

# Hitachi Single-Chip Microcomputer

## H8/3437 Series

H8/3437  
HD6473437, HD6433437

H8/3436  
HD6433436

H8/3434  
HD6473434, HD6433434

H8/3434 F-ZTAT<sup>TM</sup>  
HD64F3434

H8/3437 F-ZTAT<sup>TM</sup>  
HD64F3437

## Hardware Manual

## Preface

The H8/3437 Series is a series of high-performance microcontrollers with a fast H8/300 CPU core and a set of on-chip supporting functions optimized for embedded control. These include ROM, RAM, four types of timers, a serial communication interface, optional I<sup>2</sup>C bus interface, host interface, A/D converter, D/A converter, I/O ports, and other functions needed in control system configurations, so that compact, high-performance systems can be implemented easily. The series includes the H8/3437 with 60-kbyte ROM and 2-kbyte RAM, the H8/3436 with 48-kbyte ROM and 2-kbyte RAM, and the H8/3434 with 32-kbyte ROM and 1-kbyte RAM.

The H8/3437, H8/3436, and H8/3434 are available in mask-ROM versions. The H8/3437 and H8/3434 are also available in ZTAT<sup>TM</sup>\*1 (zero turn-around time) versions, providing a quick and flexible response to conditions from ramp-up through full-scale volume production, even for applications with frequently-changing specifications. In addition, the H8/3434 and H8/3437 have F-ZTAT<sup>TM</sup>\*2 (flexible-ZTAT) versions with on-board programmability.

This manual describes the hardware of the H8/3437 Series. Refer to the *H8/300 Series Programming Manual* for a detailed description of the instruction set.

Notes:

1. ZTAT<sup>TM</sup> is a trademark of Hitachi, Ltd.
2. F-ZTAT<sup>TM</sup> is a trademark of Hitachi, Ltd.

## Main Amendments and Additions in this Edition

Page	Title	Amendment/Addition
All	Addition of H8/3437F-ZTAT version	
	Addition of ROM descriptions	
	Section 18    Mask ROM Version/ZTAT Version	
	Section 19    32-Kbyte Flash Memory Version	
	Section 20    60-Kbyte Flash Memory Version	
491	Table 22-2    DC Characteristics (5-V Version)	Values changed for “Input capacitance” item
495	Table 22-3    DC Characteristics (4-V Version)	Values changed for “Input capacitance” item
498	Table 22-4    DC Characteristics (3-V Version)	Values changed for “Input capacitance” item
505	Table 22-10    Timing Conditions of On-Chip Supporting Modules	Values changed for “HIF write cycle” item

# Contents

<b>Section 1</b>	<b>Overview</b>	1
1.1	Overview	1
1.2	Block Diagram	5
1.3	Pin Assignments and Functions	6
1.3.1	Pin Arrangement	6
1.3.2	Pin Functions	7
<b>Section 2</b>	<b>CPU</b>	17
2.1	Overview	17
2.1.1	Features	17
2.1.2	Address Space	18
2.1.3	Register Configuration	18
2.2	Register Descriptions	19
2.2.1	General Registers	19
2.2.2	Control Registers	19
2.2.3	Initial Register Values	20
2.3	Data Formats	21
2.3.1	Data Formats in General Registers	22
2.3.2	Memory Data Formats	23
2.4	Addressing Modes	24
2.4.1	Addressing Mode	24
2.4.2	Calculation of Effective Address	26
2.5	Instruction Set	30
2.5.1	Data Transfer Instructions	31
2.5.2	Arithmetic Operations	33
2.5.3	Logic Operations	34
2.5.4	Shift Operations	34
2.5.5	Bit Manipulations	36
2.5.6	Branching Instructions	41
2.5.7	System Control Instructions	43
2.5.8	Block Data Transfer Instruction	44
2.6	CPU States	46
2.6.1	Overview	46
2.6.2	Program Execution State	47
2.6.3	Exception-Handling State	47
2.6.4	Power-Down State	47
2.7	Access Timing and Bus Cycle	48
2.7.1	Access to On-Chip Memory (RAM and ROM)	48
2.7.2	Access to On-Chip Register Field and External Devices	50

<b>Section 3</b>	<b>MCU Operating Modes and Address Space</b>	53
3.1	Overview	53
3.1.1	Mode Selection	53
3.1.2	Mode and System Control Registers	54
3.2	System Control Register (SYSCR)	54
3.3	Mode Control Register (MDCR)	57
3.4	Address Space Map in Each Operating Mode	58
<b>Section 4</b>	<b>Exception Handling</b>	61
4.1	Overview	61
4.2	Reset	61
4.2.1	Overview	61
4.2.2	Reset Sequence	61
4.2.3	Disabling of Interrupts after Reset	64
4.3	Interrupts	64
4.3.1	Overview	64
4.3.2	Interrupt-Related Registers	66
4.3.3	External Interrupts	70
4.3.4	Internal Interrupts	70
4.3.5	Interrupt Handling	71
4.3.6	Interrupt Response Time	76
4.3.7	Precaution	77
4.4	Note on Stack Handling	78
<b>Section 5</b>	<b>Wait-State Controller</b>	79
5.1	Overview	79
5.1.1	Features	79
5.1.2	Block Diagram	79
5.1.3	Input/Output Pins	80
5.1.4	Register Configuration	80
5.2	Register Description	80
5.2.1	Wait-State Control Register (WSCR)	80
5.3	Wait Modes	82
<b>Section 6</b>	<b>Clock Pulse Generator</b>	85
6.1	Overview	85
6.1.1	Block Diagram	85
6.1.2	Wait-State Control Register (WSCR)	86
6.2	Oscillator Circuit	87
6.3	Duty Adjustment Circuit	92
6.4	Prescaler	92

<b>Section 7</b>	<b>I/O Ports</b>	93
7.1	Overview	93
7.2	Port 1	96
7.2.1	Overview	96
7.2.2	Register Configuration and Descriptions	97
7.2.3	Pin Functions in Each Mode	99
7.2.4	Input Pull-Up Transistors	101
7.3	Port 2	102
7.3.1	Overview	102
7.3.2	Register Configuration and Descriptions	103
7.3.3	Pin Functions in Each Mode	105
7.3.4	Input Pull-Up Transistors	107
7.4	Port 3	108
7.4.1	Overview	108
7.4.2	Register Configuration and Descriptions	109
7.4.3	Pin Functions in Each Mode	111
7.4.4	Input Pull-Up Transistors	112
7.5	Port 4	113
7.5.1	Overview	113
7.5.2	Register Configuration and Descriptions	114
7.5.3	Pin Functions	116
7.6	Port 5	118
7.6.1	Overview	118
7.6.2	Register Configuration and Descriptions	118
7.6.3	Pin Functions	120
7.7	Port 6	121
7.7.1	Overview	121
7.7.2	Register Configuration and Descriptions	122
7.7.3	Pin Functions	124
7.7.4	Input Pull-Up Transistors	126
7.8	Port 7	127
7.8.1	Overview	127
7.8.2	Register Configuration and Descriptions	128
7.9	Port 8	129
7.9.1	Overview	129
7.9.2	Register Configuration and Descriptions	130
7.9.3	Pin Functions	132
7.10	Port 9	134
7.10.1	Overview	134
7.10.2	Register Configuration and Descriptions	135
7.10.3	Pin Functions	137
7.11	Port A	139
7.11.1	Overview	139

7.11.2	Register Configuration and Descriptions.....	139
7.11.3	Pin Functions in Each Mode.....	141
7.11.4	Input Pull-Up Transistors .....	142
7.12	Port B .....	143
7.12.1	Overview.....	143
7.12.2	Register Configuration and Descriptions.....	144
7.12.3	Pin Functions in Each Mode.....	146
7.12.4	Input Pull-Up Transistors .....	147
<b>Section 8 16-Bit Free-Running Timer .....</b>		149
8.1	Overview .....	149
8.1.1	Features.....	149
8.1.2	Block Diagram.....	150
8.1.3	Input and Output Pins .....	151
8.1.4	Register Configuration.....	151
8.2	Register Descriptions.....	152
8.2.1	Free-Running Counter (FRC).....	152
8.2.2	Output Compare Registers A and B (OCRA and OCRB).....	153
8.2.3	Input Capture Registers A to D (ICRA to ICRD).....	153
8.2.4	Timer Interrupt Enable Register (TIER).....	155
8.2.5	Timer Control/Status Register (TCSR) .....	157
8.2.6	Timer Control Register (TCR).....	159
8.2.7	Timer Output Compare Control Register (TOCR).....	161
8.3	CPU Interface .....	163
8.4	Operation .....	166
8.4.1	FRC Increment Timing.....	166
8.4.2	Output Compare Timing.....	168
8.4.3	FRC Clear Timing .....	169
8.4.4	Input Capture Timing .....	170
8.4.5	Timing of Input Capture Flag (ICF) Setting.....	173
8.4.6	Setting of Output Compare Flags A and B (OCFA and OCFB) .....	174
8.4.7	Setting of FRC Overflow Flag (OVF) .....	175
8.5	Interrupts .....	175
8.6	Sample Application .....	176
8.7	Application Notes .....	177
<b>Section 9 8-Bit Timers .....</b>		183
9.1	Overview .....	183
9.1.1	Features.....	183
9.1.2	Block Diagram.....	184
9.1.3	Input and Output Pins .....	185
9.1.4	Register Configuration.....	185

9.2	Register Descriptions.....	186
9.2.1	Timer Counter (TCNT).....	186
9.2.2	Time Constant Registers A and B (TCORA and TCORB).....	186
9.2.3	Timer Control Register (TCR).....	187
9.2.4	Timer Control/Status Register (TCSR) .....	190
9.2.5	Serial/Timer Control Register (STCR).....	192
9.3	Operation .....	193
9.3.1	TCNT Increment Timing.....	193
9.3.2	Compare-Match Timing .....	195
9.3.3	External Reset of TCNT .....	197
9.3.4	Setting of TCSR Overflow Flag (OVF).....	197
9.4	Interrupts .....	198
9.5	Sample Application .....	198
9.6	Application Notes .....	199
9.6.1	Contention between TCNT Write and Clear .....	199
9.6.2	Contention between TCNT Write and Increment .....	200
9.6.3	Contention between TCOR Write and Compare-Match .....	201
9.6.4	Contention between Compare-Match A and Compare-Match B .....	202
9.6.5	Increment Caused by Changing of Internal Clock Source .....	202
<b>Section 10      PWM Timers .....</b>		205
10.1	Overview .....	205
10.1.1	Features.....	205
10.1.2	Block Diagram.....	206
10.1.3	Input and Output Pins .....	207
10.1.4	Register Configuration.....	207
10.2	Register Descriptions.....	208
10.2.1	Timer Counter (TCNT).....	208
10.2.2	Duty Register (DTR) .....	208
10.2.3	Timer Control Register (TCR).....	209
10.3	Operation .....	211
10.3.1	Timer Increment .....	211
10.3.2	PWM Operation.....	212
10.4	Application Notes .....	213
<b>Section 11      Watchdog Timer .....</b>		215
11.1	Overview .....	215
11.1.1	Features.....	215
11.1.2	Block Diagram.....	216
11.1.3	Register Configuration.....	216
11.2	Register Descriptions.....	217
11.2.1	Timer Counter (TCNT).....	217
11.2.2	Timer Control/Status Register (TCSR) .....	217

11.2.3	Register Access.....	219
11.3	Operation .....	220
11.3.1	Watchdog Timer Mode.....	220
11.3.2	Interval Timer Mode.....	221
11.3.3	Setting the Overflow Flag.....	221
11.4	Application Notes .....	222
11.4.1	Contention between TCNT Write and Increment.....	222
11.4.2	Changing the Clock Select Bits (CKS2 to CKS0).....	222
11.4.3	Recovery from Software Standby Mode .....	222
<b>Section 12 Serial Communication Interface.....</b>		223
12.1	Overview .....	223
12.1.1	Features.....	223
12.1.2	Block Diagram.....	225
12.1.3	Input and Output Pins .....	226
12.1.4	Register Configuration.....	227
12.2	Register Descriptions.....	228
12.2.1	Receive Shift Register (RSR) .....	228
12.2.2	Receive Data Register (RDR).....	228
12.2.3	Transmit Shift Register (TSR).....	228
12.2.4	Transmit Data Register (TDR) .....	229
12.2.5	Serial Mode Register (SMR) .....	229
12.2.6	Serial Control Register (SCR) .....	232
12.2.7	Serial Status Register (SSR) .....	235
12.2.8	Bit Rate Register (BRR).....	238
12.2.9	Serial/Timer Control Register (STCR).....	243
12.3	Operation .....	244
12.3.1	Overview.....	244
12.3.2	Asynchronous Mode .....	246
12.3.3	Synchronous Mode .....	260
12.4	Interrupts .....	269
12.5	Application Notes .....	269
<b>Section 13 I<sup>2</sup>C Bus Interface [Option].....</b>		273
13.1	Overview .....	273
13.1.1	Features.....	273
13.1.2	Block Diagram.....	275
13.1.3	Input/Output Pins.....	276
13.1.4	Register Configuration.....	276
13.2	Register Descriptions.....	277
13.2.1	I <sup>2</sup> C Bus Data Register (ICDR) .....	277
13.2.2	Slave Address Register (SAR).....	277
13.2.3	I <sup>2</sup> C Bus Mode Register (ICMR).....	278

13.2.4	I <sup>2</sup> C Bus Control Register (ICCR).....	280
13.2.5	I <sup>2</sup> C Bus Status Register (ICSR).....	283
13.2.6	Serial/Timer Control Register (STCR).....	287
13.3	Operation .....	288
13.3.1	I <sup>2</sup> C Bus Data Format .....	288
13.3.2	Master Transmit Operation.....	290
13.3.3	Master Receive Operation .....	292
13.3.4	Slave Transmit Operation .....	294
13.3.5	Slave Receive Operation.....	296
13.3.6	IRIC Set Timing and SCL Control .....	297
13.3.7	Noise Canceler.....	298
13.3.8	Sample Flowcharts.....	299
13.4	Application Notes .....	303
<b>Section 14 Host Interface .....</b>		<b>305</b>
14.1	Overview .....	305
14.1.1	Block Diagram .....	306
14.1.2	Input and Output Pins .....	307
14.1.3	Register Configuration.....	308
14.2	Register Descriptions.....	309
14.2.1	System Control Register (SYSCR).....	309
14.2.2	Host Interface Control Register (HICR).....	309
14.2.3	Input Data Register (IDR1) .....	310
14.2.4	Output Data Register (ODR1) .....	311
14.2.5	Status Register (STR1) .....	311
14.2.6	Input Data Register (IDR2) .....	312
14.2.7	Output Data Register (ODR2) .....	313
14.2.8	Status Register (STR2) .....	313
14.2.9	Serial/Timer Control Register (STCR).....	315
14.3	Operation .....	316
14.3.1	Host Interface Operation.....	316
14.3.2	Control States.....	316
14.3.3	A <sub>20</sub> Gate.....	317
14.4	Interrupts .....	320
14.4.1	IBF1, IBF2.....	320
14.4.2	HIRQ <sub>11</sub> , HIRQ <sub>1</sub> , and HIRQ <sub>12</sub> .....	320
14.5	Application Note.....	321
<b>Section 15 A/D Converter .....</b>		<b>323</b>
15.1	Overview .....	323
15.1.1	Features.....	323
15.1.2	Block Diagram.....	324
15.1.3	Input Pins .....	325

15.1.4	Register Configuration.....	326
15.2	Register Descriptions.....	327
15.2.1	A/D Data Registers A to D (ADDRA to ADDRD).....	327
15.2.2	A/D Control/Status Register (ADCSR).....	328
15.2.3	A/D Control Register (ADCR) .....	330
15.3	CPU Interface .....	331
15.4	Operation .....	332
15.4.1	Single Mode (SCAN = 0) .....	332
15.4.2	Scan Mode (SCAN = 1).....	334
15.4.3	Input Sampling and A/D Conversion Time.....	336
15.4.4	External Trigger Input Timing.....	337
15.5	Interrupts .....	338
15.6	Application Notes .....	338
<b>Section 16 D/A Converter.....</b>		339
16.1	Overview .....	339
16.1.1	Features.....	339
16.1.2	Block Diagram.....	340
16.1.3	Input and Output Pins .....	341
16.1.4	Register Configuration.....	341
16.2	Register Descriptions.....	342
16.2.1	D/A Data Registers 0 and 1 (DADR0, DADR1) .....	342
16.2.2	D/A Control Register (DACR) .....	342
16.3	Operation .....	344
<b>Section 17 RAM .....</b>		345
17.1	Overview .....	345
17.1.1	Block Diagram.....	345
17.1.2	RAM Enable Bit (RAME) in System Control Register (SYSCR) .....	346
17.2	Operation .....	346
17.2.1	Expanded Modes (Modes 1 and 2) .....	346
17.2.2	Single-Chip Mode (Mode 3).....	346
<b>Section 18 ROM (Mask ROM version/ZTAT version).....</b>		347
18.1	Overview .....	347
18.1.1	Block Diagram.....	348
18.2	PROM Mode (H8/3437, H8/3434) .....	349
18.2.1	PROM Mode Setup.....	349
18.2.2	Socket Adapter Pin Assignments and Memory Map.....	349
18.3	PROM Programming .....	352
18.3.1	Programming and Verification .....	353
18.3.2	Notes on Programming .....	356
18.3.3	Reliability of Programmed Data.....	357

<b>Section 19 ROM (Flash memory 32 kbytes version) .....</b>	<b>359</b>
19.1 Flash Memory Overview .....	359
19.1.1 Flash Memory Operating Principle .....	359
19.1.2 Mode Programming and Flash Memory Address Space .....	359
19.1.3 Features.....	360
19.1.4 Block Diagram.....	361
19.1.5 Input/Output Pins.....	362
19.1.6 Register Configuration.....	362
19.2 Flash Memory Register Descriptions .....	363
19.2.1 Flash Memory Control Register (FLMCR) .....	363
19.2.2 Erase Block Register 1 (EBR1) .....	364
19.2.3 Erase Block Register 2 (EBR2) .....	365
19.2.4 Wait-State Control Register (WSCR).....	366
19.3 On-Board Programming Modes .....	369
19.3.1 Boot Mode .....	370
19.3.2 User Program Mode.....	376
19.4 Programming and Erasing Flash Memory .....	378
19.4.1 Program Mode .....	378
19.4.2 Program-Verify Mode .....	378
19.4.3 Programming Flowchart and Sample Program.....	379
19.4.4 Erase Mode .....	381
19.4.5 Erase-Verify Mode .....	381
19.4.6 Erasing Flowchart and Sample Program .....	382
19.4.7 Prewrite Verify Mode .....	395
19.4.8 Protect Modes .....	395
19.4.9 Interrupt Handling during Flash Memory Programming and Erasing.....	396
19.5 Flash Memory Emulation by RAM .....	398
19.6 Flash Memory PROM Mode (H8/3434F) .....	401
19.6.1 PROM Mode Setting .....	401
19.6.2 Socket Adapter and Memory Map.....	401
19.6.3 Operation in PROM Mode.....	403
19.7 Flash Memory Programming and Erasing Precautions .....	411
<b>Section 20 ROM (Flash memory 60 kbytes version) .....</b>	<b>419</b>
20.1 Flash Memory Overview .....	419
20.1.1 Flash Memory Operating Principle .....	419
20.1.2 Mode Programming and Flash Memory Address Space .....	419
20.1.3 Features.....	420
20.1.4 Block Diagram.....	421
20.1.5 Input/Output Pins.....	422
20.1.6 Register Configuration.....	422
20.2 Flash Memory Register Descriptions .....	423
20.2.1 Flash Memory Control Register (FLMCR) .....	423

20.2.2	Erase Block Register 1 (EBR1) .....	424
20.2.3	Erase Block Register 2 (EBR2) .....	425
20.2.4	Wait-State Control Register (WSCR).....	426
20.3	On-Board Programming Modes .....	429
20.3.1	Boot Mode .....	430
20.3.2	User Program Mode.....	436
20.4	Programming and Erasing Flash Memory.....	438
20.4.1	Program Mode .....	438
20.4.2	Program-Verify Mode .....	438
20.4.3	Programming Flowchart and Sample Program.....	439
20.4.4	Erase Mode .....	441
20.4.5	Erase-Verify Mode .....	441
20.4.6	Erasing Flowchart and Sample Program .....	442
20.4.7	Prewrite Verify Mode.....	455
20.4.8	Protect Modes .....	455
20.4.9	Interrupt Handling during Flash Memory Programming and Erasing.....	456
20.5	Flash Memory Emulation by RAM .....	458
20.6	Flash Memory PROM Mode (H8/3437F) .....	461
20.6.1	PROM Mode Setting .....	461
20.6.2	Socket Adapter and Memory Map.....	461
20.6.3	Operation in PROM Mode.....	463
20.7	Flash Memory Programming and Erasing Precautions .....	471
<b>Section 21 Power-Down State .....</b> 479		
21.1	Overview .....	479
21.1.1	System Control Register (SYSCR).....	480
21.2	Sleep Mode .....	482
21.2.1	Transition to Sleep Mode.....	482
21.2.2	Exit from Sleep Mode.....	482
21.3	Software Standby Mode .....	483
21.3.1	Transition to Software Standby Mode .....	483
21.3.2	Exit from Software Standby Mode .....	483
21.3.3	Clock Settling Time for Exit from Software Standby Mode .....	484
21.3.4	Sample Application of Software Standby Mode .....	485
21.3.5	Application Note.....	485
21.4	Hardware Standby Mode .....	487
21.4.1	Transition to Hardware Standby Mode.....	487
21.4.2	Recovery from Hardware Standby Mode .....	487
21.4.3	Timing Relationships.....	488
<b>Section 22 Electrical Specifications.....</b> 489		
22.1	Absolute Maximum Ratings .....	489
22.2	Electrical Characteristics .....	490

22.2.1	DC Characteristics .....	490
22.2.2	AC Characteristics .....	502
22.2.3	A/D Converter Characteristics.....	507
22.2.4	D/A Converter Characteristics.....	508
22.3	MCU Operational Timing.....	509
22.3.1	Bus Timing .....	509
22.3.2	Control Signal Timing .....	511
22.3.3	16-Bit Free-Running Timer Timing .....	513
22.3.4	8-Bit Timer Timing.....	514
22.3.5	Pulse Width Modulation Timer Timing .....	515
22.3.6	Serial Communication Interface Timing .....	516
22.3.7	I/O Port Timing.....	517
22.3.8	Host Interface Timing.....	518
22.3.9	I <sup>2</sup> C Bus Timing (Option).....	519
22.3.10	Reset Output Timing.....	519
22.3.11	External Clock Output Timing .....	520
Appendix A CPU Instruction Set.....		521
A.1	Instruction Set List.....	521
A.2	Operation Code Map.....	529
A.3	Number of States Required for Execution.....	531
Appendix B Internal I/O Register.....		537
B.1	Addresses .....	537
B.2	Function .....	542
Appendix C I/O Port Block Diagrams .....		597
C.1	Port 1 Block Diagram .....	597
C.2	Port 2 Block Diagram .....	598
C.3	Port 3 Block Diagram .....	599
C.4	Port 4 Block Diagrams.....	600
C.5	Port 5 Block Diagrams.....	604
C.6	Port 6 Block Diagrams.....	607
C.7	Port 7 Block Diagrams.....	611
C.8	Port 8 Block Diagrams.....	612
C.9	Port 9 Block Diagrams.....	618
C.10	Port A Block Diagram .....	624
C.11	Port B Block Diagram .....	625
Appendix D Pin States .....		626
D.1	Port States in Each Mode.....	626

Appendix E	Timing of Transition to and Recovery from Hardware Standby Mode.....	629
Appendix F	Option Lists .....	630
Appendix G	Product Code Lineup .....	632
Appendix H	Package Dimensions .....	633

# Section 1 Overview

## 1.1 Overview

The H8/3437 Series of single-chip microcomputers features an H8/300 CPU core and a complement of on-chip supporting modules implementing a variety of system functions.

The H8/300 CPU is a high-speed processor with an architecture featuring powerful bit-manipulation instructions, ideally suited for realtime control applications. The on-chip supporting modules implement peripheral functions needed in system configurations. These include ROM, RAM, four types of timers (a 16-bit free-running timer, 8-bit timers, PWM timers, and a watchdog timer), a serial communication interface (SCI), an I<sup>2</sup>C bus interface [option], a host interface (HIF), an A/D converter, a D/A converter, and I/O ports.

The H8/3437 Series can operate in single-chip mode or in two expanded modes, depending on the requirements of the application.

Besides the mask-ROM versions of the H8/3437 Series, there are ZTAT<sup>TM</sup><sup>\*1</sup> versions with on-chip PROM, and an F-ZTAT<sup>TM</sup><sup>\*2</sup> version with on-chip flash memory. The F-ZTAT<sup>TM</sup> version can be programmed or reprogrammed on-board in application systems.

Notes: 1. ZTAT<sup>TM</sup> (zero turn-around time) is a trademark of Hitachi, Ltd.  
2. F-ZTAT<sup>TM</sup> (flexible-ZTAT) is a trademark of Hitachi, Ltd.

Table 1-1 lists the features of the H8/3437 Series.

**Table 1-1 Features**

Item	Specification
CPU	<p>Two-way general register configuration</p> <ul style="list-style-type: none"><li>Eight 16-bit registers, or</li><li>Sixteen 8-bit registers</li></ul> <p>High-speed operation</p> <ul style="list-style-type: none"><li>Maximum clock rate (<math>\phi</math> clock): 16 MHz at 5 V, 12 MHz at 4 V or 10 MHz at 3 V</li><li>8- or 16-bit register-register add/subtract: 125 ns (16 MHz), 167 ns (12 MHz), 200 ns (10 MHz)</li><li><math>8 \times 8</math>-bit multiply: 875 ns (16 MHz), 1167 ns (12 MHz), 1400 ns (10 MHz)</li><li><math>16 \div 8</math>-bit divide: 875 ns (16 MHz), 1167 ns (12 MHz), 1400 ns (10 MHz)</li></ul> <p>Streamlined, concise instruction set</p> <ul style="list-style-type: none"><li>Instruction length: 2 or 4 bytes</li><li>Register-register arithmetic and logic operations</li><li>MOV instruction for data transfer between registers and memory</li></ul> <p>Instruction set features</p> <ul style="list-style-type: none"><li>Multiply instruction (8 bits <math>\times</math> 8 bits)</li><li>Divide instruction (16 bits <math>\div</math> 8 bits)</li><li>Bit-accumulator instructions</li><li>Register-indirect specification of bit positions</li></ul>
Memory	<ul style="list-style-type: none"><li>H8/3437: 60-kbyte ROM; 2-kbyte RAM</li><li>H8/3436: 48-kbyte ROM; 2-kbyte RAM</li><li>H8/3434: 32-kbyte ROM; 1-kbyte RAM</li></ul>
16-bit free-running timer (1 channel)	<ul style="list-style-type: none"><li>One 16-bit free-running counter (can also count external events)</li><li>Two output-compare lines</li><li>Four input capture lines (can be buffered)</li></ul>
8-bit timer (2 channels)	<p>Each channel has</p> <ul style="list-style-type: none"><li>One 8-bit up-counter (can also count external events)</li><li>Two time constant registers</li></ul>
PWM timer (2 channels)	<ul style="list-style-type: none"><li>Duty cycle can be set from 0 to 100%</li><li>Resolution: 1/250</li></ul>
Watchdog timer (WDT) (1 channel)	<ul style="list-style-type: none"><li>Overflow can generate a reset or NMI interrupt</li><li>Also usable as interval timer</li></ul>
Serial communication interface (SCI) (2 channels)	<ul style="list-style-type: none"><li>Asynchronous or synchronous mode (selectable)</li><li>Full duplex: can transmit and receive simultaneously</li><li>On-chip baud rate generator</li></ul>

**Table 1-1 Features (cont)**

Item	Specification
I <sup>2</sup> C bus interface (1 channel) [option]	<ul style="list-style-type: none"><li>Conforms to Philips I<sup>2</sup>C bus interface</li><li>Includes single master mode and slave mode</li></ul>
Host interface (HIF)	<ul style="list-style-type: none"><li>8-bit host interface port</li><li>Three host interrupt requests (HIRQ<sub>1</sub>, HIRQ<sub>11</sub>, HIRQ<sub>12</sub>)</li><li>Regular and fast A<sub>20</sub> gate output</li><li>Two register sets, each with two data registers and a status register</li></ul>
Keyboard controller	<ul style="list-style-type: none"><li>Controls a matrix-scan keyboard by providing a keyboard scan function with wake-up interrupts and sense ports</li></ul>
A/D converter	<ul style="list-style-type: none"><li>10-bit resolution</li><li>Eight channels: single or scan mode (selectable)</li><li>Start of A/D conversion can be externally triggered</li><li>Sample-and-hold function</li></ul>
D/A converter	<ul style="list-style-type: none"><li>8-bit resolution</li><li>Two channels</li></ul>
I/O ports	<ul style="list-style-type: none"><li>74 input/output lines (16 of which can drive LEDs)</li><li>8 input-only lines</li></ul>
Interrupts	<ul style="list-style-type: none"><li>Nine external interrupt lines: <math>\overline{\text{NMI}}</math>, <math>\overline{\text{IRQ}_0}</math> to <math>\overline{\text{IRQ}_7}</math></li><li>26 on-chip interrupt sources</li></ul>
Wait control	<ul style="list-style-type: none"><li>Three selectable wait modes</li></ul>
Operating modes	<ul style="list-style-type: none"><li>Expanded mode with on-chip ROM disabled (mode 1)</li><li>Expanded mode with on-chip ROM enabled (mode 2)</li><li>Single-chip mode (mode 3)</li></ul>
Power-down modes	<ul style="list-style-type: none"><li>Sleep mode</li><li>Software standby mode</li><li>Hardware standby mode</li></ul>
Other features	<ul style="list-style-type: none"><li>On-chip oscillator</li></ul>

**Table 1-1 Features (cont)**

Item	Specification				
Series lineup	Part Number				
	Product Name	5-V Version (16 MHz) 4-V Version (12 MHz)	3-V Version (10 MHz)	Package	ROM
H8/3437 F-ZTAT	HD64F3437F16	HD64F3437F16	100-pin QFP (FP-100B)	Flash memory	
	HD64F3437TF16	HD64F3437TF16	100-pin TQFP (TFP-100B)		
H8/3437 ZTAT	HD6473437F16	HD6473437F16	100-pin QFP (FP-100B)	PROM	
	HD6473437TF16	HD6473437TF16	100-pin TQFP (TFP-100B)		
H8/3437	HD6433437F16	HD6433437VF10	100-pin QFP (FP-100B)	Mask ROM	
	HD6433437F12	HD6433437TF16	HD6433437VTF10	100-pin TQFP (TFP-100B)	
H8/3436	HD6433436F16	HD6433436VF10	100-pin QFP (FP-100B)	Mask ROM	
	HD6433436F12	HD6433436TF16	HD6433436VTF10	100-pin TQFP (TFP-100B)	
H8/3434 F-ZTAT	HD64F3434F16	HD64F3434F16	100-pin QFP (FP-100B)	Flash memory	
	HD64F3434TF16	HD64F3434TF16	100-pin TQFP (TFP-100B)		
H8/3434 ZTAT	HD6473434F16	HD6473434F16	100-pin QFP (FP-100B)	PROM	
	HD6473434TF16	HD6473434TF16	100-pin TQFP (TFP-100B)		
H8/3434	HD6433434F16	HD6433434VF10	100-pin QFP (FP-100B)	Mask ROM	
	HD6433434F12	HD6433434TF16	HD6433434VTF10	100-pin TQFP (TFP-100B)	

The I<sup>2</sup>C bus interface is an available option. Please note the following points regarding this option.

1. Contact your local Hitachi representative to order the I<sup>2</sup>C bus interface.
2. In mask ROM versions, chips featuring the I<sup>2</sup>C bus interface include a W in the part number.  
Example: HD6433437WTF, HD6433434WF, etc.
3. Although ZTAT and F-ZTAT chips have identical part numbers, inform your local Hitachi representative when ordering the I<sup>2</sup>C bus interface.

## 1.2 Block Diagram

Figure 1-1 shows a block diagram of the H8/3437 Series.

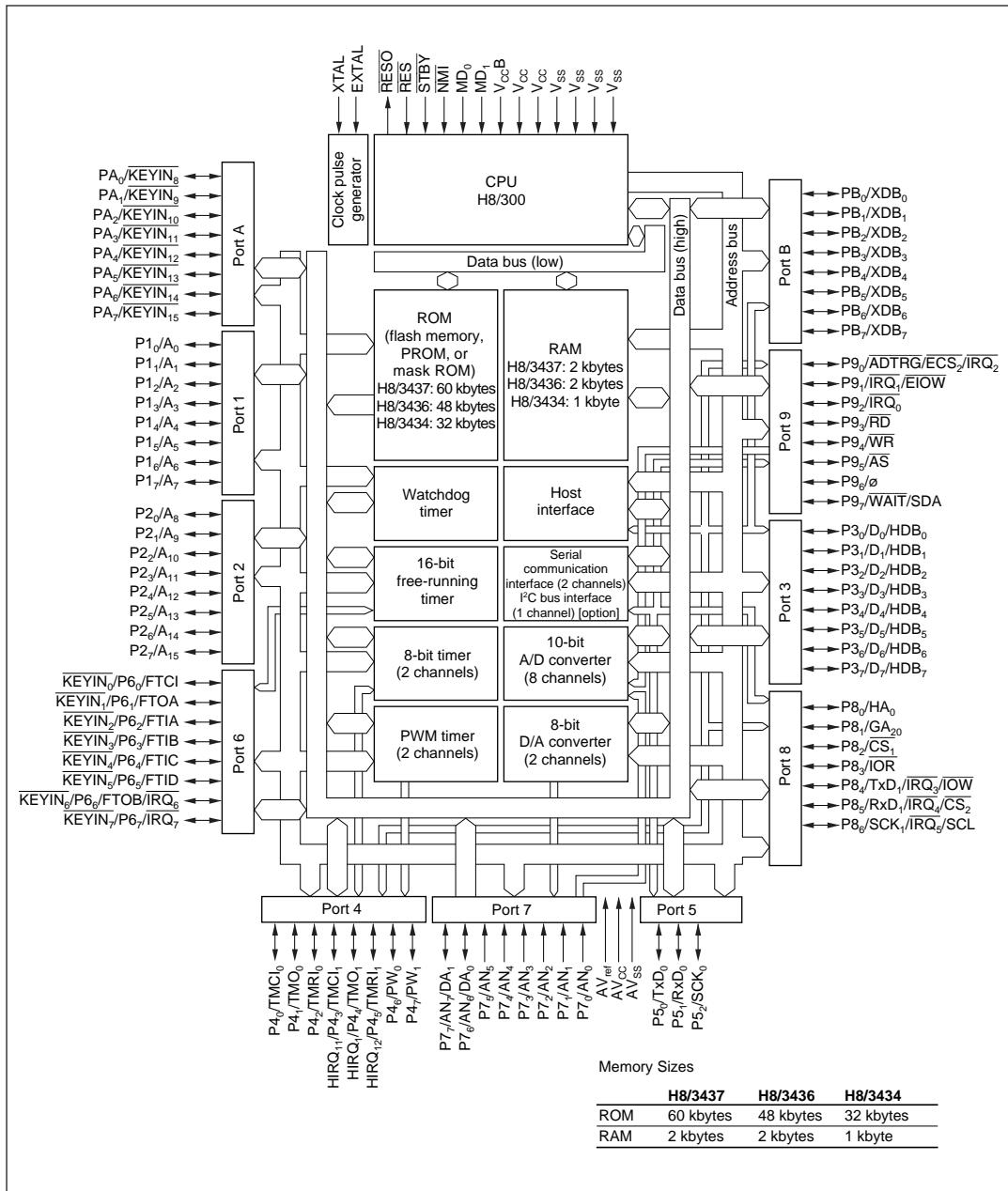


Figure 1-1 Block Diagram

# 1.3 Pin Assignments and Functions

## 1.3.1 Pin Arrangement

Figure 1-2 shows the pin arrangement of the FP-100B and TFP-100B packages.

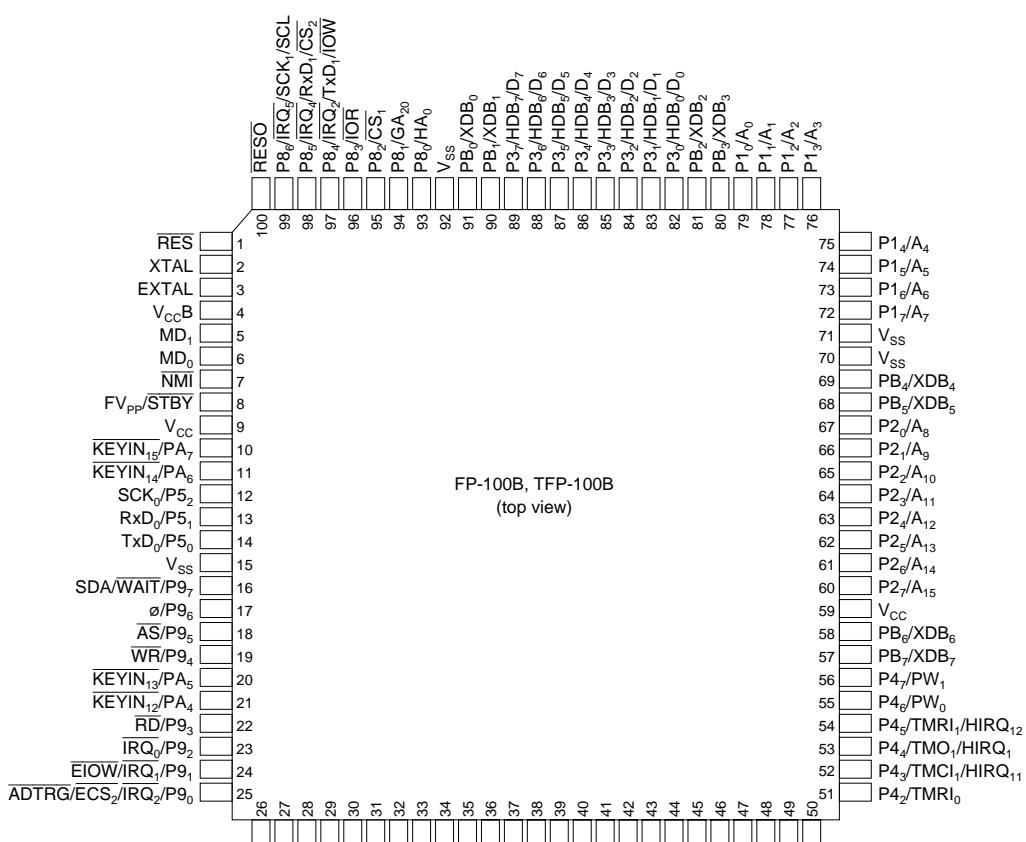


Figure 1-2 Pin Arrangement (FP-100B, TFP-100B, Top View)

### 1.3.2 Pin Functions

**(1) Pin Assignments in Each Operating Mode:** Table 1-2 lists the assignments of the pins of the FP-100B and TFP-100B packages in each operating mode.

**Table 1-2 Pin Assignments in Each Operating Mode**

Pin No.	Expanded Modes		Single-Chip Mode		EPROM PROM Mode	Flash Memory PROM Mode		
			Mode 3					
	Mode 1	Mode 2	HIF Disabled	HIF Enabled				
1	RES	RES	RES	RES	V <sub>PP</sub>	RES		
2	XTAL	XTAL	XTAL	XTAL	NC	XTAL		
3	EXTAL	EXTAL	EXTAL	EXTAL	NC	EXTAL		
4	V <sub>CCB</sub>	V <sub>CCB</sub>	V <sub>CCB</sub>	V <sub>CCB</sub>	V <sub>CC</sub>	V <sub>CC</sub>		
5	MD <sub>1</sub>	MD <sub>1</sub>	MD <sub>1</sub>	MD <sub>1</sub>	V <sub>SS</sub>	V <sub>SS</sub>		
6	MD <sub>0</sub>	MD <sub>0</sub>	MD <sub>0</sub>	MD <sub>0</sub>	V <sub>SS</sub>	V <sub>SS</sub>		
7	NMI	NMI	NMI	NMI	EA <sub>9</sub>	FA <sub>9</sub>		
8	STBY	STBY/FV <sub>PP</sub>	STBY/FV <sub>PP</sub>	STBY/FV <sub>PP</sub>	V <sub>SS</sub>	FV <sub>PP</sub>		
9	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>		
10	PA <sub>7</sub> /KEYIN <sub>15</sub>	NC	NC					
11	PA <sub>6</sub> /KEYIN <sub>14</sub>	NC	NC					
12	P5 <sub>2</sub> /SCK <sub>0</sub>	NC	NC					
13	P5 <sub>1</sub> /Rx <sub>D0</sub>	NC	NC					
14	P5 <sub>0</sub> /Tx <sub>D0</sub>	NC	NC					
15	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>		
16	P9 <sub>7</sub> /WAIT/SDA	P9 <sub>7</sub> /WAIT/SDA	P9 <sub>7</sub> /SDA	P9 <sub>7</sub> /SDA	NC	V <sub>CC</sub>		
17	Ø	Ø	P9 <sub>6</sub> /Ø	P9 <sub>6</sub> /Ø	NC	NC		
18	AS	AS	P9 <sub>5</sub>	P9 <sub>5</sub>	NC	FA <sub>16</sub>		
19	WR	WR	P9 <sub>4</sub>	P9 <sub>4</sub>	NC	FA <sub>15</sub>		
20	PA <sub>5</sub> /KEYIN <sub>13</sub>	NC	NC					
21	PA <sub>4</sub> /KEYIN <sub>12</sub>	NC	NC					
22	RD	RD	P9 <sub>3</sub>	P9 <sub>3</sub>	NC	WE		
23	P9 <sub>2</sub> /IRQ <sub>0</sub>	PGM	V <sub>SS</sub>					

Note: Pins marked NC should be left unconnected.

For details on PROM mode, refer to 18.2, PROM Mode, 19.6, Flash Memory PROM Mode (H8/3434F) and 20.6, Flash Memory PROM Mode (H8/3437F).

In this chip, the same pin is used for STBY and FV<sub>PP</sub>. When this pin is driven low, a transition is made to hardware standby mode. This occurs not only in the normal operating modes (modes 1, 2, and 3), but also when programming flash memory with a PROM writer.

Therefore, a PROM writer should be used whose specifications provide for this pin to be held at the V<sub>CC</sub> level except when programming (FV<sub>PP</sub> = 12 V).

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

Pin No.	Expanded Modes		Single-Chip Mode		EPROM PROM Mode	Flash Memory PROM Mode		
			Mode 3					
	Mode 1	Mode 2	HIF Disabled	HIF Enabled				
24	P9 <sub>1</sub> /IRQ <sub>1</sub> when HIF is disabled or STAC bit is 0 in STCR; EIOW/IRQ <sub>1</sub> when HIF is enabled and STAC bit is 1 in STCR				EA <sub>15</sub>	V <sub>CC</sub>		
25	P9 <sub>0</sub> /IRQ <sub>2</sub> /ADTRG when HIF is disabled or STAC bit is 0 in STCR; ECS <sub>2</sub> /IRQ <sub>2</sub> when HIF is enabled and STAC bit is 1 in STCR				EA <sub>16</sub>	V <sub>CC</sub>		
26	P6 <sub>0</sub> /FTCI/ KEYIN <sub>0</sub>	P6 <sub>0</sub> /FTCI/ KEYIN <sub>0</sub>	P6 <sub>0</sub> /FTCI/ KEYIN <sub>0</sub>	P6 <sub>0</sub> /FTCI/ KEYIN <sub>0</sub>	NC	NC		
27	P6 <sub>1</sub> /FTOA/ KEYIN <sub>1</sub>	P6 <sub>1</sub> /FTOA/ KEYIN <sub>1</sub>	P6 <sub>1</sub> /FTOA/ KEYIN <sub>1</sub>	P6 <sub>1</sub> /FTOA/ KEYIN <sub>1</sub>	NC	NC		
28	P6 <sub>2</sub> /FTIA/ KEYIN <sub>2</sub>	P6 <sub>2</sub> /FTIA/ KEYIN <sub>2</sub>	P6 <sub>2</sub> /FTIA/ KEYIN <sub>2</sub>	P6 <sub>2</sub> /FTIA/ KEYIN <sub>2</sub>	NC	NC		
29	P6 <sub>3</sub> /FTIB/ KEYIN <sub>3</sub>	P6 <sub>3</sub> /FTIB/ KEYIN <sub>3</sub>	P6 <sub>3</sub> /FTIB/ KEYIN <sub>3</sub>	P6 <sub>3</sub> /FTIB/ KEYIN <sub>3</sub>	V <sub>CC</sub>	V <sub>CC</sub>		
30	PA <sub>3</sub> /KEYIN <sub>11</sub>	PA <sub>3</sub> /KEYIN <sub>11</sub>	PA <sub>3</sub> /KEYIN <sub>11</sub>	PA <sub>3</sub> /KEYIN <sub>11</sub>	NC	NC		
31	PA <sub>2</sub> /KEYIN <sub>10</sub>	PA <sub>2</sub> /KEYIN <sub>10</sub>	PA <sub>2</sub> /KEYIN <sub>10</sub>	PA <sub>2</sub> /KEYIN <sub>10</sub>	NC	NC		
32	P6 <sub>4</sub> /FTIC/ KEYIN <sub>4</sub>	P6 <sub>4</sub> /FTIC/ KEYIN <sub>4</sub>	P6 <sub>4</sub> /FTIC/ KEYIN <sub>4</sub>	P6 <sub>4</sub> /FTIC/ KEYIN <sub>4</sub>	V <sub>CC</sub>	V <sub>CC</sub>		
33	P6 <sub>5</sub> /FTID/ KEYIN <sub>5</sub>	P6 <sub>5</sub> /FTID/ KEYIN <sub>5</sub>	P6 <sub>5</sub> /FTID/ KEYIN <sub>5</sub>	P6 <sub>5</sub> /FTID/ KEYIN <sub>5</sub>	NC	NC		
34	P6 <sub>6</sub> /FTOB/ IRQ <sub>6</sub> /KEYIN <sub>6</sub>	P6 <sub>6</sub> /FTOB/ IRQ <sub>6</sub> /KEYIN <sub>6</sub>	P6 <sub>6</sub> /FTOB/ IRQ <sub>6</sub> /KEYIN <sub>6</sub>	P6 <sub>6</sub> /FTOB/ IRQ <sub>6</sub> /KEYIN <sub>6</sub>	NC	NC		
35	P6 <sub>7</sub> /IRQ <sub>7</sub> / KEYIN <sub>7</sub>	P6 <sub>7</sub> /IRQ <sub>7</sub> / KEYIN <sub>7</sub>	P6 <sub>7</sub> /IRQ <sub>7</sub> / KEYIN <sub>7</sub>	P6 <sub>7</sub> /IRQ <sub>7</sub> / KEYIN <sub>7</sub>	NC	V <sub>SS</sub>		
36	AV <sub>ref</sub>	AV <sub>ref</sub>	AV <sub>ref</sub>	AV <sub>ref</sub>	V <sub>CC</sub>	V <sub>SS</sub>		
37	AV <sub>CC</sub>	AV <sub>CC</sub>	AV <sub>CC</sub>	AV <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>		
38	P7 <sub>0</sub> /AN <sub>0</sub>	P7 <sub>0</sub> /AN <sub>0</sub>	P7 <sub>0</sub> /AN <sub>0</sub>	P7 <sub>0</sub> /AN <sub>0</sub>	NC	NC		
39	P7 <sub>1</sub> /AN <sub>1</sub>	P7 <sub>1</sub> /AN <sub>1</sub>	P7 <sub>1</sub> /AN <sub>1</sub>	P7 <sub>1</sub> /AN <sub>1</sub>	NC	NC		
40	P7 <sub>2</sub> /AN <sub>2</sub>	P7 <sub>2</sub> /AN <sub>2</sub>	P7 <sub>2</sub> /AN <sub>2</sub>	P7 <sub>2</sub> /AN <sub>2</sub>	NC	NC		
41	P7 <sub>3</sub> /AN <sub>3</sub>	P7 <sub>3</sub> /AN <sub>3</sub>	P7 <sub>3</sub> /AN <sub>3</sub>	P7 <sub>3</sub> /AN <sub>3</sub>	NC	NC		
42	P7 <sub>4</sub> /AN <sub>4</sub>	P7 <sub>4</sub> /AN <sub>4</sub>	P7 <sub>4</sub> /AN <sub>4</sub>	P7 <sub>4</sub> /AN <sub>4</sub>	NC	NC		
43	P7 <sub>5</sub> /AN <sub>5</sub>	P7 <sub>5</sub> /AN <sub>5</sub>	P7 <sub>5</sub> /AN <sub>5</sub>	P7 <sub>5</sub> /AN <sub>5</sub>	NC	NC		
44	P7 <sub>6</sub> /AN <sub>6</sub> /DA <sub>0</sub>	P7 <sub>6</sub> /AN <sub>6</sub> /DA <sub>0</sub>	P7 <sub>6</sub> /AN <sub>6</sub> /DA <sub>0</sub>	P7 <sub>6</sub> /AN <sub>6</sub> /DA <sub>0</sub>	NC	NC		

Note: Pins marked NC should be left unconnected.

For details on PROM mode, refer to 18.2, PROM Mode, 19.6, Flash Memory PROM Mode (H8/3434F) and 20.6, Flash Memory PROM Mode (H8/3437F).

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

Pin No.	Expanded Modes		Single-Chip Mode		EPROM PROM Mode	Flash Memory PROM Mode		
			Mode 3					
	Mode 1	Mode 2	HIF Disabled	HIF Enabled				
45	P7 <sub>7</sub> /AN <sub>7</sub> /DA <sub>1</sub>	P7 <sub>7</sub> /AN <sub>7</sub> /DA <sub>1</sub>	P7 <sub>7</sub> /AN <sub>7</sub> /DA <sub>1</sub>	P7 <sub>7</sub> /AN <sub>7</sub> /DA <sub>1</sub>	NC	NC		
46	AV <sub>SS</sub>	AV <sub>SS</sub>	AV <sub>SS</sub>	AV <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>		
47	PA <sub>1</sub> /KEYIN <sub>9</sub>	PA <sub>1</sub> /KEYIN <sub>9</sub>	PA <sub>1</sub> /KEYIN <sub>9</sub>	PA <sub>1</sub> /KEYIN <sub>9</sub>	NC	NC		
48	PA <sub>0</sub> /KEYIN <sub>8</sub>	PA <sub>0</sub> /KEYIN <sub>8</sub>	PA <sub>0</sub> /KEYIN <sub>8</sub>	PA <sub>0</sub> /KEYIN <sub>8</sub>	NC	NC		
49	P4 <sub>0</sub> /TMCI <sub>0</sub>	P4 <sub>0</sub> /TMCI <sub>0</sub>	P4 <sub>0</sub> /TMCI <sub>0</sub>	P4 <sub>0</sub> /TMCI <sub>0</sub>	NC	NC		
50	P4 <sub>1</sub> /TMO <sub>0</sub>	P4 <sub>1</sub> /TMO <sub>0</sub>	P4 <sub>1</sub> /TMO <sub>0</sub>	P4 <sub>1</sub> /TMO <sub>0</sub>	NC	NC		
51	P4 <sub>2</sub> /TMRI <sub>0</sub>	P4 <sub>2</sub> /TMRI <sub>0</sub>	P4 <sub>2</sub> /TMRI <sub>0</sub>	P4 <sub>2</sub> /TMRI <sub>0</sub>	NC	NC		
52	P4 <sub>3</sub> /TMCI <sub>1</sub> /HIRQ <sub>11</sub> *	P4 <sub>3</sub> /TMCI <sub>1</sub> /HIRQ <sub>11</sub> *	P4 <sub>3</sub> /TMCI <sub>1</sub>	HIRQ <sub>11</sub> /TMCI <sub>1</sub>	NC	NC		
53	P4 <sub>4</sub> /TMO <sub>1</sub> /HIRQ <sub>1</sub> *	P4 <sub>4</sub> /TMO <sub>1</sub> /HIRQ <sub>1</sub> *	P4 <sub>4</sub> /TMO <sub>1</sub>	HIRQ <sub>1</sub> /TMO <sub>1</sub>	NC	NC		
54	P4 <sub>5</sub> /TMRI <sub>1</sub> /HIRQ <sub>12</sub> *	P4 <sub>5</sub> /TMRI <sub>1</sub> /HIRQ <sub>12</sub> *	P4 <sub>5</sub> /TMRI <sub>1</sub>	HIRQ <sub>12</sub> /TMRI <sub>1</sub>	NC	NC		
55	P4 <sub>6</sub> /PW <sub>0</sub>	P4 <sub>6</sub> /PW <sub>0</sub>	P4 <sub>6</sub> /PW <sub>0</sub>	P4 <sub>6</sub> /PW <sub>0</sub>	NC	NC		
56	P4 <sub>7</sub> /PW <sub>1</sub>	P4 <sub>7</sub> /PW <sub>1</sub>	P4 <sub>7</sub> /PW <sub>1</sub>	P4 <sub>7</sub> /PW <sub>1</sub>	NC	NC		
57	PB <sub>7</sub> /XDB <sub>7</sub> **	PB <sub>7</sub> /XDB <sub>7</sub> **	PB <sub>7</sub>	PB <sub>7</sub>	NC	NC		
58	PB <sub>6</sub> /XDB <sub>6</sub> **	PB <sub>6</sub> /XDB <sub>6</sub> **	PB <sub>6</sub>	PB <sub>6</sub>	NC	NC		
59	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>	V <sub>CC</sub>		
60	A <sub>15</sub>	P2 <sub>7</sub> /A <sub>15</sub>	P2 <sub>7</sub>	P2 <sub>7</sub>	CE	CE		
61	A <sub>14</sub>	P2 <sub>6</sub> /A <sub>14</sub>	P2 <sub>6</sub>	P2 <sub>6</sub>	EA <sub>14</sub>	FA <sub>14</sub>		
62	A <sub>13</sub>	P2 <sub>5</sub> /A <sub>13</sub>	P2 <sub>5</sub>	P2 <sub>5</sub>	EA <sub>13</sub>	FA <sub>13</sub>		
63	A <sub>12</sub>	P2 <sub>4</sub> /A <sub>12</sub>	P2 <sub>4</sub>	P2 <sub>4</sub>	EA <sub>12</sub>	FA <sub>12</sub>		
64	A <sub>11</sub>	P2 <sub>3</sub> /A <sub>11</sub>	P2 <sub>3</sub>	P2 <sub>3</sub>	EA <sub>11</sub>	FA <sub>11</sub>		
65	A <sub>10</sub>	P2 <sub>2</sub> /A <sub>10</sub>	P2 <sub>2</sub>	P2 <sub>2</sub>	EA <sub>10</sub>	FA <sub>10</sub>		
66	A <sub>9</sub>	P2 <sub>1</sub> /A <sub>9</sub>	P2 <sub>1</sub>	P2 <sub>1</sub>	OE	OE		
67	A <sub>8</sub>	P2 <sub>0</sub> /A <sub>8</sub>	P2 <sub>0</sub>	P2 <sub>0</sub>	EA <sub>8</sub>	FA <sub>8</sub>		
68	PB <sub>5</sub> /XDB <sub>5</sub> **	PB <sub>5</sub> /XDB <sub>5</sub> **	PB <sub>5</sub>	PB <sub>5</sub>	NC	NC		
69	PB <sub>4</sub> /XDB <sub>4</sub> **	PB <sub>4</sub> /XDB <sub>4</sub> **	PB <sub>4</sub>	PB <sub>4</sub>	NC	NC		
70	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>		

Note: Pins marked NC should be left unconnected.

For details on PROM mode, refer to 18.2, PROM Mode, 19.6, Flash Memory PROM Mode (H8/3434F), and 20.6, Flash Memory PROM Mode (H8/3437F).

\* Differs as in mode 3, depending on whether the host interface is enabled or disabled.

\*\* XDB<sub>7</sub> to XDB<sub>6</sub> can only be used when the host interface is enabled.

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

Pin No.	Expanded Modes		Single-Chip Mode		EPROM PROM Mode	Flash Memory PROM Mode		
			Mode 3					
	Mode 1	Mode 2	HIF Disabled	HIF Enabled				
71	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>		
72	A <sub>7</sub>	P1 <sub>7</sub> /A <sub>7</sub>	P1 <sub>7</sub>	P1 <sub>7</sub>	EA <sub>7</sub>	FA <sub>7</sub>		
73	A <sub>6</sub>	P1 <sub>6</sub> /A <sub>6</sub>	P1 <sub>6</sub>	P1 <sub>6</sub>	EA <sub>6</sub>	FA <sub>6</sub>		
74	A <sub>5</sub>	P1 <sub>5</sub> /A <sub>5</sub>	P1 <sub>5</sub>	P1 <sub>5</sub>	EA <sub>5</sub>	FA <sub>5</sub>		
75	A <sub>4</sub>	P1 <sub>4</sub> /A <sub>4</sub>	P1 <sub>4</sub>	P1 <sub>4</sub>	EA <sub>4</sub>	FA <sub>4</sub>		
76	A <sub>3</sub>	P1 <sub>3</sub> /A <sub>3</sub>	P1 <sub>3</sub>	P1 <sub>3</sub>	EA <sub>3</sub>	FA <sub>3</sub>		
77	A <sub>2</sub>	P1 <sub>2</sub> /A <sub>2</sub>	P1 <sub>2</sub>	P1 <sub>2</sub>	EA <sub>2</sub>	FA <sub>2</sub>		
78	A <sub>1</sub>	P1 <sub>1</sub> /A <sub>1</sub>	P1 <sub>1</sub>	P1 <sub>1</sub>	EA <sub>1</sub>	FA <sub>1</sub>		
79	A <sub>0</sub>	P1 <sub>0</sub> /A <sub>0</sub>	P1 <sub>0</sub>	P1 <sub>0</sub>	EA <sub>0</sub>	FA <sub>0</sub>		
80	PB <sub>3</sub> /XDB <sub>3</sub> **	PB <sub>3</sub> /XDB <sub>3</sub> **	PB <sub>3</sub>	PB <sub>3</sub>	NC	NC		
81	PB <sub>2</sub> /XDB <sub>2</sub> **	PB <sub>2</sub> /XDB <sub>2</sub> **	PB <sub>2</sub>	PB <sub>2</sub>	NC	NC		
82	D <sub>0</sub>	D <sub>0</sub>	P3 <sub>0</sub>	HDB <sub>0</sub>	EO <sub>0</sub>	FO <sub>0</sub>		
83	D <sub>1</sub>	D <sub>1</sub>	P3 <sub>1</sub>	HDB <sub>1</sub>	EO <sub>1</sub>	FO <sub>1</sub>		
84	D <sub>2</sub>	D <sub>2</sub>	P3 <sub>2</sub>	HDB <sub>2</sub>	EO <sub>2</sub>	FO <sub>2</sub>		
85	D <sub>3</sub>	D <sub>3</sub>	P3 <sub>3</sub>	HDB <sub>3</sub>	EO <sub>3</sub>	FO <sub>3</sub>		
86	D <sub>4</sub>	D <sub>4</sub>	P3 <sub>4</sub>	HDB <sub>4</sub>	EO <sub>4</sub>	FO <sub>4</sub>		
87	D <sub>5</sub>	D <sub>5</sub>	P3 <sub>5</sub>	HDB <sub>5</sub>	EO <sub>5</sub>	FO <sub>5</sub>		
88	D <sub>6</sub>	D <sub>6</sub>	P3 <sub>6</sub>	HDB <sub>6</sub>	EO <sub>6</sub>	FO <sub>6</sub>		
89	D <sub>7</sub>	D <sub>7</sub>	P3 <sub>7</sub>	HDB <sub>7</sub>	EO <sub>7</sub>	FO <sub>7</sub>		
90	PB <sub>1</sub> /XDB <sub>1</sub> **	PB <sub>1</sub> /XDB <sub>1</sub> **	PB <sub>1</sub>	PB <sub>1</sub>	NC	NC		
91	PB <sub>0</sub> /XDB <sub>0</sub> **	PB <sub>0</sub> /XDB <sub>0</sub> **	PB <sub>0</sub>	PB <sub>0</sub>	NC	NC		
92	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>	V <sub>SS</sub>		
93	P8 <sub>0</sub> /HA <sub>0</sub> *	P8 <sub>0</sub> /HA <sub>0</sub> *	P8 <sub>0</sub>	HA <sub>0</sub>	NC	NC		
94	P8 <sub>1</sub> /GA <sub>20</sub> *	P8 <sub>1</sub> /GA <sub>20</sub> *	P8 <sub>1</sub>	P8 <sub>1</sub> /GA <sub>20</sub>	NC	NC		
95	P8 <sub>2</sub> /CS <sub>1</sub> *	P8 <sub>2</sub> /CS <sub>1</sub> *	P8 <sub>2</sub>	CS <sub>1</sub>	NC	NC		
96	P8 <sub>3</sub> /IOR*	P8 <sub>3</sub> /IOR*	P8 <sub>3</sub>	IOR	NC	NC		

Note: Pins marked NC should be left unconnected.

For details on PROM mode, refer to 18.2, PROM Mode, 19.6, Flash Memory PROM Mode (H8/3434F) and 20.6, Flash Memory PROM Mode (H8/3437F).

\* Differs as in mode 3, depending on whether the host interface is enabled or disabled.

\*\* XDB<sub>7</sub> to XDB<sub>6</sub> can only be used when the host interface is enabled.

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

Pin No.	Expanded Modes		Single-Chip Mode		EPROM PROM Mode	Flash Memory PROM Mode		
			Mode 3					
	Mode 1	Mode 2	HIF Disabled	HIF Enabled				
97	P8 <sub>4</sub> /IRQ <sub>3</sub> /TxD <sub>1</sub> when HIF is disabled or STAC bit is 1 in STCR; IOW/IRQ <sub>3</sub> when HIF is enabled and STAC bit is 0 in STCR				NC	NC		
98	P8 <sub>5</sub> /IRQ <sub>4</sub> /RxD <sub>1</sub> when HIF is disabled or STAC bit is 1 in STCR; CS <sub>2</sub> /IRQ <sub>4</sub> when HIF is enabled and STAC bit is 0 in STCR				NC	NC		
99	P8 <sub>6</sub> /SCK <sub>1</sub> / IRQ <sub>5</sub> /SCL	P8 <sub>6</sub> /SCK <sub>1</sub> / IRQ <sub>5</sub> /SCL	P8 <sub>6</sub> /SCK <sub>1</sub> / IRQ <sub>5</sub> /SCL	P8 <sub>6</sub> /SCK <sub>1</sub> / IRQ <sub>5</sub> /SCL	NC	NC		
100	RESO	RESO	RESO	RESO	NC	NC		

Note: Pins marked NC should be left unconnected.

For details on PROM mode, refer to 18.2, PROM Mode, 19.6, Flash Memory PROM Mode (H8/3434F) and 20.6, Flash Memory PROM Mode (H8/3437F).

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

**Table 1-3 Pin Functions**

Type	Symbol	Pin No.		Name and Function
		FP-100B, TFP-100B	I/O	
Power	V <sub>CC</sub>	9, 59	I	<b>Power:</b> Connected to the power supply. Connect both V <sub>CC</sub> pins to the system power supply.
	V <sub>CCB</sub>	4	I	<b>I/O buffer power supply:</b> Power supply for input/output buffers at pins P8 <sub>6</sub> , P9 <sub>7</sub> , and PA <sub>4</sub> to PA <sub>7</sub> .
	V <sub>SS</sub>	15, 70, 71 92	I	<b>Ground:</b> Connected to ground (0 V). Connect all V <sub>SS</sub> pins to system ground (0 V).
Clock	XTAL	2	I	<b>Crystal:</b> Connected to a crystal oscillator. The crystal frequency should be the same as the desired system clock frequency. If an external clock is input at the EXTAL pin, a reverse-phase clock should be input at the XTAL pin.
	EXTAL	3	I	<b>External crystal:</b> Connected to a crystal oscillator or external clock. The frequency of the external clock should be the same as the desired system clock frequency. See section 6.2, Oscillator Circuit, for examples of connections to a crystal and external clock.
	Ø	17	O	<b>System clock:</b> Supplies the system clock to peripheral devices.
System control	RES	1	I	<b>Reset:</b> A low input causes the chip to reset.
	RESO	100	O	<b>Reset output:</b> Outputs a reset signal to external devices.
	STBY	8	I	<b>Standby:</b> A transition to the hardware standby mode (a power-down state) occurs when a low input is received at the STBY pin.
Address bus	A <sub>15</sub> to A <sub>0</sub>	60 to 67, 72 to 79	O	<b>Address bus:</b> Address output pins.
Data bus	D <sub>7</sub> to D <sub>0</sub>	89 to 82	I/O	<b>Data bus:</b> 8-bit bidirectional data bus.

**Table 1-3 Pin Functions (cont)**

Type	Symbol	Pin No.		Name and Function
		FP-100B, TFP-100B	I/O	
Bus control	WAIT	16	I	<b>Wait:</b> Requests the CPU to insert wait states into the bus cycle when an external address is accessed.
	RD	22	O	<b>Read:</b> Goes low to indicate that the CPU is reading an external address.
	WR	19	O	<b>Write:</b> Goes low to indicate that the CPU is writing to an external address.
	AS	18	O	<b>Address strobe:</b> Goes low to indicate that there is a valid address on the address bus.
Interrupt signals	NMI	7	I	<b>Nonmaskable interrupt:</b> Highest-priority interrupt request. The NMIEG bit in the system control register (SYSCR) determines whether the interrupt is recognized at the rising or falling edge of the NMI input.
	IRQ <sub>0</sub> to IRQ <sub>7</sub>	23 to 25, 97 to 99, 34, 35	I	<b>Interrupt request 0 to 7:</b> Maskable interrupt request pins.
Operating mode control	MD <sub>1</sub>	5	I	<b>Mode:</b> Input pins for setting the MCU mode operating mode according to the table below.
	MD <sub>0</sub>	6		<b>MD<sub>1</sub> MD<sub>0</sub> Mode Description</b>
			0 1	Mode 1 Expanded mode with on-chip ROM disabled
			1 0	Mode 2 Expanded mode with on-chip ROM enabled
			1 1	Mode 3 Single-chip mode
16-bit free-running timer (FRT)	FTOA	27	O	<b>FRT output compare A and B:</b> Output pins controlled by comparators A and B of the free-running timer.
	FTOB	34		
	FTCI	26	I	<b>FRT counter clock input:</b> Input pin for an external clock signal for the free-running timer.
	FTIA to FTID	28, 29, 32, 33	I	<b>FRT input capture A to D:</b> Input capture pins for the free-running timer.

**Table 1-3 Pin Functions (cont)**

Pin No.				
Type	Symbol	FP-100B, TFP-100B	I/O	Name and Function
8-bit timer	TMO <sub>0</sub>	50	O	<b>8-bit timer output (channels 0 and 1):</b> Compare-match output pins for the 8-bit timers.
	TMO <sub>1</sub>	53		
	TMCI <sub>0</sub>	49	I	<b>8-bit timer counter clock input (channels 0 and 1):</b> External clock input pins for the 8-bit timer counters.
	TMCI <sub>1</sub>	52		
	TMRI <sub>0</sub>	51	I	<b>8-bit timer counter reset input (channels 0 and 1):</b> A high input at these pins resets the 8-bit timer counters.
	TMRI <sub>1</sub>	54		
PWM timer	PW <sub>0</sub>	55	O	<b>PWM timer output (channels 0 and 1):</b> Pulse-width modulation timer output pins.
	PW <sub>1</sub>	56		
Serial communication interface (SCI)	TxD <sub>0</sub>	14	O	<b>Transmit data (channels 0 and 1):</b> Data output pins for the serial communication interface.
	TxD <sub>1</sub>	97		
	RxD <sub>0</sub>	13	I	<b>Receive data (channels 0 and 1):</b> Data input pins for the serial communication interface.
	RxD <sub>1</sub>	98		
Host interface (HIF)	SCK <sub>0</sub>	12	I/O	<b>Serial clock (channels 0 and 1):</b> Input/output pins for the serial clock.
	SCK <sub>1</sub>	99		
	HDB <sub>0</sub> to HDB <sub>7</sub>	82 to 89	I/O	<b>Host interface data bus:</b> 8-bit bidirectional bus by which a host processor accesses the host interface.
	CS <sub>1</sub> , CS <sub>2</sub>	95, 98	I	<b>Chip select 1 and 2:</b> Input pins for selecting host interface channels 1 and 2.
	IOR	96	I	<b>I/O read:</b> Read strobe input pin for the host interface.
	IOW	97	I	<b>I/O write:</b> Write strobe input pin for the host interface.
	HA <sub>0</sub>	93	I	<b>Command/data:</b> Input pin indicating data access or command access.
	GA <sub>20</sub>	94	O	<b>Gate A<sub>20</sub>:</b> A <sub>20</sub> gate control signal output pin.
	HIRQ <sub>1</sub>	53	O	<b>Host interrupts 1, 11, and 12:</b> Output pins for interrupt request signals to the host processor.
	HIRQ <sub>11</sub>	52		
	HIRQ <sub>12</sub>	54		
Keyboard control	KEYIN <sub>0</sub> to KEYIN <sub>15</sub>	26 to 29, 32 to 35, 48, 47, 31, 30, 21, 20, 11, 10	I	<b>Keyboard input:</b> Input pins from a matrix keyboard. (Keyboard scan signals are normally output from P1 <sub>0</sub> to P1 <sub>7</sub> and P2 <sub>0</sub> to P2 <sub>7</sub> , allowing a maximum 16 × 16 key matrix. The number of keys can be further increased by use of other output ports.)

**Table 1-3 Pin Functions (cont)**

Type	Symbol	Pin No. FP-100B, TFP-100B	I/O	Name and Function
Host interface (expanded modes)	XDB <sub>0</sub> to XDB <sub>7</sub>	91, 90, 81, 80, 69, 68, 58, 57	I/O	<b>Host interface data bus:</b> 8-bit bidirectional bus by which a host processor accesses the host interface.
Host interface (if enabled when STAC bit is 1 in STCR)	ECS <sub>2</sub>	25	I	<b>Host chip select 2:</b> Input pin for selecting host interface channel 2.
	EIOW	24	I	<b>I/O write:</b> Write strobe input pin for the host interface.
A/D converter	AN <sub>7</sub> to AN <sub>0</sub>	38 to 45	I	<b>Analog input:</b> Analog signal input pins for the A/D converter.
	ADTRG	25	I	<b>A/D trigger:</b> External trigger input for starting the A/D converter.
D/A converter	DA <sub>0</sub>	44	O	<b>Analog output:</b> Analog signal output pins for the D/A converter.
DA <sub>1</sub>	45			
A/D and D/A converters	AV <sub>CC</sub>	37	I	<b>Analog reference voltage:</b> Reference voltage pin for the A/D and D/A converters. If the A/D and D/A converters are not used, connect AV <sub>CC</sub> to the system power supply.
	AV <sub>SS</sub>	46	I	<b>Analog ground:</b> Ground pin for the A/D and D/A converters. Connect to system ground (0 V).
	AV <sub>ref</sub>	36	I	<b>Analog reference voltage:</b> Analog reference voltage input pins for A/D and D/A converters.
Flash memory [H8/3434, H8/3437 F-ZTAT]	FV <sub>PP</sub>	8	I	<b>Programming power supply for on-board programming:</b> Connect to a flash memory programming power supply (+12 V)
I <sup>2</sup> C bus interface [option]	SCL	99	I/O	<b>I<sup>2</sup>C clock I/O:</b> Input/output pin for I <sup>2</sup> C clock. Power is supplied by I/O buffer power supply V <sub>CCB</sub> . Features a bus drive function.
	SDA	16	I/O	<b>I<sup>2</sup>C data I/O:</b> Input/output pin for I <sup>2</sup> C data. Power is supplied by I/O buffer power supply V <sub>CCB</sub> . Features a bus drive function.

**Table 1-3 Pin Functions (cont)**

Type	Symbol	Pin No.		Name and Function
		FP-100B, TFP-100B	I/O	
I/O ports	P <sub>1<sub>7</sub></sub> to P <sub>1<sub>0</sub></sub>	72 to 79	I/O	<b>Port 1:</b> An 8-bit input/output port with programmable MOS input pull-ups and LED driving capability. The direction of each bit can be selected in the port 1 data direction register (P1DDR).
	P <sub>2<sub>7</sub></sub> to P <sub>2<sub>0</sub></sub>	60 to 67	I/O	<b>Port 2:</b> An 8-bit input/output port with programmable MOS input pull-ups and LED driving capability. The direction of each bit can be selected in the port 2 data direction register (P2DDR).
	P <sub>3<sub>7</sub></sub> to P <sub>3<sub>0</sub></sub>	89 to 82	I/O	<b>Port 3:</b> An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 3 data direction register (P3DDR).
	P <sub>4<sub>7</sub></sub> to P <sub>4<sub>0</sub></sub>	56 to 49	I/O	<b>Port 4:</b> An 8-bit input/output port. The direction of each bit can be selected in the port 4 data direction register (P4DDR).
	P <sub>5<sub>2</sub></sub> to P <sub>5<sub>0</sub></sub>	12 to 14	I/O	<b>Port 5:</b> A 3-bit input/output port. The direction of each bit can be selected in the port 5 data direction register (P5DDR).
	P <sub>6<sub>7</sub></sub> to P <sub>6<sub>0</sub></sub>	35 to 32, 29 to 26	I/O	<b>Port 6:</b> An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port 6 data direction register (P6DDR).
	P <sub>7<sub>7</sub></sub> to P <sub>7<sub>0</sub></sub>	45 to 38	I	<b>Port 7:</b> An 8-bit input port.
	P <sub>8<sub>6</sub></sub> to P <sub>8<sub>0</sub></sub>	99 to 93	I/O	<b>Port 8:</b> A 7-bit input/output port. The direction of each bit can be selected in the port 8 data direction register (P8DDR). P <sub>8<sub>6</sub></sub> is powered by I/O buffer power supply V <sub>CCB</sub> .
	P <sub>9<sub>7</sub></sub> to P <sub>9<sub>0</sub></sub>	16 to 19, 22 to 25	I/O	<b>Port 9:</b> An 8-bit input/output port. The direction of each bit (except for P <sub>9<sub>6</sub></sub> ) can be selected in the port 9 data direction register (P9DDR). P <sub>9<sub>7</sub></sub> is powered by I/O buffer power supply V <sub>CCB</sub> .
	PA <sub>7</sub> to PA <sub>0</sub>	10, 11, 20, 21, 30, 31, 47, 48	I/O	<b>Port A:</b> An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port A data direction register (PADDR). PA <sub>7</sub> to PA <sub>4</sub> are powered by I/O buffer power supply V <sub>CCB</sub> . Features a bus drive function.
	PB <sub>7</sub> to PB <sub>0</sub>	57, 58, 68, 69, 80, 81, 90, 91	I/O	<b>Port B:</b> An 8-bit input/output port with programmable MOS input pull-ups. The direction of each bit can be selected in the port B data direction register (PBDDR).

Note: In this chip, the same pin is used for STBY and FVpp. When this pin is driven low, a transition is made to hardware standby mode. This occurs not only in the normal operating modes (modes 1, 2, and 3), but also when programming flash memory with a PROM writer. Therefore, a PROM writer should be used whose specifications provide for this pin to be held at the V<sub>cc</sub> level except when programming (FVpp = 12 V).

# Section 2 CPU

## 2.1 Overview

The H8/300 CPU is a fast central processing unit with eight 16-bit general registers (also configurable as 16 eight-bit registers) and a concise instruction set designed for high-speed operation.

### 2.1.1 Features

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

- Two-way register configuration
  - Sixteen 8-bit general registers, or
  - Eight 16-bit general registers
- Instruction set with 57 basic instructions, including:
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
- Eight addressing modes
  - Register direct (Rn)
  - Register indirect (@Rn)
  - Register indirect with displacement (@(d:16, Rn))
  - Register indirect with post-increment or pre-decrement (@Rn+ or @-Rn)
  - Absolute address (@aa:8 or @aa:16)
  - Immediate (#xx:8 or #xx:16)
  - PC-relative (@(d:8, PC))
  - Memory indirect (@@aa:8)
- Maximum 64-kbyte address space
- High-speed operation
  - All frequently-used instructions are executed in two to four states
- Maximum clock rate ( $\phi$  clock): 16 MHz at 5 V, 12 MHz at 4 V or 10 MHz at 3 V
  - 8- or 16-bit register-register add or subtract: 125 ns (16 MHz), 167 ns (12 MHz), 200 ns (10 MHz)
  - $8 \times$  8-bit multiply: 875 ns (16 MHz), 1167 ns (12 MHz), 1400 ns (10 MHz)
  - $16 \div$  8-bit divide: 875 ns (16 MHz), 1167 ns (12 MHz), 1400 ns (10 MHz)
- Power-down mode
  - SLEEP instruction

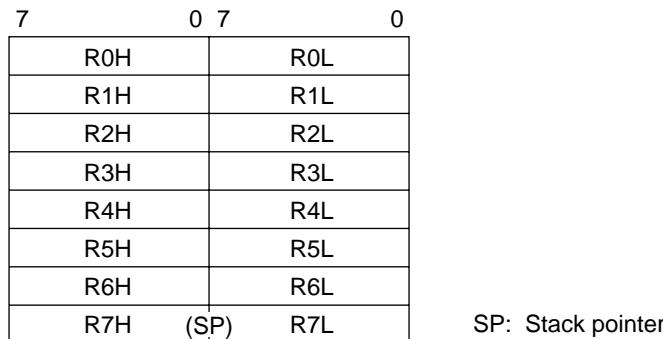
## 2.1.2 Address Space

The H8/300 CPU supports an address space with a maximum size of 64 kbytes for program code and data combined. The memory map differs depending on the mode (mode 1, 2, or 3). For details, see section 3.4, Address Space Map in Each Operating Mode.

## 2.1.3 Register Configuration

Figure 2-1 shows the internal register structure of the H8/300 CPU. There are two groups of registers: the general registers and control registers.

### General registers (Rn)



### Control registers

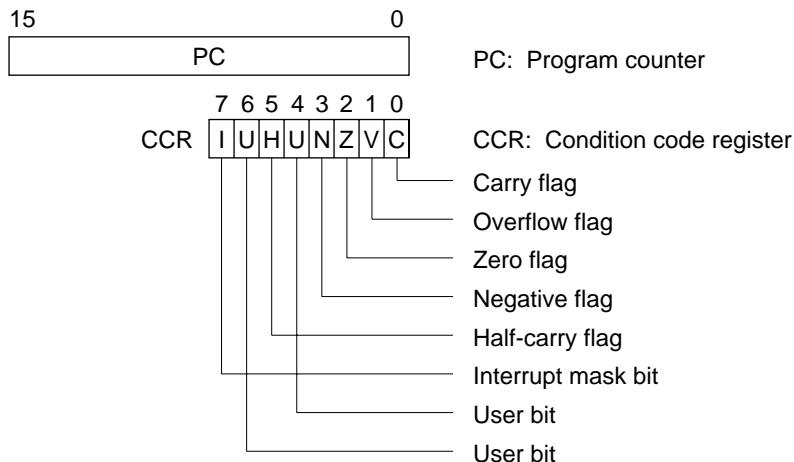


Figure 2-1 CPU Registers

## 2.2 Register Descriptions

### 2.2.1 General Registers

All the general registers can be used as both data registers and address registers. When used as address registers, the general registers are accessed as 16-bit registers (R0 to R7). When used as data registers, they can be accessed as 16-bit registers, or the high and low bytes can be accessed separately as 8-bit registers (R0H to R7H and R0L to R7L).

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

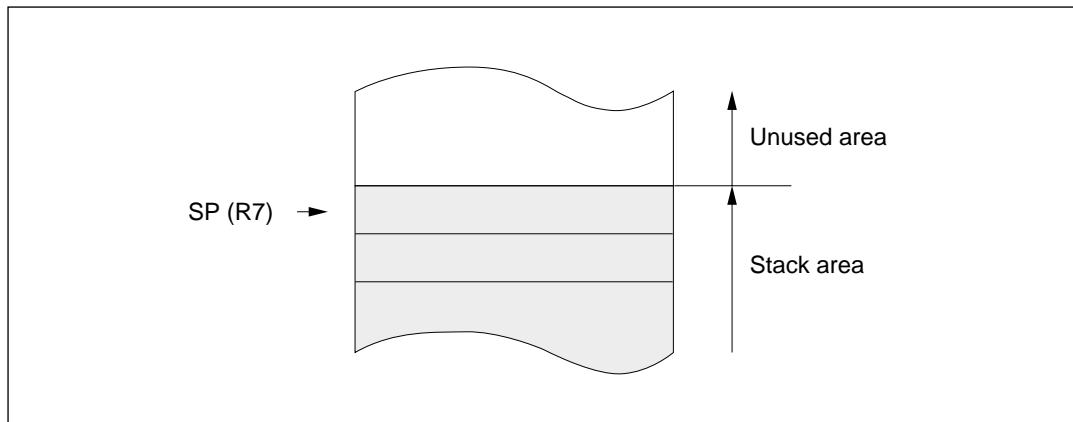


Figure 2-2 Stack Pointer

### 2.2.2 Control Registers

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

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

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

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

**Bit 6—User Bit (U):** This bit can be written and read by software (using the LDC, STC, ANDC, ORC, and XORC instructions).

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

**Bit 4—User Bit (U):** This bit can be written and read by software (using the LDC, STC, ANDC, ORC, and XORC instructions).

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

**Bit 2—Zero Flag (Z):** This flag is set to 1 to indicate a zero result and cleared to 0 to indicate a nonzero result.

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

**Bit 0—Carry Flag (C):** This flag is used by:

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

The LDC, STC, ANDC, ORC, and XORC instructions enable the CPU to load and store the CCR, and to set or clear selected bits by logic operations. The N, Z, V, and C flags are used in conditional branching instructions (B<sub>CC</sub>).

For the action of each instruction on the flag bits, see the *H8/300 Series Programming Manual*.

### 2.2.3 Initial Register Values

When the CPU is reset, the program counter (PC) is loaded from the vector table and the interrupt mask bit (I) in the CCR is set to 1. The other CCR bits and the general registers are not initialized. In particular, the stack pointer (R7) is not initialized. The stack pointer and CCR should be initialized by software, by the first instruction executed after a reset.

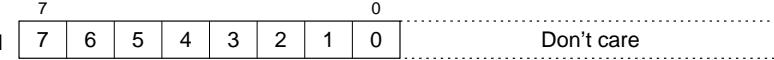
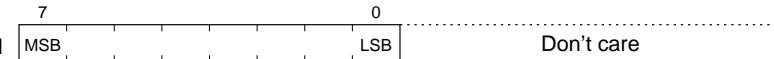
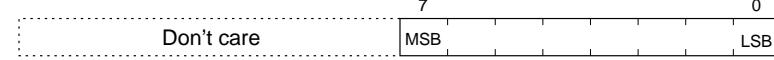
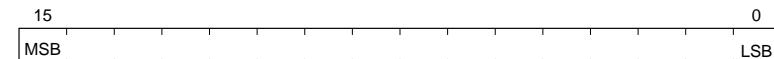
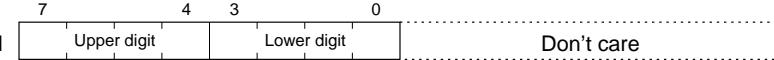
## 2.3 Data Formats

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

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

### 2.3.1 Data Formats in General Registers

Data of all the sizes above can be stored in general registers as shown in figure 2-3.

Data Type	Register No.	Data Format
1-bit data	RnH	
1-bit data	RnL	
Byte data	RnH	
Byte data	RnL	
Word data	Rn	
4-bit BCD data	RnH	
4-bit BCD data	RnL	

#### Legend

RnH: Upper digit of general register

RnL: Lower digit of general register

MSB: Most significant bit

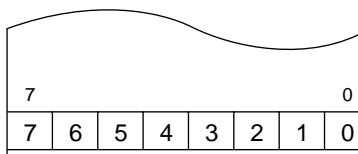
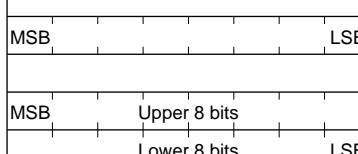
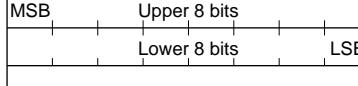
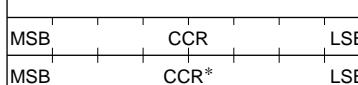
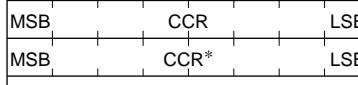
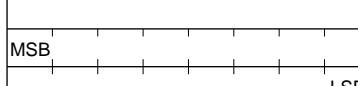
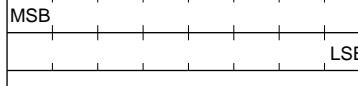
LSB: Least significant bit

**Figure 2-3 Register Data Formats**

### 2.3.2 Memory Data Formats

Figure 2-4 indicates the data formats in memory.

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

<b>Data Type</b>	<b>Address</b>	<b>Data Format</b>
1-bit data	Address n	
Byte data	Address n	
Word data	Even address	
	Odd address	
Byte data (CCR) on stack	Even address	
	Odd address	
Word data on stack	Even address	
	Odd address	

Note: \* Ignored on return

Legend  
CCR: Condition code register

**Figure 2-4 Memory Data Formats**

When the stack is addressed by register R7, it must always be accessed a word at a time. When the CCR is pushed on the stack, two identical copies of the CCR are pushed to make a complete word. When they are restored, the lower byte is ignored.

## 2.4 Addressing Modes

### 2.4.1 Addressing Mode

The H8/300 CPU supports eight addressing modes. Each instruction uses a subset of these addressing modes.

**Table 2-1 Addressing Modes**

No.	Addressing Mode	Symbol
(1)	Register direct	Rn
(2)	Register indirect	@Rn
(3)	Register indirect with displacement	@(d:16, Rn)
(4)	Register indirect with post-increment Register indirect with pre-decrement	@Rn+ @-Rn
(5)	Absolute address	@aa:8 or @aa:16
(6)	Immediate	#xx:8 or #xx:16
(7)	Program-counter-relative	@(d:8, PC)
(8)	Memory indirect	@@aa:8

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

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

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

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

- Register indirect with Post-Increment—@Rn+

The @Rn+ mode is used with MOV instructions that load registers from memory.

It is similar to the register indirect mode, but the 16-bit general register specified in the register field of the instruction is incremented after the operand is accessed. The size of the increment is 1 or 2 depending on the size of the operand: 1 for MOV.B; 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

- Register Indirect with Pre-Decrement—@-Rn

The @-Rn mode is used with MOV instructions that store register contents to memory.

It is similar to the register indirect mode, but the 16-bit general register specified in the register field of the instruction is decremented before the operand is accessed. The size of the decrement is 1 or 2 depending on the size of the operand: 1 for MOV.B; 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

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

**(6) Immediate—#xx:8 or #xx:16:** The instruction contains an 8-bit operand in its second byte, or a 16-bit operand in its third and fourth bytes. Only MOV.W instructions can contain 16-bit immediate values.

The ADDS and SUBS instructions implicitly contain the value 1 or 2 as immediate data. Some bit manipulation instructions contain 3-bit immediate data (#xx:3) in the second or fourth byte of the instruction, specifying a bit number.

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

**(8) Memory Indirect—@@aa:8:** This mode can be used by the JMP and JSR instructions. The second byte of the instruction code specifies an 8-bit absolute address from H'0000 to H'00FF (0 to 255). The word located at this address contains the branch address. The upper 8 bits of the absolute address are 0 (H'00), thus the branch address is limited to values from 0 to 255 (H'0000 to H'00FF). Note that some of the addresses in this range are also used in the vector table. Refer to section 3.4, Address Space Map in Each Operating Mode.

If an odd address is specified as a branch destination or as the operand address of a MOV.W instruction, the least significant bit is regarded as 0, causing word access to be performed at the address preceding the specified address. See section 2.3.2, Memory Data Formats, for further information.

## 2.4.2 Calculation of Effective Address

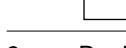
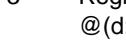
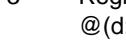
Table 2-2 shows how the H8/300 calculates effective addresses in each addressing mode.

Arithmetic, logic, and shift instructions use register direct addressing (1). The ADD.B, ADDX.B, SUBX.B, CMP.B, AND.B, OR.B, and XOR.B instructions can also use immediate addressing (6).

The MOV instruction uses all the addressing modes except program-counter relative (7) and memory indirect (8).

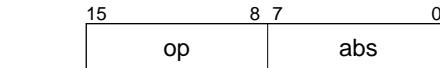
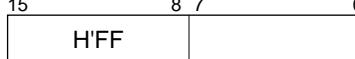
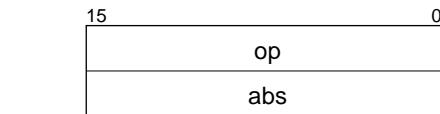
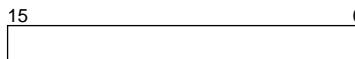
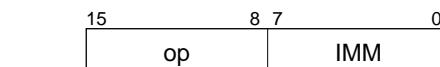
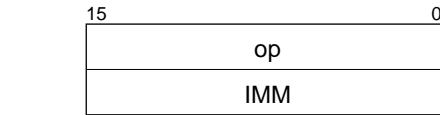
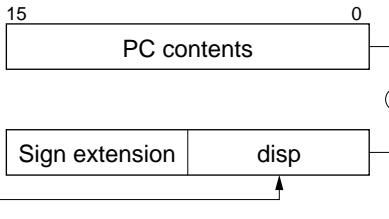
Bit manipulation instructions use register direct (1), register indirect (2), or 8-bit absolute (5) addressing to identify a byte operand, and 3-bit immediate addressing to identify a bit within the byte. The BSET, BCLR, BNOT, and BTST instructions can also use register direct addressing (1) to identify the bit.

Table 2-2 Effective Address Calculation

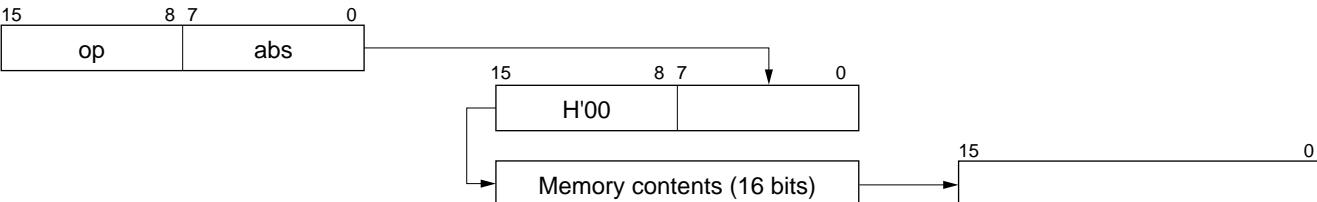
No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address
1	Register direct, Rn		
			
2	Register indirect, @Rn		
3	Register indirect with displacement, @d:16, Rn		
4	Register indirect with post-increment, @Rn+1 or 2*		
	Register indirect with pre-decrement, @-Rn		

Note: \* 1 for a byte operand, 2 for a word operand

Table 2-2 Effective Address Calculation (cont)

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective address
5	Absolute address @aa:8		
	@aa:16		
6	Immediate #xx:8		Operand is 1- or 2-byte immediate data
	#xx:16		
7	PC-relative @(d:8, PC)		

**Table 2-2 Effective Address Calculation (cont)**

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address
8	Memory indirect, @@aa:8	 <p>Diagram illustrating the Effective Address Calculation for Memory indirect addressing (Format 8). The instruction word is 16 bits long, divided into four fields: op (bits 15-8), abs (bits 7-0), and two 8-bit fields (bits 15-8 and bits 7-0). The abs field is used as an index into memory. The memory contains a 16-bit value (H'0000). This value is then used as an absolute address to read memory, which returns the final 16-bit effective address.</p>	

**Legend**

reg: General register

op: Operation code

disp: Displacement

IMM: Immediate data

abs: Absolute address

## 2.5 Instruction Set

The H8/300 CPU has 57 types of instructions, which are classified by function in table 2-3.

**Table 2-3 Instruction Classification**

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

Notes: 1. PUSH Rn is equivalent to MOV.W Rn, @-SP.

POP Rn is equivalent to MOV.W @SP+, Rn.

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

3. Not supported by the H8/3437 Series.

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

### Operation Notation

Rd	General register (destination)	#xx:3	3-Bit immediate data
Rs	General register (source)	#xx:8	8-Bit immediate data
Rn	General register	#xx:16	16-Bit immediate data
(EAd)	Destination operand	disp	Displacement
(EAs)	Source operand	+	Addition
SP	Stack pointer	-	Subtraction
PC	Program counter	×	Multiplication
CCR	Condition code register	÷	Division
N	N (negative) flag of CCR	^	AND logical
Z	Z (zero) flag of CCR	∨	OR logical
V	V (overflow) flag of CCR	⊕	Exclusive OR logical
C	C (carry) flag of CCR	→	Move
#imm	Immediate data	¬	Not

## 2.5.1 Data Transfer Instructions

Table 2-4 describes the data transfer instructions. Figure 2-5 shows their object code formats.

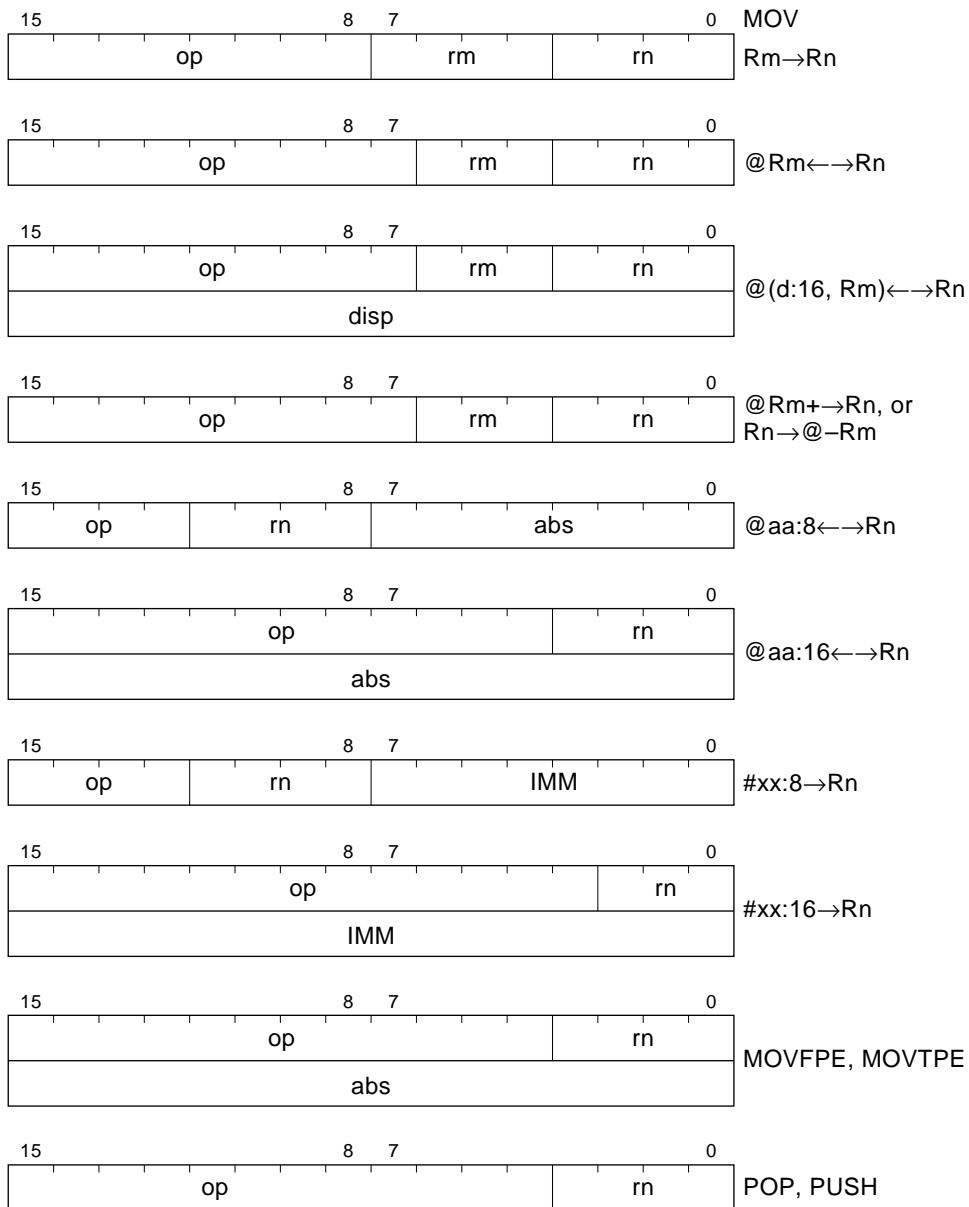
**Table 2-4 Data Transfer Instructions**

Instruction	Size*	Function
MOV	B/W	(EAs) → Rd, Rs → (EAd) Moves data between two general registers or between a general register and memory, or moves immediate data to a general register. The Rn, @Rn, @(d:16, Rn), @aa:16, #xx:8 or #xx:16, @-Rn, and @Rn+ addressing modes are available for byte or word data. The @aa:8 addressing mode is available for byte data only. The @-R7 and @R7+ modes require word operands. Do not specify byte size for these two modes.
MOVTP	B	Not supported by the H8/3437 Series.
MOVFP	B	Not supported by the H8/3437 Series.
PUSH	W	Rn → @-SP Pushes a 16-bit general register onto the stack. Equivalent to MOV.W Rn, @-SP.
POP	W	@SP+ → Rn Pops a 16-bit general register from the stack. Equivalent to MOV.W @SP+, Rn.

Note: \* Size: Operand size

B: Byte

W: Word



#### Legend

- op: Operation field
- rm, rn: Register field
- disp: Displacement
- abs: Absolute address
- IMM: Immediate data

**Figure 2-5 Data Transfer Instruction Codes**

## 2.5.2 Arithmetic Operations

Table 2-5 describes the arithmetic instructions. See figure 2-6 in section 2.5.4, Shift Operations, for their object codes.

**Table 2-5 Arithmetic Instructions**

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

Note: \* Size: Operand size

B: Byte

W: Word

### 2.5.3 Logic Operations

Table 2-6 describes the four instructions that perform logic operations. See figure 2-6 in section 2.5.4, Shift Operations, for their object codes.

**Table 2-6 Logic Operation Instructions**

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

Note: \* Size: Operand size  
B: Byte

### 2.5.4 Shift Operations

Table 2-7 describes the eight shift instructions. Figure 2-6 shows the object code formats of the arithmetic, logic, and shift instructions.

**Table 2-7 Shift Instructions**

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

Note: \* Size: Operand size  
B: Byte

15	op	8	7	rm	rn	0	ADD, SUB, CMP, ADDX, SUBX (Rm)	
15	op	8	7		rn	0	ADDS, SUBS, INC, DEC, DAA, DAS, NEG, NOT	
15	op	8	7	rm	rn	0	MULXU, DIVXU	
15	op	rn	8	7	IMM	0	ADD, ADDX, SUBX, CMP (#xx:8)	
15	op	rn	8	7	rm	rn	0	AND, OR, XOR (Rm)
15	op	rn	8	7	IMM	0	AND, OR, XOR (#xx:8)	
15	op	8	7		rn	0	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	

Legend

op: Operation field  
 rm, rn: Register field  
 IMM: Immediate data

**Figure 2-6 Arithmetic, Logic, and Shift Instruction Codes**

## 2.5.5 Bit Manipulations

Table 2-8 describes the bit-manipulation instructions. Figure 2-7 shows their object code formats.

**Table 2-8 Bit-Manipulation Instructions**

Instruction	Size*	Function
BSET	B	$1 \rightarrow (\text{bit no.} \text{ of } \text{EAd})$ Sets a specified bit in a general register or memory to 1. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BCLR	B	$0 \rightarrow (\text{bit no.} \text{ of } \text{EAd})$ Clears a specified bit in a general register or memory to 0. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BNOT	B	$\neg (\text{bit no.} \text{ of } \text{EAd}) \rightarrow (\text{bit no.} \text{ of } \text{EAd})$ Inverts a specified bit in a general register or memory. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register
BTST	B	$\neg (\text{bit no.} \text{ of } \text{EAd}) \rightarrow Z$ Tests a specified bit in a general register or memory and sets or clears the Z flag accordingly. The bit is specified by a bit number, given in 3-bit immediate data or the lower three bits of a general register.
BAND	B	$C \wedge (\text{bit no.} \text{ of } \text{EAd}) \rightarrow C$ ANDs the C flag with a specified bit in a general register or memory.
BIAND		$C \wedge [\neg (\text{bit no.} \text{ of } \text{EAd})] \rightarrow C$ ANDs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BOR	B	$C \vee (\text{bit no.} \text{ of } \text{EAd}) \rightarrow C$ ORs the C flag with a specified bit in a general register or memory.
BIOR		$C \vee [\neg (\text{bit no.} \text{ of } \text{EAd})] \rightarrow C$ ORs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BXOR	B	$C \oplus (\text{bit no.} \text{ of } \text{EAd}) \rightarrow C$ XORs the C flag with a specified bit in a general register or memory.

Note: \* Size: Operand size

B: Byte

**Table 2-8 Bit-Manipulation Instructions (cont)**

Instruction	Size*	Function
BIXOR	B	$C \oplus \neg [(\text{bit no.}) \text{ of } \text{EAd}]) \rightarrow C$ XORs the C flag with the inverse of a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.
BLD	B	$(\text{bit no.}) \text{ of } \text{EAd} \rightarrow C$ Copies a specified bit in a general register or memory to the C flag.
BILD		$\neg (\text{bit no.}) \text{ of } \text{EAd} \rightarrow C$ Copies the inverse of a specified bit in a general register or memory to the C flag. The bit number is specified by 3-bit immediate data.
BST	B	$C \rightarrow (\text{bit no.}) \text{ of } \text{EAd}$ Copies the C flag to a specified bit in a general register or memory.
BIST		$\neg C \rightarrow (\text{bit no.}) \text{ of } \text{EAd}$ Copies the inverse of the C flag to a specified bit in a general register or memory. The bit number is specified by 3-bit immediate data.

Note: \* Size: Operand size  
B: Byte

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

Step	Description	
1	Read	Read one data byte at the specified address
2	Modify	Modify one bit in the data byte
3	Write	Write the modified data byte back to the specified address

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

P4<sub>7</sub>: Input pin, low  
P4<sub>6</sub>: Input pin, high  
P4<sub>5</sub> – P4<sub>0</sub>: Output pins, low

The intended purpose of this BCLR instruction is to switch P4<sub>0</sub> from output to input.

## Before Execution of BCLR Instruction

	P4 <sub>7</sub>	P4 <sub>6</sub>	P4 <sub>5</sub>	P4 <sub>4</sub>	P4 <sub>3</sub>	P4 <sub>2</sub>	P4 <sub>1</sub>	P4 <sub>0</sub>
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

## Execution of BCLR Instruction

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

## After Execution of BCLR Instruction

	P4 <sub>7</sub>	P4 <sub>6</sub>	P4 <sub>5</sub>	P4 <sub>4</sub>	P4 <sub>3</sub>	P4 <sub>2</sub>	P4 <sub>1</sub>	P4 <sub>0</sub>
Input/output	Output	Input						
Pin state	Low	High	Low	Low	Low	Low	Low	High
DDR	1	1	1	1	1	1	1	0
DR	1	0	0	0	0	0	0	0

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

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

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

As a result, P4<sub>0</sub>DDR is cleared to 0, making P4<sub>0</sub> an input pin. In addition, P4<sub>7</sub>DDR and P4<sub>6</sub>DDR are set to 1, making P4<sub>7</sub> and P4<sub>6</sub> output pins.

**BSET, BCLR, BNOT, BTST**

15	op	8 7	IMM	rn 0
----	----	-----	-----	------

Operand: register direct (Rn)  
Bit no.: immediate (#xx:3)

15	op	8 7	rm	rn 0
----	----	-----	----	------

Operand: register direct (Rn)  
Bit no.: register direct (Rm)

15	op	8 7	rn	0 0 0 0 0
	op		IMM	0 0 0 0 0

Operand: register indirect (@Rn)  
Bit no.: immediate (#xx:3)

15	op	8 7	rn	0 0 0 0 0
	op		rm	0 0 0 0 0

Operand: register indirect (@Rn)  
Bit no.: register direct (Rm)

15	op	8 7	abs	0
	op		IMM	0 0 0 0 0

Operand: absolute (@aa:8)  
Bit no.: immediate (#xx:3)

15	op	8 7	abs	0
	op		rm	0 0 0 0 0

Operand: absolute (@aa:8)  
Bit no.: register direct (Rm)

**BAND, BOR, BXOR, BLD, BST**

15	op	8 7	IMM	rn 0
----	----	-----	-----	------

Operand: register direct (Rn)  
Bit no.: immediate (#xx:3)

15	op	8 7	rn	0 0 0 0 0
	op		IMM	0 0 0 0 0

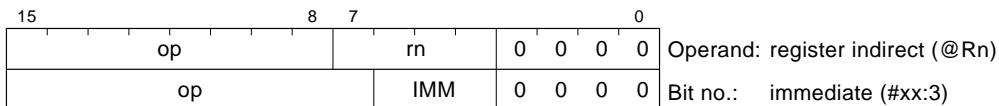
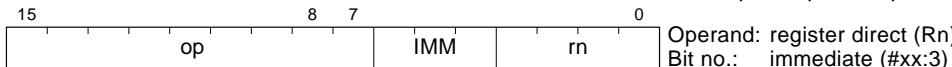
Operand: register indirect (@Rn)  
Bit no.: immediate (#xx:3)

15	op	8 7	abs	0
	op		IMM	0 0 0 0 0

Operand: absolute (@aa:8)  
Bit no.: immediate (#xx:3)

Legend  
op: Operation field  
rm, rn: Register field  
abs: Absolute address  
IMM: Immediate data

**Figure 2-7 Bit Manipulation Instruction Codes (1)**

**BIAND, BIOR, BIXOR, BILD, BIST****Legend**

op: Operation field  
rm, rn: Register field  
abs: Absolute address  
IMM: Immediate data

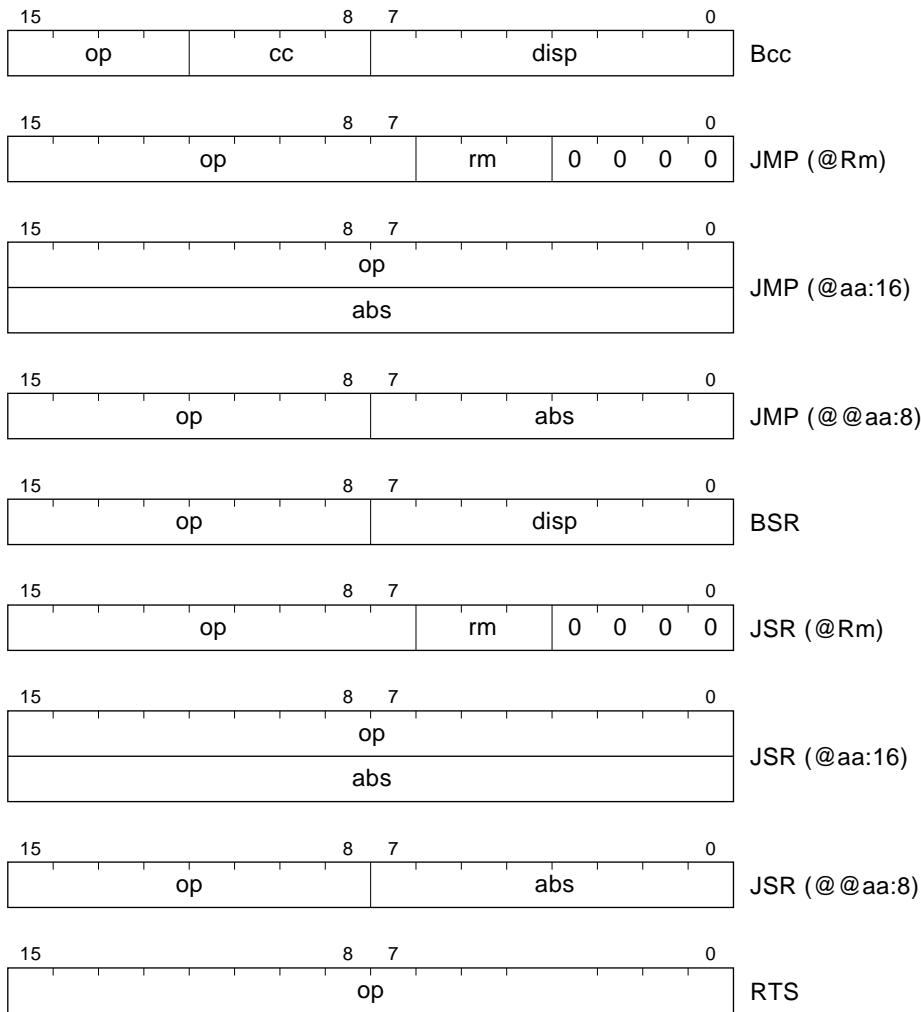
**Figure 2-7 Bit Manipulation Instruction Codes (2)**

## 2.5.6 Branching Instructions

Table 2-9 describes the branching instructions. Figure 2-8 shows their object code formats.

**Table 2-9 Branching Instructions**

Instruction	Size	Function				
		Mnemonic	cc field	Description	Condition	
Bcc	—	Branches if condition cc is true.				
		BRA (BT)	0 0 0 0	Always (true)	Always	
		BRN (BF)	0 0 0 1	Never (false)	Never	
		BHI	0 0 1 0	High	$C \vee Z = 0$	
		BLS	0 0 1 1	Low or same	$C \vee Z = 1$	
		BCC (BHS)	0 1 0 0	Carry clear (High or same)	$C = 0$	
		BCS (BLO)	0 1 0 1	Carry set (low)	$C = 1$	
		BNE	0 1 1 0	Not equal	$Z = 0$	
		BEQ	0 1 1 1	Equal	$Z = 1$	
		BVC	1 0 0 0	Overflow clear	$V = 0$	
		BVS	1 0 0 1	Overflow set	$V = 1$	
		BPL	1 0 1 0	Plus	$N = 0$	
		BMI	1 0 1 1	Minus	$N = 1$	
		BGE	1 1 0 0	Greater or equal	$N \oplus V = 0$	
		BLT	1 1 0 1	Less than	$N \oplus V = 1$	
		BGT	1 1 1 0	Greater than	$Z \vee (N \oplus V) = 0$	
		BLE	1 1 1 1	Less or equal	$Z \vee (N \oplus V) = 1$	
JMP	—	Branches unconditionally to a specified address.				
JSR	—	Branches to a subroutine at a specified address.				
BSR	—	Branches to a subroutine at a specified displacement from the current address.				
RTS	—	Returns from a subroutine.				



#### Legend

- op: Operation field
- cc: Condition field
- rm: Register field
- disp: Displacement
- abs: Absolute address

**Figure 2-8 Branching Instruction Codes**

## 2.5.7 System Control Instructions

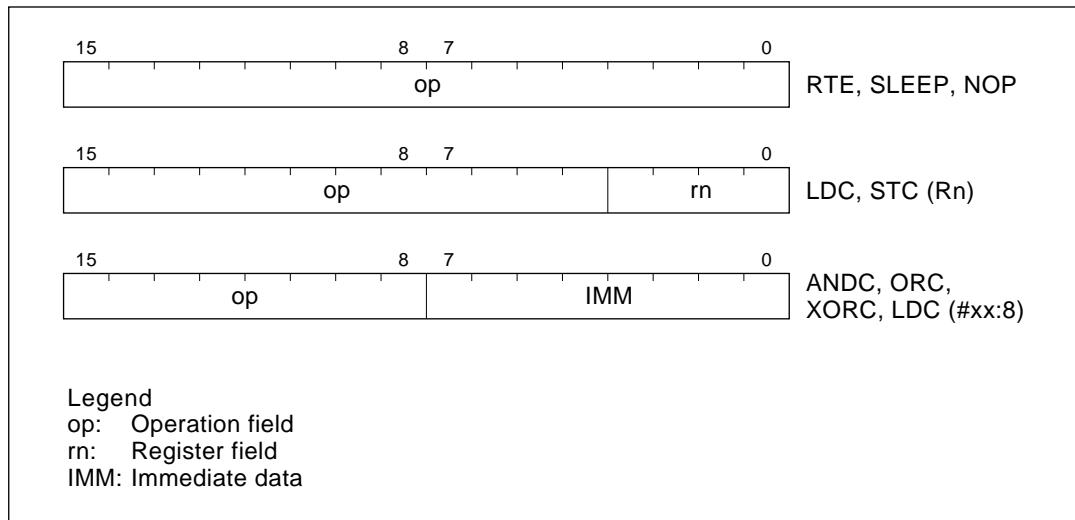
Table 2-10 describes the system control instructions. Figure 2-9 shows their object code formats.

**Table 2-10 System Control Instructions**

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

Note: \* Size: Operand size

B: Byte



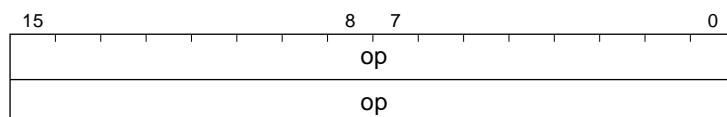
**Figure 2-9 System Control Instruction Codes**

## 2.5.8 Block Data Transfer Instruction

Table 2-11 describes the EEPMOV instruction. Figure 2-10 shows its object code format.

**Table 2-11 Block Data Transfer Instruction/EEPROM Write Operation**

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

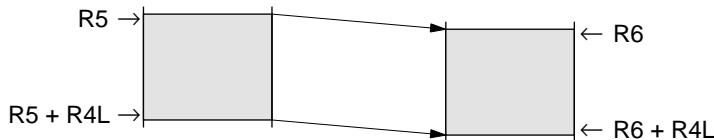


Legend  
op: Operation field

**Figure 2-10 Block Data Transfer Instruction/EEPROM Write Operation Code**

## Notes on EEPMOV Instruction

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



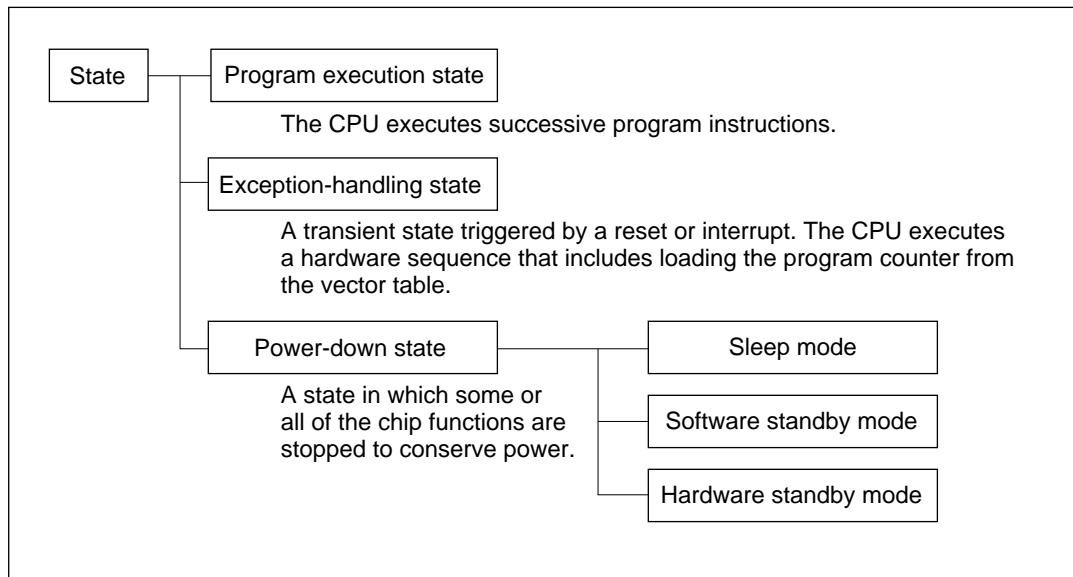
2. When setting R4L and R6, make sure that the final destination address ( $R6 + R4L$ ) does not exceed H'FFFF. The value in R6 must not change from H'FFFF to H'0000 during execution of the instruction.



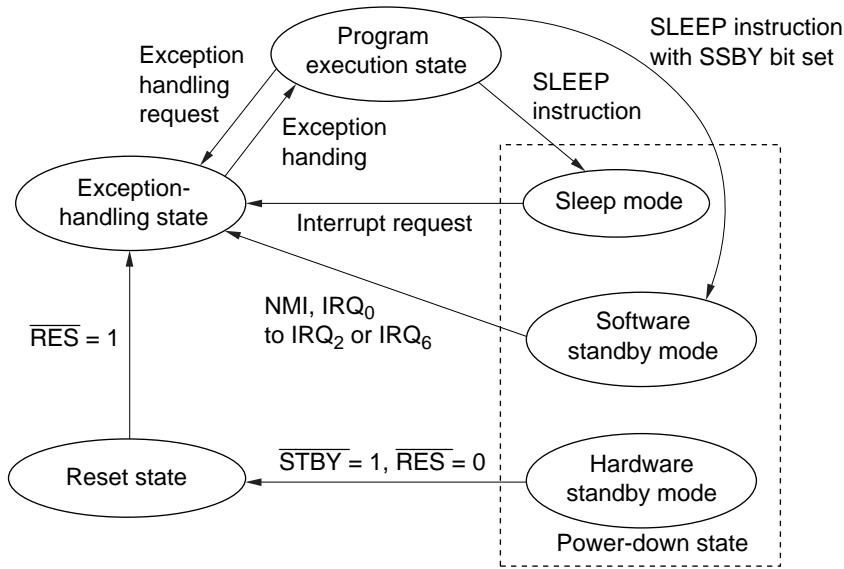
## 2.6 CPU States

### 2.6.1 Overview

The CPU has three states: the program execution state, exception-handling state, and power-down state. The power-down state is further divided into three modes: sleep mode, software standby mode, and hardware standby mode. Figure 2-11 summarizes these states, and figure 2-12 shows a map of the state transitions.



**Figure 2-11 Operating States**



Notes:

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

**Figure 2-12 State Transitions**

## 2.6.2 Program Execution State

In this state the CPU executes program instructions.

## 2.6.3 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU is reset or interrupted and changes its normal processing flow. In interrupt exception handling, the CPU references the stack pointer (R7) and saves the program counter and condition code register on the stack. For further details see section 4, Exception Handling.

## 2.6.4 Power-Down State

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

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

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

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

See section 21, Power-Down State, for further information.

## 2.7 Access Timing and Bus Cycle

The CPU is driven by the system clock ( $\phi$ ). The period from one rising edge of the system clock to the next is referred to as a “state.” Memory access is performed in a two- or three-state bus cycle. On-chip memory, on-chip supporting modules, and external devices are accessed in different bus cycles as described below.

### 2.7.1 Access to On-Chip Memory (RAM and ROM)

On-chip ROM and RAM are accessed in a cycle of two states designated  $T_1$  and  $T_2$ . Either byte or word data can be accessed, via a 16-bit data bus. Figure 2-13 shows the on-chip memory access cycle. Figure 2-14 shows the associated pin states.

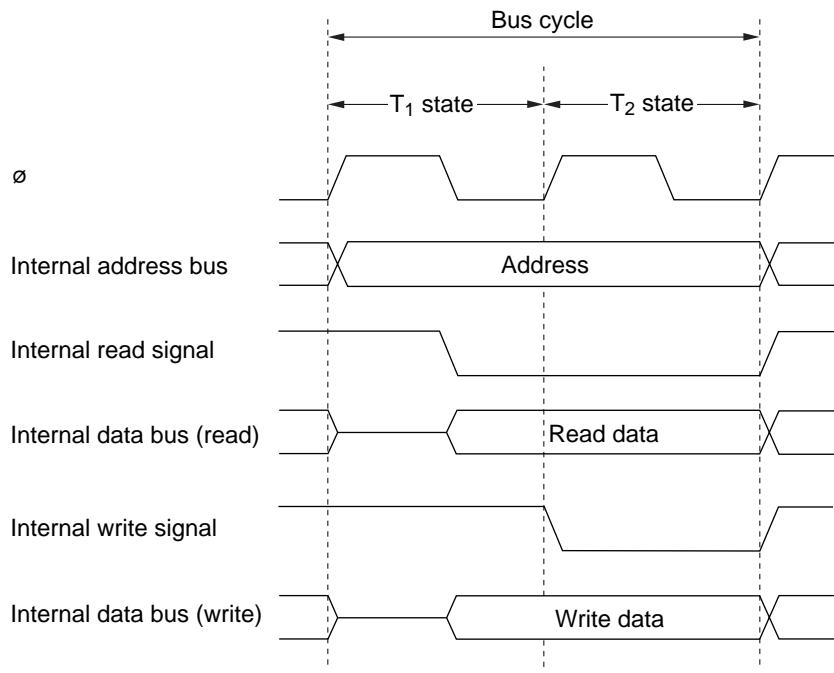


Figure 2-13 On-Chip Memory Access Cycle

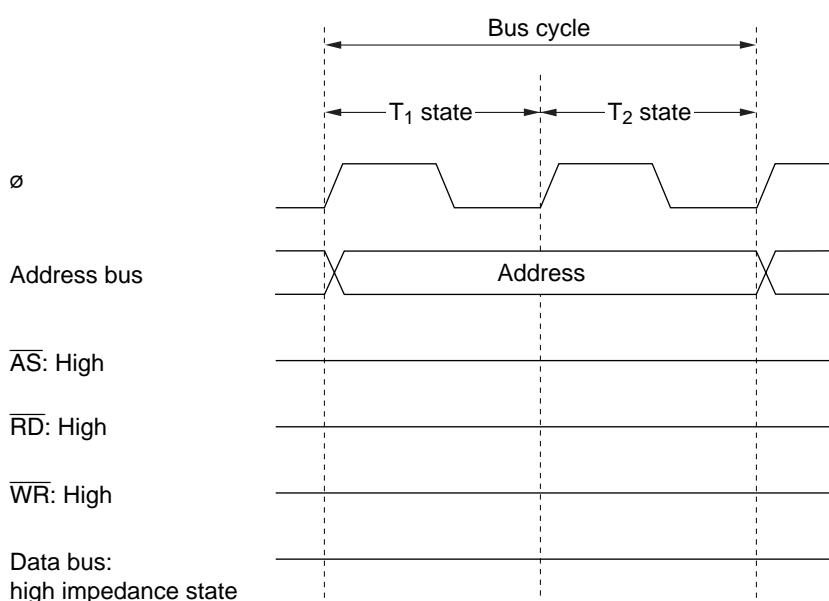


Figure 2-14 Pin States during On-Chip Memory Access Cycle

## 2.7.2 Access to On-Chip Register Field and External Devices

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

Figure 2-15 shows the access cycle for the on-chip register field. Figure 2-16 shows the associated pin states. Figures 2-17 (a) and (b) show the read and write access timing for external devices.

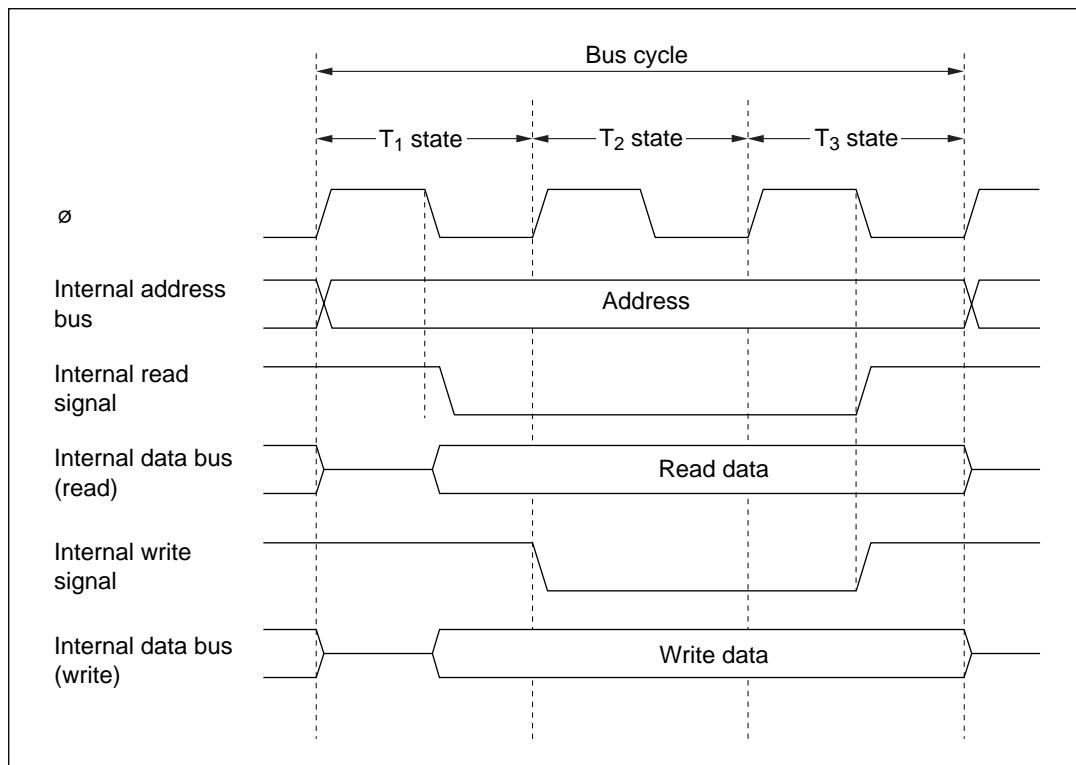


Figure 2-15 On-Chip Register Field Access Cycle

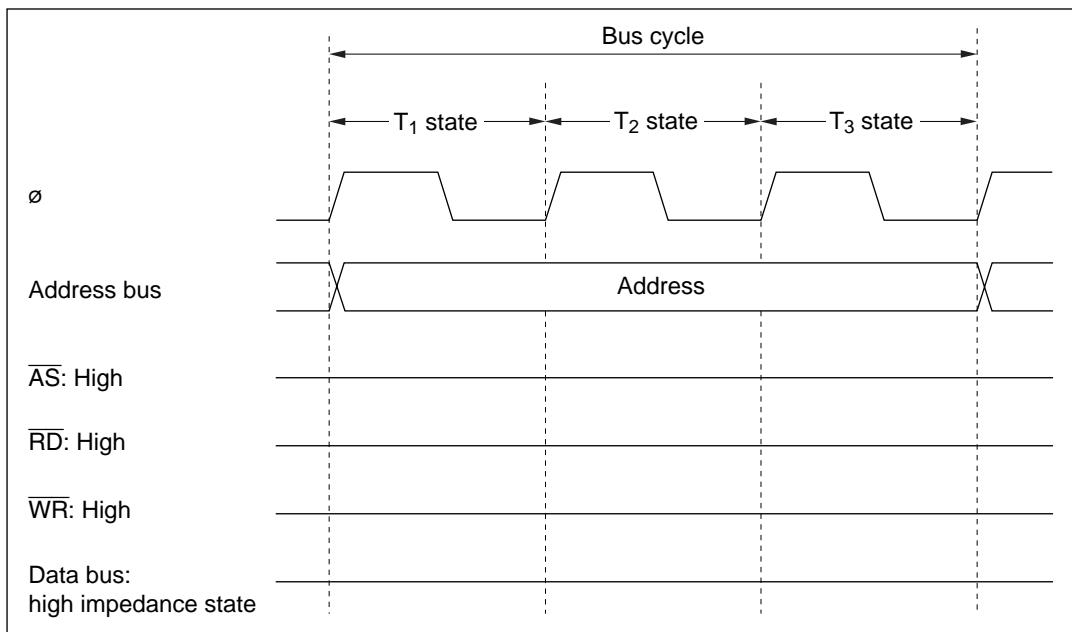


Figure 2-16 Pin States during On-Chip Register Field Access Cycle

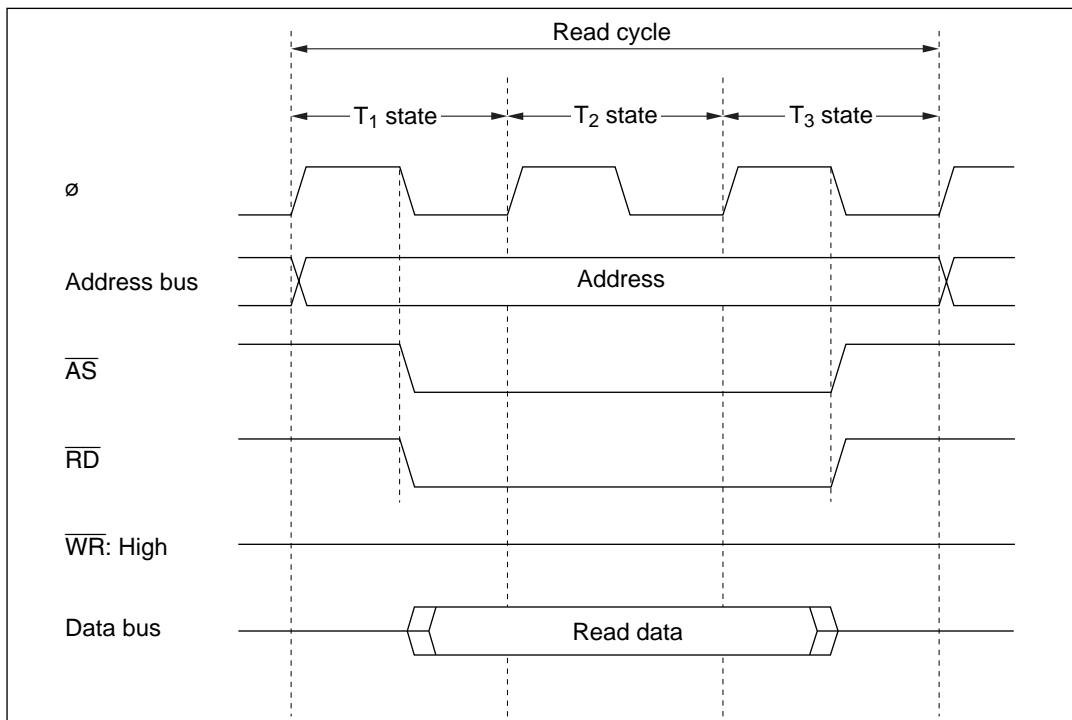
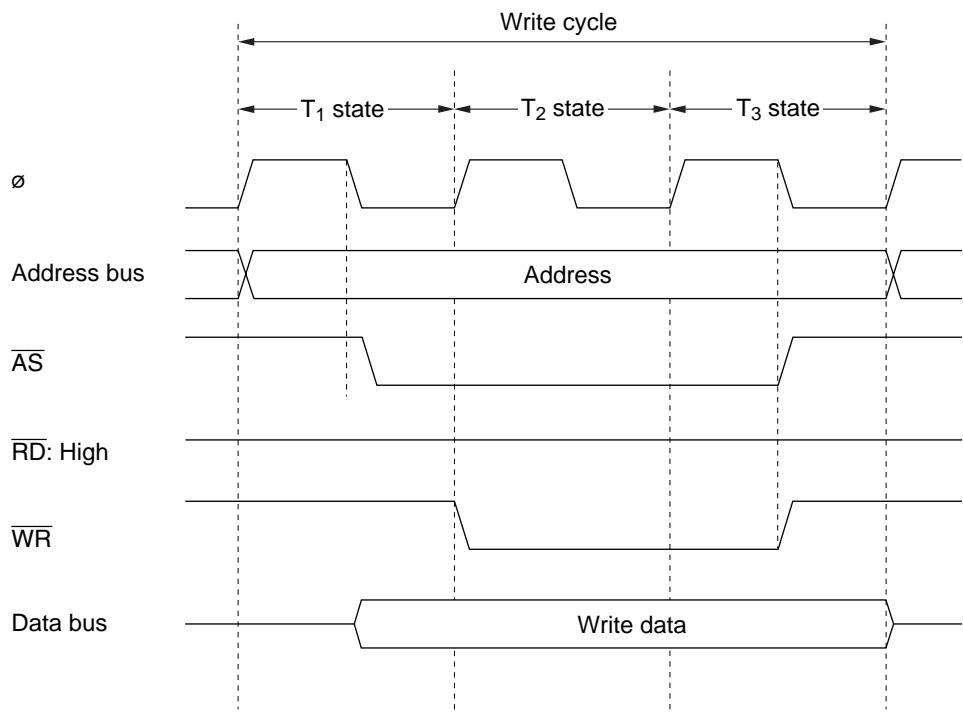


Figure 2-17 (a) External Device Access Timing (Read)



**Figure 2-17 (b) External Device Access Timing (Write)**

# Section 3 MCU Operating Modes and Address Space

## 3.1 Overview

### 3.1.1 Mode Selection

The H8/3437 Series operates in three modes numbered 1, 2, and 3. The mode is selected by the inputs at the mode pins (MD<sub>1</sub> and MD<sub>0</sub>). See table 3-1.

**Table 3-1 Operating Modes**

Mode	MD <sub>1</sub>	MD <sub>0</sub>	Address space	On-chip ROM	On-chip RAM
Mode 0	Low	Low	—	—	—
Mode 1	Low	High	Expanded	Disabled	Enabled*
Mode 2	High	Low	Expanded	Enabled	Enabled*
Mode 3	High	High	Single-chip	Enabled	Enabled

Note: \* If the RAME bit in the system control register (SYSCR) is cleared to 0, off-chip memory can be accessed instead.

Modes 1 and 2 are expanded modes that permit access to off-chip memory and peripheral devices. The maximum address space supported by these externally expanded modes is 64 kbytes.

In mode 3 (single-chip mode), only on-chip ROM and RAM and the on-chip register field are used. All ports are available for general-purpose input and output.

Mode 0 is inoperative in the H8/3437 Series. Avoid setting the mode pins to mode 0.

### 3.1.2 Mode and System Control Registers

Table 3-2 lists the registers related to the chip's operating mode: the system control register (SYSCR) and mode control register (MDCR). The mode control register indicates the inputs to the mode pins MD<sub>1</sub> and MD<sub>0</sub>.

**Table 3-2 Mode and System Control Registers**

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

## 3.2 System Control Register (SYSCR)

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

The system control register (SYSCR) is an 8-bit register that controls the operation of the chip.

**Bit 7—Software Standby (SSBY):** Enables transition to the software standby mode. For details, see section 21, Power-Down State.

On recovery from software standby mode by an external interrupt, the SSBY bit remains set to 1. It can be cleared by writing 0.

Bit 7 SSBY	Description	
0	The SLEEP instruction causes a transition to sleep mode.	(Initial value)
1	The SLEEP instruction causes a transition to software standby mode.	

**Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0):** These bits select the clock settling time when the chip recovers from the software standby mode by an external interrupt. During the selected time the CPU and on-chip supporting modules continue to stand by. These bits should be set according to the clock frequency so that the settling time is at least 8 ms. For specific settings, see section 21.3.3, Clock Settling Time for Exit from Software Standby Mode.

### ZTAT and Mask ROM Versions

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8,192 states	(Initial value)
0	0	1	Settling time = 16,384 states	
0	1	0	Settling time = 32,768 states	
0	1	1	Settling time = 65,536 states	
1	0	—	Settling time = 131,072 states	
1	1	—	Unused	

### F-ZTAT Version

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8,192 states	(Initial value)
0	0	1	Settling time = 16,384 states	
0	1	0	Settling time = 32,768 states	
0	1	1	Settling time = 65,536 states	
1	0	0	Settling time = 131,072 states	
1	0	1	Settling time = 1,024 states	
1	1	—	Unused	

Note: When 1,024 states (STS2 to STS0 = 101) is selected, the following points should be noted.

If a period exceeding  $\varnothing p/1,024$  (e.g.  $\varnothing p/2,048$ ) is specified when selecting the 8-bit timer, PWM timer, or watchdog timer clock, the counter in the timer will not count up normally when 1,024 states is specified for the settling time. To avoid this problem, set the STS value just before the transition to software standby mode (before executing the SLEEP instruction), and re-set the value of STS2 to STS0 to a value from 000 to 100 directly after software standby mode is cleared by an interrupt.

**Bit 3—External Reset (XRST):** Indicates the source of a reset. A reset can be generated by input of an external reset signal, or by a watchdog timer overflow when the watchdog timer is used. XRST is a read-only bit. It is set to 1 by an external reset, and cleared to 0 by watchdog timer overflow.

**Bit 3**

XRST	Description	
0	Reset was caused by watchdog timer overflow.	
1	Reset was caused by external input.	(Initial value)

**Bit 2—NMI Edge (NMIEG):** Selects the valid edge of the NMI input.

**Bit 2**

NMIEG	Description	
0	An interrupt is requested on the falling edge of the $\overline{NMI}$ input.	(Initial value)
1	An interrupt is requested on the rising edge of the $\overline{NMI}$ input.	

**Bit 1—Host Interface Enable (HIE):** Enables or disables the host interface function. When enabled, the host interface processes host-slave data transfers, operating in slave mode.

**Bit 1**

HIE	Description	
0	The host interface is disabled.	(Initial value)
1	The host interface is enabled (slave mode).	

**Bit 0—RAM Enable (RAME):** Enables or disables the on-chip RAM. The RAME bit is initialized by a reset, but is not initialized in the software standby mode.

**Bit 0**

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

### 3.3 Mode Control Register (MDCR)

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

Note: \* Initialized according to MD<sub>1</sub> and MD<sub>0</sub> inputs.

The mode control register (MDCR) is an 8-bit register that indicates the operating mode of the chip.

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

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

**Bit 2—Reserved:** This bit cannot be modified and is always read as 1.

**Bits 1 and 0—Mode Select 1 and 0 (MDS1 and MDS0):** These bits indicate the values of the mode pins (MD<sub>1</sub> and MD<sub>0</sub>), thereby indicating the current operating mode of the chip. MDS1 corresponds to MD<sub>1</sub> and MDS0 to MD<sub>0</sub>. These bits can be read but not written. When the mode control register is read, the levels at the mode pins (MD<sub>1</sub> and MD<sub>0</sub>) are latched in these bits.

### 3.4 Address Space Map in Each Operating Mode

Figures 3-1 and 3-2 show memory maps of the H8/3437, H8/3436, and H8/3434 in modes 1, 2, and 3.

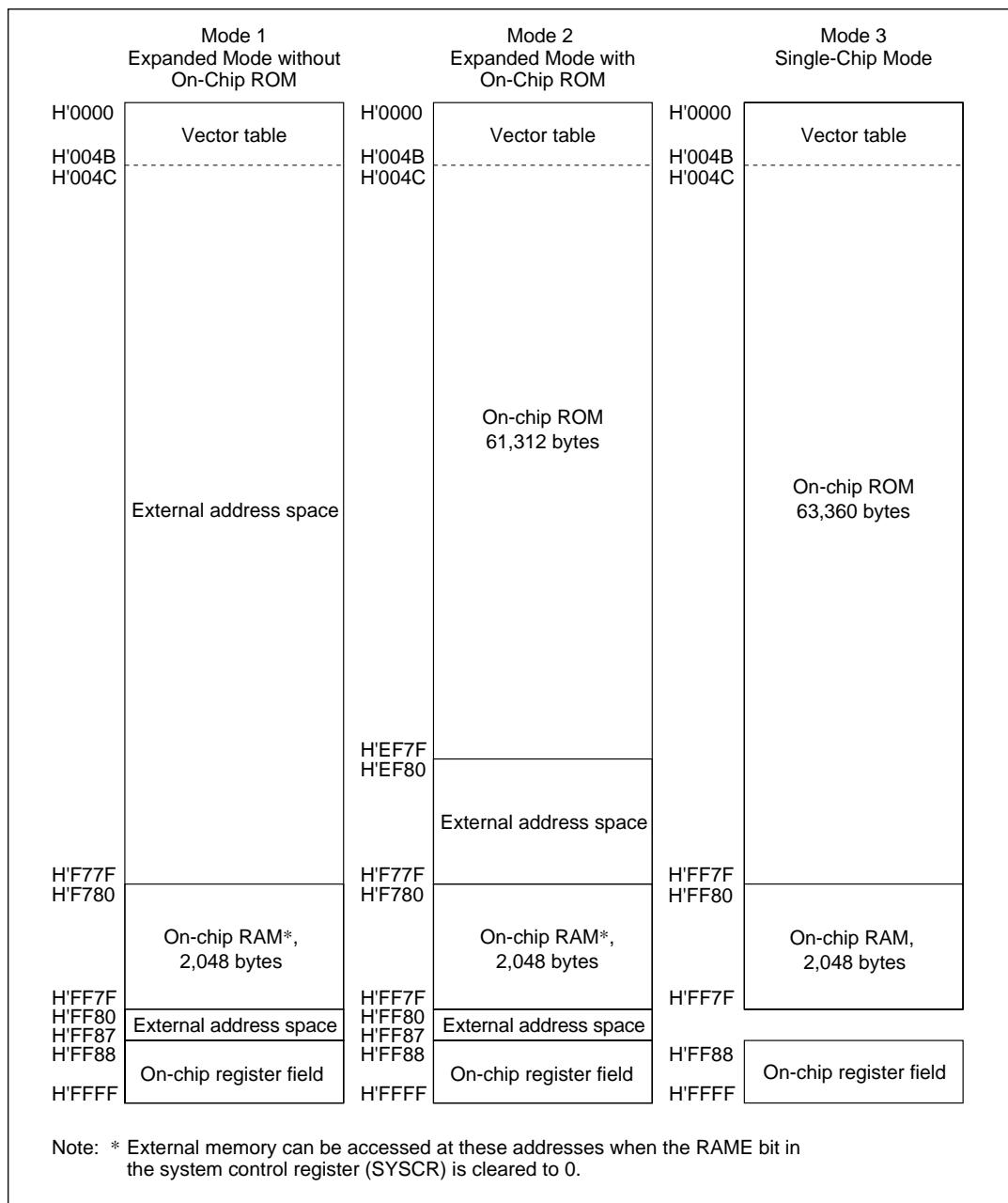
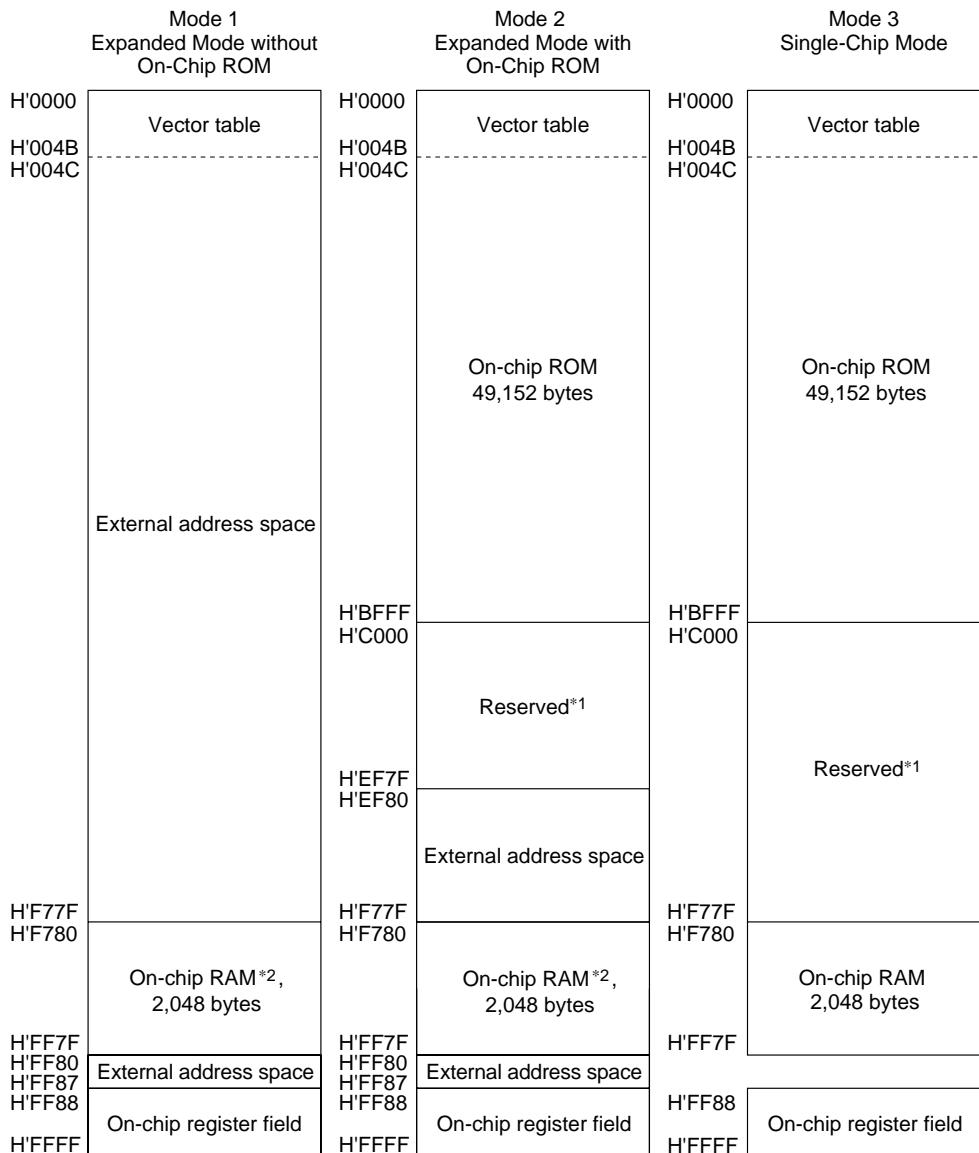


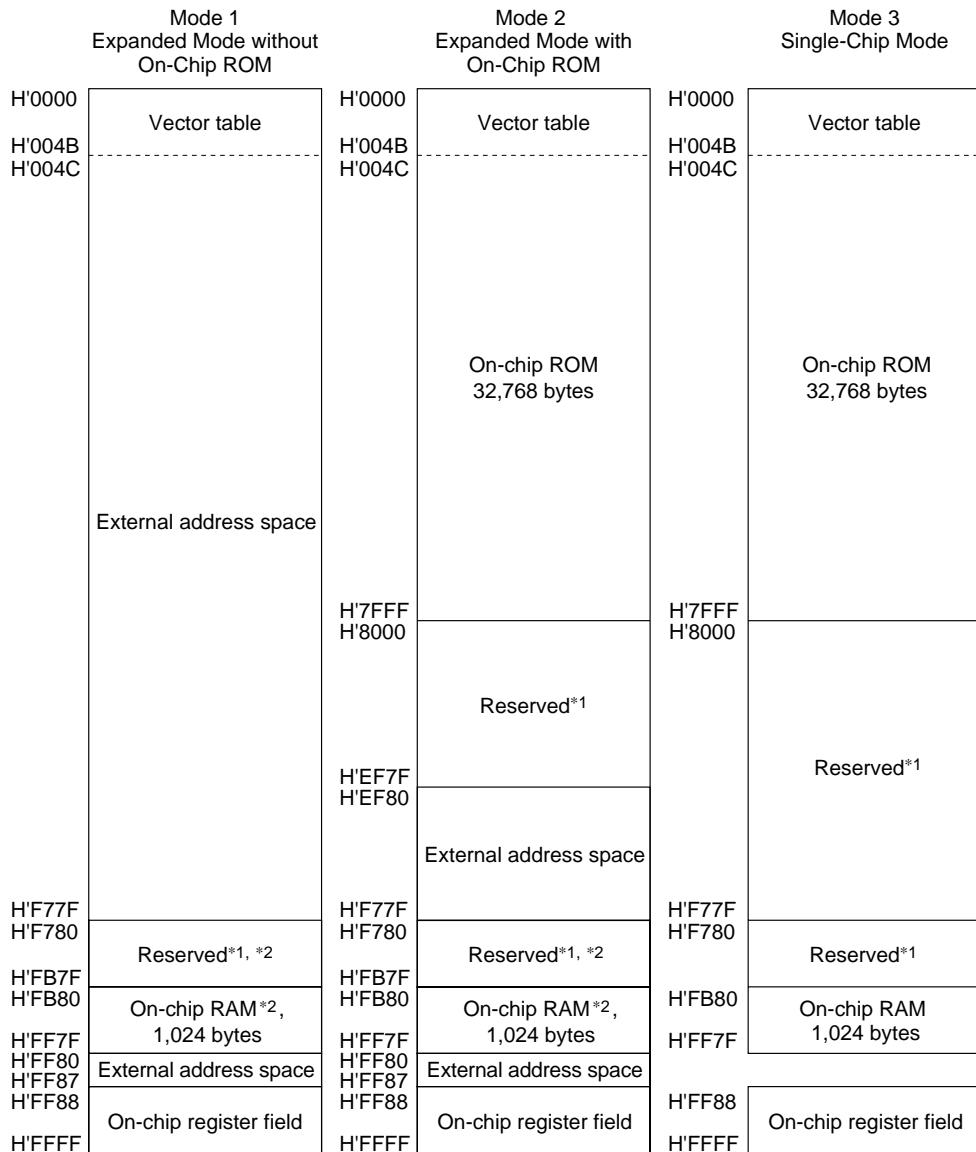
Figure 3-1 H8/3437 Address Space Map



Notes: \*1 Do not access reserved areas.

\*2 External memory can be accessed at these addresses when the RAME bit in the system control register (SYSCR) is cleared to 0.

Figure 3-2 H8/3436 Address Space Map



Notes: \*1 Do not access reserved areas.

\*2 External memory can be accessed at these addresses when the RAME bit in the system control register (SYSCR) is cleared to 0.

Figure 3-3 H8/3434 Address Space Map

# Section 4 Exception Handling

## 4.1 Overview

The H8/3437 Series recognizes two kinds of exceptions: interrupts and the reset. Table 4-1 indicates their priority and the timing of their hardware exception-handling sequence.

**Table 4-1 Hardware Exception-Handling Sequences and Priority**

Priority	Type of Exception	Detection Timing	Timing of Exception-Handling Sequence
High 	Reset	Synchronized with clock	The hardware exception-handling sequence begins as soon as <u>RES</u> changes from low to high.
	Interrupt	End of instruction execution*	When an interrupt is requested, the hardware exception-handling sequence begins at the end of the current instruction, or at the end of the current hardware exception-handling sequence.

Note: \* Not detected after ANDC, ORC, XORC, and LDC instructions.

## 4.2 Reset

### 4.2.1 Overview

A reset has the highest exception-handling priority. When the RES pin goes low or when there is a watchdog timer reset (when the reset option is selected for watchdog timer overflow), all current processing stops and the chip enters the reset state. The internal state of the CPU and the registers of the on-chip supporting modules are initialized. The reset exception-handling sequence starts when RES returns from low to high, or at the end of a watchdog reset pulse.

### 4.2.2 Reset Sequence

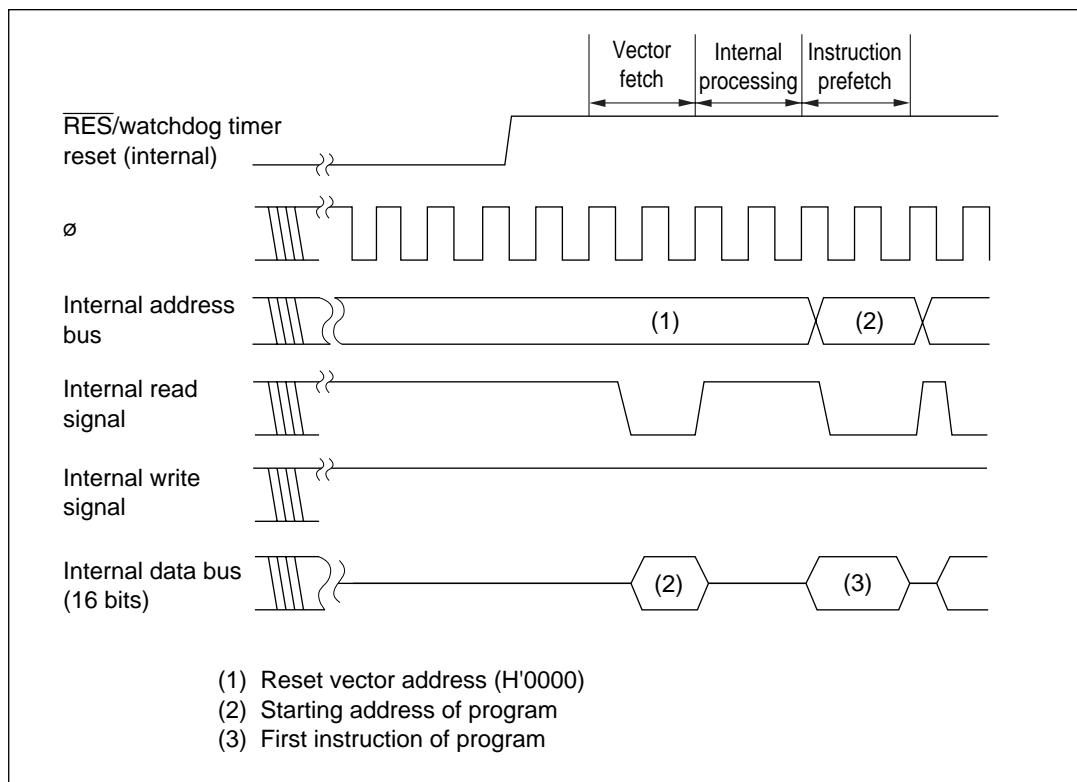
The reset state begins when RES goes low or a watchdog reset is generated. To ensure correct resetting, at power-on the RES pin should be held low for at least 20 ms. In a reset during operation, the RES pin should be held low for at least 10 system clock cycles. The watchdog reset pulse width is always 518 system clocks. For the pin states during a reset, see appendix D, Pin States.

The following sequence is carried out when reset exception handling begins.

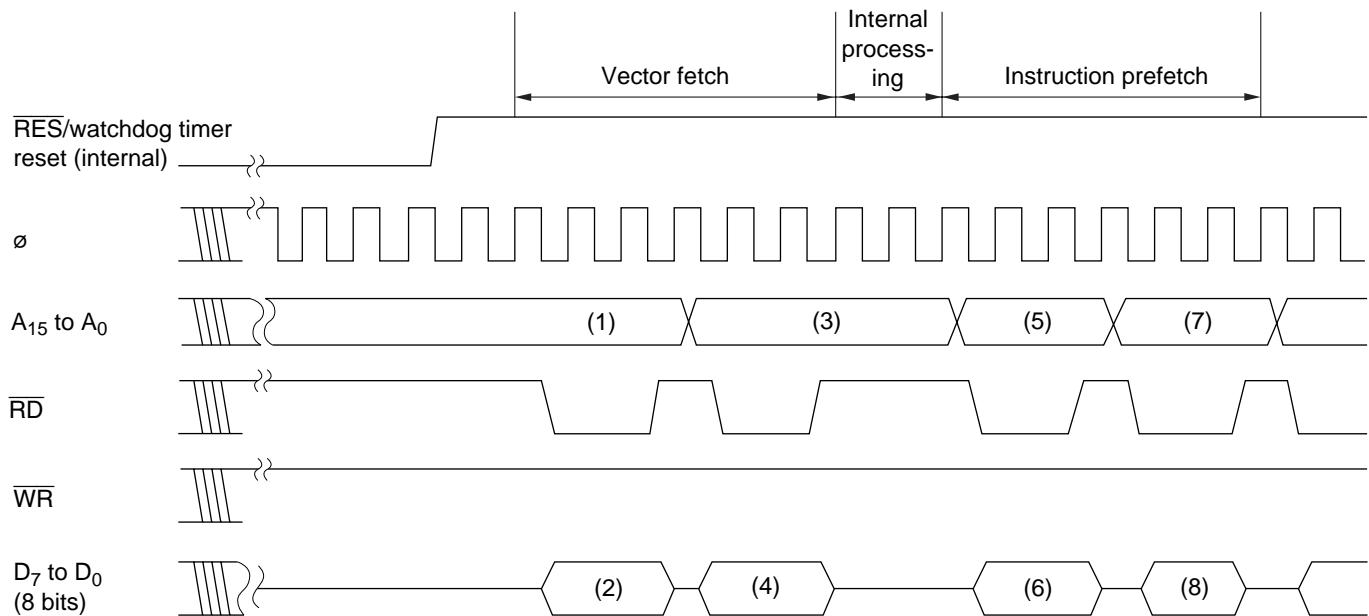
- (1) The internal state of the CPU and the registers of the on-chip supporting modules are initialized, and the I bit in the condition code register (CCR) is set to 1.
- (2) The CPU loads the program counter with the first word in the vector table (stored at addresses H'0000 and H'0001) and starts program execution.

The RES pin should be held low when power is switched off, as well as when power is switched on.

Figure 4-1 indicates the timing of the reset sequence in modes 2 and 3. Figure 4-2 indicates the timing in mode 1.



**Figure 4-1 Reset Sequence (Mode 2 or 3, Program Stored in On-Chip ROM)**



(1), (3) Reset vector address: (1) = H'0000, (3) = H'0001  
 (2), (4) Starting address of program (contents of reset vector): (2) = upper byte, (4) = lower byte  
 (5), (7) Starting address of program: (5) = (2) (4), (7) = (2) (4) + 1  
 (6), (8) First instruction of program: (6) = first byte, (8) = second byte

Figure 4-2 Reset Sequence (Mode 1)

### 4.2.3 Disabling of Interrupts after Reset

After a reset, if an interrupt were to be accepted before initialization of the stack pointer (SP: R7), the program counter and condition code register might not be saved correctly, leading to a program crash. To prevent this, all interrupts, including NMI, are disabled immediately after a reset. The first program instruction is therefore always executed. This instruction should initialize the stack pointer (example: `MOV.W #xx:16, SP`).

After reset exception handling, in order to initialize the contents of CCR, a CCR manipulation instruction can be executed before an instruction to initialize the stack pointer. Immediately after execution of a CCR manipulation instruction, all interrupts including NMI are disabled. Use the next instruction to initialize the stack pointer.

## 4.3 Interrupts

### 4.3.1 Overview

The interrupt sources include nine external sources from 23 input pins (NMI,  $\text{IRQ}_0$  to  $\text{IRQ}_7$ , and  $\text{KEYIN}_0$  to  $\text{KEYIN}_{15}$ ), and 26 internal sources in the on-chip supporting modules. Table 4-2 lists the interrupt sources in priority order and gives their vector addresses. When two or more interrupts are requested, the interrupt with highest priority is served first.

The features of these interrupts are:

- NMI has the highest priority and is always accepted. All internal and external interrupts except NMI can be masked by the I bit in the CCR. When the I bit is set to 1, interrupts other than NMI are not accepted.
- $\text{IRQ}_0$  to  $\text{IRQ}_7$  can be sensed on the falling edge of the input signal, or level-sensed. The type of sensing can be selected for each interrupt individually. NMI is edge-sensed, and either the rising or falling edge can be selected.
- All interrupts are individually vectored. The software interrupt-handling routine does not have to determine what type of interrupt has occurred.
- $\text{IRQ}_6$  is multiplexed with 16 external sources ( $\text{KEYIN}_0$  to  $\text{KEYIN}_{15}$ ).  $\text{KEYIN}_0$  to  $\text{KEYIN}_{15}$  can be masked individually by user software.
- The watchdog timer can generate either an NMI or overflow interrupt, depending on the needs of the application. For details, see section 11, Watchdog Timer.

**Table 4-2 Interrupts**

Interrupt source	No.	Vector Table Address	Priority
NMI	3	H'0006 to H'0007	High
IRQ0	4	H'0008 to H'0009	
IRQ1	5	H'000A to H'000B	
IRQ2	6	H'000C to H'000D	
IRQ3	7	H'000E to H'000F	
IRQ4	8	H'0010 to H'0011	
IRQ5	9	H'0012 to H'0013	
IRQ6	10	H'0014 to H'0015	
IRQ7	11	H'0016 to H'0017	
16-bit free-running timer	ICIA (Input capture A)	12	H'0018 to H'0019
	ICIB (Input capture B)	13	H'001A to H'001B
	ICIC (Input capture C)	14	H'001C to H'001D
	ICID (Input capture D)	15	H'001E to H'001F
	OCIA (Output compare A)	16	H'0020 to H'0021
	OCIB (Output compare B)	17	H'0022 to H'0023
	FOVI (Overflow)	18	H'0024 to H'0025
8-bit timer 0	CMI0A (Compare-match A)	19	H'0026 to H'0027
	CMI0B (Compare-match B)	20	H'0028 to H'0029
	OVI0 (Overflow)	21	H'002A to H'002B
8-bit timer 1	CMI1A (Compare-match A)	22	H'002C to H'002D
	CMI1B (Compare-match B)	23	H'002E to H'002F
	OVI1 (Overflow)	24	H'0030 to H'0031
Host interface	IBF1 (IDR1 receive end)	25	H'0032 to H'0033
	IBF2 (IDR2 receive end)	26	H'0034 to H'0035
Serial communication interface 0	ERI0 (Receive error)	27	H'0036 to H'0037
	RXI0 (Receive end)	28	H'0038 to H'0039
	TXI0 (TDR empty)	29	H'003A to H'003B
	TEI0 (TSR empty)	30	H'003C to H'003D
Serial communication interface 1	ERI1 (Receive error)	31	H'003E to H'003F
	RXI1 (Receive end)	32	H'0040 to H'0041
	TXI1 (TDR empty)	33	H'0042 to H'0043
	TEI1 (TSR empty)	34	H'0044 to H'0045
A/D converter	ADI (Conversion end)	35	H'0046 to H'0047
Watchdog timer	WOVF (WDT overflow)	36	H'0048 to H'0049
I <sup>2</sup> C bus interface	IICI (Transfer end)	37	H'004A to H'004B

Notes: 1. H'0000 and H'0001 contain the reset vector.

2. H'0002 to H'0005 are reserved in the H8/3437 Series and are not available to the user.

### 4.3.2 Interrupt-Related Registers

The interrupt-related registers are the system control register (SYSCR), IRQ sense control register (ISCR), IRQ enable register (IER), and keyboard matrix interrupt mask registers (KMIMR and KMIMRA).

**Table 4-3 Registers Read by Interrupt Controller**

Name	Abbreviation	Read/write	Address
System control register	SYSCR	R/W	H'FFC4
IRQ sense control register	ISCR	R/W	H'FFC6
IRQ enable register	IER	R/W	H'FFC7
Keyboard matrix interrupt mask register	KMIMR	R/W	H'FFF1
Keyboard matrix interrupt mask register A	KMIMRA	R/W	H'FFF3

#### System Control Register (SYSCR)

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

The valid edge on the  $\overline{\text{NMI}}$  line is controlled by bit 2 (NMIEG) in the system control register.

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

#### Bit 2

##### NMIEG      Description

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

See section 3.2, System Control Register, for information on the other SYSCR bits.

#### IRQ Sense Control Register (ISCR)

Bit	7	6	5	4	3	2	1	0
	IRQ7SC	IRQ6SC	IRQ5SC	IRQ4SC	IRQ3SC	IRQ2SC	IRQ1SC	IRQ0SC
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

**Bits 7 to 0—IRQ<sub>7</sub> to IRQ<sub>0</sub> Sense Control (IRQ7SC to IRQ0SC):** These bits determine whether IRQ<sub>7</sub> to IRQ<sub>0</sub> are level-sensed or sensed on the falling edge.

Bits 7 to 0	IRQ7SC to IRQ0SC	Description	
0		An interrupt is generated when IRQ <sub>7</sub> to IRQ <sub>0</sub> inputs are low.	(Initial state)
1		An interrupt is generated by the falling edge of the IRQ <sub>7</sub> to IRQ <sub>0</sub> inputs.	

### IRQ Enable Register (IER)

Bit	7	6	5	4	3	2	1	0
	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

**Bits 7 to 0—IRQ7 to IRQ0 Enable (IRQ7E to IRQ0E):** These bits enable or disable the IRQ<sub>7</sub> to IRQ<sub>0</sub> interrupts individually.

Bits 7 to 0	IRQ7E to IRQ0E	Description	
0		IRQ <sub>7</sub> to IRQ <sub>0</sub> interrupt requests are disabled.	(Initial state)
1		IRQ <sub>7</sub> to IRQ <sub>0</sub> interrupt requests are enabled.	

When edge sensing is selected (by setting bits IRQ7SC to IRQ0SC to 1), it is possible for an interrupt-handling routine to be executed even though the corresponding enable bit (IRQ7E to IRQ0E) is cleared to 0 and the interrupt is disabled. If an interrupt is requested while the enable bit (IRQ7E to IRQ0E) is set to 1, the request will be held pending until served. If the enable bit is cleared to 0 while the request is still pending, the request will remain pending, although new requests will not be recognized. If the interrupt mask bit (I) in the CCR is cleared to 0, the interrupt-handling routine can be executed even though the enable bit is now 0.

If execution of interrupt-handling routines under these conditions is not desired, it can be avoided by using the following procedure to disable and clear interrupt requests.

1. Set the I bit to 1 in the CCR, masking interrupts. Note that the I bit is set to 1 automatically when execution jumps to an interrupt vector.
2. Clear the desired bits from IRQ7E to IRQ0E to 0 to disable new interrupt requests.
3. Clear the corresponding IRQ7SC to IRQ0SC bits to 0, then set them to 1 again. Pending IRQn interrupt requests are cleared when I = 1 in the CCR, IRQnSC = 0, and IRQnE = 0.

## Keyboard Matrix Interrupt Mask Register (KMIMR)

To control interrupts from a  $16 \times 16$  matrix keyboard at key-sense input pins  $\overline{\text{KEYIN}}_0$  to  $\overline{\text{KEYIN}}_{15}$ , there are two keyboard matrix interrupt mask registers, KMIMR and KMIMRA. Bits KMIMR7 to KMIMR0 in KMIMR correspond to key-sense inputs  $\overline{\text{KEYIN}}_7$  to  $\overline{\text{KEYIN}}_0$ . Bits KMIMR15 to KMIMR8 in KMIMRA correspond to key-sense inputs  $\overline{\text{KEYIN}}_{15}$  to  $\overline{\text{KEYIN}}_8$ . Initially, the KMIMR6 bit that corresponds to the  $\overline{\text{IRQ}}_6/\overline{\text{KEYIN}}_6$  pin is in the interrupt-enabled state, and the other interrupt mask bits are in the interrupt-disabled state.

KMIMR is an 8-bit readable/writable register used in keyboard matrix scanning and sensing. This register initializes to a state in which only the input at the  $\overline{\text{IRQ}}_6$  pin is enabled. To enable key-sense input interrupts from two or more pins during keyboard scanning and sensing, clear the corresponding mask bits to 0.

Bit	7	6	5	4	3	2	1	0
	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0
Initial value	1	0	1	1	1	1	1	1
Read/Write	R/W							

**Bits 7 to 0—Keyboard Matrix Interrupt Mask (KMIMR7 to KMIMR0):** These bits control key-sense input interrupt requests  $\overline{\text{KEYIN}}_7$  to  $\overline{\text{KEYIN}}_0$ .

### Bits 7 to 0

#### KMIMR7 to KMIMR0

#### Description

0	Key-sense input interrupt request is enabled.
1	Key-sense input interrupt request is disabled. (Initial value)*

Note: \* Except KMIMR6, which is initially 0.

Bit	7	6	5	4	3	2	1	0
	KMIMR15	KMIMR14	KMIMR13	KMIMR12	KMIMR11	KMIMR10	KMIMR9	KMIMR8
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

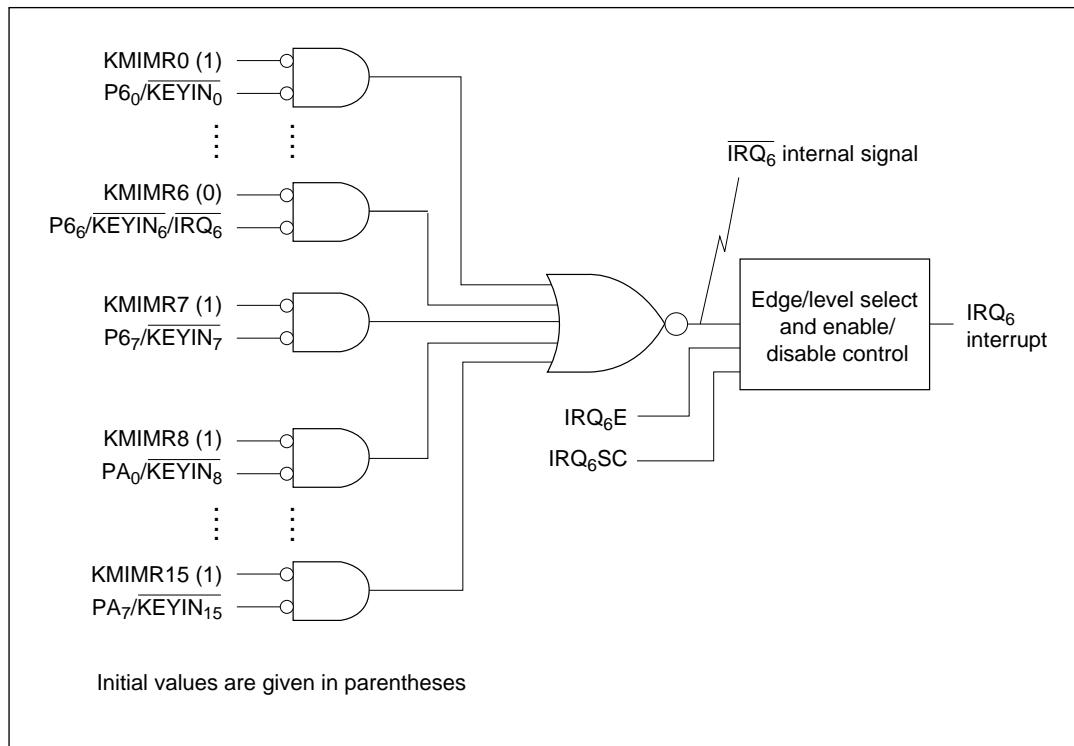
**Bits 7 to 0—Keyboard Matrix Interrupt Mask (KMIMR15 to KMIMR8):** These bits control key-sense input interrupt requests KEYIN15 to KEYIN8.

### Bits 7 to 0

#### KMIMR15 to KMIMR8      Description

0	Key-sense input interrupt request is enabled.
1	Key-sense input interrupt request is disabled. (Initial value)

Figure 4-3 shows the relationship between the  $\overline{\text{IRQ}_6}$  interrupt, KMIMR, and KMIMRA.



**Figure 4-3 KMIMR, KMIMRA, and  $\overline{\text{IRQ}_6}$  Interrupt**

### 4.3.3 External Interrupts

The nine external interrupts are NMI and IRQ<sub>0</sub> to IRQ<sub>7</sub>. NMI, IRQ<sub>0</sub>, IRQ<sub>1</sub>, IRQ<sub>2</sub>, and IRQ<sub>6</sub> can be used to recover from software standby mode.

**(1) NMI:** A nonmaskable interrupt is generated on the rising or falling edge of the  $\overline{\text{NMI}}$  input signal regardless of whether the I (interrupt mask) bit is set in the CCR. The valid edge is selected by the NMIEG bit in the system control register. The NMI vector number is 3. In the NMI hardware exception-handling sequence the I bit in the CCR is set to 1.

**(2) IRQ<sub>0</sub> to IRQ<sub>7</sub>:** These interrupt signals are level-sensed or sensed on the falling edge of the input, as selected by ISCR bits IRQ0SC to IRQ7SC. These interrupts can be masked collectively by the I bit in the CCR, and can be enabled and disabled individually by setting and clearing bits IRQ0E to IRQ7E in the IRQ enable register.

The  $\overline{\text{IRQ}_6}$  input signal can be logically ORed internally with the key sense input signals. When KEYIN<sub>0</sub> to KEYIN<sub>15</sub> pins (P6<sub>0</sub> to P6<sub>7</sub> and PA<sub>0</sub> to PA<sub>7</sub>) are used for key sense input, the corresponding KMIMR bits should be cleared to 0 to enable the corresponding key sense input interrupts. KMIMR bits corresponding to unused key sense inputs should be set to 1 to disable the interrupts. All 16 key sense interrupts are combined into a single IRQ<sub>6</sub> interrupt.

When one of these interrupts is accepted, the I bit is set to 1. IRQ<sub>0</sub> to IRQ<sub>7</sub> have interrupt vector numbers 4 to 11. They are prioritized in order from IRQ<sub>7</sub> (low) to IRQ<sub>0</sub> (high). For details, see table 4-2.

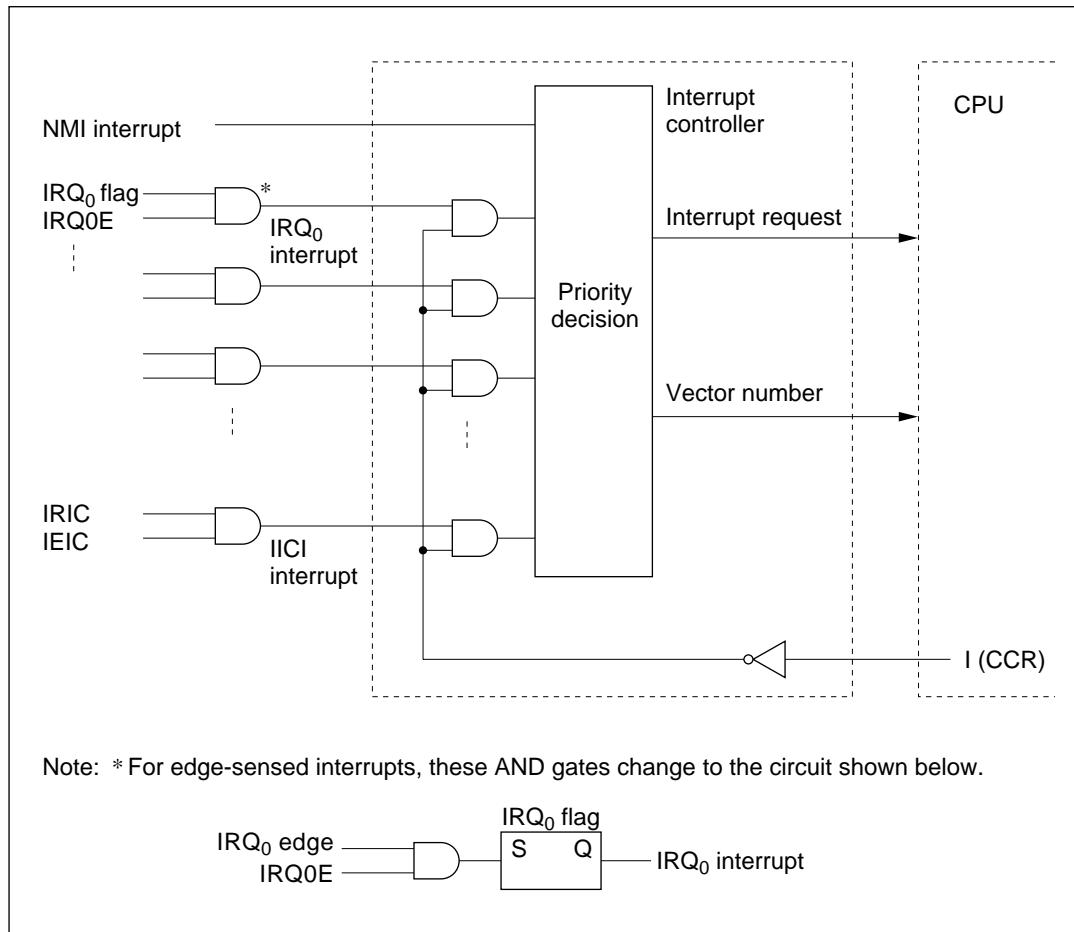
Interrupts IRQ<sub>0</sub> to IRQ<sub>7</sub> do not depend on whether pins  $\overline{\text{IRQ}_0}$  to  $\overline{\text{IRQ}_7}$  are input or output pins. When using external interrupts IRQ<sub>0</sub> to IRQ<sub>7</sub>, clear the corresponding DDR bits to 0 to set these pins to the input state, and do not use these pins as input or output pins for the timers, serial communication interface, I<sup>2</sup>C bus interface, host interface, or A/D converter.

### 4.3.4 Internal Interrupts

Twenty-six internal interrupts can be requested by the on-chip supporting modules. Each interrupt source has its own vector number, so the interrupt-handling routine does not have to determine which interrupt has occurred. All internal interrupts are masked when the I bit in the CCR is set to 1. When one of these interrupts is accepted, the I bit is set to 1 to mask further interrupts (except NMI). The vector numbers are 12 to 37. For the priority order, see table 4-2.

### 4.3.5 Interrupt Handling

Interrupts are controlled by an interrupt controller that arbitrates between simultaneous interrupt requests, commands the CPU to start the hardware interrupt exception-handling sequence, and furnishes the necessary vector number. Figure 4-4 shows a block diagram of the interrupt controller.



**Figure 4-4 Block Diagram of Interrupt Controller**

The IRQ interrupts and interrupts from the on-chip supporting modules (except for reset selected for a watchdog timer overflow) all have corresponding enable bits. When the enable bit is cleared to 0, the interrupt signal is not sent to the interrupt controller, so the interrupt is ignored. These interrupts can also all be masked by setting the CPU's interrupt mask bit (I) to 1. Accordingly, these interrupts are accepted only when their enable bit is set to 1 and the I bit is cleared to 0.

The nonmaskable interrupt (NMI) is always accepted, except in the reset state and hardware standby mode.

When an NMI or another enabled interrupt is requested, the interrupt controller transfers the interrupt request to the CPU and indicates the corresponding vector number. (When two or more interrupts are requested, the interrupt controller selects the vector number of the interrupt with the highest priority.) When notified of an interrupt request, at the end of the current instruction or current hardware exception-handling sequence, the CPU starts the hardware exception-handling sequence for the interrupt and latches the vector number.

Figure 4-5 is a flowchart of the interrupt (and reset) operations. Figure 4-7 shows the interrupt timing sequence for the case in which the software interrupt-handling routine is in on-chip ROM and the stack is in on-chip RAM.

- (1) An interrupt request is sent to the interrupt controller when an NMI interrupt occurs, and when an interrupt occurs on an IRQ input line or in an on-chip supporting module provided the enable bit of that interrupt is set to 1.
- (2) The interrupt controller checks the I bit in CCR and accepts the interrupt request if the I bit is cleared to 0. If the I bit is set to 1 only NMI requests are accepted; other interrupt requests remain pending.
- (3) Among all accepted interrupt requests, the interrupt controller selects the request with the highest priority and passes it to the CPU. Other interrupt requests remain pending.
- (4) When it receives the interrupt request, the CPU waits until completion of the current instruction or hardware exception-handling sequence, then starts the hardware exception-handling sequence for the interrupt and latches the interrupt vector number.
- (5) In the hardware exception-handling sequence, the CPU first pushes the PC and CCR onto the stack. See figure 4-6. The stacked PC indicates the address of the first instruction that will be executed on return from the software interrupt-handling routine.
- (6) Next the I bit in CCR is set to 1, masking all further interrupts except NMI.
- (7) The vector address corresponding to the vector number is generated, the vector table entry at this vector address is loaded into the program counter, and execution branches to the software interrupt-handling routine at the address indicated by that entry.

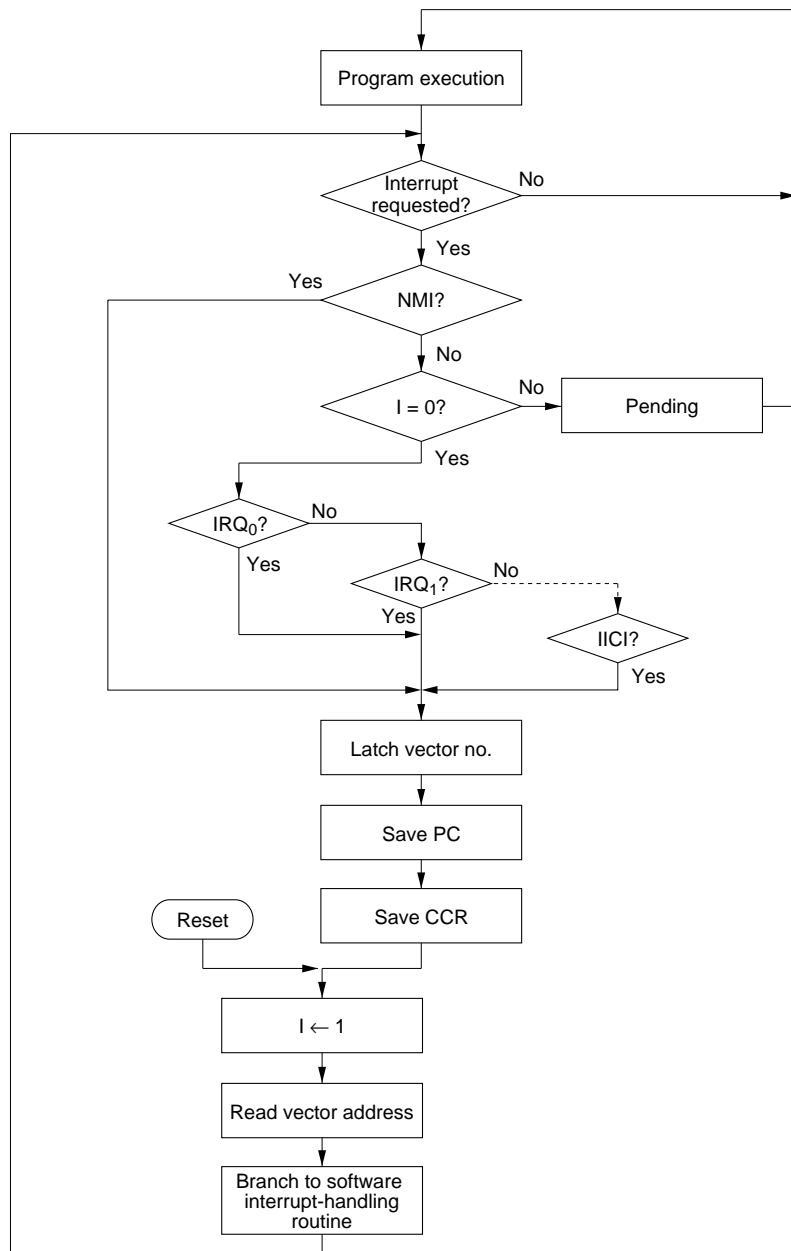
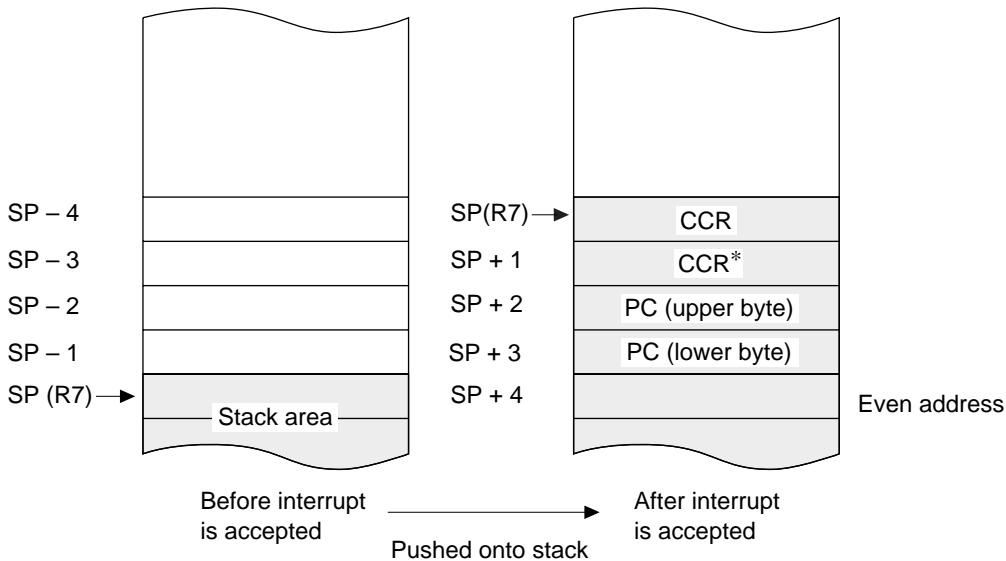


Figure 4-5 Hardware Interrupt-Handling Sequence



PC: Program counter

CCR: Condition code register

SP: Stack pointer

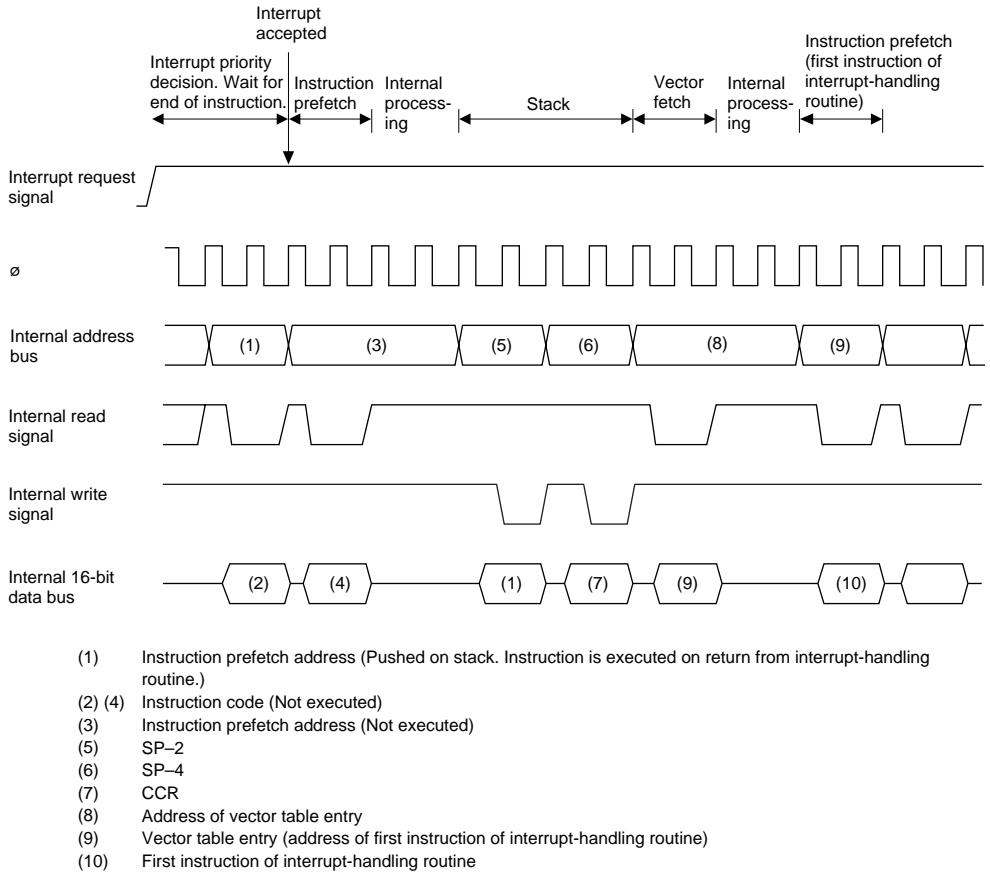
Notes:

1. The PC contains the address of the first instruction executed after return.
2. Registers must be saved and restored by word access at an even address.

\* Ignored on return.

**Figure 4-6 Usage of Stack in Interrupt Handling**

The CCR is comprised of one byte, but when it is saved to the stack, it is treated as one word of data. During interrupt processing, two identical bytes of CCR data are saved to the stack to create one word of data. When the RTE instruction is executed to restore the value from the stack, the byte located at the even address is loaded into CCR, and the byte located at the odd address is ignored.



**Figure 4-7 Timing of Interrupt Sequence**

#### 4.3.6 Interrupt Response Time

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

**Table 4-4 Number of States before Interrupt Service**

No.	Reason for Wait	Number of States	
		On-Chip Memory	External Memory
1	Interrupt priority decision	2 <sup>*3</sup>	2 <sup>*3</sup>
2	Wait for completion of current instruction <sup>*1</sup>	1 to 13	5 to 17 <sup>*2</sup>
3	Save PC and CCR	4	12 <sup>*2</sup>
4	Fetch vector	2	6 <sup>*2</sup>
5	Fetch instruction	4	12 <sup>*2</sup>
6	Internal processing	4	4
Total		17 to 29	41 to 53 <sup>*2</sup>

Notes: 1. These values do not apply if the current instruction is EEPMOV.  
2. If wait states are inserted in external memory access, add the number of wait states.  
3. 1 for internal interrupts.

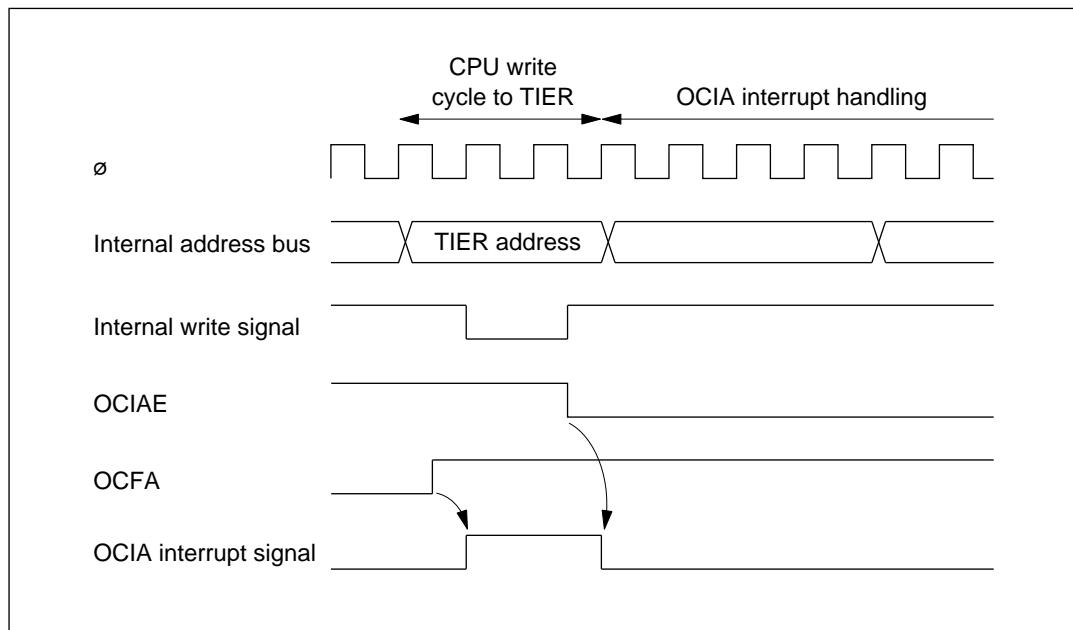
#### 4.3.7 Precaution

Note that the following type of contention can occur in interrupt handling.

When software clears the enable bit of an interrupt to 0 to disable the interrupt, the interrupt becomes disabled after execution of the clearing instruction. If an enable bit is cleared by a BCLR or MOV instruction, for example, and the interrupt is requested during execution of that instruction, at the instant when the instruction ends the interrupt is still enabled, so after execution of the instruction, the hardware exception-handling sequence is executed for the interrupt. If a higher-priority interrupt is requested at the same time, however, the hardware exception-handling sequence is executed for the higher-priority interrupt and the interrupt that was disabled is ignored.

Similar considerations apply when an interrupt request flag is cleared to 0.

Figure 4-8 shows an example in which the OCIAE bit is cleared to 0.



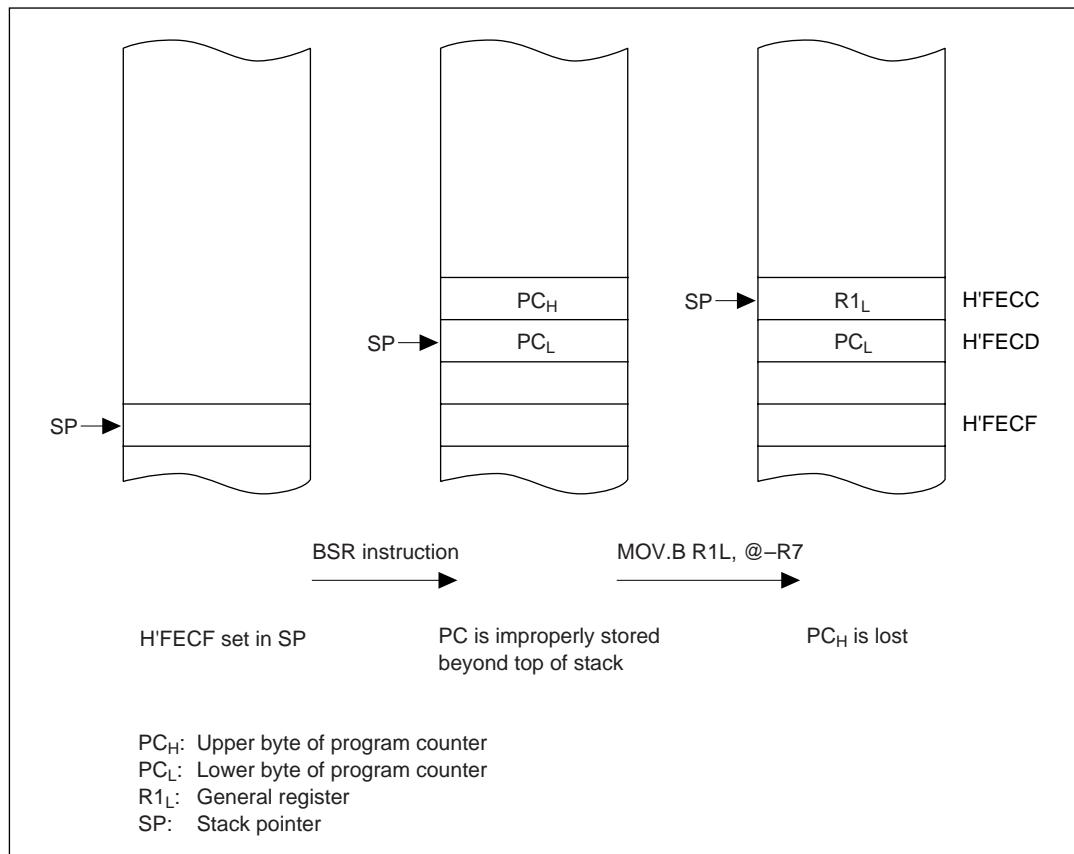
**Figure 4-8 Contention between Interrupt and Disabling Instruction**

The above contention does not occur if the enable bit or flag is cleared to 0 while the interrupt mask bit (I) is set to 1.

## 4.4 Note on Stack Handling

In word access, the least significant bit of the address is always assumed to be 0. The stack is always accessed by word access. Care should be taken to keep an even value in the stack pointer (general register R7). Use the PUSH and POP (or MOV.W Rn, @-SP and MOV.W @SP+, Rn) instructions to push and pop registers on the stack.

Setting the stack pointer to an odd value can cause programs to crash. Figure 4-9 shows an example of damage caused when the stack pointer contains an odd address.



**Figure 4-9 Example of Damage Caused by Setting an Odd Address in R7**

# Section 5 Wait-State Controller

## 5.1 Overview

The H8/3437 Series has an on-chip wait-state controller that enables insertion of wait states into bus cycles for interfacing to low-speed external devices.

### 5.1.1 Features

Features of the wait-state controller are listed below.

- Three selectable wait modes: programmable wait mode, pin auto-wait mode, and pin wait mode
- Automatic insertion of zero to three wait states

### 5.1.2 Block Diagram

Figure 5-1 shows a block diagram of the wait-state controller.

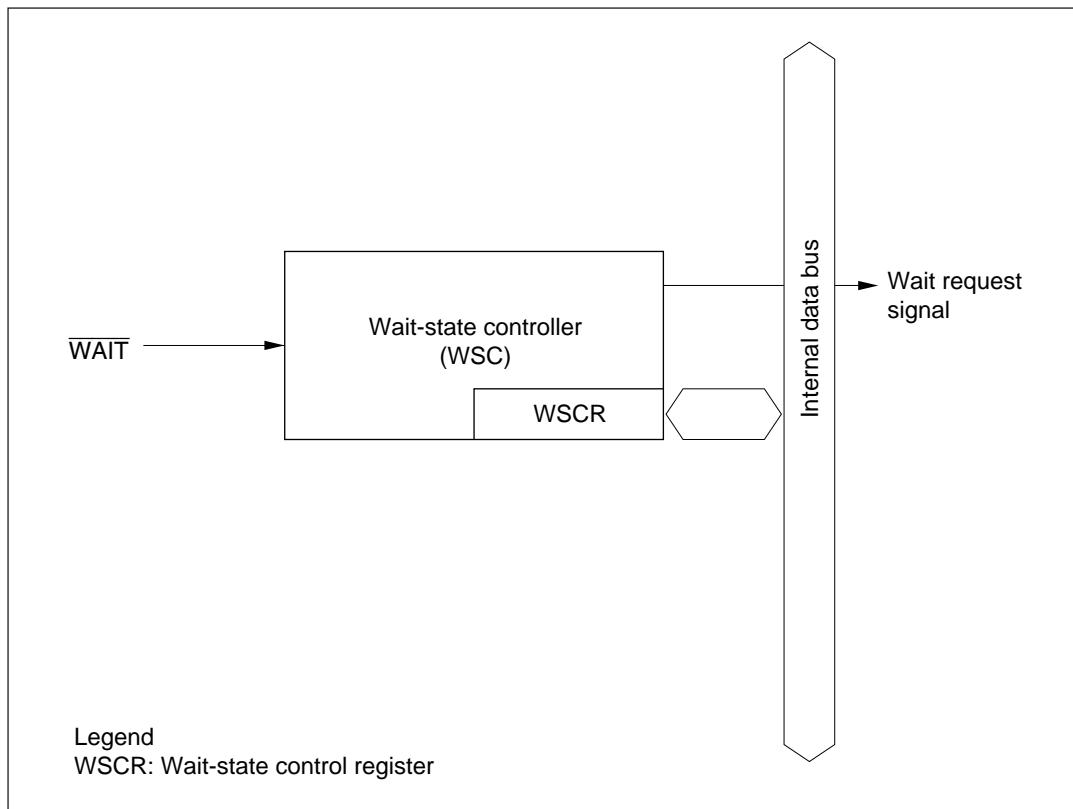


Figure 5-1 Block Diagram of Wait-State Controller

### 5.1.3 Input/Output Pins

Table 5-1 summarizes the wait-state controller's input pin.

**Table 5-1 Wait-State Controller Pins**

Name	Abbreviation	I/O	Function
Wait	WAIT	Input	Wait request signal for access to external addresses

### 5.1.4 Register Configuration

Table 5-2 summarizes the wait-state controller's register.

**Table 5-2 Register Configuration**

Address	Name	Abbreviation	R/W	Initial Value
H'FFC2	Wait-state control register	WSCR	R/W	H'08

## 5.2 Register Description

### 5.2.1 Wait-State Control Register (WSCR)

WSCR is an 8-bit readable/writable register that selects the wait mode for the wait-state controller (WSC) and specifies the number of wait states. It also controls emulation of flash memory by RAM, and frequency division of the clock signals supplied to the supporting modules.

Bit	7	6	5	4	3	2	1	0
	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	0	0	0	0	1	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

WSCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

## Bit 7—RAM Select (RAMS)

## Bit 6—RAM Area Select (RAM0)

Bits 7 and 6 select a RAM area for emulation of flash memory updates. For details, see the flash memory description in section 19, 20, ROM.

**Bit 5—Clock Double (CKDBL):** Controls frequency division of clock signals supplied to supporting modules. For details, see section 6, Clock Pulse Generator.

**Bit 4—Reserved:** This bit is reserved, but it can be written and read. Its initial value is 0.

**Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1/0):** These bits select the wait mode.

Bit 3 WMS1	Bit 2 WMS0	Description	
0	0	Programmable wait mode	
	1	No wait states inserted by wait-state controller	
1	0	Pin wait mode	(Initial value)
	1	Pin auto-wait mode	

**Bits 1 and 0—Wait Count 1 and 0 (WC1/0):** These bits select the number of wait states inserted in access to external address areas.

Bit 1 WC1	Bit 0 WC0	Description	
0	0	No wait states inserted by wait-state controller	(Initial value)
	1	1 state inserted	
1	0	2 states inserted	
	1	3 states inserted	

## 5.3 Wait Modes

**Programmable Wait Mode:** The number of wait states ( $T_W$ ) selected by bits WC1 and WC0 are inserted in all accesses to external addresses. Figure 5-2 shows the timing when the wait count is 1 ( $WC1 = 0$ ,  $WC0 = 1$ ).

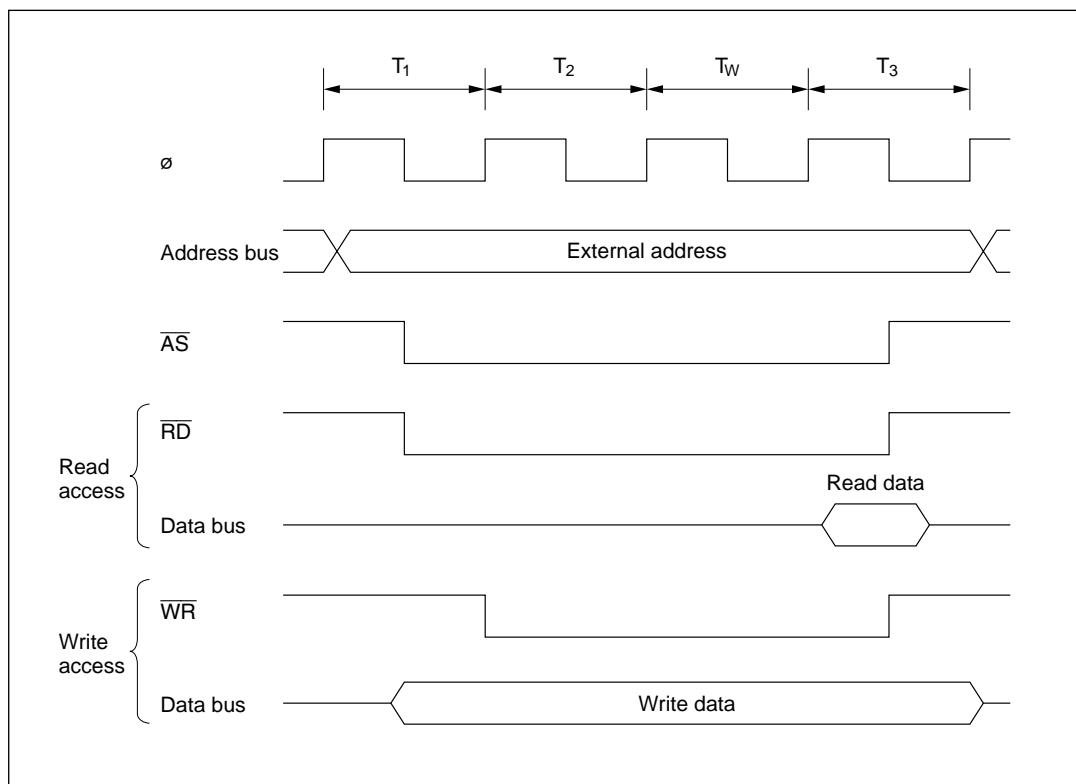
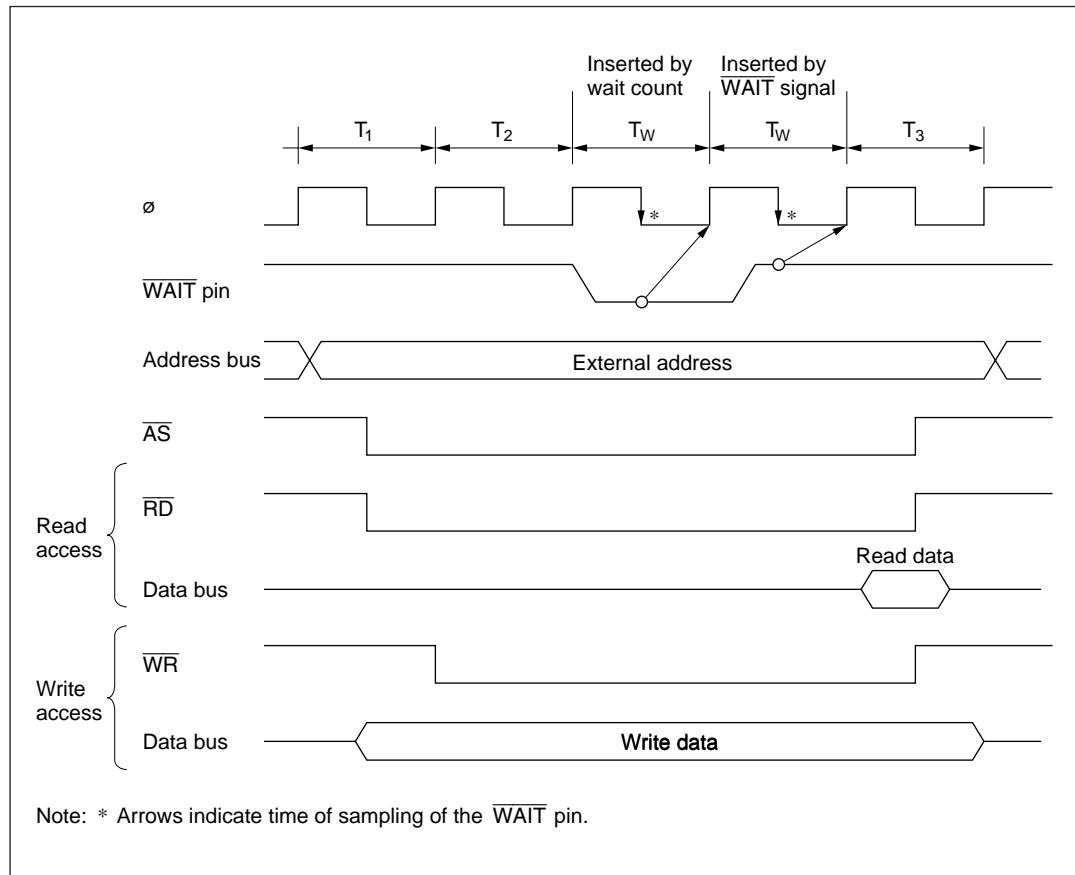


Figure 5-2 Programmable Wait Mode

**Pin Wait Mode:** In all accesses to external addresses, the number of wait states ( $T_W$ ) selected by bits WC1 and WC0 are inserted. If the  $\overline{\text{WAIT}}$  pin is low at the fall of the system clock ( $\phi$ ) in the last of these wait states, an additional wait state is inserted. If the  $\overline{\text{WAIT}}$  pin remains low, wait states continue to be inserted until the  $\overline{\text{WAIT}}$  signal goes high.

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

Figure 5-3 shows the timing when the wait count is 1 (WC1 = 0, WC0 = 1) and one additional wait state is inserted by  $\overline{\text{WAIT}}$  input.

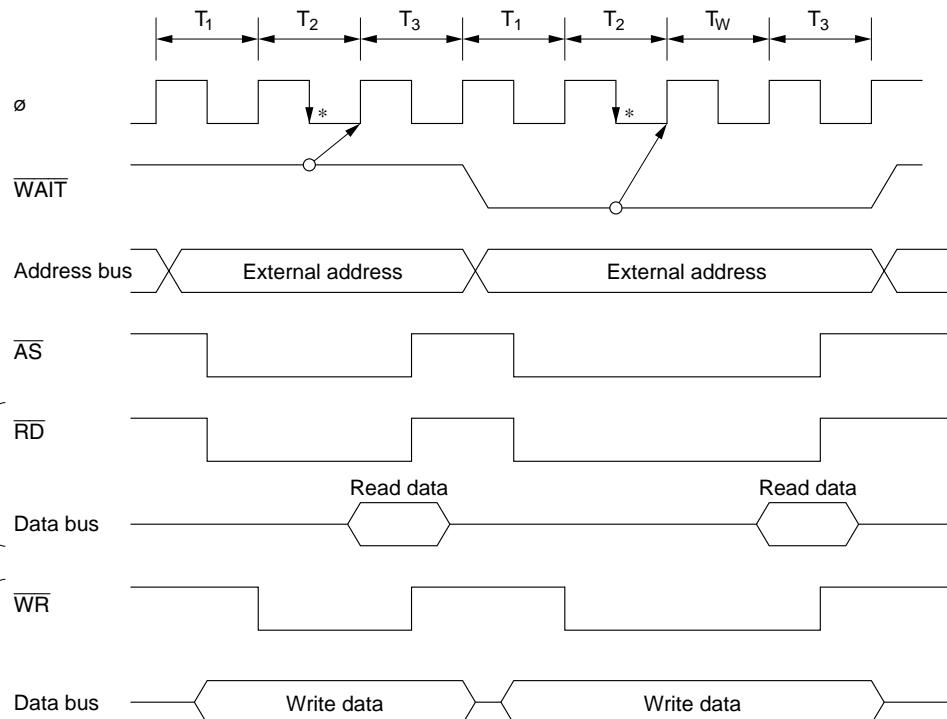


**Figure 5-3 Pin Wait Mode**

**Pin Auto-Wait Mode:** If the  $\overline{\text{WAIT}}$  pin is low, the number of wait states ( $T_W$ ) selected by bits WC1 and WC0 are inserted.

In pin auto-wait mode, if the  $\overline{\text{WAIT}}$  pin is low at the fall of the system clock ( $\phi$ ) in the  $T_2$  state, the number of wait states ( $T_W$ ) selected by bits WC1 and WC0 are inserted. No additional wait states are inserted even if the  $\overline{\text{WAIT}}$  pin remains low. Pin auto-wait mode can be used for an easy interface to low-speed memory, simply by routing the chip select signal to the  $\overline{\text{WAIT}}$  pin.

Figure 5-4 shows the timing when the wait count is 1.



Note: \* Arrows indicate time of sampling of the  $\overline{\text{WAIT}}$  pin.

**Figure 5-4 Pin Auto-Wait Mode**

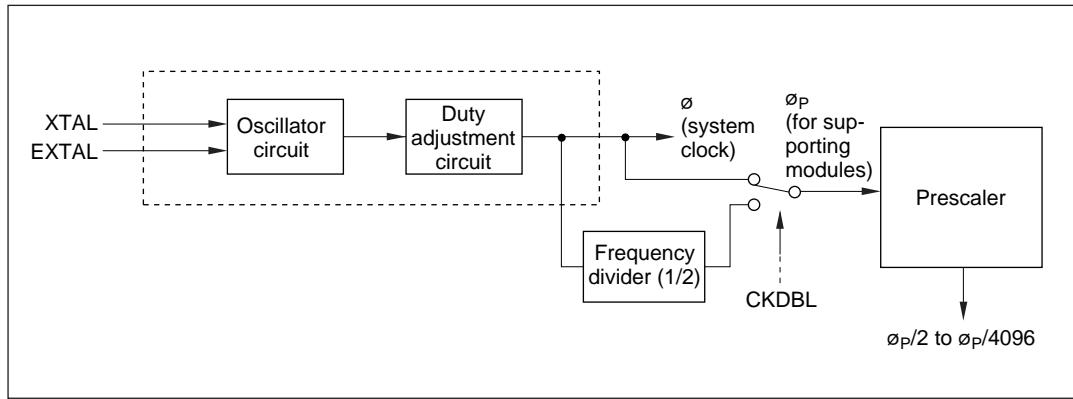
# Section 6 Clock Pulse Generator

## 6.1 Overview

The H8/3437 Series has a built-in clock pulse generator (CPG) consisting of an oscillator circuit, a duty adjustment circuit, and a divider and a prescaler that generates clock signals for the on-chip supporting modules.

### 6.1.1 Block Diagram

Figure 6-1 shows a block diagram of the clock pulse generator.



**Figure 6-1 Block Diagram of Clock Pulse Generator**

Input an external clock signal to the EXTAL pin, or connect a crystal resonator to the XTAL and EXTAL pins. The system clock frequency ( $\phi$ ) will be the same as the input frequency. This same system clock frequency ( $\phi_P$ ) can be supplied to timers and other supporting modules, or it can be divided by two. The selection is made by software, by controlling the CKDBL bit.

## 6.1.2 Wait-State Control Register (WSCR)

WSCR is an 8-bit readable/writable register that controls frequency division of the clock signals supplied to the supporting modules. It also controls wait-state insertion and emulation of flash memory by RAM.

WSCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Bit	7	6	5	4	3	2	1	0
	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	0	0	0	0	1	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

### Bit 7—RAM Select (RAMS)

### Bit 6—RAM Area Select (RAM0)

Bits 7 and 6 select a RAM area that can be used to emulate flash memory updates. For details, see the flash memory description in section 19, 20, ROM.

**Bit 5—Clock Double (CKDBL):** Controls the frequency division of clock signals supplied to supporting modules.

#### Bit 5

##### CKDBL      Description

0	The undivided system clock ( $\phi$ ) is supplied as the clock ( $\phi_P$ ) for supporting modules. (Initial value)
1	The system clock ( $\phi$ ) is divided by two and supplied as the clock ( $\phi_P$ ) for supporting modules.

**Bit 4—Reserved:** This bit is reserved, but it can be written and read. Its initial value is 0.

### Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1/0)

### Bits 1 and 0—Wait Count 1 and 0 (WC1/0)

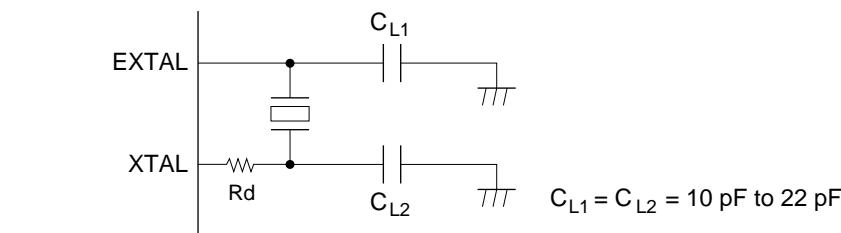
These bits control wait-state insertion. For details, see section 5, Wait-State Controller.

## 6.2 Oscillator Circuit

If an external crystal is connected across the EXTAL and XTAL pins, the on-chip oscillator circuit generates a system clock signal. Alternatively, an external clock signal can be applied to the EXTAL pin.

### 1. Connecting an External Crystal

**(1) Circuit Configuration:** An external crystal can be connected as in the example in figure 6-2. Table 6-1 indicates the appropriate damping resistance  $R_d$ . An AT-cut parallel resonance crystal should be used.



**Figure 6-2 Connection of Crystal Oscillator (Example)**

**Table 6-1 Damping Resistance**

Frequency (MHz)	2	4	8	10	12	16
$R_d$ max ( $\Omega$ )	1 k	500	200	0	0	0

**(2) Crystal Oscillator:** Figure 6-3 shows an equivalent circuit of the crystal resonator. The crystal resonator should have the characteristics listed in table 6-2.

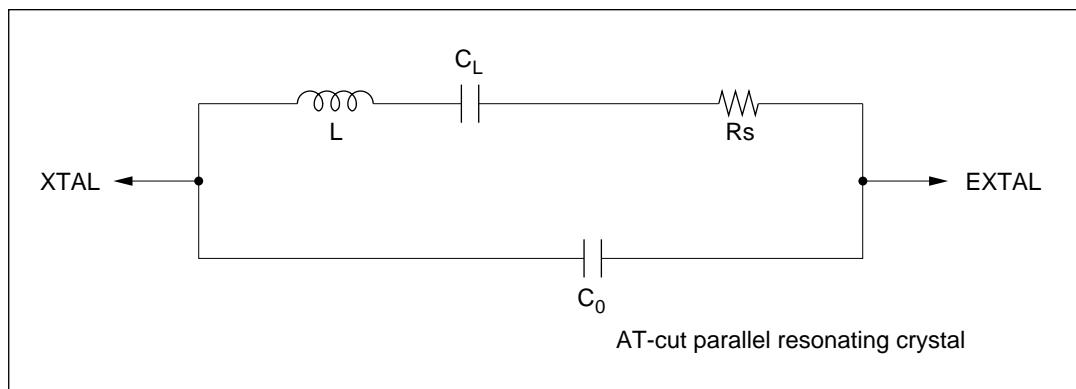


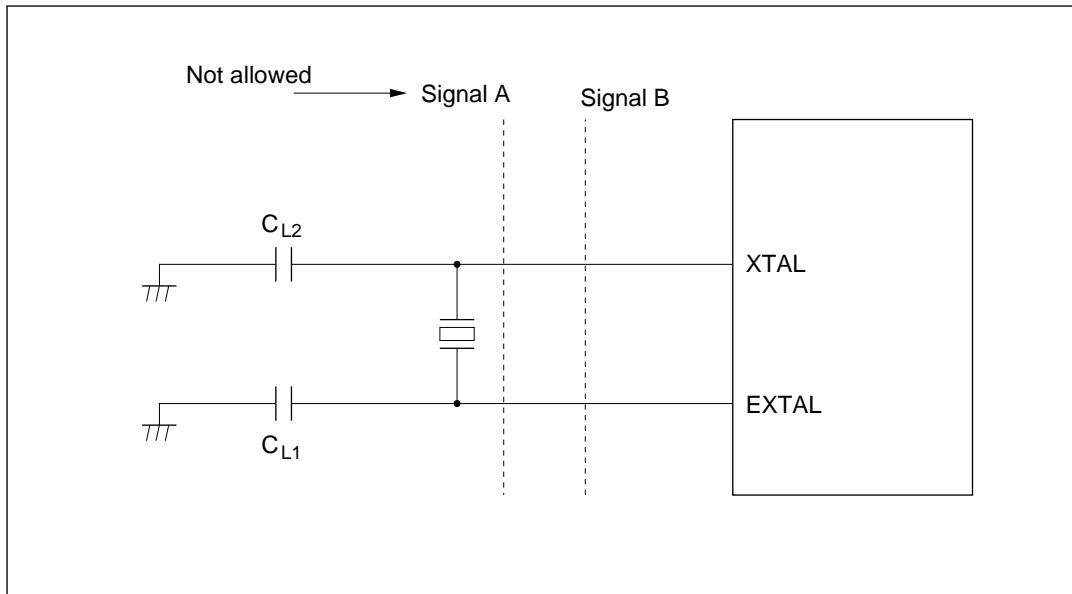
Figure 6-3 Equivalent Circuit of External Crystal

Table 6-2 External Crystal Parameters

Frequency (MHz)	2	4	8	10	12	16
Rs max ( $\Omega$ )	500	120	80	70	60	50
C0 (pF)	7 pF max					

Use a crystal with the same frequency as the desired system clock frequency ( $\phi$ ).

**(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 6-4. The crystal and its load capacitors should be placed as close as possible to the XTAL and EXTAL pins.

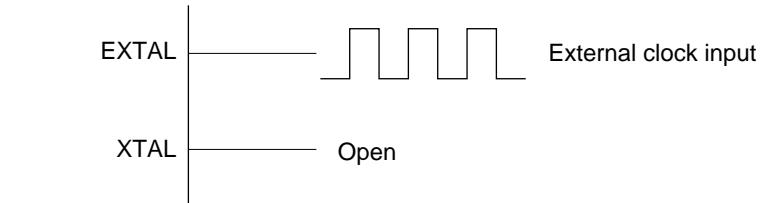


**Figure 6-4 Notes on Board Design around External Crystal**

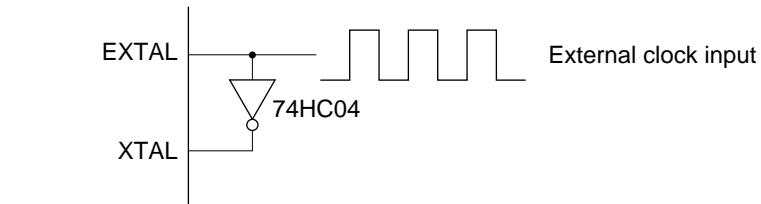
## 2. Input of External Clock Signal

**(1) Circuit Configuration:** An external clock signal can be input as shown in the examples in figure 6-5. In example (b) in figure 6-5, the external clock signal should be kept high during standby.

If the XTAL pin is left open, make sure the stray capacitance does not exceed 10 pF.



(a) Connections with XTAL pin left open



(b) Connections with inverted clock input at XTAL pin

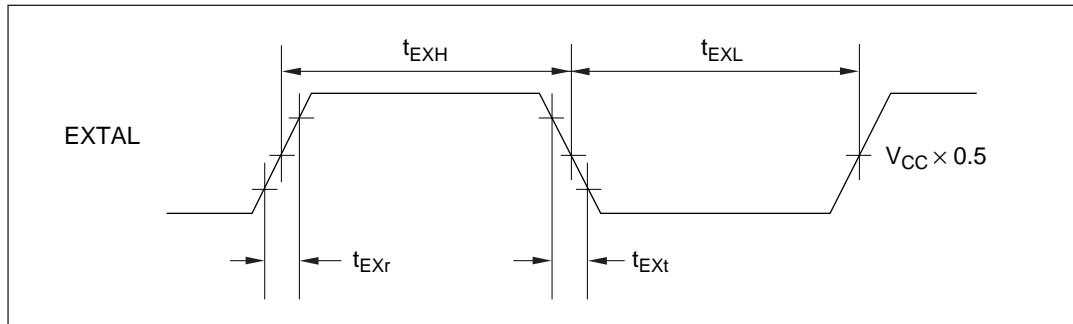
**Figure 6-5 External Clock Input (Example)**

## (2) External Clock Input

The external clock signal should have the same frequency as the desired system clock ( $\phi$ ). Clock timing parameters are given in table 6-3 and figure 6-6.

**Table 6-3 Clock Timing**

Item	Symbol	$V_{CC} = 2.7$ to 5.5 V		$V_{CC} = 4.0$ to 5.5 V		$V_{CC} = 5.0$ V $\pm 10\%$		Unit	Test Conditions
		Min	Max	Min	Max	Min	Max		
Low pulse width of external clock input	$t_{EXL}$	40	—	30	—	20	—	ns	Figure 6-6
High pulse width of external clock input	$t_{EXH}$	40	—	30	—	20	—	ns	
External clock rise time	$t_{EXr}$	—	10	—	10	—	5	ns	
External clock fall time	$t_{EXf}$	—	10	—	10	—	5	ns	
Clock pulse width low	$t_{CL}$	0.3 0.4	0.7 0.6	0.3 0.4	0.7 0.6	0.3 0.4	0.7 0.6	$t_{cyc}$	$\phi \geq 5$ MHz $\phi < 5$ MHz
Clock pulse width high	$t_{CH}$	0.3 0.4	0.7 0.6	0.3 0.4	0.7 0.6	0.3 0.4	0.7 0.6	$t_{cyc}$	$\phi \geq 5$ MHz $\phi < 5$ MHz



**Figure 6-6 External Clock Input Timing**

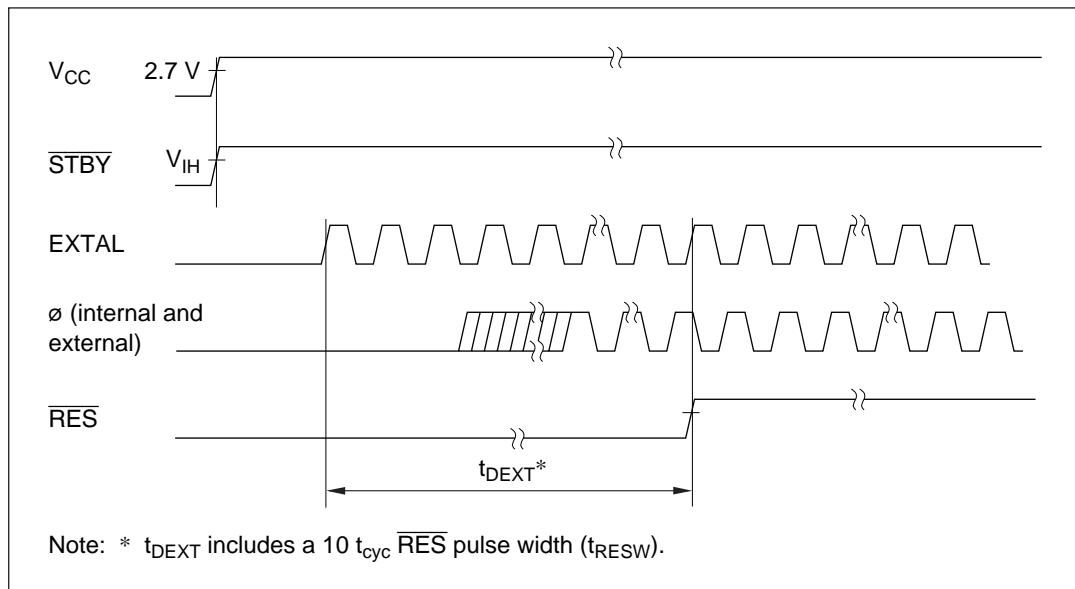
Table 6-4 lists the external clock output stabilization delay time. Figure 6-7 shows the timing for the external clock output stabilization delay time. The oscillator and duty correction circuit have the function of regulating the waveform of the external clock input to the EXTAL pin. When the specified clock signal is input to the EXTAL pin, internal clock signal output is confirmed after the elapse of the external clock output stabilization delay time ( $t_{DEXT}$ ). As clock signal output is not confirmed during the  $t_{DEXT}$  period, the reset signal should be driven low and the reset state maintained during this time.

**Table 6-4 External Clock Output Stabilization Delay Time**

Conditions:  $V_{CC} = 2.7$  to  $5.5$  V,  $AV_{CC} = 2.7$  to  $5.5$  V,  $V_{SS} = AV_{SS} = 0$  V

Item	Symbol	Min	Max	Unit	Notes
External clock output stabilization delay time	$t_{DEXT}^*$	500	—	μs	Figure 6-7

Note: \*  $t_{DEXT}$  includes a  $10 t_{cyc} \overline{RES}$  pulse width ( $t_{RESW}$ ).



**Figure 6-7 External Clock Output Stabilization Delay Time**

## 6.3 Duty Adjustment Circuit

When the clock frequency is 5 MHz or above, the duty adjustment circuit adjusts the duty cycle of the signal from the oscillator circuit to generate the system clock ( $\phi$ ).

## 6.4 Prescaler

The clock for the on-chip supporting modules ( $\phi_P$ ) has either the same frequency as the system clock ( $\phi$ ) or this frequency divided by two, depending on the CKDBL bit. The prescaler divides the frequency of  $\phi_P$  to generate internal clock signals with frequencies from  $\phi_P/2$  to  $\phi_P/4096$ .

# Section 7 I/O Ports

## 7.1 Overview

The H8/3437 Series has eight 8-bit input/output ports, one 7-bit input/output port, and one 3-bit input/output port, and are 8-bit input port.

Table 7-1 lists the functions of each port in each operating mode. As table 7-1 indicates, the port pins are multiplexed, and the pin functions differ depending on the operating mode.

Each port has a data direction register (DDR) that selects input or output, and a data register (DR) that stores output data. If bit manipulation instructions will be executed on the port data direction registers, see “Notes on Bit Manipulation Instructions” in section 2.5.5, Bit Manipulations.

Ports 1, 2, 3, 4, 6, 9, A, and B can drive one TTL load and a 90-pF capacitive load. Ports 5 and 8 can drive one TTL load and a 30-pF capacitive load. Ports 1 and 2 can drive LEDs (with 10-mA current sink). Ports 1 to 6, 8, 9, A, and B can drive a darlington transistor. Ports 1 to 3, 6, A, and B have built-in MOS pull-up transistors.

For block diagrams of the ports, see appendix C, I/O Port Block Diagrams.

Pins P8<sub>6</sub> in port 8, P9<sub>7</sub> in port 9, and PA<sub>4</sub>, PA<sub>5</sub>, PA<sub>6</sub>, and PA<sub>7</sub> in port A can be driven to operate as bus buffers. For details, see section 13, I<sup>2</sup>C Bus Interface.

**Table 7-1 Port Functions**

Port	Description	Pins	Expanded Modes		Single-Chip Mode
			Mode 1	Mode 2	Mode 3
Port 1	<ul style="list-style-type: none"> <li>• 8-bit I/O port</li> <li>• Can drive LEDs</li> <li>• Built-in input pull-ups</li> </ul>	P1 <sub>7</sub> to P1 <sub>0</sub> /A <sub>7</sub> to A <sub>0</sub>	Lower address output (A <sub>7</sub> to A <sub>0</sub> )	Lower address output (A <sub>7</sub> to A <sub>0</sub> ) or general input	General input/output
Port 2	<ul style="list-style-type: none"> <li>• 8-bit I/O port</li> <li>• Can drive LEDs</li> <li>• Built-in input pull-ups</li> </ul>	P2 <sub>7</sub> to P2 <sub>0</sub> /A <sub>15</sub> to A <sub>8</sub>	Upper address output (A <sub>15</sub> to A <sub>8</sub> )	Upper address output (A <sub>15</sub> to A <sub>8</sub> ) or general input	General input/output
Port 3	<ul style="list-style-type: none"> <li>• 8-bit I/O port</li> <li>• Built-in input pull-ups</li> <li>• HIF data bus</li> </ul>	P3 <sub>7</sub> to P3 <sub>0</sub> / D <sub>7</sub> to D <sub>0</sub> / HDB <sub>7</sub> to HDB <sub>0</sub>	Data bus (D <sub>7</sub> to D <sub>0</sub> )		HIF data bus (HDB <sub>7</sub> to HDB <sub>0</sub> ) or general input/output
Port 4	<ul style="list-style-type: none"> <li>• 8-bit I/O port</li> </ul>	P4 <sub>7</sub> /PW <sub>1</sub> P4 <sub>6</sub> /PW <sub>0</sub>	PWM timer 0/1 output (PW <sub>0</sub> , PW <sub>1</sub> ), or general input/output		
		P4 <sub>5</sub> /TMRI <sub>1</sub> /IRQ <sub>12</sub> P4 <sub>4</sub> /TMO <sub>1</sub> /IRQ <sub>1</sub> P4 <sub>3</sub> /TMCI <sub>1</sub> /IRQ <sub>11</sub>	8-bit timer 1 input/output (TMCI <sub>1</sub> , TMO <sub>1</sub> , TMRI <sub>1</sub> ), host processor interrupt request output from HIF (IRQ <sub>11</sub> , IRQ <sub>1</sub> , IRQ <sub>12</sub> ), or general input/output		
		P4 <sub>2</sub> /TMRI <sub>0</sub> P4 <sub>1</sub> /TMO <sub>0</sub> P4 <sub>0</sub> /TMCI <sub>0</sub>	8-bit timer 0 input/output (TMCI <sub>0</sub> , TMO <sub>0</sub> , TMRI <sub>0</sub> ) or general input/output		
Port 5	<ul style="list-style-type: none"> <li>• 3-bit I/O port</li> </ul>	P5 <sub>2</sub> /SCK <sub>0</sub> P5 <sub>1</sub> /RxD <sub>0</sub> P5 <sub>0</sub> /TxD <sub>0</sub>	Serial communication interface 0 input/output (TxD <sub>0</sub> , RxD <sub>0</sub> , SCK <sub>0</sub> ) or general input/output		
Port 6	<ul style="list-style-type: none"> <li>• 8-bit I/O port</li> <li>• Built-in input pull-ups</li> <li>• Key-sense interrupt inputs</li> </ul>	P6 <sub>7</sub> /KEYIN <sub>7</sub> /IRQ <sub>7</sub> P6 <sub>6</sub> /KEYIN <sub>6</sub> /FTOB/IRQ <sub>6</sub> P6 <sub>5</sub> /KEYIN <sub>5</sub> /FTID P6 <sub>4</sub> /KEYIN <sub>4</sub> /FTIC P6 <sub>3</sub> /KEYIN <sub>3</sub> /FTIB P6 <sub>2</sub> /KEYIN <sub>2</sub> /FTIA P6 <sub>1</sub> /KEYIN <sub>1</sub> /FTOA P6 <sub>0</sub> /KEYIN <sub>0</sub> /FTCI	16-bit free-running timer input/output (FTCI, FTOA, FTIA, FTIB, FTIC, FTID, FTOB), key-sense interrupt input (KEYIN <sub>7</sub> to KEYIN <sub>0</sub> ), external interrupt input (IRQ <sub>7</sub> , IRQ <sub>6</sub> ), or general input/output		
Port 7	<ul style="list-style-type: none"> <li>• 8-bit I/O port</li> </ul>	P7 <sub>7</sub> /AN <sub>7</sub> /DA <sub>1</sub> P7 <sub>6</sub> /AN <sub>6</sub> /DA <sub>0</sub>	Analog input to A/D converter (AN <sub>7</sub> , AN <sub>6</sub> ), analog output from D/A converter (DA <sub>1</sub> , DA <sub>0</sub> ), or general input		
		P7 <sub>5</sub> to P7 <sub>0</sub> / AN <sub>5</sub> to AN <sub>0</sub>	Analog input to A/D converter (AN <sub>5</sub> to AN <sub>0</sub> ) or general input		
Port 8	<ul style="list-style-type: none"> <li>• 7-bit I/O port</li> <li>• Can drive a bus line (P8<sub>6</sub>)</li> </ul>	P8 <sub>6</sub> /IRQ <sub>5</sub> /SCK <sub>1</sub> /SCL P8 <sub>5</sub> /IRQ <sub>4</sub> /RxD <sub>1</sub> /CS <sub>2</sub> P8 <sub>4</sub> /IRQ <sub>3</sub> /TxD <sub>1</sub> /IOW	Serial communication interface 1 input/output (TxD <sub>1</sub> , RxD <sub>1</sub> , SCK <sub>1</sub> ), HIF control input (CS <sub>2</sub> , IOW), I <sup>2</sup> C clock input/output (SCL), external interrupt input (IRQ <sub>5</sub> to IRQ <sub>3</sub> ), or general input/output		
		P8 <sub>3</sub> /IOR P8 <sub>2</sub> /CS <sub>1</sub> P8 <sub>1</sub> /GA <sub>20</sub> P8 <sub>0</sub> /HA <sub>0</sub>	HIF control input/output (HA <sub>0</sub> , GA <sub>20</sub> , CS <sub>1</sub> , IOR), or general input/output		

**Table 7-1 Port Functions (cont)**

Port	Description	Pins	Expanded Modes		Single-Chip Mode
			Mode 1	Mode 2	Mode 3
Port 9	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>Can drive a bus line (P9<sub>7</sub>)</li> </ul>	P9 <sub>7</sub> /WAIT/SDA	Expanded data bus control input (WAIT), I <sup>2</sup> C data input/output (SDA), or general input/output		I <sup>2</sup> C data input/output (SDA) or general input/output
		P9 <sub>6</sub> /∅	System clock (∅) output		∅ output or general input
		P9 <sub>5</sub> /AS P9 <sub>4</sub> /WR P9 <sub>3</sub> /RD	Expanded data bus control output (RD, WR, AS)		General input/output
		P9 <sub>2</sub> /IRQ <sub>0</sub> P9 <sub>1</sub> /IRQ <sub>1</sub> /EIOW P9 <sub>0</sub> /IRQ <sub>2</sub> /ECS <sub>2</sub> /ADTRG	HIF control input (ECS <sub>2</sub> , EIOW), trigger input to A/D converter (ADTRG), external interrupt input (IRQ <sub>2</sub> to IRQ <sub>0</sub> ), or general input/output		
Port A	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>Built-in input pull-ups</li> <li>Key-sense interrupt inputs</li> <li>Can drive bus lines (PA<sub>4</sub>, PA<sub>5</sub>, PA<sub>6</sub>, PA<sub>7</sub>)</li> </ul>	PA <sub>7</sub> to PA <sub>0</sub> / KEYIN <sub>15</sub> to KEYIN <sub>8</sub>	Key-sense interrupt input (KEYIN <sub>15</sub> to KEYIN <sub>8</sub> ) or general input/output		
Port B	<ul style="list-style-type: none"> <li>8-bit I/O port</li> <li>HIF data bus</li> <li>Built-in input pull-up MOS</li> </ul>	PB <sub>7</sub> to PB <sub>0</sub> / XDB <sub>7</sub> to XDB <sub>0</sub>	HIF data bus (XDB <sub>7</sub> to XDB <sub>0</sub> ) or general input/output		General input/output

## 7.2 Port 1

### 7.2.1 Overview

Port 1 is an 8-bit input/output port with the pin configuration shown in figure 7-1. The pin functions differ depending on the operating mode.

Port 1 has built-in, programmable MOS input pull-up transistors that can be used in modes 2 and 3.

Pins in port 1 can drive one TTL load and a 90-pF capacitive load. They can also drive LEDs and darlington transistors.

Port 1 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled)	Pin configuration in mode 2 (expanded mode with on-chip ROM enabled)
P1 <sub>7</sub> /A <sub>7</sub>	A <sub>7</sub> (output)	A <sub>7</sub> (output)/P1 <sub>7</sub> (input)
P1 <sub>6</sub> /A <sub>6</sub>	A <sub>6</sub> (output)	A <sub>6</sub> (output)/P1 <sub>6</sub> (input)
P1 <sub>5</sub> /A <sub>5</sub>	A <sub>5</sub> (output)	A <sub>5</sub> (output)/P1 <sub>5</sub> (input)
P1 <sub>4</sub> /A <sub>4</sub>	A <sub>4</sub> (output)	A <sub>4</sub> (output)/P1 <sub>4</sub> (input)
P1 <sub>3</sub> /A <sub>3</sub>	A <sub>3</sub> (output)	A <sub>3</sub> (output)/P1 <sub>3</sub> (input)
P1 <sub>2</sub> /A <sub>2</sub>	A <sub>2</sub> (output)	A <sub>2</sub> (output)/P1 <sub>2</sub> (input)
P1 <sub>1</sub> /A <sub>1</sub>	A <sub>1</sub> (output)	A <sub>1</sub> (output)/P1 <sub>1</sub> (input)
P1 <sub>0</sub> /A <sub>0</sub>	A <sub>0</sub> (output)	A <sub>0</sub> (output)/P1 <sub>0</sub> (input)

**Pin configuration in mode 3  
(single-chip mode)**

P1<sub>7</sub> (input/output)  
P1<sub>6</sub> (input/output)  
P1<sub>5</sub> (input/output)  
P1<sub>4</sub> (input/output)  
P1<sub>3</sub> (input/output)  
P1<sub>2</sub> (input/output)  
P1<sub>1</sub> (input/output)  
P1<sub>0</sub> (input/output)

Figure 7-1 Port 1 Pin Configuration

## 7.2.2 Register Configuration and Descriptions

Table 7-2 summarizes the port 1 registers.

**Table 7-2 Port 1 Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port 1 data direction register	P1DDR	W	H'FF (mode 1) H'00 (modes 2 and 3)	H'FFB0
Port 1 data register	P1DR	R/W	H'00	H'FFB2
Port 1 input pull-up control register	P1PCR	R/W	H'00	H'FFAC

### Port 1 Data Direction Register (P1DDR)

Bit	7	6	5	4	3	2	1	0
	P1 <sub>7</sub> DDR	P1 <sub>6</sub> DDR	P1 <sub>5</sub> DDR	P1 <sub>4</sub> DDR	P1 <sub>3</sub> DDR	P1 <sub>2</sub> DDR	P1 <sub>1</sub> DDR	P1 <sub>0</sub> DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P1DDR controls the input/output direction of each pin in port 1.

**Mode 1:** The P1DDR values are fixed at 1. Port 1 consists of lower address output pins. P1DDR values cannot be modified and are always read as 1.

In hardware standby mode, the address bus is in the high-impedance state.

**Mode 2:** A pin in port 1 is used for address output if the corresponding P1DDR bit is set to 1, and for general input if this bit is cleared to 0.

**Mode 3:** A pin in port 1 is used for general output if the corresponding P1DDR bit is set to 1, and for general input if this bit is cleared to 0.

In modes 2 and 3, P1DDR is a write-only register. Read data is invalid. If read, all bits always read 1. P1DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P1DDR bit is set to 1, the corresponding pin remains in the output state.

## Port 1 Data Register (P1DR)

Bit	7	6	5	4	3	2	1	0
	P1 <sub>7</sub>	P1 <sub>6</sub>	P1 <sub>5</sub>	P1 <sub>4</sub>	P1 <sub>3</sub>	P1 <sub>2</sub>	P1 <sub>1</sub>	P1 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

P1DR is an 8-bit register that stores data for pins P1<sub>7</sub> to P1<sub>0</sub>. When a P1DDR bit is set to 1, if port 1 is read, the value in P1DR is obtained directly, regardless of the actual pin state. When a P1DDR bit is cleared to 0, if port 1 is read the pin state is obtained.

P1DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

## Port 1 Input Pull-Up Control Register (P1PCR)

Bit	7	6	5	4	3	2	1	0
	P1 <sub>7</sub> PCR	P1 <sub>6</sub> PCR	P1 <sub>5</sub> PCR	P1 <sub>4</sub> PCR	P1 <sub>3</sub> PCR	P1 <sub>2</sub> PCR	P1 <sub>1</sub> PCR	P1 <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

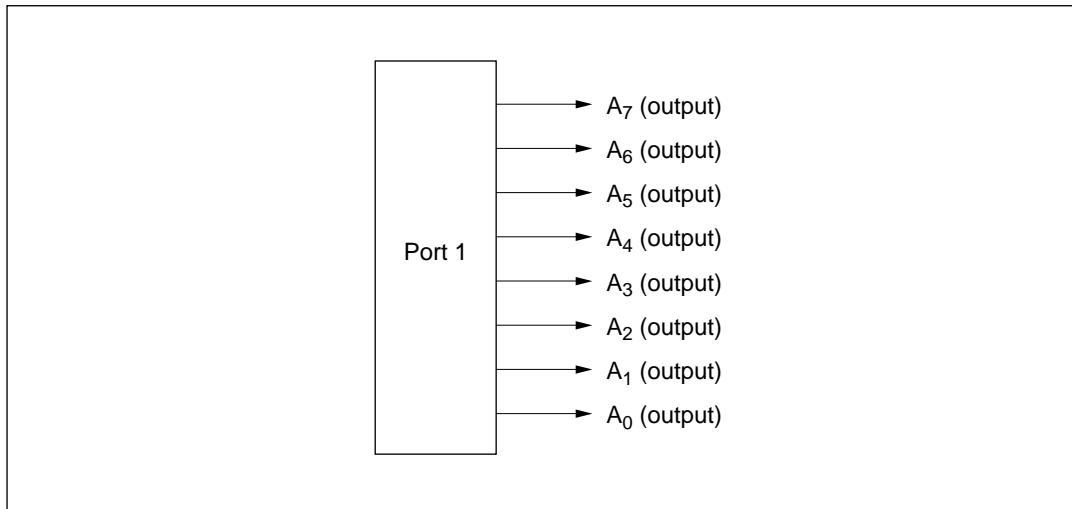
P1PCR is an 8-bit readable/writable register that controls the input pull-up transistors in port 1. If a P1DDR bit is cleared to 0 (designating input) and the corresponding P1PCR bit is set to 1, the input pull-up transistor is turned on.

P1PCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.2.3 Pin Functions in Each Mode

Port 1 has different pin functions in different modes. A separate description for each mode is given below.

**Pin Functions in Mode 1:** In mode 1 (expanded mode with on-chip ROM disabled), port 1 is automatically used for lower address output (A<sub>7</sub> to A<sub>0</sub>). Figure 7-2 shows the pin functions in mode 1.



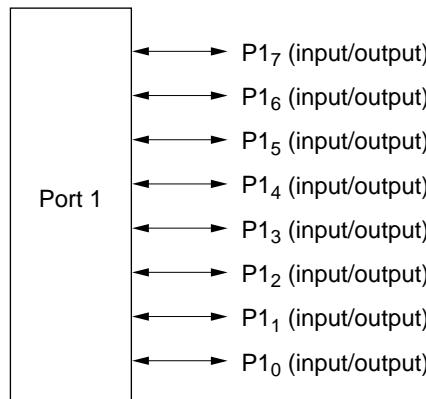
**Figure 7-2 Pin Functions in Mode 1 (Port 1)**

**Mode 2:** In mode 2 (expanded mode with on-chip ROM enabled), port 1 can provide lower address output pins and general input pins. Each pin becomes a lower address output pin if its P1DDR bit is set to 1, and a general input pin if this bit is cleared to 0. Following a reset, all pins are input pins. To be used for address output, their P1DDR bits must be set to 1. Figure 7-3 shows the pin functions in mode 2.

Port 1	When P1DDR = 1	When P1DDR = 0
	A <sub>7</sub> (output)	P1 <sub>7</sub> (input)
	A <sub>6</sub> (output)	P1 <sub>6</sub> (input)
	A <sub>5</sub> (output)	P1 <sub>5</sub> (input)
	A <sub>4</sub> (output)	P1 <sub>4</sub> (input)
	A <sub>3</sub> (output)	P1 <sub>3</sub> (input)
	A <sub>2</sub> (output)	P1 <sub>2</sub> (input)
	A <sub>1</sub> (output)	P1 <sub>1</sub> (input)
	A <sub>0</sub> (output)	P1 <sub>0</sub> (input)

**Figure 7-3 Pin Functions in Mode 2 (Port 1)**

**Mode 3:** In mode 3 (single-chip mode), the input or output direction of each pin can be selected individually. A pin becomes a general input pin when its P1DDR bit is cleared to 0 and a general output pin when this bit is set to 1. Figure 7-4 shows the pin functions in mode 3.



**Figure 7-4 Pin Functions in Mode 3 (Port 1)**

#### 7.2.4 Input Pull-Up Transistors

Port 1 has built-in programmable input pull-up transistors that are available in modes 2 and 3. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up in mode 2 or 3, set the corresponding P1PCR bit to 1 and clear the corresponding P1DDR bit to 0. P1PCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-3 indicates the states of the input pull-up transistors in each operating mode.

**Table 7-3 States of Input Pull-Up Transistors (Port 1)**

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	Off	Off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The input pull-up transistor is always off.

On/off: The input pull-up transistor is on if P1PCR = 1 and P1DDR = 0, but off otherwise.

## 7.3 Port 2

### 7.3.1 Overview

Port 2 is an 8-bit input/output port with the pin configuration shown in figure 7-5. The pin functions differ depending on the operating mode.

Port 2 has built-in, programmable MOS input pull-up transistors that can be used in modes 2 and 3.

Pins in port 2 can drive one TTL load and a 90-pF capacitive load. They can also drive LEDs and darlington transistors.

Port 2 pins	Pin configuration in mode 1 (expanded mode with on-chip ROM disabled)	Pin configuration in mode 2 (expanded mode with on-chip ROM enabled)
P2 <sub>7</sub> /A <sub>15</sub>	A <sub>15</sub> (output)	A <sub>15</sub> (output)/P2 <sub>7</sub> (input)
P2 <sub>6</sub> /A <sub>14</sub>	A <sub>14</sub> (output)	A <sub>14</sub> (output)/P2 <sub>6</sub> (input)
P2 <sub>5</sub> /A <sub>13</sub>	A <sub>13</sub> (output)	A <sub>13</sub> (output)/P2 <sub>5</sub> (input)
P2 <sub>4</sub> /A <sub>12</sub>	A <sub>12</sub> (output)	A <sub>12</sub> (output)/P2 <sub>4</sub> (input)
P2 <sub>3</sub> /A <sub>11</sub>	A <sub>11</sub> (output)	A <sub>11</sub> (output)/P2 <sub>3</sub> (input)
P2 <sub>2</sub> /A <sub>10</sub>	A <sub>10</sub> (output)	A <sub>10</sub> (output)/P2 <sub>2</sub> (input)
P2 <sub>1</sub> /A <sub>9</sub>	A <sub>9</sub> (output)	A <sub>9</sub> (output)/P2 <sub>1</sub> (input)
P2 <sub>0</sub> /A <sub>8</sub>	A <sub>8</sub> (output)	A <sub>8</sub> (output)/P2 <sub>0</sub> (input)

**Pin configuration in mode 3  
(single-chip mode)**

P2<sub>7</sub> (input/output)  
P2<sub>6</sub> (input/output)  
P2<sub>5</sub> (input/output)  
P2<sub>4</sub> (input/output)  
P2<sub>3</sub> (input/output)  
P2<sub>2</sub> (input/output)  
P2<sub>1</sub> (input/output)  
P2<sub>0</sub> (input/output)

Figure 7-5 Port 2 Pin Configuration

### 7.3.2 Register Configuration and Descriptions

Table 7-4 summarizes the port 2 registers.

**Table 7-4 Port 2 Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port 2 data direction register	P2DDR	W	H'FF (mode 1) H'00 (modes 2 and 3)	H'FFB1
Port 2 data register	P2DR	R/W	H'00	H'FFB3
Port 2 input pull-up control register	P2PCR	R/W	H'00	H'FFAD

#### Port 2 Data Direction Register (P2DDR)

Bit	7	6	5	4	3	2	1	0
	P2 <sub>7</sub> DDR	P2 <sub>6</sub> DDR	P2 <sub>5</sub> DDR	P2 <sub>4</sub> DDR	P2 <sub>3</sub> DDR	P2 <sub>2</sub> DDR	P2 <sub>1</sub> DDR	P2 <sub>0</sub> DDR
<b>Mode 1</b>								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
<b>Modes 2 and 3</b>								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P2DDR controls the input/output direction of each pin in port 2.

**Mode 1:** The P2DDR values are fixed at 1. Port 2 consists of upper address output pins. P2DDR values cannot be modified and are always read as 1.

In hardware standby mode, the address bus is in the high-impedance state.

**Mode 2:** A pin in port 2 is used for address output if the corresponding P2DDR bit is set to 1, and for general input if this bit is cleared to 0.

**Mode 3:** A pin in port 2 is used for general output if the corresponding P2DDR bit is set to 1, and for general input if this bit is cleared to 0.

In modes 2 and 3, P2DDR is a write-only register. Read data is invalid. If read, all bits always read 1. P2DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P2DDR bit is set to 1, the corresponding pin remains in the output state.

## Port 2 Data Register (P2DR)

Bit	7	6	5	4	3	2	1	0
	P2 <sub>7</sub>	P2 <sub>6</sub>	P2 <sub>5</sub>	P2 <sub>4</sub>	P2 <sub>3</sub>	P2 <sub>2</sub>	P2 <sub>1</sub>	P2 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

P2DR is an 8-bit register that stores data for pins P2<sub>7</sub> to P2<sub>0</sub>. When a P2DDR bit is set to 1, if port 2 is read, the value in P2DR is obtained directly, regardless of the actual pin state. When a P2DDR bit is cleared to 0, if port 2 is read the pin state is obtained.

P2DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

## Port 2 Input Pull-Up Control Register (P2PCR)

Bit	7	6	5	4	3	2	1	0
	P2 <sub>7</sub> PCR	P2 <sub>6</sub> PCR	P2 <sub>5</sub> PCR	P2 <sub>4</sub> PCR	P2 <sub>3</sub> PCR	P2 <sub>2</sub> PCR	P2 <sub>1</sub> PCR	P2 <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

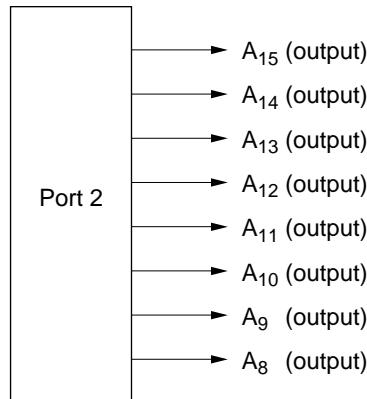
P2PCR is an 8-bit readable/writable register that controls the input pull-up transistors in port 2. If a P2DDR bit is cleared to 0 (designating input) and the corresponding P2PCR bit is set to 1, the input pull-up transistor is turned on.

P2PCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.3.3 Pin Functions in Each Mode

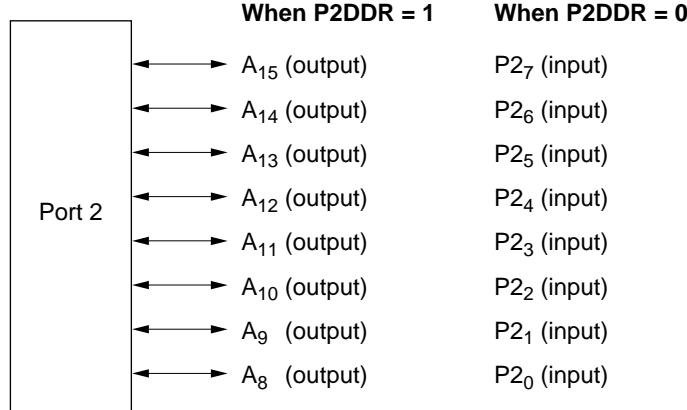
Port 2 has different pin functions in different modes. A separate description for each mode is given below.

**Pin Functions in Mode 1:** In mode 1 (expanded mode with on-chip ROM disabled), port 2 is automatically used for upper address output ( $A_{15}$  to  $A_8$ ). Figure 7-6 shows the pin functions in mode 1.



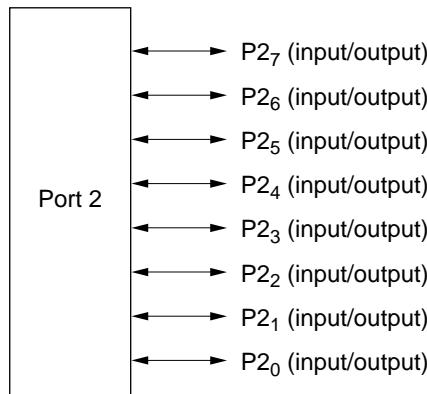
**Figure 7-6 Pin Functions in Mode 1 (Port 2)**

**Mode 2:** In mode 2 (expanded mode with on-chip ROM enabled), port 2 can provide upper address output pins and general input pins. Each pin becomes an upper address output pin if its P2DDR bit is set to 1, and a general input pin if this bit is cleared to 0. Following a reset, all pins are input pins. To be used for address output, their P2DDR bits must be set to 1. Figure 7-7 shows the pin functions in mode 2.



**Figure 7-7 Pin Functions in Mode 2 (Port 2)**

**Mode 3:** In mode 3 (single-chip mode), the input or output direction of each pin can be selected individually. A pin becomes a general input pin when its P2DDR bit is cleared to 0, and a general output pin when this bit is set to 1. Figure 7-8 shows the pin functions in mode 3.



**Figure 7-8 Pin Functions in Mode 3 (Port 2)**

#### 7.3.4 Input Pull-Up Transistors

Port 2 has built-in programmable input pull-up transistors that are available in modes 2 and 3. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up in mode 2 or 3, set the corresponding P2PCR bit to 1 and clear the corresponding P2DDR bit to 0. P2PCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-5 indicates the states of the input pull-up transistors in each operating mode.

**Table 7-5 States of Input Pull-Up Transistors (Port 2)**

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	Off	Off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The input pull-up transistor is always off.

On/off: The input pull-up transistor is on if P2PCR = 1 and P2DDR = 0, but off otherwise.

## 7.4 Port 3

### 7.4.1 Overview

Port 3 is an 8-bit input/output port that is multiplexed with the data bus and host interface data bus. Figure 7-9 shows the pin configuration of port 3. The pin functions differ depending on the operating mode.

Port 3 has built-in, programmable MOS input pull-up transistors that can be used in mode 3.

Pins in port 3 can drive one TTL load and a 90-pF capacitive load. They can also drive a darlington transistor.

<b>Port 3 pins</b>		<b>Pin configuration in mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled)</b>
	P3 <sub>7</sub> /D <sub>7</sub> /HDB <sub>7</sub>	D <sub>7</sub> (input/output)
	P3 <sub>6</sub> /D <sub>6</sub> /HDB <sub>6</sub>	D <sub>6</sub> (input/output)
	P3 <sub>5</sub> /D <sub>5</sub> /HDB <sub>5</sub>	D <sub>5</sub> (input/output)
	P3 <sub>4</sub> /D <sub>4</sub> /HDB <sub>4</sub>	D <sub>4</sub> (input/output)
	P3 <sub>3</sub> /D <sub>3</sub> /HDB <sub>3</sub>	D <sub>3</sub> (input/output)
	P3 <sub>2</sub> /D <sub>2</sub> /HDB <sub>2</sub>	D <sub>2</sub> (input/output)
	P3 <sub>1</sub> /D <sub>1</sub> /HDB <sub>1</sub>	D <sub>1</sub> (input/output)
	P3 <sub>0</sub> /D <sub>0</sub> /HDB <sub>0</sub>	D <sub>0</sub> (input/output)

<b>Pin configuration in mode 3 (single-chip mode)</b>	
<b>Master mode</b>	<b>Slave mode</b>
P3 <sub>7</sub> (input/output)	HDB <sub>7</sub> (input/output)
P3 <sub>6</sub> (input/output)	HDB <sub>6</sub> (input/output)
P3 <sub>5</sub> (input/output)	HDB <sub>5</sub> (input/output)
P3 <sub>4</sub> (input/output)	HDB <sub>4</sub> (input/output)
P3 <sub>3</sub> (input/output)	HDB <sub>3</sub> (input/output)
P3 <sub>2</sub> (input/output)	HDB <sub>2</sub> (input/output)
P3 <sub>1</sub> (input/output)	HDB <sub>1</sub> (input/output)
P3 <sub>0</sub> (input/output)	HDB <sub>0</sub> (input/output)

**Figure 7-9 Port 3 Pin Configuration**

## 7.4.2 Register Configuration and Descriptions

Table 7-6 summarizes the port 3 registers.

**Table 7-6 Port 3 Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port 3 data direction register	P3DDR	W	H'00	H'FFB4
Port 3 data register	P3DR	R/W	H'00	H'FFB6
Port 3 input pull-up control register	P3PCR	R/W	H'00	H'FFAE

### Port 3 Data Direction Register (P3DDR)

Bit	7	6	5	4	3	2	1	0
	P3 <sub>7</sub> DDR	P3 <sub>6</sub> DDR	P3 <sub>5</sub> DDR	P3 <sub>4</sub> DDR	P3 <sub>3</sub> DDR	P3 <sub>2</sub> DDR	P3 <sub>1</sub> DDR	P3 <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P3DDR is an 8-bit register that controls the input/output direction of each pin in port 3. P3DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

**Modes 1 and 2:** In mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled), the input/output directions designated by P3DDR are ignored. Port 3 automatically consists of the input/output pins of the 8-bit data bus (D<sub>7</sub> to D<sub>0</sub>).

The data bus is in the high-impedance state during reset, and during hardware and software standby.

**Mode 3:** A pin in port 3 is used for general output if the corresponding P3DDR bit is set to 1, and for general input if this bit is cleared to 0. P3DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P3DDR bit is set to 1, the corresponding pin remains in the output state.

## Port 3 Data Register (P3DR)

Bit	7	6	5	4	3	2	1	0
	P3 <sub>7</sub>	P3 <sub>6</sub>	P3 <sub>5</sub>	P3 <sub>4</sub>	P3 <sub>3</sub>	P3 <sub>2</sub>	P3 <sub>1</sub>	P3 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

P3DR is an 8-bit register that stores data for pins P3<sub>7</sub> to P3<sub>0</sub>. When a P3DDR bit is set to 1, if port 3 is read, the value in P3DR is obtained directly, regardless of the actual pin state. When a P3DDR bit is cleared to 0, if port 3 is read the pin state is obtained.

P3DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

## Port 3 Input Pull-Up Control Register (P3PCR)

Bit	7	6	5	4	3	2	1	0
	P3 <sub>7</sub> PCR	P3 <sub>6</sub> PCR	P3 <sub>5</sub> PCR	P3 <sub>4</sub> PCR	P3 <sub>3</sub> PCR	P3 <sub>2</sub> PCR	P3 <sub>1</sub> PCR	P3 <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

P3PCR is an 8-bit readable/writable register that controls the input pull-up transistors in port 3. If a P3DDR bit is cleared to 0 (designating input) and the corresponding P3PCR bit is set to 1, the input pull-up transistor is turned on.

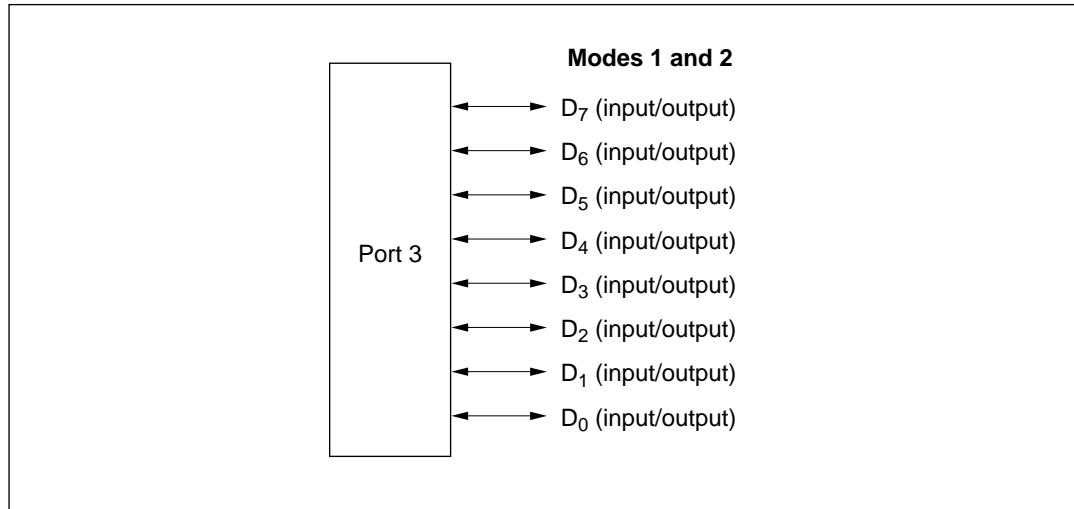
P3PCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

The input pull-ups cannot be used in slave mode (when the host interface is enabled).

### 7.4.3 Pin Functions in Each Mode

Port 3 has different pin functions in different modes. A separate description for each mode is given below.

**Pin Functions in Modes 1 and 2:** In mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled), port 3 is automatically used for the input/output pins of the data bus ( $D_7$  to  $D_0$ ). Figure 7-10 shows the pin functions in modes 1 and 2.

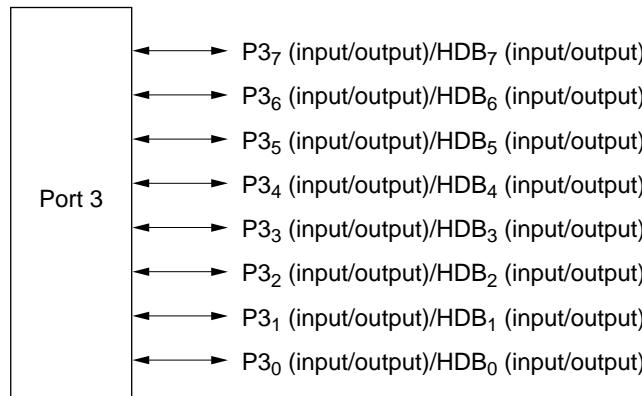


**Figure 7-10 Pin Functions in Modes 1 and 2 (Port 3)**

**Mode 3:** In mode 3 (single-chip mode), when the host interface enable bit (HIE) is cleared to 0 in the system control register (SYSCR), port 3 is a general-purpose input/output port. A pin becomes an output pin when its P3DDR bit is set to 1, and an input pin when this bit is cleared to 0.

When the HIE bit is set to 1, selecting slave mode, port 3 becomes the host interface data bus (HDB<sub>7</sub> to HDB<sub>0</sub>). P3DR and P3DDR should be cleared to H'00 in slave mode. For details, see section 14, Host Interface.

Figure 7-11 shows the pin functions in mode 3.



**Figure 7-11 Pin Functions in Mode 3 (Port 3)**

#### 7.4.4 Input Pull-Up Transistors

Port 3 has built-in programmable input pull-up transistors that are available in mode 3. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up in mode 3, set the corresponding P3PCR bit to 1 and clear the corresponding P3DDR bit to 0. P3PCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-7 indicates the states of the input pull-up transistors in each operating mode.

**Table 7-7 States of Input Pull-Up Transistors (Port 3)**

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	Off	Off
2	Off	Off	Off	Off
3	Off	Off	On/off	On/off

Notes: Off: The input pull-up transistor is always off.

On/off: The input pull-up transistor is on if P3PCR = 1 and P3DDR = 0, but off otherwise.

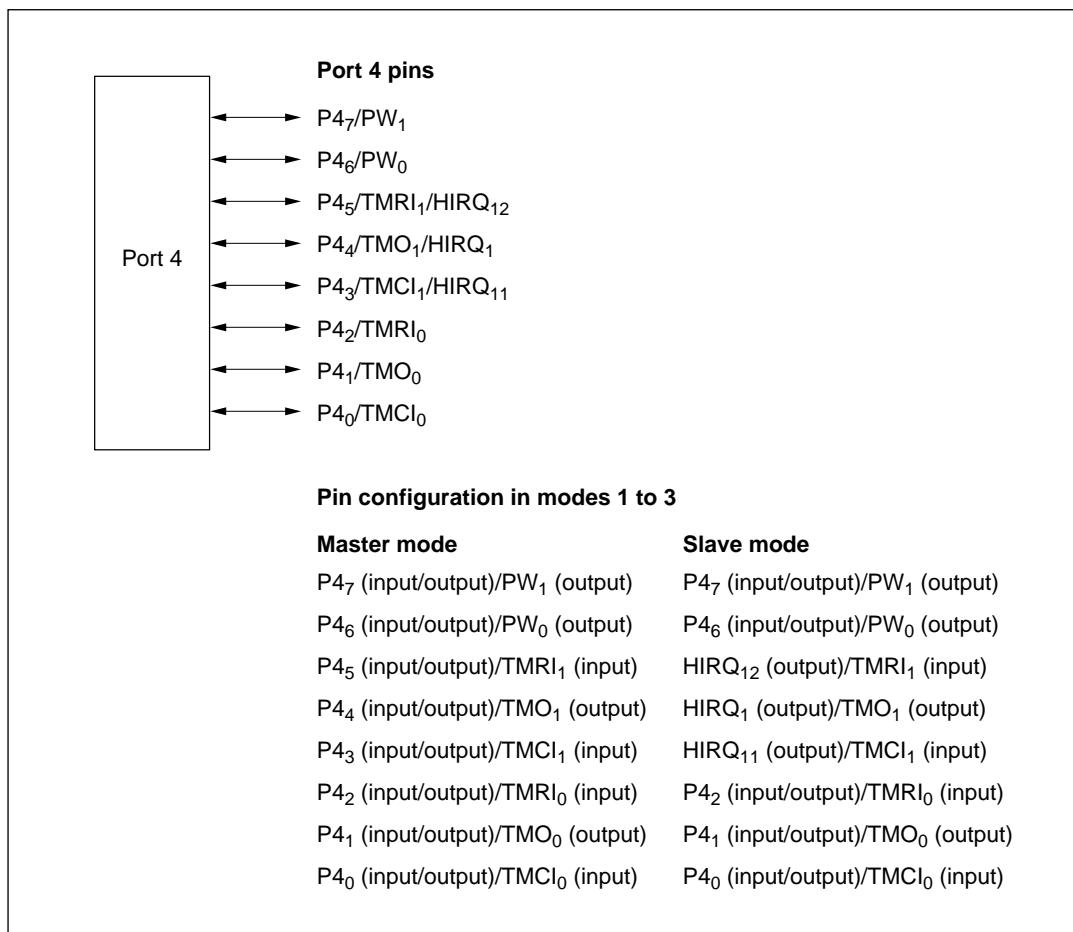
## 7.5 Port 4

### 7.5.1 Overview

Port 4 is an 8-bit input/output port that is multiplexed with input/output pins (TMRI<sub>0</sub>, TMRI<sub>1</sub>, TMCI<sub>0</sub>, TMCI<sub>1</sub>, TMO<sub>0</sub>, TMO<sub>1</sub>) of 8-bit timers 0 and 1 and output pins (PW<sub>0</sub>, PW<sub>1</sub>) of PWM timers 0 and 1. In slave mode, P4<sub>3</sub> to P4<sub>5</sub> output host interrupt requests. Pins not used by timers or for host interrupt requests are available for general input/output.

Figure 7-12 shows the pin configuration of port 4.

Pins in port 4 can drive one TTL load and a 90-pF capacitive load. They can also drive a darlington transistor.



**Figure 7-12 Port 4 Pin Configuration**

## 7.5.2 Register Configuration and Descriptions

Table 7-8 summarizes the port 4 registers.

**Table 7-8 Port 4 Registers**

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

### Port 4 Data Direction Register (P4DDR)

Bit	7	6	5	4	3	2	1	0
	P4 <sub>7</sub> DDR	P4 <sub>6</sub> DDR	P4 <sub>5</sub> DDR	P4 <sub>4</sub> DDR	P4 <sub>3</sub> DDR	P4 <sub>2</sub> DDR	P4 <sub>1</sub> DDR	P4 <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

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

P4DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

P4DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P4DDR bit is set to 1, the corresponding pin remains in the output state.

If a transition to software standby mode occurs while port 4 is being used by an on-chip supporting module (for example, for 8-bit timer output), the on-chip supporting module will be initialized, so the pin will revert to general-purpose input/output, controlled by P4DDR and P4DR.

## Port 4 Data Register (P4DR)

Bit	7	6	5	4	3	2	1	0
	P4 <sub>7</sub>	P4 <sub>6</sub>	P4 <sub>5</sub>	P4 <sub>4</sub>	P4 <sub>3</sub>	P4 <sub>2</sub>	P4 <sub>1</sub>	P4 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

P4DR is an 8-bit register that stores data for pins P4<sub>7</sub> to P4<sub>0</sub>. When a P4DDR bit is set to 1, if port 4 is read, the value in P4DR is obtained directly, regardless of the actual pin state. When a P4DDR bit is cleared to 0, if port 4 is read the pin state is obtained. This also applies to pins used by on-chip supporting modules.

P4DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.5.3 Pin Functions

Port 4 has different pin functions depending on whether the chip is or is not operating in slave mode. Table 7-9 indicates the pin functions of port 4.

**Table 7-9 Port 4 Pin Functions**

Pin	Pin Functions and Selection Method					
P4 <sub>7</sub> /PW <sub>1</sub>	Bit OE in TCR of PWM timer 1 and bit P4 <sub>7</sub> DDR select the pin function as follows					
	OE	0		1		
	P4 <sub>7</sub> DDR	0	1	0 1		
	Pin function	P4 <sub>7</sub> input	P4 <sub>7</sub> output	PW <sub>1</sub> output		
P4 <sub>6</sub> /PW <sub>0</sub>	Bit OE in TCR of PWM timer 0 and bit P4 <sub>6</sub> DDR select the pin function as follows					
	OE	0		1		
	P4 <sub>6</sub> DDR	0	1	0 1		
	Pin function	P4 <sub>6</sub> input	P4 <sub>6</sub> output	PW <sub>0</sub> output		
P4 <sub>5</sub> /TMRI <sub>1</sub> /HIRQ <sub>12</sub>	Bit P4 <sub>5</sub> DDR and the operating mode select the pin function as follows					
	P4 <sub>5</sub> DDR	0		1		
	Operating mode	—	Not slave mode	Slave mode		
	Pin function	P4 <sub>5</sub> input	P4 <sub>5</sub> output	HIRQ <sub>12</sub> output		
			TMRI <sub>1</sub> input			
	TMRI <sub>1</sub> input is usable when bits CCLR1 and CCLR0 are both set to 1 in TCR of 8-bit timer 1					
P4 <sub>4</sub> /TMO <sub>1</sub> /HIRQ <sub>1</sub>	Bits OS3 to OS0 in TCSR of 8-bit timer 1, bit P4 <sub>4</sub> DDR, and the operating mode select the pin function as follows					
	OS3 to 0	All 0		Not all 0		
	P4 <sub>4</sub> DDR	0	1	—		
	Operating mode	—	Not slave mode	Slave mode		
	Pin function	P4 <sub>4</sub> input	P4 <sub>4</sub> output	HIRQ <sub>1</sub> output		
			TMO <sub>1</sub> output			

**Table 7-9 Port 4 Pin Functions (cont)**

Pin	Pin Functions and Selection Method						
P4 <sub>3</sub> /TMCI <sub>1</sub> /IRQ <sub>11</sub>	Bit P4 <sub>3</sub> DDR and the operating mode select the pin function as follows						
	P4 <sub>3</sub> DDR	0	1				
	Operating mode	—	Not slave mode	Slave mode			
	Pin function	P4 <sub>3</sub> input	P4 <sub>3</sub> output	IRQ <sub>11</sub> output			
		TMCI <sub>1</sub> input					
TMCI <sub>1</sub> input is usable when bits CKS2 to CKS0 in TCR of 8-bit timer 1 select an external clock source							
P4 <sub>2</sub> /TMRI <sub>0</sub>	P4 <sub>2</sub> DDR	0	1				
	Pin function	P4 <sub>2</sub> input	P4 <sub>2</sub> output				
		TMRI <sub>0</sub> input					
TMRI <sub>0</sub> input is usable when bits CCLR1 and CCLR0 are both set to 1 in TCR of 8-bit timer 0							
P4 <sub>1</sub> /TMO <sub>0</sub>	Bits OS3 to OS0 in TCSR of 8-bit timer 0 and bit P4 <sub>1</sub> DDR select the pin function as follows						
	OS3 to 0	All 0		Not all 0			
	P4 <sub>1</sub> DDR	0	1	0 1			
	Pin function	P4 <sub>1</sub> input	P4 <sub>1</sub> output	TMO <sub>0</sub> output			
P4 <sub>0</sub> /TMCI <sub>0</sub>	P4 <sub>0</sub> DDR	0	1				
	Pin function	P4 <sub>0</sub> input	P4 <sub>0</sub> output				
		TMCI <sub>0</sub> input					
TMCI <sub>0</sub> input is usable when bits CKS2 to CKS0 in TCR of 8-bit timer 0 select an external clock source							

## 7.6 Port 5

### 7.6.1 Overview

Port 5 is a 3-bit input/output port that is multiplexed with input/output pins (Tx $D_0$ , Rx $D_0$ , SCK $_0$ ) of serial communication interface 0. The port 5 pin functions are the same in all operating modes. Figure 7-13 shows the pin configuration of port 5.

Pins in port 5 can drive one TTL load and a 30-pF capacitive load. They can also drive a darlington transistor.

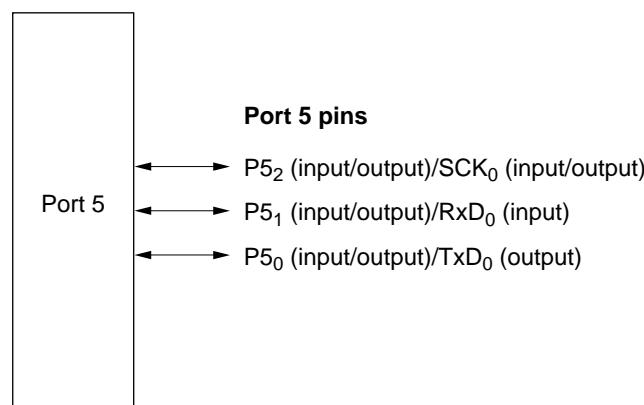


Figure 7-13 Port 5 Pin Configuration

### 7.6.2 Register Configuration and Descriptions

Table 7-10 summarizes the port 5 registers.

Table 7-10 Port 5 Registers

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

## Port 5 Data Direction Register (P5DDR)

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	P5 <sub>2</sub> DDR	P5 <sub>1</sub> DDR	P5 <sub>0</sub> DDR
Read/Write	—	—	—	—	—	W	W	W

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

P5DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

P5DDR is initialized to HF8 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P5DDR bit is set to 1, the corresponding pin remains in the output state.

If a transition to software standby mode occurs while port 5 is being used by the SCI, the SCI will be initialized, so the pin will revert to general-purpose input/output, controlled by P5DDR and P5DR.

## Port 5 Data Register (P5DR)

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	P5 <sub>2</sub>	P5 <sub>1</sub>	P5 <sub>0</sub>
Read/Write	1	1	1	1	1	0	0	0

P5DR is an 8-bit register that stores data for pins P5<sub>2</sub> to P5<sub>0</sub>. Bits 7 to 3 are reserved. They cannot be modified, and are always read as 1.

When a P5DDR bit is set to 1, if port 5 is read, the value in P5DR is obtained directly, regardless of the actual pin state. When a P5DDR bit is cleared to 0, if port 5 is read the pin state is obtained. This also applies to pins used as SCI pins.

P5DR is initialized to HF8 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.6.3 Pin Functions

Port 5 has the same pin functions in each operating mode. All pins can also be used as SCI0 input/output pins. Table 7-11 indicates the pin functions of port 5.

**Table 7-11 Port 5 Pin Functions**

Pin	Pin Functions and Selection Method				
P5 <sub>2</sub> /SCK <sub>0</sub>	Bit C/A in SMR of SCI0, bits CKE0 and CKE1 in SCR of SCI0, and bit P5 <sub>2</sub> DDR select the pin function as follows				
	CKE1	0			1
	C/A	0			—
	CKE0	0		1	—
	P5 <sub>2</sub> DDR	0	1	—	—
	Pin function	P5 <sub>2</sub> input	P5 <sub>2</sub> output	SCK <sub>0</sub> output	SCK <sub>0</sub> input
P5 <sub>1</sub> /RxD <sub>0</sub>	Bit RE in SCR of SCI0 and bit P5 <sub>1</sub> DDR select the pin function as follows				
	RE	0			1
	P5 <sub>1</sub> DDR	0		1	—
	Pin function	P5 <sub>1</sub> input		P5 <sub>1</sub> output	RxD <sub>0</sub> input
P5 <sub>0</sub> /TxD <sub>0</sub>	Bit TE in SCR of SCI0 and bit P5 <sub>0</sub> DDR select the pin function as follows				
	TE	0			1
	P5 <sub>0</sub> DDR	0		1	—
	Pin function	P5 <sub>0</sub> input		P5 <sub>0</sub> output	TxD <sub>0</sub> output

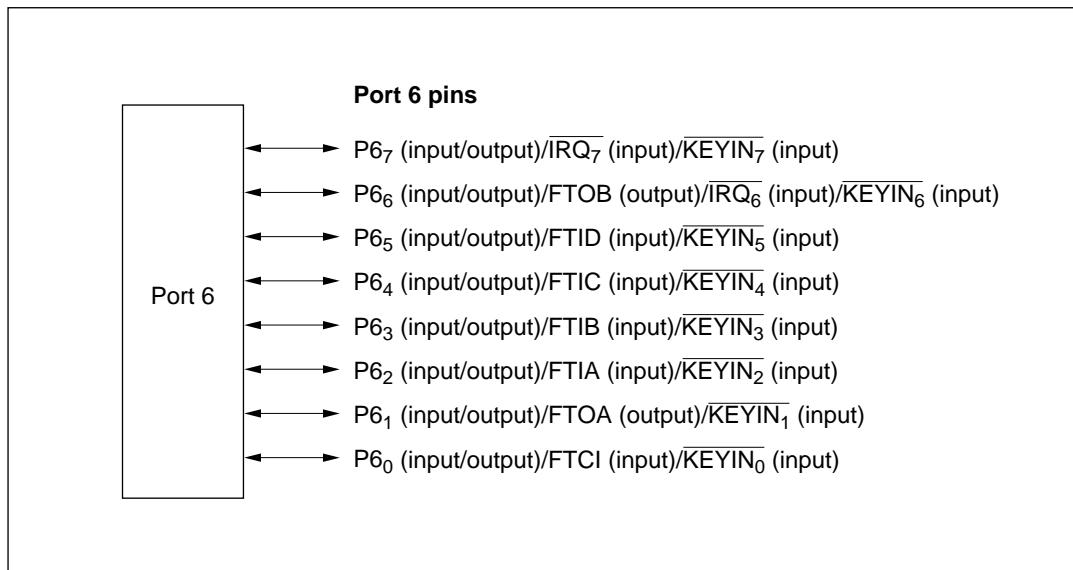
## 7.7 Port 6

### 7.7.1 Overview

Port 6 is an 8-bit input/output port that is multiplexed with input/output pins (FTOA, FTOB, FTIA to FTID, FTCI) of the 16-bit free-running timer (FRT), with key-sense input pins, and with  $\overline{\text{IRQ}_6}$  and  $\overline{\text{IRQ}_7}$  input pins. The port 6 pin functions are the same in all operating modes. Figure 7-14 shows the pin configuration of port 6.

Port 6 has built-in, programmable MOS input pull-up transistors.

Pins in port 6 can drive one TTL load and a 90-pF capacitive load. They can also drive a darlington transistor.



**Figure 7-14 Port 6 Pin Configuration**

## 7.7.2 Register Configuration and Descriptions

Table 7-12 summarizes the port 6 registers.

**Table 7-12 Port 6 Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port 6 data direction register	P6DDR	W	H'00	H'FFB9
Port 6 data register	P6DR	R/W	H'00	H'FFBB
Port 6 input pull-up control register	KMPCR	R/W	H'00	H'FFF2

### Port 6 Data Direction Register (P6DDR)

Bit	7	6	5	4	3	2	1	0
	P6 <sub>7</sub> DDR	P6 <sub>6</sub> DDR	P6 <sub>5</sub> DDR	P6 <sub>4</sub> DDR	P6 <sub>3</sub> DDR	P6 <sub>2</sub> DDR	P6 <sub>1</sub> DDR	P6 <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0

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

P6DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

P6DDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a P6DDR bit is set to 1, the corresponding pin remains in the output state.

If a transition to software standby mode occurs while port 6 is being used by the free-running timer, the timer will be initialized, so the pin will revert to general-purpose input/output, controlled by P6DDR and P6DR.

## Port 6 Data Register (P6DR)

Bit	7	6	5	4	3	2	1	0
	P6 <sub>7</sub>	P6 <sub>6</sub>	P6 <sub>5</sub>	P6 <sub>4</sub>	P6 <sub>3</sub>	P6 <sub>2</sub>	P6 <sub>1</sub>	P6 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

P6DR is an 8-bit register that stores data for pins P6<sub>7</sub> to P6<sub>0</sub>. When a P6DDR bit is set to 1, if port 6 is read, the value in P6DR is obtained directly, regardless of the actual pin state. When a P6DDR bit is cleared to 0, if port 6 is read the pin state is obtained. This also applies to pins used as FRT pins.

P6DR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

## Port 6 Input Pull-Up Control Register (KMPCR)

Bit	7	6	5	4	3	2	1	0
	KM <sub>7</sub> PCR	KM <sub>6</sub> PCR	KM <sub>5</sub> PCR	KM <sub>4</sub> PCR	KM <sub>3</sub> PCR	KM <sub>2</sub> PCR	KM <sub>1</sub> PCR	KM <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

KMPCR is an 8-bit readable/writable register that controls the input pull-up transistors in port 6. If a P6DDR bit is cleared to 0 (designating input) and the corresponding KMPCR bit is set to 1, the input pull-up transistor is turned on.

KMPCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.7.3 Pin Functions

Port 6 has the same pin functions in all operating modes. The pins are multiplexed with FRT input/output,  $\overline{IRQ_6}$  and  $\overline{IRQ_7}$  input, and key-sense input. Table 7-13 indicates the pin functions of port 6.

**Table 7-13 Port 6 Pin Functions**

Pin	Pin Functions and Selection Method			
P6 <sub>7</sub> /IRQ <sub>7</sub> / KEYIN <sub>7</sub>	P6 <sub>7</sub> DDR	0	1	
	Pin function	P6 <sub>7</sub> input		P6 <sub>7</sub> output
			IRQ <sub>7</sub> input or KEYIN <sub>7</sub> input	
$\overline{IRQ_7}$ input is usable when bit IRQ7E is set to 1 in IER				
P6 <sub>6</sub> /FTOB/ IRQ <sub>6</sub> /KEYIN <sub>6</sub>	Bit OEB in TOCR of the FRT and bit P6 <sub>6</sub> DDR select the pin function as follows			
	OEB	0	1	1
	P6 <sub>6</sub> DDR	0	1	0 1
	Pin function	P6 <sub>6</sub> input	P6 <sub>6</sub> output	FTOB output
			IRQ <sub>6</sub> input or KEYIN <sub>6</sub> input	
$\overline{IRQ_6}$ input is usable when bit IRQ6E is set to 1 in IER				
P6 <sub>5</sub> /FTID/ KEYIN <sub>5</sub>	P6 <sub>5</sub> DDR	0	1	
	Pin function	P6 <sub>5</sub> input		P6 <sub>5</sub> output
			FTID input or KEYIN <sub>5</sub> input	
P6 <sub>4</sub> /FTIC/ KEYIN <sub>4</sub>	P6 <sub>4</sub> DDR	0	1	
	Pin function	P6 <sub>4</sub> input		P6 <sub>4</sub> output
			FTIC input or KEYIN <sub>4</sub> input	

**Table 7-13 Port 6 Pin Functions (cont)**

Pin	Pin Functions and Selection Method		
P6 <sub>3</sub> /FTIB/ KEYIN <sub>3</sub>	P6 <sub>3</sub> DDR	0	1
	Pin function	P6 <sub>3</sub> input	P6 <sub>3</sub> output
		FTIB input or KEYIN <sub>3</sub> input	
P6 <sub>2</sub> /FTIA/ KEYIN <sub>2</sub>	P6 <sub>2</sub> DDR	0	1
	Pin function	P6 <sub>2</sub> input	P6 <sub>2</sub> output
		FTIA input or KEYIN <sub>2</sub> input	
P6 <sub>1</sub> /FTOA/ KEYIN <sub>1</sub>	Bit OEA in TOCR of the FRT and bit P6 <sub>1</sub> DDR select the pin function as follows		
	OE <sub>A</sub>	0	1
	P6 <sub>1</sub> DDR	0	1
	Pin function	P6 <sub>1</sub> input	P6 <sub>1</sub> output
		FTOA output KEYIN <sub>1</sub> input	
P6 <sub>0</sub> /FTCI/ KEYIN <sub>0</sub>	P6 <sub>0</sub> DDR	0	1
	Pin function	P6 <sub>0</sub> input	P6 <sub>0</sub> output
		FTCI input or KEYIN <sub>0</sub> input	
FTCI input is usable when bits CKS1 to CKS0 in TCR of the FRT select an external clock source			

#### 7.7.4 Input Pull-Up Transistors

Port 6 has built-in programmable input pull-up transistors. The pull-up for each bit can be turned on and off individually. To turn on an input pull-up, set the corresponding KMPCR bit to 1 and clear the corresponding P6DDR bit to 0. KMPCR is cleared to H'00 by a reset and in hardware standby mode, turning all input pull-ups off. In software standby mode, the previous state is maintained.

Table 7-14 indicates the states of the input pull-up transistors in each operating mode.

**Table 7-14 States of Input Pull-Up Transistors (Port 6)**

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	On/off	On/off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

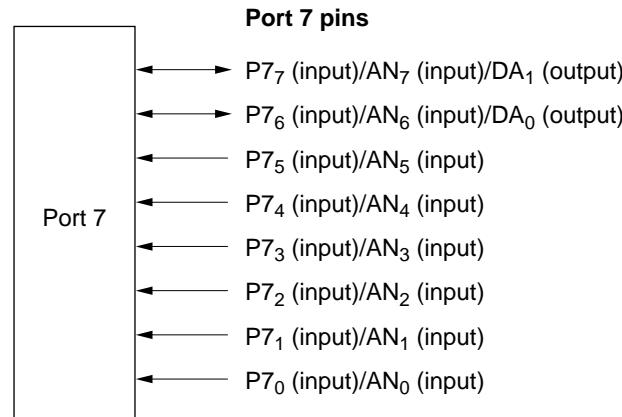
Notes: Off: The input pull-up transistor is always off.

On/off: The input pull-up transistor is on if KMPCR = 1 and P6DDR = 0, but off otherwise.

## 7.8 Port 7

### 7.8.1 Overview

Port 7 is an 8-bit input port that also provides the analog input pins for the A/D converter and analog output pins for the D/A converter. The pin functions are the same in all modes. Figure 7-15 shows the pin configuration of port 7.



**Figure 7-15 Port 7 Pin Configuration**

## 7.8.2 Register Configuration and Descriptions

Table 7-15 summarizes the port 7 registers. Port 7 is an input port, so there is no data direction register.

**Table 7-15 Port 7 Register**

Name	Abbreviation	Read/Write	Initial Value	Address
Port 7 input data register	P7PIN	R	Undetermined	H'FFBE

Note: The port 7 input data register (P7PIN) has the same address as the port B data direction register (PBDDR).

### Port 7 Input Data Register (P7PIN)

Bit	7	6	5	4	3	2	1	0
	P7 <sub>7</sub>	P7 <sub>6</sub>	P7 <sub>5</sub>	P7 <sub>4</sub>	P7 <sub>3</sub>	P7 <sub>2</sub>	P7 <sub>1</sub>	P7 <sub>0</sub>
Initial value	—*	—*	—*	—*	—*	—*	—*	—*
Read/Write	R	R	R	R	R	R	R	R

Note: \* Depends on the levels of pins P7<sub>7</sub> to P7<sub>0</sub>.

When P7PIN is read, the pin states are always read. P7PIN is a read-only register and cannot be written to. Write access results in writing to PBDDR.

## 7.9 Port 8

### 7.9.1 Overview

Port 8 is a 7-bit input/output port that is multiplexed with host interface (HIF) input pins ( $HA_0$ ,  $GA_{20}$ ,  $\overline{CS}_1$ ,  $\overline{IOR}$ ,  $\overline{IOW}$ ,  $CS_2$ ), with input/output pins ( $TxD_1$ ,  $RxD_1$ ,  $SCK_1$ ) of serial communication interface 1, with the I<sup>2</sup>C clock input/output pin (SCL), and with interrupt input pins ( $\overline{IRQ}_5$  to  $\overline{IRQ}_3$ ).

Figure 7-16 shows the pin configuration of port 8. The configuration of the pin functions of pins P8<sub>5</sub> and P8<sub>4</sub> will depend on the value of bit STAC in STCR. Pins P8<sub>6</sub> and P8<sub>3</sub> to P8<sub>0</sub> are unaffected by bit STAC.

Pins in port 8 can drive one TTL load and a 30-pF capacitive load. They can also drive a darlington transistor. Pin P8<sub>6</sub> can be driven as a bus buffer, as shown in section 13, I<sup>2</sup>C Bus Interface.

Port 8	Port 8 pins	Pin configuration in master mode, or when STAC bit is 1
	P8 <sub>6</sub> /SCK <sub>1</sub> / $\overline{IRQ}_5$ /SCL	P8 <sub>6</sub> (input/output)/ $\overline{IRQ}_5$ (input)/SCK <sub>1</sub> (input/output)
	P8 <sub>5</sub> /RxD <sub>1</sub> / $\overline{IRQ}_4$ / $\overline{CS}_2$	P8 <sub>5</sub> (input/output)/ $\overline{IRQ}_4$ (input)/RxD <sub>1</sub> (input)
	P8 <sub>4</sub> /Tx <sub>D1</sub> / $\overline{IRQ}_3$ / $\overline{IOW}$	P8 <sub>4</sub> (input/output)/ $\overline{IRQ}_3$ (input)/Tx <sub>D1</sub> (output)
	P8 <sub>3</sub> / $\overline{IOR}$	P8 <sub>3</sub> (input/output)
	P8 <sub>2</sub> / $\overline{CS}_1$	P8 <sub>2</sub> (input/output)
	P8 <sub>1</sub> /GA <sub>20</sub>	P8 <sub>1</sub> (input/output)
	P8 <sub>0</sub> /HA <sub>0</sub>	P8 <sub>0</sub> (input/output)
Pin configuration in slave mode		
When STAC bit is 0		
$\overline{IRQ}_5$ (input)/SCK <sub>1</sub> (input/output)/SCL (input/output)		
$\overline{IRQ}_4$ (input)/ $\overline{CS}_2$ (input)		
$\overline{IRQ}_3$ (input)/ $\overline{IOW}$ (input)		
$\overline{IOR}$ (input)		
$\overline{CS}_1$ (input)		
P8 <sub>1</sub> (input/output)/GA <sub>20</sub> (output)		
HA <sub>0</sub> (input)		

Figure 7-16 Port 8 Pin Configuration

## 7.9.2 Register Configuration and Descriptions

Table 7-16 summarizes the port 8 registers.

**Table 7-16 Port 8 Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port 8 data direction register	P8DDR	W	H'80	H'FFBD
Port 8 data register	P8DR	R/W	H'80	H'FFBF

Note: The port 8 data direction register (P8DDR) has the same address as the port B input data register (PBPIN).

### Port 8 Data Direction Register (P8DDR)

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

P8DDR is an 8-bit register that controls the input/output direction of each pin in port 8. A pin functions as an output pin if the corresponding P8DDR bit is set to 1, and as an input pin if this bit is cleared to 0. P8DDR is a write-only register. Read data is invalid. If read, all bits always read 1. Bit 7 is a reserved bit that always reads 1.

P8DDR is initialized to H'80 by a reset and in hardware standby mode. In software standby mode P8DDR retains its existing values, so if a transition to software standby mode occurs while a P8DDR bit is set to 1, the corresponding pin remains in the output state.

## Port 8 Data Register (P8DR)

Bit	7	6	5	4	3	2	1	0
	—	P8 <sub>6</sub>	P8 <sub>5</sub>	P8 <sub>4</sub>	P8 <sub>3</sub>	P8 <sub>2</sub>	P8 <sub>1</sub>	P8 <sub>0</sub>
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W						

P8DR is an 8-bit register that stores data for pins P8<sub>6</sub> to P8<sub>0</sub>. Bit 7 is a reserved bit that always reads 1.

When a P8DDR bit is set to 1, if port 8 is read, the value in P8DR is obtained directly, regardless of the actual pin state. When a P8DDR bit is cleared to 0, if port 8 is read the pin state is obtained. This also applies to pins used by on-chip supporting modules.

P8DR is initialized to H'80 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.9.3 Pin Functions

Pins P8<sub>6</sub> to P8<sub>0</sub> are multiplexed with HIF input/output, SCI1 input/output, I<sup>2</sup>C clock input/output, and  $\overline{\text{IRQ}_5}$  to  $\overline{\text{IRQ}_3}$  input. Table 7-17 indicates the functions of pins P8<sub>6</sub> to P8<sub>0</sub>.

**Table 7-17 Port 8 Pin Functions**

Pin	Pin Functions and Selection Method					
P8 <sub>6</sub> / $\overline{\text{IRQ}_5}$ / SCK <sub>1</sub> /SCL	Bit C/A in SMR of SCI1, bits CKE0 and CKE1 in SCR of SCI1, bit ICE in ICCR, and bit P8 <sub>6</sub> DDR select the pin function as follows					
	ICE	0				1
	CKE1	0				—
	C/A	0		1	—	—
	CKE0	0		1	—	—
	P8 <sub>6</sub> DDR	0	1	—	—	—
	Pin function	P8 <sub>6</sub> input	P8 <sub>6</sub> output	SCK <sub>1</sub> output	SCK <sub>1</sub> output	SCL input/ output
		$\overline{\text{IRQ}_5}$ input				

$\overline{\text{IRQ}_5}$  input is usable when bit IRQ5E is set to 1 in IER

P8 <sub>5</sub> / $\overline{\text{IRQ}_4}$ / CS <sub>2</sub> /RXD <sub>1</sub>	Bit RE in SCR of SCI1, bit STAC in STCR, bit P8 <sub>5</sub> DDR, and the operating mode select the pin function as follows					
	Operating mode	Slave mode			Not slave mode	
	STAC	0	1		—	
	RE	—	0		1	0
	P8 <sub>5</sub> DDR	—	0	1	—	0
	Pin function	CS <sub>2</sub> input	P8 <sub>5</sub> input	P8 <sub>5</sub> output	RxD <sub>1</sub> input	P8 <sub>5</sub> output
		$\overline{\text{IRQ}_4}$ input				

$\overline{\text{IRQ}_4}$  input is usable when bit IRQ4E is set to 1 in IER

**Table 7-17 Port 8 Pin Functions**

Pin	Pin Functions and Selection Method												
P8 <sub>4</sub> /IRQ <sub>3</sub> / IOW/TxD <sub>1</sub>	Bit TE in SCR of SCI1, bit STAC in STCR, bit P8 <sub>4</sub> DDR, and the operating mode select the pin function as follows												
	Operating mode	Slave mode				Not slave mode							
	STAC	0	1		—								
	TE	—	0		1	0							
	P8 <sub>4</sub> DDR	—	0	1	—	0	1						
	Pin function	$\overline{\text{IOW}}$ input	P8 <sub>4</sub> input	P8 <sub>4</sub> output	TxD <sub>1</sub> output	P8 <sub>4</sub> input	P8 <sub>4</sub> output						
		$\overline{\text{IRQ}_3}$ input											
IRQ <sub>3</sub> input is usable when bit IRQ3E is set to 1 in IER													
P8 <sub>3</sub> /IOR	Bit P8 <sub>3</sub> DDR and the operating mode select the pin function as follows												
	Operating mode	Slave mode			Not slave mode								
	P8 <sub>3</sub> DDR	—		0		1							
	Pin function	$\overline{\text{IOR}}$ input		P8 <sub>3</sub> input		P8 <sub>3</sub> output							
P8 <sub>2</sub> /CS <sub>1</sub>	Bit P8 <sub>2</sub> DDR and the operating mode select the pin function as follows												
	Operating mode	Slave mode			Not slave mode								
	P8 <sub>2</sub> DDR	—		0		1							
	Pin function	$\overline{\text{CS}_1}$ input		P8 <sub>2</sub> input		P8 <sub>2</sub> output							
P8 <sub>1</sub> /GA <sub>20</sub>	Bit P8 <sub>1</sub> DDR and the operating mode select the pin function as follows												
	P8 <sub>1</sub> DDR	0	1										
	FGA20E	—	0	1									
	Operating mode	—			Not slave mode		Slave mode						
	Pin function	P8 <sub>1</sub> input	P8 <sub>1</sub> output			GA <sub>20</sub> output							
P8 <sub>0</sub> /HA <sub>0</sub>	Bit P8 <sub>0</sub> DDR and the operating mode select the pin function as follows												
	Operating mode	Slave mode			Not slave mode								
	P8 <sub>0</sub> DDR	—		0		1							
	Pin function	HA <sub>0</sub> input		P8 <sub>0</sub> input		P8 <sub>0</sub> output							

## 7.10 Port 9

### 7.10.1 Overview

Port 9 is an 8-bit input/output port that is multiplexed with interrupt input pins ( $\overline{IRQ_0}$  to  $\overline{IRQ_2}$ ), input/output pins for bus control signals ( $\overline{RD}$ ,  $\overline{WR}$ ,  $\overline{AS}$ ,  $\overline{WAIT}$ ), an input pin ( $\overline{ADTRG}$ ) for the A/D converter, an output pin ( $\emptyset$ ) for the system clock, host interface (HIF) input pins ( $\overline{ECS_2}$ ,  $\overline{EIOW}$ ), and the I<sup>2</sup>C data input/output pin (SDA). Figure 7-17 shows the pin configuration of port 9. The functions of pins P9<sub>1</sub> and P9<sub>0</sub> are configured according to bit STAC in STCR. Pins P9<sub>7</sub> to P9<sub>2</sub> are unaffected by bit STAC.

Pins in port 9 can drive one TTL load and a 90-pF capacitive load. They can also drive a darlington transistor. Pin P9<sub>7</sub> can be driven as a bus buffer, as shown in section 13, I<sup>2</sup>C Bus Interface.

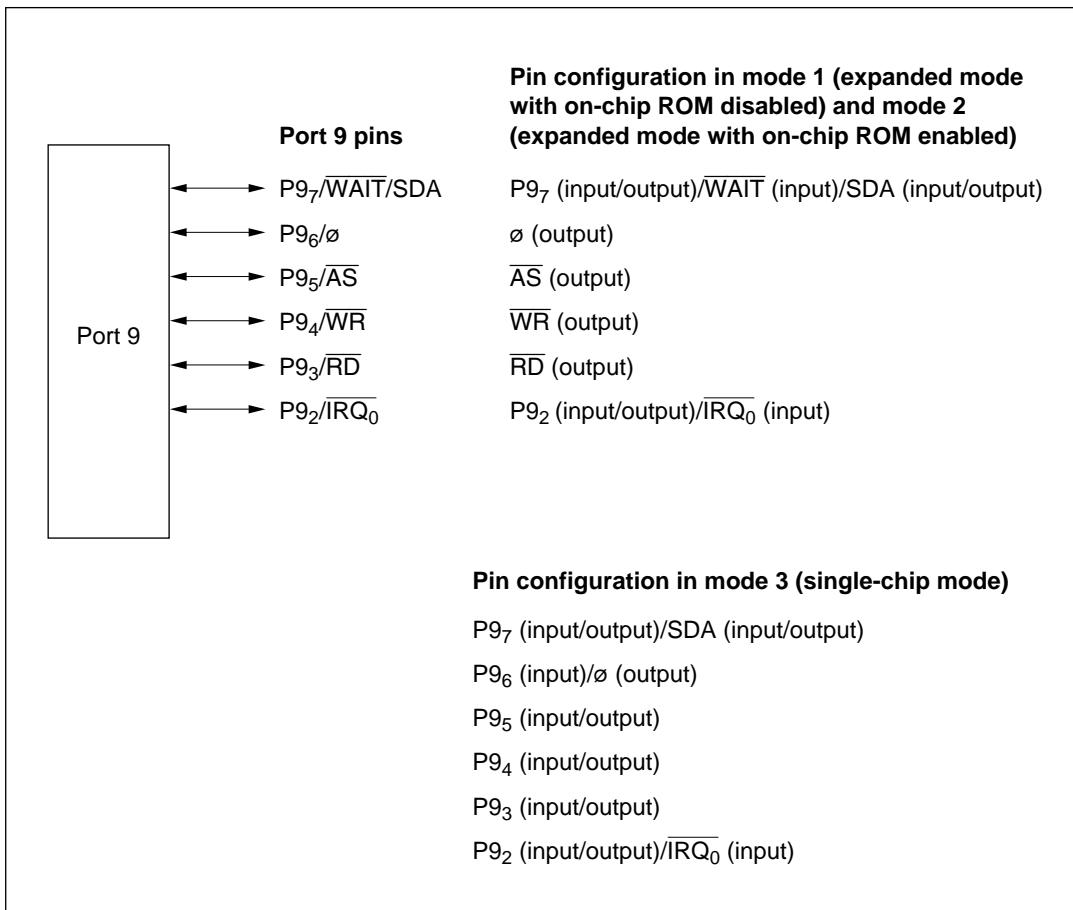
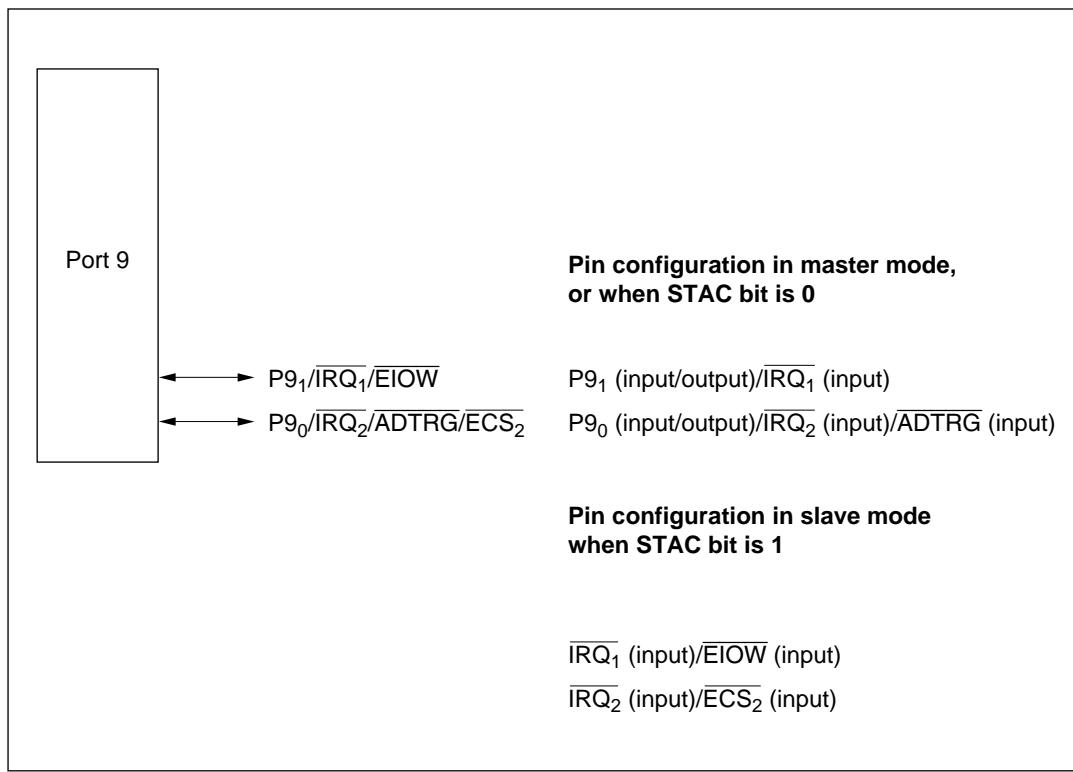


Figure 7-17 Port 9 Pin Configuration



**Figure 7-17 Port 9 Pin Configuration (cont)**

### 7.10.2 Register Configuration and Descriptions

Table 7-18 summarizes the port 9 registers.

**Table 7-18 Port 9 Registers**

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

Notes:

1. Bit 6 is read-only.

2. Bit 6 is undetermined. Other bits are initially 0.

## Port 9 Data Direction Register (P9DDR)

Bit	7	6	5	4	3	2	1	0
	P9 <sub>7</sub> DDR	P9 <sub>6</sub> DDR	P9 <sub>5</sub> DDR	P9 <sub>4</sub> DDR	P9 <sub>3</sub> DDR	P9 <sub>2</sub> DDR	P9 <sub>1</sub> DDR	P9 <sub>0</sub> DDR
Modes 1, 2								
Initial value	0	1	0	0	0	0	0	0
Read/Write	W	—	W	W	W	W	W	W
Mode 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

P9DDR is an 8-bit register that controls the input/output direction of each pin in port 9. A pin functions as an output pin if the corresponding P9DDR bit is set to 1, and as an input pin if this bit is cleared to 0. In modes 1 and 2, P9<sub>6</sub>DDR is fixed at 1 and cannot be modified.

P9DDR is a write-only register. Read data is invalid. If read, all bits always read 1.

P9DDR is initialized by a reset and in hardware standby mode. The initial value is H'40 in modes 1 and 2, and H'00 in mode 3. In software standby mode P9DDR retains its existing values, so if a transition to software standby mode occurs while a P9DDR bit is set to 1, the corresponding pin remains in the output state.

## Port 9 Data Register (P9DR)

Bit	7	6	5	4	3	2	1	0
	P9 <sub>7</sub>	P9 <sub>6</sub>	P9 <sub>5</sub>	P9 <sub>4</sub>	P9 <sub>3</sub>	P9 <sub>2</sub>	P9 <sub>1</sub>	P9 <sub>0</sub>
Initial value	0	—*	0	0	0	0	0	0
Read/Write	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* Determined by the level at pin P9<sub>6</sub>.

P9DR is an 8-bit register that stores data for pins P9<sub>7</sub> to P9<sub>0</sub>. When a P9DDR bit is set to 1, if port 9 is read, the value in P9DR is obtained directly, regardless of the actual pin state, except for P9<sub>6</sub>. When a P9DDR bit is cleared to 0, if port 9 is read the pin state is obtained. This also applies to pins used by on-chip supporting modules and for bus control signals. P9<sub>6</sub> always returns the pin state.

P9DR pins other than P9<sub>6</sub> are initialized to 0 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

### 7.10.3 Pin Functions

Port 9 has one set of pin functions in modes 1 and 2, and a different set of pin functions in mode 3. The pins are multiplexed with  $\overline{\text{IRQ}_0}$  to  $\overline{\text{IRQ}_2}$  input, bus control signal input/output, A/D converter input, system clock ( $\phi$ ) output, host interface input ( $\overline{\text{ECS}_2}$ ,  $\overline{\text{EIO}_W}$ ), and I<sup>2</sup>C data input/output (SDA). Table 7-19 indicates the pin functions of port 9.

**Table 7-19 Port 9 Pin Functions**

Pin	Pin Functions and Selection Method									
P9 <sub>7</sub> /WAIT/SDA	Bit ICE in ICCR, bit P9 <sub>7</sub> DDR, the wait mode as determined by WSCR, and the operating mode select the pin function as follows									
	Operating mode	Modes 1 and 2						Mode 3		
	Wait mode	$\overline{\text{WAIT}}$ used	$\overline{\text{WAIT}}$ not used				—			
	ICE	—	0		1	0		1		
	P9 <sub>7</sub> DDR	—	0	1	—	0	1	—		
	Pin function	$\overline{\text{WAIT}}$ input	P9 <sub>7</sub> input	P9 <sub>7</sub> output	SDA input/output	P9 <sub>7</sub> input	P9 <sub>7</sub> output	SDA input/output		
P9 <sub>6</sub> / $\phi$	Bit P9 <sub>6</sub> DDR and the operating mode select the pin function as follows									
	Operating mode	Modes 1 and 2			Mode 3					
	P9 <sub>6</sub> DDR	Always 1			0		1			
	Pin function	$\phi$ output			P9 <sub>6</sub> input		$\phi$ output			
P9 <sub>5</sub> /AS	Bit P9 <sub>5</sub> DDR and the operating mode select the pin function as follows									
	Operating mode	Modes 1 and 2			Mode 3					
	P9 <sub>5</sub> DDR	—			0		1			
	Pin function	$\overline{\text{AS}}$ output			P9 <sub>5</sub> input		P9 <sub>5</sub> output			
P9 <sub>4</sub> /WR	Bit P9 <sub>4</sub> DDR and the operating mode select the pin function as follows									
	Operating mode	Modes 1 and 2			Mode 3					
	P9 <sub>4</sub> DDR	—			0		1			
	Pin function	$\overline{\text{WR}}$ output			P9 <sub>4</sub> input		P9 <sub>4</sub> output			

**Table 7-19 Port 9 Pin Functions (cont)**

Pin	Pin Functions and Selection Method										
P9 <sub>3</sub> /RD	Bit P9 <sub>3</sub> DDR and the operating mode select the pin function as follows										
	Operating mode	Modes 1 and 2		Mode 3							
	P9 <sub>3</sub> DDR	—		0	1						
	Pin function	RD output		P9 <sub>3</sub> input	P9 <sub>3</sub> output						
P9 <sub>2</sub> /IRQ <sub>0</sub>	P9 <sub>2</sub> DDR	0		1							
	Pin function	P9 <sub>2</sub> input		P9 <sub>2</sub> output							
		IRQ <sub>0</sub> input									
IRQ <sub>0</sub> input can be used when bit IRQ0E is set to 1 in IER											
P9 <sub>1</sub> /IRQ <sub>1</sub> /EIOW	Bit STAC in STCR, bit P9 <sub>1</sub> DDR, and the operating mode select the pin function as follows										
	Operating mode	Slave mode			Not slave mode						
	STAC	0		1	—						
	P9 <sub>1</sub> DDR	0	1	—	0 1						
	Pin function	P9 <sub>1</sub> input	P9 <sub>1</sub> output	EIOW input	P9 <sub>1</sub> input P9 <sub>1</sub> output						
IRQ <sub>1</sub> input can be used when bit IRQ1E is set to 1 in IER											
P9 <sub>0</sub> /IRQ <sub>2</sub> /ADTRG/ECS <sub>2</sub>	Bit STAC in STCR, bit P9 <sub>0</sub> DDR, and the operating mode select the pin function as follows										
	Operating mode	Slave mode			Not slave mode						
	STAC	0		1	—						
	P9 <sub>0</sub> DDR	0	1	—	0 1						
	Pin function	P9 <sub>0</sub> input	P9 <sub>0</sub> output	ECS <sub>2</sub> input	P9 <sub>0</sub> input P9 <sub>0</sub> output						
IRQ <sub>2</sub> input can be used when bit IRQ2E is set to 1 in IER											
ADTRG input can be used when bit TRGE is set to 1 in ADCR											

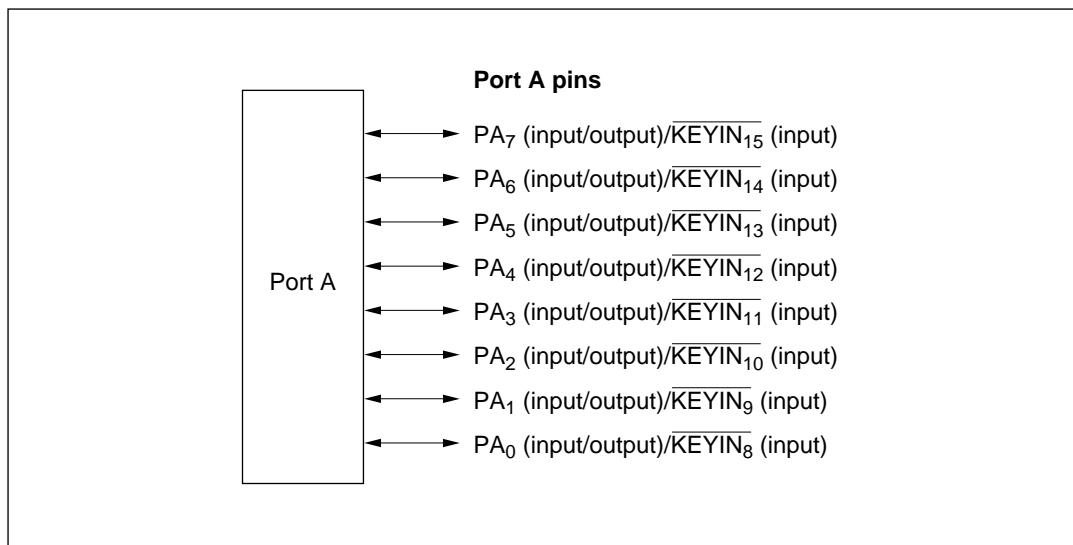
## 7.11 Port A

### 7.11.1 Overview

Port A is an 8-bit input/output port that is multiplexed with key-sense input pins. The port A pin functions are the same in all operating modes. Figure 7-18 shows the pin configuration of port A.

Port A has built-in, programming MOS input pull-up transistors.

Pins in port A can drive one TTL load and a 90-pF capacitive load. They can also drive a darlington transistor. Pins PA<sub>4</sub>, PA<sub>5</sub>, PA<sub>6</sub>, and PA<sub>7</sub> can be driven as bus buffers, as shown in section 13, I<sup>2</sup>C Bus Interface.



**Figure 7-18 Port A Pin Configuration**

### 7.11.2 Register Configuration and Descriptions

Table 7-20 summarizes the port A registers.

**Table 7-20 Port A Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port A data direction register	PADDR	W	H'00	H'FFAB
Port A output data register	PAODR	R/W	H'00	H'FFAA
Port A input data register	PAPIN	R	Undetermined	H'FFAB

Note: The data direction register (PADDR) and input data register (PAPIN) have the same address.

## Port A Data Direction Register (PADDR)

Bit	7	6	5	4	3	2	1	0
	PA <sub>7</sub> DDR	PA <sub>6</sub> DDR	PA <sub>5</sub> DDR	PA <sub>4</sub> DDR	PA <sub>3</sub> DDR	PA <sub>2</sub> DDR	PA <sub>1</sub> DDR	PA <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

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

PADDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a PADDR bit is set to 1, the corresponding pin remains in the output state.

## Port A Output Data Register (PAODR)

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

PAODR is an 8-bit register that stores data for pins PA<sub>7</sub> to PA<sub>0</sub>. PAODR can always be written to and read, regardless of the PADDR settings.

PAODR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

## Port A Input Data Register (PAPIN)

Bit	7	6	5	4	3	2	1	0
	PA <sub>7</sub>	PA <sub>6</sub>	PA <sub>5</sub>	PA <sub>4</sub>	PA <sub>3</sub>	PA <sub>2</sub>	PA <sub>1</sub>	PA <sub>0</sub>
Initial value	—*	—*	—*	—*	—*	—*	—*	—*
Read/Write	R	R	R	R	R	R	R	R

Note: \* Depends on the levels of pins PA<sub>7</sub> to PA<sub>0</sub>.

When PAPIN is read, the pin states are always read.

### 7.11.3 Pin Functions in Each Mode

Port A has the same pin functions in all operating modes. Table 7-21 indicates the pin functions of port A.

**Table 7-21 Port A Pin Functions**

Pin	Pin Functions and Selection Method		
PA <sub>7</sub> / <u>KEYIN</u> <sub>15</sub>	PA <sub>7</sub> DDR	0	1
	Pin function	PA <sub>7</sub> input	PA <sub>7</sub> output
			<u>KEYIN</u> <sub>15</sub> input
This pin is driven as a bus buffer when bit IICS is set to 1 in STCR			
PA <sub>6</sub> / <u>KEYIN</u> <sub>14</sub>	PA <sub>6</sub> DDR	0	1
	Pin function	PA <sub>6</sub> input	PA <sub>6</sub> output
			<u>KEYIN</u> <sub>14</sub> input
This pin is driven as a bus buffer when bit IICS is set to 1 in STCR			
PA <sub>5</sub> / <u>KEYIN</u> <sub>13</sub>	PA <sub>5</sub> DDR	0	1
	Pin function	PA <sub>5</sub> input	PA <sub>5</sub> output
			<u>KEYIN</u> <sub>13</sub> input
This pin is driven as a bus buffer when bit IICS is set to 1 in STCR			
PA <sub>4</sub> / <u>KEYIN</u> <sub>12</sub>	PA <sub>4</sub> DDR	0	1
	Pin function	PA <sub>4</sub> input	PA <sub>4</sub> output
			<u>KEYIN</u> <sub>12</sub> input
This pin is driven as a bus buffer when bit IICS is set to 1 in STCR			

**Table 7-21 Port A Pin Functions (cont)****Pin**      **Pin Functions and Selection Method**PA<sub>3</sub>/KEYIN<sub>11</sub>

PA <sub>3</sub> DDR	0	1
Pin function	PA <sub>3</sub> input	PA <sub>3</sub> output
KEYIN <sub>11</sub> input		

PA<sub>2</sub>/KEYIN<sub>10</sub>

PA <sub>2</sub> DDR	0	1
Pin function	PA <sub>2</sub> input	PA <sub>2</sub> output
KEYIN <sub>10</sub> input		

PA<sub>1</sub>/KEYIN<sub>9</sub>

PA <sub>1</sub> DDR	0	1
Pin function	PA <sub>1</sub> input	PA <sub>1</sub> output
KEYIN <sub>9</sub> input		

PA<sub>0</sub>/KEYIN<sub>8</sub>

PA <sub>0</sub> DDR	0	1
Pin function	PA <sub>0</sub> input	PA <sub>0</sub> output
KEYIN <sub>8</sub> input		

**7.11.4 Input Pull-Up Transistors**

Port A has built-in programmable input pull-up transistors that are available in all modes.

An input pull-up transistor is turned on if 1 is written in the corresponding PAODR bit while the corresponding PADDR bit is cleared to 0. The input pull-ups are turned off by a reset and in hardware standby mode.

Table 7-22 indicates the states of the input pull-up transistors in each operating mode.

**Table 7-22 States of Input Pull-Up Transistors (Port A)**

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	On/off	On/off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The input pull-up transistor is always off.

On/off: The input pull-up transistor is on if PAODR = 1 and PADDR = 0, but off otherwise.

## 7.12 Port B

### 7.12.1 Overview

Port B is an 8-bit input/output port that is multiplexed with the host interface data bus. The pin functions differ depending on the operating mode. Figure 7-19 shows the pin configuration of port B.

Port B has program-controllable built-in MOS input pull-up transistors.

Pins in port B can drive one TTL load and a 90-pF capacitive load. They can also drive a darlington transistor.

<b>Pin configuration in mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled)</b>			
<b>Port B pins</b>	<b>Master mode</b>	<b>Slave mode</b>	
Port B	PB <sub>7</sub> /XDB <sub>7</sub>	PB <sub>7</sub> (input/output)	XDB <sub>7</sub> (input/output)
	PB <sub>6</sub> /XDB <sub>6</sub>	PB <sub>6</sub> (input/output)	XDB <sub>6</sub> (input/output)
	PB <sub>5</sub> /XDB <sub>5</sub>	PB <sub>5</sub> (input/output)	XDB <sub>5</sub> (input/output)
	PB <sub>4</sub> /XDB <sub>4</sub>	PB <sub>4</sub> (input/output)	XDB <sub>4</sub> (input/output)
	PB <sub>3</sub> /XDB <sub>3</sub>	PB <sub>3</sub> (input/output)	XDB <sub>3</sub> (input/output)
	PB <sub>2</sub> /XDB <sub>2</sub>	PB <sub>2</sub> (input/output)	XDB <sub>2</sub> (input/output)
	PB <sub>1</sub> /XDB <sub>1</sub>	PB <sub>1</sub> (input/output)	XDB <sub>1</sub> (input/output)
	PB <sub>0</sub> /XDB <sub>0</sub>	PB <sub>0</sub> (input/output)	XDB <sub>0</sub> (input/output)
<b>Pin configuration in mode 3 (single-chip mode)</b>			
	PB <sub>7</sub> (input/output)		
	PB <sub>6</sub> (input/output)		
	PB <sub>5</sub> (input/output)		
	PB <sub>4</sub> (input/output)		
	PB <sub>3</sub> (input/output)		
	PB <sub>2</sub> (input/output)		
	PB <sub>1</sub> (input/output)		
	PB <sub>0</sub> (input/output)		

**Figure 7-19 Port B Pin Configuration**

## 7.12.2 Register Configuration and Descriptions

Table 7-23 summarizes the port B registers.

**Table 7-23 Port B Registers**

Name	Abbreviation	Read/Write	Initial Value	Address
Port B data direction register	PBDDR	W	H'00	H'FFBE
Port B output data register	PBODR	R/W	H'00	H'FFBC
Port B input data register	PBPIN	R	Undetermined	H'FFBD

Note: The port B data direction register (PBDDR) and port 7 input data register 7 (P7PIN) have the same address.

### Port B Data Direction Register (PBDDR)

Bit	7	6	5	4	3	2	1	0
	PB <sub>7</sub> DDR	PB <sub>6</sub> DDR	PB <sub>5</sub> DDR	PB <sub>4</sub> DDR	PB <sub>3</sub> DDR	PB <sub>2</sub> DDR	PB <sub>1</sub> DDR	PB <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PBDDR is an 8-bit register that controls the input/output direction of each pin in port B. A pin functions as an output pin if the corresponding PBDDR bit is set to 1, and as an input pin if this bit is cleared to 0.

PBDDR is a write-only register. Read data is invalid. If read, the values of the port 7 data input register (P7PIN) are returned, indicating the pin levels of port 7.

PBDDR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values, so if a transition to software standby mode occurs while a PBDDR bit is set to 1, the corresponding pin remains in the output state.

## Port B Output Data Register (PBODR)

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

PBODR is an 8-bit register that stores data for pins PB<sub>7</sub> to PB<sub>0</sub>. PBODR can always be written to and read, regardless of the PBDDR settings.

PBODR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode it retains its existing values.

## Port B Input Data Register (PBPIN)

Bit	7	6	5	4	3	2	1	0
	PB <sub>7</sub>	PB <sub>6</sub>	PB <sub>5</sub>	PB <sub>4</sub>	PB <sub>3</sub>	PB <sub>2</sub>	PB <sub>1</sub>	PB <sub>0</sub>
Initial value	—*	—*	—*	—*	—*	—*	—*	—*
Read/Write	R	R	R	R	R	R	R	R

Note: \* Depends on the levels of pins PB<sub>7</sub> to PB<sub>0</sub>.

When PBPIN is read, the pin states are always read.

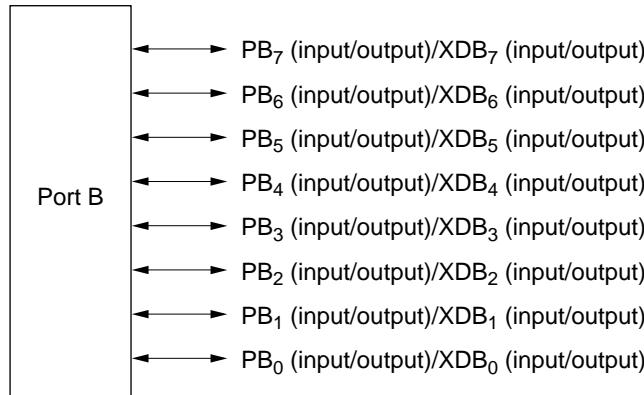
### 7.12.3 Pin Functions in Each Mode

Port B has different pin functions in different modes. A separate description for each mode is given below.

**Pin Functions in Modes 1 and 2:** In mode 1 (expanded mode with on-chip ROM disabled) and mode 2 (expanded mode with on-chip ROM enabled), when the host interface enable bit (HIE) is cleared to 0 in the system control register (SYSCR), port B is a general-purpose input/output port.

When the HIE bit is set to 1, selecting slave mode, port B becomes the host interface data bus (XDB<sub>7</sub> to XDB<sub>0</sub>). PBODR and PBDDR should be cleared to H'00 in slave mode. For details, see section 14, Host Interface.

Figure 7-20 shows the pin functions in modes 1 and 2.



**Figure 7-20 Pin Functions in Modes 1 and 2 (Port B)**

**Pin Functions in Mode 3:** In mode 3 (single-chip mode), each pin can be designated for general input or output. A pin becomes an output pin when its PBDDR bit is set to 1, and an input pin when this bit is cleared to 0. Figure 7-21 shows the pin functions in mode 3.

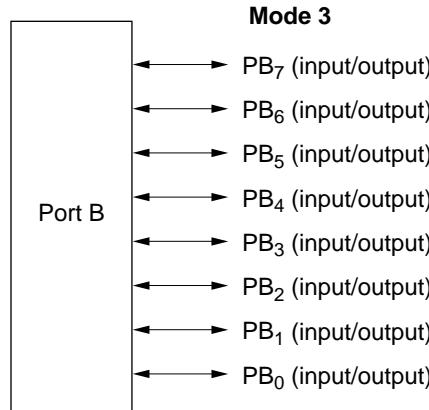


Figure 7-21 Pin Functions in Mode 3 (Port B)

#### 7.12.4 Input Pull-Up Transistors

Port B has built-in programmable input pull-up transistors that are available in mode 3. The pull-up for each bit can be turned on and off individually.

An input pull-up transistor is turned on in mode 3 if 1 is written in the corresponding PBODR bit while the corresponding PBDDR bit is cleared to 0.

The input pull-ups are turned off by a reset and in hardware standby mode. In software standby mode, the previous state is maintained.

Table 7-24 indicates the states of the input pull-up transistors in each operating mode.

Table 7-24 States of Input Pull-Up Transistors (Port B)

Mode	Reset	Hardware Standby	Software Standby	Other Operating Modes
1	Off	Off	On/off	On/off
2	Off	Off	On/off	On/off
3	Off	Off	On/off	On/off

Notes: Off: The input pull-up transistor is always off.

On/off: The input pull-up transistor is on if PBDR = 1 and PBDDR = 0, but off otherwise.

# Section 8 16-Bit Free-Running Timer

## 8.1 Overview

The H8/3437 Series has an on-chip 16-bit free-running timer (FRT) module that uses a 16-bit free-running counter as a time base. Applications of the FRT module include rectangular-wave output (up to two independent waveforms), input pulse width measurement, and measurement of external clock periods.

### 8.1.1 Features

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

- Selection of four clock sources

The free-running counter can be driven by an internal clock source ( $\phi_P/2$ ,  $\phi_P/8$ , or  $\phi_P/32$ ), or an external clock input (enabling use as an external event counter).

- Two independent comparators

Each comparator can generate an independent waveform.

- Four input capture channels

The current count can be captured on the rising or falling edge (selectable) of an input signal. The four input capture registers can be used separately, or in a buffer mode.

- Counter can be cleared under program control

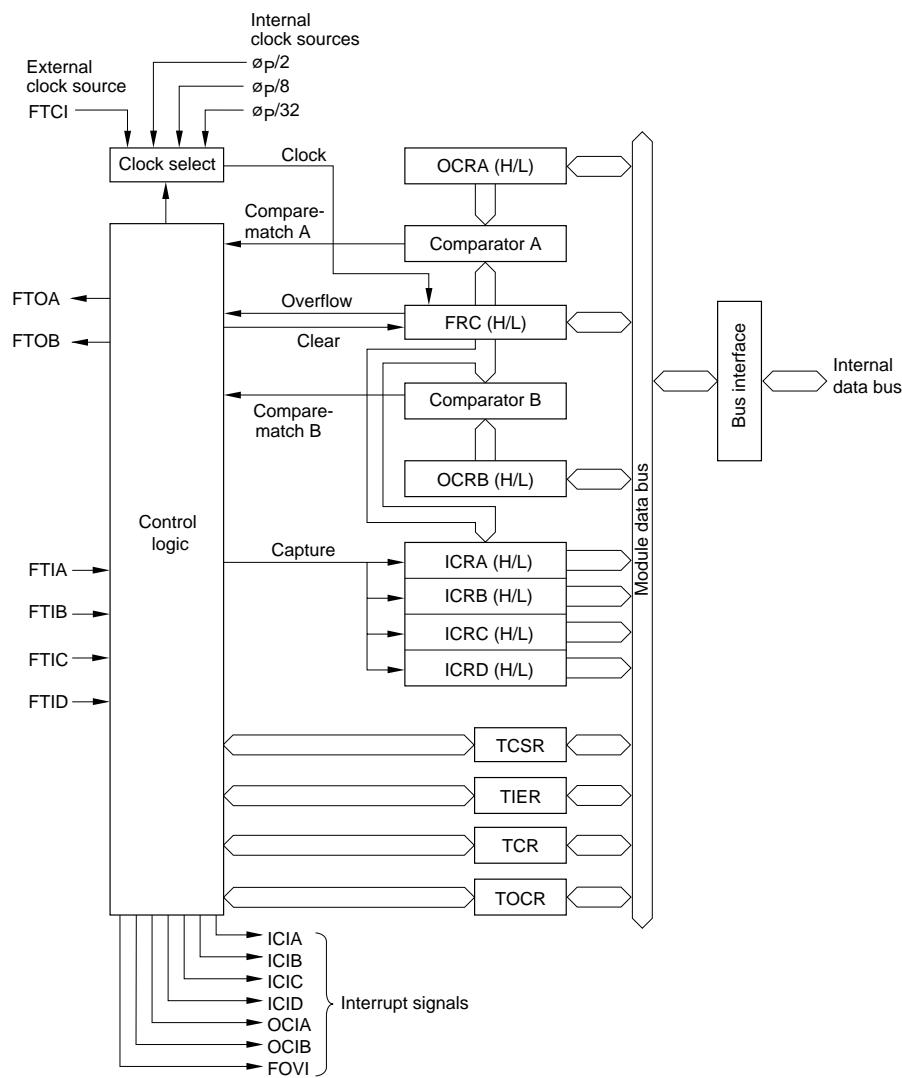
The free-running counters can be cleared on compare-match A.

- Seven independent interrupts

Compare-match A and B, input capture A to D, and overflow interrupts are requested independently.

## 8.1.2 Block Diagram

Figure 8-1 shows a block diagram of the free-running timer.



Legend

**FRC:** Free-running counter (16 bits)  
**OCRA, B:** Output compare register A, B (16 bits)  
**ICRA, B, C, D:** Input capture register A, B, C, D (16 bits)  
**TCSR:** Timer control/status register (8 bits)

**TIER:** Timer interrupt enable register (8 bits)  
**TCR:** Timer control register (8 bits)  
**TOCR:** Timer output compare control register (8 bits)

**Figure 8-1 Block Diagram of 16-Bit Free-Running Timer**

### 8.1.3 Input and Output Pins

Table 8-1 lists the input and output pins of the free-running timer module.

**Table 8-1 Input and Output Pins of Free-Running Timer Module**

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

### 8.1.4 Register Configuration

Table 8-2 lists the registers of the free-running timer module.

**Table 8-2 Register Configuration**

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

Notes:

1. Software can write a 0 to clear bits 7 to 1, but cannot write a 1 in these bits.
2. OCRA and OCRB share the same addresses. Access is controlled by the OCRS bit in TOCR.

**Table 8-2 Register Configuration (cont.)**

Name	Abbreviation	R/W	Initial Value	Address
Input capture register B (high)	ICRB (H)	R	H'00	H'FF9A
Input capture register B (low)	ICRB (L)	R	H'00	H'FF9B
Input capture register C (high)	ICRC (H)	R	H'00	H'FF9C
Input capture register C (low)	ICRC (L)	R	H'00	H'FF9D
Input capture register D (high)	ICRD (H)	R	H'00	H'FF9E
Input capture register D (low)	ICRD (L)	R	H'00	H'FF9F

## 8.2 Register Descriptions

### 8.2.1 Free-Running Counter (FRC)

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															

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

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

Because FRC is a 16-bit register, a temporary register (TEMP) is used when FRC is written or read. See section 8.3, CPU Interface, for details.

FRC is initialized to H'0000 by a reset and in the standby modes.

## 8.2.2 Output Compare Registers A and B (OCRA and OCRB)

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															

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

In addition, if the output enable bit (OEA or OEB) in the timer output compare control register (TOCR) is set to 1, when the output compare register and FRC values match, the logic level selected by the output level bit (OLVLA or OLVLB) in TOCR is output at the output compare pin (FTOA or FTOB). Following a reset, the FTOA and FTOB output levels are 0 until the first compare-match.

OCRA and OCRB share the same address. They are differentiated by the OCRS bit in TOCR. A temporary register (TEMP) is used for write access, as explained in section 8.3, CPU Interface.

OCRA and OCRB are initialized to H'FFFF by a reset and in the standby modes.

## 8.2.3 Input Capture Registers A to D (ICRA to IC RD)

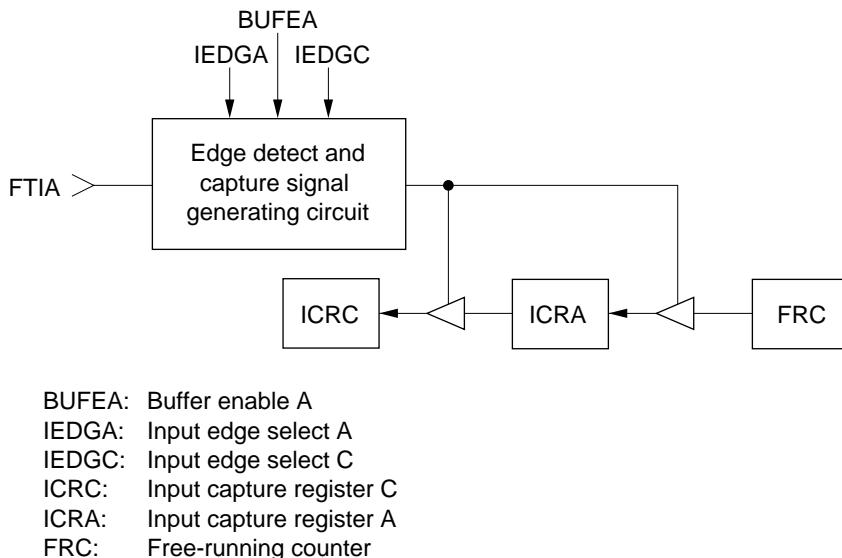
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

There are four input capture registers A to D, each of which is a 16-bit read-only register.

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

Note: \* The FRC contents are transferred to the input capture register regardless of the value of the input capture flag (ICFA/B/C/D).

Input capture can be buffered by using the input capture registers in pairs. When the BUFEA bit in TCR is set to 1, ICRC is used as a buffer register for ICRA as shown in figure 8-2. When an FTIA input is received, the old ICRA contents are moved into ICRC, and the new FRC count is copied into ICRA.



**Figure 8-2 Input Capture Buffering (Example)**

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

When input capture is buffered, if the two input edge bits are set to different values (**IEDGA**  $\neq$  **IEDGC** or **IEDGB**  $\neq$  **IEDGD**), then input capture is triggered on both the rising and falling edges of the **FTIA** or **FTIB** input signal. If the two input edge bits are set to the same value (**IEDGA** = **IEDGC** or **IEDGB** = **IEDGD**), then input capture is triggered on only one edge. See table 8-3.

**Table 8-3 Buffered Input Capture Edge Selection (Example)**

<b>IEDGA</b>	<b>IEDGC</b>	<b>Input Capture Edge</b>	
0	0	Captured on falling edge of input capture A (FTIA)	(Initial value)
0	1	Captured on both rising and falling edges of input capture A (FTIA)	
1	0		
1	1	Captured on rising edge of input capture A (FTIA)	

Because the input capture registers are 16-bit registers, a temporary register (**TEMP**) is used when they are read. See section 8.3, CPU Interface, for details.

To ensure input capture, the width of the input capture pulse should be at least 1.5 system clock periods ( $1.5 \cdot \phi$ ). When triggering is enabled on both edges, the input capture pulse width should be at least 2.5 system clock periods.

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

### 8.2.4 Timer Interrupt Enable Register (TIER)

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

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

TIER is initialized to H'01 by a reset and in the standby modes.

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

Bit 7	ICIAE	Description	
0	Input capture interrupt request A (ICIA) is disabled.		(Initial value)
1	Input capture interrupt request A (ICIA) is enabled.		

**Bit 6—Input Capture Interrupt B Enable (ICIBE):** This bit selects whether to request input capture interrupt B (ICIB) when input capture flag B (ICFB) in TCSR is set to 1.

Bit 6	ICIBE	Description	
0	Input capture interrupt request B (ICIB) is disabled.		(Initial value)
1	Input capture interrupt request B (ICIB) is enabled.		

**Bit 5—Input Capture Interrupt C Enable (ICICE):** This bit selects whether to request input capture interrupt C (ICIC) when input capture flag C (ICFC) in TCSR is set to 1.

Bit 5	ICICE	Description	
0	Input capture interrupt request C (ICIC) is disabled.		(Initial value)
1	Input capture interrupt request C (ICIC) is enabled.		

**Bit 4—Input Capture Interrupt D Enable (ICIDE):** This bit selects whether to request input capture interrupt D (ICID) when input capture flag D (ICFD) in TCSR is set to 1.

Bit 4	ICIDE	Description	
0		Input capture interrupt request D (ICID) is disabled.	(Initial value)
1		Input capture interrupt request D (ICID) is enabled.	

**Bit 3—Output Compare Interrupt A Enable (OCIAE):** This bit selects whether to request output compare interrupt A (OCIA) when output compare flag A (OCFA) in TCSR is set to 1.

Bit 3	OCIAE	Description	
0		Output compare interrupt request A (OCIA) is disabled.	(Initial value)
1		Output compare interrupt request A (OCIA) is enabled.	

**Bit 2—Output Compare Interrupt B Enable (OCIBE):** This bit selects whether to request output compare interrupt B (OCIB) when output compare flag B (OCFB) in TCSR is set to 1.

Bit 2	OCIBE	Description	
0		Output compare interrupt request B (OCIB) is disabled.	(Initial value)
1		Output compare interrupt request B (OCIB) is enabled.	

**Bit 1—Timer Overflow Interrupt Enable (OVIE):** This bit selects whether to request a free-running timer overflow interrupt (FOVI) when the timer overflow flag (OVF) in TCSR is set to 1.

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

**Bit 0—Reserved:** This bit cannot be modified and is always read as 1.

## 8.2.5 Timer Control/Status Register (TCSR)

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

Note: \* Software can write a 0 in bits 7 to 1 to clear the flags, but cannot write a 1 in these bits.

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

TCSR is initialized to H'00 by a reset and in the standby modes.

Timing is described in section 8.4, Operation.

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

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

Bit 7	
ICFA	Description
0	To clear ICFA, the CPU must read ICFA after it has been set to 1, then write a 0 in this bit. (Initial value)
1	This bit is set to 1 when an FTIA input signal causes the FRC value to be copied to ICRA.

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

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

Bit 6	
ICFB	Description
0	To clear ICFB, the CPU must read ICFB after it has been set to 1, then write a 0 in this bit. (Initial value)
1	This bit is set to 1 when an FTIB input signal causes the FRC value to be copied to ICRC.

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

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

**Bit 5**

**ICFC      Description**

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

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

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

**Bit 4**

**ICFD      Description**

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

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

**Bit 3**

**OCFA      Description**

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

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

Bit 2	OCFB	Description	
0		To clear OCFB, the CPU must read OCFB after it has been set to 1, then write a 0 in this bit.	(Initial value)
1		This bit is set to 1 when FRC = OCRB.	

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

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

**Bit 0—Counter Clear A (CCLRA):** This bit selects whether to clear 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.	

### 8.2.6 Timer Control Register (TCR)

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

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

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

**Bit 7—Input Edge Select A (IEDGA):** This bit selects the rising or falling edge of the input capture A signal (FTIA).

Bit 7		
IEDGA	Description	
0	Input capture A events are recognized on the falling edge of FTIA.	(Initial value)
1	Input capture A events are recognized on the rising edge of FTIA.	

**Bit 6—Input Edge Select B (IEDGB):** This bit selects the rising or falling edge of the input capture B signal (FTIB).

Bit 6		
IEDGB	Description	
0	Input capture B events are recognized on the falling edge of FTIB.	(Initial value)
1	Input capture B events are recognized on the rising edge of FTIB.	

**Bit 5—Input Edge Select C (IEDGC):** This bit selects the rising or falling edge of the input capture C signal (FTIC).

Bit 5		
IEDGC	Description	
0	Input capture C events are recognized on the falling edge of FTIC.	(Initial value)
1	Input capture C events are recognized on the rising edge of FTIC.	

**Bit 4—Input Edge Select D (IEDGD):** This bit selects the rising or falling edge of the input capture D signal (FTID).

Bit 4		
IEDGD	Description	
0	Input capture D events are recognized on the falling edge of FTID.	(Initial value)
1	Input capture D events are recognized on the rising edge of FTID.	

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

Bit 3		
BUFEA	Description	
0	ICRC is used for input capture C.	(Initial value)
1	ICRC is used as a buffer register for input capture A.	

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

Bit 2 BUFEB	Description	
0	ICRD is used for input capture D.	(Initial value)
1	ICRD is used as a buffer register for input capture B.	

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

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	$\phi_P/2$ internal clock source	(Initial value)
0	1	$\phi_P/8$ internal clock source	
1	0	$\phi_P/32$ internal clock source	
1	1	External clock source (rising edge)	

### 8.2.7 Timer Output Compare Control Register (TOCR)

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

TOCR is an 8-bit readable/writable register that enables output from the output compare pins, selects the output levels, and switches access between output compare registers A and B.

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

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

**Bit 4—Output Compare Register Select (OCRS):** OCRA and OCRB share the same address. When this address is accessed, the OCRS bit selects which register is accessed. This bit does not affect the operation of OCRA or OCRB.

Bit 4 OCRS	Description	
0	OCRA is selected.	(Initial value)
1	OCRB is selected.	

**Bit 3—Output Enable A (OEA):** This bit enables or disables output of the output compare A signal (FTOA).

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

**Bit 2—Output Enable B (OEB):** This bit enables or disables output of the output compare B signal (FTOB).

Bit 2		
OEB	Description	
0	Output compare B output is disabled.	(Initial value)
1	Output compare B output is enabled.	

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

Bit 1		
OLVLA	Description	
0	A 0 logic level is output for compare-match A.	(Initial value)
1	A 1 logic level is output for compare-match A.	

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

Bit 0		
OLVLB	Description	
0	A 0 logic level is output for compare-match B.	(Initial value)
1	A 1 logic level is output for compare-match B.	

## 8.3 CPU Interface

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

These registers are written and read as follows:

- **Register Write**

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

- **Register Read**

When the CPU reads the upper byte, the upper byte of data is sent to the CPU and the lower byte is placed in TEMP. When the CPU reads the lower byte, it receives the value in TEMP.

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

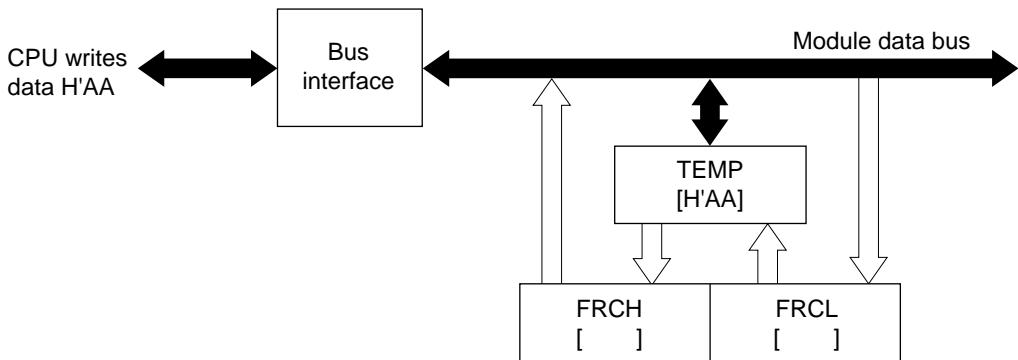
Figure 8-3 shows the data flow when FRC is accessed. The other registers are accessed in the same way. As an exception, when the CPU reads OCRA or OCRB, it reads both the upper and lower bytes directly, without using TEMP.

### Coding Examples

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

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

(1) Upper byte write



(2) Lower byte write

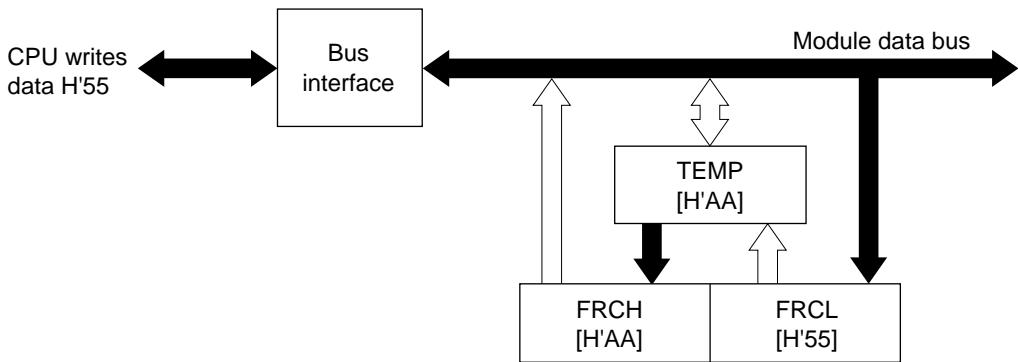
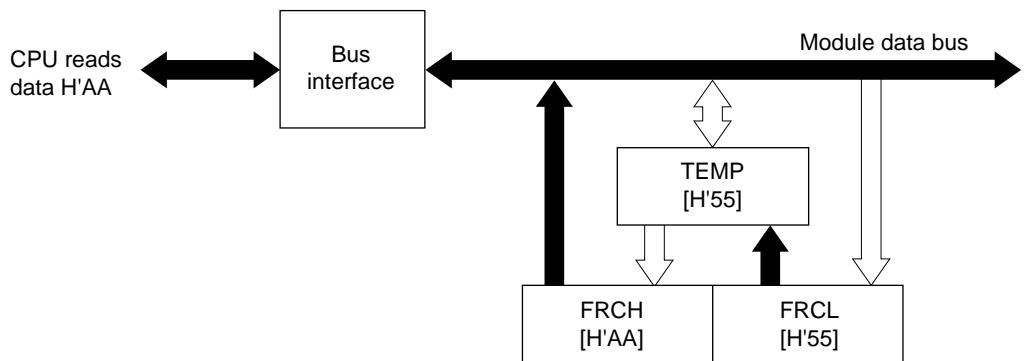
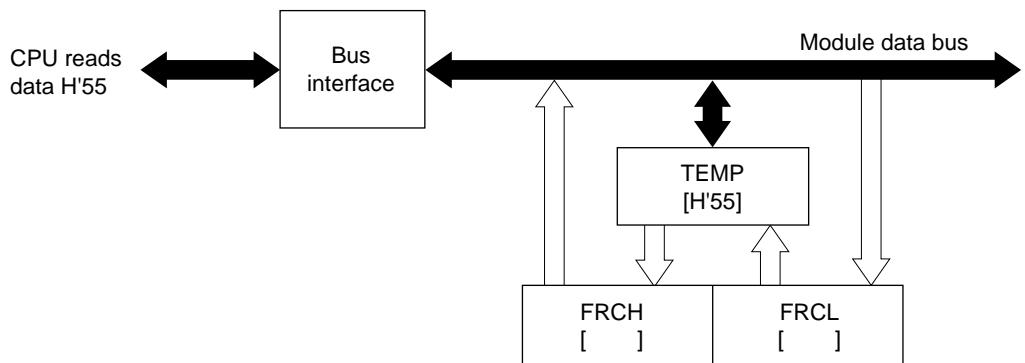


Figure 8-3 (a) Write Access to FRC (when CPU Writes H'AA55)

(1) Upper byte read



(2) Lower byte read



**Figure 8-3 (b) Read Access to FRC (when FRC Contains H'AA55)**

## 8.4 Operation

### 8.4.1 FRC Increment Timing

FRC increments on a pulse generated once for each period of the selected (internal or external) clock source. The clock source is selected by bits CKS0 and CKS1 in TCR.

**Internal Clock:** The internal clock sources ( $\phi_p/2$ ,  $\phi_p/8$ ,  $\phi_p/32$ ) are created from the system clock ( $\phi$ ) by a prescaler. FRC increments on a pulse generated from the falling edge of the prescaler output. See figure 8-4.

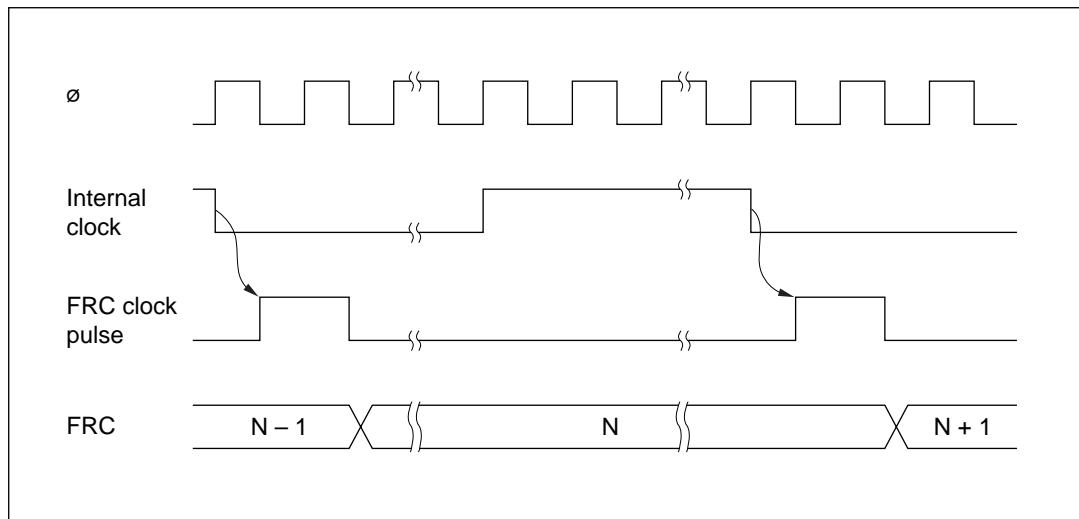
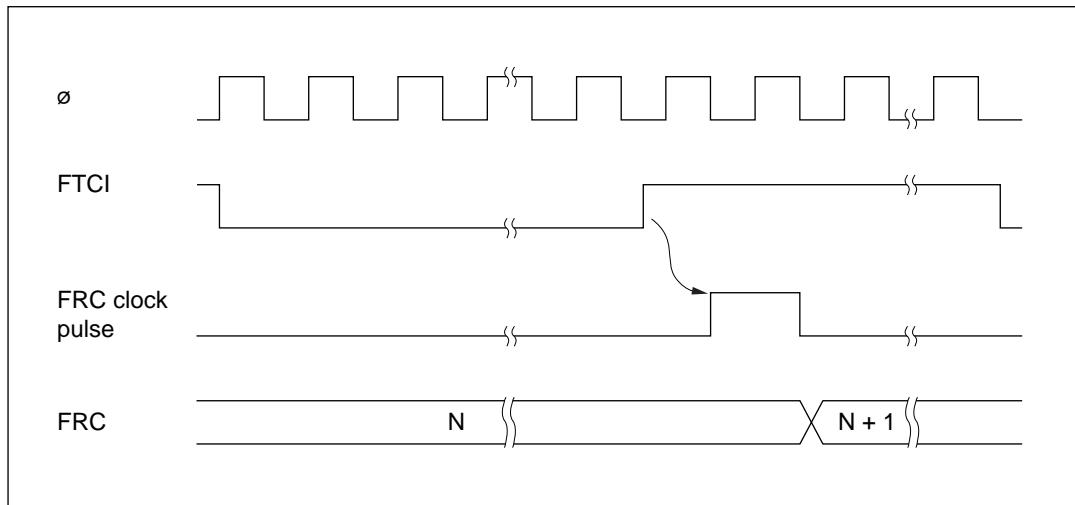


Figure 8-4 Increment Timing for Internal Clock Source

**External Clock:** If external clock input is selected, FRC increments on the rising edge of the FTCI clock signal. Figure 8-5 shows the increment timing.

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



**Figure 8-5 Increment Timing for External Clock Source**

#### 8.4.2 Output Compare Timing

When a compare-match occurs, the logic level selected by the output level bit (OLVLA or OLVLB) in TOCR is output at the output compare pin (FTOA or FTOB). Figure 8-6 shows the timing of this operation for compare-match A.

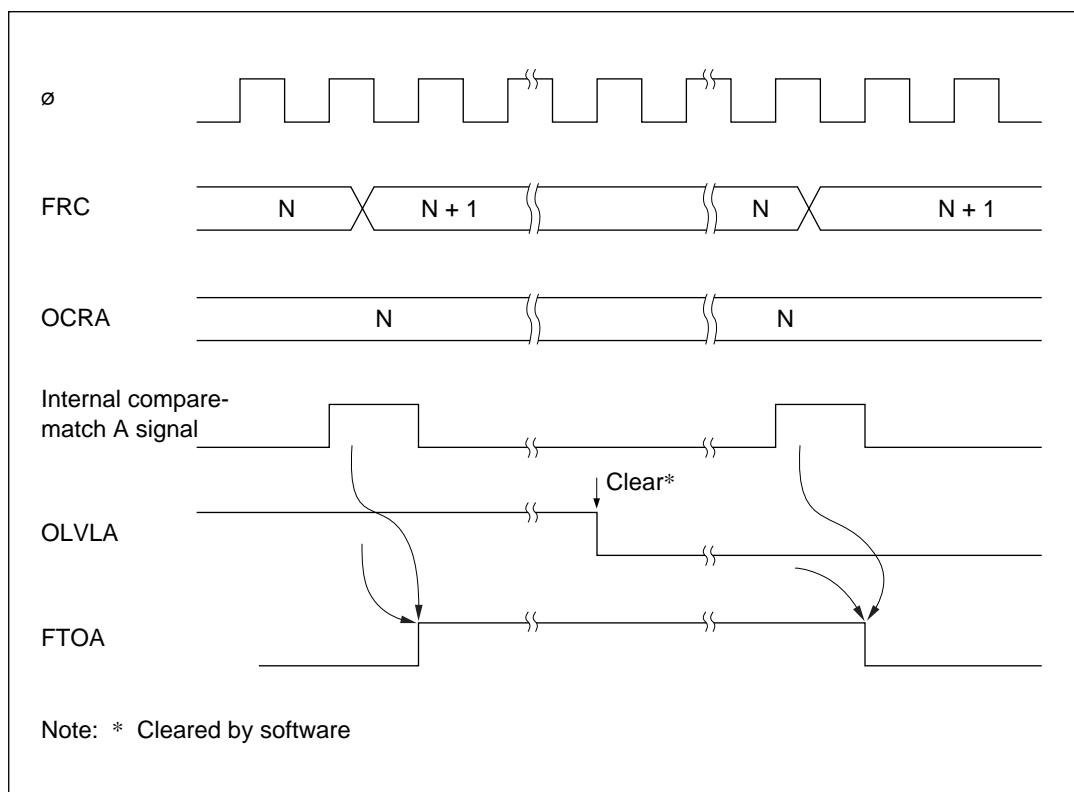
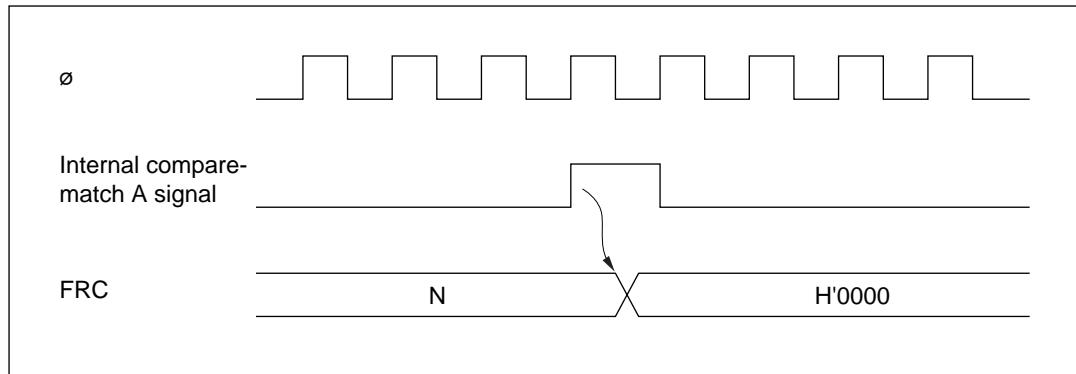


Figure 8-6 Timing of Output Compare A

### 8.4.3 FRC Clear Timing

If the CCLR<sub>A</sub> bit in TCSR is set to 1, the FRC is cleared when compare-match A occurs. Figure 8-7 shows the timing of this operation.



**Figure 8-7 Clearing of FRC by Compare-Match A**

#### 8.4.4 Input Capture Timing

**(1) Input Capture Timing:** An internal input capture signal is generated from the rising or falling edge of the signal at the input capture pin FTIx (x = A, B, C, D), as selected by the corresponding IEDGx bit in TCR. Figure 8-8 shows the usual input capture timing when the rising edge is selected (IEDGx = 1).

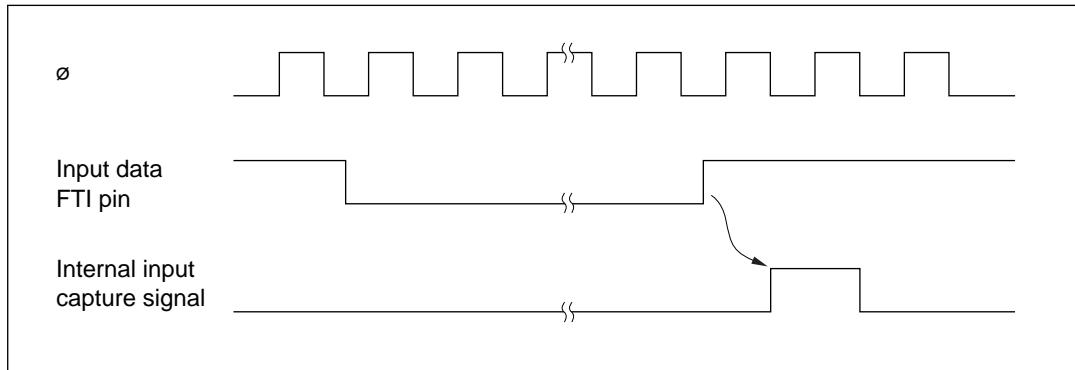


Figure 8-8 Input Capture Timing (Usual Case)

If the upper byte of ICRA/B/C/D is being read when the corresponding input capture signal arrives, the internal input capture signal is delayed by one state. Figure 8-9 shows the timing for this case.

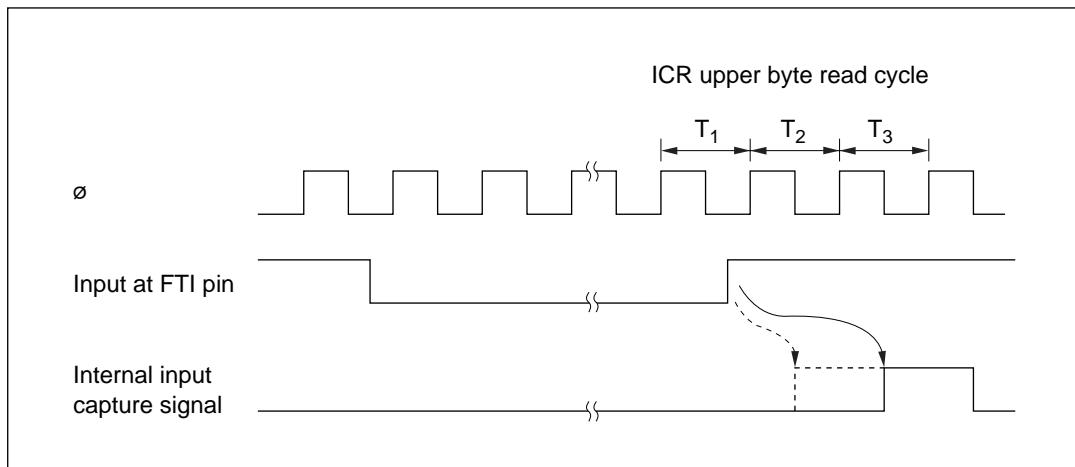


Figure 8-9 Input Capture Timing (1-State Delay Due to ICRA/B/C/D Read)

## (2) Buffered Input Capture Timing: ICRC and ICRA can operate as buffers for ICRA and ICRB.

Figure 8-10 shows how input capture operates when ICRA and ICRC are used in buffer mode and IEDGA and IEDGC are set to different values (IEDGA = 0 and IEDGC = 1, or IEDG A = 1 and IEDGC = 0), so that input capture is performed on both the rising and falling edges of FTIA.

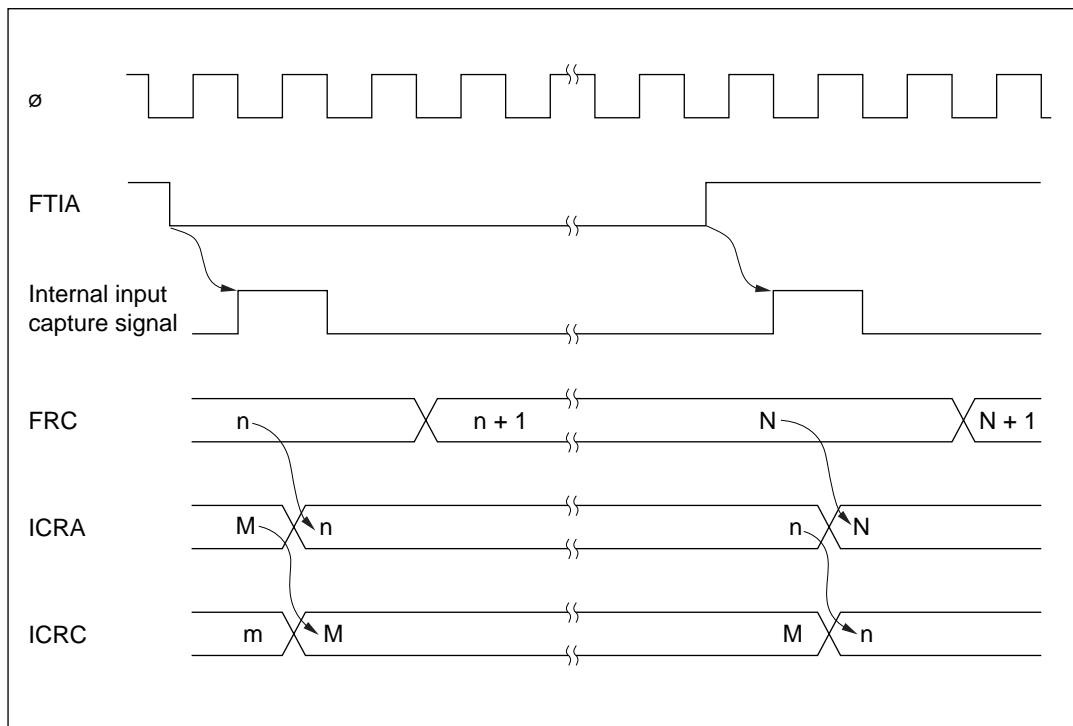
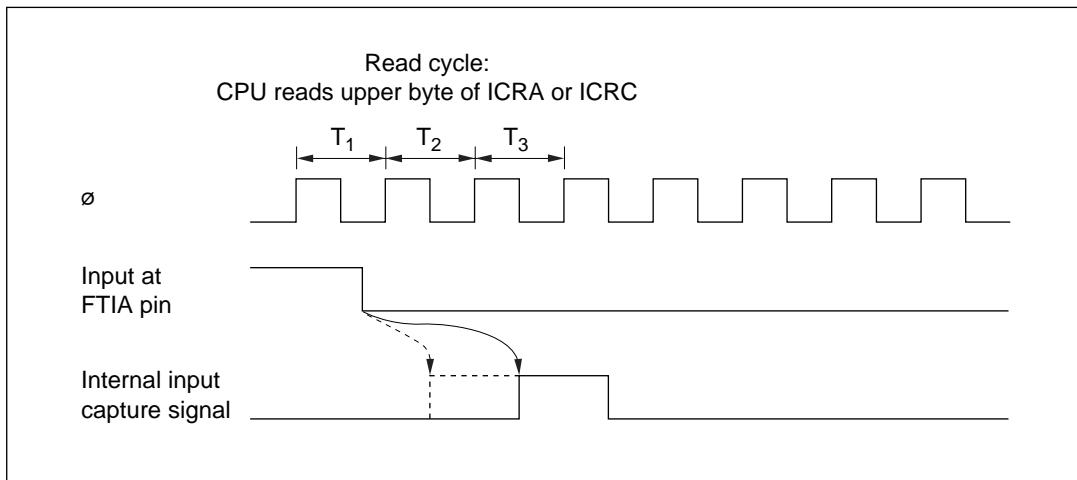


Figure 8-10 Buffered Input Capture with Both Edges Selected

When ICRC or ICRD is used as a buffer register, its input capture flag is set by the selected transition of its input capture signal. For example, if ICRC is used to buffer ICRA, when the edge transition selected by the IEDGC bit occurs on the FTIC input capture line, ICFC will be set, and if the ICIEC bit is set, an interrupt will be requested. The FRC value will not be transferred to ICRC, however.

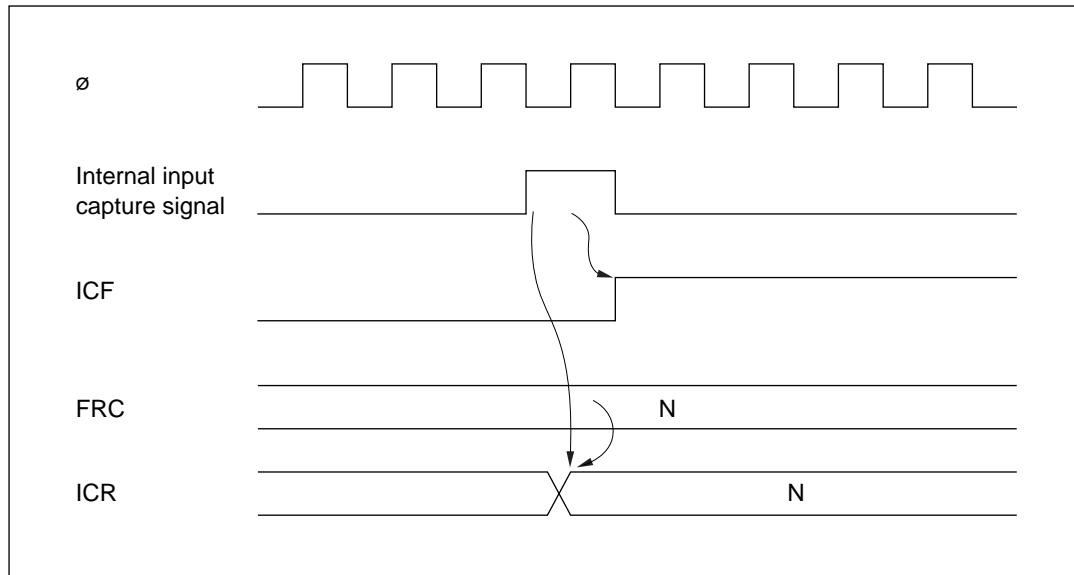
In buffered input capture, if the upper byte of either of the two registers to which data will be transferred (ICRA and ICRC, or ICRB and ICRD) is being read when the input signal arrives, input capture is delayed by one system clock ( $\phi$ ). Figure 8-11 shows the timing when BUFEA = 1.



**Figure 8-11 Input Capture Timing (1-State Delay, Buffer Mode)**

#### 8.4.5 Timing of Input Capture Flag (ICF) Setting

The input capture flag ICF<sub>x</sub> (x = A, B, C, D) is set to 1 by the internal input capture signal. Figure 8-12 shows the timing of this operation.



**Figure 8-12 Setting of Input Capture Flag**

#### 8.4.6 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 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 8-13 shows the timing of the setting of the output compare flags.

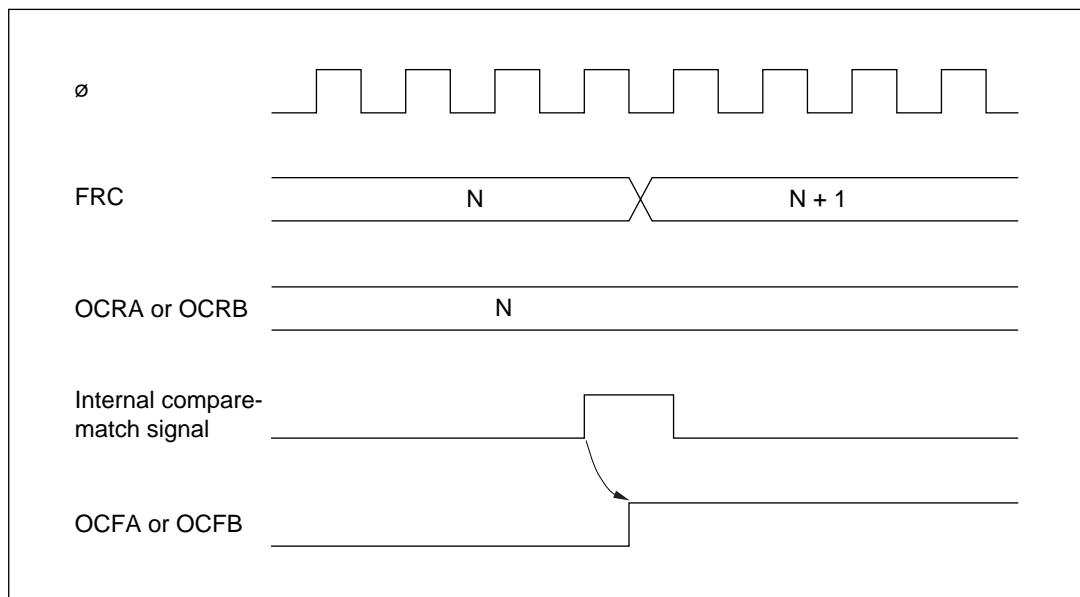


Figure 8-13 Setting of Output Compare Flags

#### 8.4.7 Setting of FRC Overflow Flag (OVF)

The FRC overflow flag (OVF) is set to 1 when FRC overflows (changes from H'FFFF to H'0000). Figure 8-14 shows the timing of this operation.

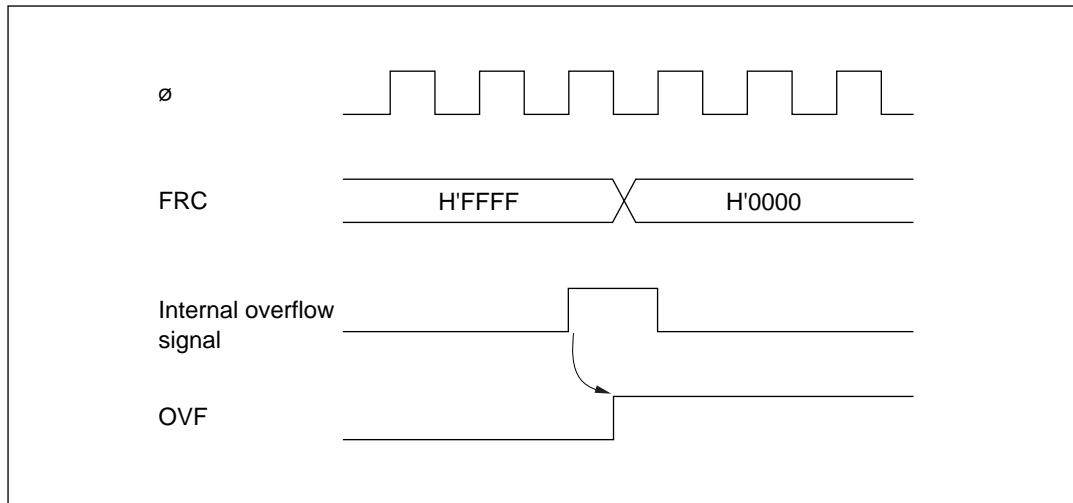


Figure 8-14 Setting of Overflow Flag (OVF)

## 8.5 Interrupts

The free-running timer can request seven interrupts (three types): input capture A to D (ICIA, ICIB, ICIC, ICID), output compare A and B (OCIA and OCIB), and overflow (FOVI). Each interrupt can be enabled or disabled by an enable bit in TIER. Independent signals are sent to the interrupt controller for each interrupt. Table 8-4 lists information about these interrupts.

Table 8-4 Free-Running Timer Interrupts

Interrupt	Description	Priority
ICIA	Requested by ICFA	High
ICIB	Requested by ICFB	
ICIC	Requested by ICFC	
ICID	Requested by ICFD	
OCIA	Requested by OCFA	
OCIB	Requested by OCFB	
FOVI	Requested by OVF	Low

## 8.6 Sample Application

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

- (1) The CCLRA bit in TCSR is set to 1.
- (2) Each time a compare-match interrupt occurs, software inverts the corresponding output level bit in TOCR (OLVLA or OLVLB).

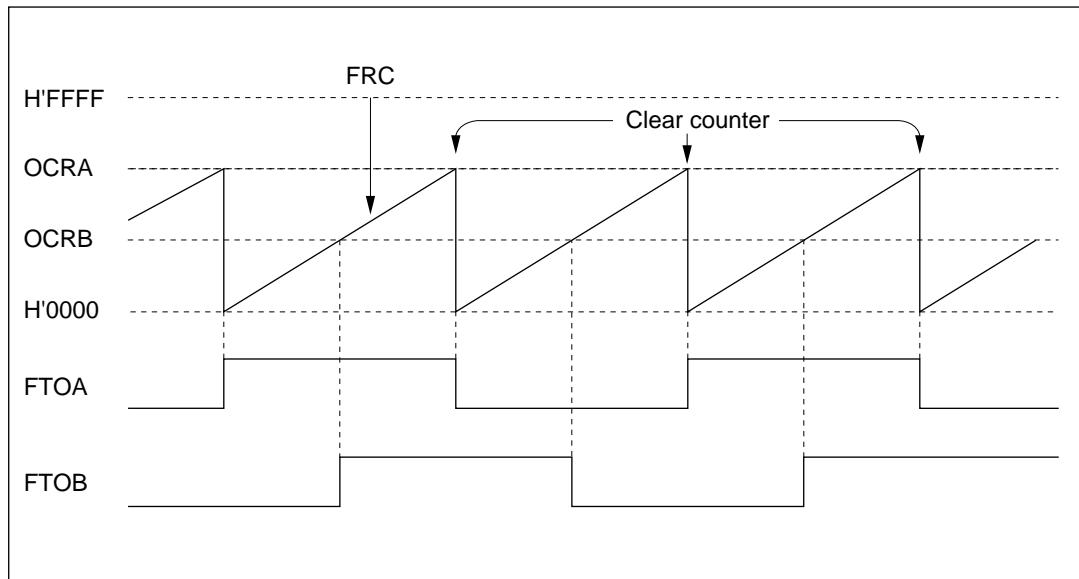


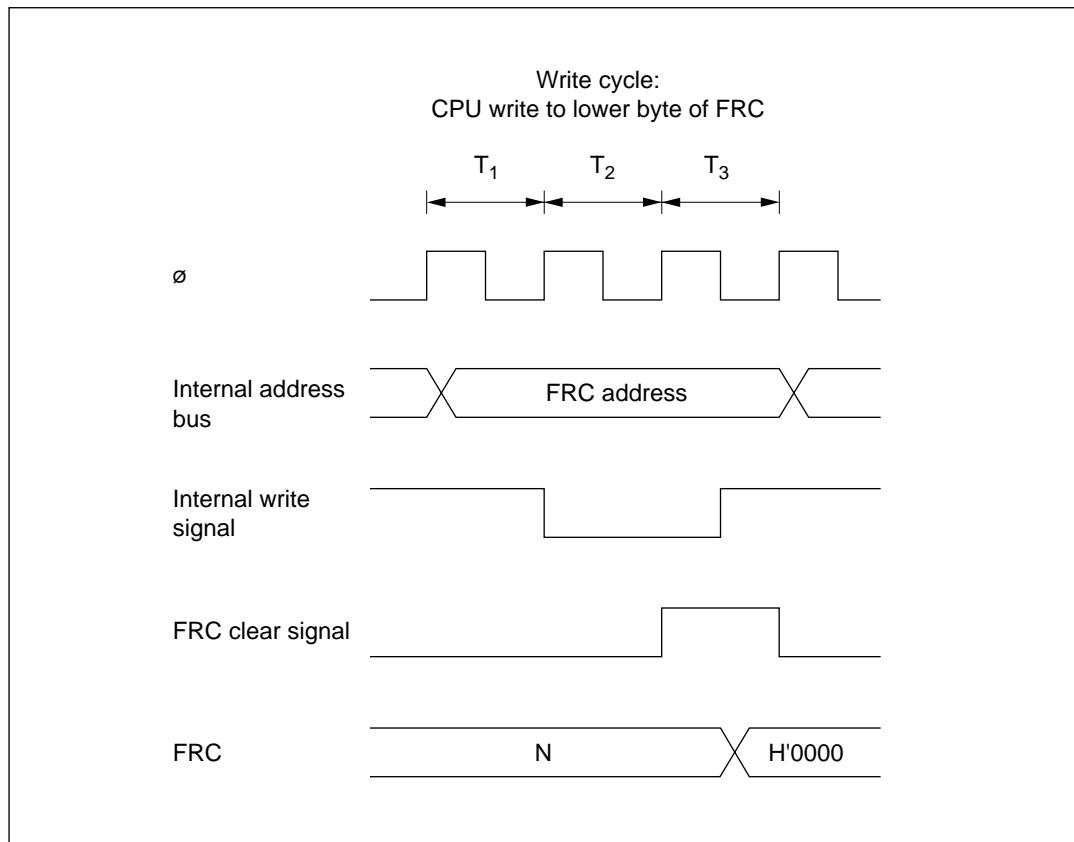
Figure 8-15 Square-Wave Output (Example)

## 8.7 Application Notes

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

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

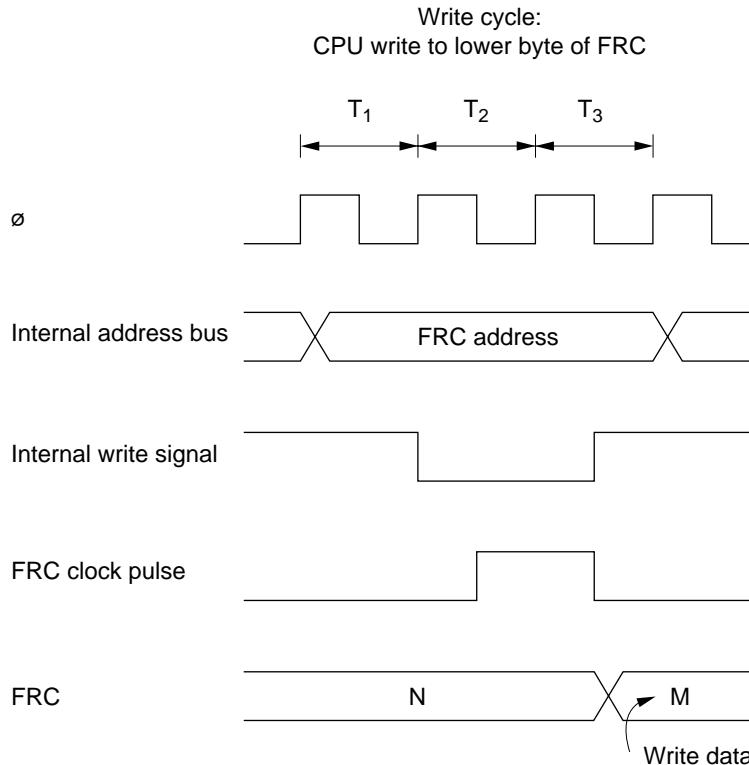
Figure 8-16 shows this type of contention.



**Figure 8-16 FRC Write-Clear Contention**

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

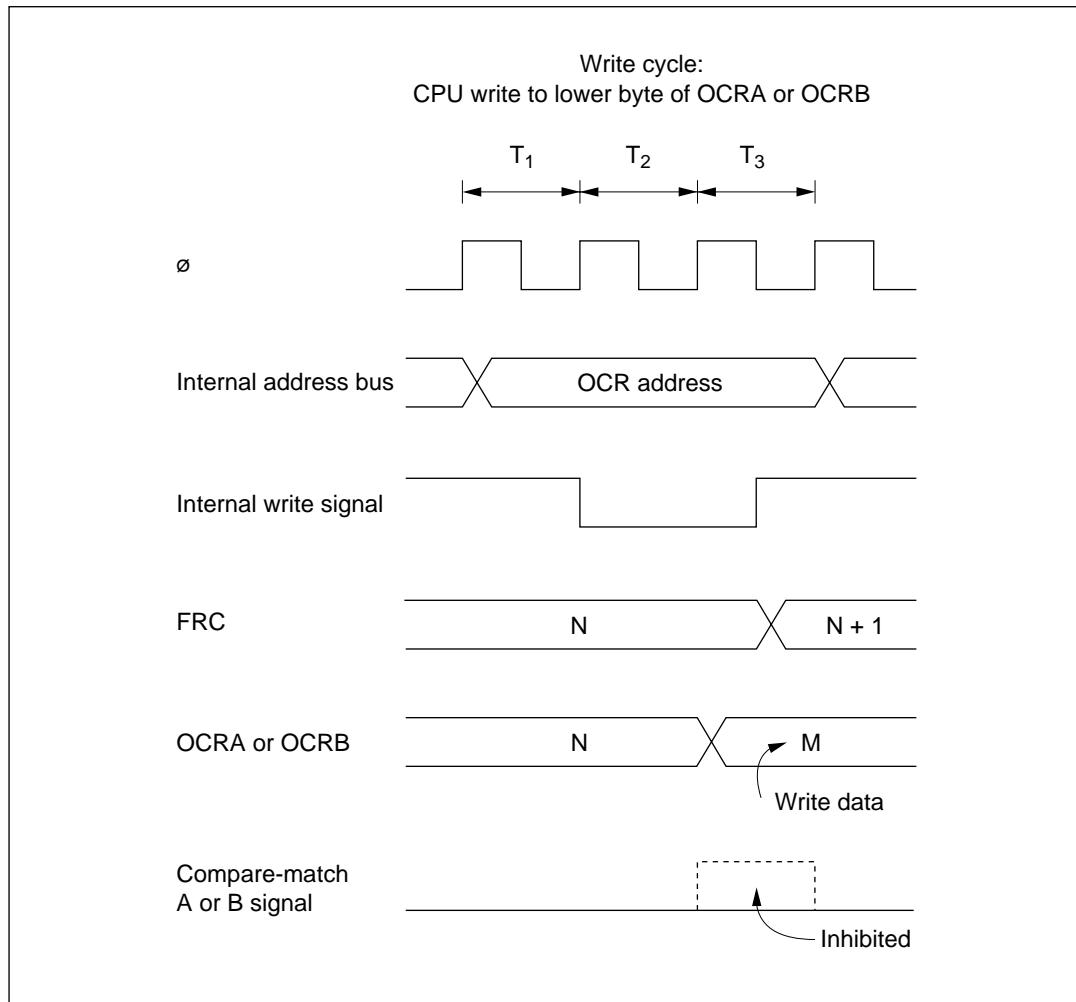
Figure 8-17 shows this type of contention.



**Figure 8-17 FRC Write-Increment Contention**

**(3) Contention between OCR Write and Compare-Match:** If a compare-match occurs during the  $T_3$  state of a write cycle to the lower byte of OCRA or OCRB, the write takes priority and the compare-match signal is inhibited.

Figure 8-18 shows this type of contention.



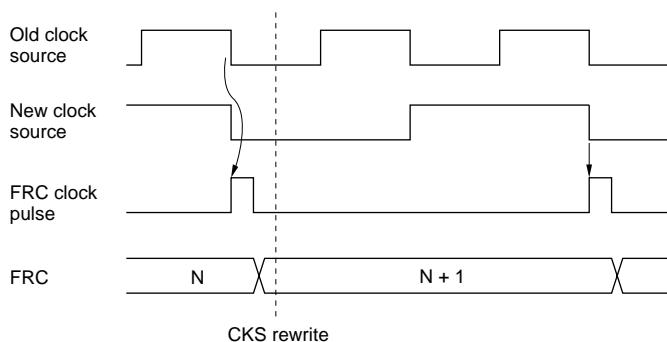
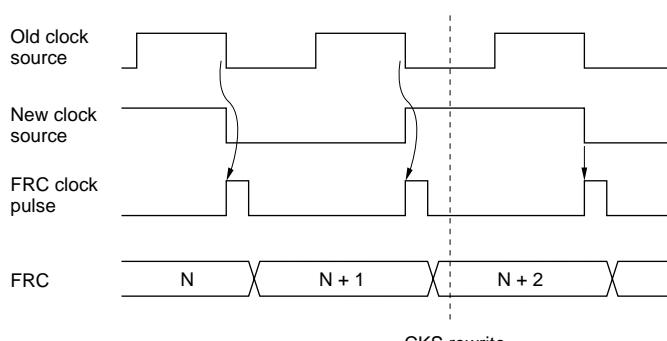
**Figure 8-18 Contention between OCR Write and Compare-Match**

**(4) Increment Caused by Changing of Internal Clock Source:** When an internal clock source is changed, the changeover may cause FRC to increment. This depends on the time at which the clock select bits (CKS1 and CKS0) are rewritten, as shown in table 8-5.

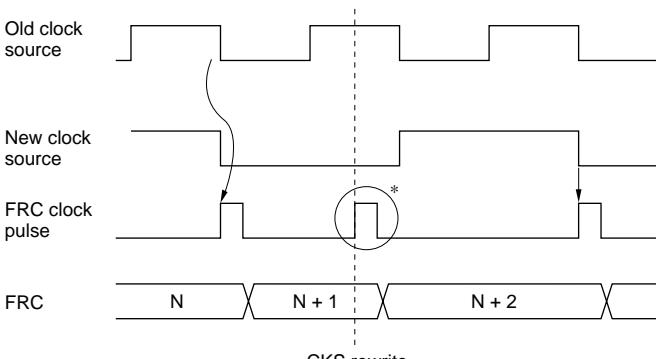
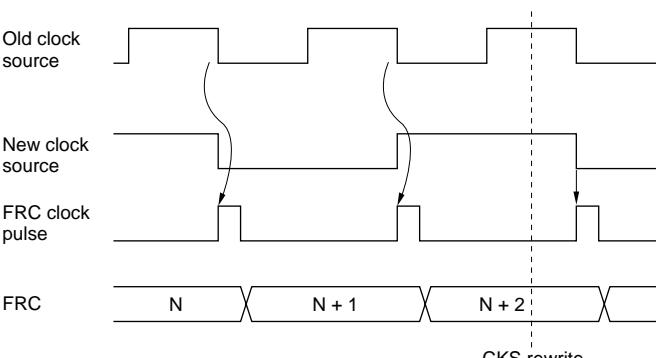
The pulse that increments FRC is generated at the falling edge of the internal clock source. If clock sources are changed when the old source is high and the new source is low, as in case no. 3 in table 8-5, the changeover generates a falling edge that triggers the FRC increment clock pulse.

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

**Table 8-5 Effect of Changing Internal Clock Sources**

No.	Description	Timing
1	Low → low: CKS1 and CKS0 are rewritten while both clock sources are low.	 <p>Old clock source</p> <p>New clock source</p> <p>FRC clock pulse</p> <p>FRC</p> <p>CKS rewrite</p>
2	Low → high: CKS1 and CKS0 are rewritten while old clock source is low and new clock source is high.	 <p>Old clock source</p> <p>New clock source</p> <p>FRC clock pulse</p> <p>FRC</p> <p>CKS rewrite</p>

**Table 8-5 Effect of Changing Internal Clock Sources (cont)**

No.	Description	Timing
3	High → low: CKS1 and CKS0 are rewritten while old clock source is high and new clock source is low.	 <p>Old clock source</p> <p>New clock source</p> <p>FRC clock pulse</p> <p>FRC</p> <p>CKS rewrite</p>
4	High → high: CKS1 and CKS0 are rewritten while both clock sources are high.	 <p>Old clock source</p> <p>New clock source</p> <p>FRC clock pulse</p> <p>FRC</p> <p>CKS rewrite</p>

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

# Section 9 8-Bit Timers

## 9.1 Overview

The H8/3437 Series includes an 8-bit timer module with two channels (numbered 0 and 1). Each channel has an 8-bit counter (TCNT) and two time constant registers (TCORA and TCORB) that are constantly compared with the TCNT value to detect compare-match events. One of the many applications of the 8-bit timer module is to generate a rectangular-wave output with an arbitrary duty cycle.

### 9.1.1 Features

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

- Selection of seven clock sources

The counters can be driven by one of six internal clock signals or an external clock input (enabling use as an external event counter).

- Selection of three ways to clear the counters

The counters can be cleared on compare-match A or B, or by an external reset signal.

- Timer output controlled by two time constants

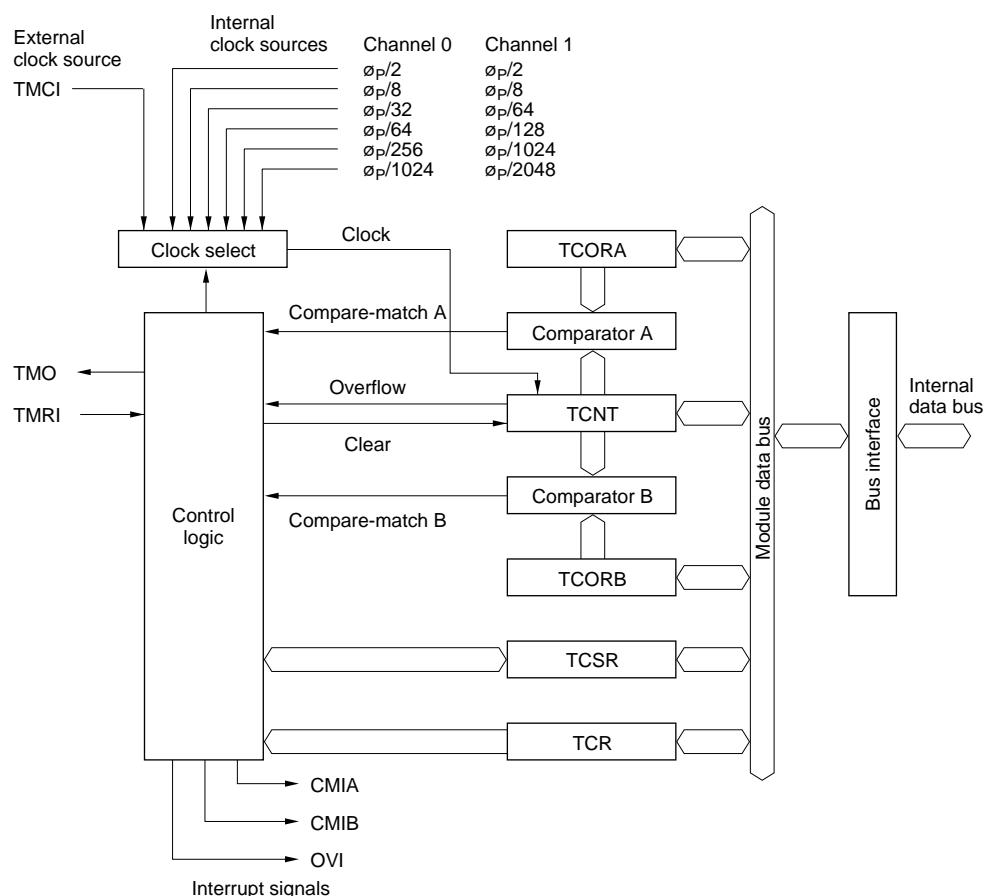
The timer output signal in each channel is controlled by two independent time constants, enabling the timer to generate output waveforms with an arbitrary duty cycle, or PWM waveforms.

- Three independent interrupts

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

## 9.1.2 Block Diagram

Figure 9-1 shows a block diagram of one channel in the 8-bit timer module.



TCR: Timer control register (8 bits)  
TCCSR: Timer control status register (8 bits)  
TCORA: Time constant register A (8 bits)  
TCORB: Time constant register B (8 bits)  
TCNT: Timer counter

Figure 9-1 Block Diagram of 8-Bit Timer (1 Channel)

### 9.1.3 Input and Output Pins

Table 9-1 lists the input and output pins of the 8-bit timer.

**Table 9-1 Input and Output Pins of 8-Bit Timer**

Name	Abbreviation*		I/O	Function
	Channel 0	Channel 1		
Timer output	TMO <sub>0</sub>	TMO <sub>1</sub>	Output	Output controlled by compare-match
Timer clock input	TMCI <sub>0</sub>	TMCI <sub>1</sub>	Input	External clock source for the counter
Timer reset input	TMRI <sub>0</sub>	TMRI <sub>1</sub>	Input	External reset signal for the counter

Note: \* In this manual, the channel subscript has been deleted, and only TMO, TMCI, and TMRI are used.

### 9.1.4 Register Configuration

Table 9-2 lists the registers of the 8-bit timer module. Each channel has an independent set of registers.

**Table 9-2 8-Bit Timer Registers**

Channel	Name	Abbreviation	R/W	Initial Value	Address
0	Timer control register	TCR	R/W	H'00	H'FFC8
	Timer control/status register	TCSR	R/(W)*	H'10	H'FFC9
	Time constant register A	TCORA	R/W	H'FF	H'FFCA
	Time constant register B	TCORB	R/W	H'FF	H'FFCB
	Timer counter	TCNT	R/W	H'00	H'FFCC
1	Timer control register	TCR	R/W	H'00	H'FFD0
	Timer control/status register	TCSR	R/(W)*	H'10	H'FFD1
	Time constant register A	TCORA	R/W	H'FF	H'FFD2
	Time constant register B	TCORB	R/W	H'FF	H'FFD3
	Timer counter	TCNT	R/W	H'00	H'FFD4
0, 1	Serial/timer control register	STCR	R/W	H'00	H'FFC3

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

## 9.2 Register Descriptions

### 9.2.1 Timer Counter (TCNT)

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

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

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

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

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

### 9.2.2 Time Constant Registers A and B (TCORA and TCORB)

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

TCORA and TCORB are 8-bit readable/writable registers. The timer count is continually compared with the constants written in these registers (except during the T<sub>3</sub> state of a write cycle to TCORA or TCORB). 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 is controlled by these compare-match signals as specified by output select bits 3 to 0 (OS3 to OS0) in the timer control/status register (TCSR).

TCORA and TCORB are initialized to H'FF by a reset and in the standby modes.

### 9.2.3 Timer Control Register (TCR)

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

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.

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

For timing diagrams, see section 9.3, Operation.

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

Bit 7 CMIEB	Description	
0	Compare-match interrupt request B (CMIB) is disabled.	(Initial value)
1	Compare-match interrupt request B (CMIB) is enabled.	

**Bit 6—Compare-match Interrupt Enable A (CMIEA):** This bit selects whether to request compare-match interrupt A (CMIA) when compare-match flag A (CMFA) in 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 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 (TMRI).

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 and bits ICKS1 and ICKS0 in the serial/timer control register (STCR) select the internal or external clock source for the timer counter. Six internal clock sources, derived by prescaling the system clock, are available for each timer channel. For internal clock sources the counter is incremented on the falling edge of the internal clock. For an external clock source, these bits can select whether to increment the counter on the rising or falling edge of the clock input (TMCI), or on both edges.

Channel	TCR			STCR		Description
	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0	
0	0	0	0	—	—	No clock source (timer stopped) (Initial value)
	0	0	1	—	0	$\phi_P/8$ internal clock, counted on falling edge
	0	0	1	—	1	$\phi_P/2$ internal clock, counted on falling edge
	0	1	0	—	0	$\phi_P/64$ internal clock, counted on falling edge
	0	1	0	—	1	$\phi_P/32$ internal clock, counted on falling edge
	0	1	1	—	0	$\phi_P/1024$ internal clock, counted on falling edge
	0	1	1	—	1	$\phi_P/256$ internal clock, counted on falling edge
	1	0	0	—	—	No clock source (timer stopped)
	1	0	1	—	—	External clock source, counted on rising edge
	1	1	0	—	—	External clock source, counted on falling edge
1	1	1	1	—	—	External clock source, counted on both rising and falling edges
	0	0	0	—	—	No clock source (timer stopped) (Initial value)
	0	0	1	0	—	$\phi_P/8$ internal clock, counted on falling edge
	0	0	1	1	—	$\phi_P/2$ internal clock, counted on falling edge
	0	1	0	0	—	$\phi_P/64$ internal clock, counted on falling edge
	0	1	0	1	—	$\phi_P/128$ internal clock, counted on falling edge
	0	1	1	0	—	$\phi_P/1024$ internal clock, counted on falling edge
	0	1	1	1	—	$\phi_P/2048$ internal clock, counted on falling edge
	1	0	0	—	—	No clock source (timer stopped)
	1	0	1	—	—	External clock source, counted on rising edge
2	1	1	0	—	—	External clock source, counted on falling edge
	1	1	1	—	—	External clock source, counted on both rising and falling edges

## 9.2.4 Timer Control/Status Register (TCSR)

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

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

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.

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

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

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

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

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

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

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

**Bit 4—Reserved:** This bit is always read as 1. It cannot be written.

**Bits 3 to 0—Output Select 3 to 0 (OS3 to OS0):** These bits specify the effect of TCOR–TCNT 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.

If compare-match A and B occur simultaneously, any conflict is resolved according to the following priority order: toggle > 1 output > 0 output.

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

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

Bit 3	Bit 2		
OS3	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	Bit 0		
OS1	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.	

## 9.2.5 Serial/Timer Control Register (STCR)

Bit	7	6	5	4	3	2	1	0
	IICS	IICD	IICX	IICE	STAC	MPE	ICKS1	ICKS0
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

STCR is an 8-bit readable/writable register that controls the I<sup>2</sup>C bus interface and host interface, controls the operating mode of the serial communication interface, and selects internal clock sources for the timer counters.

STCR is initialized to H'00 by a reset.

**Bits 7 to 4—I<sup>2</sup>C Control (IICS, IICD, IICX, IICE):** These bits control operation of the I<sup>2</sup>C bus interface. For details, see section 13, I<sup>2</sup>C Bus Interface.

**Bit 3—Slave Input Switch (STAC):** Controls the switching of the host interface input pins. For details, see section 14, Host Interface.

**Bit 2—Multiprocessor Enable (MPE):** Controls the operating mode of serial communication interfaces 0 and 1. For details, see section 12, Serial Communication Interface.

**Bits 1 and 0—Internal Clock Source Select 1 and 0 (ICKS1 and ICKS0):** These bits and bits CKS2 to CKS0 in the TCR select clock sources for the timer counters. For details, see section 9.2.3, Timer Control Register.

## 9.3 Operation

### 9.3.1 TCNT Increment Timing

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

**Internal Clock:** Internal clock sources are created from the system clock by a prescaler. The counter increments on an internal TCNT clock pulse generated from the falling edge of the prescaler output, as shown in figure 9-2. Bits CKS2 to CKS0 of TCR and bits ICKS1 and ICKS0 of STCR can select one of the six internal clocks.

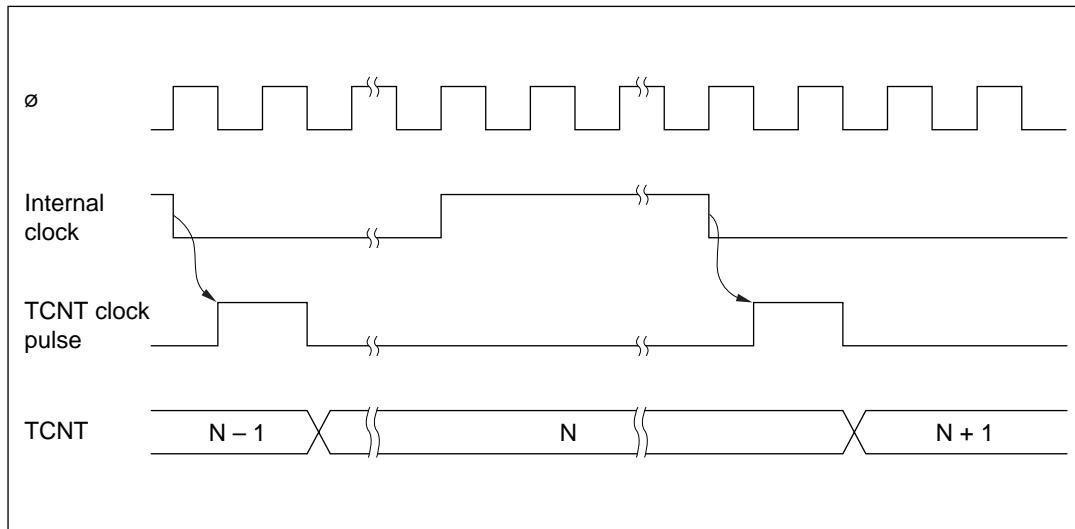
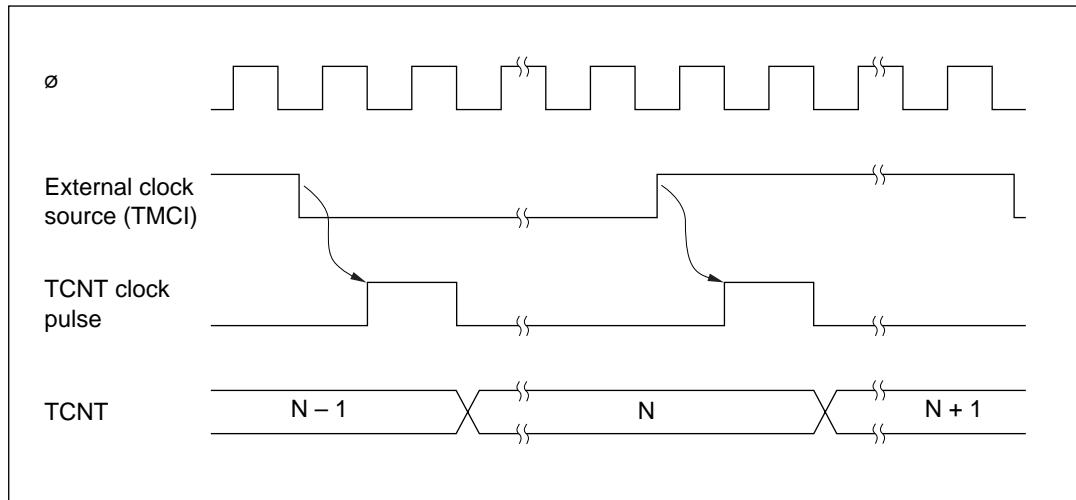


Figure 9-2 Count Timing for Internal Clock Input

**External Clock:** If external clock input (TMCI) is selected, the timer counter can increment on the rising edge, the falling edge, or both edges of the external clock signal. Figure 9-3 shows incrementation on both edges of the external clock signal.

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



**Figure 9-3 Count Timing for External Clock Input**

### 9.3.2 Compare-Match Timing

**1. Setting of Compare-Match Flags A and B (CMFA and CMFB):** The compare-match flags are set to 1 by an internal compare-match signal generated when the timer count matches the time constant in 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 9-4 shows the timing of the setting of the compare-match flags.

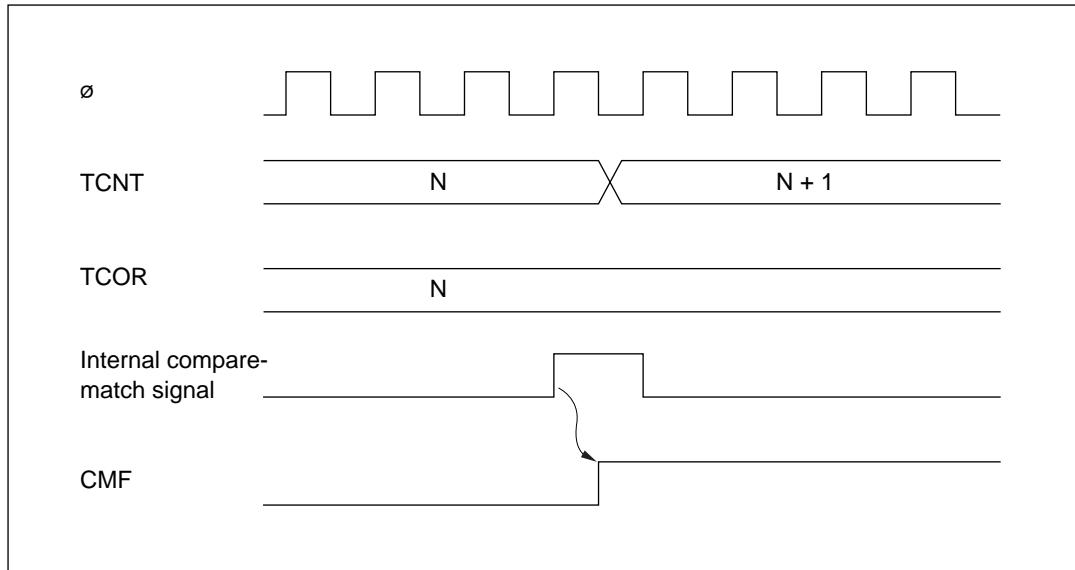
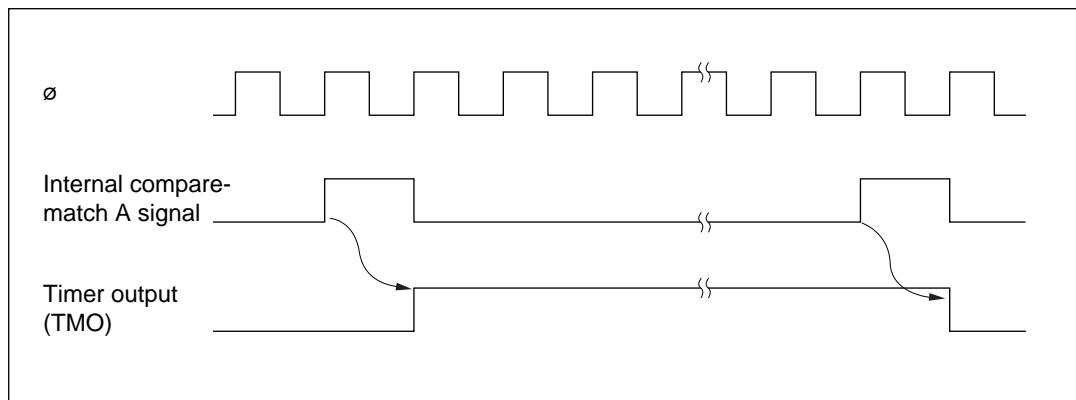


Figure 9-4 Setting of Compare-Match Flags

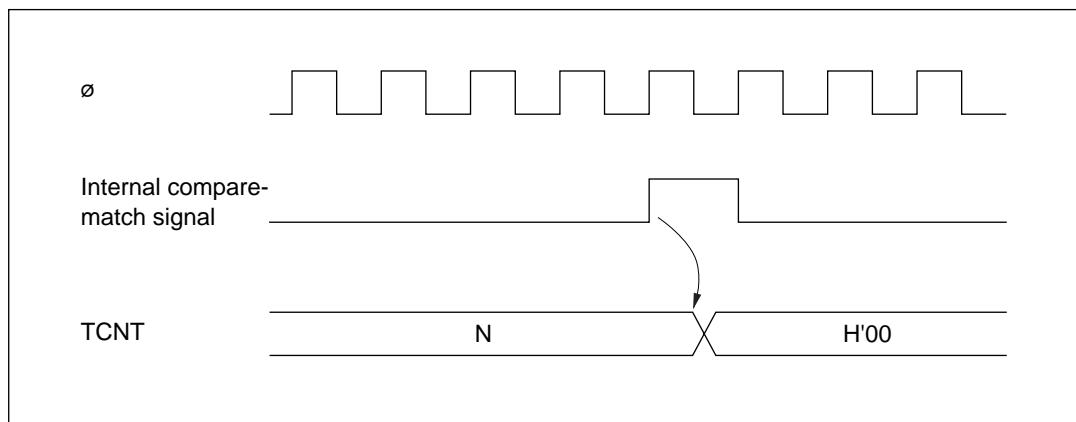
**2. Output Timing:** When a compare-match event occurs, the timer output changes as specified by the output select bits (OS3 to OS0) in the TCSR. Depending on these bits, the output can remain the same, change to 0, change to 1, or toggle.

Figure 9-5 shows the timing when the output is set to toggle on compare-match A.



**Figure 9-5 Timing of Timer Output**

**3. Timing of Compare-Match Clear:** Depending on the CCLR1 and CCLR0 bits in TCR, the timer counter can be cleared when compare-match A or B occurs. Figure 9-6 shows the timing of this operation.



**Figure 9-6 Timing of Compare-Match Clear**

### 9.3.3 External Reset of TCNT

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

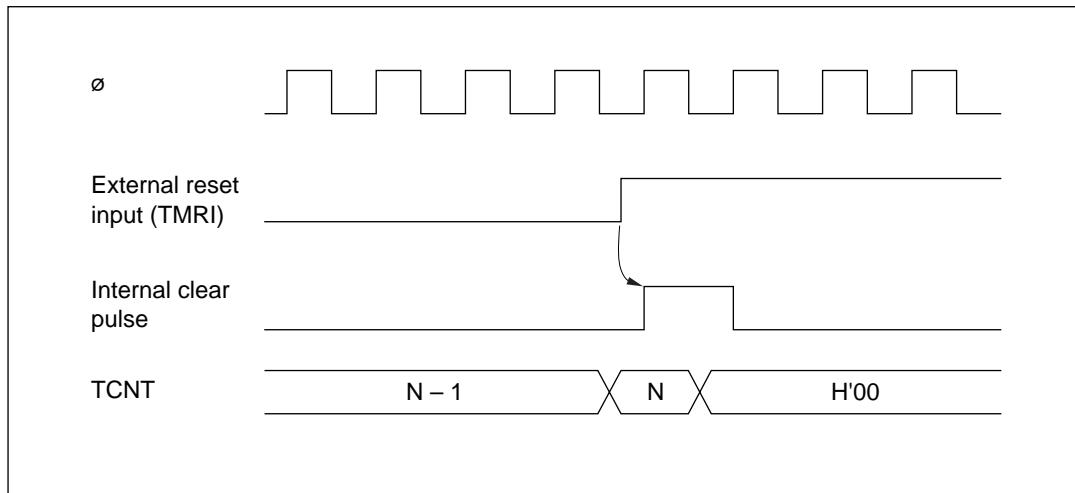


Figure 9-7 Timing of External Reset

### 9.3.4 Setting of TCSR Overflow Flag (OVF)

The overflow flag (OVF) is set to 1 when the timer count overflows (changes from H'FF to H'00). Figure 9-8 shows the timing of this operation.

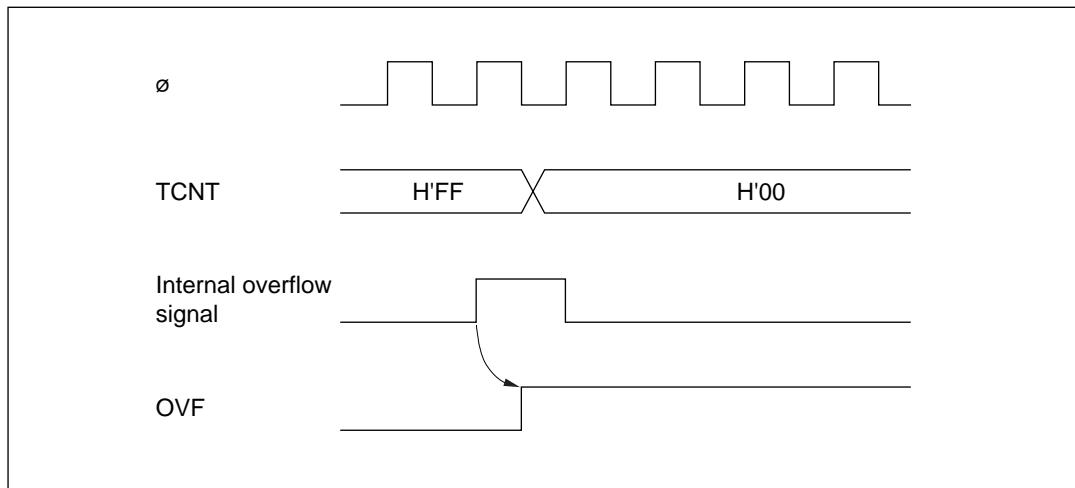


Figure 9-8 Setting of Overflow Flag (OVF)

## 9.4 Interrupts

Each channel in the 8-bit timer can generate three types of interrupts: compare-match A and B (CMIA and CMIB), and overflow (OVI). Each interrupt can be enabled or disabled by an enable bit in TCR. Independent signals are sent to the interrupt controller for each interrupt. Table 9-3 lists information about these interrupts.

**Table 9-3 8-Bit Timer Interrupts**

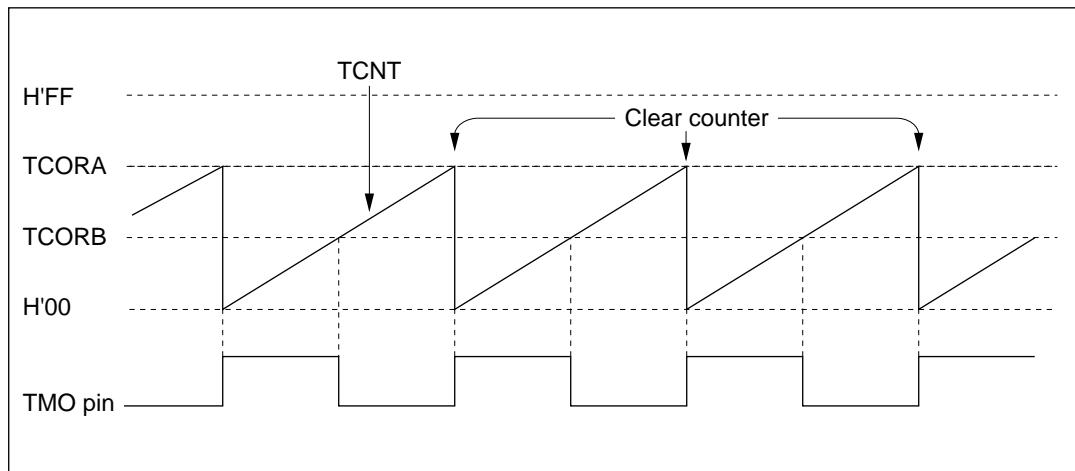
Interrupt	Description	Priority
CMIA	Requested by CMFA	High
CMIB	Requested by CMFB	
OVI	Requested by OVF	Low

## 9.5 Sample Application

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

- (1) In 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 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 9-9 Example of Pulse Output**

## 9.6 Application Notes

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

### 9.6.1 Contention between TCNT Write and Clear

If an internal counter clear signal is generated during the  $T_3$  state of a write cycle to the timer counter, the clear signal takes priority and the write is not performed.

Figure 9-10 shows this type of contention.

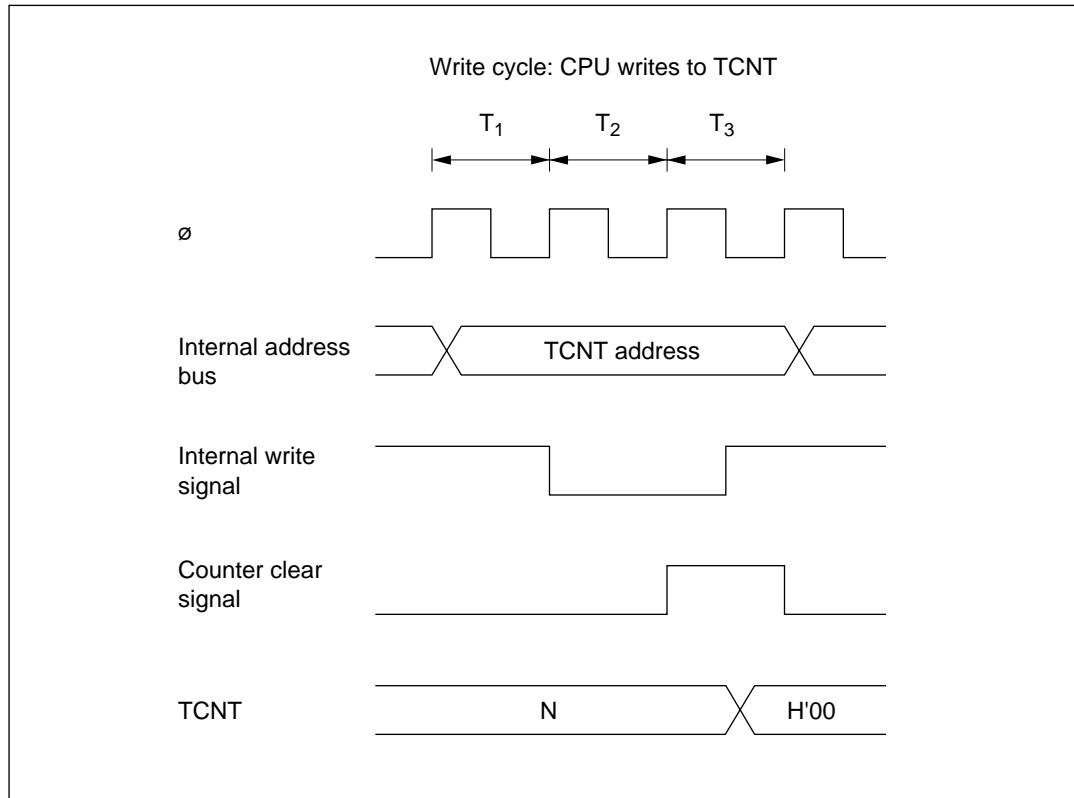


Figure 9-10 TCNT Write-Clear Contention

## 9.6.2 Contention between TCNT Write and Increment

If a timer counter increment pulse is generated during the  $T_3$  state of a write cycle to the timer counter, the write takes priority and the timer counter is not incremented.

Figure 9-11 shows this type of contention.

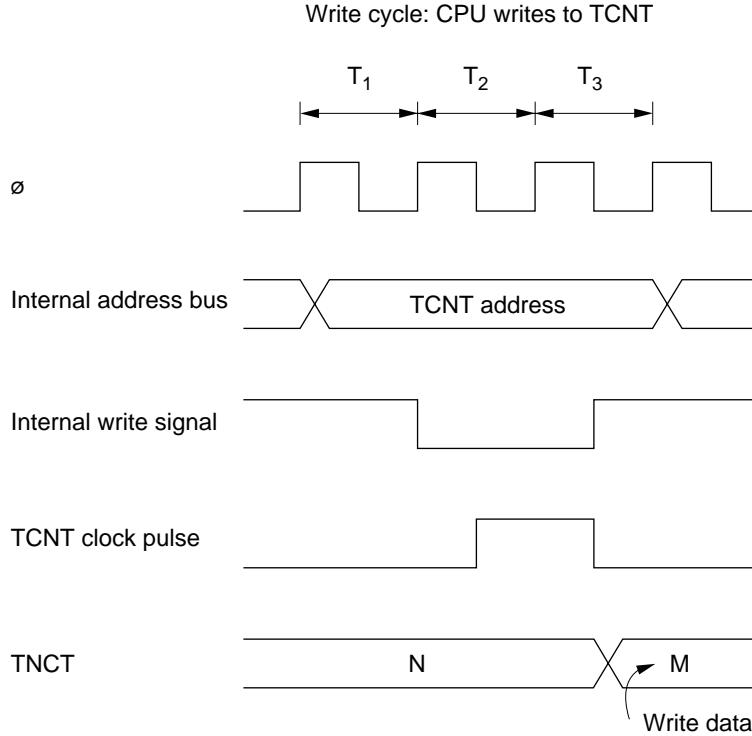


Figure 9-11 TCNT Write-Increment Contention

### 9.6.3 Contention between TCOR Write and Compare-Match

If a compare-match occurs during the  $T_3$  state of a write cycle to TCOR, the write takes priority and the compare-match signal is inhibited.

Figure 9-12 shows this type of contention.

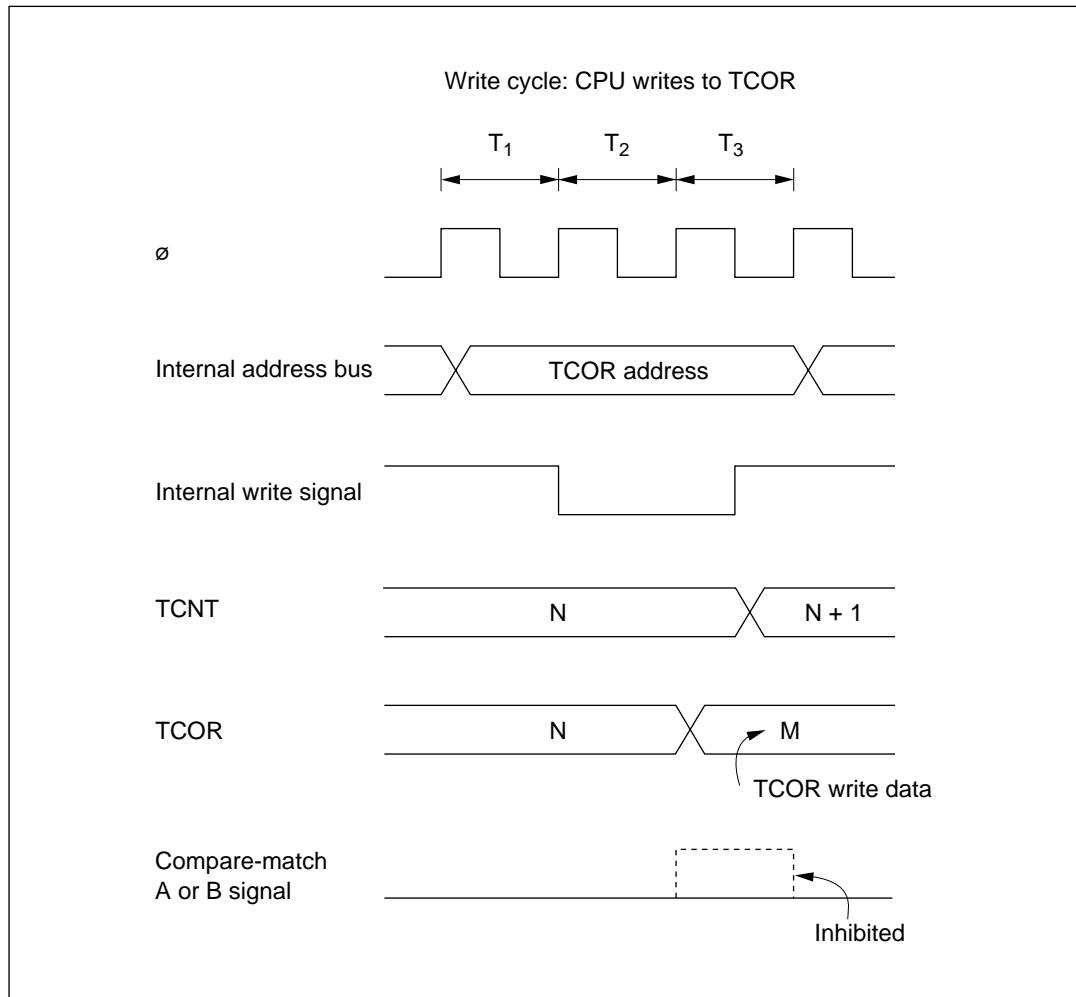


Figure 9-12 Contention between TCOR Write and Compare-Match

#### 9.6.4 Contention between Compare-Match A and Compare-Match B

If identical time constants are written in TCORA and TCORB, causing compare-match A and B to occur simultaneously, any conflict between the output selections for compare-match A and B is resolved by following the priority order in table 9-4.

**Table 9-4 Priority of Timer Output**

Output Selection	Priority
Toggle	High
1 output	
0 output	
No change	Low

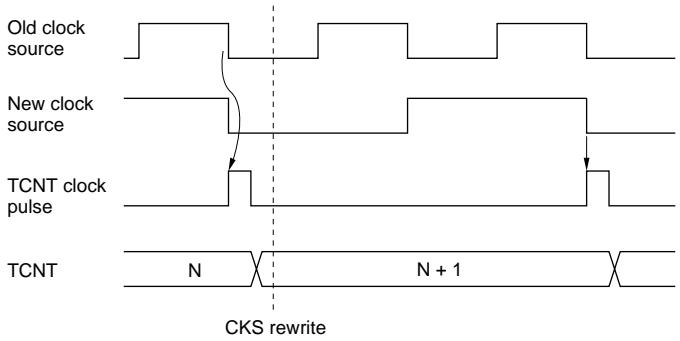
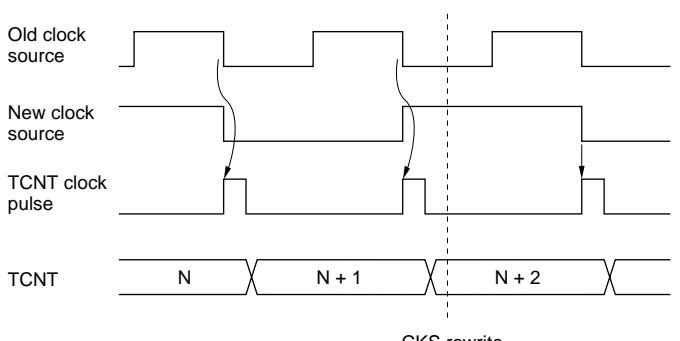
#### 9.6.5 Increment Caused by Changing of Internal Clock Source

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 (CKS1, CKS0) are rewritten, as shown in table 9-5.

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 9-5, the changeover generates a falling edge that triggers the TCNT clock pulse and increments the timer counter.

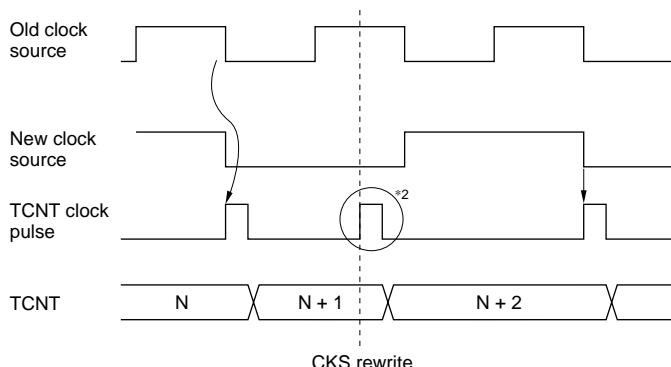
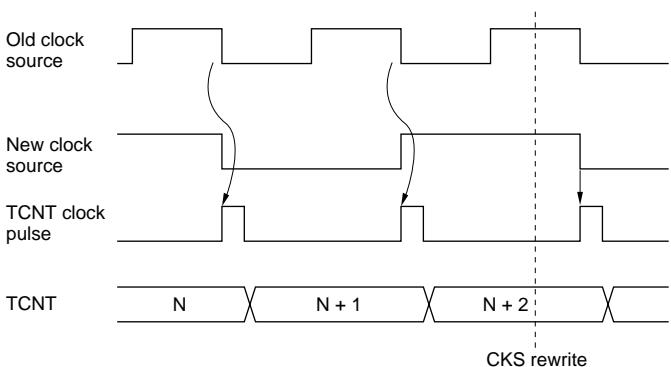
Switching between an internal and external clock source can also cause the timer counter to increment.

**Table 9-5 Effect of Changing Internal Clock Sources**

No.	Description	Timing
1	Low → low <sup>*1</sup>	
2	Low → high <sup>*2</sup>	

Notes: 1. Including a transition from low to the stopped state (CKS1 = 0, CKS0 = 0), or a transition from the stopped state to low.  
 2. Including a transition from the stopped state to high.

**Table 9-5 Effect of Changing Internal Clock Sources (cont)**

No.	Description	Timing chart
3	High → low <sup>*1</sup>	
4	High → high	

Notes: 1. Including a transition from high to the stopped state.

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

# Section 10 PWM Timers

## 10.1 Overview

The H8/3437 Series has an on-chip pulse-width modulation (PWM) timer module with two independent channels (PWM0 and PWM1). Both channels are functionally identical. Each PWM channel generates a rectangular output pulse with a duty cycle of 0 to 100%. The duty cycle is specified in an 8-bit duty register (DTR).

### 10.1.1 Features

The PWM timer module has the following features:

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

### 10.1.2 Block Diagram

Figure 10-1 shows a block diagram of one PWM timer channel.

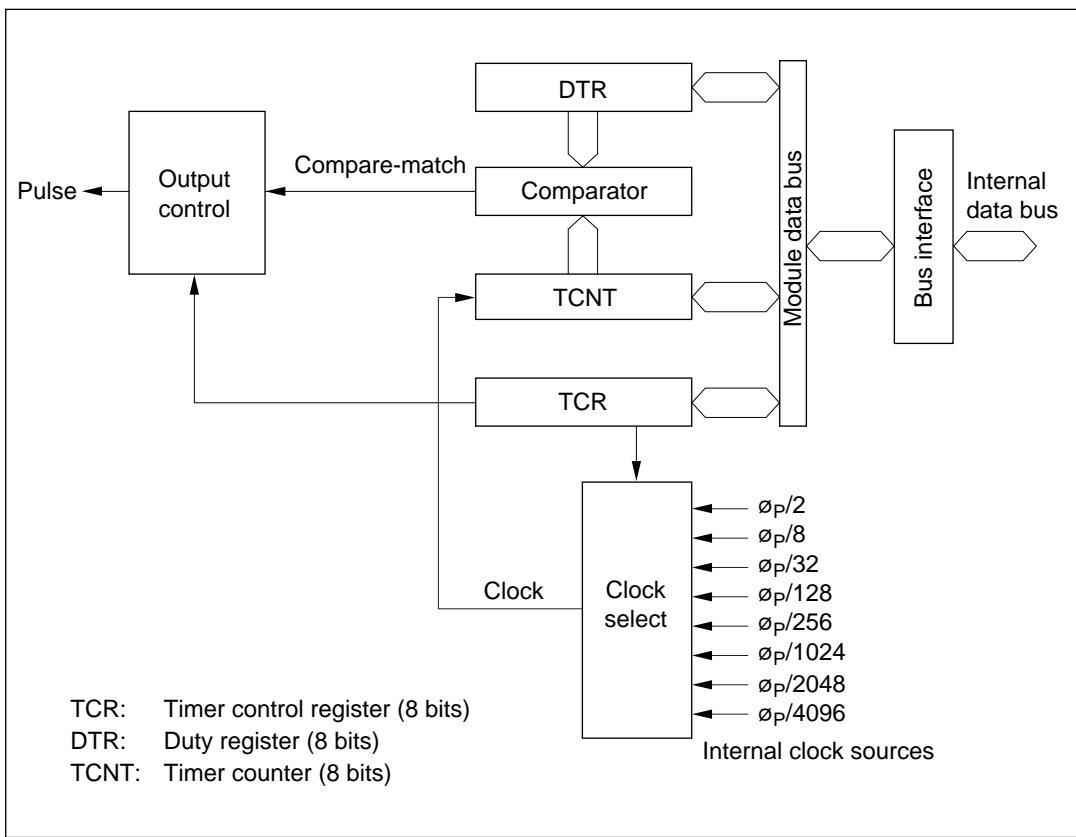


Figure 10-1 Block Diagram of PWM Timer

### 10.1.3 Input and Output Pins

Table 10-1 lists the output pins of the PWM timer module. There are no input pins.

**Table 10-1 Output Pins of PWM Timer Module**

Name	Abbreviation	I/O	Function
PWM0 output	PW <sub>0</sub>	Output	Pulse output from PWM timer channel 0.
PWM1 output	PW <sub>1</sub>	Output	Pulse output from PWM timer channel 1.

### 10.1.4 Register Configuration

The PWM timer module has three registers for each channel as listed in table 10-2.

**Table 10-2 PWM Timer Registers**

Name	Abbreviation	R/W	Initial Value	Address	
				PWM0	PWM1
Timer control register	TCR	R/W	H'38	H'FFA0	H'FFA4
Duty register	DTR	R/W	H'FF	H'FFA1	H'FFA5
Timer counter	TCNT	R/W	H'00	H'FFA2	H'FFA6

## 10.2 Register Descriptions

### 10.2.1 Timer Counter (TCNT)

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

TCNT is an 8-bit readable/writable up-counter. When the output enable bit (OE) is set to 1 in TCR, TCNT 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 count repeats from H'00. When TCNT changes from H'00 to H'01, the PWM output is placed in the 1 state, unless the DTR value is H'00, in which case the duty cycle is 0% and the PWM output remains in the 0 state.

TCNT is initialized to H'00 at a reset and in the standby modes, and when the OE bit is cleared to 0.

### 10.2.2 Duty Register (DTR)

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

DTR is an 8-bit readable/writable register that specifies the duty cycle of the output pulse. Any duty cycle from 0% to 100% can be output by setting the corresponding value in DTR. The resolution is 1/250. Writing 0 (H'00) in DTR gives a 0% duty cycle. Writing 125 (H'7D) gives a 50% duty cycle. Writing 250 (H'FA) gives a 100% duty cycle.

The DTR and TCNT values are always compared. When the values match, the PWM output is placed in the 0 state.

DTR is double-buffered. A new value written in DTR does not become valid until after the timer count changes from H'F9 to H'00. While the OE bit is cleared to 0 in TCR, however, new values written in DTR become valid immediately. When DTR is read, the value read is the currently valid value.

DTR is initialized to H'FF by a reset and in the standby modes.

### 10.2.3 Timer Control Register (TCR)

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

TCR is an 8-bit readable/writable register that selects the clock input to TCNT and controls PWM output.

TCR is initialized to H'38 by a reset and in standby mode.

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

#### Bit 7

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

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

#### Bit 6

OS	Description	
0	Positive logic; positive-going PWM pulse, 1 = high	(Initial value)
1	Negative logic; negative-going PWM pulse, 1 = low	

**Bits 5 to 3—Reserved:** These bits cannot be modified and are always read as 1.

**Bits 2, 1, and 0—Clock Select (CKS2, CKS1, and CKS0):** These bits select one of eight internal clock sources obtained by dividing the supporting-module clock ( $\phi_P$ ).

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	0	$\phi_P/2$	(Initial value)
0	0	1	$\phi_P/8$	
0	1	0	$\phi_P/32$	
0	1	1	$\phi_P/128$	
1	0	0	$\phi_P/256$	
1	0	1	$\phi_P/1024$	
1	1	0	$\phi_P/2048$	
1	1	1	$\phi_P/4096$	

From the clock source frequency, the resolution, period, and frequency of the PWM output can be calculated as follows.

$$\text{Resolution} = 1/\text{clock source frequency}$$

$$\text{PWM period} = \text{resolution} \times 250$$

$$\text{PWM frequency} = 1/\text{PWM period}$$

If the  $\phi_P$  clock frequency is 10 MHz, then the resolution, period, and frequency of the PWM output for each clock source are as shown in table 10-3.

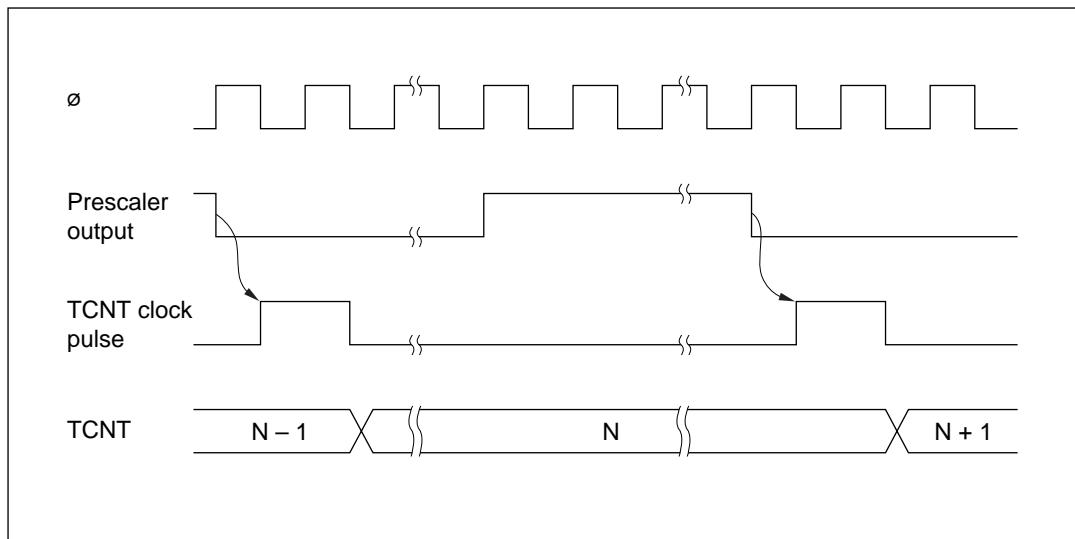
**Table 10-3 PWM Timer Parameters for 10 MHz System Clock**

Internal Clock Frequency	Resolution	PWM Period	PWM Frequency
$\phi_P/2$	200 ns	50 $\mu$ s	20 kHz
$\phi_P/8$	800 ns	200 $\mu$ s	5 kHz
$\phi_P/32$	3.2 $\mu$ s	800 $\mu$ s	1.25 kHz
$\phi_P/128$	12.8 $\mu$ s	3.2 ms	312.5 Hz
$\phi_P/256$	25.6 $\mu$ s	6.4 ms	156.3 Hz
$\phi_P/1024$	102.4 $\mu$ s	25.6 ms	39.1 Hz
$\phi_P/2048$	204.8 $\mu$ s	51.2 ms	19.5 Hz
$\phi_P/4096$	409.6 $\mu$ s	102.4 ms	9.8 Hz

## 10.3 Operation

### 10.3.1 Timer Increment

The PWM clock source is created by dividing the system clock ( $\phi$ ). The timer counter increments on a TCNT clock pulse generated from the falling edge of the prescaler output as shown in figure 10-2.



**Figure 10-2 TCNT Increment Timing**

### 10.3.2 PWM Operation

Figure 10-3 is a timing chart of the PWM operation.

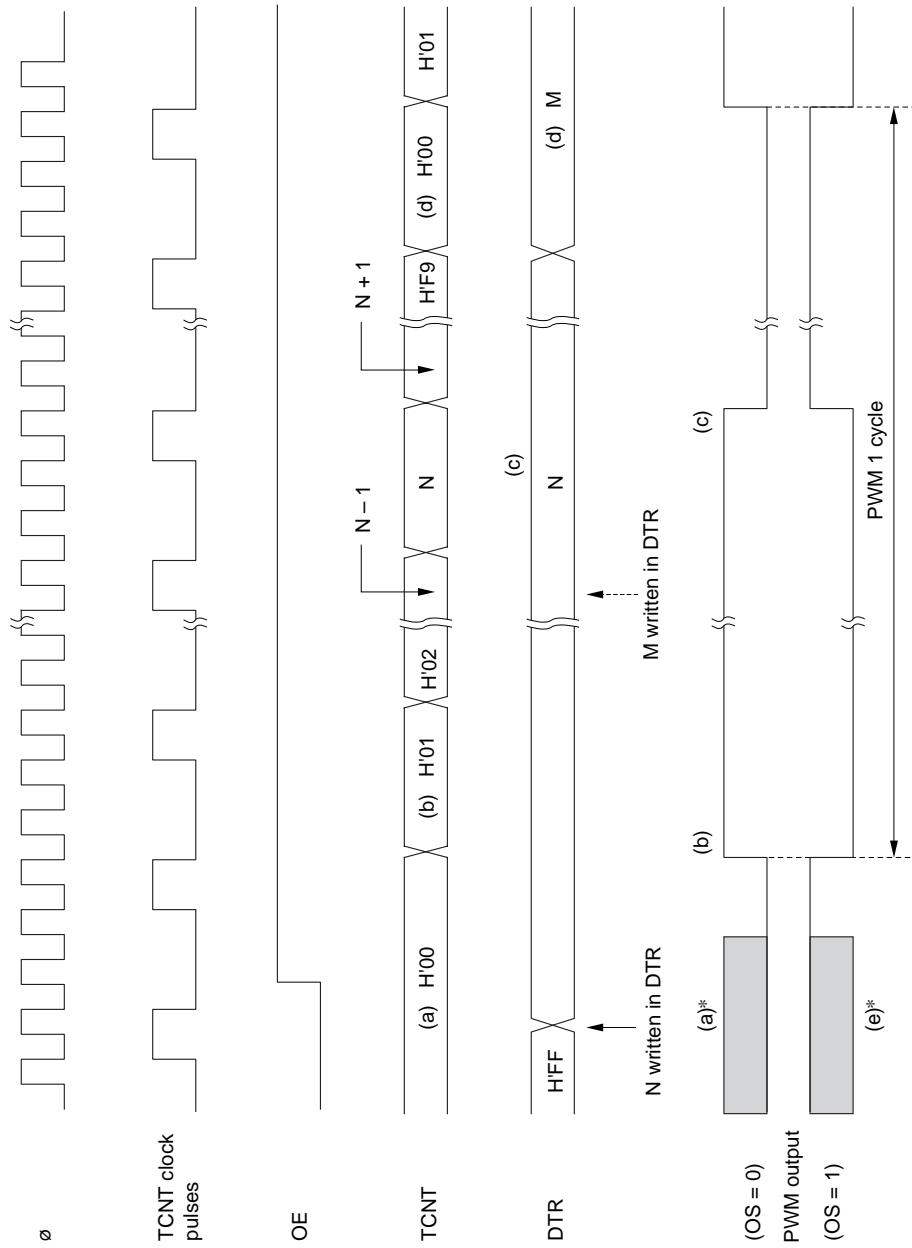


Figure 10-3 PWM Timing

## 1. Positive Logic (OS = 0)

**(1) When (OE = 0)—(a) in Figure 10-3:** The timer count is held at H'00 and PWM output is inhibited. [Pin 4<sub>6</sub> (for PW0) or pin 4<sub>7</sub> (for PW1) is used for port 4 input/output, and its state depends on the corresponding port 4 data register and data direction register.] Any value (such as N in figure 10-3) written in the DTR becomes valid immediately.

### (2) When (OE = 1)

- i) The timer counter begins incrementing. The PWM output goes high when TCNT changes from H'00 to H'01, unless DTR = H'00. [(b) in figure 10-3]
- ii) When the count passes the DTR value, the PWM output goes low. [(c) in figure 10-3]
- iii) If the DTR value is changed (by writing the data “M” in figure 10-3), the new value becomes valid after the timer count changes from H'F9 to H'00. [(d) in figure 10-3]

**2. Negative Logic (OS = 1)—(e) in Figure 10-3:** The operation is the same except that high and low are reversed in the PWM output. [(e) in figure 10-3]

## 10.4 Application Notes

Some notes on the use of the PWM timer module are given below.

- (1) Any necessary changes to the clock select bits (CKS2 to CKS0) and output select bit (OS) should be made before the output enable bit (OE) is set to 1.
- (2) If the DTR value is H'00, the duty cycle is 0% and PWM output remains constant at 0. If the DTR value is H'FA to H'FF, the duty cycle is 100% and PWM output remains constant at 1.

(For positive logic, 0 is low and 1 is high. For negative logic, 0 is high and 1 is low.)

# Section 11 Watchdog Timer

## 11.1 Overview

The H8/3437 Series has an on-chip watchdog timer (WDT) that can monitor system operation by resetting the CPU or generating a nonmaskable interrupt if a system crash allows the timer count to overflow.

When this watchdog function is not needed, the watchdog timer module can be used as an interval timer. In interval timer mode, it requests an OVF interrupt at each counter overflow.

### 11.1.1 Features

- Selection of eight clock sources
- Selection of two modes:
  - Watchdog timer mode
  - Interval timer mode
- Counter overflow generates an interrupt request or reset:
  - Reset or NMI request in watchdog timer mode
  - OVF interrupt request in interval timer mode

### 11.1.2 Block Diagram

Figure 11-1 is a block diagram of the watchdog timer.

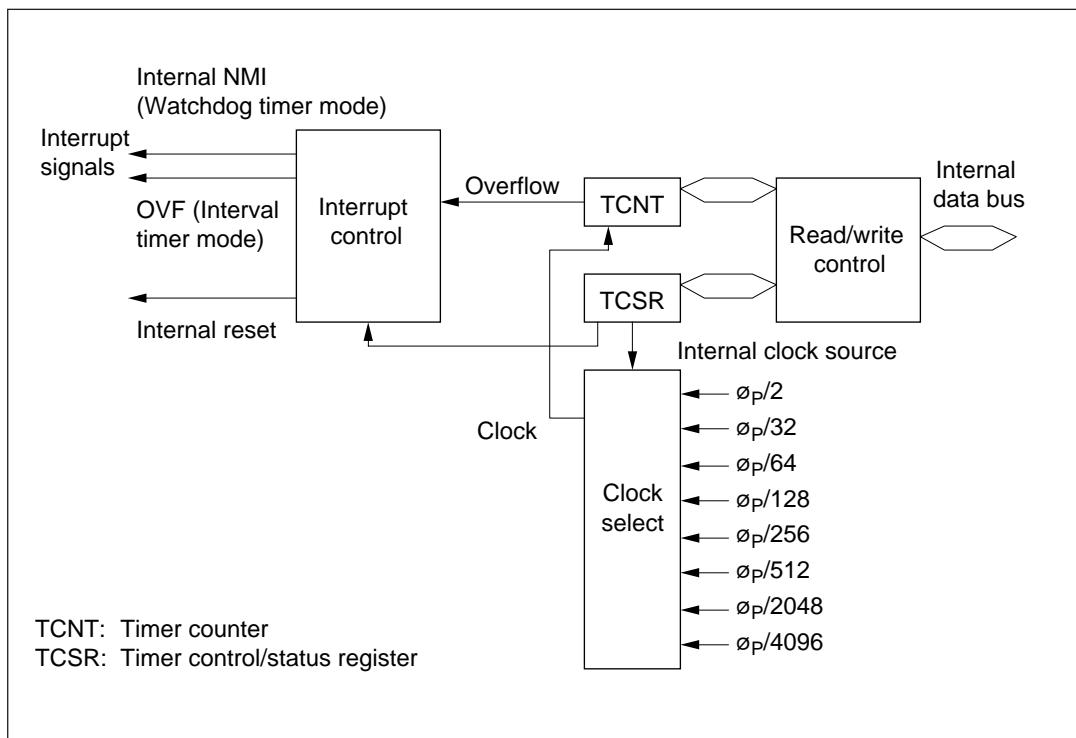


Figure 11-1 Block Diagram of Watchdog Timer

### 11.1.3 Register Configuration

Table 11-1 lists information on the watchdog timer registers.

Table 11-1 Register Configuration

Name	Abbreviation	R/W	Initial Value	Addresses	
				Write	Read
Timer control/status register	TCSR	R/(W)*	H'10	H'FFA8	H'FFA8
Timer counter	TCNT	R/W	H'00	H'FFA8	H'FFA9

Note: \* Software can write a 0 to clear the status flag bits, but cannot write 1.

## 11.2 Register Descriptions

### 11.2.1 Timer Counter (TCNT)

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

TCNT is an 8-bit readable/writable 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 TCSR. When the count overflows (changes from H'FF to H'00), an overflow flag (OVF) in TCSR is set to 1.

TCNT is initialized to H'00 by a reset and when the TME bit is cleared to 0.

Note: TCNT is more difficult to write to than other registers. See Section 11.2.3, Register Access, for details.

### 11.2.2 Timer Control/Status Register (TCSR)

Bit	7	6	5	4	3	2	1	0
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

Note: \* Software can write a 0 in bit 7 to clear the flag, but cannot write a 1 in this bit.

TCSR is an 8-bit readable/writable register that selects the timer mode and clock source and performs other functions. (TCSR is write-protected by a password. See section 11.2.3, Register Access, for details.)

Bits 7 to 5 and bit 3 are initialized to 0 by a reset and in the standby modes. Bits 2 to 0 are initialized to 0 by a reset, but retain their values in the standby modes.

**Bit 7—Overflow Flag (OVF):** Indicates that the watchdog timer count has overflowed.

Bit 7 OVF	Description	
0	To clear OVF, the CPU must read OVF after it has been set to 1, then write a 0 in this bit	(Initial value)
1	Set to 1 when TCNT changes from H'FF to H'00	

**Bit 6—Timer Mode Select (WT/IT):** Selects whether to operate in watchdog timer mode or interval timer mode. When TCNT overflows, an OVF interrupt request is sent to the CPU in interval timer mode. For watchdog timer mode, a reset or NMI interrupt is requested.

Bit 6 WT/IT	Description	
0	Interval timer mode (OVF request)	(Initial value)
1	Watchdog timer mode (reset or NMI request)	

**Bit 5—Timer Enable (TME):** Enables or disables the timer.

Bit 5 TME	Description	
0	TCNT is initialized to H'00 and stopped	(Initial value)
1	TCNT runs and requests a reset or an interrupt when it overflows	

**Bit 4—Reserved:** This bit cannot be modified and is always read as 1.

**Bit 3: Reset or NMI Select (RST/NMI):** Selects either an internal reset or the NMI function at watchdog timer overflow.

Bit 3 RST/NMI	Description	
0	NMI function enabled	(Initial value)
1	Reset function enabled	

**Bits 2–0—Clock Select (CKS2–CKS0):** These bits select one of eight clock sources obtained by dividing the system clock ( $\phi$ ).

The overflow interval is the time from when the watchdog timer counter begins counting from H'00 until an overflow occurs. In interval timer mode, OVF interrupts are requested at this interval.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Clock Source	Overflow Interval ( $\phi_P = 10$ MHz)
0	0	0	$\phi_P/2$	51.2 $\mu$ s (Initial value)
0	0	1	$\phi_P/32$	819.2 $\mu$ s
0	1	0	$\phi_P/64$	1.6 ms
0	1	1	$\phi_P/128$	3.3 ms
1	0	0	$\phi_P/256$	6.6 ms
1	0	1	$\phi_P/512$	13.1 ms
1	1	0	$\phi_P/2048$	52.4 ms
1	1	1	$\phi_P/4096$	104.9 ms

### 11.2.3 Register Access

The watchdog timer's TCNT and TCSR registers are more difficult to write to than other registers. The procedures for writing and reading these registers are given below.

**Writing to TCNT and TCSR:** Word access is required. Byte data transfer instructions cannot be used for write access.

The TCNT and TCSR registers have the same write address. The write data must be contained in the lower byte of a word written at this address. The upper byte must contain H'5A (password for TCNT) or H'A5 (password for TCSR). See figure 11-2. The result of the access depicted in figure 11-2 is to transfer the write data from the lower byte to TCNT or TCSR.

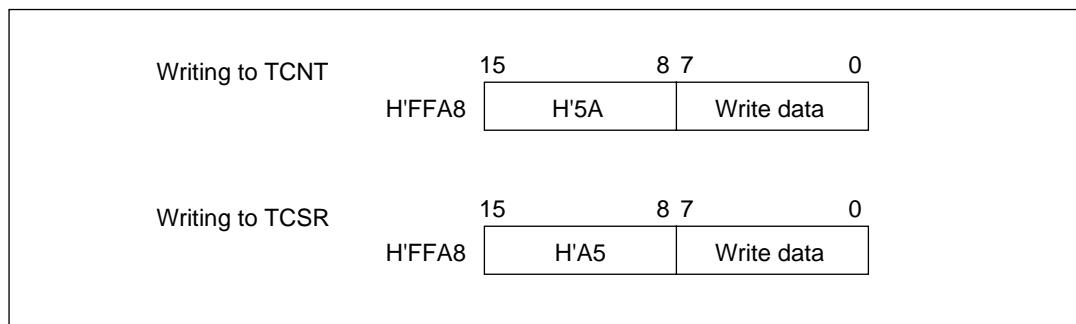


Figure 11-2 Writing to TCNT and TCSR

**Reading TCNT and TCSR:** The read addresses are H'FFA8 for TCSR and H'FFA9 for TCNT, as indicated in table 11-2.

These two registers are read like other registers. Byte access instructions can be used.

**Table 11-2 Read Addresses of TCNT and TCSR**

Read Address	Register
H'FFA8	TCSR
H'FFA9	TCNT

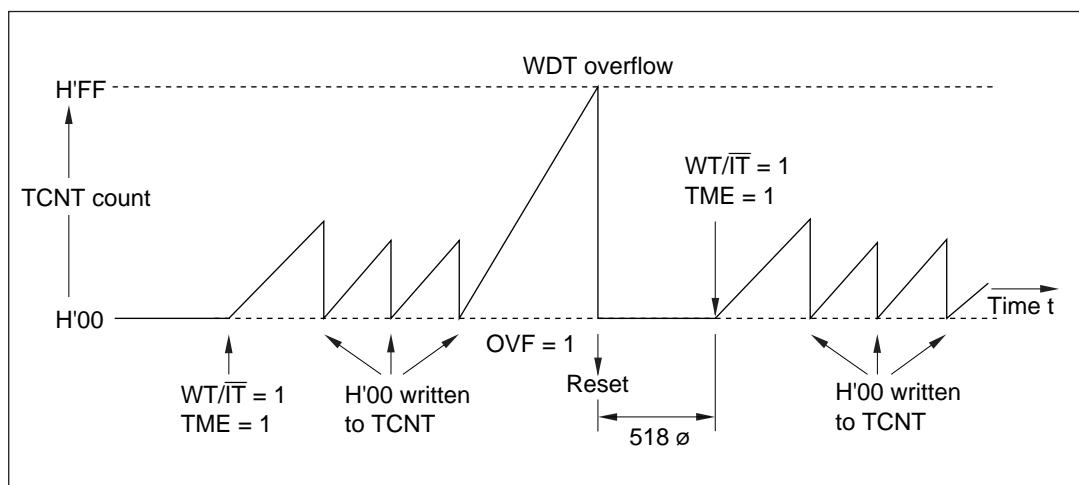
## 11.3 Operation

### 11.3.1 Watchdog Timer Mode

The watchdog timer function begins operating when software sets the WT/IT and TME bits to 1 in 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 entire chip is reset for 518 system clocks (518  $\phi$ ), or an NMI interrupt is requested. Figure 11-3 shows the operation.

NMI requests from the watchdog timer have the same vector as NMI requests from the  $\overline{\text{NMI}}$  pin. Avoid simultaneous handling of watchdog timer NMI requests and NMI requests from pin  $\overline{\text{NMI}}$ .

A reset from the watchdog timer has the same vector as an external reset from the RES pin. The reset source can be determined by the XRST bit in SYSCR.



**Figure 11-3 Operation in Watchdog Timer Mode**

### 11.3.2 Interval Timer Mode

Interval timer operation begins when the WT/IT bit is cleared to 0 and the TME bit is set to 1.

In interval timer mode, an OVF request is generated each time the timer count overflows. This function can be used to generate OVF requests at regular intervals. See figure 11-4.

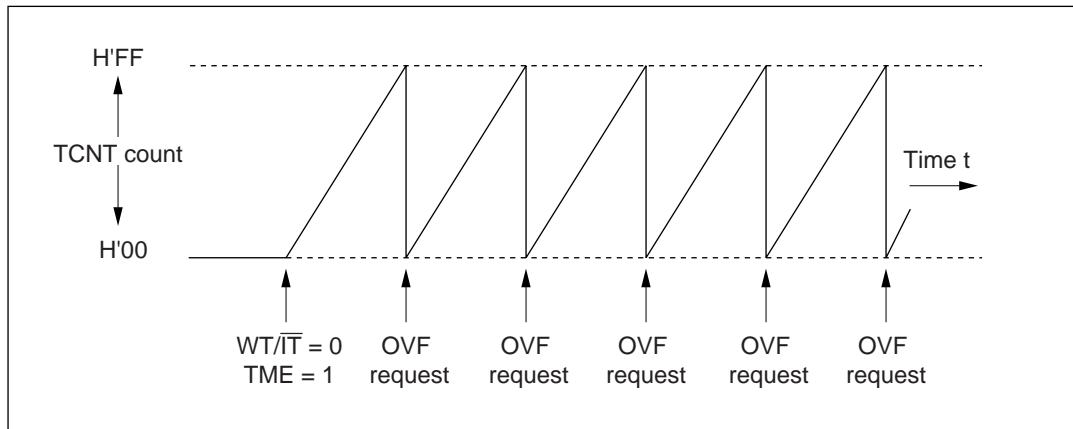


Figure 11-4 Operation in Interval Timer Mode

### 11.3.3 Setting the Overflow Flag

The OVF bit is set to 1 when the timer count overflows. Simultaneously, the WDT module requests an internal reset, NMI, or OVF interrupt. The timing is shown in figure 11-5.

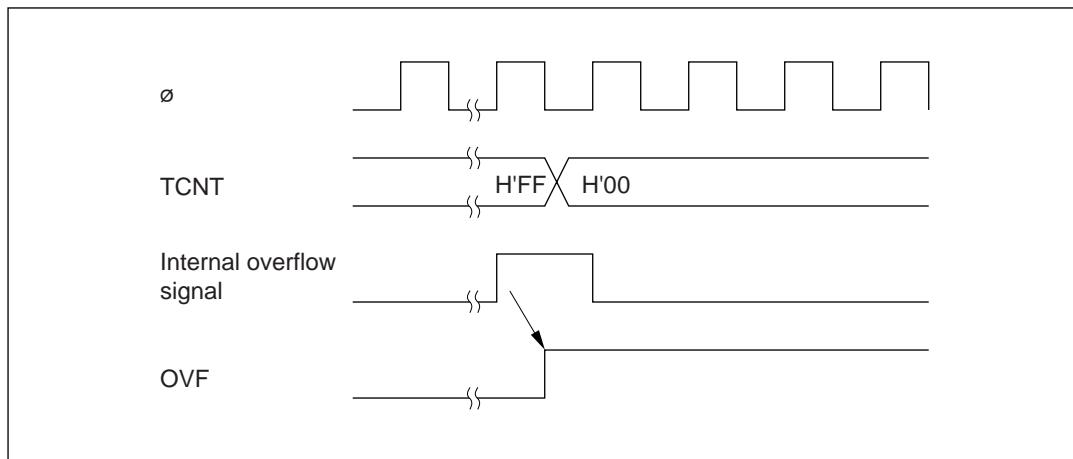


Figure 11-5 Setting the OVF Bit

## 11.4 Application Notes

### 11.4.1 Contention between TCNT Write and Increment

If a timer counter clock pulse is generated during the T<sub>3</sub> state of a write cycle to the timer counter, the write takes priority and the timer counter is not incremented. See figure 11-6.

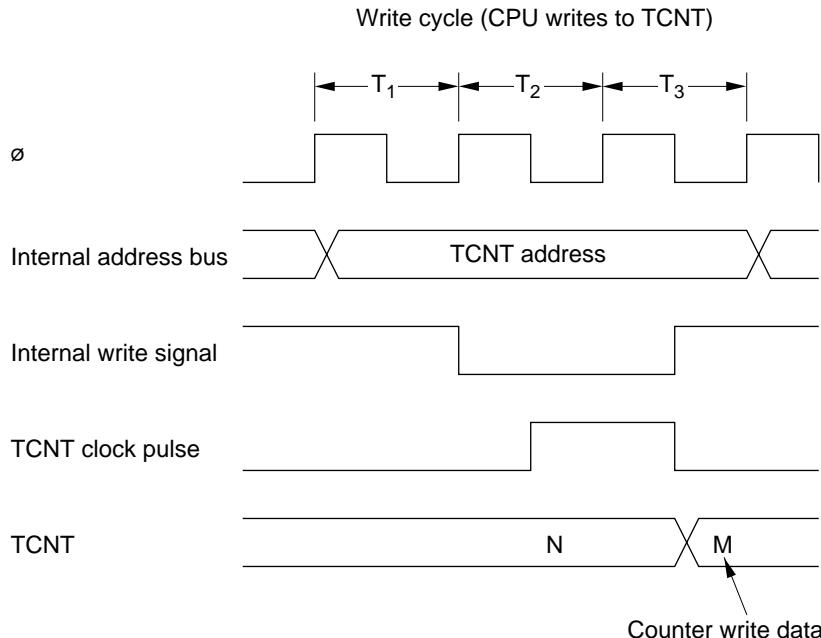


Figure 11-6 TCNT Write-Increment Contention

### 11.4.2 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.

### 11.4.3 Recovery from Software Standby Mode

TCSR bits, except bits 0–2, and the TCNT counter are reset when the chip recovers from software standby mode. Re-initialize the watchdog timer as necessary to resume normal operation.

# Section 12 Serial Communication Interface

## 12.1 Overview

The H8/3437 Series includes two serial communication interface channels (SCI0 and SCI1) for transferring serial data to and from other chips. Either synchronous or asynchronous communication can be selected.

### 12.1.1 Features

The features of the on-chip serial communication interface are:

- Asynchronous mode

The H8/3437 Series can communicate with a UART (Universal Asynchronous Receiver/Transmitter), ACIA (Asynchronous Communication Interface Adapter), or other chip that employs standard asynchronous serial communication. It also has a multiprocessor communication function for communication with other processors. Twelve data formats are available.

- Data length: 7 or 8 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Multiprocessor bit: 1 or 0
- Error detection: Parity, overrun, and framing errors
- Break detection: When a framing error occurs, the break condition can be detected by reading the level of the RxD line directly.

- Synchronous mode

The SCI can communicate with chips able to perform clocked synchronous data transfer.

- 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 SCI can operate on an internal clock signal from the baud rate generator, or an external clock signal input at the SCK0 or SCK1 pin.

- Four interrupts

TDR-empty, TSR-empty, receive-end, and receive-error interrupts are requested independently.

## 12.1.2 Block Diagram

Figure 12-1 shows a block diagram of one serial communication interface channel.

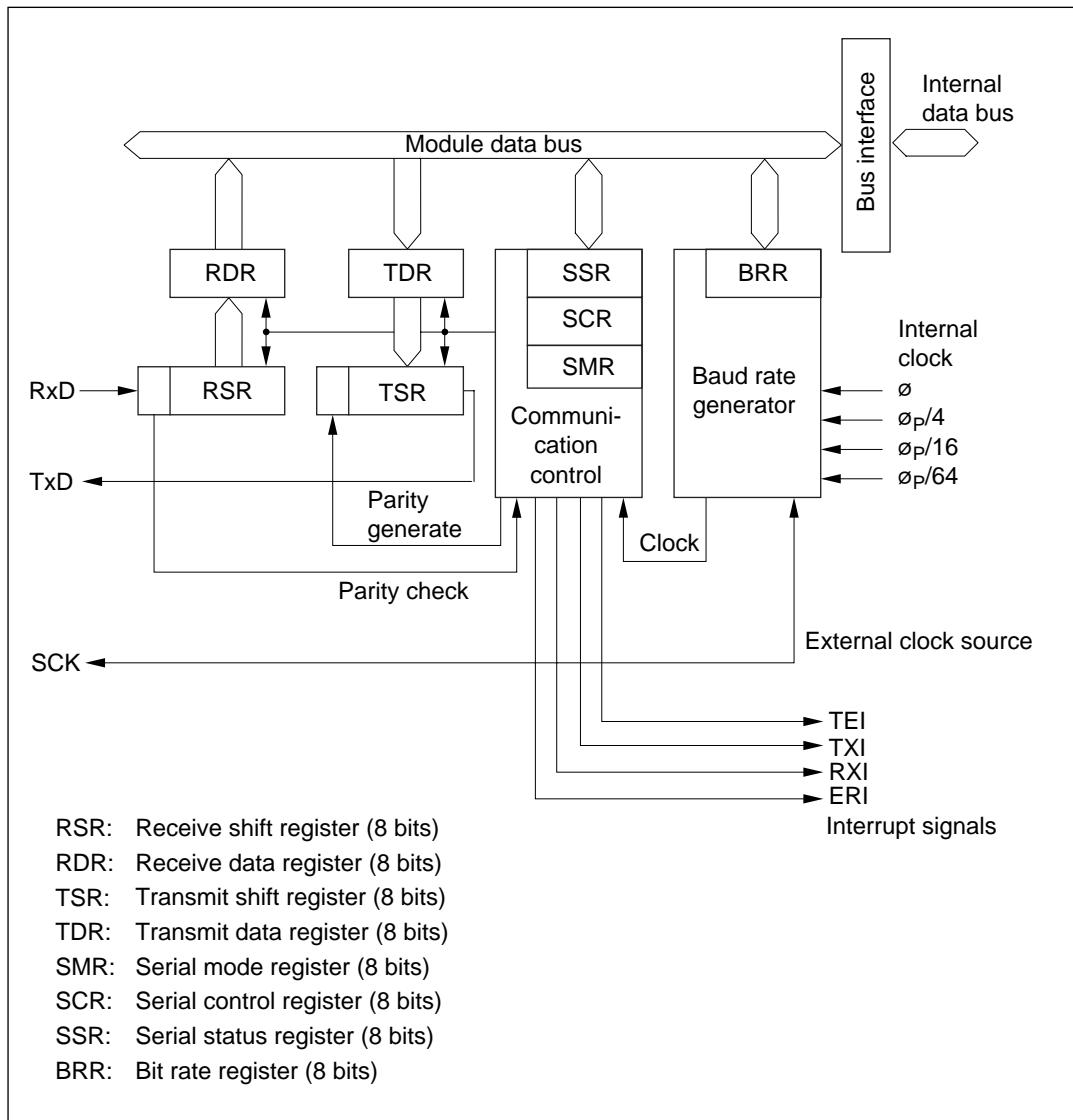


Figure 12-1 Block Diagram of Serial Communication Interface

### 12.1.3 Input and Output Pins

Table 12-1 lists the input and output pins used by the SCI module.

**Table 12-1 SCI Input/Output Pins**

Channel	Name	Abbr.	I/O	Function
0	Serial clock input/output	SCK0	Input/output	SCI0 clock input and output
	Receive data input	RxD0	Input	SCI0 receive data input
	Transmit data output	TxD0	Output	SCI0 transmit data output
1	Serial clock input/output	SCK1	Input/output	SCI1 clock input and output
	Receive data input	RxD1	Input	SCI1 receive data input
	Transmit data output	TxD1	Output	SCI1 transmit data output

Note: In this manual, the channel subscript has been deleted, and only SCK, RxD, and TxD are used.

## 12.1.4 Register Configuration

Table 12-2 lists the SCI registers. These registers specify the operating mode (synchronous or asynchronous), data format and bit rate, and control the transmit and receive sections.

**Table 12-2 SCI Registers**

Channel	Name	Abbr.	R/W	Value	Address
0	Receive shift register	RSR	—*1	—*1	—*1
	Receive data register	RDR	R	H'00	H'FFDD
	Transmit shift register	TSR	—*1	—*1	—*1
	Transmit data register	TDR	R/W	H'FF	H'FFDB
	Serial mode register	SMR*3	R/W	H'00	H'FFD8
	Serial control register	SCR	R/W	H'00	H'FFDA
	Serial status register	SSR	R/(W)*2	H'84	H'FFDC
	Bit rate register	BRR*3	R/W	H'FF	H'FFD9
1	Receive shift register	RSR	—*1	—*1	—*1
	Receive data register	RDR	R	H'00	H'FF8D
	Transmit shift register	TSR	—*1	—*1	—*1
	Transmit data register	TDR	R/W	H'FF	H'FF8B
	Serial mode register	SMR	R/W	H'00	H'FF88
	Serial control register	SCR	R/W	H'00	H'FF8A
	Serial status register	SSR	R/(W)*2	H'84	H'FF8C
	Bit rate register	BRR	R/W	H'FF	H'FF89
0 and 1	Serial/timer control register	STCR	R/W	H'00	H'FFC3

Notes:

1. Cannot be read or written to.
2. Software can write a 0 to clear the flags in bits 7 to 3, but cannot write 1 in these bits.
3. SMR and BRR have the same addresses as I<sup>2</sup>C bus interface registers ICCR and ICSR. For the access switching method and other details, see section 13, I<sup>2</sup>C Bus Interface.

## 12.2 Register Descriptions

### 12.2.1 Receive Shift Register (RSR)

Bit	7	6	5	4	3	2	1	0
Read/Write	—	—	—	—	—	—	—	—

RSR is a shift register that converts incoming serial data to parallel data. When one data character has been received, it is transferred to the receive data register (RDR).

The CPU cannot read or write RSR directly.

### 12.2.2 Receive Data Register (RDR)

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

RDR stores received data. As each character is received, it is transferred from RSR to RDR, enabling RSR to receive the next character. This double-buffering allows the SCI to receive data continuously.

RDR is a read-only register. RDR is initialized to H'00 by a reset and in the standby modes.

### 12.2.3 Transmit Shift Register (TSR)

Bit	7	6	5	4	3	2	1	0
Read/Write	—	—	—	—	—	—	—	—

TSR is a shift register that converts parallel data to serial transmit data. When transmission of one character is completed, the next character is moved from the transmit data register (TDR) to TSR and transmission of that character begins. If the TDRE bit is still set to 1, however, nothing is transferred to TSR.

The CPU cannot read or write TSR directly.

## 12.2.4 Transmit Data Register (TDR)

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

TDR is an 8-bit readable/writable register that holds the next data to be transmitted. When TSR becomes empty, the data written in TDR is transferred to TSR. Continuous data transmission is possible by writing the next data in TDR while the current data is being transmitted from TSR.

TDR is initialized to H'FF by a reset and in the standby modes.

## 12.2.5 Serial Mode Register (SMR)

Bit	7	6	5	4	3	2	1	0
	C/A	CHR	PE	O/E	STOP	MP	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

SMR is an 8-bit readable/writable register that controls the communication format and selects the clock source of the on-chip baud rate generator. It is initialized to H'00 by a reset and in the standby modes. For further information on the SMR settings and communication formats, see tables 12-5 and 12-7 in section 12.3, Operation.

**Bit 7—Communication Mode (C/A):** This bit selects asynchronous or synchronous communication mode.

Bit 7 C/A	Description	
0	Asynchronous communication	(Initial value)
1	Synchronous communication	

**Bit 6—Character Length (CHR):** This bit selects the character length in asynchronous mode. It is ignored in synchronous mode.

Bit 6	CHR	Description	
0	8 bits per character		(Initial value)
1	7 bits per character (Bits 0 to 6 of TDR and RDR are used for transmitting and receiving, respectively.)		

**Bit 5—Parity Enable (PE):** This bit selects whether to add a parity bit in asynchronous mode. It is ignored in synchronous mode, and when a multiprocessor format is used.

Bit 5	PE	Description	
0	Transmit: No parity bit is added. Receive: Parity is not checked.		(Initial value)
1	Transmit: A parity bit is added. Receive: Parity is checked.		

**Bit 4—Parity Mode (O/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, or when a multiprocessor format is used. It is also ignored in 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 synchronous mode.

Bit 3 STOP	Description	
0	One stop bit Transmit: One stop bit is added. Receive: One stop bit is checked to detect framing errors.	(Initial value)
1	Two stop bits Transmit: Two stop bits are added. Receive: The first stop bit is checked to detect framing errors. If the second stop bit is a space (0), it is regarded as the next start bit.	

**Bit 2—Multiprocessor Mode (MP):** This bit selects the multiprocessor format in asynchronous communication. When multiprocessor format is selected, the parity settings of the parity enable bit (PE) and parity mode bit (O/E) are ignored. The MP bit is ignored in synchronous communication.

The MP bit is valid only when the MPE bit in the serial/timer control register (STCR) is set to 1. When the MPE bit is cleared to 0, the multiprocessor communication function is disabled regardless of the setting of the MP bit.

Bit 2 MP	Description	
0	Multiprocessor communication function is disabled.	(Initial value)
1	Multiprocessor communication function is enabled.	

**Bits 1 and 0—Clock Select 1 and 0 (CKS1 and CKS0):** These bits select the clock source of the on-chip baud rate generator.

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	$\emptyset$ clock	(Initial value)
0	1	$\emptyset_P/4$ clock	
1	0	$\emptyset_P/16$ clock	
1	1	$\emptyset_P/64$ clock	

## 12.2.6 Serial Control Register (SCR)

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0

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

SCR is an 8-bit readable/writable register that enables or disables various SCI functions.

It is initialized to H'00 by a reset and in the standby modes.

**Bit 7—Transmit Interrupt Enable (TIE):** This bit enables or disables the TDR-empty 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 TDR-empty interrupt request (TXI) is disabled.	(Initial value)
1	The TDR-empty 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, and the receive error interrupt (ERI) requested when the overrun error (ORER), framing error (FER), or parity error (PER) bit in the serial status register (SSR) is set to 1.

### Bit 6

#### RIE    Description

0	The receive-end interrupt (RXI) and receive-error (ERI) requests are disabled.	(Initial value)
1	The receive-end interrupt (RXI) and receive-error (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 for general-purpose I/O.	(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 for general-purpose I/O.	(Initial value)
1	The receive function is enabled. The RxD pin is used for input.	

**Bit 3—Multiprocessor Interrupt Enable (MPIE):** When serial data is received in a multiprocessor format, this bit enables or disables the receive-end interrupt (RXI) and receive-error interrupt (ERI) until data with the multiprocessor bit set to 1 is received. It also enables or disables the transfer of received data from RSR to RDR, and enables or disables setting of the RDRF, FER, PER, and ORER bits in the serial status register (SSR).

The MPIE bit is ignored when the MP bit is cleared to 0, and in synchronous mode.

Clearing the MPIE bit to 0 disables the multiprocessor receive interrupt function. In this condition data is received regardless of the value of the multiprocessor bit in the receive data.

Setting the MPIE bit to 1 enables the multiprocessor receive interrupt function. In this condition, if the multiprocessor bit in the receive data is 0, the receive-end interrupt (RXI) and receive-error interrupt (ERI) are disabled, the receive data is not transferred from RSR to RDR, and the RDRF, FER, PER, and ORER bits in the serial status register (SSR) are not set. If the multiprocessor bit is 1, however, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0, the receive data is transferred from RSR to RDR, the FER, PER, and ORER bits can be set, and the receive-end and receive-error interrupts are enabled.

**Bit 3**

MPIE	Description	
0	The multiprocessor receive interrupt function is disabled. (Normal receive operation)	(Initial value)
1	The multiprocessor receive interrupt function is enabled. During the interval before data with the multiprocessor bit set to 1 is received, the receive interrupt request (RXI) and receive-error interrupt request (ERI) are disabled, the RDRF, FER, PER, and ORER bits are not set in the serial status register (SSR), and no data is transferred from the RSR to the RDR. The MPIE bit is cleared at the following times: (1) When 0 is written in MPIE. (2) When data with the multiprocessor bit set to 1 is received.	

**Bit 2—Transmit-End Interrupt Enable (TEIE):** This bit enables or disables the TSR-empty interrupt (TEI) requested when the transmit-end bit (TEND) in the serial status register (SSR) is set to 1.

**Bit 2**

TEIE	Description	
0	The TSR-empty interrupt request (TEI) is disabled.	(Initial value)
1	The TSR-empty interrupt request (TEI) is enabled.	

**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 When C/A = 1, the serial clock signal is output at the SCK pin. When C/A = 0, output depends on the CKE0 bit.	(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 asynchronous 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 synchronous mode is selected.

For further information on the communication format and clock source selection, see table 12-6 in section 12.3, 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.	

## 12.2.7 Serial Status Register (SSR)

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: \* Software can write a 0 to clear the flags, but cannot write a 1 in these bits.

SSR is an 8-bit register that indicates transmit and receive status. It is initialized to H'84 by a reset and in the standby modes.

**Bit 7—Transmit Data Register Empty (TDRE):** This bit indicates when transmit data can safely be written in TDR.

Bit 7	TDRE	Description	
0		To clear TDRE, the CPU must read TDRE after it has been set to 1, then write a 0 in this bit.	
1		This bit is set to 1 at the following times: (1) When TDR contents are transferred to TSR. (2) When the TE bit in SCR is cleared to 0.	(Initial value)

**Bit 6—Receive Data Register Full (RDRF):** This bit indicates when one character has been received and transferred to RDR.

Bit 6	RDRF	Description	
0		To clear RDRF, the CPU must read RDRF after it has been set to 1, then write a 0 in this bit.	(Initial value)
1		This bit is set to 1 when one character is received without error and transferred from RSR to RDR.	

**Bit 5—Overrun Error (ORER):** This bit indicates an overrun error during reception.

Bit 5 ORER	Description	
0	To clear ORER, the CPU must read ORER after it has been set to 1, then write a 0 in this bit.	(Initial value)
1	This bit is set to 1 if reception of the next character ends while the receive data register is still full (RDRF = 1).	

**Bit 4—Framing Error (FER):** This bit indicates a framing error during data reception in asynchronous mode. It has no meaning in synchronous mode.

Bit 4 FER	Description	
0	To clear FER, the CPU must read FER after it has been set to 1, then write a 0 in this bit.	(Initial value)
1	This bit is set to 1 if a framing error occurs (stop bit = 0).	

**Bit 3—Parity Error (PER):** This bit indicates a parity error during data reception in the asynchronous mode, when a communication format with parity bits is used.

This bit has no meaning in the synchronous mode, or when a communication format without parity bits is used.

Bit 3 PER	Description	
0	To clear PER, the CPU must read PER after it has been set to 1, then write a 0 in this bit.	(Initial value)
1	This bit is set to 1 when a parity error occurs (the parity of the received data does not match the parity selected by the O/E bit in SMR).	

**Bit 2—Transmit End (TEND):** This bit indicates that the serial communication interface has stopped transmitting because there was no valid data in TDR when the last bit of the current character was transmitted. The TEND bit is also set to 1 when the TE bit in the serial control register (SCR) is cleared to 0.

The TEND bit is a read-only bit and cannot be modified directly. To use the TEI interrupt, first start transmitting data, which clears TEND to 0, then set TEIE to 1.

<b>Bit 2</b>		
<b>TEND</b>	<b>Description</b>	
0	To clear TEND, the CPU must read TDRE after TDRE has been set to 1, then write a 0 in TDRE	
1	This bit is set to 1 when: (1) TE = 0 (2) TDRE = 1 at the end of transmission of a character	(Initial value)

**Bit 1—Multiprocessor Bit (MPB):** Stores the value of the multiprocessor bit in data received in a multiprocessor format in asynchronous communication mode. This bit retains its previous value in synchronous mode, when a multiprocessor format is not used, or when the RE bit is cleared to 0 even if a multiprocessor format is used.

MPB can be read but not written.

<b>Bit 1</b>		
<b>MPB</b>	<b>Description</b>	
0	Multiprocessor bit = 0 in receive data.	(Initial value)
1	Multiprocessor bit = 1 in receive data.	

**Bit 0—Multiprocessor Bit Transfer (MPBT):** Stores the value of the multiprocessor bit inserted in transmit data when a multiprocessor format is used in asynchronous communication mode. The MPBT bit is double-buffered in the same way as TSR and TDR. The MPBT bit has no effect in synchronous mode, or when a multiprocessor format is not used.

<b>Bit 0</b>		
<b>MPBT</b>	<b>Description</b>	
0	Multiprocessor bit = 0 in transmit data.	(Initial value)
1	Multiprocessor bit = 1 in transmit data.	

## 12.2.8 Bit Rate Register (BRR)

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

BRR is an 8-bit register that, together with the CKS1 and CKS0 bits in SMR, determines the bit rate output by the baud rate generator.

BRR is initialized to H'FF by a reset and in the standby modes.

Tables 12-3 and 12-4 show examples of BRR settings.

**Table 12-3 Examples of BRR Settings in Asynchronous Mode (When  $\phi_P = \phi$ )**

Bit Rate	$\phi$ Frequency (MHz)											
	1			1.2288			2			2.097152		
	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	—	—	—	0	6	-2.48
19200	—	—	—	0	1	0	—	—	—	—	—	—
31250	0	0	0	—	—	—	0	1	0	—	—	—
38400	—	—	—	0	0	0	—	—	—	—	—	—

Note: If possible, the error should be within 1%.

In the shaded section, if  $\phi_P = \phi/2$ , the bit rate is cut in half. In this case, BRR settings for the desired bit rate should be referenced from the column of one-half the actual system clock frequency ( $\phi$ ).

**Table 12-3 Examples of BRR Settings in Asynchronous Mode (When  $\phi_P = \phi$ ) (cont)**

Bit Rate	$\phi$ Frequency (MHz)											
	2.4576			3			3.6864			4		
	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	9	-2.34	0	11	0	0	12	+0.16
19200	0	3	0	0	4	-2.34	0	5	0	—	—	—
31250	—	—	—	0	2	0	—	—	—	0	3	0
38400	0	1	0	—	—	—	0	2	0	—	—	—

**Table 12-3 Examples of BRR Settings in Asynchronous Mode (When  $\phi_P = \phi$ ) (cont)**

Bit Rate	$\phi$ Frequency (MHz)											
	4.9152			5			6			6.144		
	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	+0.16	2	79	0
300	1	127	0	1	129	+0.16	1	155	+0.16	1	159	0
600	0	255	0	1	64	+0.16	1	77	+0.16	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	-2.34	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	-2.34	0	4	0

Note: If possible, the error should be within 1%.

In the shaded section, if  $\phi_P = \phi/2$ , the bit rate is cut in half. In this case, BRR settings for the desired bit rate should be referenced from the column of one-half the actual system clock frequency ( $\phi$ ).

**Table 12-3 Examples of BRR Settings in Asynchronous Mode (When  $\phi_P = \phi$ ) (cont)**

Bit Rate	$\phi$ Frequency (MHz)											
	7.3728			8			9.8304			10		
	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	2	177	-0.25
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

**Table 12-3 Examples of BRR Settings in Asynchronous Mode (When  $\phi_P = \phi$ ) (cont)**

Bit Rate	$\phi$ Frequency (MHz)											
	12			12.288			14.7456			16		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	212	+0.03	2	217	+0.08	3	64	+0.76	3	70	+0.03
150	2	155	+0.16	2	159	0	2	191	0	2	207	+0.16
300	2	77	+0.16	2	79	0	2	95	0	2	103	+0.16
600	1	155	+0.16	1	159	0	1	191	0	1	207	+0.16
1200	1	77	+0.16	1	79	0	1	95	0	1	103	+0.16
2400	0	155	+0.16	0	159	0	0	191	0	0	207	+0.16
4800	0	77	+0.16	0	79	0	0	95	0	0	103	+0.16
9600	0	38	+0.16	0	39	0	0	47	0	0	51	+0.16
19200	0	19	-2.34	0	19	0	0	23	0	0	25	+0.16
31250	0	11	0	0	11	+2.4	0	14	-1.7	0	15	0
38400	0	9	-2.34	0	9	0	0	11	0	0	12	+0.16

Note: If possible, the error should be within 1%.

In the shaded section, if  $\phi_P = \phi/2$ , the bit rate is cut in half. In this case, BRR settings for the desired bit rate should be referenced from the column of one-half the actual system clock frequency ( $\phi$ ).

$$B = \frac{F \times 10^6}{64 \times 2^{2n-1} \times (N + 1)} \rightarrow N = \frac{F \times 10^6}{64 \times 2^{2n-1} \times B} - 1$$

B: Bit rate (bits/second)

N: BRR value ( $0 \leq N \leq 255$ )

F:  $\phi_P$  (MHz) when  $n \neq 0$ , or  $\phi$  (MHz) when  $n = 0$

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	$\emptyset$
1	0	1	$\phi_P/4$
2	1	0	$\phi_P/16$
3	1	1	$\phi_P/64$

Bit rate error can be calculated with the formula below.

$$\text{Error (\%)} = \left\{ \frac{F \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

**Table 12-4 Examples of BRR Settings in Synchronous Mode (When  $\phi_P = \phi$ )**

Bit Rate	$\phi$ Frequency (MHz)															
	1		2		4		5		8		10		16			
n	N	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	—	—
1 k	0	249	1	124	1	249	—	—	2	124	—	—	2	249	—	—
2.5 k	0	99	0	199	1	99	1	124	1	199	1	249	2	99	—	—
5 k	0	49	0	99	0	199	0	249	1	99	1	124	1	199	—	—
10 k	0	24	0	49	0	99	0	124	0	199	0	249	1	99	—	—
25 k	0	9	0	19	0	39	0	49	0	79	0	99	0	159	—	—
50 k	0	4	0	9	0	19	0	24	0	39	0	49	0	79	—	—
100 k	—	—	0	4	0	9	—	—	0	19	0	24	0	39	—	—
250 k	0	0*	0	1	0	3	0	4	0	7	0	9	0	15	—	—
500 k	0	0*	0	1	—	—	0	3	0	4	0	4	0	7	—	—
1 M	0	0*	—	—	0	1	—	—	0	1	—	—	0	3	—	—
2.5 M	0	0*	—	—	—	—	0	0*	—	—	0	0*	—	—	—	—
4 M	0	0*	—	—	—	—	0	0*	—	—	0	0*	—	—	—	—

Notes: In the shaded section, if  $\phi_P = \phi/2$ , the bit rate is cut in half. In this case, BRR settings for the desired bit rate should be referenced from the column of one-half the actual system clock frequency ( $\phi$ ).

Blank: No setting is available.

—: A setting is available, but the bit rate is inaccurate.

\*: Continuous transfer is not possible.

$$B = \frac{F \times 10^6}{8 \times 2^{2n-1} \times (N + 1)} \rightarrow N = \frac{F \times 10^6}{8 \times 2^{2n-1} \times B} - 1$$

B: Bit rate (bits per second)

N: BRR value ( $0 \leq N \leq 255$ )

F:  $\phi_P$  (MHz) when  $n \neq 0$ , or  $\phi$  (MHz) when  $n = 0$

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	$\phi$
1	0	1	$\phi_P/4$
2	1	0	$\phi_P/16$
3	1	1	$\phi_P/64$

## 12.2.9 Serial/Timer Control Register (STCR)

Bit	7	6	5	4	3	2	1	0
	IICS	IICD	IICX	IICE	STAC	MPE	ICKS1	ICKS0
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

STCR is an 8-bit readable/writable register that controls the SCI operating mode and selects the TCNT clock source in the 8-bit timers. STCR is initialized to H'00 by a reset.

**Bits 7 to 4—I<sup>2</sup>C Control (IICS, IICD, IICX, IICE):** These bits control operation of the I<sup>2</sup>C bus interface. For details, refer to section 13, I<sup>2</sup>C Bus Interface.

**Bit 3—Slave Input Switch (STAC):** Controls the input pin of the host interface. For details, section 14, Host Interface.

**Bit 2—Multiprocessor Enable (MPE):** Enables or disables the multiprocessor communication function on channels SCI0 and SCI1.

Bit 2			
	MPE	Description	
0		The multiprocessor communication function is disabled, regardless of the setting of the MP bit in SMR.	(Initial value)
1		The multiprocessor communication function is enabled. The multi-processor format can be selected by setting the MP bit in SMR to 1.	

**Bits 1 and 0—Internal Clock Source Select 1 and 0 (ICKS1, ICKS0):** These bits select the clock input to the timer counters (TCNT) in the 8-bit timers. For details, see section 9, 8-Bit Timers.

## 12.3 Operation

### 12.3.1 Overview

The SCI supports serial data transfer in two modes. In asynchronous mode each character is synchronized individually. In synchronous mode communication is synchronized with a clock signal.

The selection of asynchronous or synchronous mode and the communication format depend on SMR settings as indicated in table 12-5. The clock source depends on the settings of the C/A bit in the SMR and the CKE1 and CKE0 bits in SCR as indicated in table 12-6.

#### Asynchronous Mode

- Data length: 7 or 8 bits can be selected.
- A parity bit or multiprocessor bit can be added, and stop bit lengths of 1 or 2 bits can be selected. (These selections determine the communication format and character length.)
- Framing errors (FER), parity errors (PER), and overrun errors (ORER) can be detected in receive data, and the line-break condition can be detected.
- SCI clock source: An internal or external clock source can be selected.
- Internal clock: The SCI is clocked by the on-chip baud rate generator and can output a clock signal at the bit-rate frequency.
- External clock: The external clock frequency must be 16 times the bit rate. (The on-chip baud rate generator is not used.)

#### Synchronous Mode

- Communication format: The data length is 8 bits.
- Overrun errors (ORER) can be detected in receive data.
- SCI clock source: An internal or external clock source can be selected.
- Internal clock: The SCI is clocked by the on-chip baud rate generator and outputs a serial clock signal to external devices.
- External clock: The on-chip baud rate generator is not used. The SCI operates on the input serial clock.

**Table 12-5** Communication Formats Used by SCI

SMR Settings						Communication Format			
Bit 7 C/A	Bit 6 CHR	Bit 2 MP	Bit 5 PE	Bit 3 STOP	Mode	Data Length	Multi-processor Bit	Parity Bit	Stop-Bit Length
0	0	0	0	0 1	Asynchronous mode	8 bits	None	None	1 bit 2 bits
				1 0 1				Present	1 bit 2 bits
			1	0 1		7 bits		None	1 bit 2 bits
				1 0 1				Present	1 bit 2 bits
0	1	—	0 1	Asynchronous mode (multi-processor format)	8 bits 7 bits		Present	None	1 bit 2 bits 1 bit 2 bits
1	—	—	—	Synchronous mode	8 bits				None

**Table 12-6** SCI Clock Source Selection

SMR			SCR			Serial Transmit/Receive Clock		
Bit 7 C/A	Bit 1 CKE1	Bit 0 CKE0	Mode	Clock Source	SCK Pin Function			
0	0	0 1	Async	Internal	Input/output port (not used by SCI)			
					Serial clock output at bit rate			
	1	0 1		External	Serial clock input at $16 \times$ bit rate			
1	0	0 1	Sync	Internal	Serial clock output			
	1	0 1		External	Serial clock input			

### 12.3.2 Asynchronous Mode

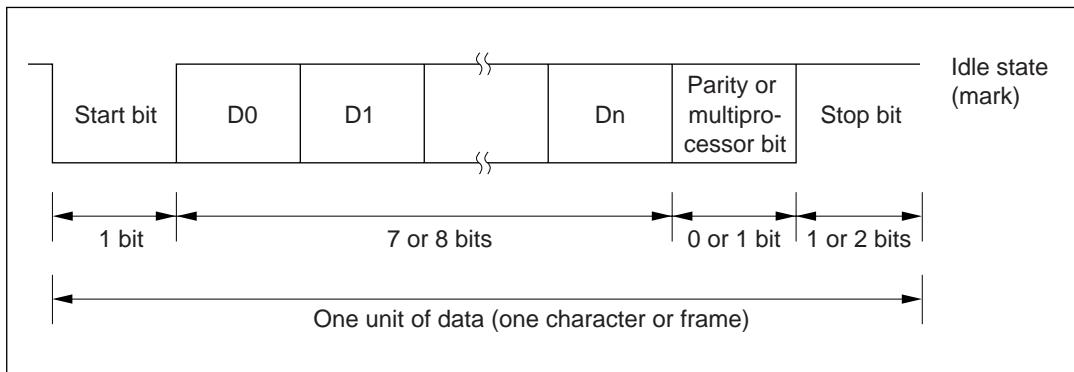
In asynchronous mode, each transmitted or received character is individually synchronized by framing it with a start bit and stop bit.

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 12-2 shows the general format of one character sent or received in 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 or multiprocessor bit, if present, then the stop bit or bits (high) confirming the end of the frame.

In receiving, the SCI synchronizes on the falling edge of the start bit, and samples each bit at the center of the bit (at the 8th cycle of the internal serial clock, which runs at 16 times the bit rate).



**Figure 12-2 Data Format in Asynchronous Mode**

**1. Data Format:** Table 12-7 lists the data formats that can be sent and received in asynchronous mode. Twelve formats can be selected by bits in the serial mode register (SMR).

**Table 12-7 Data Formats in Asynchronous Mode**

SMR Bits					1	2	3	4	5	6	7	8	9	10	11	12
CHR	PE	MP	STOP		S								8-bit data		STOP	
0	0	0	0	1	S								8-bit data		STOP	STOP
0	1	0	0	0	S								8-bit data	P	STOP	
0	1	0	1	1	S								8-bit data	P	STOP	STOP
1	0	0	0	0	S								7-bit data		STOP	
1	0	0	1	1	S								7-bit data		STOP	STOP
1	1	0	0	0	S								7-bit data	P	STOP	
1	1	0	1	1	S								7-bit data	P	STOP	STOP
0	—	1	0	0	S								8-bit data	MPB	STOP	
0	—	1	1	1	S								8-bit data	MPB	STOP	STOP
1	—	1	0	0	S								7-bit data	MPB	STOP	
1	—	1	1	1	S								7-bit data	MPB	STOP	STOP

Notes: SMR: Serial mode register

S: Start bit

STOP: Stop bit

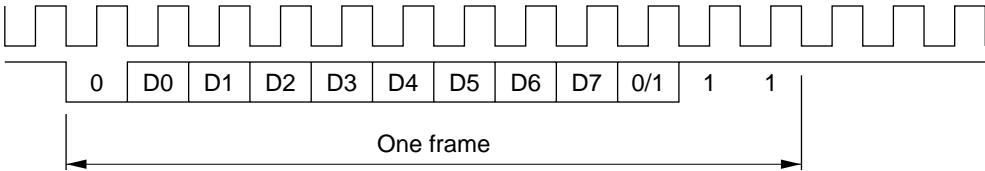
P: Parity bit

MPB: Multiprocessor bit

**2. Clock:** In 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. The selection is made by the C/A bit in the serial mode register (SMR) and the CKE1 and CKE0 bits in the serial control register (SCR). Refer to table 12-6.

If an external clock is input at the SCK pin, its frequency should be 16 times the desired bit 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 bit rate, and the clock pulse rises at the center of the transmit data bits. Figure 12-3 shows the phase relationship between the output clock and transmit data.



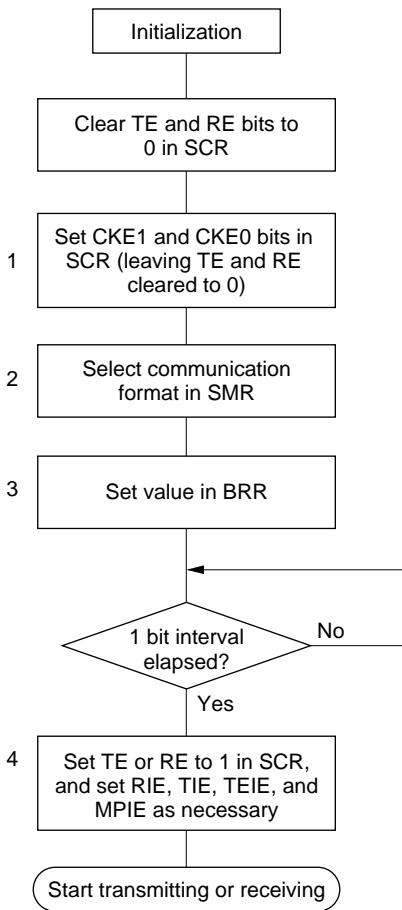
**Figure 12-3 Phase Relationship between Clock Output and Transmit Data (Asynchronous Mode)**

### 3. Transmitting and Receiving Data

- **SCI Initialization:** Before transmitting or receiving, software must clear the TE and RE bits to 0 in the serial control register (SCR), then initialize the SCI following the procedure in figure 12-4.

Note: When changing the communication mode or format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR), which retain their previous contents.

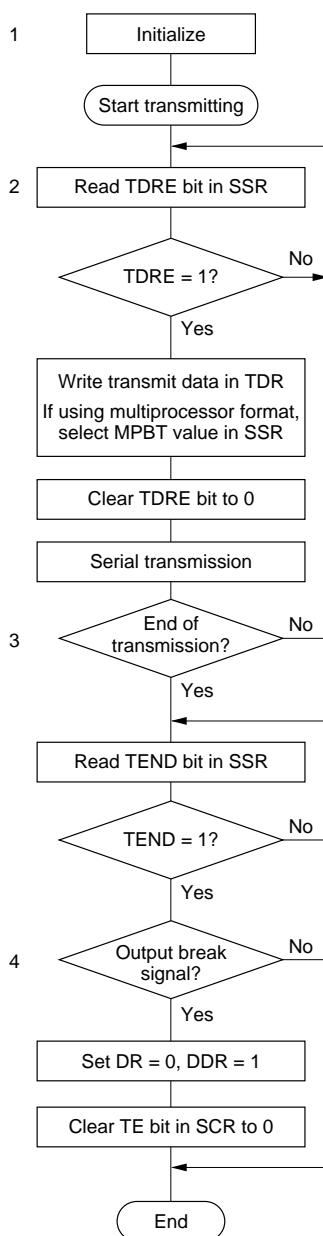
When an external clock is used, the clock should not be stopped during initialization or subsequent operation. SCI operation becomes unreliable if the clock is stopped.



1. Select interrupts and the clock source in the serial control register (SCR). Leave TE and RE cleared to 0. If clock output is selected, in asynchronous mode, clock output starts immediately after the setting is made in SCR.
2. Select the communication format in the serial mode register (SMR).
3. Write the value corresponding to the bit rate in the bit rate register (BRR). This step is not necessary when an external clock is used.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR). Setting TE or RE enables the SCI to use the TxD or RxD pin. Also set the RIE, TIE, TEIE, and MPIE bits as necessary to enable interrupts. The initial states are the mark transmit state, and the idle receive state (waiting for a start bit).

**Figure 12-4 Sample Flowchart for SCI Initialization**

- **Transmitting Serial Data:** Follow the procedure in figure 12-5 for transmitting serial data.



1. SCI initialization: the transmit data output function of the TxD pin is selected automatically.
2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. If a multiprocessor format is selected, after writing the transmit data write 0 or 1 in the multiprocessor bit transfer (MPBT) in SSR. Transition of the TDRE bit from 0 to 1 can be reported by an interrupt.
3. (a) To continue transmitting serial data: read the TDRE bit to check whether it is safe to write; if TDRE = 1, write data in TDR, then clear TDRE to 0.  
(b) To end serial transmission: end of transmission can be confirmed by checking transition of the TEND bit from 0 to 1. This can be reported by a TEI interrupt.
4. To output a break signal at the end of serial transmission: set the DDR bit to 1 and clear the DR bit to 0 (DDR and DR are I/O port registers), then clear TE to 0 in SCR.

**Figure 12-5 Sample Flowchart for Transmitting Serial Data**

In transmitting serial data, the SCI operates as follows.

1. The SCI monitors the TDRE bit in SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from TDR into the transmit shift register (TSR).

2. After loading the data from TDR into TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the TIE bit (TDR-empty interrupt enable) is set to 1 in SCR, the SCI requests a TXI interrupt (TDR-empty interrupt) at this time.

Serial transmit data are transmitted in the following order from the TxD pin:

(a) Start bit: One 0 bit is output.

(b) Transmit data: Seven or eight bits are output, LSB first.

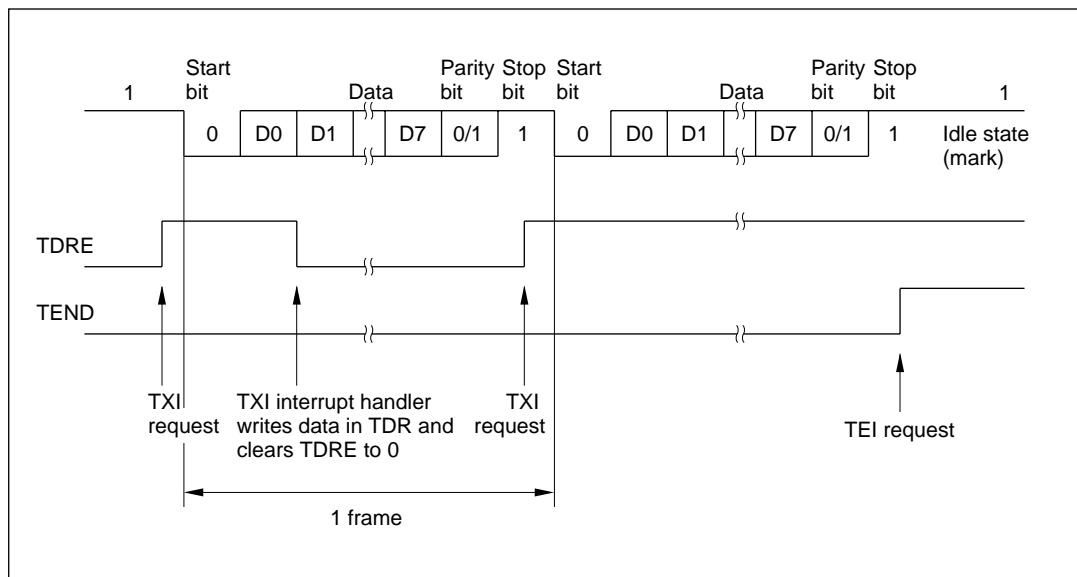
(c) Parity bit or multiprocessor bit: One parity bit (even or odd parity) or one multiprocessor bit is output. Formats in which neither a parity bit nor a multiprocessor bit is output can also be selected.

(d) Stop bit: One or two 1 bits (stop bits) are output.

(e) Mark state: Output of 1 bits continues until the start bit of the next transmit data.

3. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, after loading new data from TDR into TSR and transmitting the stop bit, the SCI begins serial transmission of the next frame. If TDRE is 1, after setting the TEND bit to 1 in SSR and transmitting the stop bit, the SCI continues 1-level output in the mark state, and if the TEIE bit (TSR-empty interrupt enable) in SCR is set to 1, the SCI generates a TEI interrupt request (TSR-empty interrupt).

Figure 12-6 shows an example of SCI transmit operation in asynchronous mode.



**Figure 12-6 Example of SCI Transmit Operation  
(8-Bit Data with Parity and One Stop Bit)**

- **Receiving Serial Data:** Follow the procedure in figure 12-7 for receiving serial data.

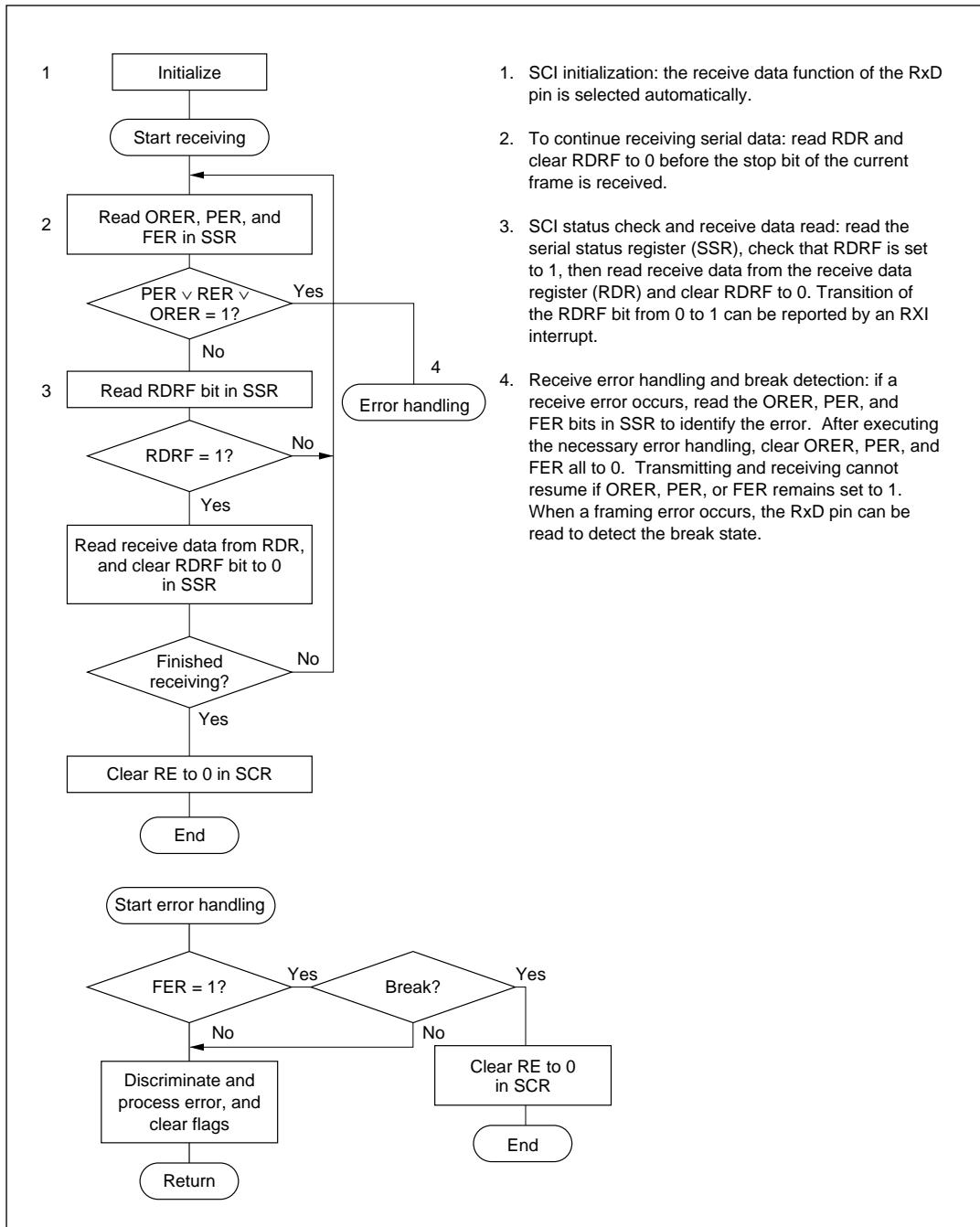


Figure 12-7 Sample Flowchart for Receiving Serial Data

In receiving, the SCI operates as follows.

1. The SCI monitors the receive data line and synchronizes internally when it detects a start bit.
2. Receive data is shifted into RSR in order from LSB to MSB.
3. The parity bit and stop bit are received.

After receiving these bits, the SCI makes the following checks:

- (a) Parity check: The number of 1s in the receive data must match the even or odd parity setting of the O/E bit in SMR.
- (b) Stop bit check: The stop bit value must be 1. If there are two stop bits, only the first stop bit is checked.
- (c) Status check: RDRF must be 0 so that receive data can be loaded from RSR into RDR.

If these checks all pass, the SCI sets RDRF to 1 and stores the received data in RDR. If one of the checks fails (receive error), the SCI operates as indicated in table 12-8.

Note: When a receive error flag is set, further receiving is disabled. The RDRF bit is not set to 1. Be sure to clear the error flags.

4. After setting RDRF to 1, if the RIE bit (receive-end interrupt enable) is set to 1 in SCR, the SCI requests an RXI (receive-end) interrupt. If one of the error flags (ORER, PER, or FER) is set to 1 and the RIE bit in SCR is also set to 1, the SCI requests an ERI (receive-error) interrupt.

**Table 12-8 Receive Error Conditions and SCI Operation**

Receive error	Abbreviation	Condition	Data Transfer
Overrun error	ORER	Receiving of next data ends while RDRF is still set to 1 in SSR	Receive data not loaded from RSR into RDR
Framing error	FER	Stop bit is 0	Receive data loaded from RSR into RDR
Parity error	PER	Parity of receive data differs from even/odd parity setting in SMR	Receive data loaded from RSR into RDR

Figure 12-8 shows an example of SCI receive operation in asynchronous mode.

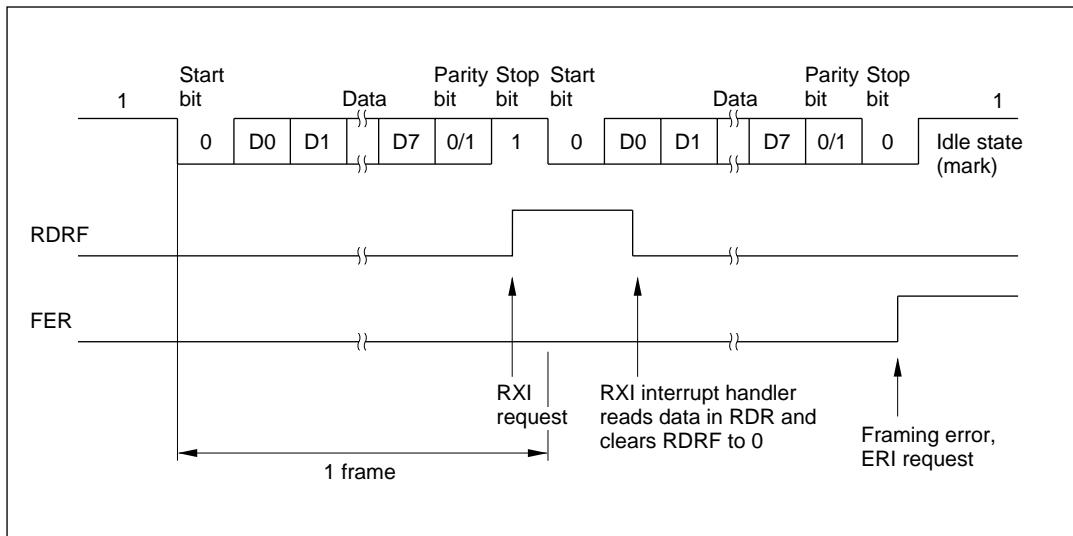


Figure 12-8 Example of SCI Receive Operation (8-Bit Data with Parity and One Stop Bit)

## 4. Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line. The processors communicate in asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by an ID.

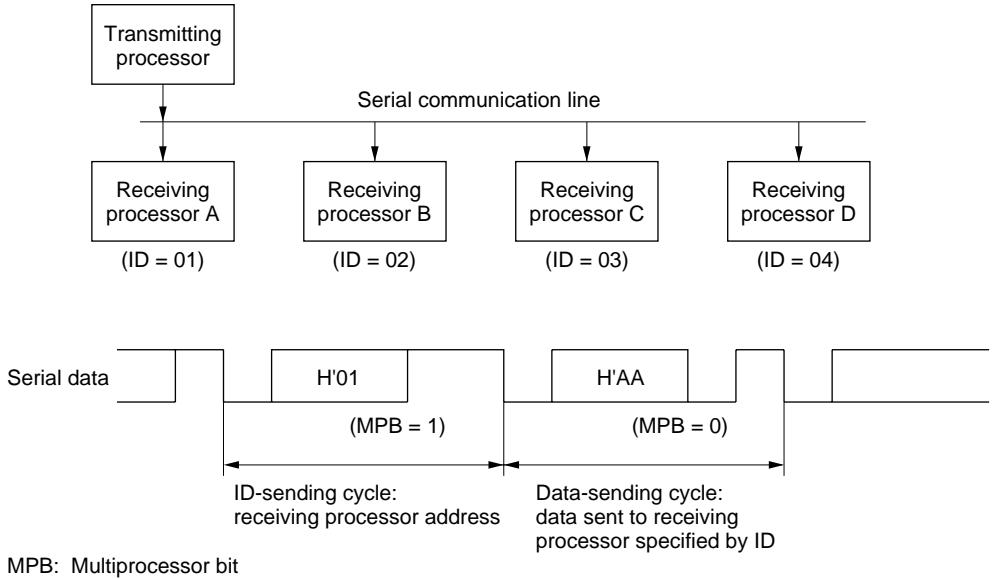
A serial communication cycle consists of two cycles: an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit distinguishes ID-sending cycles from data-sending cycles.

The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0.

Receiving processors skip incoming data until they receive data with the multiprocessor bit set to 1.

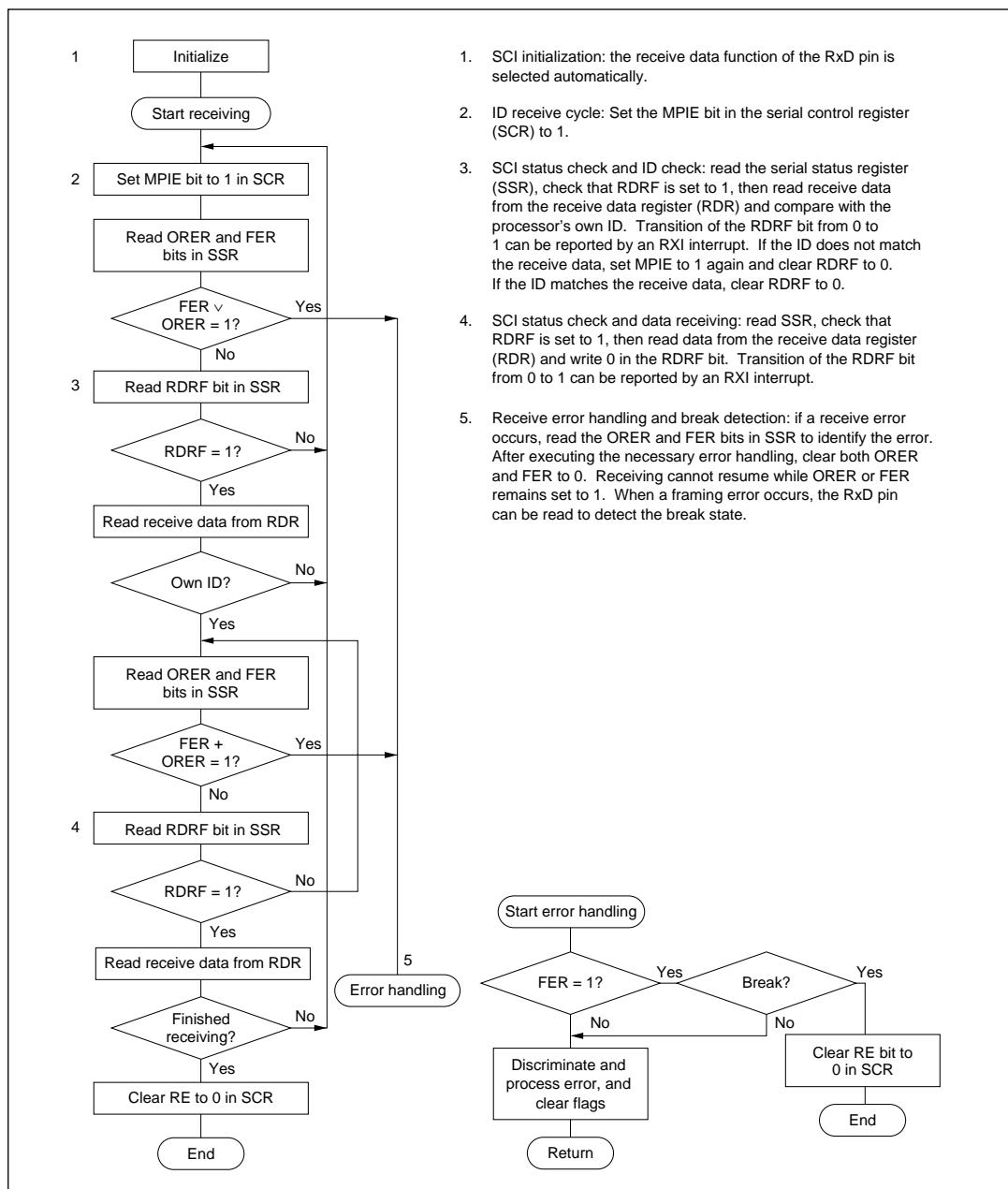
After receiving data with the multiprocessor bit set to 1, the receiving processor with an ID matching the received data continues to receive further incoming data. Multiple processors can send and receive data in this way.

Four formats are available. Parity-bit settings are ignored when a multiprocessor format is selected. For details see table 12-7.



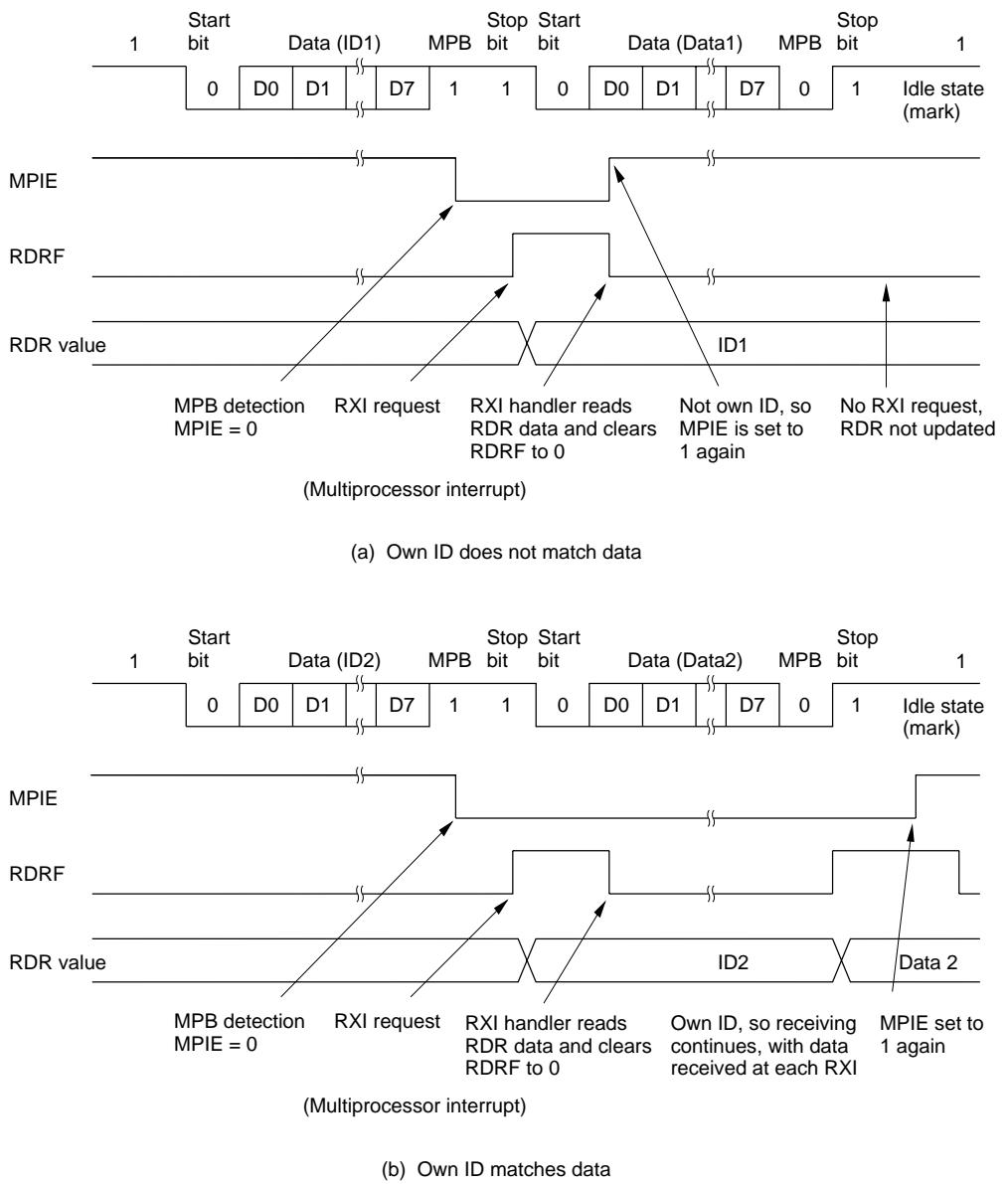
**Figure 12-9 Example of Communication among Processors using Multiprocessor Format (Sending Data H'AA to Receiving Processor A)**

- **Transmitting Multiprocessor Serial Data:** See figures 12-5 and 12-6.
- **Receiving Multiprocessor Serial Data:** Follow the procedure in figure 12-10 for receiving multiprocessor serial data.



**Figure 12-10 Sample Flowchart for Receiving Multiprocessor Serial Data**

Figure 12-11 shows an example of an SCI receive operation using a multiprocessor format (8-bit data with multiprocessor bit and one stop bit).



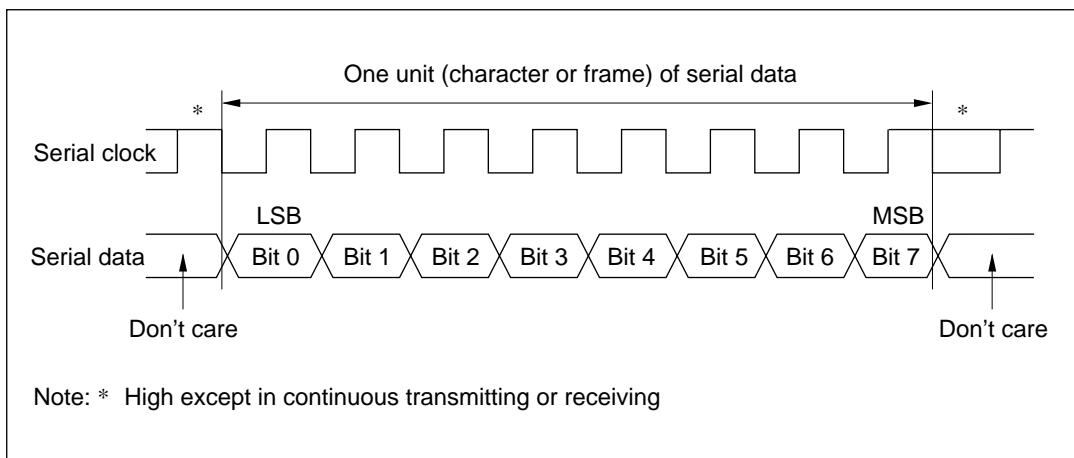
**Figure 12-11 Example of SCI Receive Operation  
(8-Bit Data with Multiprocessor Bit and One Stop Bit)**

### 12.3.3 Synchronous Mode

**(1) Overview:** In synchronous mode, the SCI transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

The SCI transmitter and receiver share the same clock but are otherwise independent, so full duplex communication is possible. The transmitter and receiver are also double buffered, so continuous transmitting or receiving is possible by reading or writing data while transmitting or receiving is in progress.

Figure 12-12 shows the general format in synchronous serial communication.



**Figure 12-12 Data Format in Synchronous Communication**

In synchronous serial communication, each data bit is sent on the communication line from one falling edge of the serial clock to the next. Data is received in synchronization with the rising edge of the serial clock.

In each character, the serial data bits are transmitted in order from LSB (first) to MSB (last). After output of the MSB, the communication line remains in the state of the MSB.

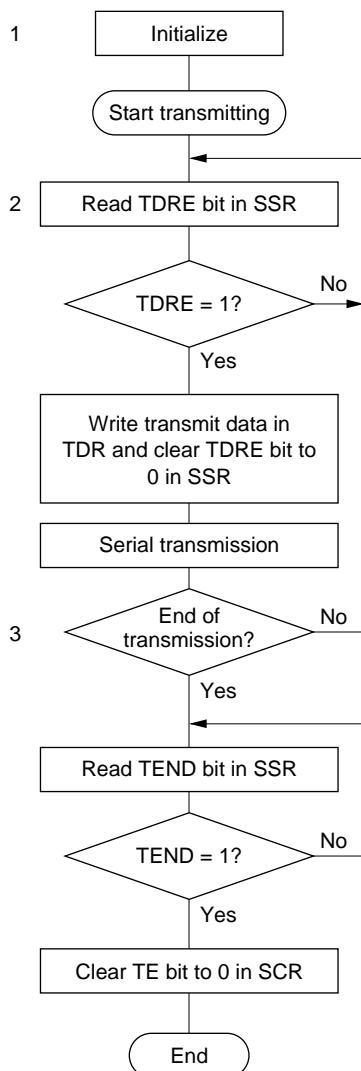
- **Communication Format:** The data length is fixed at eight bits. No parity bit or multiprocessor bit can be added.
- **Clock:** An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected by clearing or setting the CKE1 bit in the serial control register (SCR). See table 12-6.

When the SCI operates on an internal clock, it outputs the clock signal at the SCK pin. Eight clock pulses are output per transmitted or received character. When the SCI is not transmitting or receiving, the clock signal remains at the high level.

## (2) Transmitting and Receiving Data

- **SCI Initialization:** The SCI must be initialized in the same way as in asynchronous mode. See figure 12-4. When switching from asynchronous mode to synchronous mode, check that the ORER, FER, and PER bits are cleared to 0. Transmitting and receiving cannot begin if ORER, FER, or PER is set to 1.

- **Transmitting Serial Data:** Follow the procedure in figure 12-13 for transmitting serial data.



1. SCI initialization: the transmit data output function of the TxD pin is selected automatically.
2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. Transition of the TDRE bit from 0 to 1 can be reported by a TXI interrupt.
3. (a) To continue transmitting serial data: read the TDRE bit to check whether it is safe to write; if TDRE = 1, write data in TDR, then clear TDRE to 0.  
(b) To end serial transmission: end of transmission can be confirmed by checking transition of the TEND bit from 0 to 1. This can be reported by a TEI interrupt.

**Figure 12-13 Sample Flowchart for Serial Transmitting**

In transmitting serial data, the SCI operates as follows.

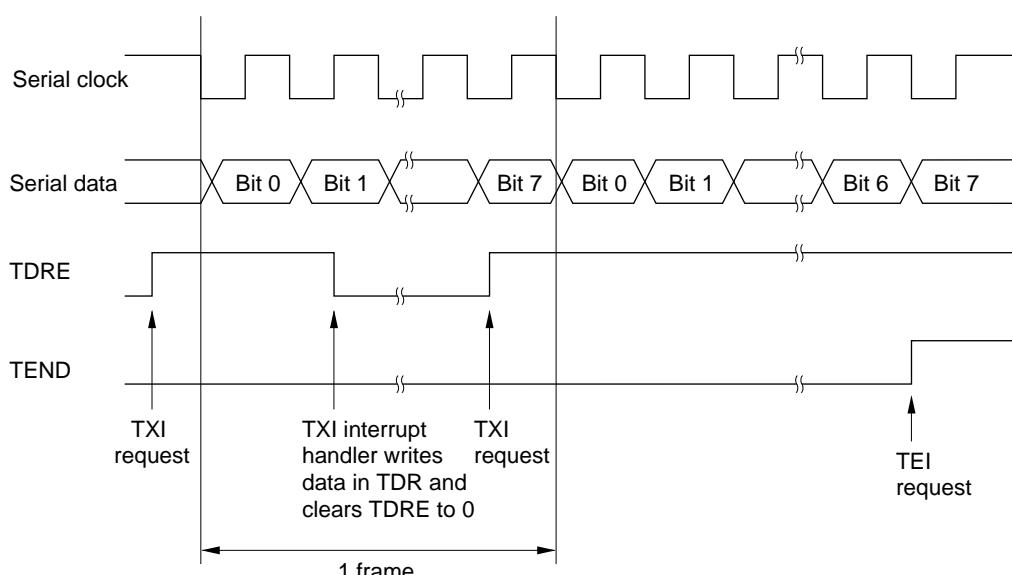
1. The SCI monitors the TDRE bit in SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from TDR into the transmit shift register (TSR).
2. After loading the data from TDR into TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the TIE bit (TDR-empty interrupt enable) in SCR is set to 1, the SCI requests a TXI interrupt (TDR-empty interrupt) at this time.

If clock output is selected the SCI outputs eight serial clock pulses, triggered by the clearing of the TDRE bit to 0. If an external clock source is selected, the SCI outputs data in synchronization with the input clock.

Data is output from the TxD pin in order from LSB (bit 0) to MSB (bit 7).

3. The SCI checks the TDRE bit when it outputs the MSB (bit 7). If TDRE is 0, the SCI loads data from TDR into TSR, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in SSR to 1, transmits the MSB, then holds the output in the MSB state. If the TEIE bit (transmit-end interrupt enable) in SCR is set to 1, a TEI interrupt (TSR-empty interrupt) is requested at this time.
4. After the end of serial transmission, the SCK pin is held at the high level.

Figure 12-14 shows an example of SCI transmit operation.



**Figure 12-14 Example of SCI Transmit Operation**

- **Receiving Serial Data:** Follow the procedure in figure 12-15 for receiving serial data. When switching from asynchronous mode to synchronous mode, be sure to check that PER and FER are cleared to 0. If PER or FER is set to 1 the RDRF bit will not be set and both transmitting and receiving will be disabled.

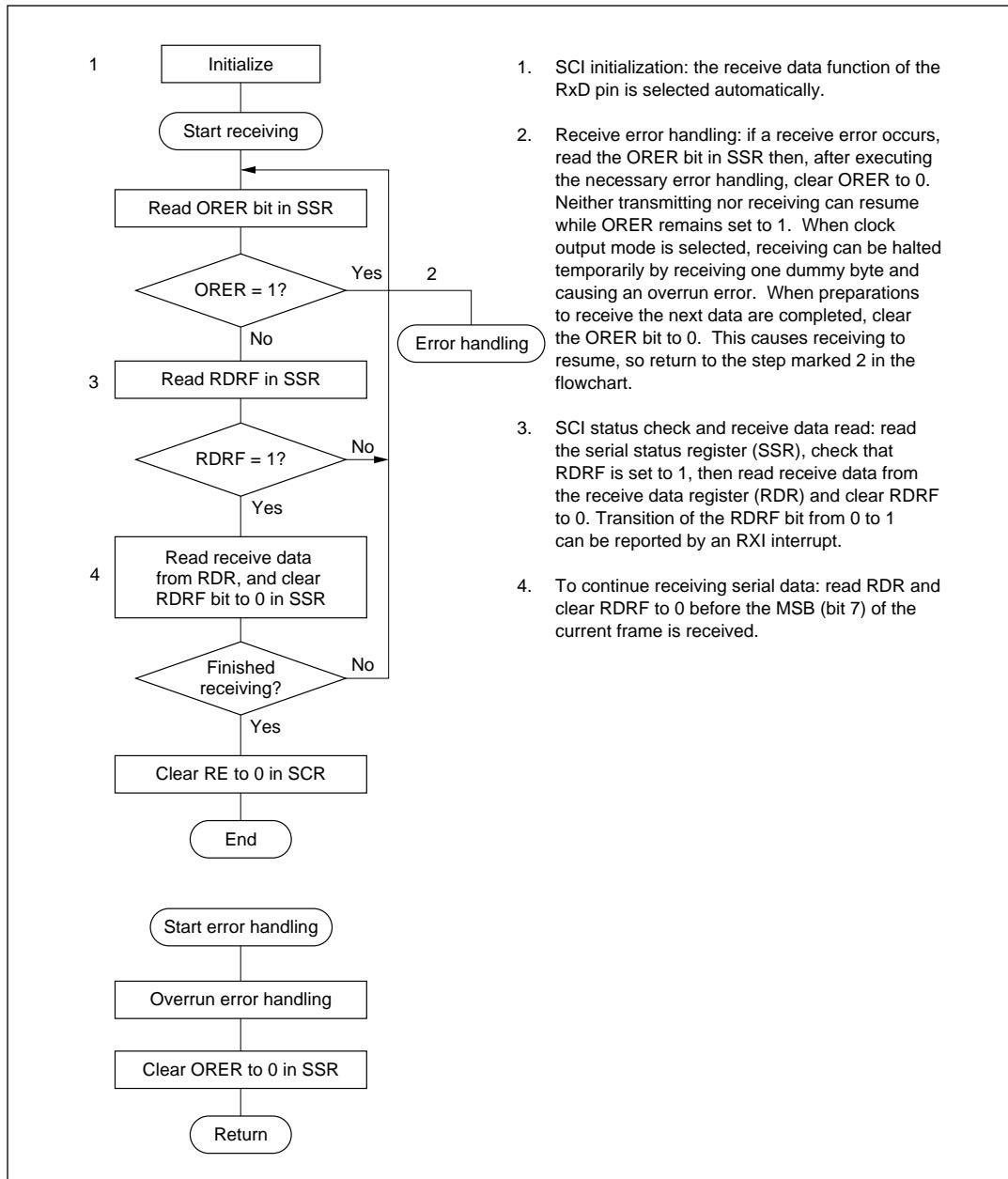


Figure 12-15 Sample Flowchart for Serial Receiving

In receiving, the SCI operates as follows.

1. If an external clock is selected, data is input in synchronization with the input clock. If clock output is selected, as soon as the RE bit is set to 1 the SCI begins outputting the serial clock and inputting data. If clock output is stopped because the ORER bit is set to 1, output of the serial clock and input of data resume as soon as the ORER bit is cleared to 0.
2. Receive data is shifted into RSR in order from LSB to MSB.

After receiving the data, the SCI checks that RDRF is 0 so that receive data can be loaded from RSR into RDR. If this check passes, the SCI sets RDRF to 1 and stores the received data in RDR. If the check does not pass (receive error), the SCI operates as indicated in table 12-8.

Note: Both transmitting and receiving are disabled while a receive error flag is set. The RDRF bit is not set to 1. Be sure to clear the error flag.

3. After setting RDRF to 1, if the RIE bit (receive-end interrupt enable) is set to 1 in SCR, the SCI requests an RXI (receive-end) interrupt. If the ORER bit is set to 1 and the RIE bit in SCR is set to 1, the SCI requests an ERI (receive-error) interrupt.

When clock output mode is selected, clock output stops when the RE bit is cleared to 0 or the ORER bit is set to 1. To prevent clock count errors, it is safest to receive one dummy byte and generate an overrun error.

Figure 12-16 shows an example of SCI receive operation.

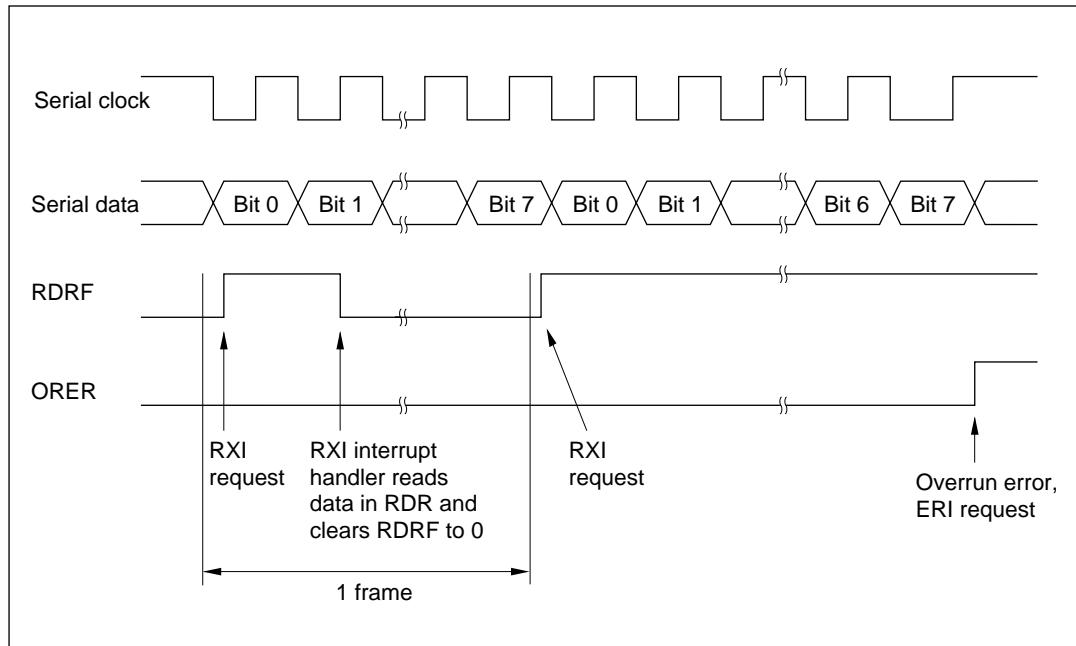
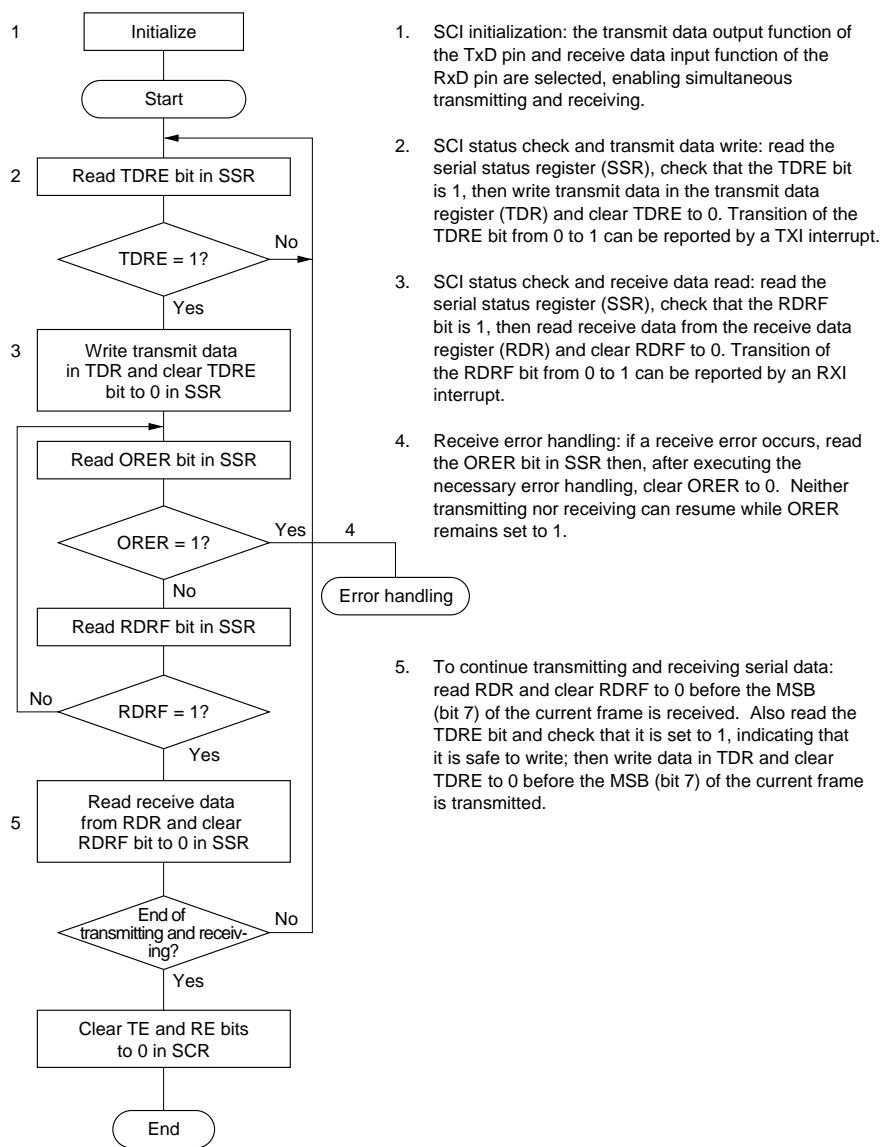


Figure 12-16 Example of SCI Receive Operation

- **Transmitting and Receiving Serial Data Simultaneously:** Follow the procedure in figure 12-17 for transmitting and receiving serial data simultaneously. If clock output mode is selected, output of the serial clock begins simultaneously with serial transmission.



**Figure 12-17 Sample Flowchart for Serial Transmitting and Receiving**

Note: In switching from transmitting or receiving to simultaneous transmitting and receiving, clear both TE and RE to 0, then set both TE and RE to 1.

## 12.4 Interrupts

The SCI can request four types of interrupts: ERI, RXI, TXI, and TEI. Table 12-9 indicates the source and priority of these interrupts. The interrupt sources can be enabled or disabled by the TIE, RIE, and TEIE bits in the SCR. Independent signals are sent to the interrupt controller for each interrupt source, except that the receive-error interrupt (ERI) is the logical OR of three sources: overrun error, framing error, and parity error.

The TXI interrupt indicates that the next transmit data can be written. The TEI interrupt indicates that the SCI has stopped transmitting data.

**Table 12-9 SCI Interrupt Sources**

Interrupt	Description	Priority
ERI	Receive-error interrupt (ORER, FER, or PER)	High
RXI	Receive-end interrupt (RDRF)	
TXI	TDR-empty interrupt (TDRE)	
TEI	TSR-empty interrupt (TEND)	Low

## 12.5 Application Notes

Application programmers should note the following features of the SCI.

**(1) TDR Write:** The TDRE bit in SSR is simply a flag that indicates that the TDR contents have been transferred to TSR. The TDR contents can be rewritten regardless of the TDRE value. If a new byte is written in TDR while the TDRE bit is 0, before the old TDR contents have been moved into TSR, the old byte will be lost. Software should check that the TDRE bit is set to 1 before writing to TDR.

**(2) Multiple Receive Errors:** Table 12-10 lists the values of flag bits in the SSR when multiple receive errors occur, and indicates whether the RSR contents are transferred to RDR.

**Table 12-10 SSR Bit States and Data Transfer when Multiple Receive Errors Occur**

Receive error	SSR Bits				RSR → RDR <sup>*2</sup>
	RDRF	ORER	FER	PER	
Overrun error	1 <sup>*1</sup>	1	0	0	No
Framing error	0	0	1	0	Yes
Parity error	0	0	0	1	Yes
Overrun and framing errors	1 <sup>*1</sup>	1	1	0	No
Overrun and parity errors	1 <sup>*1</sup>	1	0	1	No
Framing and parity errors	0	0	1	1	Yes
Overrun, framing, and parity errors	1 <sup>*1</sup>	1	1	1	No

Notes: 1. Set to 1 before the overrun error occurs.  
 2. Yes: The RSR contents are transferred to RDR.  
 No: The RSR contents are not transferred to RDR.

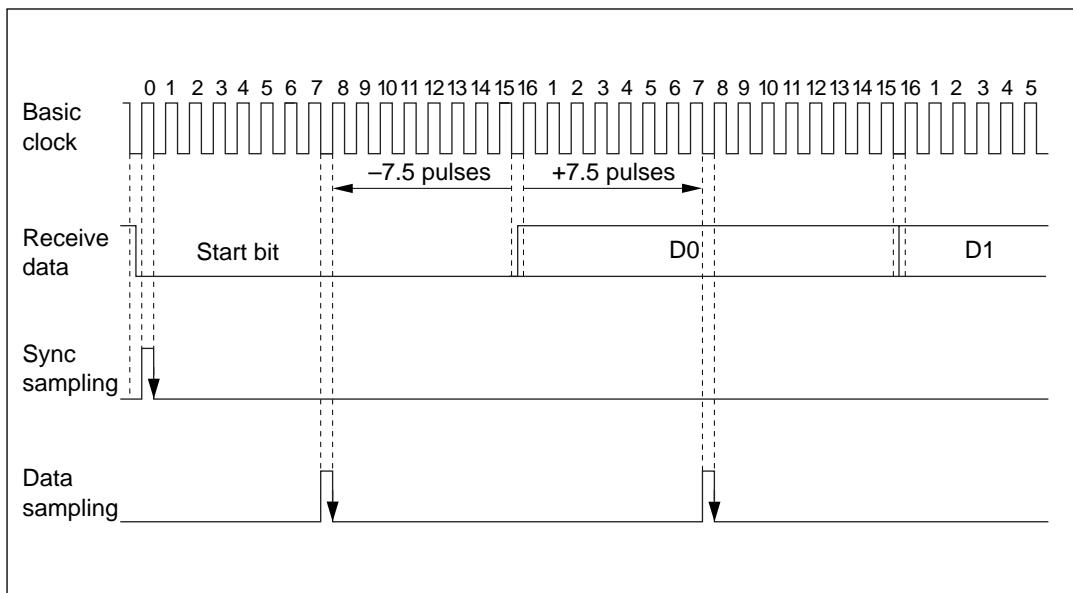
**(3) Line Break Detection:** When the RxD pin receives a continuous stream of 0's in asynchronous mode (line-break state), a framing error occurs because the SCI detects a 0 stop bit. The value H'00 is transferred from RSR to RDR. Software can detect the line-break state as a framing error accompanied by H'00 data in 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 12-18.

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 cycle 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 12-18 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 (N=16)

D: Duty factor of clock—ratio of high pulse width to low width (0.5 to 1.0)

L: Frame length (9 to 12)

F: Absolute clock frequency deviation

When D = 0.5 and F = 0

$$M = (0.5 - 1/2 \times 16) \times 100 [\%] = 46.875\% \quad (2)$$

# Section 13 I<sup>2</sup>C Bus Interface [Option]

An I<sup>2</sup>C bus interface is available as an option. Observe the following notes when using this option.

1. Please inform your Hitachi sales representative if you intend to use this option.
2. For mask-ROM versions, products that use this option have a “W” added to the product number.  
Examples: HD6433437WTF, HD6433434WF
3. The product number is identical for ZTAT and F-ZTAT versions. However, be sure to inform your Hitachi sales representative if you will be using this option.

## 13.1 Overview

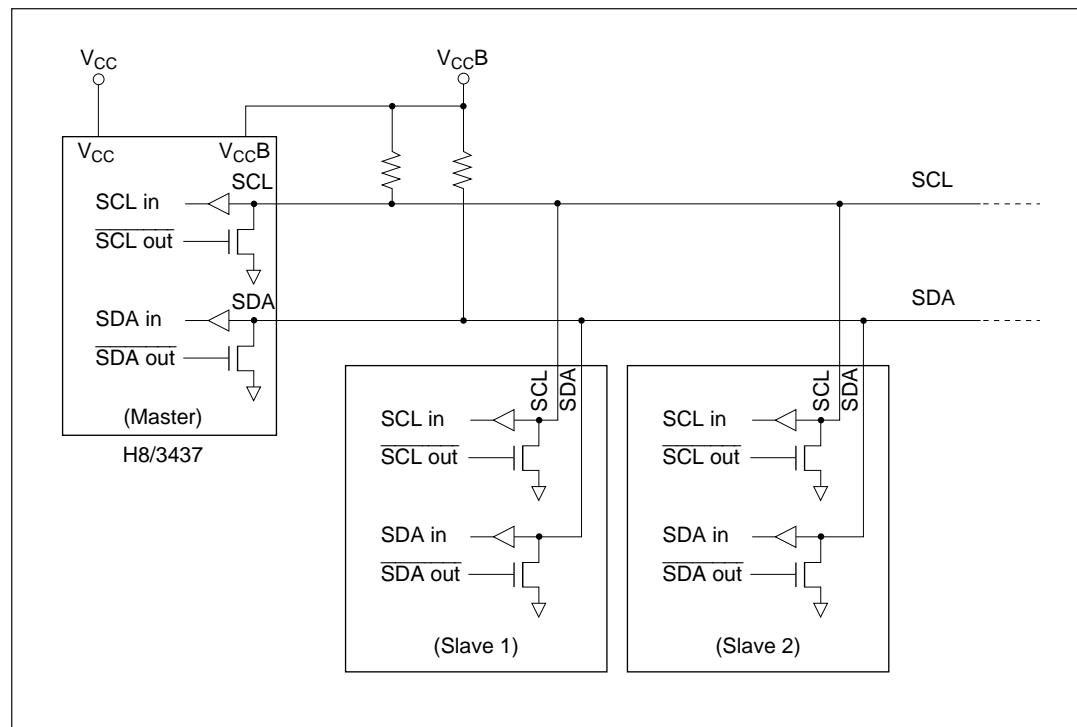
The I<sup>2</sup>C bus interface conforms to and provides a subset of the Philips I<sup>2</sup>C bus (inter-IC bus) interface functions. The register configuration that controls the I<sup>2</sup>C bus differs partly from the Philips configuration, however.

The I<sup>2</sup>C bus interface uses only one data line (SDA) and one clock line (SCL) to transfer data, so it can save board and connector space. Figure 13-1 shows typical I<sup>2</sup>C bus interface connections.

### 13.1.1 Features

- Conforms to Philips I<sup>2</sup>C bus interface
- Start and stop conditions generated automatically
- Selectable acknowledge output level when receiving
- Auto-loading of acknowledge bit when transmitting
- Selection of eight internal clocks (in master mode)
- Selection of acknowledgement mode, or serial mode without acknowledge bit
- Wait function: A wait can be inserted in acknowledgement mode by holding the SCL pin low after a data transfer, before acknowledgement of the transfer.
- Three interrupt sources
  - Data transfer end
  - In slave receive mode: slave address matched, or general call address received
  - In master transmit mode: bus arbitration lost
- Direct bus drive (pins SCL and SDA)

- In addition to pins SCL and SCA, four general port pins (PA<sub>4</sub> to PA<sub>7</sub>) can also drive the bus
- Pins P8<sub>6</sub>/SCK<sub>1</sub>/SCL, P9<sub>7</sub>/WAIT/SDA, and PA<sub>4</sub>/KEYIN<sub>12</sub> to PA<sub>7</sub>/KEYIN<sub>15</sub> (total of 6 pins) are all powered by bus power supply V<sub>CCB</sub>, separate from V<sub>CC</sub>. When the bus drive function is selected, all output is NMOS output.



**Figure 13-1 I<sup>2</sup>C Bus Interface Connection Example  
(When the H8/3437 is the Master Chip)**

### 13.1.2 Block Diagram

Figure 13-2 shows a block diagram of the I<sup>2</sup>C bus interface.

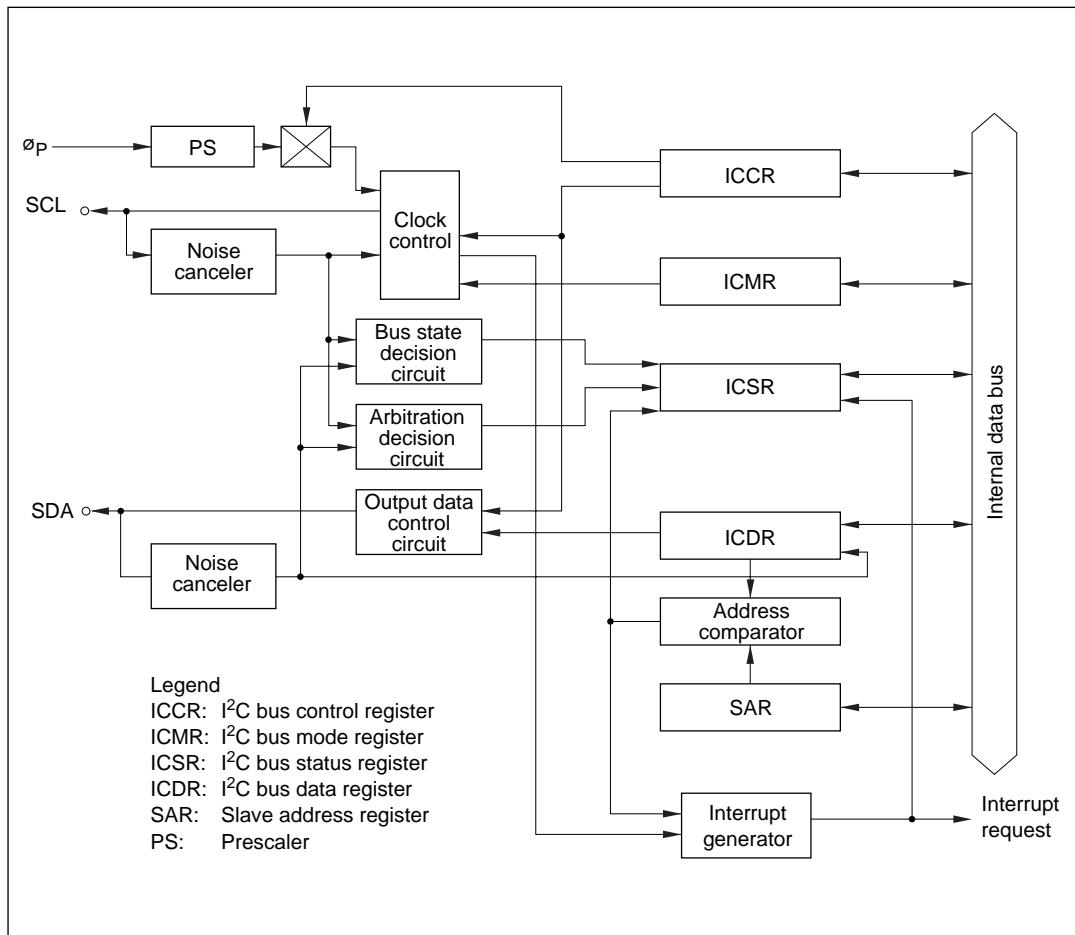


Figure 13-2 Block Diagram of I<sup>2</sup>C Bus Interface

### 13.1.3 Input/Output Pins

Table 13-1 summarizes the input/output pins used by the I<sup>2</sup>C bus interface.

**Table 13-1 Wait-State Controller Pins**

Name	Abbreviation	I/O	Function
Serial clock	SCL	Input/output	Serial clock input/output
Serial data	SDA	Input/output	Serial data input/output

### 13.1.4 Register Configuration

Table 13-2 summarizes the registers of the I<sup>2</sup>C bus interface.

**Table 13-2 Register Configuration**

Name	Abbreviation	R/W	Initial Value	Address* <sup>2</sup>
I <sup>2</sup> C bus control register	ICCR	R/W	H'00	H'FFD8
I <sup>2</sup> C bus status register	ICSR	R/W	H'30	H'FFD9
I <sup>2</sup> C bus data register	ICDR	R/W	—	H'FFDE
I <sup>2</sup> C bus mode register	ICMR	R/W	H'38	H'FFDF* <sup>1</sup>
Slave address register	SAR	R/W	H'00	H'FFDF* <sup>1</sup>
Serial timer control register	STCR	R/W	H'00	H'FFC3

Notes: 1. The register that can be written or read depends on the ICE bit in the I<sup>2</sup>C bus control register. The slave address register can be accessed when ICE = 0. The I<sup>2</sup>C bus mode register can be accessed when ICE = 1.  
2. The addresses assigned to the I<sup>2</sup>C bus interface registers are also assigned to other registers. The accessible registers are selected with bit IICE in the serial/timer control register (STCR).

## 13.2 Register Descriptions

### 13.2.1 I<sup>2</sup>C Bus Data Register (ICDR)

Bit	7	6	5	4	3	2	1	0
	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0
Initial value	—	—	—	—	—	—	—	—
Read/Write	R/W							

ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. Transmitting is started by writing data in ICDR. Receiving is started by reading data from ICDR.

ICDR is also used as a shift register, so it must not be written or read until data has been completely transmitted or received. Read or write access while data is being transmitted or received may result in incorrect data.

The ICDR value following a reset is undetermined.

### 13.2.2 Slave Address Register (SAR)

Bit	7	6	5	4	3	2	1	0
	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W						

SAR is an 8-bit readable/writable register that stores the slave address and selects the communication format. When the chip is in slave mode (and the addressing format is selected), if the upper 7 bits of SAR match the upper 7 bits of the first byte received after a start condition, the chip operates as the slave device specified by the master device. SAR is assigned to the same address as ICMR. SAR can be written and read only when the ICE bit is cleared to 0 in ICCR.

SAR is initialized to H'00 by a reset.

**Bits 7 to 1—Slave Address (SVA6 to SVA0):** Set a unique address in bits SVA6 to SVA0, differing from the addresses of other slave devices connected to the I<sup>2</sup>C bus.

**Bit 0—Format Select (FS):** Selects whether to use the addressing format or non-addressing format in slave mode. The addressing format is used to recognize slave addresses.

**Bit 0**

**FS      Description**

0	Addressing format, slave addresses recognized							(Initial value)
1	Non-addressing format							

### 13.2.3 I<sup>2</sup>C Bus Mode Register (ICMR)

Bit	7	6	5	4	3	2	1	0
	MLS	WAIT	—	—	—	BC2	BC1	BC0
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

ICMR is an 8-bit readable/writable register that selects whether the MSB or LSB is transferred first, performs wait control, and selects the transfer bit count. ICMR is assigned to the same address as SAR. ICMR can be written and read only when the ICE bit is set to 1 in ICCR.

ICMR is initialized to H'38 by a reset.

**Bit 7—MSB-First/LSB-First Select (MLS):** Selects whether data is transferred MSB-first or LSB-first.

**Bit 7**

**MLS      Description**

0	MSB-first							(Initial value)
1	LSB-first							

**Bit 6—Wait Insertion Bit (WAIT):** Selects whether to insert a wait between the transfer of data and the acknowledge bit, in acknowledgement mode. When WAIT is set to 1, after the fall of the clock for the final data bit, a wait state begins (with SCL staying at the low level). When bit IRIC is cleared in ICSR, the wait ends and the acknowledge bit is transferred. If WAIT is cleared to 0, data and acknowledge bits are transferred consecutively with no wait inserted.

**Bit 6**

**WAIT      Description**

0	Data and acknowledge transferred consecutively							(Initial value)
1	Wait inserted between data and acknowledge							

**Bits 5 to 3—Reserved:** These bits cannot be modified and are always read as 1.

**Bits 2 to 0—Bit Counter (BC2 to BC0):** BC2 to BC0 specify the number of bits to be transferred next. When the ACK bit is cleared to 0 in ICCR (acknowledgement mode), the data is transferred with one additional acknowledge bit. BC2 to BC0 settings should be made during an interval between transfer frames. If BC2 to BC0 are set to a value other than 000, the setting should be made while the SCL line is low.

The bit counter is initialized to 000 by a reset and when a start condition is detected. The value returns to 000 at the end of a data transfer, including the acknowledge.

Bit 2 BC2	Bit 1 BC1	Bit 0 BC0	Bits/Frame	
			Serial Mode	Acknowledgement Mode
0	0	0	8	9
		1	1	2
	1	0	2	3
		1	3	4
1	0	0	4	5
		1	5	6
	1	0	6	7
		1	7	8

### 13.2.4 I<sup>2</sup>C Bus Control Register (ICCR)

Bit	7	6	5	4	3	2	1	0
	ICE	IEIC	MST	TRS	ACK	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

ICCR is an 8-bit readable/writable register that enables or disables the I<sup>2</sup>C bus interface, enables or disables interrupts, and selects master or slave mode, transmit or receive, acknowledgement or serial mode, and the clock frequency.

ICCR is initialized to H'00 by a reset.

**Bit 7—I<sup>2</sup>C Bus Interface Enable (ICE):** Selects whether or not to use the I<sup>2</sup>C bus interface. When ICE is set to 1, the SCL and SDA signals are assigned to input/output pins and transfer operations are enabled. When ICE is cleared to 0, the interface module is disabled.

The SAR register can be accessed when ICE is 0. The ICMR register can be accessed when ICE is 1.

#### Bit 7

##### ICE      Description

0	Interface module disabled, with SCL and SDA signals in high-impedance state (Initial value)
1	Interface module enabled for transfer operations (pins SCL and SDA are driving the bus*)

Note: \* Pin SDA is multiplexed with the  $\overline{\text{WAIT}}$  input pin. In expanded mode,  $\overline{\text{WAIT}}$  input has priority for this pin.

**Bit 6—I<sup>2</sup>C Bus Interface Interrupt Enable (IEIC):** Enables or disables interrupts from the I<sup>2</sup>C bus interface to the CPU.

#### Bit 6

##### IEIC      Description

0	Interrupts disabled (Initial value)
1	Interrupts enabled

## **Bit 5—Master/Slave Select (MST)**

## **Bit 4—Transmit/Receive Select (TRS)**

MST selects whether the I<sup>2</sup>C bus interface operates in master mode or slave mode.

TRS selects whether the I<sup>2</sup>C bus interface operates in transmit mode or receive mode.

In master mode, when arbitration is lost, MST and TRS are both reset by hardware, causing a transition to slave receive mode. In slave receive mode with the addressing format (FS = 0), hardware automatically selects transmit or receive mode according to the R/W bit in the first byte after a start condition.

MST and TRS select the operating mode as follows.

<b>Bit 5</b> <b>MST</b>	<b>Bit 4</b> <b>TRS</b>	<b>Operating Mode</b>	
0	0	Slave receive mode	(Initial value)
	1	Slave transmit mode	
1	0	Master receive mode	
	1	Master transmit mode	

**Bit 3—Acknowledgement Mode Select (ACK):** Selects acknowledgement mode or serial mode. In acknowledgement mode (ACK = 0), data is transferred in frames consisting of the number of data bits selected by BC2 to BC0 in ICMR, plus an extra acknowledge bit. In serial mode (ACK = 1), the number of data bits selected by BC2 to BC0 in ICMR is transferred as one frame.

<b>Bit 3</b> <b>ACK</b>	<b>Description</b>	
0	Acknowledgement mode	(Initial value)
1	Serial mode	

**Bits 2 to 0—Serial Clock Select (CKS2 to CKS0):** These bits, together with the ICCX bit in the STCR register, select the serial clock frequency in master mode. They should be set according to the required transfer rate.

(STCR) IICX	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Clock	Transfer Rate*				
					$\emptyset_P = 4$ MHz	$\emptyset_P = 5$ MHz	$\emptyset_P = 8$ MHz	$\emptyset_P = 10$ MHz	$\emptyset_P = 16$ MHz
0	0	0	0	$\emptyset_P/28$	143 kHz	179 kHz	286 kHz	357 kHz	571 kHz
	0	0	1	$\emptyset_P/40$	100 kHz	125 kHz	200 kHz	250 kHz	400 kHz
	0	1	0	$\emptyset_P/48$	83.3 kHz	104 kHz	167 kHz	208 kHz	333 kHz
	0	1	1	$\emptyset_P/64$	62.5 kHz	78.1 kHz	125 kHz	156 kHz	250 kHz
	1	0	0	$\emptyset_P/80$	50.0 kHz	62.5 kHz	100 kHz	125 kHz	200 kHz
	1	0	1	$\emptyset_P/100$	40.0 kHz	50.0 kHz	80.0 kHz	100 kHz	160 kHz
	1	1	0	$\emptyset_P/112$	35.7 kHz	44.6 kHz	71.4 kHz	89.3 kHz	143 kHz
	1	1	1	$\emptyset_P/128$	31.3 kHz	39.1 kHz	62.5 kHz	78.1 kHz	125 kHz
1	0	0	0	$\emptyset_P/56$	71.4 kHz	89.3 kHz	143 kHz	179 kHz	286 kHz
	0	0	1	$\emptyset_P/80$	50.0 kHz	62.5 kHz	100 kHz	125 kHz	200 kHz
	0	1	0	$\emptyset_P/96$	41.7 kHz	52.1 kHz	83.3 kHz	104 kHz	167 kHz
	0	1	1	$\emptyset_P/128$	31.3 kHz	39.1 kHz	62.5 kHz	78.1 kHz	125 kHz
	1	0	0	$\emptyset_P/160$	25.0 kHz	31.3 kHz	50.0 kHz	62.5 kHz	100 kHz
	1	0	1	$\emptyset_P/200$	20.0 kHz	25.0 kHz	40.0 kHz	50.0 kHz	80.0 kHz
	1	1	0	$\emptyset_P/224$	17.9 kHz	22.3 kHz	35.7 kHz	44.6 kHz	71.4 kHz
	1	1	1	$\emptyset_P/256$	15.6 kHz	19.5 kHz	31.3 kHz	39.1 kHz	62.5 kHz

Note: \*  $\emptyset_P = \emptyset$ .

### 13.2.5 I<sup>2</sup>C Bus Status Register (ICSR)

Bit	7	6	5	4	3	2	1	0
	BBSY	IRIC	SCP	—	AL	AAS	ADZ	ACKB
Initial value	0	0	1	1	0	0	0	0
Read/Write	R/W	R/(W)*	W	—	R/(W)*	R/(W)*	R/(W)*	R/W

Note: \* Only 0 can be written, to clear the flag.

ICSR is an 8-bit readable/writable register with flags that indicate the status of the I<sup>2</sup>C bus interface. It is also used for issuing start and stop conditions, and recognizing and controlling acknowledge data.

ICSR is initialized to H'30 by a reset.

**Bit 7—Bus Busy (BBSY):** This bit can be read to check whether the I<sup>2</sup>C bus (SCL and SDA) is busy or free. In master mode this bit is also used in issuing start and stop conditions.

A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0.

To issue a start condition, use a MOV instruction to write 1 in BBSY and 0 in SCP. A retransmit start condition is issued in the same way. To issue a stop condition, use a MOV instruction to write 0 in BBSY and 0 in SCP. It is not possible to write to BBSY in slave mode.

Bit 7	
BBSY	Description
0	Bus is free This bit is cleared when a stop condition is detected. (Initial value)
1	Bus is busy This bit is set when a start condition is detected.

**Bit 6—I<sup>2</sup>C Bus Interface Interrupt Request Flag (IRIC):** Indicates that the I<sup>2</sup>C bus interface has issued an interrupt request to the CPU. IRIC is set to 1 at the end of a data transfer, when a slave address or general call address is detected in slave receive mode, and when bus arbitration is lost in master transmit mode. IRIC is set at different timings depending on the ACK bit in ICCR and WAIT bit in ICMR. See the item on IRIC Set Timing and SCL Control in section 13.3.6

IRIC is cleared by reading IRIC after it has been set to 1, then writing 0 in IRIC.

**Bit 6**

IRIC	Description
0	Waiting for transfer, or transfer in progress (Initial value) To clear this bit, the CPU must read IRIC when IRIC = 1, then write 0 in IRIC
1	Interrupt requested This bit is set to 1 at the following times:  Master mode <ul style="list-style-type: none"><li>• End of data transfer</li><li>• When bus arbitration is lost</li></ul> Slave mode (when FS = 0) <ul style="list-style-type: none"><li>• When the slave address is matched, and whenever a data transfer ends at the timing of a retransmit start condition after address matching or a stop condition is detected</li><li>• When a general call address is detected, and whenever a data transfer ends at the timing of a retransmit start condition after address detection or a stop condition is detected</li></ul> Slave mode (when FS = 1) <ul style="list-style-type: none"><li>• End of data transfer</li></ul>

**Bit 5—Start Condition/Stop Condition Prohibit (SCP):** Controls the issuing of start and stop conditions in master mode. To issue a start condition, write 1 in BBSY and 0 in SCP. A start condition for retransmit is issued in the same way. To issue a stop condition, write 0 in BBSY and 0 in SCP. This bit always reads 1. Written data is not stored.

**Bit 5**

SCP	Description
0	Writing 0 issues a start or stop condition, in combination with BBSY
1	Reading always results in 1 (Initial value) Writing is ignored

**Bit 4—Reserved:** This bit cannot be modified and is always read as 1.

**Bit 3—Arbitration Lost (AL):** This flag indicates that arbitration was lost in master mode. The I<sup>2</sup>C bus interface monitors the bus. When two or more master devices attempt to seize the bus at nearly the same time, if the I<sup>2</sup>C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master. At the same time, it sets the IRIC bit in ICSR to generate an interrupt request.

AL is cleared by reading AL after it has been set to 1, then writing 0 in AL. In addition, AL is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

**Bit 3**

AL	Description
0	<p>Bus arbitration won <span style="float: right;">(Initial value)</span></p> <p>This bit is cleared to 0 at the following times:</p> <ul style="list-style-type: none"> <li>• When ICDR data is written (transmit mode) or read (receive mode)</li> <li>• When AL is read while AL = 1, then 0 is written in AL</li> </ul>
1	<p>Arbitration lost</p> <p>This bit is set to 1 at the following times:</p> <ul style="list-style-type: none"> <li>• If the internal SDA signal and bus line disagree at the rise of SCL in master transmit mode</li> <li>• If the internal SCL is high at the fall of SCL in master transmit mode</li> </ul>

**Bit 2—Slave Address Recognition Flag (AAS):** When the addressing format is selected (FS = 0) in slave receive mode, this flag is set to 1 if the first byte following a start condition matches bits SVA6 to SVA0 in SAR, or if the general call address (H'00) is detected.

AAS is cleared by reading AAS after it has been set to 1, then writing 0 in AAS. In addition, AAS is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

**Bit 2**

AAS	Description
0	<p>Slave address or general call address not recognized <span style="float: right;">(Initial value)</span></p> <p>This bit is cleared to 0 at the following times:</p> <ul style="list-style-type: none"> <li>• When ICDR data is written (transmit mode) or read (receive mode)</li> <li>• When AAS is read while AAS = 1, then 0 is written in AAS</li> </ul>
1	<p>Slave address or general call address recognized</p> <p>This bit is set to 1 at the following times:</p> <ul style="list-style-type: none"> <li>• When the slave address or general call address is detected in slave receive mode</li> </ul>

**Bit 1—General Call Address Recognition Flag (ADZ):** When the addressing format is selected (FS = 0) in slave receive mode, this flag is set to 1 if the first byte following a start condition is the general call address (H'00).

ADZ is cleared by reading ADZ after it has been set to 1, then writing 0 in ADZ. In addition, ADZ is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

**Bit 1**

ADZ	Description	
0	General call address not recognized This bit is cleared to 0 at the following times: <ul style="list-style-type: none"><li>• When ICDR data is written (transmit mode) or read (receive mode)</li><li>• When ADZ is read while ADZ = 1, then 0 is written in ADZ</li></ul>	(Initial value)
1	General call address recognized This bit is set to 1 when the general call address is detected in slave receive mode	

**Bit 0—Acknowledge Bit (ACKB):** Stores acknowledge data in acknowledgement mode. In transmit mode, after the receiving device receives data, it returns acknowledge data, and this data is loaded into ACKB. In receive mode, after data has been received, the acknowledge data set in this bit is sent to the transmitting device.

When this bit is read, if TRS = 1, the value loaded from the bus line is read. If TRS = 0, the value set by internal software is read.

**Bit 0**

ACKB	Description	
0	Receive mode: 0 is output at acknowledge output timing Transmit mode: indicates that the receiving device has acknowledged the data	(Initial value)
1	Receive mode: 1 is output at acknowledge output timing Transmit mode: indicates that the receiving device has not acknowledged the data	

### 13.2.6 Serial/Timer Control Register (STCR)

Bit	7	6	5	4	3	2	1	0
	IICS	IICD	IICX	IICE	STAC	MPE	ICKS1	ICKS0
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

STCR is an 8-bit readable/writable register that controls the SCI operating mode and selects the TCNT clock source in the 8-bit timers. STCR is initialized to H'00 by a reset.

**Bit 7—I<sup>2</sup>C Extra Buffer Select (IICS):** Makes bits 7 to 4 of port A into output buffers similar to SCL and SDA. Used when an I<sup>2</sup>C bus interface is implemented by software alone.

Bit 7		
IICS	Description	
0	PA <sub>7</sub> to PA <sub>4</sub> are normal input/output pins	(Initial value)
1	PA <sub>7</sub> to PA <sub>4</sub> are input/output pins that can drive the bus	

**Bit 6—I<sup>2</sup>C Extra Buffer Reserve (IICD):** This bit is reserved, but it can be written and read. Its initial value is 0.

**Bit 5—I<sup>2</sup>C Transfer Rate Select (IICX):** This bit, in combination with bits CKS2 to CKS0 in ICCR, selects the transfer rate in master mode. For details regarding the transfer rate, refer to section 13.2.4, I<sup>2</sup>C Bus Control Register (ICCR).

**Bit 4—I<sup>2</sup>C Master Enable (IICE):** Controls CPU access to the data and control registers (ICCR, ICSR, ICDR, ICMR/SAR) of the I<sup>2</sup>C bus interface.

Bit 4		
IICE	Description	
0	CPU access to I <sup>2</sup> C bus interface data and control registers is disabled	(Initial value)
1	CPU access to I <sup>2</sup> C bus interface data and control registers is enabled	

**Bit 3—Slave Input Switch (STAC):** Switches host interface input pins. For details, see section 14, Host Interface.

**Bit 2—Multiprocessor Enable (MPE):** Enables or disables the multiprocessor communication function on channels SCI0 and SCI1. For details, see section 12, Serial Communication Interface.

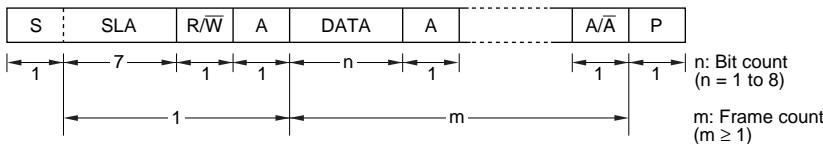
**Bits 1 and 0—Internal Clock Source Select 1 and 0 (ICKS1, ICSK0):** These bits select the clock input to the timer counters (TCNT) in the 8-bit timers. For details, see section 9, 8-Bit Timers.

## 13.3 Operation

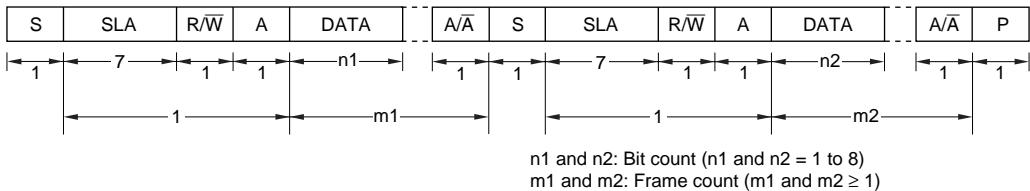
### 13.3.1 I<sup>2</sup>C Bus Data Format

The I<sup>2</sup>C bus interface has three data formats: two addressing formats, shown as (a) and (b) in figure 13-3, and a non-addressing format, shown as (c) in figure 13-4. The first byte following a start condition always consists of 8 bits. Figure 13-5 shows the I<sup>2</sup>C bus timing.

(a) Addressing format (FS = 0)

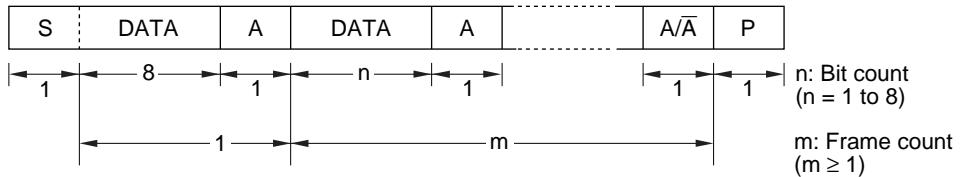


(b) Addressing format (retransmit start condition, FS = 0)



**Figure 13-3 I<sup>2</sup>C Bus Data Formats (Addressing Formats)**

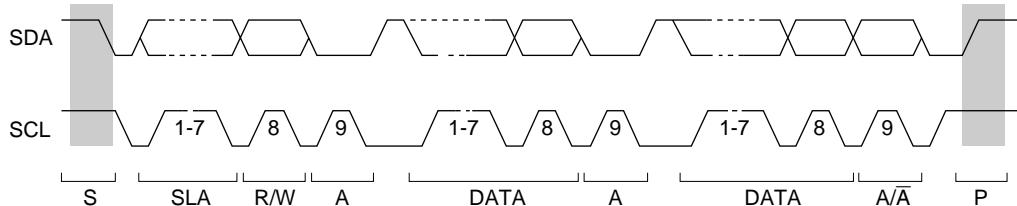
(c) Non-addressing format ( $FS = 1$ )



**Figure 13-4 I<sup>2</sup>C Bus Data Format (Non-Addressing Format)**

Legend

- S: Start condition. The master device drives SDA from high to low while SCL is high.
- SLA: Slave address, by which the master device selects a slave device.
- R/W: Indicates the direction of data transfer: from the slave device to the master device when R/W is 1, or from the master device to the slave device when R/W is 0.
- A: Acknowledge. The receiving device (the slave in master transmit mode, or the master in master receive mode) drives SDA low to acknowledge a transfer. If transfers need not be acknowledged, set the ACK bit to 1 in ICCR to keep the interface from generating the acknowledge signal and its clock pulse.
- DATA: Transferred data. The bit length is set by bits BC2 to BC0 in ICMR. The MSB-first or LSB-first format is selected by bit MLS in ICMR.
- P: Stop condition. The master device drives SDA from low to high while SCL is high.



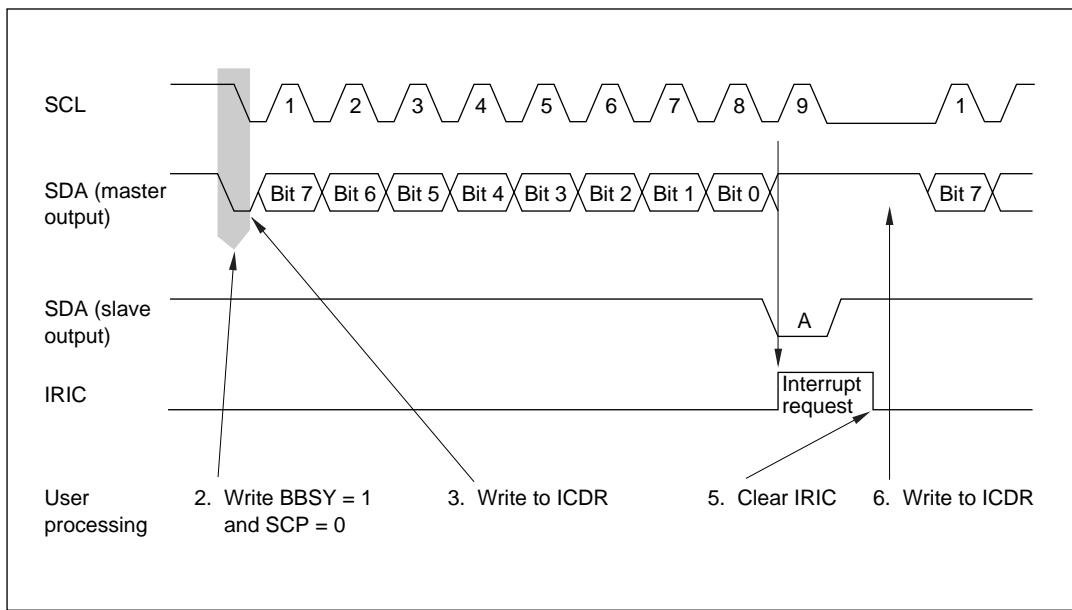
**Figure 13-5 I<sup>2</sup>C Bus Timing**

### 13.3.2 Master Transmit Operation

In master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The transmit procedure and operations in master transmit mode are described below.

1. Set bits MLS and WAIT in ICMR and bits ACK and CKS2 to CKS0 in ICCR according to the operating mode. Set bit ICE in ICCR to 1.
2. Read BBSY in ICSR, check that the bus is free, then set MST and TRS to 1 in ICCR to select master transmit mode. After that, write 1 in BBSY and 0 in SCP. This generates a start condition by causing a high-to-low transition of SDA while SCL is high.
3. Write data in ICDR. The master device outputs the written data together with a sequence of transmit clock pulses at the timing shown in figure 13-6. If FS is 0 in SAR, the first byte following the start condition contains a 7-bit slave address and indicates the transmit/receive direction. The selected slave device (the device with the matching slave address) drives SDA low at the ninth transmit clock pulse to acknowledge the data.
4. When one byte of data has been transmitted, IRIC is set to 1 in ICSR at the rise of the ninth transmit clock pulse. If IEIC is set to 1 in ICCR, a CPU interrupt is requested. After one frame has been transferred, SCL is automatically brought to the low level in synchronization with the internal clock and held low.
5. Software clears IRIC to 0 in ICSR.
6. To continue transmitting, write the next transmit data in ICDR. Transmission of the next byte will begin in synchronization with the internal clock.

Steps 4 to 6 can be repeated to transmit data continuously. To end the transmission, write 0 in BBSY and 0 in SCP in ICSR. This generates a stop condition by causing a low-to-high transition of SDA while SCL is high.



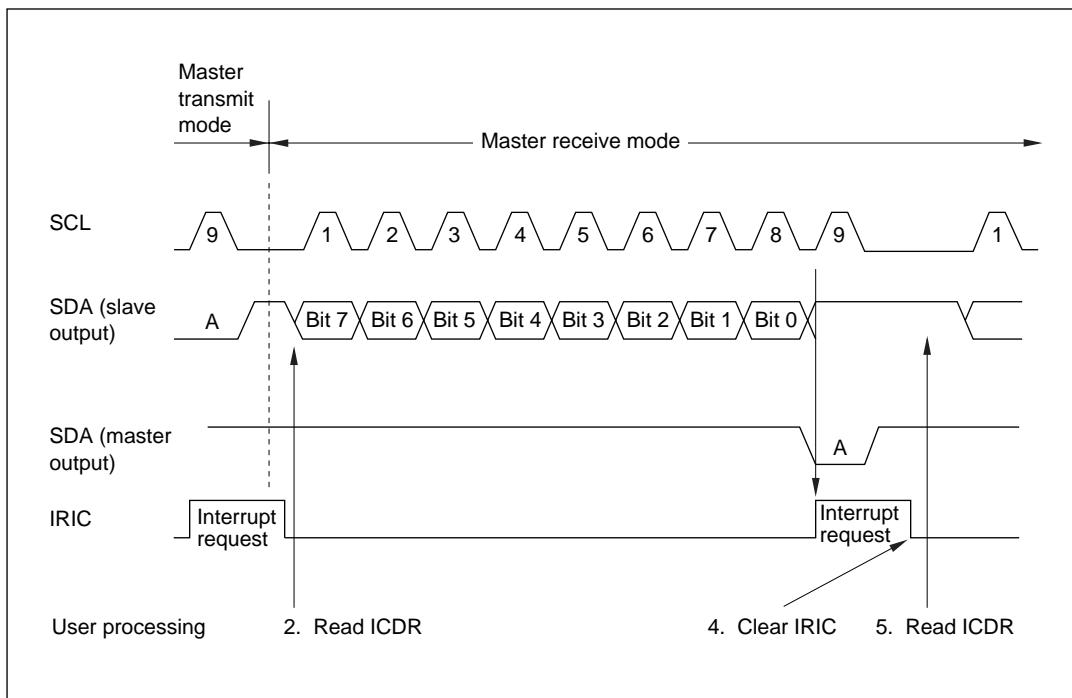
**Figure 13-6 Timing in Master Transmit Mode  
(MLS = WAIT = ACK = 0)**

### 13.3.3 Master Receive Operation

In master receive mode, the master device outputs the receive clock, receives data, and returns an acknowledge signal. The slave device transmits the data. The receive procedure and operations in master receive mode are described below. See also figure 13-7.

1. Clear TRS to 0 in ICCR to switch from transmit mode to receive mode.
2. Read ICDR to start receiving. When ICDR is read, a receive clock is output in synchronization with the internal clock, and data is received. At the ninth clock pulse the master device drives SDA low to acknowledge the data.
3. When one byte of data has been received, IRIC is set to 1 in ICSR at the rise of the ninth receive clock pulse. If IEIC is set to 1 in ICCR, a CPU interrupt is requested. After one frame has been transferred, SCL is automatically brought to the low level in synchronization with the internal clock and held low.
4. Software clears IRIC to 0 in ICSR.
5. When ICDR is read, receiving of the next data starts in synchronization with the internal clock.

Steps 3 to 5 can be repeated to receive data continuously. To stop receiving, set TRS to 1, read ICDR, then write 0 in BBSY and 0 in SCP in ICSR. This generates a stop condition by causing a low-to-high transition of SDA while SCL is high. If it is not necessary to acknowledge each byte of data, set ACKB to 1 in ICSR before receiving starts.



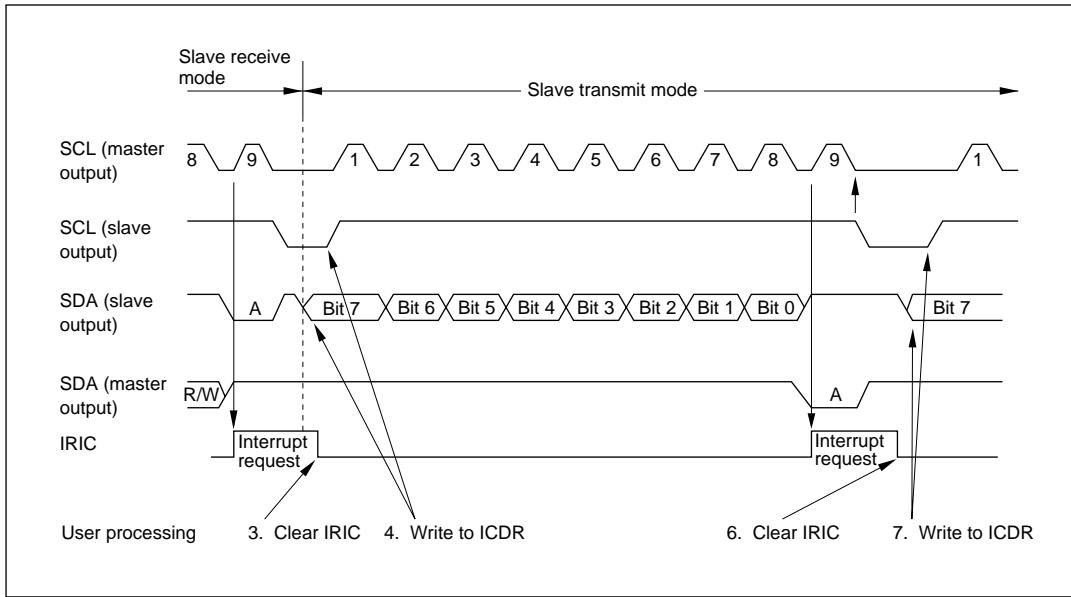
**Figure 13-7 Timing in Master Receive Mode**  
**(MLS = WAIT = ACK = ACKB = 0)**

### 13.3.4 Slave Transmit Operation

In slave transmit mode, the slave device outputs the transmit data, and the master device outputs the transmit clock and returns an acknowledge signal. The transmit procedure and operations in slave transmit mode are described below.

1. Set bits MLS and WAIT in ICMR and bits MST, TRS, ACK, and CKS2 to CKS0 in ICCR according to the operating mode. Set bit ICE in ICCR to 1, establishing slave receive mode.
2. After the slave device detects a start condition, if the first byte matches its slave address, at the ninth clock pulse the slave device drives SDA low to acknowledge the transfer. At the same time, IRIC is set to 1 in ICSR, generating an interrupt. If the eighth data bit ( $R/\overline{W}$ ) is 1, the TRS bit is set to 1 in ICCR, automatically causing a transition to slave transmit mode. The slave device holds SCL low from the fall of the transmit clock until data is written in ICDR.
3. Software clears IRIC to 0 in ICSR.
4. Write data in ICDR. The slave device outputs the written data serially in step with the clock output by the master device, with the timing shown in figure 13-8.
5. When one byte of data has been transmitted, at the rise of the ninth transmit clock pulse IRIC is set to 1 in ICSR. If IEIC is set to 1 in ICCR, a CPU interrupt is requested. The slave device holds SCL low from the fall of the transmit clock until data is written in ICDR. The master device drives SDA low at the ninth clock pulse to acknowledge the data. The acknowledge signal is stored in ACKB in ICSR, and can be used to check whether the transfer was carried out normally.
6. Software clears IRIC to 0 in ICSR.
7. To continue transmitting, write the next transmit data in ICDR.

Steps 5 to 7 can be repeated to transmit continuously. To end the transmission, write H'FF in ICDR. When a stop condition is detected (a low-to-high transition of SDA while SCL is high), BBSY will be cleared to 0 in ICSR.



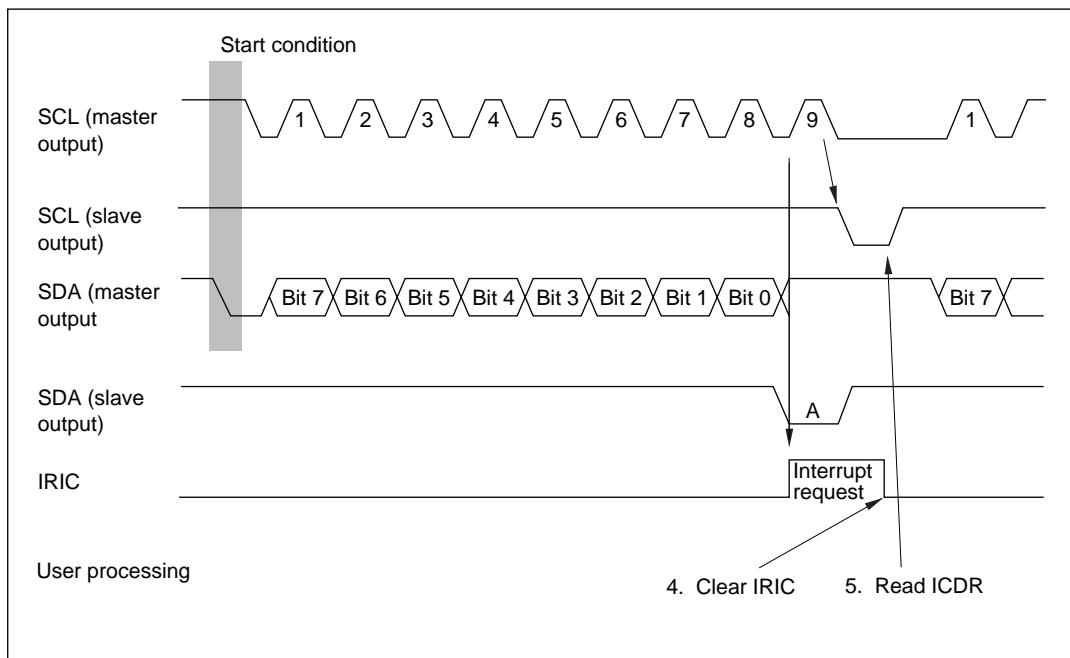
**Figure 13-8 Timing in Slave Transmit Mode  
(MLS = WAIT = ACK = 0)**

### 13.3.5 Slave Receive Operation

In slave receive mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The receive procedure and operations in slave receive mode are described below. See also figure 13-9.

1. Set bits MLS and WAIT in ICMR and bits MST, TRS, and ACK in ICCR according to the operating mode. Set bit ICE in ICCR to 1, establishing slave receive mode.
2. A start condition output by the master device sets BBSY to 1 in ICSR.
3. After the slave device detects the start condition, if the first byte matches its slave address, at the ninth clock pulse the slave device drives SDA low to acknowledge the transfer. At the same time, IRIC is set to 1 in ICSR. If IEIC is 1 in ICCR, a CPU interrupt is requested. The slave device holds SCL low from the fall of the receive clock until it has read the data in ICDR.
4. Software clears IRIC to 0 in ICSR.
5. When ICDR is read, receiving of the next data starts.

Steps 4 and 5 can be repeated to receive data continuously. When a stop condition is detected (a low-to-high transition of SDA while SCL is high), BBSY is cleared to 0 in ICSR.

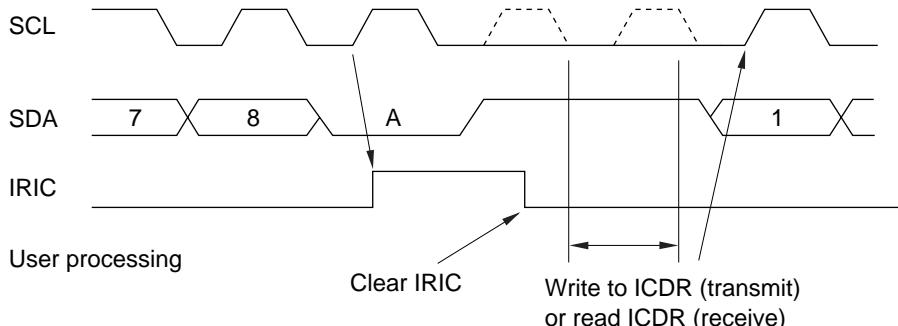


**Figure 13-9 Timing in Slave Receive Mode**  
**(MLS = WAIT = ACK = ACKB = 0)**

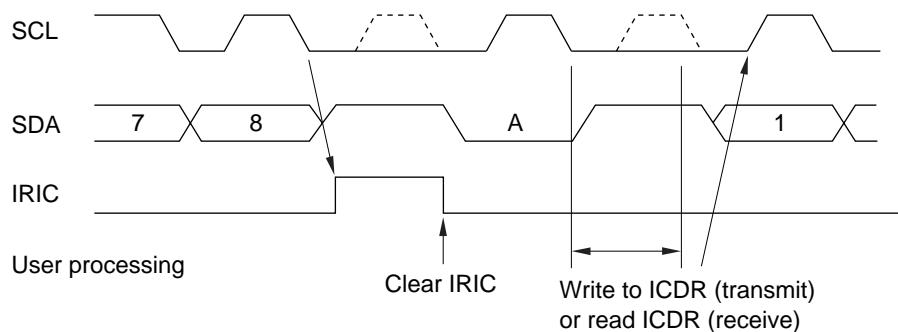
### 13.3.6 IRIC Set Timing and SCL Control

The interrupt request flag (IRIC) is set at different times depending on the WAIT bit in ICMR and ACK bit in ICCR. SCL is automatically held low after one frame has been transferred; this timing is synchronized with the internal clock. Figure 13-10 shows the IRIC set timing and SCL control.

(a) When WAIT = 0 and ACK = 0



(b) When WAIT = 1 and ACK = 0



Note: The ICDR write (transmit) or read (receive) following the clearing of IRIC should be executed after the rise of SCL (ninth clock pulse).

(c) When ACK = 1

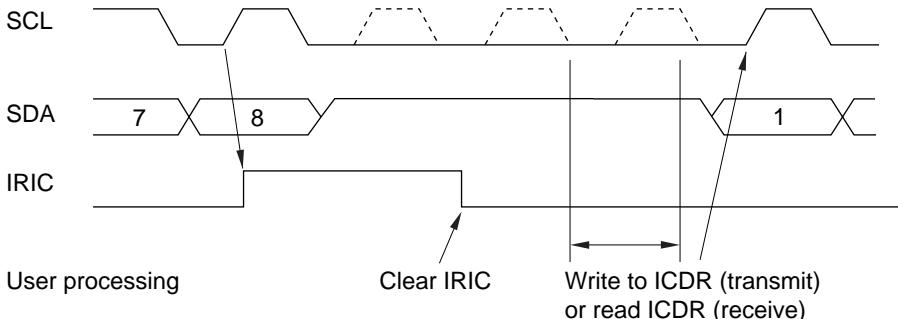


Figure 13-10 IRIC Set Timing and SCL Control

### 13.3.7 Noise Canceler

The logic levels at the SCL and SDA pins are routed through noise cancelers before being latched internally. Figure 13-11 shows a block diagram of the noise canceler.

The noise canceler consists of two cascaded latches and a match detector. The SCL (or SDA) input signal is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.

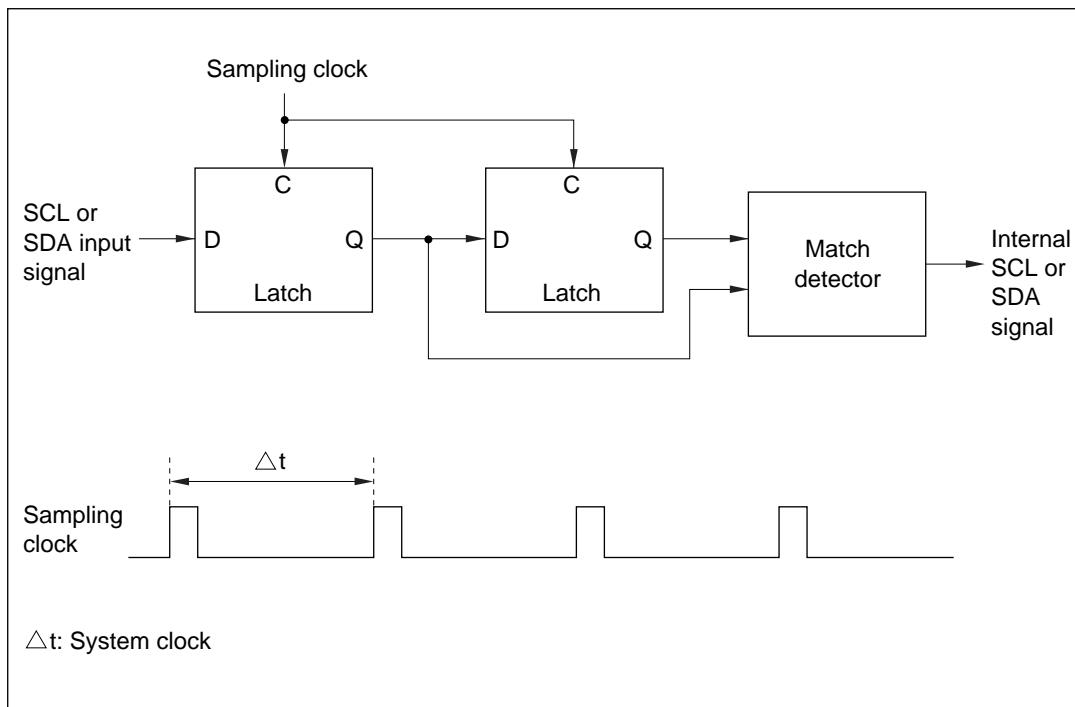


Figure 13-11 Block Diagram of Noise Canceler

### 13.3.8 Sample Flowcharts

Figures 13-12 to 13-15 show typical flowcharts for using the I<sup>2</sup>C bus interface in each mode.

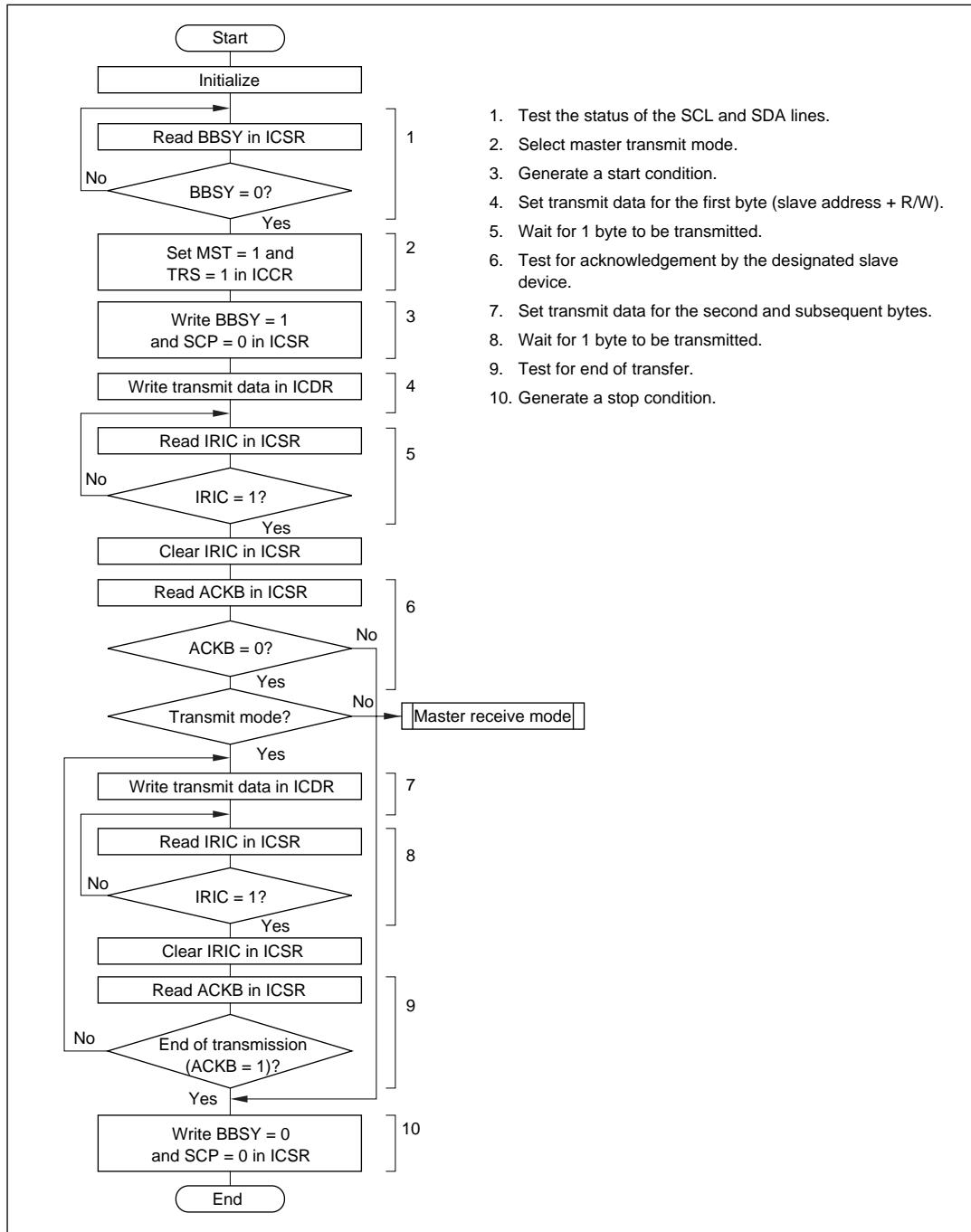
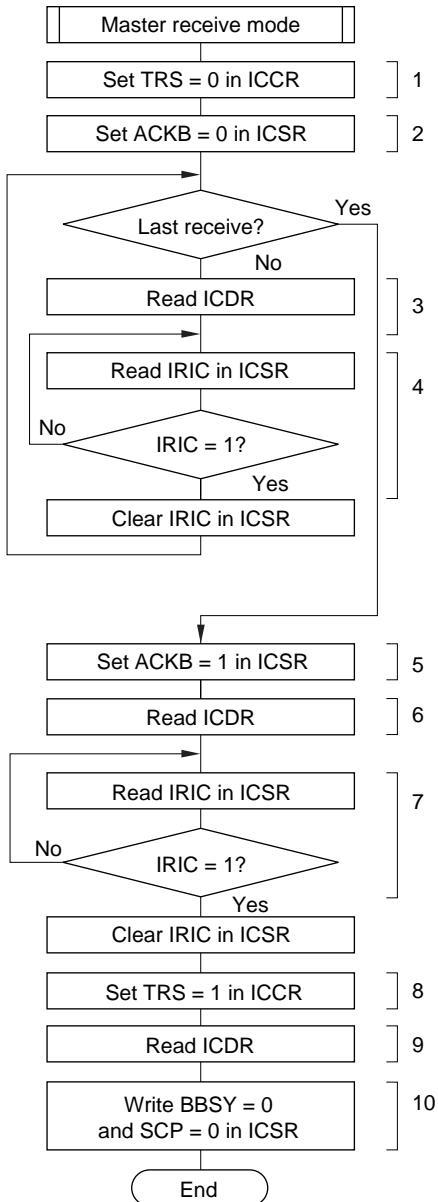
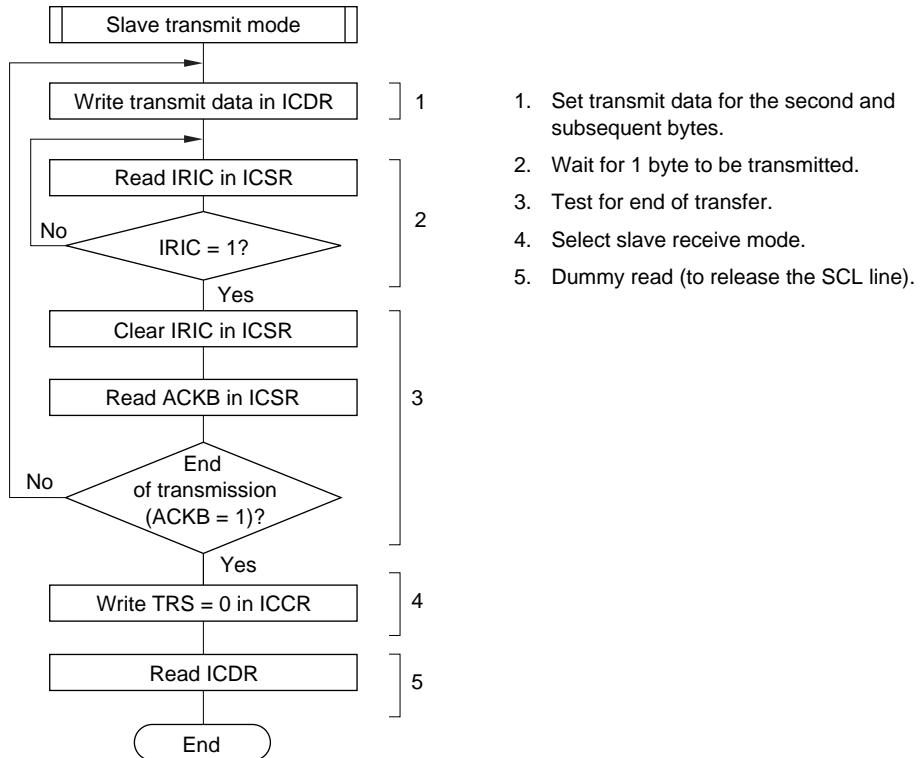


Figure 13-12 Flowchart for Master Transmit Mode (Example)



1. Select receive mode.
2. Set acknowledgement data.
3. Start receiving. The first read is a dummy read.
4. Wait for 1 byte to be received.
5. Set acknowledgement data for the last receive.
6. Start the last receive.
7. Wait for 1 byte to be received.
8. Select transmit mode.
9. Read the last receive data (if ICDR is read without selecting transmit mode, receive operations will resume).
10. Generate a stop condition.

**Figure 13-13 Flowchart for Master Receive Mode (Example)**



**Figure 13-14 Flowchart for Slave Transmit Mode (Example)**

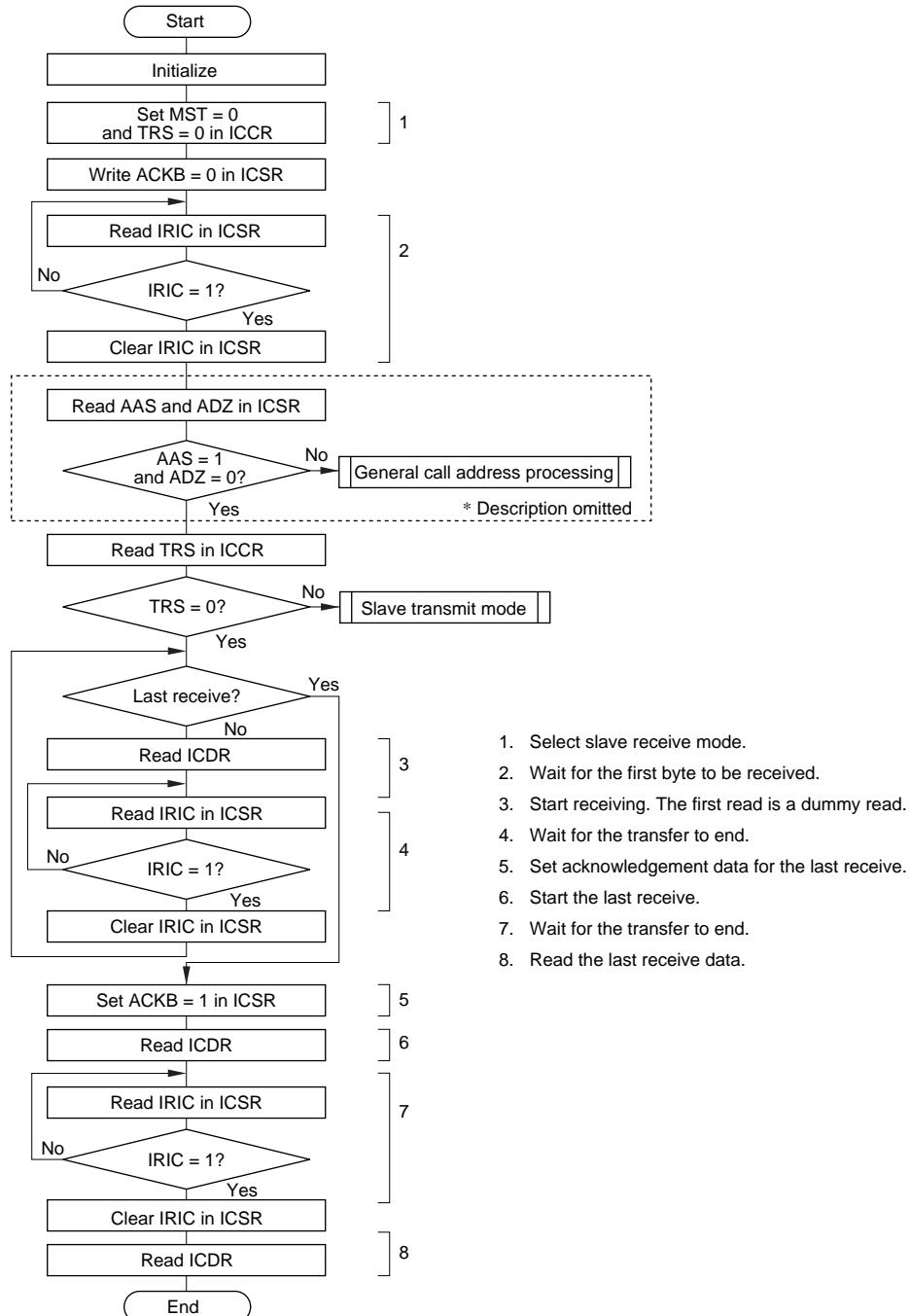


Figure 13-15 Flowchart for Slave Receive Mode (Example)

## 13.4 Application Notes

- In master mode, if an instruction to generate a start condition is immediately followed by an instruction to generate a stop condition, neither condition will be output correctly. To output consecutive start and stop conditions, after issuing the instruction that generates the start condition, read the relevant ports, check that SCL and SDA are both low, then issue the instruction that generates the stop condition.
- Either of the following two conditions will start the next transfer. Pay attention to these conditions when reading or writing to ICDR.
  1. Write access to ICDR when ICE = 1 and TRS = 1
  2. Read access to ICDR when ICE = 1 and TRS = 0
- The I<sup>2</sup>C bus interface specification for the SCL rise time tsr is under 1000 ns (300 ns for high-speed mode). In master mode, the I<sup>2</sup>C bus interface monitors the SCL line and synchronizes one bit at a time during communication. If tsr (the time for SCL to go from low to V<sub>IH</sub>) exceeds the time determined by the input clock of the I<sup>2</sup>C bus interface, the high period of SCL is extended. SCL rise time is determined by the pull-up resistance and load capacitance of the SCL line. To insure proper operation at the set transfer rate, adjust the pull-up resistance and load capacitance so that the SCL rise time falls below the values given in the table below.

CKDBL	IICX	$t_{cyc}$ Display	Time Display				
			$\phi = 4$ MHz	$\phi = 5$ MHz	$\phi = 8$ MHz	$\phi = 10$ MHz	$\phi = 16$ MHz
0	0	2.5t <sub>cyc</sub>	Normal mode	625 ns	500 ns	312 ns	250 ns
			High-speed mode	300 ns	300 ns	300 ns	
0	1	7.5t <sub>cyc</sub>	Normal mode	1000 ns	1000 ns	937 ns	750 ns
1	0		High-speed mode	300 ns	300 ns	300 ns	300 ns
1	1	17.5t <sub>cyc</sub>	Normal mode	1000 ns	1000 ns	1000 ns	1000 ns
			High-speed mode	300 ns	300 ns	300 ns	300 ns

# Section 14 Host Interface

## 14.1 Overview

The H8/3437 Series has an on-chip host interface (HIF) that provides a dual-channel parallel interface between the on-chip CPU and a host processor. The host interface is available only when the HIE bit is set to 1 in SYSCR. This mode is called slave mode, because it is designed for a master-slave communication system in which the H8/3437-Series chip is slaved to a host processor.

The host interface consists of four 1-byte data registers, two 1-byte status registers, a 1-byte control register, fast A<sub>20</sub> gate logic, and a host interrupt request circuit. Communication is carried out via five control signals from the host processor ( $\overline{CS}_1$ ,  $\overline{CS}_2$  or  $\overline{ECS}_2$ , HA<sub>0</sub>,  $\overline{IOR}$ , and  $\overline{IOW}$  or  $\overline{EIOW}$ ), four output signals to the host processor (GA<sub>20</sub>, HIRQ<sub>1</sub>, HIRQ<sub>11</sub>, and HIRQ<sub>12</sub>), and an 8-bit bidirectional command/data bus (HDB<sub>7</sub> to HDB<sub>0</sub>, or XDB<sub>7</sub> to XDB<sub>0</sub>). The  $\overline{CS}_1$  and  $\overline{CS}_2$  (or  $\overline{ECS}_2$ ) signals select one of the two interface channels.

Note: If one of the two interface channels will not be used, tie the unused  $\overline{CS}$  pin to V<sub>CC</sub>. For example, if interface channel 1 (IDR1, ODR1, STR1) is not used, tie  $\overline{CS}_1$  to V<sub>CC</sub>.

#### 14.1.1 Block Diagram

Figure 14-1 is a block diagram of the host interface.

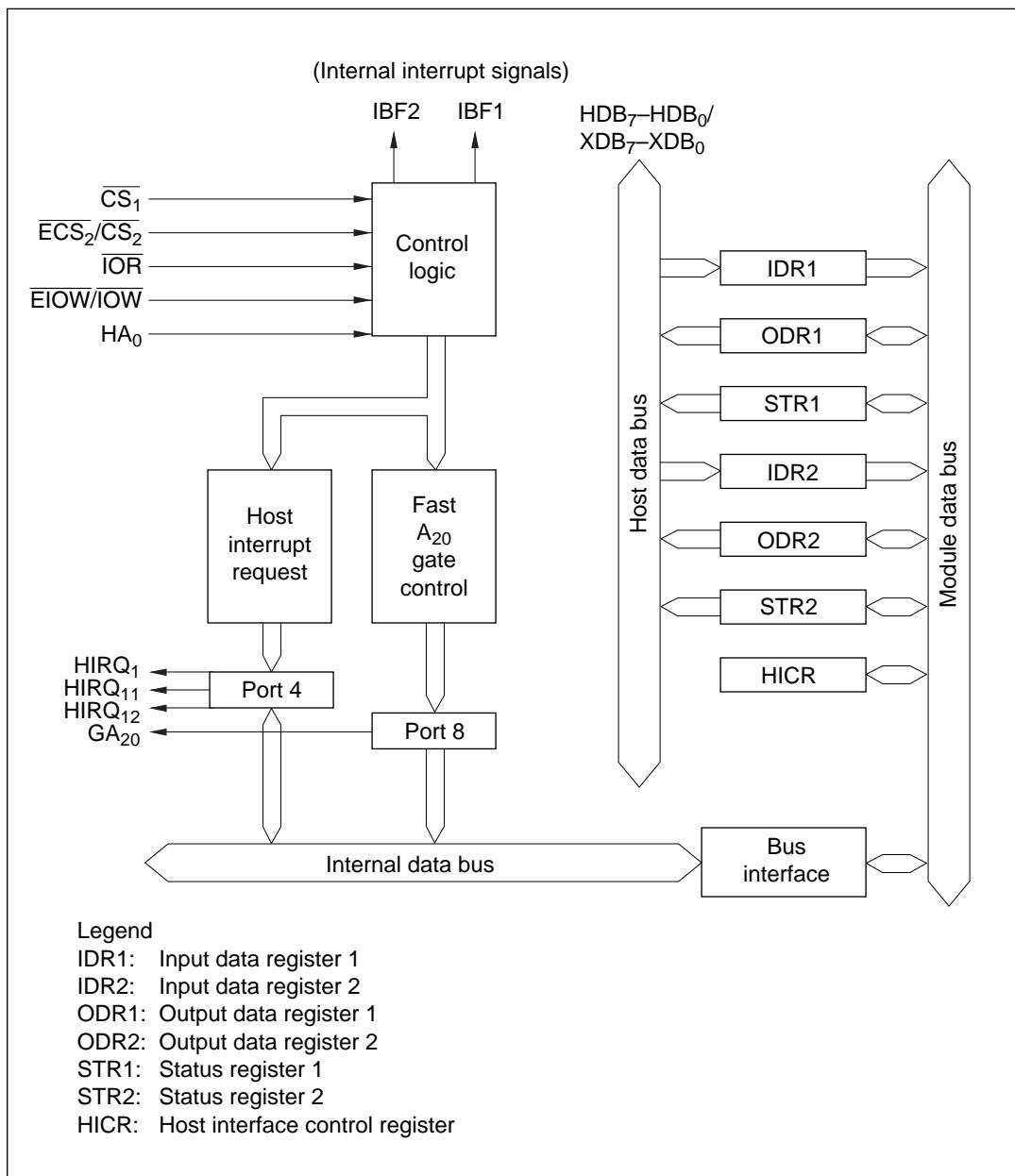


Figure 14-1 Host Interface Block Diagram

## 14.1.2 Input and Output Pins

Table 14-1 lists the input and output pins of the host interface module.

**Table 14-1 H/F Input/Output Pins**

Name	Abbreviation	Port	I/O	Function
I/O read	$\overline{IOR}$	P8 <sub>3</sub>	Input	Host interface read signal
I/O write*	$\overline{IOW}$	P8 <sub>4</sub>	Input	Host interface write signal
	$\overline{EIOW}$	P9 <sub>1</sub>		
Chip select 1	$\overline{CS}_1$	P8 <sub>2</sub>	Input	Host interface chip select signal for IDR1, ODR1, STR1
Chip select 2*	$\overline{CS}_2$	P8 <sub>5</sub>	Input	Host interface chip select signal for IDR2, ODR2, STR2
	$\overline{ECS}_2$	P9 <sub>0</sub>		
Command/data	HA <sub>0</sub>	P8 <sub>0</sub>	Input	Host interface address select signal In host read access, this signal selects the status registers (STR1, STR2) or data registers (ODR1, ODR2). In host write access to the data registers (IDR1, IDR2), this signal indicates whether the host is writing a command or data.
Data bus	HDB <sub>7</sub> –HDB <sub>0</sub>	P3 <sub>7</sub> –P3 <sub>0</sub>	I/O	Host interface data bus (single-chip mode)
	XDB <sub>7</sub> –XDB <sub>0</sub>	PB <sub>7</sub> –PB <sub>0</sub>	I/O	Host interface data bus (expanded modes)
Host interrupt 1	HIRQ <sub>1</sub>	P4 <sub>4</sub>	Output	Interrupt output 1 to host
Host interrupt 11	HIRQ <sub>11</sub>	P4 <sub>3</sub>	Output	Interrupt output 11 to host
Host interrupt 12	HIRQ <sub>12</sub>	P4 <sub>5</sub>	Output	Interrupt output 12 to host
Gate A <sub>20</sub>	GA <sub>20</sub>	P8 <sub>1</sub>	Output	A <sub>20</sub> gate control signal output

Note: \* Selection between  $\overline{IOW}$  and  $\overline{EIOW}$ , and between  $\overline{CS}_2$  and  $\overline{ECS}_2$ , is by the STAC bit in STCR.  $\overline{IOW}$  and  $\overline{CS}_2$  are used when STAC is 0.  $\overline{EIOW}$  and  $\overline{ECS}_2$  are used when STAC is 1. In this manual, both are referred to as  $\overline{IOW}$  and  $\overline{CS}_2$ .

### 14.1.3 Register Configuration

Table 14-2 lists the host interface registers.

**Table 14-2 HIF Registers**

Name	Abbreviation	R/W		Initial Value	Slave Address <sup>*3</sup>	Master Address <sup>*4</sup>		
		Slave	Host			CS <sub>1</sub>	CS <sub>2</sub>	HA <sub>0</sub>
System control register	SYSCR	R/W <sup>*1</sup>	—	H'09	H'FFC4	—	—	—
Host interface control register	HICR	R/W	—	H'F8	H'FFF0	—	—	—
Input data register 1	IDR1	R	W	—	H'FFF4	0	1	0/1 <sup>*5</sup>
Output data register 1	ODR1	R/W	R	—	H'FFF5	0	1	0
Status register 1	STR1	R/(W) <sup>*2</sup>	R	H'00	H'FFF6	0	1	1
Input data register 2	IDR2	R	W	—	H'FFFC	1	0	0
Output data register 2	ODR2	R/W	R	—	H'FFFD	1	0	0/1 <sup>*5</sup>
Status register 2	STR2	R/(W) <sup>*2</sup>	R	H'00	H'FFFE	1	0	1
Serial/timer control register	STCR	R/W	—	H'00	H'FFC3	—	—	—

Notes:

1. Bit 3 is a read-only bit.

2. The user-defined bits (bits 7 to 4) are read/write accessible from the slave processor.
3. Address when accessed from the slave processor.
4. Pin inputs used in access from the host processor.
5. The HA<sub>0</sub> input discriminates between writing of commands and data.

## 14.2 Register Descriptions

### 14.2.1 System Control Register (SYSCR)

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

SYSCR is an 8-bit read/write register which controls chip operations. Host interface functions are enabled or disabled by the HIE bit of SYSCR. See section 3.2, System Control Register, for information on other SYSCR bits. SYSCR is initialized to H'09 by an external reset or during standby mode.

**Bit 1—Host Interface Enable (HIE):** Enables or disables the host interface. When enabled, the host interface handles host-slave data transfers, operating in slave mode.

Bit 1	
HIE	Description
0	The host interface is disabled
1	The host interface is enabled (slave mode)

### 14.2.2 Host Interface Control Register (HICR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	IBFIE2	IBFIE1	FGA20E
Initial value	1	1	1	1	1	0	0	0
Slave Read/Write	—	—	—	—	—	R/W	R/W	R/W
Host Read/Write	—	—	—	—	—	—	—	—

HICR is an 8-bit read/write register which controls host interface interrupts and the fast A<sub>20</sub> gate function. HICR is initialized to H'F8 by a reset or during standby mode.

**Bits 7 to 3—Reserved:** These bits cannot be modified and are always read as 1.

**Bit 2—Input Buffer Full Interrupt Enable 2 (IBFIE2):** Enables or disables the IBF2 interrupt to the slave CPU.

**Bit 2**

**IBFIE2      Description**

0	IDR2 input buffer full interrupt is disabled	(Initial value)
1	IDR2 input buffer full interrupt is enabled	

**Bit 1—Input Buffer Full Interrupt Enable 1 (IBFIE1):** Enables or disables the IBF1 interrupt to the slave CPU.

**Bit 1**

**IBFIE1      Description**

0	IDR1 input buffer full interrupt is disabled	(Initial value)
1	IDR1 input buffer full interrupt is enabled	

**Bit 0—Fast Gate A<sub>20</sub> Enable (FGA20E):** Enables or disables the fast A<sub>20</sub> gate function. When the fast A<sub>20</sub> gate is disabled, a regular-speed A<sub>20</sub> gate signal can be implemented by using software to manipulate the P8<sub>1</sub> output.

**Bit 0**

**FGA20E      Description**

0	Disables fast A <sub>20</sub> gate function	(Initial value)
1	Enables fast A <sub>20</sub> gate function	

### 14.2.3 Input Data Register (IDR1)

Bit	7	6	5	4	3	2	1	0
Initial value	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Slave Read/Write	—	—	—	—	—	—	—	—
Host Read/Write	R	R	R	R	R	R	R	R

IDR1 is an 8-bit read-only register to the slave processor, and an 8-bit write-only register to the host processor. When  $\overline{CS}_1$  is low, information on the host data bus is written into IDR1 at the rising edge of  $\overline{IOW}$ . The HA<sub>0</sub> state is also latched into the C/D bit in STR1 to indicate whether the written information is a command or data.

The initial values of IDR1 after a reset or standby are undetermined.

#### 14.2.4 Output Data Register (ODR1)

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R/W							
Host Read/Write	R	R	R	R	R	R	R	R

ODR1 is an 8-bit read/write register to the slave processor, and an 8-bit read-only register to the host processor. The ODR1 contents are output on the host data bus when  $HA_0$  is low,  $\overline{CS}_1$  is low, and  $\overline{IOR}$  is low.

The initial values of ODR1 after a reset or standby are undetermined.

#### 14.2.5 Status Register (STR1)

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	C/D	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R
Host Read/Write	R	R	R	R	R	R	R	R

STR1 is an 8-bit register that indicates status information during host interface processing. Bits 3, 1, and 0 are read-only bits to both the host and slave processors.

STR1 is initialized to H'00 by a reset and in the standby modes.

**Bits 7 to 4 and Bit 2—Defined by User (DBU):** The user can use these bits as necessary.

**Bit 3—Command/Data (C/D):** Receives the  $HA_0$  input when the host processor writes to IDR1, and indicates whether IDR1 contains data or a command.

Bit 3		
C/D	Description	
0	Contents of IDR1 are data	(Initial value)
1	Contents of IDR1 are a command	

**Bit 1—Input Buffer Full (IBF):** Set to 1 when the host processor writes to IDR1. This bit is an internal interrupt source to the slave processor. IBF is cleared to 0 when the slave processor reads IDR1.

Bit 1	
IBF	Description
0	This bit is cleared when the slave processor reads IDR1
1	This bit is set when the host processor writes to IDR1

**Bit 0—Output Buffer Full (OBF):** Set to 1 when the slave processor writes to ODR1. Cleared to 0 when the host processor reads ODR1.

Bit 0	
OBF	Description
0	This bit is cleared when the host processor reads ODR1
1	This bit is set when the slave processor writes to ODR1

Table 14-3 shows the conditions for setting and clearing the STR1 flags.

**Table 14-3 Set/Clear Timing for STR1 Flags**

Flag	Setting Condition	Clearing Condition
C/D	Rising edge of host's write signal ( $\overline{IOW}$ ) when $HA_0$ is high	Rising edge of host's write signal ( $\overline{IOW}$ ) when $HA_0$ is low
IBF	Rising edge of host's write signal ( $\overline{IOW}$ ) when writing to IDR1	Falling edge of slave's internal read signal ( $\overline{RD}$ ) when reading IDR1
OBF	Falling edge of slave's internal write signal ( $\overline{WR}$ ) when writing to ODR1	Rising edge of host's read signal ( $\overline{IOR}$ ) when reading ODR1

#### 14.2.6 Input Data Register (IDR2)

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

IDR2 is an 8-bit read-only register to the slave processor, and an 8-bit write-only register to the host processor. When  $\overline{CS}_2$  is low, information on the host data bus is written into IDR2 at the rising edge of  $\overline{IOW}$ . The  $HA_0$  state is also latched into the C/D bit in STR2 to indicate whether the written information is a command or data.

The initial values of IDR2 after a reset or standby are undetermined.

### 14.2.7 Output Data Register (ODR2)

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Slave Read/Write	R/W							
Host Read/Write	R	R	R	R	R	R	R	R

ODR2 is an 8-bit read/write register to the slave processor, and an 8-bit read-only register to the host processor. The ODR2 contents are output on the host data bus when  $HA_0$  is low,  $\overline{CS}_2$  is low, and  $\overline{IOR}$  is low.

The initial values of ODR2 after a reset or standby are undetermined.

### 14.2.8 Status Register (STR2)

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	$C/\overline{D}$	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R
Host Read/Write	R	R	R	R	R	R	R	R

STR2 is an 8-bit register that indicates status information during host interface processing. Bits 3, 1, and 0 are read-only bits to both the host and slave processors.

STR2 is initialized to H'00 by a reset and in the standby modes.

**Bits 7 to 4 and Bit 2—Defined by User (DBU):** The user can use these bits as necessary.

**Bit 3—Command/Data ( $C/\overline{D}$ ):** Receives the  $HA_0$  input when the host processor writes to IDR2, and indicates whether IDR2 contains data or a command.

Bit 3		
C/D	Description	
0	Contents of IDR2 are data	(Initial value)
1	Contents of IDR2 are a command	

**Bit 1—Input Buffer Full (IBF):** Set to 1 when the host processor writes to IDR2. This bit is an internal interrupt source to the slave processor. IBF is cleared to 0 when the slave processor reads IDR2.

**Bit 1**

IBF	Description	
0	This bit is cleared when the slave processor reads IDR2	(Initial value)
1	This bit is set when the host processor writes to IDR2	

**Bit 0—Output Buffer Full (OBF):** Set to 1 when the slave processor writes to ODR2. Cleared to 0 when the host processor reads ODR2.

**Bit 0**

OBF	Description	
0	This bit is cleared when the host processor reads ODR2	(Initial value)
1	This bit is set when the slave processor writes to ODR2	

Table 14-4 shows the conditions for setting and clearing the STR2 flags.

**Table 14-4 Set/Clear Timing for STR2 Flags**

Flag	Setting Condition	Clearing Condition
C/D	Rising edge of host's write signal ( $\overline{IOW}$ ) when $HA_0$ is high	Rising edge of host's write signal ( $\overline{IOW}$ ) when $HA_0$ is low
IBF	Rising edge of host's write signal ( $\overline{IOW}$ ) when writing to IDR2	Falling edge of slave's internal read signal ( $\overline{RD}$ ) when reading IDR2
OBF	Falling edge of slave's internal write signal ( $\overline{WR}$ ) when writing to ODR2	Rising edge of host's read signal ( $\overline{IOR}$ ) when reading ODR2

#### 14.2.9 Serial/Timer Control Register (STCR)

Bit	7	6	5	4	3	2	1	0
	IICS	IICD	IICX	IICE	STAC	MPE	ICKS1	ICKS0
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

STCR is an 8-bit readable/writable register that controls the I<sup>2</sup>C bus interface and host interface, controls the SCI operating mode, and selects the TCNT clock source in the 8-bit timers. STCR is initialized to H'00 by a reset.

**Bits 7 to 4—I<sup>2</sup>C Control (IICS, IICD, IICX, IICE):** These bits are used to control the I<sup>2</sup>C bus interface. For details, see section 13, I<sup>2</sup>C Bus Interface.

**Bit 3—Slave Input Switch (STAC):** Controls switching of host interface input pins. Settings of this bit are valid only when the host interface is enabled (slave mode).

Bit 3		
STAC	Description	
0	In port 8, P8 <sub>5</sub> switches over to $\overline{CS}_2$ , and P8 <sub>4</sub> to $\overline{IOW}$	(Initial value)
1	In port 9, P9 <sub>1</sub> switches over to $\overline{EIOW}$ , and P9 <sub>0</sub> to $\overline{ECS}_2$	

**Bit 2—Multiprocessor Enable (MPE):** Controls the operating mode of SCI0 and SCI1. For details, see section 12, Serial Communication Interface.

**Bits 1 and 0—Internal Clock Source Select 1 and 0 (ICKS1, ICSK0):** Together with bits CKS2 to CKS0 in TCR, these bits select timer counter clock inputs. For details, see section 9, 8-Bit Timers.

## 14.3 Operation

### 14.3.1 Host Interface Operation

The host interface is activated by setting the HIE bit (bit 1) to 1 in SYSCR, establishing slave mode. Activation of the host interface (entry to slave mode) appropriates the related I/O lines in port 3 or B (data), port 8 or 9 (control) and port 4 (host interrupt requests) for interface use.

For host interface read/write timing diagrams, see section 22.3.8, Host Interface Timing.

### 14.3.2 Control States

Table 14-5 indicates the slave operations carried out in response to host interface signals from the host processor.

**Table 14-5 Host Interface Operation**

<b>CS<sub>2</sub></b>	<b>CS<sub>1</sub></b>	<b>IOR</b>	<b>IOW</b>	<b>HA<sub>0</sub></b>	<b>Operation</b>
1	0	0	0	0	Prohibited
1	0	0	0	1	Prohibited
1	0	0	1	0	Data read from output data register 1 (ODR1)
1	0	0	1	1	Status read from status register 1 (STR1)
1	0	1	0	0	Data write to input data register 1 (IDR1)
1	0	1	0	1	Command write to input data register 1 (IDR1)
1	0	1	1	0	Idle state
1	0	1	1	1	Idle state
0	1	0	0	0	Prohibited
0	1	0	0	1	Prohibited
0	1	0	1	0	Data read from output data register 2 (ODR2)
0	1	0	1	1	Status read from status register 2 (STR2)
0	1	1	0	0	Data write to input data register 2 (IDR2)
0	1	1	0	1	Command write to input data register 2 (IDR2)
0	1	1	1	0	Idle state
0	1	1	1	1	Idle state

### 14.3.3 A<sub>20</sub> Gate

The A<sub>20</sub> gate signal can mask address A<sub>20</sub> to emulate an addressing mode used by personal computers with an 8086\*-family CPU. In slave mode, a regular-speed A<sub>20</sub> gate signal can be output under software control, or a fast A<sub>20</sub> gate signal can be output under hardware control. Fast A<sub>20</sub> gate output is enabled by setting the FGA20E bit (bit 0) to 1 in HICR (H'FFF0).

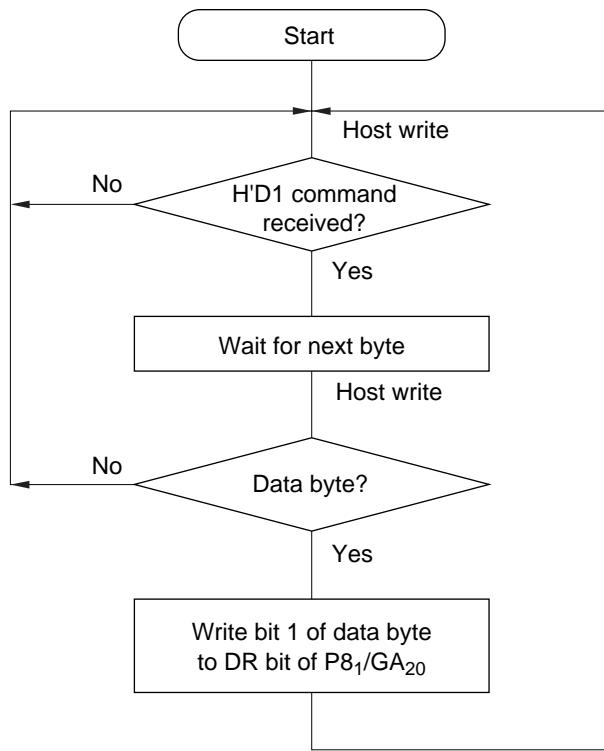
Note: \* Intel microprocessor.

**Regular A<sub>20</sub> Gate Operation:** Output of the A<sub>20</sub> gate signal can be controlled by an H'D1 command followed by data. When the slave processor receives data, it normally uses an interrupt routine activated by the IBF1 interrupt to read IDR1. If the data follows an H'D1 command, software copies bit 1 of the data and outputs it at the gate A<sub>20</sub> pin (P8<sub>1</sub>/GA<sub>20</sub>).

**Fast A<sub>20</sub> Gate Operation:** When the FGA20E bit is set to 1, P8<sub>1</sub>/GA<sub>20</sub> is used for output of a fast A<sub>20</sub> gate signal. Bit P8<sub>1</sub>DDR must be set to 1 to assign this pin for output. The initial output from this pin will be a logic 1, which is the initial DR value. Afterward, the host processor can manipulate the output from this pin by sending commands and data. This function is available only when register IDR1 is accessed using CS<sub>1</sub>. Slave logic decodes the commands input from the host processor. When an H'D1 host command is detected, bit 1 of the data following the host command is output from the GA<sub>20</sub> output pin. This operation does not depend on software or interrupts, and is faster than the regular processing using interrupts. Table 14-6 lists the conditions that set and clear GA<sub>20</sub> (P8<sub>1</sub>). Figure 14-2 describes the GA<sub>20</sub> output in flowchart form. Table 14-7 indicates the GA<sub>20</sub> output signal values.

**Table 14-6 GA<sub>20</sub> (P8<sub>1</sub>) Set/Clear Timing**

Pin Name	Setting Condition	Clearing Condition
GA <sub>20</sub> (P8 <sub>1</sub> )	Rising edge of the host's write signal (IOW) when bit 1 of the written data is 1 and the data follows an H'D1 host command	Rising edge of the host's write signal (IOW) when bit 1 of the written data is 0 and the data follows an H'D1 host command



**Figure 14-2 GA<sub>20</sub> Output**

**Table 14-7 Fast A<sub>20</sub> Gate Output Signal**

HA <sub>0</sub>	Data/Command	Internal CPU Interrupt Flag	GA <sub>20</sub> (PB <sub>1</sub> )	Remarks
1	H'D1 command	0	Q	Turn-on sequence
0	"1" data* <sup>1</sup>	0	1	
1	H'FF command	0	Q (1)	
1	H'D1 command	0	Q	Turn-off sequence
0	"0" data* <sup>2</sup>	0	0	
1	H'FF command	0	Q (0)	
1	H'D1 command	0	Q	Short turn-on sequence
0	"1" data* <sup>1</sup>	0	1	
1/0	Command other than H'FF and H'D1	1	Q (1)	
1	H'D1 command	0	Q	Short turn-off sequence
0	"0" data* <sup>2</sup>	0	0	
1/0	Command other than H'FF and H'D1	1	Q (0)	
1	H'D1 command	0	Q	Cancelled sequence
1	Command other than H'D1	1	Q	
1	H'D1 command	0	Q	Retriggered sequence
1	H'D1 command	0	Q	
1	H'D1 command	0	Q	Consecutively executed sequences
0	Any data	0	1/0	
1	H'D1 command	0	Q (1/0)	

Notes: 1. Arbitrary data with bit 1 set to 1.  
 2. Arbitrary data with bit 1 cleared to 0.

## 14.4 Interrupts

### 14.4.1 IBF1, IBF2

The host interface can request two interrupts to the slave CPU: IBF1 and IBF2. They are input buffer full interrupts for input data registers IDR1 and IDR2 respectively. Each interrupt is enabled when the corresponding enable bit is set (table 14-8).

**Table 14-8 Input Buffer Full Interrupts**

Interrupt	Description
IBF1	Requested when IBFIE1 is set to 1 and IDR1 is full
IBF2	Requested when IBFIE2 is set to 1 and IDR2 is full

### 14.4.2 HIRQ<sub>11</sub>, HIRQ<sub>1</sub>, and HIRQ<sub>12</sub>

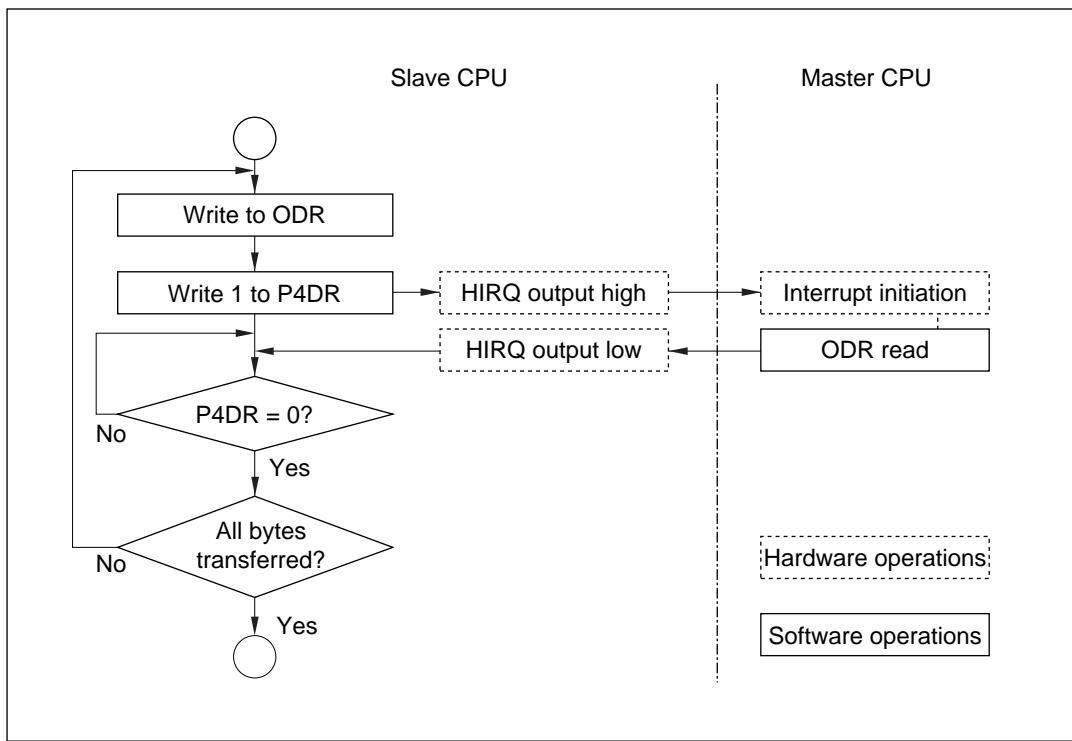
In slave mode (when HIE = 1 in SYSCR), three bits in the port 4 data register (P4DR) can be used as host interrupt request latches.

These three P4DR bits are cleared to 0 by the host processor's read signal ( $\overline{IOR}$ ). If  $\overline{CS}_1$  and  $HA_0$  are low, when  $\overline{IOR}$  goes low and the host reads ODR1, HIRQ<sub>1</sub> and HIRQ<sub>12</sub> are cleared to 0. If  $\overline{CS}_2$  and  $HA_0$  are low, when  $\overline{IOR}$  goes low and the host reads ODR2, HIRQ<sub>11</sub> is cleared to 0. To generate a host interrupt request, normally on-chip software writes 1 to the corresponding bit. In processing the interrupt, the host's interrupt-handling routine reads the output data register (ODR1 or ODR2), and this clears the host interrupt latch to 0.

Table 14-9 indicates how these bits are set and cleared. Figure 14-3 shows the processing in flowchart form.

**Table 14-9 Host Interrupt Set/Clear Conditions**

Host Interrupt Signal	Setting Condition	Clearing Condition
HIRQ <sub>11</sub> (P4 <sub>3</sub> )	Slave CPU reads 0 from P4DR bit 3, then writes 1	Slave CPU writes 0 in P4DR bit 3, or host reads output data register 2
HIRQ <sub>1</sub> (P4 <sub>4</sub> )	Slave CPU reads 0 from P4DR bit 4, then writes 1	Slave CPU writes 0 in P4DR bit 4, or host reads output data register 1
HIRQ <sub>12</sub> (P4 <sub>5</sub> )	Slave CPU reads 0 from P4DR bit 5, then writes 1	Slave CPU writes 0 in P4DR bit 5, or host reads output data register 1



**Figure 14-3 HIRQ Output Flowchart**

## 14.5 Application Note

The host interface provides buffering of asynchronous data from the host and slave processors, but an interface protocol must be followed to implement necessary functions and avoid data contention. For example, if the host and slave processors try to access the same input or output data register simultaneously, the data will be corrupted. Interrupts can be used to design a simple and effective protocol.

# Section 15 A/D Converter

## 15.1 Overview

The H8/3437 Series includes a 10-bit successive-approximations A/D converter with a selection of up to eight analog input channels.

### 15.1.1 Features

A/D converter features are listed below.

- 10-bit resolution
- Eight input channels
- Selectable analog conversion voltage range

The analog voltage conversion range can be programmed by input of an analog reference voltage at the  $AV_{ref}$  pin.

- High-speed conversion

Conversion time: minimum 8.4  $\mu$ s per channel (with 16-MHz system clock)

- Two conversion modes

Single mode: A/D conversion of one channel

Scan mode: continuous conversion on one to four channels

- Four 16-bit data registers

A/D conversion results are transferred for storage into data registers corresponding to the channels.

- Sample-and-hold function
- A/D conversion can be externally triggered
- A/D interrupt requested at end of conversion

At the end of A/D conversion, an A/D end interrupt (ADI) can be requested.

### 15.1.2 Block Diagram

Figure 15-1 shows a block diagram of the A/D converter.

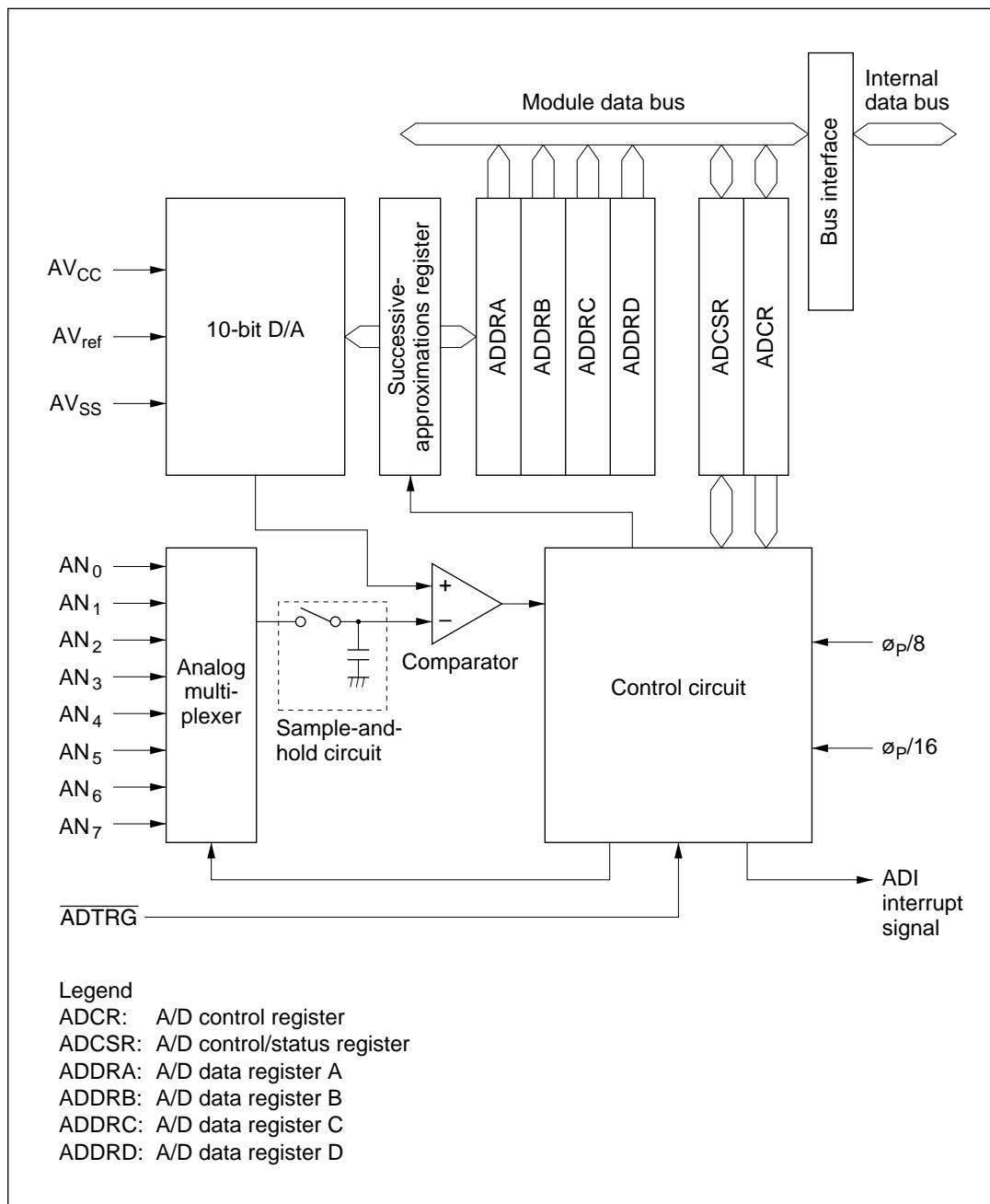


Figure 15-1 A/D Converter Block Diagram

### 15.1.3 Input Pins

Table 15-1 lists the A/D converter's input pins. The eight analog input pins are divided into two groups: group 0 (AN<sub>0</sub> to AN<sub>3</sub>), and group 1 (AN<sub>4</sub> to AN<sub>7</sub>). AV<sub>CC</sub> and AV<sub>SS</sub> are the power supply for the analog circuits in the A/D converter. AV<sub>ref</sub> is the A/D conversion reference voltage.

**Table 15-1 A/D Converter Pins**

Pin Name	Abbreviation	I/O	Function
Analog power supply pin	AV <sub>CC</sub>	Input	Analog power supply
Analog ground pin	AV <sub>SS</sub>	Input	Analog ground and reference voltage
Reference voltage pin	AV <sub>ref</sub>	Input	Analog reference voltage
Analog input pin 0	AN <sub>0</sub>	Input	Group 0 analog inputs
Analog input pin 1	AN <sub>1</sub>	Input	
Analog input pin 2	AN <sub>2</sub>	Input	
Analog input pin 3	AN <sub>3</sub>	Input	
Analog input pin 4	AN <sub>4</sub>	Input	Group 1 analog inputs
Analog input pin 5	AN <sub>5</sub>	Input	
Analog input pin 6	AN <sub>6</sub>	Input	
Analog input pin 7	AN <sub>7</sub>	Input	
A/D external trigger input pin	ADTRG	Input	External trigger input for starting A/D conversion

#### 15.1.4 Register Configuration

Table 15-2 summarizes the A/D converter's registers.

**Table 15-2 A/D Converter Registers**

Name	Abbreviation	R/W	Initial Value	Address
A/D data register A (high)	ADDRAH	R	H'00	H'FFE0
A/D data register A (low)	ADDRAL	R	H'00	H'FFE1
A/D data register B (high)	ADDRBH	R	H'00	H'FFE2
A/D data register B (low)	ADDRBL	R	H'00	H'FFE3
A/D data register C (high)	ADDRCH	R	H'00	H'FFE4
A/D data register C (low)	ADDRCL	R	H'00	H'FFE5
A/D data register D (high)	ADDRDH	R	H'00	H'FFE6
A/D data register D (low)	ADDRDL	R	H'00	H'FFE7
A/D control/status register	ADCSR	R/(W)*	H'00	H'FFE8
A/D control register	ADCR	R/W	H'7F	H'FFE9

Note: \* Only 0 can be written in bit 7, to clear the flag.

## 15.2 Register Descriptions

### 15.2.1 A/D Data Registers A to D (ADDRA to ADDRD)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	—	—	—	—	—	—
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

**Bits 15 to 6—A/D Conversion Data (AD9 to AD0):** 10-bit data giving an A/D conversion result.

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

The four A/D data registers (ADDRA to ADDRD) are 16-bit read-only registers that store the results of A/D conversion.

An A/D conversion produces 10-bit data, which is transferred for storage into the A/D data register corresponding to the selected channel. The upper 8 bits of the result are stored in the upper byte of the A/D data register. The lower 2 bits are stored in the lower byte. Bits 5 to 0 of an A/D data register are reserved bits that always read 0. Table 15-3 indicates the pairings of analog input channels and A/D data registers.

The CPU can always read the A/D data registers. The upper byte can be read directly, but the lower byte is read through a temporary register (TEMP). For details see section 15.3, CPU Interface.

The A/D data registers are initialized to H'0000 by a reset and in standby mode.

**Table 15-3 Analog Input Channels and A/D Data Registers**

Analog Input Channel		
Group 0	Group 1	A/D Data Register
AN <sub>0</sub>	AN <sub>4</sub>	ADDRA
AN <sub>1</sub>	AN <sub>5</sub>	ADDRB
AN <sub>2</sub>	AN <sub>6</sub>	ADDRC
AN <sub>3</sub>	AN <sub>7</sub>	ADDRD

## 15.2.2 A/D Control/Status Register (ADCSR)

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

Note: \* Only 0 can be written, to clear the flag.

ADCSR is an 8-bit readable/writable register that selects the mode and controls the A/D converter. ADCSR is initialized to H'00 by a reset and in standby mode.

**Bit 7—A/D End Flag (ADF):** Indicates the end of A/D conversion.

**Bit 7**

**ADF**

**Description**

0	[Clearing condition] Cleared by reading ADF while ADF = 1, then writing 0 in ADF	(Initial value)
1	[Setting conditions] 1. Single mode: A/D conversion ends 2. Scan mode: A/D conversion ends in all selected channels	

**Bit 6—A/D Interrupt Enable (ADIE):** Enables or disables the interrupt (ADI) requested at the end of A/D conversion.

**Bit 6**

**ADIE**

**Description**

0	A/D end interrupt request (ADI) is disabled	(Initial value)
1	A/D end interrupt request (ADI) is enabled	

**Bit 5—A/D Start (ADST):** Starts or stops A/D conversion. The ADST bit remains set to 1 during A/D conversion. It can also be set to 1 by external trigger input at the  $\overline{\text{ADTRG}}$  pin.

**Bit 5**

**ADST**

**Description**

0	A/D conversion is stopped	(Initial value)
1	1. Single mode: A/D conversion starts; ADST is automatically cleared to 0 when conversion ends 2. Scan mode: A/D conversion starts and continues, cycling among the selected channels, until ADST is cleared to 0 by software, by a reset, or by a transition to standby mode	

**Bit 4—Scan Mode (SCAN):** Selects single mode or scan mode. For further information on operation in these modes, see section 15.4, Operation. Clear the ADST bit to 0 before switching the conversion mode.

Bit 4	SCAN	Description	
0	Single mode		(Initial value)
1	Scan mode		

**Bit 3—Clock Select (CKS):** Selects the A/D conversion time. When  $\phi_P = \phi/2$ , the conversion time doubles. Clear the ADST bit to 0 before switching the conversion time.

Bit 3	CKS	Description	
0	Conversion time = 266 states (maximum) (when $\phi_P = \phi$ )		(Initial value)
1	Conversion time = 134 states (maximum) (when $\phi_P = \phi$ )		

**Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0):** These bits and the SCAN bit select the analog input channels. Clear the ADST bit to 0 before changing the channel selection.

Group Selection		Channel Selection		Description	
CH2	CH1	CH0	Single Mode	Scan Mode	
0	0	0	AN <sub>0</sub> (initial value)	AN <sub>0</sub>	
	0	1	AN <sub>1</sub>	AN <sub>0</sub> , AN <sub>1</sub>	
	1	0	AN <sub>2</sub>	AN <sub>0</sub> to AN <sub>2</sub>	
	1	1	AN <sub>3</sub>	AN <sub>0</sub> to AN <sub>3</sub>	
1	0	0	AN <sub>4</sub>	AN <sub>4</sub>	
	0	1	AN <sub>5</sub>	AN <sub>4</sub> , AN <sub>5</sub>	
	1	0	AN <sub>6</sub>	AN <sub>4</sub> to AN <sub>6</sub>	
	1	1	AN <sub>7</sub>	AN <sub>4</sub> to AN <sub>7</sub>	

### 15.2.3 A/D Control Register (ADCR)

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

ADCR is an 8-bit readable/writable register that enables or disables external triggering of A/D conversion. ADCR is initialized to H'7F by a reset and in standby mode.

**Bit 7—Trigger Enable (TRGE):** Enables or disables external triggering of A/D conversion.

#### Bit 7

#### TRGE      Description

0	A/D conversion cannot be externally triggered	(Initial value)
1	Enables start of A/D conversion by the external trigger input (ADTRG). (A/D conversion can be started either by an external trigger or by software.)	

**Bits 6 to 0—Reserved:** These bits cannot be modified, and are always read as 1.

## 15.3 CPU Interface

ADDRA to ADDRD are 16-bit registers, but they are connected to the CPU by an 8-bit data bus. Therefore, although the upper byte can be accessed directly by the CPU, the lower byte is read through an 8-bit temporary register (TEMP).

An A/D data register is read as follows. When the upper byte is read, the upper-byte value is transferred directly to the CPU and the lower-byte value is transferred into TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the CPU.

When reading an A/D data register, always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, incorrect data may be obtained.

Figure 15-2 shows the data flow for access to an A/D data register.

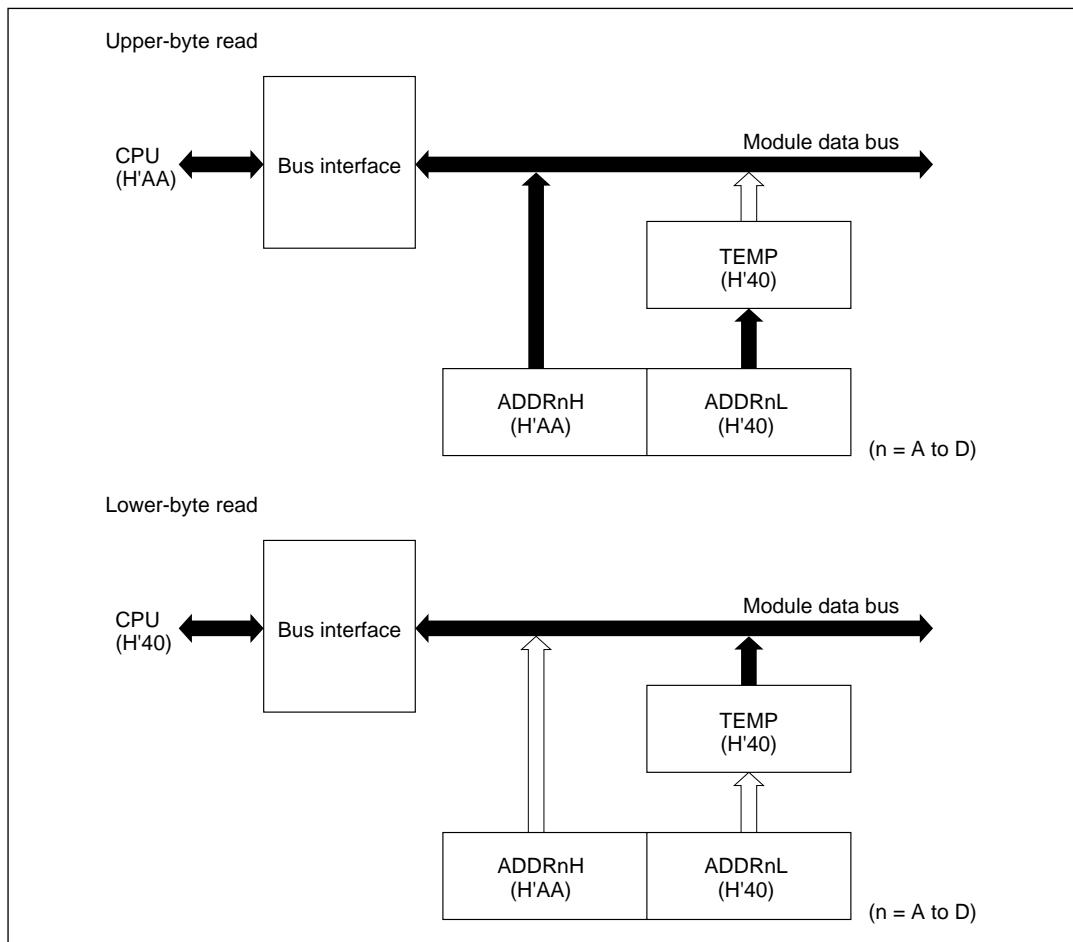


Figure 15-2 A/D Data Register Access Operation (Reading H'AA40)

## 15.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution. It has two operating modes: single mode and scan mode.

### 15.4.1 Single Mode (SCAN = 0)

Single mode should be selected when only one A/D conversion on one channel is required. A/D conversion starts when the ADST bit is set to 1 by software, or by external trigger input. The ADST bit remains set to 1 during A/D conversion and is automatically cleared to 0 when conversion ends.

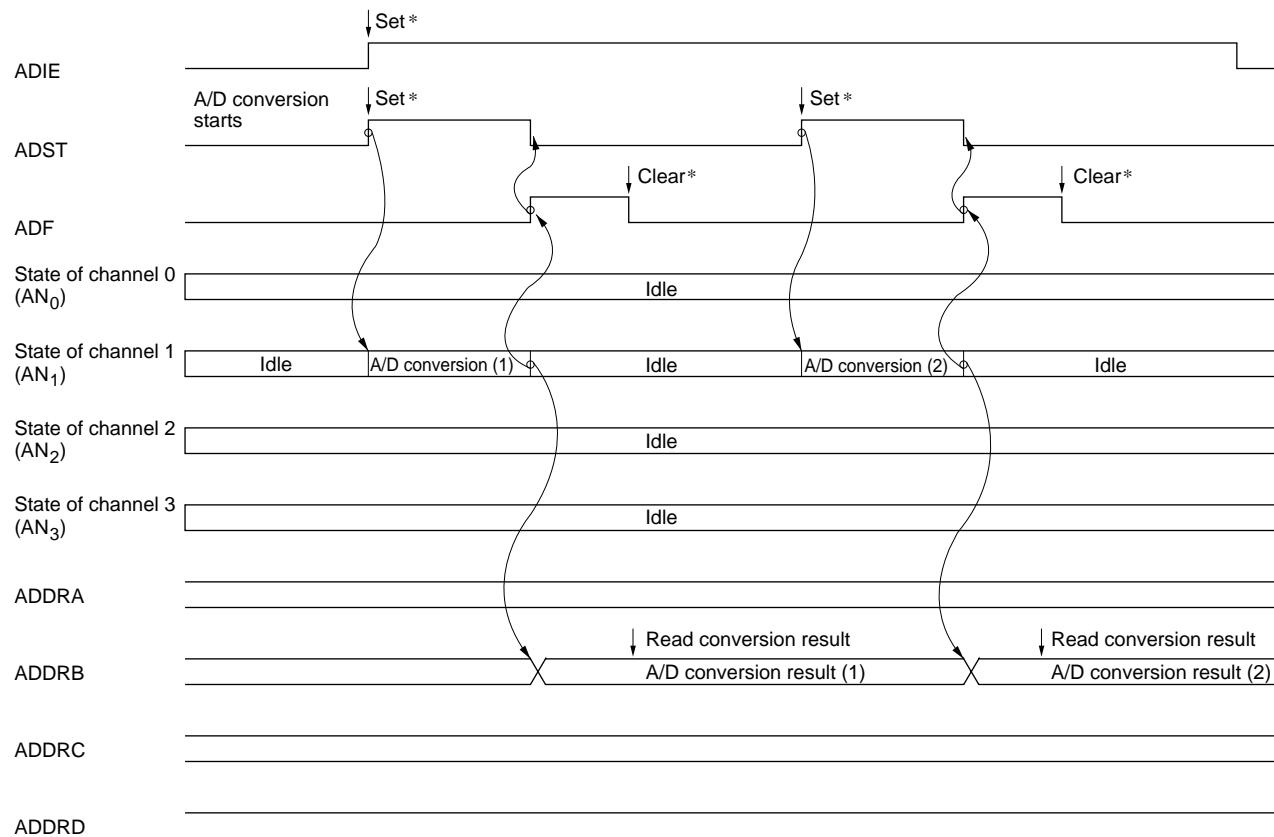
When conversion ends the ADF bit is set to 1. If the ADIE bit is also set to 1, an ADI interrupt is requested at this time. To clear the ADF flag to 0, first read ADCSR, then write 0 in ADF.

When the mode or analog input channel must be switched during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the mode or channel is changed.

Typical operations when channel 1 (AN<sub>1</sub>) is selected in single mode are described next. Figure 15-3 shows a timing diagram for this example.

1. Single mode is selected (SCAN = 0), input channel AN<sub>1</sub> is selected (CH2 = CH1 = 0, CH0 = 1), the A/D interrupt is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).
2. When A/D conversion is completed, the result is transferred into ADDR. At the same time the ADF flag is set to 1, the ADST bit is cleared to 0, and the A/D converter becomes idle.
3. Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.
4. The A/D interrupt handling routine starts.
5. The routine reads ADCSR, then writes 0 in the ADF flag.
6. The routine reads and processes the conversion result (ADDR).
7. Execution of the A/D interrupt handling routine ends.

After that, if the ADST bit is set to 1, A/D conversion starts again and steps 2 to 7 are repeated.



Note: \*Vertical arrows (↑) indicate instructions executed by software.

**Figure 15-3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)**

#### 15.4.2 Scan Mode (SCAN = 1)

Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit is set to 1 by software or external trigger input, A/D conversion starts on the first channel in the group ( $AN_0$  when  $CH2 = 0$ ,  $AN_4$  when  $CH2 = 1$ ). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel ( $AN_1$  or  $AN_5$ ) starts immediately. A/D conversion continues cyclically on the selected channels until the ADST bit is cleared to 0. The conversion results are transferred for storage into the A/D data registers corresponding to the channels.

When the mode or analog input channel selection must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1. A/D conversion will start again from the first channel in the group. The ADST bit can be set at the same time as the mode or channel selection is changed.

Typical operations when three channels in group 0 ( $AN_0$  to  $AN_2$ ) are selected in scan mode are described next. Figure 15-4 shows a timing diagram for this example.

1. Scan mode is selected (SCAN = 1), scan group 0 is selected (CH2 = 0), analog input channels  $AN_0$  to  $AN_2$  are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1).
2. When A/D conversion of the first channel ( $AN_0$ ) is completed, the result is transferred into ADDRA. Next, conversion of the second channel ( $AN_1$ ) starts automatically.
3. Conversion proceeds in the same way through the third channel ( $AN_2$ ).
4. When conversion of all selected channels ( $AN_0$  to  $AN_2$ ) is completed, the ADF flag is set to 1 and conversion of the first channel ( $AN_0$ ) starts again. If the ADIE bit is set to 1, an ADI interrupt is requested at this time.
5. Steps 2 to 4 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel ( $AN_0$ ).

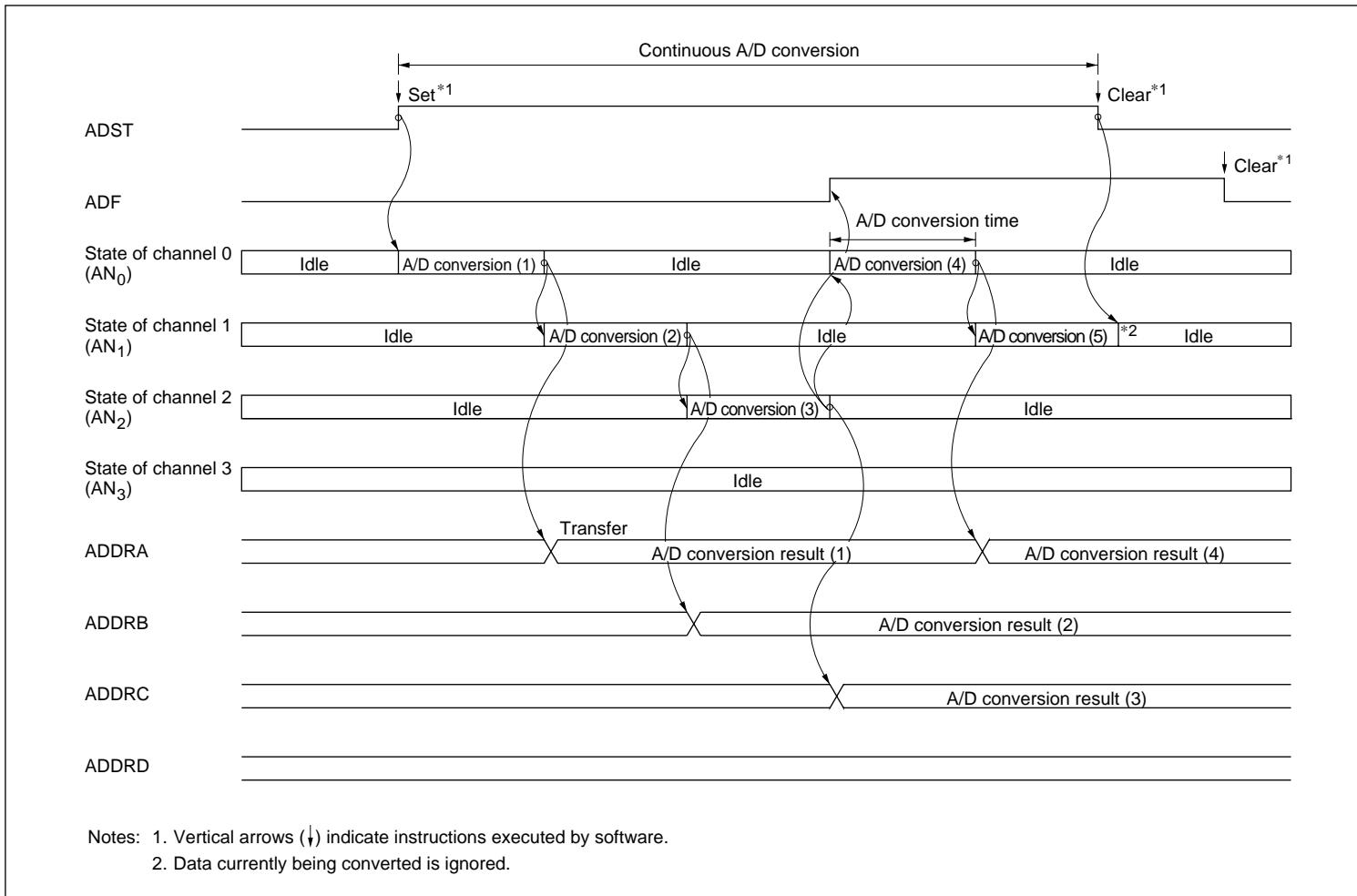


Figure 15-4 Example of A/D Converter Operation (Scan Mode, Channels AN<sub>0</sub> to AN<sub>2</sub> Selected)

### 15.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input at a time  $t_D$  after the ADST bit is set to 1, then starts conversion. Figure 15-5 shows the A/D conversion timing. Table 15-4 indicates the A/D conversion time.

As indicated in figure 15-5, the A/D conversion time includes  $t_D$  and the input sampling time. The length of  $t_D$  varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 15-4.

In scan mode, the values given in table 15-4 apply to the first conversion. In the second and subsequent conversions the conversion time is fixed at 256 states when CKS = 0 or 128 states when CKS = 1 (when  $\phi_P = \phi$ ).

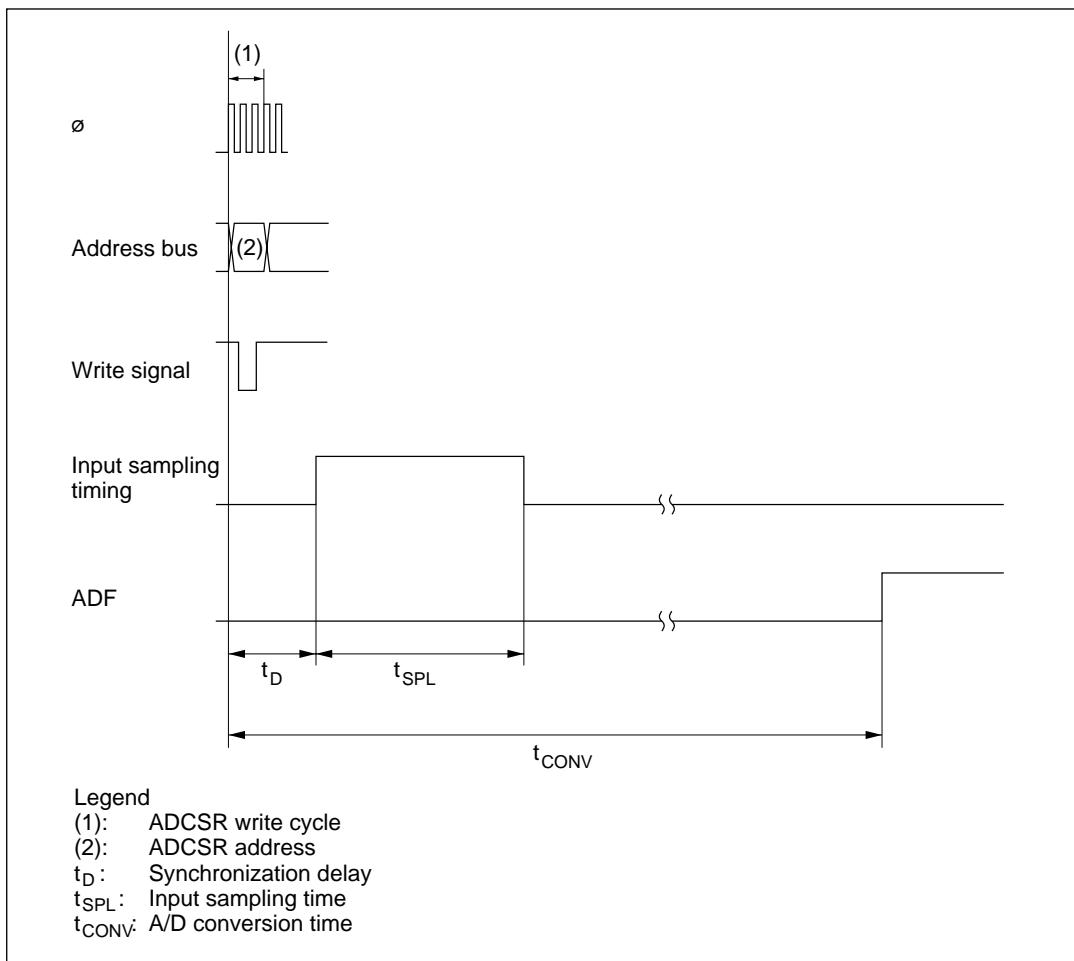


Figure 15-5 A/D Conversion Timing

**Table 15-4 A/D Conversion Time (Single Mode)**

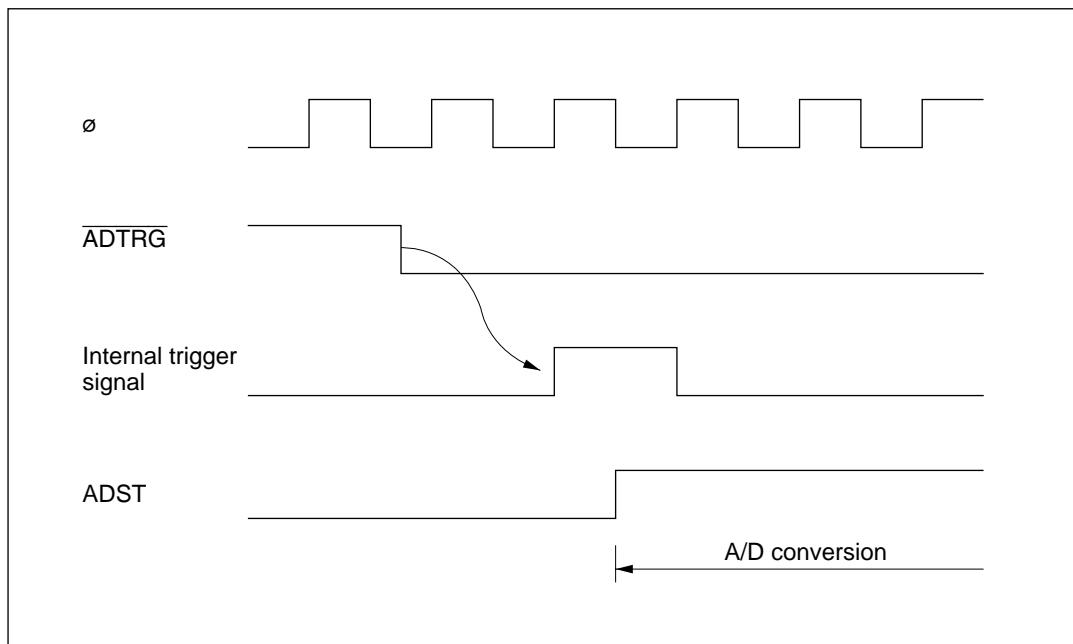
Symbol	CKS = 0			CKS = 1			
	Min	Typ	Max	Min	Typ	Max	
Synchronization delay	$t_D$	10	—	17	6	—	9
Input sampling time*	$t_{SPL}$	—	80	—	—	40	—
A/D conversion time*	$t_{CONV}$	259	—	266	131	—	134

Note: Values in the table are numbers of states.

\* Values for when  $\phi_P = \phi$ . When  $\phi_P = \phi/2$ , values are double those given in the table.

#### 15.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGE bit is set to 1 in ADCR, external trigger input is enabled at the ADTRG pin. A high-to-low transition at the ADTRG pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, in both single and scan modes, are the same as if the ADST bit had been set to 1 by software. Figure 15-6 shows the timing.



**Figure 15-6 External Trigger Input Timing**

## 15.5 Interrupts

The A/D converter generates an interrupt (ADI) at the end of A/D conversion. The ADI interrupt request can be enabled or disabled by the ADIE bit in ADCSR.

## 15.6 Application Notes

When using the A/D converter, note the following points:

**Analog Input Voltage Range:** During A/D conversion, the voltages input to the analog input pins  $AN_n$  should be in the range  $AV_{SS} \leq AN_n \leq AV_{ref}$ . ( $n = 0$  to  $7$ )

**$AV_{CC}$  and  $AV_{SS}$  Input Voltages:**  $AV_{SS}$  should be equal to  $V_{SS}$ . If the A/D converter is not used, the values should be  $AV_{CC} = V_{CC}$  and  $AV_{SS} = V_{SS}$

**$AV_{ref}$  Input Range:** The analog reference voltage input at the  $AV_{ref}$  pin should be in the range  $AV_{ref} \leq AV_{CC}$ . If the A/D converter is not used, the value should be  $AV_{ref} = V_{CC}$ .

# Section 16 D/A Converter

## 16.1 Overview

The H8/3437 Series has an on-chip D/A converter module with two channels.

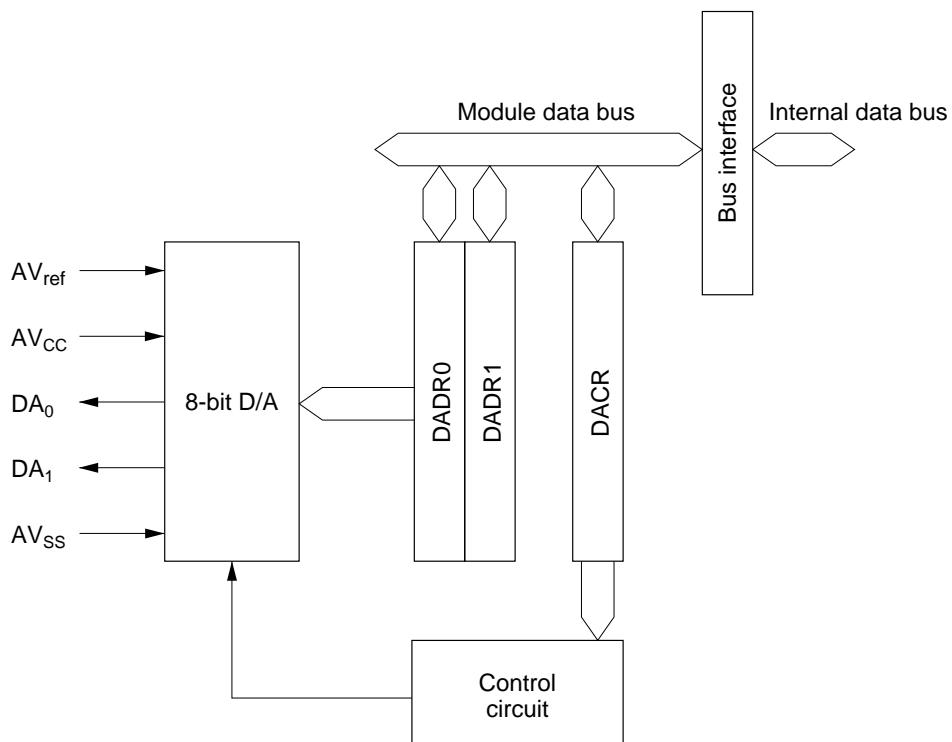
### 16.1.1 Features

Features of the D/A converter module are listed below.

- Eight-bit resolution
- Two-channel output
- Maximum conversion time: 10  $\mu$ s (with 20-pF load capacitance)
- Output voltage: 0 V to  $A V_{ref}$

### 16.1.2 Block Diagram

Figure 16-1 shows a block diagram of the D/A converter.



DACR: D/A control register

DADR0: D/A data register 0

DADR1: D/A data register 1

**Figure 16-1 D/A Converter Block Diagram**

### 16.1.3 Input and Output Pins

Table 16-1 lists the input and output pins used by the D/A converter module.

**Table 16-1 Input and Output Pins of D/A Converter Module**

Name	Abbreviation	I/O	Function
Reference voltage pin	AV <sub>ref</sub>	Input	Reference voltage for analog circuits
Analog supply voltage	AV <sub>CC</sub>	Input	Power supply and reference voltage for analog circuits
Analog ground	AV <sub>SS</sub>	Input	Ground and reference voltage for analog circuits
Analog output 0	DA <sub>0</sub>	Output	Analog output channel 0
Analog output 1	DA <sub>1</sub>	Output	Analog output channel 1

### 16.1.4 Register Configuration

Table 16-2 lists the three registers of the D/A converter module.

**Table 16-2 D/A Converter Registers**

Name	Abbreviation	R/W	Initial Value	Address
D/A data register 0	DADR0	R/W	H'00	H'FFF8
D/A data register 1	DADR1	R/W	H'00	H'FFF9
D/A control register	DACR	R/W	H'1F	H'FFFA

## 16.2 Register Descriptions

### 16.2.1 D/A Data Registers 0 and 1 (DADR0, DADR1)

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

D/A data registers 0 and 1 (DADR0 and DADR1) are 8-bit readable and writable registers that store data to be converted. When analog output is enabled, the value in the D/A data register is converted and output continuously at the analog output pin.

The D/A data registers are initialized to H'00 by a reset and in the standby modes.

### 16.2.2 D/A Control Register (DACR)

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

DACR is an 8-bit readable and writable register that controls the operation of the D/A converter module.

DACR is initialized to H'1F by a reset and in the standby modes.

**Bit 7—D/A Output Enable 1 (DAOE1):** Controls analog output from the D/A converter.

#### Bit 7

#### DAOE1      Description

---

0      Analog output at DA<sub>1</sub> is disabled.

---

1      D/A conversion is enabled on channel 1. Analog output is enabled at DA<sub>1</sub>.

**Bit 6—D/A Output Enable 0 (DAOE0):** Controls analog output from the D/A converter.

**Bit 6**  
**DAOE0      Description**

0	Analog output at DA <sub>0</sub> is disabled.
1	D/A conversion is enabled on channel 0. Analog output is enabled at DA <sub>0</sub> .

**Bit 5—D/A Enable (DAE):** Controls D/A conversion, in combination with bits DAOE0 and DAOE1. D/A conversion is controlled independently on channels 0 and 1 when DAE = 0. Channels 0 and 1 are controlled together when DAE = 1.

The decision to output the converted results is always controlled independently by DAOE0 and DAOE1.

**Bit 7      Bit 6      Bit 5**  
**DAOE1    DAOE0    DAE      D/A conversion**

0      0      —      Disabled on channels 0 and 1.

0      1      0      Enabled on channel 0.  
                    Disabled on channel 1.

0      1      1      Enabled on channels 0 and 1.

1      0      0      Disabled on channel 0.  
                    Enabled on channel 1.

1      0      1      Enabled on channels 0 and 1.

1      1      —      Enabled on channels 0 and 1.

When the DAE bit is set to 1, analog power supply current drain is the same as during A/D and D/A conversion, even if the DAOE0 and DAOE1 bits in DACR and the ADST bit in ADSCR are cleared to 0.

**Bits 4 to 0—Reserved:** These bits cannot be modified and are always read as 1.

## 16.3 Operation

The D/A converter module has two built-in D/A converter circuits that can operate independently.

D/A conversion is performed continuously whenever enabled by the D/A control register. When a new value is written in DADR0 or DADR1, conversion of the new value begins immediately. The converted result is output by setting the DAOE0 or DAOE1 bit to 1.

An example of conversion on channel 0 is given next. Figure 16-2 shows the timing.

- (1) Software writes the data to be converted in DADR0.
- (2) D/A conversion begins when the DAOE0 bit in DACR is set to 1. After a conversion delay, analog output appears at the DA0 pin. The output value is  $AV_{ref} \times (DADR0 \text{ value})/256$ . This output continues until a new value is written in DADR0 or the DAOE0 bit is cleared to 0.
- (3) If a new value is written in DADR0, conversion begins immediately. Output of the converted result begins after the conversion delay time.
- (4) When the DAOE0 bit is cleared to 0, DA0 becomes an input pin.

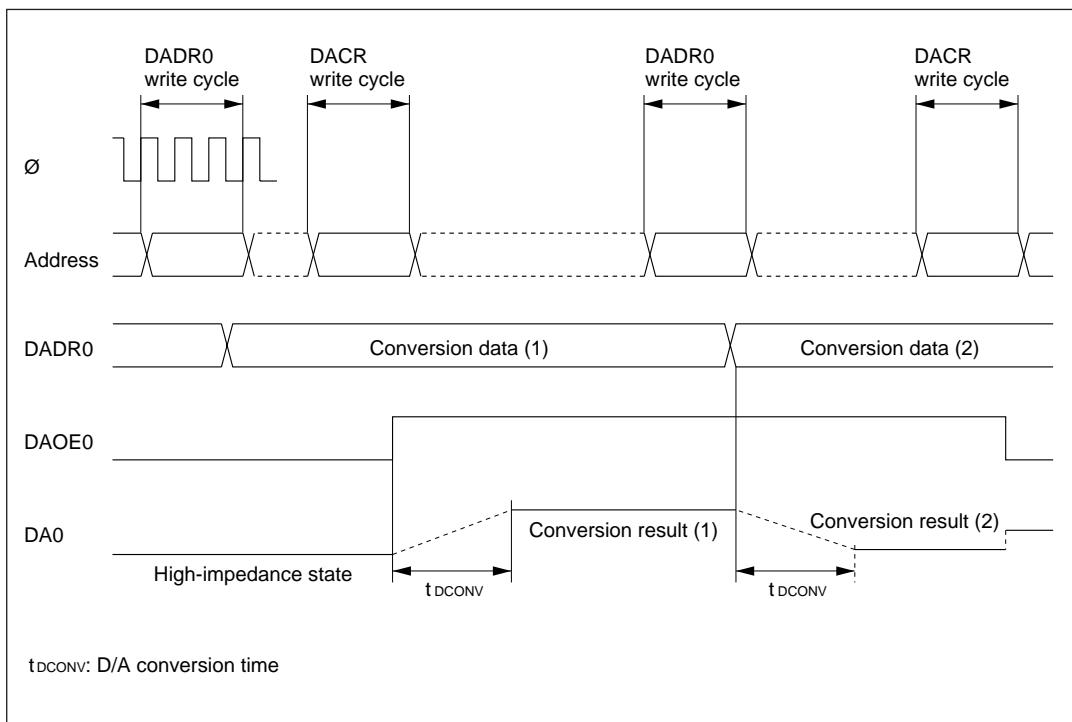


Figure 16-2 D/A Conversion (Example)

# Section 17 RAM

## 17.1 Overview

The H8/3437 and H8/3436 have 2 kbytes of on-chip static RAM. The H8/3434 has 1 kbyte. The RAM is 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'F780 to H'FF7F in the address space of the H8/3437 and H8/3436, and addresses H'FB80 to H'FF7F in the address space of the H8/3434. The RAME bit in the system control register (SYSCR) can enable or disable the on-chip RAM.

### 17.1.1 Block Diagram

Figure 17-1 is a block diagram of the on-chip RAM.

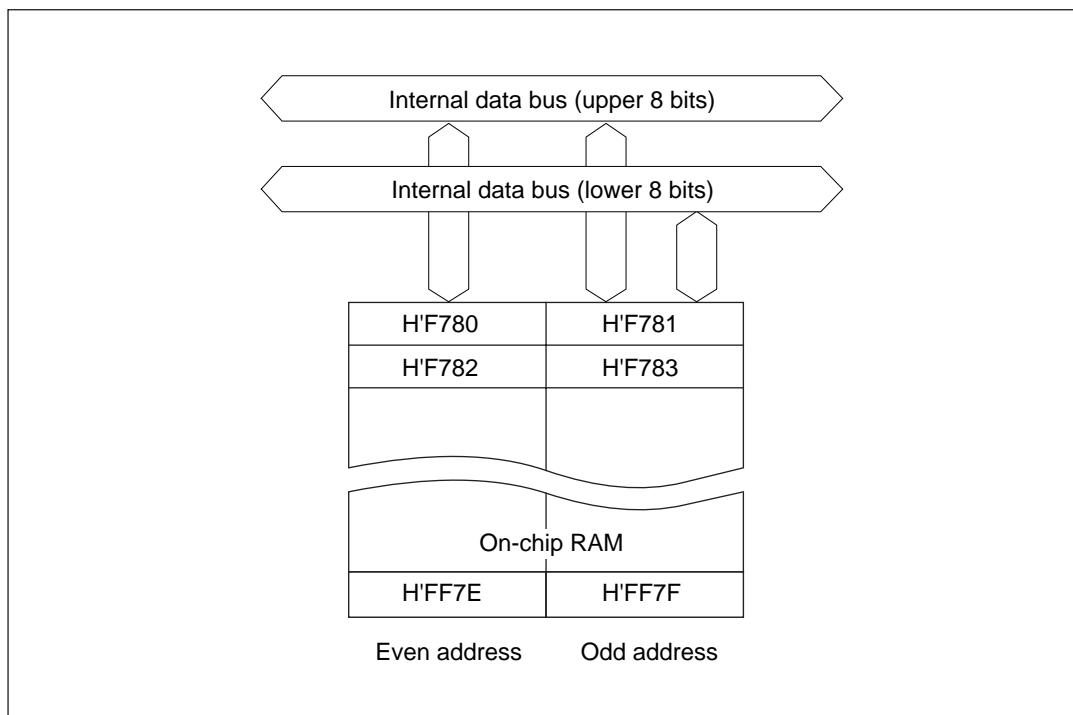


Figure 17-1 Block Diagram of On-Chip RAM (H8/3437)

### 17.1.2 RAM Enable Bit (RAME) in System Control Register (SYSCR)

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

The on-chip RAM is enabled or disabled by the RAME bit in SYSCR. See section 3.2, System Control Register, for the other SYSCR bits.

**Bit 0—RAM Enable (RAME):** This bit enables or disables the on-chip RAM. The RAME bit is initialized to 1 on the rising edge of the  $\overline{\text{RES}}$  signal. The RAME bit is not initialized in software standby mode.

Bit 0	
<b>RAME</b>	<b>Description</b>
0	On-chip RAM is disabled.
1	On-chip RAM is enabled. (Initial value)

## 17.2 Operation

### 17.2.1 Expanded Modes (Modes 1 and 2)

If the RAME bit is set to 1, accesses to addresses H'F780 to H'FF7F in the H8/3437 and H8/3436 and addresses H'FB80 to H'FF7F in the H8/3434 are directed to the on-chip RAM. If the RAME bit is cleared to 0, accesses to these addresses are directed to the external data bus.

### 17.2.2 Single-Chip Mode (Mode 3)

If the RAME bit is set to 1, accesses to addresses H'F780 to H'FF7F in the H8/3437 and H8/3436 and addresses H'FB80 to H'FF7F in the H8/3434 are directed to the on-chip RAM.

If the RAME bit is cleared to 0, the on-chip RAM data cannot be accessed. Attempted write access has no effect. Attempted read access always results in H'FF data being read.

# Section 18 ROM (Mask ROM version/ZTAT version)

## 18.1 Overview

The size of the on-chip ROM is 60 kbytes in the H8/3437, 48 kbytes in the H8/3436, and 32 kbytes in the H8/3434. The on-chip ROM is connected to the CPU via a 16-bit data bus. Both byte data and word data are accessed in two states, enabling rapid data transfer.

The on-chip ROM is enabled or disabled depending on the inputs at the mode pins (MD<sub>1</sub> and MD<sub>0</sub>). See table 18-1.

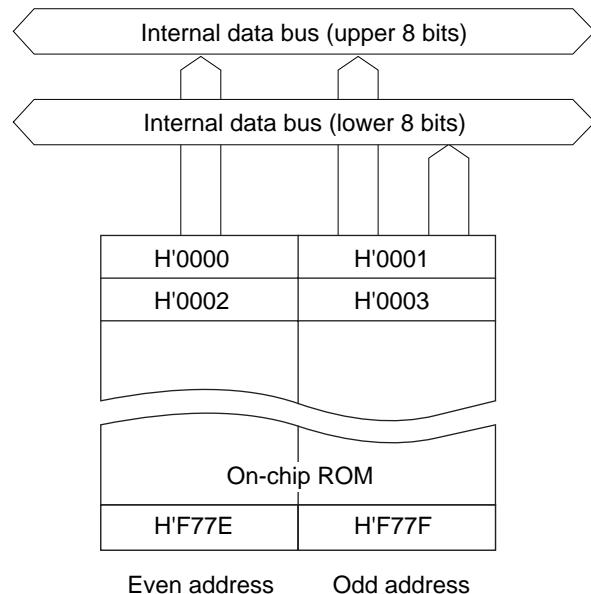
**Table 18-1 On-Chip ROM Usage in Each MCU Mode**

Mode	Mode Pins		On-chip ROM
	MD <sub>1</sub>	MD <sub>0</sub>	
Mode 1 (expanded mode)	0	1	Disabled (external addresses)
Mode 2 (expanded mode)	1	0	Enabled
Mode 3 (single-chip mode)	1	1	Enabled

The PROM versions (H8/3437 ZTAT and H8/3434 ZTAT) and flash-memory version (H8/3434 F-ZTAT) can be set to PROM mode and programmed with a general-purpose PROM programmer. In the H8/3437, the accessible ROM addresses are H'0000 to H'EF7F (61,312 bytes) in mode 2, and H'0000 to H'F77F (63,360 bytes) in mode 3. For details, see section 3, MCU Operating Modes and Address Space.

### 18.1.1 Block Diagram

Figure 18-1 is a block diagram of the on-chip ROM.



**Figure 18-1 Block Diagram of On-Chip ROM (H8/3437 Single-Chip Mode)**

## 18.2 PROM Mode (H8/3437, H8/3434)

### 18.2.1 PROM Mode Setup

In PROM mode the PROM versions of the H8/3437 and H8/3434 suspend the usual microcomputer functions to allow the on-chip PROM to be programmed. The programming method is the same as for the HN27C101.

To select PROM mode, apply the signal inputs listed in table 18-2.

**Table 18-2 Selection of PROM Mode**

Pin	Input
Mode pin MD <sub>1</sub>	Low
Mode pin MD <sub>0</sub>	Low
STBY pin	Low
Pins P6 <sub>3</sub> and P6 <sub>4</sub>	High

### 18.2.2 Socket Adapter Pin Assignments and Memory Map

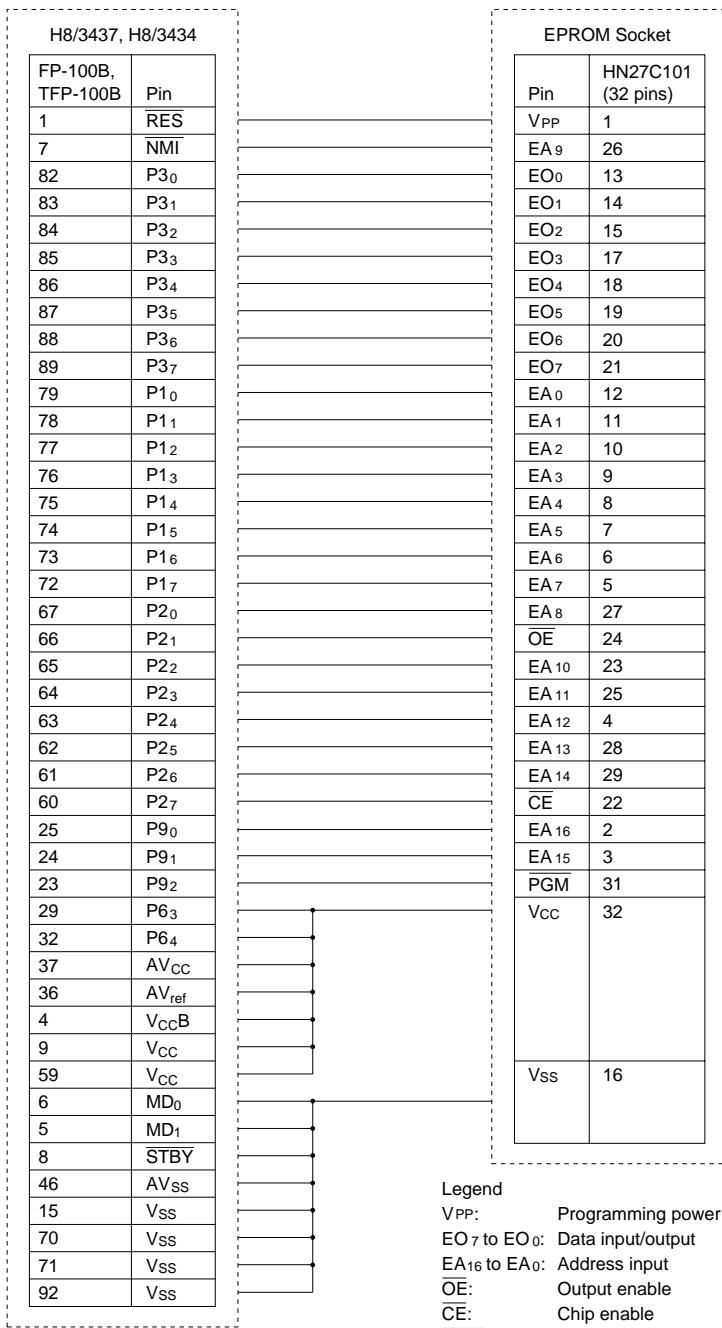
The H8/3437 and H8/3434 can be programmed with a general-purpose PROM programmer by using a socket adapter to change the pin-out to 32 pins. See table 18-3. The same socket adapter can be used for both the H8/3437 and H8/3434. Figure 18-2 shows the socket adapter pin assignments.

**Table 18-3 Socket Adapter**

Package	Socket Adapter
100-pin QFP	HS3437ESHS1H
100-pin TQFP	HS3437ESNS1H

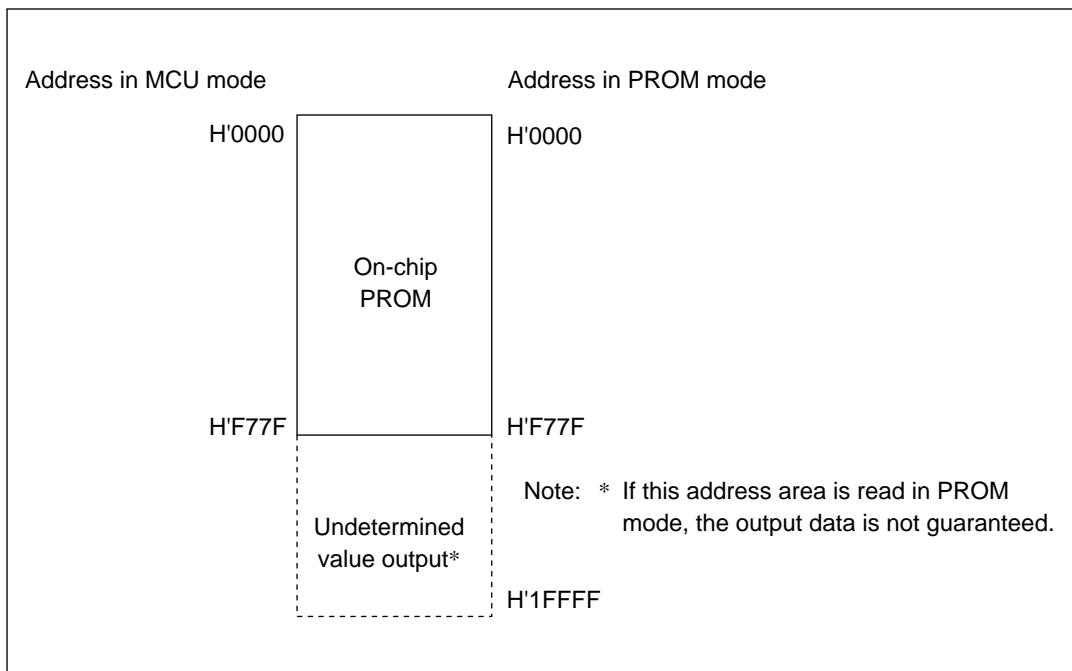
The PROM size is 60 kbytes for the H8/3437 and 32 kbytes for the H8/3434. Figures 18-3 and 18-4 show memory maps of the H8/3437 and H8/3434 in PROM mode. H'FF data should be specified for unused address areas in the on-chip PROM.

When programming with a PROM programmer, limit the program address range to H'0000 to H'F77F for the H8/3437 and H'0000 to H'7FFF for the H8/3434. Specify H'FF data for addresses H'F780 and above (H8/3437) or H'8000 and above (H8/3434). If these addresses are programmed by mistake, it may become impossible to program or verify the PROM data. The same problem may occur if an attempt is made to program the chip in page programming mode. Note that the PROM versions are one-time programmable (OTP) microcomputers, packaged in plastic packages, and cannot be reprogrammed.

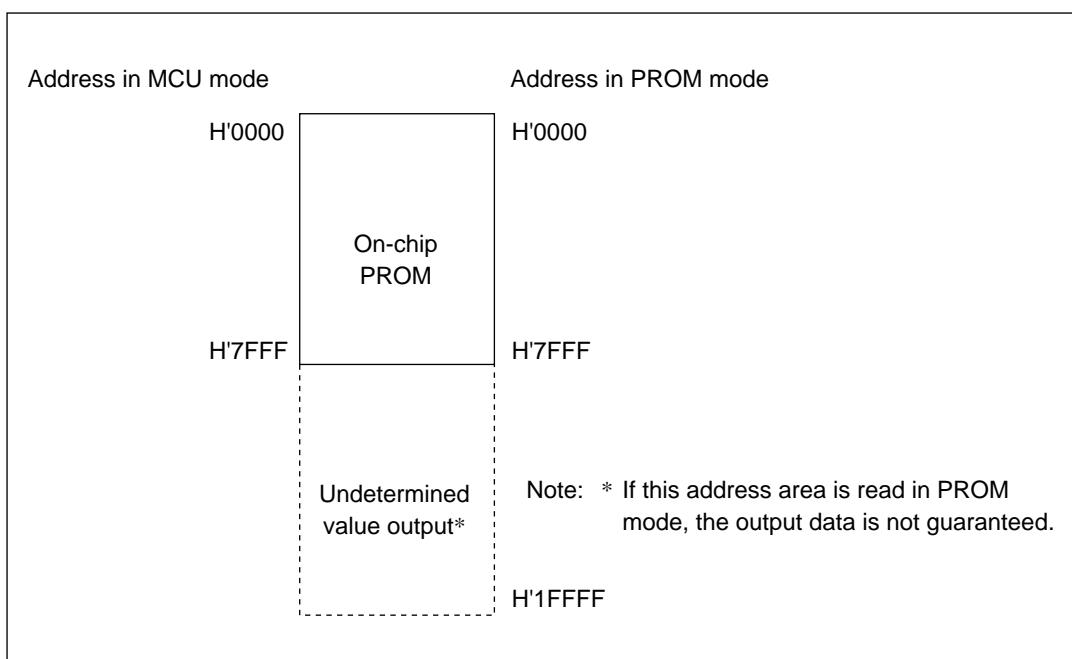


Note: All pins not listed in this figure should be left open.

Figure 18-2 Socket Adapter Pin Assignments



**Figure 18-3 H8/3437 Memory Map in PROM Mode**



**Figure 18-4 H8/3434 Memory Map in PROM Mode**

## 18.3 PROM Programming

The write, verify, and other sub-modes of the PROM mode are selected as shown in table 18-4.

**Table 18-4 Selection of Sub-Modes in PROM Mode**

Sub-Mode	CE	OE	PGM	V <sub>PP</sub>	V <sub>CC</sub>	EO <sub>7</sub> to EO <sub>0</sub>	EA <sub>16</sub> to EA <sub>0</sub>
Write	Low	High	Low	V <sub>PP</sub>	V <sub>CC</sub>	Data input	Address input
Verify	Low	Low	High	V <sub>PP</sub>	V <sub>CC</sub>	Data output	Address input
Programming inhibited	Low	Low	Low	V <sub>PP</sub>	V <sub>CC</sub>	High impedance	Address input
	Low	High	High				
	High	Low	Low				
	High	High	High				

The H8/3437 and H8/3434 PROM have the same standard read/write specifications as the HN27C101 EPROM. Page programming is not supported, however, so do not select page programming mode. PROM programmers that provide only page programming cannot be used. When selecting a PROM programmer, check that it supports a byte-at-a-time high-speed programming mode. Be sure to set the address range to H'0000 to HF77F for the H8/3437, and to H'0000 to H'7FFF for the H8/3434.

### 18.3.1 Programming and Verification

An efficient, high-speed programming procedure can be used to program and verify PROM data. This procedure programs data quickly without subjecting the chip to voltage stress and without sacrificing data reliability. It leaves the data H'FF in unused addresses.

Figure 18-5 shows the basic high-speed programming flowchart.

Tables 18-5 and 18-6 list the electrical characteristics of the chip in PROM mode. Figure 18-6 shows a program/verify timing chart.

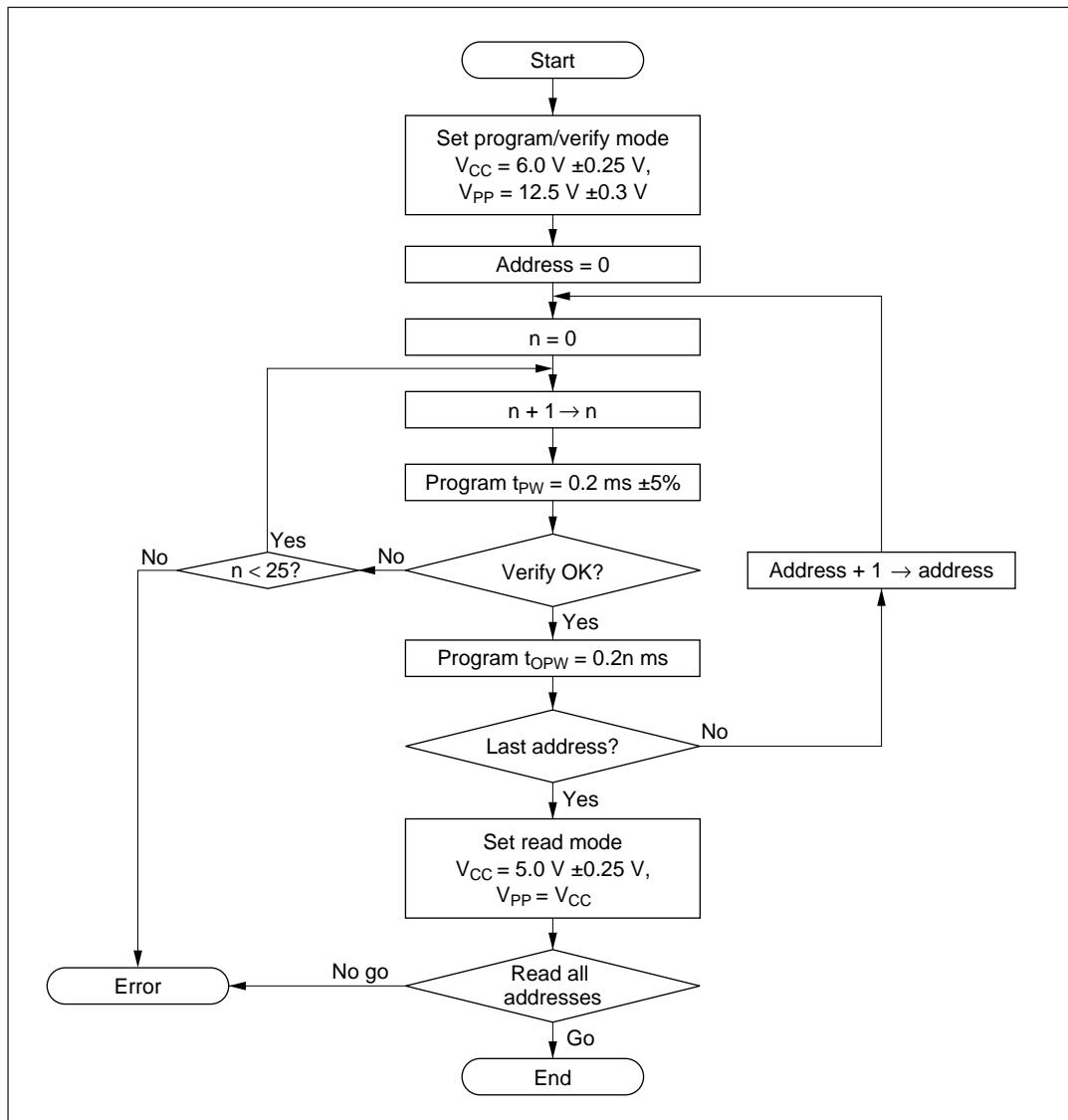


Figure 18-5 High-Speed Programming Flowchart

**Table 18-5 DC Characteristics**(when  $V_{CC} = 6.0 \text{ V} \pm 0.25 \text{ V}$ ,  $V_{PP} = 12.5 \text{ V} \pm 0.3 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Input high voltage	$EO_7 - EO_0$ , $EA_{16} - EA_0$ , $\overline{OE}$ , $CE$ , $PGM$	$V_{IH}$	2.4	—	$V_{CC} + 0.3$	V
Input low voltage	$EO_7 - EO_0$ , $EA_{16} - EA_0$ , $\overline{OE}$ , $CE$ , $PGM$	$V_{IL}$	-0.3	—	0.8	V
Output high voltage	$EO_7 - EO_0$	$V_{OH}$	2.4	—	—	V
Output low voltage	$EO_7 - EO_0$	$V_{OL}$	—	—	0.45	V
Input leakage current	$EO_7 - EO_0$ , $EA_{16} - EA_0$ , $\overline{OE}$ , $CE$ , $PGM$	$ I_{LIL} $	—	—	2	$\mu\text{A}$
$V_{CC}$ current	$I_{CC}$	—	—	40	mA	
$V_{PP}$ current	$I_{PP}$	—	—	40	mA	

**Table 18-6 AC Characteristics**(when  $V_{CC} = 6.0 \text{ V} \pm 0.25 \text{ V}$ ,  $V_{PP} = 12.5 \text{ V} \pm 0.3 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Address setup time	$t_{AS}$	2	—	—	$\mu\text{s}$	See figure 18-6*
$\overline{OE}$ setup time	$t_{OES}$	2	—	—	$\mu\text{s}$	
Data setup time	$t_{DS}$	2	—	—	$\mu\text{s}$	
Address hold time	$t_{AH}$	0	—	—	$\mu\text{s}$	
Data hold time	$t_{DH}$	2	—	—	$\mu\text{s}$	
Data output disable time	$t_{DF}$	—	—	130	ns	
$V_{PP}$ setup time	$t_{VPS}$	2	—	—	$\mu\text{s}$	
Program pulse width	$t_{PW}$	0.19	0.20	0.21	ms	
$OE$ pulse width for overwrite-programming	$t_{OPW}$	0.19	—	5.25	ms	
$V_{CC}$ setup time	$t_{VCS}$	2	—	—	$\mu\text{s}$	
$CE$ setup time	$t_{CES}$	2	—	—	$\mu\text{s}$	
Data output delay time	$t_{OE}$	0	—	150	ns	

Note: \* Input pulse level: 0.8 V to 2.2 V

Input rise/fall time  $\leq 20 \text{ ns}$ 

Timing reference levels: input—1.0 V, 2.0 V; output—0.8 V, 2.0 V

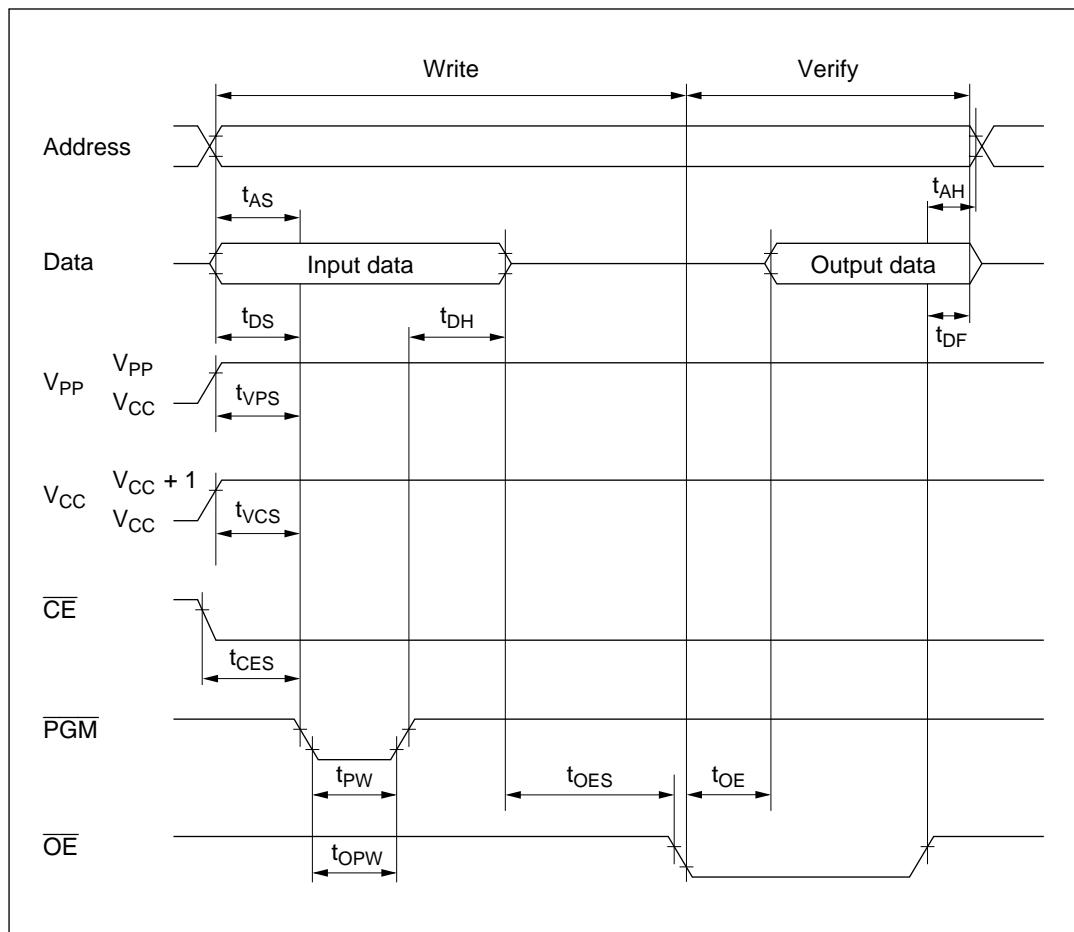


Figure 18-6 PROM Program/Verify Timing

### 18.3.2 Notes on Programming

**(1) Program with the specified voltages and timing. The programming voltage ( $V_{PP}$ ) is 12.5 V.**

**Caution:** Applied voltages in excess of the specified values can permanently destroy the chip. Be particularly careful about the PROM programmer's overshoot characteristics.

If the PROM programmer is set to HN27C101 specifications,  $V_{PP}$  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 programmer, 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.

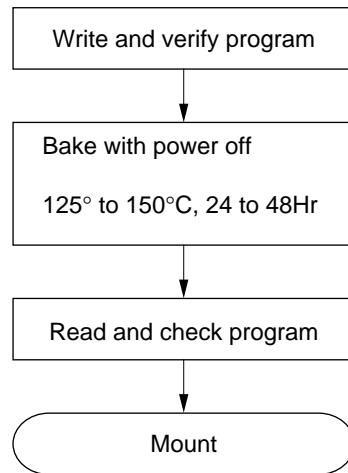
**(4) Page programming is not supported.** Do not select page programming mode.

**(5) The H8/3437 PROM size is 60 kbytes. The H8/3434 PROM size is 32 kbytes.** Set the address range to H'0000 to H'F77F for the H8/3437, and to H'0000 to H'7FFF for the H8/3434. When programming, specify H'FF data for unused address areas (H'F780 to H'1FFFF in the H8/3437, H'8000 to H'1FFFF in the H8/3434).

### 18.3.3 Reliability of Programmed 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 18-7 shows the recommended screening procedure.



**Figure 18-7 Recommended Screening Procedure**

If a series of write errors occurs while the same PROM programmer is in use, stop programming and check the PROM programmer and socket adapter for defects.

Please inform Hitachi of any abnormal conditions noted during programming or in screening of program data after high-temperature baking.

# Section 19 ROM (Flash memory 32 kbytes version)

## 19.1 Flash Memory Overview

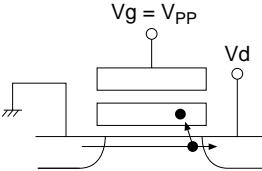
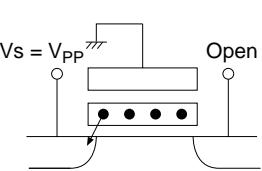
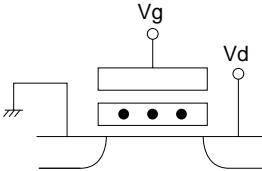
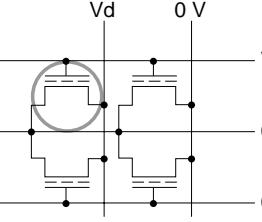
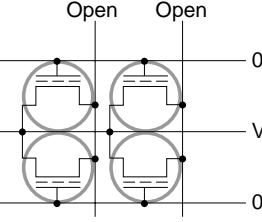
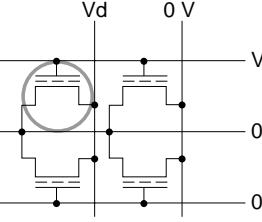
### 19.1.1 Flash Memory Operating Principle

Table 19-1 illustrates the principle of operation of the H8/3434F's on-chip flash memory.

Like EPROM, flash memory is programmed by applying a high gate-to-drain voltage that draws hot electrons generated in the vicinity of the drain into a floating gate. The threshold voltage of a programmed memory cell is therefore higher than that of an erased cell. Cells are erased by grounding the gate and applying a high voltage to the source, causing the electrons stored in the floating gate to tunnel out. After erasure, the threshold voltage drops. A memory cell is read like an EPROM cell, by driving the gate to the high level and detecting the drain current, which depends on the threshold voltage. Erasing must be done carefully, because if a memory cell is overerased, its threshold voltage may become negative, causing the cell to operate incorrectly.

Section 19.4.6 shows an optimal erase control flowchart and sample program.

**Table 19-1 Principle of Memory Cell Operation**

	Program	Erase	Read
Memory cell			
Memory array			

### 19.1.2 Mode Programming and Flash Memory Address Space

As its on-chip ROM, the H8/3434F has 32 kbytes of flash memory. The flash memory is connected to the CPU by a 16-bit data bus. The CPU accesses both byte data and word data in two states.

The H8/3434F's flash memory is assigned to addresses H'0000 to H'7FFF. The mode pins enable either on-chip flash memory or external memory to be selected for this area. Table 19-2 summarizes the mode pin settings and usage of the memory area.

**Table 19-2 Mode Pin Settings and Flash Memory Area**

Mode Pin Setting			
Mode	MD <sub>1</sub>	MD <sub>0</sub>	Memory Area Usage
Mode 0	0	0	Illegal setting
Mode 1	0	1	External memory area
Mode 2	1	0	On-chip flash memory area
Mode 3	1	1	On-chip flash memory area

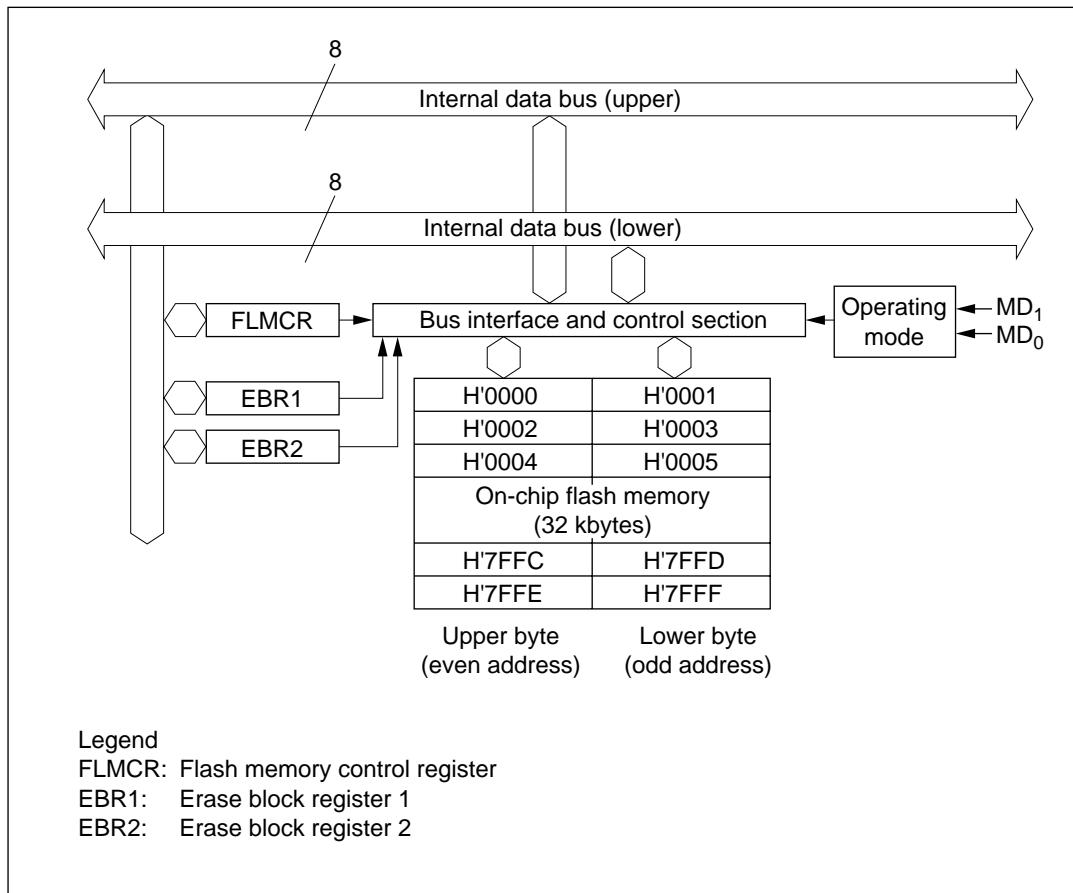
### 19.1.3 Features

Features of the flash memory are listed below.

- Five flash memory operating modes  
The flash memory has five operating modes: program mode, program-verify mode, erase mode, erase-verify mode, and prewrite-verify mode.
- Block erase designation  
Blocks to be erased in the flash memory address space can be selected by bit settings. The address space includes a large-block area (four blocks with sizes from 4 kbytes to 8 kbytes) and a small-block area (eight blocks with sizes from 128 bytes to 1 kbyte).
- Program and erase time  
Programming one byte of flash memory typically takes 50 µs. Erasing all blocks (32 kbytes) typically takes 1 s.
- Erase-program cycles  
Flash memory contents can be erased and reprogrammed up to 100 times.
- On-board programming modes  
These modes can be used to program, erase, and verify flash memory contents. There are two modes: boot mode and user programming mode.
- Automatic bit-rate alignment  
In boot-mode data transfer, the H8/3434F aligns its bit rate automatically to the host bit rate (maximum 9600 bps).
- Flash memory emulation by RAM  
Part of the RAM area can be overlapped onto flash memory, to emulate flash memory updates in real time.
- PROM mode  
As an alternative to on-board programming, the flash memory can be programmed and erased in PROM mode, using a general-purpose PROM programmer. Program, erase, verify, and other specifications are the same as for HN28F101 standard flash memory.

#### 19.1.4 Block Diagram

Figure 19-1 shows a block diagram of the flash memory.



**Figure 19-1 Flash Memory Block Diagram**

## 19.1.5 Input/Output Pins

Flash memory is controlled by the pins listed in table 19-3.

**Table 19-3 Flash Memory Pins**

Pin Name	Abbreviation	Input/Output	Function
Programming power	FV <sub>PP</sub>	Power supply	Apply 12.0 V
Mode 1	MD <sub>1</sub>	Input	H8/3434F operating mode setting
Mode 0	MD <sub>0</sub>	Input	H8/3434F operating mode setting
Transmit data	TxD <sub>1</sub>	Output	SCI1 transmit data output
Receive data	RxD <sub>1</sub>	Input	SCI1 receive data input

The transmit data and receive data pins are used in boot mode.

## 19.1.6 Register Configuration

The flash memory is controlled by the registers listed in table 19-4.

**Table 19-4 Flash Memory Registers**

Name	Abbreviation	R/W	Initial Value	Address
Flash memory control register	FLMCR	R/W*2	H'00*2	H'FF80
Erase block register 1	EBR1	R/W*2	H'F0*2	H'FF82
Erase block register 2	EBR2	R/W*2	H'00*2	H'FF83
Wait-state control register*1	WSCR	R/W	H'08	H'FFC2

Notes: Registers FLMCR, EBR1, and EBR2 are only valid when writing to or erasing flash memory, and can only be accessed while 12 V is being applied to the FV<sub>PP</sub> pin.

When 12 V is not applied to the FV<sub>PP</sub> pin, in mode 2 addresses H'FF80 to H'FF83 are external address space, and in mode 3 these addresses cannot be modified and always read H'FF.

1. The wait-state control register controls the insertion of wait states by the wait-state controller, frequency division of clock signals for the on-chip supporting modules by the clock pulse generator, and emulation of flash-memory updates by RAM in on-board programming mode.
2. In modes 2 and 3 (on-chip flash memory enabled), the initial value is H'00 for FLMCR and EBR2, and H'F0 for EBR1. In mode 1 (on-chip flash memory disabled), these registers cannot be modified and always read H'FF.

## 19.2 Flash Memory Register Descriptions

### 19.2.1 Flash Memory Control Register (FLMCR)

FLMCR is an 8-bit register that controls the flash memory operating modes. Transitions to program mode, erase mode, program-verify mode, and erase-verify mode are made by setting bits in this register. FLMCR is initialized to H'00 by a reset, in the standby modes, and when 12 V is not applied to FV<sub>PP</sub>. When 12 V is applied to the FV<sub>PP</sub> pin, a reset or entry to a standby mode initializes FLMCR to H'80.

Bit	7	6	5	4	3	2	1	0
	V <sub>PP</sub>	—	—	—	EV	PV	E	P
Initial value*	0	0	0	0	0	0	0	0
Read/Write	R	—	—	—	R/W*	R/W*	R/W*	R/W*

Note: \* The initial value is H'00 in modes 2 and 3 (on-chip flash memory enabled). In mode 1 (on-chip flash memory disabled), this register cannot be modified and always reads H'FF. For information on accessing this register, refer to in section 19.7, Flash Memory Programming and Erasing Precautions (11).

**Bit 7—Programming Power (V<sub>PP</sub>):** This status flag indicates that 12 V is applied to the FV<sub>PP</sub> pin. Refer to section 19.7, Flash Memory Programming and Erasing Precautions (5), for details on use.

#### Bit 7

V <sub>PP</sub>	Description	
0	Cleared when 12 V is not applied to FV <sub>PP</sub>	(Initial value)
1	Set when 12 V is applied to FV <sub>PP</sub>	

**Bits 6 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Erase-Verify Mode (EV):**<sup>\*1</sup> Selects transition to or exit from erase-verify mode.

#### Bit 3

EV	Description	
0	Exit from erase-verify mode	(Initial value)
1	Transition to erase-verify mode	

**Bit 2—Program-Verify Mode (PV):**<sup>\*1</sup> Selects transition to or exit from program-verify mode.

#### Bit 2

PV	Description	
0	Exit from program-verify mode	(Initial value)
1	Transition to program-verify mode	

**Bit 1—Erase Mode (E):**\*<sup>1, \*2</sup> Selects transition to or exit from erase mode.

**Bit 1**

**E**      **Description**

0	Exit from erase mode	(Initial value)
1	Transition to erase mode	

**Bit 0—Program Mode (P):**\*<sup>1, \*2</sup> Selects transition to or exit from program mode.

**Bit 0**

**P**      **Description**

0	Exit from program mode	(Initial value)
1	Transition to program mode	

Notes: 1. Do not set two or more of these bits simultaneously. Do not release or shut off the V<sub>CC</sub> or V<sub>PP</sub> power supply when these bits are set.

2. Set the P or E bit according to the instructions given in section 19.4, Programming and Erasing Flash Memory.

Set the watchdog timer beforehand to make sure that these bits do not remain set for longer than the specified times.

For notes on use, see section 19.7, Flash Memory Programming and Erasing Precautions.

### 19.2.2 Erase Block Register 1 (EBR1)

EBR1 is an 8-bit register that designates large flash-memory blocks for programming and erasure. EBR1 is initialized to H'F0 by a reset, in the standby modes, and when 12 V is not applied to FV<sub>PP</sub> pin. When a bit in EBR1 is set to 1, the corresponding block is selected and can be programmed and erased. Figure 19-2 and table 19-6 show details of a block map.

Bit	7	6	5	4	3	2	1	0
Initial value*	—	—	—	—	LB3	LB2	LB1	LB0
Read/Write	—	—	—	—	R/W*	R/W*	R/W*	R/W*

Note: \* The initial value is H'F0 in modes 2 and 3 (on-chip ROM enabled). In mode 1 (on-chip ROM disabled), this register cannot be modified and always reads H'FF.

For information on accessing this register, refer to in section 19.7, Flash Memory Programming and Erasing Precautions (11).

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

**Bits 3 to 0—Large Block 3 to 0 (LB3 to LB0):** These bits select large blocks (LB3 to LB0) to be programmed and erased.

Bits 3 to 0	LB3 to LB0	Description	
0	Block (LB3 to LB0) is not selected		(Initial value)
1	Block (LB3 to LB0) is selected		

### 19.2.3 Erase Block Register 2 (EBR2)

EBR2 is an 8-bit register that designates small flash-memory blocks for programming and erasure. EBR2 is initialized to H'00 by a reset, in the standby modes, and when 12 V is not applied to FV<sub>PP</sub> pin. When a bit in EBR2 is set to 1, the corresponding block is selected and can be programmed and erased. Figure 19-2 and table 19-6 show a block map.

Bit	7	6	5	4	3	2	1	0
	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Initial value*	0	0	0	0	0	0	0	0
Read/Write	R/W*							

Note: \* The initial value is H'00 in modes 2 and 3 (on-chip ROM enabled). In mode 1 (on-chip ROM disabled), this register cannot be modified and always reads H'FF.

For information on accessing this register, refer to in section 19.7, Flash Memory Programming and Erasing Precautions (11).

**Bits 7 to 0—Small Block 7 to 0 (SB7 to SB0):** These bits select small blocks (SB7 to SB0) to be programmed and erased.

Bits 7 to 0	SB7 to SB0	Description	
0	Block (SB7 to SB0) is not selected		(Initial value)
1	Block (SB7 to SB0) is selected		

## 19.2.4 Wait-State Control Register (WSCR)

WSCR is an 8-bit readable/writable register that enables flash-memory updates to be emulated in RAM. It also controls frequency division of clock signals supplied to the on-chip supporting modules and insertion of wait states by the wait-state controller.

WSCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Bit	7	6	5	4	3	2	1	0
	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	0	0	0	0	1	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Bits 7 and 6—RAM Select and RAM0 (RAMS and RAM0):** These bits are used to reassign an area to RAM (see table 19-5). These bits are write-enabled and their initial value is 0. They are initialized by a reset and in hardware standby mode. They are not initialized in software standby mode.

If only one of bits 7 and 6 is set, part of the RAM area can be overlapped onto the small-block flash memory area. In that case, access is to RAM, not flash memory, and all flash memory blocks are write/erase-protected (emulation protect<sup>\*1</sup>). In this state, the mode cannot be changed to program or erase mode, even if the P bit or E bit in the flash memory control register (FLMCR) is set (although verify mode can be selected). Therefore, clear both of bits 7 and 6 before programming or erasing the flash memory area.

If both of bits 7 and 6 are set, part of the RAM area can be overlapped onto the small-block flash memory area, but this overlapping begins only when an interrupt signal is input while 12 V is being applied to the FV<sub>PP</sub> pin. Up until that point, flash memory is accessed. Use this setting for interrupt handling while flash memory is being programmed or erased.<sup>\*2</sup>

**Table 19-5 RAM Area Reassignment<sup>\*3</sup>**

Bit 7 RAMS	Bit 6 RAM0	RAM Area	ROM Area
0	0	None	—
0	1	H'FC80 to H'FCFF	H'0080 to H'00FF
1	0	H'FC80 to H'FD7F	H'0080 to H'017F
1	1	H'FC00 to H'FC7F	H'0000 to H'007F

**Bit 5—Clock Double (CKDBL):** Controls frequency division of clock signals supplied to the on-chip supporting modules. For details, see section 6, Clock Pulse Generator.

**Bit 4—Reserved:** This bit is reserved, but it can be written and read. Its initial value is 0.

**Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1, WMS0)**

**Bits 1 and 0—Wait Count 1 and 0 (WC1, WC0)**

These bits control insertion of wait states by the wait-state controller. For details, see section 5, Wait-State Controller.

Notes:

1. For details on emulation protect, see section 19.4.8, Protect Modes.
2. For details on interrupt handling during programming and erasing of flash memory, see section 19.4.9, Interrupt Handling during Flash Memory Programming and Erasing.
3. RAM area that overlaps flash memory.

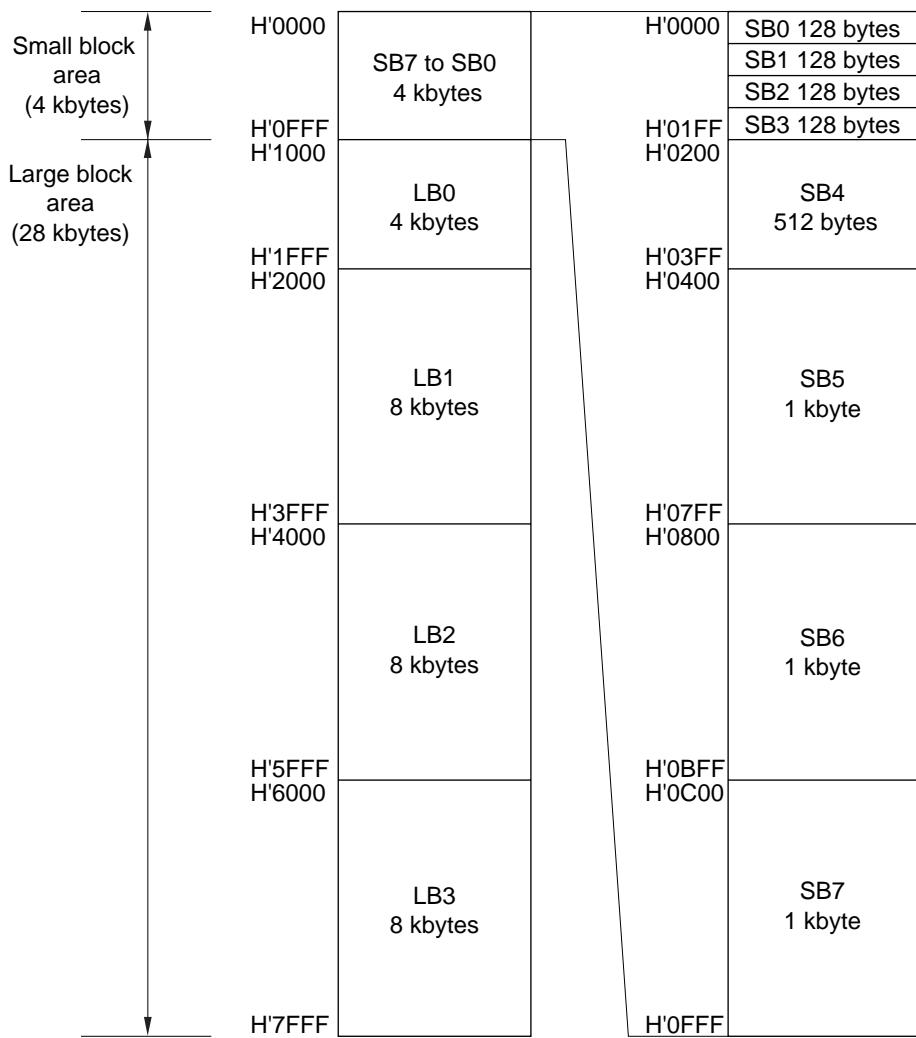


Figure 19-2 Erase Block Map

**Table 19-6 Erase Blocks and Corresponding Bits**

Register	Bit	Block	Address	Size
EBR1	0	LB0	H'1000 to H'1FFF	4 kbytes
	1	LB1	H'2000 to H'3FFF	8 kbytes
	2	LB2	H'4000 to H'5FFF	8 kbytes
	3	LB3	H'6000 to H'7FFF	8 kbytes
EBR2	0	SB0	H'0000 to H'007F	128 bytes
	1	SB1	H'0080 to H'00FF	128 bytes
	2	SB2	H'0100 to H'017F	128 bytes
	3	SB3	H'0180 to H'01FF	128 bytes
	4	SB4	H'0200 to H'03FF	512 bytes
	5	SB5	H'0400 to H'07FF	1 kbyte
	6	SB6	H'0800 to H'0BFF	1 kbyte
	7	SB7	H'0C00 to H'0FFF	1 kbyte

### 19.3 On-Board Programming Modes

When an on-board programming mode is selected, the on-chip flash memory can be programmed, erased, and verified. There are two on-board programming modes: boot mode, and user program mode. These modes are selected by inputs at the mode pins ( $MD_1$  and  $MD_0$ ) and  $FV_{PP}$  pin. Table 19-7 indicates how to select the on-board programming modes. For details on applying voltage  $V_{PP}$ , refer to section 19.7, Flash Memory Programming and Erasing Precautions (5).

**Table 19-7 On-Board Programming Mode Selection**

Mode Selections	$FV_{PP}$	$MD_1$	$MD_0$	Notes
Boot mode	Mode 2	12 V*	12 V*	0: $V_{IL}$
	Mode 3	12 V*	1	1: $V_{IH}$
User program mode	Mode 2	1	0	
	Mode 3	1	1	

Note: \* For details on the timing of 12 V application, see notes 6 to 8 in the Notes on Use of Boot Mode at the end of this section.

In boot mode, the mode control register (MDCR) can be used to monitor the mode (mode 2 or 3) in the same way as in normal mode.

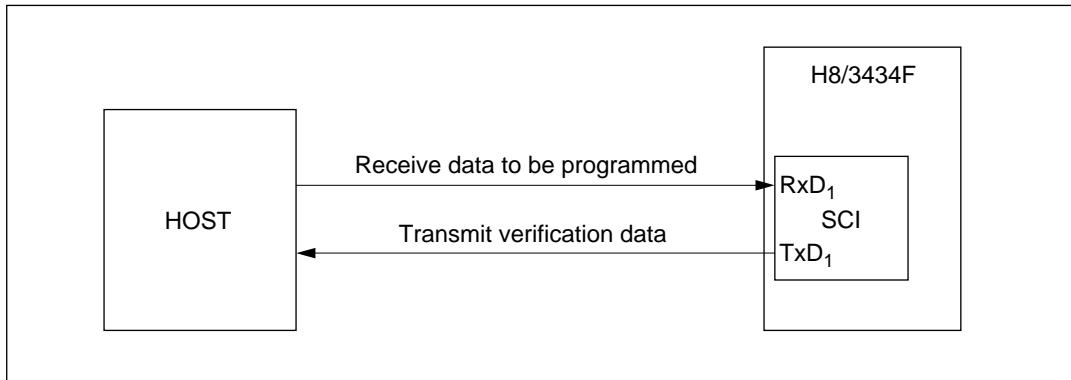
Example: Set the mode pins for mode 2 boot mode ( $MD_1 = 12 V$ ,  $MD_0 = 0 V$ ).

If the mode select bits of MDCR are now read, they will indicate mode 2 ( $MDS1 = 1$ ,  $MDS0 = 0$ ).

### 19.3.1 Boot Mode

To use boot mode, a user program for programming and erasing the flash memory must be provided in advance on the host machine (which may be a personal computer). Serial communication interface channel 1 is used in asynchronous mode (see figure 19-3). If the H8/3434F is placed in boot mode, after it comes out of reset, a built-in boot program is activated. This program starts by measuring the low period of data transmitted from the host and setting the bit rate register (BRR) accordingly. The H8/3434F's built-in serial communication interface (SCI) can then be used to download the user program from the host machine. The user program is stored in on-chip RAM.

After the program has been stored, execution branches to address H'FBE0 in the on-chip RAM, and the program stored on RAM is executed to program and erase the flash memory.



**Figure 19-3 Boot-Mode System Configuration**

## Boot-Mode Execution Procedure:

Figure 19-4 shows the boot-mode execution procedure.

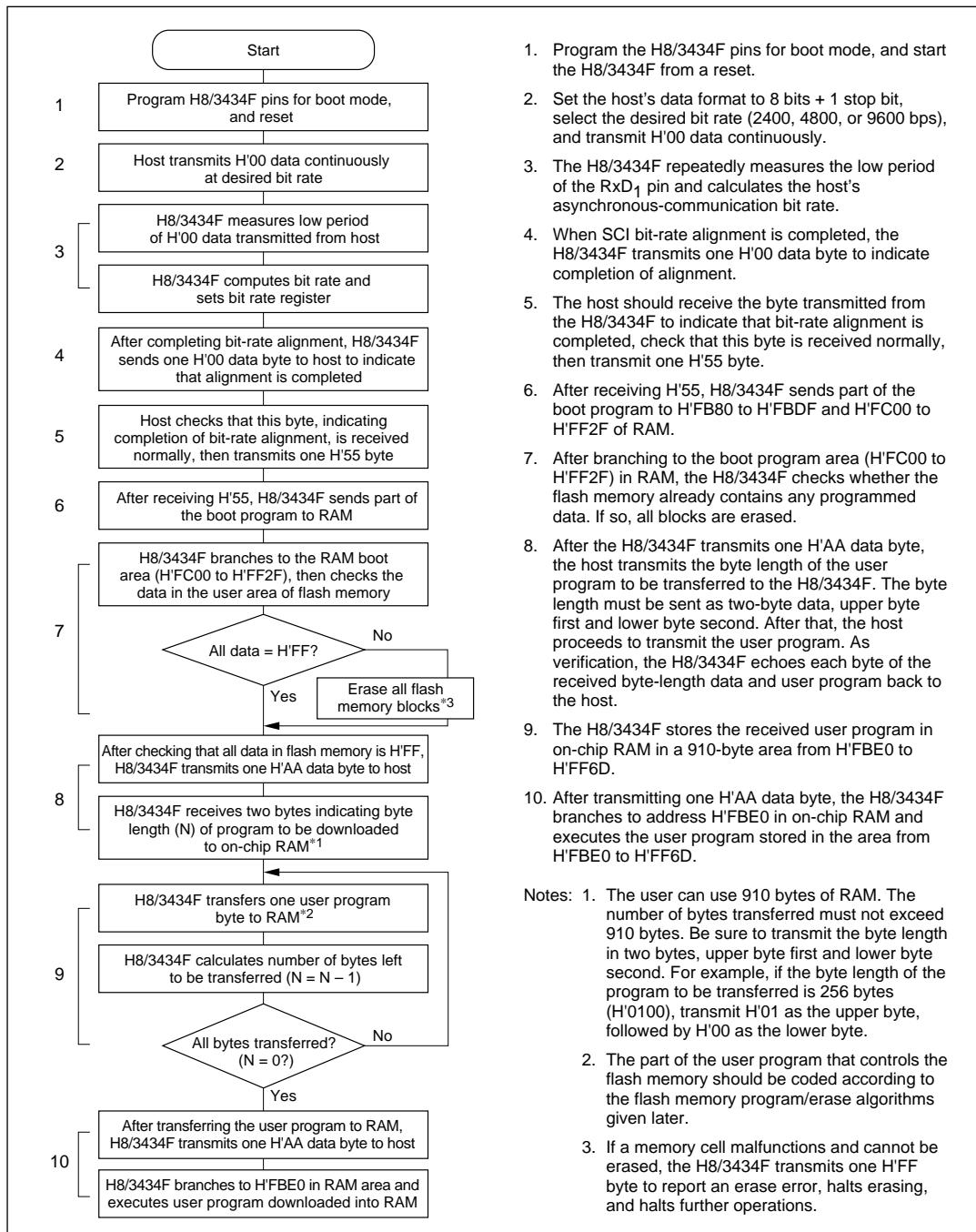
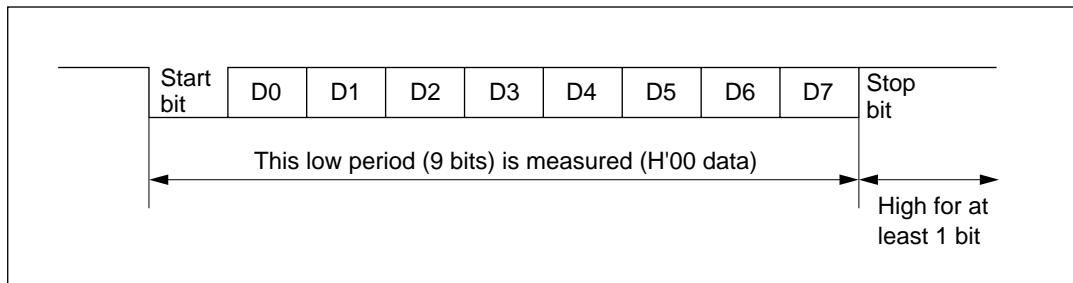


Figure 19-4 Boot Mode Flowchart

## Automatic Alignment of SCI Bit Rate



**Figure 19-5 Measurement of Low Period in Data Transmitted from Host**

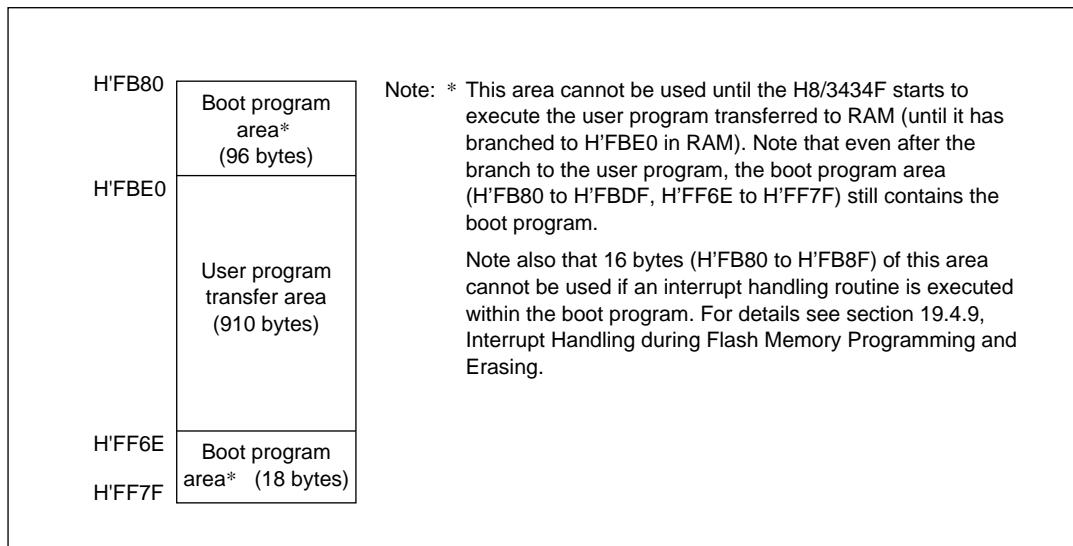
When started in boot mode, the H8/3434F measures the low period in asynchronous SCI data transmitted from the host (figure 19-5). The data format is eight data bits, one stop bit, and no parity bit. From the measured low period (9 bits), the H8/3434F computes the host's bit rate. After aligning its own bit rate, the H8/3434F sends the host 1 byte of H'00 data to indicate that bit-rate alignment is completed. The host should check that this alignment-completed indication is received normally and send one byte of H'55 back to the H8/3434F. If the alignment-completed indication is not received normally, the H8/3434F should be reset, then restarted in boot mode to measure the low period again. There may be some alignment error between the host's and H8/3434F's bit rates, depending on the host's bit rate and the H8/3434F's system clock frequency. To have the SCI operate normally, set the host's bit rate to 2400, 4800, or 9600 bps<sup>\*1</sup>. Table 19-8 lists typical host bit rates and indicates the clock-frequency ranges over which the H8/3434F can align its bit rate automatically. Boot mode should be used within these frequency ranges<sup>\*2</sup>.

**Table 19-8 System Clock Frequencies Permitting Automatic Bit-Rate Alignment by H8/3434F**

Host Bit Rate <sup>*1</sup>	System Clock Frequencies Permitting Automatic Bit-Rate Alignment by H8/3434F
9600 bps	8 MHz to 16 MHz
4800 bps	4 MHz to 16 MHz
2400 bps	2 MHz to 16 MHz

Notes: 1. Use a host bit rate setting of 2400, 4800, or 9600 bps only. No other setting should be used.  
2. Although the H8/3434F may also perform automatic bit-rate alignment with bit rate and system clock combinations other than those shown in table 19-8, there will be a slight difference between the bit rates of the host and the H8/3434F, and subsequent transfer will not be performed normally. Therefore, only a combination of bit rate and system clock frequency within one of the ranges shown in table 19-8 can be used for boot mode execution.

**RAM Area Allocation in Boot Mode:** In boot mode, the 96 bytes from H'FB80 to H'FBDF and the 18 bytes from H'FF6E to H'FF7F are reserved for use by the boot program, as shown in figure 19-6. The user program is transferred into the area from H'FBE0 to H'FF6D (910 bytes). The boot program area can be used after the transition to execution of the user program transferred into RAM. If a stack area is needed, set it within the user program.



**Figure 19-6 RAM Areas in Boot Mode**

## Notes on Use of Boot Mode

1. When the H8/3434F comes out of reset in boot mode, it measures the low period of the input at the SCI's  $\text{RxD}_1$  pin. The reset should end with  $\text{RxD}_1$  high. After the reset ends, it takes about 100 states for the H8/3434F to get ready to measure the low period of the  $\text{RxD}_1$  input.
2. In boot mode, if any data has been programmed into the flash memory (if all data is not H'FF), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, e.g. the first time on-board programming is performed, or if the update program activated in user program mode is accidentally erased.
3. Interrupts cannot be used while the flash memory is being programmed or erased.
4. The  $\text{RxD}_1$  and  $\text{TxD}_1$  pins should be pulled up on-board.
5. Before branching to the user program (at address H'FBE0 in the RAM area), the H8/3434F terminates transmit and receive operations by the on-chip SCI (by clearing the RE and TE bits of the serial control register to 0 in channel 1), but the auto-aligned bit rate remains set in bit rate register BRR. The transmit data output pin ( $\text{TxD}_1$ ) is in the high output state (in port 8, the bits  $\text{P8}_4$  DDR of the port 8 data direction register and  $\text{P8}_4$  DR of the port 8 data register are set to 1).

At this time, the values of general registers in the CPU are undetermined. Thus these registers should be initialized immediately after branching to the user program. Especially in the case of the stack pointer, which is used implicitly in subroutine calls, the stack area used by the user program should be specified.

There are no other changes to the initialized values of other registers.

6. Boot mode can be entered by starting from a reset after 12 V is applied to the  $\text{MD}_1$  and  $\text{FV}_{\text{PP}}$  pins according to the mode setting conditions listed in table 19-7. Note the following points when turning the  $\text{V}_{\text{PP}}$  power on.

When reset is released (at the rise from low to high), the H8/3434F checks for 12-V input at the  $\text{MD}_1$  and  $\text{FV}_{\text{PP}}$  pins. If it detects that these pins are programmed for boot mode, it saves that status internally. The threshold point of this voltage-level check is in the range from approximately  $\text{V}_{\text{CC}} + 2$  V to 11.4 V, so boot mode will be entered even if the applied voltage is insufficient for programming or erasure (11.4 V to 12.6 V). When the boot program is executed, the  $\text{V}_{\text{PP}}$  power supply must therefore be stabilized within the range of 11.4 V to 12.6 V before the branch to the RAM area occurs. See figure 19-20.

Make sure that the programming voltage  $\text{V}_{\text{PP}}$  does not exceed 12.6 V during the transition to boot mode (at the reset release timing) and does not go outside the range of  $12\text{ V} \pm 0.6$  V while in boot mode. Boot mode will not be executed correctly if these limits are exceeded. In addition, make sure that  $\text{V}_{\text{PP}}$  is not released or shut off while the boot program is executing or

the flash memory is being programmed or erased.\*1

Boot mode can be released by driving the reset pin low, waiting at least ten system clock cycles, then releasing the application of 12 V to the MD<sub>1</sub> and FV<sub>PP</sub> pins and releasing the reset.

The settings of external pins must not change during operation in boot mode.

During boot mode, if input of 12 V to the MD<sub>1</sub> pin stops but no reset input occurs at the RES pin, the boot mode state is maintained within the chip and boot mode continues (but do not stop applying 12 V to the FV<sub>PP</sub> pin during boot mode\*1).

If a watchdog timer reset occurs during boot mode, this does not release the internal mode state, but the internal boot program is restarted.

Therefore, to change from boot mode to another mode, the boot-mode state within the chip must be released by a reset input at the RES pin before the mode transition can take place.

7. If the input level of the MD<sub>1</sub> pin is changed during a reset (e.g., from 0 V to 5 V then to 12 V while the input to the RES pin is low), the resultant switch in the microcontroller's operating mode will affect the bus control output signals (AS, RD, and WR) and the status of ports that can be used for address output\*2.

Therefore, either set these pins so that they do not output signals during the reset, or make sure that their output signals do not collide with other signals outside the microcontroller.

8. When applying 12 V to the MD<sub>1</sub> and FV<sub>PP</sub> pins, make sure that peak overshoot does not exceed the rated limit of 13 V.

Also, be sure to connect a decoupling capacitor to the FV<sub>PP</sub> and MD<sub>1</sub> pins.

Notes: 1. For details on applying, releasing, and shutting off V<sub>PP</sub>, see note (5) in section 19.7, Flash Memory Programming and Erasing Precautions.

2. These ports output low-level address signals if the mode pins are set to mode 1 during the reset. In all other modes, these ports are in the high-impedance state. The bus control output signals are high if the mode pins are set for mode 1 or 2 during the reset. In mode 3, they are at high impedance.

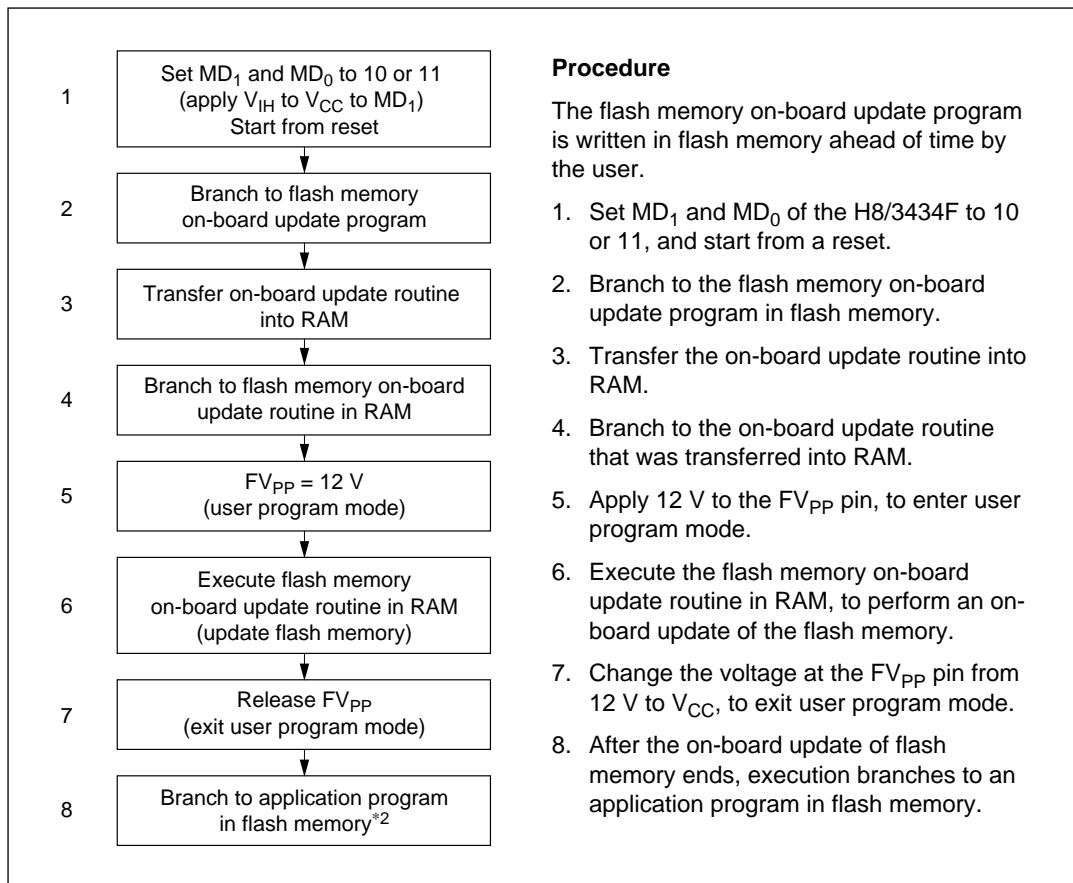
### 19.3.2 User Program Mode

When set to user program mode, the H8/3434F can erase and program its flash memory by executing a user program. On-board updates of the on-chip flash memory can be carried out by providing on-board circuits for supplying  $V_{PP}$  and data, and storing an update program in part of the program area.

To select user program mode, select a mode that enables the on-chip ROM (mode 2 or 3) and apply 12 V to the  $FV_{PP}$  pin, either during a reset, or after the reset has ended (been released) but while flash memory is not being accessed. In user program mode, the on-chip supporting modules operate as they normally would in mode 2 or 3, except for the flash memory. However, hardware standby mode cannot be set while 12 V is applied to the  $FV_{PP}$  pin.

The flash memory cannot be read while it is being programmed or erased, so the update program must either be stored in external memory, or transferred temporarily to the RAM area and executed in RAM.

**User Program Mode Execution Procedure (Example)\*1:** Figure 19-7 shows the execution procedure for user program mode when the on-board update routine is executed in RAM.



**Figure 19-7 User Program Mode Operation (Example)**

Notes:

1. Do not apply 12 V to the FV<sub>PP</sub> pin during normal operation. To prevent flash memory from being accidentally programmed or erased due to program runaway etc., apply 12 V to FV<sub>PP</sub> only when programming or erasing flash memory. Overprogramming or overerasing due to program runaway can cause memory cells to malfunction. While 12 V is applied, the watchdog timer should be running and enabled to halt runaway program execution, so that program runaway will not lead to overprogramming or overerasing. For details on applying, releasing, and shutting off V<sub>PP</sub>, see section 19.7, Flash Memory Programming and Erasing Precautions (5).
2. After the update is finished, when input of 12 V to the FV<sub>PP</sub> pin is released, the flash memory read setup time (t<sub>FRS</sub>) must elapse before any program in flash memory is executed. This is the required setup time from when the FV<sub>PP</sub> pin reaches the (V<sub>CC</sub> + 2 V) level after 12 V is released until flash memory can be read.

## 19.4 Programming and Erasing Flash Memory

The H8/3434F's on-chip flash memory is programmed and erased by software, using the CPU. The flash memory can operate in program mode, erase mode, program-verify mode, erase-verify mode, or prewrite-verify mode. Transitions to these modes can be made by setting the P, E, PV, and EV bits in the flash memory control register (FLMCR).

The flash memory cannot be read while being programmed or erased. The program that controls the programming and erasing of the flash memory must be stored and executed in on-chip RAM or in external memory. A description of each mode is given below, with recommended flowcharts and sample programs for programming and erasing.

For details on programming and erasing, refer to section 19.7, Flash Memory Programming and Erasing Precautions.

### 19.4.1 Program Mode

To write data into the flash memory, follow the programming algorithm shown in figure 19-8. This programming algorithm can write data without subjecting the device to voltage stress or impairing the reliability of programmed data.

To program data, first specify the area to be written in flash memory with erase block registers EBR1 and EBR2, then write the data to the address to be programmed, as in writing to RAM. The flash memory latches the address and data in an address latch and data latch. Next set the P bit in FLMCR, selecting program mode. The programming duration is the time during which the P bit is set. A software timer should be used to provide a programming duration of about 10 to 20  $\mu$ s. The value of N, the number of attempts, should be set so that the total programming time does not exceed 1 ms. Programming for too long a time, due to program runaway for example, can cause device damage. Before selecting program mode, set up the watchdog timer so as to prevent overprogramming.

### 19.4.2 Program-Verify Mode

In program-verify mode, after data has been programmed in program mode, the data is read to check that it has been programmed correctly.

After the programming time has elapsed, exit programming mode (clear the P bit to 0) and select program-verify mode (set the PV bit to 1). In program-verify mode, a program-verify voltage is applied to the memory cells at the latched address. If the flash memory is read in this state, the data at the latched address will be read. After selecting program-verify mode, wait 4  $\mu$ s or more before reading, then compare the programmed data with the verify data. If they agree, exit program-verify mode and program the next address. If they do not agree, select program mode again and repeat the same program and program-verify sequence. Do not repeat the program and program-verify sequence more than 50 times\* for the same bit.

Note: \* Keep the total programming time under 1 ms for each bit.

### 19.4.3 Programming Flowchart and Sample Program

#### Flowchart for Programming One Byte

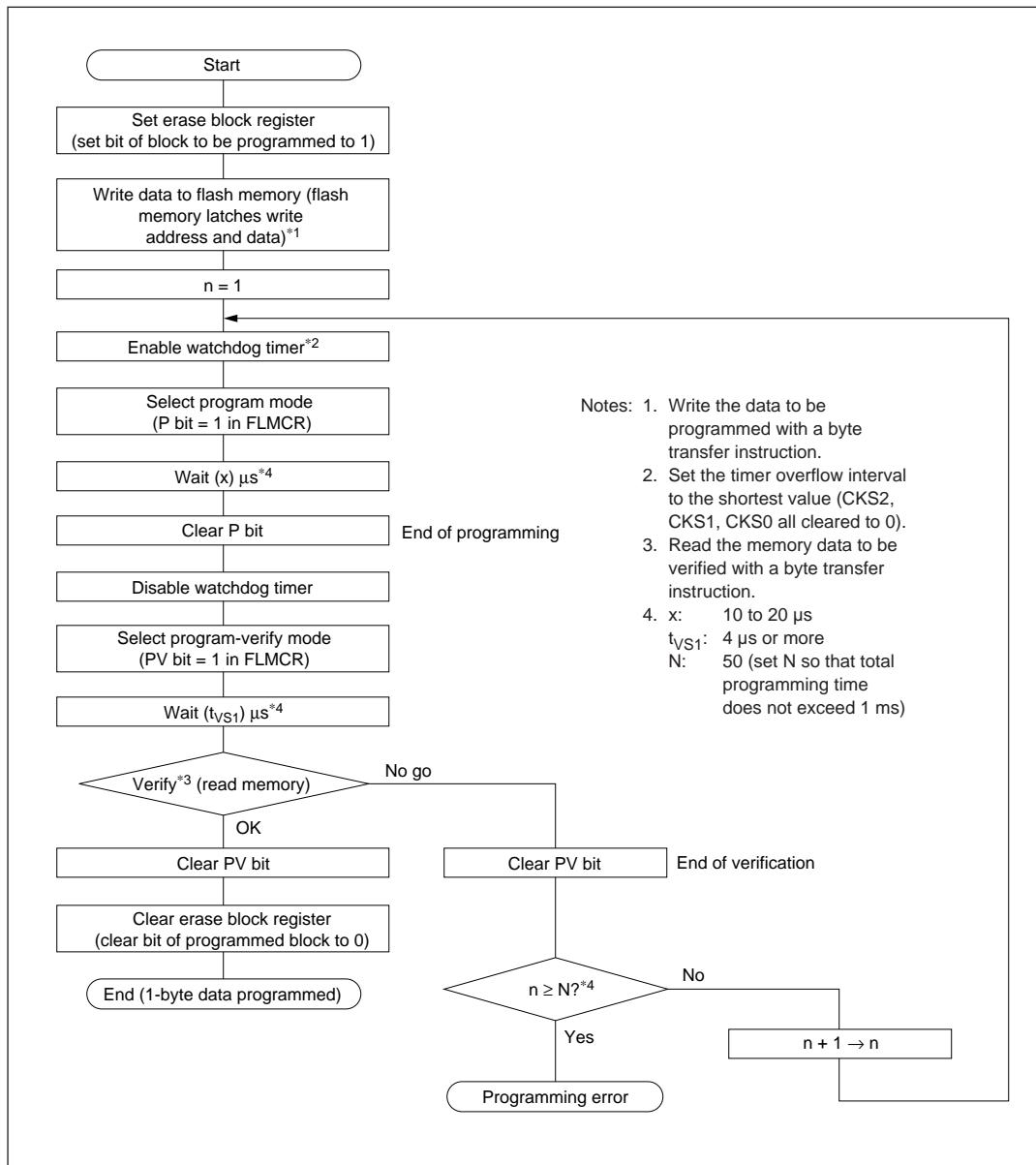


Figure 19-8 Programming Flowchart

**Sample Program for Programming One Byte:** This program uses the following registers.

- R0H: Specifies blocks to be erased.
- R1H: Stores data to be programmed.
- R1L: Stores data to be read.
- R3: Stores address to be programmed. Valid addresses are H'0000 to H'7FFF.
- R4: Sets program and program-verify timing loop counters, and also stores register setting value.
- R5: Sets program timing loop counter.
- R6L: Used for program-verify fail count.

Arbitrary data can be programmed at an arbitrary address by setting the address in R3 and the data in R1H.

The setting of #a and #b values depends on the clock frequency. Set #a and #b values according to tables 19-9 (1) and (2).

```
FLMCR: .EQU      H'FF80
EBR1:  .EQU      H'FF82
EBR2:  .EQU      H'FF83
TCSR:  .EQU      H'FFA8

        .ALIGN    2
PRGM:   MOV.B    #H'**,    R0H      ;
        MOV.B    R0H,     @EBR*:8      ; Set EBR*
        MOV.B    #H'00,    R6L      ; Program-verify fail counter
        MOV.W    #H'a,     R5       ; Set program loop counter
        MOV.B    R1H,     @R3      ; Dummy write
        INC      R6L      ; Program-verify fail counter + 1 → R6L
PRGMS:  MOV.W    #H'A578,  R4       ;
        MOV.W    R4,      @TCSR     ; Start watchdog timer
        MOV.W    R5,      R4       ; Set program loop counter
        BSET    #0,      @FLMCR:8  ; Set P bit
        SUBS    #1,      R4       ;
        MOV.W    R4,      R4       ;
        BNE     LOOP1     ; Wait loop
        BCLR    #0,      @FLMCR:8  ; Clear P bit
        MOV.W    #H'A500,  R4       ;
        MOV.W    R4,      @TCSR     ; Stop watchdog timer

        MOV.B    #H'b,    R4H      ; Set program-verify loop counter
        BSET    #2,      @FLMCR:8  ; Set PV bit
        DEC     R4H      ;
        BNE     LOOP2     ; Wait loop
        MOV.B    @R3,    R1L      ; Read programmed address
        CMP.B    R1H,    R1L      ; Compare programmed data with read data
        BEQ     PVOKE    ; Program-verify decision
        BCLR    #2,      @FLMCR:8  ; Clear PV bit
```

CMP.B	#H'32,	R6L	; Program-verify executed 50 times?
BEQ	NGEND		; If program-verify executed 50 times, branch to NGEND
BRA	PRGMS		; Program again
PVOK:	BCLR	#2,	@FLMCR:8 ; Clear PV bit
	MOV.B	#H'00,	R6L ;
	MOV.B	R6L,	@EBR*:8 ; Clear EBR*

One byte programmed

NGEND: Programming error

#### 19.4.4 Erase Mode

To erase the flash memory, follow the erasing algorithm shown in figure 19-9. This erasing algorithm can erase data without subjecting the device to voltage stress or impairing the reliability of programmed data.

To erase flash memory, before starting to erase, first place all memory data in all blocks to be erased in the programmed state (program all memory data to H'00). If all memory data is not in the programmed state, follow the sequence described later (figure 19-10) to program the memory data to zero. Select the flash memory areas to be erased with erase block registers 1 and 2 (EBR1 and EBR2). Next set the E bit in FLMCR, selecting erase mode. The erase time is the time during which the E bit is set. To prevent overerasing, use a software timer to divide the erase time into repeated 10 ms intervals, and perform erase operations a maximum of 3000 times so that the total erase time does not exceed 30 seconds. Overerasing, due to program runaway for example, can give memory cells a negative threshold voltage and cause them to operate incorrectly. Before selecting erase mode, set up the watchdog timer so as to prevent overerasing.

#### 19.4.5 Erase-Verify Mode

In erase-verify mode, after data has been erased, it is read to check that it has been erased correctly. After the erase time has elapsed, exit erase mode (clear the E bit to 0) and select erase-verify mode (set the EV bit to 1). Before reading data in erase-verify mode, write H'FF dummy data to the address to be read. This dummy write applies an erase-verify voltage to the memory cells at the latched address. If the flash memory is read in this state, the data at the latched address will be read. After the dummy write, wait 2  $\mu$ s or more before reading. When performing the initial dummy write, wait 4  $\mu$ s or more after selecting erase-verify mode. If the read data has been successfully erased, perform an erase-verify (dummy write, wait 2  $\mu$ s or more, then read) for the next address. If the read data has not been erased, select erase mode again and repeat the same erase and erase-verify sequence through the last address, until all memory data has been erased to 1. Do not repeat the erase and erase-verify sequence more than 3000 times, however.

## 19.4.6 Erasing Flowchart and Sample Program

### Flowchart for Erasing One Block

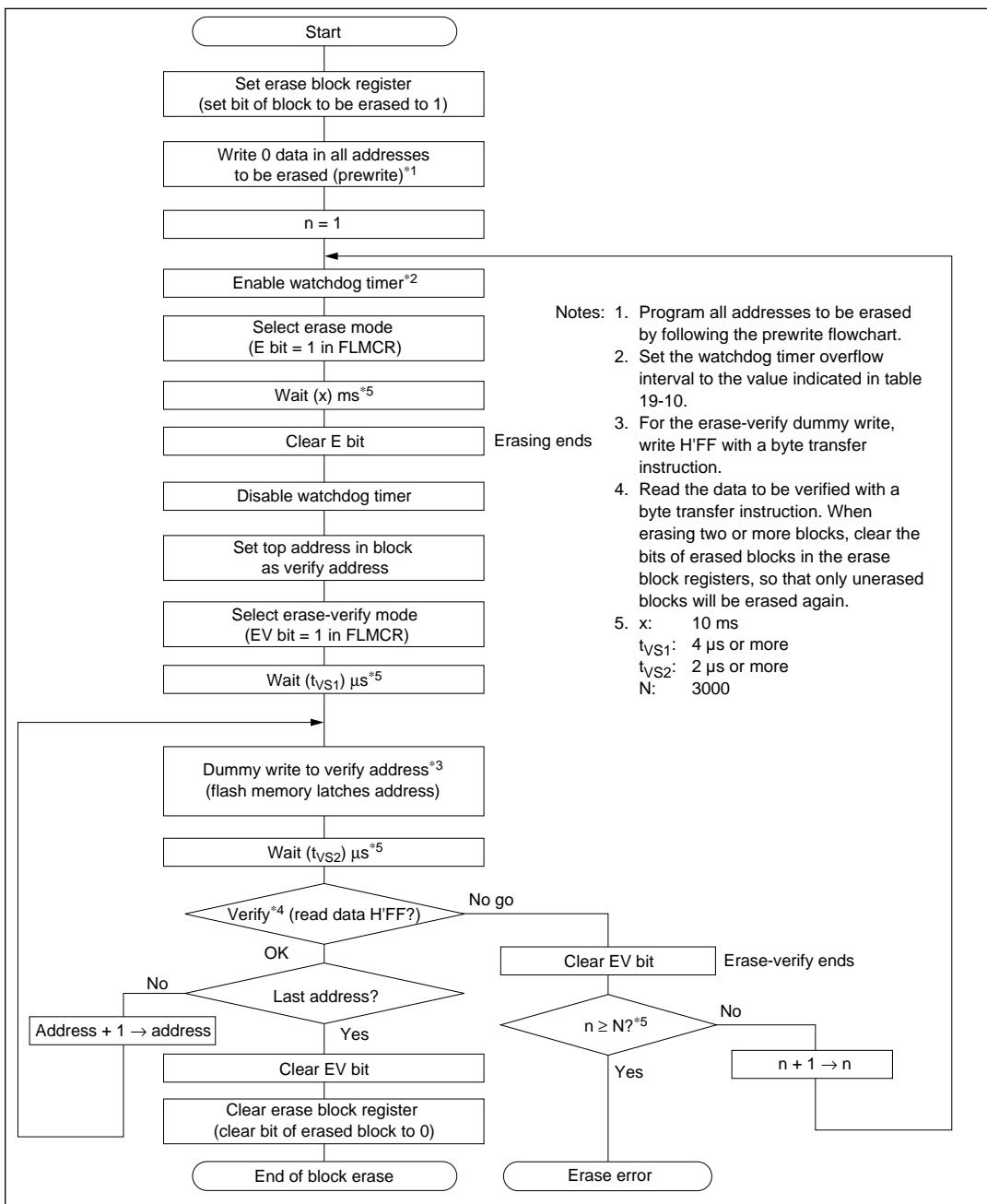


Figure 19-9 Erasing Flowchart

## Prewrite Flowchart

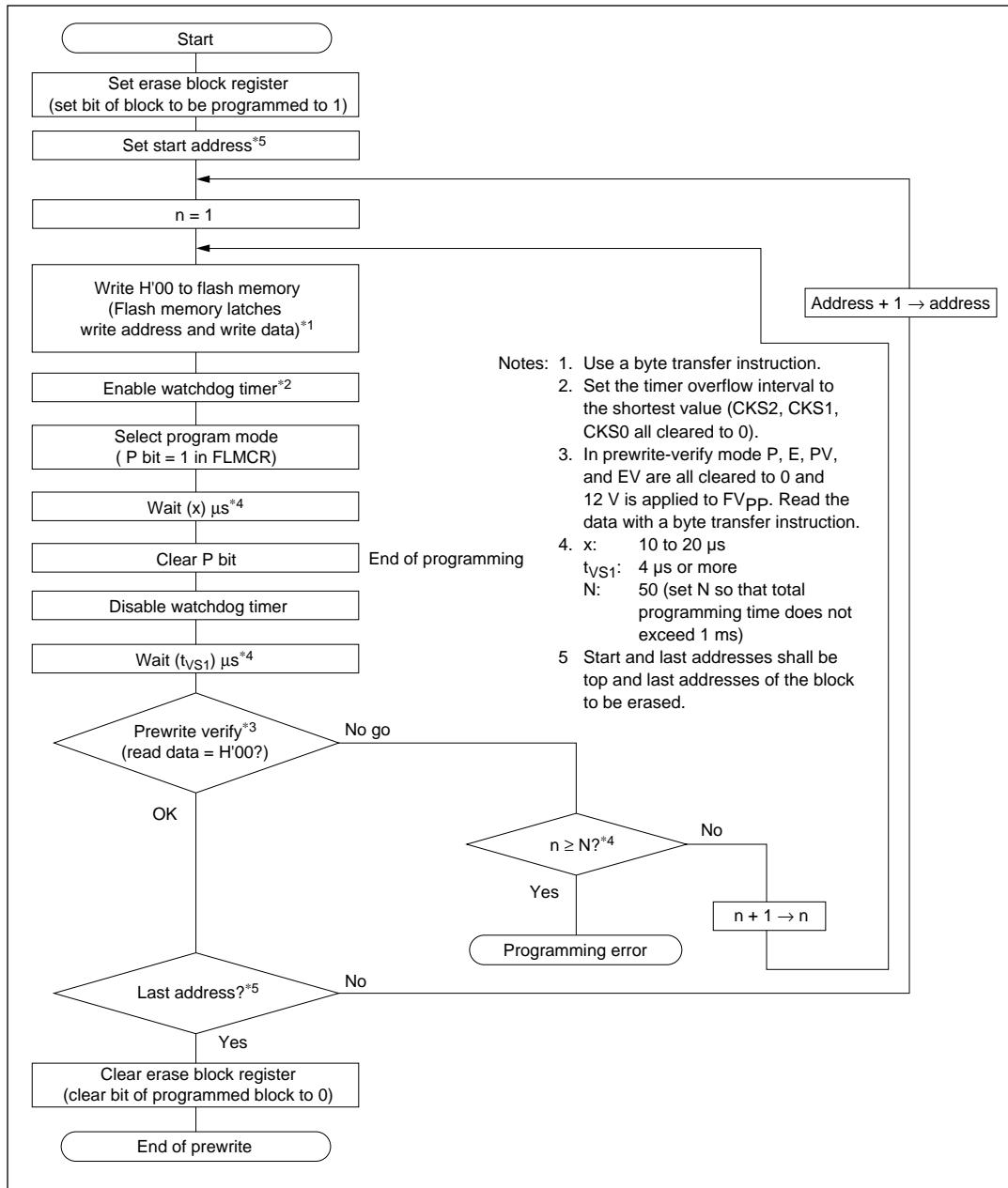


Figure 19-10 Prewrite Flowchart

**Sample Block-Erase Program:** This program uses the following registers.

- R0: Specifies block to be erased, and also stores address used in prewrite and erase-verify.
- R1H: Stores data to be read, and also used for dummy write.
- R2: Stores last address of block to be erased.
- R3: Stores address used in prewrite and erase-verify.
- R4: Sets timing loop counters for prewrite, prewrite-verify, erase, and erase-verify, and also stores register setting value.
- R5: Sets prewrite and erase timing loop counters.
- R6L: Used for prewrite-verify and erase-verify fail count.

The setting of #a, #b, #c, #d, and #e values in the program depends on the clock frequency. Set #a, #b, #c, #d, and #e values according to tables 19-9 (1) and (2), and 19-10. Erase block registers (EBR1 and EBR2) should be set according to sections 19.2.2 and 19.2.3. #BLKSTR and #BLKEND are the top and last addresses of the block to be erased. Set #BLKSTR and #BLKEND according to figure 19-2.

```

FLMCR: .EQU H'FF80
EBR1: .EQU H'FF82
EBR2: .EQU H'FF83
TCSR: .EQU H'FFA8

.ALIGN 2
MOV.B #H'**, ROH ; ; Set EBR*:8

; #BLKSTR is top address of block to be erased.
; #BLKEND is last address of block to be erased.

MOV.W #BLKSTR, R0 ; ; Top address of block to be erased
MOV.W #BLKEND, R2 ; ; Last address of block to be erased
ADDS #1, R2 ; ; Last address of block to be erased + 1 → R2

; Execute prewrite
MOV.W R0, R3 ; ; Top address of block to be erased
PREWRT: MOV.B #H'00, R6L ; ; Prewrite-verify fail counter
MOV.W #H'a, R5 ; ; Set prewrite loop counter
PREWRS: INC R6L ; ; Prewrite-verify fail counter + 1 → R6L
MOV.B #H'00 R1H ; ; Write H'00
MOV.B R1H, @R3 ; ; ;
MOV.W #H'A578, R4 ; ; ;
MOV.W R4, @TCSR ; ; Start watchdog timer
MOV.W R5, R4 ; ; Set prewrite loop counter
BSET #0, @FLMCR:8 ; ; Set P bit
LOOPR1: SUBS #1, R4 ; ;
MOV.W R4, R4 ; ;
BNE LOOPR1 ; ; Wait loop
BCLR #0, @FLMCR:8 ; ; Clear P bit
MOV.W #H'A500, R4 ; ;
MOV.W R4, @TCSR ; ; Stop watchdog timer

MOV.B #H'c, R4H ; ; Set prewrite-verify loop counter
LOOPR2: DEC R4H ; ;
BNE LOOPR2 ; ; Wait loop
MOV.B @R3, R1H ; ; Read data = H'00?
BEQ PWVFOK ; ; If read data = H'00 branch to PWVFOK
CMP.B #H'32, R6L ; ; Prewrite-verify executed 50 times?
BEQ ABEND1 ; ; If prewrite-verify executed 50 times, branch to ABEND1
BRA PREWRS ; ; Prewrite again

ABEND1: Programming error

PWVFOK: ADDS #1, R3 ; ; Address + 1 → R3
CMP.W R2, R3 ; ; Last address?
BNE PREWRT ; ; If not last address, prewrite next address

```

```

; Execute erase
ERASES: MOV.W #H'0000,R6 ; Erase-verify fail counter
         MOV.W #H'd, R5 ; Set erase loop count
ERASE:  ADDS #1, R6 ; Erase-verify fail counter + 1 → R6
         MOV.W #H'e, R4 ;
         MOV.W R4, @TCSR ; Start watchdog timer
         MOV.W R5, R4 ; Set erase loop counter
         BSET #1, @FLMCR:8 ; Set E bit

LOOPE:  NOP ;
         NOP ;
         NOP ;
         NOP ;
         SUBS #1, R4 ;
         MOV.W R4, R4 ;
         BNE LOOPE ; Wait loop
         BCLR #1, @FLMCR:8 ; Clear E bit
         MOV.W #H'A500,R4 ;
         MOV.W R4, @TCSR ; Stop watchdog timer

; Execute erase-verify
         MOV.W R0, R3 ; Top address of block to be erased
         MOV.B #H'b, R4H ; Set erase-verify loop counter
         BSET #3, @FLMCR:8 ; Set EV bit
LOOPEV: DEC R4H ;
         BNE LOOPEV ; Wait loop
EVR2:   MOV.B #H'FF, R1H ;
         MOV.B R1H, @R3 ;
         MOV.B #H'c, R4H ; Dummy write
         MOV.B @R3+, R1H ; Set erase-verify loop counter
LOOPDW: DEC R4H ;
         BNE LOOPDW ; Wait loop
         MOV.B @R3+, R1H ; Read
         CMP.B #H'FF, R1H ; Read data = H'FF?
         BNE RERASE ; If read data ≠ H'FF, branch to RERASE
         CMP.W R2, R3 ; Last address of block?
         BNE EVR2 ;
         BRA OKEND ;

RERASE: BCLR #3, @FLMCR:8 ; Clear EV bit
         SUBS #1, R3 ; Erase-verify address – 1 → R3

BRER:   MOV.W #H'0BB8,R4 ;
         CMP.W R4, R6 ; Erase-verify executed 3000 times?
         BNE ERASE ; If erase-verify not executed 3000 times, erase again
         BRA ABEND2 ; If erase-verify executed 3000 times, branch to ABEND2

OKEND:  BCLR #3, @FLMCR:8 ; Clear EV bit
         MOV.B #H'00, R6L ; ;
         MOV.B R6L, @EBR*:8 ; Clear EBR*

```

One block erased

ABEND2: Erase error

## Flowchart for Erasing Multiple Blocks

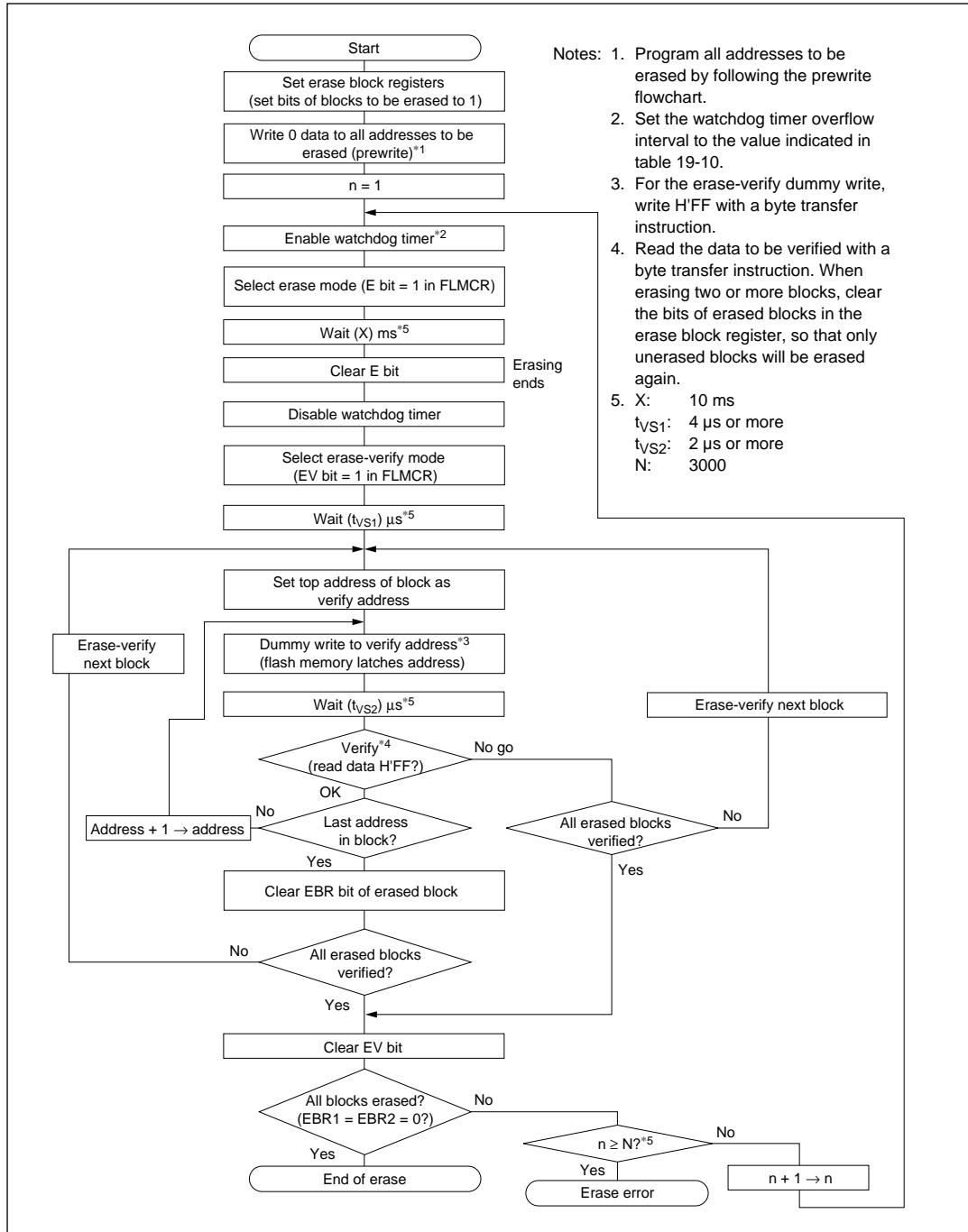


Figure 19-11 Multiple-Block Erase Flowchart

**Sample Multiple-Block Erase Program:** This program uses the following registers.

R0: Specifies blocks to be erased (set as explained below), and also stores address used in prewrite and erase-verify.

R1H: Used to test bits 8 to 11 of R0 stores register read data, and also used for dummy write.

R1L: Used to test bits 0 to 11 of R0.

R2: Specifies address where address used in prewrite and erase-verify is stored.

R3: Stores address used in prewrite and erase-verify.

R4: Stores last address of block to be erased.

R5: Sets prewrite and erase timing loop counters.

R6L: Used for prewrite-verify and erase-verify fail count.

Arbitrary blocks can be erased by setting bits in R0. Write R0 with a word transfer instruction.

A bit map of R0 and a sample setting for erasing specific blocks are shown next.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	—	—	—	—	LB3	LB2	LB1	LB0	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Corresponds to EBR1										Corresponds to EBR2						

Note: Clear bits 15, 14, 13, and 12 to 0.

Example: to erase blocks LB2, SB7, and SB0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	—	—	—	—	LB3	LB2	LB1	LB0	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Corresponds to EBR1										Corresponds to EBR2						

Setting    0    0    0    0    0    1    0    0    1    0    0    0    0    0    0    1

R0 is set as follows:

```
MOV.W    #H'0481, R0
MOV.W    R0,        @EBR1
```

The setting of #a, #b, #c, #d, and #e values in the program depends on the clock frequency. Set #a, #b, #c, #d, and #e values according to tables 19-9 (1), (2), and 19-10.

Notes:

1. In this sample program, the stack pointer (SP) is set at address FF80. As the stack area, on-chip RAM addresses FF7E and FF7F are used. Therefore, when executing this sample program, addresses FF7E and FF7F should not be used. In addition, the on-chip RAM should not be disabled.
2. In this sample program, the program written in a ROM area (including external space) is transferred into the RAM area and executed in the RAM to which the program is transferred. #RAMSTR in the program is the starting destination address in RAM to which the program is transferred. #RAMSTR must be set to an even number.
3. When executing this sample program in the on-chip ROM area or external space, #RAMSTR should be set to #START.

```

FLMCR: .RQU      H'FF80
EBR1:  .EQU      H'FF82
EBR2:  .EQU      H'FF83
TCSR:  .EQU      H'FFA8
STACK: .EQU      H'FF80

.ALIGN 2
START: MOV.W #STACK, SP           ; Set stack pointer
; Set the bits in R0 following the description on the previous page. This program is a sample program to erase
; all blocks.
        MOV.W #H'0FFF, R0           ; Select blocks to be erased (R0: EBR1/EBR2)
        MOV.W R0, @EBR1            ; Set EBR1/EBR2

; #RAMSTR is starting destination address to which program is transferred in RAM.
; Set #RAMSTR to even number.
        MOV.W #RAMSTR, R2           ; Starting transfer destination address (RAM)
        MOV.W #ERVADR, R3           ;
        ADD.W R3, R2               ; #RAMSTR + #ERVADR → R2
        MOV.W #START, R3           ;
        SUB.W R3, R2               ; Address of data area used in RAM

        MOV.B #H'00, R1L            ;
PRETST: CMP.B #H'0C, R1L            ; R1L = H'0C?
        BEQ ERASES                ; If finished checking all R0 bits, branch to ERASES
        CMP.B #H'08, R1L            ;
        BMI EBR2PW                ; Test EBR1 if R1L ≥ 8, or EBR2 if R1L < 8
        MOV.B R1L, R1H              ;
        SUBX #H'08, R1H              ; R1L - 8 → R1H
        BTST R1H, R0H              ; Test R1H bit in EBR1 (R0H)
        BNE PREWRT                ; If R1H bit in EBR1 (R0H) is 1, branch to PREWRT
        BRA PWADD1                ; If R1H bit in EBR1 (R0H) is 0, branch to PWADD1
EBR2PW: BTST R1L, R0L              ; Test R1L bit in EBR2 (R0L)
        BNE PREWRT                ; If R1L bit in EBR2 (R0H) is 1, branch to PREWRT
PWADD1: INC R1L                  ; R1L + 1 → R1L
        MOV.W @R2+, R3              ; Dummy-increment R2
        BRA PRETST                ;

; Execute prewrite

```

PREWRT:	MOV.W	@R2+, R3	; Prewrite starting address
PREW:	MOV.B	#H'00, R6L	; Prewrite-verify fail counter
	MOV.W	#H'a, R5	; Prewrite-verify loop counter
PREWRS:	INC	R6L	; Prewrite-verify fail counter + 1 → R6L
	MOV.B	#H'00 R1H	;
	MOV.B	R1H, @R3	; Write H'00
	MOV.W	#H'A578, R4	;
	MOV.W	R4, @TCSR	; Start watchdog timer
	MOV.W	R5, R4	; Set prewrite loop counter
	BSET	#0, @FLMCR:8	; Set P bit
LOOPR1:	SUBS	#1, R4	;
	MOV.W	R4, R4	;
	BNE	LOOPR1	; Wait loop
	BCLR	#0, @FLMCR:8	; Clear P bit
	MOV.W	#H'A500, R4	;
	MOV.W	R4, @TCSR	; Stop watchdog timer
	MOV.B	#H'c, R4H	; Set prewrite-verify loop counter
LOOPR2:	DEC	R4H	;
	BNE	LOOPR2	; Wait loop
	MOV.B	@R3, R1H	; Read data = H'00?
	BEQ	PWVFOK	; If read data = H'00 branch to PWVFOK
	CMP.B	#H'32, R6L	; Prewrite-verify executed 50 times?
	BEQ	ABEND1	; If prewrite-verify executed 50 times, branch to ABEND1
	BRA	PREWRS	; Prewrite again
ABEND1:	Programming error		
PWVFOK:	ADDS	#1, R3	; Address + 1 → R3
	MOV.W	@R2, R4	; Top address of next block
	CMP.W	R4, R3	; Last address?
	BNE	PREW	; If not last address, prewrite next address
PWADD2:	INC	R1L	; Used to test R1L+1 bit in R0
	BRA	PRETST	; Branch to PRETST
; Execute erase			
ERASES:	MOV.W	#H'0000, R6	; Erase-verify fail counter
	MOV.W	#H'd, R5	; Set erase loop count
ERASE:	ADDS	#1, R6	; Erase-verify fail counter + 1 → R6
	MOV.W	#H'e, R4	;
	MOV.W	R4, @TCSR	; Start watchdog timer
	MOV.W	R5, R4	; Set erase loop counter
	BSET	#1, @FLMCR:8	; Set E bit
LOOPE:	NOP		
	SUBS	#1, R4	;
	MOV.W	R4, R4	;
	BNE	LOOPE	; Wait loop
	BCLR	#1, @FLMCR:8	; Clear E bit

	MOV.W	#H'A500,	R4	;
	MOV.W	R4,	@TCSR	; Stop watchdog timer
; Execute erase-verify				
EVR :	MOV.W	#RAMSTR,	R2	; Starting transfer destination address (RAM)
	MOV.W	#ERVADR,	R3	;
	ADD.W	R3,	R2	; #RAMSTR + #ERVADR → R2
	MOV.W	#START,	R3	;
	SUB.W	R3,	R2	; Address of data area used in RAM
	MOV.B	#H'00,	R1L	; Used to test R1L bit in R0
	MOV.B	#H'b,	R4H	; Set erase-verify loop counter
	BSET	#3,	@FLMCR:8	; Set EV bit
LOOPEV :	DEC	R4H		;
	BNE	LOOPEV		; Wait loop
EBRTST :	CMP.B	#H'0C,	R1L	; R1L = H'0C?
	BEQ	HANTEI		;
	CMP.B	#H'08,	R1L	;
	BMI	EBR2EV		; Test EBR1 if R1L ≥ 8, or EBR2 if R1L < 8
	MOV.B	R1L,	R1H	;
	SUBX	#H'08,	R1H	;
	BTST	R1H,	R0H	; R1L - 8 → R1H
	BNE	ERSEVF		; Test R1H bit in EBR1 (R0H)
	BRA	ADD01		; If R1H bit in EBR1 (R0H) is 1, branch to ERSEVF
EBR2EV :	BTST	R1L,	R0L	; If R1H bit in EBR1 (R0H) is 0, branch to ADD01
	BNE	ERSEVF		; Test R1L bit in EBR2 (R0L)
ADD01 :	INC	R1L		; If R1L bit in EBR2 (R0L) is 1, branch to ERSEVF
	MOV.W	@R2+,	R3	; R1L + 1 → R1L
	BRA	EBRTST		; Dummy-increment R2
ERASE1 :	BRA	ERASE		; Branch to ERASE via Erase 1
ERSEVF :	MOV.W	@R2+,	R3	; Top address of block to be erase-verified
EVR2 :	MOV.B	#H'FF,	R1H	;
	MOV.B	R1H,	@R3	; Dummy write
	MOV.B	#H'c,	R4H	; Set erase-verify loop counter
LOOPEP :	DEC	R4H		;
	BNE	LOOPEP		; Wait loop
	MOV.B	@R3+,	R1H	; Read
	CMP.B	#H'FF,	R1H	; Read data = H'FF?
	BNE	BLKAD		; If read data ≠ H'FF branch to BLKAD
	MOV.W	@R2,	R4	; Top address of next block
	CMP.W	R4,	R3	; Last address of block?
	BNE	EVR2		
	CMP.B	#H'08,	R1L	;
	BMI	SBCLR		; Test EBR1 if R1L ≥ 8, or EBR2 if R1L < 8
	MOV.B	R1L,	R1H	;
	SUBX	#H'08,	R1H	; R1L - 8 → R1H
	BCLR	R1H,	R0H	; Clear R1H bit in EBR1 (R0H)
	BRA	BLKAD		;
SBCLR :	BCLR	R1L,	R0L	; Clear R1L bit in EBR2 (R0L)

BLKAD:	INC	R1L	;	R1L + 1 → R1L
	BRA	EBRTST	;	
HANTEI:	BCLR	#3,	@FLMCR:8	; Clear EV bit
	MOV.W	R0,	@EBR1	;
	BEQ	EOWARI		; If EBR1/EBR2 is all 0, erasing ended normally
BRER:	MOV.W	#H'0BB8,	R4	;
	CMP.W	R4,	R6	; Erase-verify executed 3000 times?
	BNE	ERASE1		; If erase-verify not executed 3000 times, erase again
	BRA	ABEND2		; If erase-verify executed 3000 times, branch to ABEND2

;<-----< Block address table used in erase-verify> ----->

.ALIGN 2

ERVADR:	.DATA.W	H'0000	;	SB0
	.DATA.W	H'0080	;	SB1
	.DATA.W	H'0100	;	SB2
	.DATA.W	H'0180	;	SB3
	.DATA.W	H'0200	;	SB4
	.DATA.W	H'0400	;	SB5
	.DATA.W	H'0800	;	SB6
	.DATA.W	H'0C00	;	SB7
	.DATA.W	H'1000	;	LB0
	.DATA.W	H'2000	;	LB1
	.DATA.W	H'4000	;	LB2
	.DATA.W	H'6000	;	LB3
	.DATA.W	H'8000	;	FLASH END

EOWARI: Erase end

ABEND2: Erase error

## Loop Counter Values in Programs and Watchdog Timer Overflow Interval Settings:

The setting of #a, #b, #c, #d, and #e values in the programs depends on the clock frequency.

Tables 19-9 (1) and (2) indicate sample loop counter settings for typical clock frequencies.

However, #e is set according to table 19-10.

As a software loop is used, calculated values including percent errors may not be the same as actual values. Therefore, the values are set so that the total programming time and total erase time do not exceed 1 ms and 30 s, respectively.

The maximum number of writes in the program, N, is set to 50.

Programming and erasing in accordance with the flowcharts is achieved by setting #a, #b, #c, and #d in the programs as shown in tables 19-9 (1) and (2). #e should be set as shown in table 19-10.

Wait state insertion is inhibited in these programs. If wait states are to be used, the setting should be made after the program ends. The setting value for the watchdog timer (WDT) overflow time is calculated based on the number of instructions between starting and stopping of the WDT, including the write time and erase time. Therefore, no other instructions should be added between starting and stopping of the WDT in this program example.

**Table 19-9 (1) #a, #b, #c, and #d Setting Values for Typical Clock Frequencies with Program Running in the On-Chip Memory (RAM)**

Variable	Time Setting	Clock Frequency			
		f = 16 MHz	f = 10 MHz	f = 8 MHz	f = 2 MHz
a <sub>(f)</sub>	Programming time	20 $\mu$ s	H'0028	H'0019	H'0014
b <sub>(f)</sub>	tvs1	4 $\mu$ s	H'0B	H'07	H'06
c <sub>(f)</sub>	tvs2	2 $\mu$ s	H'06	H'04	H'03
d <sub>(f)</sub>	Erase time	10 ms	H'2710	H'186A	H'1388
					H'04E2

**Table 19-9 (2) #a, #b, #c, and #d Setting Values for Typical Clock Frequencies with Program Running in the External Device**

Variable	Time Setting	Clock Frequency			
		f = 16 MHz	f = 10 MHz	f = 8 MHz	f = 2 MHz
a <sub>(f)</sub>	Programming time	20 $\mu$ s	H'000D	H'0008	H'0006
b <sub>(f)</sub>	tvs1	4 $\mu$ s	H'04	H'03	H'02
c <sub>(f)</sub>	tvs2	2 $\mu$ s	H'02	H'02	H'01
d <sub>(f)</sub>	Erase time	10 ms	H'0D05	H'0823	H'0682
					H'01A0

Formula: When using a clock frequency not shown in tables 19-9 (1) and (2), follow the formula below. The calculation is based on a clock frequency of 10 MHz.

After calculating a(f) and d(f) in the decimal system, omit the first decimal figures, and convert them to the hexadecimal system, so that a(f) and d(f) are set to 20  $\mu$ s or less and 10 ms or less, respectively.

After calculating b(f) and c(f) in the decimal system, raise the first decimal figures, and convert them to the hexadecimal system, so that b(f) and c(f) are set to 4  $\mu$ s or more and 2  $\mu$ s or more, respectively.

$$a(f) \text{ to } d(f) = \frac{\text{Clock Frequency } f [\text{MHz}]}{10} \times a(f = 10) \text{ to } d(f = 10)$$

Examples for a program running in on-chip memory (RAM) at a clock frequency of 12 MHz:

$$\begin{aligned}
 a(f) &= \frac{12}{10} \times 25 = 30 \approx 30 = H'001E \\
 b(f) &= \frac{12}{10} \times 7 = 8.4 \approx 9 = H'09 \\
 c(f) &= \frac{12}{10} \times 4 = 4.8 \approx 5 = H'05 \\
 d(f) &= \frac{12}{10} \times 6250 = 7500 \approx 7500 = H'1D4C
 \end{aligned}$$

**Table 19-10 Watchdog Timer Overflow Interval Settings  
(#e Setting Value According to Clock Frequency)**

Clock Frequency [MHz]	Variable e (f)
10 MHz $\leq$ frequency $\leq$ 16 MHz	H'A57F
2 MHz $\leq$ frequency $<$ 10 MHz	H'A57E

#### 19.4.7 Prewrite Verify Mode

Prewrite-verify mode is a verify mode used when programming all bits to equalize their threshold voltages before erasing them.

Program all flash memory to H'00 by writing H'00 using the prewrite algorithm shown in figure 19-10. H'00 should also be written when using RAM for flash memory emulation (when prewriting a RAM area). (This also applies when using RAM to emulate flash memory erasing with an emulator or other support tool.) After the necessary programming time has elapsed, exit program mode (by clearing the P bit to 0) and select prewrite-verify mode (leave the P, E, PV, and EV bits all cleared to 0). In prewrite-verify mode, a prewrite-verify voltage is applied to the memory cells at the read address. If the flash memory is read in this state, the data at the read address will be read. After selecting prewrite-verify mode, wait 4  $\mu$ s or more before reading.

Note: For a sample prewriting program, see the prewrite subroutine in the sample erasing program.

#### 19.4.8 Protect Modes

Flash memory can be protected from programming and erasing by software or hardware methods. These two protection modes are described below.

**Software Protection:** Prevents transitions to program mode and erase mode even if the P or E bit is set in the flash memory control register (FLMCR). Details are as follows.

Protection	Description	Function		
		Program	Erase	Verify <sup>*1</sup>
Block protect	Individual blocks can be protected from erasing and programming by the erase block registers (EBR1 and EBR2). If H'F0 is set in EBR1 and H'00 in EBR2, all blocks are protected from erasing and programming.	Disabled	Disabled	Enabled
Emulation protect <sup>*2</sup>	When the RAMS or RAM0 bit, but not both, is set in the wait-state control register (WSCR), all blocks are protected from programming and erasing.	Disabled	Disabled <sup>*3</sup>	Enabled

Notes:

1. Three modes: program-verify, erase-verify, and prewrite-verify.
2. Except in RAM areas overlapped onto flash memory.
3. All blocks are erase-disabled. It is not possible to specify individual blocks.

**Hardware Protection:** Suspends or disables the programming and erasing of flash memory, and resets the flash memory control register (FLMCR) and erase block registers (EBR1 and EBR2). Details of hardware protection are as follows.

Protection	Description	Function		
		Program	Erase	Verify <sup>*1</sup>
Programming voltage ( $V_{PP}$ ) protect	When 12 V is not applied to the $FV_{PP}$ pin, FLMCR, EBR1, and EBR2 are initialized, disabling programming and erasing. To obtain this protection, $V_{PP}$ should not exceed $V_{CC}$ . <sup>*3</sup>	Disabled	Disabled <sup>*2</sup>	Disabled
Reset and standby protect	When a reset occurs (including a watchdog timer reset) or standby mode is entered, FLMCR, EBR1, and EBR2 are initialized, disabling programming and erasing. Note that $\overline{RES}$ input does not ensure a reset unless the $\overline{RES}$ pin is held low for at least 20 ms at power-up (to enable the oscillator to settle), or at least ten system clock cycles (10 $\phi$ ) during operation.	Disabled	Disabled <sup>*2</sup>	Disabled
Interrupt protect	To prevent damage to the flash memory, if interrupt input occurs while flash memory is being programmed or erased, programming or erasing is aborted immediately. The settings in FLMCR, EBR1, and EBR2 are retained. This type of protection can be cleared only by a reset.	Disabled	Disabled <sup>*2</sup>	Enabled

Notes: 1. Three modes: program-verify, erase-verify, and prewrite-verify.  
 2. All blocks are erase-disabled. It is not possible to specify individual blocks.  
 3. For details, see section 19.7, Flash Memory Programming and Erasing Precautions.

#### 19.4.9 Interrupt Handling during Flash Memory Programming and Erasing

If an interrupt occurs<sup>\*1</sup> while flash memory is being programmed or erased (while the P or E bit of FLMCR is set), the following operating states can occur.

- If an interrupt is generated during programming or erasing, programming or erasing is aborted to protect the flash memory. Since memory cell values after a forced interrupt are indeterminate, the system will not operate correctly after such an interrupt.
- Program runaway may result because the vector table could not be read correctly in interrupt exception handling during programming or erasure<sup>\*2</sup>.

For NMI interrupts while flash memory is being programmed or erased, these malfunction and runaway problems can be prevented by using the RAM overlap function with the settings described below.

1. Do not store the NMI interrupt-handling routine<sup>\*3</sup> in the flash memory area (H'0000 to H'7FFF). Store it elsewhere (in RAM, for example).

2. Set the NMI interrupt vector in address H'FC06 in RAM (corresponding to H'0006 in flash memory).
3. After the above settings, set both the RAMS and RAM0 bits to 1 in WSCR.\*<sup>4</sup>

Due to the setting of step 3, if an interrupt signal is input while 12 V is applied to the FV<sub>PP</sub> pin, the RAM overlap function is enabled and part of the RAM (H'FC00 to H'FC7F) is overlapped onto the small-block area of flash memory (H'0000 to H'007F). As a result, when an interrupt is input, the vector is read from RAM, not flash memory, so the interrupt is handled normally even if flash memory is being programmed or erased. This can prevent malfunction and runaway.

Notes: 1. When the interrupt mask bit (I) of the condition control register (CCR) is set to 1, all interrupts except NMI are masked. For details see (2) in section 2.2.2, Control Registers.

2. The vector table might not be read correctly for one of the following reasons:
  - If flash memory is read while it is being programmed or erased (while the P or E bit of FLMCR is set), the correct value cannot be read.
  - If no value has been written for the NMI entry in the vector table yet, NMI exception handling will not be executed correctly.
3. This routine should be programmed so as to prevent microcontroller runaway.
4. For details on WSCR settings, see section 19.2.4, Wait-State Control Register.

### Notes on Interrupt Handling in Boot Mode

In boot mode, the settings described above concerning NMI interrupts are carried out, and NMI interrupt handling (but not other interrupt handling) is enabled while the boot program is executing. Note the following points concerning the user program.

- If interrupt handling is required
  - Load the NMI vector (H'FB80) into address H'FC06 in RAM (the 38th byte of the transferred user program should be H'FB80).
  - The interrupt handling routine used by the boot program is stored in addresses H'FB80 to H'FB8F in RAM. Make sure that the user program does not overwrite this area.
- If interrupt handling is not required

Since the RAMS and RAM0 bits remain set to 1 in WSCR, make sure that the user program disables the RAM overlap by clearing the RAMS and RAM0 bits both to 0.

## 19.5 Flash Memory Emulation by RAM

Erasing and programming flash memory takes time, which can make it difficult to tune parameters and other data in real time. If necessary, real-time updates of flash memory can be emulated by overlapping the small-block flash-memory area with part of the RAM (H'FC00 to H'FD7F). This RAM reassignment is performed using bits 7 and 6 of the wait-state control register (WSCR). See figure 19-12.

After a flash memory area has been overlapped by RAM, the RAM area can be accessed from two address areas: the overlapped flash memory area, and the original RAM area (H'FC00 to H'FD7F). Table 19-11 indicates how to reassign RAM.

### Wait-State Control Register (WSCR)<sup>\*2</sup>

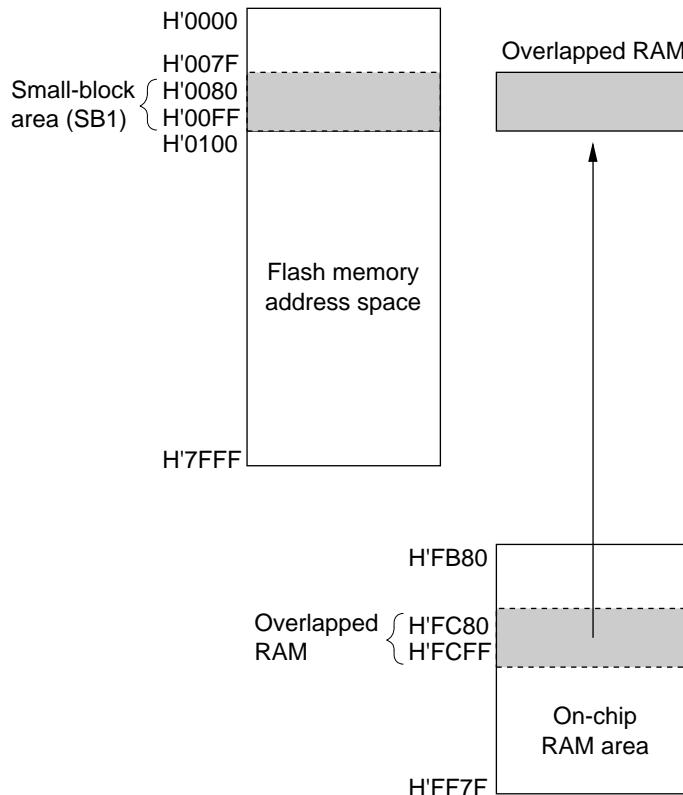
Bit	7	6	5	4	3	2	1	0
	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value <sup>*1</sup>	0	0	0	0	1	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Notes: 1. WSCR is initialized by a reset and in hardware standby mode. It is not initialized in software standby mode.  
2. For details of WSCR settings, see section 19.2.4, Wait-State Control Register (WSCR).

Table 19-11 RAM Area Selection

Bit 7 RAMS	Bit 6 RAM0	RAM Area	ROM Area
0	0	None	—
0	1	H'FC80 to H'FCFF	H'0080 to H'00FF
1	0	H'FC80 to H'FD7F	H'0080 to H'017F
1	1	H'FC00 to H'FC7F	H'0000 to H'007F

## Example of Emulation of Real-Time Flash-Memory Update



### Procedure

1. Overlap part of RAM (H'FC80 to H'FCFF) onto the area requiring real-time update (SB1). (Set WSCR bits 7 and 6 to 01.)
2. Perform real-time updates in the overlapping RAM.
3. After finalization of the update data, clear the RAM overlap (by clearing the RAMS and RAM0 bits).
4. Read the data written in RAM addresses H'FC80 to H'FCFF out externally, then program the flash memory area, using this data as part of the program data.

**Figure 19-12 Example of RAM Overlap**

## Notes on Use of RAM Emulation Function

- **Notes on Applying, Releasing, and Shutting Off the Programming Voltage (V<sub>PP</sub>)**

Care is necessary to avoid errors in programming and erasing when applying, releasing, and shutting off V<sub>PP</sub>, just as in the on-board programming modes. In particular, even if the emulation function is being used, make sure that the watchdog timer is set when the P or E bit of the flash memory control register (FLMCR) has been set, to prevent errors in programming and erasing due to program runaway while V<sub>PP</sub> is applied.

For details see section 19.7, Flash Memory Programming and Erasing Precautions (5).

## 19.6 Flash Memory PROM Mode (H8/3434F)

### 19.6.1 PROM Mode Setting

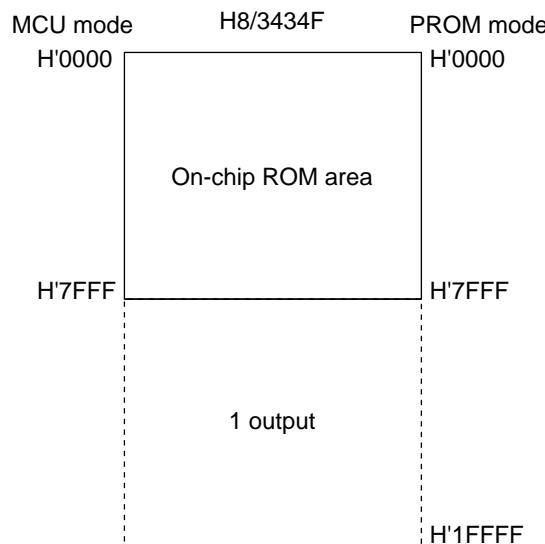
The on-chip flash memory of the H8/3434F can be programmed and erased not only in the on-board programming modes but also in PROM mode, using a general-purpose PROM programmer.

### 19.6.2 Socket Adapter and Memory Map

Programs can be written and verified by attaching a special 100-pin/32-pin socket adapter to the PROM programmer. Table 19-12 gives ordering information for the socket adapter. Figure 19-13 shows a memory map in PROM mode. Figure 19-14 shows the socket adapter pin interconnections.

**Table 19-12** Socket Adapter

Microcontroller	Package	Socket Adapter
HD64F3434F16	100-pin QFP	HS3434ESHF1H
HD64F3434TF16	100-pin TQFP	HS3434ESNF1H



**Figure 19-13** Memory Map in PROM Mode

H8/3434F		Socket Adapter	HN28F101 (32 Pins)	
Pin No.	Pin Name		Pin Name	Pin No.
FP-100B, TFP-100B				
8	STBY/FV <sub>PP</sub>		V <sub>PP</sub>	1
7	NMI		FA <sub>9</sub>	26
18	P9 <sub>5</sub>		FA <sub>16</sub>	2
19	P9 <sub>4</sub>		FA <sub>15</sub>	3
22	P9 <sub>3</sub>		WE	31
82	P3 <sub>0</sub>		FO <sub>0</sub>	13
83	P3 <sub>1</sub>		FO <sub>1</sub>	14
84	P3 <sub>2</sub>		FO <sub>2</sub>	15
85	P3 <sub>3</sub>		FO <sub>3</sub>	17
86	P3 <sub>4</sub>		FO <sub>4</sub>	18
87	P3 <sub>5</sub>		FO <sub>5</sub>	19
88	P3 <sub>6</sub>		FO <sub>6</sub>	20
89	P3 <sub>7</sub>		FO <sub>7</sub>	21
79	P1 <sub>0</sub>		FA <sub>0</sub>	12
78	P1 <sub>1</sub>		FA <sub>1</sub>	11
77	P1 <sub>2</sub>		FA <sub>2</sub>	10
76	P1 <sub>3</sub>		FA <sub>3</sub>	9
75	P1 <sub>4</sub>		FA <sub>4</sub>	8
74	P1 <sub>5</sub>		FA <sub>5</sub>	7
73	P1 <sub>6</sub>		FA <sub>6</sub>	6
72	P1 <sub>7</sub>		FA <sub>7</sub>	5
67	P2 <sub>0</sub>		FA <sub>8</sub>	27
66	P2 <sub>1</sub>		FA <sub>10</sub>	23
65	P2 <sub>2</sub>		FA <sub>11</sub>	25
64	P2 <sub>3</sub>		FA <sub>12</sub>	4
63	P2 <sub>4</sub>		FA <sub>13</sub>	28
62	P2 <sub>5</sub>		FA <sub>14</sub>	29
61	P2 <sub>6</sub>		CE	22
60	P2 <sub>7</sub>		V <sub>CC</sub>	32
24, 25, 29, 32, 16	P9 <sub>1</sub> , P9 <sub>0</sub> , P6 <sub>3</sub> , P6 <sub>4</sub> , P9 <sub>7</sub>		V <sub>SS</sub>	16
5, 6, 23, 35	MD <sub>1</sub> , MD <sub>0</sub> , P9 <sub>2</sub> , P6 <sub>7</sub>			
36, 37	AV <sub>ref</sub> , AV <sub>CC</sub>			
4, 9, 59	V <sub>CCB</sub> , V <sub>CC</sub>			
46	AV <sub>SS</sub>			
15, 70, 71, 92	V <sub>SS</sub>			
1	RES	Power-on reset circuit		
2, 3	XTAL, EXTAL	Oscillator circuit		
Other pins	NC (OPEN)			

Legend

- V<sub>PP</sub>: Programming power supply
- FO<sub>7</sub> to FO<sub>0</sub>: Data input/output
- FA<sub>16</sub> to FA<sub>0</sub>: Address input
- OE: Output enable
- CE: Chip enable
- WE: Write enable

Figure 19-14 Wiring of Socket Adapter

### 19.6.3 Operation in PROM Mode

The program/erase/verify specifications in PROM mode are the same as for the standard HN28F101 flash memory. However, since the H8/3434F does not support product name recognition mode, the programmer cannot be automatically set with the device name. Table 19-13 indicates how to select the various operating modes.

**Table 19-13 Operating Mode Selection in PROM Mode**

		Pins						
Mode		F <sub>V<sub>PP</sub></sub>	V <sub>CC</sub>	CE	OE	WE	D <sub>7</sub> to D <sub>0</sub>	A <sub>16</sub> to A <sub>0</sub>
Read	Read	V <sub>CC</sub>	V <sub>CC</sub>	L	L	H	Data output	Address input
	Output disable	V <sub>CC</sub>	V <sub>CC</sub>	L	H	H	High impedance	
	Standby	V <sub>CC</sub>	V <sub>CC</sub>	H	X	X	High impedance	
Command write	Read	V <sub>PP</sub>	V <sub>CC</sub>	L	L	H	Data output	
	Output disable	V <sub>PP</sub>	V <sub>CC</sub>	L	H	H	High impedance	
	Standby	V <sub>PP</sub>	V <sub>CC</sub>	H	X	X	High impedance	
	Write	V <sub>PP</sub>	V <sub>CC</sub>	L	H	L	Data input	

Note: Be sure to set the F<sub>V<sub>PP</sub></sub> pin to V<sub>CC</sub> in these states. If it is set to 0 V, hardware standby mode will be entered, even when in PROM mode, resulting in incorrect operation.

#### Legend

L: Low level

H: High level

V<sub>PP</sub>: V<sub>PP</sub> level

V<sub>CC</sub>: V<sub>CC</sub> level

X: Don't care

VH: 11.5 V ≤ VH ≤ 12.5 V

**Table 19-14 PROM Mode Commands**

Command	Cycles	Mode	1st Cycle			2nd Cycle	
			Address	Data	Mode	Address	Data
Memory read	1	Write	X	H'00	Read	RA	Dout
Erase setup/erase	2	Write	X	H'20	Write	X	H'20
Erase-verify	2	Write	EA	H'A0	Read	X	EVD
Auto-erase setup/ auto-erase	2	Write	X	H'30	Write	X	H'30
Program setup/ program	2	Write	X	H'40	Write	PA	PD
Program-verify	2	Write	X	H'C0	Read	X	PVD
Reset	2	Write	X	H'FF	Write	X	H'FF

PA: Program address

EA: Erase-verify address

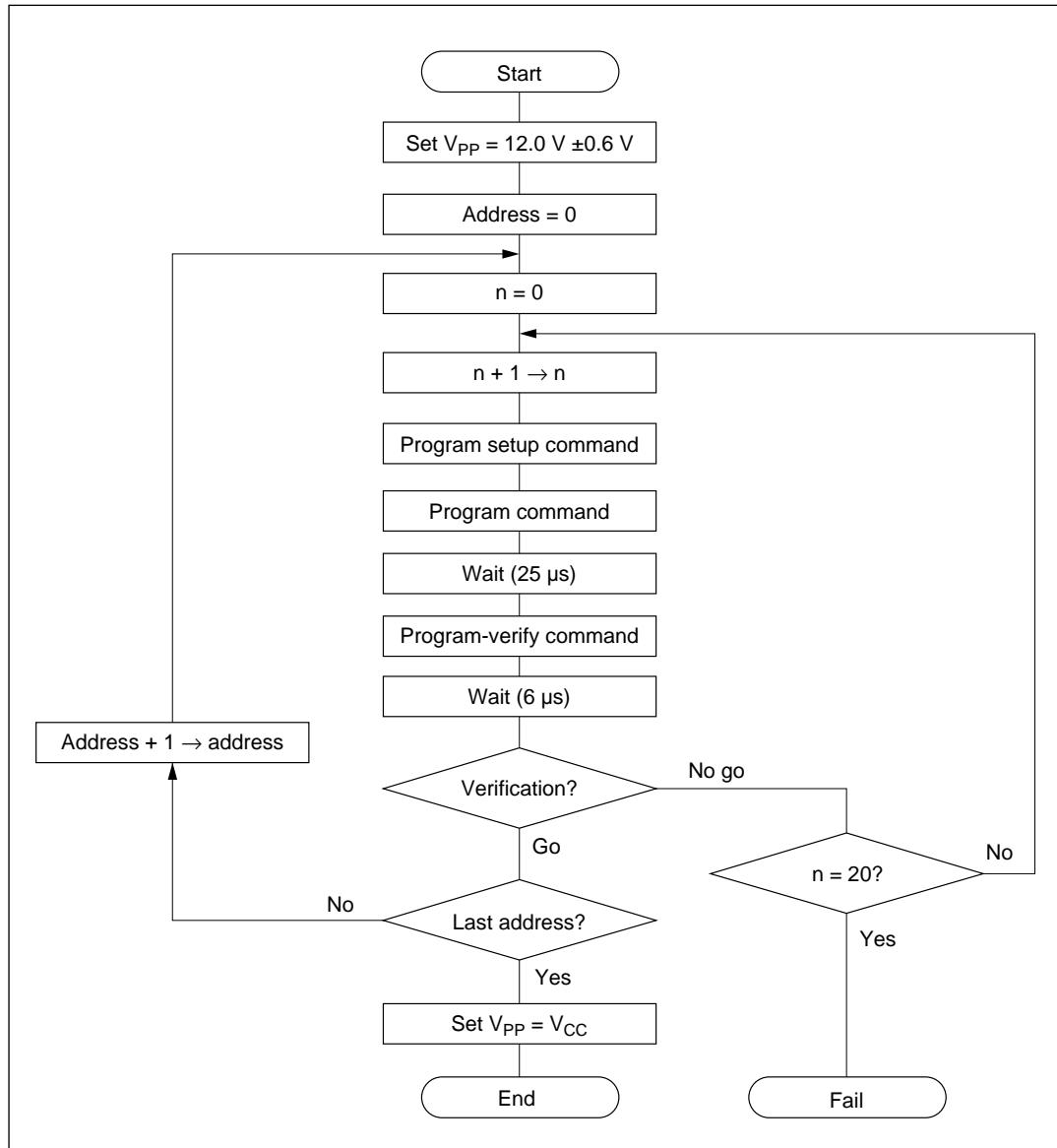
RA: Read address

PD: Program data

PVD: Program-verify output data

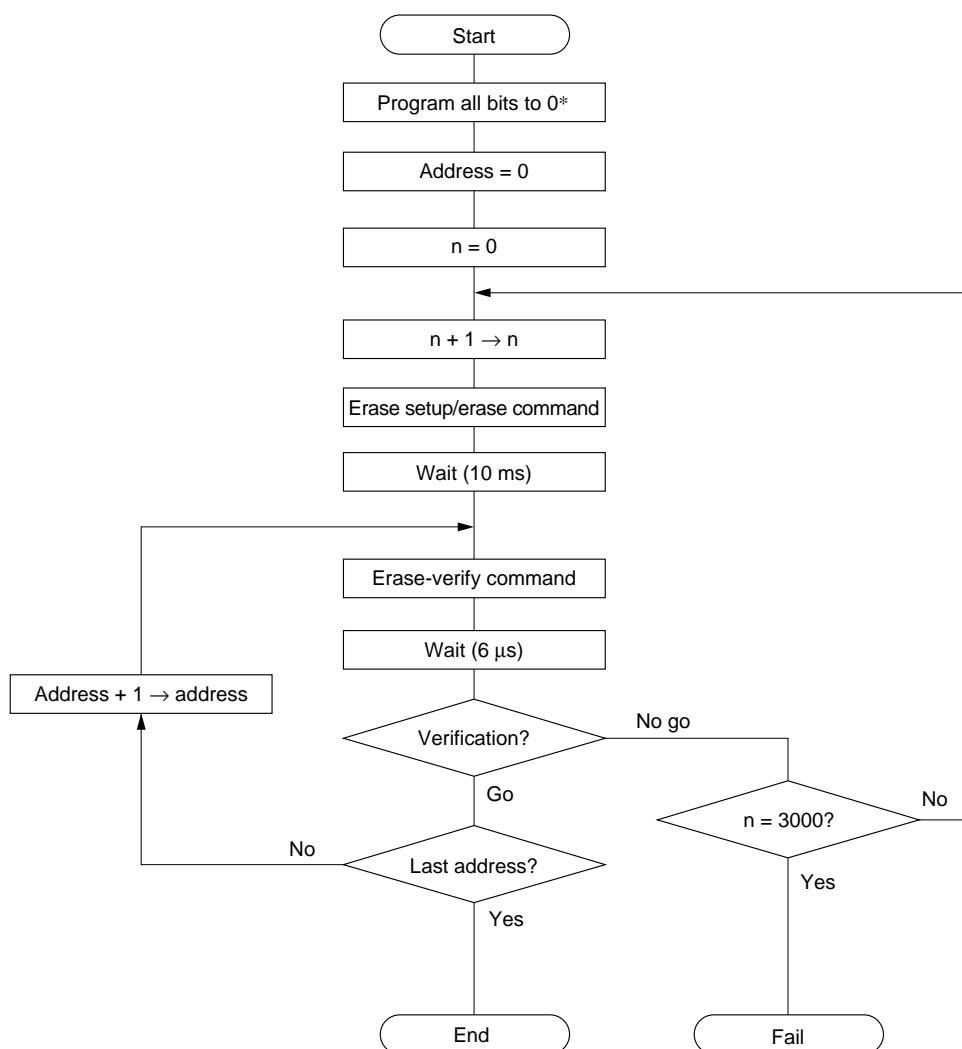
EVD: Erase-verify output data

**High-Speed, High-Reliability Programming:** Unused areas of the H8/3434F flash memory contain H'FF data (initial value). The H8/3434F flash memory uses a high-speed, high-reliability programming procedure. This procedure provides enhanced programming speed without subjecting the device to voltage stress and without sacrificing the reliability of programmed data. Figure 19-15 shows the basic high-speed, high-reliability programming flowchart. Tables 19-15 and 19-16 list the electrical characteristics during programming.



**Figure 19-15 High-Speed, High-Reliability Programming**

**High-Speed, High-Reliability Erasing:** The H8/3434F flash memory uses a high-speed, high-reliability erasing procedure. This procedure provides enhanced erasing speed without subjecting the device to voltage stress and without sacrificing data reliability. Figure 19-16 shows the basic high-speed, high-reliability erasing flowchart. Tables 19-15 and 19-16 list the electrical characteristics during erasing.



Note: \* Follow the high-speed, high-reliability programming flowchart in programming all bits. If some bits are already programmed to 0, program only the bits that have not yet been programmed.

**Figure 19-16 High-Speed, High-Reliability Erasing**

**Table 19-15 DC Characteristics in PROM Mode**(Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{PP} = 12.0 \text{ V} \pm 0.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input high voltage	FO <sub>7</sub> to FO <sub>0</sub> , FA <sub>16</sub> to FA <sub>0</sub> , OE, CE, WE	V <sub>IH</sub>	2.2	—	V <sub>CC</sub> + 0.3	V	
Input low voltage	FO <sub>7</sub> to FO <sub>0</sub> , FA <sub>16</sub> to FA <sub>0</sub> , OE, CE, WE	V <sub>IL</sub>	-0.3	—	0.8	V	
Output high voltage	FO <sub>7</sub> to FO <sub>0</sub>	V <sub>OH</sub>	2.4	—	—	V	I <sub>OH</sub> = -200 μA
Output low voltage	FO <sub>7</sub> to FO <sub>0</sub>	V <sub>OL</sub>	—	—	0.45	V	I <sub>OL</sub> = 1.6 mA
Input leakage current	FO <sub>7</sub> to FO <sub>0</sub> , FA <sub>16</sub> to FA <sub>0</sub> , OE, CE, WE	I <sub>LI</sub>	—	—	2	μA	V <sub>in</sub> = 0 to V <sub>CC</sub>
V <sub>CC</sub> current	Read	I <sub>CC</sub>	—	40	80	mA	
	Program	I <sub>CC</sub>	—	40	80	mA	
	Erase	I <sub>CC</sub>	—	40	80	mA	
FV <sub>PP</sub> current	Read	I <sub>PP</sub>	—	—	10	μA	V <sub>PP</sub> = 2.7 V to 5.5 V
			—	10	20	mA	V <sub>PP</sub> = 12.6 V
	Program	I <sub>PP</sub>	—	20	40	mA	V <sub>PP</sub> = 12.6 V
	Erase	I <sub>PP</sub>	—	20	40	mA	V <sub>PP</sub> = 12.6 V

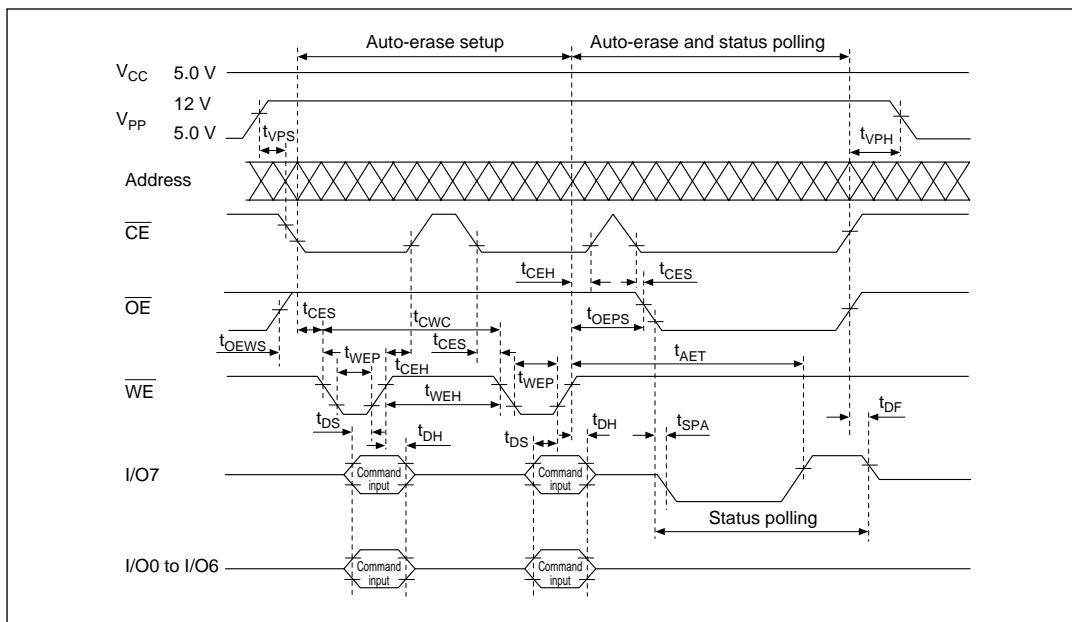
**Table 19-16 AC Characteristics in PROM Mode**(Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{PP} = 12.0 \text{ V} \pm 0.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Command write cycle	$t_{CWC}$	120	—	—	ns	Figure 19-17
Address setup time	$t_{AS}$	0	—	—	ns	Figure 19-18*
Address hold time	$t_{AH}$	60	—	—	ns	Figure 19-19
Data setup time	$t_{DS}$	50	—	—	ns	
Data hold time	$t_{DH}$	10	—	—	ns	
$\overline{CE}$ setup time	$t_{CES}$	0	—	—	ns	
$\overline{CE}$ hold time	$t_{CEH}$	0	—	—	ns	
$V_{PP}$ setup time	$t_{VPS}$	100	—	—	ns	
$V_{PP}$ hold time	$t_{VPH}$	100	—	—	ns	
$\overline{WE}$ programming pulse width	$t_{WEP}$	70	—	—	ns	
$\overline{WE}$ programming pulse high time	$t_{WEH}$	40	—	—	ns	
$\overline{OE}$ setup time before command write	$t_{OEWS}$	0	—	—	ns	
$\overline{OE}$ setup time before verify	$t_{OERS}$	6	—	—	$\mu\text{s}$	
Verify access time	$t_{VA}$	—	—	500	ns	
$\overline{OE}$ setup time before status polling	$t_{OEPS}$	120	—	—	ns	
Status polling access time	$t_{SPA}$	—	—	120	ns	
Program wait time	$t_{PPW}$	25	—	—	ns	
Erase wait time	$t_{ET}$	9	—	11	ms	
Output disable time	$t_{DF}$	0	—	40	ns	
Total auto-erase time	$t_{AET}$	0.5	—	30	s	

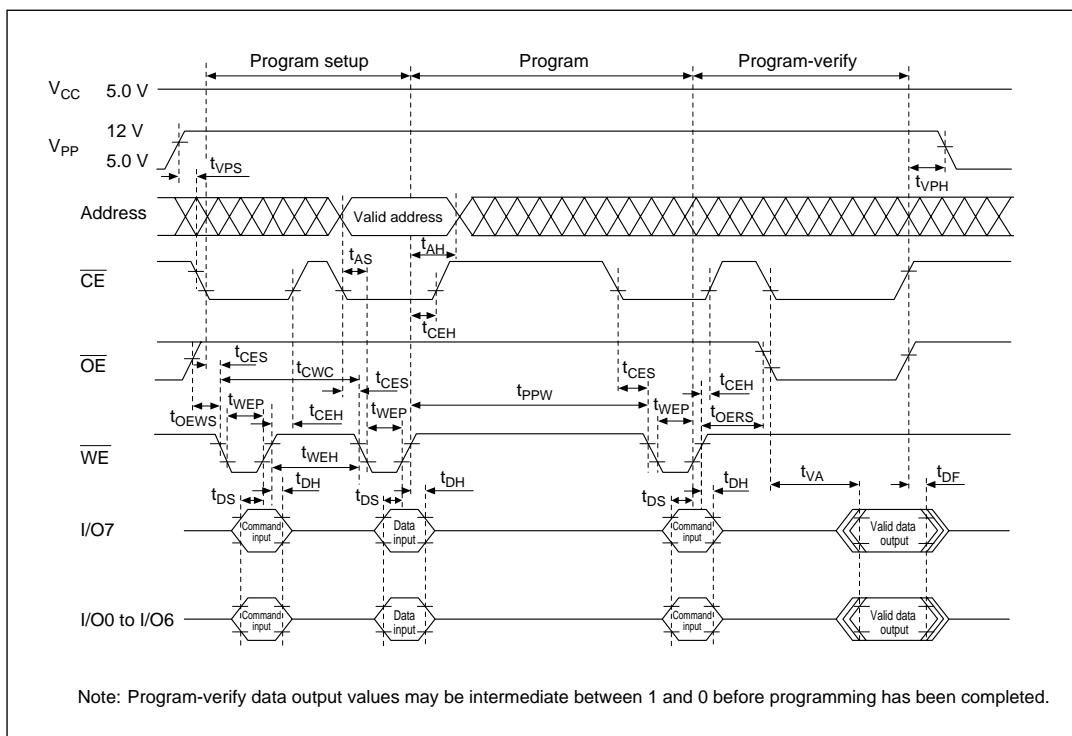
Note:  $\overline{CE}$ ,  $\overline{OE}$ , and  $\overline{WE}$  should be high during transitions of  $V_{PP}$  from 5 V to 12 V and from 12 V to 5 V.

\* Input pulse level: 0.45 V to 2.4 V  
 Input rise time and fall time  $\leq 10 \text{ ns}$

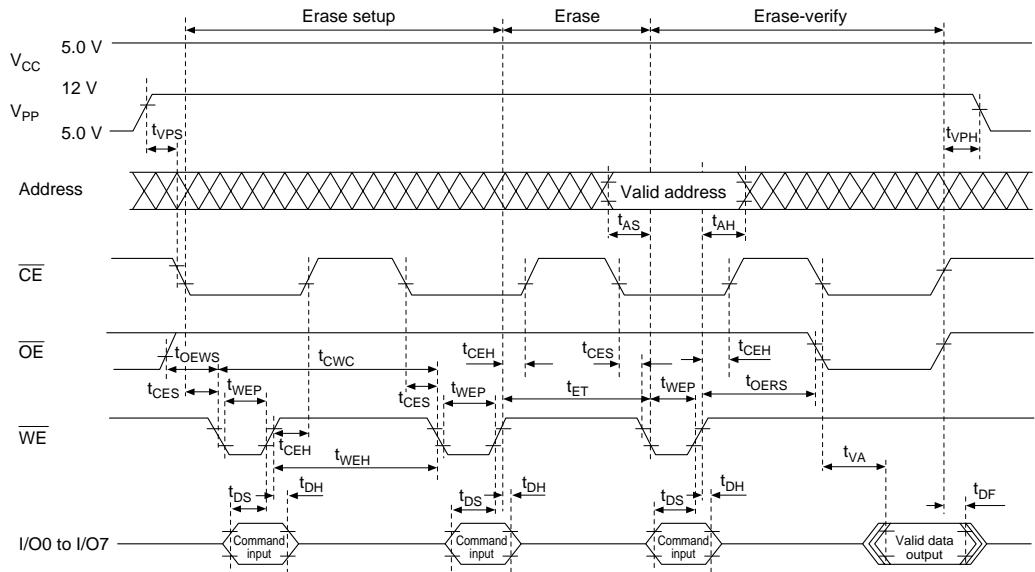
Timing reference levels: 0.8 V and 2.0 V for input; 0.8 V and 2.0 V for output



**Figure 19-17 Auto-Erase Timing**



**Figure 19-18** High-Speed, High-Reliability Programming Timing



**Figure 19-19** Erase Timing

## 19.7 Flash Memory Programming and Erasing Precautions

Read these precautions before using PROM mode, on-board programming mode, or flash memory emulation by RAM.

### (1) Program with the specified voltages and timing.

**The rated programming voltage ( $V_{PP}$ ) of the flash memory is 12.0 V.**

If the PROM programmer is set to Hitachi HN28F101 specifications,  $V_{PP}$  will be 12.0 V. Applying voltages in excess of the rating can permanently damage the device. Take particular care to ensure that the PROM programmer peak overshoot does not exceed the rated limit of 13 V.

### (2) Before programming, check that the chip is correctly mounted in the PROM programmer.

Overcurrent damage to the device can result if the index marks on the PROM programmer socket, 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) Set H'FF as the PROM programmer buffer data for addresses H'8000 to H'1FFFF.

The H8/3434F PROM size is 32 kbytes. Addresses H'8000 to H'1FFFF always read H'FF, so if H'FF is not specified as programmer data, a verify error will occur.

### (5) Notes on applying, releasing, and shutting off the programming voltage ( $V_{PP}$ )

Note: In this section, the application, release, and shutting-off of  $V_{PP}$  are defined as follows.

Application: A rise in voltage from  $V_{CC}$  to  $12\text{ V} \pm 0.6\text{ V}$ .

Release: A drop in voltage from  $12\text{ V} \pm 0.6\text{ V}$  to  $V_{CC}$ .

Shut-off: No applied voltage (floating).

- Apply the programming voltage ( $V_{PP}$ ) after the rise of  $V_{CC}$ , and release  $V_{PP}$  before shutting off  $V_{CC}$ .

To prevent unintended programming or erasing of flash memory, in these power-on and power-off timings, the application, release, and shutting-off of  $V_{PP}$  must take place when the microcontroller is in a stable operating condition as defined below.

#### Stable operating condition

- The  $V_{CC}$  voltage must be stabilized within the rated voltage range ( $V_{CC} = 2.7\text{ V}$  to  $5.5\text{ V}$ )

If  $V_{PP}$  is applied, released, or shut off while the microcontroller's  $V_{CC}$  voltage is not within the rated voltage range ( $V_{CC} = 2.7$  to  $5.5\text{ V}$ ), since microcontroller operation is unstable, the flash memory may be programmed or erased by mistake. This can occur even if  $V_{CC} = 0\text{ V}$ .

To prevent changes in the  $V_{CC}$  power supply when  $V_{PP}$  is applied, be sure that the power supply is adequately decoupled with inserting bypass capacitors.

- Clock oscillation must be stabilized (the oscillation settling time must have elapsed), and oscillation must not be stopped

When turning on  $V_{CC}$  power, hold the  $\overline{RES}$  pin low during the oscillation settling time ( $t_{OSC1} = 20$  ms), and do not apply  $V_{PP}$  until after this time.

- The microcontroller must be in the reset state, or in a state in which a reset has ended normally (reset has been released) and flash memory is not being accessed

Apply or release  $V_{PP}$  either in the reset state, or when the CPU is not accessing flash memory (when a program in on-chip RAM or external memory is executing). Flash memory cannot be read normally at the instant when  $V_{PP}$  is applied or released. Do not read flash memory while  $V_{PP}$  is being applied or released.

For a reset during operation, apply or release  $V_{PP}$  only after the  $\overline{RES}$  pin has been held low for at least ten system clock cycles (10 $\phi$ ).

- The P and E bits must be cleared in the flash memory control register (FLMCR)

When applying or releasing  $V_{PP}$ , make sure that the P or E bit is not set by mistake.

- No program runaway

When  $V_{PP}$  is applied, program execution must be supervised, e.g. by the watchdog timer.

These power-on and power-off timing requirements should also be satisfied in the event of a power failure and in recovery from a power failure. If these requirements are not satisfied, overprogramming or overerasing may occur due to program runaway etc., which could cause memory cells to malfunction.

- The  $V_{PP}$  flag is set and cleared by a threshold decision on the voltage applied to the  $FV_{PP}$  pin. The threshold level is between approximately  $V_{CC} + 2$  V to 11.4 V.

When this flag is set, it becomes possible to write to the flash memory control register (FLMCR) and the erase block registers (EBR1 and EBR2), even though the  $V_{PP}$  voltage may not yet have reached the programming voltage range of  $12.0 \pm 0.6$  V.

Do not actually program or erase the flash memory until  $V_{PP}$  has reached the programming voltage range.

The programming voltage range for programming and erasing flash memory is  $12.0 \pm 0.6$  V (11.4 V to 12.6 V). Programming and erasing cannot be performed correctly outside this range. When not programming or erasing the flash memory, ensure that the  $V_{PP}$  voltage does not

exceed the  $V_{CC}$  voltage. This will prevent unintended programming and erasing.

In this chip, the same pin is used for  $\overline{STBY}$  and  $FV_{PP}$ . When this pin is driven low, a transition is made to hardware standby mode. This happens not only in the normal operating modes (modes 1, 2, and 3), but also when programming the flash memory with a PROM programmer. When programming with a PROM programmer, therefore, use a programmer which sets this pin to the  $V_{CC}$  level when not programming ( $FV_{PP}=12$  V).

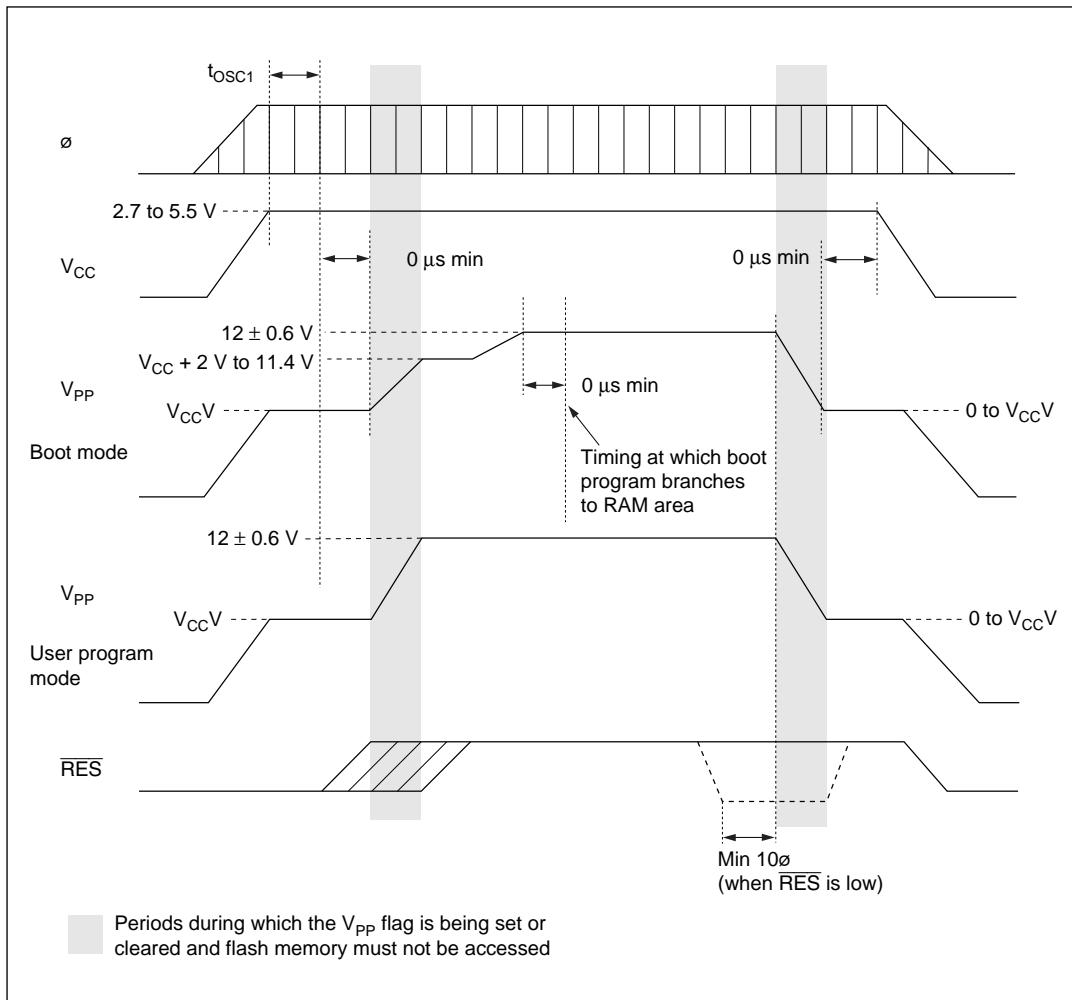


Figure 19-20  $V_{PP}$  Power-On and Power-Off Timing

**(6) Do not apply 12 V to the  $FV_{PP}$  pin during normal operation.**

To prevent accidental programming or erasing due to microcontroller program runaway etc., apply 12 V to the  $V_{PP}$  pin only when the flash memory is programmed or erased, or when flash

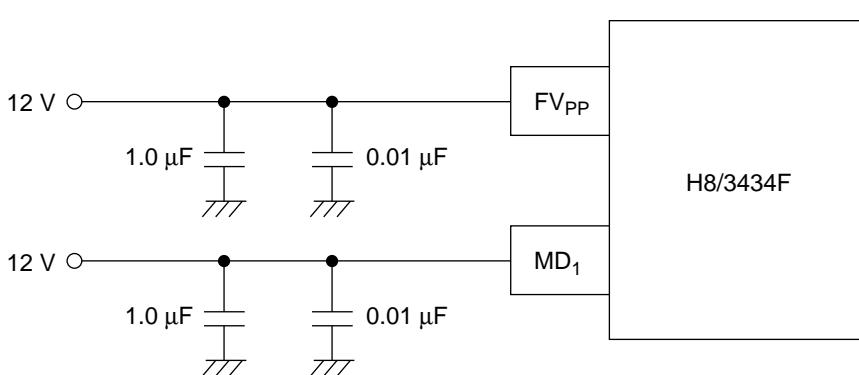
memory is emulated by RAM. Overprogramming or overerasing due to program runaway can cause memory cells to malfunction. Avoid system configurations in which 12 V is always applied to the FV<sub>PP</sub> pin.

While 12 V is applied, the watchdog timer should be running and enabled to halt runaway program execution, so that program runaway will not lead to overprogramming or overerasing.

(7) **Design a current margin into the programming voltage (V<sub>PP</sub>) power supply.** Ensure that V<sub>PP</sub> will not depart from  $12.0 \pm 0.6$  V (11.4 V to 12.6 V) during programming or erasing. Programming and erasing may become impossible outside this range.

(8) **Ensure that peak overshoot does not exceed the rated value at the FV<sub>PP</sub> and MD<sub>1</sub> pins.** Connect decoupling capacitors as close to the FV<sub>PP</sub> and MD<sub>1</sub> pins as possible.

Also connect decoupling capacitors to the MD<sub>1</sub> pin in the same way when boot mode is used.



**Figure 19-21 V<sub>PP</sub> Power Supply Circuit Design (Example)**

(9) **Use the recommended algorithms for programming and erasing flash memory.** These algorithms are designed to program and erase without subjecting the device to voltage stress and without sacrificing the reliability of programmed data.

Before setting the program (P) or erase (E) bit in the flash memory control register (FLMCR), set the watchdog timer to ensure that the P or E bit does not remain set for more than the specified time.

(10) For details on interrupt handling while flash memory is being programmed or erased, see the notes on NMI interrupt handling in section 19.4.9, Interrupt Handling during Flash Memory Programming and Erasing.

## (11) Cautions on Accessing Flash Memory Control Registers

### (a) Flash memory control register access state in each operating mode

The H8/3434F has flash memory control registers located at addresses H'FF80 (FLMCR), H'FF82 (EBR1), and H'FF83 (EBR2). These registers can only be accessed when 12 V is applied to the flash memory program power supply pin, FV<sub>PP</sub>.

Table 19-17 shows the area accessed for the above addresses in each mode, when 12 V is applied and is not applied to FV<sub>PP</sub>.

**Table 19-17 Area Accessed in Each Mode with 12V Applied and Not Applied to FV<sub>PP</sub>**

	<b>Mode 1</b>	<b>Mode 2</b>	<b>Mode 3</b>
12 V applied to FV <sub>PP</sub>	Reserved area (always H'FF)	Flash memory control register (initial value H'80)	Flash memory control register (initial value H'80)
12 V not applied to FV <sub>PP</sub>	External address space	External address space	Reserved area (always H'FF)

### (b) When a flash memory control register is accessed in mode 2 (expanded mode with on-chip ROM enabled)

When a flash memory control register is accessed in mode 2, it can be read or written to if 12 V is being applied to FV<sub>PP</sub>, but if not, external address space will be accessed. It is therefore essential to confirm that 12 V is being applied to the FV<sub>PP</sub> pin before accessing these registers.

### (c) To check for 12 V application/non-application in mode 3 (single-chip mode)

When address H'FF80 is accessed in mode 3, if 12 V is being applied to FV<sub>PP</sub>, FLMCR is read/written to, and its initial value after reset is H'80. When 12 V is not being applied to FV<sub>PP</sub>, FLMCR is a reserved area that cannot be modified and always reads H'FF. Since bit 7 (corresponding to the V<sub>PP</sub> bit) is set to 1 at this time regardless of whether 12 V is applied to FV<sub>PP</sub>, application or release of 12 V to FV<sub>PP</sub> cannot be determined simply from the 0 or 1 status of this bit. A byte data comparison is necessary to check whether 12V is being applied. The relevant coding is shown below.

```
LABEL1:    MOV.B    @H'FF80,    R1L
            CMP.B    #H'FF,    R1L
            BEQ     LABEL1
```

Sample program for detection of 12 V application to FV<sub>PP</sub> (mode 3)

**Table 19-18 DC Characteristics of Flash Memory**

Conditions:  $V_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{REF} = 2.7 \text{ V to } AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $V_{PP} = 12.0 \pm 0.6 \text{ V}$ ,  $T_a = -20^\circ\text{C} \text{ to } +75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C} \text{ to } +85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
High-voltage (12 V) threshold level*	$FV_{PP}$ , $MD_1$	$V_H$	$V_{CC} + 2$	—	11.4	V
$FV_{PP}$ current	During read	$I_{PP}$	—	10	$\mu\text{A}$	$V_{PP} = 2.7 \text{ to } 5.5 \text{ V}$
			—	10	20	$\text{mA}$
	During programming		—	20	40	$V_{PP} = 12.6 \text{ V}$
	During erasure		—	20	40	$\text{mA}$

Note: \* The listed voltages describe the threshold level at which high-voltage application is recognized. In boot mode and while flash memory is being programmed or erased, the applied voltage should be  $12.0 \text{ V} \pm 0.6 \text{ V}$ .

**Table 19-19 AC Characteristics of Flash Memory**

Conditions:  $V_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{REF} = 2.7 \text{ V to } AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $V_{PP} = 12.0 \pm 0.6 \text{ V}$ ,  $T_a = -20^\circ\text{C to } +75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to } +85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Programming time <sup>*1,*2</sup>	$t_P$	—	50	1000	$\mu\text{s}$	
Erase time <sup>*1,*3</sup>	$t_E$	—	1	30	s	
Number of writing/erasing count	$N_{WEC}$	—	—	100	Times	
Verify setup time 1 <sup>*1</sup>	$t_{VS1}$	4	—	—	$\mu\text{s}$	
Verify setup time 2 <sup>*1</sup>	$t_{VS2}$	2	—	—	$\mu\text{s}$	
Flash memory read setup time <sup>*4</sup>	$t_{FRS}$	50	—	—	$\mu\text{s}$	$V_{CC} \geq 4.5 \text{ V}$
		100	—	—		$V_{CC} < 4.5 \text{ V}$

Notes:

1. Set the times following the programming/erasing algorithm shown in section 19.
2. The programming time is the time during which a byte is programmed or the P bit in the flash memory control register (FLMCR) is set. It does not include the program-verify time.
3. The erase time is the time during which all 32-kbyte blocks are erased or the E bit in the flash memory control register (FLMCR) is set. It does not include the prewrite time before erasure or erase-verify time.
4. After power-on when using an external colck source, after return from standby mode, or after switching the programming voltage ( $V_{PP}$ ) from 12 V to  $V_{CC}$ , make sure that this read setup time has elapsed before reading flash memory.

When  $V_{PP}$  is released, the flash memory read setup time is defined as the period from when the FV<sub>PP</sub> pin has reached  $V_{CC} + 2 \text{ V}$  until flash memory can be read.

# Section 20 ROM (Flash memory 60 kbytes version)

## 20.1 Flash Memory Overview

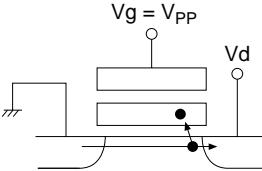
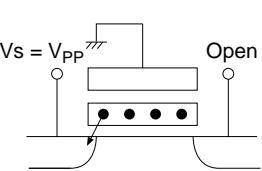
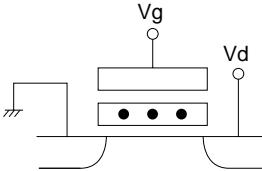
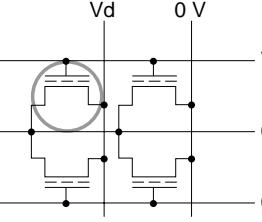
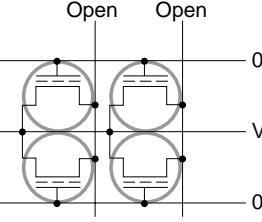
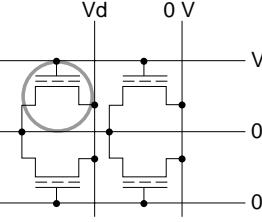
### 20.1.1 Flash Memory Operating Principle

Table 20-1 illustrates the principle of operation of the H8/3437F's on-chip flash memory.

Like EPROM, flash memory is programmed by applying a high gate-to-drain voltage that draws hot electrons generated in the vicinity of the drain into a floating gate. The threshold voltage of a programmed memory cell is therefore higher than that of an erased cell. Cells are erased by grounding the gate and applying a high voltage to the source, causing the electrons stored in the floating gate to tunnel out. After erasure, the threshold voltage drops. A memory cell is read like an EPROM cell, by driving the gate to the high level and detecting the drain current, which depends on the threshold voltage. Erasing must be done carefully, because if a memory cell is overerased, its threshold voltage may become negative, causing the cell to operate incorrectly.

Section 20.4.6 shows an optimal erase control flowchart and sample program.

**Table 20-1 Principle of Memory Cell Operation**

	Program	Erase	Read
Memory cell			
Memory array			

### 20.1.2 Mode Programming and Flash Memory Address Space

As its on-chip ROM, the H8/3437F has 60 kbytes of flash memory. The flash memory is connected to the CPU by a 16-bit data bus. The CPU accesses both byte data and word data in two states.

The H8/3437F's flash memory is assigned to addresses H'0000 to H'EF7F in mode 2, and addresses H'0000 to H'F77F in mode 3. The mode pins enable either on-chip flash memory or external memory to be selected for this area. Table 20-2 summarizes the mode pin settings and usage of the memory area.

**Table 20-2 Mode Pin Settings and Flash Memory Area**

Mode Pin Setting			
Mode	MD <sub>1</sub>	MD <sub>0</sub>	Memory Area Usage
Mode 0	0	0	Illegal setting
Mode 1	0	1	External memory area
Mode 2	1	0	On-chip flash memory area (H'0000 to H'EF7F)
Mode 3	1	1	On-chip flash memory area (H'0000 to H'F77F)

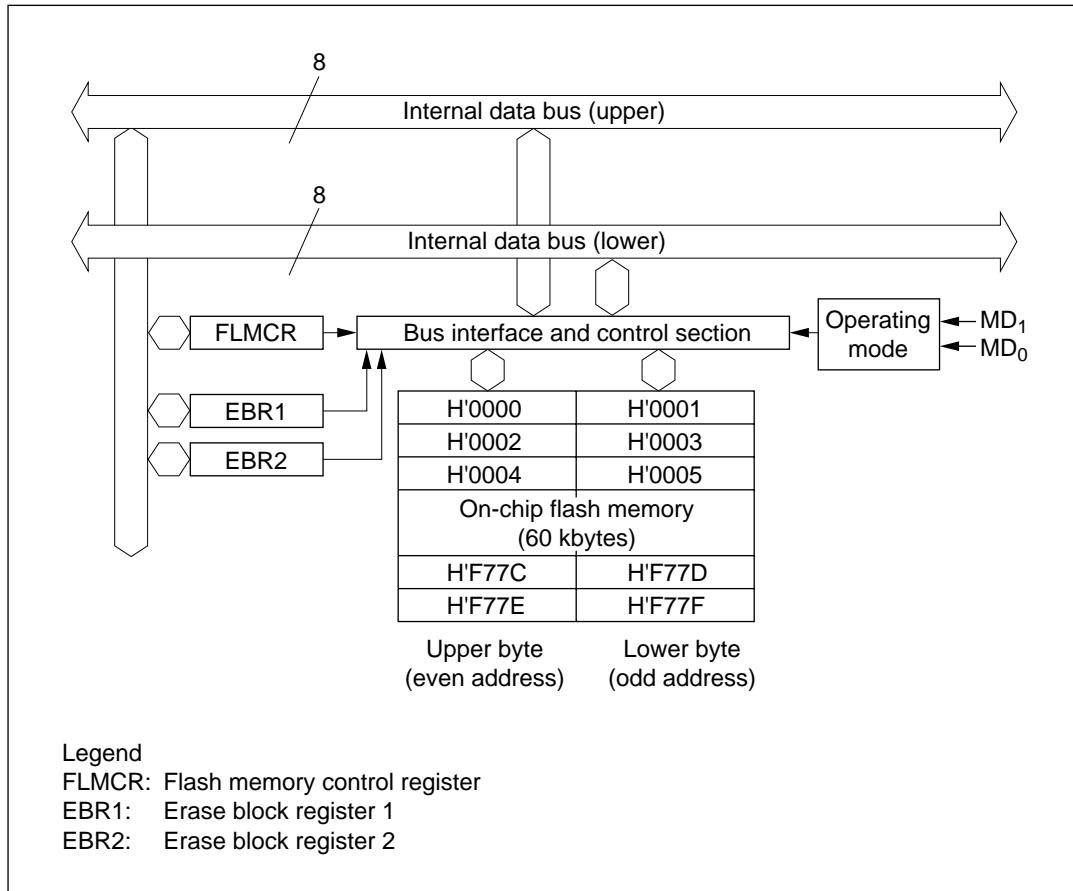
### 20.1.3 Features

Features of the flash memory are listed below.

- Five flash memory operating modes  
The flash memory has five operating modes: program mode, program-verify mode, erase mode, erase-verify mode, and prewrite-verify mode.
- Block erase designation  
Blocks to be erased in the flash memory address space can be selected by bit settings. The address space includes a large-block area (eight blocks with sizes from 2 kbytes to 12 kbytes) and a small-block area (eight blocks with sizes from 128 bytes to 1 kbyte).
- Program and erase time  
Programming one byte of flash memory typically takes 50 µs. Erasing all blocks (60 kbytes) typically takes 1 s.
- Erase-program cycles  
Flash memory contents can be erased and reprogrammed up to 100 times.
- On-board programming modes  
These modes can be used to program, erase, and verify flash memory contents. There are two modes: boot mode and user programming mode.
- Automatic bit-rate alignment  
In boot-mode data transfer, the H8/3437F aligns its bit rate automatically to the host bit rate (maximum 9600 bps).
- Flash memory emulation by RAM  
Part of the RAM area can be overlapped onto flash memory, to emulate flash memory updates in real time.
- PROM mode  
As an alternative to on-board programming, the flash memory can be programmed and erased in PROM mode, using a general-purpose PROM programmer. Program, erase, verify, and other specifications are the same as for HN28F101 standard flash memory.

#### 20.1.4 Block Diagram

Figure 20-1 shows a block diagram of the flash memory.



**Figure 20-1 Flash Memory Block Diagram**

## 20.1.5 Input/Output Pins

Flash memory is controlled by the pins listed in table 20-3.

**Table 20-3 Flash Memory Pins**

Pin Name	Abbreviation	Input/Output	Function
Programming power	FV <sub>PP</sub>	Power supply	Apply 12.0 V
Mode 1	MD <sub>1</sub>	Input	H8/3437F operating mode setting
Mode 0	MD <sub>0</sub>	Input	H8/3437F operating mode setting
Transmit data	TxD <sub>1</sub>	Output	SCI1 transmit data output
Receive data	RxD <sub>1</sub>	Input	SCI1 receive data input

The transmit data and receive data pins are used in boot mode.

## 20.1.6 Register Configuration

The flash memory is controlled by the registers listed in table 20-4.

**Table 20-4 Flash Memory Registers**

Name	Abbreviation	R/W	Initial Value	Address
Flash memory control register	FLMCR	R/W <sup>*2</sup>	H'00 <sup>*2</sup>	FF80
Erase block register 1	EBR1	R/W <sup>*2</sup>	H'00 <sup>*2</sup>	FF82
Erase block register 2	EBR2	R/W <sup>*2</sup>	H'00 <sup>*2</sup>	FF83
Wait-state control register <sup>*1</sup>	WSCR	R/W	H'08	FFC2

Notes: Registers FLMCR, EBR1, and EBR2 are only valid when writing to or erasing flash memory, and can only be accessed while 12 V is being applied to the FV<sub>PP</sub> pin. When 12 V is not applied to the FV<sub>PP</sub> pin, in mode 2 addresses H'FF80 to H'FF83 are external address space, and in mode 3 these addresses cannot be modified and always read H'FF.

1. The wait-state control register controls the insertion of wait states by the wait-state controller, frequency division of clock signals for the on-chip supporting modules by the clock pulse generator, and emulation of flash-memory updates by RAM in on-board programming mode.
2. In modes 2 and 3 (on-chip flash memory enabled), the initial value is H'00 for FLMCR, EBR1 and EBR2. In mode 1 (on-chip flash memory disabled), these registers cannot be modified and always read H'FF.

## 20.2 Flash Memory Register Descriptions

### 20.2.1 Flash Memory Control Register (FLMCR)

FLMCR is an 8-bit register that controls the flash memory operating modes. Transitions to program mode, erase mode, program-verify mode, and erase-verify mode are made by setting bits in this register. FLMCR is initialized to H'00 by a reset, in the standby modes, and when 12 V is not applied to FV<sub>PP</sub>. When 12 V is applied to the FV<sub>PP</sub> pin, a reset or entry to a standby mode initializes FLMCR to H'80.

Bit	7	6	5	4	3	2	1	0
	V <sub>PP</sub>	—	—	—	EV	PV	E	P
Initial value*	0	0	0	0	0	0	0	0
Read/Write	R	—	—	—	R/W*	R/W*	R/W*	R/W*

Note: \* The initial value is H'00 in modes 2 and 3 (on-chip flash memory enabled). In mode 1 (on-chip flash memory disabled), this register cannot be modified and always reads H'FF. For information on accessing this register, refer to in section 20.7, Flash Memory Programming and Erasing Precautions (11).

**Bit 7—Programming Power (V<sub>PP</sub>):** This status flag indicates that 12 V is applied to the FV<sub>PP</sub> pin. Refer to section 20.7, Flash Memory Programming and Erasing Precautions (5), for details on use.

Bit 7		
V <sub>PP</sub>	Description	
0	Cleared when 12 V is not applied to FV <sub>PP</sub>	(Initial value)
1	Set when 12 V is applied to FV <sub>PP</sub>	

**Bits 6 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Erase-Verify Mode (EV):**<sup>\*1</sup> Selects transition to or exit from erase-verify mode.

Bit 3		
EV	Description	
0	Exit from erase-verify mode	(Initial value)
1	Transition to erase-verify mode	

**Bit 2—Program-Verify Mode (PV):**<sup>\*1</sup> Selects transition to or exit from program-verify mode.

Bit 2		
PV	Description	
0	Exit from program-verify mode	(Initial value)
1	Transition to program-verify mode	

**Bit 1—Erase Mode (E):**\*1, \*2 Selects transition to or exit from erase mode.

**Bit 1**

**E**      **Description**

0	Exit from erase mode	(Initial value)
1	Transition to erase mode	

**Bit 0—Program Mode (P):**\*1, \*2 Selects transition to or exit from program mode.

**Bit 0**

**P**      **Description**

0	Exit from program mode	(Initial value)
1	Transition to program mode	

Notes: 1. Do not set two or more of these bits simultaneously. Do not release or shut off the  $V_{CC}$  or  $V_{PP}$  power supply when these bits are set.  
2. Set the P or E bit according to the instructions given in section 20.4, Programming and Erasing Flash Memory.  
Set the watchdog timer beforehand to make sure that these bits do not remain set for longer than the specified times.  
For notes on use, see section 20.7, Flash Memory Programming and Erasing Precautions.

### 20.2.2 Erase Block Register 1 (EBR1)

EBR1 is an 8-bit register that designates large flash-memory blocks for programming and erasure. EBR1 is initialized to H'00 by a reset, in the standby modes, and when 12 V is not applied to  $FV_{PP}$  pin. When a bit in EBR1 is set to 1, the corresponding block is selected and can be programmed and erased. Figure 20-2 and table 20-6 show details of a block map.

Bit	7	6	5	4	3	2	1	0
Initial value*1	LB7	LB6	LB5	LB4	LB3	LB2	LB1	LB0
Read/Write	R/W*1, *2	R/W*1						

Notes: \*1 The initial value is H'00 in modes 2 and 3 (on-chip ROM enabled). In mode 1 (on-chip ROM disabled), this register cannot be modified and always reads H'FF.

\*2 This bit cannot be modified in mode 2.

For information on accessing this register, refer to in section 20.7, Flash Memory Programming and Erasing Precautions (11).

**Bits 7 to 0—Large Block 7 to 0 (LB7 to LB0):** These bits select large blocks (LB7 to LB0) to be programmed and erased.

<b>Bits 7 to 0</b>		
<b>LB7 to LB0</b>	<b>Description</b>	
0	Block (LB7 to LB0) is not selected	(Initial value)
1	Block (LB7 to LB0) is selected	

### 20.2.3 Erase Block Register 2 (EBR2)

EBR2 is an 8-bit register that designates small flash-memory blocks for programming and erasure. EBR2 is initialized to H'00 by a reset, in the standby modes, and when 12 V is not applied to FV<sub>PP</sub> pin. When a bit in EBR2 is set to 1, the corresponding block is selected and can be programmed and erased. Figure 20-2 and table 20-6 show a block map.

Bit	7	6	5	4	3	2	1	0
	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Initial value*	0	0	0	0	0	0	0	0
Read/Write	R/W*							

Note: \* The initial value is H'00 in modes 2 and 3 (on-chip ROM enabled). In mode 1 (on-chip ROM disabled), this register cannot be modified and always reads H'FF.

For information on accessing this register, refer to in section 20.7, Flash Memory Programming and Erasing Precautions (11).

**Bits 7 to 0—Small Block 7 to 0 (SB7 to SB0):** These bits select small blocks (SB7 to SB0) to be programmed and erased.

<b>Bits 7 to 0</b>		
<b>SB6 to SB0</b>	<b>Description</b>	
0	Block (SB7 to SB0) is not selected	(Initial value)
1	Block (SB7 to SB0) is selected	

## 20.2.4 Wait-State Control Register (WSCR)

WSCR is an 8-bit readable/writable register that enables flash-memory updates to be emulated in RAM. It also controls frequency division of clock signals supplied to the on-chip supporting modules and insertion of wait states by the wait-state controller.

WSCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Bit	7	6	5	4	3	2	1	0
	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value	0	0	0	0	1	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Bits 7 and 6—RAM Select and RAM0 (RAMS and RAM0):** These bits are used to reassign an area to RAM (see table 20-5). These bits are write-enabled and their initial value is 0. They are initialized by a reset and in hardware standby mode. They are not initialized in software standby mode.

If only one of bits 7 and 6 is set, part of the RAM area can be overlapped onto the small-block flash memory area. In that case, access is to RAM, not flash memory, and all flash memory blocks are write/erase-protected (emulation protect<sup>\*1</sup>). In this state, the mode cannot be changed to program or erase mode, even if the P bit or E bit in the flash memory control register (FLMCR) is set (although verify mode can be selected). Therefore, clear both of bits 7 and 6 before programming or erasing the flash memory area.

If both of bits 7 and 6 are set, part of the RAM area can be overlapped onto the small-block flash memory area, but this overlapping begins only when an interrupt signal is input while 12 V is being applied to the FV<sub>PP</sub> pin. Up until that point, flash memory is accessed. Use this setting for interrupt handling while flash memory is being programmed or erased.<sup>\*2</sup>

**Table 20-5 RAM Area Reassignment<sup>\*3</sup>**

Bit 7 RAMS	Bit 6 RAM0	RAM Area	ROM Area
0	0	None	—
0	1	H'F880 to H'F8FF	H'0080 to H'00FF
1	0	H'F880 to H'F97F	H'0080 to H'017F
1	1	H'F800 to H'F87F	H'0000 to H'007F

**Bit 5—Clock Double (CKDBL):** Controls frequency division of clock signals supplied to the on-chip supporting modules. For details, see section 6, Clock Pulse Generator.

**Bit 4—Reserved:** This bit is reserved, but it can be written and read. Its initial value is 0.

#### **Bits 3 and 2—Wait Mode Select 1 and 0 (WMS1, WMS0)**

#### **Bits 1 and 0—Wait Count 1 and 0 (WC1, WC0)**

These bits control insertion of wait states by the wait-state controller. For details, see section 5, Wait-State Controller.

Notes:

1. For details on emulation protect, see section 20.4.8, Protect Modes.
2. For details on interrupt handling during programming and erasing of flash memory, see section 20.4.9, Interrupt Handling during Flash Memory Programming and Erasing.
3. RAM area that overlaps flash memory.

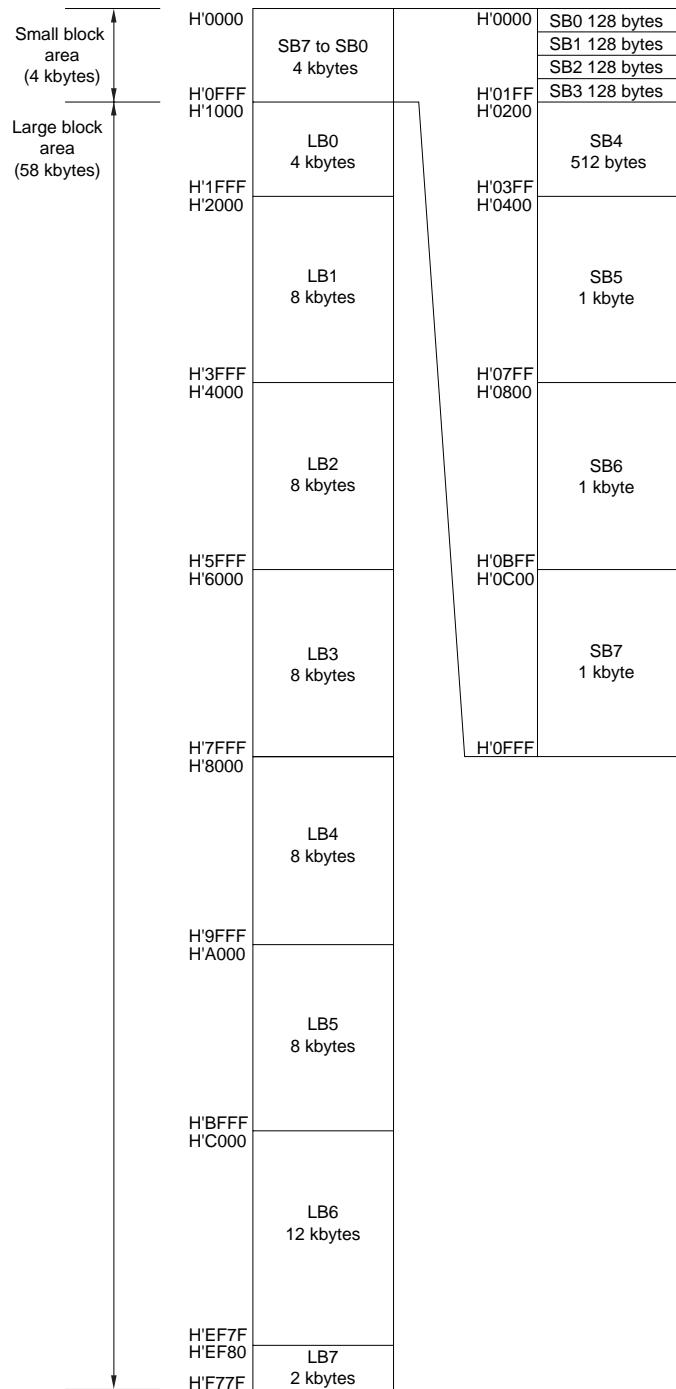


Figure 20-2 Erase Block Map

**Table 20-6 Erase Blocks and Corresponding Bits**

Register	Bit	Block	Address	Size
EBR1	0	LB0	H'1000 to H'1FFF	4 kbytes
	1	LB1	H'2000 to H'3FFF	8 kbytes
	2	LB2	H'4000 to H'5FFF	8 kbytes
	3	LB3	H'6000 to H'7FFF	8 kbytes
	4	LB4	H'8000 to H'9FFF	8 kbytes
	5	LB5	H'A000 to H'BFFF	8 kbytes
	6	LB6	H'C000 to H'EF7F	12 kbytes
	7	LB7	H'EF80 to H'F77F	2 kbytes
EBR2	0	SB0	H'0000 to H'007F	128 bytes
	1	SB1	H'0080 to H'00FF	128 bytes
	2	SB2	H'0100 to H'017F	128 bytes
	3	SB3	H'0180 to H'01FF	128 bytes
	4	SB4	H'0200 to H'03FF	512 bytes
	5	SB5	H'0400 to H'07FF	1 kbyte
	6	SB6	H'0800 to H'0BFF	1 kbyte
	7	SB7	H'0C00 to H'0FFF	1 kbyte

### 20.3 On-Board Programming Modes

When an on-board programming mode is selected, the on-chip flash memory can be programmed, erased, and verified. There are two on-board programming modes: boot mode, and user program mode. These modes are selected by inputs at the mode pins ( $MD_1$  and  $MD_0$ ) and  $FV_{PP}$  pin. Table 20-7 indicates how to select the on-board programming modes. For details on applying voltage  $V_{PP}$ , refer to section 20.7, Flash Memory Programming and Erasing Precautions (5).

**Table 20-7 On-Board Programming Mode Selection**

Mode Selections	FV <sub>PP</sub>	MD <sub>1</sub>	MD <sub>0</sub>	Notes
Boot mode	Mode 2	12 V*	12 V*	0: V <sub>IL</sub>
	Mode 3		1	1: V <sub>IH</sub>
User program mode	Mode 2		1	
	Mode 3		1	

Note: \* For details on the timing of 12 V application, see notes 6 to 8 in the Notes on Use of Boot Mode at the end of this section.

In boot mode, the mode control register (MDCR) can be used to monitor the mode (mode 2 or 3) in the same way as in normal mode.

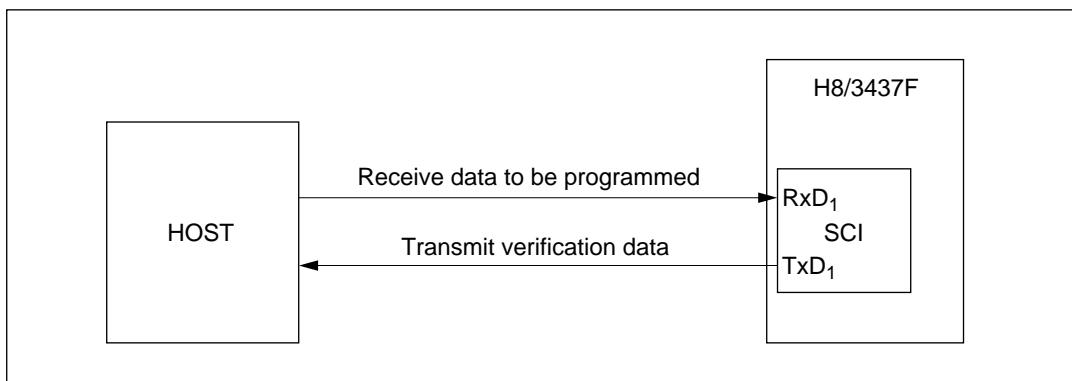
Example: Set the mode pins for mode 2 boot mode (MD<sub>1</sub> = 12 V, MD<sub>0</sub> = 0 V).

If the mode select bits of MDCR are now read, they will indicate mode 2 (MDS1 = 1, MDS0 = 0).

### 20.3.1 Boot Mode

To use boot mode, a user program for programming and erasing the flash memory must be provided in advance on the host machine (which may be a personal computer). Serial communication interface channel 1 is used in asynchronous mode (see figure 20-3). If the H8/3437F is placed in boot mode, after it comes out of reset, a built-in boot program is activated. This program starts by measuring the low period of data transmitted from the host and setting the bit rate register (BRR) accordingly. The H8/3437F's built-in serial communication interface (SCI) can then be used to download the user program from the host machine. The user program is stored in on-chip RAM.

After the program has been stored, execution branches to address H'F7E0 in the on-chip RAM, and the program stored on RAM is executed to program and erase the flash memory.



**Figure 20-3 Boot-Mode System Configuration**

## Boot-Mode Execution Procedure:

Figure 20-4 shows the boot-mode execution procedure.

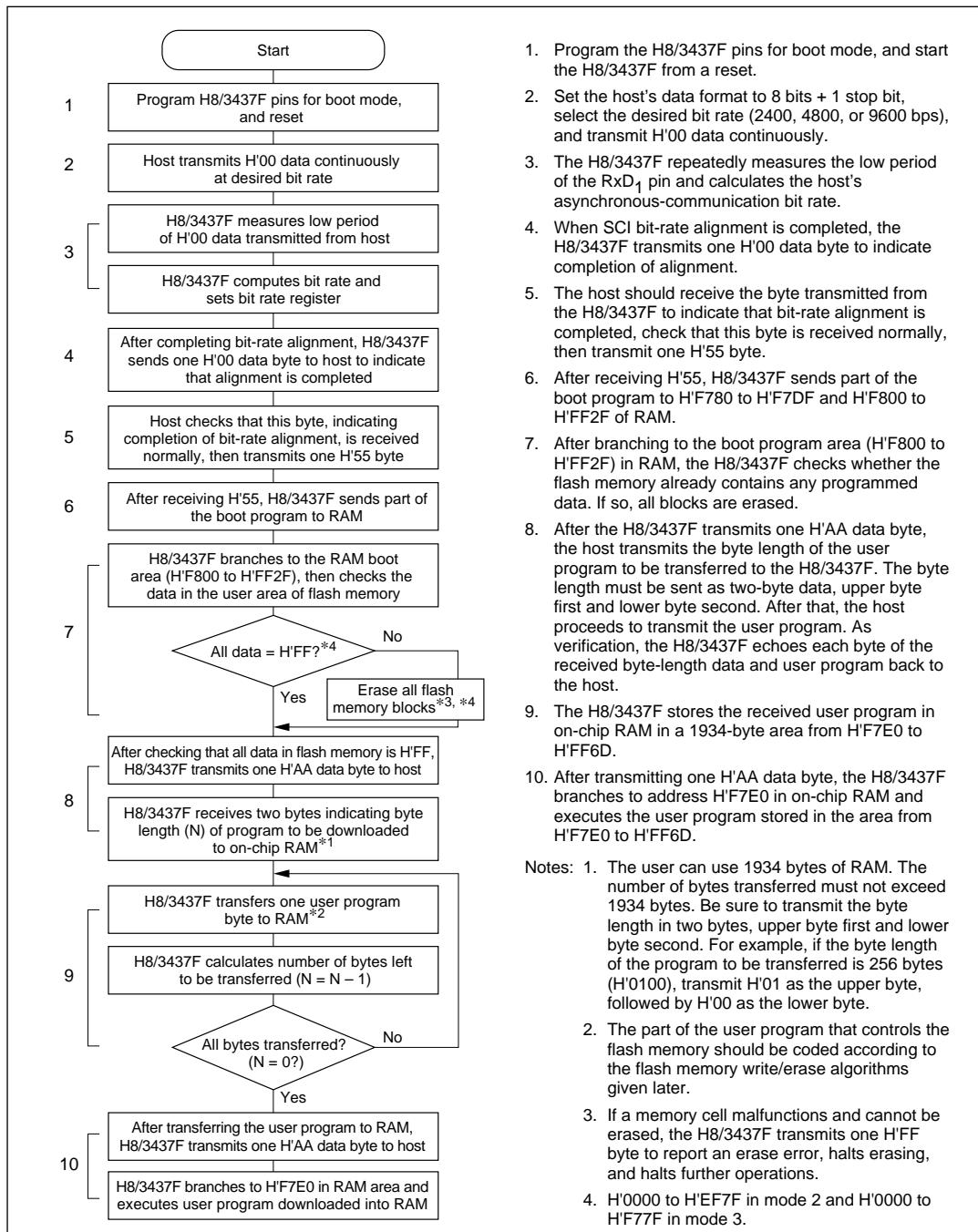
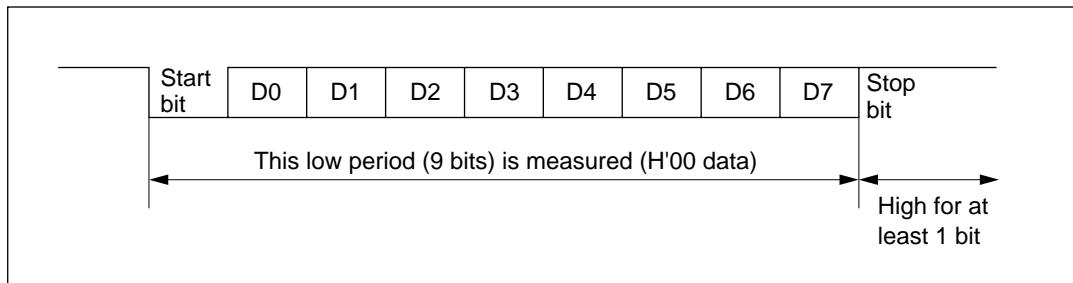


Figure 20-4 Boot Mode Flowchart

## Automatic Alignment of SCI Bit Rate



**Figure 20-5 Measurement of Low Period in Data Transmitted from Host**

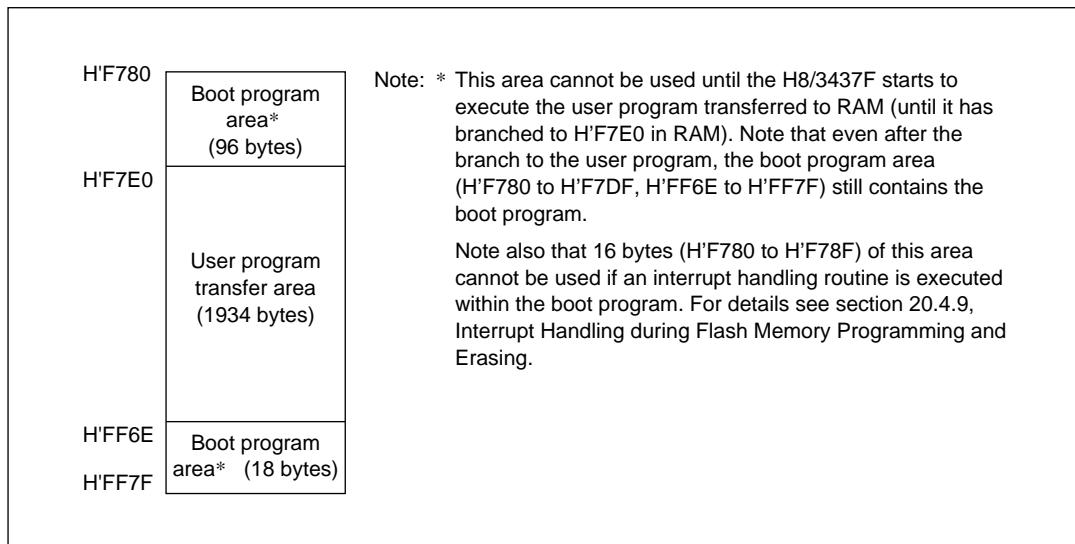
When started in boot mode, the H8/3437F measures the low period in asynchronous SCI data transmitted from the host (figure 20-5). The data format is eight data bits, one stop bit, and no parity bit. From the measured low period (9 bits), the H8/3437F computes the host's bit rate. After aligning its own bit rate, the H8/3437F sends the host 1 byte of H'00 data to indicate that bit-rate alignment is completed. The host should check that this alignment-completed indication is received normally and send one byte of H'55 back to the H8/3437F. If the alignment-completed indication is not received normally, the H8/3437F should be reset, then restarted in boot mode to measure the low period again. There may be some alignment error between the host's and H8/3437F's bit rates, depending on the host's bit rate and the H8/3437F's system clock frequency. To have the SCI operate normally, set the host's bit rate to 2400, 4800, or 9600 bps<sup>\*1</sup>. Table 20-8 lists typical host bit rates and indicates the clock-frequency ranges over which the H8/3437F can align its bit rate automatically. Boot mode should be used within these frequency ranges<sup>\*2</sup>.

**Table 20-8 System Clock Frequencies Permitting Automatic Bit-Rate Alignment by H8/3437F**

Host Bit Rate <sup>*1</sup>	System Clock Frequencies Permitting Automatic Bit-Rate Alignment by H8/3437F
9600 bps	8 MHz to 16 MHz
4800 bps	4 MHz to 16 MHz
2400 bps	2 MHz to 16 MHz

Notes: 1. Use a host bit rate setting of 2400, 4800, or 9600 bps only. No other setting should be used.  
2. Although the H8/3437F may also perform automatic bit-rate alignment with bit rate and system clock combinations other than those shown in table 20-8, there will be a slight difference between the bit rates of the host and the H8/3437F, and subsequent transfer will not be performed normally. Therefore, only a combination of bit rate and system clock frequency within one of the ranges shown in table 20-8 can be used for boot mode execution.

**RAM Area Allocation in Boot Mode:** In boot mode, the 96 bytes from H'F780 to H'F7DF and the 18 bytes from H'FF6E to H'FF7F are reserved for use by the boot program, as shown in figure 20-6. The user program is transferred into the area from H'F7E0 to H'FF6D (1934 bytes). The boot program area can be used after the transition to execution of the user program transferred into RAM. If a stack area is needed, set it within the user program.



**Figure 20-6 RAM Areas in Boot Mode**

## Notes on Use of Boot Mode

1. When the H8/3437F comes out of reset in boot mode, it measures the low period of the input at the SCI's  $\text{RxD}_1$  pin. The reset should end with  $\text{RxD}_1$  high. After the reset ends, it takes about 100 states for the H8/3437F to get ready to measure the low period of the  $\text{RxD}_1$  input.
2. In boot mode, if any data has been programmed into the flash memory (if all data<sup>\*3</sup> is not H'FF), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, e.g. the first time on-board programming is performed, or if the update program activated in user program mode is accidentally erased.
3. Interrupts cannot be used while the flash memory is being programmed or erased.
4. The  $\text{RxD}_1$  and  $\text{TxD}_1$  pins should be pulled up on-board.
5. Before branching to the user program (at address H'F7E0 in the RAM area), the H8/3437F terminates transmit and receive operations by the on-chip SCI (by clearing the RE and TE bits of the serial control register to 0 in channel 1), but the auto-aligned bit rate remains set in bit rate register BRR. The transmit data output pin ( $\text{TxD}_1$ ) is in the high output state (in port 8, the bits P8<sub>4</sub> DDR of the port 8 data direction register and P8<sub>4</sub> DR of the port 8 data register are set to 1).

At this time, the values of general registers in the CPU are undetermined. Thus these registers should be initialized immediately after branching to the user program. Especially in the case of the stack pointer, which is used implicitly in subroutine calls, the stack area used by the user program should be specified.

There are no other changes to the initialized values of other registers.

6. Boot mode can be entered by starting from a reset after 12 V is applied to the  $\text{MD}_1$  and  $\text{FV}_{\text{PP}}$  pins according to the mode setting conditions listed in table 20-7. Note the following points when turning the  $\text{V}_{\text{PP}}$  power on.

When reset is released (at the rise from low to high), the H8/3437F checks for 12-V input at the  $\text{MD}_1$  and  $\text{FV}_{\text{PP}}$  pins. If it detects that these pins are programmed for boot mode, it saves that status internally. The threshold point of this voltage-level check is in the range from approximately  $\text{V}_{\text{CC}} + 2$  V to 11.4 V, so boot mode will be entered even if the applied voltage is insufficient for programming or erasure (11.4 V to 12.6 V). When the boot program is executed, the  $\text{V}_{\text{PP}}$  power supply must therefore be stabilized within the range of 11.4 V to 12.6 V before the branch to the RAM area occurs. See figure 20-20.

Make sure that the programming voltage  $\text{V}_{\text{PP}}$  does not exceed 12.6 V during the transition to boot mode (at the reset release timing) and does not go outside the range of 12 V  $\pm$  0.6 V while in boot mode. Boot mode will not be executed correctly if these limits are exceeded. In addition, make sure that  $\text{V}_{\text{PP}}$  is not released or shut off while the boot program is executing or

the flash memory is being programmed or erased.\*1

Boot mode can be released by driving the reset pin low, waiting at least ten system clock cycles, then releasing the application of 12 V to the MD<sub>1</sub> and FV<sub>PP</sub> pins and releasing the reset.

The settings of external pins must not change during operation in boot mode.

During boot mode, if input of 12 V to the MD<sub>1</sub> pin stops but no reset input occurs at the RES pin, the boot mode state is maintained within the chip and boot mode continues (but do not stop applying 12 V to the FV<sub>PP</sub> pin during boot mode\*1).

If a watchdog timer reset occurs during boot mode, this does not release the internal mode state, but the internal boot program is restarted.

Therefore, to change from boot mode to another mode, the boot-mode state within the chip must be released by a reset input at the RES pin before the mode transition can take place.

7. If the input level of the MD<sub>1</sub> pin is changed during a reset (e.g., from 0 V to 5 V then to 12 V while the input to the RES pin is low), the resultant switch in the microcontroller's operating mode will affect the bus control output signals (AS, RD, and WR) and the status of ports that can be used for address output\*2.

Therefore, either set these pins so that they do not output signals during the reset, or make sure that their output signals do not collide with other signals outside the microcontroller.

8. When applying 12 V to the MD<sub>1</sub> and FV<sub>PP</sub> pins, make sure that peak overshoot does not exceed the rated limit of 13 V.

Also, be sure to connect a decoupling capacitor to the FV<sub>PP</sub> and MD<sub>1</sub> pins.

Note: 1. For details on applying, releasing, and shutting off V<sub>PP</sub>, see note (5) in section 20.7, Flash Memory Programming and Erasing Precautions.

2. These ports output low-level address signals if the mode pins are set to mode 1 during the reset. In all other modes, these ports are in the high-impedance state. The bus control output signals are high if the mode pins are set for mode 1 or 2 during the reset. In mode 3, they are at high impedance.
3. H'0000 to H'EF7F in mode 2 and H'0000 to H'F77F in mode 3.

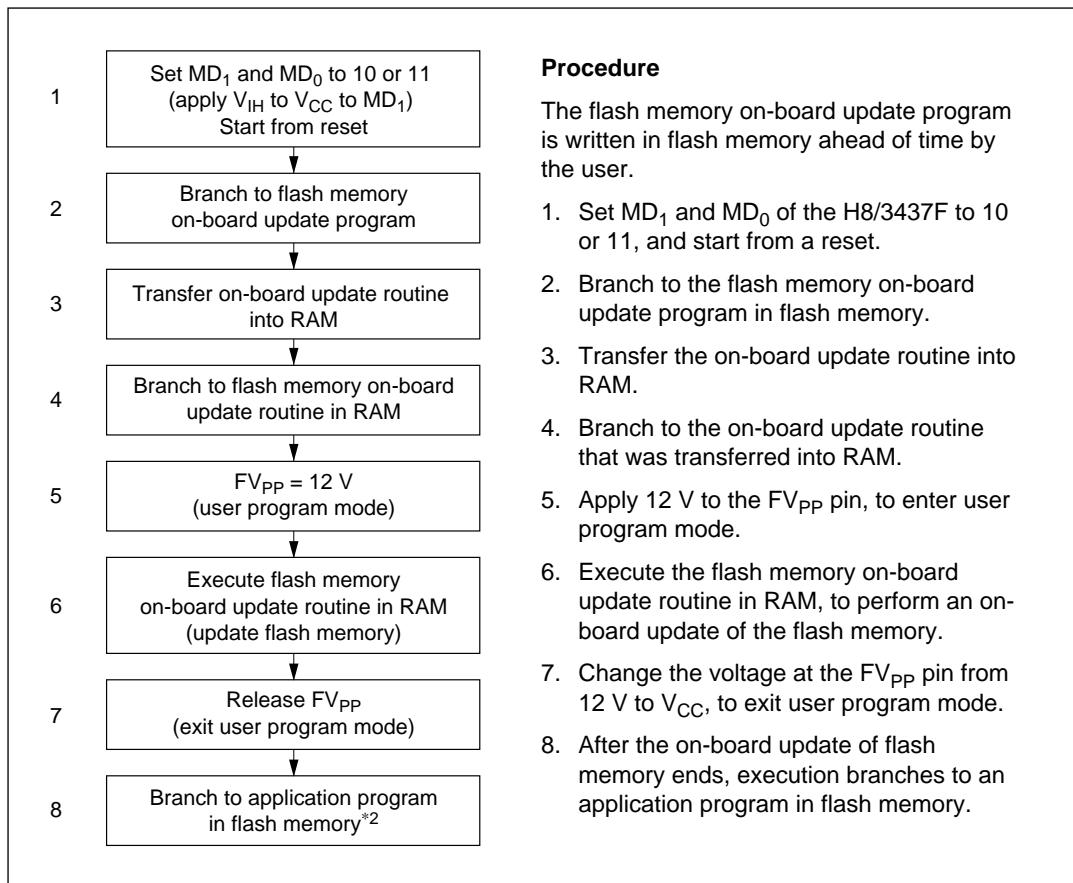
### 20.3.2 User Program Mode

When set to user program mode, the H8/3437F can erase and program its flash memory by executing a user program. On-board updates of the on-chip flash memory can be carried out by providing on-board circuits for supplying  $V_{PP}$  and data, and storing an update program in part of the program area.

To select user program mode, select a mode that enables the on-chip ROM (mode 2 or 3) and apply 12 V to the  $FV_{PP}$  pin, either during a reset, or after the reset has ended (been released) but while flash memory is not being accessed. In user program mode, the on-chip supporting modules operate as they normally would in mode 2 or 3, except for the flash memory. However, hardware standby mode cannot be set while 12 V is applied to the  $FV_{PP}$  pin.

The flash memory cannot be read while it is being programmed or erased, so the update program must either be stored in external memory, or transferred temporarily to the RAM area and executed in RAM.

**User Program Mode Execution Procedure (Example)\*1:** Figure 20-7 shows the execution procedure for user program mode when the on-board update routine is executed in RAM.



**Figure 20-7 User Program Mode Operation (Example)**

Notes:

1. Do not apply 12 V to the FV<sub>PP</sub> pin during normal operation. To prevent flash memory from being accidentally programmed or erased due to program runaway etc., apply 12 V to FV<sub>PP</sub> only when programming or erasing flash memory. Overprogramming or overerasing due to program runaway can cause memory cells to malfunction. While 12 V is applied, the watchdog timer should be running and enabled to halt runaway program execution, so that program runaway will not lead to overprogramming or overerasing. For details on applying, releasing, and shutting off V<sub>PP</sub>, see section 20.7, Flash Memory Programming and Erasing Precautions (5).
2. After the update is finished, when input of 12 V to the FV<sub>PP</sub> pin is released, the flash memory read setup time (t<sub>FRS</sub>) must elapse before any program in flash memory is executed. This is the required setup time from when the FV<sub>PP</sub> pin reaches the (V<sub>CC</sub> + 2 V) level after 12 V is released until flash memory can be read.

## 20.4 Programming and Erasing Flash Memory

The H8/3437F's on-chip flash memory is programmed and erased by software, using the CPU. The flash memory can operate in program mode, erase mode, program-verify mode, erase-verify mode, or prewrite-verify mode. Transitions to these modes can be made by setting the P, E, PV, and EV bits in the flash memory control register (FLMCR).

The flash memory cannot be read while being programmed or erased. The program that controls the programming and erasing of the flash memory must be stored and executed in on-chip RAM or in external memory. A description of each mode is given below, with recommended flowcharts and sample programs for programming and erasing.

For details on programming and erasing, refer to section 20.7, Flash Memory Programming and Erasing Precautions.

### 20.4.1 Program Mode

To write data into the flash memory, follow the programming algorithm shown in figure 20-8. This programming algorithm can write data without subjecting the device to voltage stress or impairing the reliability of programmed data.

To program data, first specify the area to be written in flash memory with erase block registers EBR1 and EBR2, then write the data to the address to be programmed, as in writing to RAM. The flash memory latches the address and data in an address latch and data latch. Next set the P bit in FLMCR, selecting program mode. The programming duration is the time during which the P bit is set. The total programming time does not exceed 1 ms. Programming for too long a time, due to program runaway for example, can cause device damage. Before selecting program mode, set up the watchdog timer so as to prevent overprogramming. For details of the programming method, refer to section 20.4.3, Programming Flowchart and Sample Programs.

### 20.4.2 Program-Verify Mode

In program-verify mode, after data has been programmed in program mode, the data is read to check that it has been programmed correctly.

After the programming time has elapsed, exit programming mode (clear the P bit to 0) and select program-verify mode (set the PV bit to 1). In program-verify mode, a program-verify voltage is applied to the memory cells at the latched address. If the flash memory is read in this state, the data at the latched address will be read. After selecting program-verify mode, wait 4  $\mu$ s or more before reading, then compare the programmed data with the verify data. If they agree, exit program-verify mode and program the next address. If they do not agree, select program mode again and repeat the same program and program-verify sequence. Do not repeat the program and program-verify sequence more than 6 times\* for the same bit.

Note: \* Keep the total programming time under 1 ms for each bit.

### 20.4.3 Programming Flowchart and Sample Program

#### Flowchart for Programming One Byte

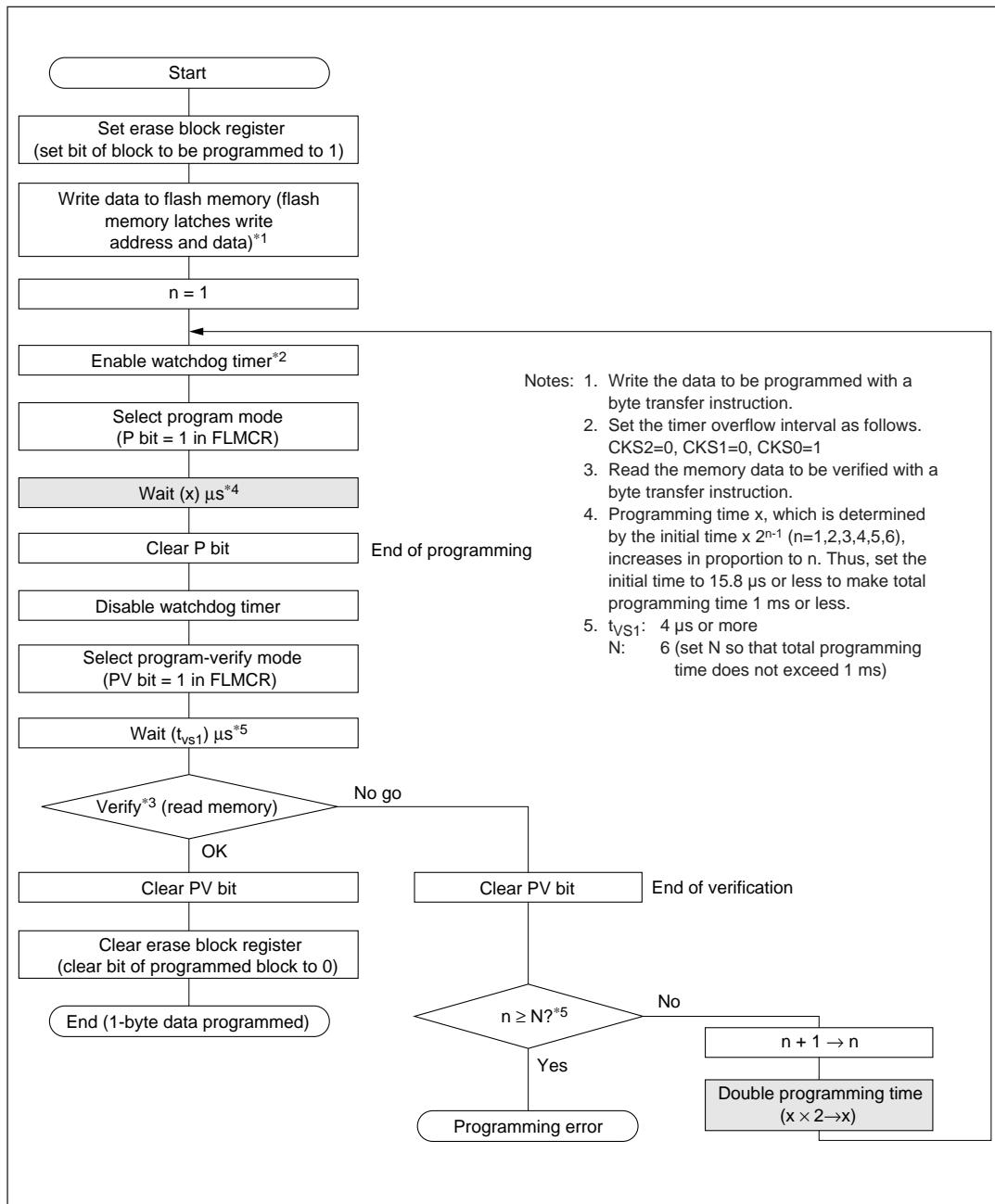


Figure 20-8 Programming Flowchart

**Sample Program for Programming One Byte:** This program uses the following registers.

R0H: Specifies blocks to be erased.

R1H: Stores data to be programmed.

R1L: Stores data to be read.

R3: Stores address to be programmed. Valid address specifications are H'0000 to H'EF7F in mode 2, and H'0000 to H'F77F in mode 3.

R4: Sets program and program-verify timing loop counters, and also stores register setting value.

R5: Sets program timing loop counter.

R6L: Used for program-verify fail count.

Arbitrary data can be programmed at an arbitrary address by setting the address in R3 and the data in R1H.

The setting of #a and #b values depends on the clock frequency. Set #a and #b values according to tables 20-9 (1) and (2).

```
FLMCR:    .EQU      H'FF80
EBR1:     .EQU      H'FF82
EBR2:     .EQU      H'FF83
TCSR:     .EQU      H'FFA8

        .ALIGN    2
PRGM:    MOV.B     #H'**,    R0H      ;
        MOV.B     R0H,     @EBR*:8    ; Set EBR*
                  .          .
                  .          .
                  .          .
PRGMS:   MOV.B     #H'00,    R6L      ; Program-verify fail counter
        MOV.W     #H'a,     R5       ; Set program loop counter
        MOV.B     R1H,     @R3      ; Dummy write
        INC      R6L      ; Program-verify fail counter + 1 → R6L
        MOV.W     #H'A579,  R4       ;
        MOV.W     R4,      @TCSR    ; Start watchdog timer
        MOV.W     R5,      R4       ; Set program loop counter
        BSET    #0,      @FLMCR:8 ; Set P bit
LOOP1:   SUBS   #1,      R4       ;
        MOV.W     R4,      R4       ;
        BNE    LOOP1    ; Wait loop
        BCLR   #0,      @FLMCR:8 ; Clear P bit
        MOV.W     #H'A500,  R4       ;
        MOV.W     R4,      @TCSR    ; Stop watchdog timer
                  .          .
                  .          .
                  .          .
LOOP2:   MOV.B     #H'b,    R4H      ; Set program-verify loop counter
        BSET   #2,      @FLMCR:8 ; Set PV bit
        DEC     R4H      ;
        BNE    LOOP2    ; Wait loop
        MOV.B     @R3,    R1L      ; Read programmed address
        CMP.B     R1H,    R1L      ; Compare programmed data with read data
        BEQ     PVOKE   ; Program-verify decision
        BCLR   #2,      @FLMCR:8 ; Clear PV bit
```

CMP.B	#H'06,	R6L	; Program-verify executed 6 times?
BEQ	NGEND		; If program-verify executed 6 times, branch to NGEND
ADD.W	R5,	R5	; Programming time $\times 2$
BRA	PRGMS		; Program again
PVOK:	BCLR	#2,	@FLMCR:8 ; Clear PV bit
	MOV.B	#H'00,	R6L ;
	MOV.B	R6L,	@EBR*:8 ; Clear EBR*

One byte programmed

NGEND: Programming error

#### 20.4.4 Erase Mode

To erase the flash memory, follow the erasing algorithm shown in figure 20-9. This erasing algorithm can erase data without subjecting the device to voltage stress or impairing the reliability of programmed data.

To erase flash memory, before starting to erase, first place all memory data in all blocks to be erased in the programmed state (program all memory data to H'00). If all memory data is not in the programmed state, follow the sequence described later (figure 20-10) to program the memory data to zero. Select the flash memory areas to be erased with erase block registers 1 and 2 (EBR1 and EBR2). Next set the E bit in FLMCR, selecting erase mode. The erase time is the time during which the E bit is set. To prevent overerasing, use a software timer to divide the time for a single erase, and ensure that the total time does not exceed 30 seconds. For the time for a single erase, refer to section 20.4.6, Erase Flowchart and Sample Programs. Overerasing, due to program runaway for example, can give memory cells a negative threshold voltage and cause them to operate incorrectly. Before selecting erase mode, set up the watchdog timer so as to prevent overerasing.

#### 20.4.5 Erase-Verify Mode

In erase-verify mode, after data has been erased, it is read to check that it has been erased correctly. After the erase time has elapsed, exit erase mode (clear the E bit to 0) and select erase-verify mode (set the EV bit to 1). Before reading data in erase-verify mode, write H'FF dummy data to the address to be read. This dummy write applies an erase-verify voltage to the memory cells at the latched address. If the flash memory is read in this state, the data at the latched address will be read. After the dummy write, wait 2  $\mu$ s or more before reading. When performing the initial dummy write, wait 4  $\mu$ s or more after selecting erase-verify mode. If the read data has been successfully erased, perform an erase-verify (dummy write, wait 2  $\mu$ s or more, then read) for the next address. If the read data has not been erased, select erase mode again and repeat the same erase and erase-verify sequence through the last address, until all memory data has been erased to 1. Do not repeat the erase and erase-verify sequence more than 602 times, however.

## 20.4.6 Erasing Flowchart and Sample Program

### Flowchart for Erasing One Block

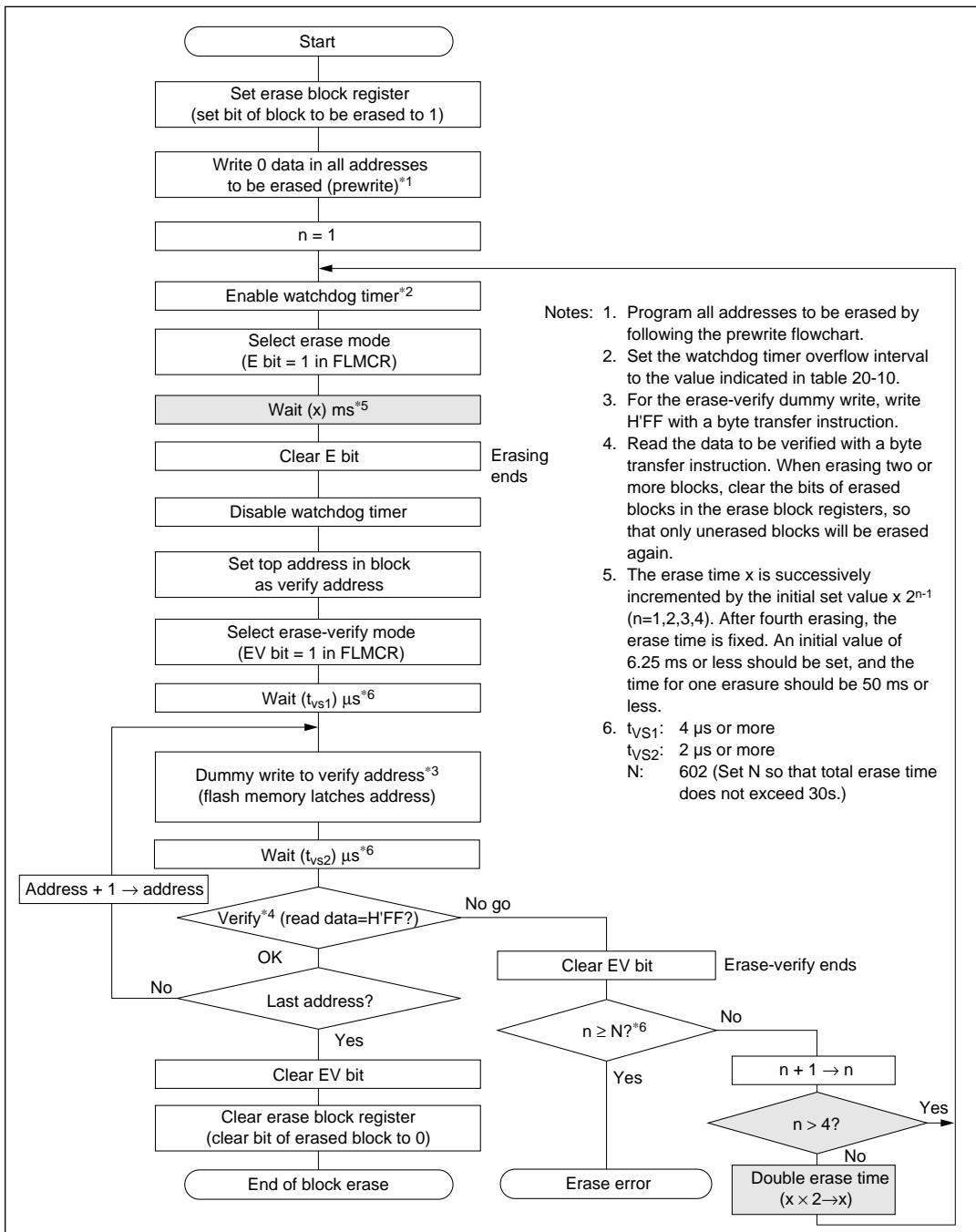


Figure 20-9 Erasing Flowchart

## Prewrite Flowchart

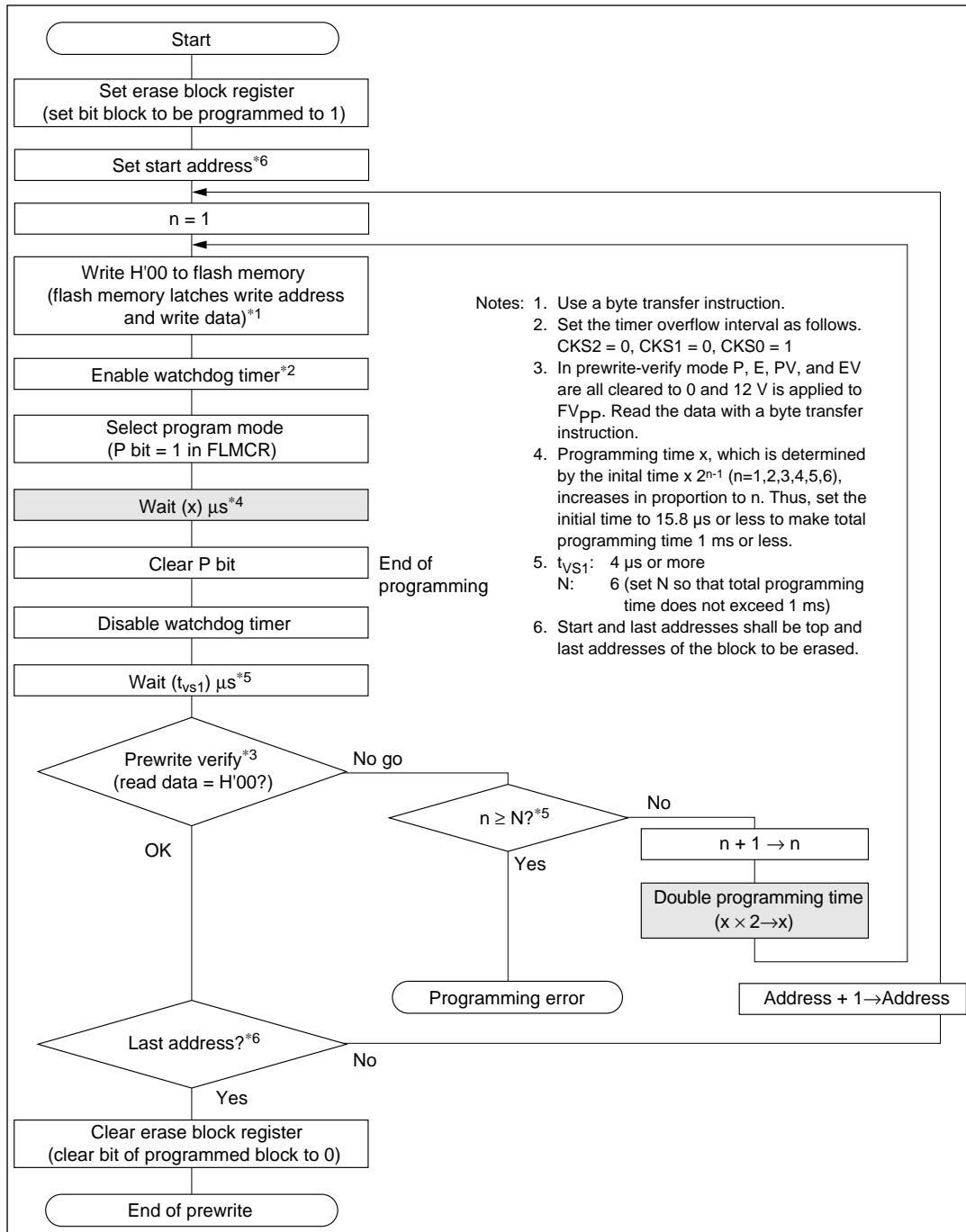


Figure 20-10 Prewrite Flowchart

**Sample Block-Erase Program:** This program uses the following registers.

- R0: Specifies block to be erased, and also stores address used in prewrite and erase-verify.
- R1H: Stores data to be read, and also used for dummy write.
- R2: Stores last address of block to be erased.
- R3: Stores address used in prewrite and erase-verify.
- R4: Sets timing loop counters for prewrite, prewrite-verify, erase, and erase-verify, and also stores register setting value.
- R5: Sets prewrite and erase timing loop counters.
- R6L: Used for prewrite-verify and erase-verify fail count.

The setting of #a, #b, #c, #d, and #e values in the program depends on the clock frequency. Set #a, #b, #c, #d, and #e values according to tables 20-9 (1) and (2), and 20-10. Erase block registers (EBR1 and EBR2) should be set according to sections 20.2.2 and 20.2.3. #BLKSTR and #BLKEND are the top and last addresses of the block to be erased. Set #BLKSTR and #BLKEND according to figure 20-2.

```

FLMCR: .EQU H'FF80
EBR1: .EQU H'FF82
EBR2: .EQU H'FF83
TCSR: .EQU H'FFA8

.ALIGN 2
MOV.B #H'**, ROH ; ; Set EBR*
MOV.B ROH, @EBR*:8 ; ; Set EBR*

; #BLKSTR is top address of block to be erased.
; #BLKEND is last address of block to be erased.
MOV.W #BLKSTR, R0 ; ; Top address of block to be erased
MOV.W #BLKEND, R2 ; ; Last address of block to be erased
ADDS #1, R2 ; ; Last address of block to be erased + 1 → R2

; Execute prewrite
MOV.W R0, R3 ; ; Top address of block to be erased
PREWRT: MOV.B #H'00, R6L ; ; Prewrite-verify fail counter
MOV.W #H'a, R5 ; ; Set prewrite loop counter
PREWRS: INC R6L ; ; Prewrite-verify fail counter + 1 → R6L
MOV.B #H'00 R1H ; ; Write H'00
MOV.B R1H, @R3 ; ; ;
MOV.W #H'A579, R4 ; ; ;
MOV.W R4, @TCSR ; ; Start watchdog timer
MOV.W R5, R4 ; ; Set prewrite loop counter
BSET #0, @FLMCR:8 ; ; Set P bit
LOOPR1: SUBS #1, R4 ; ;
MOV.W R4, R4 ; ;
BNE LOOPR1 ; ; Wait loop
BCLR #0, @FLMCR:8 ; ; Clear P bit
MOV.W #H'A500, R4 ; ;
MOV.W R4, @TCSR ; ; Stop watchdog timer

MOV.B #H'c, R4H ; ; Set prewrite-verify loop counter
LOOPR2: DEC R4H ; ;
BNE LOOPR2 ; ; Wait loop
MOV.B @R3, R1H ; ; Read data = H'00?
BEQ PWVFOK ; ; If read data = H'00 branch to PWVFOK
CMP.B #H'06, R6L ; ; Prewrite-verify executed 6 times?
BEQ ABEND1 ; ; If prewrite-verify executed 6 times, branch to ABEND1
ADD.W R5 R5 ; ; Programming time × 2
BRA PREWRS ; ; Prewrite again

ABEND1: Programming error

PWVFOK: ADDS #1, R3 ; ; Address + 1 → R3
CMP.W R2, R3 ; ; Last address?
BNE PREWRT ; ; If not last address, prewrite next address

```

```

; Execute erase
ERASES: MOV.W #H'0000, R6 ; Erase-verify fail counter
         MOV.W #H'd, R5 ; Set erase loop count
ERASE:  ADDS #1, R6 ; Erase-verify fail counter + 1 → R6
         MOV.W #H'e, R4 ;
         MOV.W R4, @TCSR ; Start watchdog timer
         MOV.W R5, R4 ; Set erase loop counter
         BSET #1, @FLMCR:8 ; Set E bit

LOOPE:  NOP ;
         NOP ;
         NOP ;
         NOP ;
         SUBS #1, R4 ;
         MOV.W R4, R4 ;
         BNE LOOPE ; Wait loop
         BCLR #1, @FLMCR:8 ; Clear E bit
         MOV.W #H'A500, R4 ;
         MOV.W R4, @TCSR ; Stop watchdog timer

; Execute erase-verify
         MOV.W R0, R3 ; Top address of block to be erased
         MOV.B #H'b, R4H ; Set erase-verify loop counter
         BSET #3, @FLMCR:8 ; Set EV bit
LOOPEV: DEC R4H ;
         BNE LOOPEV ; Wait loop
EVR2:   MOV.B #H'FF, R1H ;
         MOV.B R1H, @R3 ; Dummy write
         MOV.B #H'c, R4H ; Set erase-verify loop counter
LOOPDW: DEC R4H ;
         BNE LOOPDW ; Wait loop
         MOV.B @R3+, R1H ; Read
         CMP.B #H'FF, R1H ; Read data = H'FF?
         BNE RERASE ; If read data ≠ H'FF, branch to RERASE
         CMP.W R2, R3 ; Last address of block?
         BNE EVR2 ;
         BRA OKEND

RERASE: BCLR #3, @FLMCR:8 ; Clear EV bit
         SUBS #1, R3 ; Erase-verify address – 1 → R3

         MOV.W #H'0004, R4 ;
         CMP.W R4, R6 ; Erase-verify fail count executed 4 times?
         BPL BRER ; If R6≥4, branch to BRER (branch until R6 is 4 to 602)
         ADD.W R5, R5 ; If R6<4, Erase time × 2 (execute when R6 is 1, 2, or 3)
BRER:   MOV.W #H'025A, R4 ;
         CMP.W R4, R6 ; Erase-verify executed 602 times?
         BNE ERASE ; If erase-verify not executed 602 times, erase again
         BRA ABEND2 ; If erase-verify executed 602 times, branch to ABEND2

OKEND:  BCLR #3, @FLMCR:8 ; Clear EV bit
         MOV.B #H'00, R6L ; 
         MOV.B R6L, @EBR*:8 ; Clear EBR*

```

One block erased

ABEND2: Erase error

## Flowchart for Erasing Multiple Blocks

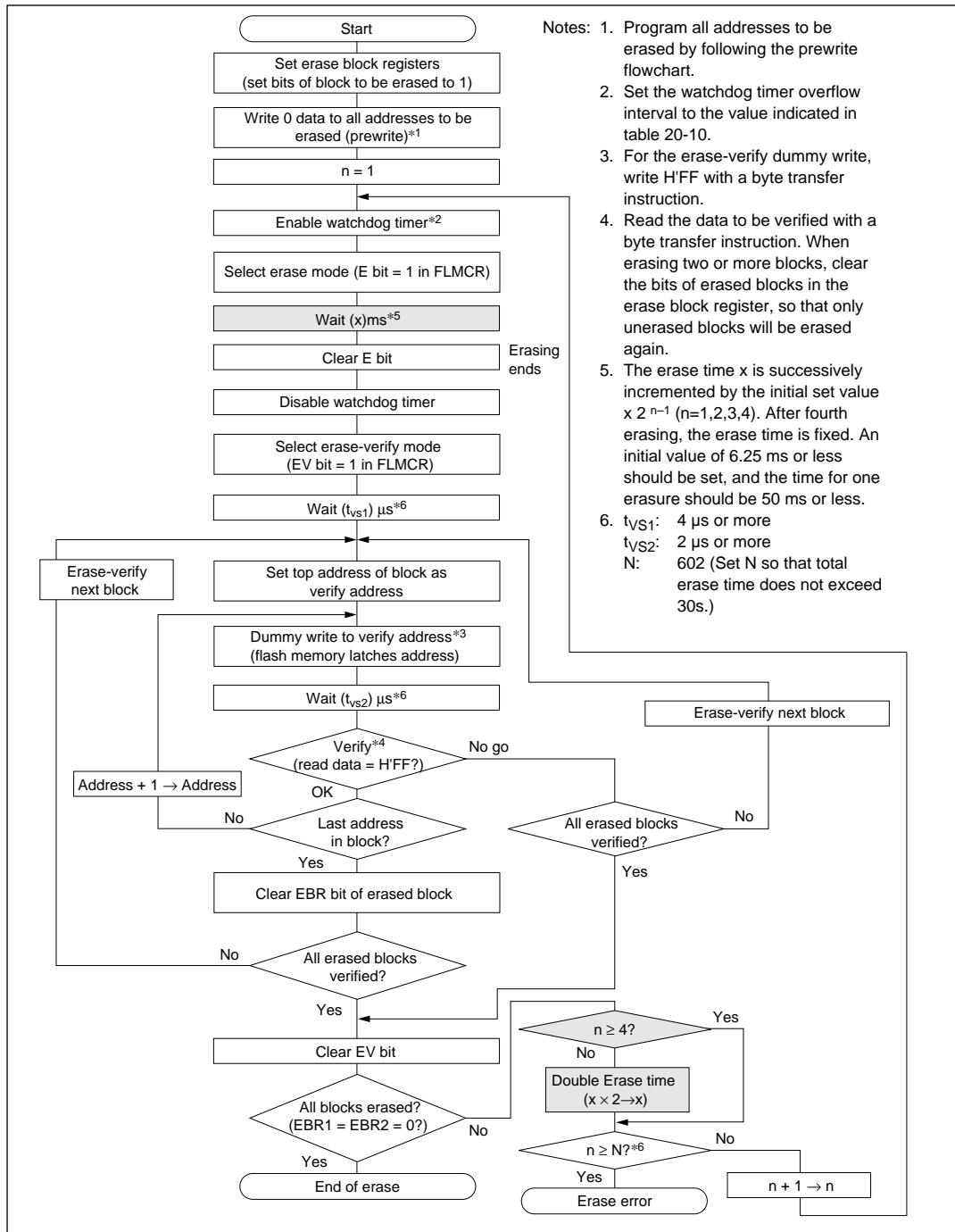


Figure 20-11 Multiple-Block Erase Flowchart

**Sample Multiple-Block Erase Program:** This program uses the following registers.

R0: Specifies blocks to be erased (set as explained below), and also stores address used in prewrite and erase-verify.

R1H: Used to test bits 8 to 15 of R0 stores register read data, and also used for dummy write.

R1L: Used to test bits 0 to 15 of R0.

R2: Specifies address where address used in prewrite and erase-verify is stored.

R3: Stores address used in prewrite and erase-verify.

R4: Stores last address of block to be erased.

R5: Sets prewrite and erase timing loop counters.

R6L: Used for prewrite-verify and erase-verify fail count.

Arbitrary blocks can be erased by setting bits in R0. Write R0 with a word transfer instruction.

A bit map of R0 and a sample setting for erasing specific blocks are shown next.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	LB7	LB6	LB5	LB4	LB3	LB2	LB1	LB0	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
	Corresponds to EBR1										Corresponds to EBR2					

Example: to erase blocks LB2, SB7, and SB0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	LB7	LB6	LB5	LB4	LB3	LB2	LB1	LB0	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
	Corresponds to EBR1										Corresponds to EBR2					
Setting	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1

R0 is set as follows:

```
MOV.W #H'0481, R0
MOV.W R0, @EBR1
```

The setting of #a, #b, #c, #d, and #e values in the program depends on the clock frequency. Set #a, #b, #c, #d, and #e values according to tables 20-9 (1), (2), and 20-10.

Notes:

1. In this sample program, the stack pointer (SP) is set at address FF80. As the stack area, on-chip RAM addresses FF7E and FF7F are used. Therefore, when executing this sample program, addresses FF7E and FF7F should not be used. In addition, the on-chip RAM should not be disabled.
2. In this sample program, the program written in a ROM area (including external space) is transferred into the RAM area and executed in the RAM to which the program is transferred. #RAMSTR in the program is the starting destination address in RAM to which the program is transferred. #RAMSTR must be set to an even number.
3. When executing this sample program in the on-chip ROM area or external space, #RAMSTR should be set to #START.

```

FLMCR: .RQU      H'FF80
EBR1:  .EQU      H'FF82
EBR2:  .EQU      H'FF83
TCSR:  .EQU      H'FFA8
STACK: .EQU      H'FF80

.ALIGN 2
START: MOV.W #STACK, SP      ; Set stack pointer
; Set the bits in R0 following the description on the previous page. This program is a sample program to erase
; all blocks.
        MOV.W #H'FFFF, R0      ; Select blocks to be erased (R0: EBR1/EBR2)
        MOV.W R0, @EBR1        ; Set EBR1/EBR2

; #RAMSTR is starting destination address to which program is transferred in RAM.
; Set #RAMSTR to even number.
        MOV.W #RAMSTR, R2      ; Starting transfer destination address (RAM)
        MOV.W #ERVADR, R3      ;
        ADD.W R3, R2           ; #RAMSTR + #ERVADR → R2
        MOV.W #START, R3       ;
        SUB.W R3, R2           ; Address of data area used in RAM

        MOV.B #H'00, R1L        ; Used to test R1L bit in R0
PRETST: CMP.B #H'10, R1L        ; R1L = H'10?
        BEQ ERASES             ; If finished checking all R0 bits, branch to ERASES
        CMP.B #H'08, R1L        ;
        BMI EBR2PW             ; Test EBR1 if R1L ≥ 8, or EBR2 if R1L < 8
        MOV.B R1L, R1H          ;
        SUBX #H'08, R1H          ; R1L - 8 → R1H
        BTST R1H, R0H           ; Test R1H bit in EBR1 (R0H)
        BNE PREWRT              ; If R1H bit in EBR1 (R0H) is 1, branch to PREWRT
        BRA PWADD1              ; If R1H bit in EBR1 (R0H) is 0, branch to PWADD1
EBR2PW: BTST R1L, R0L           ; Test R1L bit in EBR2 (R0L)
        BNE PREWRT              ; If R1L bit in EBR2 (R0H) is 1, branch to PREWRT
PWADD1: INC R1L                 ; R1L + 1 → R1L
        MOV.W @R2+, R3          ; Dummy-increment R2
        BRA PRETST              ;

; Execute prewrite

```

PREWRT:	MOV.W	@R2+, R3	; Prewrite starting address
PREW:	MOV.B	#H'00, R6L	; Prewrite-verify fail counter
	MOV.W	#H'a, R5	; Prewrite-verify loop counter
PREWRS:	INC	R6L	; Prewrite-verify fail counter + 1 → R6L
	MOV.B	#H'00 R1H	;
	MOV.B	R1H, @R3	; Write H'00
	MOV.W	#H'A579, R4	;
	MOV.W	R4, @TCSR	; Start watchdog timer
	MOV.W	R5, R4	; Set prewrite loop counter
	BSET	#0, @FLMCR:8	; Set P bit
LOOPR1:	SUBS	#1, R4	;
	MOV.W	R4, R4	;
	BNE	LOOPR1	; Wait loop
	BCLR	#0, @FLMCR:8	; Clear P bit
	MOV.W	#H'A500, R4	;
	MOV.W	R4, @TCSR	; Stop watchdog timer
	MOV.B	#H'c, R4H	; Set prewrite-verify loop counter
LOOPR2:	DEC	R4H	;
	BNE	LOOPR2	; Wait loop
	MOV.B	@R3, R1H	; Read data = H'00?
	BEQ	PWVFOK	; If read data = H'00 branch to PWVFOK
	CMP.B	#H'06, R6L	; Prewrite-verify executed 6 times?
	BEQ	ABEND1	; If prewrite-verify executed 6 times, branch to ABEND1
	ADD.W	R5, R5	; Programming time × 2
	BRA	PREWRS	; Prewrite again
ABEND1:	Programming error		
PWVFOK:	ADDS	#1, R3	; Address + 1 → R3
	MOV.W	@R2, R4	; Top address of next block
	CMP.W	R4, R3	; Last address?
	BNE	PREW	; If not last address, prewrite next address
PWADD2:	INC	R1L	; Used to test R1L+1 bit in R0
	BRA	PRETST	; Branch to PRETST
; Execute erase			
ERASES:	MOV.W	#H'0000, R6	; Erase-verify fail counter
	MOV.W	#H'd, R5	; Set erase loop count
ERASE:	ADDS	#1, R6	; Erase-verify fail counter + 1 → R6
	MOV.W	#H'e, R4	;
	MOV.W	R4, @TCSR	; Start watchdog timer
	MOV.W	R5, R4	; Set erase loop counter
	BSET	#1, @FLMCR:8	; Set E bit
LOOPE:	NOP		
	SUBS	#1, R4	;
	MOV.W	R4, R4	;
	BNE	LOOPE	; Wait loop

```

        BCLR      #1,      @FLMCR:8    ; Clear E bit
        MOV.W    #H'A500,   R4      ;
        MOV.W    R4,      @TCSR    ; Stop watchdog timer
; Execute erase-verify
EVR:    MOV.W    #RAMSTR,  R2      ; Starting transfer destination address (RAM)
        MOV.W    #ERVADR,  R3      ;
        ADD.W    R3,      R2      ; #RAMSTR + #ERVADR → R2
        MOV.W    #START,   R3      ;
        SUB.W    R3,      R2      ; Address of data area used in RAM

        MOV.B    #H'00,    R1L     ; Used to test R1L bit in R0
        MOV.B    #H'b,     R4H     ; Set erase-verify loop counter
        BSET    #3,      @FLMCR:8 ; Set EV bit
LOOPEV: DEC      R4H      ;
        BNE     LOOPEV   ; Wait loop
EBRTST: CMP.B   #H'10,    R1L     ; R1L = H'10?
        BEQ     HANTEI   ; If finished checking all R0 bits, branch to HANTEI
        CMP.B   #H'08,    R1L     ;
        BMI     EBR2EV   ; Test EBR1 if R1L ≥ 8, or EBR2 if R1L < 8
        MOV.B   R1L,     R1H     ;
        SUBX   #H'08,    R1H     ;
        BTST   R1H,     R0H     ; R1L - 8 → R1H
        BNE    ERSEVF   ; Test R1H bit in EBR1 (R0H)
        BRA    ADD01    ; If R1H bit in EBR1 (R0H) is 1, branch to ERSEVF
        BRA    ADD01    ; If R1H bit in EBR1 (R0H) is 0, branch to ADD01
EBR2EV: BTST   R1L,     R0L     ; Test R1L bit in EBR2 (R0L)
        BNE    ERSEVF   ; If R1L bit in EBR2 (R0H) is 1, branch to ERSEVF
ADD01:  INC     R1L      ;
        MOV.W   @R2+,   R3      ; Dummy-increment R2
        BRA    EBRTST  ;

ERASE1: BRA     ERASE    ; Branch to ERASE via Erase 1

ERSEVF: MOV.W   @R2+,   R3      ; Top address of block to be erase-verified
EVR2:   MOV.B   #H'FF,   R1H     ;
        MOV.B   R1H,     @R3     ; Dummy write
        MOV.B   #H'c,    R4H     ; Set erase-verify loop counter
LOOPEP: DEC      R4H      ;
        BNE     LOOPEP   ; Wait loop
        MOV.B   @R3+,   R1H     ; Read
        CMP.B   #H'FF,   R1H     ; Read data = H'FF?
        BNE    BLKAD    ; If read data ≠ H'FF branch to BLKAD
        MOV.W   @R2,    R4      ; Top address of next block
        CMP.W   R4,     R3      ; Last address of block?
        BNE    EVR2    ;

        CMP.B   #H'08,   R1L     ;
        BMI     SBCLR   ; Test EBR1 if R1L ≥ 8, or EBR2 if R1L < 8
        MOV.B   R1L,     R1H     ;
        SUBX   #H'08,   R1H     ; R1L - 8 → R1H
        BCLR   R1H,     R0H     ; Clear R1H bit in EBR1 (R0H)
        BRA    BLKAD   ;

```

SBCLR:	BCLR	R1L,	ROL	; Clear R1L bit in EBR2 (ROL)
BLKAD:	INC	R1L		; R1L + 1 → R1L
	BRA	EBRTST		;
HANTEI:	BCLR	#3,	@FLMCR:8	; Clear EV bit
	MOV.W	R0,	@EBR1	;
	BEQ	EOWARI		; If EBR1/EBR2 is all 0, erasing ended normally
	MOV.W	#H'0004,	R4	;
	CMP.W	R4,	R6	; Erase-verify fail count executed 4 times?
	BPL	BRER		; If R6≥4, branch to BRER (branch until R6 is 4 to 602)
	ADD.W	R5,	R5	; If R6<4, Erase time × 2 (execute when R6 is 1, 2, or 3)
BRER:	MOV.W	#H'025A,	R4	;
	CMP.W	R4,	R6	; Erase-verify executed 602 times?
	BNE	ERASE1		; If erase-verify not executed 602 times, erase again
	BRA	ABEND2		; If erase-verify executed 602 times, branch to ABEND2

;<-----< Block address table used in erase-verify> ----->

.ALIGN	2		
ERVADR:	.DATA.W	H'0000	; SB0
	.DATA.W	H'0080	; SB1
	.DATA.W	H'0100	; SB2
	.DATA.W	H'0180	; SB3
	.DATA.W	H'0200	; SB4
	.DATA.W	H'0400	; SB5
	.DATA.W	H'0800	; SB6
	.DATA.W	H'0C00	; SB7
	.DATA.W	H'1000	; LB0
	.DATA.W	H'2000	; LB1
	.DATA.W	H'4000	; LB2
	.DATA.W	H'6000	; LB3
	.DATA.W	H'8000	; LB4
	.DATA.W	H'A000	; LB5
	.DATA.W	H'C000	; LB6
	.DATA.W	H'EF80	; LB7
	.DATA.W	H'F780	; FLASH END

EOWARI: Erase end

ABEND2: Erase error

## Loop Counter Values in Programs and Watchdog Timer Overflow Interval Settings:

The setting of #a, #b, #c, #d, and #e values in the programs depends on the clock frequency.

Tables 20-9 (1) and (2) indicate sample loop counter settings for typical clock frequencies.

However, #e is set according to table 20-10.

As a software loop is used, calculated values including percent errors may not be the same as actual values. Therefore, the values are set so that the total programming time and total erase time do not exceed 1 ms and 30 s, respectively.

The maximum number of writes in the program, N, is set to 6.

Programming and erasing in accordance with the flowcharts is achieved by setting #a, #b, #c, and #d in the programs as shown in tables 20-9 (1) and (2). #e should be set as shown in table 20-10.

Wait state insertion is inhibited in these programs. If wait states are to be used, the setting should be made after the program ends. The setting value for the watchdog timer (WDT) overflow time is calculated based on the number of instructions between starting and stopping of the WDT, including the write time and erase time. Therefore, no other instructions should be added between starting and stopping of the WDT in this program example.

**Table 20-9 (1) #a, #b, #c, and #d Setting Values for Typical Clock Frequencies with Program Running in the On-Chip Memory (RAM)**

Variable	Time Setting	Clock Frequency			
		f = 16 MHz	f = 10 MHz	f = 8 MHz	f = 2 MHz
a <sub>(f)</sub>	Programming time (initial setting value)	15.8 $\mu$ s	H'001F	H'0013	H'000F
b <sub>(f)</sub>	tvs1	4 $\mu$ s	H'0B	H'07	H'06
c <sub>(f)</sub>	tvs2	2 $\mu$ s	H'06	H'04	H'03
d <sub>(f)</sub>	Erase time (initial setting value)	6.25 ms	H'1869	H'0F42	H'0C34

**Table 20-9 (2) #a, #b, #c, and #d Setting Values for Typical Clock Frequencies with Program Running in the External Device**

Variable	Time Setting	Clock Frequency			
		f = 16 MHz	f = 10 MHz	f = 8 MHz	f = 2 MHz
a <sub>(f)</sub>	Programming time (initial setting value)	15.8 $\mu$ s	H'000A	H'0006	H'0005
b <sub>(f)</sub>	tvs1	4 $\mu$ s	H'04	H'03	H'02
c <sub>(f)</sub>	tvs2	2 $\mu$ s	H'02	H'02	H'01
d <sub>(f)</sub>	Erase time (initial setting value)	6.25 ms	H'0823	H'0516	H'0411
					H'0104

Formula: When using a clock frequency not shown in tables 20-9 (1) and (2), follow the formula below. The calculation is based on a clock frequency of 10 MHz.

After calculating a(f) and d(f) in the decimal system, omit the first decimal figures, and convert them to the hexadecimal system, so that a(f) and d(f) are set to 15.8  $\mu$ s or less and 6.25 ms or less, respectively.

After calculating b(f) and c(f) in the decimal system, raise the first decimal figures, and convert them to the hexadecimal system, so that b(f) and c(f) are set to 4  $\mu$ s or more and 2  $\mu$ s or more, respectively.

$$a(f) \text{ to } d(f) = \frac{\text{Clock Frequency } f [\text{MHz}]}{10} \times a(f = 10) \text{ to } d(f = 10)$$

Examples for a program running in on-chip memory (RAM) at a clock frequency of 12 MHz:

$$\begin{aligned}
 a(f) &= \frac{12}{10} \times 19 = 22.8 \approx 22 = H'0016 \\
 b(f) &= \frac{12}{10} \times 7 = 8.4 \approx 9 = H'09 \\
 c(f) &= \frac{12}{10} \times 4 = 4.8 \approx 5 = H'05 \\
 d(f) &= \frac{12}{10} \times 3906 = 4687.2 \approx 4687 = H'124F
 \end{aligned}$$

**Table 20-10 Watchdog Timer Overflow Interval Settings (#e Setting Value According to Clock Frequency)**

Clock Frequency [MHz]	Variable e (f)
10 MHz $\leq$ frequency $\leq$ 16 MHz	H'A57F
2 MHz $\leq$ frequency $<$ 10 MHz	H'A57E

#### 20.4.7 Prewrite Verify Mode

Prewrite-verify mode is a verify mode used when programming all bits to equalize their threshold voltages before erasing them.

To program all bits, follow the prewrite algorithm shown in figure 20-10. The procedure is to program all flash memory data to H'00 by using H'00 write data. H'00 should also be written when using RAM for flash memory emulation (when prewriting a RAM area). (This also applies when using RAM to emulate flash memory erasing with an emulator or other support tool.) After the necessary programming time has elapsed, exit program mode (by clearing the P bit to 0) and select prewrite-verify mode (leave the P, E, PV, and EV bits all cleared to 0). In prewrite-verify mode, a prewrite-verify voltage is applied to the memory cells at the read address. If the flash memory is read in this state, the data at the read address will be read. After selecting prewrite-verify mode, wait 4  $\mu$ s or more before reading.

Note: For a sample prewriting program, see the prewrite subroutine in the sample erasing program.

#### 20.4.8 Protect Modes

Flash memory can be protected from programming and erasing by software or hardware methods. These two protection modes are described below.

**Software Protection:** Prevents transitions to program mode and erase mode even if the P or E bit is set in the flash memory control register (FLMCR). Details are as follows.

Protection	Description	Function		
		Program	Erase	Verify <sup>*1</sup>
Block protect	Individual blocks can be protected from erasing and programming by the erase block registers (EBR1 and EBR2). If H'00 is set in EBR1 and EBR2, all blocks are protected from erasing and programming.	Disabled	Disabled	Enabled
Emulation protect <sup>*2</sup>	When the RAMS or RAM0 bit, but not both, is set in the wait-state control register (WSCR), all blocks are protected from programming and erasing.	Disabled	Disabled <sup>*3</sup>	Enabled

Notes:

1. Three modes: program-verify, erase-verify, and prewrite-verify.
2. Except in RAM areas overlapped onto flash memory.
3. All blocks are erase-disabled. It is not possible to specify individual blocks.

**Hardware Protection:** Suspends or disables the programming and erasing of flash memory, and resets the flash memory control register (FLMCR) and erase block registers (EBR1 and EBR2). Details of hardware protection are as follows.

Protection	Description	Function		
		Program	Erase	Verify <sup>*1</sup>
Programming voltage (V <sub>PP</sub> ) protect	When 12 V is not applied to the FV <sub>PP</sub> pin, FLMCR, EBR1, and EBR2 are initialized, disabling programming and erasing. To obtain this protection, V <sub>PP</sub> should not exceed V <sub>CC</sub> . <sup>*3</sup>	Disabled	Disabled <sup>*2</sup>	Disabled
Reset and standby protect	When a reset occurs (including a watchdog timer reset) or standby mode is entered, FLMCR, EBR1, and EBR2 are initialized, disabling programming and erasing. Note that <u>RES</u> input does not ensure a reset unless the <u>RES</u> pin is held low for at least 20 ms at power-up (to enable the oscillator to settle), or at least ten system clock cycles (10 $\phi$ ) during operation.	Disabled	Disabled <sup>*2</sup>	Disabled
Interrupt protect	To prevent damage to the flash memory, if interrupt input occurs while flash memory is being programmed or erased, programming or erasing is aborted immediately. The settings in FLMCR, EBR1, and EBR2 are retained. This type of protection can be cleared only by a reset.	Disabled	Disabled <sup>*2</sup>	Enabled

Notes: 1. Three modes: program-verify, erase-verify, and prewrite-verify.  
 2. All blocks are erase-disabled. It is not possible to specify individual blocks.  
 3. For details, see section 20.7, Flash Memory Programming and Erasing Precautions.

#### 20.4.9 Interrupt Handling during Flash Memory Programming and Erasing

If an interrupt occurs<sup>\*1</sup> while flash memory is being programmed or erased (while the P or E bit of FLMCR is set), the following operating states can occur.

- If an interrupt is generated during programming or erasing, programming or erasing is aborted to protect the flash memory. Since memory cell values after a forced interrupt are indeterminate, the system will not operate correctly after such an interrupt.
- Program runaway may result because the vector table could not be read correctly in interrupt exception handling during programming or erasure<sup>\*2</sup>.

For NMI interrupts while flash memory is being programmed or erased, these malfunction and runaway problems can be prevented by using the RAM overlap function with the settings described below.

1. Do not store the NMI interrupt-handling routine<sup>\*3</sup> in the flash memory area (neither H'0000 to H'EF7F in mode 2 nor H'0000 to H'F77F in mode 3). Store it elsewhere (in RAM, for example).

2. Set the NMI interrupt vector in address H'F806 in RAM (corresponding to H'0006 in flash memory).
3. After the above settings, set both the RAMS and RAM0 bits to 1 in WSCR.\*<sup>4</sup>

Due to the setting of step 3, if an interrupt signal is input while 12 V is applied to the FV<sub>PP</sub> pin, the RAM overlap function is enabled and part of the RAM (H'F800 to H'F87F) is overlapped onto the small-block area of flash memory (H'0000 to H'007F). As a result, when an interrupt is input, the vector is read from RAM, not flash memory, so the interrupt is handled normally even if flash memory is being programmed or erased. This can prevent malfunction and runaway.

Notes: 1. When the interrupt mask bit (I) of the condition control register (CCR) is set to 1, all interrupts except NMI are masked. For details see (2) in section 2.2.2, Control Registers.

2. The vector table might not be read correctly for one of the following reasons:
  - If flash memory is read while it is being programmed or erased (while the P or E bit of FLMCR is set), the correct value cannot be read.
  - If no value has been written for the NMI entry in the vector table yet, NMI exception handling will not be executed correctly.
3. This routine should be programmed so as to prevent microcontroller runaway.
4. For details on WSCR settings, see section 20.2.4, Wait-State Control Register.

### Notes on Interrupt Handling in Boot Mode

In boot mode, the settings described above concerning NMI interrupts are carried out, and NMI interrupt handling (but not other interrupt handling) is enabled while the boot program is executing. Note the following points concerning the user program.

- If interrupt handling is required
  - Load the NMI vector (H'F780) into address H'F806 in RAM (the 38th byte of the transferred user program should be H'F780).
  - The interrupt handling routine used by the boot program is stored in addresses H'F780 to H'F78F in RAM. Make sure that the user program does not overwrite this area.
- If interrupt handling is not required

Since the RAMS and RAM0 bits remain set to 1 in WSCR, make sure that the user program disables the RAM overlap by clearing the RAMS and RAM0 bits both to 0.

## 20.5 Flash Memory Emulation by RAM

Erasing and programming flash memory takes time, which can make it difficult to tune parameters and other data in real time. If necessary, real-time updates of flash memory can be emulated by overlapping the small-block flash-memory area with part of the RAM (HF800 to HF97F). This RAM reassignment is performed using bits 7 and 6 of the wait-state control register (WSCR). See figure 20-11.

After a flash memory area has been overlapped by RAM, the RAM area can be accessed from two address areas: the overlapped flash memory area, and the original RAM area (HF800 to HF97F). Table 20-11 indicates how to reassign RAM.

### Wait-State Control Register (WSCR)<sup>\*2</sup>

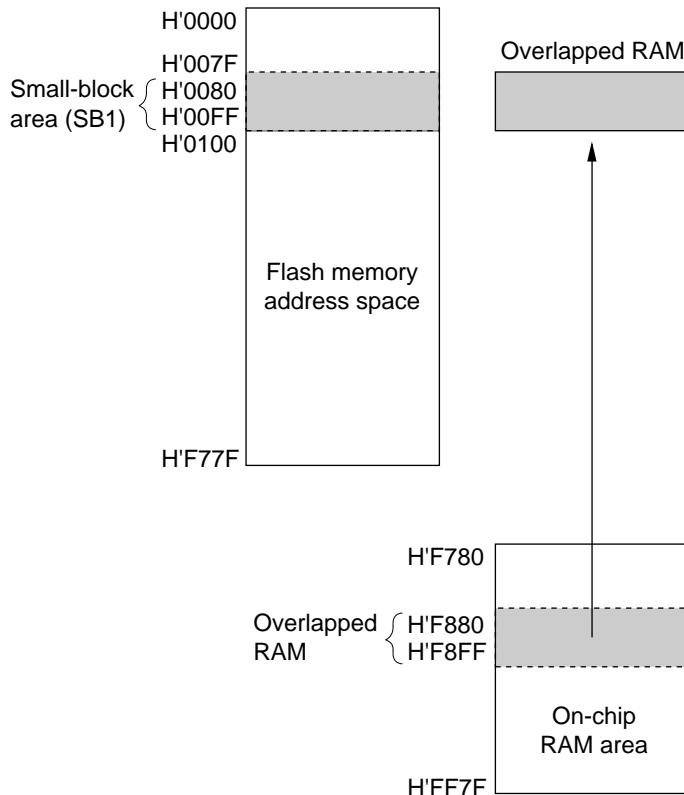
Bit	7	6	5	4	3	2	1	0
	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0
Initial value <sup>*1</sup>	0	0	0	0	1	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Notes: 1. WSCR is initialized by a reset and in hardware standby mode. It is not initialized in software standby mode.  
2. For details of WSCR settings, see section 20.2.4, Wait-State Control Register (WSCR).

Table 20-11 RAM Area Selection

Bit 7 RAMS	Bit 6 RAM0	RAM Area	ROM Area
0	0	None	—
0	1	HF880 to HF8FF	H'0080 to H'00FF
1	0	HF880 to HF97F	H'0080 to H'017F
1	1	HF800 to HF87F	H'0000 to H'007F

## Example of Emulation of Real-Time Flash-Memory Update



### Procedure

1. Overlap part of RAM (H'F880 to H'F8FF) onto the area requiring real-time update (SB1). (Set WSCR bits 7 and 6 to 01.)
2. Perform real-time updates in the overlapping RAM.
3. After finalization of the update data, clear the RAM overlap (by clearing the RAMS and RAM0 bits).
4. Read the data written in RAM addresses H'F880 to H'F8FF out externally, then program the flash memory area, using this data as part of the program data.

**Figure 20-12 Example of RAM Overlap**

## Notes on Use of RAM Emulation Function

- **Notes on Applying, Releasing, and Shutting Off the Programming Voltage (V<sub>PP</sub>)**

Care is necessary to avoid errors in programming and erasing when applying, releasing, and shutting off V<sub>PP</sub>, just as in the on-board programming modes. In particular, even if the emulation function is being used, make sure that the watchdog timer is set when the P or E bit of the flash memory control register (FLMCR) has been set, to prevent errors in programming and erasing due to program runaway while V<sub>PP</sub> is applied.

For details see section 20.7, Flash Memory Programming and Erasing Precautions (5).

## 20.6 Flash Memory PROM Mode (H8/3437F)

### 20.6.1 PROM Mode Setting

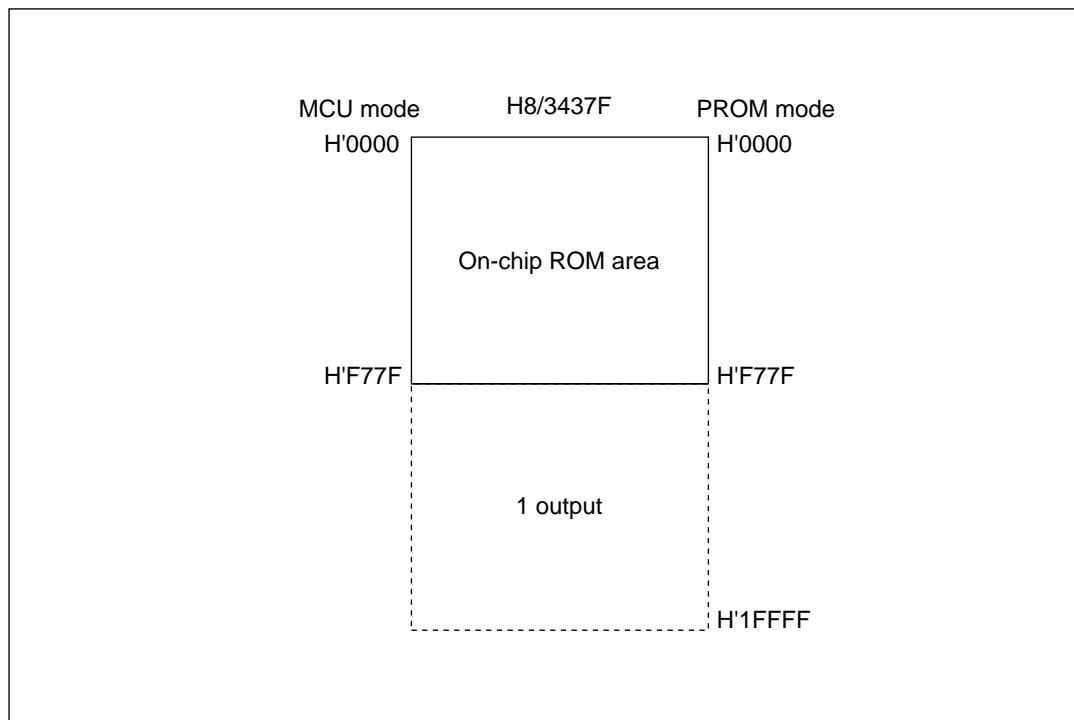
The on-chip flash memory of the H8/3437F can be programmed and erased not only in the on-board programming modes but also in PROM mode, using a general-purpose PROM programmer.

## 20.6.2 Socket Adapter and Memory Map

Programs can be written and verified by attaching a special 100-pin/32-pin socket adapter to the PROM programmer. Table 20-12 gives ordering information for the socket adapter. Figure 20-13 shows a memory map in PROM mode. Figure 20-14 shows the socket adapter pin interconnections.

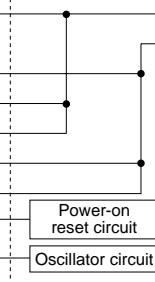
**Table 20-12** Socket Adapter

Microcontroller	Package	Socket Adapter
HD64F3437F16	100-pin QFP	HS3434ESHF1H
HD64F3437TF16	100-pin TQFP	HS3434ESNF1H



**Figure 20-13 Memory Map in PROM Mode**

H8/3437F		Socket Adapter	HN28F101 (32 Pins)	
Pin No.	Pin Name		Pin Name	Pin No.
FP-100B, TFP-100B				
8	STBY/FV <sub>PP</sub>		V <sub>PP</sub>	1
7	NMI		FA <sub>9</sub>	26
18	P9 <sub>5</sub>		FA <sub>16</sub>	2
19	P9 <sub>4</sub>		FA <sub>15</sub>	3
22	P9 <sub>3</sub>		WE	31
82	P3 <sub>0</sub>		FO <sub>0</sub>	13
83	P3 <sub>1</sub>		FO <sub>1</sub>	14
84	P3 <sub>2</sub>		FO <sub>2</sub>	15
85	P3 <sub>3</sub>		FO <sub>3</sub>	17
86	P3 <sub>4</sub>		FO <sub>4</sub>	18
87	P3 <sub>5</sub>		FO <sub>5</sub>	19
88	P3 <sub>6</sub>		FO <sub>6</sub>	20
89	P3 <sub>7</sub>		FO <sub>7</sub>	21
79	P1 <sub>0</sub>		FA <sub>0</sub>	12
78	P1 <sub>1</sub>		FA <sub>1</sub>	11
77	P1 <sub>2</sub>		FA <sub>2</sub>	10
76	P1 <sub>3</sub>		FA <sub>3</sub>	9
75	P1 <sub>4</sub>		FA <sub>4</sub>	8
74	P1 <sub>5</sub>		FA <sub>5</sub>	7
73	P1 <sub>6</sub>		FA <sub>6</sub>	6
72	P1 <sub>7</sub>		FA <sub>7</sub>	5
67	P2 <sub>0</sub>		FA <sub>8</sub>	27
66	P2 <sub>1</sub>		OE	24
65	P2 <sub>2</sub>		FA <sub>10</sub>	23
64	P2 <sub>3</sub>		FA <sub>11</sub>	25
63	P2 <sub>4</sub>		FA <sub>12</sub>	4
62	P2 <sub>5</sub>		FA <sub>13</sub>	28
61	P2 <sub>6</sub>		FA <sub>14</sub>	29
60	P2 <sub>7</sub>		CE	22
24, 25, 29, 32, 16	P9 <sub>1</sub> , P9 <sub>0</sub> , P6 <sub>3</sub> , P6 <sub>4</sub> , P9 <sub>7</sub>		V <sub>CC</sub>	32
5, 6, 23, 35	MD <sub>1</sub> , MD <sub>0</sub> , P9 <sub>2</sub> , P6 <sub>7</sub>		V <sub>SS</sub>	16
36, 37	AV <sub>ref</sub> , AV <sub>CC</sub>			
4, 9, 59	V <sub>CCB</sub> , V <sub>CC</sub>			
46	AV <sub>SS</sub>			
15, 70, 71, 92	V <sub>SS</sub>			
1	RES			
2, 3	XTAL, EXTAL			
Other pins	NC (OPEN)			



Legend

- V<sub>PP</sub>: Programming power supply
- FO<sub>7</sub> to FO<sub>0</sub>: Data input/output
- FA<sub>16</sub> to FA<sub>0</sub>: Address input
- OE: Output enable
- CE: Chip enable
- WE: Write enable

Figure 20-14 Wiring of Socket Adapter

### 20.6.3 Operation in PROM Mode

The program/erase/verify specifications in PROM mode are the same as for the standard HN28F101 flash memory. However, since the H8/3437F does not support product name recognition mode, the programmer cannot be automatically set with the device name. Table 20-13 indicates how to select the various operating modes.

**Table 20-13 Operating Mode Selection in PROM Mode**

		Pins						
Mode		F <sub>V<sub>PP</sub></sub>	V <sub>CC</sub>	CE	OE	WE	D <sub>7</sub> to D <sub>0</sub>	A <sub>16</sub> to A <sub>0</sub>
Read	Read	V <sub>CC</sub>	V <sub>CC</sub>	L	L	H	Data output	Address input
	Output disable	V <sub>CC</sub>	V <sub>CC</sub>	L	H	H	High impedance	
	Standby	V <sub>CC</sub>	V <sub>CC</sub>	H	X	X	High impedance	
Command write	Read	V <sub>PP</sub>	V <sub>CC</sub>	L	L	H	Data output	
	Output disable	V <sub>PP</sub>	V <sub>CC</sub>	L	H	H	High impedance	
	Standby	V <sub>PP</sub>	V <sub>CC</sub>	H	X	X	High impedance	
	Write	V <sub>PP</sub>	V <sub>CC</sub>	L	H	L	Data input	

Note: \* Be sure to set the F<sub>V<sub>PP</sub></sub> pin to V<sub>CC</sub> in these states. If it is set to 0 V, hardware standby mode will be entered, even when in PROM mode, resulting in incorrect operation.

#### Legend

L: Low level

H: High level

V<sub>PP</sub>: V<sub>PP</sub> level

V<sub>CC</sub>: V<sub>CC</sub> level

X: Don't care

VH: 11.5 V ≤ VH ≤ 12.5 V

**Table 20-14 PROM Mode Commands**

Command	Cycles	Mode	1st Cycle			2nd Cycle	
			Address	Data	Mode	Address	Data
Memory read	1	Write	X	H'00	Read	RA	Dout
Erase setup/erase	2	Write	X	H'20	Write	X	H'20
Erase-verify	2	Write	EA	H'A0	Read	X	EVD
Auto-erase setup/ auto-erase	2	Write	X	H'30	Write	X	H'30
Program setup/ program	2	Write	X	H'40	Write	PA	PD
Program-verify	2	Write	X	H'C0	Read	X	PVD
Reset	2	Write	X	H'FF	Write	X	H'FF

PA: Program address

EA: Erase-verify address

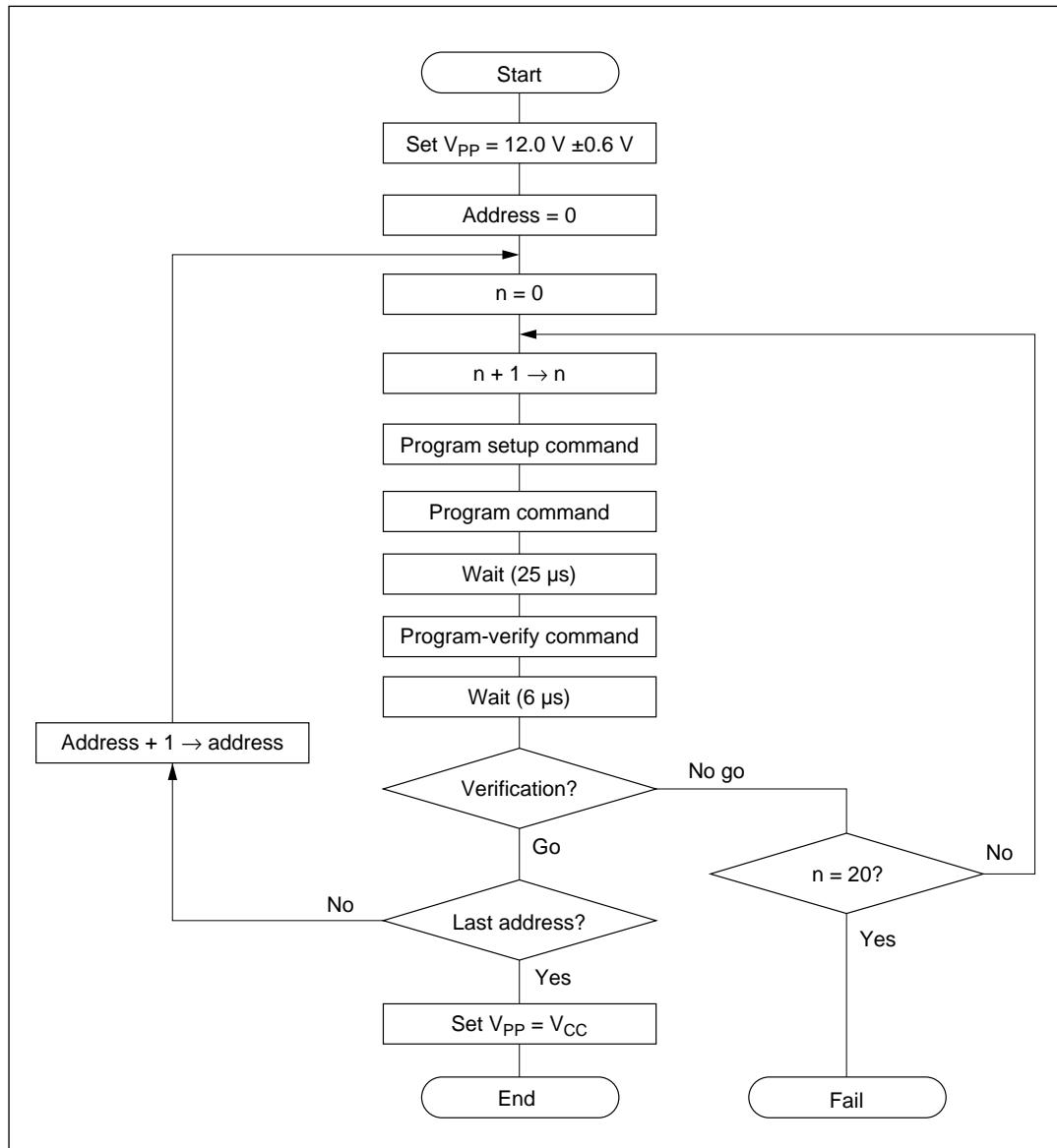
RA: Read address

PD: Program data

PVD: Program-verify output data

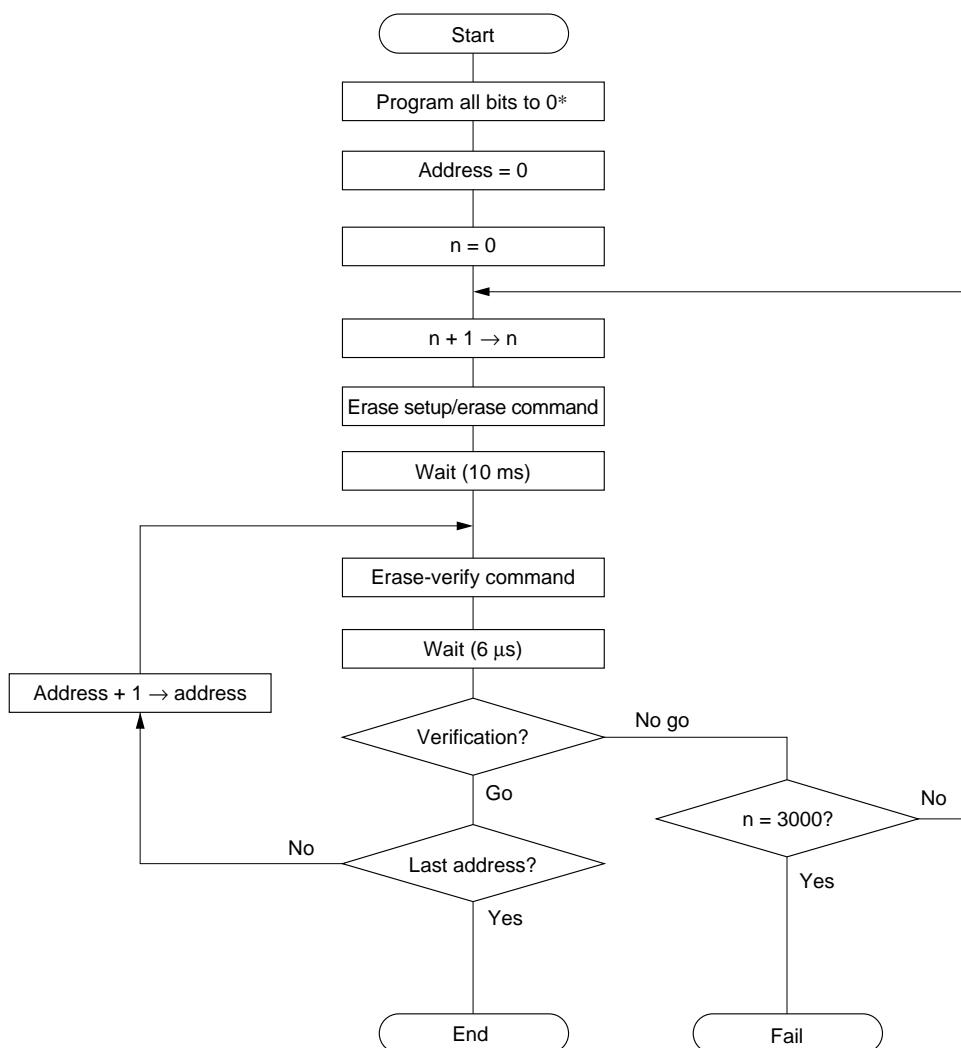
EVD: Erase-verify output data

**High-Speed, High-Reliability Programming:** Unused areas of the H8/3437F flash memory contain H'FF data (initial value). The H8/3437F flash memory uses a high-speed, high-reliability programming procedure. This procedure provides enhanced programming speed without subjecting the device to voltage stress and without sacrificing the reliability of programmed data. Figure 20-15 shows the basic high-speed, high-reliability programming flowchart. Tables 20-15 and 20-16 list the electrical characteristics during programming.



**Figure 20-15 High-Speed, High-Reliability Programming**

**High-Speed, High-Reliability Erasing:** The H8/3437F flash memory uses a high-speed, high-reliability erasing procedure. This procedure provides enhanced erasing speed without subjecting the device to voltage stress and without sacrificing data reliability. Figure 20-16 shows the basic high-speed, high-reliability erasing flowchart. Tables 20-15 and 20-16 list the electrical characteristics during erasing.



Note: \* Follow the high-speed, high-reliability programming flowchart in programming all bits. If some bits are already programmed to 0, program only the bits that have not yet been programmed.

**Figure 20-16 High-Speed, High-Reliability Erasing**

**Table 20-15 DC Characteristics in PROM Mode**(Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{PP} = 12.0 \text{ V} \pm 0.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input high voltage	FO <sub>7</sub> to FO <sub>0</sub> , FA <sub>16</sub> to FA <sub>0</sub> , OE, CE, WE	V <sub>IH</sub>	2.2	—	V <sub>CC</sub> + 0.3	V	
Input low voltage	FO <sub>7</sub> to FO <sub>0</sub> , FA <sub>16</sub> to FA <sub>0</sub> , OE, CE, WE	V <sub>IL</sub>	-0.3	—	0.8	V	
Output high voltage	FO <sub>7</sub> to FO <sub>0</sub>	V <sub>OH</sub>	2.4	—	—	V	I <sub>OH</sub> = -200 μA
Output low voltage	FO <sub>7</sub> to FO <sub>0</sub>	V <sub>OL</sub>	—	—	0.45	V	I <sub>OL</sub> = 1.6 mA
Input leakage current	FO <sub>7</sub> to FO <sub>0</sub> , FA <sub>16</sub> to FA <sub>0</sub> , OE, CE, WE	I <sub>LI</sub>	—	—	2	μA	V <sub>in</sub> = 0 to V <sub>CC</sub>
V <sub>CC</sub> current	Read	I <sub>CC</sub>	—	40	80	mA	
	Program	I <sub>CC</sub>	—	40	80	mA	
	Erase	I <sub>CC</sub>	—	40	80	mA	
FV <sub>PP</sub> current	Read	I <sub>PP</sub>	—	—	10	μA	V <sub>PP</sub> = 2.7 V to 5.5 V
			—	10	20	mA	V <sub>PP</sub> = 12.6 V
	Program	I <sub>PP</sub>	—	20	40	mA	V <sub>PP</sub> = 12.6 V
	Erase	I <sub>PP</sub>	—	20	40	mA	V <sub>PP</sub> = 12.6 V

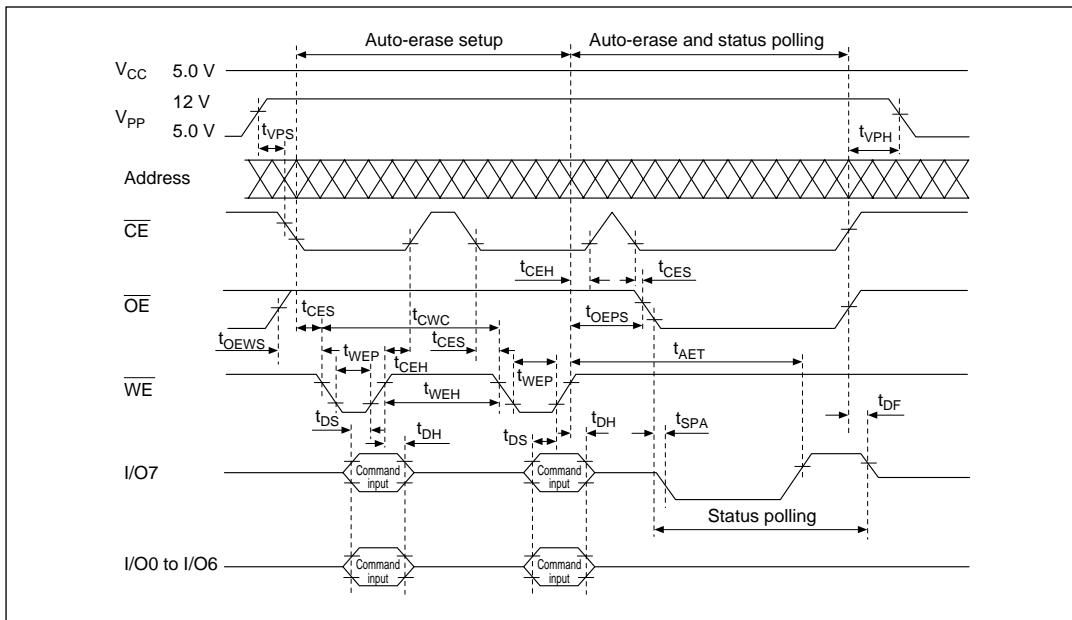
**Table 20-16 AC Characteristics in PROM Mode**(Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{PP} = 12.0 \text{ V} \pm 0.6 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Command write cycle	$t_{CWC}$	120	—	—	ns	Figure 20-17
Address setup time	$t_{AS}$	0	—	—	ns	Figure 20-18*
Address hold time	$t_{AH}$	60	—	—	ns	Figure 20-19
Data setup time	$t_{DS}$	50	—	—	ns	
Data hold time	$t_{DH}$	10	—	—	ns	
$\overline{CE}$ setup time	$t_{CES}$	0	—	—	ns	
$\overline{CE}$ hold time	$t_{CEH}$	0	—	—	ns	
$V_{PP}$ setup time	$t_{VPS}$	100	—	—	ns	
$V_{PP}$ hold time	$t_{VPH}$	100	—	—	ns	
$\overline{WE}$ programming pulse width	$t_{WEP}$	70	—	—	ns	
$\overline{WE}$ programming pulse high time	$t_{WEH}$	40	—	—	ns	
$\overline{OE}$ setup time before command write	$t_{OEWS}$	0	—	—	ns	
$\overline{OE}$ setup time before verify	$t_{OERS}$	6	—	—	$\mu\text{s}$	
Verify access time	$t_{VA}$	—	—	500	ns	
$\overline{OE}$ setup time before status polling	$t_{OEPS}$	120	—	—	ns	
Status polling access time	$t_{SPA}$	—	—	120	ns	
Program wait time	$t_{PPW}$	25	—	—	ns	
Erase wait time	$t_{ET}$	9	—	11	ms	
Output disable time	$t_{DF}$	0	—	40	ns	
Total auto-erase time	$t_{AET}$	0.5	—	30	s	

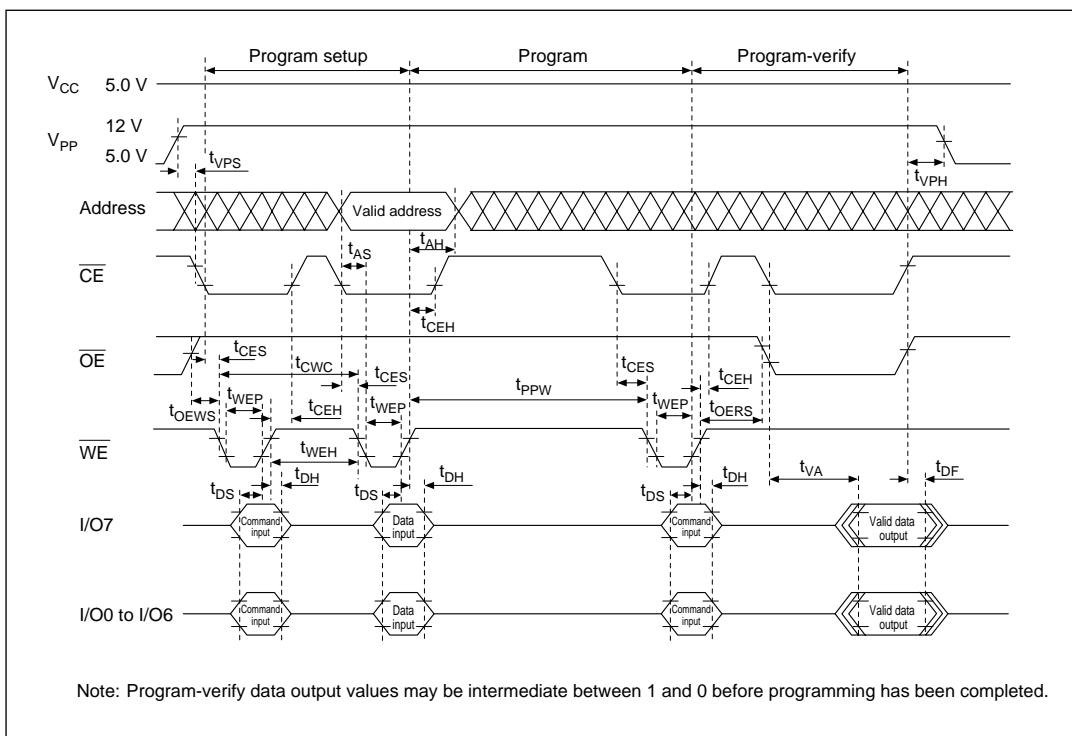
Note:  $\overline{CE}$ ,  $\overline{OE}$ , and  $\overline{WE}$  should be high during transitions of  $V_{PP}$  from 5 V to 12 V and from 12 V to 5 V.

\* Input pulse level: 0.45 V to 2.4 V  
 Input rise time and fall time  $\leq 10 \text{ ns}$

Timing reference levels: 0.8 V and 2.0 V for input; 0.8 V and 2.0 V for output



**Figure 20-17 Auto-Erase Timing**



**Figure 20-18** High-Speed, High-Reliability Programming Timing

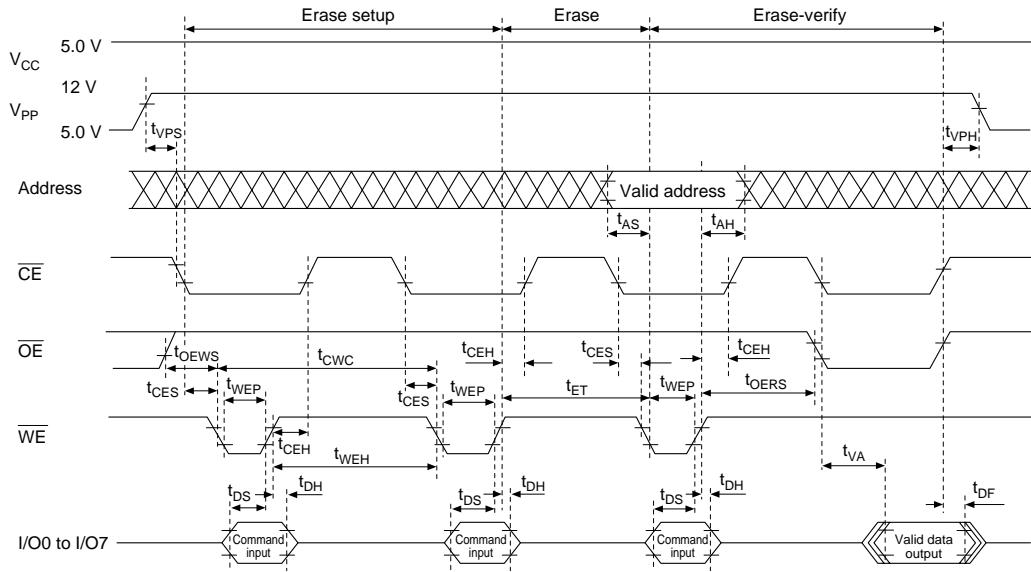


Figure 20-19 Erase Timing

## 20.7 Flash Memory Programming and Erasing Precautions

Read these precautions before using PROM mode, on-board programming mode, or flash memory emulation by RAM.

### (1) Program with the specified voltages and timing.

**The rated programming voltage ( $V_{PP}$ ) of the flash memory is 12.0 V.**

If the PROM programmer is set to Hitachi HN28F101 specifications,  $V_{PP}$  will be 12.0 V. Applying voltages in excess of the rating can permanently damage the device. Take particular care to ensure that the PROM programmer peak overshoot does not exceed the rated limit of 13 V.

### (2) Before programming, check that the chip is correctly mounted in the PROM programmer.

Overcurrent damage to the device can result if the index marks on the PROM programmer socket, 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) Set H'FF as the PROM programmer buffer data for addresses H'F780 to H'1FFF.

The H8/3437F PROM size is 60 kbytes. Addresses H'F780 to H'1FFF always read H'FF, so if H'FF is not specified as programmer data, a verify error will occur.

### (5) Notes on applying, releasing, and shutting off the programming voltage ( $V_{PP}$ )

Note: In this section, the application, release, and shutting-off of  $V_{PP}$  are defined as follows.

Application: A rise in voltage from  $V_{CC}$  to  $12\text{ V} \pm 0.6\text{ V}$ .

Release: A drop in voltage from  $12\text{ V} \pm 0.6\text{ V}$  to  $V_{CC}$ .

Shut-off: No applied voltage (floating).

- Apply the programming voltage ( $V_{PP}$ ) after the rise of  $V_{CC}$ , and release  $V_{PP}$  before shutting off  $V_{CC}$ .

To prevent unintended programming or erasing of flash memory, in these power-on and power-off timings, the application, release, and shutting-off of  $V_{PP}$  must take place when the microcontroller is in a stable operating condition as defined below.

#### Stable operating condition

- The  $V_{CC}$  voltage must be stabilized within the rated voltage range ( $V_{CC} = 2.7\text{ V}$  to  $5.5\text{ V}$ )

If  $V_{PP}$  is applied, released, or shut off while the microcontroller's  $V_{CC}$  voltage is not within the rated voltage range ( $V_{CC} = 2.7$  to  $5.5\text{ V}$ ), since microcontroller operation is unstable, the flash memory may be programmed or erased by mistake. This can occur even if  $V_{CC} = 0\text{ V}$ .

To prevent changes in the  $V_{CC}$  power supply when  $V_{PP}$  is applied, be sure that the power supply is adequately decoupled with inserting bypass capacitors.

- Clock oscillation must be stabilized (the oscillation settling time must have elapsed), and oscillation must not be stopped

When turning on  $V_{CC}$  power, hold the  $\overline{RES}$  pin low during the oscillation settling time ( $t_{OSC1} = 20$  ms), and do not apply  $V_{PP}$  until after this time.

- The microcontroller must be in the reset state, or in a state in which a reset has ended normally (reset has been released) and flash memory is not being accessed

Apply or release  $V_{PP}$  either in the reset state, or when the CPU is not accessing flash memory (when a program in on-chip RAM or external memory is executing). Flash memory cannot be read normally at the instant when  $V_{PP}$  is applied or released. Do not read flash memory while  $V_{PP}$  is being applied or released.

For a reset during operation, apply or release  $V_{PP}$  only after the  $\overline{RES}$  pin has been held low for at least ten system clock cycles (10 $\phi$ ).

- The P and E bits must be cleared in the flash memory control register (FLMCR)

When applying or releasing  $V_{PP}$ , make sure that the P or E bit is not set by mistake.

- No program runaway

When  $V_{PP}$  is applied, program execution must be supervised, e.g. by the watchdog timer.

These power-on and power-off timing requirements should also be satisfied in the event of a power failure and in recovery from a power failure. If these requirements are not satisfied, overprogramming or overerasing may occur due to program runaway etc., which could cause memory cells to malfunction.

- The  $V_{PP}$  flag is set and cleared by a threshold decision on the voltage applied to the  $FV_{PP}$  pin. The threshold level is between approximately  $V_{CC} + 2$  V to 11.4 V.

When this flag is set, it becomes possible to write to the flash memory control register (FLMCR) and the erase block registers (EBR1 and EBR2), even though the  $V_{PP}$  voltage may not yet have reached the programming voltage range of  $12.0 \pm 0.6$  V.

Do not actually program or erase the flash memory until  $V_{PP}$  has reached the programming voltage range.

The programming voltage range for programming and erasing flash memory is  $12.0 \pm 0.6$  V (11.4 V to 12.6 V). Programming and erasing cannot be performed correctly outside this range. When not programming or erasing the flash memory, ensure that the  $V_{PP}$  voltage does not

exceed the  $V_{CC}$  voltage. This will prevent unintended programming and erasing.

- In this chip, the same pin is used for  $\overline{STBY}$  and  $FV_{PP}$ . When this pin is driven low, a transition is made to hardware standby mode. This happens not only in the normal operating modes (modes 1, 2, and 3), but also when programming the flash memory with a PROM programmer. When programming with a PROM programmer, therefore, use a programmer which sets this pin to the  $V_{CC}$  level when not programming ( $FV_{PP}=12$  V).

Note: Here,  $V_{PP}$  application, release, and cutoff are defined as follows:

Application: Raising the voltage from  $V_{CC}$  to  $12 \pm 0.6$  V.

Release: Dropping the voltage from  $12 \pm 0.6$  V to  $V_{CC}$ .

Cutoff: Halting voltage application (setting the floating state).

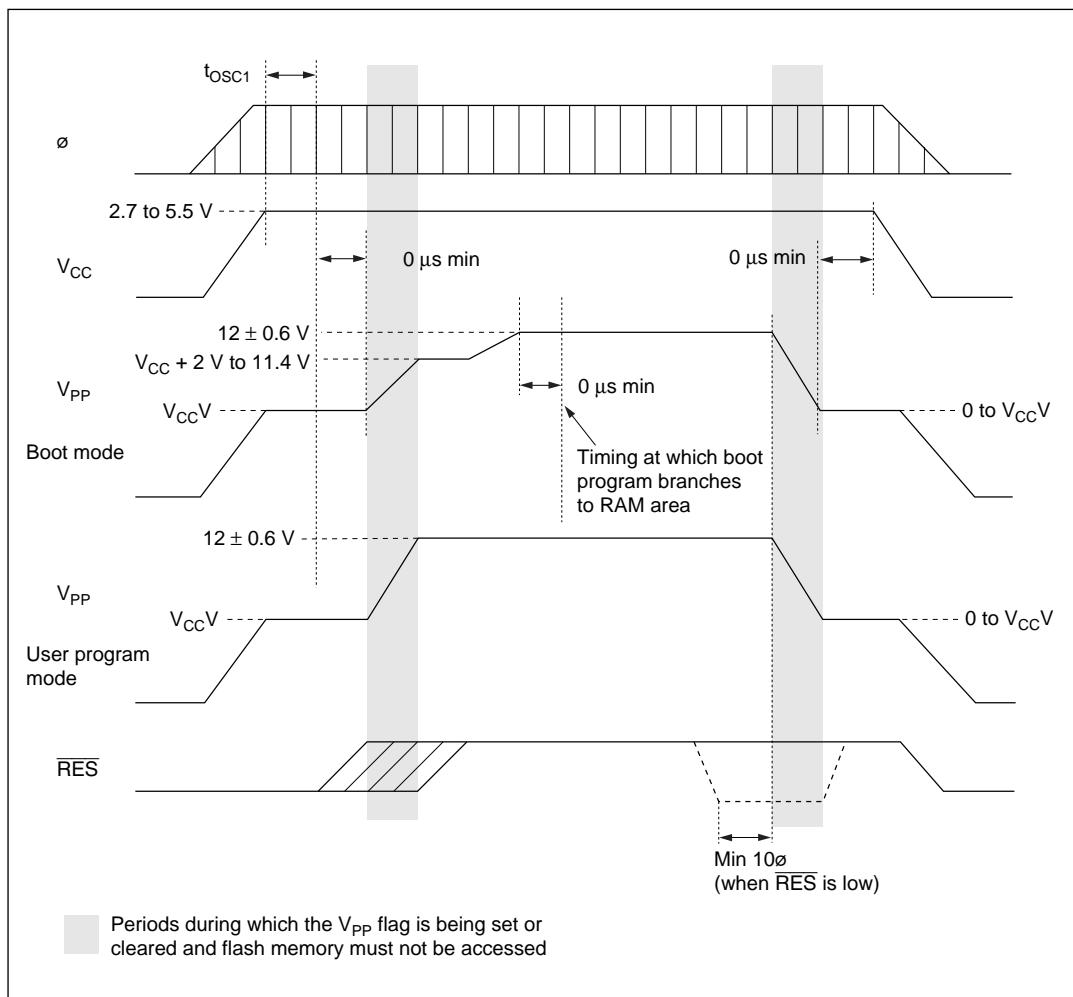


Figure 20-20  $V_{PP}$  Power-On and Power-Off Timing

**(6) Do not apply 12 V to the FV<sub>PP</sub> pin during normal operation.**

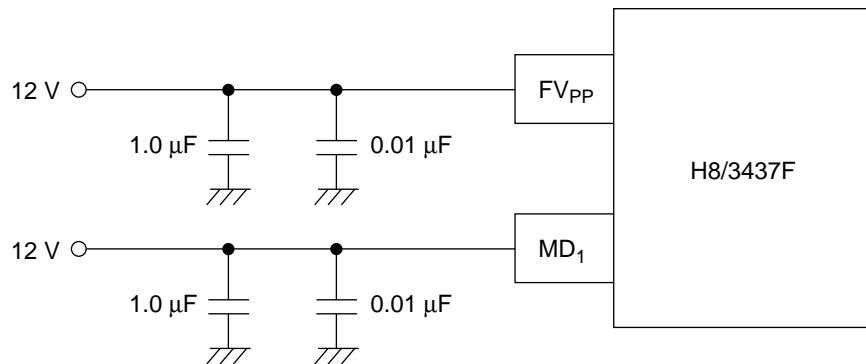
To prevent accidental programming or erasing due to microcontroller program runaway etc., apply 12 V to the V<sub>PP</sub> pin only when the flash memory is programmed or erased, or when flash memory is emulated by RAM. Overprogramming or overerasing due to program runaway can cause memory cells to malfunction. Avoid system configurations in which 12 V is always applied to the FV<sub>PP</sub> pin.

While 12 V is applied, the watchdog timer should be running and enabled to halt runaway program execution, so that program runaway will not lead to overprogramming or overerasing.

**(7) Design a current margin into the programming voltage (V<sub>PP</sub>) power supply.** Ensure that V<sub>PP</sub> will not depart from  $12.0 \pm 0.6$  V (11.4 V to 12.6 V) during programming or erasing. Programming and erasing may become impossible outside this range.

**(8) Ensure that peak overshoot does not exceed the rated value at the FV<sub>PP</sub> and MD<sub>1</sub> pins.** Connect decoupling capacitors as close to the FV<sub>PP</sub> and MD<sub>1</sub> pins as possible.

Also connect decoupling capacitors to the MD<sub>1</sub> pin in the same way when boot mode is used.



**Figure 20-21 V<sub>PP</sub> Power Supply Circuit Design (Example)**

**(9) Use the recommended algorithms for programming and erasing flash memory.** These algorithms are designed to program and erase without subjecting the device to voltage stress and without sacrificing the reliability of programmed data.

Before setting the program (P) or erase (E) bit in the flash memory control register (FLMCR), set the watchdog timer to ensure that the P or E bit does not remain set for more than the specified time.

(10) For details on interrupt handling while flash memory is being programmed or erased, see the notes on NMI interrupt handling in section 20.4.9, Interrupt Handling during Flash Memory Programming and Erasing.

## (11) Cautions on Accessing Flash Memory Control Registers

(a) Flash memory control register access state in each operating mode

The H8/3437F has flash memory control registers located at addresses H'FF80 (FLMCR), H'FF82 (EBR1), and H'FF83 (EBR2). These registers can only be accessed when 12 V is applied to the flash memory program power supply pin, FV<sub>PP</sub>.

Table 20-17 shows the area accessed for the above addresses in each mode, when 12 V is applied and is not applied to FV<sub>PP</sub>.

**Table 20-17 Area Accessed in Each Mode with 12V Applied and Not Applied to FV<sub>PP</sub>**

	Mode 1	Mode 2	Mode 3
12 V applied to FV <sub>PP</sub>	Reserved area (always H'FF)	Flash memory control register (initial value H'80)	Flash memory control register (initial value H'80)
12 V not applied to FV <sub>PP</sub>	External address space	External address space	Reserved area (always H'FF)

(b) When a flash memory control register is accessed in mode 2 (expanded mode with on-chip ROM enabled)

When a flash memory control register is accessed in mode 2, it can be read or written to if 12 V is being applied to FV<sub>PP</sub>, but if not, external address space will be accessed. It is therefore essential to confirm that 12 V is being applied to the FV<sub>PP</sub> pin before accessing these registers.

(c) To check for 12 V application/non-application in mode 3 (single-chip mode)

When address H'FF80 is accessed in mode 3, if 12 V is being applied to FV<sub>PP</sub>, FLMCR is read/written to, and its initial value after reset is H'80. When 12 V is not being applied to FV<sub>PP</sub>, FLMCR is a reserved area that cannot be modified and always reads H'FF. Since bit 7 (corresponding to the V<sub>PP</sub> bit) is set to 1 at this time regardless of whether 12 V is applied to FV<sub>PP</sub>, application or release of 12 V to FV<sub>PP</sub> cannot be determined simply from the 0 or 1 status of this bit. A byte data comparison is necessary to check whether 12V is being applied. The relevant coding is shown below.

```

LABEL1:    MOV.B    @H'FF80,    R1L
            CMP.B    #H'FF,    R1L
            BEQ     LABEL1

```

Sample program for detection of 12 V application to FV<sub>PP</sub> (mode 3)

**Table 20-18 DC Characteristics of Flash Memory**

Conditions:  $V_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{REF} = 2.7 \text{ V to } AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $V_{PP} = 12.0 \pm 0.6 \text{ V}$ ,  $T_a = -20^\circ\text{C} \text{ to } +75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C} \text{ to } +85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
High-voltage (12 V) threshold level*	$FV_{PP}$ , $MD_1$	$V_H$	$V_{CC} + 2$	—	11.4	V
$FV_{PP}$ current	During read	$I_{PP}$	—	—	10	$\mu\text{A}$
			—	10	20	$\text{mA}$
	During programming		—	20	40	$\text{mA}$
	During erasure		—	20	40	$\text{mA}$

Note: \* The listed voltages describe the threshold level at which high-voltage application is recognized. In boot mode and while flash memory is being programmed or erased, the applied voltage should be  $12.0 \text{ V} \pm 0.6 \text{ V}$ .

**Table 20-19 AC Characteristics of Flash Memory**

Conditions:  $V_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{REF} = 2.7 \text{ V to } AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $V_{PP} = 12.0 \pm 0.6 \text{ V}$ ,  $T_a = -20^\circ\text{C to } +75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to } +85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Programming time <sup>*1,*2</sup>	$t_P$	—	50	1000	$\mu\text{s}$	
Erase time <sup>*1,*3</sup>	$t_E$	—	1	30	s	
Number of writing/erasing count	$N_{WEC}$	—	—	100	Times	
Verify setup time 1 <sup>*1</sup>	$t_{VS1}$	4	—	—	$\mu\text{s}$	
Verify setup time 2 <sup>*1</sup>	$t_{VS2}$	2	—	—	$\mu\text{s}$	
Flash memory read setup time <sup>*4</sup>	$t_{FRS}$	50	—	—	$\mu\text{s}$	$V_{CC} \geq 4.5 \text{ V}$
		100	—	—		$V_{CC} < 4.5 \text{ V}$

Notes:

1. Set the times following the programming/erasing algorithm shown in section 20.
2. The programming time is the time during which a byte is programmed or the P bit in the flash memory control register (FLMCR) is set. It does not include the program-verify time.
3. The erase time is the time during which all 60-kbyte blocks are erased or the E bit in the flash memory control register (FLMCR) is set. It does not include the prewrite time before erasure or erase-verify time.
4. After power-on when using an external colck source, after return from standby mode, or after switching the programming voltage ( $V_{PP}$ ) from 12 V to  $V_{CC}$ , make sure that this read setup time has elapsed before reading flash memory.

When  $V_{PP}$  is released, the flash memory read setup time is defined as the period from when the FV<sub>PP</sub> pin has reached  $V_{CC} + 2 \text{ V}$  until flash memory can be read.

# Section 21 Power-Down State

## 21.1 Overview

The H8/3437 Series has a power-down state that greatly reduces power consumption by stopping some or all of the chip functions. The power-down state includes three modes:

- (1) Sleep mode
- (2) Software standby mode
- (3) Hardware standby mode

Table 21-1 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 21-1 Power-Down State**

Mode	Entering Procedure	Clock	CPU	CPU Reg's.	Sup. Mod.	RAM	I/O Ports	Exiting Methods
Sleep mode	Execute SLEEP instruction	Run	Halt	Held	Run	Held	Held	<ul style="list-style-type: none"><li>• <u>Interrupt</u></li><li>• <u>RES</u></li><li>• <u>STBY</u></li></ul>
Software standby mode	Set SSBY bit in SYSCR to 1, then execute SLEEP instruction	Halt	Halt	Held	Halt and initialized	Held	Held	<ul style="list-style-type: none"><li>• <u>NMI</u></li><li>• <u>IRQ<sub>0</sub>–IRQ<sub>2</sub></u></li><li>• <u>IRQ<sub>6</sub></u> (incl. KEYIN<sub>0</sub>–KEYIN<sub>15</sub>)</li><li>• <u>RES</u></li><li>• <u>STBY</u></li></ul>
Hardware standby mode	Set <u>STBY</u> pin to low level	Halt	Halt	Not held	Halt and initialized	Held	High impedance state	<ul style="list-style-type: none"><li>• <u>STBY</u> and <u>RES</u></li></ul>

Notes: 1. SYSCR: System control register

2. SSBY: Software standby bit

### 21.1.1 System Control Register (SYSCR)

Four of the eight bits in the system control register (SYSCR) control the power-down state. These are bit 7 (SSBY) and bits 6 to 4 (STS2 to STS0). See table 21-2.

**Table 21-2 System Control Register**

Name	Abbreviation	R/W	Initial Value	Address				
System control register	SYSCR	R/W	H'09	H'FFC4				
Bit	7 SSBY	6 STS2	5 STS1	4 STS0	3 XRST	2 NMIEG	1 HIE	0 RAME
Initial value	0	0	0	0	1	0	0	1
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

**Bit 7—Software Standby (SSBY):** This bit enables or disables the transition to software standby mode.

On recovery from the software standby mode by an external interrupt, SSBY remains set to 1. To clear this bit, software must write a 0.

Bit 7	SSBY	Description
0		The SLEEP instruction causes a transition to sleep mode. (Initial value)
1		The SLEEP instruction causes a transition to software standby mode.

**Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0):** These bits select the clock settling time when the chip recovers from software standby mode by an external interrupt. During the selected time, the clock oscillator runs but the CPU and on-chip supporting modules remain in standby. Set bits STS2 to STS0 according to the clock frequency to obtain a settling time of at least 8 ms. See table 21-3.

### ZTAT and Mask ROM Versions

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8,192 states	(Initial value)
0	0	1	Settling time = 16,384 states	
0	1	0	Settling time = 32,768 states	
0	1	1	Settling time = 65,536 states	
1	0	—	Settling time = 131,072 states	
1	1	—	Unused	

### F-ZTAT Version

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Settling time = 8,192 states	(Initial value)
0	0	1	Settling time = 16,384 states	
0	1	0	Settling time = 32,768 states	
0	1	1	Settling time = 65,536 states	
1	0	0	Settling time = 131,072 states	
1	0	1	Settling time = 1,024 states	
1	1	—	Unused	

Note: When 1,024 states (STS2 to STS0 = 101) is selected, the following points should be noted.

If a period exceeding  $\varnothing p/1,024$  (e.g.  $\varnothing p/2,048$ ) is specified when selecting the 8-bit timer, PWM timer, or watchdog timer clock, the counter in the timer will not count up normally when 1,024 states is specified for the settling time. To avoid this problem, set the STS value just before the transition to software standby mode (before executing the SLEEP instruction), and re-set the value of STS2 to STS0 to a value from 000 to 100 directly after software standby mode is cleared by an interrupt.

## 21.2 Sleep Mode

### 21.2.1 Transition to Sleep Mode

When the SSBY bit in the system control register is cleared to 0, execution of the SLEEP instruction causes a transition from the program execution state to sleep mode. After executing the SLEEP instruction, the CPU halts, but the contents of its internal registers remain unchanged. The on-chip supporting modules continue to operate normally.

### 21.2.2 Exit from Sleep Mode

The chip exits sleep mode when it receives an internal or external interrupt request, or a low input at the RES or STBY pin.

**1. Exit by Interrupt:** An interrupt releases sleep mode and starts the CPU's interrupt-handling sequence.

If an interrupt from an on-chip supporting module is disabled by the corresponding enable/disable bit in the module's control register, the interrupt cannot be requested, so it cannot wake the chip up. Similarly, the CPU cannot be awakened by an interrupt other than NMI if the I (interrupt mask) bit is set when the SLEEP instruction is executed.

**2. Exit by RES Pin:** When the RES pin goes low, the chip exits from sleep mode to the reset state.

**3. Exit by STBY Pin:** When the STBY pin goes low, the chip exits from sleep mode to hardware standby mode.

## 21.3 Software Standby Mode

### 21.3.1 Transition to Software Standby Mode

To enter software standby mode, set the standby bit (SSBY) in the system control register (SYSCR) to 1, then execute the SLEEP instruction.

In software standby mode, the system clock stops and chip functions halt, including both CPU functions and the functions of the on-chip supporting modules. Power consumption is reduced to an extremely low level. The on-chip supporting modules and their registers are reset to their initial states, but as long as a minimum necessary voltage supply is maintained, the contents of the CPU registers and on-chip RAM remain unchanged.

### 21.3.2 Exit from Software Standby Mode

The chip can be brought out of software standby mode by an  $\overline{\text{RES}}$  input,  $\overline{\text{STBY}}$  input, or external interrupt input at the  $\overline{\text{NMI}}$  pin,  $\overline{\text{IRQ}_0}$  to  $\overline{\text{IRQ}_2}$  pins, or  $\overline{\text{IRQ}_6}$  pin (including  $\overline{\text{KEYIN}}_0$  to  $\overline{\text{KEYIN}}_{15}$ ).

**1. Exit by Interrupt:** When an NMI, IRQ<sub>0</sub>, IRQ<sub>1</sub>, IRQ<sub>2</sub>, or IRQ<sub>6</sub> interrupt request signal is input, the clock oscillator begins operating. After the waiting time set in bits STS2 to STS0 of SYSCR, a stable clock is supplied to the entire chip, software standby mode is released, and interrupt exception-handling begins. IRQ<sub>3</sub>, IRQ<sub>4</sub>, IRQ<sub>5</sub>, and IRQ<sub>7</sub> interrupts should be disabled before the transition to software standby (clear IRQ3E, IRQ4E, IRQ5E, and IRQ7E to 0).

**2. Exit by  $\overline{\text{RES}}$  Pin:** When the  $\overline{\text{RES}}$  input goes low, the clock oscillator begins operating. When  $\overline{\text{RES}}$  is brought to the high level (after allowing time for the clock oscillator to settle), the CPU starts reset exception handling. Be sure to hold  $\overline{\text{RES}}$  low long enough for clock oscillation to stabilize.

**3. Exit by  $\overline{\text{STBY}}$  Pin:** When the  $\overline{\text{STBY}}$  input goes low, the chip exits from software standby mode to hardware standby mode.

### 21.3.3 Clock Settling Time for Exit from Software Standby Mode

Set bits STS2 to STS0 in SYSCR as follows:

- Crystal oscillator

Set STS2 to STS0 for a settling time of at least 8 ms. Table 21-3 lists the settling times selected by these bits at several clock frequencies.

- External clock

The STS bits can be set to any value. When 1,024 states (STS2 to STS0 = 101) is selected, the following points should be noted.

If a period exceeding  $\phi p/1,024$  (e.g.  $\phi p/2,048$ ) is specified when selecting the 8-bit timer, PWM timer, or watchdog timer clock, the counter in the timer will not count up normally when 1,024 states is specified for the settling time. To avoid this problem, set the STS value just before the transition to software standby mode (before executing the SLEEP instruction), and re-set the value of STS2 to STS0 to a value from 000 to 100 directly after software standby mode is cleared by an interrupt.

**Table 21-3 Times Set by Standby Timer Select Bits (Unit: ms)**

STS2	STS1	STS0	Settling Time (States)	System Clock Frequency (MHz)									
				16	12	10	8	6	4	2	1	0.5	
0	0	0	8,192	0.51	0.65	0.8	1.0	1.3	2.0	4.1	<b>8.2</b>	<b>16.4</b>	
0	0	1	16,384	1.0	1.3	1.6	2.0	2.7	4.1	<b>8.2</b>	16.4	32.8	
0	1	0	32,768	2.0	2.7	3.3	4.1	5.5	<b>8.2</b>	16.4	32.8	65.5	
0	1	1	65,536	4.1	5.5	6.6	<b>8.2</b>	<b>10.9</b>	16.4	32.8	65.5	131.1	
1	0	0/—*	131,072	<b>8.2</b>	<b>10.9</b>	<b>13.1</b>	16.4	21.8	32.8	65.5	131.1	262.1	

Notes: 1. All times are in milliseconds.

2. Recommended values are printed in boldface.

\* F-ZTAT version/ZTAT and mask-ROM versions

### 21.3.4 Sample Application of Software Standby Mode

In this example the chip enters the software standby mode when  $\overline{\text{NMI}}$  goes low and exits when  $\overline{\text{NMI}}$  goes high, as shown in figure 21-1.

The NMI edge bit (NMIEG) in the system control register is originally cleared to 0, selecting the falling edge. When  $\overline{\text{NMI}}$  goes low, the  $\overline{\text{NMI}}$  interrupt handling routine sets NMIEG to 1, sets SSBY to 1 (selecting the rising edge), then executes the SLEEP instruction. The chip enters software standby mode. It recovers from software standby mode on the next rising edge of  $\overline{\text{NMI}}$ .

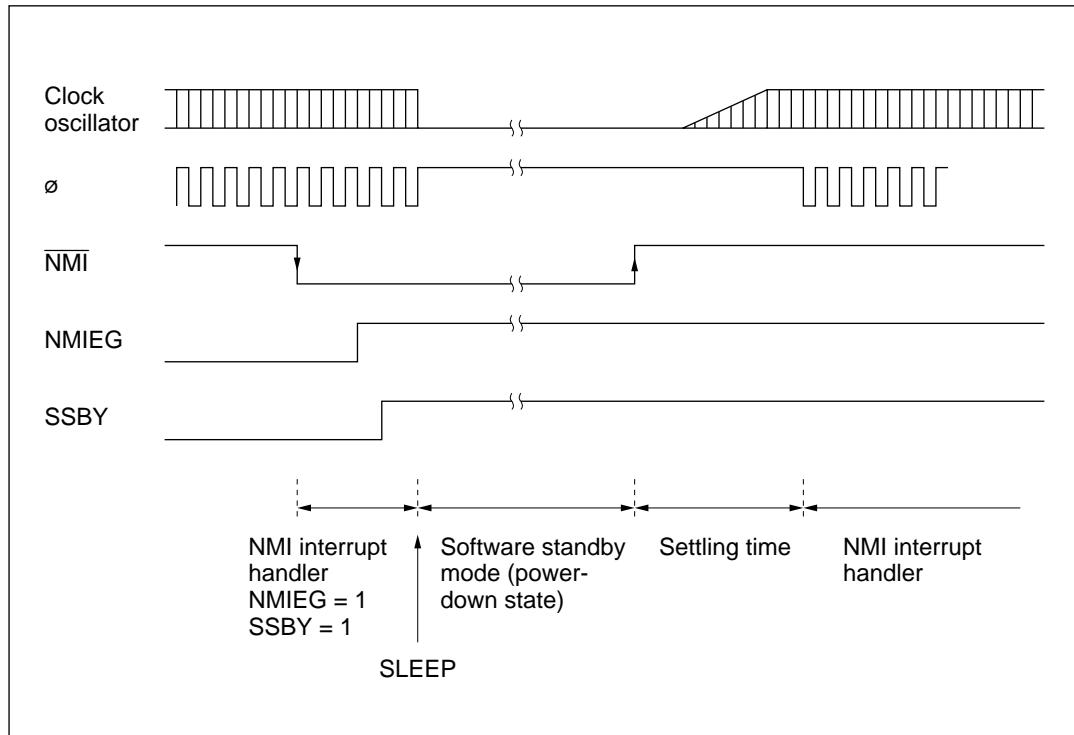


Figure 21-1 NMI Timing in Software Standby Mode

### 21.3.5 Application Note

- (1) The I/O ports retain their current states in software standby mode. If a port is in the high output state, the current dissipation caused by the output current is not reduced.

(2) For H8/3434F lots up to week code "5M\*", current dissipation is not reduced below the specification value when the SLEEP instruction is executed in the ROM space after the SSBY bit in SYSCR is set and the mode is changed to software standby mode.

This is because operation of the sense amplifier in the flash memory is not completely halted when the SLEEP instruction is executed in the ROM space and the mode is changed to software standby mode. There is a continuous Vcc current of approximately 100  $\mu$ A to 200  $\mu$ A per bit, and approximately 2 mA to 4 mA in total (when  $V_{CC} = 5.0$  V and  $T_a = 25^\circ C$ ).

Remedial measures for this problem are described below.

(a) Preliminary remedy

Current dissipation is reduced normally if the SLEEP instruction is executed in the RAM space. Please change the software as shown below.

(b) Permanent remedy

A permanent remedy will be implemented in lots with week code "6A1" onward by modifying the internal logic circuitry.

```
•  
•  
•  
BSET      #7, @SYSCR:8 ; Sets the SSBY bit  
SLEEP      ; Executes the SLEEP instruction  
•  
•  
•
```



Replace the underlined part  
(SLEEP instruction)  
with the code shown below.

```
•  
•  
•  
BSET      #7, @SYSCR:8 ; Sets the SSBY bit  
MOV.W      #H'0180, R0 ; Writes the SLEEP code H'0180  
MOV.W      R0, @H'FF00 ; to the RAM  
MOV.W      #H'5470, R0 ; Writes the RTS code H'5470  
MOV.W      R0, @H'FF02 ; to the RAM  
JSR        @H'FF00 ; Subroutine branch  
•  
•  
•
```

\*Registers and RAM addresses  
are arbitrary.

Note: The current responsible for this phenomenon also flows when the mode is changed to SLEEP mode. In order to reduce current dissipation, therefore, please use the preliminary remedy for the SLEEP instruction when changing the mode to SLEEP mode.

## 21.4 Hardware Standby Mode

### 21.4.1 Transition to Hardware Standby Mode

Regardless of its current state, the chip enters hardware standby mode whenever the  $\overline{\text{STBY}}$  pin goes low.

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.

Notes:

1. The RAME bit in the system control register should be cleared to 0 before the  $\overline{\text{STBY}}$  pin goes low.
2. Do not change the inputs at the mode pins ( $\text{MD}_1$ ,  $\text{MD}_0$ ) during hardware standby mode. Be particularly careful not to let both mode pins go low in hardware standby mode, since that places the chip in PROM mode and increases current dissipation.

### 21.4.2 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.

### 21.4.3 Timing Relationships

Figure 21-2 shows the timing relationships in hardware standby mode.

In the sequence shown, first  $\overline{\text{RES}}$  goes low, then  $\overline{\text{STBY}}$  goes low, at which point the chip enters hardware standby mode. To recover, first  $\overline{\text{STBY}}$  goes high, then after the clock settling time,  $\overline{\text{RES}}$  goes high.

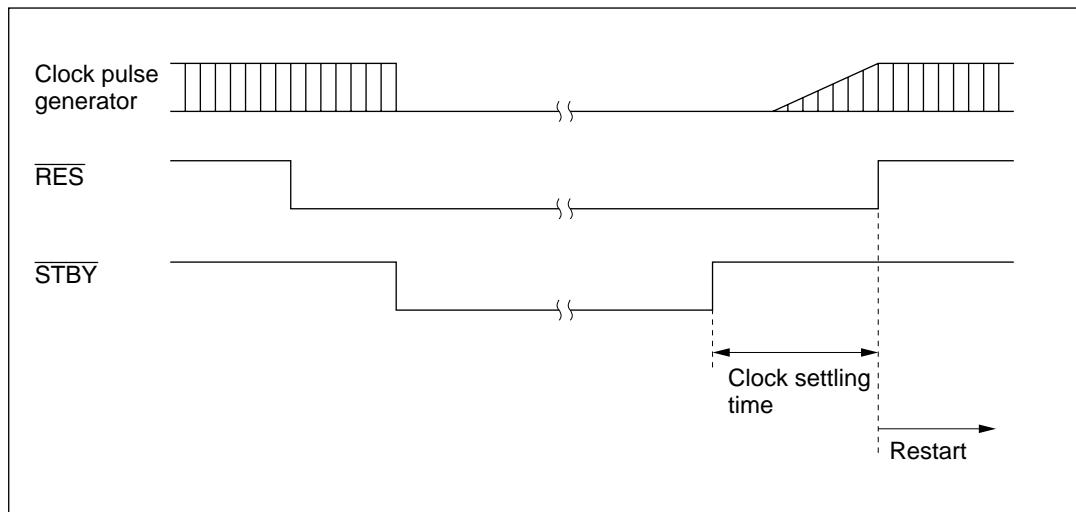


Figure 21-2 Hardware Standby Mode Timing

# Section 22 Electrical Specifications

## 22.1 Absolute Maximum Ratings

Table 22-1 lists the absolute maximum ratings.

**Table 22-1 Absolute Maximum Ratings**

Item	Symbol	Rating	Unit
Supply voltage	$V_{CC}$	−0.3 to +7.0	V
I/O buffer supply voltage	$V_{CCB}$	−0.3 to +7.0	V
Flash memory programming voltage	$FV_{PP}$	−0.3 to +13.0	V
Programming voltage	$V_{PP}$	−0.3 to +13.5	V
Input voltage	$V_{in}$ Ports 7, MD <sub>1</sub> , P8 <sub>6</sub> , P9 <sub>7</sub> , PA <sub>7</sub> to PA <sub>4</sub>	−0.3 to $V_{CC} + 0.3$	V
	$V_{in}$ P8 <sub>6</sub> , P9 <sub>7</sub> , PA <sub>7</sub> to PA <sub>4</sub>	−0.3 to $V_{CCB} + 0.3$	V
	$V_{in}$ Port 7	−0.3 to $AV_{CC} + 0.3$	V
	$V_{in}$ MD <sub>1</sub>	F-ZTAT version: −0.3 to +13.0 Other versions: −0.3 to $V_{CC} + 0.3$	V
Reference supply voltage	$AV_{ref}$	−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 Wide-range specifications: −40 to +85	°C
Storage temperature	$T_{stg}$	−55 to +125	°C

Caution: Exceeding the absolute maximum ratings shown in table 22-1 can permanently destroy the chip.\*

Note: \*  $FV_{PP}$  must not exceed 13 V and  $V_{PP}$  must not exceed 13.5 V, including allowances for peak overshoot. For the F-ZTAT version, MD<sub>1</sub> must not exceed 13 V, including an allowance for peak overshoot.

## 22.2 Electrical Characteristics

### 22.2.1 DC Characteristics

Table 22-2 lists the DC characteristics of the 5-V version. Table 22-3 lists the DC characteristics of 4-V version. Table 22-4 lists the DC characteristics of the 3 V version. Table 22-5 gives the allowable current output values of the 5-V and 4-V versions. Table 22-6 gives the allowable current output values of the 3-V version. Bus drive characteristics common to 5-V, 4-V and 3-V versions are listed in table 22-7.

**Table 22-2 DC Characteristics (5-V Version)**

Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{CC} = 5.0 \text{ V} \pm 10\%^{*1}$ ,  $AV_{ref} = 4.5 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	$V_T^-$	1.0	—	—	V	
	$V_T^+$	—	—	$V_{CC} \times 0.7$		
				$V_{CCB} \times 0.7$		
	$V_T^+ - V_T^-$	0.4	—	—		
Input high voltage	$V_{IH}$	$V_{CC} - 0.7$	—	$V_{CC} + 0.3$	V	
	$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$			
	$V_{CCB} \times 0.7$			$V_{CCB} + 0.3$		
	$P7_7$ to $P7_0$	2.0	—	$AV_{CC} + 0.3$		
	All input pins other than (1) and (2) above <sup>*7</sup>	2.0	—	$V_{CC} + 0.3$		
				$V_{CCB} + 0.3$		
Input low voltage	$V_{IL}$	-0.3	—	0.5	V	
	$P7_7$ to $P7_0$	-0.3	—	1.0		
	All input pins other than (1) and (3) above	-0.3	—	0.8		
Output high voltage	$V_{OH}$	$V_{CC} - 0.5$	—	—	V	$I_{OH} = -200 \mu\text{A}$
		$V_{CCB} - 0.5$				
		3.5	—	—		$I_{OH} = -1.0 \text{ mA}$

**Table 22-2 DC Characteristics (5-V Version) (cont)**

Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{CC} = 5.0 \text{ V} \pm 10\%^{*1}$ ,  $AV_{ref} = 4.5 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Output low voltage	All output pins (except RESO) <sup>*6</sup>	$V_{OL}$	—	—	0.4	V $I_{OL} = 1.6 \text{ mA}$
	P1 <sub>7</sub> to P1 <sub>0</sub> , P2 <sub>7</sub> to P2 <sub>0</sub>		—	—	1.0	$I_{OL} = 10.0 \text{ mA}$
	RESO		—	—	0.4	$I_{OL} = 2.6 \text{ mA}$
Input leakage current	RES, STBY	$ I_{in} $	—	—	10.0	$\mu\text{A}$ $V_{in} = 0.5 \text{ V}$ to $V_{CC} - 0.5 \text{ V}$
	NMI, MD <sub>1</sub> , MD <sub>0</sub>		—	—	1.0	
	P7 <sub>7</sub> to P7 <sub>0</sub>		—	—	1.0	$V_{in} = 0.5 \text{ V}$ to $AV_{CC} - 0.5 \text{ V}$
Leakage current in three-state (off state)	Ports 1 to 6, 8, 9, A, B, RESO <sup>*7</sup>	$ I_{TSI} $	—	—	1.0	$\mu\text{A}$ $V_{in} = 0.5 \text{ V}$ to $V_{CC} - 0.5 \text{ V}$ , $V_{in} = 0.5 \text{ V}$ to $V_{CCB} - 0.5 \text{ V}$
Input pull-up MOS current	Ports 1 to 3	$-I_P$	30	—	250	$\mu\text{A}$ $V_{in} = 0 \text{ V}$
	Ports 6, A, B		60	—	500	
Input capacitance	STBY (F-ZTAT (4) version)	$C_{in}$	—	—	120	$\text{pF}$ $V_{in} = 0 \text{ V}$ , $f = 1 \text{ MHz}$ , $T_a = 25^\circ\text{C}$
	RES, STBY (except F-ZTAT version)		—	—	60	
	NMI, MD <sub>1</sub>		—	—	50	
	PA <sub>7</sub> to PA <sub>4</sub> , P9 <sub>7</sub> , P8 <sub>6</sub>		—	—	20	
	All input pins other than (4)		—	—	15	
Current dissipation <sup>*2</sup>	Normal operation	$I_{CC}$	—	27	45	$\text{mA}$ $f = 12 \text{ MHz}$
			—	36	60	$f = 16 \text{ MHz}$
	Sleep mode		—	18	30	$f = 12 \text{ MHz}$
			—	24	40	$f = 16 \text{ MHz}$
	Standby modes <sup>*3</sup>		—	0.01	5.0	$\mu\text{A}$ $T_a \leq 50^\circ\text{C}$
			—	—	20.0	$50^\circ\text{C} < T_a$

**Table 22-2 DC Characteristics (5-V Version) (cont)**

Conditions:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{CC} = 5.0 \text{ V} \pm 10\%^{*1}$ ,  $AV_{ref} = 4.5 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Analog supply current	During A/D conversion	AI <sub>CC</sub>	—	1.2	2.0	mA
	During A/D and D/A conversion		—	1.2	2.0	
	A/D and D/A conversion idle		—	0.01	5.0	μA $AV_{CC} = 2.0 \text{ V}$ to $5.5 \text{ V}$
Reference supply current	During A/D conversion	AI <sub>ref</sub>	—	0.3	0.6	mA
	During A/D and D/A conversion		—	1.3	3.0	
	A/D and D/A conversion idle		—	0.01	5.0	μA $AV_{ref} = 2.0 \text{ V}$ to $5.5 \text{ V}$
Analog supply voltage <sup>*1</sup>	AV <sub>CC</sub>	4.5	—	5.5	V	During operation
		2.0	—	5.5		While idle or when not in use
RAM standby voltage	V <sub>RAM</sub>	2.0	—	—	V	

Notes:

1. Even when the A/D and D/A converters are not used, connect AV<sub>CC</sub> to power supply V<sub>CC</sub> and keep the applied voltage between 2.0 V and 5.5 V. At this time, make sure AV<sub>ref</sub>  $\leq$  AV<sub>CC</sub>.
2. Current dissipation values assume that  $V_{IH\ min} = V_{CC} - 0.5 \text{ V}$ ,  $V_{CCB} - 0.5 \text{ V}$ ,  $V_{IL\ max} = 0.5 \text{ V}$ , all output pins are in the no-load state, and all input pull-up transistors are off.
3. For these values it is assumed that  $V_{RAM} \leq V_{CC} < 4.5 \text{ V}$  and  $V_{IH\ min} = V_{CC} \times 0.9$ ,  $V_{CCB} \times 0.9$ ,  $V_{IL\ max} = 0.3 \text{ V}$ .
4. P6<sub>7</sub> to P6<sub>0</sub> include supporting module inputs multiplexed with them.
5.  $\overline{IRQ_2}$  includes ADTRG multiplexed with it.
6. Applies when IICS = IICE = 0. The output low level is determined separately when the bus drive function is selected.
7. The characteristics of PA<sub>7</sub> to PA<sub>4</sub>, KEYIN<sub>15</sub> to KEYIN<sub>12</sub>, P9<sub>7</sub>/WAIT, SDA, and P8<sub>6</sub>/IRQ<sub>5</sub>/SCK<sub>1</sub>, SCL depend on V<sub>CCB</sub>; the characteristics of all other pins depend on V<sub>CC</sub>.

**Table 22-3 DC Characteristics (4-V Version)**

Conditions:  $V_{CC} = 4.0$  V to  $5.5$  V,  $AV_{CCB} = 4.0$  V to  $5.5$  V,  $AV_{CC} = 4.0$  V to  $5.5$  V<sup>\*1</sup>,  $AV_{ref} = 4.0$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	$P6_7$ to $P6_0$ <sup>*4</sup> , $\overline{KEYIN}_{15}$ to, $\overline{KEYIN}_8$ , $\overline{IRQ_2}$ to $\overline{IRQ_0}$ <sup>*5</sup> , $\overline{IRQ_7}$ to $\overline{IRQ_3}$	(1) <sup>*7</sup>	$V_{T^-}$ $V_{T^+}$ $V_{T^+} - V_{T^-}$ $V_{T^-}$ $V_{T^+}$ $V_{T^+} - V_{T^-}$	1.0 — 0.4 0.8 — —	— $V_{CC} \times 0.7$ $V_{CCB} \times 0.7$ — $V_{CC} \times 0.7$ $V_{CCB} \times 0.7$ —	V $V_{CC} = 4.5$ V to $5.5$ V, $V_{CCB} =$ $4.5$ V to $5.5$ V
Input high voltage	$\overline{RES}$ , $\overline{STBY}$ , $MD_1$ , $MD_0$ , $EXTAL$ , $NMI$	(2)	$V_{IH}$	$V_{CC} - 0.7$	—	$V_{CC} + 0.3$ V
	$PA_7$ to $PA_0$ <sup>*7</sup> $SCL$ , $SDA$			$V_{CC} \times 0.7$ $V_{CCB} \times 0.7$	— —	$V_{CC} + 0.3$ $V_{CCB} + 0.3$
	$P7_7$ to $P7_0$			2.0	—	$AV_{CC} + 0.3$
	All input pins other than (1) and (2) above <sup>*7</sup>			2.0	—	$V_{CC} + 0.3$ $V_{CCB} + 0.3$
Input low voltage	$\overline{RES}$ , $\overline{STBY}$ , $MD_1$ , $MD_0$	(3)	$V_{IL}$	-0.3	—	0.5 V
	$PA_7$ to $PA_0$ <sup>*7</sup> $SCL$ , $SDA$			-0.3	—	1.0
				-0.3	—	0.8
	All input pins other than (1) and (3) above <sup>*7</sup>			-0.3	—	0.8
				-0.3	—	0.6

**Table 22-3 DC Characteristics (4-V Version) (cont)**

Conditions:  $V_{CC} = 4.0$  V to  $5.5$  V,  $AV_{CC}B = 4.0$  V to  $5.5$  V,  $AV_{CC} = 4.0$  V to  $5.5$  V<sup>\*1</sup>,  $AV_{ref} = 4.0$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Output high voltage	$V_{OH}$	$V_{CC} - 0.5$ $V_{CC}B - 0.5$	—	—	V	$I_{OH} = -200$ $\mu\text{A}$
		3.5	—	—		$I_{OH} = -1.0$ mA, $V_{CC} = 4.5$ V to $5.5$ V, $V_{CC}B =$ $4.5$ V to $5.5$ V
		2.8	—	—		$I_{OH} = -1.0$ mA, $V_{CC} = 4.0$ V to $4.5$ V, $V_{CC}B =$ $4.0$ V to $4.5$ V
Output low voltage	$V_{OL}$	—	—	0.4	V	$I_{OL} = 1.6$ mA
	$P1_7$ to $P1_0$ , $P2_7$ to $P2_0$	—	—	1.0		$I_{OL} = 10.0$ mA
	RESO	—	—	0.4		$I_{OL} = 2.6$ mA
Input leakage current	$RES, STBY$	$ I_{in} $	—	—	$\mu\text{A}$	$V_{in} = 0.5$ V to $V_{CC} - 0.5$ V
	$NMI, MD_1, MD_0$	—	—	1.0		
	$P7_7$ to $P7_0$	—	—	1.0		$V_{in} = 0.5$ V to $AV_{CC} - 0.5$ V
Leakage current in three-state (off state)	Ports 1 to 6, 8, 9, A, B, RESO <sup>*7</sup>	$ I_{TSI} $	—	—	$\mu\text{A}$	$V_{in} = 0.5$ V to $V_{CC} - 0.5$ V, $V_{in} = 0.5$ V to $V_{CC}B - 0.5$ V
Input pull-up MOS current	Ports 1 to 3	$-I_P$	30	—	$\mu\text{A}$	$V_{in} = 0$ V, $V_{CC} = 4.5$ V to $5.5$ V, $V_{CC}B =$ $4.5$ V to $5.5$ V
	Ports 6, A, B <sup>*7</sup>		60	—		
	Ports 1 to 3		20	—		$V_{in} = 0$ V, $V_{CC} = 4.0$ V to $4.5$ V, $V_{CC}B =$ $4.0$ V to $4.5$ V
	Ports 6, A, B <sup>*7</sup>		40	—		

**Table 22-3 DC Characteristics (4-V Version) (cont)**

Conditions:  $V_{CC} = 4.0$  V to  $5.5$  V,  $AV_{CCB} = 4.0$  V to  $5.5$  V,  $AV_{CC} = 4.0$  V to  $5.5$  V<sup>\*1</sup>,  $AV_{ref} = 4.0$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test
						Conditions
Input capacitance	STBY (F-ZTAT (4) version)	—	—	120	pF	$V_{in} = 0$ V, $f = 1$ MHz, $T_a = 25^\circ\text{C}$
	RES, STBY (except F-ZTAT version)	—	—	60		
	NMI, MD <sub>1</sub>	—	—	50		
	PA <sub>7</sub> to PA <sub>4</sub> , P9 <sub>7</sub> , P8 <sub>6</sub>	—	—	20		
	All input pins other than (4) above	—	—	15		
Current dissipation <sup>*2</sup>	Normal operation	I <sub>CC</sub>	—	27	45	mA $f = 12$ MHz
			—	36	60	
	Sleep mode		—	18	30	$f = 12$ MHz $f = 16$ MHz $V_{CC} = 4.5$ V to $5.5$ V
			—	24	40	
	Standby modes <sup>*3</sup>		—	0.01	5.0	$\mu\text{A}$ $T_a \leq 50^\circ\text{C}$ $50^\circ\text{C} < T_a$
			—	—	20.0	
Analog supply current	During A/D conversion	A <sub>I<sub>CC</sub></sub>	—	1.2	2.0	mA
	During A/D and D/A conversion		—	1.2	2.0	
	A/D and D/A conversion idle		—	0.01	5.0	$\mu\text{A}$ $AV_{CC} = 2.0$ V to $5.5$ V
Reference supply current	During A/D conversion	A <sub>I<sub>ref</sub></sub>	—	0.3	0.6	mA
	During A/D and D/A conversion		—	1.3	3.0	
	A/D and D/A conversion idle		—	0.01	5.0	$\mu\text{A}$ $AV_{ref} = 2.0$ V to $5.5$ V

**Table 22-3 DC Characteristics (4-V Version) (cont)**

Conditions:  $V_{CC} = 4.0$  V to  $5.5$  V,  $AV_{CC}B = 4.0$  V to  $5.5$  V,  $AV_{CC} = 4.0$  V to  $5.5$  V<sup>\*1</sup>,  $AV_{ref} = 4.0$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Analog supply voltage <sup>*1</sup>	$AV_{CC}$	4.0	—	5.5	V	During operation
		2.0	—	5.5		While idle or when not in use
RAM standby voltage	$V_{RAM}$	2.0	—	—	V	

Notes:

1. Even when the A/D and D/A converters are not used, connect  $AV_{CC}$  to power supply  $V_{CC}$  and keep the applied voltage between 2.0 V and 5.5 V. At this time, make sure  $AV_{ref} \leq AV_{CC}$ .
2. Current dissipation values assume that  $V_{IH\ min} = V_{CC} - 0.5$  V,  $V_{CC}B - 0.5$  V,  $V_{IL\ max} = 0.5$  V, all output pins are in the no-load state, and all input pull-up transistors are off.
3. For these values it is assumed that  $V_{RAM} \leq V_{CC} < 4.0$  V and  $V_{IH\ min} = V_{CC} \times 0.9$ ,  $V_{CC}B \times 0.9$ ,  $V_{IL\ max} = 0.3$  V.
4.  $P6_7$  to  $P6_0$  include supporting module inputs multiplexed with them.
5.  $\overline{IRQ}_2$  includes  $\overline{ADTRG}$  multiplexed with it.
6. Applies when  $IICS = IICE = 0$ . The output low level is determined separately when the bus drive function is selected.
7. The characteristics of  $PA_7$  to  $PA_4$ ,  $\overline{KEYIN}_{15}$  to  $\overline{KEYIN}_{12}$ ,  $P9_7/\overline{WAIT}$ ,  $SDA$ , and  $P8_6/\overline{IRQ}_5/SCK_1$ ,  $SCL$  depend on  $V_{CC}B$ ; the characteristics of all other pins depend on  $V_{CC}$ .

**Table 22-4 DC Characteristics (3-V Version)**

Conditions:  $V_{CC} = 2.7$  V to 5.5 V,  $AV_{CCB} = 2.7$  V to 5.5 V,  $AV_{CC} = 2.7$  V to 5.5 V<sup>\*1</sup>,  $AV_{ref} = 2.7$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	$P6_7$ to $P6_0$ <sup>*4</sup> , $KEYIN_{15}$ to, $KEYIN_8$ , $IRQ_2$ to $IRQ_0$ <sup>*5</sup> , $IRQ_7$ to $IRQ_3$	(1) <sup>*7</sup>	$V_{T^-}$ $V_{T^+}$ $V_{T^+} - V_{T^-}$	$V_{CC} \times 0.15$ $V_{CCB} \times 0.15$ — 0.2	— — — —	V
Input high voltage	$RES, \overline{STBY}, MD_1, MD_0, EXTAL, NMI$ $PA_7$ to $PA_0$ <sup>*7</sup> $SCL, SDA$ $P7_7$ to $P7_0$	(2)	$V_{IH}$	$V_{CC} \times 0.9$ $V_{CC} \times 0.7$ $V_{CCB} \times 0.7$ $V_{CC} \times 0.7$ $V_{CC} \times 0.7$ $V_{CCB} \times 0.7$	— — — — — —	$V_{CC} + 0.3$ $V_{CC} + 0.3$ $V_{CC} + 0.3$ $AV_{CC} + 0.3$ $V_{CC} + 0.3$ $V_{CCB} + 0.3$
Input low voltage <sup>*4</sup>	$RES, \overline{STBY}, MD_1, MD_0$ $PA_7$ to $PA_0$ <sup>*7</sup> $SCL, SDA$ All input pins other than (1) and (2) above <sup>*7</sup>	(3)	$V_{IL}$	-0.3 -0.3 -0.3	— — —	$V_{CC} \times 0.1$ $V_{CC} \times 0.15$ $V_{CC} \times 0.15$
Output high voltage	All output pins (except $\overline{RESO}$ ) <sup>*6,*7</sup>		$V_{OH}$	$V_{CC} - 0.5$ $V_{CCB} - 0.5$ $V_{CC} - 1.0$ $V_{CCB} - 1.0$	— — — —	V $I_{OH} = -200 \mu\text{A}$ $I_{OH} = -1 \text{ mA}$
Output low voltage	All output pins (except $\overline{RESO}$ ) <sup>*6</sup> $P1_7$ to $P1_0$ , $P2_7$ to $P2_0$ $\overline{RESO}$		$V_{OL}$	— — —	0.4 0.4 0.4	V $I_{OL} = 0.8 \text{ mA}$ $I_{OL} = 1.6 \text{ mA}$ $I_{OL} = 1.6 \text{ mA}$
Input leakage current	$RES, \overline{STBY}$ $NMI, MD_1, MD_0$ $P7_7$ to $P7_0$		$ I_{in} $	— — —	10.0 1.0 1.0	$\mu\text{A}$ $V_{in} = 0.5 \text{ V to } V_{CC} - 0.5 \text{ V}$ $V_{in} = 0.5 \text{ V to } AV_{CC} - 0.5 \text{ V}$

**Table 22-4 DC Characteristics (3-V Version) (cont)**

Conditions:  $V_{CC} = 2.7$  V to 5.5 V,  $AV_{CC}B = 2.7$  V to 5.5 V,  $AV_{CC} = 2.7$  V to 5.5 V<sup>\*1</sup>,  
 $AV_{ref} = 2.7$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $Ta = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Leakage current in three-state (off state)	Ports 1 to 6, 8, 9, A, B, RESO <sup>*7</sup>	$ I_{TSI} $	—	—	1.0	$\mu\text{A}$	$V_{in} = 0.5$ V to $V_{CC} - 0.5$ V, $V_{in} = 0.5$ V to $V_{CC}B - 0.5$ V
Input pull-up MOS current	Ports 1 to 3	$-I_P$	3	—	120	$\mu\text{A}$	$V_{in} = 0$ V, $V_{CC} = 2.7$ V to 3.6 V, $V_{CC}B = 2.7$ V to 3.6 V
	Ports 6, A, B <sup>*7</sup>		30	—	250		
Input capacitance	STBY (F-ZTAT (4) version)	$C_{in}$	—	—	120	$\text{pF}$	$V_{in} = 0$ V, $f = 1$ MHz, $Ta = 25^\circ\text{C}$
	RES, STBY (except F-ZTAT version)		—	—	60		
	NMI, MD <sub>1</sub>		—	—	50		
	PA <sub>7</sub> to PA <sub>4</sub> , P9 <sub>7</sub> , P8 <sub>6</sub>		—	—	20		
	All input pins other than (4) above		—	—	15		
Current dissipation <sup>*2</sup>	Normal operation	$I_{CC}$	—	7	—	$\text{mA}$	$f = 6$ MHz, $V_{CC} = 2.7$ V to 3.6 V
			—	12	22		$f = 10$ MHz, $V_{CC} = 2.7$ V to 3.6 V
			—	25	—		$f = 10$ MHz, $V_{CC} = 4.0$ V to 5.5 V
	Sleep mode		—	5	—		$f = 6$ MHz, $V_{CC} = 2.7$ V to 3.6 V
			—	9	16		$f = 10$ MHz, $V_{CC} = 2.7$ V to 3.6 V
			—	18	—		$f = 10$ MHz, $V_{CC} = 4.0$ V to 5.5 V
	Standby modes <sup>*3</sup>		—	0.01	5.0	$\mu\text{A}$	$Ta \leq 50^\circ\text{C}$
			—	—	20.0		$50^\circ\text{C} < Ta$

**Table 22-4 DC Characteristics (3-V Version) (cont)**

Conditions:  $V_{CC} = 2.7$  V to  $5.5$  V,  $AV_{CC}B = 2.7$  V to  $5.5$  V,  $AV_{CC} = 2.7$  V to  $5.5$  V<sup>\*1</sup>,  
 $AV_{ref} = 2.7$  V to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item	Symbol	Min	Typ	Max	Test Conditions	
					Unit	
Analog supply current	During A/D conversion	—	1.2	2.0	mA	$AV_{CC} = 2.0$ V to $5.5$ V
	During A/D and D/A conversion	—	1.2	2.0	mA	
	A/D and D/A conversion idle	—	0.01	5.0	µA	
Reference supply current	During A/D conversion	—	0.3	0.6	mA	$AV_{ref} = 2.0$ V to $5.5$ V
	During A/D and D/A conversion	—	1.3	3.0	mA	
	A/D and D/A conversion idle	—	0.01	5.0	µA	
Analog supply voltage <sup>*1</sup>	$AV_{CC}$	2.7	—	5.5	V	During operation
		2.0	—	5.5	V	While idle or when not in use
RAM standby voltage	$V_{RAM}$	2.0	—	—	V	

Notes:

1. Even when the A/D and D/A converters are not used, connect  $AV_{CC}$  to power supply  $V_{CC}$  and keep the applied voltage between 2.0 V and 5.5 V. At this time, make sure  $AV_{ref} \leq AV_{CC}$ .
2. Current dissipation values assume that  $V_{IH\ min} = V_{CC} - 0.5$  V,  $V_{CC}B - 0.5$  V,  $V_{IL\ max} = 0.5$  V, all output pins are in the no-load state, and all input pull-up transistors are off.
3. For these values it is assumed that  $V_{RAM} \leq V_{CC} < 2.7$  V and  $V_{IH\ min} = V_{CC} \times 0.9$ ,  $V_{CC}B \times 0.9$ ,  $V_{IL\ max} = 0.3$  V.
4. P6<sub>7</sub> to P6<sub>0</sub> include supporting module inputs multiplexed with them.
5. IRQ<sub>2</sub> includes ADTRG multiplexed with it.
6. Applies when IICS = IICE = 0. The output low level is determined separately when the bus drive function is selected.
7. The characteristics of PA<sub>7</sub> to PA<sub>4</sub>, KEYIN<sub>15</sub> to KEYIN<sub>12</sub>, P9<sub>7</sub>/WAIT, SDA, and P8<sub>6</sub>/IRQ<sub>5</sub>/SCK<sub>1</sub>, SCL depend on  $V_{CC}B$ ; the characteristics of all other pins depend on  $V_{CC}$ .

**Table 22-5 Allowable Output Current Values (5-V and 4-V Versions)**

Conditions:  $V_{CC} = 4.0 \text{ V to } 5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V to } 5.5 \text{ V}$ ,  $AV_{CC} = 4.0 \text{ V to } 5.5 \text{ V}$ ,  $AV_{ref} = 4.0 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Item		Symbol	Min	Typ	Max	Unit
Allowable output low current (per pin)	SCL, SDA, PA <sub>4</sub> to PA <sub>7</sub> (bus drive selection)	I <sub>OL</sub>	—	—	20	mA
	Ports 1 and 2		—	—	10	
	RESO		—	—	3	
	Other output pins		—	—	2	
Allowable output low current (total)	Ports 1 and 2, total	$\Sigma I_{OL}$	—	—	80	mA
	Total of all output		—	—	120	
Allowable output high current (per pin)	All output pins	-I <sub>OH</sub>	—	—	2	mA
Allowable output high current (total)	Total of all output	$\Sigma -I_{OH}$	—	—	40	mA

**Table 22-6 Allowable Output Current Values (3-V Version)**

Conditions:  $V_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ to } 5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V to } 5.5 \text{ V}$ ,  $AV_{ref} = 2.7 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item		Symbol	Min	Typ	Max	Unit
Allowable output low current (per pin)	SCL, SDA, PA <sub>4</sub> to PA <sub>7</sub> (bus drive selection)	I <sub>OL</sub>	—	—	10	mA
	Ports 1 and 2		—	—	2	
	RESO		—	—	1	
	Other output pins		—	—	1	
Allowable output low current (total)	Ports 1 and 2, total	$\Sigma I_{OL}$	—	—	40	mA
	Total of all output		—	—	60	
Allowable output high current (per pin)	All output pins	-I <sub>OH</sub>	—	—	2	mA
Allowable output high current (total)	Total of all output	$\Sigma -I_{OH}$	—	—	30	mA

Note: To avoid degrading the reliability of the chip, be careful not to exceed the output current values in tables 22-5 and 22-6. 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 22-1 and 22-2.

H8/3437

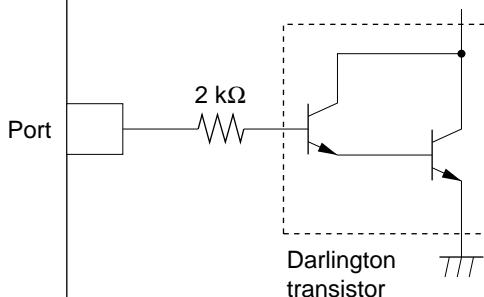


Figure 22-1 Example of Circuit for Driving a Darlington Transistor (5-V Version)

H8/3437

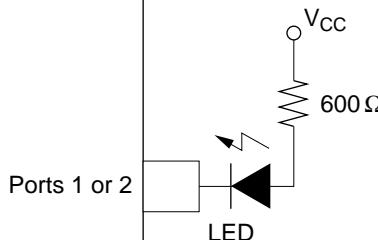


Figure 22-2 Example of Circuit for Driving an LED (5-V Version)

Table 22-7 Bus Drive Characteristics

Conditions:  $V_{CC} = 2.7$  V to 5.5 V,  $V_{SS} = 0$  V,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$ 

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Output low level voltage	$V_{OL}$	—	—	0.5	V	$V_{CCB} = 5$ V $\pm 10\%$ $I_{OL} = 16$ mA
PA <sub>4</sub> to PA <sub>7</sub> (bus drive selection)		—	—	0.5		$V_{CCB} = 2.7$ V to 5.5 V $I_{OL} = 8$ mA

## 22.2.2 AC Characteristics

The AC characteristics are listed in four tables. Bus timing parameters are given in table 22-8, control signal timing parameters in table 22-9, timing parameters of the on-chip supporting modules in table 22-10, I<sup>2</sup>C bus timing parameters in table 22-11, and external clock output stabilization delay time in table 22-12.

**Table 22-8 Bus Timing**

Condition A:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20 \text{ to } +75^\circ\text{C}$  (regular specifications),  $T_a = -40 \text{ to } +85^\circ\text{C}$  (wide-range specifications)

Condition B:  $V_{CC} = 4.0 \text{ V} \text{ to } 5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V} \text{ to } 5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20 \text{ to } +75^\circ\text{C}$  (regular specifications),  $T_a = -40 \text{ to } +85^\circ\text{C}$  (wide-range specifications)

Condition C:  $V_{CC} = 2.7 \text{ V} \text{ to } 5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ V} \text{ to } 5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20 \text{ to } +75^\circ\text{C}$

Item	Symbol	Condition C		Condition B		Condition A		Unit	Test Conditions
		10 MHz	12 MHz	10 MHz	12 MHz	16 MHz			
Clock cycle time	$t_{cyc}$	100	500	83.3	500	62.5	500	ns	Fig. 22-4
Clock pulse width low	$t_{CL}$	30	—	30	—	20	—	ns	
Clock pulse width high	$t_{CH}$	30	—	30	—	20	—	ns	
Clock rise time	$t_{Cr}$	—	20	—	10	—	10	ns	
Clock fall time	$t_{Cf}$	—	20	—	10	—	10	ns	
Address delay time	$t_{AD}$	—	50	—	35	—	30	ns	
Address hold time	$t_{AH}$	20	—	15	—	10	—	ns	
Address strobe delay time	$t_{ASD}$	—	50	—	35	—	30	ns	
Write strobe delay time	$t_{WSD}$	—	50	—	35	—	30	ns	
Strobe delay time	$t_{SD}$	—	50	—	35	—	30	ns	
Write strobe pulse width*1	$t_{WSW}$	110	—	90	—	60	—	ns	
Address setup time 1*1	$t_{AS1}$	15	—	10	—	10	—	ns	
Address setup time 2*1	$t_{AS2}$	65	—	50	—	40	—	ns	
Read data setup time	$t_{RDS}$	35	—	20	—	20	—	ns	
Read data hold time*1	$t_{RDH}$	0	—	0	—	0	—	ns	
Read data access time*1	$t_{ACC}$	—	170	—	160	—	110	ns	
Write data delay time	$t_{WDD}$	—	80/75*2	—	65/60*2	—	60	ns	
Write data setup time	$t_{WDS}$	0/5*2	—	0/5*2	—	0/5*2	—	ns	
Write data hold time	$t_{WDH}$	20	—	20	—	20	—	ns	
Wait setup time	$t_{WTS}$	40	—	35	—	30	—	ns	Fig. 22-5
Wait hold time	$t_{WTH}$	10	—	10	—	10	—	ns	

Notes: \*1 Values at maximum operating frequency

\*2 H8/3437 F-ZTAT version/other products

**Table 22-9 Control Signal Timing**

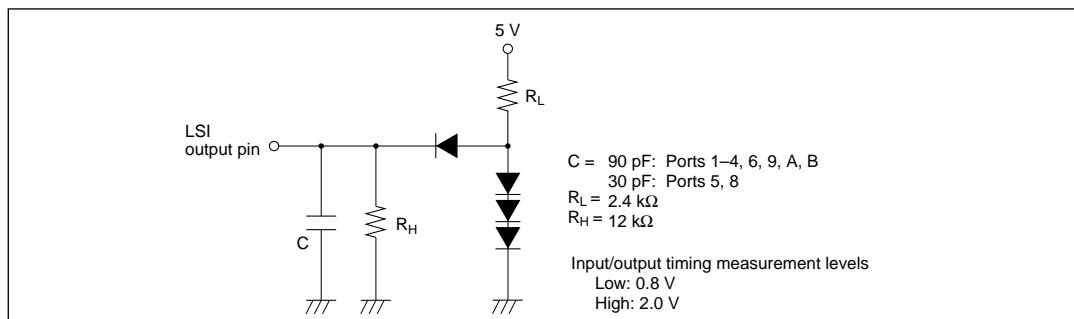
Condition A:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition B:  $V_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition C:  $V_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item	Symbol	Condition C		Condition B		Condition A		Test Conditions	
		10 MHz		12 MHz		16 MHz			
		Min	Max	Min	Max	Min	Max		
$\overline{\text{RES}}$ setup time	$t_{\text{RESS}}$	300	—	200	—	200	—	ns	
$\overline{\text{RES}}$ pulse width	$t_{\text{RESW}}$	10	—	10	—	10	—	$t_{\text{cyc}}$	
$\overline{\text{RESO}}$ output delay time	$t_{\text{RESD}}$	—	200	—	120	—	100	ns	
$\overline{\text{RESO}}$ output pulse width	$t_{\text{RESOW}}$	132	—	132	—	132	—	$t_{\text{cyc}}$	
$\overline{\text{NMI}}$ setup time ( $\overline{\text{NMI}}$ , $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_7}$ )	$t_{\text{NMIS}}$	300	—	150	—	150	—	ns	
$\overline{\text{NMI}}$ hold time ( $\overline{\text{NMI}}$ , $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_7}$ )	$t_{\text{NMIH}}$	10	—	10	—	10	—	ns	
Interrupt pulse width for recovery from soft- ware standby mode ( $\overline{\text{NMI}}$ , $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_2}$ , $\overline{\text{IRQ}_6}$ )	$t_{\text{NMIW}}$	300	—	200	—	200	—	ns	
Crystal oscillator settling time (reset)	$t_{\text{OSC1}}$	20	—	20	—	20	—	ms	
Crystal oscillator settling time (software standby)	$t_{\text{OSC2}}$	8	—	8	—	8	—	ms	

• **Measurement Conditions for AC Characteristics**



**Figure 22-3 Measurement Conditions for A/C Characteristics**

**Table 22-10 Timing Conditions of On-Chip Supporting Modules**

Condition A:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition B:  $V_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition C:  $V_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item	Symbol	Condition C		Condition B		Condition A		Unit	Test Conditions		
		10 MHz		12 MHz		16 MHz					
		Min	Max	Min	Max	Min	Max				
FRT	Timer output delay time	$t_{FTOD}$	—	150	—	100	—	100	ns		
	Timer input setup time	$t_{FTIS}$	80	—	50	—	50	—	ns		
	Timer clock input setup time	$t_{FTCS}$	80	—	50	—	50	—	ns		
	Timer clock pulse width	$t_{FTCWH}$ $t_{FTCWL}$	1.5	—	1.5	—	1.5	—	$t_{cyc}$		
TMR	Timer output delay time	$t_{TMOD}$	—	150	—	100	—	100	ns		
	Timer reset input setup time	$t_{TMRS}$	80	—	50	—	50	—	ns		
	Timer clock input setup time	$t_{TMCS}$	80	—	50	—	50	—	ns		
	Timer clock pulse width (single edge)	$t_{TMCWH}$	1.5	—	1.5	—	1.5	—	$t_{cyc}$		
	Timer clock pulse width (both edges)	$t_{TMCWL}$	2.5	—	2.5	—	2.5	—	$t_{cyc}$		
PWM	Timer output delay time	$t_{PWOD}$	—	150	—	100	—	100	ns		
SCI	Input clock cycle (Async)	$t_{Scyc}$	4	—	4	—	4	—	$t_{cyc}$		
	(Sync)	$t_{Scyc}$	6	—	6	—	6	—	$t_{cyc}$		
	Transmit data delay time (Sync)	$t_{TXD}$	—	200	—	100	—	100	ns		
	Receive data setup time (Sync)	$t_{RXS}$	150	—	100	—	100	—	ns		
	Receive data hold time (Sync)	$t_{RXH}$	150	—	100	—	100	—	ns		
	Input clock pulse width	$t_{SCKW}$	0.4	0.6	0.4	0.6	0.4	0.6	$t_{Scyc}$		

**Table 22-10 Timing Conditions of On-Chip Supporting Modules (cont)**

Condition A:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition B:  $V_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition C:  $V_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$

Item	Symbol	Condition C		Condition B		Condition A		Unit	Test Conditions		
		10 MHz		12 MHz		16 MHz					
		Min	Max	Min	Max	Min	Max				
Ports	Output data delay time	$t_{PWD}$	—	150	—	100	—	100	ns		
	Input data setup time	$t_{PRS}$	80	—	50	—	50	—	ns		
	Input data hold time	$t_{PRH}$	80	—	50	—	50	—	ns		
HIF read cycle	$\overline{CS}/\overline{HA}_0$ setup time	$t_{HAR}$	10	—	10	—	10	—	ns		
	$\overline{CS}/\overline{HA}_0$ hold time	$t_{HRA}$	10	—	10	—	10	—	ns		
	$\overline{IOR}$ pulse width	$t_{HRPW}$	220	—	120	—	120	—	ns		
	HDB delay time	$t_{HRD}$	—	200	—	100	—	100	ns		
	HDB hold time	$t_{HRF}$	0	40	0	25	0	25	ns		
	HIRQ delay time	$t_{HIRQ}$	—	200	—	120	—	120	ns		
HIF write cycle	$\overline{CS}/\overline{HA}_0$ setup time	$t_{HAW}$	10	—	10	—	10	—	ns		
	$\overline{CS}/\overline{HA}_0$ hold time	$t_{HWA}$	10	—	10	—	10	—	ns		
	$\overline{IOW}$ pulse width	$t_{HWPW}$	100	—	60	—	60	—	ns		
	HDB High-speed setup time	$t_{HDW}$	50	—	30	—	30	—	ns		
	High-speed GATE $A_{20}$ not used		85	—	55	—	45	—			
	High-speed GATE $A_{20}$ used										
	HDB hold time	$t_{HWD}$	25	—	15	—	15	—	ns		
	GA <sub>20</sub> delay time	$t_{HGA}$	—	180	—	90	—	90	ns		

**Table 22-11 I<sup>2</sup>C Bus Timing**Conditions: V<sub>CC</sub> = 2.7 V to 5.5 V, V<sub>CCB</sub> = 2.7 V to 5.5 V, V<sub>SS</sub> = 0 V, Ta = -20°C to +75°C

Item	Symbol	Min	Typ	Max	Unit	Test Conditions	Note
SCL clock cycle time	t <sub>SCL</sub>	12 t <sub>cyc</sub>	—	—	ns		Fig. 22-21
SCL clock high pulse width	t <sub>SCLH</sub>	3 t <sub>cyc</sub>	—	—	ns		
SCL clock low pulse width	t <sub>SCLL</sub>	5 t <sub>cyc</sub>	—	—	ns		
SCL and SDA rise time	t <sub>SR</sub>	—	—	1000	ns	Normal mode 100 kbits/s (max)	
		20 + 0.1C <sub>b</sub>	—	300		High-speed mode 400 kbits/s (max)	
SCL and SDA fall time	t <sub>SF</sub>	—	—	300	ns	Normal mode 100 kbits/s (max)	
		20 + 0.1C <sub>b</sub>	—	300		High-speed mode 400 kbits/s (max)	
SDA bus-free time	t <sub>BUF</sub>	7 t <sub>cyc</sub> - 300	—	—	ns		
SCL start condition hold time	t <sub>STAH</sub>	3 t <sub>cyc</sub>	—	—	ns		
SCL resend start condition setup time	t <sub>STAS</sub>	3 t <sub>cyc</sub>	—	—	ns		
SDA stop condition setup time	t <sub>STOS</sub>	3 t <sub>cyc</sub>	—	—	ns		
SDA data setup time	t <sub>SDAS</sub>	1 t <sub>cyc</sub> + 10	—	—	ns		
SDA data hold time	t <sub>SDAH</sub>	0	—	—	ns		
SDA load capacitance	C <sub>b</sub>	—	—	400	pF		

**Table 22-12 External Clock Output Stabilization Delay Time**Conditions: V<sub>CC</sub> = 2.7 V to 5.5 V, AV<sub>CC</sub> = 2.7 V to 5.5 V, V<sub>SS</sub> = AV<sub>SS</sub> = 0 V, Ta = -40°C to +85°C

Item	Symbol	Min	Max	Unit	Notes
External clock output stabilization delay time	t <sub>DEXT</sub> *	500	—	μs	Figure 22-23

Note: \* t<sub>DEXT</sub> includes a 10 t<sub>cyc</sub>  $\overline{\text{RES}}$  pulse width (t<sub>RESW</sub>).

### 22.2.3 A/D Converter Characteristics

Table 22-13 lists the characteristics of the on-chip A/D converter.

**Table 22-13 A/D Converter Characteristics**

Condition A:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{ref} = 4.5 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition B:  $V_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{ref} = 4.0 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition C:  $V_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{ref} = 2.7 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20$  to  $+75^\circ\text{C}$

Item	Condition C			Condition B			Condition A			Unit
	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Resolution	10	10	10	10	10	10	10	10	10	Bits
Conversion (single mode)*	—	—	13.4	—	—	11.2	—	—	8.4	μs
Analog input capacitance	—	—	20	—	—	20	—	—	20	pF
Allowable signal source impedance	—	—	5	—	—	10	—	—	10	kΩ
Nonlinearity error	—	—	±6.0	—	—	±3.0	—	—	±3.0	LSB
Offset error	—	—	±4.0	—	—	±3.5	—	—	±3.5	LSB
Full-scale error	—	—	±4.0	—	—	±3.5	—	—	±3.5	LSB
Quantizing error	—	—	±0.5	—	—	±0.5	—	—	±0.5	LSB
Absolute accuracy	—	—	±8.0	—	—	±4.0	—	—	±4.0	LSB

Note: \* Values at maximum operating frequency

## 22.2.4 D/A Converter Characteristics

Table 22-14 lists the characteristics of the on-chip D/A converter.

**Table 22-14 D/A Converter Characteristics**

Condition A:  $V_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $V_{CCB} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{CC} = 5.0 \text{ V} \pm 10\%$ ,  $AV_{ref} = 4.5 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition B:  $V_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{CC} = 4.0 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{ref} = 4.0 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20$  to  $+75^\circ\text{C}$  (regular specifications),  $T_a = -40$  to  $+85^\circ\text{C}$  (wide-range specifications)

Condition C:  $V_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $V_{CCB} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{CC} = 2.7 \text{ V}$  to  $5.5 \text{ V}$ ,  $AV_{ref} = 2.7 \text{ V}$  to  $AV_{CC}$ ,  $V_{SS} = AV_{SS} = 0 \text{ V}$ ,  $\phi = 2.0 \text{ MHz}$  to maximum operating frequency,  $T_a = -20$  to  $+75^\circ\text{C}$

Item	Condition C			Condition B			Condition A			Test Conditions
	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Resolution	8	8	8	8	8	8	8	8	8	Bits
Conversion time (settling time)	—	—	10.0	—	—	10.0	—	—	10.0	μs
Absolute accuracy	—	±2.0	±3.0	—	±1.0	±1.5	—	±1.0	±1.5	LSB
	—	—	±2.0	—	—	±1.0	—	—	±1.0	LSB
										2 MΩ load resistance
										4 MΩ load resistance

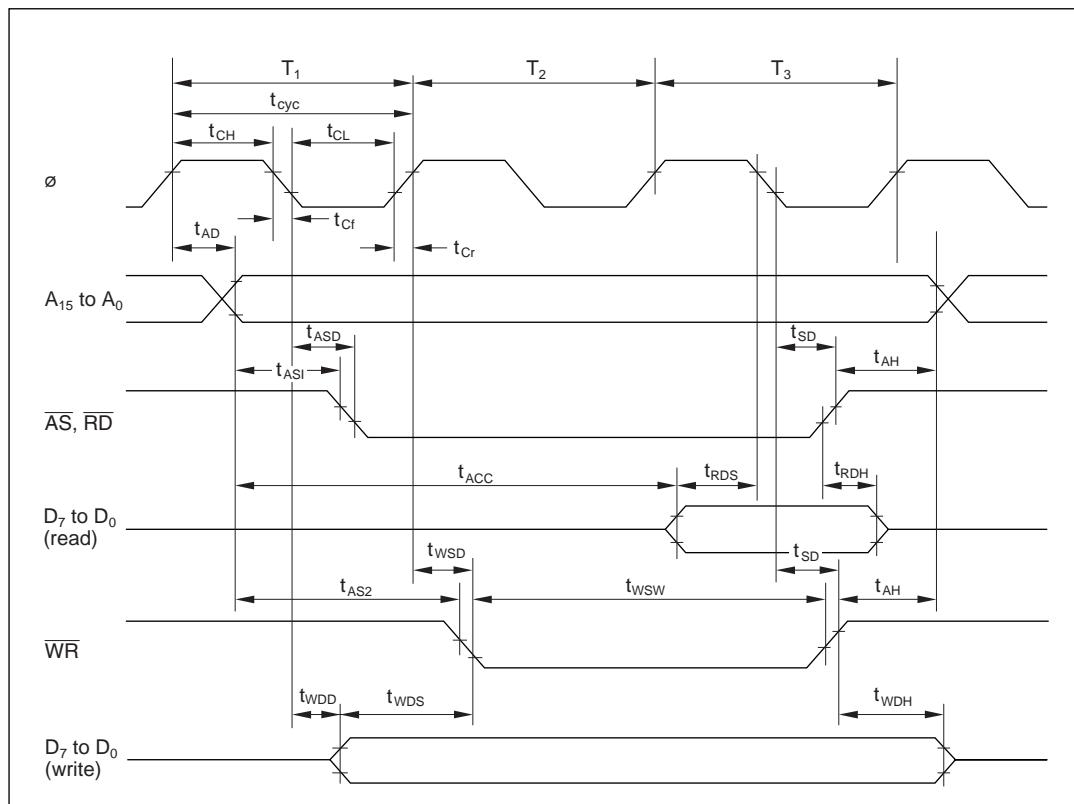
## 22.3 MCU Operational Timing

This section provides the following timing charts:

22.3.1 Bus Timing	Figures 22-4 to 22-5
22.3.2 Control Signal Timing	Figures 22-6 to 22-9
22.3.3 16-Bit Free-Running Timer Timing	Figures 22-10 to 22-11
22.3.4 8-Bit Timer Timing	Figures 22-12 to 22-14
22.3.5 PWM Timer Timing	Figure 22-15
22.3.6 SCI Timing	Figures 22-16 to 22-17
22.3.7 I/O Port Timing	Figure 22-18
22.3.8 Host Interface Timing	Figures 22-19 and 22-20
22.3.9 I <sup>2</sup> C Bus Timing	Figure 22-21
22.3.10 Reset Output Timing	Figure 22-22
22.3.11 External Clock Output Timing	Figure 22-23

### 22.3.1 Bus Timing

#### (1) Basic Bus Cycle (without Wait States) in Expanded Modes



**Figure 22-4 Basic Bus Cycle (without Wait States) in Expanded Modes**

## (2) Basic Bus Cycle (with 1 Wait State) in Expanded Modes

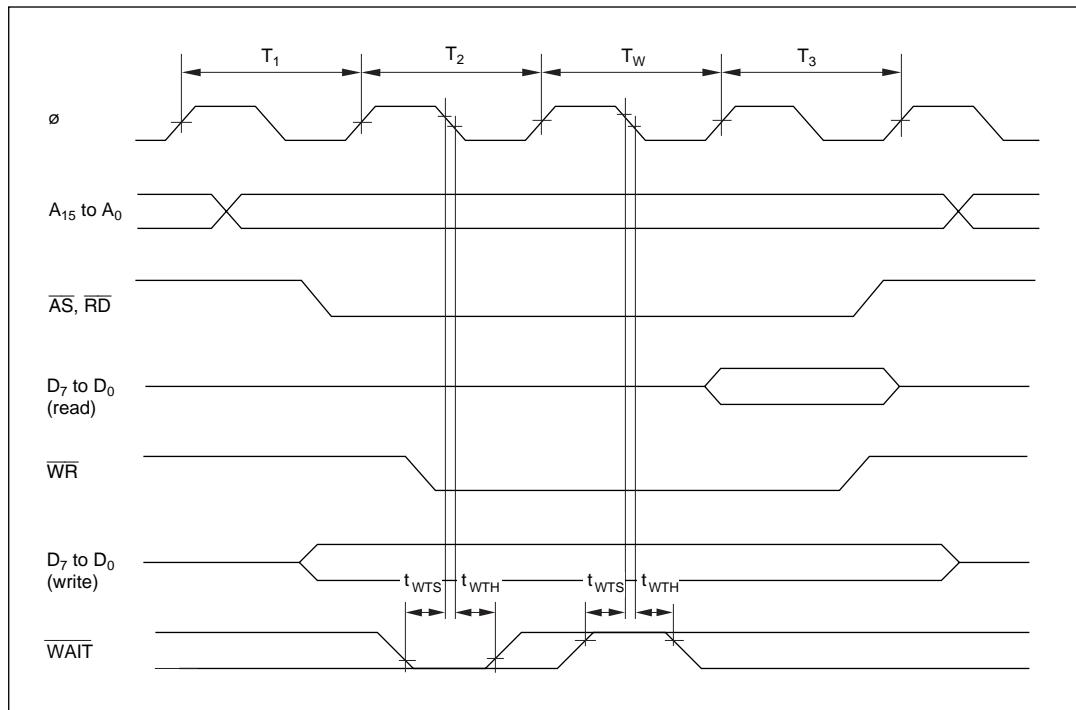


Figure 22-5 Basic Bus Cycle (with 1 Wait State) in Expanded Modes

### 22.3.2 Control Signal Timing

#### (1) Reset Input Timing

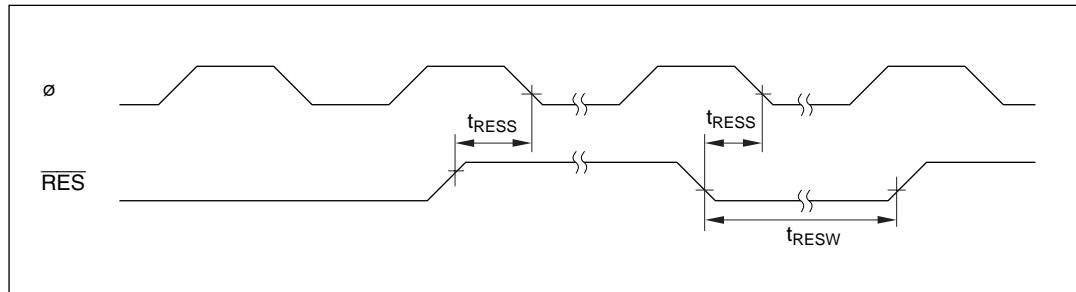


Figure 22-6 Reset Input Timing

#### (2) Interrupt Input Timing

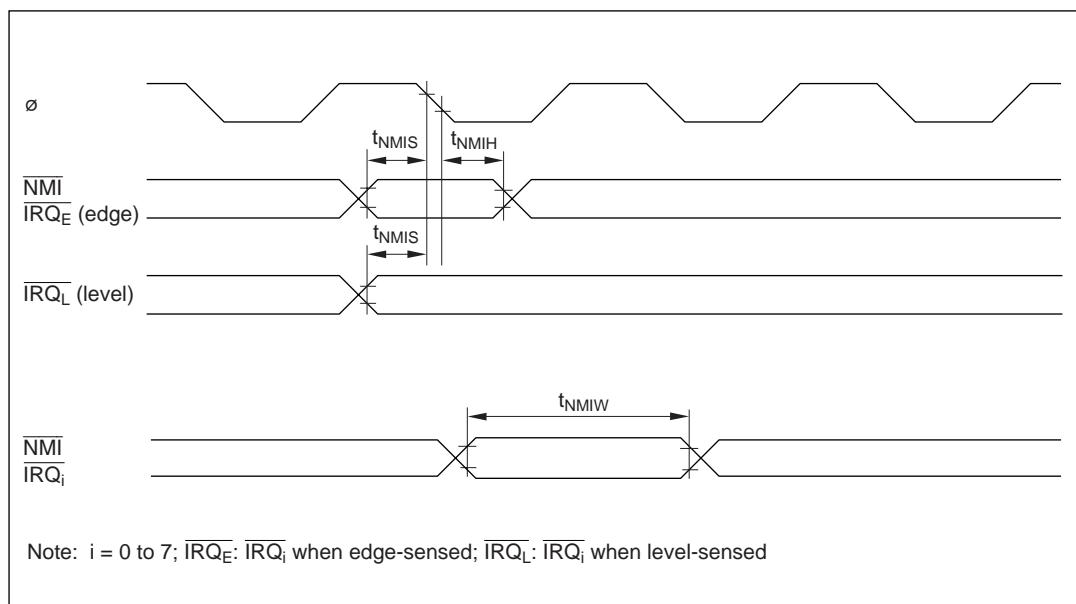


Figure 22-7 Interrupt Input Timing

### (3) Clock Settling Timing

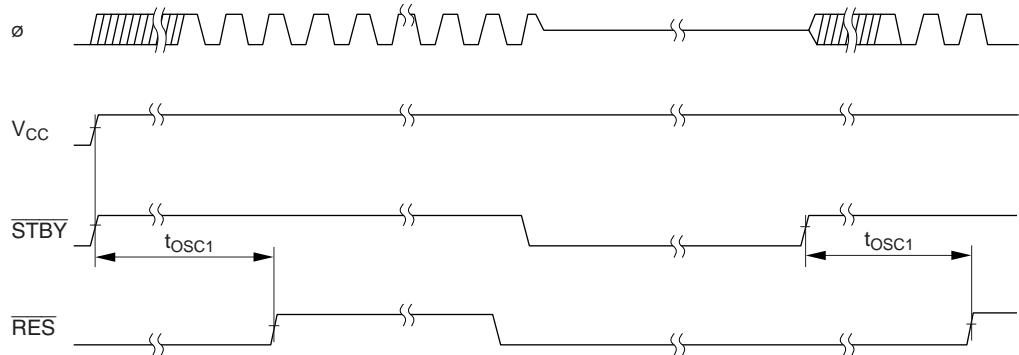


Figure 22-8 Clock Settling Timing

### (4) Clock Settling Timing for Recovery from Software Standby Mode

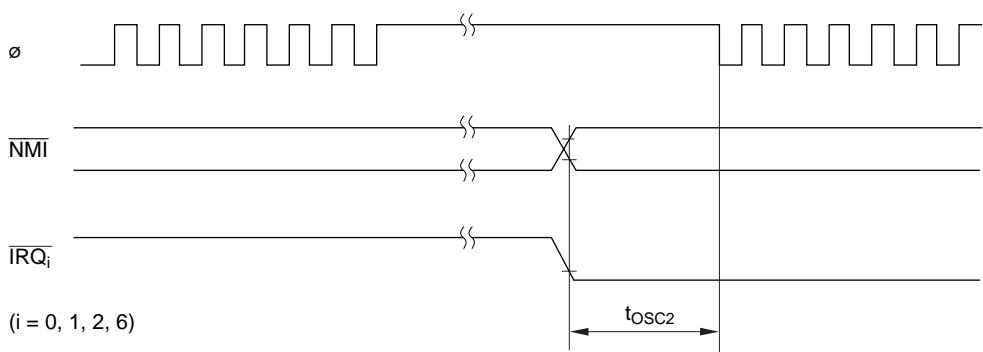


Figure 22-9 Clock Settling Timing for Recovery from Software Standby Mode

### 22.3.3 16-Bit Free-Running Timer Timing

#### (1) Free-Running Timer Input/Output Timing

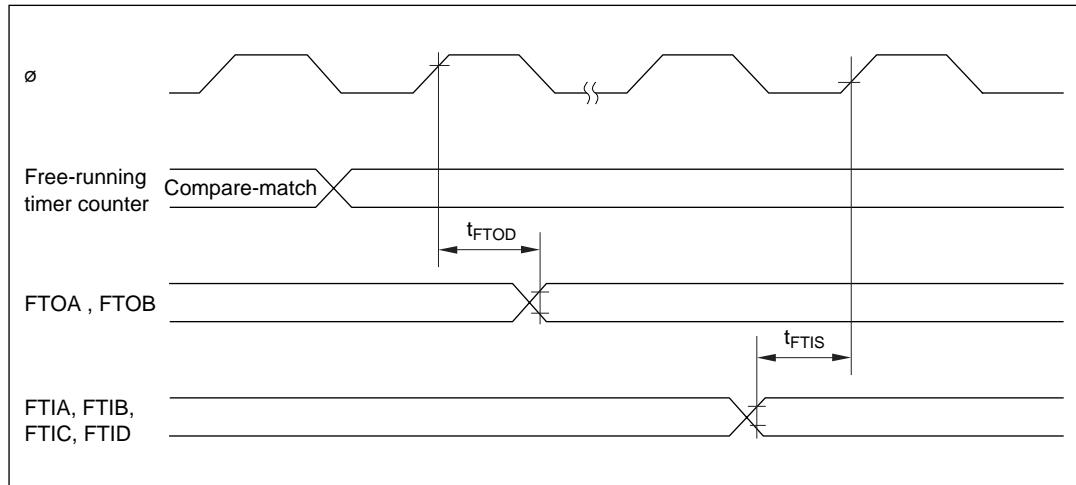


Figure 22-10 Free-Running Timer Input/Output Timing

#### (2) External Clock Input Timing for Free-Running Timer

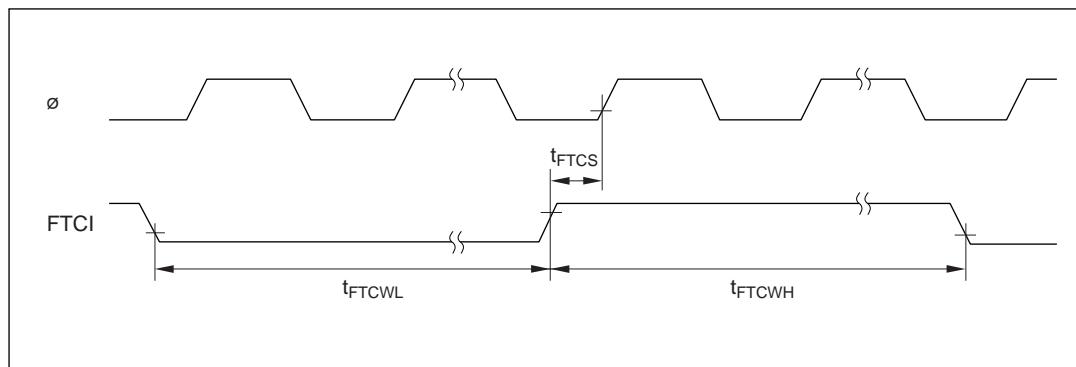


Figure 22-11 External Clock Input Timing for Free-Running Timer

## 22.3.4 8-Bit Timer Timing

### (1) 8-Bit Timer Output Timing

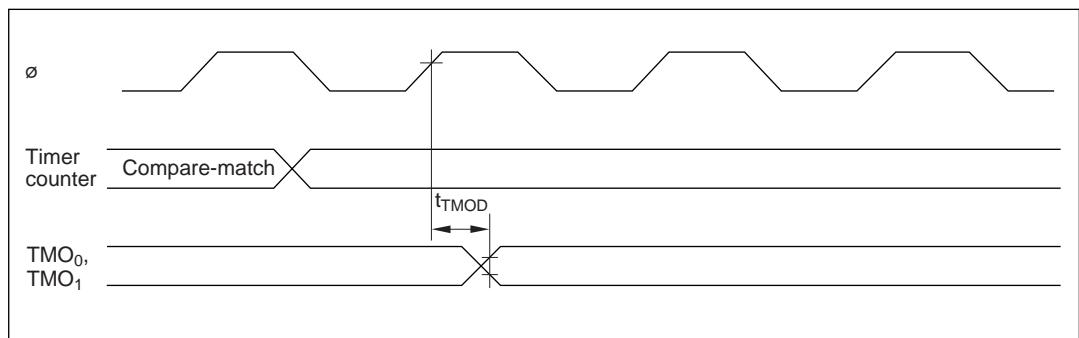


Figure 22-12 8-Bit Timer Output Timing

### (2) 8-Bit Timer Clock Input Timing

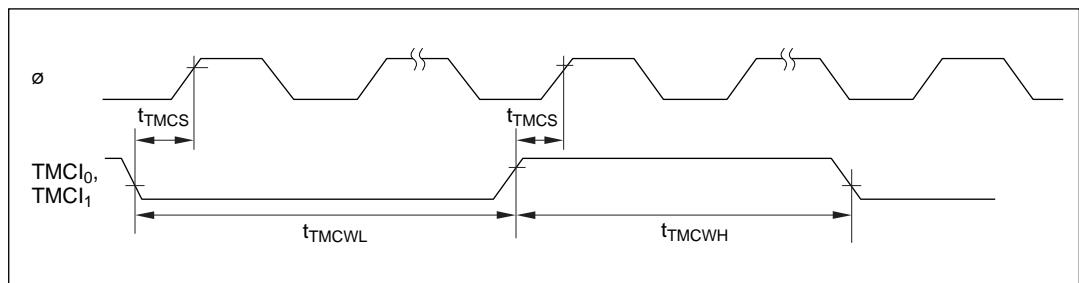


Figure 22-13 8-Bit Timer Clock Input Timing

### (3) 8-Bit Timer Reset Input Timing

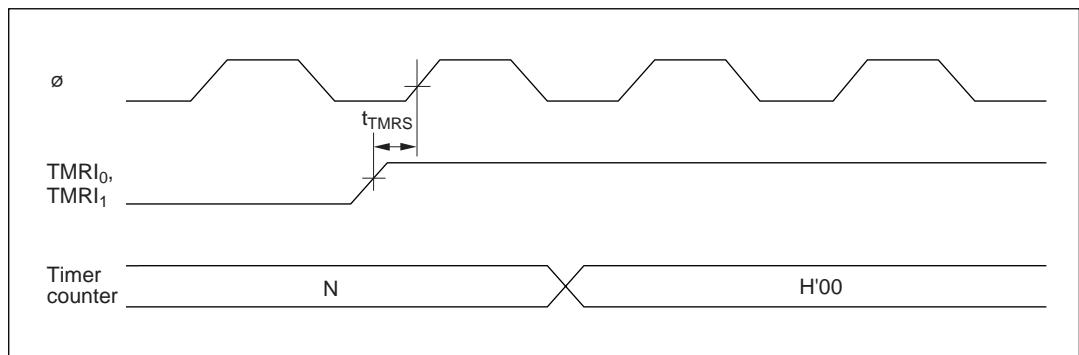


Figure 22-14 8-Bit Timer Reset Input Timing

### 22.3.5 Pulse Width Modulation Timer Timing

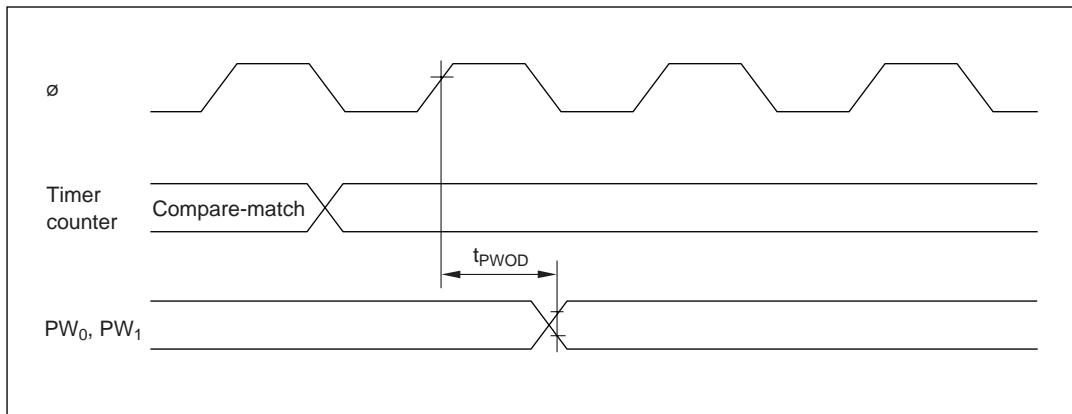


Figure 22-15 PWM Timer Output Timing

## 22.3.6 Serial Communication Interface Timing

### (1) SCI Input/Output Timing

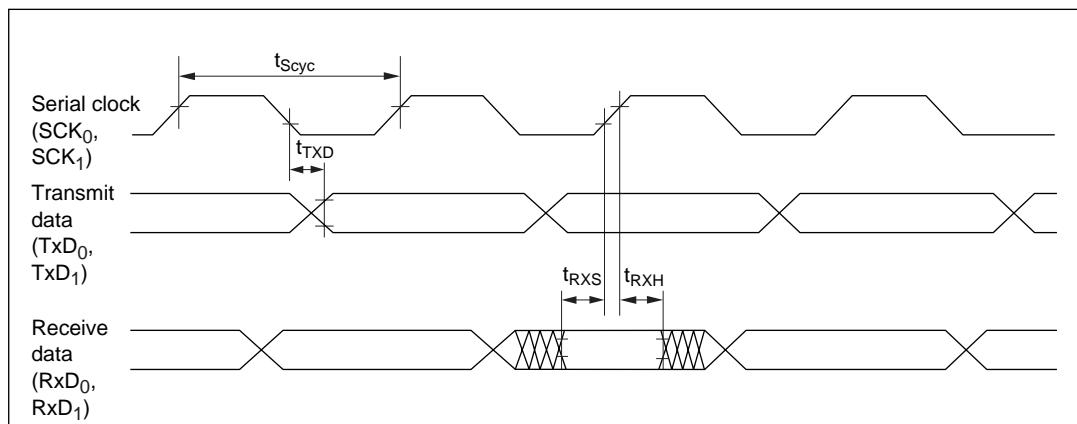


Figure 22-16 SCI Input/Output Timing (Synchronous Mode)

### (2) SCI Input Clock Timing

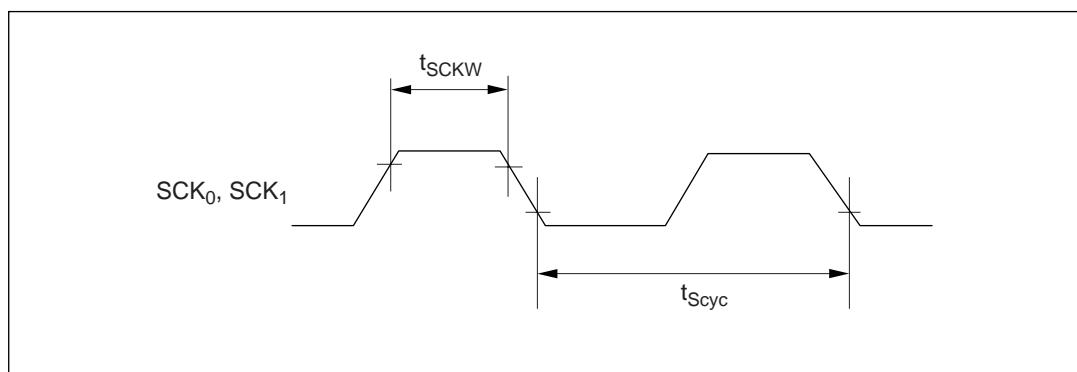
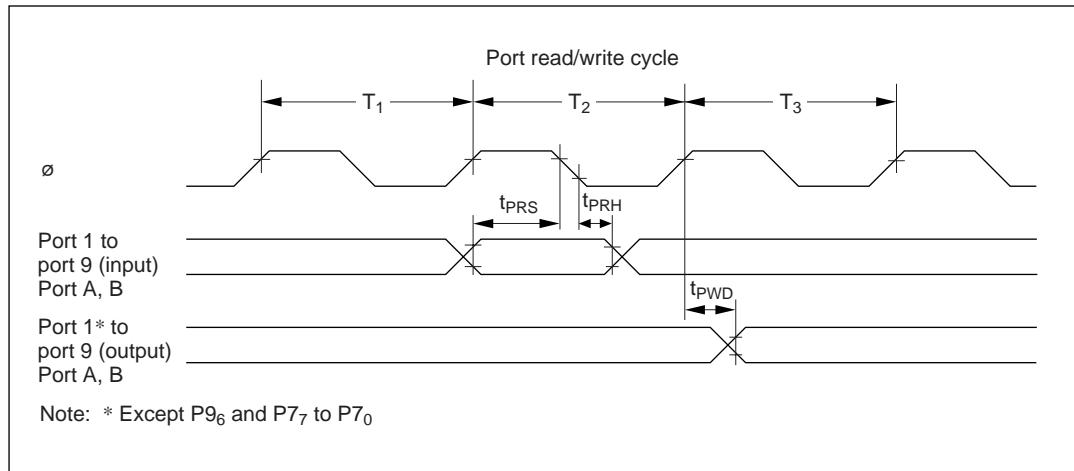


Figure 22-17 SCI Input Clock Timing

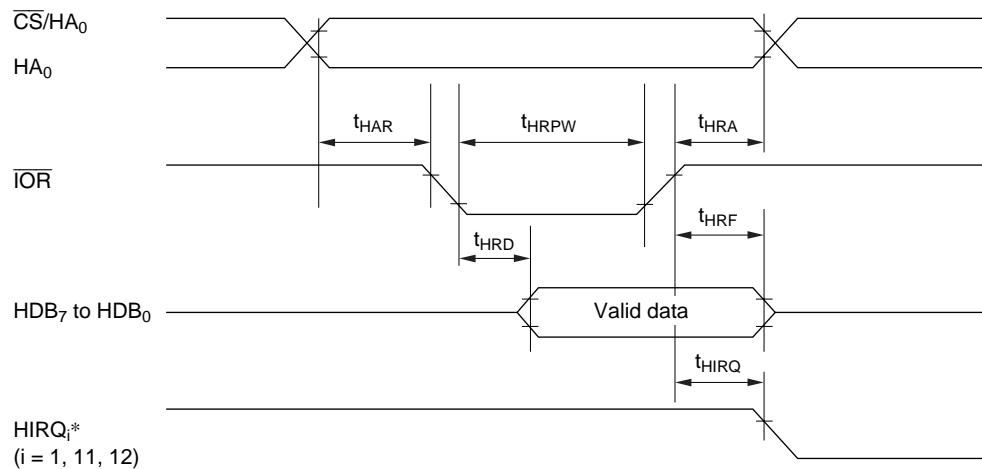
### 22.3.7 I/O Port Timing



**Figure 22-18 I/O Port Input/Output Timing**

## 22.3.8 Host Interface Timing

### (1) Host Interface Read Timing



Note: \* Rising edge timing is the same as in port 4 output timing. Refer to figure 22-18.

Figure 22-19 Host Interface Read Timing

### (2) Host Interface Write Timing

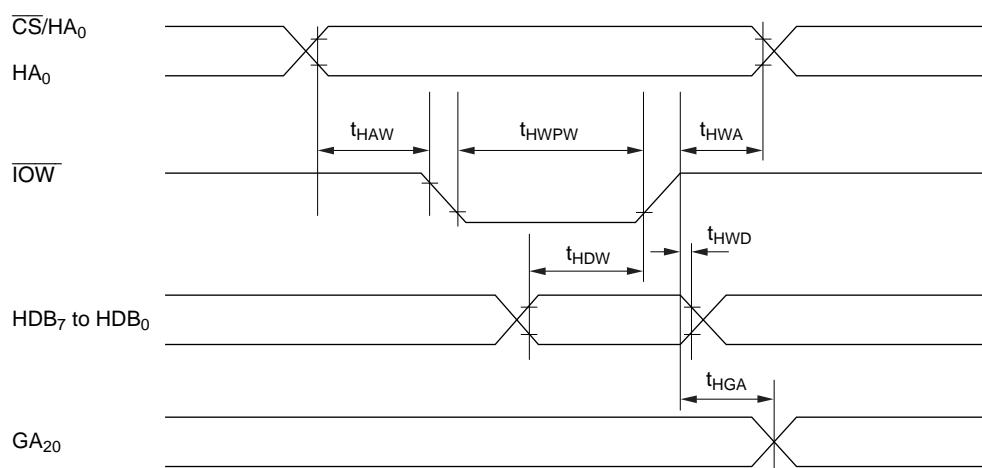
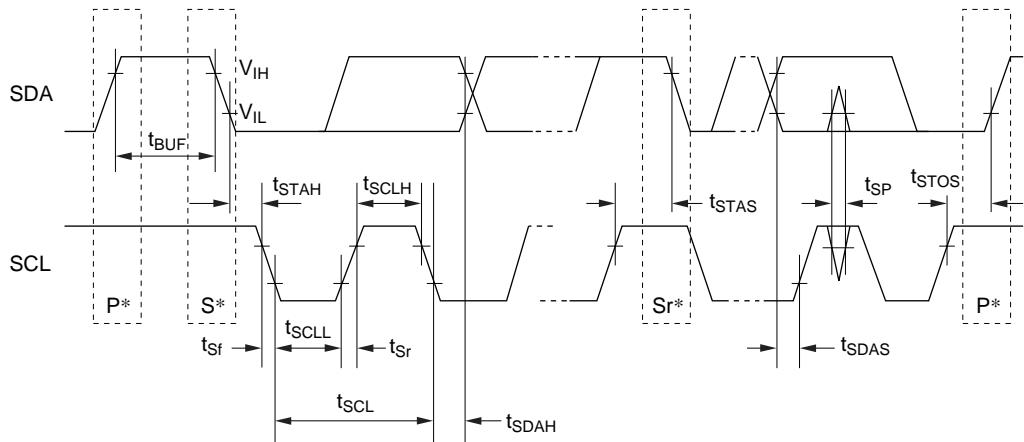


Figure 22-20 Host Interface Write Timing

### 22.3.9 I<sup>2</sup>C Bus Timing (Option)



Note: \* Conditions S, P, and Sr are defined as follows:

- S: Start condition
- P: Stop condition
- Sr: Resend "start" condition

Figure 22-21 I<sup>2</sup>C Bus Interface I/O Timing

### 22.3.10 Reset Output Timing

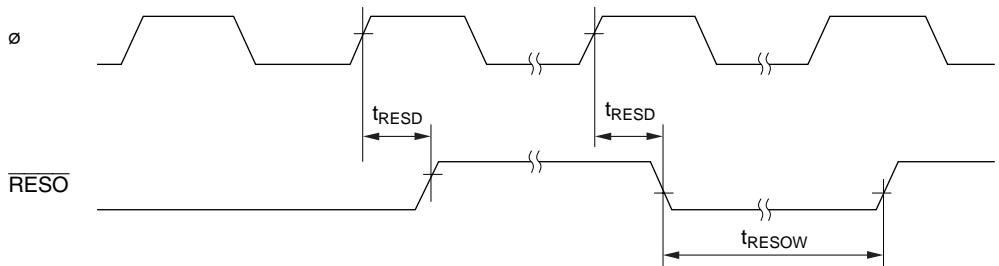
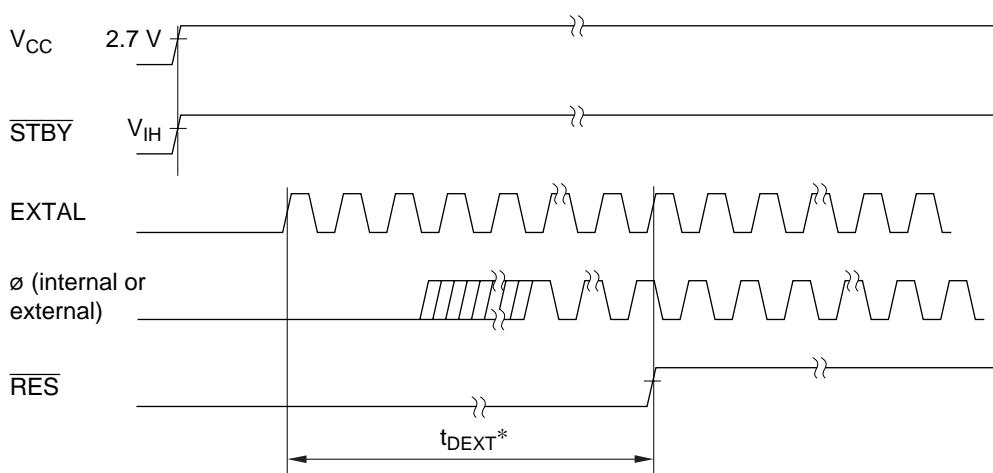


Figure 22-22 Reset Output Timing

### 22.3.11 External Clock Output Timing



Note: \*  $t_{DEXT}$  includes a 10  $t_{cyc}$   $\overline{RES}$  pulse width ( $t_{RESW}$ ).

Figure 22-23 External Clock Output Stabilization Delay Time

# Appendix A CPU Instruction Set

## A.1 Instruction Set List

### Operation Notation

Rd8/16	General register (destination) (8 or 16 bits)
Rs8/16	General register (source) (8 or 16 bits)
Rn8/16	General register (8 or 16 bits)
CCR	Condition code register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#xx:3/8/16	Immediate data (3, 8, or 16 bits)
d:8/16	Displacement (8 or 16 bits)
@aa:8/16	Absolute address (8 or 16 bits)
+	Addition
-	Subtraction
×	Multiplication
÷	Division
^	Logical AND
∨	Logical OR
⊕	Exclusive logical OR
→	Move
—	NOT (logical complement)

### Condition Code Notation

↑	Modified according to the instruction result
*	Undetermined (unpredictable)
0	Always cleared to 0
—	Not affected by the instruction result

**Table A-1 Instruction Set**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length							Condition Code						No. of States		
			#xx: 8/16	Rn	@Rn	(d:16, Rn)	(-Rn)@Rn+	@aa: 8/16	(d:8, PC)	@aa	Implied	I	H	N	Z	V	C	
			I	H	N	Z	V	C										
MOV.B #xx:8, Rd	B	#xx:8 → Rd8	2									—	—	↑	↑	0	—	2
MOV.B Rs, Rd	B	Rs8 → Rd8		2								—	—	↑	↑	0	—	2
MOV.B @Rs, Rd	B	@Rs16 → Rd8			2							—	—	↑	↑	0	—	4
MOV.B @(d:16, Rs), Rd	B	@(d:16, Rs16) → Rd8				4						—	—	↑	↑	0	—	6
MOV.B @Rs+, Rd	B	@Rs16 → Rd8 Rs16+1 → Rs16					2					—	—	↑	↑	0	—	6
MOV.B @aa:8, Rd	B	@aa:8 → Rd8					2					—	—	↑	↑	0	—	4
MOV.B @aa:16, Rd	B	@aa:16 → Rd8					4					—	—	↑	↑	0	—	6
MOV.B Rs, @Rd	B	Rs8 → @Rd16			2							—	—	↑	↑	0	—	4
MOV.B Rs, @(d:16, Rd)	B	Rs8 → @(d:16, Rd16)				4						—	—	↑	↑	0	—	6
MOV.B Rs, @-Rd	B	Rd16-1 → Rd16 Rs8 → @Rd16					2					—	—	↑	↑	0	—	6
MOV.B Rs, @aa:8	B	Rs8 → @aa:8						2				—	—	↑	↑	0	—	4
MOV.B Rs, @aa:16	B	Rs8 → @aa:16						4				—	—	↑	↑	0	—	6
MOV.W #xx:16, Rd	W	#xx:16 → Rd16	4									—	—	↑	↑	0	—	4
MOV.W Rs, Rd	W	Rs16 → Rd16		2								—	—	↑	↑	0	—	2
MOV.W @Rs, Rd	W	@Rs16 → Rd16			2							—	—	↑	↑	0	—	4
MOV.W @(d:16, Rs), Rd	W	@(d:16, Rs16) → Rd16				4						—	—	↑	↑	0	—	6
MOV.W @Rs+, Rd	W	@Rs16 → Rd16 Rs16+2 → Rs16					2					—	—	↑	↑	0	—	6
MOV.W @aa:16, Rd	W	@aa:16 → Rd16						4				—	—	↑	↑	0	—	6
MOV.W Rs, @Rd	W	Rs16 → @Rd16			2							—	—	↑	↑	0	—	4
MOV.W Rs, @(d:16, Rd)	W	Rs16 → @(d:16, Rd16)				4						—	—	↑	↑	0	—	6
MOV.W Rs, @-Rd	W	Rd16-2 → Rd16 Rs16 → @Rd16						2				—	—	↑	↑	0	—	6
MOV.W Rs, @aa:16	W	Rs16 → @aa:16						4				—	—	↑	↑	0	—	6
POP Rd	W	@SP → Rd16 SP+2 → SP						2				—	—	↑	↑	0	—	6
PUSH Rs	W	SP-2 → SP Rs16 → @SP						2				—	—	↑	↑	0	—	6

**Table A-1 Instruction Set (cont)**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length							Condition Code						No. of States	
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@-Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@ @aa	Implied	I	H	N	Z	V	C
MOVFPE @aa:16, Rd	B	Not supported															
MOVTPE Rs, @aa:16	B	Not supported															
EEPMOV	—	if R4L≠0 then Repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4L-1 → R4L Until R4L=0 else next									4	—	—	—	—	—	(4)
ADD.B #xx:8, Rd	B	Rd8+#xx:8 → Rd8	2								—	↑	↓	↓	↑	↑	2
ADD.B Rs, Rd	B	Rd8+Rs8 → Rd8	2								—	↑	↓	↓	↑	↑	2
ADD.W Rs, Rd	W	Rd16+Rs16 → Rd16	2								—	(1)	↑	↓	↓	↑	2
ADDX.B #xx:8, Rd	B	Rd8+#xx:8 +C → Rd8	2								—	↑	↓	(2)	↑	↑	2
ADDX.B Rs, Rd	B	Rd8+Rs8 +C → Rd8	2								—	↑	↓	(2)	↑	↑	2
ADDS.W #1, Rd	W	Rd16+1 → Rd16	2								—	—	—	—	—	—	2
ADDS.W #2, Rd	W	Rd16+2 → Rd16	2								—	—	—	—	—	—	2
INC.B Rd	B	Rd8+1 → Rd8	2								—	—	↑	↓	↑	↑	2
DAA.B Rd	B	Rd8 decimal adjust → Rd8	2								—	*	↑	↓	*	(3)	2
SUB.B Rs, Rd	B	Rd8-Rs8 → Rd8	2								—	↑	↓	↓	↑	↑	2
SUB.W Rs, Rd	W	Rd16-Rs16 → Rd16	2								—	(1)	↑	↓	↑	↑	2
SUBX.B #xx:8, Rd	B	Rd8-#xx:8 -C → Rd8	2								—	↑	↓	(2)	↑	↑	2
SUBX.B Rs, Rd	B	Rd8-Rs8 -C → Rd8	2								—	↑	↓	(2)	↑	↑	2
SUBS.W #1, Rd	W	Rd16-1 → Rd16	2								—	—	—	—	—	—	2
SUBS.W #2, Rd	W	Rd16-2 → Rd16	2								—	—	—	—	—	—	2
DEC.B Rd	B	Rd8-1 → Rd8	2								—	—	↑	↓	↑	↑	2
DAS.B Rd	B	Rd8 decimal adjust → Rd8	2								—	*	↑	↓	*	—	2
NEG.B Rd	B	0-Rd8 → Rd8	2								—	↑	↓	↓	↑	↑	2
CMP.B #xx:8, Rd	B	Rd8-#xx:8	2								—	↑	↓	↓	↑	↑	2
CMP.B Rs, Rd	B	Rd8-Rs8	2								—	↑	↓	↑	↑	↑	2
CMP.W Rs, Rd	W	Rd16-Rs16	2								—	(1)	↑	↓	↓	↑	2

**Table A-1 Instruction Set (cont)**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length							Condition Code						No. of States		
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@-Rn@Rn+	@aa: 8/16	@(d:8, PC)	@aa	Implied	I	H	N	Z	V	C	
MULXU.B Rs, Rd	B	Rd8 × Rs8 → Rd16	2									—	—	—	—	—	14	
DIVXU.B Rs, Rd	B	Rd16÷Rs8 → Rd16 (RdH: remainder, RdL: quotient)	2									—	—	(6)	(7)	—	14	
AND.B #xx:8, Rd	B	Rd8&#xx:8 → Rd8	2									—	—	↑	↑	0	—	2
AND.B Rs, Rd	B	Rd8&Rs8 → Rd8	2									—	—	↑	↑	0	—	2
OR.B #xx:8, Rd	B	Rd8∨#xx:8 → Rd8	2									—	—	↑	↑	0	—	2
OR.B Rs, Rd	B	Rd8∨Rs8 → Rd8	2									—	—	↑	↑	0	—	2
XOR.B #xx:8, Rd	B	Rd8⊕#xx:8 → Rd8	2									—	—	↑	↑	0	—	2
XOR.B Rs, Rd	B	Rd8⊕Rs8 → Rd8	2									—	—	↑	↑	0	—	2
NOT.B Rd	B	$\overline{Rd8} \rightarrow Rd8$	2									—	—	↑	↑	0	—	2
SHAL.B Rd	B		2									—	—	↑	↑	↑	↑	2
SHAR.B Rd	B		2									—	—	↑	↑	0	↑	2
SHLL.B Rd	B		2									—	—	↑	↑	0	↑	2
SHLR.B Rd	B		2									—	—	0	↑	0	↑	2
ROTXL.B Rd	B		2									—	—	↑	↑	0	↑	2
ROTXR.B Rd	B		2									—	—	↑	↑	0	↑	2

**Table A-1 Instruction Set (cont)**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length							Condition Code						No. of States		
			#xx: 8/16	Rn	@Rn	(d:16, Rn)	@-Rn/@Rn+	@aa: 8/16	(d:8, PC)	@ @aa	Implied	I	H	N	Z	V	C	
ROTL.B Rd	B		2									—	—	↑	↑	0	↑	2
ROTR.B Rd	B		2									—	—	↑	↑	0	↑	2
BSET #xx:3, Rd	B	(#xx:3 of Rd8) ← 1	2									—	—	—	—	—	—	2
BSET #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 1		4								—	—	—	—	—	—	8
BSET #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 1			4							—	—	—	—	—	—	8
BSET Rn, Rd	B	(Rn8 of Rd8) ← 1	2									—	—	—	—	—	—	2
BSET Rn, @Rd	B	(Rn8 of @Rd16) ← 1		4								—	—	—	—	—	—	8
BSET Rn, @aa:8	B	(Rn8 of @aa:8) ← 1			4							—	—	—	—	—	—	8
BCLR #xx:3, Rd	B	(#xx:3 of Rd8) ← 0	2									—	—	—	—	—	—	2
BCLR #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 0		4								—	—	—	—	—	—	8
BCLR #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 0			4							—	—	—	—	—	—	8
BCLR Rn, Rd	B	(Rn8 of Rd8) ← 0	2									—	—	—	—	—	—	2
BCLR Rn, @Rd	B	(Rn8 of @Rd16) ← 0		4								—	—	—	—	—	—	8
BCLR Rn, @aa:8	B	(Rn8 of @aa:8) ← 0			4							—	—	—	—	—	—	8
BNOT #xx:3, Rd	B	(#xx:3 of Rd8) ← (#xx:3 of Rd8)	2									—	—	—	—	—	—	2
BNOT #xx:3, @Rd	B	(#xx:3 of @Rd16) ← (#xx:3 of @Rd16)		4								—	—	—	—	—	—	8
BNOT #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← (#xx:3 of @aa:8)			4							—	—	—	—	—	—	8
BNOT Rn, Rd	B	(Rn8 of Rd8) ← (Rn8 of Rd8)	2									—	—	—	—	—	—	2
BNOT Rn, @Rd	B	(Rn8 of @Rd16) ← (Rn8 of @Rd16)		4								—	—	—	—	—	—	8
BNOT Rn, @aa:8	B	(Rn8 of @aa:8) ← (Rn8 of @aa:8)			4							—	—	—	—	—	—	8

**Table A-1 Instruction Set (cont)**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length							Condition Code						No. of States		
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@-Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@ @aa	Implied	I	H	N	Z	V	C	
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@-Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@ @aa	Implied	I	H	N	Z	V	C	
BTST #xx:3, Rd	B	(#xx:3 of Rd8) → Z	2									—	—	—	↑	—	—	2
BTST #xx:3, @Rd	B	(#xx:3 of @Rd16) → Z	4									—	—	—	↑	—	—	6
BTST #xx:3, @aa:8	B	(#xx:3 of @aa:8) → Z						4				—	—	—	↑	—	—	6
BTST Rn, Rd	B	(Rn8 of Rd8) → Z	2									—	—	—	↑	—	—	2
BTST Rn, @Rd	B	(Rn8 of @Rd16) → Z	4									—	—	—	↑	—	—	6
BTST Rn, @aa:8	B	(Rn8 of @aa:8) → Z							4			—	—	—	↑	—	—	6
BLD #xx:3, Rd	B	(#xx:3 of Rd8) → C	2									—	—	—	—	—	↑	2
BLD #xx:3, @Rd	B	(#xx:3 of @Rd16) → C	4									—	—	—	—	—	↑	6
BLD #xx:3, @aa:8	B	(#xx:3 of @aa:8) → C							4			—	—	—	—	—	↑	6
BILD #xx:3, Rd	B	(#xx:3 of Rd8) → C	2									—	—	—	—	—	↑	2
BILD #xx:3, @Rd	B	(#xx:3 of @Rd16) → C	4									—	—	—	—	—	↑	6
BILD #xx:3, @aa:8	B	(#xx:3 of @aa:8) → C							4			—	—	—	—	—	↑	6
BST #xx:3, Rd	B	C → (#xx:3 of Rd8)	2									—	—	—	—	—	—	2
BST #xx:3, @Rd	B	C → (#xx:3 of @Rd16)	4									—	—	—	—	—	—	8
BST #xx:3, @aa:8	B	C → (#xx:3 of @aa:8)							4			—	—	—	—	—	—	8
BIST #xx:3, Rd	B	Ā → (#xx:3 of Rd8)	2									—	—	—	—	—	—	2
BIST #xx:3, @Rd	B	Ā → (#xx:3 of @Rd16)	4									—	—	—	—	—	—	8
BIST #xx:3, @aa:8	B	Ā → (#xx:3 of @aa:8)							4			—	—	—	—	—	—	8
BAND #xx:3, Rd	B	C <sub>Λ</sub> (#xx:3 of Rd8) → C	2									—	—	—	—	—	↑	2
BAND #xx:3, @Rd	B	C <sub>Λ</sub> (#xx:3 of @Rd16) → C	4									—	—	—	—	—	↑	6
BAND #xx:3, @aa:8	B	C <sub>Λ</sub> (#xx:3 of @aa:8) → C							4			—	—	—	—	—	↑	6
BIAND #xx:3, Rd	B	C <sub>Λ</sub> (#xx:3 of Rd8) → C	2									—	—	—	—	—	↑	2
BIAND #xx:3, @Rd	B	C <sub>Λ</sub> (#xx:3 of @Rd16) → C	4									—	—	—	—	—	↑	6
BIAND #xx:3, @aa:8	B	C <sub>Λ</sub> (#xx:3 of @aa:8) → C							4			—	—	—	—	—	↑	6
BOR #xx:3, Rd	B	C <sub>∨</sub> (#xx:3 of Rd8) → C	2									—	—	—	—	—	↑	2
BOR #xx:3, @Rd	B	C <sub>∨</sub> (#xx:3 of @Rd16) → C	4									—	—	—	—	—	↑	6
BOR #xx:3, @aa:8	B	C <sub>∨</sub> (#xx:3 of @aa:8) → C							4			—	—	—	—	—	↑	6
BIOR #xx:3, Rd	B	C <sub>∨</sub> (#xx:3 of Rd8) → C	2									—	—	—	—	—	↑	2
BIOR #xx:3, @Rd	B	C <sub>∨</sub> (#xx:3 of @Rd16) → C	4									—	—	—	—	—	↑	6

**Table A-1 Instruction Set (cont)**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length						Condition Code						Condition Code						No. of States				
			#XX: 8/16		Rn		@Rn		@ (d:16, Rn)		@ -Rn/@Rn+		@ aa: 8/16		@ (d:8, PC)		@ @aa		Implied						
			Branching Condition																I	H	N	Z	V	C	
BIOR #xx:3, @aa:8	B	$C \vee (\#xx:3 \text{ of } @aa:8) \rightarrow C$																	—	—	—	—	—	—	6
BXOR #xx:3, Rd	B	$C \oplus (\#xx:3 \text{ of } Rd8) \rightarrow C$		2															—	—	—	—	—	—	2
BXOR #xx:3, @Rd	B	$C \oplus (\#xx:3 \text{ of } @Rd16) \rightarrow C$			4														—	—	—	—	—	—	6
BXOR #xx:3, @aa:8	B	$C \oplus (\#xx:3 \text{ of } @aa:8) \rightarrow C$																	—	—	—	—	—	—	6
BIXOR #xx:3, Rd	B	$C \oplus (\#xx:3 \text{ of } Rd8) \rightarrow C$		2															—	—	—	—	—	—	2
BIXOR #xx:3, @Rd	B	$C \oplus (\#xx:3 \text{ of } @Rd16) \rightarrow C$			4														—	—	—	—	—	—	6
BIXOR #xx:3, @aa:8	B	$C \oplus (\#xx:3 \text{ of } @aa:8) \rightarrow C$																	—	—	—	—	—	—	6
BRA d:8 (BT d:8)	—	$PC \leftarrow PC + d:8$																2	—	—	—	—	—	—	4
BRN d:8 (BF d:8)	—	$PC \leftarrow PC + 2$																2	—	—	—	—	—	—	4
BHI d:8	—	If condition is true then PC $\leftarrow$ PC+d:8 else next;																					4		
BLS d:8	—																								
BCC d:8 (BHS d:8)	—																								
BCS d:8 (BLO d:8)	—																								
BNE d:8	—																								
BEQ d:8	—																								
BVC d:8	—																								
BVS d:8	—																								
BPL d:8	—																								
BMI d:8	—																								
BGE d:8	—																								
BLT d:8	—																								
BGT d:8	—																								
BLE d:8	—																								
JMP @Rn	—	$PC \leftarrow Rn16$								2									—	—	—	—	—	—	4
JMP @aa:16	—	$PC \leftarrow aa:16$															4		—	—	—	—	—	—	6
JMP @@aa:8	—	$PC \leftarrow @aa:8$																2	—	—	—	—	—	—	8
BSR d:8	—	$SP-2 \rightarrow SP$ $PC \rightarrow @SP$ $PC \leftarrow PC + d:8$														2			—	—	—	—	—	—	6

**Table A-1 Instruction Set (cont)**

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (bytes)						Implied	Condition Code						No. of States	
			#xx: 8/16	Rn	@Rn	(d:16, Rn)	@-Rn/@Rn+	@aa: 8/16		(d:8, PC)	@@aa	I	H	N	Z	V	
JSR @Rn	—	SP-2 → SP PC → @SP PC ← Rn16		2								—	—	—	—	—	6
JSR @aa:16	—	SP-2 → SP PC → @SP PC ← aa:16					4					—	—	—	—	—	8
JSR @@aa:8	—	SP-2 → SP PC → @SP PC ← @aa:8							2			—	—	—	—	—	8
RTS	—	PC ← @SP SP+2 → SP								2		—	—	—	—	—	8
RTE	—	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP								2	↑	↑	↑	↑	↑	↑	10
SLEEP	—	Transit to sleep mode.								2	—	—	—	—	—	—	2
LDC #xx:8, CCR	B	#xx:8 → CCR	2								↑	↑	↑	↑	↑	↑	2
LDC Rs, CCR	B	Rs8 → CCR		2							↑	↑	↑	↑	↑	↑	2
STC CCR, Rd	B	CCR → Rd8		2							—	—	—	—	—	—	2
ANDC #xx:8, CCR	B	CCR $\wedge$ #xx:8 → CCR	2								↑	↑	↑	↑	↑	↑	2
ORC #xx:8, CCR	B	CCR $\vee$ #xx:8 → CCR	2								↑	↑	↑	↑	↑	↑	2
XORC #xx:8, CCR	B	CCR $\oplus$ #xx:8 → CCR	2								↑	↑	↑	↑	↑	↑	2
NOP	—	PC ← PC+2								2	—	—	—	—	—	—	2

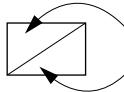
Notes: The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.

- (1) Set to 1 when there is a carry or borrow from bit 11; otherwise cleared to 0.
- (2) If the result is zero, the previous value of the flag is retained; otherwise the flag is cleared to 0.
- (3) Set to 1 if decimal adjustment produces a carry; otherwise cleared to 0.
- (4) The number of states required for execution is  $4n+8$  ( $n = \text{value of R4L}$ ).
- (5) These instructions are not supported by the H8/3437 Series.
- (6) Set to 1 if the divisor is negative; otherwise cleared to 0.
- (7) Set to 1 if the divisor is 0; otherwise cleared to 0.

## A.2 Operation Code Map

Table A-2 is a map of the operation codes contained in the first byte of the instruction code (bits 15 to 8 of the first instruction word).

Some pairs of instructions have identical first bytes. These instructions are differentiated by the first bit of the second byte (bit 7 of the first instruction word).



Instruction when first bit of byte 2 (bit 7 of first instruction word) is 0.

Instruction when first bit of byte 2 (bit 7 of first instruction word) is 1.

**Table A-2 Operation Code Map**

Low High \	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F									
0	NOP	SLEEP	STC	LDC	ORC	XORC	ANDC	LDC	ADD		INC	ADDS	MOV		ADDX	DAA									
1	SHLL SHAL	SHLR SHAR	ROTXL ROTL	ROTXR ROTR	OR	XOR	AND	NOT NEG	SUB		DEC	SUBS	CMP		SUBX	DAS									
2	MOV																								
3																									
4	BRA <sup>*2</sup>	BRN <sup>*2</sup>	BHI	BLS	BCC <sup>*2</sup>	BCS <sup>*2</sup>	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE									
5	MULXU	DIVXU			RTS	BSR	RTE			JMP															
6	BSET	BNOT	BCLR	BTS					BST	MOV <sup>*1</sup>															
7					BOR BIOR	BXOR BIXOR	BAND BIAND	BLD BILD			MOV			EEPMOV	Bit manipulation instructions										
8	ADD																								
9	ADDX																								
A	CMP																								
B	SUBX																								
C	OR																								
D	XOR																								
E	AND																								
F	MOV																								

Notes: 1. The MOVFPE and MOVTPE instructions are identical to MOV instructions in the first byte and first bit of the second byte (bits 15 to 7 of the instruction word).

The PUSH and POP instructions are identical in machine language to MOV instructions.

2. The BT, BF, BHS, and BLO instructions are identical in machine language to BRA, BRN, BCC, and BCS, respectively.

### A.3 Number of States Required for Execution

The tables below can be used to calculate the number of states required for instruction execution. Table A-3 indicates the number of states required for each cycle (instruction fetch, branch address read, stack operation, byte data access, word data access, internal operation). Table A-4 indicates the number of cycles of each type occurring in each instruction. The total number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

**Examples:** Mode 1 (on-chip ROM disabled), stack located in external memory, 1 wait state inserted in external memory access.

1. BSET #0, @FFC7

From table A-4:  $I = L = 2, J = K = M = N = 0$

From table A-3:  $S_I = 8, S_L = 3$

Number of states required for execution:  $2 \times 8 + 2 \times 3 = 22$

2. JSR @@30

From table A-4:  $I = 2, J = K = 1, L = M = N = 0$

From table A-3:  $S_I = S_J = S_K = 8$

Number of states required for execution:  $2 \times 8 + 1 \times 8 + 1 \times 8 = 32$

**Table A-3. Number of States Taken by Each Cycle in Instruction Execution**

Execution Status (Instruction Cycle)	Access location		
	On-Chip Memory	On-Chip Reg. Field	External Memory
Instruction fetch $S_I$	2	6	$6 + 2m$
Branch address read $S_J$			
Stack operation $S_K$			
Byte data access $S_L$		3	$3 + m$
Word data access $S_M$		6	$6 + 2m$
Internal operation $S_N$	1	1	1

Notes: m: Number of wait states inserted in access to external device.

**Table A-4 Number of Cycles in Each Instruction**

Instruction	Mnemonic	Instruction Fetch	Branch Addr. Read	Stack Operation	Byte Data Access	Word Data Access	Internal Operation
		I	J	K	L	M	N
ADD	ADD.B #xx:8, Rd	1					
	ADD.B Rs, Rd	1					
	ADD.W Rs, Rd	1					
ADDS	ADDS.W #1/2, Rd	1					
ADDX	ADDX.B #xx:8, Rd	1					
	ADDX.B Rs, Rd	1					
AND	AND.B #xx:8, Rd	1					
	AND.B Rs, Rd	1					
ANDC	ANDC #xx:8, CCR	1					
BAND	BAND #xx:3, Rd	1					
	BAND #xx:3, @Rd	2			1		
	BAND #xx:3, @aa:8	2			1		
Bcc	BRA d:8 (BT d:8)	2					
	BRN d:8 (BF d:8)	2					
	BHI d:8	2					
	BLS d:8	2					
	BCC d:8 (BHS d:8)	2					
	BCS d:8 (BLO d:8)	2					
	BNE d:8	2					
	BEQ d:8	2					
	BVC d:8	2					
	BVS d:8	2					
	BPL d:8	2					
	BMI d:8	2					
	BGE d:8	2					
	BLT d:8	2					
	BGT d:8	2					
	BLE d:8	2					
BCLR	BCLR #xx:3, Rd	1					
	BCLR #xx:3, @Rd	2			2		
	BCLR #xx:3, @aa:8	2			2		
	BCLR Rn, Rd	1					
	BCLR Rn, @Rd	2			2		
	BCLR Rn, @aa:8	2			2		

Note: All values left blank are zero.

**Table A-4 Number of Cycles in Each Instruction (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte Data	Word Data	Internal
		Fetch	Addr. Read	Operation	Access	Access	Operation
I	J	K	L	M	N		
BIAND	BIAND #xx:3, Rd	1					
	BIAND #xx:3, @Rd	2			1		
	BIAND #xx:3, @aa:8	2			1		
BILD	BILD #xx:3, Rd	1					
	BILD #xx:3, @Rd	2			1		
	BILD #xx:3, @aa:8	2			1		
BIOR	BIOR #xx:3, Rd	1					
	BIOR #xx:3, @Rd	2			1		
	BIOR #xx:3, @aa:8	2			1		
BIST	BIST #xx:3, Rd	1					
	BIST #xx:3, @Rd	2			2		
	BIST #xx:3, @aa:8	2			2		
BIXOR	BIXOR #xx:3, Rd	1					
	BIXOR #xx:3, @Rd	2			1		
	BIXOR #xx:3, @aa:8	2			1		
BLD	BLD #xx:3, Rd	1					
	BLD #xx:3, @Rd	2			1		
	BLD #xx:3, @aa:8	2			1		
BNOT	BNOT #xx:3, Rd	1					
	BNOT #xx:3, @Rd	2			2		
	BNOT #xx:3, @aa:8	2			2		
	BNOT Rn, Rd	1					
	BNOT Rn, @Rd	2			2		
	BNOT Rn, @aa:8	2			2		
BOR	BOR #xx:3, Rd	1					
	BOR #xx:3, @Rd	2			1		
	BOR #xx:3, @aa:8	2			1		
BSET	BSET #xx:3, Rd	1					
	BSET #xx:3, @Rd	2			2		
	BSET #xx:3, @aa:8	2			2		
	BSET Rn, Rd	1					
	BSET Rn, @Rd	2			2		
	BSET Rn, @aa:8	2			2		

Note: All values left blank are zero.

**Table A-4 Number of Cycles in Each Instruction (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte Data	Word Data	Internal
		Fetch	Addr. Read	Operation	Access	Access	Operation
I	J	K	L	M	N		
BSR	BSR d:8	2		1			
BST	BST #xx:3, Rd	1					
	BST #xx:3, @Rd	2			2		
	BST #xx:3, @aa:8	2			2		
BTST	BTST #xx:3, Rd	1					
	BTST #xx:3, @Rd	2			1		
	BTST #xx:3, @aa:8	2			1		
	BTST Rn, Rd	1					
	BTST Rn, @Rd	2			1		
	BTST Rn, @aa:8	2			1		
BXOR	BXOR #xx:3, Rd	1					
	BXOR #xx:3, @Rd	2			1		
	BXOR #xx:3, @aa:8	2			1		
CMP	CMP.B #xx:8, Rd	1					
	CMP.B Rs, Rd	1					
	CMP.W Rs, Rd	1					
DAA	DAA.B Rd	1					
DAS	DAS.B Rd	1					
DEC	DEC.B Rd	1					
DIVXU	DIVXU.B Rs, Rd	1				12	
EEPMOV	EEPMOV	2			2n+2*	1	
INC	INC.B Rd	1					
JMP	JMP @Rn	2					
	JMP @aa:16	2				2	
	JMP @@aa:8	2	1			2	
JSR	JSR @Rn	2		1			
	JSR @aa:16	2		1		2	
	JSR @@aa:8	2	1	1			
LDC	LDC #xx:8, CCR	1					
	LDC Rs, CCR	1					
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					
	MOV.B @Rs, Rd	1			1		
	MOV.B @(d:16,Rs), Rd	2			1		

Notes: All values left blank are zero.

\* n: Initial value in R4L. Source and destination are accessed n + 1 times each.

**Table A-4 Number of Cycles in Each Instruction (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte Data	Word Data	Internal
		Fetch	Addr. Read	Operation	Access	Access	Operation
I	J	K	L	M	N		
MOV	MOV.B @Rs+, Rd	1			1		2
	MOV.B @aa:8, Rd	1			1		
	MOV.B @aa:16, Rd	2			1		
	MOV.B Rs, @Rd	1			1		
	MOV.B Rs, @(d:16, Rd)	2			1		
	MOV.B Rs, @-Rd	1			1		2
	MOV.B Rs, @aa:8	1			1		
	MOV.B Rs, @aa:16	2			1		
	MOV.W #xx:16, Rd	2					
	MOV.W Rs, Rd	1					
	MOV.W @Rs, Rd	1			1		
	MOV.W @(d:16, Rs), Rd	2			1		
	MOV.W @Rs+, Rd	1			1		2
	MOV.W @aa:16, Rd	2			1		
	MOV.W Rs, @Rd	1			1		
	MOV.W Rs, @(d:16, Rd)	2			1		
	MOV.W Rs, @-Rd	1			1		2
	MOV.W Rs, @aa:16	2			1		
MOVFPE	MOVFPE @aa:16, Rd	Not supported					
MOVTPE	MOVTPE.Rs, @aa:16	Not supported					
MULXU	MULXU.Rs, Rd	1					12
NEG	NEG.B Rd	1					
NOP	NOP	1					
NOT	NOT.B Rd	1					
OR	OR.B #xx:8, Rd	1					
	OR.B Rs, Rd	1					
ORC	ORC #xx:8, CCR	1					
POP	POP Rd	1			1		2
PUSH	PUSH Rd	1			1		2
ROTL	ROTL.B Rd	1					
ROTR	ROTR.B Rd	1					
ROTXL	ROTXL.B Rd	1					
ROTXR	ROTXR.B Rd	1					
RTE	RTE	2		2			2
RTS	RTS	2		1			2

Note: All values left blank are zero.

**Table A-4 Number of Cycles in Each Instruction (cont)**

Instruction	Mnemonic	Instruction Fetch	Branch Addr. Read	Stack Operation	Byte Data Access	Word Data Access	Internal Operation
		I	J	K	L	M	N
SHAL	SHAL.B Rd	1					
SHAR	SHAR.B Rd	1					
SHLL	SHLL.B Rd	1					
SHLR	SHLR.B Rd	1					
SLEEP	SLEEP	1					
STC	STC CCR, Rd	1					
SUB	SUB.B Rs, Rd	1					
	SUB.W Rs, Rd	1					
SUBS	SUBS.W #1/2, Rd	1					
SUBX	SUBX.B #xx:8, Rd	1					
	SUBX.B Rs, Rd	1					
XOR	XOR.B #xx:8, Rd	1					
	XOR.B Rs, Rd	1					
XORC	XORC #xx:8, CCR	1					

Note: All values left blank are zero.

# Appendix B Internal I/O Register

## B.1 Addresses

Addr. (Last Byte)	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'80	FLMCR	V <sub>PP</sub>	—	—	—	EV	PV	E	P	Flash
H'81	—	—	—	—	—	—	—	—	—	memory or external addresses
H'82	EBr1	LB7/—*	LB6/—*	LB5/—*	LB4/—*	LB3	LB2	LB1	LB0	
H'83	EBr2	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0	(in expand- ed modes)
H'84										
H'85										
H'86										
H'87										
H'88	SMR	C/A	CHR	PE	O/E	STOP	MP	CKS1	CKS0	SCI1
H'89	BRR									
H'8A	SCR	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'8B	TDR									
H'8C	SSR	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT	
H'8D	RDR									
H'8E										
H'8F										
H'90	TIER	ICIAE	ICIBE	ICICE	ICIDE	OCIAE	OCIBE	OVIE	—	FRT
H'91	TCSR	ICFA	ICFB	ICFC	ICFD	OCFA	OCFB	OVF	CCLRA	
H'92	FRCH									
H'93	FRCL									
H'94	OCRAH									
	OCRBH									
H'95	OCRAL									
	OCRBL									
H'96	TCR	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0	
H'97	TOCR	—	—	—	OCRS	OEA	OEB	OLVLA	OLVLB	
H'98	ICRAH									
H'99	ICRAL									
H'9A	ICRBH									
H'9B	ICRBL									
H'9C	ICRCH									
H'9D	ICRCL									
H'9E	ICRDH									
H'9F	ICRDL									

Notes: \* H8/3437F / H8/3434F

(Continued on next page)

FRT: Free-running timer

SCI1: Serial communication interface 1

Addr. (Last Byte)	Register Name	Bit Names									Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'A0	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	PWM0	
H'A1	DTR										
H'A2	TCNT										
H'A3	—	—	—	—	—	—	—	—	—	—	
H'A4	TCR	OE	OS	—	—	—	CKS2	CKS1	CKS0	PWM1	
H'A5	DTR										
H'A6	TCNT										
H'A7	—	—	—	—	—	—	—	—	—	—	
H'A8	TCSR/ TCNT	OVF	WT/IT	TME	—	RST/NMI	CKS2	CKS1	CKS0	WDT	
H'A9	TCNT										
H'AA	PAODR	PA <sub>7</sub>	PA <sub>6</sub>	PA <sub>5</sub>	PA <sub>4</sub>	PA <sub>3</sub>	PA <sub>2</sub>	PA <sub>1</sub>	PA <sub>0</sub>	Port A	
H'AB	PAPIN/ PADDR	PA <sub>7</sub> / PA <sub>7</sub> DDR	PA <sub>6</sub> / PA <sub>6</sub> DDR	PA <sub>5</sub> / PA <sub>5</sub> DDR	PA <sub>4</sub> / PA <sub>4</sub> DDR	PA <sub>3</sub> / PA <sub>3</sub> DDR	PA <sub>2</sub> / PA <sub>2</sub> DDR	PA <sub>1</sub> / PA <sub>1</sub> DDR	PA <sub>0</sub> / PA <sub>0</sub> DDR		
H'AC	P1PCR	P1 <sub>7</sub> PCR	P1 <sub>6</sub> PCR	P1 <sub>5</sub> PCR	P1 <sub>4</sub> PCR	P1 <sub>3</sub> PCR	P1 <sub>2</sub> PCR	P1 <sub>1</sub> PCR	P1 <sub>0</sub> PCR	Port 1	
H'AD	P2PCR	P2 <sub>7</sub> PCR	P2 <sub>6</sub> PCR	P2 <sub>5</sub> PCR	P2 <sub>4</sub> PCR	P2 <sub>3</sub> PCR	P2 <sub>2</sub> PCR	P2 <sub>1</sub> PCR	P2 <sub>0</sub> PCR	Port 2	
H'AE	P3PCR	P3 <sub>7</sub> PCR	P3 <sub>6</sub> PCR	P3 <sub>5</sub> PCR	P3 <sub>4</sub> PCR	P3 <sub>3</sub> PCR	P3 <sub>2</sub> PCR	P3 <sub>1</sub> PCR	P3 <sub>0</sub> PCR	Port 3	
H'AF	—	—	—	—	—	—	—	—	—	—	
H'B0	P1DDR	P1 <sub>7</sub> DDR	P1 <sub>6</sub> DDR	P1 <sub>5</sub> DDR	P1 <sub>4</sub> DDR	P1 <sub>3</sub> DDR	P1 <sub>2</sub> DDR	P1 <sub>1</sub> DDR	P1 <sub>0</sub> DDR	Port 1	
H'B1	P2DDR	P2 <sub>7</sub> DDR	P2 <sub>6</sub> DDR	P2 <sub>5</sub> DDR	P2 <sub>4</sub> DDR	P2 <sub>3</sub> DDR	P2 <sub>2</sub> DDR	P2 <sub>1</sub> DDR	P2 <sub>0</sub> DDR	Port 2	
H'B2	P1DR	P1 <sub>7</sub>	P1 <sub>6</sub>	P1 <sub>5</sub>	P1 <sub>4</sub>	P1 <sub>3</sub>	P1 <sub>2</sub>	P1 <sub>1</sub>	P1 <sub>0</sub>	Port 1	
H'B3	P2DR	P2 <sub>7</sub>	P2 <sub>6</sub>	P2 <sub>5</sub>	P2 <sub>4</sub>	P2 <sub>3</sub>	P2 <sub>2</sub>	P2 <sub>1</sub>	P2 <sub>0</sub>	Port 2	
H'B4	P3DDR	P3 <sub>7</sub> DDR	P3 <sub>6</sub> DDR	P3 <sub>5</sub> DDR	P3 <sub>4</sub> DDR	P3 <sub>3</sub> DDR	P3 <sub>2</sub> DDR	P3 <sub>1</sub> DDR	P3 <sub>0</sub> DDR	Port 3	
H'B5	P4DDR	P4 <sub>7</sub> DDR	P4 <sub>6</sub> DDR	P4 <sub>5</sub> DDR	P4 <sub>4</sub> DDR	P4 <sub>3</sub> DDR	P4 <sub>2</sub> DDR	P4 <sub>1</sub> DDR	P4 <sub>0</sub> DDR	Port 4	
H'B6	P3DR	P3 <sub>7</sub>	P3 <sub>6</sub>	P3 <sub>5</sub>	P3 <sub>4</sub>	P3 <sub>3</sub>	P3 <sub>2</sub>	P3 <sub>1</sub>	P3 <sub>0</sub>	Port 3	
H'B7	P4DR	P4 <sub>7</sub>	P4 <sub>6</sub>	P4 <sub>5</sub>	P4 <sub>4</sub>	P4 <sub>3</sub>	P4 <sub>2</sub>	P4 <sub>1</sub>	P4 <sub>0</sub>	Port 4	
H'B8	P5DDR	—	—	—	—	—	P5 <sub>2</sub> DDR	P5 <sub>1</sub> DDR	P5 <sub>0</sub> DDR	Port 5	
H'B9	P6DDR	P6 <sub>7</sub> DDR	P6 <sub>6</sub> DDR	P6 <sub>5</sub> DDR	P6 <sub>4</sub> DDR	P6 <sub>3</sub> DDR	P6 <sub>2</sub> DDR	P6 <sub>1</sub> DDR	P6 <sub>0</sub> DDR	Port 6	
H'BA	P5DR	—	—	—	—	—	P5 <sub>2</sub>	P5 <sub>1</sub>	P5 <sub>0</sub>	Port 5	
H'BB	P6DR	P6 <sub>7</sub>	P6 <sub>6</sub>	P6 <sub>5</sub>	P6 <sub>4</sub>	P6 <sub>3</sub>	P6 <sub>2</sub>	P6 <sub>1</sub>	P6 <sub>0</sub>	Port 6	

Notes: PWM0: Pulse-width modulation timer channel 0

(Continued on next page)

PWM1: Pulse-width modulation timer channel 1

WDT: Watchdog timer

(Continued from preceding page)

Addr. (Last Byte)	Register Name	Bit Names									Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'BC	PBODR	PB <sub>7</sub>	PB <sub>6</sub>	PB <sub>5</sub>	PB <sub>4</sub>	PB <sub>3</sub>	PB <sub>2</sub>	PB <sub>1</sub>	PB <sub>0</sub>	Port B	
H'BD	P8DDR/ PBPIN	—/PB <sub>7</sub>	P8 <sub>6</sub> DDR/ PB <sub>6</sub>	P8 <sub>5</sub> DDR/ PB <sub>5</sub>	P8 <sub>4</sub> DDR/ PB <sub>4</sub>	P8 <sub>3</sub> DDR/ PB <sub>3</sub>	P8 <sub>2</sub> DDR/ PB <sub>2</sub>	P8 <sub>1</sub> DDR/ PB <sub>1</sub>	P8 <sub>0</sub> DDR/ PB <sub>0</sub>	Port 8/ Port B	
H'BE	P7PIN/ PBDDR	P7 <sub>7</sub> / PB <sub>7</sub> DDR	P7 <sub>6</sub> / PB <sub>6</sub> DDR	P7 <sub>5</sub> / PB <sub>5</sub> DDR	P7 <sub>4</sub> / PB <sub>4</sub> DDR	P7 <sub>3</sub> / PB <sub>3</sub> DDR	P7 <sub>2</sub> / PB <sub>2</sub> DDR	P7 <sub>1</sub> / PB <sub>1</sub> DDR	P7 <sub>0</sub> / PB <sub>0</sub> DDR	Port 7/ Port B	
H'BF	P8DR	—	P8 <sub>6</sub>	P8 <sub>5</sub>	P8 <sub>4</sub>	P8 <sub>3</sub>	P8 <sub>2</sub>	P8 <sub>1</sub>	P8 <sub>0</sub>	Port 8	
H'C0	P9DDR	P9 <sub>7</sub> DDR	P9 <sub>6</sub> DDR	P9 <sub>5</sub> DDR	P9 <sub>4</sub> DDR	P9 <sub>3</sub> DDR	P9 <sub>2</sub> DDR	P9 <sub>1</sub> DDR	P9 <sub>0</sub> DDR	Port 9	
H'C1	P9DR	P9 <sub>7</sub>	P9 <sub>6</sub>	P9 <sub>5</sub>	P9 <sub>4</sub>	P9 <sub>3</sub>	P9 <sub>2</sub>	P9 <sub>1</sub>	P9 <sub>0</sub>		
H'C2	WSCR	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0		
H'C3	STCR	IICS	IICD	IICX	IICE	STAC	MPE	ICKS1	ICKS0		
H'C4	SYSCR	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME		
H'C5	MDCR	—	—	—	—	—	—	MDS1	MDS0		
H'C6	ISCR	IRQ7SC	IRQ6SC	IRQ5SC	IRQ4SC	IRQ3SC	IRQ2SC	IRQ1SC	IRQ0SC		
H'C7	IER	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E		
H'C8	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR0	
H'C9	TCSR	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0		
H'CA	TCORA										
H'CB	TCORB										
H'CC	TCNT										
H'CD	—	—	—	—	—	—	—	—	—		
H'CE	—	—	—	—	—	—	—	—	—		
H'CF	—	—	—	—	—	—	—	—	—		
H'D0	TCR	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR1	
H'D1	TCSR	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0		
H'D2	TCORA										
H'D3	TCORB										
H'D4	TCNT										
H'D5	—	—	—	—	—	—	—	—	—		
H'D6	—	—	—	—	—	—	—	—	—		
H'D7	—	—	—	—	—	—	—	—	—		

Notes: TMR0: 8-bit timer channel 0

TMR1: 8-bit timer channel 1

(Continued on next page)

(Continued from preceding page)

Addr. (Last Byte)	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'D8	SMR	C/Ā	CHR	PE	O/Ē	STOP	MP	CKS1	CKS0	SCI0 and I <sup>2</sup> C
	ICCR	ICE	IEIC	MST	TRS	ACK	CKS2	CKS1	CKS0	
H'D9	BRR									
	ICSR	BBSY	IRIC	SCP	—	AL	AAS	ADZ	ACKB	
H'DA	SCR	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'DB	TDR									
H'DC	SSR	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT	
H'DD	RDR									
H'DE	—	—	—	—	—	—	—	—	—	
	ICDR	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	
H'DF	—	—	—	—	—	—	—	—	—	A/D
	ICMR/ SAR	MLS/ SVA6	WAIT/ SVA5	—/ SVA4	—/ SVA3	—/ SVA2	BC2/ SVA1	BC1/ SVA0	BC0/ FS	
H'E0	ADDRAH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
H'E1	ADDRAL	AD1	AD0	—	—	—	—	—	—	
H'E2	ADDRBH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
H'E3	ADDRBL	AD1	AD0	—	—	—	—	—	—	
H'E4	ADDRCH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
H'E5	ADDRCL	AD1	AD0	—	—	—	—	—	—	
H'E6	ADDRDH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
H'E7	ADDRDL	AD1	AD0	—	—	—	—	—	—	
H'E8	ADCSR	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0	
H'E9	ADCR	TRGE	—	—	—	—	—	—	—	
H'EA	—	—	—	—	—	—	—	—	—	
H'EB	—	—	—	—	—	—	—	—	—	
H'EC	—	—	—	—	—	—	—	—	—	
H'ED	—	—	—	—	—	—	—	—	—	
H'EE	—	—	—	—	—	—	—	—	—	
H'EF	—	—	—	—	—	—	—	—	—	

Notes: A/D: Analog-to-digital converter

(Continued on next page)

SCI0: Serial communication interface 0

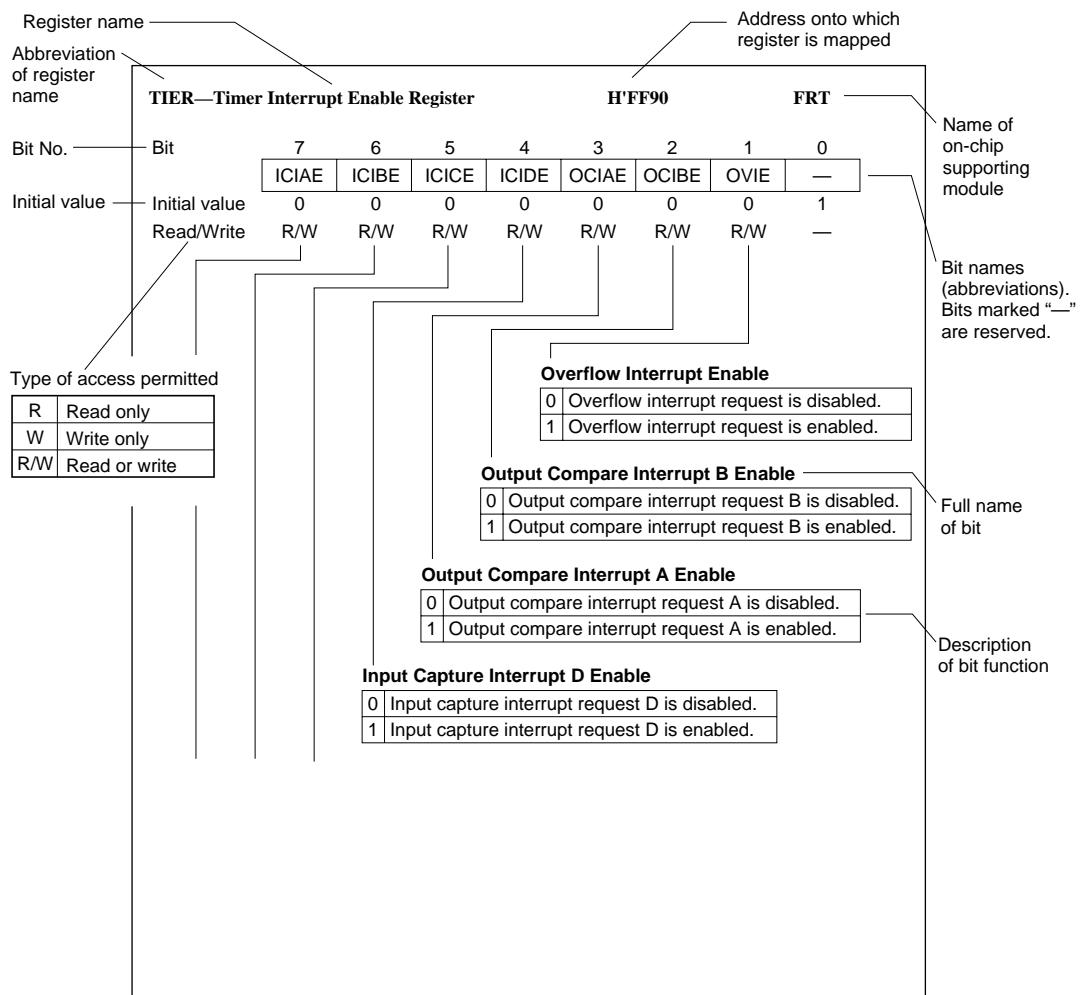
I<sup>2</sup>C: I<sup>2</sup>C bus interface

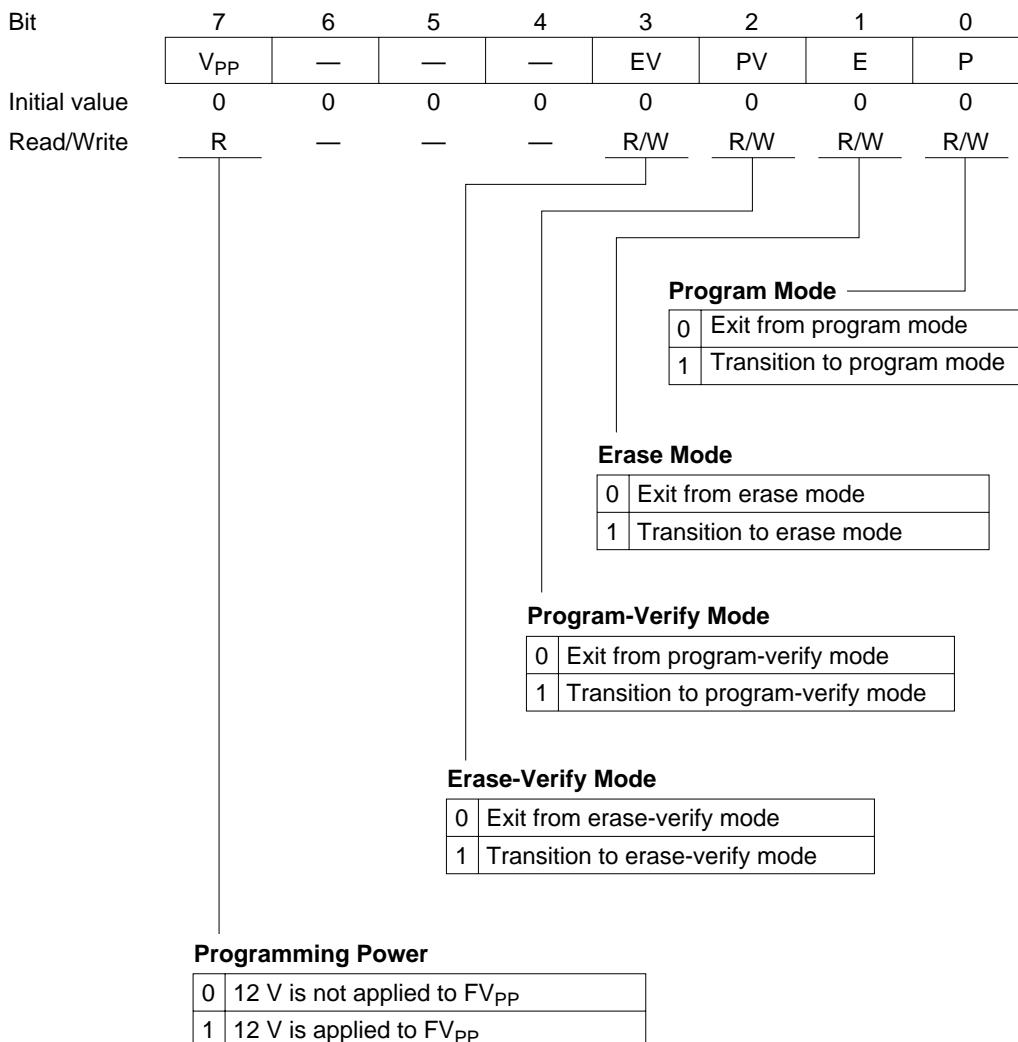
(Continued from preceding page)

Addr. (Last Byte)	Register Name	Bit Names									Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'F0	HICR	—	—	—	—	—	IBFIE2	IBFIE1	FGA20E	HIF	
H'F1	KMIMR	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0		
H'F2	KMPCR	KM <sub>7</sub> PCR	KM <sub>6</sub> PCR	KM <sub>5</sub> PCR	KM <sub>4</sub> PCR	KM <sub>3</sub> PCR	KM <sub>2</sub> PCR	KM <sub>1</sub> PCR	KM <sub>0</sub> PCR		
H'F3	KMIMRA	KMIMR15	KMIMR14	KMIMR13	KMIMR12	KMIMR11	KMIMR10	KMIMR9	KMIMR8		
H'F4	IDR1	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0	HIF1	
H'F5	ODR1	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0		
H'F6	STR1	DBU	DBU	DBU	DBU	C/D	DBU	IBF	OBF		
H'F7	—	—	—	—	—	—	—	—	—		
H'F8	DADR0										D/A
H'F9	DADR1										
H'FA	DACR	DAOE1	DAOE0	DAE	—	—	—	—	—		
H'FB	—	—	—	—	—	—	—	—	—		
H'FC	IDR2	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0	HIF2	
H'FD	ODR2	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0		
H'FE	STR2	DBU	DBU	DBU	DBU	C/D	DBU	IBF	OBF		
H'FF	—	—	—	—	—	—	—	—	—		

Note: HIF: Host interface

## B.2 Function





**EBR1—Erase Block Register 1****H'FF82****Flash memory****H8/3434 F-ZTAT**

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	LB3	LB2	LB1	LB0
Read/Write	1	1	1	1	0	0	0	0

**Large Block 3 to 0**

0	Corresponding block (LB3 to LB0) is not selected
1	Corresponding block (LB3 to LB0) is selected

**H8/3437 F-ZTAT**

Bit	7	6	5	4	3	2	1	0
Initial value	LB7	LB6	LB5	LB4	LB3	LB2	LB1	LB0
Read/Write	0	0	0	0	0	0	0	0

**Large Block 7 to 0**

0	Corresponding block (LB7 to LB0) is not selected (Initial value)
1	Corresponding block (LB7 to LB0) is selected

**EBR2—Erase Block Register 2****H'FF83****Flash memory**

Bit	7	6	5	4	3	2	1	0
Initial value	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Read/Write	0	0	0	0	0	0	0	0

**Small Block 7 to 0**

0	Corresponding block (SB7 to SB0) is not selected
1	Corresponding block (SB7 to SB0) is selected

Bit	7	6	5	4	3	2	1	0												
	C/A	CHR	PE	O/E	STOP	MP	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												
							<b>Clock Select</b>													
							<table border="1"> <tr><td>0</td><td>0</td><td>ø clock</td></tr> <tr><td>0</td><td>1</td><td>øP/4 clock</td></tr> <tr><td>1</td><td>0</td><td>øP/16 clock</td></tr> <tr><td>1</td><td>1</td><td>øP/64 clock</td></tr> </table>	0	0	ø clock	0	1	øP/4 clock	1	0	øP/16 clock	1	1	øP/64 clock	
0	0	ø clock																		
0	1	øP/4 clock																		
1	0	øP/16 clock																		
1	1	øP/64 clock																		
							<b>Multiprocessor Mode</b>													
							<table border="1"> <tr><td>0</td><td>Multiprocessor function disabled</td></tr> <tr><td>1</td><td>Multiprocessor format selected</td></tr> </table>	0	Multiprocessor function disabled	1	Multiprocessor format selected									
0	Multiprocessor function disabled																			
1	Multiprocessor format selected																			
							<b>Stop Bit Length</b>													
							<table border="1"> <tr><td>0</td><td>One stop bit</td></tr> <tr><td>1</td><td>Two stop bits</td></tr> </table>	0	One stop bit	1	Two stop bits									
0	One stop bit																			
1	Two stop bits																			
							<b>Parity Mode</b>													
							<table border="1"> <tr><td>0</td><td>Even parity</td></tr> <tr><td>1</td><td>Odd parity</td></tr> </table>	0	Even parity	1	Odd parity									
0	Even parity																			
1	Odd parity																			
							<b>Parity Enable</b>													
							<table border="1"> <tr><td>0</td><td>Transmit: No parity bit added. Receive: Parity bit not checked.</td></tr> <tr><td>1</td><td>Transmit: Parity bit added. Receive: Parity bit checked.</td></tr> </table>	0	Transmit: No parity bit added. Receive: Parity bit not checked.	1	Transmit: Parity bit added. Receive: Parity bit checked.									
0	Transmit: No parity bit added. Receive: Parity bit not checked.																			
1	Transmit: Parity bit added. Receive: Parity bit checked.																			
							<b>Character Length</b>													
							<table border="1"> <tr><td>0</td><td>8-bit data length</td></tr> <tr><td>1</td><td>7-bit data length</td></tr> </table>	0	8-bit data length	1	7-bit data length									
0	8-bit data length																			
1	7-bit data length																			
							<b>Communication Mode</b>													
							<table border="1"> <tr><td>0</td><td>Asynchronous</td></tr> <tr><td>1</td><td>Synchronous</td></tr> </table>	0	Asynchronous	1	Synchronous									
0	Asynchronous																			
1	Synchronous																			

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

Constant that determines the bit rate



**TDR—Transmit Data Register****H'FF8B****SCI1**

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

Transmit data

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

**Multiprocessor Bit Transfer**

0	Multiprocessor bit = 0 in transmit data.
1	Multiprocessor bit = 1 in transmit data.

**Multiprocessor Bit**

0	Multiprocessor bit = 0 in receive data.
1	Multiprocessor bit = 1 in receive data.

**Transmit End**

0	Cleared by reading TDRE = 1, then writing 0 in TDRE.
1	Set to 1 when TE = 0, or when TDRE = 1 at the end of character transmission.

**Parity Error**

0	Cleared by reading PER = 1, then writing 0 in PER.
1	Set when a parity error occurs (parity of receive data does not match parity selected by O/E bit in SMR).

**Framing Error**

0	Cleared by reading FER = 1, then writing 0 in FER.
1	Set when a framing error occurs (stop bit is 0).

**Overrun Error**

0	Cleared by reading ORER = 1, then writing 0 in ORER.
1	Set when an overrun error occurs (next data is completely received while RDRF bit is set to 1).

**Receive Data Register Full**

0	Cleared by reading RDRF = 1, then writing 0 in RDRF.
1	Set when one character is received normally and transferred from RSR to RDR.

**Transmit Data Register Empty**

0	Cleared by reading TDRE = 1, then writing 0 in TDRE.
1	Set when: <ol style="list-style-type: none"> <li>1. Data is transferred from TDR to TSR.</li> <li>2. TE is cleared while TDRE = 0.</li> </ol>

Note: \* Software can write a 0 in bits 7 to 3 to clear the flags, but cannot write a 1 in these bits.

**RDR—Receive Data Register****H'FF8D****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

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

**Overflow Interrupt Enable**

0	Overflow interrupt request is disabled.
1	Overflow interrupt request is enabled.

**Output Compare Interrupt B Enable**

0	Output compare interrupt request B is disabled.
1	Output compare interrupt request B is enabled.

**Output Compare Interrupt A Enable**

0	Output compare interrupt request A is disabled.
1	Output compare interrupt request A is enabled.

**Input Capture Interrupt D Enable**

0	Input capture interrupt request D is disabled.
1	Input capture interrupt request D is enabled.

**Input Capture Interrupt C Enable**

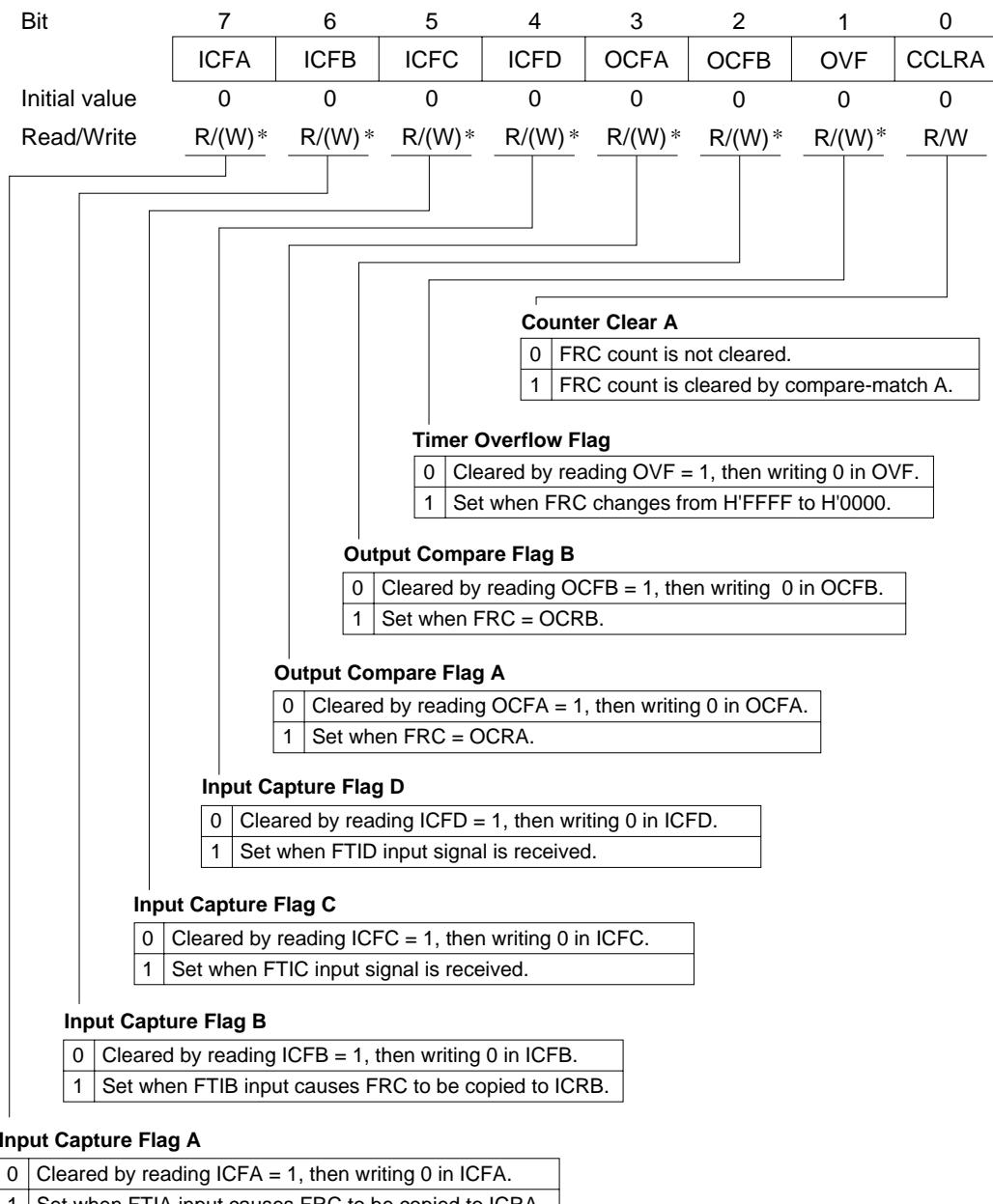
0	Input capture interrupt request C is disabled.
1	Input capture interrupt request C is enabled.

**Input Capture Interrupt B Enable**

0	Input capture interrupt request B is disabled.
1	Input capture interrupt request B is enabled.

**Input Capture Interrupt A Enable**

0	Input capture interrupt request A is disabled.
1	Input capture interrupt request A is enabled.



Note: \* Software can write a 0 in bits 7 to 1 to clear the flags, but cannot write a 1 in these bits.

**FRC (H and L)—Free-Running Counter****H'FF92, H'FF93****FRT**

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
	Count value															

**OCRA (H and L)—Output Compare Register A****H'FF94, H'FF95****FRT**

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
	Continually compared with FRC → OCFA is set when OCRA = FRC.															

**OCRB (H and L)—Output Compare Register B****H'FF94, H'FF95****FRT**

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
	Continually compared with FRC → OCFB is set when OCRB = FRC.															

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

**Clock Select**

0	0	Internal clock source: $\phi_P/2$
0	1	Internal clock source: $\phi_P/8$
1	0	Internal clock source: $\phi_P/32$
1	1	External clock source: counted on rising edge

**Buffer Enable B**

0	ICRD is used for input capture D.
1	ICRD is buffer register for input capture B.

**Buffer Enable A**

0	ICRC is used for input capture C.
1	ICRC is buffer register for input capture A.

**Input Edge Select D**

0	Falling edge of FTID is valid.
1	Rising edge of FTID is valid.

**Input Edge Select C**

0	Falling edge of FTIC is valid.
1	Rising edge of FTIC is valid.

**Input Edge Select B**

0	Falling edge of FTIB is valid.
1	Rising edge of FTIB is valid.

**Input Edge Select A**

0	Falling edge of FTIA is valid.
1	Rising edge of FTIA is valid.

**TOCR—Timer Output Compare Control Register****H'FF97****FRT**

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

**Output Level B**

0	Compare-match B causes 0 output.
1	Compare-match B causes 1 output.

**Output Level A**

0	Compare-match A causes 0 output.
1	Compare-match A causes 1 output.

**Output Enable B**

0	Output compare B output is disabled.
1	Output compare B output is enabled.

**Output Enable A**

0	Output compare A output is disabled.
1	Output compare A output is enabled.

**Output Compare Register Select**

0	The CPU can access OCRA.
1	The CPU can access OCRB.

**ICRA (H and L)—Input Capture Register A****H'FF98, H'FF99****FRT**

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

Contains FRC count captured on FTIA input.

**ICRB (H and L)—Input Capture Register B****H'FF9A, H'FF9B****FRT**

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

Contains FRC count captured on FTIB input.

**ICRC (H and L)—Input Capture Register C****H'FF9C, H'FF9D****FRT**

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

Contains FRC count captured on FTIC input, or old ICRA value in buffer mode.

**ICRD (H and L)—Input Capture Register D****H'FF9E, H'FF9F****FRT**

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

Contains FRC count captured on FTID input, or old ICRB value in buffer mode.

**TCR—Timer Control Register****H'FFA0****PWM0**

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

**Clock Select (Values when  $\phi_P = 10$  MHz)**

			Internal clock freq.	Resolution	PWM period	PWM frequency
0	0	0	$\phi_P/2$	200 ns	50 $\mu$ s	20 kHz
		1	$\phi_P/8$	800 ns	200 $\mu$ s	5 kHz
	1	0	$\phi_P/32$	3.2 $\mu$ s	800 $\mu$ s	1.25 kHz
		1	$\phi_P/128$	12.8 $\mu$ s	3.2 ms	312.5 Hz
1	0	0	$\phi_P/256$	25.6 $\mu$ s	6.4 ms	156.3 Hz
		1	$\phi_P/1024$	102.4 $\mu$ s	25.6 ms	39.1 Hz
	1	0	$\phi_P/2048$	204.8 $\mu$ s	51.2 ms	19.5 Hz
		1	$\phi_P/4096$	409.6 $\mu$ s	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'FFA1****PWM0**

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

**TCNT—Timer Counter****H'FFA2****PWM0**

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

Count value (runs from H'00 to H'F9, then repeats from H'00)

**TCR—Timer Control Register****H'FFA4****PWM1**

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

Note: Bit functions are the same as for PWM0.

**DTR—Duty Register****H'FFA5****PWM1**

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

Note: Bit functions are the same as for PWM0.

**TCNT—Timer Counter****H'FFA6****PWM1**

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

Note: Bit functions are the same as for PWM0.

Note: \* Only 0 can be written, to clear the flag.

**TCNT—Timer Counter****H'FFA9 (read),  
H'FFA8 (write)****WDT**

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

Count value

**P1PCR—Port 1 Input Pull-Up Control Register****H'FFAC****Port 1**

Bit	7	6	5	4	3	2	1	0
	P1 <sub>7</sub> PCR	P1 <sub>6</sub> PCR	P1 <sub>5</sub> PCR	P1 <sub>4</sub> PCR	P1 <sub>3</sub> PCR	P1 <sub>2</sub> PCR	P1 <sub>1</sub> PCR	P1 <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0

Port 1 Input Pull-Up Control

0	Input pull-up transistor is off.
1	Input pull-up transistor is on.

**P2PCR—Port 2 Input Pull-Up Control Register****H'FFAD****Port 2**

Bit	7	6	5	4	3	2	1	0
	P2 <sub>7</sub> PCR	P2 <sub>6</sub> PCR	P2 <sub>5</sub> PCR	P2 <sub>4</sub> PCR	P2 <sub>3</sub> PCR	P2 <sub>2</sub> PCR	P2 <sub>1</sub> PCR	P2 <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0

Port 2 Input Pull-Up Control

0	Input pull-up transistor is off.
1	Input pull-up transistor is on.

**P3PCR—Port 3 Input Pull-Up Control Register****H'FFAE****Port 3**

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

**Port 3 Input Pull-Up Control**

0	Input pull-up transistor is off.
1	Input pull-up transistor is on.

**P1DDR—Port 1 Data Direction Register****H'FFB0****Port 1**

Bit	7	6	5	4	3	2	1	0
Mode 1	P1 <sub>7</sub> DDR	P1 <sub>6</sub> DDR	P1 <sub>5</sub> DDR	P1 <sub>4</sub> DDR	P1 <sub>3</sub> DDR	P1 <sub>2</sub> DDR	P1 <sub>1</sub> DDR	P1 <sub>0</sub> DDR
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port 1 Input/Output Control**

0	Input port
1	Output port

**P1DR—Port 1 Data Register****H'FFB2****Port 1**

Bit	7	6	5	4	3	2	1	0
Initial value	P1 <sub>7</sub>	P1 <sub>6</sub>	P1 <sub>5</sub>	P1 <sub>4</sub>	P1 <sub>3</sub>	P1 <sub>2</sub>	P1 <sub>1</sub>	P1 <sub>0</sub>
Read/Write	0	0	0	0	0	0	0	0

**P2DDR—Port 2 Data Direction Register****H'FFB1****Port 2**

Bit	7	6	5	4	3	2	1	0
	P2 <sub>7</sub> DDR	P2 <sub>6</sub> DDR	P2 <sub>5</sub> DDR	P2 <sub>4</sub> DDR	P2 <sub>3</sub> DDR	P2 <sub>2</sub> DDR	P2 <sub>1</sub> DDR	P2 <sub>0</sub> DDR
Mode 1								
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	—	—	—	—	—
Modes 2 and 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port 2 Input/Output Control**

0	Input port
1	Output port

**P2DR—Port 2 Data Register****H'FFB3****Port 2**

Bit	7	6	5	4	3	2	1	0
	P2 <sub>7</sub>	P2 <sub>6</sub>	P2 <sub>5</sub>	P2 <sub>4</sub>	P2 <sub>3</sub>	P2 <sub>2</sub>	P2 <sub>1</sub>	P2 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

**P3DDR—Port 3 Data Direction Register****H'FFB4****Port 3**

Bit	7	6	5	4	3	2	1	0
	P3 <sub>7</sub> DDR	P3 <sub>6</sub> DDR	P3 <sub>5</sub> DDR	P3 <sub>4</sub> DDR	P3 <sub>3</sub> DDR	P3 <sub>2</sub> DDR	P3 <sub>1</sub> DDR	P3 <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port 3 Input/Output Control**

0	Input port
1	Output port

**P3DR—Port 3 Data Register****H'FFB6****Port 3**

Bit	7	6	5	4	3	2	1	0
	P3 <sub>7</sub>	P3 <sub>6</sub>	P3 <sub>5</sub>	P3 <sub>4</sub>	P3 <sub>3</sub>	P3 <sub>2</sub>	P3 <sub>1</sub>	P3 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

**P4DDR—Port 4 Data Direction Register****H'FFB5****Port 4**

Bit	7	6	5	4	3	2	1	0
	P4 <sub>7</sub> DDR	P4 <sub>6</sub> DDR	P4 <sub>5</sub> DDR	P4 <sub>4</sub> DDR	P4 <sub>3</sub> DDR	P4 <sub>2</sub> DDR	P4 <sub>1</sub> DDR	P4 <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port 4 Input/Output Control**

0	Input port
1	Output port

**P4DR—Port 4 Data Register****H'FFB7****Port 4**

Bit	7	6	5	4	3	2	1	0
	P4 <sub>7</sub>	P4 <sub>6</sub>	P4 <sub>5</sub>	P4 <sub>4</sub>	P4 <sub>3</sub>	P4 <sub>2</sub>	P4 <sub>1</sub>	P4 <sub>0</sub>
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

**P5DDR—Port 5 Data Direction Register****H'FFB8****Port 5**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	P5 <sub>2</sub> DDR	P5 <sub>1</sub> DDR	P5 <sub>0</sub> DDR
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	W	W	W

**Port 5 Input/Output Control**

0	Input port
1	Output port

**P5DR—Port 5 Data Register****H'FFBA****Port 5**

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	P5 <sub>2</sub>	P5 <sub>1</sub>	P5 <sub>0</sub>
Read/Write	1	1	1	1	1	0	0	0
	—	—	—	—	—	R/W	R/W	R/W

**P6DDR—Port 6 Data Direction Register****H'FFB9****Port 6**

Bit	7	6	5	4	3	2	1	0
Initial value	P6 <sub>7</sub> DDR	P6 <sub>6</sub> DDR	P6 <sub>5</sub> DDR	P6 <sub>4</sub> DDR	P6 <sub>3</sub> DDR	P6 <sub>2</sub> DDR	P6 <sub>1</sub> DDR	P6 <sub>0</sub> DDR
Read/Write	0	0	0	0	0	0	0	0

**Port 6 Input/Output Control**

0	Input port
1	Output port

**P6DR—Port 6 Data Register****H'FFBB****Port 6**

Bit	7	6	5	4	3	2	1	0
Initial value	P6 <sub>7</sub>	P6 <sub>6</sub>	P6 <sub>5</sub>	P6 <sub>4</sub>	P6 <sub>3</sub>	P6 <sub>2</sub>	P6 <sub>1</sub>	P6 <sub>0</sub>
Read/Write	0	0	0	0	0	0	0	0

**P7PIN—Port 7 Input Data Register****H'FFBE****Port 7**

Bit	7	6	5	4	3	2	1	0
Initial value	P7 <sub>7</sub>	P7 <sub>6</sub>	P7 <sub>5</sub>	P7 <sub>4</sub>	P7 <sub>3</sub>	P7 <sub>2</sub>	P7 <sub>1</sub>	P7 <sub>0</sub>
Read/Write	*	*	*	*	*	*	*	*

Note: \* Depends on the levels of pins P7<sub>7</sub> to P7<sub>0</sub>.

**P8DDR—Port 8 Data Direction Register****H'FFBD****Port 8**

Bit	7	6	5	4	3	2	1	0
	—	P8 <sub>6</sub> DDR	P8 <sub>5</sub> DDR	P8 <sub>4</sub> DDR	P8 <sub>3</sub> DDR	P8 <sub>2</sub> DDR	P8 <sub>1</sub> DDR	P8 <sub>0</sub> DDR
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

**Port 8 Input/Output Control**

0	Input port
1	Output port

**P8DR—Port 8 Data Register****H'FFBF****Port 8**

Bit	7	6	5	4	3	2	1	0
	—	P8 <sub>6</sub>	P8 <sub>5</sub>	P8 <sub>4</sub>	P8 <sub>3</sub>	P8 <sub>2</sub>	P8 <sub>1</sub>	P8 <sub>0</sub>
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W						

**P9DDR—Port 9 Data Direction Register****H'FFC0****Port 9**

Bit	7	6	5	4	3	2	1	0
	P9 <sub>7</sub> DDR	P9 <sub>6</sub> DDR	P9 <sub>5</sub> DDR	P9 <sub>4</sub> DDR	P9 <sub>3</sub> DDR	P9 <sub>2</sub> DDR	P9 <sub>1</sub> DDR	P9 <sub>0</sub> DDR
Modes 1 and 2								
Initial value	0	1	0	0	0	0	0	0
Read/Write	W	—	W	W	W	W	W	W
Mode 3								
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port 9 Input/Output Control**

0	Input port
1	Output port

**P9DR—Port 9 Data Register****H'FFC1****Port 9**

Bit	7	6	5	4	3	2	1	0
	P9 <sub>7</sub>	P9 <sub>6</sub>	P9 <sub>5</sub>	P9 <sub>4</sub>	P9 <sub>3</sub>	P9 <sub>2</sub>	P9 <sub>1</sub>	P9 <sub>0</sub>
Initial value	0	*	0	0	0	0	0	0
Read/Write	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* Depends on the level of pin P9<sub>6</sub>.

**PADDR—Port A Data Direction Register****H'AB****Port A**

Bit	7	6	5	4	3	2	1	0
	PA <sub>7</sub> DDR	PA <sub>6</sub> DDR	PA <sub>5</sub> DDR	PA <sub>4</sub> DDR	PA <sub>3</sub> DDR	PA <sub>2</sub> DDR	PA <sub>1</sub> DDR	PA <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port A Input/Output Control**

0	Input port
1	Output port

**PAPIN—Port A Input Data Register****H'AB****Port A**

Bit	7	6	5	4	3	2	1	0
	PA <sub>7</sub>	PA <sub>6</sub>	PA <sub>5</sub>	PA <sub>4</sub>	PA <sub>3</sub>	PA <sub>2</sub>	PA <sub>1</sub>	PA <sub>0</sub>
Initial value	—*	—*	—*	—*	—*	—*	—*	—*
Read/Write	R	R	R	R	R	R	R	R

Note: \* Depends on the levels of pins PA<sub>7</sub> to PA<sub>0</sub>.

**PAODR—Port A Output Data Register****H'AA****Port A**

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

**Port A Output Data/Input Pull-Up Control**

	During output	During input
0	0 output	Input pull-up transistor off
1	1 output	Input pull-up transistor on

**PBDDR—Port B Data Direction Register****H'BE****Port B**

Bit	7	6	5	4	3	2	1	0
	PB <sub>7</sub> DDR	PB <sub>6</sub> DDR	PB <sub>5</sub> DDR	PB <sub>4</sub> DDR	PB <sub>3</sub> DDR	PB <sub>2</sub> DDR	PB <sub>1</sub> DDR	PB <sub>0</sub> DDR
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

**Port B Input/Output Control**

0	Input port
1	Output port

**PBPIN—Port B Input Data Register****H'BD****Port B**

Bit	7	6	5	4	3	2	1	0
	PB <sub>7</sub>	PB <sub>6</sub>	PB <sub>5</sub>	PB <sub>4</sub>	PB <sub>3</sub>	PB <sub>2</sub>	PB <sub>1</sub>	PB <sub>0</sub>
Initial value	—*	—*	—*	—*	—*	—*	—*	—*
Read/Write	R	R	R	R	R	R	R	R

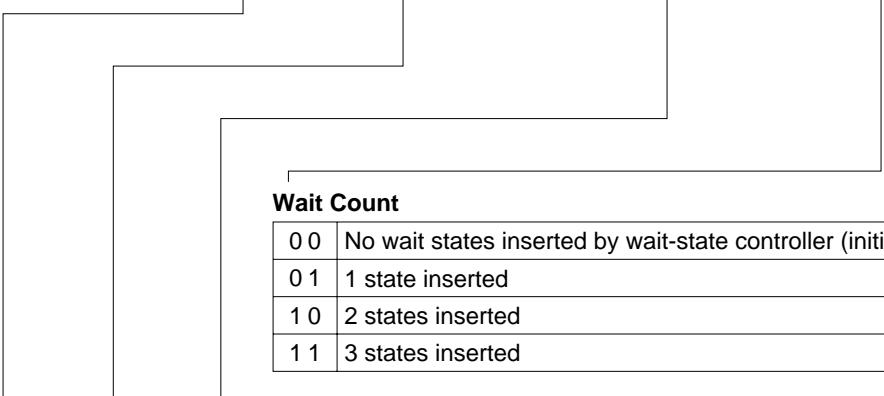
Note: \* Depends on the levels of pins PB<sub>7</sub> to PB<sub>0</sub>.

**PBODR—Port B Output Data Register****H'BC****Port B**

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

**Port B Output Data/Input Pull-Up Control**

	During output	During input
0	0 output	Input pull-up transistor off
1	1 output	Input pull-up transistor on

Bit	7	6	5	4	3	2	1	0								
Initial value	RAMS	RAM0	CKDBL	—	WMS1	WMS0	WC1	WC0								
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W								
																
<b>Wait Count</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>0 0</td><td>No wait states inserted by wait-state controller (initial value)</td></tr> <tr><td>0 1</td><td>1 state inserted</td></tr> <tr><td>1 0</td><td>2 states inserted</td></tr> <tr><td>1 1</td><td>3 states inserted</td></tr> </table>									0 0	No wait states inserted by wait-state controller (initial value)	0 1	1 state inserted	1 0	2 states inserted	1 1	3 states inserted
0 0	No wait states inserted by wait-state controller (initial value)															
0 1	1 state inserted															
1 0	2 states inserted															
1 1	3 states inserted															
<b>Wait Mode Select</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>0 0</td><td>Programmable wait mode</td></tr> <tr><td>0 1</td><td>No wait states inserted by wait-state controller</td></tr> <tr><td>1 0</td><td>Pin wait mode (initial value)</td></tr> <tr><td>1 1</td><td>Pin auto-wait mode</td></tr> </table>									0 0	Programmable wait mode	0 1	No wait states inserted by wait-state controller	1 0	Pin wait mode (initial value)	1 1	Pin auto-wait mode
0 0	Programmable wait mode															
0 1	No wait states inserted by wait-state controller															
1 0	Pin wait mode (initial value)															
1 1	Pin auto-wait mode															
<b>Clock Double</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>0</td><td>Supporting module clock frequency is not divided (<math>\phi_P = \phi</math>) (initial value)</td></tr> <tr><td>1</td><td>Supporting module clock frequency is divided by two (<math>\phi_P = \phi/2</math>)</td></tr> </table>									0	Supporting module clock frequency is not divided ( $\phi_P = \phi$ ) (initial value)	1	Supporting module clock frequency is divided by two ( $\phi_P = \phi/2$ )				
0	Supporting module clock frequency is not divided ( $\phi_P = \phi$ ) (initial value)															
1	Supporting module clock frequency is divided by two ( $\phi_P = \phi/2$ )															

### RAM Select and RAM Area Select

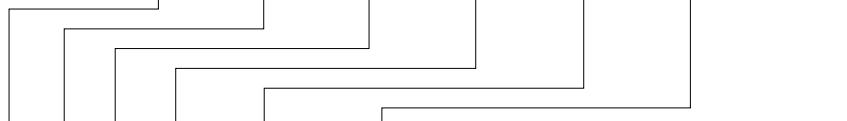
#### H8/3434 F-ZTAT

RAMS, RAM0	RAM Area	ROM Area
0 0	None	—
0 1	H'FC80 to H'FCFF	H'0080 to H'00FF
1 0	H'FC80 to H'FD7F	H'0080 to H'017F
1 1	H'FC00 to H'FC7F	H'0000 to H'007F

#### H8/3437 F-ZTAT

RAMS, RAM0	RAM Area	ROM Area
0 0	None	—
0 1	H'F880 to H'F8FF	H'0080 to H'00FF
1 0	H'F880 to H'F97F	H'0080 to H'017F
1 1	H'F800 to H'F87F	H'0000 to H'007F

Bit	7	6	5	4	3	2	1	0
	IICS	IICD	IICX	IICE	STAC	MPE	ICKS1	ICKS0
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

**Internal Clock Source Select**

See TCR under TMR0 and TMR1.

**Multiprocessor Enable**

0	Multiprocessor communication function is disabled.
1	Multiprocessor communication function is enabled.

**Slave Mode Control Input Switch**

0	$\overline{CS}_2$ and $\overline{IOW}$ are enabled
1	$ECS_2$ and $EIOW$ are enabled

**I<sup>2</sup>C Master Enable**

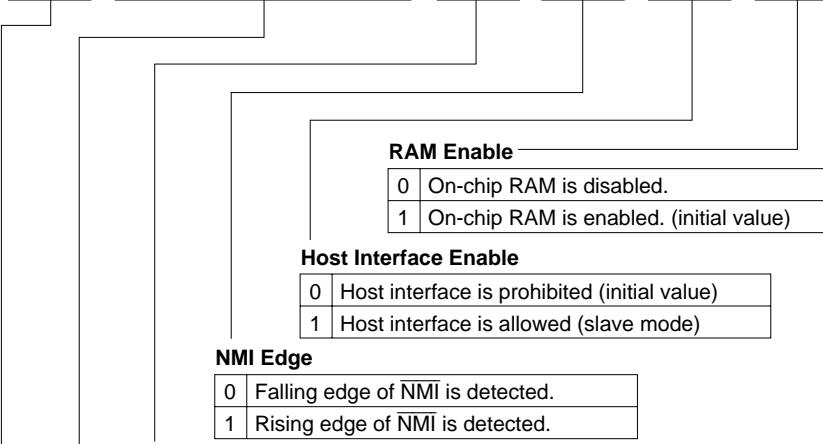
0	I <sup>2</sup> C bus interface data registers and control registers are disabled (initial value)
1	I <sup>2</sup> C bus interface data registers and control registers are enabled

**I<sup>2</sup>C Transfer Rate Select**

IICX	CKS2*	CKS1*	CKS0*	Clock	Transfer Rate				
					$\phi_p = 4$ MHz	$\phi_p = 5$ MHz	$\phi_p = 8$ MHz	$\phi_p = 10$ MHz	$\phi_p = 16$ MHz
0	0	0	0	$\phi_p/28$	143 kHz	179 kHz	286 kHz	357 kHz	571 kHz
			1	$\phi_p/40$	100 kHz	125 kHz	200 kHz	250 kHz	400 kHz
		1	0	$\phi_p/48$	83.3 kHz	104 kHz	167 kHz	208 kHz	333 kHz
			1	$\phi_p/64$	62.5 kHz	78.1 kHz	125 kHz	156 kHz	250 kHz
	1	0	0	$\phi_p/80$	50.0 kHz	62.5 kHz	100 kHz	125 kHz	200 kHz
			1	$\phi_p/100$	40.0 kHz	50.0 kHz	80.0 kHz	100 kHz	160 kHz
		1	0	$\phi_p/112$	35.7 kHz	44.6 kHz	71.4 kHz	89.3 kHz	143 kHz
			1	$\phi_p/128$	31.3 kHz	39.1 kHz	62.5 kHz	78.1 kHz	125 kHz
	1	0	0	$\phi_p/56$	71.4 kHz	89.3 kHz	143 kHz	179 kHz	286 kHz
			1	$\phi_p/80$	50.0 kHz	62.5 kHz	100 kHz	125 kHz	200 kHz
		1	0	$\phi_p/96$	41.7 kHz	52.1 kHz	83.3 kHz	104 kHz	167 kHz
			1	$\phi_p/128$	31.3 kHz	39.1 kHz	62.5 kHz	78.1 kHz	125 kHz
	1	0	0	$\phi_p/160$	25.0 kHz	31.3 kHz	50.0 kHz	62.5 kHz	100 kHz
			1	$\phi_p/200$	20.0 kHz	25.0 kHz	40.0 kHz	50.0 kHz	80.0 kHz
		1	0	$\phi_p/224$	17.9 kHz	22.3 kHz	35.7 kHz	44.6 kHz	71.4 kHz
			1	$\phi_p/256$	15.6 kHz	19.5 kHz	31.3 kHz	39.1 kHz	62.5 kHz

Notes:  $\phi_p = \phi$ .\* CKS2 to CKS0 are bits 2 to 0 of the I<sup>2</sup>C bus control register in the I<sup>2</sup>C bus interface.**I<sup>2</sup>C Extra Buffer Reserve****I<sup>2</sup>C Extra Buffer Select**

0	PA <sub>7</sub> to PA <sub>4</sub> are normal input/output pins
1	PA <sub>7</sub> to PA <sub>4</sub> are selected for bus drive

Bit	7	6	5	4	3	2	1	0																												
Initial value	SSBY	STS2	STS1	STS0	XRST	NMIEG	HIE	RAME																												
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W																												
																																				
<b>RAM Enable</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>On-chip RAM is disabled.</td> </tr> <tr> <td>1</td> <td>On-chip RAM is enabled. (initial value)</td> </tr> </table>									0	On-chip RAM is disabled.	1	On-chip RAM is enabled. (initial value)																								
0	On-chip RAM is disabled.																																			
1	On-chip RAM is enabled. (initial value)																																			
<b>Host Interface Enable</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>Host interface is prohibited (initial value)</td> </tr> <tr> <td>1</td> <td>Host interface is allowed (slave mode)</td> </tr> </table>									0	Host interface is prohibited (initial value)	1	Host interface is allowed (slave mode)																								
0	Host interface is prohibited (initial value)																																			
1	Host interface is allowed (slave mode)																																			
<b>NMI Edge</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>Falling edge of <math>\overline{NMI}</math> is detected.</td> </tr> <tr> <td>1</td> <td>Rising edge of <math>\overline{NMI}</math> is detected.</td> </tr> </table>									0	Falling edge of $\overline{NMI}$ is detected.	1	Rising edge of $\overline{NMI}$ is detected.																								
0	Falling edge of $\overline{NMI}$ is detected.																																			
1	Rising edge of $\overline{NMI}$ is detected.																																			
<b>External Reset</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>Reset was caused by watchdog timer overflow</td> </tr> <tr> <td>1</td> <td>Reset was caused by external reset signal (initial value)</td> </tr> </table>									0	Reset was caused by watchdog timer overflow	1	Reset was caused by external reset signal (initial value)																								
0	Reset was caused by watchdog timer overflow																																			
1	Reset was caused by external reset signal (initial value)																																			
<b>Standby Timer Select 2 to 0 (ZTAT and Mask ROM Versions)</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Clock settling time = 8,192 states (initial value)</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Clock settling time = 16,384 states</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Clock settling time = 32,768 states</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Clock settling time = 65,536 states</td> </tr> <tr> <td>1</td> <td>0</td> <td>—</td> <td>Clock settling time = 131,072 states</td> </tr> <tr> <td>1</td> <td>1</td> <td>—</td> <td>Unused</td> </tr> </table>									0	0	0	Clock settling time = 8,192 states (initial value)	0	0	1	Clock settling time = 16,384 states	0	1	0	Clock settling time = 32,768 states	0	1	1	Clock settling time = 65,536 states	1	0	—	Clock settling time = 131,072 states	1	1	—	Unused				
0	0	0	Clock settling time = 8,192 states (initial value)																																	
0	0	1	Clock settling time = 16,384 states																																	
0	1	0	Clock settling time = 32,768 states																																	
0	1	1	Clock settling time = 65,536 states																																	
1	0	—	Clock settling time = 131,072 states																																	
1	1	—	Unused																																	
<b>Standby Timer Select 2 to 0 (F-ZTAT Version)</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Settling time = 8,192 states (initial value)</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Settling time = 16,384 states</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Settling time = 32,768 states</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Settling time = 65,536 states</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>Settling time = 131,072 states</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>Settling time = 1,024 states</td> </tr> <tr> <td>1</td> <td>1</td> <td>—</td> <td>Unused</td> </tr> </table>									0	0	0	Settling time = 8,192 states (initial value)	0	0	1	Settling time = 16,384 states	0	1	0	Settling time = 32,768 states	0	1	1	Settling time = 65,536 states	1	0	0	Settling time = 131,072 states	1	0	1	Settling time = 1,024 states	1	1	—	Unused
0	0	0	Settling time = 8,192 states (initial value)																																	
0	0	1	Settling time = 16,384 states																																	
0	1	0	Settling time = 32,768 states																																	
0	1	1	Settling time = 65,536 states																																	
1	0	0	Settling time = 131,072 states																																	
1	0	1	Settling time = 1,024 states																																	
1	1	—	Unused																																	
<b>Software Standby</b> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0</td> <td>SLEEP instruction causes transition to sleep mode. (initial value)</td> </tr> <tr> <td>1</td> <td>SLEEP instruction causes transition to software standby mode.</td> </tr> </table>									0	SLEEP instruction causes transition to sleep mode. (initial value)	1	SLEEP instruction causes transition to software standby mode.																								
0	SLEEP instruction causes transition to sleep mode. (initial value)																																			
1	SLEEP instruction causes transition to software standby mode.																																			

**MDCR—Mode Control Register****H'FFC5****System Control**

Bit	7	6	5	4	3	2	1	0
Initial value	—	—	—	—	—	—	MDS1	MDS0
Read/Write	—	—	—	—	—	—	R	R
							<b>Mode Select Bits</b>	

Value at mode pins.

Note: \* Determined by inputs at pins MD<sub>1</sub> and MD<sub>0</sub>.

**ISCR—IRQ Sense Control Register****H'FFC6****System Control**

Bit	7	6	5	4	3	2	1	0
Initial value	IRQ7SC	IRQ6SC	IRQ5SC	IRQ4SC	IRQ3SC	IRQ2SC	IRQ1SC	IRQ0SC
Read/Write	0	0	0	0	0	0	0	0

**IRQ<sub>0</sub> to IRQ<sub>7</sub> Sense Control**

0	IRQ <sub>0</sub> to IRQ <sub>7</sub> are level-sensed (active low).
1	IRQ <sub>0</sub> to IRQ <sub>7</sub> are edge-sensed (falling edge).

**IER—IRQ Enable Register****H'FFC7****System Control**

Bit	7	6	5	4	3	2	1	0
Initial value	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E
Read/Write	0	0	0	0	0	0	0	0

**IRQ<sub>0</sub> to IRQ<sub>7</sub> Enable**

0	IRQ <sub>0</sub> to IRQ <sub>7</sub> are disabled.
1	IRQ <sub>0</sub> to IRQ <sub>7</sub> are enabled.

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

**Clock Select**

TCR			STCR		Description
CKS2	CKS1	CKS0	ICKS1	ICKS0	
0	0	0	—	—	Timer stopped
0	0	1	—	0	$\phi_P/8$ internal clock, falling edge
0	0	1	—	1	$\phi_P/2$ internal clock, falling edge
0	1	0	—	0	$\phi_P/64$ internal clock, falling edge
0	1	0	—	1	$\phi_P/32$ internal clock, falling edge
0	1	1	—	0	$\phi_P/1024$ internal clock, falling edge
0	1	1	—	1	$\phi_P/256$ internal clock, falling edge
1	0	0	—	—	Timer stopped
1	0	1	—	—	External clock, rising edge
1	1	0	—	—	External clock, falling edge
1	1	1	—	—	External clock, 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**

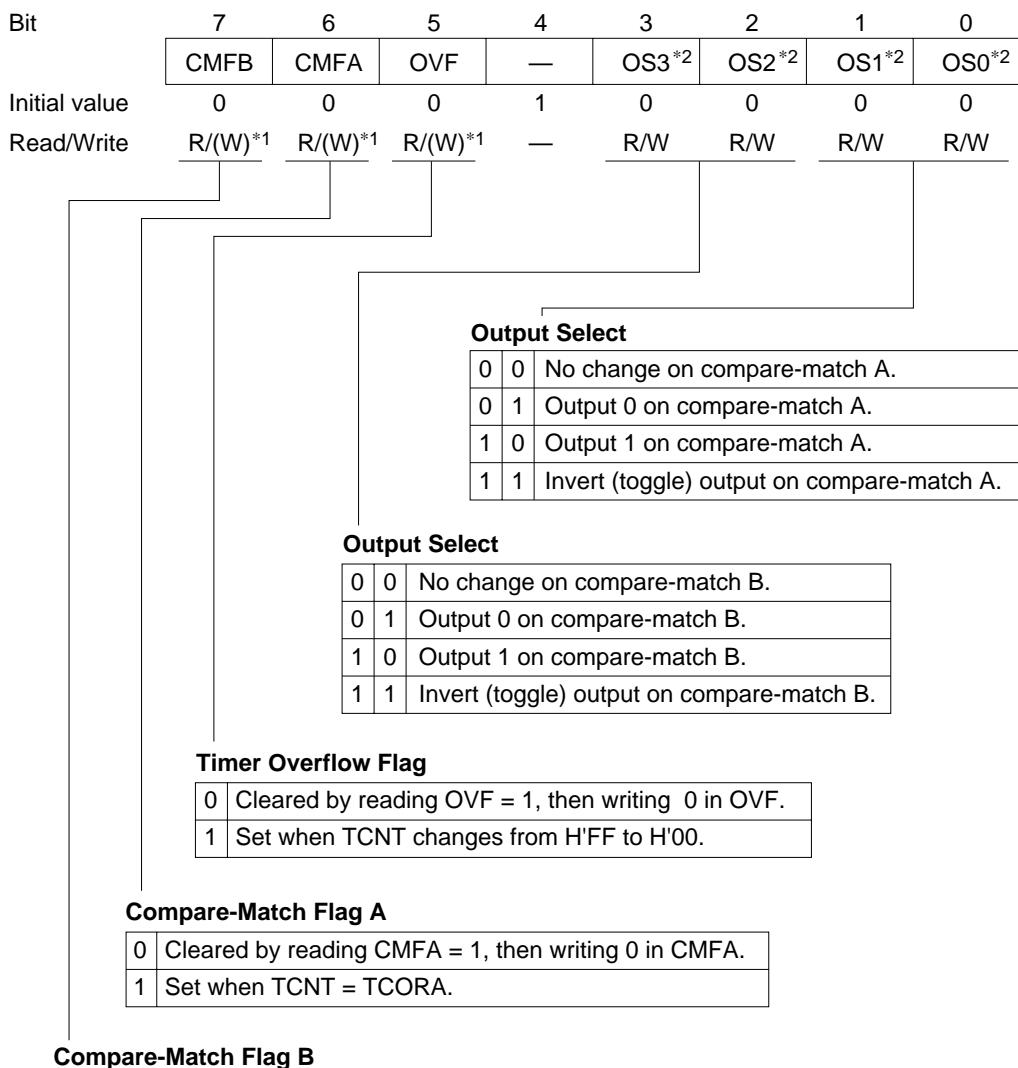
0	Overflow interrupt request is disabled.
1	Overflow interrupt request is enabled.

**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.



Notes: 1. Software can write a 0 in bits 7 to 5 to clear the flags, but cannot write a 1 in these bits.  
 2. When all four bits (OS3 to OS0) are cleared to 0, output is disabled.

**TCORA—Time Constant Register A****H'FFCA****TMR0**

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

The CMFA bit is set to 1 when TCORA = TCNT.

**TCORB—Time Constant Register B****H'FFCB****TMR0**

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

The CMFB bit is set to 1 when TCORB = TCNT.

**TCNT—Timer Counter****H'FFCC****TMR0**

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

Count value

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

**Clock Select**

CKS2	CKS1	CKS0	TCR		STCR	Description
			ICKS1	ICKS0		
0	0	0	—	—	Timer stopped	
0	0	1	0	—	$\phi_P/8$ internal clock, falling edge	
0	0	1	1	—	$\phi_P/2$ internal clock, falling edge	
0	1	0	0	—	$\phi_P/64$ internal clock, falling edge	
0	1	0	1	—	$\phi_P/128$ internal clock, falling edge	
0	1	1	0	—	$\phi_P/1024$ internal clock, falling edge	
0	1	1	1	—	$\phi_P/2048$ internal clock, falling edge	
1	0	0	—	—	Timer stopped	
1	0	1	—	—	External clock, rising edge	
1	1	0	—	—	External clock, falling edge	
1	1	1	—	—	External clock, rising and falling edges	

**Counter Clear**

0	0	Counter is not cleared.
0	1	Cleared by compare-match A.
1	0	Cleared by compare-match B.
1	1	Cleared on rising edge of external reset input.

**Timer Overflow Interrupt Enable**

0	Overflow interrupt request is disabled.
1	Overflow interrupt request is enabled.

**Compare-Match Interrupt Enable A**

0	Compare-match A interrupt request is disabled.
1	Compare-match A interrupt request is enabled.

**Compare-Match Interrupt Enable B**

0	Compare-match B interrupt request is disabled.
1	Compare-match B interrupt request is enabled.

**TCSR—Timer Control/Status Register****H'FFD1****TMR1**

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

Notes: Bit functions are the same as for TMR0.

1. Software can write a 0 in bits 7 to 5 to clear the flags, but cannot write a 1 in these bits.
2. When all four bits (OS3 to OS0) are cleared to 0, output is disabled.

**TCORA—Time Constant Register A****H'FFD2****TMR1**

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

Note: Bit functions are the same as for TMR0.

**TCORB—Time Constant Register B****H'FFD3****TMR1**

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

Note: Bit functions are the same as for TMR0.

**TCNT—Timer Counter****H'FFD4****TMR1**

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

Note: Bit functions are the same as for TMR0.

Note: Bit functions are the same as for SCI1

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

### Transfer Clock Select

IICX*	CKS2	CKS1	CKS0	Clock	Transfer Rate				
					$\phi_P = 4$ MHz	$\phi_P = 5$ MHz	$\phi_P = 8$ MHz	$\phi_P = 10$ MHz	$\phi_P = 16$ MHz
0	0	0	0	$\phi_P/28$	143 kHz	179 kHz	286 kHz	357 kHz	571 kHz
			1	$\phi_P/40$	100 kHz	125 kHz	200 kHz	250 kHz	400 kHz
		1	0	$\phi_P/48$	83.3 kHz	104 kHz	167 kHz	208 kHz	333 kHz
		1	1	$\phi_P/64$	62.5 kHz	78.1 kHz	125 kHz	156 kHz	250 kHz
	1	0	0	$\phi_P/80$	50.0 kHz	62.5 kHz	100 kHz	125 kHz	200 kHz
			1	$\phi_P/100$	40.0 kHz	50.0 kHz	80.0 kHz	100 kHz	160 kHz
		1	0	$\phi_P/112$	35.7 kHz	44.6 kHz	71.4 kHz	89.3 kHz	143 kHz
		1	1	$\phi_P/128$	31.3 kHz	39.1 kHz	62.5 kHz	78.1 kHz	125 kHz
	1	0	0	$\phi_P/56$	71.4 kHz	89.3 kHz	143 kHz	179 kHz	286 kHz
			1	$\phi_P/80$	50.0 kHz	62.5 kHz	100 kHz	125 kHz	200 kHz
		1	0	$\phi_P/96$	41.7 kHz	52.1 kHz	83.3 kHz	104 kHz	167 kHz
		1	1	$\phi_P/128$	31.3 kHz	39.1 kHz	62.5 kHz	78.1 kHz	125 kHz
	1	0	0	$\phi_P/160$	25.0 kHz	31.3 kHz	50.0 kHz	62.5 kHz	100 kHz
			1	$\phi_P/200$	20.0 kHz	25.0 kHz	40.0 kHz	50.0 kHz	80.0 kHz
		1	0	$\phi_P/224$	17.9 kHz	22.3 kHz	35.7 kHz	44.6 kHz	71.4 kHz
		1	1	$\phi_P/256$	15.6 kHz	19.5 kHz	31.3 kHz	39.1 kHz	62.5 kHz

Note: When  $\phi_P = \emptyset$ .

\* IICX is bit 5 of the serial timer control register (STCR).

### Acknowledgement Mode Select

0	Acknowledgement mode
1	Serial mode

### Master/Slave Select and Transmit/Receive Select

0	0	Slave receive mode
	1	Slave transmit mode
1	0	Master receive mode
	1	Master transmit mode

### I<sup>2</sup>C Bus Interface Interrupt Enable

0	Interrupts disabled
1	Interrupts enabled

### I<sup>2</sup>C Bus Interface Enable

0	Interface module disabled, with pins SCL and SDA operating as ports
1	Interface module enabled for transfer operations, with pins SCL and SDA capable of bus drive

Bit	7	6	5	4	3	2	1	0
	BBSY	IRIC	SCP	—	AL	AAS	ADZ	ACKB
Initial value	0	0	1	1	0	0	0	0
Read/Write	R/W	R/(W)*	W	—	R/(W)*	R/(W)*	R/(W)*	R/W

**Acknowledge Bit**

0	Receive mode: 0 is output at acknowledge output timing Transmit mode: indicates that the receiving device has acknowledged the data
1	Receive mode: 1 is output at acknowledge output timing Transmit mode: indicates that the receiving device has not acknowledged the data

**General Call Address Recognition Flag**

0	General call address not recognized Cleared when ICDR data is written (transmit mode) or read (receive mode) Cleared by reading ADZ = 1, then writing 0
1	General call address recognized Set when the general call address is detected in slave receive mode

**Slave Address Recognition Flag**

0	Slave address or general call address not recognized (Initial value) Cleared when ICDR data is written (transmit mode) or read (receive mode) Cleared by reading AAS = 1, then writing 0
1	Slave address or general call address recognized Set when the slave address or general call address is detected in slave receive mode

**Arbitration Lost Flag**

0	Bus arbitration won Cleared when ICDR data is written (transmit mode) or read (receive mode) Cleared by reading AL = 1, then writing 0
1	Arbitration lost Set if the internal SDA and bus line disagree at the rise of SCL in master transmit mode Set if the internal SCL is high at the fall of SCL in master transmit mode

**Start Condition/Stop Condition Prohibit**

0	Writing 0 issues a start or stop condition, in combination with BBSY
1	Reading always results in 1 Writing is ignored

**I<sup>2</sup>C Bus Interface Interrupt Request Flag**

0	Waiting for transfer, or transfer in progress Cleared by reading IRIC = 1, then writing 0
1	Interrupt requested Set to 1 at the following times: Master mode • End of data transfer • Bus arbitration lost Slave mode (when FS = 0) • When the slave address is matched, and whenever a data transfer ends at timing of a retransmit start condition after address matching or a stop condition is detected • When a general call address is detected, and whenever a data transfer ends at timing of a retransmit start condition after address detection or a stop condition is detected Slave mode (when FS = 1) • End of data transfer

**Bus Busy**

0	Bus is free Cleared by detection of a stop condition
1	Bus is busy Set by detection of a start condition

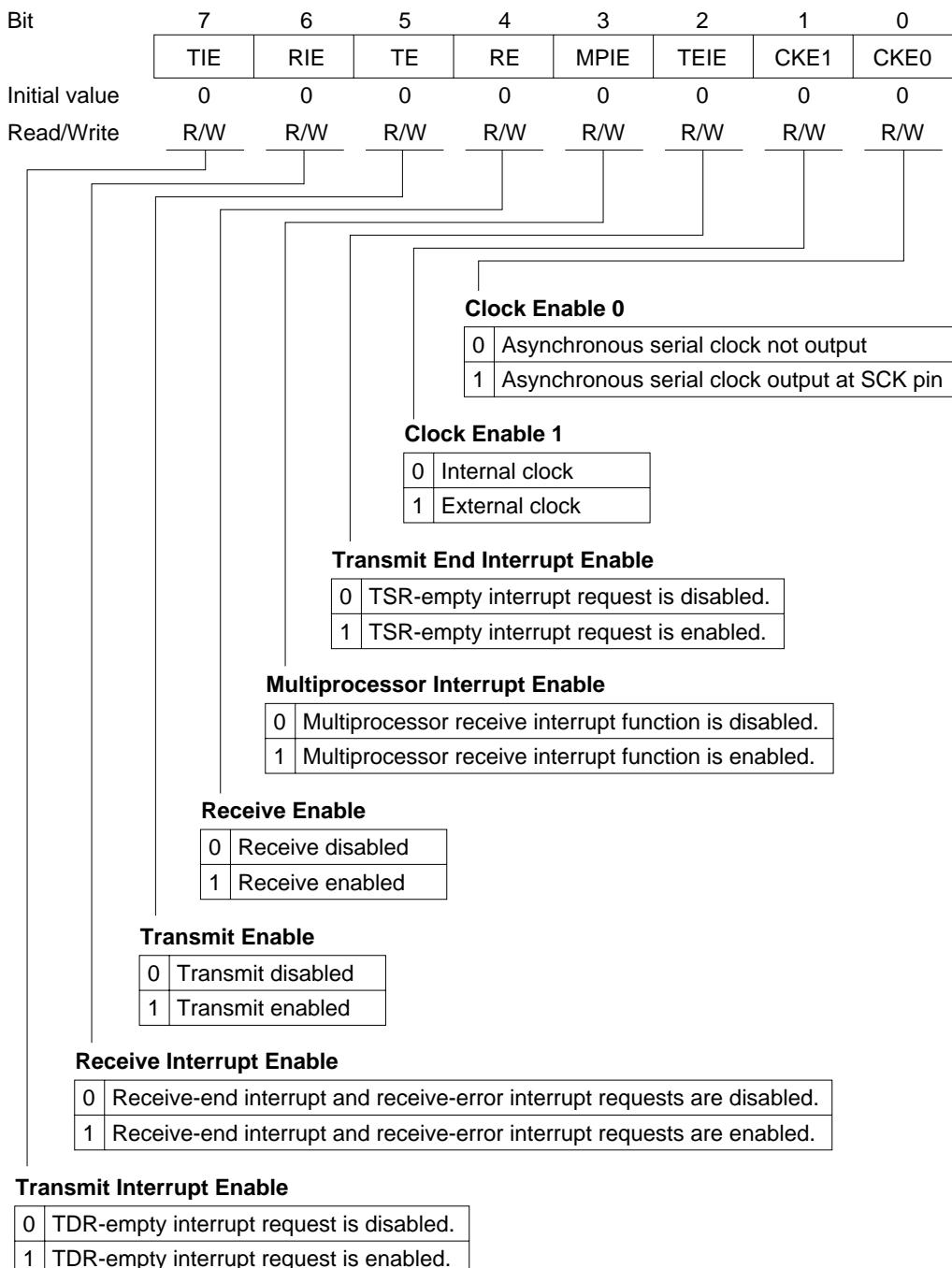
Note: \* Only 0 can be written, to clear the flag.

**BRR—Bit Rate Register****H'FFD9****SCI0**

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

Constant that determines the bit rate

Note: Bit functions are the same as for SCI1.



Note: Bit functions are the same as for SCI1.

**TDR—Transmit Data Register****H'FFDB****SCI0**

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

↓  
Transmit data

Note: Bit functions are the same as for SCI1.

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

**Multiprocessor Bit Transfer**

0	Multiprocessor bit = 0 in transmit data.
1	Multiprocessor bit = 1 in transmit data.

**Multiprocessor Bit**

0	Multiprocessor bit = 0 in receive data.
1	Multiprocessor bit = 1 in receive data.

**Transmit End**

0	Cleared by reading TDRE = 1, then writing 0 in TDRE.
1	Set to 1 when TE = 0, or when TDRE = 1 at the end of character transmission.

**Parity Error**

0	Cleared by reading PER = 1, then writing 0 in PER.
1	Set when a parity error occurs (parity of receive data does not match parity selected by O/E bit in SMR).

**Framing Error**

0	Cleared by reading FER = 1, then writing 0 in FER.
1	Set when a framing error occurs (stop bit is 0).

**Overrun Error**

0	Cleared by reading ORER = 1, then writing 0 in ORER.
1	Set when an overrun error occurs (next data is completely received while RDRF bit is set to 1).

**Receive Data Register Full**

0	Cleared by reading RDRF = 1, then writing 0 in RDRF.
1	Set when one character is received normally and transferred from RSR to RDR.

**Transmit Data Register Empty**

0	Cleared by reading TDRE = 1, then writing 0 in TDRE.
1	Set when: <ol style="list-style-type: none"> <li>1. Data is transferred from TDR to TSR.</li> <li>2. TE is cleared while TDRE = 0.</li> </ol>

Note: \* Software can write a 0 in bits 7 to 3 to clear the flags, but cannot write a 1 in these bits.  
 Bit functions are the same as for SCI1.

583

**RDR—Receive Data Register****H'FFDD****SCI0**

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

Note: Bit functions are the same as for SCI1.

**ICDR—I<sup>2</sup>C Bus Data Register****H'FFDE****I<sup>2</sup>C**

Bit	7	6	5	4	3	2	1	0
	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0
Initial value	—	—	—	—	—	—	—	—

↓  
Transmit/receive data

**SAR—Slave Address Register****H'FFDF****I<sup>2</sup>C**

Bit	7	6	5	4	3	2	1	0
	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS
Initial value	0	0	0	0	0	0	0	0

↓  
Slave address

**Format Select**

0	Addressing format, slave address recognized
1	Non-addressing format

Bit	7	6	5	4	3	2	1	0				
Initial value	MLS	WAIT	—	—	—	BC2	BC1	BC0				
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W				
<b>Bit Counter</b>												
			Bits/Frame									
			Serial Mode		Acknowledgement Mode							
			0	0	0	8	9					
					1	1	2					
					1	0	3					
					1	3	4					
			1	0	0	4	5					
					1	5	6					
					1	0	7					
					1	7	8					
<b>Wait Insertion Bit</b>												
<table border="1"> <tr> <td>0</td><td>Data and acknowledge transferred consecutively</td></tr> <tr> <td>1</td><td>Wait inserted between data and acknowledge</td></tr> </table>									0	Data and acknowledge transferred consecutively	1	Wait inserted between data and acknowledge
0	Data and acknowledge transferred consecutively											
1	Wait inserted between data and acknowledge											

**MSB-First/LSB-First**

0	MSB-first
1	LSB-first

## **ADDRA (H and L)—A/D Data Register A**

**H'FFE0, H'FFE1**

A/D

#### ADDRB (H and L)=A/D Data Register B

H'FFE2, H'FFE3

A/D

#### ADDRC (H and L)=A/D Data Register C

H'FFE4, H'FFE5

A/D

**ADDRD (H and L)—A/D Data Register D****H'FFE6, H'FFE7****A/D**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	—	—	—	—	—	—
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

**ADDRDH**      **ADDRDL**

**A/D Conversion Data**      **Reserved Bits**

10-bit data giving an 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 <sub>0</sub>	AN <sub>0</sub>
		1	AN <sub>1</sub>	AN <sub>0</sub> , AN <sub>1</sub>
	1	0	AN <sub>2</sub>	AN <sub>0</sub> to AN <sub>2</sub>
		1	AN <sub>3</sub>	AN <sub>0</sub> to AN <sub>3</sub>
	1	0	AN <sub>4</sub>	AN <sub>4</sub>
		1	AN <sub>5</sub>	AN <sub>4</sub> , AN <sub>5</sub>
	1	0	AN <sub>6</sub>	AN <sub>4</sub> to AN <sub>6</sub>
		1	AN <sub>7</sub>	AN <sub>4</sub> to AN <sub>7</sub>

**Clock Select**

0	Conversion time = 266 states (max)
1	Conversion time = 134 states (max)

Note: When  $\emptyset P = \emptyset$ **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/C conversion starts and continues cyclically on all selected channels until 0 is written in this bit.

**A/D Interrupt Enable**

0	The A/D interrupt request (ADI) is disabled.
1	The A/D interrupt request (ADI) is enabled.

**A/D End Flag**

0	Cleared from 1 to 0 when CPU reads ADF = 1, then writes 0 in ADF.
1	Set to 1 at the following times: 1. Single mode: at the completion of A/D conversion 2. Scan mode: when all selected channels have been converted.

Note: \* Only 0 can be written, to clear the flag.

**ADCR—A/D Control Register****H'FFE9****A/D**

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

**Trigger Enable**

0	ADTRG is disabled.
1	ADTRG is enabled. A/D conversion can be started by external trigger, or by software.

**HICR—Host Interface Control Register****H'FFF0****HIF**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	IBFIE2	IBFIE1	FGA20E
Initial value	1	1	1	1	1	0	0	0
Host Read/Write	—	—	—	—	—	—	—	—
Slave Read/Write	—	—	—	—	—	R/W	R/W	R/W

**Fast Gate A20 Enable**

0	Fast A20 gate function disabled
1	Fast A20 gate function enabled

**Input Buffer Full Interrupt Enable 1**

0	IDR1 input buffer full interrupt disabled
1	IDR1 input buffer full interrupt enabled

**Input Buffer Full Interrupt Enable 2**

0	IDR2 input buffer full interrupt disabled
1	IDR2 input buffer full interrupt enabled

**KMIMR—Keyboard Matrix Interrupt Mask Register****H'FFF1****HIF**

Bit	7	6	5	4	3	2	1	0
	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0
Initial value	1	0	1	1	1	1	1	1
Read/Write	R/W							

**Keyboard Matrix Interrupt Mask**

0	Key-sense input interrupt request enabled
1	Key-sense input interrupt request disabled (initial value)*

Note: \* Initial value of KMIMR6 is 0.

**KMPCR—Port 6 Input Pull-Up Control Register****H'FFF2****HIF (port 6)**

Bit	7	6	5	4	3	2	1	0
	KM <sub>7</sub> PCR	KM <sub>6</sub> PCR	KM <sub>5</sub> PCR	KM <sub>4</sub> PCR	KM <sub>3</sub> PCR	KM <sub>2</sub> PCR	KM <sub>1</sub> PCR	KM <sub>0</sub> PCR
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W							

**Port 6 Input Pull-Up Control**

0	Input pull-up transistor is off. (initial value)
1	Input pull-up transistor is on.

**KMIMRA—Keyboard Matrix Interrupt Mask Register A****HIF**

Bit	7	6	5	4	3	2	1	0
	KMIMR15	KMIMR14	KMIMR13	KMIMR12	KMIMR11	KMIMR10	KMIMR9	KMIMR8
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

**Keyboard Matrix Interrupt Mask**

0	Key-sense input interrupt request enabled
1	Key-sense input interrupt request disabled (initial value)

**IDR1—Input Data Register 1****H'FFF4****HIF**

Bit	7	6	5	4	3	2	1	0
	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Initial value	—	—	—	—	—	—	—	—
Host Read/Write	W	W	W	W	W	W	W	W
Slave Read/Write	R	R	R	R	R	R	R	R

Input data (command or data input from host processor)

**ODR1—Output Data Register 1****H'FFF5****HIF**

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Host Read/Write	R	R	R	R	R	R	R	R
Slave Read/Write	R/W							

Output data (data output to host processor)

Bit	7	6	5	4	3	2	1	0
Initial value	DBU	DBU	DBU	DBU	C/D	DBU	IBF	OBF
Host Read/Write	R	R	R	R	R	R	R	R
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R

**Defined By User**

**Command/Data**

0	IDR1 contains data
1	IDR1 contains a command

**Defined By User**

**Output Buffer Full**

0	Host has read ODR1
1	Slave has written to ODR1

**Input Buffer Full**

0	Slave has read IDR1
1	Host has written to IDR1

**DADR0—D/A Data Register 0****H'FFF8****D/A**

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

Data to be converted

**DADR1—D/A Data Register 1****H'FFF9****D/A**

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

Data to be converted

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

**D/A Enable**

Bit 7	Bit 6	Bit 5	Description
DAOE1	DAOE0	DAE	
0	0	—	Channels 0 and 1 disabled
	1	0	Channel 0 enabled, channel 1 disabled
	1	1	Channels 0 and 1 enabled
1	0	0	Channel 0 disabled, channel 1 enabled
	1	1	Channels 0 and 1 enabled
	1	—	Channels 0 and 1 enabled

**D/A Output Enable 0**

0	Analog output at DA0 disabled
1	Analog conversion in channel 0 and output at DA0 enabled

**D/A Output Enable 1**

0	Analog output at DA1 disabled
1	Analog conversion in channel 1 and output at DA1 enabled

**IDR2—Input Data Register 2****H'FFFC****HIF**

Bit	7	6	5	4	3	2	1	0
	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
Initial value	—	—	—	—	—	—	—	—
Host Read/Write	W	W	W	W	W	W	W	W
Slave Read/Write	R	R	R	R	R	R	R	R

Input data (command or data input from host processor)

**ODR2—Output Data Register 2****H'FFFD****HIF**

Bit	7	6	5	4	3	2	1	0
	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
Initial value	—	—	—	—	—	—	—	—
Host Read/Write	R	R	R	R	R	R	R	R
Slave Read/Write	R/W							

Output data (data output to host processor)

**STR2—Status Register 2****H'FFFE****HIF**

Bit	7	6	5	4	3	2	1	0
	DBU	DBU	DBU	DBU	C/D	DBU	IBF	OBF
Initial value	0	0	0	0	0	0	0	0
Host Read/Write	R	R	R	R	R	R	R	R
Slave Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R

**Output Buffer Full**

0	Host has read ODR2
1	Slave has written to ODR2

**Input Buffer Full**

0	Slave has read IDR2
1	Host has written to IDR2

**Defined By User**

**Command/Data**

0	IDR2 contains data
1	IDR2 contains a command

**Defined By User**

# Appendix C I/O Port Block Diagrams

## C.1 Port 1 Block Diagram

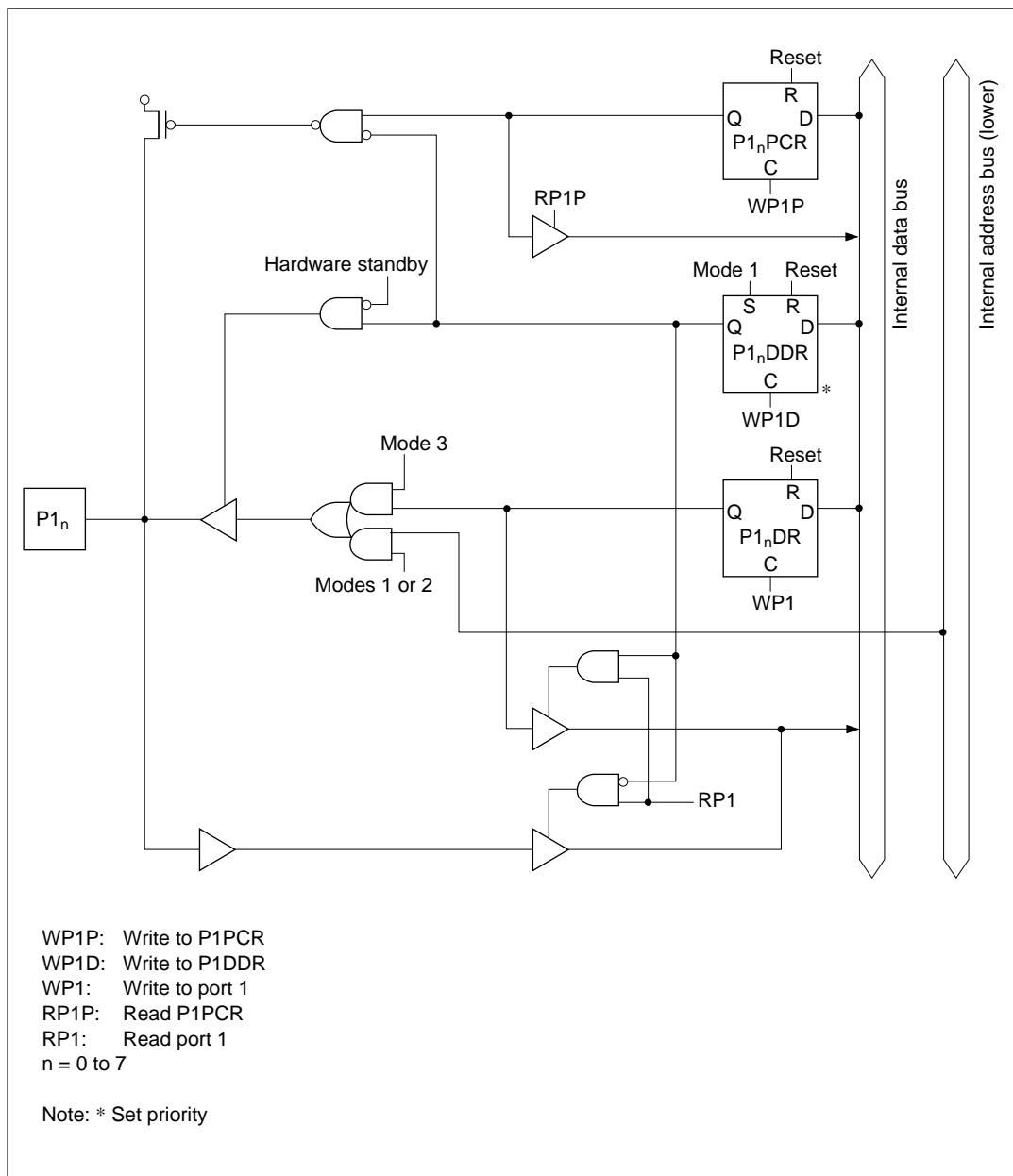
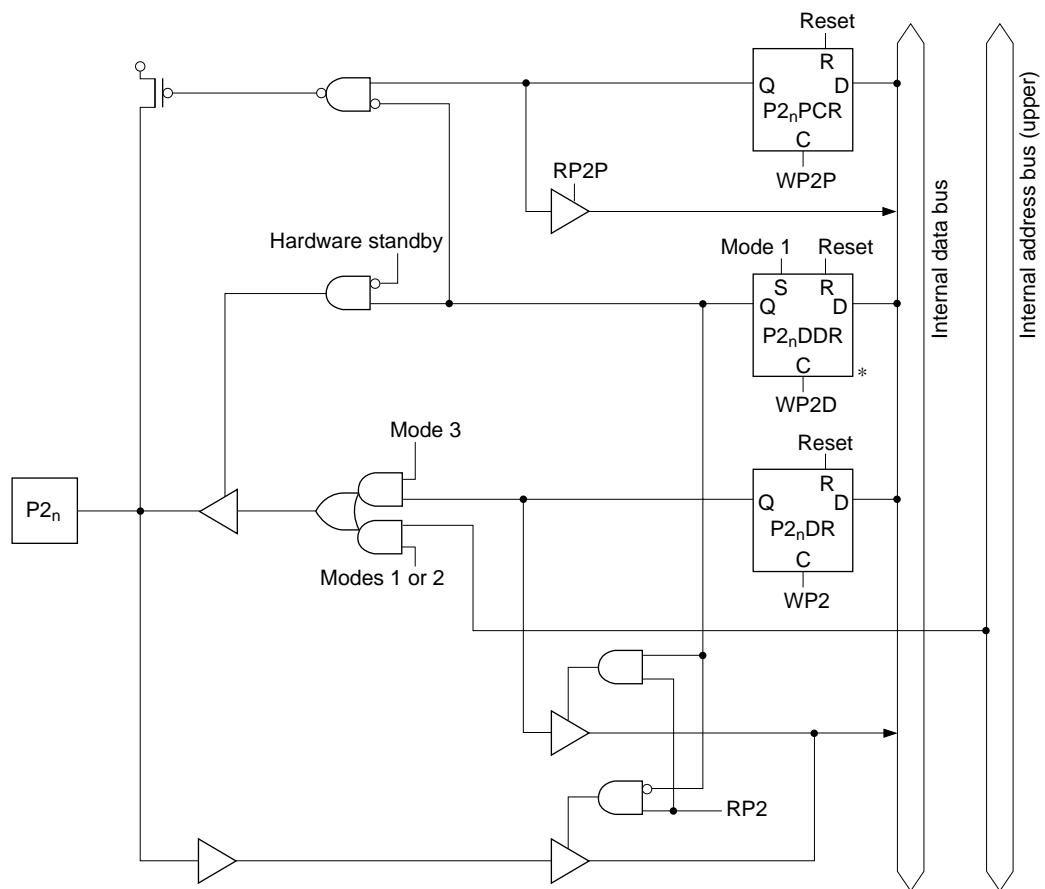


Figure C-1 Port 1 Block Diagram

## C.2 Port 2 Block Diagram



WP2P: Write to P2PCR

WP2D: Write to P2DDR

WP2: Write to port 2

RP2P: Read P2PCR

RP2: Read port 2

n = 0 to 7

Note: \* Set priority

Figure C-2 Port 2 Block Diagram

### C.3 Port 3 Block Diagram

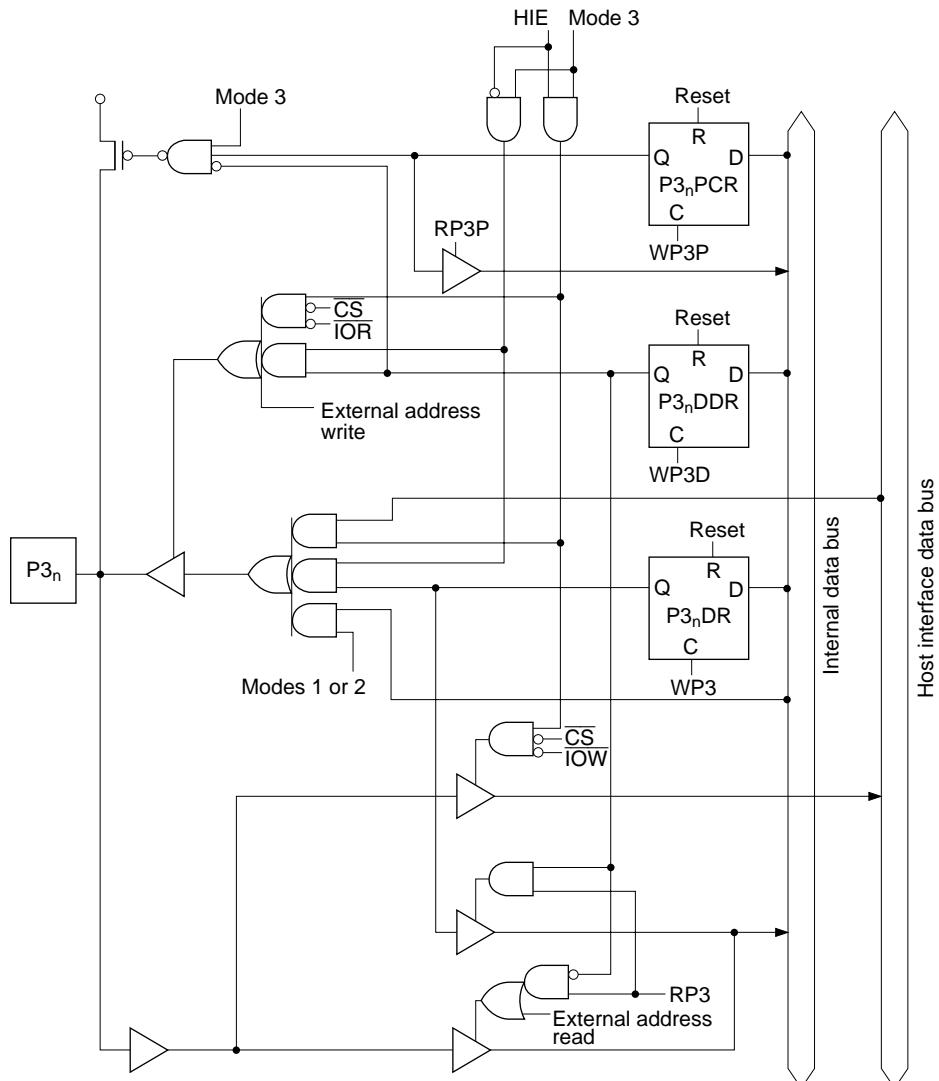


Figure C-3 Port 3 Block Diagram

## C.4 Port 4 Block Diagrams

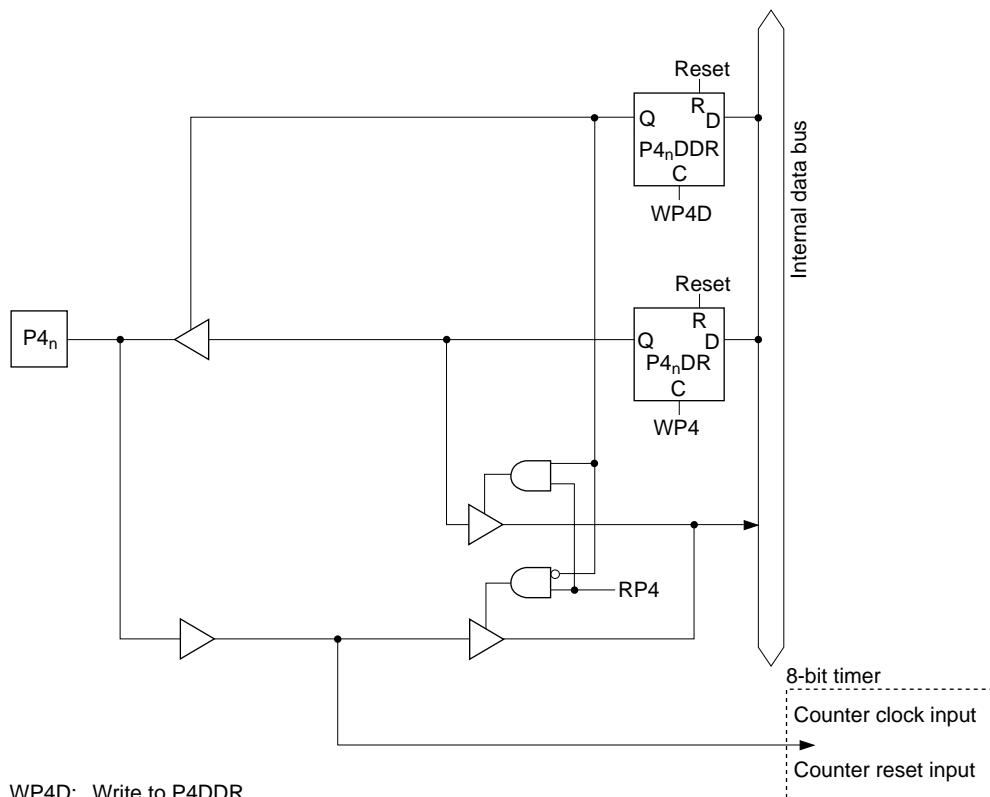
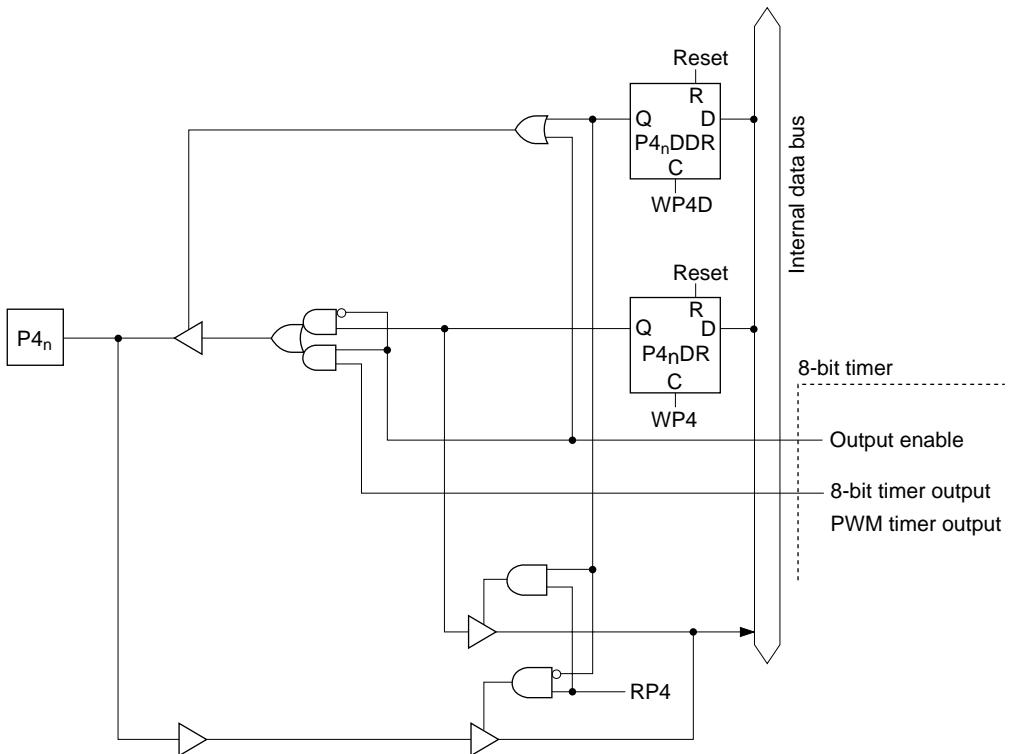


Figure C-4 (a) Port 4 Block Diagram (Pins  $P4_0, P4_2$ )



WP4D: Write to P4DDR

WP4: Write to port 4

RP4: Read port 4

n = 1, 6, 7

**Figure C-4 (b) Port 4 Block Diagram (Pins P4<sub>1</sub>, P4<sub>6</sub>, P4<sub>7</sub>)**

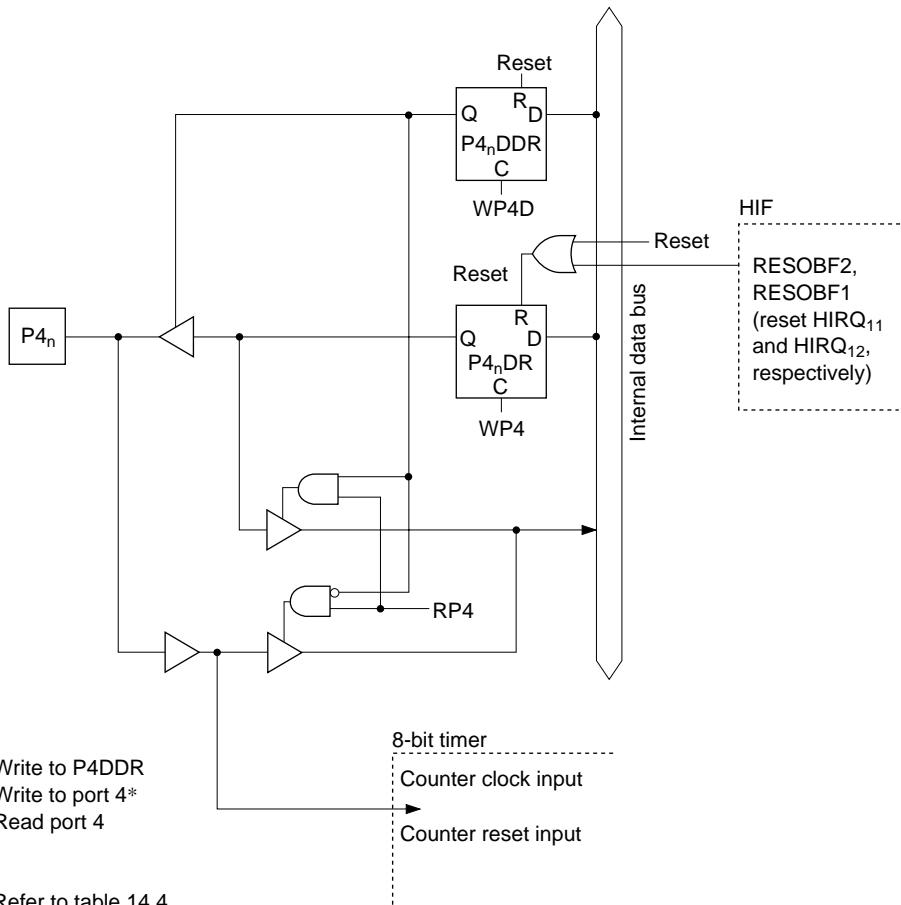
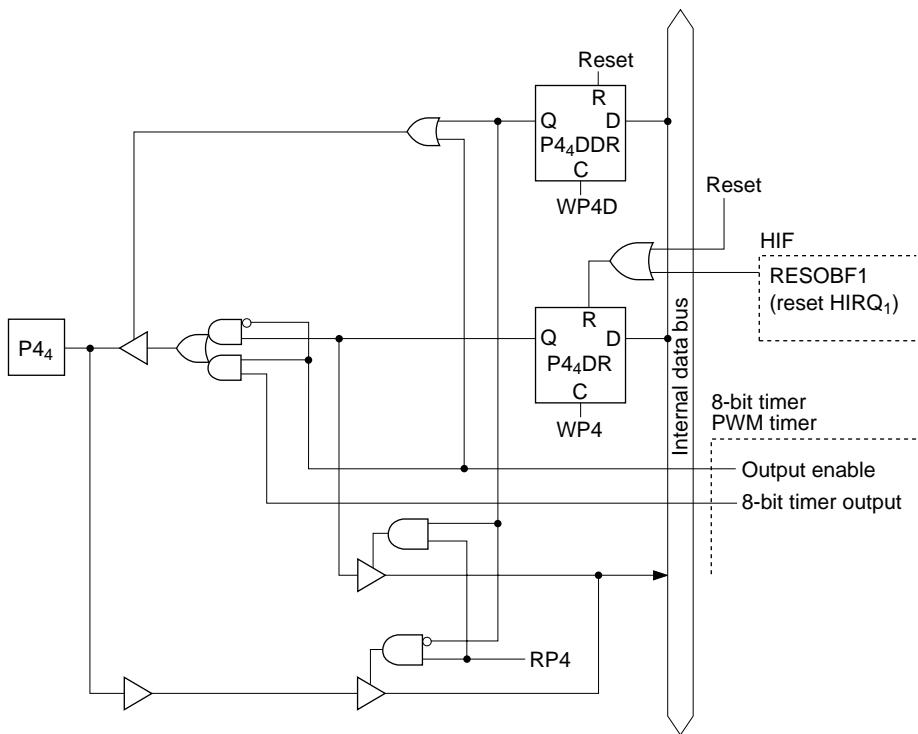


Figure C-4 (c) Port 4 Block Diagram (Pins P4<sub>3</sub>, P4<sub>5</sub>)



WP4D: Write to P4DDR

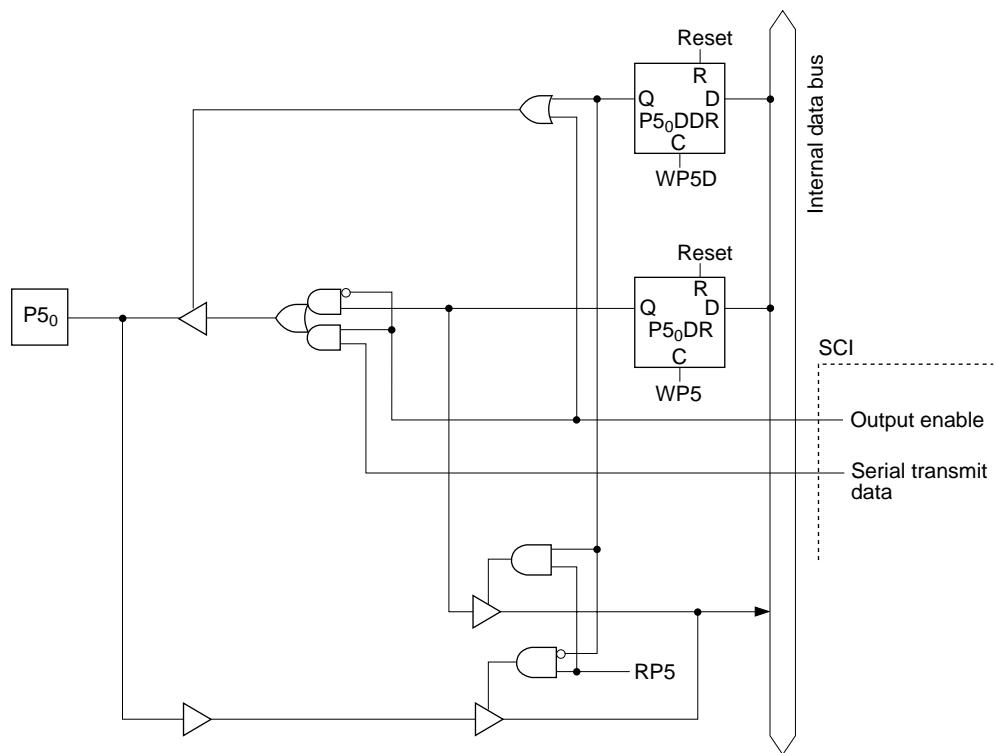
WP4: Write to port 4\*

RP4: Read port 4

Note: \* Refer to table 14.4.

**Figure C-4 (d) Port 4 Block Diagram (Pin P44)**

## C.5 Port 5 Block Diagrams



WP5D: Write to P5DDR

WP5: Write to port 5

RP5: Read port 5

Figure C-5 (a) Port 5 Block Diagram (Pin P50)

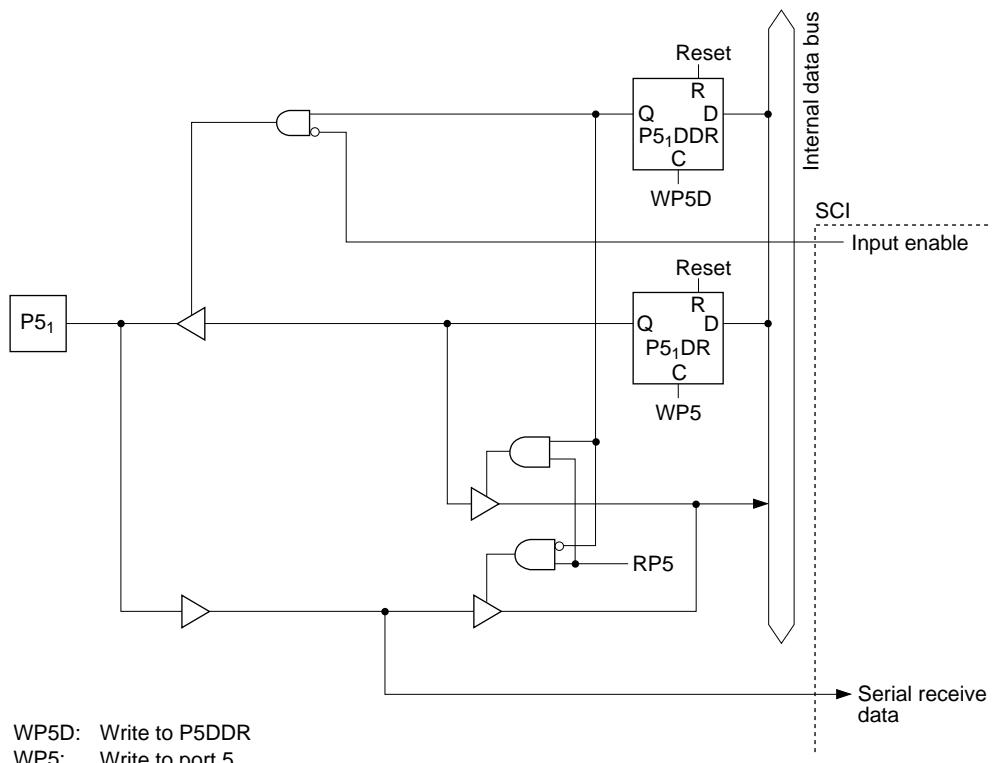
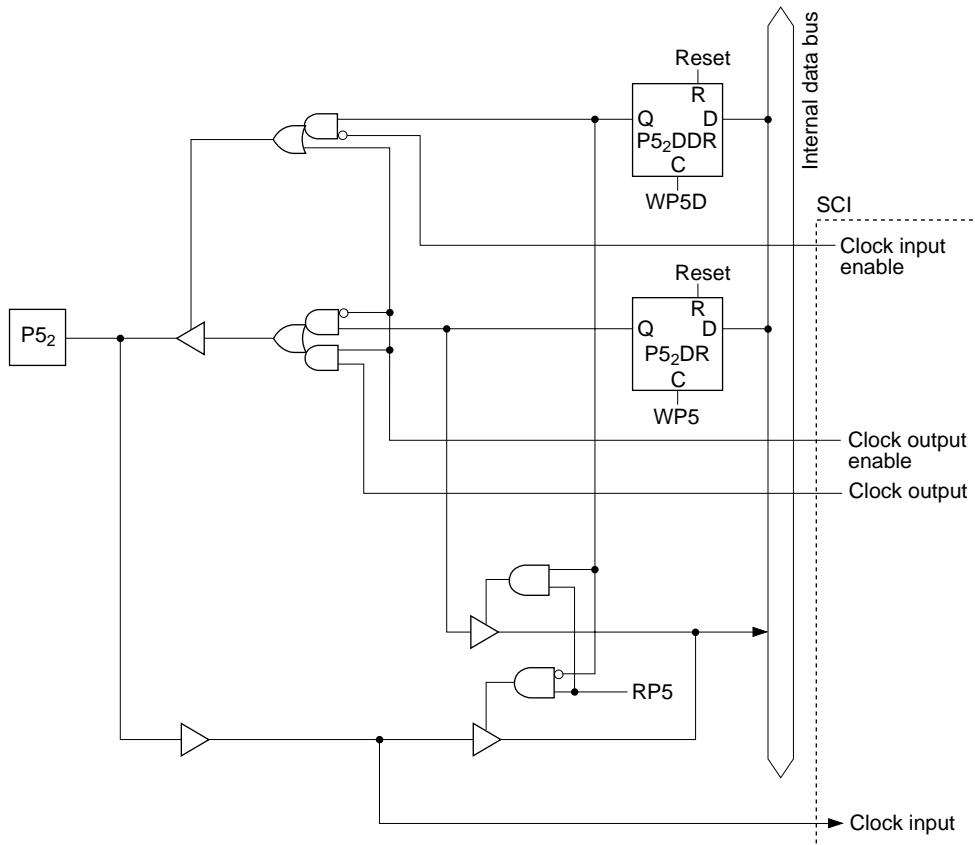
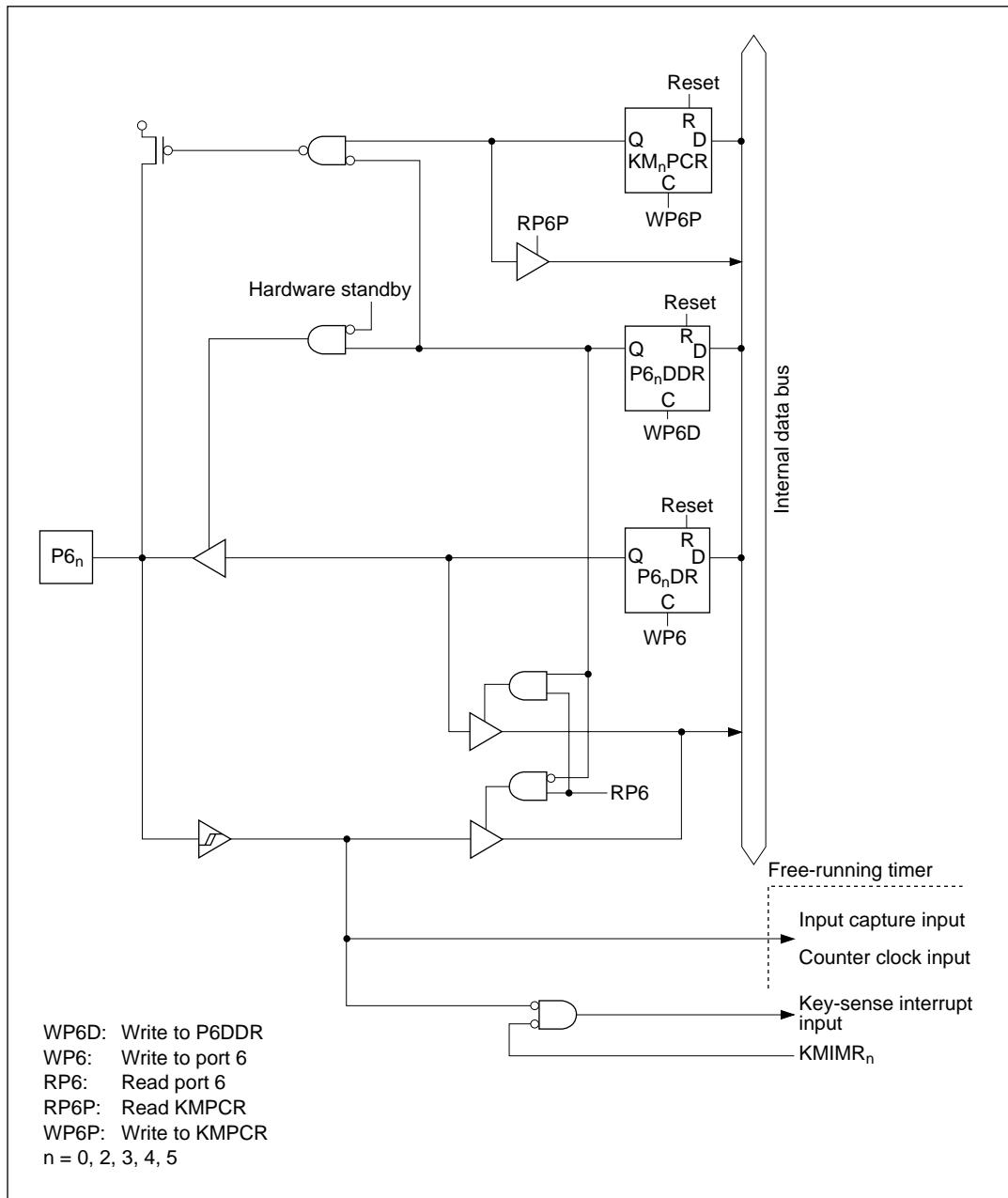


Figure C-5 (b) Port 5 Block Diagram (Pin P5<sub>1</sub>)

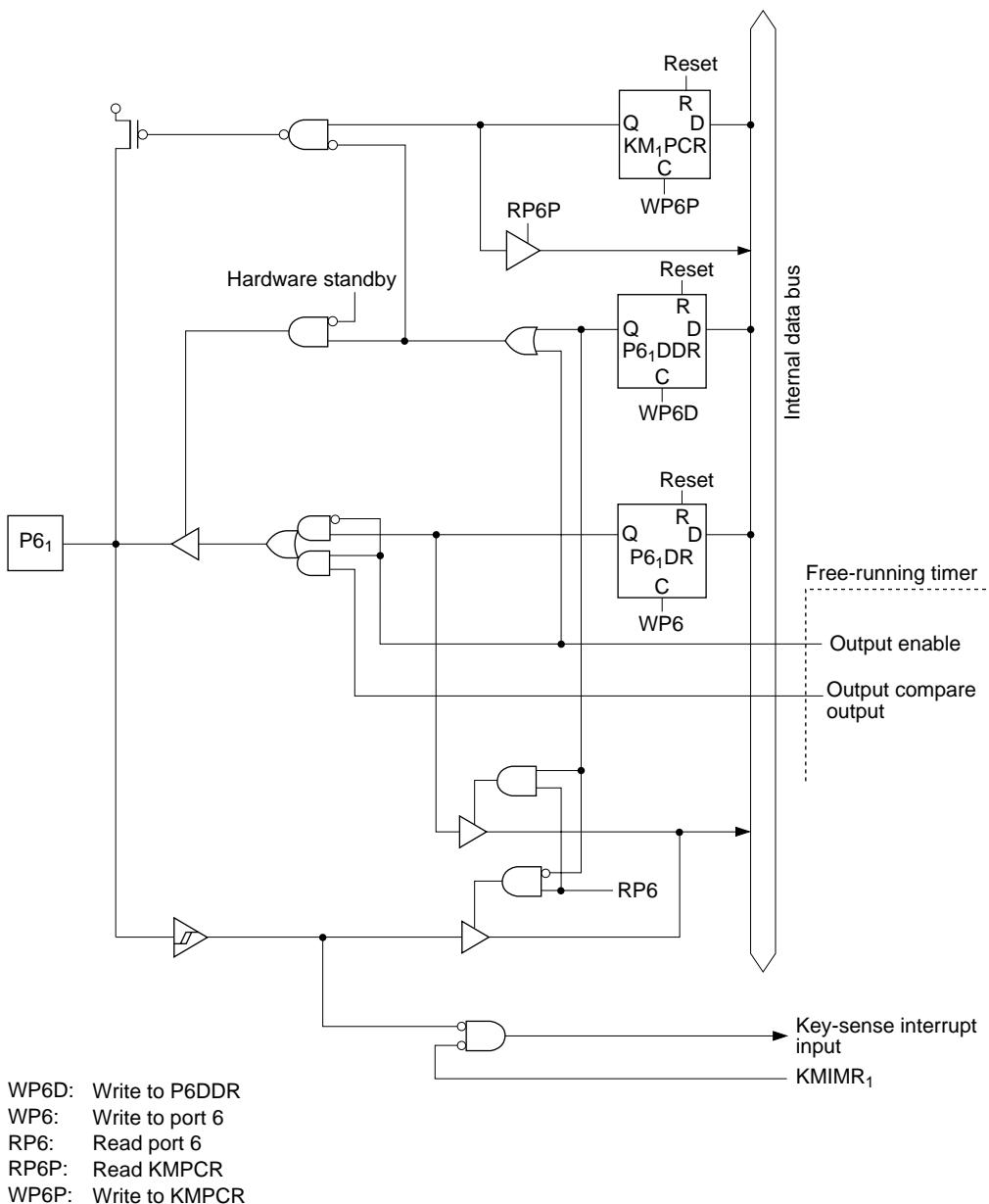


**Figure C-5 (c) Port 5 Block Diagram (Pin P5<sub>2</sub>)**

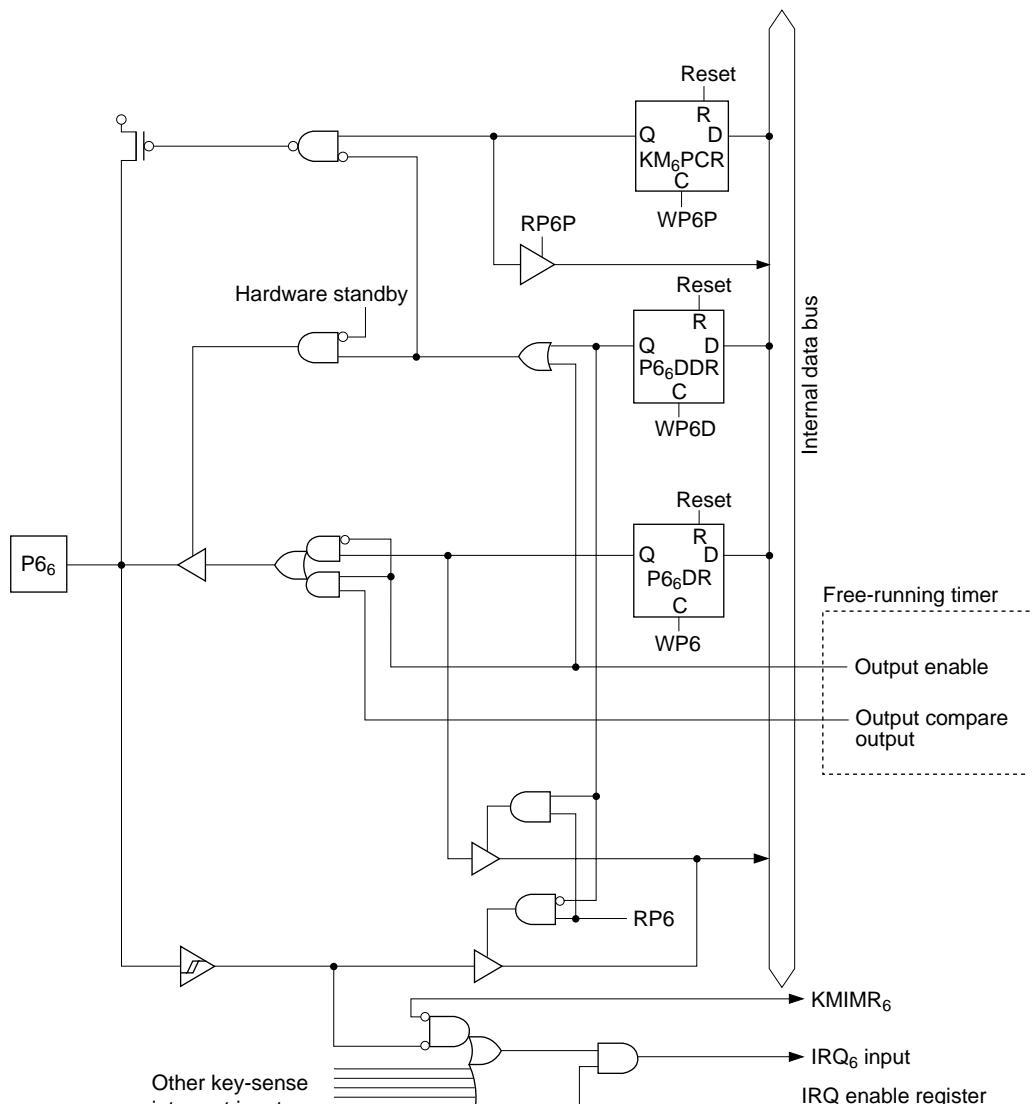
## C.6 Port 6 Block Diagrams



**Figure C-6 (a) Port 6 Block Diagram (Pins P6<sub>0</sub>, P6<sub>2</sub>, P6<sub>3</sub>, P6<sub>4</sub>, P6<sub>5</sub>)**



**Figure C-6 (b) Port 6 Block Diagram (Pin P6<sub>1</sub>)**



WP6D: Write to P6DDR  
 WP6: Write to port 6  
 RP6: Read port 6  
 RP6P: Read KMPCR  
 WP6P: Write to KMPCR

Figure C-6 (c) Port 6 Block Diagram (Pin P6<sub>6</sub>)

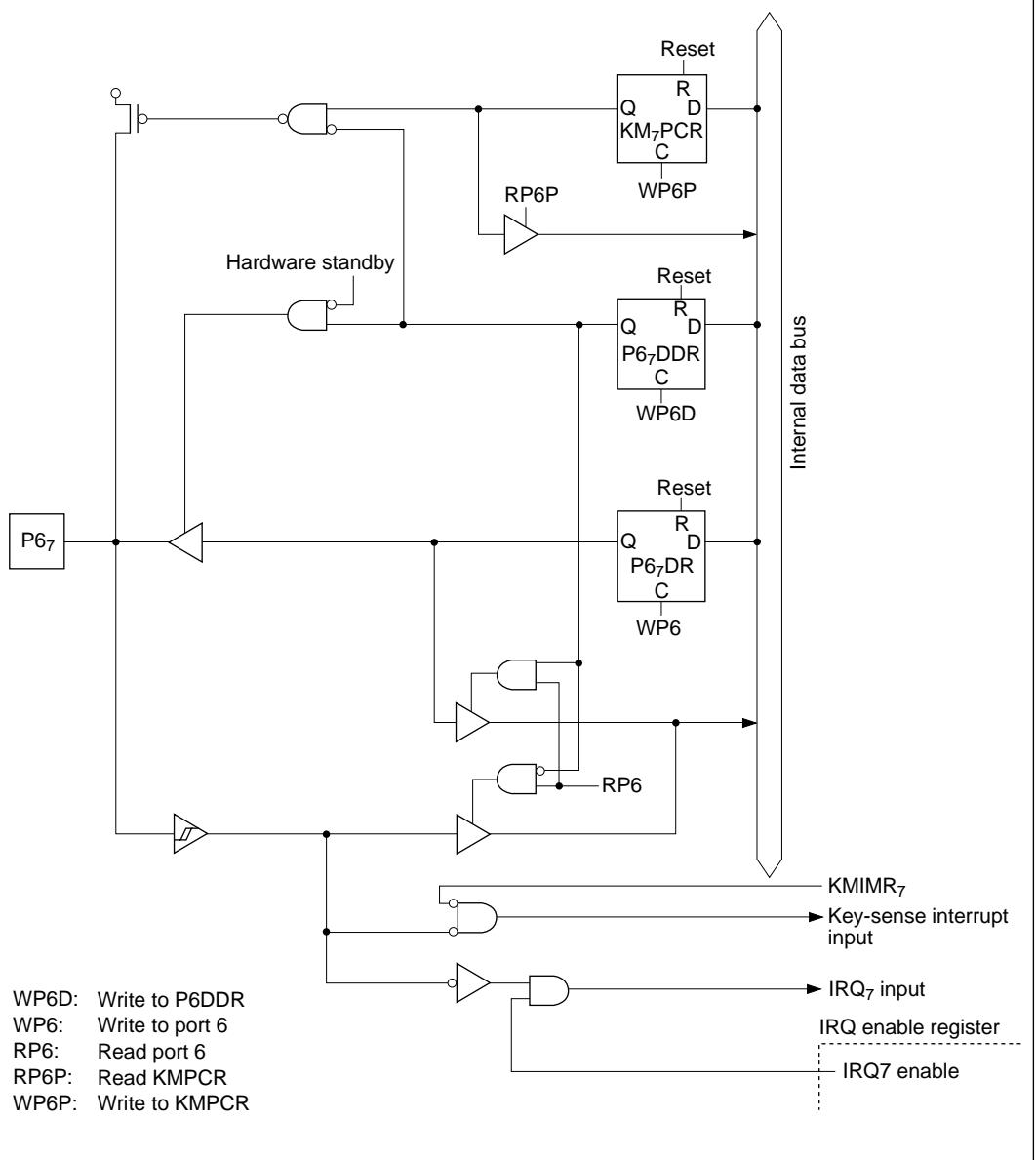
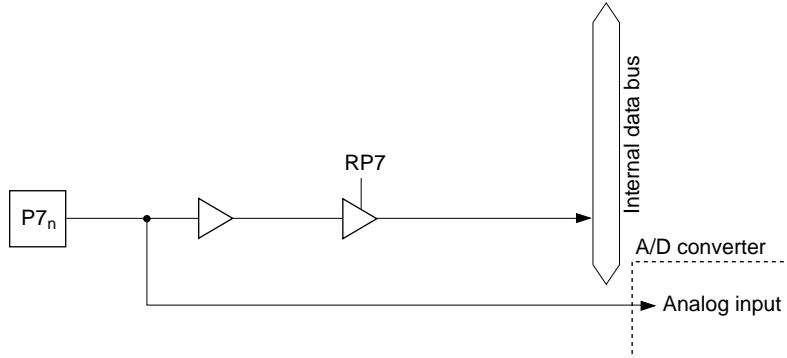


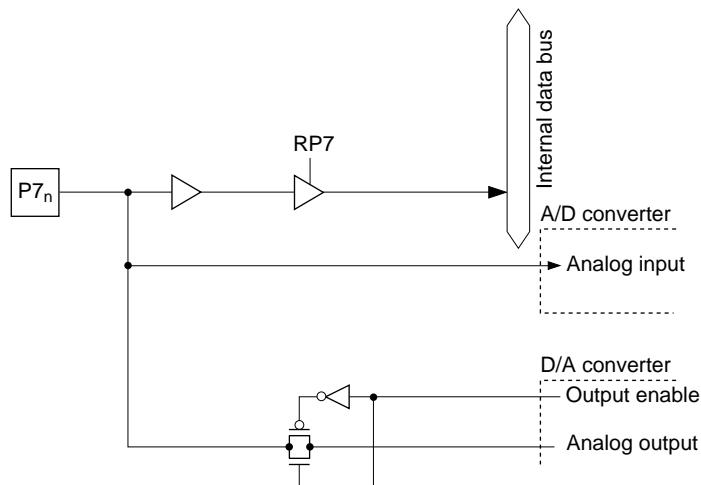
Figure C-6 (d) Port 6 Block Diagram (Pin P6<sub>7</sub>)

## C.7 Port 7 Block Diagrams



RP7: Read port 7  
n = 0 to 5

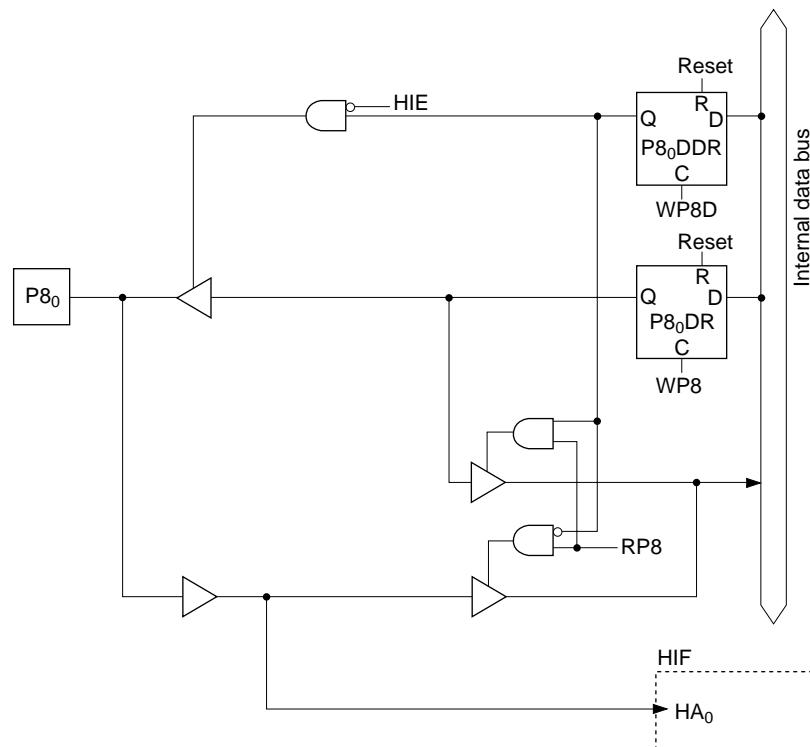
**Figure C-7 (a) Port 7 Block Diagram (Pins P7<sub>0</sub> to P7<sub>5</sub>)**



RP7: Read port 7  
n = 6, 7

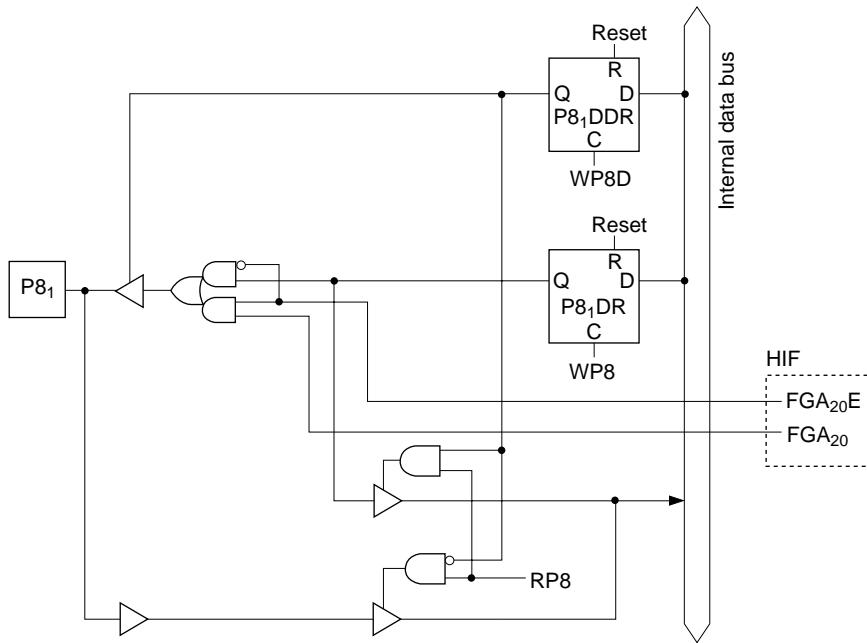
**Figure C-7 (b) Port 7 Block Diagram (Pins P7<sub>6</sub> and P7<sub>7</sub>)**

## C.8 Port 8 Block Diagrams



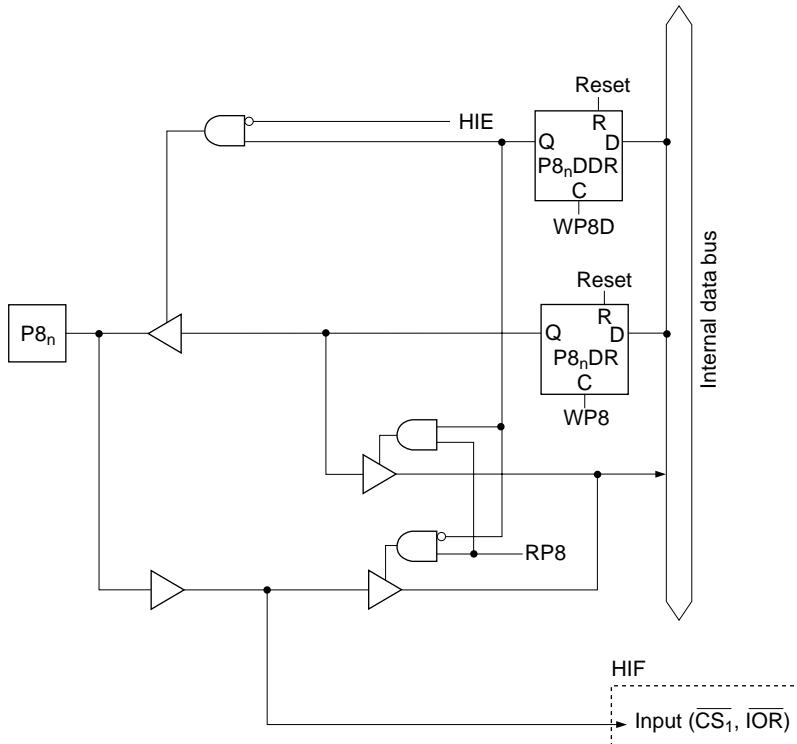
WP8D: Write to P8DDR  
WP8: Write to port 8  
RP8: Read port 8

Figure C-8 (a) Port 8 Block Diagram (Pin P8<sub>0</sub>)

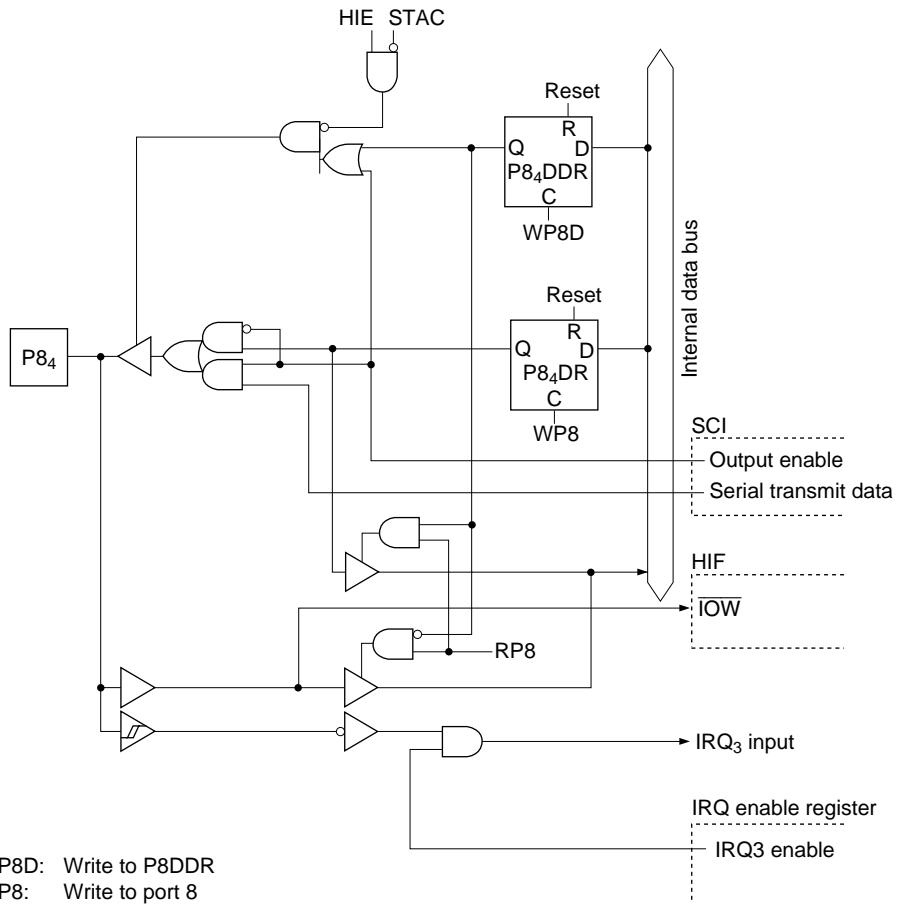


WP8D: Write to P8DDR  
 WP8: Write to port 8  
 RP8: Read port 8

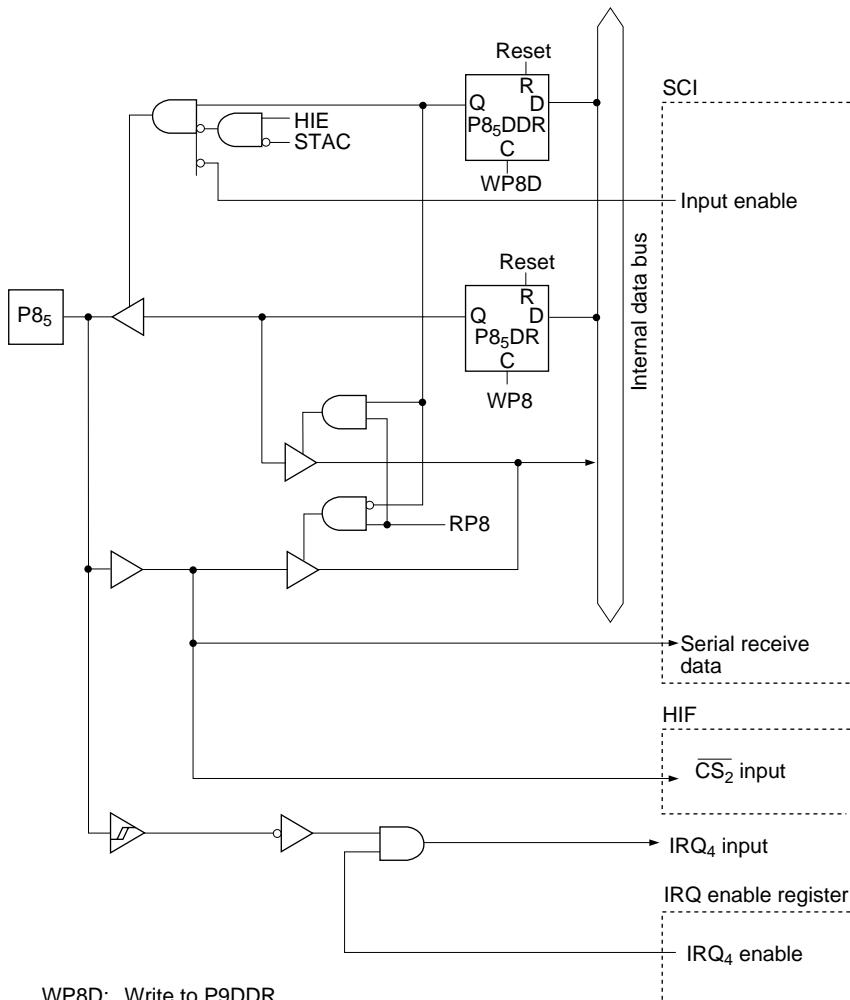
**Figure C-8 (b) Port 8 Block Diagram (Pin P8<sub>1</sub>)**



**Figure C-8 (c) Port 8 Block Diagram (Pins P8<sub>2</sub>, P8<sub>3</sub>)**

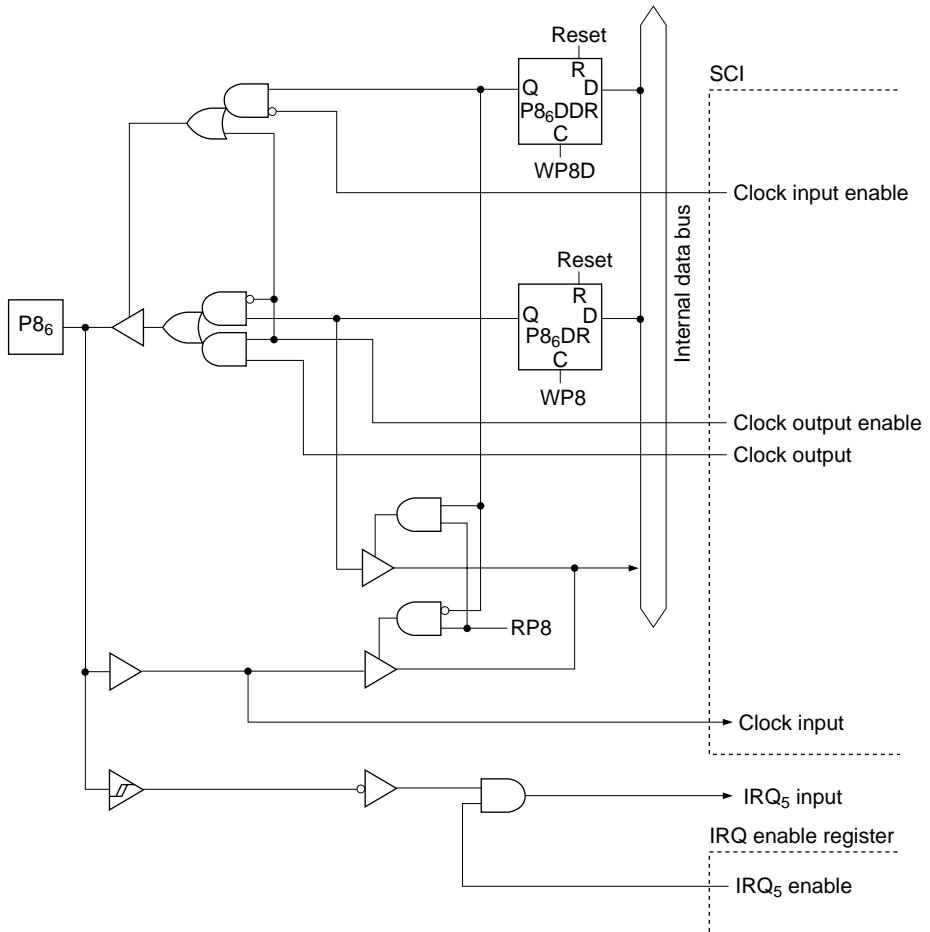


**Figure C-8 (d) Port 8 Block Diagram (Pin P84)**



WP8D: Write to P9DDR  
 WP8: Write to port 8  
 RP8: Read port 8

Figure C-8 (e) Port 8 Block Diagram (Pin P85)



WP8D: Write to P8DDR  
 WP8: Write to port 8  
 RP8: Read port 8

Note: For a block diagram when the SCL pin function is selected, see section 13, I<sup>2</sup>C Bus Interface.

**Figure C-8 (f) Port 8 Block Diagram (Pin P8<sub>6</sub>)**

## C.9 Port 9 Block Diagrams

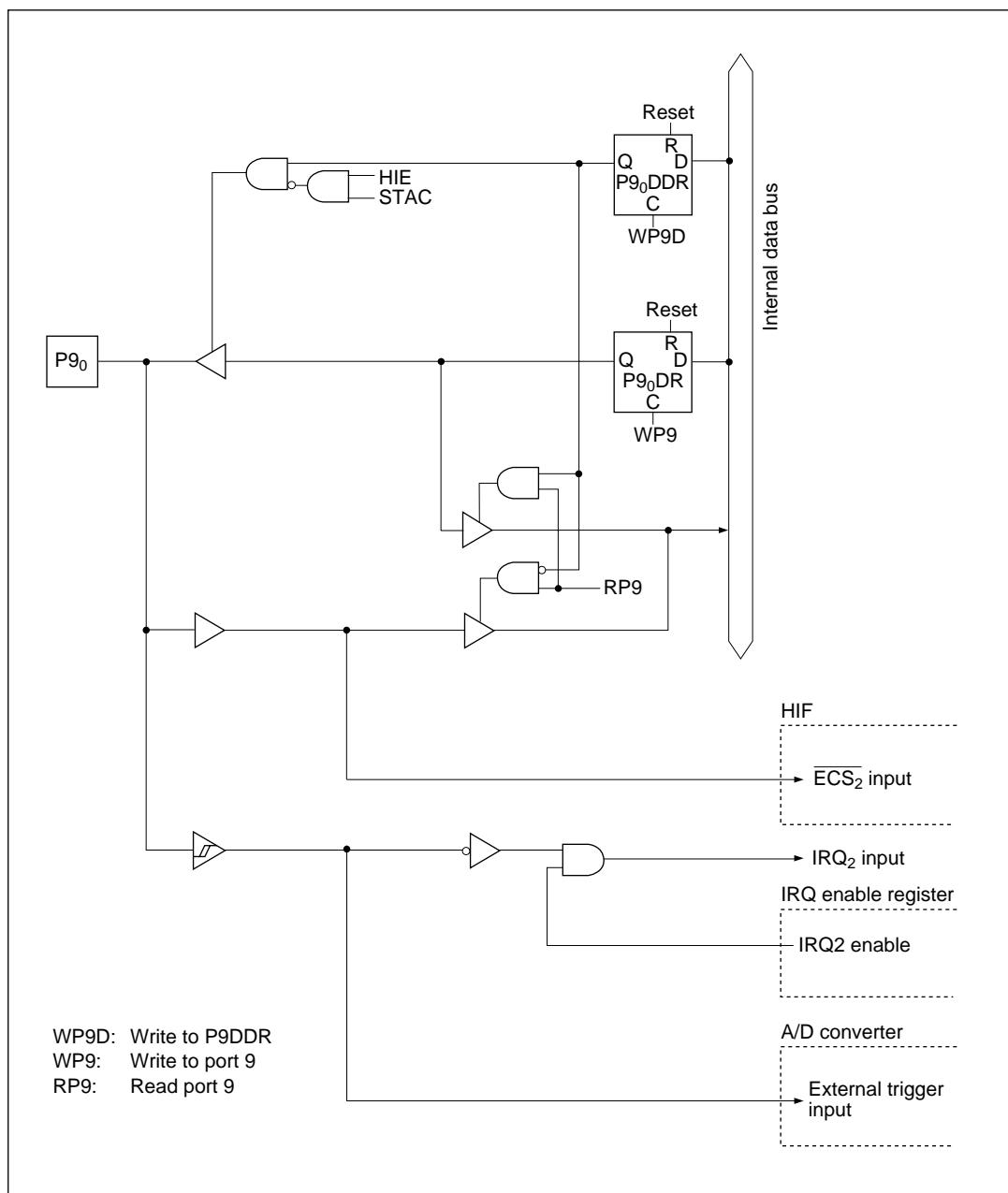
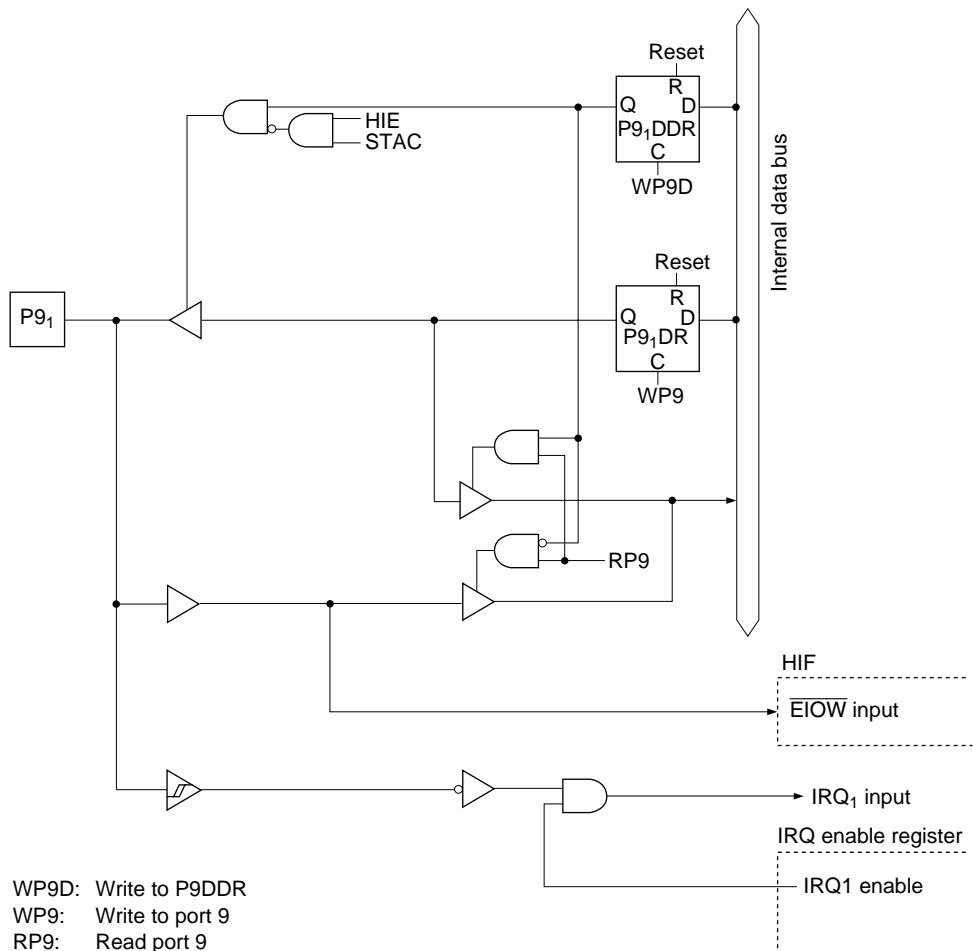


Figure C-9 (a) Port 9 Block Diagram (Pin P9<sub>0</sub>)



**Figure C-9 (b) Port 9 Block Diagram (Pin P9<sub>1</sub>)**

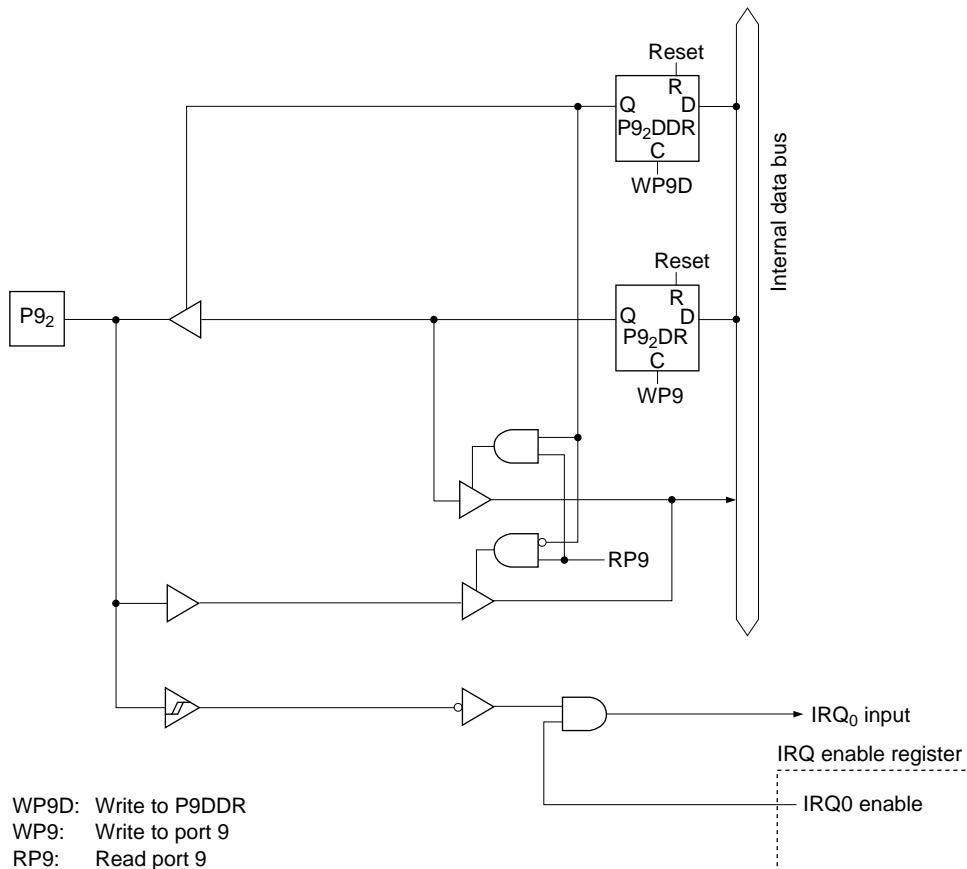
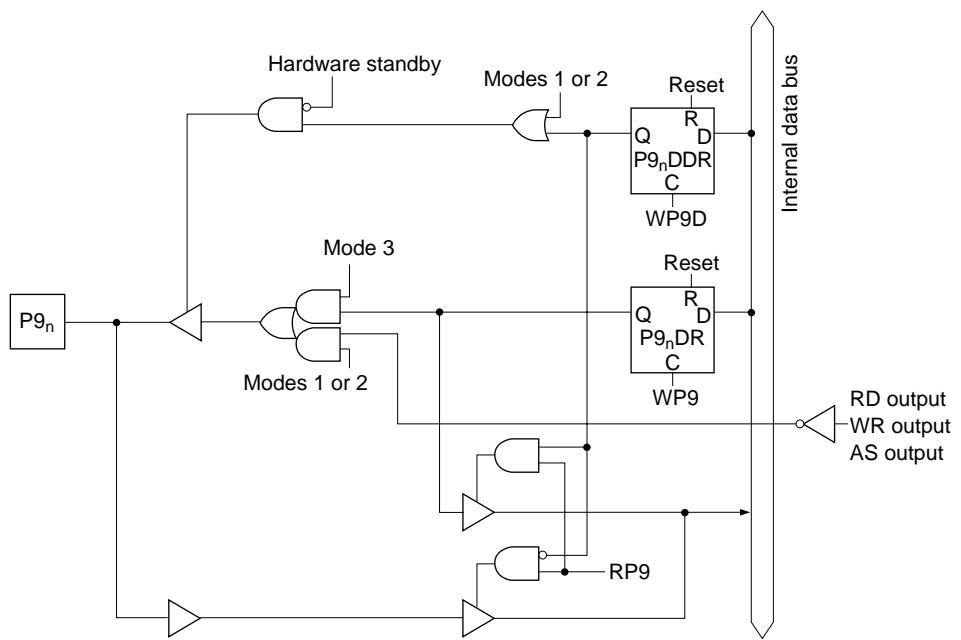
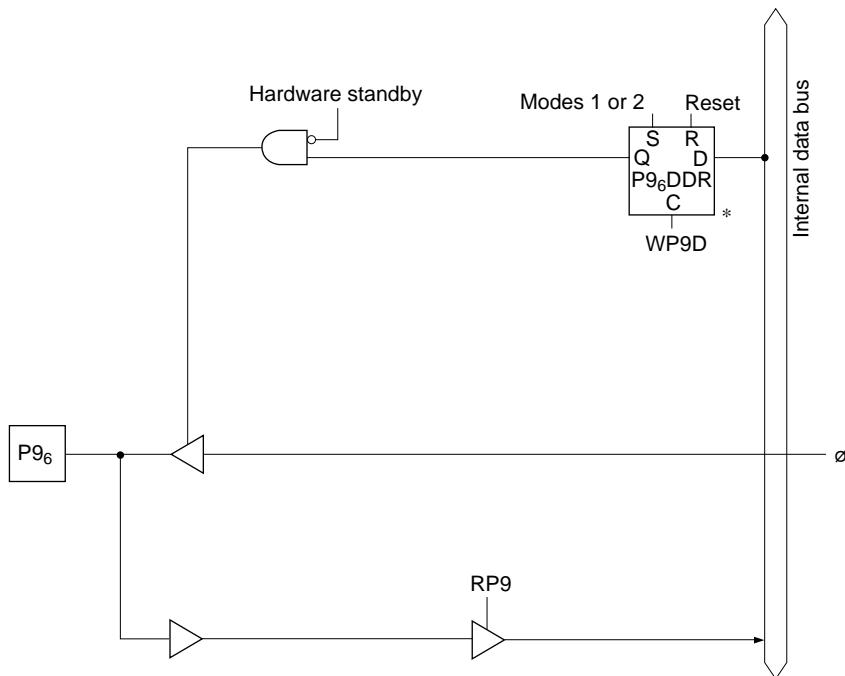


Figure C-9 (c) Port 9 Block Diagram (Pin P9<sub>2</sub>)



WP9D: Write to P9DDR  
 WP9: Write to port 9  
 RP9: Read port 9  
 n = 3, 4, 5

Figure C-9 (d) Port 9 Block Diagram (Pins P9<sub>3</sub>, P9<sub>4</sub>, P9<sub>5</sub>)



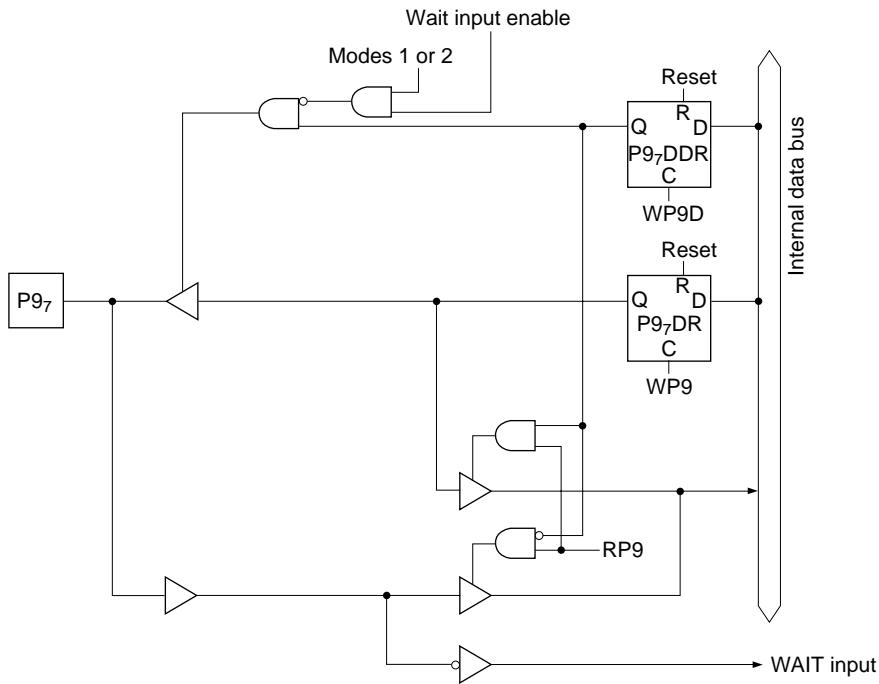
### WP9D: Write to P9DDR

WP9: Write to port 9

RP9: Read port 9

Note: \* Set priority

**Figure C-9 (e) Port 9 Block Diagram (Pin P9<sub>6</sub>)**



WP9D: Write to P9DDR

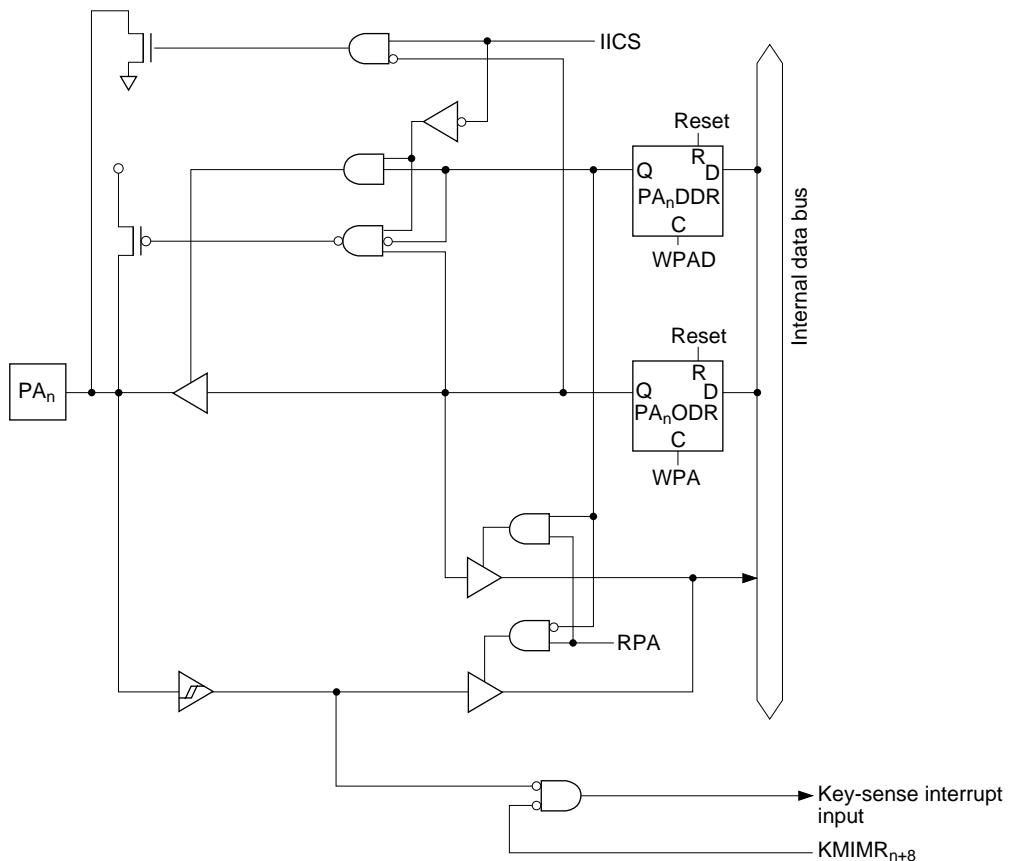
WP9: Write to port 9

RP9: Read port 9

Note: For a block diagram when the SDA pin function is selected, see section 13, I<sup>2</sup>C Bus Interface.

**Figure C-9 (f) Port 9 Block Diagram (Pin P9<sub>7</sub>)**

## C.10 Port A Block Diagram



WPAD: Write to PADDR

WPA: Write to port A

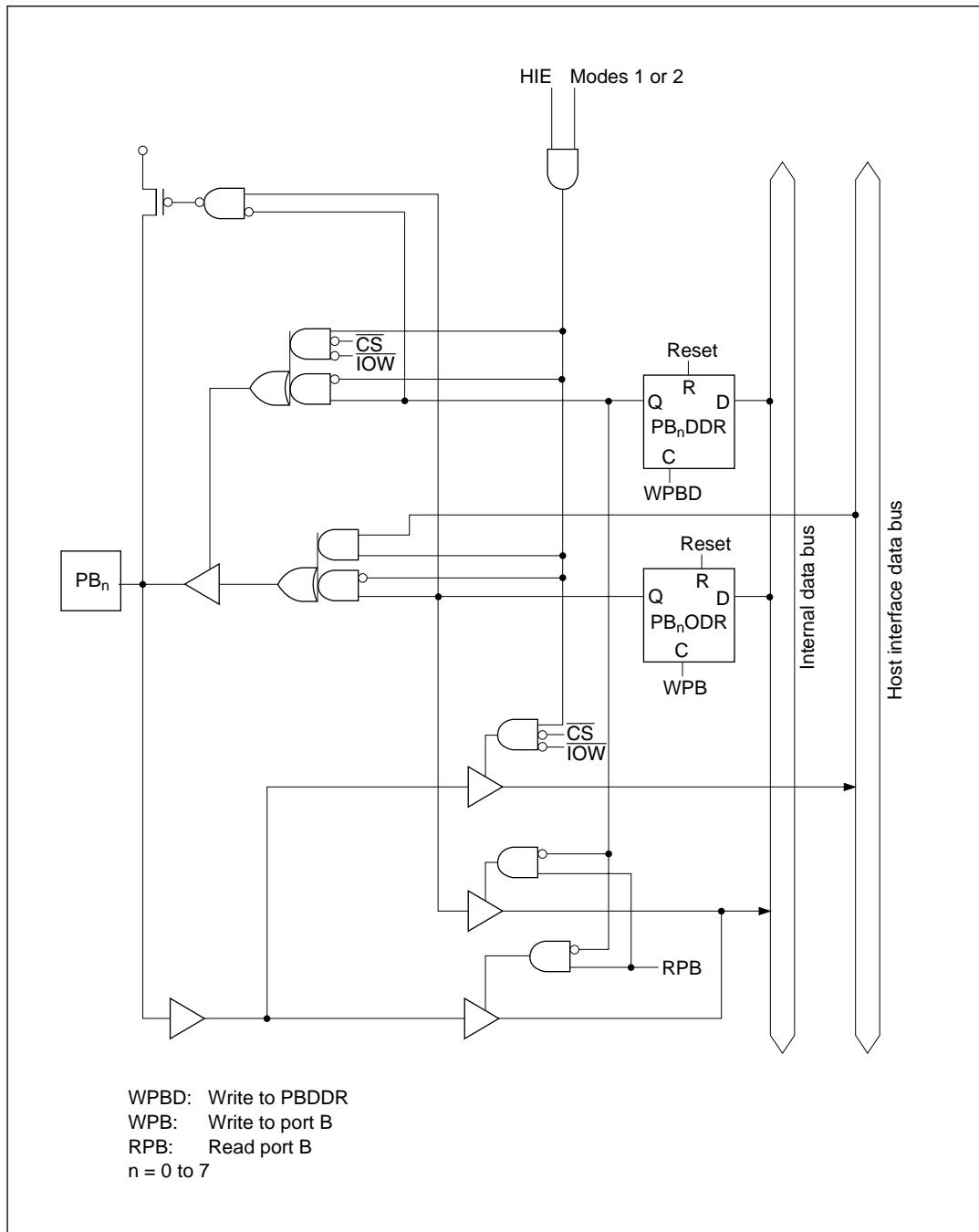
RPA: Read port A

n = 0 to 7

Note: For a block diagram when pins PA<sub>7</sub> to PA<sub>4</sub> are used as bus buffer input/output pins, see section 13, I<sup>2</sup>C Bus Interface.

**Figure C-10 Port A Block Diagram (Pins PA<sub>0</sub> to PA<sub>7</sub>)**

## C.11 Port B Block Diagram



WPBD: Write to PB<sub>n</sub>DDR

WPB: Write to port B

RPB: Read port B

$n = 0$  to 7

Figure C-11 Port B Block Diagram

# Appendix D Pin States

## D.1 Port States in Each Mode

Table D-1 Port States

Pin Name	Mode	Reset	Hardware Standby	Software Standby	Sleep Mode	Normal Operation
P1 <sub>7</sub> to P1 <sub>0</sub>	1	Low	3-state	Low	Prev. state (Addr. output pins: last address accessed)	A <sub>7</sub> to A <sub>0</sub>
A <sub>7</sub> to A <sub>0</sub>	2	3-state		Low if DDR = 1, prev. state if DDR = 0		Addr. output or input port
	3			Prev. state		I/O port
P2 <sub>7</sub> to P2 <sub>0</sub>	1	Low	3-state	Low	Prev. state (Addr. output pins: last address accessed)	A <sub>15</sub> to A <sub>8</sub>
A <sub>15</sub> to A <sub>8</sub>	2	3-state		Low if DDR = 1, prev. state if DDR = 0		Addr. output or input port
	3			Prev. state		I/O port
P3 <sub>7</sub> to P3 <sub>0</sub>	1	3-state	3-state	3-state	3-state	D <sub>7</sub> to D <sub>0</sub>
D <sub>7</sub> to D <sub>0</sub>	2					
	3			Prev. state	Prev. state	I/O port
P4 <sub>7</sub> to P4 <sub>0</sub>	1	3-state	3-state	Prev. state*	Prev. state	I/O port
	2					
	3					
P5 <sub>2</sub> to P5 <sub>0</sub>	1	3-state	3-state	Prev. state*	Prev. state	I/O port
	2					
	3					

Notes:

1. 3-state: High-impedance state
2. Prev. state: Previous state. Input ports are in the high-impedance state (with the MOS pull-up on if DDR = 0, PCR = 1). Output ports hold their previous output level.
3. I/O port: Direction depends on the data direction (DDR) bit. Note that these pins may also be used by the on-chip supporting modules.  
See section 7, I/O Ports, for further information.

\* On-chip supporting modules are initialized, so these pins revert to I/O ports according to the DDR and DR bits.

**Table D-1 Port States (cont)**

Pin Name	Mode	Reset	Hardware Standby	Software Standby	Sleep Mode	Normal Operation
P6 <sub>7</sub> to P6 <sub>0</sub>	1	3-state	3-state	Prev. state*	Prev. state	I/O port
	2					
	3					
P7 <sub>7</sub> to P7 <sub>0</sub>	1	3-state	3-state	3-state	3-state	Input port
	2					
	3					
P8 <sub>6</sub> to P8 <sub>0</sub>	1	3-state	3-state	Prev. state*	Prev. state	I/O port
	2					
	3					
P9 <sub>7</sub> /WAIT	1	3-state	3-state	3-state/ Prev. state*	3-state/ Prev. state	WAIT/ I/O port
	2					
	3			Prev. state*	Prev. state	I/O port
P9 <sub>6</sub> /Ø	1	Clock output	3-state	High	Clock output	Clock output
	2					
	3	3-state		High if DDR = 1, 3-state if DDR = 0	Clock output if DDR = 1, 3-state if DDR = 0	Clock output if DDR = 1, input port if DDR = 0
P9 <sub>5</sub> to P9 <sub>3</sub> , AS, WR, RD	1	High	3-state	High	High	AS, WR, RD
	2					
	3	3-state		Prev. state	Prev. state	I/O port
P9 <sub>2</sub> to P9 <sub>0</sub>	1	3-state	3-state	Prev. state	Prev. state	I/O port
	2					
	3					

Notes: 1. 3-state: High-impedance state

2. Prev. state: Previous state. Input ports are in the high-impedance state (with the MOS pull-up on if DDR = 0, PCR = 1). Output ports hold their previous output level.
3. I/O port: Direction depends on the data direction (DDR) bit. Note that these pins may also be used by the on-chip supporting modules.  
See section 7, I/O Ports, for further information.

\* On-chip supporting modules are initialized, so these pins revert to I/O ports according to the DDR and DR bits.

**Table D-1 Port States (cont)**

Pin Name	Mode	Reset	Hardware Standby	Software Standby	Sleep Mode	Normal Operation
PA <sub>7</sub> to PA <sub>0</sub>	1	3-state	3-state	Prev. state*	Prev. state	I/O port
	2					
	3					
PB <sub>7</sub> to PB <sub>0</sub>	1	3-state	3-state	Prev. state*	Prev. state	I/O port
	2					
	3					

Notes: 1. 3-state: High-impedance state

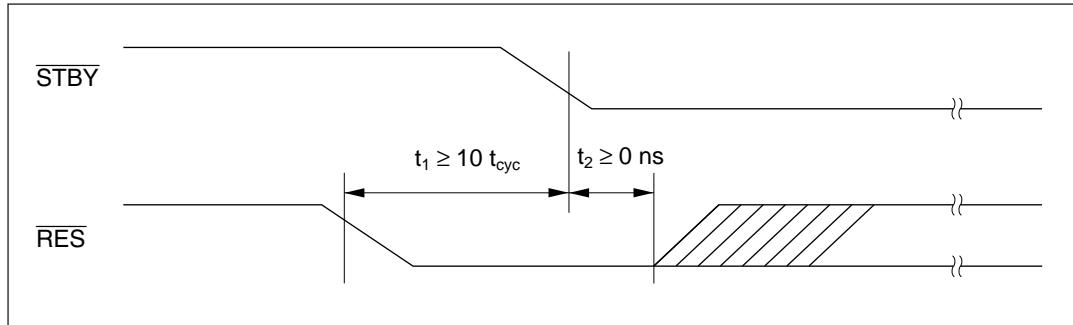
2. Prev. state: Previous state. Input ports are in the high-impedance state (with the MOS pull-up on if DDR = 0, PCR = 1). Output ports hold their previous output level.
3. I/O port: Direction depends on the data direction (DDR) bit. Note that these pins may also be used by the on-chip supporting modules.  
See section 7, I/O Ports, for further information.

\* On-chip supporting modules are initialized, so these pins revert to I/O ports according to the DDR and DR bits.

## Appendix E Timing of Transition to and Recovery from Hardware Standby Mode

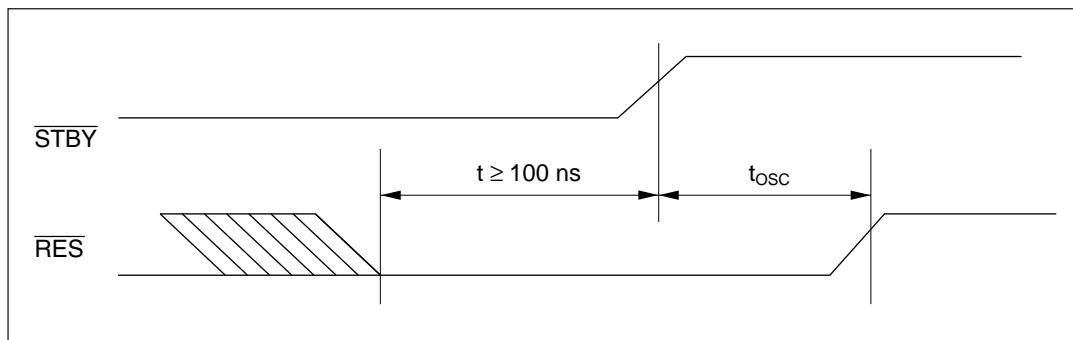
### Timing of Transition to Hardware Standby Mode

(1) To retain RAM contents when the RAME bit in SYSCR is set to 1, drive the  $\overline{\text{RES}}$  signal low 10 system clock cycles before the  $\overline{\text{STBY}}$  signal goes low, as shown below.  $\overline{\text{RES}}$  must remain low until  $\overline{\text{STBY}}$  goes low (minimum delay from  $\overline{\text{STBY}}$  low to  $\overline{\text{RES}}$  high: 0 ns).



(2) When the RAME bit in SYSCR is cleared to 0 or when it is not necessary to retain RAM contents,  $\overline{\text{RES}}$  does not have to be driven low as in (1).

**Timing of Recovery From Hardware Standby Mode:** Drive the  $\overline{\text{RES}}$  signal low approximately 100 ns before  $\overline{\text{STBY}}$  goes high.



## Appendix F Option Lists

### HD6433437, HD6433436, HD6433434 Option List

Please check off the appropriate applications and enter the necessary information.

#### 1 ROM Size

<input type="checkbox"/> HD6433434: 32-kbyte
<input type="checkbox"/> HD6433436: 48-kbyte
<input type="checkbox"/> HD6433437: 60-kbyte

Date of order	
Customer	
Department	
Name	
ROM code name	
LSI number (Hitachi entry)	

#### 2 System Oscillator

<input type="checkbox"/> Ceramic oscillator	f =	MHz
<input type="checkbox"/> External clock	f =	MHz

#### 3 Power Supply Voltage/Maximum Operating Frequency

<input type="checkbox"/> $V_{CC} = 4.5 \text{ V to } 5.5 \text{ V (16 MHz max.)}$
<input type="checkbox"/> $V_{CC} = 4.0 \text{ V to } 5.5 \text{ V (12 MHz max.)}$
<input type="checkbox"/> $V_{CC} = 2.7 \text{ V to } 5.5 \text{ V (10 MHz max.)}$

Notes: 1. Please select the power supply voltage/operating frequency version according to the power supply voltage used.

Example: For use at  $V_{CC} = 4.5 \text{ V to } 5.5 \text{ V}$   $f = 10 \text{ MHz}$ ,  
select  $V_{CC} = 4.5 \text{ V to } 5.5 \text{ V (16 MHz max.)}$ .

2. The power supply voltage and maximum operating frequency of the selected version should also be entered on the Single-Chip Microcomputer Ordering Specifications Sheet.

Continued on the following page.

Continued from the preceding page.

ROM code name	
LSI number (Hitachi entry)	

#### 4 I<sup>2</sup>C Bus Option

<input type="checkbox"/> I <sup>2</sup> C bus used
<input type="checkbox"/> I <sup>2</sup> C bus not used

Notes:

1. The "I<sup>2</sup>C bus used" option includes all cases where data transfer is performed via the SCL and SDA pins using the on-chip I<sup>2</sup>C bus interface function (hardware module). If the I<sup>2</sup>C bus interface function (hardware module) is used, various bus interfaces with different bus specifications and names are also included in "I<sup>2</sup>C bus used". The case in which only the bus drive function of pins PA7 to PA4 in port A is used is not included.
2. When "I<sup>2</sup>C bus not used" is selected, values cannot be set in registers relating to the I<sup>2</sup>C bus interface (ICCR, ICSR, ICDR, ICMR). These register always read H'FF. With emulators, and ZTAT and F-ZTAT versions, the "I<sup>2</sup>C bus used" option is selected. If the "I<sup>2</sup>C bus not used" option is selected, it is essential to ensure that I<sup>2</sup>C bus interface related registers are not accessed.

For the Microcomputer Family item in 1. Basic Specifications in the Single-Chip Microcomputer Ordering Specifications Sheet\*, please specify the appropriate item from the table below according to the combination of items 1 and 4 above. If the "I<sup>2</sup>C bus used" option is selected, please also specify this in Special Specifications (Product Specifications, Mark Specifications, etc.) in 1. Basic Specifications.

\* Please contact the relevant sales department for information on the Single-Chip Microcomputer Ordering Specifications Sheet.

ROM Size \ I <sup>2</sup> C	I <sup>2</sup> C bus used	I <sup>2</sup> C bus not used
32-kbyte	HD6433434W	HD6433434
48-kbyte	HD6433436W	HD6433436
60-kbyte	HD6433437W	HD6433437

# Appendix G Product Code Lineup

**Table G-1 H8/3437 Series Product Code Lineup**

Product Type		Product Code	Mark Code	Package (Hitachi Package Code)
H8/3437	Flash memory version	F-ZTAT	HD64F3437F16	100-pin QFP (FP-100B)
			HD64F3437TF16	100-pin TQFP (TFP-100B)
	PROM version	ZTAT version	HD6473437F16	100-pin QFP (FP-100B)
			HD6473437TF16	100-pin TQFP (TFP-100B)
	Mask ROM version	With I <sup>2</sup> C interface	HD6433437WF	100-pin QFP (FP-100B)
			HD6433437WTF	100-pin TQFP (TFP-100B)
	Without I <sup>2</sup> C interface	With I <sup>2</sup> C interface	HD6433437F	100-pin QFP (FP-100B)
			HD6433437TF	100-pin TQFP (TFP-100B)
H8/3436	Mask ROM version	With I <sup>2</sup> C interface	HD6433436WF	100-pin QFP (FP-100B)
			HD6433436WTF	100-pin TQFP (TFP-100B)
	Without I <sup>2</sup> C interface	With I <sup>2</sup> C interface	HD6433436F	100-pin QFP (FP-100B)
			HD6433436TF	100-pin TQFP (TFP-100B)
H8/3434	Flash memory version	F-ZTAT	HD64F3434F16	100-pin QFP (FP-100B)
			HD64F3434TF16	100-pin TQFP (TFP-100B)
	PROM version	ZTAT version	HD6473434F16	100-pin QFP (FP-100B)
			HD6473434TF16	100-pin TQFP (TFP-100B)
	Mask ROM version	With I <sup>2</sup> C interface	HD6433434WF	100-pin QFP (FP-100B)
			HD6433434WTF	100-pin TQFP (TFP-100B)
	Without I <sup>2</sup> C interface	With I <sup>2</sup> C interface	HD6433434F	100-pin QFP (FP-100B)
			HD6433434TF	100-pin TQFP (TFP-100B)

Note: (\*\*\* in mask ROM versions is the ROM code.

The I<sup>2</sup>C interface is an option. Please note the following points when using this optional function.

- (1) Notify your Hitachi sales representative that you will be using an optional function.
- (2) With mask ROM versions, optional functions can be used if the product code includes the letter W (e.g. HD6433437WF, HD6433434WTF).
- (3) The product code is the same for ZTAT versions, but please be sure to notify Hitachi if you are going to use this optional function.

## Appendix H Package Dimensions

Figure H-1 shows the dimensions of the FP-100B package. Figure H-2 shows the dimensions of the TFP-100B package.

Unit: mm

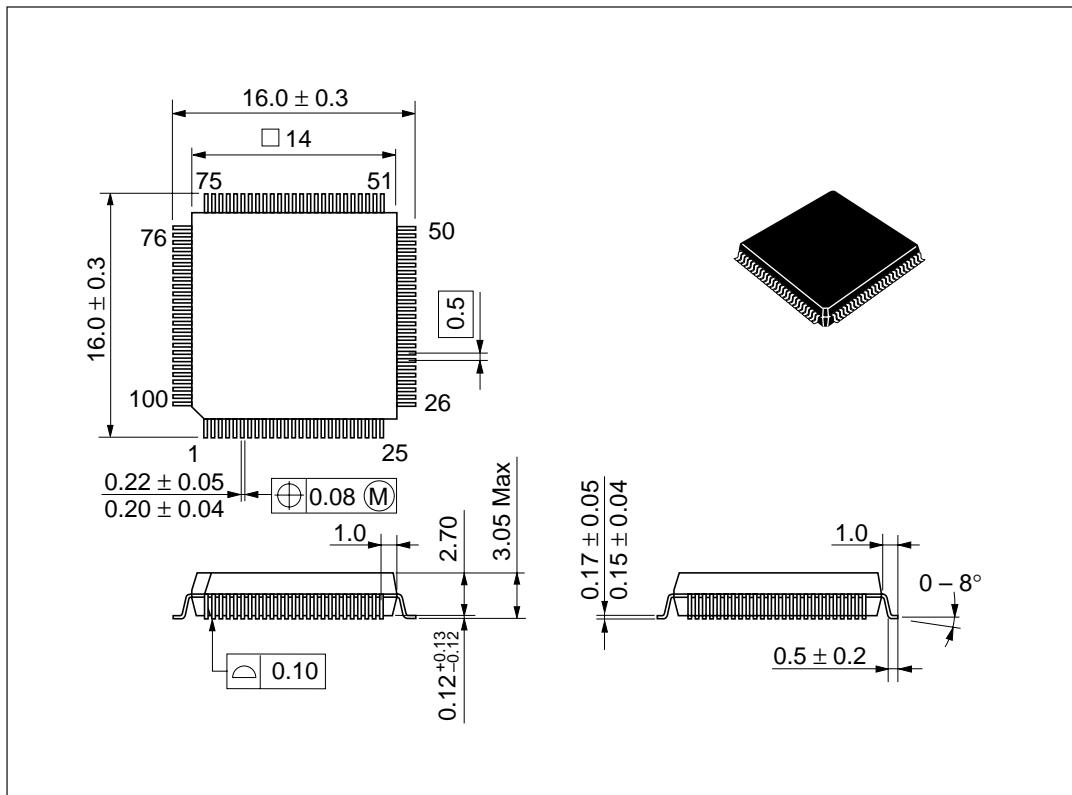
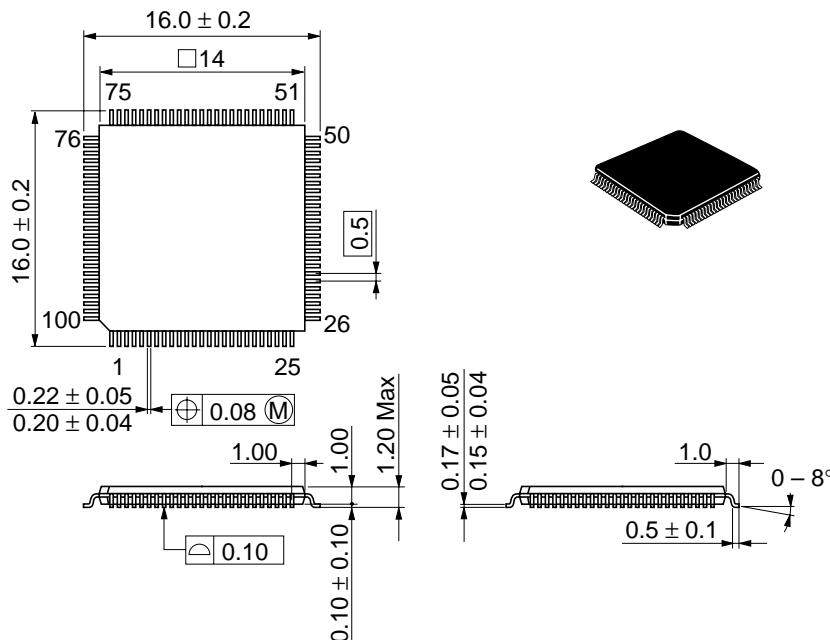


Figure H-1 Package Dimensions (FP-100B)



**Figure H-2 Package Dimensions (TFP-100B)**

---

**H8/3437 Series, H8/3434 F-ZTAT<sup>TM</sup> H8/3437 F-ZTAT<sup>TM</sup>  
Hardware Manual**

Publication Date: 1st Edition, September 1994  
4th Edition, December 1996

Published by: Semiconductor and IC Div.  
Hitachi, Ltd.

Edited by: Technical Documentation Center  
Hitachi Microcomputer System Ltd.

Copyright © Hitachi, Ltd., 1994. All rights reserved. Printed in Japan.