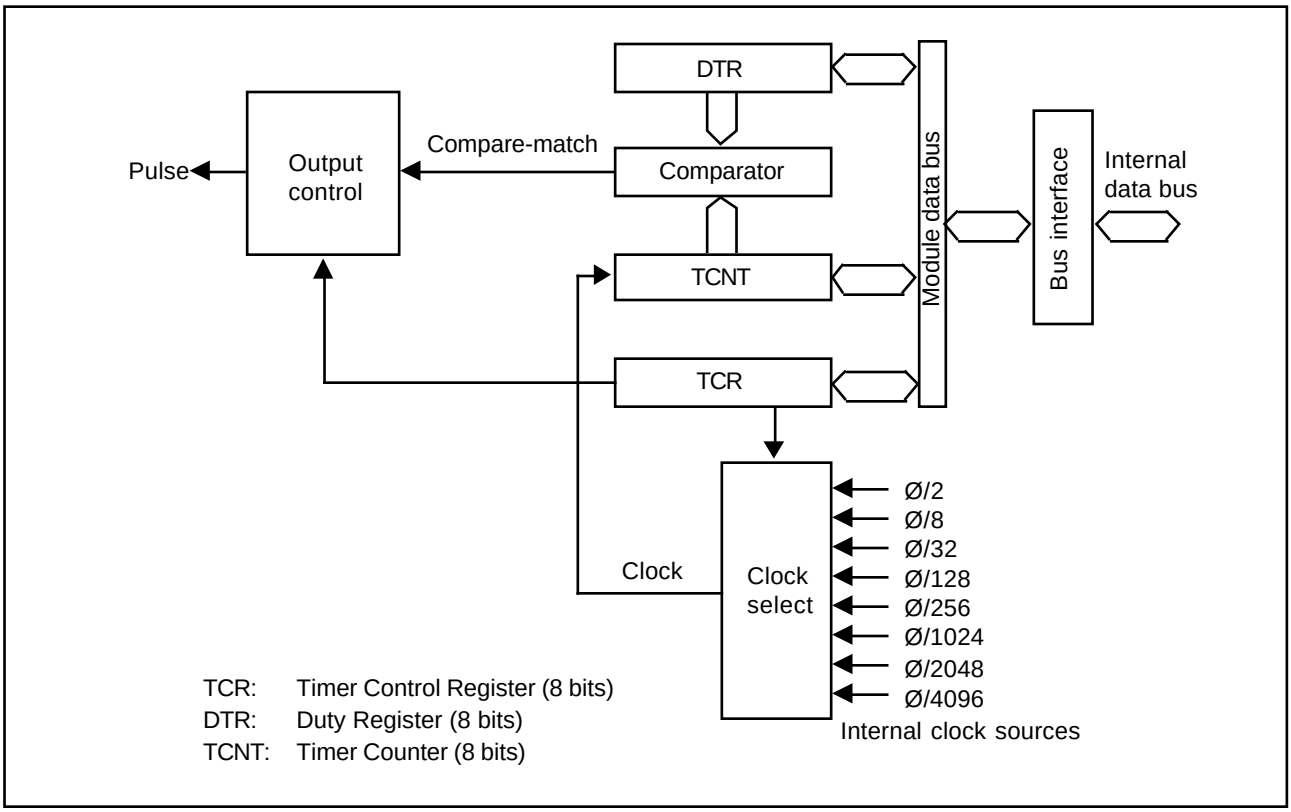


Figure 8-1. Block Diagram of PWM Timer

8.1.3 Input and Output Pins

Table 8-1 lists the output pins of the PWM timer module. There are no input pins.

Table 8-1. Output Pins of PWM Timer Module

Name	Abbreviation	I/O	Function
PWM0 output	PW0	Output	Pulse output from PWM timer channel 0.
PWM1 output	PW1	Output	Pulse output from PWM timer channel 1.

8.1.4 Register Configuration

The PWM timer module has three registers for each channel as listed in Table 8-2.

Table 8-2. PWM Timer Registers

Name	Abbreviation	R/W	Initial value	Address PWM0	Address PWM1
Timer control register	TCR	R/W	H’38	H’FFA0	H’FFA4
Duty register	DTR	R/W	H’FF	H’FFA1	H’FFA5
Timer counter	TCNT	R/(W)*	H’00	H’FFA2	H’FFA6

* The timer counters are read/write registers, but the write function is for test purposes only. Application programs should never write to these registers.

8.2 Register Descriptions

8.2.1 Timer Counter (TCNT) – H’FFA2 (PWM0), H’FFA6 (PWM1)

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

The PWM timer counters (TCNT) are 8-bit up-counters. When the output enable bit (OE) in the timer control register (TCR) is set to “1,” the timer counter starts counting pulses of an internal clock source selected by clock select bits 2 to 0 (CKS2 to CKS0). After counting from H’00 to H’F9, the timer counter repeats from H’00.

The PWM timer counters can be read and written, but the write function is for test purposes only. Application software should never write to a PWM timer counter, because this may have unpredictable effects.

The PWM timer counters are initialized to H'00 at a reset and in the standby modes, and when the OE bit is cleared to "0."

8.2.2 Duty Register (DTR) – H'FFA1 (PWM0), H'FFA5 (PWM1)

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

The duty registers (DTR) are 8-bit readable/writable registers that specify the duty factor of the output pulse. Any duty factor from 0 to 100% can be selected, with a resolution of 1/250. Writing 0 (H'00) in a DTR gives a 0% duty factor; writing 125 (H'7D) gives a 50% duty factor; writing 250 (H'FA) gives a 100% duty factor.

The timer count is continually compared with the DTR contents. If the DTR value is not 0, when the count increments from H'00 to H'01 the PWM output signal is set to "1." When the count increments past the DTR value, the PWM output returns to "0." If the DTR value is 0 (duty factor 0%), the PWM output remains constant at "0."

The DTRs are double-buffered. A new value written in a DTR while the timer counter is running does not become valid until after the count changes from H'F9 to H'00. When the timer counter is stopped (while the OE bit is "0"), new values become valid as soon as written. When a DTR is read, the value read is the currently valid value.

The DTRs are initialized to H'FF at a reset and in the standby modes.

8.2.3 Timer Control Register (TCR) – H'FFA0 (PWM0), H'FFA4 (PWM1)

Bit	7	6	5	4	3	2	1	0
	OE	OS	—	—	—	CKS2	CKS1	CKS0
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

The TCRs are 8-bit readable/writable registers that select the clock source and control the PWM outputs.

The TCRs are initialized to H’38 at a reset and in the standby modes.

Bit 7 – Output Enable (OE): This bit enables the timer counter and the PWM output.

Bit 7 OE	Description	
0	PWM output is disabled. TCNT is cleared to H’00 and stopped.	(Initial value)
1	PWM output is enabled. TCNT runs.	

Bit 6 – Output Select (OS): This bit selects positive or negative logic for the PWM output.

Bit 6 OS	Description	
0	Positive logic; positive-going PWM pulse, “1” = High	(Initial value)
1	Negative logic; negative-going PWM pulse, “1” = Low	

Bits 5 to 3 – Reserved: These bits cannot be modified and are always read as “1.”

Bits 2, 1, and 0 – Clock Select (CKS2, CKS1, and CKS0): These bits select one of eight internal clock sources obtained by dividing the system clock (Ø).

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	0	Ø/2	(Initial value)
0	0	1	Ø/8	
0	1	0	Ø/32	
0	1	1	Ø/128	
1	0	0	Ø/256	
1	0	1	Ø/1024	
1	1	0	Ø/2048	
1	1	1	Ø/4096	

From the clock source frequency, the resolution, period, and frequency of the PWM output can be calculated as follows.

Resolution = 1/clock source frequency
PWM period = resolution × 250
PWM frequency = 1/PWM period

If the system clock frequency is 10MHz, then the resolution, period, and frequency of the PWM output for each clock source are given in Table 8-3.

Table 8-3. PWM Timer Parameters for 10MHz System Clock

Internal clock frequency	Resolution	PWM period	PWM frequency
Ø/2	200ns	50µs	20kHz
Ø/8	800ns	200µs	5kHz
Ø/32	3.2µs	800µs	1.25kHz
Ø/128	12.8µs	3.2ms	312.5Hz
Ø/256	25.6µs	6.4ms	156.3Hz
Ø/1024	102.4µs	25.6ms	39.1Hz
Ø/2048	204.8µs	51.2ms	19.5Hz
Ø/4096	409.6µs	102.4ms	9.8Hz

8.3 Operation

8.3.1 Timer Incrementation

The PWM clock source is created from the system clock (Ø) by a prescaler. The timer counter increments on a TCNT clock pulse generated from the falling edge of the prescaler output as shown in Figure 8-2.

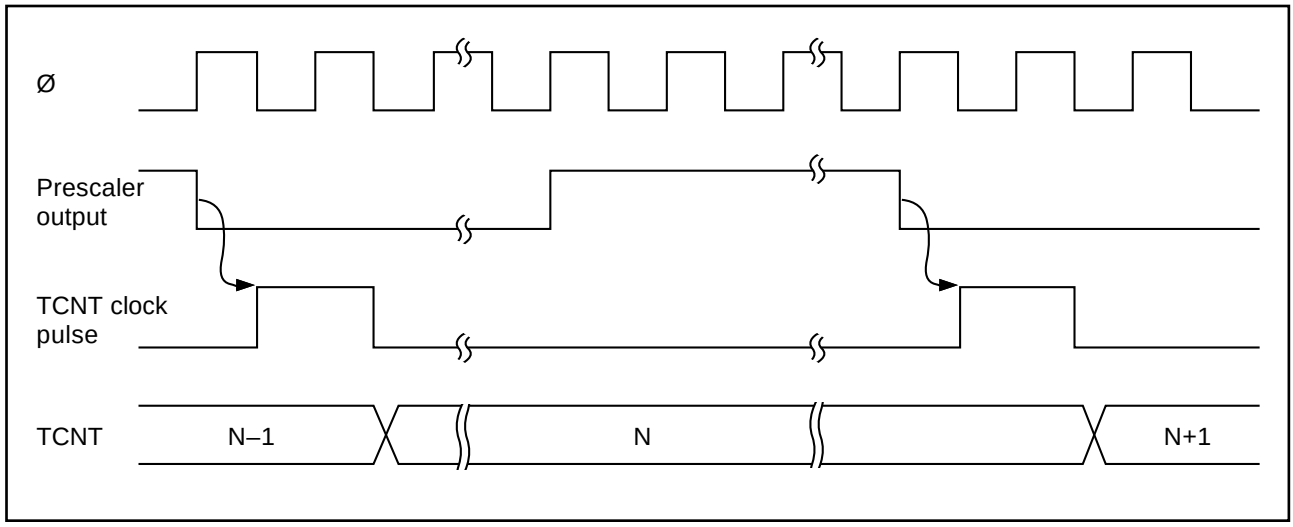


Figure 8-2. TCNT Increment Timing

8.3.2 PWM Operation

Figure 8-3 is a timing chart of the PWM operation.

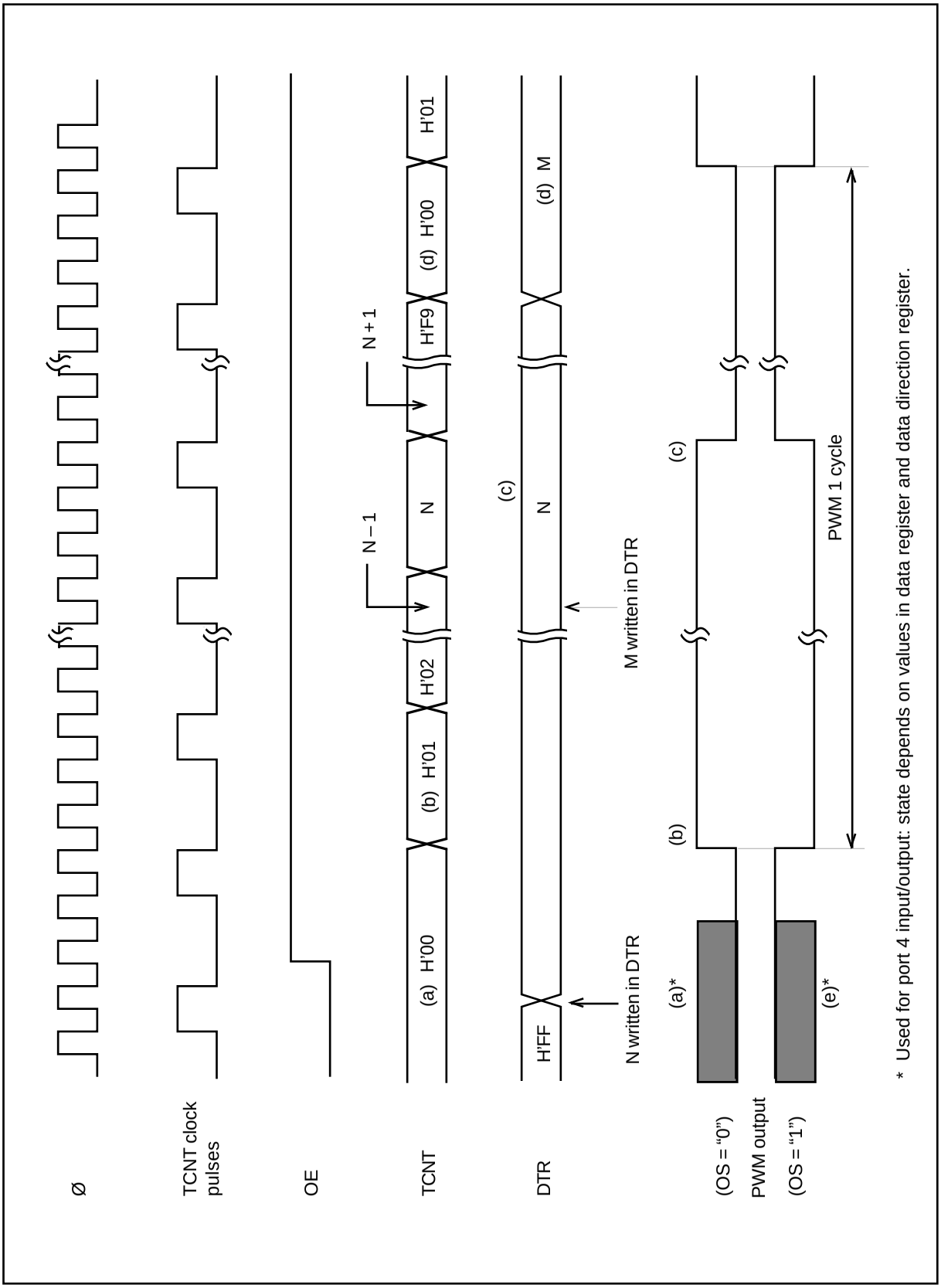


Figure 8-3. PWM Timing

(1) Positive Logic (OS = “0”)

(1) When (OE = “0”) – (a) in Figure 8-3: The timer count is held at H’00 and PWM output is inhibited. (Pin 46 (for PW0) or pin 47 (for PW1)is used for port 4 input/output, and its state depends on the corresponding port 4 data register and data direction register.) Any value (such as N in Figure 8-3) written in the DTR becomes valid immediately.

(2) When (OE = “1”)

i) The timer counter begins incrementing. The PWM output goes High when TCNT changes from H’00 to H’01, unless DTR = H’00. [(b) in Figure 8-3]

ii) When the count passes the DTR value, the PWM output goes Low. [(c) in Figure 8-3]

iii) If the DTR value is changed (by writing the data “M” in Figure 8-3), the new value becomes valid after the timer count changes from H’F9 to H’00. [(d) in Figure 8-3]

(2) Negative Logic (OS = “1”) – (e) in Figure 8-3: The operation is the same except that High and Low are reversed in the PWM output . [(e) in Figure 8-3]

8.4 Application Notes

Some notes on the use of the PWM timer module are given below.

(1) Any necessary changes to the clock select bits (CKS2 to CKS0) and output select bit (OS) should be made before the output enable bit (OE) is set to “1.”

(2) If the DTR value is H’00, the duty factor is 0% and PWM output remains constant at “0.” If the DTR value is H’FA to H’FF, the duty factor is 100% and PWM output remains constant at “1.”
(For positive logic, “0” is Low and “1” is High. For negative logic, “0” is High and “1” is Low.)

(3) When the DTR is read, the currently valid value is obtained. Due to the double buffering, this may not be the value most recently written.

(4) Software should never write to a PWM timer counter. The write function is for test purposes only and may have unintended effects in normal operation.

Section 9. Serial Communication Interface

9.1 Overview

The H8/330 chip includes a single-channel serial communication interface (SCI) for transferring serial data to and from other chips. Either the synchronous or asynchronous communication mode can be selected. Communication control functions are provided by internal registers.

9.1.1 Features

The features of the on-chip serial communication interface are:

- Separate pins for asynchronous and synchronous modes
  - Asynchronous mode
    The SCI can communicate with a UART (Universal Asynchronous Receiver/Transmitter), ACIA (Asynchronous Communication Interface Adapter), or other chip that employs standard asynchronous serial communication. Eight data formats are available.
    - Data length: 7 or 8 bits
    - Stop bit length: 1 or 2 bits
    - Parity: Even, odd, or none
    - Error detection: Parity, overrun, and framing errors
  - Synchronous mode
    The SCI can communicate with chips able to perform clocked serial data transfer.
    - Data length: 8 bits
    - Error detection: Overrun errors
- Full duplex communication
  The transmitting and receiving sections are independent, so the SCI can transmit and receive simultaneously. Both the transmit and receive sections use double buffering, so continuous data transfer is possible in either direction.
- Built-in baud rate generator
  Any specified baud rate can be generated.
- Internal or external clock source
  The baud rate generator can operate on an internal clock source, or an external clock signal input at the ASCK or CSCK pin.
- Three interrupts
  Transmit-end, receive-end, and receive-error interrupts are requested independently.

9.1.2 Block Diagram

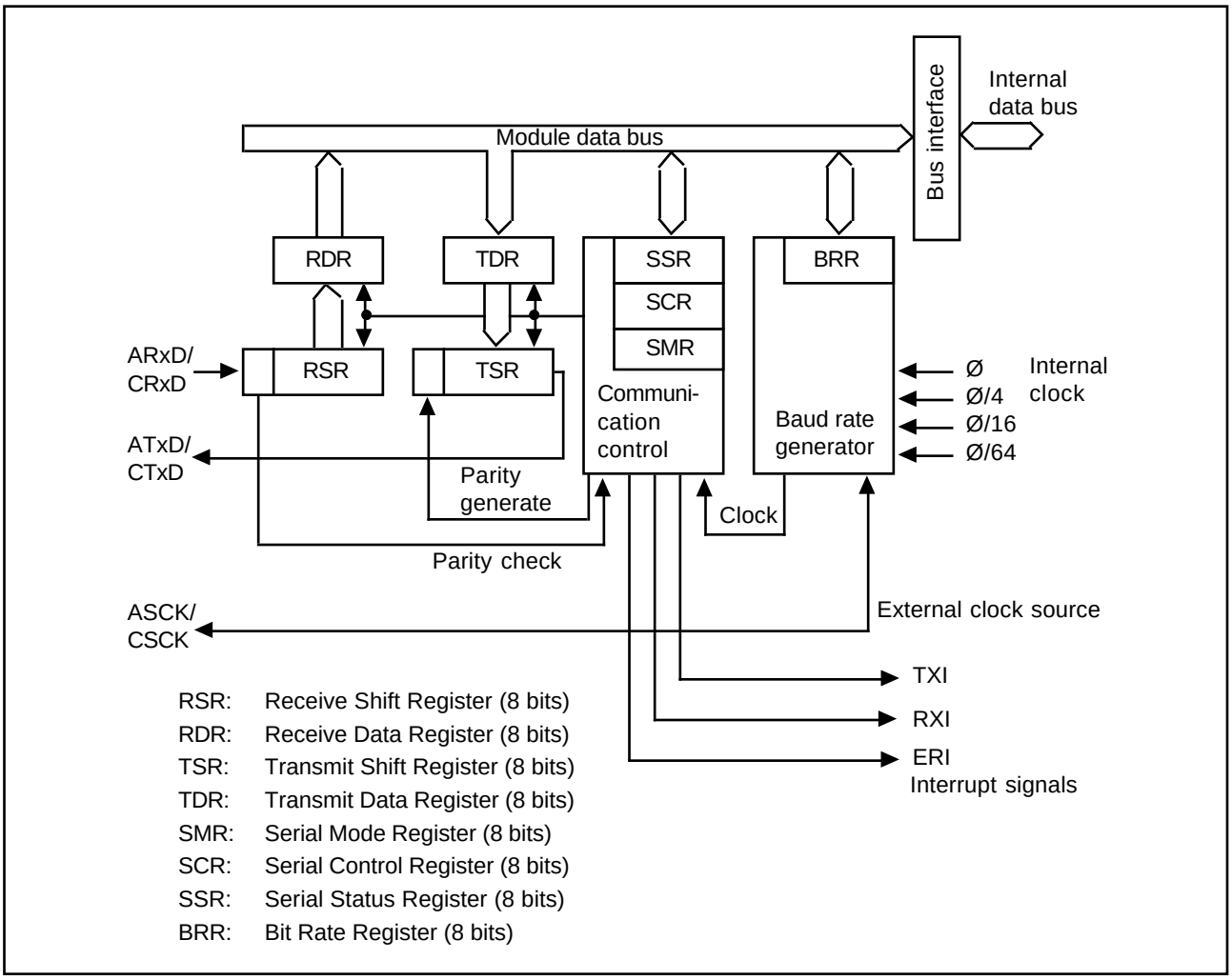


Figure 9-1. Block Diagram of Serial Communication Interface

9.1.3 Input and Output Pins

Table 9-1 lists the input and output pins used by the SCI module.

Table 9-1. SCI Input/Output Pins

Name	Abbreviation	I/O	Function
Asynchronous serial clock	ASCK	Input/output	Serial clock input and output.
Asynchronous receive data	ARxD	Input	Receive data input.
Asynchronous transmit data	ATxD	Output	Transmit data output.
Synchronous serial clock	CSCK	Input/output	Serial clock input and output.
Synchronous receive data	CRxD	Input	Receive data input.
Synchronous transmit data	CTxD	Output	Transmit data output.

9.1.4 Register Configuration

Table 9-2 lists the SCI registers.

Table 9-2. SCI Registers

Name	Abbreviation	R/W	Initial value	Address
Receive shift register	RSR	—	—	—
Receive data register	RDR	R	H’00	H’FFDD
Transmit shift register	TSR	—	—	—
Transmit data register	TDR	R/W	H’FF	H’FFDB
Serial mode register	SMR	R/W	H’04	H’FFD8
Serial control register	SCR	R/W	H’0C	H’FFDA
Serial status register	SSR	R/(W)*	H’87	H’FFDC
Bit rate register	BRR	R/W	H’FF	H’FFD9

Notes:

* Software can write a “0” to clear the status flag bits, but cannot write a “1.”

9.2 Register Descriptions

9.2.1 Receive Shift Register (RSR)

Bit	7	6	5	4	3	2	1	0
Read/Write	—	—	—	—	—	—	—	—

The RSR receives incoming data bits. When one data character (1 byte) has been received, it is transferred to the receive data register (RDR).

The CPU cannot read or write the RSR directly.

9.2.2 Receive Data Register (RDR) – H'FFDD

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

The RDR stores received data. As each character is received, it is transferred from the RSR to the RDR, enabling the RSR to receive the next character. This double-buffering allows the SCI to receive data continuously.

The CPU can read but not write the RDR. The RDR is initialized to H'00 at a reset and in the standby modes.

9.2.3 Transmit Shift Register (TSR)

Bit	7	6	5	4	3	2	1	0
Read/Write	—	—	—	—	—	—	—	—

The TSR holds the character currently being transmitted. When transmission of this character is completed, the next character is moved from the transmit data register (TDR) to the TSR and transmission of that character begins. If the CPU has not written the next character in the TDR, no data are transmitted.

The CPU cannot read or write the TSR directly.

9.2.4 Transmit Data Register (TDR) – H'FFDB

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

The TDR is an 8-bit readable/writable register that holds the next character to be transmitted. When the TSR becomes empty, the character written in the TDR is transferred to the TSR. Continuous data transmission is possible by writing the next byte in the TDR while the current byte is being transmitted from the TSR.

The TDR is initialized to H’FF at a reset and in the standby modes.

9.2.5 Serial Mode Register (SMR) – H’FFD8

Bit	7	6	5	4	3	2	1	0
	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	CKS1	CKS0
Initial value	0	0	0	0	0	1	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	—	R/W	R/W

The SMR is an 8-bit readable/writable register that controls the communication format and selects the clock rate for the internal clock source. It is initialized to H’04 at a reset and in the standby modes.

Bit 7 – Communication Mode (C/$\overline{A}$): This bit selects the asynchronous or synchronous communication mode.

Bit 7

C/$\overline{A}$	**Description**	
0	Asynchronous communication.	(Initial value)
1	Clock-synchronized communication.	

Bit 6 – Character Length (CHR): This bit selects the character length in asynchronous mode. It is ignored in synchronous mode.

Bit 6

CHR	**Description**	
0	8 Bits per character.	(Initial value)
1	7 Bits per character.	

Bit 5 – Parity Enable (PE): This bit selects whether to add a parity bit in asynchronous mode. It is ignored in synchronous mode.

Bit 5

PE	**Description**	
0	Transmit: No parity bit is added. Receive: Parity is not checked.	(Initial value)
1	Transmit: A parity bit is added. Receive: Parity is checked.	

Bit 4 – Parity Mode (O/$\overline{E}$): In asynchronous mode, when parity is enabled (PE = “1”), this bit selects even or odd parity.

Even parity means that a parity bit is added to the data bits for each character to make the total number of 1’s even. Odd parity means that the total number of 1’s is made odd.

This bit is ignored when PE = “0,” and in the synchronous mode.

Bit 4

O/$\overline{E}$	**Description**	
0	Even parity.	(Initial value)
1	Odd parity.	

Bit 3 – Stop Bit Length (STOP): This bit selects the number of stop bits. It is ignored in the synchronous mode.

Bit 3

STOP	**Description**	
0	1 Stop bit.	(Initial value)
1	2 Stop bits.	

Bit 2 – Reserved: This bit cannot be modified and is always read as “1.”

Bits 1 and 0 – Clock Select 1 and 0 (CKS1 and CKS0): These bits select the internal clock source when the baud rate generator is clocked from within the H8/330 chip.

Bit 1	**Bit 0**		
CKS1	**CKS0**	**Description**	
0	0	Ø clock	(Initial value)
0	1	Ø/4 clock	
1	0	Ø/16 clock	
1	1	Ø/64 clock	

For further information about SMR settings, see Tables 9-5 to 9-7 in Section 9.3, "Operation."

9.2.6 Serial Control Register (SCR) – H'FFDA

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	—	—	CKE1	CKE0
Initial value	0	0	0	0	1	1	0	0
Read/Write	R/W	R/W	R/W	R/W	—	—	R/W	R/W

The SCR is an 8-bit readable/writable register that enables or disables various SCI functions. It is initialized to H'0C at a reset and in the standby modes.

Bit 7 – Transmit Interrupt Enable (TIE): This bit enables or disables the transmit-end interrupt (TxI) requested when the transmit data register empty (TDRE) bit in the serial status register (SSR) is set to "1."

Bit 7

TIE	Description	
0	The transmit-end interrupt request (TxI) is disabled.	(Initial value)
1	The transmit-end interrupt request (TxI) is enabled.	

Bit 6 – Receive Interrupt Enable (RIE): This bit enables or disables the receive-end interrupt (RxI) requested when the receive data register full (RDRF) bit in the serial status register (SSR) is set to "1."

Bit 6

RIE	Description	
0	The receive-end interrupt (RxI) request is disabled.	(Initial value)
1	The receive-end interrupt (RxI) request is enabled.	

Bit 5 – Transmit Enable (TE): This bit enables or disables the transmit function. When the transmit function is enabled, the ATxD or CTxD pin is automatically used for output. When the transmit function is disabled, the ATxD or CTxD pin can be used as a general-purpose I/O port.

Bit 5

TE	Description	
0	The transmit function is disabled. The ATxD and CTxD pins can be used for general-purpose I/O.	(Initial value)
1	The transmit function is enabled. When $C/\overline{A} = 0$, the ATxD pin is used for output. When $C/\overline{A} = 1$, the CTxD pin is used for output.	

Bit 4 – Receive Enable (RE): This bit enables or disables the receive function. When the receive function is enabled, the ARxD or CRxD pin is automatically used for input. When the receive function is disabled, the ARxD or CRxD pin is available as a general-purpose I/O port.

Bit 4

RE	Description	
0	The receive function is disabled. The ARxD and CRxD pins can be used for general-purpose I/O.	(Initial value)
1	The receive function is enabled. When C/$\overline{A}$ = 0, the ARxD pin is used for input. When C/$\overline{A}$ = 1, the CRxD pin is used for input.	

Bits 3 and 2 – Reserved: These bits cannot be modified and are always read as "1."

Bit 1 – Clock Enable 1 (CKE1): This bit selects the internal or external clock source for the baud rate generator. When the external clock source is selected, the ASCK or CSCK pin is automatically used for input of the external clock signal.

Bit 1

CKE1	Description	
0	Internal clock source. (When C/$\overline{A}$ = 1, the CSCK pin is used for output.)	(Initial value)
1	External clock source. (When C/$\overline{A}$ = 1, the CSCK pin is used for input. When C/$\overline{A}$ = 0, the ASCK pin is used for input.)	

Bit 0 – Clock Enable 0 (CKE0): When an internal clock source is used in asynchronous mode, this bit enables or disables serial clock output at the ASCK pin.

This bit is ignored when the external clock is selected, or when the synchronous mode is selected.

Bit 0

CKE0	Description	
0	The ASCK pin is not used by the SCI (and is available as a general-purpose I/O port).	(Initial value)
1	The ASCK pin is used for serial clock output.	

For further information on clock source selection, see Table 9-6 in Section 9.3, "Operation."

9.2.7 Serial Status Register (SSR) – H'FFDC

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	—	—	—
Initial value	1	0	0	0	0	1	1	1
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	—	—	—

* Software can write a "0" to clear the flags, but cannot write a "1" in these bits.

The SSR is an 8-bit register that indicates transmit and receive status. It is initialized to H'87 at a reset and in the standby modes.

Bit 7 – Transmit Data Register Empty (TDRE): This bit indicates when the TDR contents have been transferred to the TSR and the next character can safely be written in the TDR.

Bit 7

TDRE	Description
0	To clear TDRE, the CPU must read TDRE after it has been set to "1," then write a "0" in this bit.
1	This bit is set to 1 at the following times: (Initial value) (1) When TDR contents are transferred to the TSR. (2) When the TE bit in the SCR is cleared to "0."

Bit 6 – Receive Data Register Full (RDRF): This bit indicates when one character has been received and transferred to the RDR.

Bit 6

RDRF	Description
0	To clear RDRF, the CPU must read RDRF after it has been set to "1," then write a "0" in this bit. (Initial value)
1	This bit is set to 1 when one character is received without error and transferred from the RSR to the RDR.

Bit 5 – Overrun Error (ORER): This bit indicates an overrun error during reception.

Bit 5 ORER	Description	
0	To clear ORER, the CPU must read ORER after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 if reception of the next character ends while the receive data register is still full (RDRF = “1”).	

Bit 4 – Framing Error (FER): This bit indicates a framing error during data reception in the asynchronous mode. It has no meaning in the synchronous mode.

Bit 4 FER	Description	
0	To clear FER, the CPU must read FER after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 if a framing error occurs (stop bit = “0”).	

Bit 3 – Parity Error (PER): This bit indicates a parity error during data reception in the asynchronous mode, when a communication format with parity bits is used.

This bit has no meaning in the synchronous mode, or when a communication format without parity bits is used.

Bit 3 PER	Description	
0	To clear PER, the CPU must read PER after it has been set to "1," then write a “0” in this bit.	(Initial value)
1	This bit is set to 1 when a parity error occurs (the parity of the received data does not match the parity selected by the O/$\overline{E}$ bit in the SMR).	

Bits 2 to 0 – Reserved: These bits cannot be modified and are always read as “1.”

9.2.8 Bit Rate Register (BRR) – H'FFD9

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

The BRR is an 8-bit register that, together with the CKS1 and CKS0 bits in the SMR, determines the baud rate output by the baud rate generator.

The BRR is initialized to H'FF (the slowest rate) at a reset and in the standby modes.

Tables 9-3 and 9-4 show examples of BRR (N) and CKS (n) settings for commonly used bit rates.

Table 9-3. Examples of BRR Settings in Asynchronous Mode (1)

	XTAL Frequency (MHz)											
	2			2.4576			4			4.194304		
Bit rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	70	+0.03	1	86	+0.31	1	141	+0.03	1	148	–0.04
150	0	207	+0.16	0	255	0	1	103	+0.16	1	108	+0.21
300	0	103	+0.16	0	127	0	0	207	+0.16	0	217	+0.21
600	0	51	+0.16	0	63	0	0	103	+0.16	0	108	+0.21
1200	0	25	+0.16	0	31	0	0	51	+0.16	0	54	–0.70
2400	0	12	+0.16	0	15	0	0	25	+0.16	0	26	+1.14
4800	—	—	—	0	7	0	0	12	+0.16	0	13	–2.48
9600	—	—	—	0	3	0	—	—	—	—	—	—
19200	—	—	—	0	1	0	—	—	—	—	—	—
31250	—	—	—	—	—	—	0	1	0	—	—	—
38400	—	—	—	0	0	0	—	—	—	—	—	—

Table 9-3. Examples of BRR Settings in Asynchronous Mode (2)

	XTAL Frequency (MHz)											
	4.9152			6			7.3728			8		
Bit rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	174	–0.26	2	52	+0.50	2	64	+0.70	2	70	+0.03
150	1	127	0	1	155	+0.16	1	191	0	1	207	+0.16
300	0	255	0	1	77	+0.16	1	95	0	1	103	+0.16
600	0	127	0	0	155	+0.16	0	191	0	0	207	+0.16
1200	0	63	0	0	77	+0.16	0	95	0	0	103	+0.16
2400	0	31	0	0	38	+0.16	0	47	0	0	51	+0.16
4800	0	15	0	0	19	–2.34	0	23	0	0	25	+0.16
9600	0	7	0	—	—	—	0	11	0	0	12	+0.16
19200	0	3	0	—	—	—	0	5	0	—	—	—
31250	—	—	—	0	2	0	—	—	—	0	3	0
38400	0	1	0	—	—	—	0	2	0	—	—	—

Table 9-3. Examples of BRR Settings in Asynchronous Mode (3)

	XTAL Frequency (MHz)											
	9.8304			10			12			12.288		
Bit rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	86	+0.31	2	88	–0.25	2	106	–0.44	2	108	+0.08
150	1	255	0	2	64	+0.16	2	77	0	2	79	0
300	1	127	0	1	129	+0.16	1	155	0	1	159	0
600	0	255	0	1	64	+0.16	1	77	0	1	79	0
1200	0	127	0	0	129	+0.16	0	155	+0.16	0	159	0
2400	0	63	0	0	64	+0.16	0	77	+0.16	0	79	0
4800	0	31	0	0	32	–1.36	0	38	+0.16	0	39	0
9600	0	15	0	0	15	+1.73	0	19	–2.34	0	19	0
19200	0	7	0	0	7	+1.73	—	—	—	0	4	0
31250	0	4	–1.70	0	4	0	0	5	0	0	5	+2.40
38400	0	3	0	0	3	+1.73	—	—	—	—	—	—

Table 9-3. Examples of BRR Settings in Asynchronous Mode (4)

	XTAL Frequency (MHz)											
	14.7456			16			19.6608			20		
Bit rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	130	–0.07	2	141	+0.03	2	174	–0.26	3	43	+0.88
150	2	95	0	2	103	+0.16	2	127	0	2	129	+0.16
300	1	191	0	1	207	+0.16	1	255	0	2	64	+0.16
600	1	95	0	1	103	+0.16	1	127	0	1	129	+0.16
1200	0	191	0	0	207	+0.16	0	255	0	1	64	+0.16
2400	0	95	0	0	103	+0.16	0	127	0	0	129	+0.16
4800	0	47	0	0	51	+0.16	0	63	0	0	64	+0.16
9600	0	23	0	0	25	+0.16	0	31	0	0	32	–1.36
19200	0	11	0	0	12	+0.16	0	15	0	0	15	+1.73
31250	—	—	—	0	7	0	0	9	–1.70	0	9	0
38400	0	5	0	—	—	—	0	7	0	0	7	+1.73

Note: If possible, the error should be within 1%.

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

N: BRR value ($0 \le N \le 255$)

OSC: Crystal oscillator frequency in MHz

B: Bit rate (bits/second)

n: Internal clock source (0, 1, 2, or 3)

The meaning of n is given by the table below:

n	CKS1	CKS0	Clock
0	0	0	Ø
1	0	1	Ø/4
2	1	0	Ø/16
3	1	1	Ø/64

Table 9-4. Examples of BRR Settings in Synchronous Mode

	XTAL Frequency (MHz)											
Bit	**2**		**4**		**8**		**10**		**16**		**20**	
rate	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**
100	—	—	—	—	—	—	—	—	—	—	—	—
250	1	249	2	124	2	249	—	—	3	124	—	—
500	1	124	1	249	2	124	—	—	2	249	—	—
1k	0	249	1	124	1	249	—	—	2	124	—	—
2.5k	0	99	0	199	1	99	1	124	1	199	1	249
5k	0	49	0	99	0	199	0	249	1	99	1	124
10k	0	24	0	49	0	99	0	124	0	199	0	249
25k	0	9	0	19	0	39	0	49	0	79	0	99
50k	0	4	0	9	0	19	0	24	0	39	0	49
100k	—	—	0	4	0	9	—	—	0	19	0	24
250k	0	0*	0	1	0	3	0	4	0	7	0	9
500k			0	0*	0	1	—	—	0	3	0	4
1M					0	0*	—	—	0	1	—	—
2.5M											0	0*

Notes:

Blank: No setting is available.

—: A setting is available, but the bit rate is inaccurate.

*: Continuous transfer is not possible.

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

N: BRR value ($0 \le N \le 255$)

OSC: Crystal oscillator frequency in MHz

B: Bit rate (bits per second)

n: Internal clock source (0, 1, 2, or 3)

The meaning of n is given by the table below:

n	**CKS1**	**CKS0**	**Clock**
0	0	0	Ø
1	0	1	Ø/4
2	1	0	Ø/16
3	1	1	Ø/64

9.3 Operation

9.3.1 Overview

The SCI supports serial data transfer in both asynchronous and synchronous modes.

The communication format depends on settings in the SMR as indicated in Table 9-5. The clock source and usage of the ASCK and CSCK pins depend on settings in the SMR and SCR as indicated in Table 9-6.

Table 9-5. Communication Formats Used by SCI

	SMR						
C/$\overline{A}$	**CHR**	**PE**	**STOP**	**Mode**	**Format**	**Parity**	**Stop bit length**
0	0	0	0	Asynchronous	8-Bit data	None	1
			1				2
		1	0			Yes	1
			1				2
	1	0	0		7-Bit data	None	1
			1				2
		1	0			Yes	1
			1				2
1	—	—	—	Synchronous	8-Bit data	—	—

Table 9-6. SCI Clock Source Selection

SMR	SCR		Clock		
C/$\overline{A}$	**CKE1**	**CKE0**	**source**	**ASCK pin**	**CSCK pin**
0 (Async mode)	0	0	Internal	Input/output port*	Input/output port*
		1		Serial clock output at bit rate	Input/output port*
	1	0	External	Serial clock input at 16 × bit rate	Input/output port*
		1			
1 (Sync mode)	0	0	Internal	Input/output port*	Serial clock output
		1			
	1	0	External	Input/output port*	Serial clock input
		1			

* Not used by the SCI.

Transmitting and receiving operations in the two modes are described next.

9.3.2 Asynchronous Mode

In asynchronous mode, each character is individually synchronized by framing it with a start bit and stop bit.

Full duplex data transfer is possible because the SCI has independent transmit and receive sections. Double buffering in both sections enables the SCI to be programmed for continuous data transfer.

Figure 9-2 shows the general format of one character sent or received in the asynchronous mode. The communication channel is normally held in the mark state (High). Character transmission or reception starts with a transition to the space state (Low).

The first bit transmitted or received is the start bit (Low). It is followed by the data bits, in which the least significant bit (LSB) comes first. The data bits are followed by the parity bit, if present, then the stop bit or bits (High) confirming the end of the frame.

In receiving, the SCI synchronizes on the falling edge of the start bit, and samples each bit at the center of the bit (at the 8th cycle of the internal serial clock, which runs at 16 times the bit rate).

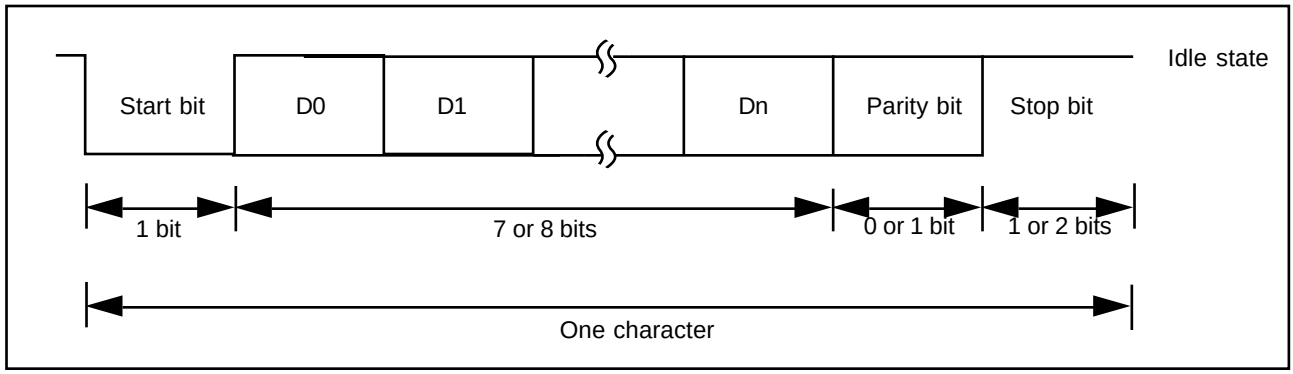


Figure 9-2. Data Format in Asynchronous Mode

(1) Data Format: Table 9-7 lists the data formats that can be sent and received in asynchronous mode. Eight formats can be selected by bits in the SMR.

Table 9-7. Data Formats in Asynchronous Mode

SMR bits			
CHR	**PE**	**STOP**	**Data format**
0	0	0	START | 8-Bit data | STOP
0	0	1	START | 8-Bit data | STOP | STOP
0	1	0	START | 8-Bit data | P | STOP
0	1	1	START | 8-Bit data | P | STOP | STOP
1	0	0	START | 7-Bit data | STOP
1	0	1	START | 7-Bit data | STOP | STOP
1	1	0	START | 7-Bit data | P | STOP
1	1	1	START | 7-Bit data | P | STOP | STOP

Note

START: Start bit

STOP: Stop bit

P: Parity bit

(2) Clock: In the asynchronous mode it is possible to select either an internal clock created by the on-chip baud rate generator, or an external clock input at the ASCK pin. Refer to Table 9-6.

If an external clock is input at the ASCK pin, its frequency should be 16 times the desired baud rate.

If the internal clock provided by the on-chip baud rate generator is selected and the ASCK pin is used for clock output, the output clock frequency is equal to the baud rate, and the clock pulse rises at the center of the transmit data bits. Figure 9-3 shows the phase relationship between the output clock and transmit data.

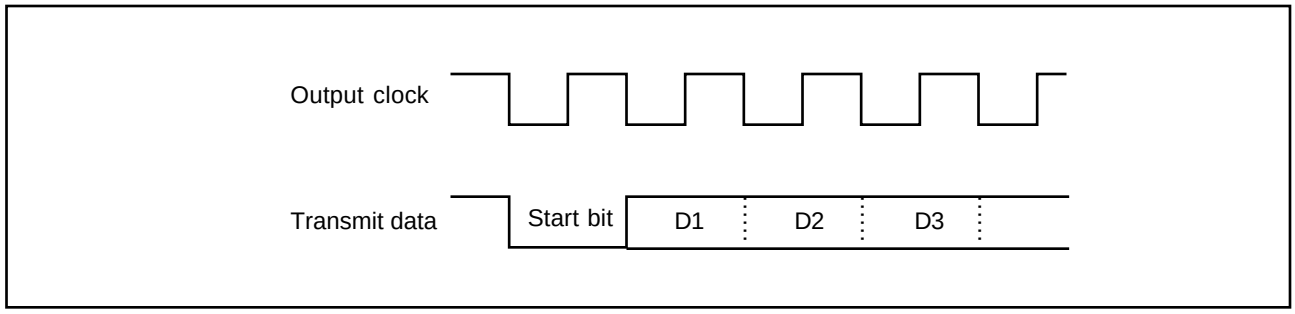


Figure 9-3. Phase Relationship Between Clock Output and Transmit Data

(3) Data Transmission and Reception

- **SCI Initialization:** Before data can be transmitted or received, the SCI must be initialized by software. To initialize the SCI, software must clear the TE and RE bits to "0," then execute the following procedure.

(1) Write the value corresponding to the desired bit rate in the BRR. (This step is not necessary if an external clock is used.)

(2) Select the desired communication parameters in the SCR. Leave bit 0 (CKE0) cleared to zero.

(3) Select clocked synchronous mode in the SMR.

(4) Set the TE and/or RE bit in the SCR to "1."

The TE and RE bits must both be cleared to "0" whenever the operating mode or data format is changed.

After changing the operating mode or data format, before setting the TE and RE bits to "1" software must wait for at least the transfer time for 1 bit at the selected baud rate, to make sure the SCI is initialized. If an external clock is used, the clock must not be stopped.
When clearing the TDRE bit during data transmission, to assure transfer of the correct data, do not clear the TDRE bit until after writing data in the TDR. Similarly, in receiving data, do not clear the RDRF bit until after reading data from the RDR.

- **Data Transmission:** The procedure for transmitting data is as follows.

(1) Set up the desired transmitting conditions in the SMR, SCR, and BRR.

(2) Set the TE bit in the SCR to "1."
The ATxD pin will automatically be switched to output and one frame* of all 1's will be transmitted, after which the SCI is ready to transmit data.

(3) Check that the TDRE bit is set to "1," then write the first byte of transmit data in the TDR. Next clear the TDRE bit to "0."

(4) The first byte of transmit data is transferred from the TDR to the TSR and sent in the designated format as follows.
   i) Start bit (one “0” bit).
   ii) Transmit data (seven or eight bits, starting from bit 0)
   iii) Parity bit (odd or even parity bit, or no parity bit)
   iv) Stop bit (one or two consecutive “1” bits)

(5) Transfer of the transmit data from the TDR to the TSR makes the TDR empty, so the TDRE bit is set to “1.”
   If the TIE bit is set to “1,” a transmit-end interrupt (TxI) is requested.
   When the transmit function is enabled but the TDR is empty (TDRE = “1”), the output at the ATxD pin is held at “1” until the TDRE bit is cleared to “0.”

* A frame is the data for one character, including the start bit and stop bit(s).

• **Data Reception:** The procedure for receiving data is as follows.

(1) Set up the desired receiving conditions in the SMR, SCR, and BRR.

(2) Set the RE bit in the SCR to “1.”
   The ARxD pin is automatically be switched to input and the SCI is ready to receive data.

(3) The SCI synchronizes with the incoming data by detecting the start bit, and places the received bits in the RSR. At the end of the data, the SCI checks that the stop bit is “1.”

(4) When a complete frame has been received, the SCI transfers the received data from the RSR to the RDR so that it can be read. If the character length is 7 bits, the most significant bit of the RDR is cleared to “0.”
   At the same time, the SCI sets the RDRF bit in the SSR to “1.” If the RIE bit is set to “1,” a receive-end interrupt (RxI) is requested.

(5) The RDRF bit is cleared to “0” when software reads the SSR, then writes a “0” in the RDRF bit. The RDR is then ready to receive the next character from the RSR.

When a frame is not received correctly, a receive error occurs. There are three types of receive errors, listed in Table 9-8.

If a receive error occurs, the RDRF bit in the SSR is not set to “1.” (For an overrun error, RDRF is already set to "1.") The corresponding error flag is set to “1” instead. If the RIE bit in the SCR is set to “1,” a receive-error interrupt (ERI) is requested.

When a framing or parity error occurs, the RSR contents are transferred to the RDR. If an overrun error occurs, however, the RSR contents are not transferred to the RDR.

If multiple receive errors occur simultaneously, all the corresponding error flags are set to “1.”

To clear a receive-error flag (ORER, FER, or PER), software must read the SSR and then write a “0” in the flag bit.

Table 9-8. Receive Errors

Name	Abbreviation	Description
Overrun error	ORER	Reception of the next frame ends while the RDRF bit is still set to “1.” The RSR contents are not transferred to the RDR.
Framing error	FER	A stop bit is “0.” The RSR contents are transferred to the RDR.
Parity error	PER	The parity of a frame does not match the value selected by the O/$\overline{E}$ bit in the SMR. The RSR contents are transferred to the RDR.

9.3.3 Synchronous Mode

The synchronous mode is suited for high-speed, continuous data transfer. Each bit of data is synchronized with a serial clock pulse at the CSCK pin.

Continuous data transfer is enabled by the double buffering employed in both the transmit and receive sections of the SCI. Full duplex communication is possible because the transmit and receive sections are independent.

(1) Data Format: Figure 9-4 shows the communication format used in the synchronous mode. The data length is 8 bits for both the transmit and receive directions. The least significant bit (LSB) is sent and received first. Each bit of transmit data is output from the falling edge of the serial clock pulse to the next falling edge. Received bits are latched on the rising edge of the serial clock pulse.

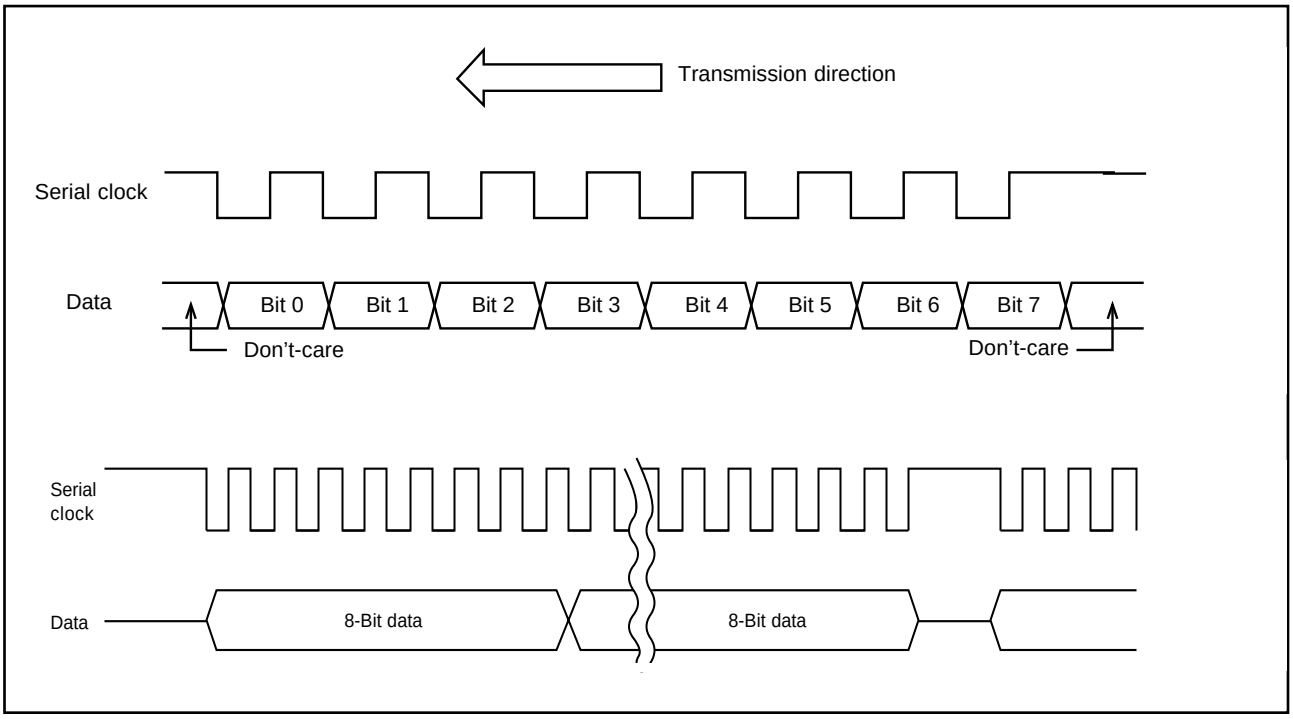


Figure 9-4. Data Format in Synchronous Mode

(2) Clock: Either the internal serial clock created by the on-chip baud rate generator or an external clock input at the CSCK pin can be selected in the synchronous mode. See Table 9-6 for details.

(3) Data Transmission and Reception

• **SCI Initialization:** Before data can be transmitted or received, the SCI must be initialized by software. To initialize the SCI, software must clear the TE and RE bits to "0" to disable both the transmit and receive functions, then execute the following procedure.

(1) Write the value corresponding to the desired bit rate in the BRR. (This step is not necessary if an external clock is used.)

(2) Select the clock and enable desired interrupts in the SCR. Leave bit 0 (CKE0) cleared to "0."

(3) Select the synchronous mode in the SMR.

(4) Set the TE and/or RE bit in the SCR to "1."

The TE and RE bits must both be cleared to "0" whenever the operating mode or data format is changed. After changing the operating mode or data format, before setting the TE and RE bits to "1" software must wait for at least 1 bit transfer time at the selected communication speed, to make sure the SCI is initialized.

When clearing the TDRE bit during data transmission, to assure correct data transfer, do not clear the TDRE bit until after writing data in the TDR. Similarly, in receiving data, do not clear the RDRF bit until after reading data from the RDR.

- **Data Transmission:** The procedure for transmitting data is as follows.

(1) Set up the desired transmitting conditions in the SMR, BRR, and SCR.

(2) Set the TE bit in the SCR to "1."
The CTxD pin will automatically be switched to output, after which the SCI is ready to transmit data.

(3) Check that the TDRE bit is set to "1," then write the first byte of transmit data in the TDR. Next clear the TDRE bit to "0."

(4) The first byte of transmit data is transferred from the TDR to the TSR and sent, each bit synchronized with a clock pulse. Bit 0 is sent first.
Transfer of the transmit data from the TDR to the TSR makes the TDR empty, so the TDRE bit is set to "1." If the TIE bit is set to "1," a transmit-end interrupt (TxI) is requested.

The TDR and TSR function as a double buffer. Continuous data transmission can be achieved by writing the next transmit data in the TDR and clearing the TDRE bit to "0" while the SCI is transmitting the current data from the TSR.

If an internal clock source is selected, after transferring the transmit data from the TDR to the TSR, while transmitting the data from the TSR the SCI also outputs a serial clock signal at the CSCK pin. When all data bits in the TSR have been transmitted, if the TDR is empty (TDRE = "1"), serial clock output is suspended until the next data byte is written in the TDR and the TDRE bit is cleared to "0." During this interval the CTxD pin continues to output the value of the last bit of the previous data.

If the external clock source is selected, data transmission is synchronized with the clock signal input at the CSCK pin. When all data bits in the TSR have been transmitted, if the TDR is empty (TDRE = "1") but external clock pulses continue to arrive, the CTxD pin outputs the value of last bit of the previous data.

- **Data Reception:** The procedure for receiving data is as follows.

(1) Set up the desired receiving conditions in the SMR, BRR, and SCR.

(2) Set the RE bit in the SCR to “1.”
The CRxD pin is automatically be switched to input and the SCI is ready to receive data.

(3) Incoming data bits are latched in the RSR on eight clock pulses.
When 8 bits of data have been received, the SCI sets the RDRF bit in the SSR to “1.” If the RIE bit is set to “1,” a receive-end interrupt (RxI) is requested.

(4) The SCI transfers the received data byte from the RSR to the RDR so that it can be read.
The RDRF bit is cleared when software reads the RDRF bit in the SSR, then writes a “0” in the RDRF bit.

The RDR and RSR function as a double buffer. Data can be received continuously by reading each byte of data from the RDR and clearing the RDRF bit to “0” before the last bit of the next byte is received.

In general, an external clock source should be used for receiving data.

If an internal clock source is selected, the SCI starts receiving data as soon as the RE bit is set to “1.” The serial clock is also output at the CSCK pin. The SCI continues receiving until the RE bit is cleared to “0.”

If the last bit of the next data byte is received while the RDRF bit is still set to “1,” an overrun error occurs and the ORER bit is set to “1.” If the RIE bit is set to “1,” a receive-error interrupt (ERI) is requested. The data received in the RSR are not transferred to the RDR when an overrun error occurs.

After an overrun error, reception of the next data is enabled when the ORER bit is cleared to “0.”

• **Simultaneous Transmit and Receive:** The procedure for transmitting and receiving simultaneously is as follows:

(1) Set up the desired communication conditions in the SMR, BRR, and SCR.

(2) Set the TE and RE bits in the SCR to “1.”
The CTxD and CRxD pins are automatically switched to output and input, respectively, and the SCI is ready to transmit and receive data.

(3) Data transmitting and receiving start when the TDRE bit in the SSR is cleared to “0.”

(4) Data are sent and received in synchronization with eight clock pulses.

(5) First, the transmit data are transferred from the TDR to the TSR. This makes the TDR empty, so the TDRE bit is set to “1.” If the TIE bit is set to “1,” a transmit-end interrupt (TxI) is requested.
If continuous data transmission is desired, software must read the TDRE bit in the SSR, write the next transmit data in the TDR, then clear the TDRE bit to “0.”
If the TDRE bit is not cleared to “0” by the time the SCI finishes sending the current byte from the TSR, the CTxD pin continues to output the value of last bit of the previous data.

(6) In the receiving section, when 8 bits of data have been received they are transferred from the RSR to the RDR and the RDRF bit in the SSR is set to “1.” If the RIE bit is set to “1,” a receive-end interrupt (RxI) is requested.

(7) To clear the RDRF bit software should read the RDRF bit in the SSR, read the data in the RDR, then write a “0” in the RDRF bit.
For continuous data reception, software should read the RDRF bit in the SSR, read the data in the RDR, then clear the RDRF bit to “0.”

If the last bit of the next byte is received while the RDRF bit is still set to “1,” an overrun error occurs. The error is handled as described under “Data Reception” above. The overrun error does not affect the transmit section of the SCI, which continues to transmit normally.

9.4 Interrupts

The SCI can request three types of interrupts: transmit-end (TxI), receive-end (RxI), and receive-error (ERI). Interrupt requests are enabled or disabled by the TIE and RIE bits in the SCR. Independent signals are sent to the interrupt controller for each type of interrupt. The transmit-end and receive-end interrupt request signals are obtained from the TDRE and RDRF flags. The receive-error interrupt request signal is the logical OR of the three error flags: overrun error (ORER), framing error (FER), and parity error (PER). Table 9-9 lists information about these interrupts.

Table 9-9. SCI Interrupts

Interrupt	Description	Priority
ERI	Receive-error interrupt, requested when ORER, FER, or PER is set. RIE must also be set.	High
RxI	Receive-end interrupt, requested when RDRF and RIE are set.	
TxI	Transmit-end interrupt, requested when TDRE and TIE are set.	Low

Figure 9-5 shows the timing of the RxI interrupt signal. The timing of TxI and ERI is similar.

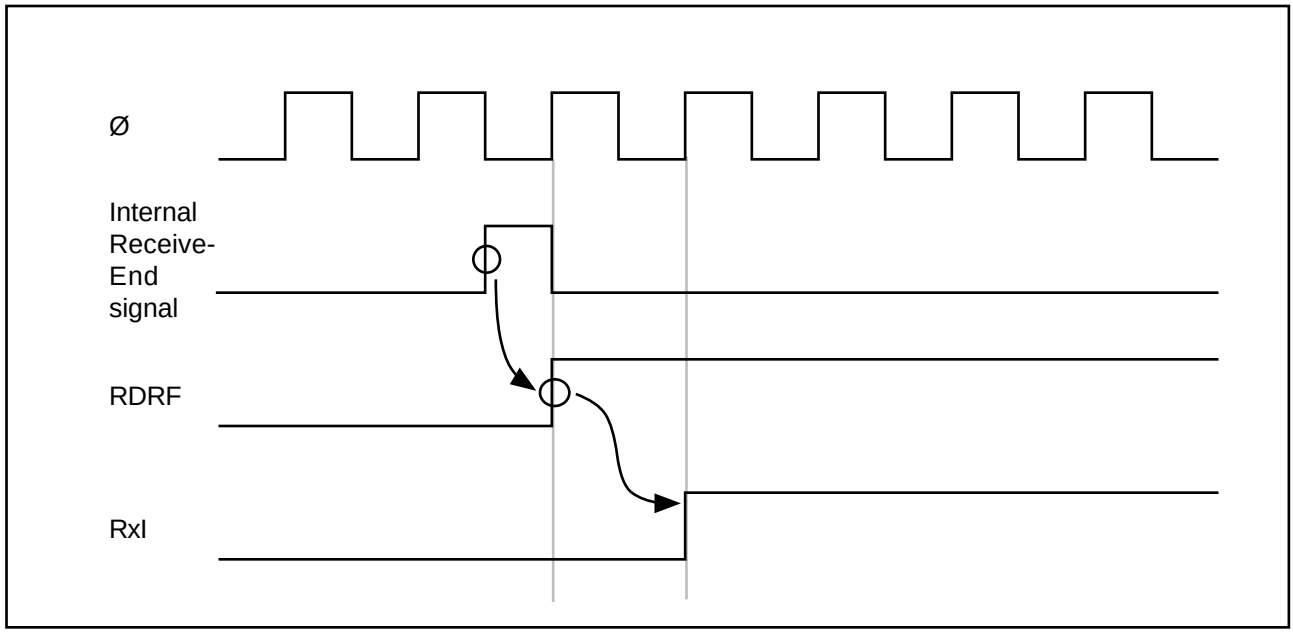


Figure 9-5. Timing of Interrupt Signal

9.5 Application Notes

Application programmers should note the following features of the SCI.

(1) TDR Write: The TDRE bit in the SSR is simply a flag that indicates that the TDR contents have been transferred to the TSR. The TDR contents can be rewritten regardless of the TDRE value. If a new byte is written in the TDR while the TDRE bit is “0,” before the old TDR contents have been moved into the TSR, the old byte will be lost. Normally, software should check that the TDRE bit is set to “1” before writing to the TDR.

(2) Multiple Receive Errors: Table 9-10 lists the values of flag bits in the SSR when multiple receive errors occur, and indicates whether the RSR contents are transferred to the RDR.

Table 9-10. SSR Bit States and Data Transfer When Multiple Receive Errors Occur

	SSR Bits				
Receive error	**RDRF**	**ORER**	**FER**	**PER**	**RSR → RDR*2**
Overrun error	1*1	1	0	0	No
Framing error	0	0	1	0	Yes
Parity error	0	0	0	1	Yes
Overrun + framing errors	1*1	1	1	0	No
Overrun + parity errors	1*1	1	0	1	No
Framing + parity errors	0	0	1	1	Yes
Overrun + framing + parity errors	1*1	1	1	1	No

*1 Set to “1” before the overrun error occurs.

*2 Yes: The RSR contents are transferred to the RDR.

No: The RSR contents are not transferred to the RDR.

(3) Line Break Detection: When the ARxD pin receives a continuous stream of 0’s in the asynchronous mode (line-break state), a framing error occurs because the SCI detects a “0” stop bit. The value H’00 is transferred from the RSR to the RDR. Software can detect the line-break state as a framing error accompanied by H’00 data in the RDR.

The SCI continues to receive data, so if the FER bit is cleared to “0” another framing error will occur.

(4) Sampling Timing and Receive Margin in Asynchronous Mode: The serial clock used by the SCI in asynchronous mode runs at 16 times the baud rate. The falling edge of the start bit is detected by sampling the ARxD input on the falling edge of this clock. After the start bit is detected, each bit of receive data in the frame (including the start bit, parity bit, and stop bit or bits) is sampled on the rising edge of the serial clock pulse at the center of the bit. See Figure 9-6.

It follows that the receive margin can be calculated as in equation (1).

When the absolute frequency deviation of the clock signal is 0 and the clock duty factor is 0.5, data can theoretically be received with distortion up to the margin given by equation (2). This is a theoretical limit, however. In practice, system designers should allow a margin of 20% to 30%.

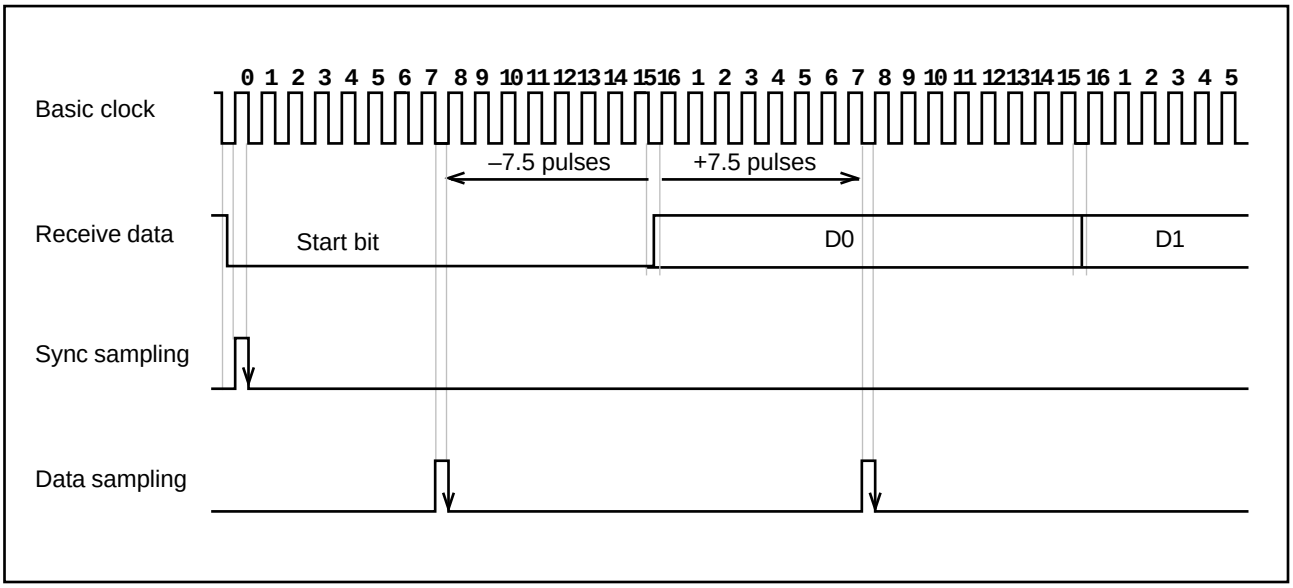


Figure 9-6. Sampling Timing (Asynchronous Mode)

$$M = \{(0.5 - 1/2N) - (D - 0.5)/N - (L - 0.5)F\} \times 100\ [\%] \quad (1)$$

M: Receive margin
N: Ratio of basic clock to baud rate (N=16)
D: Duty factor of clock—ratio of High pulse width to Low width (0.5 to 1.0)
L: Frame length (9 to 12)
F: Absolute clock frequency deviation

When D = 0.5 and F= 0

$$M = (0.5 - 1/2 \times 16) \times 100\ [\%] = 46.875\% \quad (2)$$

Section 10. A/D Converter

10.1 Overview

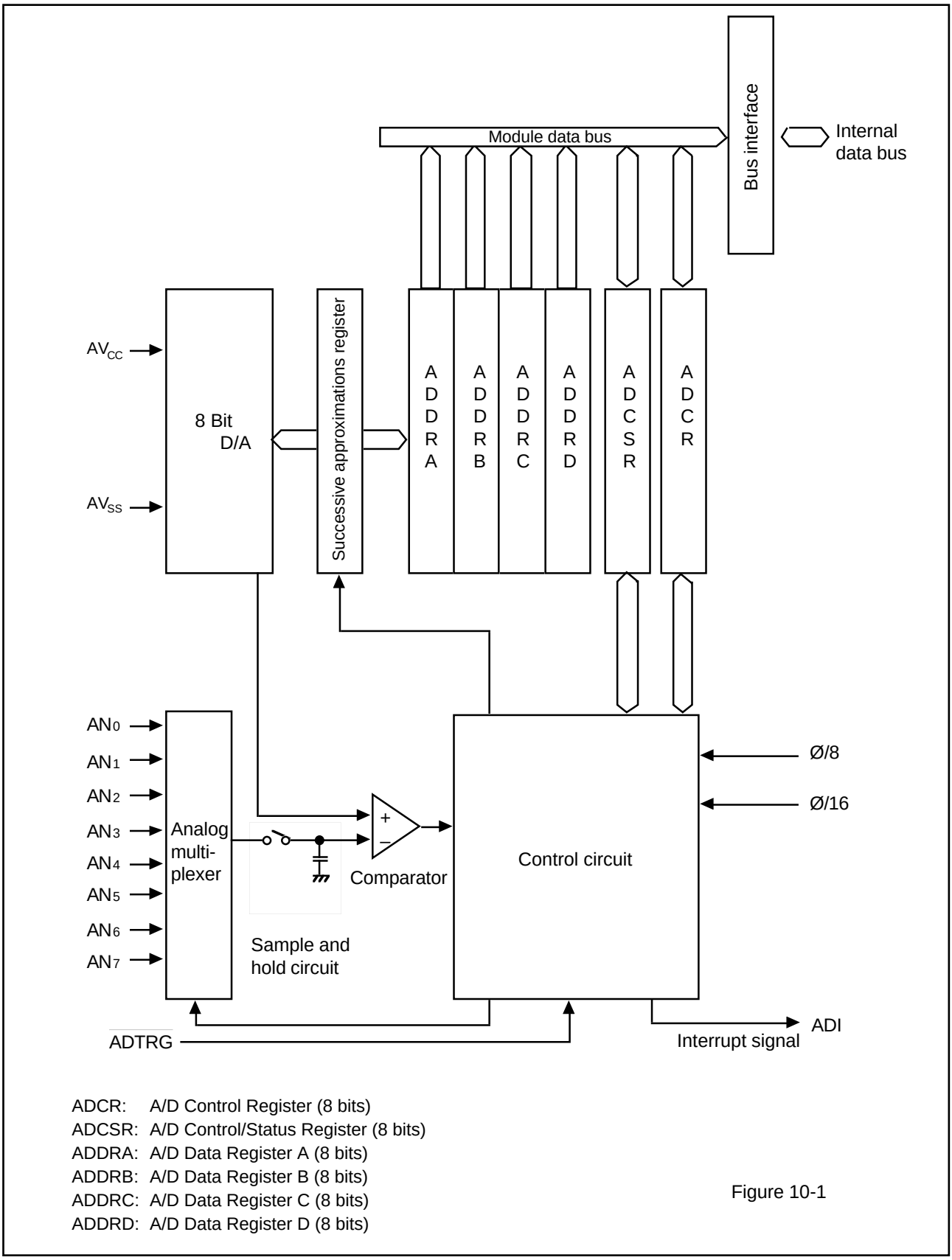
The H8/330 chip includes an analog-to-digital converter module with eight input channels. A/D conversion is performed by the successive approximations method with 8-bit resolution.

10.1.1 Features

The features of the on-chip A/D module are:

- Eight analog input channels
- 8-bit resolution
- Rapid conversion
  Conversion time is 12.2µs per channel (minimum) with a 10MHz system clock
- External triggering can be selected
- Single and scan modes
  – Single mode: A/D conversion is performed once.
  – Scan mode: A/D conversion is performed in a repeated cycle on one to four channels.
- Four 8-bit data registers
  These registers store A/D conversion results for up to four channels.
- A CPU interrupt (ADI) can be requested at the completion of each A/D conversion cycle.

10.1.2 Block Diagram



ADCR: A/D Control Register (8 bits)
ADCSR: A/D Control/Status Register (8 bits)
ADDRA: A/D Data Register A (8 bits)
ADDRB: A/D Data Register B (8 bits)
ADDRC: A/D Data Register C (8 bits)
ADDRD: A/D Data Register D (8 bits)

Figure 10-1

Figure 10-1. Block Diagram of A/D Converter

10.1.3 Input Pins

Table 10-1 lists the input pins used by the A/D converter module.

The eight analog input pins are divided into two groups, consisting of analog inputs 0 to 3 (AN_0 to AN_3) and analog inputs 4 to 7 (AN_4 to AN_7), respectively.

Table 10-1. A/D Input Pins

Name	Abbreviation	I/O	Function
Analog supply voltage	AV_{CC}	Input	Power supply and reference voltage for the analog circuits.
Analog ground	AV_{SS}	Input	Ground and reference voltage for the analog circuits.
Analog input 0	AN_0	Input	Analog input pins, group 0
Analog input 1	AN_1	Input	
Analog input 2	AN_2	Input	
Analog input 3	AN_3	Input	
Analog input 4	AN_4	Input	Analog input pins, group 1
Analog input 5	AN_5	Input	
Analog input 6	AN_6	Input	
Analog input 7	AN_7	Input	
A/D external trigger	$\overline{ADTRG}$	Input	External trigger for starting A/D conversion

10.1.4 Register Configuration

Table 10-2 lists the registers of the A/D converter module.

Table 10-2. A/D Registers

Name	Abbreviation	R/W	Initial value	Address
A/D data register A	ADDRA	R	H’00	H’FFE0
A/D data register B	ADDRB	R	H’00	H’FFE2
A/D data register C	ADDRC	R	H’00	H’FFE4
A/D data register D	ADDRD	R	H’00	H’FFE6
A/D control/status register	ADCSR	R/(W)*	H’00	H’FFE8
A/D control register	ADCR	R/W	H’7F	H’FFEA

* Software can write a "0" to clear bit 7, but cannot write a "1" in this bit.

10.2 Register Descriptions

10.2.1 A/D Data Registers (ADDR) – H’FFE0 to H'FFE6

Bit	7	6	5	4	3	2	1	0
ADDRn								
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

(n = A to D)

The four A/D data registers (ADDRA to ADDRD) are 8-bit read-only registers that store the results of A/D conversion. Each data register is assigned to two analog input channels as indicated in Table 10-3.

The A/D data registers are always readable by the CPU.

The A/D data registers are initialized to H’00 at a reset and in the standby modes.

Table 10-3. Assignment of Data Registers to Analog Input Channels

Analog input channel		
Group 0	**Group 1**	**A/D data register**
AN_0	AN_4	ADDRA
AN_1	AN_5	ADDRB
AN_2	AN_6	ADDRC
AN_3	AN_7	ADDRD

10.2.2 A/D Control/Status Register (ADCSR) – H’FFE8

Bit	7	6	5	4	3	2	1	0
	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W	R/W

* Software can write a “0” in bit 7 to clear the flag, but cannot write a “1” in this bit.

The A/D control/status register (ADCSR) is an 8-bit readable/writable register that controls the operation of the A/D converter module.

The ADCSR is initialized to H’00 at a reset and in the standby modes.

Bit 7 – A/D End Flag (ADF): This status flag indicates the end of one cycle of A/D conversion.

Bit 7 ADF	Description	
0	To clear ADF, the CPU must read ADF after it has been set to "1," then write a "0" in this bit.	(Initial value)
1	This bit is set to 1 at the following times: (1) Single mode: when one A/D conversion is completed. (2) Scan mode: when inputs on all selected channels have been converted.	

Bit 6 – A/D Interrupt Enable (ADIE): This bit selects whether to request an A/D interrupt (ADI) when A/D conversion is completed.

Bit 6 ADIE	Description	
0	The A/D interrupt request (ADI) is disabled.	(Initial value)
1	The A/D interrupt request (ADI) is enabled.	

Bit 5 – A/D Start (ADST): The A/D converter operates while this bit is set to “1.” In the single mode, this bit is automatically cleared to “0” at the end of each A/D conversion.

Bit 5 ADST	Description	
0	A/D conversion is halted.	(Initial value)
1	(1) Single mode: One A/D conversion is performed. The ADST bit is automatically cleared to “0” at the end of the conversion. (2) Scan mode: A/D conversion starts and continues cyclically on the selected channels until the ADST bit is cleared to “0” by software (or a reset, or by entry to a standby mode).	

Bit 4 – Scan Mode (SCAN): This bit selects the scan mode or single mode of operation.
See Section 10.3, “Operation” for descriptions of these modes.
The mode should be changed only when the ADST bit is cleared to “0.”

Bit 4

SCAN	Description	
0	Single mode	(Initial value)
1	Scan mode	

Bit 3 – Clock Select (CKS): This bit controls the A/D conversion time.

The conversion time should be changed only when the ADST bit is cleared to “0.”

Bit 3

CKS	Description	
0	Conversion time = 242 states (max.)	(Initial value)
1	Conversion time = 122 states (max.)	

Bits 2 to 0 – Channel Select 2 to 0 (CH2 to CH0): These bits and the SCAN bit combine to select one or more analog input channels.

The channel selection should be changed only when the ADST bit is cleared to “0.”

Group select	Channel select		Selected channels	
CH2	CH1	CH0	Single mode	Scan mode
0	0	0	AN_0 (Initial value)	AN_0
	0	1	AN_1	AN_0, AN_1
	1	0	AN_2	AN_0 to AN_2
	1	1	AN_3	AN_0 to AN_3
1	0	0	AN_4	AN_4
	0	1	AN_5	AN_4, AN_5
	1	0	AN_6	AN_4 to AN_6
	1	1	AN_7	AN_4 to AN_7

10.2.3 A/D Control Register (ADCR) – H'FFEA

Bit	7	6	5	4	3	2	1	0
	TRGE	—	—	—	—	—	—	—
Initial value	0	1	1	1	1	1	1	1
Read/Write	R/W	—	—	—	—	—	—	—

The A/D control register (ADCR) is an 8-bit readable/writable register that enables or disables the A/D external trigger signal.

The ADCR is initialized to H'7F at a reset and in the standby modes.

Bit 7 – Trigger Enable (TRGE): This bit enables the $\overline{\text{ADTRG}}$ (A/D external trigger) signal to set the ADST bit and start A/D conversion.

Bit 7 TRGE	Description	
0	A/D external trigger is disabled. $\overline{\text{ADTRG}}$ does not set the ADST bit.	(Initial value)
1	A/D external trigger is enabled. $\overline{\text{ADTRG}}$ sets the ADST bit. (The ADST bit can also be set by software.)	

Bits 6 to 0 – Reserved: These bits cannot be modified and are always read as "1."

10.3 Operation

The A/D converter performs 8 successive approximations to obtain a result ranging from H'00 (corresponding to AV_{SS}) to H'FF (corresponding to AV_{CC}). Figure 10-2 shows the response of the A/D converter.

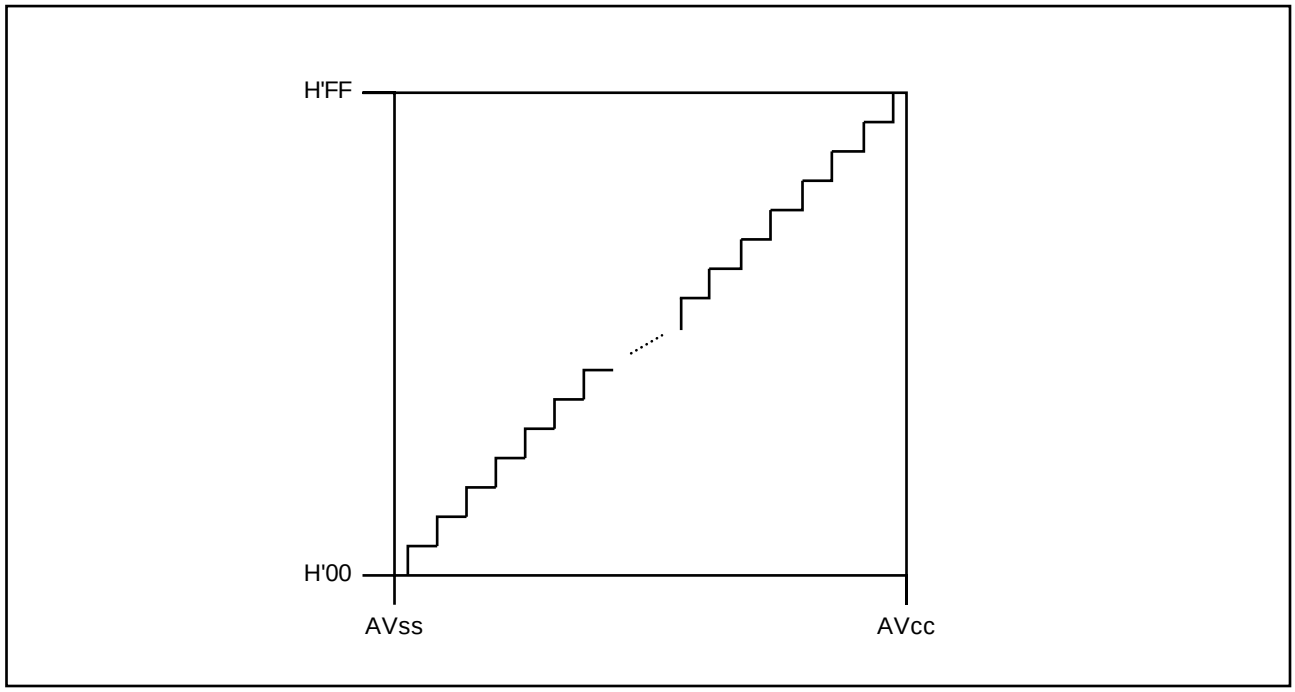


Figure 10-2. The Response of the A/D Converter

The A/D converter module can be programmed to operate in single mode or scan mode as explained below.

10.3.1 Single Mode (SCAN = 0)

The single mode is suitable for obtaining a single data value from a single channel. A/D conversion starts when the ADST bit is set to “1,” either by software or by a High-to-Low transition of the $\overline{\text{ADTRG}}$ signal (if enabled). During the conversion process the ADST bit remains set to “1.” When conversion is completed, the ADST bit is automatically cleared to “0.”

When the conversion is completed, the ADF bit is set to “1.” If the interrupt enable bit (ADIE) is also set to “1,” an A/D conversion end interrupt (ADI) is requested, so that the converted data can be processed by an interrupt-handling routine. The ADF bit is cleared when software reads the A/D control/status register (ADCSR), then writes a “0” in this bit.

Before selecting the single mode, clock, and analog input channel, software should clear the ADST bit to “0” to make sure the A/D converter is stopped. Changing the mode, clock, or channel selection while A/D conversion is in progress can lead to conversion errors. A/D conversion begins when the ADST bit is set to "1" again. The same instruction can be used to alter the mode and channel selection and set ADST to "1."

The following example explains the A/D conversion process in single mode when channel 1 (AN1) is selected and the external trigger is disabled. Figure 10-3 shows the corresponding timing chart.

(1) Software clears the ADST bit to “0,” then selects the single mode (SCAN = “0”) and channel 1 (CH2 to CH0 = “001”), enables the A/D interrupt request (ADIE = “1”), and sets the ADST bit to “1” to start A/D conversion.

Coding Example: (when using the slow clock, CKS = “0”)

```
BCLR #5, @H'FFE8        ;Clear ADST
MOV.B #H'7F, ROL
MOV.B ROL, @H'FFEA      ;Disable external trigger
MOV.B #H'61, ROL
MOV.B ROL, @H'FFE8      ;Select mode and channel and set ADST to "1"
```

Value set in ADCSR:

ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
0	1	1	0	0	0	0	1

(2) The A/D converter converts the voltage level at the the AN_1 input pin to a digital value. At the end of the conversion process the A/D converter transfers the result to register ADDRB, sets the ADF bit is set to “1,” clears the ADST bit to “0,” and halts.

(3) ADF = “1” and ADIE = “1,” so an A/D interrupt is requested.

(4) The user-coded A/D interrupt-handling routine is started.

(5) The interrupt-handling routine reads the ADCSR value, then writes a “0” in the ADF bit to clear this bit to “0.”

(6) The interrupt-handling routine reads and processes the A/D conversion result.

(7) The routine ends.

Steps (2) to (7) can now be repeated by setting the ADST bit to “1” again.

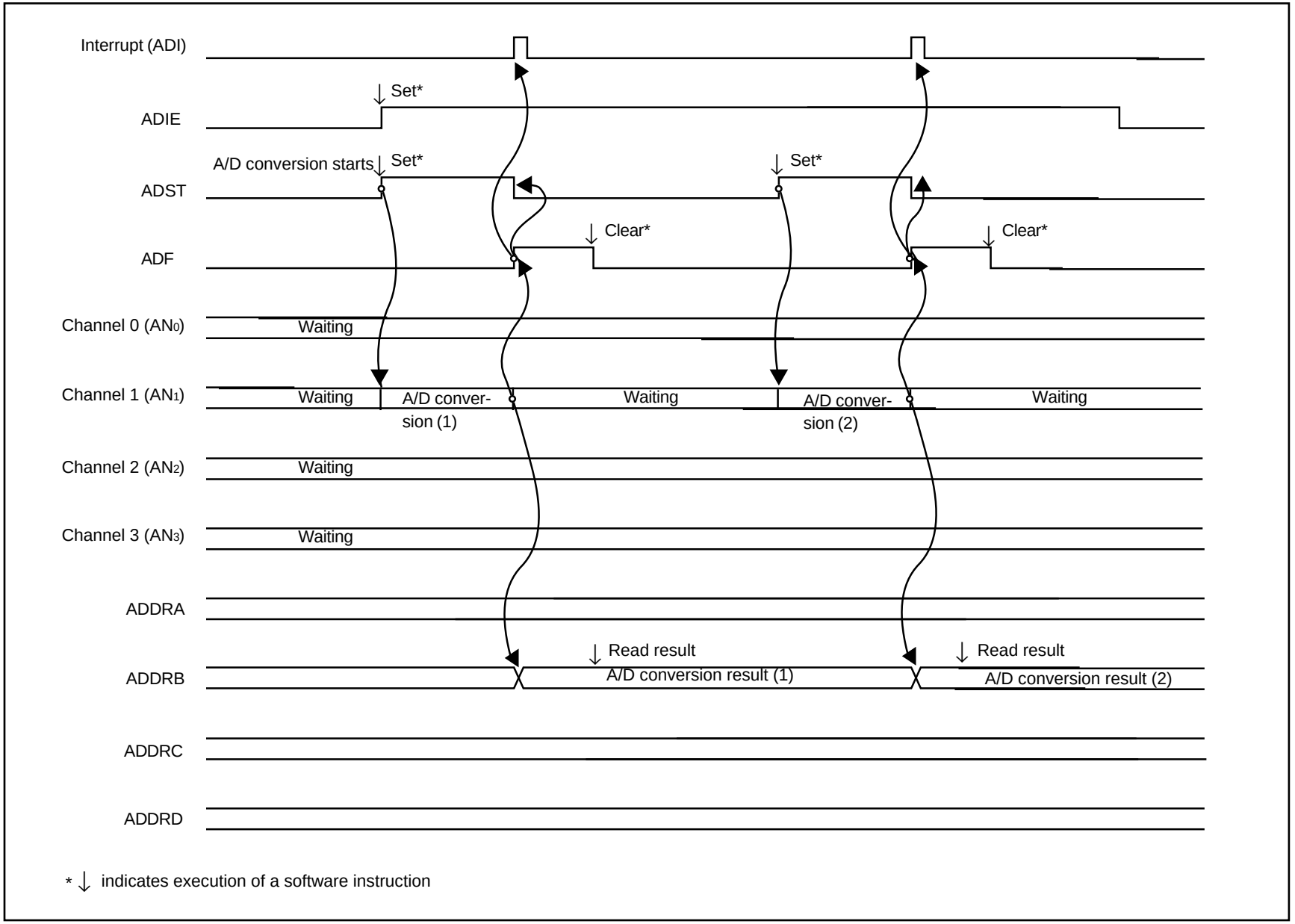


Figure 10-3. A/D Operation in Single Mode (When Channels 1 is Selected)

10.3.2 Scan Mode (SCAN = 1)

The scan mode can be used to monitor analog inputs on one or more channels. When the ADST bit is set to “1,” either by software or by a High-to-Low transition of the ADTRG signal (if enabled), A/D conversion starts from the first channel selected by the CH bits. When CH2 = “0” the first channel is AN_0. When CH2 = “1” the first channel is AN_4.

If the scan group includes more than one channel (i.e. if bit CH1 or CH0 is set), conversion of the next channel begins as soon as conversion of the first channel ends.

Conversion of the selected channels continues cyclically until the ADST bit is cleared to “0.” The conversion results are placed in the data registers corresponding to the selected channels. The A/D data registers are readable by the CPU.

Before selecting the scan mode, clock, and analog input channels, software should clear the ADST bit to “0” to make sure the A/D converter is stopped. Changing the mode, clock, or channel selection while A/D conversion is in progress can lead to conversion errors. A/D conversion begins when the ADST bit is set to "1" again. The same instruction can be used to alter the mode and channel selection and set ADST to "1."

The following example explains the A/D conversion process when three channels in group 0 are selected (AN_0, AN_1, and AN_2) and the external trigger is disabled. Figure 10-4 shows the corresponding timing chart.

(1) Software clears the ADST bit to “0,” then selects the scan mode (SCAN = “1”), scan group 0 (CH2 = “0”), and analog input channels AN0 to AN2 (CH1 and CH0 = “0”) and sets the ADST bit to “1” to start A/D conversion.

Coding Example: (with slow clock and ADI interrupt enabled)

```
BCLR #5, @H'FFE8        ;Clear ADST
MOV.B #H'7F, ROL
MOV.B ROL, @H'FFEA      ;Disable external trigger
MOV.B #H'72, ROL
MOV.B ROL, @H'FFE8      ;Select mode and channels and set ADST to "1"
```

Value set in ADCSR

ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
0	1	1	1	0	0	1	0

(2) The A/D converter converts the voltage level at the AN_0 input pin to a digital value, and transfers the result to register ADDRA.

(3) Next the A/D converter converts AN_1 and transfers the result to ADDRB. Then it converts AN_2 and transfers the result to ADDRC.

(4) After all selected channels (AN_0 to AN_2) have been converted, the AD converter sets the ADF bit to "1." If the ADIE bit is set to "1," an A/D interrupt (ADI) is requested. Then the A/D converter begins converting AN_0 again.

(5) Steps (2) to (4) are repeated cyclically as long as the ADST bit remains set to "1."

To stop the A/D converter, software must clear the ADST bit to "0."

Regardless of which channel is being converted when the ADST bit is cleared to "0," when the ADST bit is set to "1" again, conversion begins from the the first selected channel (AN_0 or AN_4).

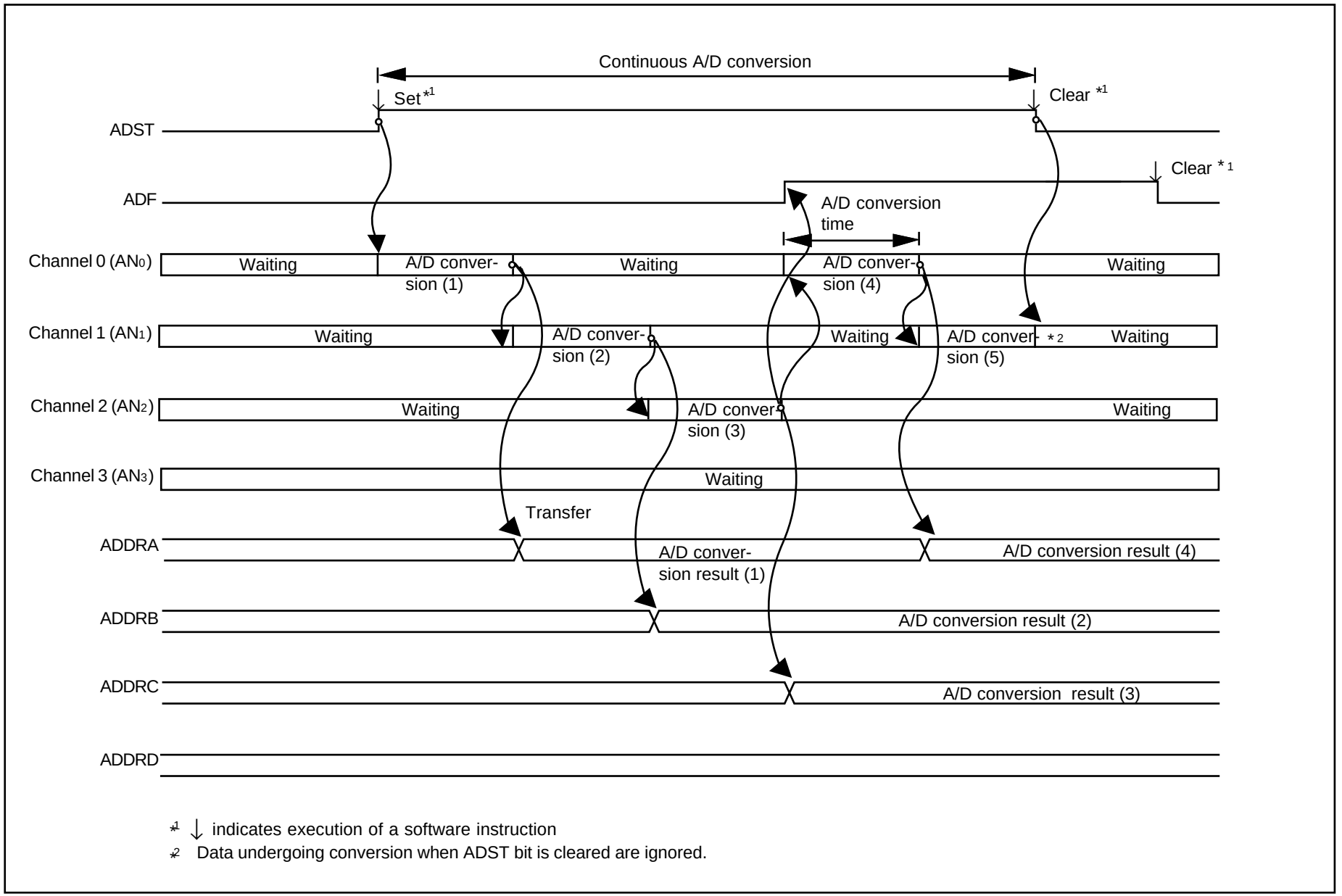


Figure 10-4. A/D Operation in Scan Mode (When Channels 0 to 2 are Selected)

Note: If the ADST bit is cleared to "0" when two or more channels are selected in the scan mode, incorrect values may be left in the A/D data registers.

For this reason, in the scan mode the A/D data registers should be read while the ADST bit is still set to "1."

Example: The following coding example sets up a four-channel A/D scan, and shows the first part of an ADI interrupt handler for reading the converted data. Note that the data are read before the ADST bit is cleared.

```
        MOV.B   #5B     , R0L
        MOV.B   R0L     , @ADCSR  ; Four-channel scan mode
        BSET    #5      , @ADCSR  ; Start conversion (set ADST)

                (Conversion of four channels)

ADI:    MOV.B   @ADDRA  , R1      ; Read ADDRA
        MOV.B   @ADDRB  , R2      ; Read ADDRB
        MOV.B   @ADDRC  , R3      ; Read ADDRC
        MOV.B   @ADDRD  , R4      ; Read ADDRD
        BCLR    #5      , @ADCSR  ; Clear ADST
        BCLR    #7      , @ADCSR  ; Clear ADF
```

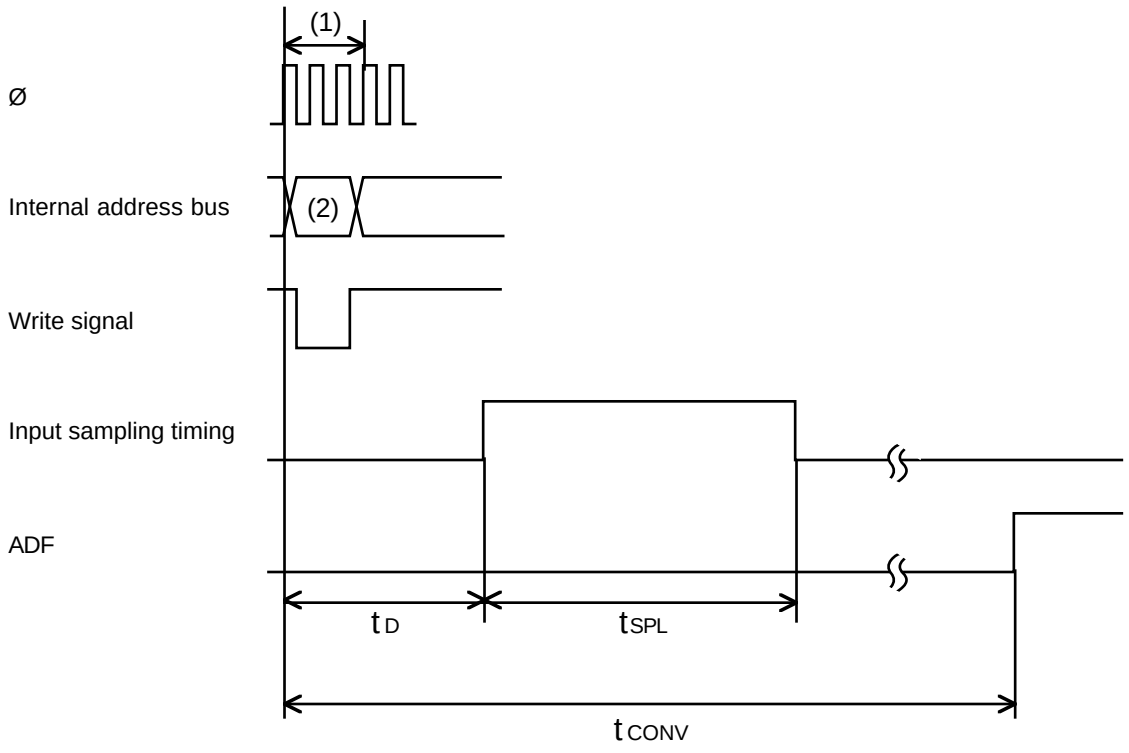
(It is not necessary to clear the ADST bit in order to read ADDRA to ADDRD.)

10.3.3 Input Sampling Time and A/D Conversion Time

The A/D converter includes a built-in sample-and-hold circuit. Sampling of the input starts at a time t_D after the ADST bit is set to "1." The sampling process lasts for a time t_{SPL}. The actual A/D conversion begins after sampling is completed. Figure 10-5 shows the timing of these steps. Table 10-4 (a) lists the conversion times for the single mode. Table 10-4 (b) lists the conversion times for the scan mode.

The total conversion time (t_{CONV}) includes t_D and t_{SPL}. The purpose of t_D is to synchronize the ADCSR write time with the A/D conversion process, so the length of t_D is variable. The total conversion time therefore varies within the minimum to maximum ranges indicated in table 10-4 (a) and (b).

In the scan mode, the ranges given in table 10-4 (b) apply to the first conversion. The length of the second and subsequent conversion processes is fixed at 256 states (when CKS = "0") or 128 states (when CKS = "1").



(Notation)

(1)	ADCSR write cycle
(2)	ADCSR address
t_D	Synchronization delay
t_{SPL}	Input sampling time
t_{CONV}	Total A/D conversion time

Figure 10-5. A/D Conversion Timing

Table 10-4 (a). A/D Conversion Time (Single Mode)

		CKS = "0"			CKS = "1"		
Item	**Symbol**	**min**	**typ**	**max**	**min**	**typ**	**max**
Synchronization delay	t_D	18	—	33	10	—	17
Input sampling time	t_{SPL}	—	63	—	—	31	—
Total A/D conversion time	t_{CONV}	227	—	242	115	—	122

Table 10-4 (b). A/D Conversion Time (Scan Mode)

		CKS = "0"			CKS = "1"		
Item	**Symbol**	**min**	**typ**	**max**	**min**	**typ**	**max**
Synchronization delay	t_D	18	—	33	10	—	17
Input sampling time	t_{SPL}	—	63	—	—	31	—
Total A/D conversion time	t_{CONV}	259	—	274	131	—	138

Note: Values in the tables above are numbers of states.

10.3.4 External Trigger Input Timing

A/D conversion can be started by external trigger input at the $\overline{\text{ADTRG}}$ pin. This input is enabled or disabled by the TRGE bit in the A/D control register (ADCR). If the TRGE bit is set to "1," when a falling edge of $\overline{\text{ADTRG}}$ is detected the ADST bit is set to "1" and A/D conversion begins. Subsequent operation is the same as when the ADST bit is set to "1" by software.

Figure 10-6 shows the trigger timing.

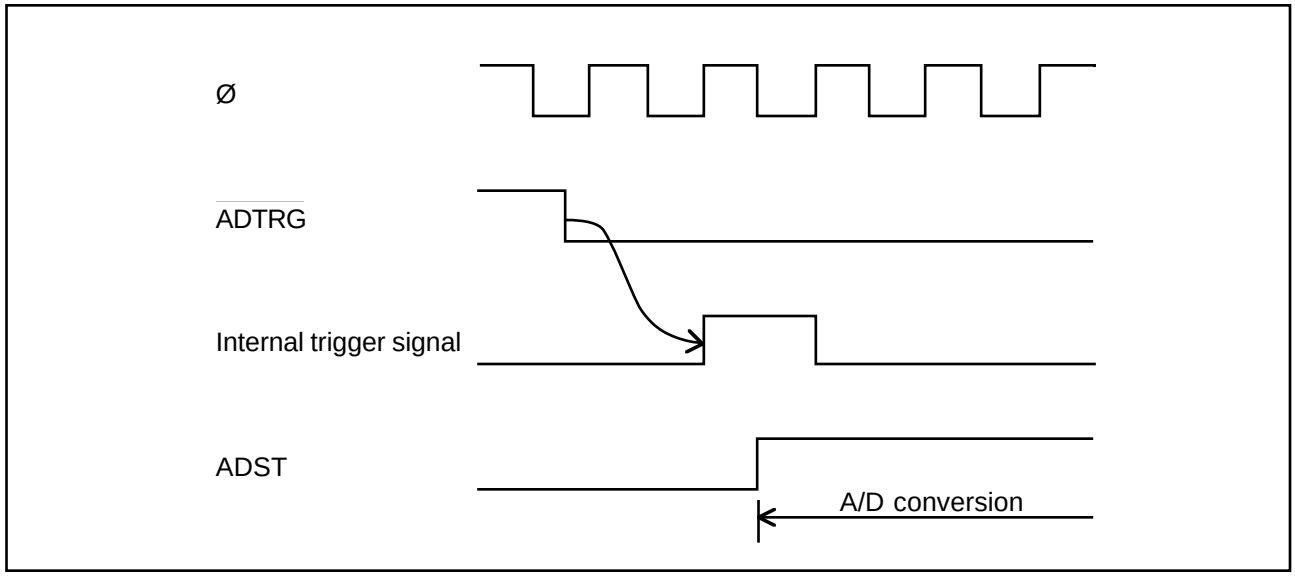


Figure 10-6. External Trigger Input Timing

10.4 Interrupts

The A/D conversion module generates an A/D-end interrupt request (ADI) at the end of A/D conversion.

The ADI interrupt request can be enabled or disabled by the ADIE bit in the A/D control/status register (ADCSR).

Section 11. Dual-Port RAM (Parallel Communication Interface)

11.1 Overview

The H8/330 has an on-chip dual-port RAM (DPRAM) that can be accessed by both the CPU on the H8/330 chip and a master CPU on another chip. The dual-port RAM can be used only in the single-chip mode (mode 3), and only when the DPME bit in the system control register (SYSCR) is set to "1." The dual-port-RAM-enabled mode is called slave mode because it is designed for a master-slave system in which the dual-port RAM provides a parallel communication interface with a master CPU.

In this section the CPU on the H8/330 chip will be referred to as the H8/300 CPU.

11.1.1 Features

- 15-Byte capacity
  Fifteen 8-bit parallel communication data registers
- Standard external memory interface
  The master CPU can be connected to the dual-port RAM in the same way as to a memory chip.
- Simple data-transfer protocol
- Can generate master CPU interrupts

11.1.2 Block Diagram

Figure 11-1 shows a block diagram of the dual-port RAM.

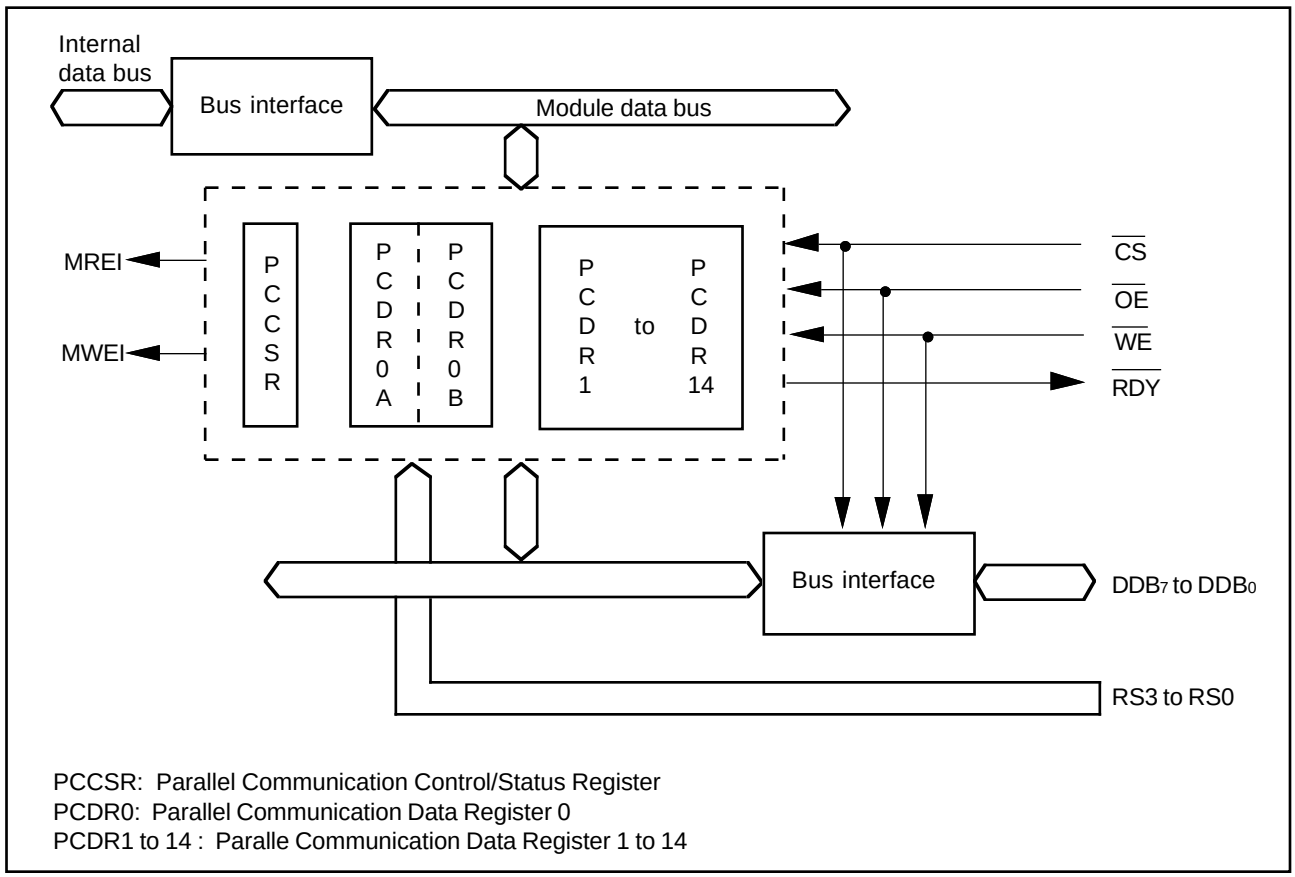


Figure 11-1. Block Diagram of Dual-Port RAM

11.1.3 Input and Output Pins

Table 11-1 lists the input and output pins of the dual-port RAM.

Table 11-1. Dual-Port RAM Input and Output Pins

Name	Abbreviation	I/O	Function
DPRAM data bus	DDB_7 to DDB_0	Input/output	An 8-bit parallel data bus by which the master CPU can access the dual-port RAM.
Chip select	$\overline{CS}$	Input	Chip select input pin for selecting the dual-port RAM.
Register select	RS_3 to RS_0	Input	Dual-port RAM address input.
Output enable	$\overline{OE}$	Input	Enables output on the DPRAM data bus.
Write enable	$\overline{WE}$	Input	Enables data to be written in the dual-port RAM via the DPRAM data bus.
Ready	$\overline{RDY}$	Output *	Indicates that the dual-port RAM is ready to be written or read by the master CPU. (NMOS open-drain output)

* NMOS open drain output.

11.1.4 Register Configuration

Table 11-2 lists the registers of the dual-port RAM.

Table 11-2. Dual-Port RAM Register Configuration

		Read/write				External address			
		H8/300	Master	Initial	On-chip				
Name	Abbr.	CPU	CPU	value	address	RS_3	RS_2	RS_1	RS_0
Parallel communication control/status register	PCCSR	R/(W)*	R/(W)*	H’00	H’FFF0	0	0	0	0
Parallel communication data register 0	PCDR0A PCDR0B	R W	W R	Undeter-mined	H’FFF1 H’FFF1	0 0	0 0	0 0	1 1
Parallel communication data register 1	PCDR1	R/W	R/W	Undeter-mined	H’FFF2	0	0	1	0
Parallel communication data register 2	PCDR2	R/W	R/W	Undeter-mined	H’FFF3	0	0	1	1
Parallel communication data register 3	PCDR3	R/W	R/W	Undeter-mined	H’FFF4	0	1	0	0

Table 11-2. Dual-Port RAM Register Configuration (cont.)

Name	Abbr.	Read/write: H8/300 CPU	Read/write: Master CPU	Initial value	On-chip address	External address: RS_3	RS_2	RS_1	RS_0
Parallel communication data register 4	PCDR4	R/W	R/W	Undetermined	H’FFF5	0	1	0	1
Parallel communication data register 5	PCDR5	R/W	R/W	Undetermined	H’FFF6	0	1	1	0
Parallel communication data register 6	PCDR6	R/W	R/W	Undetermined	H’FFF7	0	1	1	1
Parallel communication data register 7	PCDR7	R/W	R/W	Undetermined	H’FFF8	1	0	0	0
Parallel communication data register 8	PCDR8	R/W	R/W	Undetermined	H’FFF9	1	0	0	1
Parallel communication data register 9	PCDR9	R/W	R/W	Undetermined	H’FFFA	1	0	1	0
Parallel communication data register 10	PCDR10	R/W	R/W	Undetermined	H’FFFB	1	0	1	1
Parallel communication data register 11	PCDR11	R/W	R/W	Undetermined	H’FFFC	1	1	0	0
Parallel communication data register 12	PCDR12	R/W	R/W	Undetermined	H’FFFD	1	1	0	1
Parallel communication data register 13	PCDR13	R/W	R/W	Undetermined	H’FFFE	1	1	1	0
Parallel communication data register 14	PCDR14	R/W	R/W	Undetermined	H’FFFF	1	1	1	1

Note: The H8/300 CPU can write only bits 6, 4, and 2 of the PCCSR. The master CPU can write only bit 4.

11.2 Register Descriptions

11.2.1 Dual Port RAM Enable Bit (DPME)

The dual-port RAM is enabled or disabled by the DPRAM Enable (DPME) bit in the system control register (SYSCR). In the extended modes the dual-port RAM is always disabled. In the single-chip mode, the dual-port RAM is initially disabled but can be enabled by setting the DPME bit to "1."

System Control Register (SYSCR)—H'FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W

The only bit in the system control register that concerns the dual-port RAM is the DPME bit. See section 2.4.2, "System Control Register" for the other bits.

Bit 1 – Dual-Port RAM Enable (DPME): This bit enables or disables the dual-port RAM. The dual-port RAM can be enabled only in the single-chip mode (mode 3).

Bit 1

DPME	Description	
0	The dual-port RAM is disabled.	(Initial value)
1	The dual-port RAM is enabled (in single-chip mode only).	

If the DPME bit is set to "1" while the H8/330 is operating in the single-chip mode, the following pins are automatically assigned to dual-port RAM functions, regardless of their data direction register settings:

Port 3 ($P3_7$ to $P3_0$) $\rightarrow$ DDB_7 to DDB_0 (parallel communication data bus; input/output)
Port 8 ($P8_3$ to $P8_0$) $\rightarrow$ RS3 to RS0 (dual-port RAM register select; input)
In port 9, $P9_7 \rightarrow \overline{WE}$ (Write Enable; input)
$P9_5 \rightarrow \overline{RDY}$ (Ready; output)
$P9_4 \rightarrow \overline{OE}$ (Output Enable; input)
$P9_3 \rightarrow \overline{CS}$ (Chip Select; input)

The DPME bit is initialized to "0" by a reset and in the hardware standby mode.

Setting the DPME bit to "1" in the expanded modes (modes 1 and 2) has no effect.

11.2.2 Parallel Communication Data Register 0 (PCDR0) – H'FFF1

(a) Parallel Communication Data Register 0A (PCDR0A)

Bit		7	6	5	4	3	2	1	0
Initial value		—	—	—	—	—	—	—	—
R/W	H8/300 CPU	R	R	R	R	R	R	R	R
	Master CPU	W	W	W	W	W	W	W	W

(b) Parallel Communication Data Register 0B (PCDR0B)

Bit		7	6	5	4	3	2	1	0
Initial value		—	—	—	—	—	—	—	—
R/W	H8/300 CPU	W	W	W	W	W	W	W	W
	Master CPU	R	R	R	R	R	R	R	R

Parallel communication data register 0 consists of two separate 8-bit registers with the same address. As shown in Figure 11-2, PCDR0A is written by the H8/300 CPU and read by the master CPU; PCDR0B is written by the master CPU and read by the H8/300 CPU. This arrangement prevents contention even if both CPUs write to PCDR0 at the same time. When either CPU reads PCDR0, it is assured of reading data written by the other CPU.

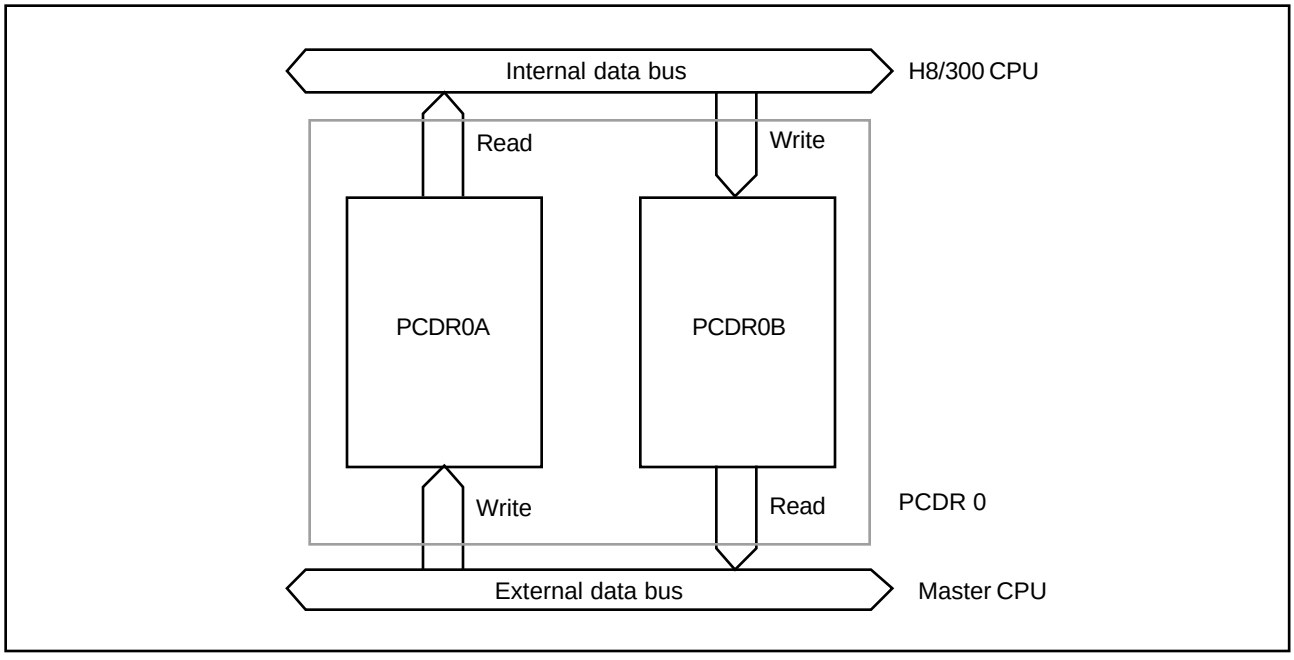


Figure 11-2. Parallel Communication Data Register 0

The value in PCDR0 after a reset is undetermined. In non-slave modes, the value obtained by reading PCDR0 is unpredictable.

11.2.3 Parallel Communication Data Registers 1 to 14 – H'FFF2 (PCDR1) to H'FFFF (PCDR1-14)

Bit		7	6	5	4	3	2	1	0
Initial value		—	—	—	—	—	—	—	—
R/W	H8/300 CPU	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	Master CPU	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Parallel communication data registers 1 to 14 are 8-bit registers which can be written and read by either the H8/300 CPU or the master CPU. The H8/300 CPU can read and write these registers regardless of the operating mode of the H8/330 chip. The master CPU can read and write them only when the H8/330 chip is operating in slave mode.

In non-slave modes, these registers can be used as 14 bytes of data memory. Note that access requires three states per byte, which is slower than the on-chip RAM.

The values in PCDR1 to PCDR14 after a reset are undetermined.

11.2.4 Parallel Communication Control/Status Register (PCCSR) – H'FFF0

Bit		7	6	5	4	3	2	1	0
		MWEF	EMWI	SWEF	EAKAR	MREF	EMRI	MWMF	SWMF
Initial value		0	0	0	0	0	0	0	0
R/W	H8/300 CPU	R	R/W	R	R/W	R	R/W	R	R
	Master CPU	R	R	R	R/W	R	R	R	R

The PCCSR is an 8-bit readable and partly writable register that provides protocol and interrupt control functions. Either CPU can read and write bit 4, which enables the RDY signal. The H8/300 CPU can read and write bits 6, 4 and 2, which enable interrupts. The other bits are read-only bits.

The PCCSR is initialized to H'00 at a reset and in the standby modes.

In the bit names that follow, the H8/300 is referred to as the slave and the master CPU as the master.

Bit 7 – Master Write End Flag (MWEF): This flag bit is used to indicate that the master CPU has finished writing data in the parallel communication data registers. It is set when the master CPU writes to PCDR14 and cleared when the H8/300 CPU reads PCDR14.

Bit 7

MWEF	Description	
0	The H8/300 CPU has read PCDR14 while the dual-port RAM was in the master write mode (MWMF = "1").	(Initial state)
1	The master CPU has written data in PCDR14.	

Bit 6 – Enable Master Write Interrupt (EMWI): This bit enables or disables the master write end interrupt (MWEI).

Bit 6

EMWI	Description	
0	The master write end interrupt request (MWEI) is disabled.	(Initial state)
1	The master write end interrupt request (MWEI) is enabled.	

Bit 5 – Slave Write End Flag (SWEF): This flag bit is used to indicate that the H8/300 CPU has finished writing data in the parallel communication data registers. It is set when the H8/300 CPU writes to PCDR14 and cleared when the master CPU reads PCDR14.

Bit 5

SWEF	Description	
0	The master CPU has read PCDR14.	(Initial state)
1	The H8/300 CPU has written data in PCDR14.	

Bit 4 – Enable Acknowledge and Request (EAKAR): This bit enables or disables the $\overline{\text{RDY}}$ signal output by the H8/330 chip. If enabled:

- The $\overline{\text{RDY}}$ signal goes Low when the H8/300 CPU reads PCDR0 while the dual-port RAM is in the master write mode (MWMF = "1"), or when the H8/300 CPU writes to PCDR14.
- The $\overline{\text{RDY}}$ signal goes High when the master CPU reads PCDR14 or the PCCSR, or when either the master or H8/300 CPU writes to PCDR0.

In the non-slave modes this bit has no effect.

Bit 4

EAKAR	Description	
0	$\overline{\text{RDY}}$ output is disabled. $\overline{\text{RDY}}$ remains in the high-impedance state.	(Initial state)
1	$\overline{\text{RDY}}$ output is enabled.	

Bit 3 – Master Read End Flag (MREF): This flag indicates whether the master CPU has finished reading data set in the parallel communication data registers.

Bit 3

MREF	Description	
0	This bit is cleared to "0" when: • The H8/300 CPU reads or writes PCDR0. • The master CPU writes to PCDR0.	(Initial state)
1	This bit is set to "1" when the master CPU reads PCDR0.	

Bit 2 – Enable Master Read Interrupt (EMRI): This bit enables or disables the master read end interrupt (MREI).

Bit 2

EMRI	Description	
0	The master read end interrupt request (MREI) is disabled.	(Initial state)
1	The master read end interrupt request (MREI) is enabled.	

Bit 1 – Master Write Mode Flag (MWMF): This bit indicates when the dual-port RAM is in the master write mode. The master CPU should check that this bit is set to "1" before writing to parallel communication data registers 1 to 14. The H8/300 CPU cannot write in those registers while this bit is set to "1."

Bit 1

MWMF	Description	
0	This bit is cleared to "0" when the H8/300 CPU reads PCDR0. The dual-port RAM is not in the master write mode. The master CPU should avoid writing in PCDR1 to PCDR14.	(Initial state)
1	This bit is set to "1" if the master CPU writes to PCDR0 while the SWMF flag is cleared to "0." The dual-port RAM is in the master write mode. Only the master CPU can write in PCDR1 to PCDR14.	

Bit 0 – Slave Write Mode Flag (SWMF): This bit indicates when the dual-port RAM is in the slave write mode. The H8/300 CPU should check that this bit is set to “1” before writing to parallel communication data registers 1 to 14. The master CPU cannot write in those registers while this bit is set to “1.”

Bit 0 SWMF	Description	
0	This bit is cleared to “0” when the master CPU reads PCDR0. The dual-port RAM is not in the slave write mode. The H8/300 CPU should avoid writing in PCDR1 to PCDR14.	(Initial state)
1	This bit is set to “1” if the H8/300 CPU writes to PCDR0 while the MWMF flag is cleared to “0.” The dual-port RAM is in the slave write mode. Only the H8/300 CPU can write in PCDR1 to PCDR14.	

11.3 Usage

The dual-port RAM has a simple protocol for controlling the use of the data registers and parallel communication data bus. The basic rule is that when either CPU writes to the dual-port RAM, it should write to PCDR0 first and PCDR14 last. Conversely, in reading the dual-port RAM, the CPU should read PCDR14 first and PCDR0 last.

Procedures for data transfer in both directions are given below. Figure 11-3 shows a timing chart.

11.3.1 Data Transfer from Master CPU to H8/300 CPU

The following procedure should be used when the master CPU sends data to the H8/300 CPU via the dual-port RAM:

(1) The master CPU writes the first byte of data in PCDR0. If the dual-port RAM is not currently in the slave write mode, MWMF is set to "1," placing it in the master write mode and preventing the H8/300 CPU from writing in PCDR1 to PCDR14.

(2) The master CPU reads the PCCSR and checks MWMF. If MWMF is set to “1,” the master CPU may continue writing in PCDR1 to PCDR14. If MWMF is cleared to "0," the dual-port RAM is presumably in the slave write mode.

(3) The master CPU writes data in PCDR1 to PCDR13 as required, then writes the last byte in PCDR14. This sets the master write end flag (MWEF) to “1,” notifying the H8/300 CPU that the master CPU has finished writing. If EMWI is set to “1,” a master write end interrupt is requested.

(4) After the master CPU has finished writing data, the H8/300 CPU first reads PCDR14. This clears the master write end flag. Then the H8/300 CPU reads data from PCDR1 to PCDR13 as required. Finally, the H8/300 CPU reads PCDR0. This clears MWMF, so the dual-port RAM is no longer in the master write mode. If the EAKAR bit is set to “1,” the $\overline{\text{RDY}}$ signal goes Low to acknowledge the received data.

(5) If the master CPU has more data to send, it should check that MWMF is cleared to "0," then repeat the above procedure from step (1). If MWMF is still set to "1," that indicates that the H8/300 CPU has not read all the data sent previously.

11.3.2 Data Transfer from H8/300 CPU to Master CPU

The following procedure should be used when the H8/300 CPU sends data to the master CPU via the dual-port RAM:

(1) The H8/300 CPU writes the first byte of data in PCDR0. If the dual-port RAM is not currently in the master write mode, SWMF is set to "1," placing it in the slave write mode and preventing the master CPU from writing in PCDR1 to PCDR14.

(2) The H8/300 CPU reads the PCCSR and checks SWMF. If SWMF is set to “1,” the H8/300 CPU may continue writing in PCDR1 to PCDR14. If SWMF is cleared to "0," the dual-port RAM is presumably in the master write mode.

(3) The H8/300 CPU writes data in PCDR1 to PCDR13 as required, then writes the last byte in PCDR14. This sets the slave write end flag (SWEF) to “1.” If the EAKAR bit is set to “1,” the $\overline{\text{RDY}}$ signal goes Low to notify the master CPU that the H8/300 CPU has finished writing.

(4) After the H8/300 CPU has finished writing data, the master CPU first reads PCDR14. This clears the slave write end flag. Then the master CPU reads data from PCDR1 to PCDR13 as required. Finally, the master CPU reads PCDR0. This clears the SWMF bit, so the dual-port RAM is no longer in the slave write mode. It also sets the master read end flag (MREF). If EMRI is set to “1,” a master read end interrupt is requested to notify the H8/300 CPU that the master CPU has finished reading the data.

(5) If the H8/300 CPU has more data to send, it should check that SWMF is cleared to "0," then repeat the above procedure from step (1). If SWMF is still set to "1," that indicates that the master CPU has not read all the data sent previously.

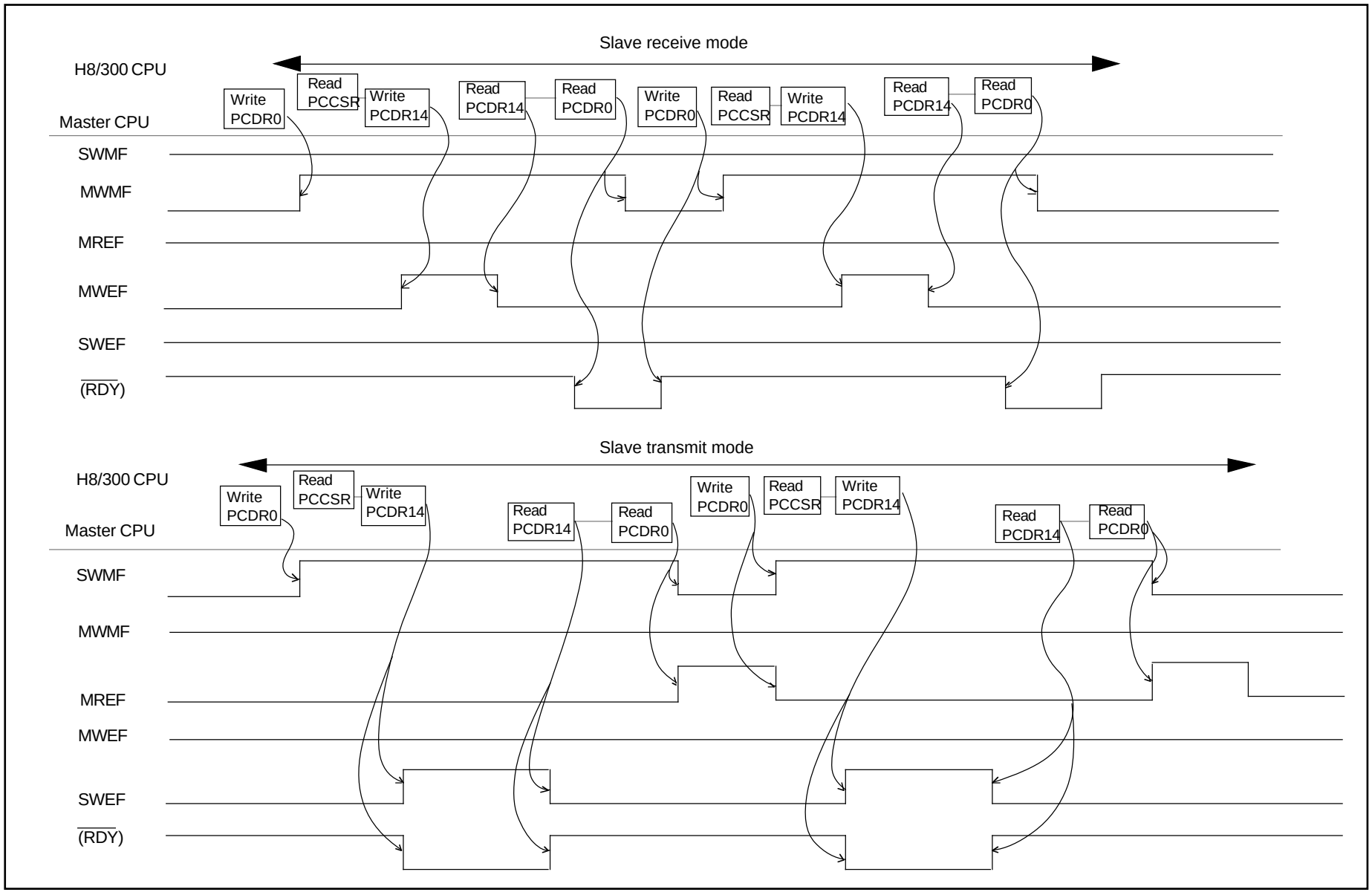
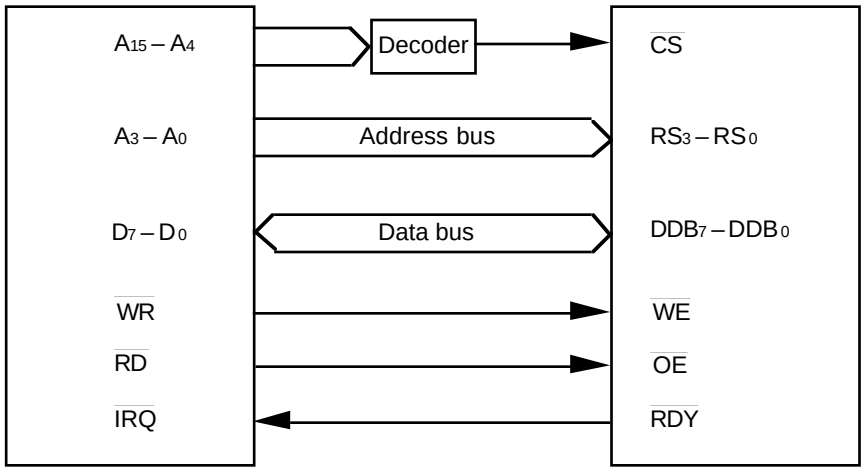


Figure 11-3. Dual-Port RAM Timing Chart

11.4 Master-Slave Interconnections

Figure 11-4 shows an example of the master-slave interconnections when the master chip is an H8/532.



Note: An external pull-up resistor should be connected to the RDY output pin.

Figure 11-4. Interconnection to H8/532 (Example)

Section 12. RAM

12.1 Overview

The H8/330 includes 512 bytes of on-chip static RAM, connected to the CPU by a 16-bit data bus. Both byte and word access to the on-chip RAM are performed in two states, enabling rapid data transfer and instruction execution.

The on-chip RAM is assigned to addresses H’FD80 to H’FF7F in the chip’s address space. The RAME bit in the system control register (SYSCR) can enable or disable the on-chip RAM, permitting these addresses to be allocated to external memory instead, if so desired.

12.2 Block Diagram

Figure 12-1 is a block diagram of the on-chip RAM.

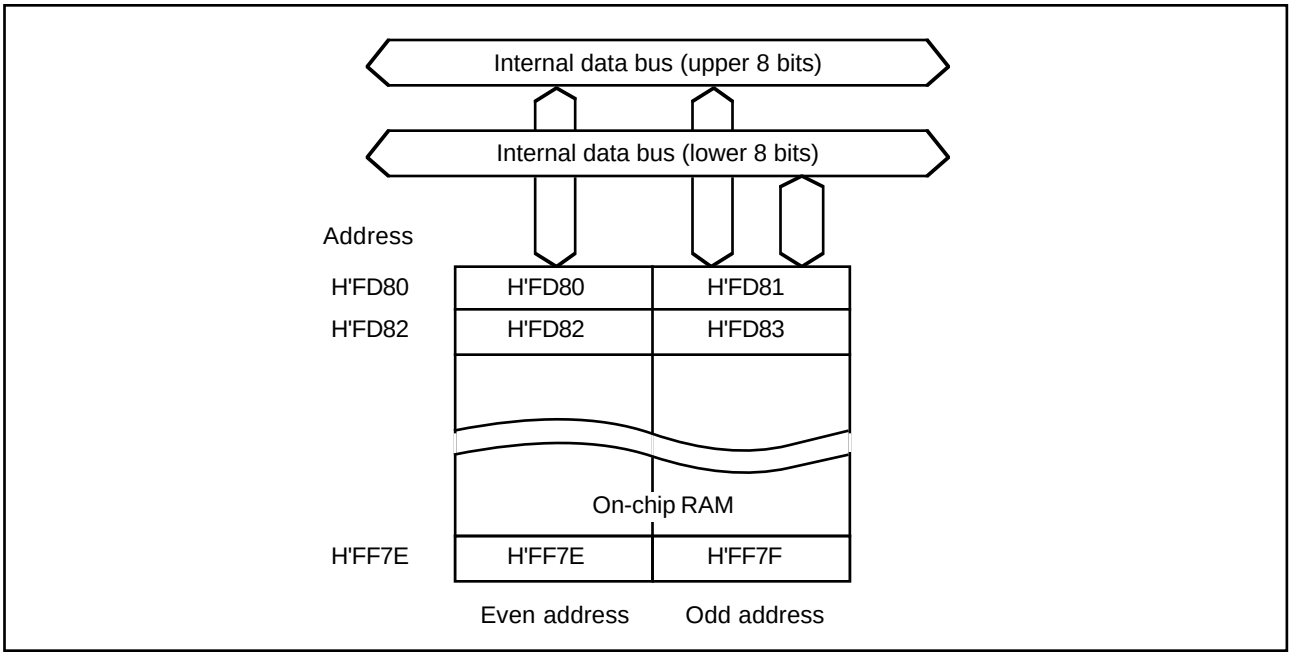


Figure 12-1. Block Diagram of On-Chip RAM

12.3 RAM Enable Bit (RAME)

The on-chip RAM is enabled or disabled by the RAME (RAM Enable) bit in the system control register (SYSCR). Table 12-1 lists information about the system control register.

Table 12-1. System Control Register

Name	Abbreviation	R/W	Initial value	Address
System control register	SYSCR	R/W	H’09	H’FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W

The only bit in the system control register that concerns the on-chip RAM is the RAME bit. See section 2.4.2, "System Control Register" for the other bits.

Bit 0 – RAM Enable (RAME): This bit enables or disables the on-chip RAM.

The RAME bit is initialized to “1” on the rising edge of the $\overline{\text{RES}}$ signal, so a reset enables the on-chip RAM. The RAME bit is not initialized in the software standby mode.

Bit 7

RAME	Description	
0	On-chip RAM is disabled.	
1	On-chip RAM is enabled.	(Initial value)

12.4 Operation

12.4.1 Expanded Modes (Modes 1 and 2)

If the RAME bit is set to “1,” accesses to addresses H’FD80 to H’FF7F are directed to the on-chip RAM. If the RAME bit is cleared to “0,” accesses to addresses H’FD80 to H’FF7F are directed to the external data bus.

12.4.2 Single-Chip Mode (Mode 3)

If the RAME bit is set to “1,” accesses to addresses H’FD80 to H’FF7F are directed to the on-chip RAM.

If the RAME bit is cleared to “0,” the on-chip RAM data cannot be accessed. Attempted write access has no effect. Attempted read access always results in H’FF data being read.

Section 13. ROM

13.1 Overview

The H8/330 includes 16K bytes of high-speed, on-chip ROM. The on-chip ROM is connected to the CPU via a 16-bit data bus. Both byte data and word data are accessed in two states, enabling rapid data transfer and instruction fetching.

The H8/330 is available in two versions: one with electrically programmable ROM (PROM); the other with masked ROM. The PROM version has a PROM mode in which the chip can be programmed with a standard PROM writer.

The on-chip ROM is enabled or disabled depending on the MCU operating mode, which is determined by the inputs at the mode pins (MD_1 and MD_0) when the chip comes out of the reset state. See table 13-1.

Table 13-1. On-Chip ROM Usage in Each MCU Mode

	Mode pins		
Mode	**MD_1**	**MD_0**	**On-chip ROM**
Mode 1 (expanded mode)	0	1	Disabled (external addresses)
Mode 2 (expanded mode)	1	0	Enabled
Mode 3 (single-chip mode)	1	1	Enabled

13.1.1 Block Diagram

Figure 13-1 is a block diagram of the on-chip ROM.

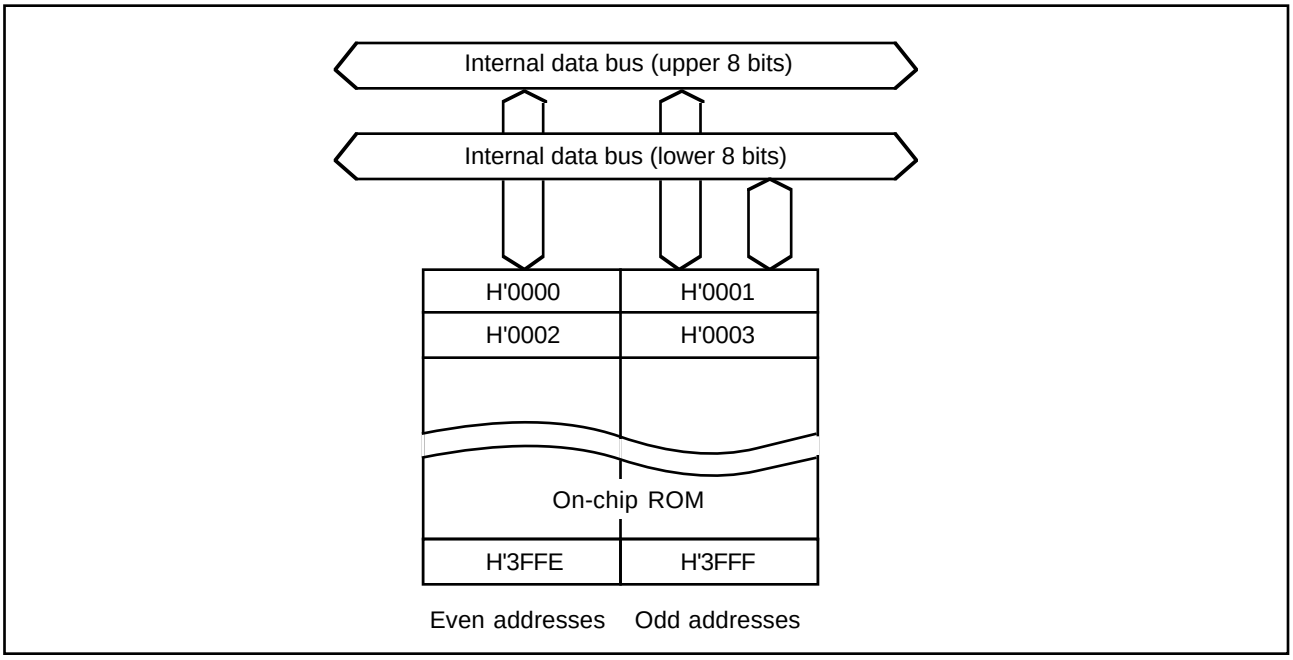


Figure 13-1. Block Diagram of On-Chip ROM

13.2 PROM Mode

13.2.1 PROM Mode Setup

In the PROM mode of the PROM version of the H8/330, the usual microcomputer functions are halted to allow the on-chip PROM to be programmed. The programming method is the same as for the HN27C256.

To select the PROM mode, apply the signal inputs listed in Table 13-2.

Table 13-2. Selection of PROM Mode

Pin	Input
Mode pin MD_1	Low
Mode pin MD_0	Low
$\overline{\text{STBY}}$ pin	Low
Pins $P8_0$ and $P8_1$	High

13.2.2 Socket Adapter Pin Assignments and Memory Map

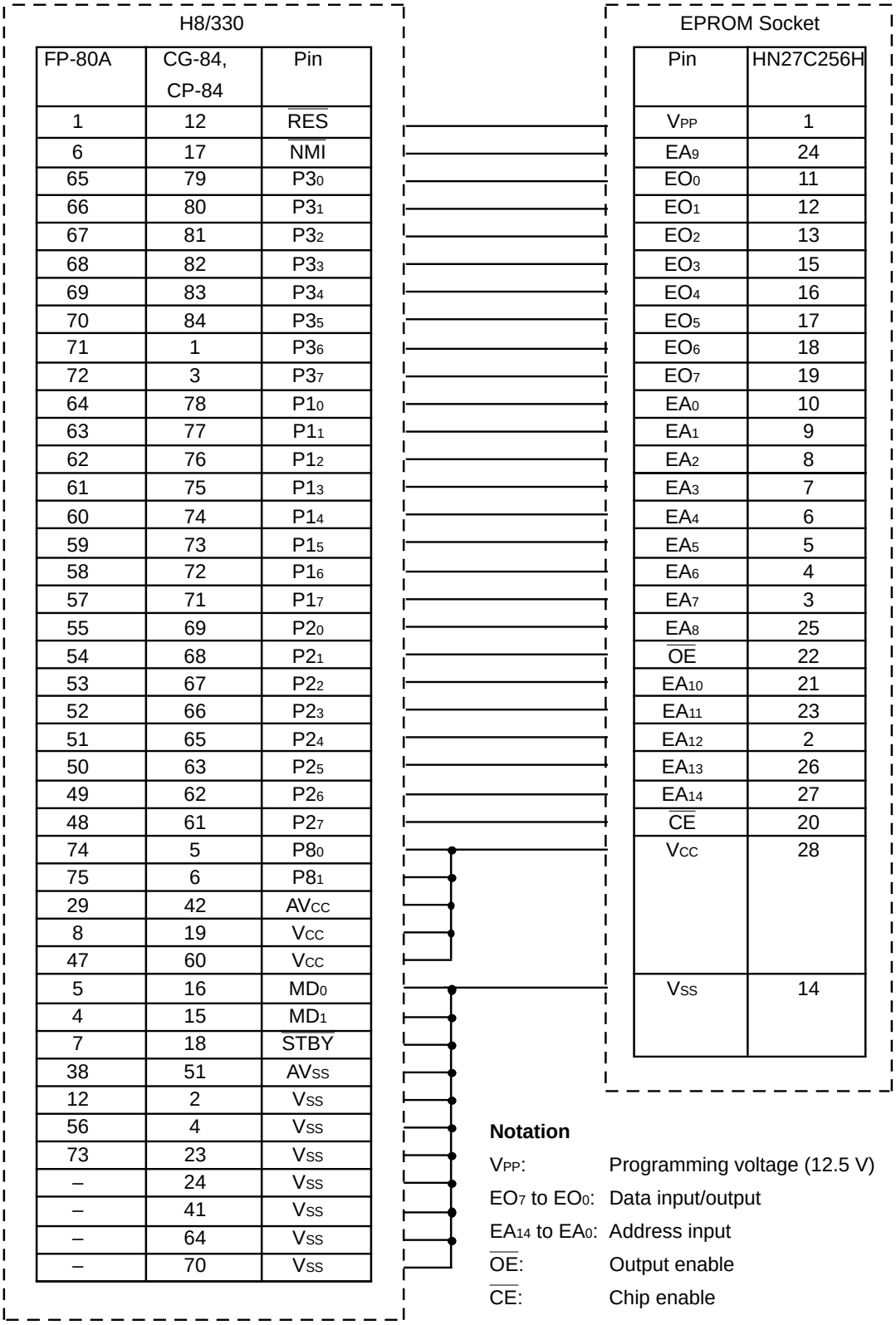
The H8/330 can be programmed with a general-purpose PROM writer. Since the H8/330 package has 80 or 84 pins instead of 28, a socket adapter is necessary. Table 13-3 lists recommended socket adapters. Figure 13-2 shows the socket adapter pin assignments by giving the correspondence between H8/330 pins and HN27C256 pin functions.

Figure 13-3 shows a memory map in the PROM mode. Since the H8/330 has only 16K bytes of on-chip PROM, the address range should be specified as H’0000 to H’3FFF. H’FF data should be specified for unused address areas.

It is important to limit the program address range to H'0000 to H'3FFF and specify H'FF data for H'4000 and higher addresses. If data (other than H’FF) are written by mistake in addresses equal to or greater than H’4000, it may become impossible to program or verify the PROM data. With a windowed package, it is possible to erase the data and reprogram, but this cannot be done with a plastic package, so particular care is required.

Table 13-3. Recommended Socket Adapters

Package	Recommended Socket Adapter
84-Pin PLCC	HS338ESC01H
84-Pin windowed LCC	HS338ESG01H
80-Pin QFP	HS338ESH01H



Note: All pins not listed in this figure should be left open.

Figure 13-2. Socket Adapter Pin Assignments

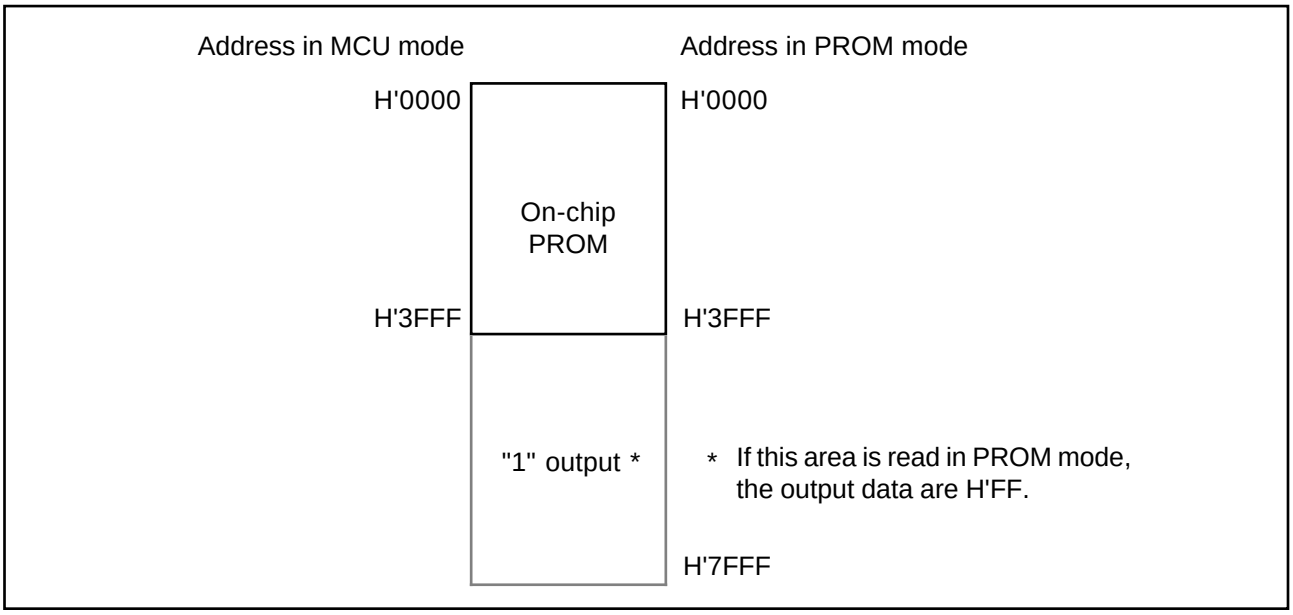


Figure 13-3. Memory Map in PROM Mode

13.3 Programming

The write, verify, inhibited, and read sub-modes of the PROM mode are selected as shown in Table 13-4.

Table 13-4. Selection of Sub-Modes in PROM Mode

	Pins					
Sub-mode	**$\overline{CE}$**	**$\overline{OE}$**	**V_{PP}**	**V_{CC}**	**$EO_7 – EO_0$**	**$EA_{14} – EA_0$**
Write	Low	High	V_{PP}	V_{CC}	Data input	Address input
Verify	High	Low	V_{PP}	V_{CC}	Data output	Address input
Programming inhibited	High	High	V_{PP}	V_{CC}	High-impedance	Address input

Note: The V_{PP} and V_{CC} pins must be held at the V_{PP} and V_{CC} voltage levels.

The H8/330 PROM uses the same, standard read/write specifications as the HN27C256 and HN27256.

13.3.1 Writing and Verifying

An efficient, high-speed programming procedure can be used to write and verify PROM data. This procedure writes data quickly without subjecting the chip to voltage stress and without sacrificing data reliability. It leaves the data H’FF written in unused addresses.

Figure 13-4 shows the basic high-speed programming flowchart.

Tables 13-5 and 13-6 list the electrical characteristics of the chip in the PROM mode. Figure 13-5 shows a write/verify timing chart.

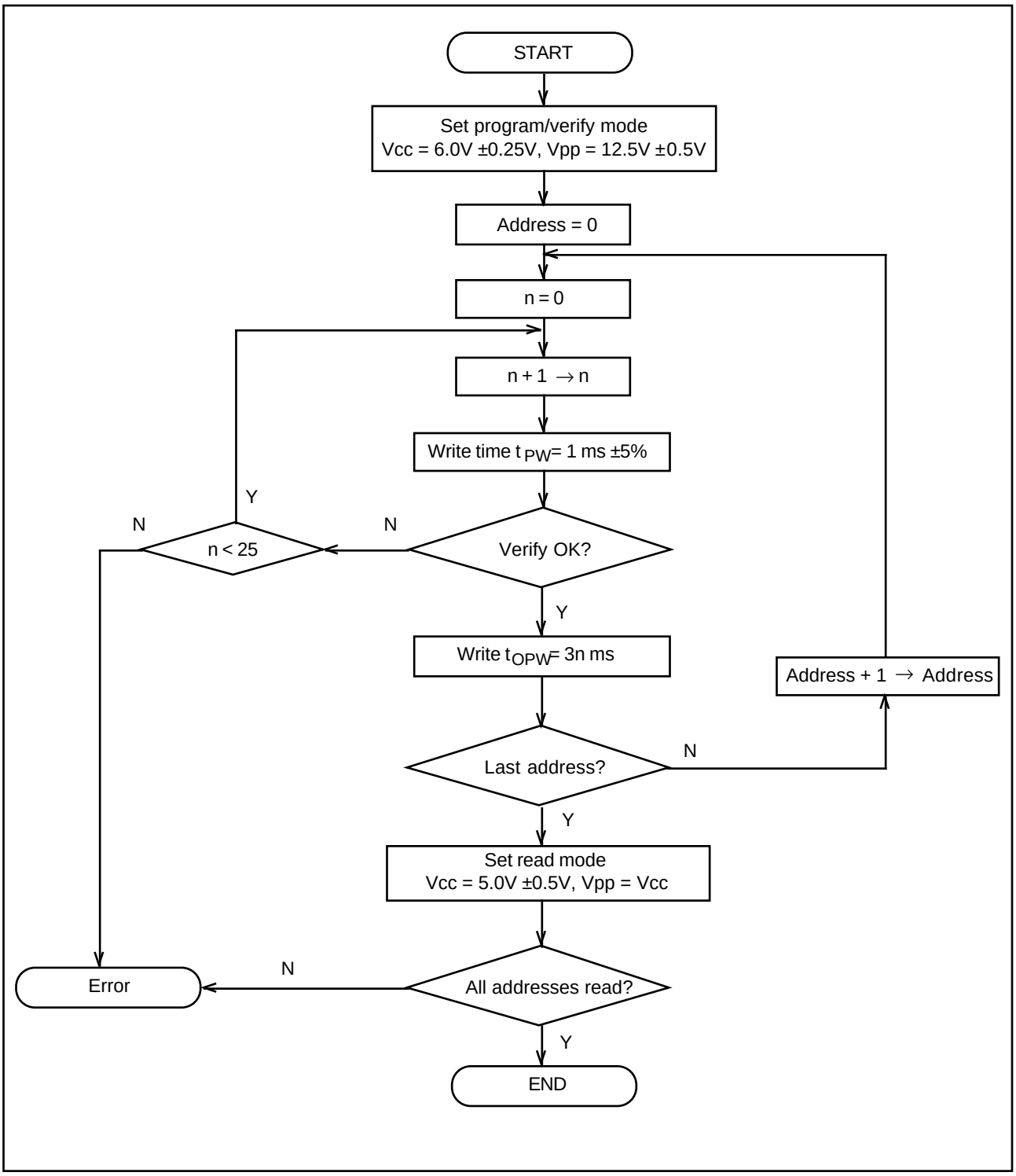


Figure 13-4. High-Speed Programming Flowchart

Table 13-5. DC Characteristics (When V_{CC} = 6.0V ±0.25V, V_{PP} = 12.5V ±0.3V, V_{SS} = 0V, Ta = 25˚C ±5˚C)

Item		Symbol	min	typ	max	Unit	Measurement conditions
Input High voltage	EO_7 – EO_0, EA_{14} – EA_{10}, EA_8 – EA_0, $\overline{OE}$, $\overline{CE}$	V_{IH}	2.4	—	V_{CC} + 0.3	V	
	EA_9		$V_{CC} \times 0.7$	—	V_{CC} + 0.3	V	
Input Low voltage	EO_7 – EO_0, EA14 – EA_0, $\overline{OE}$, $\overline{CE}$	V_{IL}	– 0.3	—	0.8	V	
Output High voltage	EO_7 – EO_0	V_{OH}	2.4	—	—	V	I_{OH} = –200µA
Output Low voltage	EO_7 – EO_0	V_{OL}	—	—	0.45	V	I_{OL} = 1.6mA
Input leakage current	EO_7 – EO_0, EA_{14} – EA_0, $\overline{OE}$, $\overline{CE}$	$\|I_{LI}\|$	—	—	2	µA	V_{in} = 5.25V/ 0.5V
Vcc current		I_{CC}	—	—	40	mA	
Vpp current		Ipp	—	—	40	mA	

Table 13-6. AC Characteristics (When V_{CC} = 6.0V ±0.25V, V_{PP} = 12.5V ±0.3V, Ta = 25˚C ±5˚C)

Item	Symbol	min	typ	max	Unit	Measurement conditions
Address setup time	t_{AS}	2	—	—	µs	See Figure 13-5*
$\overline{OE}$ setup time	t_{OES}	2	—	—	µs	
Data setup time	t_{DS}	2	—	—	µs	
Address hold time	t_{AH}	0	—	—	µs	
Data hold time	t_{DH}	2	—	—	µs	
Data output disable time	t_{DF}	—	—	130	ns	
Vpp setup time	t_{VPS}	2	—	—	µs	
Program pulse width	t_{PW}	0.95	1.0	1.05	ms	

* Input pulse level: 0.8V to 2.2V
Input rise/fall time ≦ 20ns
Timing reference levels: input—1.0V, 2.0V; output—0.8V, 2.0V

Table 13-6. AC Characteristics (cont.)

(When V_{CC} = 6.0V ±0.25V, V_{PP} = 12.5V ±0.3V, Ta = 25˚C ±5˚C)

Item	Symbol	min	typ	max	Unit	Measurement conditions
$\overline{OE}$ pulse width for overwrite-programming	t_{OPW}	2.85	—	78.75	ms	See Figure 13-5*
Vcc setup time	t_{VCS}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	500	ns	

* Input pulse level: 0.8V to 2.2V
Input rise/fall time ≦ 20ns
Timing reference levels: input—1.0V, 2.0V; output—0.8V, 2.0V

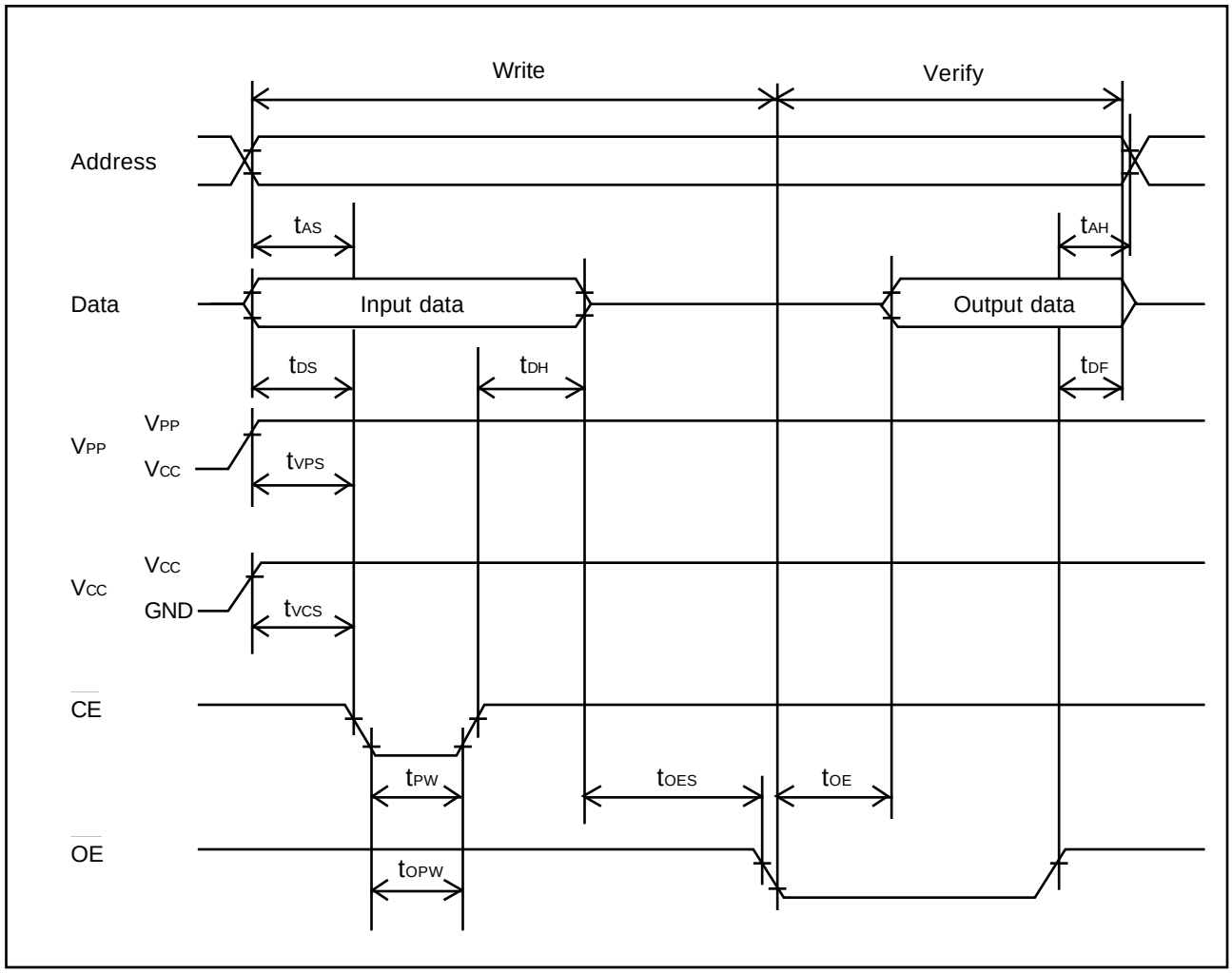


Figure 13-5. PROM Write/Verify Timing

13.3.2 Notes on Writing

(1) Write with the specified voltages and timing. The programming voltage (Vpp) is 12.5V.

Caution: Applied voltages in excess of the specified values can permanently destroy the chip. Be particularly careful about the PROM writer's overshoot characteristics.

If the PROM writer is set to Intel specifications or Hitachi HN27256 or HN27C256 specifications, V_{PP} will be 12.5V.

(2) Before writing data, check that the socket adapter and chip are correctly mounted in the PROM writer. Overcurrent damage to the chip can result if the index marks on the PROM writer, socket adapter, and chip are not correctly aligned.

(3) Don't touch the socket adapter or chip while writing. Touching either of these can cause contact faults and write errors.

13.3.3 Reliability of Written Data

An effective way to assure the data holding characteristics of the programmed chips is to bake them at 150˚C, then screen them for data errors. This procedure quickly eliminates chips with PROM memory cells prone to early failure.

Figure 13-6 shows the recommended screening procedure.

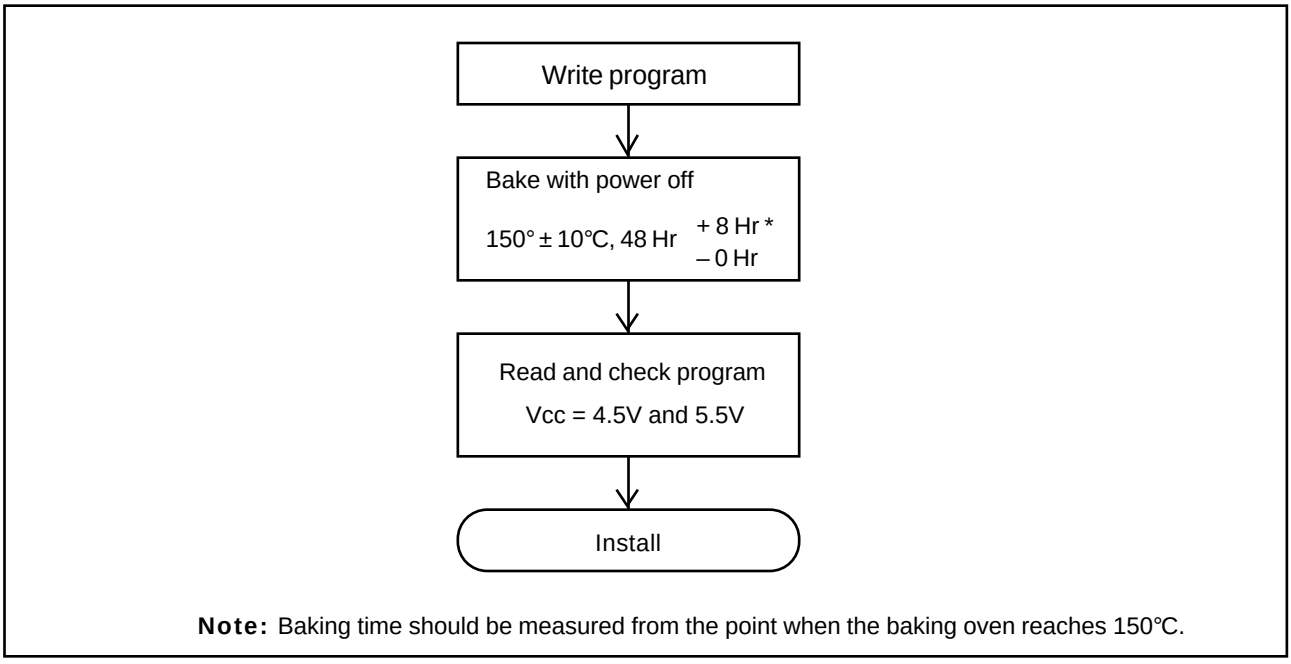


Figure 13-6. Recommended Screening Procedure

If a series of write errors occurs while the same PROM writer is in use, stop programming and check the PROM writer and socket adapter for defects, using a microcomputer chip with a windowed package and on-chip EPROM.

Please inform Hitachi of any abnormal conditions noted during programming or in screening of program data after high-temperature baking.

13.3.4 Erasing of Data

The windowed package enables data to be erased by illuminating the window with ultraviolet light. Table 13-7 lists the erasing conditions.

Table 13-7. Erasing Conditions

Item	Value
Ultraviolet wavelength	253.7 nm
Minimum illumination	15W·s/cm^2

The conditions in Table 13-7 can be satisfied by placing a 12000µW/cm^2 ultraviolet lamp 2 or 3 centimeters directly above the chip and leaving it on for about 20 minutes.

13.4 Handling of Windowed Packages

(1) Glass Erasing Window: Rubbing the glass erasing window of a windowed package with a plastic material or touching it with an electrically charged object can create a static charge on the window surface which may cause the chip to malfunction.

If the erasing window becomes charged, the charge can be neutralized by a short exposure to ultraviolet light. This returns the chip to its normal condition, but it also reduces the charge stored in the floating gates of the PROM, so it is recommended that the chip be reprogrammed afterward.

Accumulation of static charge on the window surface can be prevented by the following precautions:

(1) When handling the package, ground yourself. Don’t wear gloves. Avoid other possible sources of static charge.
(2) Avoid friction between the glass window and plastic or other materials that tend to accumulate static charge.

(3) Be careful when using cooling sprays, since they may have a slight ion content.
(4) Cover the window with an ultraviolet-shield label, preferably a label including a conductive material. Besides protecting the PROM contents from ultraviolet light, the label protects the chip by distributing static charge uniformly.

(2) Handling after Programming: Fluorescent light and sunlight contain small amounts of ultraviolet, so prolonged exposure to these types of light can cause programmed data to invert. In addition, exposure to any type of intense light can induce photoelectric effects that may lead to chip malfunction. It is recommended that after programming the chip, you cover the erasing window with a light-proof label (such as an ultraviolet-shield label).

(3) Note on 84-Pin LCC Package: A socket should always be used when the 84-pin LCC package is mounted on a printed-circuit board. Table 13.8 lists the recommended socket.

Table 13-8. Recommended Socket for Mounting 84-Pin LCC Package

Manufacturer	Code
Sumitomo 3-M	284-1273-00-1102J

Section 14. Power-Down State

14.1 Overview

The H8/330 has a power-down state that greatly reduces power consumption by stopping some or all of the chip functions. The power-down state includes three modes:

(1) Sleep mode – a software-triggered mode in which the CPU halts but the rest of the chip remains active
(2) Software standby mode – a software-triggered mode in which the entire chip is inactive
(3) Hardware standby mode – a hardware-triggered mode in which the entire chip is inactive

Table 14-1 lists the conditions for entering and leaving the power-down modes. It also indicates the status of the CPU, on-chip supporting modules, etc. in each power-down mode.

Table 14-1. Power-Down State

Mode	Entering procedure	Clock	CPU	CPU Reg's.	Sup. Mod.*	RAM	I/O ports	Exiting methods
Sleep mode	Execute SLEEP instruction	Run	Halt	Held	Run	Held	Held	• Interrupt • $\overline{\text{RES}}$ • $\overline{\text{STBY}}$
Soft-ware standby mode	Set SSBY bit in SYSCR to "1," then execute SLEEP instruction	Halt	Halt	Held	Halt and initial-ized	Held	Held	• $\overline{\text{NMI}}$ • $\overline{\text{IRQ}}_0$ – $\overline{\text{IRQ}}_2$ • $\overline{\text{STBY}}$ • $\overline{\text{RES}}$
Hard-ware standby mode	Set $\overline{\text{STBY}}$ pin to Low level	Halt	Halt	Not held	Halt and initialized	Held	High impe-dance state	• $\overline{\text{STBY}}$ High, then $\overline{\text{RES}}$ Low → High

Notes

1. SYSCR: System control register
2. SSBY: Software standby bit
3. On-chip supporting modules, including the dual-port RAM.

14.2 System Control Register: Power-Down Control Bits

Bits 7 to 4 of the system control register (SYSCR) concern the power-down state. Specifically, they concern the software standby mode.

Table 14-2 lists the attributes of the system control register.

Table 14-2. System Control Register

Name	Abbreviation	R/W	Initial value	Address
System control register	SYSCR	R/W	H’09	H’FFC4

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W

Bit 7 – Software Standby (SSBY): This bit enables or disables the transition to the software standby mode.

On recovery from the software standby mode by an external interrupt, SSBY remains set to "1." To clear this bit, software must write a "0."

Bit 7

SSBY	Description	
0	The SLEEP instruction causes a transition to the sleep mode.	(Initial value)
1	The SLEEP instruction causes a transition to the software standby mode.	

Bits 6 to 4 – Standby Timer Select 2 to 0 (STS2 to STS0): These bits select the clock settling time when the chip recovers from the software standby mode by an external interrupt. During the selected time, the clock oscillator runs but clock pulses are not supplied to the CPU or the on-chip supporting modules.

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8192 states	(Initial value)
0	0	1	Settling time = 16384 states	
0	1	0	Settling time = 32768 states	
0	1	1	Settling time = 65536 states	
1	—	—	Settling time = 131072 states	

When the H8/330's on-chip clock generator is used, the STS bits should be set to allow a settling time of at least 10ms. Table 14-3 lists the settling times selected by these bits at several clock frequencies and indicates the recommended settings.

When the H8/330 is externally clocked, the STS bits can be set to any value. The minimum value (STS2 = STS1 = STS0 = "0") is recommended.

Table 14-3. Times Set by Standby Timer Select Bits (Unit: ms)

			Settling time	System clock frequency (MHz)						
STS2	**STS1**	**STS0**	**(states)**	**10**	**8**	**6**	**4**	**2**	**1**	**0.5**
0	0	0	8192	0.8	1.0	1.3	2.0	4.1	8.2	**16.4**
0	0	1	16384	1.6	2.0	2.7	4.1	8.2	**16.4**	32.8
0	1	0	32768	3.3	4.1	5.5	8.2	**16.4**	32.8	65.5
0	1	1	65536	6.6	8.2	**10.9**	**16.4**	32.8	65.5	131.1
1	—	—	131072	**13.1**	**16.4**	21.8	32.8	65.5	131.1	262.1

Notes:

1. All times are in milliseconds.
2. Recommended values are printed in boldface.

14.3 Sleep Mode

The sleep mode provides an effective way to conserve power while the CPU is waiting for an external interrupt or an interrupt from an on-chip supporting module.

14.3.1 Transition to Sleep Mode

When the SSBY bit in the system control register is cleared to "0," execution of the SLEEP instruction causes a transition from the program execution state to the sleep mode. After executing the SLEEP instruction, the CPU halts, but the contents of its internal registers remain unchanged. The on-chip supporting modules continue to operate normally.

14.3.2 Exit from Sleep Mode

The chip wakes up from the sleep mode when it receives an internal or external interrupt request, or a Low input at the $\overline{\text{RES}}$ or $\overline{\text{STBY}}$ pin.

(1) Wake-Up by Interrupt: An interrupt releases the sleep mode and starts the CPU's interrupt-handling sequence.

If an interrupt from an on-chip supporting module is disabled by the corresponding enable/disable bit in the module's control register, the interrupt cannot be requested, so it cannot wake the chip up. Similarly, the CPU cannot be awoken by an interrupt other than NMI if the I (interrupt mask) bit in the CCR (condition code register) is set when the SLEEP instruction is executed.

(2) Wake-Up by $\overline{\text{RES}}$ pin: When the $\overline{\text{RES}}$ pin goes Low, the chip exits from the sleep mode to the reset state.

(3) Wake-Up by $\overline{\text{STBY}}$ pin: When the $\overline{\text{STBY}}$ pin goes Low, the chip exits from the sleep mode to the hardware standby mode.

14.4 Software Standby Mode

In the software standby mode, the system clock stops and chip functions halt, including both CPU functions and the functions of the on-chip supporting modules. Power consumption is reduced to an extremely low level. The on-chip supporting modules and their registers are reset to their initial states, but as long as a minimum necessary voltage supply is maintained (at least 2V), the contents of the CPU registers and on-chip RAM remain unchanged.

14.4.1 Transition to Software Standby Mode

To enter the software standby mode, set the standby bit (SSBY) in the system control register (SYSCR) to “1,” then execute the SLEEP instruction.

14.4.2 Exit from Software Standby Mode

The chip can be brought out of the software standby mode by an input at one of six pins: $\overline{\text{NMI}}$, $\overline{\text{IRQ}}_0$, $\overline{\text{IRQ}}_1$, $\overline{\text{IRQ}}_2$, $\overline{\text{RES}}$, or $\overline{\text{STBY}}$.

(1) Recovery by External Interrupt: When an $\overline{\text{NMI}}$, $\overline{\text{IRQ}}_0$, $\overline{\text{IRQ}}_1$, or $\overline{\text{IRQ}}_2$ request signal is received, the clock oscillator begins operating. After the waiting time set in the system control register (bits STS2 to STS0), clock pulses are supplied to the CPU and on-chip supporting modules. The CPU executes the interrupt-handling sequence for the requested interrupt, then returns to the instruction after the SLEEP instruction. The SSBY bit is not cleared.

See Section 14.2, “System Control Register: Power-Down Control Bits” for information about the STS bits.

Interrupts $\overline{\text{IRQ}}_3$ to $\overline{\text{IRQ}}_7$ should be disabled before entry to the software standby mode. Clear IRQ3E to IRQ7E to "0" in the interrupt enable register (IER).

(2) Recovery by $\overline{\text{RES}}$ Pin: When the $\overline{\text{RES}}$ pin goes Low, the clock oscillator starts. Next, when the $\overline{\text{RES}}$ pin goes High, the CPU begins executing the reset sequence. The SSBY bit is cleared to “0.”

The $\overline{\text{RES}}$ pin must be held Low long enough for the clock to stabilize.

(3) Recovery by $\overline{\text{STBY}}$ Pin: When the $\overline{\text{STBY}}$ pin goes Low, the chip exits from the software standby mode to the hardware standby mode.

14.4.3 Sample Application of Software Standby Mode

In this example the H8/330 enters the software standby mode when $\overline{\text{NMI}}$ goes Low and exits when $\overline{\text{NMI}}$ goes High, as shown in Figure 14-1.

The NMI edge bit (NMIEG) in the system control register is originally cleared to "0," selecting the falling edge. When $\overline{\text{NMI}}$ goes Low, the $\overline{\text{NMI}}$ interrupt handling routine sets NMIEG to "1," sets SSBY to "1" (selecting the rising edge), then executes the SLEEP instruction. The H8/330 enters the software standby mode. It recovers from the software standby mode on the next rising edge of $\overline{\text{NMI}}$.

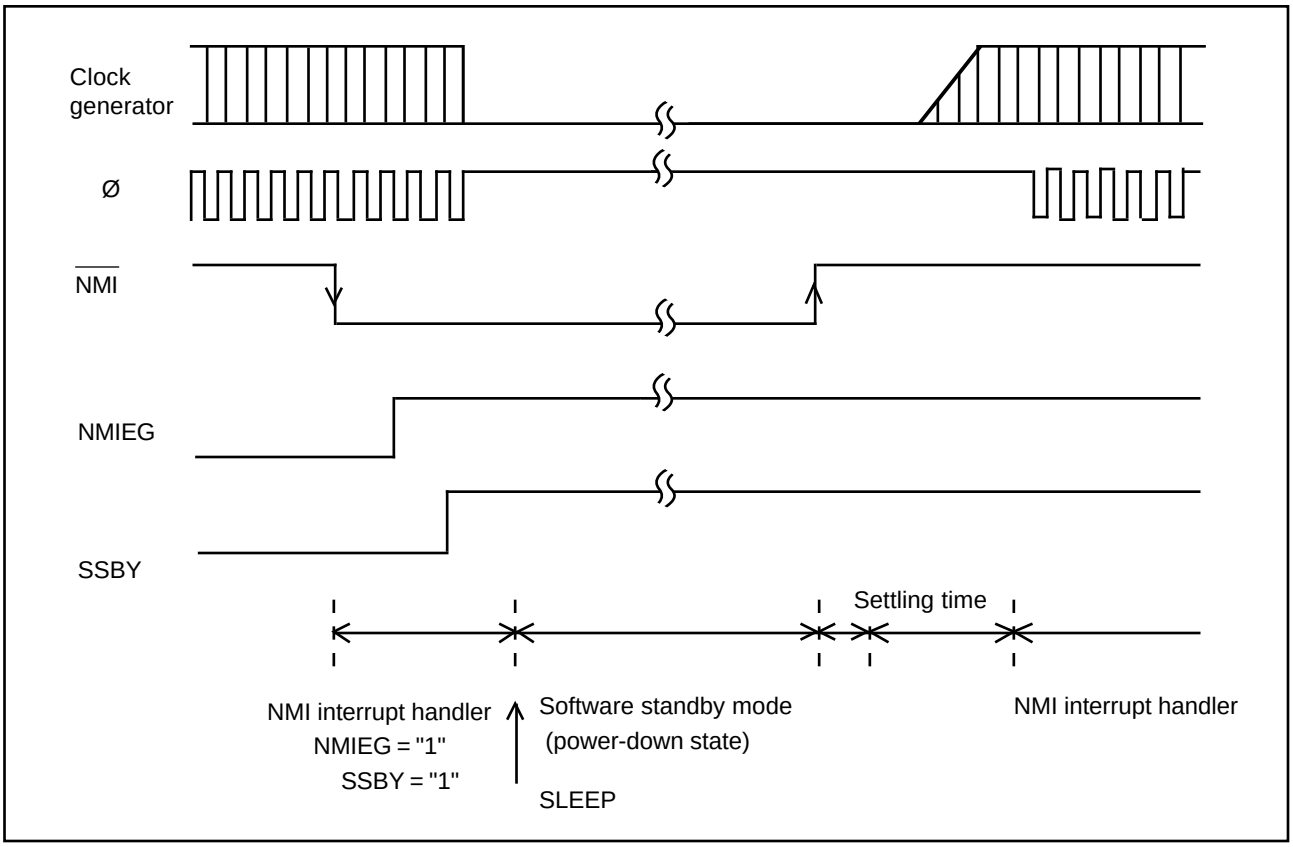


Figure 14-1. Software Standby Mode (when) NMI Timing

14.4.4 Application Notes

(1) The I/O ports retain their current states in the software standby mode. If a port is in the High output state, the current dissipation caused by the High output current is not reduced.

(2) If the software standby mode is entered under either condition (1) or condition (2) below, current dissipation is greater than in normal standby mode.

(1) In single-chip mode (mode 3): if software standby mode is entered by executing an instruction stored in on-chip ROM, after even one instruction not stored in on-chip ROM has been fetched (e.g. from on-chip RAM).

(2) In expanded mode with on-chip ROM enabled (mode 2): if software standby mode is entered by executing an instruction stored in on-chip ROM, after even one instruction not stored in on-chip ROM has been fetched (e.g. from external memory or on-chip RAM).

Note that the H8/300 CPU pre-fetches instructions. If an instruction stored in the last two bytes of on-chip ROM is executed (at addresses H'3FFE and H'3FFF in the H8/330), the contents of the next two bytes (H'4000 and H'4001), which are not in on-chip ROM, will be fetched as the next instruction.

This problem does not occur in expanded mode when on-chip ROM is disabled (mode 1). In hardware standby mode there is no such additional current dissipation, regardless of the conditions when hardware standby mode is entered.

14.5 Hardware Standby Mode

14.5.1 Transition to Hardware Standby Mode

Regardless of its current state, the chip enters the hardware standby mode whenever the $\overline{\text{STBY}}$ pin goes Low.

The hardware standby mode reduces power consumption drastically by halting the CPU, stopping all the functions of the on-chip supporting modules, and placing I/O ports in the high-impedance state. The registers of the on-chip supporting modules are reset to their initial values. Only the on-chip RAM is held unchanged, provided the minimum necessary voltage supply is maintained (at least 2V).

Notes:

1. The RAME bit in the system control register should be cleared to “0” before the $\overline{\text{STBY}}$ pin goes Low, to disable the on-chip RAM during the hardware standby mode.

2. Do not change the inputs at the mode pins (MD1, MD0) during hardware standby mode. Be particularly careful not to let both mode pins go Low in hardware standby mode, since that places the chip in PROM mode and increases current dissipation.

14.5.2 Recovery from Hardware Standby Mode

Recovery from the hardware standby mode requires inputs at both the $\overline{\text{STBY}}$ and $\overline{\text{RES}}$ pins.

When the $\overline{\text{STBY}}$ pin goes High, the clock oscillator begins running. The $\overline{\text{RES}}$ pin should be Low at this time and should be held Low long enough for the clock to stabilize. When the $\overline{\text{RES}}$ pin changes from Low to High, the reset sequence is executed and the chip returns to the program execution state.

14.5.3 Timing Relationships

Figure 14-2 shows the timing relationships in the hardware standby mode.
In the sequence shown, first $\overline{\text{RES}}$ goes Low, then $\overline{\text{STBY}}$ goes Low, at which point the H8/330 enters the hardware standby mode. To recover, first $\overline{\text{STBY}}$ goes High, then after the clock settling time, $\overline{\text{RES}}$ goes High.

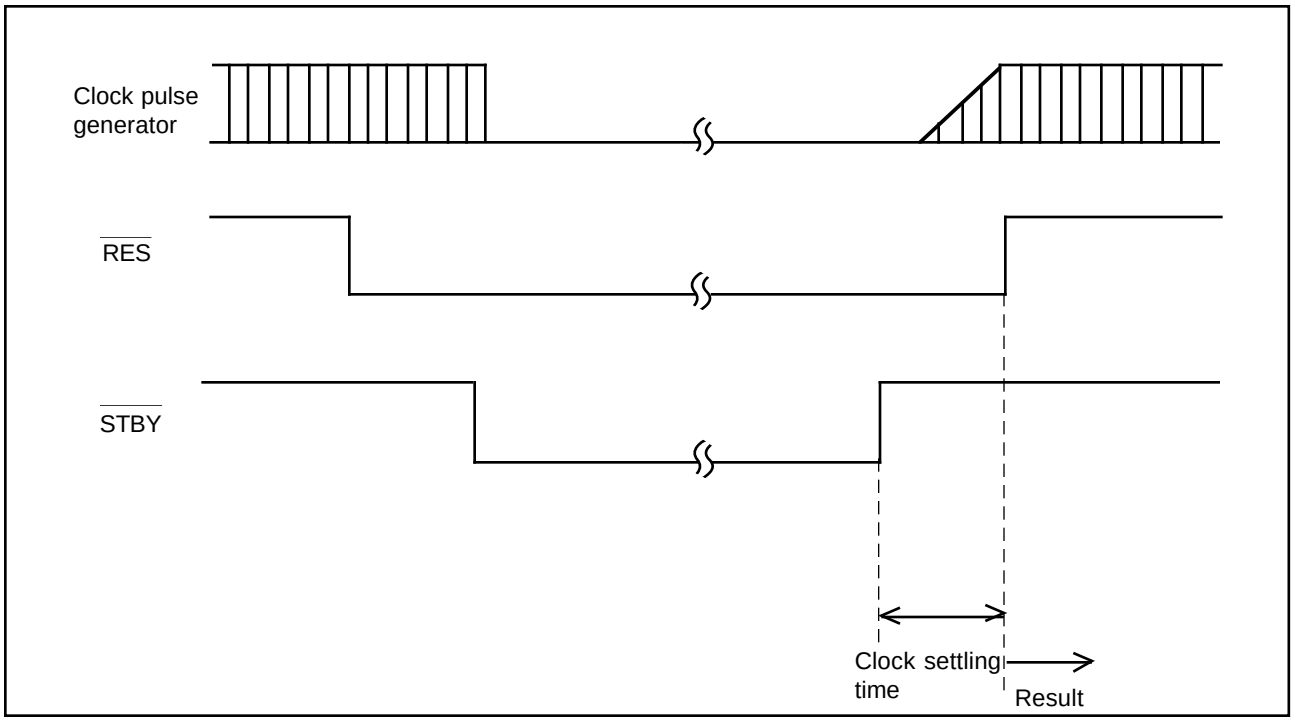


Figure 14-2. Hardware Standby Mode Timing

Section 15. E-Clock Interface

15.1 Overview

For interfacing to peripheral devices that require it, the H8/330 can generate an E clock output. Special instructions (MOVTPE, MOVFPE) perform data transfers synchronized with the E clock.

The E clock is created by dividing the system clock (Ø) by 8. The E clock is output at the $P8_0$ pin when the $P8_0$DDR bit in the port 8 data direction register (P8DDR) is set to "1." It is output only in the expanded modes (mode 1 and mode 2); it is not output in the single-chip mode. Output begins immediately after a reset.

When the CPU executes an instruction that synchronizes with the E clock, the address strobe ($\overline{AS}$), the address on the address bus, and the $\overline{IOS}$ signal are output as usual, but the $\overline{RD}$ and $\overline{WR}$ signal lines and the data bus do not become active until the falling edge of the E clock is detected. The length of the access cycle for an instruction synchronized with the E clock accordingly varies from 9 to 16 states. Figures 15-1 and 15-2 show the timing in the cases of maximum and minimum synchronization delay.

It is not possible to insert wait states (T_W) during the execution of an instruction synchronized with the E clock by input at the $\overline{WAIT}$ pin.

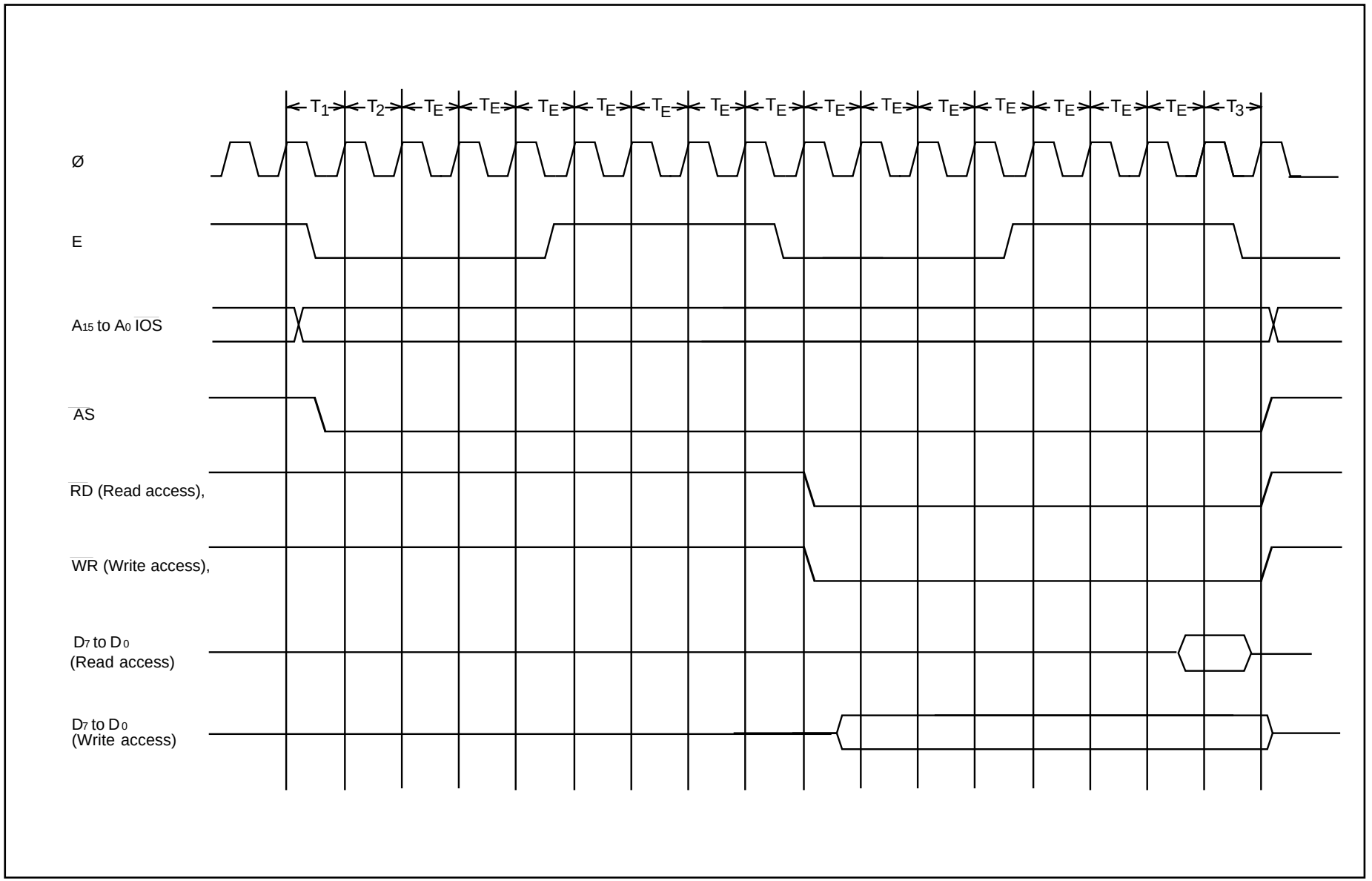


Figure 15-1. Execution Cycle of Instruction Synchronized with E Clock in Expanded Modes (Maximum Synchronization Delay)

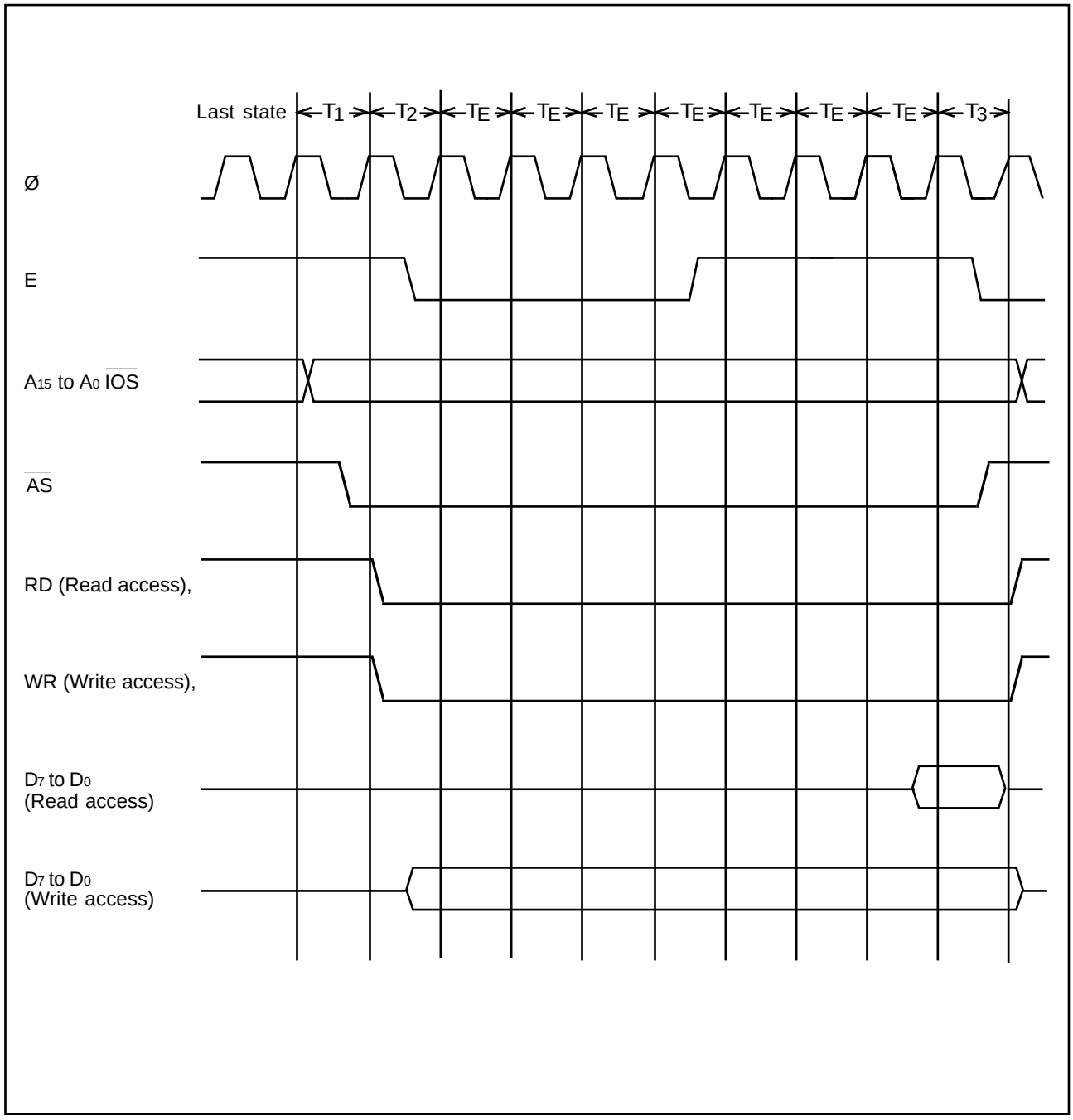


Figure 15-2. Execution Cycle of Instruction Synchronized with E Clock in Expanded Modes (Minimum Synchronization Delay)

Section 16. Clock Pulse Generator

16.1 Overview

The H8/330 chip has a built-in clock pulse generator (CPG) consisting of an oscillator circuit, a system (Ø) clock divider, an E clock divider, and a prescaler. The prescaler generates clock signals for the on-chip supporting modules.

16.1.1 Block Diagram

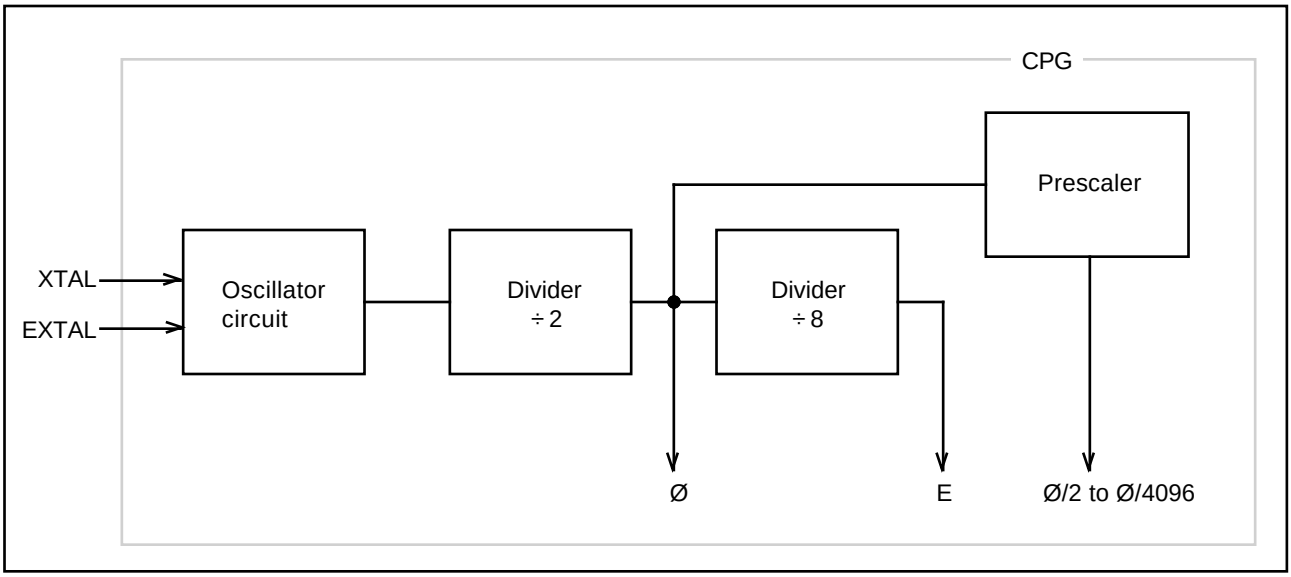


Figure 16-1. Block Diagram of Clock Pulse Generator

16.2 Oscillator Circuit

If an external crystal is connected across the EXTAL and XTAL pins, the on-chip oscillator circuit generates a clock signal for the system clock divider. Alternatively, an external clock signal can be applied to the EXTAL pin.

(1) Connecting an External Crystal

(1) Circuit Configuration: An external crystal can be connected as in the example in Figure 16-2. An AT-cut parallel resonating crystal should be used.

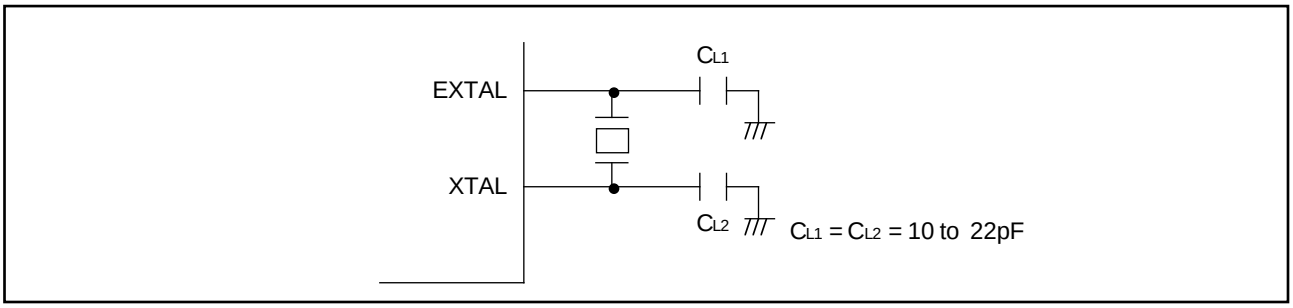


Figure 16-2. Connection of Crystal Oscillator (Example)

(2) Crystal Oscillator: The external crystal should have the characteristics listed in Table 16-1.

Table 16-1. External Crystal Parameters

Frequency (MHz)	2	4	8	12	16	20
Rs max (Ω)	500	120	60	40	30	20
C_0 (pF)			7 pF max			

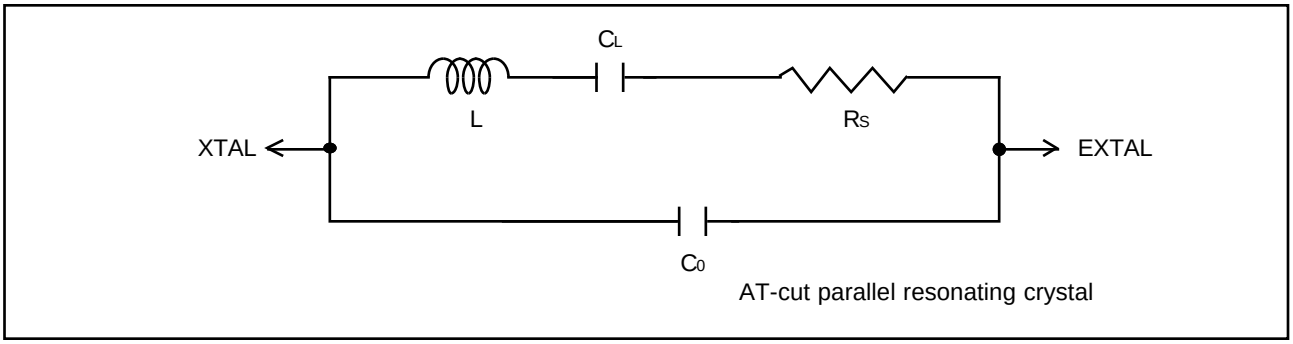


Figure 16-3. Equivalent Circuit of External Crystal

(3) Note on Board Design: When an external crystal is connected, other signal lines should be kept away from the crystal circuit to prevent induction from interfering with correct oscillation. See Figure 16-4. The crystal and its load capacitors should be placed as close as possible to the XTAL and EXTAL pins.

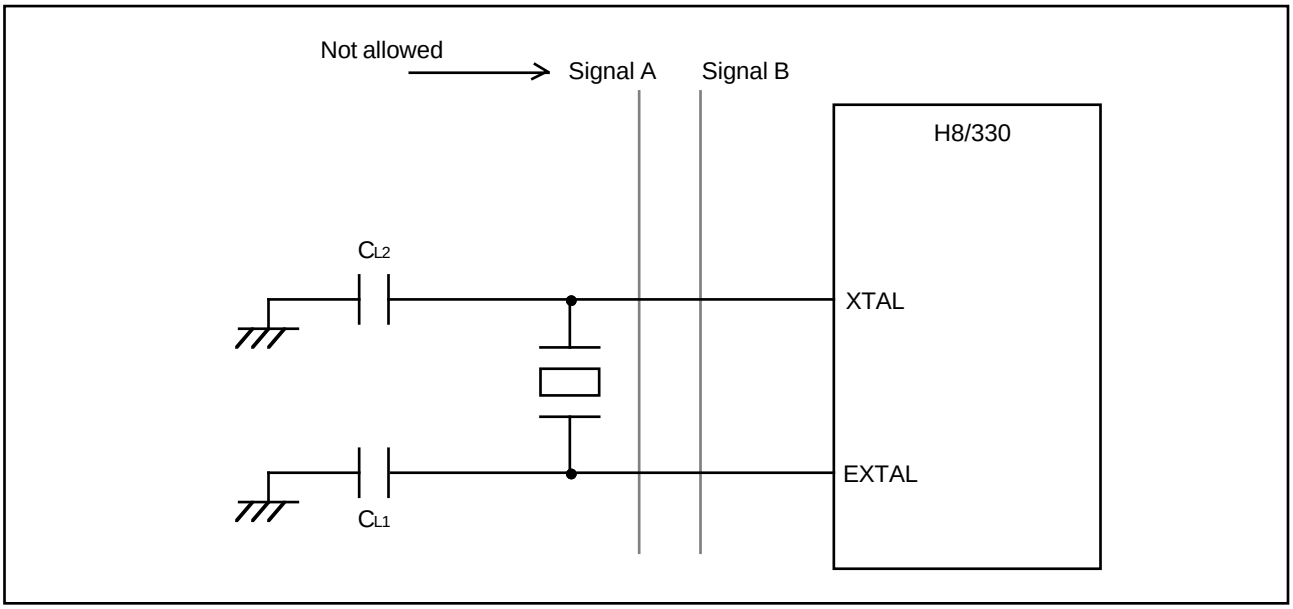


Figure 16-4. Notes on Board Design around External Crystal

(2) Input of External Clock Signal

(1) Circuit Configuration: An external clock signal can be input at the EXTAL pin. The reverse-phase clock signal should be input at the XTAL pin, as shown in the example in Figure 16-5.

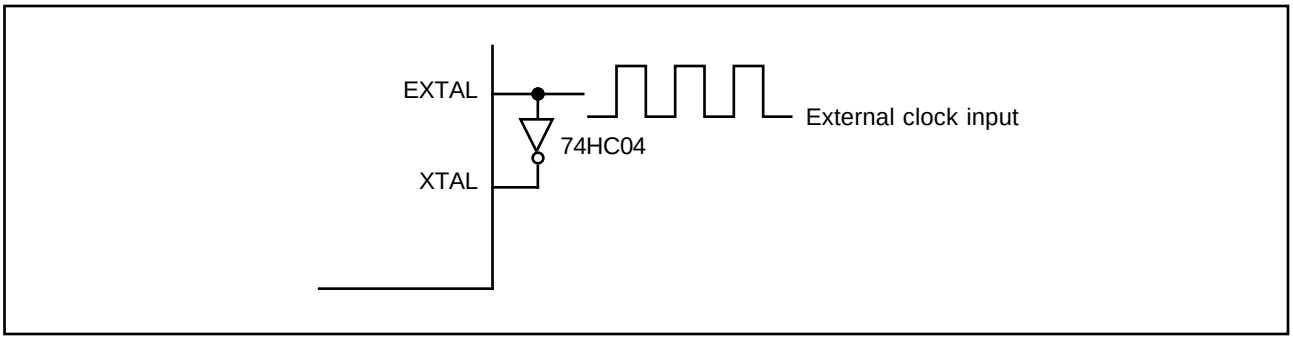


Figure 16-5. External Clock Input (Example)

(2) External Clock Input

Frequency	Double the system clock (Ø) frequency
Duty factor	45% to 55%

16.3 System Clock Divider

The system clock divider divides the crystal oscillator or external clock frequency by 2 to create the system clock (Ø).

An E clock signal is created by dividing the system clock by 8.

Figure 16-6 shows the phase relationship of the E clock to the system clock.

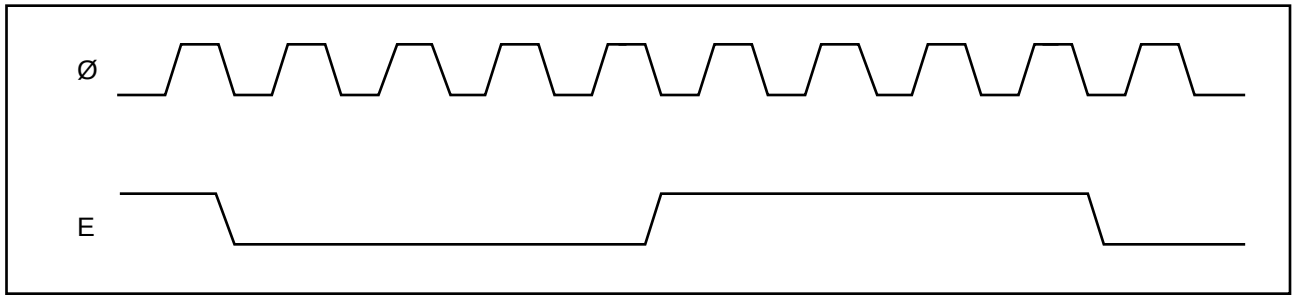


Figure 16-6. Phase Relationship of System Clock and E Clock

Section 17. Electrical Specifications

17.1 Absolute Maximum Ratings

Table 17-1 lists the absolute maximum ratings.

Table 17-1. Absolute Maximum Ratings

Item	Symbol	Rating	Unit
Supply voltage	V_{CC}	–0.3 to +7.0	V
Programming voltage	V_{PP}	–0.3 to +13.5	V
Input voltage			
Ports 1 – 6, 8, 9	V_{in}	–0.3 to V_{CC} + 0.3	V
Port 7	V_{in}	–0.3 to AV_{CC} + 0.3	V
Analog supply voltage	AV_{CC}	–0.3 to +7.0	V
Analog input voltage	V_{AN}	–0.3 to AV_{CC} + 0.3	V
Operating temperature	T_{opr}	Regular specifications: –20 to +75	˚C
		Wide-range specifications: – 40 to +85	˚C
Storage temperature	T_{stg}	–55 to +125	˚C

Note: The input pins have protection circuits that guard against high static voltages and electric fields, but these high input-impedance circuits should never receive overvoltages exceeding the absolute maximum ratings shown in table 17-1.

17.2 Electrical Characteristics

17.2.1 DC Characteristics

Table 17-2 lists the DC characteristics of the H8/330.

Table 17-2. DC Characteristics

Conditions: V_{CC} = 5.0V ±10%*, AV_{CC} = 5.0V ±10%, V_{SS} = AV_{SS} = 0V,
Ta = –20 to 75˚C (regular specifications), Ta = –40 to 85˚C (wide-range specifications)

Item		Symbol	min	typ	max	Unit	Measurement conditions
Schmitt trigger input voltage (1)	$P6_7$ – $P6_2$, $P6_0$, $P8_6$ – $P8_0$, $P9_7$, $P9_4$ – $P9_0$	V_T^-	1.0	–	–	V	
		V_T^+	–	–	3.5	V	
		$V_T^+ - V_T^-$	0.4	–	–	V	
Input High voltage (2)	$\overline{RES}$, $\overline{STBY}$, MD_1, MD_0	V_{IH}	V_{CC} – 0.7	–	V_{CC} + 0.3	V	
	EXTAL, $\overline{NMI}$		$V_{CC} \times 0.7$	–	V_{CC} + 0.3	V	
	$P7_7$ – $P7_0$		2.2	–	AV_{CC} + 0.3	V	
Input High voltage	Input pins other than (1) and (2)	V_{IH}	2.2	–	V_{CC} + 0.3	V	
Input Low voltage (3)	$\overline{RES}$, $\overline{STBY}$ MD_1, MD_0	V_{IL}	–0.3	–	0.5	V	
Input Low voltage	Input pins other than (1) and (3)	V_{IL}	–0.3	–	0.8	V	
Output High voltage	All output pins	V_{OH}	V_{CC} – 0.5	–	–	V	I_{OH} = –200µA
			3.5	–	–	V	I_{OH} = –1.0mA
Output Low voltage	All output pins	V_{OL}	–	–	0.4	V	I_{OL} = 1.6mA
	Ports 1 and 2		–	–	1.0	V	I_{OL} = 10.0mA
Input leakage current	$\overline{RES}$	$\|I_{in}\|$	–	–	10.0	µA	V_{in} = 0.5V to V_{CC} – 0.5V
	$\overline{STBY}$, $\overline{NMI}$, MD_1, MD_0		–	–	1.0	µA	
	$P7_7$ – $P7_0$		–	–	1.0	µA	V_{in} = 0.5V to AV_{CC} – 0.5V
Leakage current in 3-state (off state)	Ports 1, 2, 3 4, 5, 6, 8, 9	$\|I_{TSI}\|$	–	–	1.0	µA	V_{in} = 0.5V to V_{CC} – 0.5V
Input pull-up MOS current	Ports 1, 2, 3 4, 5, 6, 8, 9	-Ip	30	–	250	µA	V_{in} = 0V

* Connect AV_{CC} to the power supply (+5V) even when the A/D converter is not used.

Table 17-2. DC Characteristics (cont.)

Conditions: V_{CC} = AV_{CC} = 5.0V ±10%, V_{SS} = AV_{SS} = 0V, Ta = –20 to 75˚C (regular specifications)
Ta = –40 to 85˚C (wide-range specifications)

Item		Symbol	min	typ	max	Unit	Measurement conditions
Input capacitance	$\overline{RES}$ (V_{PP})	C_{in}	–	–	60	pF	V_{in} = 0V
	$\overline{NMI}$		–	–	30	pF	
	All input pins except $\overline{RES}$ and $\overline{NMI}$		–	–	15	pF	f = 1MHz Ta = 25˚C
Current dissipation*1	Normal operation	I_{CC}	–	12	25	*m*A	f = 6MHz
			–	16	30	*m*A	f = 8MHz
			–	20	40	*m*A	f = 10MHz
	Sleep mode		–	8	15	*m*A	f = 6MHz
			–	10	20	*m*A	f = 8MHz
			–	12	25	*m*A	f = 10MHz
	Standby modes*2		–	0.01	5.0	µA	
Analog supply current	During A/D conversion	AI_{CC}	–	0.6	1.5	*m*A	
	Waiting		–	0.01	5.0	µA	
RAM standby voltage		V_{RAM}	2.0	–	–	V	

*1 Current dissipation values assume that $V_{IH\ min.}$ = V_{CC} – 0.5V, $V_{IL\ max.}$ = 0.5V, all output pins are in the no-load state, and all MOS input pull-ups are off.

*2 For these values it is assumed that $V_{RAM} \leq V_{CC} < 4.5V$ and $V_{IH\ min.} = V_{CC} \times 0.9$, $V_{IL\ max.}$ = 0.3V.

Table 17-3. Allowable Output Current Sink Values

Conditions: V_{CC} = AV_{CC} = 5.0V ±10%, V_{SS} = AV_{SS} = 0V, Ta = –20 to 75˚C (regular specifications)
Ta = –40 to 85˚C (wide-range specifications)

Item		Symbol	min	typ	max	Unit
Allowable output Low current sink (per pin)	Ports 1 and 2	I_{OL}	–	–	10	mA
	Other output pins		–	–	2.0	mA
Allowable output Low current sink (total)	Ports 1 and 2, total	ΣI_{OL}	–	–	80	mA
	All output pins		–	–	120	mA
Allowable output High current sink (per pin)	All output pins	$-I_{OH}$	–	–	2.0	mA
Allowable output High current sink (total)	Total of all output	$\Sigma -I_{OH}$	–	–	40	mA

Note: To avoid degrading the reliability of the chip, be careful not to exceed the output current sink values in table 17-3. In particular, when driving a Darlington transistor pair or LED directly, be sure to insert a current-limiting resistor in the output path. See figures 17-1 and 17-2.

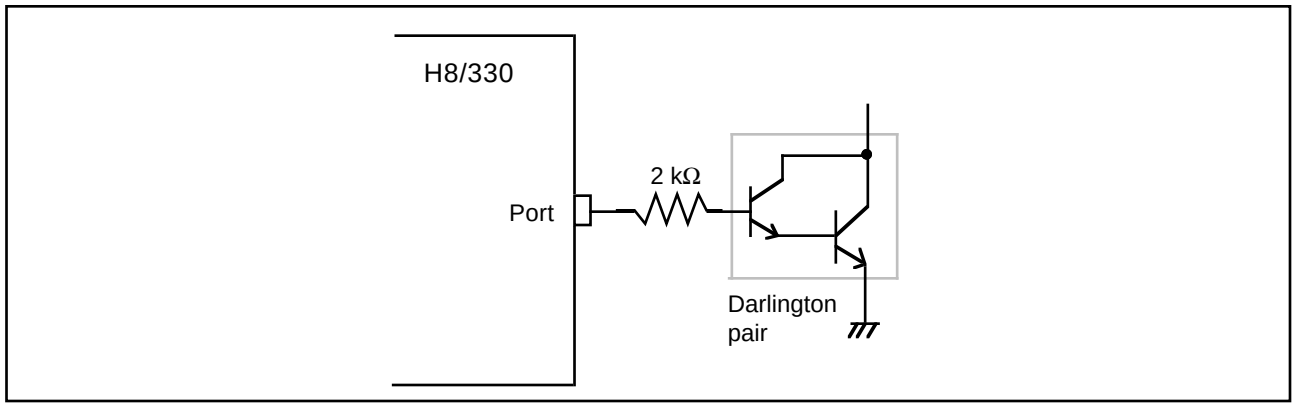


Figure 17-1. Example of Circuit for Driving a Darlington Pair

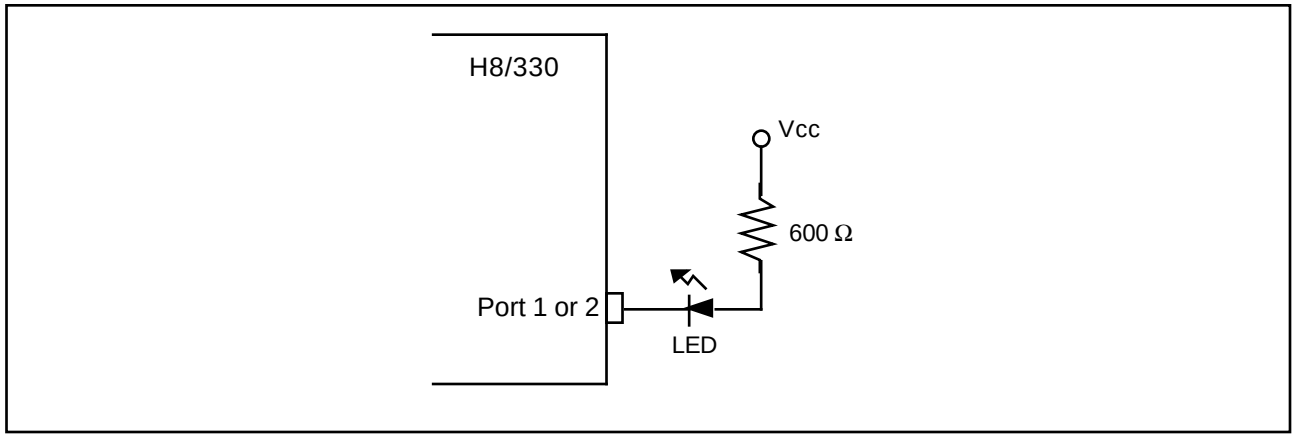


Figure 17-2. Example of Circuit for Driving a LED

17.2.2 AC Characteristics

The AC characteristics of the H8/330 chip are listed in three tables. Bus timing parameters are given in table 17-4, control signal timing parameters in table 17-5, and timing parameters of the on-chip supporting modules in table 17-6.

Table 17-4. Bus Timing

Conditions: V_{CC} = 5.0V ±10%, Ø = 0.5 to 10MHz, V_{SS} = 0V,
Ta = –20 to 75˚C (regular specifications), Ta = –40 to 85˚C (wide-range specifications)

Measurement Item	Symbol	Measurement conditions	6MHz min	6MHz max	8MHz min	8MHz max	10MHz min	10MHz max	Unit
Clock cycle time	t_{cyc}	Fig. 17-4	166.7	2000	125	2000	100	2000	ns
Clock pulse width Low	t_{CL}	Fig. 17-4	65	–	45	–	35	–	ns
Clock pulse width High	t_{CH}	Fig. 17-4	65	–	45	–	35	–	ns
Clock rise time	t_{Cr}	Fig. 17-4	–	15	–	15	–	15	ns
Clock fall time	t_{Cf}	Fig. 17-4	–	15	–	15	–	15	ns
Address delay time	t_{AD}	Fig. 17-4	–	70	–	60	–	55	ns
Address hold time	t_{AH}	Fig. 17-4	30	–	25	–	20	–	ns
Address strobe delay time	t_{ASD}	Fig. 17-4	–	70	–	60	–	50	ns
Write strobe delay time	t_{WSD}	Fig. 17-4	–	70	–	60	–	50	ns
Strobe delay time	t_{SD}	Fig. 17-4	–	70	–	60	–	50	ns
Write strobe pulse width	t_{WSW}	Fig. 17-4	200	–	150	–	120	–	ns
Address setup time 1	t_{AS1}	Fig. 17-4	25	–	20	–	15	–	ns
Address setup time 2	t_{AS2}	Fig. 17-4	105	–	80	–	65	–	ns
Read data setup time	t_{RDS}	Fig. 17-4	70	–	55	–	50	–	ns
Read data hold time	t_{RDH}	Fig. 17-4	0	–	0	–	0	–	ns
Write data delay time	t_{WDD}	Fig. 17-4	–	70	–	60	–	60	ns
Read data access time	t_{ACC}	Fig. 17-4	–	270	–	190	–	160	ns
Write data setup time	t_{WDS}	Fig. 17-4	30	–	15	–	10	–	ns
Write data hold time	t_{WDH}	Fig. 17-4	30	–	25	–	20	–	ns
Wait setup time	t_{WTS}	Fig. 17-5	40	–	40	–	40	–	ns
Wait hold time	t_{WTH}	Fig. 17-5	10	–	10	–	10	–	ns
E clock delay time	t_{ED}	Fig. 17-6	–	20	–	20	–	20	ns
E clock rise time	t_{Er}	Fig. 17-6	–	15	–	15	–	15	ns
E clock fall time	t_{Ef}	Fig. 17-6	–	15	–	15	–	15	ns
Read data hold time (for E clock)	t_{RDHE}	Fig. 17-6	0	–	0	–	0	–	ns
Write data hold time (for E clock)	t_{WDHE}	Fig. 17-6	50	–	40	–	30	–	ns

Table 17-5. Control Signal Timing

Conditions: V_{CC} = 5.0V ±10%, Ø = 0.5 to 10MHz, V_{SS} = 0V,
Ta = –20 to 75˚C (regular specifications), Ta = –40 to 85˚C (wide-range specifications)

Measurement Item	Symbol	Measurement conditions	6MHz min	6MHz max	8MHz min	8MHz max	10MHz min	10MHz max	Unit
$\overline{RES}$ setup time	t_{RESS}	Fig. 17-7	200	–	200	–	200	–	ns
$\overline{RES}$ pulse width	t_{RESW}	Fig. 17-7	10	–	10	–	10	–	t_{cyc}
Mode programming setup time	t_{MDS}	Fig. 17-7	4	–	4	–	4	–	t_{cyc}
$\overline{NMI}$ setup time ($\overline{NMI}$, $\overline{IRQ_0}$ to $\overline{IRQ_7}$)	t_{NMIS}	Fig. 17-8	110	–	110	–	110	–	ns
NMI hold time ($\overline{NMI}$, $\overline{IRQ_0}$ to $\overline{IRQ_7}$)	t_{NMIH}	Fig. 17-8	10	–	10	–	10	–	ns
Interrupt pulse width for recovery from software standby mode ($\overline{NMI}$, $\overline{IRQ_0}$ to $\overline{IRQ_2}$)	t_{NMIW}	Fig. 17-8	200	–	200	–	200	–	ns
Crystal oscillator settling time (reset)	t_{OSC1}	Fig. 17-9	20	–	20	–	20	–	ms
Crystal oscillator settling time (software standby)	t_{OSC2}	Fig. 17-10	10	–	10	–	10	–	ms

Table 17-6. Timing Conditions of On-Chip Supporting Modules

Conditions: V_{CC} = 5.0V ±10%, Ø = 0.5 to 10MHz, V_{SS} = 0V,
Ta = –20 to 75˚C (regular specifications), Ta = –40 to 85˚C (wide-range specifications)

	Item	Symbol	Measurement conditions	6MHz min	6MHz max	8MHz min	8MHz max	10MHz min	10MHz max	Unit
FRT	Timer output delay time	t_{FTOD}	Fig. 17-11	–	100	–	100	–	100	ns
	Timer input setup time	t_{FTIS}	Fig. 17-11	50	–	50	–	50	–	ns
	Timer clock input setup time	t_{FTCS}	Fig. 17-12	50	–	50	–	50	–	ns
	Timer clock pulse width	t_{FTCWH} t_{FTCWL}	Fig. 17-12	1.5	–	1.5	–	1.5	–	t_{cyc}

Table 17-6. Timing Conditions of On-Chip Supporting Modules (cont.)

Conditions: V_{CC} = 5.0V ±10%, Ø = 0.5 to 10MHz, V_{SS} = 0V,
Ta = –20 to 75˚C (regular specifications), Ta = –40 to 85˚C (wide-range specifications)

	Item	Symbol	Measurement conditions	6MHz min	6MHz max	8MHz min	8MHz max	10MHz min	10MHz max	Unit
TMR	Timer output delay time	t_{TMOD}	Fig. 17-13	–	100	–	100	–	100	ns
	Timer reset input setup time	t_{TMRS}	Fig. 17-15	50	–	50	–	50	–	ns
	Timer clock input setup time	t_{TMCS}	Fig. 17-14	50	–	50	–	50	–	ns
	Timer clock pulse width (single edge)	t_{TMCWH}	Fig. 17-14	1.5	–	1.5	–	1.5	–	t_{cyc}
	Timer clock pulse width (both edges)	t_{TMCWL}	Fig. 17-14	2.5	–	2.5	–	2.5	–	t_{cyc}
PWM	Timer output delay time	t_{PWOD}	Fig. 17-16	–	100	–	100	–	100	ns
SCI	Input clock cycle (Async)	t_{scyc}	Fig. 17-17	2	–	2	–	2	–	t_{cyc}
	(Sync)	t_{scyc}	Fig. 17-17	4	–	4	–	4	–	t_{cyc}
	Transmit data delay time (Sync)	t_{TXD}	Fig. 17-17	–	100	–	100	–	100	ns
	Receive data setup time (Sync)	t_{RXS}	Fig. 17-17	100	–	100	–	100	–	ns
	Receive data hold time (Sync)	t_{RXH}	Fig. 17-17	100	–	100	–	100	–	ns
	Input clock pulse width	t_{SCKW}	Fig. 17-18	0.4	0.6	0.4	0.6	0.4	0.6	t_{scyc}
Ports	Output data delay time	t_{PWD}	Fig. 17-19	–	100	–	100	–	100	ns
	Input data setup time	t_{PRS}	Fig. 17-19	50	–	50	–	50	–	ns
	Input data hold time	t_{PRH}	Fig. 17-19	50	–	50	–	50	–	ns

Table 17-6. Timing Conditions of On-Chip Supporting Modules (cont.)

Conditions: V_{CC} = 5.0V ±10%, Ø = 0.5 to 10MHz, V_{SS} = 0V,
Ta = –20 to 75˚C (regular specifications), Ta = –40 to 85˚C (wide-range specifications)

	Item	Symbol	Measurement conditions	6MHz min	6MHz max	8MHz min	8MHz max	10MHz min	10MHz max	Unit
Dual-port RAM read cycle	Address access time	t_{DAA}	Fig. 17-20	–	150	–	150	–	150	ns
	$\overline{CS}$ access time	t_{DACS}	Fig. 17-20	–	130	–	130	–	130	ns
	$\overline{OE}$ output delay time	t_{DOE}	Fig. 17-20	–	70	–	70	–	70	ns
	$\overline{CS}$ output floating time	t_{DCHZ}	Fig. 17-20	–	50	–	50	–	50	ns
	$\overline{OE}$ output floating time	t_{DOHZ}	Fig. 17-20	–	50	–	50	–	50	ns
	Output data hold time	t_{DOH}	Fig. 17-20	0	–	0	–	0	–	ns
Dual-port RAM write cycle	Chip select time	t_{DCW}	Fig. 17-21	100	–	100	–	100	–	ns
	Address valid time	t_{DAW}	Fig. 17-21	100	–	100	–	100	–	ns
	Address setup time	t_{DAS}	Fig. 17-21	20	–	20	–	20	–	ns
	Write pulse width	t_{DWP}	Fig. 17-21	90	–	90	–	90	–	ns
	Address hold time	t_{DWR}	Fig. 17-21	20	–	20	–	20	–	ns
	Input data setup time	t_{DDW}	Fig. 17-21	60	–	60	–	60	–	ns
	Input data hold time	t_{DDH}	Fig. 17-21	15	–	15	–	15	–	ns

• **Measurement Conditions for AC Characteristics**

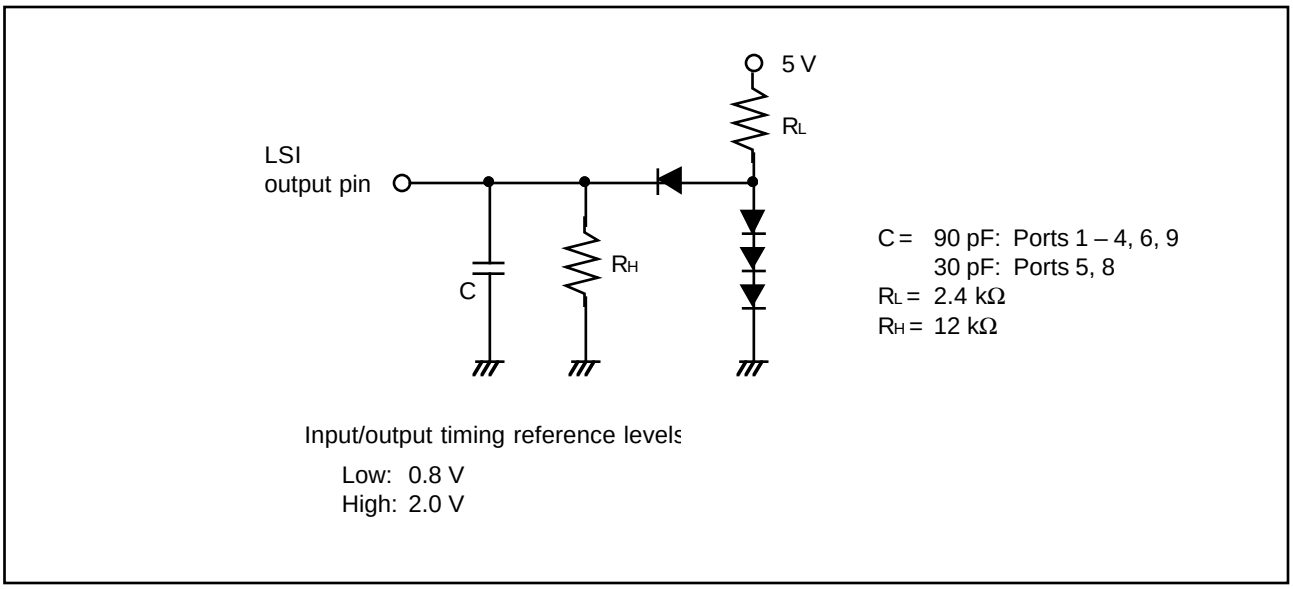


Figure 17-3. Output Load Circuit

17.2.3 A/D Converter Characteristics

Table 17-7 lists the characteristics of the on-chip A/D converter.

Table 17-7. A/D Converter Characteristics

Conditions: V_{CC} = AV_{CC} = 5.0V ±10%, V_{SS} = AV_{SS} = 0V, Ta = –20 to 75˚C (regular specifications)
Ta = –40 to 85˚C (wide-range specifications)

	6MHz			8MHz			10MHz			
Item	**min**	**typ**	**max**	**min**	**typ**	**max**	**min**	**typ**	**max**	**Unit**
Resolution	8	8	8	8	8	8	8	8	8	Bits
Conversion time (single mode)	—	—	20.4	—	—	15.25	—	—	12.2	µs
Analog input capacitance	—	—	20	—	—	20	—	—	20	pF
Allowable signal source impedance	—	—	10	—	—	10	—	—	10	kΩ
Nonlinearity error	—	—	±1	—	—	±1	—	—	±1	LSB
Offset error	—	—	±1	—	—	±1	—	—	±1	LSB
Full-scale error	—	—	±1	—	—	±1	—	—	±1	LSB
Quantizing error	—	—	±0.5	—	—	±0.5	—	—	±0.5	LSB
Absolute accuracy	—	—	±1.5	—	—	±1.5	—	—	±1.5	LSB

17.3 MCU Operational Timing

This section provides the following timing charts:

17.3.1 Bus Timing

(1) Basic Bus Cycle (Without Wait States) in Expanded Modes

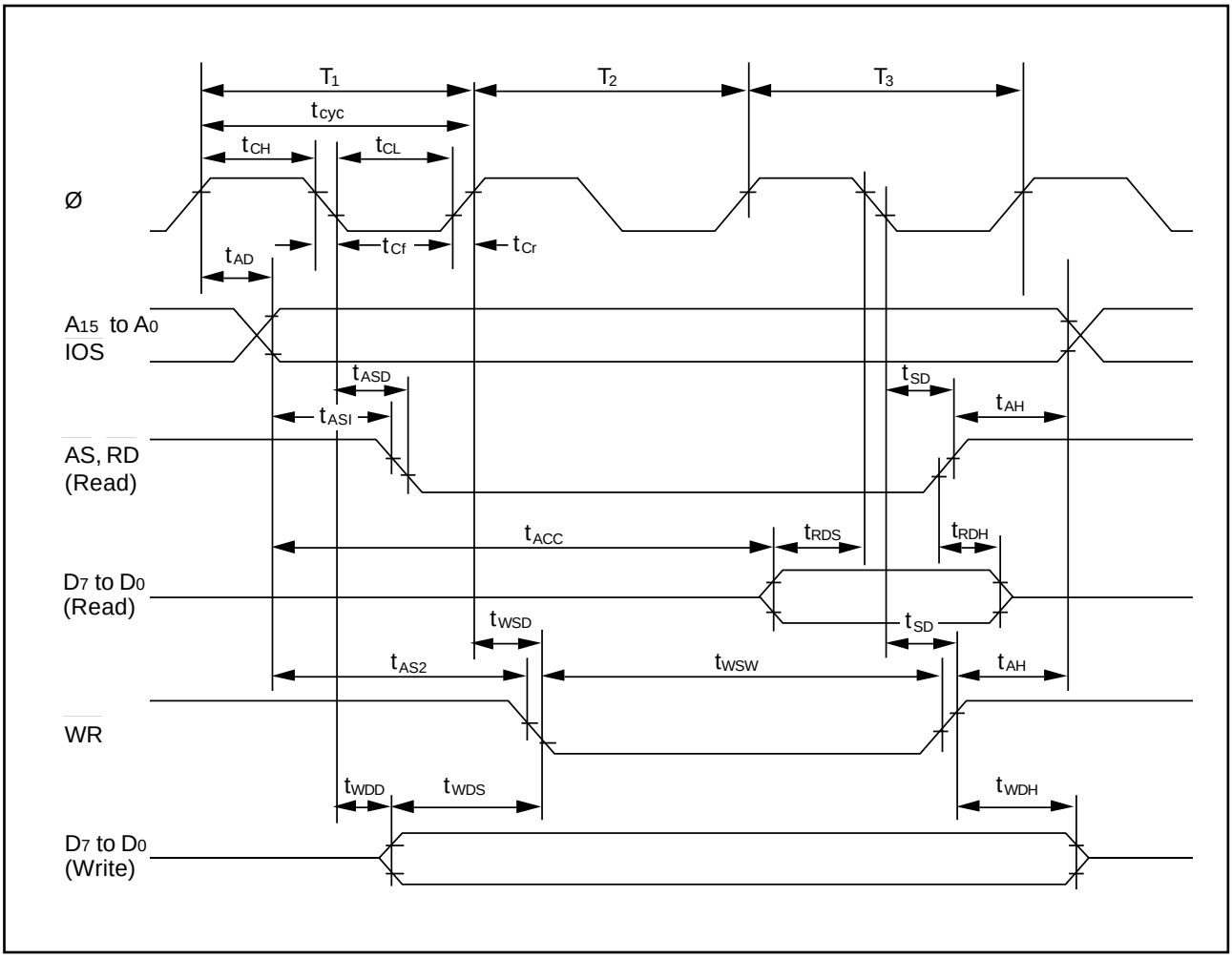


Figure 17-4. Basic Bus Cycle (Without Wait States) in Expanded Modes

(2) Basic Bus Cycle (With 1 Wait State) in Expanded Modes

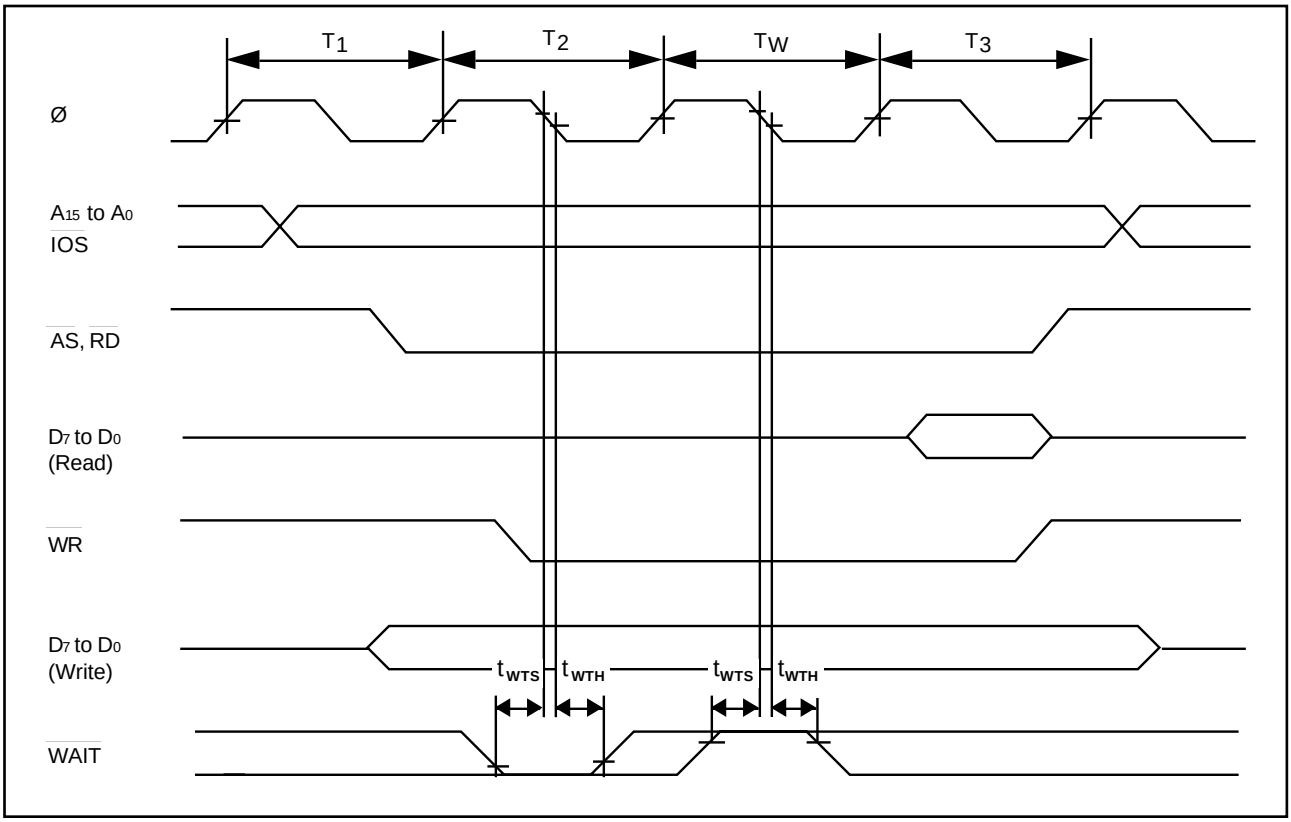


Figure 17-5. Basic Bus Cycle (With 1 Wait State) in Expanded Modes

(3) E Clock Bus Cycle

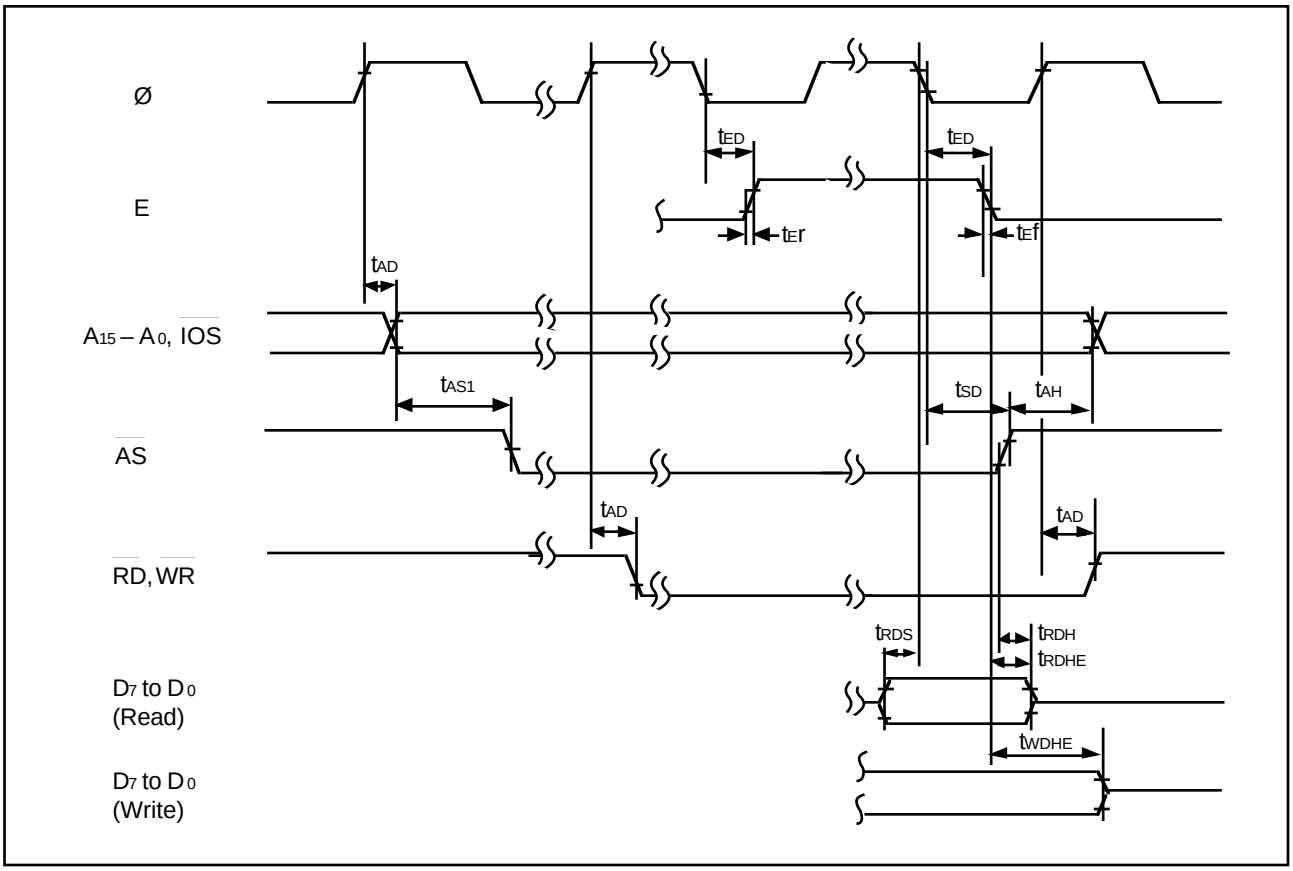


Figure 17-6. E Clock Bus Cycle

17.3.2 Control Signal Timing

(1) Reset Input Timing

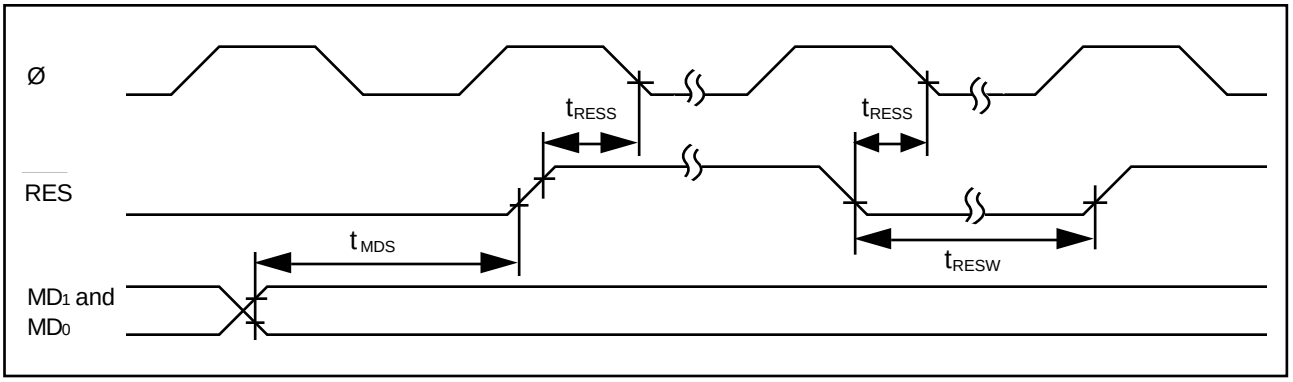


Figure 17-7. Reset Input Timing

(2) Interrupt Input Timing

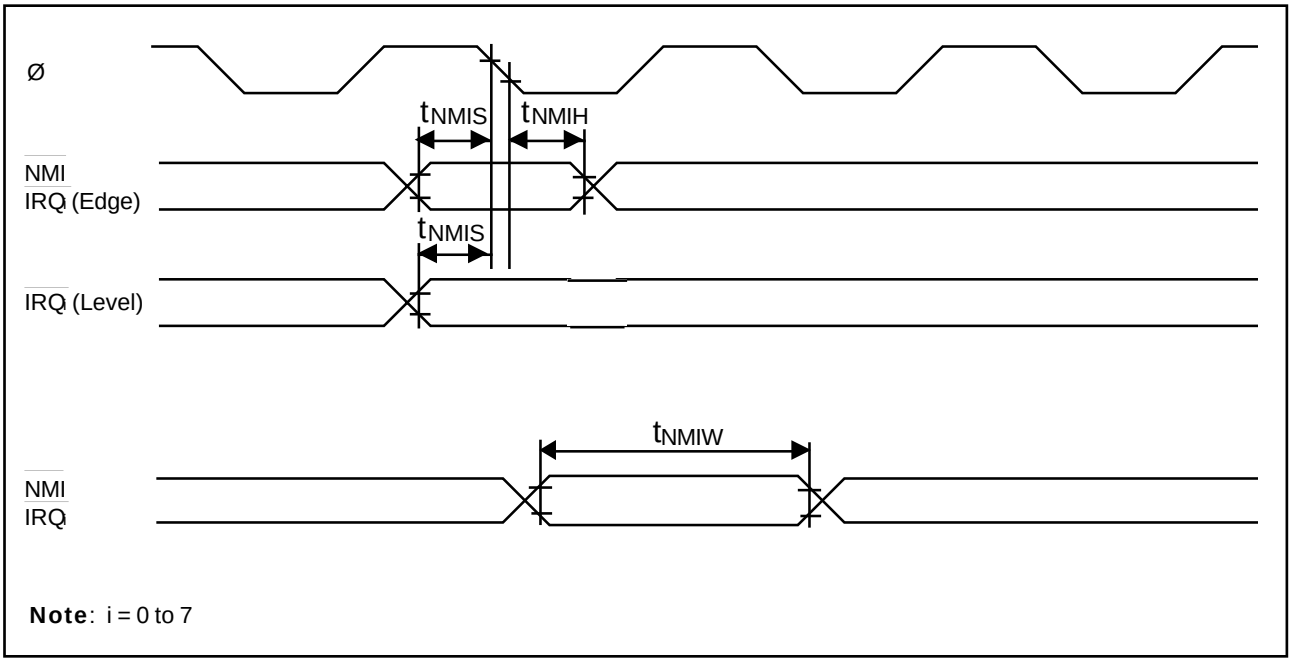


Figure 17-8. Interrupt Input Timing

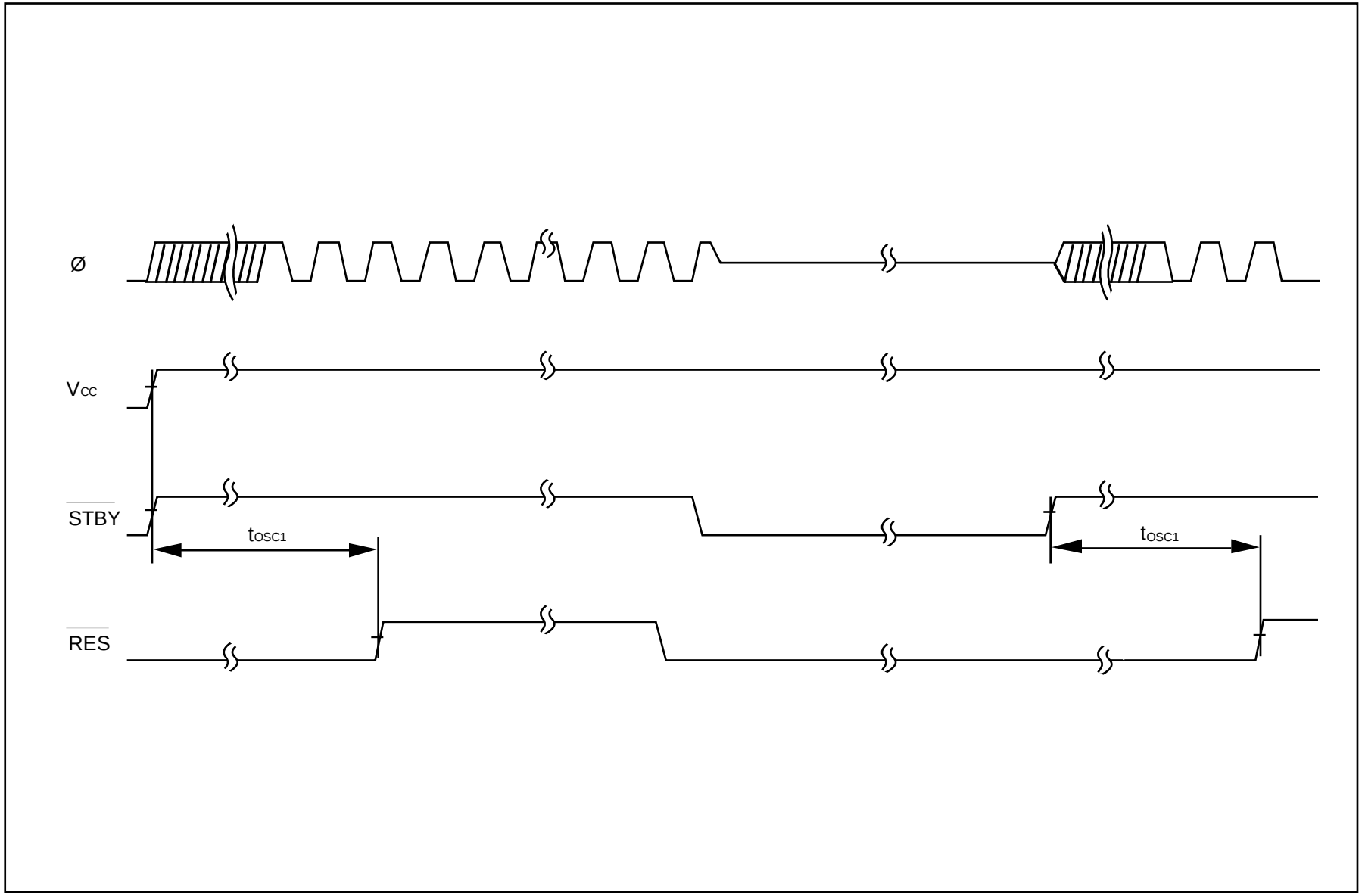


Figure 17-9. Clock Setting Timing

(4) Clock Settling Timing for Recovery from Software Standby Mode

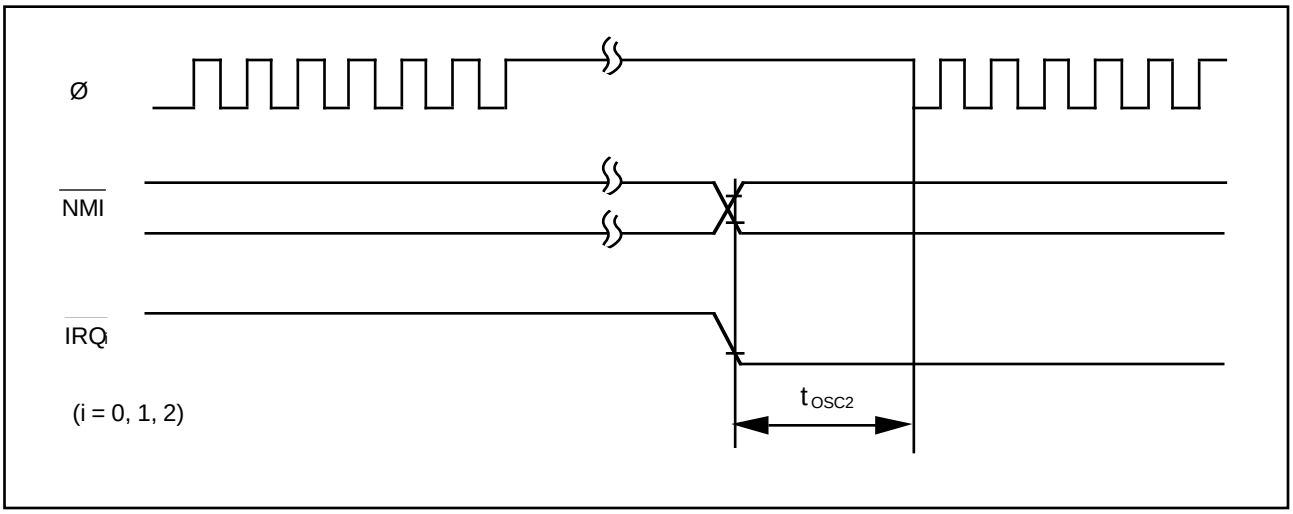


Figure 17-10. Clock Settling Timing for Recovery from Software Standby Mode

17.3.3 16-Bit Free-Running Timer Timing

(1) Free-Running Timer Input/Output Timing

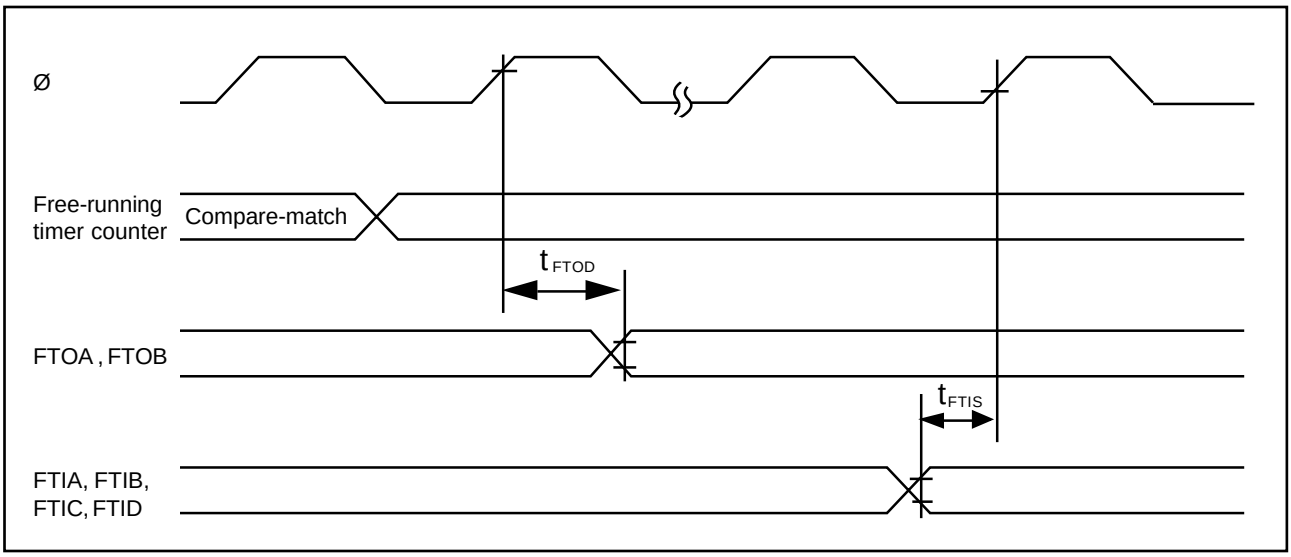


Figure 17-11. Free-Running Timer Input/Output Timing

(2) External Clock Input Timing for Free-Running Timer

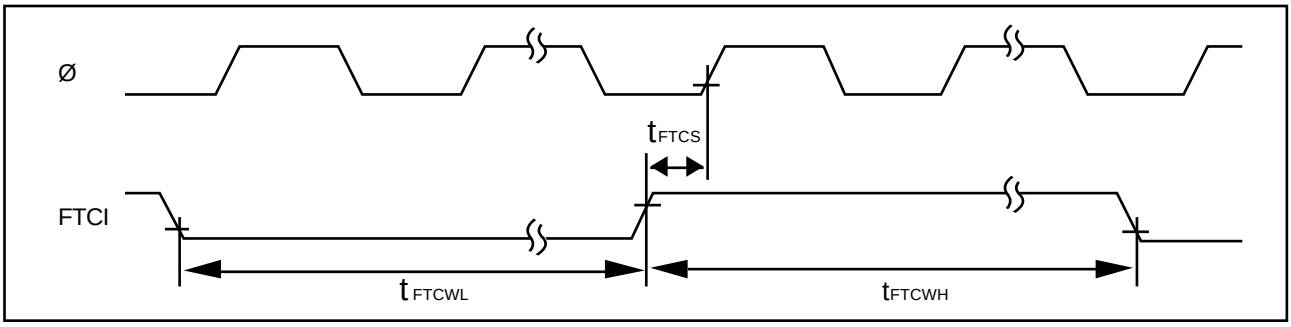


Figure 17-12. External Clock Input Timing for Free-Running Timer

17.3.4 8-Bit Timer Timing

(1) 8-Bit Timer Output Timing

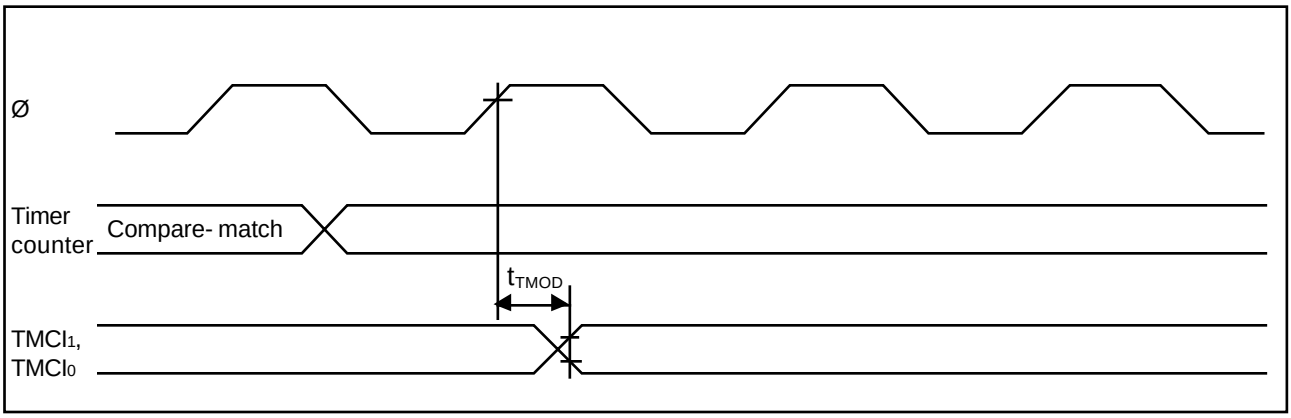


Figure 17-13. 8-Bit Timer Output Timing

(2) 8-Bit Timer Clock Input Timing

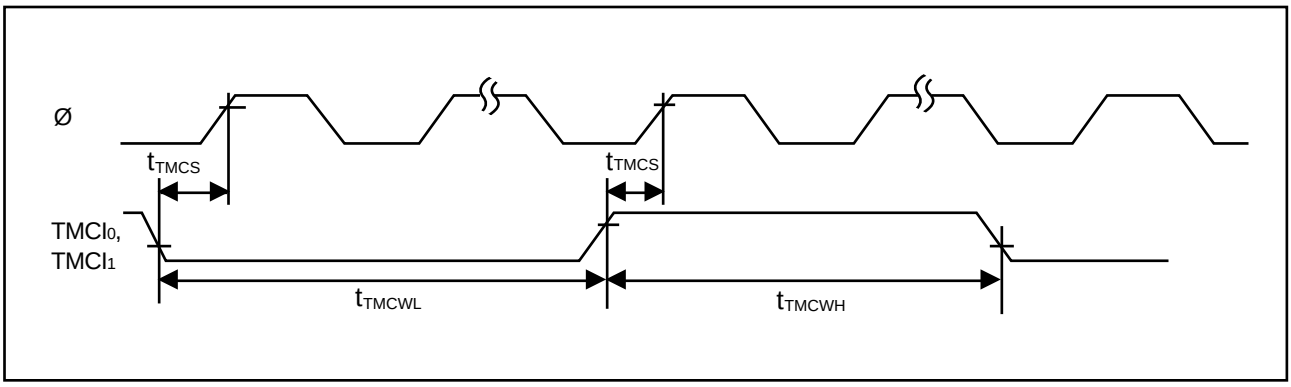


Figure 17-14. 8-Bit Timer Clock Input Timing

(3) 8-Bit Timer Reset Input Timing

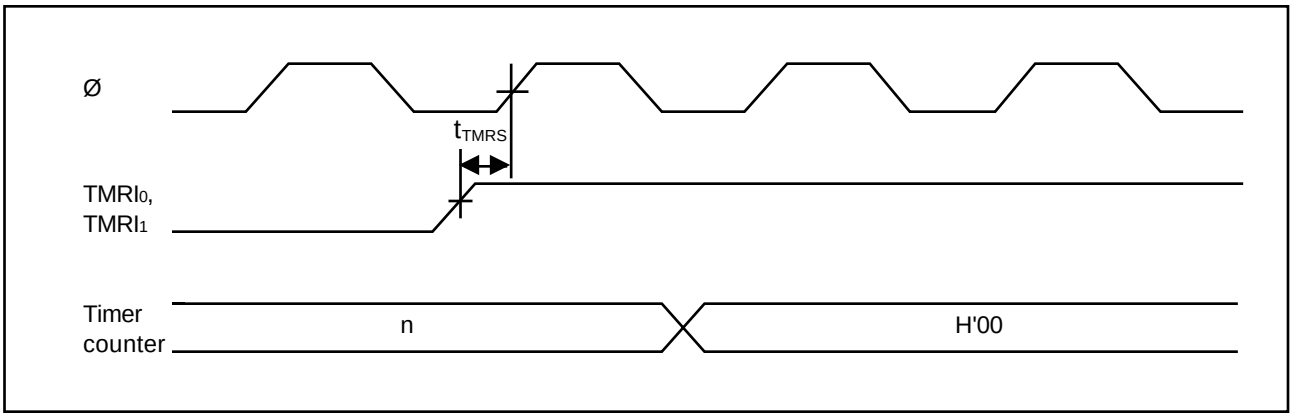


Figure 17-15. 8-Bit Timer Reset Input Timing

17.3.5 Pulse Width Modulation Timer Timing

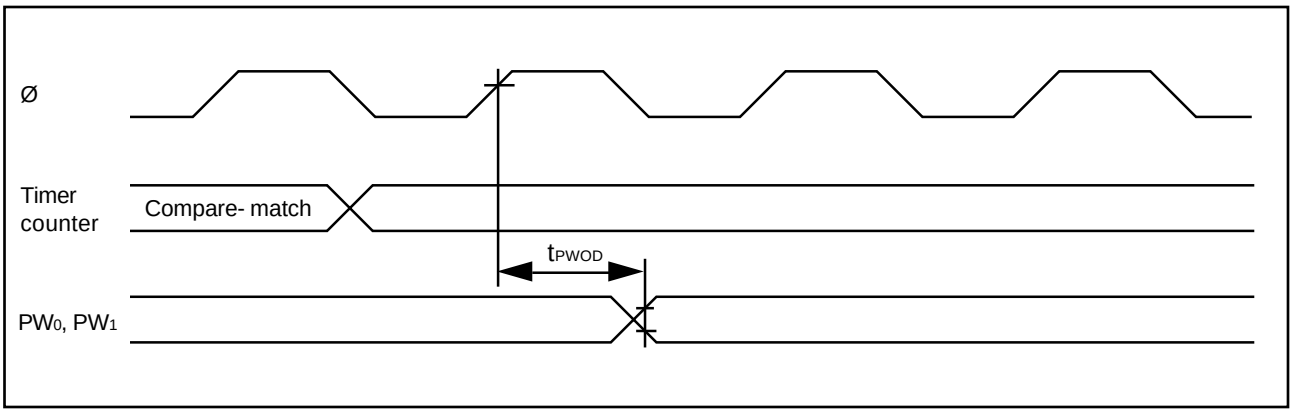


Figure 17-16. PWM Timer Output Timing

17.3.6 Serial Communication Interface Timing

(1) SCI Input/Output Timing

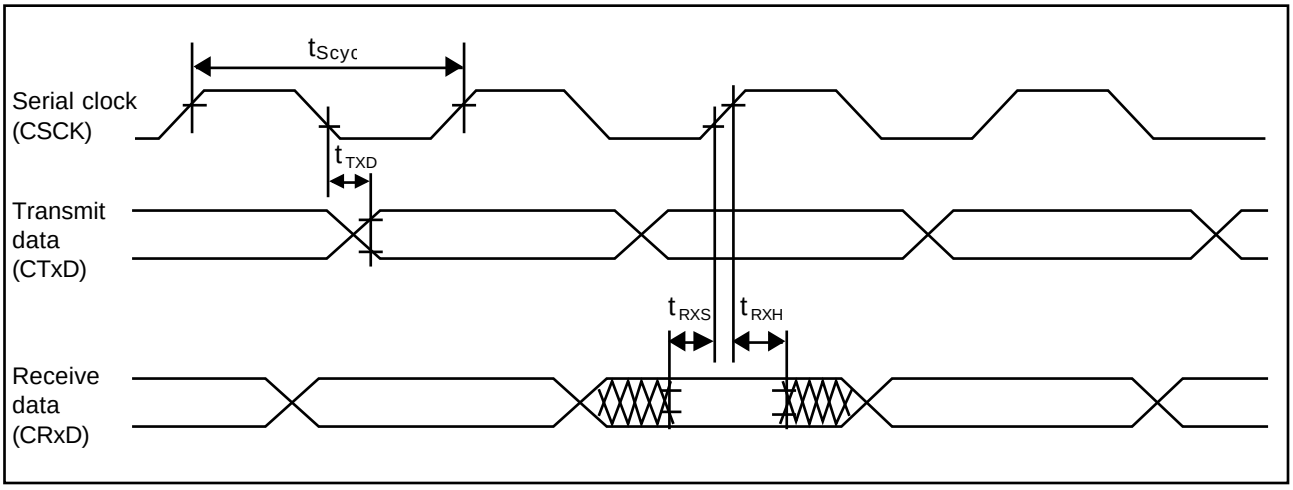


Figure 17-17. SCI Input/Output Timing (Synchronous Mode)

(2) SCI Input Clock Timing

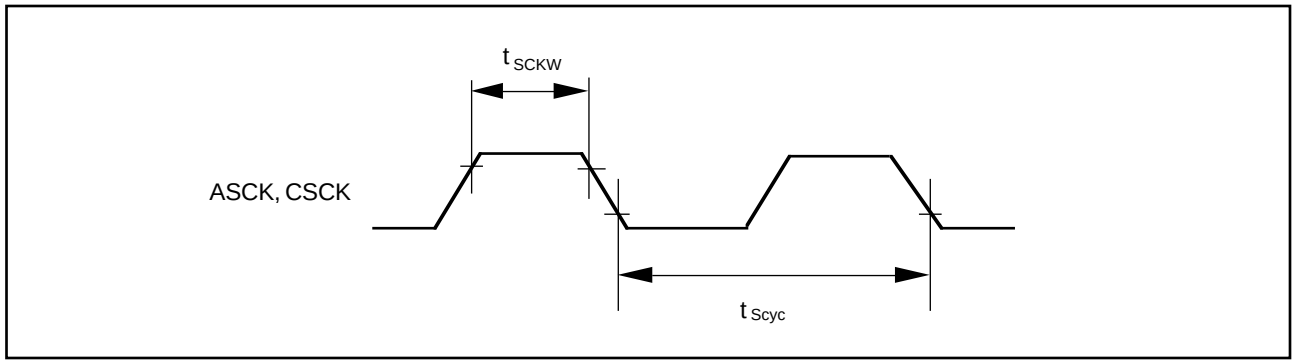


Figure 17-18. SCI Input Clock Timing

17.3.7 I/O Port Timing

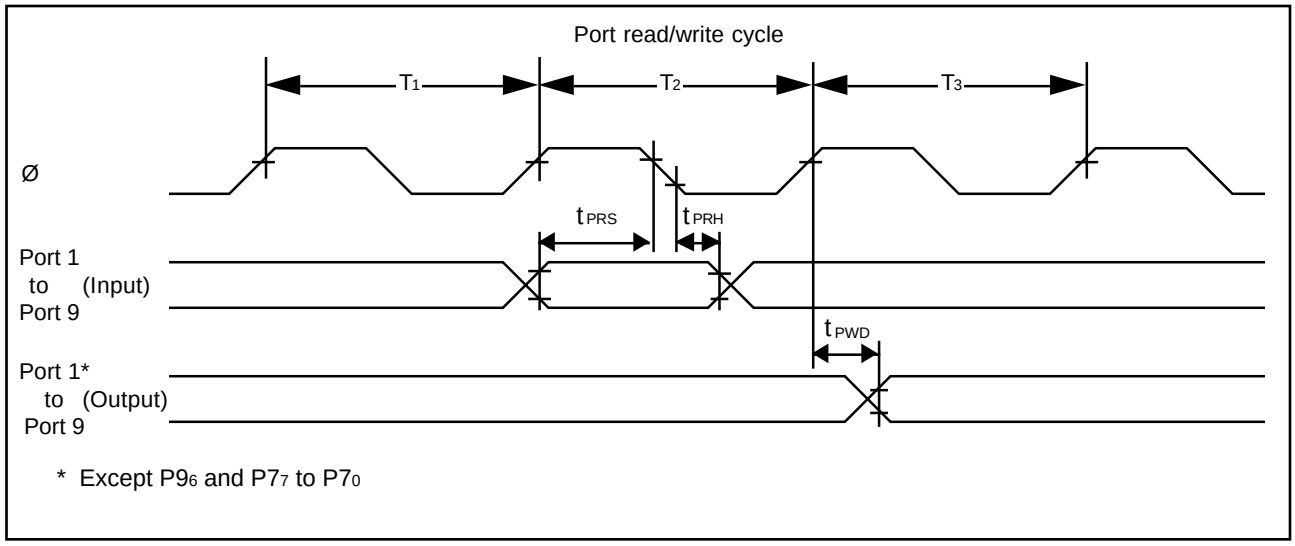
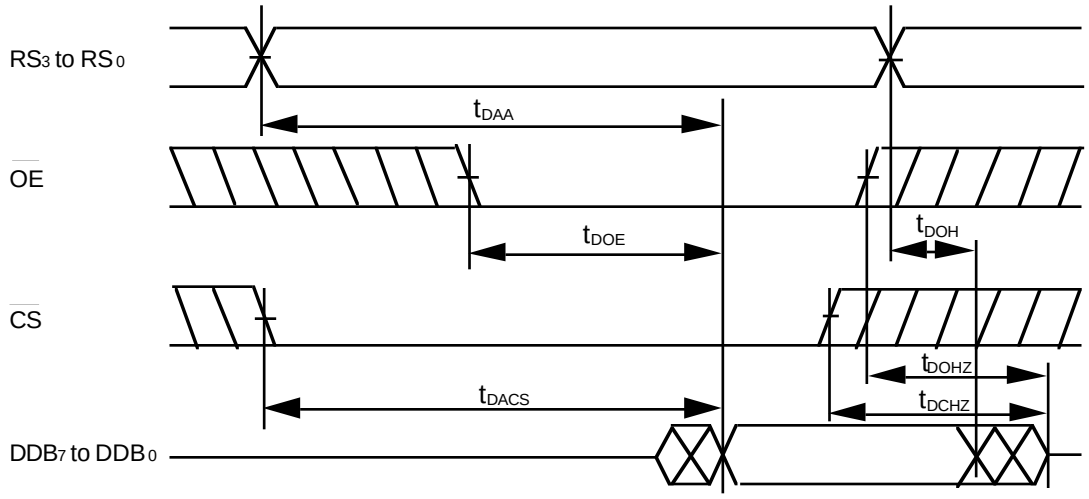


Figure 17-19. I/O Port Input/Output Timing

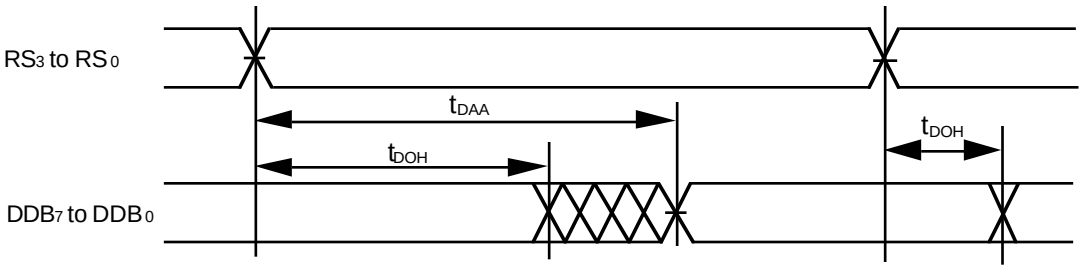
17.3.8 Dual-Port RAM Timing

(1) Read Cycle 1



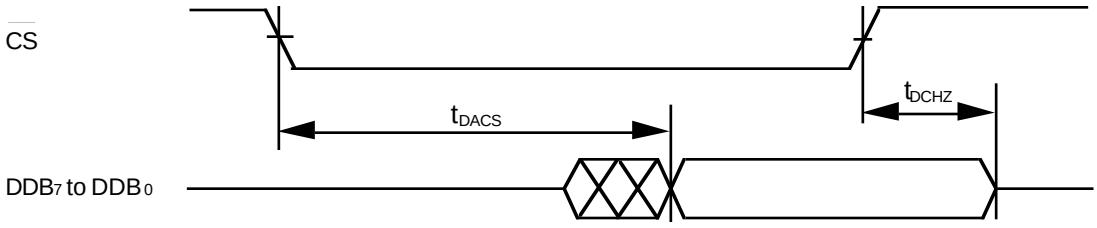
Note: $\overline{WE}$ should be High during a read cycle.

(2) Read Cycle 2



Notes: 1. $\overline{WE}$ should be High during a read cycle.
2. $\overline{CS}$ = V_{IL}
3. $\overline{OE}$ = V_{IL}

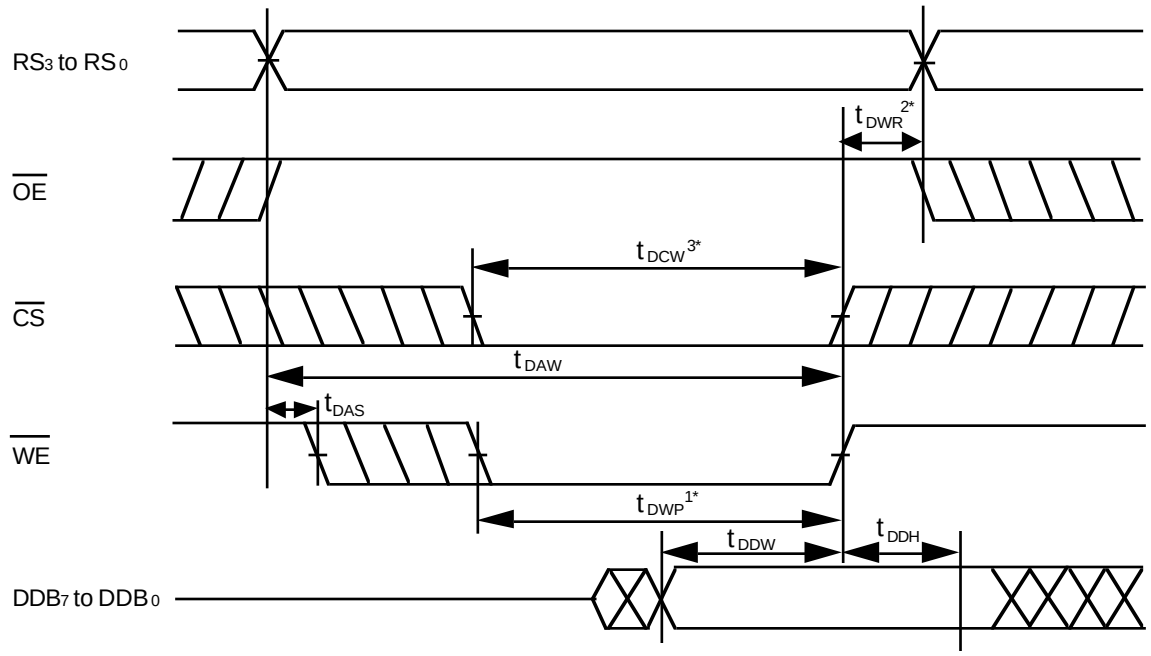
(3) Read Cycle 3



Notes: 1. $\overline{WE}$ should be High during a read cycle.
2. The address on the register select lines should be set up by the time $\overline{CS}$ goes Low, or before that time.
3. $\overline{OE}$ = V_{IL}

Figure 17-20. Dual-Port RAM Read Timing

(4) Write Cycle



Notes: 1. Data are written while $\overline{CS}$ and $\overline{WE}$ are both low (t_{DWP}).
2. t_{DWR} is measured from the rise of $\overline{CS}$ or $\overline{WE}$, whichever rises first.
3. If $\overline{CS}$ goes Low at the same time as $\overline{WE}$ goes Low or after $\overline{WE}$ goes Low, the output remains in the high-impedance state.

Figure 17-21. Dual-Port RAM Write Timing

Appendix A. CPU Instruction Set

A.1 Instruction Set List

Operation Notation

Rd8/16	General register (destination) (8 or 16 bits)
Rs8/16	General register (source) (8 or 16 bits)
Rn8/16	General register (8 or 16 bits)
CCR	Condition code register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#xx:3/8/16	Immediate data (3, 8, or 16 bits)
d:8/16	Displacement (8 or 16 bits)
@aa:8/16	Absolute address (8 or 16 bits)
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	AND logical
∨	OR logical
⊕	Exclusive OR logical
→	Move
—	Not

Condition Code Notation

↕	Modified according to the instruction result
*	Undetermined (unpredictable)
0	Always cleared to “0”
—	Not affected by the instruction result

Table A-1. Instruction Set

Mnemonic	Operand size	Operation	Addressing mode/ instruction length									Condition code						No. of states
			#xx:8/16	Rn	@Rn	@(d:16,Rn)	@-Rn/@Rn+	@aa:8/16	@(d:8,PC)	@@aa	Implied	I	H	N	Z	V	C	
MOV.B #xx:8,Rd	B	#xx:8 → Rd8	2									–	–	↕	↕	0	–	2
MOV.B Rs,Rd	B	Rs8 → Rd8		2								–	–	↕	↕	0	–	2
MOV.B @Rs,Rd	B	@Rs16 → Rd8			2							–	–	↕	↕	0	–	4
MOV.B @(d:16,Rs),Rd	B	@(d:16,Rs16) → Rd8				4						–	–	↕	↕	0	–	6
MOV.B @Rs+,Rd	B	@Rs16 → Rd8 Rs16+1 → Rs16					2					–	–	↕	↕	0	–	6
MOV.B @aa:8,Rd	B	@aa:8 → Rd8						2				–	–	↕	↕	0	–	4
MOV.B @aa:16,Rd	B	@aa:16 → Rd8						4				–	–	↕	↕	0	–	6
MOV.B Rs,@Rd	B	Rs8 → @Rd16			2							–	–	↕	↕	0	–	4
MOV.B Rs,@(d:16,Rd)	B	Rs8 → @(d:16,Rd16)				4						–	–	↕	↕	0	–	6
MOV.B Rs,@–Rd	B	Rd16–1 → Rd16 Rs8 → @Rd16					2					–	–	↕	↕	0	–	6
MOV.B Rs,@aa:8	B	Rs8 → @aa:8						2				–	–	↕	↕	0	–	4
MOV.B Rs,@aa:16	B	Rs8 → @aa:16						4				–	–	↕	↕	0	–	6
MOV.W #xx:16,Rd	W	#xx:16 → Rd16	4									–	–	↕	↕	0	–	4
MOV.W Rs,Rd	W	Rs16 → Rd16		2								–	–	↕	↕	0	–	2
MOV.W @Rs,Rd	W	@Rs16 → Rd16			2							–	–	↕	↕	0	–	4
MOV.W @(d:16,Rs),Rd	W	@(d:16,Rs16) → Rd16				4						–	–	↕	↕	0	–	6
MOV.W @Rs+,Rd	W	@Rs16 → Rd16 Rs16+2 → Rs16					2					–	–	↕	↕	0	–	6
MOV.W @aa:16,Rd	W	@aa:16 → Rd16						4				–	–	↕	↕	0	–	6
MOV.W Rs,@Rd	W	Rs16 → @Rd16			2							–	–	↕	↕	0	–	4
MOV.W Rs,@(d:16,Rd)	W	Rs16 → @(d:16,Rd16)				4						–	–	↕	↕	0	–	6
MOV.W Rs,@–Rd	W	Rd16–2 → Rd16 Rs16 → @Rd16					2					–	–	↕	↕	0	–	6
MOV.W Rs, @aa:16	W	Rs16 → @aa:16						4				–	–	↕	↕	0	–	6
POP Rd	W	@SP → Rd16 SP+2 → SP					2					–	–	↕	↕	0	–	6
PUSH Rs	W	SP–2 → SP Rs16 → @SP					2					–	–	↕	↕	0	–	6
MOVFPE @aa:16,Rd	B	Not supported																
MOVTPE Rs,@aa:16	B	Not supported																
EEPMOV	–	if R4L≠0 then Repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4L–1 → R4L Until R4L=0 else next									4	–	–	–	–	–	–	[4]

Table A-1. Instruction Set (cont.)

Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
			#xx:8/16	Rn	@Rn	@(d:16,Rn)	@-Rn/@Rn+	@aa:8/16	@(d:8,PC)	@@aa	I	H	N	Z	V	C	
ADD.B #xx:8,Rd	B	Rd8+#xx:8 → Rd8	2								–	↕	↕	↕	↕	↕	2
ADD.B Rs,Rd	B	Rs8+Rd8 → Rd8		2							–	↕	↕	↕	↕	↕	2
ADD.W Rs,Rd	W	Rs16+Rd16 → Rd16		2							–	[1]	↕	↕	↕	↕	2
ADDX.B #xx:8,Rd	B	Rd8+#xx:8 +C → Rd8	2								–	↕	↕	[2]	↕	↕	2
ADDX.B Rs,Rd	B	Rd8+Rs8 +C → Rd8		2							–	↕	↕	[2]	↕	↕	2
ADDS.W #1,Rd	W	Rd16+1 → Rd16		2							–	–	–	–	–	–	2
ADDS.W #2,Rd	W	Rd16+2 → Rd16		2							–	–	–	–	–	–	2
INC.B Rd	B	Rd8+1 → Rd8		2							–	–	↕	↕	↕	–	2
DAA.B Rd	B	Rd8 decimal adjust → Rd8		2							–	*	↕	↕	*	[3]	2
SUB.B Rs,Rd	B	Rd8–Rs8 → Rd8		2							–	↕	↕	↕	↕	↕	2
SUB.W Rs,Rd	W	Rd16–Rs16 → Rd16		2							–	[1]	↕	↕	↕	↕	2
SUBX.B #xx:8,Rd	B	Rd8–#xx:8 –C → Rd8	2								–	↕	↕	[2]	↕	↕	2
SUBX.B Rs,Rd	B	Rd8–Rs8 –C → Rd8		2							–	↕	↕	[2]	↕	↕	2
SUBS.W #1,Rd	W	Rd16–1 → Rd16		2							–	–	–	–	–	–	2
SUBS.W #2,Rd	W	Rd16–2 → Rd16		2							–	–	–	–	–	–	2
DEC.B Rd	B	Rd8–1 → Rd8		2							–	–	↕	↕	↕	–	2
DAS.B Rd	B	Rd8 decimal adjust → Rd8		2							–	*	↕	↕	*	–	2
NEG.B Rd	B	0–Rd → Rd		2							–	↕	↕	↕	↕	↕	2
CMP.B #xx:8,Rd	B	Rd8–#xx:8	2								–	↕	↕	↕	↕	↕	2
CMP.B Rs,Rd	B	Rd8–Rs8		2							–	↕	↕	↕	↕	↕	2
CMP.W Rs,Rd	W	Rd16–Rs16		2							–	[1]	↕	↕	↕	↕	2
MULXU.B Rs,Rd	B	Rd8×Rs8 → Rd16		2							–	–	–	–	–	–	14
DIVXU.B Rs,Rd	B	Rd16÷Rs8 → Rd16 (RdH:remainder,RdL:quotient)		2							–	–	[6]	[7]	–	–	14
AND.B #xx:8,Rd	B	Rd8∧#xx:8 → Rd8	2								–	–	↕	↕	0	–	2
AND.B Rs,Rd	B	Rd8∧Rs8 → Rd8		2							–	–	↕	↕	0	–	2
OR.B #xx:8,Rd	B	Rd8∨#xx:8 → Rd8	2								–	–	↕	↕	0	–	2
OR.B Rs,Rd	B	Rd8∨Rs8 → Rd8		2							–	–	↕	↕	0	–	2
XOR.B #xx:8,Rd	B	Rd8⊕#xx:8 → Rd8	2								–	–	↕	↕	0	–	2
XOR.B Rs,Rd	B	Rd8⊕Rs8 → Rd8		2							–	–	↕	↕	0	–	2
NOT.B Rd	B	$\overline{\text{Rd}}$ → Rd		2							–	–	↕	↕	0	–	2

Table A-1. Instruction Set (cont.)

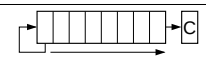
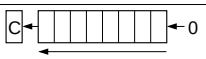
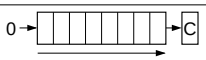
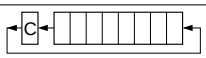
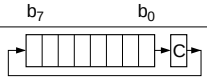
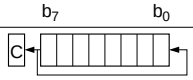
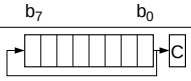
Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
			#xx:8/16	Rn	@Rn	@(d:16,Rn)	@-Rn/@Rn+	@aa:8/16	@(d:8,PC)	@@aa	I	H	N	Z	V	C	
SHAL.B Rd	B			2							–	–	↕	↕	↕	↕	2
SHAR.B Rd	B			2							–	–	↕	↕	0	↕	2
SHLL.B Rd	B			2							–	–	↕	↕	0	↕	2
SHLR.B Rd	B			2							–	–	0	↕	0	↕	2
ROTXL.B Rd	B			2							–	–	↕	↕	0	↕	2
ROTXR.B Rd	B			2							–	–	↕	↕	0	↕	2
ROTL.B Rd	B			2							–	–	↕	↕	0	↕	2
ROTR.B Rd	B			2							–	–	↕	↕	0	↕	2
BSET #xx:3,Rd	B	(#xx:3 of Rd8) ← 1		2							–	–	–	–	–	–	2
BSET #xx:3,@Rd	B	(#xx:3 of @Rd16) ← 1			4						–	–	–	–	–	–	8
BSET #xx:3,@aa:8	B	(#xx:3 of @aa:8) ← 1						4			–	–	–	–	–	–	8
BSET Rn,Rd	B	(Rn8 of Rd8) ← 1		2							–	–	–	–	–	–	2
BSET Rn,@Rd	B	(Rn8 of @Rd16) ← 1			4						–	–	–	–	–	–	8
BSET Rn,@aa:8	B	(Rn8 of @aa:8) ← 1						4			–	–	–	–	–	–	8
BCLR #xx:3,Rd	B	(#xx:3 of Rd8) ← 0		2							–	–	–	–	–	–	2
BCLR #xx:3,@Rd	B	(#xx:3 of @Rd16) ← 0			4						–	–	–	–	–	–	8
BCLR #xx:3,@aa:8	B	(#xx:3 of @aa:8) ← 0						4			–	–	–	–	–	–	8
BCLR Rn,Rd	B	(Rn8 of Rd8) ← 0		2							–	–	–	–	–	–	2
BCLR Rn,@Rd	B	(Rn8 of @Rd16) ← 0			4						–	–	–	–	–	–	8
BCLR Rn,@aa:8	B	(Rn8 of @aa:8) ← 0						4			–	–	–	–	–	–	8
BNOT #xx:3,Rd	B	(#xx:3 of Rd8) ← ($\overline{\text{#xx:3 of Rd8}}$)		2							–	–	–	–	–	–	2
BNOT #xx:3,@Rd	B	(#xx:3 of @Rd16) ← ($\overline{\text{#xx:3 of @Rd16}}$)			4						–	–	–	–	–	–	8
BNOT #xx:3,@aa:8	B	(#xx:3 of @aa:8) ← ($\overline{\text{#xx:3 of @aa:8}}$)						4			–	–	–	–	–	–	8

Table A-1. Instruction Set (cont.)

Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
			#xx:8/16	Rn	@Rn	@(d:16,Rn)	@-Rn/@Rn+	@aa:8/16	@(d:8,PC)	@@aa	I	H	N	Z	V	C	
BNOT Rn,Rd	B	(Rn8 of Rd8) ← ($\overline{\text{Rn8}}$ of $\overline{\text{Rd8}}$)		2							–	–	–	–	–	–	2
BNOT Rn,@Rd	B	(Rn8 of @Rd16) ← ($\overline{\text{Rn8}}$ of $\overline{\text{@Rd16}}$)			4						–	–	–	–	–	–	8
BNOT Rn,@aa:8	B	(Rn8 of @aa:8) ← ($\overline{\text{Rn8}}$ of $\overline{\text{@aa:8}}$)						4			–	–	–	–	–	–	8
BTST #xx:3,Rd	B	($\overline{\text{#xx:3}}$ of $\overline{\text{Rd8}}$) → Z		2							–	–	–	↕	–	–	2
BTST #xx:3,@Rd	B	($\overline{\text{#xx:3}}$ of $\overline{\text{@Rd16}}$) → Z			4						–	–	–	↕	–	–	6
BTST #xx:3,@aa:8	B	($\overline{\text{#xx:3}}$ of $\overline{\text{@aa:8}}$) → Z						4			–	–	–	↕	–	–	6
BTST Rn,Rd	B	($\overline{\text{Rn8}}$ of $\overline{\text{Rd8}}$) → Z		2							–	–	–	↕	–	–	2
BTST Rn,@Rd	B	($\overline{\text{Rn8}}$ of $\overline{\text{@Rd16}}$) → Z			4						–	–	–	↕	–	–	6
BTST Rn,@aa:8	B	($\overline{\text{Rn8}}$ of $\overline{\text{@aa:8}}$) → Z						4			–	–	–	↕	–	–	6
BLD #xx:3,Rd	B	(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
BLD #xx:3,@Rd	B	(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
BLD #xx:3,@aa:8	B	(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
BILD #xx:3,Rd	B	($\overline{\text{#xx:3}}$ of $\overline{\text{Rd8}}$) → C		2							–	–	–	–	–	↕	2
BILD #xx:3,@Rd	B	($\overline{\text{#xx:3}}$ of $\overline{\text{@Rd16}}$) → C			4						–	–	–	–	–	↕	6
BILD #xx:3,@aa:8	B	($\overline{\text{#xx:3}}$ of $\overline{\text{@aa:8}}$) → C						4			–	–	–	–	–	↕	6
BST #xx:3,Rd	B	C → (#xx:3 of Rd8)		2							–	–	–	–	–	–	2
BST #xx:3,@Rd	B	C → (#xx:3 of @Rd16)			4						–	–	–	–	–	–	8
BST #xx:3,@aa:8	B	C → (#xx:3 of @aa:8)						4			–	–	–	–	–	–	8
BIST #xx:3,Rd	B	$\overline{\text{C}}$ → (#xx:3 of Rd8)		2							–	–	–	–	–	–	2
BIST #xx:3,@Rd	B	$\overline{\text{C}}$ → (#xx:3 of @Rd16)			4						–	–	–	–	–	–	8
BIST #xx:3,@aa:8	B	$\overline{\text{C}}$ → (#xx:3 of @aa:8)						4			–	–	–	–	–	–	8
BAND #xx:3,Rd	B	C∧(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
BAND #xx:3,@Rd	B	C∧(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
BAND #xx:3,@aa:8	B	C∧(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
BIAND #xx:3,Rd	B	C∧($\overline{\text{#xx:3}}$ of $\overline{\text{Rd8}}$) → C		2							–	–	–	–	–	↕	2
BIAND #xx:3,@Rd	B	C∧($\overline{\text{#xx:3}}$ of $\overline{\text{@Rd16}}$) → C			4						–	–	–	–	–	↕	6
BIAND #xx:3, @aa:8	B	C∧($\overline{\text{#xx:3}}$ of $\overline{\text{@aa:8}}$) → C						4			–	–	–	–	–	↕	6
BOR #xx:3,Rd	B	C∨(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
BOR #xx:3,@Rd	B	C∨(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
BOR #xx:3,@aa:8	B	C∨(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
BIOR #xx:3,Rd	B	C∨($\overline{\text{#xx:3}}$ of $\overline{\text{Rd8}}$) → C		2							–	–	–	–	–	↕	2

Table A-1. Instruction Set (cont.)

Mnemonic	Operand size	Operation	Branching condition	Addressing mode/ instruction length								Condition code						No. of states
				#xx:8/16	Rn	@Rn	@(d:16,Rn)	@-Rn/@Rn+	@aa:8/16	@(d:8,PC)	@@aa	I	H	N	Z	V	C	
BIOR #xx:3,@Rd	B	C∨(#xx:3 of @Rd16) → C				4						–	–	–	–	–	↕	6
BIOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C							4			–	–	–	–	–	↕	6
BXOR #xx:3,Rd	B	C⊕(#xx:3 of Rd8) → C			2							–	–	–	–	–	↕	2
BXOR #xx:3,@Rd	B	C⊕(#xx:3 of @Rd16) → C				4						–	–	–	–	–	↕	6
BXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C							4			–	–	–	–	–	↕	6
BIXOR #xx:3,Rd	B	C⊕(#xx:3 of Rd8) → C			2							–	–	–	–	–	↕	2
BIXOR #xx:3,@Rd	B	C⊕(#xx:3 of @Rd16) → C				4						–	–	–	–	–	↕	6
BIXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C							4			–	–	–	–	–	↕	6
BRA d:8 (BT d:8)	–	PC ← PC+d:8								2		–	–	–	–	–	–	4
BRNd:8 (BF d:8)	–	PC ← PC+2								2		–	–	–	–	–	–	4
BHI d:8	–	if condition	C∨Z = 0							2		–	–	–	–	–	–	4
BLS d:8	–	is true then	C∨Z = 1							2		–	–	–	–	–	–	4
BCC d:8 (BHS d:8)	–	PC ← PC+d:8	C = 0							2		–	–	–	–	–	–	4
BCS d:8 (BLO d:8)	–	else next;	C = 1							2		–	–	–	–	–	–	4
BNE d:8	–		Z = 0							2		–	–	–	–	–	–	4
BEQ d:8	–		Z = 1							2		–	–	–	–	–	–	4
BVC d:8	–		V = 0							2		–	–	–	–	–	–	4
BVS d:8	–		V = 1							2		–	–	–	–	–	–	4
BPL d:8	–		N = 0							2		–	–	–	–	–	–	4
BMI d:8	–		N = 1							2		–	–	–	–	–	–	4
BGE d:8	–		N⊕V = 0							2		–	–	–	–	–	–	4
BLT d:8	–		N⊕V = 1							2		–	–	–	–	–	–	4
BGT d:8	–		Z∨(N⊕V) = 0							2		–	–	–	–	–	–	4
BLE d:8	–		Z∨(N⊕V) = 1							2		–	–	–	–	–	–	4
JMP @Rn	–	PC ← Rn16				2						–	–	–	–	–	–	4
JMP @aa:16	–	PC ← aa:16							4			–	–	–	–	–	–	6
JMP @@aa:8	–	PC ← @aa:8									2	–	–	–	–	–	–	8
BSR d:8	–	SP–2 → SP PC → @SP PC ← PC+d:8								2		–	–	–	–	–	–	6

Table A-1. Instruction Set (cont.)

Mnemonic	Operand size	Operation	Addressing mode/ instruction length									Condition code						No. of states
			#xx:8/16	Rn	@Rn	@(d:16,Rn)	@-Rn/@Rn+	@aa:8/16	@(d:8,PC)	@@aa	Implied	I	H	N	Z	V	C	
JSR @Rn	–	SP–2 → SP PC → @SP PC ← Rn16			2							–	–	–	–	–	–	6
JSR @aa:16	–	SP–2 → SP PC → @SP PC ← aa:16						4				–	–	–	–	–	–	8
JSR @@aa:8	–	SP–2 → SP PC → @SP PC ← @aa:8								2		–	–	–	–	–	–	8
RTS	–	PC ← @SP SP+2 → SP									2	–	–	–	–	–	–	8
RTE	–	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
SLEEP	–	Transit to sleep mode.									2	–	–	–	–	–	–	2
LDC #xx:8,CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
LDC Rs,CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
STC CCR,Rd	B	CCR → Rd8		2								–	–	–	–	–	–	2
ANDC #xx:8,CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
ORC #xx:8,CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
XORC #xx:8,CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
NOP	–	PC ← PC+2									2	–	–	–	–	–	–	2

Notes: The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.

[1] Set to “1” when there is a carry or borrow from bit 11; otherwise cleared to “0”.

[2] If the result is zero, the previous value of the flag is retained; otherwise the flag is cleared to “0”.

[3] Set to “1” if decimal adjustment produces a carry; otherwise cleared to “0”.

[4] The number of states required for execution is 4n+8 (n = value of R4L)

[5] These instructions are not supported by the H8/338 Series.

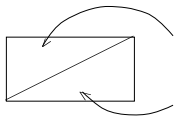
[6] Set to “1” if the divisor is negative; otherwise cleared to ”0”.

[7] Cleared to “0:”if the divisor is not zero; “1” when the divisor is zero.

A.2 Operation Code Map

Table A-2 is a map of the operation codes contained in the first byte of the instruction code (bits 15 to 8 of the first instruction word).

Some pairs of instructions have identical first bytes. These instructions are differentiated by the first bit of the second byte (bit 7 of the first instruction word).



Instruction when first bit of byte 2 (bit 7 of first instruction word) is "0."
Instruction when first bit of byte 2 (bit 7 of first instruction word) is "1."

Table A-2. Operation Code Map

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SLEEP	STC	LDC	ORC	XORC	ANDC	LDC	ADD		INC	ADDS	MOV		ADDX	DAA
1	SHLL / SHAL	SHLR / SHAR	ROTXL / ROTL	ROTXR / ROTR	OR	XOR	AND	NOT / NEG	SUB		DEC	SUBS	CMP		SUBX	DAS
2	MOV															
3																
4	BRA *2	BRN *2	BHI	BLS	BCC *2	BCS *2	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU			RTS	BSR	RTE			JMP				JSR		
6	BSET	BNOT	BCLR	BTST				BST / BIST	MOV *1							
7					BOR / BIOR	BXOR / BIXOR	BAND / BIAND	BLD / BILD		MOV		EEPMOV	Bit manipulation instruction			
8	ADD															
9	ADDX															
A	CMP															
B	SUBX															
C	OR															
D	XOR															
E	AND															
F	MOV															

*1 The MOVFPE and MOVTPE instructions are identical to MOV instructions in the first byte and first bit of the second byte (bits 15 to 7 of the instruction word). The PUSH and POP instructions are identical in machine language to MOV instructions.

*2 The BT, BF, BHS, and BLO instructions are identical in machine language to BRA, BRN, BCC, and BCS, respectively.

A.3 Number of States Required for Execution

The tables below can be used to calculate the number of states required for instruction execution. Table A-3 indicates the number of states required for each cycle (instruction fetch, branch address read, stack operation, byte data access, word data access, internal operation). Table A-4 indicates the number of cycles of each type occurring in each instruction. The total number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

Examples: Mode 1 (on-chip ROM disabled), stack located in external memory, 1 wait state inserted in external memory access.

1. BSET #0, @FFC7
   From table A-4: $I = L = 2$, $J = K = M = N = 0$
   From table A-3: $S_I = 8$, $S_L = 3$
   Number of states required for execution: $2 \times 8 + 2 \times 3 = 22$

2. JSR @@30
   From table A-4: $I = 2$, $J = K = 1$, $L = M = N = 0$
   From table A-3: $S_I = S_J = S_K = 8$
   Number of states required for execution: $2 \times 8 + 1 \times 8 + 1 \times 8 = 32$

Table A-3. Number of States Taken by Each Cycle in Instruction Execution

Execution Status (instruction cycle)		Access Location		
		On-Chip Memory	On-Chip Reg. Field	External Memory
Instruction fetch	S_I	2	6	6 + 2m
Branch address read	S_J			
Stack operation	S_K			
Byte data access	S_L		3	3 + m
Word data access	S_M		6	6 + 2m
Internal operation	S_N		2	

Notes: 1. m: Number of wait states inserted in access to external device.
2. The byte data access cycle to an external device by the MOVFPE and MOVTPE instructions requires 9 to 16 states since it is synchronized with the E clock. See section 15, "E-Clock Interface" for timing details.

Table A-4. Number of Cycles in Each Instruction

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
ADD	ADD.B #xx:8, Rd	1					
	ADD.B Rs, Rd	1					
	ADD.W Rs, Rd	1					
ADDS	ADDS.W #1/2, Rd	1					
ADDX	ADDX.B #xx:8, Rd	1					
	ADDX.B Rs, Rd	1					
AND	AND.B #xx:8, Rd	1					
	AND.B Rs, Rd	1					
ANDC	ANDC #xx:8, CCR	1					
BAND	BAND #xx:3, Rd	1					
	BAND #xx:3, @Rd	2			1		
	BAND #xx:3, @aa:8	2			1		
Bcc	BRA d:8 (BT d:8)	2					
	BRN d:8 (BF d:8)	2					
	BHI d:8	2					
	BLS d:8	2					
	BCC d:8 (BHS d:8)	2					
	BCS d:8 (BLO d:8)	2					
	BNE d:8	2					
	BEQ d:8	2					
	BVC d:8	2					
	BVS d:8	2					
	BPL d:8	2					
	BMI d:8	2					
	BGE d:8	2					
	BLT d:8	2					
	BGT d:8	2					
	BLE d:8	2					
BCLR	BCLR #xx:3, Rd	1					
	BCLR #xx:3, @Rd	2			2		
	BCLR #xx:3, @aa:8	2			2		
	BCLR Rn, Rd	1					
	BCLR Rn, @Rd	2			2		
	BCLR Rn, @aa:8	2			2		

Table A-4. Number of Cycles in Each Instruction (cont.)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BIAND	BIAND #xx:3, Rd	1					
	BIAND #xx:3, @Rd	2			1		
	BIAND #xx:3, @aa:8	2			1		
BILD	BILD #xx:3, Rd	1					
	BILD #xx:3, @Rd	2			1		
	BILD #xx:3, @aa:8	2			1		
BIOR	BIOR #xx:3 Rd	1					
	BIOR #xx:3 @Rd	2			1		
	BIOR #xx:3 @aa:8	2			1		
BIST	BIST #xx:3, Rd	1					
	BIST #xx:3, @Rd	2			2		
	BIST #xx:3, @aa:8	2			2		
BIXOR	BIXOR #xx:3, Rd	1					
	BIXOR #xx:3, @Rd	2			1		
	BIXOR #xx:3, @aa:8	2			1		
BLD	BLD #xx:3, Rd	1					
	BLD #xx:3, @Rd	2			1		
	BLD #xx:3, @aa:8	2			1		
BNOT	BNOT #xx:3, Rd	1					
	BNOT #xx:3, @Rd	2			2		
	BNOT #xx:3, @aa:8	2			2		
	BNOT Rn, Rd	1					
	BNOT Rn, @Rd	2			2		
	BNOT Rn, @aa:8	2			2		
BOR	BOR #xx:3, Rd	1					
	BOR #xx:3, @Rd	2			1		
	BOR #xx:3, @aa:8	2			1		
BSET	BSET #xx:3, Rd	1					
	BSET #xx:3, @Rd	2			2		
	BSET #xx:3, @aa:8	2			2		
	BSET Rn, Rd	1					
	BSET Rn, @Rd	2			2		
	BSET Rn, @aa:8	2			2		

Table A-4. Number of Cycles in Each Instruction (cont.)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BSR	BSR d:8	2		1			
BST	BST #xx:3, Rd	1					
	BST #xx:3, @Rd	2			2		
	BST #xx:3, @aa:8	2			2		
BTST	BTST #xx:3, Rd	1					
	BTST #xx:3, @Rd	2			1		
	BTST #xx:3, @aa:8	2			1		
	BTST Rn, Rd	1					
	BTST Rn, @Rd	2			1		
	BTST Rn, @aa:8	2			1		
BXOR	BXOR #xx:3, Rd	1					
	BXOR #xx:3, @Rd	2			1		
	BXOR #xx:3, @aa:8	2			1		
CMP	CMP.B #xx:8, Rd	1					
	CMP.B Rs, Rd	1					
	CMP.W Rs, Rd	1					
DAA	DAA.B Rd	1					
DAS	DAS.B Rd	1					
DEC	DEC.B Rd	1					
DIVXU	DIVXU.B Rs, Rd	1					6
EEPMOV	EEPMOV	2			2n+2*1		
INC	INC.B Rd	1					
JMP	JMP @Rn	2					
	JMP @aa:16	2					1
	JMP @@aa:8	2	1				1
JSR	JSR @Rn	2		1			
	JSR @aa:16	2		1			1
	JSR @@aa:8	2	1	1			
LDC	LDC #xx:8, CCR	1					
	LDC Rs, CCR	1					
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					
	MOV.B @Rs, Rd	1			1		
	MOV.B @(d:16,Rs), Rd	2			1		

Table A-4. Number of Cycles in Each Instruction (cont.)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
MOV	MOV.B @Rs+, Rd	1			1		1
	MOV.B @aa:8, Rd	1			1		
	MOV.B @aa:16, Rd	2			1		
	MOV.B Rs, @Rd	1			1		
	MOV.B Rs, @(d:16, Rd)	2			1		
	MOV.B Rs, @–Rd	1			1		1
	MOV.B Rs, @aa:8	1			1		
	MOV.B Rs, @aa:16	2			1		
	MOV.W #xx:16, Rd	2					
	MOV.W Rs, Rd	1					
	MOV.W @Rs, Rd	1				1	
	MOV.W @(d:16, Rs), Rd	2				1	
	MOV.W @Rs+, Rd	1				1	1
	MOV.W @aa:16, Rd	2				1	
	MOV.W Rs, @Rd	1				1	
	MOV.W Rs, @(d:16, Rd)	2				1	
	MOV.W Rs, @–Rd	1				1	1
	MOV.W Rs, @aa:16	2				1	
MOVFPE	MOVFPE @aa:16, Rd	2			1*2		
MOVTPE	MOVTPE.Rs, @aa:16	2			1*2		
MULXU	MULXU.Rs, Rd	1					6
NEG	NEG.B Rd	1					
NOP	NOP	1					
NOT	NOT.B Rd	1					
OR	OR.B #xx:8, Rd	1					
	OR.B Rs, Rd	1					
ORC	ORC #xx:8, CCR	1					
ROTL	ROTL.B Rd	1					
ROTR	ROTR.B Rd	1					
ROTXL	ROTXL.B Rd	1					
ROTXR	ROTXR.B Rd	1					
RTE	RTE	2		2			1
RTS	RTS	2		1			1

Table A-4. Number of Cycles in Each Instruction (cont.)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
SHAL	SHAL.B Rd	1					
SHAR	SHAR.B Rd	1					
SHLL	SHLL.B Rd	1					
SHLR	SHLR.B Rd	1					
SLEEP	SLEEP	1					
STC	STC CCR, Rd	1					
SUB	SUB.B Rs, Rd	1					
	SUB.W Rs, Rd	1					
SUBS	SUBS.W #1/2, Rd	1					
SUBX	SUBX.B #xx:8, Rd	1					
	SUBX.B Rs, Rd	1					
XOR	XOR.B #xx:8, Rd	1					
	XOR.B Rs, Rd	1					
XORC	XORC #xx:8, CCR	1					

Notes:

*1 n: Initial value in R4L. Source and destination are accessed n + 1 times each.

*2 Data access requires 9 to 16 states.

Appendix B. Register Field

B.1 Register Addresses and Bit Names

Addr. (last byte)	Register name	Bit names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'80										External addresses (in expanded modes)
H'81										
H'82										
H'83										
H'84										
H'85										
H'86										
H'87										
H'88										
H'89										
H'8A										
H'8B										
H'8C										
H'8D										
H'8E										
H'8F										
H’90	TIER	ICIAE	ICIBE	ICICE	ICIDE	OCIAE	OCIBE	OVIE	—	FRT
H’91	TCSR	ICFA	ICFB	ICFC	ICFD	OCFA	OCFB	OVF	CCLRA	
H’92	FRC (H)									
H’93	FRC (L)									
H’94	OCRA (H)									
	OCRB (H)									
H’95	OCRA (L)									
	OCRB (L)									
H’96	TCR	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0	
H’97	TOCR	—	—	—	OCRS	OEA	OEB	OLVLA	OLVLB	
H’98	ICRA (H)									
H’99	ICRA (L)									
H’9A	ICRB (H)									
H’9B	ICRB (L)									
H’9C	ICRC (H)									
H’9D	ICRC (L)									
H’9E	ICRD (H)									
H’9F	ICRD (L)									

Notes: FRT: Free-Running Timer

(Continued on next page)

(Continued from previous page)

Addr. (last byte)	Register name	Bit names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H’A0	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	PWM0
H’A1	DTR									
H’A2	TCNT									
H’A3	—	—	—	—	—	—	—	—	—	
H’A4	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	PWM1
H’A5	DTR									
H’A6	TCNT									
H’A7	—	—	—	—	—	—	—	—	—	
H'A8										External addresses (in expanded modes)
H'A9										
H'AA										
H'AB										
H'AC										
H'AD										
H'AE										
H'AF										
H’B0	P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	Port 1
H’B1	P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR	Port 2
H’B2	P1DR	P17	P16	P15	P14	P13	P12	P11	P10	Port 1
H’B3	P2DR	P27	P26	P25	P24	P23	P22	P21	P20	Port 2
H’B4	P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	Port 3
H’B5	P4DDR	P47DDR	P46DDR	P45DDR	P44DDR	P43DDR	P42DDR	P41DDR	P40DDR	Port 4
H’B6	P3DR	P37	P36	P35	P34	P33	P32	P31	P30	Port 3
H’B7	P4DR	P47	P46	P45	P44	P43	P42	P41	P40	Port 4
H’B8	P5DDR	—	—	—	—	—	P52DDR	P51DDR	P50DDR	Port 5
H’B9	P6DDR	P67DDR	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	Port 6
H’BA	P5DR	—	—	—	—	—	P52	P51	P50	Port 5
H’BB	P6DR	P67	P66	P65	P64	P63	P62	P61	P60	Port 6
H’BC	—	—	—	—	—	—	—	—	—	—
H’BD	P8DDR	—	P86DDR	P85DDR	P84DDR	P83DDR	P82DDR	P81DDR	P80DDR	Port 8
H’BE	P7DR	P77	P76	P75	P74	P73	P72	P71	P70	Port 7
H’BF	P8DR	—	P86	P85	P84	P83	P82	P81	P80	Port 8

(Continued on next page)

Notes: PWM0: Pulse-Width Modulation timer channel 0
PWM1: Pulse-Width Modulation timer channel 1

(Continued from preceding page)

Addr. (last byte)	Register name	Bit names								
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'C0	P9DDR	$P9_7$DDR	$P9_6$DDR	$P9_5$DDR	$P9_4$DDR	$P9_3$DDR	$P9_2$DDR	$P9_1$DDR	$P9_0$DDR	Port 9
H'C1	P9DR	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	$P9_0$	
H'C2	—	—	—	—	—	—	—	—	—	
H'C3	—	—	—	—	—	—	—	—	—	
H'C4	SYSCR	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME	System
H'C5	MDCR	—	—	—	—	—	—	MDS1	MDS0	control
H'C6	ISCR	IRQ_7SC	IRQ_6SC	IRQ_5SC	IRQ_4SC	IRQ_3SC	IRQ_2SC	IRQ_1SC	IRQ_0SC	
H'C7	IER	IRQ_7E	IRQ_6E	IRQ_5E	IRQ_4E	IRQ_3E	IRQ_2E	IRQ_1E	IRQ_0E	
H'C8	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR0
H'C9	TCSR	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	
H'CA	TCORA									
H'CB	TCORB									
H'CC	TCNT									
H'CD	—	—	—	—	—	—	—	—	—	
H'CE	—	—	—	—	—	—	—	—	—	
H'CF	—	—	—	—	—	—	—	—	—	
H'D0	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR1
H'D1	TCSR	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	
H'D2	TCORA									
H'D3	TCORB									
H'D4	TCNT									
H'D5	—	—	—	—	—	—	—	—	—	
H'D6	—	—	—	—	—	—	—	—	—	
H'D7	—	—	—	—	—	—	—	—	—	
H'D8	SMR	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	CKS1	CKS0	SCI
H'D9	BRR									
H'DA	SCR	TIE	RIE	TE	RE	—	—	CKE1	CKE0	
H'DB	TDR									
H'DC	SSR	TDRE	RDRF	ORER	FER	PER	—	—	—	
H'DD	RDR									
H'DE	—	—	—	—	—	—	—	—	—	
H'DF	—	—	—	—	—	—	—	—	—	

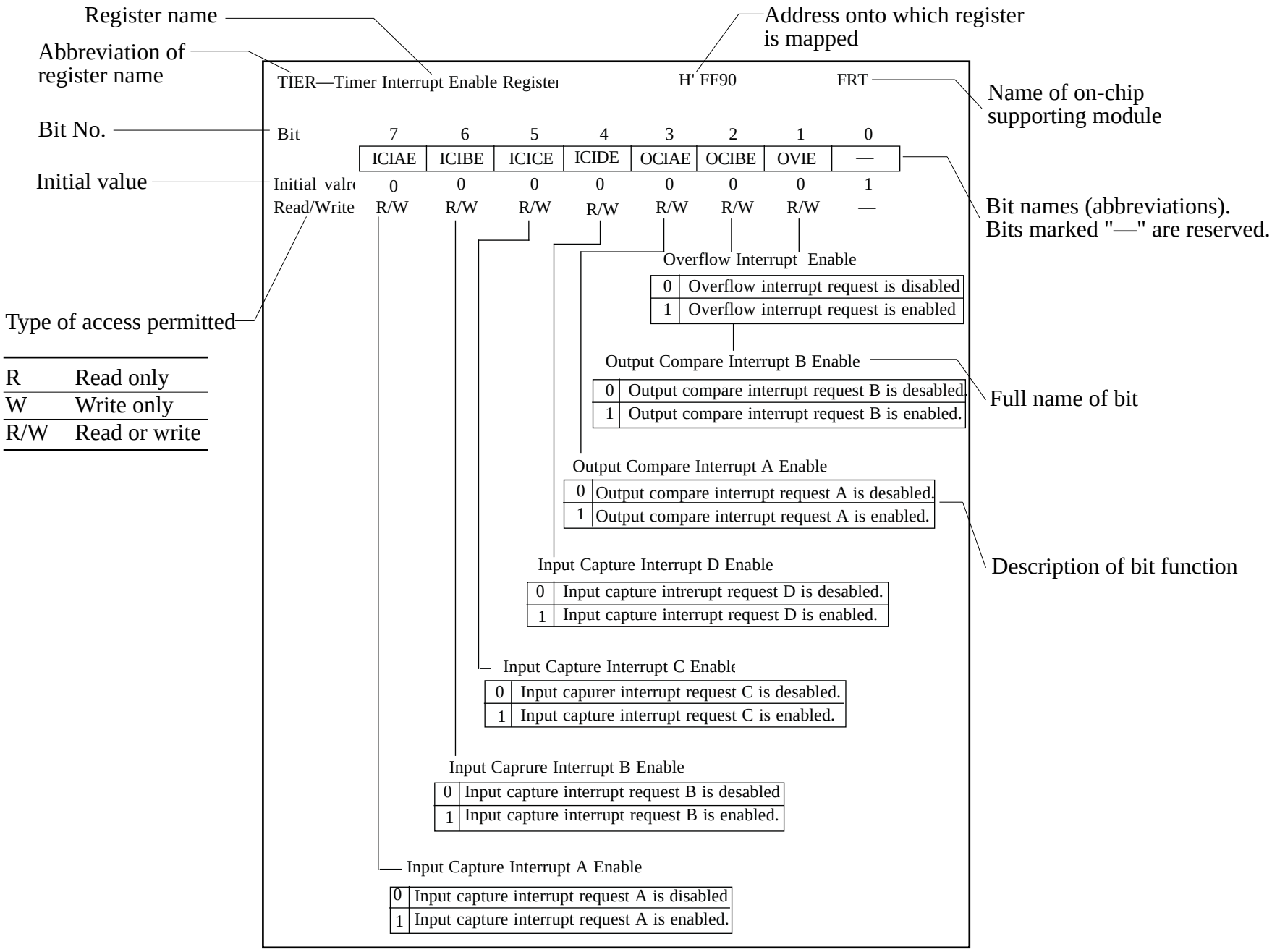
(Continued on next page)

Notes: TMR0: 8-Bit Timer channel 0
TMR1: 8-Bit Timer channel 1
SCI: Serial Communication Interface

(Continued from preceding page)

Addr. (last byte)	Register name	Bit names								
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'E0	ADDRA									A/D
H'E1	—	—	—	—	—	—	—	—	—	
H'E2	ADDRB									
H'E3	—	—	—	—	—	—	—	—	—	
H'E4	ADDRC									
H'E5	—	—	—	—	—	—	—	—	—	
H'E6	ADDRD									
H'E7	—	—	—	—	—	—	—	—	—	
H'E8	ADCSR	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0	
H'E9	—	—	—	—	—	—	—	—	—	
H'EA	ADCR	TRGE	—	—	—	—	—	—	—	
H'EB	—	—	—	—	—	—	—	—	—	
H'EC	—	—	—	—	—	—	—	—	—	
H'ED	—	—	—	—	—	—	—	—	—	
H'EE	—	—	—	—	—	—	—	—	—	
H'EF	—	—	—	—	—	—	—	—	—	
H'F0	PCCSR	MWEF	EMWI	SWEF	EAKAR	MREF	EMRI	MWMF	SWMF	DPRAM
H'F1	PCDR0									
H'F2	PCDR1									
H'F3	PCDR2									
H'F4	PCDR3									
H'F5	PCDR4									
H'F6	PCDR5									
H'F7	PCDR6									
H'F8	PCDR7									
H'F9	PCDR8									
H'FA	PCDR9									
H'FB	PCDR10									
H'FC	PCDR11									
H'FD	PCDR12									
H'FE	PCDR13									
H'FF	PCDR14									

Note: A/D: Analog-to-Digital converter
DPRAM: Dual-port RAM



R	Read only
W	Write only
R/W	Read or write

TIER—Timer Interrupt Enable Register H'FF90 FRT

Bit	7	6	5	4	3	2	1	0
	ICIAE	ICIBE	ICICE	ICIDE	OCIAE	OCIBE	OVIE	—
Initial value	0	0	0	0	0	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—

Overflow Interrupt Enable

0	Overflow interrupt request is disabled.
1	Overflow interrupt request is enabled.

Output Compare Interrupt B Enable

0	Output compare interrupt request B is disabled.
1	Output compare interrupt request B is enabled.

Output Compare Interrupt A Enable

0	Output compare interrupt request A is disabled.
1	Output compare interrupt request A is enabled.

Input Capture Interrupt D Enable

0	Input capture interrupt request D is disabled.
1	Input capture interrupt request D is enabled.

Input Capture Interrupt C Enable

0	Input capture interrupt request C is disabled.
1	Input capture interrupt request C is enabled.

Input Capture Interrupt B Enable

0	Input capture interrupt request B is disabled.
1	Input capture interrupt request B is enabled.

Input Capture Interrupt A Enable

0	Input capture interrupt request A is disabled.
1	Input capture interrupt request A is enabled.

Bit	7	6	5	4	3	2	1	0
	ICFA	ICFB	ICFC	ICFD	OCFA	OCFB	OVF	CCLRA
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/W

Counter Clear A

0	FRC count is not cleared.
1	FRC count is cleared by compare-match A.

Timer Overflow Flag

0	Cleared when CPU reads OVF = "1," then writes "0" in OVF.
1	Set when FRC changes from H'FFFF to H'0000.

Output Compare Flag B

0	Cleared when CPU reads OCFB = "1", then writes "0" in OCFB.
1	Set when FRC = OCRB.

Output Compare Flag A

0	Cleared when CPU reads OCFA = "1", then writes "0" in OCFA.
1	Set when FRC = OCRA.

Input Capture Flag D

0	Cleared when CPU reads ICFD = "1", then writes "0" in ICFD.
1	Set by FTID input.

Input Capture Flag C

0	Cleared when CPU reads ICFC = "1", then writes "0" in ICFC.
1	Set by FTIC input.

Input Capture Flag B

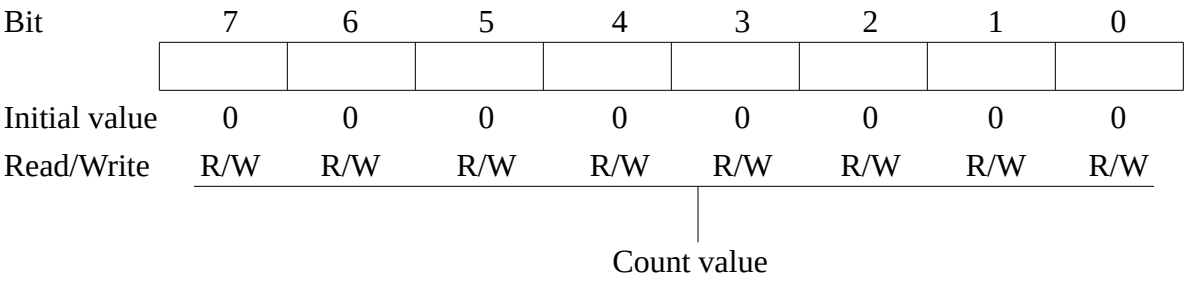
0	Cleared when CPU reads ICFB = "1", then writes "0" in ICFB.
1	Set when FTIB input causes FRC to be copied to ICRB.

Input Capture Flag A

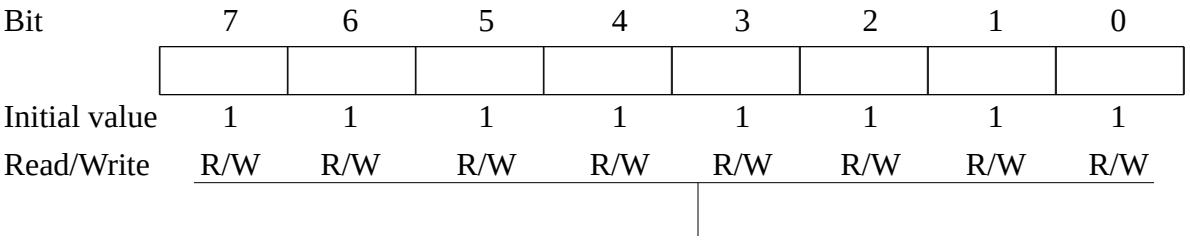
0	Cleared when CPU reads ICFA = "1", then writes "0" in ICFA.
1	Set when FTIA input causes FRC to be copied to ICRA.

* Software can write a "0" in bits 7 to 1 to clear the flags, but cannot write a "1" in these bits

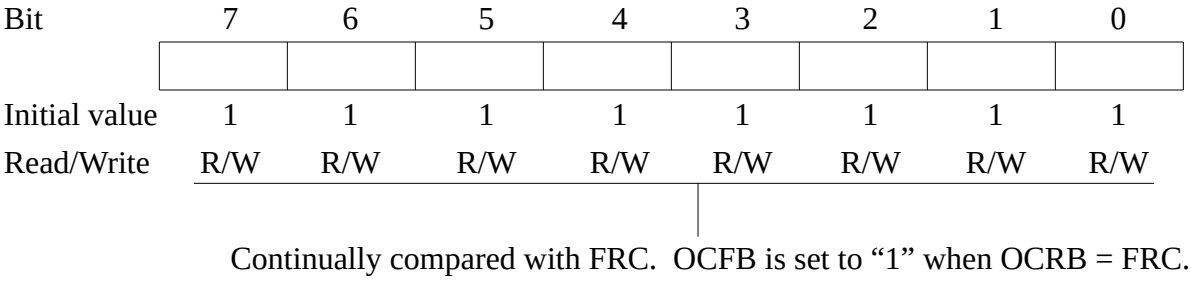
FRC (H and L)—Free-Running Counter H’FF92, H’FF93 FRT



OCRA (H and L)—Output Compare Register A H’FF94, H’FF95 FRT



OCRB (H and L)—Output Compare Register B H’FF94, H’FF95 FRT



Bit	7	6	5	4	3	2	1	0
	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock Select

0	0	Internal clock source: Ø/2
0	1	Internal clock source: Ø/8
1	0	Internal clock source: Ø/32
1	1	External clock source: counted on rising edge

Buffer Enable B

0	ICRD is used for input capture D.
1	ICRD is buffer register for input capture B.

Buffer Enable A

0	ICRC is used for input capture C.
1	ICRC is buffer register for input capture A.

Input Edge Select D

0	Falling edge of FTID is valid.
1	Rising edge of FTID is valid.

Input Edge Select C

0	Falling edge of FTIC is valid.
1	Rising edge of FTIC is valid.

Input Edge Select B

0	Falling edge of FTIB is valid.
1	Rising edge of FTIB is valid.

Input Edge Select A

0	Falling edge of FTIA is valid.
1	Rising edge of FTIA is valid.

TOCR—Timer Output Control Register H'FF97 FRT

Bit	7	6	5	4	3	2	1	0
	—	—	—	OCRS	OEA	OEB	OLVLA	OLVLB
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

Output Level B

0	Compare-match B causes "0" output.
1	Compare-match B causes "1" output.

Output Level A

0	Compare-match A causes "0" output.
1	Compare-match A causes "1" output.

Output Enable B

0	Output compare B output is disabled.
1	Output compare B output is enabled.

Output Enable A

0	Output compare A output is disabled.
1	Output compare A output is enabled.

Output Compare Register Select

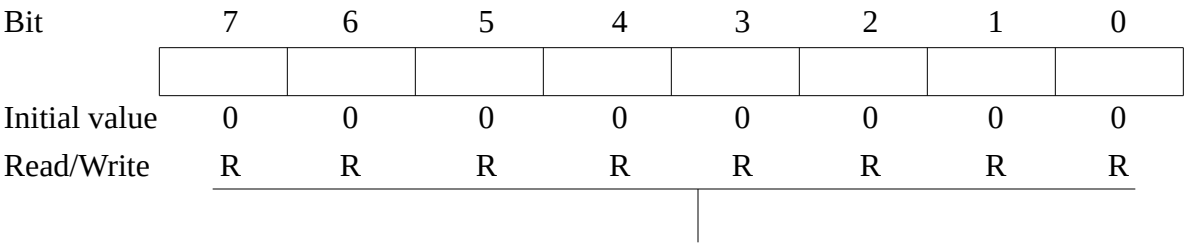
0	The CPU can access OCRA at addresses H'FF94 – H'FF95.
1	The CPU can access OCRB at addresses H'FF94 – H'FF95.

ICRA (H and L)—Input Capture Register H'FF98, H'FF99 FRT

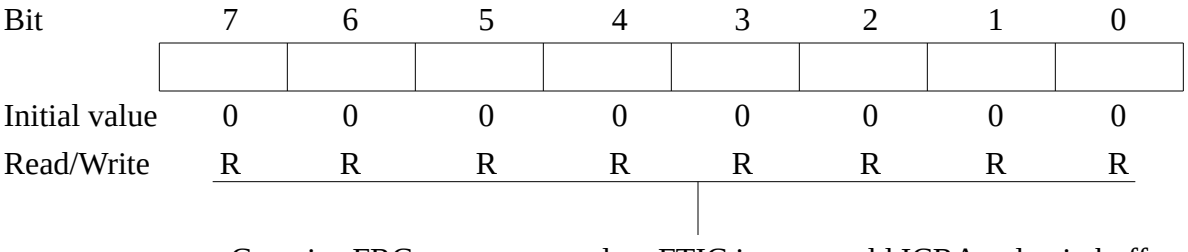
Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Contains FRC count captured on FTIA input.

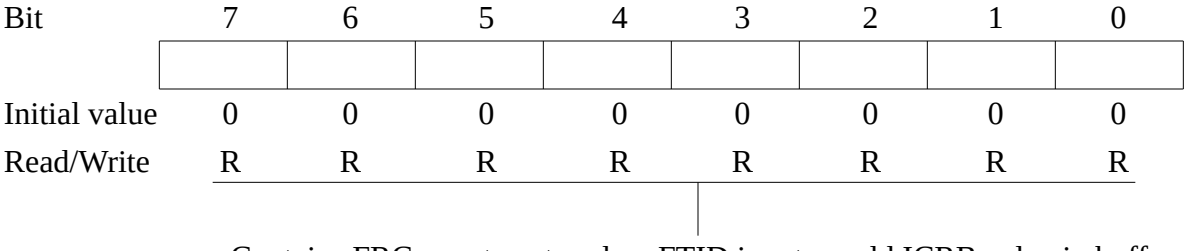
ICRB (H and L)—Input Capture Register H’FF9A, H’FF9B FRT



ICRC (H and L)—Input Capture Register H’FF9C, H’FF9D FRT



ICRD (H and L)—Input Capture Register H’FF9E, H’FF9F FRT



TCR—Timer Control Register **H'FFA0** **PWM0**

Bit	7	6	5	4	3	2	1	0
	OE	OS	—	—	—	CKS2	CKS1	CKS0
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

Clock Select (Values When Ø = 10MHz)

			Internal clock Freq.	Reso-lution	PWM period	PWM frequency
0	0	0	Ø/2	200ns	50µs	20kHz
0	0	1	Ø/8	800ns	200µs	5kHz
0	1	0	Ø/32	3.2µs	800µs	1.25kHz
0	1	1	Ø/128	12.8µs	3.2ms	312.5Hz
1	0	0	Ø/256	25.6µs	6.4ms	156.3Hz
1	0	1	Ø/1024	102.4µs	25.6ms	39.1Hz
1	1	0	Ø/2048	204.8µs	51.2ms	19.5Hz
1	1	1	Ø/4096	409.6µs	102.4ms	9.8Hz

Output Select

0	Positive logic
1	Negative logic

Output Enable

0	PWM output disabled; TCNT cleared to H'00 and stops.
1	PWM output enabled; TCNT runs.

DTR—Duty Register **H'FFA1** **PWM0**

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

Pulse duty factor

TCNT—Timer Counter **H’FFA2** **PWM0**

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

Count value (runs from H’00 to H’F9, then repeats from H’00)

TCR—Timer Control Register **H’FFA4** **PWM1**

Bit	7	6	5	4	3	2	1	0
	OE	OS	—	—	—	CKS2	CKS1	CKS0
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

Note: Bit functions are the same as for PWM0.

DTR—Duty Register **H’FFA5** **PWM1**

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

Note: Bit functions are the same as for PWM0.

TCNT—Timer Counter **H'FFA6** **PWM1**

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

Note: Bit functions are the same as for PWM0.

P1DDR—Port 1 Data Direction Register **H'FFB0** **Port 1**

Bit	7	6	5	4	3	2	1	0
	$P1_7DDR$	$P1_6DDR$	$P1_5DDR$	$P1_4DDR$	$P1_3DDR$	$P1_2DDR$	$P1_1DDR$	$P1_0DDR$
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 1 Input/Output Control

0	Input port
1	Output port

P1DR—Port 1 Data Register **H'FFB2** **Port 1**

Bit	7	6	5	4	3	2	1	0
	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P2DDR—Port 2 Data Direction Register **H'FFB1** **Port 2**

Bit	7	6	5	4	3	2	1	0
	$P2_7$DDR	$P2_6$DDR	$P2_5$DDR	$P2_4$DDR	$P2_3$DDR	$P2_2$DDR	$P2_1$DDR	$P2_0$DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 2 Input/Output Control

0	Input port
1	Output port

P2DR—Port 2 Data Register **H'FFB3** **Port 2**

Bit	7	6	5	4	3	2	1	0
	$P2_7$	$P2_6$	$P2_5$	$P2_4$	$P2_3$	$P2_2$	$P2_1$	$P2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3DDR—Port 3 Data Direction Register **H'FFB4** **Port 3**

Bit	7	6	5	4	3	2	1	0
	$P3_7$DDR	$P3_6$DDR	$P3_5$DDR	$P3_4$DDR	$P3_3$DDR	$P3_2$DDR	$P3_1$DDR	$P3_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 3 Input/Output Control

0	Input port
1	Output port

P3DR—Port 3 Data Register **H'FFB6** **Port 3**

Bit	7	6	5	4	3	2	1	0
	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P4DDR—Port 4 Data Direction Register **H'FFB5** **Port 4**

Bit	7	6	5	4	3	2	1	0
	$P4_7$DDR	$P4_6$DDR	$P4_5$DDR	$P4_4$DDR	$P4_3$DDR	$P4_2$DDR	$P4_1$DDR	$P4_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 4 Input/Output Control

0	Input port
1	Output port

P4DR—Port 4 Data Register **H'FFB7** **Port 4**

Bit	7	6	5	4	3	2	1	0
	$P4_7$	$P4_6$	$P4_5$	$P4_4$	$P4_3$	$P4_2$	$P4_1$	$P4_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P5DDR—Port 5 Data Direction Register **H'FFB8** **Port 5**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$P5_2$DDR	$P5_1$DDR	$P5_0$DDR
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	W	W	W

Port 5 Input/Output Control

0	Input port
1	Output port

P5DR—Port 5 Data Register **H'FFBA** **Port 5**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$P5_2$	$P5_1$	$P5_0$
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

P6DDR—Port 6 Data Direction Register **H'FFB9** **Port 6**

Bit	7	6	5	4	3	2	1	0
	$P6_7$DDR	$P6_6$DDR	$P6_5$DDR	$P6_4$DDR	$P6_3$DDR	$P6_2$DDR	$P6_1$DDR	$P6_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 6 Input/Output Control

0	Input port
1	Output port

P6DR—Port 6 Data Register **H'FFBB** **Port 6**

Bit	7	6	5	4	3	2	1	0
	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P7DR—Port 7 Data Register **H'FFBE** **Port 7**

Bit	7	6	5	4	3	2	1	0
	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$
Initial value	*	*	*	*	*	*	*	*
Read/Write	R	R	R	R	R	R	R	R

* Depends on the levels of pins $P7_7$ to $P7_0$.

P8DDR—Port 8 Data Direction Register **H'FFBD** **Port 8**

Bit	7	6	5	4	3	2	1	0
	—	P8$_6$DDR	P8$_5$DDR	P8$_4$DDR	P8$_3$DDR	P8$_2$DDR	P8$_1$DDR	P8$_0$DDR
Initial value								
Modes 1 and 2	1	0	0	0	0	0	0	1
Mode 3	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

Port 8 Input/Output Control

0	Input port
1	Output port

P8DR—Port 8 Data Register **H'FFBF** **Port 8**

Bit	7	6	5	4	3	2	1	0
	—	P8$_6$	P8$_5$	P8$_4$	P8$_3$	P8$_2$	P8$_1$	P8$_0$
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P9DDR—Port 9 Data Direction Register **H'FFC0** **Port 9**

Bit	7	6	5	4	3	2	1	0
	P9$_7$DDR	P9$_6$DDR	P9$_5$DDR	P9$_4$DDR	P9$_3$DDR	P9$_2$DDR	P9$_1$DDR	P9$_0$DDR
Modes 1 and 2								
Initial value	0	1	0	0	0	0	0	0
Read/Write	W	—	W	W	W	W	W	W
Mode 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 9 Input/Output Control

0	Input port
1	Output port

P9DR—Port 9 Data Register **H'FFC1** **Port 9**

Bit	7	6	5	4	3	2	1	0
	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	$P9_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SYSCR—System Control Register **H'FFC4** **System Control**

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	—	NMIEG	DPME	RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W

RAM Enable

0	On-chip RAM is disabled.
1	On-chip RAM is enabled.

Dual-Port RAM Enable

0	Dual-port RAM is disabled.
1	(1) Single-chip mode: dual-port RAM is enabled. (2) Expanded modes: no effect

NMI Edge

0	Falling edge of $\overline{\text{NMI}}$ is detected.
1	Rising edge of $\overline{\text{NMI}}$ is detected.

Standby Timer Select

0	0	0	Clock settling time = 8192 states
0	0	1	Clock settling time = 16384 states
0	1	0	Clock settling time = 32768 states
0	1	1	Clock settling time = 65536 states
1	–	–	Clock settling time = 131072 states

Software Standby

0	SLEEP instruction causes transition to sleep mode.
1	SLEEP instruction causes transition to software standby mode.

MDCR—Mode Control Register **H'FFC5** **System Control**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	MDS1	MDS0
Initial value	1	1	1	0	0	1	*	*
Read/Write	—	—	—	—	—	—	R	R

Mode Select Bits

Value at mode pins.

* Determined by inputs at pins MD_1 and MD_0.

ISCR—IRQ Sense Control Register **H'FFC6** **System Control**

Bit	7	6	5	4	3	2	1	0
	IRQ_7SC	IRQ_6SC	IRQ_5SC	IRQ_4SC	IRQ_3SC	IRQ_2SC	IRQ_1SC	IRQ_0SC
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IRQ_0 to IRQ_7 Sense Control

0	IRQ_i is level-sensed (active Low).
1	IRQ_i is edge-sensed (falling edge).

IER—IRQ Enable Register **H'FFC7** **System Control**

Bit	7	6	5	4	3	2	1	0
	IRQ_7E	IRQ_6E	IRQ_5E	IRQ_4E	IRQ_3E	IRQ_2E	IRQ_1E	IRQ_0E
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IRQ_0 to IRQ_7 Enable

0	IRQ_i is disabled.
1	IRQ_i is enabled.

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock Select

0	0	0	No clock source; timer stops.
0	0	1	Internal clock source: Ø/8, counted on falling edge.
0	1	0	Internal clock source: Ø/64, counted on falling edge.
0	1	1	Internal clock source: Ø/1024, counted on falling edge.
1	0	0	No clock source; timer stops.
1	0	1	External clock source, counted on rising edge.
1	1	0	External clock source, counted on falling edge.
1	1	1	External clock source, counted on both rising and falling edges.

Counter Clear

0	0	Counter is not cleared.
0	1	Cleared by compare-match A.
1	0	Cleared by compare-match B.
1	1	Cleared on rising edge of external reset input.

Timer Overflow Interrupt Enable

0	Overflow interrupt request is disabled.
1	Overflow interrupt request is enabled.

Compare-Match Interrupt Enable A

0	Compare-match A interrupt request is disabled.
1	Compare-match A interrupt request is enabled.

Compare-Match Interrupt Enable B

0	Compare-match B interrupt request is disabled.
1	Compare-match B interrupt request is enabled.

TCSR—Timer Control/Status Register H'FFC9 TMR0

Bit	7	6	5	4	3	2	1	0
	CMFB	CMFA	OVF	—	OS3*2	OS2*2	OS1*2	OS0*2
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/(W)*1	R/(W)*1	R/(W)*1	—	R/W	R/W	R/W	R/W

Output Select

0	0	No change on compare-match A.
0	1	Output "0" on compare-match A.
1	0	Output "1" on compare-match A.
1	1	Invert (toggle) output on compare-match A.

Output Select

0	0	No change on compare-match B.
0	1	Output "0" on compare-match B.
1	0	Output "1" on compare-match B.
1	1	Invert (toggle) output on compare-match B.

Timer Overflow Flag

0	Cleared when CPU reads OVF = "1," then writes "0" in OVF.
1	Set when TCNT changes from H'FF to H'00.

Compare-Match Flag A

0	Cleared when CPU reads CMFA = "1," then writes "0" in CMFA.
1	Set when TCNT = TCORA.

Compare-Match Flag B

0	Cleared from when CPU reads CMFB = "1," then writes "0" in CMFB.
1	Set when TCNT = TCORB.

*1 Software can write a "0" in bits 7 to 5 to clear the flags, but cannot write a "1" in these bits.

*2 When all four bits (OS3 to OS0) are cleared to "0," output is disabled.

TCORA—Time Constant Register A **H'FFCA** **TMR0**

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

The CMFA bit is set to "1" when TCORA = TCNT.

TCORB—Time Constant Register B **H'FFCB** **TMR0**

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

The CMFB bit is set to "1" when TCORB = TCNT.

TCNT—Timer Counter **H'FFCC** **TMR0**

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

Count value

TCR—Timer Control Register **H'FFD0** **TMR1**

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for TMR0.

TCSR—Timer Control/Status Register **H'FFD1** **TMR1**

Bit	7	6	5	4	3	2	1	0
	CMFB	CMFA	OVF	—	OS3*2	OS2*2	OS1*2	OS0*2
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/(W)*1	R/(W)*1	R/(W)*1	—	R/W	R/W	R/W	R/W

Note: Bit functions are the same as for TMR0.

*1 Software can write a "0" in bits 7 to 5 to clear the flags, but cannot write a "1" in these bits.

*2 When all four bits (OS3 to OS0) are cleared to "0," output is disabled.

TCORA—Time Constant Register A **H'FFD2** **TMR1**

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

Note: Bit functions are the same as for TMR0.

TCORB—Time Constant Register B **H'FFD3** **TMR1**

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

Note: Bit functions are the same as for TMR0.

TCNT—Timer Counter **H'FFD4** **TMR1**

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

Note: Bit functions are the same as for TMR0.

Bit	7	6	5	4	3	2	1	0
	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	CKS1	CKS0
Initial value	0	0	0	0	0	1	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	—	R/W	R/W

Clock Select

0	0	Ø clock
0	1	Ø/4 clock
1	0	Ø/16 clock
1	1	Ø/64 clock

Stop Bit Length

0	One stop bit
1	Two stop bits

Parity Mode

0	Even parity
1	Odd parity

Parity Enable

0	Transmit: No parity bit added. Receive: Parity bit not checked.
1	Transmit: Parity bit added. Receive: Parity bit checked.

Character Length

0	8-Bit data length
1	7-Bit data length

Communication Mode

0	Asynchronous
1	Synchronous

BRR—Bit Rate Register **H'FFD9** **SCI**

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

Constant that determines the bit rate

SCR—Serial Control Register **H'FFDA** **SCI**

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	—	—	CKE1	CKE0
Initial value	0	0	0	0	1	1	0	0
Read/Write	R/W	R/W	R/W	R/W	—	—	R/W	R/W

Clock Enable 0

0	Asynchronous serial clock not output
1	Asynchronous serial clock output at ASCK pin

Clock Enable 1

0	Internal clock
1	External clock, input at ASCK or CSCK pin

Receive Enable

0	Receive disabled
1	Receive enabled

Transmit Enable

0	Transmit disabled
1	Transmit enabled

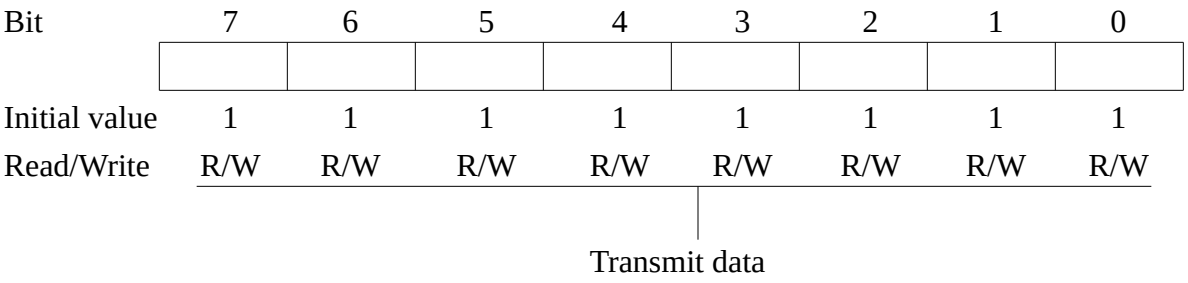
Receive Interrupt Enable

0	Receive interrupt request (RxI) is disabled.
1	Receive interrupt request (RxI) is enabled.

Transmit Interrupt Enable

0	Transmit interrupt request (TxI) is disabled.
1	Transmit interrupt request (TxI) is enabled.

TDR—Transmit Data Register **H’FFDB** **SCI**



SSR—Serial Status Register **H'FFDC** **SCI**

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	—	—	—
Initial value	1	0	0	0	0	1	1	1
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	—	—	—

Parity Error

0	Cleared when CPU reads PER = "1," then writes "0" in PER.
1	Set when a parity error occurs (parity of receive data does not match parity selected by O/E bit).

Framing Error

0	Cleared when CPU reads FER = "1," then writes "0" in FER.
1	Set when a framing error occurs (stop bit is "0").

Overrun Error

0	Cleared when CPU reads ORER = "1," then writes "0" in ORER.
1	Set when an overrun error occurs (next data is completely received while RDRF bit is set to "1").

Receive Data Register Full

0	Cleared when CPU reads RDRF = "1," then writes "0" in RDRF.
1	Set when one character is received normally and transferred from RSR to RDR.

Transmit Data Register Empty

0	Cleared when CPU reads TDRE = "1," then writes "0" in TDRE
1	Set when: 1. Data is transferred from TDR to TSR. 2. TE is cleared while TDRE = "0."

* Software can write a "0" in bits 7 to 3 to clear the flags, but cannot write a "1" in these bits.

RDR—Receive Data Register **H'FFDD** **SCI**

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Receive data

ADDRn—A/D Data Register n (n = A, B, C, D) **A/D**

H’FFE0, H’FFE2, H’FFE4, H’FFE6

Bit	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

A/D conversion result

Note: The least significant bit of the register address is ignored.

ADCSR—A/D Control/Status Register H'FFE8 A/D

Bit	7	6	5	4	3	2	1	0
	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Channel Select

CH2	CH1	CH0	Single mode	Scan mode
0	0	0	AN_0	AN_0
	0	1	AN_1	AN_0, AN_1
	1	0	AN_2	AN_0 to AN_2
	1	1	AN_3	AN_0 to AN_3
1	0	0	AN_4	AN_4
	0	1	AN_5	AN_4, AN_5
	1	0	AN_6	AN_4 to AN_6
	1	1	AN_7	AN_4 to AN_7

Clock Select

0	Conversion time = 242 states (max)
1	Conversion time = 122 states (max)

Scan Mode

0	Single mode
1	Scan mode

A/D Start

0	A/D conversion is halted.
1	1. Single mode: One A/D conversion is performed, then this bit is automatically cleared to "0." 2. Scan mode: A/D conversion starts and continues cyclically on all selected channels until "0" is written in this bit.

A/D Interrupt Enable

0	The A/D interrupt request (ADI) is disabled.
1	The A/D interrupt request (ADI) is enabled.

A/D End Flag

0	Cleared from "1" to "0" when CPU reads ADF = "1," then writes "0" in ADF.
1	Set to "1" at the following times: 1. Single mode: at the completion of A/D conversion 2. Scan mode: when all selected channels have been converted.

* Software can write a "0" in bit 7 to clear the flag, but cannot write a "1" in this bit.

Bit	7	6	5	4	3	2	1	0
	TRGE	—	—	—	—	—	—	—
Initial value	0	1	1	1	1	1	1	1
Read/Write	R/W	—	—	—	—	—	—	—

Trigger Enable

0	ADTRG is disabled.
1	ADTRG is enabled. A/D conversion can be started by external trigger, or by software.

PCCSR—Parallel Communication Control/Status Register H'FFF0 DPRAM

Bit	7	6	5	4	3	2	1	0
	MWEF	EMWI	SWEF	EAKAR	MREF	EMRI	MWMF	SWMF
Initial value	0	0	0	0	0	0	0	0
Read/Write								
H8/300 CPU:	R	R/W	R	R/W	R	R/W	R	R
Master CPU:	R	R	R	R/W	R	R	R	R

Master Write End Flag

0	H8/300 CPU has read PCDR14 while MWMF = "1."
1	Master CPU has written data in PCDR14.

Enable Master Write Interrupt

0	Master write end interrupt (MWEI) is disabled.
1	Master write end interrupt (MWEI) is enabled.

Slave Write End Flag

0	Master CPU has read PCDR14.
1	H8/300 CPU has written data in PCDR14.

Enable Acknowledge and Request

0	$\overline{\text{RDY}}$ output is disabled. Remains in high-impedance state.
1	$\overline{\text{RDY}}$ output is enabled.

Master Read End Flag

0	Cleared when H8/300 CPU reads or writes PCDR0, or master CPU writes to PCDR0.
1	Set when master CPU reads PCDR0.

Enable Master Read Interrupt

0	Master read end interrupt (MREI) is disabled.
1	Master read end interrupt (MREI) is enabled.

Master Write Mode Flag

0	Not master write mode. Cleared when H8/300 CPU reads PCDR0.
1	Master write mode. Set if the master CPU writes to PCDR0 while SWMF = "0."

Slave Write Mode Flag

0	Not slave write mode. Cleared when master CPU reads PCDR0.
1	Slave write mode. Set if H8/300 CPU writes to PCDR0 while MWMF = "0."

PCDR0A—Parallel Communication Data Register 0A H’FFF1 DPRAM

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	—	—	—
Read/Write								
H8/300 CPU:	R	R	R	R	R	R	R	R
Master CPU:	W	W	W	W	W	W	W	W

PCDR0B—Parallel Communication Data Register 0B H’FFF1 DPRAM

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	—	—	—
Read/Write								
H8/300 CPU:	W	W	W	W	W	W	W	W
Master CPU:	R	R	R	R	R	R	R	R

PCDR1 to PCDR14—Parallel Communication Data Registers 1 to 14 H’FFF2 – H’FFFF DPRAM

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	—	—	—
Read/Write								
H8/300 CPU:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Master CPU:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Appendix C. Pin States

C.1 Pin States in Each Mode

Table C-1. Pin States

Pin name	MCU mode	Reset	Hardware standby	Software standby	Sleep mode	Normal operation
$P1_7$ – $P1_0$	1	Low	3-State	Low	Prev. state	Addr. output
A_7 – A_0	2	3-State		Low if DDR = 1, Prev. state if DDR = 0	(Addr. output pins: last address accessed)	Addr. output or input port
	3			Prev. state		I/O port
$P2_7$ – $P2_0$	1	Low	3-State	Low	Prev. state	Addr. output
A_{15} – A_8	2	3-State		Low if DDR = 1, Prev. state if DDR = 0	(Addr. output pins: last address accessed)	Addr. output or input port
	3			Prev. state		I/O port
$P3_7$ – $P3_0$	1	3-State	3-State	3-state	3-State	D_7 – D_0
D_7 – D_0	2					
	3			Prev. state	Prev. state	I/O port
$P4_7$ – $P4_0$,	1	3-State	3-State	Prev. state (note 3)	Prev. state	I/O port
	2					
	3					
$P5_2$ – $P5_0$,	1	3-State	3-State	Prev. state (note 3)	Prev. state	I/O port
	2					
	3					
$P6_7$ – $P6_0$,	1	3-State	3-State	Prev. state (note 3)	Prev. state	I/O port
	2					
	3					
$P7_7$ – $P7_0$,	1	3-State	3-State	3-State	3-State	Input port
	2					
	3					
$P8_6$ – $P8_1$,	1	3-State	3-State	Prev. state (note 3)	Prev. state	I/O port
	2					
	3					

Table C-1. Pin States (cont.)

Pin name	MCU mode	Reset	Hardware standby	Software standby	Sleep mode	Normal operation
$P8_0$/E	1	E clock	3-State	Low if	E clock if	E clock if
	2	output		DDR = 1,	DDR = 1,	DDR = 1,
				3-state if	3-state if	3-state if
				DDR = 0	DDR = 0	DDR = 0
	3	3-State		Prev. state	Prev. state	I/O port
$P9_7$/$\overline{WAIT}$	1	3-State	3-State	3-State	3-State	$\overline{WAIT}$
	2					
	3			Prev. state	Prev. state	I/O port
$P9_6$/Ø	1	Clock	3-state	High	Clock	Clock
	2	output			output	output
	3	3-State		High if	Clock output	Clock output
				DDR = 1,	if DDR = 1,	if DDR = 1,
				3-state if	3-state if	input port if
				DDR = 0	DDR = 0	DDR = 0
$P9_5$ – $P9_3$,	1	High	3-State	High	High	$\overline{AS}$, $\overline{WR}$,
$\overline{AS}$, $\overline{WR}$, $\overline{RD}$	2					$\overline{RD}$
	3	3-State		Prev. state	Prev. state	I/O port
$P9_2$ – $P9_0$,	1	3-State	3-State	Prev. state	Prev. state	I/O port
	2					
	3					

Notes:

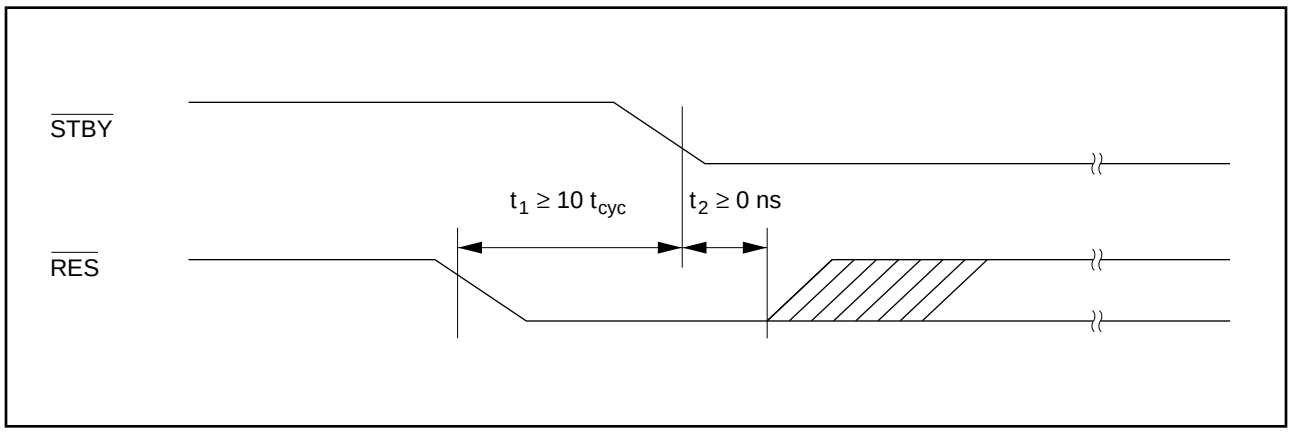
1. 3-state: High-impedance state
2. Prev. state: Previous state. Input ports are in the high-impedance state (with the MOS pull-up on if DDR = 0 and DR = 1). Output ports hold their previous output level.
3. On-chip supporting modules are initialized, so these pins revert to I/O ports according to the DDR and DR bits.
4. I/O port: Direction depends on the data direction (DDR) bit. Note that these pins may also be used by the on-chip supporting modules.

See section 5, “I/O Ports” for further information.

Appendix D. Timing of Transition to and Recovery From Hardware Standby Mode

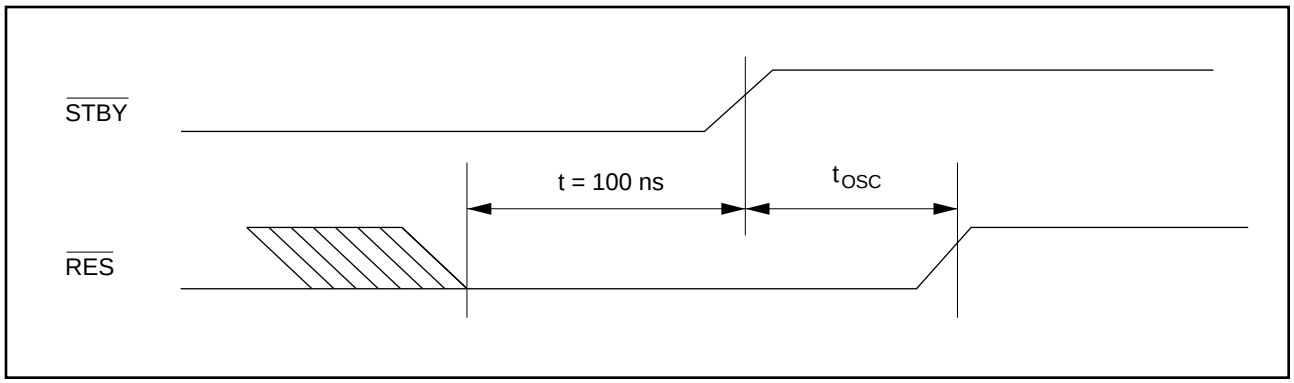
Timing of Transition to Hardware Standby Mode

(1). To retain RAM contents, drive the $\overline{\text{RES}}$ signal low 10 system clock cycles before the $\overline{\text{STBY}}$ signal goes low, as shown below. $\overline{\text{RES}}$ must remain low until $\overline{\text{STBY}}$ goes low (minimum delay from $\overline{\text{STBY}}$ low to $\overline{\text{RES}}$ high: 0 ns).



(2). When it is not necessary to retain RAM contents, RES does not have to be driven low as in (1).

Timing of Recovery From Hardware Standby Mode: Drive the $\overline{\text{RES}}$ signal low approximately 100 ns before $\overline{\text{STBY}}$ goes high.



Appendix E. Package Dimensions

Figure E-1 shows the dimensions of the CG-84 package. Figure E-2 shows the dimensions of the CP-84 package. Figure E-3 shows the dimensions of the FP-80A package.

Unit: mm

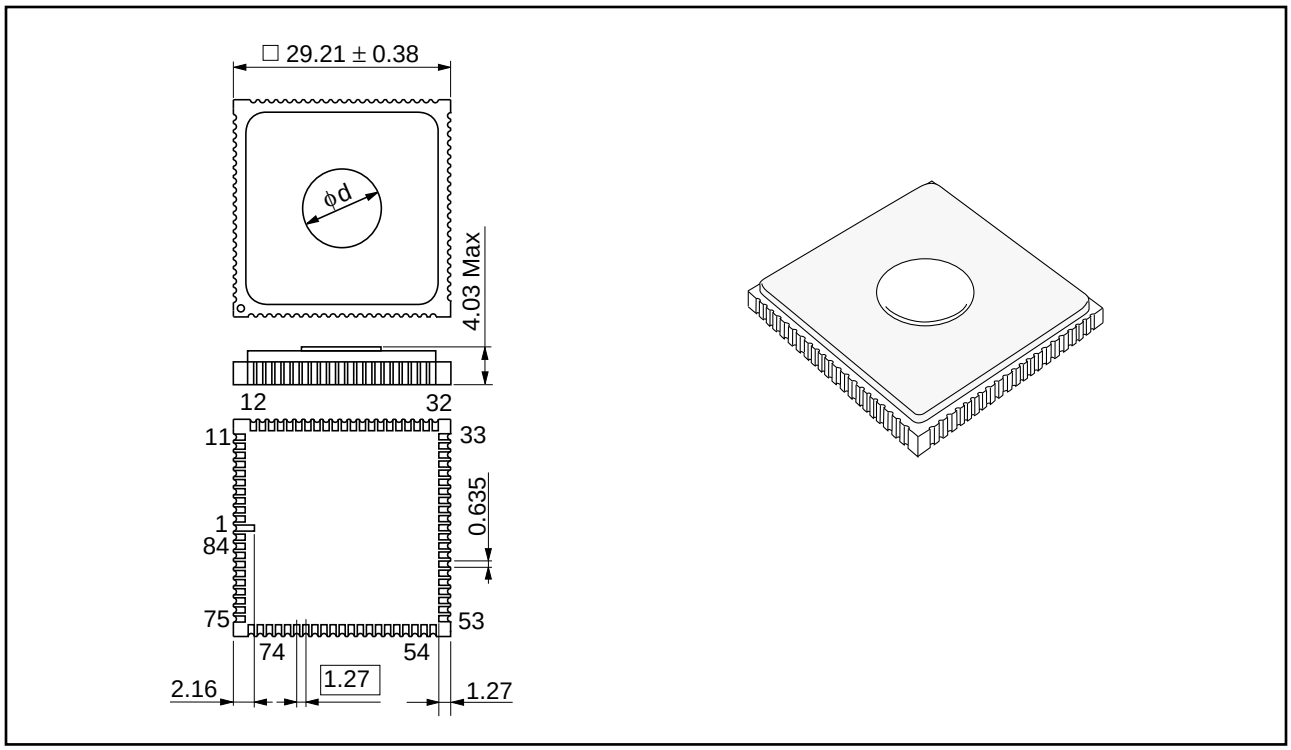


Figure E-1. Package Dimensions (CG-84)

Unit: mm

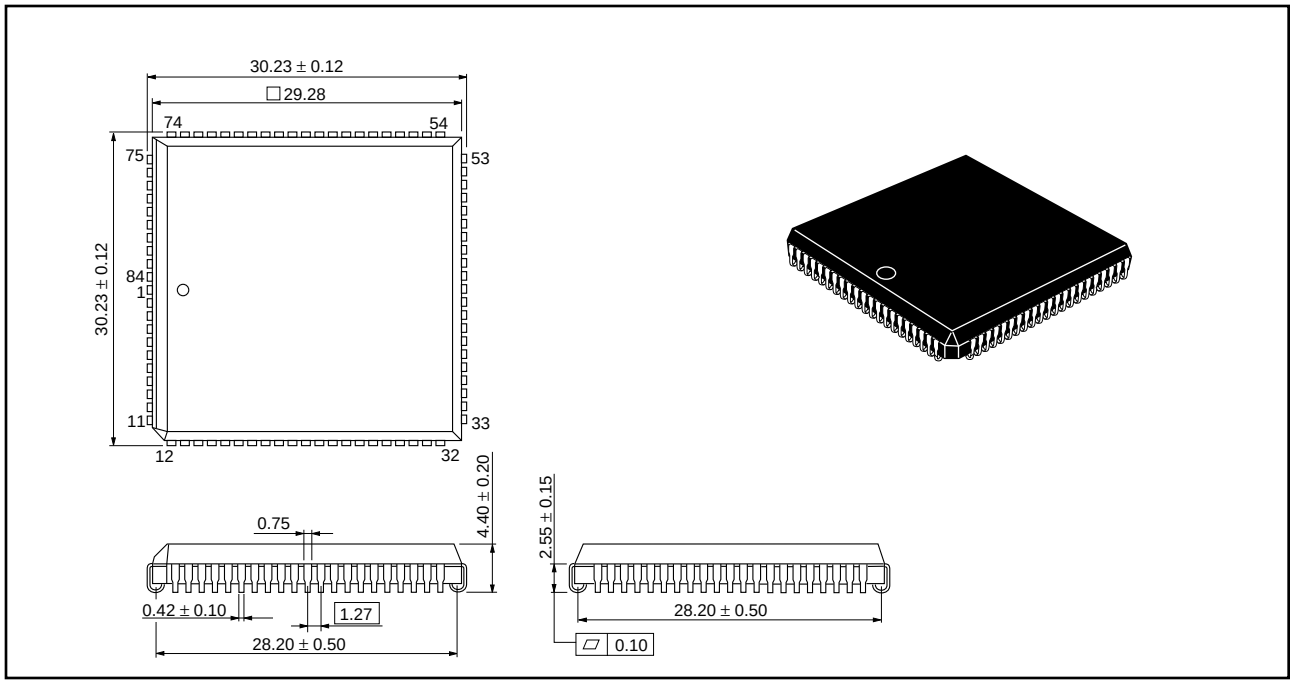


Figure E-2. Package Dimensions (CP-84)

Unit: mm

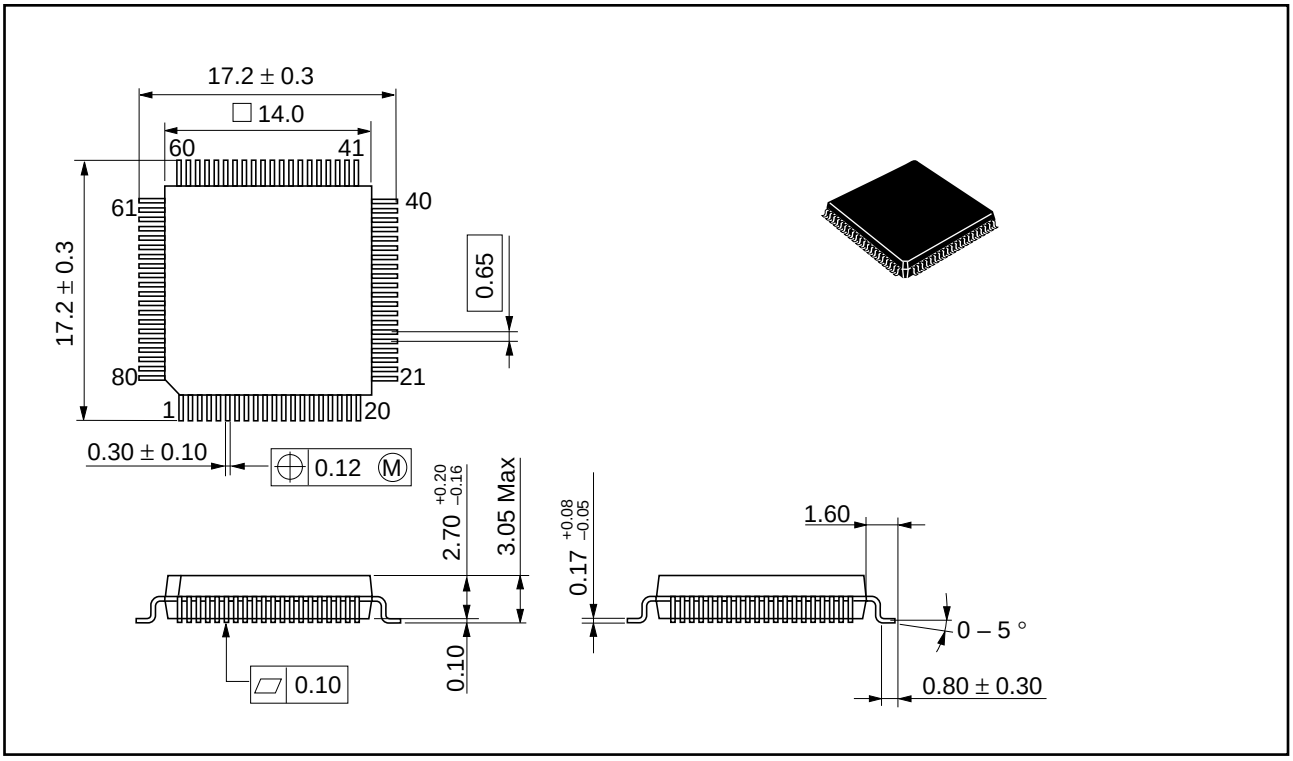


Figure E-3. Package Dimensions (FP-80A)