Hitachi Single-Chip Microcomputer
H8/534, H8/536
HD6475348R, HD6435348R
HD6475368R, HD6435368R
HD6475348S, HD6435348S
HD6475368S, HD6435368S
Hardware Manual

ADE-602-038B

Preface

The H8/534 and H8/536 are high-performance single-chip Hitachi-original microcomputers, featuring a high-speed CPU with 16-bit internal data paths and a full complement of on-chip supporting modules. They are ideal microcontrollers for a wide variety of medium-scale devices, including both office and industrial equipment and consumer products.

The CPU has a general-register architecture. Its instruction set is highly orthogonal and is optimized for fast execution of programs coded in the high-level C language. For further speed, the existing 10-MHz lineup has been extended to include high-speed versions that operate at 16 MHz. Low-voltage versions that operate at 3 V and 2.7 V have also been developed.

On-chip facilities include large RAM and ROM memories, numerous timers, serial I/O, an A/D converter, I/O ports, and other functions for compact implementation of high-performance application systems.

H8/534 and H8/536 are available in both a ZTAT™ version* with on-chip PROM, ideal for the early stages of production or for products with frequently-changing specifications, and a masked-ROM version suitable for volume production.

This manual gives a hardware description of the H8/534 and H8/536. For details of the instruction set, refer to the *H8/500 Series Programming Manual*, which applies to all chips in the H8/500 Series.

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

Contents

Section 13 Watchdog Timer

Section 14 Serial Communication Interface

Section 15 A/D Converter

Section 16 RAM

Section 17 ROM

Figures

Tables

Section 1 Overview

1.1 Features

The H8/534 and H8/536 are CMOS microcomputer units (MCUs) comprising a CPU core plus a full range of supporting functions—an entire system integrated onto a single chip.

The CPU features a highly orthogonal instruction set that permits addressing modes and data sizes to be specified independently in each instruction. An internal 16-bit architecture and 16-bit access to on-chip memory enhance the CPU's data-processing capability and provide the speed needed for realtime control applications.

The on-chip supporting functions include RAM, ROM, timers, a serial communication interface (SCI), A/D conversion, and I/O ports. An on-chip data transfer controller (DTC) can transfer data in either direction between memory and I/O independently of the CPU.

For the on-chip ROM, a choice is offered between masked ROM and programmable ROM (PROM). The PROM version can be programmed by the user with a general-purpose PROM writer.

Table 1-1 lists the main features of the H8/534 and H8/536.

Table 1-1 Features

Feature	Description
CPU	General-register machine • Eight 16-bit general registers • Five 8-bit and two 16-bit control registers High speed • Maximum clock rate: 10 MHz (oscillator frequency: 20 MHz, R-mask versions) 16 MHz (oscillator frequency: 32 MHz, S-mask versions) Expanded operating modes supporting external memory • Minimum mode: up to 64-kbyte address space • Maximum mode: up to 1 M-byte address space Highly orthogonal instruction set • Addressing modes and data size can be specified independently for each instruction 1.5 Addressing modes • Register-register operations • Register-memory operations Instruction set optimized for C language • Special short formats for frequently-used instructions and addressing modes
Memory (H8/534)	• 2-kbyte high-speed RAM on-chip • 32-kbyte programmable or masked ROM on-chip
Memory (H8/536)	• 2-kbyte high-speed RAM on-chip • 62-kbyte programmable or masked ROM on-chip
16-Bit free-running timer (FRT) (3 channels)	Each channel provides: • 1 free-running counter (which can count external events) • 2 output-compare registers • 1 input capture register
8-Bit timer (1 channel)	• One 8-bit up-counter (which can count external events) • 2 time constant registers
PWM timer (3 channels)	• Generates pulses with any duty ratio from 0 to 100% • Resolution: 1/250
Watchdog timer (WDT) (1 channel)	• An overflow generates a nonmaskable interrupt • Can also be used as an interval timer

Table 1-1 Features (cont)

Feature	Description
Serial communication interface (SCI) (2 channels)	• Asynchronous or synchronous mode (selectable) • Full duplex: can send and receive simultaneously • Built-in baud rate generator
A/D converter	• 10-Bit resolution • 8 channels, controllable in single mode or scan mode (selectable) • Sample-and-hold function • Start of A/D conversion can be externally triggered
I/O ports	• 57 Input/output pins (six 8-bit ports, one 5-bit port, one 4-bit port) • 8 Input-only pins (one 8-bit port)
Interrupt controller (INTC)	• 7 external interrupt pins (NMI, $\overline{IRQ0}$, $\overline{IRQ1}$ to $\overline{IRQ5}$) • 23 internal interrupts • 8 priority levels
Data transfer controller (DTC)	Performs bidirectional data transfer between memory and I/O independently of the CPU
Wait-state controller (WSC)	Can insert wait states in access to external memory or I/O
Operating modes	5 MCU operating modes • Expanded minimum modes, supporting up to 64 kbytes external memory with or without using on-chip ROM (Modes 1 and 2) • Expanded maximum modes, supporting up to 1 Mbyte external memory with or without using on-chip ROM (Modes 3 and 4) • Single-chip mode (Mode 7) 3 power-down modes • Sleep mode • Software standby mode • Hardware standby mode
Other features	• E clock output available • Clock generator on-chip

Product line-up (H8/534 R-mask versions)

Model Name	Package Options	ROM
HD6475348RCG	84-Pin windowed LCC (CG-84)	PROM
HD6475348RCP	84-Pin PLCC (CP-84)	
HD6475348RF	80-Pin QFP (FP-80A)	
HD6435348RCP	84-Pin PLCC (CP-84)	Mask
HD6435348RF	80-Pin QFP (FP-80A)	ROM

Product line-up (H8/534 S-mask versions)

Model Name	Package Options	ROM
HD6475348SCG	84-Pin windowed LCC (CG-84)	PROM
HD6475348SCP	84-Pin PLCC (CP-84)	
HD6475348SF	80-Pin QFP (FP-80A)	
HD6475348STF	80-Pin TQFP (TFP-80C)	
HD6435348SCP	84-Pin PLCC (CP-84)	Mask
HD6435348SF	80-Pin QFP (FP-80A)	ROM
HD6435348STF	80-Pin TQFP (TFP-80C)	

Table 1-1 Features (cont)

Feature	Description		
Product line-up (H8/536 R-mask versions)	**Model Name**	**Package Options**	**ROM**
	HD6475368RCG	84-Pin windowed LCC (CG-84)	PROM
	HD6475368RCP	84-Pin PLCC (CP-84)	
	HD6475368RF	80-Pin QFP (FP-80A)	
	HD6435368RCP	84-Pin PLCC (CP-84)	Mask
	HD6435368RF	80-Pin QFP (FP-80A)	ROM
Product line-up (H8/536 S-mask versions)	**Model Name**	**Package Options**	**ROM**
	HD6475368SCG	84-Pin windowed LCC (CG-84)	PROM
	HD6475368SCP	84-Pin PLCC (CP-84)	
	HD6475368SF	80-Pin QFP (FP-80A)	
	HD6475368STF	80-Pin TQFP (TFP-80C)	
	HD6435368SCP	84-Pin PLCC (CP-84)	Mask
	HD6435368SF	80-Pin QFP (FP-80A)	ROM
	HD6435368STF	80-Pin TQFP (TFP-80C)	

Product line-up		**Regular Versions**	**16-MHz High-Speed Versions**	**3-V Low-Voltage Versions***	**2.7-V Low-Voltage Versions***
Model name	PROM	HD6475368R HD6475348R	HD6475368S HD6475348S	HD6475368SV HD6475348SV	HD6475368SV HD6475348SV
	Mask ROM	HD6435368R HD6435348R	HD6435368S HD6435348S	HD6435368SV HD6435348SV	HD6435368SV HD6435348SV
Clock speed Supply voltage		0.5 MHz to 10 MHz 5 V ± 10%	2 MHz to 16 MHz 5 V ± 10%	2 MHz to 10 MHz 3 V to 5.5 V	2 MHz to 8 MHz 2.7 V to 5.5 V

Notes: The product codes of the 3-V and 2.7-V low-voltage versions include a suffix that identifies the clock speed. Examples are shown below for the H8/536 PROM version in an 80-pin QFP package.

Examples: 3-V versions: HD6475368SVF10
2.7-V versions: HD6475368SVF8

***** Under development

1.2 Block Diagram

Figure 1-1 shows a block diagram of the H8/534 and H8/536.

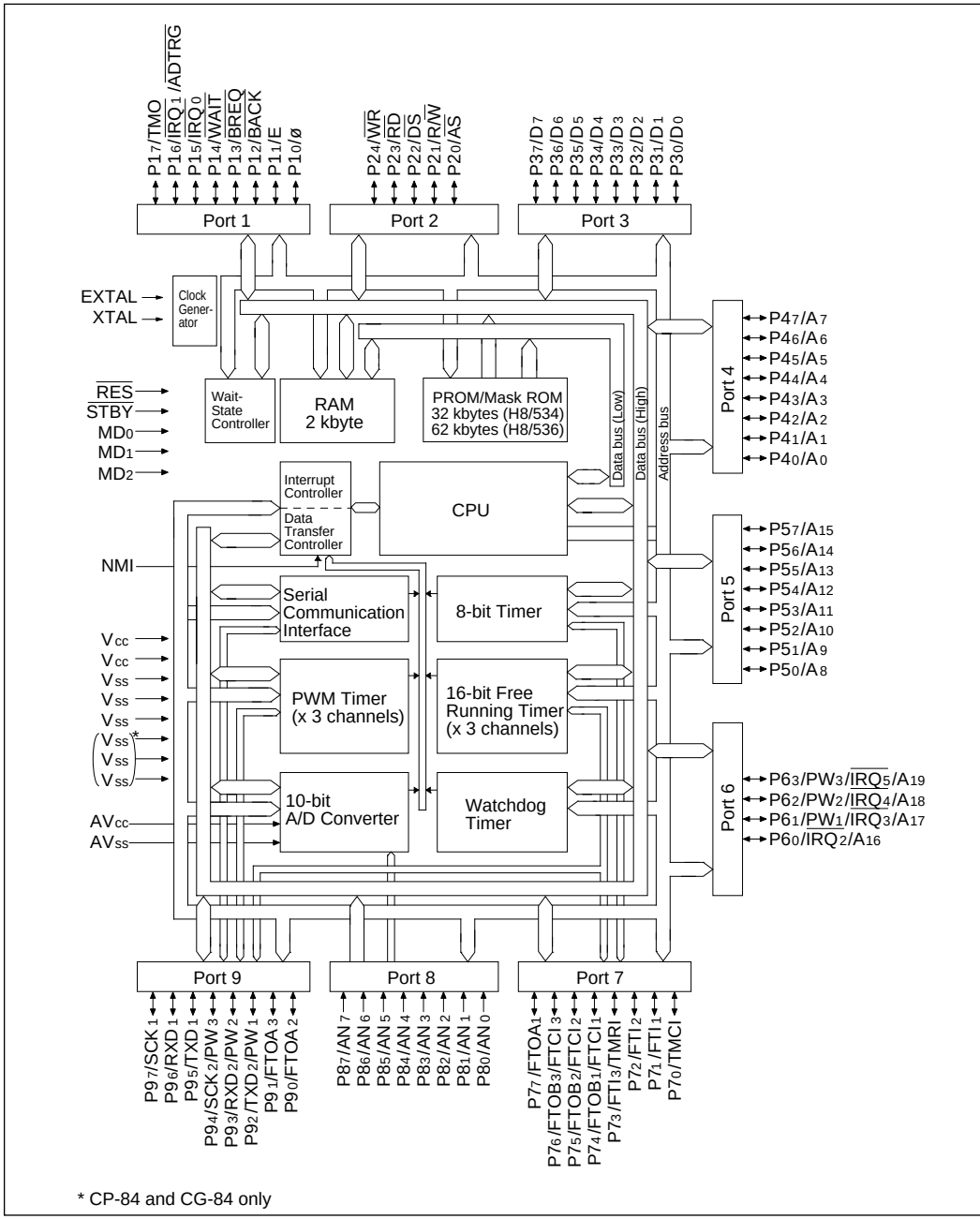


Figure 1-1 Block Diagram

1.3 Pin Arrangements and Functions

1.3.1 Pin Arrangement

Figure 1-2 shows the pin arrangement of the CP-84 package. Figure 1-3 shows the pin arrangement of the CG-84 package. Figure 1-4 shows the pin arrangement of the FP-80A package. These pin arrangements apply to both the H8/534 and H8/536.

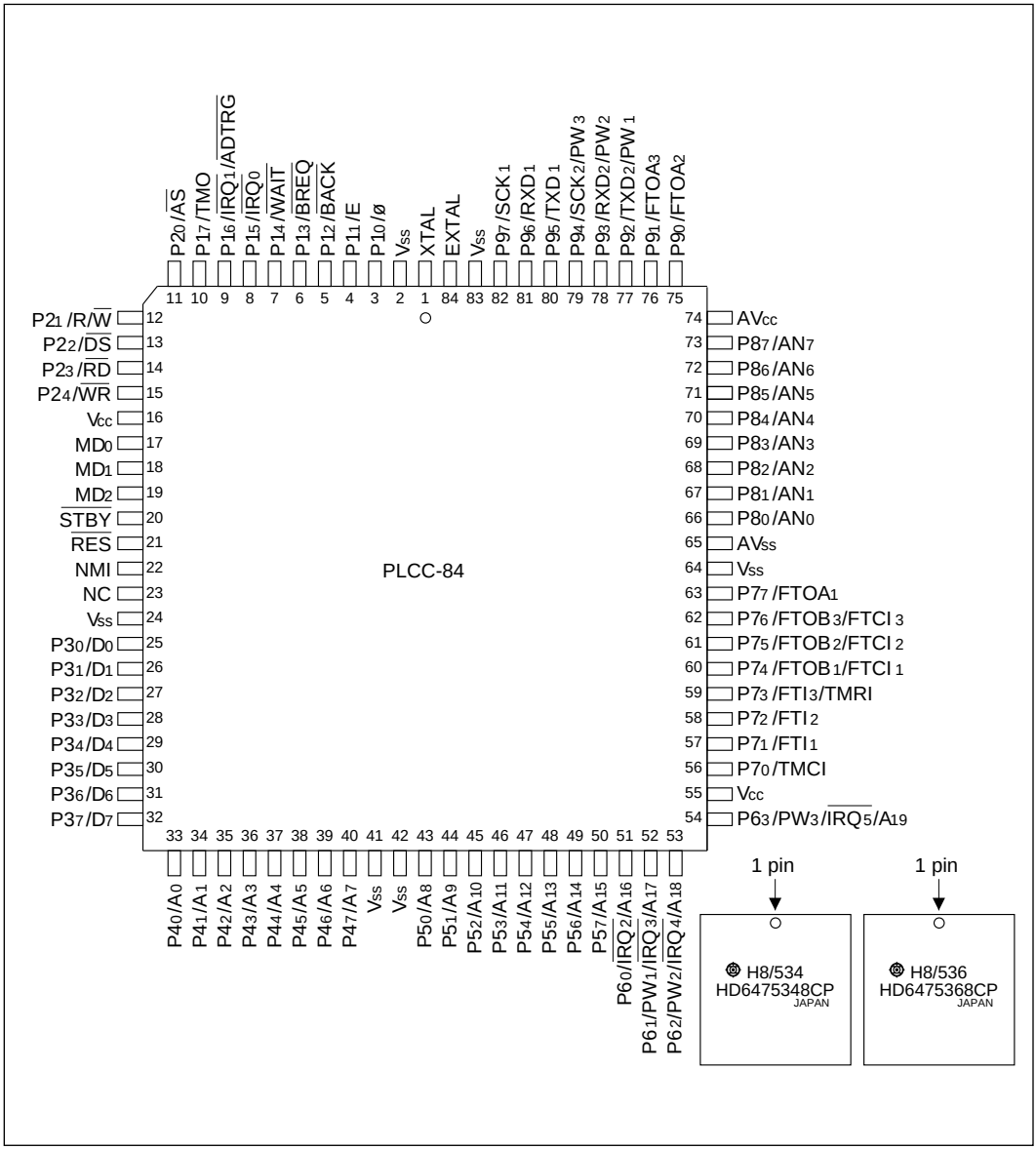


Figure 1-2 Pin Arrangement (CP-84, Top View)

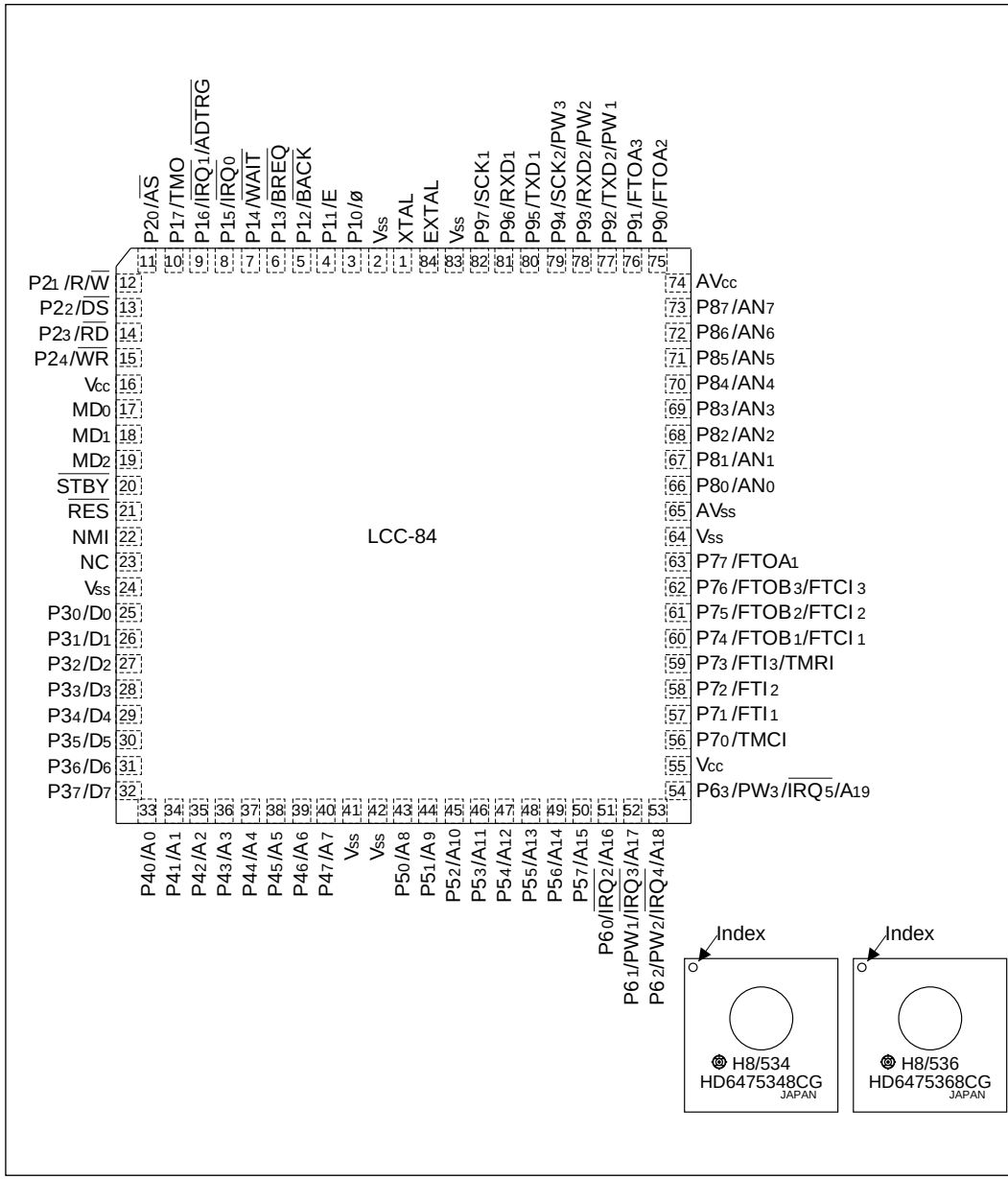


Figure 1-3 Pin Arrangement (CG-84, Top View)

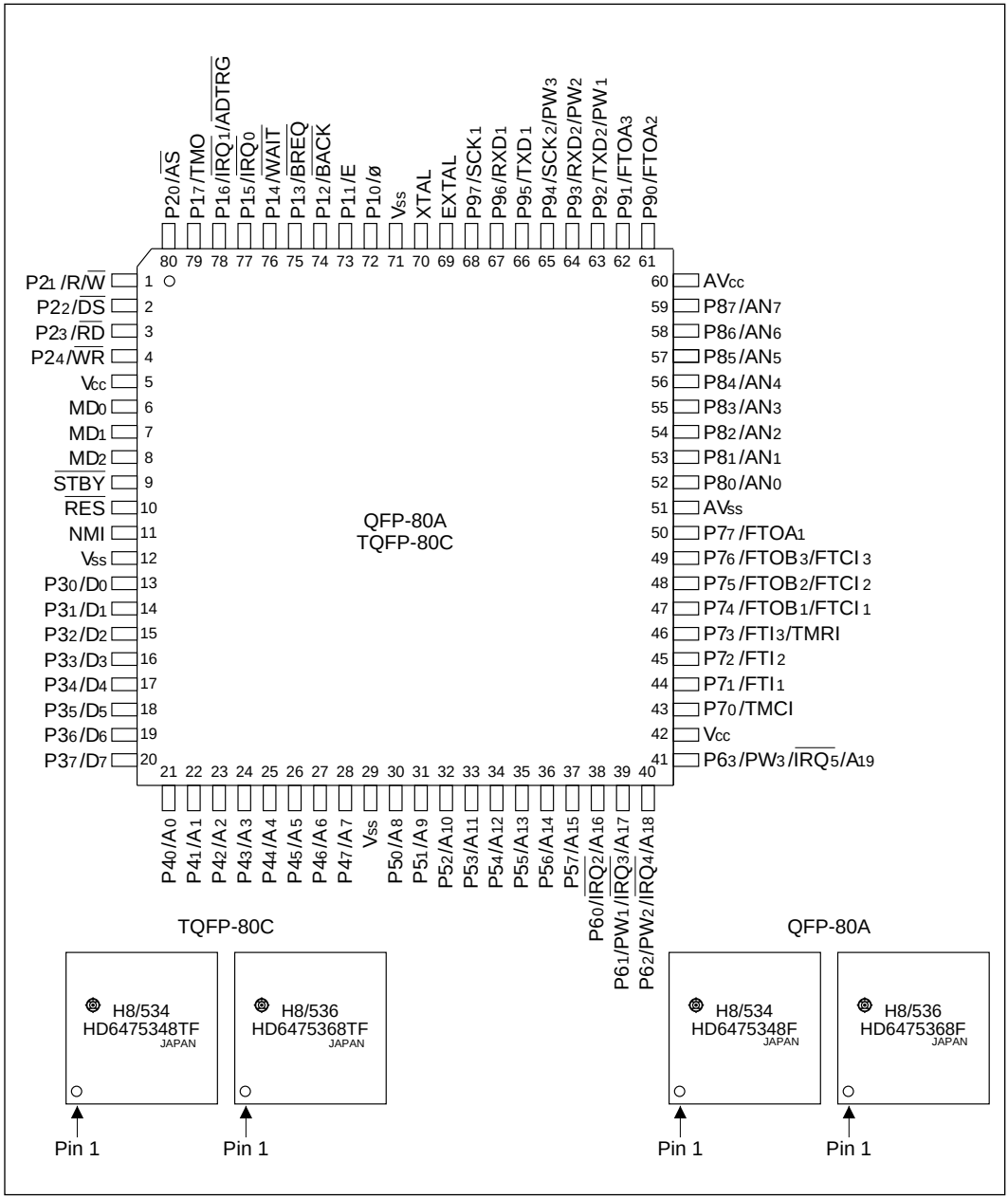


Figure 1-4 Pin Arrangement (FP-80A, TFP-80C, Top View)

1.3.2 Pin Functions

Pin Arrangements in Each Operating Mode: Table 1-2 lists the arrangements of the pins of the CP-84 and CG-84 packages in each operating mode. Table 1-3 lists the arrangements for the FP-80A package.

Table 1-2 Pin Arrangements in Each Operating Mode (CP-84, CG-84)

	Pin Name						
	Expanded Minimum Modes		Expanded Maximum Modes		Single-Chip Mode	PROM Mode	
Pin No.	Mode 1	Mode 2	Mode 3	Mode 4	Mode 7	H8/534	H8/536
1	XTAL	XTAL	XTAL	XTAL	XTAL	NC	NC
2	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
3	$P1_0$/ø	$P1_0$/ø	$P1_0$/ø	$P1_0$/ø	$P1_0$/ø	NC	NC
4	$P1_1$/E	$P1_1$/E	$P1_1$/E	$P1_1$/E	$P1_1$/E	NC	NC
5	$P1_2$ / $\overline{BACK}$	$P1_2$ / $\overline{BACK}$	$P1_2$ / $\overline{BACK}$	$P1_2$ / $\overline{BACK}$	$P1_2$	NC	NC
6	$P1_3$ / $\overline{BREQ}$	$P1_3$ / $\overline{BREQ}$	$P1_3$ / $\overline{BREQ}$	$P1_3$ / $\overline{BREQ}$	$P1_3$	NC	NC
7	$P1_4$ / $\overline{WAIT}$	$P1_4$ / $\overline{WAIT}$	$P1_4$ / $\overline{WAIT}$	$P1_4$ / $\overline{WAIT}$	$P1_4$	NC	A_{15}
8	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	NC	A_{16}
9	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	NC	$\overline{PGM}$
10	$P1_7$ / TMO	$P1_7$ / TMO	$P1_7$ / TMO	$P1_7$ / TMO	$P1_7$ / TMO	NC	NC
11	$\overline{AS}$	$\overline{AS}$	$\overline{AS}$	$\overline{AS}$	$P2_0$	NC	NC
12	R/$\overline{W}$	R/$\overline{W}$	R/$\overline{W}$	R/$\overline{W}$	$P2_1$	NC	NC
13	$\overline{DS}$	$\overline{DS}$	$\overline{DS}$	$\overline{DS}$	$P2_2$	NC	NC
14	$\overline{RD}$	$\overline{RD}$	$\overline{RD}$	$\overline{RD}$	$P2_3$	NC	NC
15	$\overline{WR}$	$\overline{WR}$	$\overline{WR}$	$\overline{WR}$	$P2_4$	NC	NC
16	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}
17	MD_0	MD_0	MD_0	MD_0	MD_0	V_{SS}	V_{SS}
18	MD_1	MD_1	MD_1	MD_1	MD_1	V_{SS}	V_{SS}

Notes: 1. For the PROM mode, see section 17, “ROM.”
2. Pins marked NC should be left unconnected.

Table 1-2 Pin Arrangements in Each Operating Mode (CP-84, CG-84) (cont)

Pin No.	Pin Name: Expanded Minimum Modes		Expanded Maximum Modes		Single-Chip Mode	PROM Mode	
	Mode 1	Mode 2	Mode 3	Mode 4	Mode 7	H8/534	H8/536
19	MD_2	MD_2	MD_2	MD_2	MD_2	V_{SS}	V_{SS}
20	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	V_{SS}	V_{SS}
21	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	V_{PP}	V_{PP}
22	NMI	NMI	NMI	NMI	NMI	A_9	A_9
23	NC	NC	NC	NC	NC	NC	NC
24	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
25	D_0	D_0	D_0	D_0	$P3_0$	O_0	O_0
26	D_1	D_1	D_1	D_1	$P3_1$	O_1	O_1
27	D_2	D_2	D_2	D_2	$P3_2$	O_2	O_2
28	D_3	D_3	D_3	D_3	$P3_3$	O_3	O_3
29	D_4	D_4	D_4	D_4	$P3_4$	O_4	O_4
30	D_5	D_5	D_5	D_5	$P3_5$	O_5	O_5
31	D_6	D_6	D_6	D_6	$P3_6$	O_6	O_6
32	D_7	D_7	D_7	D_7	$P3_7$	O_7	O_7
33	A_0	A_0	A_0	A_0	$P4_0$	A_0	A_0
34	A_1	A_1	A_1	A_1	$P4_1$	A_1	A_1
35	A_2	A_2	A_2	A_2	$P4_2$	A_2	A_2
36	A_3	A_3	A_3	A_3	$P4_3$	A_3	A_3
37	A_4	A_4	A_4	A_4	$P4_4$	A_4	A_4
38	A_5	A_5	A_5	A_5	$P4_5$	A_5	A_5
39	A_6	A_6	A_6	A_6	$P4_6$	A_6	A_6
40	A_7	A_7	A_7	A_7	$P4_7$	A_7	A_7
41	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}

Notes: 1. For the PROM mode, see section 17, “ROM.”
2. Pins marked NC should be left unconnected.

Table 1-2 Pin Arrangements in Each Operating Mode (CP-84, CG-84) (cont)

Pin No.	Pin Name: Expanded Minimum Modes		Expanded Maximum Modes		Single-Chip Mode	PROM Mode	
	Mode 1	Mode 2	Mode 3	Mode 4	Mode 7	H8/534	H8/536
42	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
43	A_8	$P5_0$ / A_8	A_8	$P5_0$ / A_8	$P5_0$	A_8	A_8
44	A_9	$P5_1$ / A_9	A_9	$P5_1$ / A_9	$P5_1$	$\overline{OE}$	$\overline{OE}$
45	A_{10}	$P5_2$ / A_{10}	A_{10}	$P5_2$ / A_{10}	$P5_2$	A_{10}	A_{10}
46	A_{11}	$P5_3$ / A_{11}	A_{11}	$P5_3$ / A_{11}	$P5_3$	A_{11}	A_{11}
47	A_{12}	$P5_4$ / A_{12}	A_{12}	$P5_4$ / A_{12}	$P5_4$	A_{12}	A_{12}
48	A_{13}	$P5_5$ / A_{13}	A_{13}	$P5_5$ / A_{13}	$P5_5$	A_{13}	A_{13}
49	A_{14}	$P5_6$ / A_{14}	A_{14}	$P5_6$ / A_{14}	$P5_6$	A_{14}	A_{14}
50	A_{15}	$P5_7$ / A_{15}	A_{15}	$P5_7$ / A_{15}	$P5_7$	$\overline{CE}$	$\overline{CE}$
51	$P6_0$ / $\overline{IRQ_2}$	$P6_0$ / $\overline{IRQ_2}$	A_{16}	$P6_0$ / $\overline{IRQ_2}$ / A_{16}	$P6_0$ / $\overline{IRQ_2}$	V_{CC}	V_{CC}
52	$P6_1$ / PW_1 / $\overline{IRQ_3}$	$P6_1$ / PW_1 / $\overline{IRQ_3}$	A_{17}	$P6_1$ / $\overline{IRQ_3}$ / A_{17}	$P6_1$ / PW_1 / $\overline{IRQ_3}$	V_{CC}	V_{CC}
53	$P6_2$ / PW_2 / $\overline{IRQ_4}$	$P6_2$ / PW_2 / $\overline{IRQ_4}$	A_{18}	$P6_2$ / $\overline{IRQ_4}$ / A_{18}	$P6_2$ / PW_2 / $\overline{IRQ_4}$	NC	NC
54	$P6_3$ / PW_3 / $\overline{IRQ_5}$	$P6_3$ / PW_3 / $\overline{IRQ_5}$	A_{19}	$P6_3$ / $\overline{IRQ_5}$ / A_{19}	$P6_3$ / PW_3 / $\overline{IRQ_5}$	NC	NC
55	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}
56	$P7_0$ / TMCI	$P7_0$ / TMCI	$P7_0$ / TMCI	$P7_0$ / TMCI	$P7_0$ / TMCI	NC	NC
57	$P7_1$ / FTI_1	$P7_1$ / FTI_1	$P7_1$ / FTI_1	$P7_1$ / FTI_1	$P7_1$ / FTI_1	NC	NC
58	$P7_2$ / FTI_2	$P7_2$ / FTI_2	$P7_2$ / FTI_2	$P7_2$ / FTI_2	$P7_2$ / FTI_2	NC	NC
59	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	NC	NC
60	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	NC	NC
61	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	NC	NC

Notes: 1. For the PROM mode, see section 17, “ROM.”
2. Pins marked NC should be left unconnected.

Table 1-2 Pin Arrangements in Each Operating Mode (CP-84, CG-84) (cont)

Pin No.	Pin Name: Expanded Minimum Modes Mode 1	Expanded Minimum Modes Mode 2	Expanded Maximum Modes Mode 3	Expanded Maximum Modes Mode 4	Single-Chip Mode Mode 7	PROM Mode H8/534	PROM Mode H8/536
62	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$/ $FTOB_3$ / $FTCI_3$	NC	NC
63	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	NC	NC
64	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
65	AV_{SS}	AV_{SS}	AV_{SS}	AV_{SS}	AV_{SS}	V_{SS}	V_{SS}
66	$P8_0$ / AN_0	$P8_0$ / AN_0	$P8_0$ / AN_0	$P8_0$ / AN_0	$P8_0$ / AN_0	NC	NC
67	$P8_1$ / AN_1	$P8_1$ / AN_1	$P8_1$ / AN_1	$P8_1$ / AN_1	$P8_1$ / AN_1	NC	NC
68	$P8_2$ / AN_2	$P8_2$ / AN_2	$P8_2$ / AN_2	$P8_2$ / AN_2	$P8_2$ / AN_2	NC	NC
69	$P8_3$ / AN_3	$P8_3$ / AN_3	$P8_3$ / AN_3	$P8_3$ / AN_3	$P8_3$ / AN_3	NC	NC
70	$P8_4$ / AN_4	$P8_4$ / AN_4	$P8_4$ / AN_4	$P8_4$ / AN_4	$P8_4$ / AN_4	NC	NC
71	$P8_5$ / AN_5	$P8_5$ / AN_5	$P8_5$ / AN_5	$P8_5$ / AN_5	$P8_5$ / AN_5	NC	NC
72	$P8_6$ / AN_6	$P8_6$ / AN_6	$P8_6$ / AN_6	$P8_6$ / AN_6	$P8_6$ / AN_6	NC	NC
73	$P8_7$ / AN_7	$P8_7$ / AN_7	$P8_7$ / AN_7	$P8_7$ / AN_7	$P8_7$ / AN_7	NC	NC
74	AV_{CC}	AV_{CC}	AV_{CC}	AV_{CC}	AV_{CC}	V_{CC}	V_{CC}
75	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	NC	NC
76	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	NC	NC
77	$P9_2$ / TXD_2 / PW_1	$P9_2$ / TXD_2 / PW_1	$P9_2$ / TXD_2 / PW_1	$P9_2$ / TXD_2 / PW_1	$P9_2$ / TXD_2 / PW_1	NC	NC
78	$P9_3$ / RXD_2 / PW_2	$P9_3$ / RXD_2 / PW_2	$P9_3$ / RXD_2 / PW_2	$P9_3$ / RXD_2 / PW_2	$P9_3$ / RXD_2 / PW_2	NC	NC
79	$P9_4$ / SCK_2 / PW_3	$P9_4$ / SCK_2 / PW_3	$P9_4$ / SCK_2 / PW_3	$P9_4$ / SCK_2 / PW_3	$P9_4$ / SCK_2 / PW_3	NC	NC
80	$P9_5$ / TXD_1	$P9_5$ / TXD_1	$P9_5$ / TXD_1	$P9_5$ / TXD_1	$P9_5$ / TXD_1	NC	NC
81	$P9_6$ / RXD_1	$P9_6$ / RXD_1	$P9_6$ / RXD_1	$P9_6$ / RXD_1	$P9_6$ / RXD_1	NC	NC
82	$P9_7$ / SCK_1	$P9_7$ / SCK_1	$P9_7$ / SCK_1	$P9_7$ / SCK_1	$P9_7$ / SCK_1	NC	NC
83	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
84	EXTAL	EXTAL	EXTAL	EXTAL	EXTAL	NC	NC

Notes: 1. For the PROM mode, see section 17, “ROM.”

2. Pins marked NC should be left unconnected.

Table 1-3 Pin Arrangements in Each Operating Mode (FP-80A, TFP-80C)

Pin No.	Pin Name: Expanded Minimum Modes		Expanded Maximum Modes		Single-Chip Mode	PROM Mode	
	Mode 1	Mode 2	Mode 3	Mode 4	Mode 7	H8/534	H8/536
1	R/$\overline{W}$	R/$\overline{W}$	R/$\overline{W}$	R/$\overline{W}$	$P2_1$	NC	NC
2	$\overline{DS}$	$\overline{DS}$	$\overline{DS}$	$\overline{DS}$	$P2_2$	NC	NC
3	$\overline{RD}$	$\overline{RD}$	$\overline{RD}$	$\overline{RD}$	$P2_3$	NC	NC
4	$\overline{WR}$	$\overline{WR}$	$\overline{WR}$	$\overline{WR}$	$P2_4$	NC	NC
5	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}
6	MD_0	MD_0	MD_0	MD_0	MD_0	V_{SS}	V_{SS}
7	MD_1	MD_1	MD_1	MD_1	MD_1	V_{SS}	V_{SS}
8	MD_2	MD_2	MD_2	MD_2	MD_2	V_{SS}	V_{SS}
9	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	$\overline{STBY}$	V_{SS}	V_{SS}
10	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	$\overline{RES}$	V_{PP}	V_{PP}
11	NMI	NMI	NMI	NMI	NMI	A_9	A_9
12	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
13	D_0	D_0	D_0	D_0	$P3_0$	O_0	O_0
14	D_1	D_1	D_1	D_1	$P3_1$	O_1	O_1
15	D_2	D_2	D_2	D_2	$P3_2$	O_2	O_2
16	D_3	D_3	D_3	D_3	$P3_3$	O_3	O_3
17	D_4	D_4	D_4	D_4	$P3_4$	O_4	O_4
18	D_5	D_5	D_5	D_5	$P3_5$	O_5	O_5
19	D_6	D_6	D_6	D_6	$P3_6$	O_6	O_6
20	D_7	D_7	D_7	D_7	$P3_7$	O_7	O_7
21	A_0	A_0	A_0	A_0	$P4_0$	A_0	A_0

Notes: 1. For the PROM mode, see section 17, “ROM.”
2. Pins marked NC should be left unconnected.

Table 1-3 Pin Arrangements in Each Operating Mode (FP-80A, TFP-80C) (cont)

Pin No.	Pin Name: Expanded Minimum Modes Mode 1	Expanded Minimum Modes Mode 2	Expanded Maximum Modes Mode 3	Expanded Maximum Modes Mode 4	Single-Chip Mode Mode 7	PROM Mode H8/534	PROM Mode H8/536
22	A_1	A_1	A_1	A_1	$P4_1$	A_1	A_1
23	A_2	A_2	A_2	A_2	$P4_2$	A_2	A_2
24	A_3	A_3	A_3	A_3	$P4_3$	A_3	A_3
25	A_4	A_4	A_4	A_4	$P4_4$	A_4	A_4
26	A_5	A_5	A_5	A_5	$P4_5$	A_5	A_5
27	A_6	A_6	A_6	A_6	$P4_6$	A_6	A_6
28	A_7	A_7	A_7	A_7	$P4_7$	A_7	A_7
29	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
30	A_8	$P5_0$ / A_8	A_8	$P5_0$/ A_8	$P5_0$	A_8	A_8
31	A_9	$P5_1$ / A_9	A_9	$P5_1$/ A_9	$P5_1$	$\overline{OE}$	$\overline{OE}$
32	A_{10}	$P5_2$ / A_{10}	A_{10}	$P5_2$/ A_{10}	$P5_2$	A_{10}	A_{10}
33	A_{11}	$P5_3$ / A_{11}	A_{11}	$P5_3$ / A_{11}	$P5_3$	A_{11}	A_{11}
34	A_{12}	$P5_4$ / A_{12}	A_{12}	$P5_4$ / A_{12}	$P5_4$	A_{12}	A_{12}
35	A_{13}	$P5_5$ / A_{13}	A_{13}	$P5_5$ / A_{13}	$P5_5$	A_{13}	A_{13}
36	A_{14}	$P5_6$ / A_{14}	A_{14}	$P5_6$ / A_{14}	$P5_6$	A_{14}	A_{14}
37	A_{15}	$P5_7$ / A_{15}	A_{15}	$P5_7$ / A_{15}	$P5_7$	$\overline{CE}$	$\overline{CE}$
38	$P6_0$ / $\overline{IRQ_2}$	$P6_0$ / $\overline{IRQ_2}$	A_{16}	$P6_0$ / $\overline{IRQ_2}$ / A_{16}	$P6_0$ / $\overline{IRQ_2}$	V_{CC}	V_{CC}
39	$P6_1$ / PW_1 / $\overline{IRQ_3}$	$P6_1$ / PW_1 / $\overline{IRQ_3}$	A_{17}	$P6_1$ / $\overline{IRQ_3}$ / A_{17}	$P6_1$ / PW_1 / $\overline{IRQ_3}$	V_{CC}	V_{CC}
40	$P6_2$ / PW_2 / $\overline{IRQ_4}$	$P6_2$ / PW_2 / $\overline{IRQ_4}$	A_{18}	$P6_2$ / $\overline{IRQ_4}$ / A_{18}	$P6_2$ / PW_2 / $\overline{IRQ_4}$	NC	NC
41	$P6_3$ / PW_3 / $\overline{IRQ_5}$	$P6_3$ / PW_3 / $\overline{IRQ_5}$	A_{19}	$P6_3$ / $\overline{IRQ_5}$ / A_{19}	$P6_3$ / PW_3 / $\overline{IRQ_5}$	NC	NC
42	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}

Notes: 1. For the PROM mode, see section 17, “ROM.”
2. Pins marked NC should be left unconnected.

Table 1-3 Pin Arrangements in Each Operating Mode (FP-80A, TFP-80C) (cont)

Pin No.	Pin Name: Expanded Minimum Modes Mode 1	Expanded Minimum Modes Mode 2	Expanded Maximum Modes Mode 3	Expanded Maximum Modes Mode 4	Single-Chip Mode Mode 7	PROM Mode H8/534	PROM Mode H8/536
43	$P7_0$ / TMCI	$P7_0$/ TMCI	$P7_0$/ TMCI	$P7_0$/ TMCI	$P7_0$/ TMCI	NC	NC
44	$P7_1$ / FTI_1	$P7_1$/ FTI_1	$P7_1$/ FTI_1	$P7_1$/ FTI_1	$P7_1$/ FTI_1	NC	NC
45	$P7_2$ / FTI_2	$P7_2$ / FTI_2	$P7_2$ / FTI_2	$P7_2$ / FTI_2	$P7_2$ / FTI_2	NC	NC
46	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	$P7_3$ / FTI_3 / TMRI	NC	NC
47	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	$P7_4$/ $FTOB_1$ / $FTCI_1$	$P7_4$ / $FTOB_1$ / $FTCI_1$	NC	NC
48	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	$P7_5$ / $FTOB_2$ / $FTCI_2$	NC	NC
49	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	$P7_6$ / $FTOB_3$ / $FTCI_3$	NC	NC
50	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	$P7_7$ / $FTOA_1$	NC	NC
51	AV_{SS}	AV_{SS}	AV_{SS}	AV_{SS}	AV_{SS}	V_{SS}	V_{SS}
52	$P8_0$ / AN_0	$P8_0$ / AN_0	$P8_0$ / AN_0	$P8_0$ / AN_0	$P8_0$ / AN_0	NC	NC
53	$P8_1$ / AN_1	$P8_1$ / AN_1	$P8_1$ / AN_1	$P8_1$ / AN_1	$P8_1$ / AN_1	NC	NC
54	$P8_2$ / AN_2	$P8_2$ / AN_2	$P8_2$ / AN_2	$P8_2$ / AN_2	$P8_2$ / AN_2	NC	NC
55	$P8_3$ / AN_3	$P8_3$ / AN_3	$P8_3$ / AN_3	$P8_3$ / AN_3	$P8_3$ / AN_3	NC	NC
56	$P8_4$ / AN_4	$P8_4$ / AN_4	$P8_4$ / AN_4	$P8_4$ / AN_4	$P8_4$ / AN_4	NC	NC
57	$P8_5$ / AN_5	$P8_5$ / AN_5	$P8_5$ / AN_5	$P8_5$ / AN_5	$P8_5$ / AN_5	NC	NC
58	$P8_6$ / AN_6	$P8_6$ / AN_6	$P8_6$ / AN_6	$P8_6$ / AN_6	$P8_6$ / AN_6	NC	NC
59	$P8_7$ / AN_7	$P8_7$ / AN_7	$P8_7$ / AN_7	$P8_7$ / AN_7	$P8_7$ / AN_7	NC	NC

Notes: 1. For the PROM mode, see section 17, “ROM.”

2. Pins marked NC should be left unconnected.

Table 1-3 Pin Arrangements in Each Operating Mode (FP-80A, TFP-80C) (cont)

Pin No.	Pin Name: Expanded Minimum Modes Mode 1	Expanded Minimum Modes Mode 2	Expanded Maximum Modes Mode 3	Expanded Maximum Modes Mode 4	Single-Chip Mode Mode 7	PROM Mode H8/534	PROM Mode H8/536
60	AV_{CC}	AV_{CC}	AV_{CC}	AV_{CC}	AV_{CC}	V_{CC}	V_{CC}
61	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	$P9_0$ / $FTOA_2$	NC	NC
62	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	$P9_1$ / $FTOA_3$	NC	NC
63	$P9_2$ / PW_1	$P9_2$ / PW_1	$P9_2$ / PW_1	$P9_2$ / PW_1	$P9_2$ / PW_1	NC	NC
64	$P9_3$ / PW_2	$P9_3$ / PW_2	$P9_3$ / PW_2	$P9_3$ / PW_2	$P9_3$ / PW_2	NC	NC
65	$P9_4$ / PW_3	$P9_4$ / PW_3	$P9_4$ / PW_3	$P9_4$ / PW_3	$P9_4$ / PW_3	NC	NC
66	$P9_5$ / TXD	$P9_5$ / TXD	$P9_5$ / TXD	$P9_5$ / TXD	$P9_5$ / TXD	NC	NC
67	$P9_6$ / RXD	$P9_6$ / RXD	$P9_6$ / RXD	$P9_6$ / RXD	$P9_6$ / RXD	NC	NC
68	$P9_7$ / SCK	$P9_7$ / SCK	$P9_7$ / SCK	$P9_7$ / SCK	$P9_7$ / SCK	NC	NC
69	EXTAL	EXTAL	EXTAL	EXTAL	EXTAL	NC	NC
70	XTAL	XTAL	XTAL	XTAL	XTAL	NC	NC
71	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}	V_{SS}
72	$P1_0$ / ø	$P1_0$ / ø	$P1_0$ / ø	$P1_0$ / ø	$P1_0$ / ø	NC	NC
73	$P1_1$ / E	$P1_1$ / E	$P1_1$ / E	$P1_1$ / E	$P1_1$ / E	NC	NC
74	$P1_2$ / $\overline{BACK}$	$P1_2$ / $\overline{BACK}$	$P1_2$ / $\overline{BACK}$	$P1_2$ / $\overline{BACK}$	$P1_2$	NC	NC
75	$P1_3$ / $\overline{BREQ}$	$P1_3$ / $\overline{BREQ}$	$P1_3$ / $\overline{BREQ}$	$P1_3$ / $\overline{BREQ}$	$P1_3$	NC	NC
76	$P1_4$ / $\overline{WAIT}$	$P1_4$ / $\overline{WAIT}$	$P1_4$ / $\overline{WAIT}$	$P1_4$ / $\overline{WAIT}$	$P1_4$	NC	A_{15}
77	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	$P1_5$ / $\overline{IRQ_0}$	NC	A_{16}
78	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	$P1_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$	NC	$\overline{PGM}$
79	$P1_7$ / TMO	$P1_7$ / TMO	$P1_7$ / TMO	$P1_7$ / TMO	$P1_7$ / TMO	NC	NC
80	$\overline{AS}$	$\overline{AS}$	$\overline{AS}$	$\overline{AS}$	$P2_0$	NC	NC

Notes: 1. For the PROM mode, see section 17, “ROM.”

2. Pins marked NC should be left unconnected.

Pin Functions: Table 1-4 gives a concise description of the function of each pin.

Table 1-4 Pin Functions

Type	Symbol	Pin No. CP-84, CG-84	Pin No. FP-80A, TFP-80C	I/O	Name and Function
Power	V_{CC}	16, 55	5, 42	I	**Power:** Connected to the power supply (+5 V). Connect both V_{CC} pins to the system power supply (+5 V). The chip will not operate if either pin is left unconnected.
	V_{SS}	2, 24 41, 42 64, 83	12, 29 71	I	**Ground:** Connected to ground (0 V). Connect all V_{SS} pins to the system power supply (0 V). The chip will not operate if any V_{SS} pin is left unconnected.
Clock	XTAL	1	70	I	**Crystal:** Connected to a crystal oscillator. The crystal frequency should be double the desired ø clock frequency. If an external clock is input at the EXTAL pin, leave the XTAL pin unconnected.
	EXTAL	84	69	I	**External Crystal:** Connected to a crystal oscillator or external clock. The frequency of the external clock should be double the desired ø clock frequency. See section 8.2, “Oscillator Circuit” for examples of connections to a crystal and external clock.
	ø	3	72	O	**System Clock:** Supplies the ø clock to peripheral devices.
	E	4	73	O	**Enable Clock:** Supplies an E clock to E clock based peripheral devices.
System control	$\overline{BACK}$	5	74	O	**Bus Request Acknowledge:** Indicates that the bus right has been granted to an external device. Notifies an external device that issued a $\overline{BREQ}$ signal that it now has control of the bus.

Table 1-4 Pin Functions (cont)

Type	Symbol	Pin No. CP-84, CG-84	Pin No. FP-80A, TFP-80C	I/O	Name and Function
System control	$\overline{\text{BREQ}}$	6	75	I	**Bus Request:** Sent by an external device to the H8/534 or H4/536 to request the bus right.
	$\overline{\text{STBY}}$	20	9	I	**Standby:** A transition to the hardware standby mode (a power-down state) occurs when a Low input is received at the $\overline{\text{STBY}}$ pin.
	$\overline{\text{RES}}$	21	10	I/O	**Reset:** Low input or low output due to watchdog timer overflow causes the H8/534 or H8/536 chip to reset.
Address bus	$A_{19} – A_8$ $A_7 – A_0$	54 – 43 40 – 33	41 – 30 28 – 21	O	**Address Bus:** Address output pins.
Data bus	$D_7 – D_0$	32 – 25	20 – 13	I/O	**Data Bus:** 8-Bit bidirectional data bus.
Bus control	$\overline{\text{WAIT}}$	7	76	I	**Wait:** Requests the CPU to insert one or more Tw states when accessing an off-chip address.
	$\overline{\text{AS}}$	11	80	O	**Address Strobe:** Goes Low to indicate that there is a valid address on the address bus.
	R/$\overline{\text{W}}$	12	1	O	**Read/Write:** Indicates whether the CPU is reading or writing data on the bus. • High—Read • Low—Write
	$\overline{\text{DS}}$	13	2	O	**Data Strobe:** Goes Low to indicate the presence of valid data on the data bus.
	$\overline{\text{RD}}$	14	3	O	**Read:** Goes Low to indicate that the CPU is reading an external address.
	$\overline{\text{WR}}$	15	4	O	**Write:** Goes Low to indicate that the CPU is writing to an external address.

Table 1-4 Pin Functions (cont)

Type	Symbol	Pin No. CP-84, CG-84	Pin No. FP-80A, TFP-80C	I/O	Name and Function
Interrupt	NMI	22	11	I	**NonMaskable Interrupt:** Highest-priority interrupt request. The port 1 control register (P1CR) determines whether the interrupt is requested on the rising or falling edge of the NMI input.
	$\overline{IRQ_0}$	8	77	I	**Interrupt Request 0 and 1:** Maskable interrupt request pins.
	$\overline{IRQ_1}$	9	78		
	$\overline{IRQ_2}$	51	38		
	$\overline{IRQ_3}$	52	39		
	$\overline{IRQ_4}$	53	40		
	$\overline{IRQ_5}$	54	41		
Operating mode control	MD_2	19	8	I	**Mode:** Input pins for setting the MCU operating mode according to the table below.
	MD_1	18	7		
	MD_0	17	6		

MD_2	MD_1	MD_0	Mode	Description
0	0	0	Mode 0	—
0	0	1	Mode 1	Expanded minimum mode (ROM disabled)
0	1	0	Mode 2	Expanded minimum mode (ROM enabled)
0	1	1	Mode 3	Expanded maximum mode (ROM disabled)
1	0	0	Mode 4	Expanded maximum mode (ROM enabled)
1	0	1	Mode 5	—
1	1	0	Mode 6	—
1	1	1	Mode 7	Single-chip mode

The inputs at these pins must not be changed while the chip is operating.

Table 1-4 Pin Functions (cont)

Type	Symbol	Pin No. CP-84, CG-84	Pin No. FP-80A, TFP-80C	I/O	Name and Function
16-Bit free-running timer (FRT)	FTOA1	63	50	O	**FRT Output Compare A (channels 1, 2, and 3):** Output pins for the output compare A function of free-running timer channels 1, 2, and 3.
	FTOA2	75	61		
	FTOA3	76	62		
	FTOB1	60	47	O	**FRT Output Compare B (channels 1, 2, and 3):** Output pins for the output compare B function of free-running timer channels 1, 2, and 3.
	FTOB2	61	48		
	FTOB3	62	49		
	FTCI1	60	47	I	**FRT Counter Clock Input (channels 1, 2, and 3):** External clock input pins for the free-running counters (FRCs) of free-running timer channels 1, 2, and 3.
	FTCI2	61	48		
	FTCI3	62	49		
	FTI1	57	44	I	**FRT Input Capture (channels 1, 2, and 3):** Input capture pins for free-running timer channels 1, 2, and 3.
	FTI2	58	45		
	FTI3	59	46		
8-Bit timer	TMO	10	79	O	**8-bit Timer Output:** Compare-match output pin for the 8-bit timer.
	TMCI	56	43	I	**8-bit Timer Clock Input:** External clock input pin for the 8-bit timer counter.
	TMRI	59	46	I	**8-bit Timer Counter Reset Input:** A high input at this pin resets the 8-bit timer counter.
PWM timer	PW1	77	63	O	**PWM Timer Output (channels 1, 2, and 3):** Pulse-width modulation timer output pulses.
	PW2	78	64		
	PW3	79	65		

Table 1-4 Pin Functions (cont)

Type	Symbol	Pin No. CP-84, CG-84	Pin No. FP-80A, TFP-80C	I/O	Name and Function
Serial communication interface signals	TXD_1	80	66	O	**Transmit Data:** Data output pins for serial communication interfaces 1 and 2.
	TXD_2	77	63		
	RXD_1	81	67	I	**Receive Data:** Data input pins for serial communication interfaces 1 and 2.
	RXD_2	78	64		
	SCK_1	82	68	I/O	**Serial Clock:** Input/output pins for the serial clock of serial interface 1 and 2.
	SCK_2	79	65		
A/D converter	AN_7 – AN_0	73 – 66	59 – 52	I	**Analog Input:** Analog signal input pins.
	AV_{CC}	74	60	I	**Analog Reference Voltage:** Reference voltage and power supply pin for the A/D converter.
	AV_{SS}	65	51	I	**Analog Ground:** Ground pin for the A/D converter.
	$\overline{ADTRG}$	9	78	I	**External Trigger:** External trigger input pin for the A/D converter.
Parallel I/O	$P1_7$ – $P1_0$	10 – 3	79 – 72	I/O	**Port 1:** An 8-bit input/output port. The direction of each bit is determined by the port 1 data direction register (P1DDR).
	$P2_4$ – $P2_0$	15 – 11	4 – 1, 80	I/O	**Port 2:** A 5-bit input/output port. The direction of each bit is determined by the port 2 data direction register (P2DDR).
	$P3_7$ – $P3_0$	32 – 25	20 – 13	I/O	**Port 3:** An 8-bit input/output port. The direction of each bit is determined by the port 3 data direction register (P3DDR).
	$P4_7$ – $P4_0$	40 – 33	28 – 21	I/O	**Port 4:** An 8-bit input/output port. The direction of each bit is determined by the port 4 data direction register (P4DDR). These pins can drive LED indicators.

Table 1-4 Pin Functions (cont)

Type	Symbol	Pin No. CP-84, CG-84	Pin No. FP-80A, TFP-80C	I/O	Name and Function
Parallel I/O	$P5_7$ – $P5_0$	50 – 43	37 – 30	I/O	**Port 5:** An 8-bit input/output port. The direction of each bit is determined by the port 5 data direction register (P5DDR). These pins have built-in MOS input pull-ups.
	$P6_3$ – $P6_0$	54 – 51	41 – 38	I/O	**Port 6:** A 4-bit input/output port. The direction of each bit is determined by the port 6 data direction register (P6DDR). These pins have built-in MOS input pull-ups.
	$P7_7$ – $P7_0$	63 – 56	50 – 43	I/O	**Port 7:** An 8-bit input/output port. The direction of each bit is determined by the port 7 data direction register (P7DDR). These pins have Schmitt inputs.
	$P8_7$ – $P8_0$	73 – 66	59 – 52	I	**Port 8:** An 8-bit input port
	$P9_7$ – $P9_0$	82 – 75	68 – 61	I/O	**Port 9:** An 8-bit input/output port. The direction of each bit is determined by the port 9 data direction register (P9DDR).

Section 2 MCU Operating Modes and Address Space

2.1 Overview

The H8/534 or H8/536 microcomputer unit (MCU) operates in five modes numbered 1, 2, 3, 4, and 7. The mode is selected by the inputs at the mode pins (MD_2 to MD_0) at the instant when the chip comes out of a reset. As indicated in table 2-1, the MCU mode determines the size of the address space, the usage of on-chip ROM, and the operating mode of the CPU. The MCU mode also affects the functions of I/O pins.

Table 2-1 Operating Modes

MD_2	MD_1	MD_0	MCU Mode	Address Space	On-Chip ROM	CPU Mode
0	0	0	—	—	—	—
0	0	1	Mode 1	Expanded minimum	Disabled	Minimum mode
0	1	0	Mode 2	Expanded minimum	Enabled	Minimum mode
0	1	1	Mode 3	Expanded maximum	Disabled	Maximum mode
1	0	0	Mode 4	Expanded maximum	Enabled	Maximum mode
1	0	1	—	—	—	—
1	1	0	—	—	—	—
1	1	1	Mode 7	Single-chip only	Enabled	Minimum mode

Notation: 0: Low level
1: High level
—: Cannot be used

Modes 1 to 4 are referred to as “expanded” because they permit access to off-chip memory and peripheral addresses. The expanded minimum modes (modes 1 and 2) support a maximum address space of 64 kbytes. The expanded maximum modes (modes 3 and 4) support a maximum address space of 1 Mbyte.

Interrupt service is slightly slower in the expanded maximum modes than in the other modes because the CPU has to save its code page register.

In single-chip mode all ports are available for general-purpose input and output, but off-chip addresses cannot be accessed.

The H8/534 and H8/536 cannot be set to modes 0, 5, and 6. The mode pins should never be set to these values.

The inputs at the mode pins must not be changed while the chip is operating.

2.2 Mode Descriptions

The five MCU modes are described below. For further information on the I/O pin functions in each mode, see section 9, “I/O Ports.”

Mode 1 (Expanded Minimum Mode): Mode 1 supports a maximum 64-kbyte address space which does not include any on-chip ROM. Ports 1 to 5 are used for bus lines and bus control signals as follows:

Control signals: Ports 1* and 2
Data bus: Port 3
Address bus: Ports 4 and 5

* The functions of individual pins of port 1 are software-selectable.

Mode 2 (Expanded Minimum Mode): Mode 2 supports a maximum 64-kbyte address space of which the first part is in on-chip ROM. Ports 1 to 5 are used for bus lines and bus control signals as follows:

Control signals: Ports 1* and 2
Data bus: Port 3
Address bus: Ports 4 and 5*

* The functions of individual pins in ports 1 and 5 are software-selectable.

Note: In mode 2, port 5 is initially a general-purpose input port. Software must change it to output before using it for the address bus. See section 9.6, “Port 5” for details. The following instruction makes all pins of port 5 into output pins:

```
MOV.B  #H'FF, @H'FE88*
```

* H'xx or H'xxxx express the hexadecimal number.

Mode 3 (Expanded Maximum Mode): Mode 3 supports a maximum 1-Mbyte address space which does not include any on-chip ROM. Ports 1 to 6 are used for bus lines and bus control signals as follows:

Control signals: Ports 1* and 2
Data bus: Port 3
Address bus: Ports 4, 5, and 6

* The functions of individual pins of port 1 are software-selectable.

Mode 4 (Expanded Maximum Mode): Mode 4 supports a maximum 1-Mbyte address space of which the first part is in on-chip ROM. Ports 1 to 6 are used for bus lines and bus control signals as follows:

Control signals: Ports 1* and 2
Data bus: Port 3
Address bus: Ports 4, 5*, and 6*

* The functions of individual pins in ports 1, 5, and 6 are software-selectable.

Note: In mode 4, ports 5 and 6 are initially general-purpose input ports. Software must change them to output before using them for the address bus. See section 9.6, “Port 5” and 10.7, “Port 6” for details. The following instruction sets all pins of ports 5 and 6 to output:

```
MOV.W #H'FFFF, @H'FE88
```

Mode 7 (Single-Chip Mode): In this mode all memory is on-chip. It is not possible to access off-chip addresses.

The single-chip mode provides the maximum number of ports. All the pins associated with the address and data buses in the expanded modes are available as general-purpose input/output ports in the single-chip mode.

2.3 Address Space Map

2.3.1 Page Segmentation

The address space is segmented into 64-kbyte pages. In the single-chip mode and expanded minimum modes there is just one page: page 0. In the expanded maximum modes there can be up to 16 pages. Figure 2-1 shows the address space of the H8/534 in each mode and indicates which parts are on- and off-chip. Figure 2-2 shows the address space of the H8/536.

2.3.2 Page 0 Address Allocations

The high and low address areas in page 0 are reserved for registers and vector tables.

Vector Tables: The low address area contains the exception vector table and DTC vector table. The CPU accesses the exception vector table to obtain the addresses of user-coded exception-handling routines. The DTC vector table contains pointers to tables of register information used by the on-chip chip data transfer controller. The size of these tables depends on the CPU operating mode. Details are given in section 4.1.3, “Exception Factors and Vector Table,” section 5.2.3, “Interrupt Vector Table,” and section 6.3.2, “DTC Vector Table.”

In modes 2 and 4 the vector tables are located in on-chip ROM. In modes 1, 3, and 7 the vector tables are in external memory.

Register Field: The highest 384 addresses in page 0 (addresses H'FE80 to H'FFFF) belong to control, status, and data registers used by the I/O ports and on-chip supporting modules. Program code cannot be located at these addresses.

The CPU accesses addresses in this register field like other addresses in the address space. By reading and writing at these addresses the CPU controls the on-chip supporting modules and communicates via the I/O ports. A complete map of the register field is given in appendix B.

On-Chip RAM: One of the control registers in the register field is a RAM control register (RAMCR) containing a RAM enable bit (RAME) that enables or disables the 2-kbyte on-chip RAM. When this bit is set to 1 (its default value), addresses H'F680 to H'FE7F are located on-chip. When this bit is cleared to 0, these addresses are located in external memory and the on-chip RAM is not used. See section 16, “RAM” for further information.

The RAME bit is bit 7 at address H'FF11.

Coding Example:
To enable on-chip RAM: `BSET.B #7, @H'FF11`
To disable on-chip RAM: `BCLR.B #7, @H'FF11`

Note: If on-chip RAM is disabled in the single-chip mode, access to addresses H'F680 to H'FE7F causes an address error.

2.4 Mode Control Register (MDCR)

Another control register in the register field in page 0 is the mode control register (MDCR). The mode control register can be read by the CPU, but not written. Table 3-2 lists the attributes of this register.

Table 2-2 Mode Control Register

Name	Abbreviation	Read/Write	Address
Mode control register	MDCR	Read only	H'FF12

The bit configuration of this register is shown below.

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	MDS2	MDS1	MDS0
Initial value	1	1	0	0	0	*	*	*
Read/Write	—	—	—	—	—	R	R	R

* Initialized according to MD_2 to MD_0.

Bits 7 and 6—Reserved: These bits cannot be modified and are always read as 1.

Bits 5 to 3—Reserved: These bits cannot be modified and are always read as 0.

Bits 2 to 0—Mode Select 2 to 0 (MDS2 to MDS0): These bits indicate the values of the mode pins (MD_2 to MD_0) latched on the rising edge of the signal. MDS2 corresponds to MD_2, MDS1 to MD_1, and MDS0 to MD_0. These bits can be read but not written.

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

```
CMP:G.B #H'C1, @H'FF12
```

The comparison is with H'C1 instead of H'01 because bits 7 and 6 are always read as 1.

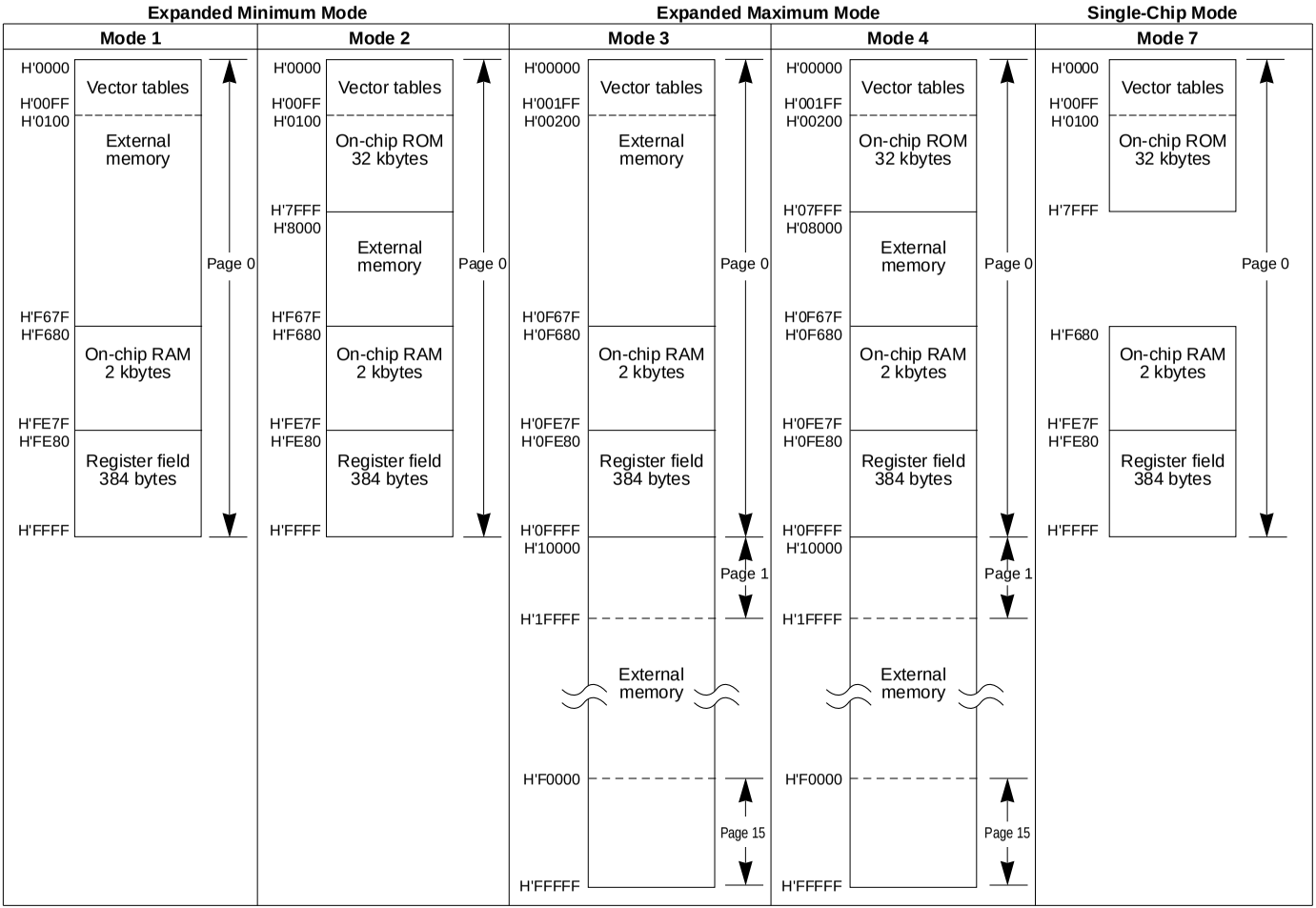


Figure 2-1 H8/534 Memory Map in Each Operating Mode

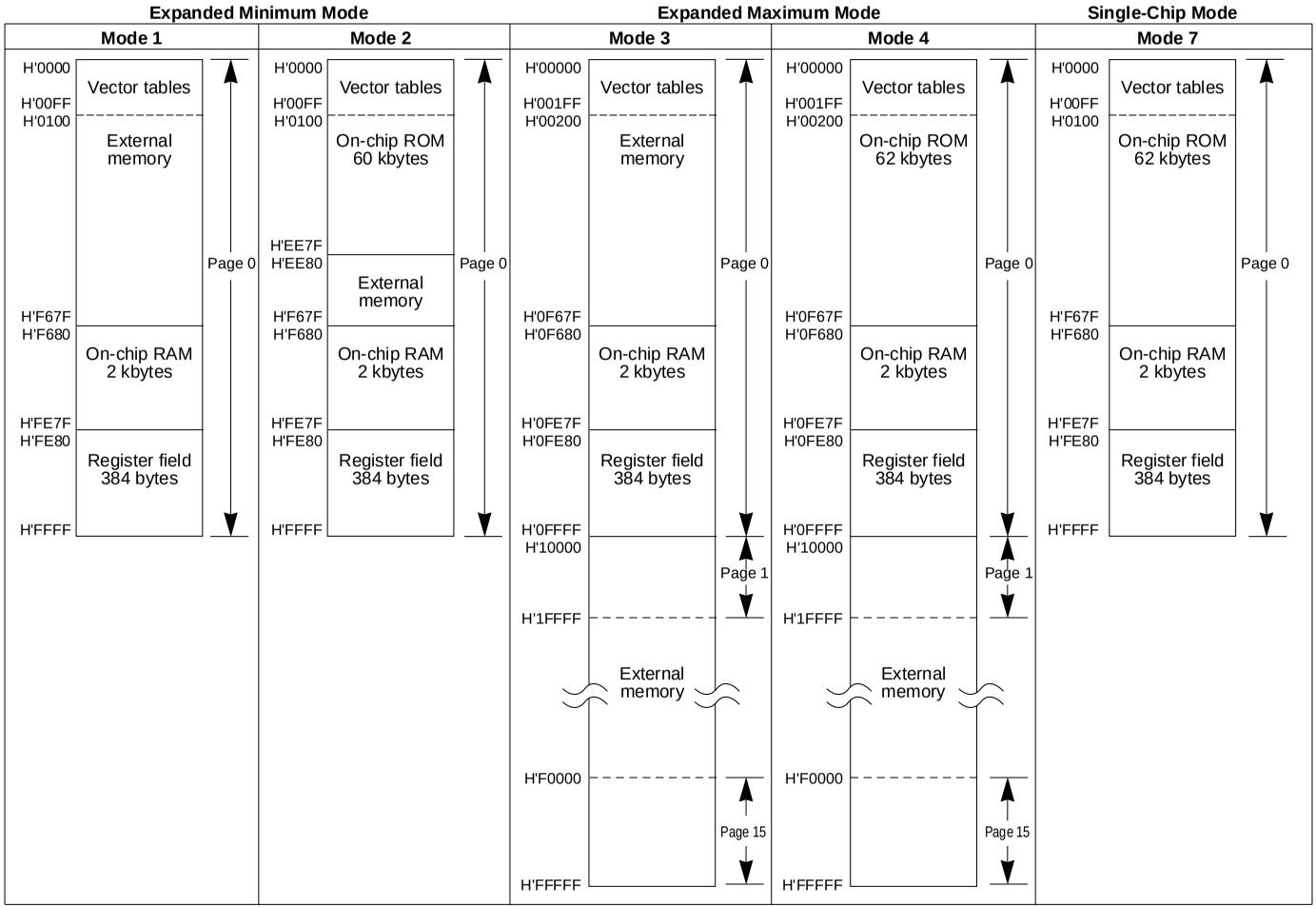


Figure 2-2 H8/536 Memory Map in Each Operating Mode

Section 3 CPU

3.1 Overview

The H8/534 and H8/536 have the H8/500 Family CPU: a high-speed central processing unit designed for realtime control of a wide range of medium-scale office and industrial equipment. Its Hitachi-original architecture features eight 16-bit general registers, internal 16-bit data paths, and an optimized instruction set.

Section 3 summarizes the CPU architecture and instruction set.

3.1.1 Features

The main features of the H8/500 CPU are listed below.

- General-register machine
  — Eight 16-bit general registers
  — Seven control registers (two 16-bit registers, five 8-bit registers)
- High speed: maximum 16 MHz (S-mask versions)
  At 16 MHz a register-register add operation takes only 125 ns.
- Address space managed in 64-kbyte pages, expandable to 1 Mbyte*
  Page registers make four pages available simultaneously: a code page, stack page, data page, and extended page.
- Two CPU operating modes:
  — Minimum mode: Maximum 64-kbyte address space
  — Maximum mode: Maximum 1 Mbyte address space*
- Highly orthogonal instruction set
  Addressing modes and data sizes can be specified independently within each instruction.
- 1.5 Addressing modes
  Register-register and register-memory operations are supported.
- Optimized for efficient programming in C language
  In addition to the general registers and orthogonal instruction set, the CPU has special short formats for frequently-used instructions and addressing modes.

* The CPU architecture supports up to 16 Mbytes of external memory, but the H8/534 and H8/536 have only enough address pins to address 1 Mbyte.

3.1.2 Address Space

The address space size depends on the operating mode.

The H8/534 or H8/536 MCU has five operating modes, which are selected by the input to the mode pins (MD_2 to MD_0) when the chip comes out of a reset. The CPU, however, has only two operating modes. The MCU operating mode determines the CPU operating mode, which in turn determines the maximum address space size as indicated in figure 3-1.

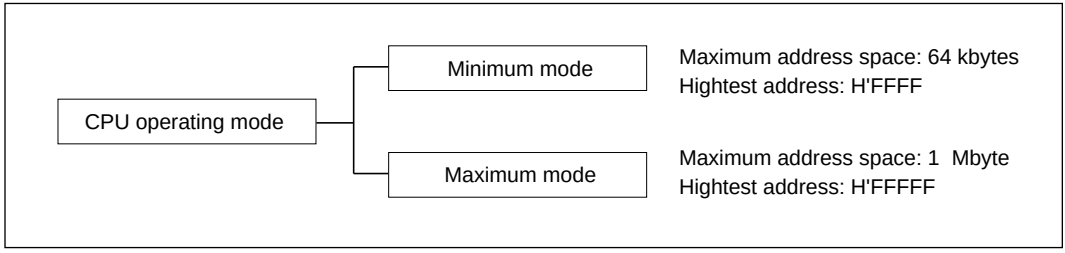


Figure 3-1 CPU Operating Modes

3.1.3 Register Configuration

Figure 3-2 shows the register structure of the CPU. There are two groups of registers: the general registers (Rn) and control registers (CR).

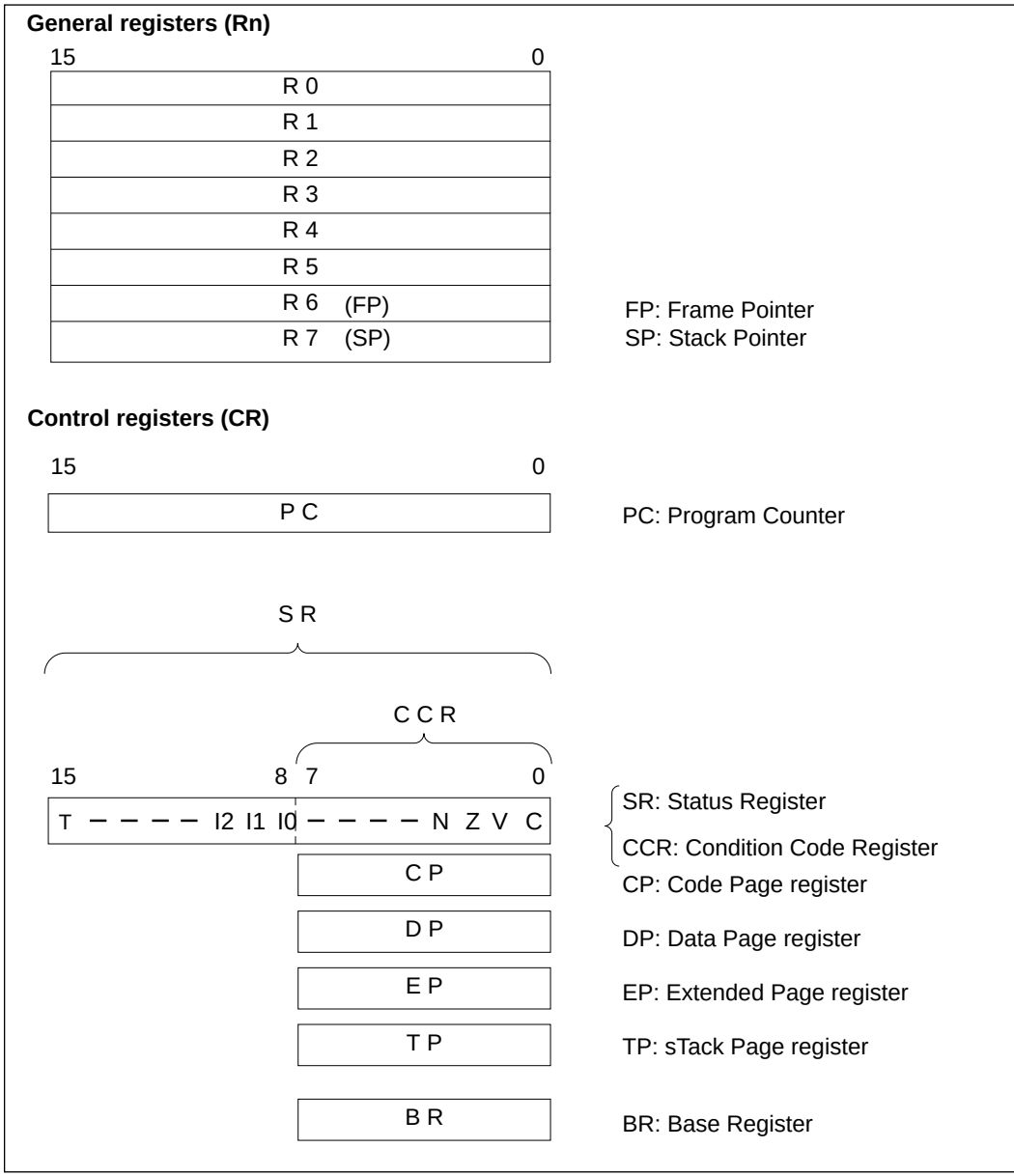


Figure 3-2 Registers in the CPU

3.2 CPU Register Descriptions

3.2.1 General Registers

All eight of the 16-bit general registers are functionally alike; there is no distinction between data registers and address registers. When these registers are accessed as data registers, either byte or word size can be selected.

R6 and R7, in addition to functioning as general registers, have special assignments.

R7 is the stack pointer, used implicitly in exception handling and subroutine calls. It can be designated by the name SP, which is synonymous with R7. As indicated in figure 3-3, it points to the top of the stack. It is also used implicitly by the LDM and STM instructions, which load and store multiple registers from and to the stack and pre-decrement or post-increment R7 accordingly.

R6 functions as a frame pointer (FP). The LINK and UNLK instructions use R6 implicitly to reserve or release a stack frame.

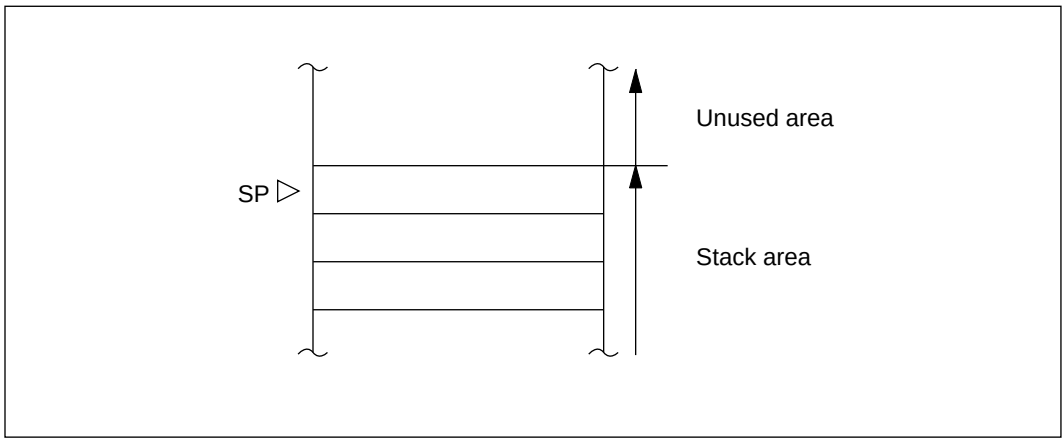


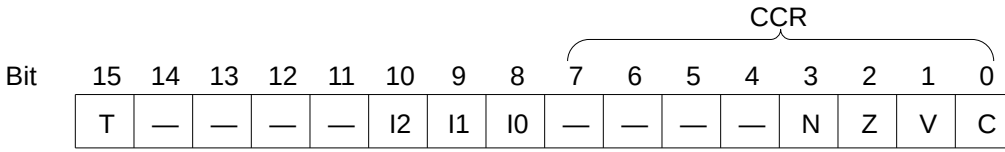
Figure 3-3 Stack Pointer

3.2.2 Control Registers

The CPU control registers (CR) include a 16-bit program counter (PC), a 16-bit status register (SR), four 8-bit page registers, and one 8-bit base register (BR).

Program Counter (PC): This 16-bit register indicates the address of the next instruction the CPU will execute.

Status Register (SR): This 16-bit register contains internal status information. The lower half of the status register is referred to as the condition code register (CCR): it can be accessed as a separate condition code byte.



Bit 15—Trace (T): When this bit is set to 1, the CPU operates in trace mode and generates a trace exception after every instruction. See section 4.4, “Trace” for a description of the trace exception-handling sequence.

When the value of this bit is 0, instructions are executed in normal continuous sequence. This bit is cleared to 0 at a reset.

Bits 14 to 11—Reserved: These bits cannot be modified and are always read as 0.

Bits 10 to 8—Interrupt Mask (I2, I1, I0): These bits indicate the interrupt request mask level (0 to 7). As shown in table 3-1, an interrupt request is not accepted unless it has a higher level than the value of the mask. A nonmaskable interrupt (NMI), which has level 8, is accepted at any mask level. After an interrupt is accepted, I2, I1, and I0 are changed to the level of the interrupt. Table 3-2 indicates the values of the I bits after an interrupt is accepted.

A reset sets all three bits (I2, I1, and I0) to 1, masking all interrupts except NMI.

Table 3-1 Interrupt Mask Levels

Priority	Mask Level	Mask Bits I2	I1	I0	Interrupts Accepted
High	7	1	1	1	NMI
	6	1	1	0	Level 7 and NMI
	5	1	0	1	Levels 6 to 7 and NMI
	4	1	0	0	Levels 5 to 7 and NMI
	3	0	1	1	Levels 4 to 7 and NMI
	2	0	1	0	Levels 3 to 7 and NMI
	1	0	0	1	Levels 2 to 7 and NMI
Low	0	0	0	0	Levels 1 to 7 and NMI

Table 3-2 Interrupt Mask Bits after an Interrupt is Accepted

Level of Interrupt Accepted	I2	I1	I0
NMI (8)	1	1	1
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	0	1

Bits 7 to 4—Reserved: These bits cannot be modified and are always read as 0.

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 set to 1 when a carry or borrow occurs at the most significant bit, and is cleared to 0 (or left unchanged) at other times.

The specific changes that occur in the condition code bits when each instruction is executed are listed in appendix A.1 “Instruction Tables.” See the *H8/500 Series Programming Manual* for further details.

Page Registers: The code page register (CP), data page register (DP), extended page register (EP), and stack page register (TP) are 8-bit registers that are used only in the maximum mode. No use of their contents is made in the minimum mode.

In the maximum mode, the page registers combine with the program counter and general registers to generate 24-bit effective addresses as shown in figure 3-4, thereby expanding the program area, data area, and stack area.

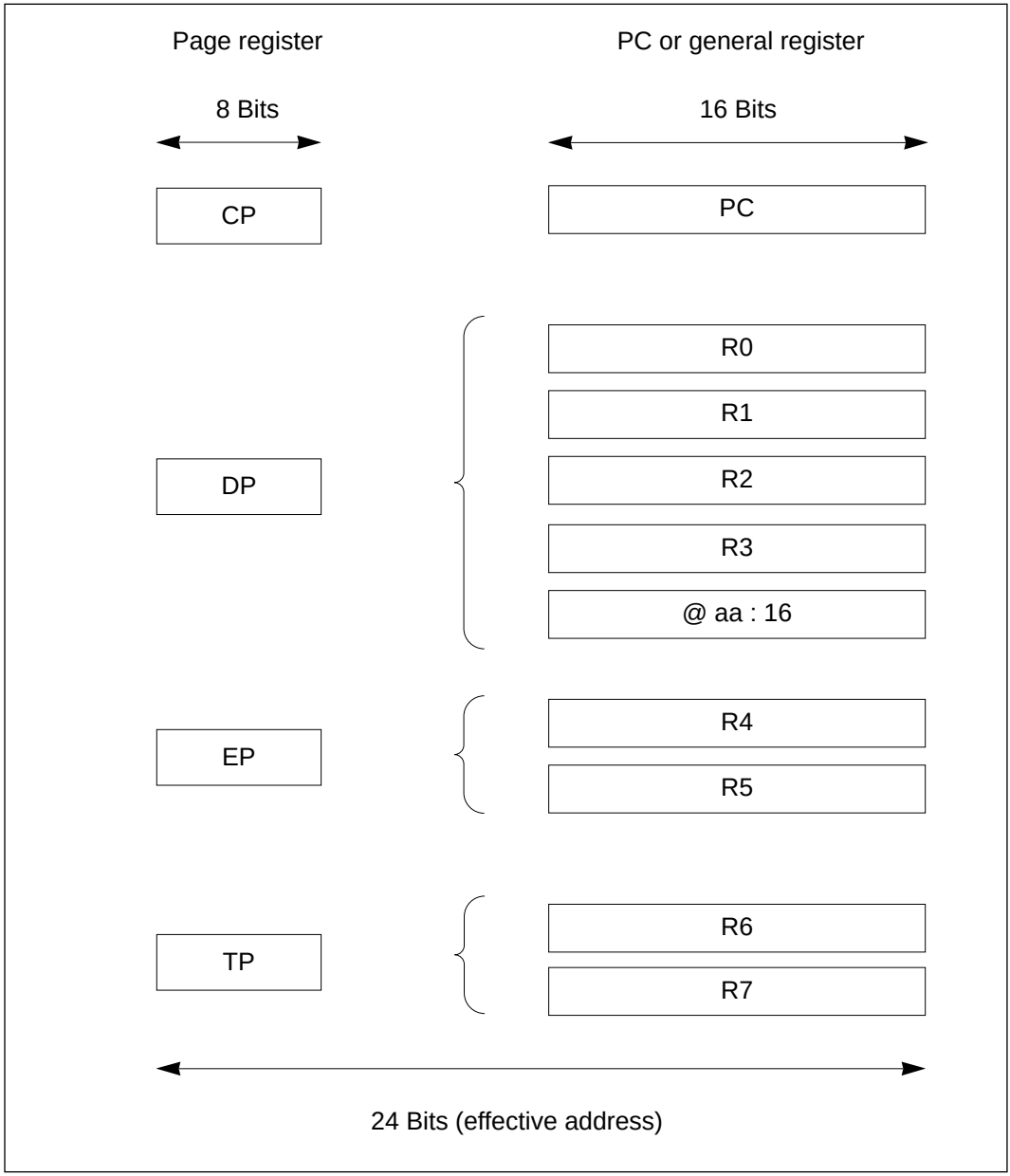


Figure 3-4 Combinations of Page Registers with Other Registers

Code Page Register (CP): The code page register and the program counter combine to generate a 24-bit program code address. In the maximum mode, the code page register is initialized at a reset to a value loaded from the vector table, and both the code page register and program counter

are saved and restored in exception handling.

Data Page Register (DP): The data page register combines with general registers R0 to R3 to generate a 24-bit effective address. The data page register contains the upper 8 bits of the address. It is used to calculate effective addresses in the register indirect addressing mode using R0 to R3, and in the 16-bit absolute addressing mode (@aa:16).

The data page register is rewritten by the LDC instruction.

Extended Page Register (EP): The extended page register combines with general register R4 or R5 to generate a 24-bit operand address. The extended page register contains the upper 8 bits of the address. It is used to calculate effective addresses in the register indirect addressing mode using R4 or R5.

The extended page can be used as an additional data page.

Stack Page Register (TP): The stack page register combines with R6 (FP) or R7 (SP) to generate a 24-bit stack address. The stack page register contains the upper 8 bits of the address. It is used to calculate effective addresses in the register indirect addressing mode using R6 or R7, in exception handling, and subroutine calls.

Base Register (BR): This 8-bit register stores the base address used in the short absolute addressing mode (@aa:8). In this addressing mode a 16-bit effective address in page 0 is generated by using the contents of the base register as the upper 8 bits and an address given in the instruction code as the lower 8 bits. See figure 3-5.

In the short absolute addressing mode the address is always located in page 0.

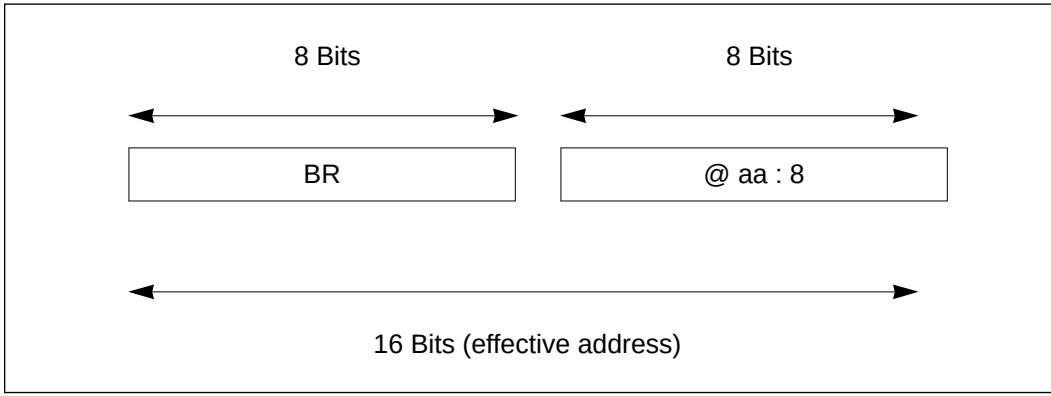


Figure 3-5 Short Absolute Addressing Mode and Base Register

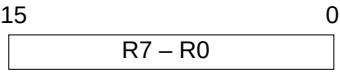
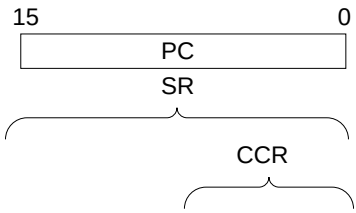
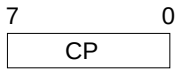
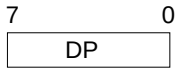
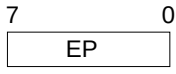
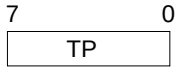
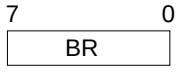
3.2.3 Initial Register Values

When the CPU is reset, its internal registers are initialized as shown in table 3-3. Note that the stack pointer (R7) and base register (BR) are not initialized to fixed values. Also, of the page registers used in maximum mode, only the code page register (CP) is initialized; the other three page registers come out of the reset state with undetermined values.

Accordingly, in the minimum mode the first instruction executed after a reset should initialize the stack pointer. The base register must also be initialized before the short absolute addressing mode (@aa:8) is used.

In the maximum mode, the first instruction executed after a reset should initialize the stack page register (TP) and the next instruction should initialize the stack pointer. Later instructions should initialize the base register and the other page registers as necessary.

Table 3-3 Initial Values of Registers

Register	Initial Value: Minimum Mode	Initial Value: Maximum Mode
General registers		
	Undetermined	Undetermined
Control registers		
	H'070x (x: undetermined)	H'070x (x: undetermined)
	Undetermined	Loaded from vector table
	Undetermined	Undetermined
	Undetermined	Undetermined
	Undetermined	Undetermined
	Undetermined	Undetermined

3.3 Data Formats

The H8/500 CPU can process 1-bit data, 4-bit BCD data, 8-bit (byte) data, 16-bit (word) data, and 32-bit (longword) data.

- Bit manipulation instructions operate on 1-bit data.
- Decimal arithmetic instructions operate on 4-bit BCD data.
- Almost all instructions operate on byte and word data.
- Multiply and divide instructions operate on longword data.

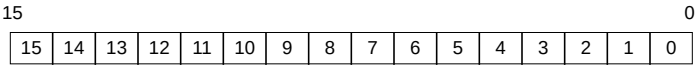
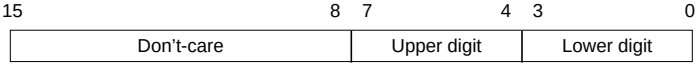
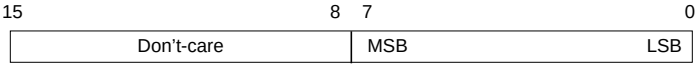
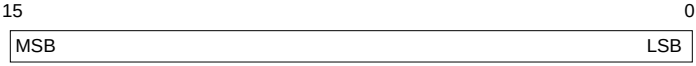
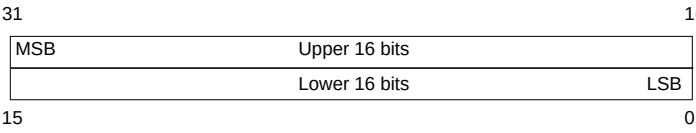
3.3.1 Data Formats in General Registers

Data of all the sizes above can be stored in general registers as shown in table 3-4.

Bit data locations are specified by bit number. Bit 15 is the most significant bit. Bit 0 is the least significant bit. BCD and byte data are stored in the lower 8 bits of a general register. Word data use all 16 bits of a general register. Longword data use two general registers: the upper 16 bits are stored in Rn (n must be an even number); the lower 16 bits are stored in Rn+1.

Operations performed on BCD data or byte data do not affect the upper 8 bits of the register.

Table 3-4 General Register Data Formats

Data Type	Register No.	Data Structure
1-Bit	Rn	
BCD	Rn	
Byte	Rn	
Word	Rn	
Longword	Rn* Rn+1*	

* For longword data n must be even (0, 2, 4, or 6).

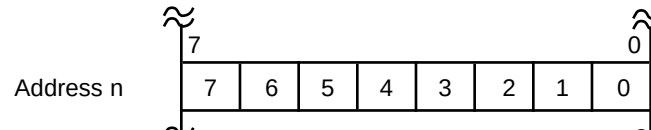
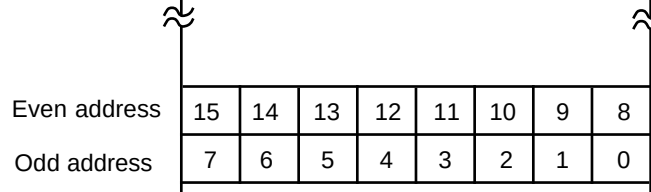
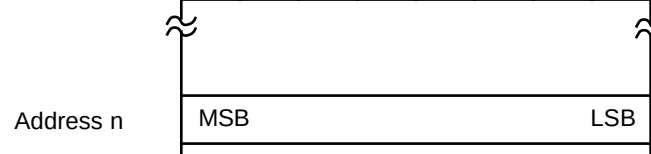
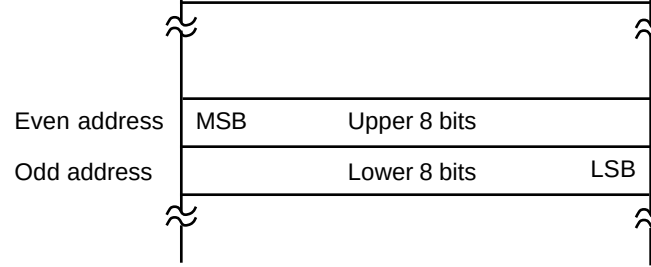
3.3.2 Data Formats in Memory

Table 3-5 indicates the data formats in memory.

Instructions that access bit data in memory have byte or word operands. The instruction specifies a bit number to indicate a specific bit in the operand.

Access to word data in memory must always begin at an even address. Access to word data starting at an odd address causes an address error. The upper 8 bits of word data are stored in address n (where n is an even number); the lower 8 bits are stored in address n+1.

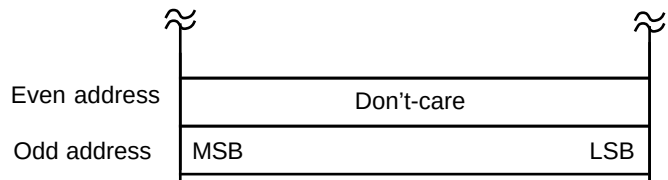
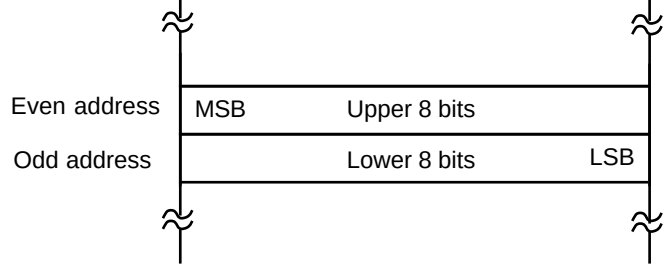
Table 3-5 Data Formats in Memory

Data Type	Data Format
1-Bit (in byte operand data)	
1-Bit (in word operand data)	
Byte	
Word	

When the stack is accessed in exception processing (to save or restore the program counter, code page register, or status register), word access is always performed, regardless of the actual data size. Similarly, when the stack is accessed by an instruction using the pre-decrement or post-increment register indirect addressing mode specifying R7 (@–R7 or @R7+), which is the stack pointer, word access is performed regardless of the operand size specified in the instruction. An address error will therefore occur if the stack pointer indicates an odd address. Programs should be coded so that the stack pointer always indicates an even address.

Table 3-6 shows the data formats on the stack.

Table 3-6 Data Formats on the Stack

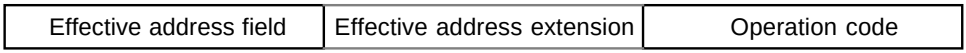
Data Type	Data Format
Byte data on stack	
Word data on stack	

3.4 Instructions

3.4.1 Basic Instruction Formats

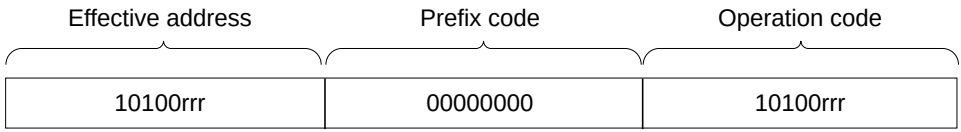
There are two basic CPU instruction formats: the general format and the special format.

General Format: This format consists of an effective address (EA) field, an effective address extension field, and an operation code (OP) field. The effective address is placed before the operation code because this results in faster execution of the instruction.

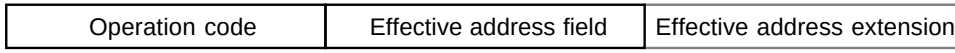


- Effective address field: One byte containing information used to calculate the effective address of an operand.
- Effective address extension: Zero to two bytes containing a displacement value, immediate data, or an absolute address. The size of the effective address extension is specified in the effective address field.
- Operation code: Defines the operation to be carried out on the operand located at the address calculated from the effective address information. Some instructions (DADD, DSUB, MOVFPE, MOVTPE) have an extended format in which the operand code is preceded by a one-byte prefix code.

- (Example of prefix code in DADD instruction)



Special Format: In this format the operation code comes first, followed by the effective address field and effective address extension. This format is used in branching instructions, system control instructions, and other instructions that can be executed faster if the operation is specified before the operand.



- Operation code: One or two bytes defining the operation to be performed by the instruction.

- Effective address field and effective address extension: Zero to three bytes containing information used to calculate an effective address.

3.4.2 Addressing Modes

The CPU supports 7 addressing modes: (1) register direct; (2) register indirect; (3) register indirect with displacement; (4) register indirect with pre-decrement or post-increment; (5) immediate; (6) absolute; and (7) PC-relative.

Due to the highly orthogonal nature of the instruction set, most instructions having operands can use any applicable addressing mode from (1) through (6). The PC-relative mode (7) is used by branching instructions.

In most instructions, the addressing mode is specified in the effective address field. The effective-address extension, if present, contains a displacement, immediate data, or an absolute address.

Table 3-7 indicates how the addressing mode is specified in the effective address field.

Table 3-7 Addressing Modes

No.	Addressing Mode	Mnemonic	EA Field	EA Extension
1	Register direct	Rn	1 0 1 0 Sz r r r *1 *2	None
2	Register indirect	@Rn	1 1 0 1 Sz r r r	None
3	Register indirect with displacement	@(d:8,Rn)	1 1 1 0 Sz r r r	Displacement (1 byte)
		@(d:16,Rn)	1 1 1 1 Sz r r r	Displacement (2 bytes)
4	Register indirect with pre-decrement	@–Rn	1 0 1 1 Sz r r r	None
	Register indirect with post-increment	@Rn+	1 1 0 0 Sz r r r	
5	Immediate	#xx:8	0 0 0 0 0 1 0 0	Immediate data (1 byte)
		#xx:16	0 0 0 0 1 1 0 0	Immediate data (2 bytes)
6	Absolute *3	@aa:8	0 0 0 0 Sz 1 0 1	1-Byte absolute address (offset from BR)
		@aa:16	0 0 0 1 Sz 1 0 1	2-Byte absolute address
7	PC-relative	disp	No EA field. Addressing mode is specified in the operation code.	1- or 2-byte displacement

Notes: * 1 Sz: Specifies the operand size.
When Sz = 0: byte operand
When Sz = 1: word operand

* 2 rrr: Register number field, specifying a general register number.
0 0 0 — R0 0 0 1 — R1 0 1 0 — R2 0 1 1 — R3
1 0 0 — R4 1 0 1 — R5 1 1 0 — R6 1 1 1 — R7

* 3 The @aa:8 addressing mode is also referred to as the short absolute addressing mode.

3.4.3 Effective Address Calculation

Table 3-8 explains how the effective address is calculated in each addressing mode.

Table 3-8 Effective Address Calculation

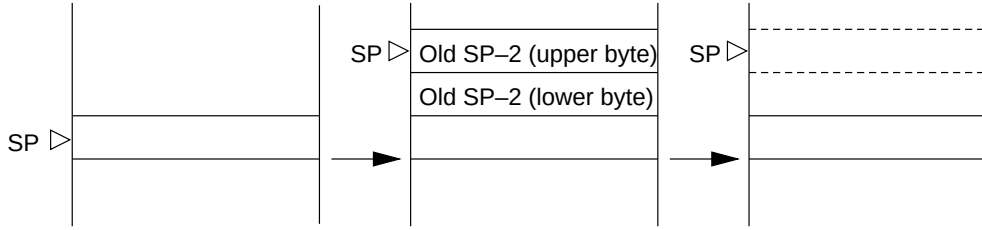
No.	Addressing Mode	Effective Address Calculation	Effective Address
1	Register direct Rn 1010Sz | rrr	—	Operand is contents of Rn
2	Register indirect @Rn 1101Sz | rrr	—	23 15 0 DP *1 | Rn Or TP or EP *2
3	Register indirect with displacement @(d:8,Rn) 1110Sz | rrr	8 Bits 15 0 Rn 15 0 Displacement with sign extension ⊕	23 15 0 DP *1 | Result Or TP or EP *2
	@(d:16,Rn) 1111Sz | rrr	16 Bits 15 0 Rn 15 0 Displacement ⊕	23 15 0 DP *1 | Result Or TP or EP *2
4	Register indirect with pre-decrement @–Rn 1011Sz | rrr	15 0 Rn 1 or 2 ⊖ Rn is decremented by –1 or –2 before instruction execution.*3*4*5	23 15 0 DP *1 | Result Or TP or EP *2
	Register indirect with post-increment @Rn+ 1100Sz | rrr	— Rn is incremented by +1 or +2 after instruction execution.*3*4*5	23 15 0 DP *1 | Rn Or TP or EP *2

Table 3-8 Effective Address Calculation (cont)

No.	Addressing Mode	Effective Address Calculation	Effective Address
5	Absolute address @aa:8 0000Sz101	—	23 15 0 H'00 | BR | EA extension data
	@aa:16 0001Sz101	—	23 15 0 DP | EA extension data
6	Immediate #xx:8 00000100	—	Operand is 1-byte EA extension data.
	#xx:16 00001100	—	Operand is 2-byte EA extension data.
7	PC-relative disp:8 No EA code Specified in OP code	8 Bits 15 0 PC 15 0 Displacement with sign extension	23 15 0 CP *1 | Result
	disp:16 No EA code Specified in OP code	16 Bits 15 0 PC 15 0 Displacement	23 15 0 CP *1 | Result

Notes: * 1 The page register is ignored in minimum mode.

* 2 The page register used in addressing modes 2, 3, and 4 depends on the general register : DP for R0, R1, R2, or R3; EP for R4 or R5; TP for R6 or R7.
* 3 Decrement by –1 for a byte operand, and by –2 for a word operand.
* 4 The pre-decrement or post-increment is always ±2 when R7 is specified, even if the operand is byte size.
* 5 The drawing below shows what happens when the @-SP and @ SP+ addressing modes are used to save and restore the stack pointer.



MOV.W SP, @–SP MOV.W @SP+, SP

3.5 Instruction Set

3.5.1 Overview

The main features of the CPU instruction set are:

- A general-register architecture.
- Orthogonality. Addressing modes and data sizes can be specified independently in each instruction.
- 1.5 addressing modes (supporting register-register and register-memory operations)
- Affinity for high-level languages, particularly C, with short formats for frequently-used instructions and addressing modes.

The CPU instruction set includes 63 types of instructions, listed by function in table 3-9.

Table 3-9 Instruction Classification

Function	Instructions	Types
Data transfer	MOV, LDM, STM, XCH, SWAP, MOVTPE, MOVFPE	7
Arithmetic operations	ADD, SUB, ADDS, SUBS, ADDX, SUBX, DADD, DSUB, MULXU, DIVXU, CMP, EXTS, EXTU, TST, NEG, CLR, TAS	17
Logic operations	AND, OR, XOR, NOT	4
Shift	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	8
Bit manipulation	BSET, BCLR, BTST, BNOT	4
Branch	Bcc*, JMP, PJMP, BSR, JSR, PJSR, RTS, PRTD, PRTS, RTD, SCB (/F, /NE, /EQ)	11
System control	TRAPA, TRAP/VS, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP, LINK, UNLK	12
	Total	63

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

Tables 3-10 to 3-16 give a concise summary of the instructions in each functional category. The MOV, ADD, and CMP instructions have special short formats, which are listed in table 3-17. For detailed descriptions of the instructions, refer to the *H8/500 Series Programming Manual*.

The notation used in tables 3-10 to 3-17 is defined below.

Operation Notation	
Rd	General register (destination)
Rs	General register (source)
Rn	General register
(EAd)	Destination operand
(EAs)	Source operand
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
CR	Control register
PC	Program counter
CP	Code page register
SP	Stack pointer
FP	Frame pointer
#IMM	Immediate data
disp	Displacement
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	AND logical
∨	OR logical
⊕	Exclusive OR logical
→	Move
↔	Exchange
¬	Not

3.5.2 Data Transfer Instructions

Table 3-10 describes the seven data transfer instructions.

Table 3-10 Data Transfer Instructions

Instruction		Size*	Function
Data transfer	MOV		(EAs) → (EAd), #IMM → (EAd)
	MOV:G	B/W	Moves data between two general registers, or between a general register and memory, or moves immediate data to a general register or memory.
	MOV:E	B	
	MOV:I	W	
	MOV:F	B/W	
	MOV:L	B/W	
	MOV:S	B/W	
	LDM	W	Stack → Rn (register list) Pops data from the stack to one or more registers.
	STM	W	Rn (register list) → stack Pushes data from one or more registers onto the stack.
	XCH	W	Rs ↔ Rd Exchanges data between two general registers.
	SWAP	B	Rd (upper byte) ↔ Rd (lower byte) Exchanges the upper and lower bytes in a general register.
	MOVTPE	B	Rn → (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.

Note: B—byte; W—word

3.5.3 Arithmetic Instructions

Table 3-11 describes the 17 arithmetic instructions.

Table 3-11 Arithmetic Instructions

Instruction		Size	Function
Arithmetic operations	ADD		Rd ± (EAs) → Rd, (EAd) ± #IMM → (EAd)
	ADD:G	B/W	Performs addition or subtraction on data in a general register and data in another general register or memory, or on immediate data and data in a general register or memory.
	ADD:Q	B/W	
	SUB	B/W	
	ADDS	B/W	
	SUBS	B/W	
	ADDX	B/W	Rd ± (EAs) ± C → Rd
	SUBX	B/W	Performs addition or subtraction with carry or borrow on data in a general register and data in another general register or memory, or on immediate data and data in a general register or memory.
	DADD	B	$(Rd)_{10} \pm (Rs)_{10} \pm C \rightarrow (Rd)_{10}$
	DSUB	B	Performs decimal addition or subtraction on data in two general registers.
	MULXU	B/W	Rd × (EAs) → Rd Performs 8-bit × 8-bit or 16-bit × 16-bit unsigned multiplication on data in a general register and data in another general register or memory, or on data in a general register and immediate data.
	DIVXU	B/W	Rd ÷ (EAs) → Rd Performs 16-bit ÷ 8-bit or 32-bit ÷ 16-bit unsigned division on data in a general register and data in another general register or memory, or on data in a general register and immediate data.
	CMP		Rn – (EAs), (EAd) – #IMM
	CMP:G	B/W	Compares data in a general register with data in another general register or memory, or with immediate data, or compares immediate data with data in memory.
	CMP:E	B	
	CMP:I	W	

Note: B—byte; W—word

Table 3-11 Arithmetic Instructions (cont)

Instruction		Size	Function
Arithmetic operations	EXTS	B	(<bit 7> of <Rd>) → (<bits 15 to 8> of <Rd>) Converts byte data in a general register to word data by extending the sign bit.
	EXTU	B	0 → (<bits 15 to 8> of <Rd>) Converts byte data in a general register to word data by padding with zero bits.
	TST	B/W	(EAd) – 0 Compares general register or memory contents with 0.
	NEG	B/W	0 – (EAd) → (EAd) Obtains the two's complement of general register or memory contents.
	CLR	B/W	0 → (EAd) Clears general register or memory contents to 0.
	TAS	B	(EAd) — 0, $(1)_2$ → (<bit 7> of <EAd>) Tests general register or memory contents, then sets the most significant bit (bit 7) to 1.

Note: B—byte; W—word

3.5.4 Logic Operations

Table 3-12 lists the four instructions that perform logic operations.

Table 3-12 Logic Operation Instructions

Instruction		Size	Function
Logical operations	AND	B/W	Rd∧(EAs) → Rd Performs a logical AND operation on a general register and another general register, memory, or immediate data.
	OR	B/W	Rd∨(EAs) → Rd Performs a logical OR operation on a general register and another general register, memory, or immediate data.
	XOR	B/W	Rd⊕(EAs) → Rd Performs a logical exclusive OR operation on a general register and another general register, memory, or immediate data.
	NOT	B/W	¬ (EAd) → (EAd) Obtains the one's complement of general register or memory contents.

Note: B—byte; W—word

3.5.5 Shift Operations

Table 3-13 lists the eight shift instructions.

Table 3-13 Shift Instructions

Instruction		Size	Function
Shift operations	SHAL	B/W	(EAd) shift → (EAd)
	SHAR	B/W	Performs an arithmetic shift operation on general register or memory contents.
	SHLL	B/W	(EAd) shift → (EAd)
	SHLR	B/W	Performs a logical shift operation on general register or memory contents.
	ROTL	B/W	(EAd) shift → (EAd)
	ROTR	B/W	Rotates general register or memory contents.
	ROTXL	B/W	(EAd) rotate through carry → (EAd)
	ROTXR	B/W	Rotates general register or memory contents through the C (carry) bit.

Note: B—byte; W—word

3.5.6 Bit Manipulations

Table 3-14 describes the four bit-manipulation instructions.

Table 3-14 Bit-Manipulation Instructions

Instruction		Size	Function
Bit manipulations	BSET	B/W	¬ (<bit-No.> of <EAd>) → Z, 1 → (<bit-No.> of <EAd>) Tests a specified bit in a general register or memory, then sets the bit to 1. The bit is specified by a bit number given in immediate data or a general register.
	BCLR	B/W	¬ (<bit-No.> of <EAd>) → Z, 0 → (<bit-No.> of <EAd>) Tests a specified bit in a general register or memory, then clears the bit to 0. The bit is specified by a bit number given in immediate data or a general register.
	BNOT	B/W	¬ (<bit-No.> of <EAd>) → Z, → (<bit-No.> of <EAd>) Tests a specified bit in a general register or memory, then inverts the bit. The bit is specified by a bit number given in immediate data or a general register.
	BTST	B/W	¬ (<bit-No.> of <EAd>) → Z Tests a specified bit in a general register or memory. The bit is specified by a bit number given in immediate data or a general register.

Note: B—byte; W—word

3.5.7 Branching Instructions

Table 3-15 describes the 11 branching instructions.

Table 3-15 Branching Instructions

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

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

JMP	—	Branches unconditionally to a specified address in the same page.
PJMP	—	Branches unconditionally to a specified address in a specified page.
BSR	—	Branches to a subroutine at a specified address in the same page.
JSR	—	Branches to a subroutine at a specified address in the same page.
PJSR	—	Branches to a subroutine at a specified address in a specified page.
RTS	—	Returns from a subroutine in the same page.

Table 3-15 Branching Instructions (cont)

Instruction		Size	Function
Branch	PRTS	—	Returns from a subroutine in a different page.
	RTD	—	Returns from a subroutine in the same page and adjusts the stack pointer.
	PRTD	—	Returns from a subroutine in a different page and adjusts the stack pointer.
	SCB/F	—	Controls a loop using a loop counter and/or a specified termination condition.
	SCB/NE	—	
	SCB/EQ	—	

3.5.8 System Control Instructions

Table 3-16 describes the 12 system control instructions.

Table 3-16 System Control Instructions

Instruction		Size	Function
System control	TRAPA	—	Generates a trap exception with a specified vector number.
	TRAP/VS	—	Generates a trap exception if the V bit is set to 1 when the instruction is executed.
	RTE	—	Returns from an exception-handling routine.
	LINK	—	FP → @–SP; SP → FP; SP + #IMM → SP Creates a stack frame.
	UNLK	—	FP → SP; @SP+ → FP Deallocates a stack frame created by the LINK instruction.
	SLEEP	—	Causes a transition to the power-down state.
	LDC	B/W*	(EAs) → CR Moves immediate data or general register or memory contents to a specified control register.
	STC	B/W*	CR → (EAd) Moves control register data to a specified general register or memory location.
	ANDC	B/W*	CR ∧ #IMM → CR Logically ANDs a control register with immediate data.
	ORC	B/W*	CR ∨ #IMM → CR Logically ORs a control register with immediate data.
	XORC	B/W*	CR ⊕ #IMM → CR Logically exclusive-ORs a control register with immediate data.
	NOP	—	PC + 1 → PC No operation. Only increments the program counter.

* The size depends on the control register.

Note on Stack Operation by LDC and STC Instructions of H8/500 CPU

When using the LDC and STC instructions to stack and unstack the BR, CCR, TP, DP, and EP control registers in the H8/500 family, note the following point.

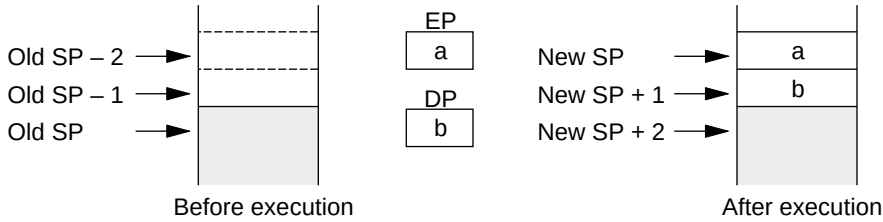
H8/500 hardware does not permit byte access to the stack. If the LDC.B or STC.B assembler mnemonic is coded with the @R7 + (@SP+) or @–R7 (@–SP) addressing mode, the stack-pointer addressing mode takes precedence and hardware automatically performs word access.

Specifically, the LDC.B and STC.B instructions are executed as follows.

The following applies only to the stack-pointer addressing modes. In addressing modes that do not use the stack pointer, byte data access is performed as specified by the assembler mnemonic.

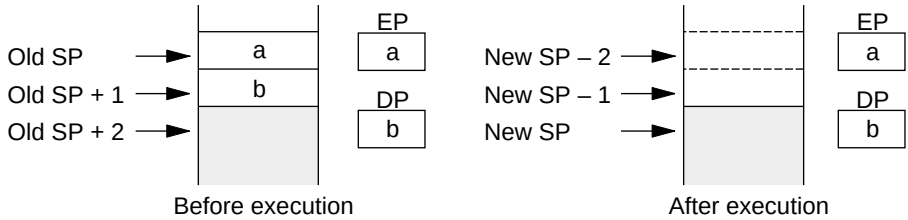
(1) `STC.B EP, @–SP`

When word data access is applied to EP, both EP and DP are accessed. This instruction stores EP at address SP (old) –2, and DP at address SP (old) –1.



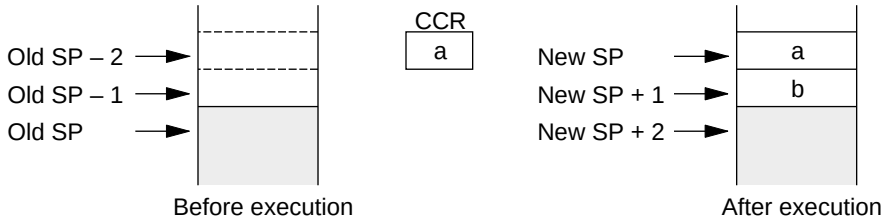
(2) `LDC.B @SP+, EP`

When word data access is applied to EP, both EP and DP are accessed. This instruction loads EP from address SP (old), and DP from address SP (old) +1, updating the DP value as well as the EP value.



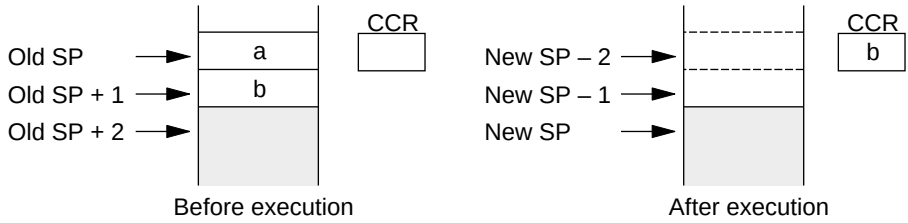
(3) `STC.B CCR, @–SP`

When word data access is applied to CCR, only CCR is accessed. This instruction stores identical CCR contents at both address SP (old) –2 and address SP (old) –1.



(4) `LDC.B @SP+, CCR`

When word data access is applied to CCR, only CCR is accessed. This instruction loads CCR from address SP (old) +1. Note that the value in address SP (old) is not loaded.



BR, DP, and TP are accessed in the same way as CCR. When DP is specified, both EP and DP are accessed, but when CCR, BR, DP, or TP is specified, only the specified register is accessed.

3.5.9 Short-Format Instructions

The ADD, CMP, and MOV instructions have special short formats. Table 3-17 lists these short formats together with the equivalent general formats.

The short formats are a byte shorter than the corresponding general formats, and most of them execute one state faster.

Table 3-17 Short-Format Instructions and Equivalent General Formats

Short-Format Instruction	Length	Execution States *2	Equivalent General-Format Instruction	Length	Execution States *2
ADD:Q #xx,Rd *1	2	2	ADD:G #xx:8,Rd	3	3
CMP:E #xx:8,Rd	2	2	CMP:G.B #xx:8,Rd	3	3
CMP:I #xx:16,Rd	3	3	CMP:G.W #xx:16,Rd	4	4
MOV:E #xx:8,Rd	2	2	MOV:G.B #xx:8,Rd	3	3
MOV:I #xx:16,Rd	3	3	MOV:G.W #xx:16,Rd	4	4
MOV:L @aa:8,Rd	2	5	MOV:G @aa:8,Rd	3	5
MOV:S Rs,@aa:8	2	5	MOV:G Rs,@aa:8	3	5
MOV:F @(d:8,R6),Rd	2	5	MOV:G @(d:8,R6),Rd	3	5
MOV:F Rs,@(d:8,R6)	2	5	MOV:G Rs,@(d:8,R6)	3	5

Notes: * 1 The ADD:Q instruction accepts other destination operands in addition to a general register, but the immediate data value (#xx) is limited to ±1 or ±2.

* 2 Number of execution states for access to on-chip memory.

3.6 Operating Modes

The CPU operates in one of two modes: the minimum mode or the maximum mode. These modes are selected by the mode pins (MD_2 to MD_0).

3.6.1 Minimum Mode

The minimum mode supports a maximum address space of 64 kbytes. The page registers are ignored. Instructions that branch across page boundaries (PJMP, PJSR, PRTS, PRTD) are invalid.

3.6.2 Maximum Mode

In the maximum mode the page registers are valid, expanding the maximum address space to 1 Mbyte.

The address space is divided into 64-kbyte pages. The pages are separate; it is not possible to move continuously across a page boundary.

It is possible to move from one page to another with branching instructions (PJMP, PJSR, PRTS, PRTD). The TRAPA instruction and branches to interrupt-handling routines can also jump across page boundaries. It is not necessary for a program to be contained in a single 64-kbyte page.

When data access crosses a page boundary, the program must rewrite the page register before it can access the data in the next page.

For further information on the operating modes, see section 2, “MCU Operating Modes and Address Space.”

3.7 Basic Operational Timing

3.7.1 Overview

The CPU operates on a system clock (ø) which is created by dividing an oscillator frequency (fosc) by two. One period of the system clock is referred to as a “state.” The CPU accesses memory in a cycle consisting of 2 or 3 states. The CPU uses different methods to access on-chip memory, the on-chip register field, and external devices.

Access to On-Chip Memory (RAM, ROM): For maximum speed, access to on-chip memory (RAM, ROM) is performed in two states, using a 16-bit-wide data bus.

Figure 3-6 shows the on-chip memory access cycle. Figure 3-7 indicates the pin states. The bus control output signals go to the nonactive state during the access.

Access to On-Chip Register Field (Addresses H'FE80 to H'FFFF): The access cycle consists of three states. The data bus is 8 bits wide.

Figure 3-8 shows the on-chip supporting module access cycle. Figure 3-9 indicates the pin states.

Access to External Devices: The access cycle consists of three states. The data bus is 8 bits wide. Figure 3-10 (a) and (b) shows the external access cycle. Additional wait states (Tw) can be inserted by the wait-state controller (WSC).

3.7.2 On-Chip Memory Access Cycle

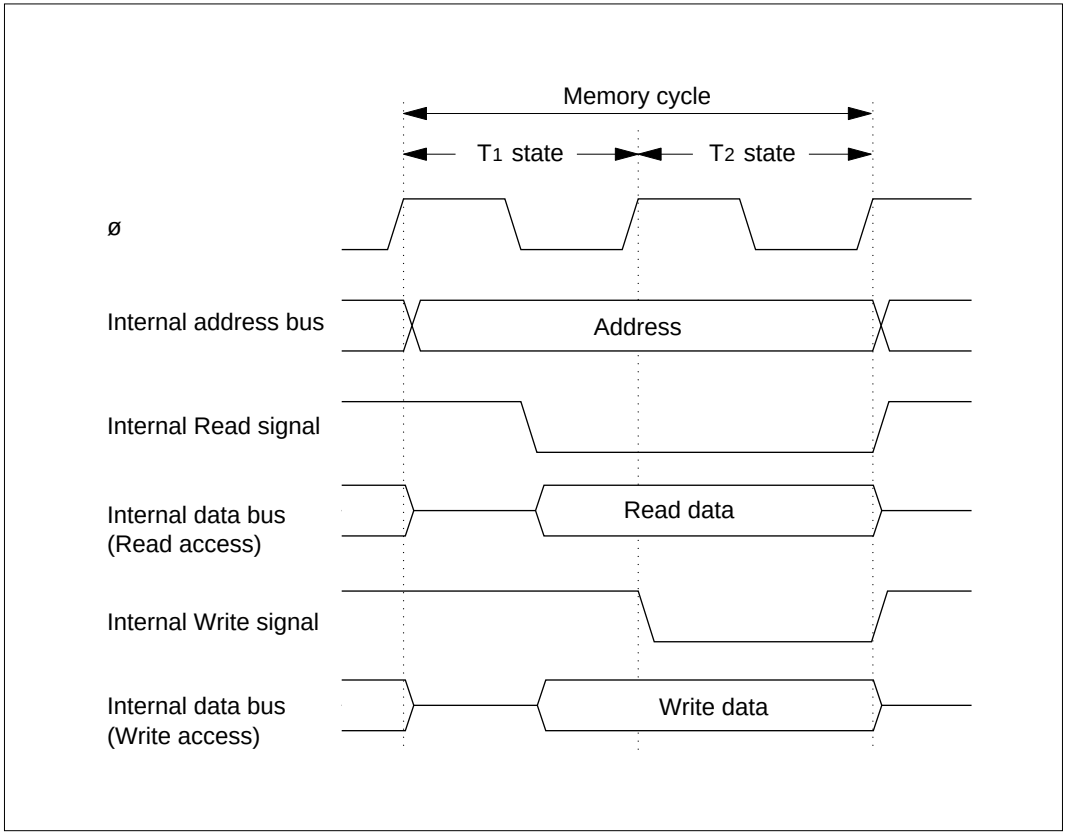


Figure 3-6 On-Chip Memory Access Timing

3.7.3 Pin States during On-Chip Memory Access

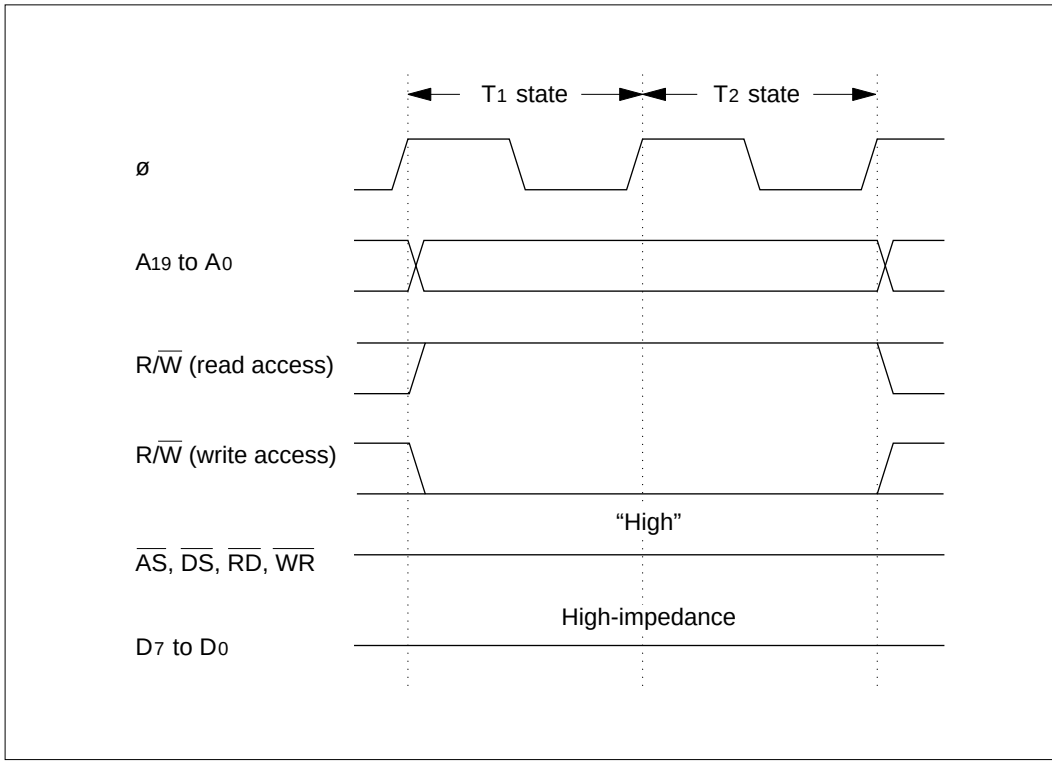


Figure 3-7 Pin States during Access to On-Chip Memory

3.7.4 Register Field Access Cycle (Addresses H'FE80 to H'FFFF)

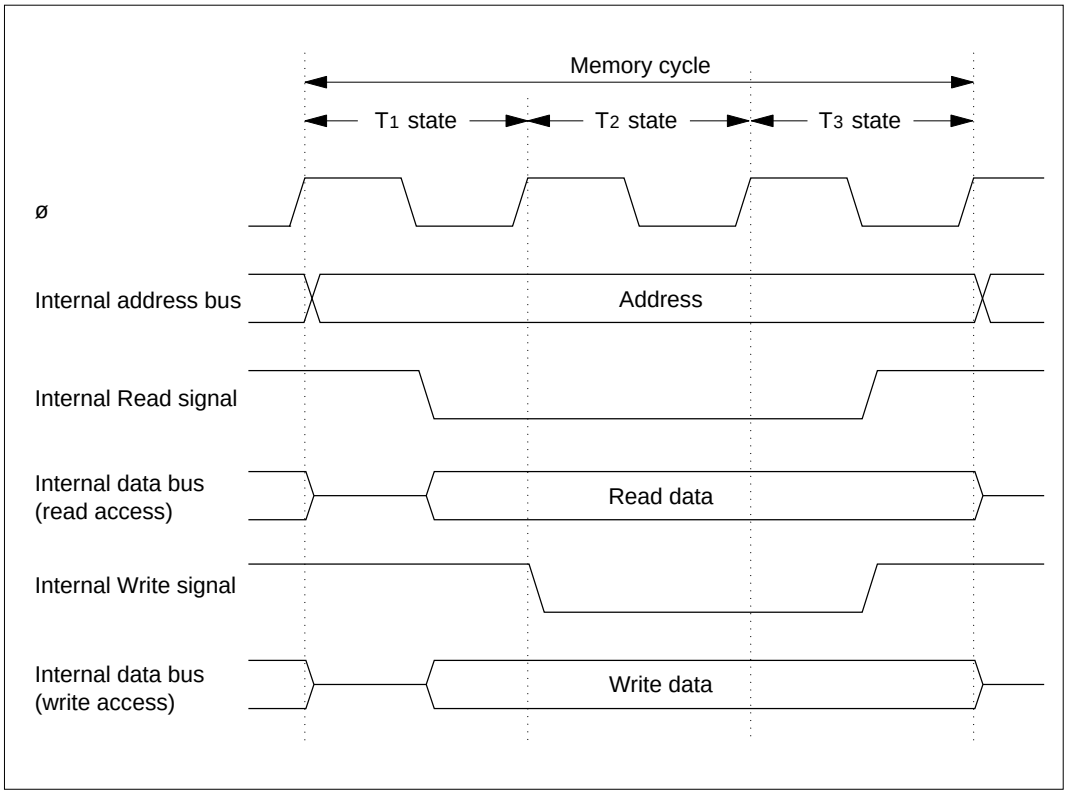


Figure 3-8 Register Field Access Timing

3.7.5 Pin States during Register Field Access (Addresses H'FE80 to H'FFFF)

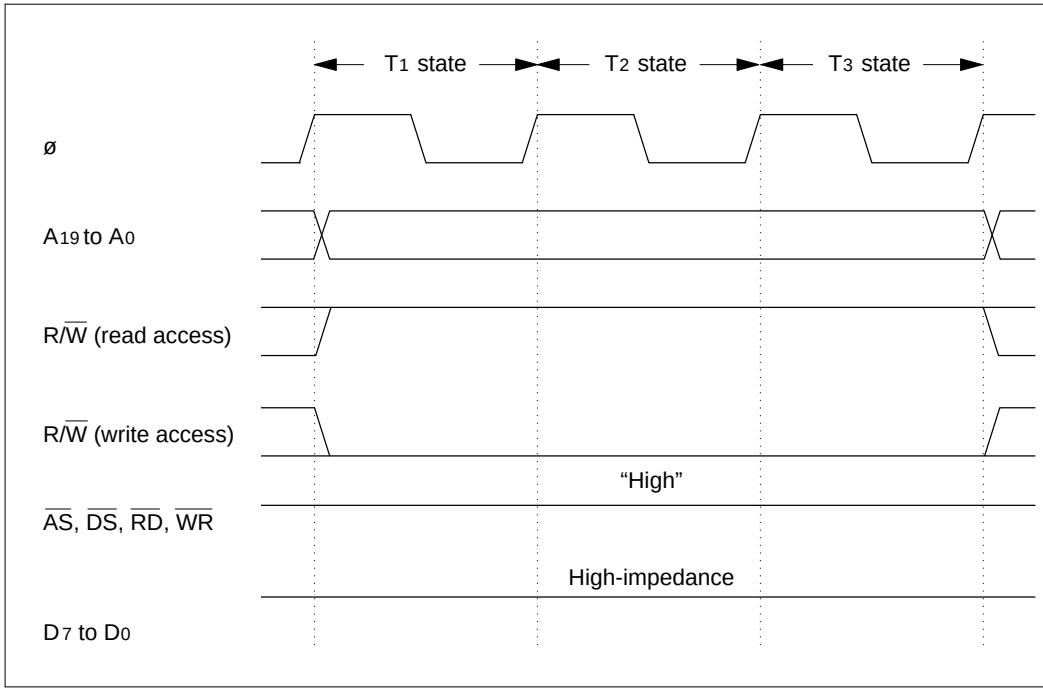


Figure 3-9 Pin States during Register Field Access

3.7.6 External Access Cycle

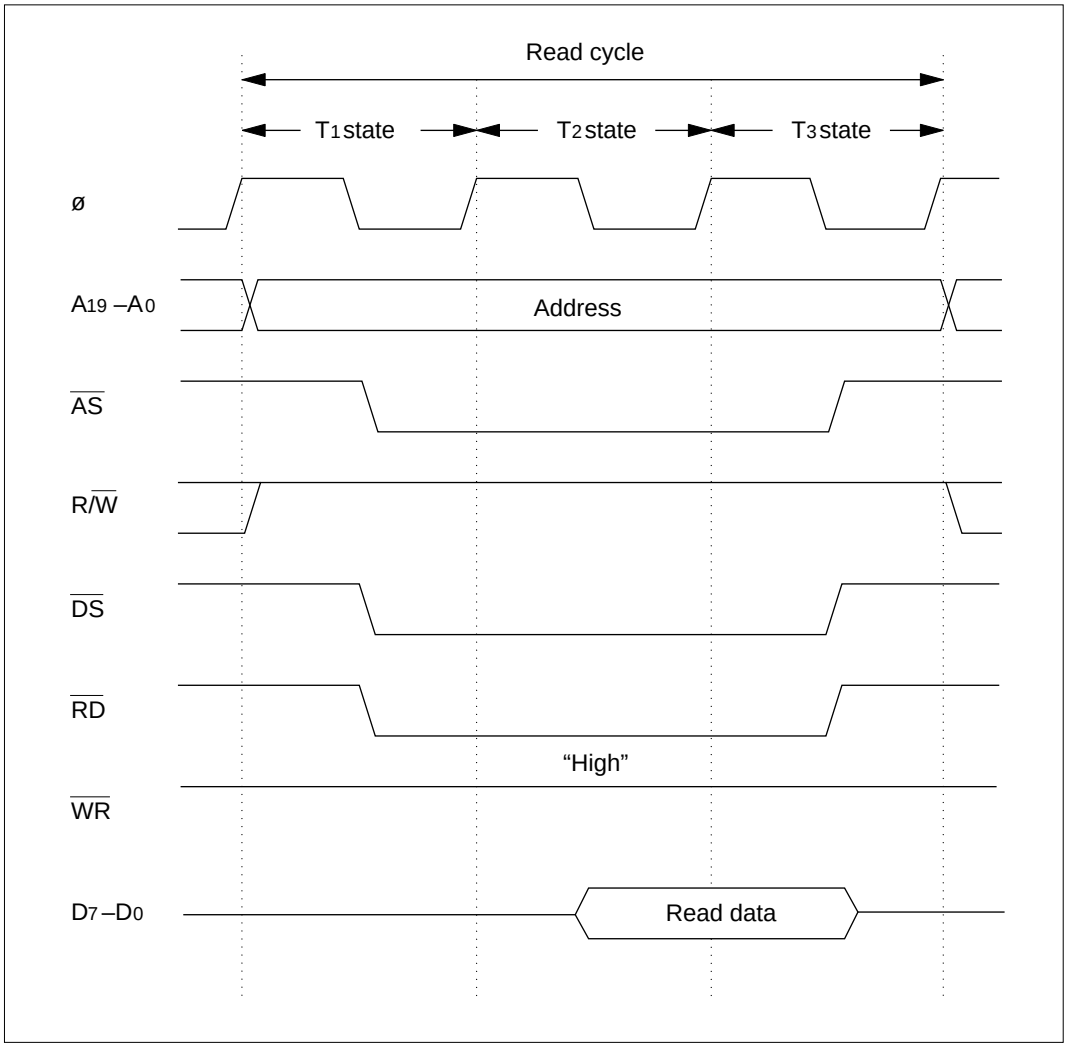


Figure 3-10 (a) External Access Cycle (Read Access)

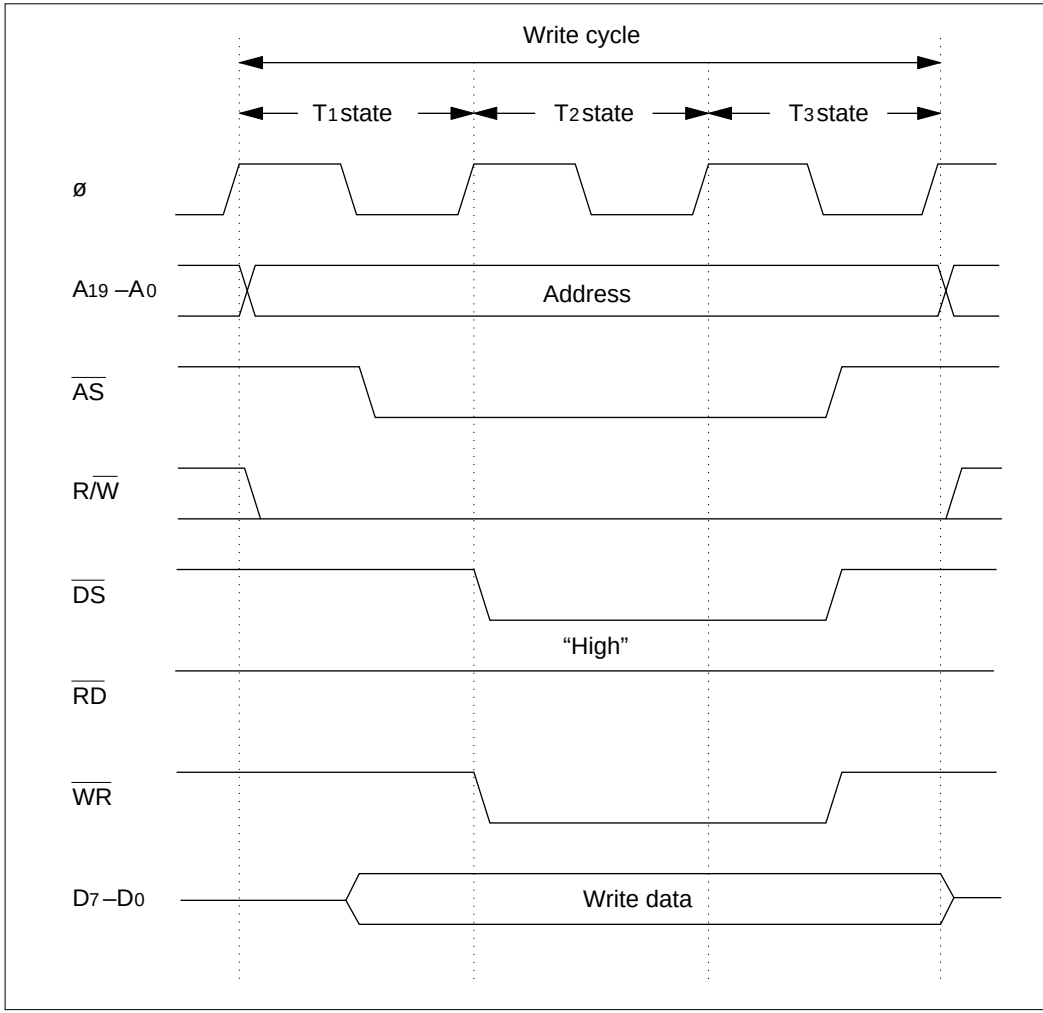


Figure 3-10 (b) External Access Cycle (Write Access)

3.8 CPU States

3.8.1 Overview

The CPU has five states: the program execution state, exception-handling state, bus-released state, reset state, and power-down state. The power-down state is further divided into 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.

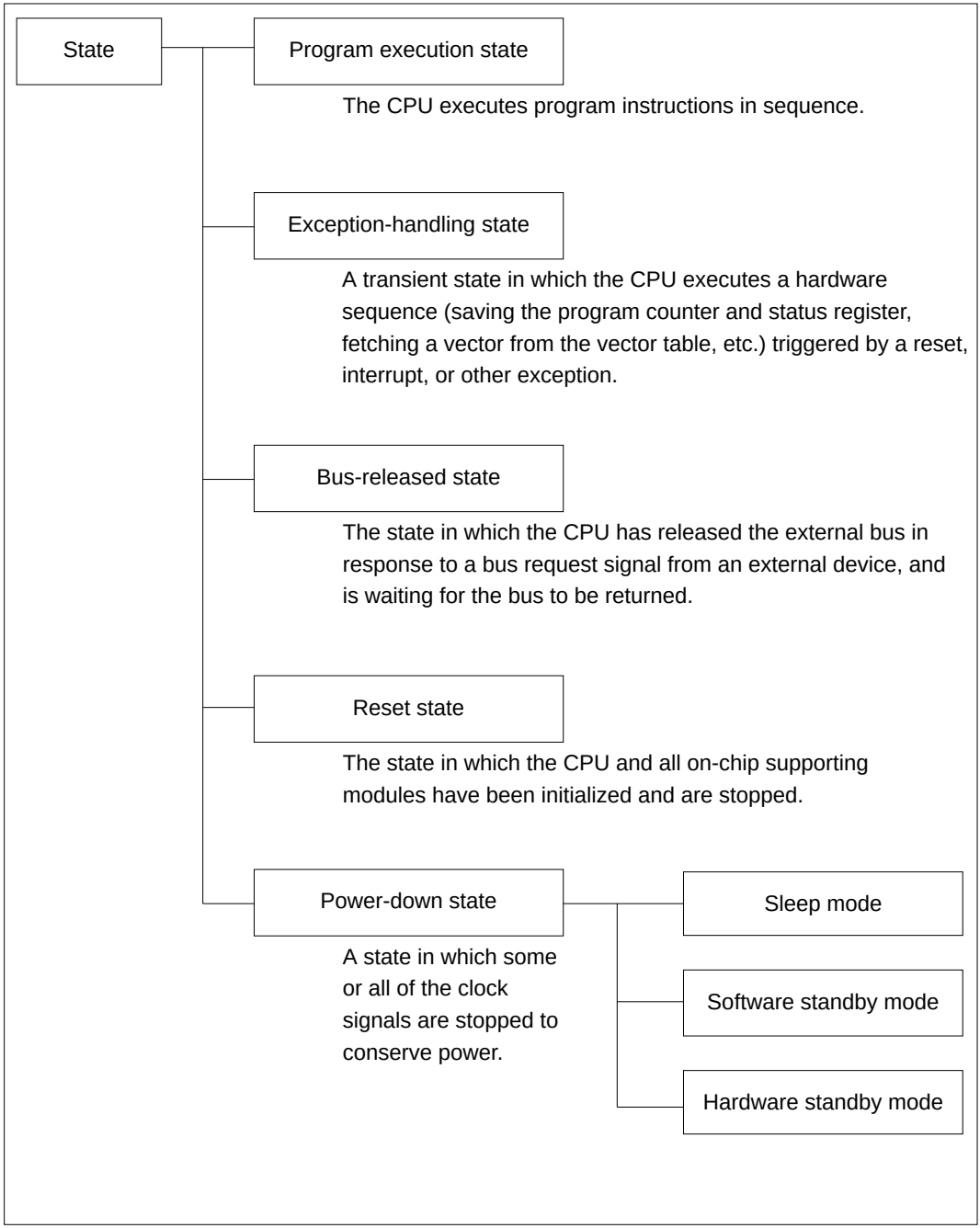


Figure 3-11 Operating States

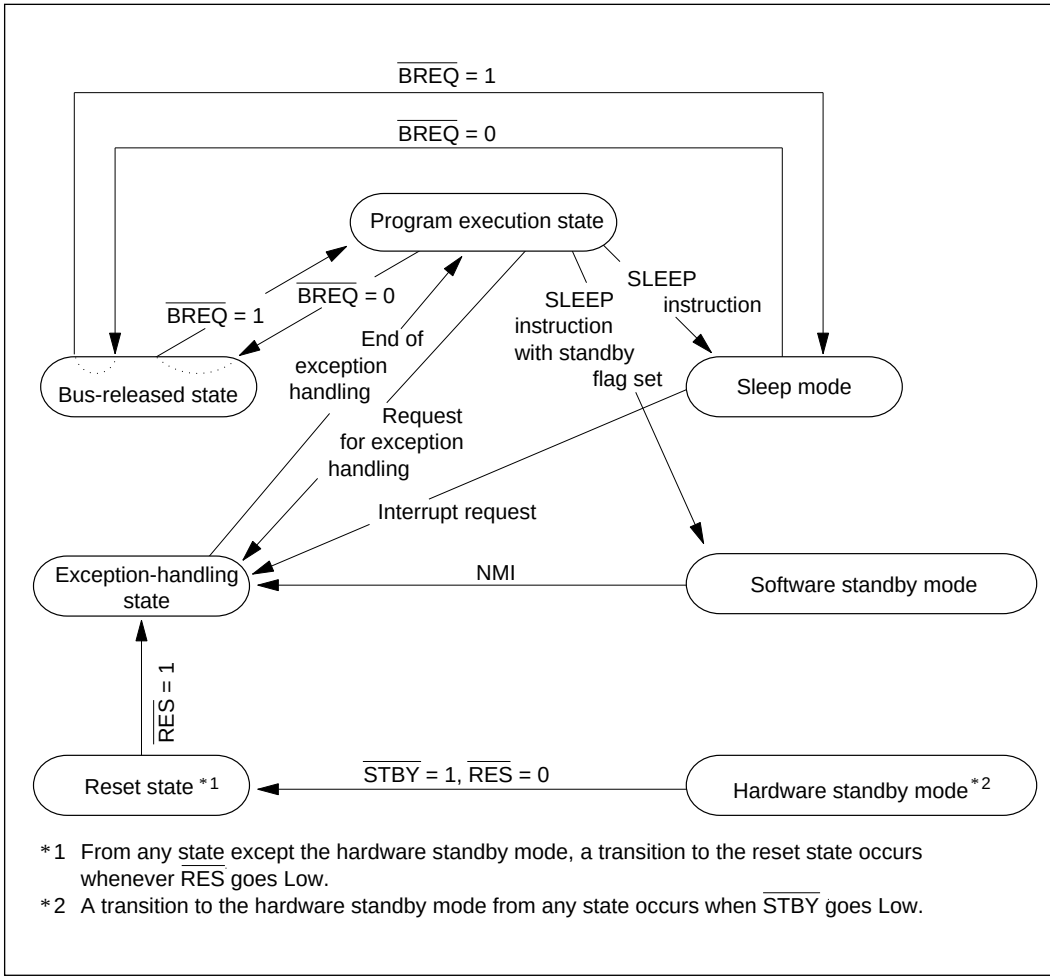


Figure 3-12 State Transitions

3.8.2 Program Execution State

In this state the CPU executes program instructions in normal sequence.

3.8.3 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU alters the normal program flow due to an interrupt, trap instruction, address error, or other exception. 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 status register (in minimum mode) or program counter, code page register, and status register (in maximum mode) to the stack.
2. Clears the T bit in the status register to 0.
3. Fetches the start address of the exception-handling routine from the exception 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.

3.8.4 Bus-Released State

When so requested, the CPU can grant control of the external bus to an external device. While an external device has the bus right, the CPU is said to be in the bus-released state. The bus right is controlled by two pins:

- $\overline{\text{BREQ}}$: Input pin for the Bus Request signal from an external device
- $\overline{\text{BACK}}$: Output pin for the Bus Request Acknowledge signal from the CPU, indicating that the CPU has released the bus

The procedure by which the CPU enters and leaves the bus-released state is:

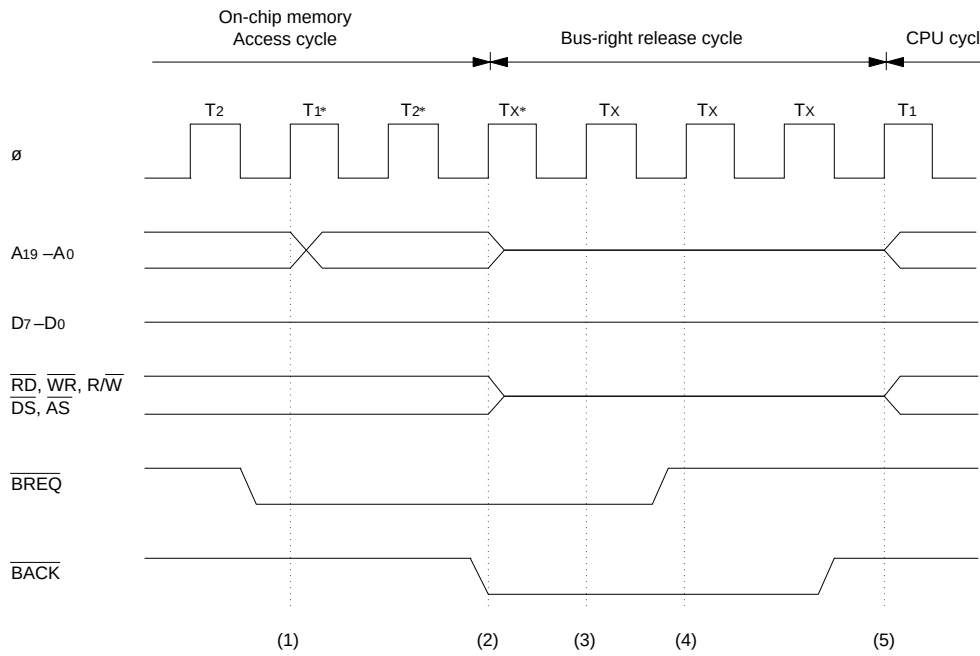
1. The CPU receives a Low $\overline{\text{BREQ}}$ signal from an external device.
2. The CPU places the address bus pins ($A_{19} – A_0$), data bus pins ($D_7 – D_0$) and bus control pins ($\overline{\text{RD}}$, $\overline{\text{WR}}$, R/$\overline{\text{W}}$, $\overline{\text{DS}}$, and $\overline{\text{AS}}$) in the high-impedance state, sets the $\overline{\text{BACK}}$ pin to the Low level to indicate that it has released the bus, then halts.
3. The external device that requested the bus (with the $\overline{\text{BREQ}}$ signal) becomes the bus master. It can use the data bus and address bus. The external device is responsible for manipulating the bus control signals ($\overline{\text{RD}}$, $\overline{\text{WR}}$, R/$\overline{\text{W}}$, $\overline{\text{DS}}$, and $\overline{\text{AS}}$).
4. When the external device finishes using the bus, it clears the $\overline{\text{BREQ}}$ signal to the High level. The CPU then reassumes control of the bus and returns to the program execution state.

Bus Release Timing: The CPU can release the bus right at the following times:

1. The $\overline{\text{BREQ}}$ signal is sampled during every memory access cycle (instruction prefetch or data read/write). If $\overline{\text{BREQ}}$ is Low, the CPU releases the bus right at the end of the cycle. (In word data access to external memory or an address from H'FE80 to H'FFFF, the CPU does not release the bus right until it has accessed both the upper and lower data bytes.)
2. During execution of the MULXU and DIVXU instructions, since considerable time may pass without an instruction prefetch or data read/write, $\overline{\text{BREQ}}$ is also sampled at internal machine cycles, and the bus right is released if $\overline{\text{BREQ}}$ is Low.
3. The bus right can also be released in the sleep mode.

The CPU does not recognize interrupts while the bus is released.

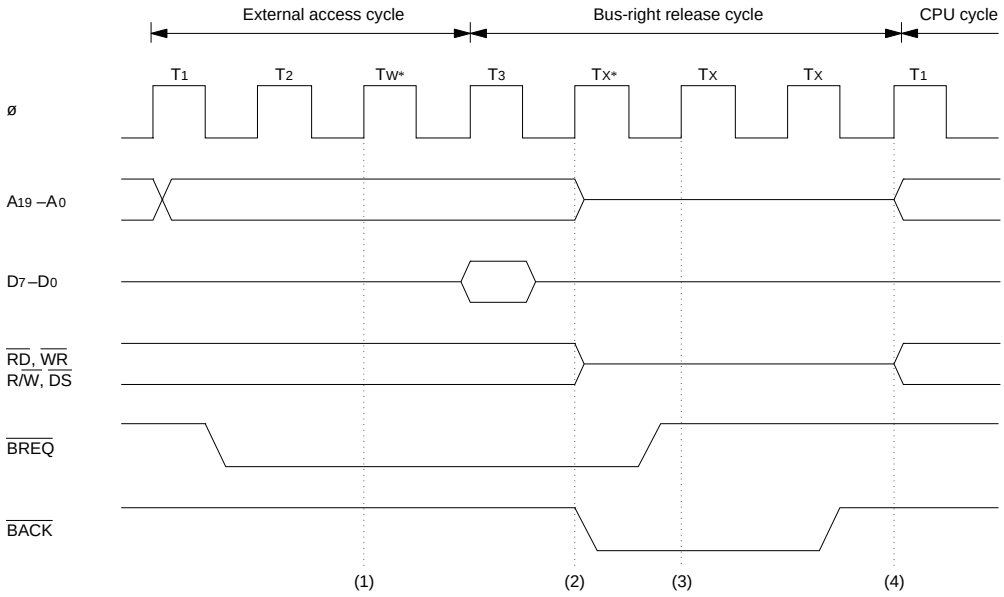
Timing Charts: Timing charts of the operation by which the bus is released are shown in figure 3-13 for the case of bus release during an on-chip memory read cycle, in figure 3-14 for bus release during an external memory read cycle, and in figure 3-15 for bus release while the CPU is performing an internal operation.



(1) The $\overline{\text{BREQ}}$ pin is sampled at the start of the T_1 state and the Low level is detected.
(2) At the end of the memory access cycle, the $\overline{\text{BACK}}$ pin goes Low and the CPU releases the bus.
(3) While the bus is released, the $\overline{\text{BREQ}}$ pin is sampled at each Tx state.
(4) A High level is detected at the $\overline{\text{BREQ}}$ pin.
(5) The $\overline{\text{BACK}}$ pin is returned to the High level, ending the bus-right release cycle.

* T_1 and T_2: On-chip memory access states.
Tx : Bus-right released state.

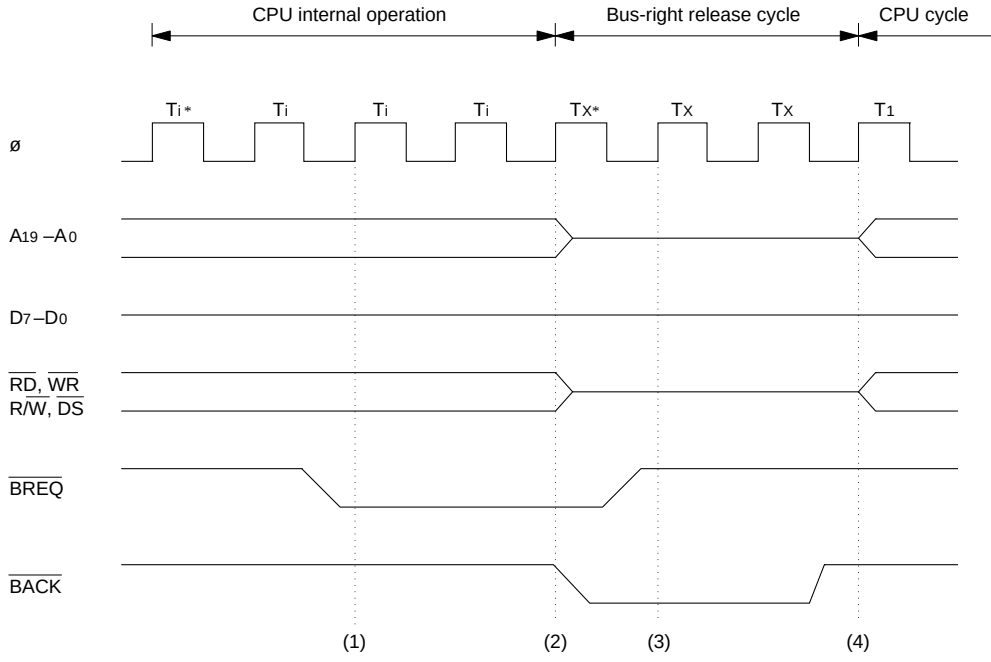
Figure 3-13 Bus-Right Release Cycle (During On-Chip Memory Access Cycle)



(1) The $\overline{\text{BREQ}}$ pin is sampled at the start of the Tw state and the Low level is detected.
(2) At the end of the external access cycle, the $\overline{\text{BACK}}$ pin goes Low and the CPU releases the bus.
(3) The $\overline{\text{BREQ}}$ pin is sampled at the Tx state and a High level is detected.
(4) The $\overline{\text{BACK}}$ pin is returned to the High level, ending the bus-right release cycle.

* Tw : Wait state.
Tx : Bus-right released state.

Figure 3-14 Bus-Right Release Cycle (During External Access Cycle)



(1) The $\overline{\text{BREQ}}$ pin is sampled at the start of a T_I state and the Low level is detected.
(2) At the end of the internal operation cycle, the $\overline{\text{BACK}}$ pin goes Low and the CPU releases the bus.
(3) The $\overline{\text{BREQ}}$ pin is sampled at the T_X state and a High level is detected.
(4) The $\overline{\text{BACK}}$ pin is returned to the High level, ending the bus-right release cycle.

* T_I : Internal CPU operation state.
T_X : Bus-right released state.

Figure 3-15 Bus-Right Release Cycle (During Internal CPU Operation)

Notes: The $\overline{\text{BREQ}}$ signal must be held Low until $\overline{\text{BACK}}$ goes Low. If $\overline{\text{BREQ}}$ returns to the High level before $\overline{\text{BACK}}$ goes Low, the bus release operation may be executed incorrectly.

To leave the bus-released state, the High level at the $\overline{\text{BREQ}}$ pin must be sampled two times. If $\overline{\text{BREQ}}$ returns to Low before it is sampled two times, the bus released cycle will not end.

The bus release operation is enabled only when the BRLE bit in the port 1 control register (P1CR) is set to 1. When this bit is cleared to 0 (its initial value), the $\overline{\text{BREQ}}$ and $\overline{\text{BACK}}$ pins are used for general-purpose input and output, as $P1_3$ and $P1_2$.

An instruction that sets the BRLE bit is: `BSET.B #3, @H'FEFC`

Note the following point when using the bus release function.

If the $\overline{\text{BREQ}}$ signal is asserted and an interrupt is requested simultaneously during execution of the SLEEP instruction, the $\overline{\text{BACK}}$ signal may fail to be output even though the CPU has released the bus. This may cause the system to stop for the interval during which $\overline{\text{BREQ}}$ is asserted, with no device in control of the bus. The interrupts that can cause this state include NMI, IRQ, and all the interrupts from on-chip supporting modules. When the $\overline{\text{BREQ}}$ signal is deasserted, ending this state, the CPU takes control of the bus again and resumes normal instruction execution.

The following methods can be used to avoid entering this state.

Method 1: If the $\overline{\text{BREQ}}$ signal is used, do not use the SLEEP instruction.

Method 2: Disable the $\overline{\text{BREQ}}$ signal during execution of the SLEEP instruction. This can be done by clearing the bus release enable bit (BRLE) in the port 1 control register (P1CR) to 0 immediately before executing the SLEEP instruction. (When the BRLE bit is cleared, low inputs on the $\overline{\text{BREQ}}$ line are not latched on-chip.) Place instructions to set the BRLE bit to 1 at the beginning of interrupt-handling routines. If the data transfer controller (DTC) is used, place an instruction to set the BRLE bit immediately after the SLEEP instruction.

If method 2 is used, $\overline{\text{BREQ}}$ inputs will be ignored while the chip is in sleep mode.

(Coding example)

Main Program

```
BCLR.B  #3, @SYSCR1
SLEEP
BSET.B  #3, @SYSCR1
```

Interrupt-Handling Routine

```
BSET.B  #3, @SYSCR1

RTE
```

3.8.5 Reset State

In the reset state, the CPU and all on-chip supporting modules are initialized and placed in the stopped state. The CPU enters the reset state whenever the $\overline{\text{RES}}$ pin goes Low, unless the CPU is currently in the hardware standby mode. It remains in the reset state until the $\overline{\text{RES}}$ pin goes High.

See section 4.2, “Reset,” for further information on the reset state.

3.8.6 Power-Down State

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

See section 18, “Power-Down State,” for further information.

3.9 Programming Notes

3.9.1 Restriction on Address Location

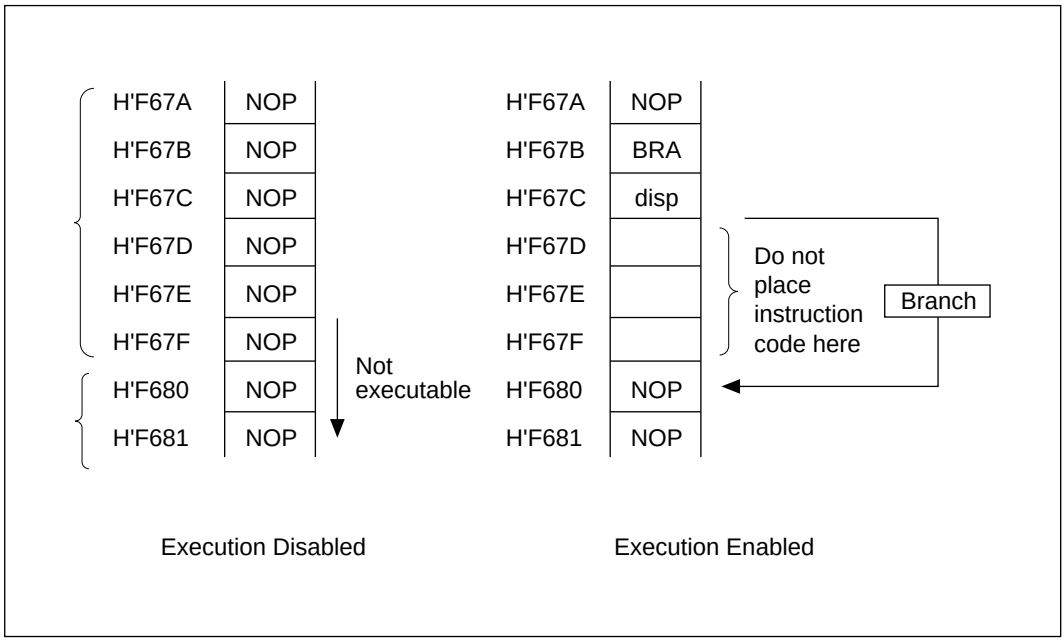
The following restriction applies when instructions are located in on-chip RAM.

- Restriction

  Instruction execution cannot proceed continuously from an external address to on-chip RAM.

- Solution

  To execute instructions located in on-chip RAM, use a branch instruction (examples: Bcc, JMP, etc.) to branch to the first instruction located in on-chip RAM. Do not place instruction code in the last three bytes of external memory (H'F67D to H'F67F).



Section 4 Exception Handling

4.1 Overview

4.1.1 Types of Exception Handling and Their Priority

As indicated in table 4-1 (a) and (b), exception handling can be initiated by a reset, address error, trace, interrupt, or instruction. An instruction initiates exception handling if the instruction is an invalid instruction, a trap instruction, or a DIVXU instruction with zero divisor. Exception handling begins with a hardware exception-handling sequence which prepares for the execution of a user-coded software exception-handling routine.

There is a priority order among the different types of exceptions, as shown in table 4-1 (a). If two or more exceptions occur simultaneously, they are handled in their order of priority. An instruction exception cannot occur simultaneously with other types of exceptions.

Table 4-1 (a) Exceptions and Their Priority

Priority	Exception Type	Source	Detection Timing	Start of Exception-Handling Sequence
High	Reset	External, internal	$\overline{\text{RES}}$ Low-to-High transition	Immediately
↕	Address error	Internal	Instruction fetch or data read/write bus cycle	End of instruction execution
	Trace	Internal	End of instruction execution, if T = 1 in status register	End of instruction execution
	Interrupt	External, internal	End of instruction execution or end of exception-handling sequence	End of instruction execution
Low				

Table 4-1 (b) Instruction Exceptions

Exception Type	Start of Exception-Handling Sequence
Invalid instruction	Attempted execution of instruction with undefined code
Trap instruction	Started by execution of trap instruction
Zero divide	Attempted execution of DIVXU instruction with zero divisor

4.1.2 Hardware Exception-Handling Sequence

The hardware exception-handling sequence varies depending on the type of exception. When exception handling is initiated by a factor other than a reset, the CPU:

1. Saves the program counter and status register (in minimum mode) or program counter, code page register, and status register (in maximum mode) to the stack.
2. Clears the T bit in the status register to 0.
3. Fetches the start address of the exception-handling routine from the exception vector table.
4. Branches to that address.

For an interrupt, the CPU also alters the interrupt mask level in bits I2 to I0 of the status register.

For a reset, step 1 is omitted. See section 4.2, “Reset”, for the full reset sequence.

4.1.3 Exception Factors and Vector Table

The factors that initiate exception handling can be classified as shown in figure 4-1.

The starting addresses of the exception-handling routines for each factor are contained in an exception vector table located in the low addresses of page 0. The vector addresses are listed in table 4-2. Note that there are different addresses for the minimum and maximum modes.

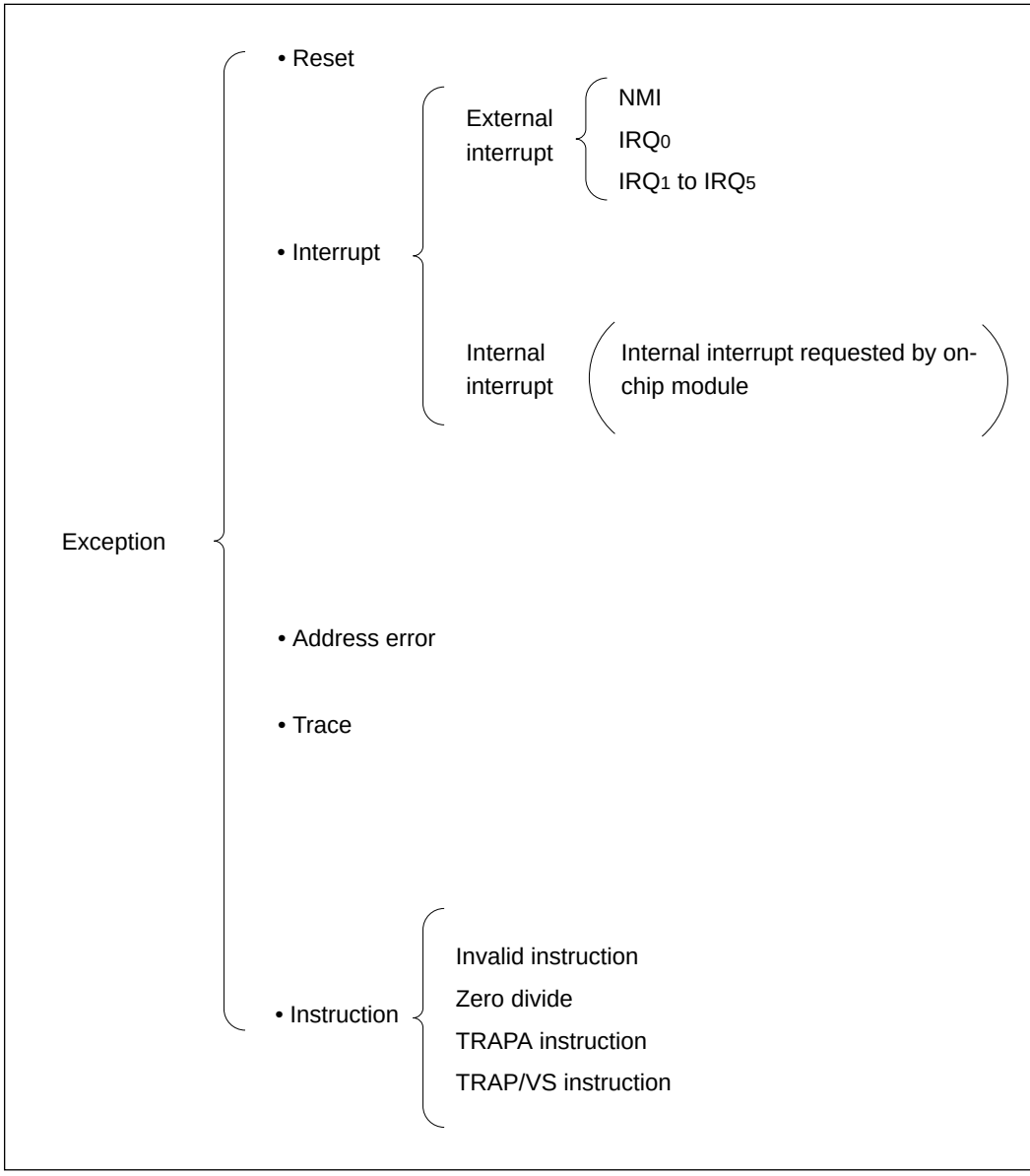


Figure 4-1 Types of Factors Causing Exception Handling

Table 4-2 Exception Vector Table

Type of Exception		Vector Address	
		Minimum Mode	Maximum Mode *1
Reset (initialize PC)		H'0000 to H'0001	H'0000 to H'0003
— (Reserved for system)		H'0002 to H'0003	H'0004 to H'0007
Invalid instruction		H'0004 to H'0005	H'0008 to H'000B
DIVXU instruction (zero divide)		H'0006 to H'0007	H'000C to H'000F
TRAP/VS instruction		H'0008 to H'0009	H'0010 to H'0013
— (Reserved for system)		H'000A to H'000B to H'000E to H'000F	H'0014 to H'0017 to H'001C to H'001F
Address error		H'0010 to H'0011	H'0020 to H'0023
Trace		H'0012 to H'0013	H'0024 to H'0027
— (Reserved for system)		H'0014 to H'0015	H'0028 to H'002B
Nonmaskable external interrupt (NMI)		H'0016 to H'0017	H'002C to H'002F
— (Reserved for system)		H'0018 to H'0019 to H'001E to H'001F	H'0030 to H'0033 to H'003C to H'003F
TRAPA instruction (16 vectors)		H'0020 to H'0021 to H'003E to H'003F	H'0040 to H'0043 to H'007C to H'007F
External interrupts	IRQ_0	H'0040 to H'0041	H'0080 to H'0083
	IRQ_1	H'0048 to H'0049	H'0090 to H'0093
	IRQ_2	H'0050 to H'0051	H'00A0 to H'00A3
	IRQ_3	H'0052 to H'0053	H'00A4 to H'00A7
	IRQ_4	H'0058 to H'0059	H'00B0 to H'00B3
	IRQ_5	H'005A to H'005B	H'00B4 to H'00B7
Internal interrupts *2		H'0060 to H'0061 to H'0098 to H'0099	H'00C0 to H'00C3 to H'0130 to H'0133

Notes: * 1. The exception vector table is located at the beginning of page 0.

* 2. For details of the internal interrupt vectors, see table 5-2.

4.2 Reset

4.2.1 Overview

A reset has the highest exception-handling priority.

When the $\overline{\text{RES}}$ pin goes Low, all current processing is halted and the H8/534 or H8/536 chip enters the reset state.

A reset initializes the internal status of the CPU and the registers of the on-chip supporting modules and I/O ports. It does not initialize the on-chip RAM.

When the $\overline{\text{RES}}$ pin returns from Low to High, the chip comes out of the reset state and begins executing the hardware reset sequence.

4.2.2 Reset Sequence

The Reset signal is detected when the $\overline{\text{RES}}$ pin goes Low.

To ensure that the H8/534 or H8/536 is reset, the $\overline{\text{RES}}$ pin should be held Low for at least 20 ms at power-up. To reset the H8/534 or H8/536 during operation, the $\overline{\text{RES}}$ pin should be held Low for at least 6 system clock cycles. See table D-1, “Status of Ports” in appendix D for the status of other pins in the reset state.

When the $\overline{\text{RES}}$ pin returns to the High state after being held Low for the necessary time, the hardware reset exception-handling sequence begins, during which:

1. In the status register (SR), the T bit is cleared to disable the trace mode, and the interrupt mask level (bits I2 to I0) is set to 7. A reset disables all interrupts, including NMI.
2. The CPU loads the reset start address from the vector table into the program counter and begins executing the program at that address.

The contents of the vector table differs between minimum mode and maximum mode as indicated in figure 4-2. This affects step 3 as follows:

Minimum Mode: One word is copied from addresses H'0000 and H'0001 in the vector table to the program counter. Program execution then begins from the address in the program counter (PC).

Maximum Mode: Two words are read from addresses H'0000 to H'0003 in the vector table. The byte in address H'0000 is ignored. The byte in address H'0001 is copied to the code page register (CP). The contents of addresses H'0002 and H'0003 are copied to the program counter. Program execution starts from the address indicated by the code page register and program counter.

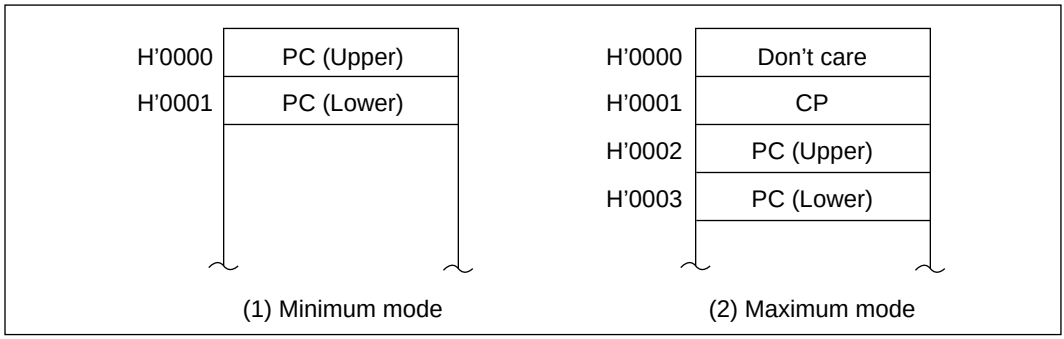


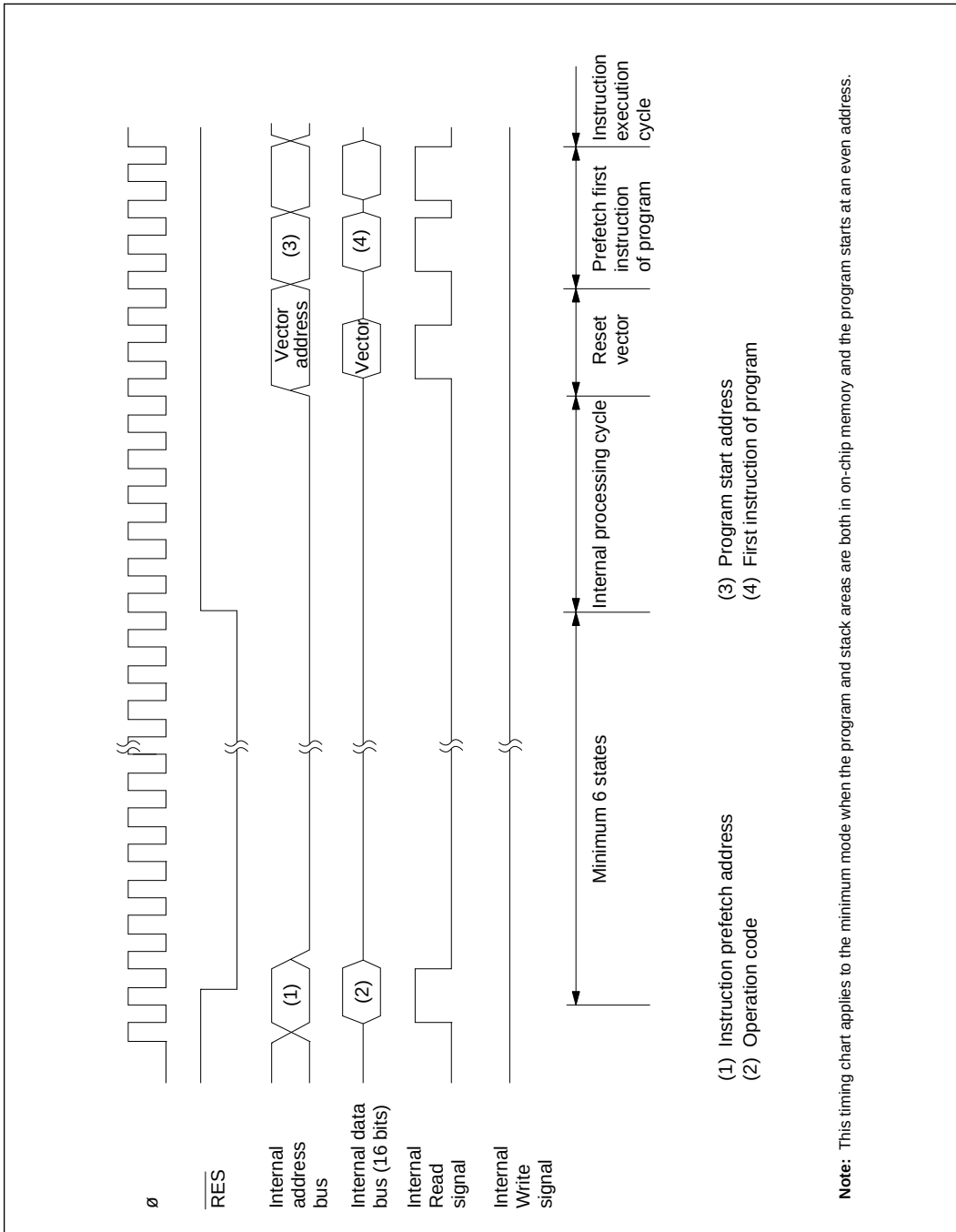
Figure 4-2 Reset Vector

Figure 4-3 shows the timing of the reset sequence in minimum mode. Figure 4-4 shows the timing of the reset sequence in maximum mode.

4.2.3 Stack Pointer Initialization

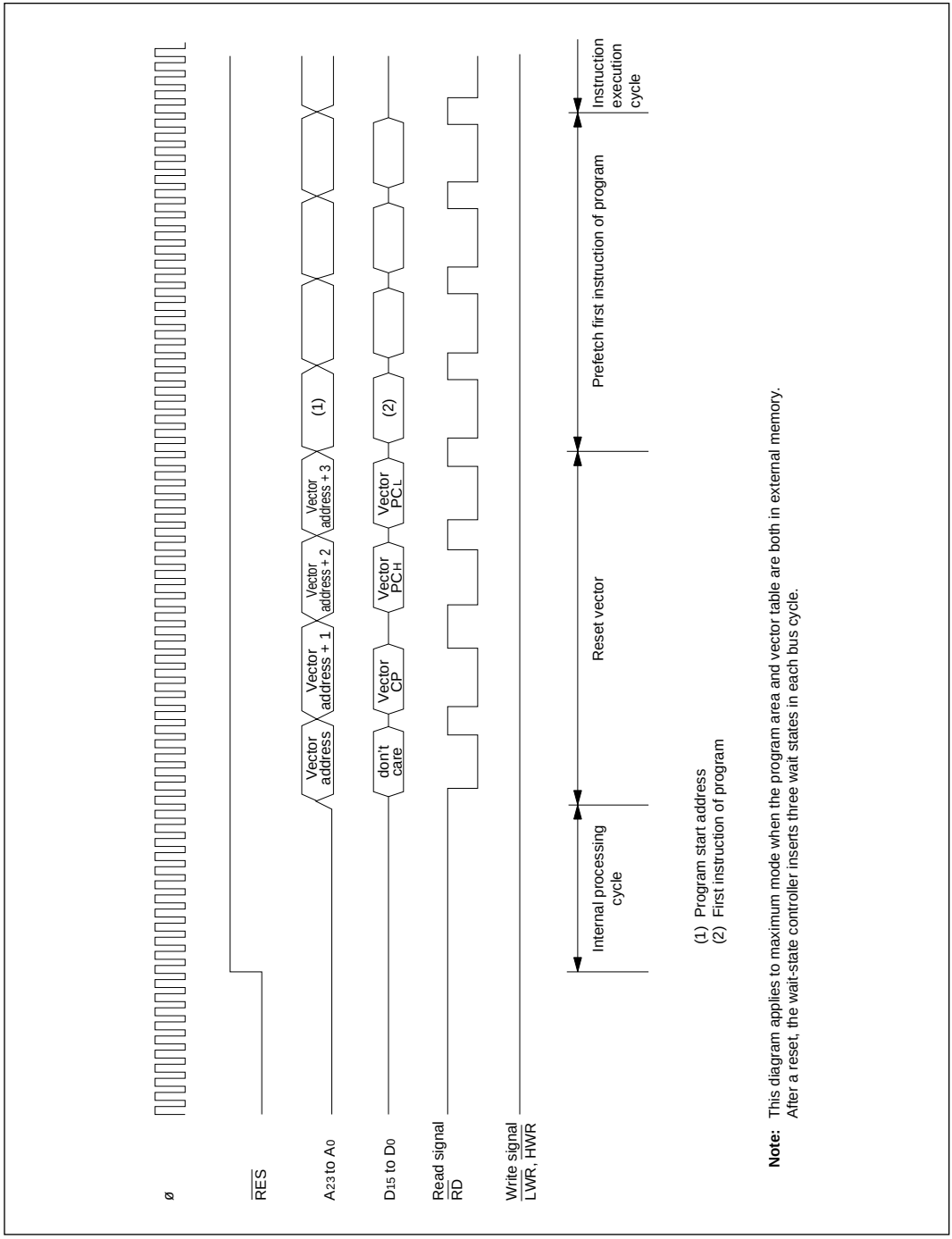
The hardware reset sequence does not initialize the stack pointer, so this must be done by software. If an interrupt were to be accepted after a reset and before the stack pointer (SP) is initialized, the program counter and status register would not be saved correctly, causing a program crash. This danger can be avoided by coding the reset routine as explained next.

When the chip comes out of the reset state all interrupts, including NMI, are disabled, so the instruction at the reset start address is always executed. In the minimum mode, this instruction should initialize the stack pointer (SP). In the maximum mode, this instruction should be an LDC instruction initializing the stack page register (TP), and the next instruction should initialize the stack pointer. Execution of the LDC instruction disables interrupts again, ensuring that the stack pointer initializing instruction is executed.



Note: This timing chart applies to the minimum mode when the program and stack areas are both in on-chip memory and the program starts at an even address.

Figure 4-3 Reset Sequence (Minimum Mode, On-Chip Memory)



Note: This diagram applies to maximum mode when the program area and vector table are both in external memory.
After a reset, the wait-state controller inserts three wait states in each bus cycle.

Figure 4-4 Reset Sequence (Maximum Mode, External Memory)

4.3 Address Error

There are three causes of address errors:
- Illegal instruction prefetch
- Word data access at odd address
- Off-chip access in single-chip mode

An address error initiates the address error exception-handling sequence. This sequence clears the T bit of the status register to 0 to disable the trace mode, but does not affect the interrupt mask level in bits I2 to I0.

4.3.1 Illegal Instruction Prefetch

An attempt to prefetch an instruction from the register field in memory addresses H'FE80 to H'FFFF causes an address error regardless of the MCU operating mode.

Handling of this address error begins when the prefetch cycle that caused the error has been completed and execution of the current instruction has also been completed. The program counter value pushed on the stack is the address of the instruction immediately following the last instruction executed.

Program code should not be located in addresses H'FE7D to H'FE7F. If the CPU executes an instruction in these addresses, it will attempt to prefetch the next instruction from the register field, causing an address error.

4.3.2 Word Data Access at Odd Address

If an attempt is made to access word data starting at an odd address, an address error occurs regardless of the MCU operating mode. The program counter value pushed on the stack in the handling of this error is the address of the next instruction (or next but one) after the instruction that attempted the illegal word access.

4.3.3 Off-Chip Address Access in Single-Chip Mode

In the single-chip mode there is no external memory, so in addition to the address errors described above, the following two types of address errors can occur.

Access to Addresses H'8000 to H'F67F(H8/534): These addresses exist neither in on-chip ROM or RAM nor in the on-chip register field, so an address error occurs if they are accessed for any purpose: for instruction prefetch, byte data access, or word data access.

Program code should not be located in the last three bytes of on-chip ROM (addresses H'7FFD to

H'7FFF). If the CPU excutes an instruction in these addresses, it will attempt to prefetch the next instruction from addresses H'8000 to H'8002, causing an address error.

Access to Disabled RAM Area: The on-chip RAM area (H'F680 to H'FE7F) can be disabled by clearing the RAME bit in the RAM control register (RAMCR). If any form of RAM access is attempted in this state in the single-chip mode, an address error occurs.

4.4 Trace

When the T bit of the status register is set to 1, the CPU operates in trace mode. A trace exception occurs at the completion of each instruction. The trace mode can be used to execute a program for debugging by a debugger.

In the trace exception sequence the T bit of the status register is cleared to 0 to disable the trace mode while the trace routine is executing. The interrupt mask level in bits I2 to I0 is not changed. Interrupts are accepted as usual during the trace routine.

In the status-register data saved on the stack, the T bit is set to 1. When the trace routine returns with the RTE instruction, the status register is popped from the stack and the trace mode resumes.

If an address error occurs during execution of the first instruction after the return from the trace routine, since the address error has higher priority, the address error exception-handling sequence is initiated, clearing the T bit in the status register to 0 and making it impossible to trace this instruction.

4.5 Interrupts

Interrupts can be requested from seven external sources (NMI, IRQ_0, and IRQ_1 to IRQ_5) and eight on-chip supporting modules: the 16-bit free-running timers (FRT1 to FRT3), the 8-bit timer, the serial communication interfaces (SCI1 and SCI2), the A/D converter, and the watchdog timer (WDT). The on-chip interrupt sources can request a total of nineteen different types of interrupts, each having its own interrupt vector. Figure 4-5 lists the interrupt sources and the number of different interrupts from each source.

Each interrupt source has a priority. NMI interrupts have the highest priority, and are normally accepted unconditionally. The priorities of the other interrupt sources are set in control registers (IPR A to D) in the register field at the high end of page 0 and can be changed by software. Priority levels range from 0 (low) to 7 (high), with NMI considered to be on level 8. IRQ_0 and IRQ_1 can be prioritized individually. IRQ_2 and IRQ_3 are prioritized as a pair. IRQ_4 and IRQ_5 are also prioritized as a pair. The on-chip supporting modules are prioritized as modules.

The on-chip interrupt controller decides whether an interrupt can be accepted by comparing its priority with the interrupt mask level, and determines the order in which to accept competing interrupt requests. Interrupts that are not accepted immediately remain pending until they can be accepted later.

When it accepts an interrupt, the interrupt controller also decides whether to interrupt the CPU or start the on-chip data transfer controller (DTC). This decision is controlled by bits set in four data transfer enable registers (DTEA to DTEF) in the register field. The DTC is started if the corresponding bit in DTEA to DTEF is set to 1; otherwise a CPU interrupt is generated. DTC interrupts provide an efficient way to send and receive blocks of data via the serial communication interface, or to transfer data between memory and I/O without detailed CPU programming. The CPU stops while the DTC is operating. DTC interrupts are described in section 6, “Data Transfer Controller.”

The hardware exception-handling sequence for a CPU interrupt clears the T bit in the status register to 0 and sets the interrupt mask level in bits I2 to I0 to the level of the interrupt it has accepted. This prevents the interrupt-handling routine from being interrupted except by a higher-level interrupt. The previous interrupt mask level is restored on the return from the interrupt-handling routine.

For further information on interrupts, see section 5, “Interrupt Controller.”

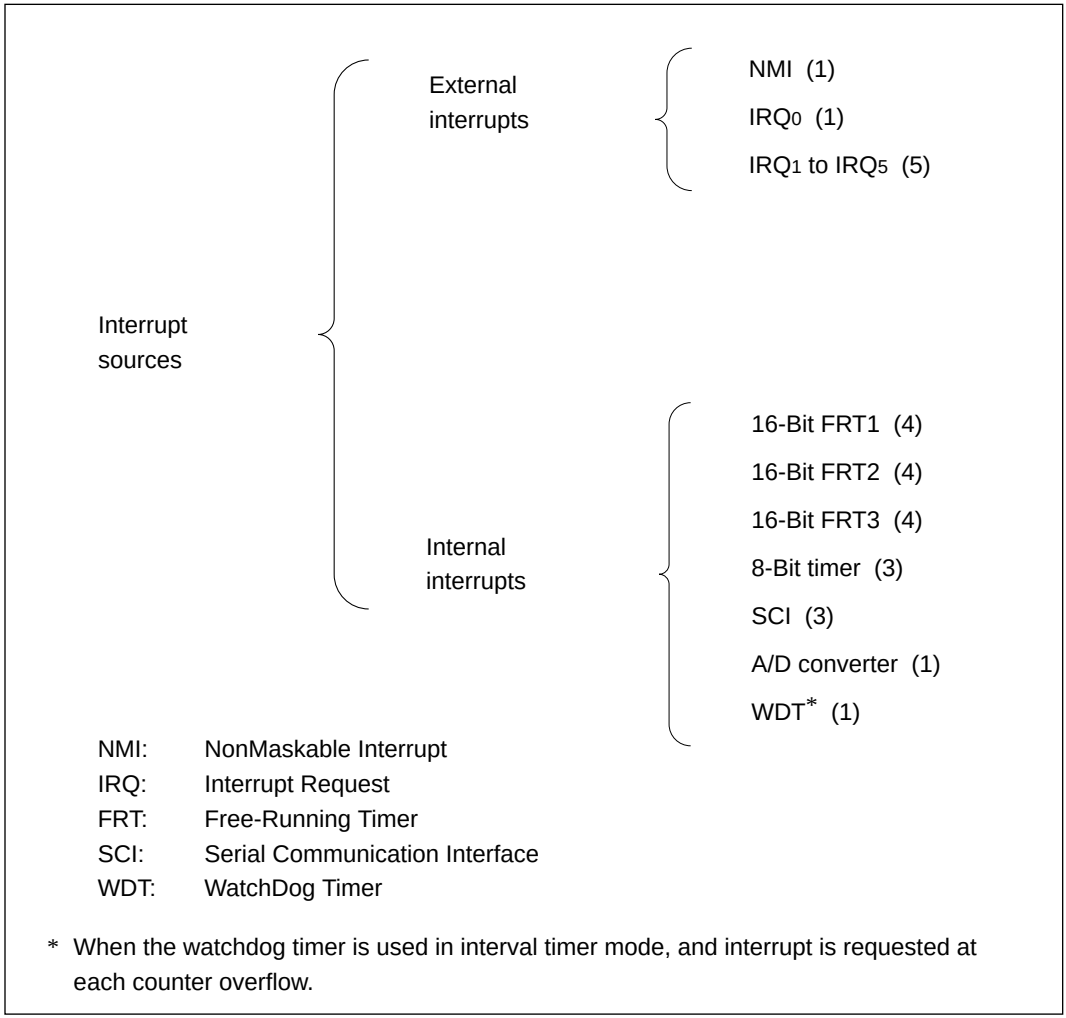


Figure 4-5 Interrupt Sources (and Number of Interrupt Types)

4.6 Invalid Instruction

An invalid instruction exception occurs if an attempt is made to execute an instruction with an undefined operation code or illegal addressing mode specification. The program counter value pushed on the stack is the value of the program counter when the invalid instruction code was detected.

In the invalid instruction exception-handling sequence the T bit of the status register is cleared to 0, but the interrupt mask level (I2 to I0) is not affected.

4.7 Trap Instructions and Zero Divide

A trap exception occurs when the TRAPA or TRAP/VS instruction is executed. A zero divide exception occurs if an attempt is made to execute a DIVXU instruction with a zero divisor.

In the exception-handling sequences for these exceptions the T bit of the status register is cleared to 0, but the interrupt mask level (I2 to I0) is not affected. If a normal interrupt is requested while a trap or zero-divide instruction is being executed, after the trap or zero-divide exception-handling sequence, the normal interrupt exception-handling sequence is carried out.

TRAPA Instruction: The TRAPA instruction always causes a trap exception. The TRAPA instruction includes a vector number from 0 to 15, allowing the user to provide up to sixteen different trap-handling routines.

TRAP/VS Instruction: When the TRAP/VS instruction is executed, a trap exception occurs if the overflow (V) bit in the condition code register is set to 1. If the V bit is cleared to 0, no exception occurs and the next instruction is executed.

DIVXU Instruction with Zero Divisor: An exception occurs if an attempt is made to divide by zero in a DIVXU instruction.

4.8 Cases in Which Exception Handling is Deferred

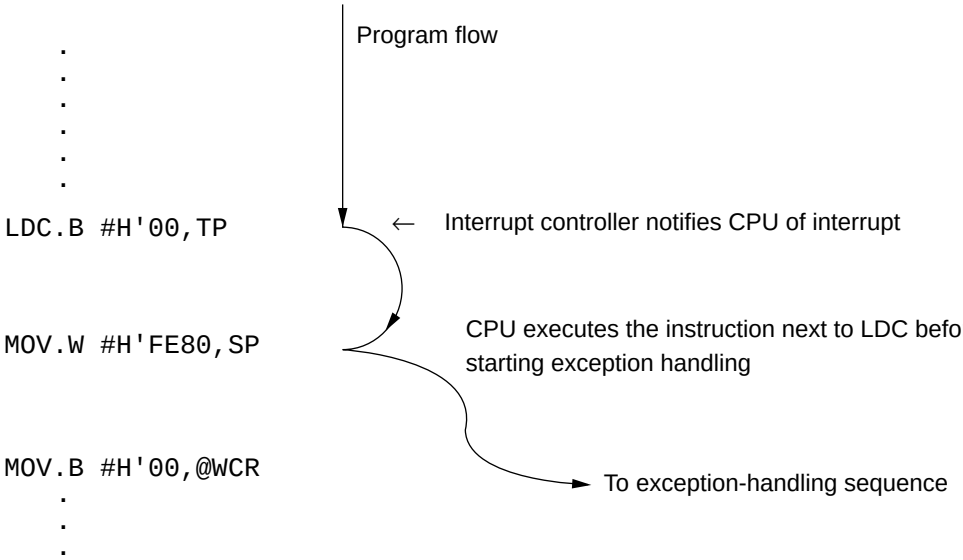
In the cases described next, the address error exception, trace exception, external interrupt (NMI, IRQ_0, and IRQ_1 to IRQ_5) requests, and internal interrupt requests (23 types) are not accepted immediately but are deferred until after the next instruction has been executed.

4.8.1 Instructions that Disable Interrupts

Interrupts are disabled immediately after the execution of five instructions: XORC, ORC, ANDC, LDC, and RTE.

Suppose that an internal interrupt is requested and the interrupt controller, after checking the interrupt priority and interrupt mask level, notifies the CPU of the interrupt, but the CPU is

currently executing one of the five instructions listed above. After executing this instruction the CPU always proceeds to the next instruction. (And if the next instruction is one of these five, the CPU also proceeds to the next instruction after that.) The exception-handling sequence starts after the next instruction that is not one of these five has been executed. The following is an example: (Example)



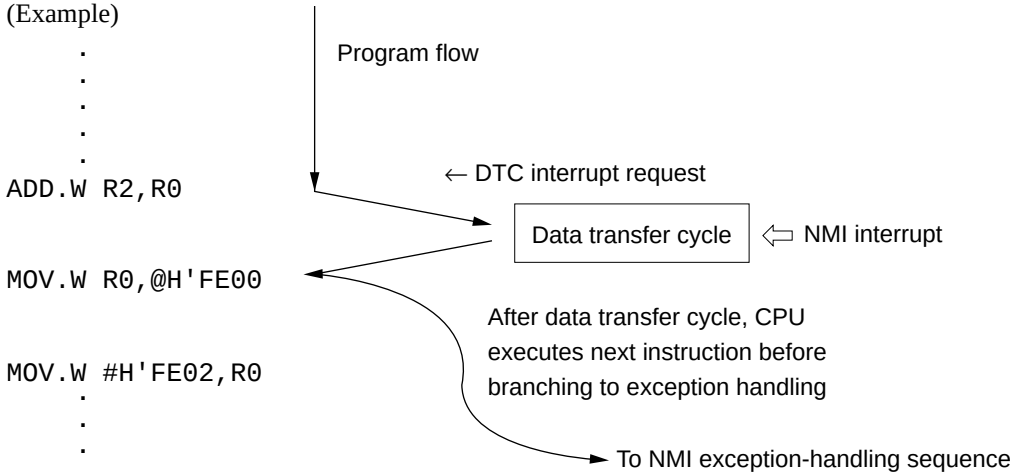
4.8.2 Disabling of Exceptions Immediately after a Reset

If an interrupt is accepted after a reset and before the stack pointer (SP) is initialized, the program counter and status register will not be saved correctly, leading to a program crash. To prevent this, when the chip comes out of the reset state all interrupts, including the NMI, are disabled, so the first instruction of the reset routine is always executed. As noted earlier, in the minimum mode, this instruction should initialize the stack pointer (SP). In the maximum mode, the first instruction should be an LDC instruction that initializes the stack page register (TP); the next instruction should initialize the stack pointer.

4.8.3 Disabling of Interrupts after a Data Transfer Cycle

If an interrupt starts the data transfer controller and another interrupt is requested during the data transfer cycle, when the data transfer cycle ends, the CPU always executes the next instruction before handling the second interrupt.

Even if a nonmaskable interrupt (NMI) occurs during a data transfer cycle, it is not accepted until the next instruction has been executed. An example of this is shown below.



4.9 Stack Status after Completion of Exception Handling

The status of the stack after an exception-handling sequence is described below.

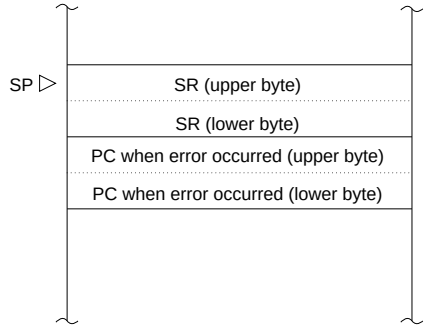
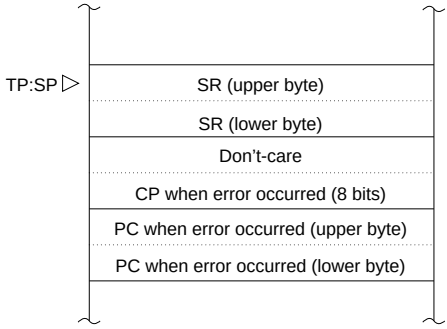
Table 4-3 shows the stack after completion of the exception-handling sequence for various types of exceptions in the minimum and maximum modes.

Table 4-3 Stack after Exception Handling Sequence

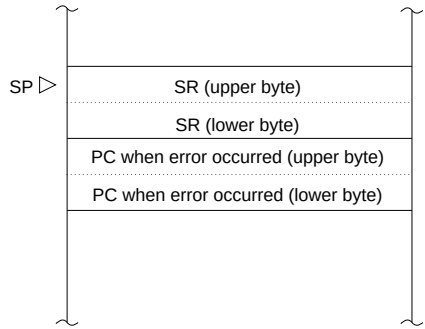
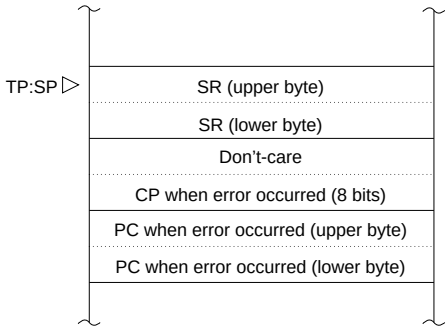
Exception Factor	Minimum Mode	Maximum Mode
Trace Interrupt Trap Zero divide (DIVXU)	SP ▷ SR (upper byte) SR (lower byte) Next instruction address (upper byte) Next instruction address (lower byte)	TP:SP ▷ SR (upper byte) SR (lower byte) Don't-care Next instruction page (8 bits) Next instruction address (upper byte) Next instruction address (lower byte)

Note: The RTE instruction returns to the next instruction after the instruction being executed when the exception occurred.

Table 4-3 Stack after Exception Handling Sequence (cont)

Exception Factor	Minimum Mode	Maximum Mode
Invalid instruction		

Note: The program counter value pushed on the stack is not necessarily the address of the first byte of the invalid instruction.

Address error		

Note: The program counter value pushed on the stack is the address of the next instruction after the last instruction successfully executed.

4.9.1 PC Value Pushed on Stack for Trace, Interrupts, Trap Instructions, and Zero Divide Exceptions

The program counter value pushed on the stack for a trace, interrupt, trap, or zero divide exception is the address of the next instruction at the time when the interrupt was accepted. The RTE instruction accordingly returns to the next instruction after the instruction executed before the exception-handling sequence.

4.9.2 PC Value Pushed on Stack for Address Error and Invalid Instruction Exceptions

The program counter value pushed on the stack for an address error or invalid instruction exception differs depending on the conditions when the exception occurred.

4.10 Notes on Use of the Stack

If the stack pointer is set to an odd address, an address error will occur when the stack is accessed during interrupt handling or for a subroutine call. The stack pointer should always point to an even address. To keep the stack pointer pointing to an even address, a program should use word data size when saving or restoring registers to and from the stack.

In the @–SP or @SP+ addressing mode, the CPU performs word access even if the instruction specifies byte size. (This is not true in the @–Rn and @Rn+ addressing modes when Rn is a register from R0 to R6.)

Section 5 Interrupt Controller

5.1 Overview

The interrupt controller decides which interrupts to accept, and how to deal with multiple interrupts. It also decides whether an interrupt should be served by the CPU or by the data transfer controller (DTC). This section explains the features of the interrupt controller, describes its internal structure and control registers, and details the handling of interrupts.

For detailed information on the data transfer controller, see section 6, “Data Transfer Controller.”

5.1.1 Features

Three main features of the interrupt controller are:

- Interrupt priorities are user-programmable.
  User programs can set priority levels from 7 (high) to 0 (low) in six interrupt priority (IPR) registers for IRQ_0, IRQ_1 to IRQ_5, and each of the on-chip supporting modules—for every interrupt, that is, except the nonmaskable interrupt (NMI). NMI has the highest priority level (8) and is normally always accepted. An interrupt with priority level 0 is always masked.
- Multiple interrupts on the same level are served in a default priority order.
  Lower-priority interrupts remain pending until higher-priority interrupts have been handled.
- For most interrupts, software can select whether to have the interrupt served by the CPU or the on-chip data transfer controller (DTC).
  User programs can make this selection by setting and clearing bits in four data transfer enable (DTE) registers. The data transfer controller can be started by any interrupts except NMI, the error interrupt (ERI) from the on-chip serial communication interface, and the overflow interrupts (FOVI and OVI) from the on-chip timers.

5.1.2 Block Diagram

Figure 5-1 shows the block configuration of the interrupt controller.

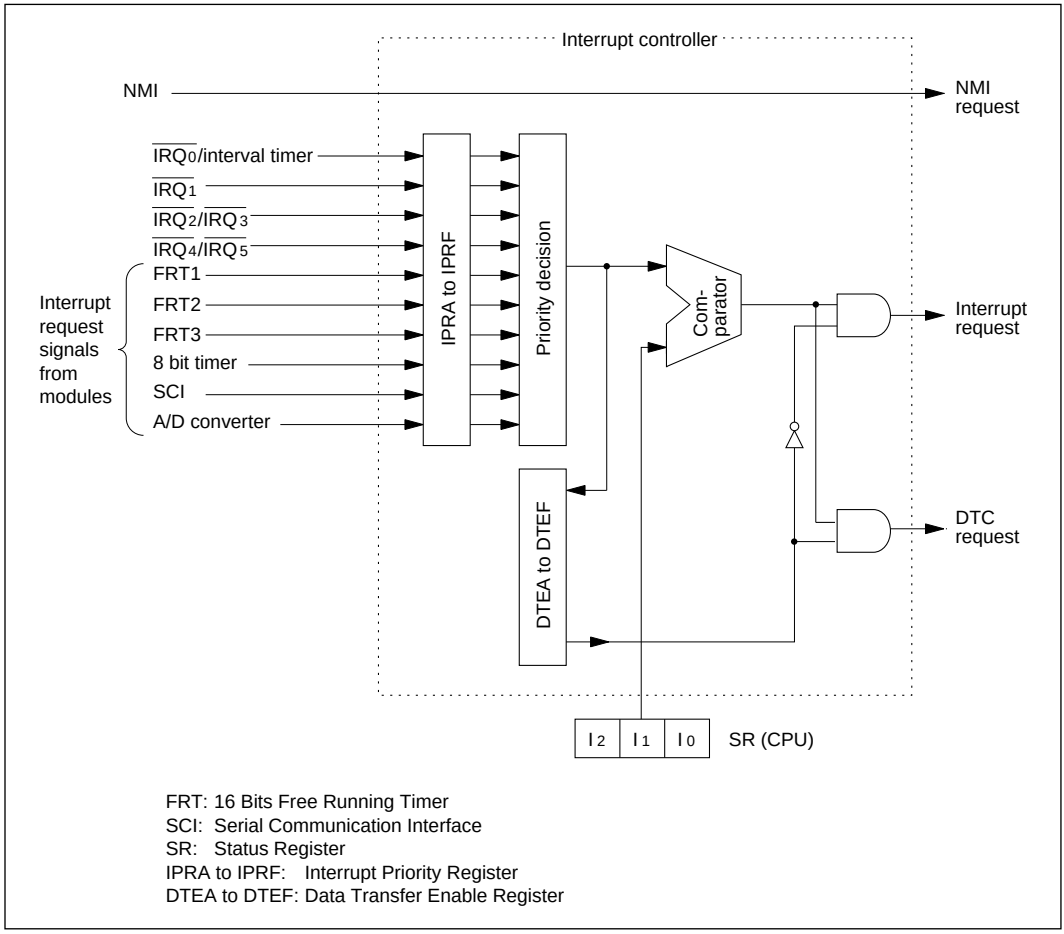


Figure 5-1 Interrupt Controller Block Diagram

5.1.3 Register Configuration

The six interrupt priority registers (IPRA to IPRF) and six data transfer enable registers (DTEA to DTEF) are 8-bit registers located at addresses H'FF00 to H'FF0D in the register field in page 0 of the address space. Table 5-1 lists their attributes.

Table 5-1 Interrupt Controller Registers

Name		Abbreviation	Read/Write	Address	Initial Value
Interrupt	A	IPRA	R/W	H'FF00	H'00
priority	B	IPRB	R/W	H'FF01	H'00
register	C	IPRC	R/W	H'FF02	H'00
	D	IPRD	R/W	H'FF03	H'00
	E	IPRE	R/W	H'FF04	H'00
	F	IPRF	R/W	H'FF05	H'00
Data transfer	A	DTEA	R/W	H'FF08	H'00
enable	B	DTEB	R/W	H'FF09	H'00
register	C	DTEC	R/W	H'FF0A	H'00
	D	DTED	R/W	H'FF0B	H'00
	E	DTEE	R/W	H'FF0C	H'00
	F	DTEF	R/W	H'FF0D	H'00

See section 6.2.5, “Data Transfer Enable Registers A to F” for further information about DTEA to DTEF.

5.2 Interrupt Types

There are 30 distinct types of interrupts: 7 external interrupts originating off-chip and 23 internal interrupts originating in the on-chip supporting modules.

5.2.1 External Interrupts

The seven external interrupts are NMI, IRQ_0, and IRQ_1 to IRQ_5.

NMI (NonMaskable Interrupt): This interrupt has the highest priority level (8) and cannot be masked. An NMI is generated by input to the NMI pin, and can also be generated by a watchdog timer (WDT) overflow. The input at the NMI pin is edge-sensed. A user program can select whether to have the interrupt occur on the rising edge or falling edge of the NMI input by setting or clearing the nonmaskable interrupt edge bit (NMIEG) in system control register 1 (SYSCR1).

In the NMI exception-handling sequence, the T (Trace) bit in the CPU status register (SR) is cleared to "0," and the interrupt mask level in I2 to I0 is set to 7, masking all other interrupts. The interrupt controller holds the NMI request until the NMI exception-handling sequence begins,

then clears the NMI request, so if another interrupt is requested at the NMI pin during the NMI exception-handling sequence, the NMI exception-handling sequence will be carried out again.

Coding Examples:

To select the rising edge of the NMI input: `BSET.B #4, @H'FEFC`
To select the falling edge of the NMI input: `BCLR.B #4, @H'FEFC`

IRQ0 (Interrupt Request 0): An IRQ_0 interrupt can be requested by a Low input to the $\overline{IRQ_0}$ pin. A Low IRQ_0 input requests an IRQ_0 interrupt if the interrupt request enable 0 bit (IRQ_0E) in SYSCR1 is set to 1. $\overline{IRQ_0}$ must be held Low until the CPU accepts the interrupt. Otherwise the request will be ignored.

The IRQ_0 interrupt can be assigned any priority level from 7 to 0 by setting the corresponding value in the upper four bits of IPRA. If bit 4 of data transfer enable register A (DTEA) is set to 1, an IRQ_0 interrupt starts the data transfer controller. Otherwise the interrupt is served by the CPU.

In the CPU interrupt-handling sequence for IRQ_0, the T bit of the status register is cleared to 0, and the interrupt mask level is set to the value in the upper four bits of IPRA.

Coding Examples:

To enable IRQ_0 to be requested by $\overline{IRQ_0}$ input: `BSET.B #5, @H'FEFC`
To assign priority level 7 to IRQ_0: `OR.B  #70, @H'FF00`
To have IRQ_0 start the DTC: `BSET.B #4, @H'FF08`

IRQ_1 to IRQ_5 (Interrupt Request 1 to 5): An IRQ_1 to IRQ_5 interrupt is requested by a High-to-Low transition at the $\overline{IRQ_1}$ to $\overline{IRQ_5}$ pin. The IRQ_1 interrupt is enabled only when the interrupt request enable 1 bit (IRQ_1E) in SYSCR1 is set to 1. IRQ_2 to IRQ_5 are controlled by bits IRQ_2E to IRQ_5E in SYSCR2. (see section 9.7, “Port 6.”)

Interrupts IRQ_1 to IRQ_5 can be assigned any priority level from 7 (high) to 0 (low) by setting the corresponding value in IPRA and IPRB. The lower four bits of IPRA determine the priority of IRQ_1. The upper four bits of IPRB determine the priority of IRQ_2 and IRQ_3. The lower four bits of IPRB determine the priority of IRQ_4 and IRQ_5. Interrupt requests IRQ_1 to IRQ_5 are held in the interrupt controller and cleared during the corresponding interrupt exception-handling sequence. Contention among IRQ_1 to IRQ_5 is resolved when the CPU accepts the interrupt by taking the interrupt with the highest priority first and holding lower-priority interrupts pending. (Contention between IRQ_2 and IRQ_3, or between IRQ_4 and IRQ_5, is resolved by the priority order shown in table 5-2.)

During the interrupt-handling routine, if the same external interrupt is requested again the request is held, but the exception-handling sequence is not carried out immediately because the interrupt is masked by bits I2 to I0 in the status register. On return from the interrupt-handling routine one more instruction is executed, then the pending exception-handling sequence is carried out.

Interrupts IRQ_1 to IRQ_5 are served by the CPU or DTC depending on DTEA bit 0 and DTEB bits 0, 1, 4, and 5.

In the CPU interrupt exception-handling sequence for IRQ_1 to IRQ_5, the T bit of the CPU status register is cleared to 0, and the interrupt mask level is set to the value in IPRA or IPRB.

Coding Examples:

To enable IRQ_1 to be requested by IRQ_1 input:	`BSET.B #6, @H'FEFC`
To assign priority level 7 to IRQ_0 and level 5 to IRQ_1:	`MOV.B #75, @H'FF00`
To have IRQ_1 start the DTC:	`BSET.B #0, @H'FF08`

5.2.2 Internal Interrupts

Twenty-three types of internal interrupts can be requested by the on-chip supporting modules. Each interrupt is separately vectored in the exception vector table, so it is not necessary for the user-coded interrupt handler routine to determine which type of interrupt has occurred.

Each of the internal interrupts can be enabled or disabled by setting or clearing an enable bit in the control register of the on-chip supporting module.

An interrupt priority level from 7 to 0 can be assigned to each on-chip supporting module by setting interrupt priority registers C to F. Within each module, different interrupts have a fixed priority order. For most of these interrupts, values set in data transfer enable registers C to F can select whether to have the interrupt served by the CPU or the data transfer controller.

In the CPU interrupt-handling sequence, the T bit of the CPU status register is cleared to 0, and the interrupt mask level in bits I2 to I0 is set to the value in the IPR. Unlike external interrupt requests, internal interrupt requests are not held in the interrupt controller, so the bits that generate internal interrupts must be cleared by software.

5.2.3 Interrupt Vector Table

Table 5-2 lists the addresses of the exception vector table entries for each interrupt, and explains how their priority is determined. For the on-chip supporting modules, the priority level set in the interrupt priority register applies to the module as a whole: all interrupts from that module have the same priority level. A separate priority order is established among interrupts from the same module. If the same priority level is assigned to two or more modules and two interrupts are requested simultaneously from these modules, they are served in the priority order indicated in the rightmost column in table 5-2.

A reset clears the interrupt priority registers so that all interrupts except NMI start with priority level 0, meaning that they are unconditionally masked.

Table 5-2 Interrupts, Vectors, and Priorities

Interrupt		Assignable Priority Levels (Initial Level)	IPR Bits	Priority within Module	Vector Table Entry Address		Priority among Interrupts on Same Level*
					Minimum Mode	Maximum Mode	
NMI		8(8)	—	—	H'16 - H'17	H'2C - H'2F	High
IRQ_0		7 to 0	IPRA	1	H'40 - H'41	H'80 - H'83	
Interval timer		(0)	bits 6 to 4	0	H'42 - H'43	H'84 - H'87	
IRQ_1		7 to 0	IPRA	—	H'48 - H'49	H'90 - H'93	
		(0)	bits 2 to 0				
IRQ_2		7 to 0	IPRB	1	H'50 - H'51	H'A0 - H'A3	
IRQ_3		(0)	bits 6 to 4	0	H'52 - H'53	H'A4 - H'A7	
IRQ_4		7 to 0	IPRB	1	H'58 - H'59	H'B0 - H'B3	
IRQ_5		(0)	bits 2 to 0	0	H'5A - H'5B	H'B4 - H'B7	
FRT1	ICI	7 to 0	IPRC	3	H'60 - H'61	H'C0 - H'C3	
	OCIA	(0)	bits 6 to 4	2	H'62 - H'63	H'C4 - H'C7	
	OCIB			1	H'64 - H'65	H'C8 - H'CB	
	FOVI			0	H'66 - H'67	H'CC - H'CF	
FRT2	ICI	7 to 0	IPRC	3	H'68 - H'69	H'D0 - H'D3	
	OCIA	(0)	bits 2 to 0	2	H'6A - H'6B	H'D4 - H'D7	
	OCIB			1	H'6C - H'6D	H'D8 - H'DB	
	FOVI			0	H'6E - H'6F	H'DC - H'DF	
FRT3	ICI	7 to 0	IPRD	3	H'70 - H'71	H'E0 - H'E3	
	OCIA	(0)	bits 6 to 4	2	H'72 - H'73	H'E4 - H'E7	
	OCIB			1	H'74 - H'75	H'E8 - H'EB	
	FOVI			0	H'76 - H'77	H'EC - H'EF	
8-bit	CMIA	7 to 0	IPRD	2	H'78 - H'79	H'F0 - H'F3	
timer	CMIB	(0)	bits 2 to 0	1	H'7A - H'7B	H'F4 - H'F7	
	OVI			0	H'7C - H'7D	H'F8 - H'FB	
SCI1	ERI	7 to 0	IPRE	2	H'80 - H'81	H'100 - H'103	
	RXI	(0)	bits 6 to 4	1	H'82 - H'83	H'104 - H'107	
	TXI			0	H'84 - H'85	H'108 - H'10B	
SCI2	ERI	7 to 0	IPRE	2	H'88 - H'89	H'110 - H'113	
	RXI	(0)	bits 2 to 0	1	H'8A - H'8B	H'114 - H'117	
	TXI			0	H'8C - H'8D	H'118 - H'11B	
A/D	ADI	7 to 0	IPRF	—	H'90 - H'91	H'120 - H'123	
converter		(0)	bits 6 to 4				Low

* If two or more interrupts are requested simultaneously, they are handled in order of priority level, as set in registers IPRA to IPRF. If they have the same priority level because they are requested from the same on-chip supporting module, they are handled in a fixed priority order within the module. If they are requested from different modules to which the same priority level is assigned, they are handled in the order indicated in the right-hand column.

5.3 Register Descriptions

5.3.1 Interrupt Priority Registers A to F (IPRA to IPRF)

IRQ_0, IRQ_1 to IRQ_5, and the on-chip supporting modules are each assigned three bits in one of the six interrupt priority registers (IPRA to IPRF). These bits specify a priority level from 7 (high) to 0 (low) for interrupts from the corresponding source. The drawing below shows the configuration of the interrupt priority registers. Table 5-3 lists their assignments to interrupt sources.

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

Note: Bits 7 and 3 are reserved. They cannot be modified and are always read as 0.

Table 5-3 Assignment of Interrupt Priority Registers

	Interrupt Request Source	
Register	**Bits 6 to 4**	**Bits 2 to 0**
IPRA	$\overline{IRQ_0}$	$\overline{IRQ_1}$
IPRB	$\overline{IRQ_2}$, $\overline{IRQ_3}$	$\overline{IRQ_4}$, $\overline{IRQ_5}$
IPRC	FRT1	FRT2
IPRD	FRT3	8-bit timer
IPRE	SCI1	SCI2
IPRF	A/D converter	—

As table 5-3 indicates, each interrupt priority register specifies priority levels for two interrupt sources. A user program can assign desired levels to these interrupt sources by writing “000” in bits 6 to 4 or bits 2 to 0 to set priority level 0, for example, or “111” to set priority level 7.

A reset clears registers IPRA to IPRF to H'00, so all interrupts except NMI are initially masked.

When the interrupt controller receives one or more interrupt requests, it selects the request with the highest priority and compares its priority level with the interrupt mask level set in bits I2 to I0 in the CPU status register. If the priority level is higher than the mask level, the interrupt controller passes the interrupt request to the CPU (or starts the data transfer controller). If the priority level is lower than the mask level, the interrupt controller leaves the interrupt request pending until the interrupt mask is altered to a lower level or the interrupt priority is raised. Similarly, if it receives two interrupt requests with the same priority level, the interrupt controller determines their priority as explained in table 5-2 and leaves the interrupt request with the lower priority pending.

5.3.2 Timing of Priority Setting

The interrupt controller requires two system clock (ø) periods to determine the priority level of an interrupt. Accordingly, when an instruction modifies an instruction priority register, the new priority does not take effect until after the next instruction has been executed.

5.4 Interrupt Handling Sequence

5.4.1 Interrupt Handling Flow

The interrupt-handling sequence follows the flowchart in figure 5-2. Note that address error, trace exception, and NMI requests bypass the interrupt controller's priority decision logic and are routed directly to the CPU.

1. Interrupt requests are generated by one or more on-chip supporting modules or external interrupt sources.
2. The interrupt controller checks the interrupt priorities set in IPRA to IPRF and selects the interrupt with the highest priority. Interrupts with lower priorities remain pending. Among interrupts with the same priority level, the interrupt controller determines priority as explained in table 5-2.
3. The interrupt controller compares the priority level of the selected interrupt request with the mask level in the CPU status register (bits I2 to I0). If the priority level is equal to or less than the mask level, the interrupt request remains pending. If the priority level is higher than the mask level, the interrupt controller accepts the interrupt request and proceeds to the next step.
4. The interrupt controller checks the corresponding bit (if any) in the data transfer enable registers (DTEA to DTEF). If this bit is set to 1, the data transfer controller is started. Otherwise, the CPU interrupt exception-handling sequence is started.

   When the data transfer controller is started, the interrupt request is cleared (except for interrupt requests from the serial communication interface, which are cleared by writing to the TDR or reading the RDR).

If the data transfer enable bit is cleared to 0 (or is nonexistent), the sequence proceeds as follows. For the case in which the data transfer controller is started, see section 6, “Data Transfer Controller.”

5. After the CPU has finished executing the current instruction, the program counter and status register (in minimum mode) or program counter, code page register, and status register (in maximum mode) are saved to the stack, leaving the stack in the condition shown in figure 5-3 (a) or (b). The program counter value saved on the stack is the address of the next instruction to be executed.
6. The T (Trace) bit of the status register is cleared to 0, and the priority level of the interrupt is copied to bits I2 to I0, thus masking further interrupts unless they have a higher priority level. When an NMI is accepted, the interrupt mask level in bits I2 to I0 is set to 7.
7. The interrupt controller generates the vector address of the interrupt, and the entry at this address in the exception vector table is read to obtain the starting address of the user-coded interrupt handling routine.

In step 7, the same difference between the minimum and maximum modes exists as in the reset handling sequence. In the minimum mode, one word is copied from the vector table to the program counter, then the interrupt-handling routine starts executing from the address indicated in the program counter. In the maximum mode, two words are read. The lower byte of the first word is copied to the code page register. The second word is copied to the program counter. The interrupt-handling routine starts executing from the address indicated in the code page register and program counter.

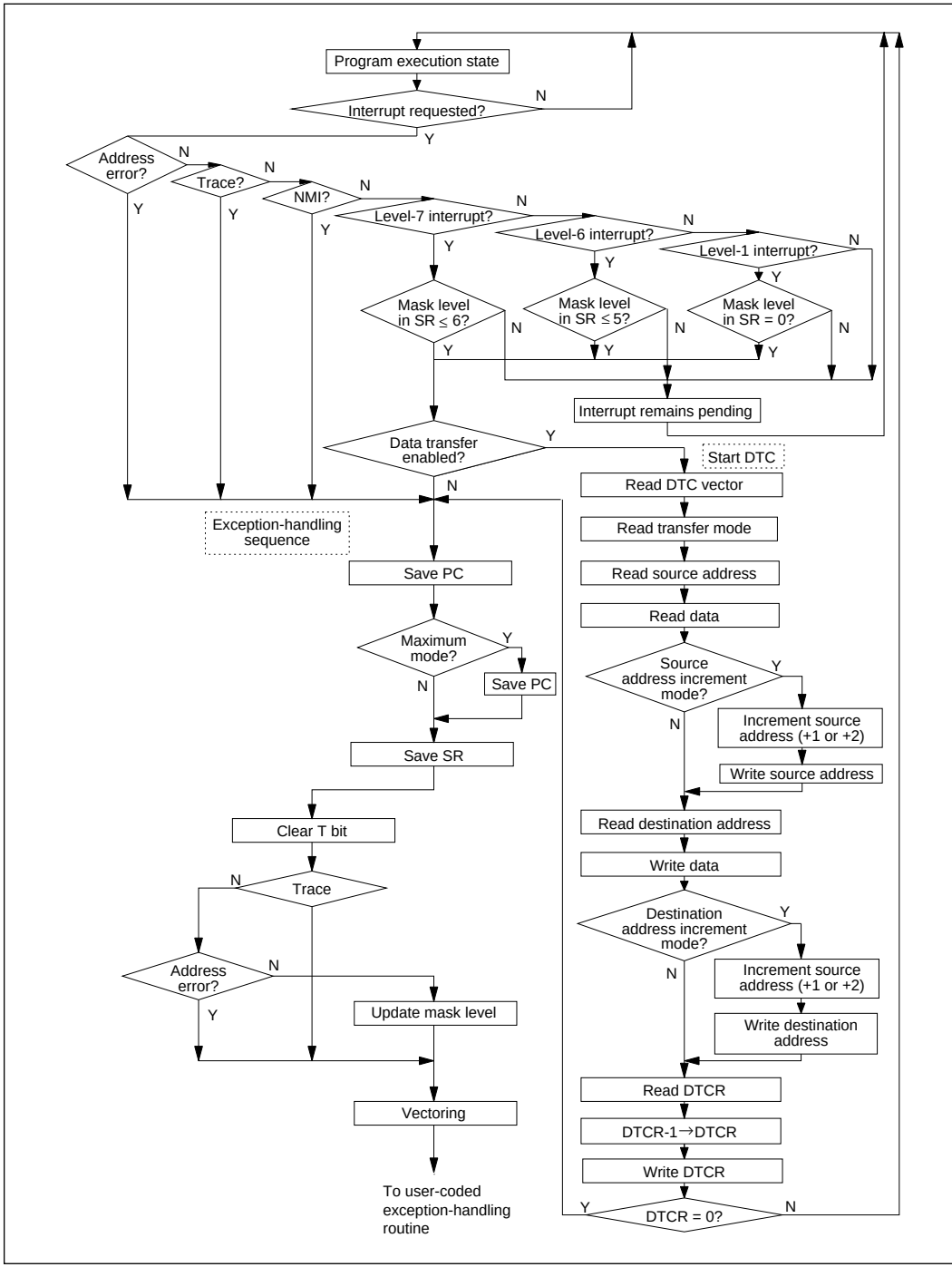
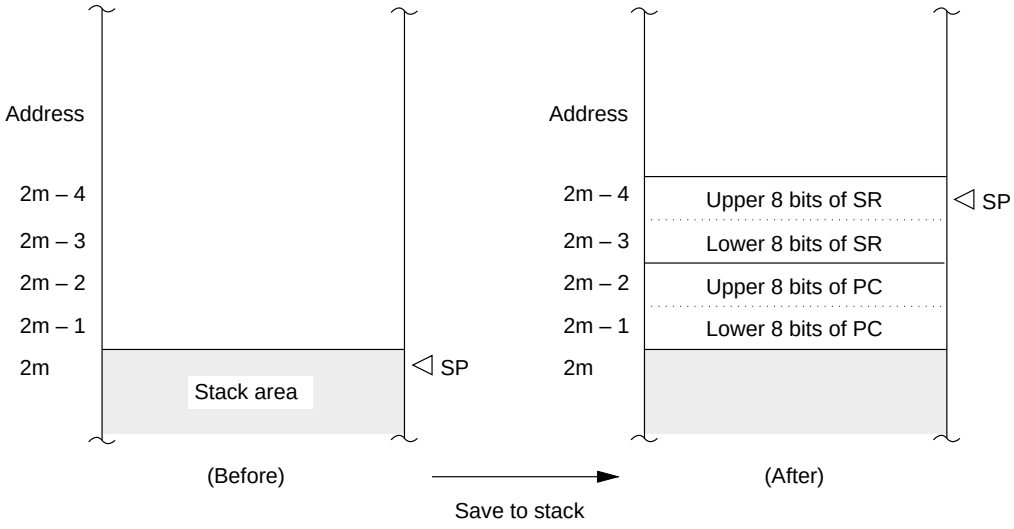


Figure 5-2 Interrupt Handling Flowchart

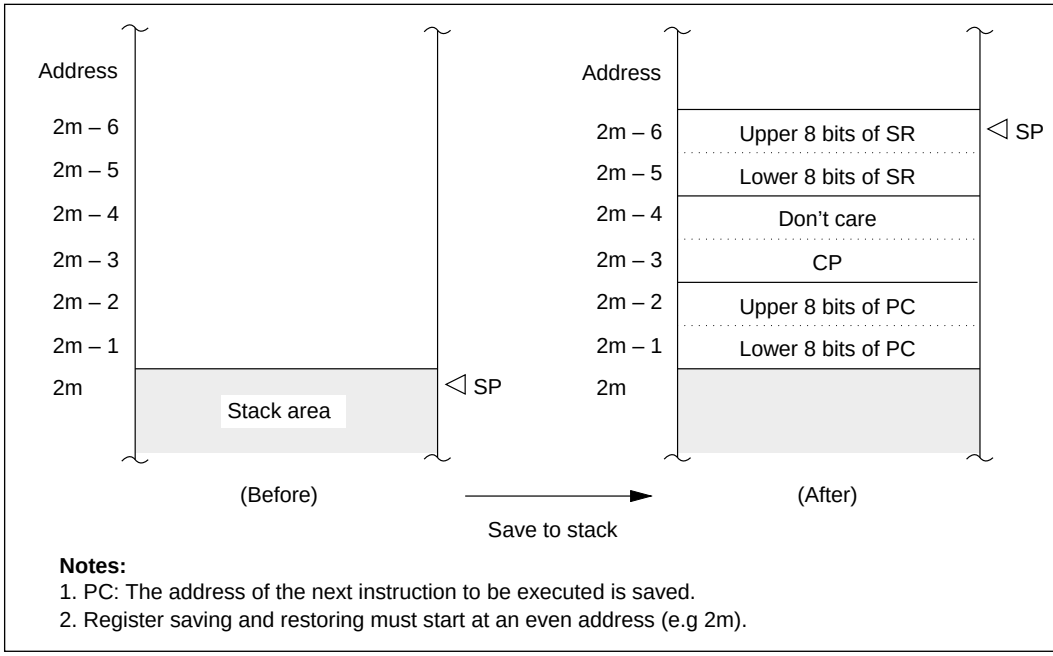
5.4.2 Stack Status after Interrupt Handling Sequence

Figure 5-3 (a) and (b) show the stack before and after the interrupt exception-handling sequence.



Notes:
1. PC: The address of the next instruction to be executed is saved.
2. Register saving and restoring must start at an even address (e.g 2m).

Figure 5-3 (a) Stack before and after Interrupt Exception-Handling (Minimum Mode)



Notes:
1. PC: The address of the next instruction to be executed is saved.
2. Register saving and restoring must start at an even address (e.g 2m).

Figure 5-3 (b) Stack before and after Interrupt Exception-Handling (Maximum Mode)

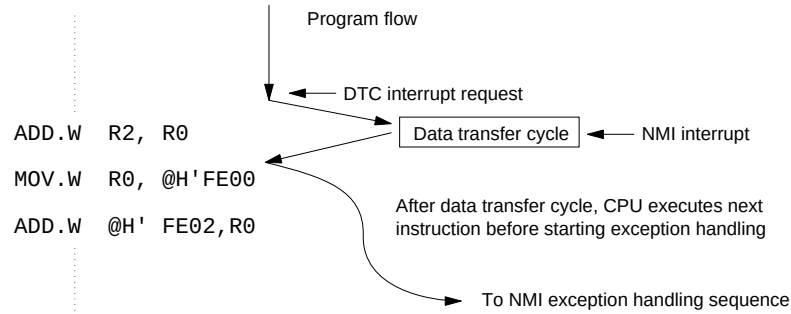
5.4.3 Timing of Interrupt Exception-Handling Sequence

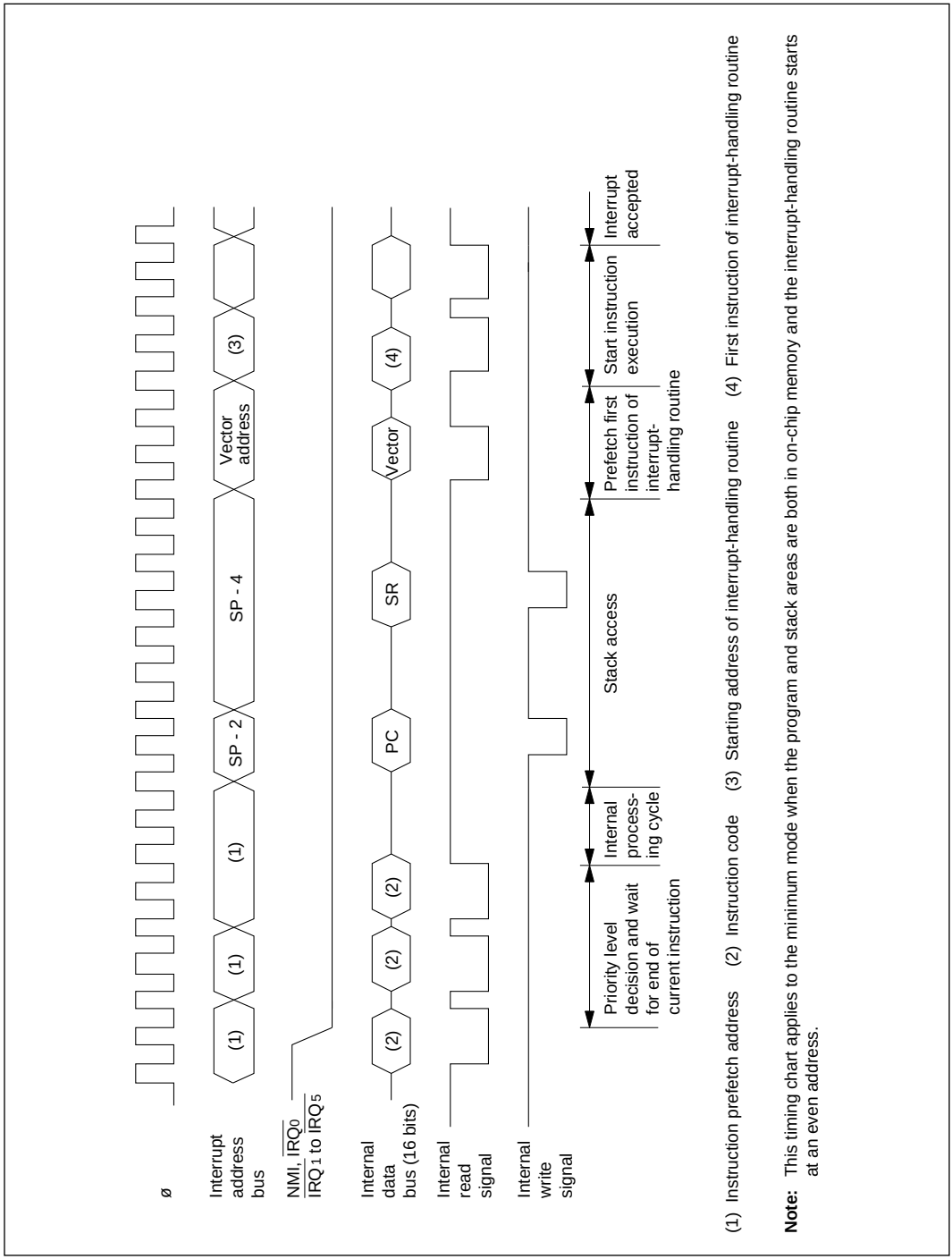
Figure 5-4 shows the timing of the exception-handling sequence for an interrupt in minimum mode when the program area and stack area are both in on-chip memory and the user-coded interrupt handling routine starts at an even address.
Figure 5-5 shows the timing of the exception-handling sequence for an interrupt in maximum mode when the program area and stack area are both in external memory.

5.5 Interrupts During Operation of the Data Transfer Controller

If an interrupt is requested during a DTC data transfer cycle, the interrupt is not accepted until the data transfer cycle has been completed and the next instruction has been executed. This is true even if the interrupt is an NMI. An example is shown below.

(Example)

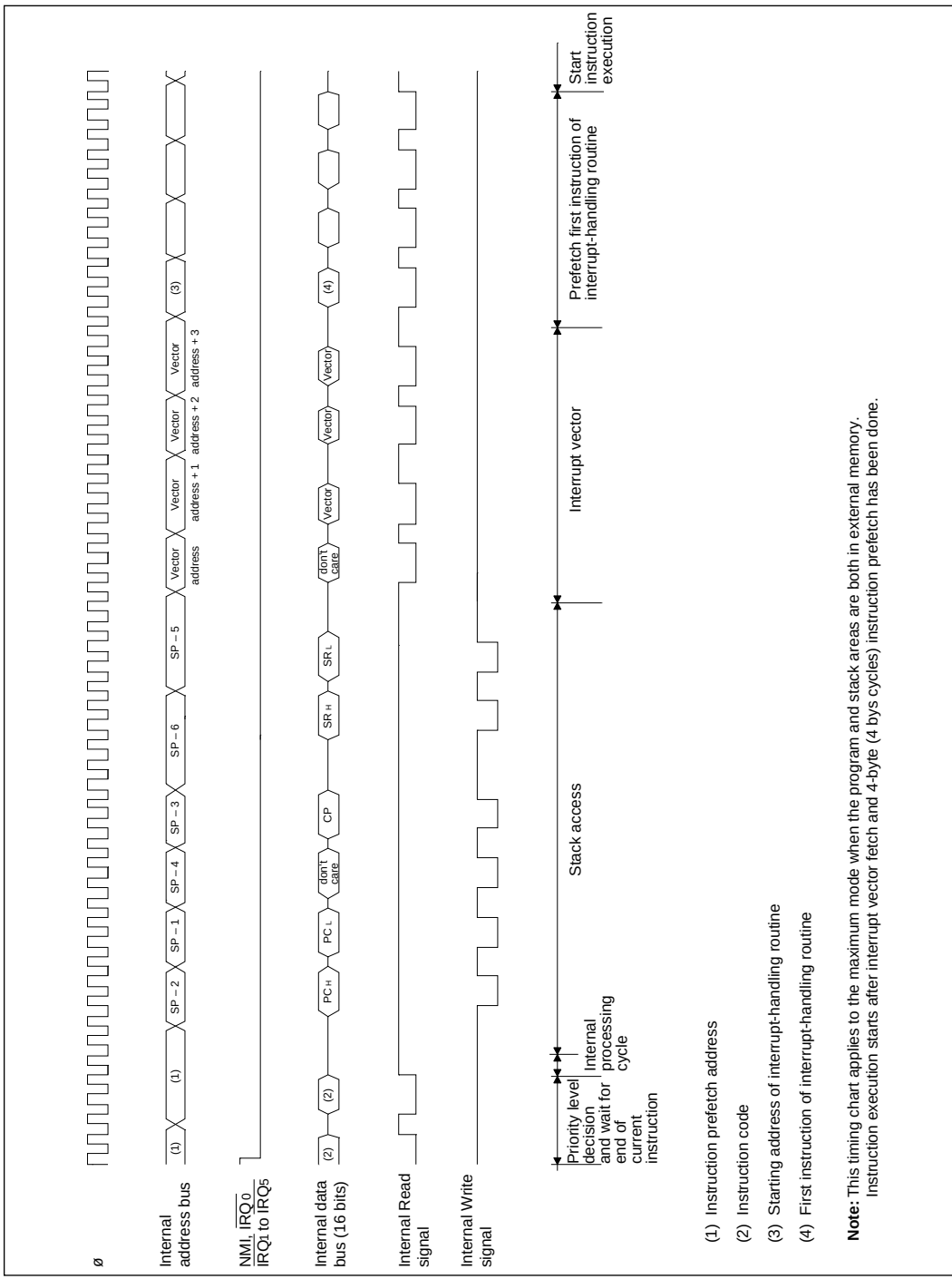




(1) Instruction prefetch address (2) Instruction code (3) Starting address of interrupt-handling routine (4) First instruction of interrupt-handling routine

Note: This timing chart applies to the minimum mode when the program and stack areas are both in on-chip memory and the interrupt-handling routine starts at an even address.

Figure 5-4 Interrupt Sequence (Minimum Mode, On-Chip Memory)



(1) Instruction prefetch address

(2) Instruction code

(3) Starting address of interrupt-handling routine

(4) First instruction of interrupt-handling routine

Note: This timing chart applies to the maximum mode when the program and stack areas are both in external memory.
Instruction execution starts after interrupt vector fetch and 4-byte (4 bys cycles) instruction prefetch has been done.

Figure 5-5 Interrupt Sequence (Maximum Mode, External Memory)

5.6 Interrupt Response Time

Table 5-4 indicates the number of states that may elapse between the generation of an interrupt request and the execution of the first instruction of the interrupt-handling routine, assuming that the interrupt is not masked and not preempted by a higher-priority interrupt. Since word access is performed to on-chip memory areas, fastest interrupt service can be obtained by placing the program in on-chip ROM and the stack in on-chip RAM.

Table 5-4 Number of States before Interrupt Service

			Number of States	
No.	**Reason for Wait**		**Minimum Mode**	**Maximum Mode**
1	Interrupt priority decision and comparison with mask level in CPU status register		2 states	
2	Maximum number of states to completion of current instruction	Instruction is in on-chip memory	x (x = 38 for LDM instruction specifying all registers)	
		Instruction is in external memory	y (y = 74 + 16m for LDM instruction specifying all registers)	
3	Saving of PC and SR or PC, CP, and SR and instruction prefetch	Stack is in on-chip RAM	16	21
		Stack is in external memory	28 + 6m	41 + 10m
Total	Stack is in on-chip RAM	Instruction is in on-chip memory	18 + x (56)	23 + x (61)
		Instruction is in external memory	18 + y (92 + 16m)	23 + y (97 + 16m)
	Stack is in external RAM	Instruction is in on-chip memory	30 + 6m + x (68 + 6m)	43 + 10m + x (81 + 10m)
		Instruction is in external memory	30 + 6m + y (104 + 22m)	43 + 10m + y (117 + 26m)

Note: m: Number of wait states inserted in external memory access.
Values in parentheses are for the LDM instruction.

Section 6 Data Transfer Controller

6.1 Overview

The H8/534 and H8/536 include a data transfer controller (DTC) that can be started by designated interrupts to transfer data from a source address to a destination address located in page 0. These addresses include in particular the registers of the on-chip supporting modules and I/O ports. Typical uses of the DTC are to change the setting of a control register of an on-chip supporting module in response to an interrupt from that module, or to transfer data from memory to an I/O port or the serial communication interface. Once set up, the transfer is interrupt-driven, so it proceeds independently of program execution, although program execution temporarily stops while each byte or word is being transferred.

6.1.1 Features

The main features of the DTC are listed below.

- The source address and destination address can be set anywhere in the 64-kbyte address space of page 0.
- The DTC can be programmed to transfer one byte or one word of data per interrupt.
- The DTC can be programmed to increment the source address and/or destination address after each byte or word is transferred.
- After transferring a designated number of bytes or words, the DTC generates a CPU interrupt with the vector of the interrupt source that started the DTC.
- This designated data transfer count can be set from 1 to 65,536 bytes or words.

6.1.2 Block Diagram

Figure 6-1 shows a block diagram of the DTC.

The four DTC control registers (DTMR, DTSR, DTDR, and DTCR) are invisible to the CPU, but corresponding information is kept in a register information table in memory. A separate table is maintained for each DTC interrupt type. When an interrupt requests DTC service, the DTC loads its control registers from the table in memory, transfers the byte or word of data, and writes any altered register information back to memory.

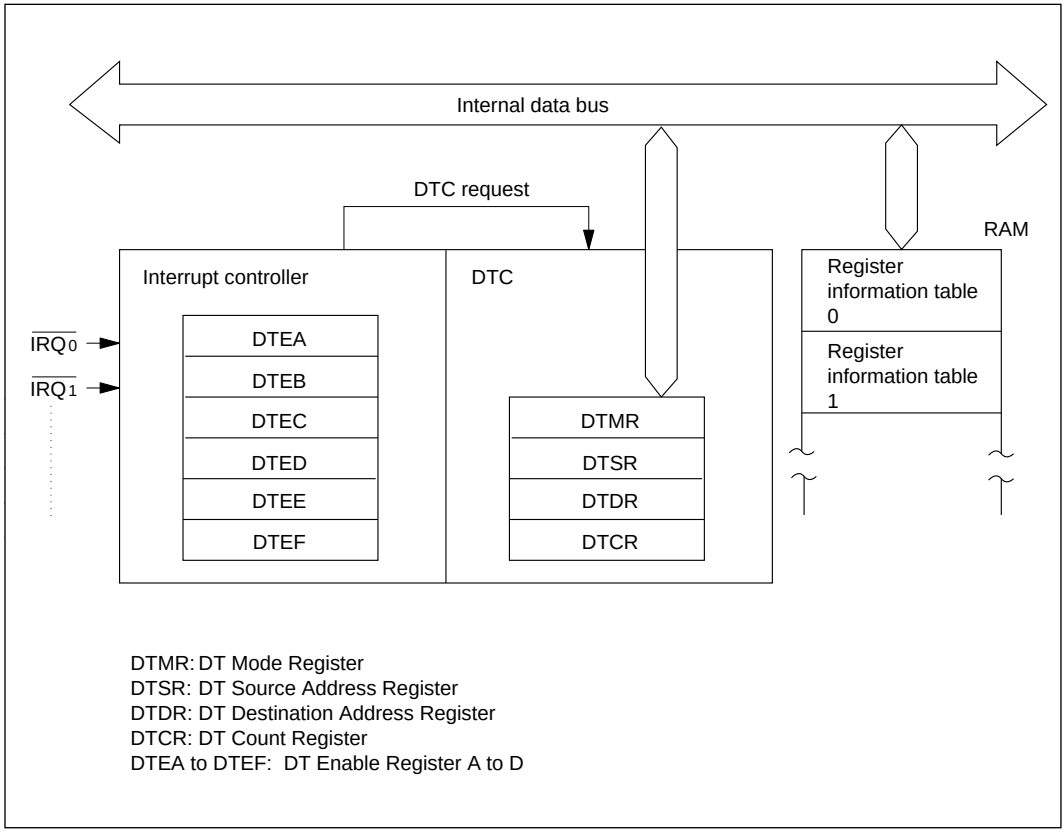


Figure 6-1 Block Diagram of Data Transfer Controller

6.1.3 Register Configuration

The four DTC control registers are listed in table 6-1. These registers are not located in the address space and cannot be written or read by the CPU. To set information in these registers, a program must write the information in a table in memory from which it will be loaded by the DTC.

Table 6-1 Internal Control Registers of the DTC

Name	Abbreviation	Read/Write
Data transfer mode register	DTMR	Disabled
Data transfer source address register	DTSR	Disabled
Data transfer destination address register	DTDR	Disabled
Data transfer count register	DTCR	Disabled

Starting of the DTC is controlled by the six data transfer enable registers, which are located in high addresses in page 0. Table 6-2 lists these registers.

Table 6-2 Data Transfer Enable Registers

Name	Abbreviation	Read/Write	Address	Initial Value
Data transfer enable register A	DTEA	R/W	H'FF08	H'00
Data transfer enable register B	DTEB	R/W	H'FF09	H'00
Data transfer enable register C	DTEC	R/W	H'FF0A	H'00
Data transfer enable register D	DTED	R/W	H'FF0B	H'00
Data transfer enable register E	DTEE	R/W	H'FF0C	H'00
Data transfer enable register F	DTEF	R/W	H'FF0D	H'00

6.2 Register Descriptions

6.2.1 Data Transfer Mode Register (DTMR)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Sz	SI	DI	—	—	—	—	—	—	—	—	—	—	—	—	—
Read/Write	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

The data transfer mode register is a 16-bit register, the first three bits of which designate the data size and specify whether to increment the source and destination addresses.

Bit 15—Sz (Size): This bit designates the size of the data transferred.

Bit 15

Sz	Description
0	Byte transfer
1	Word transfer* (two bytes at a time)

* For word transfer, the source and destination addresses must be even addresses.

Bit 14—SI (Source Increment): This bit specifies whether to increment the source address.

Bit 14

SI	Description
0	Source address is not incremented.
1	1) If Sz = 0: Source address is incremented by +1 after each data transfer. 2) If Sz = 1: Source address is incremented by +2 after each data transfer.

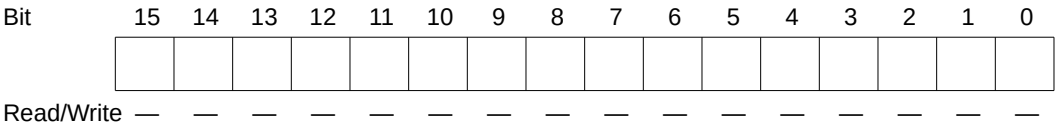
Bit 13—DI (Destination Increment): This bit specifies whether to increment the destination address.

Bit 13

DI	Description
0	Destination address is not incremented.
1	1) If Sz = 0: Destination address is incremented by +1 after each data transfer. 2) If Sz = 1: Destination address is incremented by +2 after each data transfer.

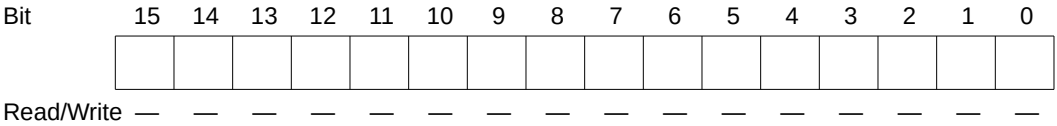
Bits 12 to 0—Reserved Bits: These bits are reserved.

6.2.2 Data Transfer Source Address Register (DTSR)



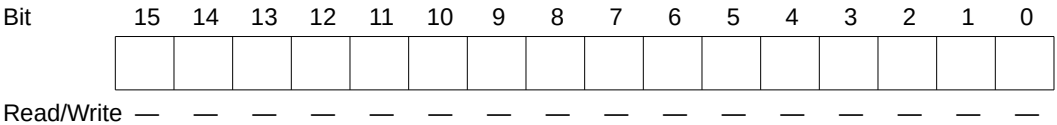
The data transfer source register is a 16-bit register that designates the data transfer source address. For word transfer this must be an even address. In the maximum mode, this address is implicitly located in page 0.

6.2.3 Data Transfer Destination Register (DTDR)



The data transfer destination register is a 16-bit register that designates the data transfer destination address. For word transfer this must be an even address. In the maximum mode, this address is implicitly located in page 0.

6.2.4 Data Transfer Count Register (DTCR)



The data transfer count register is a 16-bit register that counts the number of bytes or words of data remaining to be transferred. The initial count can be set from 1 to 65,536. A register value of 0 designates an initial count of 65,536.

The data transfer count register is decremented automatically after each byte or word is transferred. When its value reaches 0, indicating that the designated number of bytes or words have been transferred, a CPU interrupt is generated with the vector of the interrupt that requested the data transfer.

6.2.5 Data Transfer Enable Registers A to F (DTEA to DTEF)

These six registers designate whether an interrupt starts the DTC. The bits in these registers are assigned to interrupts as indicated in table 6-3. No bits are assigned to the NMI, FOVI, OVI, and ERI interrupts, which cannot request data transfers.

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

Table 6-3 Assignment of Data Transfer Enable Registers

Register	Interrupt Source or Module	Bits 7 to 4				Interrupt Source or Module	Bits 3 to 0			
		7	6	5	4		3	2	1	0
DTEA	$\overline{IRQ_0}$	—	—	—	IRQ_0	$\overline{IRQ_1}$	—	—	—	IRQ_1
DTEB	$\overline{IRQ_2}$, $\overline{IRQ_3}$	—	—	IRQ_3	IRQ_2	$\overline{IRQ_4}$, $\overline{IRQ_5}$	—	—	IRQ_5	IRQ_4
DTEC	16-Bit FRT1	—	OCIB1	OCIA1	ICI1	16-Bit FRT2	—	OCIB2	OCIA2	ICI2
DTED	16-Bit FRT3	—	OCIB3	OCIA3	ICI3	8-Bit Timer	—	—	CMIB	CMIA
DTEE	SCI1	—	TXI1	RXI1	—	SCI2	—	TXI2	RXI2	—
DTEF	A/D converter	—	—	—	ADI		—	—	—	—

Note: Bits marked "—" should always be cleared to 0.

If the bit for a certain interrupt is set to 1, that interrupt is regarded as a request for DTC service. If the bit is cleared to 0, the interrupt is regarded as a CPU interrupt request.

Only the interrupts indicated in table 6-3 can request DTC service. DTE bits not assigned to any interrupt (indicated by "—" in table 6-3) should be left cleared to 0.

• **Note on Timing of DTE Modifications:** The interrupt controller requires two system clock (ø) periods to determine the priority level of an interrupt. Accordingly, when an instruction modifies a data transfer enable register, the new setting does not take effect until the third state after taht instruction has been executed.

6.3 Data Transfer Operation

6.3.1 Data Transfer Cycle

When started by an interrupt, the DTC executes the following data transfer cycle:

1. From the DTC vector table, the DTC reads the address at which the register information table for that interrupt is located in memory.
2. The DTC loads the data transfer mode register and source address register from this table and reads the data (one byte or word) from the source address.
3. If so specified in the mode register, the DTC increments the source address register and writes the new source address back to the table in memory.
4. The DTC loads the data transfer destination address register and writes the byte or word of data to the destination address.
5. If so specified in the mode register, the DTC increments the destination address register and writes the new destination address back to the table in memory.
6. The DTC loads the data transfer count register from the table in memory, decrements the data count, and writes the new count back to memory.
7. If the data transfer count is now 0, the DTC generates a CPU interrupt. The interrupt vector is the vector of the interrupt type that started the DTC.

At an appropriate point during this procedure the DTC also clears the interrupt request by clearing the corresponding flag bit in the status register of the on-chip supporting module to 0.

But the DTC does not clear the data transfer enable bit in the data transfer enable register. This action, if necessary, must be taken by the user-coded interrupt-handling routine invoked at the end of the transfer.

The data transfer cycle is shown in a flowchart in figure 6-2.

For the steps from the occurrence of the interrupt up to the start of the data transfer cycle, see section 5.4.1, "Interrupt Handling Flow."

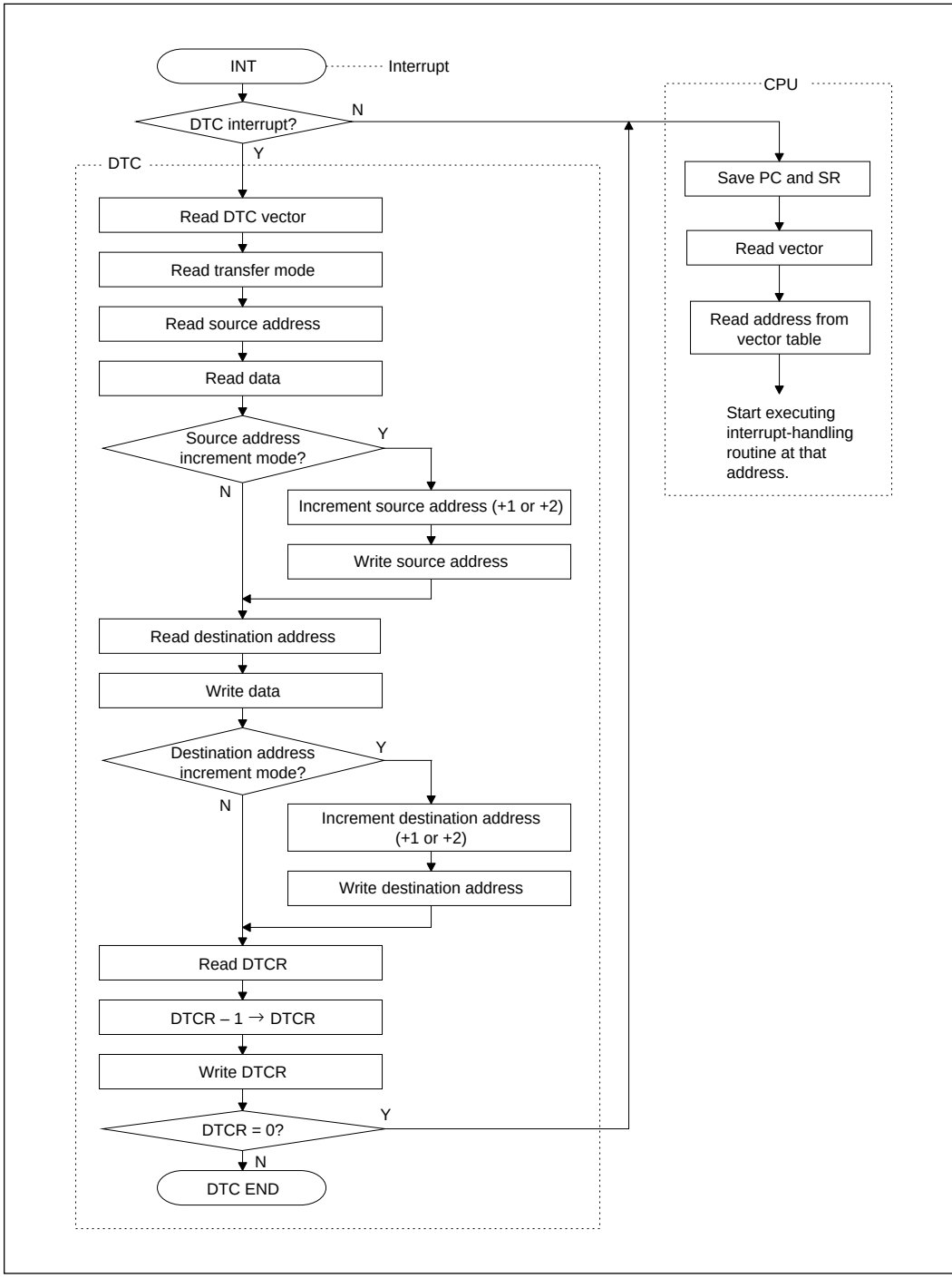
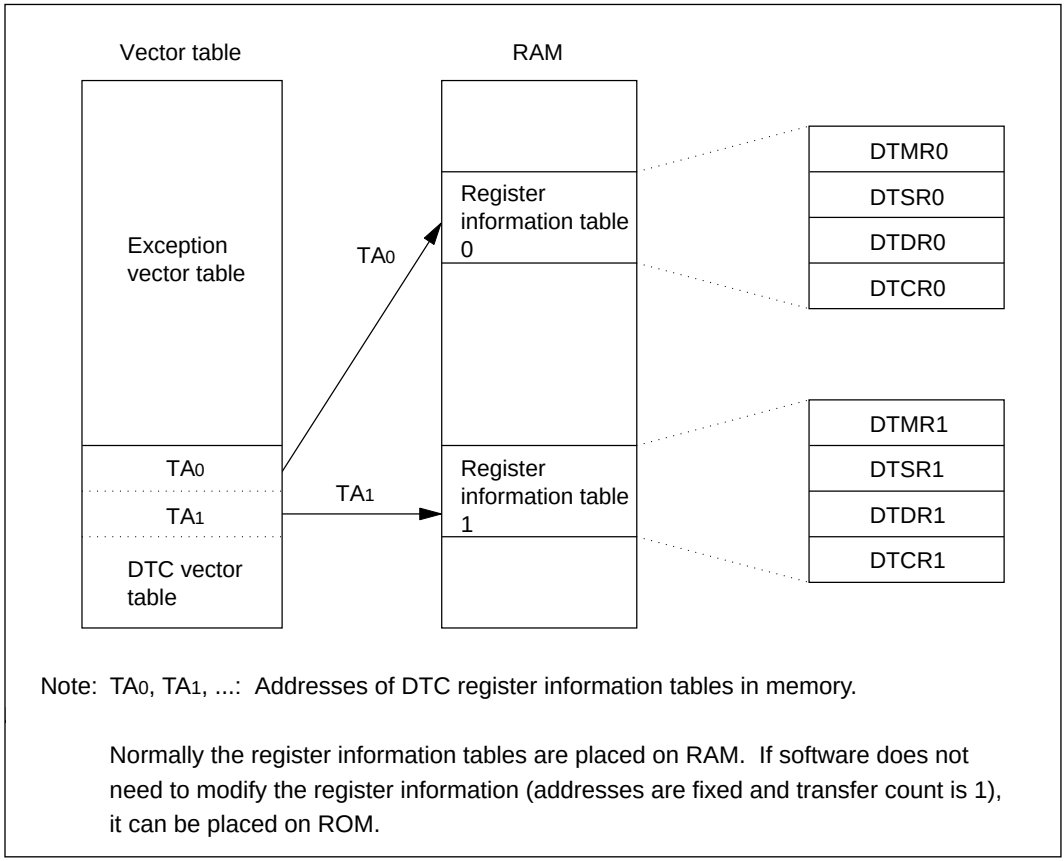


Figure 6-2 Flowchart of Data Transfer Cycle

6.3.2 DTC Vector Table

The DTC vector table is located immediately following the exception vector table at the beginning of page 0 in memory. For each interrupt that can request DTC service, the DTC vector table provides a pointer to an address in memory where the table of DTC control register information for that interrupt is stored. The register information tables can be placed in any available locations in page 0.



Note: TA_0, TA_1, ...: Addresses of DTC register information tables in memory.

Normally the register information tables are placed on RAM. If software does not need to modify the register information (addresses are fixed and transfer count is 1), it can be placed on ROM.

Figure 6-3 DTC Vector Table

In minimum mode, each entry in the DTC vector table consists of two bytes, pointing to an address in page 0. In maximum mode, for compatibility reasons, each DTC vector table entry consists of four bytes but the first two bytes are ignored; the last two bytes point to an address which is implicitly assumed to be in page 0, regardless of the current page specifications.

Figure 6-4 shows one DTC vector table entry in minimum and maximum mode.

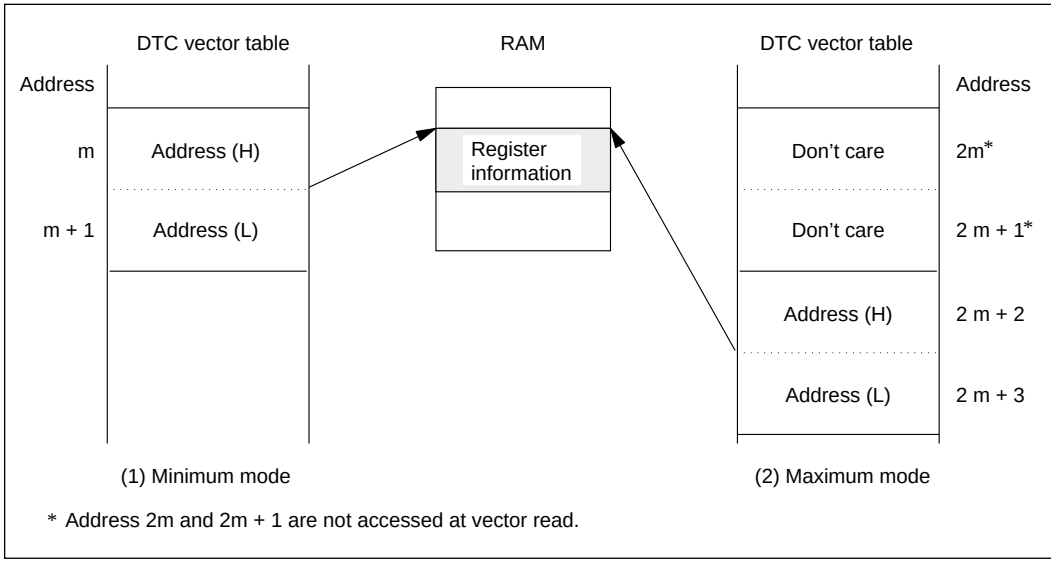


Figure 6-4 DTC Vector Table Entry

Table 6-4 lists the addresses of the entries in the DTC vector table for each interrupt.

Table 6-4 Addresses of DTC Vectors

		Address of DTC Vector	
Interrupt		**Minimum Mode**	**Maximum Mode**
IRQ_0		H'00C0 - H'00C1	H'0180 - H'0183
Interval timer		H'00C2 - H'00C3	H'0184 - H'0187
IRQ_1		H'00C8 - H'00C9	H'0190 - H'0193
IRQ_2		H'00D0 - H'00D1	H'01A0 - H'01A3
IRQ_3		H'00D2 - H'00D3	H'01A4 - H'01A7
IRQ_4		H'00D8 - H'00D9	H'01B0 - H'01B3
IRQ_5		H'00DA - H'00DB	H'01B4 - H'01B7
FRT1	ICI	H'00E0 - H'00E1	H'01C0 - H'01C3
	OCIA	H'00E2 - H'00E3	H'01C4 - H'01C7
	OCIB	H'00E4 - H'00E5	H'01C8 - H'01CB
FRT2	ICI	H'00E8 - H'00E9	H'01D0 - H'01D3
	OCIA	H'00EA - H'00EB	H'01D4 - H'01D7
	OCIB	H'00EC - H'00ED	H'01D8 - H'01DB
FRT3	ICI	H'00F0 - H'00F1	H'01E0 - H'01E3
	OCIA	H'00F2 - H'00F3	H'01E4 - H'01E7
	OCIB	H'00F4 - H'00F5	H'01E8 - H'01EB

Table 6-4 Addresses of DTC Vectors (cont)

Interrupt		Address of DTC Vector	
		Minimum Mode	Maximum Mode
8-Bit	CMIA	H'00F8 - H'00F9	H'01F0 - H'01F3
timer	CMIB	H'00FA - H'00FB	H'01F4 - H'01F7
SCI1	RXI	H'00A2 - H'00A3	H'0144 - H'0147
	TXI	H'00A4 - H'00A5	H'0148 - H'014B
SCI2	RXI	H'00AA - H'00AB	H'0154 - H'0157
	TXI	H'00AC - H'00AD	H'0158 - H'015B
A/D converter	ADI	H'00B0 - H'00B1	H'0160 - H'0163

6.3.3 Location of Register Information in Memory

For each interrupt, the DTC control register information is stored in four consecutive words in memory in the order shown in figure 6-5.

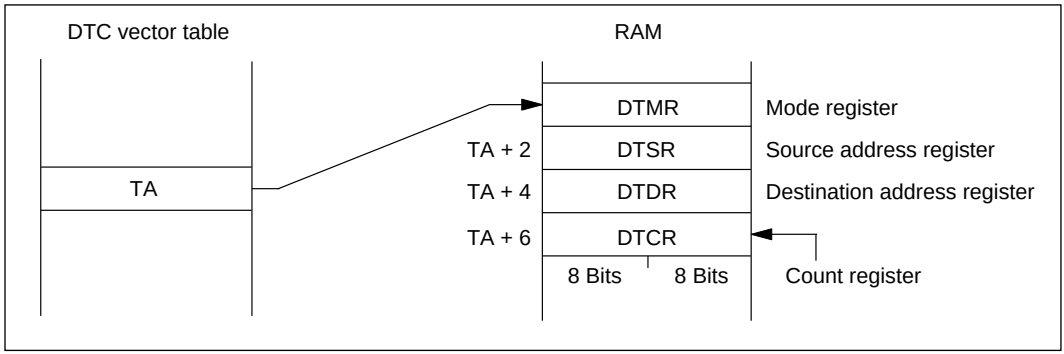


Figure 6-5 Order of Register Information

6.3.4 Length of Data Transfer Cycle

Table 6-5 lists the number of states required per data transfer, assuming that the DTC control register information is stored in on-chip RAM. This is the number of states required for loading and saving the DTC control registers and transferring one byte or word of data. Two cases are considered: a transfer between on-chip RAM and a register belonging to an I/O port or on-chip supporting module (i.e., a register in the register field from addresses H'FE80 to H'FFFF); and a transfer between such a register and external RAM.

Table 6-5 Number of States per Data Transfer

Increment Mode		On-Chip RAM ↔ Module or I/O Register		External RAM ↔ Module or I/O Register	
Source (SI)	Destination (DI)	Byte Transfer	Word Transfer	Byte Transfer	Word Transfer
0	0	31	34	32	38
0	1	33	36	34	40
1	0	33	36	34	40
1	1	35	38	36	42

Note: Numbers in the table are the number of states.

The values in table 6-5 are calculated from the formula:

$$N = 26 + 2 \times SI + 2 \times DI + M_S + M_D$$

Where M_S and M_D have the following meanings:

M_S: Number of states for reading source data

M_D: Number of states for writing destination data

The values of M_S and M_D depend on the data location as follows:

① Byte or word data in on-chip RAM: ⇨ 2 states

② Byte data in external RAM or register field: ⇨ 3 states

③ Word data in external RAM or register field: ⇨ 6 states

If the DTC control register information is stored in external RAM, $20 + 4 \times SI + 4 \times DI$ must be added to the values in table 6-5.

The values given above do not include the time between the occurrence of the interrupt request and the starting of the DTC. This time includes two states for the interrupt controller to check priority and a variable wait until the end of the current CPU instruction. At maximum, this time equals the sum of the values indicated for items No. 1 and 2 in table 6-6.

If the data transfer count is 0 at the end of a data transfer cycle, the number of states from the end of the data transfer cycle until the first instruction of the user-coded interrupt-handling routine is executed is the value given for item No. 3 in table 6-6.

Table 6-6 Number of States before Interrupt Service

No.	Reason for Wait		Number of States: Minimum Mode	Number of States: Maximum Mode
1	Interrupt priority decision and comparison with mask level in CPU status register		2 states	
2	Maximum number of states to completion of current instruction	Instruction is in on-chip memory	38 (LDM instruction specifying all registers)	
		Instruction is in external memory	74 + 16m (LDM instruction specifying all registers)	
3	Saving of PC and SR or PC, CP, and SR and instruction prefetch	Stack is in on-chip RAM	16	21
		Stack is in external memory	28 + 6m	41 + 10m

m: Number of wait states inserted in external memory access

6.4 Procedure for Using the DTC

A program that uses the DTC to transfer data must do the following:

1. Set the appropriate DTMR, DTSR, DTDR, and DTCR register information in the memory location indicated in the DTC vector table.
2. Set the data transfer enable bit of the pertinent interrupt to 1, and set the priority of the interrupt source (in the interrupt priority register) and the interrupt mask level (in the CPU status register) so that the interrupt can be accepted.
3. Set the interrupt enable bit in the control register for the interrupt source (or set the IRQ enable bit).

Following these preparations, the DTC will be started each time the interrupt occurs. When the number of bytes or words designated by the DTCR value have been transferred, after transferring the last byte or word, the DTC generates a CPU interrupt.

The user-coded interrupt-handling routine must take action to prepare for or disable further DTC data transfer: by readjusting the data transfer count, for example, or clearing the interrupt enable bit. If no action is taken, the next interrupt of the same type will start the DTC with an initial data transfer count of 65,536.

6.5 Example

Purpose: To receive 128 bytes of serial data the serial communication interface 1.

Conditions:

- Operating mode: Minimum mode
- Received data are to be stored in consecutive addresses starting at H'FC00.
- DTC control register information for the RXI interrupt is stored at addresses H'FB80 to H'FB87.
- Accordingly, the DTC vector table contains H'FB at address H'00A2 and H'80 at address H'00A3.
- The desired interrupt mask level in the CPU status register is 4, and the desired SCI1 interrupt priority level is 5.

Procedure

1. The user program sets DTC control register information in addresses H'FB80 to H'FB87 as shown in table 6-7.

Table 6-7 DTC Control Register Information Set in RAM

Address	Register	Description	Value Set
H'FB80	DTMR	Byte transfer Source address fixed Increment destination address	H'2000
H'FB82	DTSR	Address of SCI1 receive data register	H'FEDD
H'FB84	DTDR	Address H'FC00	H'FC00
H'FB86	DTCR	Number of bytes to be received: 128	H'0080

2. The program sets the RI (SCI1 Receive Interrupt) bit in the data transfer enable register (bit 5 of register DTEE) to 1.
3. The program sets the interrupt mask in the CPU status register to 4, and the SCI1 interrupt priority in bits 6 to 4 of interrupt priority register IPRE to 5.
4. The program sets SCI1 to the appropriate receive mode, and sets the receive interrupt enable (RIE) bit in the serial control register (SCR) to 1 to enable receive interrupts.
5. Thereafter, each time SCI1 receives one byte of data, it requests an RXI interrupt, which the interrupt controller directs toward the DTC. The DTC transfers the byte from the SCI's receive data register (RDR) into RAM, and clears the interrupt request before ending.

6. When 128 bytes have been transferred (DTCR = 0), the DTC generates a CPU interrupt. The interrupt type is RXI from SCI1.
7. The user-coded RXI interrupt-handling routine processes the received data and disables further data transfer (by clearing the RIE bit, for example).

Figure 6-6 shows the DTC vector table and data in RAM for this example.

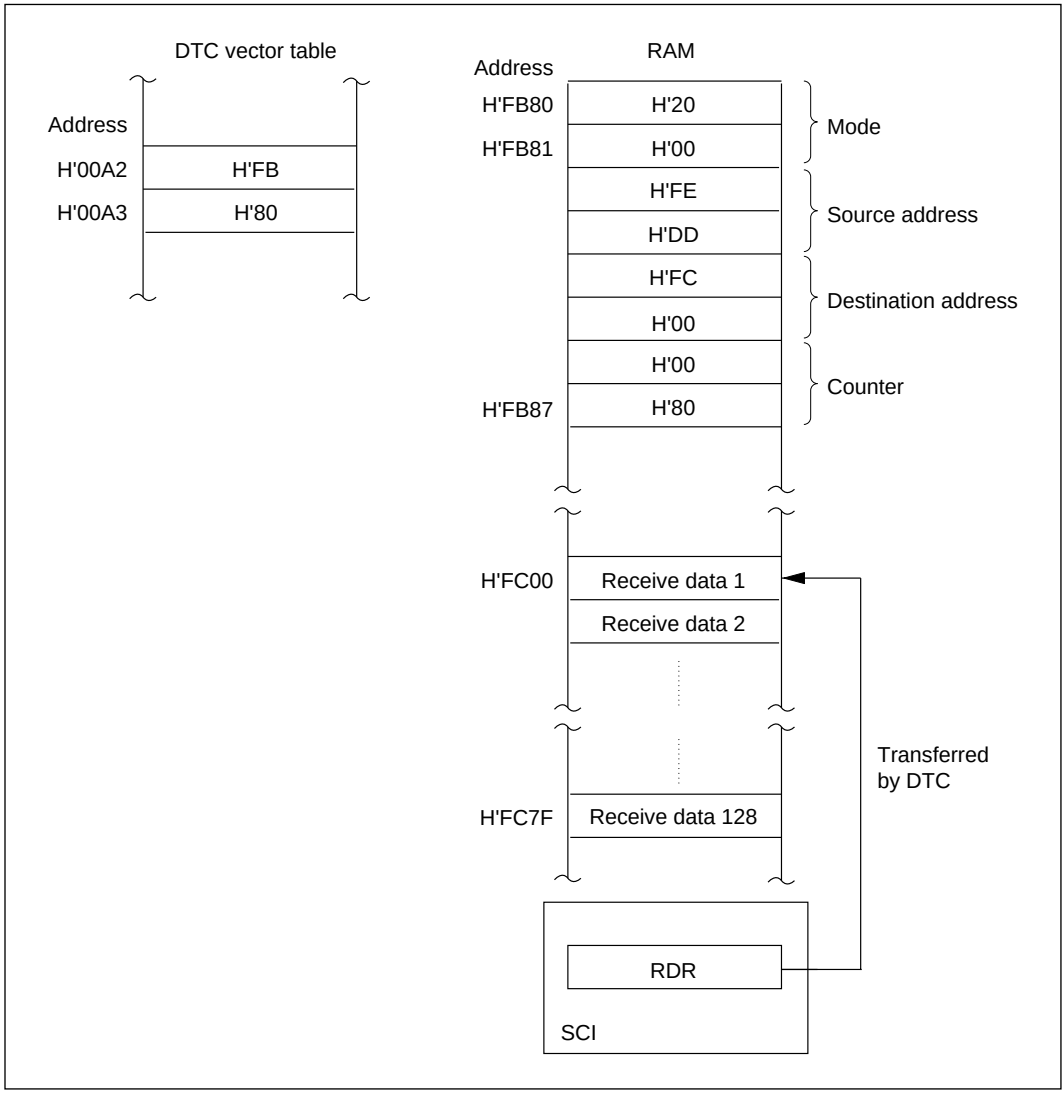


Figure 6-6 Use of DTC to Receive Data via Serial Communication Interface 1

Section 7 Wait-State Controller

7.1 Overview

To simplify interfacing to low-speed external devices, the H8/534 and H8/536 have an on-chip wait-state controller (WSC) that can insert wait states (T_W) to prolong bus cycles.

The wait-state function can be used in CPU and DTC access cycles to external addresses. It is not used in access to on-chip supporting modules. The T_W states are inserted between the T_2 state and T_3 state in the bus cycle. The number of wait states can be selected by a value set in the wait-state control register (WCR), or by holding the $\overline{\text{WAIT}}$ pin Low for the required interval.

7.1.1 Features

The main features of the wait-state controller are:

- Selection of three operating modes
  Programmable wait mode, pin wait mode, or pin auto-wait mode
- 0, 1, 2, or 3 wait states can be inserted.
  And in the pin wait mode, 4 or more states can be inserted by holding the $\overline{\text{WAIT}}$ pin Low.

7.1.2 Block Diagram

Figure 7-1 shows a block diagram of the wait-state controller.

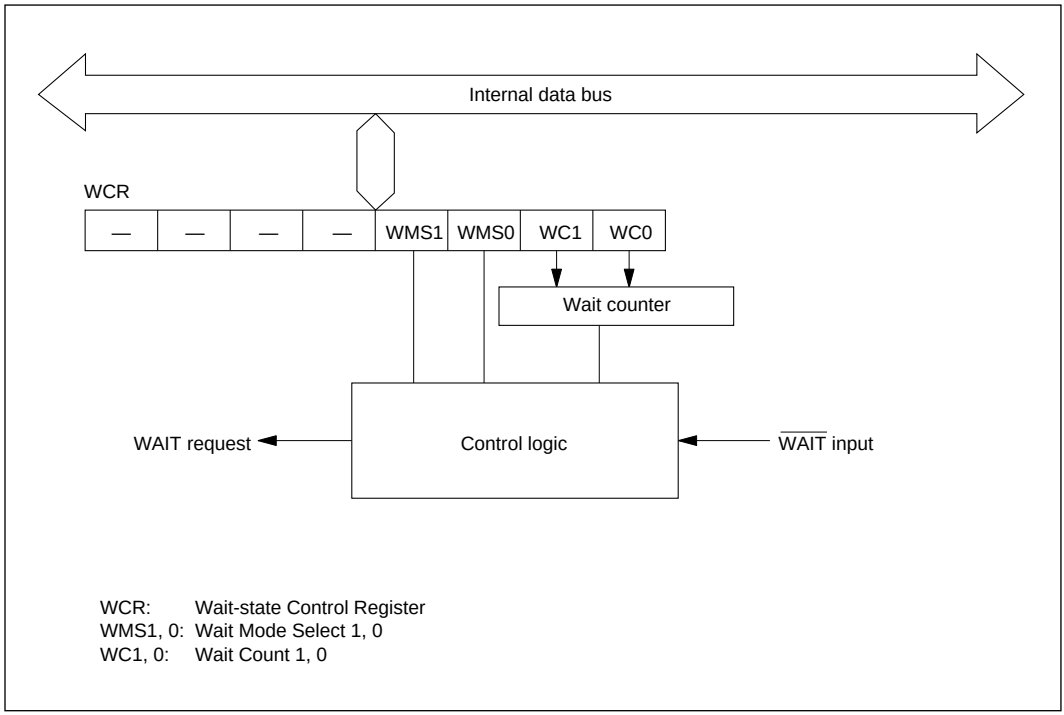


Figure 7-1 Block Diagram of Wait-State Controller

7.1.3 Register Configuration

The wait-state controller has one control register: the wait-state control register described in table 7-1.

Table 7-1 Register Configuration

Name	Abbreviation	Read/Write	Initial Value	Address
Wait-state control register	WCR	R/W	H'F3	H'FF10

7.2 Wait-State Control Register

The wait-state control register (WCR) is an 8-bit register that specifies the wait mode and the number of wait states to be inserted. A reset initializes the WCR to specify the programmable wait mode with three wait states. The WCR is not initialized in the software standby mode.

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	WMS1	WMS0	WC1	WC0
Initial value	1	1	1	1	0	0	1	1
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

Bits 7 to 4—Reserved: These bits cannot be modified and are always read as 1.

Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1 and WMS0): These bits select the wait mode as shown below.

Bit 3 WMS1	Bit 2 WMS0	Description
0	0	Programmable wait mode (Initial value)
0	1	No wait states are inserted, regardless of the wait count.
1	0	Pin wait mode
1	1	Pin auto-wait mode

Bits 1 and 0—Wait Count (WC1 and WC0): These bits specify the number of wait states to be inserted.
Wait states are inserted only in bus cycles in which the CPU or DTC accesses an external address.

Bit 1 WC1	Bit 0 WC0	Description
0	0	No wait states are inserted, except in pin wait mode.
0	1	1 Wait state is inserted.
1	0	2 Wait states are inserted.
1	1	3 Wait states are inserted. (Initial value)

7.3 Operation in Each Wait Mode

Table 7-2 summarizes the operation of the three wait modes.

Table 7-2 Wait Modes

Mode	$\overline{\text{WAIT}}$ Pin Function	Insertion Conditions	Number of Wait States Inserted
Programmable wait mode WMS1 = 0 WMS0 = 0	Disabled	Inserted on access to an off-chip address	1 to 3 wait states are inserted, as specified by bits WC0 and WC1.
Pin wait mode WMS1 = 1 WMS0 = 0	Enabled	Inserted on access to an off-chip address	0 to 3 wait states are inserted, as specified by bits WC0 and WC1, plus additional wait states while the $\overline{\text{WAIT}}$ pin is held Low.
Pin auto-wait mode WMS1 = 1 WMS0 = 1	Enabled	Inserted on access to an off-chip address if the $\overline{\text{WAIT}}$ pin is Low	1 to 3 wait states are inserted, as specified by bits WC0 and WC1.

7.3.1 Programmable Wait Mode

The programmable wait mode is selected when WMS1 = 0 and WMS0 = 0.

Whenever the CPU or DTC accesses an off-chip address, the number of wait states set in bits WC1 and WC0 are inserted. The $\overline{\text{WAIT}}$ pin is not used for wait control; it is available as an I/O pin.

Figure 7-2 shows the timing of the operation in this mode when the wait count is 1 (WC1 = 0, WC0 = 1).

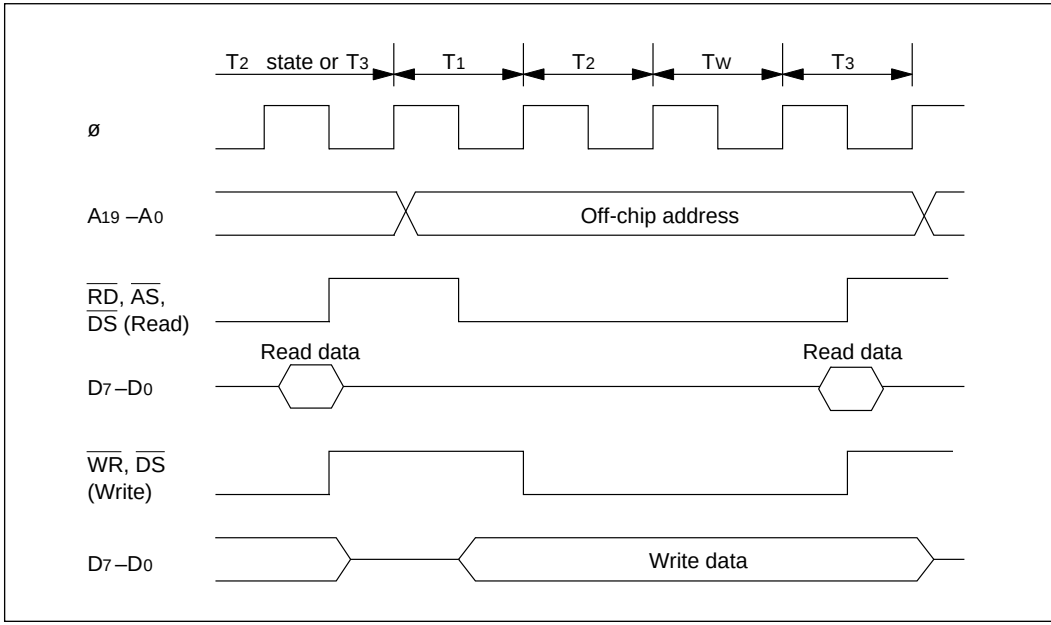


Figure 7-2 Programmable Wait Mode

7.3.2 Pin Wait Mode

The pin wait mode is selected when WMS1 = 1 and WMS0 = 0.

In this mode the $\overline{\text{WAIT}}$ function of the P14 /$\overline{\text{WAIT}}$ pin is used automatically.

The number of wait states indicated by bits WC1 and WC0 are inserted into any bus cycle in which the CPU or DTC accesses an off-chip address. In addition, wait states continue to be inserted as long as the $\overline{\text{WAIT}}$ pin is held low. In particular, if the wait count is 0 but the $\overline{\text{WAIT}}$ pin is Low at the rising edge of the ø clock in the T2 state, wait states are inserted until the $\overline{\text{WAIT}}$ pin goes High.

This mode is useful for inserting four or more wait states, or when different external devices require different numbers of wait states.

Figure 7-3 shows the timing of the operation in this mode when the wait count is 1 (WC1 = 0, WC0 = 1) and the $\overline{\text{WAIT}}$ pin is held Low to insert one additional wait state.

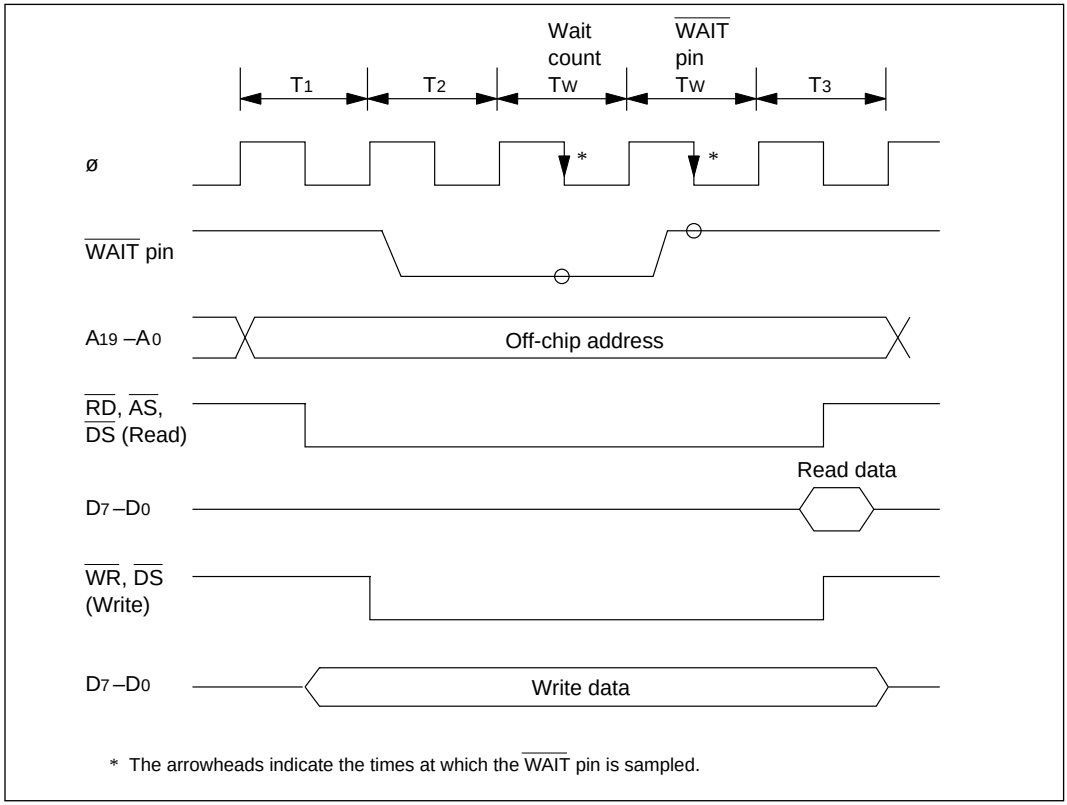


Figure 7-3 Pin Wait Mode

7.3.3 Pin Auto-Wait Mode

The pin auto-wait mode is selected when WMS1 = 1 and WMS0 = 1.

In this mode the $\overline{\text{WAIT}}$ function of the P14 /$\overline{\text{WAIT}}$ pin is used automatically.

In this mode, the number of wait states indicated by bits WC1 and WC0 are inserted, but only if there is a Low input at the $\overline{\text{WAIT}}$ pin.

Figure 7-4 shows the timing of this operation when the wait count is 1.

In the pin auto-wait mode, the $\overline{\text{WAIT}}$ pin is sampled only once, on the falling edge of the ø clock in the T_2 state. If the $\overline{\text{WAIT}}$ pin is Low at this time, the wait-state controller inserts the number of wait states indicated by bits WC1 and WC0. The $\overline{\text{WAIT}}$ pin is not sampled during the Tw and T_3 states, so no additional wait states are inserted even if the $\overline{\text{WAIT}}$ pin continues to be held Low.

This mode offers a simple way to interface a low-speed device: the wait states can be inserted by routing a decoded address signal to the $\overline{\text{WAIT}}$ pin.

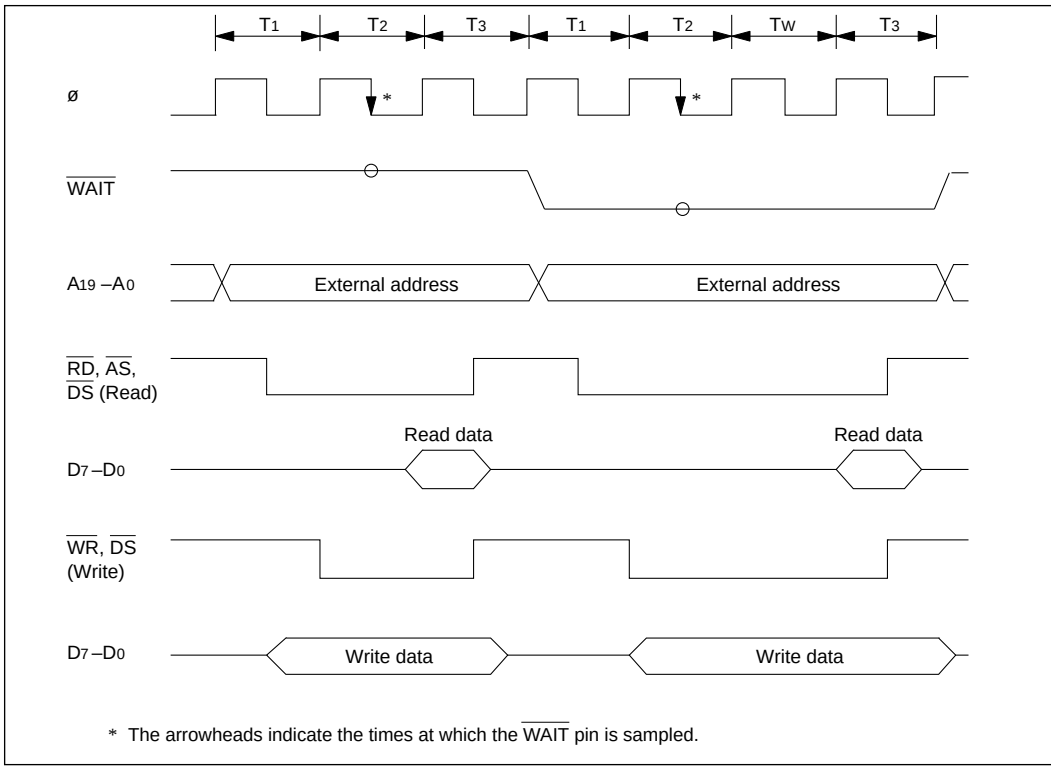


Figure 7-4 Pin Auto-Wait Mode

Section 8 Clock Pulse Generator

8.1 Overview

The H8/534 and H8/536 have a built-in clock pulse generator (CPG) consisting of an oscillator circuit, a system (ø) clock divider, an E clock divider, and a group of prescalers. The prescalers generate clock signals for the on-chip supporting modules.

8.1.1 Block Diagram

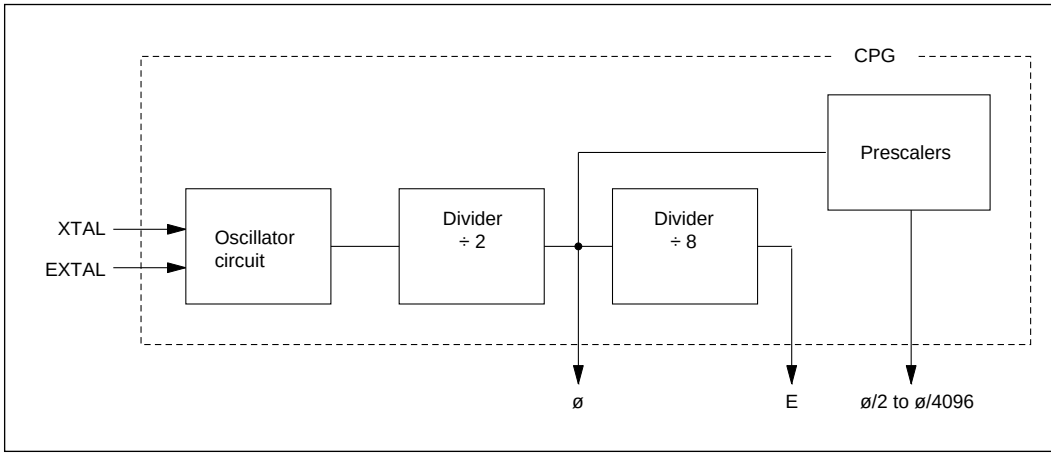


Figure 8-1 Block Diagram of Clock Pulse Generator

8.2 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.

Connecting an External Crystal

(1) **Circuit Configuration:** An external crystal can be connected as in the example in figure 8-2. An AT-cut parallel resonating crystal should be used.

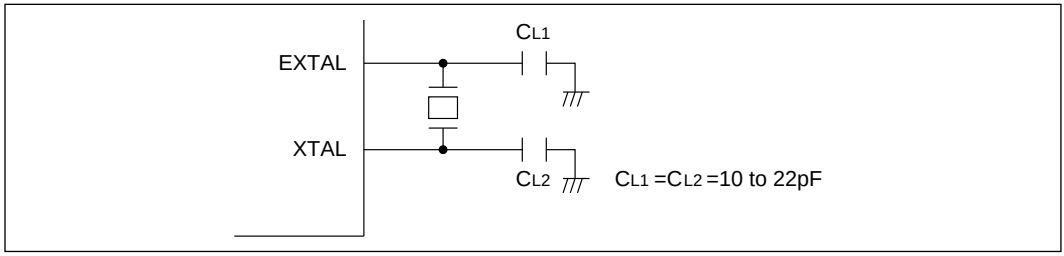


Figure 8-2 Connection of Crystal Oscillator (Example)

(2) **Crystal Oscillator:** The external crystal should have the characteristics listed in table 8-1.

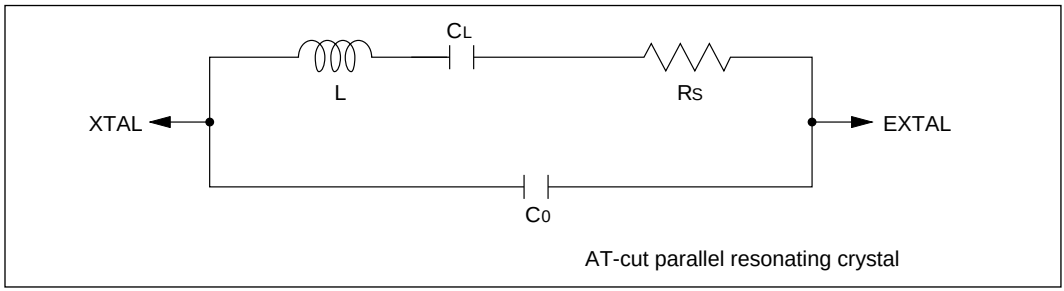


Figure 8-3 Crystal Oscillator Equivalent Circuit

Table 8-1 (1) External Crystal Parameters (HD6475368R, HD6475348R, HD6435368R, HD6435348R)

Frequency (MHz)	2	4	8	12	16	20
Rs max (Ω)	500	120	60	40	30	20
C_0 (pF)	7pF max					

Table 8-1 (2) External Crystal Parameters (HD6475368S, HD6475348S, HD6435368S, HD6435348S)

Frequency (MHz)	4	8	12	16	20	24
Rs max (Ω)	120	80	60	50	40	40
C_0 (pF)	7pF max					

Note: Use a fundamental-mode crystal (not an overtone 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 8-4.

When the board is designed, the crystal and its load capacitors should be placed as close as possible to the XTAL and EXTAL pins.

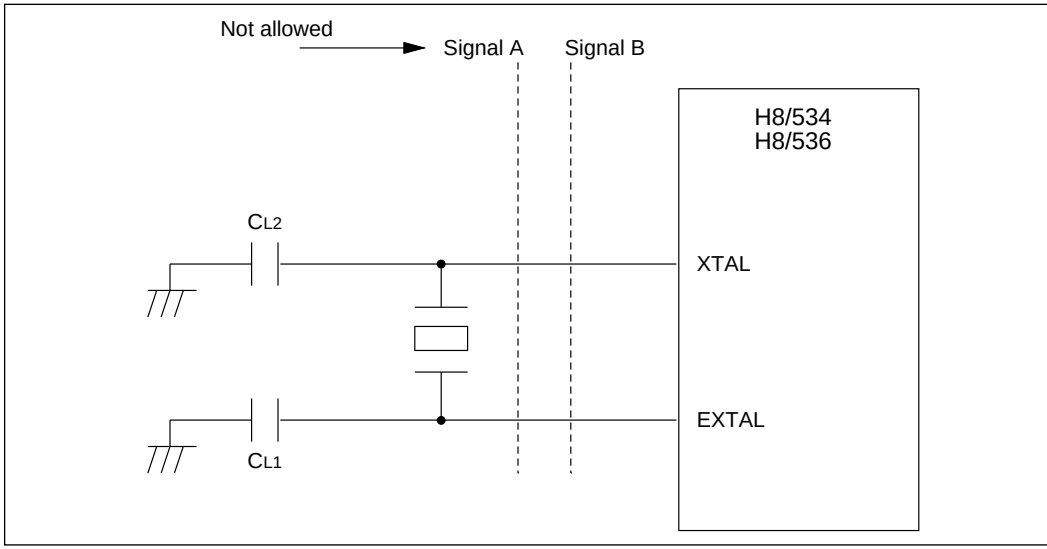


Figure 8-4 Notes on Board Design around External Crystal

Input of External Clock Signal

(1) **Circuit Configuration (HD6475368R, HD6475348R, HD6435368R, HD6435348R):** When using an external clock, input complementary clock signals to the EXTAL and XTAL pins as shown in figure 8-5. Make sure the external clock does not go high during standby mode.

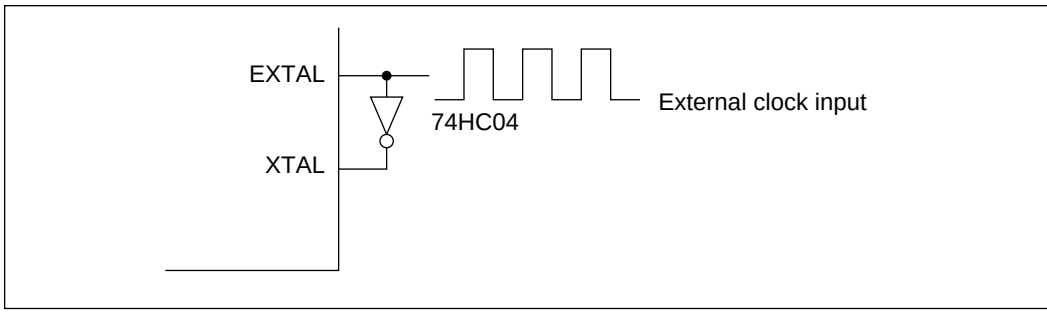


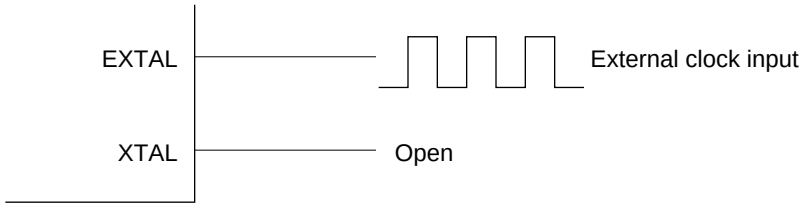
Figure 8-5 External Clock Input (Example)

(2) **External Clock Input**

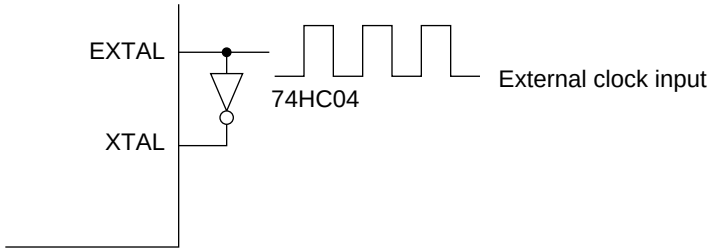
Frequency	Double the system clock (ø) frequency
Duty cycle	45% to 55%

Note: Mask-ROM versions can operate on external clock input to the EXTAL pin alone, with the XTAL pin left open. ZTAT™ versions can also operate with the XTAL pin left open if the external clock frequency is 16 MHz or less.

(3) **Circuit Configuration (HD6475368S, HD6475348S, HD6435368S, HD6435348S):** Figure 8-6 shows examples of external clock input. When using figure 8-6 (b), make sure the external clock does not go high during standby mode. When the XTAL pin is open, make sure the parasitic capacifance is less than 10 pF.



(a) XTAL pin left open



(b) Complementary clock input at XTAL pin

Figure 8-6 External Clock Input (Examples)

(4) **External Clock Input**

Frequency	Double the system clock (ø) frequency
Duty cycle	40% to 60%

8.3 System Clock Divider

The system clock divider divides the crystal oscillator or external clock frequency (fosc) by 2 to create the ø clock.

An E clock signal is created by dividing the ø clock by 8. The E clock is used for interfacing to E clock based devices.

Figure 8-7 shows the phase relationship of the E clock to the ø clock.

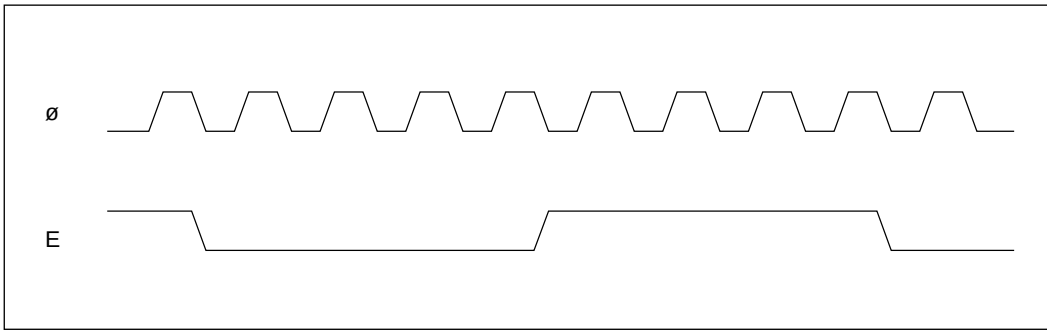


Figure 8-7 Phase Relationship of ø Clock and E Clock

Section 9 I/O Ports

9.1 Overview

The H8/534 and H8/536 have nine ports. Ports 1, 3, 4, 5, 7, and 9 are eight-bit input/output ports. Port 2 is a five-bit input/output port. Port 6 is a four-bit input/output port. Port 8 is an eight-bit input-only port. Table 9-1 summarizes the functions of each port.

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 page 0 of the address space. Each port (except port 8) also has a data direction register (DDR) which determines which pins are used for input and which for output. Additional system control registers (SYSCR1 and SYSCR2) control the functions of pins in ports 1, 6, and 9.

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.

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.

As table 9-1 indicates, all of the I/O port pins have dual functions. For example, pin 7 of port 1 can be used either as a general-purpose I/O pin ($P1_7$), or for output of the TMO signal from the on-chip 8-bit timer. The function is determined by the MCU operating mode, or by a value set in a control register.

Outputs from ports 1 to 6 can drive one TTL load and a 90 pF capacitive load. Outputs from ports 7 and 9 can drive one TTL load and a 30 pF capacitive load.

Outputs from ports 1 to 7 and 9 can also drive a Darlington transistor pair. Outputs from port 4 can drive a light-emitting diode (with 10mA current sink). Ports 5 and 6 have built-in MOS pull-ups for each input. Port 7 has Schmitt inputs.

Schematic diagrams of the I/O port circuits are shown in appendix C.

Table 9-1 Input/Output Port Summary

Port	Description	Pins	Expanded Modes				Single-Chip Mode
			Mode 1	Mode 2	Mode 3	Mode 4	(Mode 7)
Port 1	8-Bit input/output	P1$_7$ / TMO P1$_6$ / $\overline{IRQ_1}$ / $\overline{ADTRG}$ P1$_5$ / $\overline{IRQ_0}$	These input/output pins double as $\overline{IRQ_1}$, $\overline{IRQ_0}$, and $\overline{ADTRG}$ inputs, and as an output pin (TMO) for the 8-bit timer.				
		P1$_4$ / $\overline{WAIT}$ P1$_3$ / $\overline{BREQ}$ P1$_2$ / $\overline{BACK}$	These pins function as $\overline{WAIT}$, $\overline{BREQ}$, and $\overline{BACK}$ when necessary control-register bits are set to 1.				Input/output port
		P1$_1$ / E P1$_0$ / ø	These pins function as input pins or as clock (E, ø) output pins, depending on the data direction register setting.				
Port 2	5-Bit input/output port	P2$_4$ / $\overline{WR}$ P2$_3$ / $\overline{RD}$ P2$_2$ / $\overline{DS}$ P2$_1$ / R/$\overline{W}$ P2$_0$ / $\overline{AS}$	Bus control signal outputs ($\overline{WR}$, $\overline{RD}$, $\overline{DS}$, R/$\overline{W}$, $\overline{AS}$)				Input/output port
Port 3	8-Bit input/output port	P3$_7$ - P3$_0$ / D$_7$ – D$_0$	Data bus (D$_7$ – D$_0$)				Input/output port
Port 4	8-Bit input/output port Can drive a LED	P4$_7$ – P4$_0$ / A$_7$ – A$_0$	Low address bus (A$_7$ – A$_0$)				Input/output port
Port 5	8-Bit input/output port Built-in input pull-up (MOS)	P5$_7$ – P5$_0$ / A$_{15}$ – A$_8$	High address bus (A$_{15}$ – A$_8$)	High address bus if DDR is set to 1	High address bus (A$_{15}$ – A$_8$)	High address bus if DDR is set to 1	Input/output port
Port 6	4-Bit input/output port Built-in input pull-up (MOS)	P6$_3$ / PW$_3$ / $\overline{IRQ_5}$ / A$_{19}$ P6$_2$ / PW$_2$ / $\overline{IRQ_4}$ / A$_{18}$ P6$_1$ / PW$_1$ / $\overline{IRQ_3}$ / A$_{17}$ P6$_0$ / $\overline{IRQ_2}$ / A$_{16}$	Output for PWM timers 1, 2, and 3, input for $\overline{IRQ_2}$ to $\overline{IRQ_5}$, and input/output port.		Page address bus (A$_{19}$ – A$_{16}$)	Page address bus if DDR is set to 1, input port and $\overline{IRQ_2}$ to $\overline{IRQ_5}$ input pins if DDR is set to 0	Input/output port

Table 9-1 Input/Output Port Summary (cont)

Port	Description	Pins	Expanded Modes (Mode 1 Mode 2 Mode 3 Mode 4)	Single-Chip Mode (Mode 7)
Port 7	8-Bit input/output port (Schmitt inputs)	$P7_7$ / $FTOA_1$ $P7_6$ / $FTOB_3$ / $FTCI_3$ $P7_5$ / $FTOB_2$ / $FTCI_2$ $P7_4$ / $FTOB_1$ / $FTCI_1$ / $P7_3$ / FTI_3 TMRI $P7_2$ / FTI_2 $P7_1$ / FTI_1 $P7_0$ / TMCI	Input/output for free-running timers 1, 2 and 3 (FTI_1 to FTI_3, $FTCI_1$ to $FTCI_3$, $FTOB_1$ to $FTOB_3$, $FTOA_1$),input for 8-bit timer input (TMCI, TMRI), and 8-bit input/output port ($P7_7$ to $P7_0$)	
Port 8	8-Bit input port	$P8_0$ – $P8_7$ AN_7 – AN_0	Analog input pins for A/D converter, and 8-bit input port	
Port 9	8-Bit input/output port	$P9_7$ / SCK_1 $P9_6$ / RXD_1 $P9_5$ / TXD_1 $P9_4$ / SCK_2 / PW_3 $P9_3$ / RXD_2 / PW_2 $P9_2$ / TXD_2 / PW_1 $P9_1$ / $FTOA_3$ $P9_0$ / $FTOA_2$	Output for free-running timers 2 and 3 ($FTOA_2$, $FTOA_3$), PWM timer output (PW_1, PW_2, PW_3), serial communication interface (SCI1 and SCI2) input/output (SCK_1, RXD_1, TXD_1, SCK_2, RXD_2, TXD_2), and 8-bit input/output port	

9.2 Port 1

9.2.1 Overview

Port 1 is an 8-bit input/output port with the pin configuration shown in figure 9-1. All pins have dual functions, except that in the single-chip mode pins 4, 3, and 2 do not have the $\overline{\text{WAIT}}$, $\overline{\text{BREQ}}$, and $\overline{\text{BACK}}$ functions (because the CPU does not access an external bus).

Outputs from port 1 can drive one TTL load and a 90 pF capacitive load. They can also drive a Darlington transistor pair.

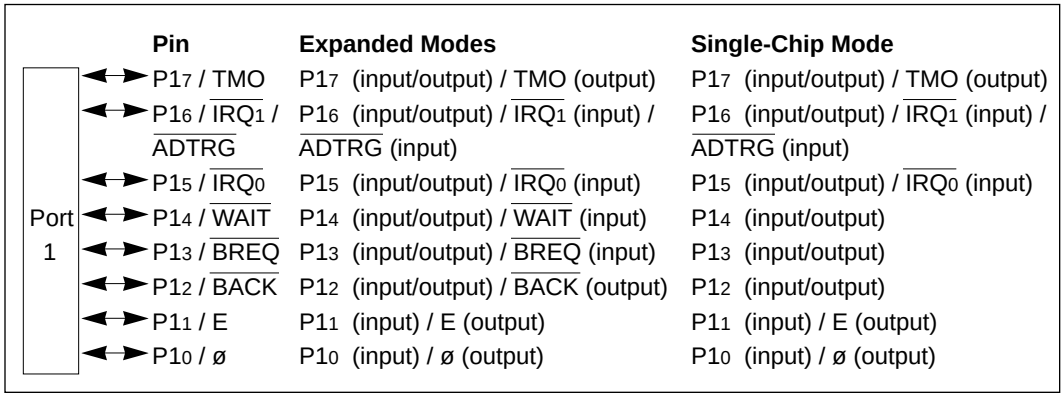


Figure 9-1 Pin Functions of Port 1

9.2.2 Port 1 Registers

Register Configuration: Table 9-2 lists the registers of port 1.

Table 9-2 Port 1 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 1 data direction register	P1DDR	W	H'03	H'FE80
Port 1 data register	P1DR	R/W*1	Undetermined*2	H'FE82
System control register 1	SYSCR1	R/W	H'87	H'FEFC

*1 Bits 1 and 0 are read-only.

*2 Bits 1 and 0 are undetermined. Other bits are initialized to 0.

1. Port 1 Data Direction Register (P1DDR)—H'FE80

Bit	7	6	5	4	3	2	1	0
	$P1_7DDR$	$P1_6DDR$	$P1_5DDR$	$P1_4DDR$	$P1_3DDR$	$P1_2DDR$	$P1_1DDR$	$P1_0DDR$
Initial value	0	0	0	0	0	0	1	1
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.

P1DDR can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

A reset initializes P1DDR to H'03, so that pins $P1_1$ and $P1_0$ carry clock outputs and the other pins are set for input. In the hardware standby mode, P1DDR is cleared to H'00, stopping the clock outputs. P1DDR is not initialized in the software standby mode, so if a P1DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 1 data register (or the ø or E clock).

2. Port 1 Data Register (P1DR)—H'FE82

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	—	—
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R

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 pin status directly.
Note that when pins $P1_1$ and $P1_0$ are used for output, they output the clock signals (ø and E), not the contents of P1DR. If the CPU reads Pl_1 and Pl_0 (when $Pl_1DDR = Pl_0DDR = 1$), it obtains the clock values at the current instant.

3. System Control Register 1 (SYSCR1)—H'FEFC

Bit	7	6	5	4	3	2	1	0
	—	IRQ_1E	IRQ_0E	NMIEG	BRLE	—	—	—
Initial value	1	0	0	0	0	1	1	1
Read/Write	—	R/W	R/W	R/W	R/W	—	—	—

SYSCR1 selects the functions of four of the port 1 pins. It also selects the input edge of the NMI pin.

At a reset and in the hardware standby mode, SYSCR1 is initialized to H'87. It is not initialized in the software standby mode.

Bit 7—Reserved: This bit cannot be modified and is always read as 1.

Bit 6—Interrupt Request 1 Enable (IRQ1E): This bit selects the function of pin P16.

Bit 6 IRQ1E	Description	
0	P16 functions as an input/output pin.	(Initial value)
1	P16 functions as the $\overline{\text{IRQ}}_1$ input pin, regardless of the value set in P16DDR. (However, the CPU can still read the pin status by reading P1DR.)	

Bit 5—Interrupt Request 0 Enable (IRQ0E): This bit selects the function of pin P15.

Bit 5 IRQ0E	Description	
0	P15 functions as an input/output pin.	(Initial value)
1	P15 functions as the $\overline{\text{IRQ}}_0$ input pin, regardless of the value set in P15DDR. (However, the CPU can still read the pin status by reading P1DR.)	

Bit 4—Nonmaskable Interrupt Edge (NMIEG): This bit selects the input edge of the NMI pin. It is not related to port 0.

Bit 4 NMIEG	Description	
0	A nonmaskable interrupt is generated on the falling edge of the input at the NMI pin.	(Initial value)
1	A nonmaskable interrupt is generated on the rising edge of the input at the NMI pin.	

Bit 3—Bus Release Enable (BRLE): This bit selects the functions of pins P12 and P13. It is valid only in the expanded modes (modes 1, 2, 3, and 4). In the single-chip mode, pins P12 and P13 function as input/output pins regardless of the value of the BRLE bit.

Bit 3 BRLE	Description	
0	P13 and P12 function as input/output pins.	(Initial value)
1	P13 functions as the $\overline{\text{BREQ}}$ input pin. P12 functions as the $\overline{\text{BACK}}$ output pin.	

Bits 2 to 0—Reserved: These bits cannot be modified and are always read as 1.

9.2.3 Pin Functions in Each Mode

Port 1 operates differently in the expanded modes (modes 1, 2, 3, and 4) and the single-chip mode (mode 7). Table 9-3 explains how the pin functions are selected in the expanded mode. Table 9-4 explains how the pin functions are selected in the single-chip mode.

Table 9-3 Port 1 Pin Functions in Expanded Modes

Pin — **Selection of Pin Functions**

P17 / TMO — The function depends on output select bits 3 to 0 (OS3 to OS0) of the 8-bit timer control/status register (TCSR) and on the P17DDR bit as follows:

OS3 to OS0	All four bits are 0		At least one bit is 1	
P17DDR	0	1	0	1
Pin function	P17 input	P17 output	TMO output	

P16 / $\overline{\text{IRQ1}}$ / $\overline{\text{ADTRG}}$ — The function depends on the IRQ1E bit and the trigger enable bit (TRGE) in the A/D control register (ADCR) as follows:

IRQ1E	0		1	
TRGE	0	1	0	1
Pin function	P16 input/ output	$\overline{\text{ADTRG}}$ input	$\overline{\text{IRQ1}}$ input	$\overline{\text{IRQ1}}$ and $\overline{\text{ADTRG}}$ input

When used for P16 input/output, the input or output function is selected by P16DDR.

P15 / $\overline{\text{IRQ0}}$ — The function depends on the IRQ0E bit and the P15DDR bit as follows:

IRQ0E	0		1	
P15DDR	0	1	0	1
Pin function	P15 input	P15 output	$\overline{\text{IRQ0}}$ input	

Table 9-3 Port 1 Pin Functions in Expanded Modes (cont)

Pin	Selection of Pin Functions
$P1_4$ / $\overline{WAIT}$	The function depends on the wait mode select 1 bit (WMS1) of the wait-state control register (WCR) and the $P1_4$DDR bit as follows:

WMS1	0		1	
$P1_4$DDR	0	1	0	1
Pin function	$P1_4$ input	$P1_4$ output	$\overline{WAIT}$ input	

$P1_3$ / $\overline{BREQ}$	The function depends on the BRLE bit and the $P1_3$DDR bit as follows:

BRLE	0		1	
$P1_3$DDR	0	1	0	1
Pin function	$P1_3$ input	$P1_3$ output	$\overline{BREQ}$ input	

$P1_2$ / $\overline{BACK}$	The function depends on the BRLE bit and the $P1_2$DDR bit as follows:

BRLE	0		1	
$P1_2$DDR	0	1	0	1
Pin function	$P1_2$ input	$P1_2$ output	$\overline{BACK}$ output	

$P1_1$ / E

$P1_1$DDR	0	1
Pin function	Input	E clock output

$P1_0$ / ø

$P1_0$DDR	0	1
Pin function	Input	ø clock output

Table 9-4 Port 1 Pin Functions in Single-Chip Modes

Pin	Selection of Pin Functions

P17 / TMO — The function depends on output select bits 3 to 0 (OS3 to OS0) of the 8-bit timer control/status register (TCSR) and on the P17DDR bit as follows:

OS3 to OS0	All four bits are 0		At least one bit is 1	
P17DDR	0	1	0	1
Pin function	P17 input	P17 output	TMO output	

P16 / $\overline{\text{IRQ1}}$ / $\overline{\text{ADTRG}}$ — The function depends on the IRQ1E bit and the trigger enable bit (TRGE) in the A/D control register (ADCR) as follows:

IRQ1E	0		1	
TRGE	0	1	0	1
Pin function	P16 input/ output	$\overline{\text{ADTRG}}$ input	$\overline{\text{IRQ1}}$ input	$\overline{\text{IRQ1}}$ and $\overline{\text{ADTRG}}$ input

When used for P16 input/output, the input or output function is selected by P16DDR.

P15 / $\overline{\text{IRQ0}}$ — The function depends on the IRQ0E bit and the P15DDR bit as follows:

IRQ0E	0		1	
P15DDR	0	1	0	1
Pin function	P15 input	P15 output	$\overline{\text{IRQ0}}$ input	

P14

P14DDR	0	1
Pin function	Input	Output

P13

P13DDR	0	1
Pin function	Input	Output

Table 9-4 Port 1 Pin Functions in Single-Chip Modes (cont)

Pin	Selection of Pin Functions		
$P1_2$	$P1_2$DDR	0	1
	Pin function	Input	Output
$P1_1$ / E	$P1_1$DDR	0	1
	Pin function	Input	E clock output
$P1_0$ / ø	$P1_0$DDR	0	1
	Pin function	Input	ø clock output

9.3 Port 2

9.3.1 Overview

Port 2 is a five-bit input/output port with the pin configuration shown in figure 9-2. It functions as an input/output port only in the single-chip mode. In the expanded modes it is used for output of bus control signals.

Outputs from port 2 can drive one TTL load and a 90 pF capacitive load. They can also drive a Darlington transistor pair.

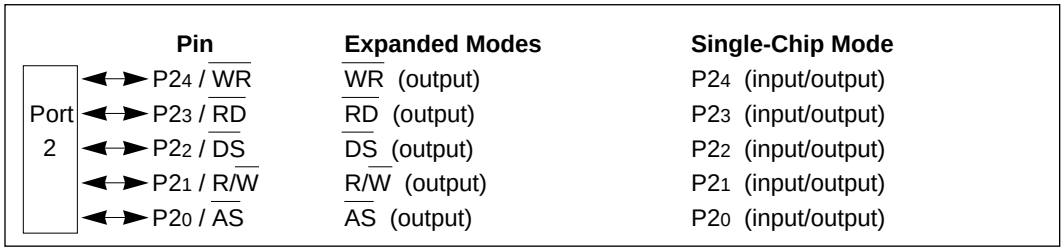


Figure 9-2 Pin Functions of Port 2

9.3.2 Port 2 Registers

Register Configuration: Table 9-5 lists the registers of port 2.

Table 9-5 Port 2 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 2 data direction register	P2DDR	W	H'E0	H'FE81
Port 2 data register	P2DR	R/W	H'E0	H'FE83

1. Port 2 Data Direction Register (P2DDR)—H'FE81

Bit	7	6	5	4	3	2	1	0
	—	—	—	$P2_4DDR$	$P2_3DDR$	$P2_2DDR$	$P2_1DDR$	$P2_0DDR$
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	W	W	W	W	W

P2DDR is an 8-bit register that selects the direction of each pin in port 2.

Single-Chip Mode: 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.

Bits 4 to 0 can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

Bits 7 to 5 are reserved. They cannot be modified and are always read as 1.

At a reset and in the hardware standby mode, P2DDR is initialized to H'E0, making all five pins input pins. P2DDR is not initialized in the software standby mode, so if a P2DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 2 data register.

Expanded Modes: All bits of P2DDR are fixed at 1 and cannot be modified.

2. Port 2 Data Register (P2DR)—H'FE83

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

P2DR is an 8-bit register containing the data for pins $P2_4$ to $P2_0$.

Bits 7 to 5 are reserved. They cannot be modified and are always read as 1.

When the CPU reads P2DR, for output pins it reads the value in the P2DR latch, but for input pins, it obtains the pin status directly.

9.3.3 Pin Functions in Each Mode

Port 2 has different functions in the expanded modes (modes 1, 2, 3, 4) and the single-chip mode (mode 7). Separate descriptions are given below.

Pin Functions in Expanded Modes: In the expanded modes (modes 1, 2, 3, and 4), all pins of P2DDR is automatically set to 1 for output. Port 2 outputs the bus control signals ($\overline{AS}$, R/$\overline{W}$, $\overline{DS}$, $\overline{RD}$, $\overline{WR}$).

Figure 9-3 shows the pin functions in the expanded modes.

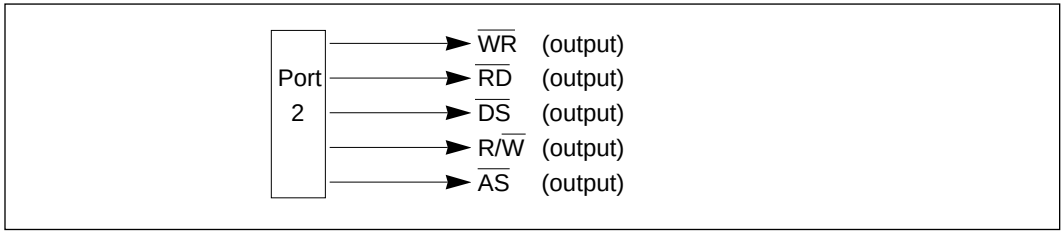


Figure 9-3 Port 2 Pin Functions in Expanded Modes

Pin Functions in Single-Chip Mode: In the single-chip mode (mode 7), each of the port 2 pins can be designated as an input pin or an output pin, as indicated in figure 9-4, by setting the corresponding bit in P2DDR to 1 for output or clearing it to 0 for input.

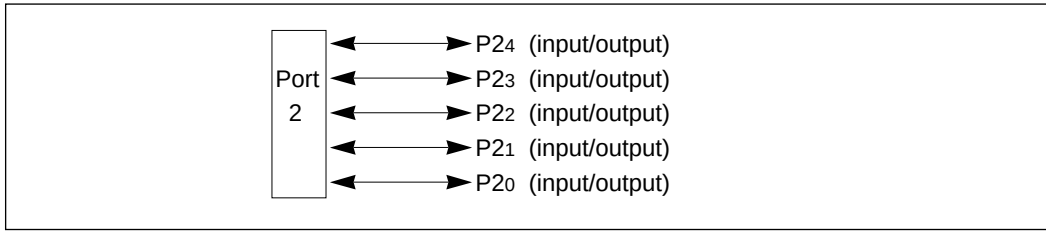


Figure 9-4 Port 2 Pin Functions in Single-Chip Mode

9.4 Port 3

9.4.1 Overview

Port 3 is an 8-bit input/output port with the pin configuration shown in figure 9-5. In the expanded modes it operates as the external data bus (D_7 – D_0). In the single-chip mode it operates as a general-purpose input/output port.

Outputs from port 3 can drive one TTL load and a 90pF capacitive load. They can also drive a Darlington transistor pair.

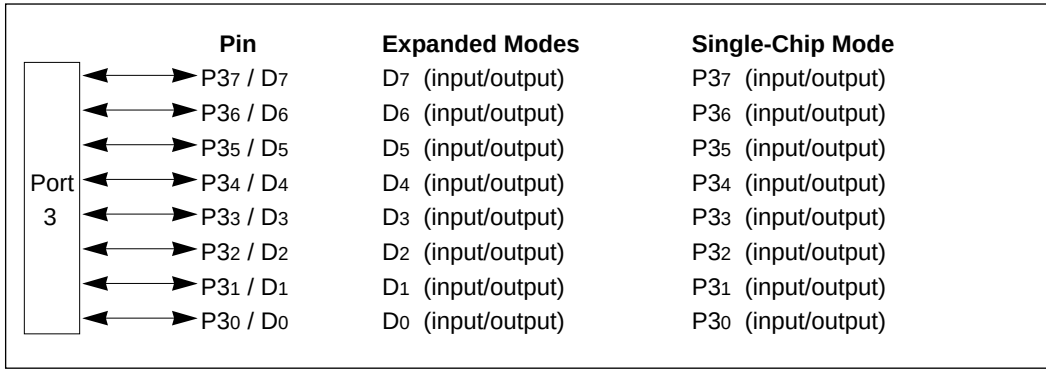


Figure 9-5 Pin Functions of Port 3

9.4.2 Port 3 Registers

Register Configuration: Table 9-6 lists the registers of port 3.

Table 9-6 Port 3 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 3 data direction register	P3DDR	W	H'00	H'FE84
Port 3 data register	P3DR	R/W	H'00	H'FE86

1. Port 3 Data Direction Register (P3DDR)—H'FE84

Bit	7	6	5	4	3	2	1	0
	$P3_7DDR$	$P3_6DDR$	$P3_5DDR$	$P3_4DDR$	$P3_3DDR$	$P3_2DDR$	$P3_1DDR$	$P3_0DDR$
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.

Single-Chip Mode: 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.

P3DDR can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

At a reset and in the hardware standby mode, P3DDR is initialized to H'00, making all eight pins input pins. P3DDR is not initialized in the software standby mode, so if a P3DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 3 data register.

Expanded Modes: P3DDR is not used.

2. Port 3 Data Register (P3DR)—H'FE86

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 $P3_0$.

At a reset and in the hardware standby mode, P3DR is initialized to H'00.

When the CPU reads P3DR, for output pins it reads the value in the P3DR latch, but for input pins, it obtains the pin status directly.

9.4.3 Pin Functions in Each Mode

Port 3 has different functions in the expanded modes (modes 1, 2, 3, 4) and the single-chip mode (mode 7). Separate descriptions are given below.

Pin Functions in Expanded Modes: In the expanded modes (modes 1, 2, 3, and 4), port 3 is automatically used as the data bus and P3DDR is ignored. Figure 9-6 shows the pin functions for the expanded modes.

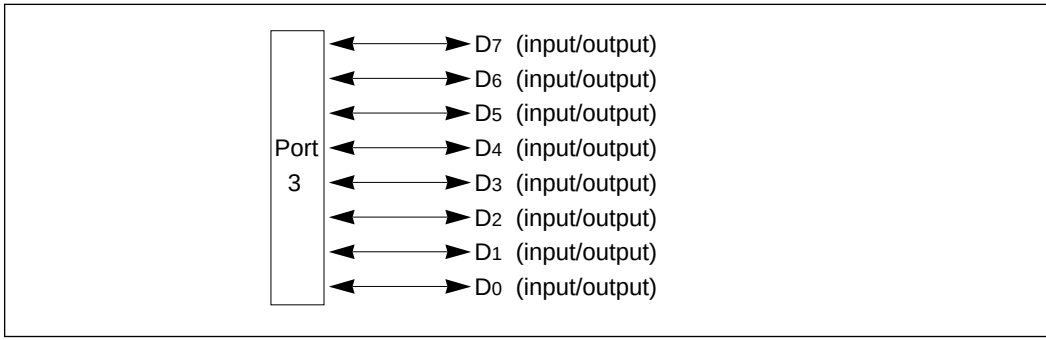


Figure 9-6 Port 3 Pin Functions in Expanded Modes

Pin Functions in Single-Chip Mode: In the single-chip mode (mode 7), each of the port 3 pins can be designated as an input pin or an output pin, as indicated in figure 9-7, by setting the corresponding bit in P3DDR to 1 for output or clearing it to 0 for input.

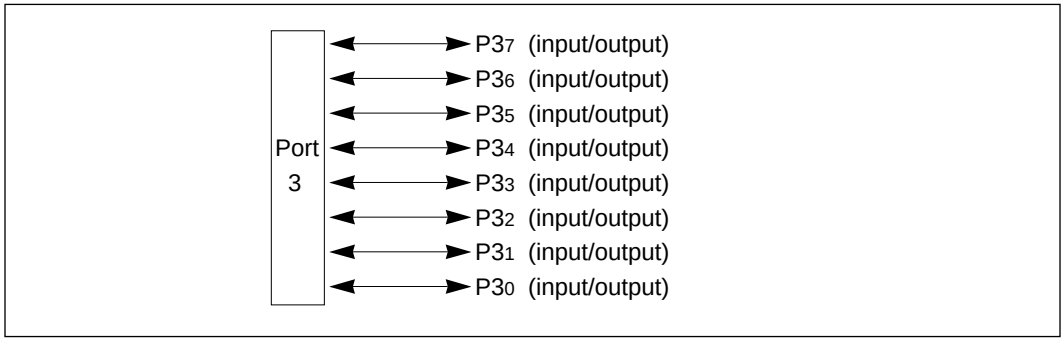


Figure 9-7 Port 3 Pin Functions in Single-Chip Mode

9.5 Port 4

9.5.1 Overview

Port 4 is an 8-bit input/output port with the pin configuration shown in figure 9-8. In the expanded modes it provides the low bits (A_7 – A_0) of the address bus. In the single-chip mode it operates as a general-purpose input/output port.

Outputs from port 4 can drive one TTL load and a 90 pF capacitive load. They can also drive a Darlington transistor pair or LED (with 10 mA current sink).

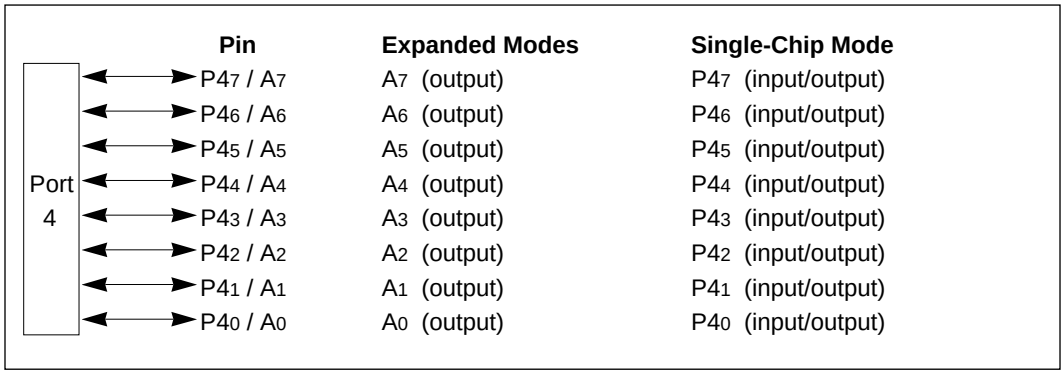


Figure 9-8 Pin Functions of Port 4

9.5.2 Port 4 Registers

Register Configuration: Table 9-7 lists the registers of port 4.

Table 9-7 Port 4 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 4 data direction register	P4DDR	W	H'00	H'FE85
Port 4 data register	P4DR	R/W	H'00	H'FE87

1. Port 4 Data Direction Register (P4DDR)—H'FE85

Bit	7	6	5	4	3	2	1	0
	$P4_7DDR$	$P4_6DDR$	$P4_5DDR$	$P4_4DDR$	$P4_3DDR$	$P4_2DDR$	$P4_1DDR$	$P4_0DDR$
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.

Single-Chip Mode: A pin functions as an output pin if the corresponding bit in P4DDR is set to 1, and as in input pin if the bit is cleared to 0.

P4DDR can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

At a reset and in the hardware standby mode, P4DDR is initialized to H'00, making all eight pins input pins. P4DDR is not initialized in the software standby mode, so if a P4DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 4 data register.

Expanded Modes: All bits of P4DDR are fixed at 1 and cannot be modified.

2. Port 4 Data Register (P4DR)—H'FE87

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 $P4_0$.

At a reset and in the hardware standby mode, P4DR is initialized to H'00.

When the CPU reads P4DR, for output pins it reads the value in the P4DR latch, but for input pins, it obtains the pin status directly.

9.5.3 Pin Functions in Each Mode

Port 4 has different functions in the expanded modes (modes 1, 2, 3, 4) and the single-chip mode (mode 7). Separate descriptions are given below.

Pin Functions in Expanded Modes: In the expanded modes (modes 1, 2, 3, and 4), port 4 is used for output of the low bits (A_7 – A_0) of the address bus. P4DDR is automatically set for output. Figure 9-9 shows the pin functions for the expanded modes.

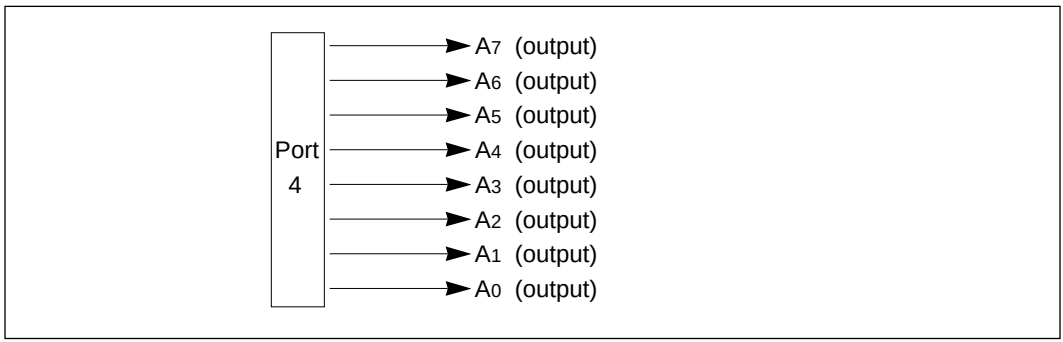


Figure 9-9 Port 4 Pin Functions in Expanded Modes

Pin Functions in Single-Chip Mode: In the single-chip mode (mode 7), each of the port 4 pins can be designated as an input pin or an output pin, as indicated in figure 9-10, by setting the corresponding bit in P4DDR to 1 for output or clearing it to 0 for input.

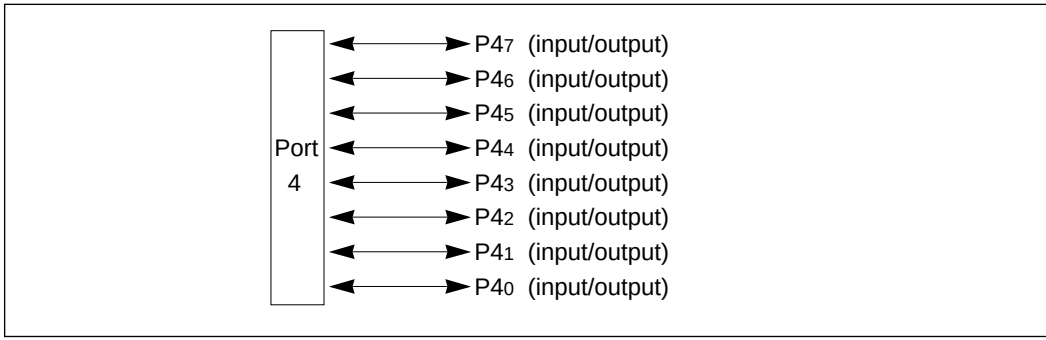


Figure 9-10 Port 4 Pin Functions in Single-Chip Mode

9.6 Port 5

9.6.1 Overview

Port 5 is an 8-bit input/output port with the pin configuration shown in figure 9-11. In the expanded modes that use the on-chip ROM (modes 2 and 4), the pins of port 5 function either as general-purpose input pins or as bits A_{15} – A_8 of the address bus, depending on the port 5 data direction register (P5DDR).

Port 5 has built-in MOS pull-ups that can be turned on or off under program control.

Outputs from port 5 can drive one TTL load and a 90 pF capacitive load. They can also drive a Darlington transistor pair.

	Pin	Modes 1 and 3	Modes 2 and 4	Single-Chip Mode
Port 5	$P5_7$ / A_{15}	A_{15} (output)	$P5_7$ (input) / A_{15} (output)	$P5_7$ (input/output)
	$P5_6$ / A_{14}	A_{14} (output)	$P5_6$ (input) / A_{14} (output)	$P5_6$ (input/output)
	$P5_5$ / A_{13}	A_{13} (output)	$P5_5$ (input) / A_{13} (output)	$P5_5$ (input/output)
	$P5_4$ / A_{12}	A_{12} (output)	$P5_4$ (input) / A_{12} (output)	$P5_4$ (input/output)
	$P5_3$ / A_{11}	A_{11} (output)	$P5_3$ (input) / A_{11} (output)	$P5_3$ (input/output)
	$P5_2$ / A_{10}	A_{10} (output)	$P5_2$ (input) / A_{10} (output)	$P5_2$ (input/output)
	$P5_1$ / A_9	A_9 (output)	$P5_1$ (input) / A_9 (output)	$P5_1$ (input/output)
	$P5_0$ / A_8	A_8 (output)	$P5_0$ (input) / A_8 (output)	$P5_0$ (input/output)

Figure 9-11 Pin Functions of Port 5

9.6.2 Port 5 Registers

Register Configuration: Table 9-8 lists the registers of port 5.

Table 9-8 Port 5 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 5 data direction register	P5DDR	W	H'00	H'FE88
Port 5 data register	P5DR	R/W	H'00	H'FE8A

1. Port 5 Data Direction Register (P5DDR)—H'FE88

Bit	7	6	5	4	3	2	1	0
	$P5_7DDR$	$P5_6DDR$	$P5_5DDR$	$P5_4DDR$	$P5_3DDR$	$P5_2DDR$	$P5_1DDR$	$P5_0DDR$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P5DDR is an 8-bit register that selects the direction of each pin in port 5.

Single-Chip Mode: 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.

P5DDR can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

At a reset and in the hardware standby mode, P5DDR is initialized to H'00, making all eight pins input pins. P5DDR is not initialized in the software standby mode, so if a P5DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 5 data register.

Expanded Modes Using On-Chip ROM (Modes 2 and 4): If a 1 is set in P5DDR, the corresponding pin is used for address output. If a 0 is set in P5DDR, the pin is used for general-purpose input. P5DDR is initialized to H'00 at a reset and in the hardware standby mode.

Expanded Modes Not Using On-Chip ROM (Modes 1 and 3): All bits of P5DDR are fixed at 1 and cannot be modified. Port 5 is used for address output.

Port 5 Data Register (P5DR)—H'FE8A

Bit	7	6	5	4	3	2	1	0
	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
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

P5DR is an 8-bit register containing the data for pins $P5_7$ to $P5_0$.

At a reset and in the hardware standby mode, P5DR is initialized to H'00.

When the CPU reads P5DR, for output pins it reads the value in the P5DR latch, but for input pins, it obtains the pin status directly.

9.6.3 Pin Functions in Each Mode

Port 5 operates in one way in modes 1 and 3, in another way in modes 2 and 4, and in a third way in mode 7. Separate descriptions are given below.

Pin Functions in Modes 1 and 3: In modes 1 and 3 (expanded modes in which the on-chip ROM is not used), all bits of P5DDR are automatically set to 1 for output, and the pins of port 5 carry bits A_{15} – A_8 of the address bus. Figure 9-12 shows the pin functions for modes 1 and 3.

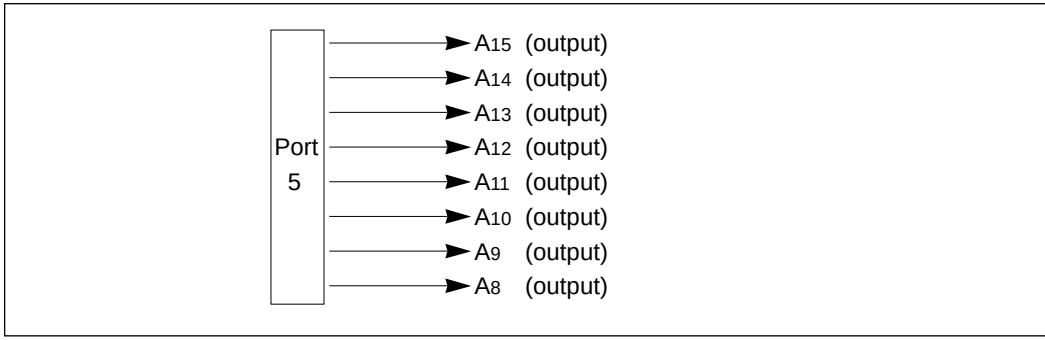


Figure 9-12 Port 5 Pin Functions in Modes 1 and 3

Pin Functions in Modes 2 and 4: In modes 2 and 4, (expanded modes in which the on-chip ROM is used), software can select whether to use port 5 for general-purpose input, or for output of bits A_{15} – A_8 of the address bus.

If a bit in P5DDR is set to 1, the corresponding pin is used for address output. If the bit is cleared to 0, the pin is used for input. A reset clears all P5DDR bits to 0, so before the address bus is used, all necessary bits in P5DDR must be set to 1.

Figure 9-13 shows the pin functions in modes 2 and 4.

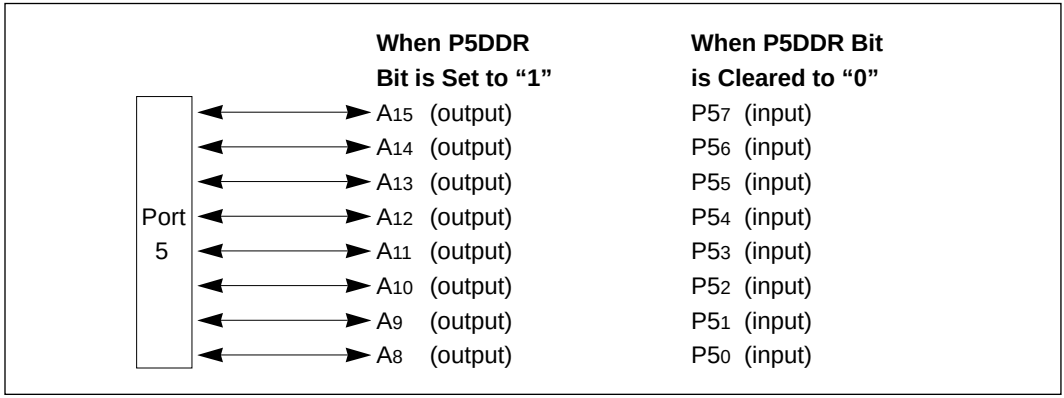


Figure 9-13 Port 5 Pin Functions in Modes 2 and 4

Pin Functions in Single-Chip Mode: In the single-chip mode (mode 7), each of the port 5 pins can be designated as an input pin or an output pin, as indicated in figure 9-14, by setting the corresponding bit in P5DDR to 1 for output or clearing it to 0 for input.

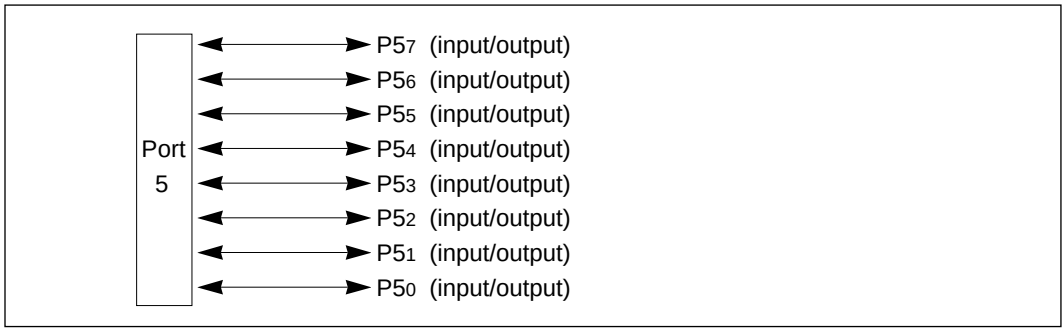


Figure 9-14 Port 5 Pin Functions in Single-Chip Mode

9.6.4 Built-In MOS Pull-Up

The MOS input pull-ups of port 5 are turned on by clearing the corresponding bit in P5DDR to 0 and writing a 1 in P5DR. These pull-ups are turned off at a reset and in the hardware standby mode. Table 9-9 indicates the status of the MOS pull-ups in various modes.

Table 9-9 Status of MOS Pull-Ups for Port 5

Mode	Reset	Hardware Standby Mode	Other Operating States*
1	OFF	OFF	OFF
2			ON/OFF
3			OFF
4 7			ON/OFF

* Including the software standby mode.

Notation:

OFF: The MOS pull-up is always off.

ON/OFF: The MOS pull-up is on when P5DDR = 0 and P5DR = 1, and off otherwise.

Note on Usage of MOS Pull-Ups

If the bit manipulation instructions listed below are executed on input/output ports 5 and 6 which have selectable MOS pull-ups, the logic levels at input pins will be transferred to the DR latches, causing the MOS pull-ups to be unintentionally switched on or off.

This can occur with the following bit manipulation instructions: BSET, BCLR, BNOT

(1) Specific Example (BSET Instruction): An example will be shown in which the BSET instruction is executed for port 5 under the following conditions:

P5$_7$: Input pin, low, MOS pull-up transistor on

P5$_6$: Input pin, high, MOS pull-up transistor off

P5$_5$ – P5$_0$: Output pins, low

The intended purpose of this BSET instruction is to switch the output level at P5$_0$ from low to high.

A: Before Execution of BSET Instruction

	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
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	On	Off	Off	Off	Off	Off	Off	Off

B: Execution of BSET Instruction

```
BSET.B  #0      @PORT5
```

;set bit 0 in data register

C: After Execution of BSET Instruction

	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
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 5. Since $P5_7$ and $P5_6$ are input pins, the CPU reads the level of these pins directly, not the value in the data register. It reads $P5_7$ as low (0) and $P5_6$ as high (1).

Since $P5_5$ to $P5_0$ are output pins, for these pins the CPU reads the value in the data register (0). The CPU therefore reads the value of port 5 as H'40, although the actual value in P5DR 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 P5DR to complete the BSET instruction.

As a result, bit $P5_0$ is set to 1, switching pin $P5_0$ to high output. In addition, bits $P5_7$ and $P5_6$ are both modified, changing the on/off settings of the MOS pull-up transistors of pins $P5_7$ and $P5_6$.

Programming Solution: The switching of the pull-ups for $P5_7$ and $P5_6$ in the preceding example can be avoided by using a byte in RAM as a work area for P5DR, performing bit manipulations on the work area, then writing the result to P5DR.

A: Before Execution of BSET Instruction

```
MOV.B  #80, R0      ;write data (H'80) for data register
MOV.B  R0,  @RAM0   ;write to work area (RAM0)
MOV.B  R0,  @PORT5  ;write to P5DR
```

	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
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	On	Off	Off	Off	Off	Off	Off	Off
RAM0	1	0	0	0	0	0	0	0

B: Execution of BSET Instruction

```
BSET.B  #0, @RAM0   ;set bit 0 in work area (RAM0)
```

C: After Execution of BSET Instruction

```
MOV.B  @RAM0, R0    ;get value in work area (RAM0)
MOV.B  R0, @PORT5   ;write value to P5DR
```

	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
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	On	Off	Off	Off	Off	Off	Off	Off
RAM0	1	0	0	0	0	0	0	0

9.7 Port 6

9.7.1 Overview

Port 6 is a 4-bit input/output port with the pin configuration shown in figure 9-15. In modes 7, 2, and 1, port 6 is used for IRQ_2 to IRQ_5 input and PWM timer output. In mode 4, port 6 is used for IRQ_2 to IRQ_5 input and page address output. In mode 3, port 6 is used for page address output.

Port 6 has built-in MOS pull-ups that can be turned on or off under program control.

Outputs from port 6 can drive one TTL load and a 90 pF capacitive load. They can also drive a Darlington transistor pair.

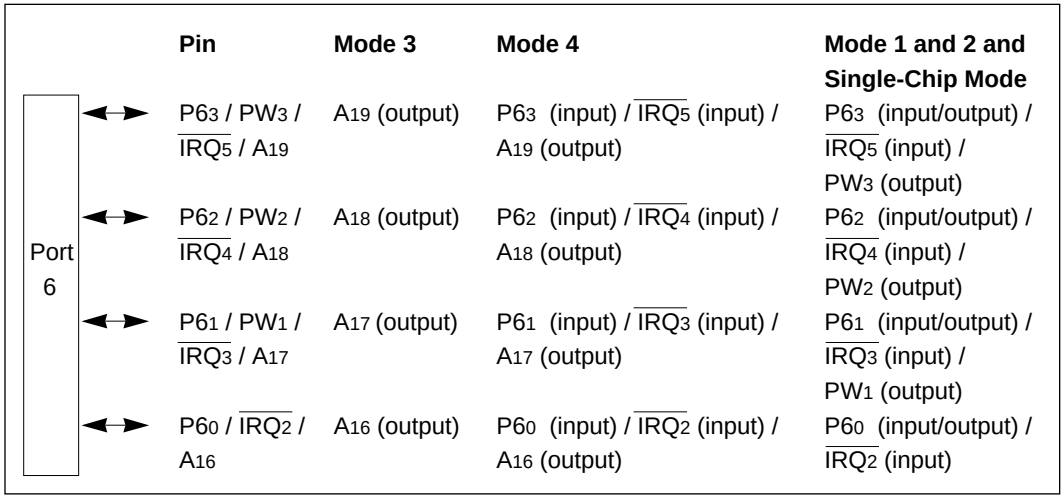


Figure 9-15 Pin Functions of Port 6

9.7.2 Port 6 Registers

Register Configuration: Table 9-10 lists the registers of port 6.

Table 9-10 Port 6 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 6 data direction register	P6DDR	W	H'F0	H'FE89
Port 6 data register	P6DR	R/W	H'F0	H'FE8B
System control register 2	SYSCR2	R/W	H'80	H'FEFD

1. Port 6 Data Direction Register (P6DDR)—H'FE89

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	P63DDR	P62DDR	P61DDR	P60DDR
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	W	W	W	W

P6DDR is an 8-bit register that selects the direction of each pin in port 6.

Single-Chip Mode and Expanded Minimum Modes: A pin functions as an output pin if the corresponding bit in P6DDR is set to 1, and as in input pin if the bit is cleared to 0.

Bits 7 to 4 are reserved. They cannot be modified and are always read as 1.

Bits 3 to 0 can be written but not read. An attempt to read these bits does not cause an error, but all bits are read as 1, regardless of their true values.

At a reset and in the hardware standby mode, P6DDR is initialized to H'F0, making all four pins input pins. P6DDR is not initialized in the software standby mode. In the single-chip mode, if a P6DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 6 data register.

Expanded Maximum Mode Using On-Chip ROM (Mode 4): If a 1 is set in P6DDR, the corresponding pin is used for address output. If a 0 is set in P6DDR, the pin is used for input. P6DDR is initialized to H'F0 at a reset and in the hardware standby mode.

Expanded Maximum Mode Not Using On-Chip ROM (Mode 3): All bits of P6DDR are fixed at 1 and cannot be modified.

2. Port 6 Data Register (P6DR)—H'FE8B

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

P6DR is an 8-bit register containing data for pins $P6_3$ to $P6_0$.

Bits 7 to 4 are reserved. They cannot be modified and are always read as 1.

At a reset and in the hardware standby mode, P6DR is initialized to H'F0.

When the CPU reads P6DR, for output pins it reads the value in the P6DR latch, but for input pins, it obtains the pin status directly.

3. System Control Register 2 (SYSCR2)—H'FEFD

Bit	7	6	5	4	3	2	1	0
	—	IRQ_5E	IRQ_4E	IRQ_3E	IRQ_2E	P6PWME	P9PWME	P9SCI2E
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

SYSCR2 controls the functions of port 6 and the functions of some pins in port 9.

SYSCR2 is initialized to H'80 by a reset and in the hardware standby mode. It is not initialized in the software standby mode.

Bit 7—Reserved: This bit cannot be modified and is always read as 1.

Bit 6—Interrupt Request 5 Enable (IRQ5E): Selects the function of pin P63.

Bit 6 IRQ5E	Description	
0	P63 functions as an input/output pin (but as the PW3 output pin if P6PWME and the OE bit of PWM timer 3 are both set to 1).	(Initial value)
1	P63 is the $\overline{\text{IRQ}}_5$ input pin regardless of the value of P63DDR (although the logic level of the pin can still be read).	

Bit 5—Interrupt Request 4 Enable (IRQ4E): Selects the function of pin P62.

Bit 5 IRQ4E	Description	
0	P62 functions as an input/output pin (but as the PW2 output pin if P6PWME and the OE bit of PWM timer 2 are both set to 1).	(Initial value)
1	P62 is the $\overline{\text{IRQ}}_4$ input pin regardless of the value of P62DDR (although the logic level of the pin can still be read).	

Bit 4—Interrupt Request 3 Enable (IRQ3E): Selects the function of pin P61.

Bit 4 IRQ3E	Description	
0	P61 functions as an input/output pin (but as the PW1 output pin if P6PWME and the OE bit of PWM timer 1 are both set to 1).	(Initial value)
1	P61 is the $\overline{\text{IRQ}}_3$ input pin regardless of the value of P61DDR (although the logic level of the pin can still be read).	

Bit 3—Interrupt Request 2 Enable (IRQ2E): Selects the function of pin P60.

Bit 3 IRQ2E	Description	
0	P60 functions as an input/output pin.	(Initial value)
1	P60 is the $\overline{\text{IRQ}}_2$ input pin regardless of the value of P60DDR (although the logic level of the pin can still be read).	

Bit 2—Port 6 PWM Enable (P6PWME): Controls pin functions of port 6.

Bit 2 P6PWME	Description
0	$P6_3$ to $P6_1$ function as input/output pins (Initial value) (or as $\overline{IRQ}$ input pins when bits IRQ5E to IRQ3E are set to 1).
1	$P6_3$ to $P6_1$ function as PWM output pins if the corresponding OE bit of PWM3 to PWM1 is set to 1. If the OE bit is cleared to 0 or the IRQE bit is set to 1, the pin functions as an input/output pin.

Bit 1—Port 9 PWM Enable (P9PWME): Controls pin functions of port 9.

Bit 1 P9PWME	Description
0	The PWM functions of $P9_4$ to $P9_2$ are disabled. (Initial value) (See section 9.10.3, “Pin Functions.”)
1	The PWM functions of $P9_4$ to $P9_2$ are enabled. (See section 9.10.3, “Pin Functions.”)

Bit 0—Port 9 SCI2 Enable (P9PWME): Controls pin functions of port 9.

Bit 1 P9SCI2E	Description
0	The serial communication interface functions of $P9_4$ to $P9_2$ (Initial value) are disabled. (See section 9.10.3, “Pin Functions.”)
1	The serial communication interface functions of P94 to P92 are enabled. (See section 9.10.3, “Pin Functions.”)

9.7.3 Pin Functions in Each Mode

The usage of port 6 depends on the MCU operating mode. Separate descriptions are given below.

Pin Functions in Mode 3: In mode 3 (the expanded maximum mode in which the on-chip ROM is not used), P6DDR is automatically set for output, and the pins of port 6 carry the page address bits (A_{19} – A_{16}) of the address bus. Figure 9-16 shows the pin functions for mode 3.

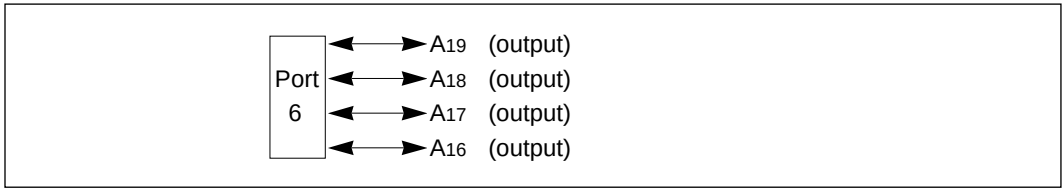


Figure 9-16 Port 6 Pin Functions in Mode 3

Pin Functions in Mode 4: In mode 4, (the expanded maximum mode in which the on-chip ROM is used), software can select whether to use port 6 for general-purpose input, $\overline{IRQ_2}$ to $\overline{IRQ_5}$ input, or output of page address bits.

If a bit in P6DDR is set to 1, the corresponding pin is used for page address output. If the P6DDR bit is cleared to 0 and the corresponding IRQnE bit is cleared to 0, the pin is used for general-purpose input. If the P6DDR bit is cleared to 0 and the corresponding IRQnE bit is set to 1, the pin is used for $\overline{IRQ_2}$ to $\overline{IRQ_5}$ input. A reset initializes these pins to the general-purpose input function, so when the address bus is used, all necessary bits in P6DDR must first be set to 1.

Figure 9-17 shows the pin functions in mode 4.

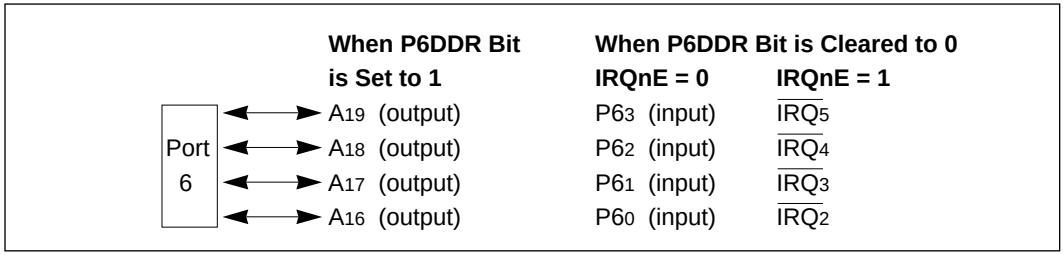


Figure 9-17 Port 6 Pin Functions in Mode 4

Pin Functions in Single-Chip Mode and Expanded Minimum Modes: In the single-chip mode (mode 7) and expanded minimum modes (modes 1 and 2), the port 6 pins can be designated individually as input or output pins.

Port 6 can be used for general-purpose input/output, $\overline{IRQ}$ input, or PWM output, depending on the combination of settings of the IRQE and P6PWME bits in system control register 2 and the OE

bits of the three PWM timers.

Figure 9-18 shows the pin functions in modes 7, 2, and 1.

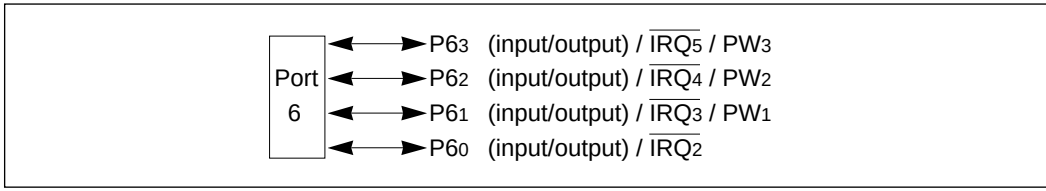


Figure 9-18 Port 6 Pin Functions in Modes 7, 2, and 1

Table 9-11 Port 6 Pin Functions in Modes 7, 2, and 1

Pin	Selection of Pin Functions
P63 / $\overline{\text{IRQ5}}$ / PW3	The function depends on the interrupt request 5 enable bit (IRQ5E) and port 6 PWM enable bit (P6PWME) in system control register 2 (SYSCR2), and the output enable bit (OE) of PWM timer 3.

IRQ5E	0				1			
P6PWME	0		1		0		1	
OE	0	1	0	1	0	1	0	1
Pin function	P63 input/output		PW3 output		$\overline{\text{IRQ5}}$ input		$\overline{\text{IRQ5}}$ input	

When used for P63 input/output, the input or output function is selected by P63DDR.

P62 / $\overline{\text{IRQ4}}$ / PW2	The function depends on the interrupt request 4 enable bit (IRQ4E) and P6PWME bit in SYSCR2, and the OE bit of PWM timer 2.

IRQ4E	0				1			
P6PWME	0		1		0		1	
OE	0	1	0	1	0	1	0	1
Pin function	P62 input/output		PW2 output		$\overline{\text{IRQ4}}$ input		$\overline{\text{IRQ4}}$ input	

When used for P62 input/output, the input or output function is selected by P62DDR.

Table 9-11 Port 6 Pin Functions in Modes 7, 2, and 1 (cont)

Pin	Selection of Pin Functions
$P6_1$ / $\overline{IRQ_3}$ / PW_1	The function depends on the interrupt request 3 enable bit (IRQ_3E) and P6PWME bit in SYSCR2, and the OE bit of PWM timer 1.

IRQ_3E	0				1			
P6PWME	0		1		0		1	
OE	0	1	0	1	0	1	0	1
Pin function	$P6_1$ input/output		PW_1 output		$\overline{IRQ_3}$ input		$\overline{IRQ_3}$ input	

When used for $P6_1$ input/output, the input or output function is selected by $P6_1$DDR.

$P6_0$ / $\overline{IRQ_2}$	The function depends on the interrupt request 2 enable bit (IRQ_2E) in SYSCR2.

IRQ_2E	0	1
Pin function	$P6_0$ input/output	$\overline{IRQ_2}$ input

When used for $P6_0$ input/output, the input or output function is selected by $P6_0$DDR.

9.7.4 Built-In MOS Pull-Up

Port 6 has programmable MOS input pull-ups which are turned on by clearing the corresponding bit in P6DDR to 0 and writing a 1 in P6DR. These pull-ups are turned off at a reset and in the hardware standby mode. Table 9-12 indicates the status of the MOS pull-ups in various modes.

Table 9-12 Status of MOS Pull-Ups for Port 5

Mode	Reset	Hardware Standby Mode	Other Operating States*
1	OFF	OFF	ON/OFF
2			
3			OFF
4			ON/OFF
7			

* Including software standby mode.

Notation:

OFF: The MOS pull-up is always off.

ON/OFF: The MOS pull-up is on when P6DDR = 0 and P6DR = 1, and off otherwise.

Note: When $P6_1$, $P6_2$, and $P6_3$ are used for PWM timer output, their MOS pull-ups are switched off regardless of the values in P6DDR and P6DR.

9.8 Port 7

9.8.1 Overview

Port 7 is an 8-bit input/output port with the pin configuration shown in figure 9-19. Its pins also carry input and output signals for the on-chip free-running timers (FRT1, FRT2, and FRT3), and two input signals for the on-chip 8-bit timer.

Port 7 has Schmitt inputs. Outputs from port 7 can drive one TTL load and a 30 pF capacitive load. They can also drive a Darlington transistor pair.

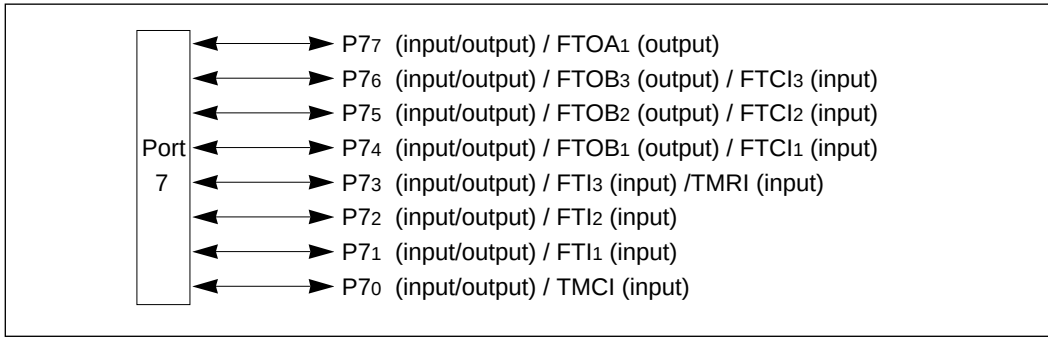


Figure 9-19 Pin Functions of Port 7

9.8.2 Port 7 Registers

Register Configuration: Table 9-13 lists the registers of port 7.

Table 9-13 Port 7 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 7 data direction register	P7DDR	W	H'00	H'FE8C
Port 7 data register	P7DR	R/W	H'00	H'FE8E

1. Port 7 Data Direction Register (P7DDR)—H'FE8C

Bit	7	6	5	4	3	2	1	0
	P7₇DDR	P7₆DDR	P7₅DDR	P7₄DDR	P7₃DDR	P7₂DDR	P7₁DDR	P7₀DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P7DDR is an 8-bit register that selects the direction of each pin in port 7. A pin functions as an output pin if the corresponding bit in P7DDR is set to 1, and as an input pin if the bit is cleared to 0.

P7DDR can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

At a reset and in the hardware standby mode, P7DDR is initialized to H'00, setting all pins for input. P7DDR is not initialized in the software standby mode, so if a P7DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 7 data register.

A transition to the software standby mode initializes the on-chip supporting modules, so any pins of port 7 that were being used by an on-chip timer when the transition occurs revert to general-purpose input or output, controlled by P7DDR and P7DR.

2. Port 7 Data Register (P7DR)—H'FE8E

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	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 is an 8-bit register containing the data for pins $P7_7$ to $P7_0$. When the CPU reads P7DR, for output pins it reads the value in the P7DR latch, but for input pins, it obtains the pin status directly.

9.8.3 Pin Functions

The pin functions of port 7 are the same in all MCU operating modes. As figure 9-19 indicated, these pins are used for input and output of on-chip timer signals as well as for general-purpose input and output. For some pins, two or more functions can be enabled simultaneously.

Table 9-14 shows how the functions of the pins of port 7 are selected.

Table 9-14 Port 7 Pin Functions

Pin	Selection of Pin Functions
$P7_7$ / $FTOA_1$	The function depends on the output enable A bit (OEA) of the FRT1 timer control register (TCR) and on the $P7_7$DDR bit as follows:

OEA	0		1	
$P7_7$DDR	0	1	0	1
Pin function	$P7_7$ input	$P7_7$ output	$FTOA_1$ output	

$P7_6$ / $FTOB_3$ / $FTCI_3$	The function depends on the output compare B bit (OEB) of the FRT3 timer control register (TCR) and on the $P7_6$DDR bit as follows:

OEB	0		1	
$P7_6$DDR	0	1	0	1
Pin function	$P7_6$ input	$P7_6$ output	$FTOB_3$ output	
	$FTCI_3$ input			

$P7_5$ / $FTOB_2$ / $FTCI_2$	The function depends on the output compare B bit (OEB) of the FRT2 timer control register (TCR) and on the $P7_5$DDR bit as follows:

OEB	0		1	
$P7_5$DDR	0	1	0	1
Pin function	$P7_5$ input	$P7_5$ output	$FTOB_2$ output	
	$FTCI_2$ input			

$P7_4$ / $FTOB_1$ / $FTCI_1$	The function depends on the output compare B bit (OEB) of the FRT1 timer control register (TCR) and on the $P7_4$DDR bit as follows:

OEB	0		1	
$P7_4$DDR	0	1	0	1
Pin function	$P7_4$ input	$P7_4$ output	$FTOB_1$ output	
	$FTCI_1$ input			

Table 9-14 Port 7 Pin Functions (cont)

Pin	Selection of Pin Functions
$P7_3$ / FTI_3 / TMRI	The function depends on the counter clear bits 1 and 0 (CCLR1 and CCLR0) in the timer control register (TCR) of the 8-bit timer, and on the $P7_3$DDR bit as follows: The TMRI function is operative when bits CCLR0 and CCLR1 in the timer control register (TCR) of the 8-bit timer are both set to 1.

$P7_3$DDR	0	1
Pin function	$P7_3$ input	$P7_3$ output
	FTI_3 input and TMRI input	

$P7_2$ / FTI_2

$P7_2$DDR	0	1
Pin function	$P7_2$ input	$P7_2$ output
	FTI_2 input	

$P7_1$ / FTI_1

$P7_1$DDR	0	1
Pin function	$P7_1$ input	$P7_1$ output
	FTI_1 input	

$P7_0$ / TMCI — This pin always has a general-purpose input/output function, and can simultaneously be used for external clock input for the 8-bit timer, depending on clock select bits 2 to 0 (CKS2, CKS1, and CKS0) in the timer control register (TCR). See section 11, “8-Bit Timer” for details.

$P7_0$DDR	0	1
Pin function	$P7_0$ input	$P7_0$ output
	TMCI input	

9.9 Port 8

9.9.1 Overview

Port 8 is an 8-bit input port that also receives inputs for the on-chip A/D converter. The pin functions are the same in all MCU operating modes, as shown in figure 9-20.

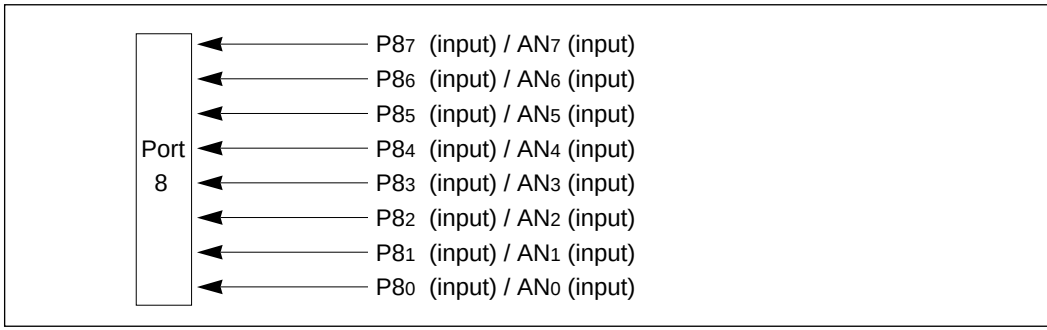


Figure 9-20 Pin Functions of Port 8

9.9.2 Port 8 Registers

Register Configuration: Port 8 has only the data register described in table 9-15. Since it is exclusively an input port, there is no data direction register.

Table 9-15 Port 8 Registers

Name	Abbreviation	Read/Write	Address
Port 8 data register	P8DR	R	H'FE8F

1. Port 8 Data Register (P8DR)—H'FE8F

Bit	7	6	5	4	3	2	1	0
	$P8_7$	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

When the CPU reads P8DR it always reads the current status of each pin, except that during A/D conversion, the pin being used for analog input reads 1 regardless of the input voltage at that pin.

9.10 Port 9

9.10.1 Overview

Port 9 is an 8-bit input/output port with the pin configuration shown in figure 9-21. In addition to general-purpose input and output, its pins are used for the output compare A signals from free-running timers 2 and 3, for PWM timer output, and for input and output by the on-chip serial communication interfaces (SCI1 and SCI2). The pin functions are the same in all MCU operating modes.

Outputs from port 9 can drive one TTL load and a 30 pF capacitive load. They can also drive a Darlington transistor pair.

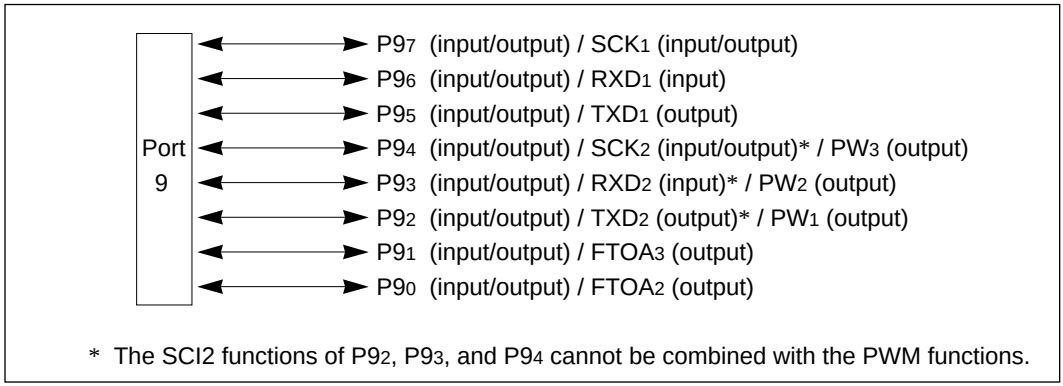


Figure 9-21 Pin Functions of Port 9

9.10.2 Port 9 Registers

Register Configuration: Table 9-16 lists the registers of port 9.

Table 9-16 Port 9 Registers

Name	Abbreviation	Read/Write	Initial Value	Address
Port 9 data direction register	P9DDR	W	H'00	H'FEFE
Port 9 data register	P9DR	R/W	H'00	H'FEFF

1. Port 9 Data Direction Register (P9DDR)—H'FEFE

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
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 an input pin if the bit is cleared to 0.

P9DDR can be written but not read. An attempt to read this register does not cause an error, but all bits are read as 1, regardless of their true values.

At a reset and in the hardware standby mode, P9DDR is initialized to H'00, setting all pins for input. P9DDR is not initialized in the software standby mode, so if a P9DDR bit is set to 1 when the chip enters the software standby mode, the corresponding pin continues to output the value in the port 9 data register.

A transition to the software standby mode initializes the on-chip supporting modules, so any pins of port 9 that were being used by an on-chip module (example: free-running timer output) when the transition occurs revert to general-purpose input or output, controlled by P9DDR and P9DR.

2. Port 9 Data Register (P9DR)—H'FEFF

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 it reads the value in the P9DR latch, but for input pins, it obtains the pin status directly.

9.10.3 Pin Functions

The pin functions of port 9 are the same in all MCU operating modes. As figure 9-21 indicated, these pins are used for output of on-chip timer signals and for input and output of serial data and clock signals as well as for general-purpose input and output. Specifically, they carry output signals for free-running timers 2 and 3, pulse-width modulation (PWM) timer output signals, and input and output signals for the serial communication interfaces.

Table 9-17 shows how the functions of the pins of port 9 are selected.

Table 9-17 Port 9 Pin Functions

Pin	Selection of Pin Functions
$P9_7$ / SCK_1	The function depends on the communication mode bit (C/$\overline{A}$) in the SCI1 serial mode register (SMR) and the clock enable 1 and 0 bits (CKE1 and CKE0) in the SCI1 serial control register (SCR).

C/$\overline{A}$	0				1			
CKE1	0		1		0		1	
CKE0	0	1	0	1	0	1	0	1
Pin function	$P9_7$ input/ output	SCI1 internal clock output	SCI1 external clock input		SCI1 internal clock output		SCI1 external clock input	

When used for $P9_7$ input/output, the input or output function is selected by $P9_7$DDR.

$P9_6$ / RXD_1	The function depends on the receive enable bit (RE) in the SCI1 serial control register (SCR) and on the $P9_6$DDR bit as follows.

RE	0		1	
$P9_6$DDR	0	1	0	1
Pin function	$P9_6$ input	$P9_6$ output	RXD_1 input	

$P9_5$ / TXD_1	The function depends on the transmit enable bit (TE) in SCI1's SCR and on the $P9_6$DDR bit as follows.

TE	0		1	
$P9_5$DDR	0	1	0	1
Pin function	$P9_5$ input	$P9_5$ output	TXD_1 output	

Table 9-17 Port 9 Pin Functions (cont)

Pin	Selection of Pin Functions
$P9_4$ / SCK_2 / PW_3	The function depends on the output enable bit (OE) of PWM timer 3's timer control register (TCR), the C/$\overline{A}$ bit in SCI2's SMR, the CKE1 and CKE0 bits in SCI2's SCR, and the port 9 PWM enable bit (P9PWME) and port 9 serial enable bit (P9SCI2E) in system control register 2 (SYSCR2).

P9SCI2E	1						0		1	0
P9PWME	0						1		1	0
OE	0/1						0	1	0/1	
C/$\overline{A}$	0				1		0/1	0/1	0/1	
CKE1	0		1		0	1	0/1	0/1	0/1	
CKE0	0	1	0	1	0	1	0/1	0/1	0/1	
Pin function	$P9_4$ input/ output	SCI2 internal clock output	SCI2 external clock input	SCI2 internal clock output	SCI2 external clock input		$P9_4$ input/ output	PW_3 output	$P9_4$ input/ output	

When used for $P9_4$ input/output, the input or output function is selected by $P9_4$DDR.

$P9_3$ / RXD_2 / PW_2	The function depends on the OE bit in PWM timer 2's TCR, the RE bit in SCI2's SCR, and the P9PWME bit and P9SCI2E bit in SYSCR2.

P9SCI2E	1				0			0	1
P9PWME	0				1			0	1
OE	0	1	0	1	0	1		0/1	0/1
RE	0		1		0	1	0	1	0/1	0/1
Pin function	$P9_3$ input/ output		RXD_2 input		$P9_3$ input/ output	PW_2 output		$P9_3$ input/ output	

When used for $P9_3$ input/output, the input or output function is selected by $P9_3$DDR.

Table 9-17 Port 9 Pin Functions (cont)

Pin	Selection of Pin Functions
$P9_2$ / TXD_2 / PW_1	The function depends on the OE bit in PWM timer 1's TCR, the TE bit in SCI2's SCR, and the P9PWME bit and P9SCI2E bit in SYSCR2.

P9SCI2E	1				0				0	1
P9PWME	0				1				0	1
OE	0	1	0	1	0		1		0/1	0/1
TE	0		1		0	1	0	1	0/1	0/1
Pin function	$P9_2$ input/ output		TXD_2 output		$P9_2$ input/ output		PW_1 output		$P9_2$ input/ output	

When used for $P9_2$ input/output, the input or output function is selected by $P9_2$DDR.

$P9_1$ / $FTOA_3$	The function depends on the output enable A bit (OEA) in FRT3's TCR and on the $P9_1$DDR bit as follows.

OEA	0		1	
$P9_1$DDR	0	1	0	1
Pin function	$P9_1$ input	$P9_1$ output	$FTOA_3$ output	

$P9_0$ / $FTOA_2$	The function depends on the output enable A bit (OEA) in FRT2's TCR and on the $P9_0$DDR bit as follows.

OEA	0		1	
$P9_0$DDR	0	1	0	1
Pin function	$P9_0$ input	$P9_0$ output	$FTOA_2$ output	

Section 10 16-Bit Free-Running Timers

10.1 Overview

The H8/534 and H8/536 have an on-chip 16-bit free-running timer (FRT) module with three independent channels (FRT1, FRT2, and FRT3). All three channels are functionally identical.

Each channel has a 16-bit free-running counter that it uses as a time base. Applications of the FRT module include rectangular-wave output (up to two independent waveforms per channel), input pulse width measurement, and measurement of external clock periods.

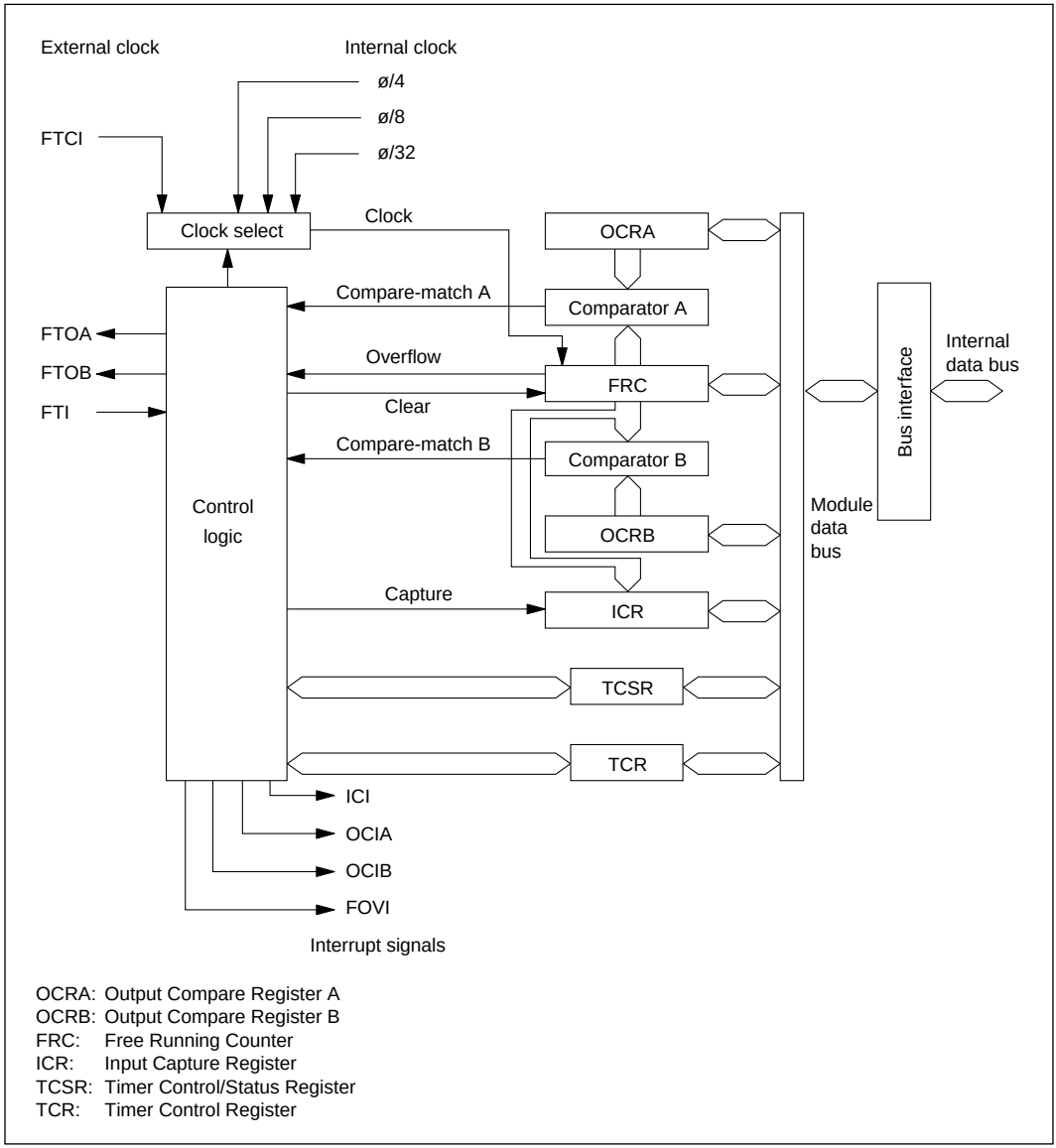
10.1.1 Features

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

- Selection of four clock sources
  The free-running counters can be driven by an internal clock source (ø/4, ø/8, or ø/32), or an external clock input (enabling use as an external event counter).
- Two independent comparators
  Each free-running timer channel can generate two independent waveforms.
- Input capture function
  The current count can be captured on the rising or falling edge (selectable) of an input signal.
- Four types of interrupts
  Compare-match A and B, input capture, and overflow interrupts can be requested independently.
  The compare-match and input capture interrupts can be served by the data transfer controller (DTC), enabling interrupt-driven data transfer with minimal CPU programming.
- Counter can be cleared under program control
  The free-running counters can be cleared on compare-match A.

10.1.2 Block Diagram

Figure 10-1 shows a block diagram of one free-running timer channel.



OCRA: Output Compare Register A
OCRB: Output Compare Register B
FRC: Free Running Counter
ICR: Input Capture Register
TCSR: Timer Control/Status Register
TCR: Timer Control Register

Figure 10-1 Block Diagram of 16-Bit Free-Running Timer

10.1.3 Input and Output Pins

Table 10-1 lists the input and output pins of the free-running timer module.

Table 10-1 Input and Output Pins of Free-Running Timer Module

Channel	Name	Abbreviation	I/O	Function
1	Output compare A	$FTOA_1$	Output	Output controlled by comparator A of FRT1
	Output compare B or counter clock input	$FTOB_1$ / $FTCI_1$	Output / Input	Output controlled by comparator B of FRT1, or input of external clock source for FRT1
	Input capture	FTI_1	Input	Trigger for capturing current count of FRT1
2	Output compare A	$FTOA_2$	Output	Output controlled by comparator A of FRT2
	Output compare B or counter clock input	$FTOB_2$ / $FTCI_2$	Output / Input	Output controlled by comparator B of FRT2, or input of external clock source for FRT2
	Input capture	FTI_2	Input	Trigger for capturing current count of FRT2
3	Output compare A	$FTOA_3$	Output	Output controlled by comparator A of FRT3
	Output compare B or counter clock input	$FTOB_3$ / $FTCI_3$	Output / Input	Output controlled by comparator B of FRT3, or input of external clock source for FRT3
	Input capture	FTI_3	Input	Trigger for capturing current count of FRT3

10.1.4 Register Configuration

Table 10-2 lists the registers of each free-running timer channel.

Table 10-2 Register Configuration

Channel	Name	Abbreviation	R/W	Initial Value	Address
	Timer control register	TCR	R/W	H'00	H'FE90
	Timer control/status register	TCSR	R/(W)*	H'00	H'FE91
	Free-running counter (High)	FRC (H)	R/W	H'00	H'FE92
	Free-running counter (Low)	FRC (L)	R/W	H'00	H'FE93
1	Output compare register A (High)	OCRA (H)	R/W	H'FF	H'FE94
	Output compare register A (Low)	OCRA (L)	R/W	H'FF	H'FE95
	Output compare register B (High)	OCRB (H)	R/W	H'FF	H'FE96
	Output compare register B (Low)	OCRB (L)	R/W	H'FF	H'FE97
	Input capture register (High)	ICR (H)	R	H'00	H'FE98
	Input capture register (Low)	ICR (L)	R	H'00	H'FE99
	Timer control register	TCR	R/W	H'00	H'FEA0
	Timer control/status register	TCSR	R/(W)*	H'00	H'FEA1
	Free-running counter (High)	FRC (H)	R/W	H'00	H'FEA2
	Free-running counter (Low)	FRC (L)	R/W	H'00	H'FEA3
2	Output compare register A (High)	OCRA (H)	R/W	H'FF	H'FEA4
	Output compare register A (Low)	OCRA (L)	R/W	H'FF	H'FEA5
	Output compare register B (High)	OCRB (H)	R/W	H'FF	H'FEA6
	Output compare register B (Low)	OCRB (L)	R/W	H'FF	H'FEA7
	Input capture register (High)	ICR (H)	R	H'00	H'FEA8
	Input capture register (Low)	ICR (L)	R	H'00	H'FEA9

* Software can write a 0 to clear bits 7 to 4, but cannot write a 1 in these bits.

Table 10-2 Register Configuration (cont)

Channel	Name	Abbreviation	R/W	Initial Value	Address
	Timer control register	TCR	R/W	H'00	H'FEB0
	Timer control/status register	TCSR	R/(W)*	H'00	H'FEB1
	Free-running counter (High)	FRC (H)	R/W	H'00	H'FEB2
	Free-running counter (Low)	FRC (L)	R/W	H'00	H'FEB3
3	Output compare register A (High)	OCRA (H)	R/W	H'FF	H'FEB4
	Output compare register A (Low)	OCRA (L)	R/W	H'FF	H'FEB5
	Output compare register B (High)	OCRB (H)	R/W	H'FF	H'FEB6
	Output compare register B (Low)	OCRB (L)	R/W	H'FF	H'FEB7
	Input capture register (High)	ICR (H)	R	H'00	H'FEB8
	Input capture register (Low)	ICR (L)	R	H'00	H'FEB9

* Software can write a 0 to clear bits 7 to 4, but cannot write a 1 in these bits.

10.2 Register Descriptions

10.2.1 Free-Running Counter (FRC)—H'FE92, H'FEA2, H'FEB2

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

Each 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).

The FRC can be cleared by compare-match A.

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 10.3, “CPU Interface” for details.

The FRCs are initialized to H'0000 at a reset and in the standby modes.

10.2.2 Output Compare Registers A and B (OCRA and OCRB)—H'FE94 and H'FE96, H'FEA4 and H'FEA6, H'FEB4 and H'FEB6

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 control register (TCR) 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 timer control status register (TCSR) is output at the output compare pin (FTOA or FTOB).

The FTOA and FTOB output are 0 before the first compare-match.

Because OCRA and OCRB are 16-bit registers, a temporary register (TEMP) is used when they are written. See section 10.3, “CPU Interface” for details.

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

10.2.3 Input Capture Register (ICR)—H'FE98, H'FEA8, H'FEB8

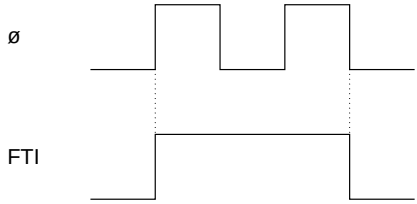
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The ICR is a 16-bit read-only register.

When the rising or falling edge of the signal at the input capture input pin is detected, the current value of the FRC is copied to the ICR. At the same time, the input capture flag (ICF) in the timer control/status register (TCSR) is set to 1. The input capture edge is selected by the input edge select bit (IEDG) in the TCSR.

Because the ICR is a 16-bit register, a temporary register (TEMP) is used when the ICR is written or read. See section 10.3, “CPU Interface” for details.

To ensure input capture, the pulse width of the input capture signal should be at least 1.5 system clock periods (1.5·ø).



Minimum FTI Pulse Width

The ICR is 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 ICR even if the input capture flag (ICF) is already set.

10.2.4 Timer Control Register (TCR)

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	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 FRC clock source, enables the output compare signals, and enables interrupts.

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

Bit 7—Input Capture Interrupt Enable (ICIE): This bit selects whether to request an input capture interrupt (ICI) when the input capture flag (ICF) in the timer status/control register (TCSR) is set to 1.

Bit 7

ICIE	Description
0	The input capture interrupt request (ICI) is disabled. (Initial value)
1	The input capture interrupt request (ICI) is enabled.

Bit 6—Output Compare Interrupt Enable B (OCIEB): 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 6

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

Bit 5—Output Compare Interrupt Enable A (OCIEA): 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 5

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

Bit 4—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 4

OVIE	Description
0	The free-running timer overflow interrupt request (FOVI) is disabled. (Initial value)
1	The free-running timer overflow interrupt request (FOVI) is enabled.

Bit 3—Output Enable B (OEB): This bit selects whether to enable or disable output of the logic level selected by the OLVLB bit in the timer status/control register (TCSR) at the output compare B pin when the FRC and OCRB values match.

Bit 3

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

Bit 2—Output Enable A (OEA): This bit selects whether to enable or disable output of the logic level selected by the OLVLA bit in the timer status/control register (TCSR) at the output compare A pin when the FRC and OCRA values match.

Bit 2 OEA	Description	
0	Output compare A output is disabled.	(Initial value)
1	Output compare A output is enabled.	

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	Internal clock source (ø/4)	(Initial value)
0	1	Internal clock source (ø/8)	
1	0	Internal clock source (ø/32)	
1	1	External clock source (counted on the rising edge)	

10.2.5 Timer Control/Status Register (TCSR)

Bit	7	6	5	4	3	2	1	0
	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	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 that selects the input capture edge and output compare levels, and specifies whether to clear the counter on compare-match A. It also contains four status flags.

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

* Software can write a 0 in bits 7 to 4 to clear the flags, but cannot write a 1 in these bits.

Bit 7—Input Capture Flag (ICF): This status flag is set to 1 to indicate an input capture event. It signifies that the FRC value has been copied to the ICR.

Bit 7 ICF	Description
0	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the ICF bit after it has been set to 1, then writes a 0 in this bit. 2. The data transfer controller (DTC) serves an input capture interrupt .
1	This bit is set to 1 when an input capture signal causes the FRC value to be copied to the ICR.

Bit 6—Output Compare Flag B (OCFB): This status flag is set to 1 when the FRC value matches the OCRB value.

Bit 6 OCFB	Description
0	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the OCFB bit after it has been set to 1, then writes a 0 in this bit. 2. The data transfer controller (DTC) serves output compare interrupt B.
1	This bit is set to 1 when FRC = OCRB.

Bit 5—Output Compare Flag A (OCFA): This status flag is set to 1 when the FRC value matches the OCRA value.

Bit 5 OCFA	Description
0	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the OCFA bit after it has been set to 1, then writes a 0 in this bit. 2. The data transfer controller (DTC) serves output compare interrupt A.
1	This bit is set to 1 when FRC = OCRA.

Bit 4—Timer Overflow Flag (OVF): This status flag is set to 1 when the FRC overflows (changes from H'FFFF to H'0000).

Bit 4 OVF	Description
0	This bit is cleared from 1 to 0 when the CPU reads the OVF bit after it has been set to 1, then writes a 0 in this bit. (Initial value)
1	This bit is set to 1 when FRC changes from H'FFFF to H'0000.

Bit 3—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 3 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.	

Bit 2—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 2 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 1—Input Edge Select (IEDG): This bit selects whether to capture the count on the rising or falling edge of the input capture signal.

Bit 1 IEDG	Description	
0	The FRC value is copied to the ICR on the falling edge of the input capture signal.	(Initial value)
1	The FRC value is copied to the ICR on the rising edge of the input capture signal.	

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.

10.3 CPU Interface

The FRC, OCRA, OCRB, and ICR are 16-bit registers, but they are connected to an 8-bit data bus. When the CPU accesses these four 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 upper 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.

Programs that access these four registers should normally use word access. Equivalently, they may access first the upper byte, then the lower byte. Data will not be transferred correctly if the bytes are accessed in reverse order, or if only one byte is accessed.

Coding Examples : Write the contents of R0 into OCRA in FRT1

```
MOV.W R0, @H'FE94
```

: Read ICR of FRT2

```
MOV.W, @H'FEA8, R0
```

The same considerations apply to access by the DTC.

Figure 10-2 shows the data flow when the FRC is accessed. The other registers are accessed in the same way, except that when OCRA or OCRB is read, the upper and lower bytes are both transferred directly to the CPU without using the temporary register.

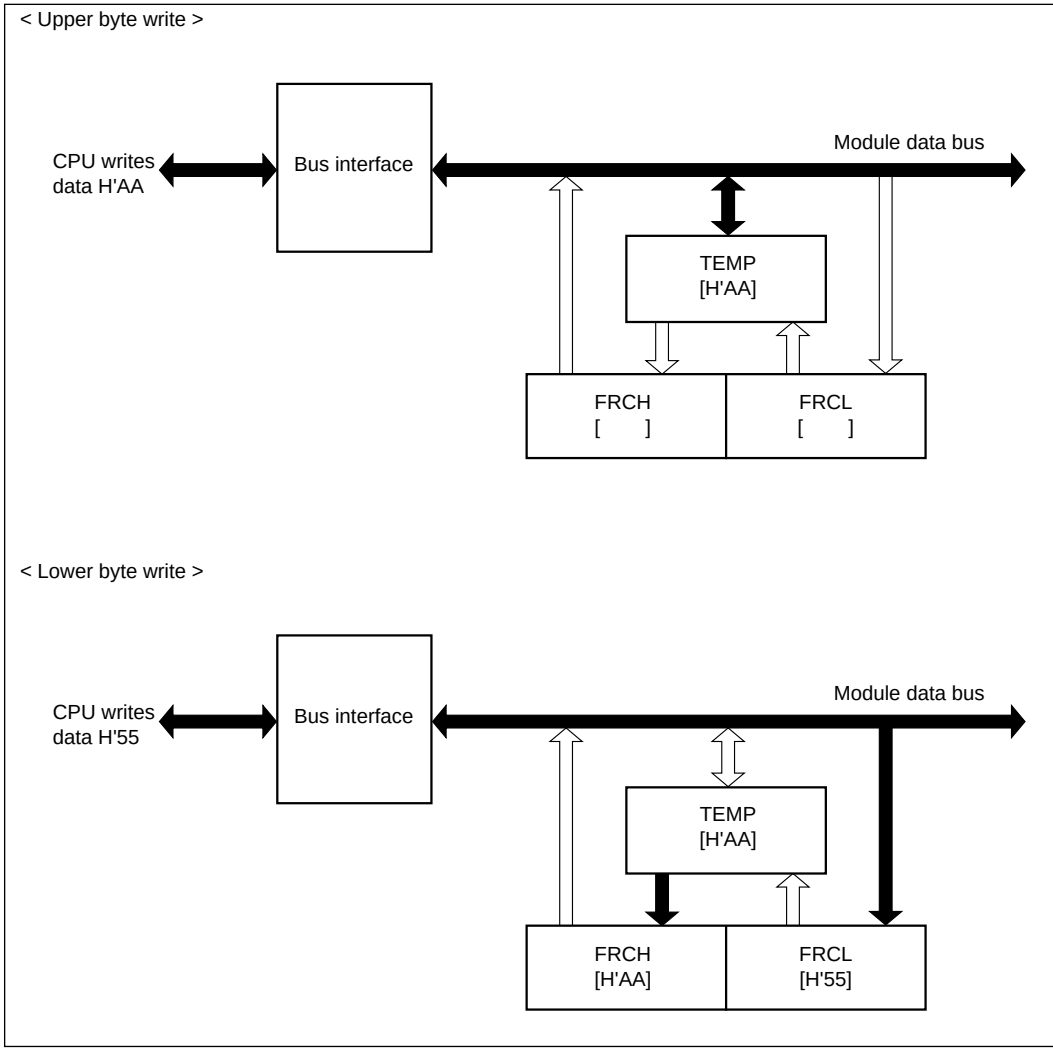


Figure 10-2 (a) Write Access to FRC (When CPU Writes H'AA55)

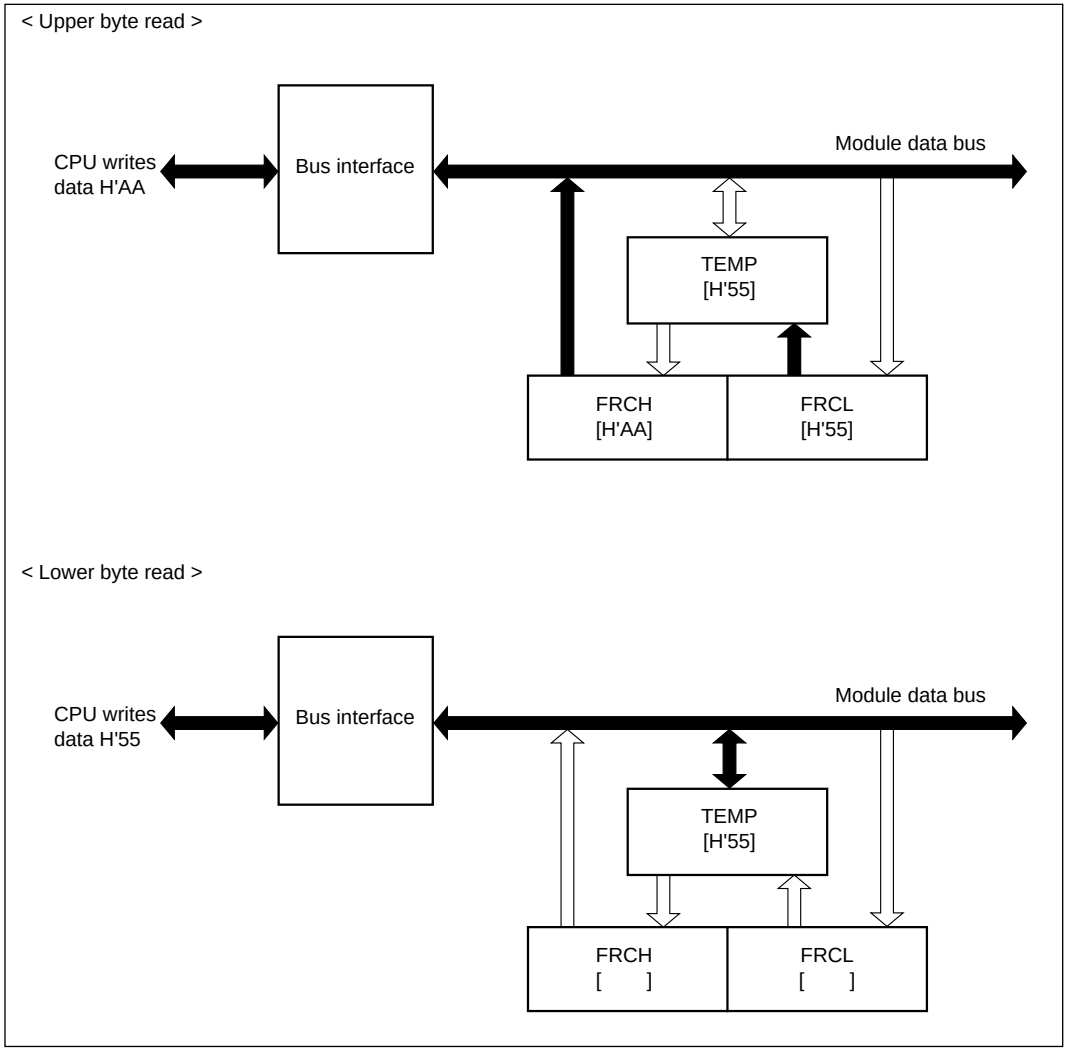


Figure 10-2 (b) Read Access to FRC (When FRC Contains H'AA55)

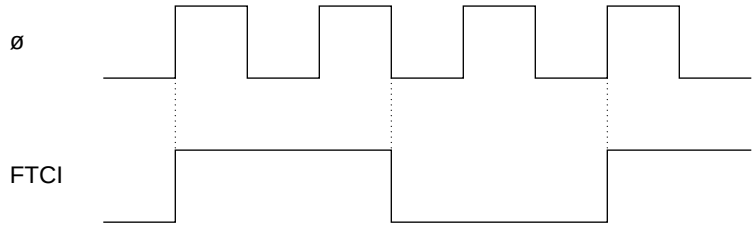
10.4 Operation

10.4.1 FRC Incrementation Timing

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

If external clock input is selected, the FRC increments on the rising edge of the clock signal. Figure 10-3 shows the increment timing.

The pulse width of the external clock signal must be at least 1.5·ø clock periods. The counter will not increment correctly if the pulse width is shorter than 1.5·ø clock periods.



Minimum FTCI Pulse Width

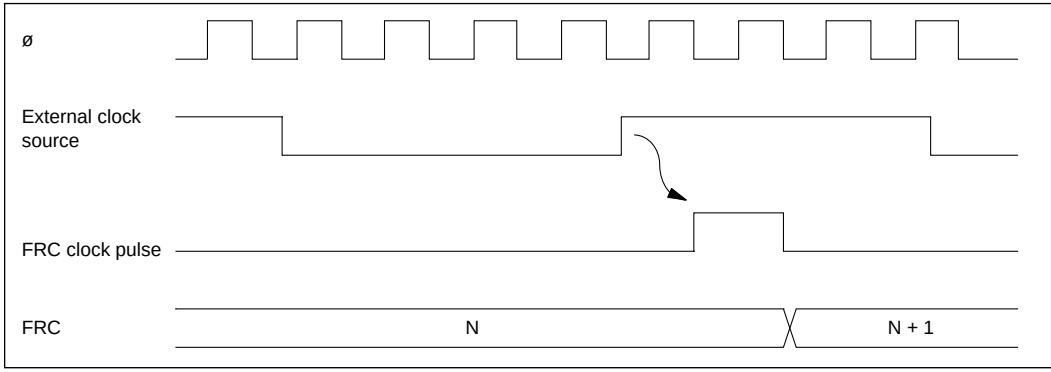


Figure 10-3 Increment Timing for External Clock Input

10.4.2 Output Compare Timing

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 10-4 shows the timing of the setting of the output compare flags.

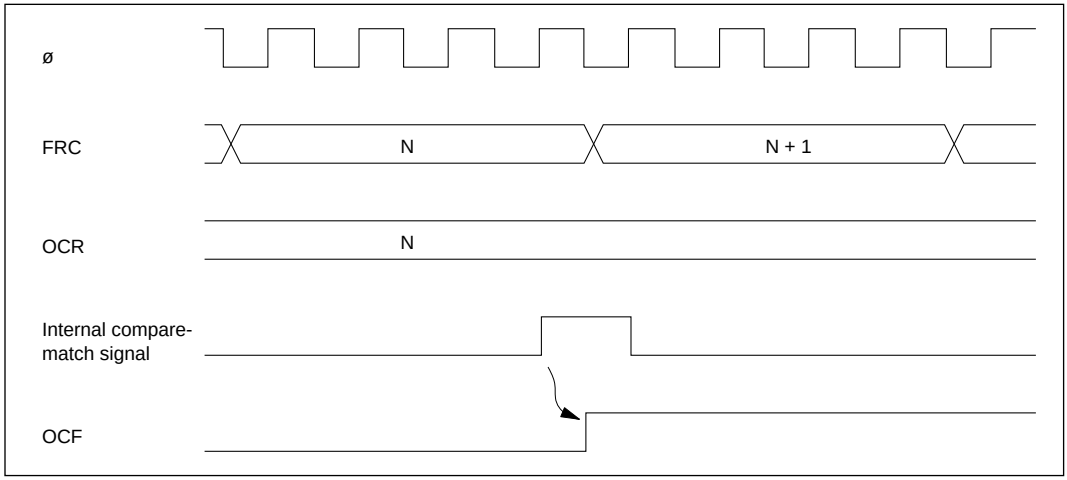


Figure 10-4 Setting of Output Compare Flags

Output Timing: When a compare-match occurs, the logic level selected by the output level bit (OLVLA or OLVLB) in the TCSR is output at the output compare pin (FTOA or FTOB). Figure 10-5 shows the timing of this operation for compare-match A.

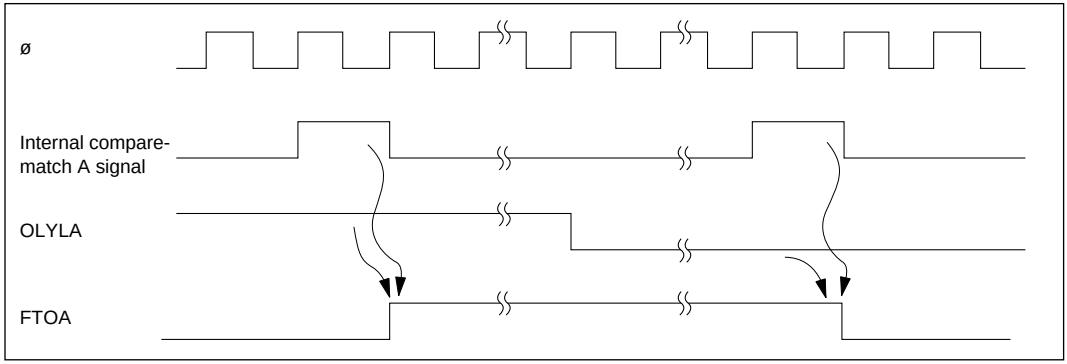


Figure 10-5 Timing of Output Compare A

FRC Clear Timing: If the CCLRA bit is set to 1, the FRC is cleared when compare-match A occurs. Figure 10-6 shows the timing of this operation.

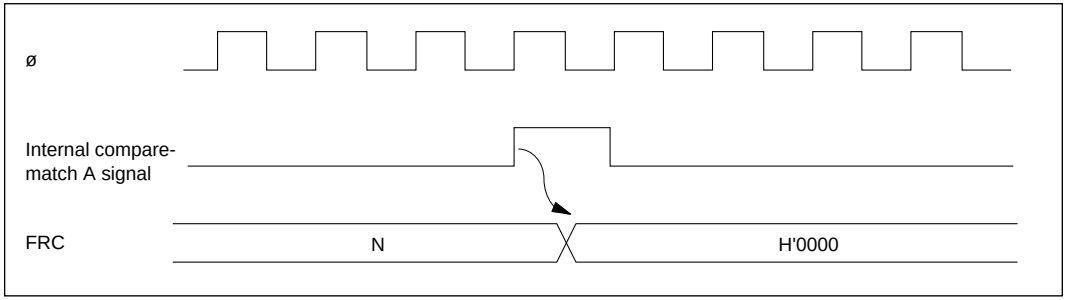


Figure 10-6 Clearing of FRC by Compare-Match A

10.4.3 Input Capture Timing

1. **Input Capture Timing:** An internal input capture signal is generated from the rising or falling edge of the input at the input capture pin (FTI), as selected by the IEDG bit in the TCSR. Figure 10-7 shows the usual input capture timing when the rising edge is selected (IEDG = 1).

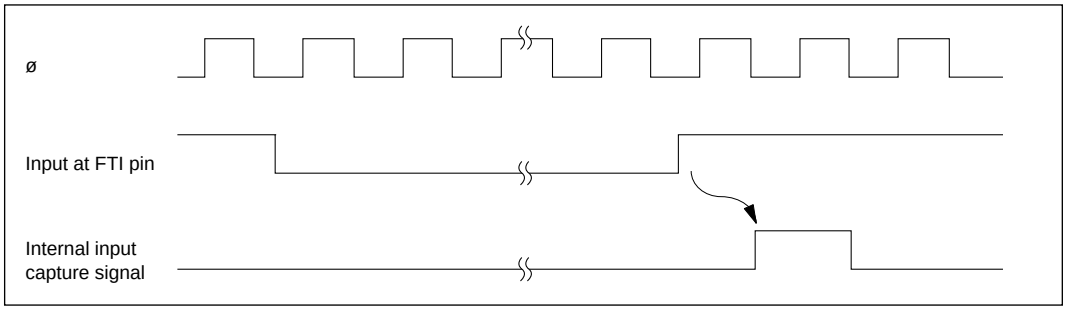


Figure 10-7 Input Capture Timing (Usual Case)

But if the upper byte of the ICR is being read when the input capture signal arrives, the internal input capture signal is delayed by one state. Figure 10-8 shows the timing for this case.

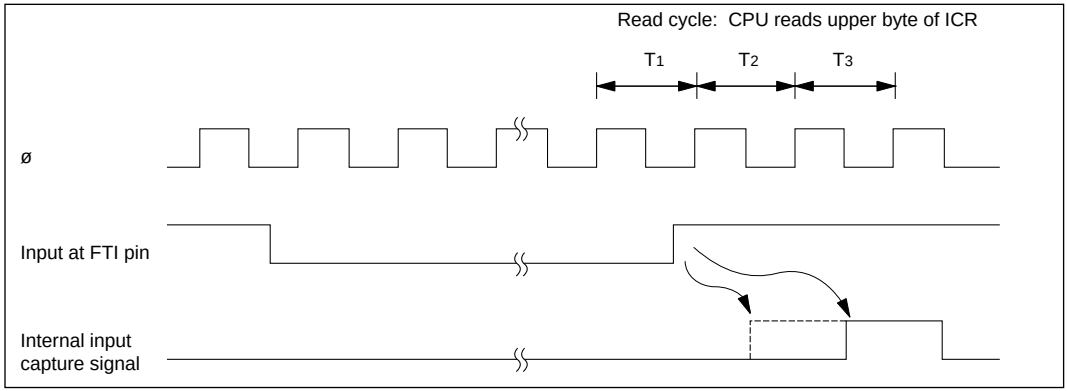


Figure 10-8 Input Capture Timing (1-State Delay)

Timing of Input Capture Flag (ICF) Setting: The input capture flag (ICF) is set to 1 by the internal input capture signal. Figure 10-9 shows the timing of this operation.

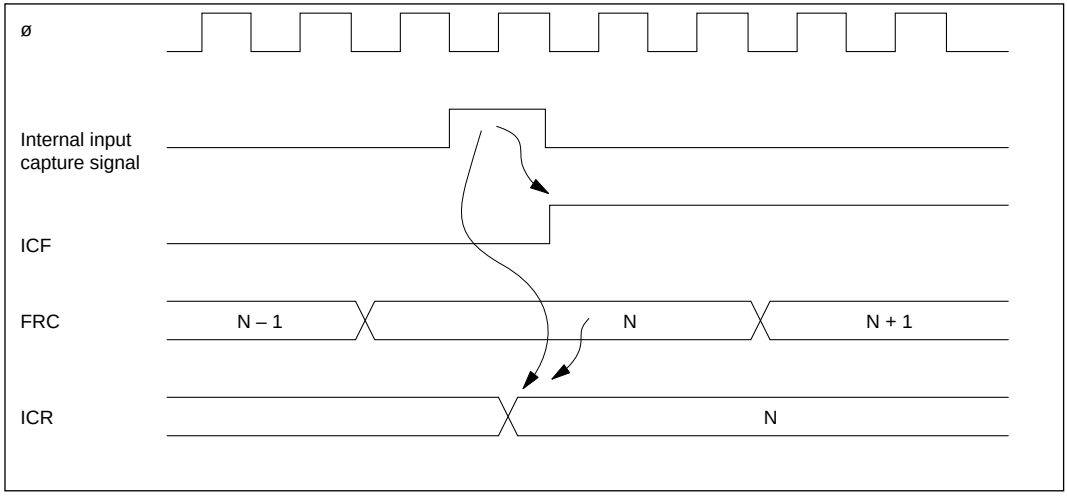


Figure 10-9 Setting of Input Capture Flag

10.4.4 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 10-10 shows the timing of this operation.

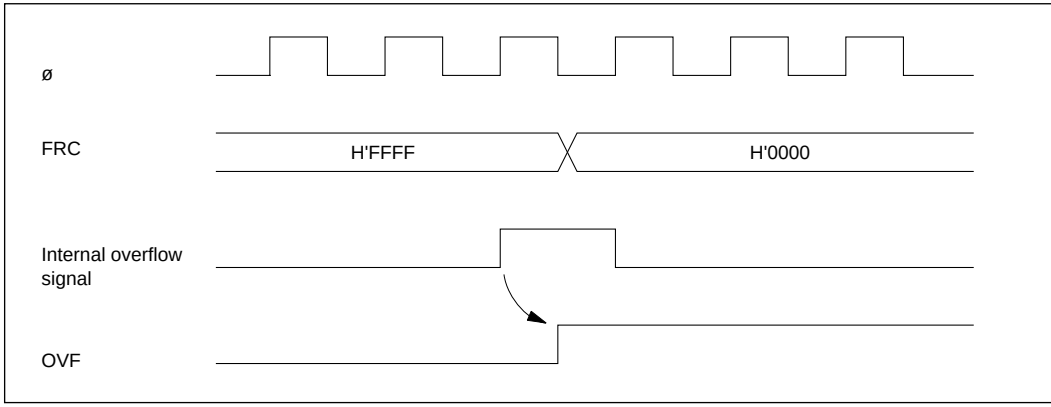


Figure 10-10 Setting of Overflow Flag (OVF)

10.5 CPU Interrupts and DTC Interrupts

Each free-running timer channel can request four types of interrupts: input capture (ICI), 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 10-3 lists information about these interrupts.

Table 10-3 Free-Running Timer Interrupts

Interrupt	Description	DTC Service Available?	Priority
ICI	Requested when ICF is set	Yes	High
OCIA	Requested when OCFA is set	Yes	↑
OCIB	Requested when OCFB is set	Yes	
FOVI	Requested when OVF is set	No	Low

The ICI, OCIA, and OCIB interrupts can be directed to the data transfer controller (DTC) to have a data transfer performed in place of the usual interrupt-handling routine.

When the DTC serves one of these interrupts, it automatically clears the ICF, OCFA, or OCFB flag to 0. See section 6, “Data Transfer Controller” for further information on the DTC.

10.6 Synchronization of Free-Running Timers 1 to 3

10.6.1 Synchronization after a Reset

The three free-running timer channels are synchronized at a reset and remained synchronized until:

- the clock source is changed;
- FRC contents are rewritten; or
- an FRC is cleared.

After a reset, each free-running counter operates on the ø/4 internal clock source.

10.6.2 Synchronization by Writing to FRCs

When synchronization among free-running timers 1 to 3 is lost, it can be restored by writing to the free-running counters.

Synchronization on Internal Clock Source: When an internal clock is selected, free-running timers 1 to 3 can be synchronized by writing data to their free-running counters as indicated in table 10-4.

Table 10-4 Synchronization by Writing to FRCs

Clock Source	Write Interval	Write Data	
ø/4	4n (states)	m	(FRC1)
ø/8	8n (states)	m + n	(FRC2)
ø/32	32n (states)	m + 2n	(FRC3)

m, n: Arbitrary integers

After writing these data, synchronization can be checked by reading the three free-running counters at the same interval as the write interval. If the read data have the same relative differences as the write data, the three free-running timers are synchronized.

Programs for synchronizing the timers are shown next. Examples a, b, and c can be used when the program is stored in on-chip memory. Examples d, e, and f can be used when the program is stored in external memory. These programs assume that no wait states (T_W) are inserted and there is no NMI input.

Example a: ø/4 clock source, 12-state write interval (n = 3), on-chip memory

```
LA:   LDC.B #H'FE,BR        ; Initialize base register for short-format instruction (MOV:S)
      LDC.W #H'0700,SR      ; Raise interrupt mask level to 7
      MOV.W #m,R1           ; Data for free-running timer 1
      MOV.W #m+3,R2         ; Data for free-running timer 2 (m + n = m + 3)
      MOV.W #m+6,R3         ; Data for free-running timer 3 (m + 2n = m + 2 × 3)
      BSR   SET4            ; Call write routine
            ⋮
      .ALIGN 2              ; Align write instructions (MOV:S) at even address
SET4:MOV:S.W R1,@H'92:8     ; Write to FRC 1 (address H'FE92)   9 states
      BRN SET4:8            ; 2-Byte dummy instruction          3 states
      MOV:S.W R2,@H'A2:8    ; Write to FRC 2 (address H'FEA2)        Total 12 states
      BRN SET4:8            ; 2-Byte dummy instruction
      MOV:S.W R3,@H'B2:8    ; Write to FRC 3 (address H'FEB2)
      RTS
```

Example b: ø/8 clock source, 16-state write interval (n = 2), on-chip memory

```
LB:   LDC.B #H'FE,BR
      LDC.W #H'0700,SR
      MOV.W #m,R1
      MOV.W #m+2,R2
      MOV.W #m+4,R3
      BSR   SET8
            ⋮
      .ALIGN 2
SET8:MOV:S.W R1,@H'92:8     ; 9 States
      BRN SET8:8            ; 3 States      Total 16 states
      XCH R1,R1             ; 4 States
      MOV:S.W R2,@H'A2:8
      BRN SET8:8
      XCH R2,R2
      MOV:S.W R3,@H'B2:8
      RTS
```

Example c: ø/32 clock source, 32-state write interval (n = 1), on-chip memory

```
LC:     LDC.B #H'FE,BR
        LDC.W #H'0700,SR
        MOV.W #m,R1
        MOV.W #m+1,R2
        MOV.W #m+2,R3
        BSR   SET32
              :
        .ALIGN 2                        ; Align on even address
SET32:  MOV:S.W R1,@H'92:8              ; 2 Bytes, 9 states
        BSR WAIT:8                      ; 2 Bytes, 9 states
        MOV:S.W R2,@H'A2:8
        BSR WAIT:8
        MOV:S.W R3,@H'B2:8
        RTS

        .ALIGN 2                        ; Align on even address
WAIT:   NOP                             ; 2 States
        XCH  R1,R1                      ; 4 States
        RTS                             ; 8 States
```

Total 32 states

Note: The stack is assumed to be in on-chip RAM.

Example d: ø/4 clock source, 20-state write interval (n = 5), external memory

```
LD:     LDC.B  #H'FE,BR
        LDC.W  #H'0700,SR               ; Set interrupt mask level to 7
        CLR.B  H'FF10                   ; Disable wait states
        MOV.W  #m,R1
        MOV.W  #m+5,R2
        MOV.W  #m+10,R3
        MOV:S.W R1,@H'92:8              ; 13 States
        BRN    LD:8                     ; 2 Bytes, 7 states
        MOV:S.W R2,@H'A2:8
        BRN    LD:8
        MOV:S.W R3,@H'B2:8
```

Total 20 states

Example e: ø/8 clock source, 24-state write interval (n = 3), external memory

```
LE:   LDC.B #H'FF,BR
      LDC.W #H'0700,SR
      CLR.B @H'F8"8
      MOV.W #m,R1
      MOV.W #m+3,R2
      MOV.W #m+6,R3
      MOV:S.W R1,@H'92:8      ; 13 States
      BRN   LE:8              ; 2 Bytes,          7 states
      NOP                     ; 1 Byte,           4 states
      MOV:S.W R2,@H'A2:8
      BRN   LE:8
      NOP
      MOV:S.W R3,@H'B2:8
```

Total 24 states

Example f: ø/32 clock source, 32-state write interval (n = 1), external memory

```
LF:   LDC.B #H'FF,BR
      LDC.W #H'0700,SR
      CLR.B @H'F8:8
      MOV.W #m,R1
      MOV.W #m+1,R2
      MOV.W #m+2,R3
      MOV:S.W R1,@H'92:8      ; External memory, so 13 states
      XCH   R0,R0             ;                   8 states
      BRN   LF:8              ; 2 Bytes,          7 states
      NOP                     ;                   4 states
      MOV:S.W R2,@H'A2:8
      XCH   R0,R0
      BRN   LF:8
      NOP
      MOV:S.W R3,@H'B2:8
```

Total 32 states

Synchronization on External Clock Source: When the external clock source is selected, the free-running timers can be synchronized by halting their external clock inputs, then writing identical values in their free-running counters.

10.7 Sample Application

In the example below, one 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 the TCSR.

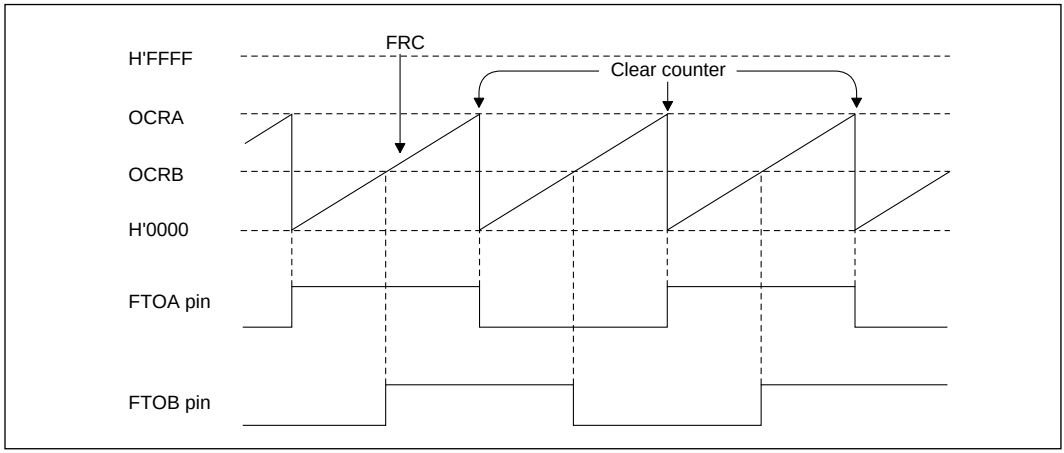


Figure 10-11 Square-Wave Output (Example)

10.8 Application Notes

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

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 a free-running counter, the clear signal takes priority and the write is not performed.

Figure 10-12 shows this type of contention.

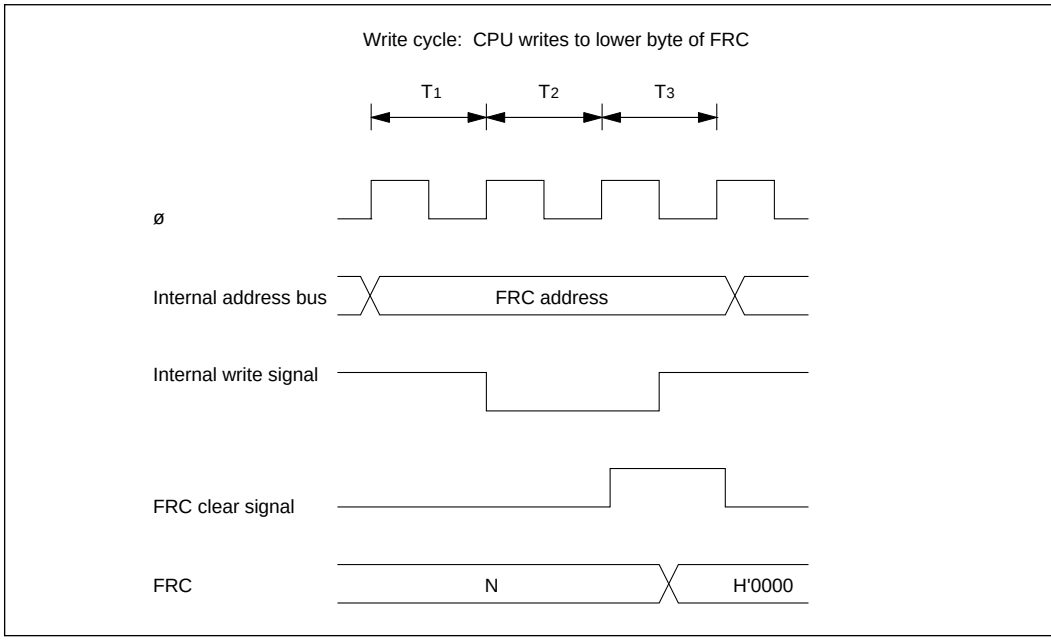


Figure 10-12 FRC Write-Clear Contention

Contention between FRC Write and Increment: If an FRC increment pulse is generated during the T3 state of a write cycle to the lower byte of a free-running counter, the write takes priority and the FRC is not incremented.

Figure 10-13 shows this type of contention.

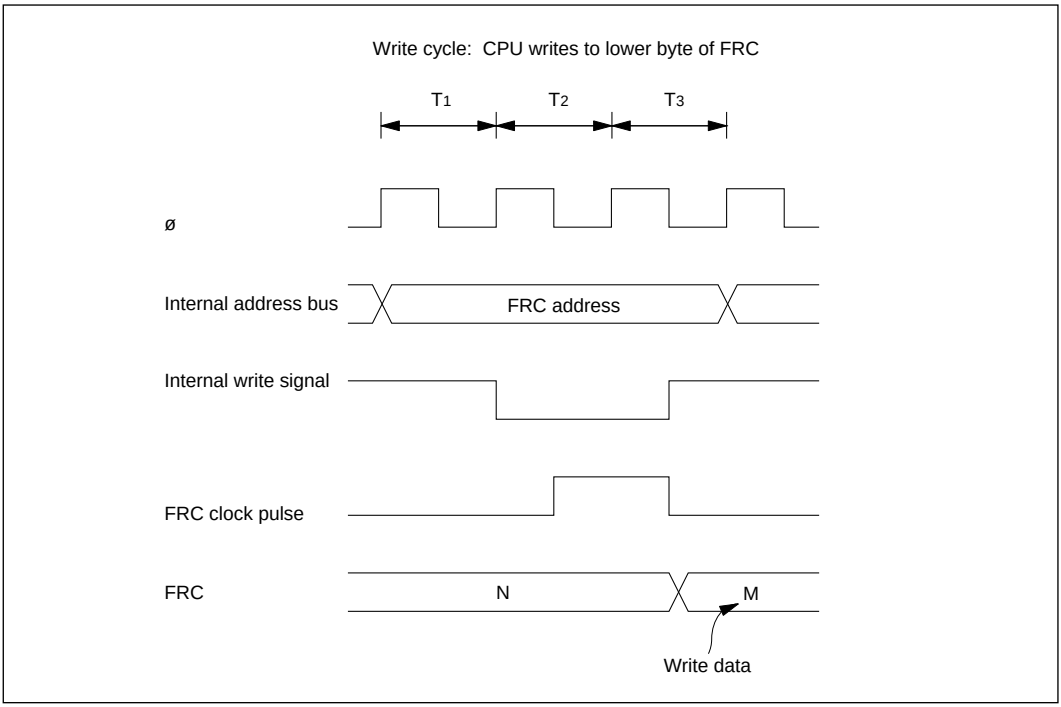


Figure 10-13 FRC Write-Increment Contention

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 10-14 shows this type of contention.

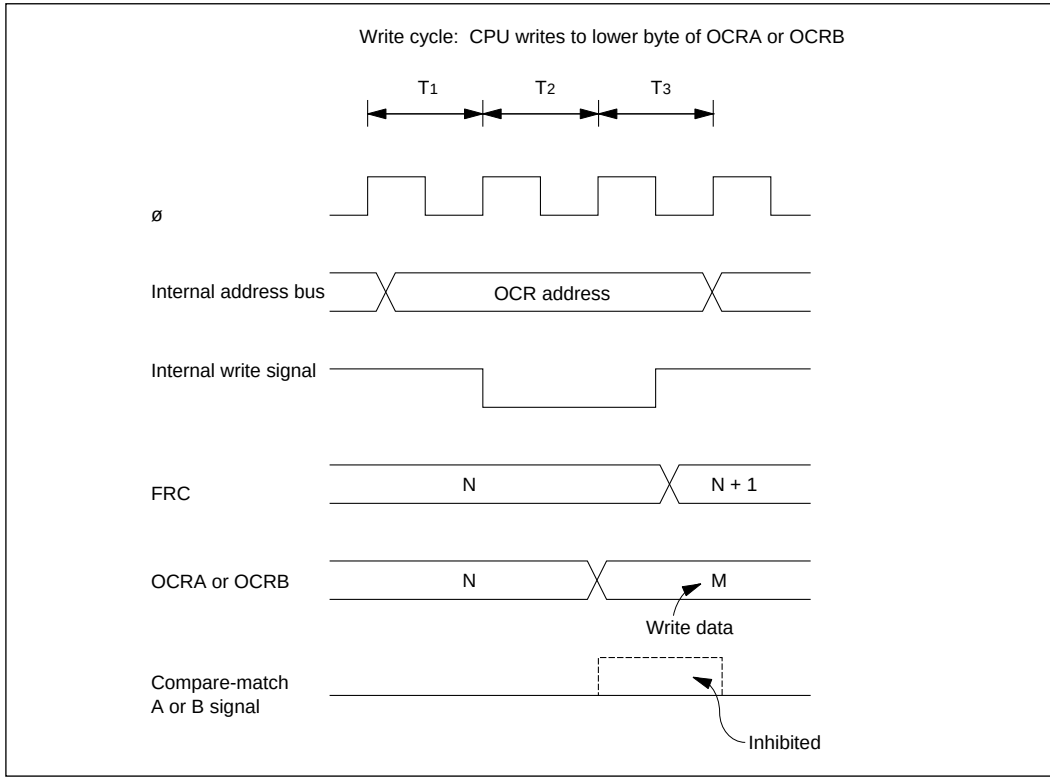


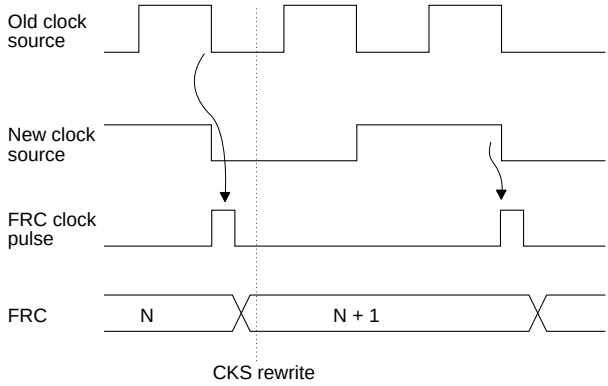
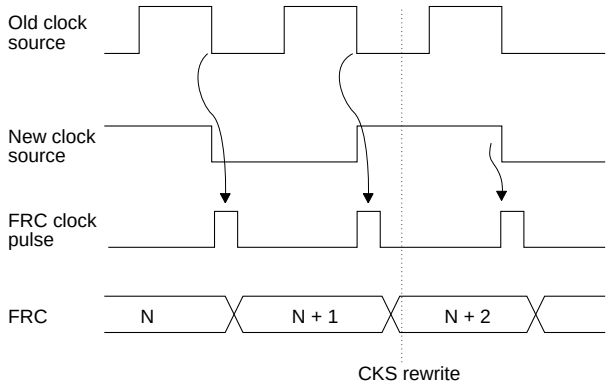
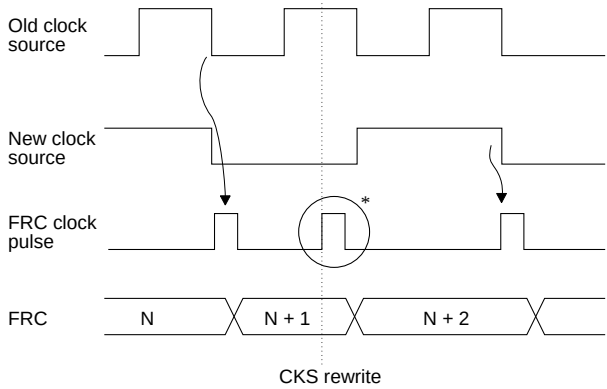
Figure 10-14 Contention between OCR Write and Compare-Match

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 10-5.

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 10-5, the changeover generates a falling edge that triggers the FRC increment pulse.

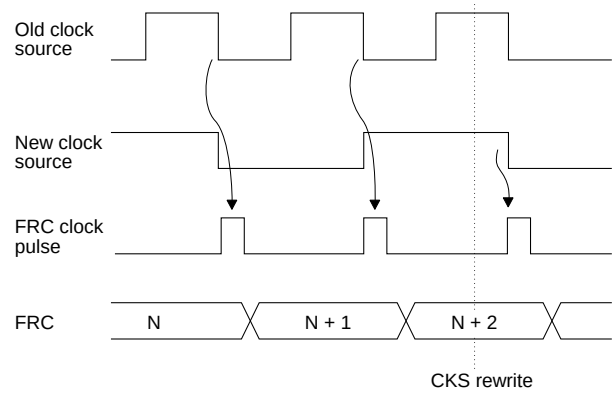
Switching between an internal and external clock source can also cause the FRC to increment.

Table 10-5 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 10-5 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 11 8-Bit Timer

11.1 Overview

The H8/534 and H8/536 have a single 8-bit timer based on an 8-bit counter (TCNT). The timer has 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 is to generate a rectangular-wave output with an arbitrary duty factor.

11.1.1 Features

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

- Selection of four clock sources
  The counter 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 counter
  The counter can be cleared on compare-match A or B, or by an external reset signal.
- Timer output controlled by two time constants
  The single timer output (TMO) is controlled by two independent time constants, enabling the timer to generate output waveforms with an arbitrary duty factor.
- Three types of interrupts
  Compare-match A and B and overflow interrupts can be requested independently.
  The compare match interrupts can be served by the data transfer controller (DTC), enabling interrupt-driven data transfer with minimal CPU programming.

11.1.2 Block Diagram

Figure 11-1 shows a block diagram of 8-bit timer.

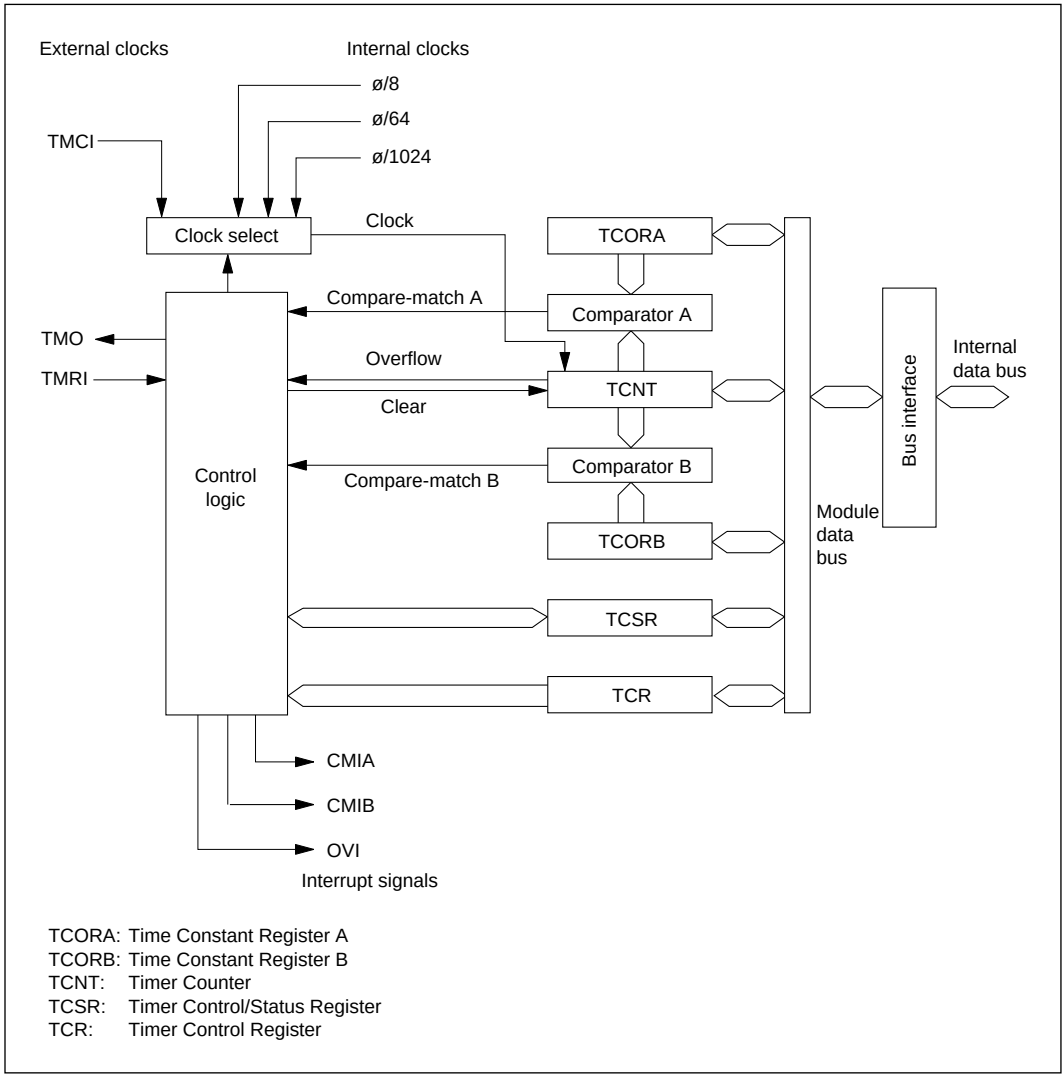


Figure 11-1 Block Diagram of 8-Bit Timer

11.1.3 Input and Output Pins

Table 11-1 lists the input and output pins of the 8-bit timer.

Table 11-1 Input and Output Pins of 8-Bit Timer

Name	Abbreviation	I/O	Function
Timer output	TMO	Output	Output controlled by compare-match
Timer clock input	TMCI	Input	External clock source for the counter
Timer reset input	TMRI	Input	External reset signal for the counter

11.1.4 Register Configuration

Table 11-2 lists the registers of the 8-bit timer.

Table 11-2 8-Bit Timer Registers

Name	Abbreviation	R/W	Initial Value	Address
Timer control register	TCR	R/W	H'00	H'FED0
Timer control/status register	TCSR	R/(W)*	H'10	H'FED1
Timer constant register A	TCORA	R/W	H'FF	H'FED2
Timer constant register B	TCORB	R/W	H'FF	H'FED3
Timer counter	TCNT	R/W	H'00	H'FED4

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

11.2 Register Descriptions

11.2.1 Timer Counter (TCNT)—H'FED4

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

11.2.2 Time Constant Registers A and B (TCORA and TCORB)—H'FED2 and H'FED3

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 (TMO) is controlled by these compare-match signals as specified by output select bits 1 to 0 (OS1 to OS0) in the timer status/control register (TCSR).

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

11.2.3 Timer Control Register (TCR)—H'FED0

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

The 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 TCR is 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 status/control 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 status/control 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 status/control 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.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description
0	0	0	No clock source (timer stopped). (Initial value)
0	0	1	Internal clock source (ø/8).
0	1	0	Internal clock source (ø/64).
0	1	1	Internal clock source (ø/1024).
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.

11.2.4 Timer Control/Status Register (TCSR)—H'FED1

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

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 (TMO).

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

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

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.

Bit 7 CMFB	Description
0	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the CMFB bit after it has been set to 1, then writes a 0 in this bit. 2. Compare-match interrupt B is served by the data transfer controller (DTC).
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.

Bit 6 CMFA	Description
0	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the CMFA bit after it has been set to 1, then writes a 0 in this bit. 2. Compare-match interrupt A is served by the data transfer controller (DTC).
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).

Bit 5 OVF	Description
0	This bit is cleared from 1 to 0 when the CPU reads the OVF bit after it has been set to 1, then writes 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 cannot be modified and is always read as 1.

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 (TMO). 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.

When all four output select bits are cleared to 0 the TMO signal is not output. The TMO output is 0 before the first compare-match.

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.

11.3 Operation

11.3.1 TCNT Incrementation Timing

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

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.

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

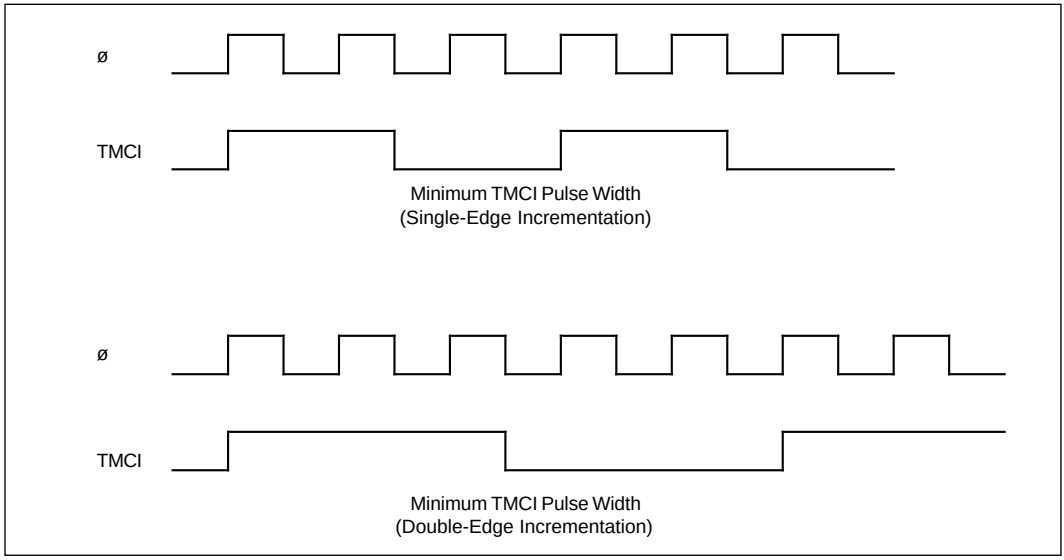


Figure 11-2 shows the count timing for incrementation on both edges.

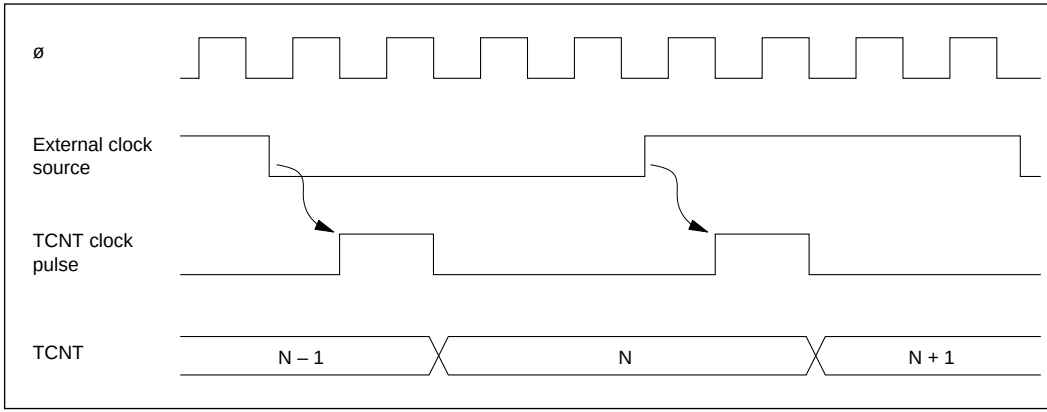


Figure 11-2 Count Timing for External Clock Input

11.3.2 Compare Match Timing

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 TCORA or TCORB. 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 11-3 shows the timing of the setting of the compare-match flags.

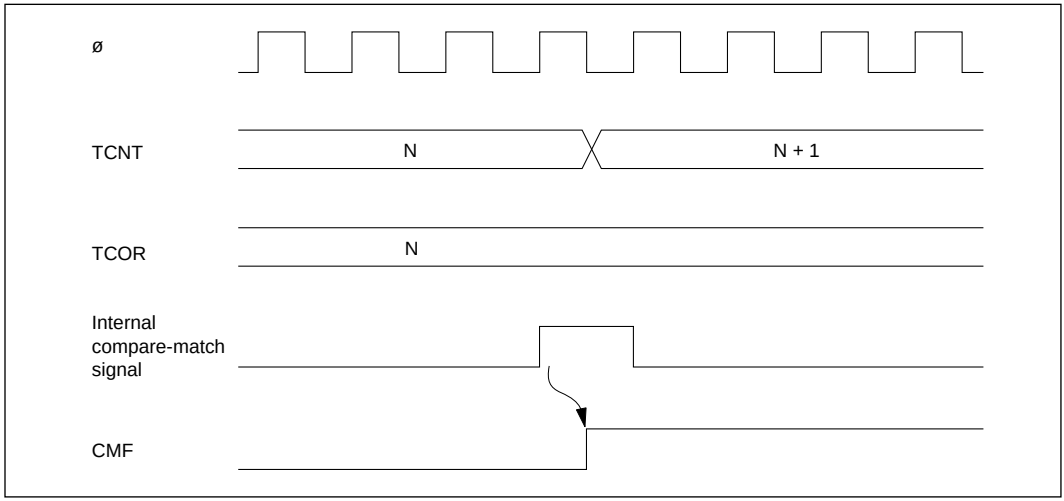


Figure 11-3 Setting of Compare-Match Flags

Output Timing: When a compare-match event occurs, the timer output (TMO) 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.

Figure 11-4 shows the timing when the output is set to toggle on compare-match A.

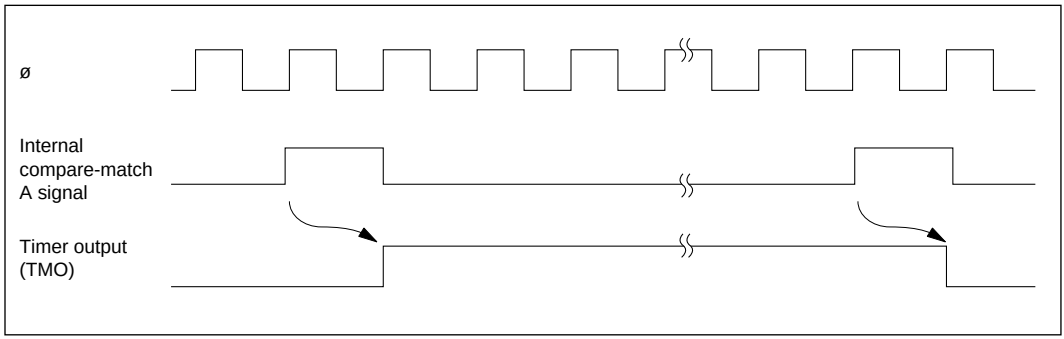


Figure 11-4 Timing of Timer Output

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 11-5 shows the timing of this operation.

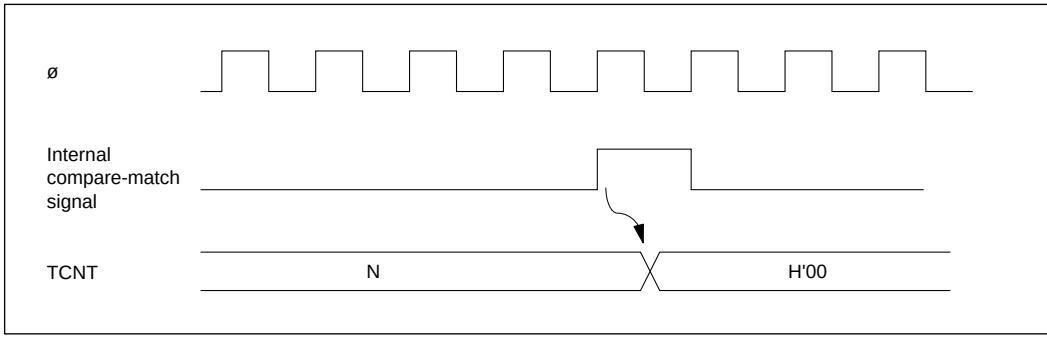


Figure 11-5 Timing of Compare-Match Clear

11.3.3 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 11-6 shows the timing of this operation.

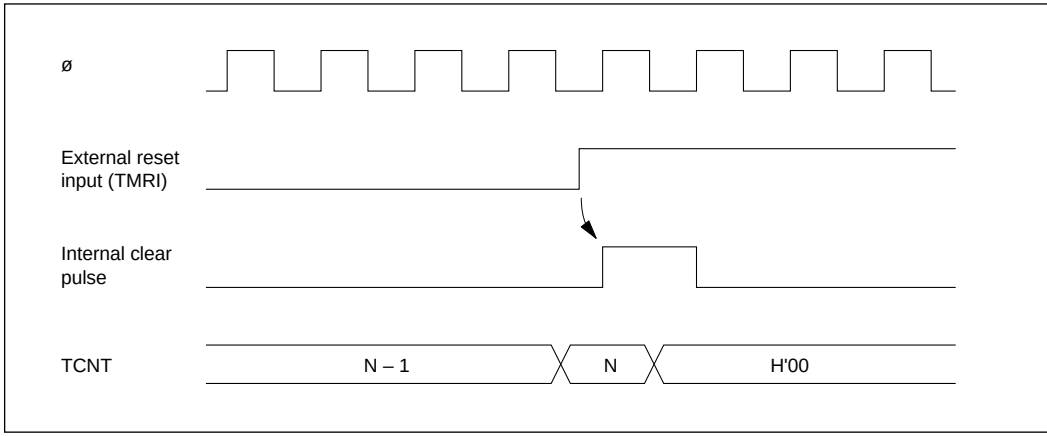


Figure 11-6 Timing of External Reset

11.3.4 Setting of TCNT Overflow Flag

The overflow flag (OVF) is set to 1 when the timer count overflows (changes from H'FF to H'00). Figure 11-7 shows the timing of this operation.

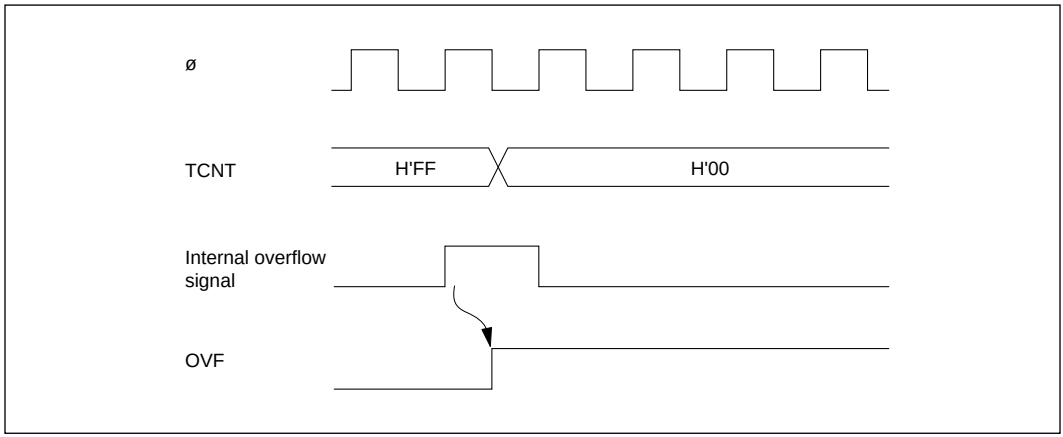


Figure 11-7 Setting of Overflow Flag (OVF)

11.4 CPU Interrupts and DTC Interrupts

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 and flag bits are set in the TCR and TCSR. Independent signals are sent to the interrupt controller for each type of interrupt. Table 11-3 lists information about these interrupts.

Table 11-3 8-Bit Timer Interrupts

Interrupt	Description	DTC Service Available?	Priority
CMIA	Requested when CMFA is set	Yes	High
CMIB	Requested when CMFB is set	Yes	↑
OVI	Requested when OVF is set	No	Low

The CMIA and CMIB interrupts can be served by the data transfer controller (DTC) to have a data transfer performed.

When the DTC serves one of these interrupts, it automatically clears the CMFA or CMFB flag to 0. See section 6, “Data Transfer Controller” for further information on the DTC.

11.5 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.

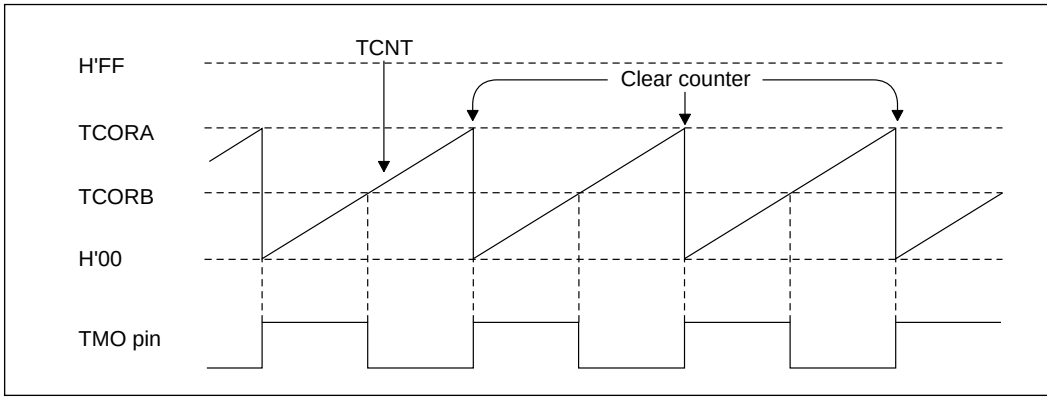


Figure 11-8 Example of Pulse Output

11.6 Application Notes

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

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 11-9 shows this type of contention.

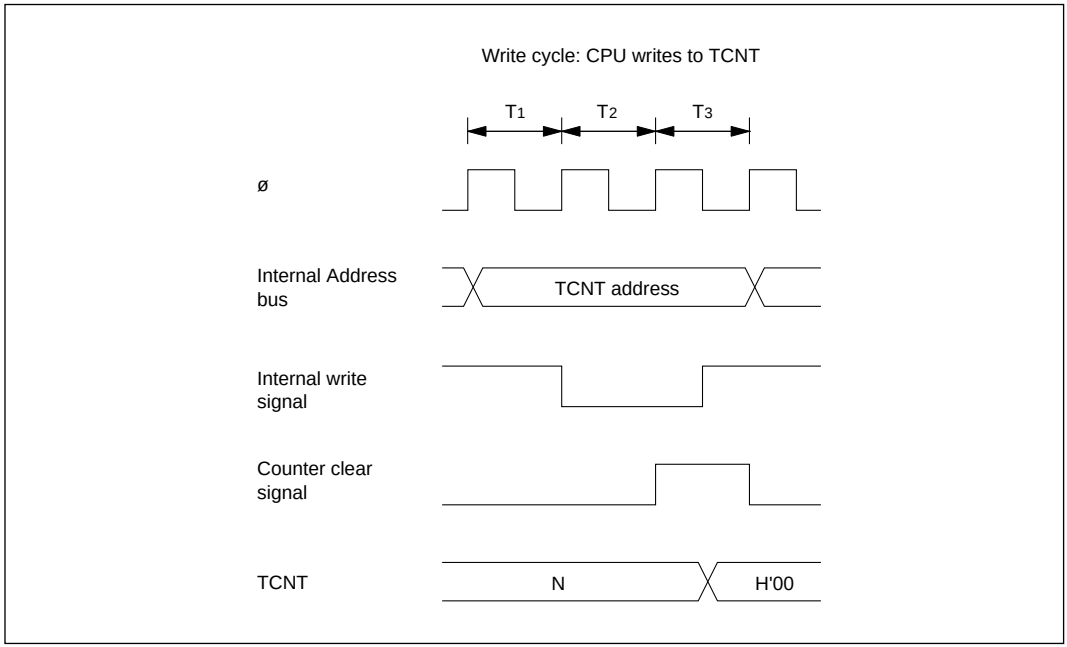


Figure 11-9 TCNT Write-Clear Contention

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 11-10 shows this type of contention.

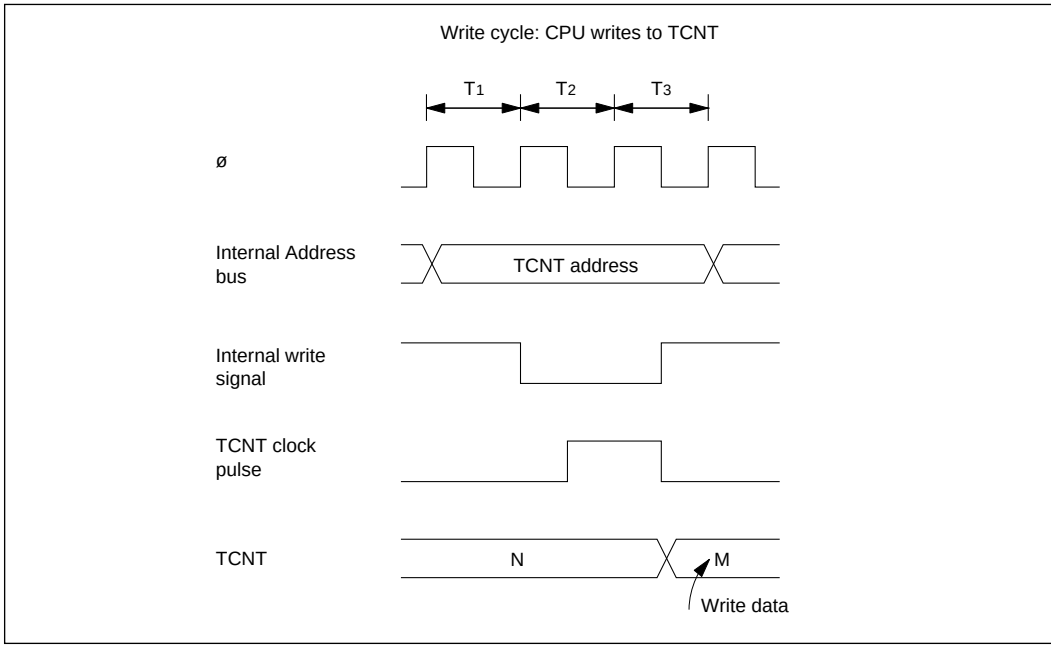


Figure 11-10 TCNT Write-Increment Contention

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 11-11 shows this type of contention.

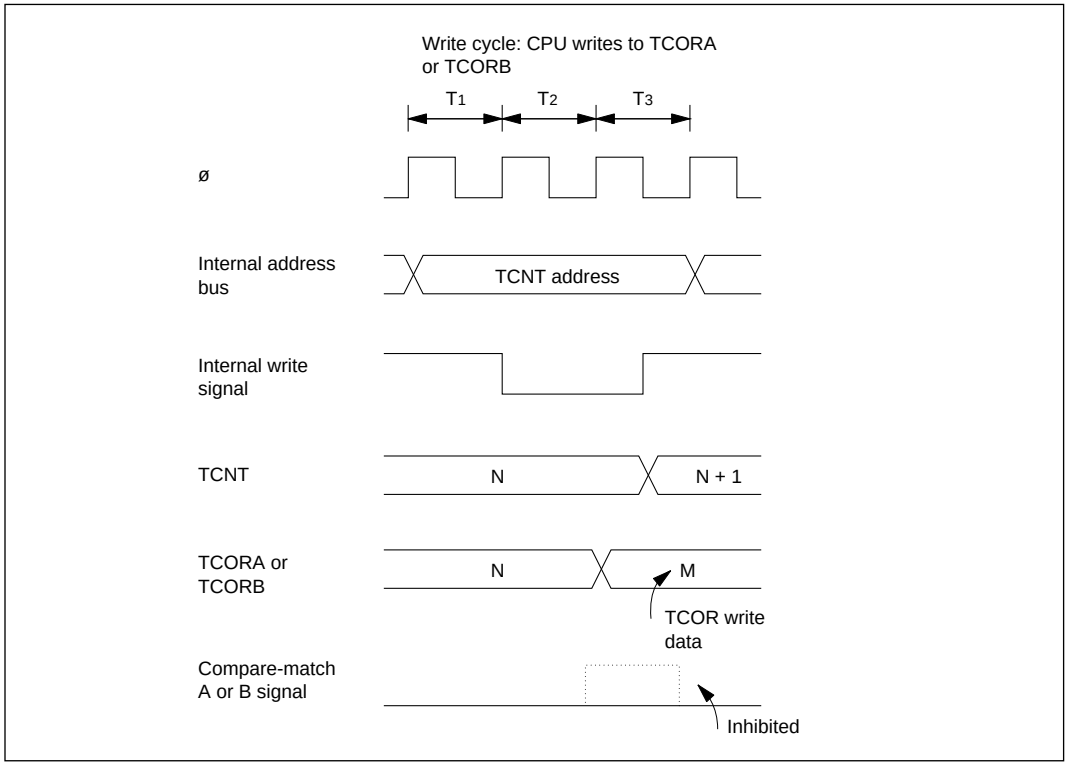


Figure 11-11 Contention between TCOR Write and Compare-Match

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 11-4.

Table 11-4 Priority Order of Timer Output

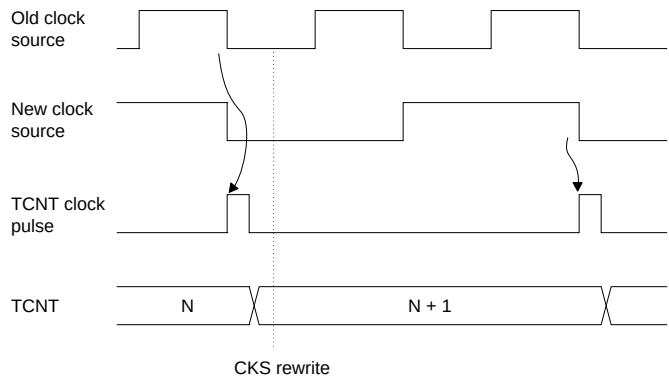
Output Selection	Priority
Toggle	High
1 Output	↑
0 Output	
No change	Low

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 11-5.

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 11-5, the changeover generates a falling edge that triggers the TCNT clock pulse and increments the timer counter.

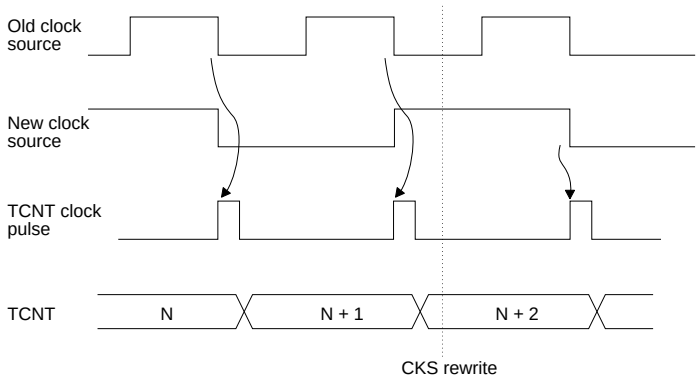
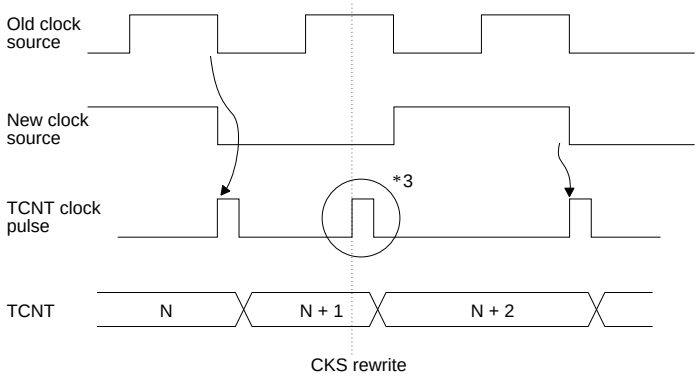
Switching between an internal and external clock source can also cause the timer counter to increment.

Table 11-5 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.	

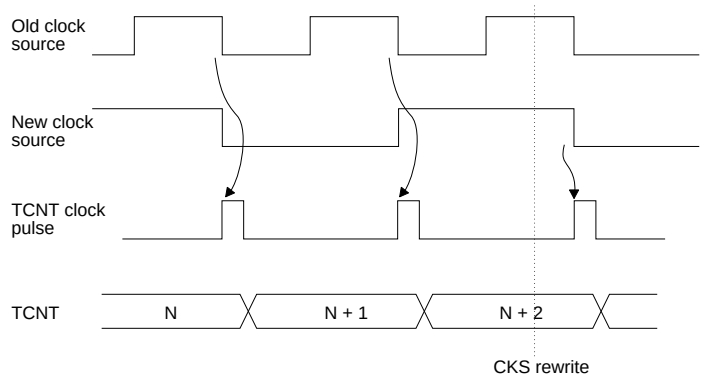
Note: *1 Including a transition from Low to the stopped state (CKS1 = 0, CKS0 = 0), or a transition from the stopped state to Low.

Table 11-5 Effect of Changing Internal Clock Sources (cont)

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

Note: *1 Including a transition from the stopped state to High.
*2 Including a transition from High to the stopped state.
*3 The switching of clock sources is regarded as a falling edge that increments the TCNT.

Table 11-5 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 12 PWM Timer

12.1 Overview

The H8/534 and H8/536 have an on-chip pulse-width modulation (PWM) timer module with three independent channels (PWM1, PWM2, and PWM3). All three channels are functionally identical. Using an 8-bit timer counter, 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).

12.1.1 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

12.1.2 Block Diagram

Figure 12-1 shows a block diagram of one PWM timer channel.

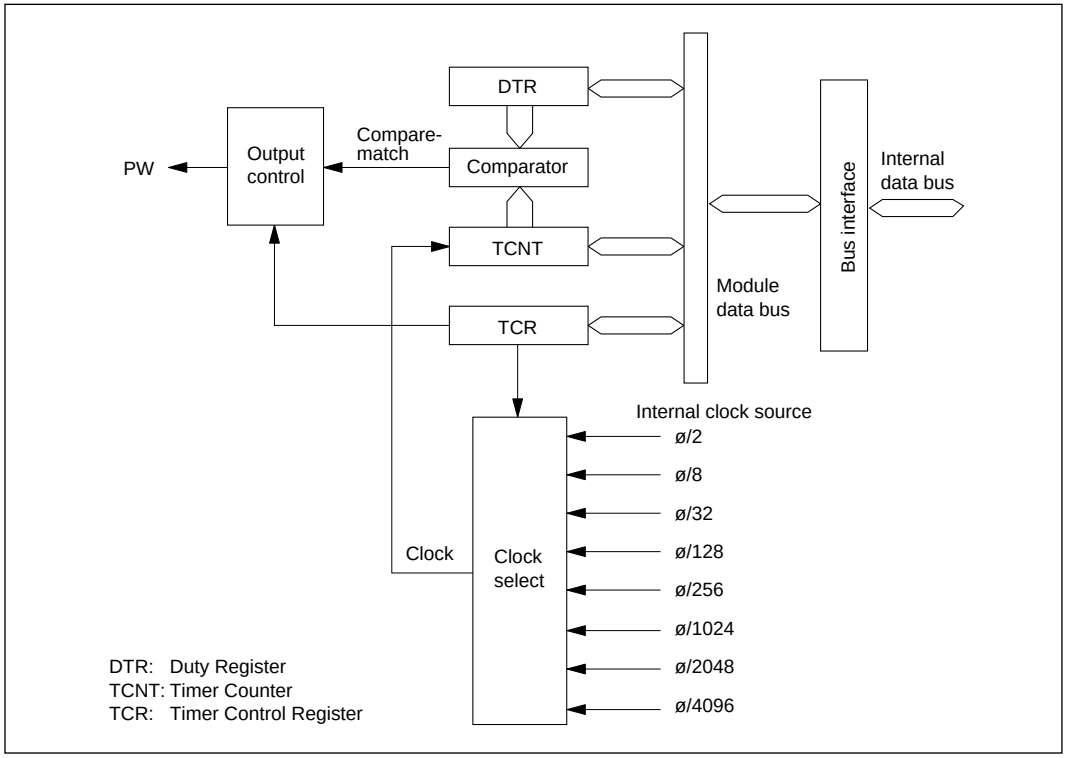


Figure 12-1 Block Diagram of PWM Timer

12.1.3 Input and Output Pins

Table 12-1 lists the output pins of the PWM timer module. There are no input pins.

Table 12-1 Output Pins of PWM Timer Module

Name	Abbreviation	I/O	Function
PWM1 output	PW_1	Output	Pulse output from PWM timer channel 1.
PWM2 output	PW_2	Output	Pulse output from PWM timer channel 2.
PWM3 output	PW_3	Output	Pulse output from PWM timer channel 3.

12.1.4 Register Configuration

The PWM timer module has three registers for each channel as listed in table12-2.

Table 12-2 PWM Timer Registers

Channel	Name	Abbreviation	R/W	Initial Value	Address
1	Timer control register	TCR	R/W	H'38	H'FEC0
	Duty register	DTR	R/W	H'FF	H'FEC1
	Timer counter	TCNT	R/(W)*	H'00	H'FEC2
2	Timer control register	TCR	R/W	H'38	H'FEC4
	Duty register	DTR	R/W	H'FF	H'FEC5
	Timer counter	TCNT	R/(W)*	H'00	H'FEC6
3	Timer control register	TCR	R/W	H'38	H'FEC8
	Duty register	DTR	R/W	H'FF	H'FEC9
	Timer counter	TCNT	R/(W)*	H'00	H'FECA

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

12.2 Register Descriptions

12.2.1 Timer Counter (TCNT)—H'FEC2, H'FEC4, H'FECA

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 PW 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.

12.2.2 Duty Register (DTR)—H'FEC1, H'FEC5, H'FEC9

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) 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 to 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.

12.2.3 Timer Control Register (TCR)—H'FEC0, H'FEC4, H'FEC8

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 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 ø clock frequency is 10 MHz, then the resolution, period, and frequency of the PWM output for each clock source are given in table12-3.

Table 12-3 PWM Timer Parameters for 10 MHz System Clock

Internal Clock Frequency	Resolution	PWM Period	PWM Frequency
ø/2	200 ns	50 µs	20 kHz
ø/8	800 ns	200 µs	5 kHz
ø/32	3.2 µs	800 µs	1.25 kHz
ø/128	12.8 µs	3.2 ms	312.5 Hz
ø/256	25.6 µs	6.4 ms	156.3 Hz
ø/1024	102.4 µs	25.6 ms	39.1 Hz
ø/2048	204.8 µs	51.2 ms	19.5 Hz
ø/4096	409.6 µs	102.4 ms	9.8 Hz

12.3 Operation

Figure 12-2 shows the timing of the PWM timer operation.

1. Positive Logic (OS = 0)

(1) When OE = 0—(a) in Figure 12-2: The timer count is held at H'00 and PWM output is inhibited. (The pin is used for port 9 input/output, and its state depends on the corresponding port 9 data register and data direction register.) Any value (such as N in figure 12-2) written in the DTR becomes valid immediately.

(2) When OE = 1

i) The timer counter begins incrementing, and the PWM output goes High. [(b) in figure 12-2]
ii) When the count reaches the DTR value, the PWM output goes Low. [(c) in figure 12-2]
iii) If the DTR value is changed (by writing the data M in figure 12-2), the new value becomes valid after the timer count changes from H'F9 to H'00. [(d) in figure 12-2]

2. Negative Logic (OS = 1): The operation is the same except that High and Low are reversed in the PWM output. [(e) in figure 12-2]

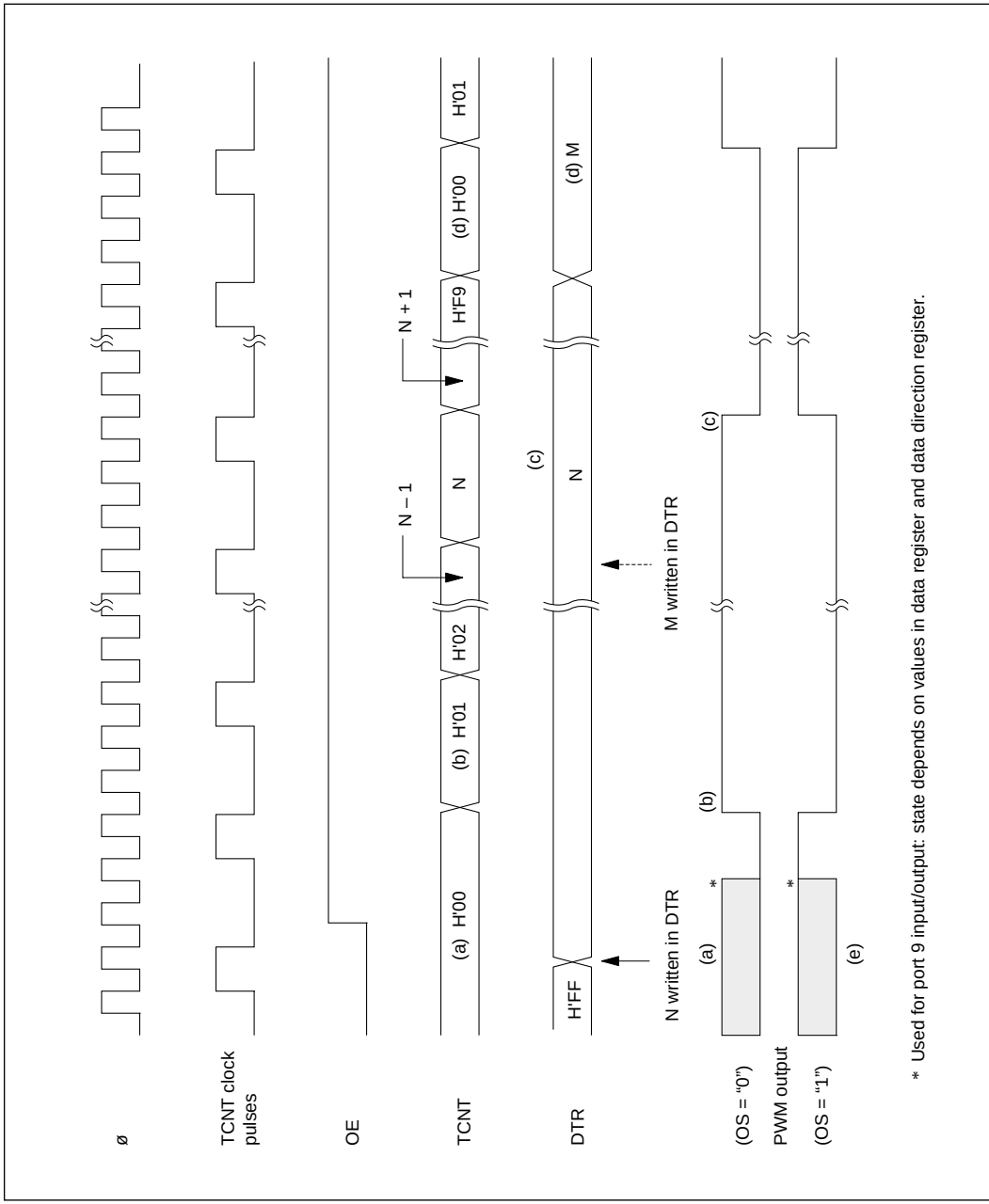


Figure 12-2 PWM Timing

12.4 Application Notes

Notes on the use of the PWM timer module are given below.

To use port 9 for PWM output, first set the P9PWME bit to 1 and clear the P9SCI2E bit to 0 in system control register 2 (SYSCR2).

Similarly, to use port 6 for PWM output, first set the P6PWME bit to 1 and clear the corresponding interrupt enable bit or bits (IRQ3E, IRQ4E, IRQ5E) to 0 in SYSCR2.

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 PW output remains constant at 0. If the DTR value is H'FA to H'FF, the duty factor is 100% and PW 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. PWM output and serial communication interface functions cannot be mixed among pins P94, P93, and P92.

Section 13 Watchdog Timer

13.1 Overview

The H8/534 and H8/536 have an on-chip watchdog timer (WDT) module. This module can monitor system operation by generating a signal that resets the entire chip if a system crash allows the timer count to overflow.

When this watchdog function is not needed, the WDT module can be used as an interval timer. In the interval timer mode, an interval timer interrupt is requested at each counter overflow.

The WDT module is also used in recovering from the software standby mode.

13.1.1 Features

The basic features of the watchdog timer module are summarized as follows:

- Selection of eight clock sources
- Selection of two modes: watchdog timer mode and interval timer mode
- Counter overflow generates a reset signal or interrupt request
  Reset signal in watchdog timer mode; interval timer interrupt request in interval timer mode.
- External output of reset signal
  The reset signal generated in watchdog timer mode resets the entire H8/534 or H8/536 chip. Depending on a reset output enable bit, the reset signal can also be output from the $\overline{\text{RES}}$ pin to reset devices controlled by the H8/534 or H8/536.

13.1.2 Block Diagram

Figure 13-1 is a block diagram of the watchdog timer.

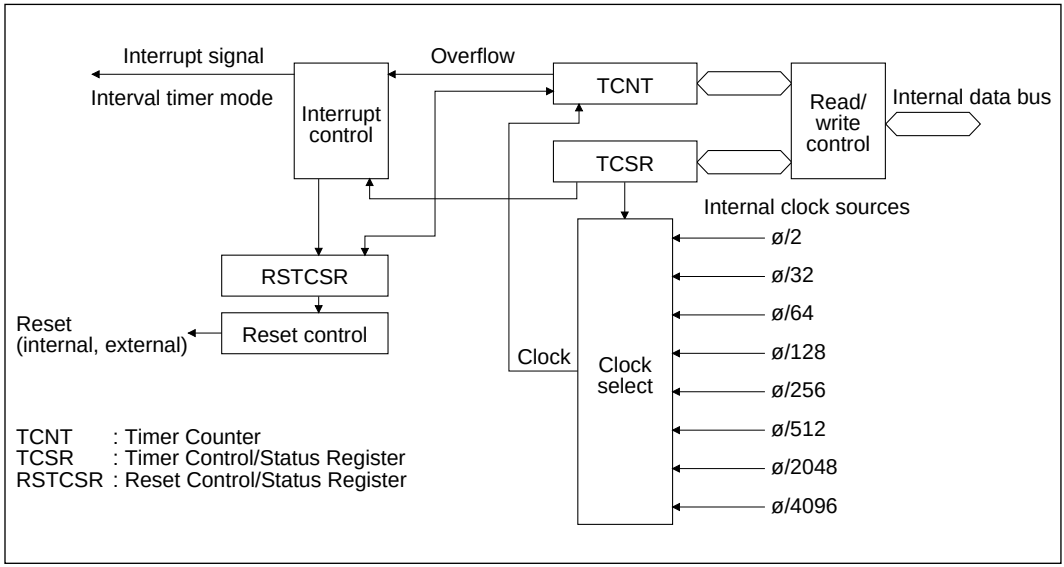


Figure 13-1 Block Diagram of Timer Counter

13.1.3 Register Configuration

Table 13-1 lists information on the watchdog timer registers.

Table 13-1 Register Configuration

			Initial	Addresses	
Name	**Abbreviation**	**R/W**	**Value**	**Write**	**Read**
Timer control/status register	TCSR	R/(W)*	H'18	H'FEEC	H'FEEC
Timer counter	TCNT	R/W	H'00	H'FEEC	H'FEED
Reset control/status register	RSTCSR	R/(W)*	H'3F	H'FF14	H'FF15

* Software can write a 0 to clear the status flag bits, but cannot write 1.

13.2 Register Descriptions

13.2.1 Timer Counter TCNT—H'FEEC (Write), H'FEED (Read)

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 watchdog timer counter (TCNT) is a readable/writable* 8-bit up-counter. When the timer enable bit (TME) in the timer control/status register (TCSR) 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) in the TCSR. When the count overflows (changes from H'FF to H'00), an overflow flag (OVF) in the TCSR is set to 1.

The watchdog timer counter is initialized to H'00 at a reset and when the TME bit is cleared to 0.

* TCNT is write-protected by a password. See section 13.2.4, “Notes on Register Access” for details.

13.2.2 Timer Control/Status Register (TCSR)—H'FEEC

Bit	7	6	5	4	3	2	1	0
	OVF	WT/$\overline{IT}$	TME	—	—	CKS2	CKS1	CKS0
Initial value	0	0	0	1	1	0	0	0
Read/Write	R/(W)*1	R/W	R/W	—	—	R/W	R/W	R/W

The watchdog timer control/status register (TCSR) is an 8-bit readable/writable*2 register that selects the timer mode and clock source and performs other functions.

Bits 7 to 5 are initialized to 0 at a reset and in the standby modes. Bits 2 to 0 are initialized to 0 at a reset, but retain their values in the standby modes.

*1 Software can write a 0 in bit 7 to clear the flag, but cannot set this bit to 1.
*2 The TCSR is write-protected by a password. See section 13.2.4, “Notes on Register Access” for details.

Bit 7—Overflow Flag (OVF): This bit indicates that the watchdog timer count has overflowed.

Bit 7 OVF	Description	
0	This bit is cleared to from 1 to 0 when the CPU reads the OVF bit after it has been set to 1, then writes a 0 in this bit.	(Initial value)
1	This bit is set to 1 when TCNT changes from H'FF to H'00.*	

* OVF is not set in watchdog timer mode.

Bit 6—Timer Mode Select (WT/$\overline{\text{IT}}$): This bit selects whether to operate in the watchdog timer mode or interval timer mode.

Bit 6 WT/$\overline{\text{IT}}$	Description	
0	Interval timer mode (interval timer interrupt request)	(Initial value)
1	Watchdog timer mode (reset)	

Bit 5—Timer Enable (TME): This bit enables or disables the timer.

Bit 5 TME	Description	
0	TCNT is initialized to H'00 and stopped.	(Initial value)
1	TCNT runs. A reset or interrupt request is generated when the count overflows.	

Bits 4 and 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 clock sources obtained by dividing the system clock (ø).

The overflow interval listed in the table below is the time from when the watchdog timer counter begins counting from H'00 until an overflow occurs.

Bit 2	Bit 1	Bit 0	Description	
CKS2	CKS1	CKS0	Clock Source	Overflow Interval (ø = 10 MHz)
0	0	0	ø/2	51.2µs (Initial value)
0	0	1	ø/32	819.2µs
0	1	0	ø/64	1.6ms
0	1	1	ø/128	3.3ms
1	0	0	ø/256	6.6ms
1	0	1	ø/512	13.1ms
1	1	0	ø/2048	52.4ms
1	1	1	ø/4096	104.9ms

13.2.3 Reset Control/Status Register (RSTCSR)—H'FF14 (Write), H'FF15 (Read)

Bit	7	6	5	4	3	2	1	0
	WRST	RSTOE						
Initial value	0	0	1	1	1	1	1	1
Read/Write	R/(W)*1	R/W	—	—	—	—	—	—

The reset control/status register (RSTCSR) is an 8-bit readable/writable*2 register that indicates when a reset has been caused by a watchdog timer overflow, and controls external output of the reset signal.

Bit 6 is not initialized by the reset caused by the watchdog timer overflow. It is initialized, however, by a reset caused by input at the $\overline{\text{RES}}$ pin.

*1 Software can write a 0 in bit 7 to clear the flag, but cannot set this bit to 1.
*2 The RSTCSR is write-protected by a password. See section 13.2.4, “Notes on Register Access” for details.

Bit 7—Watchdog Timer Reset (WRST): This bit indicates that a reset signal has been generated by a watchdog timer overflow in the watchdog timer mode.

The reset signal generated by the overflow resets the entire H8/534 or H8/536 chip. In addition, if the reset output enable (RSTOE) bit is set to 1, the reset signal (Low) is output at the $\overline{\text{RES}}$ pin to reset devices connected to the H8/534 or H8/536.

The WRST bit can be cleared by software by writing a 0. It is also cleared when a reset signal from an external device is received at the $\overline{\text{RES}}$ pin.

Bit 7 WRST	Description
0	This bit is cleared to 0 by a reset signal input from the $\overline{RES}$ pin, or when the CPU reads WRST after it has been set to 1, then writes a 0 in this bit. (Initial state)
1	This bit is set to 1 when the watchdog timer overflows in the watchdog timer mode and an internal reset signal is generated.

Bit 6—Reset Output Enable (RSTOE): This bit selects whether the reset signal generated by a watchdog timer overflow in the watchdog timer mode is output from the $\overline{RES}$ pin.

Bit 6 RSTOE	Description
0	The reset signal generated by watchdog timer overflow is not output to external devices. (Initial state)
1	The reset signal generated by watchdog timer overflow is output to external devices.

Bits 5 to 0—Reserved: These bits cannot be modified and are always read as 1.

13.2.4 Notes on Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in being more difficult to write. The procedures for writing and reading these registers are given below.

Writing to TCNT and TCSR: These registers must be written by word access. Programs cannot write to them by byte access. The word must contain the write data and a password.

The watchdog timer's TCNT and TCSR registers both have the same write address. The write data must be contained in the lower byte of the word written at this address. The upper byte must contain H'5A (password for TCNT) or H'A5 (password for TCSR). See figure 13-2.

The result of the access depicted in figure 13-2 is to transfer the write data from the lower byte to the TCNT or TCSR.

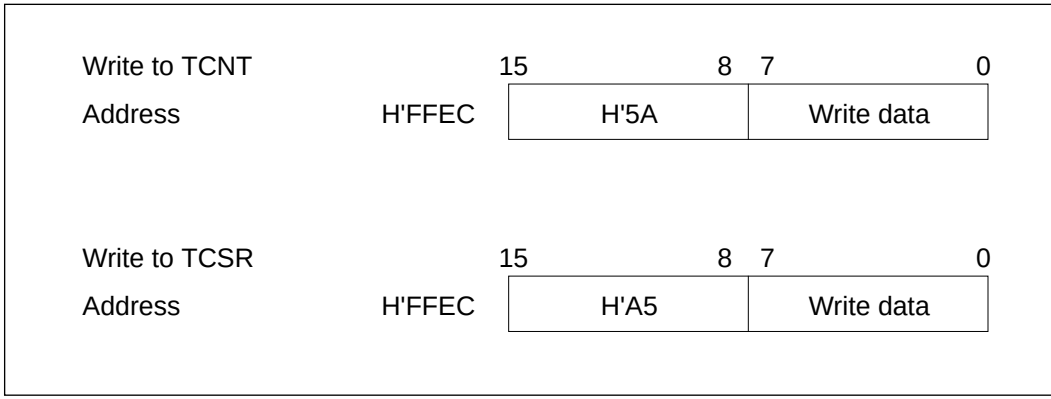


Figure 13-2 Writing to TCNT and TCSR

Writing to RSTCSR: The RSTCSR must be written by moving word data to address H'FF14. It cannot be written by byte access.

The upper byte of the word must contain a password. Separate passwords are used for clearing the WRST bit and for writing a 1 or 0 to the RSTOE bit.

To clear the WRST bit, the word written at address H'FF14 must contain the password H'A5 in the upper byte and the data H'00 in the lower byte. This clears the WRST bit to 0.

To set or clear the RSTOE bit, the word written at address H'FF14 must contain the password H'5A in the upper byte and the write data in the lower byte. The value of bit 6 in the lower byte is written in the RSTOE bit.

These write operations are illustrated in figure 13-3.

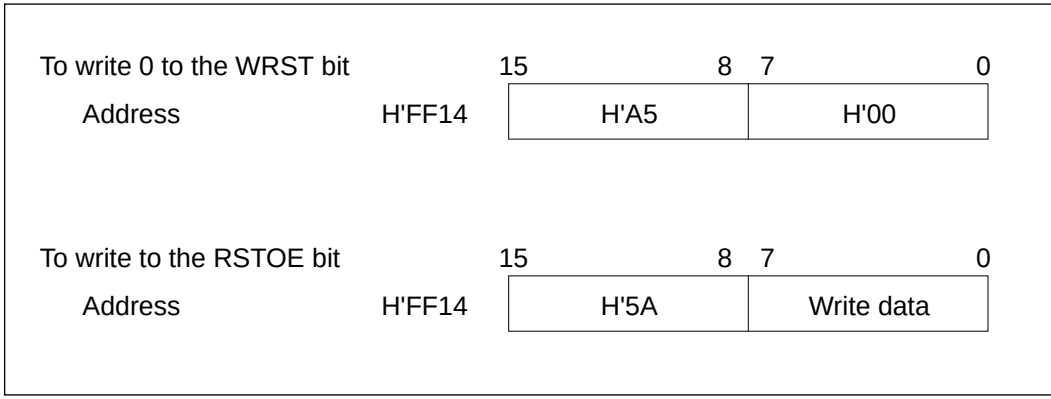


Figure 13-3 Writing to RSTCSR

Reading TCNT, TCSR, and RSTCSR: The read addresses are H'FEEC for TCSR, H'FEED for TCNT, and H'FF15 for RSTCSR as indicated in table 13-2.

These three registers are read like other registers. Byte access instructions can be used.

Table 13-2 Read Addresses of TCNT and TCSR

Read Address	Register
H'FFEC	TCSR
H'FFED	TCNT
H'FF15	RSTCSR

13.3 Operation

13.3.1 Watchdog Timer Mode

The watchdog timer function begins operating when software sets the WT/$\overline{IT}$ and TME bits to 1 in the TCSR.

Thereafter, software should periodically rewrite the contents of the timer counter (normally by writing H'00) to prevent the count from overflowing. If a program crash allows the timer count to overflow, the watchdog timer generates a reset as shown in figure 13-4.

The reset signal from the watchdog timer can also be output from the $\overline{RES}$ pin to reset external devices. This reset output signal is a Low pulse with a duration of 132 ø clock periods. The reset signal is output only if the RSTOE bit in the RSTCSR is set to 1.

The reset generated by the watchdog timer has the same vector as a reset generated by Low input at the $\overline{RES}$ pin. Software should check the WRST bit in the RSTCSR to determine the source of the reset.

If a watchdog timer overflow occurs at the same time as a Low input at the $\overline{RES}$ pin, priority is given to one type of reset or the other depending on the value of the RSTOE bit in the RSTCSR.

If the RSTOE bit is set to 1 when both types of reset occur simultaneously, the watchdog timer's reset signal takes precedence. The internal state of the H8/534 or H8/536 chip is reset and the $\overline{RES}$ pin is held Low for 132 ø clock periods. If at the end of 520 ø clock periods there is still an external Low input to the $\overline{RES}$ pin, the external reset takes effect, clearing the WRST and RSTOE bits to 0. Note that if the external reset occurs before the watchdog timer overflows, it takes effect immediately and clears the RSTOE bit.

If the RSTOE bit is cleared to 0 when both types of reset occur simultaneously, the reset signal input from the $\overline{RES}$ pin takes precedence and the WRST bit is cleared to 0.

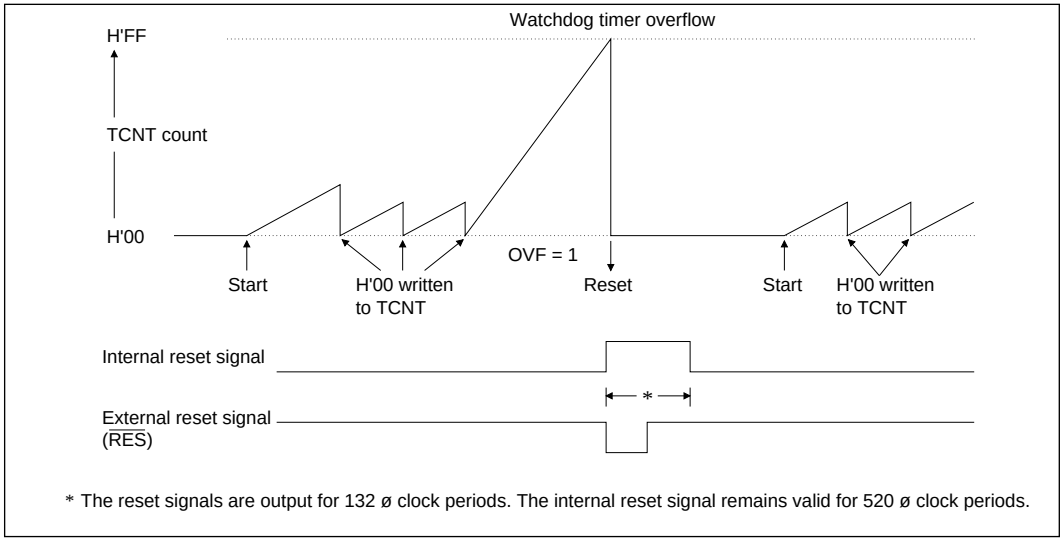


Figure 13-4 Operation in Watchdog Timer Mode

13.3.2 Interval Timer Mode

Interval timer operation begins when the WT/$\overline{IT}$ bit is cleared to 0 and the TME bit is set to 1.

In the interval timer mode, an interval timer interrupt request is generated each time the timer count overflows. This function can be used to generate interrupts at regular intervals.
See figure 13-5.

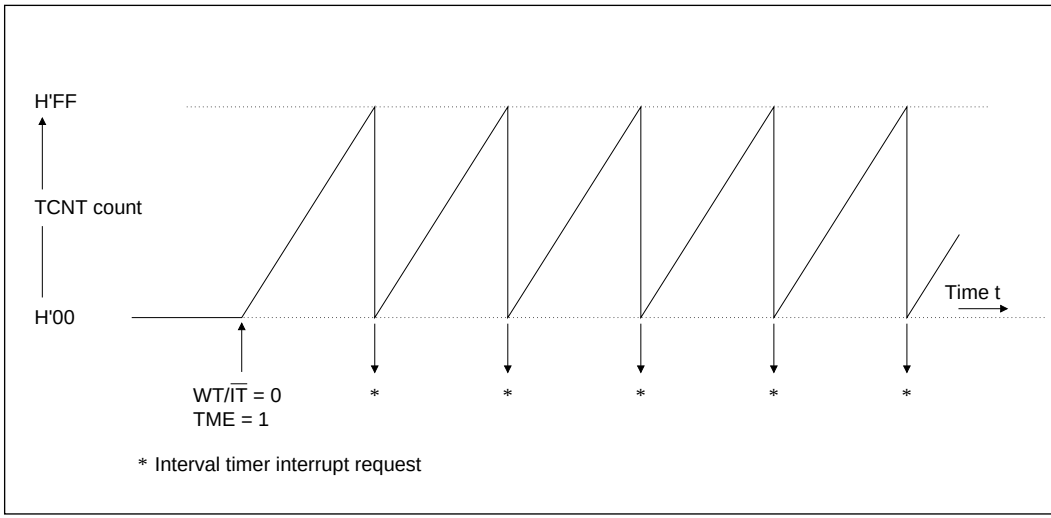


Figure 13-5 Operation in Interval Timer Mode

13.3.3 Operation in Software Standby Mode

The watchdog timer has a special function in recovery from software standby mode. Specific watchdog timer settings are required when the software standby mode is used.

Before Transition to the Software Standby Mode: The TME bit must be cleared to 0 to stop the watchdog timer counter before a transition to the software standby mode. The chip cannot enter the software standby mode while the TME bit is set to 1. Before entering the software standby mode, software should also set the clock select bits (CKS2 to CKS0) to a value that makes the timer overflow interval equal to or greater than the stabilization time of the clock oscillator.

Recovery from the Software Standby Mode: Recovery from the software standby mode can be triggered by an NMI request. In this case the recovery proceeds as follows:

When an NMI request signal is received, the clock oscillator starts running and the watchdog timer starts counting at the rate selected by the clock select bits before the software standby mode was entered. When the count overflows from H'FF to H'00, the ø clock is presumed to be stable and usable, clock signals are supplied to all modules on the chip, the standby mode ends, and the NMI interrupt-handling routine starts executing.

13.3.4 Setting of Overflow Flag

The OVF bit is set to 1 when the timer count overflows in the interval timer mode. Simultaneously, the WDT module requests an interval timer interrupt. The timing is shown in figure 13-6.

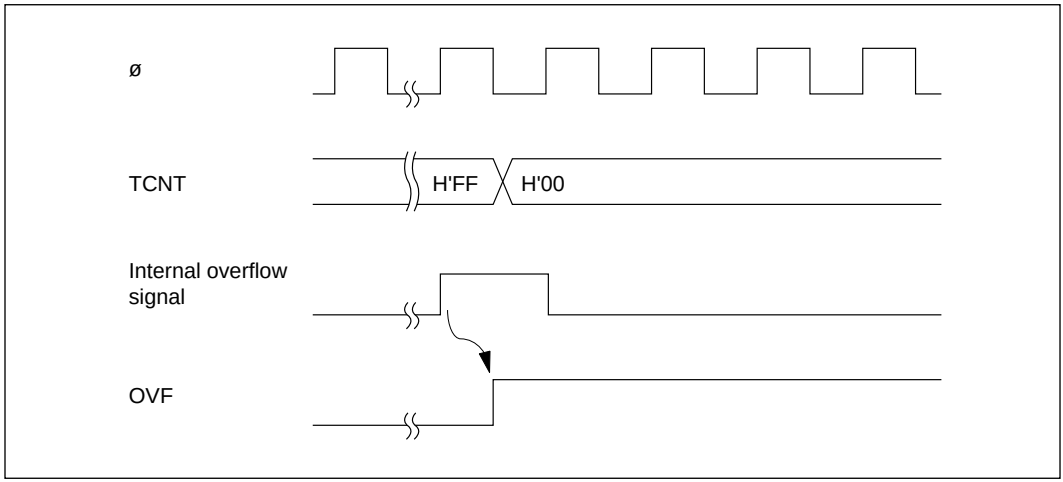


Figure 13-6 Setting of OVF Bit

13.3.5 Setting of Watchdog Timer Reset (WRST) Bit

The WRST bit is valid when WT/$\overline{IT}$ = 1 and TME = 1.

The WRST bit is set to 1 when the timer count overflows. An internal reset signal is simultaneously generated for the entire H8/534 or 536 chip. The timing is shown in figure 13-7.

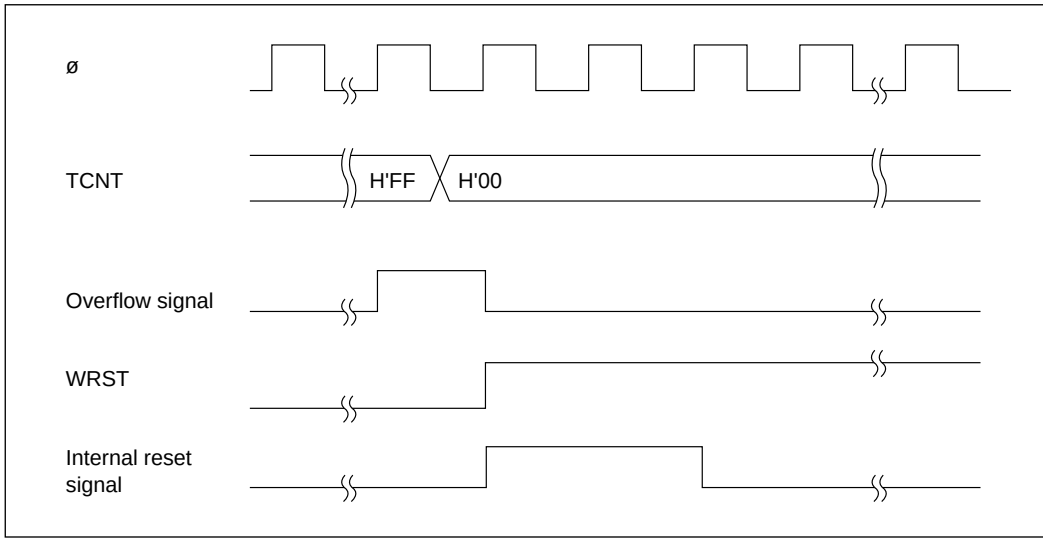


Figure 13-7 Setting of WRST Bit and Internal Reset Signal

13.4 Application Notes

Contention between TCNT Write and Increment: If a timer counter clock pulse is generated during the T_3 state of a write cycle to the timer counter, the write operation takes priority and the timer counter is not incremented. See figure 13-8.

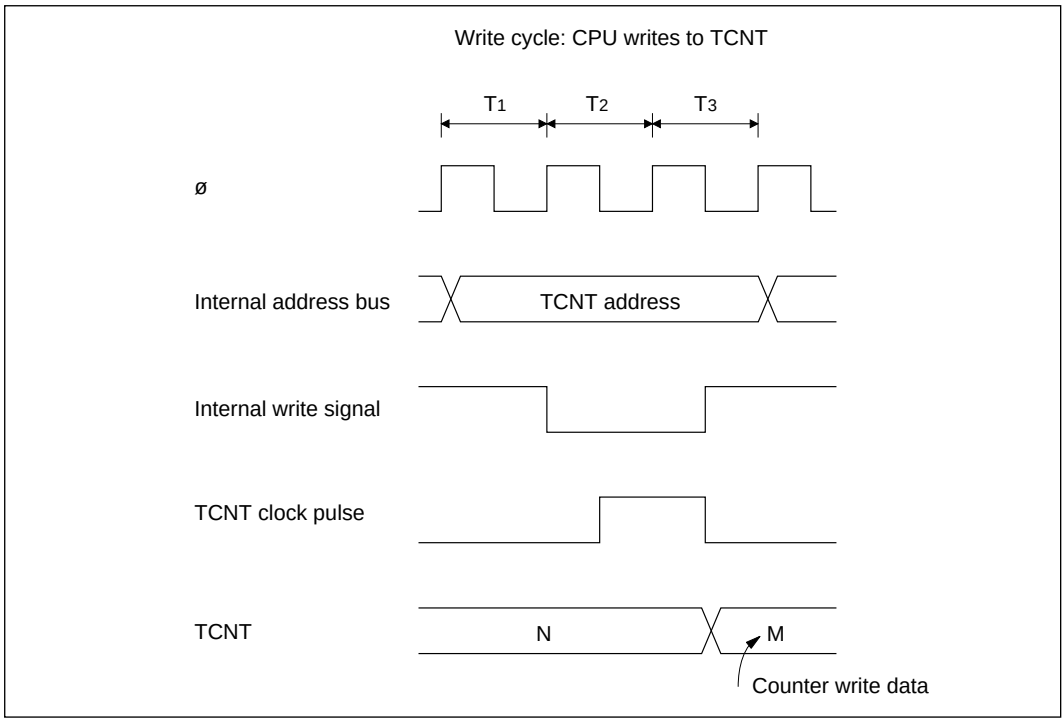


Figure 13-8 TCNT Write-Increment Contention

Changing the Clock Select Bits (CKS2 to CKS0): Software should stop the watchdog timer (by clearing the TME bit to 0) before changing the value of the clock select bits. If the clock select bits are modified while the watchdog timer is running, the timer count may be incremented incorrectly.

Use of Reset Output: When the reset signal is output to external devices, special circuitry is needed for input of the external reset signal.

The reset output is an NMOS open-drain output.

Figure 13-9 shows an example of a reset circuit.

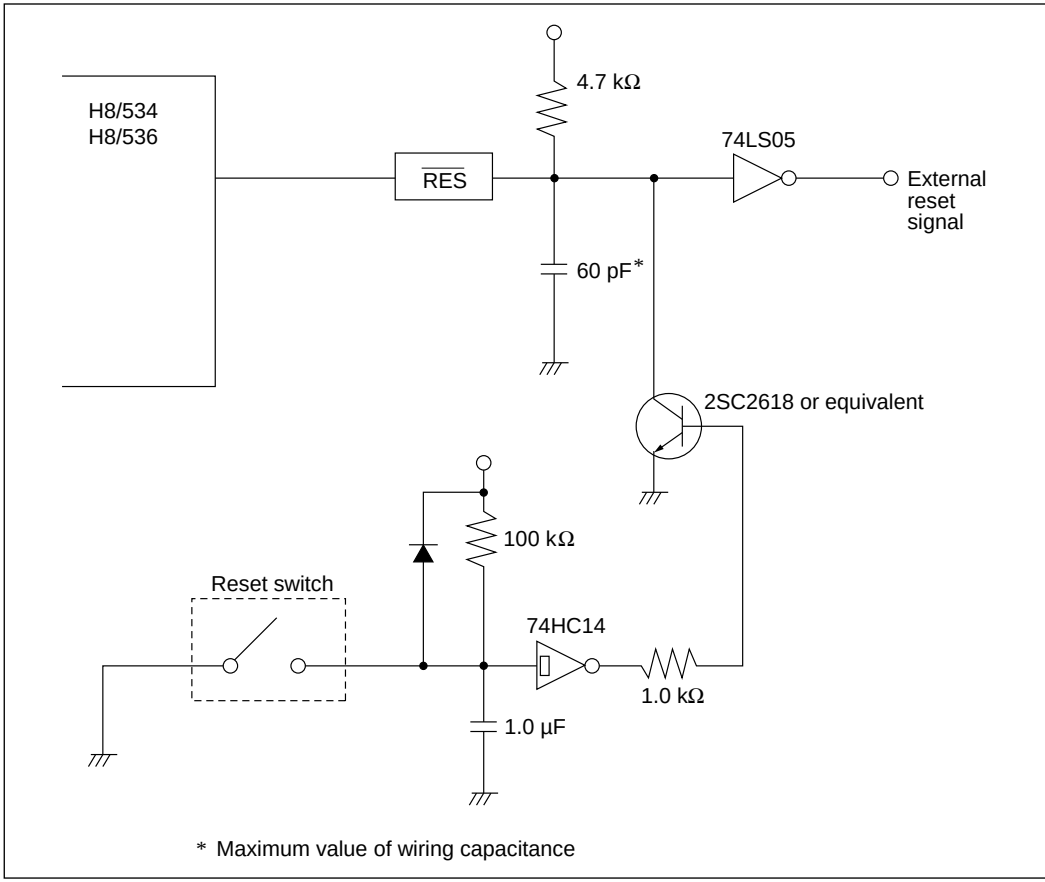


Figure 13-9 Reset Circuit (Example)

Section 14 Serial Communication Interface

14.1 Overview

The H8/534 and H8/536 have two serial communication interface channels (SCI1 and SCI2) for transferring serial data to and from other chips. Each channel supports both synchronous and asynchronous data transfer. Communication control functions are provided by eight internal registers.

14.1.1 Features

The features of the on-chip serial communication interface are:

- Selection of asynchronous or synchronous mode
  - — Asynchronous mode
    SCI1 and SCI2 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
    SCI1 and SCI2 can communicate with chips able to synchronize data transfers with clock pulses.
    - — Data length: 8 bits
    - — Error detection: Overrun errors
- Full duplex communication
  The transmitting and receiving sections are independent, so each channel 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 bit 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 SCK pin.
- Three interrupts
  Transmit-end, receive-end, and receive-error interrupts are requested independently. The transmit-end and receive-end interrupts can be served by the on-chip data transfer controller (DTC), providing a convenient way to transfer data with minimal CPU programming.

14.1.2 Block Diagram

Figure 14-1 shows a block diagram of one serial communication interface channel.

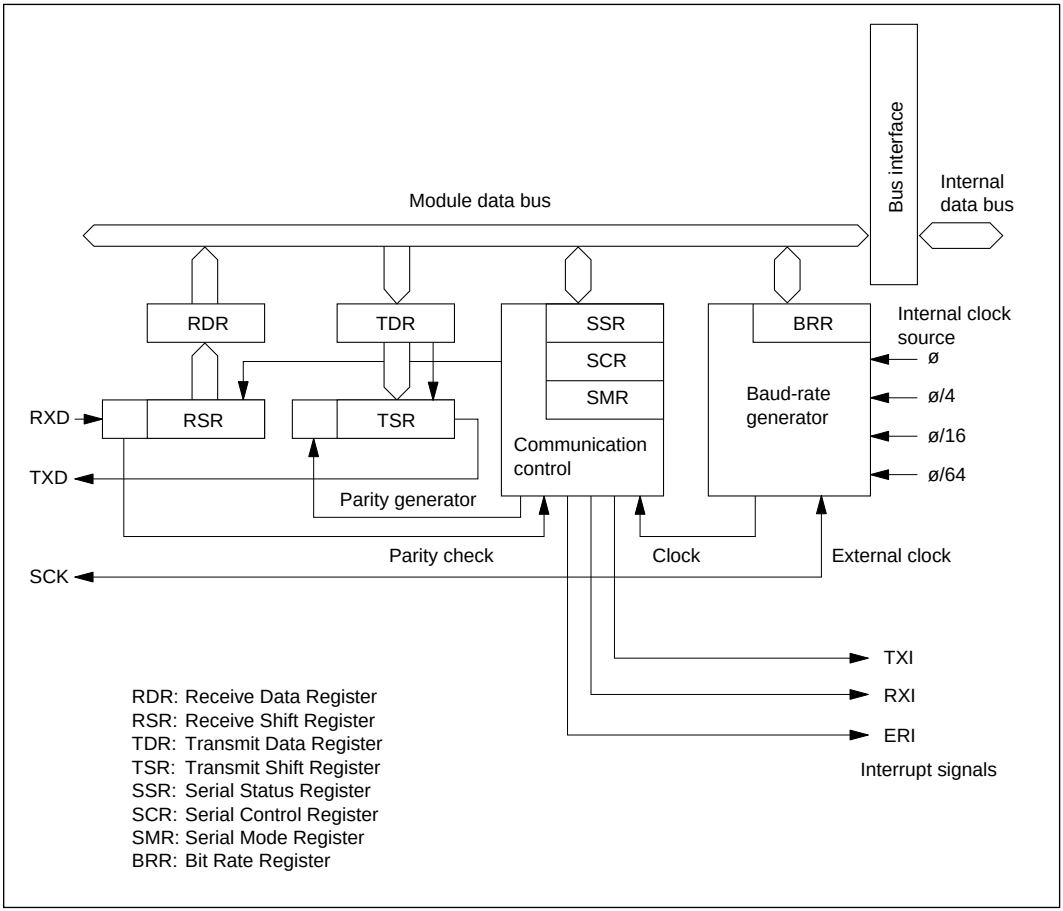


Figure 14-1 Block Diagram of Serial Communication Interface

14.1.3 Input and Output Pins

Table 14-1 lists the input and output pins used by the SCI module.

Table 14-1 SCI Input/Output Pins

Channel	Name	Abbreviation	I/O	Function
1	Serial clock	SCK_1	Input/output	Serial clock input and output.
	Receive data	RXD_1	Input	Receive data input.
	Transmit data	TXD_1	Output	Transmit data output.
2	Serial clock	SCK_2	Input/output	Serial clock input and output.
	Receive data	RXD_2	Input	Receive data input.
	Transmit data	TXD_2	Output	Transmit data output.

14.1.4 Register Configuration

Table 14-2 lists the SCI registers.

Table 14-2 SCI Registers

Channel	Name	Abbreviation	R/W	Initial Value	Address
1	Receive shift register	RSR	—	—	—
	Receive data register	RDR	R	H'00	H'FEDD
	Transmit shift register	TSR	—	—	—
	Transmit data register	TDR	R/W	H'FF	H'FEDB
	Serial mode register	SMR	R/W	H'04	H'FED8
	Serial control register	SCR	R/W	H'0C	H'FEDA
	Serial status register	SSR	R/(W)*	H'87	H'FEDC
	Bit rate register	BRR	R/W	H'FF	H'FED9
2	Receive shift register	RSR	—	—	—
	Receive data register	RDR	R	H'00	H'FEF5
	Transmit shift register	TSR	—	—	—
	Transmit data register	TDR	R/W	H'FF	H'FEF3
	Serial mode register	SMR	R/W	H'04	H'FEF0
	Serial control register	SCR	R/W	H'0C	H'FEF2
	Serial status register	SSR	R/(W)*	H'87	H'FEF4
	Bit rate register	BRR	R/W	H'FF	H'FEF1

* Software can write a 0 to clear the status flag bits, but cannot write a 1.

14.2 Register Descriptions

14.2.1 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 has been received, it is transferred to the receive data register (RDR).

The CPU cannot read or write the RSR directly.

14.2.2 Receive Data Register (RDR)—H'FEDD, H'FEF5

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.

14.2.3 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 TDR does not contain valid data, the SCI stops transmitting.

The CPU cannot read or write the TSR directly.

14.2.4 Transmit Data Register (TDR)—H'FEDB, H'FEF3

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.

14.2.5 Serial Mode Register (SMR)—H'FED8, H'FEF0

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	Communication is synchronized with the serial clock.	

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 not checked.	

Bit 4—Parity Mode (O/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/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 H8/534 or H8/536 chip.

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	ø clock	(Initial value)
0	1	ø/4 clock	
1	0	ø/16 clock	
1	1	ø/64 clock	

14.2.6 Serial Control Register (SCR)—H'FEDA, H'FEF2

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. It also enables and disables the receive-error interrupt (ERI) request.

Bit 6

RIE	Description	
0	The receive-end interrupt (RXI) and receive-error interrupt (ERI) requests are disabled.	(Initial value)
1	The receive-end interrupt (RXI) and receive-error interrupt (ERI) requests are enabled.	

Bit 5—Transmit Enable (TE): This bit enables or disables the transmit function. When the transmit function is enabled, the TXD pin is automatically used for output. When the transmit function is disabled, the TXD pin can be used as a general-purpose I/O port.

Bit 5

TE	Description	
0	The transmit function is disabled. The TXD pin can be used as a general-purpose I/O port.	(Initial value)
1	The transmit function is enabled. The TXD 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 RXD pin is automatically used for input. When the receive function is disabled, the RXD pin is available as a general-purpose I/O port.

Bit 4 RE	Description	
0	The receive function is disabled. The RXD pin can be used as a general-purpose I/O port.	(Initial value)
1	The receive function is enabled. The RXD 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 SCK pin is automatically used for input of the external clock signal.

Bit 1 CKE1	Description	
0	Internal clock source.	(Initial value)
1	External clock source. (The SCK pin is used for input.)	

Bit 0—Clock Enable 0 (CKE0): When an internal clock source is used in synchronous mode, this bit enables or disables serial clock output at the SCK pin.

This bit is ignored when the external clock is selected, or when the asynchronous mode is selected.

For further information on the communication format and clock source selection, see tables 14-5 and 14-6 in section 14.3, “Operation.”

Bit 0 CKE0	Description	
0	The SCK pin is not used by the SCI (and is available as a general-purpose I/O port).	(Initial value)
1	The SCK pin is used for serial clock output.	

14.2.7 Serial Status Register (SSR)—H'FEDC, H'FEF4

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	This bit is cleared from 1 to 0 when: 1. The CPU reads the TDRE bit after it has been set to 1, then writes a 0 in this bit. 2. The data transfer controller (DTC) writes data in the TDR.
1	This bit is set to 1 at the following times: (Initial value) 1. The chip is reset or enters a standby mode. 2. When TDR contents are transferred to the TSR. 3. When TDRE = 0 and the TE bit 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	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the RDRF bit after it has been set to 1, then writes a 0 in this bit. 2. The data transfer controller (DTC) reads the RDR. 3. The chip is reset or enters a standby mode.
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	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the ORER bit after it has been set to 1, then writes a 0 in this bit. 2. The chip is reset or enters a standby mode.
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 synchronous mode. It has no meaning in the asynchronous mode.

Bit 4 FER	Description
0	This bit is cleared to from 1 to 0 when: (Initial value) 1. The CPU reads the FER bit after it has been set to 1, then writes a 0 in this bit. 2. The chip is reset or enters a standby mode.
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	This bit is cleared from 1 to 0 when: (Initial value) 1. The CPU reads the PER bit after it has been set to 1, then writes a 0 in this bit. 2. The chip is reset or enters a standby mode.
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 bit in the SMR).

Bits 2 to 0—Reserved: These bits cannot be modified and are always read as 1.

14.2.8 Bit Rate Register (BRR)—H'FED9, H'FEF1

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 bit 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 14-3 and 14-4 show examples of BRR (N) and CKS (n) settings for commonly used bit rates.

Table 14-3 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 14-3 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 14-3 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	9	0
31250	0	4	–1.70	0	4	0	0	5	0	0	5	+2.40
38400	0	3	0	0	3	+1.73	—	—	—	0	4	0

Table 14-3 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

	XTAL Frequency (MHz)											
	24			24.576			28			29.4912			32		
Bit Rate	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	212	0.03	2	217	0.08	2	248	–0.17	3	64	0.70	3	70	0.03
150	2	155	0.16	2	159	0.00	2	181	0.16	2	191	0.00	2	207	0.16
300	2	77	0.16	2	79	0.00	2	90	0.16	2	95	0.00	2	103	0.16
600	1	155	0.16	1	159	0.00	1	181	0.16	1	191	0.00	1	207	0.16
1200	1	77	0.16	1	79	0.00	1	90	0.16	1	95	0.00	1	103	0.16
2400	0	155	0.16	0	159	0.00	0	181	0.16	0	191	0.00	0	207	0.16
4800	0	77	0.16	0	79	0.00	0	90	0.16	0	95	0.00	0	103	0.16
9600	0	38	0.16	0	39	0.00	0	45	–0.93	0	47	0.00	0	51	0.16
19200	0	19	–2.34	0	19	0.00	0	22	–0.93	0	23	0.00	0	25	0.16
31250	0	11	0.00	0	11	2.40	0	13	0.00	0	14	–1.70	0	15	0.00
38400	0	9	–2.34	0	9	0.00	0	10	3.57	0	11	0.00	0	12	0.16

Note: If possible, select a setting such that the error is 1% or less.

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

B : Bit rate
N : BRR value ($0 \leq N \leq 255$)
OSC : Crystal oscillator frequency in MHz
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

The error in asynchronous mode is calculated as follows:

$$\text{Error (\%)} = \left\{ \frac{\text{OSC} \times 10^6}{\text{B} \times 64 \times 2^{2n} \times (\text{N} + 1)} - 1 \times 100 \right\}$$

Table 14-4 Examples of BRR Settings in Synchronous Mode

	XTAL Frequency (MHz)													
Bit	**2**		**4**		**8**		**10**		**16**		**20**		**32**	
Rate	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**	**n**	**N**
100	—	—	—	—	—	—	—	—	—	—	—	—	—	—
250	1	249	2	124	2	249	—	—	3	124	—	—	3	249
500	1	124	1	249	2	124	—	—	2	249	—	—	3	124
1k	0	249	1	124	1	249	—	—	2	124	—	—	2	249
2.5k	0	99	0	199	1	99	1	124	1	199	1	249	2	99
5k	0	49	0	99	0	199	0	249	1	99	1	124	1	199
10k	0	24	0	49	0	99	0	124	0	199	0	249	1	99
25k	0	9	0	19	0	39	0	49	0	79	0	99	0	159
50k	0	4	0	9	0	19	0	24	0	39	0	49	0	79
100k	—	—	0	4	0	9	—	—	0	19	0	24	0	39
250k	0	0*	0	1	0	3	0	4	0	7	0	9	0	15
500k			0	0*	0	1	—	—	0	3	0	4	0	7
1M					0	0*	—	—	0	1	—	—	0	3
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.

$$B = OSC/[8 \times 2^{2n} \times (N + 1)]$$

B : Bit rate
N : BRR value ($0 \leq N \leq 255$)
OSC : Crystal oscillator frequency in MHz
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

14.3 Operation

14.3.1 Overview

Each serial communication interface channel supports serial data transfer in both asynchronous and synchronous modes.

The communication format depends on settings in the SMR as indicated in table 14-5. The clock source and usage of the SCK pin depend on settings in the SMR and SCR as indicated in table 14-6.

Table 14-5 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 14-6 SCI Clock Source Selection

SMR	SCR		Clock	
C/$\overline{A}$	CKE1	CKE0	Source	SCK Pin
0	0	0	Internal	I/O port*
(Async		1		Clock output at same frequency as baud rate
mode)	1	0	External	Clock input at 16 times the baud rate frequency
		1		
1	0	0	Internal	Serial clock output
(Sync		1		
mode)	1	0	External	Serial clock input
		1		

* Cannot be used by the SCI.

Transmitting and receiving operations in the two modes are described next.

14.3.2 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 14-2 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 bit (at the 8th cycle of the internal serial clock, which runs at 16 times the bit rate).

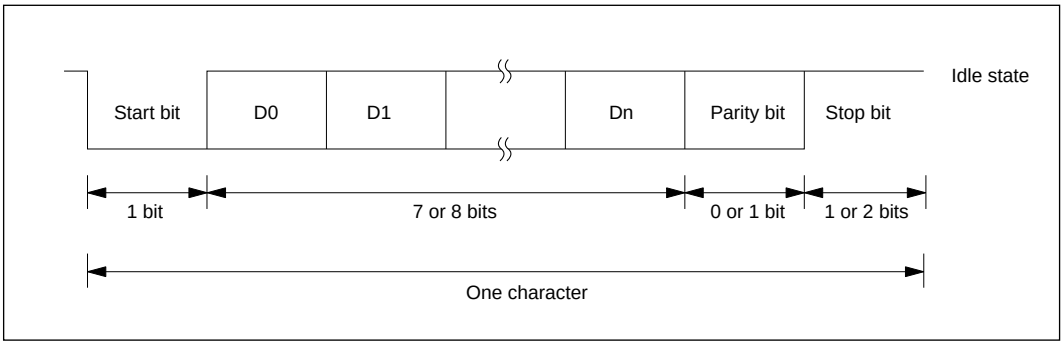


Figure 14-2 Data Format in Asynchronous Mode

1. **Data Format:** Table 14-7 lists the data formats that can be sent and received in asynchronous mode. Eight formats can be selected by bits in the SMR.

Table 14-7 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 SCK pin. Refer to table 14-6.

If an external clock is input at the SCK 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 SCK 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 14-3 shows the phase relationship between the output clock and transmit data.

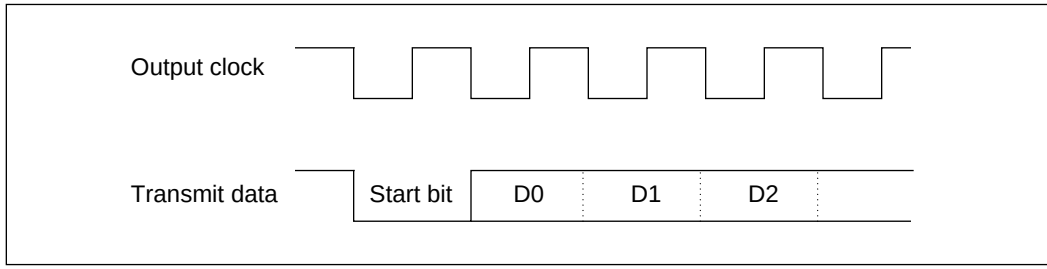


Figure 14-3 Phase Relationship between Clock Output and Transmit Data

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) Set the desired communication format in the SMR.

(2) Write the value corresponding to the desired bit rate in the BRR. (This step is not necessary if an external clock is used.)

(3) Select the clock and enable desired interrupts in the SCR.

(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 TXD 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.

* A frame is the data for one character, including the start bit and stop bit(s).

(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 TXD pin is held at 1 until the TDRE bit is cleared to 0.

• **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 RXD pin will 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 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 the CPU reads the SSR, then writes a 0 in the RDRF bit, or when the RDR is read by the data transfer controller (DTC). 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 14-8.

If a receive error occurs, the RDRF bit in the SSR is not 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, then write a 0 in the flag bit.

Table 14-8 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 bit in the SMR. The RSR contents are transferred to the RDR.

14.3.3 Synchronous Mode

The synchronous mode is suited for high-speed, continuous data transfer. Each bit of data is synchronized with a serial clock pulse.

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 14-4 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.

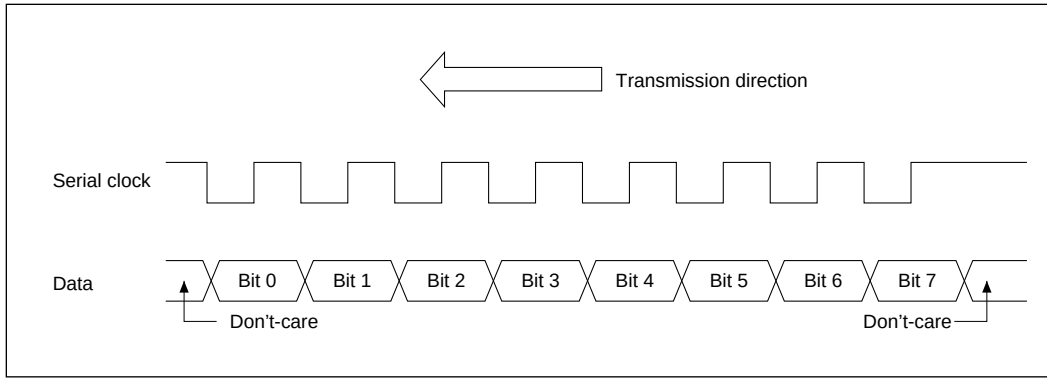


Figure 14-4 Data Format in Synchronous Mode

2. **Clock:** Either the internal serial clock created by the on-chip baud rate generator or an external clock input at the SCK pin can be selected in the synchronous mode. See table 14-6 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 in the SCR.

     (3) Select the synchronous mode in the SMR*.

     (4) Set the TE and/or RE bit to 1, and enable desired interrupts in the SCR.

   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.

   * The SCK pin is used for input or output according to the C/$\overline{A}$ bit in the serial mode register (SMR) and the CKE0 and CKE1 bits in the serial control register (SCR). (See table 14-6.) To prevent unwanted output at the SCK pin, pay attention to the order in which you set SMR and SCR.

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 TXD 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 SCK 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 TXD pin is held at the value of the last bit transmitted.

If the external clock source is selected, data transmission is synchronized with the clock signal input at the SCK 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 TXD output remains high.

- **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 RXD pin will 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 to the RDR so that it can be read.
The RDRF bit is cleared when the program reads the RDRF bit in the SSR, then writes a 0 in the RDRF bit, or when the data transfer controller (DTC) reads the RDR.

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 SCK 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 TXD and RXD 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, the CPU must read the TDRE bit in the SSR, write the next transmit data in the TDR, then clear the TDRE bit to 0. Alternatively, the DTC can write the next transmit data in the TDR, in which case the TDRE bit is cleared automatically.
If the TDRE bit is not cleared to 0 by the time the SCI finishes sending the current byte from the TSR, the TXD pin continues to output the last bit in the TSR.

(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 read the RDRF bit in the SSR, read the data in the RDR, then write a 0 in the RDRF bit. Alternatively, the DTC can read the RDR, in which case the RDRF bit is cleared automatically.
For continuous data reception, the RDRF bit must be cleared to 0 before the last bit of the next byte of data is received.

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.

14.4 CPU Interrupts and DTC 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 14-9 lists information about these interrupts.

Table 14-9 SCI Interrupts

Interrupt	Description	DTC Service Available?	Priority
ERI	Receive-error interrupt, requested when ORER, FER, or PER is set.	No	High
RXI	Receive-end interrupt, requested when RDRF is set.	Yes	
TXI	Transmit-end interrupt, requested when TDRE is set.	Yes	
			Low

The TXI and RXI interrupts can be served by the data transfer controller (DTC) to have a data transfer performed. When the DTC serves one of these interrupts, it clears the TDRE or RDRF bit to 0 under the following conditions, which differ between the two bits.

When invoked by a TXI request, if the DTC writes to the TDR, it automatically clears the TDRE bit to 0. When invoked by an RXI request, if the DTC reads from the RDR, it automatically clears the RDRF bit to 0.

See section 6, “Data Transfer Controller” for further information on the DTC.

14.5 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 14-10 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 14-10 SSR Bit States and Data Transfer When Multiple Receive Errors Occur

	SSR Bits				
Receive Error	**RDRF**	**ORER**	**FER**	**PER**	**RSR to 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

Notes: *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 RXD 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 bit rate. The falling edge of the start bit is detected by sampling the RXD 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 14-5.

   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%.

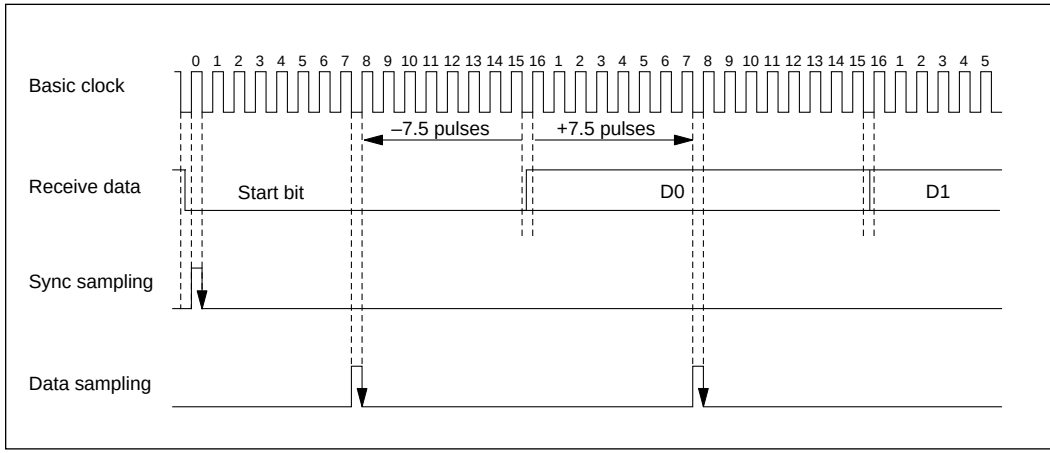


Figure 14-5 Sampling Timing (Asynchronous Mode)

$$M = \{(0.5 - 1/2N) - (D - 0.5)/N - (L - 0.5)F\} \times 100\ [\%] \quad (1)$$

M: Receive margin
N: Ratio of basic clock to bit rate (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

$$M = (0.5 - 1/2 \times 16) \times 100\ [\%] = 46.875\% \quad (2)$$

5. **Note on Transmitting in Synchronous Mode:** When setting up serial communication interface 1 or 2 to transmit in synchronous mode, make sure the ORER bit is cleared to 0. Transmit operation will fail to start if the ORER bit is set to 1. The same is true in simultaneous transmitting and receiving.

Section 15 A/D Converter

15.1 Overview

The H8/534 and H8/536 have an analog-to-digital converter module which can be programmed for input of analog signal on up to eight channels. A/D conversion is performed by the successive approximations method with 10-bit resolution.

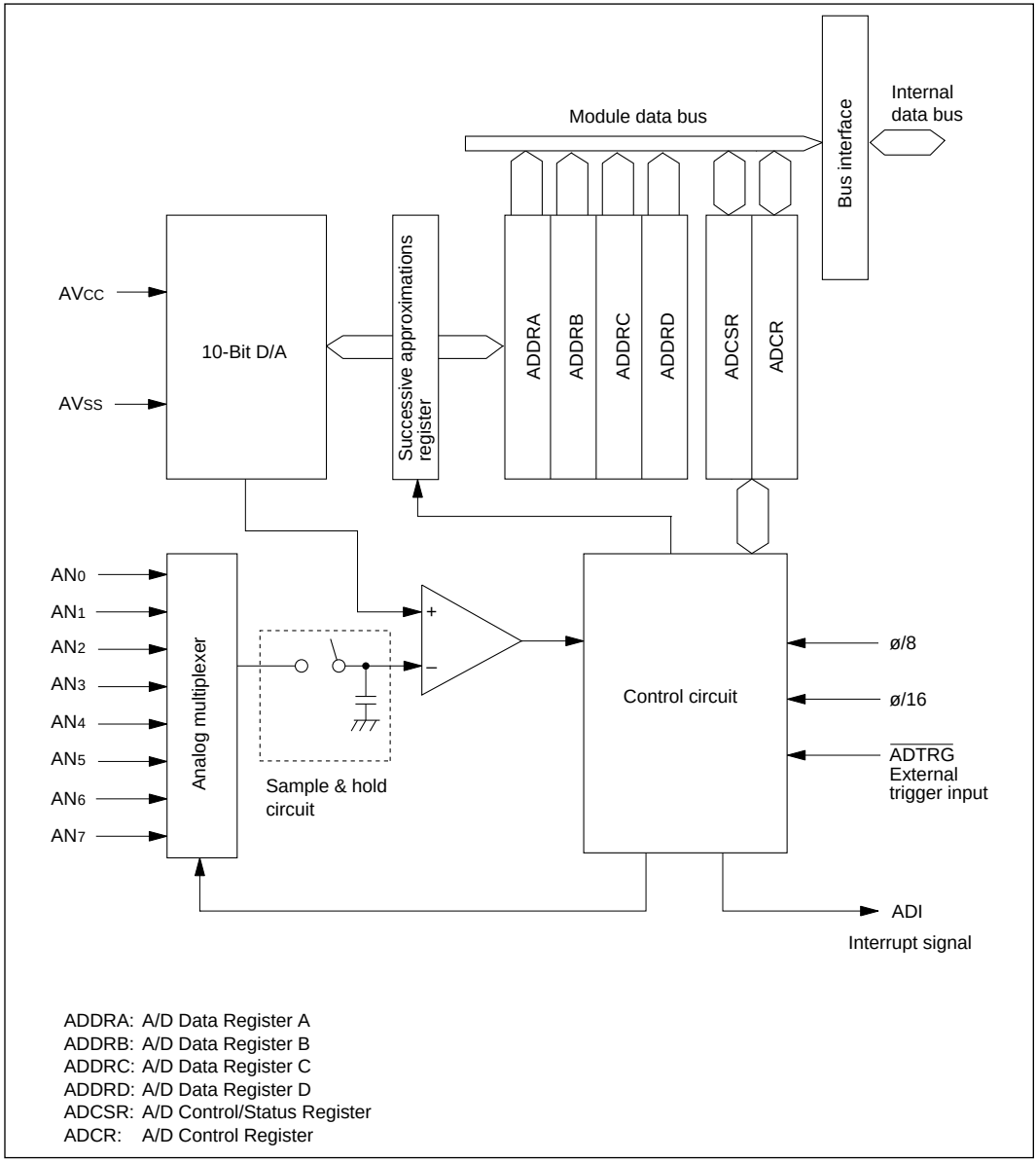
15.1.1 Features

The features of the on-chip A/D module are:

- Eight analog input channels
- Sample and hold circuit
- 10-Bit resolution
- Rapid conversion
  Conversion time is 13.8 µs per channel (at ø = 10 MHz)
- 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 16-bit data registers
  These registers store A/D conversion results for up to four channels.
- A/D conversion can be started by external trigger input.
- A CPU interrupt (ADI) can be requested at the completion of each A/D conversion cycle. This interrupt can also be served by the on-chip data transfer controller (DTC), providing a convenient way to move results into memory.

15.1.2 Block Diagram

Figure 15-1 shows a block diagram of A/D converter.



ADDRA: A/D Data Register A
ADDRB: A/D Data Register B
ADDRC: A/D Data Register C
ADDRD: A/D Data Register D
ADCSR: A/D Control/Status Register
ADCR: A/D Control Register

Figure 15-1 Block Diagram of A/D Converter

15.1.3 Input Pins

Table 15-1 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 15-1 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 input	$\overline{ADTRG}$	Input	External trigger input

15.1.4 Register Configuration

Table 15-2 lists the registers of the A/D converter module.

Table 15-2 A/D Registers

Name	Abbreviation	R/W	Initial Value	Address
A/D data register A (High)	ADDRA (H)	R	H'00	H'FEE0
A/D data register A (Low)	ADDRA (L)	R	H'00	H'FEE1
A/D data register B (High)	ADDRB (H)	R	H'00	H'FEE2
A/D data register B (Low)	ADDRB (L)	R	H'00	H'FEE3
A/D data register C (High)	ADDRC (H)	R	H'00	H'FEE4
A/D data register C (Low)	ADDRC (L)	R	H'00	H'FEE5
A/D data register D (High)	ADDRD (H)	R	H'00	H'FEE6
A/D data register D (Low)	ADDRD (L)	R	H'00	H'FEE7
A/D control/status register	ADCSR	R/(W)*	H'00	H'FEE8
A/D control register	ADCR	R/W	H'7F	H'FEE9

* Software can write 0 to clear the status flag bits but cannot write 1.

15.2 Register Descriptions

15.2.1 A/D Data Registers (ADDR)—H'FEE0 to H'FEE7

Bit	7	6	5	4	3	2	1	0
ADDRn H	AD_9	AD_8	AD_7	AD_6	AD_5	AD_4	AD_3	AD_2
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

(n = A to D)

Bit	7	6	5	4	3	2	1	0
ADDRn H	AD_1	AD_0	—	—	—	—	—	—
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 16-bit read-only registers that store the results of A/D conversion.

Each result consist of 10 bits. The first 8 bits are stored in the upper byte of the data register corresponding to the selected channel. The last two bits are stored in the lower data register byte. Each data register is assigned to two analog input channels as indicated in table 15-3.

The A/D data registers are always readable by the CPU. The upper byte can be read directly. The lower byte is read via a temporary register. See section 15-3, “CPU Interface” for details.

The unused bits (bits 5 to 0) of the lower data register byte are always read as 0.

The A/D data registers are initialized to H'0000 at a reset and in the standby modes.

Table 15-3 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

15.2.2 A/D Control/Status Register (ADCSR)—H'FEE8

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	This bit is cleared from 1 to 0 when: (Initial value) 1. The chip is reset or placed in a standby mode. 2. The CPU reads the ADF bit after it has been set to 1, then writes a 0 in this bit. 3. An A/D interrupt is served by the data transfer controller (DTC).
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.	

Bit 4—Scan Mode (SCAN): This bit selects the scan mode or single mode of operation. See section 15.4, “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 = 274 states (maximum)	(Initial value)
1	Conversion time = 138 states (maximum)	

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	AN_0
	0	1	AN_1	AN_0 and 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 and AN_5
	1	0	AN_6	AN_4 to AN_6
	1	1	AN_7	AN_4 to AN_7

15.2.3 A/D Control Register (ADCR)—H'FEE9

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 or disables the $\overline{\text{ADTRG}}$ (A/D external trigger) signal.

Bit 7

TRGE	Description	
0	External triggering of A/D conversion is disabled.	(Initial value)
1	A High-to-Low transition of $\overline{\text{ADTRG}}$ starts A/D conversion.	

Bit 6 to 0—Reserved: These bits cannot be modified and are always read as 1.

15.3 CPU Interface

The A/D data registers (ADDRA to ADDRD) are 16-bit registers. The upper byte of each register can be read directly, but the lower byte is accessed through an 8-bit temporary register (TEMP).

When the CPU or DTC reads the upper byte of an A/D data register, at the same time as the upper byte is placed on the internal data bus, the lower byte is transferred to TEMP. When the lower byte is accessed, the value in TEMP is placed on the internal data bus.

A program that requires all 10 bits of an A/D result should perform word access, or should read first the upper byte, then the lower byte of the A/D data register. Either way, it is assured of obtaining consistent data. Consistent data are not assured if the program reads the lower byte first.

A program that requires only 8-bit A/D accuracy should perform byte access to the upper byte of the A/D data register. The value in TEMP can be left unread.

Figure 15-2 shows the data flow when the CPU (or DTC) reads an A/D data register.

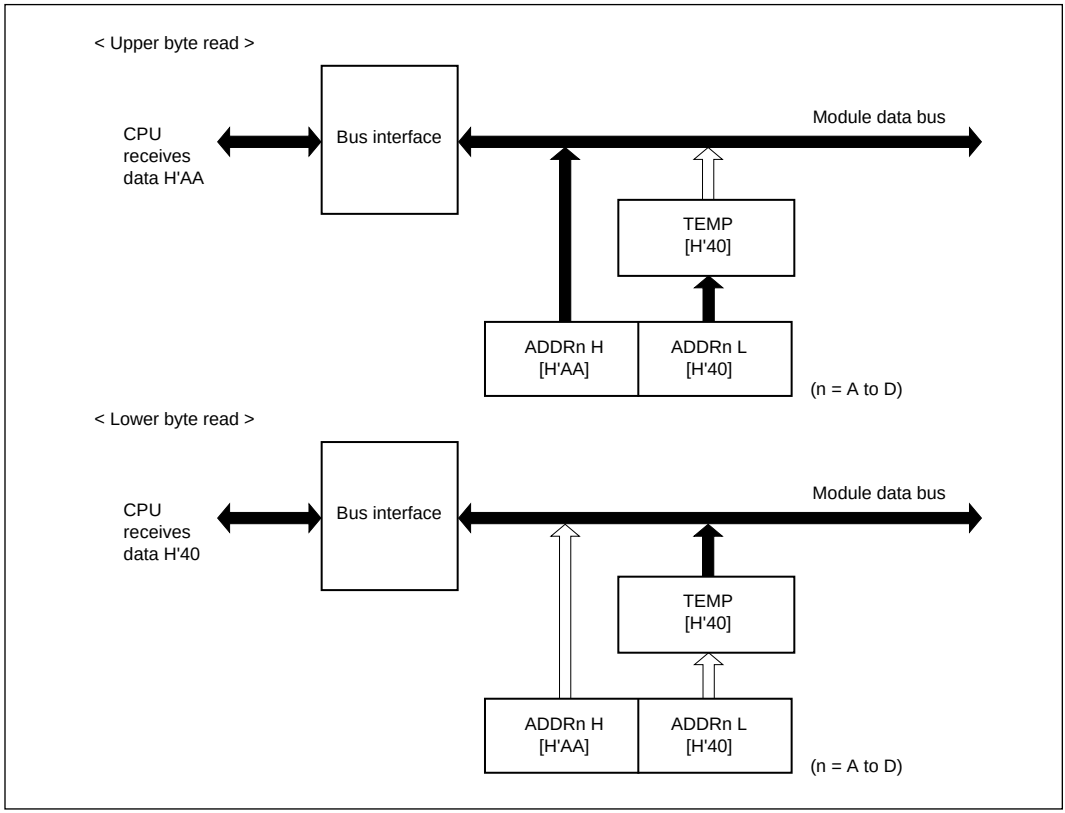


Figure 15-2 Read Access to A/D Data Register (When Register Contains H'AA40)

15.4 Operation

The A/D converter performs 10 successive approximations to obtain a result ranging from H'0000 (corresponding to AV_{SS}) to H'FFC0 (corresponding to AV_{CC}). Only the first 10 bits of the result are significant.

The A/D converter module can be programmed to operate in single mode or scan mode as explained below.

15.4.1 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. 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. Alternatively, the interrupt can be served by the data transfer controller (DTC).

When an A/D interrupt is served by the DTC, the DTC automatically clears the ADF bit to 0. When an A/D interrupt is served by the CPU, however, the ADF bit remains set until the CPU reads the ADCSR, then writes a 0 in the ADF 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.

The following example explains the A/D conversion process in single mode when channel 1 (AN_1) is selected. Figure 15-3 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. (Selection of mode, clock channel and setting the ADST bit can be done at same time.)
   Coding Example: (when using the slow clock, CKS = 0)
   ```
   BCLR #5, @H'FEE8
   MOV.B #H'61, @H'FEE8
   ```
2. The A/D converter samples the AN_1 input and converts the voltage level 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.

If the data transfer enable (DTE) bit is set to 1, the interrupt is served by the data transfer controller (DTC). Steps 4 to 7 then change as follows.

4'. The DTC is started.
5'. The DTC automatically clears the ADF bit to 0.
6'. The DTC transfers the A/D conversion result from ADDRB to a specified destination address.
7'. The DTC ends.

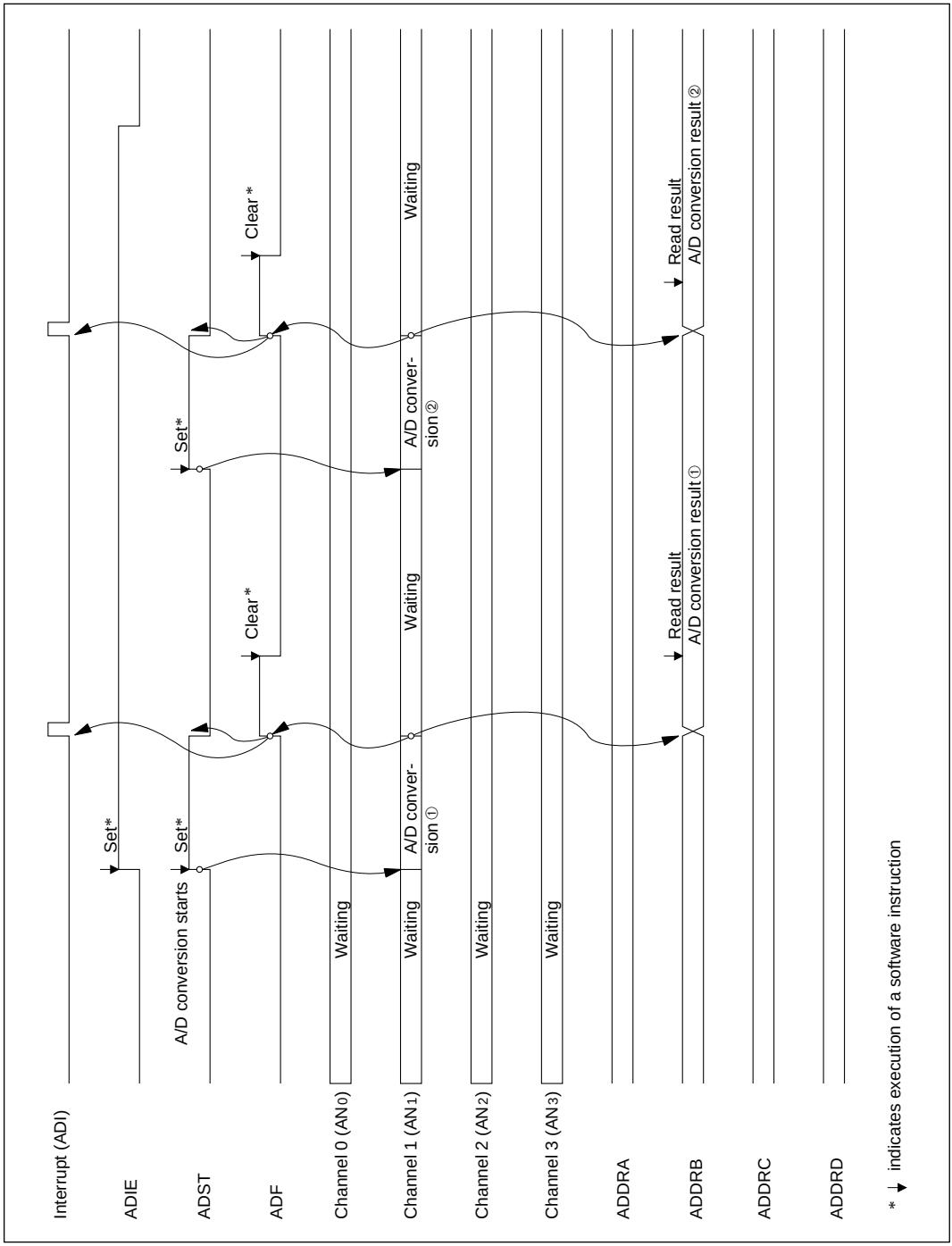


Figure 15-3 A/D Operation in Single Mode (When Channel 1 is Selected)

15.4.2 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, 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.

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.

The following example explains the A/D conversion process when three channels in group 0 are selected (AN_0, AN_1, and AN_2). Figure 15-4 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 AN_0 to AN_2 (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'FEE8
   MOV.B #H'72, @FEE8
   ```
2. The A/D converter samples the input at AN_0, converts the voltage level to a digital value, and transfers the result to register ADDRA.
3. Next the A/D converter samples and converts AN_1 and transfers the result to ADDRB. Then it samples and 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.

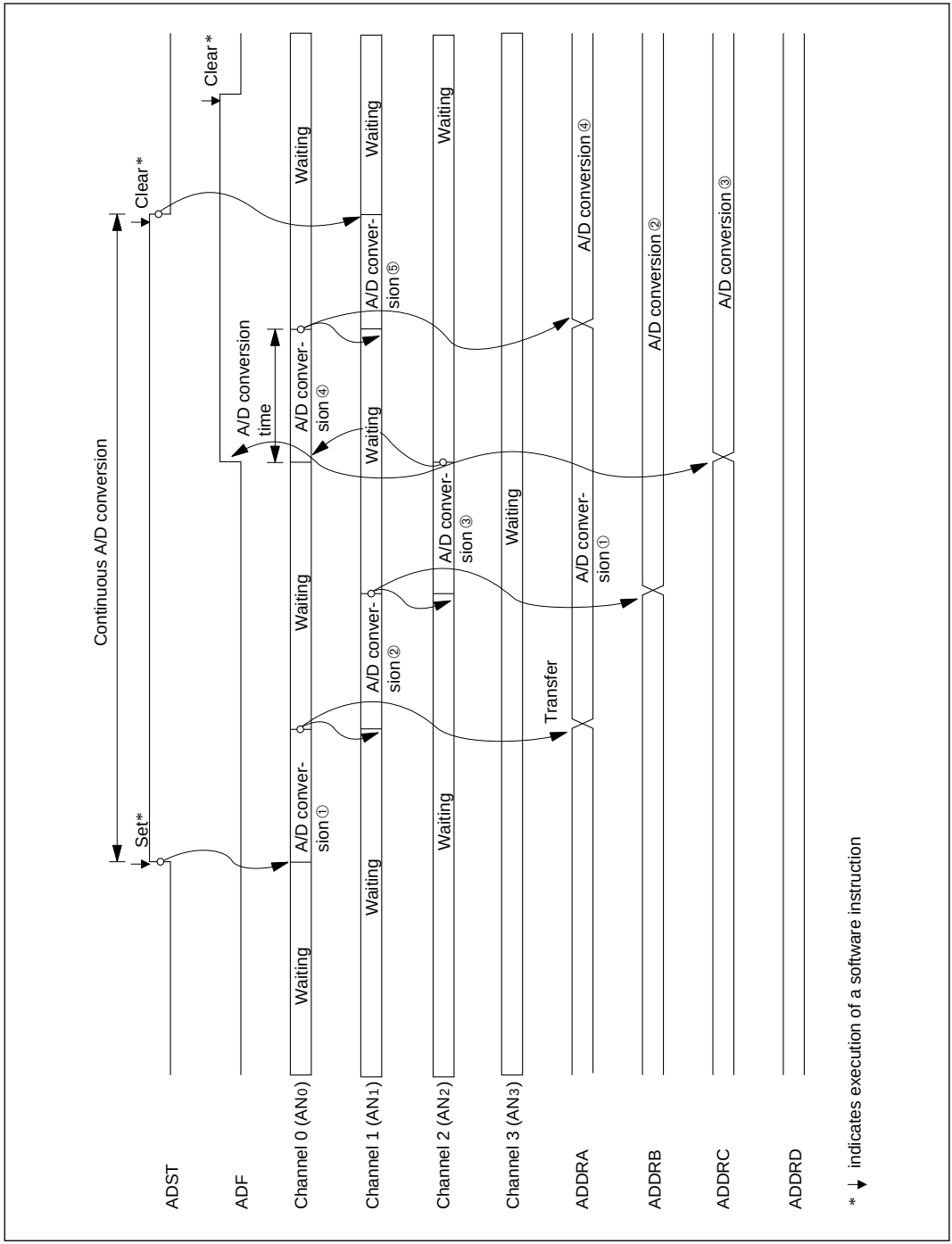


Figure 15-4 A/D Operation in Scan Mode (When Channels 0 to 2 are Selected)

15.4.3 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 15-5 shows the timing of these steps, and table 15-4 lists the total conversion times (t_{CONV}) for the single mode.

The total conversion time 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 15-4.

In the scan mode, the ranges given in table 15-4 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).

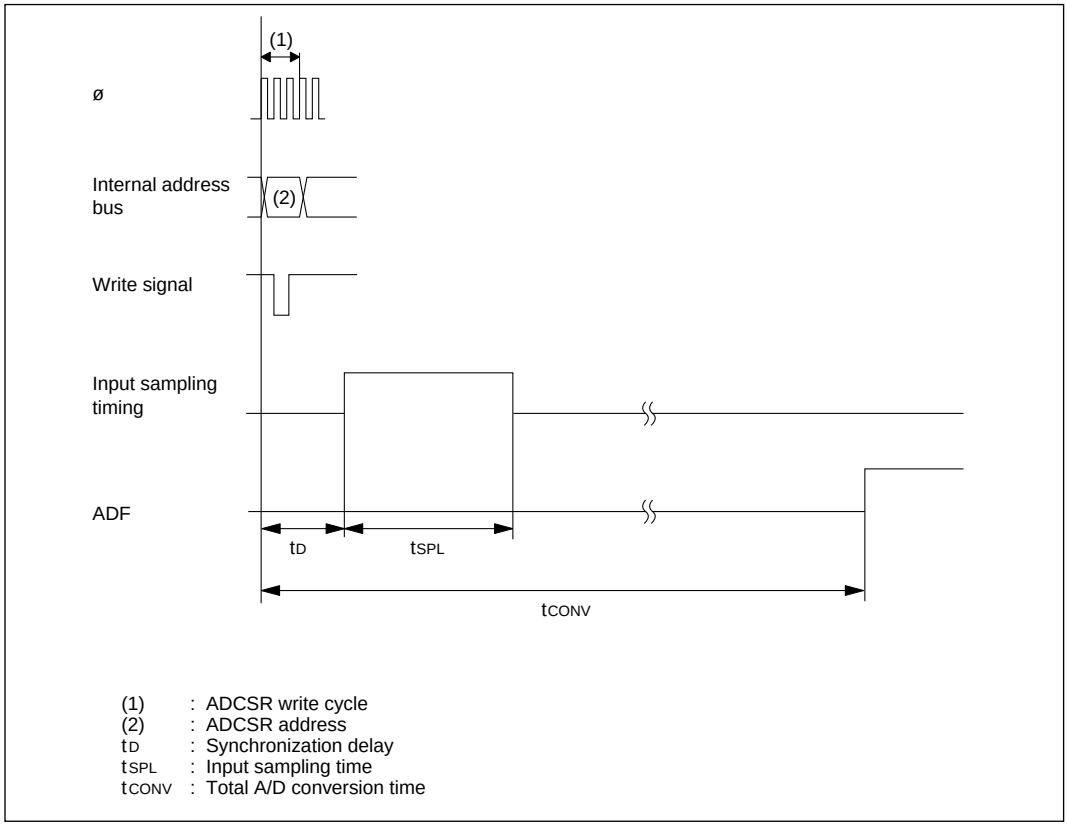


Figure 15-5 A/D Conversion Timing

Table 15-4 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}	259	—	274	131	—	138

Note: Values in the table are numbers of states.

15.4.4 External Triggering of A/D Conversion

A/D conversion can be started by an external trigger input.

External trigger input is enabled at the $\overline{\text{ADTRG}}$ pin when the TRGE bit in the ADCR is set to 1. Between 1.5 and 2 ø clock cycles after the $\overline{\text{ADTRG}}$ input goes Low, the ADST bit in the ADCSR is set to 1 and A/D conversion commences.

The timing of external triggering is shown in figure 15-6.

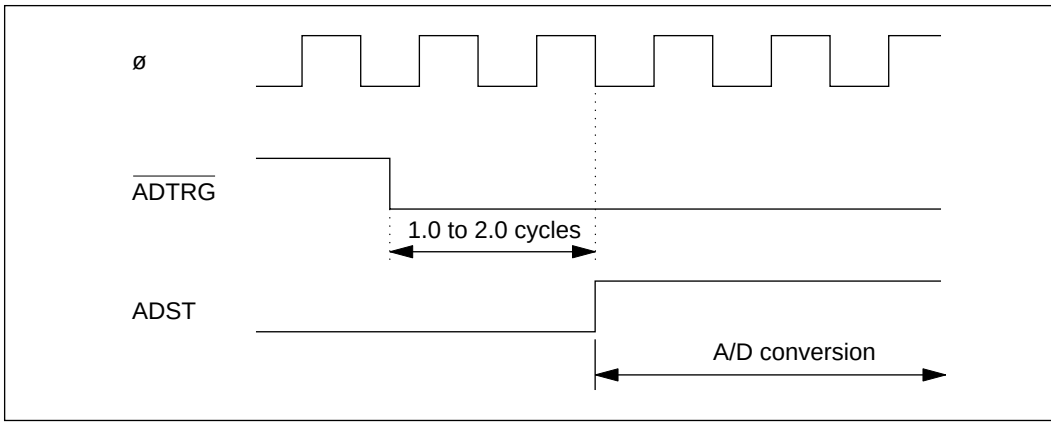


Figure 15-6 Timing of Setting of ADST Bit

15.5 Interrupts and the Data Transfer Controller

The ADI interrupt request is enabled or disabled by the ADIE bit in the ADCSR.

When the ADI bit in data transfer enable register DTEF (bit 4 at address H'FF0D) is set to 1, the ADI interrupt is served by the data transfer controller. The DTC can be used to transfer A/D results to a buffer in memory, or to an I/O port. The DTC automatically clears the ADF bit to 0.

Note: In scan mode, the DTC can transfer data for only one channel per interrupt, even if two or more channels are selected.

Section 16 RAM

16.1 Overview

The H8/534 and H8/536 include 2 kbytes 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'F680 to H'FE7F in the chip's address space. A RAM control register (RAMCR) can enable or disable the on-chip RAM, permitting these addresses to be allocated to external memory instead, if so desired.

16.1.1 Block Diagram

Figure 16-1 shows the block diagram of the on-chip RAM.

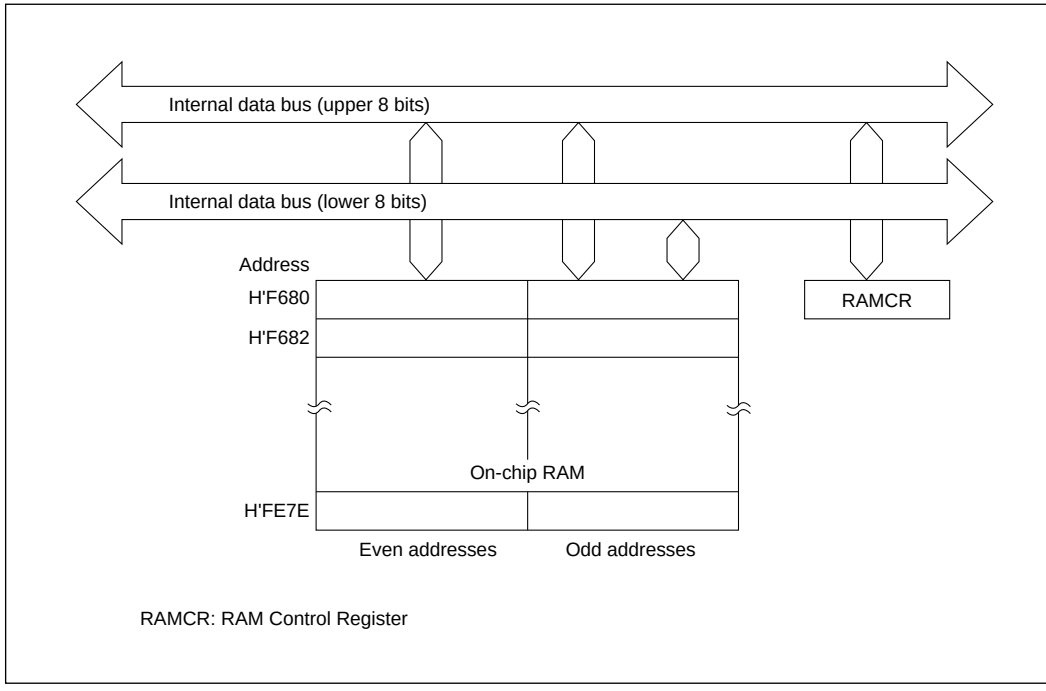


Figure 16-1 Block Diagram of On-Chip RAM

16.1.2 Register Configuration

The on-chip RAM is controlled by the register described in table 16-1.

Table 16-1 RAM Control Register

Name	Abbreviation	R/W	Initial Value	Address
RAM control register	RAMCR	R/W	H'FF	H'FF11

16.2 RAM Control Register (RAMCR)

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

The RAM control register (RAMCR) is an 8-bit register that enables or disable the on-chip RAM.

Bit 7—RAM Enable (RAME): This bit enables or disables the on-chip RAM.

The RAME bit is initialized on the rising edge of the reset signal. It 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)

Bits 6 to 0—Reserved: These bits cannot be modified and are always read as 1.

16.3 Operation

16.3.1 Expanded Modes (Modes 1, 2, 3, and 4)

If the RAME bit is set to 1, accesses to addresses H'F680 to H'FE7F are directed to the on-chip RAM. If the RAME bit is cleared to 0, accesses to addresses H'F680 to H'FE7F are directed to the external data bus.

16.3.2 Single-Chip Mode (Mode 7)

If the RAME bit is set to 1, accesses to addresses H'F680 to H'FE7F are directed to the on-chip RAM. If the RAME bit is cleared to 0, access of any type (instruction fetch or data read or write) to addresses H'F680 to H'FE7F causes an address error and initiates the CPU's exception-handling sequence.

Section 17 ROM

17.1 Overview

The H8/534 includes 32 kbytes of high-speed, on-chip ROM. The H8/536 has 62 kbytes of on-chip ROM. The on-chip ROM is connected to the CPU via a 16-bit data bus and is accessed in two states.

Users wishing to program the chip themselves can request electrically programmable ROM (PROM). The PROM version has a PROM mode in which the chip can be programmed with a standard, external PROM writer. The chip is also available with masked ROM.

The on-chip ROM is enabled or disabled depending on the MCU operating mode, which is determined by the inputs at the mode pins when the chip comes out of the reset state.
See table 17-1.

Table 17-1 ROM Usage in Each MCU Mode

	Mode Pins			
Mode	**MD_2**	**MD_1**	**MD_0**	**ROM**
Mode 1 (expanded minimum mode)	0	0	1	Disabled (external addresses)
Mode 2 (expanded minimum mode)	0	1	0	Enabled
Mode 3 (expanded maximum mode)	0	1	1	Disabled (external addresses)
Mode 4 (expanded maximum mode)	1	0	0	Enabled
Mode 7 (single-chip mode)	1	1	1	Enabled

17.1.1 Block Diagram

Figure 17-1 shows the block diagram of the on-chip ROM.

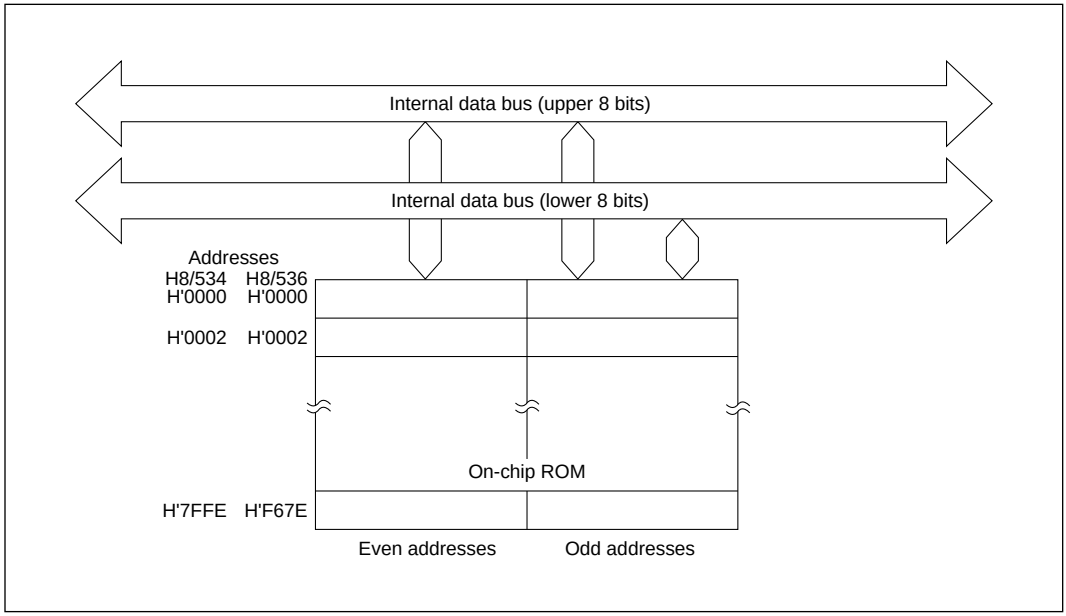


Figure 17-1 Block Diagram of On-Chip ROM

17.2 PROM Mode

17.2.1 PROM Mode Setup

The PROM version has a PROM mode in which the usual microcomputer functions of the H8/534 or H8/536 are halted to allow the on-chip PROM to be programmed.

To select the PROM mode, apply the signal inputs listed in table 17-2.

Table 17-2 Selection of PROM Mode

Pin	Input
Mode pins (MD_2, MD_1, and MD_0)	Low
$\overline{STBY}$ pin	Low
$P6_1$ and $P6_0$	High

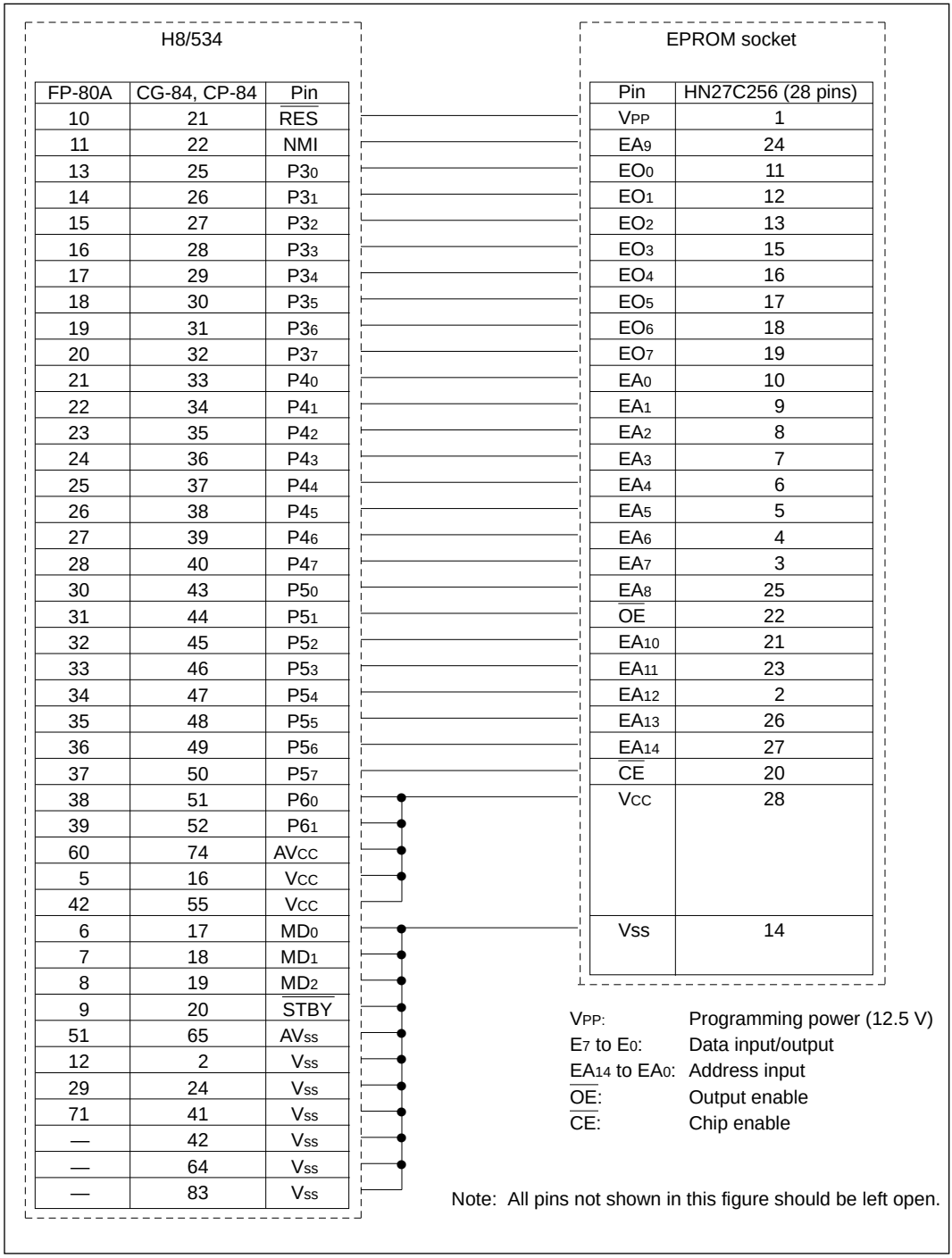
17.2.2 Socket Adapter Pin Arrangements and Memory Map

The H8/534 or H8/536 can be programmed with a general-purpose PROM writer by attaching a socket adapter as listed in table 17-3. The socket adapter depends on the type of package. Figure 17-2(a) and (b) show the socket adapter pin arrangements. Figure 17-3 is a memory map.

Table 17-3 Socket Adapter

Chip	Package	Socket Adapter
H8/534	84-Pin PLCC (CP-84)	HS538ESC01H
	84-Pin windowed LCC (CG-84)	HS538ESG01H
	80-Pin QFP (FP-80A)	HS538ESH01H
	80-Pin TQFP (TFP-80C)	HS5348ESN01H*
H8/536	84-Pin PLCC (CP-84)	HS538ESC02H
	84-Pin windowed LCC (CG-84)	HS538ESG02H
	80-Pin QFP (FP-80A)	HS538ESH02H
	80-Pin TQFP (TFP-80C)	HS5368ESN01H*

Note: * Under development.



FP-80A	CG-84, CP-84	Pin
10	21	$\overline{RES}$
11	22	NMI
13	25	$P3_0$
14	26	$P3_1$
15	27	$P3_2$
16	28	$P3_3$
17	29	$P3_4$
18	30	$P3_5$
19	31	$P3_6$
20	32	$P3_7$
21	33	$P4_0$
22	34	$P4_1$
23	35	$P4_2$
24	36	$P4_3$
25	37	$P4_4$
26	38	$P4_5$
27	39	$P4_6$
28	40	$P4_7$
30	43	$P5_0$
31	44	$P5_1$
32	45	$P5_2$
33	46	$P5_3$
34	47	$P5_4$
35	48	$P5_5$
36	49	$P5_6$
37	50	$P5_7$
38	51	$P6_0$
39	52	$P6_1$
60	74	AV_{CC}
5	16	V_{CC}
42	55	V_{CC}
6	17	MD_0
7	18	MD_1
8	19	MD_2
9	20	$\overline{STBY}$
51	65	AV_{SS}
12	2	V_{SS}
29	24	V_{SS}
71	41	V_{SS}
—	42	V_{SS}
—	64	V_{SS}
—	83	V_{SS}

Pin	HN27C256 (28 pins)
V_{PP}	1
EA_9	24
EO_0	11
EO_1	12
EO_2	13
EO_3	15
EO_4	16
EO_5	17
EO_6	18
EO_7	19
EA_0	10
EA_1	9
EA_2	8
EA_3	7
EA_4	6
EA_5	5
EA_6	4
EA_7	3
EA_8	25
$\overline{OE}$	22
EA_{10}	21
EA_{11}	23
EA_{12}	2
EA_{13}	26
EA_{14}	27
$\overline{CE}$	20
V_{CC}	28
V_{SS}	14

V_{PP}: Programming power (12.5 V)
E_7 to E_0: Data input/output
EA_{14} to EA_0: Address input
$\overline{OE}$: Output enable
$\overline{CE}$: Chip enable

Note: All pins not shown in this figure should be left open.

Figure 17-2(a) Socket Adapter Pin Arrangements (H8/534)

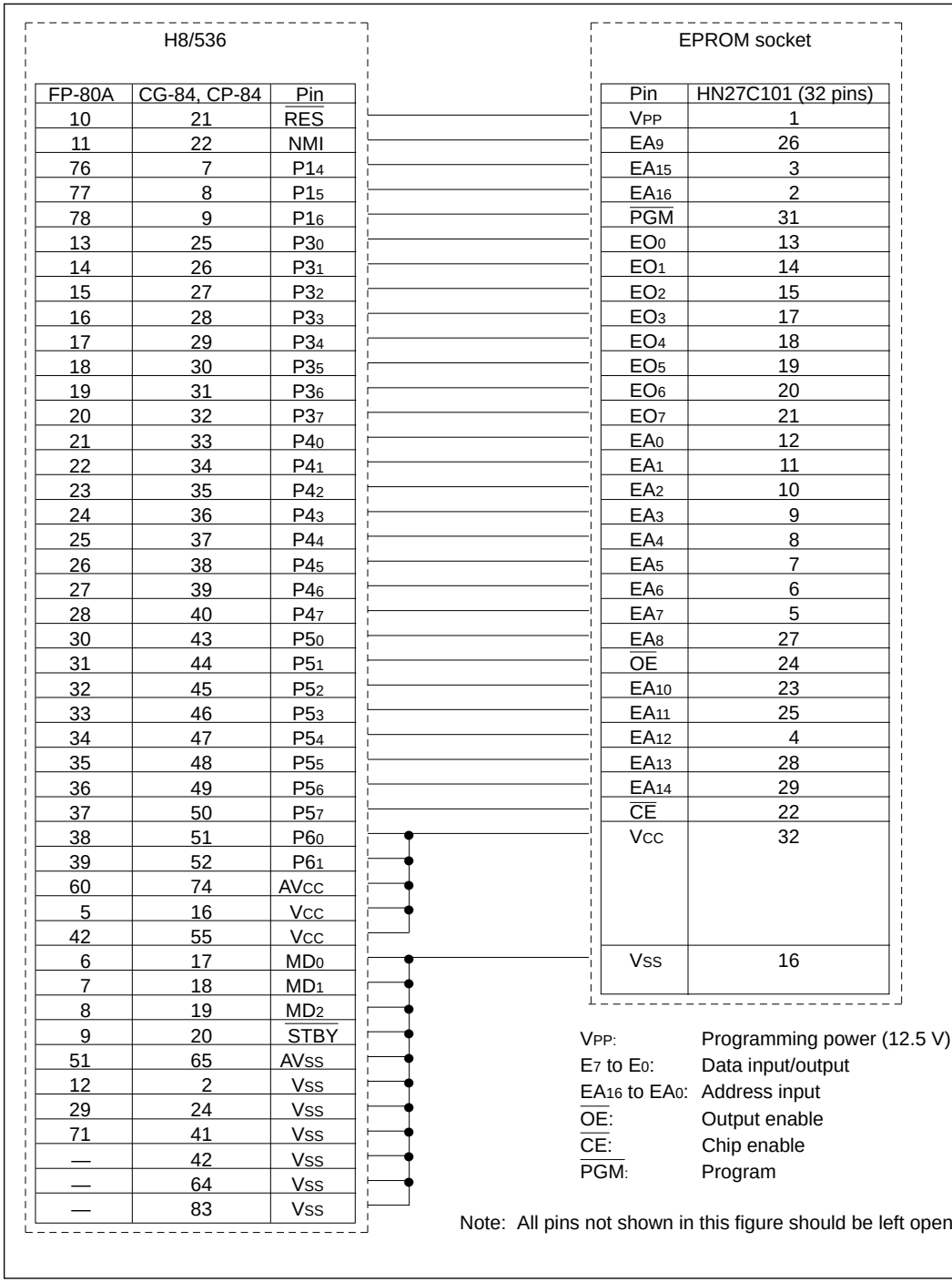


Figure 17-2(b) Socket Adapter Pin Arrangements (H8/536)

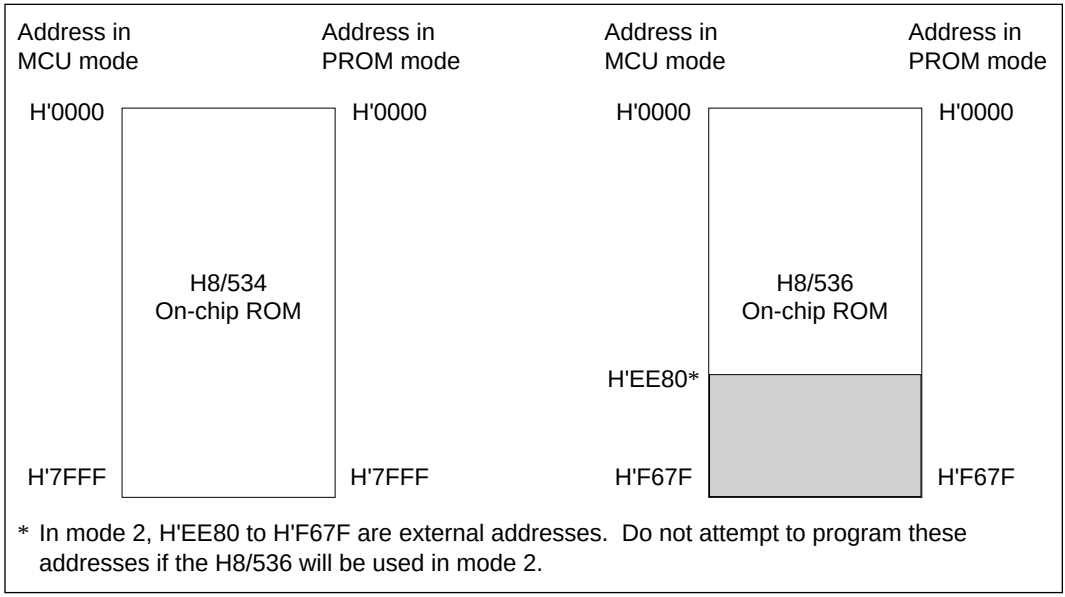


Figure 17-3 Memory Map in PROM Mode

17.3 H8/534 Programming

The write, verify, and inhibited sub-modes of the PROM mode are selected as shown in table 17-4.

Table 17-4 Selection of Sub-Modes in PROM Mode (H8/534)

	Pins					
Mode	**$\overline{CE}$**	**$\overline{OE}$**	**V_{PP}**	**V_{CC}**	**O_7 to O_0**	**A_{14} to A_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 H8/534 PROM uses the same, standard read/write specifications as the HN27C256 and HN27256.

17.3.1 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 17-4 shows the basic high-speed programming flowchart.

Tables 17-5 and 17-6 list the electrical characteristics of the chip in the PROM mode. Figure 17-5 shows a write/verify timing chart.

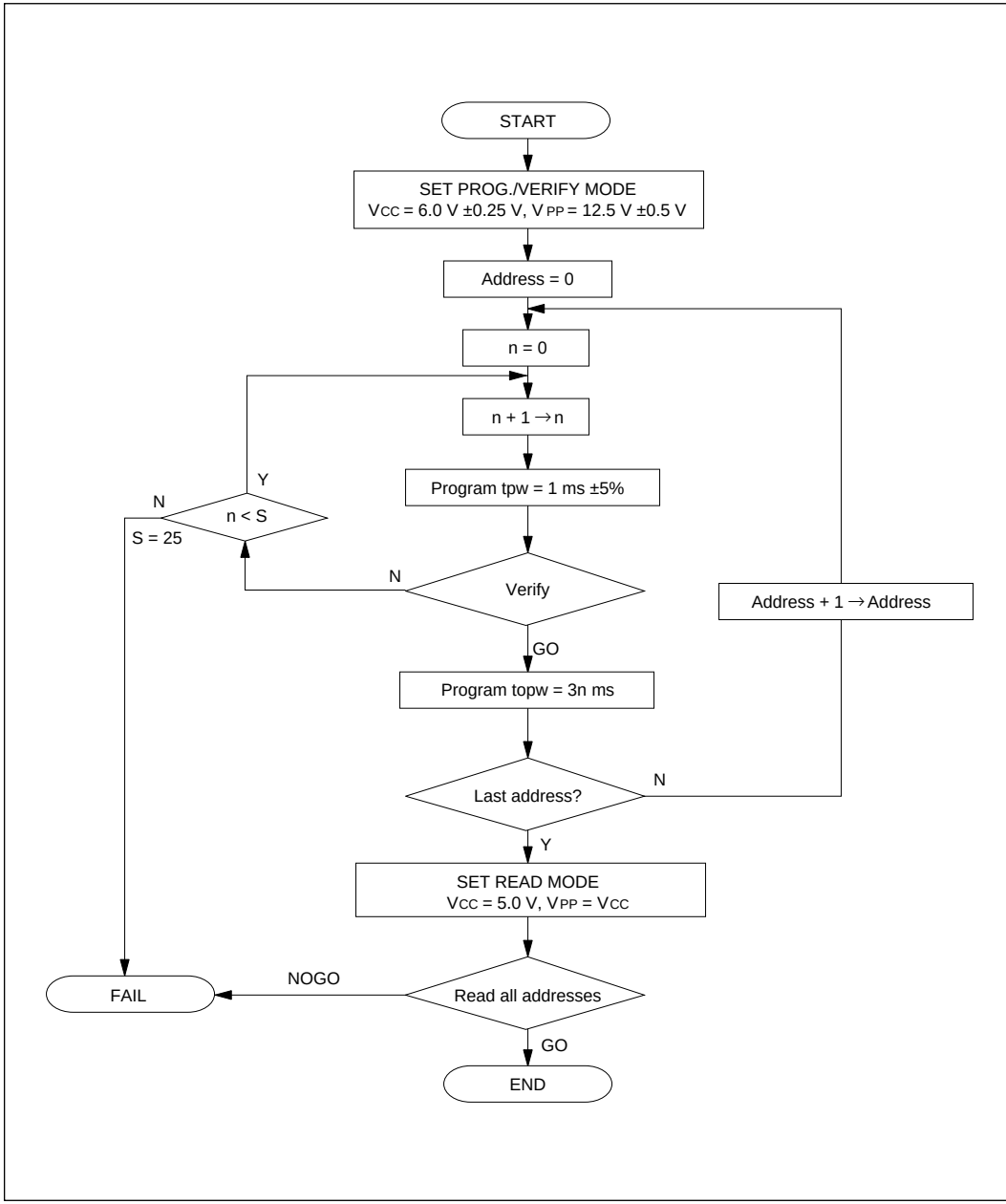


Figure 17-4 High-Speed Programming Flowchart (H8/534)

Table 17-5 DC Characteristics (H8/534)
(When V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, V_{SS} = 0 V, Ta = 25˚C ±5˚C)

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Input High voltage	O_7 to O_0, A_{14} to A_0, $\overline{OE}$, $\overline{CE}$	V_{IH}	2.4	—	V_{CC} + 0.3	V	
Input Low voltage	O_7 to O_0, A_{14} to A_0, $\overline{OE}$, $\overline{CE}$	V_{IL}	–0.3	—	0.8	V	
Input High voltage	O_7 to O_0	V_{OH}	2.4	—	—	V	I_{OH} = –200 µA
Input Low voltage	O_7 to O_0	V_{OL}	—	—	0.45	V	I_{OL} = 1.6 mA
Input leakage current	O_7 to O_0, A_{14} to A_0, $\overline{OE}$, $\overline{CE}$	$\|I_{LI}\|$	—	—	2	µA	V_{in} = 5.25 V/0.5 V
V_{CC} current		I_{CC}	—	—	40	mA	
V_{PP} current		I_{PP}	—	—	40	mA	

Table 17-6 AC Characteristics (H8/534)
(When V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, Ta = 25˚C ±5˚C)

Item	Symbol	Min	Typ	Max	Unit	Measurement Conditions
Address setup time	t_{AS}	2	—	—	µs	See figure
$\overline{OE}$ setup time	t_{OES}	2	—	—	µs	17-5*
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	
V_{PP} setup time	t_{VPS}	2	—	—	µs	
Program pulse width	t_{PW}	0.95	1.0	1.05	ms	
$\overline{OE}$ pulse width for overwrite-programming	t_{OPW}	2.85	—	78.75	ms	
V_{CC} setup time	t_{VCS}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	500	ns	

* Input pulse level: 0.8 V to 2.2 V
Input rise/fall time ≤ 20 ns
Timing reference levels: input—1.0 V, 2.0 V; output—0.8 V, 2.0 V

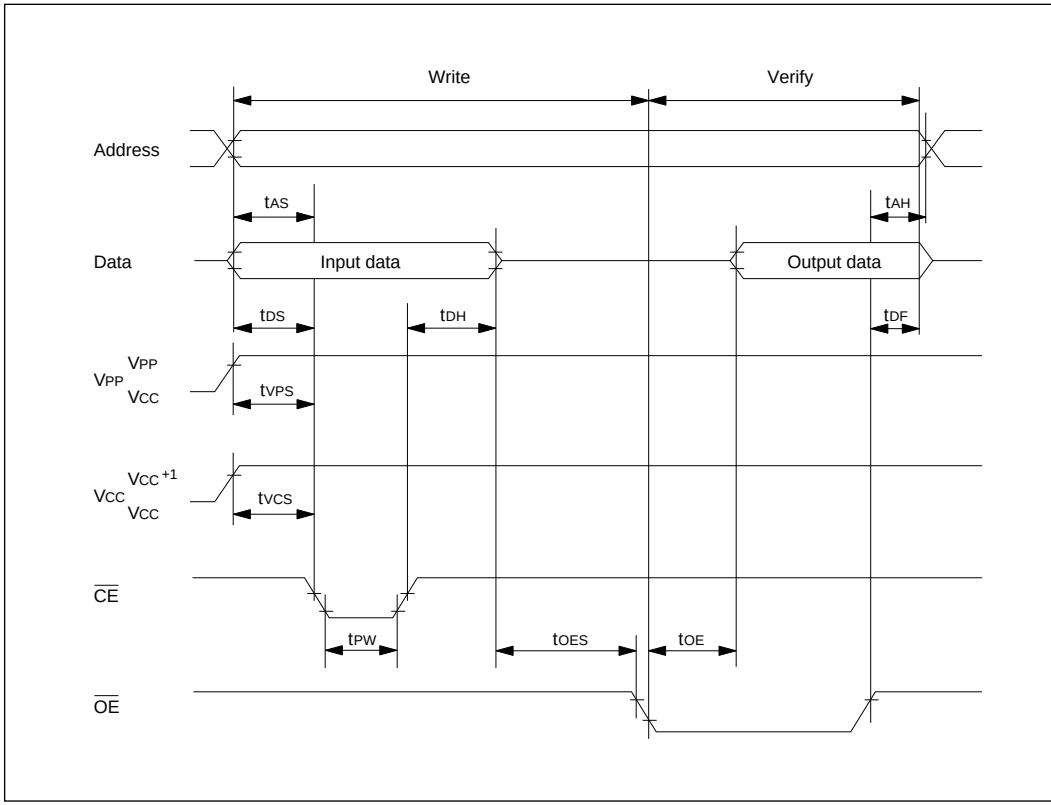


Figure 17-5 PROM Write/Verify Timing (H8/534)

17.3.2 Notes on Writing

1. **Write with the specified voltages and timing. The programming voltage (V_{PP}) in the PROM mode is 12.5 V.**

Caution: Applied voltages in excess of the specified values can permanently destroy to 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, Vpp will be 12.5 V.

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.

17.4 H8/536 Programming

The write, verify, and other sub-modes of PROM mode are selected as shown in table 17-7.

Table 17-7 Selection of Sub-Modes in PROM Mode (H8/536)

	Pins						
Mode	**$\overline{CE}$**	**$\overline{OE}$**	**$\overline{PGM}$**	**V_{PP}**	**V_{CC}**	**O_7 to O_0**	**A_{16} to A_0**
Write	Low	High	Low	V_{PP}	V_{CC}	Data input	Address input
Verify	Low	Low	High	V_{PP}	V_{CC}	Data output	Address input
Programming inhibited	Low	Low	Low	V_{PP}	V_{CC}	High-impedance	Address input
	Low	High	High				
	High	Low	Low				
	High	High	High				

Note: The V_{PP} and V_{CC} pins must be held at the V_{PP} and V_{CC} voltage levels.

Standard EPROM read/write specifications are used, the same as for the HN27C101. The HN27C101 has two programming modes: page programming and byte programming.
The H8/536 does not support page programming, so select byte programming.

17.4.1 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 17-6 shows the basic high-speed programming flowchart.

Tables 17-8 and 17-9 list the electrical characteristics of the chip during programming.
Figure 17-7 shows a timing diagram.

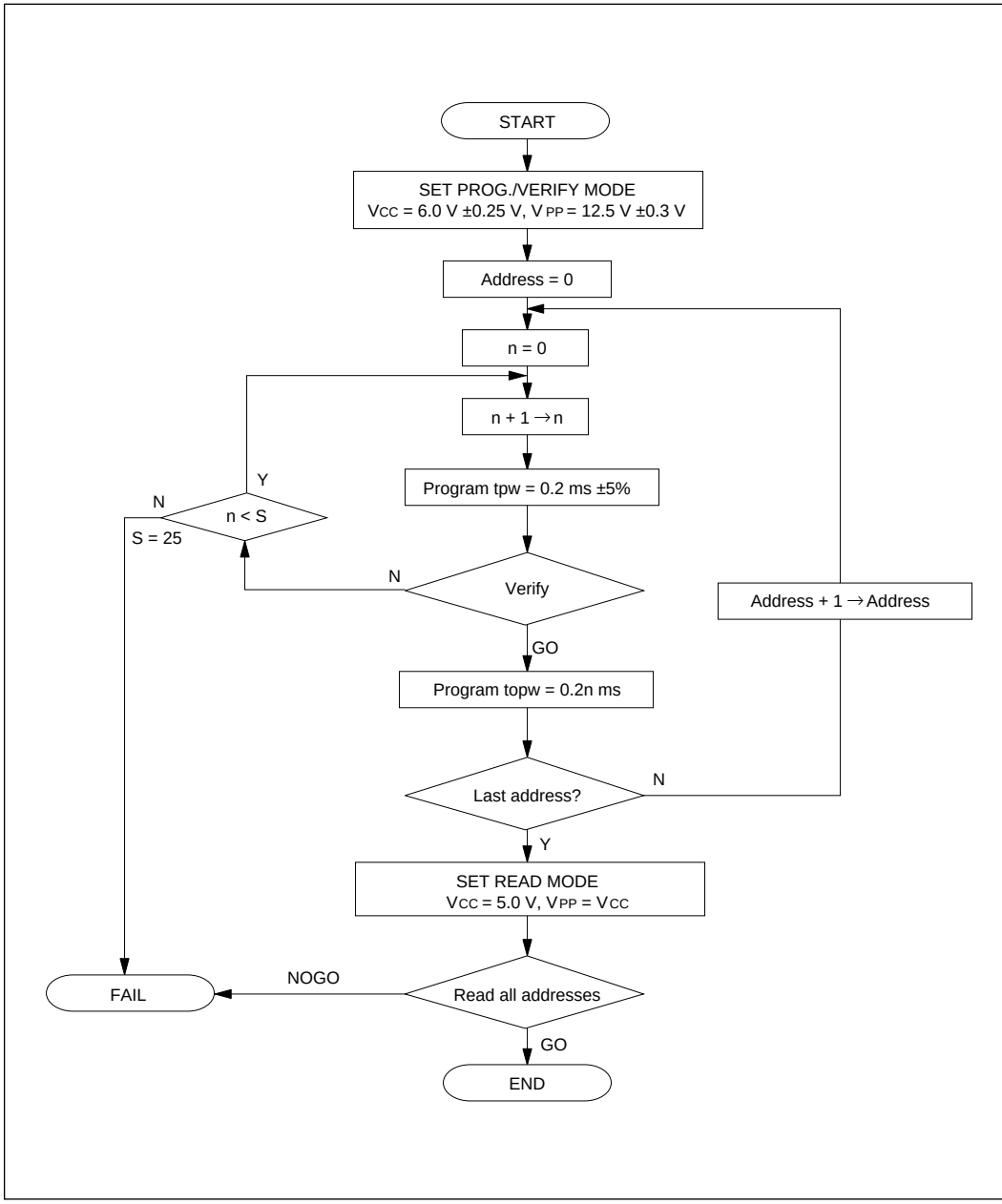


Figure 17-6 High-Speed Programming Flowchart (H8/536)

Table 17-8 DC Characteristics (H8/536)
(When V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, V_{SS} = 0 V, Ta = 25˚C ±5˚C)

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input high voltage	O_7 to O_0, A_{16} to A_0, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	V_{IH}	2.4	—	V_{CC} + 0.3	V	
Input low voltage	O_7 to O_0, A_{16} to A_0, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	V_{IL}	–0.3	—	0.8	V	
Output high voltage	O_7 to O_0	V_{OH}	2.4	—	—	V	I_{OH} = –200 µA
Output low voltage	O_7 to O_0	V_{OL}	—	—	0.45	V	I_{OL} = 1.6 mA
Input leakage current	O_7 to O_0, A_{16} to A_0, $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	$\|I_{LI}\|$	—	—	2	µA	V_{in} = 5.25 V/ 0.5 V
V_{CC} current		I_{CC}	—	—	40	mA	
V_{PP} current		I_{PP}	—	—	40	mA	

Table 17-9 AC Characteristics (H8/536)
(When V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, Ta = 25˚C ±5˚C)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Address setup time	t_{AS}	2	—	—	µs	See figure
$\overline{OE}$ setup time	t_{OES}	2	—	—	µs	17-7*
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	
V_{PP} setup time	t_{VPS}	2	—	—	µs	
Program pulse width	t_{PW}	0.19	0.20	0.21	ms	
$\overline{OE}$ pulse width for overwrite-programming	t_{OPW}	0.19	—	5.25	ms	
V_{CC} setup time	t_{VCS}	2	—	—	µs	
$\overline{OE}$ setup time	t_{CES}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	150	ns	

* Input pulse level: 0.8 V to 2.2 V
Input rise/fall time ≤ 20 ns
Timing reference levels: input—1.0 V, 2.0 V; output—0.8 V, 2.0 V

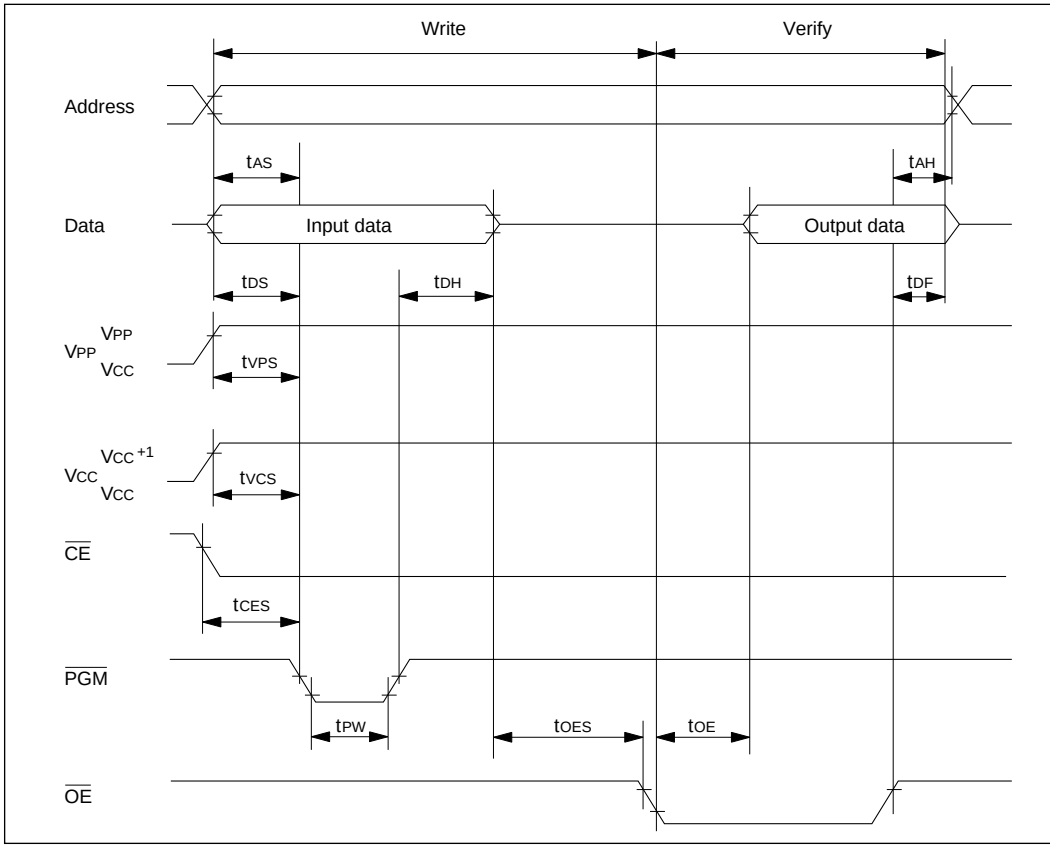


Figure 17-7 PROM Write/Verify Timing (H8/536)

17.4.2 Notes on Programming

1. Program with the specified voltages and timing.
The programming voltage (V_{PP}) in PROM mode is 12.5 V.

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 Hitachi HN27C101 specifications, V_{PP} will be 12.5 V.

2. Before programming, 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 programming.** Touching either of these can cause contact faults and write errors.

4. **The H8/536 uses the HN27C101’s byte programming mode.** Note that some PROM writers do not support the HN27C101’s byte programming mode. Table 17-10 lists the PROM writers recommended for use with the HD6475368R.

Table 17-10 PROM Writers

Recommended PROM Writers

Vendor	Model	
Data I/O	29B + Unipak 2B	V21.0*
	212	V2.0*
	288A	V4.1*
	SI000	V15.0*
	UNISITE 40	V3.0*
	2900	V1.0*
Aval Data	PKW-3100	
	PKW-1100	
Minato Electronics	Model 1892 80-pin QFP type: 84-pin PLCC type:	 GA91-15 GA91-16
	Model 1891 80-pin QFP type: 84-pin PLCC type:	 GA91-15 GA91-16

Note: * Use PROM writers with the indicated or higher version numbers.

5. **The H8/536 PROM size is 62 kbytes.** When programming, leave data H'FF in addresses H'F680 to H'1FFFF.

17.5 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 17-8 shows the recommended screening procedure.

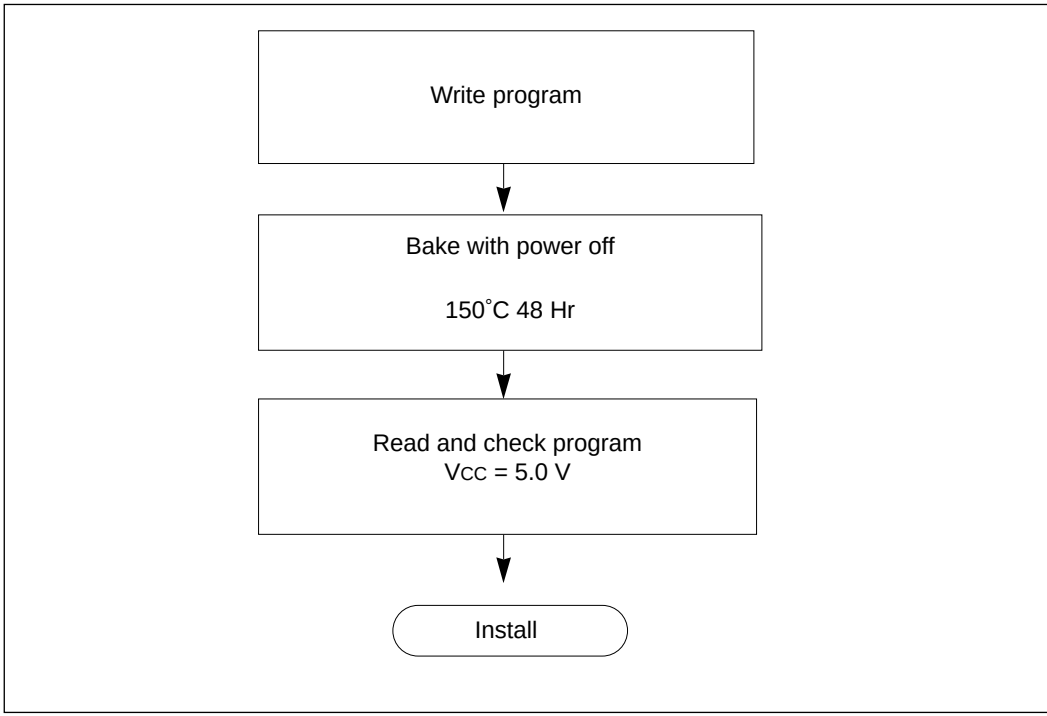


Figure 17-8 Recommended Screening Procedure

If a series of write errors occur while the same PROM writer is in use, stop programming and check the PROM writer and socket adapter for defects, using a microcomputer 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.

17.6 Erasing of Data

The windowed package enables data to be erased by illuminating the window with ultraviolet light. Table 17-11 lists the erasing conditions.

Table 17-11 Erasing Conditions

Item	Value
Ultraviolet wavelength	253.7 nm
Minimum illumination	15 W·s/cm^2

The conditions in table 17-11 can be satisfied by placing a 12000-µW/cm^2 ultraviolet lamp 2 or 3 centimeters directly above the chip and leaving it on for about 20 minutes.

17.7 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. **84-Pin LCC Package Mounting:** When mounted on a printed circuit board, the 84-pin LCC package must be mounted in a socket. The recommended socket is listed in table 17-12.

Table 17-12 Socket for 84-Pin LCC Package

Manufacturer	Product Code
Sumitomo 3-M	284-1273-00-1102J

Section 18 Power-Down State

18.1 Overview

The H8/534 and H8/536 have a power-down state that greatly reduces power consumption by stopping the CPU 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

The sleep mode and software standby mode are entered from the program execution state by executing the SLEEP instruction under the conditions given in table 18-1. The hardware standby mode is entered from any other state by a Low input at the $\overline{\text{STBY}}$ pin.

Table 18-1 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 18-1 Power-Down State

Mode	Entering Procedure	Clock	CPU	CPU Reg's.	Sup. Mod's.	RAM	I/O Ports	Exiting Methods
Sleep mode	Execute SLEEP instruction	Run	Halt	Held	Run	Held	Held	• Interrupt • $\overline{\text{RES}}$ Low • $\overline{\text{STBY}}$ Low
Software standby mode	Set SSBY bit in SBYCR to 1, then execute SLEEP instruction*	Halt	Halt	Held	Halt and initialized	Held	Held	• NMI • $\overline{\text{RES}}$ Low • $\overline{\text{STBY}}$ Low
Hardware standby mode	Set $\overline{\text{STBY}}$ pin to Low level	Halt	Halt	Not held	Halt and initialized	Held	High impedance state	• $\overline{\text{STBY}}$ High, then $\overline{\text{RES}}$ Low → High

* The watchdog timer must also be stopped.

Notes: SBYCR Software standby control register
SSBY Software standby bit

18.2 Sleep Mode

18.2.1 Transition to Sleep Mode

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 functions of the on-chip supporting modules do not stop in the sleep mode.

18.2.2 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 either the CPU's interrupt-handling sequence or the data transfer controller (DTC).

   If the interrupt is served by the DTC, after the data transfer is completed the CPU executes the instruction following the SLEEP instruction, unless the count in the data transfer count register (DTCR) is 0.

   If an interrupt on a level equal to or less than the mask level in the CPU's status register (SR) is requested, the interrupt is left pending and the sleep mode continues. Also, 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.

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.

18.3 Software Standby Mode

18.3.1 Transition to Software Standby Mode

A program enters the software standby mode by setting the standby bit (SSBY) in the software standby control register (SBYCR) to 1, then executing the SLEEP instruction. Table 18-2 lists the attributes of the software standby control register.

Table 18-2 Software Standby Control Register

Name	Abbreviation	R/W	Initial Value	Address
Software standby control register	SBYCR	R/W	H'7F	H'FF13

In the software standby mode, the CPU, clock, and the on-chip supporting module functions all stop, reducing power consumption to an extremely low level. The on-chip supporting modules and their registers are reset to their initial state, but as long as a minimum necessary voltage supply is maintained (at least 2 V), the contents of the CPU registers and on-chip RAM remain unchanged. The I/O ports also remain in their current states.

18.3.2 Software Standby Control Register (SBYCR)

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

The software standby control register (SBYCR) is an 8-bit register that controls the action of the SLEEP instruction.

Bit 7—Software Standby (SSBY): This bit enables or disables the transition to the software standby mode.

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.

The watchdog timer must be stopped before the chip can enter the software standby mode. To stop the watchdog timer, clear the timer enable bit (TME) in the watchdog timer's timer control/status register (TCSR) to 0. The SSBY bit cannot be set to 1 while the TME bit is set to 1.

When the chip is recovered from the software standby mode by a nonmaskable interrupt (NMI), the SSBY bit is automatically cleared to 0. It is also cleared to 0 by a reset or transition to the hardware standby mode.

Bits 6 to 0—Reserved: These bits cannot be modified and are always read as 1.

18.3.3 Exit from Software Standby Mode

The chip can be brought out of the software standby mode by an input at one of three pins: the NMI pin, $\overline{\text{RES}}$ pin, or $\overline{\text{STBY}}$ pin.

1. **Recovery by NMI Pin:** When an NMI request signal is received, the clock oscillator begins operating but clock pulses are supplied only to the watchdog timer (WDT). The watchdog timer begins counting from H'00 at the rate determined by the clock select bits (CKS2 to CKS0) in its timer status/control register (TCSR). This rate should be set slow enough to allow the clock oscillator to stabilize before the count reaches H'FF. When the count overflows from H'FF to H'00, clock pulses are supplied to the whole chip, the software standby mode ends, and execution of the NMI interrupt-handling sequence begins.

   The clock select bits (CKS2 to CKS0) should be set as follows.

   (1) Crystal oscillator: Set CKS2 to CKS0 to a value that makes the watchdog timer interval equal to or greater than 10ms, which is the clock stabilization time.

   (2) External clock input: CKS2 to CKS0 can be set to any value. The minimum value (CKS2 = CKS1 = CKS0 = 0) is recommended.

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.

   When the chip recovers from the software standby mode by a reset, clock pulses are supplied to the entire chip at once. Be sure to hold the $\overline{\text{RES}}$ pin Low long enough for the clock to stabilize.

3. **Recovery by $\overline{\text{STBY}}$ Pin:** When $\overline{\text{STBY}}$ the pin goes Low, the chip exits from the software standby mode to the hardware standby mode.

18.3.4 Sample Application of Software Standby Mode

In this example the chip enters the software standby mode on the falling edge of the NMI input and recovers from the software standby mode on the rising edge of NMI. Figure 18-1 shows a timing chart of the transitions.

The nonmaskable interrupt edge bit (NMIEG) in the port 1 control register (P1CR) is originally cleared to 0, selecting the falling edge as the NMI trigger. After accepting an NMI interrupt in this condition, software changes the NMIEG bit to 1, sets the SSBY bit to 1, and executes the SLEEP instruction to enter the software standby mode. The chip recovers from the software standby mode on the next rising edge at the NMI pin.

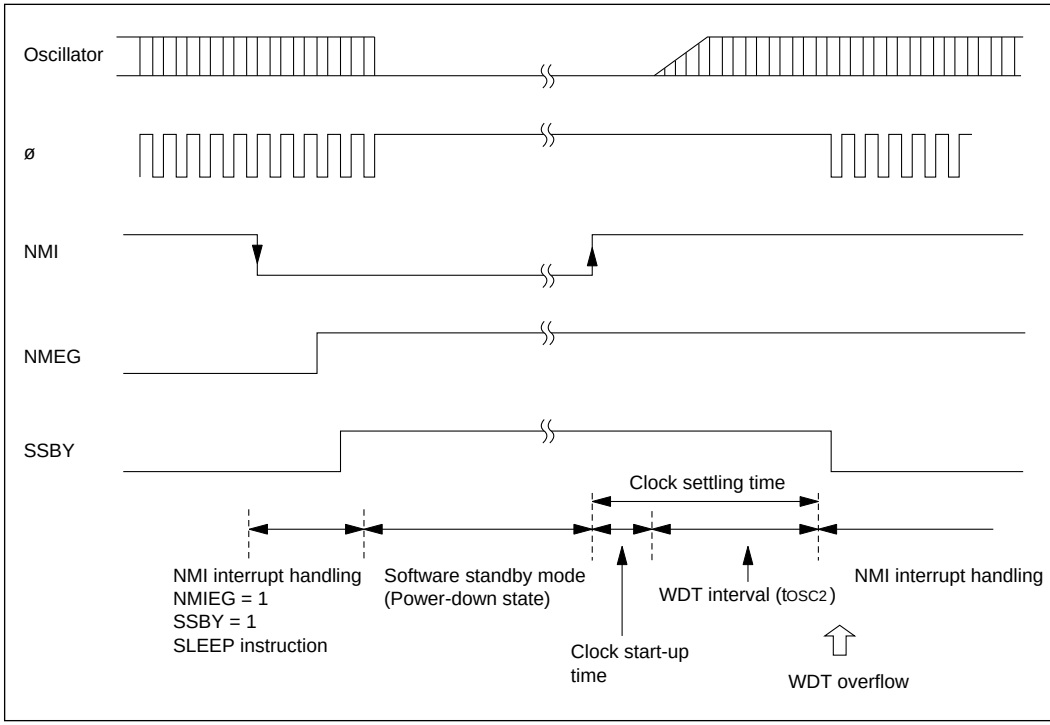


Figure 18-1 NMI Timing of Software Standby Mode (Application Example)

18.3.5 Application Notes

The I/O ports remain in their current states in the software standby mode. If a port is in the High output state, the output current is not reduced in the software standby mode.

18.4 Hardware Standby Mode

18.4.1 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 (see note 1).

Notes: 1. The RAME bit in the RAM 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 (MD2, MD1, MD0) during hardware standby mode. Be particularly careful not to let all three mode inputs go low, since that would place the chip in PROM mode, causing increased current dissipation.

18.4.2 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.

Note: During standby mode, power must still be supplied to AV_{CC}, and the mode pins must be held at the selected mode.

18.4.3 Timing Sequence of Hardware Standby Mode

Figure 18-2 shows the usual sequence for entering and leaving the hardware standby mode.

First the $\overline{\text{RES}}$ pin goes Low, placing the chip in the reset state. Then the $\overline{\text{STBY}}$ pin goes Low, placing the chip in the hardware standby mode and stopping the clock. In the recovery sequence first the $\overline{\text{STBY}}$ pin goes High; then after the clock stabilizes, the $\overline{\text{RES}}$ pin is returned to the High level.

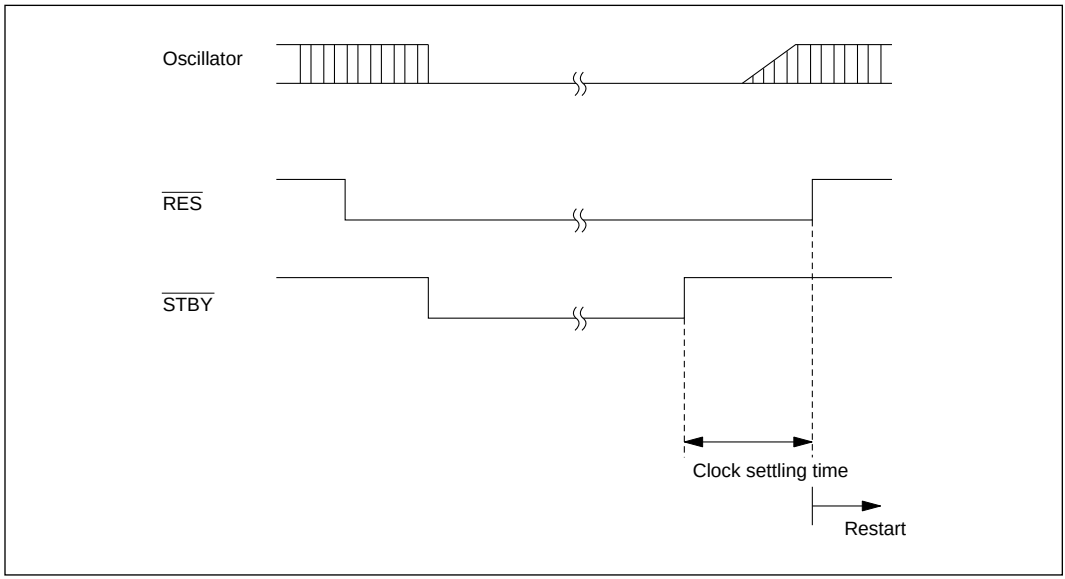


Figure 18-2 Hardware Standby Sequence

Section 19 E Clock Interface

19.1 Overview

For interfacing to E clock based peripheral devices, the H8/534 and H8/536 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 $P1_1$ pin when the $P1_1$DDR bit in the port 1 data direction register (P1DDR) is set to 1.

When the CPU executes an instruction that synchronizes with the E clock, the address is output on the address bus as usual, but the data bus and the R/$\overline{W}$, $\overline{DS}$, $\overline{RD}$, and $\overline{WR}$ signal lines 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 is accordingly variable. Figures 19-1 and 19-2 show the timing in the cases of maximum and minimum synchronization delay.

The wait state controller (WSC) does not insert any wait states (Tw) during the execution of an instruction synchronized with the E clock.

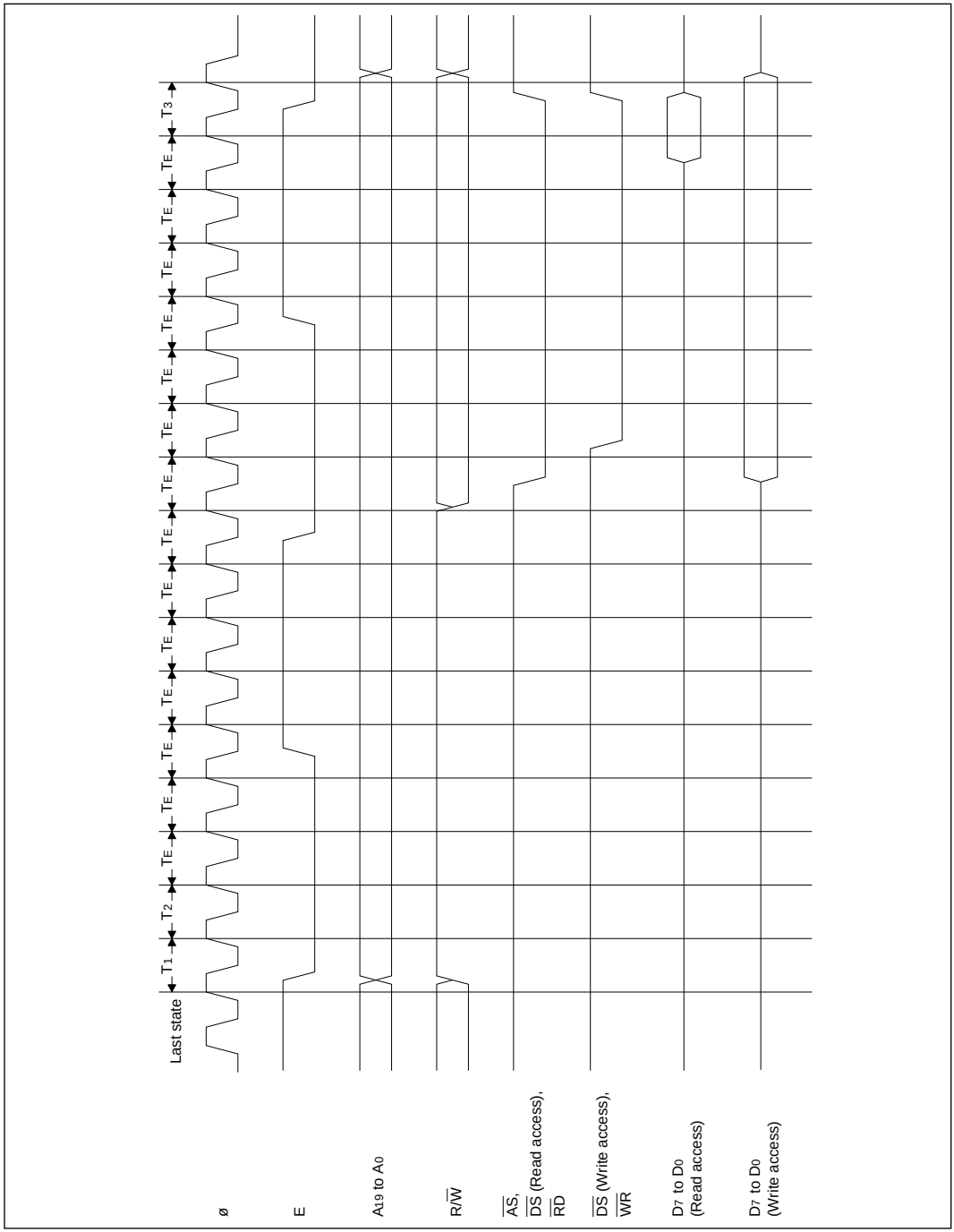


Figure 19-1 Execution Cycle of Instruction Synchronized with E Clock in Expanded Modes (Maximum Synchronization Delay)

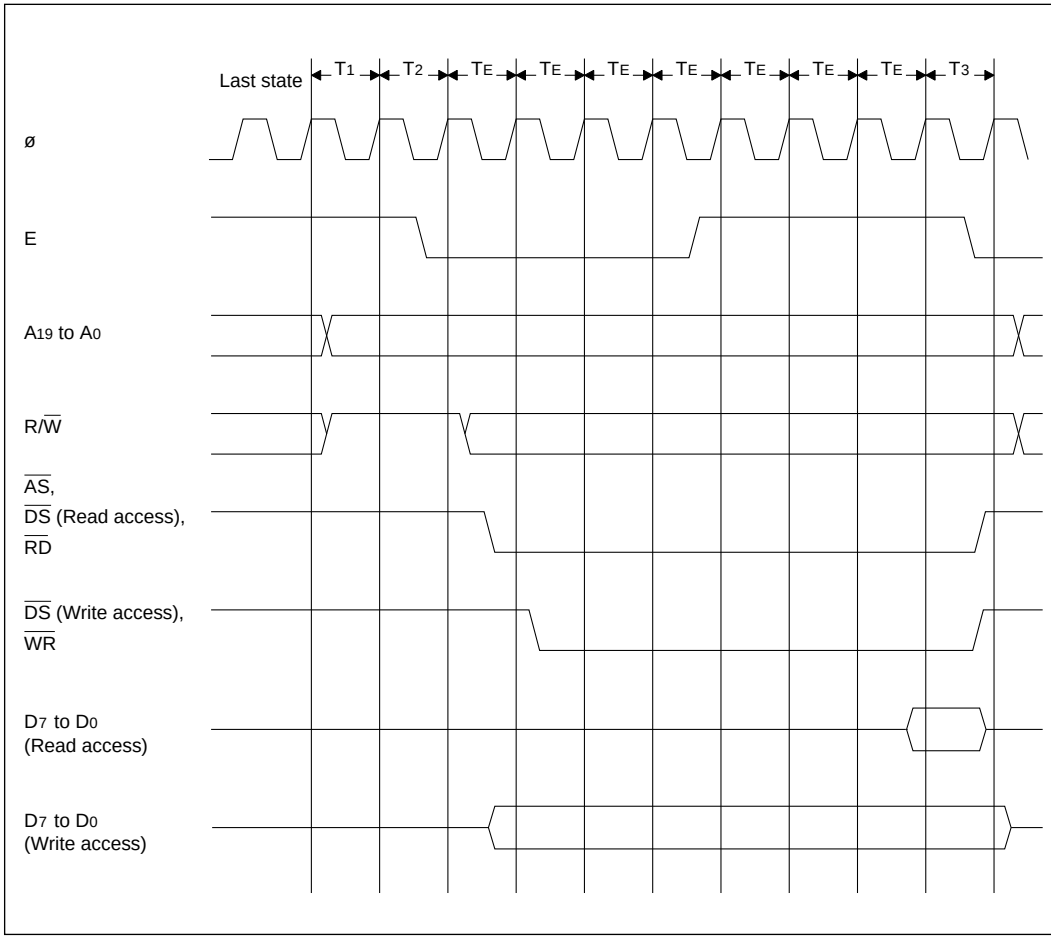


Figure 19-2 Execution Cycle of Instruction Synchronized with E Clock in Expanded Modes (Minimum Synchronization Delay)

Section 20 Electrical Specifications

20.1 Absolute Maximum Ratings

Table 20-1 lists the absolute maximum ratings.

Table 20-1 Absolute Maximum Ratings

Item		Symbol	Rating	Unit
Supply voltage		V_{CC}	–0.3 to +7.0	V
Programming voltage	R-mask	V_{PP}	–0.3 to +13.5	V
	S-mask		–0.3 to +13.0	V
Input voltage	(except Port 8)	V_{in}	–0.3 to V_{CC} + 0.3	V
	(Port 8)	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: Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions.

20.2 Electrical Characteristics

20.2.1 DC Characteristics

Table 20-2 lists the DC characteristics.

Table 20-2 DC Characteristics (5-V Versions) **– Preliminary for S-Mask Versions–**

Conditions: V_{CC} = 5.0 V ±10%, AV_{CC} = 5.0 V ±10%*1, V_{SS} = AV_{SS} = 0 V,
T_a = –20 to +75°C (Regular Specifications)
T_a = –40 to +85°C (Wide-Range Specifications)

Item			Symbol	Min	Typ	Max	Unit	Test Conditions
Input High voltage	$\overline{RES}$, $\overline{STBY}$, MD_2, MD_1, MD_0		V_{IH}	V_{CC} – 0.7	–	V_{CC} + 0.3	V	
	EXTAL			$V_{CC} \times 0.7$	–	V_{CC} + 0.3	V	
	Port 8			2.2	–	AV_{CC} + 0.3	V	
	Other input pins (except port 7)			2.2	–	V_{CC} + 0.3	V	
Input Low voltage	$\overline{RES}$, $\overline{STBY}$, MD_2, MD_1, MD_0		V_{IL}	–0.3	–	0.5	V	
	Other input pins (except port 7)			–0.3	–	0.8	V	
Schmitt trigger input voltage	Port 7		V_{T^-}	1.0	–	2.5	V	
			V_{T^+}	2.0	–	3.5	V	
			$V_{T^+} – V_{T^-}$	0.4	–	–	V	
Input leakage current	$\overline{RES}$		$\lvert I_{in} \rvert$	–	–	10.0	µA	V_{in} = 0.5 to
	$\overline{STBY}$, NMI, MD_2, MD_1, MD_0			–	–	1.0	µA	V_{CC} – 0.5 V
	Port 8			–	–	1.0	µA	V_{in} = 0.5 to AV_{CC} – 0.5 V
Leakage current in 3-state (off state)	Port 9, ports 7 to 1		$\lvert I_{TSI} \rvert$	–	–	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input pull-up MOS current	Ports 6 and 5	R-mask	$-I_P$	50	–	200	µA	V_{in} = 0 V
		S-mask		50	–	300	µA	
Output High voltage	All output pins		V_{OH}	V_{CC} – 0.5	–	–	V	I_{OH} = –200 µA
				3.5	–	–	V	I_{OH} = –1 mA
Output Low voltage	All output pins (except $\overline{RES}$)		V_{OL}	–	–	0.4	V	I_{OL} = 1.6 mA
	Port 4	R-mask		–	–	1.0	V	I_{OL} = 8 mA
				–	–	1.2	V	I_{OL} = 10 mA
		S-mask		–	–	1.0	V	I_{OL} = 10 mA
	$\overline{RES}$			–	–	0.4	V	I_{OL} = 2.6 mA

Note: *1 AVcc must be connected to a power supply line, even when the A/D converter is not used and even in standby mode.

Item			Symbol	Min	Typ	Max	Unit	Test Conditions
Input capacitance	$\overline{RES}$	H8/534	C_{in}	–	–	60	pF	V_{in} = 0 V
		H8/536		–	–	100	pF	f = 1 MHz
	NMI	R-mask		–	–	30	pF	T_a = 25°C
		S-mask		–	–	50	pF	
	All input pins except $\overline{RES}$, NMI			–	–	15	pF	
Current dissipation*2	Normal operation	R-mask	I_{CC}	–	25	40	mA	f = 6 MHz
				–	30	50	mA	f = 8 MHz
				–	35	60	mA	f = 10 MHz
		S-mask		–	40	60	mA	f = 16 MHz
	Sleep mode	R-mask		–	12	25	mA	f = 6 MHz
				–	16	30	mA	f = 8 MHz
				–	20	35	mA	f = 10 MHz
		S-mask		–	23	35	mA	f = 16 MHz
	Standby			–	0.01	5.0	µA	$T_a \leq 50°C$
				–	–	20.0	µA	$T_a > 50°C$
Analog supply current	During A/D conversion	R-mask	AI_{CC}	–	1.2	2.0	mA	
		S-mask		–	1.5	3.0	mA	
	While waiting			–	0.01	5.0	µA	
RAM standby voltage			V_{RAM}	2.0	–	–	V	

Note: *2 Current dissipation values assume that V_{IH} min = V_{CC} – 0.5 V, V_{IL} max = 0.5 V, all output pins are in the no-load state, and all MOS input pull-ups are off.

Table 20-3 DC Characteristics (3-V S-Mask Versions) –Preliminary–

Conditions: V_{CC} = 3.0 to 5.5 V, V_{SS} = AV_{SS} = 0 V, T_a = –20 to +75°C (Regular Specifications), AV_{CC} = 3.0 to 5.5 V*1

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input High voltage	$\overline{RES}$, $\overline{STBY}$, MD_2 to MD_0	V_{IH}	$V_{CC} \times 0.85$	–	V_{CC} + 0.3	V	
	EXTAL		$V_{CC} \times 0.7$	–	V_{CC} + 0.3	V	
	Port 8		2.2	–	AV_{CC} + 0.3	V	
	Other input pins (except port 7)		2.2	–	V_{CC} + 0.3	V	
Input Low voltage	$\overline{RES}$, $\overline{STBY}$, MD_2 to MD_0, EXTAL	V_{IL}	–0.3	–	0.4	V	
	Other input pins (except port 7)		–0.3	–	0.8	V	$V_{CC} \geq 4.0$ V
			–0.3	–	$V_{CC} \times 0.2$	V	V_{CC} < 4.0 V
Schmitt trigger input voltages	Port 7	V_{T}^{-}	$V_{CC} \times 0.2$	–	$V_{CC} \times 0.5$	V	
		V_{T}^{+}	$V_{CC} \times 0.4$	–	$V_{CC} \times 0.7$	V	
		$V_{T}^{+} - V_{T}^{-}$	$V_{CC} \times 0.07$	–	–	V	
Input leakage current	$\overline{RES}$	$\lvert I_{in} \rvert$	–	–	10.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
	$\overline{STBY}$, NMI, MD_2, MD_1, MD_0		–	–	1.0	µA	
	Port 8		–	–	1.0	µA	V_{in} = 0.5 to AV_{CC} – 0.5 V
Leakage current in 3-state (off-state)	Port 9, ports 7 to 1	$\lvert I_{TSI} \rvert$	–	–	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input pull-up MOS current	Ports 6 and 5	$-I_P$	15	–	300	µA	V_{in} = 0 V
Output High voltage	All output pins	V_{OH}	V_{CC} – 0.4	–	–	V	I_{OH} = –200 µA
			V_{CC} – 1.0	–	–	V	I_{OH} = –1 mA

Note: *1 AV_{CC} must be connected to a power supply line, even when the A/D converter is not used, and even in standby mode.

Item			Symbol	Min	Typ	Max	Unit	Test Conditions
Output Low voltage	All output pins (except $\overline{RES}$)		V_{OL}	–	–	0.4	V	I_{OL} = 1.6 mA
	Port 4			–	–	1.0	V	I_{OL} = 5 mA
	$\overline{RES}$			–	–	0.4	V	I_{OL} = 1.6 mA
Input capacitance	$\overline{RES}$	H8/534	C_{in}	–	–	60	pF	V_{in} = 0 V f = 1 MHz T_a = 25°C
		H8/536				100		
	NMI			–	–	50	pF	
	All input pins except $\overline{RES}$ and NMI			–	–	15	pF	
Current dissipation*2	Normal operation		I_{CC}	–	27	40	mA	f = 10 MHz, V_{CC} = 5 V
				–	17	25	mA	f = 10 MHz, V_{CC} = 3 V
	Sleep mode			–	15	25	mA	f = 10 MHz, V_{CC} = 5 V
				–	10	15	mA	f = 10 MHz, V_{CC} = 3 V
	Standby			–	0.01	5.0	µA	$T_a \leq 50°C$
				–	–	20.0	µA	50°C < T_a
Analog supply current	During A/D conversion		AI_{CC}	–	1.5	3.0	mA	AV_{CC} = 5 V
				–	0.5	1.0	mA	AV_{CC} = 3 V
	While waiting			–	0.01	5.0	µA	
RAM standby voltage			V_{RAM}	2.0	–	–	V	

Note: *2 Current dissipation values assume that V_{IH} min = V_{CC} – 0.5 V and V_{IL} max = 0.5 V, all output pins are in the no-load state, and all MOS input pull-ups are off.

Table 20-4 DC Characteristics (2.7-V S-Mask Versions) –Preliminary–

Conditions: V_{CC} = 2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V, T_a = –20 to +75°C (Regular Specifications), AV_{CC} = 2.7 to 5.5 V*1

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input High voltage	$\overline{RES}$, $\overline{STBY}$, MD_2 to MD_0	V_{IH}	$V_{CC} \times 0.85$	–	V_{CC} + 0.3	V	
	EXTAL		$V_{CC} \times 0.7$	–	V_{CC} + 0.3	V	
	Port 8		2.2	–	AV_{CC} + 0.3	V	
	Other input pins (except port 7)		2.2	–	V_{CC} + 0.3	V	
Input Low voltage	$\overline{RES}$, $\overline{STBY}$, MD_2 to MD_0, EXTAL	V_{IL}	–0.3	–	0.4	V	
	Other input pins (except port 7)		–0.3	–	0.8	V	$V_{CC} \geq 4.0$ V
			–0.3	–	$V_{CC} \times 0.2$	V	$V_{CC} < 4.0$ V
Schmitt trigger input voltages	Port 7	V_{T^-}	$V_{CC} \times 0.2$	–	$V_{CC} \times 0.5$	V	
		V_{T^+}	$V_{CC} \times 0.4$	–	$V_{CC} \times 0.7$	V	
		$V_{T^+} - V_{T^-}$	$V_{CC} \times 0.07$	–	–	V	
Input leakage current	$\overline{RES}$	$\lvert I_{in} \rvert$	–	–	10.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
	$\overline{STBY}$, NMI, MD_2, MD_1, MD_0		–	–	1.0	µA	
	Port 8		–	–	1.0	µA	V_{in} = 0.5 to AV_{CC} – 0.5 V
Leakage current in 3-state (off-state)	Port 9, ports 7 to 1	$\lvert I_{TSI} \rvert$	–	–	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input pull-up MOS current	Ports 6 and 5	$-I_P$	15	–	300	µA	V_{in} = 0 V
Output High voltage	All output pins	V_{OH}	V_{CC} – 0.4	–	–	V	I_{OH} = –200 µA
			V_{CC} – 1.0	–	–	V	I_{OH} = –1 mA

Note: *1 AV_{CC} must be connected to a power supply line, even when the A/D converter is not used, and even in standby mode.

Item			Symbol	Min	Typ	Max	Unit	Test Conditions
Output Low voltage	All output pins (except $\overline{RES}$)		V_{OL}	–	–	0.4	V	I_{OL} = 1.6 mA
	Port 4			–	–	1.0	V	I_{OL} = 5 mA
	$\overline{RES}$			–	–	0.4	V	I_{OL} = 1.6 mA
Input capacitance	$\overline{RES}$	H8/534	C_{in}	–	–	60	pF	V_{in} = 0 V
		H8/536				100		f = 1 MHz
	NMI			–	–	50	pF	T_a = 25°C
	All input pins except $\overline{RES}$ and NMI			–	–	15	pF	
Current dissipation*2	Normal operation		I_{CC}	–	23	35	mA	f = 8 MHz, V_{CC} = 5 V
				–	14	22	mA	f = 8 MHz, V_{CC} = 3 V
	Sleep mode			–	12	22	mA	f = 8 MHz, V_{CC} = 5 V
				–	8	14	mA	f = 8 MHz, V_{CC} = 3 V
	Standby			–	0.01	5.0	µA	$T_a \leq 50°C$
				–	–	20.0	µA	50°C < T_a
Analog supply current	During A/D conversion		AI_{CC}	–	1.5	3.0	mA	AV_{CC} = 5 V
				–	0.5	1.0	mA	AV_{CC} = 3 V
	While waiting			–	0.01	5.0	µA	
RAM standby voltage			V_{RAM}	2.0	–	–	V	

Note: *2 Current dissipation values assume that V_{IH} min = V_{CC} – 0.5 V and V_{IL} max = 0.5 V, all output pins are in the no-load state, and all MOS input pull-ups are off.

Table 20-5 Allowable Output Current Values (5-V Versions) – Preliminary for S-Mask Versions–

Conditions: V_{CC} = 5.0 V ±10%, AV_{CC} = 5.0 V ±10%, V_{SS} = AV_{SS} = 0 V,
T_a = –20 to +75°C (Regular Specifications)
T_a = –40 to +85°C (Wide-Range Specifications)

Item		Symbol	Min	Typ	Max	Unit
Allowable output Low current (per pin)	Port 4	I_{OL}	–	–	10	mA
	$\overline{RES}$		–	–	3.0	mA
	Other output pins		–	–	2.0	mA
Allowable output Low current (total)	Port 4, total of 8 pins	ΣI_{OL}	–	–	40	mA
	Total of all output pins		–	–	80	mA
Allowable output High current (per pin)	All output pins	$-I_{OH}$	–	–	2.0	mA
Allowable output High current (total)	Total of all output pins	$\Sigma -I_{OH}$	–	–	25	mA

Table 20-6 Allowable Output Current Values (3-V S-Mask Versions) –Preliminary–

Conditions: V_{CC} = 3.0 to 5.5 V, V_{SS} = AV_{SS} = 0 V, T_a = –20 to +75°C (Regular Specifications),
AV_{CC} = 3.0 to 5.5 V*1

Item		Symbol	Min	Typ	Max	Unit
Allowable output Low current (per pin)	Port 4	I_{OL}	–	–	10	mA
	$\overline{RES}$		–	–	3.0	mA
	Other output pins		–	–	2.0	mA
Allowable output Low current (total)	Port 4, total of 8 pins	ΣI_{OL}	–	–	40	mA
	Total of all output pins		–	–	80	mA
Allowable output High current (per pin)	All output pins	$-I_{OH}$	–	–	2.0	mA
Allowable output High current (total)	Total of all output pins	$\Sigma -I_{OH}$	–	–	25	mA

Note: *1 To avoid degrading the reliability of the chip, be careful not to exceed the output current sink values in table 20-5. 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 20-1 and 20-2.

Table 20-7 Allowable Output Current Values (2.7-V S-Mask Versions) –Preliminary–

Conditions: V_{CC} = 2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V, T_a = –20 to +75˚C (Regular Specifications), AV_{CC} = 2.7 to 5.5 V*1

Item		Symbol	Min	Typ	Max	Unit
Allowable output Low current (per pin)	Port 4	I_{OL}	–	–	10	mA
	$\overline{RES}$		–	–	3.0	mA
	Other output pins		–	–	2.0	mA
Allowable output Low current (total)	Port 4, total of 8 pins	ΣI_{OL}	–	–	40	mA
	Total of all output pins		–	–	80	mA
Allowable output High current (per pin)	All output pins	$-I_{OH}$	–	–	2.0	mA
Allowable output High current (total)	Total of all output pins	$\Sigma -I_{OH}$	–	–	25	mA

Note: *1 To avoid degrading the reliability of the chip, be careful not to exceed the output current sink values in table 20-5. 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 20-1 and 20-2.

The S-mask versions (high-speed and low-voltage versions) are identical to the existing R-mask versions functionally and in their pin arrangement. Due to the higher-speed design, however, there are differences in the fabrication process, which lead to some differences in electrical specifications, operating margin, noise margin, and other characteristics. These differences should be noted during board design, and when switching from an R-mask to an S-mask version.

–Preliminary–

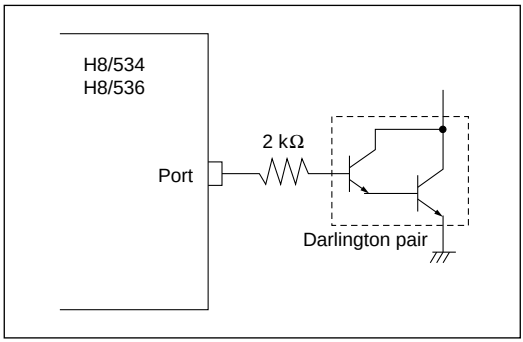


Figure 20-1 Example of Circuit for Driving a Darlington Transistor Pair

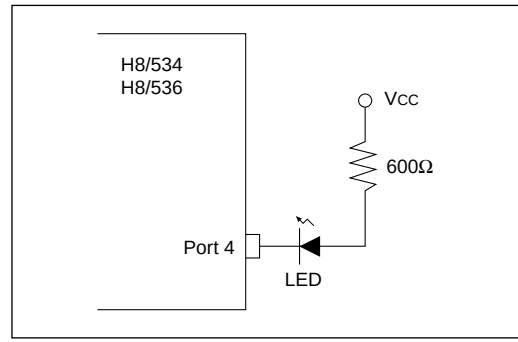


Figure 20-2 Example of Circuit for Driving an LED

20.2.2 AC Characteristics

The AC characteristics of the H8/534 and H8/536 are listed in three tables. Bus timing parameters are given in table 20-8, control signal timing parameters in table 20-9, and timing parameters of the on-chip supporting modules in table 20-10.

Table 20-8 (1) Bus Timing (R-Mask Versions)

Condition A (R-mask): V_{CC} = 5.0 V ±10%, ø = 0.5 to 10 MHz, V_{SS} = 0 V
T_a = –20 to +75˚C (Regular Specifications)
T_a = –40 to +85˚C (Wide-Range Specifications)

		Condition A							
		6 MHz		8 MHz		10 MHz			Test
Item	Symbol	Min	Max	Min	Max	Min	Max	Unit	Conditions
Clock cycle time	t_{cyc}	166.7	2000	125	2000	100	2000	ns	See figure 20-4
Clock pulse width Low	t_{CL}	65	–	45	–	35	–	ns	
Clock pulse width High	t_{CH}	65	–	45	–	35	–	ns	
Clock rise time	t_{Cr}	–	15	–	15	–	15	ns	
Clock fall time	t_{Cf}	–	15	–	15	–	15	ns	
Address delay time	t_{AD}	–	70	–	60	–	55	ns	
Address hold time	t_{AH}	30	–	25	–	20	–	ns	
Data strobe delay time 1	t_{DSD1}	–	70	–	60	–	40	ns	
Data strobe delay time 2	t_{DSD2}	–	70	–	60	–	50	ns	
Data strobe delay time 3	t_{DSD3}	–	70	–	60	–	50	ns	
Write data strobe pulse width	t_{DSWW}	200	–	150	–	120	–	ns	
Address setup time 1	t_{AS1}	25	–	20	–	15	–	ns	
Address setup time 2	t_{AS2}	105	–	80	–	65	–	ns	
Read data setup time	t_{RDS}	60	–	50	–	40	–	ns	
Read data hold time	t_{RDH}	0	–	0	–	0	–	ns	
Read data access time	t_{ACC}	–	280	–	190	–	160	ns	
Write data delay time	t_{WDD}	–	70	–	65	–	65	ns	
Write data setup time	t_{WDS}	30	–	15	–	10	–	ns	
Write data hold time	t_{WDH}	30	–	25	–	20	–	ns	

Table 20-8 (1) Bus Timing (R-Mask Versions) (cont)

		Condition A							
		8 MHz		10 MHz		16 MHz			Test
Item	Symbol	Min	Max	Min	Max	Min	Max	Unit	Conditions
Wait setup time	t_{WTS}	40	–	40	–	40	–	ns	See figure 20-5
Wait hold time	t_{WTH}	10	–	10	–	10	–	ns	
Bus request setup time	t_{BRQS}	40	–	40	–	40	–	ns	See figure 20-10
Bus acknowledge delay time 1	t_{BACD1}	–	70	–	60	–	55	ns	
Bus acknowledge delay time 2	t_{BACD2}	–	70	–	60	–	55	ns	
Bus floating delay time	t_{BZD}	–	t_{BACD1}	–	t_{BACD1}	–	t_{BACD1}	ns	
E clock delay time	t_{ED}	–	20	–	15	–	15	ns	See figure 20-11
E clock rise time	t_{Er}	–	15	–	15	–	15	ns	
E clock fall time	t_{Ef}	–	15	–	15	–	15	ns	
Read data hold time (E clock sync)	t_{RDHE}	0	–	0	–	0	–	ns	See figure 20-6
Write data hold time (E clock sync)	t_{WDHE}	50	–	40	–	30	–	ns	

Table 20-8 (2) Bus Timing (S-Mask Versions) **–Preliminary–**

Condition B (5-V S-mask): V_{CC} = 5.0 V ±10%, ø = 2.0 to 16 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications),
T_a = –40 to +85˚C (Wide-Range Specifications)
Condition C (3-V S-mask): V_{CC} = 3.0 to 5.5 V, ø = 2.0 to 10 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications)
Condition D (2.7-V S-mask): V_{CC} = 2.7 to 5.5 V, ø = 2.0 to 8 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications)

		Condition D		Condition C		Condition B			
		8 MHz		10 MHz		16 MHz			Test
Item	Symbol	Min	Max	Min	Max	Min	Max	Unit	Conditions
Clock cycle time	t_{cyc}	125	500	100	500	62.5	500	ns	See figure 20-4
Clock pulse width Low	t_{CL}	35	–	30	–	20	–	ns	
Clock pulse width High	t_{CH}	35	–	30	–	20	–	ns	
Clock rise time	t_{Cr}	–	20	–	20	–	10	ns	
Clock fall time	t_{Cf}	–	20	–	20	–	10	ns	
Address delay time	t_{AD}	–	60	–	55	–	30	ns	
Address hold time	t_{AH}	20	–	10	–	5	–	ns	
Data strobe delay time 1	t_{DSD1}	–	60	–	40	–	30	ns	
Data strobe delay time 2	t_{DSD2}	–	60	–	50	–	30	ns	
Data strobe delay time 3	t_{DSD3}	–	60	–	50	–	30	ns	
Write data strobe pulse width	t_{DSWW}	150	–	120	–	70	–	ns	
Address setup time 1	t_{AS1}	20	–	15	–	10	–	ns	
Address setup time 2	t_{AS2}	80	–	65	–	30	–	ns	
Read data setup time	t_{RDS}	50	–	40	–	20	–	ns	
Read data hold time	t_{RDH}	0	–	0	–	0	–	ns	
Read data access time	t_{ACC}	–	190	–	160	–	100	ns	
Write data delay time	t_{WDD}	–	75	–	70	–	50	ns	
Write data setup time	t_{WDS}	15	–	10	–	10	–	ns	
Write data hold time	t_{WDH}	25	–	20	–	10	–	ns	
Wait setup time	t_{WTS}	40	–	40	–	30	–	ns	See figure 20-5
Wait hold time	t_{WTH}	10	–	10	–	10	–	ns	
Bus request setup time	t_{BRQS}	40	–	40	–	30	–	ns	See figure 20-10
Bus acknowledge delay time 1	t_{BACD1}	–	60	–	55	–	30	ns	

		Conditions D		Conditions C		Conditions B			
		8 MHz		10 MHz		16 MHz			
Item	**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Test Condition**
Bus acknowledge delay time 2	t_{BACD2}	–	60	–	55	–	30	ns	See figure 20-10
Bus floating delay time	t_{BZD}	–	t_{BACD1}	–	t_{BACD1}	–	t_{BACD1}	ns	
E clock delay time	t_{ED}	–	20	–	20	–	10	ns	See figure 20-11
E clock rise time	t_{Er}	–	20	–	20	–	10	ns	
E clock fall time	t_{Ef}	–	20	–	20	–	10	ns	
Read data hold time (E clock sync)	t_{RDHE}	0	–	0	–	0	–	ns	See figure 20-6
Write data hold time (E clock sync)	t_{WDHE}	40	–	30	–	10	–	ns	

Table 20-9 (1) Control Signal Timing (R-Mask Versions)

Condition A (R-mask): V_{CC} = 5.0 V ±10%, ø = 0.5 to 10 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications),
T_a = –40 to +85˚C (Wide-Range Specifications)

		Condition A							
		6 MHz		8 MHz		10 MHz			Test
Item	Symbol	Min	Max	Min	Max	Min	Max	Unit	Condition
$\overline{RES}$ setup time	t_{RESS}	200	–	200	–	200	–	ns	See figure 20-7
$\overline{RES}$ pulse width 1*	t_{RESW1}	6.0	–	6.0	–	6.0	–	t_{cyc}	
$\overline{RES}$ pulse width 2*	t_{RESW2}	520	–	520	–	520	–	t_{cyc}	
$\overline{RES}$ output delay time	t_{RESD}	–	100	–	100	–	100	ns	See figure 20-8
$\overline{RES}$ output pulse width	t_{RESOW}	132	–	132	–	132	–	t_{cyc}	
NMI setup time	t_{NMIS}	150	–	150	–	150	–	ns	See figure 20-9
NMI hold time	t_{NMIH}	10	–	10	–	10	–	ns	
IRQ_0 setup time	t_{IRQ0S}	50	–	50	–	50	–	ns	
IRQ_1 setup time	t_{IRQ1S}	50	–	50	–	50	–	ns	
IRQ_1 hold time	t_{IRQ1H}	10	–	10	–	10	–	ns	
A/D trigger setup time	t_{TRGS}	50	–	50	–	50	–	ns	See figure 20-22
A/D trigger hold time	t_{TRGH}	10	–	10	–	10	–	ns	
NMI pulse width (for recovery from software standby mode)	t_{NMIW}	200	–	200	–	200	–	ns	
Crystal oscillator settling time (reset)	t_{OSC1}	20	–	20	–	20	–	ms	See figure 20-12
Crystal oscillator settling time (software standby)	t_{OSC2}	10	–	10	–	10	–	ms	See figure 18-1

Note: * t_{RESW2} applies at power-on and when the RSTOE bit in the reset control/status register (RSTCSR) is set to 1. t_{RESW1} applies when RSTOE is cleared to 0.

Table 20-9 (2) Control Signal Timing (S-Mask Versions) **–Preliminary–**

Condition B (5-V S-mask): V_{CC} = 5.0 V ±10%, ø = 2.0 to 16 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications),
T_a = –40 to +85˚C (Wide-Range Specifications)
Condition C (3-V S-mask): V_{CC} = 3.0 to 5.5 V, ø = 2.0 to 10 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications)
Condition D (2.7-V S-mask): V_{CC} = 2.7 to 5.5 V, ø = 2.0 to 8 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications)

		Condition D		Condition C		Condition B			
		8 MHz		10 MHz		16 MHz			Test
Item	Symbol	Min	Max	Min	Max	Min	Max	Unit	Conditions
$\overline{RES}$ setup time	t_{RESS}	200	–	200	–	200	–	ns	See figure 20-7
$\overline{RES}$ pulse width 1*	t_{RESW1}	6.0	–	6.0	–	6.0	–	t_{cyc}	
$\overline{RES}$ pulse width 2*	t_{RESW2}	520	–	520	–	520	–	t_{cyc}	
$\overline{RES}$ output delay time	t_{RESD}	–	100	–	100	–	100	ns	See figure 20-8
$\overline{RES}$ output pulse width	t_{RESOW}	132	–	132	–	132	–	t_{cyc}	
NMI setup time	t_{NMIS}	200	–	200	–	150	–	ns	See figure 20-9
NMI hold time	t_{NMIH}	10	–	10	–	10	–	ns	
IRQ_0 setup time	t_{IRQ0S}	50	–	50	–	50	–	ns	
IRQ_1 setup time	t_{IRQ1S}	50	–	50	–	50	–	ns	
IRQ_1 hold time	t_{IRQ1H}	10	–	10	–	10	–	ns	
A/D trigger setup time	t_{TRGS}	50	–	50	–	50	–	ns	See figure 20-22
A/D trigger hold time	t_{TRGH}	10	–	10	–	10	–	ns	
NMI pulse width (for recovery from software standby mode)	t_{NMIW}	200	–	200	–	200	–	ns	
Crystal oscillator settling time (reset)	t_{OSC1}	20	–	20	–	20	–	ms	See figure 20-12
Crystal oscillator settling time (software standby)	t_{OSC2}	10	–	10	–	10	–	ms	See figure 18-1

Note: * t_{RESW2} applies at power-on and when the RSTOE bit in the reset contol/status register (RSTCSR) is set to 1. t_{RESW1} applies when RSTOE is cleared to 0.

Table 20-10 Timing Conditions of On-Chip Supporting Modules **– Preliminary for S-Mask Versions–**

Condition A (R-mask): V_{CC} = 5.0 V ±10%, ø = 0.5 to 10 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C(Regular Specifications),
T_a = –40 to +85˚C (Wide-Range Specifications)

Condition B (5-V S-mask): V_{CC} = 5.0 V ±10%, ø = 2.0 to 16 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications),
T_a = –40 to +85˚C (Wide-Range Specifications)

Condition C (3-V S-mask): V_{CC} = 3.0 to 5.5 V, ø = 2.0 to 10 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications)

Condition D (2.7-V S-mask): V_{CC} = 2.7 to 5.5 V, ø = 2.0 to 8 MHz, V_{SS} = 0 V,
T_a = –20 to +75˚C (Regular Specifications)

			Condition A									
			Condition D				Condition C		Condition B			
			6 MHz		8 MHz		10 MHz		16 MHz			Test
Item		Symbol	Min	Max	Min	Max	Min	Max	Min	Max	Unit	Conditions
FRT	Timer output delay time	t_{FTOD}	–	100	–	100	–	100	–	100	ns	See figure 20-14
	Timer input setup time	t_{FTIS}	50	–	50	–	50	–	50	–	ns	
	Timer clock input setup time	t_{FTCS}	50	–	50	–	50	–	50	–	ns	See figure 20-15
	Timer clock pulse width	t_{FTCWL}, t_{FTCWH}	1.5	–	1.5	–	1.5	–	1.5	–	tcyc	
TMR	Timer output delay time	t_{TMOD}	–	100	–	100	–	100	–	100	ns	See figure 20-16
	Timer clock input setup time	t_{TMCS}	50	–	50	–	50	–	50	–	ns	See figure 20-17
	Timer clock pulse width	t_{TMCWL}, t_{TMCWH}	1.5	–	1.5	–	1.5	–	1.5	–	tcyc	
	Timer reset input setup time	t_{TMRS}	50	–	50	–	50	–	50	–	ns	See figure 20-18
PWM	Timer output delay time	t_{PWOD}	–	100	–	100	–	100	–	100	ns	See figure 20-19

				Condition A									
						Condition D		Condition C		Condition B			
				6 MHz		8 MHz		10 MHz		16 MHz			Test
Item			**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Conditions**
SCI	Input clock cycle	(Async)	t_{Scyc}	2	–	2	–	2	–	2	–	t_{cyc}	See figure 20-20
		(Sync)		4	–	4	–	4	–	4	–	t_{cyc}	
	Input pulse width		t_{SCKW}	0.4	0.6	0.4	0.6	0.4	0.6	0.4	0.6	t_{Scyc}	
	Transmit data delay	(Sync)	t_{TXD}	–	100	–	100	–	100	–	100	ns	See figure 20-21
	Receive data setup time	(Sync)	t_{RXS}	100	–	100	–	100	–	100	–	ns	
	Receive data hold time	(Sync)	t_{RXH}	100	–	100	–	100	–	100	–	ns	
Port	Output data delay time		t_{PWD}	–	100	–	100	–	100	–	100	ns	See figure 20-13
	Input data setup time		t_{PRS}	50	–	50	–	50	–	50	–	ns	
	Input data hold time		t_{PRH}	50	–	50	–	50	–	50	–	ns	

- **Measurement Conditions for AC Characteristics**

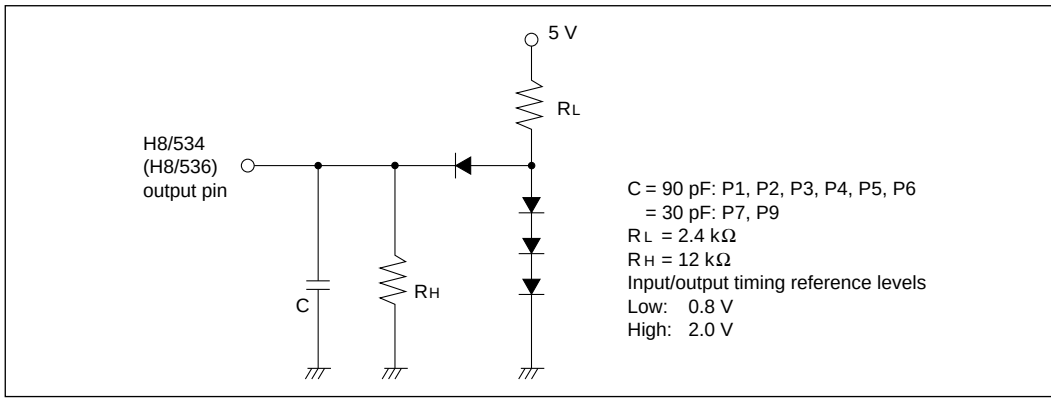


Figure 20-3 Output Load Circuit

t_{AH}, t_{DSWW}, t_{AS1}, t_{AS2}, and t_{ACC} depend on t_{cyc} as shown below. **–Preliminary–**

(1) V_{CC} = 5.0 V ±10% (S-mask)

t_{AH} = $0.5 \times t_{cyc} - 26$ (ns)
t_{DSWW} = $1.5 \times t_{cyc} - 24$ (ns)
t_{AS1} = $0.5 \times t_{cyc} - 21$ (ns)
t_{AS2} = $t_{cyc} - 32$ (ns)
t_{ACC} = $2.5 \times t_{cyc} - 56$ (ns)

(2) V_{CC} = 3.0 V (S-mask)

t_{AH} = $0.5 \times t_{cyc} - 40$ (ns)
t_{DSWW} = $1.5 \times t_{cyc} - 30$ (ns)
t_{AS1} = $0.5 \times t_{cyc} - 35$ (ns)
t_{AS2} = $t_{cyc} - 35$ (ns)
t_{ACC} = $2.5 \times t_{cyc} - 90$ (ns)

(3) V_{CC} = 2.7 V (S-mask)

t_{AH} = $0.5 \times t_{cyc} - 42$ (ns)
t_{DSWW} = $1.5 \times t_{cyc} - 37$ (ns)
t_{AS1} = $0.5 \times t_{cyc} - 42$ (ns)
t_{AS2} = $t_{cyc} - 45$ (ns)
t_{ACC} = $2.5 \times t_{cyc} - 122$ (ns)

20.2.3 A/D Converter Characteristics

Tables 20-11 and 20-12 list the characteristics of the on-chip A/D converter.

Tables 20-11 A/D Converter Characteristics – Preliminary for S-Mask Versions– (5-V Versions)

Conditions: V_{CC} = 5.0V ±10%, AV_{CC} = 5.0V ±10%, V_{SS} = AV_{SS} = 0V,
T_a = –20 to +75˚C (Regular Specifications)
T_a = –40 to +85˚C (Wide-Range Specifications)

	R-Mask									S-Mask			
	6 MHz			8 MHz			10 MHz			16 MHz			
Item	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Unit**
Resolution	10	10	10	10	10	10	10	10	10	10	10	10	Bits
Conversion time	—	—	23.0	—	—	17.25	—	—	13.8	—	—	8.625	µs
Analog input capacitance	—	—	20	—	—	20	—	—	20	—	—	20	pF
Allowable signal-source impedance	—	—	10	—	—	10	—	—	10	—	—	5	kΩ
Nonlinearity error	—	—	±2.0	—	—	±2.0	—	—	±2.0	—	—	±2.0	LSB
Offset error	—	—	±2.0	—	—	±2.0	—	—	±2.0	—	—	±2.0	LSB
Full-scale error	—	—	±2.0	—	—	±2.0	—	—	±2.0	—	—	±2.0	LSB
Quantizing error	—	—	±0.5	—	—	±0.5	—	—	±0.5	—	—	±0.5	LSB
Absolute accuracy	—	—	±2.5	—	—	±2.5	—	—	±2.5	—	—	±2.5	LSB

Condition C (3-V S-mask): V_{CC} = 3.0 to 5.5 V, V_{SS} = AV_{SS} = 0 V, T_a = –20 to +75˚C
(Regular Specifications), AV_{CC} = 3.0 to 5.5 V
Condition D (2.7-V S-mask): V_{CC} = 2.7 to 5.5 V, V_{SS} = AV_{SS} = 0 V, T_a = –20 to +75˚C
(Regular Specifications), AV_{CC} = 2.7 to 5.5 V

	Condition D*1			Condition C*2			
	8 MHz			10 MHz			
Item	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**	**Unit**
Resolution	10	10	10	10	10	10	Bits
Conversion time	–	–	17.25	–	–	13.8	µs
Analog input capacitance	–	–	20	–	–	20	pF
Allowable signal-source impedance	–	–	5	–	–	5	kΩ
Nonlinearity error	–	–	±3.5	–	–	±3.5	LSB
Offset error	–	–	±3.5	–	–	±3.5	LSB
Full-scale error	–	–	±3.5	–	–	±3.5	LSB
Quantizing error	–	–	±0.5	–	–	±0.5	LSB
Absolute accuracy	–	–	±4.0	–	–	±4.0	LSB

Notes: Maximum operating frequency of A/D converter:
*1 AV_{CC} = 2.7 to 3.0 V: 8 MHz (conversion time: 17.25 µs)
*2 AV_{CC} = 3.0 to 4.5 V: 10 MHz (conversion time: 13.8 µs)

20.3 MCU Operational Timing

This section provides the following timing charts:

20.3.1 Bus Timing

1. Basic Bus Cycle (without Wait States) in Expanded Modes

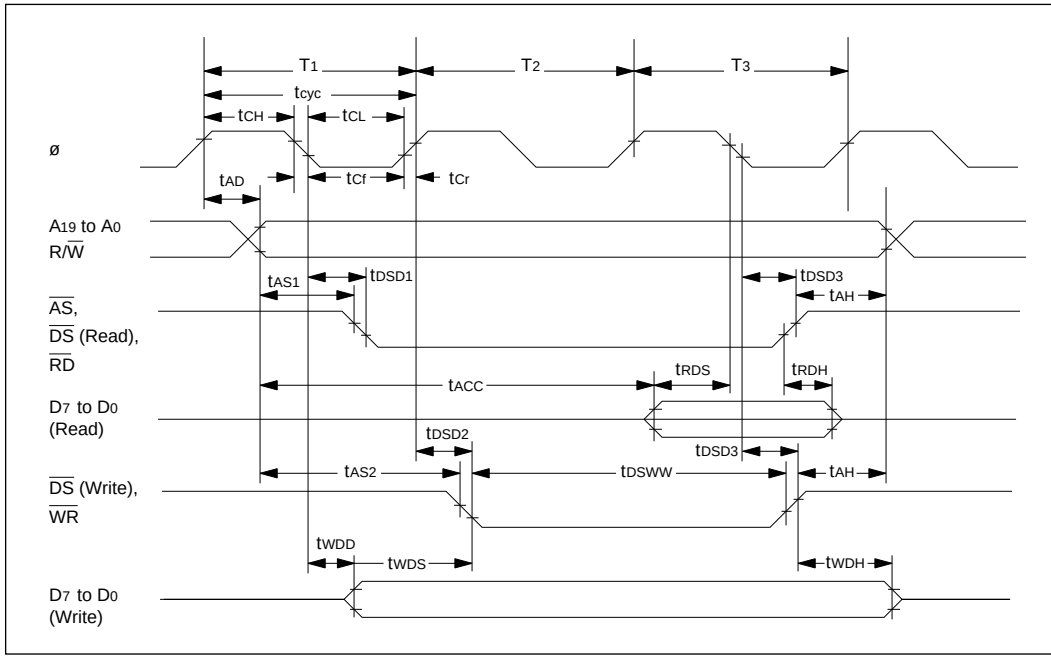


Figure 20-4 Basic Bus Cycle (without Wait States) in Expanded Modes

2. Basic Bus Cycle (with 1 Wait State) in Expanded Modes

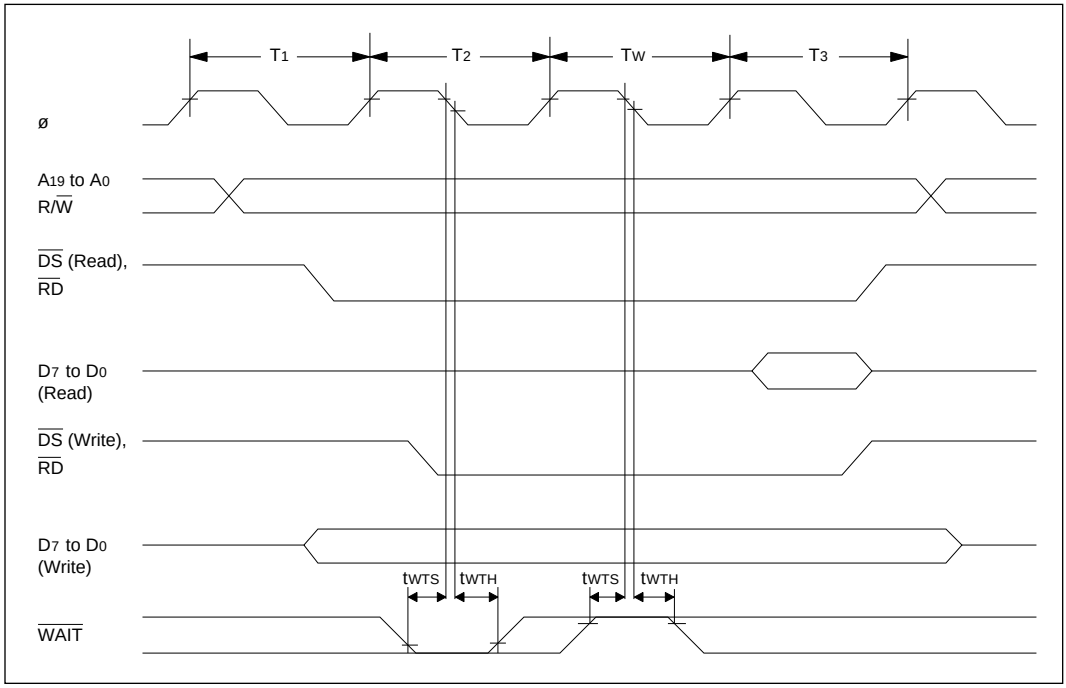


Figure 20-5 Basic Bus Cycle (with 1 Wait State) in Expanded Modes

3. Bus Cycle Synchronized with E Clock

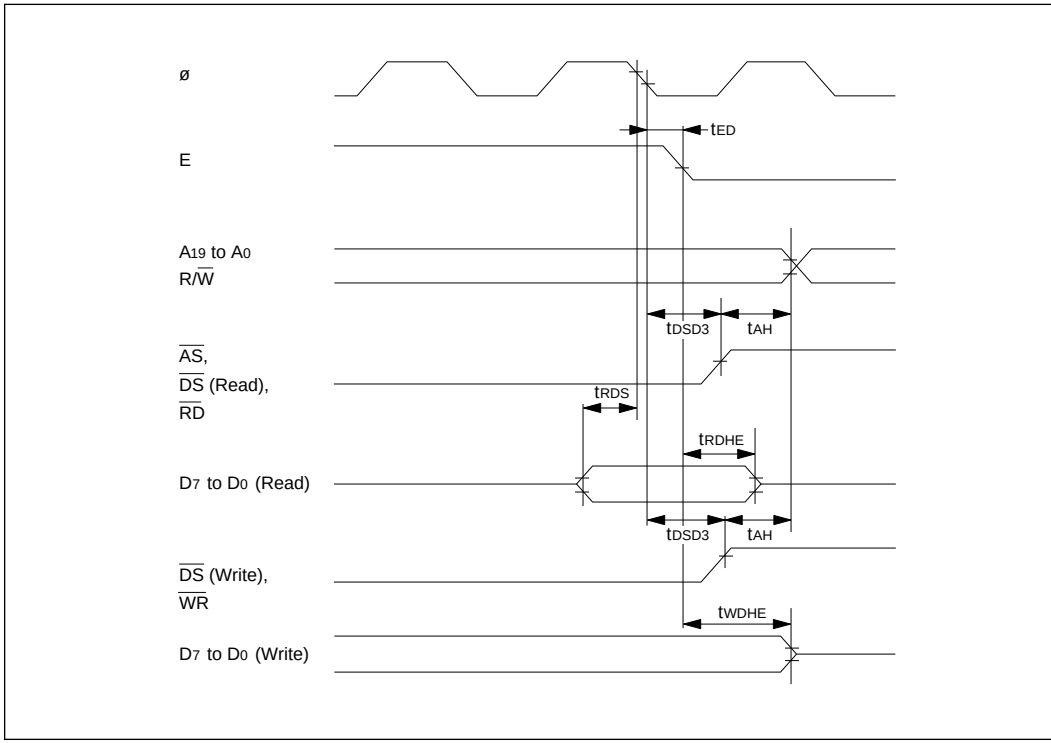


Figure 20-6 Bus Cycle Synchronized with E Clock

20.3.2 Control Signal Timing

1. Reset Input Timing

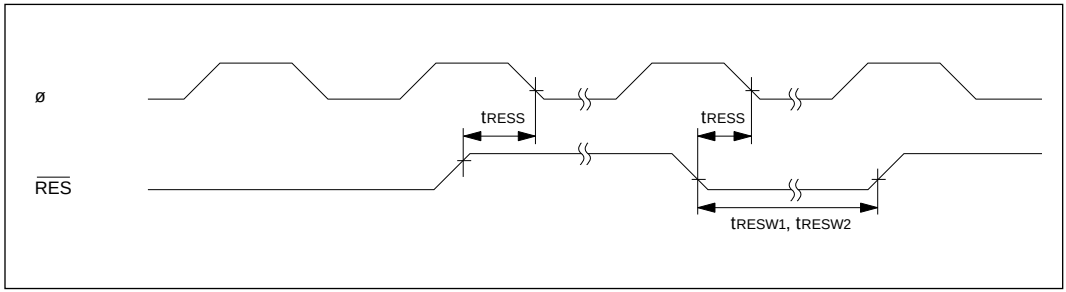


Figure 20-7 Reset Input Timing

2. Reset Output Timing

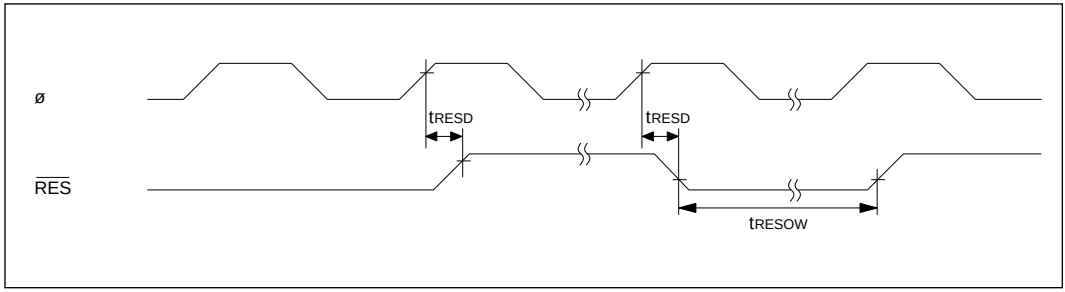


Figure 20-8 Reset Output Timing

3. NMI Pulse Width

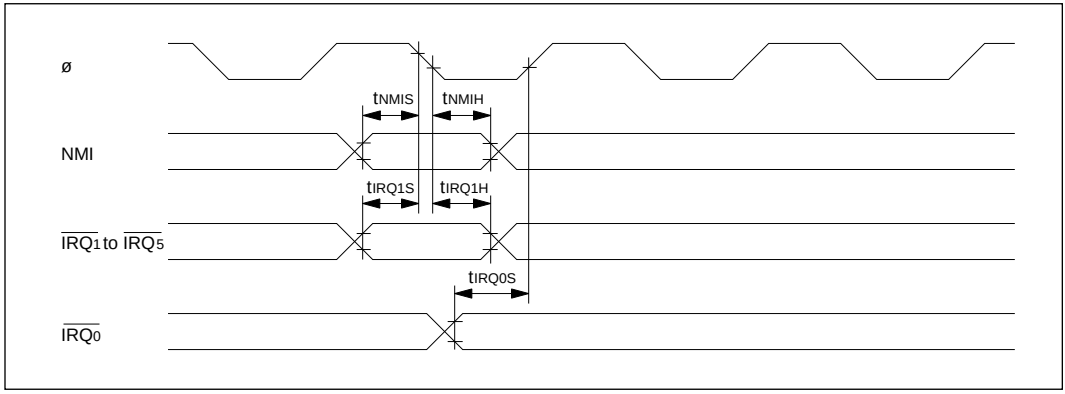


Figure 20-9 Interrupt Input Timing

4. Bus Release State Timing

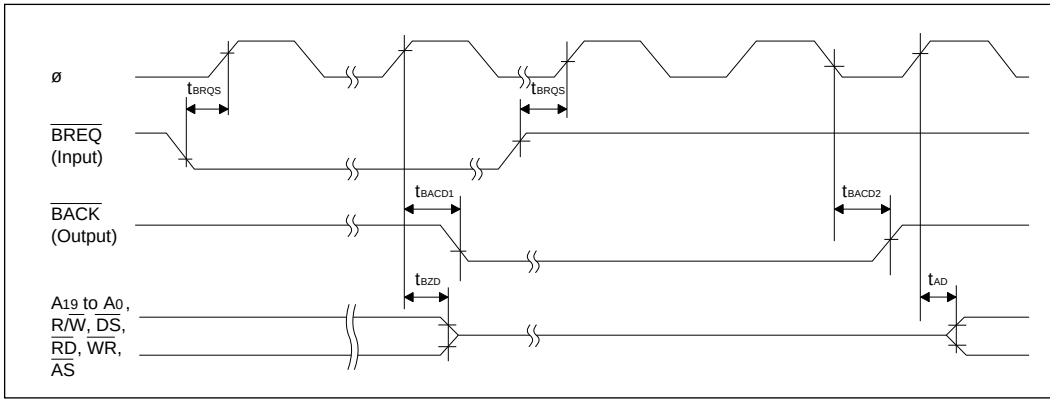


Figure 20-10 Bus Release State Timing

20.3.3 Clock Timing

1. E Clock Timing

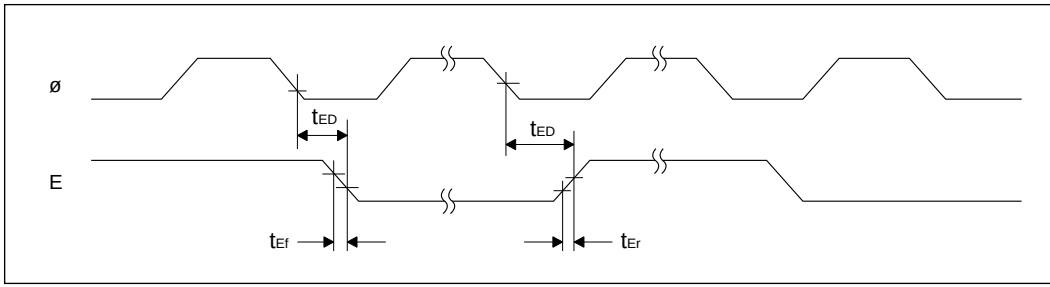


Figure 20-11 E Clock Timing

2. Clock Oscillator Stabilization Timing

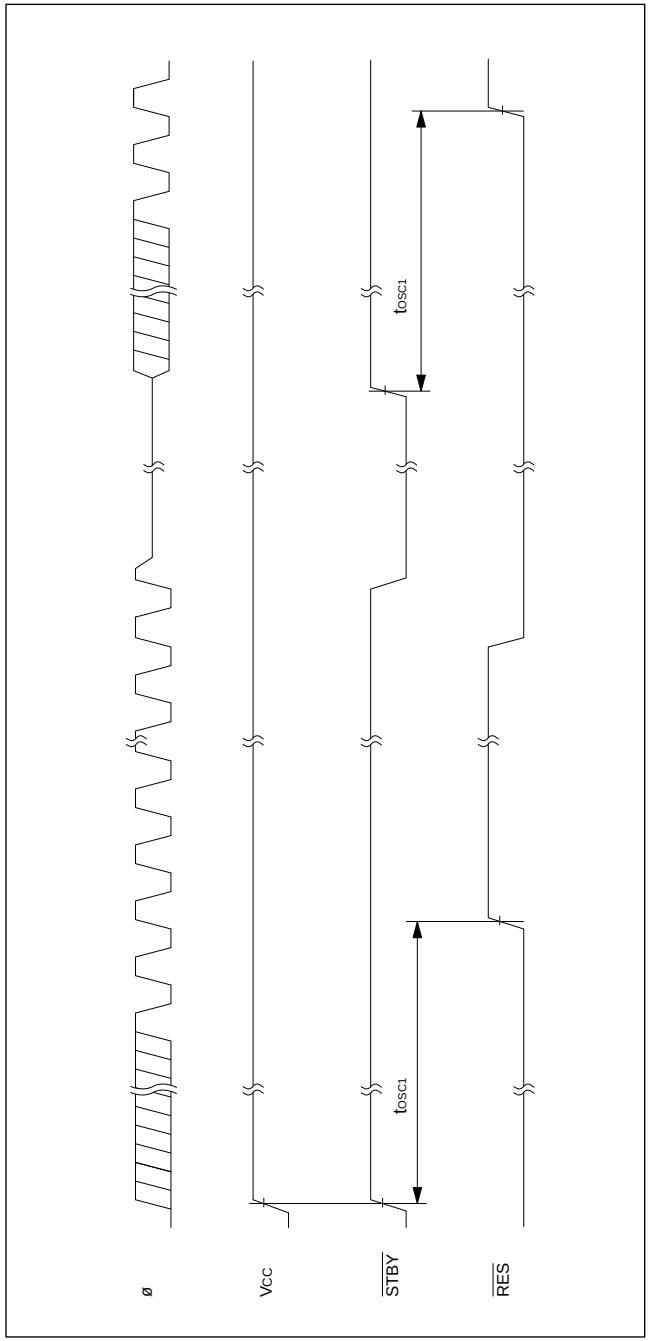


Figure 20-12 Clock Oscillator Stabilization Timing

20.3.4 I/O Port Timing

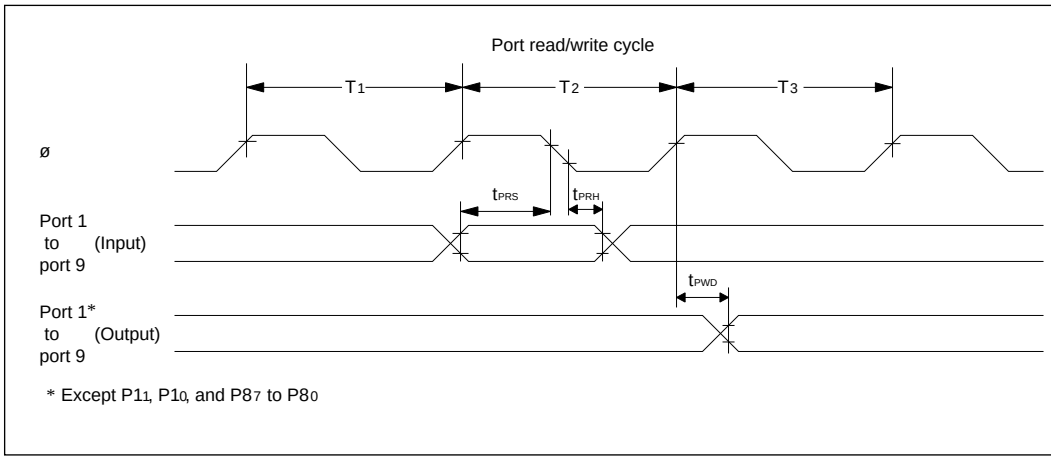


Figure 20-13 I/O Port Input/Output Timing

20.3.5 16-Bit Free-Running Timer Timing

1. Free-Running Timer Input/Output Timing

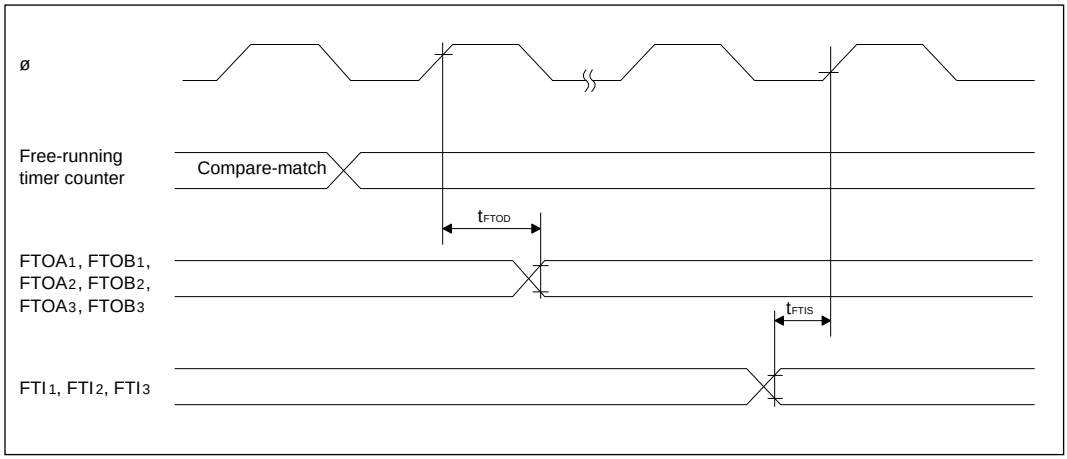


Figure 20-14 Free-Running Timer Input/Output Timing

2. External Clock Input Timing for Free-Running Timers

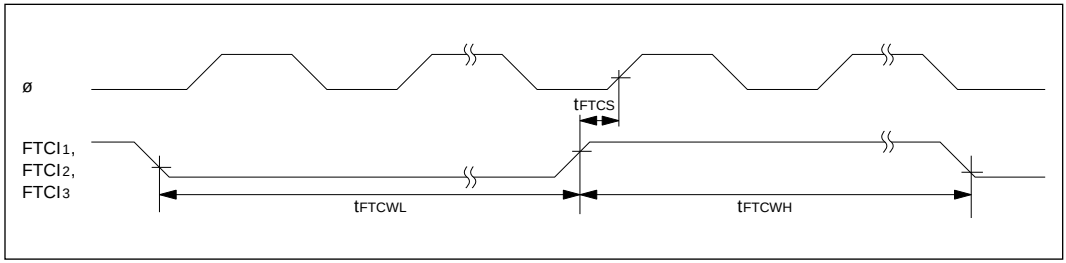


Figure 20-15 External Clock Input Timing for Free-Running Timers

20.3.6 8-Bit Timer Timing

1. 8-Bit Timer Output Timing

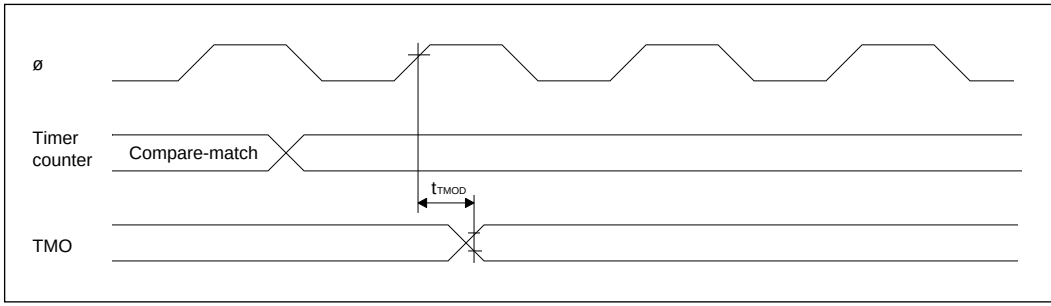


Figure 20-16 8-Bit Timer Output Timing

2. 8-Bit Timer Clock Input Timing

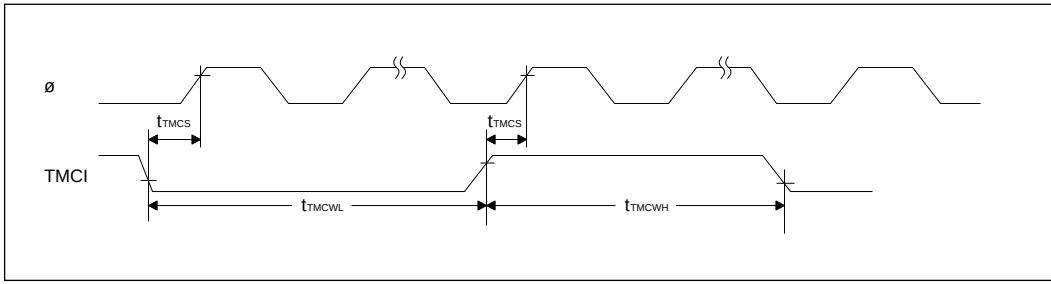


Figure 20-17 8-Bit Timer Clock Input Timing

3. 8-Bit Timer Reset Input Timing

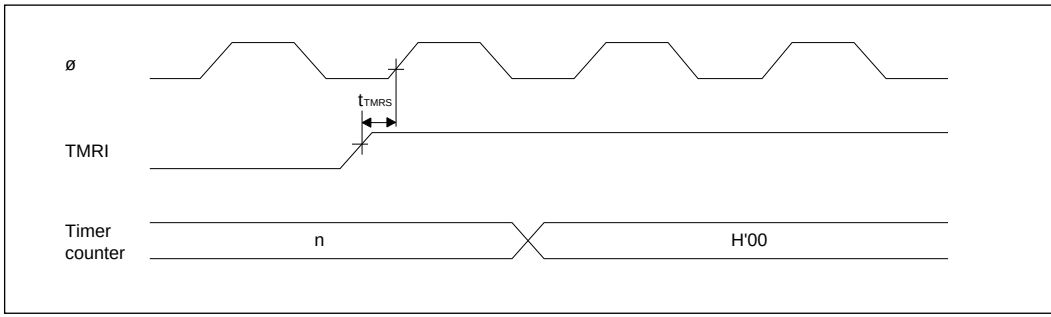


Figure 20-18 8-Bit Timer Reset Input Timing

20.3.7 Pulse Width Modulation Timer Timing

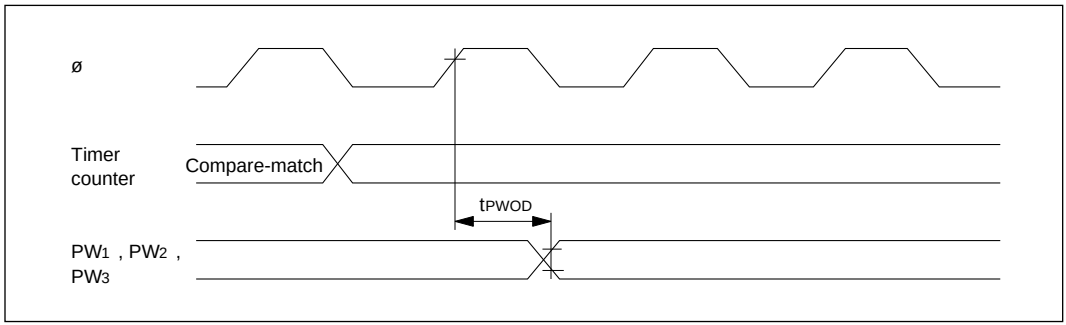


Figure 20-19 PWM Timer Output Timing

20.3.8 Serial Communication Interface Timing

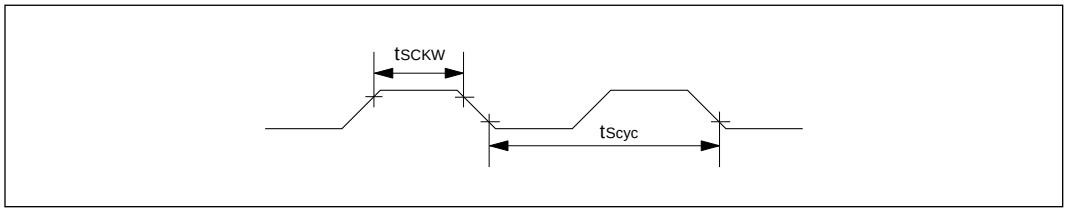


Figure 20-20 SCI Input Clock Timing

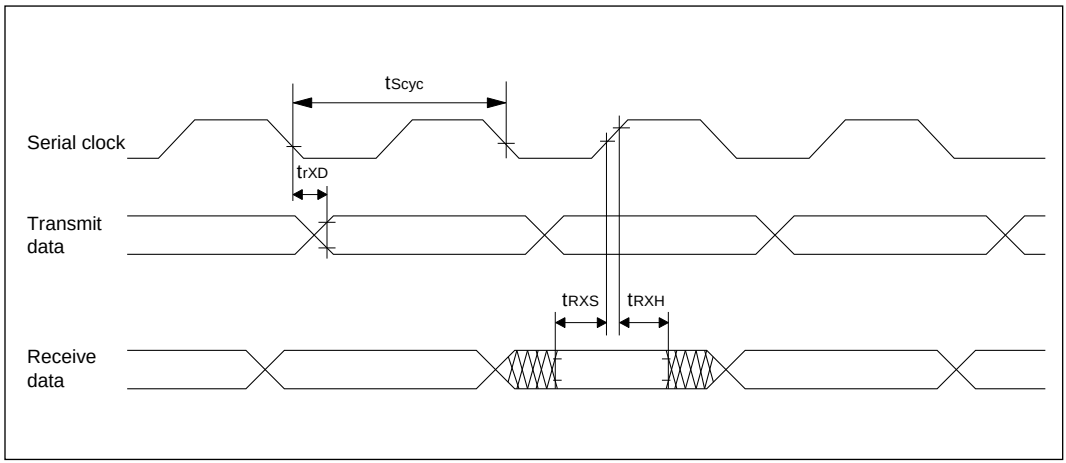


Figure 20-21 SCI Input/Output Timing (Synchronous Mode)

20.3.9 A/D Trigger Signal Input Timing

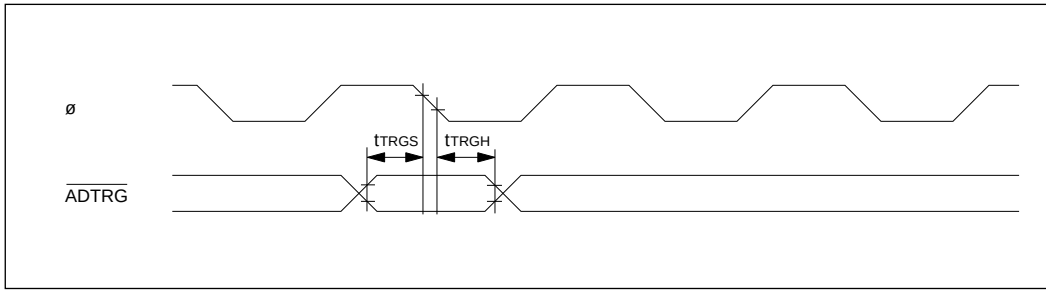


Figure 20-22 A/D Trigger Signal Input Timing

Appendix A Instructions

A.1 Instruction Set

Operation Notation

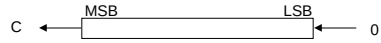
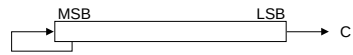
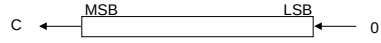
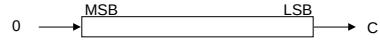
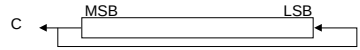
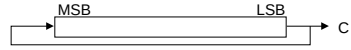
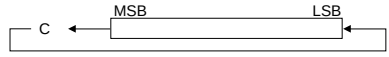
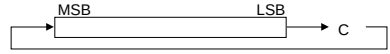
Rd	General register (destination operand)
Rs	General register (source operand)
Rn	General register
(EAd)	Destination operand
(EAs)	Source operand
CCR	Condition code register
N	N (Negative) flag in CCR
Z	Z (Zero) flag in CCR
V	V (Overflow) flag in CCR
C	C (Carry) flag in CCR
CR	Control register
PC	Program counter
CP	Code page register
SP	Stack pointer

FP	Frame pointer
#IMM	Immediate data
disp	Displacement
+	Add
–	Subtract
×	Multiply
÷	Divide
∧	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Move
↔	Swap
¬	Logical NOT

Condition Code Notation

↕	Changed after instruction execution
0	Cleared to 0
1	Set to 1
—	Value before operation is retained
Δ	Changed depending on condition

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
Data transfer	MOV: G	(EAs) ⟶ Rd Rs ⟶ (EAd) #IMM ⟶ (EAd)	B/W	↕	↕	0	—
	MOV: E	#IMM ⟶ Rd (short format)	B	↕	↕	0	—
	MOV: F	@ (d: 8, FP) ⟶ Rd Rs ⟶ @ (d: 8, FP) (short format)	B/W	↕	↕	0	—
	MOV: I	#IMM ⟶ Rd (short format)	W	↕	↕	0	—
	MOV: L	(@aa: 8) ⟶ Rd (short format)	B/W	↕	↕	0	—
	MOV: S	Rs ⟶ (@aa: 8) (short format)	B/W	↕	↕	0	—
	LDM	@ SP + ⟶ Rn (register list)	W	—	—	—	—
	STM	Rn (register list) ⟶ @ – SP	W	—	—	—	—
	XCH	Rs ⟷ Rd	W	—	—	—	—
	SWAP	Rd (upper byte) ⟷ Rd (lower byte)	B	↕	↕	0	—
	MOVTPE	Rs ⟶ (EAd) Synchronized with E clock	B	—	—	—	—
	MOVFPE	(EAs) ⟶ Rd Synchronized with E clock	B	—	—	—	—
Arithmetic operations	ADD: G	Rd + (EAs) ⟶ Rd	B/W	↕	↕	↕	↕
	ADD: Q	(EAd) + #IMM ⟶ (EAd) (#IMM = ±1, ±2) (short format)	B/W	↕	↕	↕	↕
	ADDS	Rd + (EAs) ⟶ Rd (Rd is always word size)	B/W	—	—	—	—
	ADDX	Rd + (EAs) + C ⟶ Rd	B/W	↕	↕	↕	↕
	DADD	$(Rd)_{10}$ + $(Rs)_{10}$ + C ⟶ $(Rd)_{10}$	B	—	↕	—	↕
	SUB	Rd – (EAs) ⟶ Rd	B/W	↕	↕	↕	↕
	SUBS	Rd – (EAs) ⟶ Rd	B/W	—	—	—	—
	SUBX	Rd – (EAs) – C ⟶ Rd	B/W	↕	↕	↕	↕
	DSUB	$(Rd)_{10}$ – $(Rs)_{10}$ – C ⟶ $(Rd)_{10}$	B	—	↕	—	↕
	MULXU	Rd × (EAs) ⟶ Rd 8 × 8 (Unsigned) 16 × 16	B/W	↕	↕	0	0
	DIVXU	Rd ÷ (EAs) ⟶ Rd 16 ÷ 8 (Unsigned) 32 ÷ 16	B/W	↕	↕	↕	0
	CMP: G	Rd – (EAs), Set CCR (EAd) – #IMM, Set CCR	B/W	↕	↕	↕	↕
	CMP: E	Rd – #IMM, Set CCR (short format)	B	↕	↕	↕	↕
	CMP: I	Rd – #IMM, Set CCR (short format)	W	↕	↕	↕	↕

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
Arithmetic operations	EXTS	(< Bit 7 > of < Rd >) ⟶ (< Bit 15 to 8 > of < Rd >)	B	↕	↕	0	0
	EXTU	0 ⟶ (<Bit 15 to 8 > of < Rd >)	B	0	↕	0	0
	TST	(EAd) – 0, Set CCR	B/W	↕	↕	0	0
	NEG	0 – (EAd) ⟶ (EAd)	B/W	↕	↕	0	↕
	CLR	0 ⟶ (EAd)	B/W	0	1	0	0
	TAS	(EAd) – 0, Set CCR $(1)_2$ ⟶ (< Bit 7 > of < EAd >)	B	↕	↕	0	0
Shift operations	SHAL		B/W	↕	↕	↕	↕
	SHAR		B/W	↕	↕	0	↕
	SHLL		B/W	↕	↕	0	↕
	SHLR		B/W	0	↕	0	↕
	ROTL		B/W	↕	↕	0	↕
	ROTR		B/W	↕	↕	0	↕
	ROTXL		B/W	↕	↕	0	↕
	ROTXR		B/W	↕	↕	0	↕
Logic operations	AND	Rd ∧ (EAs) ⟶ Rd	B/W	↕	↕	0	—
	OR	Rd ∨ (EAs) ⟶ Rd	B/W	↕	↕	0	—
	XOR	Rd ⊕ (EAs) ⟶ Rd	B/W	↕	↕	0	—
	NOT	¬ (EAd) ⟶ (EAd)	B/W	↕	↕	0	—
Bit manipulations	BSET	¬ (< Bit number > of < EAd >) ⟶ Z 1 ⟶ (< Bit number > of < EAd >)	B/W	—	↕	—	—
	BCLR	¬ (< Bit number > of < EAd >) ⟶ Z 0 ⟶ (< Bit number > of < EAd >)	B/W	—	↕	—	—
	BTST	¬ (< Bit number > of < EAd >) ⟶ Z	B/W	—	↕	—	—
	BNOT	¬ (< Bit number > of < EAd >) ⟶ Z ⟶ (< Bit number > of < EAd >)	B/W	—	↕	—	—

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
Branching instructions	Bcc	If condition is true then PC + disp ⟶ PC else next;	—	—	—	—	—

Mnemonic		Description	Condition
BRA	(BT)	Always (True)	True
BRN	(BF)	Never (False)	False
BHI		HIgh	C ∨ Z = 0
BLS		Low or Same	C ∨ Z = 1
BCC	(BHS)	Carry Clear (High or Same)	C = 0
BCS	(BLO)	Carry Set (LOw)	C = 1
BNE		Not Equal	Z = 0
BEQ		EQual	Z = 1
BVC		oVerflow Clear	V = 0
BVS		oVerflow Set	V = 1
BPL		PLus	N = 0
BMI		MInus	N = 1
BGE		Greater or Equal	N ⊕ V = 0
BLT		Less Than	N ⊕ V = 1
BGT		Greater Than	Z ∨ (N ⊕ V) = 0
BLE		Less or Equal	Z ∨ (N ⊕ V) = 1

	JMP	Effective address ⟶ PC	—	—	—	—	—
	PJMP	Effective address ⟶ CP, PC	—	—	—	—	—
	BSR	PC ⟶ @ – SP PC + disp ⟶ PC	—	—	—	—	—
	JSR	PC ⟶ @ – SP Effective address ⟶ PC	—	—	—	—	—
	PJSR	PC ⟶ @ – SP CP ⟶ @ – SP Effective address ⟶ CP, PC	—	—	—	—	—
	RTS	@ SP + ⟶ PC	—	—	—	—	—
	PRTS	@ SP + ⟶ CP @ SP + ⟶ PC	—	—	—	—	—
	RTD	@ SP + ⟶ PC SP + #IMM ⟶ SP	—	—	—	—	—
	PRTD	@ SP + ⟶ CP @ SP + ⟶ PC SP + #IMM ⟶ SP	—	—	—	—	—
	SCB SCB/F SCB/NE SCB/EQ	If condition is true then next; else Rn – 1 ⟶ Rn; If Rn = –1 then next; else PC + disp ⟶ PC;	—	—	—	—	—

Mnemonic	Description	Condition
SCB/F		False
SCB/NE	Not Equal	Z = 0
SCB/EQ	Equal	Z = 1

	Mnemonic	Operation	Size B/W	CCR Bit N	Z	V	C
System control	TRAPA	PC ⟶ @ – SP (If MAX MODE CP ⟶ @ – SP) SR ⟶ @ – SP (If MAX MODE < vector > ⟶ CP) < vector > ⟶ PC	—	—	—	—	—
	TRAP/VS	If V bit = “1” then TRAP else next;	—	—	—	—	—
	RTE	@ SP + ⟶ SR (If MAX MODE @ SP + ⟶ CP) @ SP + ⟶ PC	—	↕	↕	↕	↕
	LINK	FP (R6) ⟶ @ – SP SP ⟶ FP (R6) SP + #IMM ⟶ SP	—	—	—	—	—
	UNLK	FP (R6) ⟶ SP @SP + ⟶ FP	—	—	—	—	—
	SLEEP	Normal running mode ⟶ power-down state	—	—	—	—	—
	LDC	(EAs) ⟶ CR	B/W*	△	△	△	△
	STC	CR ⟶ (EAd)	B/W*	—	—	—	—
	ANDC	CR ∧ #IMM ⟶ CR	B/W*	△	△	△	△
	ORC	CR ∨ #IMM ⟶ CR	B/W*	△	△	△	△
	XORC	CR ⊕ #IMM ⟶ CR	B/W*	△	△	△	△
	NOP	PC + 1 ⟶ PC	—	—	—	—	—

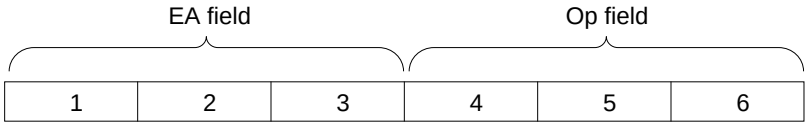
* Depends on the CR.

A.2 Instruction Codes

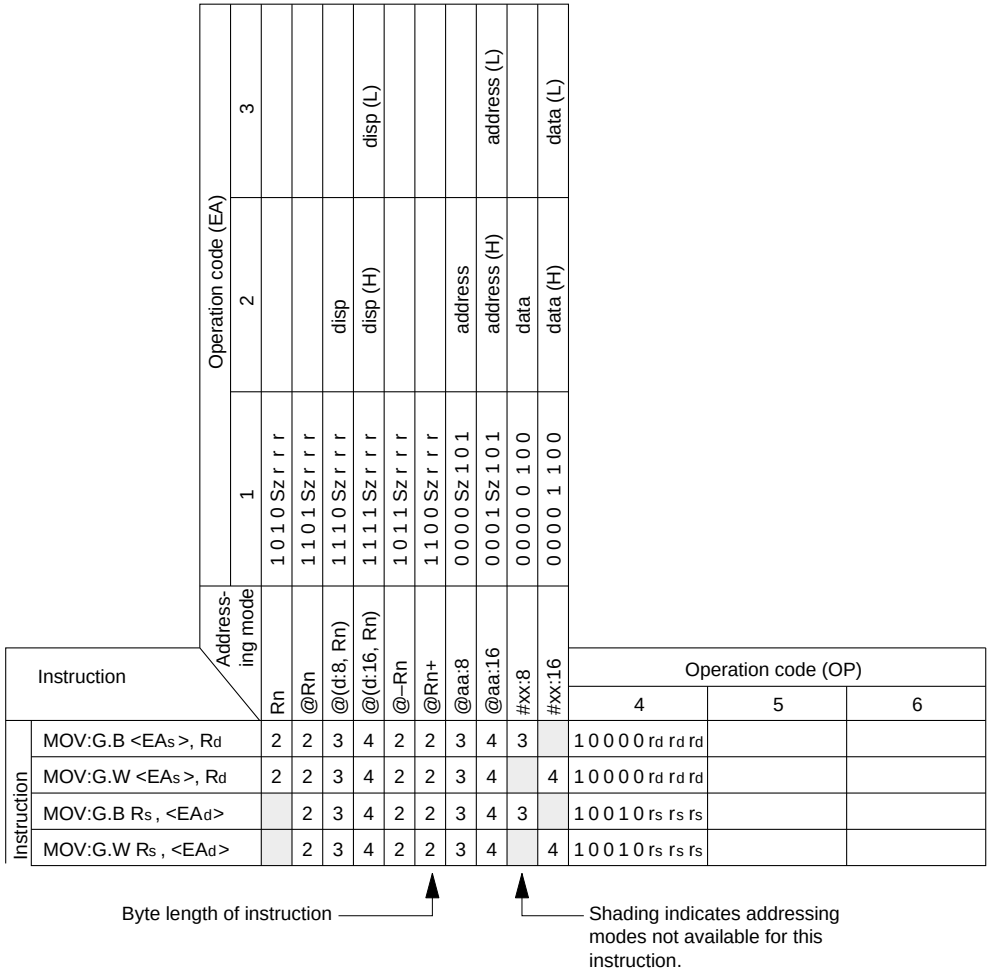
Table A-1(a) to (d) shows the machine-language coding of each instruction.

- **How to read table A-1 (a) to (d)**

The general operand format consists of an effective address (EA) field and operation-code (OP) field specified in the following order.



Bytes 2, 3, 5, 6 are not present in all instructions.



Some instructions have a special format in which the operation code comes first.

The following notation is used in the tables.

- Sz: Operand size (byte or word)
  Byte: Sz = 0
  Word: Sz = 1

• rrr : General register number field

rrr	Sz = 0 (Byte)		Sz = 1 (Word)
	15 8	7 0	15 0
000	Not used	R0	R0
001	Not used	R1	R1
010	Not used	R2	R2
011	Not used	R3	R3
100	Not used	R4	R4
101	Not used	R5	R5
110	Not used	R6	R6
111	Not used	R7	R7

• ccc : Control register number field

ccc	Sz = 0 (Byte)	Sz = 1 (Word)
000	(Not allowed*)	15 0 SR
	7 0	
001	CCR	(Not allowed)
010	(Not allowed)	(Not allowed)
011	BR	(Not allowed)
100	EP	(Not allowed)
101	DP	(Not allowed)
110	(Not allowed)	(Not allowed)
111	TP	(Not allowed)

* “Not allowed” means that this combination of bits must not be specified. Specifying a disallowed combination may cause abnormal results.

• register list: A byte in which bits indicate general registers as follows

Bit	7	6	5	4	3	2	1	0
	R7	R6	R5	R4	R3	R2	R1	R0

• #VEC: Four bits designating a vector number from 0 to 15. The vector numbers correspond to addresses of entries in the exception vector table as follows:

	Vector Address	
#VEC	**Minimum Mode**	**Maximum Mode**
0	H'0020 – H'0021	H'0040 – H'0043
1	H'0022 – H'0023	H'0044 – H'0047
2	H'0024 – H'0025	H'0048 – H'004B
3	H'0026 – H'0027	H'004C – H'004F
4	H'0028 – H'0029	H'0050 – H'0053
5	H'002A – H'002B	H'0054 – H'0057
6	H'002C – H'002D	H'0058 – H'005B
7	H'002E – H'002F	H'005C – H'005F

	Vector Address	
#VEC	**Minimum Mode**	**Maximum Mode**
8	H'0030 – H'0031	H'0060 – H'0063
9	H'0032 – H'0033	H'0064 – H'0067
10	H'0034 – H'0035	H'0068 – H'006B
11	H'0036 – H'0037	H'006C – H'006F
12	H'0038 – H'0039	H'0070 – H'0073
13	H'003A – H'003B	H'0074 – H'0077
14	H'003C – H'003D	H'0078 – H'007B
15	H'003E – H'003F	H'007C – H'007F

• **Examples of machine-language coding**

Example 1: ADD:G.B @R0, R1

	EA Field	**OP Field**	**Notes**
Table A-1 (a)	1101Szrrr	00100rdrdrd	Machine code for ADD:G.B @Rs, Rd
Machine code	11010000	00100 0 0 1	Sz = 0 (byte)
	H'D021		Rs = R0, Rd = R1

Example 2: ADD:G.W @H'11:8, R1

	EA Field		**OP Field**	**Notes**
Table A-1 (a)	0000Sz101	00010001	00100rdrdrd	Machine code for ADD:G.W @aa:8, Rd
Machine code	0000 1 101	00010001	00100 0 0 1	Sz = 1 (word)
	H'0D1121			aa = H'11, Rd = R1

Table A-1 (a) Machine Language Coding [General Format]

	Instruction	Addressing mode	Operation code (EA) 1	Operation code (EA) 2	Operation code (EA) 3	Operation code (OP) 4	Operation code (OP) 5	Operation code (OP) 6
Data transfer instruction	MOV:G.B <EAs>, Rd	2	1 0 1 0 Sz r r r			1 0 0 0 0 rd rd rd		
	MOV:G.W <EAs>, Rd	2	1 0 1 0 Sz r r r			1 0 0 0 0 rd rd rd		
	MOV:G.B Rs, <EAd>	2	1 1 0 1 Sz r r r			1 0 0 1 0 rs rs rs		
	MOV:G.W Rs, <EAd>	2	1 1 0 1 Sz r r r			1 0 0 1 0 rs rs rs		
	MOV:G.B #xx:8, <EAd>	3	1 1 1 0 Sz r r r	disp		0 0 0 0 0 1 1 0	data	
	MOV:G.W #xx:8, <EAd>	3	1 1 1 0 Sz r r r	disp		0 0 0 0 0 1 1 0	data	
	MOV:G.W #xx:16, <EAd>	4	1 1 1 1 Sz r r r	disp (H)	disp (L)	0 0 0 0 0 1 1 1	data (H)	data (L)
	LDM.W @SP+, <register list>	2	1 0 1 1 Sz r r r			0 0 0 0 0 0 1 0	register list	
	STM.W <register list>,@–SP	2	1 1 0 0 Sz r r r			0 0 0 0 0 0 1 0	register list	
	XCH.W Rs, Rd	2				1 0 0 1 0 rd rd rd		
	SWAP.B Rd	2				0 0 0 1 0 0 0 0		
	MOVTPE.B Rs, <EAd>	3	0 0 0 0 Sz 1 0 0	address		0 0 0 0 0 0 0 0	1 0 0 1 0 rs rs rs	
	MOVTPE.B <EAs>, Rd	3	0 0 0 1 Sz 1 0 1	address (H)	address (L)	0 0 0 0 0 0 0 0	1 0 0 1 0 rd rd rd	
Arithmetic operation instruction	ADD:G.B <EAs>, Rd	2	0 0 0 0 Sz 1 0 0	data	data (L)	0 0 1 0 0 rd rd rd		
	ADD:G.W <EAs>, Rd	2	0 0 0 0 Sz 1 0 0	data (H)	data (L)	0 0 1 0 0 rd rd rd		
	ADD:Q.B #1, <EAd>*	2	0 0 0 0 1 1 0 0			0 0 0 0 1 0 0 0		
	ADD:Q.W #1, <EAd>*	2	0 0 0 0 1 1 0 0			0 0 0 0 1 0 0 0		
	ADD:Q.B #2, <EAd>*	2	0 0 0 0 1 1 0 1			0 0 0 0 1 0 0 1		
	ADD:Q.W #2, <EAd>*	2	0 0 0 0 1 1 0 1			0 0 0 0 1 0 0 1		
	ADD:Q.B #-1, <EAd>*	2	0 0 0 0 1 1 1 0			0 0 0 0 1 1 0 0		
	ADD:Q.W #-1, <EAd>*	2	0 0 0 0 1 1 1 0			0 0 0 0 1 1 0 0		
	ADD:Q.B #-2, <EAd>*	2	0 0 0 0 1 1 1 1			0 0 0 0 1 1 0 1		
	ADD:Q.W #-2, <EAd>*	2	0 0 0 0 1 1 1 1			0 0 0 0 1 1 0 1		
	ADDS.B <EAs>, Rd	2	0 0 0 0 1 0 1 1			0 0 1 0 1 rd rd rd		
	ADDS.W <EAs>, Rd	2	0 0 0 0 1 0 1 1			0 0 1 0 1 rd rd rd		
	ADDX.B <EAs>, Rd	2	0 0 0 0 1 0 0 0			1 0 1 0 0 rd rd rd		
	ADDX.W <EAs>, Rd	2	0 0 0 0 1 0 0 0			1 0 1 0 0 rd rd rd		

Note: * Short format instruction

Table A-1 (a) Machine Language Coding [General Format] (cont)

	Instruction	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16	Operation code (OP) 4	5	6
	Operation code (EA) 1	1 0 1 0 Sz r r r	1 1 0 1 Sz r r r	1 1 1 0 Sz r r r	1 1 1 1 Sz r r r	1 0 1 1 Sz r r r	1 1 0 0 Sz r r r	0 0 0 0 Sz 1 0 1	0 0 0 1 Sz 1 0 1	0 0 0 0 0 1 0 0	0 0 0 0 1 1 0 0			
	Operation code (EA) 2			disp	disp (H)			address	address (H)	data	data (H)			
	Operation code (EA) 3				disp (L)				address (L)		data (L)			
Arithmetic operation instruction	DADD.B Rs ,Rd											0 0 0 0 0 0 0 0	1 0 1 0 0 rd rd rd	
	SUB.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0 0 1 1 0 rd rd rd		
	SUB.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	0 0 1 1 0 rd rd rd		
	SUBS.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		0 0 1 1 1 rd rd rd		
	SUBS.W <EAs>,Rd	2	2	3	4	2	2	3	4		4	0 0 1 1 1 rd rd rd		
	SUBX.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		1 0 1 1 0 rd rd rd		
	SUBX.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	1 0 1 1 0 rd rd rd		
	DSUB.B Rs, Rd	3										0 0 0 0 0 0 0 0	1 0 1 1 0 rd rd rd	
	MULXU.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		1 0 1 0 1 rd rd rd		
	MULXU.X <EAs>, Rd	2	2	3	4	2	2	3	4		4	1 0 1 0 1 rd rd rd		
	DIVXU.B <EAs>, Rd	2	2	3	4	2	2	3	4	3		1 0 1 1 1 rd rd rd		
	DIVXU.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	1 0 1 1 1 rd rd rd		
	CMP:G.B <EAs>, Rd	2	3	4	5	3	3	4	5	3		0 1 1 1 0 rd rd rd		
	CMP:G.W <EAs>, Rd	2	2	3	4	2	2	3	4		4	0 1 1 1 0 rd rd rd		
	CMP:G.B #xx, <EAd>		3	4	5	3	3	4	5			0 0 0 0 0 1 0 0	data	
	CMP:G.W #xx, <EAd>		4	5	6	4	4	5	6			0 0 0 0 0 1 0 1	data (H)	data (L)
	EXTS.B Rd	2										0 0 0 1 0 0 0 1		
	EXTU.B Rd	2										0 0 0 1 0 0 1 0		
	TST.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 1 0		
	TST.W <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 1 0		
	NEG.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 0 0		
	NEG.W <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 0 0		
	CLR.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 0 1 1		
	CLR.W <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 0 1 1		
	TAS.B <EAd>	2	2	3	4	2	2	3	4			0 0 0 1 0 1 1 1		

Table A-1 (a) Machine Language Coding [General Format] (cont)

	Instruction	Addressing mode	Operation code (EA) 1	2	3	Operation code (OP) 4	5	6
Shift instruction	SHAL.B <EAd>	Rn	1 0 1 0 Sz r r r			0 0 0 1 1 0 0 0		
	SHAL.W <EAd>	@Rn	1 1 0 1 Sz r r r			0 0 0 1 1 0 0 0		
	SHAR.B <EAd>	@(d:8, Rn)	1 1 1 0 Sz r r r	disp		0 0 0 1 1 0 0 1		
	SHAR.W <EAd>	@(d:16, Rn)	1 1 1 1 Sz r r r	disp (H)	disp (L)	0 0 0 1 1 0 0 1		
	SHLL.B <EAd>	@–Rn	1 0 1 1 Sz r r r			0 0 0 1 1 0 1 0		
	SHLL.W <EAd>	@Rn+	1 1 0 0 Sz r r r			0 0 0 1 1 0 1 0		
	SHLR.B <EAd>	@aa:8	0 0 0 0 Sz 1 0 1	address		0 0 0 1 1 0 1 1		
	SHLR.W <EAd>	@aa:16	0 0 0 1 Sz 1 0 1	address (H)	address (L)	0 0 0 1 1 0 1 1		
	ROTL.B <EAd>	#xx:8	0 0 0 0 0 1 0 0	data		0 0 0 1 1 1 0 0		
	ROTL.W <EAd>	#xx:16	0 0 0 0 1 1 0 0	data (H)	data (L)	0 0 0 1 1 1 0 0		
	ROTR.B <EAd>					0 0 0 1 1 1 0 1		
	ROTR.W <EAd>					0 0 0 1 1 1 0 1		
	ROTXL.B <EAd>					0 0 0 1 1 1 1 0		
	ROTXL.W <EAd>					0 0 0 1 1 1 1 0		
	ROTXR.B <EAd>					0 0 0 1 1 1 1 1		
	ROTXR.W <EAd>					0 0 0 1 1 1 1 1		
Logic operation instruction	AND.B <EAs>, Rd					0 1 0 1 0 rd rd rd		
	AND.W <EAs>, Rd					0 1 0 1 0 rd rd rd		
	OR.B.B <EAs>, Rd					0 1 0 0 0 rd rd rd		
	OR.B.W <EAs>, Rd					0 1 0 0 0 rd rd rd		
	XOR.B <EAs>, Rd					0 1 1 0 0 rd rd rd		
	XOR.W <EAs>, Rd					0 1 1 0 0 rd rd rd		
	NOT.B <EAd>					0 0 0 1 0 1 0 1		
	NOT.W <EAd>					0 0 0 1 0 1 0 1		

Table A-1 (a) Machine Language Coding [General Format] (cont)

	Instruction	Addressing mode	Operation code (EA) 1	Operation code (EA) 2	Operation code (EA) 3	Operation code (OP) 4	Operation code (OP) 5	Operation code (OP) 6
Bit manipulate instruction	BSET.B #xx, <EAd>	Rn	2	2	3	4	2	2	3	4			1 1 0 0 (data)		
	BSET.W #xx, <EAd>	@Rn	2	2	3	4	2	2	3	4			1 1 0 0 (data)		
	BSET.B Rs, <EAd>	@(d:8, Rn)	2	2	3	4	2	2	3	4			0 1 0 0 1 rs rs rs		
	BSET.W Rs, <EAd>	@(d:16, Rn)	2	2	3	4	2	2	3	4			0 1 0 0 1 rs rs rs		
	BCLR.B #xx, <EAd>	@–Rn	2	2	3	4	2	2	3	4			1 1 0 1 (data)		
	BCLR.W #xx, <EAd>	@Rn+	2	2	3	4	2	2	3	4			1 1 0 1 (data)		
	BCLR.B Rs, <EAd>	@aa:8	2	2	3	4	2	2	3	4			0 1 0 1 1 rs rs rs		
	BCLR.W Rs, <EAd>	@aa:16	2	2	3	4	2	2	3	4			0 1 0 1 1 rs rs rs		
	BTST.B #xx, <EAd>	#xx:8	2	2	3	4	2	2	3	4			1 1 1 1 (data)		
	BTST.W #xx, <EAd>	#xx:16	2	2	3	4	2	2	3	4			1 1 1 1 (data)		
	BTST.B Rs, <EAd>		2	2	3	4	2	2	3	4			0 1 1 1 1 rs rs rs		
	BTST.W Rs, <EAd>		2	2	3	4	2	2	3	4			0 1 1 1 1 rs rs rs		
	BNOT.B #xx, <EAd>		2	2	3	4	2	2	3	4			1 1 1 0 (data)		
	BNOT.W #xx, <EAd>		2	2	3	4	2	2	3	4			1 1 1 0 (data)		
	BNOT.B Rs, <EAd>		2	2	3	4	2	2	3	4			0 1 1 0 1 rs rs rs		
	BNOT.W Rs, <EAd>		2	2	3	4	2	2	3	4			0 1 1 0 1 rs rs rs		
System control instruction	LDC.B <EAs>, CR		2	2	3	4	2	2	3	4	3		1 0 0 0 1 c c c		
	LDC.W <EAs>, CR		2	2	3	4	2	2	3	4		4	1 0 0 0 1 c c c		
	STC.B CR, <EAd>		2	2	3	4	2	2	3	4			1 0 0 1 1 c c c		
	STC.W CR, <EAd>		2	2	3	4	2	2	3	4			1 0 0 1 1 c c c		
	ANDC.B #xx:8, CR										3		0 1 0 1 1 c c c		
	ANDC.W #xx:16, CR											4	0 1 0 1 1 c c c		
	ORC.B #xx:8, CR										3		0 1 0 0 1 c c c		
	ORC.W #xx:16, CR											4	0 1 0 0 1 c c c		
	XORC.B #xx:8, CR										3		0 1 1 0 1 c c c		
	XORC.W #xx:16, CR											4	0 1 1 0 1 c c c		

Table A-1 (b) Machine Language Coding [Special Format: Short Format]

Instruction	Bytes	Operation code			
		1	2	3	4
MOV:E,B #xx:8,Rd	2	01010rdrdrd	data		
MOV:I.W #xx:16,Rd	3	01011rdrdrd	data (H)	data (L)	
MOV:L.B @aa:8,Rd	2	01100rdrdrd	address (L)		
MOV:L.W @aa:8,Rd	2	01101rdrdrd	address (L)		
MOV:S.B Rs,@aa:8	2	01110rsrsrs	address (L)		
MOV:S.W Rs,@aa:8	2	01111rsrsrs	address (L)		
MOV:F.B @(d:8,R6),Rd	2	10000rdrdrd	disp		
MOV:F.W @(d:8,R6),Rd	2	10001rdrdrd	disp		
MOV:F.B Rs @(d:8,R6)	2	10010rsrsrs	disp		
MOV:F.W Rs,@(d:8,R6)	2	10011rsrsrs	disp		
CMP:E #xx:8,Rd	2	01000rdrdrd	data		
CMP:I #xx:16,Rd	3	01001rdrdrd	data (H)	data (L)	

Table A-1 (c) Machine Language Coding [Special Format: Branch Instruction]

Instruction		Bytes	Operation code			
			1	2	3	4
Bcc d:8	BRA (BT)	2	00100000	disp		
	BRN (BF)		00100001	disp		
	BHI		00100010	disp		
	BLS		00100011	disp		
	BCC (BHS)		00100100	disp		
	BCS (BLO)		00100101	disp		
	BNE		00100110	disp		
	BEQ		00100111	disp		
	BVC		00101000	disp		
	BVS		00101001	disp		
	BPL		00101010	disp		
	BMI		00101011	disp		
	BGE		00101100	disp		
	BLT		00101101	disp		
	BGT		00101110	disp		
	BLE		00101111	disp		
Bcc d:16	BRA (BT)	3	00110000	disp (H)	disp (L)	
	BRN (BF)		00110001	disp (H)	disp (L)	
	BHI		00110010	disp (H)	disp (L)	
	BLS		00110011	disp (H)	disp (L)	
	BCC (BHS)		00110100	disp (H)	disp (L)	
	BCS (BLO)		00110101	disp (H)	disp (L)	
	BNE		00110110	disp (H)	disp (L)	
	BEQ		00110111	disp (H)	disp (L)	
	BVC		00111000	disp (H)	disp (L)	
	BVS		00111001	disp (H)	disp (L)	
	BPL		00111010	disp (H)	disp (L)	
	BMI		00111011	disp (H)	disp (L)	
	BGE		00111100	disp (H)	disp (L)	
	BLT		00111101	disp (H)	disp (L)	
	BGT		00111110	disp (H)	disp (L)	
	BLE		00111111	disp (H)	disp (L)	
JMP @Rn		2	00010001	11010rrr		
JMP @aa:16		3	00010000	address (H)	address (L)	

Table A-1 (c) Machine Language Coding [Special Format: Branch Instruction] (cont)

Instruction		Bytes	Operation code			
			1	2	3	4
JMP @(d:8,Rn)		3	00010001	11100rrr	disp	
JMP @(d:16,Rn)		4	00010001	11110rrr	disp (H)	disp (L)
BSR d:8		2	00001110	disp		
BSR d:16		3	00011110	disp (H)	disp (L)	
JSR @Rn		2	00010001	11011rrr		
JSR @aa:16		3	00011000	address (H)	address (L)	
JSR @(d:8,Rn)		3	00010001	11101rrr	disp	
JSR @(d:16,Rn)		4	00010001	11111rrr	disp (H)	disp (L)
RTS		1	00011001			
RTD #xx:8		2	00010100	data		
RTD #xx:16		3	00011100	data (H)	data (L)	
SCB/cc Rn,disp	SCB/F	3	00000001	10111rrr	disp	
	SCB/NE		00000110	10111rrr	disp	
	SCB/EQ		00000111	10111rrr	disp	
PJMP @aa:24		4	00010011	page	address (H)	address (L)
PJMP @Rn		2	00010001	11000rrr		
PJSR @aa:24		4	00000011	page	address (H)	address (L)
PJSR @Rn		2	00010001	11001rrr		
PRTS		2	00010001	00011001		
PRTD #xx:8		3	00010001	00010100	data	
PRTD #xx:16		4	00010001	00011100	data (H)	data (L)

Table A-1 (d) Machine Language Coding [Special Format: System Control Instructions]

Instruction	Bytes	Operation code			
		1	2	3	4
TRAPA #xx	2	00001000	0001 #VEC		
TRAP/VS	1	00001001			
RTE	1	00001010			
LINK FP,#xx:8	2	00010111	data		
LINK FP,#xx:16	3	00011111	data (H)	data (L)	
UNLK FP	1	00001111			
SLEEP	1	00011010			
NOP	1	00000000			

A.3 Operation Code Map

Tables A-2 through A-6 are maps of the operation codes. Table A-2 shows the meaning of the first byte of the instruction code, indicating both operation codes and addressing modes. Tables A-2 through A-6 indicate the meanings of operation codes in the second and third bytes.

Table A-2 Operation Codes in Byte 1

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SCB/F See Tbl. A-6	LDM	PJSR @aa:24	#xx:8 See Tbl. A-5	#aa:8.B See Tbl. A-4	SCB/NE See Tbl. A-6	SCB/EQ See Tbl. A-6	TRAPA	TRAP/VS	RTE		#xx:16 See Tbl. A-5	@aa:8.W See Tbl. A-4	BSR d:8	UNLK
1	JMP	See Tbl. A-6 *	STM	PJMP @aa:24	RTD #xx:8	@aa:16.B See Tbl. A-4		LINK #xx:8	JSR	RTS	SLEEP		RTD #xx:16	@aa:16.W See Tbl. A-4	BSR d:16	LINK #xx:16
2	BRA d:8	BRN	BHI	BLS	Bcc	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
3	BRA d:16	BRN	BHI	BLS	Bcc	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
4	CMP:E #xx:8, Rn								CMP:I #xx:16, Rn							
	R0	R1	R2	R3	R4	R5	R6	R7	R0	R1	R2	R3	R4	R5	R6	R7
5	MOV:E #xx:8, Rn								MOV:I #xx:16, Rn							
6	MOV:L.B @aa:8, Rn								MOV:L.W @aa:8, Rn							
7	MOV:S.B Rn, @aa:8								MOV:S.W Rn, @aa:8							
8	MOV:F.B @(d:8, R6), Rn								MOV:F.W @(:8, R6), Rn							
9	MOV:F.B Rn, @(d:8, R6)								MOV:F.W Rn, @(d:8,R6)							
A			Rn			(Byte)	See Table A-3				Rn		(Word)		See Table A-3	
B			@–Rn			(Byte)	See Table A-4				@–Rn		(Word)		See Table A-4	
C			@Rn+			(Byte)	See Table A-4				@Rn+		(Word)		See Table A-4	
D			@Rn			(Byte)	See Table A-4				@Rn		(Word)		See Table A-4	
E			@(d:8,Rn)			(Byte)	See Table A-4				@(d:8,Rn)		(Word)		See Table A-4	
F			@(d:16,Rn)			(Byte)	See Table A-4				@(d:16,Rn)		(Word)		See Table A-4	

Notes:

References to tables A-3 through A-6 indicate that the instruction code has one or more additional bytes, described in those tables.

* H'11 is the first operation code byte of the following instructions:
JMP, JSR, PJSR (register indirect addressing mode)
JMP, JSR (register indirect addressing mode with displacement)
PRTS, PRTD (all addressing modes)

Table A-3 Operation Codes in Byte 2 (Axxx)

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	See Tbl. A-6*								ADD:Q #1	ADD:Q #2			ADD:Q #-1	ADD:Q #-2		
1	SWAP	EXTS	EXTU	CLR	NEG	NOT	TST	TAS	SHAL	SHAR	SHLL	SHLR	ROTL	ROTR	ROTXL	ROTXR
2	R0	R1	R2	ADD R3	R4	R5	R6	R7	R0	R1	ADDS R2	R3	R4	R5	R6	R7
3				SUB							SUBS					
4				OR							BSET (Register indirect specification of bit number)					
5				AND							BCLR (Register indirect specification of bit number)					
6				XOR							BNOT (Register indirect specification of bit number)					
7				CMP							BTST (Register indirect specification of bit number)					
8				MOV							LDC					
9				XCH							STC					
A				ADDX							MULXU					
B				SUBX							DIVXU					
C	b0	b1	b2	b3	b4	b5	BSET (Immediate specification of bit number) b6	b7	b8	b9	b10	b11	b12	b13	b14	b15
D							BCLR (Immediate specification of bit number)									
E							BNOT (Immediate specification of bit number)									
F							BTST (Immediate specification of bit number)									

Note: * Prefix code for DADD, DSUB, MOVTPE, and MOVFPE. The operation code is in byte 3, given in table A-6.

Table A-4 Operation Codes in Byte 2 (05xx, 15xx, 0Dxx, 1Dxx, Bxxx, Cxxx, Dxxx, Exxx, Fxxx)

HI \ LO	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	See Tbl. A-6*				CMP #xx:8	CMP #xx:16	MOV #xx:8	MOV #xx:16	ADD:Q #1	ADD:Q #2			ADD:Q #-1	ADD:Q #-2		
1				CLR	NEG	NOT	TST	TAS	SHAL	SHAR	SHLL	SHLR	ROTL	ROTR	ROTXL	ROTXR
2				ADD						ADDS						
3				SUB						SUBS						
4				OR						BSET (Register indirect specification of bit number)						
5				AND						BCLR (Register indirect specification of bit number)						
6				XOR						BNOT (Register indirect specification of bit number)						
7				CMP						BTST (Register indirect specification of bit number)						
8				MOV	(load)					LDC						
9				MOV	(store)					STC						
A				ADDX						MULXU						
B				SUBX						DIVXU						
C							BSET (Immediate specification of bit number)									
D							BCLR (Immediate specification of bit number)									
E							BNOT (Immediate specification of bit number)									
F							BTST (Immediate specification of bit number)									

Note: * Prefix code for DADD, DSUB, MOVTPE, and MOVFPE. The operation code is in byte 3, given in table A-6.

Table A-5 Operation Codes in Byte 2 (04xx, 0Cxx)

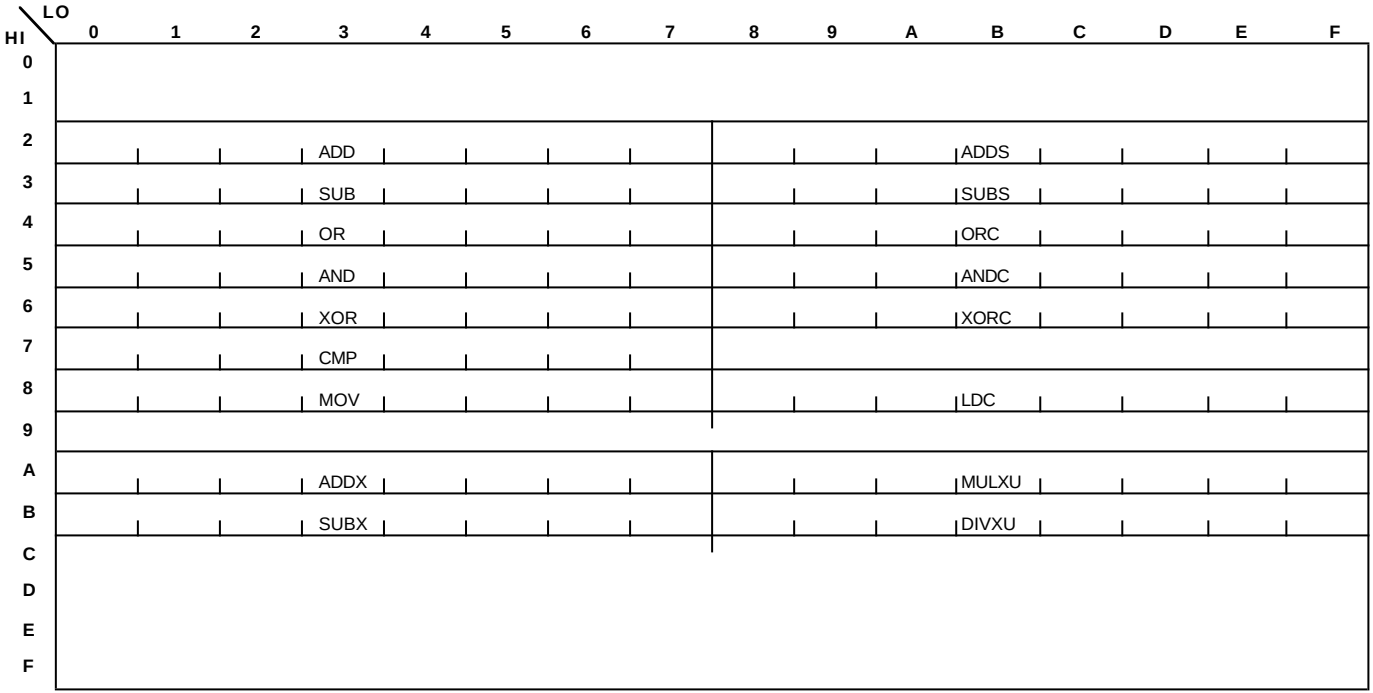
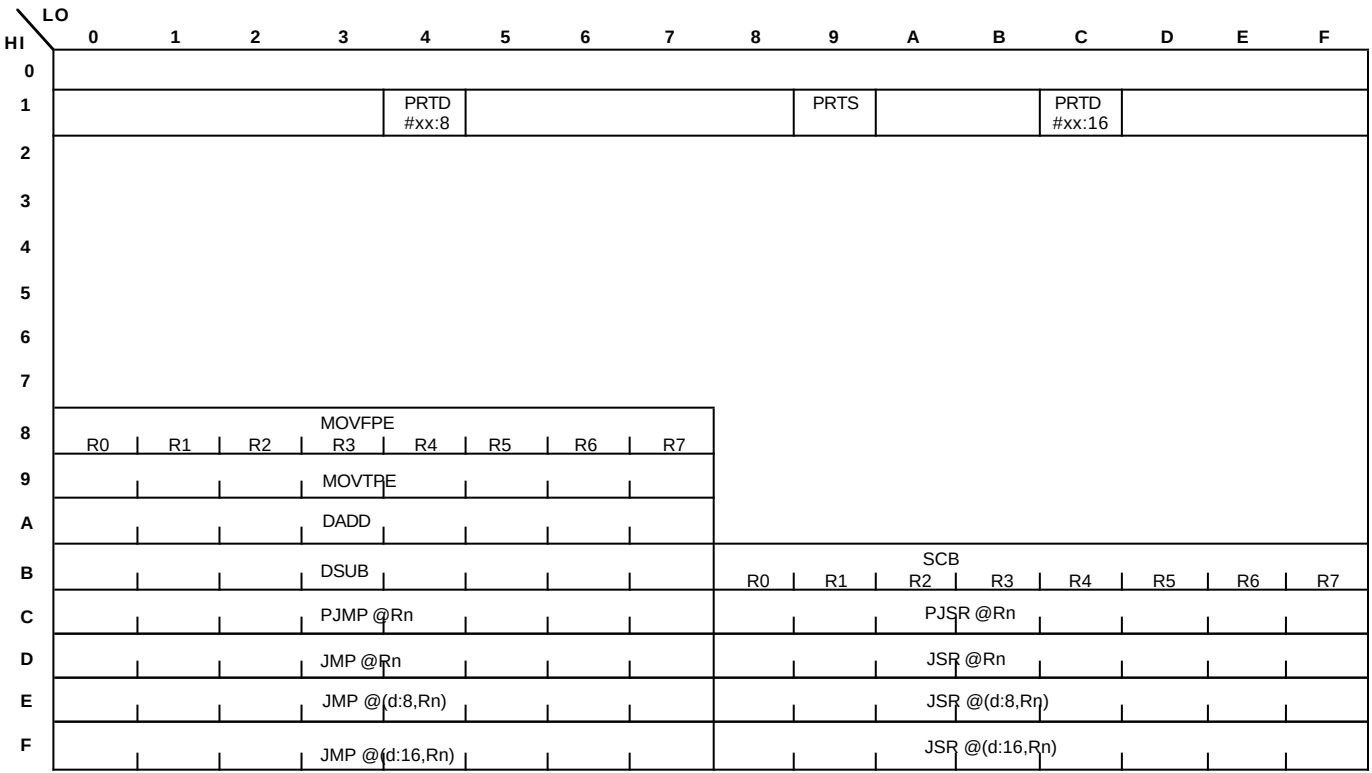


Table A-6 Operation Codes in Bytes 2 and 3 (11xx, 01xx, 06xx, 07xx, xx00xx)



A.4 Instruction Execution Cycles

Tables A-7 (1) through (6) list the number of cycles required by the CPU to execute each instruction in each addressing mode.

The meaning of the symbols in the tables is explained below. The values of I, J, and K are used to calculate the number of execution cycles when off-chip memory is accessed for an instruction fetch or operand read/write. The formulas for these calculations are given next.

A.4.1 Calculation of Instruction Execution States

One state is one system clock cycle (ø). When ø = 10 MHz, one state = 100 ns.

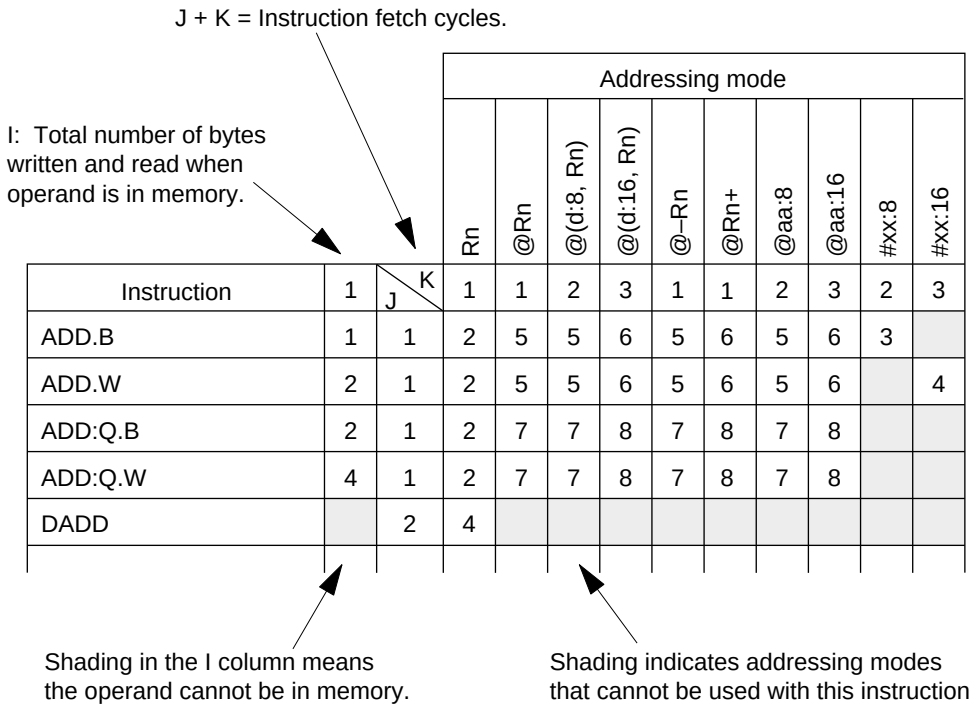
Instruction Fetch	Operand Read/Write	Number of States	
On-chip memory *1	On-chip memory, general register, or no operand	(Value given in table A-7) + (Value in table A-8)	
	On-chip memory module or off-chip memory *2	Byte	(Value in table A-7) + (Value in table A-8) + I
		Word	(Value in table A-7) + (Value in table A-8) + 2I
Off-chip memory *2	On-chip memory, general register, or no operand	(Value given in table A-7) + 2(J + K)	
	On-chip supporting module or off-chip memory *2	Byte	(Value in table A-7) + I + 2(J + K)
		Word	(Value in table A-7) + 2(I + J + K)

Notes: *1 When the instruction is fetched from on-chip memory (ROM or RAM), the number of execution states varies by 1 or 2 depending of whether the instruction is stored at an even or odd address. This difference must be noted when software is used for timing, and in other cases in which the exact number of states is important.

*2 If wait states are inserted in access to external memory, add the necessary number of cycles.

A.4.2 Tables of Instruction Execution Cycles

Tables A-7 (1) through (6) should be read as shown below:



• **Examples of Calculation of Number of States Required for Execution**

(Example 1) Instruction fetch from on-chip memory: ADD:G.W @R0, R1

Operand Read/Write	Start Addr.	Assembler Notation			Table A-7 + Table A-8	Number of States
		Address	Code	Mnemonic		
On-chip memory or general register	Even	H'0100	H'D821	ADD:G.W @R0, R1	5 + 1	6
	Odd	H'0101	H'D821	ADD:G.W @R0, R1	5 + 0	5

(Example 2) Instruction fetch from on-chip memory: JSR @R0

Branch Addr.	Assembler Notation			Table A-7 + Table A-8 + 2I	Number of States
	Address	Code	Mnemonic		
Even	H'FC00	H'11D8	JSR @R0	9 + 0 + 2 × 2	13
Odd	H'FC01	H'11D8	JSR @R0	9 + 1 + 2 × 2	14

(Example 3) Instruction fetch from external memory

Operand Read/Write	Assembler Notation			Table A-7 + 2(J + K)	Number of States
	Address	Code	Mnemonic		
On-chip memory or general register	H'9002	H'D821	ADD:G.W @R0, R1	5 + 2 × (1 + 1)	9
On-chip module or external memory	H'9002	H'D821	ADD:G.W @R0, R1	5 + 2 × (2 + 1 + 1)	13

Table A-7 Instruction Execution Cycles (1)

Instruction		1	J \ K	Addressing mode									
				Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
			1	1	1	2	3	1	1	2	3	2	3
ADD:G.B <EAs>, Rd		1	1	2	5	5	6	5	6	5	6	3	
ADD:G.W <EAs>, Rd		2	1	2	5	5	6	5	6	5	6		4
ADD:Q.B #xx, <EAd>		2	1	2	7	7	8	7	8	7	8		
ADD:Q.W #xx, <EAd>		4	1	2	7	7	8	7	8	7	8		
ADDS.B <EAs>, Rd		1	1	3	5	5	6	5	6	5	6	3	
ADDS.W <EAs>, Rd		2	1	3	5	5	6	5	6	5	6		4
ADDX.B <EAs>, Rd		1	1	2	5	5	6	5	6	5	6	3	
ADDX.W <EAs>, Rd		2	1	2	5	5	6	5	6	5	6		4
AND.B <EAs>, Rd		1	1	2	5	5	6	5	6	5	6	3	
AND.W <EAs>, Rd		2	1	2	5	5	6	5	6	5	6		4
ANDC #xx, CR			1									5	9
BCLR.B #xx, <EAd>	*	2	1	4	7	7	8	7	8	7	8		
BCLR.W #xx, <EAd>	*	4	1	4	7	7	8	7	8	7	8		
BNOT.B #xx, <EAd>	*	2	1	4	7	7	8	7	8	7	8		
BNOT.W #xx, <EAd>	*	4	1	4	7	7	8	7	8	7	8		
BSET.B #xx, <EAd>	*	2	1	4	7	7	8	7	8	7	8		
BSET.W #xx, <EAd>	*	4	1	4	7	7	8	7	8	7	8		
BTST.B #xx, <EAd>	*	1	1	3	5	5	6	5	6	5	6		
BTST.W #xx, <EAd>	*	2	1	3	5	5	6	5	6	5	6		
CLR.B <EAd>		1	1	2	5	5	6	5	6	5	6		
CLR.W <EAd>		2	1	2	5	5	6	5	6	5	6		
CMP:G.B <EAs>, Rd		1	1	2	5	5	6	5	6	5	6	3	
CMP:G.W <EAs>, Rd		2	1	2	5	5	6	5	6	5	6		4
CMP:G.B #XX:8, <EA>		1	2		6	6	7	6	7	6	7		
CMP:G.B #XX:16, <EA>		2	3		7	7	8	7	8	7	8		

* Rs can also be specified as the source operand.

Table A-7 Instruction Execution Cycles (2)

			Addressing mode									
			Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
Instruction	1	J \ K	1	1	2	3	1	1	2	3	2	3
CMP:E #xx:8, Rd		0									2	
CMP:I #xx:16, Rd		0										3
DADD Rs, Rd		2	4									
DIVXU.B <EAs>, Rd	1	1	20	23	23	24	23	24	23	24	21	
DIVXU.W <EAs>, Rd	2	1	26	29	29	30	29	30	29	30		28
DSUB Rs, Rd		2	4									
EXTS Rd		1	3									
EXTU Rd		1	3									
LDC.B <EAs>, CR	1	1	3	6	6	7	6	7	6	7	4	
LDC.W <EAs>, CR	2	1	4	7	7	8	7	8	7	8		6
MOV:G.B	1	1	2	5	5	6	5	6	5	6	3	
MOV:G.W	2	1	2	5	5	6	5	6	5	6		4
MOV.G.B #xx:8, <EAd>	1	2		7	7	8	7	8	7	8		
MOV.G.B #xx:16, <EAd>	2	3		8	8	9	8	9	8	9		
MOV:E #xx:8, Rd		0									2	
MOV:I #xx:16, Rd		0										3
MOV:L.B @aa:8, Rd	1	0							5			
MOV:L.W @aa:8, Rd	2	0							5			
MOV:S.B Rs,@aa:8	1	0							5			
MOV:S.W Rs,@aa:8	2	0							5			
MOV:F.B @(d:8, R6), Rd	1	0			5							
MOV:F.W @(d:8, R6), Rd	2	0			5							
MOV:F.B Rs, @(d:8, R6)	1	0			5							
MOV:F.W Rs, @(d:8, R6)	2	0			5							

Table A-7 Instruction Execution Cycles (3)

Instruction	1	J \ K	Addressing mode									
			Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
			1	1	2	3	1	1	2	3	2	3
MOVFPE <EAs>, Rd	0	2		13 — 20	13 — 20	14 — 21	13 — 20	14 — 21	13 — 20	14 — 21		
MOVTPE Rs, <EAd>	0	2		13 — 20	13 — 20	14 — 21	13 — 20	14 — 21	13 — 20	14 — 21		
MULXU.B <EAs>, Rd	1	1	16	19	19	20	19	20	19	20	18	
MULXU.W <EAs>, Rd	2	1	23	25	25	26	25	26	25	26		25
NEG.B <EAd>	2	1	2	7	7	8	7	8	7	8		
NEG.W <EAd>	4	1	2	7	7	8	7	8	7	8		
NOT.B <EAd>	2	1	2	7	7	8	7	8	7	8		
NOT.W <EAd>	4	1	2	7	7	8	7	8	7	8		
OR.B <EAs>, Rd	1	1	2	5	5	6	5	6	5	6	3	
OR.W <EAs>, Rd	2	1	2	5	5	6	5	6	5	6		4
ORC #xx, CR		1									5	9
ROTL.B <EAd>	2	1	2	7	7	8	7	8	7	8		
ROTL.W <EAd>	4	1	2	7	7	8	7	8	7	8		
ROTR.B <EAd>	2	1	2	7	7	8	7	8	7	8		
ROTR.W <EAd>	4	1	2	7	7	8	7	8	7	8		
ROTXL.B <EAd>	2	1	2	7	7	8	7	8	7	8		
ROTXL.W <EAd>	4	1	3	7	7	8	7	8	7	8		
ROTXR.B <EAd>	2	1	2	7	7	8	7	8	7	8		
ROTXR.W <EAd>	4	1	2	7	7	8	7	8	7	8		
SHAL.B <EAd>	2	1	2	7	7	8	7	8	7	8		
SHAL.W <EAd>	4	1	2	7	7	8	7	8	7	8		
SHAR.B <EAd>	2	1	2	7	7	8	7	8	7	8		
SHAR.W <EAd>	4	1	2	7	7	8	7	8	7	8		
SHLL.B <EAd>	2	1	2	7	7	8	7	8	7	8		
SHLL.W <EAd>	4	1	2	7	7	8	7	8	7	8		

Table A-7 Instruction Execution Cycles (4)

Instruction	1	J \ K	Addressing mode: Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
SHLR.B <EAd>	2	1	2	7	7	8	7	8	7	8		
SHLR.W <EAd>	4	1	2	7	7	8	7	8	7	8		
STC.B CR, <EAd>	1	1	2	7	7	8	7	8	7	8		
STC.W CR, <EAd>	2	1	2	7	7	8	7	8	7	8		
SUB.B <EAs>, Rd	1	1	2	5	5	6	5	6	5	6	3	
SUB.W <EAs>, Rd	2	1	2	5	5	6	5	6	5	6		4
SUBS.B <EAs>, Rd	1	1	3	5	5	6	5	6	5	6	3	
SUBS.W <EAs>, Rd	2	1	3	5	5	6	5	6	5	6		4
SUBX.B <EAs>, Rd	1	1	2	5	5	6	5	6	5	6	3	
SUBX.W <EAs>, Rd	2	1	2	5	5	6	5	6	5	6		4
SWAP Rd		1	3									
TAS <EAd>	2	1	4	7	7	8	7	8	7	8		
TST.B <EAd>	1	1	2	5	5	6	5	6	5	6		
TST.W <EAd>	2	1	2	5	5	6	5	6	5	6		
XCH Rs , Rd		1	4									
XOR.B <EAs>, Rd	1	1	2	5	5	6	5	6	5	6	3	
XOR.W <EAs>, Rd	4	1	4	5	5	6	5	6	5	6		4
XORC #xx, CR		1									5	9

		* ↱										
DIVXU.B	Zero divide, minimum mode	6/7	1	20	23	23	24	23	24	23	24	21	
DIVXU.B	Zero divide, maximum mode	10/11	1	25	28	28	29	28	29	28	29	21	
DIVXU.W	Zero divide, minimum mode	6/8	1	20	23	23	24	23	24	23	24		27
DIVXU.W	Zero divide, maximum mode	10/12	1	25	28	28	29	28	29	28	29		27
DIVXU.B	Overflow	1	1	8	11	11	12	11	12	11	12	9	
DIVXU.W	Overflow	2	1	8	11	11	12	11	12	11	12		10

* ↱ For register and immediate operands

/ For memory operand

Table A-7 Instruction Execution Cycles (5)

Instruction	(Condition)	Execution Cycles	I	J + K
Bcc d:8	Condition false, branch not taken	3		2
	Condition true, branch taken	7		5
Bcc d:16	Condition false, branch not taken	3		3
	Condition true, branch taken	7		6
BSR	d:8	9	2	4
	d:16	9	2	5
JMP	@aa:16	7		5
	@Rn	6		5
	@(d:8, Rn)	7		5
	@(d:16, Rn)	8		6
JSR	@aa:16	9	2	5
	@Rn	9	2	5
	@(d:8, Rn)	9	2	5
	@(d:16, Rn)	10	2	6
LDM		6 + 4n*	2n	2
LINK	#xx:8	6	2	2
	#xx:16	7	2	3
NOP		2		1
RTD	#xx:8	9	2	4
	#xx:16	9	2	5
RTE	Minimum mode	13	4	4
	Maximum mode	15	6	4
RTS		8	2	4
SCB	Condition false, branch not taken	3		3
	Count = –1, branch not taken	4		3
	Other than the above, branch taken	8		6
SLEEP	Cycles preceding transition to power-down mode	2		0
STM		6 + 3n*	2n	2

* n is the number of registers specified in the register list.

Table A-7 Instruction Execution Cycles (6)

Instruction	(Condition)	Execution Cycles	I	J + K
TRAPA	Minimum mode	17	6	4
	Maximum mode	22	10	4
TRAP/VS	V = 0, trap not taken	3		1
	V = 1, trap taken, minimum mode	18	6	4
	V = 1, trap taken, maximum mode	23	10	4
UNLK		5	2	1
PJMP	@aa:24	9		6
	@Rn	8		5
PJSR	@aa:24	15	4	6
	@Rn	13	4	5
PRTS		12	4	5
PRTD	#xx:8	13	4	5
	#xx:16	13	4	6

Table A-8 (a) Adjusted Value (Branch Instruction)

Instruction	Address	Adjusted Value
BSR, JMP, JSR, RTS, RTD, RTE	even	0
TRAPA, PJMP, PJSR, PRTS, PRTD	odd	1
Bcc, SCB, TRAP/VS (When branch	even	0
is taken)	odd	1

Table A-8 (b) Adjusted Value (Other Instructions by Addressing Modes)

Instruction	Start address	Rn	@Rn	@(d:8, Rn)	@(d:16, Rn)	@–Rn	@Rn+	@aa:8	@aa:16	#xx:8	#xx:16
MOV.B #xx:8, <EA>	even		1	1	1	1	1	1	1		
	odd		1	1	1	1	1	1	1		
MOV.W #xx:16, <EA>	even		2	0	2	2	2	0	2		
	odd		0	2	0	0	0	2	0		
Instruction other than above	even	0	1	0	1	1	1	0	1	0	0
	odd	0	0	1	0	0	0	1	0	0	0

Appendix B Register Field

B.1 Register Addresses and Bit Names

Addr. (upper byte)	Addr. (lower byte)	Register Name	Bit Names								
			Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'FE	H'80	P1DDR	$P1_7$DDR	$P1_6$DDR	$P1_5$DDR	$P1_4$DDR	$P1_3$DDR	$P1_2$DDR	$P1_1$DDR	$P1_0$DDR	Port 1
	H'81	P2DDR	—	—	—	$P2_4$DDR	$P2_3$DDR	$P2_2$DDR	$P2_1$DDR	$P2_0$DDR	Port 2
	H'82	P1DR	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$	Port 1
	H'83	P1DR	—	—	—	$P2_4$	$P2_3$	$P2_2$	$P2_1$	$P2_0$	Port 2
	H'84	P3DDR	$P3_7$DDR	$P3_6$DDR	$P3_5$DDR	$P3_4$DDR	$P3_3$DDR	$P3_2$DDR	$P3_1$DDR	$P3_0$DDR	Port 3
	H'85	P4DDR	$P4_7$DDR	$P4_6$DDR	$P4_5$DDR	$P4_4$DDR	$P4_3$DDR	$P4_2$DDR	$P4_1$DDR	$P4_0$DDR	Port 4
	H'86	P3DR	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$	Port 3
	H'87	P4DR	$P4_7$	$P4_6$	$P4_5$	$P4_4$	$P4_3$	$P4_2$	$P4_1$	$P4_0$	Port 4
	H'88	P5DDR	$P5_7$DDR	$P5_6$DDR	$P5_5$DDR	$P5_4$DDR	$P5_3$DDR	$P5_2$DDR	$P5_1$DDR	$P5_0$DDR	Port 5
	H'89	P6DDR	—	—	—	—	$P6_3$DDR	$P6_2$DDR	$P6_1$DDR	$P6_0$DDR	Port 6
	H'8A	P5DR	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$	Port 5
	H'8B	P6DR	—	—	—	—	$P6_3$	$P6_2$	$P6_1$	$P6_0$	Port 6
	H'8C	P7DDR	$P7_7$DDR	$P7_6$DDR	$P7_5$DDR	$P7_4$DDR	$P7_3$DDR	$P7_2$DDR	$P7_1$DDR	$P7_0$DDR	Port 7
	H'8D	—	—	—	—	—	—	—	—	—	—
	H'8E	P7DR	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$	Port 7
	H'8F	P8DR	$P8_7$	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$	Port 8
H'FE	H'90	TCR	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0	FRT 1
	H'91	TCSR	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	CCLRA	
	H'92	FRC(H)									
	H'93	FRC(L)									
	H'94	OCRA(H)									
	H'95	OCRA(L)									
	H'96	OCRB(H)									
	H'97	OCRB(L)									
	H'98	ICR(H)									
	H'99	ICR(L)									
	H'9A	—	—	—	—	—	—	—	—	—	
	H'9B	—	—	—	—	—	—	—	—	—	
	H'9C	—	—	—	—	—	—	—	—	—	
	H'9D	—	—	—	—	—	—	—	—	—	
	H'9E	—	—	—	—	—	—	—	—	—	
	H'9F	—	—	—	—	—	—	—	—	—	

Note: (Continued on next page)

FRT1: Free-Running Timer channel 1

(Continued from preceding page)

Addr. (upper byte)	Addr. (lower 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	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0	
	H'A1	TCSR	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	CCLRA	
	H'A2	FRC(H)									
	H'A3	FRC(L)									
	H'A4	OCRA(H)									
	H'A5	OCRA(L)									
	H'A6	OCRB(H)									
H'FE	H'A7	OCRB(L)									
	H'A8	ICR(H)									FRT2
	H'A9	ICR(L)									
	H'AA	—	—	—	—	—	—	—	—	—	
	H'AB	—	—	—	—	—	—	—	—	—	
	H'AC	—	—	—	—	—	—	—	—	—	
	H'AD	—	—	—	—	—	—	—	—	—	
	H'AE	—	—	—	—	—	—	—	—	—	
	H'AF	—	—	—	—	—	—	—	—	—	
	H'B0	TCR	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	CKS1	CKS0	
	H'B1	TCSR	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	CCLRA	
	H'B2	FRC(H)									
	H'B3	FRC(L)									
	H'B4	OCRA(H)									
	H'B5	OCRA(L)									
	H'B6	OCRB(H)									
H'FE	H'B7	OCRB(L)									
	H'B8	ICR(H)									FRT 3
	H'B9	ICR(L)									
	H'BA	—	—	—	—	—	—	—	—	—	
	H'BB	—	—	—	—	—	—	—	—	—	
	H'BC	—	—	—	—	—	—	—	—	—	
	H'BD	—	—	—	—	—	—	—	—	—	
	H'BE	—	—	—	—	—	—	—	—	—	
	H'BF	—	—	—	—	—	—	—	—	—	

Notes:

(Continued on next page)

FRT2: Free-Running Timer channel 2

FRT3: Free-Running Timer channel 3

(Continued from preceding page)

Addr. (upper byte)	Addr. (lower byte)	Register Name	Bit Names								
			Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Module**
	H'C0	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	
	H'C1	DTR									PWM1
	H'C2	TCNT									
	H'C3	—	—	—	—	—	—	—	—	—	
	H'C4	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	
	H'C5	DTR									PWM2
	H'C6	TCNT									
H'FE	H'C7	—	—	—	—	—	—	—	—	—	
	H'C8	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	
	H'C9	DTR									PWM3
	H'CA	TCNT									
	H'CB	—	—	—	—	—	—	—	—	—	
	H'CC	—	—	—	—	—	—	—	—	—	
	H'CD	—	—	—	—	—	—	—	—	—	—
	H'CE	—	—	—	—	—	—	—	—	—	
	H'CF	—	—	—	—	—	—	—	—	—	
	H'D0	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	
	H'D1	TCSR	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	
	H'D2	TCORA									
	H'D3	TCORB									TMR
	H'D4	TCNT									
	H'D5	—	—	—	—	—	—	—	—	—	
	H'D6	—	—	—	—	—	—	—	—	—	
H'FE	H'D7	—	—	—	—	—	—	—	—	—	
	H'D8	SMR	C/A	CHR	PE	O/E	STOP	—	CKS1	CKS0	
	H'D9	BRR									
	H'DA	SCR	TIE	RIE	TE	RE	—	—	CKE1	CKE0	
	H'DB	TDR									SCI1
	H'DC	SSR	TDRE	RDRF	ORER	FER	PER	—	—	—	
	H'DD	RDR									
	H'DE	—	—	—	—	—	—	—	—	—	
	H'DF	—	—	—	—	—	—	—	—	—	

Notes:

(Continued on next page)

PWM1: Pulse-Width Modulation timer channel 1
PWM2: Pulse-Width Modulation timer channel 2
PWM3: Pulse-Width Modulation timer channel 3
TMR: 8-Bit Timer
SCI1: Serial Communication Interface 1

(Continued from preceding page)

Addr. (upper byte)	Addr. (lower byte)	Register Name	Bit Names								
			Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'FE	H'E0	ADDRA(H)	AD_9	AD_8	AD_7	AD_6	AD_5	AD_4	AD_3	AD_2	A/D
	H'E1	ADDRA(L)	AD_1	AD_0	—	—	—	—	—	—	
	H'E2	ADDRB(H)	AD_9	AD_8	AD_7	AD_6	AD_5	AD_4	AD_3	AD_2	
	H'E3	ADDRB(L)	AD_1	AD_0	—	—	—	—	—	—	
	H'E4	ADDRC(H)	AD_9	AD_8	AD_7	AD_6	AD_5	AD_4	AD_3	AD_2	
	H'E5	ADDRC(L)	AD_1	AD_0	—	—	—	—	—	—	
	H'E6	ADDRD(H)	AD_9	AD_8	AD_7	AD_6	AD_5	AD_4	AD_3	AD_2	
	H'E7	ADDRD(L)	AD_1	AD_0	—	—	—	—	—	—	
	H'E8	ADCSR	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0	
	H'E9	—	—	—	—	—	—	—	—	—	
	H'EA	—	—	—	—	—	—	—	—	—	
	H'EB	—	—	—	—	—	—	—	—	—	
	H'EC	TCSR*	OVF	WT/IT	TME	—	—	CKS2	CKS1	CKS0	WDT
	H'ED	TCNT*	—	—	—	—	—	—	—	—	
	H'EE	—	—	—	—	—	—	—	—	—	—
	H'EF	—	—	—	—	—	—	—	—	—	
H'FE	H'F0	SMR	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	CKS2	CKS0	SCI2
	H'F1	BRR									
	H'F2	SCR	TIE	RIE	TE	RE	—	—	OKE1	OKE2	
	H'F3	TDR									
	H'F4	SSR	TDRE	RDRF	ORER	FER	—	—	—	—	
	H'F5	RDR									
	H'F6	—	—	—	—	—	—	—	—	—	
	H'F7	—	—	—	—	—	—	—	—	—	
	H'F8	—	—	—	—	—	—	—	—	—	—
	H'F9	—	—	—	—	—	—	—	—	—	
	H'FA	—	—	—	—	—	—	—	—	—	
	H'FB	—	—	—	—	—	—	—	—	—	
	H'FC	SYSCR1	—	IRQ_1E	IRQ_0E	NMIEG	BRLE	—	—	—	Port 1
	H'FD	SYSCR2	—	IRQ_5E	IRQ_4E	IRQ_3E	IRQ_2E	P6PWME	P9PWME	P9SCI2E	Port 6,9
	H'FE	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'FF	P9DR	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	$P9_0$	

Notes:

(Continued on next page)

A/D: Analog-to-Digital converter

WDT: Watchdog Timer

SCI2: Serial Communication Interface 2

* Read addresses are shown. Write addresses of both TCSR and TCNT are H'FEED. See section 13.2.4, “Notes on Register Access” for details.

(Continued from preceding page)

Addr. (upper byte)	Addr. (lower byte)	Register Name	Bit Names								
			Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'FF	H'00	IPRA	—		IRQ_0		—		IRQ_1		INTC
	H'01	IPRB	—		IRQ_2/IRQ_3		—		IRQ_4/IRQ_5		
	H'02	IPRC	—		FRT1		—		FRT2		
	H'03	IPRD	—		FRT3		—		8 Bit Timer		
	H'04	IPRE	—		SCI1		—		SCI2		
	H'05	IPRF	—		A/D		—	—	—	—	
	H'06	—	—	—	—	—	—	—	—	—	
	H'07	—	—	—	—	—	—	—	—	—	
	H'08	DTEA	—	—	—	IRQ_0	—	—	—	IRQ_1	
	H'09	DTEB	—	—	IRQ_3	IRQ_2	—	—	IRQ_5	IRQ_4	
	H'0A	DTEC	—	OCIB1	OCIA1	ICI1	—	OCIB2	OCIA2	ICI2	
	H'0B	DTED	—	OCIB3	OCIA3	ICI3	—	—	CMIB	CMIA	
	H'0C	DTEE	—	TXI1	RXI1	—	—	TXI2	RXI2	—	
	H'0D	DTEF	—	—	—	ADI	—	—	—	—	
	H'0E	—	—	—	—	—	—	—	—	—	
	H'0F	—	—	—	—	—	—	—	—	—	
H'FF	H'10	WCR	—	—	—	—	WMS1	WMS0	WC1	WC0	WSC
	H'11	RAMCR	RAME	—	—	—	—	—	—	—	RAM
	H'12	MDCR	—	—	—	—	—	MDS2	MDS1	MDS0	—
	H'13	SBYCR	SSBY	—	—	—	—	—	—	—	
	H'14	WCR									WDT
	H'15	RSTCSR	WRST	RSTOE	—	—	—	—	—	—	
H'FF	H'16	—	—	—	—	—	—	—	—	—	—
	H'17	—	—	—	—	—	—	—	—	—	
	H'18	—	—	—	—	—	—	—	—	—	
	H'19	—	—	—	—	—	—	—	—	—	
	H'1A	—	—	—	—	—	—	—	—	—	
	H'1B	—	—	—	—	—	—	—	—	—	
	H'1C	—	—	—	—	—	—	—	—	—	
	H'1D	—	—	—	—	—	—	—	—	—	
	H'1E	—	—	—	—	—	—	—	—	—	
	H'1F	—	—	—	—	—	—	—	—	—	

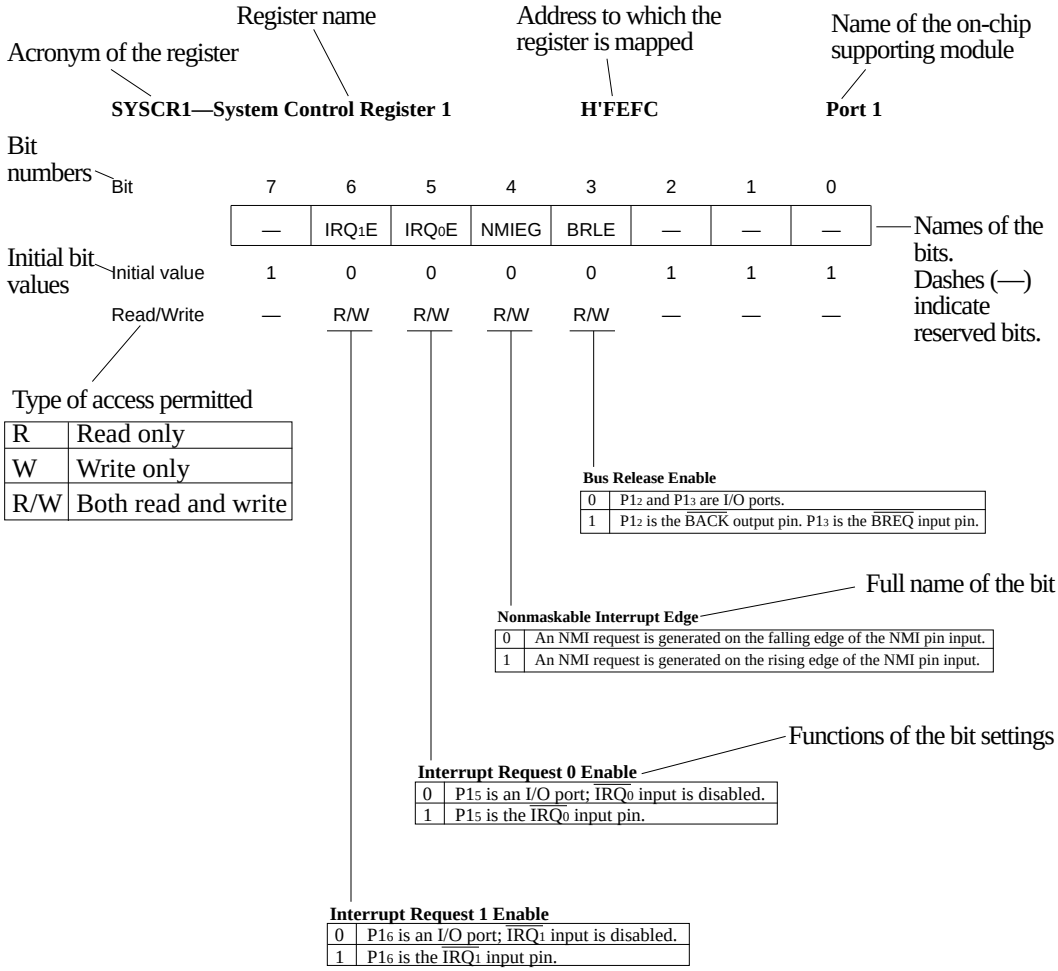
Notes:

INTC: Interrupt Controller

WSC: Wait State Controller

WDT: Watchdog Timer

B.2 Register Descriptions



P1DDR—Port 1 Data Direction Register **H'FE80** **Port 1**

Bit	7	6	5	4	3	2	1	0
	$P1_7DDR$	$P1_6DDR$	$P1_5DDR$	$P1_4DDR$	$P1_3DDR$	$P1_2DDR$	$P1_1DDR$	$P1_0DDR$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 1 Input/Output Selection

0	Input port
1	Output port

P2DDR—Port 2 Data Direction Register **H'FE81** **Port 2**

Bit	7	6	5	4	3	2	1	0
	—	—	—	$P2_4DDR$	$P2_3DDR$	$P2_2DDR$	$P2_1DDR$	$P2_0DDR$
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	W	W	W	W	W

Port 2 Input/Output Selection

0	Input port
1	Output port

P1DR—Port 1 Data Register **H'FE82** **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	—	—
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R

P2DR—Port 2 Data Register **H'FE83** **Port 2**

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

P3DDR—Port 3 Data Direction Register **H'FE84** **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 Selection

0	Input port
1	Output port

P4DDR—Port 4 Data Direction Register **H'FE85** **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 Selection

0	Input port
1	Output port

P3DR—Port 3 Data Register **H'FE86** **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

P4DR—Port 4 Data Register **H'FE87** **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'FE88** **Port 5**

Bit	7	6	5	4	3	2	1	0
	$P5_7$DDR	$P5_6$DDR	$P5_5$DDR	$P5_4$DDR	$P5_3$DDR	$P5_2$DDR	$P5_1$DDR	$P5_0$DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 5 Input/Output Selection

0	Input port
1	Output port

P6DDR—Port 6 Data Direction Register **H'FE89** **Port 6**

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

Port 6 Input/Output Selection

0	Input port
1	Output port

P5DR—Port 5 Data Register **H'FE8A** **Port 5**

Bit	7	6	5	4	3	2	1	0
	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
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—Port 6 Data Register **H'FE8B** **Port 6**

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

P7DDR—Port 7 Data Direction Register **H'FE8C** **Port 7**

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

Port 7 Input/Output Selection

0	Input port
1	Output port

P7DR—Port 7 Data Register **H'FE8E** **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	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

P8DR—Port 8 Data Register **H'FE8F** **Port 8**

Bit	7	6	5	4	3	2	1	0
	$P8_7$	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$
Read/Write	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	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

00	Internal clock source: ø4
01	Internal clock source: ø8
10	Internal clock source: ø32
11	External clock source: counted on rising edge

Output Enable A

0	Compare-A output is disabled.
1	Compare-A output is enabled.

Output Enable B

0	Compare-B output is disabled.
1	Compare-B output is enabled.

Timer Overflow Interrupt Enable

0	Overflow interrupt request is disabled.
1	Overflow interrupt request is enabled.

Output Compare Interrupt Enable A

0	Compare-match A interrupt request is disabled.
1	Compare-match A interrupt request is enabled.

Output Compare Interrupt Enable B

0	Compare-match B interrupt request is disabled.
1	Compare-match B interrupt request is enabled.

Input Capture Interrupt Enable

0	Input capture interrupt is disabled.
1	Input capture interrupt is enabled.

Bit	7	6	5	4	3	2	1	0
	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	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.

Input Edge Select

0	Count is captured on falling edge of input capture signal (FTI).
1	Count is captured on rising edge of input capture signal.

Output Level A

0	Compare-match A causes 0 output.
1	Compare-match A causes 1 output.

Output Level B

0	Compare-match B causes 0 output.
1	Compare-match B causes 1 output.

Timer Overflow

0	Cleared from 1 to 0 when CPU reads OVF = 1, then writes 0 in OVF.
1	Set to 1 when FRC changes from H'FFFF to H'0000.

Output Compare Flag A

0	Cleared from 1 to 0 when: 1. CPU reads OCFA = 1, then writes 0 in OCFA. 2. OCIA interrupt is served by DTC.
1	Set to 1 when FRC = OCRA.

Output Compare Flag B

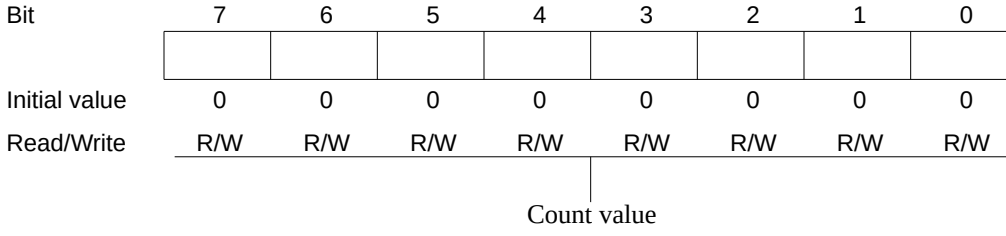
0	Cleared from 1 to 0 when: 1. CPU reads OCFB = 1, then writes 0 in OCFB. 2. OCIB interrupt is served by DTC.
1	Set to 1 when FRC = OCRB.

Input Capture Flag

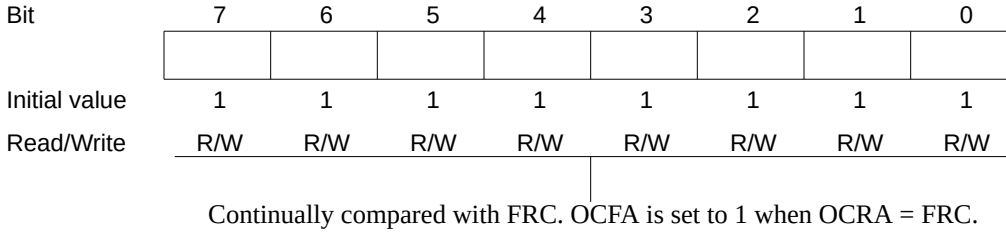
0	Cleared from 1 to 0 when: 1. CPU reads ICF = 1, then writes 0 in ICF. 2. ICI interrupt is served by DTC.
1	Set to 1 when input capture signal is received and FRC count is copied to ICR.

* Only writing of a 0 to clear the flag is enabled.

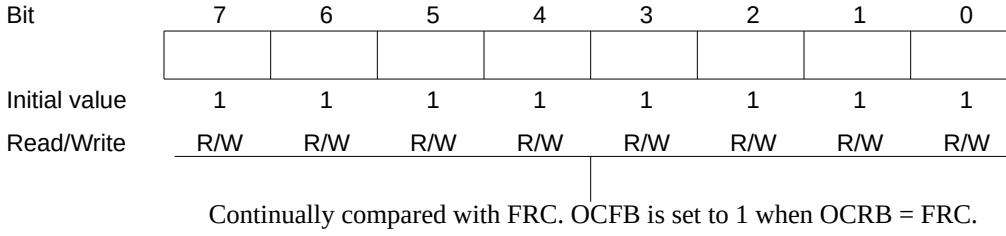
FRC (H and L)—Free-Running Counter H'FE92, H'FE93 FRT 1



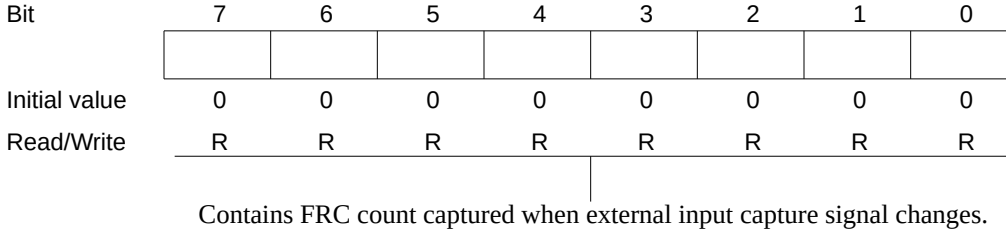
OCRA (H and L)—Output Compare Register A H'FE94, H'FE95 FRT 1



OCRB (H and L)—Output Compare Register B H'FE96, H'FE97 FRT 1



ICR (H and L)—Input Capture Register H'FE98, H'FE99 FRT 1



TCR—Timer Control Register **H'FEA0** **FRT 2**

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	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 FRT1.

TCSR—Timer Control/Status Register **H'FEA1** **FRT 2**

Bit	7	6	5	4	3	2	1	0
	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	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

Note: Bit functions are the same as for FRT1.

* Only writing of a 0 to clear the flag is enabled.

FRC (H and L)—Free-Running Counter **H'FEA2, H'FEA3** **FRT 2**

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 FRT1.

OCRA (H and L)—Output Compare Register A H'FEA4, H'FEA5 FRT 2

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 FRT1.

OCRB (H and L)—Output Compare Register B H'FEA6, H'FEA7 FRT 2

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 FRT1.

ICR (H and L)—Input Capture Register H'FEA8, H'FEA9 FRT 2

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

Note: Bit functions are the same as for FRT1.

TCR—Timer Control Register H'FEB0 FRT 3

Bit	7	6	5	4	3	2	1	0
	ICIE	OCIEB	OCIEA	OVIE	OEB	OEA	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 FRT1.

TCSR—Timer Control/Status Register H'FEB1 FRT 3

Bit	7	6	5	4	3	2	1	0
	ICF	OCFB	OCFA	OVF	OLVLB	OLVLA	IEDG	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

Note: Bit functions are the same as for FRT1.

* Only writing of 0 to clear the flag is enabled.

FRC (H and L)—Free-Running Counter H'FEB2, H'FEB3 FRT 3

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 FRT1.

OCRA (H and L)—Output Compare Register A H'FEB4, H'FEB5 FRT 3

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 FRT1.

OCRB (H and L)—Output Compare Register B H'FEB6, H'FEB7 FRT 3

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 FRT1.

ICR (H and L)—Input Capture Register H'FEB8, H'FEB9 FRT 3

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

Note: Bit functions are the same as for FRT1.

TCR—Timer Control Register **H'FEC0** **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

Clock Select (Values When ø = 10MHz)

	Internal Clock Freq.	Reso-lution	PWM Period	PWM Frequency
000	ø/2	200 ns	50 µs	20 kHz
001	ø/8	800 ns	200 µs	5 kHz
010	ø/32	3.2 µs	800 µs	1.25 kHz
011	ø/128	12.8 µs	3.2 ms	312.5 kHz
100	ø/256	25.6 µs	6.4 ms	156.3 Hz
101	ø/1024	102.4 µs	25.6 ms	39.1 Hz
110	ø/2048	204.8 µs	51.2 ms	19.5 Hz
111	ø/4096	409.6 µs	102.4 ms	9.8 Hz

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'FEC1** **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

Pulse duty factor

TCNT—Timer Counter **H'FEC2** **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)*

Count value (runs from H'00 to H'F9, then repeats from H'00)

* Write function is for test purposes only. Writing to this register during normal operation may have unpredictable effects

---

TCR—Timer Control Register **H'FEC4** **PWM2**

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 PWM1.

DTR—Duty Register **H'FEC5** **PWM2**

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 PWM1.

TCNT—Timer Counter **H'FEC6** **PWM2**

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 PWM1.

* Write function is for test purposes only. Writing to this register during normal operation may have unpredictable effects

---

TCR—Timer Control Register **H'FEC8** **PWM3**

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 PWM1.

DTR—Duty Register **H'FEC9** **PWM3**

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 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 PWM1.

* Write function is for test purposes only. Writing to this register during normal operation may have unpredictable effects.

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.

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 from 1 to 0 when CPU reads OVF = 1, then writes 0 in OVF.
1	Set to 1 when TCNT changes from H'FF to H'00.

Compare-Match Flag A

0	Cleared from 1 to 0 when: 1. CPU reads CMFA = 1, then writes 0 in CMFA. 2. CMA interrupt is served by the DTC.
1	Set to 1 when TCNT = TCORA.

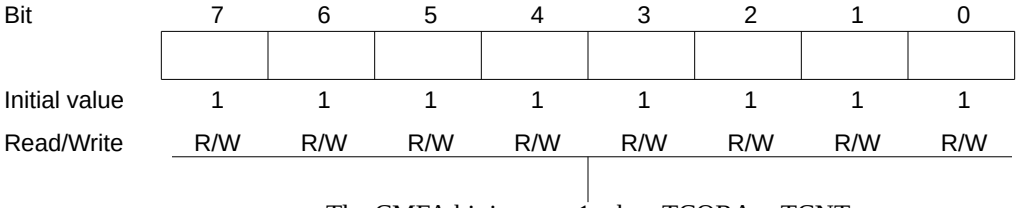
Compare-Match Flag B

0	Cleared from 1 to 0 when: 1. CPU reads CMFB = 1, then writes 0 in CMFB. 2. CMB interrupt is served by the DTC.
1	Set to 1 when TCNT = TCORB.

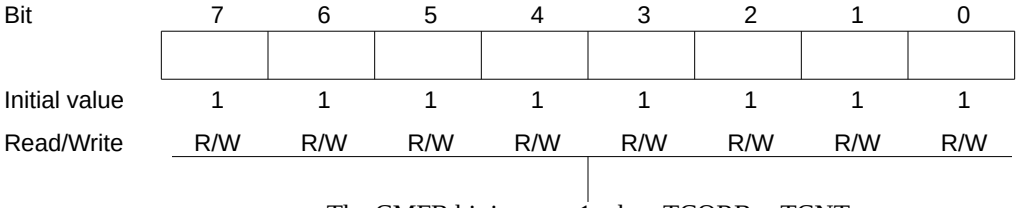
*1 Only writing of 0 to clear the flag is enabled.

*2 When all four bits (OS3 to OS0) are cleared to 0, output is disabled.

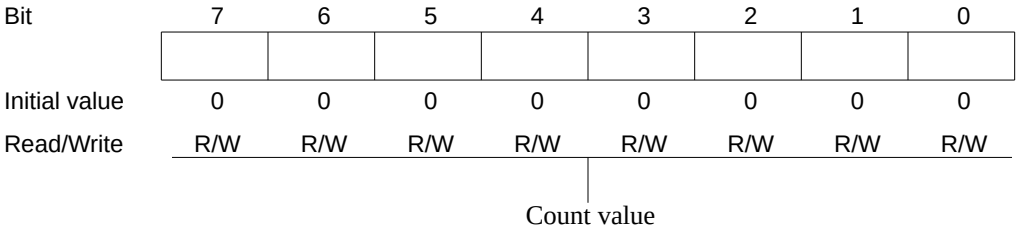
TCORA—Time Constant Register A **H'FED2** **TMR**



TCORB—Time Constant Register B **H'FED3** **TMR**



TCNT—Timer Counter **H'FED4** **TMR**



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'FED9** **SCI1**

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 baud rate

SCR—Serial Control Register **H'FEDA** **SCI1**

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	SCK pin is NOT USED.
1	SCK pin is used for output.

Clock Enable 1

0	Internal clock
1	External clock, input at SCK pin

Receive Enable

0	Receive disabled
1	Receive enabled

Transmit Enable

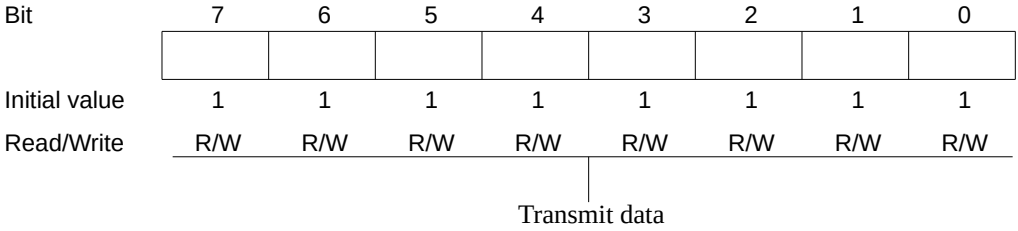
0	Transmit disabled
1	Transmit enabled

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.


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
Transmit data

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 from 1 to 0 when: 1. CPU reads PER = 1, then writes 0 in PER. 2. The chip is reset or enters a standby mode.
1	Set to 1 when a parity error occurs (parity of receive data does not match parity selected by bit).

Framing Error

0	Cleared from 1 to 0 when: 1. CPU reads FER = 1, then writes 0 in FER. 2. The chip is reset or enters a standby mode.
1	Set to 1 when a framing error occurs (stop bit is 0).

Overrun Error

0	Cleared from 1 to 0 when: 1. CPU reads ORER = 1, then writes 0 in ORER. 2. The chip is reset or enters a standby mode.
1	Set to 1 when an overrun error occurs (next data is completely received while RDRF bit is set to 1).

Receive Data Register Full

0	Cleared from 1 to 0 when: 1. CPU reads RDRF = 1, then writes 0 in RDRF. 2. RDR is read by the DTC. 3. The chip is reset or enters a standby mode.
1	Set to 1 when one character is received normally and transferred from RSR to RDR.

Transmit Data Register Empty

0	Cleared from 1 to 0 when: 1. CPU reads TDRE = 1, then writes 0 in TDRE. 2. The DTC writes data in TDR.
1	Set to 1 when: 1. The chip is reset or enters a standby mode. 2. Data is transferred from TDR to TSR. 3. CPU reads TDRE = 0, then clears 0 in TE.

* Only writing of 0 to clear the flag is enabled.

RDR—Receive Data Register **H'FEDD** **SCI1**

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 (H)—A/D Data Register n (High)
H'FEE0, H'FEE2, H'FEE4, H'FEE6 **(n = A, B, C, D)** **A/D**

Bit	7	6	5	4	3	2	1	0
	AD_9	AD_8	AD_7	AD_6	AD_5	AD_4	AD_3	AD_2
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Upper 8 bits of 10-bit A/D conversion result

ADDRn (L)—A/D Data Register n (Low)
H'FEE1, H'FEE3, H'FEE5, H'FEE7 **(n = A, B, C, D)** **A/D**

Bit	7	6	5	4	3	2	1	0
	AD_1	AD_0	—	—	—	—	—	—
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Lower 2 bits of 10-bit A/D conversion result

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 = 274 states
1	Conversion time = 138 states

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: 1. The chip is reset or enters a standby mode. 2. CPU reads ADF = 1, then writes 0 in ADF. 3. DTC is served by ADI.
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.

* Only writing of 0 to clear the flag is enabled.

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	The A/D external trigger is disabled.
1	The A/D external trigger is enabled. A/D conversion starts on the falling edge of ADTRG.

Bit	7	6	5	4	3	2	1	0
	OVF	WT/$\overline{IT}$	TME	—	—	CKS2	CKS1	CKS0
Initial value	0	0	0	1	1	0	0	0
Read/Write	R/(W)*3	R/W	R/W	—	—	R/W	R/W	R/W

Clock Select

0	0	0	ø/2	(51.2 µs)*4
0	0	1	ø/32	(819.2 µs)
0	1	0	ø/64	(1.6 ms)
0	1	1	ø/128	(3.3 ms)
1	0	0	ø/256	(6.6 ms)
1	0	1	ø/512	(13.1 ms)
1	1	0	ø/2048	(52.4 ms)
1	1	1	ø/4096	(104.9 ms)

Timer Enable

0	Timer is disabled. • TCNT is initialized to H'00 and stopped.
1	Timer is enabled. • TCNT starts incrementing. • CPU interrupt request is enabled.

Timer Mode Select

0	Interval timer mode (IRQ0 interrupt request)
1	Watchdog timer mode (NMI interrupt request)

Overflow Flag

0	Cleared from 1 to 0 when CPU reads OVF = 1, then wtites 0 in OVF.
1	Set to 1 when TCNT changes from H'FF to H'00.

*1 Read address
*2 Write address
*3 Only writing of 0 to clear the flag is enabled.
*4 Times in parentheses are the times for TCNT to increment from H'00 to H'FF and change to H'00 again when ø = 10 MHz.

TCNT—Timer Counter **H'FEED** **WDT**

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

SMR—Serial Mode Register **H'FEF0** **SCI2**

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

Note: Bit functions are the same as for SCI1.

BRR—Bit Rate Register **H'FEF1** **SCI2**

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 SCI1.

SCR—Serial Control Register **H'FEF2** **SCI2**

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

Note: Bit functions are the same as for SCI1.

TDR—Transmit Data Register **H'FEF3** **SCI2**

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 SCI1.

SSR—Serial Status Register **H'FEF4** **SCI2**

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER			
Initial value	0	0	0	0	0	1	1	1
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	—	—	—

Note: Bit functions are the same as for SCI1.
* Only writing of 0 to clear the flag is enabled.

RDR—Receive Data Register **H'FEF5** **SCI2**

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 SCI1.

Bit	7	6	5	4	3	2	1	0
	—	IRQ1E	IRQ0E	NMIEG	BRLE	—	—	—
Initial value	1	0	0	0	0	1	1	1
Read/Write	—	R/W	R/W	R/W	R/W	—	—	—

Bus Release Enable

0	P12 and P13 are I/O ports.
1	P12 is the $\overline{\text{BACK}}$ output pin and P13 is the $\overline{\text{BREQ}}$ input pin.

Nonmaskable Interrupt Edge

0	An NMI request is generated on the falling edge of the NMI pin input.
1	An NMI request is generated on the rising edge of the NMI pin input.

Interrupt Request 0 Enable

0	P15 is an I/O port; $\overline{\text{IRQ}}_0$ input is disabled.
1	P15 is the $\overline{\text{IRQ}}_0$ input pin.

Interrupt Request 1 Enable

0	P16 is an I/O port; $\overline{\text{IRQ}}_1$ input is disabled.
1	P16 is the $\overline{\text{IRQ}}_1$ input pin.

Bit	7	6	5	4	3	2	1	0
	—	IRQ5E	IRQ5E	IRQ5E	IRQ5E	P6PWME	P9PWME	P9SCI2E
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

Port 9 SCI2 Enable

0	P92, P93, and P94 cannot be used for serial communication.
1	P92, P93, and P94 can be used for serial communication (see port 9 pin functions).

Port 9 PWM Enable

0	P92, P93, and P94 cannot be used for PWM output.
1	P92, P93, and P94 can be used for PWM output (see port 9 pin functions).

Port 6 PWM Enable

0	P61, P62, and P63 cannot be used for PWM output.
1	P61, P62, and P63 can be used for PWM output (see port 6 pin functions).

Interrupt Request 2 Enable

0	P60 is not used for $\overline{\text{IRQ2}}$ signal input.
1	P60 is used for $\overline{\text{IRQ2}}$ signal input.

Interrupt Request 3 Enable

0	P61 is not used for $\overline{\text{IRQ3}}$ signal input.
1	P61 is used for $\overline{\text{IRQ3}}$ signal input.

Interrupt Request 4 Enable

0	P62 is not used for $\overline{\text{IRQ4}}$ signal input.
1	P62 is used for $\overline{\text{IRQ4}}$ signal input.

Interrupt Request 5 Enable

0	P63 is not used for $\overline{\text{IRQ5}}$ signal input.
1	P63 is used for $\overline{\text{IRQ5}}$ signal input.

P9DDR—Port 9 Data Direction Register **H'FEFE** **Port 9**

Bit	7	6	5	4	3	2	1	0
	$P9_7DDR$	$P9_6DDR$	$P9_5DDR$	$P9_4DDR$	$P9_3DDR$	$P9_2DDR$	$P9_1DDR$	$P9_0DDR$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 9 Input/Output Selection

0	Input port
1	Output port

P9DR—Port 9 Data Register **H'FEFF** **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

IPRA—Interrupt Priority Register A **H'FF00** **INTC**

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

$\overline{IRQ_0}$ interrupt priority level (0 to 7)

$\overline{IRQ_1}$ interrupt priority level (0 to 7)

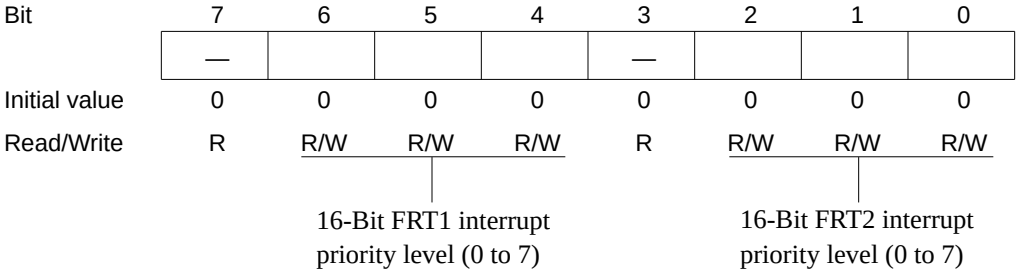
IPRB—Interrupt Priority Register B **H'FF01** **INTC**

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

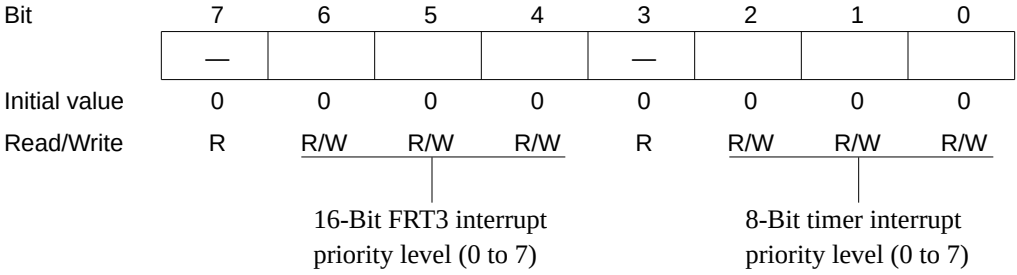
$\overline{IRQ_2}$ and $\overline{IRQ_3}$ interrupt priority level (0 to 7)

$\overline{IRQ_4}$ and $\overline{IRQ_5}$ interrupt priority level (0 to 7)

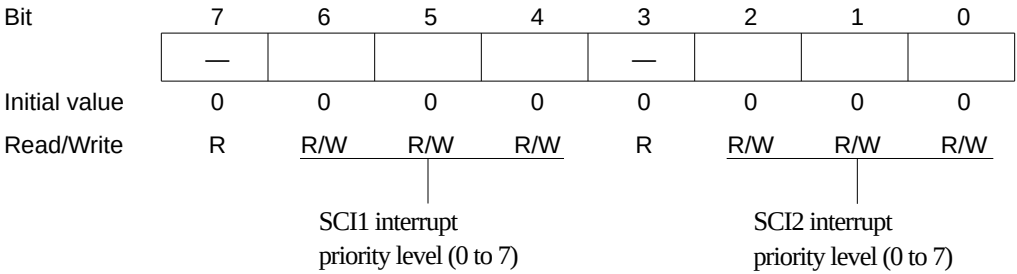
IPRC—Interrupt Priority Register C **H'FF02** **INTC**



IPRD—Interrupt Priority Register D **H'FF03** **INTC**



IPRE—Interrupt Priority Register E **H'FF04** **INTC**



IPRF—Interrupt Priority Register F **H'FF05** **INTC**

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

A/D interrupt priority level (0 to 7)

Unused

DTEA—Data Transfer Enable Register A **H'FF08** **INTC**

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

$\overline{IRQ_0}$

0	Served by CPU
1	Served by DTC

$\overline{IRQ_1}$

0	Served by CPU
1	Served by DTC

DTEB—Data Transfer Enable Register B **H'FF09** **INTC**

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

$\overline{IRQ_2}$

0	Served by CPU
1	Served by DTC

$\overline{IRQ_4}$

0	Served by CPU
1	Served by DTC

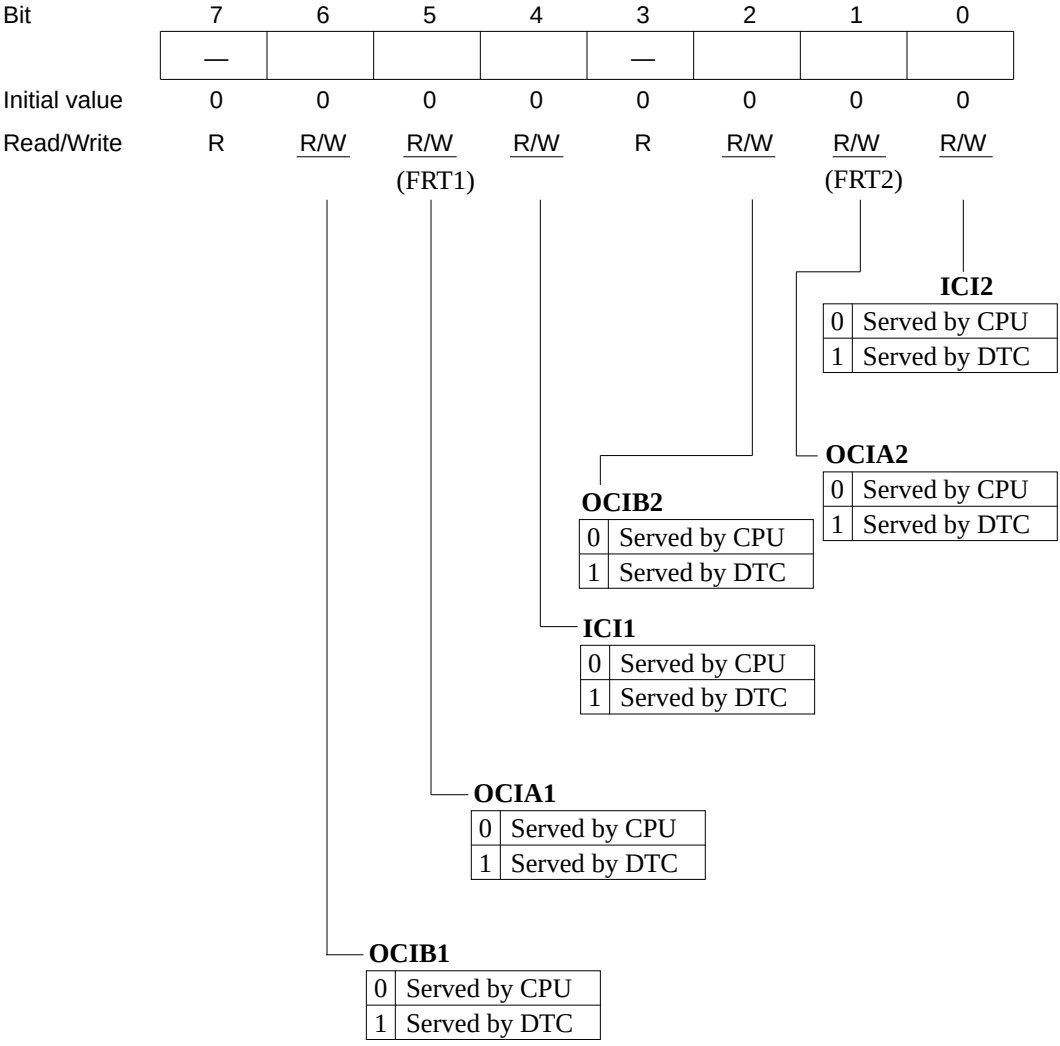
$\overline{IRQ_3}$

0	Served by CPU
1	Served by DTC

$\overline{IRQ_5}$

0	Served by CPU
1	Served by DTC

DTEC—Data Transfer Enable Register C **H'FF0A** **INTC**



DTED—Data Transfer Enable Register D **H'FF0B** **INTC**

Bit	7	6	5	4	3	2	1	0
	—				—	—		
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R/W	R/W	R/W	R	R/W	R/W	R/W
			(FRT3)				(8-Bit timer)	

ICI3

0	Served by CPU
1	Served by DTC

CMIA

0	Served by CPU
1	Served by DTC

CMIB

0	Served by CPU
1	Served by DTC

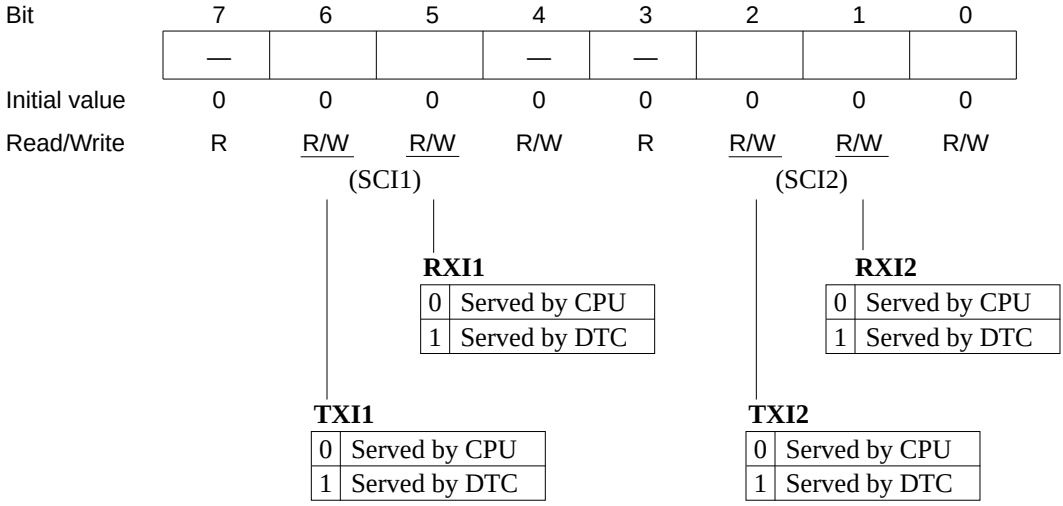
OCIA3

0	Served by CPU
1	Served by DTC

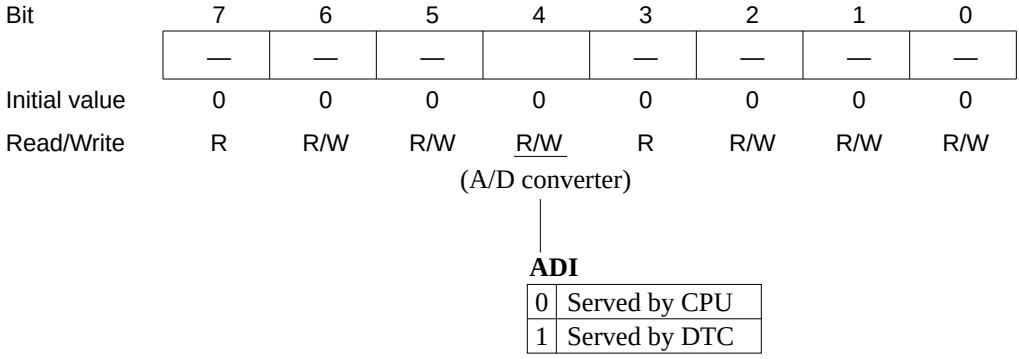
OCIB3

0	Served by CPU
1	Served by DTC

DTEE—Data Transfer Enable Register E **H'FF0C** **INTC**



DTEF—Data Transfer Enable Register F **H'FF0D** **INTC**



WCR—Wait-State Control Register **H'FF10** **WSC**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	WMS1	WMS0	WC1	WC0
Initial value	1	1	1	1	0	0	1	1
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

Wait Count 1 and 0

0	0	No wait states (Tw) are inserted.
0	1	1 Wait states are inserted.
1	0	2 Wait states are inserted.
1	1	3 Wait state is inserted.

Wait Mode Select 1 and 0

0	0	Programmable wait mode
0	1	No wait states are inserted, regardless of the wait count.
1	0	Pin wait mode
1	1	Pin auto-wait mode

RAMCR—RAM Control Register **H'FF11** **RAM**

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

RAM Enable

0	On-chip RAM is disabled.
1	On-chip RAM is enabled.

MDCR—Mode Control Register H'FF12

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	MDS2	MDS1	MDS0
Initial value	1	1	0	0	0	—*	—*	—*
Read/Write	—	—	—	—	—	R	R	R

Mode Select

Value input at mode pins

* Initialized according to the inputs at pins MD_2, MD_1, and MD_0.

SBYCR—Software Standby Control Register H'FF13

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

Software Standby

0	SLEEP instruction causes transition to sleep mode.
1	SLEEP instruction causes transition to software standby mode.

RSTCSR—Reset Status/Control Register H'FF15 WDT

Bit	7	6	5	4	3	2	1	0
	WRST	RSTOE	—	—	—	—	—	—
Initial value	0	0	1	1	1	1	1	1
Read/Write	R/(W)*	R/W	—	—	—	—	—	—

Reset Output Enable

0	The reset signal is not output externally.
1	The reset signal is output externally.

Watchdog Timer Reset

0	Cleared from 1 to 0 by software, or by a Low input at the $\overline{RES}$ pin.
1	Set to 1 when TCNT overflows and a reset signal is generated.

* Software can write a 0 in bit 7 to clear the flag but cannot write a 1.

Appendix C I/O Port Schematic Diagrams

C.1 Schematic Diagram of Port 1

Figure C-1 (a) to (g) gives a schematic view of the port 1 input/output circuits.

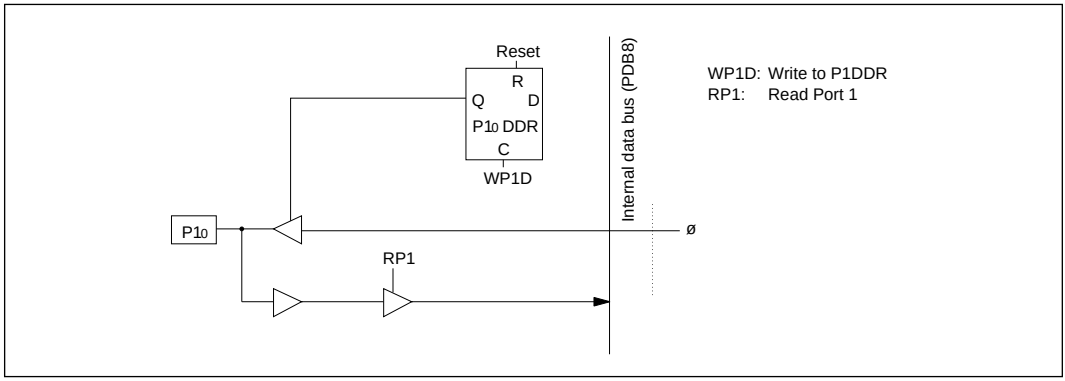


Figure C-1 (a) Schematic Diagram of Port 1, Pin P1$_0$

Table C-1 (a) Port 1 Port Read (Pin P1$_0$)

Setting	Port Read Data
DDR = 0	Pin value
DDR = 1	ø

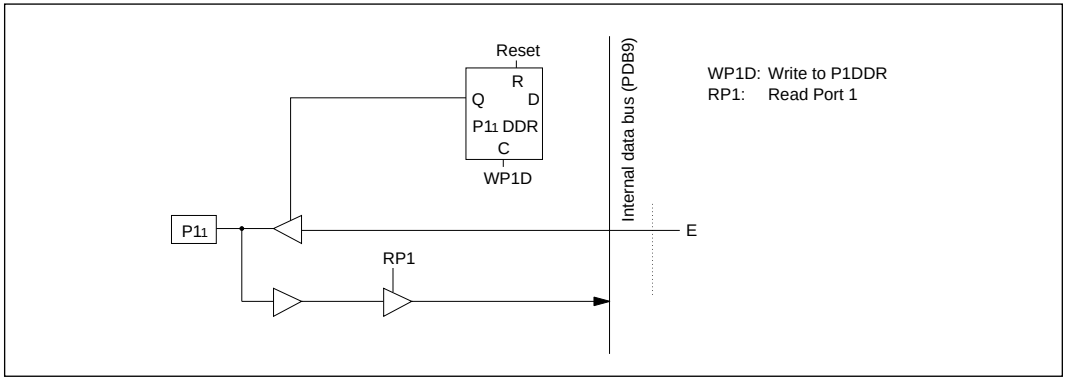


Figure C-1 (b) Schematic Diagram of Port 1, Pin P1$_1$

Table C-1 (b) Port 1 Port Read (Pin P11)

Setting	Port Read Data
DDR = 0	Pin value
DDR = 1	E

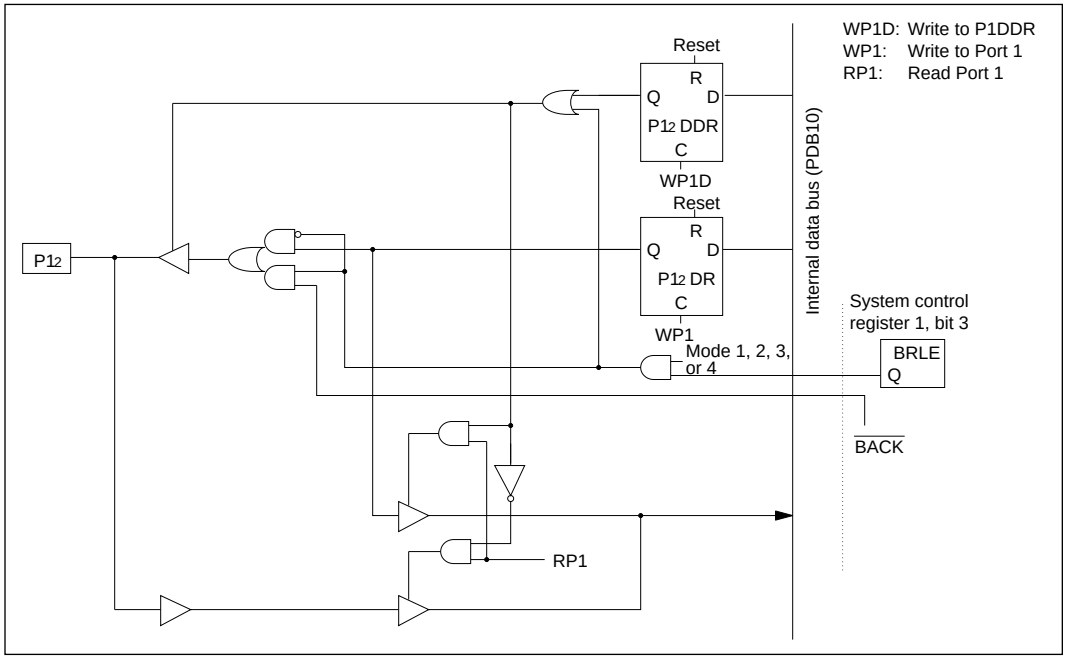


Figure C-1 (c) Schematic Diagram of Port 1, Pin P12

Table C-1 (c) Port 1 Port Read (Pin P12)

Mode	Setting		Port Read Data
1, 2, 3, 4	BRLE = 1		DR value
	BRLE = 0	DDR = 0	Pin value
		DDR = 1	DR value
7	DDR = 0		Pin value
	DDR = 1		DR value

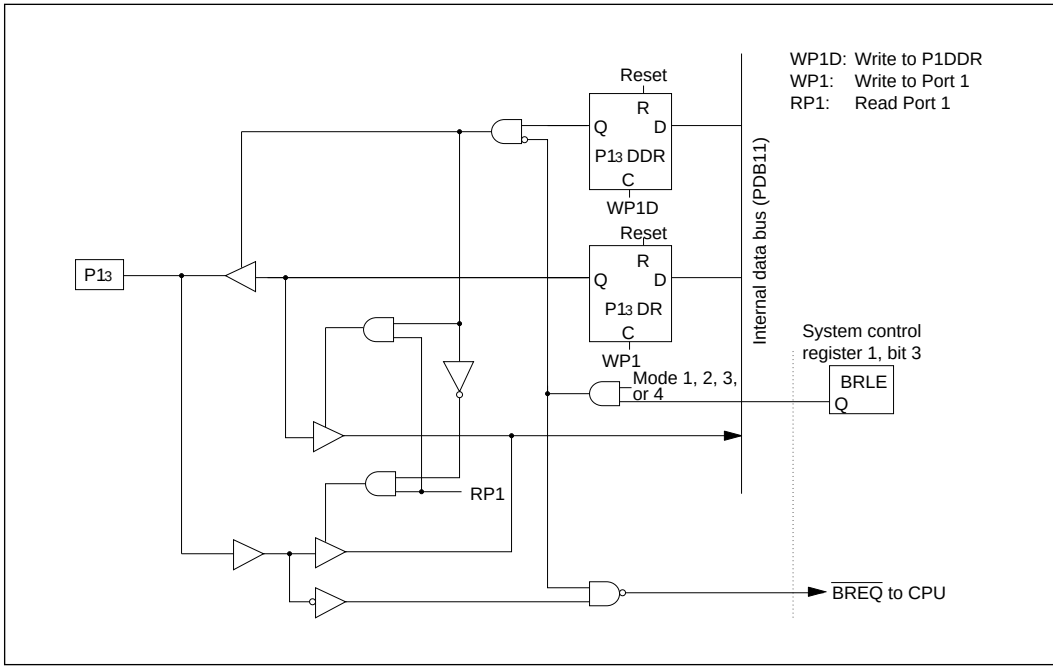


Figure C-1 (d) Schematic Diagram of Port 1, Pin P13

Table C-1 (d) Port 1 Port Read (Pin P13)

Mode	Setting		Port Read Data
1, 2, 3, 4	BRLE = 1		Pin value
	BRLE = 0	DDR = 0	Pin value
		DDR = 1	DR value
7	DDR = 0		Pin value
	DDR = 1		DR value

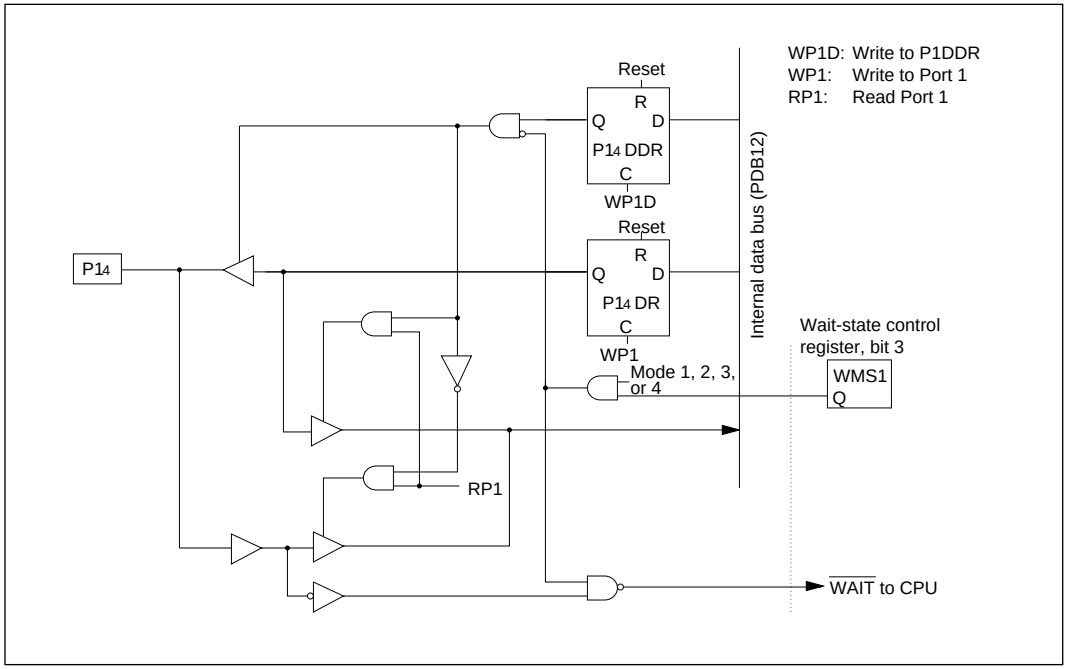


Figure C-1 (e) Schematic Diagram of Port 1, Pin P14

Table C-1 (e) Port 1 Port Read (Pin P14)

Mode	Setting		Port Read Data
1, 2, 3, 4	WMS 1 = 1		Pin value
	WMS 1 = 0	DDR = 0	Pin value
		DDR = 1	DR value
7	DDR = 0		Pin value
	DDR = 1		DR value

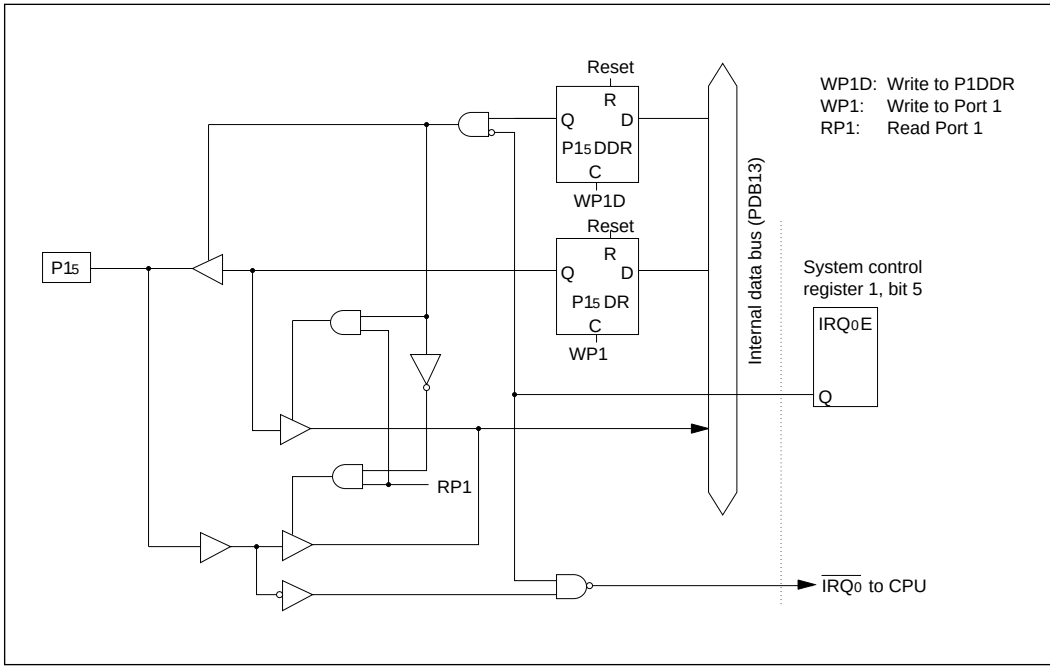


Figure C-1 (f) Schematic Diagram of Port 1, Pin P15

Table C-1 (f) Port 1 Port Read (Pin P15)

Setting		Port Read Data
IRQ0E = 1	Pin value	
IRQ0E = 0	DDR = 0	Pin value
	DDR = 1	DR value

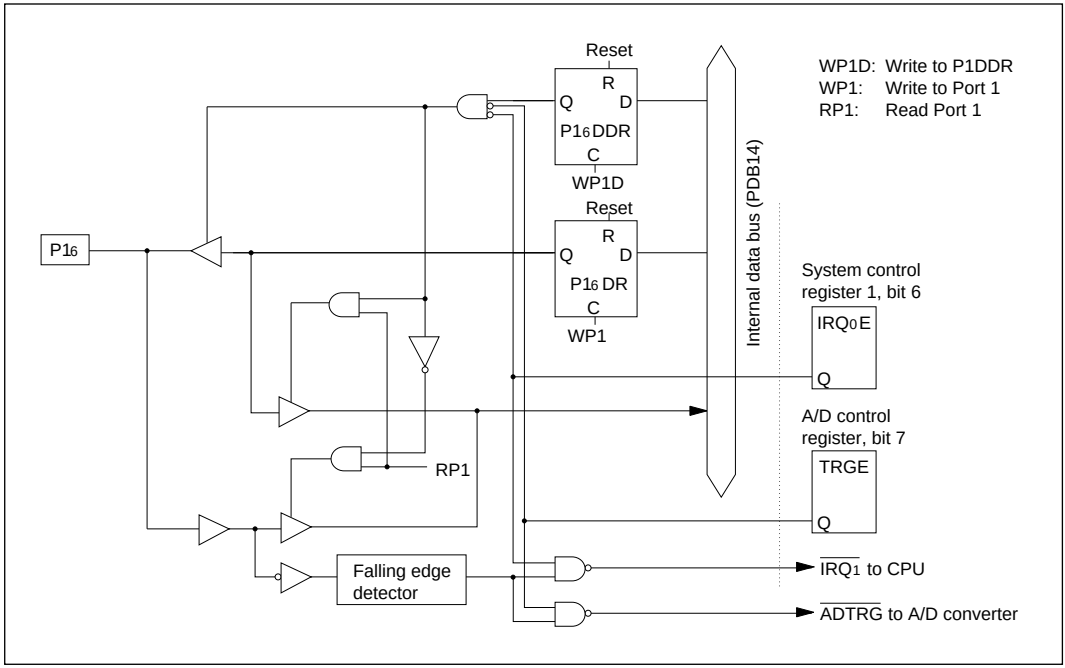


Figure C-1 (g) Schematic Diagram of Port 1, Pin P16

Table C-1 (g) Port 1 Port Read (Pin P16)

Setting		Port Read Data
TRGE or IRQ1E = 1		Pin value
TRGE and IRQ1E = 0	DDR = 0	Pin value
	DDR = 1	DR value

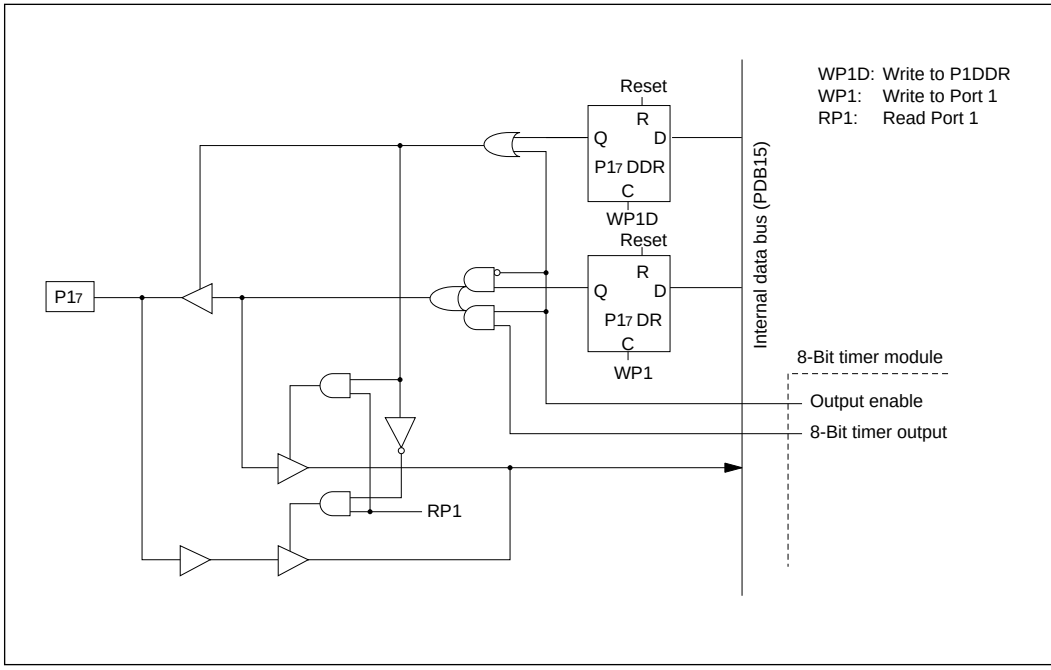


Figure C-1 (h) Schematic Diagram of Port 1, Pin P17

Table C-1 (h) Port 1 Port Read (Pin P17)

Setting		Port Read Data
8-bit timer output enable		8-bit timer output value
8-bit timer	DDR = 0	Pin value
output disable	DDR = 1	DR value

C.2 Schematic Diagram of Port 2

Figure C-2 gives a schematic view of the port 2 input/output circuits.

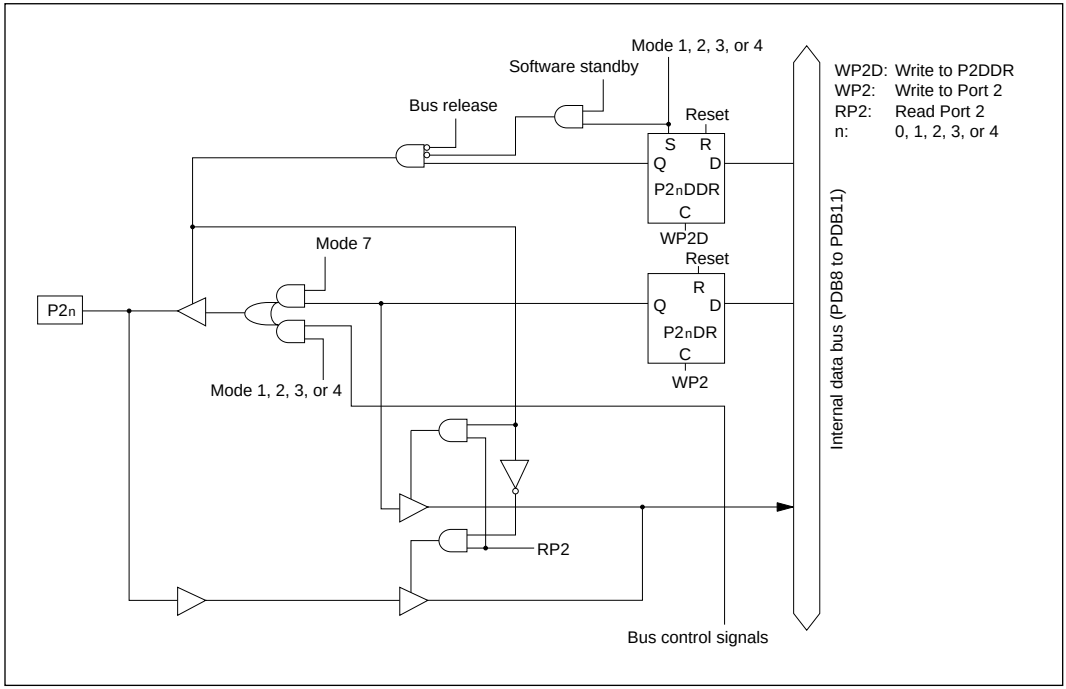


Figure C-2 Schematic Diagram of Port 2

Table C-2 Port 2 Port Read

Mode		Port Read Data
1, 2, 3, 4		DR value
7	DDR = 0	Pin value
	DDR = 1	DR value

C.3 Schematic Diagram of Port 3

Figure C-3 gives a schematic view of the port 3 input/output circuits.

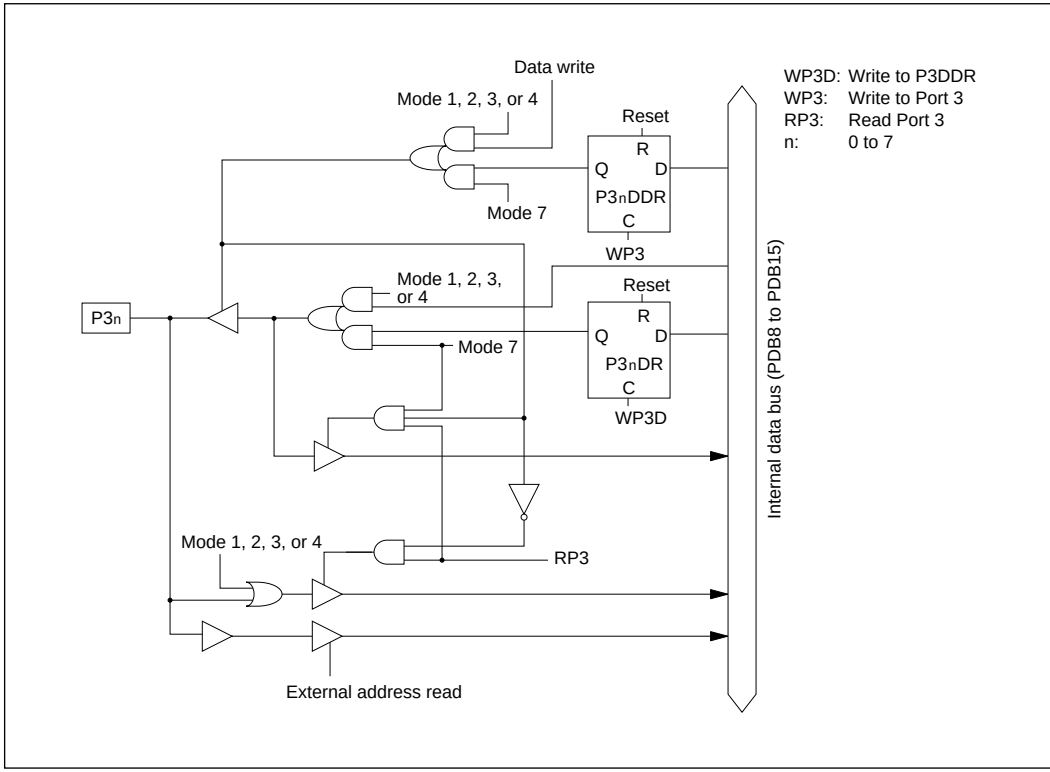


Figure C-3 Schematic Diagram of Port 3

Table C-3 Port 3 Port Read

Mode		Port Read Data
1, 2, 3, 4		Always reads 1
7	DDR = 0	Pin value
	DDR = 1	DR value

C.4 Schematic Diagram of Port 4

Figure C-4 gives a schematic view of the port 4 input/output circuits.

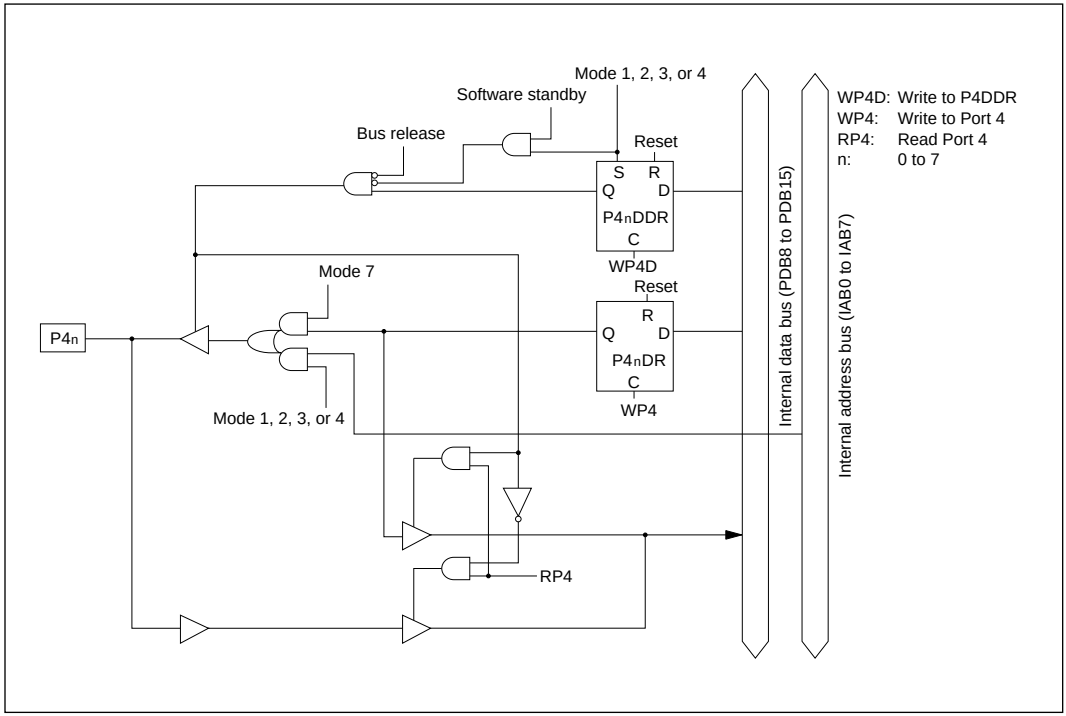


Figure C-4 Schematic Diagram of Port 4

Table C-4 Port 4 Port Read

Mode		Port Read Data
1, 2, 3, 4		DR value
7	DDR = 0	Pin value
	DDR = 1	DR value

C.5 Schematic Diagram of Port 5

Figure C-5 gives a schematic view of the port 5 input/output circuits.

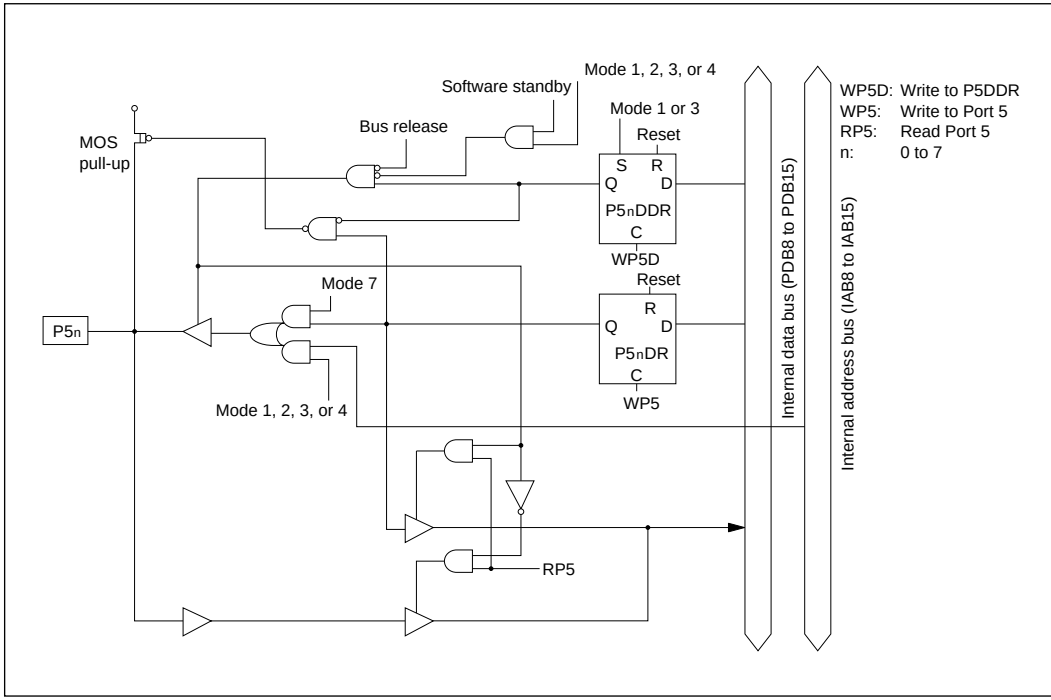


Figure C-5 Schematic Diagram of Port 5

Table C-5 Port 5 Port Read

Mode		Port Read Data
1, 3		DR value
2, 4, 7	DDR = 0	Pin value
	DDR = 1	DR value

C.6 Schematic Diagram of Port 6

Figure C-6 gives a schematic view of the port 6 input/output circuits.

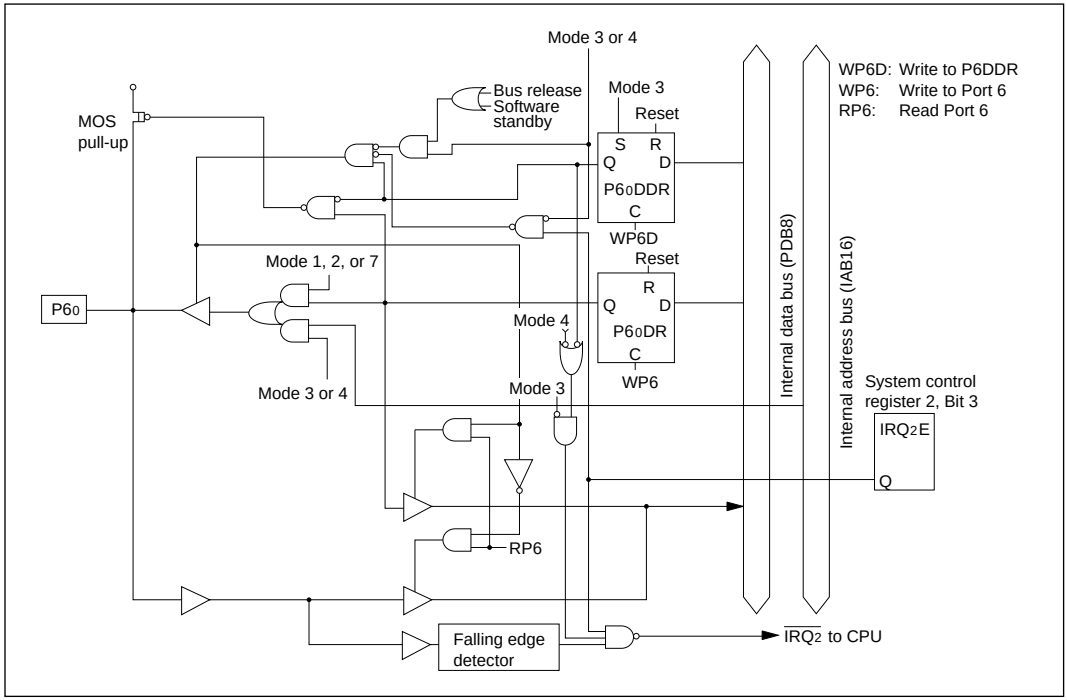


Figure C-6 (a) Schematic Diagram of Port 6, Pin P60

Table C-6 (a) Port 6 Port Read (Pin P60)

Mode			Port Read Data
3			DR value
1, 2, 4, 7	IRQ2E = 0	DDR = 0	Pin value
		DDR = 1	DR value
	IRQ2E = 1		Pin value

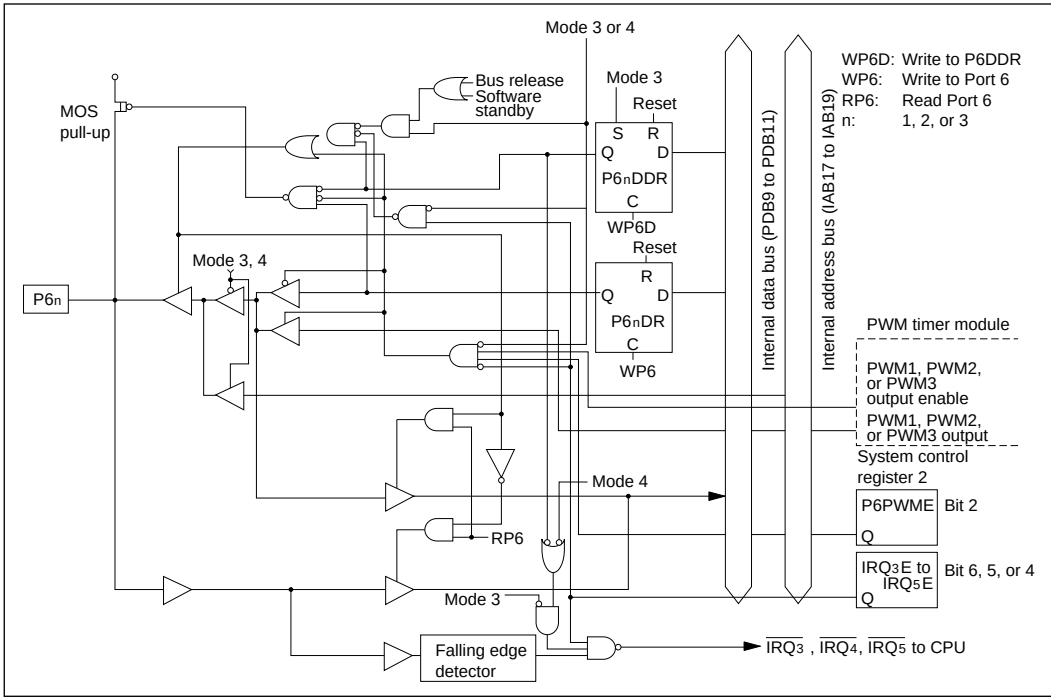


Figure C-6 (b) Schematic Diagram of Port 6, Pin P61 to P63

Table C-6 (b) Port 6 Port Read (Pin P61 to P63)

Mode and Setting				Port Read Data
3				DR value
4	DDR = 0			Pin value
	DDR = 1			DR value
1, 2, 7	IRQnE = 1			Pin value
	IRQnE = 0	P6PWME = 1	PWM output enable	PWM output value
		Other than the above settings	DDR = 0	Pin value
			DDR = 1	DR value

C.7 Schematic Diagram of Port 7

Figure C-7 (a) to (e) gives a schematic view of the port 7 input/output circuits.

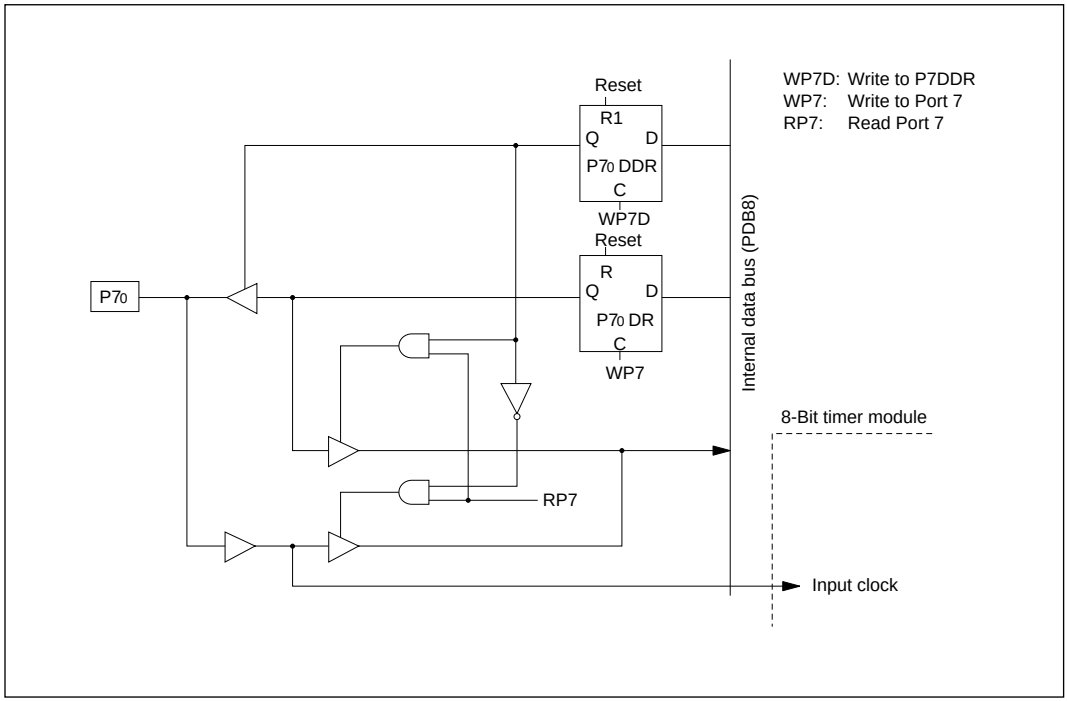


Figure C-7 (a) Schematic Diagram of Port 7, Pin P7₀

Table C-7 (a) Port 7 Port Read (Pin P7₀)

Setting	Port Read Data
DDR = 0	Pin value
DDR = 1	DR value

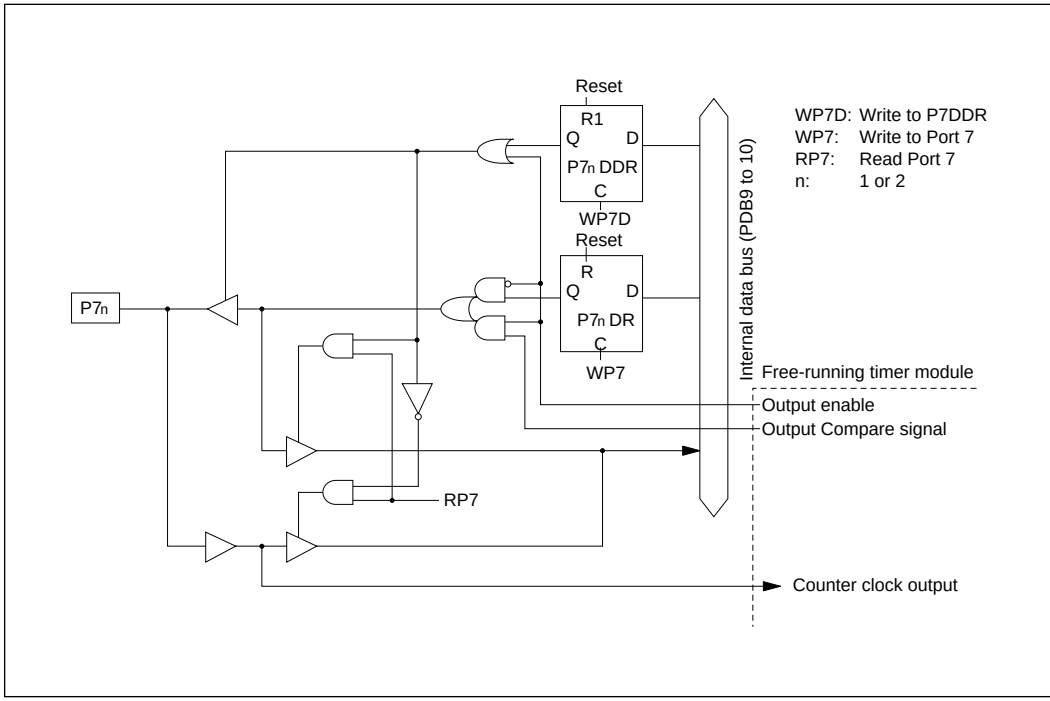


Figure C-7 (b) Schematic Diagram of Port 7, Pins P71 and P72

Table C-7 (b) Port 7 Port Read (Pins P71, P72)

Setting	Port Read Data
DDR = 0	Pin value
DDR = 1	DR value

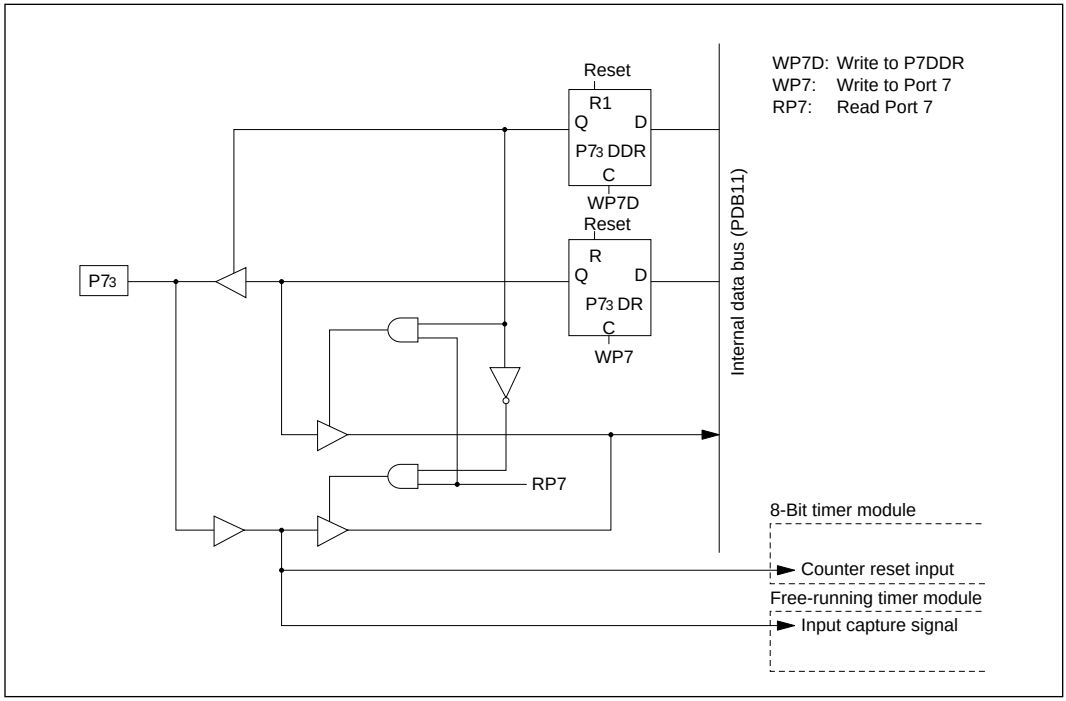


Figure C-7 (c) Schematic Diagram of Port 7, Pin P73

Table C-7 (c) Port 7 Port Read (Pin P73)

Setting	Port Read Data
DDR = 0	Pin value
DDR = 1	DR value

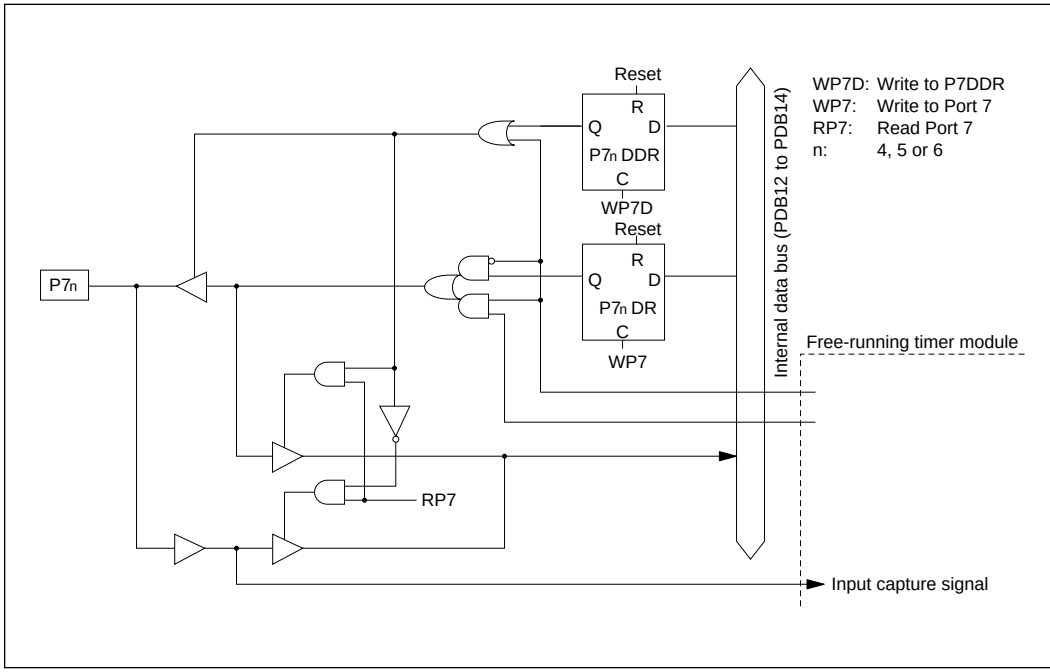


Figure C-7 (d) Schematic Diagram of Port 7, Pins P74, P75 and P76

Table C-7 (d) Port 7 Port Read (Pins P74 to P76)

Setting		Port Read Data
Output enable		Output compare output value
Output disable	DDR = 0	Pin value
	DDR = 1	DR value

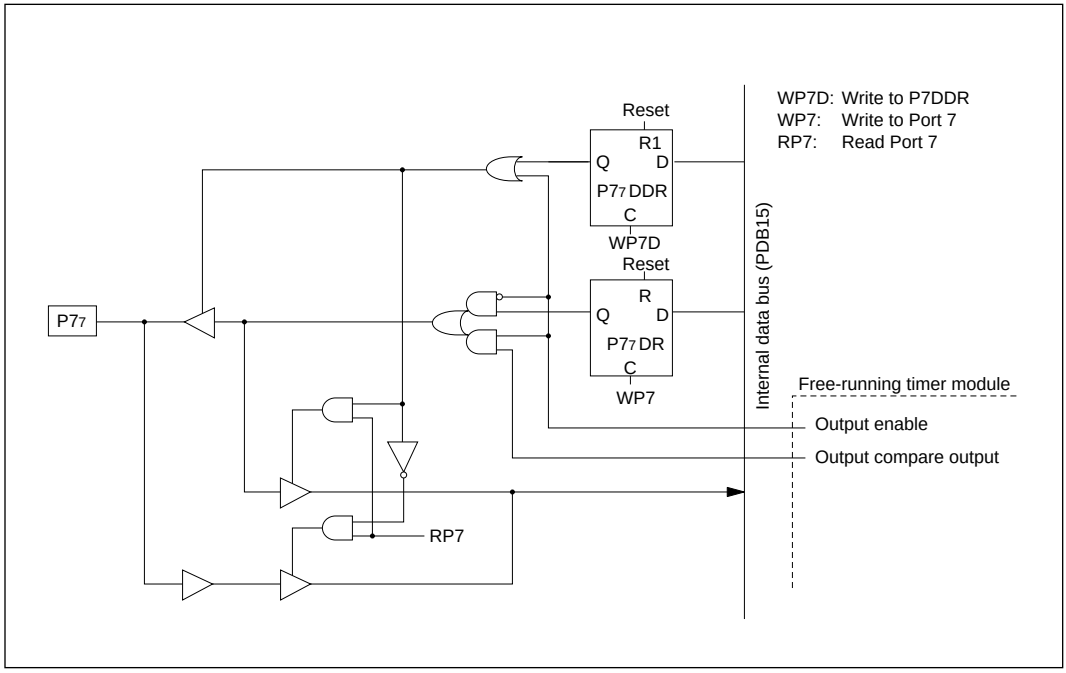


Figure C-7 (e) Schematic Diagram of Port 7, Pin P77

Table C-7 (e) Port 7 Port Read (Pin P77)

Setting		Port Read Data
Output enable		Output compare output value
Output disable	DDR = 0	Pin value
	DDR = 1	DR value

C.8 Schematic Diagram of Port 8

Figure C-8 gives a schematic view of the port 8 input circuits.

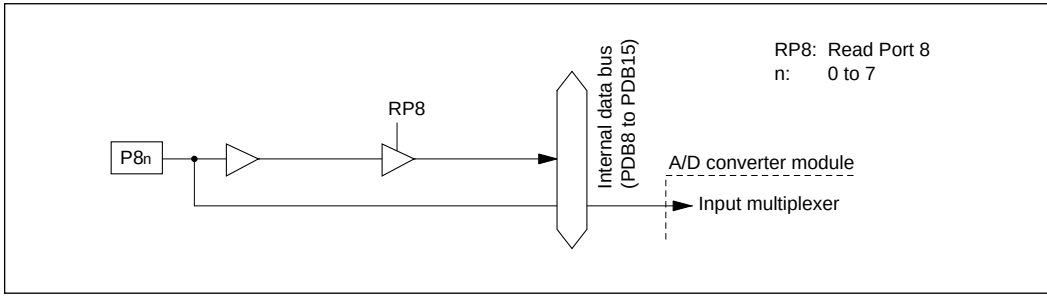


Figure C-8 Schematic Diagram of Port 8

C.9 Schematic Diagram of Port 9

Figure C-9 (a) to (g) gives a schematic view of the port 9 input/output circuits.

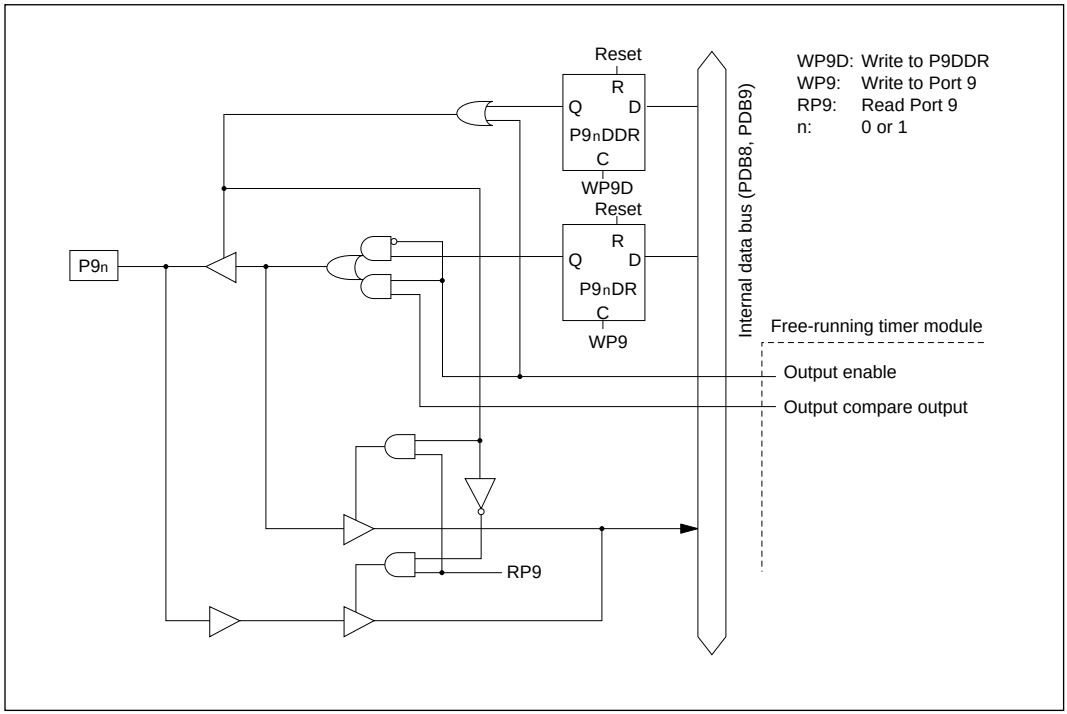


Figure C-9 (a) Schematic Diagram of Port 9, Pins P9$_0$ and P9$_1$

Table C-9 (a) Port 9 Port Read (Pins P9$_0$, P9$_1$)

Setting		Port Read Data
Output enable		Output compare output value
Output disable	DDR = 0	Pin value
	DDR = 1	DR value

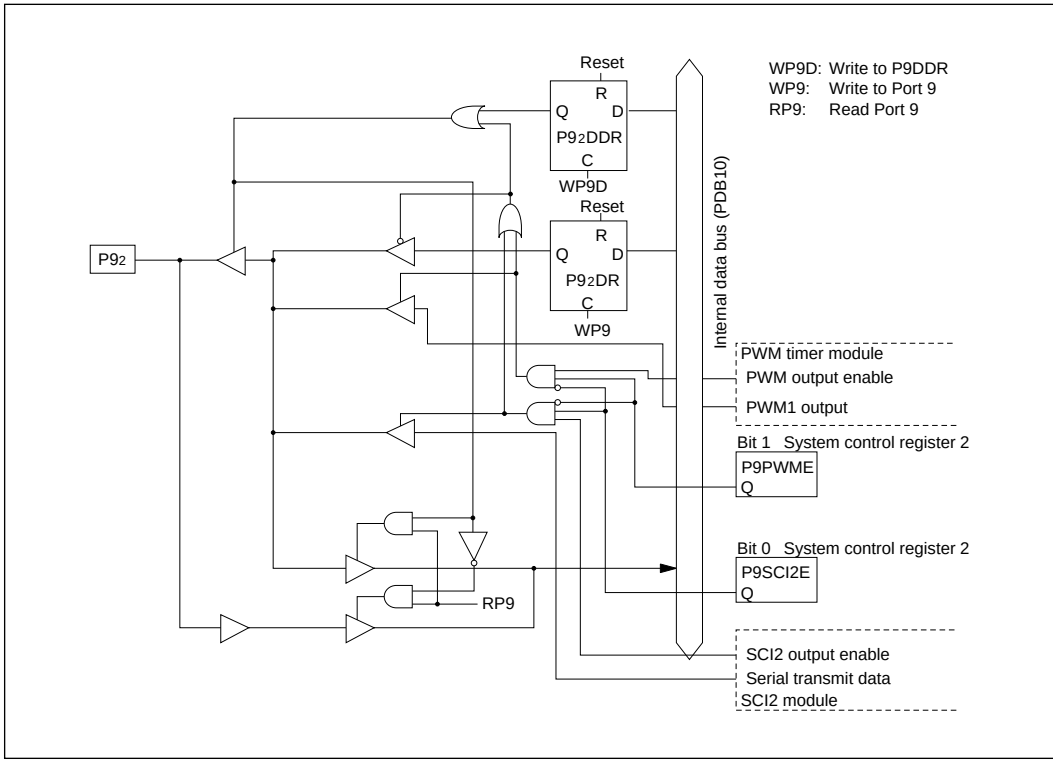


Figure C-9 (b) Schematic Diagram of Port 9, Pin P9$_2$

Table C-9 (b) Port 9 Port Read (Pin P9$_2$)

Setting				Port Read Data (Pin P9$_2$)
Port 9	Port 9	SCI2 output enable		Serial transmit data value
SCI2	PWM	SCI2 output disable	DDR = 0	Pin value
enable	disable		DDR = 1	DR value
Port 9	Port 9	PWM output enable		PWM1 output value
SCI2	PWM	PWM output disable	DDR = 0	Pin value
disable	enalbe		DDR = 1	DR value
Port 9	Port 9	PWM and SCI2	DDR = 0	Pin value
SCI2	PWM	output either enabled	DDR = 1	DR value
disable	disable	or disabled		
Port 9	Port 9		DDR = 0	Pin value
SCI2	PWM		DDR = 1	DR value
enable	enable			

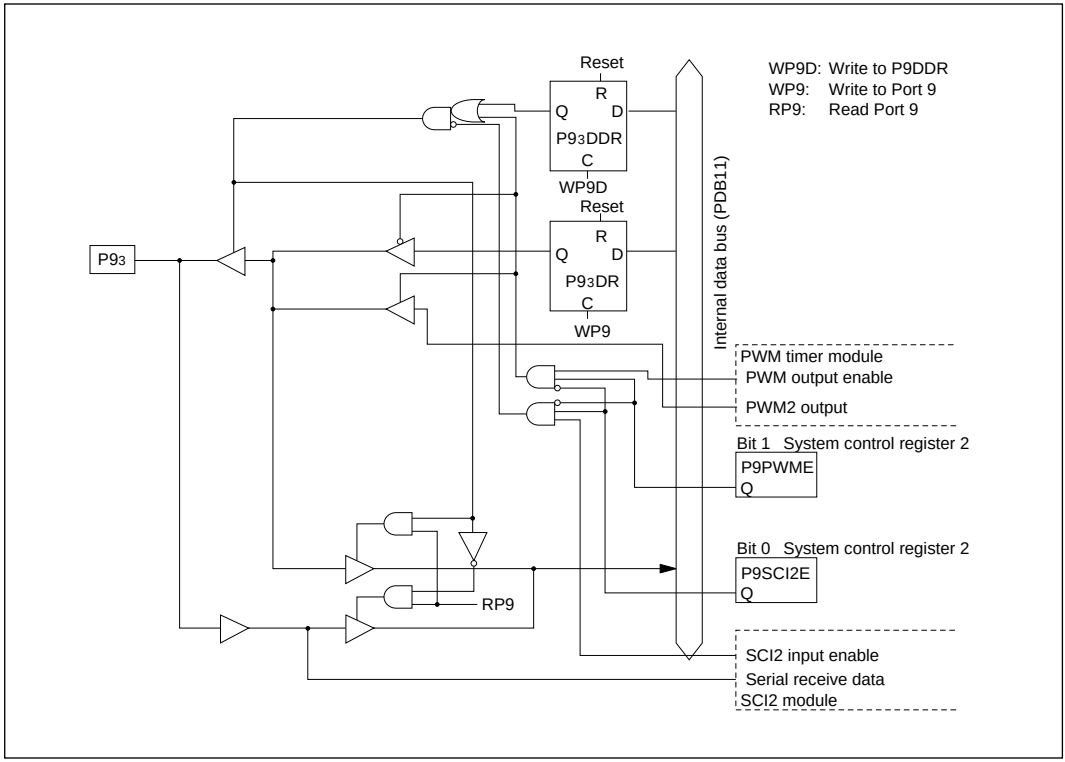


Figure C-9 (c) Schematic Diagram of Port 9, Pin P93

Table C-9 (c) Port 9 Port Read (Pin P93)

Setting				Port Read Data (Pin P93)
Port 9	Port 9	SCI2 input enable		Serial receive data value
SCI2	PWM	SCI2 input disable	DDR = 0	Pin value
enable	disable		DDR = 1	DR value
Port 9	Port 9	PWM output enable		PWM2 output value
SCI2	PWM	PWM output disable	DDR = 0	Pin value
disable	enalbe		DDR = 1	DR value
Port 9	Port 9	PWM and SCI2	DDR = 0	Pin value
SCI2	PWM	input either enabled	DDR = 1	DR value
disable	disable	or disabled		
Port 9	Port 9		DDR = 0	Pin value
SCI2	PWM		DDR = 1	DR value
enable	enable			

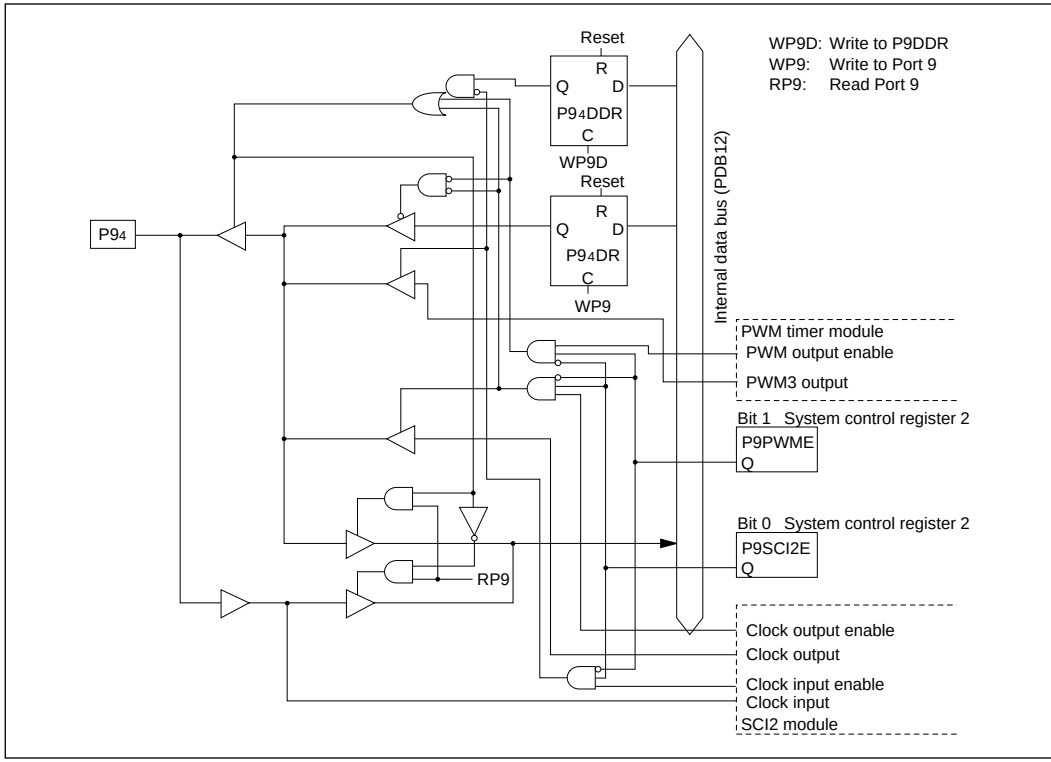


Figure C-9 (d) Schematic Diagram of Port 9, Pin P94

Table C-9 (d) Port 9 Port Read (Pin P94)

Setting				Port Read Data (Pin P94)
Port 9	Port 9	Clock input enable		Input clock value
SCI2	PWM	Clock output enable		Output clock value
enable	disable	Clock input and output disable	DDR = 0	Pin value
			DDR = 1	DR value
Port 9	Port 9	Clock input, clock output, and PWM	DDR = 0	Pin value
SCI2	PWM	output enabled or disabled	DDR = 1	DR value
enable	enalbe			
Port 9		PWM output enable		PWM3 output value
SCI2		PWM output disable	DDR = 0	Pin value
disable			DDR = 1	DR value
Port 9	Port 9	Clock input, clock output, and PWM	DDR = 0	Pin value
SCI2	PWM	output either enabled or disabled	DDR = 1	DR value
disable	disable			

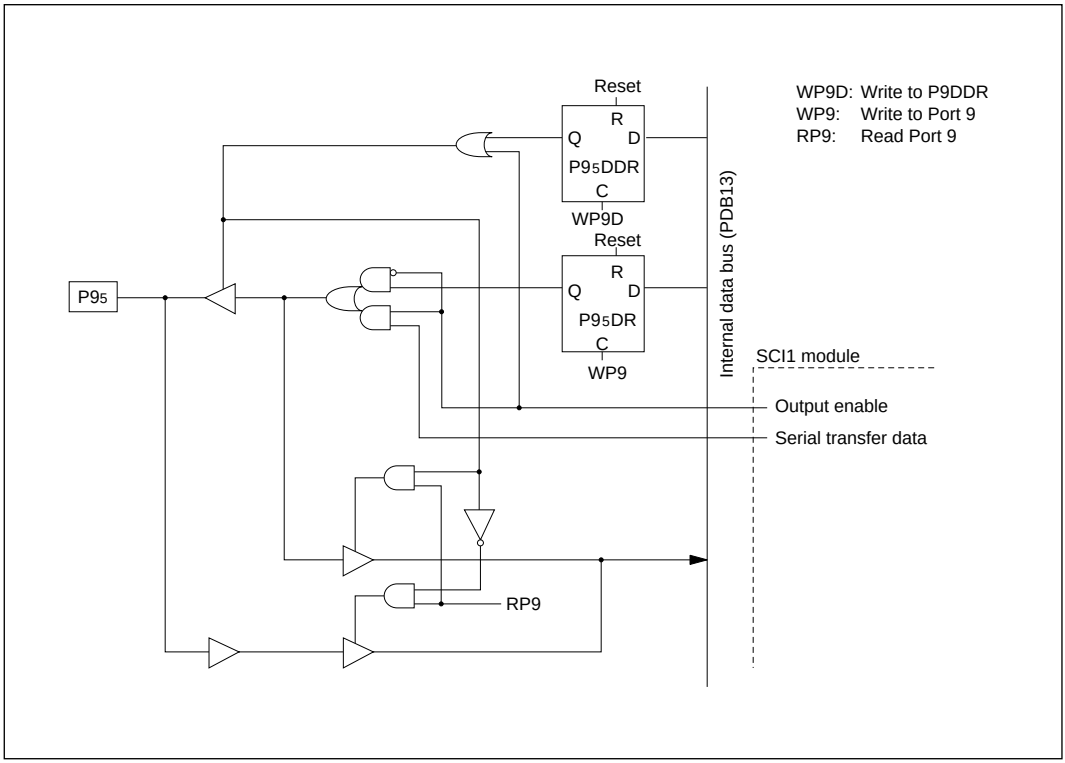


Figure C-9 (e) Schematic Diagram of Port 9, Pin P95

Table C-9 (e) Port 9 Port Read (Pin P95)

Setting		Port Read Data
Output enable		Serial transfer data
Output disable	DDR = 0	Pin value
	DDR = 1	DR value

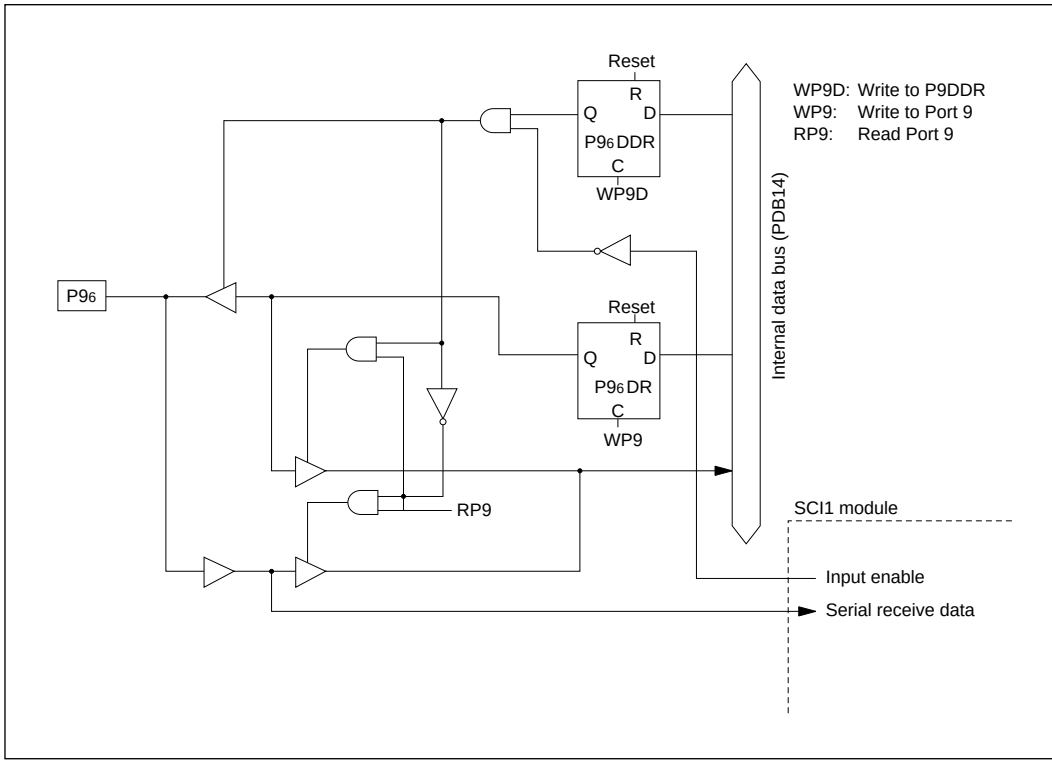


Figure C-9 (f) Schematic Diagram of Port 9, Pin P96

Table C-9 (f) Port 9 Port Read (Pin P96)

Setting		Port Read Data
Output enable		Serial transfer data
Output disable	DDR = 0	Pin value
	DDR = 1	DR value

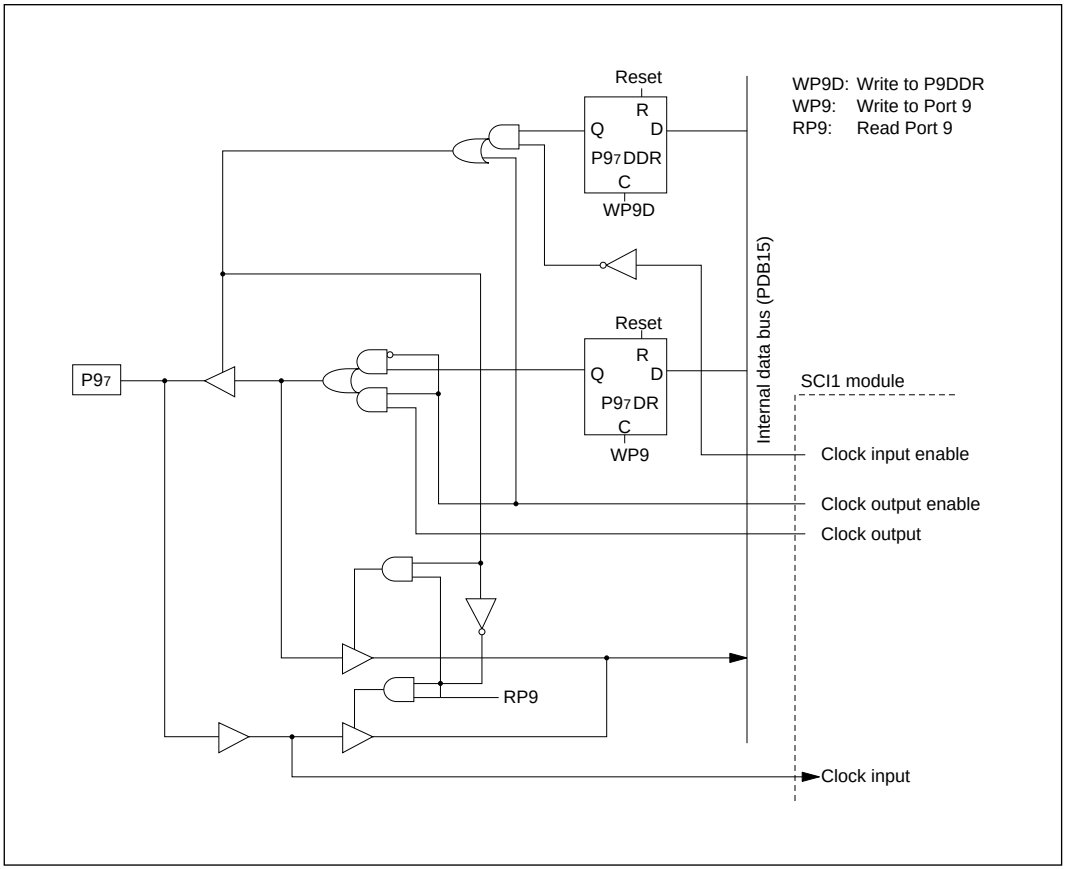


Figure C-9 (g) Schematic Diagram of Port 9, Pin P9$_7$

Table C-9 (g) Port 9 Port Read (Pin P9$_7$)

Setting		Port Read Data
Clock input enable		Input clock value
Clock output enable		Output clock value
Clock input/output enable	DDR = 0	Pin value
	DDR = 1	DR value

Appendix D Memory Maps

Table D-1 H8/534 Memory Map

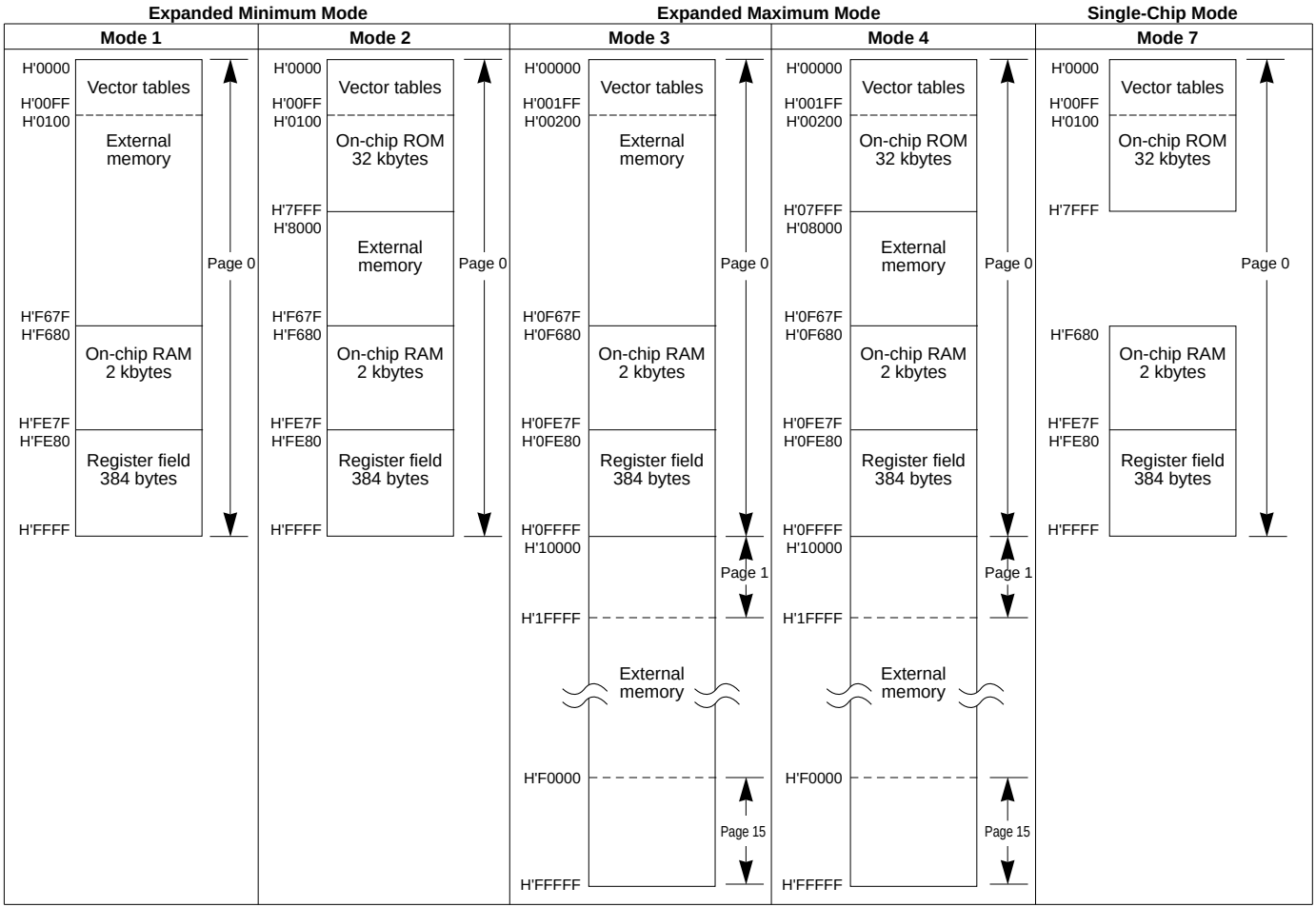
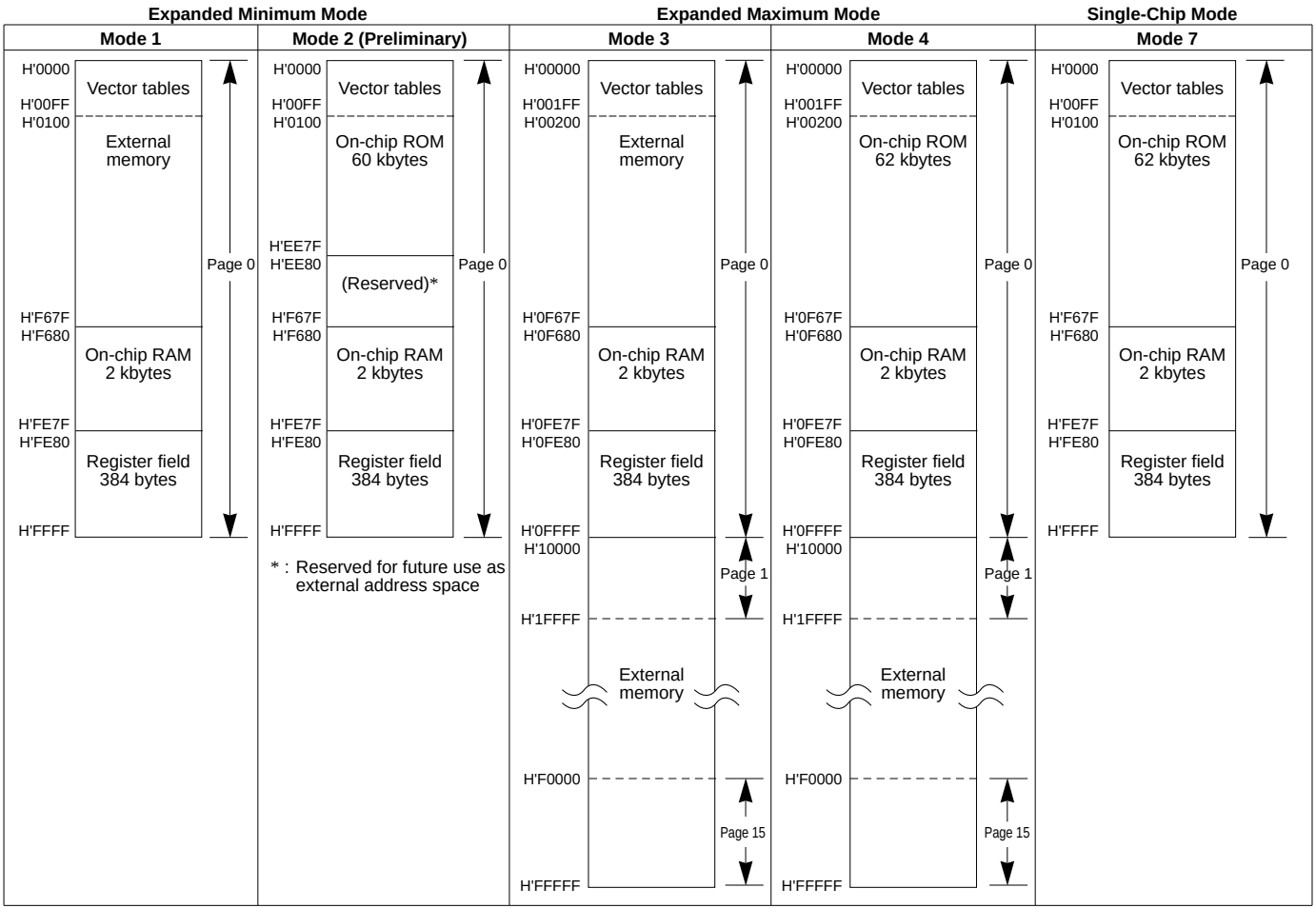


Table D-2 H8/536 Memory Map



Appendix E Pin States

E.1 Port State of Each Pin State

Table E-1 Port State

Port Pin Name	Mode	Reset	Hardware Standby Mode	Software Standby Mode	Sleep Mode	Bus Release Mode	Program Execution State (Normal Operation)
$P1_7$ to $P1_2$ TMO, $\overline{IRQ_1}$, $\overline{IRQ_0}$ $\overline{WAIT}$, BREQ, $\overline{BACK}$	1	T	T	keep*1	keep*3	keep*4	Input/output port or control signal Input/ output
	2						
	3						
	4						
	7			keep*2	keep	——	Input/output port
$P1_1$/E $P1_0$/ø	1	Clock output	T	(DDR = 1) ø = H E = L (DDR = 0) T	(DDR = 1) Clock output (DDR = 0) T	(DDR = 1) Clock output (DDR = 0) T	(DDR = 1) Clock output (DDR = 0) Input port
	2						
	3						
	4						
	7			T		——	
$P2_4$ to $P2_0$ $\overline{WR}$, $\overline{RD}$, $\overline{DS}$, R/$\overline{W}$, $\overline{AS}$	1	H		T	H	T	$\overline{WR}$, $\overline{RD}$, $\overline{DS}$, R/$\overline{W}$, $\overline{AS}$
	2						
	3						
	4						
	7	T		keep	keep	——	Input/output port
$P3_7$ to $P3_0$ D_7 to D_0	1	T	T	T	T	T	D_7 to D_0
	2						
	3						
	4						
	7			keep	keep	——	Input/output port
$P4_7$ to $P4_0$ A_7 to A_0	1	L	T	T	L	T	A_7 to A_0
	2						
	3						
	4						
	7	T		keep	keep	——	Input/output port
$P5_7$ to $P5_0$ A_{15} to A_8	1	L	T	T	L	T	A_{15} to A_8
	2	T		T*6	*5	T*6	Address bus or input port
	3	L		T	L	T	A_{15} to A_8
	4	T		T*6	*5	T*6	Address bus or input port
	7			keep	keep	——	Input/output port

(Continued on next page)

Table E-1 Port State (cont)

Port Pin Name	Mode	Reset	Hardware Standby Mode	Software Standby Mode	Sleep Mode	Bus Release Mode	Program Execution State (Normal Operation)
$P6_3$ to $P6_0$	1			keep	keep	keep	Input/output port
A_{19} to A_{16}	2	T					
	3	L	T	T	L	T	A_{19} to A_{16}
	4	T		T*6	*5	T*6	Address bus or input port
	7			keep	keep	——	Input/output port
$P7_7$ to $P7_0$	1						
	2						
	3	T	T	keep*2	keep	keep	Input/output port
	4						
	7						
$P8_7$ to $P8_0$	1						
	2						
	3	T	T	T	T	T	Input port
	4						
	7						
$P9_7$ to $P9_0$	1						
	2						
	3	T	T	keep*2	keep	keep	Input/output port
	4						
	7						

H: High logic level

L: Low logic level

T: High-Impedance state

keep: Input ports are in the high-impedance state. Output ports hold their previous output values. If DDR = 0 and DR = 1 in ports 5 and 6, the MOS pull-ups remain on.

*1 The on-chip supporting modules are reset, so $P1_7$ becomes an input or output port controlled by DDR and DR. If $P1_2$ is programmed for $\overline{BACK}$ output, it goes to the high-impedance state.

*2 The on-chip supporting modules are reset, so these pins become input or output ports controlled by DDR and DR.

*3 $\overline{BREQ}$ can be received. $\overline{BACK}$ is High.

*4 $\overline{BACK}$ is Low.

*5 Address outputs are Low. Input ports are in the high-impedance state, or the MOS pull-ups are on.

*6 Pins used as input ports with the MOS pull-up on (DDR = 0, DR = 1) do not go to the high-impedance state. The MOS pull-up remains on.

Table E-2 MOS Pull-Up State

Port	Mode	Reset	Hardware Standby Mode	Other Operating States*
$P5_7$ to $P5_0$ A_{15} to A_8	1	OFF	OFF	OFF
	2			ON/OFF
	3			OFF
	4			ON/OFF
	7			
$P6_3$ to $P6_0$ A_{19} to A_{16}	1	OFF	OFF	ON/OFF
	2			
	3			OFF
	4			ON/OFF
	7			

Notes

OFF: The MOS pull-up is always OFF.

ON/OFF: The MOS pull-up is on when DDR = 0 and DR = 1, and is off at other times.

* Including software standby mode.

E.2 Pin States in Reset State

1. Mode 1

Figure E-1 shows how the pin states change when the $\overline{\text{RES}}$ pin goes Low during external memory access in mode 1.

As soon as $\overline{\text{RES}}$ goes Low, all ports are initialized to the input (high-impedance) state. The $\overline{\text{AS}}$, $\overline{\text{DS}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ signals all go High. The data bus (D_7 to D_0) is placed in the high-impedance state.

The address bus and the R/$\overline{\text{W}}$ signal are initialized 1.5 ø clock periods after the Low state of the $\overline{\text{RES}}$ pin is sampled. All address bus signals are made Low. The R/$\overline{\text{W}}$ signal is made High.

The clock output pins $P1_0$/ø and $P1_1$/E are initialized 0.5 ø clock periods after the Low state of the $\overline{\text{RES}}$ pin is sampled. Both pins are initialized to the output state.

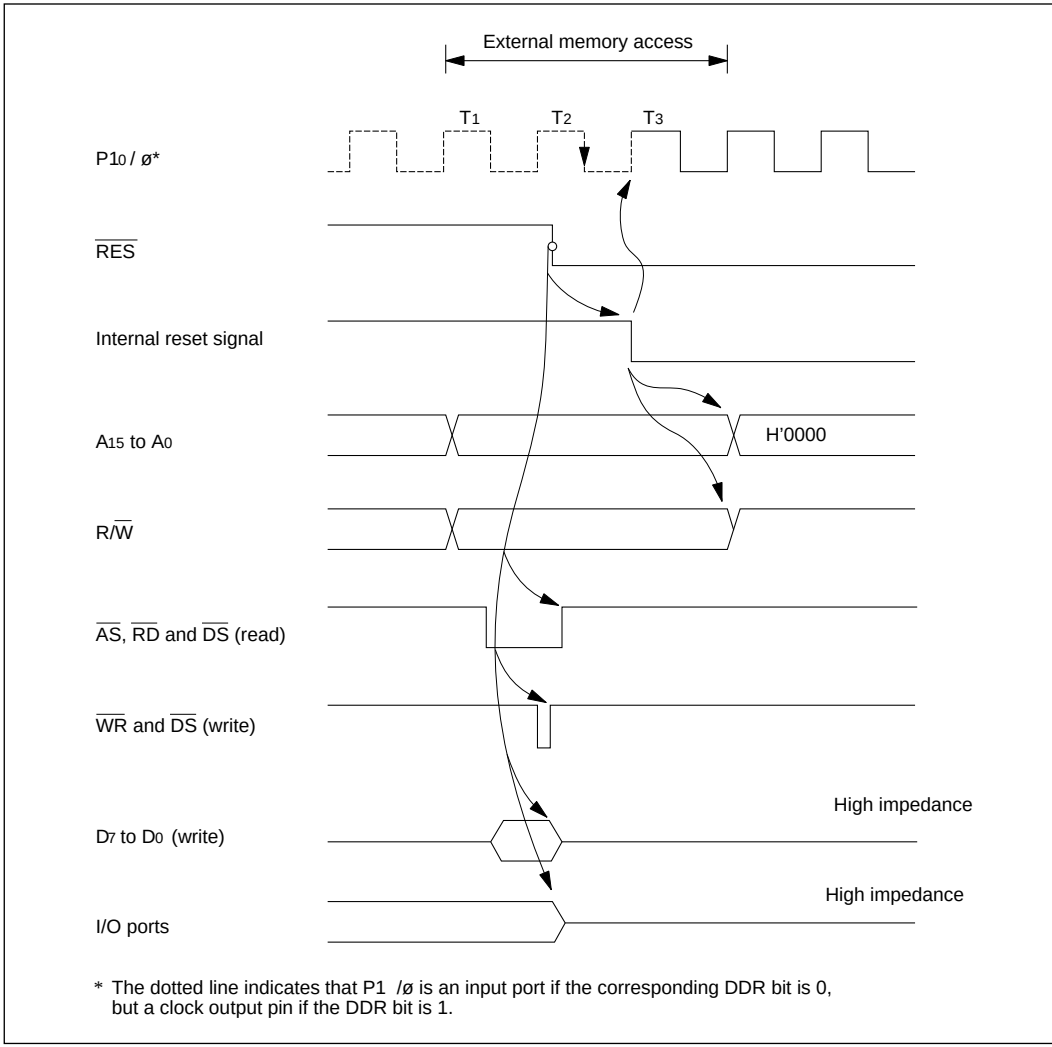


Figure E-1 Reset during Memory Access (Mode 1)

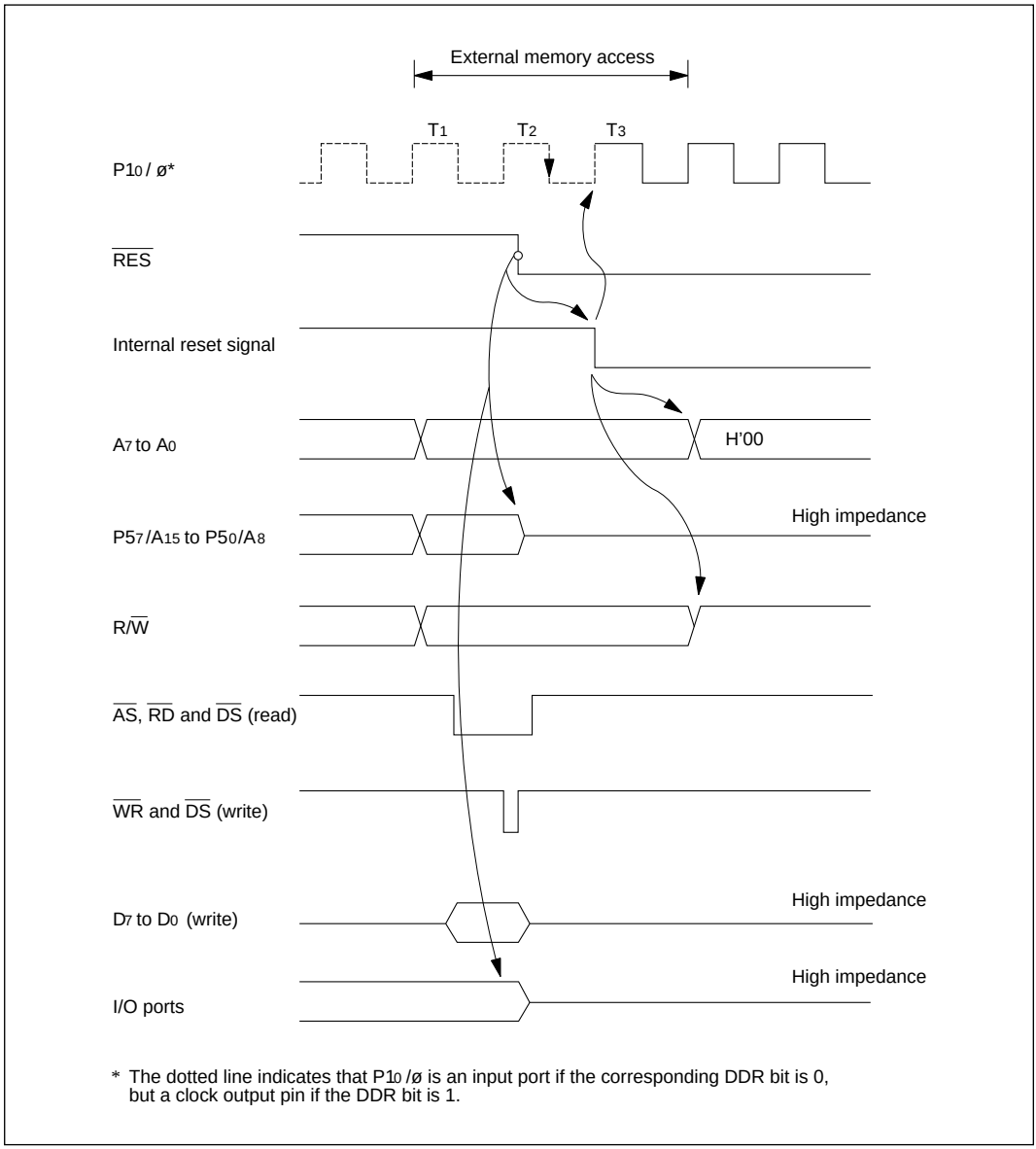
2. Mode 2

Figure E-4 shows how the pin states change when the $\overline{\text{RES}}$ pin goes Low during external memory access in mode 2.

As soon as $\overline{\text{RES}}$ goes Low, all ports are initialized to the input (high-impedance) state. The $\overline{\text{AS}}$, $\overline{\text{DS}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ signals all go High. The data bus (D_7 to D_0) is placed in the high-impedance state. Pins $P5_7/A_{15}$ to $P5_0/A_8$ of the address bus are initialized as input ports.

Pins A7 to A0 of the address bus and the R/$\overline{W}$ signal are initialized 1.5 ø clock periods after the Low state of the $\overline{RES}$ pin is sampled. Pins A7 to A0 are made Low. The R/$\overline{W}$ signal is made High.

The clock output pins P10/ø and P11/E are initialized 0.5 ø clock periods after the Low state of the $\overline{RES}$ pin is sampled. Both pins are initialized to the output state.



* The dotted line indicates that P10 /ø is an input port if the corresponding DDR bit is 0, but a clock output pin if the DDR bit is 1.

Figure E-2 Reset during Memory Access (Mode 2)

3. Mode 3

Figure E-4 shows how the pin states change when the $\overline{\text{RES}}$ pin goes Low during external memory access in mode 3.

As soon as $\overline{\text{RES}}$ goes Low, all ports are initialized to the input (high-impedance) state. The $\overline{\text{AS}}$, $\overline{\text{DS}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ signals all go High. The data bus (D_7 to D_0) is placed in the high-impedance state.

The address bus and the R/$\overline{\text{W}}$ signal are initialized 1.5 ø clock periods after the Low state of the $\overline{\text{RES}}$ pin is sampled. All address bus signals are made Low. The R/$\overline{\text{W}}$ signal is made High.

The clock output pins $P1_0$/ø and $P1_1$/E are initialized 0.5 ø clock periods after the Low state of the $\overline{\text{RES}}$ pin is sampled. Both pins are initialized to the output state.

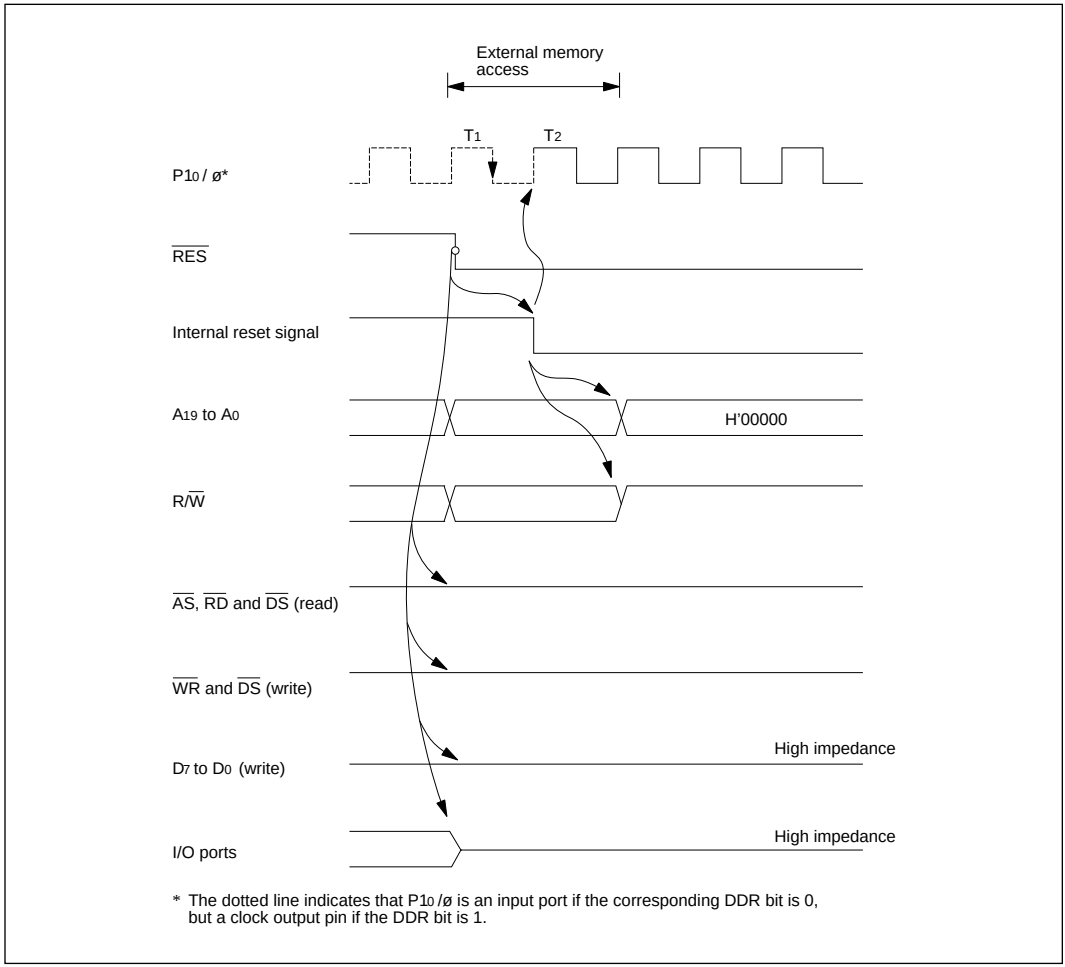


Figure E-3 Reset during Memory Access (Mode 3)

4. Mode 4

Figure E-4 shows how the pin states change when the $\overline{\text{RES}}$ pin goes Low during external memory access in mode 4.

As soon as $\overline{\text{RES}}$ goes Low, all ports are initialized to the input (high-impedance) state. The $\overline{\text{AS}}$, $\overline{\text{DS}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ signals all go High. The data bus (D7 to D0) is placed in the high-impedance state. Pins P57/A15 to P50/A8 of the address bus and pins P63/A19 to P60/A16 of the page address bus are initialized as input ports.

Pins A7 to A0 of the address bus and the R/$\overline{W}$ signal are initialized 1.5 ø clock periods after the Low state of the $\overline{RES}$ pin is sampled. Pins A7 to A0 are made Low. The R/$\overline{W}$ signal is made High.

The clock output pins P10/ø and P11/E are initialized 0.5 ø clock periods after the Low state of the $\overline{RES}$ pin is sampled. Both pins are initialized to the output state.

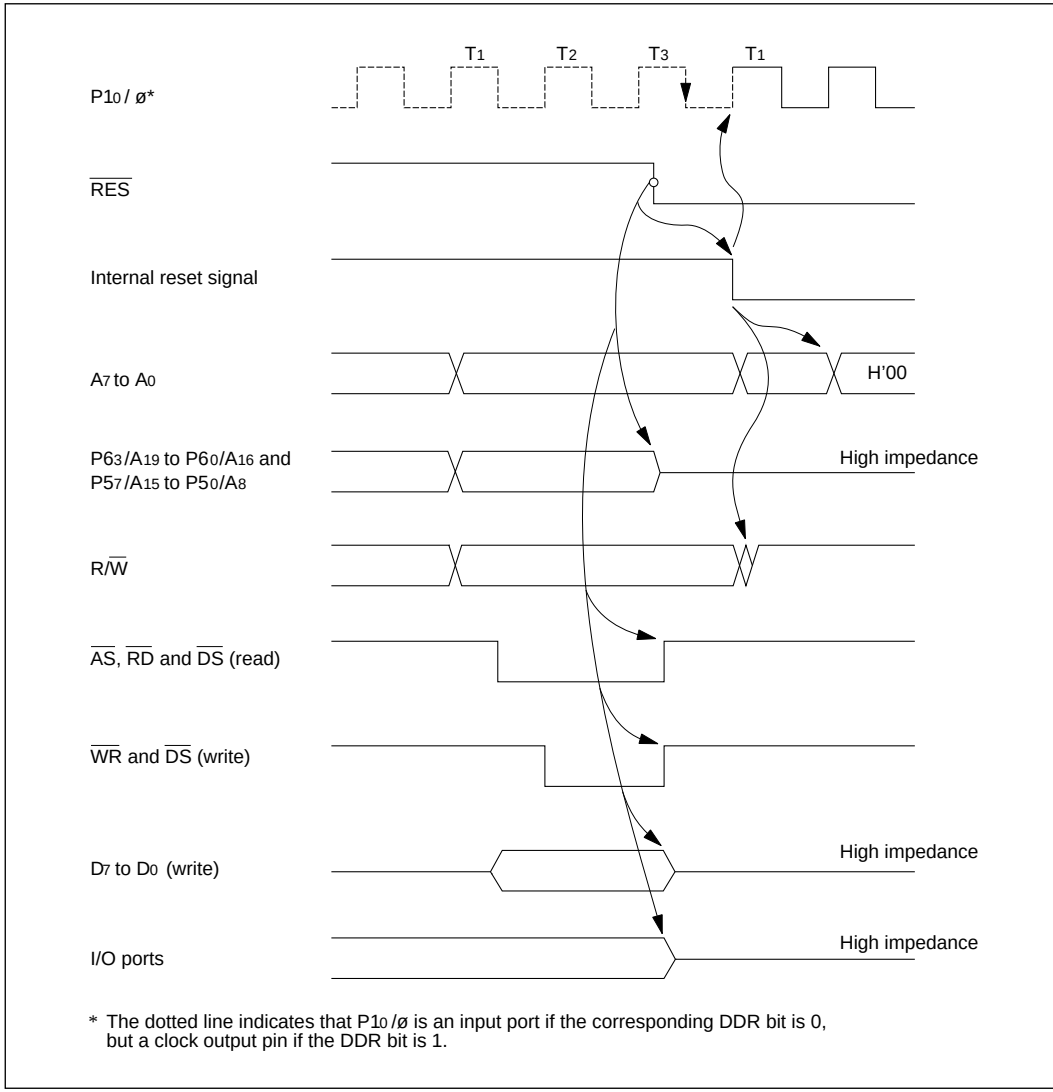


Figure E-4 Reset during Memory Access (Mode 4)

5. Mode 7

Figure E-5 shows how the pin states change when the $\overline{\text{RES}}$ pin goes Low in mode 7.

As soon as $\overline{\text{RES}}$ goes Low, all ports are initialized to the input (high-impedance) state.

The clock output pins P1$_0$/ø and P1$_1$/E are initialized 0.5 ø clock periods after the Low state of the $\overline{\text{RES}}$ pin is sampled. Both pins are initialized to the output state.

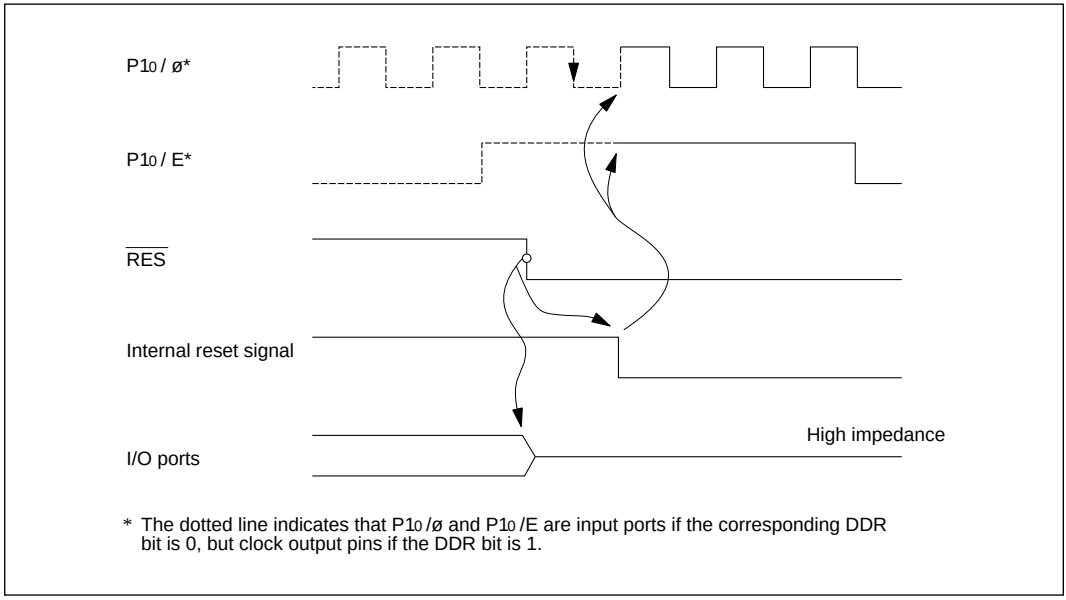
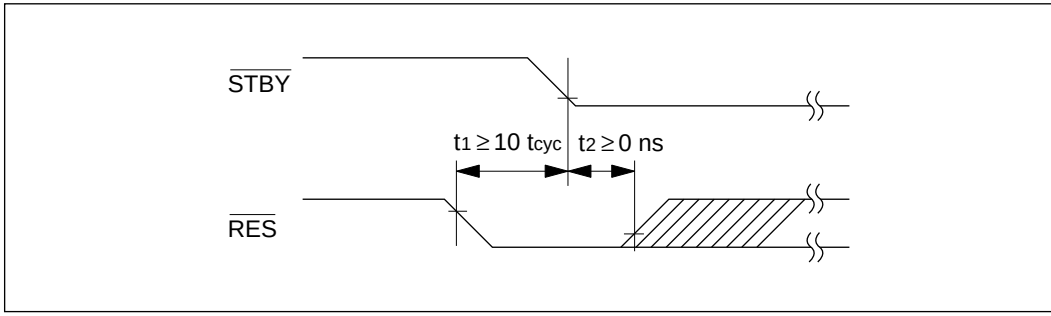


Figure E-5 Reset during Memory Access (Mode 7)

Appendix F Timing of Transition to and Recovery from Hardware Standby Mode

Timing of Transition to Hardware Standby Mode

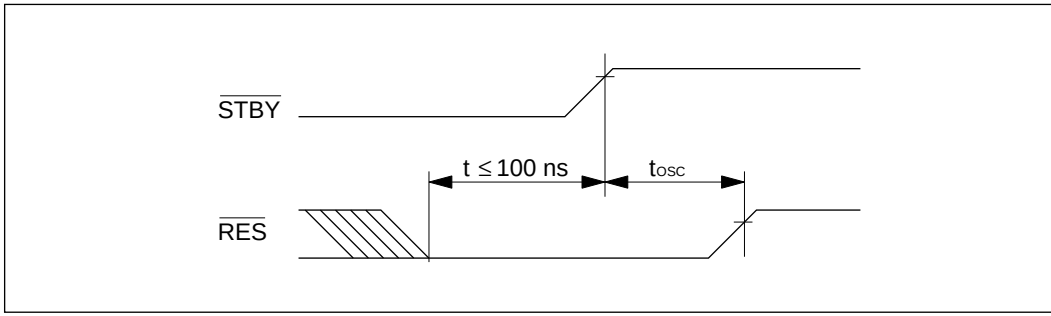
(1) To retain RAM contents when the RAME bit in RAMCR is set to 1, drive the $\overline{\text{RES}}$ signal line low 10 system clock cycles before the $\overline{\text{STBY}}$ signal, at a time when RAM is not being accessed.



(2) When the RAME bit in RAMCR is cleared to 0, or when it is not necessary to retain RAM contents, $\overline{\text{RES}}$ need not be driven low as in (1).

Timing of Exit from Hardware Standby Mode

Drive the $\overline{\text{RES}}$ signal line low approximately 100 ns before the rise of the $\overline{\text{STBY}}$ signal.



Appendix G Package Dimensions

Figure G-1 shows the dimensions of the CP-84 package. Figure G-2 shows the dimensions of the CG-84 package. Figure G-3 shows the dimensions of the FP-80A package.

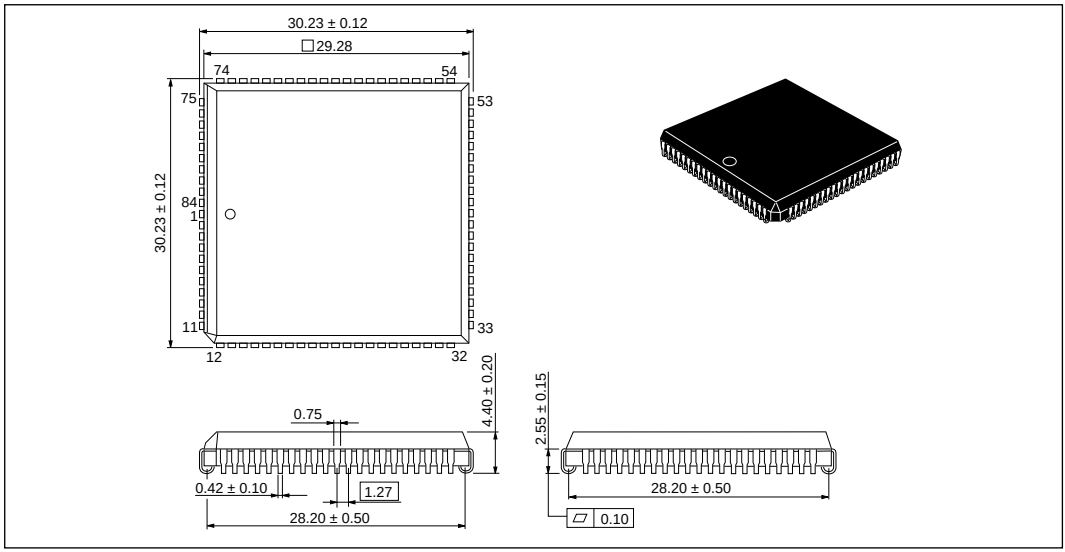


Figure G-1 Package Dimensions (CP-84)

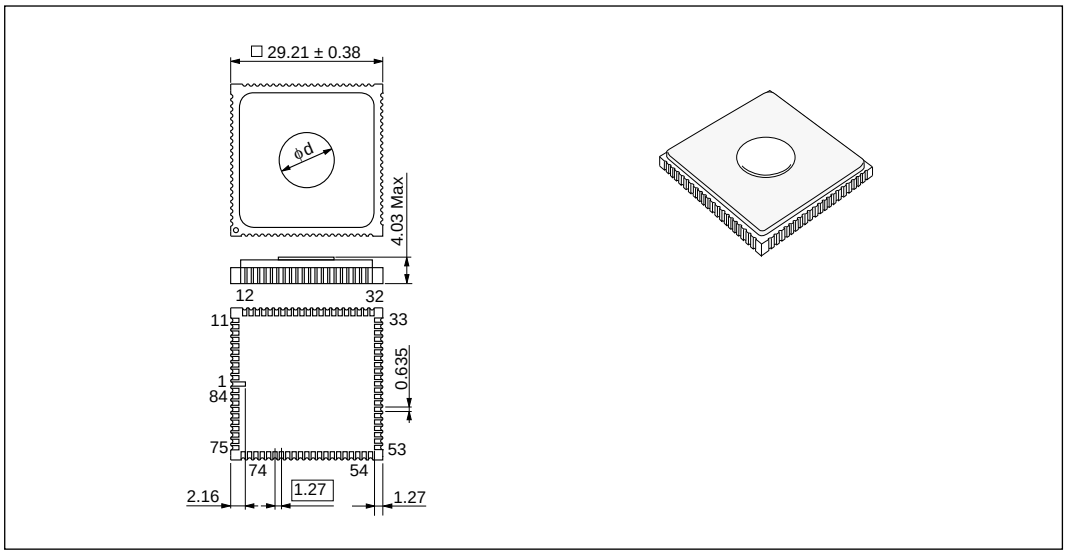


Figure G-2 Package Dimensions (CG-84)

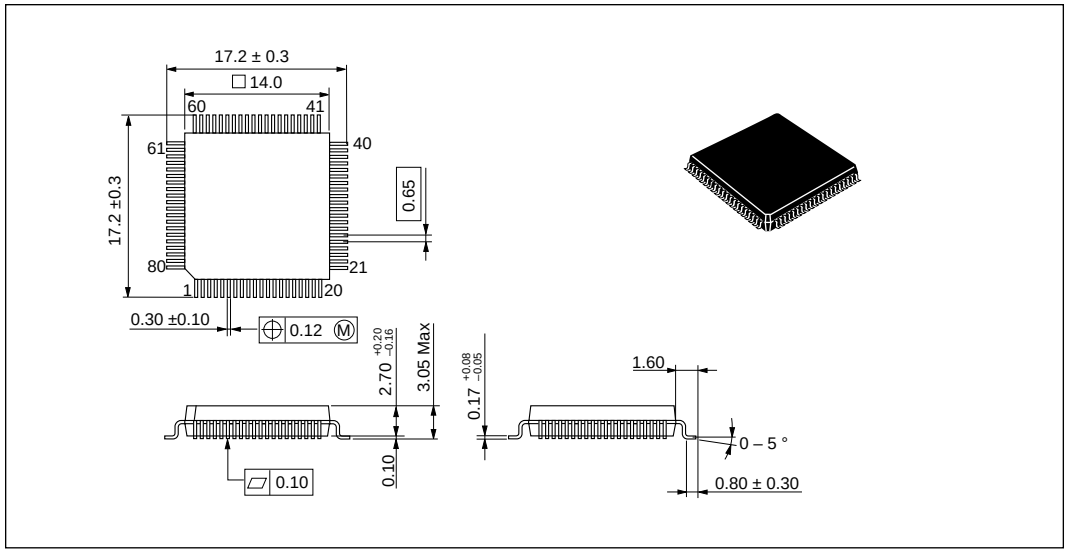


Figure G-3 Package Dimensions (FP-80A)

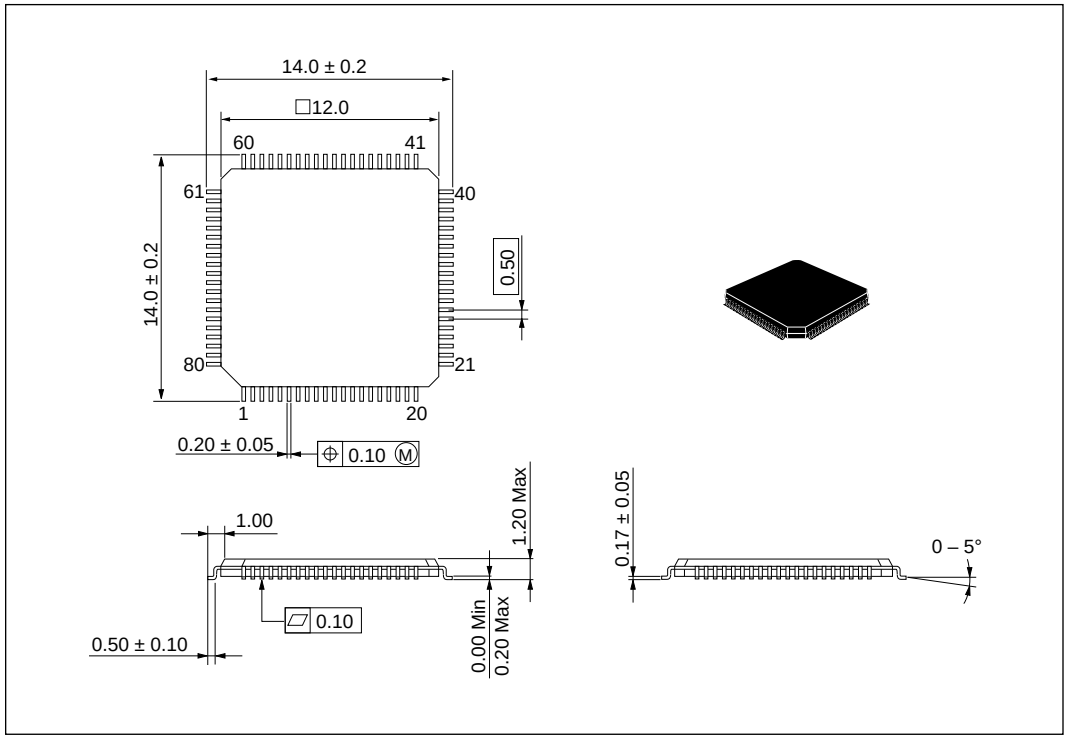


Figure G-4 Package Dimensions (TFP-80C)