To all our customers

Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: http://www.renesas.com

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.

H8/3627 Series

H8/3627	HD6433627, HD6473627
H8/3626	HD6433626
H8/3625	HD6433625
H8/3624S	HD6433624S
H8/3623S	HD6433623S
H8/3622S	HD6433622S

Hardware Manual

ADE-602-174
Rev. 1.0
3/5/03
Hitachi, Ltd.
MC-Setsu

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.

2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.

3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.

4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.

5. This product is not designed to be radiation resistant.

6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.

7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

Preface

The H8/300L Series of single-chip microcomputers has a high-speed H8/300L CPU core, with many necessary peripheral system functions on-chip. The H8/300L CPU instruction set is compatible with the H8/300 CPU.

On-chip peripheral functions of the H8/3627 Series include a high-precision DTMF generator for tone dialing, three types of timers, two serial communication interface channels, and an A/D converter.

This manual describes the hardware of the H8/3627 Series. For details on the H8/3627 Series instruction set, refer to the *H8/300L Series Programming Manual.*

Contents

Section 1 Overview 1
1.1 Overview 1
1.2 Internal Block Diagram 5
1.3 Pin Arrangement and Functions 6
1.3.1 Pin Arrangement 6
1.3.2 Pin Functions 7

Section 2 CPU 11
2.1 Overview 11
2.1.1 Features 11
2.1.2 Address Space 12
2.1.3 Register Configuration 12
2.2 Register Descriptions 13
2.2.1 General Registers 13
2.2.2 Control Registers 13
2.2.3 Initial Register Values 15
2.3 Data Formats 15
2.3.1 Data Formats in General Registers 16
2.3.2 Memory Data Formats 17
2.4 Addressing Modes 18
2.4.1 Addressing Modes 18
2.4.2 Effective Address Calculation 20
2.5 Instruction Set 24
2.5.1 Data Transfer Instructions 26
2.5.2 Arithmetic Operations 28
2.5.3 Logic Operations 29
2.5.4 Shift Operations 29
2.5.5 Bit Manipulations 31
2.5.6 Branching Instructions 35
2.5.7 System Control Instructions 37
2.5.8 Block Data Transfer Instruction 38
2.6 Basic Operational Timing 40
2.6.1 Access to On-Chip Memory (RAM, ROM) 40
2.6.2 Access to On-Chip Peripheral Modules 41
2.7 CPU States 42
2.7.1 Overview 42
2.7.2 Program Execution State 44
2.7.3 Program Halt State 44
2.7.4 Exception-Handling State 44

2.8 Memory Map 45
2.9 Application Notes 46
2.9.1 Notes on Data Access 46
2.9.2 Notes on Bit Manipulation 48
2.9.3 Notes on Use of the EEPMOV Instruction 54

Section 3 Exception Handling 55
3.1 Overview 55
3.2 Reset 55
3.2.1 Overview 55
3.2.2 Reset Sequence 55
3.2.3 Interrupt Immediately after Reset 56
3.3 Interrupts 57
3.3.1 Overview 57
3.3.2 Interrupt Control Registers 59
3.3.3 External Interrupts 67
3.3.4 Internal Interrupts 67
3.3.5 Interrupt Operations 68
3.3.6 Interrupt Response Time 73
3.4 Application Notes 74
3.4.1 Notes on Stack Area Use 74
3.4.2 Notes on Rewriting Port Mode Registers 74

Section 4 Clock Pulse Generators 77
4.1 Overview 77
4.1.1 Block Diagram 77
4.1.2 System Clock and Subclock 77
4.2 System Clock Generator 78
4.3 Subclock Generator 81
4.4 Prescalers 82
4.5 Note on Oscillators 82

Section 5 Power-Down Modes 85
5.1 Overview 85
5.1.1 System Control Registers 88
5.2 Sleep Mode 91
5.2.1 Transition to Sleep Mode 91
5.2.2 Clearing Sleep Mode 91
5.3 Standby Mode 92
5.3.1 Transition to Standby Mode 92
5.3.2 Clearing Standby Mode 92
5.3.3 Oscillator Settling Time after Standby Mode is Cleared 93
5.3.4 Transition to Standby Mode and Pin States 94

5.4 Watch Mode 95
5.4.1 Transition to Watch Mode 95
5.4.2 Clearing Watch Mode 95
5.4.3 Oscillator Settling Time after Watch Mode is Cleared 95
5.5 Subsleep Mode 96
5.5.1 Transition to Subsleep Mode 96
5.5.2 Clearing Subsleep Mode 96
5.6 Subactive Mode 97
5.6.1 Transition to Subactive Mode 97
5.6.2 Clearing Subactive Mode 97
5.6.3 Operating Frequency in Subactive Mode 97
5.7 Active (medium-speed) Mode 98
5.7.1 Transition to Active (medium-speed) Mode 98
5.7.2 Clearing Active (medium-speed) Mode 98
5.7.3 Operating Frequency in Active (medium-speed) Mode 98
5.8 Direct Transfer 99
5.8.1 Overview 99
5.8.2 Direct Transfer Time 100

Section 6 ROM 103
6.1 Overview 103
6.1.1 Block Diagram 103
6.2 PROM Mode 104
6.2.1 Selection of PROM Mode 104
6.2.2 Socket Adapter Pin Arrangement and Memory Map 104
6.3 Programming 107
6.3.1 Programming and Verification 108
6.3.2 Programming Precautions 112
6.4 Reliability of Programmed Data 113

Section 7 RAM 115
7.1 Overview 115
7.1.1 Block Diagram 115

Section 8 I/O Ports 117
8.1 Overview 117
8.2 Port 1 119
8.2.1 Overview 119
8.2.2 Register Configuration and Description 119
8.2.3 Pin Functions 123
8.2.4 Pin States 124
8.2.5 MOS Input Pull-Up 125
8.3 Port 2 126

8.3.1 Overview 126
8.3.2 Register Configuration and Description 126
8.3.3 Pin Functions 131
8.3.4 Pin States 133
8.3.5 MOS Input Pull-Up 133
8.4 Port 5 134
8.4.1 Overview 134
8.4.2 Register Configuration and Description 134
8.4.3 Pin Functions 136
8.4.4 Pin States 137
8.4.5 MOS Input Pull-Up 137
8.5 Port 6 138
8.5.1 Overview 138
8.5.2 Register Configuration and Description 138
8.5.3 Pin Functions 140
8.5.4 Pin States 140
8.5.5 MOS Input Pull-Up 140
8.6 Port 7 141
8.6.1 Overview 141
8.6.2 Register Configuration and Description 141
8.6.3 Pin Functions 143
8.6.4 Pin States 143
8.7 Port 8 144
8.7.1 Overview 144
8.7.2 Register Configuration and Description 144
8.7.3 Pin Functions 146
8.7.4 Pin States 146
8.8 Port A 147
8.8.1 Overview 147
8.8.2 Register Configuration and Description 147
8.8.3 Pin Functions 149
8.8.4 Pin States 149
8.9 Port B 150
8.9.1 Overview 150
8.9.2 Register Configuration and Description 150

Section 9 Timers 151
9.1 Overview 151
9.2 Timer A 152
9.2.1 Overview 152
9.2.2 Register Descriptions 153
9.2.3 Timer Operation 155
9.2.4 Timer A Operation States 156

9.3 Timer F 157
9.3.1 Overview 157
9.3.2 Register Descriptions 159
9.3.3 Interface with the CPU 166
9.3.4 Timer Operation 169
9.3.5 Application Notes 171
9.4 Timer G 173
9.4.1 Overview 173
9.4.2 Register Descriptions 175
9.4.3 Noise Canceller Circuit 179
9.4.4 Timer Operation 180
9.4.5 Application Notes 184
9.4.6 Sample Timer G Application 188

Section 10 Serial Communication Interface 189
10.1 Overview 189
10.2 SCI1 189
10.2.1 Overview 189
10.2.2 Register Descriptions 191
10.2.3 Operation 195
10.2.4 Interrupt Sources 197
10.3 SCI3 198
10.3.1 Overview 198
10.3.2 Register Descriptions 200
10.3.3 Operation 217
10.3.4 Operation in Asynchronous Mode 222
10.3.5 Operation in Synchronous Mode 230
10.3.6 Multiprocessor Communication Function 238
10.3.7 Interrupts 244
10.3.8 Application Notes 245

Section 11 DTMF Generator 249
11.1 Overview 249
11.1.1 Features 250
11.1.2 Block Diagram 251
11.1.3 Pin Configuration 252
11.1.4 Register Configuration 252
11.2 Register Descriptions 253
11.2.1 DTMF Control Register (DTCR) 253
11.2.2 DTMF Load Register (DTLR) 255
11.3 Operation 256
11.3.1 Output Waveform 256
11.3.2 Operation Flow 257

11.4 Typical Use 258
11.5 Application Notes 258

Section 12 A/D Converter 259
12.1 Overview 259
12.1.1 Features 259
12.1.2 Block Diagram 260
12.1.3 Pin Configuration 260
12.1.4 Register Configuration 261
12.2 Register Descriptions 261
12.2.1 A/D Result Register (ADRR) 261
12.2.2 A/D Mode Register (AMR) 262
12.2.3 A/D Start Register (ADSR) 263
12.3 Operation 264
12.3.1 A/D Conversion Operation 264
12.3.2 Start of A/D Conversion by External Trigger Input 264
12.4 Interrupts 265
12.5 Typical Use 265
12.6 Application Notes 268

Section 13 Power Supply Circuit 269
13.1 Overview 269
13.2 Internal Power Supply Step-Down Circuit Formats 269

Section 14 Electrical Characteristics 271
14.1 Absolute Maximum Ratings 271
14.2 Electrical Characteristics 272
14.2.1 Power Supply Voltage and Operating Range 272
14.2.2 DC Characteristics 274
14.2.3 AC Characteristics 279
14.2.4 A/D Converter Characteristics 282
14.2.5 DTMF Characteristics 283
14.3 Operation Timing 284
14.4 Output Load Circuits 287

Appendix A CPU Instruction Set 289
A.1 Instructions 289
A.2 Operation Code Map 297
A.3 Number of Execution States 299

Appendix B On-Chip Registers 306
B.1 I/O Registers (1) 306
B.2 I/O Registers (2) 310

Appendix C I/O Port Block Diagrams 344
C.1 Port 1 Block Diagrams 344
C.2 Port 2 Block Diagrams 351
C.3 Port 5 Block Diagram 359
C.4 Port 6 Block Diagram 360
C.5 Port 7 Block Diagram 361
C.6 Port 8 Block Diagram 362
C.7 Port A Block Diagram 363
C.8 Port B Block Diagram 363

Appendix D Port States in the Different Processing States 364

Appendix E Product Line-Up 365

Appendix F Package Dimensions 366

Section 1 Overview

1.1 Overview

The H8/300L Series is a series of single-chip microcomputers (MCU: microcomputer unit), built around the high-speed H8/300L CPU and equipped with peripheral system functions on-chip.

Within the H8/300L Series, the H8/3627 Series of single-chip microcomputers features a high-precision DTMF generator for tone dialing. Other on-chip peripheral functions include three types of timers, two serial communication interface channels, and an A/D converter. The H8/3627 Series includes six models, the H8/3627, H8/3626, H8/3625, H8/3624, H8/3623, and H8/3622, with different amounts of on-chip memory: the H8/3627 has 60 kbytes of ROM and 2 kbytes of RAM; the H8/3626 has 48 kbytes of ROM and 2 kbytes of RAM; the H8/3625 has 40 kbytes of ROM and 2 kbytes of RAM; the H8/3624S has 32 kbytes of ROM and 1 kbyte of RAM; the H8/3623S has 24 kbytes of ROM and 1 kbyte of RAM; and the H8/3622S has 16 kbytes of ROM and 1 kbyte of RAM. In addition, thanks to the improvement of the power supply circuit, low power consumption and low radiation noise have been realized.

The H8/3627 has a ZTAT™* version with user-programmable on-chip PROM.

Table 1.1 summarizes the features of the H8/3627 Series.

Note: * ZTAT™ is a trademark of Hitachi, Ltd.

Table 1.1 Features

Item	Description
CPU	High-speed H8/300L CPU • General-register architecture — General registers: Sixteen 8-bit registers (can be used as eight 16-bit registers) • Operating speed — Max. operating speed: 5 MHz — Add/subtract: 0.4 μs (operating at ø= 5 MHz) — Multiply/divide: 2.8 μs (operating at ø= 5 MHz) — Can run on 32.768 kHz subclock • Instruction set compatible with H8/300 CPU — Instruction length of 2 bytes or 4 bytes — Basic arithmetic operations between registers — MOV instruction for data transfer between memory and registers • Instruction features — Multiply (8 bits × 8 bits) — Divide (16 bits ÷ 8 bits) — Bit accumulator — Register-indirect designation of bit position
Interrupts	29 interrupt sources • 13 external interrupt sources: IRQ_4 to IRQ_0, WKP_7 to WKP_0 • 16 internal interrupt sources
Clock pulse generators	Two on-chip clock pulse generators • System clock pulse generator: 1 MHz to 10 MHz • Subclock pulse generator: 32.768 kHz
Power-down modes	Six power-down modes • Sleep mode • Standby mode • Watch mode • Subsleep mode • Subactive mode • Active (medium-speed) mode

Table 1.1 Features (cont)

Item	Description
Memory	Large on-chip memory • H8/3627: 60-kbyte ROM, 2-kbyte RAM • H8/3626: 48-kbyte ROM, 2-kbyte RAM • H8/3625: 40-kbyte ROM, 2-kbyte RAM • H8/3624S: 32-kbyte ROM, 1-kbyte RAM • H8/3623S: 24-kbyte ROM, 1-kbyte RAM • H8/3622S: 16-kbyte ROM, 1-kbyte RAM
I/O ports	53 I/O ports • I/O pins: 50 • Input pins: 3
Timers	3 on-chip timers • Timer A: 8-bit timer with built-in interval/watch clock time base function — Count-up timer with selection of eight internal clock signals divided from the system clock (ø)* and four clock signals divided from the watch clock ($ø_w$)* • Timer F: 16-bit timer with built-in output compare function — Can be used as two independent 8-bit timers. — Count-up timer with selection of four internal clock signals or event input from external pin — Compare-match function with toggle output • Timer G: 8-bit timer with built-in input capture/interval functions — Count-up timer with selection of four internal clock signals — Input capture function with built-in noise canceller circuit
Serial communication interface	Two serial communication interface channels on chip • SCI1: synchronous serial interface — Choice of 8-bit or 16-bit data transfer • SCI3: 8-bit synchronous or asynchronous serial interface — Built-in function for multiprocessor communication

Note: * ø and $ø_w$ are defined in section 4, Clock Pulse Generators.

Table 1.1 Features (cont)

Item	Description			
A/D converter	8-bit successive-approximations A/D converter using a resistance ladder • 2-channel analog input port • Conversion time: 31/ø, 62/ø or 124/ø per channel			
DTMF generator	Built-in tone dialer supporting OSC clock frequencies from 1.2 MHz to 10 MHz in 400-kHz steps			
Product lineup	**Product Code**			
	Mask ROM Version	**ZTAT™ Version**	**Package**	**ROM/RAM Size**
	HD6433627H	HD6473627H	64-pin QFP (FP-64A)	ROM: 60 kbytes RAM: 2 kbytes
	HD6433627FP	HD6473627FP	64-pin LQFP (FP-64E)	
	HD6433626H	—	64-pin QFP (FP-64A)	ROM: 48 kbytes RAM: 2 kbytes
	HD6433626FP	—	64-pin LQFP (FP-64E)	
	HD6433625H	—	64-pin QFP (FP-64A)	ROM: 40 kbytes RAM: 2 kbytes
	HD6433625FP	—	64-pin LQFP (FP-64E)	
	HD6433624SH	—	64-pin QFP (FP-64A)	ROM: 32 kbytes RAM: 1 kbyte
	HD6433624SFP	—	64-pin LQFP (FP-64E)	
	HD6433623SH	—	64-pin QFP (FP-64A)	ROM: 24 kbytes RAM: 1 kbyte
	HD6433623SFP	—	64-pin LQFP (FP-64E)	
	HD6433622SH	—	64-pin QFP (FP-64A)	ROM: 16 kbytes RAM: 1 kbyte
	HD6433622SFP	—	64-pin LQFP (FP-64E)	

1.2 Internal Block Diagram

Figure 1.1 shows a block diagram of the H8/3627 Series.

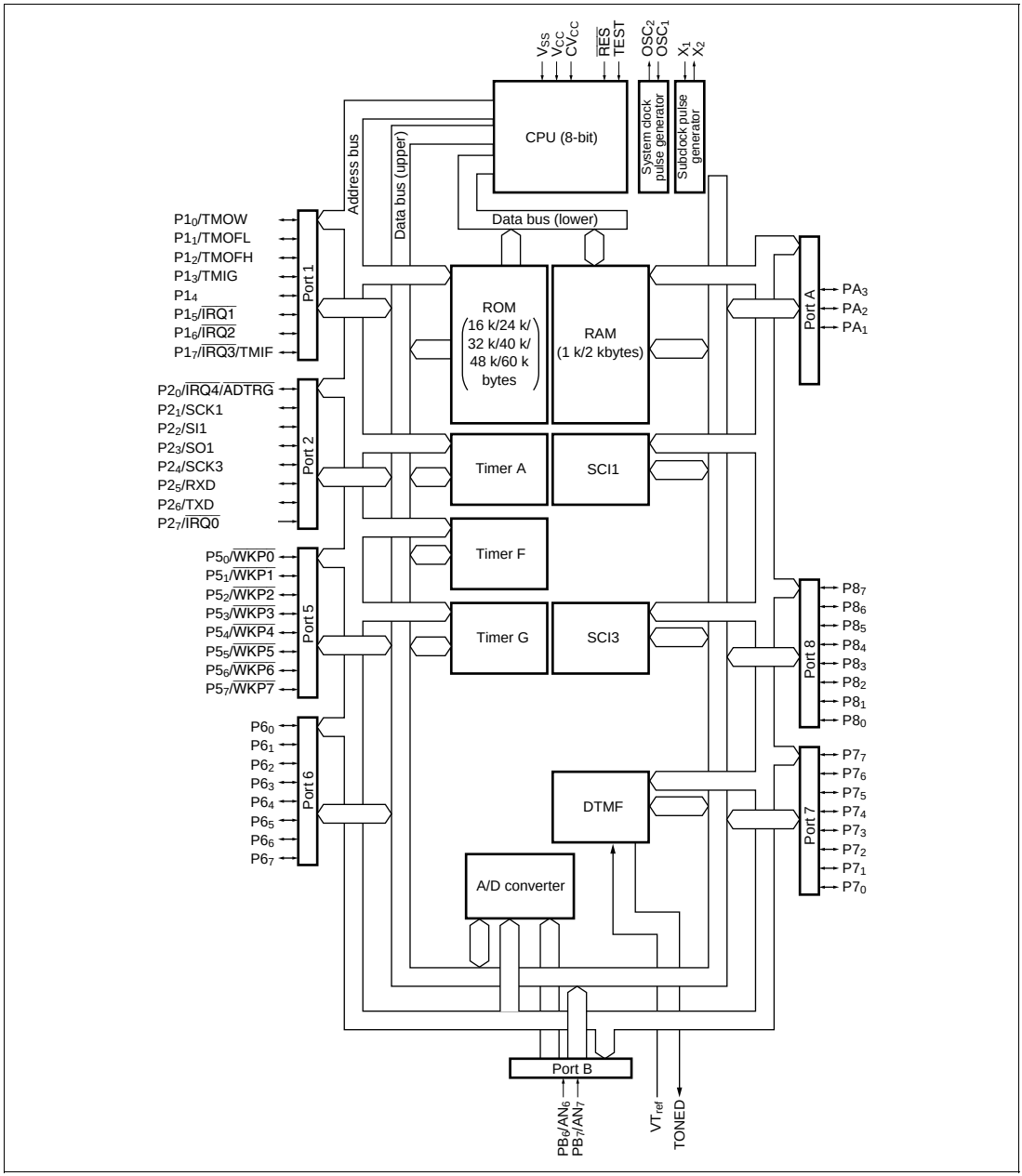


Figure 1.1 Block Diagram

1.3 Pin Arrangement and Functions

1.3.1 Pin Arrangement

The H8/3627 Series pin arrangement is shown in figure 1.2.

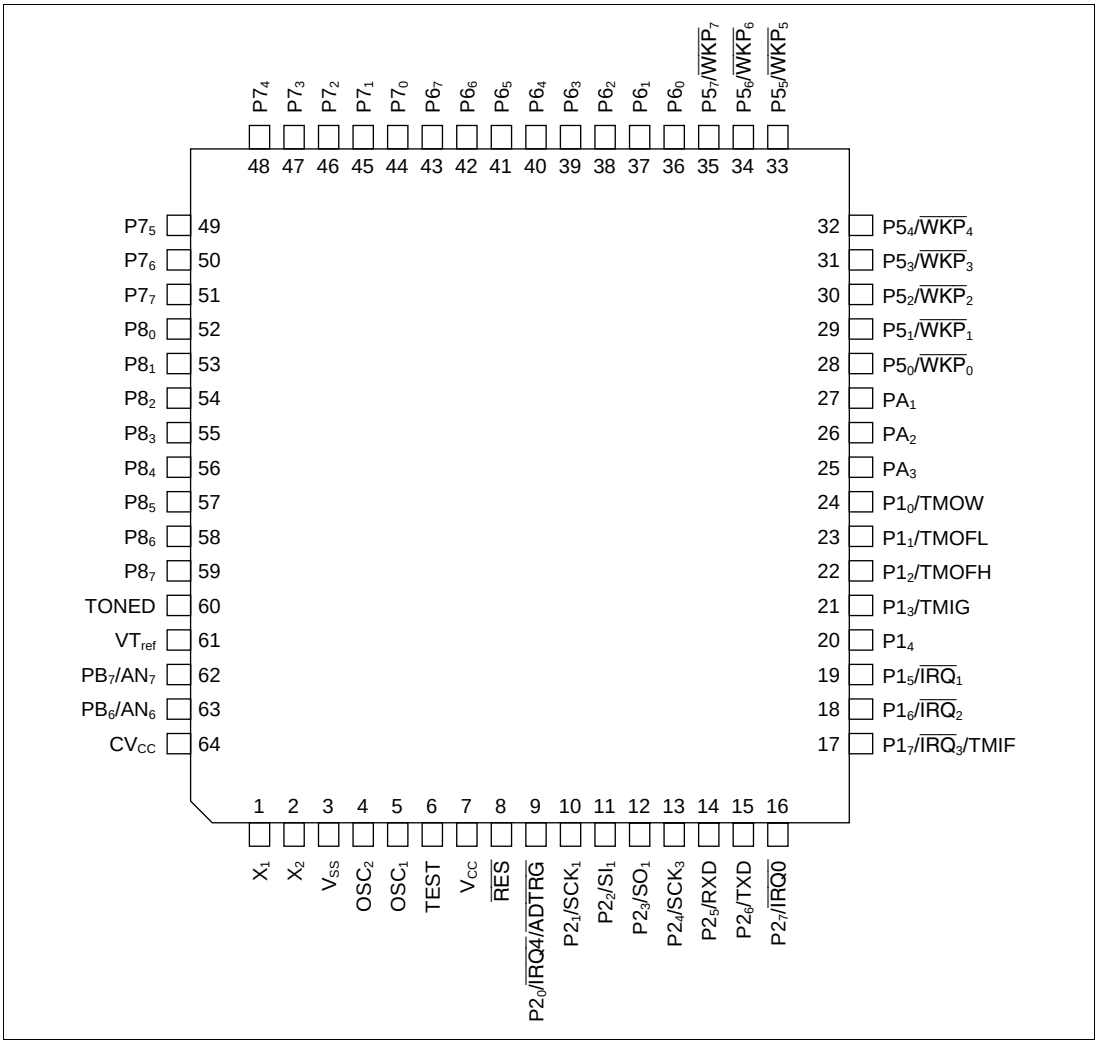


Figure 1.2 Pin Arrangement (FP-64A, FP-64E: Top View)

1.3.2 Pin Functions

Table 1.2 outlines the pin functions.

Table 1.2 Pin Functions

Type	Symbol	Pin No. FP-64A FP-64E	I/O	Name and Functions
Power source pins	V_{CC}	7	Input	**Power supply:** All V_{CC} pins should be connected to the system power supply (+5 V)
	V_{SS}	3	Input	**Ground:** All V_{SS} pins should be connected to the system power supply (0 V)
	CV_{CC}	64	Input	Connected a 0.1 µF stabilization capacitor between the CV_{CC} pin and ground.
	VT_{ref}	61	Input	**DTMF generator reference level:** This is a power supply pin for the reference level for DTMF.
Clock pins	OSC_1	5	Input	**System clock:** These pins connect to a crystal or ceramic oscillator, or can be used to input external an clock. See section 4, Clock Pulse Generators, for a typical connection diagram.
	OSC_2	4	Output	
	X_1	1	Input	**Subclock:** These pins connect to a 32.768-kHz crystal oscillator. See section 4, Clock Pulse Generators, for a typical connection diagram.
	X_2	2	Output	
System control	$\overline{RES}$	8	Input	**Reset:** When this pin is driven low, the chip is reset
	TEST	6	Input	**Test:** This is a test pin, not for use in applica-tion systems. It should be connected to V_{SS}.
Interrupt pins	$\overline{IRQ_0}$ $\overline{IRQ_1}$ $\overline{IRQ_2}$ $\overline{IRQ_3}$ $\overline{IRQ_4}$	16 19 18 17 9	Input	**External interrupt request 0 to 4:** These are input pins for external interrupts for which there is a choice between rising and falling edge sensing
	$\overline{WKP_7}$ to $\overline{WKP_0}$	35 to 28	Input	**Wakeup interrupt request 0 to 7:** These are input pins for external interrupts that are detected at the falling edge

Table 1.2 Pin Functions (cont)

Type	Symbol	Pin No. FP-64A FP-64E	I/O	Name and Functions
Timer pins	TMOW	24	Output	**Clock output:** This is an output pin for wave-forms generated by the timer A output circuit
	TMIF	17	Input	**Timer F event counter input:** This is an event input pin for input to the timer F counter
	TMOFL	23	Output	**Timer FL output:** This is an output pin for waveforms generated by the timer FL output compare function
	TMOFH	22	Output	**Timer FH output:** This is an output pin for waveforms generated by the timer FH output compare function
	TMIG	21	Input	**Timer G capture input:** This is an input pin for the timer G input capture function
I/O ports	PB_7, PB_6	62 to 63	Input	**Port B:** This is a 2-bit input port
	PA_3 to PA_1	25 to 27	I/O	**Port A:** This is a 3-bit I/O port. Input or output can be designated for each bit by means of port control register A (PCRA).
	$P1_7$ to $P1_0$	17 to 24	I/O	**Port 1:** This is an 8-bit I/O port. Input or output can be designated for each bit be means of port control register 1 (PCR1).
	$P2_7$	16	Input	**Port 2 (bit 7):** This is a 1-bit input port.
	$P2_6$ to $P2_0$	15 to 9	I/O	**Port 2 (bits 6 to 0):** This is a 7-bit I/O port. Input or output can be designated for each bit by means of port control register 2 (PCR2).
	$P5_7$ to $P5_0$	35 to 28	I/O	**Port 5:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 5 (PCR5).
	$P6_7$ to $P6_0$	43 to 36	I/O	**Port 6:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 6 (PCR6).
	$P7_7$ to $P7_0$	51 to 44	I/O	**Port 7:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 7 (PCR7).
	$P8_7$ to $P8_0$	59 to 52	I/O	**Port 8:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 8 (PCR8).

Table 1.2 Pin Functions (cont)

Type	Symbol	Pin No. FP-64A FP-64E	I/O	Name and Functions
Serial communication interface (SCI)	SI_1	11	Input	**SCI1 receive data input:** This is the SCI1 data input pin
	SO_1	12	Output	**SCI1 send data output:** This is the SCI1 data output pin
	SCK_1	10	I/O	**SCI1 clock I/O :**This is the SCI1 clock I/O pin
	RXD	14	Input	**SCI3 receive data input:** This is the SCI3 data input pin
	TXD	15	Output	**SCI3 send data output:** This is the SCI3 data output pin
	SCK_3	13	I/O	**SCI3 clock I/O:** This is the SCI3 clock I/O pin
A/D converter	AN_7, AN_6	62, 63	Input	**Analog input channels 6, 7:** These are analog data input channels to the A/D converter
	$\overline{ADTRG}$	9	Input	**A/D converter trigger input:** This is the external trigger input pin to the A/D converter
DTMF generator	TONED	60	Output	**DTMF signal:** This is the output pin for the DTMF signal

Section 2 CPU

2.1 Overview

The H8/300L CPU has sixteen 8-bit general registers, which can also be paired as eight 16-bit registers. Its concise, optimized instruction set is designed for high-speed operation.

2.1.1 Features

Features of the H8/300L CPU are listed below.

- General-register architecture
  Sixteen 8-bit general registers, also usable as eight 16-bit general registers
- Instruction set with 55 basic instructions, including:
  — Multiply and divide instructions
  — Powerful bit-manipulation instructions
- Eight addressing modes
  — Register direct
  — Register indirect
  — Register indirect with displacement
  — Register indirect with post-increment or pre-decrement
  — Absolute address
  — Immediate
  — Program-counter relative
  — Memory indirect
- 64-kbyte address space
- High-speed operation
  — All frequently used instructions are executed in two to four states
  — High-speed arithmetic and logic operations
    8- or 16-bit register-register add or subtract: 0.4 µs*
    8 × 8-bit multiply: 2.8 µs*
    16 ÷ 8-bit divide: 2.8 µs*
- Low-power operation modes
  SLEEP instruction for transition to low-power operation

Note: * These values are at ø = 5 MHz.

2.1.2 Address Space

The H8/300L CPU supports an address space of up to 64 kbytes for storing program code and data.

See 2.8, Memory Map, for details of the memory map.

2.1.3 Register Configuration

Figure 2.1 shows the register structure of the H8/300L CPU. There are two groups of registers: the general registers and control registers.

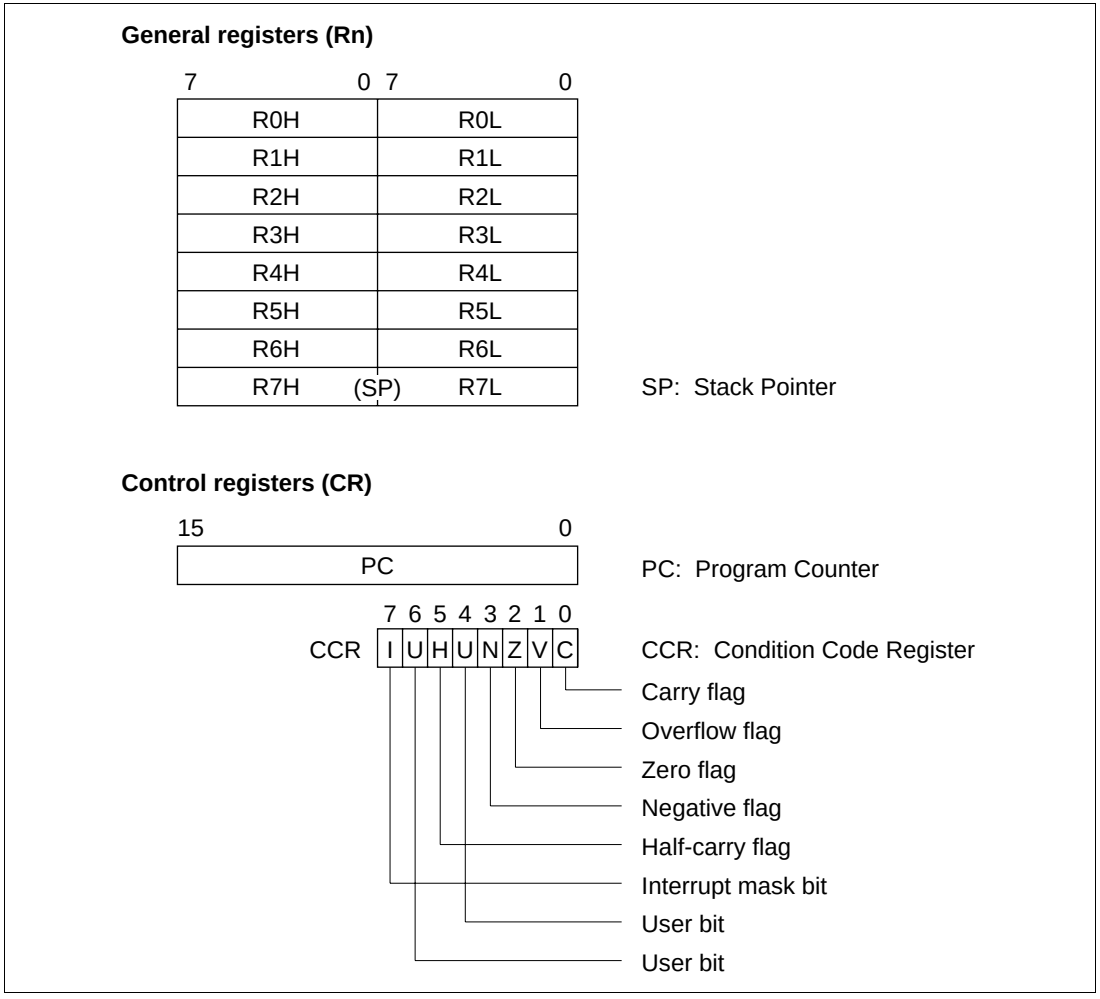


Figure 2.1 CPU Registers

2.2 Register Descriptions

2.2.1 General Registers

All the general registers have the same functions, and can be used as both data registers and address registers.

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

When used as address registers, the general registers are accessed as 16-bit registers (R0 to R7).

R7 also functions as the stack pointer (SP), used implicitly by hardware in exception handling and subroutine calls. When it functions as the stack pointer, as indicated in figure 2.2, SP (R7) points to the top of the stack.

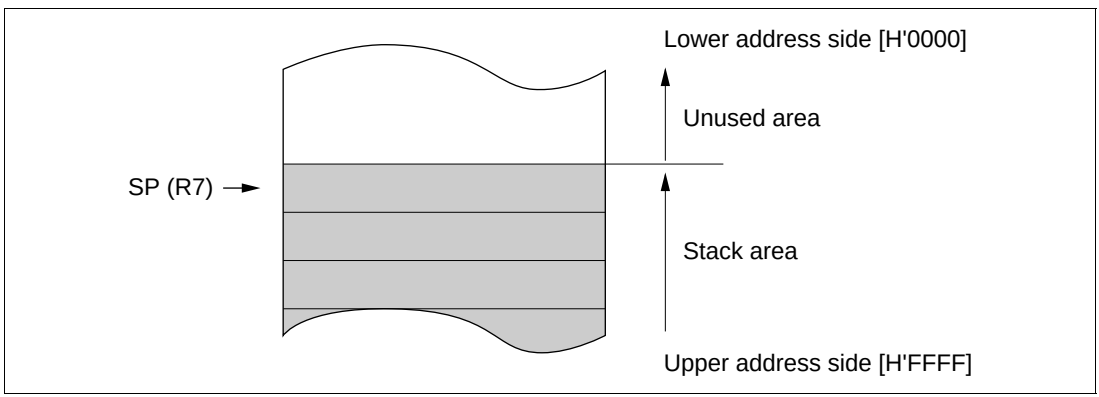


Figure 2.2 Stack Pointer

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. All instructions are fetched 16 bits (1 word) at a time, 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 the interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags. These bits can be read and written by software (using the LDC, STC, ANDC, ORC, and XORC instructions). The N, Z, V, and C flags are used as branching conditions for conditional branching (Bcc) instructions.

Bit 7—Interrupt Mask Bit (I): When this bit is set to 1, interrupts are masked. This bit is set to 1 automatically at the start of exception handling. The interrupt mask bit may be read and written by software. For further details, see 3.3, Interrupts.

Bit 6—User Bit (U): Can be used freely by the user.

Bit 5—Half-Carry Flag (H): When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and is cleared to 0 otherwise.

The H flag is used implicitly by the DAA and DAS instructions.

When the ADD.W, SUB.W, or CMP.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and is cleared to 0 otherwise.

Bit 4—User Bit (U): Can be used freely by the user.

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

Bit 2—Zero Flag (Z): Set to 1 to indicate a zero result, and cleared to 0 to indicate a non-zero result.

Bit 1—Overflow Flag (V): Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.

Bit 0—Carry Flag (C): Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:

- Add instructions, to indicate a carry
- Subtract instructions, to indicate a borrow
- Shift/rotate carry

The carry flag is also used as a bit accumulator by bit manipulation instructions.

Some instructions leave some or all of the flag bits unchanged.

Refer to the *H8/300L Series Programming Manual* for the action of each instruction on the flag bits.

2.2.3 Initial Register Values

When the CPU is reset, the program counter (PC) is initialized to the value stored at address H'0000 in the vector table, and the I bit 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. R7 initialization should therefore be carried out immediately after a reset.

2.3 Data Formats

The H8/300L 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 in a byte operand (n = 0, 1, 2, ..., 7).

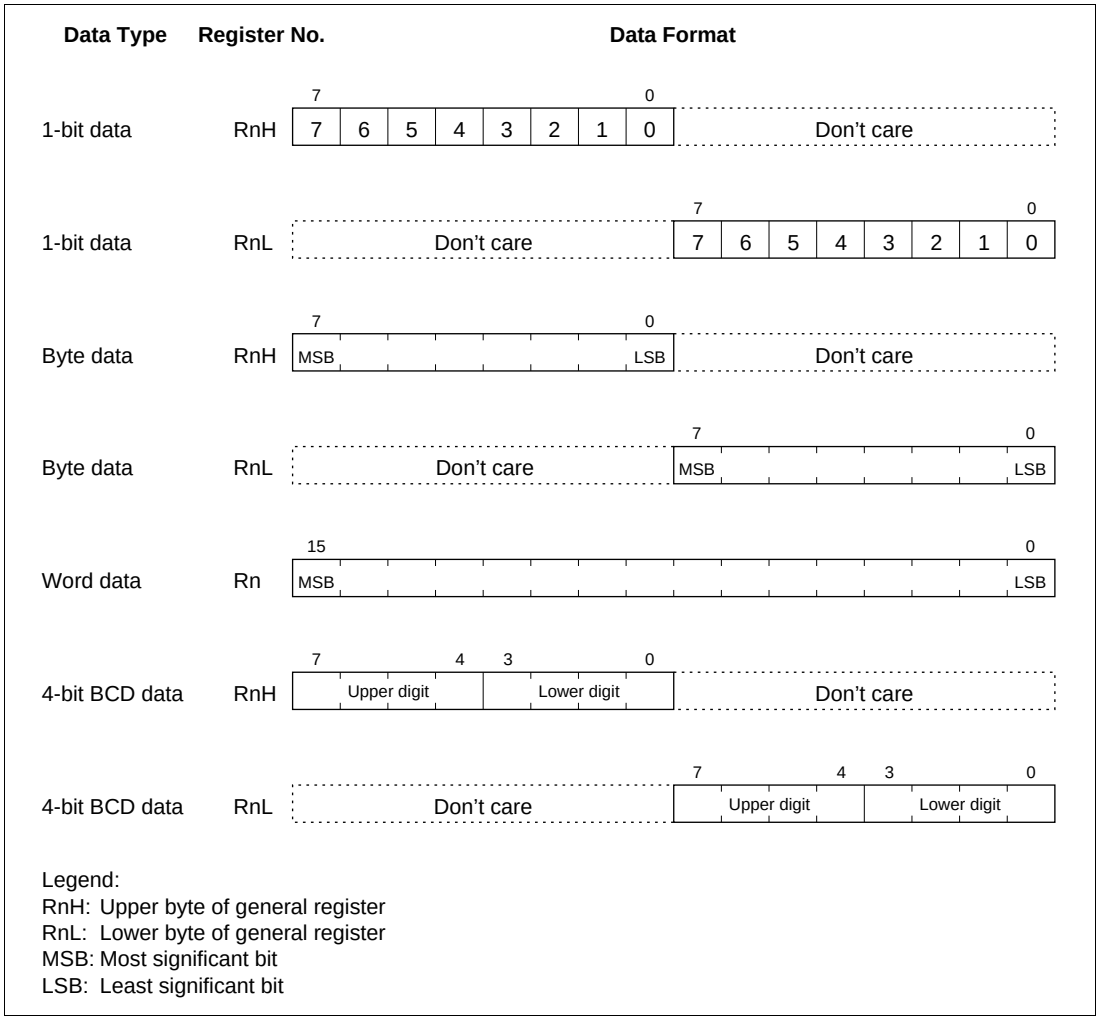
All arithmetic and logic instructions except ADDS and SUBS can operate on byte data.

The MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (8 bits × 8 bits), and DIVXU (16 bits ÷ 8 bits) instructions operate on word data.

The DAA and DAS instructions perform decimal arithmetic adjustments on byte data in two-digit 4-bit BCD form.

2.3.1 Data Formats in General Registers

General register data formats are shown in figure 2.3.



Legend:
RnH: Upper byte of general register
RnL: Lower byte 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. For access by the H8/300L CPU, 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, the access is performed at the preceding even address. This rule affects the MOV.W instruction, and also applies to instruction fetching.

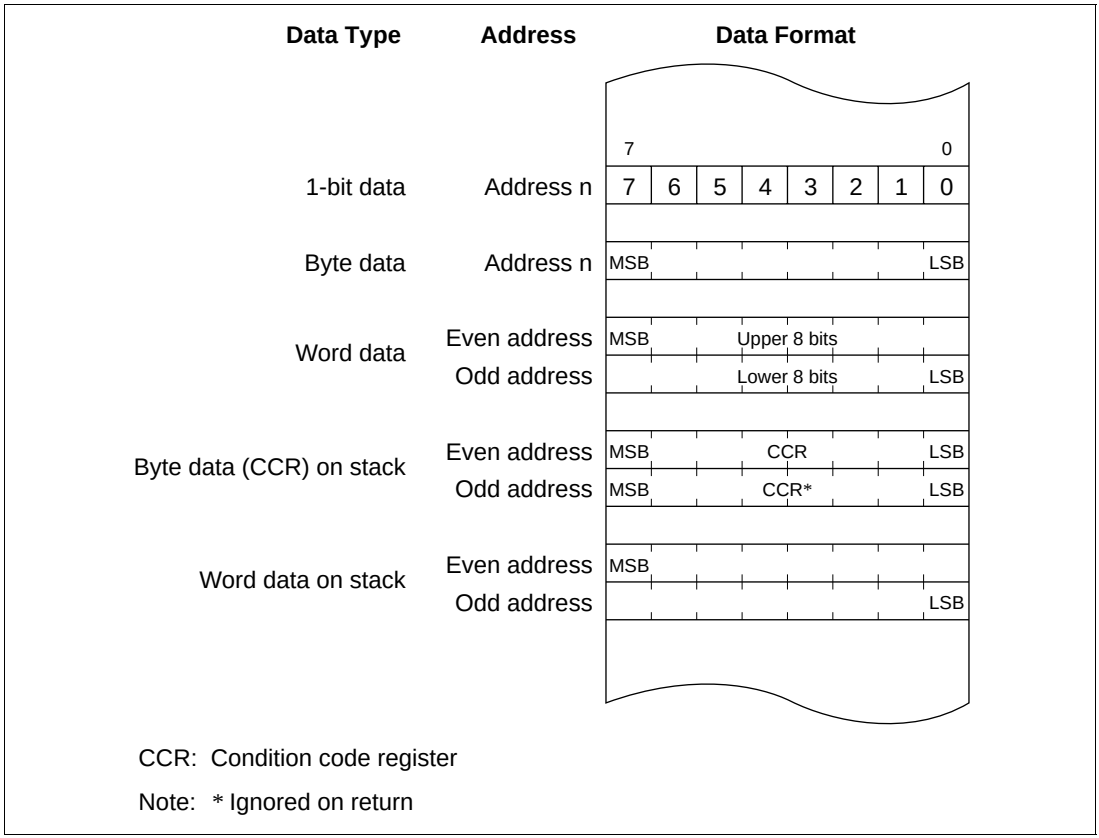


Figure 2.4 Memory Data Formats

When the stack is accessed using R7 as an address register, word access should always be performed. 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 Modes

The H8/300L CPU supports the eight addressing modes listed in table 2.1. Each instruction uses a subset of these addressing modes.

Table 2.1 Addressing Modes

No.	Address Modes	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.

Only the MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (8 bits × 8 bits), and DIVXU (16 bits ÷ 8 bits) instructions have 16-bit operands.

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

3. Register Indirect with Displacement—@(d:16, Rn): The instruction has a second word (bytes 3 and 4) containing a displacement which is added to the contents of the specified general register to obtain the operand address in memory.

This mode is used only in MOV instructions. 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.

  The register field of the instruction specifies a 16-bit general register containing the address of the operand. After the operand is accessed, the register is incremented by 1 for MOV.B or 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.

  The register field of the instruction specifies a 16-bit general register which is decremented by 1 or 2 to obtain the address of the operand in memory. The register retains the decremented value. The size of the decrement is 1 for MOV.B or 2 for MOV.W. For MOV.W, the original contents of the register must be even.

5. Absolute Address—@aa:8 or @aa:16: The instruction specifies the absolute address of the operand in memory.

The absolute address may be 8 bits long (@aa:8) or 16 bits long (@aa:16). The MOV.B and bit manipulation instructions can use 8-bit absolute addresses. The MOV.B, MOV.W, JMP, and JSR instructions can use 16-bit absolute addresses.

For an 8-bit absolute address, the upper 8 bits are assumed to be 1 (H'FF). The address range is H'FF00 to H'FFFF (65280 to 65535).

6. Immediate—#xx:8 or #xx:16: The instruction contains an 8-bit operand (#xx:8) in its second byte, or a 16-bit operand (#xx:16) 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 in the second or fourth byte of the instruction, specifying a bit number.

7. Program-Counter Relative—@(d:8, PC): This mode is used in the Bcc and BSR instructions. An 8-bit displacement in byte 2 of the instruction code is sign-extended to 16 bits and added to the program counter contents to generate a branch destination address. The possible branching range is –126 to +128 bytes (–63 to +64 words) from the current address. The result of the addition should be an even number.

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. The word located at this address contains the branch destination address.

The upper 8 bits of the absolute address are assumed to be 0 (H'00), so the address range is H'0000 to H'00FF (0 to 255). Note that with the H8/300L Series, the lower end of the address area is also used as a vector area. See 3.3, Interrupts, for details on the vector area.

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 2.3.2, Memory Data Formats, for further information.

2.4.2 Effective Address Calculation

Table 2.2 shows how effective addresses are calculated in each of the addressing modes.

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

Data transfer instructions can use all 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 addressing (5) to specify a byte operand, and 3-bit immediate addressing (6) to specify a bit position in that byte. The BSET, BCLR, BNOT, and BTST instructions can also use register direct addressing (1) to specify the bit position.

Table 2.2 Effective Address Calculation

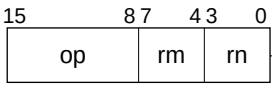
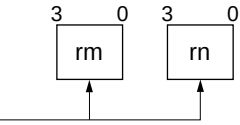
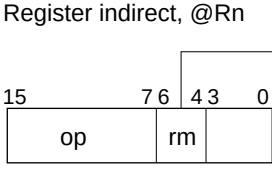
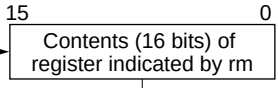
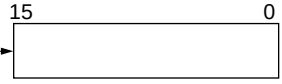
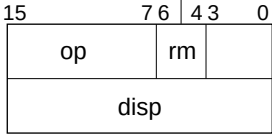
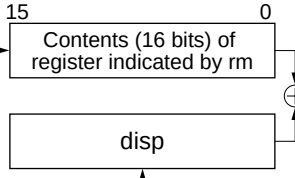
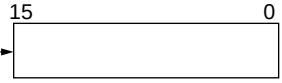
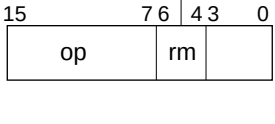
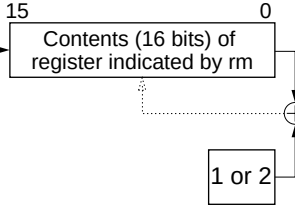
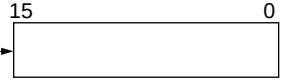
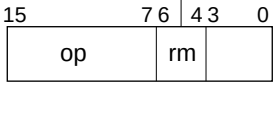
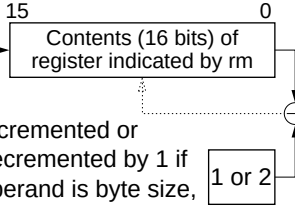
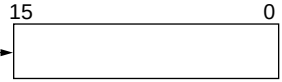
No.	Addressing Mode and Instruction Format	Effective Address Calculation Method	Effective Address (EA)
1	Register direct, Rn		 Operand is contents of registers indicated by rm/rn
2	Register indirect, @Rn	  	
3	Register indirect with displacement, @(d:16, Rn)	  	
4	Register indirect with post-increment, @Rn+ Register indirect with pre-decrement, @–Rn	      Incremented or decremented by 1 if operand is byte size, and by 2 if word size	

Table 2.2 Effective Address Calculation (cont)

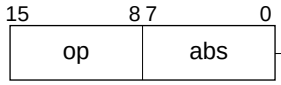
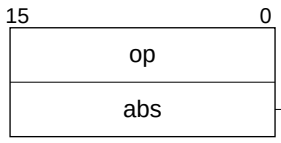
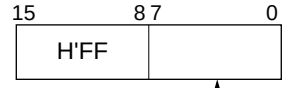
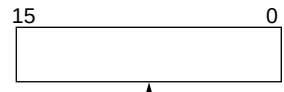
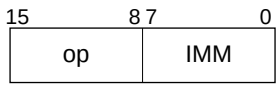
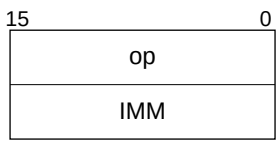
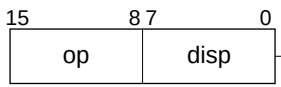
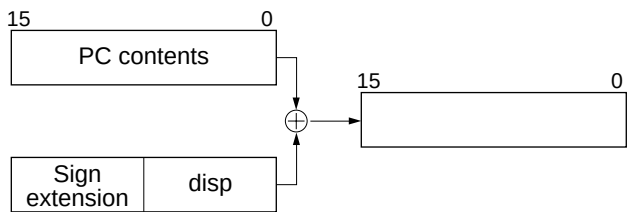
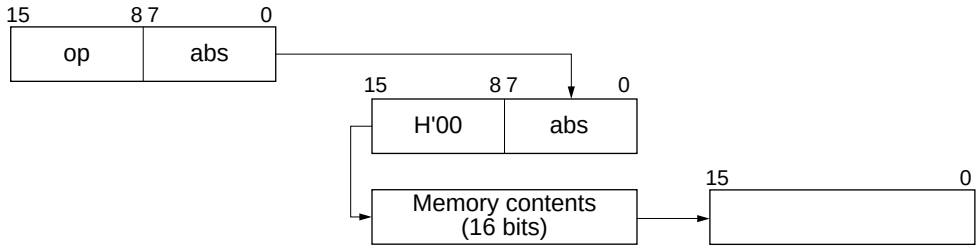
No.	Addressing Mode and Instruction Format	Effective Address Calculation Method	Effective Address (EA)
5	Absolute address @aa:8  @aa:16  	  	
6	Immediate #xx:8  #xx:16  		Operand is 1- or 2-byte immediate data
7	Program-counter relative @(d:8, PC)  	 	

Table 2.2 Effective Address Calculation (cont)

No.	Addressing Mode and Instruction Format	Effective Address Calculation Method	Effective Address (EA)
8	Memory indirect, @@aa:8		

Legend:

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

2.5 Instruction Set

The H8/300L Series can use a total of 55 instructions, which are grouped by function in table 2.3.

Table 2.3 Instruction Set

Function	Instructions	Number
Data transfer	MOV, PUSH[*1], POP[*1]	1
Arithmetic operations	ADD, SUB, ADDX, SUBX, INC, DEC, ADDS, SUBS, DAA, DAS, MULXU, DIVXU, CMP, NEG	14
Logic operations	AND, OR, XOR, NOT	4
Shift	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	14
Branch	Bcc[*2], JMP, BSR, JSR, RTS	5
System control	RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	8
Block data transfer	EEPMOV	1
	Total:	55

Notes: 1. PUSH Rn is equivalent to MOV.W Rn, @–SP.
POP Rn is equivalent to MOV.W @SP+, Rn. The machine language is also the same.
2. Bcc is a conditional branch instruction in which cc represents a condition code.

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

The functions of the instructions are shown in tables 2.4 to 2.11. The meaning of the operation symbols used in the tables is as follows.

Notation

Rd	General register (destination)
Rs	General register (source)
Rn	General register
(EAd), <EAd>	Destination operand
(EAs), <EAs>	Source operand
CCR	Condition code register
N	N (negative) flag of CCR
Z	Z (zero) flag of CCR
V	V (overflow) flag of CCR
C	C (carry) flag of CCR
PC	Program counter
SP	Stack pointer
#IMM	Immediate data
disp	Displacement
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	AND logical
∨	OR logical
⊕	Exclusive OR logical
→	Move
~	Logical negation (logical complement)
:3	3-bit length
:8	8-bit length
:16	16-bit length
( ), < >	Contents of operand indicated by effective address

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: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.
POP	W	@SP+ → Rn Pops a 16-bit general register from the stack. Equivalent to MOV.W @SP+, Rn.
PUSH	W	Rn → @–SP Pushes a 16-bit general register onto the stack. Equivalent to MOV.W Rn, @–SP.

Note: * Size: Operand size
B: Byte
W: Word

Certain precautions are required in data access. See 2.9.1, Notes on Data Access, for details.

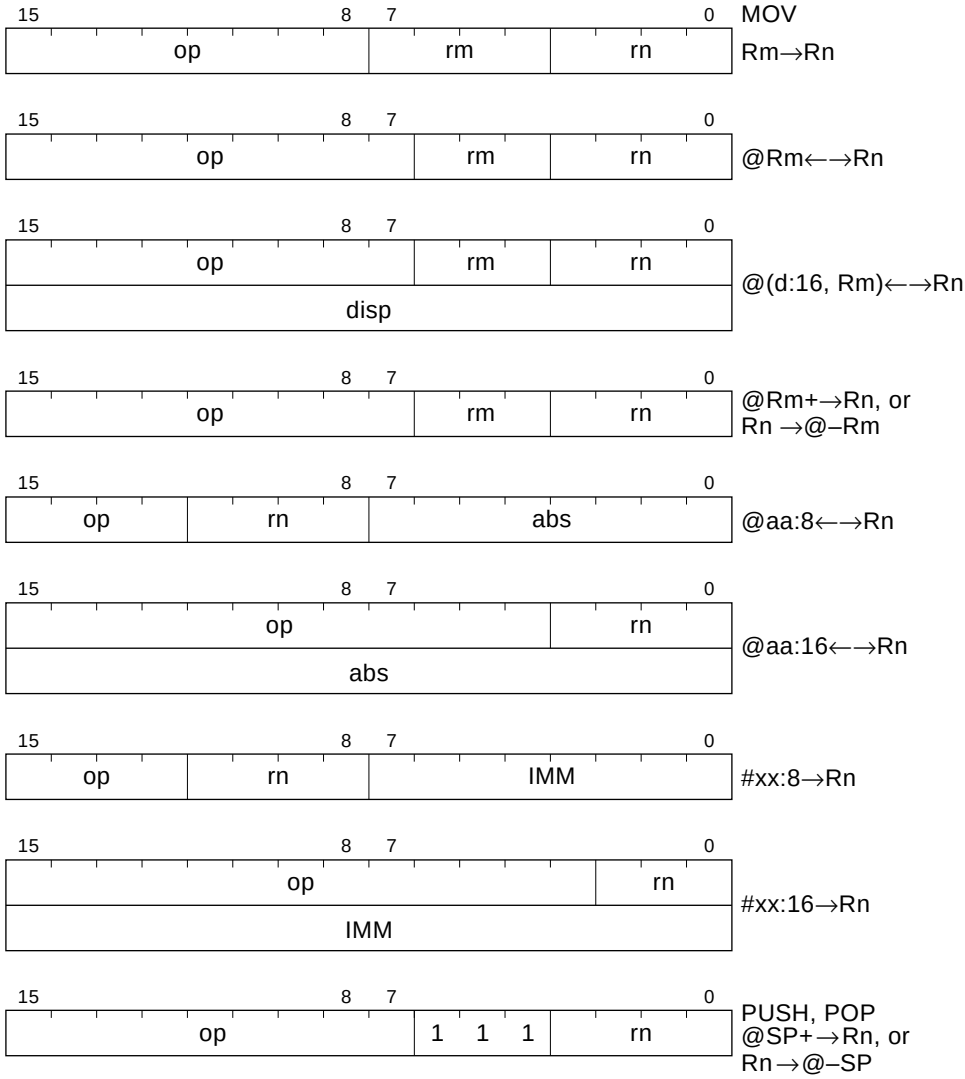


Figure 2.5 Data Transfer Instruction Codes

2.5.2 Arithmetic Operations

Table 2.5 describes the arithmetic instructions.

Table 2.5 Arithmetic Instructions

Instruction	Size*	Function
ADD	B/W	Rd ± Rs → Rd, Rd + #IMM → 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 ± Rs ± C → Rd, Rd ± #IMM ± C → 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 ± 1 → Rd
DEC		Increments or decrements a general register
ADDS	W	Rd ± 1 → Rd, Rd ± 2 → 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 → Rd
DAS		Decimal-adjusts (adjusts to packed 4-bit BCD) an addition or subtraction result in a general register by referring to the CCR
MULXU	B	Rd × Rs → Rd
		Performs 8-bit × 8-bit unsigned multiplication on data in two general registers, providing a 16-bit result
DIVXU	B	Rd ÷ Rs → Rd
		Performs 16-bit ÷ 8-bit unsigned division on data in two general registers, providing an 8-bit quotient and 8-bit remainder
CMP	B/W	Rd – Rs, Rd – #IMM
		Compares data in a general register with data in another general register or with immediate data, and the result is stored in the CCR. Word data can be compared only between two general registers.
NEG	B	0 – Rd → Rd
		Obtains the two's complement (arithmetic complement) of data in a general register

Note: * Size: Operand size
B: Byte
W: Word

2.5.3 Logic Operations

Table 2.6 describes the four instructions that perform logic operations.

Table 2.6 Logic Operation Instructions

Instruction	Size*	Function
AND	B	Rd ∧ Rs → Rd, Rd ∧ #IMM → Rd Performs a logical AND operation on a general register and another general register or immediate data
OR	B	Rd ∨ Rs → Rd, Rd ∨ #IMM → Rd Performs a logical OR operation on a general register and another general register or immediate data
XOR	B	Rd ⊕ Rs → Rd, Rd ⊕ #IMM → Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data
NOT	B	~ Rd → 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.

Table 2.7 Shift Instructions

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

Note: * Size: Operand size
B: Byte

Figure 2.6 shows the instruction code format of arithmetic, logic, and shift instructions.

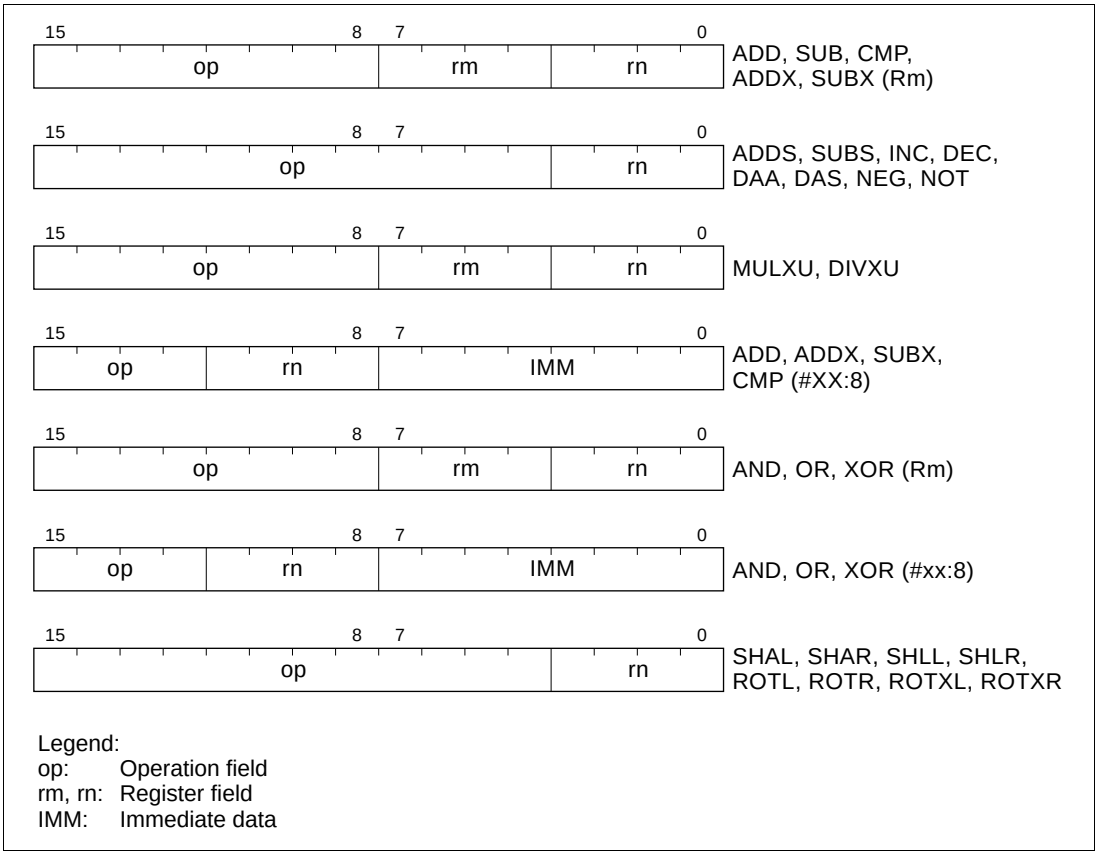


Figure 2.6 Arithmetic, Logic, and Shift Instruction Codes

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

Note: * Size: Operand size
B: Byte

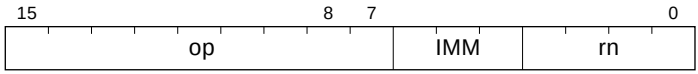
Table 2.8 Bit-Manipulation Instructions (cont)

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

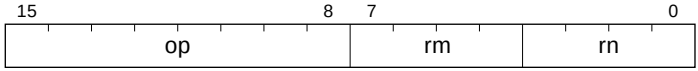
Note: * Size: Operand size
B: Byte

Certain precautions are required in bit manipulation. See 2.9.2, Notes on Bit Manipulation, for details.

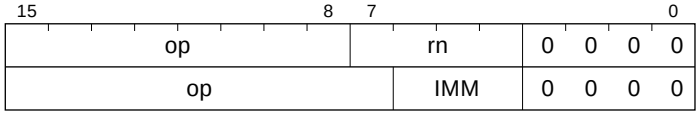
BSET, BCLR, BNOT, BTST



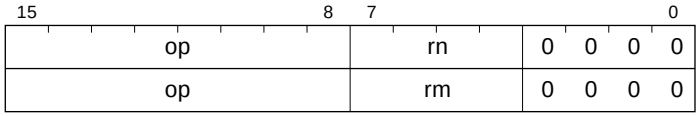
Operand: register direct (Rn)
Bit No.: immediate (#xx:3)



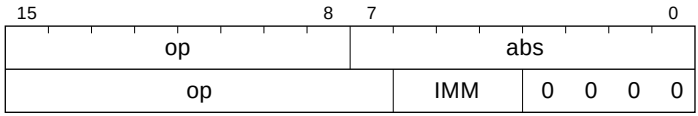
Operand: register direct (Rn)
Bit No.: register direct (Rm)



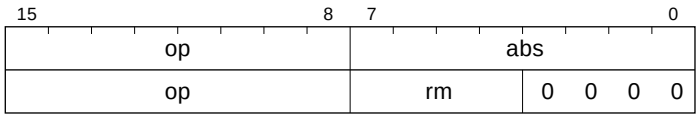
Operand: register indirect (@Rn)
Bit No.: immediate (#xx:3)



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

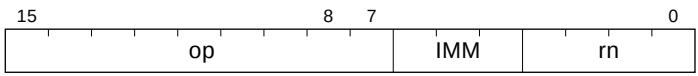


Operand: absolute (@aa:8)
Bit No.: immediate (#xx:3)

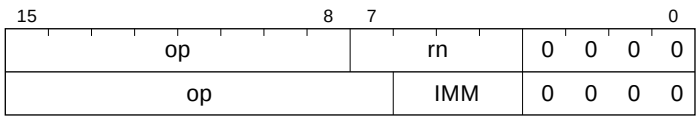


Operand: absolute (@aa:8)
Bit No.: register direct (Rm)

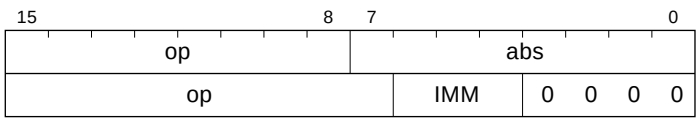
BAND, BOR, BXOR, BLD, BST



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



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



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

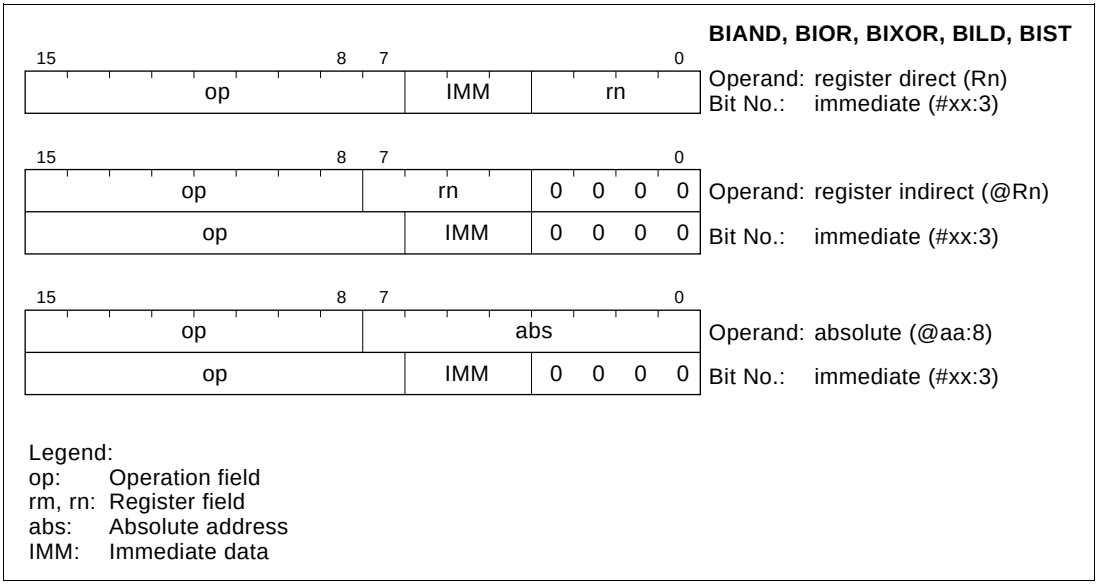


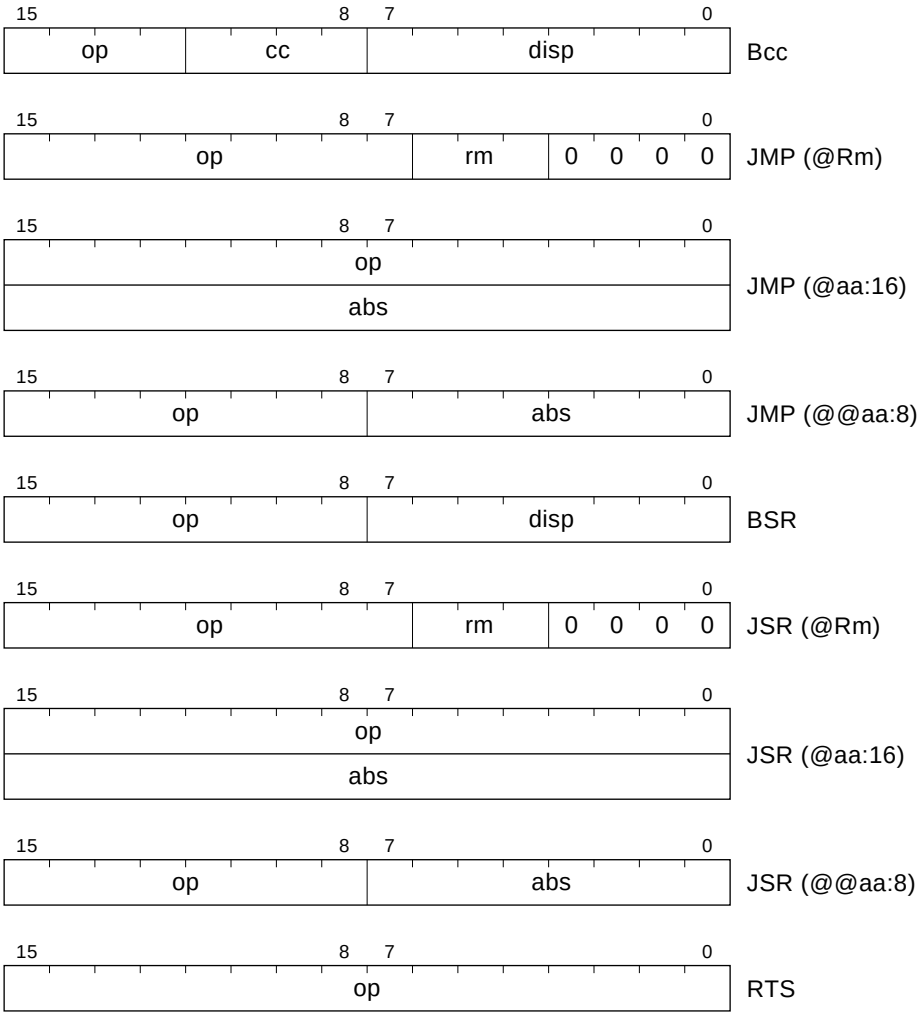
Figure 2.7 Bit Manipulation Instruction Codes (cont)

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		
Bcc	—	Branches to the designated address if the specified condition is true. The branching conditions are given below.		
		Mnemonic	**Description**	**Condition**
		BRA (BT)	Always (true)	Always
		BRN (BF)	Never (false)	Never
		BHI	High	$C \vee Z = 0$
		BLS	Low or same	$C \vee Z = 1$
		BCC (BHS)	Carry clear (high or same)	$C = 0$
		BCS (BLO)	Carry set (low)	$C = 1$
		BNE	Not equal	$Z = 0$
		BEQ	Equal	$Z = 1$
		BVC	Overflow clear	$V = 0$
		BVS	Overflow set	$V = 1$
		BPL	Plus	$N = 0$
		BMI	Minus	$N = 1$
		BGE	Greater or equal	$N \oplus V = 0$
		BLT	Less than	$N \oplus V = 1$
		BGT	Greater than	$Z \vee (N \oplus V) = 0$
		BLE	Less or equal	$Z \vee (N \oplus V) = 1$
JMP	—	Branches unconditionally to a specified address		
BSR	—	Branches to a subroutine at a specified address		
JSR	—	Branches to a subroutine at a specified 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 from active mode to a power-down mode. See section 5, Power-Down Modes, for details
LDC	B	Rs → CCR, #IMM → CCR Moves immediate data or general register contents to the condition code register
STC	B	CCR → Rd Copies the condition code register to a specified general register
ANDC	B	CCR ∧ #IMM → CCR Logically ANDs the condition code register with immediate data
ORC	B	CCR ∨ #IMM → CCR Logically ORs the condition code register with immediate data
XORC	B	CCR ⊕ #IMM → CCR Logically exclusive-ORs the condition code register with immediate data
NOP	—	PC + 2 → PC Only increments the program counter

Note: * Size: Operand size
B: Byte

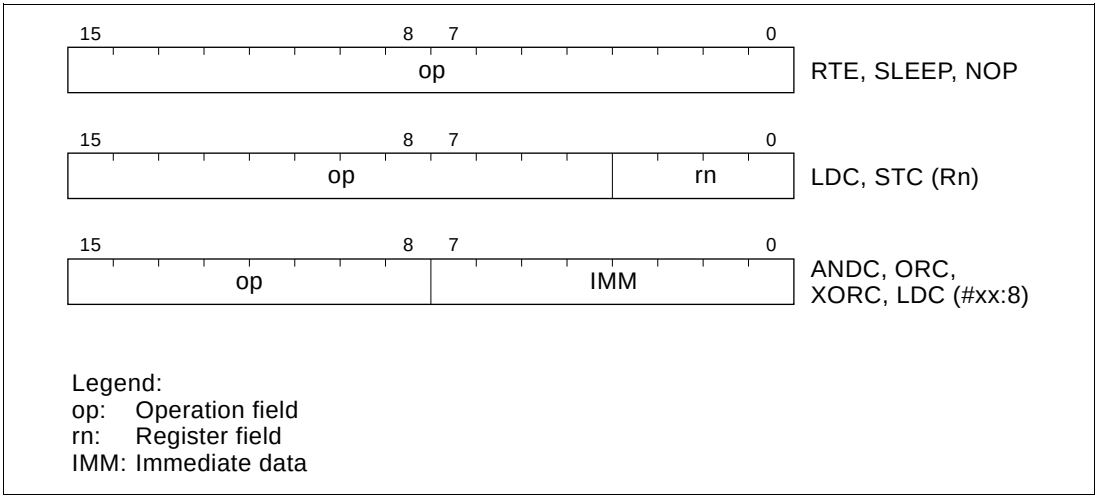


Figure 2.9 System Control Instruction Codes

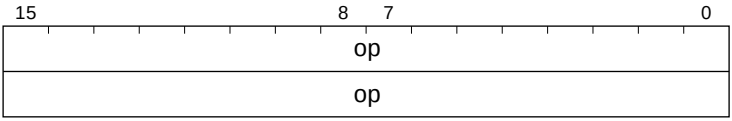
2.5.8 Block Data Transfer Instruction

Table 2.11 describes the block data transfer instruction. Figure 2.10 shows its object code format.

Table 2.11 Block Data Transfer Instruction

Instruction	Size	Function
EEPMOV	—	If R4L ≠ 0 then repeat @R5+ → @R6+, R4L – 1 → R4L until R4L = 0 else next; Block transfer instruction. Transfers the number of bytes specified by R4L, from locations starting at the address specified by R5, to locations starting at the address specified by R6. On completion of the transfer, the next instruction is executed.

Certain precautions are required in using the EEPMOV instruction. See 2.9.3, Notes on Use of the EEPMOV Instruction, for details.



Legend:
op: Operation field

Figure 2.10 Block Data Transfer Instruction Code

2.6 Basic Operational Timing

CPU operation is synchronized by a system clock (ø) or a subclock ($ø_{SUB}$). For details on these clock signals see section 4, Clock Pulse Generators. The period from a rising edge of ø or $ø_{SUB}$ to the next rising edge is called one state. A bus cycle consists of two states or three states. The cycle differs depending on whether access is to on-chip memory or to on-chip peripheral modules.

2.6.1 Access to On-Chip Memory (RAM, ROM)

Acess to on-chip memory takes place in two states. The data bus width is 16 bits, allowing access in byte or word size. Figure 2.11 shows the on-chip memory access cycle.

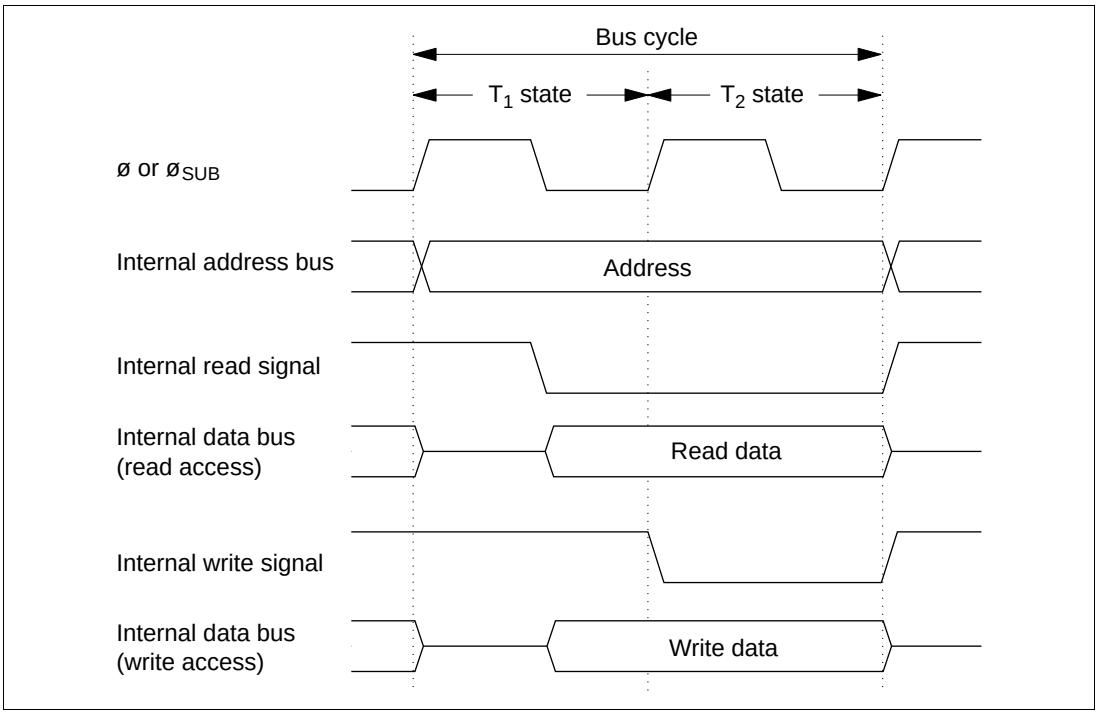


Figure 2.11 On-Chip Memory Access Cycle

2.6.2 Access to On-Chip Peripheral Modules

On-chip peripheral modules are accessed in two states or three states. The data bus width is 8 bits, so access is by byte size only. This means that for accessing word data, two instructions must be used.

Two-state access to on-chip peripheral modules

Figure 2.12 shows operation timings for accessing on-chip peripheral modules in 2 states.

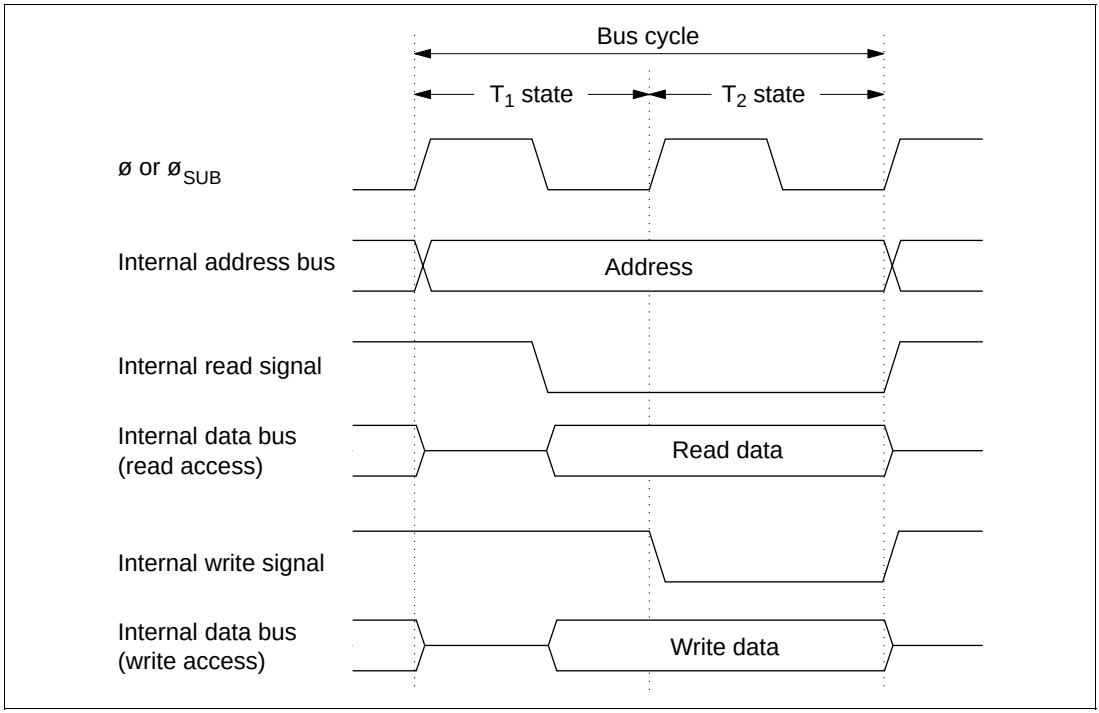


Figure 2.12 On-Chip Peripheral Module Access Cycle (2-State Access)

Three-state access to on-chip peripheral modules

Figure 2.13 shows operation timings for accessing on-chip peripheral modules in 3 states.

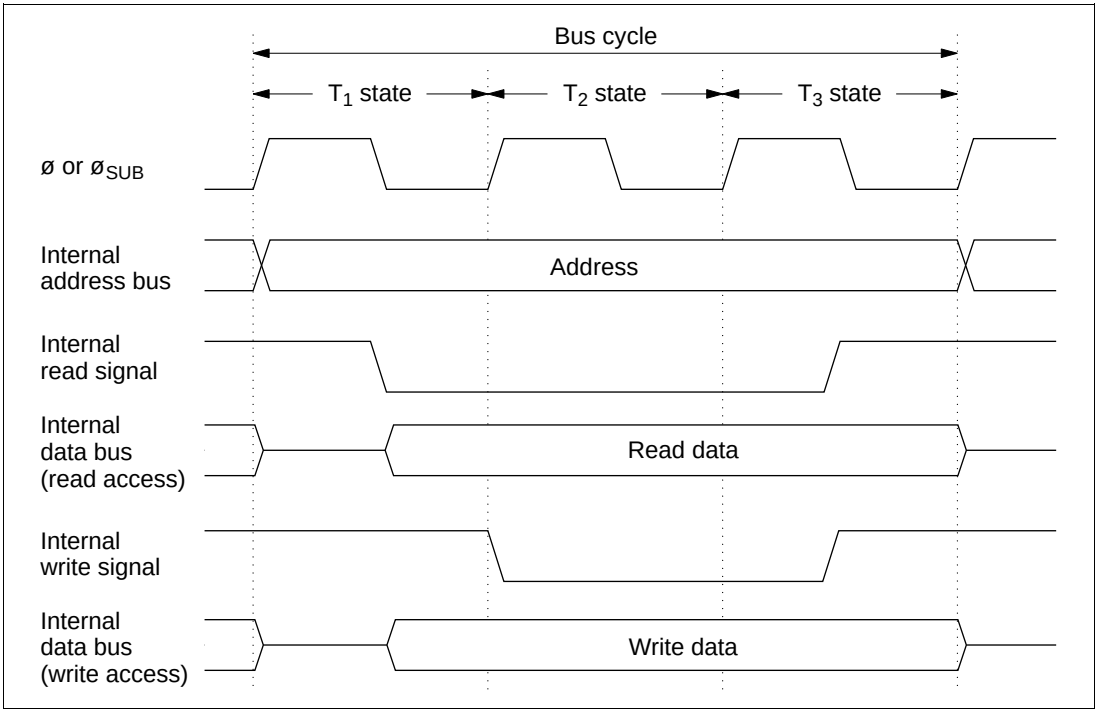


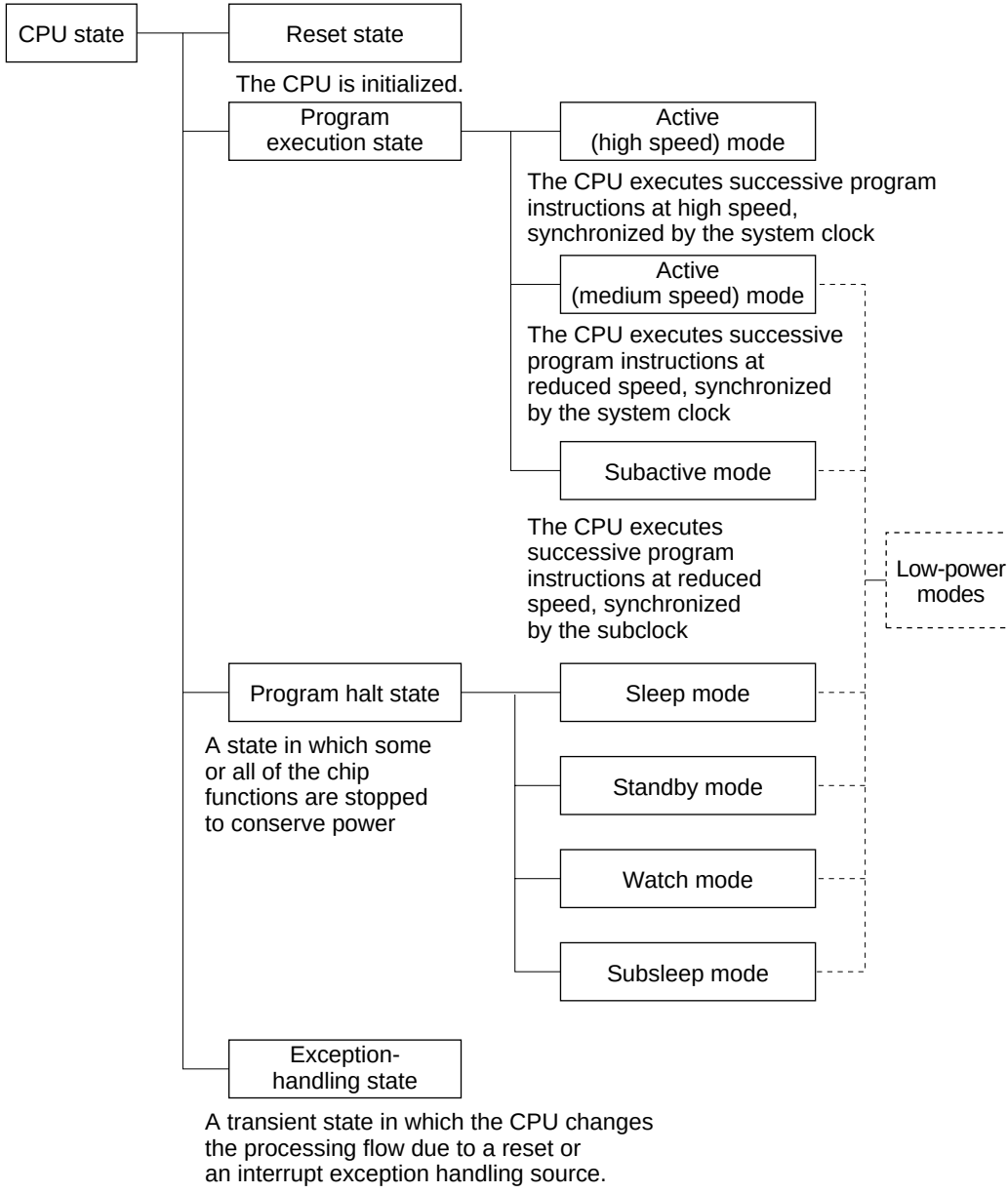
Figure 2.13 On-Chip Peripheral Module Access Cycle (3-State Access)

2.7 CPU States

2.7.1 Overview

There are four CPU states: the reset state, program execution state, program halt state, and exception-handling state. The program execution state includes active (high-speed or medium-speed) mode and subactive mode. In the program halt state there are a sleep mode, standby mode, watch mode, and sub-sleep mode. These states are shown in figure 2.14.

Figure 2.15 shows the state transitions.



Note: See section 5, Power-Down Modes, for details on the modes and their transitions.

Figure 2.14 CPU Operation States

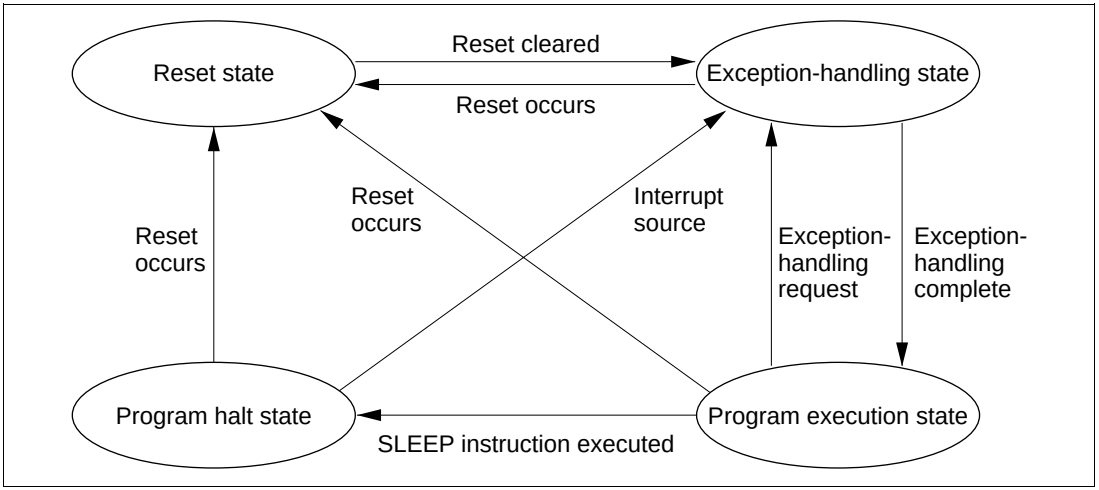


Figure 2.15 State Transitions

2.7.2 Program Execution State

In the program execution state the CPU executes program instructions in sequence.

There are three modes in this state, two active modes (high speed and medium speed) and one subactive mode. Operation is synchronized with the system clock in active mode (high speed and medium speed), and with the subclock in subactive mode. See section 5, Power-Down Modes for details on these modes.

2.7.3 Program Halt State

In the program halt state there are four modes: sleep mode, standby mode, watch mode, and subsleep mode. See section 5, Power-Down Modes for details on these modes.

2.7.4 Exception-Handling State

The exception-handling state is a transient state occurring when exception handling is started by a reset or interrupt and the CPU changes its normal processing flow. In exception handling caused by an interrupt, SP (R7) is referenced and the PC and CCR values are saved on the stack.

For details on interrupt handling, see 3.3, Interrupts.

2.8 Memory Map

Figure 2.16 shows a memory map for the H8/3627 Series.

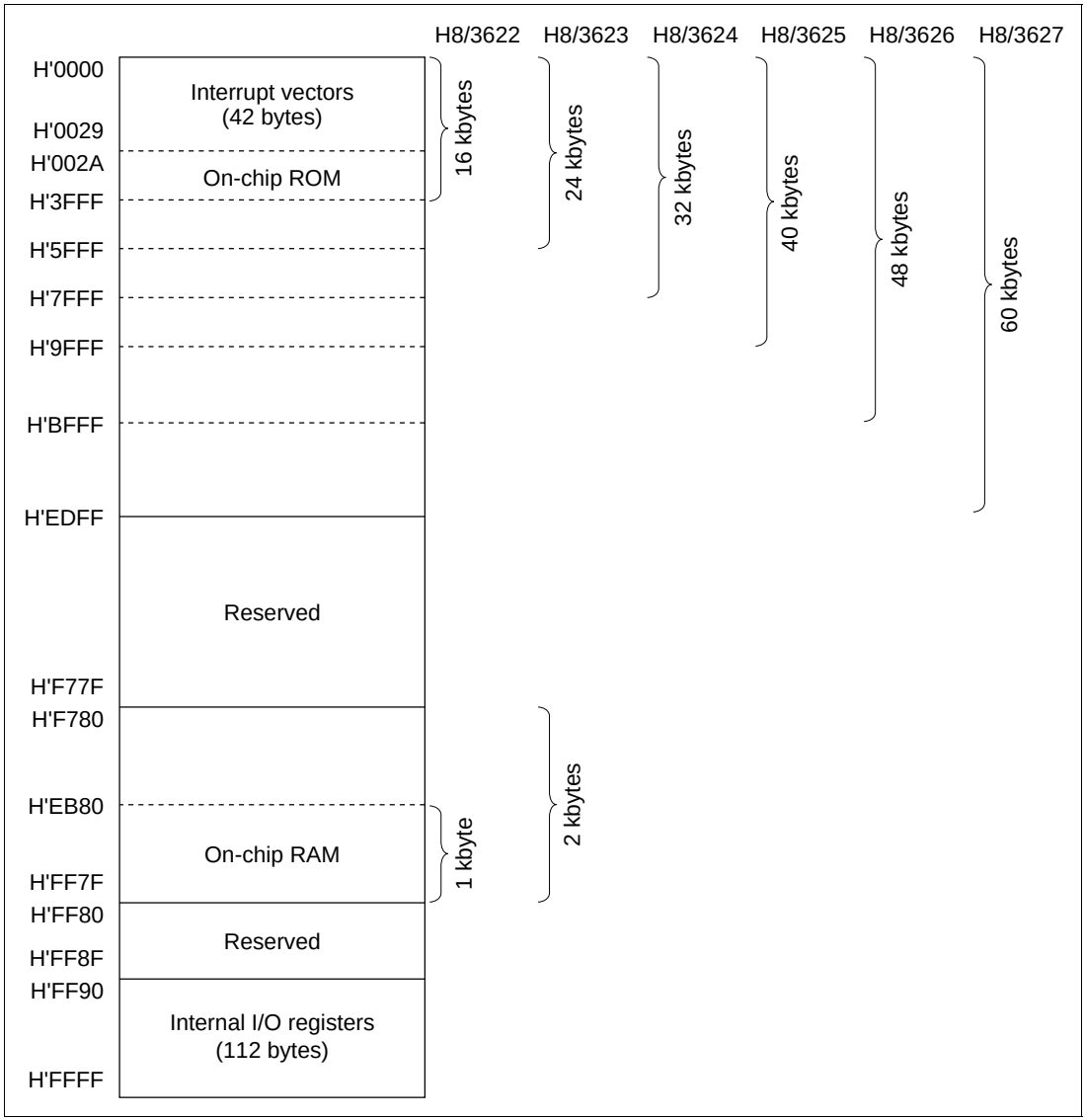


Figure 2.16 H8/3627 Series Memory Map

2.9 Application Notes

2.9.1 Notes on Data Access

Access to Empty Area: The address space of the H8/300L CPU includes empty areas in addition to the RAM, registers, and ROM areas available to the user. If these empty areas are mistakenly accessed by an application program, the following results will occur.

- Data transfer from CPU to empty area:
  The transferred data will be lost. This action may also cause the CPU to misoperate.
- Data transfer from empty area to CPU:
  Unpredictable data is transferred.

Access to the Internal I/O Register: Internal data transfer to or from on-chip modules other than the ROM and RAM areas makes use of an 8-bit data width. If word access is attempted to these areas, the following results will occur.

- Word access from CPU to I/O register area:
  Upper byte: Will be written to I/O register.
  Lower byte: Transferred data will be lost.

- Word access from I/O register to CPU:
  Upper byte: Will be written to upper part of CPU register.
  Lower byte: Unpredictable data will be written to lower part of CPU register.

Byte size instructions should therefore be used when transferring data to or from I/O registers other than the on-chip ROM and RAM areas.

Figure 2.17 shows the data size and number of states in which on-chip peripheral modules can be accessed.

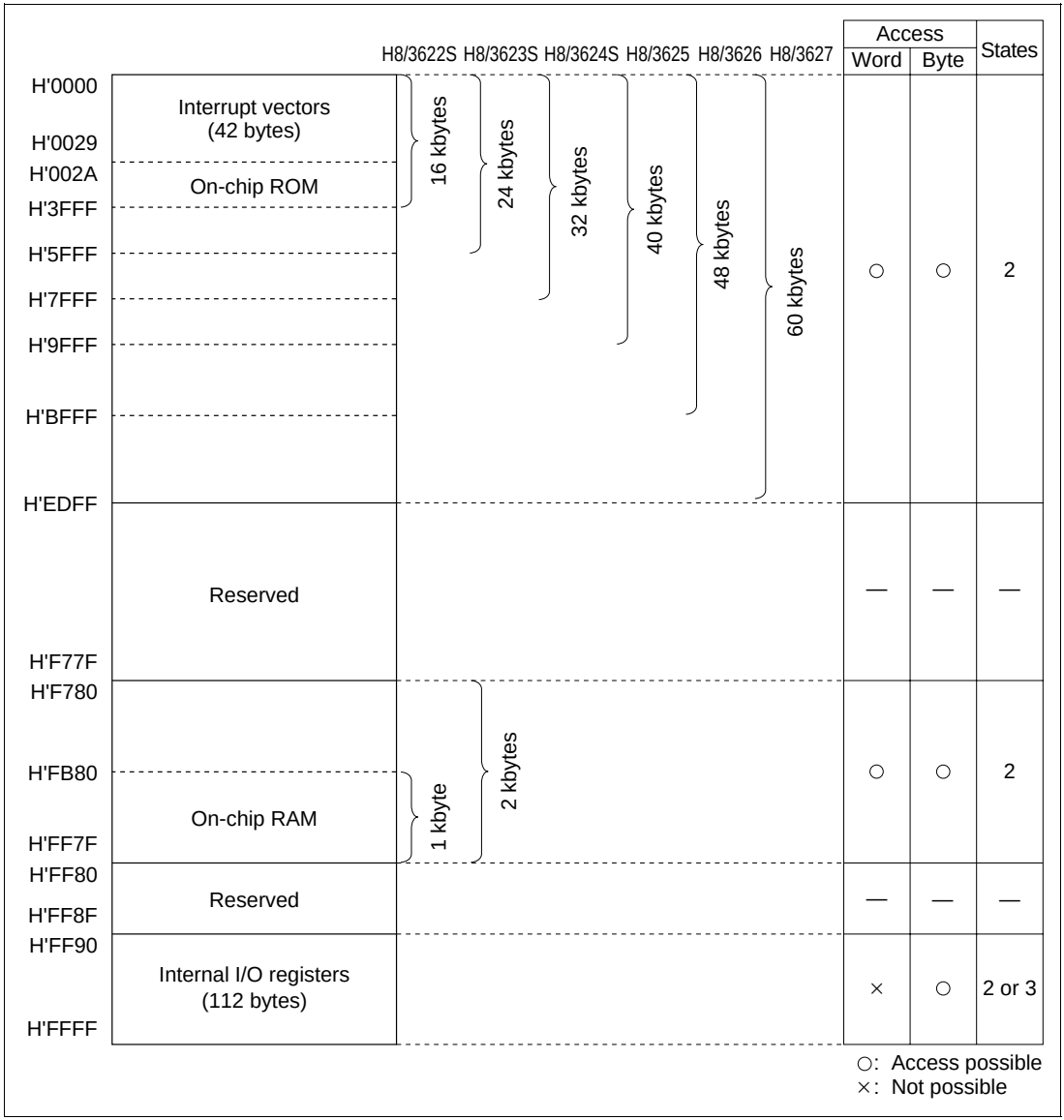


Figure 2.17 Data Size and Number of States for Access to and from On-Chip Peripheral Modules

2.9.2 Notes on Bit Manipulation

The BSET, BCLR, BNOT, BST, and BIST instructions read one byte of data, modify the data, then write the data byte again. Special care is required when using these instructions in cases where two registers are assigned to the same address, in the case of registers that include write-only bits, and when the instruction accesses an I/O.

Order of Operation		Operation
1	Read	Read byte data at the designated address
2	Modify	Modify a designated bit in the read data
3	Write	Write the altered byte data to the designated address

Bit Manipulation in Two Registers Assigned to the Same Address

Example 1: Bit manipulation to the timer load register and the timer counter

Figure 2.18 shows an example in which two timer registers share the same address. When a bit manipulation instruction accesses the timer load register and timer counter of a reloadable timer, since these two registers share the same address, the following operations take place.

Order of Operation		Operation
1	Read	Timer counter data is read (one byte)
2	Modify	The CPU modifies (sets or resets) the bit designated in the instruction
3	Write	The altered byte data is written to the timer load register

The timer counter is counting, so the value read is not necessarily the same as the value in the timer load register. As a result, bits other than the intended bit in the timer load register may be modified to the timer counter value.

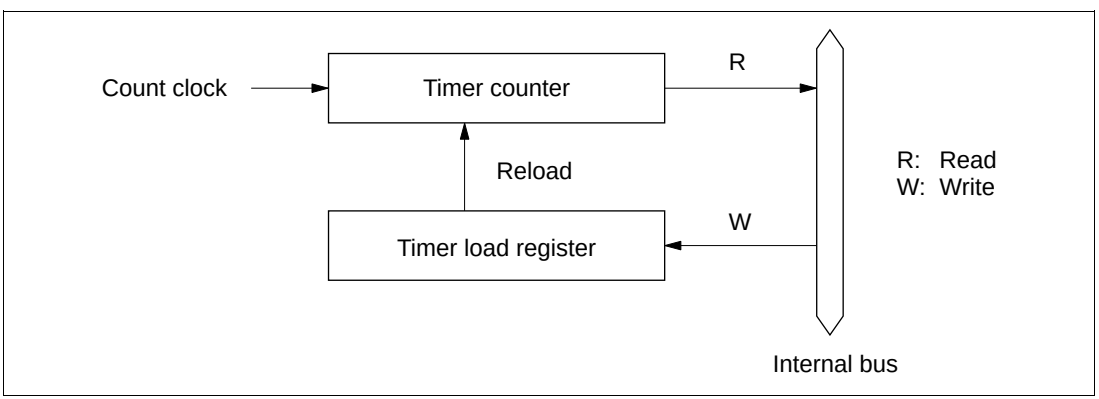


Figure 2.18 Timer Configuration Example

Example 2: Here a BSET instruction is executed designating port 6.

$P6_7$ and $P6_6$ are designated as input pins, with a low-level signal input at $P6_7$ and a high-level signal at $P6_6$. The remaining pins, $P6_5$ to $P6_0$, are output pins and output low-level signals. In this example, the BSET instruction is used to change pin $P6_0$ to high-level output.

[A: Prior to executing BSET]

	$P6_7$	**$P6_6$**	**$P6_5$**	**$P6_4$**	**$P6_3$**	**$P6_2$**	**$P6_1$**	**$P6_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR6	0	0	1	1	1	1	1	1
PDR6	1	0	0	0	0	0	0	0

[B: BSET instruction executed]

```
BSET   #0,  @PDR6
```

The BSET instruction is executed designating port 6.

[C: After executing BSET]

	$P6_7$	**$P6_6$**	**$P6_5$**	**$P6_4$**	**$P6_3$**	**$P6_2$**	**$P6_1$**	**$P6_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR6	0	0	1	1	1	1	1	1
PDR6	0	1	0	0	0	0	0	1

[D: Explanation of how BSET operates]

When the BSET instruction is executed, first the CPU reads port 6.

Since $P6_7$ and $P6_6$ are input pins, the CPU reads the pin states (low-level and high-level input). $P6_5$ to $P6_0$ are output pins, so the CPU reads the value in PDR6. In this example PDR6 has a value of H'80, but the value read by the CPU is H'40.

Next, the CPU sets bit 0 of the read data to 1, changing the PDR6 data to H'41. Finally, the CPU writes this value (H'41) to PDR6, completing execution of BSET.

As a result of this operation, bit 0 in PDR6 becomes 1, and $P6_0$ outputs a high-level signal. However, bits 7 and 6 of PDR6 end up with different values.

To avoid this problem, store a copy of the PDR6 data in a work area in memory. Perform the bit manipulation on the data in the work area, then write this data to PDR6.

[A: Prior to executing BSET]

```
MOV. B  #80,  R0L
MOV. B  R0L,  @RAM0
MOV. B  R0L,  @PDR6
```

The PDR6 value (H'80) is written to a work area in memory (RAM0) as well as to PDR6.

	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR6	0	0	1	1	1	1	1	1
PDR6	1	0	0	0	0	0	0	0
RAM0	1	0	0	0	0	0	0	0

[B: BSET instruction executed]

```
BSET    #0,  @RAM0
```

The BSET instruction is executed designating the PDR6 work area (RAM0).

[C: After executing BSET]

```
MOV. B  @RAM0, R0L
MOV. B  R0L,  @PDR6
```

The work area (RAM0) value is written to PDR6.

	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR6	0	0	1	1	1	1	1	1
PDR6	1	0	0	0	0	0	0	1
RAM0	1	0	0	0	0	0	0	1

Bit Manipulation in a Register Containing a Write-only Bit

Example 3: In this example, the port 6 control register PCR6 is accessed by a BCLR instruction.

As in the examples above, $P6_7$ and $P6_6$ are input pins, with a low-level signal input at $P6_7$ and a high-level signal at $P6_6$. The remaining pins, $P6_5$ to $P6_0$, are output pins that output low-level signals. In this example, the BCLR instruction is used to change pin $P6_0$ to an input port. It is assumed that a high-level signal will be input to this input pin.

[A: Prior to executing BCLR]

	$P6_7$	**$P6_6$**	**$P6_5$**	**$P6_4$**	**$P6_3$**	**$P6_2$**	**$P6_1$**	**$P6_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR6	0	0	1	1	1	1	1	1
PDR6	1	0	0	0	0	0	0	0

[B: BCLR instruction executed]

```
BCLR   #0,  @PCR6
```

The BCLR instruction is executed designating PCR6.

[C: After executing BCLR]

	$P6_7$	**$P6_6$**	**$P6_5$**	**$P6_4$**	**$P6_3$**	**$P6_2$**	**$P6_1$**	**$P6_0$**
Input/output	Output	Output	Output	Output	Output	Output	Output	Input
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR6	1	1	1	1	1	1	1	0
PDR6	1	0	0	0	0	0	0	0

[D: Explanation of how BCLR operates]

When the BCLR instruction is executed, first the CPU reads PCR6. Since PCR6 is a write-only register, the CPU reads a value of H'FF, even though the PCR6 value is actually H'3F.

Next, the CPU clears bit 0 in the read data to 0, changing the data to H'FE. Finally, this value (H'FE) is written to PCR6 and BCLR instruction execution ends.

As a result of this operation, bit 0 in PCR6 becomes 0, making $P6_0$ an input port. However, bits 7 and 6 in PCR6 change to 1, so that $P6_7$ and $P6_6$ change from input pins to output pins.

To avoid this problem, store a copy of the PCR6 data in a work area in memory. Perform the bit manipulation on the data in the work area, then write this data to PCR6.

[A: Prior to executing BCLR]

```
MOV. B  #3F,  R0L
MOV. B  R0L,  @RAM0
MOV. B  R0L,  @PCR6
```

The PCR6 value (H'3F) is written to a work area in memory (RAM0) as well as to PCR6.

	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR6	0	0	1	1	1	1	1	1
PDR6	1	0	0	0	0	0	0	0
RAM0	0	0	1	1	1	1	1	1

[B: BCLR instruction executed]

```
BCLR    #0,  @RAM0
```

The BCLR instruction is executed designating the PCR6 work area (RAM0).

[C: After executing BCLR]

```
MOV. B  @RAM0, R0L
MOV. B  R0L,  @PCR6
```

The work area (RAM0) value is written to PCR6.

	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR6	0	0	1	1	1	1	1	0
PDR6	1	0	0	0	0	0	0	0
RAM0	0	0	1	1	1	1	1	0

Table 2.12 lists the registers with shared addresses. Table 2.13 lists the registers that contain write-only bits.

Table 2.12 Registers with Shared Addresses

Register Name	Abbreviation	Address
Port data register 1*	PDR1	H'FFD4
Port data register 2*	PDR2	H'FFD5
Port data register 5*	PDR5	H'FFD8
Port data register 6*	PDR6	H'FFD9
Port data register 7*	PDR7	H'FFDA
Port data register 8*	PDR8	H'FFDB
Port data register A*	PDRA	H'FFDD

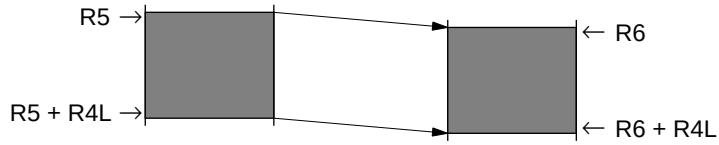
Note: * The port data register addresses are also assigned directly to input pins.

Table 2.13 Registers with Write-Only Bits

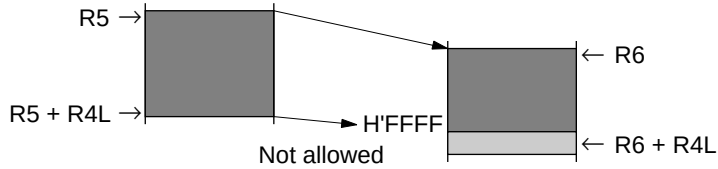
Register Name	Abbreviation	Address
Port control register 1	PCR1	H'FFE4
Port control register 2	PCR2	H'FFE5
Port control register 5	PCR5	H'FFE8
Port control register 6	PCR6	H'FFE9
Port control register 7	PCR7	H'FFEA
Port control register 8	PCR8	H'FFEB
Port control register A	PCRA	H'FFED
Timer control register F	TCRF	H'FFB6

2.9.3 Notes on Use of the EEPMOV Instruction

- 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.



- 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.



Section 3 Exception Handling

3.1 Overview

Exception handling is performed in the H8/3627 Series when a reset or interrupt occurs. Table 3.1 shows the priorities of these two types of exception handling.

Table 3.1 Exception Handling Types and Priorities

Priority	Exception Source	Time of Start of Exception Handling
High	Reset	Exception handling starts as soon as the reset state is cleared
↑	Interrupt	When an interrupt is requested, exception handling starts after execution of the present instruction or the exception handling in progress is completed
Low		

3.2 Reset

3.2.1 Overview

A reset is the highest-priority exception. The internal state of the CPU and the registers of the on-chip peripheral modules are initialized.

3.2.2 Reset Sequence

As soon as the $\overline{\text{RES}}$ pin goes low, all processing is stopped and the H8/3637 Series enters the reset state.

To make sure the chip is reset properly, observe the following precautions.

- At power on: Hold the $\overline{\text{RES}}$ pin low until the clock pulse generator output stabilizes.
- When an external clock or ceramic oscillator is used, also, at power on the $\overline{\text{RES}}$ pin must be held low for the crystal oscillator oscillation stabilization time shown in table 14.3 in section 14, Electrical Characteristics.
- Resetting during operation: Hold the $\overline{\text{RES}}$ pin low for at least 18 system clock cycles.

Reset exception handling begins when the $\overline{\text{RES}}$ pin is held low for a given period, then returned to the high level. Reset exception handling takes place as follows.

- The CPU internal state and the registers of on-chip peripheral modules are initialized, with the I bit of the condition code register (CCR) set to 1.

- The PC is loaded from the reset exception handling vector address (H'0000 to H'0001), after which the program starts executing from the address indicated in PC.

When system power is turned on or off, the $\overline{\text{RES}}$ pin should be held low.

Figure 3.1 shows the reset sequence.

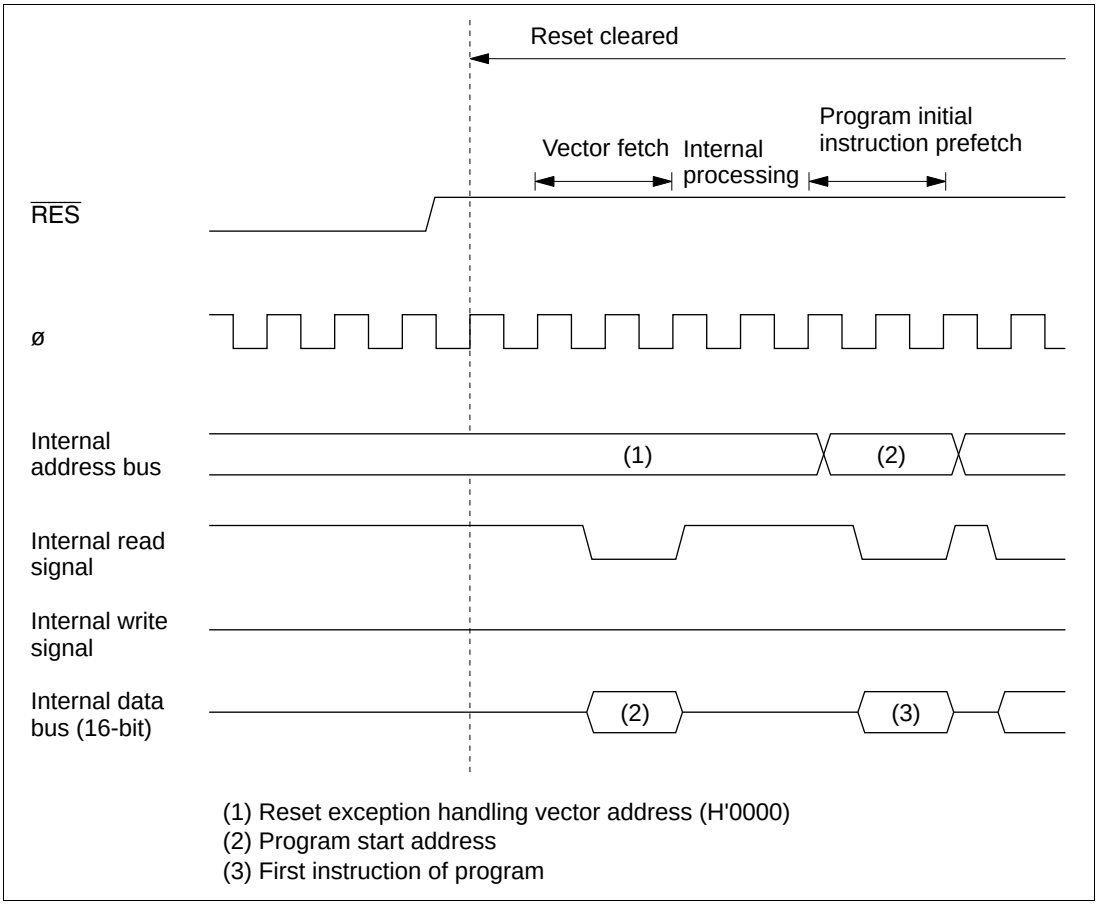


Figure 3.1 Reset Sequence

3.2.3 Interrupt Immediately after Reset

After a reset, if an interrupt were to be accepted before the stack pointer (SP: R7) was initialized, PC and CCR would not be pushed onto the stack correctly, resulting in program runaway. To prevent this, immediately after reset exception handling all interrupts are masked. For this reason, the initial program instruction is always executed immediately after a reset. This instruction should initialize the stack pointer (e.g. MOV.W #xx: 16, SP).

3.3 Interrupts

3.3.1 Overview

The interrupt sources include 13 external interrupts (WKP_0 to WKP_7, IRQ_0 to IRQ_4), and 16 internal interrupts from on-chip peripheral modules. Table 3.2 shows the interrupt sources, their priorities, and their vector addresses. When more than one interrupt is requested, the interrupt with the highest priority is processed.

The interrupts have the following features:

- Internal and external interrupts can be masked by the I bit of CCR. When this bit is set to 1, interrupt request flags are set but interrupts are not accepted.
- IRQ_0 to IRQ_4 can each be set independently to either rising edge sensing or falling edge sensing.

Table 3.2 Interrupt Sources and Priorities

Interrupt Source	Interrupt	Vector Number	Vector Address	Priority
$\overline{RES}$	Reset	0	H'0000 to H'0001	High
$\overline{IRQ_0}$	IRQ_0	4	H'0008 to H'0009	
$\overline{IRQ_1}$	IRQ_1	5	H'000A to H'000B	
$\overline{IRQ_2}$	IRQ_2	6	H'000C to H'000D	
$\overline{IRQ_3}$	IRQ_3	7	H'000E to H'000F	
$\overline{IRQ_4}$	IRQ_4	8	H'0010 to H'0011	
$\overline{WKP_0}$	WKP_0	9	H'0012 to H'0013	
$\overline{WKP_1}$	WKP_1			
$\overline{WKP_2}$	WKP_2			
$\overline{WKP_3}$	WKP_3			
$\overline{WKP_4}$	WKP_4			
$\overline{WKP_5}$	WKP_5			
$\overline{WKP_6}$	WKP_6			
$\overline{WKP_7}$	WKP_7			
SCI1	SCI1 transfer complete	10	H'0014 to H'0015	
Timer A	Timer A overflow	11	H'0016 to H'0017	
Timer FL	Timer FL compare match	14	H'001C to H'001D	
	Timer FL overflow			
Timer FH	Timer FH compare match	15	H'001E to H'001F	
	Timer FH overflow			
Timer G	Timer G input capture	16	H'0020 to H'0021	
	Timer G overflow			
SCI3	SCI3 receive data full	18	H'0024 to H'0025	
	SCI3 transmit data empty			
	SCI3 transmit end			
	SCI3 overrun error			
	SCI3 framing error			
	SCI3 parity error			
A/D converter	A/D conversion end	19	H'0026 to H'0027	
(SLEEP instruction executed)	Direct transfer	20	H'0028 to H'0029	Low

Note: Vector addresses H'0002 to H'0007, H'0018 to H'001B, and H'0022 to H'0023 are reserved and cannot be used.

3.3.2 Interrupt Control Registers

Table 3.3 lists the registers that control interrupts.

Table 3.3 Interrupt Control Registers

Name	Abbreviation	R/W	Initial Value	Address
Interrupt edge select register	IEGR	R/W	H'60	H'FFF2
Interrupt enable register 1	IENR1	R/W	H'00	H'FFF3
Interrupt enable register 2	IENR2	R/W	H'01	H'FFF4
Interrupt request register 1	IRR1	R/W*	H'20	H'FFF6
Interrupt request register 2	IRR2	R/W*	H'01	H'FFF7
Wakeup interrupt request register	IWPR	R/W*	H'00	H'FFF9

Note: * Write is enabled only for writing of 0 to clear a flag.

Interrupt Edge Select Register (IEGR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	IEG4	IEG3	IEG2	IEG1	IEG0
Initial value	0	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

IEGR is an 8-bit read/write register, used to designate whether pins $\overline{IRQ_0}$ to $\overline{IRQ_4}$ are set to rising edge sensing or falling edge sensing.

Bit 7—Reserved Bit: Bit 7 is reserved: it is always read as 0, and should be used cleared to 0.

Bits 6 and 5—Reserved Bits: Bits 6 and 5 are reserved; they are always read as 1, and cannot be modified.

Bit 4—$\overline{IRQ_4}$ Edge Select (IEG4): Bit 4 selects the input sensing of pin $\overline{IRQ_4}/\overline{ADTRG}$.

Bit 4: IEG4	Description	
0	Falling edge of $\overline{IRQ_4}/\overline{ADTRG}$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ_4}/\overline{ADTRG}$ pin input is detected	

Bit 3—$\overline{IRQ_3}$ Edge Select(IEG3): Bit 3 selects the input sensing of pin $\overline{IRQ_3}$/TMIF.

Bit 3: IEG3	Description	
0	Falling edge of $\overline{IRQ_3}$/TMIF pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ_3}$/TMIF pin input is detected	

Bit 2—$\overline{IRQ_2}$ Edge Select(IEG2): Bit 2 selects the input sensing of pin $\overline{IRQ_2}$.

Bit 2: IEG2	Description	
0	Falling edge of $\overline{IRQ_2}$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ_2}$ pin input is detected	

Bit 1—$\overline{IRQ_1}$ Edge Select(IEG1): Bit 1 selects the input sensing of pin $\overline{IRQ_1}$.

Bit 1: IEG1	Description	
0	Falling edge of $\overline{IRQ_1}$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ_1}$ pin input is detected	

Bit 0—$\overline{IRQ_0}$ Edge Select(IEG0): Bit 0 selects the input sensing of pin $\overline{IRQ_0}$.

Bit 0: IEG0	Description	
0	Falling edge of $\overline{IRQ_0}$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ_0}$ pin input is detected	

Interrupt Enable Register 1 (IENR1)

IENR1 is an 8-bit read/write register that enables or disables interrupt requests.

Bit	7	6	5	4	3	2	1	0
	IENTA	IENS1	IENWP	IEN4	IEN3	IEN2	IEN1	IEN0
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

Bit 7—Timer A Interrupt Enable (IENTA): Bit 7 enables or disables timer A overflow interrupt requests.

Bit 7: IENTA	Description	
0	Disables timer A interrupts	(initial value)
1	Enables timer A interrupts	

Bit 6—SCI1 Interrupt Enable (IENS1): Bit 6 enables or disables SCI1 transfer complete interrupt requests.

Bit 6: IENS1	Description	
0	Disables SCI1 interrupts	(initial value)
1	Enables SCI1 interrupts	

Bit 5—Wakeup Interrupt Enable (IENWP): Bit 5 enables or disables WKP_7 to WKP_0 interrupt requests.

Bit 5: IENWP	Description	
0	Disables interrupt requests from $\overline{WKP}_7$ to $\overline{WKP}_0$	(initial value)
1	Enables interrupt requests from $\overline{WKP}_7$ to $\overline{WKP}_0$	

Bits 4 to 0: IRQ_4 to IRQ_0 Interrupt Enable (IEN4 to IEN0): Bits 4 to 0 enable or disable IRQ_4 to IRQ_0 interrupt requests.

Bits 4 to 0: IEN4 to IEN0	Description	
0	Disables interrupt requests from $\overline{IRQ}_4$ to $\overline{IRQ}_0$	(initial value)
1	Enables interrupt requests from $\overline{IRQ}_4$ to $\overline{IRQ}_0$	

Interrupt Enable Register 2 (IENR2)

Bit	7	6	5	4	3	2	1	0
	IENDT	IENAD	—	IENTG	IENTFH	IENTFL	—	—
Initial value	0	0	0	0	0	0	0	1
Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	—

IENR2 is an 8-bit read/write register that enables or disables interrupt requests.

Bit 7—Direct Transfer Interrupt Enable (IENDT): Bit 7 enables or disables direct transfer interrupt requests.

Bit 7: IENDT	Description	
0	Disables direct transfer interrupt requests	(initial value)
1	Enables direct transfer interrupt requests	

Bit 6—A/D Converter Interrupt Enable (IENAD): Bit 6 enables or disables A/D converter end interrupt requests.

Bit 6: IENAD	Description	
0	Disables A/D converter interrupt requests	(initial value)
1	Enables A/D converter interrupt requests	

Bit 5—Reserved Bit: Bit 5 is reserved: it is always read as 0, and should be used cleared to 0.

Bit 4—Timer G Interrupt Enable (IENTG): Bit 4 enables or disables timer G input capture and overflow interrupt requests.

Bit 4: IENTG	Description	
0	Disables timer G interrupts	(initial value)
1	Enables timer G interrupts	

Bit 3—Timer FH Interrupt Enable (IENTFH): Bit 3 enables or disables timer FH compare match and overflow interrupt requests.

Bit 3: IENTFH	Description	
0	Disables timer FH interrupts	(initial value)
1	Enables timer FH interrupts	

Bit 2—Timer FL Interrupt Enable (IENTFL): Bit 2 enables or disables timer FL compare match and overflow interrupt requests.

Bit 2: IENTFL	Description	
0	Disables timer FL interrupts	(initial value)
1	Enables timer FL interrupts	

Bit 1—Reserved Bit: Bit 1 is reserved: it is always read as 0, and should be used cleared to 0.

Bit 0—Reserved Bit: Bit 0 is reserved: it is always read as 1, and cannot be modified.

For details of SCI3 interrupt control, see Serial Control Register 3 (SCR3), in section 10.3.2.

Interrupt Request Register 1 (IRR1)

Bit	7	6	5	4	3	2	1	0
	IRRTA	IRRS1	—	IRRI4	IRRI3	IRRI2	IRRI1	IRRI0
Initial value	0	0	1	0	0	0	0	0
Read/Write	R/W*	R/W*	—	R/W*	R/W*	R/W*	R/W*	R/W*

Note: * Only a write of 0 for flag clearing is possible.

IRR1 is an 8-bit read/write register, in which the corresponding bit is set to 1 when a timer A, SCI1, or IRQ_4 to IRQ_0 interrupt is requested. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit 7—Timer A Interrupt Request Flag (IRRTA)

Bit 7: IRRTA	Description	
0	[Clearing conditions] When IRRTA = 1, it is cleared by writing 0	(initial value)
1	[Setting conditions] When the timer A counter value overflows (goes from H'FF to H'00)	

Bit 6—SCI1 Interrupt Request Flag (IRRS1)

Bit 6: IRRS1	Description	
0	[Clearing conditions] When IRRS1 = 1, it is cleared by writing 0	(initial value)
1	[Setting conditions] When an SCI1 transfer is completed	

Bit 5—Reserved Bit: Bit 5 is reserved; it is always read as 1, and cannot be modified.

Bits 4 to 0—IRQ_4 to IRQ_0 Interrupt Request Flags (IRRI4 to IRRI0)

Bits 4 to 0: IRRI4 to IRRI0	Description
0	[Clearing conditions] (initial value) When IRRIn = 1, it is cleared by writing 0 to IRRIn.
1	[Setting conditions] IRRIn is set when pin $\overline{IRQ}_n$ is set to interrupt input, and the designated signal edge is detected.

(n = 4 to 0)

Interrupt Request Register 2 (IRR2)

Bit	7	6	5	4	3	2	1	0
	IRRDT	IRRAD	—	IRRTG	IRRTFH	IRRTFL	—	—
Initial value	0	0	0	0	0	0	0	1
Read/Write	R/W*	R/W*	—	R/W*	R/W*	R/W*	—	—

Note: * Only a write of 0 for flag clearing is possible.

IRR2 is an 8-bit register containing direct transfer, A/D converter, timer G, timer FH, and timer FL, interrupt flags. When a direct transfer, A/D converter, timer G, timer FH, or timer FL, interrupt is requested, the corresponding flag is set to 1. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit 7—Direct Transfer Interrupt Request Flag (IRRDT)

Bit 7: IRRDT	Description
0	[Clearing conditions] (initial value) When IRRDT = 1, it is cleared by writing 0
1	[Setting conditions] When DTON = 1 and a direct transfer is made immediately after a SLEEP instruction is executed

Bit 6—A/D Converter Interrupt Request Flag (IRRAD)

Bit 6: IRRAD	Description
0	[Clearing conditions] (initial value) When IRRAD = 1, it is cleared by writing 0
1	[Setting conditions] When A/D conversion is completed and ADSF is reset

Bit 5—Reserved Bit: Bit 5 is reserved: it is always read as 0, and should be used cleared to 0.

Bit 4—Timer G Interrupt Request Flag (IRRTG)

Bit 4: IRRTG	Description
0	[Clearing conditions] (initial value) When IRRTG = 1, it is cleared by writing 0
1	[Setting conditions] When pin TMIG is set to TMIG input and the designated signal edge is detected, or when TCG overflows (from H'FF to H'00) while TMG OVIE is set to 1

Bit 3—Timer FH Interrupt Request Flag (IRRTFH)

Bit 3: IRRTFH	Description
0	[Clearing conditions] (initial value) When IRRTFH = 1, it is cleared by writing 0
1	[Setting conditions] When counter FH matches output compare register FH in 8-bit timer mode, or when 16-bit counter F (TCFL, TCFH) matches output compare register F (OCRFL, OCRFH) in 16-bit timer mode

Bit 2—Timer FL Interrupt Request Flag (IRRTFL)

Bit 2: IRRTFL	Description
0	[Clearing conditions] (initial value) When IRRTFL = 1, it is cleared by writing 0
1	[Setting conditions] When counter FL matches output compare register FL in 8-bit timer mode

Bit 1—Reserved Bit: Bit 1 is reserved: it is a always read as 0, and should be used cleared to 0.

Bit 0—Reserved Bit: Bit 0 is reserved: it is a always read as 1, and cannot be modified.

Wakeup Interrupt Request Register (IWPR)

Bit	7	6	5	4	3	2	1	0
	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0
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 a write of 0 for flag clearing is possible.

IWPR is an 8-bit read/write register, in which the corresponding bit is set to 1 when pins $\overline{WKP}_7$ to $\overline{WKP}_0$ are set to wakeup input and a pin receives a falling edge input. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bits 7 to 0—Wakeup Interrupt Request Flags (IWPF7 to IWPF0)

Bits 7 to 0: IWPF7 to IWPF0	Description
0	[Clearing conditions] (initial value) When IWPFn = 1, it is cleared by writing 0 to IWPFn.
1	[Setting conditions] IWPFn is set when pin $\overline{WKP}_n$ is set to wakeup interrupt input, and a falling edge input is detected at the pin.

(n = 7 to 0)

3.3.3 External Interrupts

There are 13 external interrupts, WKP_0 to WKP_7 and IRQ_0 to IRQ_4.

Interrupts WKP_0 to WKP_7: Interrupts WKP_0 to WKP_7 are requested by falling edge inputs at pins $\overline{WKP_0}$ to $\overline{WKP_7}$. When these pins are designated as $\overline{WKP_0}$ to $\overline{WKP_7}$ pins in port mode register 5 (PMR5) and falling edge input is detected, the corresponding bit in the wakeup interrupt request register (IWPR) is set to 1, requesting an interrupt. Wakeup interrupt requests can be disabled by clearing the IENWP bit in IENR1 to 0. It is also possible to mask all interrupts by setting the CCR I bit to 1.

When an interrupt exception handling request is received for interrupts WKP_0 to WKP_7, the CCR I bit is set to 1. The vector number for interrupts WKP_0 to WKP_7 is 9. Since all eight interrupts are assigned the same vector number, the interrupt source must be determined by the exception handling routine.

Interrupts IRQ_0 to IRQ_4: Interrupts IRQ_0 to IRQ_4 are requested by inputs into pins $\overline{IRQ_0}$ to $\overline{IRQ_4}$. These interrupts are detected by either rising edge sensing or falling edge sensing, depending on the settings of bits IEG0 to IEG4 in the edge select register (IEGR).

When these pins are designated as pins $\overline{IRQ_0}$ to $\overline{IRQ_4}$ in port mode registers 1 and 2 (PMR1 and PMR2) and the designated edge is input, the corresponding bit in IRR1 is set to 1, requesting an interrupt. Interrupts IRQ_0 to IRQ_4 can be disabled by clearing bits IEN0 to IEN4 in IENR1 to 0. All interrupts can be masked by setting the I bit in CCR to 1.

When IRQ_0 to IRQ_4 interrupt exception handling is initiated, the I bit in CCR is set to 1. Vector numbers 4 to 8 are assigned to interrupts IRQ_0 to IRQ_4. The order of priority is from IRQ_0 (high) to IRQ_4 (low). Table 3.2 gives details.

3.3.4 Internal Interrupts

There are 16 internal interrupts that can be requested by the on-chip peripheral modules. When a peripheral module requests an interrupt, the corresponding bit in IRR1 or IRR2 is set to 1. Individual interrupt requests can be disabled by clearing the corresponding bit in IENR1 or IENR2 to 0. All interrupts can be masked by setting the I bit in CCR to 1. When an internal interrupt request is accepted, the I bit in CCR is set to 1. Vector numbers 10 to 20 are assigned to these interrupts. Table 3.2 shows the order of priority of interrupts from on-chip peripheral modules.

3.3.5 Interrupt Operations

Interrupts are controlled by an interrupt controller. Figure 3.2 shows a block diagram of the interrupt controller. Figure 3.3 shows the flow up to interrupt acceptance.

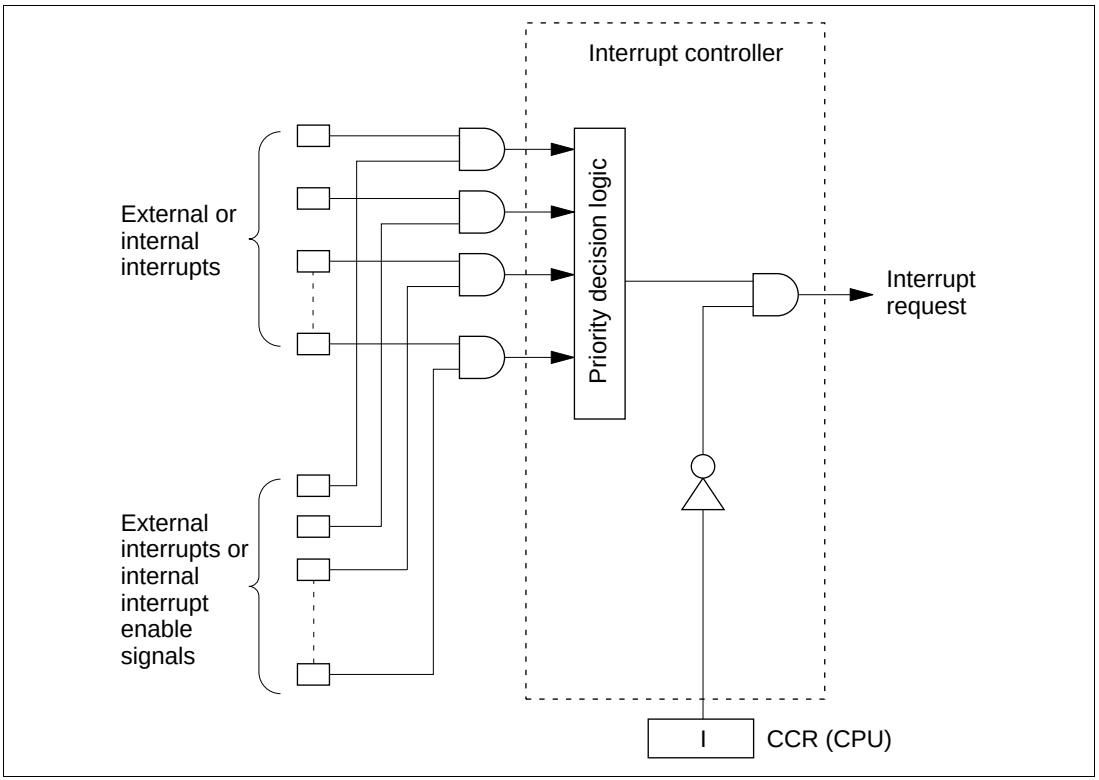


Figure 3.2 Block Diagram of Interrupt Controller

Interrupt operation is described as follows.

1. When an interrupt condition is met while the interrupt enable register bit is set to 1, an interrupt request signal is sent to the interrupt controller.
2. When the interrupt controller receives an interrupt request, it sets the interrupt request flag.
3. From among the interrupts with interrupt request flags set to 1, the interrupt controller selects the interrupt request with the highest priority and holds the others pending. (Refer to table 3.2 for a list of interrupt priorities.)
4. The interrupt controller checks the I bit of CCR. If the I bit is 0, the selected interrupt request is accepted; if the I bit is 1, the interrupt request is held pending.
5. If the interrupt is accepted, after processing of the current instruction is completed, both PC and CCR are pushed onto the stack. The state of the stack at this time is shown in figure 3.4. The PC value pushed onto the stack is the address of the first instruction to be executed upon return from interrupt handling.
6. The I bit of CCR is set to 1, masking all further interrupts.
7. The vector address corresponding to the accepted interrupt is generated, and the interrupt handling routine located at the address indicated by the contents of the vector address is executed.

Notes:
1. When disabling interrupts by clearing bits in an interrupt enable register, or when clearing bits in an interrupt request register, always do so while interrupts are masked (I = 1).
2. If the above clear operations are performed while I = 0, and as a result a conflict arises between the clear instruction and an interrupt request, exception processing for the interrupt will be executed after the clear instruction has been executed.

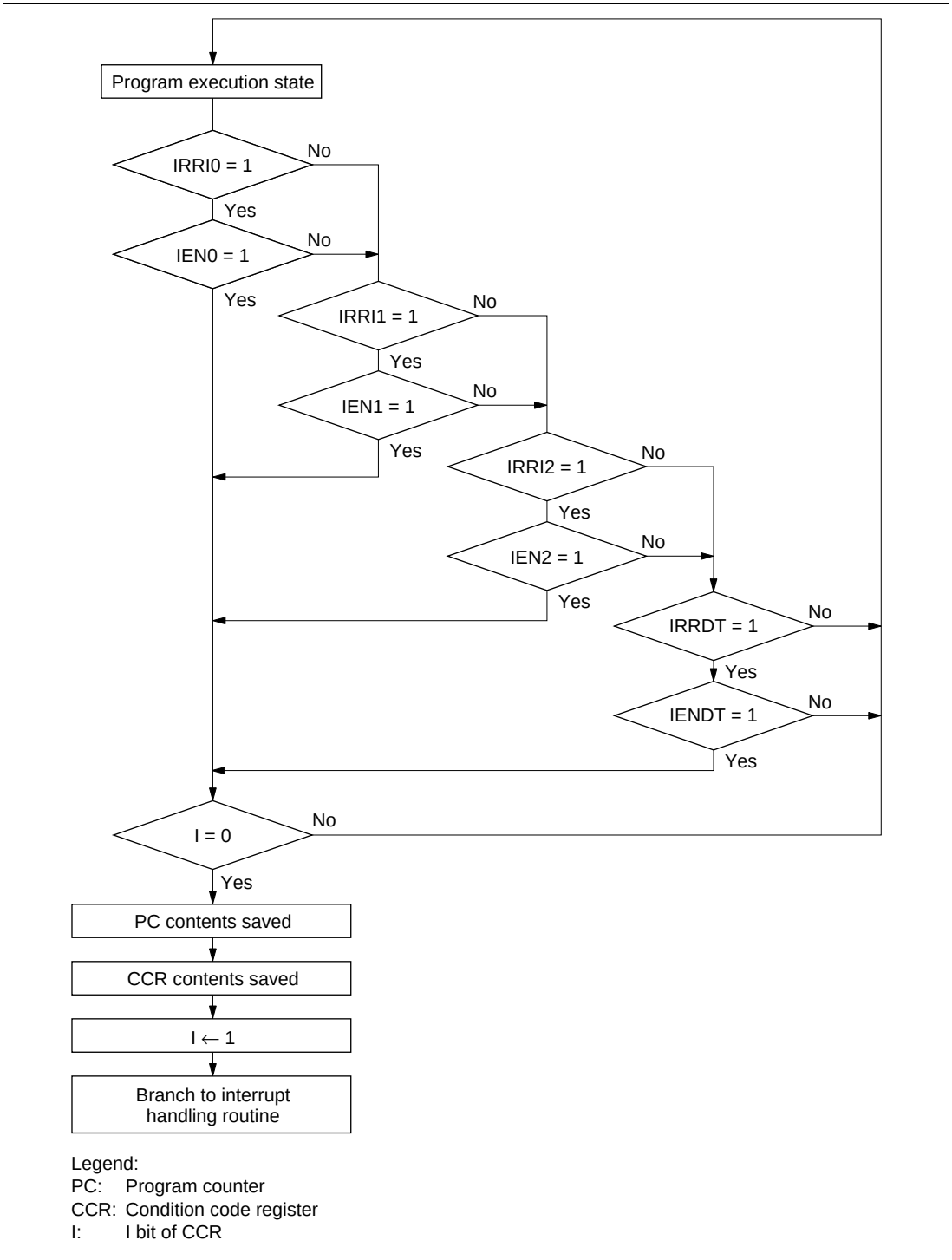


Figure 3.3 Flow up to Interrupt Acceptance

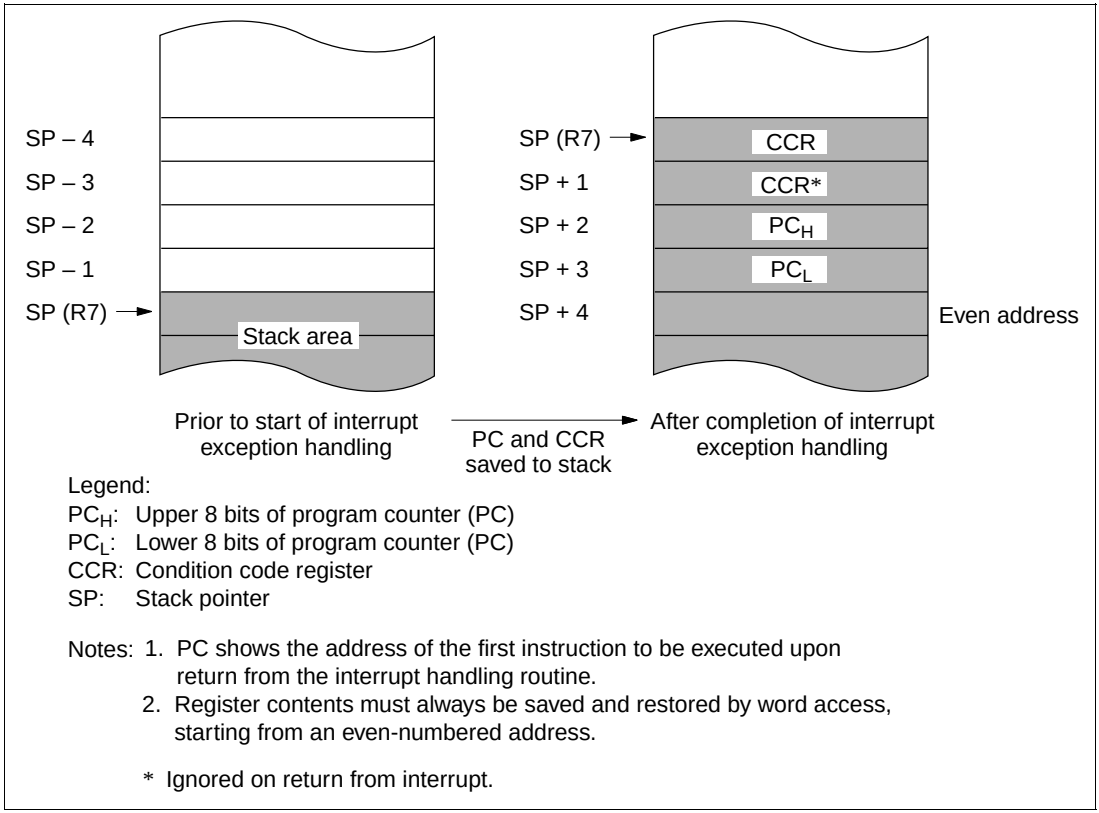


Figure 3.4 Stack State after Completion of Interrupt Exception Handling

Figure 3.5 shows a typical interrupt sequence where the program area is in the on-chip ROM and the stack area is in the on-chip RAM.

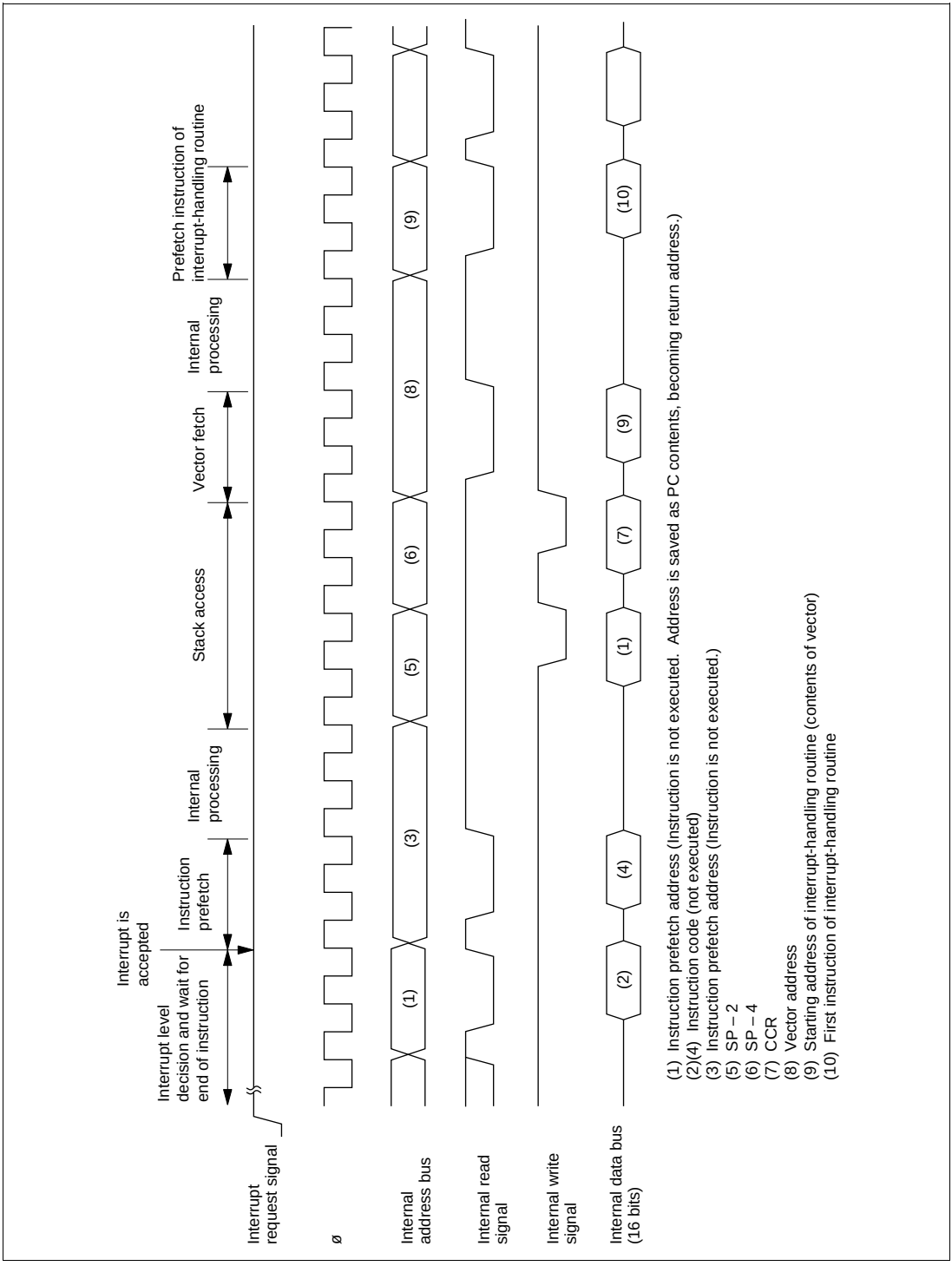


Figure 3.5 Interrupt Sequence

3.3.6 Interrupt Response Time

Table 3.4 shows the number of wait states after an interrupt request flag is set until the first instruction of the interrupt handler is executed.

Table 3.4 Interrupt Wait States

Item	States	Total
Waiting time for completion of executing instruction*	1 to 13	15 to 27
Saving of PC and CCR to stack	4	
Vector fetch	2	
Instruction fetch	4	
Internal processing	4	

Note: * Not including EEPMOV instruction.

3.4 Application Notes

3.4.1 Notes on Stack Area Use

When word data is accessed in the H8/3637 Series, the least significant bit of the address is regarded as 0. Access to the stack always takes place in word size, so the stack pointer (SP: R7) should never indicate an odd address. Use PUSH Rn (MOV.W Rn, @–SP) or POP Rn (MOV.W @SP+, Rn) to save or restore register values.

Setting an odd address in SP may cause a program to crash. An example is shown in figure 3.6.

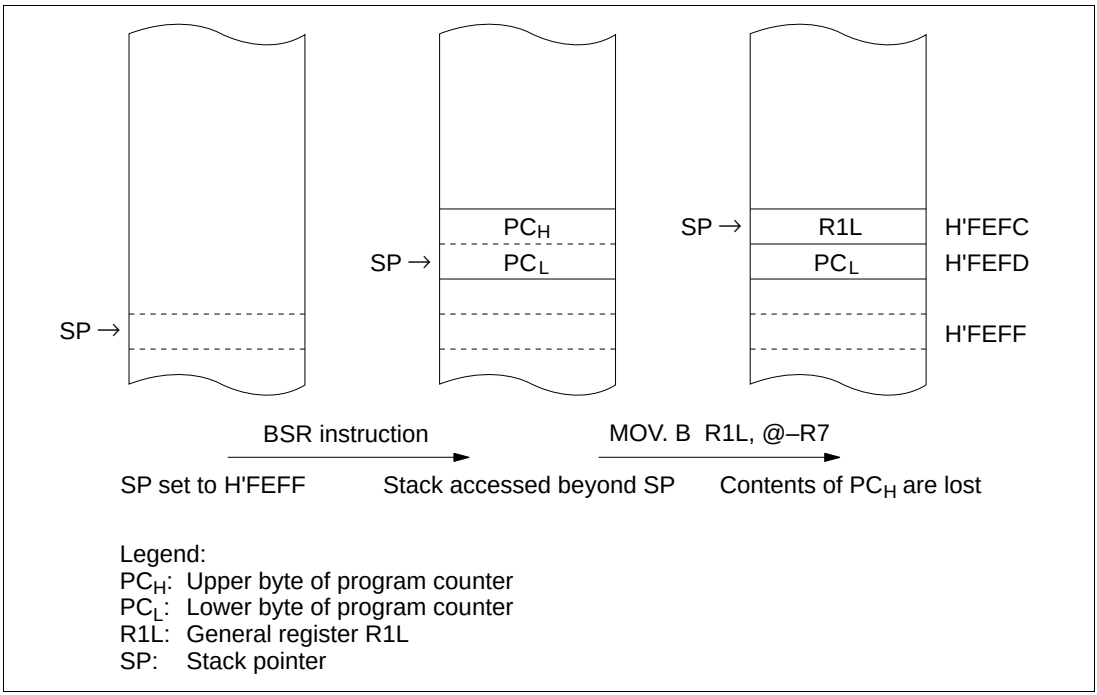


Figure 3.6 Operation when Odd Address is Set in SP

When CCR contents are saved to the stack during interrupt exception handling or restored when RTE is executed, this also takes place in word size. Both the upper and lower bytes of word data are saved to the stack; on return, the even address contents are restored to CCR while the odd address contents are ignored.

3.4.2 Notes on Rewriting Port Mode Registers

When a port mode register is rewritten to switch the functions of external interrupt pins, the following points should be observed.

When an external interrupt pin function is switched by rewriting the port mode register that controls these pins ($\overline{IRQ}_4$ to $\overline{IRQ}_0$, and $\overline{WKP}_7$ to $\overline{WKP}_0$), the interrupt request flag may be set to 1 at the time the pin function is switched, even if no valid interrupt is input at the pin. Be sure to clear the interrupt request flag to 0 after switching pin functions. Table 3.5 shows the conditions under which interrupt request flags are set to 1 in this way.

Table 3.5 Conditions under which Interrupt Request Flag is Set to 1

Interrupt Request Flags Set to 1		Conditions
IRR1	IRRI4	• When PMR2 bit IRQ4 is changed from 0 to 1 while pin $\overline{IRQ}_4$ is low and IEGR bit IEG4 = 0. • When PMR2 bit IRQ4 is changed from 1 to 0 while pin $\overline{IRQ}_4$ is low and IEGR bit IEG4 = 1.
	IRRI3	• When PMR1 bit IRQ3 is changed from 0 to 1 while pin $\overline{IRQ}_3$ is low and IEGR bit IEG3 = 0. • When PMR1 bit IRQ3 is changed from 1 to 0 while pin $\overline{IRQ}_3$ is low and IEGR bit IEG3 = 1.
	IRRI2	• When PMR1 bit IRQ2 is changed from 0 to 1 while pin $\overline{IRQ}_2$ is low and IEGR bit IEG2 = 0. • When PMR1 bit IRQ2 is changed from 1 to 0 while pin $\overline{IRQ}_2$ is low and IEGR bit IEG2 = 1.
	IRRI1	• When PMR1 bit IRQ1 is changed from 0 to 1 while pin $\overline{IRQ}_1$ is low and IEGR bit IEG1 = 0. • When PMR1 bit IRQ1 is changed from 1 to 0 while pin $\overline{IRQ}_1$ is low and IEGR bit IEG1 = 1.
	IRRI0	• When PMR2 bit IRQ0 is changed from 0 to 1 while pin $\overline{IRQ}_0$ is low and IEGR bit IEG0 = 0. • When PMR2 bit IRQ0 is changed from 1 to 0 while pin $\overline{IRQ}_0$ is low and IEGR bit IEG0 = 1.
IWPR	IWPF7	When PMR5 bit WKP7 is changed from 0 to 1 while pin $\overline{WKP}_7$ is low
	IWPF6	When PMR5 bit WKP6 is changed from 0 to 1 while pin $\overline{WKP}_6$ is low
	IWPF5	When PMR5 bit WKP5 is changed from 0 to 1 while pin $\overline{WKP}_5$ is low
	IWPF4	When PMR5 bit WKP4 is changed from 0 to 1 while pin $\overline{WKP}_4$ is low
	IWPF3	When PMR5 bit WKP3 is changed from 0 to 1 while pin $\overline{WKP}_3$ is low
	IWPF2	When PMR5 bit WKP2 is changed from 0 to 1 while pin $\overline{WKP}_2$ is low
	IWPF1	When PMR5 bit WKP1 is changed from 0 to 1 while pin $\overline{WKP}_1$ is low
	IWPF0	When PMR5 bit WKP0 is changed from 0 to 1 while pin $\overline{WKP}_0$ is low

Figure 3.7 shows the procedure for setting a bit in a port mode register and clearing the interrupt request flag.

When switching a pin function, mask the interrupt before setting the bit in the port mode register. After accessing the port mode register, execute at least one instruction (e.g., NOP), then clear the interrupt request flag from 1 to 0. If the instruction to clear the flag is executed immediately after the port mode register access without executing an intervening instruction, the flag will not be cleared.

An alternative method is to avoid the setting of interrupt request flags when pin functions are switched by keeping the pins at the high level so that the conditions in table 3.5 do not occur.

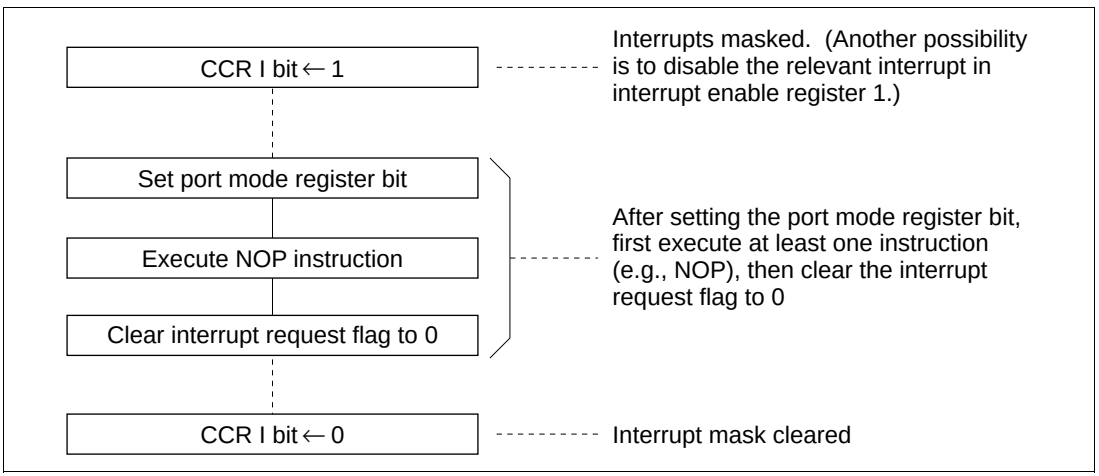


Figure 3.7 Port Mode Register Setting and Interrupt Request Flag Clearing Procedure

Section 4 Clock Pulse Generators

4.1 Overview

Clock oscillator circuitry (CPG: clock pulse generator) is provided on-chip, including both a system clock pulse generator and a subclock pulse generator. The system clock pulse generator consists of a system clock oscillator and system clock dividers. The subclock pulse generator consists of a subclock oscillator circuit and a subclock divider.

4.1.1 Block Diagram

Figure 4.1 shows a block diagram of the clock pulse generators.

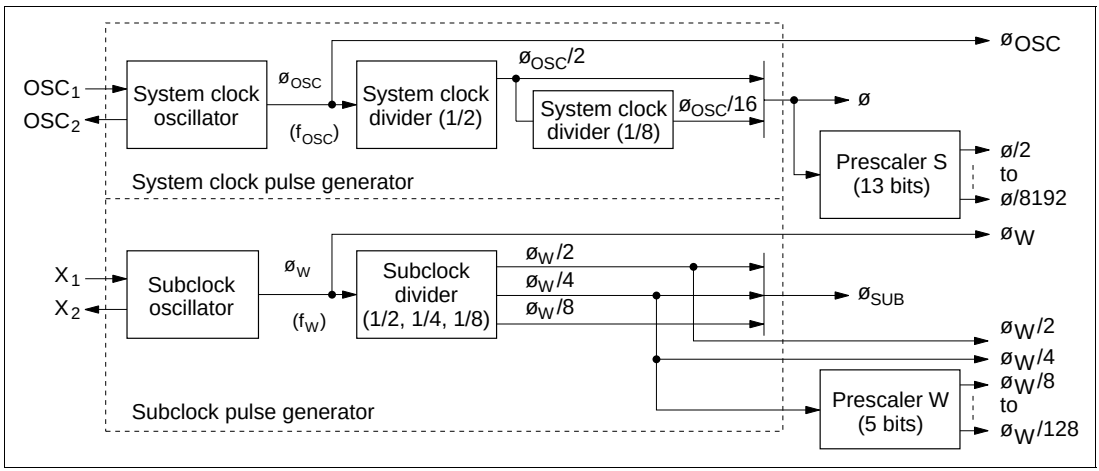


Figure 4.1 Block Diagram of Clock Pulse Generators

4.1.2 System Clock and Subclock

The basic clock signals that drive the CPU and on-chip peripheral modules are ø and $ø_{SUB}$. Four of the clock signals have names: ø is the system clock, $ø_{SUB}$ is the subclock, $ø_{OSC}$ is the oscillator clock, and $ø_W$ is the watch clock.

The clock signals available for use by peripheral modules are $ø_{OSC}$, ø/2, ø/4, ø/8, ø/16, ø/32, ø/64, ø/128, ø/256, ø/512, ø/1024, ø/2048, ø/4096, ø/8192, $ø_W$, $ø_W/2$, $ø_W/4$, $ø_W/8$, $ø_W/16$, $ø_W/32$, $ø_W/64$, and $ø_W/128$. The clock requirements differ from one module to another.

4.2 System Clock Generator

Clock pulse can be supplied to the system clock divider either by connecting a crystal or ceramic oscillator, or by providing external clock input.

Connecting a Crystal Oscillator: Figure 4.2 shows a typical method of connecting a crystal oscillator.

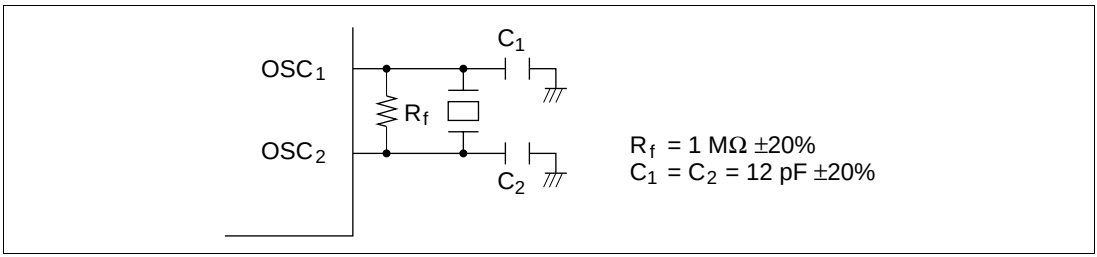


Figure 4.2 Typical Connection to Crystal Oscillator

Figure 4.3 shows the equivalent circuit of a crystal oscillator. An oscillator having the characteristics given in table 4.1 should be used.

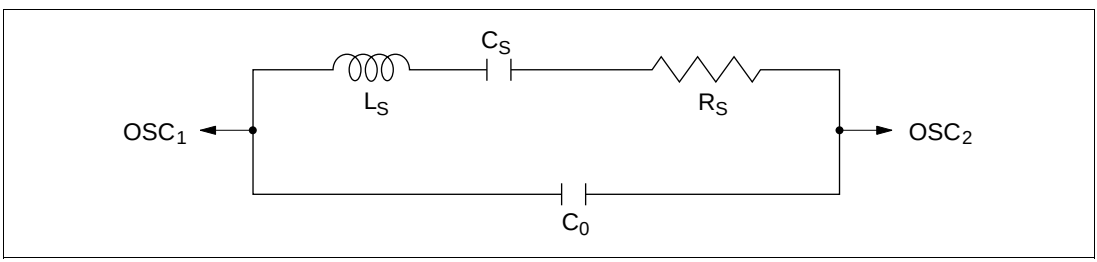


Figure 4.3 Equivalent Circuit of Crystal Oscillator

Table 4.1 Crystal Oscillator Parameters

Frequency	2 MHz	4 MHz	8 MHz	10 MHz
R_S (max)	500 Ω	100 Ω	50 Ω	30 Ω
C_0 (max)	7 pF	7 pF	7 pF	7 pF

Connecting a Ceramic Oscillator: Figure 4.4 shows a typical method of connecting a ceramic oscillator.

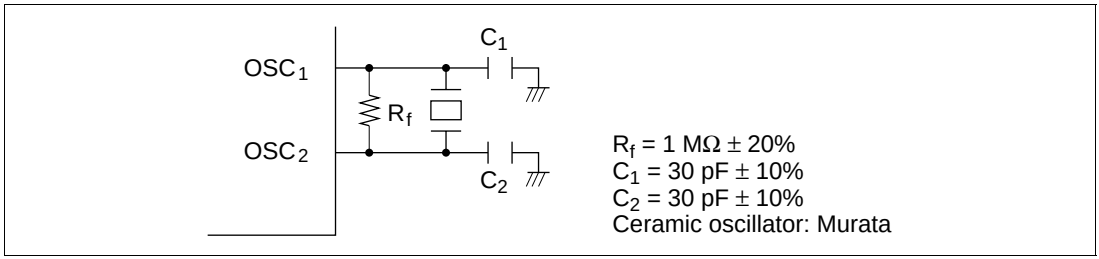


Figure 4.4 Typical Connection to Ceramic Oscillator

Notes on Board Design: When generating clock pulses by connecting a crystal or ceramic oscillator, pay careful attention to the following points.

Avoid running signal lines close to the oscillator circuit, since the oscillator may be adversely affected by induction currents. (See figure 4.5.)

The board should be designed so that the oscillator and load capacitors are located as close as possible to pins OSC_1 and OSC_2.

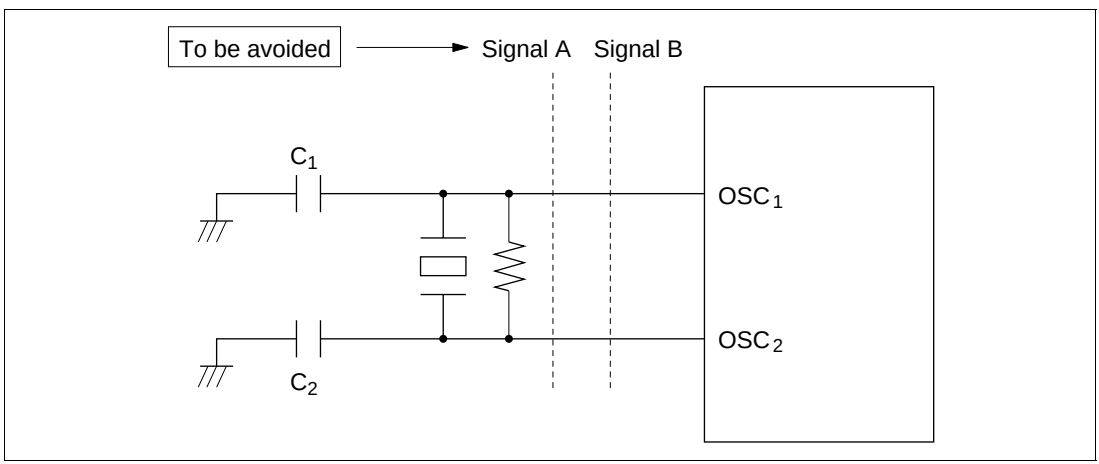


Figure 4.5 Board Design of Oscillator Circuit

External Clock Input Method: Connect an external clock signal to pin OSC_1, and leave pin OSC_2 open. Figure 4.6 shows a typical connection.

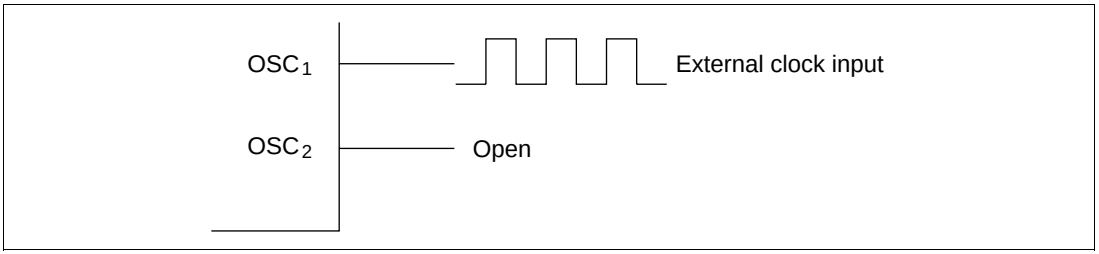


Figure 4.6 External Clock Input (Example)

Frequency	Oscillator Clock ($ø_{osc}$)
Duty cycle	45% to 55%

4.3 Subclock Generator

Connecting a 32.768-kHz Crystal Oscillator: Clock pulses can be supplied to the subclock divider by connecting a 32.768-kHz crystal oscillator, as shown in figure 4.7. Follow the same precautions as noted in 4.2, Notes on Board Design.

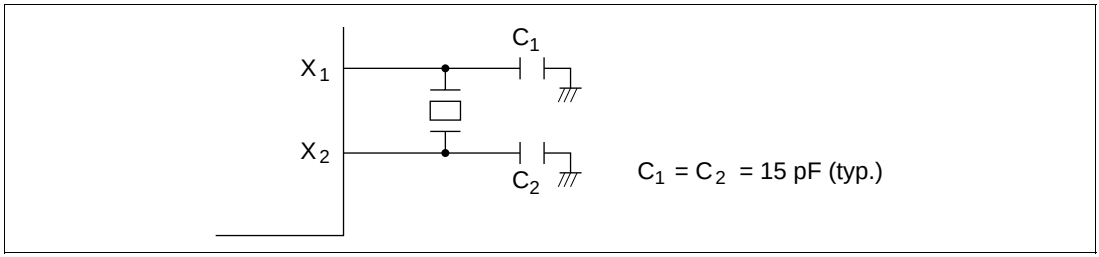


Figure 4.7 Typical Connection to 32.768-kHz Crystal Oscillator

Figure 4.8 shows the equivalent circuit of the 32.768-kHz crystal oscillator.

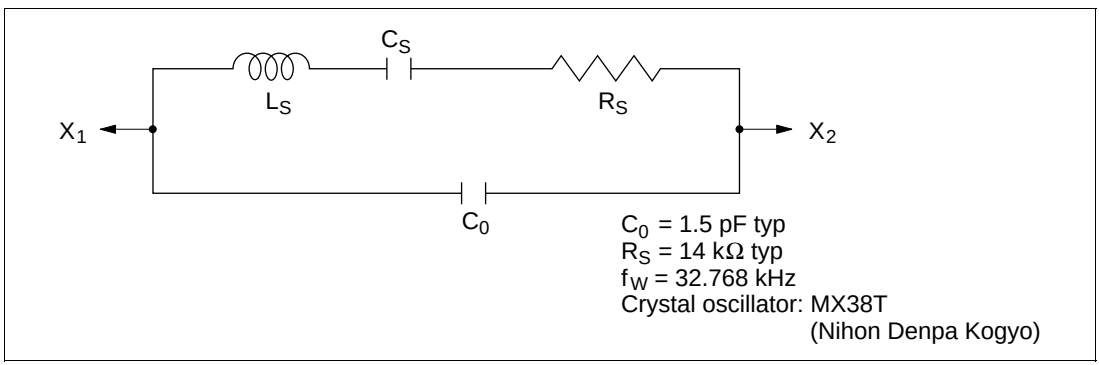


Figure 4.8 Equivalent Circuit of 32.768-kHz Crystal Oscillator

Pin Connection when Not Using Subclock: When the subclock is not used, connect pin X_1 to V_{CC} and leave pin X_2 open, as shown in figure 4.9.

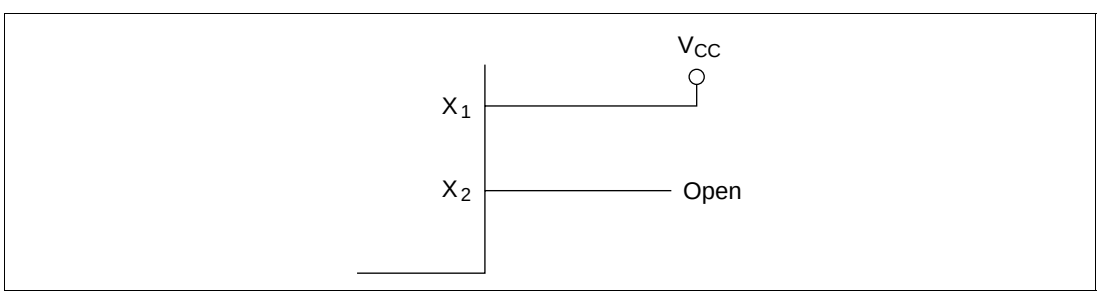


Figure 4.9 Pin Connection when not Using Subclock

4.4 Prescalers

The H8/3637 Series is equipped with two on-chip prescalers having different input clocks (prescaler S and prescaler W). Prescaler S is a 13-bit counter using the system clock (ø) as its input clock. Its prescaled outputs provide internal clock signals for on-chip peripheral modules. Prescaler W is a 5-bit counter using a 32.768-kHz signal divided by 4 ($ø_W/4$) as its input clock. Its prescaled outputs are used by timer A as a time base for timekeeping.

Prescaler S (PSS): Prescaler S is a 13-bit counter using the system clock (ø) as its input clock. It is incremented once per clock period.

Prescaler S is initialized to H'0000 by a reset, and starts counting on exit from the reset state.

In standby mode, watch mode, subactive mode, and subsleep mode, the system clock pulse generator stops. Prescaler S also stops and is initialized to H'0000.

The CPU cannot read or write prescaler S.

The output from prescaler S is shared by the on-chip peripheral modules. The divider ratio can be set separately for each on-chip peripheral function.

In active (medium-speed) mode the clock input to prescaler S is $ø_{OSC}/16$.

Prescaler W (PSW): Prescaler W is a 5-bit counter using a 32.768 kHz signal divided by 4 ($ø_W/4$) as its input clock.

Prescaler W is initialized to H'00 by a reset, and starts counting on exit from the reset state.

Even in standby mode, watch mode, subactive mode, or subsleep mode, prescaler W continues functioning so long as clock signals are supplied to pins X_1 and X_2.

Prescaler W can be reset by setting 1s in bits TMA3 and TMA2 of timer mode register A (TMA).

Output from prescaler W can be used to drive timer A, in which case timer A functions as a time base for timekeeping.

4.5 Note on Oscillators

Oscillator characteristics of both the masked ROM and ZTAT™ versions are closely related to board design and should be carefully evaluated by the user, referring to the examples shown in this section and figure 4.10, Example of Crystal and Ceramic Oscillator Layout. Oscillator circuit constants will differ depending on the oscillator element, stray capacitance in its interconnecting circuit, and other factors. Suitable constants should be determined in consultation with the oscillator element manufacturer. Design the circuit so that the oscillator element never receives voltages exceeding its maximum rating.

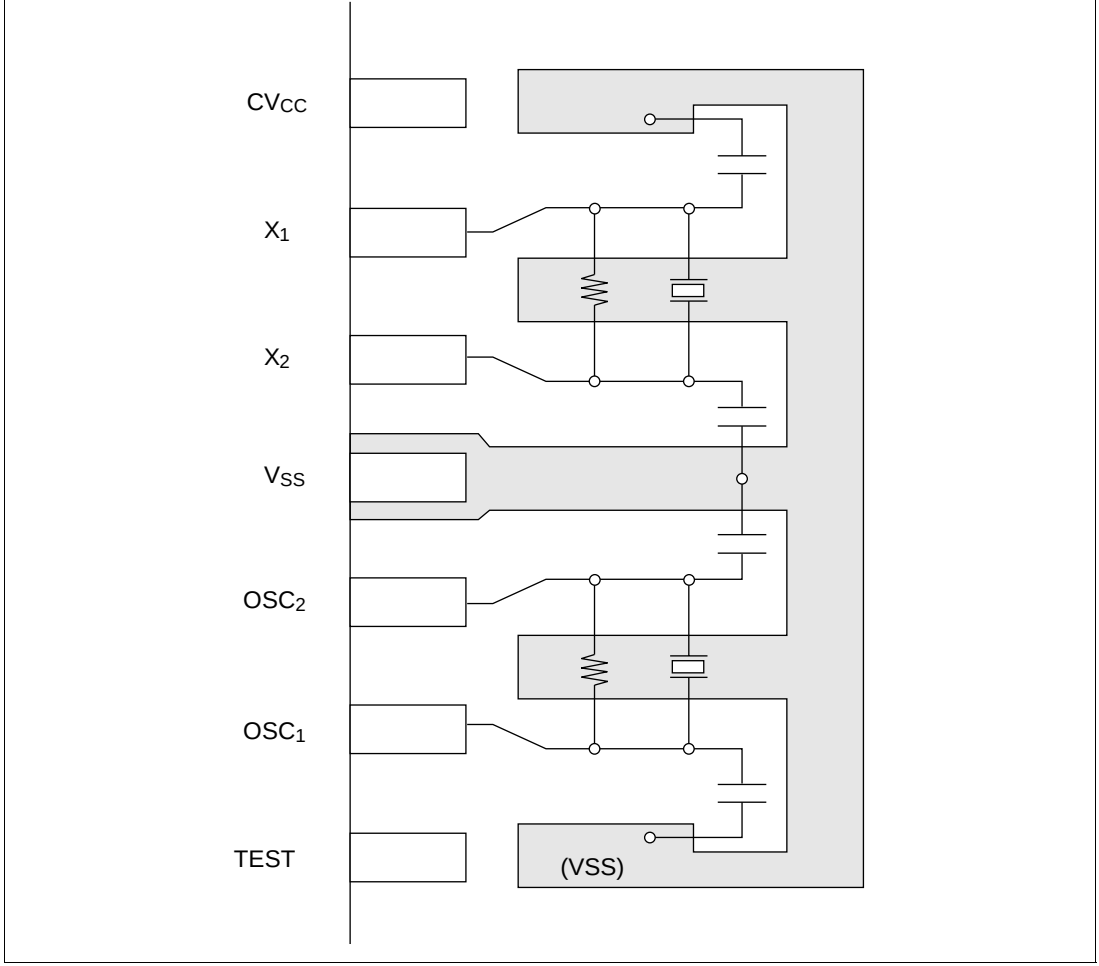


Figure 4.10 Example of Crystal and Ceramic Oscillator Layout.

Section 5 Power-Down Modes

5.1 Overview

The H8/3627 Series has seven modes of operation after a reset. These include six power-down modes, in which power dissipation is significantly reduced.

Table 5.1 gives a summary of the seven operation modes.

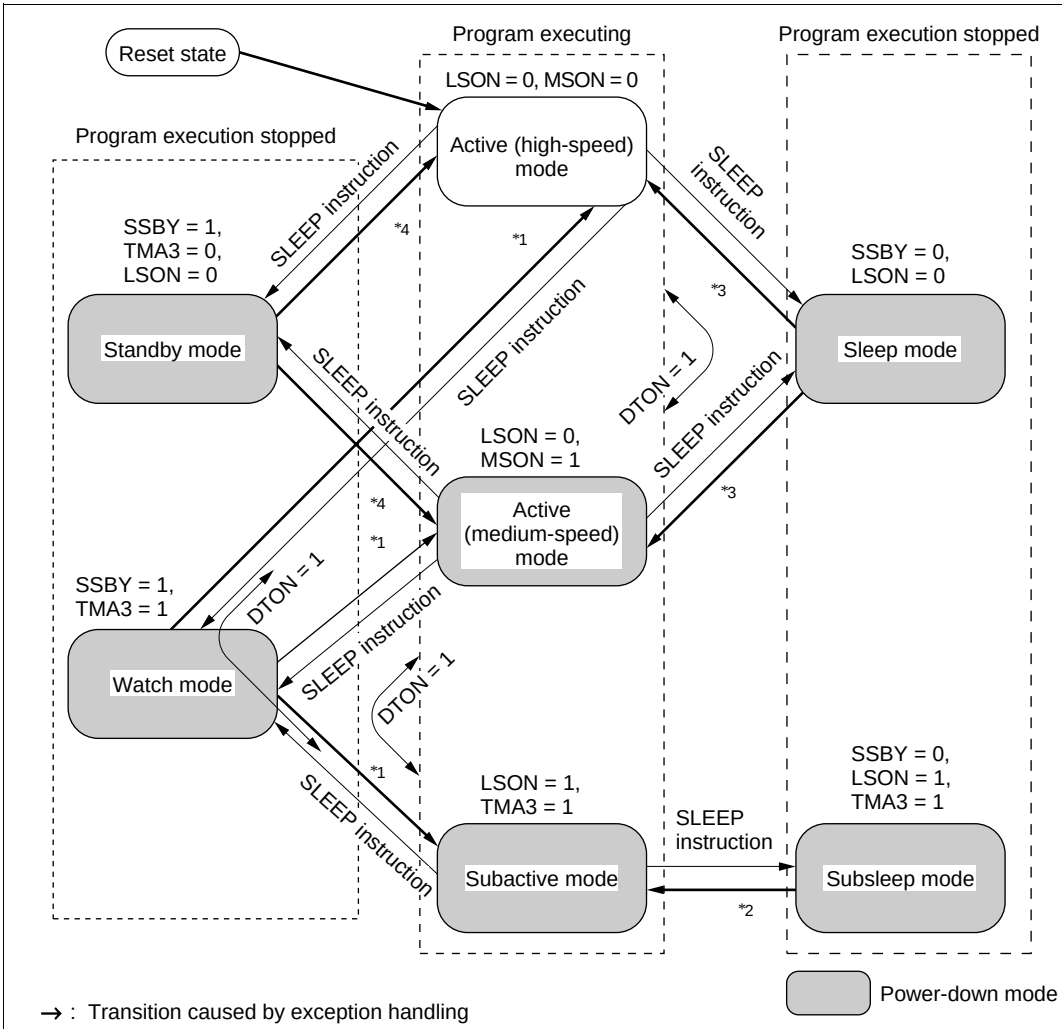
Table 5.1 Operation Modes

Operating Mode	Description
Active (high-speed) mode	The CPU runs on the system clock, executing program instructions at high speed
Active (medium-speed) mode	The CPU runs on the system clock, executing program instructions at reduced speed
Subactive mode	The CPU runs on the subclock, executing program instructions at reduced speed
Sleep mode	The CPU halts. On-chip peripheral modules continue to operate on the system clock.
Subsleep mode	The CPU halts. Timer A, and timer G, continue to operate on the subclock.
Watch mode	The CPU halts. The time-base function of Timer A continues to operate on the subclock.
Standby mode	The CPU and all on-chip peripheral modules stop operating

All but the active (high-speed) mode are power-down modes.

In this section the two active modes (high-speed and medium-speed) are referred to collectively as active mode.

Figure 5.1 shows the transitions among these operation modes. Table 5.2 indicates the internal states in each mode.



→ : Transition caused by exception handling

A transition between different modes cannot be made to occur simply because an interrupt request is generated. Make sure that the interrupt is accepted and interrupt handling is performed.
Details on the mode transition conditions are given in the explanations of each mode, in sections 5.2 through 5.8.

Notes: 1. Timer A interrupt, IRQ_0 interrupt, WKP_0 to WKP_7 interrupts
2. Timer A interrupt, timer G interrupt, IRQ_0 to IRQ_4 interrupts, WKP_0 to WKP_7 interrupts
3. All interrupts
4. IRQ_0 interrupt, IRQ_1 interrupt, WKP_0 to WKP_7 interrupts

Figure 5.1 Operation Mode Transition Diagram

Table 5.2 Internal State in Each Operation Mode

Function		Active Mode		Sleep Mode	Watch Mode	Subactive Mode	Subsleep Mode	Standby Mode
		High Speed	Medium Speed					
System clock oscillator		Functional	Functional	Functional	Stopped	Stopped	Stopped	Stopped
Subclock oscillator		Functional	Functional	Functional	Functional	Functional	Functional	Functional
CPU operation	Instructions	Functional	Functional	Stopped	Stopped	Functional	Stopped	Stopped
	RAM			Retained	Retained		Retained	Retained
	Registers							
	I/O							Retained*1
External interrupts	IRQ_0	Functional	Functional	Functional	Functional	Functional	Functional	Functional
	IRQ_1				Retained*4			
	IRQ_2							Retained*4
	IRQ_3							
	IRQ_4							
	WKP_0	Functional	Functional	Functional	Functional	Functional	Functional	Functional
	WKP_1							
	WKP_2							
	WKP_3							
	WKP_4							
	WKP_5							
	WKP_6							
	WKP_7							
Peripheral module functions	Timer A	Functional	Functional	Functional	Functional*3	Functional*3	Functional*3	Retained
	Timer F				Retained	Retained	Retained	
	Timer G					Functional/ Retained*2	Functional/ Retained*2	
	SCI1	Functional	Functional	Functional	Retained	Retained	Retained	Retained
	SCI3				Reset	Reset	Reset	Reset
	DTMF	Functional	Functional	Functional	Reset	Reset	Reset	Reset
	A/D	Functional	Functional	Functional	Retained	Retained	Retained	Retained

Notes: 1. Register contents held; high-impedance output.
2. Functional only if $\phi_W/2$ internal clock is selected; otherwise stopped and retained.
3. Functional when timekeeping time-base function is selected.
4. External interrupt requests are ignored. The interrupt request register contents are not affected.

5.1.1 System Control Registers

The operation mode is selected using the system control registers described in table 5.3.

Table 5.3 System Control Register

Name	Abbreviation	R/W	Initial Value	Address
System control register 1	SYSCR1	R/W	H'07	H'FFF0
System control register 2	SYSCR2	R/W	H'E0	H'FFF1

System Control Register 1 (SYSCR1)

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

SYSCR1 is an 8-bit read/write register for control of the power-down modes.

Upon reset, SYSCR1 is initialized to H'07.

Bit 7—Software Standby (SSBY): This bit designates transition to standby mode or watch mode.

Bit 7: SSBY	Description
0	• When a SLEEP instruction is executed in active mode, a transition is made to sleep mode. • When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode. (initial value)
1	• When a SLEEP instruction is executed in active mode, a transition is made to standby mode or watch mode. • When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode.

Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0): These bits designate the time the CPU and peripheral modules wait for stable clock operation after exiting from standby mode or watch mode to active mode due to an interrupt. The designation should be made according to the clock frequency so that the waiting time is at least 10 ms.

Bit 6: STS2	Bit 5: STS1	Bit 4: STS0	Description	
0	0	0	Wait time = 8,192 states	(initial value)
		1	Wait time = 16,384 states	
	1	0	Wait time = 32,768 states	
		1	Wait time = 65,536 states	
1	*	*	Wait time = 131,072 states	

Note: * Don't care

Bit 3—Low Speed on Flag (LSON): This bit chooses the system clock (ø) or subclock ($ø_{SUB}$) as the CPU operating clock when watch mode is cleared. The resulting operation mode depends on the combination of other control bits and interrupt input.

Bit 3: LSON	Description	
0	The CPU operates on the system clock (ø)	(initial value)
1	The CPU operates on the subclock ($ø_{SUB}$)	

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

System Control Register 2 (SYSCR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	NESEL	DTON	MSON	SA1	SA0
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

SYSCR2 is an 8-bit read/write register for power-down mode control.

Upon reset, SYSCR2 is initialized to H'E0.

Bits 7 to 5—Reserved Bits: These bits are reserved; they are always read as 1, and cannot be modified.

Bit 4— Noise Elimination Sampling Frequency Select (NESEL): This bit selects the frequency at which the watch clock signal ($ø_W$) generated by the subclock pulse generator is sampled, in relation to the oscillator clock ($ø_{OSC}$) generated by the system clock pulse generator. When $ø_{OSC}$ = 2 to 10 MHz, clear NESEL to 0.

Bit 4: NESEL	Description	
0	Sampling rate is $ø_{OSC}$/16	(initial value)
1	Sampling rate is $ø_{OSC}$/4	

Bit 3—Direct Transfer on Flag (DTON): This bit designates whether or not to make direct transitions among active (high-speed), active (medium-speed) and subactive mode when a SLEEP instruction is executed. The mode to which the transition is made after the SLEEP instruction is executed depends on a combination of this and other control bits.

Bit 3: DTON	Description
0	• When a SLEEP instruction is executed in active mode, a transition is made to standby mode, watch mode, or sleep mode. (initial value) • When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode or subsleep mode.
1	• When a SLEEP instruction is executed in active (high-speed) mode, a direct transition is made to active (medium-speed) mode if SSBY = 0, MSON = 1, and LSON = 0, or to subactive mode if SSBY = 1, TMA3 = 1, and LSON = 1. • When a SLEEP instruction is executed in active (medium-speed) mode, a direct transition is made to active (high-speed) mode if SSBY = 0, MSON = 0, and LSON = 0, or to subactive mode if SSBY = 1, TMA3 = 1, and LSON = 1. • When a SLEEP instruction is executed in subactive mode, a direct transition is made to active (high-speed) mode if SSBY = 1, TMA3 = 1, LSON = 0, and MSON = 0, or to active (medium-speed) mode if SSBY = 1, TMA3 = 1, LSON = 0, and MSON = 1.

Bit 2—Medium Speed on Flag (MSON): After standby, watch, or sleep mode is cleared, this bit selects active (high-speed) or active (medium-speed) mode.

Bit 2: MSON	Description	
0	Operation is in active (high-speed) mode	(initial value)
1	Operation is in active (medium-speed) mode	

Bits 1 and 0—Subactive Mode Clock Select (SA1 and SA0): These bits select the CPU clock rate ($ø_W/2$, $ø_W/4$, or $ø_W/8$) in subactive mode. SA1 and SA0 cannot be modified in subactive mode.

Bit 1: SA1	Bit 0: SA0	Description	
0	0	$ø_W/8$	(initial value)
	1	$ø_W/4$	
1	*	$ø_W/2$	

Note: * Don't care

5.2 Sleep Mode

5.2.1 Transition to Sleep Mode

The system goes from active mode to sleep mode when a SLEEP instruction is executed while the SSBY and LSON bits in system control register 1 (SYSCR1) are cleared to 0. In sleep mode CPU operation is halted but the on-chip peripheral functions are operational. The CPU register contents are retained.

5.2.2 Clearing Sleep Mode

Sleep mode is cleared by an interrupt (timer A, timer F, timer G, IRQ_0 to IRQ_4, WKP_0 to WKP_7, SCI1, SCI3, A/D converter) or by reset input.

Clearing by Interrupt: When an interrupt is requested, sleep mode is cleared and interrupt exception handling starts. Operation resumes in active (high-speed) mode if MSON = 0 in SYSCR2, or active (medium-speed) mode if MSON = 1. Sleep mode is not cleared if the I bit of the condition code register (CCR) is set to 1 or the particular interrupt is disabled in the interrupt enable register.

Clearing by Reset Input: When the $\overline{RES}$ pin goes low, the CPU goes into the reset state and sleep mode is cleared.

5.3 Standby Mode

5.3.1 Transition to Standby Mode

The system goes from active mode to standby mode when a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1, the LSON bit is cleared to 0, and bit TMA3 in timer mode register A (TMA) is cleared to 0. In standby mode the clock pulse generator stops, so the CPU and on-chip peripheral modules stop functioning. As long as a minimum required voltage is applied, the contents of the CPU registers and some on-chip peripheral function internal registers, and data in the on-chip RAM, will be retained. The I/O ports go to the high-impedance state.

5.3.2 Clearing Standby Mode

Standby mode is cleared by an interrupt (IRQ_0, IRQ_1, WKP_0 to WKP_7) or by input at the $\overline{RES}$ pin.

Clearing by Interrupt: When an interrupt is requested, the system clock pulse generator starts. After the time set in bits STS2 to STS0 in SYSCR1 has elapsed, a stable system clock signal is supplied to the entire chip, standby mode is cleared, and interrupt exception handling starts. Operation resumes in active (high-speed) mode if MSON = 0 in SYSCR2, or active (medium-speed) mode if MSON = 1. Standby mode is not cleared if the I bit of CCR is set to 1 or the particular interrupt is disabled in the interrupt enable register.

Clearing by $\overline{RES}$ Input: When the $\overline{RES}$ pin goes low, the system clock pulse generator starts. After the pulse generator output has stabilized, if the $\overline{RES}$ pin is driven high, the CPU starts reset exception handling.

Since system clock signals are supplied to the entire chip as soon as the system clock pulse generator starts functioning, the $\overline{RES}$ pin should be kept at the low level until the pulse generator output stabilizes.

5.3.3 Oscillator Settling Time after Standby Mode is Cleared

Bits STS2 to STS0 in SYSCR1 should be set as follows.

When a Crystal Oscillator is Used: Table 5.4 gives settings for various operating frequencies. Set bits STS2 to STS0 for a waiting time of at least 10 ms.

Table 5.4 Clock Frequency and Settling Time (times are in ms)

STS2	STS1	STS0	Waiting Time	5 MHz	4 MHz	2 MHz	1 MHz	0.5 MHz
0	0	0	8,192 states	1.6	2.0	4.1	8.2	16.4
0	0	1	16,384 states	3.2	4.1	8.2	16.4	32.8
0	1	0	32,768 states	6.6	8.2	16.4	32.8	65.5
0	1	1	65,536 states	13.1	16.4	32.8	65.5	131.1
1	*	*	131,072 states	26.2	32.8	65.5	131.1	262.1

Note: * Don't care

When an External Clock is Used: Any values may be set. Normally the minimum time (STS2 = STS1 = STS0 = 0) should be set.

5.3.4 Transition to Standby Mode and Pin States

The system goes from active (high-speed) mode or active (medium-speed) mode to standby mode when a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, and bit TMA3 in TMA is cleared to 0. At the same time, pins go to the high-impedance state (except pins with MOS pull-up turned on). The timing in this case is shown in figure 5.2.

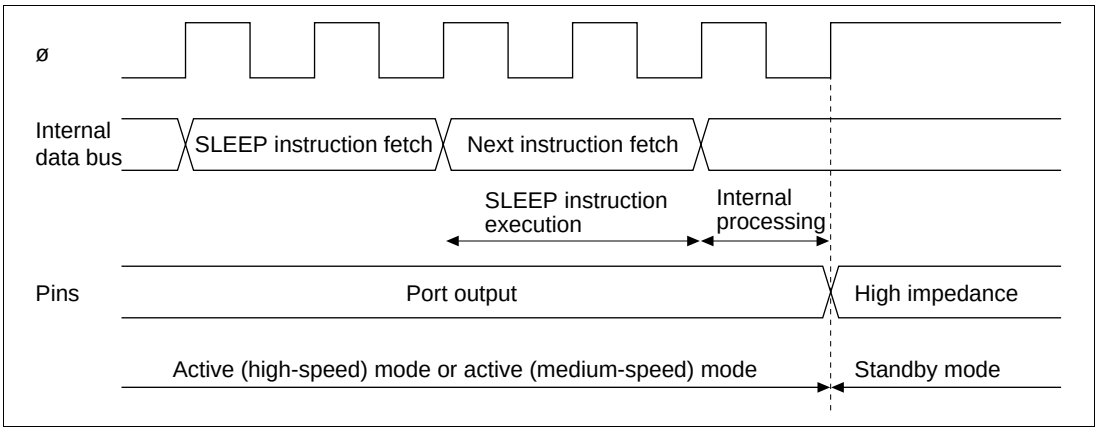


Figure 5.2 Transition to Standby Mode and Pin States

5.4 Watch Mode

5.4.1 Transition to Watch Mode

The system goes from active or subactive mode to watch mode when a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1 and bit TMA3 in TMA is set to 1.

In watch mode, operation of on-chip peripheral modules other than timer A is halted. As long as a minimum required voltage is applied, the contents of CPU registers and some registers of the on-chip peripheral modules*, and the on-chip RAM contents, are retained. I/O ports keep the same states as before the transition.

Note: * The contents of SCI3, DTMF generator registers are reset.

5.4.2 Clearing Watch Mode

Watch mode is cleared by an interrupt (timer A, IRQ_0, WKP_0 to WKP_7) or by a low input at the $\overline{RES}$ pin.

Clearing by Interrupt: When watch mode is cleared by a timer A, IRQ_0, or WKP_0 to WKP_7 interrupt request, the mode to which a transition is made depends on the settings of LSON in SYSCR1 and MSON in SYSCR2. If both LSON and MSON are cleared to 0, transition is to active (high-speed) mode; if LSON = 0 and MSON = 1, transition is to active (medium-speed) mode; if LSON = 1, transition is to subactive mode. When the transition is to active mode, after the time set in SYSCR1 bits STS2 to STS0 has elapsed, a stable clock signal is supplied to the entire chip, watch mode is cleared, and interrupt exception handling starts. Watch mode is not cleared if the I bit of CCR is set to 1 or the particular interrupt is disabled in the interrupt enable register.

Clearing by $\overline{RES}$ Input: Clearing by $\overline{RES}$ pin is the same as for standby mode; see 5.3.2, Clearing Standby Mode.

5.4.3 Oscillator Settling Time after Watch Mode is Cleared

The waiting time is the same as for standby mode; see 5.3.3, Oscillator Settling Time after Standby Mode is Cleared.

5.5 Subsleep Mode

5.5.1 Transition to Subsleep Mode

The system goes from subactive mode to subsleep mode when a SLEEP instruction is executed while the SSBY bit in SYSCR1 is cleared to 0, LSON is set to 1, and bit TMA3 in TMA is set to 1.

In subsleep mode, operation of on-chip peripheral modules other than timer A and timer G is halted. As long as a minimum required voltage is applied, the contents of CPU registers and some registers of the on-chip peripheral modules*, and the on-chip RAM contents, are retained. I/O ports keep the same states as before the transition.

Note: * The contents of SCI3, DTMF generator registers are reset.

5.5.2 Clearing Subsleep Mode

Subsleep mode is cleared by an interrupt (timer A, timer G, IRQ_0 to IRQ_4, WKP_0 to WKP_7) or by a low input at the $\overline{RES}$ pin.

Clearing by Interrupt: When an interrupt is requested, subsleep mode is cleared and interrupt exception handling starts. Subsleep mode is not cleared if the I bit of CCR is set to 1 or the particular interrupt is disabled in the interrupt enable register.

Clearing by $\overline{RES}$ Input: Clearing by $\overline{RES}$ input is the same as for standby mode; see 5.3.2, Clearing Standby Mode.

5.6 Subactive Mode

5.6.1 Transition to Subactive Mode

Subactive mode is entered from watch mode if a timer A, IRQ_0, or WKP_0 to WKP_7 interrupt is requested while the LSON bit in SYSCR1 is set to 1. From subsleep mode, subactive mode is entered if a timer A, timer G, IRQ_0 to IRQ_4, or WKP_0 to WKP_7 interrupt is requested. A transition to subactive mode does not take place if the I bit of CCR is set to 1 or the particular interrupt is disabled in the interrupt enable register.

The contents of SCI3, DTMF generator registers are reset.

5.6.2 Clearing Subactive Mode

Subactive mode is cleared by a SLEEP instruction or by a low input at the $\overline{RES}$ pin.

Clearing by SLEEP Instruction: If a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1 and bit TMA3 in TMA is set to 1, subactive mode is cleared and watch mode is entered. If a SLEEP instruction is executed while SSBY = 0 and LSON = 1 in SYSCR1 and TMA3 = 1 in TMA, subsleep mode is entered. Direct transfer to active mode is also possible; see 5.8, Direct Transfer, below.

Clearing by $\overline{RES}$ Pin: Clearing by $\overline{RES}$ pin is the same as for standby mode; see 5.3.2, Clearing Standby Mode.

5.6.3 Operating Frequency in Subactive Mode

The operating frequency in subactive mode is set in bits SA1 and SA0 in SYSCR2. The choices are $ø_W/2$, $ø_W/4$, and $ø_W/8$.

5.7 Active (medium-speed) Mode

5.7.1 Transition to Active (medium-speed) Mode

If the MSON bit in SYSCR2 is set to 1 while the LSON bit in SYSCR1 is cleared to 0, a transition to active (medium-speed) mode results from IRQ_0, IRQ_1, or WKP_0 to WKP_7 interrupts in standby mode, timer A, IRQ_0, or WKP_0 to WKP_7 interrupts in watch mode, or any interrupt in sleep mode. A transition to active (medium-speed) mode does not take place if the I bit of CCR is set to 1 or the particular interrupt is disabled in the interrupt enable register.

5.7.2 Clearing Active (medium-speed) Mode

Active (medium-speed) mode is cleared by a SLEEP instruction or by a low input at the reset.

Clearing by SLEEP Instruction: A transition to standby mode takes place if a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, and bit TMA3 in TMA is cleared to 0. The system goes to watch mode if the SSBY bit in SYSCR1 is set to 1 and bit TMA3 in TMA is set to 1 when a SLEEP instruction is executed. Sleep mode is entered if both SSBY and LSON are cleared to 0 when a SLEEP instruction is executed. Direct transfer to active (high-speed) mode or to subactive mode is also possible. See 5.8, Direct Transfer, below for details.

Clearing by Reset: When the $\overline{\text{RES}}$ pin goes low, the CPU goes into the reset state and active (medium-speed) mode is cleared.

5.7.3 Operating Frequency in Active (medium-speed) Mode

In active (medium-speed) mode, the CPU is clocked at 1/8 the frequency in active (high-speed) mode. The DTMF generator, however, continues to operate on the OSC clock ($ø_{OSC}$).

5.8 Direct Transfer

5.8.1 Overview

The CPU can execute programs in three modes: active (high-speed) mode, active (medium-speed) mode, and subactive mode. A direct transfer is a transition among these three modes without the stopping of program execution. A direct transfer can be made by executing a SLEEP instruction while the DTON bit in SYSCR2 is set to 1. After the mode transition, direct transfer interrupt exception handling starts.

If the direct transfer interrupt is disabled in interrupt enable register 2 (IENR2), a transition is made instead to sleep mode or watch mode. Note that if a direct transition is attempted while the I bit in CCR is set to 1, sleep mode or watch mode will be entered, and it will be impossible to clear the resulting mode by means of an interrupt.

Direct Transfer from Active (High-Speed) Mode to Active (Medium-Speed) Mode: When a SLEEP instruction is executed in active (high-speed) mode while the SSBY and LSON bits in SYSCR1 are cleared to 0, the MSON bit in SYSCR2 is set to 1, and the DTON bit in SYSCR2 is set to 1, a transition is made to active (medium-speed) mode via sleep mode.

Direct Transfer from Active (High-Speed) Mode to Active (High-Speed) Mode: When a SLEEP instruction is executed in active (medium-speed) mode while the SSBY and LSON bits in SYSCR1 are cleared to 0, the MSON bit in SYSCR2 is cleared to 0, and the DTON bit in SYSCR2 is set to 1, a transition is made to active (high-speed) mode via sleep mode.

Direct Transfer from Active (High-Speed) Mode to Subactive Mode: When a SLEEP instruction is executed in active (high-speed) mode while the SSBY and LSON bits in SYSCR1 are set to 1, the DTON bit in SYSCR2 is set to 1, and bit TMA3 in TMA is set to 1, a transition is made to subactive mode via watch mode.

Direct Transfer from Subactive Mode to Active (High-Speed) Mode: When a SLEEP instruction is executed in subactive mode while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, the MSON bit in SYSCR2 is cleared to 0, the DTON bit in SYSCR2 is set to 1, and bit TMA3 in TMA is set to 1, a transition is made directly to active (high-speed) mode via watch mode after the waiting time set in SYSCR1 bits STS2 to STS0 has elapsed.

Direct Transfer from Active (Medium-Speed) Mode to Subactive Mode: When a SLEEP instruction is executed in active (medium-speed) mode while the SSBY and LSON bits in SYSCR1 are set to 1, the DTON bit in SYSCR2 is set to 1, and bit TMA3 in TMA is set to 1, a transition is made to subactive mode via watch mode.

Direct Transfer from Subactive Mode to Active (Medium-Speed) Mode: When a SLEEP instruction is executed in subactive mode while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, the MSON bit in SYSCR2 is set to 1, the DTON bit in SYSCR2 is set

to 1, and bit TMA3 in TMA is set to 1, a transition is made directly to active (medium-speed) mode via watch mode after the waiting time set in SYSCR1 bits STS2 to STS0 has elapsed.

5.8.2 Direct Transfer Time

Time for Direct Transfer from Active (High-Speed) Mode to Active (Medium-Speed) Mode: When a SLEEP instruction is executed in active (high-speed) mode while the SSBY bit in SYSCR1 is cleared to 0, the LSON bit in SYSCR1 is cleared to 0, the MSON bit in SYSCR2 is set to 1, and the DTON bit in SYSCR2 is set to 1, a transition is made directly to active (medium-speed) mode. In this case, the time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transfer time) is given by equation (1) below:

Direct transfer time ={ (Number of SLEEP instruction execution states) + (number of internal processing states) } × (t_{cyc} before transition) + (number of interrupt exception handling execution states) × (t_{cyc} after transition) ..... (1)

Example: H8/3627 Series direct transfer time = $(2 + 1) \times 2t_{osc} + 14 \times 16t_{osc} = 230t_{osc}$

Legend: t_{osc}: OSC clock cycle time
t_{cyc}: System clock (ø) cycle time

Time for Direct Transfer from Active (Medium-Speed) Mode to Active (High-Speed) Mode: When a SLEEP instruction is executed in active (medium-speed) mode while the SSBY bit in SYSCR1 is cleared to 0, the LSON bit in SYSCR1 is cleared to 0, the MSON bit in SYSCR2 is cleared to 0, and the DTON bit in SYSCR2 is set to 1, a transition is made directly to active (high-speed) mode. In this case, the time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transfer time) is given by equation (2) below:

Direct transfer time ={ (Number of SLEEP instruction execution states) + (number of internal processing states) } × (t_{cyc} before transition) + (number of interrupt exception handling execution states) × (t_{cyc} after transition) ..... (2)

Example: H8/3627 Series direct transfer time = $(2 + 1) \times 16t_{osc} + 14 \times 2t_{osc} = 76t_{osc}$

Legend: t_{osc}: OSC clock cycle time
t_{cyc}: System clock (ø) cycle time

Time for Direct Transfer from Subactive Mode to Active (High-Speed) Mode: When a SLEEP instruction is executed in subactive mode while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, the MSON bit in SYSCR2 is cleared to 0, the DTON bit in SYSCR2 is set to 1, and bit TMA3 in TMA is set to 1, a transition is made directly to active (high-speed) mode. In this case, the time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transfer time) is given by equation (3) below:

Direct transfer time = { (Number of SLEEP instruction execution states) + (number of internal processing states) } × (t_{subcyc} before transition) + { (standby time set in STS2 to STS0) + (number of interrupt exception handling execution states) } × (t_{cyc} after transition) ..... (3)

Example: H8/3627 Series direct transfer time = $(2 + 1) \times 8t_w + (8192 + 14) \times 2t_{osc} = 24t_w + 16412t_{osc}$
(When $ø_w$/8 CPU operating clock and 8192-state standby time are selected)

Legend:
t_{osc}: OSC clock cycle time
t_w: Watch clock cycle time
t_{cyc}: System clock (ø) cycle time
t_{subcyc}: Subclock ($ø_{SUB}$) cycle time

Time for Direct Transfer from Subactive Mode to Active (Medium-Speed) Mode: When a SLEEP instruction is executed in subactive mode while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, the MSON bit in SYSCR2 is set to 1, the DTON bit in SYSCR2 is set to 1, and bit TMA3 in TMA is set to 1, a transition is made directly to active (medium-speed) mode. In this case, the time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transfer time) is given by equation (4) below:

Direct transfer time = { (Number of SLEEP instruction execution states) + (number of internal processing states) } × (t_{subcyc} before transition) + { (standby time set in STS2 to STS0) + (number of interrupt exception handling execution states) } × (t_{cyc} after transition) ..... (4)

Example: H8/3627 Series direct transfer time = $(2 + 1) \times 8t_w + (8192 + 14) \times 16t_{osc} = 24t_w + 13129t_{osc}$
(When $ø_w$/8 CPU operating clock and 8192-state standby time are selected)

Legend:
t_{osc}: OSC clock cycle time
t_w: Watch clock cycle time
t_{cyc}: System clock (ø) cycle time
t_{subcyc}: Subclock ($ø_{SUB}$) cycle time

Section 6 ROM

6.1 Overview

The H8/3627 has 60 kbytes of on-chip mask ROM, while the H8/3626 has 48 kbytes the H8/3625 has 40 kbytes the H8/3624S has 32 kbytes the H8/3623S has 24 kbytes and the H8/3622S has 16 kbytes. The H8/3627 also has 60 kbytes of on-chip PROM. The ROM is connected to the CPU by a 16-bit data bus, allowing high-speed 2-state access for both byte data and word data.

6.1.1 Block Diagram

Figure 6.1 shows a block diagram of the on-chip ROM.

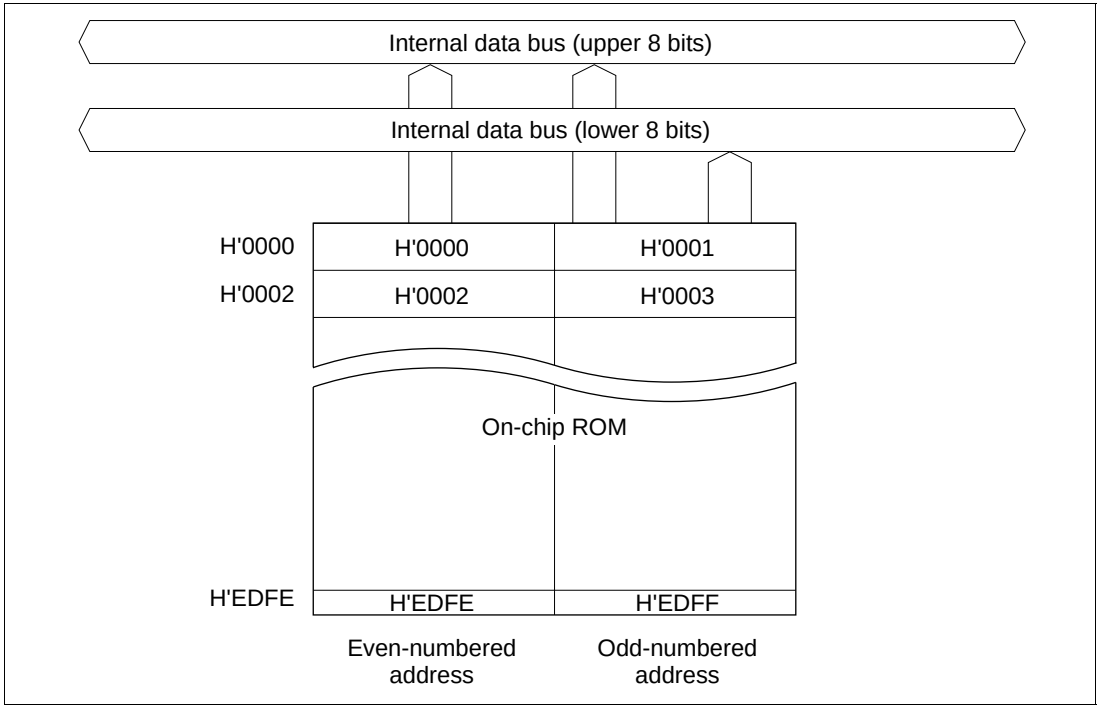


Figure 6.1 ROM Block Diagram (H8/3627)

6.2 PROM Mode

6.2.1 Selection of PROM Mode

If the on-chip ROM is PROM, setting the chip to PROM mode stops operation as a microcomputer and allows the on-chip PROM to be programmed in the same way as the HN27C101, except that page programming is not supported. Table 6.1 shows how to set PROM mode.

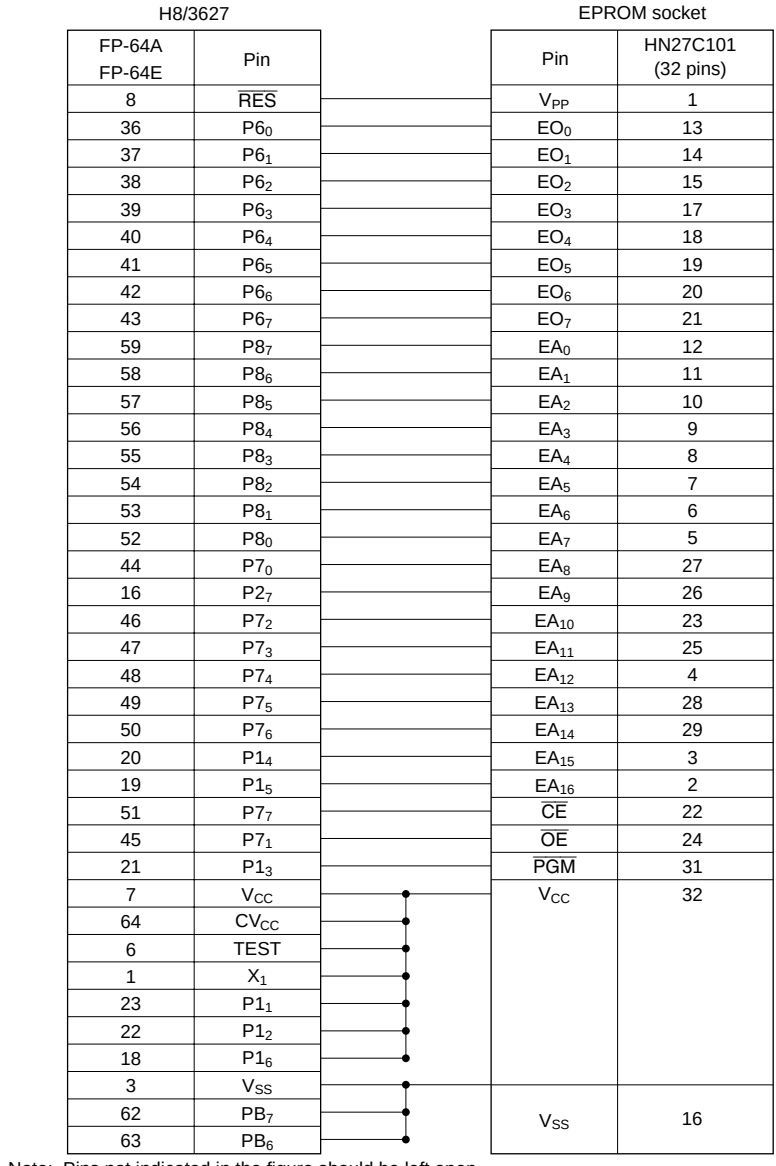
Table 6.1 Setting PROM Mode

Pin Name	Setting
TEST	High level
PB_7/AN_7	Low level
PB_6/AN_6	

6.2.2 Socket Adapter Pin Arrangement and Memory Map

A standard PROM programmer can be used to program the PROM. A socket adapter is required for conversion to 32 pins.

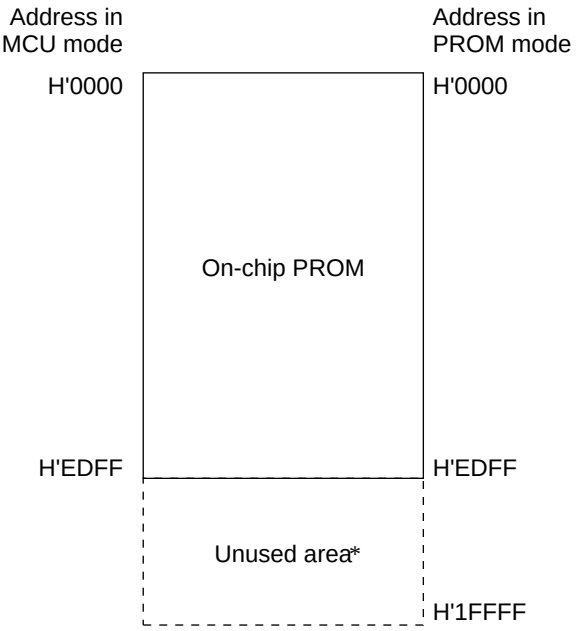
Figure 6.2 shows the pin-to-pin wiring of the socket adapter. Figure 6.3 shows a memory map.



H8/3627 FP-64A FP-64E	H8/3627 Pin	EPROM socket Pin	EPROM socket HN27C101 (32 pins)
8	$\overline{RES}$	V_{PP}	1
36	$P6_0$	EO_0	13
37	$P6_1$	EO_1	14
38	$P6_2$	EO_2	15
39	$P6_3$	EO_3	17
40	$P6_4$	EO_4	18
41	$P6_5$	EO_5	19
42	$P6_6$	EO_6	20
43	$P6_7$	EO_7	21
59	$P8_7$	EA_0	12
58	$P8_6$	EA_1	11
57	$P8_5$	EA_2	10
56	$P8_4$	EA_3	9
55	$P8_3$	EA_4	8
54	$P8_2$	EA_5	7
53	$P8_1$	EA_6	6
52	$P8_0$	EA_7	5
44	$P7_0$	EA_8	27
16	$P2_7$	EA_9	26
46	$P7_2$	EA_{10}	23
47	$P7_3$	EA_{11}	25
48	$P7_4$	EA_{12}	4
49	$P7_5$	EA_{13}	28
50	$P7_6$	EA_{14}	29
20	$P1_4$	EA_{15}	3
19	$P1_5$	EA_{16}	2
51	$P7_7$	$\overline{CE}$	22
45	$P7_1$	$\overline{OE}$	24
21	$P1_3$	$\overline{PGM}$	31
7	V_{CC}	V_{CC}	32
64	CV_{CC}		
6	TEST		
1	X_1		
23	$P1_1$		
22	$P1_2$		
18	$P1_6$		
3	V_{SS}	V_{SS}	16
62	PB_7		
63	PB_6		

Note: Pins not indicated in the figure should be left open.

Figure 6.2 Socket Adapter Pin Correspondence



Note: * Unpredictable data may be output if this area is read in PROM mode.
When programming with a PROM programmer, be sure to specify addresses from H'0000 to H'EDFF.
If H'EE00 and higher addresses are programmed by mistake, it may become impossible to program or verify the PROM. When programming, specify H'FF for this address area (H'EE00 to H'1FFFF).

Figure 6.3 Memory Map in PROM Mode

6.3 Programming

The program, verify, and other modes are selected as shown in table 6.2 in PROM mode.

Table 6.2 Mode Selection in PROM Mode

	Pin						
Mode	**$\overline{CE}$**	**$\overline{OE}$**	**$\overline{PGM}$**	**V_{PP}**	**V_{CC}**	**EO_7 to EO_0**	**EA_{16} to EA_0**
Write	L	H	L	V_{PP}	V_{CC}	Data input	Address input
Verify	L	L	H	V_{PP}	V_{CC}	Data output	Address input
Programming disabled	L	L	L	V_{PP}	V_{CC}	High impedance	Address input
	L	H	H				
	H	L	L				
	H	H	H				

Legend:
L: Low level
H: High level
V_{PP}: V_{PP} level
V_{CC}: V_{CC} level

The programming and verifying specifications in PROM mode are the same as the specifications of the standard HN27C101 EPROM. Page programming is not supported, however. The PROM programmer must not be set to page mode. PROM programmers that support only page programming cannot be used. When selecting a PROM programmer, make sure that it supports a byte-by-byte high-speed, high-reliability programming method. Be sure to set the address range to H'0000 to H'EDFF.

6.3.1 Programming and Verification

An efficient, high-speed, high-reliability programming procedure can be used to program and verify data. This procedure programs the chip quickly without subjecting it to voltage stress and without sacrificing data reliability. Data in unused address areas is H'FF. Figure 6.4 shows the basic high-speed, high-reliability programming flow chart.

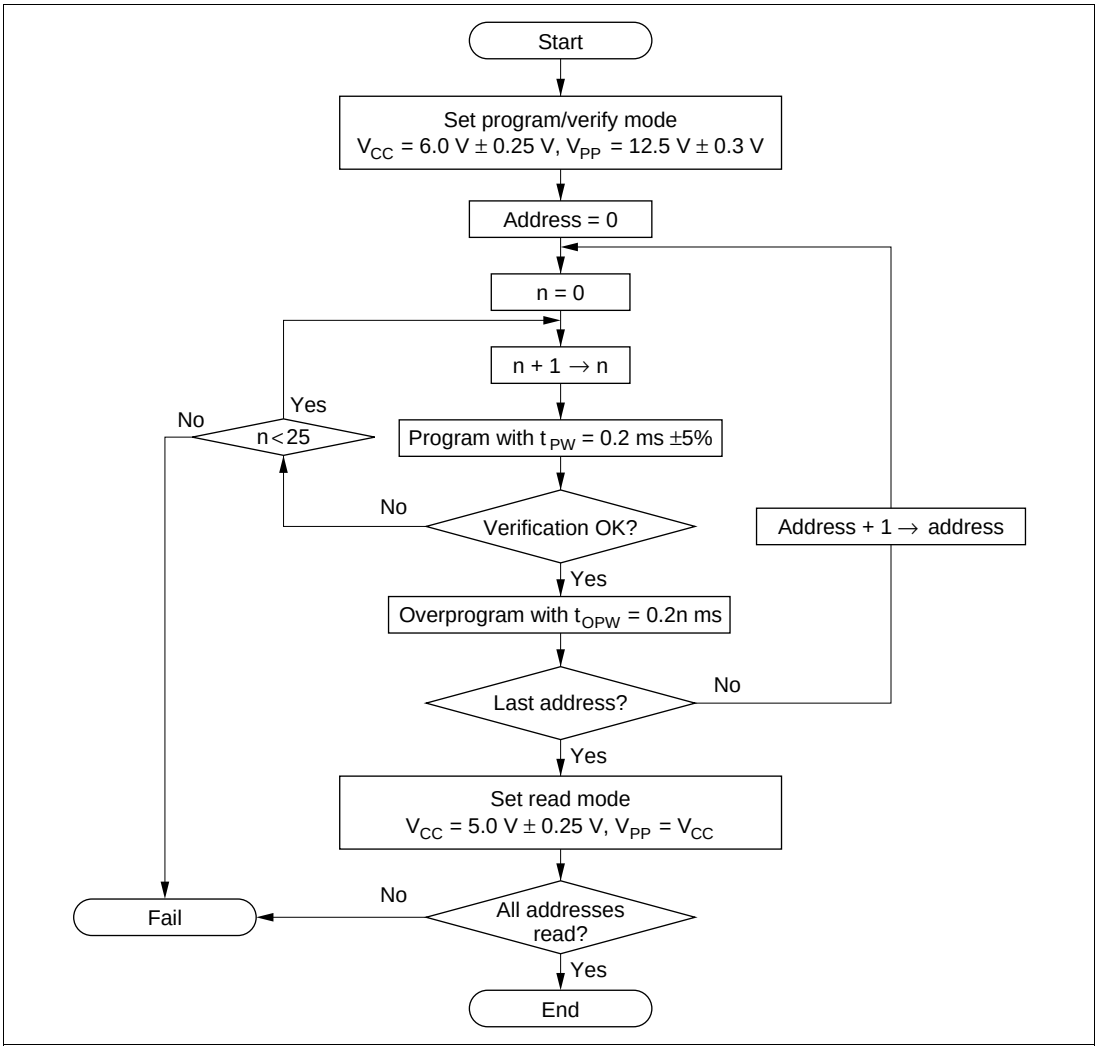


Figure 6.4 High-Speed, High-Reliability Programming Flow Chart

Table 6.3 and table 6.4 give the electrical characteristics in programming mode.

Table 6.3 DC Characteristics

(Conditions: V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item		Symbol	Min	Typ	Max	Unit	Test Condition
Input high-level voltage	EO_7 to EO_0, EA_{16} to EA_0 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	V_{IH}	2.4	—	V_{CC} + 0.3	V	
Input low-level voltage	EO_7 to EO_0, EA_{16} to EA_0 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	V_{IL}	–0.3	—	0.8	V	
Output high-level voltage	EO_7 to EO_0	V_{OH}	2.4	—	—	V	I_{OH} = –200 µA
Output low-level voltage	EO_7 to EO_0	V_{OL}	—	—	0.45	V	I_{OL} = 0.8 mA
Input leakage current	EO_7 to EO_0, EA_{16} to EA_0 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	$\lvert I_{LI} \rvert$	—	—	2	µA	V_{in} = 5.25 V/ 0.5 V
V_{CC} current		I_{CC}	—	—	40	mA	
V_{PP} current		I_{PP}	—	—	40	mA	

Table 6.4 AC Characteristics

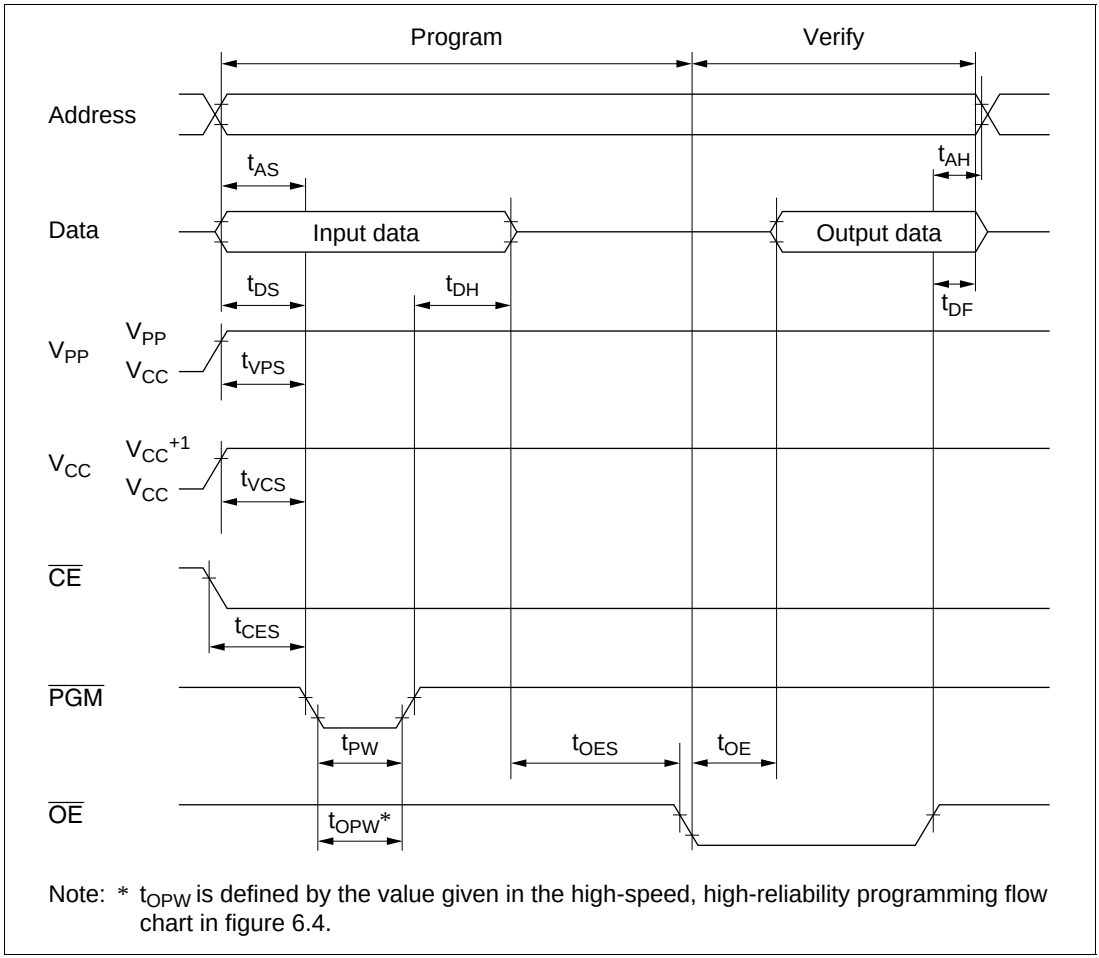
(Conditions: V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Address setup time	t_{AS}	2	—	—	µs	Figure 6.5*[1]
$\overline{OE}$ setup time	t_{OES}	2	—	—	µs	
Data setup time	t_{DS}	2	—	—	µs	
Address hold time	t_{AH}	0	—	—	µs	
Data hold time	t_{DH}	2	—	—	µs	
Data output disable time	t_{DF}*[2]	—	—	130	ns	
V_{PP} setup time	t_{VPS}	2	—	—	µs	
Programming pulse width	t_{PW}	0.19	0.20	0.21	ms	
$\overline{PGM}$ pulse width for over-programming	t_{OPW}*[3]	0.19	—	5.25	ms	
V_{CC} setup time	t_{VCS}	2	—	—	µs	
$\overline{CE}$ setup time	t_{CES}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	200	ns	

Notes: 1. Input pulse level: 0.45 V to 2.4 V
Input rise time/fall time ≤ 20 ns
Timing reference levels: Input: 0.8 V, 2.0 V
Output: 0.8 V, 2.0 V

2. t_{DF} is defined at the point at which the output is floating and the output level cannot be read.
3. t_{OPW} is defined by the value given in the hi-speed, hi-reliability programming flow chart in figure 6.4.

Figure 6.5 shows a program/verify timing diagram.



Note: * t_{OPW} is defined by the value given in the high-speed, high-reliability programming flow chart in figure 6.4.

Figure 6.5 PROM Program/Verify Timing

6.3.2 Programming Precautions

- Use the specified programming voltage and timing.

  The programming voltage in PROM mode (V_{PP}) is 12.5 V. Use of a higher voltage can permanently damage the chip. Be especially careful with respect to PROM programmer overshoot.

  Setting the PROM programmer to Hitachi specifications for the HN27C101 will result in correct V_{PP} of 12.5 V.

- Make sure the index marks on the PROM programmer socket, socket adapter, and chip are properly aligned. If they are not, the chip may be destroyed by excessive current flow. Before programming, be sure that the chip is properly mounted in the PROM programmer.

- Avoid touching the socket adapter or chip while programming, since this may cause contact faults and write errors.

- Select the programming mode carefully. The chip cannot be programmed in page programming mode.

- When programming with a PROM programmer, be sure to specify addresses from H'0000 to H'EDFF. If address H'EE00 and higher addresses are programmed by mistake, it may become impossible to program the PROM or verify the programmed data. When programming, assign H'FF data to the address area from H'EE00 to H'1FFFF.

6.4 Reliability of Programmed Data

A highly effective way of assuring data retention characteristics after programming is to screen the chips by baking them at a temperature of 150°C. This quickly eliminates PROM memory cells causing initial data retention failure.

Figure 6.6 shows a flowchart of this screening procedure.

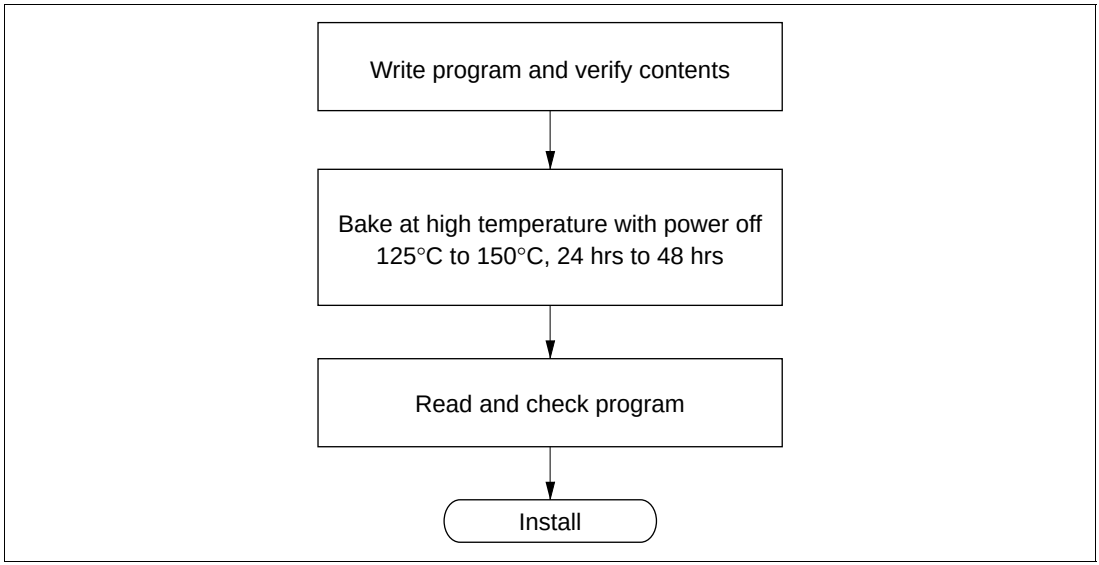


Figure 6.6 Recommended Screening Procedure

If write errors occur repeatedly while the same PROM programmer is being used, stop programming and check for problems in the PROM programmer and socket adapter, etc.

Please notify your Hitachi representative of any problems occurring during programming or in screening after high-temperature baking.

7.1 Overview

The H8/3627 Series has 1 kbyte or 2 kbytes of high-speed static RAM on-chip. The RAM is connected to the CPU by a 16-bit data bus, allowing high-speed 2-state access for both byte data and word data.

7.1.1 Block Diagram

Figure 7.1 shows a block diagram of the on-chip RAM.

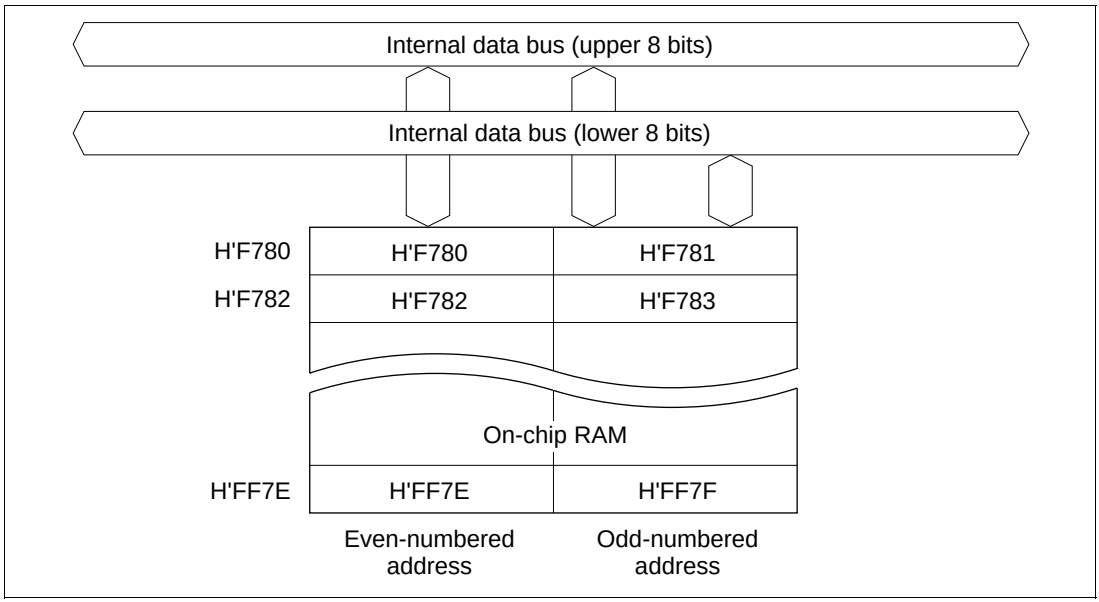


Figure 7.1 RAM Block Diagram (Example of 2 kbytes RAM)

Section 8 I/O Ports

8.1 Overview

The H8/3627 Series is provided with five 8-bit I/O ports, one 7-bit I/O port, one 3-bit I/O port, one 2-bit input-only port, and one 1-bit input-only port. Table 8.1 indicates the functions of each port.

Each port has of a port control register (PCR) that controls input and output, and a port data register (PDR) for storing output data. Input or output can be controlled by individual bits.
See 2.9.2, Notes on Bit Manipulation, for information on executing bit-manipulation instructions to write data in PCR or PDR.

Block diagrams of each port are given in Appendix C, I/O Port Block Diagrams.

Table 8.1 Port Functions

Port	Description	Pins	Other Functions	Function Switching Register
Port 1	• 8-bit I/O port • Input pull-up MOS option	$P1_7$ to $P1_5$/ $\overline{IRQ}_3$ to $\overline{IRQ}_1$/ TMIF	External interrupts 3 to 1 Timer event input TMIF	PMR1 TCRF
		$P1_4$	None	
		$P1_3$/TMIG	Timer G input capture	PMR1
		$P1_2$, $P1_1$/TMOFH, TMOFL	Timer F output compare	PMR1
		$P1_0$/TMOW	Timer A clock output	PMR1
Port 2	• 1-bit input-only port	$P2_7$	External interrupt 0	PMR2
	• 7-bit I/O port • Input pull-up MOS option	$P2_6$/TXD $P2_5$/RXD $P2_4$/SCK_3	SCI3 data output (TXD), data input (RXD), clock input/output (SCK_3)	SCR3 SMR3 PMR6
		$P2_3$/SO_1 $P2_2$/SI_1 $P2_1$/SCK_1	SCI1 data output (SO_1), data input (SI_1), clock input/output (SCK_1)	PMR2
		$P2_0$/$\overline{IRQ}_4$/ $\overline{ADTRG}$	External interrupt 4 and A/D converter external trigger	PMR2
Port 5	• 8-bit I/O port • Input pull-up MOS option	$P5_7$ to $P5_0$/ $\overline{WKP}_7$ to $\overline{WKP}_0$	Wakeup input ($\overline{WKP}_7$ to $\overline{WKP}_0$)	PMR5
Port 6	• 8-bit I/O port • Input pull-up MOS option	$P6_7$ to $P6_0$	None	
Port 7	• 8-bit I/O port	$P7_7$ to $P7_0$	None	
Port 8	• 8-bit I/O port	$P8_7$ to $P8_0$	None	
Port A	• 3-bit I/O port	PA_3 to PA_1	None	
Port B	• 2-bit input port	PB_7, PB_6/AN_7, AN_6	A/D converter analog input	AMR

8.2 Port 1

8.2.1 Overview

Port 1 is an 8-bit I/O port. Figure 8.1 shows its pin configuration.

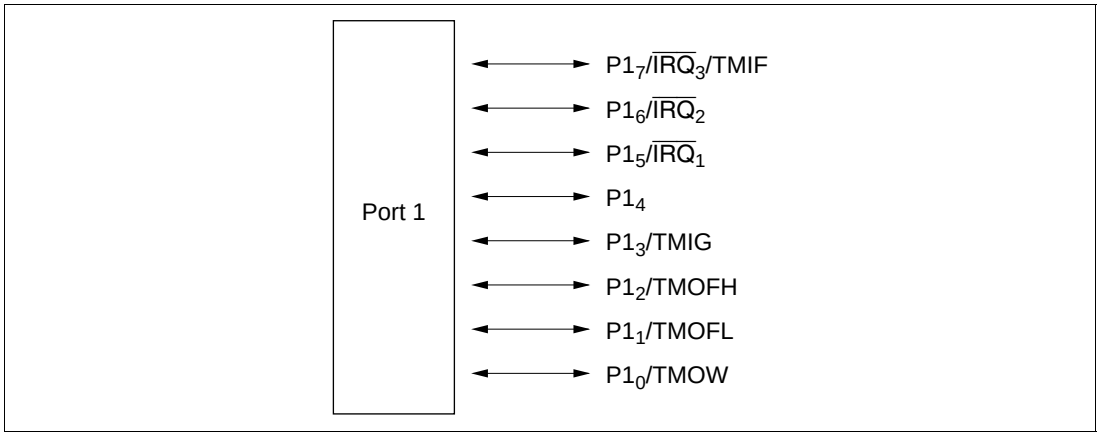


Figure 8.1 Port 1 Pin Configuration

8.2.2 Register Configuration and Description

Table 8.2 shows the port 1 register configuration.

Table 8.2 Port 1 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 1	PDR1	R/W	H'00	H'FFD4
Port control register 1	PCR1	W	H'00	H'FFE4
Port pull-up control register 1	PUCR1	R/W	H'00	H'FF9C
Port mode register 1	PMR1	R/W	H'00	H'FF98

Port Data Register 1 (PDR1)

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

PDR1 is an 8-bit register that stores data for pins $P1_7$ to $P1_0$ of port 1. If port 1 is read while PCR1 bits are set to 1, the values stored in PDR1 are directly read, regardless of the actual pin states. If port 1 is read while PCR1 bits are cleared to 0, the pin states are read.

Upon reset, PDR1 is initialized to H'00.

Port Control Register 1 (PCR1)

Bit	7	6	5	4	3	2	1	0
	$PCR1_7$	$PCR1_6$	$PCR1_5$	$PCR1_4$	$PCR1_3$	$PCR1_2$	$PCR1_1$	$PCR1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR1 is an 8-bit register for controlling whether each of the port 1 pins $P1_7$ to $P1_0$ functions as an input pin or output pin. Setting a PCR1 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. The settings in PCR1 and in PDR1 are valid only when the corresponding pin is designated in PMR1 as a general I/O pin.

Upon reset, PCR1 is initialized to H'00.

PCR1 is a write-only register. All bits are read as 1.

Port Pull-Up Control Register 1 (PUCR1)

Bit	7	6	5	4	3	2	1	0
	$PUCR1_7$	$PUCR1_6$	$PUCR1_5$	$PUCR1_4$	$PUCR1_3$	$PUCR1_2$	$PUCR1_1$	$PUCR1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR1 controls whether the MOS pull-up of each port 1 pins $P1_7$ to $P1_0$ is on or off. When a PCR1 bit is cleared to 0, setting the corresponding PUCR1 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR1 is initialized to H'00.

Port Mode Register 1 (PMR1)

Bit	7	6	5	4	3	2	1	0
	IRQ3	IRQ2	IRQ1	—	TMIG	TMOFH	TMOFL	TMOW
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

PMR1 is an 8-bit read/write register, controlling the selection of pin functions for port 1 pins.

Upon reset, PMR1 is initialized to H'00.

Bit 7—$P1_7$/$\overline{IRQ_3}$/TMIF Pin Function Switch (IRQ3): This bit selects whether pin $P1_7$/$\overline{IRQ_3}$/TMIF is used as $P1_7$ or as $\overline{IRQ_3}$/TMIF.

Bit 7: IRQ3	Description	
0	Functions as $P1_7$ I/O pin	(initial value)
1	Functions as $\overline{IRQ_3}$/TMIF input pin	

Note: Rising or falling edge sensing can be designated for IRQ_3/TMIF.
For information about TMIF pin settings, see 9.3.2 (3), Timer Control Register F (TCRF).

Bit 6—$P1_6$/$\overline{IRQ_2}$ Pin Function Switch (IRQ2): This bit selects whether pin $P1_6$/$\overline{IRQ_2}$ is used as $P1_6$ or as $\overline{IRQ_2}$.

Bit 6: IRQ2	Description	
0	Functions as $P1_6$ I/O pin	(initial value)
1	Functions as $\overline{IRQ_2}$ input pin	

Note: Rising or falling edge sensing can be designated for $\overline{IRQ_2}$.

Bit 5—$P1_5$/$\overline{IRQ_1}$ Pin Function Switch (IRQ1): This bit selects whether pin $P1_5$/$\overline{IRQ_1}$ is used as $P1_5$ or as $\overline{IRQ_1}$.

Bit 5: IRQ1	Description	
0	Functions as $P1_5$ I/O pin	(initial value)
1	Functions as $\overline{IRQ_1}$ input pin	

Note: Rising or falling edge sensing can be designated for IRQ_1.

Bit 4—Reserved Bit: Bit 4 is reserved: it is a always read as 0, and should be used cleared to 0.

Bit 3—$P1_3$/TMIG Pin Function Switch (TMIG): This bit selects whether pin $P1_3$/TMIG is used as $P1_3$ or as TMIG.

Bit 3: TMIG	Description	
0	Functions as $P1_3$ I/O pin	(initial value)
1	Functions as TMIG input pin	

Bit 2—$P1_2$/TMOFH Pin Function Switch (TMOFH): This bit selects whether pin $P1_2$/TMOFH is used as $P1_2$ or as TMOFH.

Bit 2: TMOFH	Description	
0	Functions as $P1_2$ I/O pin	(initial value)
1	Functions as TMOFH output pin	

Bit 1—$P1_1$/TMOFL Pin Function Switch (TMOFL): This bit selects whether pin $P1_1$/TMOFL is used as $P1_1$ or as TMOFL.

Bit 1: TMOFL	Description	
0	Functions as $P1_1$ I/O pin	(initial value)
1	Functions as TMOFL output pin	

Bit 0—$P1_0$/TMOW Pin Function Switch (TMOW): This bit selects whether pin $P1_0$/TMOW is used as $P1_0$ or as TMOW.

Bit 0: TMOW	Description	
0	Functions as $P1_0$ I/O pin	(initial value)
1	Functions as TMOW output pin	

8.2.3 Pin Functions

Table 8.3 shows the port 1 pin functions.

Table 8.3 Port 1 Pin Functions

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P1_7$/$\overline{IRQ}_3$/TMIF</td><td>The pin function depends on bit IRQ3 in PMR1, bits CKSL2 to CKSL0 in TCRF, and bit $PCR1_7$ in PCR1.
<table>
<tr><td>IRQ3</td><td colspan="2">0</td><td colspan="2">1</td></tr>
<tr><td>$PCR1_7$</td><td>0</td><td>1</td><td colspan="2">*</td></tr>
<tr><td>CKSL2 to CKSL0</td><td colspan="2">*</td><td>Not 0**</td><td>0**</td></tr>
<tr><td>Pin function</td><td>$P1_7$ input pin</td><td>$P1_7$ output pin</td><td>$\overline{IRQ}_3$ input pin</td><td>$\overline{IRQ}_3$/TMIF input pin</td></tr>
</table>
Note: When using this pin for TMIF input, clear bit IEN3 to 0 in IENR1 to disable IRQ_3 interrupts.</td></tr>
<tr><td>$P1_6$/$\overline{IRQ}_2$</td><td>The pin function depends on bit IRQ2 in PMR1, bit $PCR1_6$ in PCR1.
<table>
<tr><td>IRQ2</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR1_6$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P1_6$ input pin</td><td>$P1_6$ output pin</td><td>$\overline{IRQ}_2$ input pin</td></tr>
</table></td></tr>
<tr><td>$P1_5$/$\overline{IRQ}_1$</td><td>The pin function depends on bit IRQ1 in PMR1 and bit $PCR1_5$ in PCR1.
<table>
<tr><td>IRQ1</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR1_5$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P1_5$ input pin</td><td>$P1_5$ output pin</td><td>$\overline{IRQ}_1$ input pin</td></tr>
</table></td></tr>
<tr><td>$P1_4$</td><td>The pin function depends on bit $PCR1_4$ in PCR1.
<table>
<tr><td>$PCR1_4$</td><td>0</td><td>1</td></tr>
<tr><td>Pin function</td><td>$P1_4$ input pin</td><td>$P1_4$ output pin</td></tr>
</table></td></tr>
<tr><td>$P1_3$/TMIG</td><td>The pin function depends on bit TMIG in PMR1 and bit $PCR1_3$ in PCR1.
<table>
<tr><td>TMIG</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR1_3$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P1_3$ input pin</td><td>$P1_3$ output pin</td><td>TMIG input pin</td></tr>
</table></td></tr>
</table>

Table 8.3 Port 1 Pin Functions (cont)

Pin	Pin Functions and Selection Method		
$P1_2$/TMOFH	The pin function depends on bit TMOFH in PMR1 and bit $PCR1_2$ in PCR1.		
	TMOFH	0	1
	$PCR1_2$	0	1	*
	Pin function	$P1_2$ input pin	$P1_2$ output pin	TMOFH output pin
$P1_1$/TMOFL	The pin function depends on bit TMOFL in PMR1 and bit $PCR1_1$ in PCR1.		
	TMOFL	0	1
	$PCR1_1$	0	1	*
	Pin function	$P1_1$ input pin	$P1_1$ output pin	TMOFL output pin
$P1_0$/TMOW	The pin function depends on bit TMOW in PMR1 and bit $PCR1_0$ in PCR1.		
	TMOW	0	1
	$PCR1_0$	0	1	*
	Pin function	$P1_0$ input pin	$P1_0$ output pin	TMOW output pin

Note: * Don't care

8.2.4 Pin States

Table 8.4 shows the port 1 pin states in each operating mode.

Table 8.4 Port 1 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P1_7/\overline{IRQ_3}$/TMIF $P1_6/\overline{IRQ_2}$ $P1_5/\overline{IRQ_1}$ $P1_4$ $P1_3$/TMIG $P1_2$/TMOFH $P1_1$/TMOFL $P1_0$/TMOW	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

Note: * A high-level signal is output when the MOS pull-up is in the on state.

8.2.5 MOS Input Pull-Up

Port 1 has a built-in MOS input pull-up function that can be controlled by software. When a PCR1 bit is cleared to 0, setting the corresponding PUCR1 bit to 1 turns on the MOS pull-up for that pin. The MOS input pull-up function is in the off state after a reset.

$PCR1_n$	0		1
$PUCR1_n$	0	1	*
MOS input pull-up	Off	On	Off

Note: * Don't care
(n = 7 to 0)

8.3 Port 2

8.3.1 Overview

Port 2 is an 7-bit I/O port and 1-bit input port. Figure 8.2 shows its pin configuration.

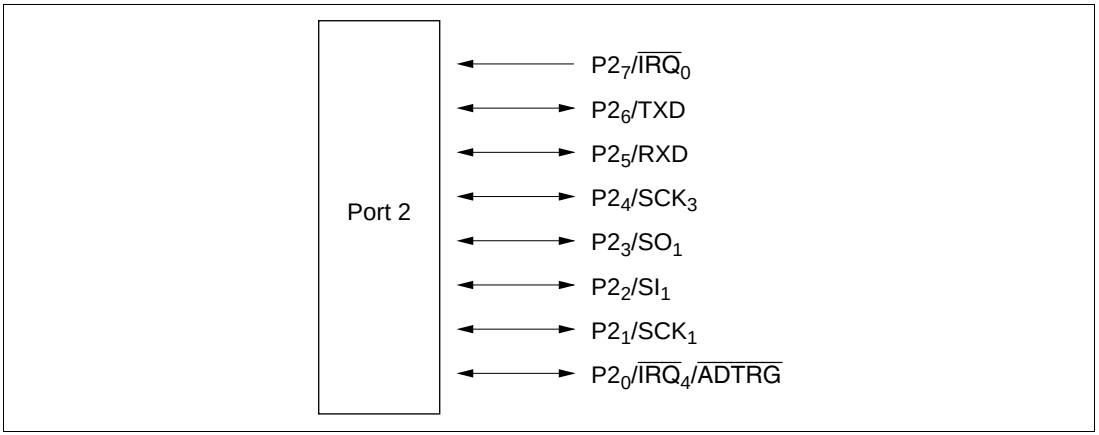


Figure 8.2 Port 2 Pin Configuration

8.3.2 Register Configuration and Description

Table 8.5 shows the port 2 register configuration.

Table 8.5 Port 2 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 2	PDR2	R/W	H'00	H'FFD5
Port control register 2	PCR2	W	H'00	H'FFE5
Port mode register 2	PMR2	R/W	H'40	H'FF99
Port mode register 6	PMR6	R/W	H'F8	H'FF9A
Port pull-up control register 2	PUCR2	R/W	H'00	H'FF9D

Port Data Register 2 (PDR2)

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

PDR2 is an 8-bit register that stores data for pins $P2_7$ to $P2_0$ of port 2. If port 2 is read while PCR2 bits are set to 1, the values stored in PDR2 are directly read, regardless of the actual pin states. If port 2 is read while PCR2 bits are cleared to 0, the pin states are read.

Upon reset, PDR2 is initialized to H'00.

Port Control Register 2 (PCR2)

Bit	7	6	5	4	3	2	1	0
	$PCR2_7$	$PCR2_6$	$PCR2_5$	$PCR2_4$	$PCR2_3$	$PCR2_2$	$PCR2_1$	$PCR2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR2 is an 8-bit register for controlling whether each of the port 2 pins $P2_7$ to $P2_0$ functions as an input pin or output pin. Setting a PCR2 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. The settings in PCR2 and in PDR2 are valid only when the corresponding pin is designated in PMR2 as a general I/O pin.

Upon reset, PCR2 is initialized to H'00.

PCR2 is a write-only register, which always reads all 1s.

Note: As $P2_7$ is an input-only pin, it becomes a high-impedance output when $PCR2_7$ is set to 1.

Port Mode Register 2 (PMR2)

Bit	7	6	5	4	3	2	1	0
	IRQ0	—	POF1	NCS	SO1	SI1	SCK1	IRQ4
Initial value	0	1	0	0	0	0	0	0
Read/Write	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W

PMR2 is an 8-bit read/write register for controlling the selection of pin functions for pins $P2_0$ to $P2_3$ and $P2_7$, controlling the PMOS on/off option for pin $P2_3/SO_1$, and controlling the TMIG input noise canceller.

Upon reset, PMR2 is initialized to H'40.

Bits 7—$P2_7/\overline{IRQ_0}$ Pin Function Switch (IRQ0): This bit selects whether pin $P2_7/\overline{IRQ_0}$ is used as $P2_7$ or as $\overline{IRQ_0}$.

Bit 7: IRQ0	Description	
0	Functions as $P2_7$ input pin	(initial value)
1	Functions as $\overline{IRQ_0}$ input pin	

Bit 6—Reserved Bit: Bit 6 is reserved: it is always read as 1, and cannot be modified.

Bit 5—$P2_3/SO_1$ pin PMOS control (POF1): This bit turns on and off the PMOS transistor in the $P2_3/SO_1$ pin output buffer.

Bit 5: POF1	Description	
0	CMOS output	(initial value)
1	NMOS open-drain output	

Bit 4—TMIG noise canceller select (NCS): This bit controls the noise canceller circuit for input capture at pin TMIG.

Bit 4: NCS	Description	
0	Noise canceller function not selected	(initial value)
1	Noise canceller function selected	

Bit 3—$P2_3/SO_1$ Pin Function Switch (SO1): This bit selects whether pin $P2_3/SO_1$ is used as $P2_3$ or as SO_1.

Bit 3: SO1	Description	
0	Functions as $P2_3$ I/O pin	(initial value)
1	Functions as SO_1 output pin	

Bit 2—$P2_2/SI_1$ Pin Function Switch (SI1): This bit selects whether pin $P2_2/SI_1$ is used as $P2_2$ or as SI_1.

Bit 2: SI1	Description	
0	Functions as $P2_2$ I/O pin	(initial value)
1	Functions as SI_1 input pin	

Bit 1—$P2_1/SCK_1$ Pin Function Switch (SCK1): This bit selects whether pin $P2_1/SCK_1$ is used as $P2_1$ or as SCK_1.

Bit 1: SCK1	Description	
0	Functions as $P2_1$ I/O pin	(initial value)
1	Functions as SCK_1 I/O pin	

Bit 0—$P2_0/\overline{IRQ_4}/\overline{ADTRG}$ Pin Function Switch (IRQ4): This bit selects whether pin $P2_0/\overline{IRQ_4}/\overline{ADTRG}$ is used as $P2_0$ or as $\overline{IRQ_4}/\overline{ADTRG}$.

Bit 0: IRQ4	Description	
0	Functions as $P2_0$ I/O pin	(initial value)
1	Functions as $\overline{IRQ_4}/\overline{ADTRG}$ input pin	

Note: For information about $\overline{ADTRG}$ pin settings, see 12.3.2, Start of A/D Conversion by External Trigger Input.

Port Mode Register 6 (PMR6)

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

Bits 7, 6, and 3: Reserved Bits: Bits 7, 6, and 3 are reserved bits. They are always read as 1 and cannot be modified.

Bit 5, 4—Reserved Bit: Bit 5, 4 is reserved: it should be used set to 1.

Bit 2—$P2_6$/TXD Pin Function Switch (TXD): This bits selects whether the $P2_6$/TXD pin is used as $P2_6$ or as TXD.

Bit 2: TXD	Description	
0	Functions as $P2_6$ I/O pin	(initial value)
1	Functions as TXD output pin	

Bits 1 and 0—Reserved Bits: Bits 1 and 0 are reserved: they should be used cleared to 0.

Port Pull-Up Control Register 2 (PUCR2)

Bit	7	6	5	4	3	2	1	0
	$PUCR2_7$	$PUCR2_6$	$PUCR2_5$	$PUCR2_4$	$PUCR2_3$	$PUCR2_2$	$PUCR2_1$	$PUCR2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR2 controls whether the MOS pull-up of each port 2 pin is on or off. When a PCR2 bit is cleared to 0, setting the corresponding PUCR2 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR2 is initialized to H'00.

Note: As $P2_7$ is an input-only pin, the MOS pull-up is turned off regardless of whether $PUCR2_7$ is set to 1 or cleared to 0.

8.3.3 Pin Functions

Table 8.6 shows the port 2 pin functions.

Table 8.6 Port 2 Pin Functions

Pin	Pin Functions and Selection Method
$P2_7/\overline{IRQ}_0$	The pin function depends on bit IRQ0 in PMR2 and bit $PCR2_7$ in PCR2.

IRQ0	0		1
$PCR2_7$	0	1	*
Pin function	$P2_7$ input pin	High-impedance	$\overline{IRQ}_0$ input pin

$P2_6$/TXD — The pin function depends on bit TXD in PMR6 and bit $PCR2_6$ in PCR2.

TXD	0		1
$PCR2_6$	0	1	*
Pin function	$P2_6$ input pin	$P2_6$ output pin	TXD output pin

$P2_5$/RXD — The pin function depends on bit RE in SCR of SCI3 and bit $PCR2_5$ in PCR2.

RE	0		1
$PCR2_5$	0	1	*
Pin function	$P2_5$ input pin	$P2_5$ output pin	RXD input pin

$P2_4/SCK_3$ — The pin function depends on bits CKE1 and CKE0 in SCR of SCI3, bit COM in SMR of SCI3, and bit $PCR2_4$ in PCR2.

CKE1	0				1
CKE0	0		0	1	*
COM	0		1	*	*
$PCR2_4$	0	1	*		*
Pin function	$P2_4$ input pin	$P2_4$ output pin	SCK_3 output pin		SCK_3 input pin

$P2_3/SO_1$ — The pin function depends on bit SO1 in PMR2 and bit $PCR2_3$ in PCR2.

SO1	0		1
$PCR2_3$	0	1	*
Pin function	$P2_3$ input pin	$P2_3$ output pin	SO_1 output pin

Table 8.6 Port 2 Pin Functions (cont)

Pin	Pin Functions and Selection Method
$P2_2/SI_1$	The pin function depends on bit SI1 in PMR2 and bit $PCR2_2$ in PCR2.

SI1	0		1
$PCR2_2$	0	1	*
Pin function	$P2_2$ input pin	$P2_2$ output pin	SI_1 input pin

$P2_1/SCK_1$	The pin function depends on bit SCK1 in PMR2, bit CKS3 in SCR1, and bit $PCR2_1$ in PCR2.

SCK1	0		1	
CKS_3	*		0	1
$PCR2_1$	0	1	*	*
Pin function	$P2_1$ input pin	$P2_1$ output pin	SCK_1 output pin	SCK_1 input pin

$P2_0/\overline{IRQ_4}/\overline{ADTRG}$	The pin function depends on bit IRQ4 in PMR2, bit TRGE in AMR, and bit $PCR2_0$ in PCR2.

IRQ4	0		1	
$PCR2_0$	0	1	*	
TRGE	*		0	1
Pin function	$P2_0$ input pin	$P2_0$ output pin	$\overline{IRQ_4}$ input pin	$\overline{IRQ_4}/\overline{ADTRG}$ input pin

Note: When using this pin for $\overline{ADTRG}$ input, clear bit IEN4 to 0 in IENR1 to disable IRQ_4 interrupts.

Note: * Don't care

8.3.4 Pin States

Table 8.7 shows the port 2 pin states in each operating mode.

Table 8.7 Port 2 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P2_7/\overline{IRQ_0}$	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance
$P2_6$/TXD $P2_5$/RXD $P2_4/SCK_3$ $P2_3/SO_1$ $P2_2/SI_1$ $P2_1/SCK_1$ $P2_0/\overline{IRQ_4}$/ $\overline{ADTRG}$	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

Note: * High level output if the MOS pull-up is on.

8.3.5 MOS Input Pull-Up

Port 2 has a built-in MOS input pull-up function that can be controlled by software. When a PCR2 bit is cleared to 0, setting the corresponding PUCR2 bit to 1 turns on the MOS pull-up for that pin. The MOS input pull-up function is in the off state after a reset.

$PCR2_n$	0		1
$PUCR2_n$	0	1	*
MOS input pull-up	Off	On	Off

Note: * Don't care
(n = 6 to 0)

Note: As $P2_7$ is an input-only pin, the MOS pull-up is turned off regardless of whether $PUCR2_7$ is set to 1 or cleared to 0.

8.4 Port 5

8.4.1 Overview

Port 5 is an 8-bit I/O port, configured as shown in figure 8.3.

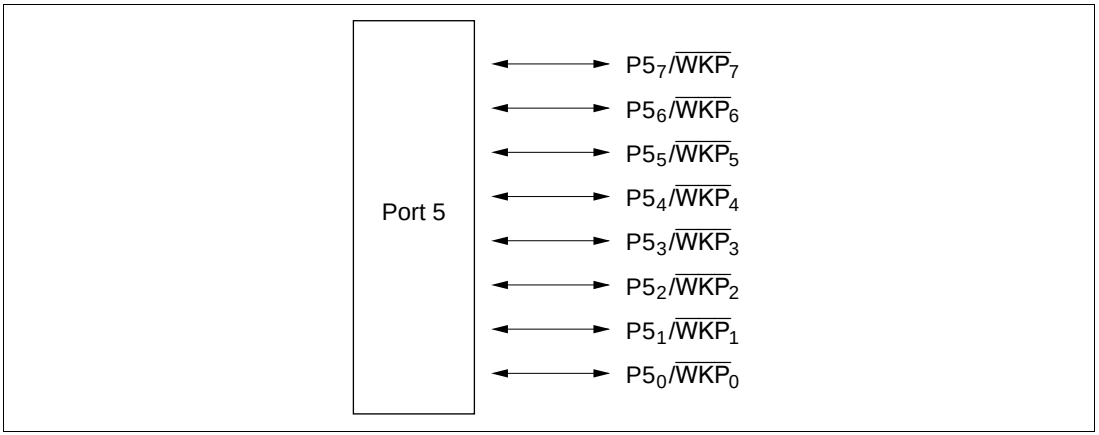


Figure 8.3 Port 5 Pin Configuration

8.4.2 Register Configuration and Description

Table 8.8 shows the port 5 register configuration.

Table 8.8 Port 5 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 5	PDR5	R/W	H'00	H'FFD8
Port control register 5	PCR5	W	H'00	H'FFE8
Port pull-up control register 5	PUCR5	R/W	H'00	H'FF9E
Port mode register 5	PMR5	R/W	H'00	H'FF9B

Port Data Register 5 (PDR5)

Bit	7	6	5	4	3	2	1	0
	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PDR5 is an 8-bit register that stores data for port 5 pins $P5_7$ to $P5_0$. If port 5 is read while PCR5 bits are set to 1, the values stored in PDR5 are directly read, regardless of the actual pin states. If port 5 is read while PCR5 bits are cleared to 0, the pin states are read.

Upon reset, PDR5 is initialized to H'00.

Port Control Register 5 (PCR5)

Bit	7	6	5	4	3	2	1	0
	$PCR5_7$	$PCR5_6$	$PCR5_5$	$PCR5_4$	$PCR5_3$	$PCR5_2$	$PCR5_1$	$PCR5_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR5 is an 8-bit register for controlling whether each of the port 5 pins $P5_7$ to $P5_0$ functions as an input pin or output pin. Setting a PCR5 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. The settings in PCR5 and in PDR5 are valid only when the corresponding pin is designated as a general I/O pin in PMR5.

Upon reset, PCR5 is initialized to H'00.

PCR5 is a write-only register. All bits are read as 1.

Port Pull-Up Control Register 5 (PUCR5)

Bit	7	6	5	4	3	2	1	0
	$PUCR5_7$	$PUCR5_6$	$PUCR5_5$	$PUCR5_4$	$PUCR5_3$	$PUCR5_2$	$PUCR5_1$	$PUCR5_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR5 controls whether the MOS pull-up of each port 5 pin is on or off. When a PCR5 bit is cleared to 0, setting the corresponding PUCR5 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR5 is initialized to H'00.

Port Mode Register 5 (PMR5)

Bit	7	6	5	4	3	2	1	0
	WKP_7	WKP_6	WKP_5	WKP_4	WKP_3	WKP_2	WKP_1	WKP_0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PMR5 is an 8-bit read/write register, controlling the selection of pin functions for port 5 pins.

Upon reset, PMR5 is initialized to H'00.

Bit n: $P5_n/\overline{WKP}_n$ Pin Function Switch (WKPn): This bit selects whether it is used as $P5_n$ or as $\overline{WKP}_n$.

Bit n: WKPn	Description	
0	Functions as $P5_n$ I/O pin	(initial value)
1	Functions as $\overline{WKP}_n$ input pin	

(n = 7 to 0)

8.4.3 Pin Functions

Table 8.9 shows the port 5 pin functions.

Table 8.9 Port 5 Pin Functions

Pin	Pin Functions and Selection Method
$P5_7/\overline{WKP}_7$ to $P5_0/\overline{WKP}_0$	The pin function depends on bit WKP_n in PMR5 and bit $PCR5_n$ in PCR5. (n = 7 to 0)

WKP_n	0		1
$PCR5_n$	0	1	*
Pin function	$P5_n$ input pin	$P5_n$ output pin	$\overline{WKP}_n$ input pin

Note: * Don't care

8.4.4 Pin States

Table 8.10 shows the port 5 pin states in each operating mode.

Table 8.10 Port 5 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P5_7/\overline{WKP}_7$ to $P5_0/\overline{WKP}_0$	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

Note: * A high-level signal is output when the MOS pull-up is in the on state.

8.4.5 MOS Input Pull-Up

Port 5 has a built-in MOS input pull-up function that can be controlled by software. When a PCR5 bit is cleared to 0, setting the corresponding PUCR5 bit to 1 turns on the MOS pull-up for that pin. The MOS pull-up function is in the off state after a reset.

$PCR5_n$	0		1
$PUCR5_n$	0	1	*
MOS input pull-up	Off	On	Off

Note: * Don't care
(n = 7 to 0)

8.5 Port 6

8.5.1 Overview

Port 6 is an 8-bit I/O port, configured as shown in figure 8.4.

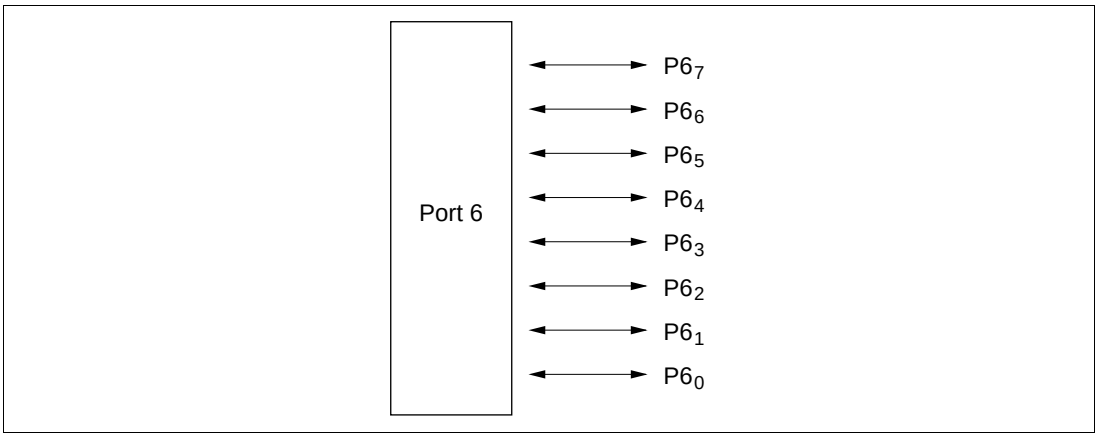


Figure 8.4 Port 6 Pin Configuration

8.5.2 Register Configuration and Description

Table 8.11 shows the port 6 register configuration.

Table 8.11 Port 6 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 6	PDR6	R/W	H'00	H'FFD9
Port control register 6	PCR6	W	H'00	H'FFE9
Port pull-up control register 6	PUCR6	R/W	H'00	H'FF9F

Port Data Register 6 (PDR6)

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

PDR6 is an 8-bit register that stores data for port 6 pins $P6_7$ to $P6_0$. If port 6 is read while PCR6 bits are set to 1, the values stored in PDR6 are directly read, regardless of the actual pin states. If port 6 is read while PCR6 bits are cleared to 0, the pin states are read.

Upon reset, PDR6 is initialized to H'00.

Port Control Register 6 (PCR6)

Bit	7	6	5	4	3	2	1	0
	$PCR6_7$	$PCR6_6$	$PCR6_5$	$PCR6_4$	$PCR6_3$	$PCR6_2$	$PCR6_1$	$PCR6_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR6 is an 8-bit register for controlling whether each of the port 6 pins $P6_7$ to $P6_0$ functions as an input pin or output pin. Setting a PCR6 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCR6 is initialized to H'00.

PCR6 is a write-only register. All bits are read as 1.

Port Pull-Up Control Register 6 (PUCR6)

Bit	7	6	5	4	3	2	1	0
	$PUCR6_7$	$PUCR6_6$	$PUCR6_5$	$PUCR6_4$	$PUCR6_3$	$PUCR6_2$	$PUCR6_1$	$PUCR6_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR6 controls whether the MOS pull-up of each port 6 pin is on or off. When a PCR6 bit is cleared to 0, setting the corresponding PUCR6 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR6 is initialized to H'00.

8.5.3 Pin Functions

Table 8.12 shows the port 6 pin functions.

Table 8.12 Port 6 Pin Functions

Pin	Pin Functions and Selection Method
$P6_7$ to $P6_0$	The pin function depends on bit $PCR6_n$ in PCR6. (n = 7 to 0)

$PCR6_n$	0	1
Pin function	$P6_n$ input pin	$P6_n$ output pin

8.5.4 Pin States

Table 8.13 shows the port 6 pin states in each operating mode.

Table 8.13 Port 6 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P6_7$ to $P6_0$	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

Note: * A high-level signal is output when the MOS pull-up is in the on state.

8.5.5 MOS Input Pull-Up

Port 6 has a built-in MOS input pull-up function that can be controlled by software. When a PCR6 bit is cleared to 0, setting the corresponding PUCR6 bit to 1 turns on the MOS pull-up for that pin. The MOS pull-up function is in the off state after a reset.

$PCR6_n$	0		1
$PUC6_n$	0	1	*
MOS input pull-up	Off	On	Off

Note: * Don't care
(n = 7 to 0)

8.6 Port 7

8.6.1 Overview

Port 7 is an 8-bit I/O port, configured as shown in figure 8.5.

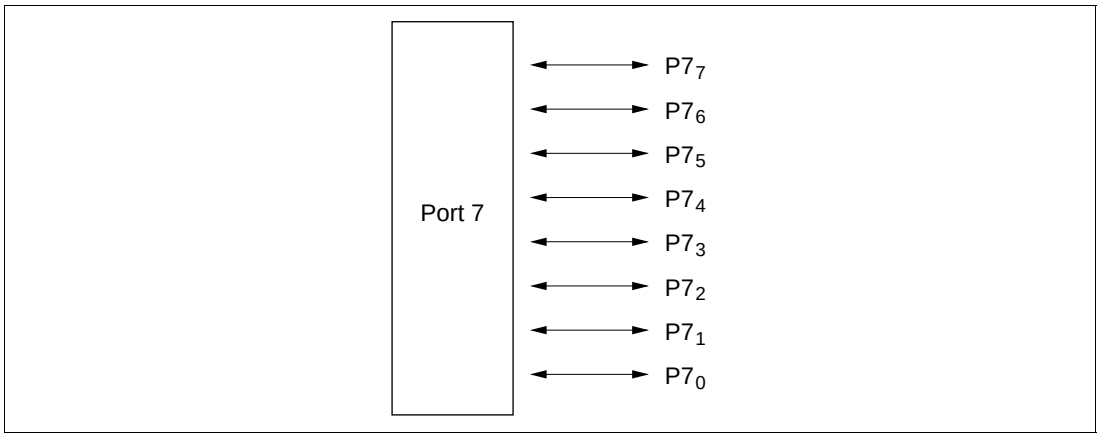


Figure 8.5 Port 7 Pin Configuration

8.6.2 Register Configuration and Description

Table 8.14 shows the port 7 register configuration.

Table 8.14 Port 7 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 7	PDR7	R/W	H'00	H'FFDA
Port control register 7	PCR7	W	H'00	H'FFEA

Port Data Register 7 (PDR7)

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

PDR7 is an 8-bit register that stores data for port 7 pins $P7_7$ to $P7_0$. If port 7 is read while PCR7 bits are set to 1, the values stored in PDR7 are directly read, regardless of the actual pin states. If port 7 is read while PCR7 bits are cleared to 0, the pin states are read.

Upon reset, PDR7 is initialized to H'00.

Port Control Register 7 (PCR7)

Bit	7	6	5	4	3	2	1	0
	$PCR7_7$	$PCR7_6$	$PCR7_5$	$PCR7_4$	$PCR7_3$	$PCR7_2$	$PCR7_1$	$PCR7_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR7 is an 8-bit register for controlling whether each of the port 7 pins $P7_7$ to $P7_0$ functions as an input pin or output pin. Setting a PCR7 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCR7 is initialized to H'00.

PCR7 is a write-only register. All bits are read as 1.

8.6.3 Pin Functions

Table 8.15 shows the port 7 pin functions.

Table 8.15 Port 7 Pin Functions

Pin	Pin Functions and Selection Method
$P7_7$ to $P7_0$	The pin function depends on bit $PCR7_n$ in PCR7. (n = 7 to 0)

$PCR7_n$	0	1
Pin function	$P7_n$ input pin	$P7_n$ output pin

8.6.4 Pin States

Table 8.16 shows the port 7 pin states in each operating mode.

Table 8.16 Port 7 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P7_7$ to $P7_0$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.7 Port 8

8.7.1 Overview

Port 8 is an 8-bit I/O port configured as shown in figure 8.6.

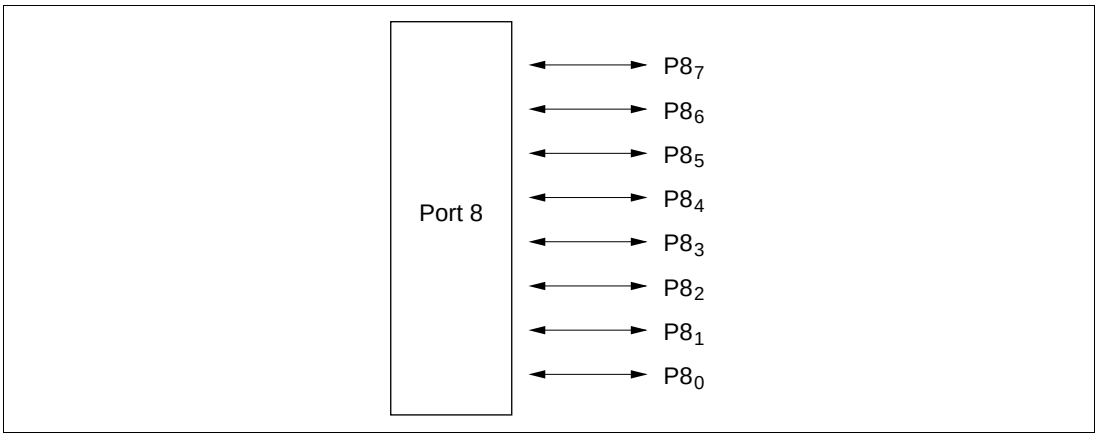


Figure 8.6 Port 8 Pin Configuration

8.7.2 Register Configuration and Description

Table 8.17 shows the port 8 register configuration.

Table 8.17 Port 8 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 8	PDR8	R/W	H'00	H'FFDB
Port control register 8	PCR8	W	H'00	H'FFEB

Port Data Register 8 (PDR8)

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

PDR8 is an 8-bit register that stores data for port 8 pins $P8_7$ to $P8_0$. If port 8 is read while PCR8 bits are set to 1, the values stored in PDR8 are directly read, regardless of the actual pin states. If port 8 is read while PCR8 bits are cleared to 0, the pin states are read.

Upon reset, PDR8 is initialized to H'00.

Port Control Register 8 (PCR8)

Bit	7	6	5	4	3	2	1	0
	$PCR8_7$	$PCR8_6$	$PCR8_5$	$PCR8_4$	$PCR8_3$	$PCR8_2$	$PCR8_1$	$PCR8_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR8 is an 8-bit register for controlling whether each of the port 8 pins $P8_7$ to $P8_0$ functions as an input or output pin. Setting a PCR8 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCR8 is initialized to H'00.

PCR8 is a write-only register. All bits are read as 1.

8.7.3 Pin Functions

Table 8.18 gives the port 8 pin functions.

Table 8.18 Port 8 Pin Functions

Pin	Pin Functions and Selection Method
$P8_7$ to $P8_0$	The pin function depends on bit $PCR8_n$ in PCR8. (n = 7 to 0)

$PCR8_n$	0	1
Pin function	$P8_n$ input pin	$P8_n$ output pin

8.7.4 Pin States

Table 8.19 shows the port 8 pin states in each operating mode.

Table 8.19 Port 8 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P8_7$ to $P8_0$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.8 Port A

8.8.1 Overview

Port A is a 3-bit I/O port configured as shown in figure 8.7.

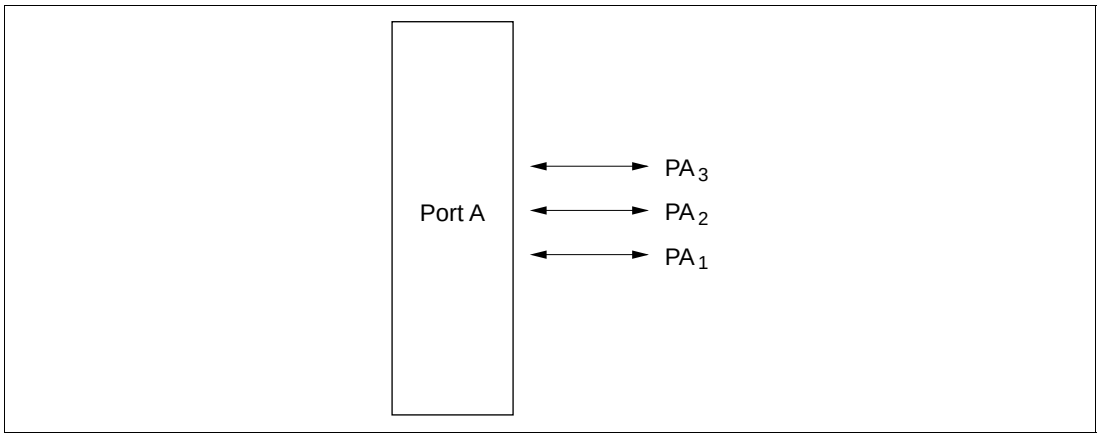


Figure 8.7 Port A Pin Configuration

8.8.2 Register Configuration and Description

Table 8.20 shows the port A register configuration.

Table 8.20 Port A Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register A	PDRA	R/W	H'F0	H'FFDD
Port control register A	PCRA	W	H'F1	H'FFED

Port Data Register A (PDRA)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PA_3	PA_2	PA_1	—
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	—

PDRA is an 8-bit register that stores data for port A pins PA_3 to PA_1. If port A is read while PCRA bits are set to 1, the values stored in PDRA are directly read, regardless of the actual pin states. If port A is read while PCRA bits are cleared to 0, the pin states are read.

Upon reset, PDRA is initialized to H'F0.

Port Control Register A (PCRA)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	$PCRA_3$	$PCRA_2$	$PCRA_1$	—
Initial value	1	1	1	1	0	0	0	1
Read/Write	—	—	—	—	W	W	W	—

PCRA is an 8-bit register for controlling whether each of the port A pins PA_3 to PA_1 functions as an input or output pin. Setting a PCRA bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCRA is initialized to H'F1.

PCRA is a write-only register. All bits are read as 1.

8.8.3 Pin Functions

Table 8.21 gives the port A pin functions.

Table 8.21 Port A Pin Functions

Pin	Pin Functions and Selection Method
PA_3 to PA_1	The pin function depends on bit $PCRA_n$ in PCRA. (n = 3 to 1)

$PCRA_n$	0	1
Pin function	PA_n input pin	PA_n output pin

8.8.4 Pin States

Table 8.22 shows the port A pin states in each operating mode.

Table 8.22 Port A Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
PA_3 to PA_1	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.9 Port B

8.9.1 Overview

Port B is an 2-bit input-only port configured as shown in figure 8.8.

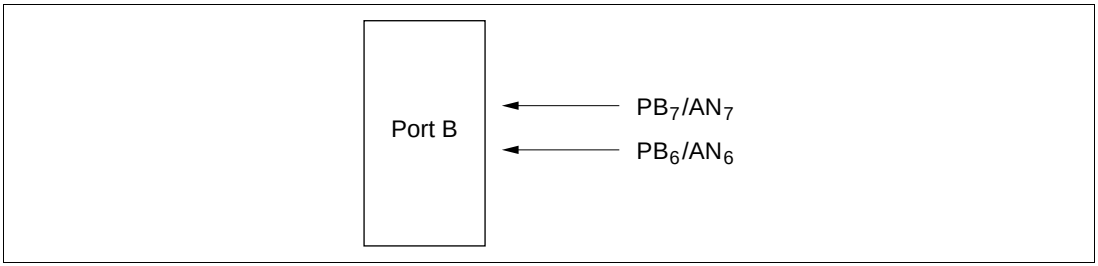


Figure 8.8 Port B Pin Configuration

8.9.2 Register Configuration and Description

Table 8.23 shows the port B register configuration.

Table 8.23 Port B Register

Name	Abbrev.	R/W	Address
Port data register B	PDRB	R	H'FFDE

Port Data Register B (PDRB)

Bit	7	6	5	4	3	2	1	0
	PB_7	PB_6	—	—	—	—	—	—
Read/Write	R	R	—	—	—	—	—	—

Reading PDRB always gives the pin states. However, if a port B pin is selected as an analog input channel for the A/D converter by AMR bits CH3 to CH0, that pin reads 0 regardless of the input voltage.

Section 9 Timers

9.1 Overview

The H8/3627 Series provides three timers (timers A, F, and G) on-chip.

Table 9.1 outlines the functions of timers A, F, and G.

Table 9.1 Timer Functions

Name	Functions		Internal Clock	Event Input Pin	Waveform Output Pin	Remarks
Timer A	8-bit timer	Interval timer	ø/8 to ø/8192 (8 choices)	—	—	
		Time base	$ø_W$/128 (choice of 4 overflow periods)	—	—	
		Clock output	ø/4 to ø/32, $ø_W$/4 to $ø_W$/32 (8 choices)	—	TMOW	
Timer F	• 16-bit free-running timer • Event counter • Can be used as two independent 8-bit timers • Output compare		ø/2 to ø/32 (4 choices)	TMIF	TMOFL TMOFH	
Timer G	• 8-bit timer • Input capture • Interval timer		ø/2 to ø/64, $ø_W$/2 (4 choices)	TMIG	—	• Counter clear designation possible • Built-in noise canceller circuit for input capture

9.2 Timer A

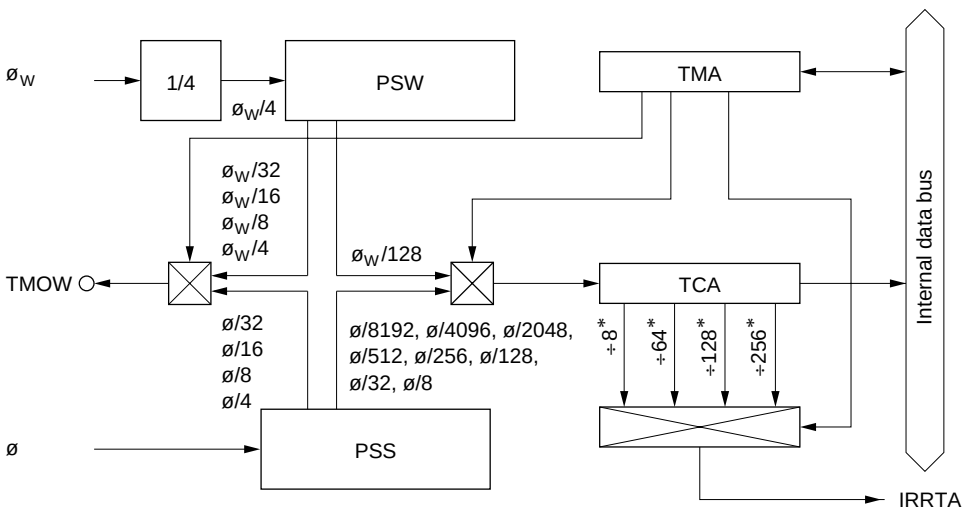
9.2.1 Overview

Timer A is an 8-bit timer with interval timing and real-time clock time-base functions. The clock time-base function is available when a 32.768-kHz crystal oscillator is connected. A clock signal divided from 32.768 kHz or from the system clock can be output at the TMOW pin.

Features: Features of timer A are given below.

- Choice of eight internal clock sources (ø/8192, ø/4096, ø/2048, ø/512, ø/256, ø/128, ø/32, ø/8).
- Choice of four overflow periods (1 s, 0.5 s, 0.25 s, 31.25 ms) when timer A is used as a clock time base (using a 32.768 kHz crystal oscillator).
- An interrupt is requested when the counter overflows.
- Any of eight clock signals can be output from pin TMOW: 32.768 kHz divided by 32, 16, 8, or 4 (1 kHz, 2 kHz, 4 kHz, 8 kHz), or the system clock divided by 32, 16, 8, or 4.

Block Diagram: Figure 9.1 shows a block diagram of timer A.



Legend:
TMA: Timer mode register A
TCA: Timer counter A
IRRTA: Timer A overflow interrupt request flag
PSW: Prescaler W
PSS: Prescaler S

Note: Can be selected only when the prescaler W output ($ø_W$/128) is used as the TCA input clock.

Figure 9.1 Block Diagram of Timer A

Pin Configuration: Table 9.2 shows the timer A pin configuration.

Table 9.2 Pin Configuration

Name	Abbrev.	I/O	Function
Clock output	TMOW	Output	Output of waveform generated by timer A output circuit

Register Configuration: Table 9.3 shows the register configuration of timer A.

Table 9.3 Timer A Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register A	TMA	R/W	H'10	H'FFB0
Timer counter A	TCA	R	H'00	H'FFB1

9.2.2 Register Descriptions

Timer Mode Register A (TMA)

Bit	7	6	5	4	3	2	1	0
	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0
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

TMA is an 8-bit read/write register for selecting the prescaler, input clock, and output clock.

Upon reset, TMA is initialized to H'10.

Bits 7 to 5—Clock Output Select (TMA7 to TMA5): Bits 7 to 5 choose which of eight clock signals is output at the TMOW pin. The system clock divided by 32, 16, 8, or 4 can be output in active mode and sleep mode. A 32.768 kHz signal divided by 32, 16, 8, or 4 can be output in active mode, sleep mode, and subactive mode.

Bit 7: TMA7	Bit 6: TMA6	Bit 5: TMA5	Clock Output	
0	0	0	ø/32	(initial value)
		1	ø/16	
	1	0	ø/8	
		1	ø/4	
1	0	0	$ø_w$/32	
		1	$ø_w$/16	
	1	0	$ø_w$/8	
		1	$ø_w$/4	

Bit 4—Reserved Bit: Bit 4 is reserved; it is always read as 1, and cannot be modified.

Bits 3 to 0—Internal Clock Select (TMA3 to TMA0): Bits 3 to 0 select the clock input to TCA. The selection is made as follows.

				Description		
Bit 3: TMA3	**Bit 2: TMA2**	**Bit 1: TMA1**	**Bit 0: TMA0**	**Prescaler and Divider Ratio or Overflow Period**		**Function**
0	0	0	0	PSS, ø/8192	(initial value)	Interval timer
			1	PSS, ø/4096		
		1	0	PSS, ø/2048		
			1	PSS, ø/512		
	1	0	0	PSS, ø/256		
			1	PSS, ø/128		
		1	0	PSS, ø/32		
			1	PSS, ø/8		
1	0	0	0	PSW, 1 s		Clock time base
			1	PSW, 0.5 s		
		1	0	PSW, 0.25 s		
			1	PSW, 0.03125 s		
	1	0	0	PSW and TCA are reset		
			1			
		1	0			
			1			

Timer Counter A (TCA)

Bit	7	6	5	4	3	2	1	0
	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TCA is an 8-bit read-only up-counter, which is incremented by internal clock input. The clock source for input to this counter is selected by bits TMA3 to TMA0 in timer mode register A (TMA). TCA values can be read by the CPU in active mode, but cannot be read in subactive mode. When TCA overflows, the IRRTA bit in interrupt request register 1 (IRR1) is set to 1.

TCA is cleared by setting bits TMA3 and TMA2 of TMA to 11.

Upon reset, TCA is initialized to H'00.

9.2.3 Timer Operation

Interval Timer Operation: When bit TMA3 in timer mode register A (TMA) is cleared to 0, timer A functions as an 8-bit interval timer.

Upon reset, TCA is cleared to H'00 and bit TMA3 is cleared to 0, so up-counting and interval timing resume immediately. The clock input to timer A is selected by bits TMA2 to TMA0 in TMA; any of eight internal clock signals output by prescaler S can be selected.

After the count value in TCA reaches H'FF, the next clock signal input causes timer A to overflow, setting bit IRRTA to 1 in interrupt request register 1 (IRR1). If IENTA = 1 in interrupt enable register 1 (IENR1), a CPU interrupt is requested.*

At overflow, TCA returns to H'00 and starts counting up again. In this mode timer A functions as an interval timer that generates an overflow output at intervals of 256 input clock pulses.

Note: * For details on interrupts, see 3.3, Interrupts.

Real-Time Clock Time Base Operation: When bit TMA3 in TMA is set to 1, timer A functions as a real-time clock time base by counting clock signals output by prescaler W.

The overflow period of timer A is set by bits TMA1 and TMA0 in TMA. A choice of four periods is available. In time base operation (TMA3 = 1), setting bit TMA2 to 1 clears both TCA and prescaler W to their initial values of H'00.

Clock Output: Setting bit TMOW in port mode register 1 (PMR1) to 1 causes a clock signal to be output at pin TMOW. Eight different clock output signals can be selected by means of bits TMA7 to TMA5 in TMA. The system clock divided by 32, 16, 8, or 4 can be output in active mode and sleep mode. A 32.768 kHz signal divided by 32, 16, 8, or 4 can be output in active mode, sleep mode, and subactive mode.

9.2.4 Timer A Operation States

Table 9.4 summarizes the timer A operation states.

Table 9.4 Timer A Operation States

Operation Mode		Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCA	Interval	Reset	Functions	Functions	Halted	Halted	Halted	Halted
	Clock time base	Reset	Functions	Functions	Functions	Functions	Functions	Halted
TMA		Reset	Functions	Retained	Retained	Functions	Retained	Retained

Note: When real-time clock time-base functions are selected as the internal clock of TCA in active mode or sleep mode, the internal clock is not synchronous with the system clock, so it is synchronized by a synchronizing circuit. This may result in a maximum error of 1/ø (s) in the count cycle.

9.3 Timer F

9.3.1 Overview

Timer F is a 16-bit timer with an output compare function. Compare match signals can be used to reset the counter, request an interrupt, or toggle the output. Timer F can also be used for external event counting, and can operate as two independent 8-bit timers, timer FH and timer FL.

Features: Features of timer F are given below.

- Choice of four internal clock sources (ø/32, ø/16, ø/4, ø/2) or an external clock (can be used as an external event counter).
- Output from pin TMOFH is toggled by one compare match signal (the initial value of the toggle output can be set).
- Counter can be reset by the compare match signal.
- Two interrupt sources: counter overflow and compare match.
- Can operate as two independent 8-bit timers (timer FH and timer FL) in 8-bit mode.

  Timer FH
  - 8-bit timer (clocked by timer FL overflow signals when timer F operates as a 16-bit timer).
  - Choice of four internal clocks (ø/32, ø/16, ø/4, ø/2).
  - Output from pin TMOFH is toggled by one compare match signal (the initial value of the toggle output can be set).
  - Counter can be reset by the compare match signal.
  - Two interrupt sources: counter overflow and compare match.

  Timer FL
  - 8-bit timer/event counter
  - Choice of four internal clocks (ø/32, ø/16, ø/4, ø/2) or event input at pin TMIF.
  - Output from pin TMOFL is toggled by one compare match signal (the initial value of the toggle output can be set).
  - Counter can be reset by the compare match signal.
  - Two interrupt sources: counter overflow and compare match.

Block Diagram: Figure 9.2 shows a block diagram of timer F.

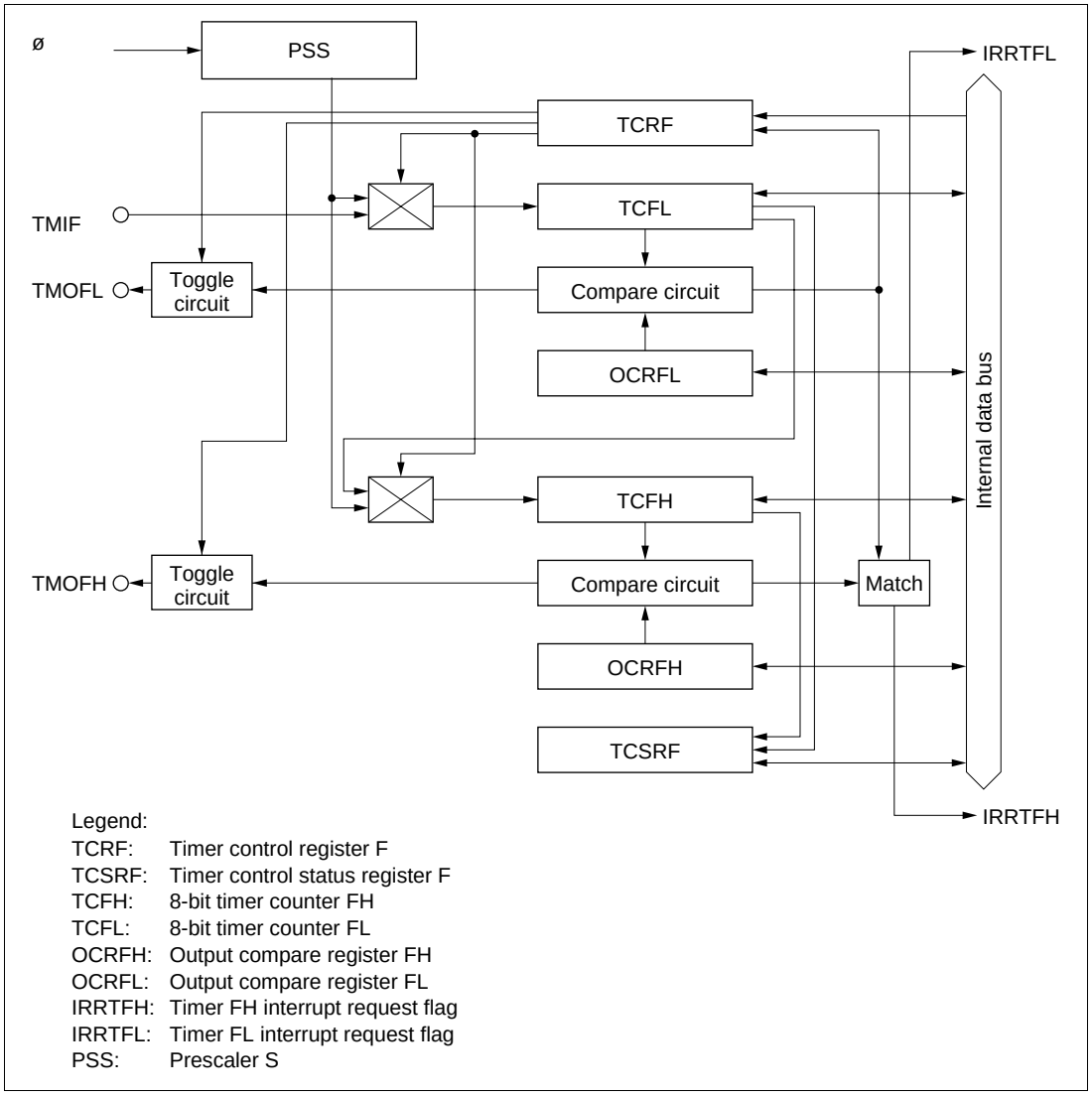


Figure 9.2 Block Diagram of Timer F

Pin Configuration: Table 9.5 shows the timer F pin configuration.

Table 9.5 Pin Configuration

Name	Abbrev.	I/O	Function
Timer F event input	TMIF	Input	Event input to TCFL
Timer FH output	TMOFH	Output	Timer FH toggle output
Timer FL output	TMOFL	Output	Timer FL toggle output

Register Configuration: Table 9.6 shows the register configuration of timer F.

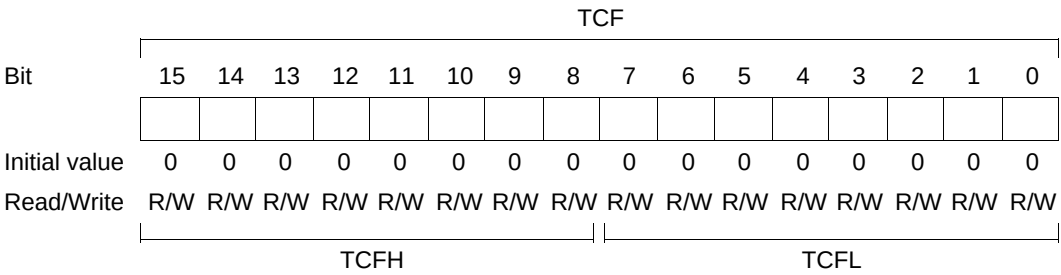
Table 9.6 Timer F Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer control register F	TCRF	W	H'00	H'FFB6
Timer control/status register F	TCSRF	R/W	H'00	H'FFB7
8-bit timer counter FH	TCFH	R/W	H'00	H'FFB8
8-bit timer counter FL	TCFL	R/W	H'00	H'FFB9
Output compare register FH	OCRFH	R/W	H'FF	H'FFBA
Output compare register FL	OCRFL	R/W	H'FF	H'FFBB

9.3.2 Register Descriptions

16-Bit Timer Counter (TCF)

8-Bit Timer Counter (TCFH)
8-Bit Timer Counter (TCFL)



TCF is a 16-bit read/write up-counter consisting of two cascaded 8-bit timer counters, TCFH and TCFL. TCF can be used as a 16-bit counter, with TCFH as the upper 8 bits and TCFL as the lower 8 bits of the counter, or TCFH and TCFL can be used as independent 8-bit counters.

TCFH and TCFL can be read and written by the CPU, but in 16-bit mode, data transfer with the CPU takes place via a temporary register (TEMP). For details see 9.3.3, Interface with the CPU.

Upon reset, TCFH and TCFL are each initialized to H'00.

16-Bit Mode (TCF): 16-bit mode is selected by clearing bit CKSH2 to 0 in timer control register F (TCRF). The TCF input clock is selected by TCRF bits CKSL2 to CKSL0.

Timer control status register F (TCSRF) can be set so that counter TCF will be cleared by compare match.

When TCF overflows from H'FFFF to H'0000, the overflow flag (OVFH) in TCSRF is set to 1. If bit OVIEH in TCSRF is set to 1 when an overflow occurs, bit IRRTFH in interrupt request register 2 (IRR2) will be set to 1; and if bit IENTFH in interrupt enable register 2 (IENR2) is set to 1, a CPU interrupt will be requested.

8-Bit Mode (TCFH, TCFL): When bit CKSH2 in timer control register F (TCRF) is set to 1, timer F functions as two separate 8-bit counters, TCFH and TCFL. The TCFH (TCFL) input clock is selected by TCRF bits CKSH2 to CKSH0 (CKSL2 to CKSL0).

TCFH (TCFL) can be cleared by a compare match signal. This designation is made in bit CCLRH (CCLRL) in TCSRF.

When TCFH (TCFL) overflows from H'FF to H'00, the overflow flag OVFH (OVFL) in TCSRF is set to 1. If bit OVIEH (OVIEL) in TCSRF is set to 1 when an overflow occurs, bit IRRTFH (IRRTHL) in interrupt request register 2 (IRR2) will be set to 1; and if bit IENTFH (IENTFL) in interrupt enable register 2 (IENR2) is set to 1, a CPU interrupt will be requested.

16-Bit Output Compare Register (OCRF)

8-Bit Output Compare Register (OCRFH)
8-Bit Output Compare Register (OCRFL)

	OCRF															
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
	OCRFH								OCRFL							

OCRF is a 16-bit read/write output compare register consisting of two 8-bit read/write registers OCRFH and OCRFL. It can be used as a 16-bit output compare register, with OCRFH as the

upper 8 bits and OCRFL as the lower 8 bits of the register, or OCRFH and OCRFL can be used as independent 8-bit registers.

OCRFH and OCRFL can be read and written by the CPU, but in 16-bit mode, data transfer with the CPU takes place via a temporary register (TEMP). For details see 9.3.3, Interface with the CPU.

Upon reset, OCRFH and OCRFL are each initialized to H'FF.

16-Bit Mode (OCRF): 16-bit mode is selected by clearing bit CKSH2 to 0 in timer control register F (TCRF). The OCRF contents are always compared with the 16-bit timer counter (TCF). When the contents match, the compare match flag (CMFH) in TCSRF is set to 1. Also, IRRTFH in interrupt request register 2 (IRR2) is set to 1. If bit IENTFH in interrupt enable register 2 (IENR2) is set to 1, a CPU interrupt is requested.

Output for pin TMOFH can be toggled by compare match. The output level can also be set to high or low by bit TOLH of timer control register F (TCRF).

8-Bit Mode (OCRFH, OCRFL): Setting bit CKSH2 in TCRF to 1 results in two 8-bit independent registers, OCRFH and OCRFL.

The OCRFH contents are always compared with TCFH, and the OCRFL contents are always compared with TCFL. When the contents match, the compare match flag (CMFH or CMFL) in TCSRF is set to 1. Also, bit IRRTFH (IRRTFL) in interrupt request register 2 (IRR2) set to 1. If bit IENTFH (IENTFL) in interrupt enable register 2 (IENR2) is set to 1 at this time, a CPU interrupt is requested.

The output at pin TMOFH (TMOFL) can be toggled by compare match. The output level can also be set to high or low by bit TOLH (TOLL) of the timer control register (TCRF).

Timer Control Register F (TCRF)

Bit	7	6	5	4	3	2	1	0
	TOLH	CKSH2	CKSH1	CKSH0	TOLL	CKSL2	CKSL1	CKSL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

TCRF is an 8-bit write-only register. It is used to switch between 16-bit mode and 8-bit mode, to select among four internal clocks and an external event, and to select the output level at pins TMOFH and TMOFL.

Upon reset, TCRF is initialized to H'00.

Bit 7—Toggle Output Level H (TOLH): Bit 7 sets the output level at pin TMOFH. The setting goes into effect immediately after this bit is written.

Bit 7: TOLH	Description	
0	Low level	(initial value)
1	High level	

Bits 6 to 4—Clock Select H (CKSH2 to CKSH0): Bits 6 to 4 select the input to TCFH from four internal clock signals or the overflow of TCFL.

Bit 6: CKSH2	Bit 5: CKSH1	Bit 4: CKSH0	Description
0	*	*	16-bit mode selected. TCFL overflow signals are counted. (initial value)
1	0	0	Internal clock: ø/32
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	Internal clock: ø/2

Note: * Don't care

Bit 3—Toggle Output Level L (TOLL): Bit 3 sets the output level at pin TMOFL. The setting goes into effect immediately after this bit is written.

Bit 3: TOLL	Description	
0	Low level	(initial value)
1	High level	

Bits 2 to 0—Clock Select L (CKSL2 to CKSL0): Bits 2 to 0 select the input to TCFL from four internal clock signals or external event input.

Bit 2: CKSL2	Bit 1: CKSL1	Bit 0: CKSL0	Description
0	*	*	External event (TMIF). Rising or falling edge is counted (see note). (initial value)
1	0	0	Internal clock: ø/32
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	Internal clock: ø/2

*: Don't care

Note: The edge of the external event signal is selected by bit IEG3 in the IRQ edge select register (IEGR). See 3.3.2, Interrupt Control Registers, for details on the IRQ edge select register. Note that switching the TMIF pin function by changing bit IRQ3 in port mode register 1 (PMR1) from 0 to 1 or from 1 to 0 while the TMIF pin is at the low level may cause the timer F counter to be incremented.

Timer Control/Status Register F (TCSRF)

Bit	7	6	5	4	3	2	1	0
	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL
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

TCSRF is an 8-bit read/write register. It is used for counter clear selection, overflow and compare match indication, and enabling of interrupts caused by timer overflow.

Upon reset, TCSRF is initialized to H'00.

Bit 7—Timer Overflow Flag H (OVFH): Bit 7 is a status flag indicating TCFH overflow (H'FF to H'00). This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 7: OVFH	Description
0	[Clearing conditions] (initial value) After reading OVFH = 1, cleared by writing 0 to OVFH
1	[Setting conditions] Set when the value of TCFH goes from H'FF to H'00

Bit 6—Compare Match Flag H (CMFH): Bit 6 is a status flag indicating a compare match between TCFH and OCRFH. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 6: CMFH	Description	
0	[Clearing conditions] After reading CMFH = 1, cleared by writing 0 to CMFH	(initial value)
1	[Setting conditions] Set when the TCFH value matches OCRFH value	

Bit 5—Timer Overflow Interrupt Enable H (OVIEH): Bit 5 enables or disables TCFH overflow interrupts.

Bit 5: OVIEH	Description	
0	TCFH overflow interrupt disabled	(initial value)
1	TCFH overflow interrupt enabled	

Bit 4—Counter Clear H (CCLRH): In 16-bit mode, bit 4 selects whether or not TCF is cleared when a compare match occurs between TCF and OCRF.

In 8-bit mode, bit 4 selects whether or not TCFH is cleared when a compare match occurs between TCFH and OCRFH.

Bit 4: CCLRH	Description	
0	16-bit mode: TCF clearing by compare match disabled 8-bit mode: TCFH clearing by compare match disabled	(initial value)
1	16-bit mode: TCF clearing by compare match enabled 8-bit mode: TCFH clearing by compare match enabled	

Bit 3—Timer Overflow Flag L (OVFL): Bit 3 is a status flag indicating TCFL overflow (H'FF to H'00). This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 3: OVFL	Description	
0	[Clearing conditions] After reading OVFL = 1, cleared by writing 0 to OVFL	(initial value)
1	[Setting conditions] Set when the value of TCFL goes from H'FF to H'00	

Bit 2—Compare Match Flag L (CMFL): Bit 2 is a status flag indicating a compare match between TCFL and OCRFL. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 2: CMFL	Description	
0	[Clearing conditions] After reading CMFL = 1, cleared by writing 0 to CMFL	(initial value)
1	[Setting conditions] Set when the TCFL value matches the OCRFL value	

Bit 1—Timer Overflow Interrupt Enable L (OVIEL): Bit 1 enables or disables TCFL overflow interrupts.

Bit 1: OVIEL	Description	
0	TCFL overflow interrupt disabled	(initial value)
1	TCFL overflow interrupt enabled	

Bit 0—Counter Clear L (CCLRL): Bit 0 selects whether or not TCFL is cleared when a compare match occurs between TCFL and OCRFL.

Bit 0: CCLRL	Description	
0	TCFL clearing by compare match disabled	(initial value)
1	TCFL clearing by compare match enabled	

9.3.3 Interface with the CPU

TCF and OCRF are 16-bit read/write registers, whereas the data bus between the CPU and on-chip peripheral modules has an 8-bit width. For this reason, when the CPU accesses TCF or OCRF, it makes use of an 8-bit temporary register (TEMP).

In 16-bit mode, when reading or writing TCF or writing OCRF, always use two consecutive byte size MOV instructions, and always access the upper byte first. Data will not be transferred properly if only the upper byte or only the lower byte is accessed. In 8-bit mode there is no such restriction on the order of access.

Write Access: When the upper byte is written, the upper-byte data is loaded into the TEMP register. Next when the lower byte is written, the data in TEMP goes to the upper byte of the register, and the lower-byte data goes directly to the lower byte of the register. Figure 9.3 shows a TCF write operation when H'AA55 is written to TCF.

Read Access: When the upper byte of TCF is read, the upper-byte data is sent directly to the CPU, and the lower byte is loaded into TEMP. Next when the lower byte is read, the lower byte in TEMP is sent to the CPU.

When the upper byte of OCRF is read, the upper-byte data is sent directly to the CPU. Next when the lower byte is read, the lower-byte data is sent directly to the CPU.

Figure 9.4 shows a TCF read operation when H'AAFF is read from TCF.

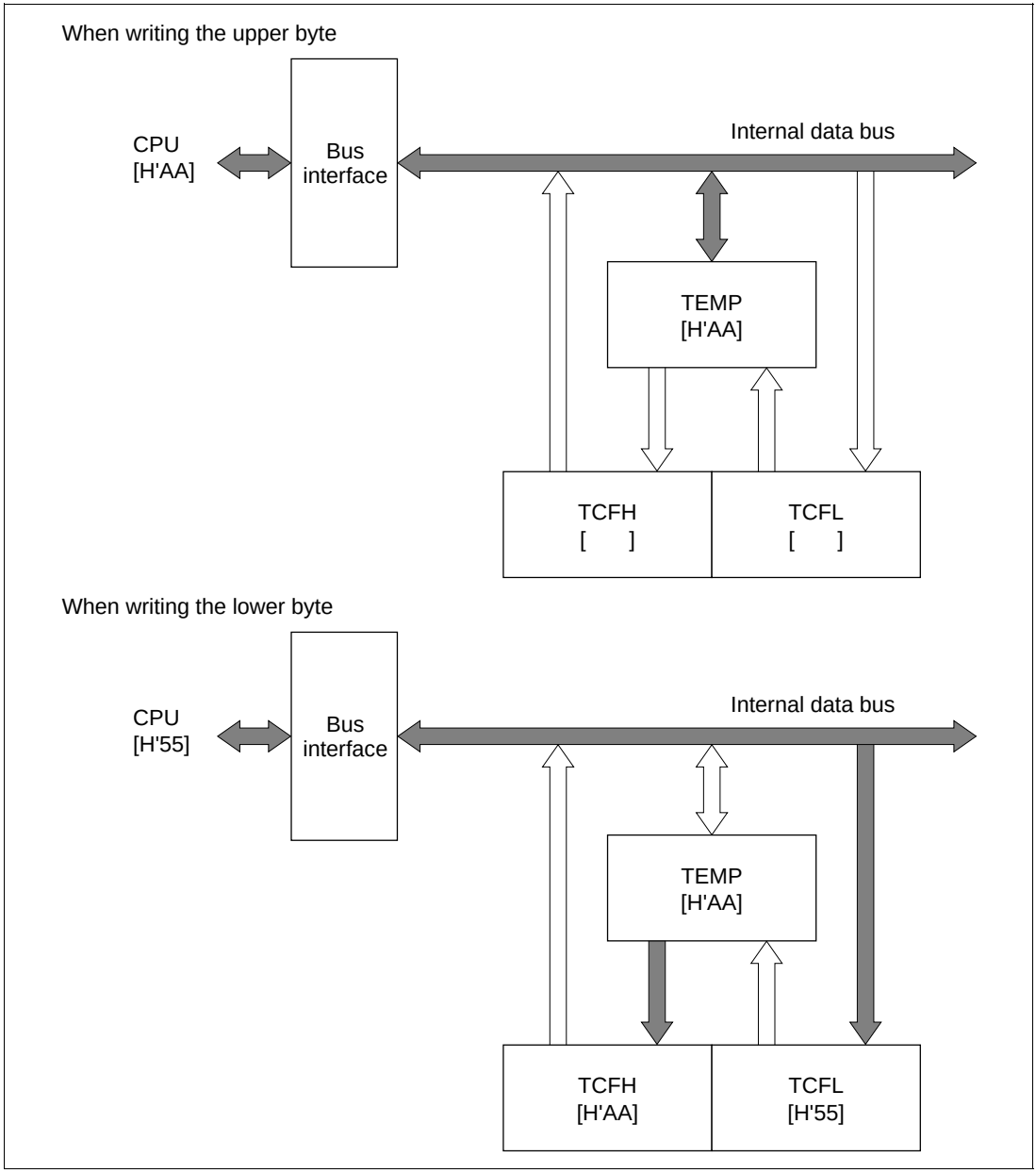
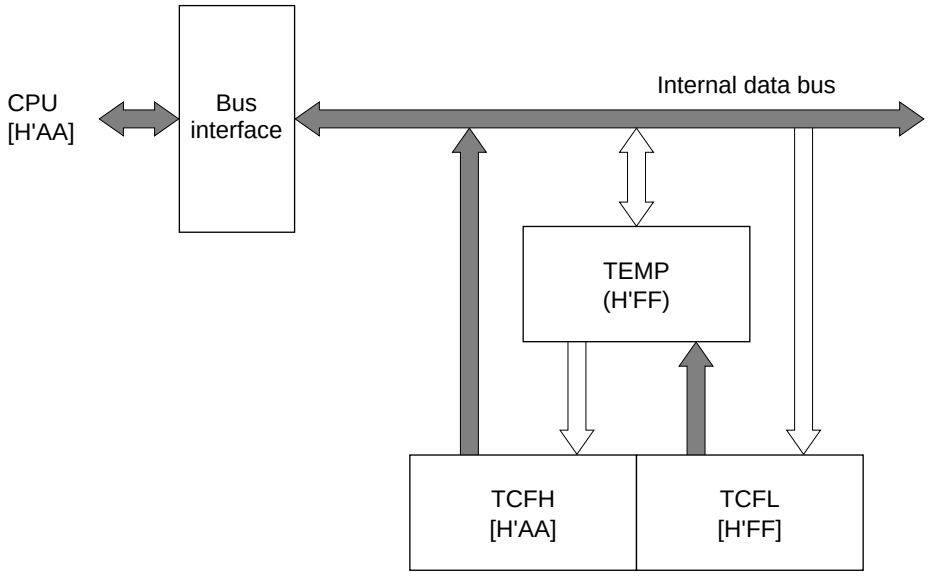
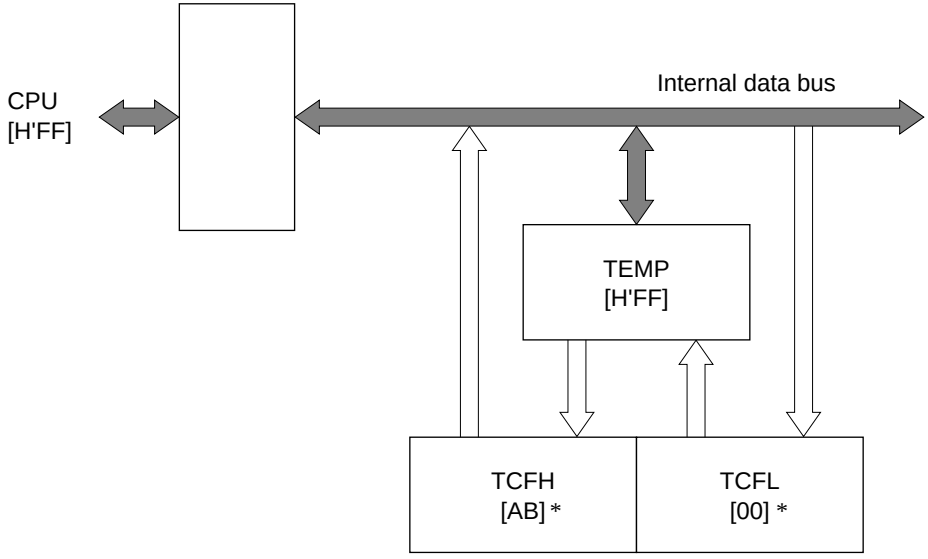


Figure 9.3 TCF Write Operation (CPU → TCF)

When reading the upper byte



When reading the lower byte



Note: * Becomes H'AB00 if counter is incremented once.

Figure 9.4 TCF Read Operation (TCF → CPU)

9.3.4 Timer Operation

Timer F is a 16-bit timer/counter that increments with each input clock. When the value set in output compare register F matches the count in timer F, the timer can be cleared, an interrupt can be requested, and the port output can be toggled. Timer F can also be used as two independent 8-bit timers.

Timer F Operation: Timer F can operate in either 16-bit timer mode or 8-bit timer mode. These modes are described below.

- 16-bit timer mode

  Timer F operates in 16-bit timer mode when the CKSH2 bit in timer control register F (TCRF) is cleared to 0.

  A reset initializes timer counter F (TCF) to H'0000, output compare register F (OCRF) to H'FFFF, and timer control register F (TCRF) and timer control status register F (TCSRF) to H'00. Timer F begins counting external event input signals (TMIF). The edge of the external event signal is selected by the IEG3 bit in the IRQ edge select register (IEGR).

  The operational clock of timer F can be selected by setting bits CSKL2 through CKSL0 in TCRF, from four internal clocks output from prescaler S as well as from an external clock.

  TCF is continuously compared with the contents of OCRF. When these two values match, the CMFH bit in TCSRF is set to 1. At this time if IENTFH of IENR2 is 1, a CPU interrupt is requested and the output at pin TMOFH is toggled. If the CCLRH bit in TCSRF is 1, TCF is cleared. The output at pin TMOFH can also be set by the TOLH bit in TCRF.

  If timer F overflows (from H'FFFF to H'0000), the OVFH bit in TCSRF is set to 1. At this time, if the OVIEH bit in TCSRF and the IENTFH bit in IENR2 are both 1, CPU interrupt is requested.

- 8-bit timer mode

  When the CKSH2 bit in TCRF is set to 1, timer F operates as two independent 8-bit timers, TCFH and TCFL. The input clock of TCFH/TCFL is selected by bits CKSH2 to CKSH0/CKSL2 to CKSL0 in TCRF.

  When TCFH/TCFL and the contents of OCRFH/OCRFL match, the CMFH/CMFL bit in TCSRF is set to 1. If the IENTFH/IENTFL bit in IENR2 is 1, a CPU interrupt is requested and the output at pin TMOFH/TMOFL is toggled. If the CCLRH/CCLRL bit in TCRF is 1, TCFH/TCFL is cleared. The output at pin TMOFH/TMOFL can also be set by the TOLH/TOLL bit in TCRF.

  When TCFH/TCFL overflows from H'FF to H'00, the OVFH/OVFL bit in TCSRF is set to 1. At this time, if the OVIEH/OVIEL bit in TCSRF and the IENTFH/IENTFL bit in IENR2 are both 1, a CPU interrupt is requested.

TCF Count Timing: TCF is incremented by each pulse of the input clock (internal clock or external event).

- Internal clock

  The settings of bits CKSH2 to CKSH0 or bits CKSL2 to CKSL0 in TCRF select one of four internal clock signals (ø/32, ø/16, ø/4, or ø/2) divided from the system clock (ø).

- External event

  External event input is selected by clearing bit CKSL2 to 0 in TCRF. Either rising or falling edges of the clock input can be counted. The edge is selected by bit IEG3 in IEGR. An external clock pulse width of at least two system clock cycles (ø) is necessary; otherwise the counter will not operate properly.

TMOFH and TMOFL Output Timing: The outputs at pins TMOFH and TMOFL are the values set in bits TOLH and TOLL in TCRF. When a compare match occurs, the output value is inverted. Figure 9.5 shows the output timing.

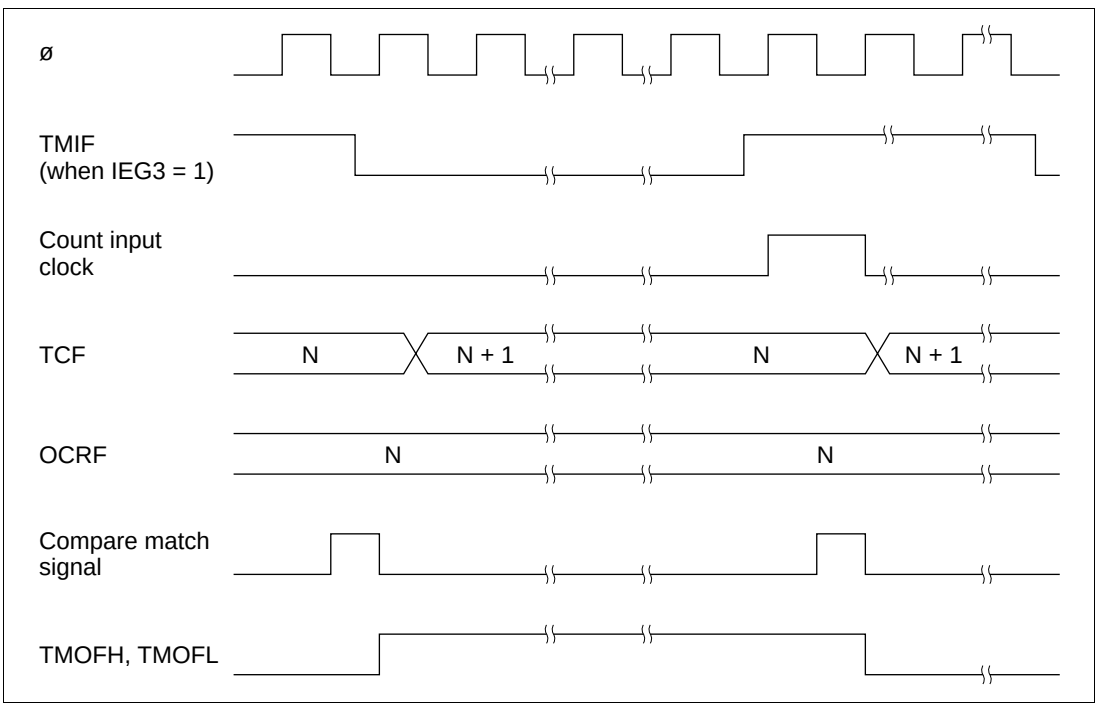


Figure 9.5 TMOFH, TMOFL Output Timing

TCF Clear Timing: TCF can be cleared at compare match with OCRF.

Timer Overflow Flag (OVF) Set Timing: OVF is set to 1 when TCF overflows (goes from H'FFFF to H'0000).

Compare Match Flag Set Timing: The compare match flags (CMFH or CMFL) are set to 1 when a compare match occurs between TCF and OCRF. A compare match signal is generated in the final state in which the values match (when TCF changes from the matching count value to the next value). When TCF and OCRF match, a compare match signal is not generated until the next counter clock pulse.

Timer F Operation States: Table 9.7 summarizes the timer F operation states.

Table 9.7 Timer F Operation States

Operation Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCF	Reset	Functions	Functions	Halted	Halted	Halted	Halted
OCRF	Reset	Functions	Retained	Retained	Retained	Retained	Retained
TCRF	Reset	Functions	Retained	Retained	Retained	Retained	Retained
TCSRF	Reset	Functions	Retained	Retained	Retained	Retained	Retained

9.3.5 Application Notes

The following conflicts can arise in timer F operation.

16-Bit Timer Mode: The output at pin TMOFH toggles when all 16 bits match and a compare match signal is generated. If the compare match signal occurs at the same time as new data is written in TCRF by a MOV instruction, however, the new value written in bit TOLH will be output at pin TMOFH. The TMOFL output in 16-bit mode is indeterminate, so this output should not be used. Use the pin as a general input or output port.

If an OCRFL write occurs at the same time as a compare match signal, the compare match signal is inhibited. If a compare match occurs between the written data and the counter value, however, a compare match signal will be generated at that point. The compare match signal is output in synchronization with the TCFL clock, so if this clock is stopped no compare match signal will be generated, even if a compare match occurs.

Compare match flag CMFH is set when all 16 bits match and a compare match signal is generated; bit CMFL is set when the setting conditions are met for the lower 8 bits.

The overflow flag (OVFH) is set when TCF overflows; bit OVFL is set if the setting conditions are met when the lower 8 bits overflow. If a write to TCFL occurs at the same time as an overflow signal, the overflow signal is not output.

- TCFH and OCRFH

  The output at pin TMOFH toggles when there is a compare match. If the compare match signal occurs at the same time as new data is written in TCRF by a MOV instruction, however, the new value written in bit TOLH will be output at pin TMOFH.

  If an OCRFH write occurs at the same time as a compare match signal, the compare match signal is inhibited. If a compare match occurs between the written data and the counter value, however, a compare match signal will be generated at that point. The compare match signal is output in synchronization with the TCFH clock.

  If a TCFH write occurs at the same time as an overflow signal, the overflow signal is not output.

- TCFL and OCRFL

  The output at pin TMOFL toggles when there is a compare match. If the compare match signal occurs at the same time as new data is written in TCRF by a MOV instruction, however, the new value written in bit TOLL will be output at pin TMOFL.

  If an OCRFL write occurs at the same time as a compare match signal, the compare match signal is inhibited. If a compare match occurs between the written data and the counter value, however, a compare match signal will be generated at that point. The compare match signal is output in synchronization with the TCFL clock, so if this clock is stopped no compare match signal will be generated, even if a compare match occurs.

  If a TCFL write occurs at the same time as an overflow signal, the overflow signal is not output.

9.4 Timer G

9.4.1 Overview

Timer G is an 8-bit timer, with input capture/interval functions for separately capturing the rising edge and falling edge of pulses input at the input capture pin (input capture input signal). Timer G has a built-in noise canceller circuit that can eliminate high-frequency noise from the input capture signal, enabling accurate measurement of its duty cycle. When timer G is not used for input capture, it functions as an 8-bit interval timer.

Features: Features of timer G are given below.

- Choice of four internal clock sources (ø/64, ø/32, ø/2, $ø_W$/2)
- Input capture function
  Separate input capture functions are provided for the rising and falling edges.
- Counter overflow detection
  Can detect whether overflow occurred when the input capture signal was high or low.
- Choice of counter clear
  It is possible to select whether or not the counter is cleared at the rising edge, falling edge, or both edges of the input capture input signal.
- Two interrupt sources
  There is one input capture interrupt source and one overflow interrupt source. For input capture, the rising or falling edge can be selected.
- Built-in noise-canceller circuit
  The noise canceller circuit can eliminate high-frequency noise in the input capture signal.
- Operates in subactive and subsleep modes
  When $ø_W$/2 is selected as the internal clock source, timer G can operate in the subactive and subsleep modes.

Block Diagram: Figure 9.6 shows a block diagram of timer G.

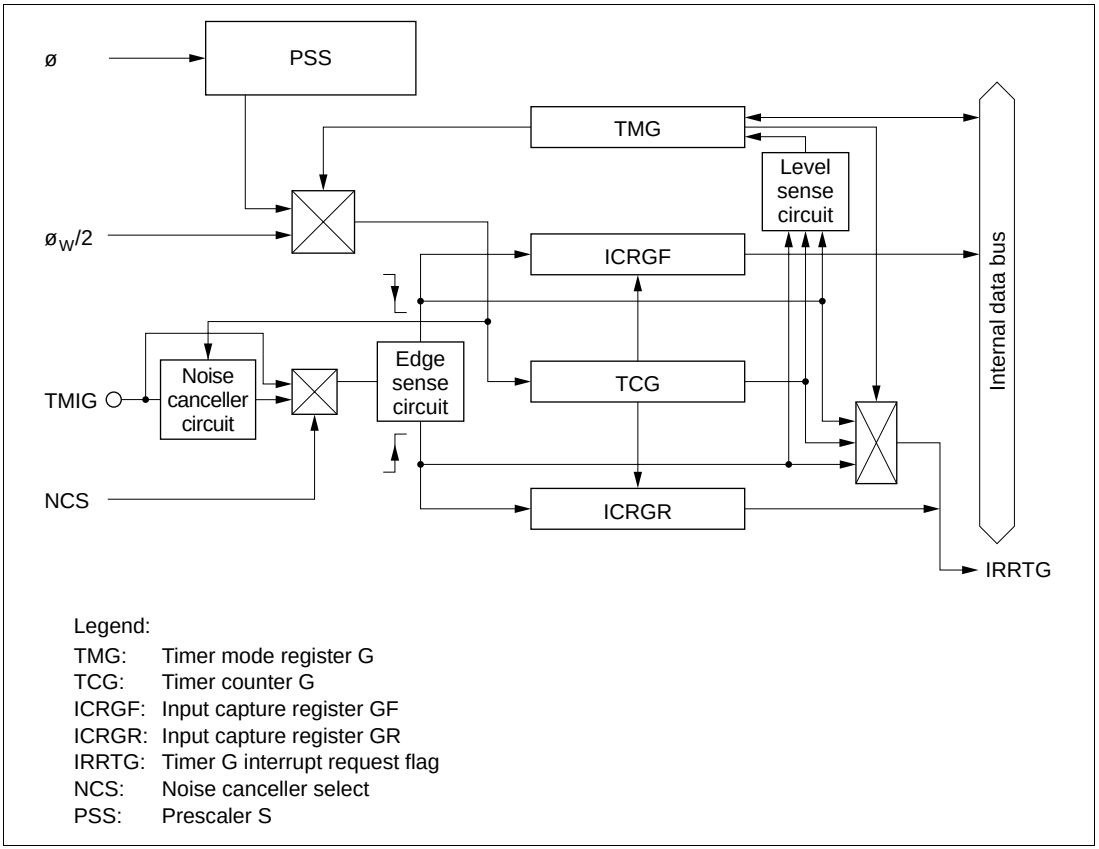


Figure 9.6 Block Diagram of Timer G

Pin Configuration: Table 9.8 shows the timer G pin configuration.

Table 9.8 Pin Configuration

Name	Abbrev.	I/O	Function
Input capture input	TMIG	Input	Input capture

Register Configuration: Table 9.9 shows the register configuration of timer G.

Table 9.9 Timer G Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register G	TMG	R/W	H'00	H'FFBC
Timer counter G	TCG	—	H'00	—
Input capture register GF	ICRGF	R	H'00	H'FFBD
Input capture register GR	ICRGR	R	H'00	H'FFBE

9.4.2 Register Descriptions

Timer Counter G (TCG)

Bit	7	6	5	4	3	2	1	0
	TCG7	TCG6	TCG5	TCG4	TCG3	TCG2	TCG1	TCG0
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	—	—	—

Timer counter G (TCG) is an 8-bit up-counter which is incremented by an input clock. The input clock signal is selected by bits CKS1 and CKS0 in timer mode register G (TMG).

To use TCG as an input capture timer, set bit TMIG to 1 in PMR1; to use TCG as an interval timer, clear bit TMIG to 0.* When TCG is used as an input capture timer, the TCG value can be cleared at the rising edge, falling edge, or both edges of the input capture signal, depending on settings in TMG.

When TCG overflows (goes from H'FF to H'00), if the timer overflow interrupt enable bit (OVIE) is set to 1 in TMG, bit IRRTG in interrupt request register 2 (IRR2) is set to 1. If in addition bit IENTG in interrupt enable register 2 (IENR2) is set to 1, a CPU interrupt is requested. Details on interrupts are given in 3.3, Interrupts.

TCG cannot be read or written by the CPU.

Upon reset, TCG is initialized to H'00.

Note: * An input capture signal may be generated when TMIG is rewritten.

Input Capture Register GF (ICRGF)

Bit	7	6	5	4	3	2	1	0
	ICRGF7	ICRGF6	ICRGF5	ICRGF4	ICRGF3	ICRGF2	ICRGF1	ICRGF0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRGF is an 8-bit read-only register. When the falling edge of the input capture signal is detected, the TCG value at that time is transferred to ICRGF. If the input capture interrupt select bit (IIEGS) is set to 1 in TMG, bit IRRTG in interrupt request register 2 (IRR2) is set to 1. If in addition bit IENTG in interrupt enable register 2 (IENR2) is set to 1, a CPU interrupt is requested. Details on interrupts are given in 3.3, Interrupts.

To ensure proper input capture when the noise canceller is not used, the pulse width of the input capture signal should be at least 2ø or $2ø_{SUB}$.

Upon reset, ICRGF is initialized to H'00.

Input Capture Register GR (ICRGR)

Bit	7	6	5	4	3	2	1	0
	ICRGR7	ICRGR6	ICRGR5	ICRGR4	ICRGR3	ICRGR2	ICRGR1	ICRGR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRGR is an 8-bit read-only register. When the rising edge of the input capture signal is detected, the TCG value at that time is sent to ICRGR. If the IIEGS bit is cleared to 0 in TMG, bit IRRTG in interrupt request register 2 (IRR2) is set to 1. If in addition bit IENTG in interrupt enable register 2 (IENR2) is set to 1, a CPU interrupt is requested. Details on interrupts are given in 3.3, Interrupts.

To ensure proper input capture when the noise canceller is not used, the pulse width of the input capture signal should be at least 2ø or $2ø_{SUB}$.

Upon reset, ICRGR is initialized to H'00.

Timer Mode Register G (TMG)

Bit	7	6	5	4	3	2	1	0
	OVFH	OVFL	OVIE	IIEGS	CCLR1	CCLR0	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/W*	R/W	R/W	R/W	R/W	R/W	R/W

Note: * Only 0 can be written, to clear flag.

TMG is an 8-bit read/write register. It controls the choice of four internal clocks, counter clear selection, and edge selection for input capture interrupt requests. It also indicates overflow status and enables or disables overflow interrupt requests.

Upon reset, TMG is initialized to H'00.

Bit 7—Timer Overflow Flag H (OVFH): Bit 7 is a status flag indicating that TCG overflowed (from H'FF to H'00) when the input capture signal was high. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 7: OVFH	Description	
0	[Clearing conditions] After reading OVFH = 1, cleared by writing 0 to OVFH	(initial value)
1	[Setting conditions] Set when the value of TCG overflows from H'FF to H'00	

Bit 6—Timer Overflow Flag L (OVFL): Bit 6 is a status flag indicating that TCG overflowed (from H'FF to H'00) when the input capture signal was low, or in interval timer operation. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 6: OVFL	Description	
0	[Clearing conditions] After reading OVFL = 1, cleared by writing 0 to OVFL	(initial value)
1	[Setting conditions] Set when the value of TCG overflows from H'FF to H'00	

Bit 5—Timer Overflow Interrupt Enable (OVIE): Bit 5 enables or disables TCG overflow interrupts.

Bit 5: OVIE	Description	
0	TCG overflow interrupt disabled	(initial value)
1	TCG overflow interrupt enabled	

Bit 4—Input Capture Interrupt Edge Select (IIEGS): Bit 4 selects the input signal edge at which input capture interrupts are requested.

Bit 4: IIEGS	Description
0	Interrupts are requested at the rising edge of the input capture signal (initial value)
1	Interrupts are requested at the falling edge of the input capture signal

Bits 3, 2—Counter Clear 1, 0 (CCLR1, CCLR0): Bits 3 and 2 designate whether TCG is cleared at the rising, falling, or both edges of the input capture signal, or is not cleared.

Bit 3: CCLR1	Bit 2: CCLR0	Description
0	0	TCG is not cleared (initial value)
	1	TCG is cleared at the falling edge of the input capture signal
1	0	TCG is cleared at the rising edge of the input capture signal
	1	TCG is cleared at both edges of the input capture signal

Bits 1, 0—Clock Select (CKS1, CKS0): Bits 1 and 0 select the clock input to TCG from four internal clock signals.

Bit 1: CKS1	Bit 0: CKS0	Description
0	0	Internal clock: ø/64 (initial value)
	1	Internal clock: ø/32
1	0	Internal clock: ø/2
	1	Internal clock: $ø_W$/2

9.4.3 Noise Canceller Circuit

The noise canceller circuit built into the H8/3637 Series is a digital low-pass filter that rejects high-frequency pulse noise in the input at the input capture pin. The noise canceller circuit is enabled by the noise canceller select (NCS)* bit in port mode register 2 (PMR2).

Figure 9.7 shows a block diagram of the noise canceller circuit.

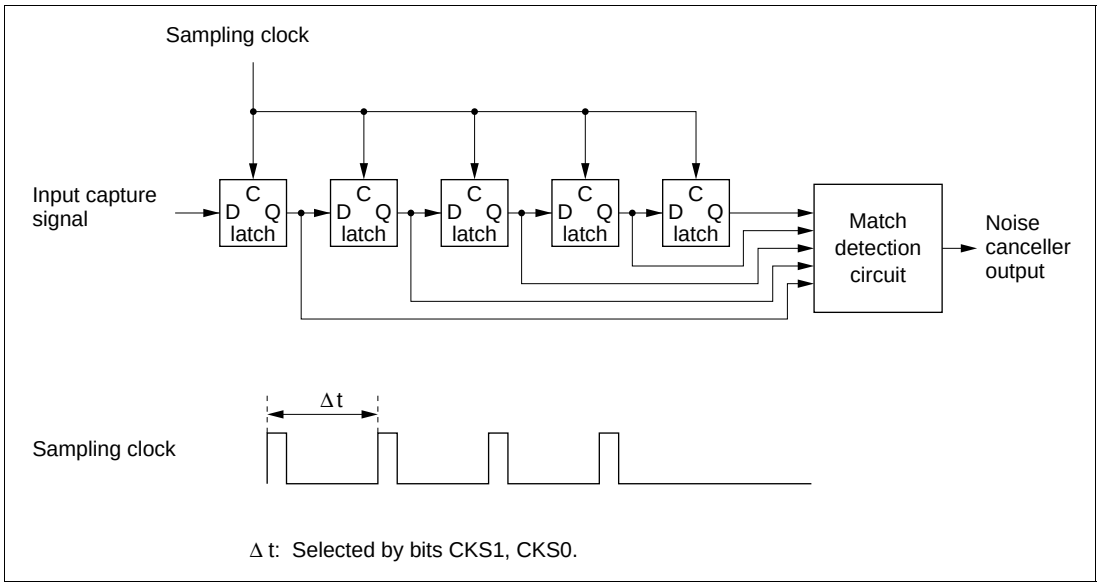


Figure 9.7 Block Diagram of Noise Canceller Circuit

The noise canceller consists of five latch circuits connected in series, and a match detection circuit. When the noise canceller function is disabled (NCS = 0), the system clock is selected as the sampling clock. When the noise canceller is enabled (NCS = 1), the internal clock selected by bits CKS1 and CKS0 in TMG becomes the sampling clock. The input signal is sampled at the rising edge of this clock pulse. Data is considered correct when the outputs of all five latch circuits match. If they do not match, the previous value is retained. Upon reset, the noise canceller output is initialized after the falling edge of the input capture signal has been sampled five times. Accordingly, after the noise canceller function is enabled, pulses that have a pulse width five times greater than the sampling clock will be recognized as input capture signals.

If the noise canceller circuit is not used, the input capture signal pulse width must be at least 2ø or $2ø_{SUB}$ in order to ensure proper input capture operation.

Note: Rewriting the NCS bit may cause an internal input capture signal to be generated.

Figure 9.8 shows a typical timing diagram for the noise canceller circuit. In this example, a high-level input at the input capture pin is rejected as noise because its pulse width is less than five sampling clock ø cycles.

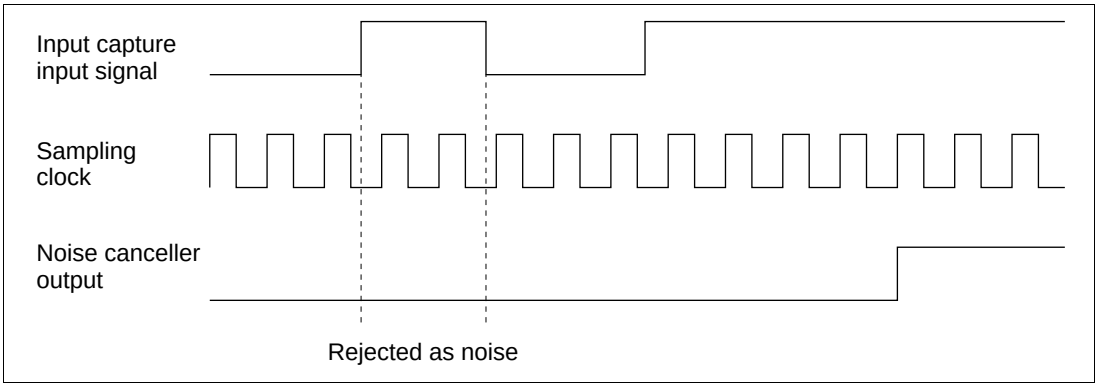


Figure 9.8 Noise Canceller Circuit Timing (Example)

9.4.4 Timer Operation

Timer G Functions: Timer G is an 8-bit up-counter that functions as an input capture timer or an interval timer. These two functions are described below.

- Input capture timer operation

  Timer G functions as an input capture timer when bit TMIG of port mode register 1 (PMR1) is set to 1.*

  At reset, timer mode register G (TMG), timer counter G (TCG), input capture register GF (ICRGF), and input capture register GR (ICRGR) are all initialized to H'00.

  Immediately after reset, TCG begins counting an internal clock with a frequency of ø divided by 64 (ø/64). The clock to be input can be selected by using bits CKS1 and CKS0 in TMG from four internal clock sources.

  At the rising edge/falling edge of the input capture signal input to pin TMIG, the value of TCG is copied into ICRGR/ICRGF. If the input edge is the same as the edge selected by the IIEGS bit of TMG, then bit IRRTG is set to 1 in IRR2. If bit IENTG is also set to 1 in IENR2, a CPU interrupt is requested. For details on interrupts, see 3.3, Interrupts.

  TCG can be cleared to 0 at the rising edge, falling edge, or both edges of the input capture signal as determined with bits CCLR1 and CCLR0 of TMG. If TCG overflows while the input capture signal is high, bit OVFH of TMG is set. If TCG overflows while the input capture signal is low, bit OVFL of TMG is set. When either of these bits is set, if bit OVIE of TMG is currently set to 1, then bit IRRTG is set to 1 in IRR2. If bit IENTG is also set to 1 in IENR2, then timer G requests a CPU interrupt. For further details see 3.3, Interrupts.

Timer G has a noise canceller circuit that rejects high-frequency pulse noise in the input to pin TMIG. See 9.4.3, Noise Canceller Circuit, for details.

Note: * Rewriting the TMIG bit may cause an internal input capture signal to be generated.

- Interval timer operation

  Timer G functions as an interval timer when bit TMIG is cleared to 0 in PMR1. Following a reset, TCG starts counting cycles of the ø/64 internal clock. This is one of four internal clock sources that can be selected by bits CKS1 and CKS0 of TMG. TCG counts up according to the selected clock source. When it overflows from H'FF to H'00, bit OVFL of TMG is set to 1. If bit OVIE of TMG is currently set to 1, then bit IRRTG is set to 1 in IRR2. If bit IENTG is also set to 1 in IENR2, then timer G requests a CPU interrupt. For further details see 3.3, Interrupts.

Count Timing: TCG is incremented by input pulses from an internal clock. TMG bits CKS1 and CKS0 select one of four internal clocks (ø/64, ø/32, ø/2, $ø_W$/2) derived by dividing the system clock (ø) and the watch clock ($ø_W$).

Timing of Internal Input Capture Signals:

- Timing with noise canceller function disabled

  Separate internal input capture signals are generated from the rising and falling edges of the external input signal.

  Figure 9.9 shows the timing of these signals.

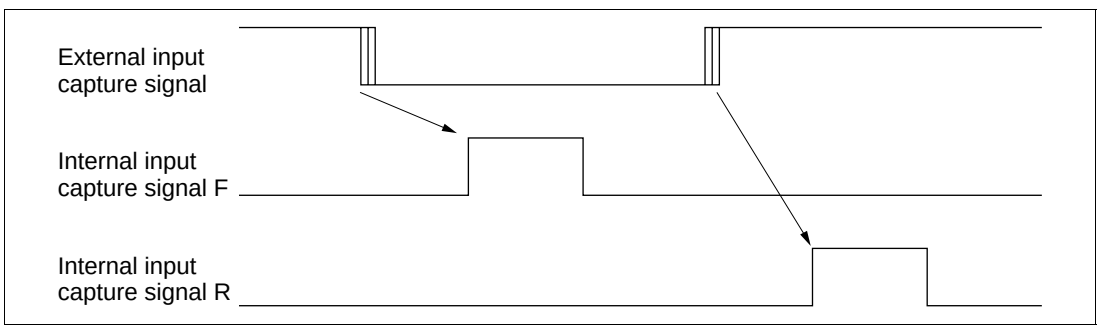


Figure 9.9 Input Capture Signal Timing (Noise Canceller Function Disabled)

- Timing with noise canceller function enabled

  When input capture noise cancelling is enabled, the external input capture signal is routed via the noise canceller circuit, so the internal signals are delayed from the input edge by five sampling clock cycles. Figure 9.10 shows the timing.

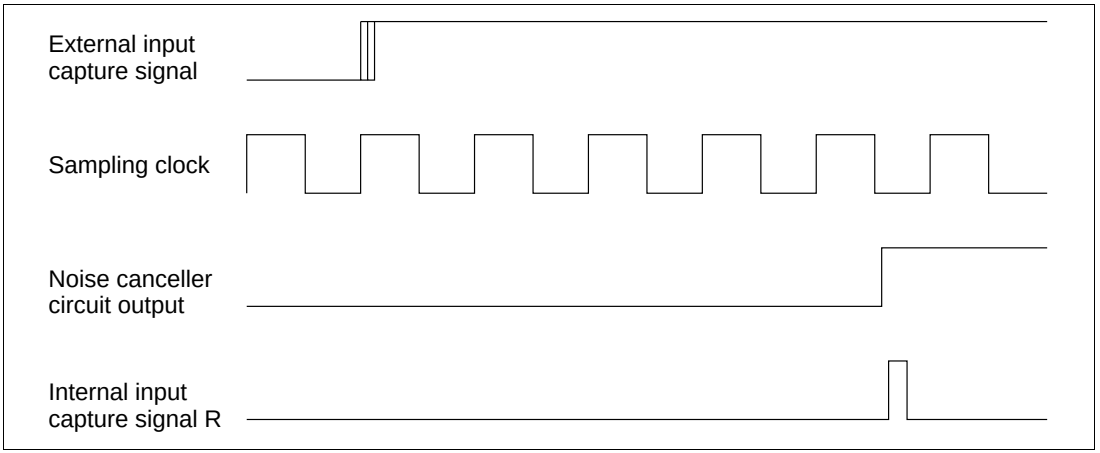


Figure 9.10 Input Capture Signal Timing (Noise Canceller Function Enabled)

Timing of Input Capture: Figure 9.11 shows the input capture timing in relation to the internal input capture signal.

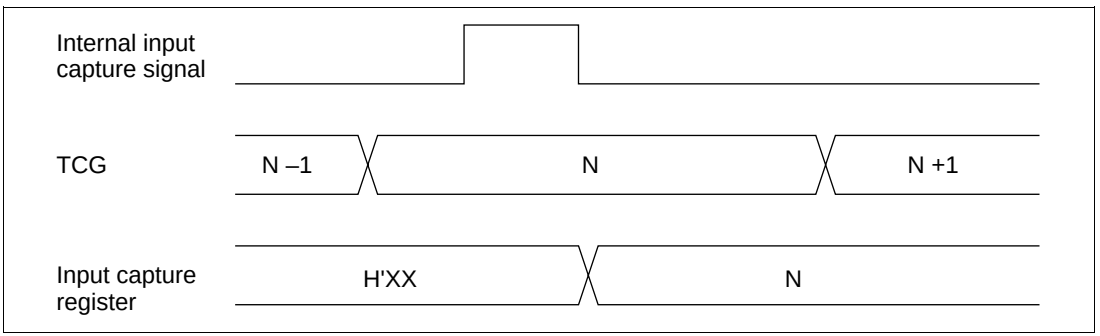


Figure 9.11 Input Capture Timing

TCG Clear Timing: TCG can be cleared at the rising edge, falling edge, or both edges of the external input capture signal. Figure 9.12 shows the timing for clearing at both edges.

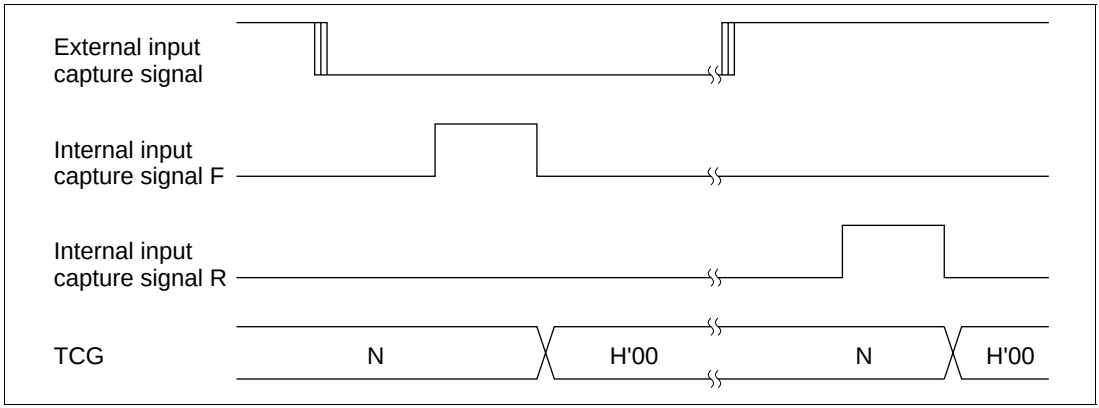


Figure 9.12 TCG Clear Timing

Timer G Operation States: Table 9.10 summarizes the timer G operation states.

Table 9.10 Timer G Operation States

Operation Mode		Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCG	Input capture	Reset	Functions*	Functions*	Halted	Functions/ Halted*	Functions/ Halted*	Halted
	Interval	Reset	Functions*	Functions*	Retained	Functions/ Halted*	Functions/ Halted*	Halted
ICRGF		Reset	Functions*	Functions*	Retained	Functions/ Halted*	Functions/ Halted*	Retained
ICRGR		Reset	Functions*	Functions*	Retained	Functions/ Halted*	Functions/ Halted*	Retained
TMG		Reset	Functions	Retained	Retained	Functions	Retained	Retained

Note: * In active mode and sleep mode, if $ø_w/2$ is selected as the TCG internal clock, since the system clock and internal clock are not synchronized with each other, a synchronization circuit is used. This may result in a count cycle error of up to 1/ø (s). In subactive mode and subsleep mode, if $ø_w/2$ is selected as the TCG internal clock, regardless of the subclock ø/SUB ($ø_w/2$, $ø_w/4$, $ø_w/8$) TCG and the noise canceller circuit run on an internal clock of $ø_w/2$. If any other internal clock is chosen, TCG and the noise canceller circuit will not run, and the input capture function will not operate.

9.4.5 Application Notes

Input Clock Switching and TCG Operation: Depending on when the input clock is switched, there will be cases in which TCG is incremented in the process. Table 9.11 shows the relation between internal clock switchover timing (selected in bits CKS1 and CKS0) and TCG operation. If an internal clock (derived from the system clock ø or subclock $ø_{SUB}$) is used, an increment pulse is generated when a falling edge of the internal clock is detected. For this reason, in a case like No. 3 in table 9.11, where the clock is switched at a time such that the clock signal goes from high level before switching to low level after switching, the switchover is seen as a falling edge to generate the count clock, causing TCG to be incremented.

Table 9.11 Internal Clock Switching and TCG Operation

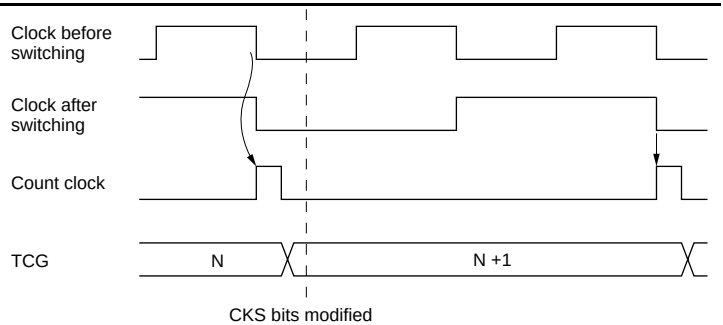
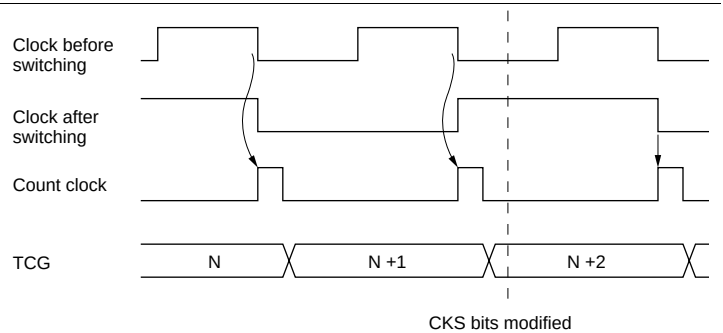
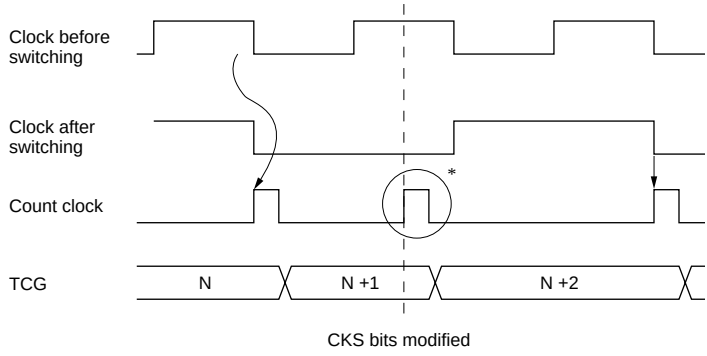
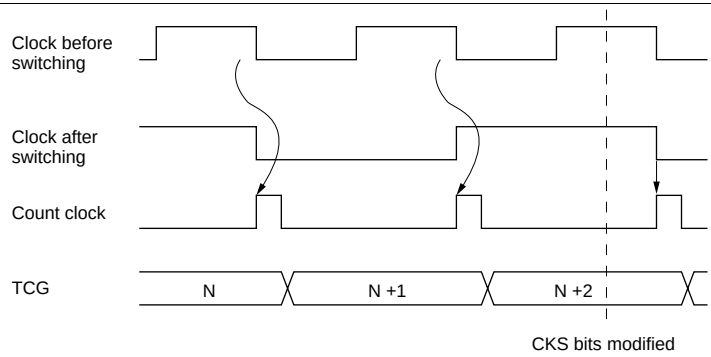
No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	TCG Operation
1	Goes from low level to low level	
2	Goes from low level to high level	

Table 9.11 Internal Clock Switching and TCG Operation (cont)

No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	TCG Operation
3	Goes from high level to low level	
4	Goes from high level to high level	

Note: * The switchover is seen as a falling edge of the clock pulse, and TCG is incremented.

Note on Rewriting Port Mode Registers: When a port mode register setting is modified to enable or disable the input capture function or input capture noise canceling function, note the following points.

- Switching the function of the input capture pin

  When the function of the input capture pin is switched by modifying port mode register 1 (PMR1) bit 3 (the TMIG bit), an input capture edge may be recognized even though no valid signal edge has been input. This occurs under the conditions listed in table 9.12.

Table 9.12 False Input Capture Edges Generating by Switching of Input Capture Pin Function

Input Capture Edge	Conditions
Rising edge recognized	TMIG pin level is high, and TMIG bit is changed from 0 to 1
	TMIG pin level is high and NCS bit is changed from 0 to 1, then TMIG bit is changed from 0 to 1 before noise canceller circuit completes five samples
Falling edge recognized	TMIG pin level is high, and TMIG bit is changed from 1 to 0
	TMIG pin level is low and NCS bit is changed from 0 to 1, then TMIG bit is changed from 0 to 1 before noise canceller circuit completes five samples
	TMIG pin level is high and NCS bit is changed from 0 to 1, then TMIG bit is changed from 1 to 0 before noise canceller circuit completes five samples

Note: When pin $P1_3$ is not used for input capture, the input capture signal input to timer G is low.

- Switching the input capture noise canceling function

  When modifying port mode register 2 (PMR2) bit 4 (the NCS bit) to enable or disable the input capture noise canceling function, first clear the TMIG bit to 0. Otherwise an input capture edge may be recognized even though no valid signal edge has been input. This occurs under the conditions listed in table 9.13.

Table 9.13 False Input Capture Edges Generating by Switching of Noise Canceling Function

Input Capture Edge	Conditions
Rising edge recognized	TMIG bit is set to 1 and TMIG pin level changes from low to high, then NCS bit is changed from 1 to 0 before noise canceller circuit completes five samples
Falling edge recognized	TMIG bit is set to 1 and TMIG pin level changes from high to low, then NCS bit is changed from 1 to 0 before noise canceller circuit completes five samples

If switching of the pin function generates a false input capture edge matching the edge selected by the input capture interrupt edge select bit (IIEGS), the interrupt request flag will be set to 1, making it necessary to clear this flag to 0 before using the interrupt function. Figure 9.13 shows the procedure for modifying port mode register settings and clearing the interrupt request flag. The first step is to mask interrupts before modifying the port mode register. After modifying the port mode register setting, wait long enough for an input capture edge to be recognized (at least two system clocks when noise canceling is disabled; at least five sampling clocks when noise canceling is enabled), then clear the interrupt request flag to 0 (assuming it has been set to 1). An alternative procedure is to avoid having the interrupt request flag set when the pin function is switched, either by controlling the level of the input capture pin so that it does not satisfy the conditions in tables 9.12 and 9.13, or by setting the IIEGS bit of TMG to select the edge opposite to the falsely generated edge.

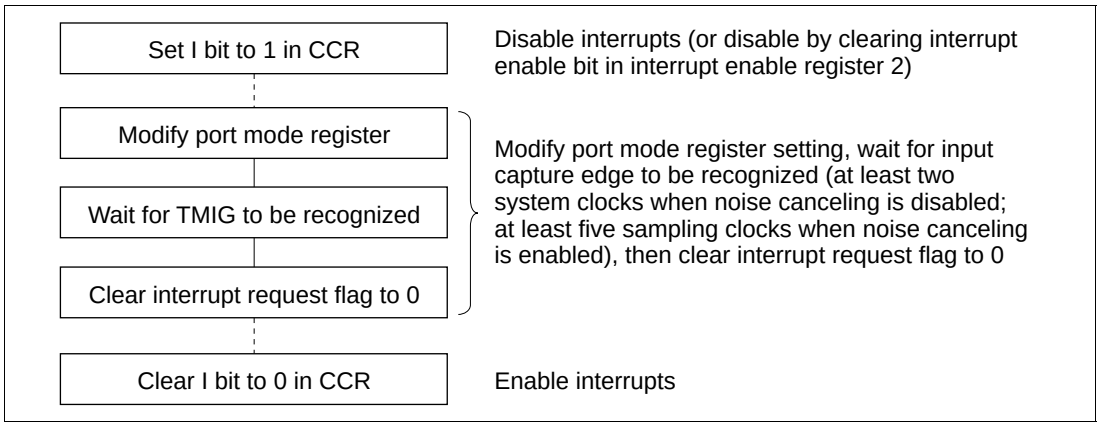


Figure 9.13 Procedure for Modifying Port Mode Register and Clearing Interrupt Request Flag

9.4.6 Sample Timer G Application

The absolute values of the high and low widths of the input capture signal can be measured by using timer G. The CCLR1 and CCLR0 bits of TMG should be set to 1. Figure 9.14 shows an example of this operation.

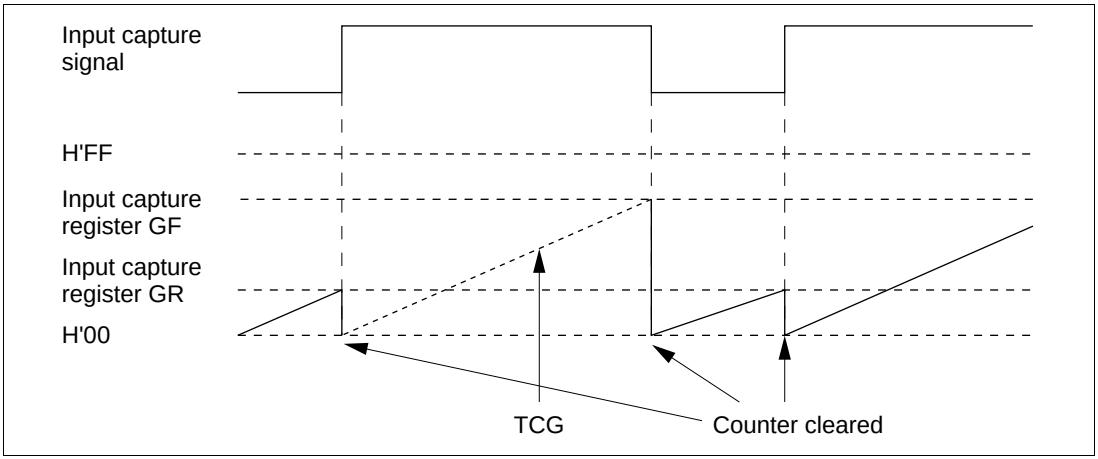


Figure 9.14 Sample Timer G Application

Section 10 Serial Communication Interface

10.1 Overview

The H8/3627 Series is provided with a two-channel serial communication interface (SCI), SCI1 and SCI3. Table 10.1 summarizes the functions and features of the two SCI channels.

Table 10.1 Serial Communication Interface Functions

Channel	Functions	Features
SCI1	Synchronous serial transfer • Choice of 8-bit or 16-bit data length • Continuous clock output	• Choice of 8 internal clocks (ø/1024 to ø/2) or external clock • Open drain output possible • Interrupt requested at completion of transfer
SCI3	Synchronous serial transfer • 8-bit data transfer • Send, receive, or simultaneous send/receive Asynchronous serial transfer • Multiprocessor communication function • Choice of 7-bit or 8-bit data length • Choice of 1-bit or 2-bit stop bit length • Odd or even parity	• Built-in baud rate generator • Receive error detection • Break detection • Interrupt requested at completion of transfer or error

10.2 SCI1

10.2.1 Overview

Serial communication interface 1 (SCI1) performs synchronous serial transfer of 8-bit or 16-bit data.

Features: Features of SCI1 are given below.

- Choice of 8-bit or 16-bit data length
- Choice of eight internal clock sources (ø/1024, ø/256, ø/64, ø/32, ø/16, ø/8, ø/4, ø/2) or an external clock
- Interrupt requested at completion of transfer

Block Diagram: Figure 10.1 shows a block diagram of SCI1.

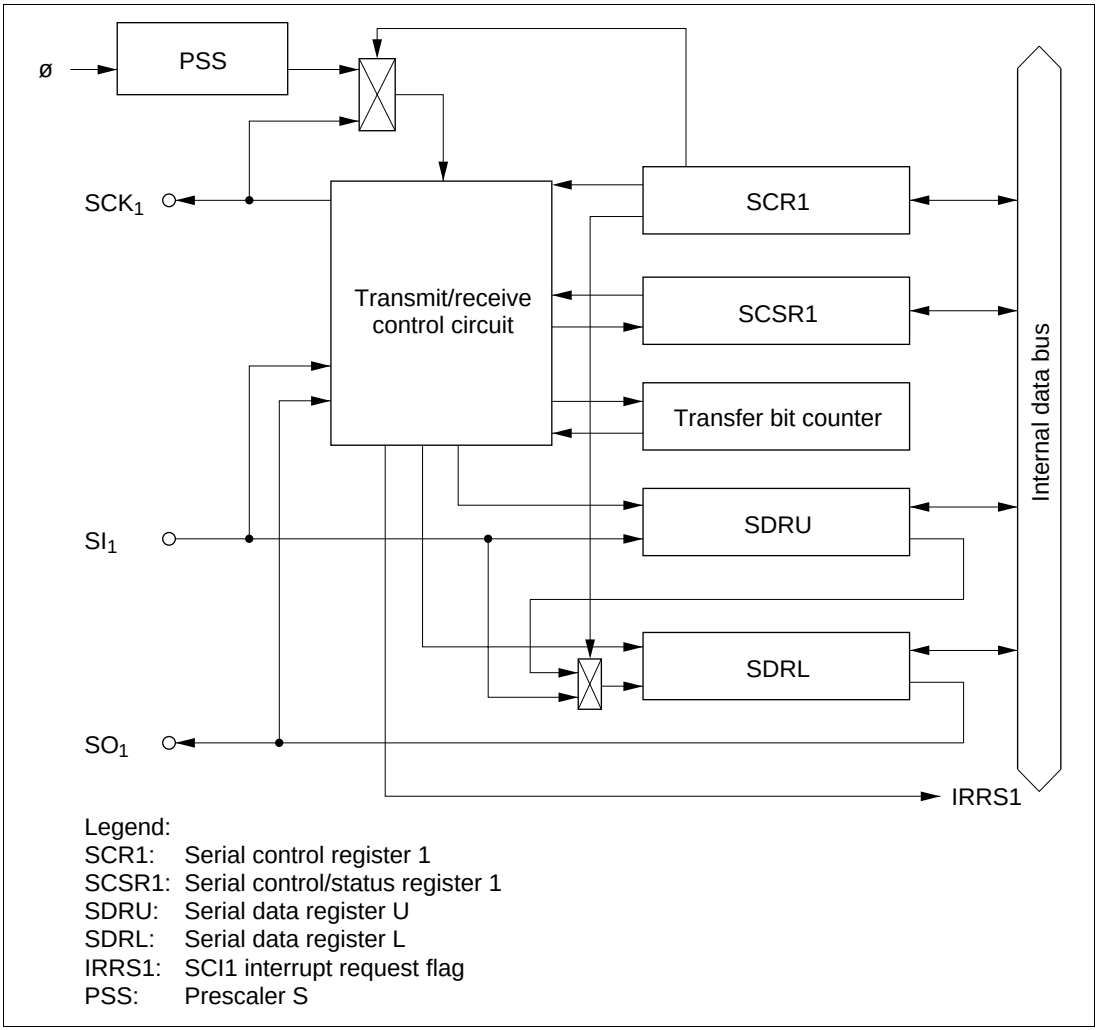


Figure 10.1 SCI1 Block Diagram

Pin Configuration: Table 10.2 shows the SCI1 pin configuration.

Table 10.2 Pin Configuration

Name	Abbrev.	I/O	Function
SCI1 clock pin	SCK_1	I/O	SCI1 clock input or output
SCI1 data input pin	SI_1	Input	SCI1 receive data input
SCI1 data output pin	SO_1	Output	SCI1 transmit data output

Register Configuration: Table 10.3 shows the SCI1 register configuration.

Table 10.3 SCI1 Registers

Name	Abbrev.	R/W	Initial Value	Address
Serial control register 1	SCR1	R/W	H'00	H'FFA0
Serial control status register 1	SCSR1	R/W	H'9C	H'FFA1
Serial data register U	SDRU	R/W	Undefined	H'FFA2
Serial data register L	SDRL	R/W	Undefined	H'FFA3

10.2.2 Register Descriptions

Serial Control Register 1 (SCR1)

Bit	7	6	5	4	3	2	1	0
	SNC1	SNC0	—	—	CKS3	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

SCR1 is an 8-bit read/write register for selecting the operation mode, the transfer clock source, and the prescaler division ratio.

Upon reset, SCR1 is initialized to H'00. Writing to this register stops a transfer in progress.

Bits 7 and 6—Operation Mode Select 1, 0 (SNC1, SNC0): Bits 7 and 6 select the operation mode.

Bit 7: SNC1	Bit 6: SNC0	Description	
0	0	8-bit synchronous transfer mode	(initial value)
	1	16-bit synchronous transfer mode	
1	0	Continuous clock output mode*1	
	1	Reserved*2	

Notes: 1. Pins SI_1 and SO_1 should be used as general input or output ports.
2. Don't set bits SNC1 and SNC0 to 11.

Bits 5 and 4—Reserved Bits: Bits 5 and 4 are reserved: they should always be cleared to 0.

Bit 3—Clock Source Select 3 (CKS3): Bit 3 selects the clock source and sets pin SCK_1 as an input or output pin.

Bit 3: CKS3	Description	
0	Clock source is prescaler S, and pin SCK_1 is output pin	(initial value)
1	Clock source is external clock, and pin SCK_1 is input pin	

Bits 2 to 0—Clock Select 2 to 0 (CKS2 to CKS 0): When CKS3 = 0, bits 2 to 0 select the prescaler division ratio and the serial clock cycle.

				Serial Clock Cycle	
Bit 2: CKS2	**Bit 1: CKS1**	**Bit 0: CKS0**	**Prescaler Division**	**ø = 5 MHz**	**ø = 2.5 MHz**
0	0	0	ø/1024 (initial value)	204.8 µs	409.6 µs
		1	ø/256	51.2 µs	102.4 µs
	1	0	ø/64	12.8 µs	25.6 µs
		1	ø/32	6.4 µs	12.8 µs
1	0	0	ø/16	3.2 µs	6.4 µs
		1	ø/8	1.6 µs	3.2 µs
	1	0	ø/4	0.8 µs	1.6 µs
		1	ø/2	—	0.8 µs

Serial Control/Status Register 1 (SCSR1)

Bit	7	6	5	4	3	2	1	0
	—	SOL	ORER	—	—	—	—	STF
Initial value	1	0	0	1	1	1	0	0
Read/Write	—	R/W	R/(W)*	—	—	—	R	R/W

Note: * Only a write of 0 for flag clearing is possible.

SCSR1 is an 8-bit read/write register indicating operation status and error status.

Upon reset, SCSR1 is initialized to H'9C.

Bit 7—Reserved Bit: Bit 7 is reserved; it is always read as 1, and cannot be modified.

Bit 6—Extended Data Bit (SOL): Bit 6 sets the SO_1 output level. When read, SOL returns the output level at the SO_1 pin. After completion of a transmission, SO_1 continues to output the value of the last bit of transmitted data. The SO_1 output can be changed by writing to SOL before or after a transmission. The SOL bit setting remains valid only until the start of the next transmission. To control the level of the SO_1 pin after transmission ends, it is necessary to write to the SOL bit at the end of each transmission. Do not write to this register while transmission is in progress, because that may cause a malfunction.

Bit 6: SOL	Description		
0	Read	SO_1 pin output level is low	(initial value)
	Write	SO_1 pin output level changes to low	
1	Read	SO_1 pin output level is high	
	Write	SO_1 pin output level changes to high	

Bit 5—Overrun Error Flag (ORER): When an external clock is used, bit 5 indicates the occurrence of an overrun error. If a clock pulse is input after transfer completion, this bit is set to 1 indicating an overrun. If noise occurs during a transfer, causing an extraneous pulse to be superimposed on the normal serial clock, incorrect data may be transferred.

Bit 5: ORER	Description	
0	[Clearing conditions] After reading ORER = 1, cleared by writing 0 to ORER	(initial value)
1	[Setting conditions] Set if a clock pulse is input after transfer is complete, when an external clock is used	

Bits 4 to 2—Reserved Bits: Bits 4 to 2 are reserved; they are always read as 1, and cannot be modified.

Bit 1—Reserved Bit: Bit 1 is reserved; and cannot be modified. This bit will be read as 0 after a reset, but its value is undefined at other times.

Bit 0—Start Flag (STF): Bit 0 controls the start of a transfer. Setting this bit to 1 causes SCI1 to start transferring data.

During the transfer or while waiting for start bit, this bit remains set to 1. It is cleared to 0 upon completion of the transfer. It can therefore be used as a busy flag.

Bit 0: STF	Description		
0	Read	Indicates that transfer is stopped	(initial value)
	Write	Invalid	
1	Read	Indicates transfer in progress	
	Write	Starts a transfer operation	

Serial Data Register U (SDRU)

Bit	7	6	5	4	3	2	1	0
	SDRU7	SDRU6	SDRU5	SDRU4	SDRU3	SDRU2	SDRU1	SDRU0
Initial value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SDRU is an 8-bit read/write register. It is used as the data register for the upper 8 bits in 16-bit transfer (SDRL is used for the lower 8 bits).

Data written to SDRU is output to SDRL starting from the least significant bit (LSB). This data is then replaced by LSB-first data input at pin SI_1, which is shifted in the direction from the most significant bit (MSB) toward the LSB.

SDRU must be written or read only after data transmission or reception is complete. If this register is written or read while a data transfer is in progress, the data contents are not guaranteed.

The SDRU value upon reset is not fixed.

Serial Data Register L (SDRL)

Bit	7	6	5	4	3	2	1	0
	SDRL7	SDRL6	SDRL5	SDRL4	SDRL3	SDRL2	SDRL1	SDRL0
Initial value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SDRL is an 8-bit read/write register. It is used as the data register in 8-bit transfer, and as the data register for the lower 8 bits in 16-bit transfer (SDRU is used for the upper 8 bits).

In 8-bit transfer, data written to SDRL is output from pin SO_1 starting from the least significant bit (LSB). This data is than replaced by LSB-first data input at pin SI_1, which is shifted in the direction from the most significant bit (MSB) toward the LSB.

In 16-bit transfer, operation is the same as for 8-bit transfer, except that input data is fed in via SDRU.

SDRL must be written or read only after data transmission or reception is complete. If this register is read or written while a data transfer is in progress, the data contents are not guaranteed.

The SDRL value upon reset is not fixed.

10.2.3 Operation

Data can be sent and received in an 8-bit or 16-bit format, synchronized to an internal or external clock. Overrun errors can be detected when an external clock is used.

(1) Clock

The serial clock can be selected from a choice of eight internal clocks and an external clock. When an internal clock source is selected, pin SCK_1 becomes the clock output pin. When continuous clock output mode is selected (SCR1 bits SNC1 and SNC0 are set to 10), the clock signal (ø/1024 to ø/2) selected in bits CKS2 to CKS0 is output continuously from pin SCK_1. When an external clock is used, pin SCK_1 is the clock input pin.

(2) Data transfer format

Figure 10.2 shows the data transfer format. Data is sent and received starting from the least significant bit, in LSB-first format. Transmit data is output from one falling edge of the serial clock until the next falling edge. Receive data is latched at the rising edge of the serial clock.

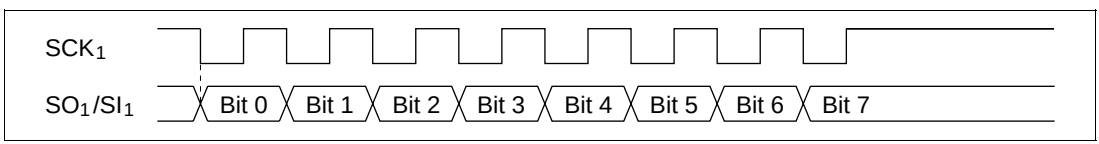


Figure 10.2 Transfer Format

(3) Data transfer Operations

Transmitting: A transmit operation is carried out as follows.

1. Set bits SO1 and SCK1 in PMR2 to 1, selecting the SO_1 and SCK_1 pin functions. If necessary, set bit POF1 in PMR2 for NMOS open drain output at pin SO_1.
2. Clear bit SNC1 in SCR1 to 0, and set bit SNC0 to 1 or 0, designating 8- or 16-bit synchronous transfer mode. Select the serial clock in bits CKS3 to CKS0. Writing data to SCR1 initializes the internal state of SCI1.
3. Write transmit data in SDRL and SDRU, as follows.
   8-bit transfer mode: SDRL
   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL
4. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating and outputs transmit data at pin SO_1.

5. After data transmission is complete, bit IRRS1 in interrupt request register 1 (IRR1) is set to 1.

When an internal clock is used, a serial clock is output from pin SCK_1 in synchronization with the transmit data. After data transmission is complete, the serial clock is not output until the next time the start flag is set to 1. During this time, pin SO_1 continues to output the value of the last bit transmitted.

When an external clock is used, data is transmitted in synchronization with the serial clock input at pin SCK_1. After data transmission is complete, an overrun occurs if the serial clock continues to be input; no data is transmitted and the SCSR1 overrun error flag (bit ORER) is set to 1.

While transmission is stopped, the output value of pin SO_1 can be changed by rewriting bit SOL in SCSR1.

Receiving: A receive operation is carried out as follows.

1. Set bits SI1 and SCK1 in PMR2 to 1, selecting the SI_1 and SCK_1 pin functions.
2. Clear bit SNC1 in SCR1 to 0, and set bit SNC0 to 1 or 0, designating 8- or 16-bit synchronous transfer mode. Select the serial clock in bits CKS3 to CKS0. Writing data to SCR1 initializes the internal state of SCI1.
3. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating and receives data at pin SI_1.
4. After data reception is complete, bit IRRS1 in interrupt request register 1 (IRR1) is set to 1.
5. Read the received data from SDRL and SDRU, as follows.
   8-bit transfer mode: SDRL
   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL
6. After data reception is complete, an overrun occurs if the serial clock continues to be input; no data is received and the SCSR1 overrun error flag (bit ORER) is set to 1.

Simultaneous Transmit/Receive: A simultaneous transmit/receive operation is carried out as follows.

1. Set bits SO1, SI1, and SCK1 in PMR2 to 1, selecting the SO_1, SI_1, and SCK_1 pin functions. If necessary, set bit POF1 in PMR2 for NMOS open drain output at pin SO_1.
2. Clear bit SNC1 in SCR1 to 0, and set bit SNC0 to 1 or 0, designating 8- or 16-bit synchronous transfer mode. Select the serial clock in bits CKS3 to CKS0. Writing data to SCR1 initializes the internal state of SCI1.
3. Write transmit data in SDRL and SDRU, as follows.
   8-bit transfer mode: SDRL
   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL
4. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating. Transmit data is output at pin SO_1. Receive data is input at pin SI_1.
5. After data transmission and reception are complete, bit IRRS1 in IRR1 is set to 1.

6. Read the received data from SDRL and SDRU, as follows.
   8-bit transfer mode: SDRL
   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL

When an internal clock is used, a serial clock is output from pin SCK_1 in synchronization with the transmit data. After data transmission is complete, the serial clock is not output until the next time the start flag is set to 1. During this time, pin SO_1 continues to output the value of the last bit transmitted.

When an external clock is used, data is transmitted and received in synchronization with the serial clock input at pin SCK_1. After data transmission and reception are complete, an overrun occurs if the serial clock continues to be input; no data is transmitted or received and the SCSR1 overrun error flag (bit ORER) is set to 1.

While transmission is stopped, the output value of pin SO_1 can be changed by rewriting bit SOL in SCSR1.

10.2.4 Interrupt Sources

SCI1 can generate an interrupt at the end of a data transfer.

When an SCI1 transfer is complete, bit IRRS1 in interrupt request register 1 (IRR1) is set to 1. SCI1 interrupt requests can be enabled or disabled by bit IENS1 of interrupt enable register 1 (IENR1).

For further details, see 3.3, Interrupts.

10.3 SCI3

10.3.1 Overview

Serial communication interface 3 (SCI3) has both synchronous and asynchronous serial data communication capabilities. It also has a multiprocessor communication function for serial data communication among two or more processors.

Features: SCI3 features are listed below.

- Selection of asynchronous or synchronous mode
  - — Asynchronous mode

    Serial data communication is performed using the asynchronous method, in which synchronization is achieved character by character.

    SCI3 can communicate with a UART (universal asynchronous receiver/transmitter), ACIA (asynchronous communication interface adapter), or other chip that employs standard asynchronous serial communication. It can also communicate with two or more other processors using the multiprocessor communication function. There are twelve selectable serial data communication formats.
    - Data length: seven or eight bits
    - Stop bit length: one or two bits
    - Parity: even, odd, or none
    - Multiprocessor bit: one or none
    - Receive error detection: parity, overrun, and framing errors
    - Break detection: by reading the RXD level directly when a framing error occurs
  - — Synchronous mode

    Serial data communication is synchronized with a clock signal. SCI3 can communicate with other chips having a clocked synchronous communication function.
    - Data length: eight bits
    - Receive error detection: overrun errors

- Full duplex communication

  The transmitting and receiving sections are independent, so SCI3 can transmit and receive simultaneously. Both sections use double buffering, so continuous data transfer is possible in both the transmit and receive directions.

- Built-in baud rate generator with selectable bit rates.

- Internal or external clock may be selected as the transfer clock source.

- There are six interrupt sources: transmit end, transmit data register empty, receive data register full, overrun error, framing error, and parity error.

Block Diagram: Figure 10.3 shows a block diagram of SCI3.

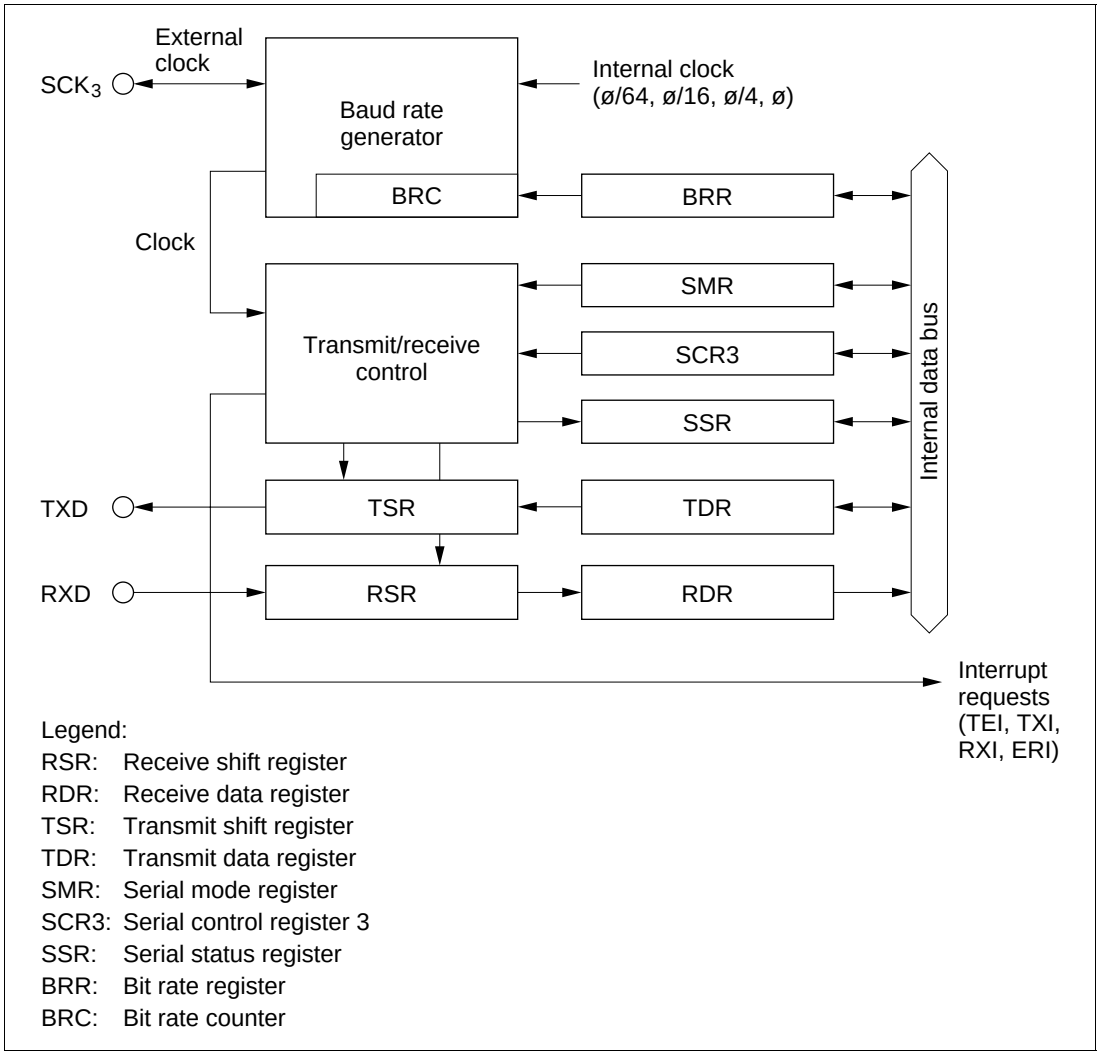


Figure 10.3 SCI3 Block Diagram

Pin Configuration: Table 10.4 shows the SCI3 pin configuration.

Table 10.4 Pin Configuration

Name	Abbrev.	I/O	Function
SCI3 clock	SCK_3	I/O	SCI3 clock input/output
SCI3 receive data input	RXD	Input	SCI3 receive data input
SCI3 transmit data output	TXD	Output	SCI3 transmit data output

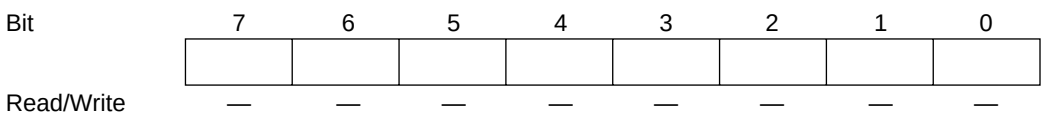
Register Configuration: Table 10.5 shows the SCI3 internal register configuration.

Table 10.5 SCI3 Registers

Name	Abbrev.	R/W	Initial Value	Address
Serial mode register	SMR	R/W	H'00	H'FFA8
Bit rate register	BRR	R/W	H'FF	H'FFA9
Serial control register 3	SCR3	R/W	H'00	H'FFAA
Transmit data register	TDR	R/W	H'FF	H'FFAB
Serial status register	SSR	R/W	H'84	H'FFAC
Receive data register	RDR	R	H'00	H'FFAD
Transmit shift register	TSR	Not possible	—	—
Receive shift register	RSR	Not possible	—	—
Bit rate counter	BRC	Not possible	—	—

10.3.2 Register Descriptions

Receive Shift Register (RSR)



The receive shift register (RSR) is for receiving serial data.

Serial data is input in LSB-first order into RSR from pin RXD, converting it to parallel data. After each byte of data has been received, the byte is automatically transferred to the receive data register (RDR).

RSR cannot be read or written directly by the CPU.

Receive Data Register (RDR)

Bit	7	6	5	4	3	2	1	0
	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

The receive data register (RDR) is an 8-bit register for storing received serial data.

Each time a byte of data is received, the received data is transferred from the receive shift register (RSR) to RDR, completing a receive operation. Thereafter RSR again becomes ready to receive new data. RSR and RDR form a double buffer mechanism that allows data to be received continuously.

RDR is exclusively for receiving data and cannot be written by the CPU.

RDR is initialized to H'00 upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Transmit Shift Register (TSR)

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

The transmit shift register (TSR) is for transmitting serial data.

Transmit data is first transferred from the transmit data register (TDR) to TSR, then is transmitted from pin TXD, starting from the LSB (bit 0).

After one byte of data has been sent, the next byte is automatically transferred from TDR to TSR, and the next transmission begins. If no data has been written to TDR (1 is set in TDRE), there is no data transfer from TDR to TSR.

TSR cannot be read or written directly by the CPU.

Transmit Data Register (TDR)

Bit	7	6	5	4	3	2	1	0
	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The transmit data register (TDR) is an 8-bit register for holding transmit data.

When SCI3 detects that the transmit shift register (TSR) is empty, it shifts transmit data written in TDR to TSR and starts serial data transmission. While TSR is transmitting serial data, the next byte to be transmitted can be written to TDR, realizing continuous transmission.

TDR can be read or written by the CPU at all times.

TDR is initialized to H'FF upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Serial Mode Register (SMR)

Bit	7	6	5	4	3	2	1	0
	COM	CHR	PE	PM	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

The serial mode register (SMR) is an 8-bit register for setting the serial data communication format and for selecting the clock source of the baud rate generator. SMR can be read and written by the CPU at any time.

SMR is initialized to H'00 upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Bit 7—Communication Mode (COM): Bit 7 selects asynchronous mode or synchronous mode as the serial data communication mode.

Bit 7: COM	Description	
0	Asynchronous mode	(initial value)
1	Synchronous mode	

Bit 6—Character Length (CHR): Bit 6 selects either 7 bits or 8 bits as the data length in asynchronous mode. In synchronous mode the data length is always 8 bits regardless of the setting here.

Bit 6: CHR	Description	
0	8-bit data	(initial value)
1	7-bit data*	

Note: * When 7-bit data is selected as the character length in asynchronous mode, the MSB (bit 7) in the transmit data register is not transmitted.

Bit 5—Parity Enable (PE): In asynchronous mode, bit 5 selects whether or not a parity bit is to be added to transmitted data and checked in received data. In synchronous mode there is no adding or checking of parity regardless of the setting here.

Bit 5: PE	Description	
0	Parity bit adding and checking disabled	(initial value)
1	Parity bit adding and checking enabled*	

Note: * When PE is set to 1, then either odd or even parity is added to transmit data, depending on the setting of the parity mode bit (PM). When data is received, it is checked for odd or even parity as designated in bit PM.

Bit 4—Parity Mode (PM): In asynchronous mode, bit 4 selects whether odd or even parity is to be added to transmitted data and checked in received data. The setting here is valid only if parity adding/checking is enabled in bit PE. In synchronous mode, or if parity adding/checking is disabled in asynchronous mode, bit PM is ignored.

Bit 4: PM	Description	
0	Even parity*[1]	(initial value)
1	Odd parity*[2]	

Notes: 1. When even parity is designated, a parity bit is added to the transmitted data so that the sum of 1s in the resulting data is an even number. When data is received, the sum of 1s in the data plus parity bit is checked to see if the result is an even number.

2. When odd parity is designated, a parity bit is added to the transmitted data so that the sum of 1s in the resulting data is an odd number. When data is received, the sum of 1s in the data plus parity bit is checked to see if the result is an odd number.

Bit 3—Stop Bit Length (STOP): Bit 3 selects 1 bit or 2 bits as the stop bit length in asynchronous mode. This setting is valid only in asynchronous mode. In synchronous mode a stop bit is not added, so this bit is ignored.

Bit 3: STOP	Description	
0	1 stop bit*[1]	(initial value)
1	2 stop bits*[2]	

Notes: 1. When data is transmitted, one “1” bit is added at the end of each transmitted character as the stop bit.
2. When data is transmitted, two “1” bits are added at the end of each transmitted character as the stop bits.

When data is received, only the first stop bit is checked regardless of the stop bit length. If the second stop bit value is 1 it is treated as a stop bit; if it is 0, it is treated as the start bit of the next character.

Bit 2—Multiprocessor Mode (MP): Bit 2 enables or disables the multiprocessor communication function. When the multiprocessor communication function is enabled, the parity enable (PE) and parity mode (PM) settings are ignored. The MP bit is valid only in asynchronous mode; it should be cleared to 0 in synchronous mode.

See 10.3.6, for details on the multiprocessor communication function.

Bit 2: MP	Description	
0	Multiprocessor communication function disabled	(initial value)
1	Multiprocessor communication function enabled	

Bits 1 and 0—Clock Select 1, 0 (CKS1, CKS0): Bits 1 and 0 select the clock source for the built-in baud rate generator. A choice of ø/64, ø/16, ø/4, or ø is made in these bits.

See 8, Bit rate register, below for information on the clock source and bit rate register settings, and their relation to the baud rate.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	ø clock	(initial value)
	1	ø/4 clock	
1	0	ø/16 clock	
	1	ø/64 clock	

Serial Control Register 3 (SCR3)

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

Serial control register 3 (SCR3) is an 8-bit register that controls SCI3 transmit and receive operations, enables or disables serial clock output in asynchronous mode, enables or disables interrupts, and selects the serial clock source. SCR3 can be read and written by the CPU at any time.

SCR3 is initialized to H'00 upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Bit 7—Transmit Interrupt Enable (TIE): Bit 7 enables or disables the transmit data empty interrupt request (TXI) when data is transferred from TDR to TSR and the transmit data register empty bit (TDRE) in the serial status register (SSR) is set to 1. The TXI interrupt can be cleared by clearing bit TDRE to 0, or by clearing bit TIE to 0.

Bit 7: TIE	Description	
0	Transmit data empty interrupt request (TXI) disabled	(initial value)
1	Transmit data empty interrupt request (TXI) enabled	

Bit 6—Receive Interrupt Enable (RIE): Bit 6 enables or disables the receive error interrupt request (ERI), and the receive data full interrupt request (RXI) when data is transferred from RSR to RDR and the receive data register full bit (RDRF) in the serial status register (SSR) is set to 1. Receive errors include overrun errors, framing errors, and parity errors. RXI and ERI interrupts can be cleared by clearing SSR flag RDRF, or flags FER, PER, and OER to 0, or by clearing bit RIE to 0.

Bit 6: RIE	Description	
0	Receive data full interrupt request (RXI) and receive error interrupt request (ERI) disabled	(initial value)
1	Receive data full interrupt request (RXI) and receive error interrupt request (ERI) enabled	

Bit 5—Transmit Enable (TE): Bit 5 enables or disables the start of a transmit operation.

Bit 5: TE	Description	
0	Transmit operation disabled*[1] (TXD is the transmit data pin)	(initial value)
1	Transmit operation enabled*[2] (TXD is the transmit data pin)	

Notes: 1. The transmit data register empty bit (TDRE) in the serial status register (SSR) is fixed at 1. Transmit operations are disabled, but the TXD pin functions as the transmit data pin. To use the TXD pin as an I/O pin, clear bit TXD in PMR6 to 0.

2. In this state, writing transmit data in TDR clears bit TDRE in SSR to 0 and starts serial data transmission.

Before setting TE to 1 it is necessary to set the transmit format in SMR.

Bit 4—Receive Enable (RE): Bit 4 enables or disables the start of a receive operation.

Bit 4: RE	Description	
0	Receive operation disabled*[1] (RXD is a general I/O port)	(initial value)
1	Receive operation enabled*[2] (RXD is the receive data pin)	

Notes: 1. When RE is cleared to 0, this has no effect on the SSR flags RDRF, FER, PER, and OER, which retain their states.

2. Serial data receiving begins when, in this state, a start bit is detected in asynchronous mode, or serial clock input is detected in synchronous mode.

Before setting RE to 1 it is necessary to set the receive format in SMR.

Bit 3—Multiprocessor Interrupt Enable (MPIE): Bit 3 enables or disables multiprocessor interrupt requests. This setting is valid only in asynchronous mode, and only when the multiprocessor mode bit (MP) in the serial mode register (SMR) is set to 1. It applies only to data receiving. This bit is ignored when COM is set to 1 or when bit MP is cleared to 0.

Bit 3: MPIE	Description
0	Multiprocessor interrupt request disabled (ordinary receive operation) (initial value) [Clearing condition] Multiprocessor bit receives a data value of 1
1	Multiprocessor interrupt request enabled*

Note: * SCI3 does not transfer receive data from RSR to RDR, does not detect receive errors, and does not set status flags RDRF, FER, and OER in SSR. Until a multiprocessor bit value of 1 is received, the receive data full interrupt (RXI) and receive error interrupt (ERI) are disabled and serial status register (SSR) flags RDRF, FER, and OER are not set. When the multiprocessor bit receives a 1, the MPBR bit of SSR is set to 1, MPIE is automatically cleared to 0, RXI and ERI interrupts are enabled (provided bits TIE and RIE in SCR3 are set to 1), and setting of the RDRF, FER, and OER flags is enabled.

Bit 2—Transmit End Interrupt Enable (TEIE): Bit 2 enables or disables the transmit end interrupt (TEI) requested if there is no valid transmit data in TDR when the MSB is transmitted.

Bit 2: TEIE	Description	
0	Transmit end interrupt (TEI) disabled	(initial value)
1	Transmit end interrupt (TEI) enabled*	

Note: * A TEI interrupt can be cleared by clearing the SSR bit TDRE to 0 and clearing the transmit end bit (TEND) to 0, or by clearing bit TEIE to 0.

Bits 1 and 0—Clock Enable 1, 0 (CKE1, CKE0): Bits 1 and 0 select the clock source and enable or disable clock output at pin SCK_3. The combination of bits CKE1 and CKE0 determines whether pin SCK_3 is a general I/O port, a clock output pin, or a clock input pin.

Note that the CKE0 setting is valid only when operation is in asynchronous mode using an internal clock (CKE1 = 0). This bit is invalid in synchronous mode or when using an external clock (CKE1 = 1). In synchronous mode and in external clock mode, clear CKE0 to 0. After setting bits CKE1 and CKE0, the operation mode must first be set in the serial mode register (SMR).

See table 10.10 in 10.3.3, Operation, for details on clock source selection.

		Description		
Bit 1: CKE1	**Bit 0: CKE0**	**Communication Mode**	**Clock Source**	**SCK_3 Pin Function**
0	0	Asynchronous	Internal clock	I/O port*1
		Synchronous	Internal clock	Serial clock output*1
	1	Asynchronous	Internal clock	Clock output*2
		Synchronous	Reserved	Reserved
1	0	Asynchronous	External clock	Clock input*3
		Synchronous	External clock	Serial clock input
	1	Asynchronous	Reserved	Reserved
		Synchronous	Reserved	Reserved

Notes: 1. Initial value
2. A clock is output with the same frequency as the bit rate.
3. Input a clock with a frequency 16 times the bit rate.

Serial Status Register (SSR)

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	OER	FER	PER	TEND	MPBR	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: * Only a write of 0 for flag clearing is possible.

The serial status register (SSR) is an 8-bit register containing status flags for indicating SCI3 states, and containing the multiprocessor bits.

SSR can be read and written by the CPU at any time, but the CPU cannot write a 1 to the status flags TDRE, RDRF, OER, PER, and FER. To clear these flags to 0 it is first necessary to read a 1. Bit 2 (TEND) and bit 1 (MPBR) are read-only bits and cannot be modified.

SSR is initialized to H'84 upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Bit 7—Transmit Data Register Empty (TDRE): Bit 7 is a status flag indicating that data has been transferred from TDR to TSR.

Bit 7: TDRE	Description
0	Indicates that transmit data written to TDR has not been transferred to TSR [Clearing conditions] • After reading TDRE = 1, cleared by writing 0 to TDRE. • When data is written to TDR by an instruction.
1	Indicates that no transmit data has been written to TDR, or the transmit data written to TDR has been transferred to TSR (initial value) [Setting conditions] • When bit TE in SCR3 is cleared to 0. • When data is transferred from TDR to TSR.

Bit 6—Receive Data Register Full (RDRF): Bit 6 is a status flag indicating whether there is receive data in RDR.

Bit 6: RDRF	Description
0	Indicates there is no receive data in RDR (initial value) [Clearing conditions] • After reading RDRF = 1, cleared by writing 0 to RDRF. • When data is read from RDR by an instruction.
1	Indicates that there is receive data in RDR [Setting condition] When receiving ends normally, with receive data transferred from RSR to RDR

Note: If a receive error is detected during data receiving, or if bit RE in serial control register 3 (SCR3) is cleared to 0, RDR and RDRF are unaffected and keep their previous states. An overrun error (OER) occurs if receiving of data is completed while bit RDRF remains set to 1. If this happens, receive data will be lost.

Bit 5—Overrun Error (OER): Bit 5 is a status flag indicating that an overrun error has occurred during data receiving.

Bit 5: OER	Description
0	Indicates that data receiving is in progress or has been completed*1 (initial value) [Clearing condition] After reading OER = 1, cleared by writing 0 to OER
1	Indicates that an overrun error occurred in data receiving*2 [Setting condition] When data receiving is completed while RDRF is set to 1

Notes: 1. When bit RE in serial control register 3 (SCR3) is cleared to 0, OER is unaffected and keeps its previous state.
2. RDR keeps the data received prior to the overrun; data received after that is lost. While OER is set to 1, data receiving cannot be continued. In synchronous mode, data transmitting cannot be continued either.

Bit 4: Framing Error (FER): Bit 4 is a status flag indicating that a framing error has occurred during asynchronous receiving.

Bit 4: FER	Description
0	Indicates that data receiving is in progress or has been completed*1 (initial value) [Clearing condition] After reading FER = 1, cleared by writing 0 to FER
1	Indicates that a framing error occurred in data receiving [Setting condition] The stop bit at the end of receive data is checked and found to be 0*2

Notes: 1. When bit RE in serial control register 3 (SCR3) is cleared to 0, FER is unaffected and keeps its previous state.
2. When two stop bits are used only the first stop bit is checked, not the second. When a framing error occurs, receive data is transferred to RDR but RDRF is not set. While FER is set to 1, data receiving cannot be continued. In synchronous mode, data transmission and reception cannot be performed if FER is set to 1.

Bit 3—Parity Error (PER): Bit 3 is a status flag indicating that a parity error has occurred during asynchronous receiving.

Bit 3: PER	Description
0	Indicates that data receiving is in progress or has been completed*1 (initial value) [Clearing condition] After reading PER = 1, cleared by writing 0 to PER
1	Indicates that a parity error occurred in data receiving*2 [Setting condition] When the sum of 1s in received data plus the parity bit does not match the parity mode bit (PM) setting in the serial mode register (SMR)

Notes: 1. When bit RE in serial control register 3 (SCR3) is cleared to 0, PER is unaffected and keeps its previous state.
2. When a parity error occurs, receive data is transferred to RDR but RDRF is not set. While PER is set to 1, data receiving cannot be continued. While PER is set to 1 in synchronous mode, data transmission and reception cannot be performed.

Bit 2—Transmit End (TEND): Bit 2 is a status flag indicating that TDRE was set to 1 when the last bit of a transmitted character was sent. TEND is a read-only bit and cannot be modified.

Bit 2: TEND	Description
0	Indicates that transmission is in progress [Clearing conditions] • After reading TDRE = 1, cleared by writing 0 to TDRE. • When data is written to TDR by an instruction.
1	Indicates that a transmission has ended (initial value) [Setting conditions] • When bit TE in SCR3 is cleared to 0. • If TDRE is set to 1 when the last bit of a transmitted character is sent.

Bit 1—Multiprocessor Bit Receive (MPBR): Bit 1 holds the multiprocessor bit in data received in asynchronous mode using a multiprocessor format. MPBR is a read-only bit and cannot be modified.

Bit 1: MPBR	Description
0	Indicates reception of data in which the multiprocessor bit is 0* (initial value)
1	Indicates reception of data in which the multiprocessor bit is 1

Note: * If bit RE in SCR3 is cleared to 0 while a multiprocessor format is in use, MPBR retains its previous state.

Bit 0—Multiprocessor Bit Transmit (MPBT): Bit 0 holds the multiprocessor bit to be added to transmitted data when a multiprocessor format is used in asynchronous mode. Bit MPBT is ignored when synchronous mode is chosen, when the multiprocessor communication function is disabled, or when data transmission is disabled.

Bit 0: MPBT	Description
0	The multiprocessor bit in transmit data is 0 (initial value)
1	The multiprocessor bit in transmit data is 1

Bit Rate Register (BRR)

Bit	7	6	5	4	3	2	1	0
	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The bit rate register (BRR) is an 8-bit register which, together with the baud rate generator clock selected by bits CKS1 and CKS0 in the serial mode register (SMR), sets the transmit/receive bit rate.

BRR can be read or written by the CPU at any time.

BRR is initialized to H'FF upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Table 10.6 gives examples of how BRR is set in asynchronous mode. The values in table 10.6 are for active (high-speed) mode.

Table 10.6 BRR Settings and Bit Rates in Asynchronous Mode

	OSC (MHz)											
	2			2.4576			4			4.194304		
Bit Rate (bits/s)	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	—	—	—	—	—	—

Table 10.6 BRR Settings and Bit Rates in Asynchronous Mode (cont)

	OSC (MHz)											
	4.9152			6			7.3728			8		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	174	–0.26	1	212	+0.03	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 10.6 BRR Settings and Bit Rates in Asynchronous Mode (cont)

	OSC (MHz)					
	9.8304			10		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)
110	2	86	+0.31	2	88	–0.25
150	1	255	0	2	64	+0.16
300	1	127	0	1	129	+0.16
600	0	255	0	1	64	+0.16
1200	0	127	0	0	129	+0.16
2400	0	63	0	0	64	+0.16
4800	0	31	0	0	32	–1.36
9600	0	15	0	0	15	+1.73
19200	0	7	0	0	7	+1.73
31250	0	4	–1.70	0	4	0
38400	0	3	0	0	3	+1.73

Notes: 1. Settings should be made so that error is within 1%.

2. BRR setting values are derived by the following equation.

$$N = \frac{OSC}{64 \times 2^{2n} \times B} \times 10^6 - 1$$

B: Bit rate (bits/s)

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

OSC: Value of $ø_{OSC}$ (MHz)

n: Baud rate generator input clock number (n = 0, 1, 2, 3)
The meaning of n is shown in table 10.7.

Table 10.7 Relation between n and Clock

		SMR Setting	
n	Clock	CKS1	CKS0
0	ø	0	0
1	ø/4	0	1
2	ø16	1	0
3	ø/64	1	1

3. The error values in table 10.6 were derived by performing the following calculation and rounding off to two decimal places.

$$\text{Error (\%)} = \frac{B - R}{R} \times 100$$

B: Bit rate found from n, N, and OSC
R: Bit rate listed in left column of table 10.6

Table 10.8 shows the maximum bit rate for selected frequencies in asynchronous mode. Values in table 10.8 are for active (high-speed) mode.

Table 10.8 Maximum Bit Rate at Selected Frequencies (Asynchronous Mode)

		Setting	
OSC (MHz)	**Maximum Bit Rate (bits/s)**	**n**	**N**
2	31250	0	0
2.4576	38400	0	0
4	62500	0	0
4.194304	65536	0	0
4.9152	76800	0	0
6	93750	0	0
7.3728	115200	0	0
8	125000	0	0
9.8304	153600	0	0
10	156250	0	0

Table 10.9 shows typical BRR settings in synchronous mode. Values in table 10.9 are for active (high-speed) mode.

Table 10.9 Typical BRR Settings and Bit Rates (Synchronous Mode)

Bit Rate (bits/s)	OSC (MHz)							
	2		4		8		10	
	n	N	n	N	n	N	n	N
110	—	—	—	—	—	—	—	—
250	1	249	2	124	2	249	—	—
500	1	124	1	249	2	124	—	—
1K	0	249	1	124	1	249	—	—
2.5K	0	99	0	199	1	99	1	124
5K	0	49	0	99	0	199	0	249
10K	0	24	0	49	0	99	0	124
25K	0	9	0	19	0	39	0	49
50K	0	4	0	9	0	19	0	24
100K	—	—	0	4	0	9	—	—
250K	0	0*	0	1	0	3	0	4
500K			0	0*	0	1	—	—
1M					0	0*	—	—
2.5M								

Blank: Cannot be set

—: Can be set, but error will result

*: Continuous transmit/receive operation is not possible at this setting

Note: BRR setting values are derived by the following equation.

$$N = \frac{OSC}{8 \times 2^{2n} \times B} \times 10^6 - 1$$

B: Bit rate (bits/s)

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

OSC: Value of $ø_{OSC}$ (MHz)

n: Baud rate generator input clock number (n = 0, 1, 2, 3)
The meaning of n is shown in table 10.10.

Table 10.10 Relation between n and Clock

		SMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	ø	0	0
1	ø/4	0	1
2	ø16	1	0
3	ø/64	1	1

10.3.3 Operation

SCI3 supports serial data communication in both asynchronous mode, where each character transferred is synchronized separately, and synchronous mode, where transfer is synchronized by clock pulses.

The choice of asynchronous mode or synchronous mode, and the communication format, is made in the serial mode register (SMR), as shown in table 10.11. The SCI3 clock source is determined by bit COM in SMR and bits CKE1 and CKE0 in serial control register 3 (SCR3), as shown in table 10.12.

Asynchronous Mode:

- Data length: choice of 7 bits or 8 bits
- Choice for the addition of a parity bit, the multiprocessor bit as well as one or two stop bits (these options determine the transmit/receive format and the character length).
- Framing error (FER), parity error (PER), overrun error (OER), and break signal can be detected when data is received.
- Clock source: Choice of internal clocks or an external clock
  - — When an internal clock is selected: Operates on baud rate generator clock. A clock can be output with the same frequency as the bit rate.
  - — When an external clock is selected: A clock input with a frequency 16 times the bit rate is required (internal baud rate generator is not used).

Synchronous Mode:

- Transfer format: 8 bits
- Overrun errors can be detected when data is received.
- Clock source: Choice of internal clocks or an external clock
  - — When an internal clock is selected: Operates on baud rate generator clock, and outputs a serial clock.

— When an external clock is selected: The internal baud rate generator is not used. Operation is synchronous with the input clock.

Table 10.11 SMR Settings and SCI3 Communication Format

SMR Setting						Communication Format			
Bit 7: COM	**Bit 6: CHR**	**Bit 2: MP**	**Bit 5: PE**	**Bit 3: STOP**	**Mode**	**Data Length**	**Multipro-cessor Bit**	**Parity Bit**	**Stop Bit Length**
0	0	0	0	0	Asynchronous mode	8-bit data	None	None	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	1		0	0		7-bit data		None	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	0	1	*	0	Asynchronous mode (multiprocessor format)	8-bit data	Yes	None	1 bit
			*	1					2 bits
	1		*	0		7-bit data			1 bit
			*	1					2 bits
1	*	0	*	*	Synchronous mode	8-bit data	None	None	None

Note: * Don’t care

Table 10.12 SMR and SCR3 Settings and Clock Source Selection

SMR	SCR3			Transmit/Receive Clock	
Bit 7: COM	**Bit 1: CKE1**	**Bit 0: CKE0**	**Mode**	**Clock Source**	**Pin SCK_3 Function**
0	0	0	Asynchronous mode	Internal	I/O port (SCK_3 pin not used)
		1			Outputs clock with same frequency as bit rate
	1	0		External	Clock should be input with frequency 16 times the desired bit rate
1	0	0	Synchronous mode	Internal	Outputs a serial clock
	1	0		External	Inputs a serial clock
0	1	1	Reserved (illegal settings)		
1	0	1			
1	1	1			

Continuous Transmit/Receive Operation Using Interrupts: Continuous transmit and receive operations are possible with SCI3, using the RXI or TXI interrupts. Table 10.13 explains this use of these interrupts.

Table 10.13 Transmit/Receive Interrupts

Interrupt	Flag	Interrupt Conditions	Remarks
RXI	RDRF RIE	When serial data is received normally and receive data is transferred from RSR to RDR, RDRF is set to 1. If RIE is 1 at this time, RXI is enabled and an interrupt occurs. (See figure 10.4 (a).)	The RXI interrupt handling routine reads the receive data from RDR and clears RDRF to 0. Continuous data reception is possible by performing these operations before the reception of the next serial data in RSR is completed.
TXI	TDRE TIE	When TSR empty (previous transmission complete) is detected and the transmit data set in TDR is transferred to TSR, TDRE is set to 1. If TIE is 1 at this time, TXI is enabled and an interrupt occurs. (See figure 10.4 (b).)	The TXI interrupt handling routine writes the next transmit data to TDR and clears TDRE to 0. Continuous data transmission is possible by performing these operations before the transmission of data transferred to TSR is completed.
TEI	TEND TEIE	When the last bit of the TSR transmit character has been sent, if TDRE is 1, then 1 is set in TEND. If TEIE is 1 at this time, TEI is enabled and an interrupt occurs. (See figure 10.4 (c).)	TEI indicates that, when the last bit of the TSR transmit character was sent, the next transmit data had not been written to TDR.

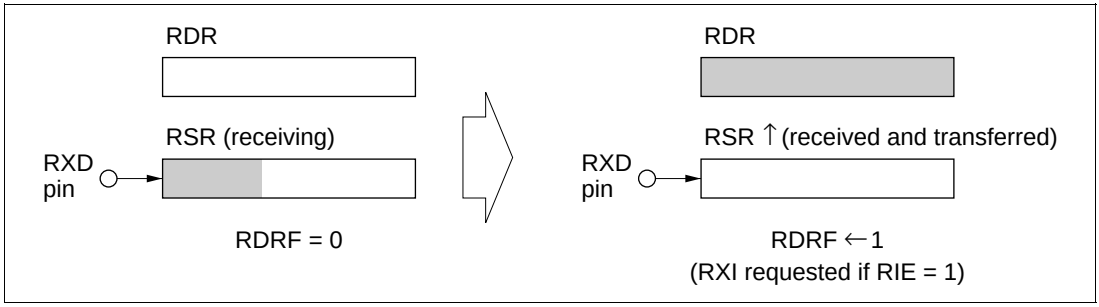


Figure 10.4 (a) RDRF Setting and RXI Interrupt

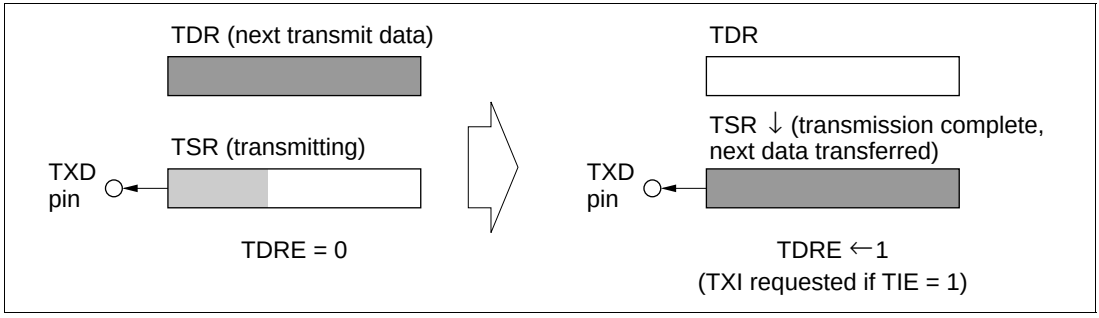


Figure 10.4 (b) TDRE Setting and TXI Interrupt

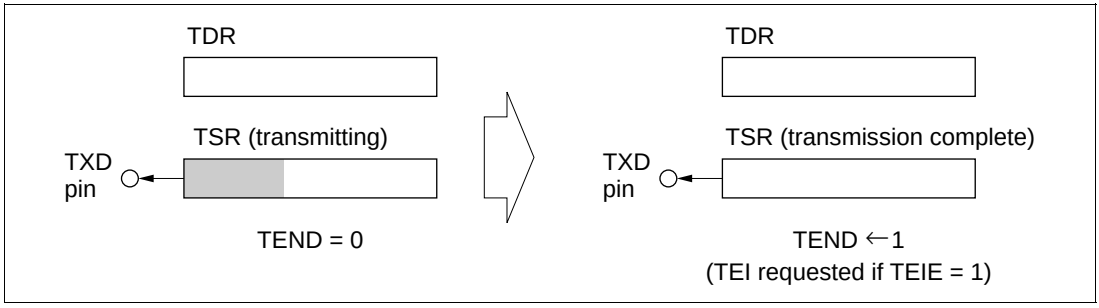


Figure 10.4 (c) TEND Setting and TEI Interrupt

10.3.4 Operation in Asynchronous Mode

In asynchronous communication mode, a start bit indicating the start of communication and a stop bit indicating the end of communication are added to each character that is sent/received. In this way synchronization is achieved for each character as a self-contained unit.

SCI3 consists of independent transmit and receive modules, giving it the capability of full duplex communication. Both the transmit and receive modules have a double-buffer configuration, allowing data to be read or written during communication operations so that data can be transmitted and received continuously.

(1) Transmit/Receive Formats: Figure 10.5 shows the general format for asynchronous serial communication.

The communication line in asynchronous communication mode normally stays at the high level, in the “mark” state. SCI3 monitors the communication line, and begins serial data communication when it detects a “space” (low-level signal), which is regarded as a start bit.

One character consists of a start bit (low level), transmit/receive data (in LSB-first order), a parity bit (high or low level), and finally a stop bit (high level), in this order.

In asynchronous data receiving, synchronization is carried out at the falling edge of the start bit. SCI3 samples data on the 8th pulse of a clock that has 16 times the frequency of one-bit interval, so each bit of data is latched at its center.

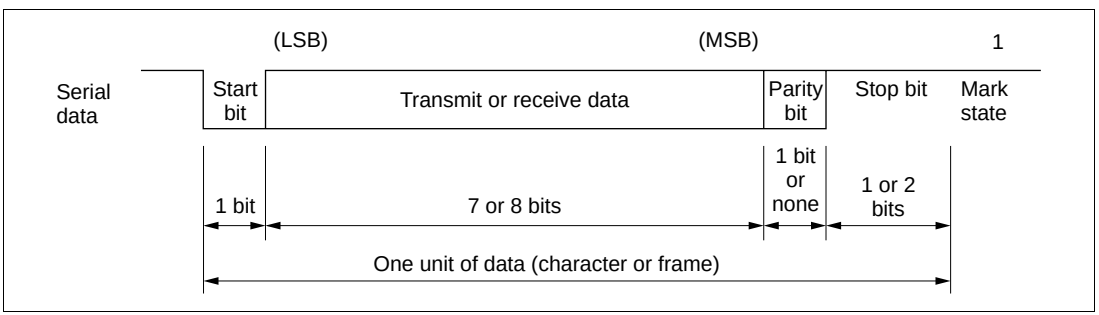


Figure 10.5 Data Format in Asynchronous Serial Communication Mode

Table 10.14 shows the 12 formats that can be selected in asynchronous mode. The format is selected in the serial mode register (SMR).

Table 10.14 Serial Communication Formats in Asynchronous Mode

SMR Settings				Serial Communication Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
0	*	1	0	S	8-bit data								MPB	STOP	
0	*	1	1	S	8-bit data								MPB	STOP	STOP
1	*	1	0	S	7-bit data							MPB	STOP		
1	*	1	1	S	7-bit data							MPB	STOP	STOP	

Legend:
S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

Note: * Don't care

(2) Clock: The clock source is determined by bit COM in SMR and bits CKE1 and CKE0 in serial control register 3 (SCR3). See table 10.12 for the settings. Either an internal clock source can be used to run the built-in baud rate generator, or an external clock source can be input at pin SCK_3.

When an external clock source is input to pin SCK_3, it should have a frequency 16 times the desired bit rate.

When an internal clock source is used, SCK_3 can be used as the clock output pin. The clock output has the same frequency as the serial bit rate, and is synchronized as in figure 10.6 so that the rising edge of the clock occurs in the center of each bit of transmit/receive data.

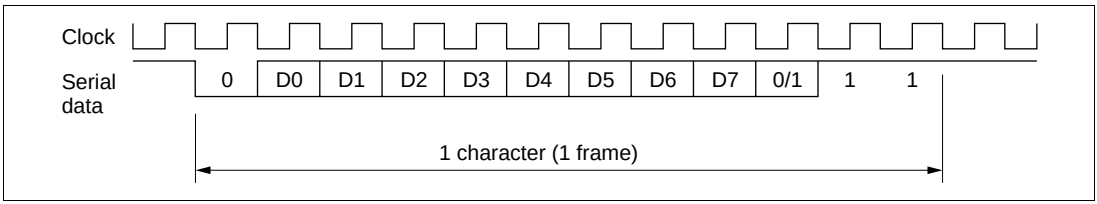


Figure 10.6 Phase Relation of Output Clock and Communication Data in Asynchronous Mode (8-Bit Data, Parity Bit Added, and 2 Stop Bits)

(3) Data Transmit/Receive Operations

SCI3 Initialization: Before data is sent or received, bits TE and RE in serial control register 3 (SCR3) must be cleared to 0, after which initialization can be performed using the procedure.

Note: When modifying the operation mode, transfer format or other settings, always be sure to clear bits TE and RE first. When TE is cleared to 0, bit TDRE will be set to 1. Clearing RE does not clear the status flags RDRF, PER, FER, or OER, or alter the contents of the receive data register (RDR).

When an external clock is used in asynchronous mode, do not stop the clock during operation, including during initialization.

Figure 10.7 shows a typical flow chart for SCI3 initialization.

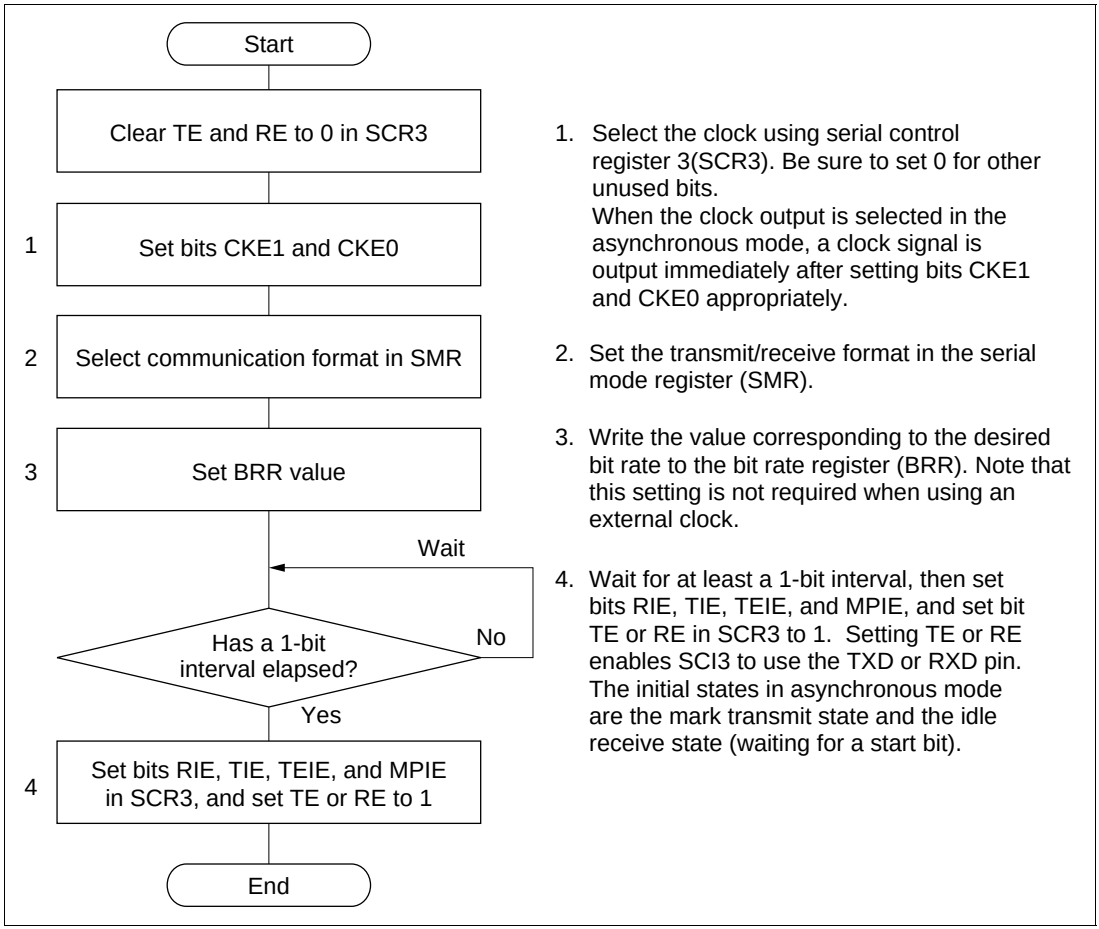


Figure 10.7 Typical Flow Chart when SCI3 Is Initialized

Transmitting: Figure 10.8 shows a typical flow chart for data transmission. After SCI3 initialization, follow the procedure below.

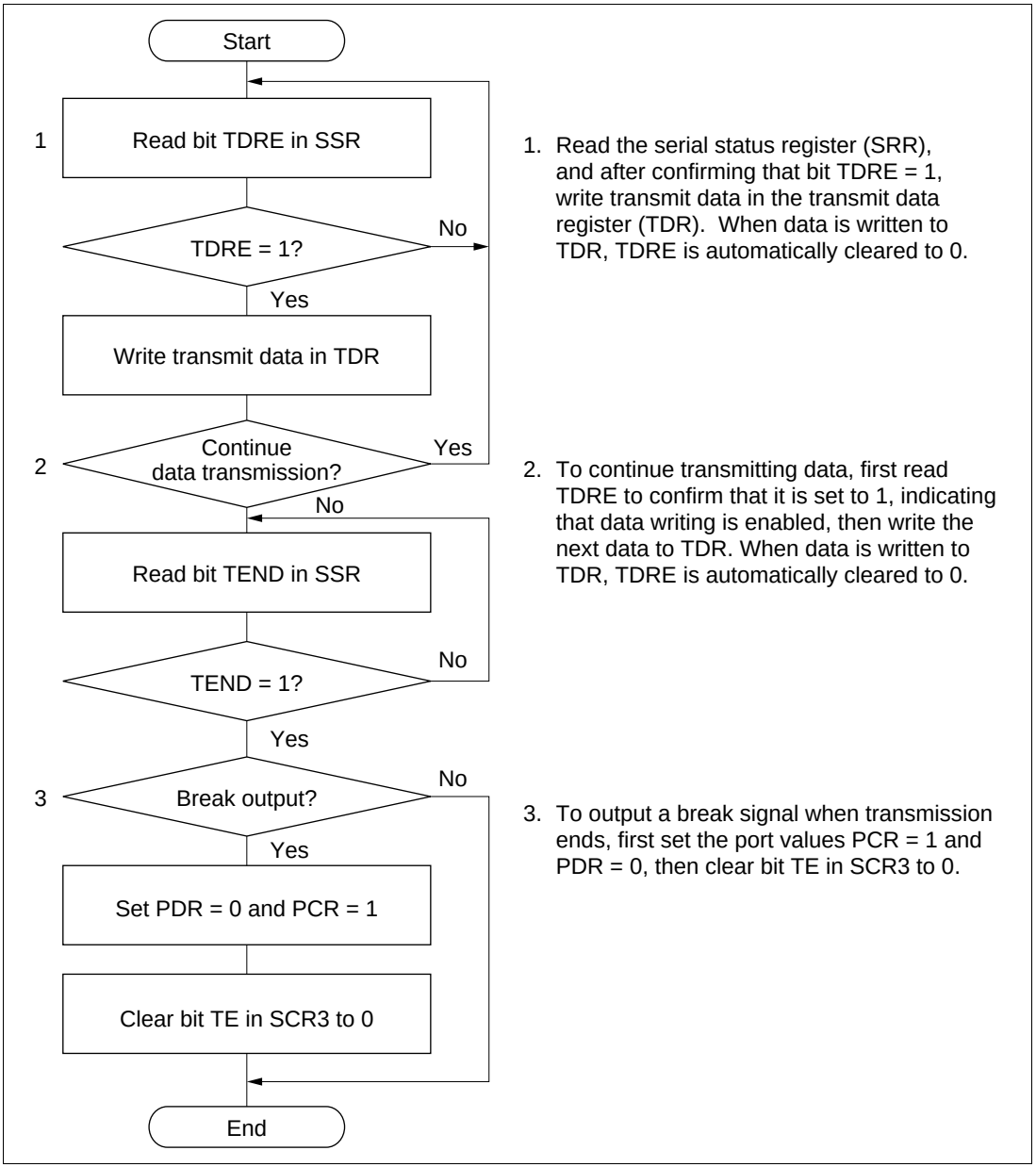


Figure 10.8 Typical Data Transmission Flow Chart (Asynchronous Mode)

SCI3 operates as follows during data transmission in asynchronous mode.

SCI3 monitors bit TDRE in SSR. When this bit is cleared to 0, SCI3 recognizes that there is data written in the transmit data register (TDR), which it transfers to the transmit shift register (TSR). Then TDRE is set to 1 and transmission starts. If bit TIE in SCR3 is set to 1, a TXI interrupt is requested.

Serial data is transmitted from pin TXD using the communication format outlined in table 10.14. After that, it checks TDRE at the same timing which it transmits the stop bit.

If TDRE is 0, data is transferred from TDR to TSR, and after the stop bit is sent, transmission of the next frame starts. If TDRE is 1, the TEND bit in SSR is set to 1, and after the stop bit is sent, the “mark state” is entered, in which 1 is continuously output. A TEI interrupt is requested in this state if bit TEIE in SCR3 is set to 1.

Figure 10.9 shows a typical operation in asynchronous transmission mode.

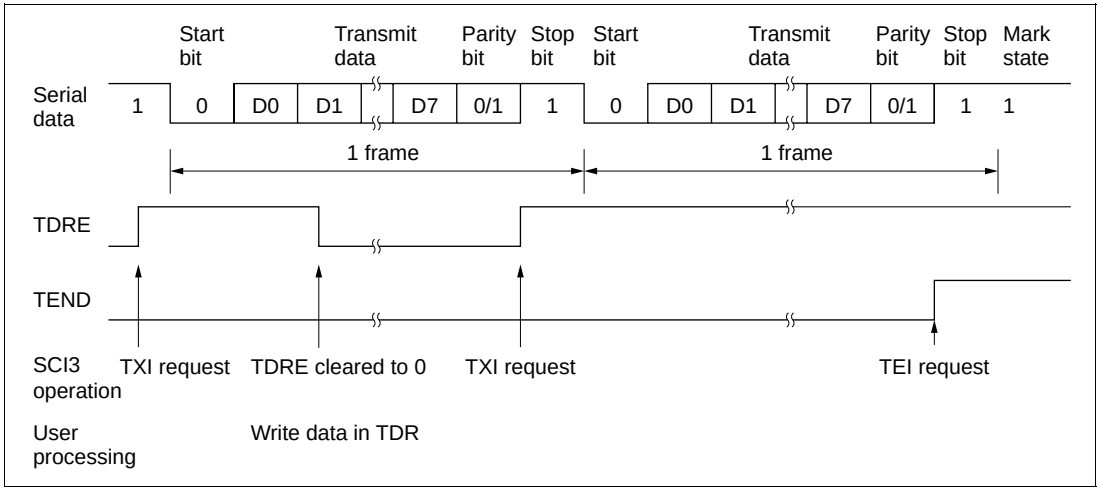
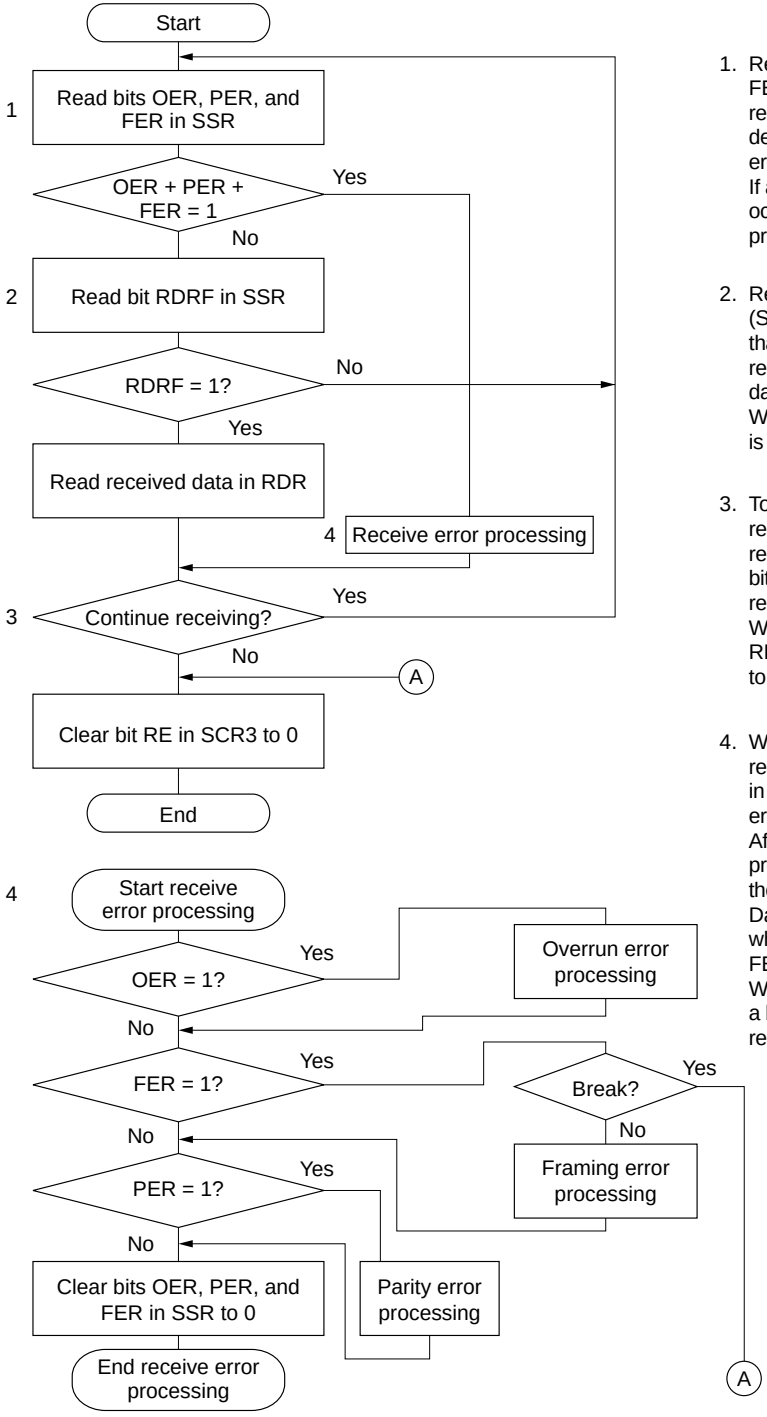


Figure 10.9 Typical Transmit Operation in Asynchronous Mode (8-Bit Data, Parity Bit Added, and 1 Stop Bit)

Receiving: Figure 10.10 shows a typical flow chart for receiving serial data. After SCI3 initialization, follow the procedure below.



1. Read bits OER, PER, and FER in the serial status register (SSR) to determine if a receive error has occurred.
   If a receive error has occurred, receive error processing is executed.

2. Read the serial status register (SSR), and after confirming that bit RDRF = 1, read received data from the receive data register (RDR).
   When RDR data is read, RDRF is automatically cleared to 0.

3. To continue receiving data, read bit RDRF and finish reading RDR before the stop bit of the present frame is received.
   When data is read from RDR, RDRF is automatically cleared to 0.

4. When a receive error occurs, read bits OER, PER, and FER in SSR to determine which error (s) occurred.
   After the necessary error processing, be sure to clear the above bits all to 0.
   Data receiving cannot be resumed while any of bits OER, PER, or FER is set to 1.
   When a framing error occurs, a break can be detected by reading the RXD pin value.

Figure 10.10 Typical Serial Data Receiving Flow Chart in Asynchronous Mode

SCI3 operates as follows when receiving serial data in asynchronous mode.

SCI3 monitors the communication line, and when a start bit (0) is detected it performs internal synchronization and starts receiving. The communication format for data receiving is as outlined in table 10.14. Received data is set in RSR in order of LSB to MSB, then the parity bit and stop bit(s) are received. After receiving the data, SCI3 performs the following checks:

- Parity check: The number of 1s in receive data is checked to see if it matches the odd or even parity selected in bit PM of SMR.
- Stop bit check: The stop bit is checked for a value of 1. If there are two stop bits, only the first bit is checked.
- Status check: The RDRF bit is checked for a value of 0 to make sure received data can be transferred from RSR to RDR.

If no receive error is detected by the above checks, bit RDRF is set to 1 and the received data is stored in RDR. At that time, if bit RIE in SCR3 is set to 1, an RXI interrupt is requested. If the error check detects a receive error, the appropriate error flag (OER, PER, or FER) is set to 1. RDRF retains the same value as before the data was received. If at this time bit RIE in SCR3 is set to 1, an ERI interrupt is requested.

Table 10.15 gives the receive error detection conditions and the processing of received data in each case.

Note: Data receiving cannot be continued while a receive error flag is set. Before continuing the receive operation it is necessary to clear the OER, FER, PER, and RDRF flags to 0.

Table 10.15 Receive Error Conditions and Received Data Processing

Receive Error	Abbrev.	Detection Conditions	Received Data Processing
Overrun error	OER	Receiving of the next data ends while bit RDRF in SSR is still set to 1	Received data is not transferred from RSR to RDR
Framing error	FER	Stop bit is 0	Received data is transferred from RSR to RDR
Parity error	PER	Received data does not match the parity (odd/even) set in SMR	Received data is transferred from RSR to RDR

Figure 10.11 shows a typical SCI3 data receive operation in asynchronous mode.

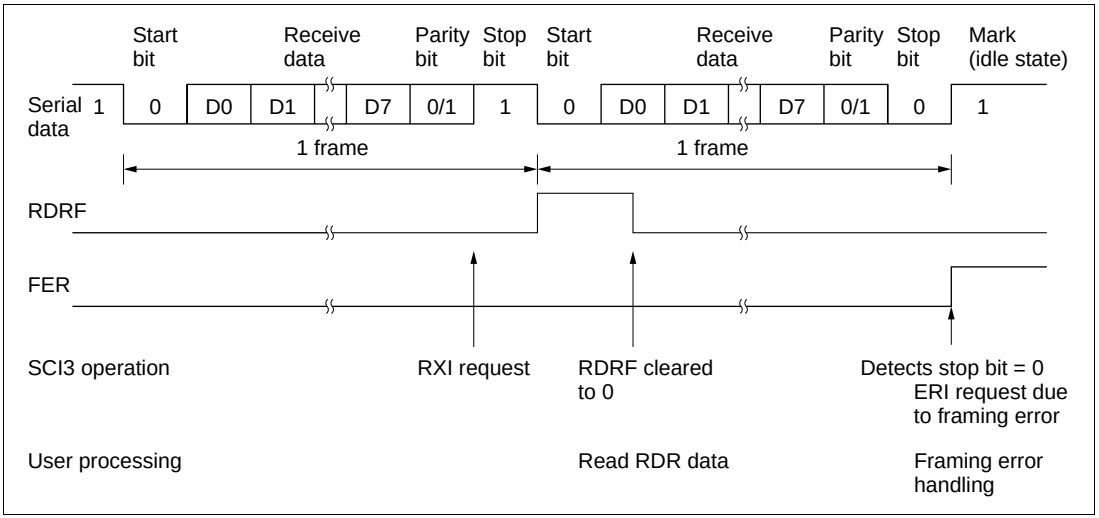


Figure 10.11 Typical Receive Operation in Asynchronous Mode (8-Bit Data, Parity Bit Added, and 1 Stop Bit)

10.3.5 Operation in Synchronous Mode

In synchronous mode, data is sent or received in synchronization with clock pulses. This mode is suited to high-speed serial communication.

SCI3 consists of independent transmit and receive modules, so full duplex communication is possible, sharing the same clock between both modules. Both the transmit and receive modules have a double-buffer configuration. This allows data to be written during a transmit operation so that data can be transmitted continuously, and enables data to be read during a receive operation so that data can be received continuously.

(1) Transmit/Receive Format: Figure 10.12 shows the general communication data format for synchronous communication.

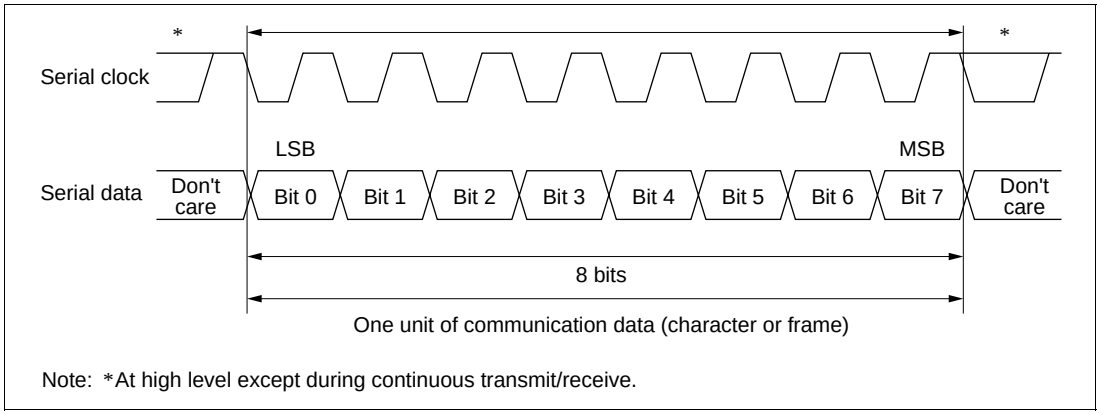


Figure 10.12 Data Format in Synchronous Communication Mode

In synchronous communication, data is output to the communication line during the period from one falling edge of the synchronous clock to the next falling edge. Data fixing is guaranteed at the rising edge of the synchronous clock.

One character of data starts from the LSB and ends with the MSB. The communication line retains the MSB state after the MSB is output.

In synchronous receive mode, SCI3 latches receive data in synchronization with the rising edge of the serial clock.

The transmit/receive format is fixed at 8-bit data. No parity bit or multiprocessor bit is added in this mode.

(2) Clock: Either an internal clock from the built-in baud rate generator is used, or an external clock is input at pin SCK_3. The choice of clock sources is designated by bit COM in SMR and bits CKE1 and CKE0 in serial control register 3 (SCR3). See table 10.12 for details on selecting the clock source.

When operation is based on an internal clock, a serial clock is output at pin SCK_3. Eight clock pulses are output per character of transmit/receive data. When no transmit or receive operation is being performed, the pin is held at the high level.

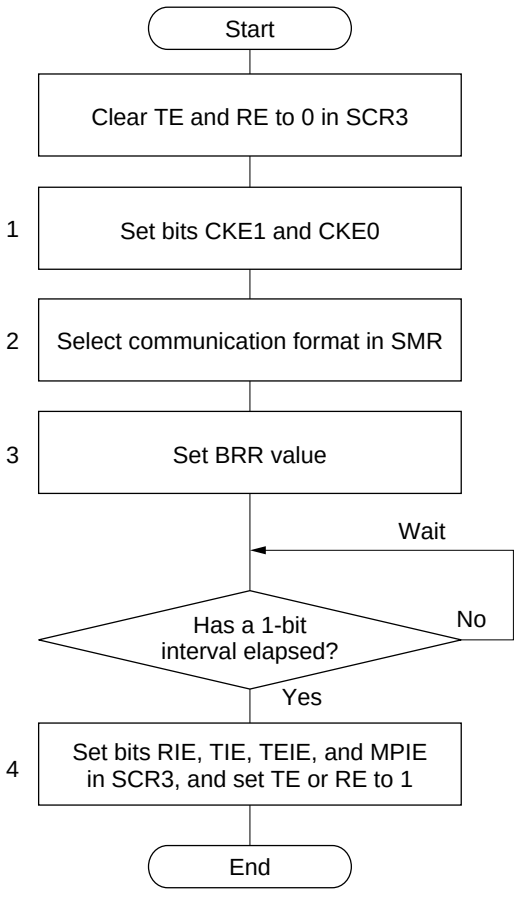
(3) Data Transmit/Receive Operations

SCI3 Initialization: Before data is sent or received, bits TE and RE in serial control register 3 (SCR3) must be cleared to 0, after which initialization can be performed using the procedure.

Note: When modifying the operation mode, transfer format or other settings, always be sure to clear bits TE and RE first. When TE is cleared to 0, bit TDRE will be set to 1. Clearing RE does not clear the status flags RDRF, PER, FER, or OER, or alter the contents of the receive data register (RDR).

When an external clock is used in synchronous mode, do not supply the clock during initialization.

Figure 10.13 shows a typical flow chart for SCI3 initialization.



1. Select the clock using serial control register 3(SCR3). Be sure to set 0 for other unused bits.
   When the clock output is selected in the asynchronous mode, a clock signal is output immediately after setting bits CKE1 and CKE0 appropriately. When the clock output is selected in the synchronous mode, a clock signal is output immediately after setting bits CKE1 and CKE0 appropriately and setting bit RE to 1.
2. Set the transmit/receive format in the serial mode register (SMR).
3. Write the value corresponding to the desired bit rate to the bit rate register (BRR). Note that this setting is not required when using an external clock.
4. Wait for at least a 1-bit interval, then set bits RIE, TIE, TEIE, and MPIE, and set bit TE or RE in SCR3 to 1. Setting TE or RE enables SCI3 to use the TXD or RXD pin. The initial states in asynchronous mode are the mark transmit state and the idle receive state (waiting for a start bit).

Note: In simultaneous transmit/receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

Figure 10.13 Typical Flow Chart when SCI3 Is Initialized

Transmitting: Figure 10.14 shows a typical flow chart for data transmission. After SCI3 initialization, follow the procedure below.

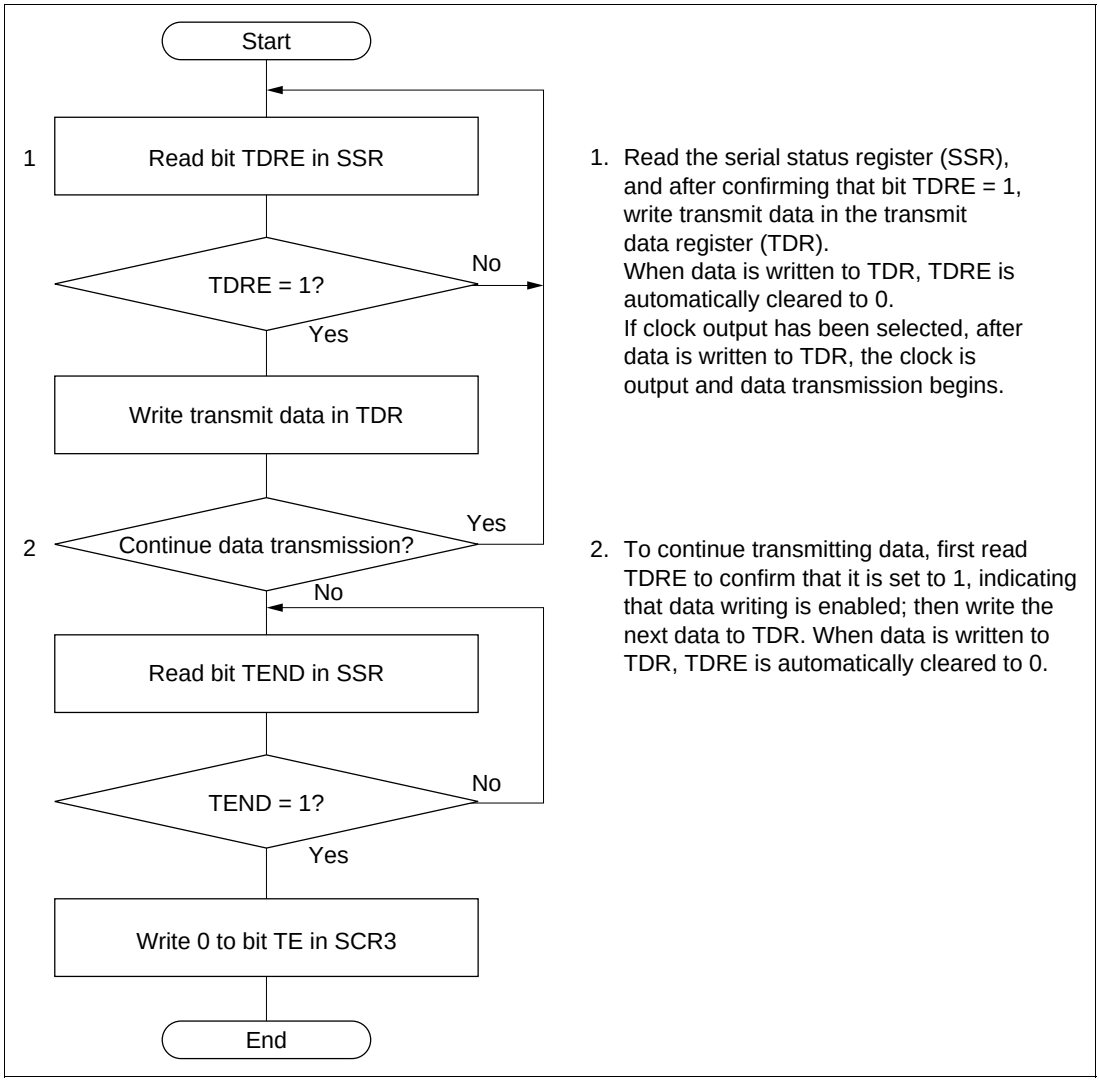


Figure 10.14 Typical Data Transmission Flow Chart in Synchronous Mode

SCI3 operates as follows during data transmission in synchronous mode.

SCI3 monitors bit TDRE in SSR. When this bit is cleared to 0, SCI3 recognizes that there is data written in the transmit data register (TDR), which it transfers to the transmit shift register (TSR). Then TDRE is set to 1 and transmission starts. If bit TIE in SCR3 is set to 1, a TXI interrupt is requested.

If clock output is selected, SCI3 outputs eight serial clock pulses. If an external clock is used, data is output in synchronization with the clock input.

Serial data is transmitted from pin TXD in order from LSB (bit 0) to MSB (bit 7).

After that, it checks TDRE at the same timing which it transmits the MSB (bit 7). If TDRE is 0, data is transferred from TDR to TSR, and after the MSB (bit 7) is sent, transmission of the next frame starts. If TDRE is 1, the TEND bit in SSR is set to 1, and after the MSB (bit 7) is sent, the MSB state is maintained. A TEI interrupt is requested in this state if bit TEIE in SCR3 is set to 1.

After data transmission ends, pin SCK_3 is held at the high level.

Note: Data transmission cannot take place while any of the receive error flags (OER, FER, PER) is set to 1. Be sure to confirm that these error flags are cleared to 0 before starting transmission.

Figure 10.15 shows a typical SCI3 transmit operation in synchronous mode.

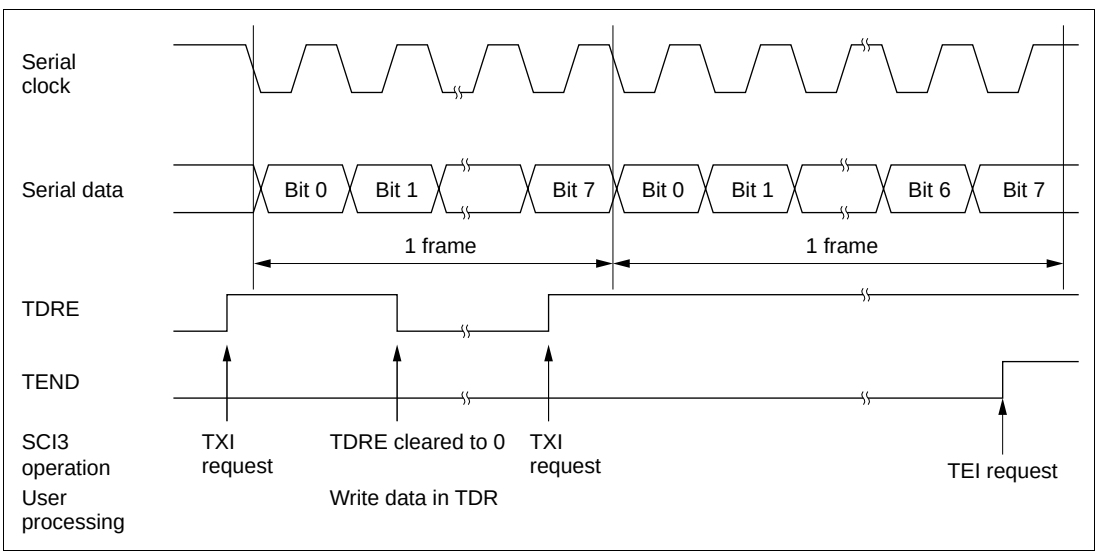


Figure 10.15 Typical SCI3 Transmit Operation in Synchronous Mode

Receiving: Figure 10.16 shows a typical flow chart for receiving data. After SCI3 initialization, follow the procedure below.

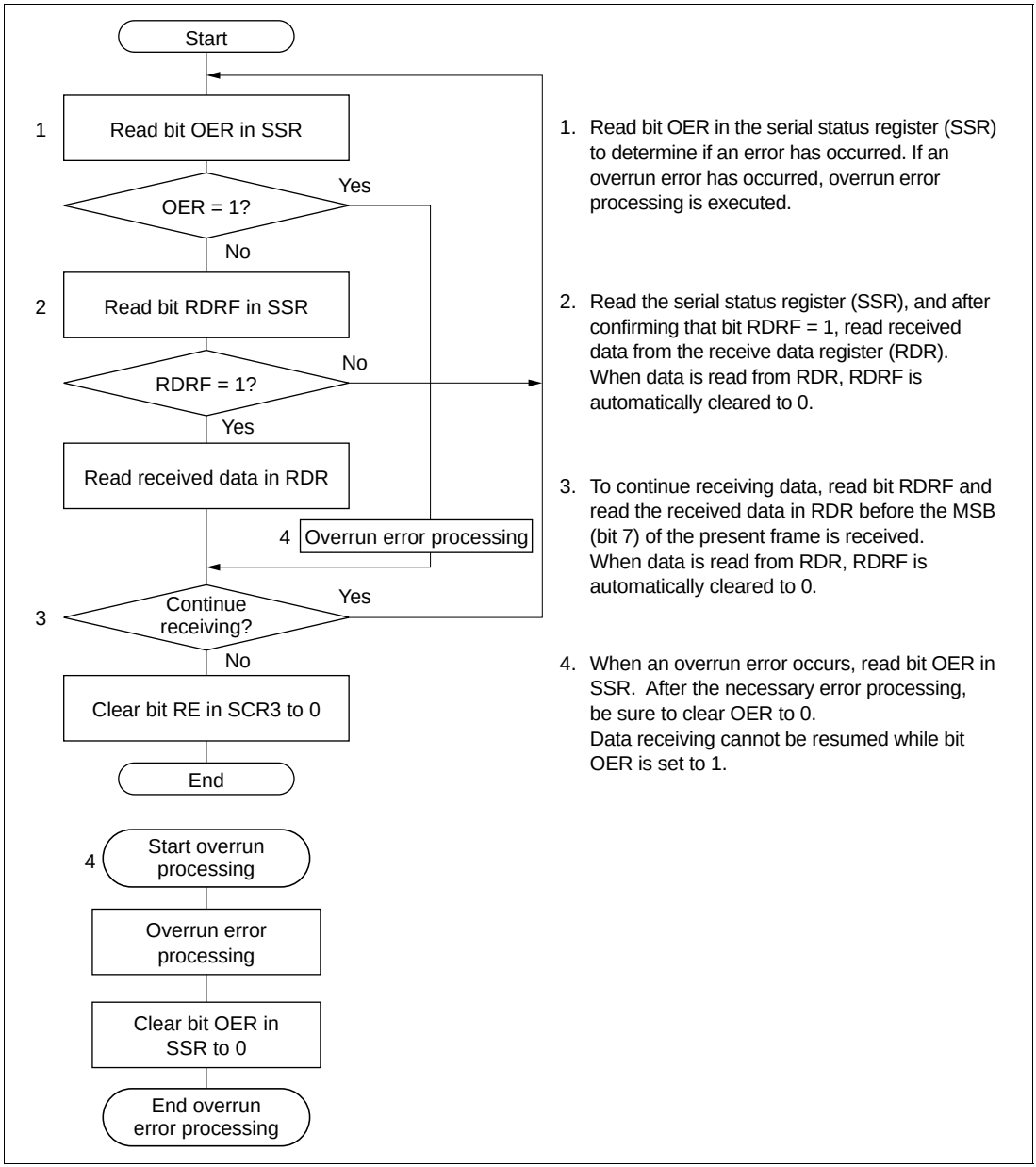


Figure 10.16 Typical Data Receiving Flow Chart in Synchronous Mode

SCI3 operates as follows when receiving serial data in synchronous mode.

In synchronization with the input or output of the serial clock, SCI3 initializes internally and starts receiving. Received data is set in RSR from LSB to MSB.

After data has been received, SCI3 checks to confirm that the value of bit RDRF is 0 indicating that received data can be transferred from RSR to RDR. If this check passes, RDRF is set to 1 and the received data is stored in RDR. At this time, if bit RIE in SCR3 is set to 1, an RXI interrupt is requested. If an overrun error is detected, OER is set to 1 and RDRF remains set to 1. Then if bit RIE in SCR3 is set to 1, an ERI interrupt is requested.

For the overrun error detection conditions and receive data processing, see table 10.15.

Note: Data receiving cannot be continued while a receive error flag is set. Before continuing the receive operation it is necessary to clear the OER, FER, PER, and RDRF flags to 0.

Figure 10.17 shows a typical receive operation in synchronous mode.

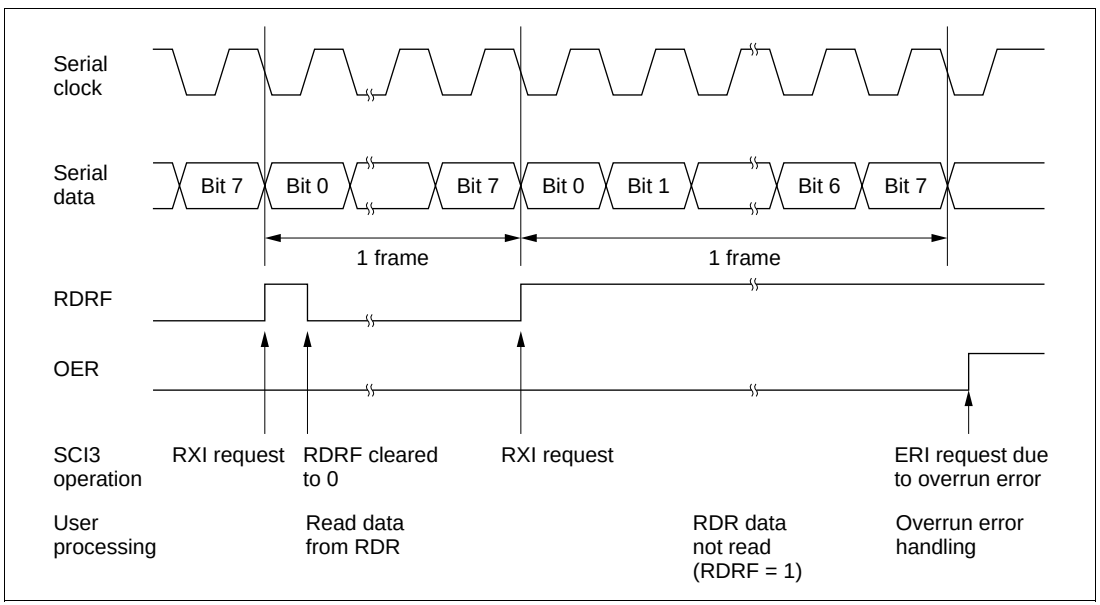
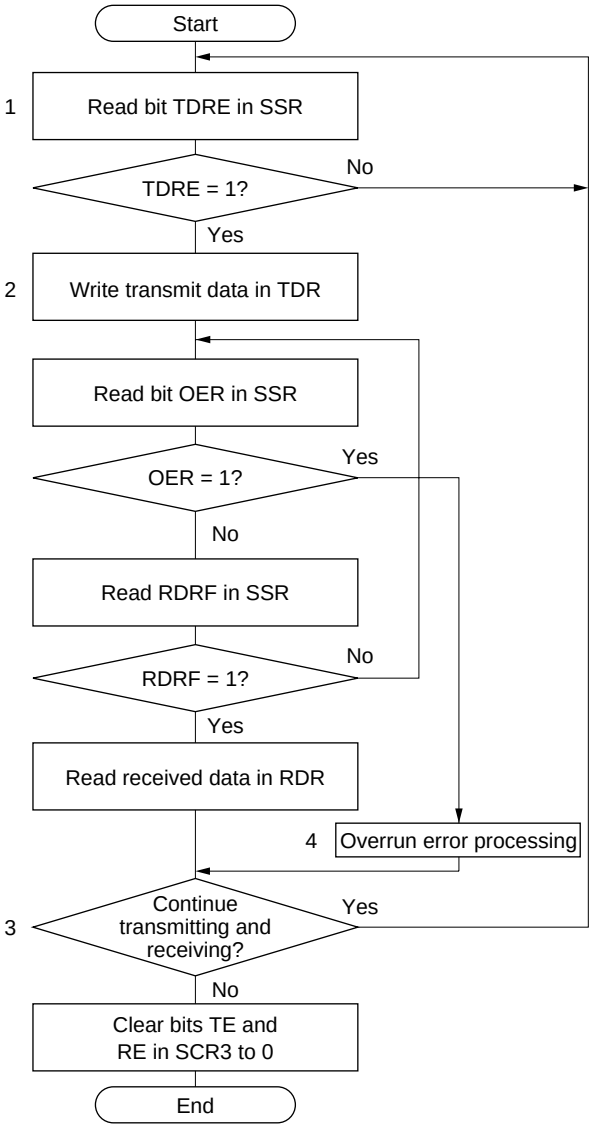


Figure 10.17 Typical Receive Operation in Synchronous Mode

Simultaneous Transmit/Receive: Figure 10.18 shows a typical flow chart for transmitting and receiving simultaneously. After SCI3 synchronization, follow the procedure below.



1. Read the serial status register (SSR), and after confirming that bit TDRE = 1, write transmit data in the transmit data register (TDR). When data is written to TDR, TDRE is automatically cleared to 0.
2. Read the serial status register (SSR), and after confirming that bit RDRF = 1, read the received data from the receive data register (RDR). When data is read from RDR, RDRF is automatically cleared to 0.
3. To continue transmitting and receiving serial data, read bit RDRF and finish reading RDR before the MSB (bit 7) of the present frame is received. Also read bit TDRE, check that it is set to 1, and write the next data in TDR before the MSB (bit 7) of the current frame has been transmitted. When data is written to TDR, TDRE is automatically cleared to 0; and when data is read from RDR, RDRF is automatically cleared to 0.
4. When an overrun error occurs, read bit OER in SSR. After the necessary error processing, be sure to clear OER to 0. Data transmission and reception cannot take place while bit OER is set to 1. See figure 10.16 for overrun error processing.

Note: When switching from transmit or receive operation to simultaneous transmit/ receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

Figure 10.18 Simultaneous Transmit/Receive Flow Chart in Synchronous Mode

Notes: 1. To switch from transmitting to simultaneous transmitting and receiving, first confirm that TDRE and TEND are both set to 1 and that SCI3 has finished transmitting. Next clear TE to 0. Then set both TE and RE to 1.

2. To switch from receiving to simultaneous transmitting and receiving, after confirming that SCI3 has finished receiving, clear RE to 0. Next, after confirming that RDRF and the error flags (OER FER, PER) are all 0, set both TE and RE to 1.

10.3.6 Multiprocessor Communication Function

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 code. A serial communication cycle consists of two cycles: an ID-sending cycle that identifies the receiving processor, and a data-sending cycle in which communication data is sent to the specified receiving processor. The multiprocessor bit is used to distinguish between the ID-sending cycle and the data-sending cycle. The multiprocessor bit is 1 in an ID-sending cycle, and 0 in a data-sending cycle.

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. When a receiving processor receives data with the multiprocessor bit set to 1, it compares the data with its own ID. If the data matches its ID, the receiving processor continues to receive incoming data. If the data does not match its ID, the receiving processor skips further incoming data until it again receives data with the multiprocessor bit set to 1. Multiple processors can send and receive data in this way.

Figure 10.19 shows an example of communication among different processors using a multiprocessor format.

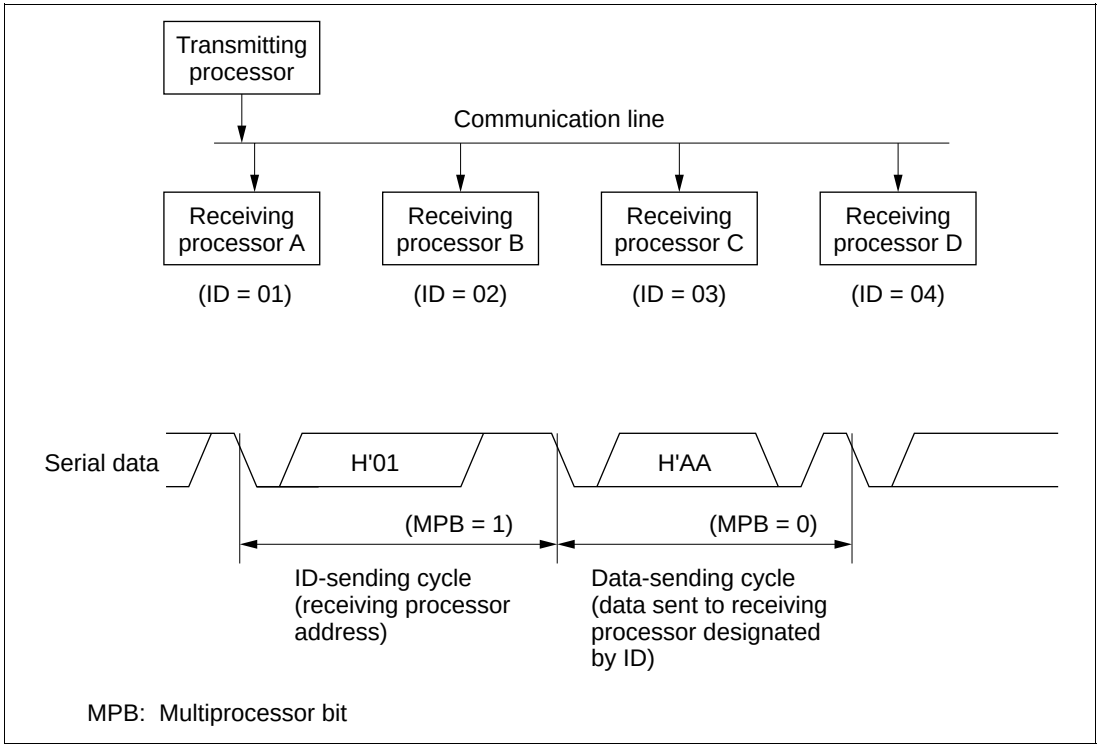
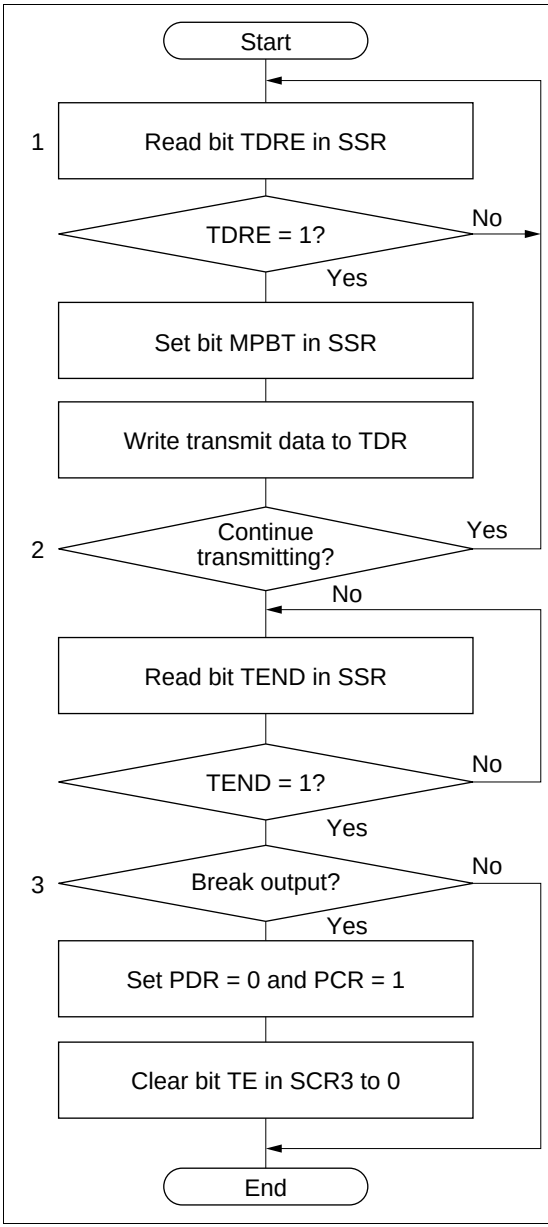


Figure 10.19 Example of Interprocessor Communication Using Multiprocessor Format (Data H'AA Sent to Receiving Processor A)

Four communication formats are available. Parity-bit settings are ignored when a multiprocessor format is selected. For details see table 10.14.

For a description of the clock used in multiprocessor communication, see 10.3.4, Operation in Asynchronous Mode.

Transmitting Multiprocessor Data: Figure 10.20 shows a typical flow chart for multiprocessor serial data transmission. After SCI3 initialization, follow the procedure below.



1. Read the serial status register (SSR), and after confirming that bit TDRE = 1, set bit MPBT (multiprocessor bit transmit) in SSR to 0 or 1, then write transmit data in the transmit data register (TDR).
   When data is written to TDR, TDRE is automatically cleared to 0.

2. To continue transmitting data, read bit TDRE to make sure it is set to 1, indicating that data writing is enabled. Then write the next data to TDR.
   When data is written to TDR, TDRE is automatically cleared to 0.

3. To output a break signal at the end of data transmission, first set the port values PCR = 1 and PDR = 0, then clear bit TE in SCR3 to 0.

Figure 10.20 Typical Multiprocessor Data Transmission Flow Chart

SCI3 operates as follows during data transmission using a multiprocessor format.

SCI3 monitors bit TDRE in SSR. When this bit is cleared to 0, SCI3 recognizes that there is data written in the transmit data register (TDR), which it transfers to the transmit shift register (TSR). Then TDRE is set to 1 and transmission starts. If bit TIE in SCR3 is set to 1, a TXI interrupt is requested.

Serial data is transmitted from pin TXD using the communication format outlined in table 10.14.

Next, TDRE is checked as the stop bit is being transmitted. If TDRE is 0, data is transferred from TDR to TSR, and after the stop bit is sent, transmission of the next frame starts. If TDRE is 1, the TEND bit in SSR is set to 1, and after the stop bit is sent the output remains at 1 (mark state). A TEI interrupt is requested in this state if bit TEIE (transmit end interrupt enable) in SCR3 is set to 1.

Figure 10.21 shows a typical SCI3 operation in multiprocessor communication mode.

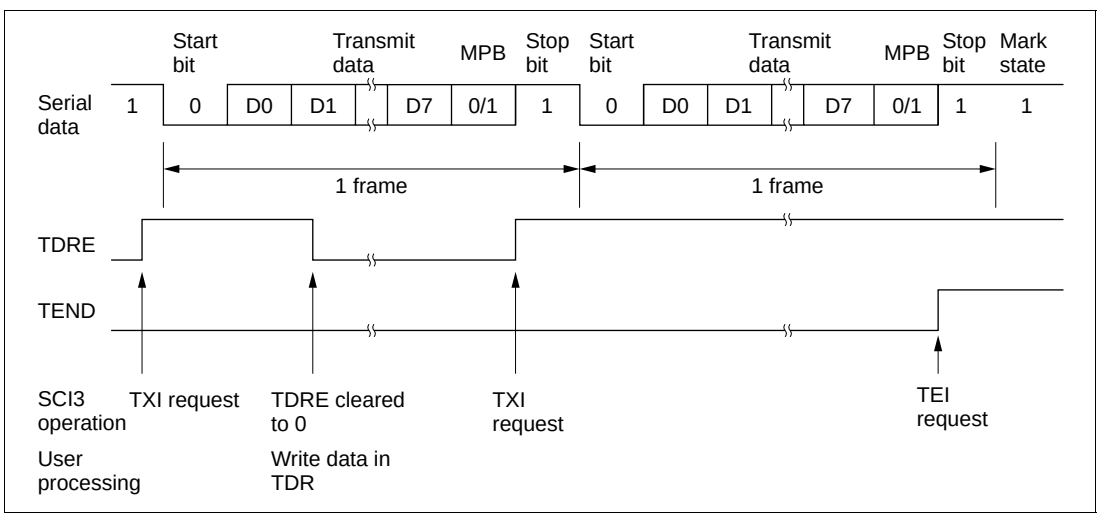


Figure 10.21 Typical Multiprocessor Format Transmit Operation (8-Bit Data, Multiprocessor Bit Added, and 1 Stop Bit)

Receiving Multiprocessor Ddata: Figure 10.22 shows a typical flow chart for receiving data using a multiprocessor format. After SCI3 initialization, follow the procedure below.

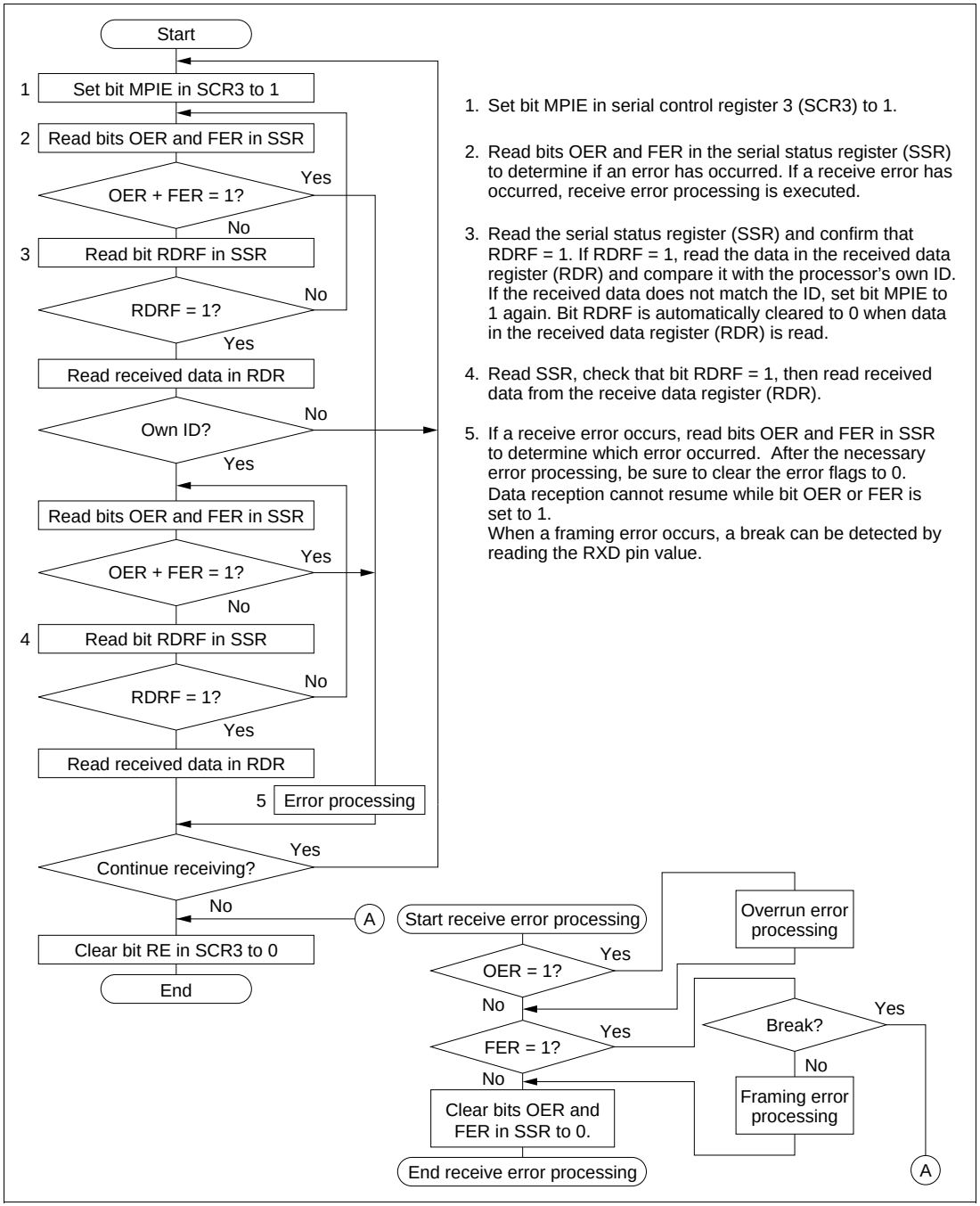


Figure 10.22 Typical Flow Chart for Receiving Serial Data Using Multiprocessor Format

Figure 10.23 gives an example of data reception using a multiprocessor format.

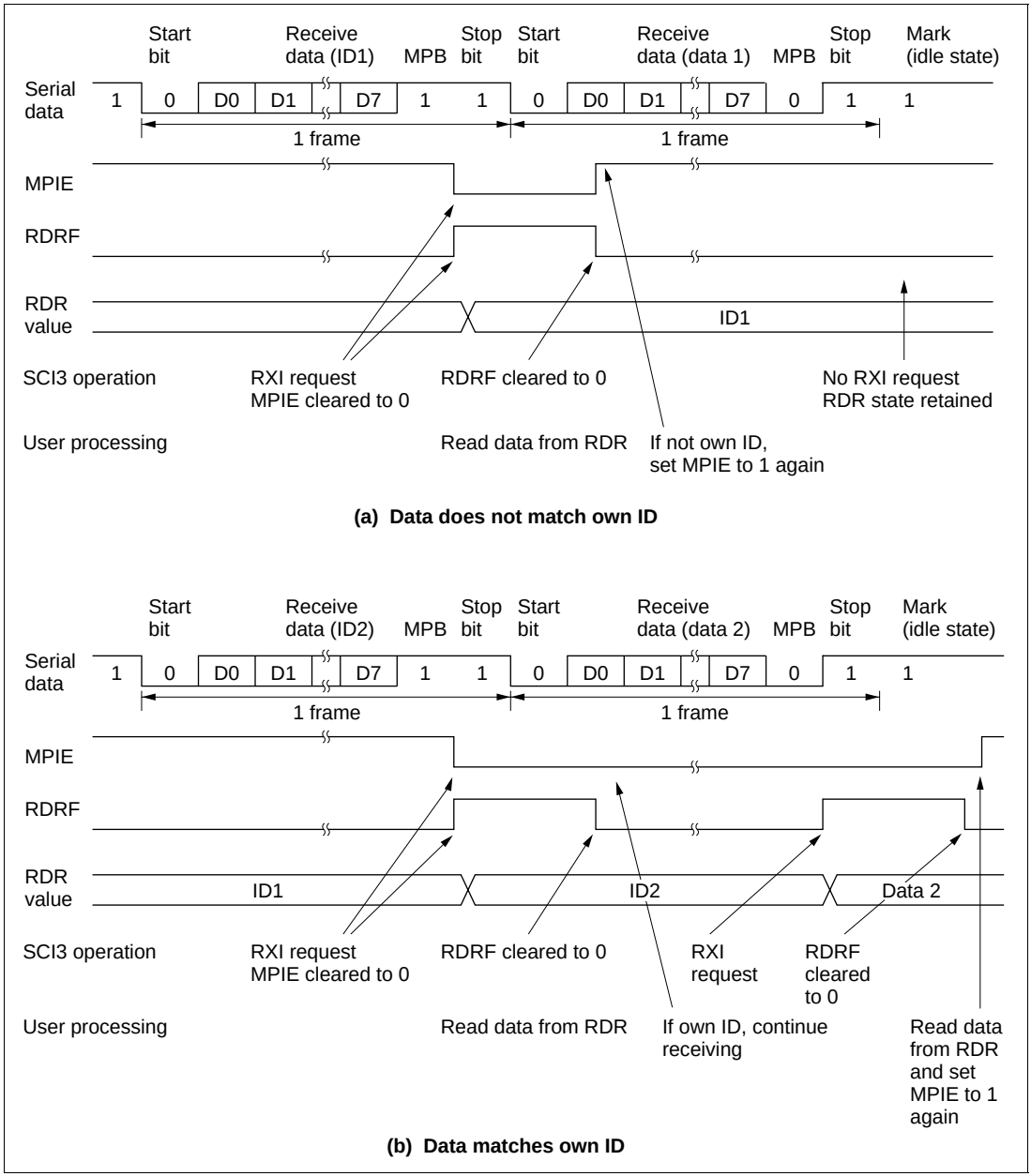


Figure 10.23 Example of Multiprocessor Format Receive Operation (8-Bit Data, Multiprocessor Bit Added, and 1 Stop Bit)

10.3.7 Interrupts

SCI3 has six interrupt sources: transmit end, transmit data empty, receive data full, and the three receive error interrupts (overrun error, framing error, and parity error). All share a common interrupt vector. Table 10.16 describes each interrupt.

Table 10.16 SCI3 Interrupts

Interrupt	Interrupt Request	Vector Address
RXI	Interrupt request due to receive data register full (RDRF)	H'0024
TXI	Interrupt request due to transmit data register empty (TDRE)	
TEI	Interrupt request due to transmit end (TEND)	
ERI	Interrupt request due to receive error (OER, FER, or PER)	

The interrupt requests are enabled and disabled by bits TIE and RIE of SCR3.

When bit TDRE in SSR is set to 1, TXI is requested. When bit TEND in SSR is set to 1, TEI is requested. These two interrupt requests occur during data transmission.

The initial value of bit TDRE is 1. Accordingly, if the transmit data empty interrupt request (TXI) is enabled by setting bit TIE to 1 in SCR3 before placing transmit data in TDR, TXI will be requested even though no transmit data has been readied.

Likewise, the initial value of bit TEND is 1. Accordingly, if the transmit end interrupt request (TEI) is enabled by setting bit TEIE to 1 in SCR3 before placing transmit data in TDR, TEI will be requested even though no data has been transmitted.

These interrupt features can be used to advantage by programming the interrupt handler to move the transmit data into TDR. When this technique is not used, the interrupt enable bits (TIE and TEIE) should not be set to 1 until after TDR has been loaded with transmit data, to avoid unwanted TXI and TEI interrupts.

When bit RDRF in SSR is set to 1, RXI is requested. When any of SSR bits OER, FER, or PER is set to 1, ERI is requested. These two interrupt requests occur during the receiving of data.

Details on interrupts are given in 3.3, Interrupts.

10.3.8 Application Notes

When using SCI3, attention should be paid to the following matters.

Relation between Bit TDRE and Writing Data to TDR: Bit TDRE in the serial status register (SSR) is a status flag indicating that TDR does not contain new transmit data. TDRE is automatically cleared to 0 when data is written to TDR. When SCI3 transfers data from TDR to TSR, bit TDRE is set to 1.

Data can be written to TDR regardless of the status of bit TDRE. However, if new data is written to TDR while TDRE is cleared to 0, assuming the data held in TDR has not yet been shifted to TSR, it will be lost. For this reason, it is recommended for securing serial data transmission that writing transmit data to TDR should be performed only once (not two or more times), always after confirming that bit TDRE is set to 1.

Operation when Multiple Receive Errors Occur at the Same Time: When two or more receive errors occur at the same time, the status flags in SSR are set as shown in table 10.17. If an overrun error occurs, data is not transferred from RSR to RDR, and receive data is lost.

Table 10.17 SSR Status Flag States and Transfer of Receive Data

SSR Status Flags				Receive Data Transfer	
RDRF*	OER	FER	PER	(RSR → RDR)	Receive Error Status
1	1	0	0	×	Overrun error
0	0	1	0	○	Framing error
0	0	0	1	○	Parity error
1	1	1	0	×	Overrun error + framing error
1	1	0	1	×	Overrun error + parity error
0	0	1	1	○	Framing error + parity error
1	1	1	1	×	Overrun error + framing error + parity error

○ : Receive data transferred from RSR to RDR

× : Receive data not transferred from RSR to RDR

Note: * RDRF keeps the same state as before the data was received.

Break Detection and Processing: Break signals can be detected by reading the RXD pin directly when a framing error (FER) is detected. In the break state the input from the RXD pin consists of all 0s, so FER is set and the parity error flag (PER) may also be set. In the break state SCI3 continues to receive, so if the FER bit is cleared to 0 it will be set to 1 again.

Sending a Mark or Break Signal: When the TXD bit in PMR6 is cleared to 0, the TXD pin becomes an I/O port, the level and direction (input or output) of which are determined by the PDR and PCR bits. This feature can be used to place the TXD pin in the mark state or to send a break signal.

To place the serial communication line in the mark (1) state before TE is set to 1, set the PDR and PCR bits both to 1. The TXD pin becomes an I/O port outputting the value 1.

To send a break signal during transmission, set the PCR bit to 1 and clear the PDR bit to 0, then clear the TXD bit in PMR6 to 0.

When the TXD bit in PMR6 is cleared to 0, the TXD pin becomes an I/O port outputting 0, regardless of the current transmission status.

Receive Error Flags and Transmit Operation (Sysnchronous Mode Only): When a receive error flag (ORER, PER, or FER) is set to 1, SCI3 will not start transmitting even if TDRE is cleared to 0. Be sure to clear the receive error flags to 0 when starting to transmit. Note that clearing RE to 0 does not clear the receive error flags.

Receive Data Sampling Timing and Receive Margin in Asynchronous Mode: In asynchronous mode SCI3 operates on a base clock with 16 times the bit rate frequency. In receiving, SCI3 synchronizes internally with the falling edge of the start bit, which it samples on the base clock. Receive data is latched at the rising edge of the eighth base clock pulse. See figure 10.24.

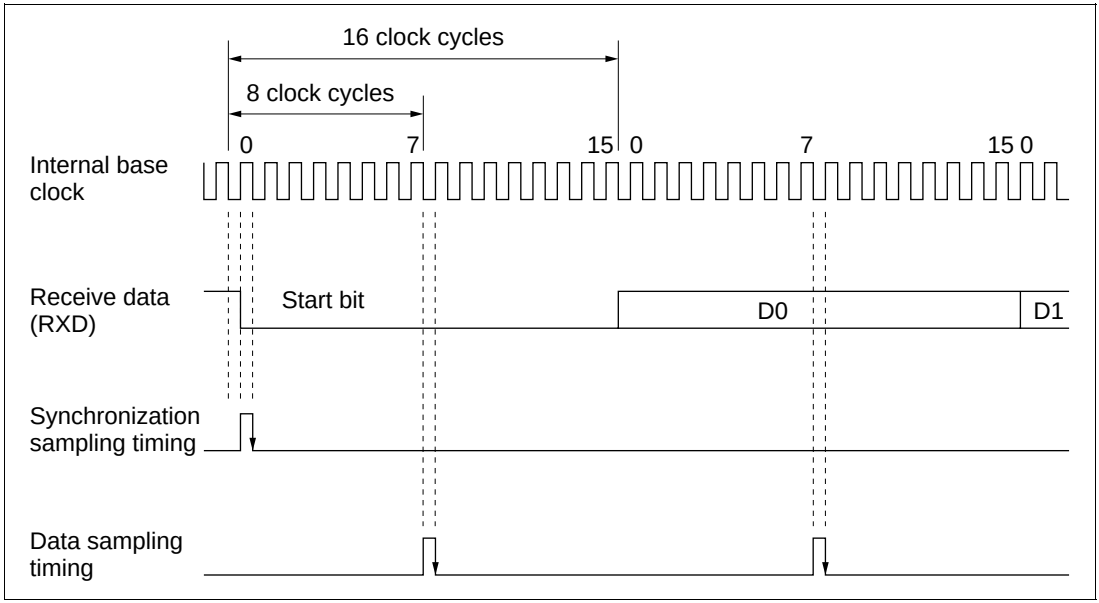


Figure 10.24 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be derived from the following equation.

$$M = \left\{(0.5 - \frac{1}{2N}) - \frac{D - 0.5}{N} - (L - 0.5)\,F\right\} \times 100\% \quad \text{..... Equation (1)}$$

M: Receive margin (%)
N: Ratio of clock frequency to bit rate (N = 16)
D: Clock duty cycle (D = 0.5 to 1)
L: Frame length (L = 9 to 12)
F: Absolute value of clock frequency error

In equation (1), if F (absolute value of clock frequency error) = 0 and D (clock duty cycle) = 0.5, the receive margin is 46.875% as given by equation (2) below.

When D = 0.5 and F = 0,

$$M = \{0.5 - 1/(2 \times 16)\} \times 100\% = 46.875\% \quad \text{..... Equation (2)}$$

This value is theoretical. In actual system designs a margin of from 20 to 30 percent should be allowed.

Relationship between Bit RDRF and Reading RDR: While SCI3 is receiving, it checks the RDRF flag. If the RDRF flag is cleared to 0 when the reception of one frame of data is completed, data reception ends normally. If RDRF is set to 1, an overrun error occurs.

RDRF is automatically cleared to 0 when the contents of RDR are read. If RDR is read more than once, the second and later reads will be performed with RDRF cleared to 0. Note that when RDR is read while RDRF is 0, data from the next frame may be read if this reading operation is carried out at the same time that the reception of the next frame is completed. This is illustrated in figure 10.25.

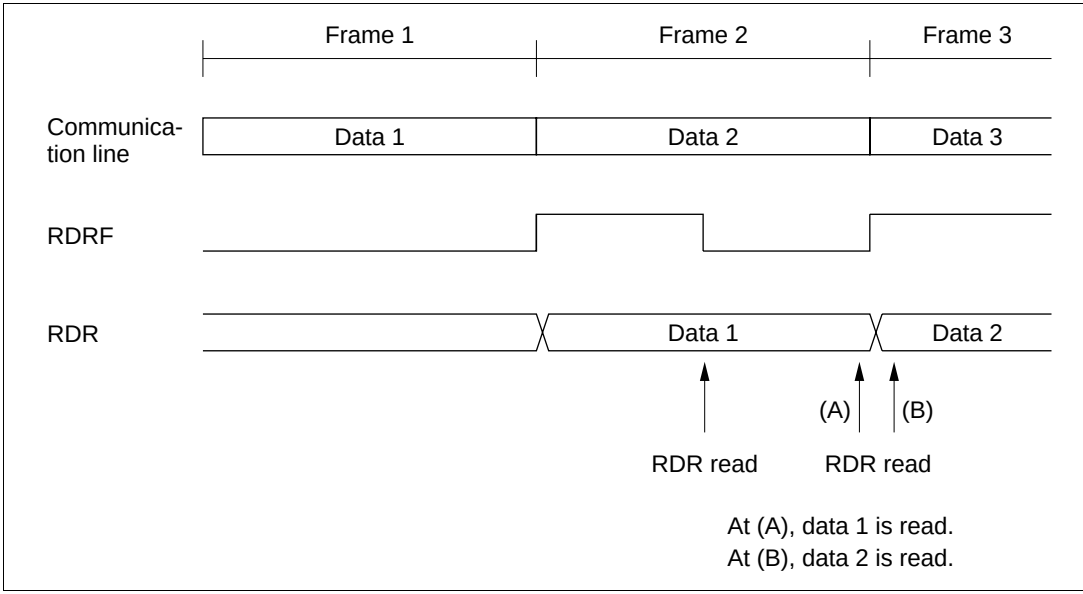


Figure 10.25 Relationship between Data and RDR Read Timing

To avoid the situation described above, RDR reading should be carried out only once (not two or more times) after confirming that bit RDRF is set to 1.

When reading RDR more than once, be sure to copy any data read for the first time to RAM, for example, and use the copied data. Also note that RDR reading should be carried out with a safe margin just before reception of the next frame is completed. More concretely, RDR reading should be completed before transferring bit 7 in the synchronous mode or before transferring the stop bit in the asynchronous mode.

Section 11 DTMF Generator

11.1 Overview

The H8/3627 Series has an on-chip dual-tone multifrequency (DTMF) generator that can generate DTMF signals.

A DTMF signal accesses a telephone switching system by a pair of sine waves. Figure 11.1 shows the frequency matrix. The DTMF generator generates frequencies corresponding to the numbers and symbols on the keypad of a telephone set or facsimile machine.

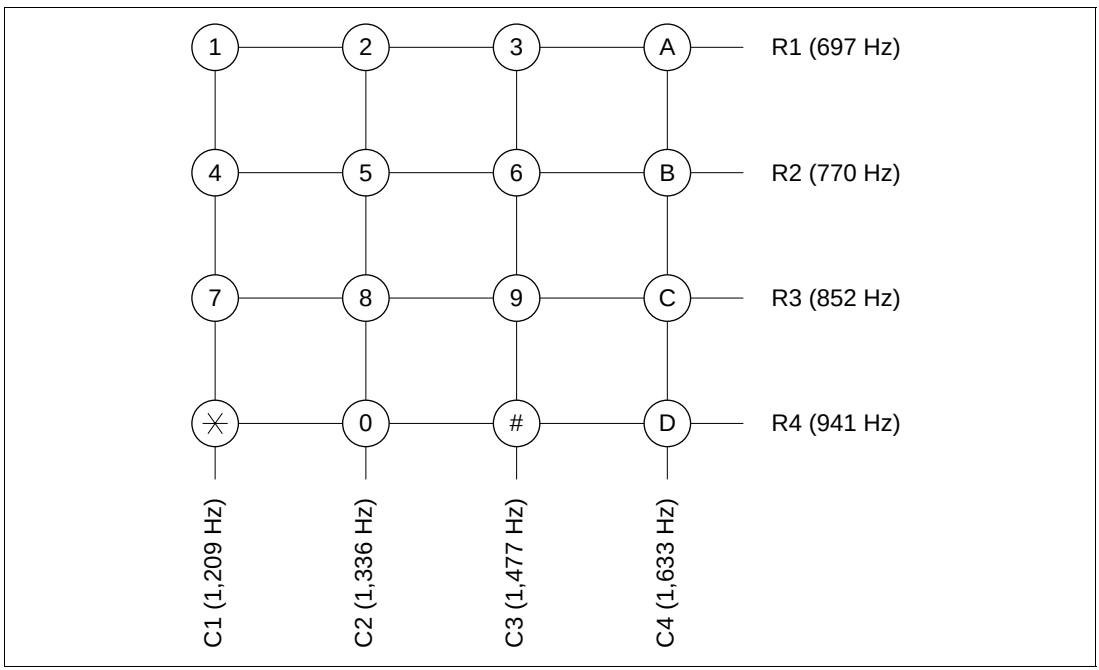


Figure 11.1 DTMF Frequencies

11.1.1 Features

Features of the DTMF generator are as follows.

- Generates sine waves with DTMF frequencies from the system clock input at the OSC pins (f_{OSC})

  The OSC clock (1.2 MHz to 10 MHz, selectable in 400-kHz steps) is divided to generate a 400-kHz clock. Input to a feedback loop with a modified programmable divider and sine-wave counter, this clock is used to generate sine waves with the DTMF frequencies.
- Stable sine-wave output with low distortion

  Sine waves are output from a high-precision resistor-ladder-type D/A converter. Each cycle is divided into 32 segments to give a stable waveform with low distortion.
- Composite or single waveform output

  Register settings can select combined row-and-column-group output, or independent row-group or column-group output.

11.1.2 Block Diagram

Figure 11.2 shows a block diagram of the DTMF generator.

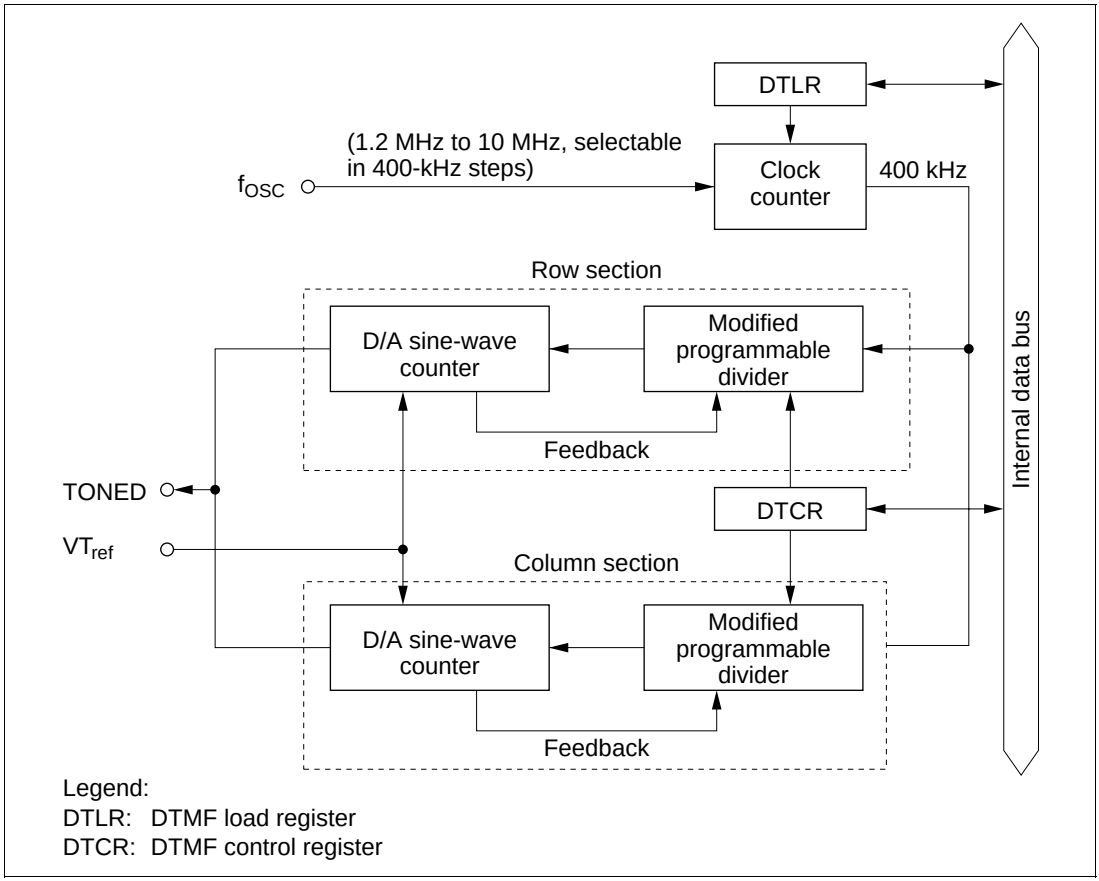


Figure 11.2 DTMF Generator Block Diagram

11.1.3 Pin Configuration

Table 11.1 shows the pins assigned to the DTMF generator.

Table 11.1 Pin Configuration

Name	Abbrev.	I/O	Function
DTMF output reference level power supply pin	VT_{ref}	—	Reference level voltage for DTMF output
DTMF signal output pin	TONED	Output	DTMF signal output

11.1.4 Register Configuration

Table 11.2 shows the register configuration of the DTMF generator.

Table 11.2 Register Configuration

Name	Abbrev.	R/W	Initial Value	Address
DTMF control register	DTCR	R/W	H'40	H'FFB2
DTMF load register	DTLR	R/W	H'E0	H'FFB3

11.2 Register Descriptions

11.2.1 DTMF Control Register (DTCR)

Bit	7	6	5	4	3	2	1	0
	DTEN	—	CLOE	RWOE	CLF1	CLF0	RWF1	RWF0
Initial value	0	1	0	0	0	0	0	0
Read/Write	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W

DTCR is an 8-bit read/write register that enables the DTMF generator, enables row and column output, and selects the output frequencies.

Upon reset, DTCR is initialized to H'40.

Bit 7—DTMF Generator Enable (DTEN): Bit 7 enables or disables operation of the DTMF generator.

Bit 7: DTEN	Description	
0	DTMF generator is halted	(initial value)
1	DTMF generator operates	

Bit 6—Reserved Bit: Bit 6 is reserved: it is always read as 1, and cannot be modified.

Bit 5—Column Output Enable (CLOE): Bit 5 enables or disables DTMF column signal output.

Bit 5: CLOE	Description	
0	DTMF column signal output is disabled (high-impedance)	(initial value)
1	DTMF column signal output is enabled	

Bit 4—Row Output Enable (RWOE): Bit 4 enables or disables DTMF row signal output.

Bit 4: RWOE	Description	
0	DTMF row signal output is disabled (high-impedance)	(initial value)
1	DTMF row signal output is enabled	

Bits 3 and 2—DTMF Column Signal Output Frequency 1 and 0 (CLF1, CLF0): Bits 3 and 2 select the DTMF column signal frequency (C1 to C4).

Bit 3: CLF1	Bit 2: CLF0	Description
0	0	DTMF column signal output frequency: 1209 Hz (C1) (initial value)
	1	DTMF column signal output frequency: 1336 Hz (C2)
1	0	DTMF column signal output frequency: 1447 Hz (C3)
	1	DTMF column signal output frequency: 1633 Hz (C4)

Bits 1 and 0—DTMF Row Signal Output Frequency 1 and 0 (RWF1, RWF0): Bits 1 and 0 select the DTMF row signal frequency (R1 to R4).

Bit 1: RWF1	Bit 0: RWF0	Description
0	0	DTMF row signal output frequency: 697 Hz (R1) (initial value)
	1	DTMF row signal output frequency: 770 Hz (R2)
1	0	DTMF row signal output frequency: 852 Hz (R3)
	1	DTMF row signal output frequency: 941 Hz (R4)

11.2.2 DTMF Load Register (DTLR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	DTL4	DTL3	DTL2	DTL1	DTL0
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

DTLR is an 8-bit read/write register that specifies the ratio by which the clock frequency at the OSC pins is divided for input to the DTMF generator.

Upon reset, DTLR is initialized to H'E0.

Bits 7 to 5—Reserved Bits: Bits 7 to 5 are reserved: they are always read as 1, and cannot be modified.

Bits 4 to 0—OSC Clock Division Ratio 4 to 0 (DTL4 to DTL0): Bits 4 to 0 specify a division ratio of the OSC clock frequency which will generate a 400-kHz clock for input to the DTMF generator. The ratio is set as a counter value from 3 to 25, corresponding to OSC clock frequencies of 1.2 to 10 MHz (in 400-kHz steps).

					Description	
Bit 4: DTL4	**Bit 3: DTL3**	**Bit 2: DTL2**	**Bit 1: DTL1**	**Bit 0: DTL0**	**Division Ratio**	**OSC Clock Frequency**
0	0	0	0	0	Illegal setting	(initial value)
				1	Illegal setting	
			1	0	Illegal setting	
				1	3	1.2 MHz
		1	0	0	4	1.6 MHz
:	:	:	:	:	:	:
1	1	0	0	1	25	10 MHz
			1	*	Illegal setting	
		1	*	*	Illegal setting	

Note: * Don’t care

These bits must be set to the correct value. Normal DTMF signal output frequencies will not be obtained if these bits are set to a value not matching the clock input at the OSC pins. Operation is not guaranteed if these bits are set to a value other than 3 to 25.

11.3 Operation

11.3.1 Output Waveform

The DTMF generator outputs a row-group or column-group sine wave (DTMF signal) or a combined row-column waveform at the TONED pin. These signals are generated by a high-precision resistor-ladder-type D/A converter circuit. The output frequency is selected by DTCR.

Figure 11.3 shows an equivalent circuit for the TONED output. Figure 11.4 shows the output waveform of an independent row-group or column-group signal. One cycle of the output is divided into 32 segments, giving a stable output with low distortion.

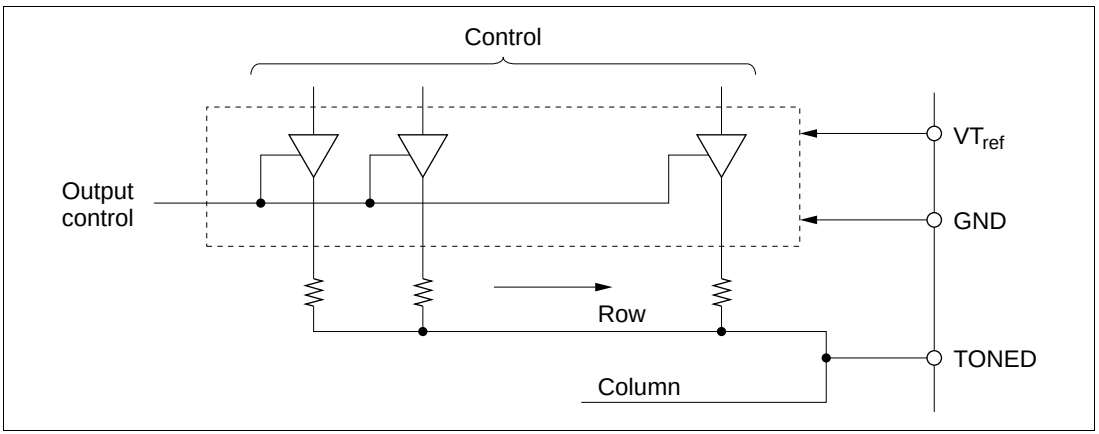


Figure 11.3 Equivalent Circuit for TONED Output

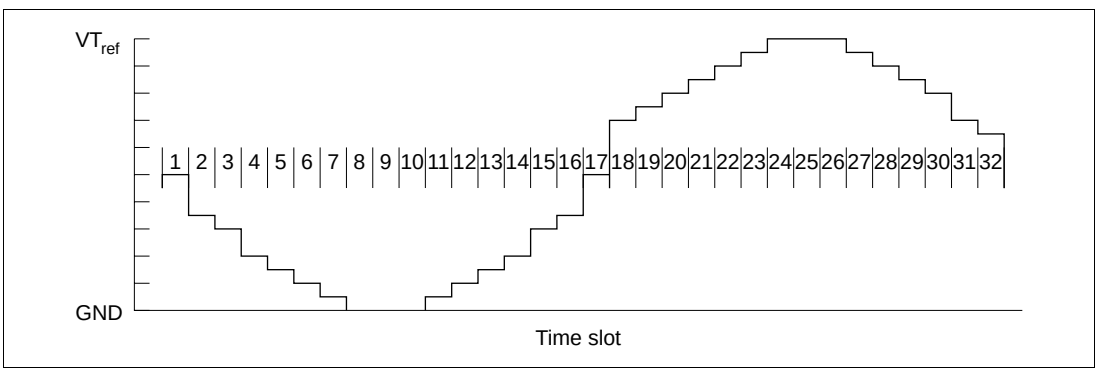


Figure 11.4 TONED Output Waveform (Independent Row- or Column-Group Output)

Table 11.3 indicates the frequency deviation between the DTMF signals output by the DTMF generator and the nominal (standard) signal values.

Table 11.3 Frequency Deviation of DTMF Signals from Nominal Signals

Symbol	Standard Signal Frequency (Hz)	DTMF Signal Output (Hz)	Frequency Deviation (%)
R1	697	694.44	–0.37
R2	770	769.23	–0.10
R3	852	851.06	–0.11
R4	941	938.97	–0.22
C1	1209	1212.12	0.26
C2	1336	1333.33	–0.20
C3	1477	1481.48	0.30
C4	1633	1639.34	0.39

11.3.2 Operation Flow

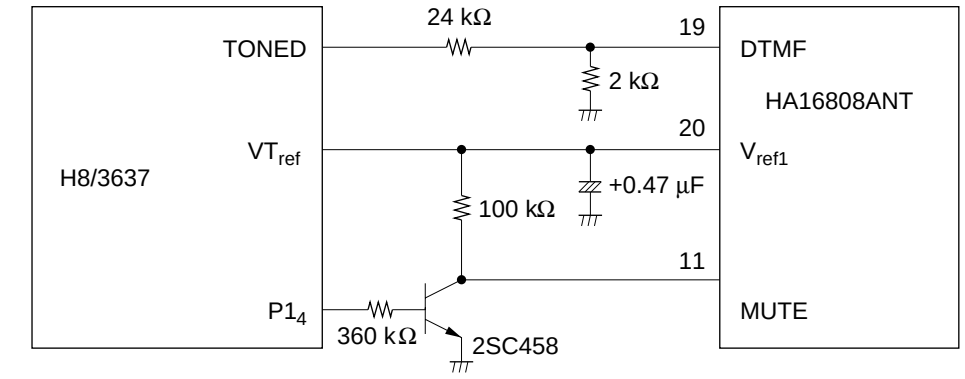
The procedure for using the DTMF generator is given below.

1. Set the OSC clock division ratio in DTLR to match the frequency of the connected system clock oscillator (1.2 MHz to 10 MHz in 400-kHz steps).
2. Select a row (R1 to R4) and/or column (C1 to C4) frequency with bits CLF1, CLF0, RWF1, and RWF0 in DTCR.
3. Select row and/or column output with the CLOE and RWOE bits in DTCR, and set the DTEN bit to 1 to enable the DTMF generator.

This procedure outputs the selected DTMF signal from the TONED pin.

11.4 Typical Use

Figure 11.5 shows an example of the use of the DTMF generator.



Note: Numbers at ends of signal lines are pin numbers of HA16808ANT.

Figure 11.5 Connection to HA16808 ANT

11.5 Application Notes

When using the DTMF generator, note the following point:

Be sure that the DTLR setting (DTL4 to DTL0) matches the system clock frequency at the OSC pins. Normal DTMF signal output frequencies will not be obtained unless the DTLR setting matches the OSC frequency.

Section 12 A/D Converter

12.1 Overview

The H8/3627 Series includes on-chip a resistance-ladder-based successive-approximation analog-to-digital converter, and can convert up to two channels of analog input.

12.1.1 Features

The A/D converter has the following features.

- 8-bit resolution
- 2 input channels
- Conversion time: approx. 12.4 μs per channel (at 5 MHz operation)
- Built-in sample-and-hold function
- Interrupt requested on completion of A/D conversion
- A/D conversion can be started by external trigger input

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the A/D converter.

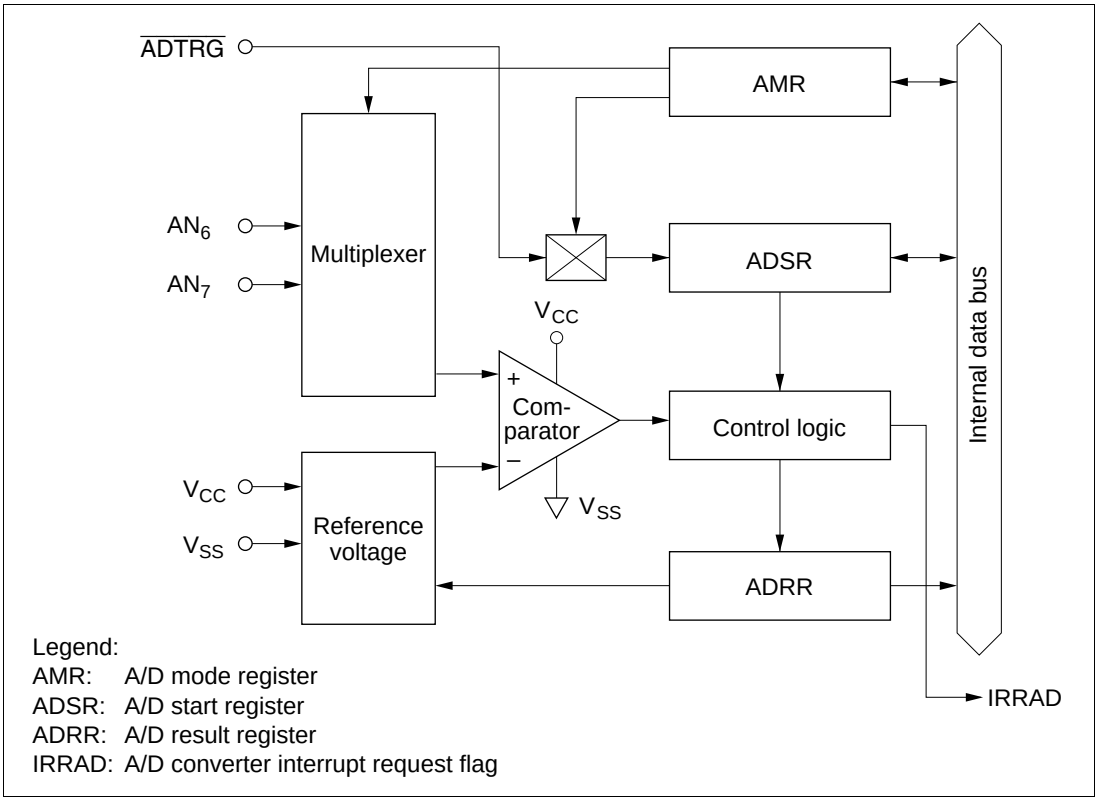


Figure 12.1 Block Diagram of the A/D Converter

12.1.3 Pin Configuration

Table 12.1 shows the A/D converter pin configuration.

Table 12.1 Pin Configuration

Name	Abbrev.	I/O	Function
Power supply pin	V_{CC}	Input	Power supply
Ground pin	V_{SS}	Input	Ground and reference voltage
Analog input pin 6	AN_6	Input	Analog input channel 6
Analog input pin 7	AN_7	Input	Analog input channel 7
External trigger input pin	$\overline{ADTRG}$	Input	External trigger input for starting A/D conversion

12.1.4 Register Configuration

Table 12.2 shows the A/D converter register configuration.

Table 12.2 Register Configuration

Name	Abbrev.	R/W	Initial Value	Address
A/D mode register	AMR	R/W	H'10	H'FFC4
A/D start register	ADSR	R/W	H'7F	H'FFC6
A/D result register	ADRR	R	Undefined	H'FFC5

12.2 Register Descriptions

12.2.1 A/D Result Register (ADRR)

Bit	7	6	5	4	3	2	1	0
	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
Initial value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Read/Write	R	R	R	R	R	R	R	R

The A/D result register (ADRR) is an 8-bit read-only register for holding the results of analog-to-digital conversion.

ADRR can be read by the CPU at any time, but the ADRR values during A/D conversion are undefined.

After A/D conversion is complete, the conversion result is stored in ADRR as 8-bit data; this data is held in ADRR until the next conversion operation starts.

ADRR is not cleared on reset.

12.2.2 A/D Mode Register (AMR)

Bit	7	6	5	4	3	2	1	0
	CKS	TRGE	CKS1	—	CH3	CH2	CH1	CH0
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

AMR is an 8-bit read/write register for specifying the A/D conversion speed, external trigger option, and the analog input pins.

Upon reset, AMR is initialized to H'10.

Bit 7—Clock Select (CKS): Bits CKS and CKS1 select the A/D conversion speed.

			Conversion Time	
Bit 5: CKS1	**Bit 7: CKS**	**Conversion Period**	**ø = 2 MHz**	**ø = 5 MHz**
0	0	Reserved (initial value)	—	—
	1	124/ø	62 µs	24.8 µs
1	0	62/ø	31 µs	12.4 µs
	1	31/ø	15.5 µs	—*

Note: * Operation is not guaranteed if the conversion time is less than 12.4 µs. Set the bits to get a value of at least 12.4 µs.

Bit 6—External Trigger Select (TRGE): Bit 6 enables or disables the start of A/D conversion by external trigger input.

Bit 6: TRGE	**Description**
0	Disables start of A/D conversion by external trigger (initial value)
1	Enables start of A/D conversion by rising or falling edge of external trigger at pin $\overline{\text{ADTRG}}$*

Note: * The external trigger ($\overline{\text{ADTRG}}$) edge is selected by bit IEG4 of the interrupt edge select register (IEGR). See 3.3.2 for details.

Bit 5—Clock Select 1 (CKS1): Bits CKS and CKS1 select the A/D conversion speed. See bit 7, clock select (CKS) for details.

Bit 4—Reserved Bit: Bit 4 is reserved; it is always read as 1, and cannot be modified.

Bits 3 to 0—Channel Select 3 to 0 (CH3 to CH0): Bits 3 to 0 select the analog input channel.

The channel selection should be made while bit ADSF is cleared to 0.

Bit 3: CH3	Bit 2: CH2	Bit 1: CH1	Bit 0: CH0	Analog Input Channel	
0	0	*	*	No channel selected	(initial value)
	1	*	*	Reserved	
1	0	0	*	Reserved	
		1	0	AN_6	
			1	AN_7	
	1	*	*	Reserved	

Note: * Don't care

12.2.3 A/D Start Register (ADSR)

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

The A/D start register (ADSR) is an 8-bit read/write register for starting and stopping A/D conversion.

A/D conversion is started by writing 1 to the A/D start flag (ADSF) or by input of the designated edge of the external trigger signal, which also sets ADSF to 1. When conversion is complete, the converted data is set in the A/D result register (ADRR), and at the same time ADSF is cleared to 0.

Bit 7—A/D Start Flag (ADSF): Bit 7 controls and indicates the start and end of A/D conversion.

Bit 7: ADSF	Description		
0	Read access	Indicates the completion of A/D conversion.	(initial value)
	Write access	Stops A/D conversion.	
1	Read access	Indicates A/D conversion in progress.	
	Write access	Starts A/D conversion.	

Bits 6 to 0—Reserved Bits: Bits 6 to 0 are reserved; they are always read as 1, and cannot be modified.

12.3 Operation

12.3.1 A/D Conversion Operation

The A/D converter operates by successive approximations, and yields its conversion result as 8-bit data.

A/D conversion begins when software sets the A/D start flag (bit ADSF) to 1. Bit ADSF keeps a value of 1 during A/D conversion, and is cleared to 0 automatically when conversion is complete.

The completion of conversion also sets bit IRRAD in interrupt request register 2 (IRR2) to 1. An A/D conversion end interrupt is requested if bit IENAD in interrupt enable register 2 (IENR2) is set to 1.

If the conversion time or input channel needs to be changed in the A/D mode register (AMR) during A/D conversion, bit ADSF should first be cleared to 0, stopping the conversion operation, in order to avoid malfunction.

12.3.2 Start of A/D Conversion by External Trigger Input

The A/D converter can be made to start A/D conversion by input of an external trigger signal. External trigger input is enabled at pin $\overline{\text{ADTRG}}$ when bit IRQ4 in port mode register 2 for the I/O port (PMR2) is set to 1, and bit TRGE in AMR is set to 1. Then when the input signal edge designated in bit IEG4 of the IRQ edge select register (IEGR) is detected at pin $\overline{\text{ADTRG}}$, bit ADSF in ADSR will be set to 1, starting A/D conversion.

Figure 12.2 shows the timing.

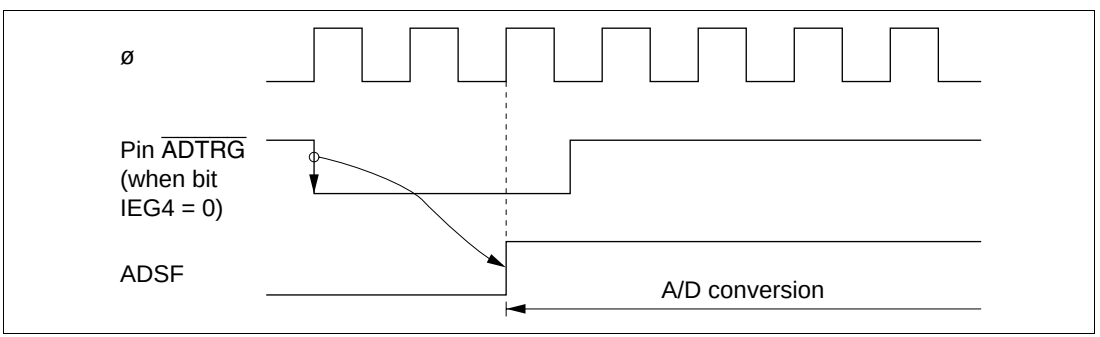


Figure 12.2 External Trigger Input Timing

12.4 Interrupts

When A/D conversion ends (ADSF changes from 1 to 0), bit IRRAD in interrupt request register 2 (IRR2) is set to 1.

A/D conversion end interrupts can be enabled or disabled by means of bit IENAD in interrupt enable register 2 (IENR2).

For further details see 3.3, Interrupts.

12.5 Typical Use

An example of how the A/D converter can be used is given below, using channel 6 (pin AN_6) as the analog input channel. Figure 12.3 shows the operation timing.

1. Bits CH3 to CH0 of the A/D mode register (AMR) are set to 1010, making pin AN_6 the analog input channel. A/D interrupts are enabled by setting bit IENAD to 1, and A/D conversion is started by setting bit ADSF to 1.
2. When A/D conversion is complete, bit IRRAD is set to **1**, and the A/D conversion result is stored in the A/D result register (ADRR). At the same time ADSF is cleared to 0, and the A/D converter goes to the idle state.
3. Bit IENAD = 1, so an A/D conversion end interrupt is requested.
4. The A/D interrupt handling routine starts.
5. The A/D conversion result is read and processed.
6. The A/D interrupt handling routine ends.

If ADSF is set to 1 again afterward, A/D conversion starts and steps 2 through 6 take place.

Figures 12.4 and 12.5 show flow charts of procedures for using the A/D converter.

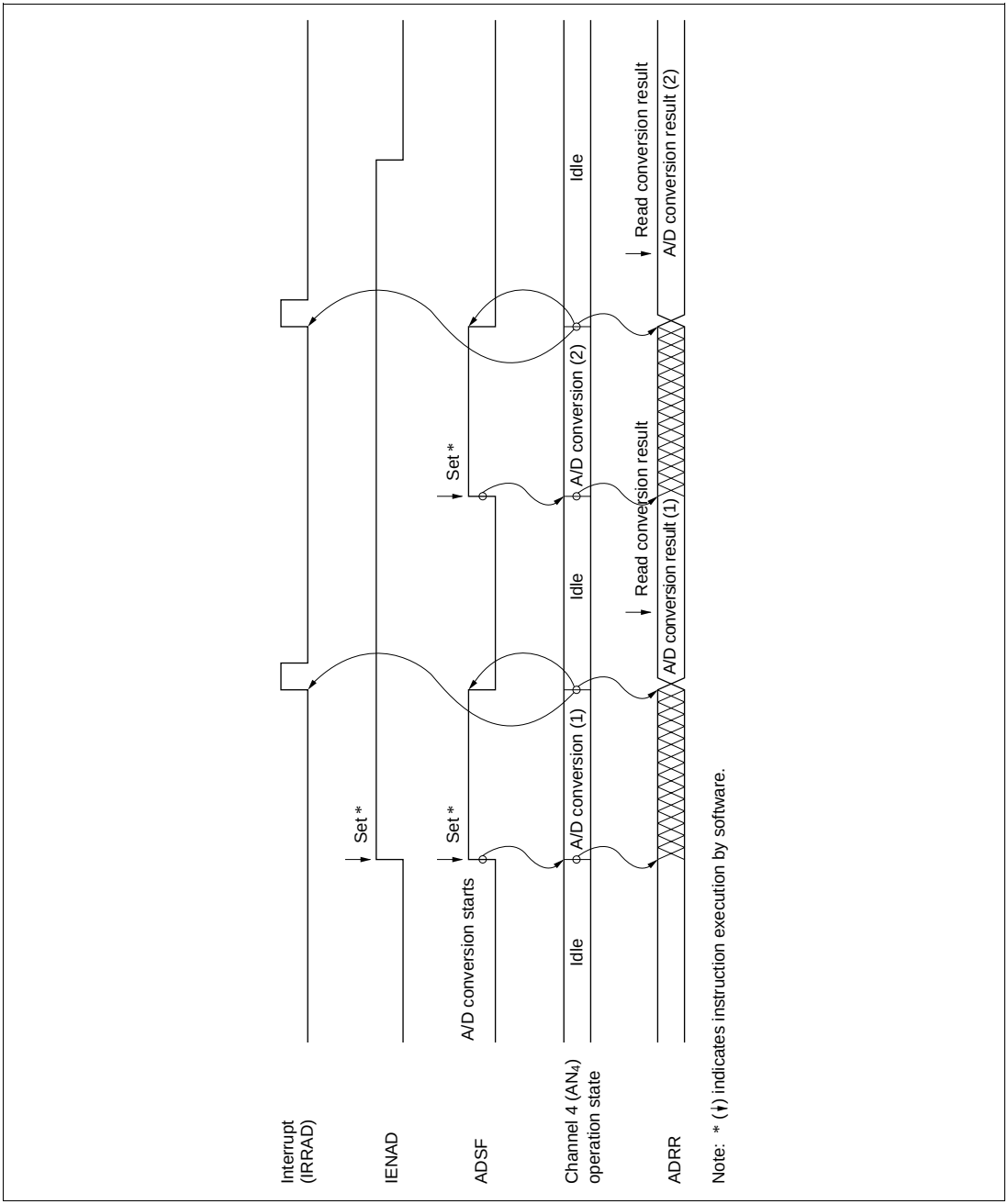


Figure 12.3 Typical A/D Converter Operation Timing

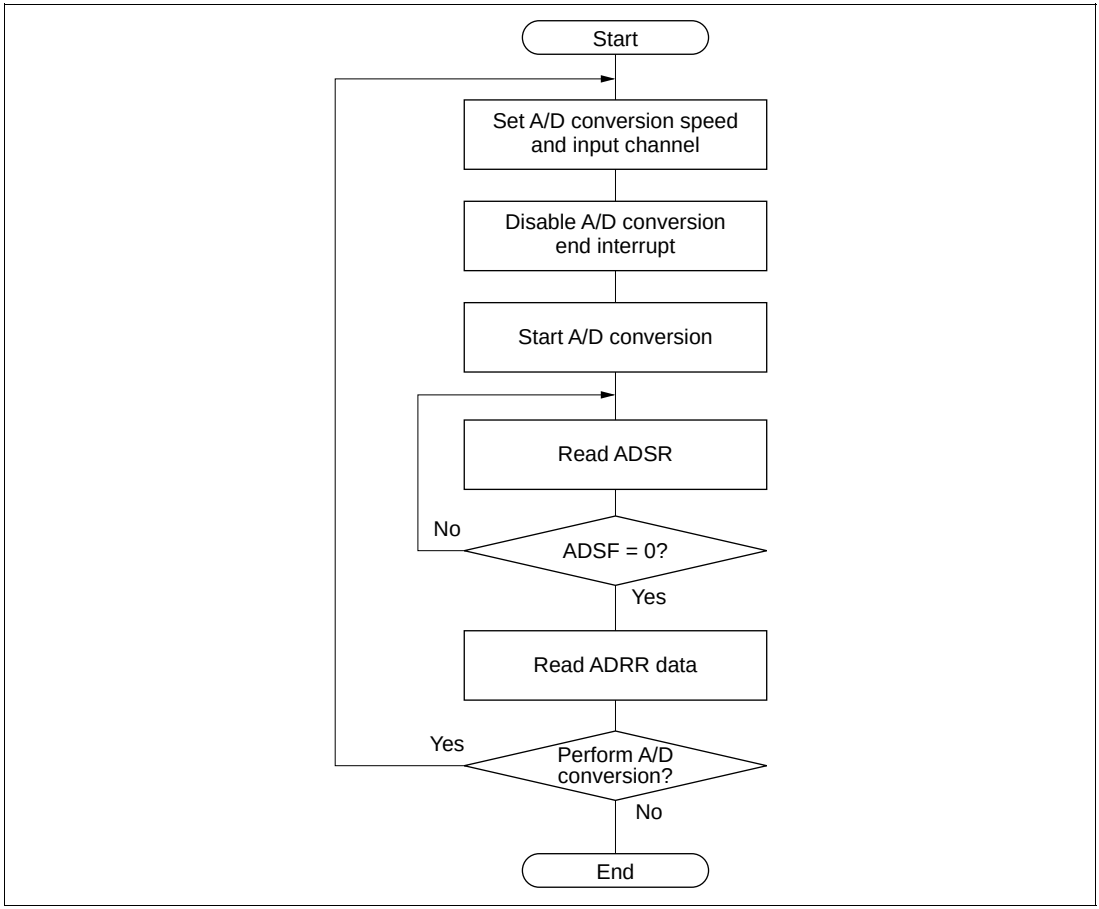


Figure 12.4 Flow Chart of Procedure for Using A/D Converter (1) (Polling by Software)

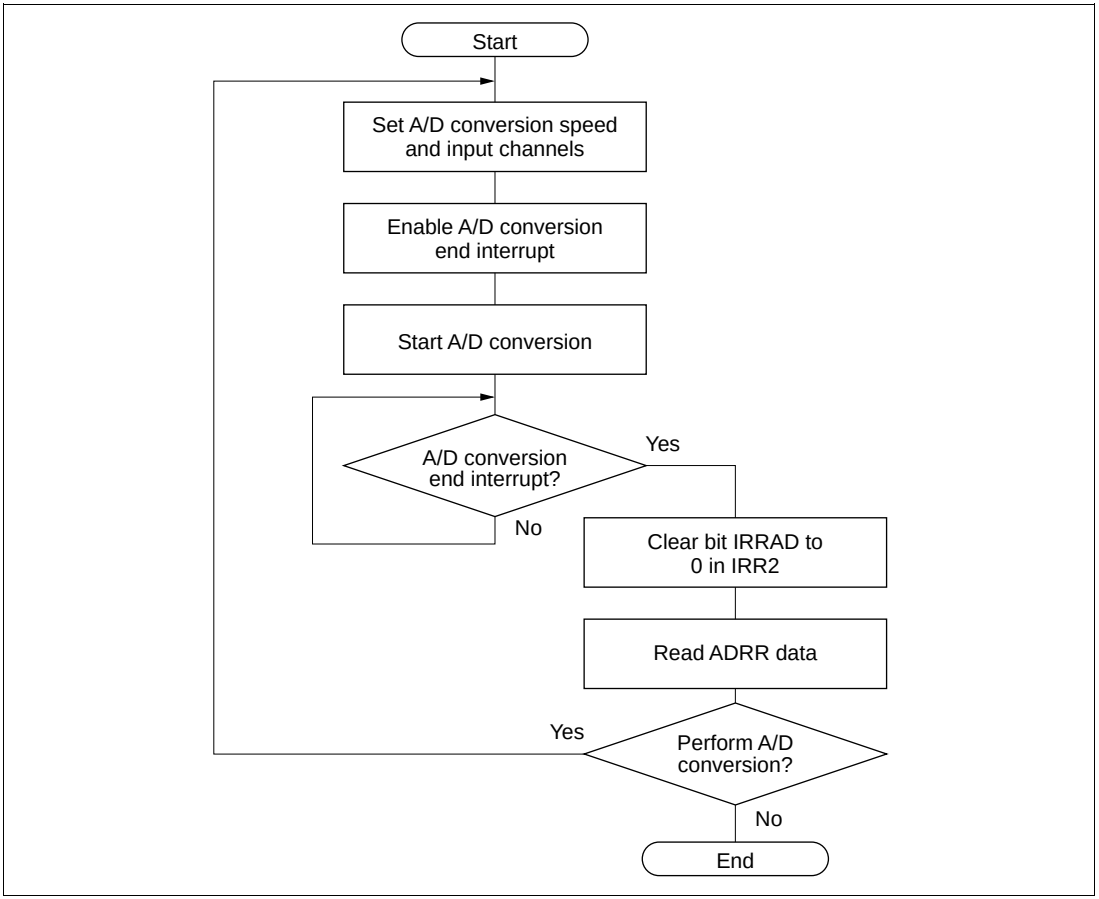


Figure 12.5 Flow Chart of Procedure for Using A/D Converter (2) (Interrupts Used)

12.6 Application Notes

- Data in the A/D result register (ADRR) should be read only when the A/D start flag (ADSF) in the A/D start register (ADSR) is cleared to 0.
- Changing the digital input signal at an adjacent pin during A/D conversion may adversely affect conversion accuracy.

Section 13 Power Supply Circuit

13.1 Overview

The H8/3627 Series incorporates an internal power supply step-down circuit. Use of this circuit enables the internal power supply to be fixed at a constant level of approximately 3.0 V, independent of the voltage of the power supply connected to the external V_{CC} pin. As a result, rise of the current consumption when an external power supply is used at 3.0 V or above can be held down.

13.2 Configuration of the Internal Power Supply Step-Down Circuit

Connect the external power supply to the V_{CC} pin, and connect a capacitance of approximately 0.1 µF between CV_{CC} and V_{SS}, as shown in figure 13.1. Use without the capacitance may cause malfunction.

Note: In the external circuit interface, the external power supply voltage connected to V_{CC} and the GND potential connected to V_{SS} are the reference levels. For example, for port input/output levels, the V_{CC} level is the reference for the high level, and the V_{SS} level is that for the low level.

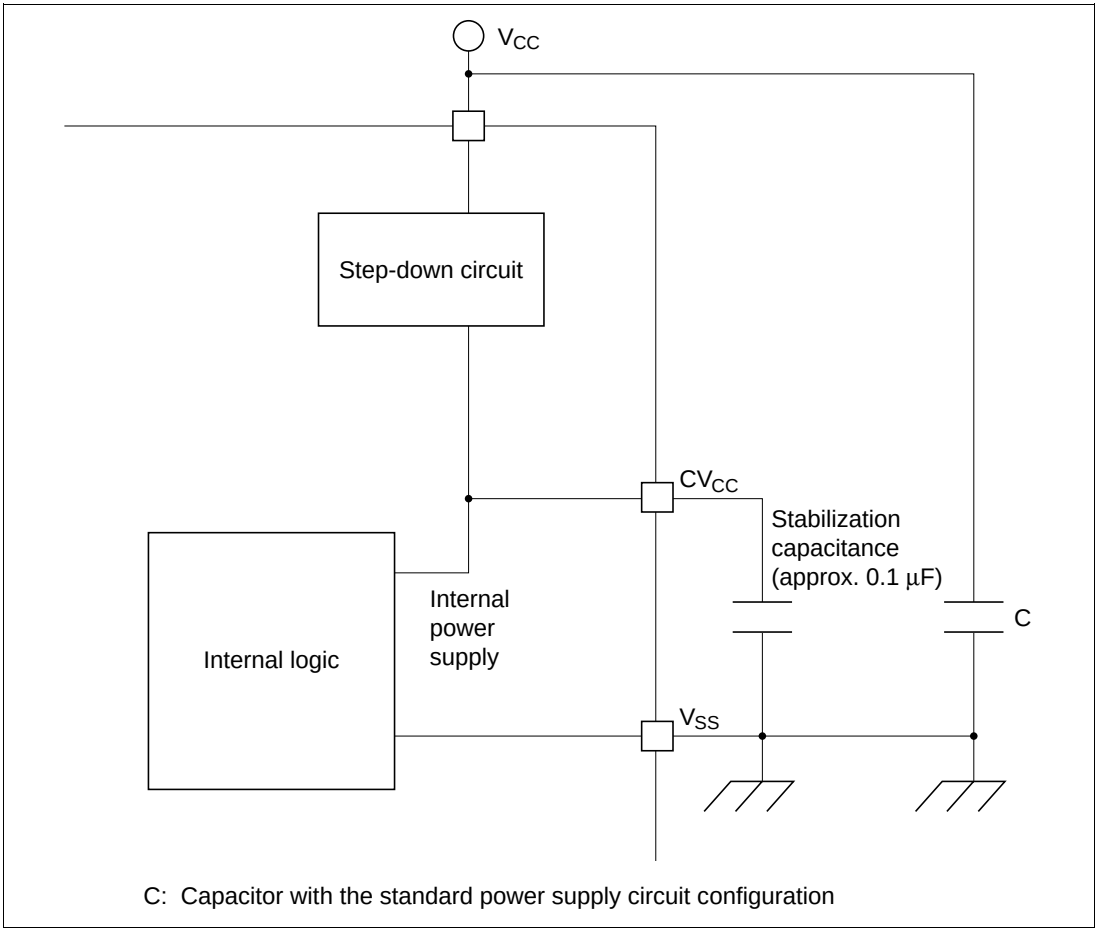


Figure 13.1 Power Supply Connection when Internal Step-Down Circuit is Used

Section 14 Electrical Characteristics

14.1 Absolute Maximum Ratings

Table 14.1 lists the absolute maximum ratings.

Table 14.1 Absolute Maximum Ratings

Item		Symbol	Value	Unit	Notes
Power supply voltage		V_{CC}, CV_{CC}	–0.3 to +7.0	V	1
Reference level supply voltage		VT_{ref}	–0.3 to V_{CC} +0.3	V	1
Programming voltage		V_{PP}	–0.3 to +13.0	V	1
Input voltage	Ports other than port B	V_{in}	–0.3 to V_{CC} +0.3	V	1
	Port B	AV_{in}	–0.3 to V_{CC} +0.3	V	1
Operating temperature		T_{opr}	–20 to +75	°C	1
Storage temperature		T_{stg}	–55 to +125	°C	1

Note: 1. Permanent damage may occur to the chip if maximum ratings are exceeded. Normal operation should be under the conditions specified in Electrical Characteristics. Exceeding these values can result in incorrect operation and reduced reliability.

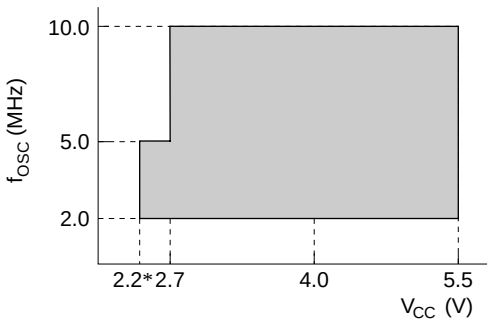
14.2 Electrical Characteristics

14.2.1 Power Supply Voltage and Operating Range

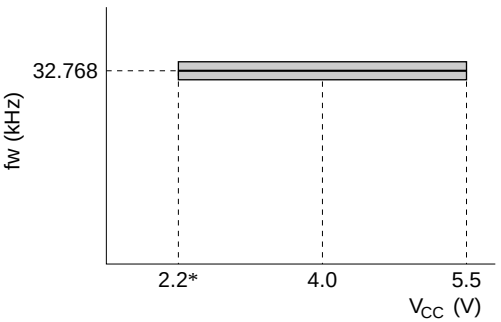
The power supply voltage and operating range are indicated by the shaded region in the figures below.

Note: Caution is required during development, since the guaranteed operating ranges of the chip and development tools are different.

Power Supply Voltage vs. Oscillator Frequency Range



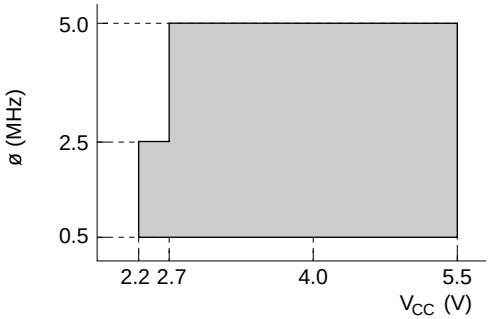
• Active mode (high and medium speeds)
• Sleep mode



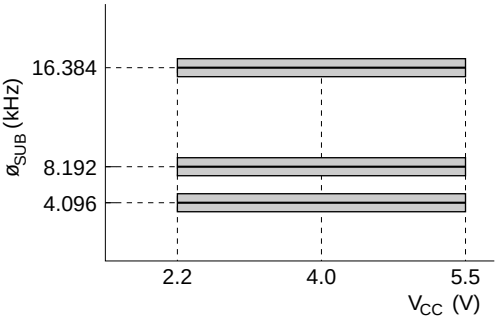
• All operating modes

Note: * The oscillation start voltage is 2.5 V.

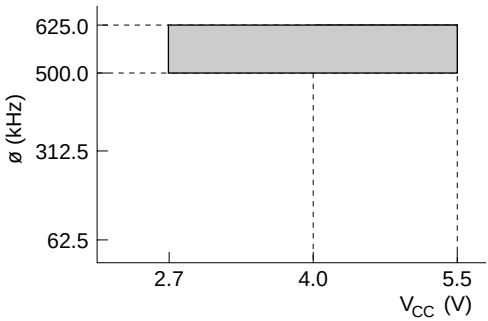
Power Supply Voltage vs. Clock Frequency Range



- Active mode (high speed)
- Sleep mode (except CPU)

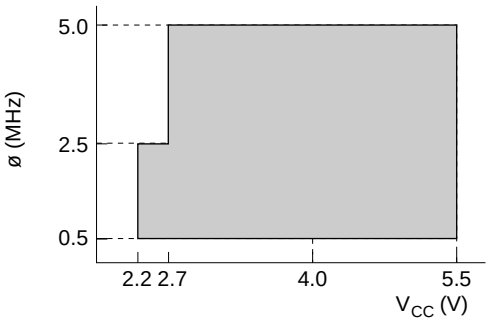


- Subactive mode
- Subsleep mode (except CPU)
- Watch mode (except CPU)

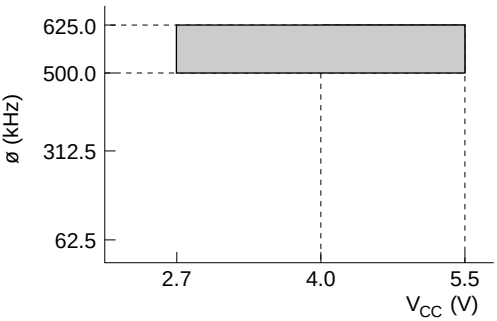


- Active mode (medium speed)

Analog Power Supply Voltage vs. A/D Converter Operating Range



- Active (high speed) mode
- Sleep mode



- Active (medium speed) mode

14.2.2 DC Characteristics

Table 14.2 lists the DC characteristics.

Table 14.2 DC Characteristics

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Note
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{WKP}_0$ to $\overline{WKP}_7$,	0.8 V_{CC}	—	V_{CC} +0.3	V	V_{CC} = 2.7 V to 5.5 V	
		$\overline{IRQ}_0$ to $\overline{IRQ}_4$, TMIF, TMIG, SCK_1, SCK_3, $\overline{ADTRG}$	0.9 V_{CC}	—	V_{CC} +0.3			
		SI_1, RXD	0.7 V_{CC}	—	V_{CC} +0.3	V	V_{CC} = 2.7 V to 5.5 V	
			0.8 V_{CC}	—	V_{CC} +0.3			
		OSC_1	V_{CC} –0.5	—	V_{CC} +0.3	V	V_{CC} = 2.7 V to 5.5 V	
			V_{CC} –0.3	—	V_{CC} +0.3			
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$,	0.7 V_{CC}	—	V_{CC} +0.3	V	V_{CC} = 2.7 V to 5.5 V	
		$P7_0$ to $P7_7$, $P8_0$ to $P8_7$, PA_1 to PA_3, PB_6, PB_7	0.8 V_{CC}	—	V_{CC} +0.3			
Input low voltage	V_{IL}	$\overline{RES}$, $\overline{WKP}_0$ to $\overline{WKP}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_4$,	–0.3	—	0.2 V_{CC}	V	V_{CC} = 2.7 V to 5.5 V	
		TMIF, TMIG, SCK_1, SCK_3, $\overline{ADTRG}$	–0.3	—	0.1 V_{CC}			
		SI_1, RXD	–0.3	—	0.3 V_{CC}	V	V_{CC} = 2.7 V to 5.5 V	
			–0.3	—	0.2 V_{CC}			
		OSC_1	–0.3	—	0.5	V	V_{CC} = 2.7 V to 5.5 V	
			–0.3	—	0.3			

Note: Connect pin TEST to V_{SS}.

Table 14.2 DC Characteristics (cont)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Note
Input low voltage	V_{IL}	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$,	–0.3	—	0.3 V_{CC}	V	V_{CC} = 2.7 V to 5.5 V	
		$P7_0$ to $P7_7$, $P8_0$ to $P8_7$, PA_1 to PA_3, PB_6, PB_7	–0.3	—	0.2 V_{CC}			
Output high voltage	V_{OH}	$P1_0$ to $P1_7$, $P2_0$ to $P2_6$, $P5_0$ to $P5_7$,	V_{CC} – 1.0	—	—	V	V_{CC} = 4.0 V to 5.5 V $-I_{OH}$ = 1.0 mA	
		$P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$,	V_{CC} – 0.5	—	—		V_{CC} = 4.0 V to 5.5 V $-I_{OH}$ = 0.5 mA	
		PA_1 to PA_3,	V_{CC} – 0.5	—	—		$-I_{OH}$ = 0.1 mA	
Output low voltage	V_{OL}	$P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, PA_1 to PA_3,	—	—	0.5	V	I_{OL} = 0.4 mA	
		$P1_0$ to $P1_7$, $P2_0$ to $P2_6$	—	—	1.5		V_{CC} = 4.0 V to 5.5 V I_{OL} = 10 mA	
			—	—	0.6		V_{CC} = 4.0 V to 5.5 V I_{OL} = 1.6 mA	
			—	—	0.5		I_{OL} = 0.4 mA	

Table 14.2 DC Characteristics (cont)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Note
Input leakage current	$\|I_{IL}\|$	$\overline{RES}$, $P2_7$	—	—	20	µA	V_{IN} = 0.5 V to V_{CC} – 0.5 V	3
			—	—	1			2
		OSC_1, $P1_0$ to $P1_7$, $P2_0$ to $P2_6$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, PA_1 to PA_3, PB_6, PB_7	—	—	1	µA	V_{IN} = 0.5 V to V_{CC} – 0.5 V	
Pull-up MOS current	$-I_P$	$P1_0$ to $P1_7$, $P2_0$ to $P2_6$	50	—	300	µA	V_{CC} = 5 V, V_{IN} = 0 V	
		$P5_0$ to $P5_7$, $P6_0$ to $P6_7$	—	35	—		V_{CC} = 2.7 V, V_{IN} = 0 V	Reference value
Input capacitance	C_{IN}	All input pins except power supply pins	—	—	15	pF	f = 1 MHz, V_{IN} = 0 V, T_a = 25°C	
		$\overline{RES}$	—	—	80			3
		$P2_7$	—	—	50			

Notes: 2. Applies to HD6433622, HD6433623, HD6433624, HD6433625, HD6433626, and HD6433627.
3. Applies to HD6473627.

Table 14.2 DC Characteristics (cont)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Notes
Active mode current dissipation	I_{OPE1}	V_{CC}	—	6	9	mA	Active mode (high speed), V_{CC} = 5 V, f_{osc} = 10 MHz	4, 5
	I_{OPE2}	V_{CC}	—	2	4	mA	Active mode (medium speed), V_{CC} = 5 V, f_{osc} = 10 MHz	4, 5
Sleep mode current dissipation	I_{SLEEP}	V_{CC}	—	3.5	6	mA	V_{CC} = 5 V, f_{osc} = 10 MHz	4, 5
Subactive mode current dissipation	I_{SUB}	V_{CC}	—	12	30	µA	V_{CC} = 2.7 V, 32-kHz crystal oscillator ($ø_{SUB}$ = øw/2)	4, 5
			—	6	—	µA	V_{CC} = 2.7 V, 32-kHz crystal oscillator ($ø_{SUB}$ = øw/8)	Reference value 4, 5
Subsleep mode current dissipation	I_{SUBSP}	V_{CC}	—	5	15	µA	V_{CC} = 2.7 V, 32-kHz crystal oscillator ($ø_{SUB}$ = øw/2)	4, 5
Watch mode current dissipation	I_{WATCH}	V_{CC}	—	—	6	µA	V_{CC} = 2.7 V, 32-kHz crystal oscillator	4, 5
Standby mode current dissipation	I_{STBY}	V_{CC}	—	—	5	µA	32-kHz crystal oscillator not used	4, 5
RAM data retaining voltage	V_{RAM}	V_{CC}	2	—	—	V		

Notes: 4. Pin states during current measurement are shown below.

Mode	Other Pins	Internal State	Oscillator Pins
Active mode (high and mediumspeed)	V_{CC}	Operates	System clock oscillator: Crystal Subclock oscillator: Pin X_1 = V_{CC}
Sleep mode	V_{CC}	Only timer operates	
Subactive mode	V_{CC}	Operates	System clock oscillator: Crystal Subclock oscillator: Crystal
Subsleep mode	V_{CC}	Only timer operates, CPU stops	
Watch mode	V_{CC}	Only time-base clock operates, CPU stops	
Standby mode	V_{CC}	CPU and timers all stop	System clock oscillator: Crystal Subclock oscillator: Pin X_1 = V_{CC}

5. Excludes current in pull-up MOS transistors and output buffers.

Table 14.2 DC Characteristics (cont)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition
Allowable output low current (per pin)	I_{OL}	Output pins except in ports 1 and 2	—	—	2	mA	V_{CC} = 4.0 V to 5.5 V
		Ports 1 and 2	—	—	10		V_{CC} = 4.0 V to 5.5 V
		All output pins	—	—	0.5		
Allowable output low current (total)	ΣI_{OL}	Output pins except in ports 1 and 2	—	—	40	mA	V_{CC} = 4.0 V to 5.5 V
		Ports 1 and 2	—	—	80		V_{CC} = 4.0 V to 5.5 V
		All output pins	—	—	20		
Allowable output high current (per pin)	$-I_{OH}$	All output pins	—	—	2	mA	V_{CC} = 4.0 V to 5.5 V
			—	—	0.2		
Allowable output high current (total)	$\Sigma-I_{OH}$	All output pins	—	—	15	mA	V_{CC} = 4.0 V to 5.5 V
			—	—	10		

14.2.3 AC Characteristics

Table 14.3 lists the control signal timing, and tables 14.4 and 14.5 list the serial interface timing.

Table 14.3 Control Signal Timing

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise specified.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Reference Figure
System clock oscillation frequency	f_{OSC}	OSC_1, OSC_2	2	—	10	MHz	V_{CC} = 2.7 V to 5.5 V	
			2	—	5			
OSC clock ($ø_{OSC}$) cycle time	t_{OSC}	OSC_1, OSC_2	100	—	1000	ns	V_{CC} = 2.7 V to 5.5 V	1
			200	—	1000			Figure 14.1
System clock (ø) cycle time	t_{cyc}		2	—	16	t_{OSC}		1
			—	—	2000	ns		
Subclock oscillation frequency	f_W	X_1, X_2	—	32.768	—	kHz		
Watch clock cycle time ($ø_W$)	t_W	X_1, X_2	—	30.5	—	µs		
Subclock ($ø_{SUB}$) cycle time	t_{subcyc}		2	—	8	t_W		2
Instruction cycle time			2	—	—	t_{cyc} t_{subcyc}		
Oscillation stabilization time (crystal oscillator)	t_{rc}	OSC_1, OSC_2	—	—	40	ms	V_{CC} = 4.0 V to 5.5 V	
			—	—	60		V_{CC} = 2.5 V to 5.5 V	
Oscillation stabilization time	t_{rc}	X_1, X_2	—	—	2	s		
External clock high width	t_{CPH}	OSC_1	40	—	—	ns	V_{CC} = 2.7 V to 5.5 V	Figure 14.1
			80	—	—			
External clock low width	t_{CPL}	OSC_1	40	—	—	ns	V_{CC} = 2.7 V to 5.5 V	Figure 14.1
			80	—	—			
External clock rise time	t_{CPr}		—	—	15	ns	V_{CC} = 2.7 V to 5.5 V	Figure 14.1
			—	—	20			
External clock fall time	t_{CPf}		—	—	15	ns	V_{CC} = 2.7 V to 5.5 V	Figure 14.1
			—	—	20			
Pin $\overline{RES}$ low width	t_{REL}	$\overline{RES}$	18	—	—	t_{cyc} t_{subcyc}		Figure 14.2

Notes: 1. A frequency between 1 MHz to 10 MHz is required when an external clock is input.
2. Selected with SA1 and SA0 of system clock control register 2 (SYSCR2).

Table 14.3 Control Signal Timing (cont)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise specified.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Reference Figure
Oscillation start voltage	Vstart	OSC_1, OSC_2	2.5	—	—	V		
		X_1, X_2	2.5	—	—			
Input pin high width	t_{IH}	$\overline{IRQ_0}$ to $\overline{IRQ_4}$, $\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{ADTRG}$, TMIF, TMIG	2	—	—	t_{cyc} t_{subcyc}		Figure 14.3
Input pin low width	t_{IL}	$\overline{IRQ_0}$ to $\overline{IRQ_4}$, $\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{ADTRG}$, TMIF, TMIG	2	—	—	t_{cyc} t_{subcyc}		Figure 14.3

Table 14.4 Serial Interface Timing (SCI1)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise specified.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Reference Figure
Input serial clock cycle time	t_{scyc}	SCK_1	2	—	—	t_{cyc}		Figure 14.4
Input serial clock high width	t_{SCKH}	SCK_1	0.4	—	—	t_{scyc}		Figure 14.4
Input serial clock low width	t_{SCKL}	SCK_1	0.4	—	—	t_{scyc}		Figure 14.4
Input serial clock rise time	t_{SCKr}	SCK_1	—	—	60	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.4
			—	—	80			
Input serial clock fall time	t_{SCKf}	SCK_1	—	—	60	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.4
			—	—	80			
Serial output data delay time	t_{SOD}	SO_1	—	—	200	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.4
			—	—	350			
Serial input data setup time	t_{SIS}	SI_1	200	—	—	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.4
			400	—	—			
Serial input data hold time	t_{SIH}	SI_1	200	—	—	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.4
			400	—	—			

Table 14.5 Serial Interface Timing (SCI3)

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, including subactive mode, unless otherwise specified.

Item		Symbol	Min	Typ	Max	Unit	Test Condition	Reference Figure
Input clock cycle	Asynchronous	t_{scyc}	4	—	—	t_{cyc}		Figure 14.5
	Synchronous		6	—	—			
Input clock pulse width		t_{SCKW}	0.4	—	0.6	t_{scyc}		Figure 14.5
Transmit data delay time (synchronous mode)		t_{TXD}	—	—	1	t_{cyc}	V_{CC} = 4.0 V to 5.5 V	Figure 14.6
			—	—	1			
Receive data setup time (synchronous mode)		t_{RXS}	200	—	—	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.6
			400	—	—			
Receive data hold time (synchronous mode)		t_{RXH}	200	—	—	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.6
			400	—	—			

14.2.4 A/D Converter Characteristics

Table 14.6 shows the A/D converter characteristics.

Table 14.6 A/D Converter Characteristics

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Notes
Analog input voltage	AV_{IN}	AN_6, AN_7	–0.3	—	V_{CC} + 0.3	V		
Analog input capacitance	C_{AIN}	AN_6, AN_7	—	—	30	pF		
Allowable signal source impedance	R_{AIN}		—	—	10	kΩ		
Resolution			—	—	8	bit		
Non-linearity error			—	—	±2.0	LSB		
Quantization error			—	—	±0.5	LSB		
Absolute accuracy			—	—	±2.5	LSB		
Conversion time			12.4	—	248	µs	V_{CC} = 4.5 V to 5.5 V	
			24.8	—	248	µs		

14.2.5 DTMF Characteristics

Table 14.7 lists the DTMF generator characteristics.

Table 14.7 DTMF Characteristics

V_{CC} = 2.2 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20 to +75˚C, unless otherwise specified

Item	Symbol	Applicable Pins	Min	Typ	Max	Unit	Test Condition	Notes
Reference level supply voltage	VT_{ref}	VT_{ref}	2.2	—	V_{CC} + 0.3	V		
DTMF output voltage (row)	V_{OR}	TONED	550	723	—	mVrms	VT_{ref} – GND = 2.2 V R_L = 100 kΩ	Figure 14.81
DTMF output voltage (column)	V_{OC}	TONED	567	760	—	mVrms	VT_{ref} – GND = 2.2 V R_L = 100 kΩ	Figure 14.81
DTMF output distortion	%DISDT	TONED	—	3	7	%	VT_{ref} – GND = 2.2 V R_L = 100 kΩ	Figure 14.8
DTMF output level	dB_{CR}	TONED	—	2.5	—	dB	VT_{ref} – GND = 2.2 V R_L = 100 kΩ	Figure 14.8

Notes: 1. V_{OR} and V_{OC} indicate the output voltage during single wave output.

14.3 Operation Timing

Figures 14.1 to 14.6 show operation timings.

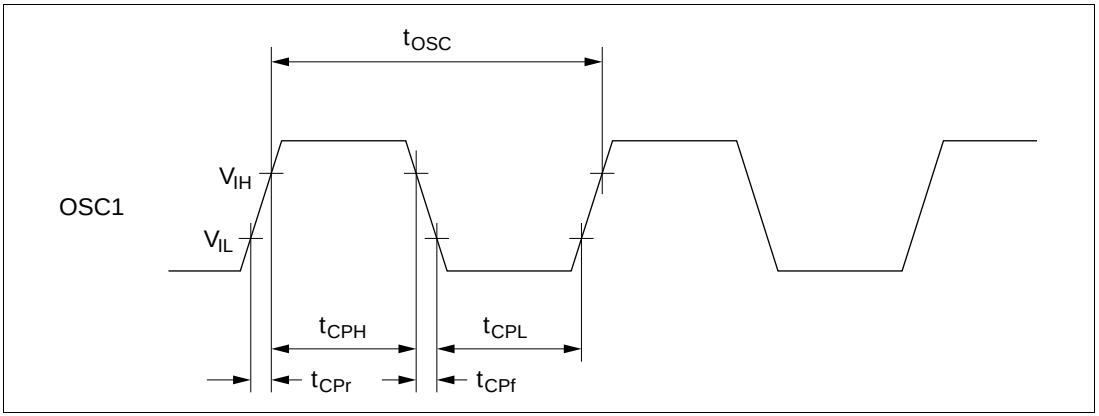


Figure 14.1 System Clock Input Timing

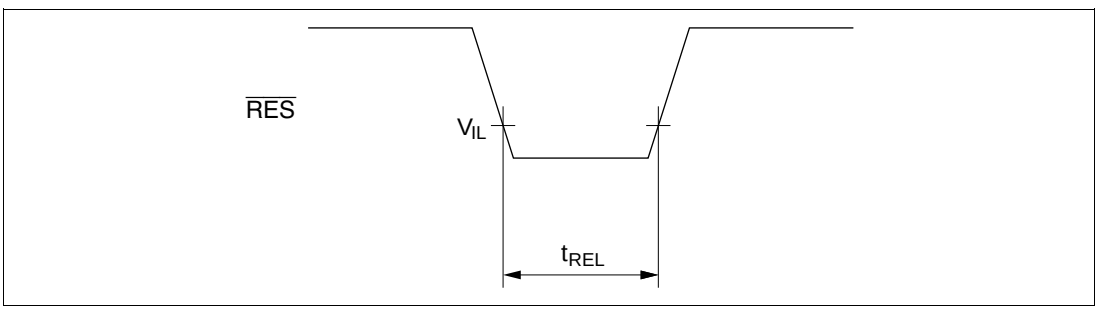


Figure 14.2 $\overline{RES}$ Low Width Timing

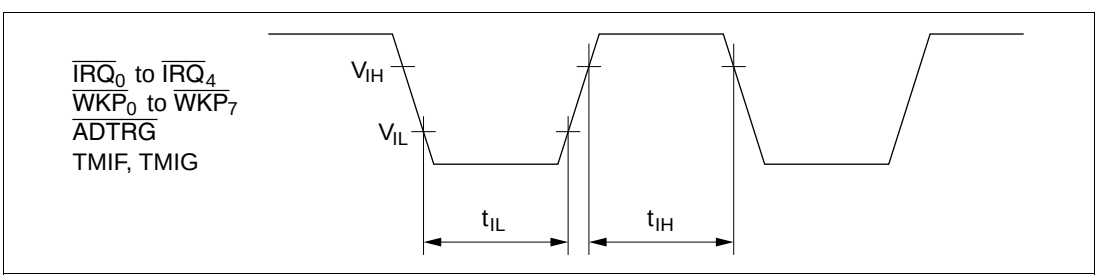
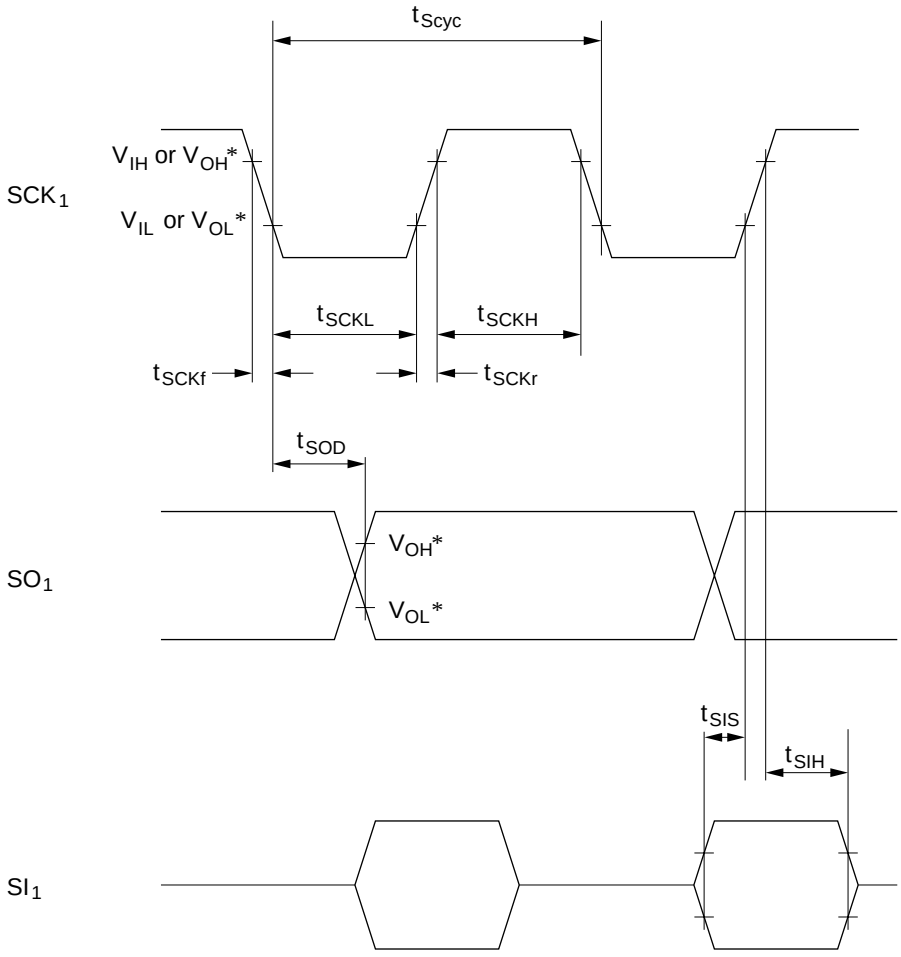


Figure 14.3 Input Timing



Notes: * Output timing reference levels
Output high: V_{OH} = 2.0 V
Output low: V_{OL} = 0.8 V
Refer to figure 14.7 for output load condition.

Figure 14.4 Serial Interface 1 Input/Output Timing

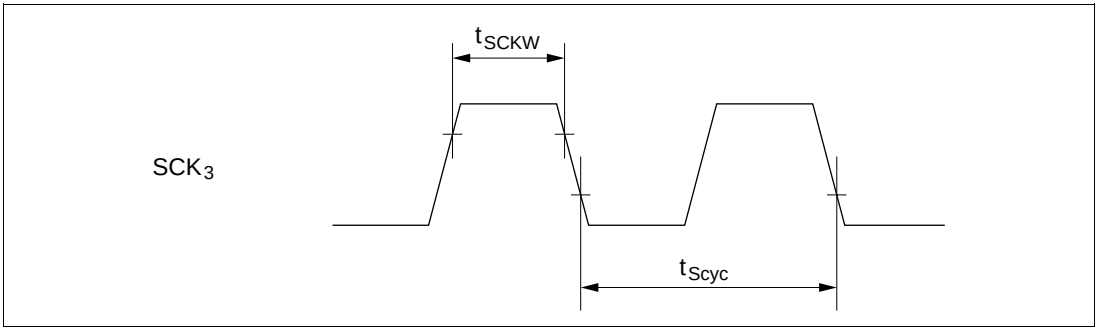
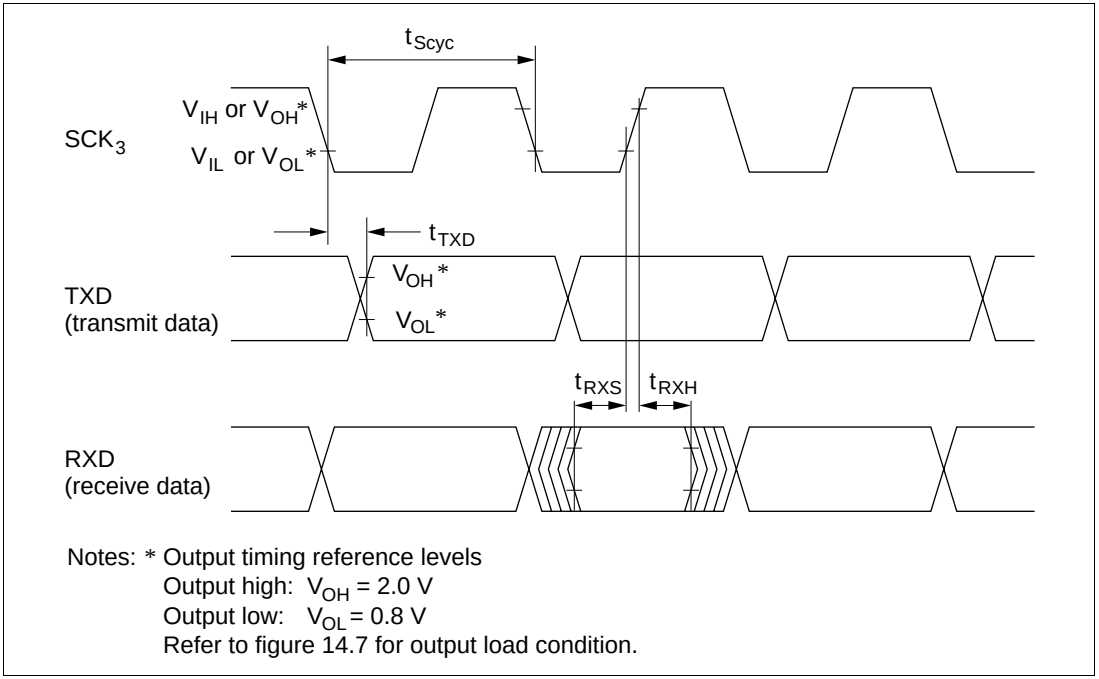


Figure 14.5 SCK_3 Input Clock Timing



Notes: * Output timing reference levels
Output high: V_{OH} = 2.0 V
Output low: V_{OL} = 0.8 V
Refer to figure 14.7 for output load condition.

Figure 14.6 Input/Output Timing of Serial Interface 3 in Synchronous Mode

14.4 Output Load Circuits

Figure 14.7 shows an output load condition.

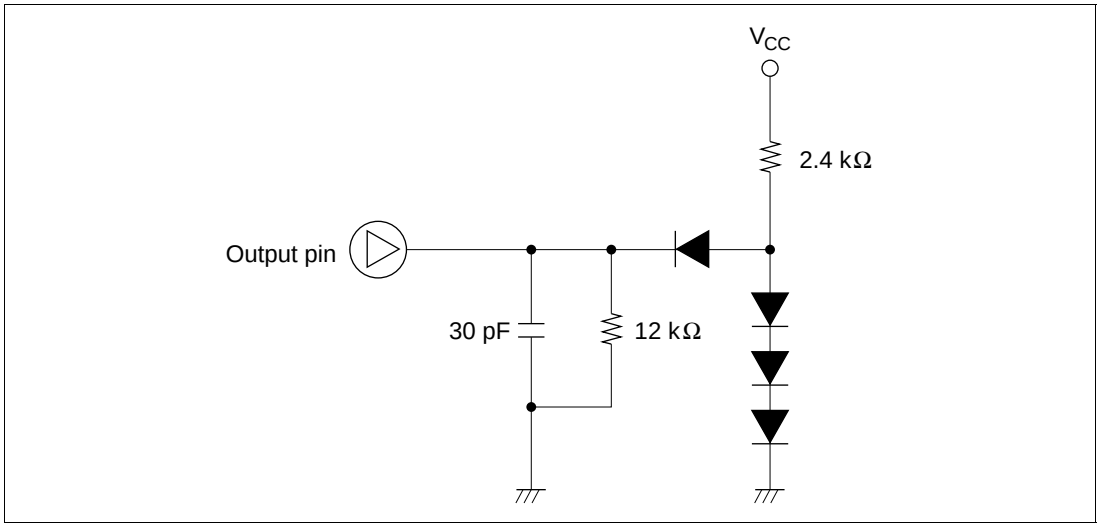


Figure 14.7 Output Load Condition

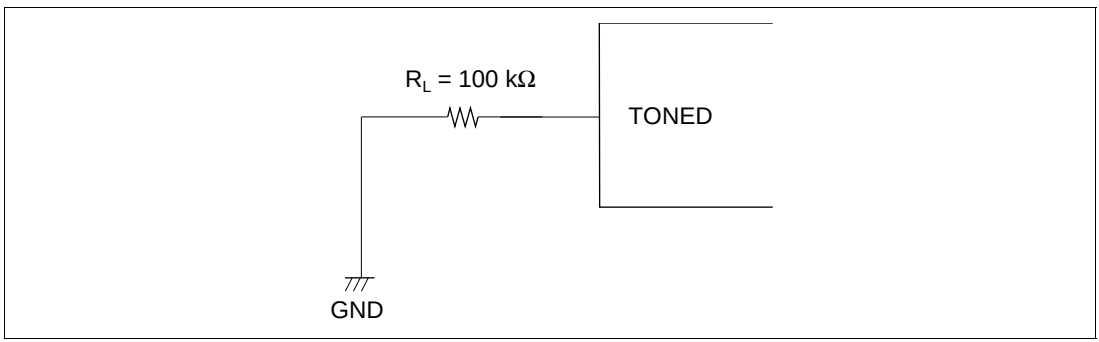


Figure 14.8 TONED Load Circuit

Appendix A CPU Instruction Set

A.1 Instructions

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
¬	Logical complement

Condition Code Notation

Symbol	
↕	Modified according to the instruction result
*	Undefined (value not guaranteed)
0	Always cleared to 0
—	Not affected by the instruction execution result

Table A.1 lists the H8/300L CPU instruction set.

Table A.1 Instruction Set

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
MOV.B #xx:8, Rd	B	#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
MOV.B Rs, Rd	B	Rs8 → Rd8		2								—	—	↕	↕	0	—	2
MOV.B @Rs, Rd	B	@Rs16 → Rd8			2							—	—	↕	↕	0	—	4
MOV.B @(d:16, Rs), Rd	B	@(d:16, Rs16) → Rd8				4						—	—	↕	↕	0	—	6
MOV.B @Rs+, Rd	B	@Rs16 → Rd8 Rs16+1 → Rs16					2					—	—	↕	↕	0	—	6
MOV.B @aa:8, Rd	B	@aa:8 → Rd8						2				—	—	↕	↕	0	—	4
MOV.B @aa:16, Rd	B	@aa:16 → Rd8						4				—	—	↕	↕	0	—	6
MOV.B Rs, @Rd	B	Rs8 → @Rd16			2							—	—	↕	↕	0	—	4
MOV.B Rs, @(d:16, Rd)	B	Rs8 → @(d:16, Rd16)				4						—	—	↕	↕	0	—	6
MOV.B Rs, @–Rd	B	Rd16–1 → Rd16 Rs8 → @Rd16					2					—	—	↕	↕	0	—	6
MOV.B Rs, @aa:8	B	Rs8 → @aa:8						2				—	—	↕	↕	0	—	4
MOV.B Rs, @aa:16	B	Rs8 → @aa:16						4				—	—	↕	↕	0	—	6
MOV.W #xx:16, Rd	W	#xx:16 → Rd	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

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									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	
PUSH Rs	W	SP–2 → SP Rs16 → @SP					2					—	—	↕	↕	0	—	6
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–Rd → Rd		2								—	↕	↕	↕	↕	↕	2
CMP.B #xx:8, Rd	B	Rd8–#xx:8	2									—	↕	↕	↕	↕	↕	2
CMP.B Rs, Rd	B	Rd8–Rs8		2								—	↕	↕	↕	↕	↕	2
CMP.W Rs, Rd	W	Rd16–Rs16		2								—	(1)	↕	↕	↕	↕	2
MULXU.B Rs, Rd	B	Rd8 × Rs8 → Rd16		2								—	—	—	—	—	—	14

Table A.1 Instruction Set (cont)

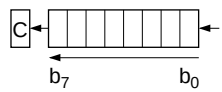
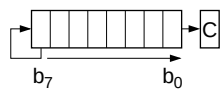
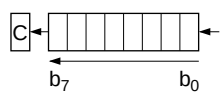
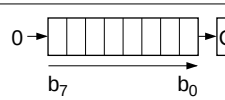
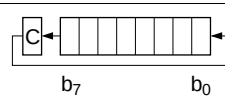
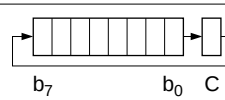
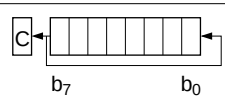
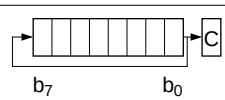
Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
DIVXU.B Rs, Rd	B	Rd16÷Rs8 → Rd16 (RdH: remainder, RdL: quotient)		2								—	—	(5)	(6)	—	—	14
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	$\overline{Rd}$ → Rd		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
ROTL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTR.B Rd	B			2								—	—	↕	↕	0	↕	2

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									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	
BSET #xx:3, Rd	B	(#xx:3 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BSET Rn, Rd	B	(Rn8 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET Rn, @Rd	B	(Rn8 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET Rn, @aa:8	B	(Rn8 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BCLR #xx:3, Rd	B	(#xx:3 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BCLR Rn, Rd	B	(Rn8 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR Rn, @Rd	B	(Rn8 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR Rn, @aa:8	B	(Rn8 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BNOT #xx:3, Rd	B	(#xx:3 of Rd8) ← $\overline{(\#xx:3 \text{ of } Rd8)}$		2								—	—	—	—	—	—	2
BNOT #xx:3, @Rd	B	(#xx:3 of @Rd16) ← $\overline{(\#xx:3 \text{ of } @Rd16)}$			4							—	—	—	—	—	—	8
BNOT #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← $\overline{(\#xx:3 \text{ of } @aa:8)}$						4				—	—	—	—	—	—	8
BNOT Rn, Rd	B	(Rn8 of Rd8) ← $\overline{(Rn8 \text{ of } Rd8)}$		2								—	—	—	—	—	—	2
BNOT Rn, @Rd	B	(Rn8 of @Rd16) ← $\overline{(Rn8 \text{ of } @Rd16)}$			4							—	—	—	—	—	—	8
BNOT Rn, @aa:8	B	(Rn8 of @aa:8) ← $\overline{(Rn8 \text{ of } @aa:8)}$						4				—	—	—	—	—	—	8
BTST #xx:3, Rd	B	$\overline{(\#xx:3 \text{ of } Rd8)} \rightarrow Z$		2								—	—	—	↕	—	—	2
BTST #xx:3, @Rd	B	$\overline{(\#xx:3 \text{ of } @Rd16)} \rightarrow Z$			4							—	—	—	↕	—	—	6
BTST #xx:3, @aa:8	B	$\overline{(\#xx:3 \text{ of } @aa:8)} \rightarrow Z$						4				—	—	—	↕	—	—	6
BTST Rn, Rd	B	$\overline{(Rn8 \text{ of } Rd8)} \rightarrow Z$		2								—	—	—	↕	—	—	2
BTST Rn, @Rd	B	$\overline{(Rn8 \text{ of } @Rd16)} \rightarrow Z$			4							—	—	—	↕	—	—	6
BTST Rn, @aa:8	B	$\overline{(Rn8 \text{ of } @aa:8)} \rightarrow Z$						4				—	—	—	↕	—	—	6

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									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	
BLD #xx:3, Rd	B	(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BLD #xx:3, @Rd	B	(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BLD #xx:3, @aa:8	B	(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BILD #xx:3, Rd	B	$(\overline{\text{#xx:3 of Rd8}}) \rightarrow$ C		2								—	—	—	—	—	↕	2
BILD #xx:3, @Rd	B	$(\overline{\text{#xx:3 of @Rd16}}) \rightarrow$ C			4							—	—	—	—	—	↕	6
BILD #xx:3, @aa:8	B	$(\overline{\text{#xx:3 of @aa:8}}) \rightarrow$ C						4				—	—	—	—	—	↕	6
BST #xx:3, Rd	B	C → (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BST #xx:3, @Rd	B	C → (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BST #xx:3, @aa:8	B	C → (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BIST #xx:3, Rd	B	$\overline{\text{C}} \rightarrow$ (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BIST #xx:3, @Rd	B	$\overline{\text{C}} \rightarrow$ (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BIST #xx:3, @aa:8	B	$\overline{\text{C}} \rightarrow$ (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BAND #xx:3, Rd	B	C∧(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BAND #xx:3, @Rd	B	C∧(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BAND #xx:3, @aa:8	B	C∧(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIAND #xx:3, Rd	B	$\text{C} \wedge (\overline{\text{#xx:3 of Rd8}}) \rightarrow$ C		2								—	—	—	—	—	↕	2
BIAND #xx:3, @Rd	B	$\text{C} \wedge (\overline{\text{#xx:3 of @Rd16}}) \rightarrow$ C			4							—	—	—	—	—	↕	6
BIAND #xx:3, @aa:8	B	$\text{C} \wedge (\overline{\text{#xx:3 of @aa:8}}) \rightarrow$ C						4				—	—	—	—	—	↕	6
BOR #xx:3, Rd	B	C∨(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BOR #xx:3, @Rd	B	C∨(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIOR #xx:3, Rd	B	$\text{C} \vee (\overline{\text{#xx:3 of Rd8}}) \rightarrow$ C		2								—	—	—	—	—	↕	2
BIOR #xx:3, @Rd	B	$\text{C} \vee (\overline{\text{#xx:3 of @Rd16}}) \rightarrow$ C			4							—	—	—	—	—	↕	6
BIOR #xx:3, @aa:8	B	$\text{C} \vee (\overline{\text{#xx:3 of @aa:8}}) \rightarrow$ C						4				—	—	—	—	—	↕	6
BXOR #xx:3, Rd	B	C⊕(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BXOR #xx:3, @Rd	B	C⊕(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIXOR #xx:3, Rd	B	$\text{C} \oplus (\overline{\text{#xx:3 of Rd8}}) \rightarrow$ C		2								—	—	—	—	—	↕	2

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Branching Condition	Addressing Mode/ Instruction Length (Bytes)									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	
BIXOR #xx:3, @Rd	B	C⊕(#xx:3 of @Rd16) → C				4							—	—	—	—	—	↕	6
BIXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C							4				—	—	—	—	—	↕	6
BRA d:8 (BT d:8)	—	PC ← PC+d:8								2			—	—	—	—	—	—	4
BRN d:8 (BF d:8)	—	PC ← PC+2								2			—	—	—	—	—	—	4
BHI d:8	—	If condition is true then PC ← PC+d:8 else next;	C ∨ Z = 0							2			—	—	—	—	—	—	4
BLS d:8	—		C ∨ Z = 1							2			—	—	—	—	—	—	4
BCC d:8 (BHS d:8)	—		C = 0							2			—	—	—	—	—	—	4
BCS d:8 (BLO d:8)	—		C = 1							2			—	—	—	—	—	—	4
BNE d:8	—		Z = 0							2			—	—	—	—	—	—	4
BEQ d:8	—		Z = 1							2			—	—	—	—	—	—	4
BVC d:8	—		V = 0							2			—	—	—	—	—	—	4
BVS d:8	—		V = 1							2			—	—	—	—	—	—	4
BPL d:8	—		N = 0							2			—	—	—	—	—	—	4
BMI d:8	—		N = 1							2			—	—	—	—	—	—	4
BGE d:8	—		N⊕V = 0							2			—	—	—	—	—	—	4
BLT d:8	—		N⊕V = 1							2			—	—	—	—	—	—	4
BGT d:8	—		Z ∨ (N⊕V) = 0							2			—	—	—	—	—	—	4
BLE d:8	—		Z ∨ (N⊕V) = 1							2			—	—	—	—	—	—	4
JMP @Rn	—	PC ← Rn16				2							—	—	—	—	—	—	4
JMP @aa:16	—	PC ← aa:16							4				—	—	—	—	—	—	6
JMP @@aa:8	—	PC ← @aa:8									2		—	—	—	—	—	—	8
BSR d:8	—	SP–2 → SP PC → @SP PC ← PC+d:8								2			—	—	—	—	—	—	6
JSR @Rn	—	SP–2 → SP PC → @SP PC ← Rn16				2							—	—	—	—	—	—	6
JSR @aa:16	—	SP–2 → SP PC → @SP PC ← aa:16							4				—	—	—	—	—	—	8

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
JSR @@aa:8		SP–2 → SP PC → @SP PC ← @aa:8								2		—	—	—	—	—	—	8
RTS	—	PC ← @SP SP+2 → SP									2	—	—	—	—	—	—	8
RTE	—	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
SLEEP	—	Transit to sleep mode.									2	—	—	—	—	—	—	2
LDC #xx:8, CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
LDC Rs, CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
STC CCR, Rd	B	CCR → Rd8		2								—	—	—	—	—	—	2
ANDC #xx:8, CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
ORC #xx:8, CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
XORC #xx:8, CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
NOP	—	PC ← PC+2									2	—	—	—	—	—	—	2
EEPMOV	—	if R4L≠0 then Repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4L–1 → R4L Until R4L=0 else next;									4	—	—	—	—	—	—	(4)

Notes: (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 retains value prior to arithmetic operation.

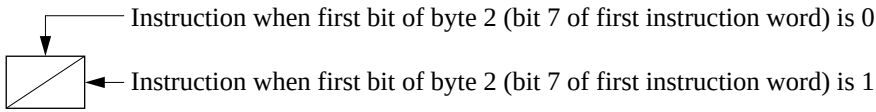
(4) The number of states required for execution is 4n + 9 (n = value of R4L).

(5) Set to 1 if the divisor is negative; otherwise cleared to 0.

(6) Set to 1 if the divisor is zero; otherwise cleared to 0.

A.2 Operation Code Map

Table A.2 is an operation code map. It shows the operation codes contained in the first byte of the instruction code (bits 15 to 8 of the first instruction word).



High \ Low	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	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU			RTS	BSR	RTE			JMP				JSR		
6	BSET	BNOT	BCLR	BTST				BST / BIST	MOV*							
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															

Note: * The PUSH and POP instructions are identical in machine language to MOV instructions.

A.3 Number of Execution States

The following describes the operation status in each instruction provided for the H8/300 L CPU, as well as a calculation of the number of execution states. Table A.4 gives the number of cycles (as the operation status) for such operations as an instruction fetch, data read/write performed during an instruction execution. Table A.3 gives the number of execution states required for each cycle (operation status). The total number of states required for the execution of an instruction can be calculated by using the following equation:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

Examples: When instruction is fetched from on-chip ROM, and an on-chip RAM is accessed.

1. BSET #0, @FF00

   From table A.4:
   I = L = 2, J = K = M = N= 0

   From table A.3:
   $S_I = 2$, $S_L = 2$

   Number of states required for execution = $2 \times 2 + 2 \times 2 = 8$

   When instruction is fetched from on-chip ROM, branch address is read from on-chip ROM, and on-chip RAM is used for stack area.

2. JSR @@ 30

   From table A.4:
   I = 2, J = K = 1, L = M = N = 0

   From table A.3:
   $S_I = S_J = S_K = 2$

   Number of states required for execution = $2 \times 2 + 1 \times 2 + 1 \times 2 = 8$

Table A.3 Number of Cycles in Each Instruction

Execution Status (Instruction Cycle)		Access Location	
		On-Chip Memory	On-Chip Peripheral Module
Instruction fetch	S_I	2	—
Branch address read	S_J		
Stack operation	S_K		
Byte data access	S_L		2 or 3*
Word data access	S_M		—
Internal operation	S_N	1	1

Note: * Depends on which on-chip module is accessed. See 2.9.1, Notes on Data Access for details.

Table A.4 Number of Cycles in Each Instruction

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
ADD	ADD.B #xx:8, Rd	1					
	ADD.B Rs, Rd	1					
	ADD.W Rs, Rd	1					
ADDS	ADDS.W #1, Rd	1					
	ADDS.W #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					

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BCLR	BCLR Rn, @Rd	2			2		
	BCLR Rn, @aa:8	2			2		
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		

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BSET	BSET Rn, @aa:8	2			2		
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					

Note: n: Initial value in R4L. The source and destination operands are accessed n + 1 times each.

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					
	MOV.B @Rs, Rd	1			1		
	MOV.B @(d:16, Rs), Rd	2			1		
	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	
MULXU	MULXU.B 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					
ROTL	ROTL.B Rd	1					
ROTR	ROTR.B Rd	1					

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
ROTXL	ROTXL.B Rd	1					
ROTXR	ROTXR.B Rd	1					
RTE	RTE	2		2			2
RTS	RTS	2		1			2
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, Rd	1					
	SUBS.W #2, Rd	1					
POP	POP Rd	1		1			2
PUSH	PUSH Rs	1		1			2
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					

B.1 I/O Registers (1)

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'90										
H'91										
H'92										
H'93										
H'94										
H'95										
H'96										
H'97										
H'98	PMR1	IRQ3	IRQ2	IRQ1	—	TMIG	TMOFH	TMOFL	TMOW	I/O ports
H'99	PMR2	IRQ0	—	POF1	NCS	SO1	SI1	SCK1	IRQ4	
H'9A	PMR6	—	—	—	—	—	TXD	—	—	
H'9B	PMR5	WKP_7	WKP_6	WKP_5	WKP_4	WKP_3	WKP_2	WKP_1	WKP_0	
H'9C	PUCR1	$PUCR1_7$	$PUCR1_6$	$PUCR1_5$	$PUCR1_4$	$PUCR1_3$	$PUCR1_2$	$PUCR1_1$	$PUCR1_0$	
H'9D	PUCR2	$PUCR2_7$	$PUCR2_6$	$PUCR2_5$	$PUCR2_4$	$PUCR2_3$	$PUCR2_2$	$PUCR2_1$	$PUCR2_0$	
H'9E	PUCR5	$PUCR5_7$	$PUCR5_6$	$PUCR5_5$	$PUCR5_4$	$PUCR5_3$	$PUCR5_2$	$PUCR5_1$	$PUCR5_0$	
H'9F	PUCR6	$PUCR6_7$	$PUCR6_6$	$PUCR6_5$	$PUCR6_4$	$PUCR6_3$	$PUCR6_2$	$PUCR6_1$	$PUCR6_0$	
H'A0	SCR1	SNC1	SNC0	—	—	CKS3	CKS2	CKS1	CKS0	SCI1
H'A1	SCSR1	—	SOL	ORER	—	—	—	—	STF	
H'A2	SDRU	SDRU7	SDRU6	SDRU5	SDRU4	SDRU3	SDRU2	SDRU1	SDRU0	
H'A3	SDRL	SDRL7	SDRL6	SDRL5	SDRL4	SDRL3	SDRL2	SDRL1	SDRL0	
H'A4										
H'A5										
H'A6										
H'A7										
H'A8	SMR	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0	SCI3
H'A9	BRR	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0	
H'AA	SCR3	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	

Legend:

SCI1: Serial communication interface 1

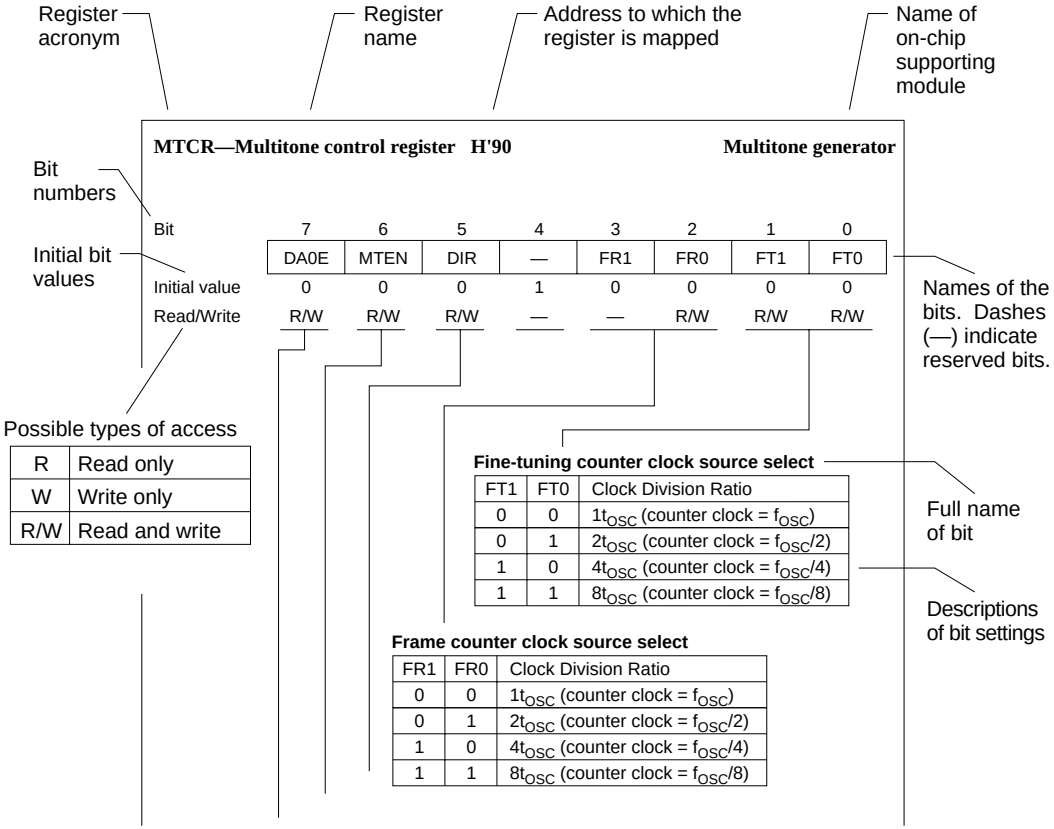
SCI3: Serial communication interface 3

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'AB	TDR	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0	SCI3
H'AC	SSR	TDRE	RDRF	OER	FER	PER	TEND	MPBR	MPBT	
H'AD	RDR	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0	
H'AE										
H'AF										
H'B0	TMA	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0	Timer A
H'B1	TCA	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0	
H'B2	DTCR	DTEN	—	CLOE	RWOE	CLF1	CLF0	RWF1	RWF0	DTMF generator
H'B3	DTLR	—	—	—	DTL4	DTL3	DTL2	DTL1	DTL0	
H'B4										
H'B5										
H'B6	TCRF	TOLH	CKSH2	CKSH1	CKSH0	TOLL	CKSL2	CKSL1	CKSL0	Timer F
H'B7	TCSRF	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL	
H'B8	TCFH	TCFH7	TCFH6	TCFH5	TCFH4	TCFH3	TCFH2	TCFH1	TCFH0	
H'B9	TCFL	TCFL7	TCFL6	TCFL5	TCFL4	TCFL3	TCFL2	TCFL1	TCFL0	
H'BA	OCRFH	OCRFH7	OCRFH6	OCRFH5	OCRFH4	OCRFH3	OCRFH2	OCRFH1	OCRFH0	
H'BB	OCRFL	OCRFL7	OCRFL6	OCRFL5	OCRFL4	OCRFL3	OCRFL2	OCRFL1	OCRFL0	
H'BC	TMG	OVFH	OVFL	OVIE	IIEGS	CCLR1	CCLR0	CKS1	CKS0	Timer G
H'BD	ICRGF	ICRGF7	ICRGF6	ICRGF5	ICRGF4	ICRGF3	ICRGF2	ICRGF1	ICRGF0	
H'BE	ICRGR	ICRGR7	ICRGR6	ICRGR5	ICRGR4	ICRGR3	ICRGR2	ICRGR1	ICRGR0	
H'BF										
H'C0										
H'C1										
H'C2										
H'C3										
H'C4	AMR	CKS	TRGE	CKS1	—	CH3	CH2	CH1	CH0	A/D converter
H'C5	ADRR	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	
H'C6	ADSR	ADSF	—	—	—	—	—	—	—	
H'C7										
H'C8										
H'C9										
H'CA										
H'CB										
H'CC										

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'CD										
H'CE										
H'CF										
H'D0										
H'D1										
H'D2										
H'D3										I/O ports
H'D4	PDR1	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$	
H'D5	PDR2	$P2_7$	$P2_6$	$P2_5$	$P2_4$	$P2_3$	$P2_2$	$P2_1$	$P2_0$	
H'D6										
H'D7										
H'D8	PDR5	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$	I/O ports
H'D9	PDR6	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$	
H'DA	PDR7	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$	
H'DB	PDR8	$P8_7$	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$	
H'DC										
H'DD	PDRA	—	—	—	—	PA_3	PA_2	PA_1	—	
H'DE	PDRB	PB_7	PB_6	—	—	—	—	—	—	
H'DF										
H'E0										
H'E1										
H'E2										
H'E3										I/O ports
H'E4	PCR1	$PCR1_7$	$PCR1_6$	$PCR1_5$	$PCR1_4$	$PCR1_3$	$PCR1_2$	$PCR1_1$	$PCR1_0$	
H'E5	PCR2	$PCR2_7$	$PCR2_6$	$PCR2_5$	$PCR2_4$	$PCR2_3$	$PCR2_2$	$PCR2_1$	$PCR2_0$	
H'E6										
H'E7										
H'E8	PCR5	$PCR5_7$	$PCR5_6$	$PCR5_5$	$PCR5_4$	$PCR5_3$	$PCR5_2$	$PCR5_1$	$PCR5_0$	I/O ports
H'E9	PCR6	$PCR6_7$	$PCR6_6$	$PCR6_5$	$PCR6_4$	$PCR6_3$	$PCR6_2$	$PCR6_1$	$PCR6_0$	
H'EA	PCR7	$PCR7_7$	$PCR7_6$	$PCR7_5$	$PCR7_4$	$PCR7_3$	$PCR7_2$	$PCR7_1$	$PCR7_0$	
H'EB	PCR8	$PCR8_7$	$PCR8_6$	$PCR8_5$	$PCR8_4$	$PCR8_3$	$PCR8_2$	$PCR8_1$	$PCR8_0$	
H'EC										
H'ED	PCRA	—	—	—	—	$PCRA_3$	$PCRA_2$	$PCRA_1$	—	

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'EE										
H'EF										
H'F0	SYSCR1	SSBY	STS2	STS1	STS0	LSON	—	—	—	System control
H'F1	SYSCR2	—	—	—	NESEL	DTON	MSON	SA1	SA0	
H'F2	IEGR	—	—	—	IEG4	IEG3	IEG2	IEG1	IEG0	
H'F3	IENR1	IENTA	IENS1	IENWP	IEN4	IEN3	IEN2	IEN1	IEN0	
H'F4	IENR2	IENDT	IENAD	—	IENTG	IENTFH	IENTFL	—	—	
H'F5										
H'F6	IRR1	IRRTA	IRRS1	—	IRRI4	IRRI3	IRRI2	IRRI1	IRRI0	System control
H'F7	IRR2	IRRDT	IRRAD	—	IRRTG	IRRTFH	IRRTFL	—	—	
H'F8										
H'F9	IWPR	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0	System control
H'FA										
H'FB										
H'FC										
H'FD										
H'FE										
H'FF										

B.2 I/O Registers (2)



Bit	7	6	5	4	3	2	1	0
	IRQ3	IRQ2	IRQ1	—	TMIG	TMOFH	TMOFL	TMOW
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

$P1_0$/TMOW pin function switch

0	Functions as $P1_0$ I/O pin
1	Functions as TMOW output pin

$P1_1$/TMOFL pin function switch

0	Functions as $P1_1$ I/O pin
1	Functions as TMOFL output pin

$P1_2$/TMOFH pin function switch

0	Functions as $P1_2$ I/O pin
1	Functions as TMOFH output pin

$P1_3$/TMIG pin function switch

0	Functions as $P1_3$ I/O pin
1	Functions as TMIG input pin

$P1_5/\overline{IRQ_1}$ pin function switch

0	Functions as $P1_5$ I/O pin
1	Functions as $\overline{IRQ_1}$ input pin

$P1_6/\overline{IRQ_2}$ pin function switch

0	Functions as $P1_6$ I/O pin
1	Functions as $\overline{IRQ_2}$ input pin

$P1_7/\overline{IRQ_3}$/TMIF pin function switch

0	Functions as $P1_7$ I/O pin
1	Functions as $\overline{IRQ_3}$/TMIF input pin

Bit	7	6	5	4	3	2	1	0
	IRQ0	—	POF1	NCS	SO1	SI1	SCK1	IRQ4
Initial value	0	1	0	0	0	0	0	0
Read/Write	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W

$P2_0$/$\overline{IRQ_4}$/$\overline{ADTRG}$ pin function switch

0	Functions as $P2_0$ I/O pin
1	Functions as $\overline{IRQ_4}$ /$\overline{ADTRG}$ input pin

$P2_1$/SCK_1 pin function switch

0	Functions as $P2_1$ I/O pin
1	Functions as SCK_1 I/O pin

$P2_2$/SI_1 pin function switch

0	Functions as $P2_2$ I/O pin
1	Functions as SI_1 input pin

$P2_3$/SO_1 pin function switch

0	Functions as $P2_3$ I/O pin
1	Functions as SO_1 output pin

TMIG noise canceller select

0	Noise canceller function not selected
1	Noise canceller function selected

$P2_3$/SO_1 pin PMOS control

0	CMOS output
1	NMOS open drain output

$P2_7$/$\overline{IRQ_0}$ pin function switch

0	Functions as $P2_7$ input pin
1	Functions as $\overline{IRQ_0}$ input pin

PMR6—Port mode register 6 **H'9A** **I/O ports**

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

$P2_6$/TXD pin function switch

0	Functions as $P2_6$ I/O pin
1	Functions as TXD output pin

PMR5—Port mode register 5 **H'9B** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	WKP_7	WKP_6	WKP_5	WKP_4	WKP_3	WKP_2	WKP_1	WKP_0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

$P5_n$/$\overline{WKP_n}$ pin function switch

0	Functions as $P5_n$ I/O pin
1	Functions as $\overline{WKP_n}$ input pin

(n = 7 to 0)

PUCR1—Port pull-up control register 1 **H'9C** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PUCR1_7$	$PUCR1_6$	$PUCR1_5$	$PUCR1_4$	$PUCR1_3$	$PUCR1_2$	$PUCR1_1$	$PUCR1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR2—Port pull-up control register 2 **H'9D** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PUCR2_7$	$PUCR2_6$	$PUCR_5$	$PUCR_4$	$PUCR_3$	$PUCR_2$	$PUCR_1$	$PUCR_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR5—Port pull-up control register 5 **H'9E** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PUCR5_7$	$PUCR5_6$	$PUCR5_5$	$PUCR5_4$	$PUCR5_3$	$PUCR5_2$	$PUCR5_1$	$PUCR5_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR6—Port pull-up control register 6 **H'9F** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PUCR6_7$	$PUCR6_6$	$PUCR6_5$	$PUCR6_4$	$PUCR6_3$	$PUCR6_2$	$PUCR6_1$	$PUCR6_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
	SNC1	SNC0	—	—	CKS3	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock select

Bit 2	Bit 1	Bit 0	Prescaler Division	Transfer Clock Cycle	
				Synchronous	
CKS2	CKS1	CKS0		ø = 5 MHz	ø = 2.5 MHz
0	0	0	ø/1024	204.8 µs	409.6 µs
		1	ø/256	51.2 µs	102.4 µs
	1	0	ø/64	12.8 µs	25.6 µs
		1	ø/32	6.4 µs	12.8 µs
1	0	0	ø/16	3.2 µs	6.4 µs
		1	ø/8	1.6 µs	3.2 µs
	1	0	ø/4	0.8 µs	1.6 µs
		1	ø/2	—	0.8 µs

Clock source select

0	Clock source is prescaler S, and pin SCK_1 is output pin
1	Clock source is external clock, and pin SCK_1 is input pin

Operation mode select

0	0	8-bit synchronous mode
	1	16-bit synchronous mode
1	0	Continuous clock output mode
	1	Reserved

SCSR1—Serial control/status register 1 **H'A1** **SCI1**

Bit	7	6	5	4	3	2	1	0
	—	SOL	ORER	—	—	—	—	STF
Initial value	1	0	0	1	1	1	0	0
Read/Write	—	R/W	R/(W)*	—	—	—	R	R/W

Start flag

0	Read	Indicates that transfer is stopped
	Write	Invalid
1	Read	Indicates transfer in progress
	Write	Starts a transfer operation

Overrun error flag

0	[Clearing condition] After reading 1, cleared by writing 0
1	[Setting condition] Set if a clock pulse is input after transfer is complete, when an external clock is used

Extended data bit

0	Read	SO_1 pin output level is low
	Write	SO_1 pin output level changes to low
1	Read	SO_1 pin output level is high
	Write	SO_1 pin output level changes to high

Note: * Only a write of 0 for flag clearing is possible.

SDRU—Serial data register U **H'A2** **SCI1**

Bit	7	6	5	4	3	2	1	0
	SDRU7	SDRU6	SDRU5	SDRU4	SDRU3	SDRU2	SDRU1	SDRU0
Initial value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Stores transmit and receive data

8-bit transfer mode: Not used
16-bit transfer mode: Upper 8 bits of data

SDRL—Serial data register L **H'A3** **SCI1**

Bit	7	6	5	4	3	2	1	0
	SDRL7	SDRL6	SDRL5	SDRL4	SDRL3	SDRL2	SDRL1	SDRL0
Initial value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Stores transmit and receive data

8-bit transfer mode: 8-bit data
16-bit transfer mode: Lower 8 bits of data

SMR—Serial mode register **H'A8** **SCI3**

Bit	7	6	5	4	3	2	1	0
	COM	CHR	PE	PM	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

Clock select 0, 1

0	0	ø clock
	1	ø/4 clock
1	0	ø/16 clock
	1	ø/64 clock

Multiprocessor mode

0	Multiprocessor communication function disabled
1	Multiprocessor communication function enabled

Stop bit length

0	1 stop bit
1	2 stop bits

Parity mode

0	Even parity
1	Odd parity

Parity enable

0	Parity bit adding and checking disabled
1	Parity bit adding and checking enabled

Character length

0	8-bit data
1	7-bit data

Communication mode

0	Asynchronous mode
1	Synchronous mode

Bit	7	6	5	4	3	2	1	0
	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
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

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

Clock enable

Bit 1	Bit 0	Description		
CKE1	CKE0	Communication Mode	Clock Source	SCK_3 Pin Function
0	0	Asynchronous	Internal clock	I/O port
		Synchronous	Internal clock	Serial clock output
	1	Asynchronous	Internal clock	Clock output
		Synchronous	Reserved	Reserved
1	0	Asynchronous	External clock	Clock output
		Synchronous	External clock	Serial clock input
	1	Asynchronous	Reserved	Reserved
		Synchronous	Reserved	Reserved

Transmit end interrupt enable

0	Transmit end interrupt (TEI) disabled
1	Transmit end interrupt (TEI) enabled

Multiprocessor interrupt enable

0	Multiprocessor interrupt request disabled (ordinary receive operation) [Clearing condition] Multiprocessor bit receives a data value of 1
1	Multiprocessor interrupt request enabled Until a multiprocessor bit value of 1 is received, the receive data full interrupt (RXI) and receive error interrupt (ERI) are disabled, and serial status register (SSR) flags RDRF, FER, and OER are not set.

Receive enable

0	Receive operation disabled (RXD is a general I/O port)
1	Receive operation enabled (RXD is the receive data pin)

Transmit enable

0	Transmit operation disabled (TXD is the transmit data pin)
1	Transmit operation enabled (TXD is the transmit data pin)

Receive interrupt enable

0	Receive data full interrupt request (RXI) and receive error interrupt request (ERI) disabled
1	Receive data full interrupt request (RXI) and receive error interrupt request (ERI) enabled

Transmit interrupt enable

0	Transmit data empty interrupt request (TXI) disabled
1	Transmit data empty interrupt request (TXI) enabled

Bit	7	6	5	4	3	2	1	0
	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
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

Data to be transferred to TSR

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	OER	FER	PER	TEND	MPBR	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 receive

0	Indicates reception of data in which the multiprocessor bit is 0
1	Indicates reception of data in which the multiprocessor bit is 1

Multiprocessor bit transmit

0	The multiprocessor bit in transmit data is 0
1	The multiprocessor bit in transmit data is 1

Transmit end

0	Indicates that transmission is in progress [Clearing conditions] • After reading TDRE = 1, cleared by writing 0 to TDRE. • When data is written to TDR by an instruction.
1	Indicates that a transmission has ended [Setting conditions] • When bit TE in serial control register 3 (SCR3) is 0. • If TDRE is set to 1 when the last bit of a transmitted character is sent.

Parity error

0	Indicates that data receiving is in progress or has been completed [Clearing conditions] After reading PER = 1, cleared by writing 0
1	Indicates that a parity error occurred in data receiving [Setting conditions] When the sum of 1s in received data plus the parity bit does not match the parity mode bit (PM) setting in the serial mode register (SMR)

Framing error

0	Indicates that data receiving is in progress or has been completed [Clearing conditions] After reading FER = 1, cleared by writing 0
1	Indicates that a framing error occurred in data receiving [Setting conditions] The stop bit at the end of receive data is checked and found to be 0

Overrun error

0	Indicates that data receiving is in progress or has been completed [Clearing conditions] After reading OER = 1, cleared by writing 0
1	Indicates that an overrun error occurred in data receiving [Setting conditions] When data receiving is completed while RDRF is set to 1

Receive data register full

0	Indicates there is no receive data in RDR [Clearing conditions] • After reading RDRF = 1, cleared by writing 0. • When data is read from RDR by an instruction.
1	Indicates that there is receive data in RDR [Setting conditions] When receiving ends normally, with receive data transferred from RSR to RDR

Transmit data register empty

0	Indicates that transmit data written to TDR has not been transferred to TSR [Clearing conditions] • After reading TDRE = 1, cleared by writing 0. • When data is written to TDR by an instruction.
1	Indicates that no transmit data has been written to TDR, or the transmit data written to TDR has been transferred to TSR [Setting conditions] • When bit TE in serial control register 3 (SCR3) is 0. • When data is transferred from TDR to TSR.

Note: * Only a write of 0 for flag clearing is possible.

Bit	7	6	5	4	3	2	1	0
	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TMA—Timer mode register A H'B0 Timer A

Bit	7	6	5	4	3	2	1	0
	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0
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

Clock output select

0	0	0	ø/32
		1	ø/16
	1	0	ø/8
		1	ø/4
1	0	0	$ø_W$/32
		1	$ø_W$/16
	1	0	$ø_W$/8
		1	$ø_W$/4

Internal clock select

TMA3	TMA2	TMA1	TMA0	Prescaler and Divider Ratio or Overflow Period		Function
0	0	0	0	PSS	ø/8192	Interval timer
			1	PSS	ø/4096	
		1	0	PSS	ø/2048	
			1	PSS	ø/512	
	1	0	0	PSS	ø/256	
			1	PSS	ø/128	
		1	0	PSS	ø/32	
			1	PSS	ø/8	
1	0	0	0	PSW	1 s	Time base
			1	PSW	0.5 s	
		1	0	PSW	0.25 s	
			1	PSW	0.03125 s	
	1	0	0	PSW and TCA are reset		
			1			
		1	0			
			1			

Bit	7	6	5	4	3	2	1	0
	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Count value

Bit	7	6	5	4	3	2	1	0
	DTEN	—	CLOE	RWOE	CLF1	CLF0	RWF1	RWF0
Initial value	0	1	0	0	0	0	0	0
Read/Write	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W

DTMF row signal output frequency 1 and 0

RWF1	RWF0	DTMF row signal output frequency
0	0	697 Hz (R1)
	1	770 Hz (R2)
1	0	852 Hz (R3)
	1	941 Hz (R4)

DTMF column signal output frequency 1 and 0

CLF1	CLF0	DTMF column signal output frequency
0	0	1209 Hz (C1)
	1	1336 Hz (C2)
1	0	1447 Hz (C3)
	1	1633 Hz (C4)

Row output enable

0	DTMF row signal output is disabled (high-impedance)
1	DTMF row signal output is enabled

Column output enable

0	DTMF column signal output is disabled (high-impedance)
1	DTMF column signal output is enabled

DTMF generator enable

0	DTMF generator is halted
1	DTMF generator operates

Bit	7	6	5	4	3	2	1	0
	—	—	—	DTL4	DTL3	DTL2	DTL1	DTL0
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

OSC clock division ratio 4 to 0

DTL4	DTL3	DTL2	DTL1	DTL0	Division Ratio	OSC Clock Frequency
0	0	0	0	0	Illegal setting	(initial value)
0	0	0	0	1	Illegal setting	
0	0	0	1	0	Illegal setting	
0	0	0	1	1	3	1.2 MHz
0	0	1	0	0	4	1.6 MHz
⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	0	0	1	25	10 MHz
1	1	0	1	*	Illegal setting	
1	1	1	*	*	Illegal setting	

Note: * Don’t care

Bit	7	6	5	4	3	2	1	0
	TOLH	CKSH2	CKSH1	CKSH0	TOLL	CKSL2	CKSL1	CKSL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Toggle output level H

0	Low level
1	High level

Clock select L

0	*	*	External event (TMIF): Rising or falling edge
1	0	0	Internal clock: ø/32
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	Internal clock: ø/2

Toggle output level L

0	Low level
1	High level

Clock select H

0	*	*	16-bit mode selected. TCFL overflow signals are counted.
1	0	0	Internal clock: ø/32
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	Internal clock: ø/2

Note: * Don't care

Bit	7	6	5	4	3	2	1	0
	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL
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

Timer overflow interrupt enable L

0	TCFL overflow interrupt disabled
1	TCFL overflow interrupt enabled

Compare match flag L

0	[Clearing condition] After reading CMFL = 1, cleared by writing 0 to CMFL
1	[Setting condition] When the TCFL value matches the OCRFL value

Timer overflow flag L

0	[Clearing condition] After reading OVFL = 1, cleared by writing 0 to OVFL
1	[Setting condition] When the value of TCFL goes from H'FF to H'00

Counter clear H

0	16-bit mode: TCF clearing by compare match disabled 8-bit mode: TCFH clearing by compare match disabled
1	16-bit mode: TCF clearing by compare match enabled 8-bit mode: TCFH clearing by compare match enabled

Timer overflow interrupt enable H

0	TCFH overflow interrupt disabled
1	TCFH overflow interrupt enabled

Counter clear L

0	TCFL clearing by compare match disabled
1	TCFL clearing by compare match enabled

Compare match flag H

0	[Clearing condition] After reading CMFH = 1, cleared by writing 0 to CMFH
1	[Setting condition] When the TCFH value matches the OCRFH value

Timer overflow flag H

0	[Clearing condition] After reading OVFH = 1, cleared by writing 0 to OVFH
1	[Setting condition] When the value of TCFH goes from H'FF to H'00

TCFH—8-bit timer counter FH **H'B8** **Timer F**

Bit	7	6	5	4	3	2	1	0
	TCFH7	TCFH6	TCFH5	TCFH4	TCFH3	TCFH2	TCFH1	TCFH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

TCFL—8-bit timer counter FL **H'B9** **Timer F**

Bit	7	6	5	4	3	2	1	0
	TCFL7	TCFL6	TCFL5	TCFL4	TCFL3	TCFL2	TCFL1	TCFL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

OCRFH—Output compare register FH **H'BA** **Timer F**

Bit	7	6	5	4	3	2	1	0
	OCRFH7	OCRFH6	OCRFH5	OCRFH4	OCRFH3	OCRFH2	OCRFH1	OCRFH0
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

OCRFL—Output compare register FL **H'BB** **Timer F**

Bit	7	6	5	4	3	2	1	0
	OCRFL7	OCRFL6	OCRFL5	OCRFL4	OCRFL3	OCRFL2	OCRFL1	OCRFL0
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

Bit	7	6	5	4	3	2	1	0
	OVFH	OVFL	OVIE	IIEGS	CCLR1	CCLR0	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W

Clock select

0	0	Internal clock: ø/64
	1	Internal clock: ø/32
1	0	Internal clock: ø/2
	1	Internal clock: $ø_W$/2

Counter clear

0	0	TCG is not cleared
	1	TCG is cleared at the falling edge of the input capture signal
1	0	TCG is cleared at the rising edge of the input capture signal
	1	TCG is cleared at both edges of the input capture signal

Input capture interrupt edge select

0	Interrupts are requested at the rising edge of the input capture signal
1	Interrupts are requested at the falling edge of the input capture signal

Timer overflow interrupt enable

0	TCG overflow interrupt disabled
1	TCG overflow interrupt enabled

Timer overflow flag L

0	[Clearing condition] After reading OVFL = 1, cleared by writing 0 to OVFL
1	[Setting condition] When the value of TCG overflows from H'FF to H'00

Timer overflow flag H

0	[Clearing condition] After reading OVFH = 1, cleared by writing 0 to OVFH
1	[Setting condition] When the value of TCG overflows from H'FF to H'00

Note: * Only a write of 0 for flag clearing is possible.

ICRGF—Input capture register GF **H'BD** **Timer G**

Bit	7	6	5	4	3	2	1	0
	ICRGF7	ICRGF6	ICRGF5	ICRGF4	ICRGF3	ICRGF2	ICRGF1	ICRGF0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRGR—Input capture register GR **H'BE** **Timer G**

Bit	7	6	5	4	3	2	1	0
	ICRGR7	ICRGR6	ICRGR5	ICRGR4	ICRGR3	ICRGR2	ICRGR1	ICRGR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
	CKS	TRGE	CKS1	—	CH3	CH2	CH1	CH0
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

Channel select

Bit 3	Bit 2	Bit 1	Bit 0	
CH3	CH2	CH1	CH0	Analog input channel
0	0	*	*	No channel selected
	1	*	*	Reserved
1	0	0	*	Reserved
		1	0	AN_6
			1	AN_7
	1	*	*	Reserved

* Don't care

External trigger select

0	Disables start of A/D conversion by external trigger
1	Enables start of A/D conversion by rising or falling edge of external trigger at pin $\overline{\text{ADTRG}}$

Clock select

Bit 7	Bit 5		Conversion Time	
CKS	CKS1	Conversion Period	ø = 2 MHz	ø = 5 MHz
0	0	Reserved	—	—
0	1	124/ø	62 µs	24.8 µs
1	0	62/ø	31 µs	12.4 µs
1	1	31/ø	15.5 µs	—*

Note: * Operation is not guaranteed if the conversion time is less than 12.4 µs.
Set bits 5 and 7 for a value of at least 12.4 µs.

ADRR—A/D result register **H'C5** **A/D converter**

Bit	7	6	5	4	3	2	1	0
	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
Initial value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
Read/Write	R	R	R	R	R	R	R	R

A/D conversion result

ADSR—A/D start register **H'C6** **A/D converter**

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

A/D start flag

0	[Read] Indicates the completion of A/D conversion [Write] Stops A/D conversion
1	[Read] Indicates A/D conversion in progress [Write] Starts A/D conversion

PDR1—Port data register 1 **H'D4** **I/O ports**

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

PDR2—Port data register 2 **H'D5** **I/O ports**

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

PDR5—Port data register 5 **H'D8** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PDR6—Port data register 6 **H'D9** **I/O ports**

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

PDR7—Port data register 7 **H'DA** **I/O ports**

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

PDR8—Port data register 8 **H'DB** **I/O ports**

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

PDRA—Port data register A **H'DD** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PA_3	PA_2	PA_1	—
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	—

PDRB—Port data register B **H'DE** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	PB_7	PB_6	—	—	—	—	—	—
Initial value								
Read/Write	R	R	—	—	—	—	—	—

PCR1—Port control register 1 **H'E4** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR1_7$	$PCR1_6$	$PCR1_5$	$PCR1_4$	$PCR1_3$	$PCR1_2$	$PCR1_1$	$PCR1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 1 input/output select

0	Input pin
1	Output pin

PCR2—Port control register 2 **H'E5** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR2_7$	$PCR2_6$	$PCR2_5$	$PCR2_4$	$PCR2_3$	$PCR2_2$	$PCR2_1$	$PCR2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 2 input/output select

0	Input pin
1	Output pin

Note: As $P2_7$ is an input-only pin, it becomes a high-impedance output when $PCR2_7$ is set to 1.

PCR5—Port control register 5 **H'E8** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR5_7$	$PCR5_6$	$PCR5_5$	$PCR5_4$	$PCR5_3$	$PCR5_2$	$PCR5_1$	$PCR5_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 5 input/output select

0	Input pin
1	Output pin

PCR6—Port control register 6 **H'E9** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR6_7$	$PCR6_6$	$PCR6_5$	$PCR6_4$	$PCR6_3$	$PCR6_2$	$PCR6_1$	$PCR6_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 6 input/output select

0	Input pin
1	Output pin

PCR7—Port control register 7 **H'EA** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR7_7$	$PCR7_6$	$PCR7_5$	$PCR7_4$	$PCR7_3$	$PCR7_2$	$PCR7_1$	$PCR7_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 7 input/output select

0	Input pin
1	Output pin

PCR8—Port control register 8 **H'EB** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR8_7$	$PCR8_6$	$PCR8_5$	$PCR8_4$	$PCR8_3$	$PCR8_2$	$PCR8_1$	$PCR8_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 8 input/output select

0	Input pin
1	Output pin

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	$PCRA_3$	$PCRA_2$	$PCRA_1$	—
Initial value	1	1	1	1	0	0	0	1
Read/Write	—	—	—	—	W	W	W	—

Port A input/output select

0	Input pin
1	Output pin

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

Low speed on flag

0	The CPU operates on the system clock (ø)
1	The CPU operates on the subclock ($ø_{SUB}$)

Standby timer select 2 to 0

0	0	0	Wait time = 8,192 states
		1	Wait time = 16,384 states
	1	0	Wait time = 32,768 states
		1	Wait time = 65,536 states
1	*	*	Wait time = 131,072 states

Software standby

0	When a SLEEP instruction is executed in active mode, a transition is made to sleep mode. When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode.
1	When a SLEEP instruction is executed in active mode, a transition is made to standby mode or watch mode. When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode.

Note: * Don’t care

Bit	7	6	5	4	3	2	1	0
	—	—	—	NESEL	DTON	MSON	SA1	SA0
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

Medium speed on flag

0	Operates in active (high-speed) mode
1	Operates in active (medium-speed) mode

Subactive mode clock select

0	0	$ø_W/8$
	1	$ø_W/4$
1	*	$ø_W/2$

Direct transfer on flag

0	When a SLEEP instruction is executed in active mode, a transition is made to standby mode, watch mode, or sleep mode. When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode or subsleep mode.
1	When a SLEEP instruction is executed in active (high-speed) mode, a direct transition is made to active (medium-speed) mode if SSBY = 0, MSON = 1, and LSON = 0, or to subactive mode if SSBY = 1, TMA3 = 1, and LSON = 1. When a SLEEP instruction is executed in active (medium-speed) mode, a direct transition is made to active (high-speed) mode if SSBY = 0, MSON = 0, and LSON = 0, or to subactive mode if SSBY = 1, TMA3 = 1, and LSON = 1. When a SLEEP instruction is executed in subactive mode, a direct transition is made to active (high-speed) mode if SSBY = 1, TMA3 = 1, LSON = 0, and MSON = 0, or to active (medium-speed) mode if SSBY = 1, TMA3 = 1, LSON = 0, and MSON = 1.

Noise elimination sampling frequency select

0	Sampling rate is $ø_{OSC}/16$
1	Sampling rate is $ø_{OSC}/4$

Note: * Don’t care

Bit	7	6	5	4	3	2	1	0
	—	—	—	IEG4	IEG3	IEG2	IEG1	IEG0
Initial value	1	1	1	0	0	0	0	0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W

IRQ_0 edge select

0	Falling edge of $\overline{IRQ_0}$ pin input is detected
1	Rising edge of $\overline{IRQ_0}$ pin input is detected

IRQ_1 edge select

0	Falling edge of $\overline{IRQ_1}$ pin input is detected
1	Rising edge of $\overline{IRQ_1}$ pin input is detected

IRQ_2 edge select

0	Falling edge of $\overline{IRQ_2}$ pin input is detected
1	Rising edge of $\overline{IRQ_2}$ pin input is detected

IRQ_3 edge select

0	Falling edge of $\overline{IRQ_3}$/TMIF pin input is detected
1	Rising edge of $\overline{IRQ_3}$/TMIF pin input is detected

IRQ_4 edge select

0	Falling edge of $\overline{IRQ_4}$/$\overline{ADTRG}$ pin input is detected
1	Rising edge of $\overline{IRQ_4}$/$\overline{ADTRG}$ pin input is detected

Bit	7	6	5	4	3	2	1	0
	IENTA	IENS1	IENWP	IEN4	IEN3	IEN2	IEN1	IEN0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IRQ_4 to IRQ_0 interrupt enable

0	Disables interrupt requests from $\overline{IRQ}_4$ to $\overline{IRQ}_0$
1	Enables interrupt requests from $\overline{IRQ}_4$ to $\overline{IRQ}_0$

Wakeup interrupt enable

0	Disables interrupt requests from $\overline{WKP}_7$ to $\overline{WKP}_0$
1	Enables interrupt requests from $\overline{WKP}_7$ to $\overline{WKP}_0$

SCI1 interrupt enable

0	Disables SCI1 interrupts
1	Enables SCI1 interrupts

Timer A interrupt enable

0	Disables timer A interrupts
1	Enables timer A interrupts

Bit	7	6	5	4	3	2	1	0
	IENDT	IENAD	—	IENTG	IENTFH	IENTFL	—	—
Initial value	0	0	0	0	0	0	0	1
Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	—

Timer FL interrupt enable

0	Disables timer FL interrupts
1	Enables timer FL interrupts

Timer FH interrupt enable

0	Disables timer FH interrupts
1	Enables timer FH interrupts

Timer G interrupt enable

0	Disables timer G interrupts
1	Enables timer G interrupts

A/D converter interrupt enable

0	Disables A/D converter interrupt requests
1	Enables A/D converter interrupt requests

Direct transfer interrupt enable

0	Disables direct transfer interrupt requests
1	Enables direct transfer interrupt requests

Bit	7	6	5	4	3	2	1	0
	IRRTA	IRRS1	—	IRRI4	IRRI3	IRRI2	IRRI1	IRRI0
Initial value	0	0	1	0	0	0	0	0
Read/Write	R/W*	R/W*	—	R/W*	R/W*	R/W*	R/W*	R/W*

IRQ_4 to IRQ_0 interrupt request flag

0	[Clearing condition] When IRRIn = 1, it is cleared by writing 0 to IRRIn.
1	[Setting condition] When pin $\overline{IRQ}_n$ is set to interrupt input and the designated signal edge is detected.

(n = 4 to 0)

SCI1 interrupt request flag

0	[Clearing condition] When IRRS1 = 1, it is cleared by writing 0
1	[Setting condition] When an SCI1 transfer is completed

Timer A interrupt request flag

0	[Clearing condition] When IRRTA = 1, it is cleared by writing 0
1	[Setting condition] When the timer A counter overflows from H'FF to H'00

Note: * Only a write of 0 for flag clearing is possible.

Bit	7	6	5	4	3	2	1	0
	IRRDT	IRRAD	—	IRRTG	IRRTFH	IRRTFL	—	—
Initial value	0	0	0	0	0	0	0	1
Read/Write	R/W*	R/W*	—	R/W*	R/W*	R/W*	—	—

Timer FL interrupt request flag

0	[Clearing condition] When IRRTFL = 1, it is cleared by writing 0
1	[Setting condition] When TCFL matches in 8-bit mode

Timer FH interrupt request flag

0	[Clearing condition] When IRRTFH = 1, it is cleared by writing 0
1	[Setting condition] When counter FH matches output compare register FH in 8-bit mode, or when 16-bit counter F (TCFL, TCFH) matches 16-bit output compare register F (OCRFL, OCRFH) in 16-bit mode

Timer G interrupt request flag

0	[Clearing condition] When IRRTG = 1, it is cleared by writing 0
1	[Setting condition] When pin TMIG is set to TMIG input and the designated signal edge is detected

A/D converter interrupt request flag

0	[Clearing condition] When IRRAD = 1, it is cleared by writing 0
1	[Setting condition] When A/D conversion is completed and ADSF is reset

Direct transfer interrupt request flag

0	[Clearing condition] When IRRDT = 1, it is cleared by writing 0
1	[Setting condition] A SLEEP instruction is executed when DTON = 1 and a direct transfer is made

Note: * Only a write of 0 for flag clearing is possible.

Bit	7	6	5	4	3	2	1	0
	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0
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*

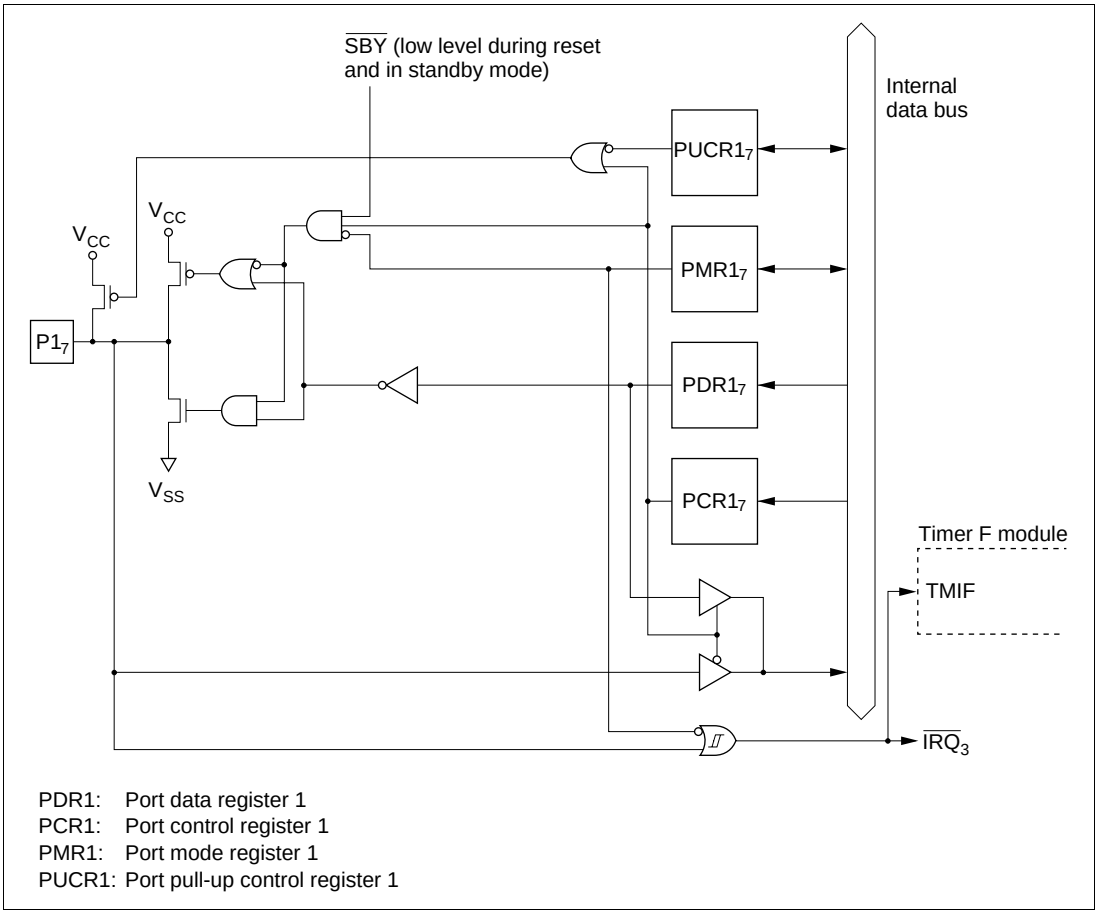
Wakeup interrupt request flag

0	[Clearing condition] When IWPFn = 1, it is cleared by writing 0.
1	[Setting condition] When pin $\overline{WKP}_n$ is set to interrupt input and a falling signal edge is detected.

(n = 7 to 0)

Note: * Only a write of 0 for flag clearing is possible.

C.1 Port 1 Block Diagrams



PDR1: Port data register 1
PCR1: Port control register 1
PMR1: Port mode register 1
PUCR1: Port pull-up control register 1

Figure C.1 (a) Port 1 Block Diagram (Pin P1$_7$)

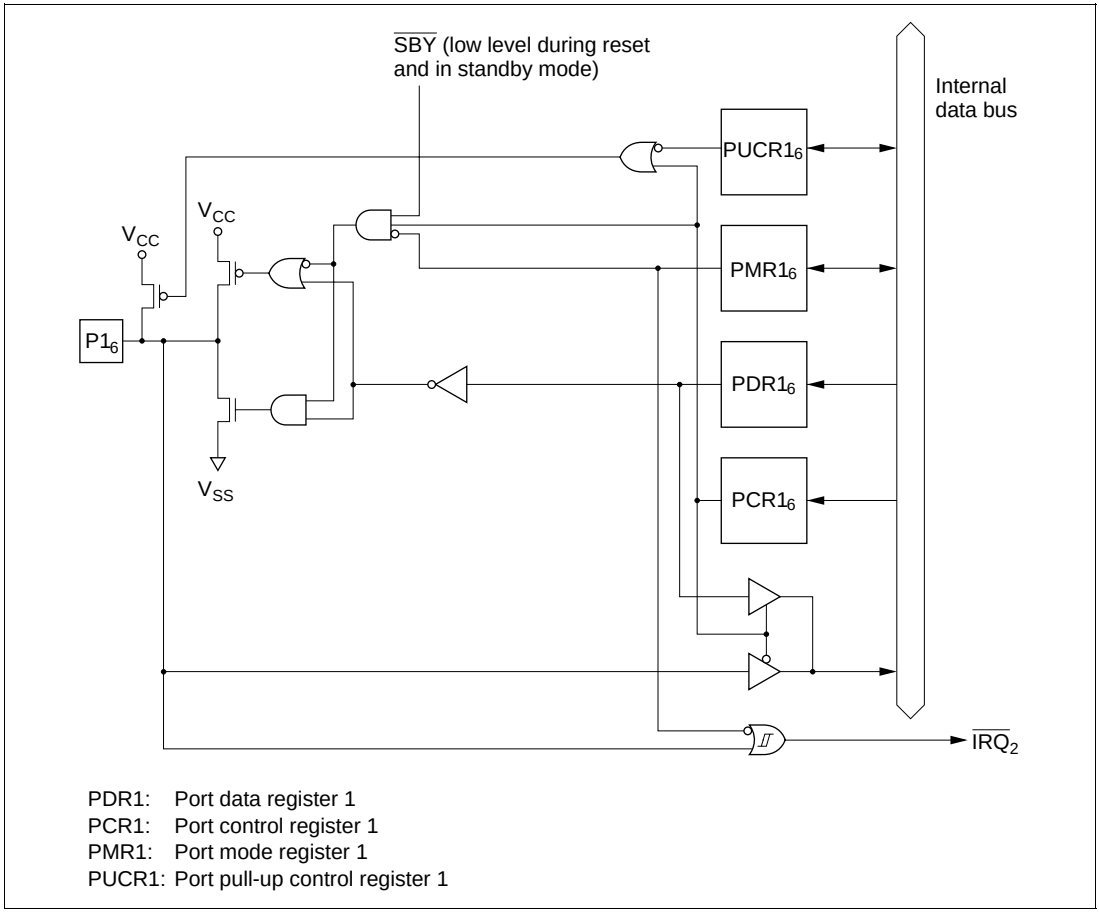
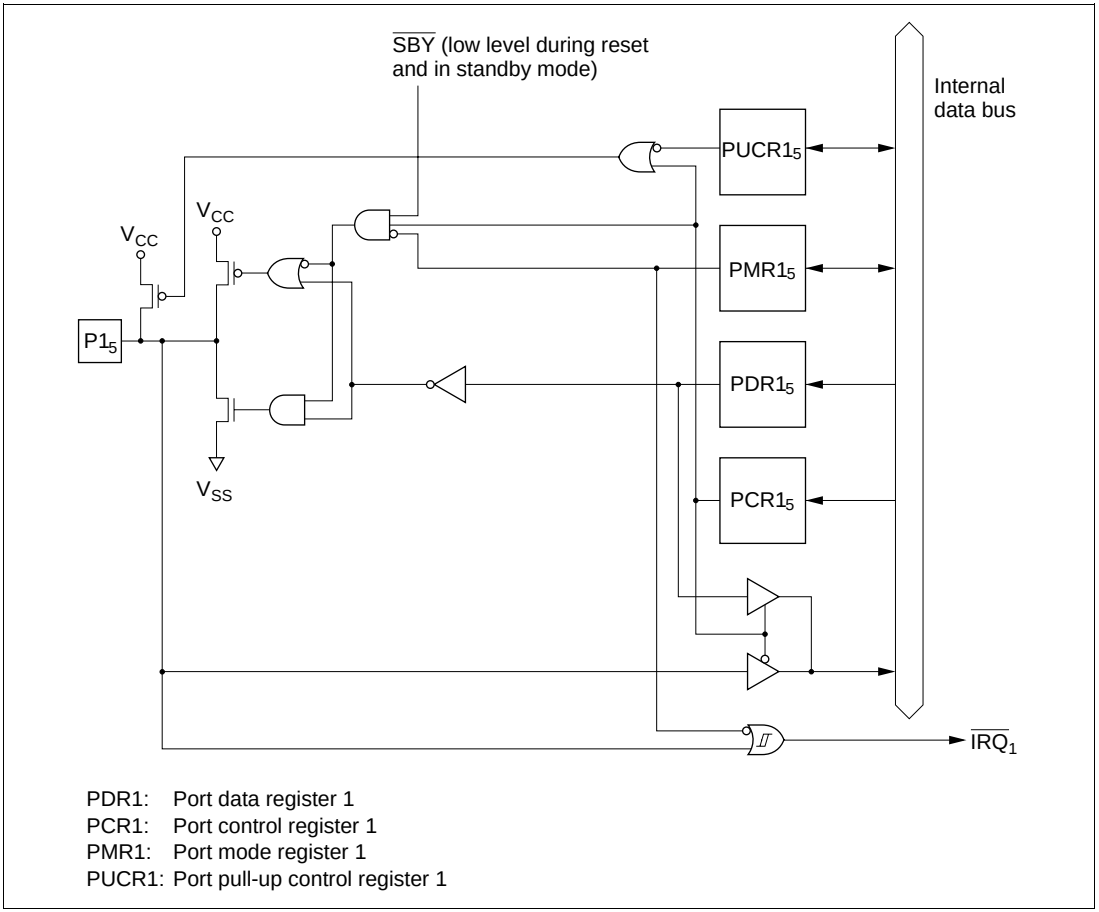
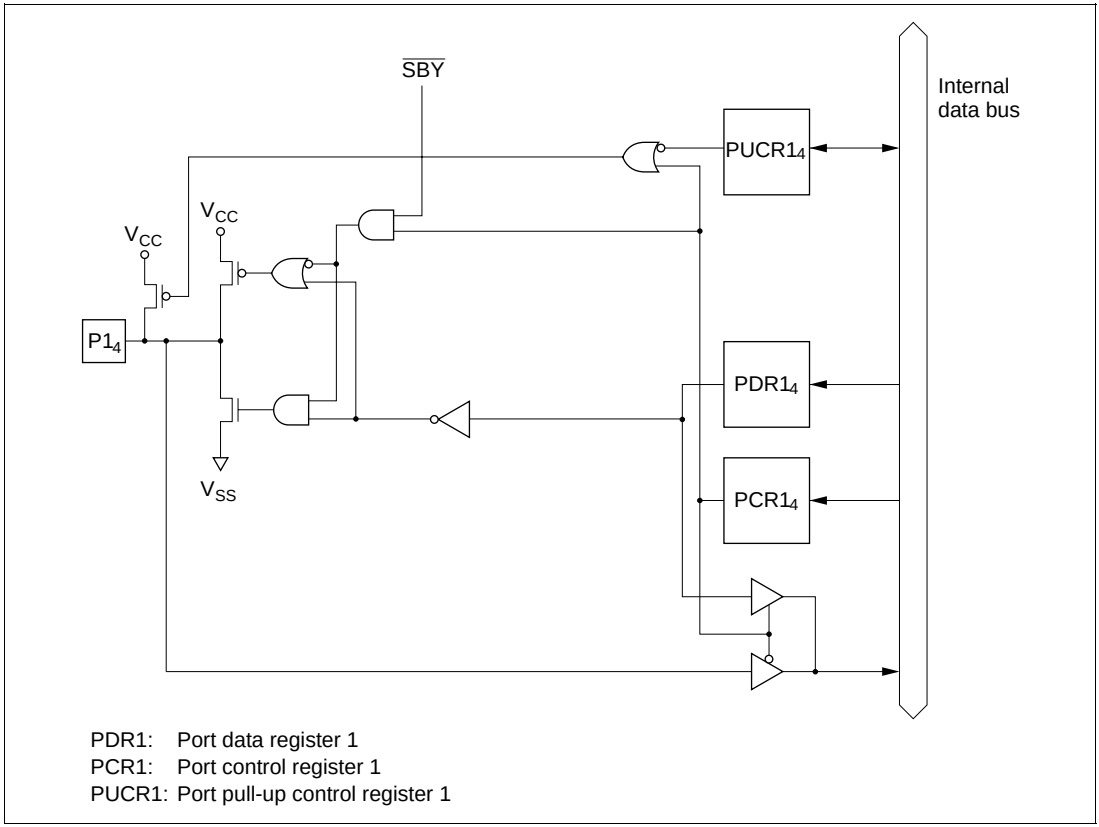


Figure C.1 (b) Port 1 Block Diagram (Pin $P1_6$)



PDR1: Port data register 1
PCR1: Port control register 1
PMR1: Port mode register 1
PUCR1: Port pull-up control register 1

Figure C.1 (c) Port 1 Block Diagram (Pin $P1_5$)



PDR1: Port data register 1
PCR1: Port control register 1
PUCR1: Port pull-up control register 1

Figure C.1 (d) Port 1 Block Diagram (Pin $P1_4$)

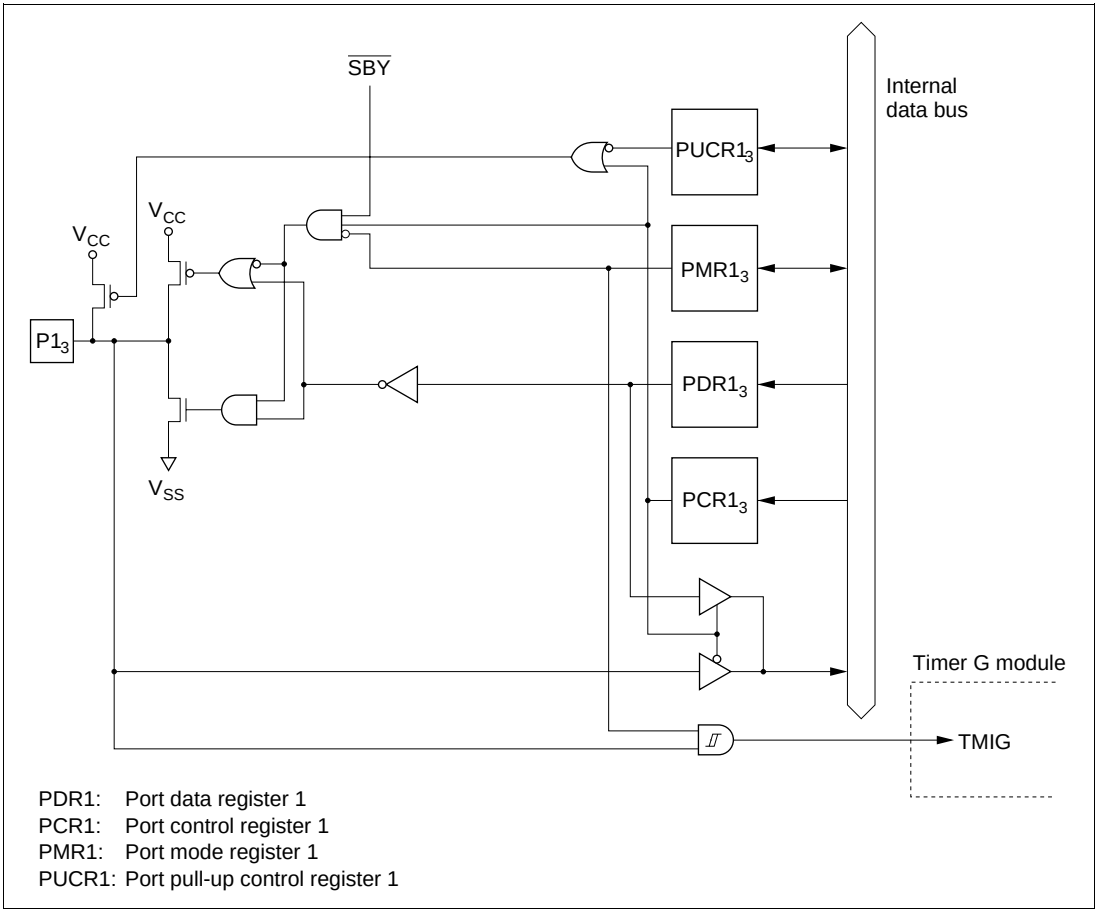


Figure C.1 (e) Port 1 Block Diagram (Pin $P1_3$)

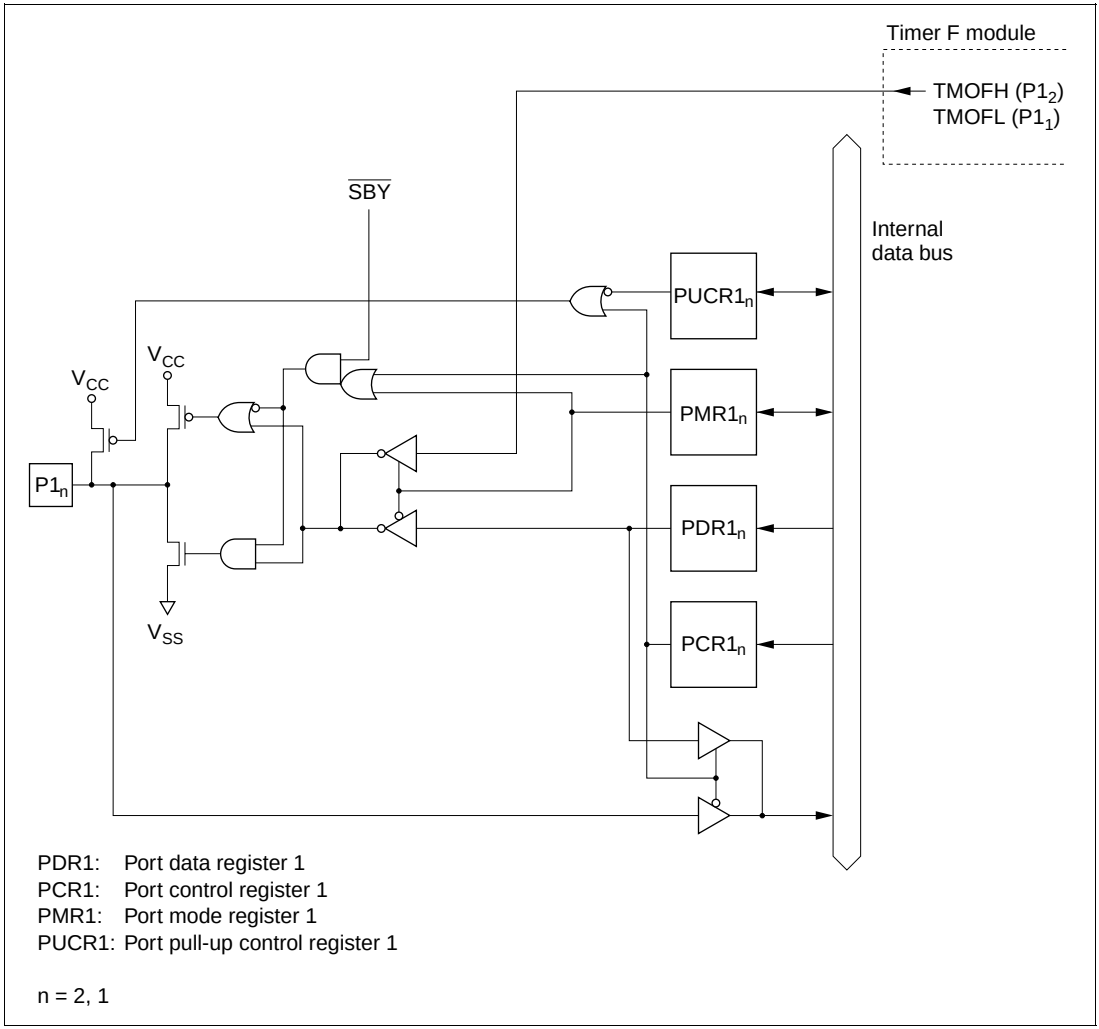


Figure C.1 (f) Port 1 Block Diagram (Pins P1$_2$ and P1$_1$)

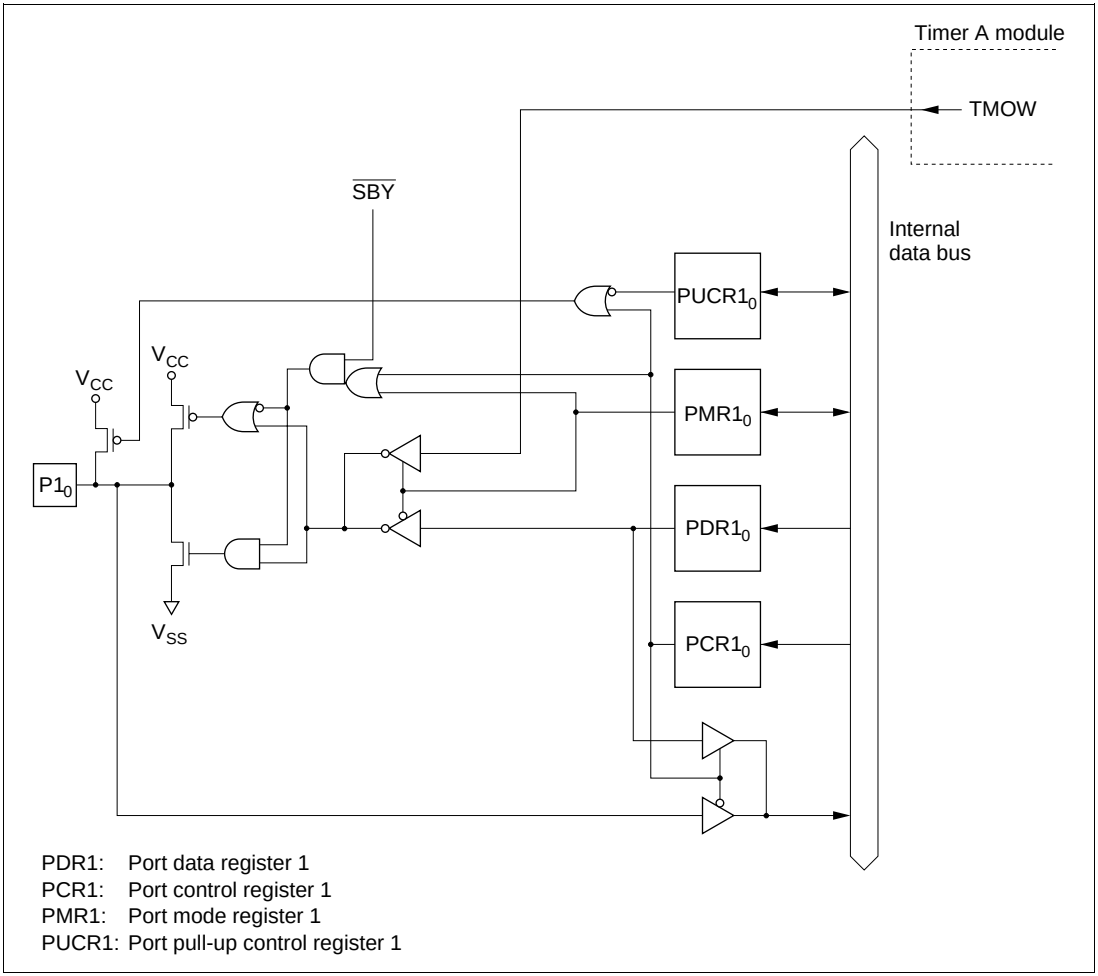
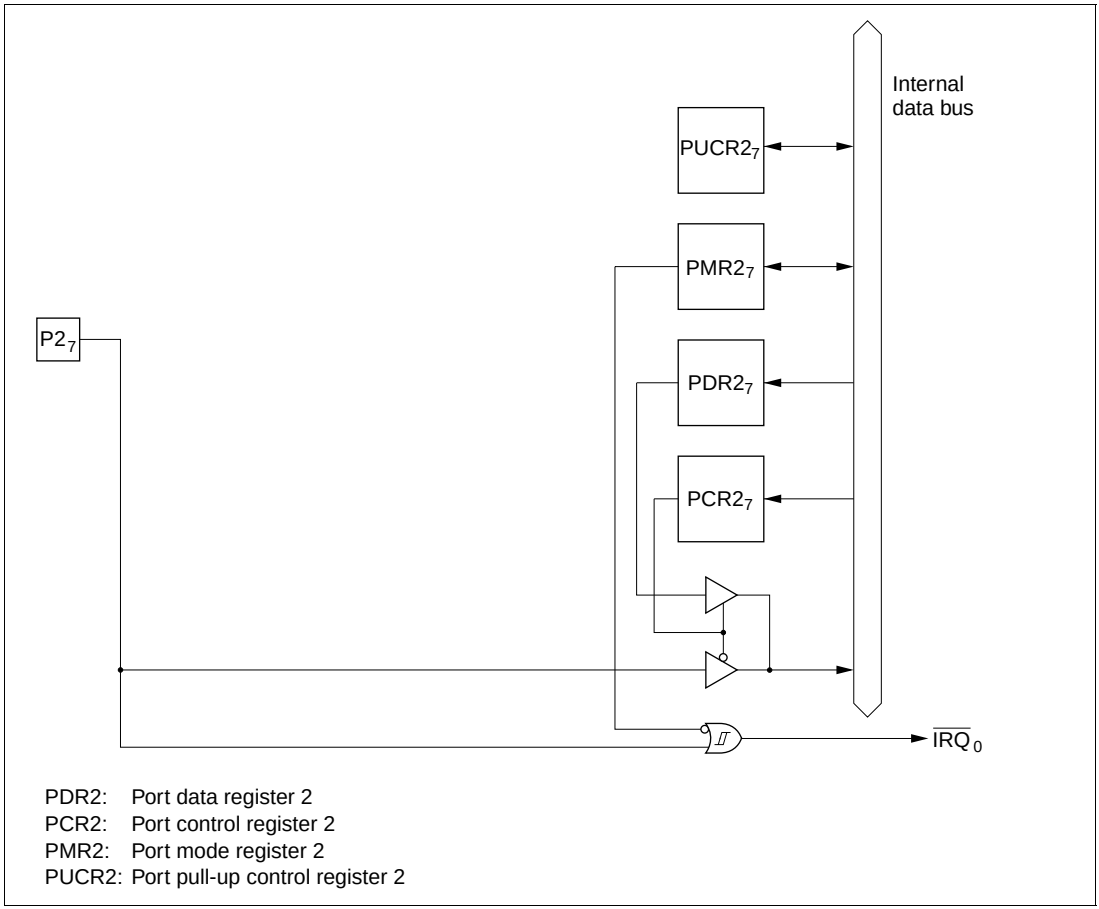


Figure C.1 (g) Port 1 Block Diagram (Pin $P1_0$)

C.2 Port 2 Block Diagrams



PDR2: Port data register 2
PCR2: Port control register 2
PMR2: Port mode register 2
PUCR2: Port pull-up control register 2

Figure C.2 (a) Port 2 Block Diagram (Pin P2$_7$)

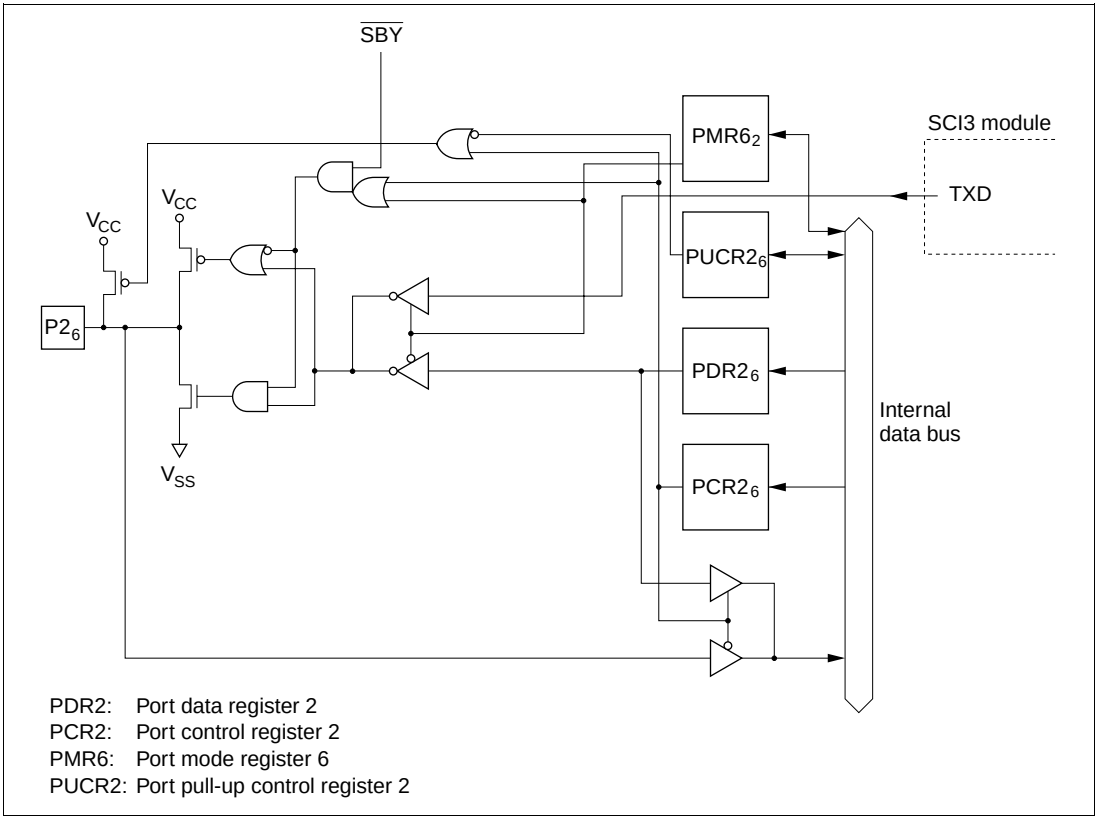
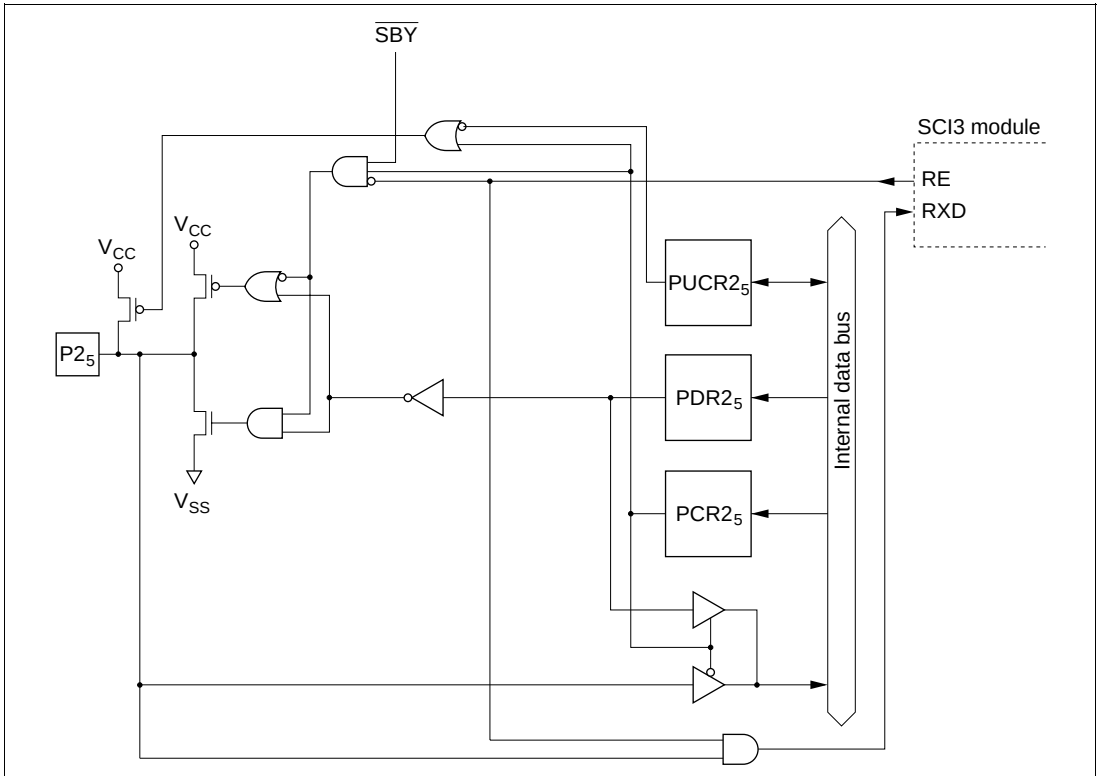
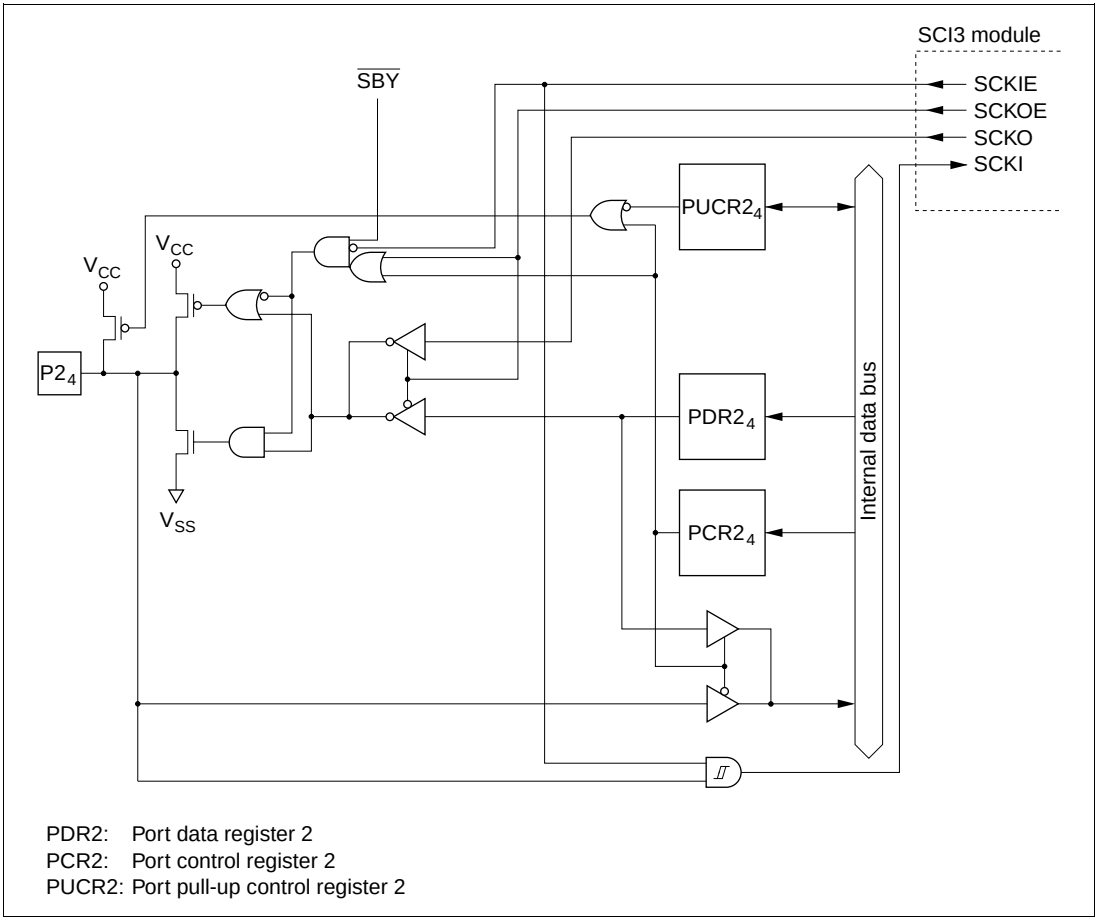


Figure C.2 (b) Port 2 Block Diagram (Pin $P2_6$)



PDR2: Port data register 2
PCR2: Port control register 2
PUCR2: Port pull-up control register 2

Figure C.2 (c) Port 2 Block Diagram (Pin $P2_5$)



PDR2: Port data register 2
PCR2: Port control register 2
PUCR2: Port pull-up control register 2

Figure C.2 (d) Port 2 Block Diagram (Pin $P2_4$)

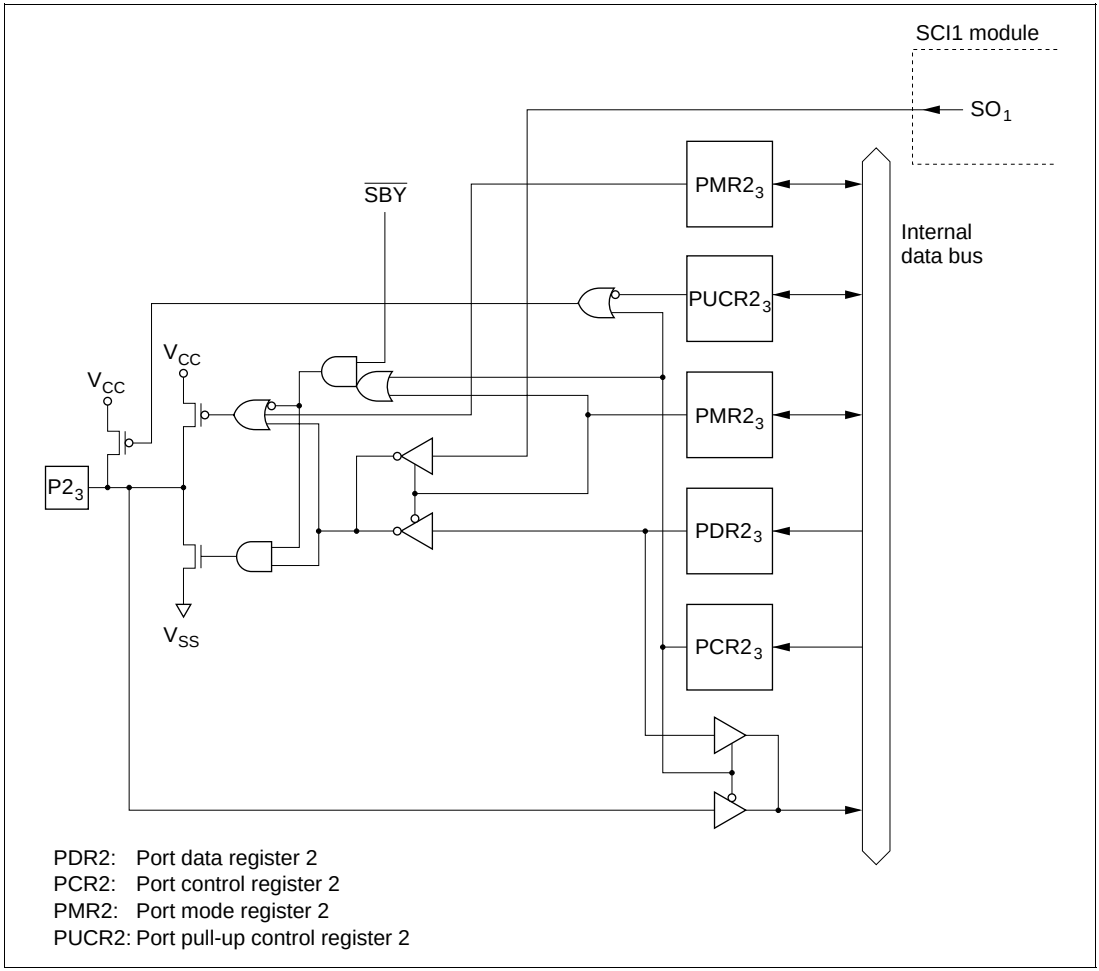


Figure C.2 (e) Port 2 Block Diagram (Pin $P2_3$)

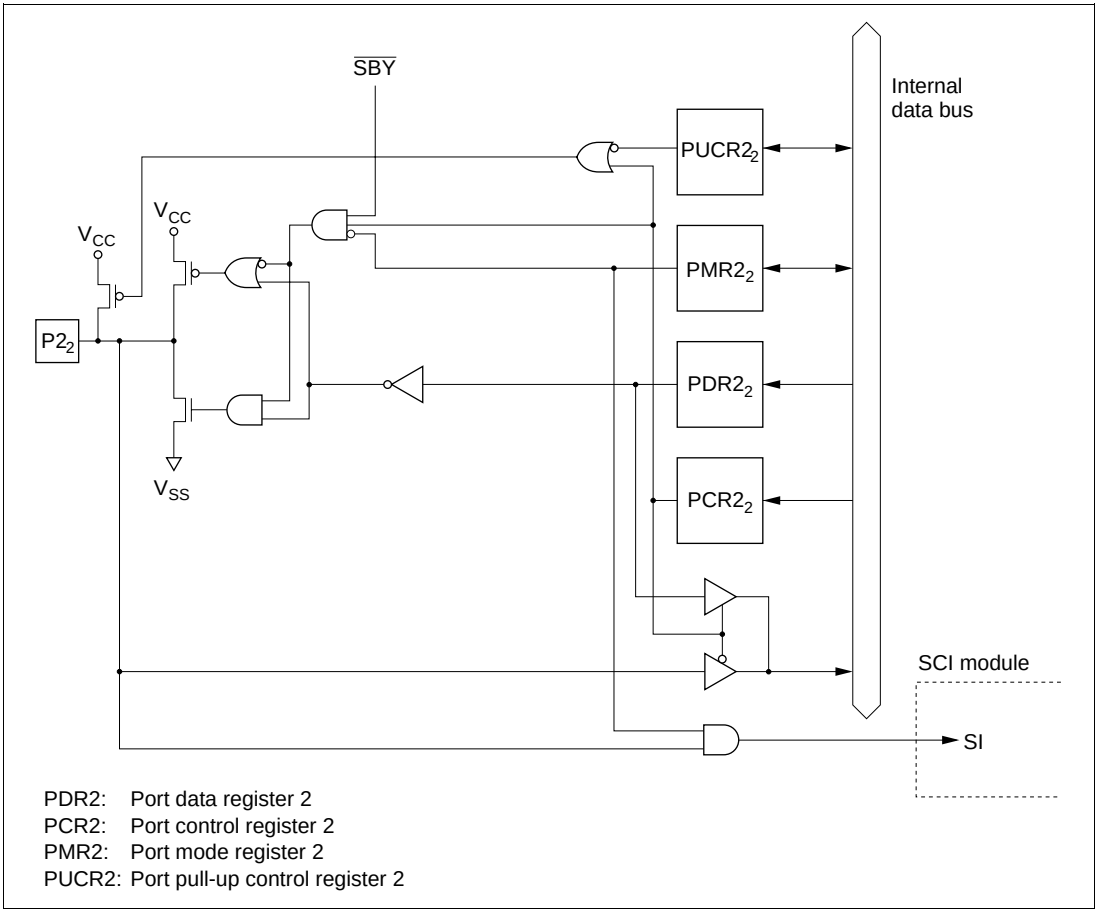
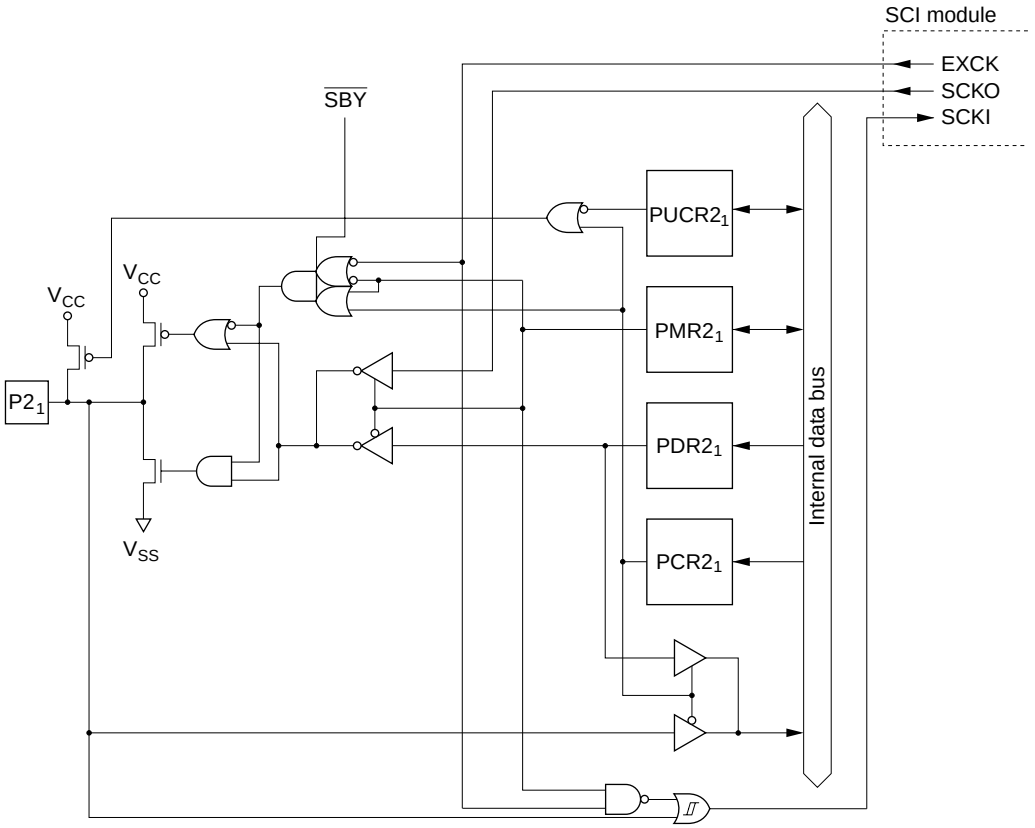


Figure C.2 (f) Port 2 Block Diagram (Pin $P2_2$)



PDR2: Port data register 2
PCR2: Port control register 2
PMR2: Port mode register 2
PUCR2: Port pull-up control register 2

Figure C.2 (g) Port 2 Block Diagram (Pin P2$_1$)

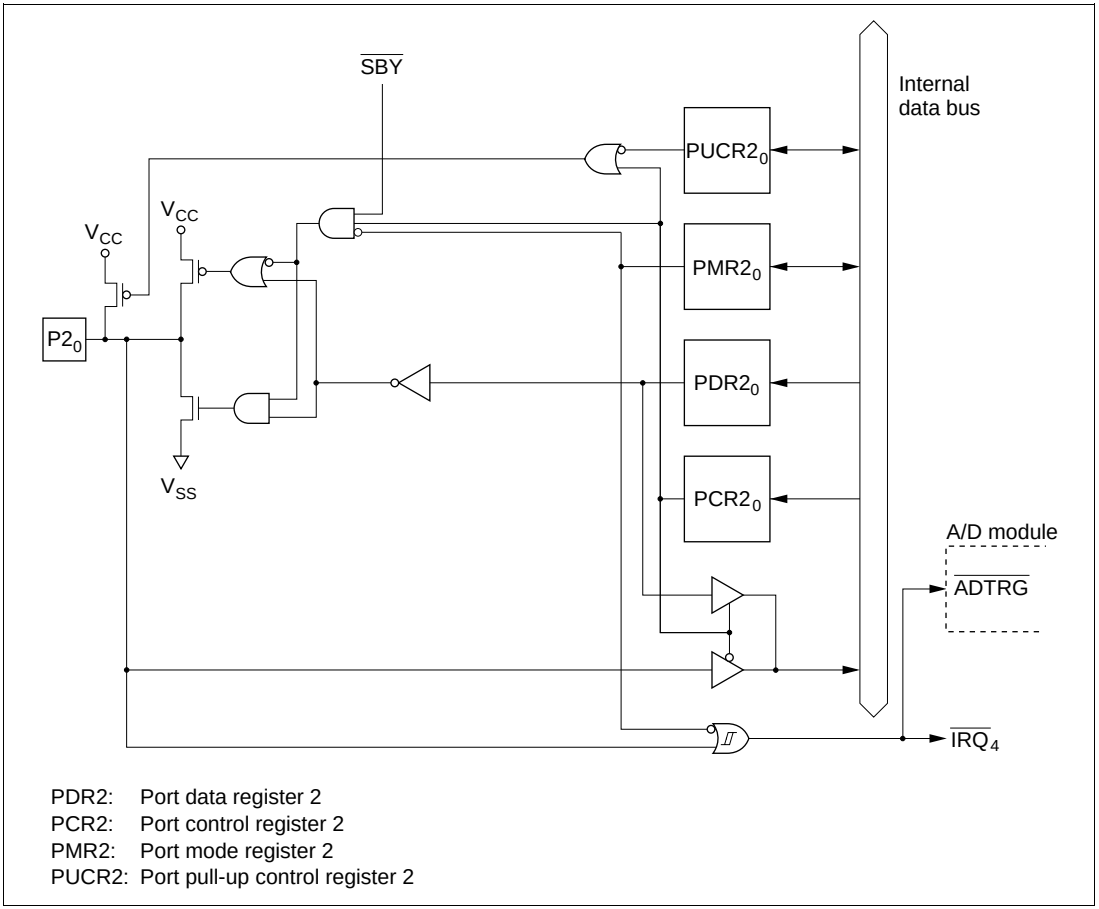
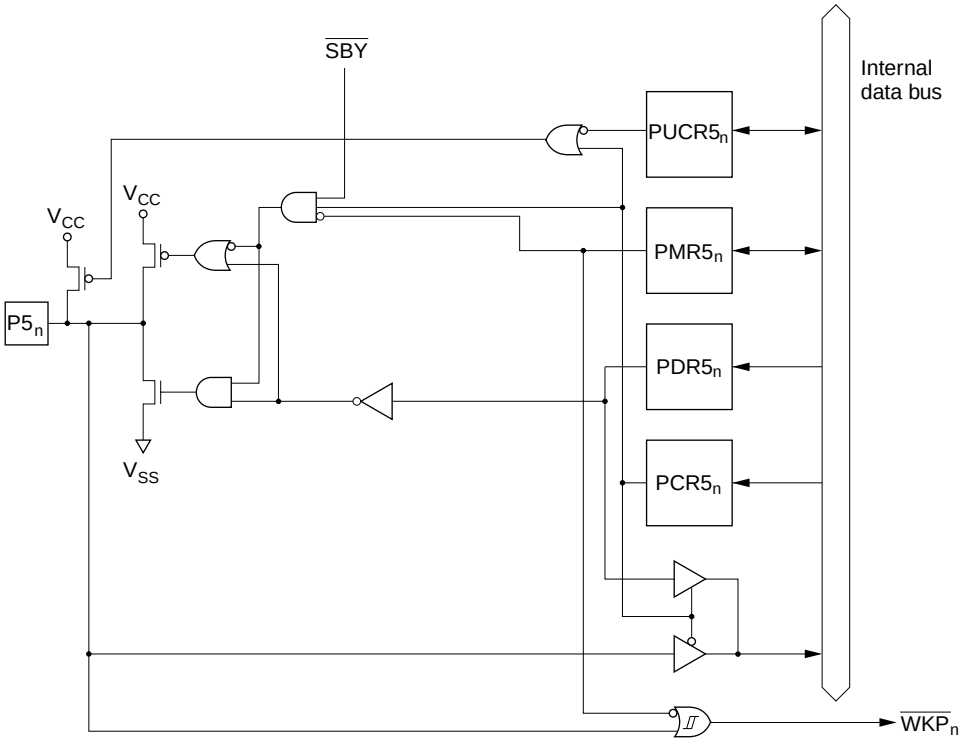


Figure C.2 (h) Port 2 Block Diagram (Pin $P2_0$)

C.3 Port 5 Block Diagram

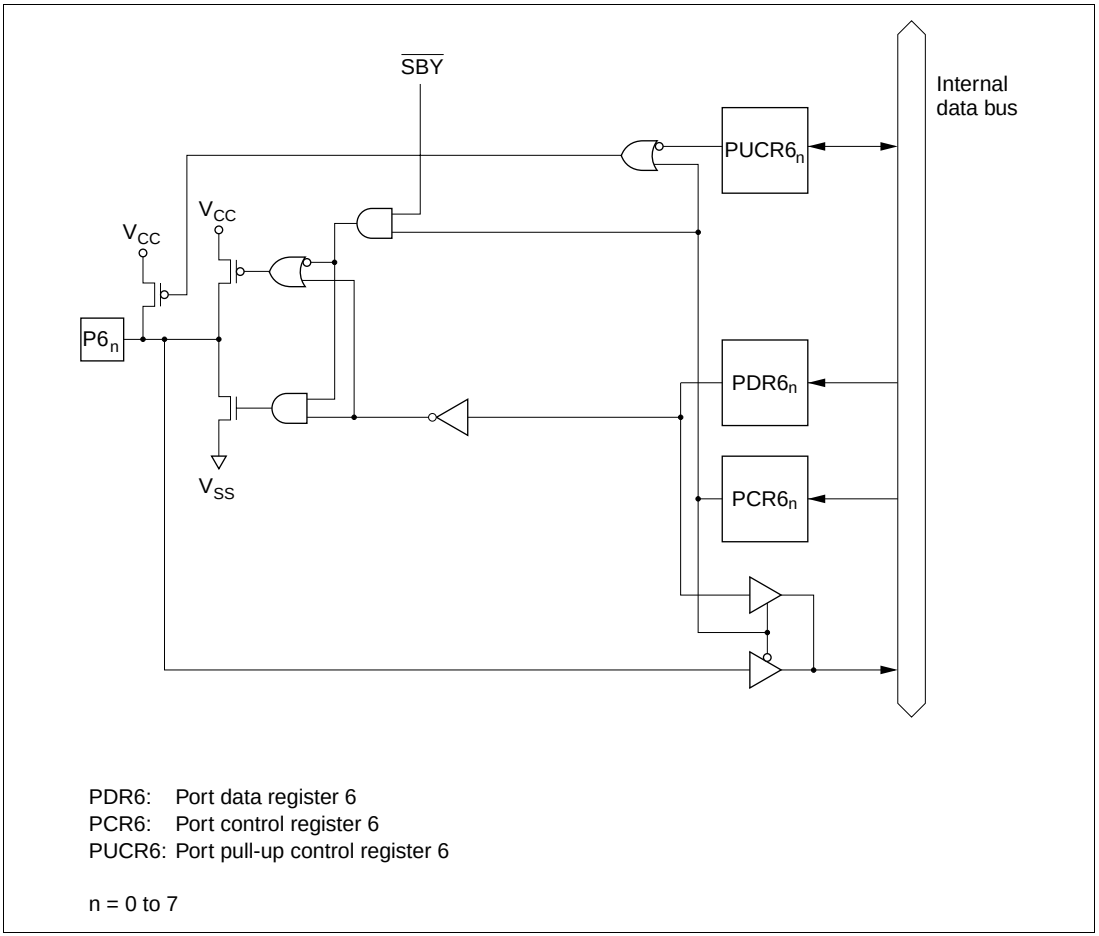


PDR5: Port data register 5
PCR5: Port control register 5
PMR5: Port mode register 5
PUCR5: Port pull-up control register 5

n = 0 to 7

Figure C.3 Port 5 Block Diagram

C.4 Port 6 Block Diagram



PDR6: Port data register 6
PCR6: Port control register 6
PUCR6: Port pull-up control register 6

n = 0 to 7

Figure C.4 Port 6 Block Diagram

C.5 Port 7 Block Diagram

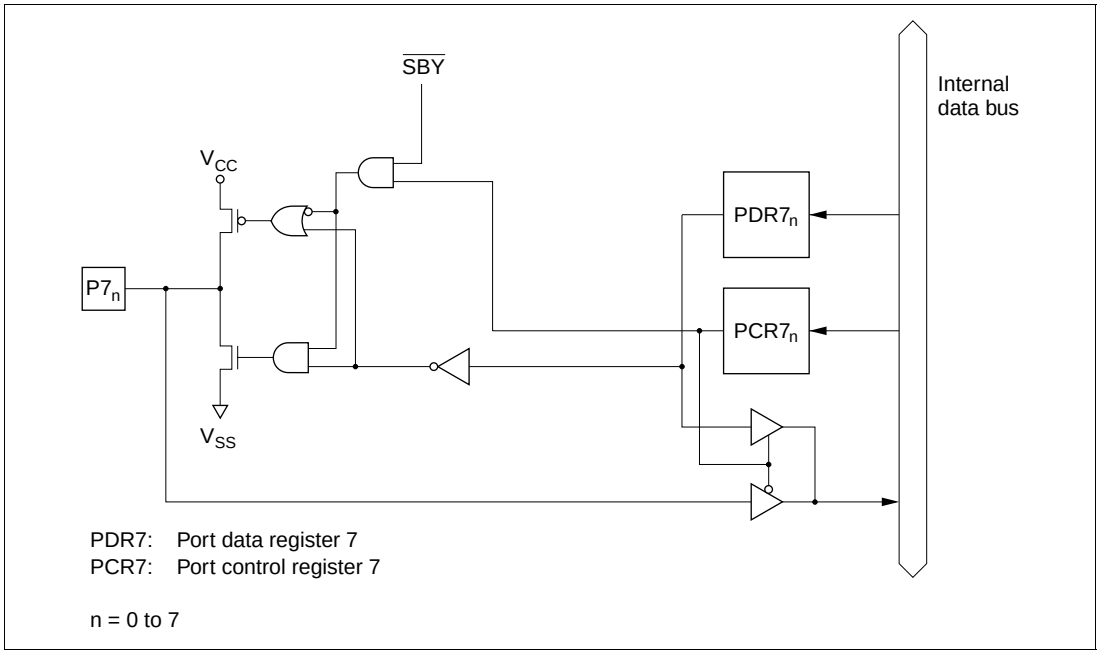


Figure C.5 Port 7 Block Diagram

C.6 Port 8 Block Diagram

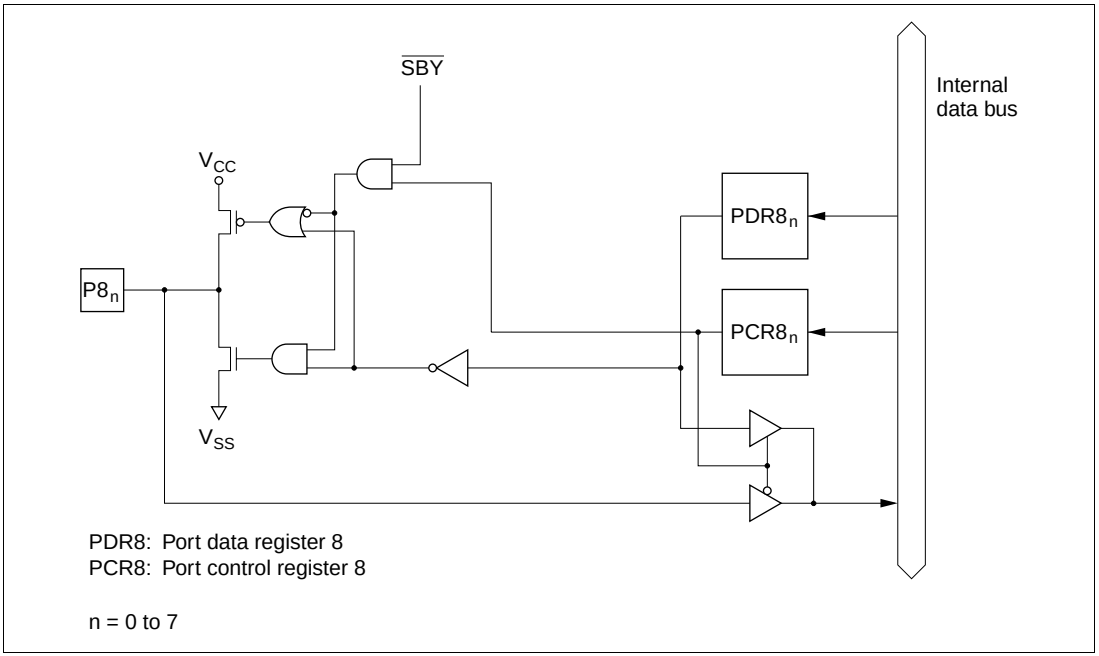


Figure C.6 Port 8 Block Diagram

C.7 Port A Block Diagram

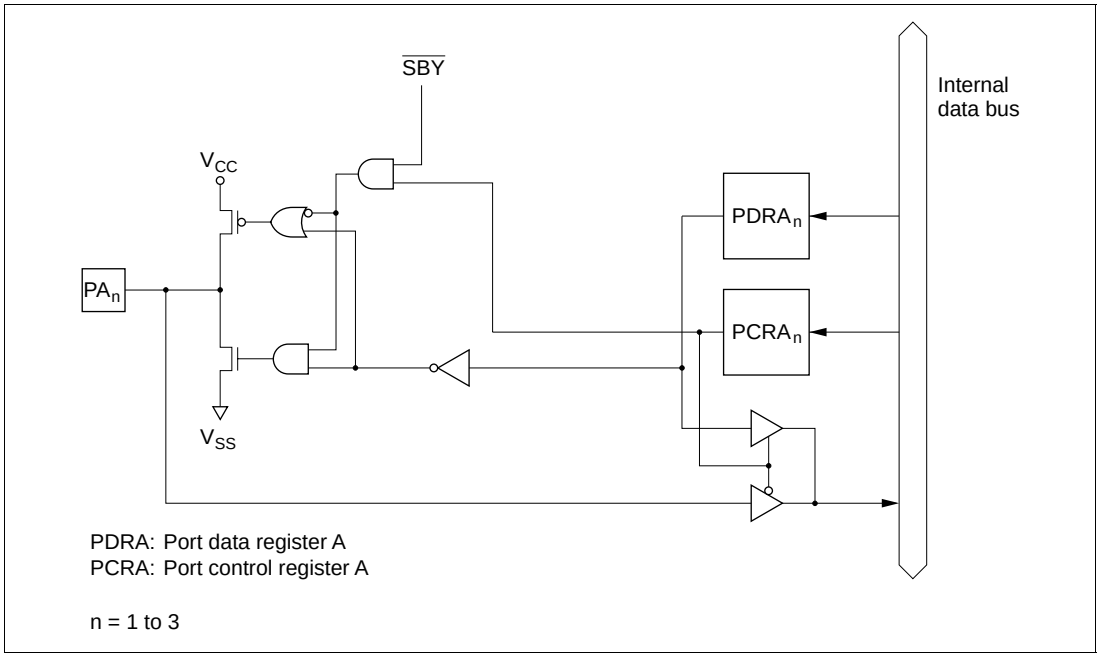


Figure C.7 Port A Block Diagram

C.8 Port B Block Diagram

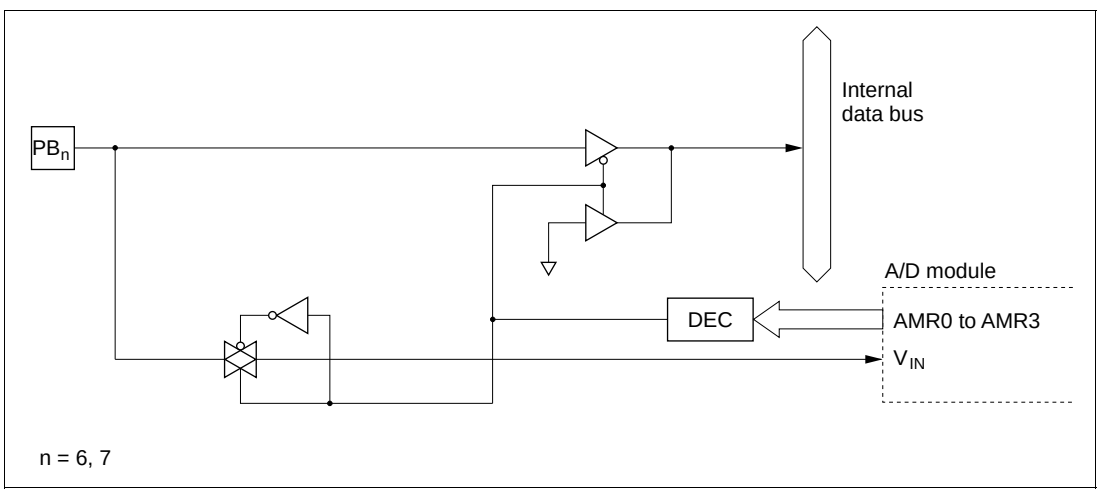


Figure C.8 Port B Block Diagram

Appendix D Port States in the Different Processing States

Table D.1 Port States Overview

Port	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P1_7$ to $P1_0$	High impedance	Retained	Retained	High impedance*	Retained	Functional	Functional
$P2_6$ to $P2_0$	High impedance	Retained	Retained	High impedance*	Retained	Functional	Functional
$P5_7$ to $P5_0$	High impedance	Retained	Retained	High impedance*	Retained	Functional	Functional
$P6_7$ to $P6_0$	High impedance	Retained	Retained	High impedance*	Retained	Functional	Functional
$P7_7$ to $P7_0$	High impedance	Retained	Retained	High impedance	Retained	Functional	Functional
$P8_7$ to $P8_0$	High impedance	Retained	Retained	High impedance	Retained	Functional	Functional
PA_3 to PA_1	High impedance	Retained	Retained	High impedance	Retained	Functional	Functional
$P2_7$, PB_7, PB_6	High impedance	High impedance	High impedance	High impedance	High impedance	High impedance	High impedance

Note: * High level output when MOS pull-up is in on state.

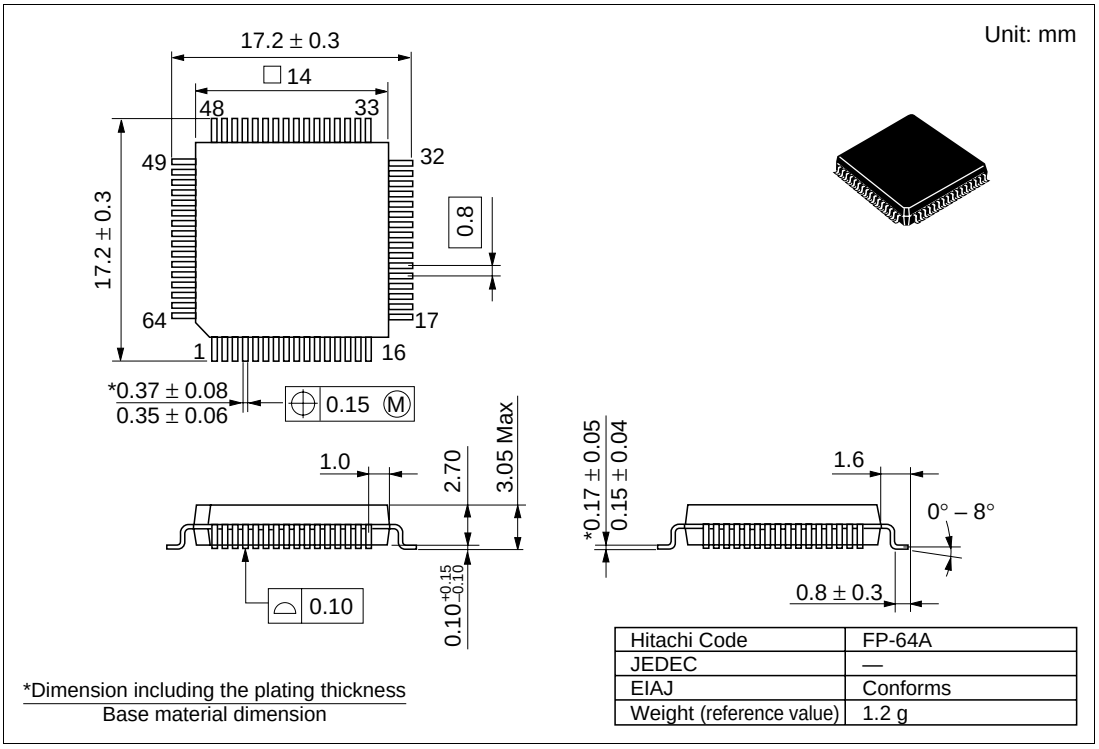
Appendix E Product Line-Up

Product Type			Product Code	Mark Code	Package (Hitachi Package Code)
H8/3627	ZTAT version	Standard products	HD6473627H	HD6473627H	64-pin QFP (FP-64A)
			HD6473627FP	HD6473627FP	64-pin LQFP (FP-64E)
	Mask ROM version	Standard products	HD6433627H	HD6433627(***)H	64-pin QFP (FP-64A)
			HD6433627FP	HD6433627(***)FP	64-pin LQFP (FP-64E)
H8/3626	Mask ROM version	Standard products	HD6433626H	HD6433626(***)H	64-pin QFP (FP-64A)
			HD6433626FP	HD6433626(***)FP	64-pin LQFP (FP-64E)
H8/3625	Mask ROM version	Standard products	HD6433625H	HD6433625(***)H	64-pin QFP (FP-64A)
			HD6433625FP	HD6433625(***)FP	64-pin LQFP (FP-64E)
H8/3624S	Mask ROM version	Standard products	HD6433624SH	HD6433624S(***)H	64-pin QFP (FP-64A)
			HD6433624SFP	HD6433624S(***)FP	64-pin LQFP (FP-64E)
H8/3623S	Mask ROM version	Standard products	HD6433623SH	HD6433623S(***)H	64-pin QFP (FP-64A)
			HD6433623SFP	HD6433623S(***)FP	64-pin LQFP (FP-64E)
H8/3622S	Mask ROM version	Standard products	HD6433622SH	HD6433622S(***)H	64-pin QFP (FP-64A)
			HD6433622SFP	HD6433622S(***)FP	64-pin LQFP (FP-64E)

Note: (***) in mask ROM versions is the ROM code.

Appendix F Package Dimensions

Dimensional drawings of the H8/3627 Series in FP-64A, and FP-64E packages are shown in figure F.1, and F.2, respectively.

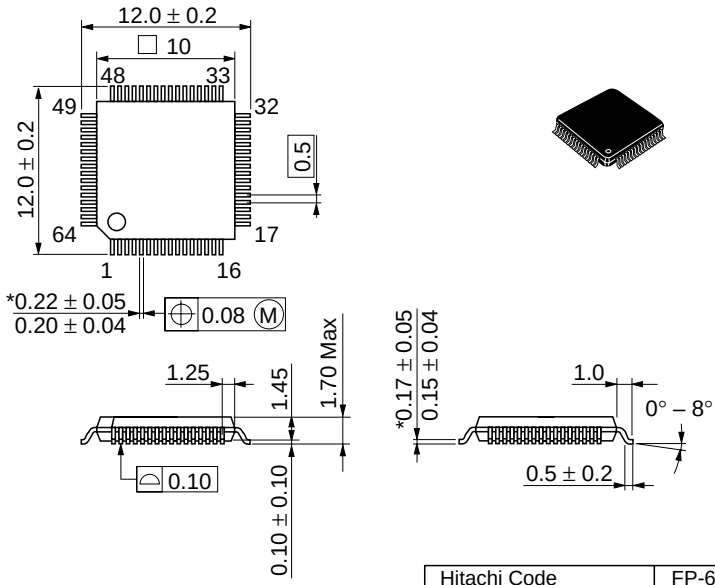


Hitachi Code	FP-64A
JEDEC	—
EIAJ	Conforms
Weight (reference value)	1.2 g

Figure F.1 FP-64A Package Dimensions

Note: In case of inconsistencies arising within figures, dimensional drawings listed in the Hitachi Semiconductor Packages Manual take precedence and are considered correct.

Unit: mm



Hitachi Code	FP-64E
JEDEC	—
EIAJ	Conforms
Weight (reference value)	0.4 g

*Dimension including the plating thickness
Base material dimension

Figure F.2 FP-64E Package Dimensions

H8/3627 Series Hardware Manual

Publication Date: 1st Edition, March 1999
Published by: Electronic Devices Business Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.
Copyright © Hitachi, Ltd., 1999. All rights reserved. Printed in Japan.