Hitachi Single-Chip Microcomputer

H8/3802 Series

H8/3802	HD6473802, HD6433802
H8/3801	HD6433801
H8/3800	HD6433800

Hardware Manual

HITACHI

ADE-602-203A
Rev. 2.0
1/9/01
Hitachi Ltd.

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.

List of Items Revised or Added for This Version

Section	Page	Item	Description
1.1 Overview	3	Table 1.1 Features	Description of time specification amended
2.8.1 Memory Map	46	Figure 2.16(2) H8/3801 Memory Map	Figure amended
	47	Figure 2.16(3) H8/3800 Memory Map	Figure amended
3.3.1 Overview	60	Table 3.2 Interrupt Sources and Their Priorities	Amended
3.3.2 Interrupt Control Registers	61	Table 3.3 Interrupt Control Registers	Initial values amended
		1. IRQ edge select register (IEGR)	Bits 4 to 2 amended
	62	2. Interrupt enable register 1 (IENR1)	Bits 6, 4, and 3 amended
	63 to 65	3. Interrupt enable register 2 (IENR2)	Bits 5, 4, and 1 amended
	65	4. Interrupt request register 1 (IRR1)	Bits 6, 4, and 3 amended
	67, 68	5. Interrupt request register 2 (IRR2)	Bits 5, 4, and 1 amended
3.3.5 Interrupt Operations	74	Figure 3.3 Flow up to Interrupt Acceptance	Figure amended
3.4.2 Notes on Rewriting Port Mode Registers	79	Table 3.5 Conditions under which Interrupt Request Flag is Set to 1	IRREC2 flag condition amended
3.4.3 Interrupt Request Flag Clearing Methods	80	3.4.3 Interrupt Request Flag Clearing Method	Description added
4.5 Note on Oscillators	90 to 92	4.5.1 Definition of Oscillation Setting Standby Time 4.5.2 Notes on Use of Crystal Oscillator Element(Excluding Ceramic Oscillator Element)	Description added
5.1 Overview	95	Table 5.2 Internal State in Each Operating Mode	Note 7 amended
5.3.3 Oscillator Setting Time after Standby Mode is Cleared	103	Table 5.4 Clock Frequency and Setting Time	Changed

Section	Page	Item	Description
5.5.2 Clearing Subsleep Mode	108	• Clearing by interrupt	Description amended
5.6 Subactive Mode	109	5.6.1 Transition to Subactive Mode	Description amended
6.3.1 Writing and Verifying	122	Figure 6.4 High-Speed,High-Reliability Programming Flow Chart	Write time t_{OPW} amended
8.1 Overview	131, 132	Table 8.1 Port Functions	Other function of port 3 and description of port 9 amended
8.2.2 Register Configuration and Description	133	Table 8.2 Port 3 Registers	Amended and register added
	134	1. Port data register 3 (PDR3)	Bit 0 and description amended
		2. Port control register 3 (PCR3)	Bit 0 and description amended
		3. Port pull-up control register 3 (PUCR3)	Bit 0 and description amended
	135, 136	4. Port mode register 3 (PMR3)	Bits 5 to 3 and 0, and description amended
	136	5. Port mode register 2 (PMR2)	Added
8.3.2 Register Configuration and Description	139	Table 8.5 Port 4 Register	Initial value amended
	140, 141	3. Port mode register 2 (PMR2)	Bits 2 and 1, and description amended
8.3.3 Pin Functions	141	Table 8.6 Port 4 Pin Functions	Description amended
8.7.2 Register Configuration and Description	155	Table 8.17 Port 8 Registers	Initial value amended
	156	1. Port data register 8 (PDR8)	Bits 7 to 1 amended
		2. Port control register 8 (PCR8)	Bits 7 to 1 amended
8.8 Port 9	158	8.8.1 Overview	Description amended
8.8.2 Register Configuration and Description		Table 8.20 Port 9 Registers	Initial value amended
	159	2. Port mode register 9 (PMR9)	Bit 2 amended, description added, and Note changed
8.10.2 Register Configuration and Description	165	Table 8.26 Port B Register	Initial values added

Section	Page	Item	Description
8.11.2 Register Configuration and Descriptions	168, 169	Serial Port Control Register (SPCR)	Bits 4, 1, and 0, and description amended
8.12 Application Note	170	8.12.1 How to Handle an Unused Pin	Description added
9.1 Overview	171	Table 9.1 Timer Functions	Internal clock of asynchronous event counter amended
9.2.1 Overview	174	Table 9.2 Timer A Registers	Initial value amended
9.2.2 Register Descriptions	174	1. Timer mode register A (TMA)	Bits 7 to 5 amended
9.2.5 Application Note	178	9.2.5 Application Note	Description added
9.3.4 Operation	192	1. Timer F operation a. Operation in 16-bit timer mode	Description amended
9.3.5 Application Note	196, 197	3. Clear timer FH, timer FL interrupt request flags (IRRTFH, IRRTFL), timer overflow flags H, L (OVFH, OVFL) and compare match flags H, L (CMFH, CMFL) 4. Timer counter (TCF) read/write	Description added
9.4.2 Register Configurations	202	5. Input pin edge selection register (AEGSR)	Bit name amended
	204	6. Event counter control register (ECCR)	Bit name, R/W form, and description amended
	205	7. Event counter control/status register (ECCSR)	Bit name, R/W form, and description amended
10.1.4 Register Configuration	220	Table 10.2 Registers	Initial value of serial port control register amended
10.2 Register Descriptions	227	10.2.6 Serial control register 3 (SCR3)	Description of bit 5 amended
	240	10.2.10 Serial Port Control Register (SPCR)	Bits 4, 1, and 0, and description amended
12.2 Register Descriptions	286, 287	12.2.2 A/D Mode Register (AMR)	Bit 6 amended
12.6 Application Notes	294	12.6 Application Notes	4th note added
13.1.4 Register Configuration	297	Table 13.2 LCD Controller/Driver Registers	Initial values amended

Section	Page	Item	Description
13.2 Register Descriptions	298	13.2.1 LCD Port Control Register (LPCR)	Bit 4 amended
	302	13.2.3 LCD Control Register 2 (LCR2)	Bits 4 to 0 amended
14.1 H8/3802 Series Absolute Maximum Ratings	313	Table 14.1 Absolute Maximum Ratings	Input voltage amended
14.2.2 DC Characteristics	318 to 321	Table 14.2 DC Characteristics	Added and amended
14.2.3 AC Characteristics	323	Table 14.3 Control Signal Timing	Amended
14.2.4 A/D Converter Characteristics	325	Table 14.5 A/D Converter Characteristics	Test conditions amended
14.3 Operation Timing	329	Figure 14.5 SCI3 Synchronous Mode Input/Output Timing	Reference figure in note amended
14.6 Usage Note	331	14.6 Usage Note	Added
B.2 Functions	356	SPCR	Initial values and R/W forms amended
	357	AEGSR	R/W form of bit 0 amended
	358	ECCR	R/W form of bit 0 amended
	359	ECCSR	R/W form of bit 5 amended
	366	TMA	Initial values and R/W forms of bits 7 to 5 amended
	368	TCRF	Description of bits 6 to 4 amended
	371	LPCR	Initial value and R/W form of bit 4 amended
	372	LCR	Description of bits 3 to 0 amended
	373	LCR2	Initial values and R/W forms of bits 4 to 0 amended
	374	AMR	Initial value and R/W form of bit 6 amended
	375	ADRRH	R/W forms amended
	376	PMR2	Initial values and R/W forms amended
	377	PMR3	R/W forms amended

Section	Page	Item	Description
B.2 Functions	380	PDR3	Initial value amended and description added
		PDR4	Descriptions added
	381	PDR5	Description added
		PDR6	Description added
		PDR7	Description added
		PDR8	Initial values amended and description added
	382	PDR9	Description added
		PDRA	Description added
		PDRB	Description added
		PUCR3	Initial value and R/W form amended
	383	PCR3	Initial value and R/W form amended
	385	PCR8	Initial values and R/W forms amended
		PMR9	Initial value and R/W form of bit 2 amended
	386	PCRA	Initial values of bits 7 to 4 amended
		PMRB	Initial values of bits 7 to 4 and 2 to 0 amended
	389	IEGR	Initial values and R/W forms of bits 7, 4 to 2 and description of bit 1 amended
	390	IENR1	Initial values and R/W forms of bits 6, 4, and 3 amended
	391	IENR2	Initial values and R/W forms of bits 5, 4, and 1 amended
	392	IRR1	Initial values and R/W forms of bits 6, 4, and 3 amended
	393	IRR2	Initial values and R/W forms of bits 5, 4, and 1 amended

Section	Page	Item	Description
C.1 Block Diagrams of Port 3	398	Figure C.1(b) Port 3 Block Diagram (Pin $P3_5$)	Added
	399	Figure C.1(c) Port 3 Block Diagram (Pins $P3_4$ and $P3_3$)	Pin $P3_5$ deleted
C.7 Block Diagrams of Port 9	409	Figure C.7(a) Port 9 Block Diagram (Pins $P9_1$ and $P9_0$)	Figure amended

Preface

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

The H8/3802 Series has a system-on-a-chip architecture that includes such peripheral functions as an LCD controller/driver, three timers, a two-channel 10-bit PWM, a serial communication interface, and an A/D converter. This allows H8/3802 Series devices to be used as embedded microcomputers in systems requiring LCD display.

This manual describes the hardware of the H8/3802 Series. For details on the H8/3802 Series instruction set, refer to the H8/300L Series Programming Manual.

Contents

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 H8/3802 Series comprises single-chip microcomputers equipped with a controller/driver. Other on-chip peripheral functions include three timers, a two-channel 10-bit pulse width modulator (PWM), a serial communication interface, and an A/D converter. Together, these functions make the H8/3800 Series ideally suited for embedded applications in systems requiring low power consumption and LCD display. Models in the H8/3802 Series are the H8/3802, with on-chip 16-kbyte ROM and 1-kbyte RAM, the H8/3801, with 12-kbyte ROM and 512 byte RAM, and the H8/3800, with 8-kbyte ROM and 512 byte RAM.

The H8/3802 is also available in a ZTAT™* version with on-chip PROM which can be programmed as required by the user.

Table 1.1 summarizes the features of the H8/3802 Series.

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

Table 1.1 Features

Item	Specification
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: 8 MHz — Add/subtract: 0.25 µs (operating at 8 MHz) — Multiply/divide: 1.75 µs (operating at 8 MHz) — Can run on 32.768 kHz or 38.4 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 • Typical instructions — Multiply (8 bits $\times$ 8 bits) — Divide (16 bits $\div$ 8 bits) — Bit accumulator — Register-indirect designation of bit position
Interrupts	18 interrupt sources • 11 external interrupt sources (IRQ_1, IRQ_0, WKP_7 to WKP_0, IRQAEC) • 7 internal interrupt sources
Clock pulse generators	Two on-chip clock pulse generators • System clock pulse generator: 1.0 to 16 MHz • Subclock pulse generator: 32.768 kHz, 38.4 kHz
Power-down modes	Seven power-down modes • Sleep (high-speed) mode • Sleep (medium-speed) mode • Standby mode • Watch mode • Subsleep mode • Subactive mode • Active (medium-speed) mode

Item	Specification
Memory	Large on-chip memory • H8/3802: 16-kbyte ROM, 1-kbyte RAM • H8/3801: 12-kbyte ROM, 512 byte RAM • H8/3800: 8-kbyte ROM, 512 byte RAM
I/O ports	50 pins • 39 I/O pins • 5 input pins • 6 output pins
Timers	Three on-chip timers • Timer A: 8-bit timer 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 ($ø_W$)* • Asynchronous event counter: 16-bit timer — Count-up timer able to count asynchronous external events independently of the MCU's internal clocks Asynchronous external events can be counted (both rising and falling edge detection possible) • Timer F: 16-bit timer — Can be used as two independent 8-bit timers — Count-up by an event input from the four internal clocks — Provision for toggle output by means of compare-match function
Serial communication interface	• SCI3: 8-bit synchronous/asynchronous serial interface Incorporates multiprocessor communication function
10-bit PWM	Pulse-division PWM output for reduced ripple • Can be used as a 10-bit D/A converter by connecting to an external low-pass filter.
A/D converter	Successive approximations using a resistance ladder • 4-channel analog input pins • Conversion time: 31/ø or 62/ø per channel

Item	Specification			
LCD controller/driver	LCD controller/driver equipped with a maximum of 25 segment pins and four common pins • Choice of four duty cycles (static, 1/2, 1/3, or 1/4) • Segment pins can be switched to general-purpose port function in 4-bit units			
Product lineup	**Product Code**			
	Mask ROM Version	**ZTAT Version**	**Package**	**ROM/RAM Size**
	HD6433802H	HD6473802H	64-pin QFP (FP-64A)	ROM 16 kbytes
	HD6433802FP	HD6473802FP	64-pin LQFP (FP-64E)	RAM 1 kbytes
	HD6433802P	HD6473802P	64-pin DILP (DP-64S)	
	HD6433801H	—	64-pin QFP (FP-64A)	ROM 12 kbytes
	HD6433801FP	—	64-pin LQFP (FP-64E)	RAM 512 bytes
	HD6433801P	—	64-pin DILP (DP-64S)	
	HD6433800H	—	64-pin QFP (FP-64A)	ROM 8 kbytes
	HD6433800FP	—	64-pin LQFP (FP-64E)	RAM 512 bytes
	HD6433800P	—	64-pin DILP (DP-64S)	

Note: * See section 4, Clock Pulse Generator, for the definition of ø and. $ø_w$.

1.2 Internal Block Diagram

Figure 1.1 shows a block diagram of the H8/3802 Series.

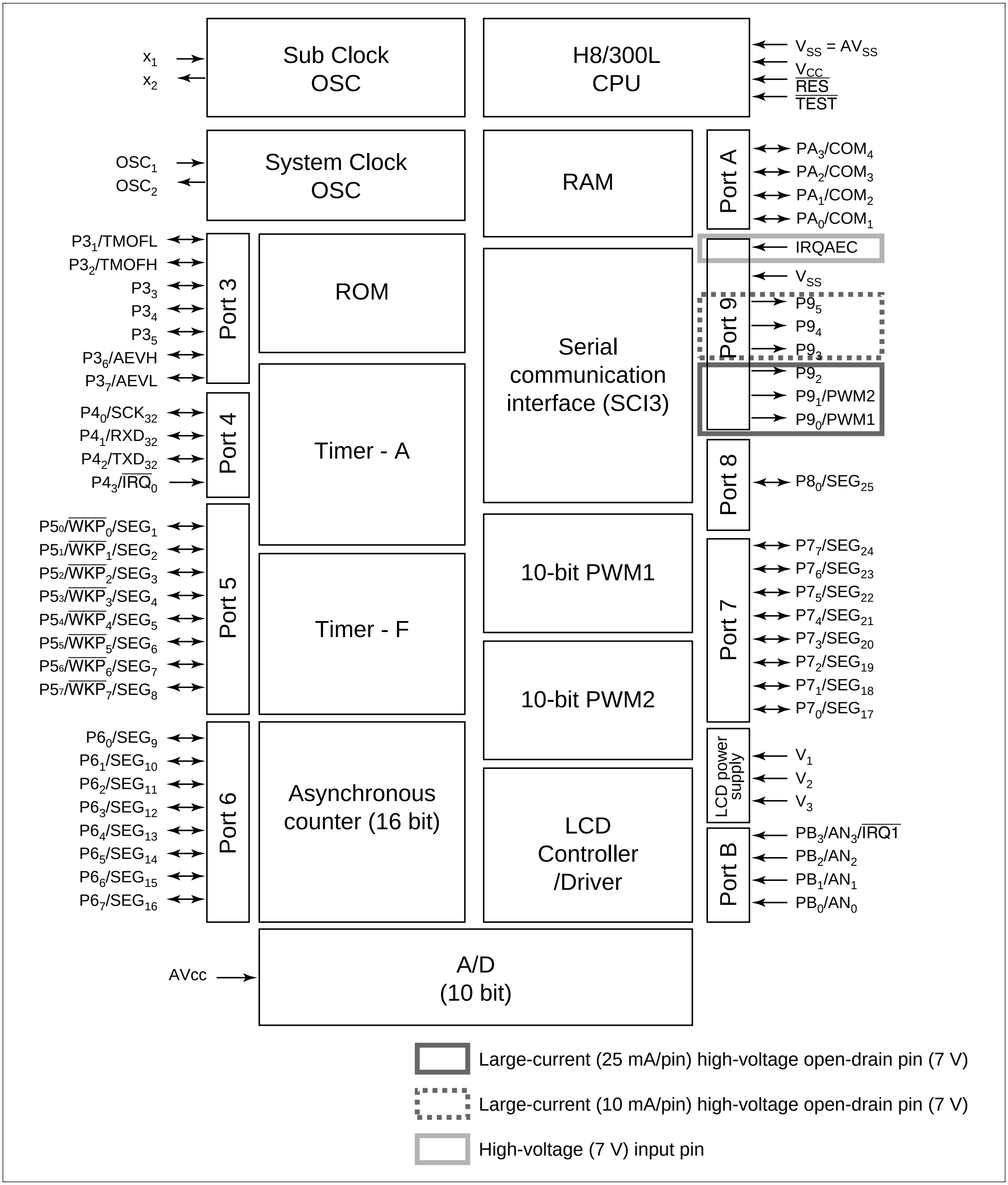


Figure 1.1 Block Diagram

1.3 Pin Arrangement and Functions

1.3.1 Pin Arrangement

The H8/3802 Series pin arrangement is shown in figures 1.2 and 1.3.

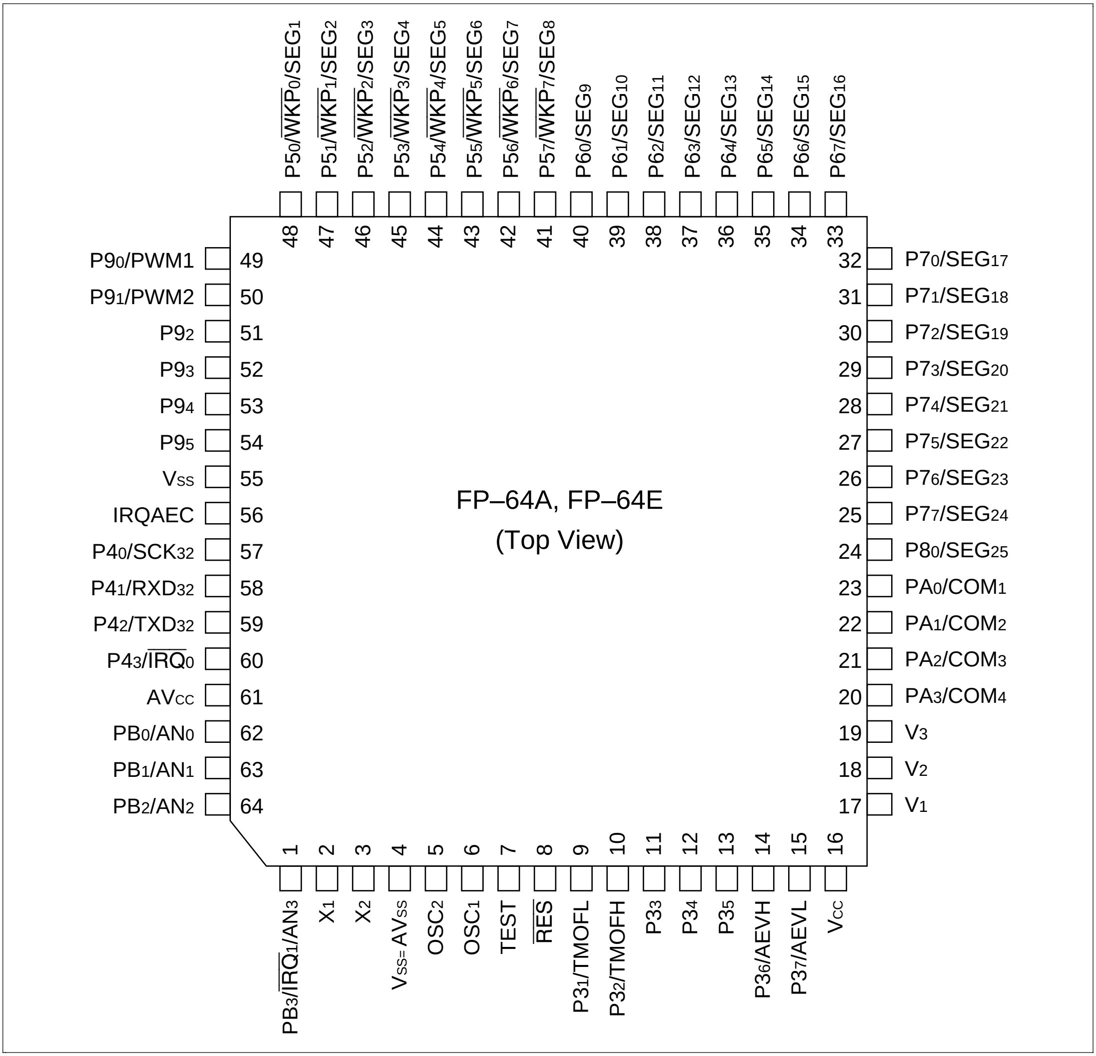


Figure 1.2 Pin Arrangement (FP-64A, FP-64E: Top View)

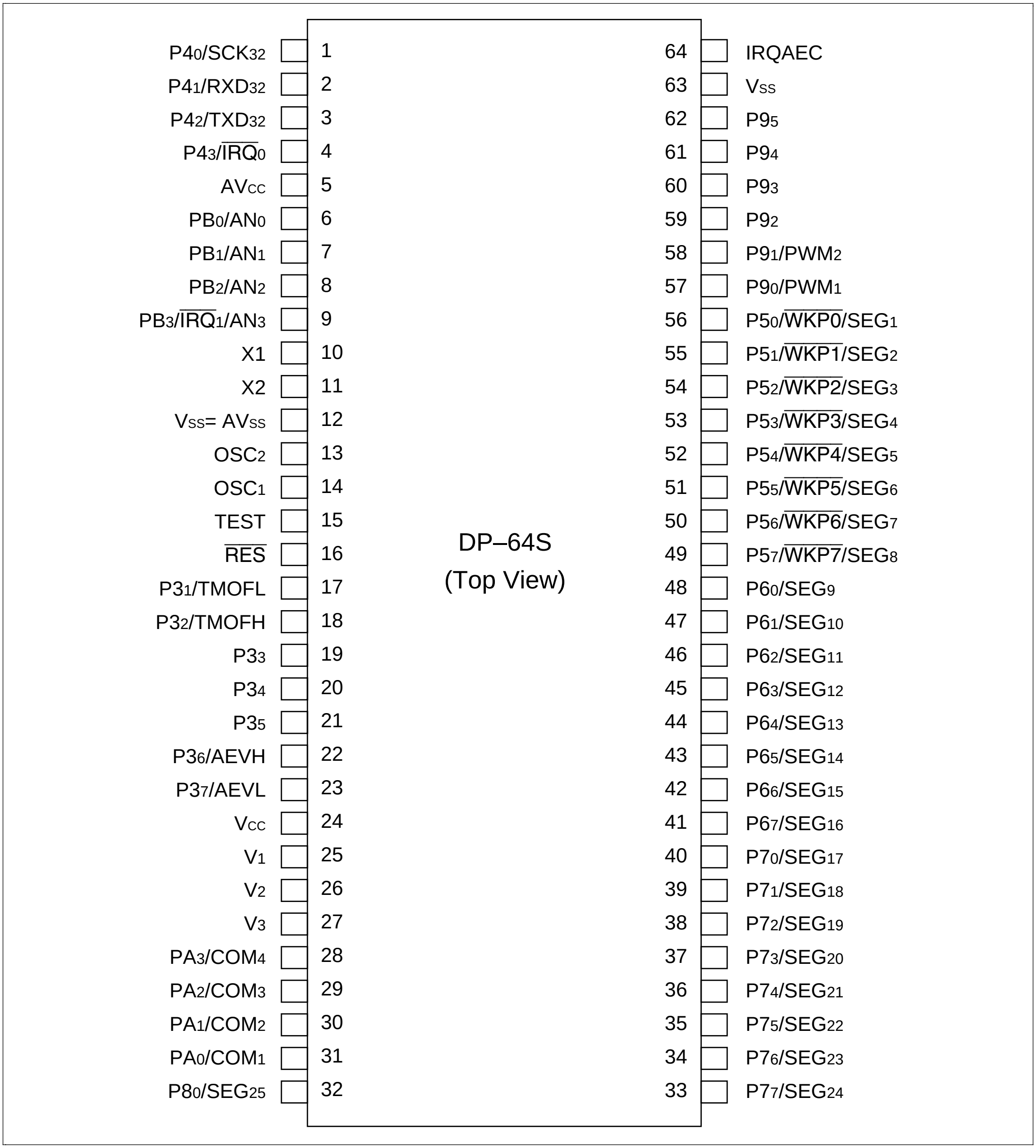


Figure 1.3 Pin Arrangement (DP-64S: Top View)

1.3.2 Pin Functions

Table 1.2 outlines the pin functions of the H8/3802 Series.

Table 1.2 Pin Functions

Type	Symbol	Pin No. FP-64A FP-64E	Pin No. DP-64S	I/O	Name and Functions
Power source pins	V_{CC}	16	24	Input	**Power supply:** All V_{CC} pins should be connected to the system power supply.
	V_{SS}	4 (= AV_{SS}) 55	12 (= AV_{SS}) 63	Input	**Ground:** All V_{SS} pins should be connected to the system power supply (0 V).
	AV_{CC}	61	5	Input	**Analog power supply:** This is the power supply pin for the A/D converter. When the A/D converter is not used, connect this pin to the system power supply.
	AV_{SS}	4 (= V_{SS})	12 (= V_{SS})	Input	**Analog ground:** This is the A/D converter ground pin. It should be connected to the system power supply (0V).
	V_1 V_2 V_3	17 18 19	25 26 27	Input	**LCD power supply:** These are the power supply pins for the LCD controller/driver.
Clock pins	OSC_1	6	14	Input	These pins connect to a crystal or ceramic oscillator, or can be used to input an external clock. See section 4, Clock Pulse Generators, for a typical connection diagram.
	OSC_2	5	13	Output	
	X_1	2	10	Input	These pins connect to a 32.768-kHz or 38.4-kHz crystal oscillator. See section 4, Clock Pulse Generators, for a typical connection diagram.
	X_2	3	11	Output	
System control	$\overline{RES}$	8	16	Input	**Reset:** When this pin is driven low, the chip is reset
	TEST	7	15	Input	**Test pin:** This pin is reserved and cannot be used. It should be connected to V_{SS}.

Type	Symbol	Pin No. FP-64A FP-64E	DP-64S	I/O	Name and Functions
Interrupt pins	$\overline{IRQ_0}$ $\overline{IRQ_1}$	60 1	4 9	Input	**IRQ interrupt request 0 and 1:** These are input pins for edge-sensitive external interrupts, with a selection of rising or falling edge
	IRQAEC	56	64	Input	**Asynchronous event counter event signal:** This is an interrupt input pin for enabling asynchronous event input.
	$\overline{WKP_7}$ to $\overline{WKP_0}$	41 to 48	49 to 56	Input	**Wakeup interrupt request 0 to 7:** These are input pins for rising or falling-edge-sensitive external interrupts.
Timer pins	AEVL AEVH	15 14	23 22	Input	**Asynchronous event counter event input:** This is an event input pin for input to the asynchronous event counter.
	TMOFL	9	17	Output	**Timer FL output:** This is an output pin for waveforms generated by the timer FL output compare function.
	TMOFH	10	18	Output	**Timer FH output:** This is an output pin for waveforms generated by the timer FH output compare function.
10-bit PWM pin	PWM1 PWM2	49 50	57 58	Output	**10-bit PWM output:** These are output pins for waveforms generated by the channel 1 and 2 10-bit PWMs.
I/O ports	$P3_7$ to $P3_1$	15 to 9	23 to 17	I/O	**Port 3:** This is an 7-bit I/O port. Input or output can be designated for each bit by means of port control register 3 (PCR3).
	$P4_3$	60	4	Input	**Port 4 (bit 3):** This is a 1-bit input port.
	$P4_2$ to $P4_0$	59 to 57	3 to 1	I/O	**Port 4 (bits 2 to 0):** This is a 3-bit I/O port. Input or output can be designated for each bit by means of port control register 4 (PCR4).
	$P5_7$ to $P5_0$	41 to 48	49 to 56	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).

Type	Symbol	Pin No. FP-64A FP-64E	Pin No. DP-64S	I/O	Name and Functions
I/O ports	$P6_7$ to $P6_0$	33 to 40	41 to 48	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 $P7_0$	25 to 32	33 to 40	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).
	$P8_0$	24	32	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).
	$P9_5$ to $P9_0$	54 to 49	62 to 57	Output	**Port 9:** This is a 6-bit output port.
	PA_3 to PA_0	20 to 23	28 to 31	I/O	**Port A:** This is a 4-bit I/O port. Input or output can be designated for each bit by means of port control register A (PCRA).
	PB_3 to PB_0	1, 64 to 62	9 to 6	Input	**Port B:** This is a 4-bit input port.
Serial communi-cation interface (SCI)	RXD_{32}	58	2	Input	**SCI3 receive data input:** This is the SCI3 data input pin.
	TXD_{32}	59	3	Output	**SCI3 transmit data output:** This is the SCI3 data output pin.
	SCK_{32}	57	1	I/O	**SCI3 clock I/O:** This is the SCI3 clock I/O pin.
A/D converter	AN3 to An0	1 64 to 62	9 to 6	Input	**Analog input channels 3 to 0:** These are analog data input channels to the A/D converter
LCD controller/ driver	COM_4 to COM_1	20 to 23	28 to 31	Output	**LCD common output:** These are the LCD common output pins.
	SEG_{25} to SEG_1	24 to 48	32 to 56	Output	**LCD segment output:** These are the LCD segment output pins.

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 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: 0.25 µs*
  — 8 × 8-bit multiply: 1.75 µs*
  — 16 ÷ 8-bit divide: 1.75 µs*

Note: * These values are at ø = 8 MHz.

- Low-power operation modes
  SLEEP instruction for transfer to low-power operation

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.

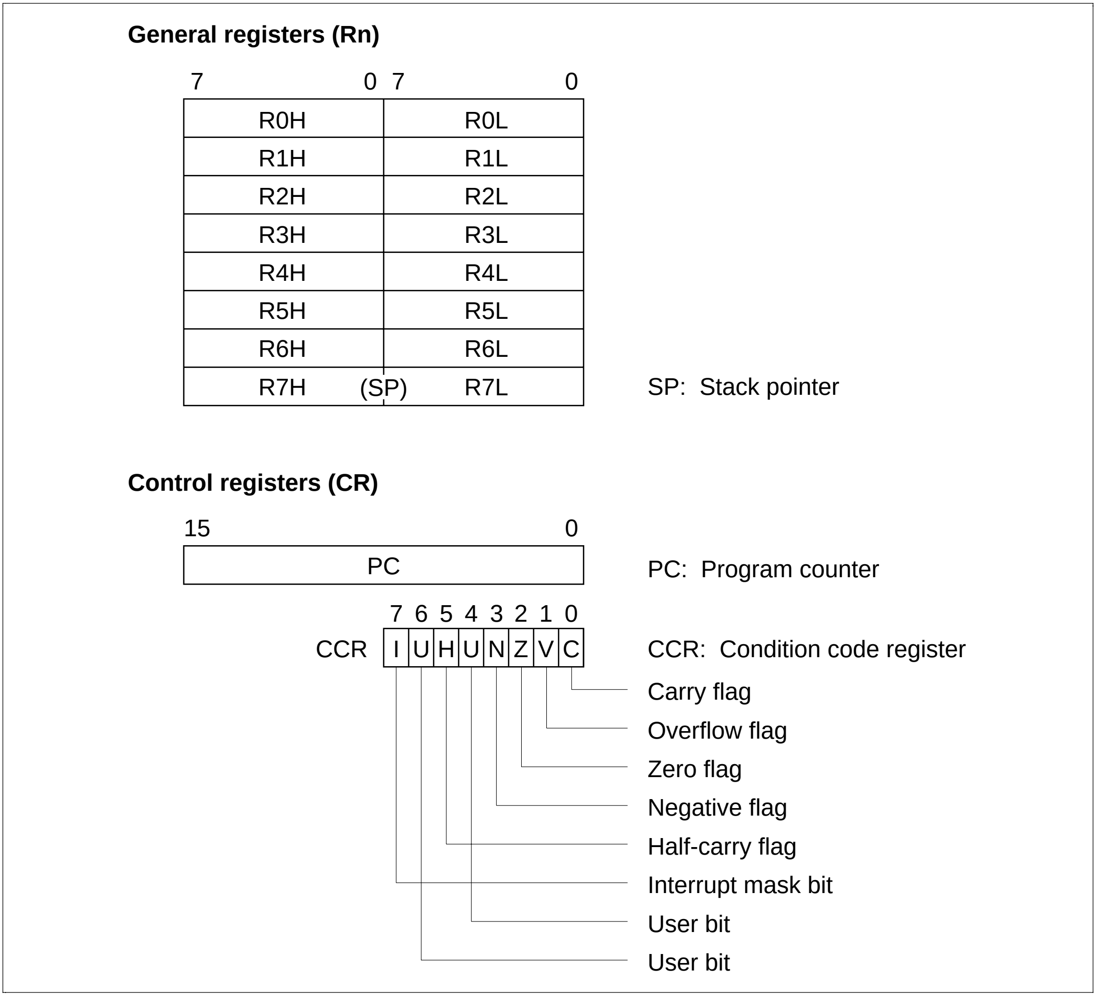


Figure 2.1 CPU Registers

2.2 Register Descriptions

2.2.1 General Registers

All the general registers 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 processing 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.

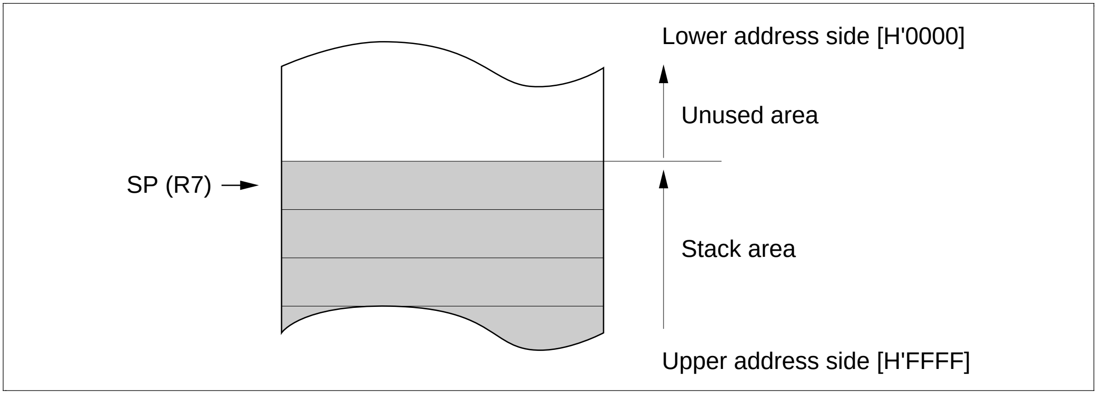


Figure 2.2 Stack Pointer

2.2.2 Control Registers

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

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).

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 section 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 and rotate instructions, to store the value shifted out of the end bit

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. The stack pointer should be initialized by software, by the first instruction executed 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 packed BCD form. Each nibble of the byte is treated as a decimal digit.

2.3.1 Data Formats in General Registers

Data of all the sizes above can be stored in general registers as shown in figure 2.3.

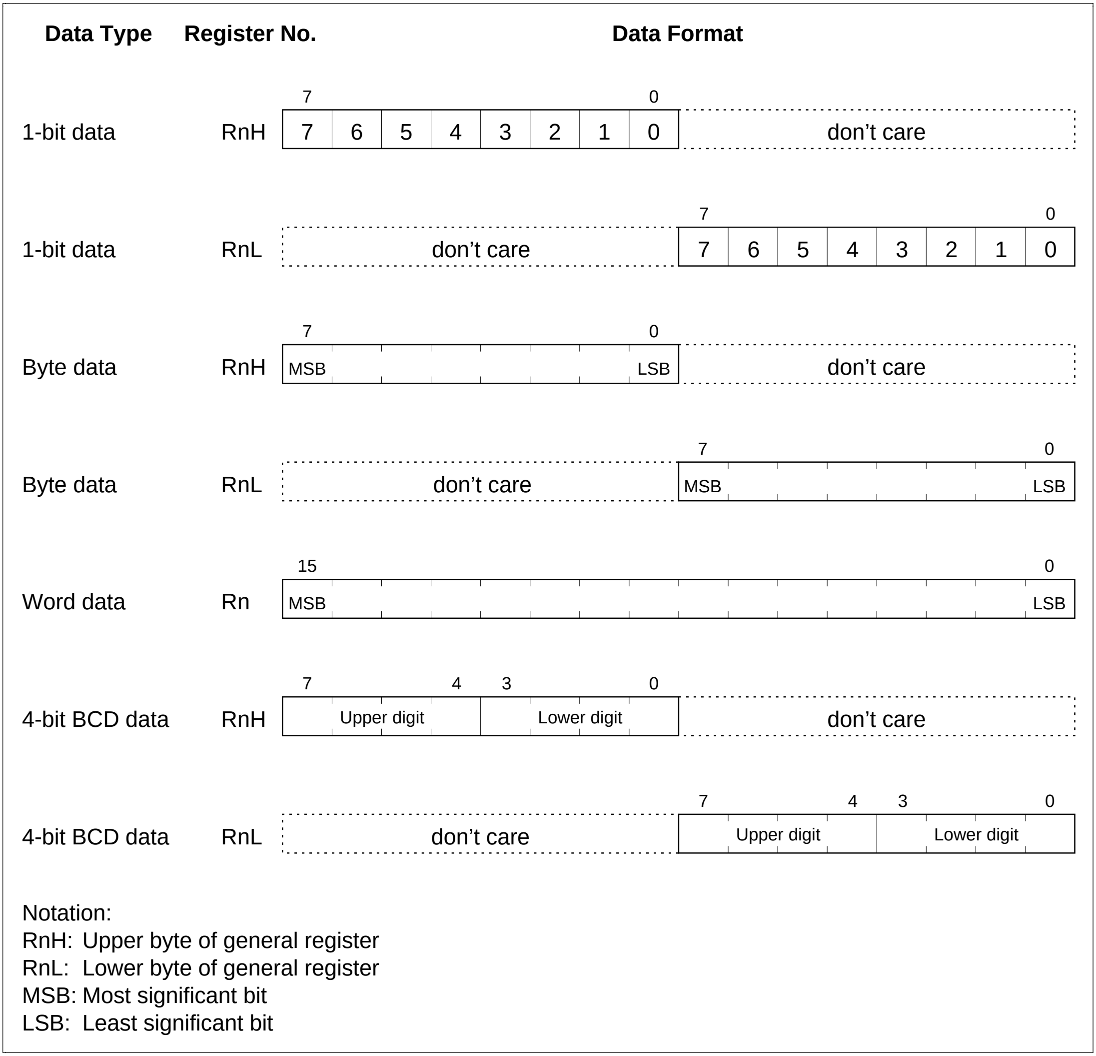
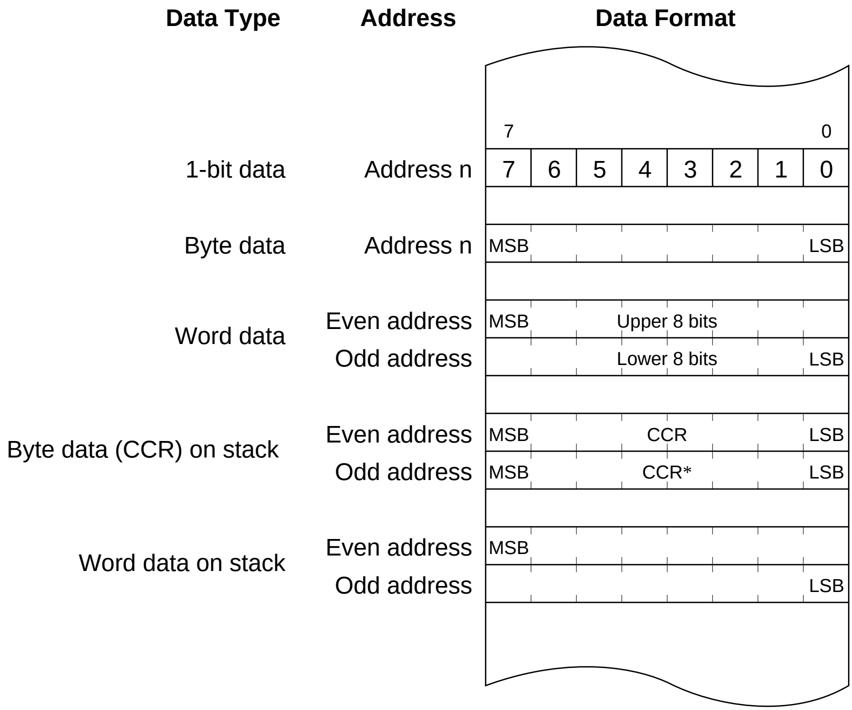


Figure 2.3 Register Data Formats

2.3.2 Memory Data Formats

Figure 2.4 indicates the data formats in memory. The H8/300L CPU can access word data stored in memory (MOV.W instruction), but the word data must always begin at an even address. If word data starting at an odd address is accessed, the least significant bit of the address is regarded as 0, and the word data starting at the preceding address is accessed. The same applies to instruction codes.



CCR: Condition code register

Note: * Ignored on return

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 displacement 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 from 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

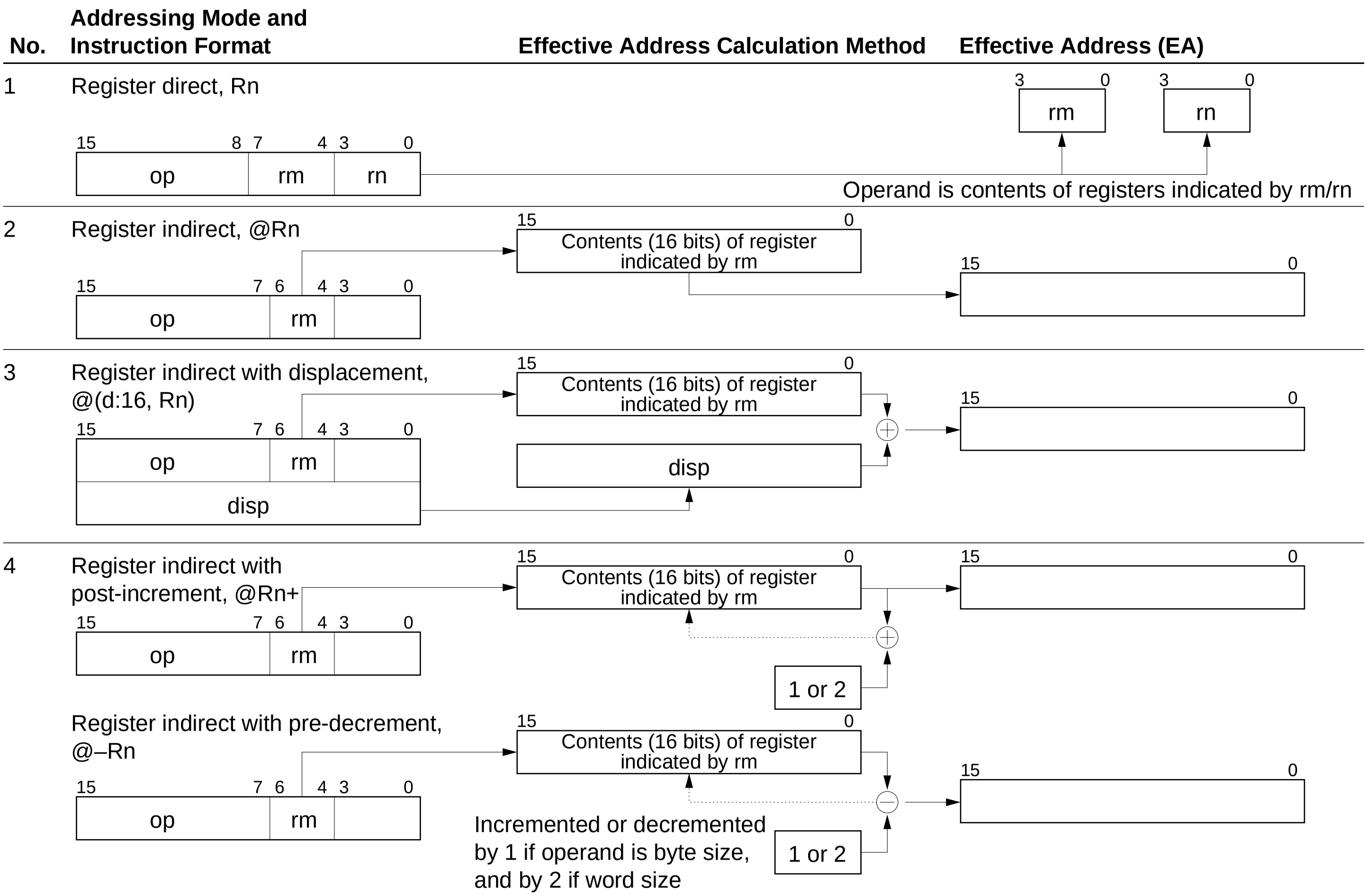
Table 2.2 shows how effective addresses are calculated in each of the addressing modes.

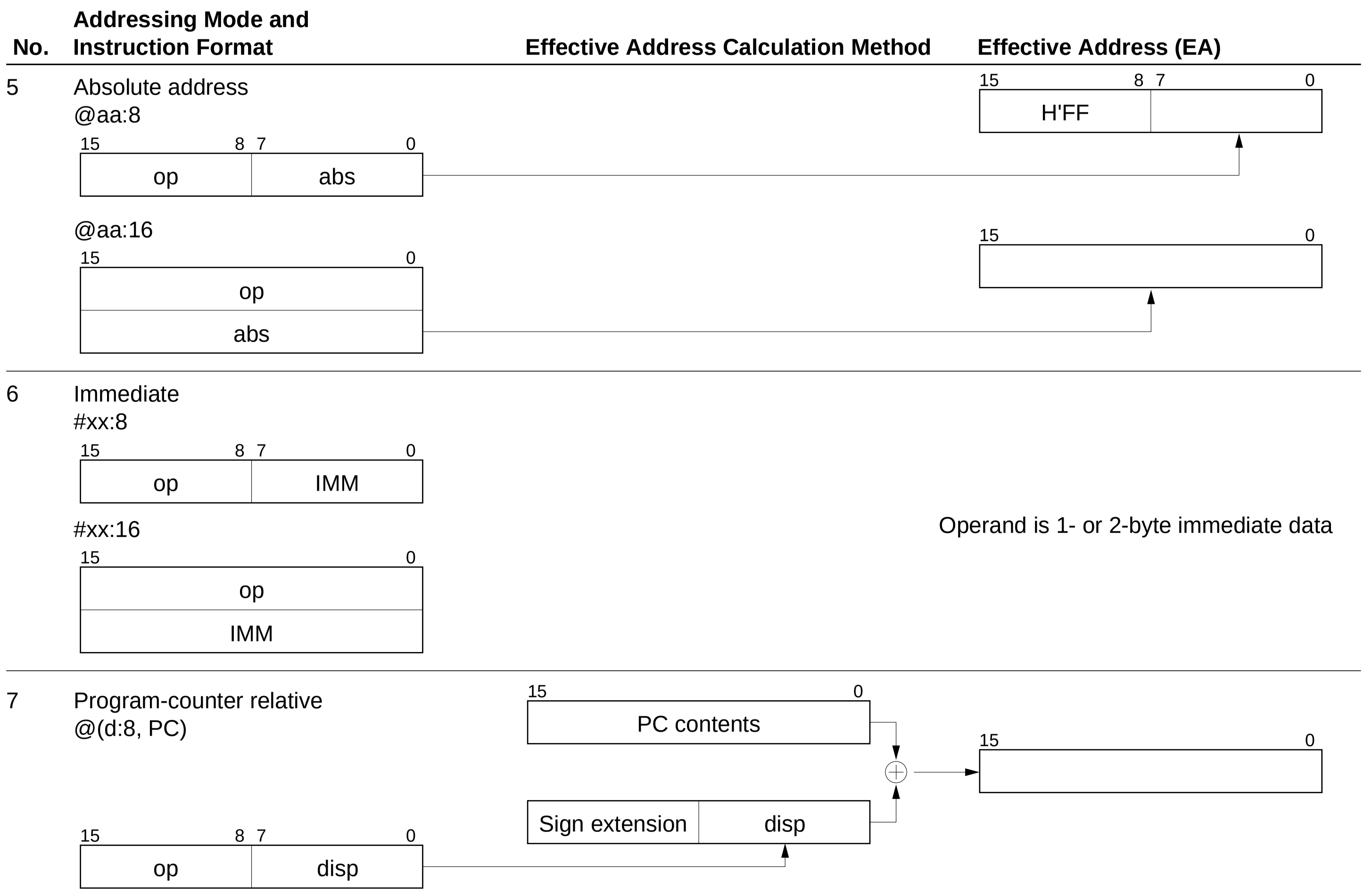
Arithmetic and logic instructions use register direct addressing (1). The ADD.B, ADDX, SUBX, CMP.B, AND, OR, and XOR instructions can also use immediate addressing (6).

Data transfer instructions can use all addressing modes except program-counter relative (7) and memory indirect (8).

Bit manipulation instructions can use register direct (1), register indirect (2), or 8-bit absolute addressing (5) to specify the operand. Register indirect (1) (BSET, BCLR, BNOT, and BTST instructions) or 3-bit immediate addressing (6) can be used independently to specify a bit position in the operand.

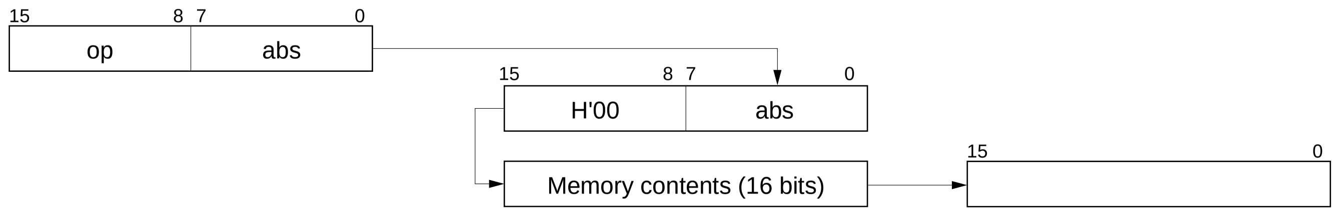
Table 2.2 Effective Address Calculation





No.
Addressing Mode and Instruction Format
Effective Address Calculation Method
Effective Address (EA)
5
Absolute address
@aa:8
15 8 7 0
op
abs
15 8 7 0
H'FF
@aa:16
15 0
op
abs
15 0
6
Immediate
#xx:8
15 8 7 0
op
IMM
#xx:16
15 0
op
IMM
Operand is 1- or 2-byte immediate data
7
Program-counter relative
@(d:8, PC)
15 8 7 0
op
disp
15 0
PC contents
Sign extension
disp
15 0

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



Notation:

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 same applies to the machine language.
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.

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

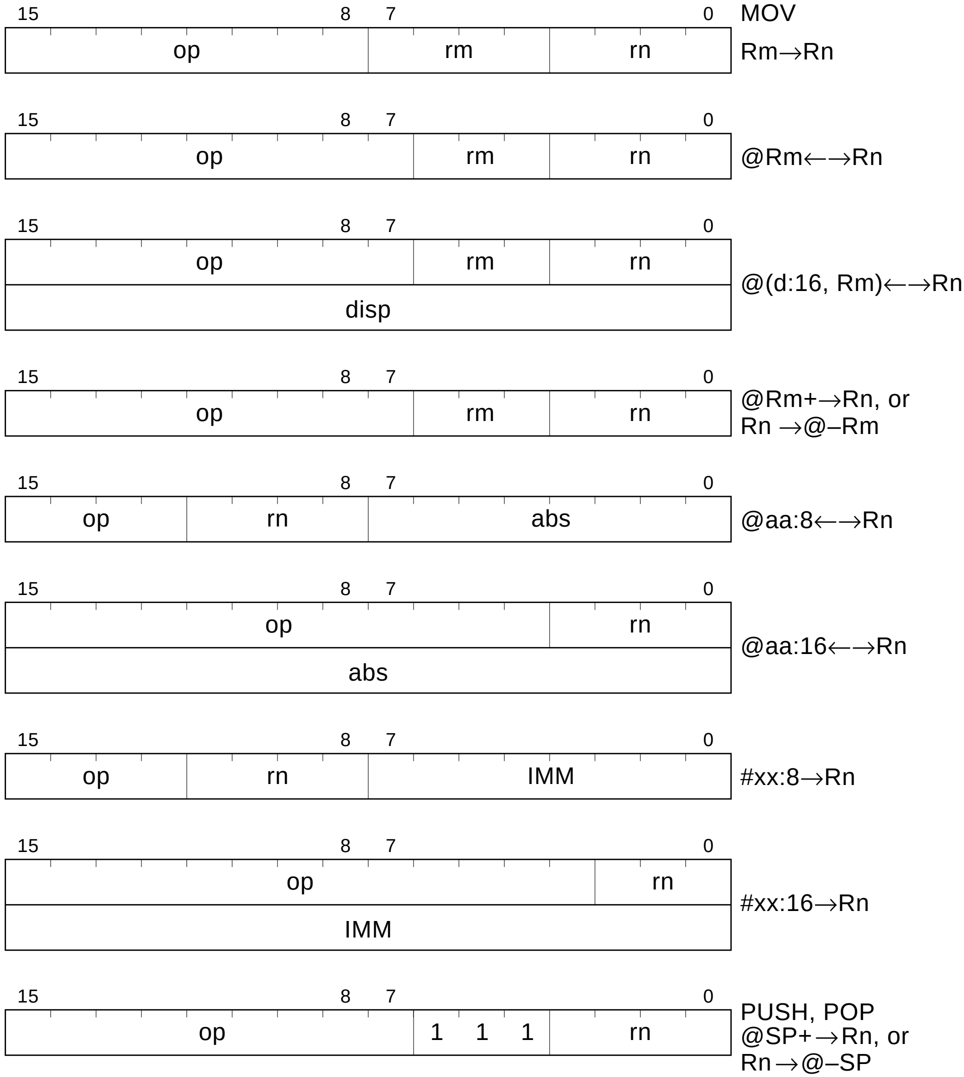
Table 2.4 describes the data transfer instructions. Figure 2.5 shows their object code formats.

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 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.

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

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



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

Figure 2.5 Data Transfer Instruction Codes

2.5.2 Arithmetic Operations

Table 2.5 describes the arithmetic instructions.

Table 2.5 Arithmetic Instructions

Instruction	Size*	Function
ADD SUB	B/W	Rd ± Rs → Rd, Rd + #IMM → Rd Performs addition or subtraction on data in two general registers, or addition on immediate data and data in a general register. Immediate data cannot be subtracted from data in a general register. Word data can be added or subtracted only when both words are in general registers.
ADDX SUBX	B	Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd Performs addition or subtraction with carry or borrow on byte data in two general registers, or addition or subtraction on immediate data and data in a general register.
INC DEC	B	Rd ± 1 → Rd Increments or decrements a general register by 1.
ADDS SUBS	W	Rd ± 1 → Rd, Rd ± 2 → Rd Adds or subtracts 1 or 2 to or from a general register
DAA DAS	B	Rd decimal adjust → Rd Decimal-adjusts (adjusts to 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 indicates the result 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

Notes: * 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	$Rd \wedge Rs \rightarrow Rd$, $Rd \wedge \#IMM \rightarrow Rd$ Performs a logical AND operation on a general register and another general register or immediate data
OR	B	$Rd \vee Rs \rightarrow Rd$, $Rd \vee \#IMM \rightarrow Rd$ Performs a logical OR operation on a general register and another general register or immediate data
XOR	B	$Rd \oplus Rs \rightarrow Rd$, $Rd \oplus \#IMM \rightarrow Rd$ Performs a logical exclusive OR operation on a general register and another general register or immediate data
NOT	B	$\sim Rd \rightarrow Rd$ Obtains the one's complement (logical complement) of general register contents

Notes: * Size: Operand size
B: Byte

2.5.4 Shift Operations

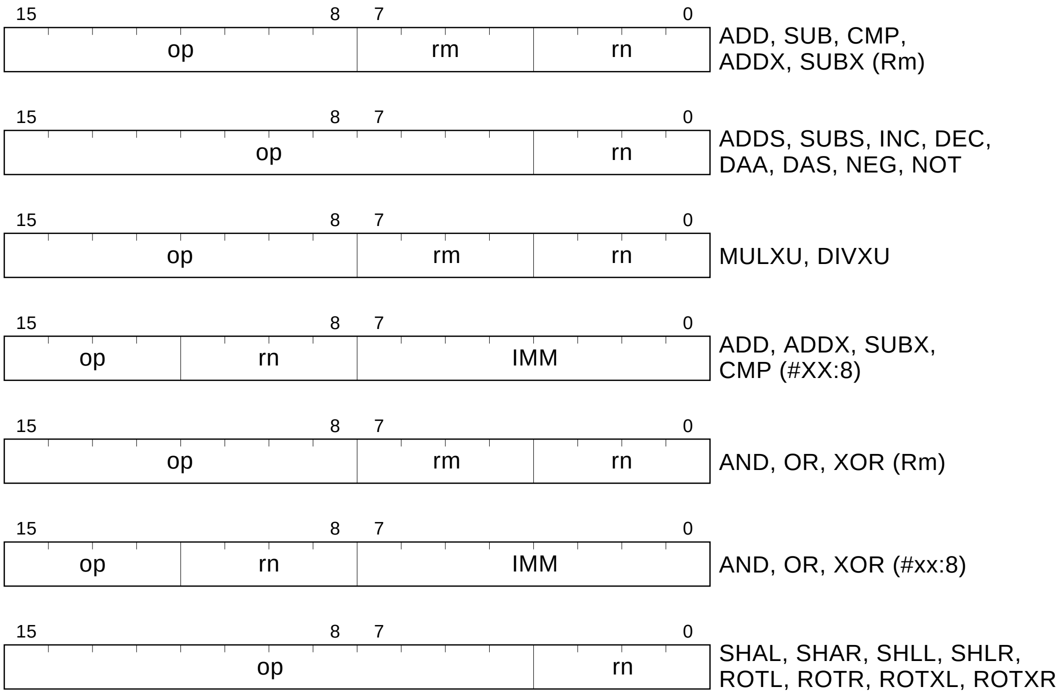
Table 2.7 describes the eight shift instructions.

Table 2.7 Shift Instructions

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

Notes: * Size: Operand size
B: Byte

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



Notation:

op: Operation field

rm, rn: Register field

IMM: Immediate data

Figure 2.6 Arithmetic, Logic, and Shift Instruction Codes

2.5.5 Bit Manipulations

Table 2.8 describes the bit-manipulation instructions. Figure 2.7 shows their object code formats.

Table 2.8 Bit-Manipulation Instructions

Instruction	Size*	Function
BSET	B	1 → (<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.

Notes: * Size: Operand size
B: Byte

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.

Notes: * 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

15–8		7–0	
op		IMM	rn

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

15–8	7–0	
op	rm	rn

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

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

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

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

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

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

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

15–8	7–0				
op	abs				
op	rm	0	0	0	0

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

BAND, BOR, BXOR, BLD, BST

15–8		7–0	
op		IMM	rn

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

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

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

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

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

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

Figure 2.7 Bit Manipulation Instruction Codes

BIAND, BIOR, BIXOR, BILD, BIST

15	8	7	0	
op		IMM	rn	Operand: register direct (Rn) Bit No.: immediate (#xx:3)

15	8	7				0	
op		rn	0	0	0	0	Operand: register indirect (@Rn)
op		IMM	0	0	0	0	Bit No.: immediate (#xx:3)

15	8	7				0	
op		abs					Operand: absolute (@aa:8)
op		IMM	0	0	0	0	Bit No.: immediate (#xx:3)

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

Figure 2.7 Bit Manipulation Instruction Codes (cont)

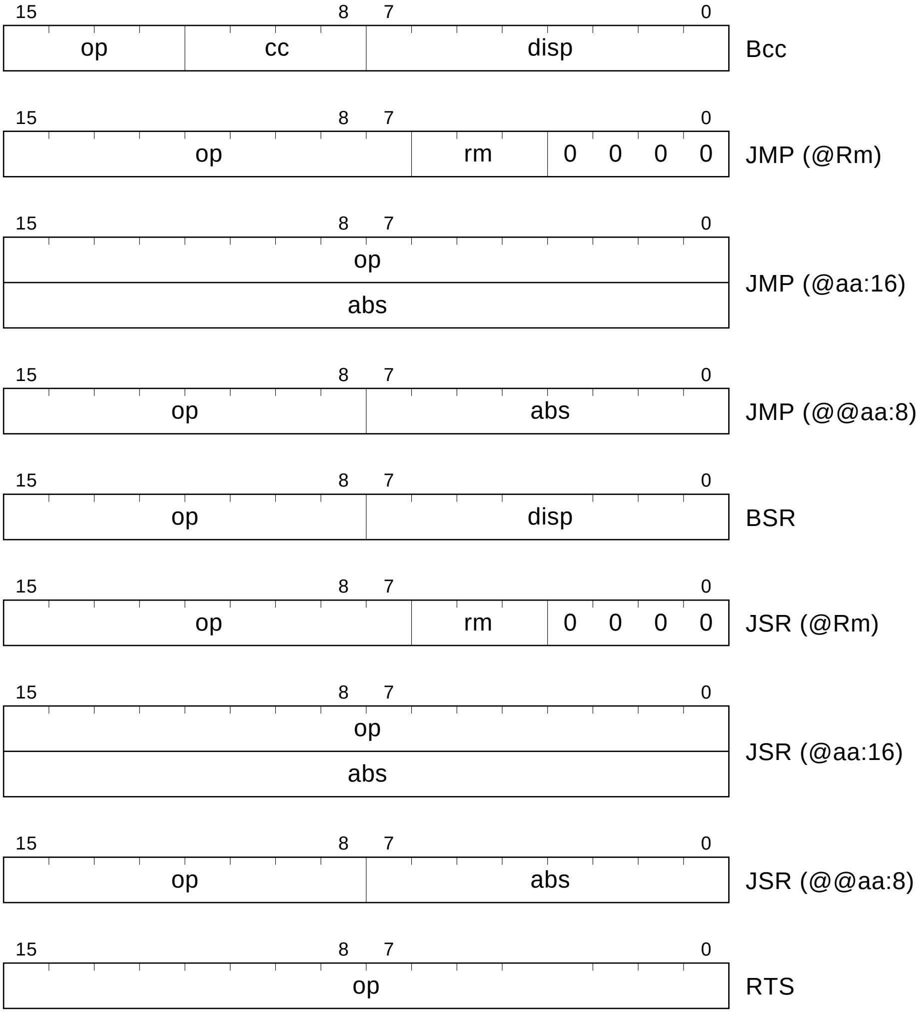
2.5.6 Branching Instructions

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 condition cc is true. The branching conditions are given below.
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

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	$Z \vee (N \oplus V) = 1$



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

Figure 2.8 Branching Instruction Codes

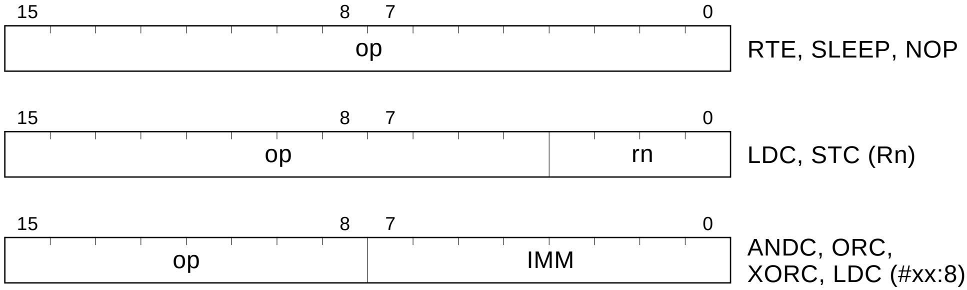
2.5.7 System Control Instructions

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 $\wedge$ #IMM → CCR Logically ANDs the condition code register with immediate data
ORC	B	CCR $\vee$ #IMM → CCR Logically ORs the condition code register with immediate data
XORC	B	CCR $\oplus$ #IMM → CCR Logically exclusive-ORs the condition code register with immediate data
NOP	—	PC + 2 → PC Only increments the program counter

Notes: * Size: Operand size
B: Byte



Notation:
op: Operation field
rn: Register field
IMM: Immediate data

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 data bytes specified by R4L from locations starting at the address indicated by R5 to locations starting at the address indicated by R6. After 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.

15 … 8	7 … 0
op	
op	

Notation:
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)

Access 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.

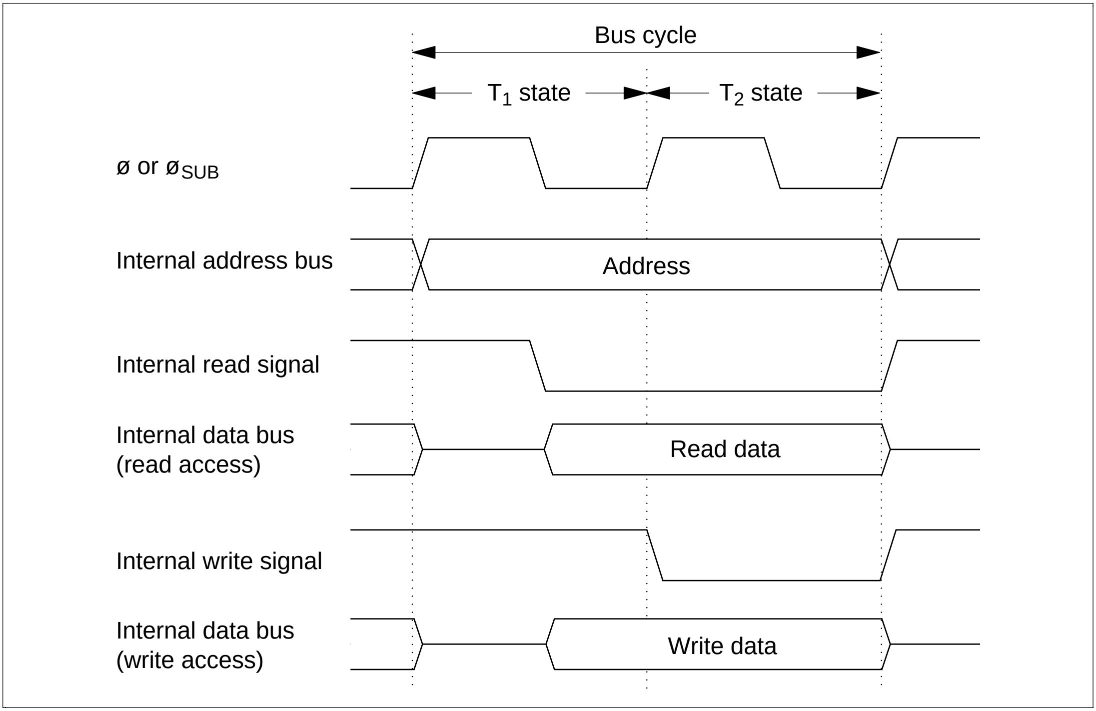


Figure 2.11 On-Chip Memory Access Cycle

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. Figures 2.12 and 2.13 show the on-chip peripheral module access cycle.

Two-state access to on-chip peripheral modules

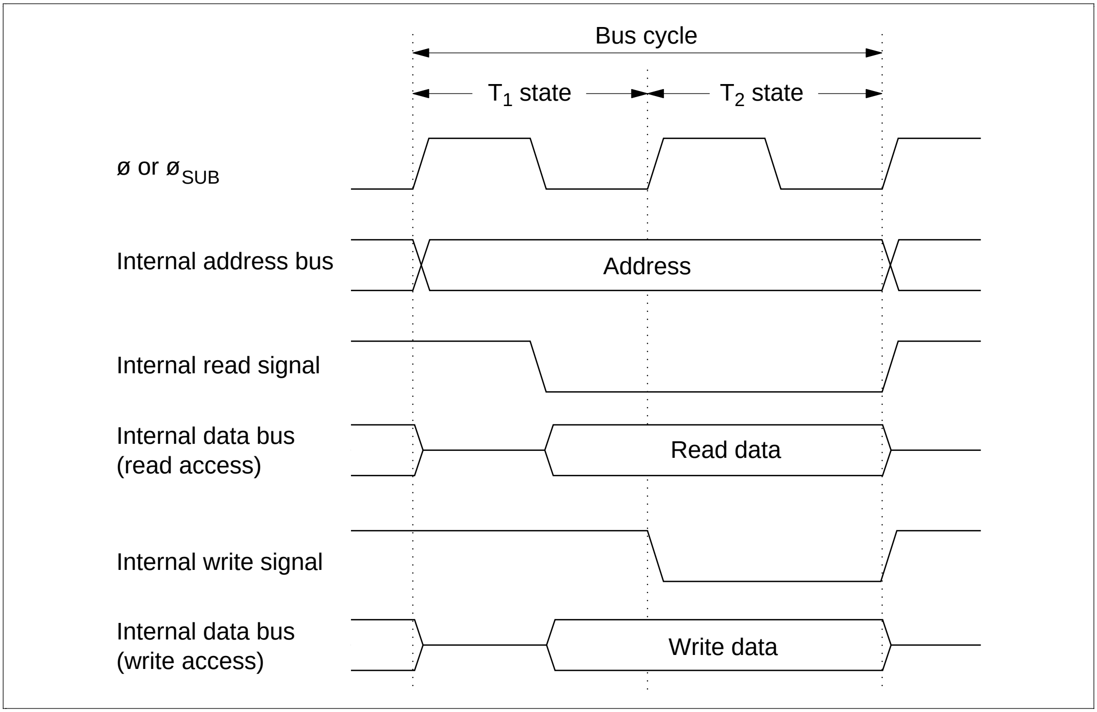


Figure 2.12 On-Chip Peripheral Module Access Cycle (2-State Access)

Three-state access to on-chip peripheral modules

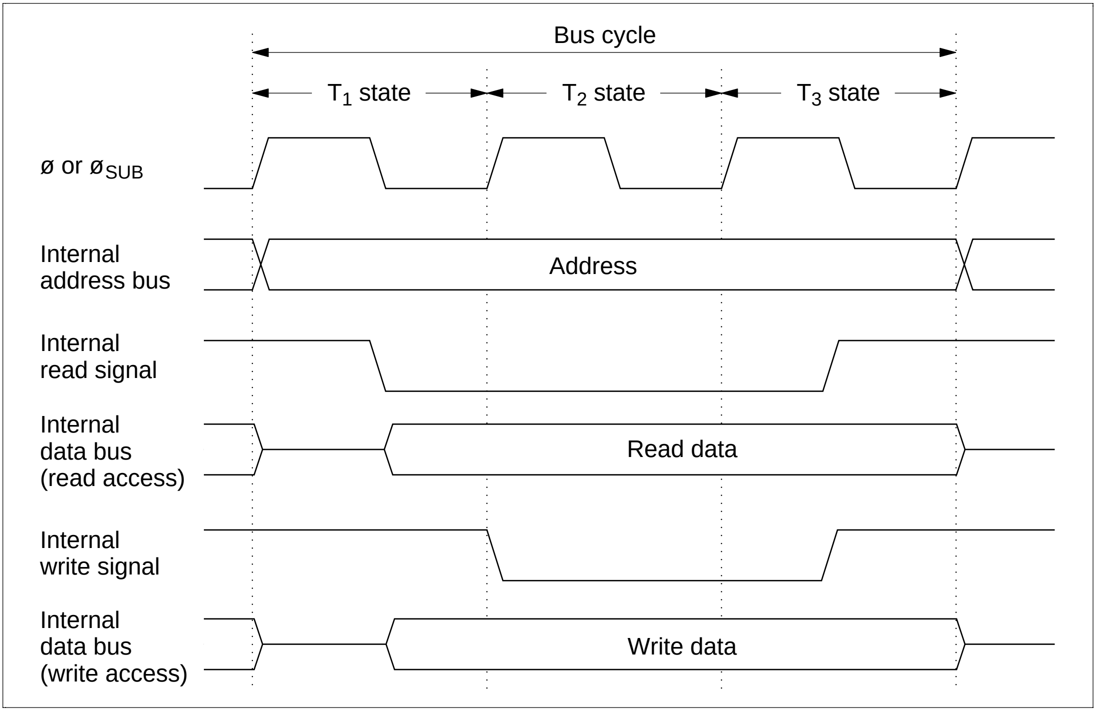
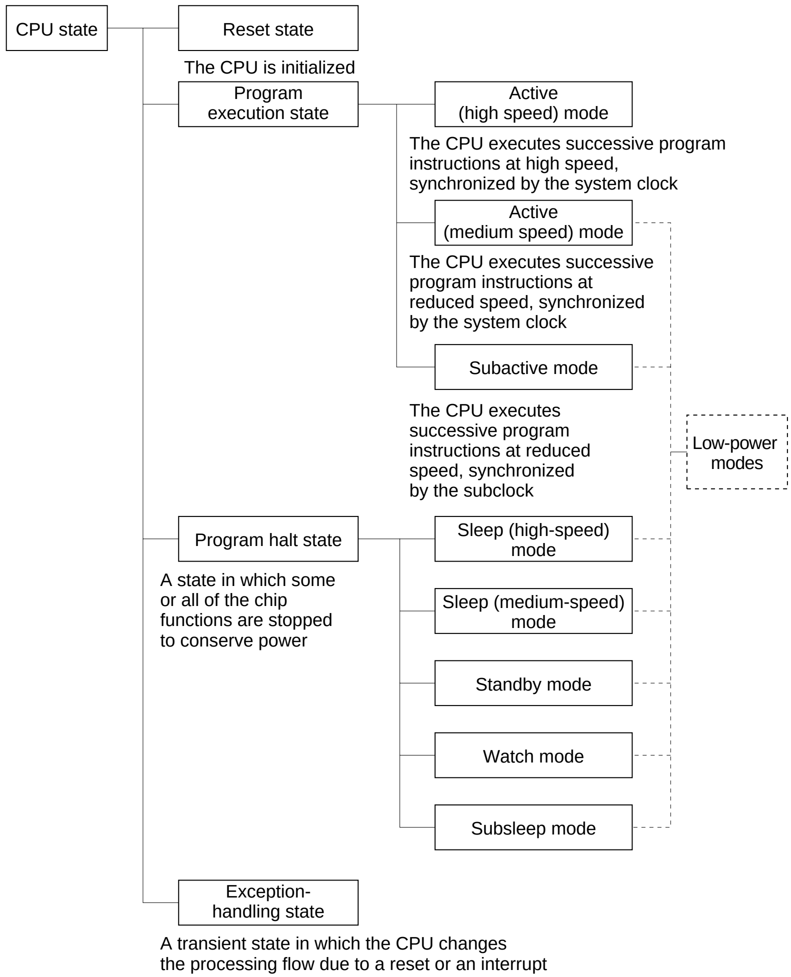


Figure 2.13 On-Chip Peripheral Module Access Cycle (3-State Access)

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 (high-speed or medium-speed) 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

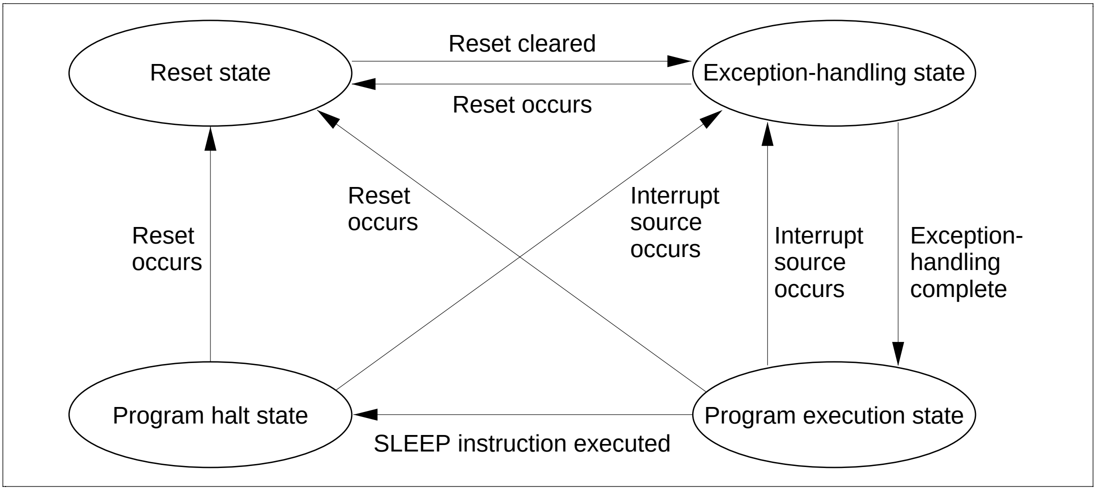


Figure 2.15 State Transitions

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 five modes: two sleep modes (high speed and medium speed), 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 section 3.3, Interrupts.

2.8 Memory Map

2.8.1 Memory Map

The memory map of the H8/3802 is shown in figure 2.16 (1), that of the H8/3801 in figure 2.16 (2), and that of the H8/3800 in figure 2.16 (3).

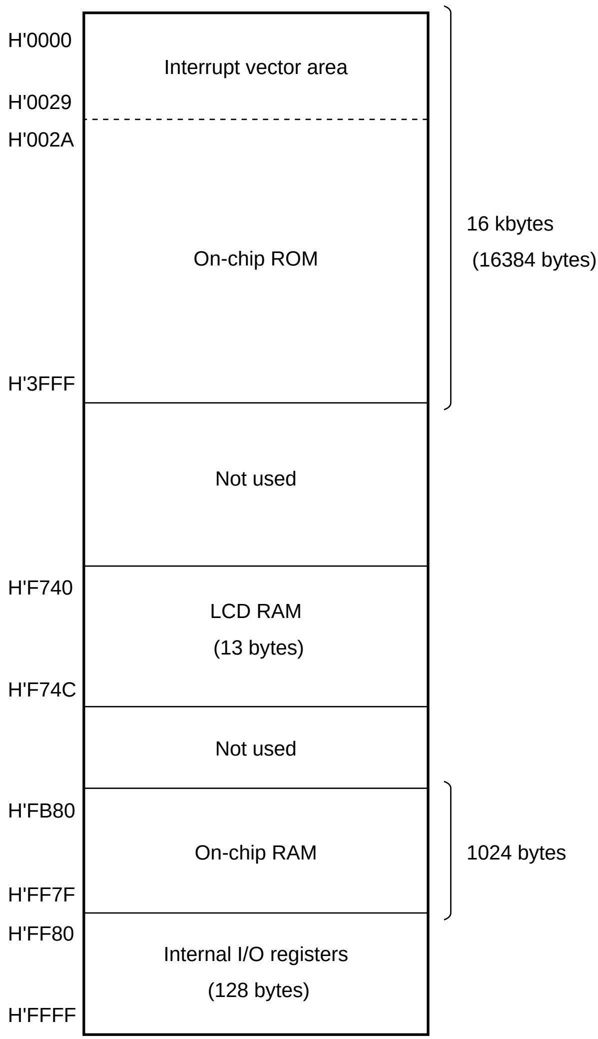


Figure 2.16 (1) H8/3802 Memory Map

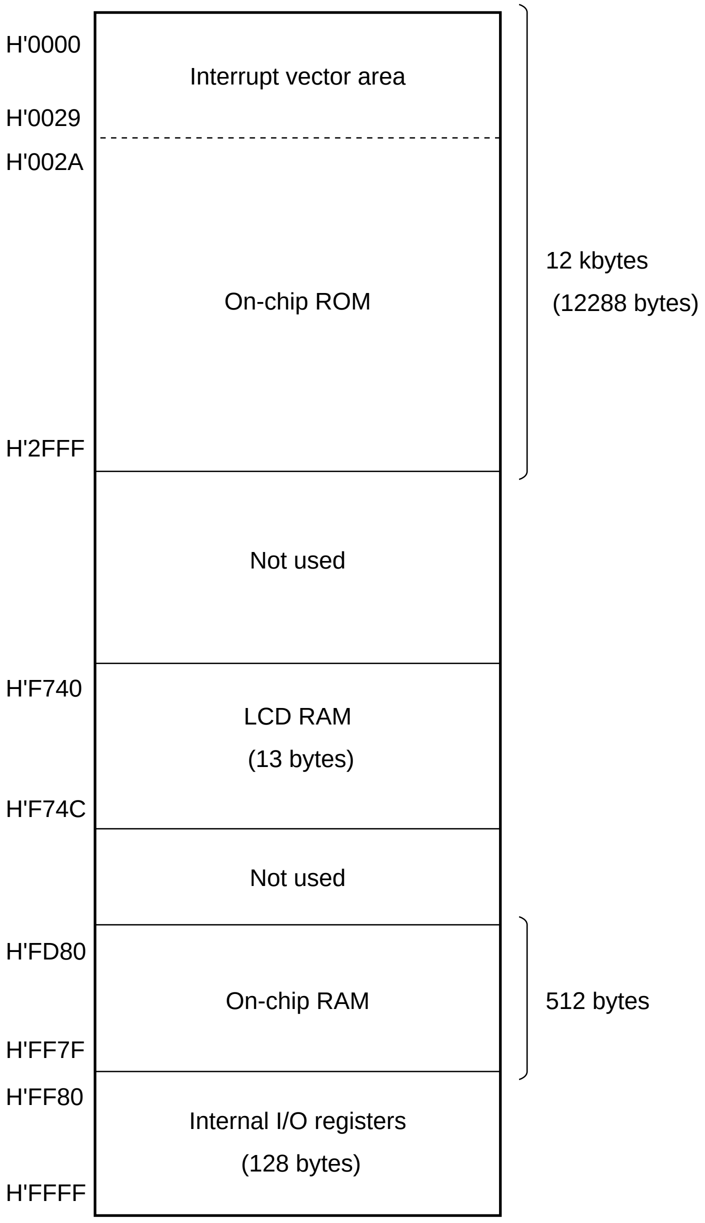


Figure 2.16 (2)   H8/3801 Memory Map

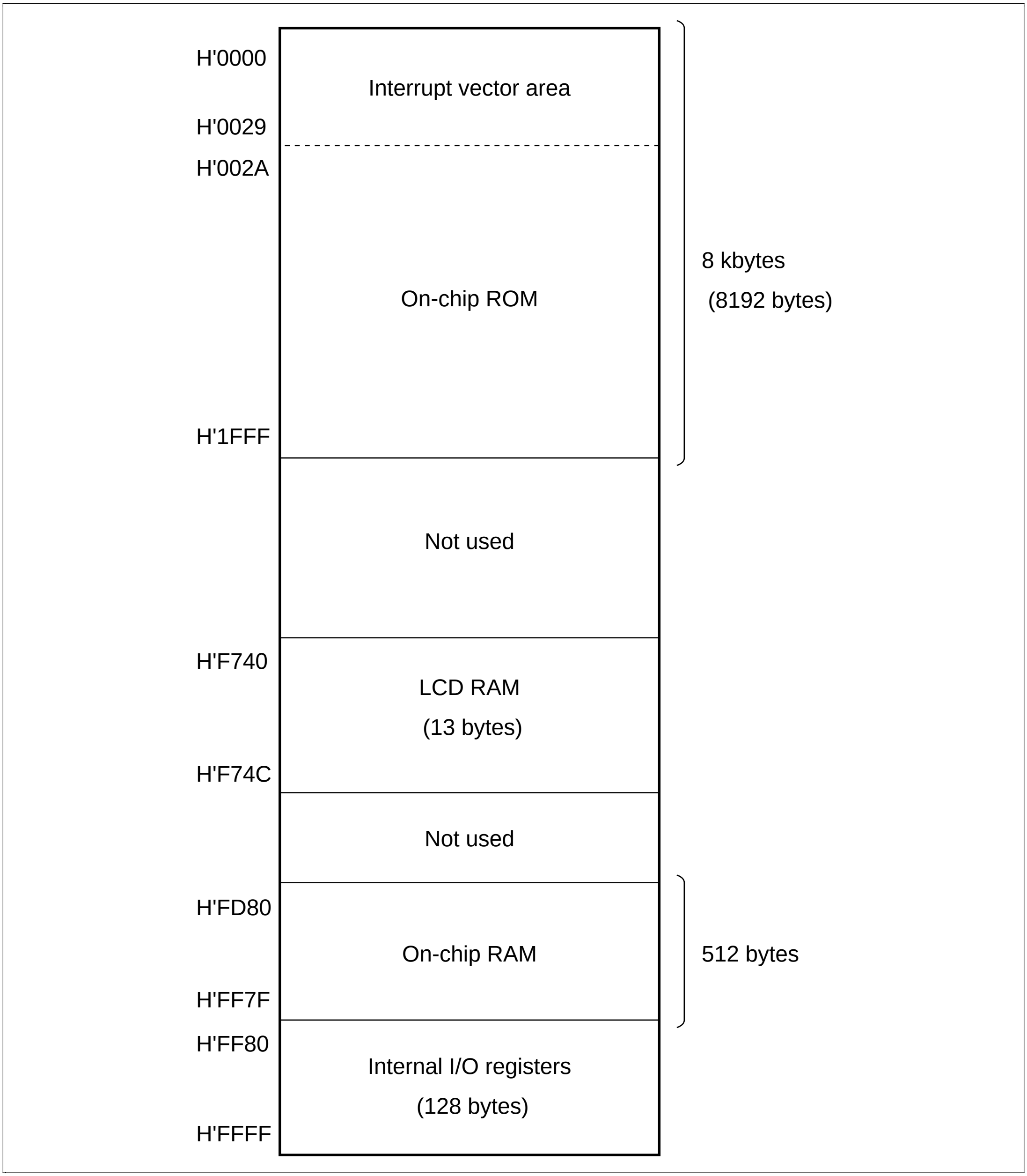


Figure 2.16 (3) H8/3800 Memory Map

2.9 Application Notes

2.9.1 Notes on Data Access

1. Access to Empty Areas:

   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.

2. Access to Internal I/O Registers:

   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.

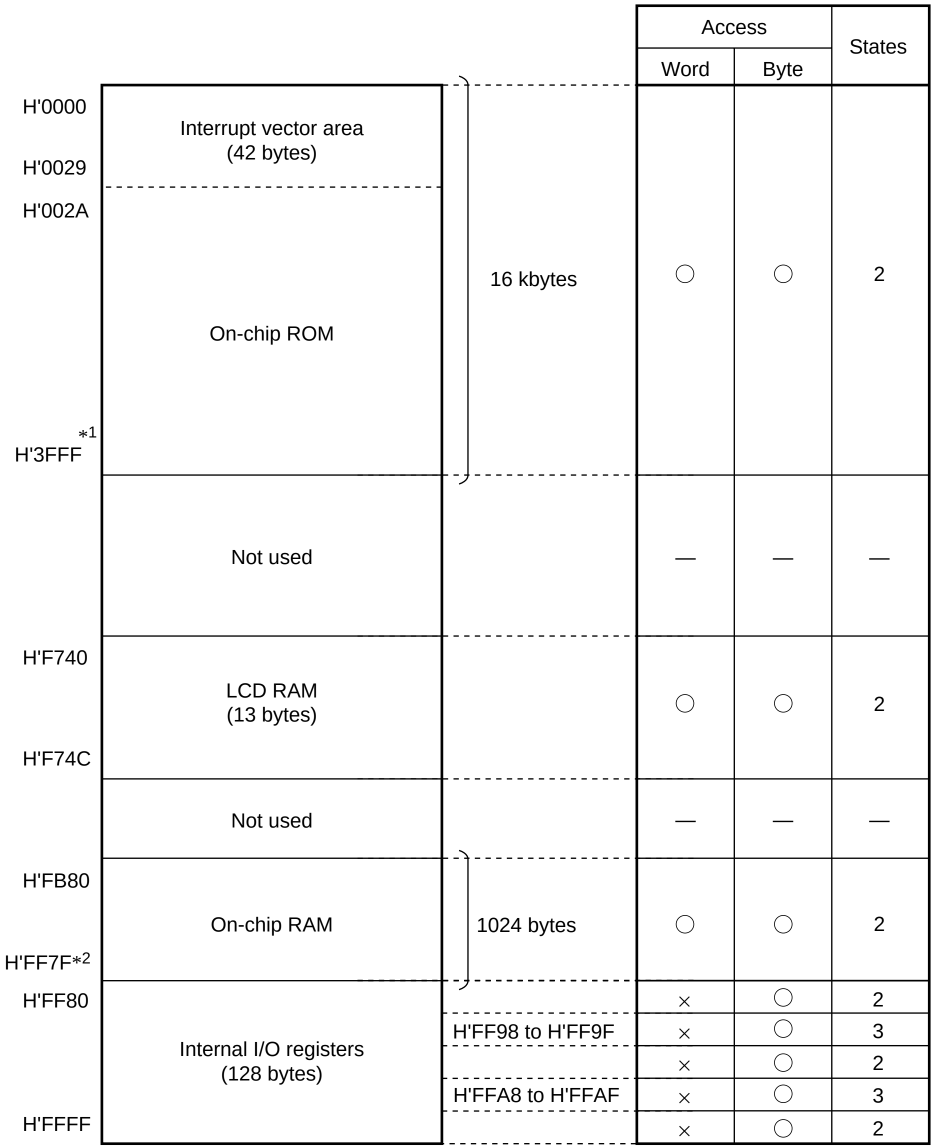
      Lower byte: Transferred data will be lost.

   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.



Notes: The example of the H8/3802 is shown here.

1. This address is H'3FFF in the H8/3802 (16-kbyte on-chip ROM), H'2FFF in the H8/3801 (12-kbyte on-chip ROM), H'1FFF in the H8/3800 (8-kbyte on-chip ROM).
2. This address is H'FD80 to H'FF7F in the H8/3801 and H8/3800 (512 bytes of on-chip RAM).

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 port.

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

1. Bit manipulation in two registers assigned to the same address

Example 1: timer load register and 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.

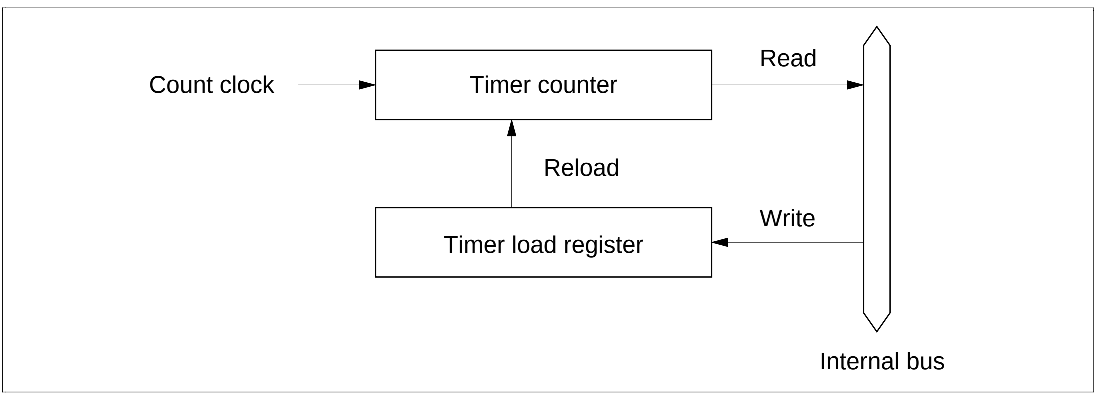


Figure 2.18 Timer Configuration Example

Example 2: BSET instruction executed designating port 3

$P3_7$ and $P3_6$ are designated as input pins, with a low-level signal input at $P3_7$ and a high-level signal at $P3_6$. The remaining pins, $P3_5$ to $P3_1$, are output pins and output low-level signals. In this example, the BSET instruction is used to change pin $P3_1$ to high-level output.

[A: Prior to executing BSET]

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	—
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	1

[B: BSET instruction executed]

```
BSET    #1  ,  @PDR3
```

The BSET instruction is executed designating port 3.

[C: After executing BSET]

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	High level	—
PCR3	0	0	1	1	1	1	1	1
PDR3	0	1	0	0	0	0	1	1

[D: Explanation of how BSET operates]

When the BSET instruction is executed, first the CPU reads port 3.

Since $P3_7$ and $P3_6$ are input pins, the CPU reads the pin states (low-level and high-level input). $P3_5$ to $P3_1$ are output pins, so the CPU reads the value in PDR3. In this example PDR3 has a value of H'81, but the value read by the CPU is H'41.

Next, the CPU sets bit 1 of the read data to 1, changing the PDR3 data to H'43. Finally, the CPU writes this value (H'43) to PDR3, completing execution of BSET.

As a result of this operation, bit 1 in PDR3 becomes 1, and $P3_1$ outputs a high-level signal. However, bits 7 and 6 of PDR3 end up with different values.

To avoid this problem, store a copy of the PDR3 data in a work area in memory. Perform the bit manipulation on the data in the work area, then write this data to PDR3.

[A: Prior to executing BSET]

```
MOV. B  #81,  R0L
MOV. B  R0L,  @RAM0
MOV. B  R0L,  @PDR3
```

The PDR3 value (H'81) is written to a work area in memory (RAM0) as well as to PDR3

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	—
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	1
RAM0	1	0	0	0	0	0	0	1

[B: BSET instruction executed]

```
BSET   #1  ,  @RAM0
```

The BSET instruction is executed designating the PDR3 work area (RAM0).

[C: After executing BSET]

```
MOV. B  @RAM0, R0L
MOV. B  R0L,  @PDR3
```

The work area (RAM0) value is written to PDR3.

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	High level	—
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	1	1
RAM0	1	0	0	0	0	0	1	1

2. Bit manipulation in a register containing a write-only bit

Example 3: BCLR instruction executed designating port 3 control register PCR3

As in the examples above, $P3_7$ and $P3_6$ are input pins, with a low-level signal input at $P3_7$ and a high-level signal at $P3_6$. The remaining pins, $P3_5$ to $P3_1$, are output pins that output low-level signals. In this example, the BCLR instruction is used to change pin $P3_1$ to an input port. It is assumed that a high-level signal will be input to this input pin.

[A: Prior to executing BCLR]

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	—
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	1

[B: BCLR instruction executed]

```
BCLR    #1   ,   @PCR3
```

The BCLR instruction is executed designating PCR3.

[C: After executing BCLR]

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Output	Output	Output	Output	Output	Output	Input	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	High level	—
PCR3	1	1	1	1	1	1	0	1
PDR3	1	0	0	0	0	0	0	1

[D: Explanation of how BCLR operates]

When the BCLR instruction is executed, first the CPU reads PCR3. Since PCR3 is a write-only register, the CPU reads a value of H'FF, even though the PCR3 value is actually H'3F.

Next, the CPU clears bit 1 in the read data to 0, changing the data to H'FD. Finally, this value (H'FD) is written to PCR3 and BCLR instruction execution ends.

As a result of this operation, bit 1 in PCR3 becomes 0, making $P3_1$ an input port. However, bits 7 and 6 in PCR3 change to 1, so that $P3_7$ and $P3_6$ change from input pins to output pins.

To avoid this problem, store a copy of the PCR3 data in a work area in memory. Perform the bit manipulation on the data in the work area, then write this data to PCR3.

[A: Prior to executing BCLR]

```
MOV. B  #3F,  R0L
MOV. B  R0L,  @RAM0
MOV. B  R0L,  @PCR3
```

The PCR3 value (H'3F) is written to a work area in memory (RAM0) as well as to PCR3.

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	—
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	1
RAM0	0	0	1	1	1	1	1	1

[B: BCLR instruction executed]

```
BCLR   #1  ,  @RAM0
```

The BCLR instruction is executed designating the PCR3 work area (RAM0).

[C: After executing BCLR]

```
MOV. B  @RAM0, R0L
MOV. B  R0L,  @PCR3
```

The work area (RAM0) value is written to PCR3.

	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	—
Input/output	Input	Input	Output	Output	Output	Output	Output	—
Pin state	Low level	High level	Low level	Low level	Low level	Low level	High level	—
PCR3	0	0	1	1	1	1	0	1
PDR3	1	0	0	0	0	0	0	1
RAM0	0	0	1	1	1	1	0	1

Table 2.12 lists the pairs of registers that share identical 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 3*	PDR3	H'FFD6
Port data register 4*	PDR4	H'FFD7
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

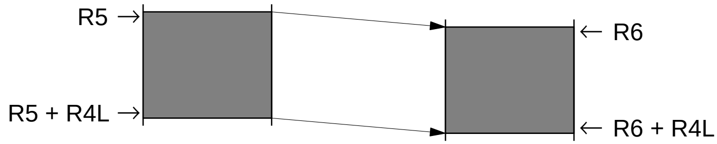
Note: * Port data registers have the same addresses as input pins.

Table 2.13 Registers with Write-Only Bits

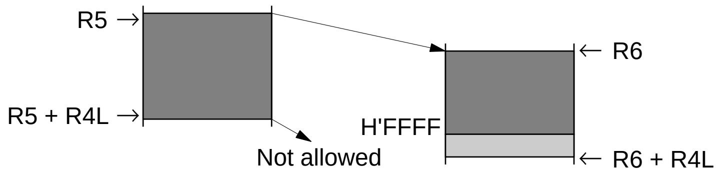
Register Name	Abbreviation	Address
Port control register 3	PCR3	H'FFE6
Port control register 4	PCR4	H'FFE7
Port control register 5	PCR5	H'FFE8
Port control register 6	PCR6	H'FFE9
Port control register 7	PCR7	H'FFEA
Port control register 8	PCR8	H'FFEB
Port control register A	PCRA	H'FFED
Timer control register F	TCRF	H'FFB6
PWM1 control register	PWCR1	H'FFD0
PWM1 data register U	PWDRU1	H'FFD1
PWM1 data register L	PWDRL1	H'FFD2
PWM2 control register	PWCR2	H'FFCD
PWM2 data register U	PWDRU2	H'FFCE
PWM2 data register L	PWDRL2	H'FFCF

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 H8/3802 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
↑ Low	Interrupt	When an interrupt is requested, exception handling starts after execution of the present instruction or the exception handling in progress is completed

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 chip 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.
- Resetting during operation: Hold the $\overline{\text{RES}}$ pin low for at least 10 system clock cycles.

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 starting from $\overline{\text{RES}}$ input.

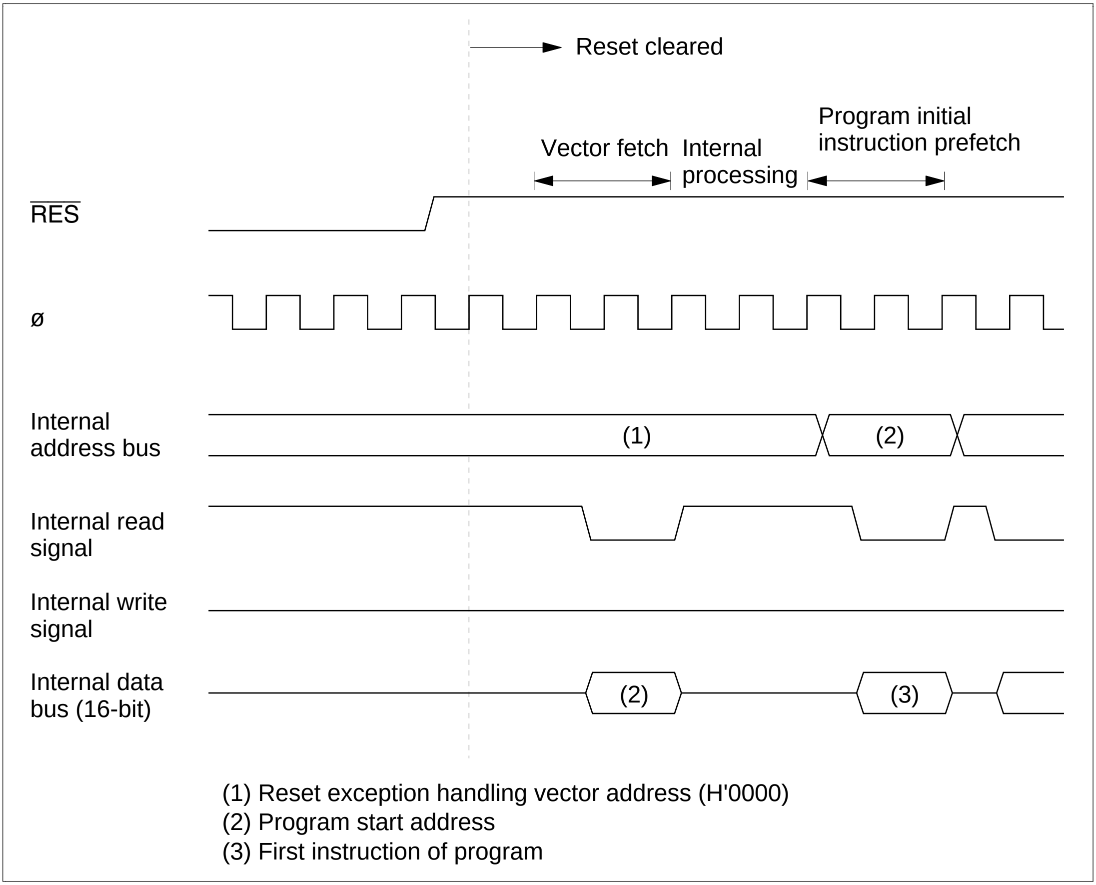


Figure 3.1 Reset Sequence

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 11 external interrupts (WKP_7 to WKP_0, IRQ_1 to IRQ_0, IRQAEC) and 7 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 in CCR. When the I bit is set to 1, interrupt request flags can be set but the interrupts are not accepted.
- IRQAEC, IRQ_1 to IRQ_0, and WKP_7 to WKP_0 can be set to either rising edge sensing or falling edge sensing.

Table 3.2 Interrupt Sources and Their 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	
IRQAEC	IRQAEC	6	H'000C to H'000D	
$\overline{WKP_0}$ $\overline{WKP_1}$ $\overline{WKP_2}$ $\overline{WKP_3}$ $\overline{WKP_4}$ $\overline{WKP_5}$ $\overline{WKP_6}$ $\overline{WKP_7}$	WKP_0 WKP_1 WKP_2 WKP_3 WKP_4 WKP_5 WKP_6 WKP_7	9	H'0012 to H'0013	
Timer A	Timer A overflow	11	H'0016 to H'0017	
Asynchronous event counter	Asynchronous event counter overflow	12	H'0018 to H'0019	
Timer FL	Timer FL compare match Timer FL overflow	14	H'001C to H'001D	
Timer FH	Timer FH compare match Timer FH overflow	15	H'001E to H'001F	
SCI3	SCI3 transmit end SCI3 transmit data empty SCI3 receive data full SCI3 overrun error SCI3 framing error SCI3 parity error	18	H'0024 to H'0025	
A/D	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'000E to H'0011, H'0014 to H'0015, H'001A to H'001B, and H'0020 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
IRQ edge select register	IEGR	R/W	—	H'FFF2
Interrupt enable register 1	IENR1	R/W	—	H'FFF3
Interrupt enable register 2	IENR2	R/W	—	H'FFF4
Interrupt request register 1	IRR1	R/W*	—	H'FFF6
Interrupt request register 2	IRR2	R/W*	—	H'FFF7
Wakeup interrupt request register	IWPR	R/W*	H'00	H'FFF9
Wakeup edge select register	WEGR	R/W	H'00	H'FF90

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

1. IRQ edge select register (IEGR)

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

IEGR is an 8-bit read/write register used to designate whether pins $\overline{IRQ_1}$ and $\overline{IRQ_0}$ are set to rising edge sensing or falling edge sensing.

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 2: Reserved bits

Bits 4 to 2 are reserved; only 0 can be written to these bits.

Bit 1: IRQ_1 edge select (IEG1)

Bit 1 selects the input sensing of the $\overline{IRQ_1}$ pin.

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: 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	

2. Interrupt enable register 1 (IENR1)

Bit	7	6	5	4	3	2	1	0
	IENTA	—	IENWP	—	—	IENEC2	IEN1	IEN0
Initial value	0	—	0	—	—	0	0	0
Read/Write	R/W	W	R/W	W	W	R/W	R/W	R/W

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

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 interrupt requests	(initial value)
1	Enables timer A interrupt requests	

Bit 6: Reserved bit

Bit 6 is reserved; only 0 can be written to this bit.

Bit 5: Wakeup interrupt enable (IENWP)

Bit 5 enables or disables WKP_7 to WKP_0 interrupt requests.

Bit 5 IENWP	Description	
0	Disables $\overline{WKP_7}$ to $\overline{WKP_0}$ interrupt requests	(initial value)
1	Enables $\overline{WKP_7}$ to $\overline{WKP_0}$ interrupt requests	

Bits 4 and 3: Reserved bits

Bits 4 and 3 are reserved; only 0 can be written to these bits.

Bit 2: IRQAEC interrupt enable (IENEC2)

Bit 2 enables or disables IRQAEC interrupt requests.

Bit 2 IENEC2	Description	
0	Disables IRQAEC interrupt requests	(initial value)
1	Enables IRQAEC interrupt requests	

Bits 1 and 0: IRQ_1 and IRQ_0 interrupt enable (IEN1 and IEN0)

Bits 1 and 0 enable or disable IRQ_1 and IRQ_0 interrupt requests.

Bit n IENn	Description	
0	Disables interrupt requests from pin $\overline{IRQn}$	(initial value)
1	Enables interrupt requests from pin $\overline{IRQn}$	

(n = 1 or 0)

3. Interrupt enable register 2 (IENR2)

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

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

Bits 5 and 4: Reserved bits

Bits 5 and 4 are reserved; only 0 can be written to these bits.

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 interrupt requests	(initial value)
1	Enables timer FH interrupt requests	

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 interrupt requests	(initial value)
1	Enables timer FL interrupt requests	

Bit 1: Reserved bit

Bit 1 is reserved; only 0 can be written to this bit.

Bit 0: Asynchronous event counter interrupt enable (IENEC)

Bit 0 enables or disables asynchronous event counter interrupt requests.

Bit 0 IENEC	Description	
0	Disables asynchronous event counter interrupt requests	(initial value)
1	Enables asynchronous event counter interrupt requests	

For details of SCI3 interrupt control, see 10.2.6. Serial control register 3 (SCR3).

4. Interrupt request register 1 (IRR1)

Bit	7	6	5	4	3	2	1	0
	IRRTA	—	—	—	—	IRREC2	IRRI1	IRRI0
Initial value	0	—	1	—	—	0	0	0
Read/Write	R/W*	W	—	W	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 a corresponding flag is set to 1 when a timer A, IRQAEC or IRQ_1, 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 from H'FF to H'00	

Bits 6, 4, and 3: Reserved bits

Bits 6, 4, and 3 are reserved; only 0 can be written to these bits.

Bit 5: Reserved bit

Bit 5 is reserved; it is always read as 1 and cannot be modified.

Bit 2: IRQAEC interrupt request flag (IRREC2)

Bit 2 IRREC2	Description
0	Clearing conditions: (initial value) When IRREC2 = 1, it is cleared by writing 0
1	Setting conditions: When pin IRQAEC is designated for interrupt input and the designated signal edge is input

Bits 1 and 0: IRQ_1 and IRQ_0 interrupt request flags (IRRI1 and IRRI0)

Bit n IRRIn	Description
0	Clearing conditions: (initial value) When IRRIn = 1, it is cleared by writing 0
1	Setting conditions: When pin $\overline{\text{IRQn}}$ is designated for interrupt input and the designated signal edge is input

(n = 1 or 0)

5. Interrupt request register 2 (IRR2)

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

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

IRR2 is an 8-bit read/write register, in which a corresponding flag is set to 1 when a direct transfer, A/D converter, Timer FH, or Timer FL asynchronous event counter 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: Direct transfer interrupt request flag (IRRDT)

Bit 7 IRRDT	Description	
0	Clearing conditions: When IRRDT = 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When a direct transfer is made by executing a SLEEP instruction while DTON = 1 in SYSCR2	

Bit 6: A/D converter interrupt request flag (IRRAD)

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

Bits 5 and 4: Reserved bits

Bits 5 and 4 are reserved; only 0 can be written to these bits.

Bit 3: Timer FH interrupt request flag (IRRTFH)

Bit 3 IRRTFH	Description	
0	Clearing conditions: When IRRTFH = 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When TCFH and OCRFH match in 8-bit timer mode, or when TCF (TCFL, TCFH) and OCRF (OCRFL, OCRFH) match in 16-bit timer mode	

Bit 2: Timer FL interrupt request flag (IRRTFL)

Bit 2 IRRTFL	Description	
0	Clearing conditions: When IRRTFL= 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When TCFL and OCRFL match in 8-bit timer mode	

Bit 1: Reserved bit

Bit 1 is reserved; only 0 can be written to this bit.

Bit 0: Asynchronous event counter interrupt request flag (IRREC)

Bit 0 IRREC	Description	
0	Clearing conditions: When IRREC = 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When ECH overflows in 16-bit counter mode, or ECH or ECL overflows in 8-bit counter mode	

6. 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 containing wakeup interrupt request flags. When one of pins $\overline{WKP}_7$ to $\overline{WKP}_0$ is designated for wakeup input and a rising or falling edge is input at that pin, the corresponding flag in IWPR is set to 1. A flag is not cleared automatically when the corresponding interrupt is accepted. Flags must be cleared by writing 0.

Bits 7 to 0: Wakeup interrupt request flags (IWPF7 to IWPF0)

Bit n IWPFn	Description
0	Clearing conditions: (initial value) When IWPFn= 1, it is cleared by writing 0
1	Setting conditions: When pin $\overline{WKP}_n$ is designated for wakeup input and a rising or falling edge is input at that pin

(n = 7 to 0)

7. Wakeup Edge Select Register (WEGR)

Bit	7	6	5	4	3	2	1	0
	WKEGS7	WKEGS6	WKEGS5	WKEGS4	WKEGS3	WKEGS2	WKEGS1	WKEGS0
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

WEGR is an 8-bit read/write register that specifies rising or falling edge sensing for pins $\overline{\text{WKP}}$n.

WEGR is initialized to H'00 by a reset.

Bit n: $\overline{\text{WKP}}$n edge select (WKEGSn)

Bit n selects $\overline{\text{WKP}}$n pin input sensing.

Bit n WKEGSn	Description	
0	$\overline{\text{WKP}}$n pin falling edge detected	(initial value)
1	$\overline{\text{WKP}}$n pin rising edge detected	

(n = 7 to 0)

3.3.3 External Interrupts

There are 11 external interrupts: WKP_7 to WKP_0, IRQ_1 to IRQ_0, and IRQAEC.

1. Interrupts WKP_7 to WKP_0

Interrupts WKP_7 to WKP_0 are requested by either rising or falling edge input to pins $\overline{WKP}_7$ to $\overline{WKP}_0$. When these pins are designated as pins $\overline{WKP}_7$ to $\overline{WKP}_0$ in port mode register 5 and a rising or falling edge is input, the corresponding bit in IWPR is set to 1, requesting an interrupt. Recognition of wakeup interrupt requests can be disabled by clearing the IENWP bit to 0 in IENR1. These interrupts can all be masked by setting the I bit to 1 in CCR.

When WKP_7 to WKP_0 interrupt exception handling is initiated, the I bit is set to 1 in CCR. Vector number 9 is assigned to interrupts WKP_7 to WKP_0. All eight interrupt sources have the same vector number, so the interrupt-handling routine must discriminate the interrupt source.

2. Interrupts IRQ_1 and IRQ_0

Interrupts IRQ_1 and IRQ_0 are requested by input signals to pins $\overline{IRQ_1}$ and $\overline{IRQ_0}$. These interrupts are detected by either rising edge sensing or falling edge sensing, depending on the settings of bits IEG1 and IEG0 in IEGR.

When these pins are designated as pins $\overline{IRQ_1}$ and $\overline{IRQ_0}$ in port mode register B and 2 and the designated edge is input, the corresponding bit in IRR1 is set to 1, requesting an interrupt. Recognition of these interrupt requests can be disabled individually by clearing bits IEN1 and IEN0 to 0 in IENR1. These interrupts can all be masked by setting the I bit to 1 in CCR.

When IRQ_1 and IRQ_0 interrupt exception handling is initiated, the I bit is set to 1 in CCR. Vector numbers 5 and 4 are assigned to interrupts IRQ_1 and IRQ_0. The order of priority is from IRQ_0 (high) to IRQ_1 (low). Table 3.2 gives details.

3. IRQAEC Interrupt

The IRQAEC interrupt is requested by an input signal to pin IRQAEC. This interrupt is detected by rising edge, falling edge, or both edge sensing, depending on the settings of bits AIAGS1 and AIAGS0 in AEGSR.

When bit IENEC2 in IENR1 is 1 and the designated edge is input, the corresponding bit in IRR1 is set to 1, requesting an interrupt.

When IRQAEC interrupt exception handling is initiated, the I bit is set to 1 in CCR. Vector number 6 is assigned to the IRQAEC interrupt. Table 3.2 gives details.

3.3.4 Internal Interrupts

There are 7 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. Recognition of individual interrupt requests can be disabled by clearing the corresponding bit in IENR1 or IENR2. All these interrupts can be masked by setting the I bit to 1 in CCR. When internal interrupt handling is initiated, the I bit is set to 1 in CCR. Vector numbers from 20 to 18, 15, 14, 12, and 11 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.

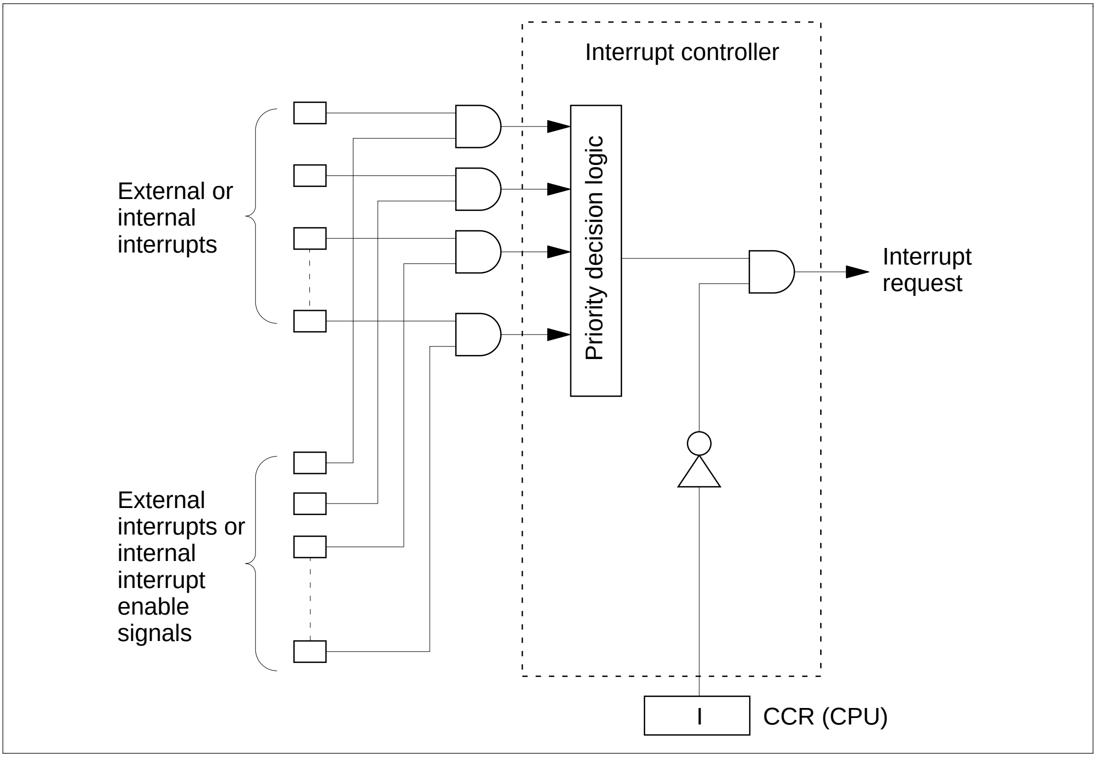


Figure 3.2 Block Diagram of Interrupt Controller

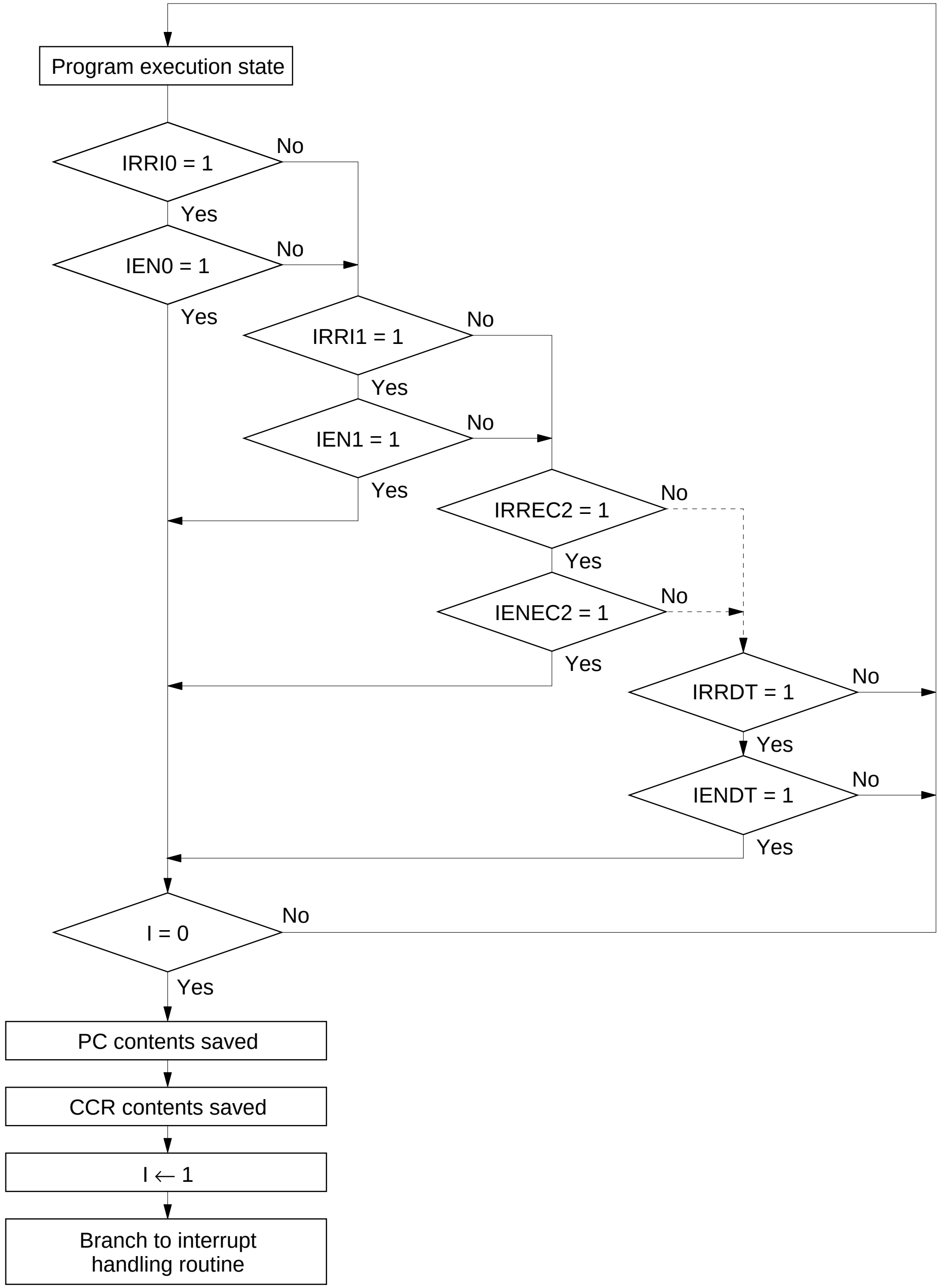
Interrupt operation is described as follows.

- 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.
- When the interrupt controller receives an interrupt request, it sets the interrupt request flag.
- 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.)
- 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.

- 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.
- The I bit of CCR is set to 1, masking further interrupts.
- 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.

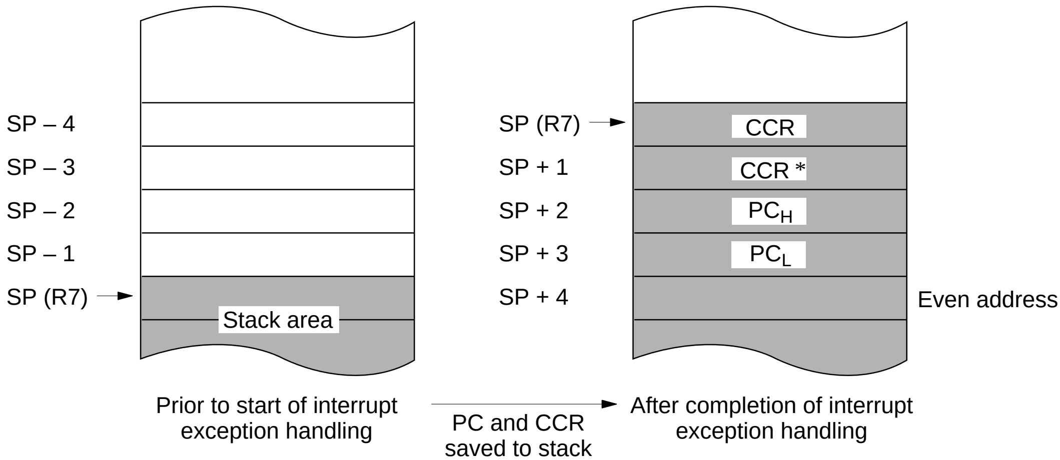
Notes:

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.



Notation:
PC: Program counter
CCR: Condition code register
I: I bit of CCR

Figure 3.3 Flow up to Interrupt Acceptance



Notation:

PC_H: Upper 8 bits of program counter (PC)
PC_L: Lower 8 bits of program counter (PC)
CCR: Condition code register
SP: Stack pointer

Notes: 1. PC shows the address of the first instruction to be executed upon return from the interrupt handling routine.
2. Register contents must always be saved and restored by word access, starting from an even-numbered address.
* Ignored on return.

Figure 3.4 Stack State after Completion of Interrupt Exception Handling

Figure 3.5 shows a typical interrupt sequence.

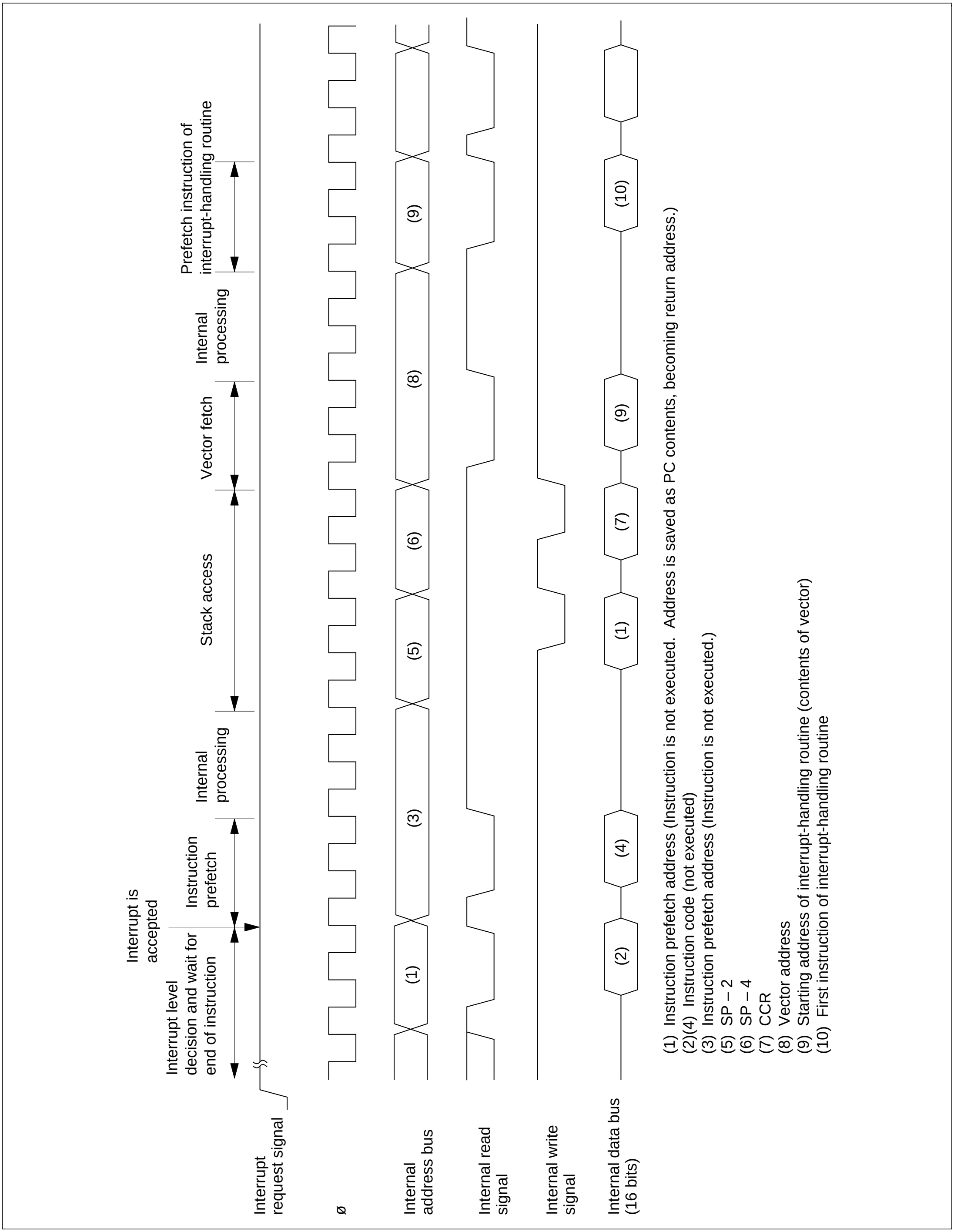


Figure 3.5 Interrupt Sequence

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 H8/3802 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.

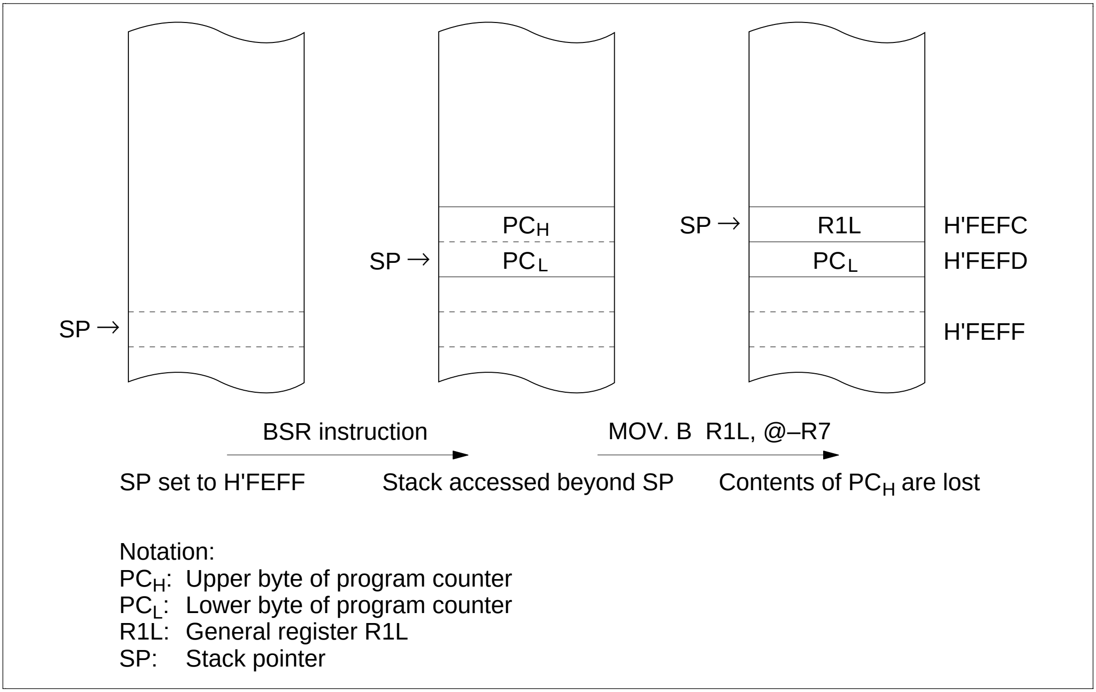


Figure 3.6 Operation when Odd Address is Set in SP

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 pins IRQAEC, $\overline{\text{IRQ}}_1$, $\overline{\text{IRQ}}_0$, $\overline{\text{WKP}}_7$ to $\overline{\text{WKP}}_0$, 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	IRREC2	When the edge designated by AIEGS1 and AIEGS0 in AEGSR is input while IENEC2 in IENRI is set to 1.
	IRRI1	When PMRB bit IRQ1 is changed from 0 to 1 while pin $\overline{\text{IRQ}}_1$ is low and IEGR bit IEG1 = 0. When PMRB bit IRQ1 is changed from 1 to 0 while pin $\overline{\text{IRQ}}_1$ is low and IEGR bit IEG1 = 1.
	IRRI0	When PMR2 bit IRQ0 is changed from 0 to 1 while pin $\overline{\text{IRQ}}_0$ is low and IEGR bit IEG0 = 0. When PMR2 bit IRQ0 is changed from 1 to 0 while pin $\overline{\text{IRQ}}_0$ is low and IEGR bit IEG0 = 1.
IWPR	IWPF7	When PMR5 bit WKP7 is changed from 0 to 1 while pin $\overline{\text{WKP}}_7$ is low.
	IWPF6	When PMR5 bit WKP6 is changed from 0 to 1 while pin $\overline{\text{WKP}}_6$ is low.
	IWPF5	When PMR5 bit WKP5 is changed from 0 to 1 while pin $\overline{\text{WKP}}_5$ is low.
	IWPF4	When PMR5 bit WKP4 is changed from 0 to 1 while pin $\overline{\text{WKP}}_4$ is low.
	IWPF3	When PMR5 bit WKP3 is changed from 0 to 1 while pin $\overline{\text{WKP}}_3$ is low.
	IWPF2	When PMR5 bit WKP2 is changed from 0 to 1 while pin $\overline{\text{WKP}}_2$ is low.
	IWPF1	When PMR5 bit WKP1 is changed from 0 to 1 while pin $\overline{\text{WKP}}_1$ is low.
	IWPF0	When PMR5 bit WKP0 is changed from 0 to 1 while pin $\overline{\text{WKP}}_0$ 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.

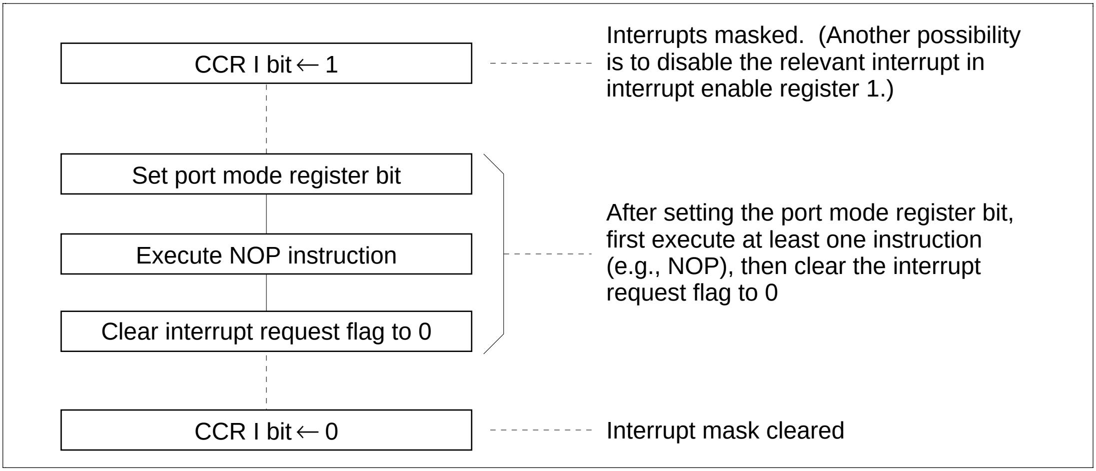


Figure 3.7 Port Mode Register Setting and Interrupt Request Flag Clearing Procedure

3.4.3 Interrupt Request Flag Clearing Method

Use the following recommended method for flag clearing in the interrupt request registers (IRR1, IRR2, and IWPR).

Recommended Method: Perform flag clearing with only one instruction. Either a bit manipulation instruction or a data transfer instruction in bytes can be used. Two examples of coding for clearing IRRI1 (bit 1 in IRR1) are shown below:

- BCLR #1,@IRR1:8
- MOV.B R1L,@IRR1:8 (Set B'11111101 into R1L in advance)

Malfunction Example: When flag clearing is performed with several instructions, a flag, other than the intended one, which was set while executing one of those instructions may be accidentally cleared, and thus cause incorrect operations to occur.

An example of coding for clearing IRRI1 (bit 1 in IRR1), in which IRRI0 is also cleared and the interrupt becomes invalid is shown below.

```
MOV.B @IRR1:8,R1L          At this point, IRRI0 is 0.
AND.B #B'11111101,R1L      IRRI0 becomes 1 here.
MOV.B R1L,@IRR1:8          IRRI0 is cleared to 0.
```

In the above example, an IRQ0 interrupt occurs while the AND.B instruction is executed. Since not only the original target IRRI1, but also IRRI0 is cleared to 0, the IRQ0 interrupt becomes invalid.

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.

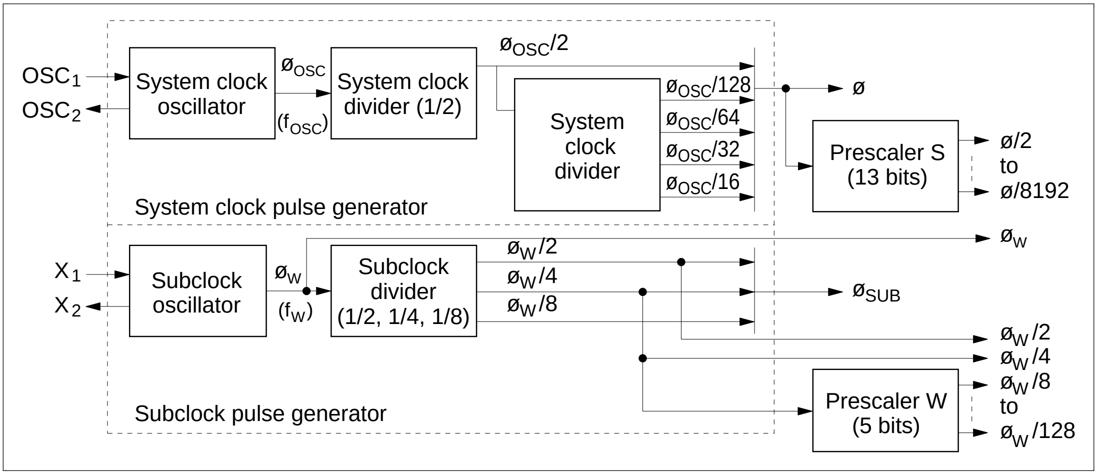


Figure 4.1 Block Diagram of Clock Pulse Generators

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.

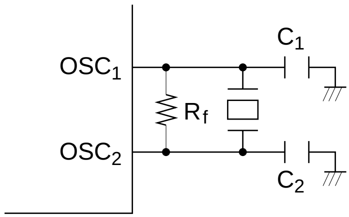
The clock signals available for use by peripheral modules are ø/2, ø/4, ø/8, ø/16, ø/32, ø/64, ø/128, ø/256, ø/512, ø/1024, ø/2048, ø/4096, ø/8192, $ø_W$, $ø_W$/2, $ø_W$/4, $ø_W$/8, $ø_W$/16, $ø_W$/32, $ø_W$/64, and $ø_W$/128. The clock requirements differ from one module to another.

4.2 System Clock Generator

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

1. Connecting a crystal oscillator

Figure 4.2 shows a typical method of connecting a crystal oscillator.



R_f = 1 MΩ ±20%

Frequency	Crystal oscillator	C_1, C_2 Recommendation value
4.19 MHz	NDK	12 pF ±20%

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.

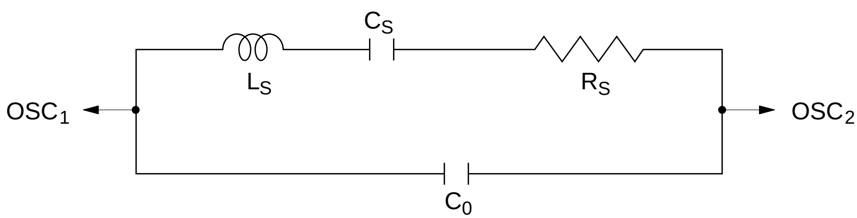


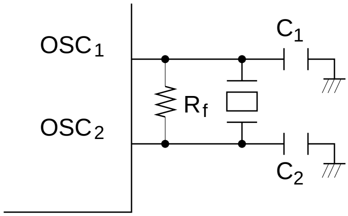
Figure 4.3 Equivalent Circuit of Crystal Oscillator

Table 4.1 Crystal Oscillator Parameters

Frequency (MHz)	4.193
RS max (Ω)	100
C_0 max (pF)	16

2. Connecting a ceramic oscillator

Figure 4.4 shows a typical method of connecting a ceramic oscillator.



R_f = 1 MΩ ±20%

Frequency	Ceramic oscillator	C_1, C_2 Recommendation value
4.0 MHz	Murata	30 pF ±10%

Figure 4.4 Typical Connection to Ceramic Oscillator

3. 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.

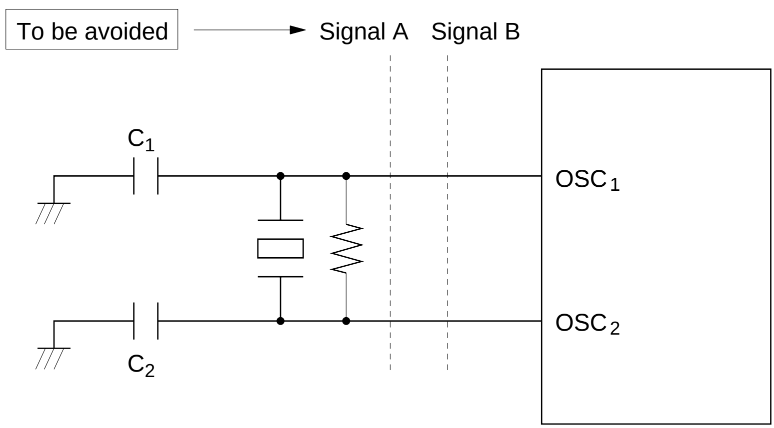


Figure 4.5 Board Design of Oscillator Circuit

4. 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.

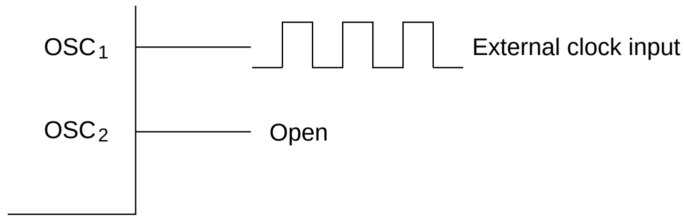


Figure 4.6 External Clock Input (Example)

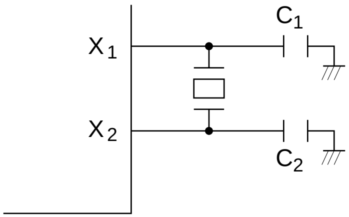
Frequency	Oscillator Clock ($ø_{OSC}$)
Duty cycle	45% to 55%

Note: The circuit parameters above are recommended by the crystal or ceramic oscillator manufacturer.
The circuit parameters are affected by the crystal or ceramic oscillator and floating capacitance when designing the board. When using the oscillator, consult with the crystal or ceramic oscillator manufacturer to determine the circuit parameters.

4.3 Subclock Generator

1. Connecting a 32.768 kHz/38.4 kHz crystal oscillator

Clock pulses can be supplied to the subclock divider by connecting a 32.768 kHz/38.4 kHz crystal oscillator, as shown in figure 4.7. Follow the same precautions as noted under 3. notes on board design for the system clock in 4.2.

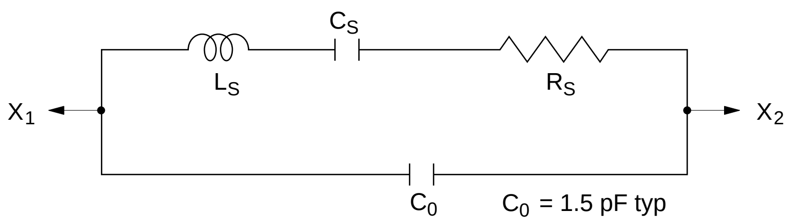


$C_1 = C_2 = 15$ pF (typ.)

Frequency	Crystal oscillator	Products Name
32.768 kHz	Nihon Denpa Kogyo	MX73P
38.4 kHz	Seiko Instrument Inc.	VTC-200

Figure 4.7 Typical Connection to 32.768 kHz/38.4 kHz Crystal Oscillator (Subclock)

Figure 4.8 shows the equivalent circuit of the 32.768 kHz/38.4 kHz crystal oscillator.



C_0 = 1.5 pF typ
R_S = 14 kΩ typ
f_W = 32.768 kHz/38.4kHz

Figure 4.8 Equivalent Circuit of 32.768 kHz/38.4 kHz Crystal Oscillator

2. Pin connection when not using subclock

When the subclock is not used, connect pin X_1 to GND and leave pin X_2 open, as shown in figure 4.9.

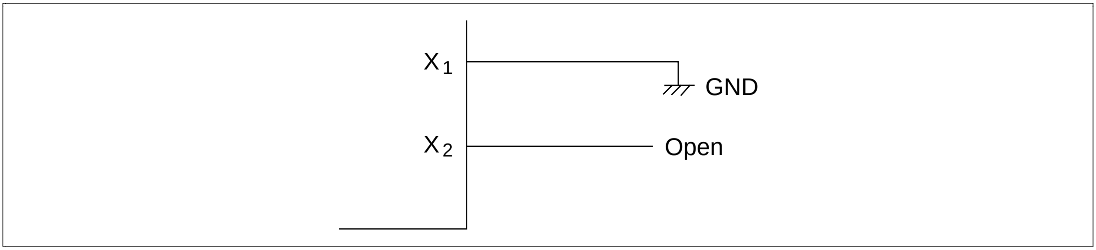


Figure 4.9 Pin Connection when not Using Subclock

3. External clock input

Connect the external clock to the X1 pin and leave the X2 pin open, as shown in figure 4.10.

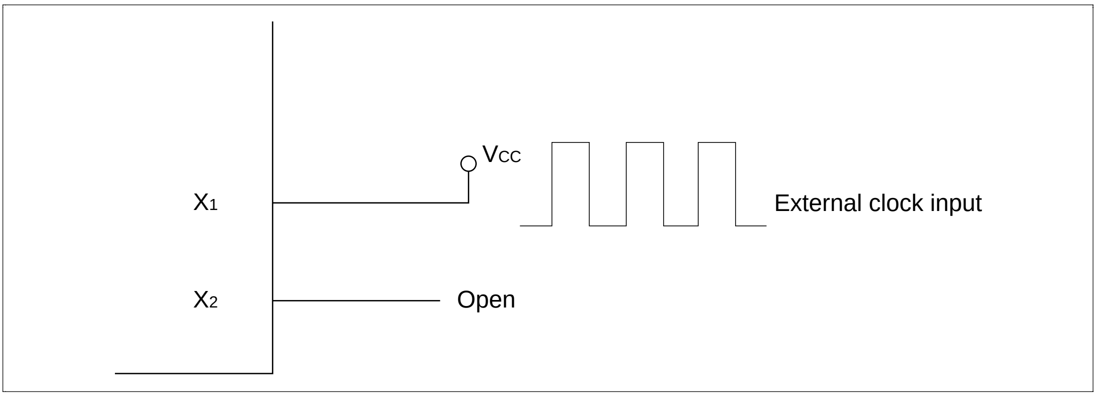


Figure 4.10 Pin Connection when Inputting External Clock

Frequency	Subclock (øw)
Duty	45% to 55%

4.4 Prescalers

The H8/3802 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 or 38.4 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.

1. 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 timer A, timer F, SCI3, the A/D converter, the LCD controller, and the 10-bit PWM. 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 øosc/16, øosc/32, øosc/64, or øosc/128.

2. Prescaler W (PSW)

Prescaler W is a 5-bit counter using a 32.768 kHz/38.4 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 X1 and X2.

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 are closely related to board design and should be carefully evaluated by the user in mask ROM and ZTAT™ versions, referring to the examples shown in this section. 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.

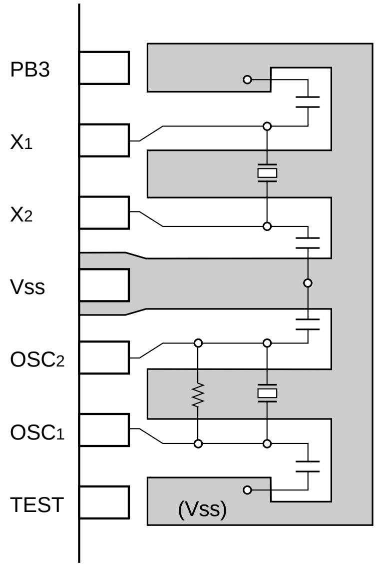


Figure 4.11 Example of Crystal and Ceramic Oscillator Element Arrangement

4.5.1 Definition of Oscillation Settling Standby Time

Figure 4.12 shows the oscillation waveform (OSC2), system clock (ø), and microcomputer operating mode when a transition is made from standby mode, watch mode, or subactive mode, to active (high-speed/medium-speed) mode, with an oscillator element connected to the system clock oscillator.

As shown in figure 4.12, as the system clock oscillator is halted in standby mode, watch mode, and subactive mode, when a transition is made to active (high-speed/medium-speed) mode, the sum of the following two times (oscillation settling time and standby time) is required.

1. Oscillation settling time (t_{rc})

The time from the point at which the system clock oscillator oscillation waveform starts to change when an interrupt is generated, until the amplitude of the oscillation waveform increases and the oscillation frequency stabilizes.

2. Standby time

The time required for the CPU and peripheral functions to begin operating after the oscillation waveform frequency and system clock have stabilized.

The standby time setting is selected with standby timer select bits 2 to 0 (STS2 to STS0) (bits 6 to 4 in system control register 1 (SYSCR1)).

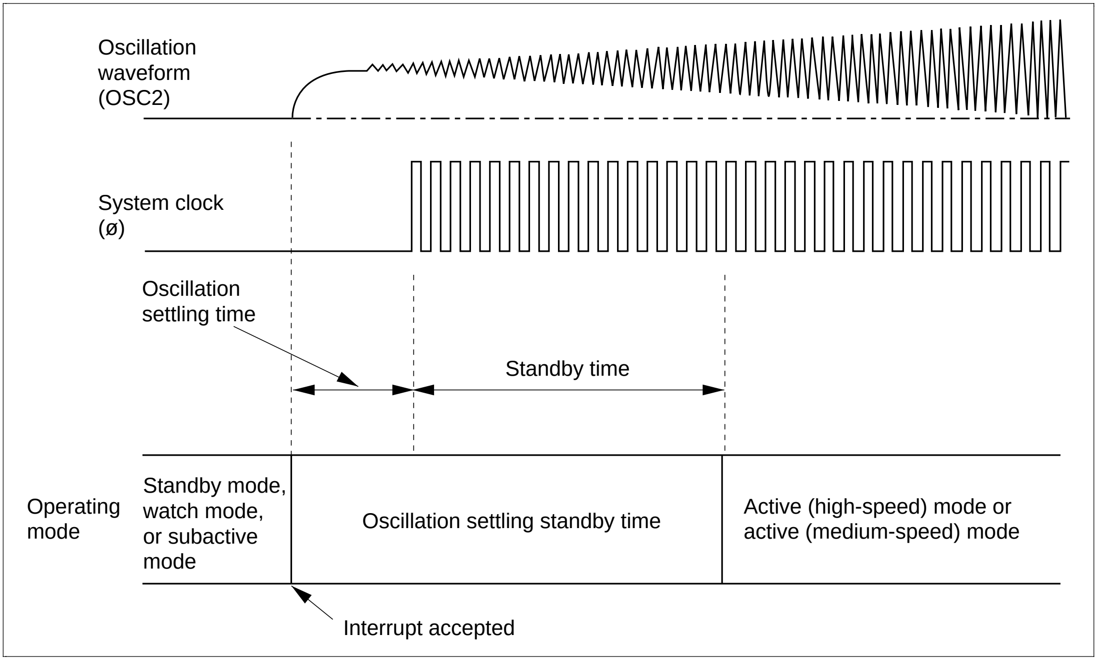


Figure 4.12 Oscillation Settling Standby Time

When standby mode, watch mode, or subactive mode is cleared by an interrupt or reset, and a transition is made to active (high-speed/medium-speed) mode, the oscillation waveform begins to change at the point at which the interrupt is accepted. Therefore, when an oscillator element is connected in standby mode, watch mode, or subactive mode, since the system clock oscillator is halted, the time from the point at which this oscillation waveform starts to change until the amplitude of the oscillation waveform increases and the oscillation frequency stabilizes—that is, the oscillation settling time—is required.

The oscillation settling time in the case of these state transitions is the same as the oscillation settling time at power-on (the time from the point at which the power supply voltage reaches the prescribed level until the oscillation stabilizes), specified by "oscillation settling time t_{rc}" in the AC characteristics.

Meanwhile, once the system clock has halted, a standby time of at least 8 states is necessary in order for the CPU and peripheral functions to operate normally.

Thus, the time required from interrupt generation until operation of the CPU and peripheral functions is the sum of the above described oscillation settling time and standby time. This total time is called the oscillation settling standby time, and is expressed by equation (1) below.

Oscillation settling standby time = oscillation settling time + standby time

$$= t_{rc} + (8 \text{ to } 16{,}384 \text{ states}) \quad .... (1)$$

Therefore, when a transition is made from standby mode, watch mode, or subactive mode, to active (high-speed/medium-speed) mode, with an oscillator element connected to the system clock oscillator, careful evaluation must be carried out on the installation circuit before deciding on the oscillation settling standby time. In particular, since the oscillation settling time is affected by installation circuit constants, stray capacitance, and so forth, suitable constants should be determined in consultation with the oscillator element manufacturer.

4.5.2 Notes on Use of Crystal Oscillator Element (Excluding Ceramic Oscillator Element)

When a microcomputer operates, the internal power supply potential fluctuates slightly in synchronization with the system clock. Depending on the individual crystal oscillator element characteristics, the oscillation waveform amplitude may not be sufficiently large immediately after the oscillation settling standby time, making the oscillation waveform susceptible to influence by fluctuations in the power supply potential. In this state, the oscillation waveform may be disrupted, leading to an unstable system clock and erroneous operation of the microcomputer.

If erroneous operation occurs, change the setting of standby timer select bits 2 to 0 (STS2 to STS0) (bits 6 to 4 in system control register 1 (SYSCR1)) to give a longer standby time.

For example, if erroneous operation occurs with a standby time setting of 16 states, check the operation with a standby time setting of 1,024 states or more.

If the same kind of erroneous operation occurs after a reset as after a state transition, hold the $\overline{\text{RES}}$ pin low for a longer period.

Section 5 Power-Down Modes

5.1 Overview

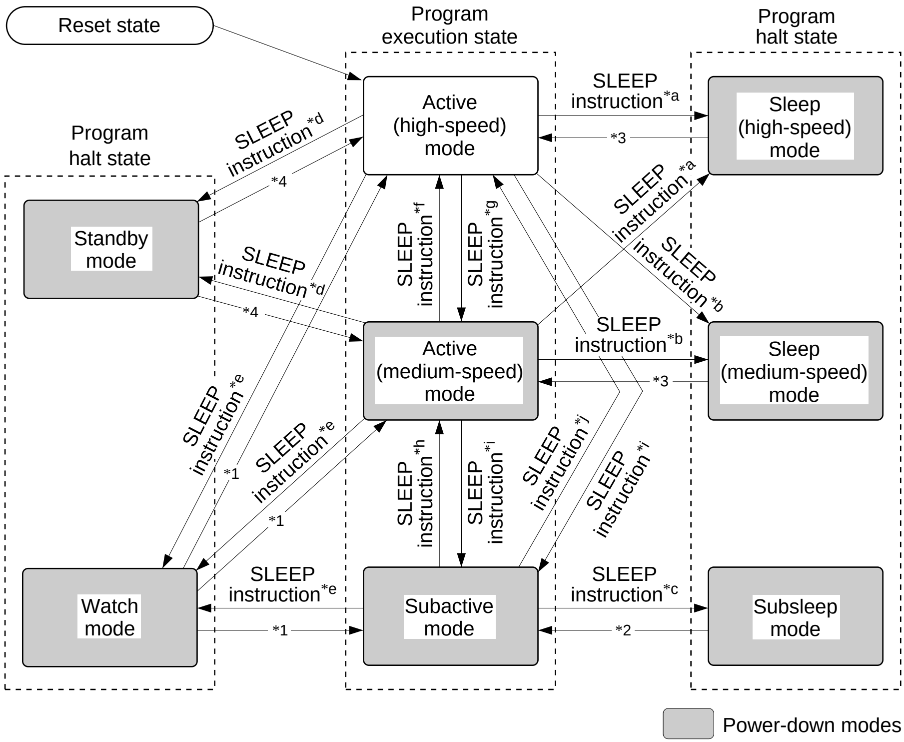
The H8/3802 Series has nine modes of operation after a reset. These include eight power-down modes, in which power dissipation is significantly reduced. Table 5.1 gives a summary of the eight operating modes.

Table 5.1 Operating Modes

Operating Mode	Description
Active (high-speed) mode	The CPU and all on-chip peripheral functions are operable on the system clock in high-speed operation
Active (medium-speed) mode	The CPU and all on-chip peripheral functions are operable on the system clock in low-speed operation
Subactive mode	The CPU is operable on the subclock in low-speed operation
Sleep (high-speed) mode	The CPU halts. On-chip peripheral functions are operable on the system clock
Sleep (medium-speed) mode	The CPU halts. On-chip peripheral functions operate at a frequency of 1/64, 1/32, 1/16, or 1/8 of the system clock frequency
Subsleep mode	The CPU halts. The time-base function of timer A, timer F, SCI3, AEC and LCD controller/driver are operable on the subclock
Watch mode	The CPU halts. The time-base function of timer A, timer F, AEC and LCD controller/driver are operable on the subclock
Standby mode	The CPU and all on-chip peripheral functions halt
Module standby mode	Individual on-chip peripheral functions specified by software enter standby mode and halt

Of these nine operating modes, all but the active (high-speed) mode are power-down modes. In this section the two active modes (high-speed and medium speed) will be 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.



Mode Transition Conditions (1)

	LSON	MSON	SSBY	TMA3	DTON
a	0	0	0	*	0
b	0	1	0	*	0
c	1	*	0	1	0
d	0	*	1	0	0
e	*	*	1	1	0
f	0	0	0	*	1
g	0	1	0	*	1
h	0	1	1	1	1
i	1	*	1	1	1
j	0	0	1	1	1

* : Don't care

Mode Transition Conditions (2)

	Interrupt Sources
1	Timer A, Timer F interrupt, IRQ_0 interrupt, WKP_7 to WKP_0 interrupts
2	Timer A, Timer F, SCI3 interrupt, IRQ_1 and IRQ_0 interrupts, IRQAEC, WKP_7 to WKP_0 interrupts, AEC
3	All interrupts
4	IRQ_1 or IRQ_0 interrupt, WKP_7 to WKP_0 interrupts

Notes: 1. A transition between different modes cannot be made to occur simply because an interrupt request is generated. Make sure that interrupt handling is performed after the interrupt is accepted.

2. Details on the mode transition conditions are given in the explanations of each mode, in sections 5-2 through 5-8.

Figure 5.1 Mode Transition Diagram

Table 5.2 Internal State in Each Operating Mode

Function		Active Mode		Sleep Mode					
		High-Speed	Medium-Speed	High-Speed	Medium-Speed	Watch Mode	Subactive Mode	Subsleep Mode	Standby Mode
System clock oscillator		Functions	Functions	Functions	Functions	Halted	Halted	Halted	Halted
Subclock oscillator		Functions	Functions	Functions	Functions	Functions	Functions	Functions	Functions
CPU operations	Instructions	Functions	Functions	Halted	Halted	Halted	Functions	Halted	Halted
	RAM			Retained	Retained	Retained		Retained	Retained
	Registers								
	I/O ports								Retained[*1]
External interrupts	IRQ_0	Functions	Functions	Functions	Functions	Functions	Functions	Functions	Functions
	IRQ_1					Retained[*5]			
	IRQAEC								Retained[*5]
	WKP_0	Functions	Functions	Functions	Functions	Functions	Functions	Functions	Functions
	WKP_1								
	WKP_2								
	WKP_3								
	WKP_4								
	WKP_5								
	WKP_6								
	WKP_7								
Peripheral functions	Timer A	Functions	Functions	Functions	Functions	Functions[*4]	Functions[*4]	Functions[*4]	Retained
	Asynchronous counter					Functions[*6]	Functions	Functions	Functions[*6]
	Timer F					Functions/ Retained[*7]	Functions/ Retained[*7]	Functions/ Retained[*7]	Retained
	SCI3					Reset	Functions/ Retained[*2]	Functions/ Retained[*2]	Reset
	PWM					Retained	Retained	Retained	Retained
	A/D converter					Retained	Retained	Retained	Retained
	LCD					Functions/ Retained[*3]	Functions/ Retained[*3]	Functions/ Retained[*3]	Retained

Notes:
1. Register contents are retained, but output is high-impedance state.
2. Functions if $ø_W/2$ is selected as the internal clock; otherwise halted and retained.
3. Functions if $ø_W$, $ø_W/2$ or $ø_W/4$ is selected as the operating clock; otherwise halted and retained.
4. Functions if the timekeeping time-base function is selected.
5. External interrupt requests are ignored. Interrupt request register contents are not altered.
6. Incrementing is possible, but interrupt generation is not.
7. Functions if the $ø_W/4$ internal clock is selected; otherwise halted and retained.

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 Registers

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'F0	H'FFF1

1. System control register 1 (SYSCR1)

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	LSON	—	MA1	MA0
Initial value	0	0	0	0	0	1	1	1
Read/Write	R/W	R/W	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 (initial value) • 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

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 operating frequency so that the waiting time is at least equal to the oscillation settling time.

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description	
0	0	0	Wait time = 8,192 states	(initial value)
0	0	1	Wait time = 16,384 states	
0	1	0	Wait time = 1,024 states	
0	1	1	Wait time = 2,048 states	
1	0	0	Wait time = 4,096 states	
1	0	1	Wait time = 2 states	(External clock mode)
1	1	0	Wait time = 8 states	
1	1	1	Wait time = 16 states	

Note: In the case that external clock is input, set up the "Standby timer select" selection to External clock mode before Mode Transition. Also, do not set up to external clock mode, in the case that it does not use external clock.

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 (øSUB)	

Bits 2: Reserved bit

Bit 2 is reserved: it is always read as 1 and cannot be modified.

Bits 1 and 0: Active (medium-speed) mode clock select (MA1, MA0)

Bits 1 and 0 choose $ø_{osc}/128$, $ø_{osc}/64$, $ø_{osc}/32$, or $ø_{osc}/16$ as the operating clock in active (medium-speed) mode and sleep (medium-speed) mode. MA1 and MA0 should be written in active (high-speed) mode or subactive mode.

Bit 1 MA1	Bit 0 MA0	Description	
0	0	$ø_{osc}/16$	
0	1	$ø_{osc}/32$	
1	0	$ø_{osc}/64$	
1	1	$ø_{osc}/128$	(initial value)

2. System control register 2 (SYSCR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	NESEL	DTON	MSON	SA1	SA0
Initial value	1	1	1	1	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.

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 16 MHz, clear NESEL to 0.

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

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 in active (high-speed) mode (initial value)
1	Operation 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)
0	1	$ø_W/4$	
1	*	$ø_W/2$	

* : Don't care

5.2 Sleep Mode

5.2.1 Transition to Sleep Mode

1. Transition to sleep (high-speed) mode

The system goes from active mode to sleep (high-speed) mode when a SLEEP instruction is executed while the SSBY and LSON bits in SYSCR1 are cleared to 0, the MSON and DTON bits in SYSCR2 are cleared to 0. In sleep mode CPU operation is halted but the on-chip peripheral functions. CPU register contents are retained.

2. Transition to sleep (medium-speed) mode

The system goes from active mode to sleep (medium-speed) mode when a SLEEP instruction is executed 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 cleared to 0. In sleep (medium-speed) mode, as in sleep (high-speed) mode, CPU operation is halted but the on-chip peripheral functions are operational. The clock frequency in sleep (medium-speed) mode is determined by the MA1 and MA0 bits in SYSCR1. CPU register contents are retained.
Furthermore, it sometimes acts with half state early timing at the time of transition to sleep (medium-speed) mode.

5.2.2 Clearing Sleep Mode

Sleep mode is cleared by any interrupt (timer A, timer F, asynchronous counter, IRQAEC, IRQ_1, IRQ_0, WKP_7 to WKP_0, SCI3, A/D converter), or by input at the $\overline{RES}$ pin.

- Clearing by interrupt

When an interrupt is requested, sleep mode is cleared and interrupt exception handling starts. A transition is made from sleep (high-speed) mode to active (high-speed) mode, or from sleep (medium-speed) mode to active (medium-speed) mode. 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.
Interrupt signal and system clock are mutually asynchronous. Synchronization error time in a maximum is 2/ø (s).

- Clearing by $\overline{RES}$ input

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

5.2.3 Clock Frequency in Sleep (Medium-Speed) Mode

Operation in sleep (medium-speed) mode is clocked at the frequency designated by the MA1 and MA0 bits in SYSCR1.

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 in SYSCR1 is cleared to 0, and bit TMA3 in TMA is cleared to 0. In standby mode the clock pulse generator stops, so the CPU and on-chip peripheral modules stop functioning, but as long as the rated voltage is supplied, the contents of CPU registers, on-chip RAM, and some on-chip peripheral module registers are retained. On-chip RAM contents will be further retained down to a minimum RAM data retention voltage. The I/O ports go to the high-impedance state.

5.3.2 Clearing Standby Mode

Standby mode is cleared by an interrupt (IRQ_1 or IRQ_0), WKP_7 to WKP_0 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

The table below gives settings for various operating frequencies. Set bits STS2 to STS0 for a waiting time at least as long as the oscillation settling time.

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

STS2	STS1	STS0	Waiting Time	5 MHz	2 MHz
0	0	0	8,192 states	1.638	4.1
0	0	1	16,384 states	3.277	8.2
0	1	0	1,024 states	0.205	0.512
0	1	1	2,048 states	0.410	1.024
1	0	0	4,096 states	0.819	2.048
1	0	1	2 states (Use prohibited)	0.0004	0.001
1	1	0	8 states	0.0002	0.004
1	1	1	16 states	0.003	0.008

- When an external clock is used

STS2 = 1, STS1 = 0, and STS0 = 1 should be set. Other values possible use, but CPU sometimes will start operation before waiting time completion.

5.3.4 Standby Mode Transition and Pin States

When a SLEEP instruction is executed in active (high-speed) mode or active (medium-speed) mode while bit SSBY is set to 1 and bit LSON is cleared to 0 in SYSCR1, and bit TMA3 is cleared to 0 in TMA, a transition is made to standby mode. At the same time, pins go to the high-impedance state (except pins for which the pull-up MOS is designated as on). Figure 5.2 shows the timing in this case.

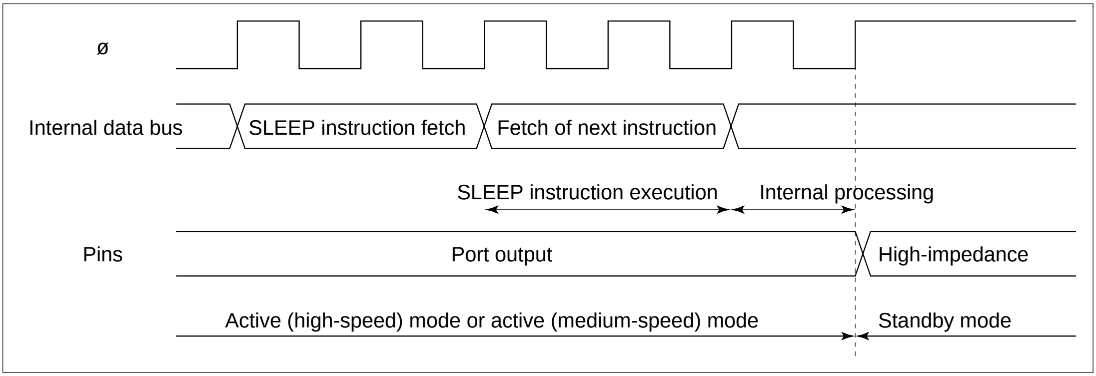


Figure 5.2 Standby Mode Transition and Pin States

5.3.5 Notes on External Input Signal Changes before/after Standby Mode

1. When external input signal changes before/after standby mode or watch mode

   When an external input signal such as $\overline{IRQ}$, $\overline{WKP}$, or IRQAEC is input, both the high- and low-level widths of the signal must be at least two cycles of system clock ø or subclock $ø_{SUB}$ (referred to together in this section as the internal clock). As the internal clock stops in standby mode and watch mode, the width of external input signals requires careful attention when a transition is made via these operating modes. Ensure that external input signals conform to the conditions stated in 3, Recommended timing of external input signals, below

2. When external input signals cannot be captured because internal clock stops

   The case of falling edge capture is illustrated in figure 5.3

   As shown in the case marked "Capture not possible," when an external input signal falls immediately after a transition to active (high-speed or medium-speed) mode or subactive mode, after oscillation is started by an interrupt via a different signal, the external input signal cannot be captured if the high-level width at that point is less than 2 t_{cyc} or 2 t_{subcyc}.

3. Recommended timing of external input signals

   To ensure dependable capture of an external input signal, high- and low-level signal widths of at least 2 t_{cyc} or 2 t_{subcyc} are necessary before a transition is made to standby mode or watch mode, as shown in "Capture possible: case 1."

   External input signal capture is also possible with the timing shown in "Capture possible: case 2" and "Capture possible: case 3," in which a 2 t_{cyc} or 2 t_{subcyc} level width is secured.

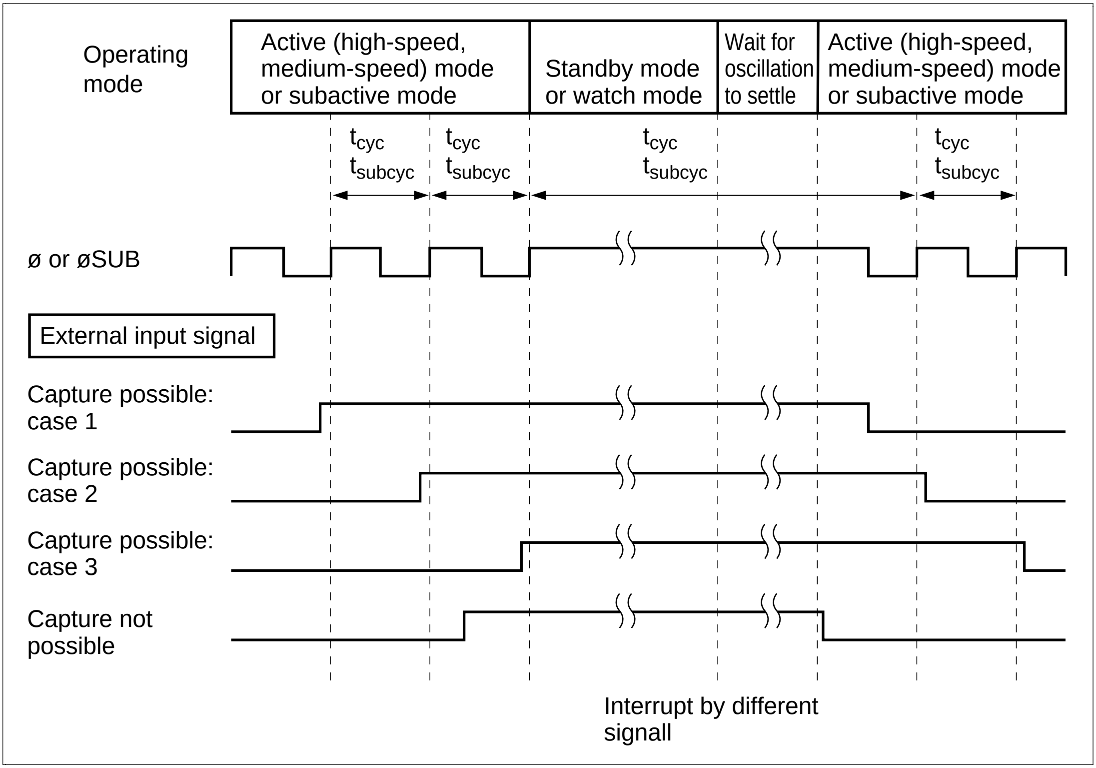


Figure 5.3 External Input Signal Capture when Signal Changes before/after Standby Mode or Watch Mode

4. Input pins to which these notes apply:

$\overline{IRQ_1}$ to $\overline{IRQ_0}$, $\overline{WKP_7}$ to $\overline{WKP_0}$, IRQAEC

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 is halted except for timer A, timer F, AEC and the LCD controller/driver (for which operation or halting can be set) is halted. As long as a minimum required voltage is applied, the contents of CPU registers, the on-chip RAM and some registers of the on-chip peripheral modules, are retained. I/O ports keep the same states as before the transition.

5.4.2 Clearing Watch Mode

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

• Clearing by interrupt

When watch mode is cleared by interrupt, 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 2. Clearing by $\overline{RES}$ pin in 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.

5.4.4 Notes on External Input Signal Changes before/after Watch Mode

See 5.3.5, Notes on External Input Signal Changes before/after Standby Mode.

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 bit in SYSCR1 is set to 1, and TMA3 bit in TMA is set to 1. In subsleep mode, operation of on-chip peripheral modules other than the A/D converter, and PWM is halted. As long as a minimum required voltage is applied, the contents of CPU registers, the on-chip RAM and some registers of the on-chip peripheral modules are retained. I/O ports keep the same states as before the transition.

5.5.2 Clearing Subsleep Mode

Subsleep mode is cleared by an interrupt (timer A, timer F, asynchronous counter, SCI3, IRQAEC, IRQ_1, IRQ_0, WKP_7 to WKP_0) 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.
Interrupt signal and system clock are mutually asynchronous. Synchronization error time in a maximum is $2/ø_{SUB}$ (s).

- Clearing by $\overline{RES}$ input

Clearing by $\overline{RES}$ pin is the same as for standby mode; see Clearing by $\overline{RES}$ pin in 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, timer F, IRQ_0, or WKP_7 to WKP_0 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 F, asynchronous event counter, SCI3, IRQAEC, IRQ_1, IRQ_0, or WKP_7 to WKP_0 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.

5.6.2 Clearing Subactive Mode

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

- Clearing by SLEEP instruction

If a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1 and TMA3 bit 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 $\overline{\text{RES}}$ pin

Clearing by $\overline{\text{RES}}$ pin is the same as for standby mode; see Clearing by $\overline{\text{RES}}$ pin in 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 $\overline{RES}$ pin is driven low, active (medium-speed) mode is entered. If the LSON 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_7 to WKP_0 interrupts in standby mode, timer A, timer F, IRQ_0, or WKP_7 to WKP_0 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.
Furthermore, it sometimes acts with half state early timing at the time of transition to active (medium-speed) mode.

5.7.2 Clearing Active (Medium-Speed) Mode

Active (medium-speed) mode is cleared by a SLEEP instruction.

- Clearing by SLEEP instruction

A transition to standby mode takes place if the SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1, the LSON bit in SYSCR1 is cleared to 0, and the TMA3 bit 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.

When both SSBY and LSON are cleared to 0 in SYSCR1 and a SLEEP instruction is executed, sleep mode is entered. Direct transfer to active (high-speed) mode or to subactive mode is also possible. See 5.8, Direct Transfer, below for details.

- Clearing by $\overline{RES}$ pin

When the $\overline{RES}$ pin is driven low, a transition is made to the reset state and active (medium-speed) mode is cleared.

5.7.3 Operating Frequency in Active (Medium-Speed) Mode

Operation in active (medium-speed) mode is clocked at the frequency designated by the MA1 and MA0 bits in SYSCR1.

5.8 Direct Transfer

5.8.1 Overview of Direct Transfer

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, 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 (medium-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 the TMA3 bit 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 the TMA3 bit 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) while the SSBY and LSON bits in SYSCR1 are set to 1, the DTON bit in SYSCR2 is set to 1, and the TMA3 bit 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 the TMA3 bit 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 Transition Times

1. Time for direct transition from active (high-speed) mode to active (medium-speed) mode

A direct transition from active (high-speed) mode to active (medium-speed) mode is performed by executing a SLEEP instruction in active (high-speed) mode while bits SSBY and LSON are both cleared to 0 in SYSCR1, and bits MSON and DTON are both set to 1 in SYSCR2. The time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transition time) is given by equation (1) below.

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

Example: Direct transition time = (2 + 1) × 2tosc + 14 × 16tosc = 230tosc (when ø/8 is selected as the CPU operating clock)

Notation:
tosc: OSC clock cycle time
tcyc: System clock (ø) cycle time

2. Time for direct transition from active (medium-speed) mode to active (high-speed) mode

A direct transition from active (medium-speed) mode to active (high-speed) mode is performed by executing a SLEEP instruction in active (medium-speed) mode while bits SSBY and LSON are both cleared to 0 in SYSCR1, and bit MSON is cleared to 0 and bit DTON is set to 1 in SYSCR2. The time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transition time) is given by equation (2) below.

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

Example: Direct transition time = (2 + 1) × 16tosc + 14 × 2tosc = 76tosc (when ø/8 is selected as the CPU operating clock)

Notation:
tosc: OSC clock cycle time
tcyc: System clock (ø) cycle time

3. Time for direct transition from subactive mode to active (high-speed) mode

A direct transition from subactive mode to active (high-speed) mode is performed by executing a SLEEP instruction in subactive mode while bit SSBY is set to 1 and bit LSON is cleared to 0 in SYSCR1, bit MSON is cleared to 0 and bit DTON is set to 1 in SYSCR2, and bit TMA3 is set to 1 in TMA. The time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transition time) is given by equation (3) below.

Direct transition time = { (Number of SLEEP instruction execution states) + (number of internal processing states) } × (tsubcyc before transition) + { (wait time set in STS2 to STS0) + (number of interrupt exception handling execution states) } × (tcyc after transition) .... (3)

Example: Direct transition time = (2 + 1) × 8tw + (8192 + 14) × 2tosc = 24tw + 16412tosc (when øw/8 is selected as the CPU operating clock, and wait time = 8192 states)

Notation:
tosc: OSC clock cycle time
tw: Watch clock cycle time
tcyc: System clock (ø) cycle time
tsubcyc: Subclock (øSUB) cycle time

4. Time for direct transition from subactive mode to active (medium-speed) mode

A direct transition from subactive mode to active (medium-speed) mode is performed by executing a SLEEP instruction in subactive mode while bit SSBY is set to 1 and bit LSON is cleared to 0 in SYSCR1, bits MSON and DTON are both set to 1 in SYSCR2, and bit TMA3 is set to 1 in TMA. The time from execution of the SLEEP instruction to the end of interrupt exception handling (the direct transition time) is given by equation (4) below.

Direct transition time = { (Number of SLEEP instruction execution states) + (number of internal processing states) } × (tsubcyc before transition) + { (wait time set in STS2 to STS0) + (number of interrupt exception handling execution states) } × (tcyc after transition) .... (4)

Example: Direct transition time = (2 + 1) × 8tw + (8192 + 14) × 16tosc = 24tw + 131296tosc
(when øw/8 or ø8 is selected as the CPU operating clock, and wait time = 8192 states)

Notation:

tosc: OSC clock cycle time
tw: Watch clock cycle time
tcyc: System clock (ø) cycle time
tsubcyc: Subclock (øSUB) cycle time

5.8.3 Notes on External Input Signal Changes before/after Direct Transition

1. Direct transition from active (high-speed) mode to subactive mode
   Since the mode transition is performed via watch mode, see 5.3.5, Notes on External Input Signal Changes before/after Standby Mode.

2. Direct transition from active (medium-speed) mode to subactive mode
   Since the mode transition is performed via watch mode, see 5.3.5, Notes on External Input Signal Changes before/after Standby Mode.

3. Direct transition from subactive mode to active (high-speed) mode
   Since the mode transition is performed via watch mode, see 5.3.5, Notes on External Input Signal Changes before/after Standby Mode.

4. Direct transition from subactive mode to active (medium-speed) mode
   Since the mode transition is performed via watch mode, see 5.3.5, Notes on External Input Signal Changes before/after Standby Mode.

5.9 Module Standby Mode

5.9.1 Setting Module Standby Mode

Module standby mode is set for individual peripheral functions. All the on-chip peripheral modules can be placed in module standby mode. When a module enters module standby mode, the system clock supply to the module is stopped and operation of the module halts. This state is identical to standby mode.

Module standby mode is set for a particular module by setting the corresponding bit to 0 in clock stop register 1 (CKSTPR1) or clock stop register 2 (CKSTPR2). (See table 5.5.)

5.9.2 Clearing Module Standby Mode

Module standby mode is cleared for a particular module by setting the corresponding bit to 1 in clock stop register 1 (CKSTPR1) or clock stop register 2 (CKSTPR2). (See table 5.5.)

Following a reset, clock stop register 1 (CKSTPR1) and clock stop register 2 (CKSTPR2) are both initialized to H'FF.

Table 5.5

Register Name	Bit Name		Operation
CKSTPR1	TACKSTP	1	Timer A module standby mode is cleared
		0	Timer A is set to module standby mode
	TFCKSTP	1	Timer F module standby mode is cleared
		0	Timer F is set to module standby mode
	ADCKSTP	1	A/D converter module standby mode is cleared
		0	A/D converter is set to module standby mode
	S32CKSTP	1	SCI3 module standby mode is cleared
		0	SCI3 is set to module standby mode

Register Name	Bit Name		Operation
CKSTPR2	LDCKSTP	1	LCD module standby mode is cleared
		0	LCD is set to module standby mode
	PW1CKSTP	1	PWM1 module standby mode is cleared
		0	PWM1 is set to module standby mode
	AECKSTP	1	Asynchronous event counter module standby mode is cleared
		0	Asynchronous event counter is set to module standby mode
	PW2CKSTP	1	PWM2 module standby mode is cleared
		0	PWM2 is set to module standby mode

Note: For details of module operation, see the sections on the individual modules.

Section 6 ROM

6.1 Overview

The H8/3802 has 16 kbytes of on-chip mask ROM, the H8/3801 has 12 kbytes, and the H8/3800 has 8 kbytes. The ROM is connected to the CPU by a 16-bit data bus, allowing high-speed two-state access for both byte data and word data. The H8/3802 has a ZTAT™ version with 16-kbyte PROM.

6.1.1 Block Diagram

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

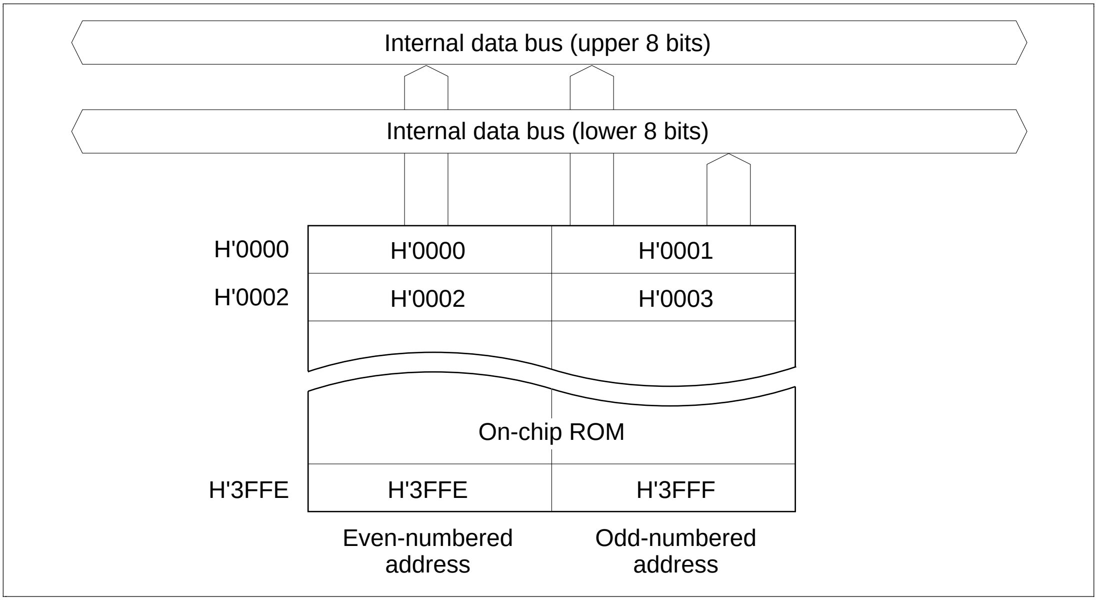


Figure 6.1 ROM Block Diagram (H8/3802)

6.2 H8/3802 PROM Mode

6.2.1 Setting to PROM Mode

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

Table 6.1 Setting to PROM Mode

Pin Name	Setting
TEST	High level
PB_0/AN_0	Low level
PB_1/AN_1	
PB_2/AN_2	High level

6.2.2 Socket Adapter Pin Arrangement and Memory Map

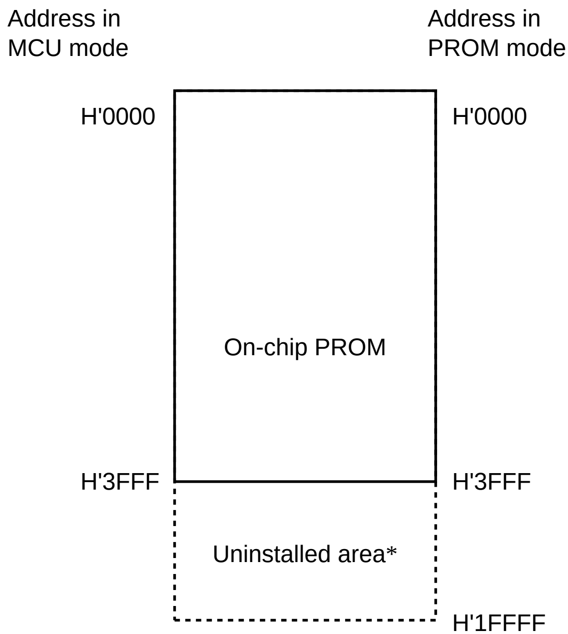
A standard PROM programmer can be used to program the PROM. A socket adapter is required for conversion to 32 pins.

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

H8/3802 FP-64A, FP-64E	H8/3802 DP-64S	H8/3802 Pin	EPROM socket Pin	EPROM socket HN27C101 (32-pin)
8	16	$\overline{RES}$	V_{PP}	1
40	48	$P6_0$	EO_0	13
39	47	$P6_1$	EO_1	14
38	46	$P6_2$	EO_2	15
37	45	$P6_3$	EO_3	17
36	44	$P6_4$	EO_4	18
35	43	$P6_5$	EO_5	19
34	42	$P6_6$	EO_6	20
33	41	$P6_7$	EO_7	21
57	1	$P4_0$	EA_0	12
58	2	$P4_1$	EA_1	11
10	18	$P3_2$	EA_2	10
11	19	$P3_3$	EA_3	9
12	20	$P3_4$	EA_4	8
13	21	$P3_5$	EA_5	7
14	22	$P3_6$	EA_6	6
15	23	$P3_7$	EA_7	5
32	40	$P7_0$	EA_8	27
60	4	$P4_3$	EA_9	26
30	38	$P7_2$	EA_{10}	23
29	37	$P7_3$	EA_{11}	25
28	36	$P7_4$	EA_{12}	4
27	35	$P7_5$	EA_{13}	28
26	34	$P7_6$	EA_{14}	29
52	60	$P9_3$	EA_{15}	3
53	61	$P9_4$	EA_{16}	2
25	33	$P7_7$	$\overline{CE}$	22
31	39	$P7_1$	$\overline{OE}$	24
51	59	$P9_2$	$\overline{PGM}$	31
16	24	V_{CC}	V_{CC}	32
61	5	AV_{CC}		
7	15	TEST		
2	10	X_1		
64	8	PB_2		
49	57	$P9_0$		
50	58	$P9_1$		
54	62	$P9_5$		
55	63	V_{SS}	V_{SS}	16
4	12	AV_{SS}		
62	6	PB_0		
63	7	PB_1		

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

Figure 6.2 Socket Adapter Pin Correspondence (with HN27C101)



Note: * The output data is not guaranteed if this address area is read in PROM mode. Therefore, when programming with a PROM programmer, be sure to specify addresses from H'0000 to H'3FFF. If programming is inadvertently performed from H'4000 onward, it may not be possible to continue PROM programming and verification.
When programming, H'FF should be set as the data in this address area (H'4000 to H'1FFFF).

Figure 6.3 H8/3802 Memory Map in PROM Mode

6.3 H8/3802 Programming

The write, verify, and other modes are selected as shown in table 6.2 in H8/3802 PROM mode.

Table 6.2 Mode Selection in PROM Mode (H8/3802)

	Pins						
Mode	$\overline{\textbf{CE}}$	$\overline{\textbf{OE}}$	$\overline{\textbf{PGM}}$	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				

Notation

L: Low level

H: High level

V_{PP}: V_{PP} level

V_{CC}: V_{CC} level

The specifications for writing and reading are identical to those for the standard HN27C101 EPROM. However, page programming is not supported, and so page programming mode must not be set. A PROM programmer that only supports page programming mode cannot be used. When selecting a PROM programmer, ensure that it supports high-speed, high-reliability byte-by-byte programming. Also, be sure to specify addresses from H'0000 to H'3FFF.

6.3.1 Writing and Verifying

An efficient, high-speed, high-reliability method is available for writing and verifying the PROM data. This method achieves high speed without voltage stress on the device and without lowering the reliability of written data. The basic flow of this high-speed, high-reliability programming method is shown in figure 6.4.

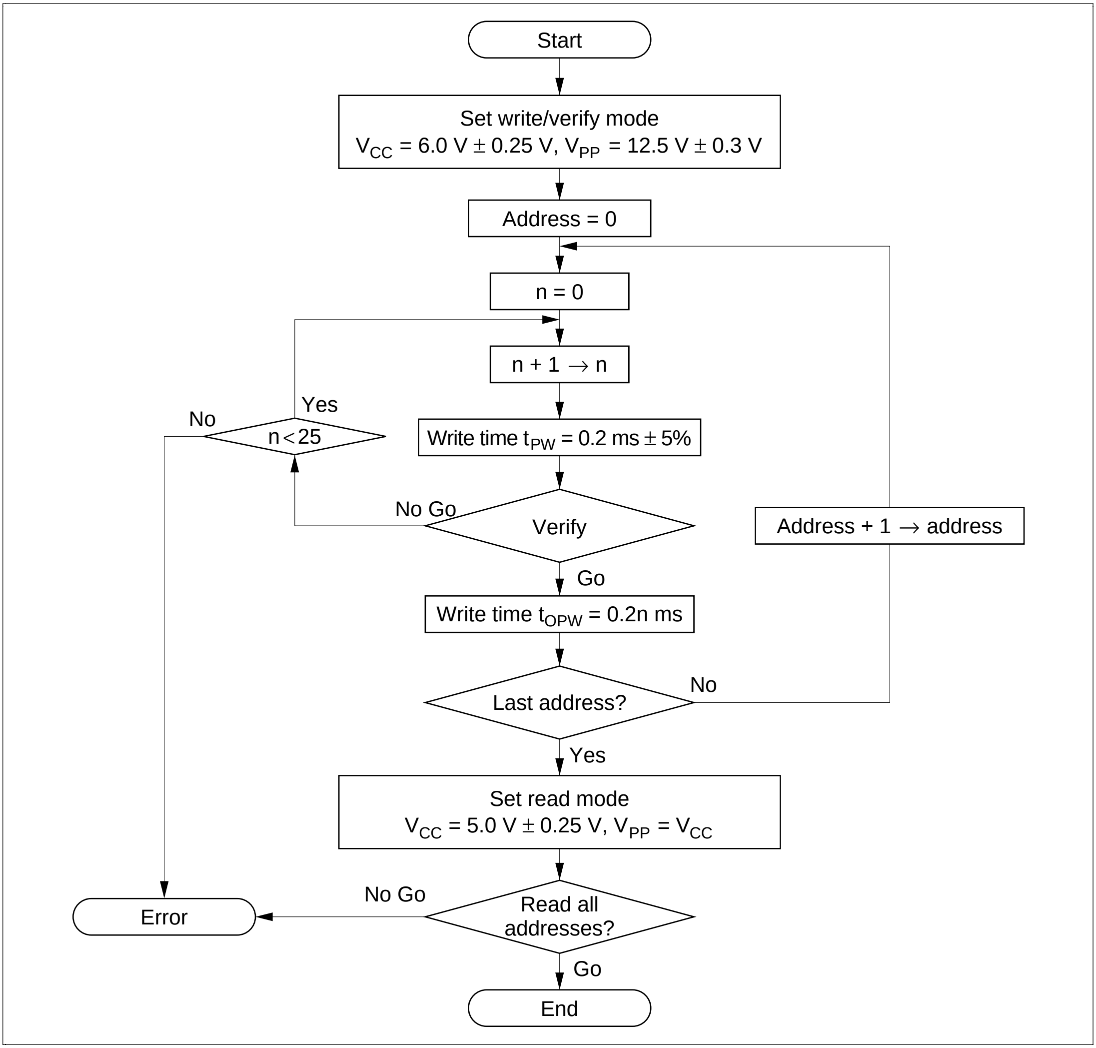


Figure 6.4 High-Speed, High-Reliability Programming Flow Chart

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 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	$\|I_{LI}\|$	—	—	2	µA	V_{in} = 5.25 V/ 0.5 V
V_{CC} current		I_{CC}	—	—	40	mA	
V_{PP} current		I_{PP}	—	—	40	mA	

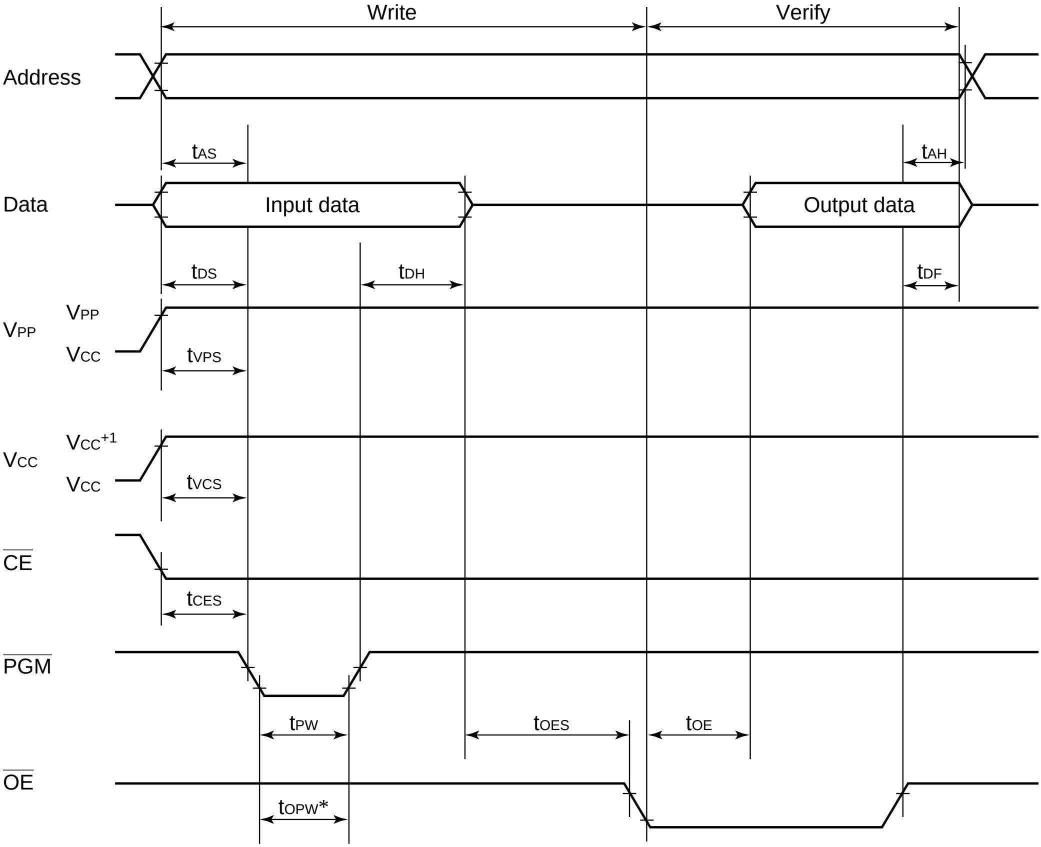
Table 6.4 AC Characteristics

(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 Condition
Address setup time	t_{AS}	2	—	—	µs	Figure 6.5*1
$\overline{\text{OE}}$ 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	
$\overline{\text{PGM}}$ pulse width for overwrite programming	t_{OPW}*3	0.19	—	5.25	ms	
$\overline{\text{CE}}$ setup time	t_{CES}	2	—	—	µs	
V_{CC} setup time	t_{VCS}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	200	ns	

Notes: 1. Input pulse level: 0.45 V to 2.2 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 figure 6.4, High-Speed, High-Reliability Programming Flow Chart.

Figure 6.5 shows a PROM write/verify timing diagram.



Note: * t_{OPW} is defined by the value shown in figure 6.4, High-Speed, High-Reliability Programming Flowchart.

Figure 6.5 PROM Write/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.
- Take care when setting the programming mode, as page programming is not supported.
- When programming with a PROM programmer, be sure to specify addresses from H'0000 to H'3FFF. If programming is inadvertently performed from H'4000 onward, it may not be possible to continue PROM programming and verification. When programming, H'FF should be set as the data in address area H'4000 to H'1FFFF.

6.4 Reliability of Programmed Data

A highly effective way to improve data retention characteristics is to bake the programmed chips at 150°C, then screen them for data errors. This procedure quickly eliminates chips with PROM memory cells prone to early failure.

Figure 6.6 Shows the recommended screening procedure.

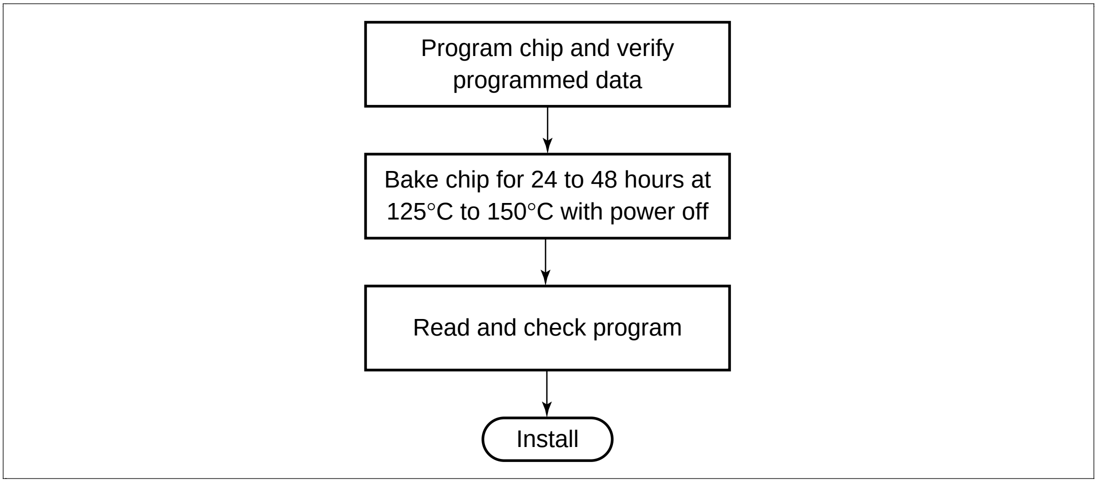


Figure 6.6 Recommended Screening Procedure

If a series of programming errors occurs while the same PROM programmer is in use, stop programming and check the PROM programmer and socket adapter for defects. Please inform Hitachi of any abnormal conditions noted during or after programming or in screening of program data after high-temperature baking.

Section 7 RAM

7.1 Overview

The H8/3802 has 1 kbyte of high-speed static RAM on-chip, and the H8/3801 and H8/3800 have 512 bytes. 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.

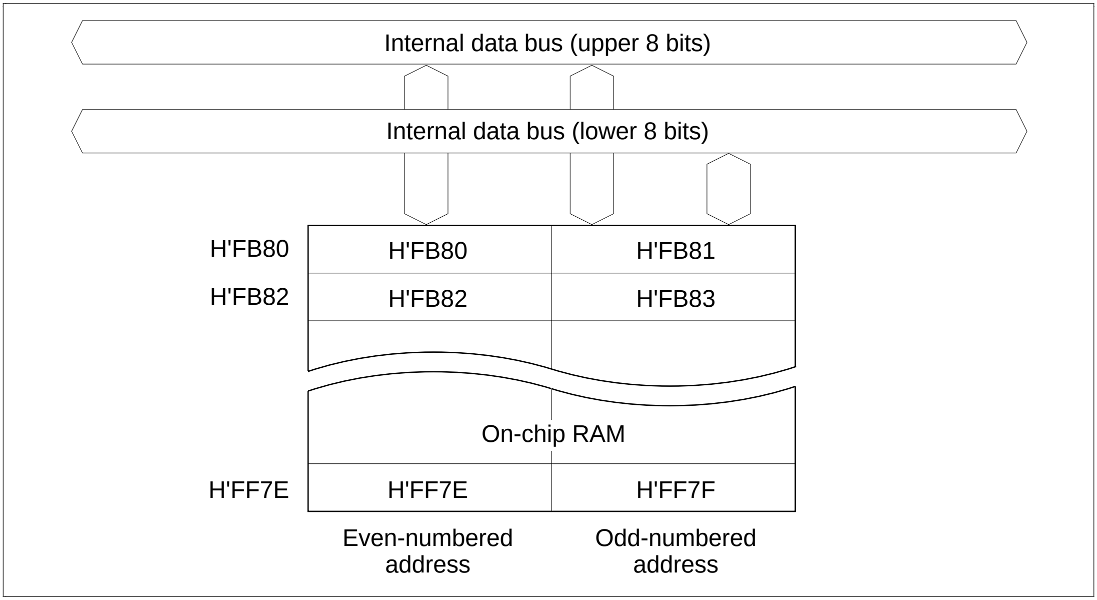


Figure 7.1 RAM Block Diagram (H8/3802)

Section 8 I/O Ports

8.1 Overview

The H8/3802 Series is provided with three 8-bit I/O ports, one 7-bit I/O port, one 4-bit I/O port, one 3-bit I/O port, one 1-bit I/O port, one 4-bit input-only port, one 1-bit input-only port, and one 6-bit output-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 assigned to individual bits.
See 2.9.2, Notes on Bit Manipulation, for information on executing bit-manipulation instructions to write data in PCR or PDR.

Ports 5, 6, 7, 8, and A are also used as liquid crystal display segment and common pins, selectable in 4-bit units.

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 Registers
Port 3	• 7-bit I/O port • MOS input pull-up option • Large-current port	$P3_7$/AEVL $P3_6$/AEVH $P3_5$, $P3_4$, $P3_3$	Asynchronous event counter event inputs AEVL, AEVH	PMR3
		$P3_2$, TMOFH $P3_1$, TMOFL	Timer F output compare output	PMR3
Port 4	• 1-bit input port • 3-bit I/O port	$P4_3/\overline{\text{IRQ}_0}$	External interrupt 0	PMR2
		$P4_2/TXD_{32}$ $P4_1/RXD_{32}$ $P4_0/SCK_{32}$	SCI3 data output (TXD_{32}), data input (RXD_{32}), clock input/output (SCK_{32})	SCR3 SMR
Port 5	• 8-bit I/O port • MOS input pull-up option	$P5_7$ to $P5_0$/ $\overline{\text{WKP}}_7$ to $\overline{\text{WKP}}_0$/ SEG_8 to SEG_1	Wakeup input ($\overline{\text{WKP}}_7$ to $\overline{\text{WKP}}_0$), segment output (SEG_8 to SEG_1)	PMR5 LPCR

Port	Description	Pins	Other Functions	Function Switching Registers
Port 6	• 8-bit I/O port • MOS input pull-up option	$P6_7$ to $P6_0$/ SEG_{16} to SEG_9	Segment output (SEG_{16} to SEG_9)	LPCR
Port 7	• 8-bit I/O port	$P7_7$ to $P7_0$/ SEG_{24} to SEG_{17}	Segment output (SEG_{24} to SEG_{17})	LPCR
Port 8	• 1-bit I/O port	$P8_0$/SEG_{25},	Segment output	LPCR
Port 9	• 6-bit output port	$P9_5$ to $P9_2$	None	
	• High-voltage, large-current port	$P9_1$, $P9_0$/ PWM2, PWM1	10-bit PWM output	PMR9
	• High-voltage port	IRQAEC	None	
Port A	4-bit I/O port	PA_3 to PA_0/ COM_4 to COM_1	Common output (COM_4 to COM_1)	LPCR
Port B	4-bit input port	PB_3 to PB_0/ AN_3 to AN_0	A/D converter analog input	AMR

8.2 Port 3

8.2.1 Overview

Port 3 is a 7-bit I/O port, configured as shown in figure 8.1.

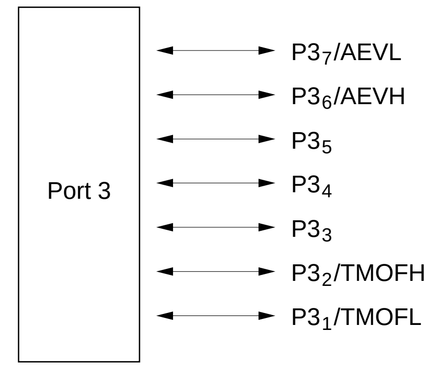


Figure 8.1 Port 3 Pin Configuration

8.2.2 Register Configuration and Description

Table 8.2 shows the port 3 register configuration.

Table 8.2 Port 3 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 3	PDR3	R/W	—	H'FFD6
Port control register 3	PCR3	W	—	H'FFE6
Port pull-up control register 3	PUCR3	R/W	—	H'FFE1
Port mode register 3	PMR3	R/W	—	H'FFCA
Port mode register 2	PMR2	R/W	—	H'FFC9

1. Port data register 3 (PDR3)

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

PDR3 is an 8-bit register that stores data for port 3 pins $P3_7$ to $P3_1$. If port 3 is read while PCR3 bits are set to 1, the values stored in PDR3 are read, regardless of the actual pin states. If port 3 is read while PCR3 bits are cleared to 0, the pin states are read.

2. Port control register 3 (PCR3)

Bit	7	6	5	4	3	2	1	0
	$PCR3_7$	$PCR3_6$	$PCR3_5$	$PCR3_4$	$PCR3_3$	$PCR3_2$	$PCR3_1$	—
Initial value	0	0	0	0	0	0	0	—
Read/Write	W	W	W	W	W	W	W	W

PCR3 is an 8-bit register for controlling whether each of the port 3 pins $P3_7$ to $P3_1$ functions as an input pin or output pin. Setting a PCR3 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 PCR3 and in PDR3 are valid only when the corresponding pin is designated in PMR3 as a general I/O pin.

PCR3 is a write-only register. Bits 7 to 1 are always read as 1. Bit 0 is reserved; only 0 can be written to this bit.

3. Port pull-up control register 3 (PUCR3)

Bit	7	6	5	4	3	2	1	0
	$PUCR3_7$	$PUCR3_6$	$PUCR3_5$	$PUCR3_4$	$PUCR3_3$	$PUCR3_2$	$PUCR3_1$	—
Initial value	0	0	0	0	0	0	0	—
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	W

PUCR3 controls whether the MOS pull-up of each of the port 3 pins $P3_7$ to $P3_1$ is on or off. When a PCR3 bit is cleared to 0, setting the corresponding PUCR3 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.

Bit 0 is reserved; only 0 can be written to this bit.

4. Port mode register 3 (PMR3)

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

PMR3 is an 8-bit read/write register, controlling the selection of pin functions for port 3 pins.

Bit 7: $P3_7$/AEVL pin function switch (AEVL)

This bit selects whether pin $P3_7$/AEVL is used as $P3_7$ or as AEVL.

Bit 7 AEVL	Description	
0	Functions as $P3_7$ I/O pin	(initial value)
1	Functions as AEVL input pin	

Bit 6: $P3_6$/AEVH pin function switch (AEVH)

This bit selects whether pin $P3_6$/AEVH is used as $P3_6$ or as AEVH.

Bit 6 AEVH	Description	
0	Functions as $P3_6$ I/O pin	(initial value)
1	Functions as AEVH input pin	

Bits 5 to 3: Reserved bits

Bits 5 to 3 are reserved; only 0 can be written to these bits.

Bit 2: $P3_2$/TMOFH pin function switch (TMOFH)

This bit selects whether pin $P3_2$/TMOFH is used as $P3_2$ or as TMOFH.

Bit 2 TMOFH	Description	
0	Functions as $P3_2$ I/O pin	
1	Functions as TMOFH output pin	(initial value)

Bit 1: $P3_1$/TMOFL pin function switch (TMOFL)

This bit selects whether pin $P3_1$/TMOFL is used as $P3_1$ or as TMOFL.

Bit 1 TMOFL	Description	
0	Functions as $P3_1$ I/O pin	(initial value)
1	Functions as TMOFL output pin	

Bit 0: Reserved bit

Bit 0 is reserved; only 0 can be written to this bit.

5. Port mode register 2 (PMR2)

Bit	7	6	5	4	3	2	1	0
	—	—	POF1	—	—	—	—	IRQ_0
Initial value	1	1	0	1	1	—	—	0
Read/Write	—	—	R/W	—	—	W	W	R/W

PMR2 is an 8-bit read/write register controlling the PMOS on/off state for the $P3_5$ pin.

Bit 5: $P3_5$ pin PMOS control (POF1)

This bit controls the on/off state of the $P3_5$ pin output buffer PMOS.

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

8.2.3 Pin Functions

Table 8.3 shows the port 3 pin functions.

Table 8.3 Port 3 Pin Functions

Pin	Pin Functions and Selection Method
$P3_7$/AEVL	The pin function depends on bit AEVL in PMR3 and bit $PCR3_2$ in PCR3.

AEVL	0		1
$PCR3_7$	0	1	*
Pin function	$P3_7$ input pin	$P3_7$ output pin	AEVL input pin

Pin	Pin Functions and Selection Method
$P3_6$/AEVH	The pin function depends on bit AEVH in PMR3 and bit $PCR3_6$ in PCR3.

AEVH	0		1
$PCR3_6$	0	1	*
Pin function	$P3_6$ input pin	$P3_6$ output pin	AEVH input pin

Pin	Pin Functions and Selection Method
$P3_5$ to $P3_3$	The pin function depends on the corresponding bit in PCR3.

PCR3n	0	1
Pin function	$P3_n$ input pin	$P3_n$ output pin

(n = 5 to 3)

Pin	Pin Functions and Selection Method
$P3_2$/TMOFH	The pin function depends on bit TMOFH in PMR3 and bit $PCR3_2$ in PCR3.

TMOFH	0		1
$PCR3_2$	0	1	*
Pin function	$P3_2$ input pin	$P3_2$ output pin	TMOFH output pin

Pin	Pin Functions and Selection Method
$P3_1$/TMOFL	The pin function depends on bit TMOFL in PMR3 and bit $PCR3_1$ in PCR3.

TMOFL	0		1
$PCR3_1$	0	1	*
Pin function	$P3_1$ input pin	$P3_1$ output pin	THOFL output pin

*: Don’t care

8.2.4 Pin States

Table 8.4 shows the port 3 pin states in each operating mode.

Table 8.4 Port 3 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P3_7$/AEVL $P3_6$/AEVH $P3_5$ $P3_4$ $P3_3$ $P3_2$/TMOFH $P3_1$/TMOFL	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 3 has a built-in MOS input pull-up function that can be controlled by software. When a PCR3 bit is cleared to 0, setting the corresponding PUCR3 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.

$PCR3_n$	0	0	1
$PUCR3_n$	0	1	*
MOS input pull-up	Off	On	Off

(n = 7 to 1)

*: Don't care

8.3 Port 4

8.3.1 Overview

Port 4 is a 3-bit I/O port and 1-bit input port, configured as shown in figure 8.2.

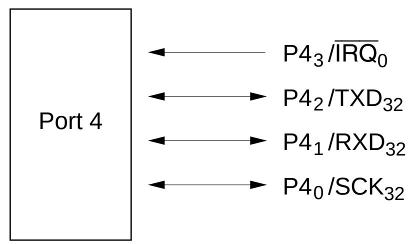


Figure 8.2 Port 4 Pin Configuration

8.3.2 Register Configuration and Description

Table 8.5 shows the port 4 register configuration.

Table 8.5 Port 4 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 4	PDR4	R/W	H'F8	H'FFD7
Port control register 4	PCR4	W	H'F8	H'FFE7
Port mode register 2	PMR2	R/W	—	H'FFC9

1. Port data register 4 (PDR4)

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

PDR4 is an 8-bit register that stores data for port 4 pins $P4_2$ to $P4_0$. If port 4 is read while PCR4 bits are set to 1, the values stored in PDR4 are read, regardless of the actual pin states. If port 4 is read while PCR4 bits are cleared to 0, the pin states are read.

Upon reset, PDR4 is initialized to H'F8.

2. Port control register 4 (PCR4)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PCR4_2$	$PCR4_1$	$PCR4_0$
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	W	W	W

PCR4 is an 8-bit register for controlling whether each of port 4 pins $P4_2$ to $P4_0$ functions as an input pin or output pin. Setting a PCR4 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. PCR4 and PDR4 settings are valid when the corresponding pins are designated for general-purpose input/output by SCR3-2.

Upon reset, PCR4 is initialized to H'F8.

PCR4 is a write-only register, which is always read as all 1s.

3. Port mode register 2 (PMR2)

Bit	7	6	5	4	3	2	1	0
	—	—	POF1	—	—	—	—	IRQ_0
Initial value	1	1	0	1	1	—	—	0
Read/Write	—	—	R/W	—	—	W	W	R/W

PMR2 is an 8-bit read/write register controlling the selection of the $P4_3$/IRQ_0 pin function and the PMOS on/off state for the $P3_5$ pin. Upon reset, PMR2 is initialized to H'DE.

Bits 7, 6, 4, and 3: Reserved bits

Bits 7, 6, and 4 to 1 are reserved; they are always read as 1 and cannot be modified.

Bit 5: $P3_5$ pin PMOS control (POF1)

This bit controls the on/off state of the $P3_5$ pin output buffer PMOS.

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

Bits 2 and 1: Reserved bits

Bits 2 and 1 are reserved; only 0 can be written to these bits.

Bit 0: $P4_3/IRQ_0$ pin function switch (IRQ_0)

This bit selects whether pin $P4_3/\overline{IRQ_0}$ is used as $P4_3$ or as $\overline{IRQ_0}$.

Bit 0 IRQ_0	Description	
0	Functions as $P4_3$ input pin	(initial value)
1	Functions as $\overline{IRQ_0}$ input pin	

8.3.3 Pin Functions

Table 8.6 shows the port 4 pin functions.

Table 8.6 Port 4 Pin Functions

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P4_3/\overline{IRQ_0}$</td><td>The pin function depends on bit IRQ0 in PMR2.
<table>
<tr><td>IRQ0</td><td>0</td><td>1</td></tr>
<tr><td>Pin function</td><td>$P4_3$ input pin</td><td>$\overline{IRQ_0}$ input pin</td></tr>
</table></td></tr>
<tr><td>$P4_2/TXD_{32}$</td><td>The pin function depends on bit TE in SCR3, bit SPC32 in SPCR, and bit $PCR4_2$ in PCR4.
<table>
<tr><td>SPC32</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>TE</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR4_2$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P4_2$ input pin</td><td>$P4_2$ output pin</td><td>TXD_{32} output pin</td></tr>
</table></td></tr>
<tr><td>$P4_1/RXD_{32}$</td><td>The pin function depends on bit RE in SCR3 and bit $PCR4_1$ in PCR4.
<table>
<tr><td>RE</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR4_1$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P4_1$ input pin</td><td>$P4_1$ output pin</td><td>RXD_{32} input pin</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Pin</th><th colspan="6">Pin Functions and Selection Method</th></tr>
<tr><td rowspan="6">$P4_0/SCK_{32}$</td><td colspan="6">The pin function depends on bit CKE1 and CKE0 in SCR3, bit COM in SMR, and bit $PCR4_0$ in PCR4.</td></tr>
<tr><td>CKE1</td><td colspan="4">0</td><td>1</td></tr>
<tr><td>CKE0</td><td colspan="3">0</td><td>1</td><td>*</td></tr>
<tr><td>COM</td><td colspan="2">0</td><td>1</td><td>*</td><td>*</td></tr>
<tr><td>$PCR4_0$</td><td>0</td><td>1</td><td colspan="2">*</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P4_0$ input pin</td><td>$P4_0$ output pin</td><td colspan="2">SCK_{32} output pin</td><td>SCK_{32} input pin</td></tr>
</table>

*: Don't care

8.3.4 Pin States

Table 8.7 shows the port 4 pin states in each operating mode.

Table 8.7 Port 4 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P4_3/IRQ_0$ $P4_2/TXD_{32}$ $P4_1/RXD_{32}$ $P4_0/SCK_{32}$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.4 Port 5

8.4.1 Overview

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

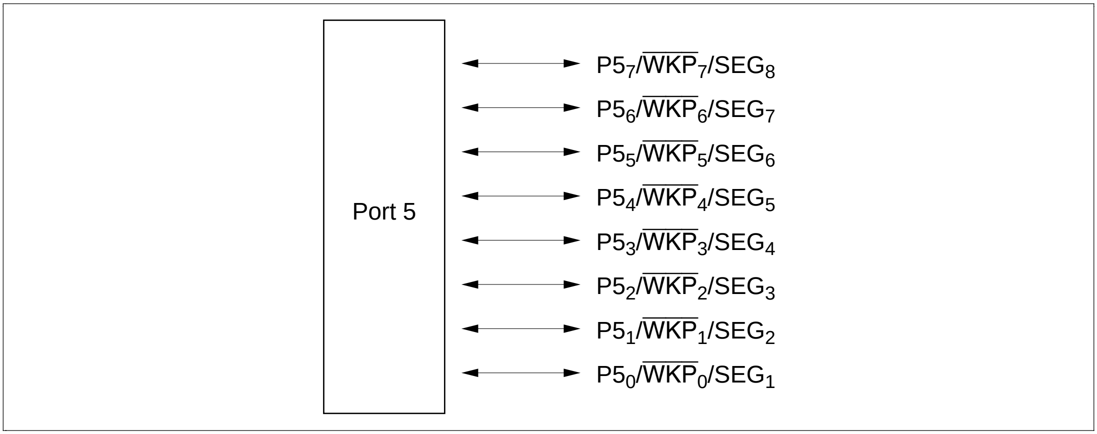


Figure 8.3 Port 5 Pin Configuration

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'FFE2
Port mode register 5	PMR5	R/W	H'00	H'FFCC

1. 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 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.

2. 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. PCR5 and PDR5 settings are valid when the corresponding pins are designated for general-purpose input/output by PMR5 and bits SGS3 to SGS0 in LPCR.

Upon reset, PCR5 is initialized to H'00.

PCR5 is a write-only register, which is always read as all 1s.

3. 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 of port 5 pins $P5_7$ to $P5_0$ 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.

4. 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: $P5_n/\overline{WKP}_n/SEG_{n+1}$ pin function switch (WKPn)

When pin P5n/$\overline{\text{WKP}}$n/SEGn+1 is not used as SEG_{n+1}, these bits select whether the pin is used as P5n or $\overline{WKP}_n$.

Bit n WKPn	Description	
0	Functions as P5n I/O pin	(initial value)
1	Functions as $\overline{WKP}_n$ input pin	

(n = 7 to 0)

Note: For use as SEG_{n+1}, see 13.2.1, LCD Port Control Register (LPCR).

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
$P5_7/\overline{WKP}_7/SEG_8$ to $P5_0/\overline{WKP}_0/SEG_1$	The pin function depends on bit WKP_n in PMR5, bit $PCR5_n$ in PCR5, and bits SGS3 to SGS0 in LPCR.

$P5_7$ to $P5_4$ (n = 7 to 4)

SGS3 to SGS0	Other than 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001			0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001
WKP_n	0		1	*
$PCR5_n$	0	1	*	*
Pin function	$P5_n$ input pin	$P5_n$ output pin	$\overline{WKPn}$ input pin	SEGn+1 output pin

$P5_3$ to $P5_0$ (m= 3 to 0)

SGS3 to SGS0	Other than 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000			0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000
WKP_m	0		1	*
$PCR5_m$	0	1	*	*
Pin function	$P5_m$ input pin	$P5_m$ output pin	$\overline{WKPm}$ output pin	SEGm+1 output pin

*: 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
$P5_7/\overline{WKP_7}/SEG_8$ to $P5_0/\overline{WKP_0}/SEG_1$	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	0	1
$PUCR5_n$	0	1	*
MOS input pull-up	Off	On	Off

(n = 7 to 0)

*: Don’t care

8.5 Port 6

8.5.1 Overview

Port 6 is an 8-bit I/O port. The port 6 pin configuration is shown in figure 8.4.

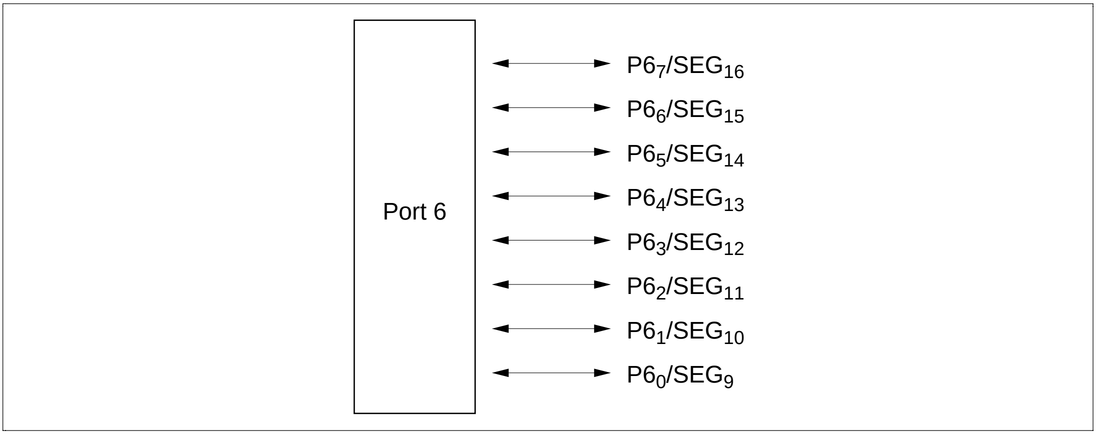


Figure 8.4 Port 6 Pin Configuration

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'FFE3

1. 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 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.

2. 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 ($P6_7$ to $P6_0$) an output pin, while clearing the bit to 0 makes the pin an input pin. PCR6 and PDR6 settings are valid when the corresponding pins are designated for general-purpose input/output by bits SGS3 to SGS0 in LPCR.

Upon reset, PCR6 is initialized to H'00.

PCR6 is a write-only register, which is always read as all 1s.

3. 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 of the port 6 pins $P6_7$ to $P6_0$ 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
$P6_7/SEG_{16}$ to $P6_0/SEG_9$	The pin function depends on bit PCR6n in PCR6 and bits SGS3 to SGS0 in LPCR.

$P6_7$ to $P6_4$ (n = 7 to 4)

SEG3 to SEGS0	Other than 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011		0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011
$PCR6_n$	0	1	*
Pin function	$P6_n$ input pin	$P6_n$ output pin	SEG_{n+9} output pin

$P6_3$ to $P6_0$ (m = 3 to 0)

SEG3 to SEGS0	Other than 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010		0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010
$PCR6_m$	0	1	*
Pin function	$P6_m$ input pin	$P6_m$ output pin	SEG_{n+9} output pin

*: Don’t care

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

Pin	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P6_7$/SEG_{16} to $P6_0$/SEG_9	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 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	0	1
$PUCR6_n$	0	1	*
MOS input pull-up	Off	On	Off

(n = 7 to 0)

*: Don't care

8.6 Port 7

8.6.1 Overview

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

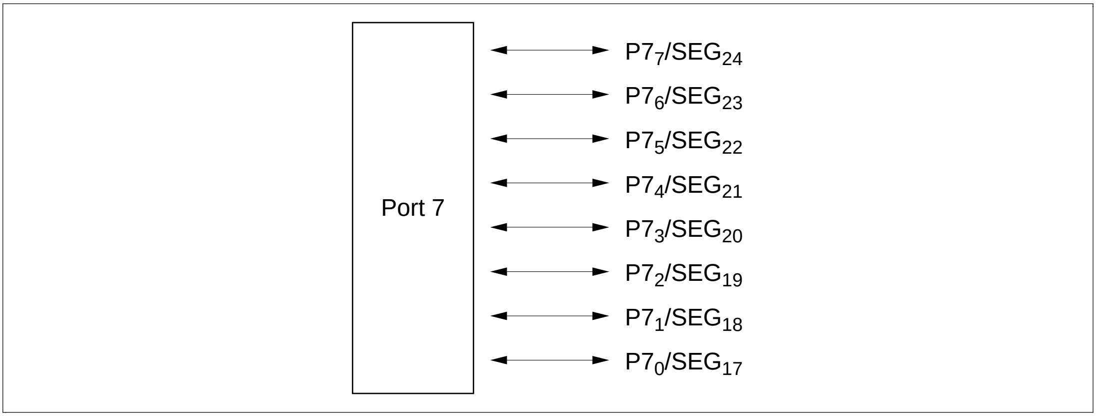


Figure 8.5 Port 7 Pin Configuration

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

1. 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 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.

2. 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. PCR7 and PDR7 settings are valid when the corresponding pins are designated for general-purpose input/output by bits SGS3 to SGS0 in LPCR.

Upon reset, PCR7 is initialized to H'00.

PCR7 is a write-only register, which is always read as all 1s.

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
$P7_7/SEG_{24}$ to $P7_0/SEG_{17}$	The pin function depends on bit $PCR7_n$ in PCR7 and bits SGS3 to SGS0 in LPCR.

$P7_7$ to $P7_4$ (n = 7 to 4)

SEGS3 to SEGS0	Other than 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101		0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101
$PCR7_n$	0	1	*
Pin function	$P7_n$ input pin	$P7_n$ output pin	SEG_{n+17} output pin

$P7_3$ to $P7_0$ (m = 3 to 0)

SEGS3 to SEGS0	Other than 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100		0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100
$PCR7_m$	0	1	*
Pin function	$P7_m$ input pin	$P7_m$ output pin	SEG_{m+17} output pin

*: Don’t care

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
$P7_7/SEG_{24}$ to $P7_0/SEG_{17}$	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 1-bit I/O port configured as shown in figure 8.6.

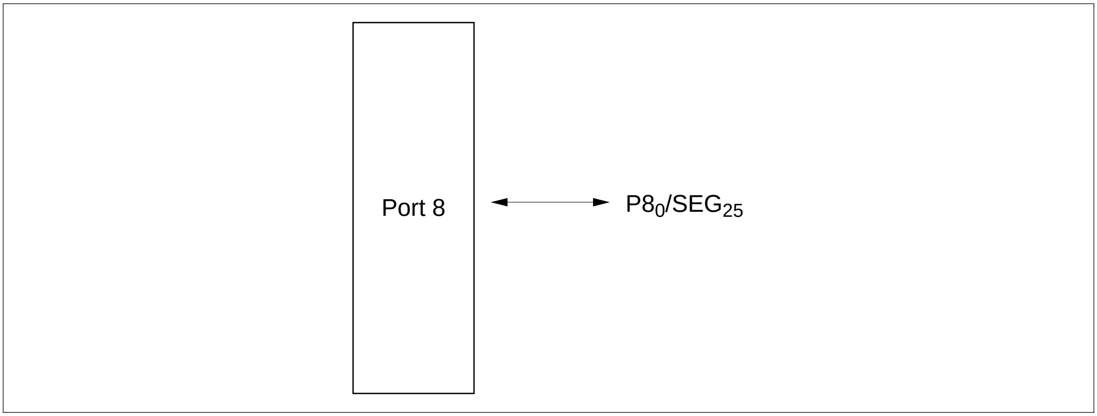


Figure 8.6 Port 8 Pin Configuration

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'FFDB
Port control register 8	PCR8	W	—	H'FFEB

1. Port data register 8 (PDR8)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	$P8_0$
Initial value	—	—	—	—	—	—	—	0
Read/Write	—	—	—	—	—	—	—	R/W

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

2. Port control register 8 (PCR8)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	PCR8
Initial value	—	—	—	—	—	—	—	0
Read/Write	W	W	W	W	W	W	W	W

PCR8 is an 8-bit register for controlling whether the port 8 pin $P8_0$ 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. PCR8 and PDR8 settings are valid when the corresponding pins are designated for general-purpose input/output by bits SGS3 to SGS0 in LPCR.

PCR8 is a write-only register, which is always read as all 1s.

Bits 7 to 1: Reserved bits

Bits 7 to 1 are reserved; only 0 can be written to these bits.

8.7.3 Pin Functions

Table 8.18 shows the port 8 pin functions.

Table 8.18 Port 8 Pin Functions

<table>
<tr><th>Pin</th><th colspan="4">Pin Functions and Selection Method</th></tr>
<tr><td rowspan="4">$P8_0/SEG_{25}$</td><td colspan="4">The pin function depends on bit $PCR8_n$ in PCR8 and bits SGS3 to SGS0 in LPCR.</td></tr>
<tr><td>SEGS3 to SEGS0</td><td colspan="2">Other than 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110</td><td>0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110</td></tr>
<tr><td>$PCR8_0$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P8_0$ input pin</td><td>$P8_0$ output pin</td><td>SEG_{25} output pin</td></tr>
</table>

*: Don’t care

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
$P8_0/SEG_{25}$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.8 Port 9

8.8.1 Overview

Port 9 is a 6-bit output-only port, configured as shown in figure 8.7.

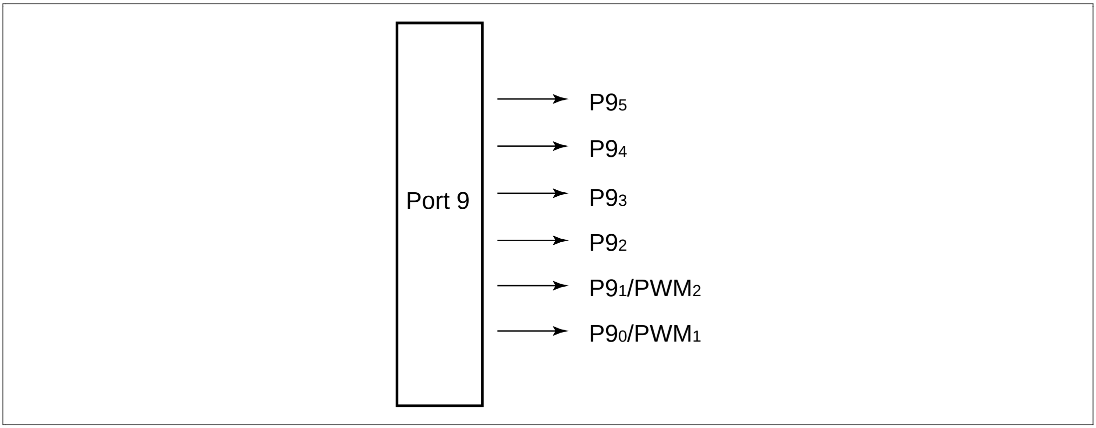


Figure 8.7 Port 5 Pin Configuration

8.8.2 Register Configuration and Description

Table 8.20 shows the port 9 register configuration.

Table 8.20 Port 9 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 9	PDR9	R/W	H'FF	H'FFDC
Port mode register 9	PMR9	R/W	—	H'FFEC

1. Port data register 9 (PDR9)

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

PDR9 is an 8-bit register that stores data for port 9 pins $P9_5$ to $P9_0$.

Upon reset, PDR9 is initialized to H'FF.

2. Port mode register 9 (PMR9)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PIOFF	—	PWM_2	PWM_1
Initial value	1	1	1	1	0	—	0	0
Read/Write	—	—	—	—	R/W	W	R/W	R/W

PMR9 is an 8-bit read/write register controlling the selection of the $P9_0$ and $P9_1$ pin functions.

Bit 3: $P9_2$ to $P9_0$ step-up circuit control (PIOFF)

Bit 3 turns the $P9_2$ to $P9_0$ step-up circuit on and off.

Bit 3 PIOFF	Description	
0	Large-current port step-up circuit is turned on	(initial value)
1	Large-current port step-up circuit is turned off	

Note: When turning the step-up circuit on or off, the register must be rewritten only when the buffer NMOS is off (port data is 1).

When turning the step-up circuit on, first clear PIOFF to 0, then wait for the elapse of 30 system clock before turning the buffer NMOS on (clearing port data to 0).

Without the elapse of the 30 system clock interval the step-up circuit will not start up, and it will not be possible for a large current to flow, making operation unstable.

Port 9 Pin Output Low Level Permitted Currents

Pin	Symbol	Test Conditions	Min	Typ	Max	PIOFF Bit Value
$P9_2$ to $P9_0$	I_{OL}	V_{CC} = 1.8 V to 5.5 V*	—	—	25 mA*	0
					10 mA	1
$P9_3$ to $P9_5$			—	—	10 mA	—

Note: * For details, see section 14.2.2, DC Characteristics.

Bit 2: Reserved bit

Bit 2 is reserved; only 0 can be written to this bit.

Bits 1 and 0: $P9_n$/PWM pin function switches

These pins select whether pin P9n/PWMn+1 is used as P9n or as PWMn+1.

Bit n WKPn+1	Description	
0	Functions as $P9_n$ output pin	(initial value)
1	Functions as PWM_{n+1} output pin	

(n = 0 or 1)

8.8.3 Pin Functions

Table 8.21 shows the port 9 pin functions.

Table 8.21 Port 9 Pin Functions

Pin	Pin Functions and Selection Method		
$P9_1/PWM_{n+1}$ to $P9_0/PWM_{n+1}$	The pin function depends on bit WKP_n in PMR5, bit $PCR5_n$ in PCR5, and bits SGS3 to SGS0 in LPCR. (n = 1 or 0)		
	$PMR9_n$	0	1
	Pin function	$P9_n$ output pin	PWM_{n+1} output pin

*: Don’t care

8.8.4 Pin States

Table 8.22 shows the port 5 pin states in each operating mode.

Table 8.22 Port 5 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P9_5$ to $P9_2$ $P9_n/PWM_{n+1}$ to $P9_n/PWM_{n+1}$	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

(n = 1 or 0)

8.9 Port A

8.9.1 Overview

Port A is a 4-bit I/O port, configured as shown in figure 8.8.

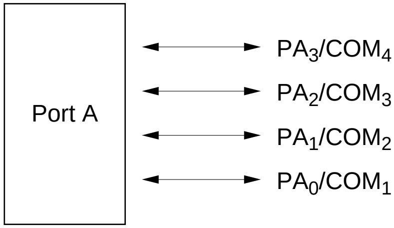


Figure 8.8 Port A Pin Configuration

8.9.2 Register Configuration and Description

Table 8.23 shows the port A register configuration.

Table 8.23 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'F0	H'FFED

1. Port data register A (PDRA)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PA_3	PA_2	PA_1	PA_0
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

PDRA is an 8-bit register that stores data for port A pins PA_3 to PA_0. If port A is read while PCRA bits are set to 1, the values stored in PDRA are 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.

2. Port control register A (PCRA)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	$PCRA_3$	$PCRA_2$	$PCRA_1$	$PCRA_0$
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

PCRA controls whether each of port A pins PA_3 to PA_0 functions as an input pin 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. PCRA and PDRA settings are valid when the corresponding pins are designated for general-purpose input/output by LPCR.

Upon reset, PCRA is initialized to H'F0.

PCRA is a write-only register, which is always read as all 1s.

8.9.3 Pin Functions

Table 8.24 shows the port A pin functions.

Table 8.24 Port A Pin Functions

Pin	Pin Functions and Selection Method			
PA_3/COM_4	The pin function depends on bit $PCRA_3$ in PCRA and bits SGS3 to SGS0.			
	SEGS3 to SEGS0	0000	0000	Not 0000
	$PCRA_3$	0	1	*
	Pin function	PA_3 input pin	PA_3 output pin	COM_4 output pin
PA_2/COM_3	The pin function depends on bit $PCRA_2$ in PCRA and bits SGS3 to SGS0.			
	SEGS3 to SEGS0	0000	0000	Not 0000
	$PCRA_2$	0	1	*
	Pin function	PA_2 input pin	PA_2 output pin	COM_3 output pin
PA_1/COM_2	The pin function depends on bit $PCRA_1$ in PCRA and bits SGS3 to SGS0.			
	SEGS3 to SEGS0	0000	0000	Not 0000
	$PCRA_1$	0	1	*
	Pin function	PA_1 input pin	PA_1 output pin	COM_2 output pin
PA_0/COM_1	The pin function depends on bit $PCRA_0$ in PCRA and bits SGS3 to SGS0.			
	SEGS3 to SEGS0	0000		Not 0000
	$PCRA_0$	0	1	*
	Pin function	PA_0 input pin	PA_0 output pin	COM_1 output pin

*: Don't care

8.9.4 Pin States

Table 8.25 shows the port A pin states in each operating mode.

Table 8.25 Port A Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
PA_3/COM_4 PA_2/COM_3 PA_1/COM_2 PA_0/COM_1	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.10 Port B

8.10.1 Overview

Port B is a 4-bit input-only port, configured as shown in figure 8.9.

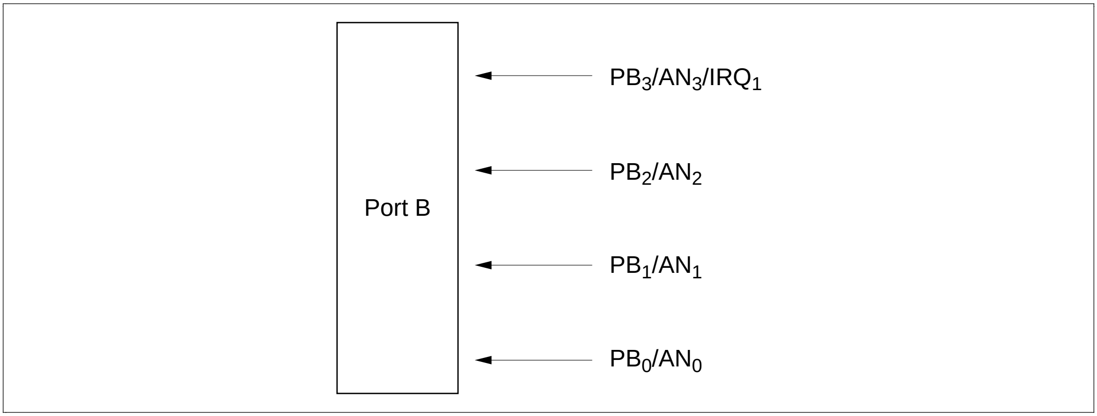


Figure 8.9 Port B Pin Configuration

8.10.2 Register Configuration and Description

Table 8.26 shows the port B register configuration.

Table 8.26 Port B Register

Name	Abbrev.	R/W	Initial Value	Address
Port data register B	PDRB	R	—	H'FFDE
Port mode register B	PMRB	R/W	H'F7	H'FFEE

1. Port Data Register B (PDRB)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PB_3	PB_2	PB_1	PB_0
Read/Write	—	—	—	—	R	R	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.

2. Port mode register B (PMRB)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	IRQ1	—	—	—
Initial value	1	1	1	1	0	1	1	1
Read/Write	—	—	—	—	R/W	—	—	—

PMRB is an 8-bit read/write register controlling the selection of the PB_3 pin function. Upon reset, PMRB is initialized to H'F7.

Bits 7 to 4 and 2 to 0: Reserved bits

Bits 7 to 4 and 2 to 0 are reserved; they are always read as 1 and cannot be modified.

Bit 3: $PB_3/AN_3/\overline{IRQ_1}$ pin function switch (IRQ1)

These bits select whether pin $PB_3/AN_3/\overline{IRQ_1}$ is used as PB_3/AN_3 or as $\overline{IRQ_1}$.

Bit 3 IRQ1	Description	
0	Functions as PB_3/AN_3 input pin	(initial value)
1	Functions as $\overline{IRQ_1}$ input pin	

Note: Rising or falling edge sensing can be selected for the $\overline{IRQ_1}$ pin.

8.10.3 Pin Functions

Table 8.27 shows the port B pin functions.

Table 8.27 Port B Pin Functions

Pin	Pin Functions and Selection Method
PB_3/AN_3/$\overline{IRQ_1}$	The pin function depends on bits CH3 to CH0 in AMR and bit IRQ_1 in PMRB.

IRQ_1	0		1
CH3 to CH0	Not 0111	1	*
Pin function	PB_3 input pin	AN_3 input pin	$\overline{IRQ_1}$ input pin

Pin	Pin Functions and Selection Method
PB_2/AN_2	The pin function depends on bits CH3 to CH0 in AMR.

CH3 to CH0	Not 0110	0110
Pin function	PB_2 input pin	AN_2 input pin

Pin	Pin Functions and Selection Method
PB_1/AN_1	The pin function depends on bits CH3 to CH0 in AMR.

CH3 to CH0	Not 0101	Not 0000
Pin function	PB_1 input pin	AN_1 input pin

Pin	Pin Functions and Selection Method
PB_0/AN_0	The pin function depends on bits CH3 to CH0 in AMR.

CH3 to CH0	Not 0100	0100
Pin function	PB_0 input pin	AN_0 input pin

*: Don't care

8.11 Input/Output Data Inversion Function

8.11.1 Overview

With input pin RXD_{32} and output pin TXD_{32}, the data can be handled in inverted form.

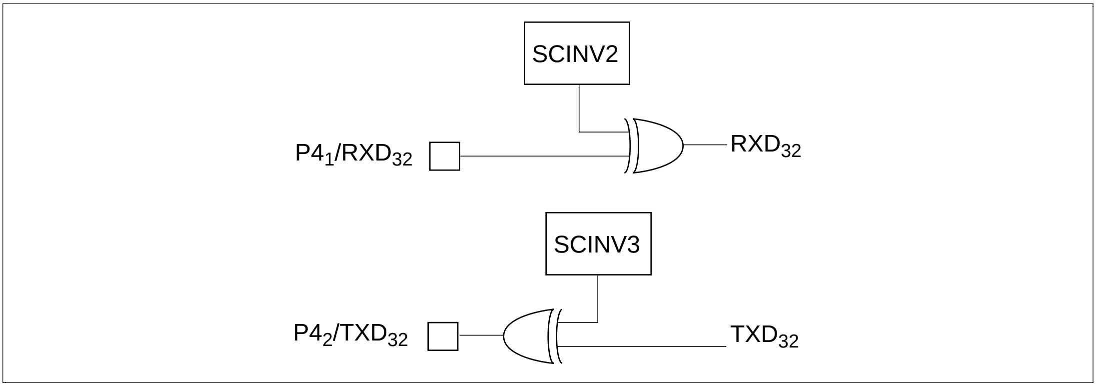


Figure 8.10 Input/Output Data Inversion Function

8.11.2 Register Configuration and Descriptions

Table 8.28 shows the registers used by the input/output data inversion function.

Table 8.28 Register Configuration

Name	Abbreviation	R/W	Address
Serial port control register	SPCR	R/W	H'FF91

Serial Port Control Register (SPCR)

Bit	7	6	5	4	3	2	1	0
	—	—	SPC32	—	SCINV3	SCINV2	—	—
Initial value	1	1	0	—	0	0	—	—
Read/Write	—	—	R/W	W	R/W	R/W	W	W

SPCR is an 8-bit readable/writable register that performs RXD_{32} and TXD_{32} pin input/output data inversion switching.

Bits 7 and 6: Reserved bits

Bits 7 and 6 are reserved; they are always read as 1 and cannot be modified.

Bit 5: $P4_2/TXD_{32}$ pin function switch (SPC32)

This bit selects whether pin $P4_2/TXD_{32}$ is used as $P4_2$ or as TXD_{32}.

Bit 5 SPC32	Description	
0	Functions as $P4_2$ I/O pin	(initial value)
1	Functions as TXD_{32} output pin*	

Note: * Set the TE bit in SCR3 after setting this bit to 1.

Bit 4: Reserved bit

Bit 4 is reserved; only 0 can be written to this bit.

Bit 3: TXD_{32} pin output data inversion switch

Bit 3 specifies whether or not TXD_{32} pin output data is to be inverted.

Bit 3 SCINV3	Description	
0	TXD_{32} output data is not inverted	(initial value)
1	TXD_{32} output data is inverted	

Bit 2: RXD_{32} pin input data inversion switch

Bit 2 specifies whether or not RXD_{32} pin input data is to be inverted.

Bit 2 SCINV2	Description	
0	RXD_{32} input data is not inverted	(initial value)
1	RXD_{32} input data is inverted	

Bits 1 and 0: Reserved bits

Bits 1 and 0 are reserved; only 0 can be written to these bits.

8.11.3 Note on Modification of Serial Port Control Register

When a serial port control register is modified, the data being input or output up to that point is inverted immediately after the modification, and an invalid data change is input or output. When modifying a serial port control register, do so in a state in which data changes are invalidated.

8.12 Application Note

8.12.1 How to Handle an Unused Pin

If an I/O pin not used by the user system is floating, pull it up or down.

- If an unused pin is an input pin, handle it in one of the following ways:
  - — Pull it up to V_{CC} with an on-chip pull-up MOS.
  - — Pull it up to V_{CC} with an external resistor of approximately 100 kΩ.
  - — Pull it down to V_{SS} with an external resistor of approximately 100 kΩ.
  - — For a pin also used by the A/D converter, pull it up to AV_{CC}.
- If an unused pin is an output pin, handle it in one of the following ways:
  - — Set the output of the unused pin to high and pull it up to V_{CC} with an on-chip pull-up MOS.
  - — Set the output of the unused pin to high and pull it up to V_{CC} with an external resistor of approximately 100 kΩ.
  - — Set the output of the unused pin to low and pull it down to GND with an external resistor of approximately 100 kΩ.

9.1 Overview

The H8/3802 Series provides three timers: timers A, F, and an asynchronous event counter. The functions of these timers are outlined in table 9.1.

Table 9.1 Timer Functions

Name	Functions	Internal Clock	Event Input Pin	Waveform Output Pin	Remarks
Timer A	• 8-bit interval timer	ø/8 to ø/8192	—	—	
	• Interval function	(8 choices)			
	• Time base	$ø_W$/128 (choice of 4 overflow periods)			
Timer F	• 16-bit timer • Also usable as two independent8-bit timers. • Output compare output function	ø/4 to ø/32, $ø_W$/4 (4 choices)	—	TMOFL TMOFH	
Asynchronous event counter	• 16-bit counter • Also usable as two independent 8-bit counters • Counts events asynchronous to ø and øw • Can count asynchronous events (rising/falling/both edges) independently of the MCU's internal clock	ø/2 to ø/8 (3 choices)	AEVL AEVH IRQAEC	—	

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.

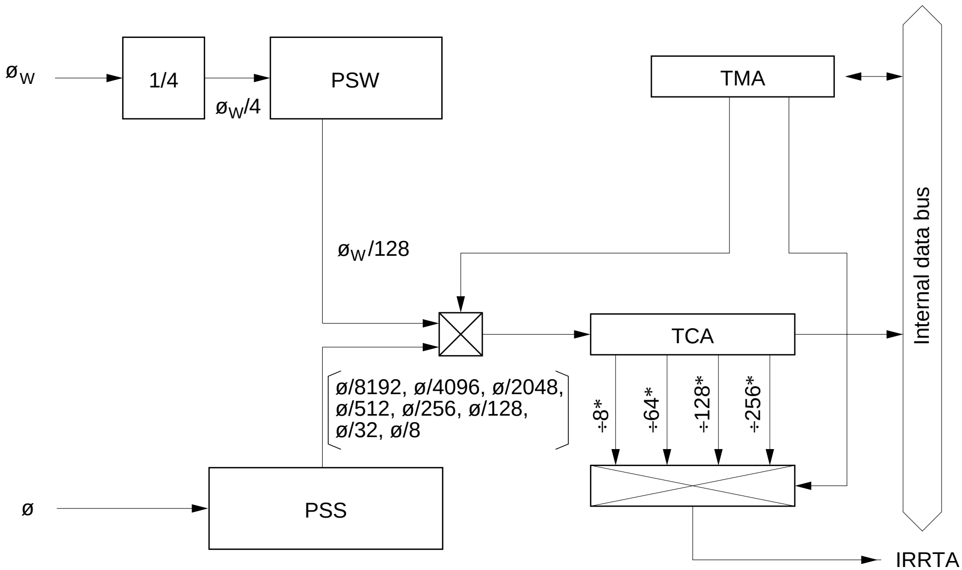
1. Features

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.
- Use of module standby mode enables this module to be placed in standby mode independently when not used.

2. Block diagram

Figure 9.1 shows a block diagram of timer A.



Notation:

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

3. Register configuration

Table 9.2 shows the register configuration of timer A.

Table 9.2 Timer A Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register A	TMA	R/W	—	H'FFB0
Timer counter A	TCA	R	H'00	H'FFB1
Clock stop register 1	CKSTPR1	R/W	H'FF	H'FFFA

9.2.2 Register Descriptions

1. Timer mode register A (TMA)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	TMA3	TMA2	TMA1	TMA0
Initial value	—	—	—	1	0	0	0	0
Read/Write	W	W	W	—	R/W	R/W	R/W	R/W

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

Bits 7 to 5: Reserved bits

Bits 7 to 5 are reserved; only 0 can be written to these bits.

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 (when using 32.768 kHz)
			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		

2. 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.

3. Clock stop register 1 (CKSTPR1)

Bit:	7	6	5	4	3	2	1	0
	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP
Initial value:	1	1	1	1	1	1	1	1
Read/Write:	—	—	R/W	R/W	—	R/W	—	R/W

CKSTPR1 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bit relating to timer A is described here. For details of the other bits, see the sections on the relevant modules.

Bit 0: Timer A module standby mode control (TACKSTP)

Bit 0 controls setting and clearing of module standby mode for timer A.

TACKSTP	Description	
0	Timer A is set to module standby mode	
1	Timer A module standby mode is cleared	(initial value)

9.2.3 Timer Operation

1. 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.

2. 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.

9.2.4 Timer A Operation States

Table 9.3 summarizes the timer A operation states.

Table 9.3 Timer A Operation States

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

Note: When the real-time clock time base function is 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.

9.2.5 Application Note

When bit 0 (TACKSTP) of the clock stop register 1 (CKSTPR1) is cleared to 0, bit 3 (TMA3) of the timer mode register A (TMA) cannot be rewritten.

Set bit 0 (TACKSTP) of the clock stop register 1 (CKSTPR1) to 1 before rewriting bit 3 (TMA3) of the timer mode register A (TMA).

9.3 Timer F

9.3.1 Overview

Timer F is a 16-bit timer with a built-in output compare function. Timer F also provides for counter resetting, interrupt request generation, toggle output, etc., using compare match signals. Timer F can also be used as two independent 8-bit timers (timer FH and timer FL).

1. Features

Features of timer F are given below.

- Choice of four internal clock sources (ø/32, ø/16, ø/4, øw/4)
- TMOFH pin toggle output provided using a single compare match signal (toggle output initial value can be set)
- Counter resetting by a compare match signal
- Two interrupt sources: one compare match, one overflow
- Can operate as two independent 8-bit timers (timer FH and timer FL) (in 8-bit mode).

	Timer FH 8-Bit Timer*	Timer FL 8-Bit Timer/Event Counter
Internal clock	Choice of 4 (ø/32, ø/16, ø/4, øw/4)	
Toggle output	One compare match signal, output to TMOFH pin(initial value settable)	One compare match signal, output to TMOFL pin (initial value settable)
Counter reset	Counter can be reset by compare match signal	
Interrupt sources	One compare match One overflow	

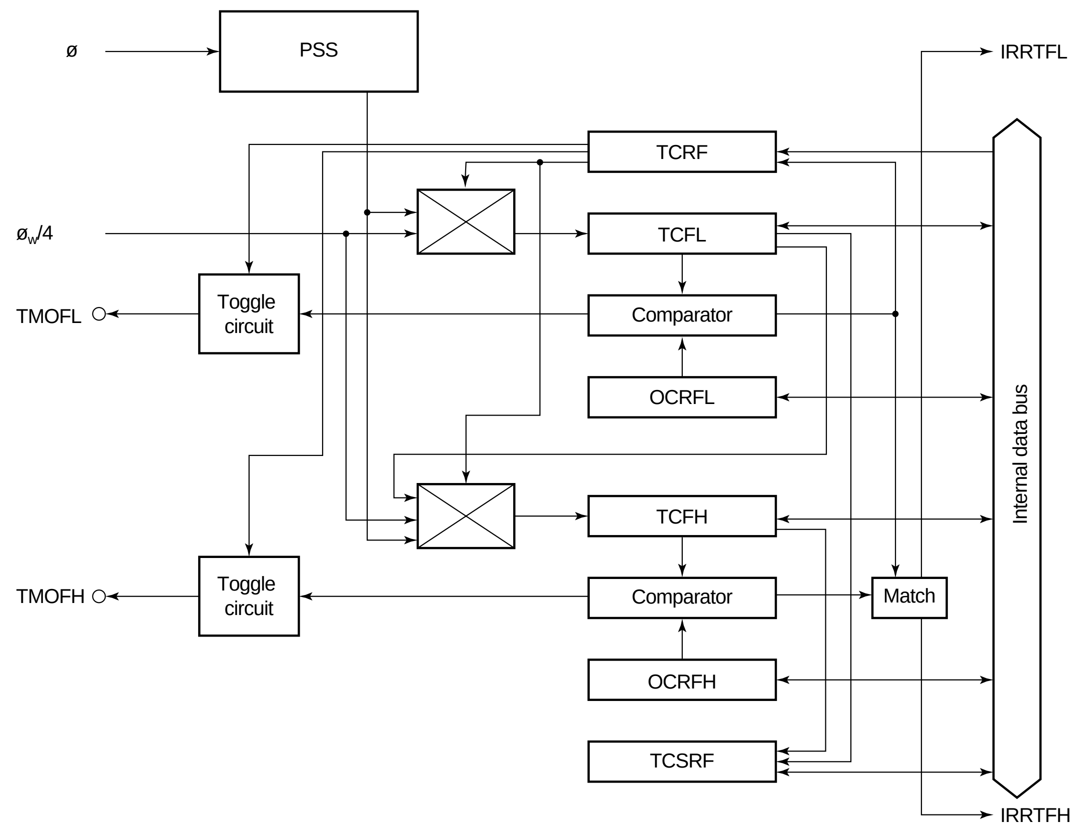
Note: * When timer F operates as a 16-bit timer, it operates on the timer FL overflow signal.

- Operation in watch mode, subactive mode, and subsleep mode

  When øw/4 is selected as the internal clock, timer F can operate in watch mode, subactive mode, and subsleep mode.
- Use of module standby mode enables this module to be placed in standby mode independently when not used.

2. Block diagram

Figure 9.2 shows a block diagram of timer F.



Notation:

TCRF:	Timer control register F
TCSRF:	Timer control/status register F
TCFH:	8-bit timer counter FH
TCFL:	8-bit timer counter FL
OCRFH:	Output compare register FH
OCRFL:	Output compare register FL
IRRTFH:	Timer FH interrupt request flag
IRRTFL:	Timer FL interrupt request flag
PSS:	Prescaler S

Figure 9.2 Block Diagram of Timer F

3. Pin configuration

Table 9.4 shows the timer F pin configuration.

Table 9.4 Pin Configuration

Name	Abbrev.	I/O	Function
Timer FH output	TMOFH	Output	Timer FH toggle output pin
Timer FL output	TMOFL	Output	Timer FL toggle output pin

4. Register configuration

Table 9.5 shows the register configuration of timer F.

Table 9.5 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
Clock stop register 1	CKSTPR1	R/W	H'FF	H'FFFA

9.3.2 Register Descriptions

1. 16-bit timer counter (TCF)
   8-bit timer counter (TCFH)
   8-bit timer counter (TCFL)

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

TCF is a 16-bit read/write up-counter configured by cascaded connection of 8-bit timer counters TCFH and TCFL. In addition to the use of TCF as a 16-bit counter with TCFH as the upper 8 bits and TCFL as the lower 8 bits, TCFH and TCFL can also be used as independent 8-bit counters.

TCFH and TCFL can be read and written by the CPU, but when they are used in 16-bit mode, data transfer to and from the CPU is performed via a temporary register (TEMP). For details of TEMP, see 9.3.3, CPU Interface.

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

a. 16-bit mode (TCF)

   When CKSH2 is cleared to 0 in TCRF, TCF operates as a 16-bit counter. The TCF input clock is selected by bits CKSL2 to CKSL0 in TCRF.

   TCF can be cleared in the event of a compare match by means of CCLRH in TCSRF.

   When TCF overflows from H'FFFF to H'0000, OVFH is set to 1 in TCSRF. If OVIEH in TCSRF is 1 at this time, IRRTFH is set to 1 in IRR2, and if IENTFH in IENR2 is 1, an interrupt request is sent to the CPU.

b. 8-bit mode (TCFL/TCFH)

   When CKSH2 is set to 1 in TCRF, TCFH and TCFL operate as two independent 8-bit counters. The TCFH (TCFL) input clock is selected by bits CKSH2 to CKSH0 (CKSL2 to CKSL0) in TCRF.

   TCFH (TCFL) can be cleared in the event of a compare match by means of CCLRH (CCLRL) in TCSRF.

   When TCFH (TCFL) overflows from H'FF to H'00, OVFH (OVFL) is set to 1 in TCSRF. If OVIEH (OVIEL) in TCSRF is 1 at this time, IRRTFH (IRRTFL) is set to 1 in IRR2, and if IENTFH (IENTFL) in IENR2 is 1, an interrupt request is sent to the CPU.

2. 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 register composed of the two registers OCRFH and OCRFL. In addition to the use of OCRF as a 16-bit register with OCRFH as the upper 8 bits and OCRFL as the lower 8 bits, OCRFH and OCRFL can also be used as independent 8-bit registers.

OCRFH and OCRFL can be read and written by the CPU, but when they are used in 16-bit mode, data transfer to and from the CPU is performed via a temporary register (TEMP). For details of TEMP, see 9.3.3, CPU Interface.

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

a. 16-bit mode (OCRF)

   When CKSH2 is cleared to 0 in TCRF, OCRF operates as a 16-bit register. OCRF contents are constantly compared with TCF, and when both values match, CMFH is set to 1 in TCSRF. At the same time, IRRTFH is set to 1 in IRR2. If IENTFH in IENR2 is 1 at this time, an interrupt request is sent to the CPU.

   Toggle output can be provided from the TMOFH pin by means of compare matches, and the output level can be set (high or low) by means of TOLH in TCRF.

b. 8-bit mode (OCRFH/OCRFL)

   When CKSH2 is set to 1 in TCRF, OCRFH and OCRFL operate as two independent 8-bit registers. OCRFH contents are compared with TCFH, and OCRFL contents are with TCFL. When the OCRFH (OCRFL) and TCFH (TCFL) values match, CMFH (CMFL) is set to 1 in TCSRF. At the same time, IRRTFH (IRRTFL) is set to 1 in IRR2. If IENTFH (IENTFL) in IENR2 is 1 at this time, an interrupt request is sent to the CPU.

   Toggle output can be provided from the TMOFH pin (TMOFL pin) by means of compare matches, and the output level can be set (high or low) by means of TOLH (TOLL) in TCRF.

3. 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 that switches between 16-bit mode and 8-bit mode, selects the input clock from among four internal clock sources and sets the output level of the TMOFH and TMOFL pins.

TCRF is initialized to H'00 upon reset.

Bit 7: Toggle output level H (TOLH)

Bit 7 sets the TMOFH pin output level. The output level is effective 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 clock input to TCFH from among four internal clock sources or TCFL overflow.

Bit 6 CKSH2	Bit 5 CKSH1	Bit 4 CKSH0	Description	
0	0	0	16-bit mode, counting on TCFL overflow signal	(initial value)
0	0	1		
0	1	0		
0	1	1	Use prohibited	
1	0	0	Internal clock: counting on ø/32	
1	0	1	Internal clock: counting on ø/16	
1	1	0	Internal clock: counting on ø/4	
1	1	1	Internal clock: counting on øw/4	

Bit 3: Toggle output level L (TOLL)

Bit 3 sets the TMOFL pin output level. The output level is effective 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 clock input to TCFL from among four internal clock sources or external event input.

Bit 2 CKSL2	Bit 1 CKSL1	Bit 0 CKSL0	Description	
0	0	0	Non-operational	(initial value)
0	0	1	Use prohibited	
0	1	0		
0	1	1		
1	0	0	Internal clock: counting on ø/32	
1	0	1	Internal clock: counting on ø/16	
1	1	0	Internal clock: counting on ø/4	
1	1	1	Internal clock: counting on øw/4	

4. 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

Note: * Bits 7, 6, 3, and 2 can only be written with 0, for flag clearing.

TCSRF is an 16-bit read/write register that performs counter clear selection, overflow flag setting, and compare match flag setting, and controls enabling of overflow interrupt requests.

TCSRF is initialized to H'00 upon reset.

Bit 7: Timer overflow flag H (OVFH)

Bit 7 is a status flag indicating that TCFH has overflowed from 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: After reading OVFH = 1, cleared by writing 0 to OVFH	(initial value)
1	Setting conditions: Set when TCFH overflows from H’FF to H’00	

Bit 6: Compare match flag H (CMFH)

Bit 6 is a status flag indicating that TCFH has matched 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 the OCRFH value	

Bit 5: Timer overflow interrupt enable H (OVIEH)

Bit 5 selects enabling or disabling of interrupt generation when TCFH overflows.

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

Bit 4: Counter clear H (CCLRH)

In 16-bit mode, bit 4 selects whether TCF is cleared when TCF and OCRF match.

In 8-bit mode, bit 4 selects whether TCFH is cleared when TCFH and OCRFH match.

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

Bit 3: Timer overflow flag L (OVFL)

Bit 3 is a status flag indicating that TCFL has overflowed from 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 TCFL overflows from H’FF to H’00	

Bit 2: Compare match flag L (CMFL)

Bit 2 is a status flag indicating that TCFL has matched 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 selects enabling or disabling of interrupt generation when TCFL overflows.

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

Bit 0: Counter clear L (CCLRL)

Bit 0 selects whether TCFL is cleared when TCFL and OCRFL match.

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

5. Clock stop register 1 (CKSTPR1)

Bit:	7	6	5	4	3	2	1	0
	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP
Initial value:	1	1	1	1	1	1	1	1
Read/Write:	—	—	R/W	R/W	—	R/W	—	R/W

CKSTPR1 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bit relating to timer F is described here. For details of the other bits, see the sections on the relevant modules.

Bit 2: Timer F module standby mode control (TFCKSTP)

Bit 2 controls setting and clearing of module standby mode for timer F.

TFCKSTP	Description	
0	Timer F is set to module standby mode	
1	Timer F module standby mode is cleared	(initial value)

9.3.3 CPU Interface

TCF and OCRF are 16-bit read/write registers, but the CPU is connected to the on-chip peripheral modules by an 8-bit data bus. When the CPU accesses these registers, it therefore uses an 8-bit temporary register (TEMP).

In 16-bit mode, TCF read/write access and OCRF write access must be performed 16 bits at a time (using two consecutive byte-size MOV instructions), and the upper byte must be accessed before the lower byte. Data will not be transferred correctly if only the upper byte or only the lower byte is accessed.

In 8-bit mode, there are no restrictions on the order of access.

1. Write access

Write access to the upper byte results in transfer of the upper-byte write data to TEMP. Next, write access to the lower byte results in transfer of the data in TEMP to the upper register byte, and direct transfer of the lower-byte write data to the lower register byte.

Figure 9.3 shows an example in which H'AA55 is written to TCF.

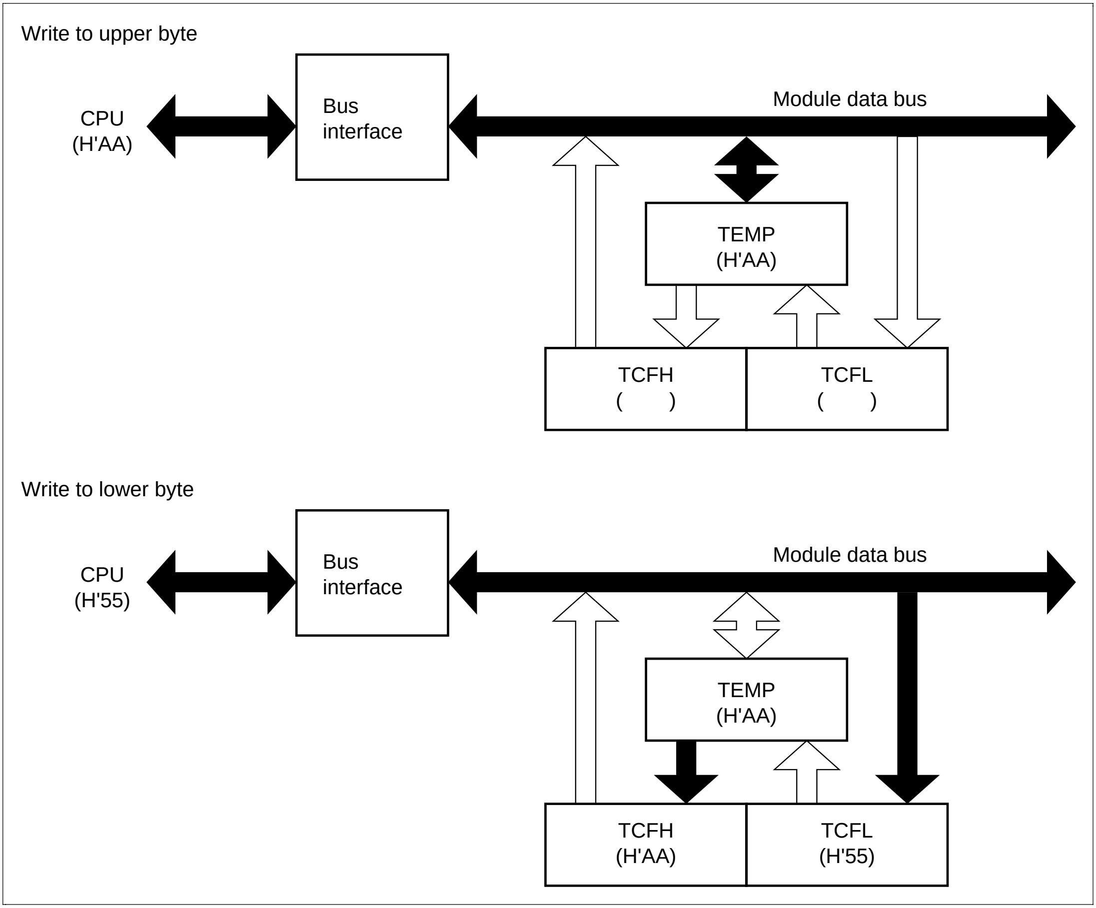


Figure 9.3 Write Access to TCR (CPU → TCF)

2. Read access

In access to TCF, when the upper byte is read the upper-byte data is transferred directly to the CPU and the lower-byte data is transferred to TEMP. Next, when the lower byte is read, the lower-byte data in TEMP is transferred to the CPU.

In access to OCRF, when the upper byte is read the upper-byte data is transferred directly to the CPU. When the lower byte is read, the lower-byte data is transferred directly to the CPU.

Figure 9.4 shows an example in which TCF is read when it contains H'AAFF.

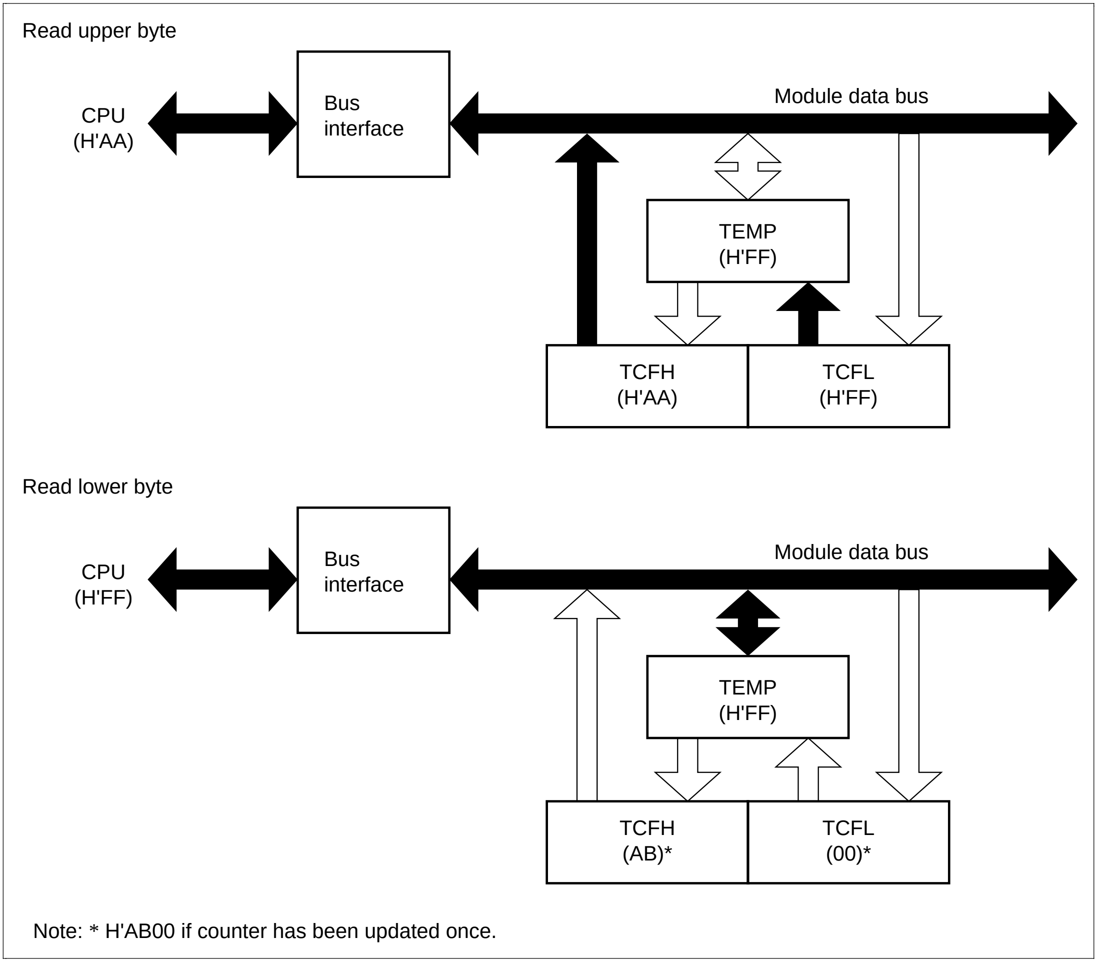


Figure 9.4 Read Access to TCF (TCF → CPU)

9.3.4 Operation

Timer F is a 16-bit counter that increments on each input clock pulse. The timer F value is constantly compared with the value set in output compare register F, and the counter can be cleared, an interrupt requested, or port output toggled, when the two values match. Timer F can also function as two independent 8-bit timers.

1. Timer F operation

Timer F has two operating modes, 16-bit timer mode and 8-bit timer mode. The operation in each of these modes is described below.

a. Operation in 16-bit timer mode

   When CKSH2 is cleared to 0 in timer control register F (TCRF), timer F operates as a 16-bit timer.

   Following a reset, timer counter F (TCF) is initialized 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.

   The timer F operating clock can be selected from three internal clocks output by prescaler S by means of bits CKSL2 to CKSL0 in TCRF.

   OCRF contents are constantly compared with TCF, and when both values match, CMFH is set to 1 in TCSRF. If IENTFH in IENR2 is 1 at this time, an interrupt request is sent to the CPU, and at the same time, TMOFH pin output is toggled. If CCLRH in TCSRF is 1, TCF is cleared. TMOFH pin output can also be set by TOLH in TCRF.

   When TCF overflows from H'FFFF to H'0000, OVFH is set to 1 in TCSRF. If OVIEH in TCSRF and IENTFH in IENR2 are both 1, an interrupt request is sent to the CPU.

b. Operation in 8-bit timer mode

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

   When the OCRFH/OCRFL and TCFH/TCFL values match, CMFH/CMFL is set to 1 in TCSRF. If IENTFH/IENTFL in IENR2 is 1, an interrupt request is sent to the CPU, and at the same time, TMOFH pin/TMOFL pin output is toggled. If CCLRH/CCLRL in TCSRF is 1, TCFH/TCFL is cleared. TMOFH pin/TMOFL pin output can also be set by TOLH/TOLL in TCRF.

   When TCFH/TCFL overflows from H'FF to H'00, OVFH/OVFL is set to 1 in TCSRF. If OVIEH/OVIEL in TCSRF and IENTFH/IENTFL in IENR2 are both 1, an interrupt request is sent to the CPU.

2. TCF increment timing

TCF is incremented by clock input (internal clock input). Bits CKSH2 to CKSH0 or CKSL2 to CKSL0 in TCRF select one of four internal clock sources (ø/32, ø/16, ø/4, or øw/4) created by dividing the system clock (ø or øw).

3. TMOFH/TMOFL output timing

In TMOFH/TMOFL output, the value set in TOLH/TOLL in TCRF is output. The output is toggled by the occurrence of a compare match. Figure 9.5 shows the output timing.

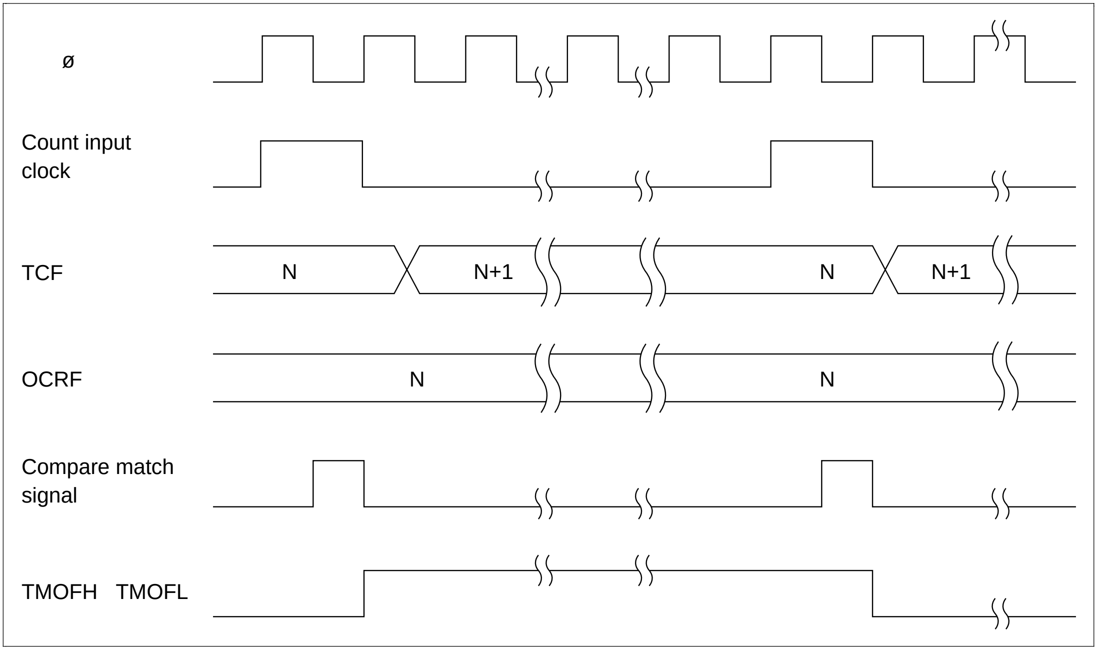


Figure 9.5 TMOFH/TMOFL Output Timing

4. TCF clear timing

TCF can be cleared by a compare match with OCRF.

5. Timer overflow flag (OVF) set timing

OVF is set to 1 when TCF overflows from H'FFFF to H'0000.

6. Compare match flag set timing

The compare match flag (CMFH or CMFL) is set to 1 when the TCF and OCRF values match. The compare match signal is generated in the last state during which the values match (when TCF is updated from the matching value to a new value). When TCF matches OCRF, the compare match signal is not generated until the next counter clock.

7. Timer F operation modes

Timer F operation modes are shown in table 9.6.

Table 9.6 Timer F Operation Modes

Operation Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby	Module Standby
TCF	Reset	Functions	Functions	Functions/ Halted*	Functions/ Halted*	Functions/ Halted*	Halted	Halted
OCRF	Reset	Functions	Held	Held	Functions	Held	Held	Held
TCRF	Reset	Functions	Held	Held	Functions	Held	Held	Held
TCSRF	Reset	Functions	Held	Held	Functions	Held	Held	Held

Note: * When $ø_W/4$ is selected as the TCF internal clock in active mode or sleep mode, since the system clock and internal clock are mutually asynchronous, synchronization is maintained by a synchronization circuit. This results in a maximum count cycle error of 1/ø (s). When the counter is operated in subactive mode, watch mode, or subsleep mode, $ø_W/4$ must be selected as the internal clock. The counter will not operate if any other internal clock is selected.

9.3.5 Application Notes

The following types of contention and operation can occur when timer F is used.

1. 16-bit timer mode

In toggle output, TMOFH pin output is toggled when all 16 bits match and a compare match signal is generated. If a TCRF write by a MOV instruction and generation of the compare match signal occur simultaneously, TOLH data is output to the TMOFH pin as a result of the TCRF write. TMOFL pin output is unstable in 16-bit mode, and should not be used; the TMOFL pin should be used as a port pin.

If an OCRFL write and compare match signal generation occur simultaneously, the compare match signal is invalid. However, if the written data and the counter value match, a compare match signal will be generated at that point. As the compare match signal is output in synchronization with the TCFL clock, a compare match will not result in compare match signal generation if the clock is stopped.

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

When TCF overflows, OVFH is set. OVFL is set if the setting conditions are satisfied when the lower 8 bits overflow. If a TCFL write and overflow signal output occur simultaneously, the overflow signal is not output.

2. 8-bit timer mode

a. TCFH, OCRFH

   In toggle output, TMOFH pin output is toggled when a compare match occurs. If a TCRF write by a MOV instruction and generation of the compare match signal occur simultaneously, TOLH data is output to the TMOFH pin as a result of the TCRF write.

   If an OCRFH write and compare match signal generation occur simultaneously, the compare match signal is invalid. However, if the written data and the counter value match, 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 and overflow signal output occur simultaneously, the overflow signal is not output.

b. TCFL, OCRFL

   In toggle output, TMOFL pin output is toggled when a compare match occurs. If a TCRF write by a MOV instruction and generation of the compare match signal occur simultaneously, TOLL data is output to the TMOFL pin as a result of the TCRF write.

If an OCRFL write and compare match signal generation occur simultaneously, the compare match signal is invalid. However, if the written data and the counter value match, a compare match signal will be generated at that point. As the compare match signal is output in synchronization with the TCFL clock, a compare match will not result in compare match signal generation if the clock is stopped.

If a TCFL write and overflow signal output occur simultaneously, the overflow signal is not output.

3. Clear timer FH, timer FL interrupt request flags (IRRTFH, IRRTFL), timer overflow flags H, L (OVFH, OVFL) and compare match flags H, L (CMFH, CMFL)

When øw/4 is selected as the internal clock, “Interrupt factor generation signal” will be operated with øw and the signal will be outputted with øw width. And, “Overflow signal” and “Compare match signal” are controlled with 2 cycles of øw signals. Those signals are outputted with 2 cycles width of øw (figure 9.6)

In active (high-speed, medium-speed) mode, even if you cleared interrupt request flag during the term of validity of “Interrupt factor generation signal”, same interrupt request flag is set. (figure 9.6 1) And, you cannot be cleared timer overflow flag and compare match flag during the term of validity of “Overflow signal” and “Compare match signal”.

For interrupt request flag is set right after interrupt request is cleared, interrupt process to one time timer FH, timer FL interrupt might be repeated. (figure 9.6 2) Therefore, to definitely clear interrupt request flag in active (high-speed, medium-speed) mode, clear should be processed after the time that calculated with below (1) formula. And, to definitely clear timer overflow flag and compare match flag, clear should be processed after read timer control status register F (TCSRF) after the time that calculated with below (1) formula. For ST of (1) formula, please substitute the longest number of execution states in used instruction. (10 states of RTE instruction when MULXU, DIVXU instruction is not used, 14 states when MULXU, DIVXU instruction is used) In subactive mode, there are not limitation for interrupt request flag, timer overflow flag, and compare match flag clear.

The term of validity of “Interrupt factor generation signal”

= 1 cycle of øw + waiting time for completion of executing instruction
+ interrupt time synchronized with ø = $1/øw + ST \times (1/ø) + (2/ø)$ (second).....(1)

ST: Executing number of execution states

Method 1 is recommended to operate for time efficiency.

Method 1

1. Prohibit interrupt in interrupt handling routine (set IENFH, IENFL to 0).
2. After program process returned normal handling, clear interrupt request flags (IRRTFH, IRRTFL) after more than that calculated with (1) formula.

3. After read timer control status register F (TCSRF), clear timer overflow flags (OVFH, OVFL) and compare match flags (CMFH, CMFL).
4. Operate interrupt permission (set IENFH, IENFL to 1).

Method 2

1. Set interrupt handling routine time to more than time that calculated with (1) formula.
2. Clear interrupt request flags (IRRTFH, IRRTFL) at the end of interrupt handling routine.
3. After read timer control status register F (TCSRF), clear timer overflow flags (OVFH, OVFL) and compare match flags (CMFH, CMFL).

All above attentions are also applied in 16-bit mode and 8-bit mode.

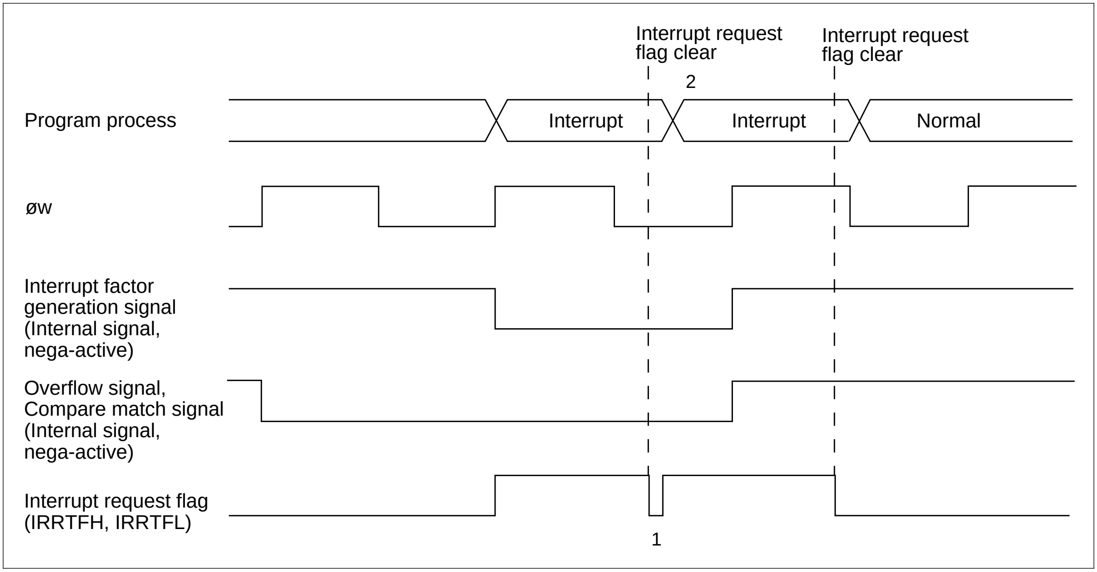


Figure 9.6 Clear Interrupt Request Flag when Interrupt Factor Generation Signal is Valid

4. Timer counter (TCF) read/write

When øw/4 is selected as the internal clock in active (high-speed, medium-speed) mode, write on TCF is impossible. And, when read TCF, as the system clock and internal clock are mutually asynchronous, TCF synchronizes with synchronization circuit. This results in a maximum TCF read value error of ±1.

When read/write TCF in active (high-speed, medium-speed) mode is needed, please select internal clock except for øw/4 before read/write.

In subactive mode, even øw/4 is selected as the internal clock, normal read/write TCF is possible.

9.4 Asynchronous Event Counter (AEC)

9.4.1 Overview

The asynchronous event counter is incremented by external event clock or internal clock input.

1. Features

Features of the asynchronous event counter are given below.

- Can count asynchronous events

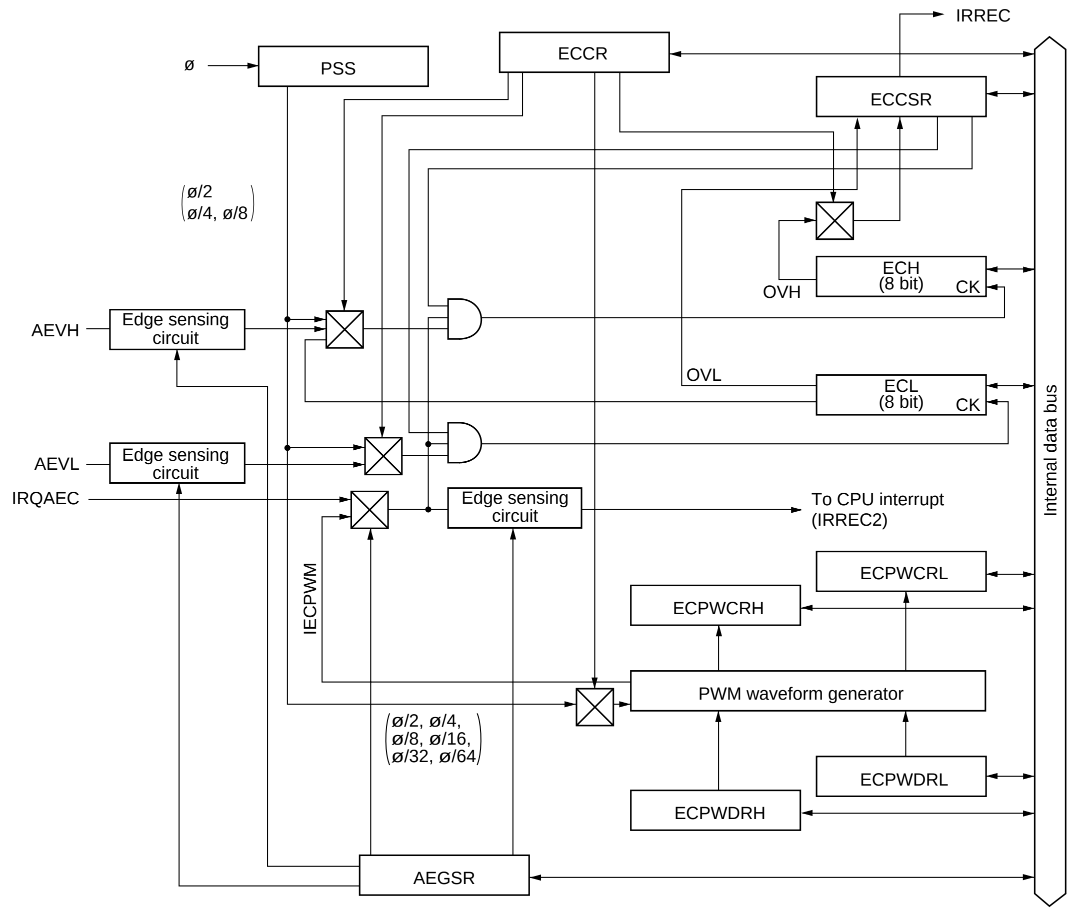
Can count external events input asynchronously without regard to the operation of base clocks ø and $ø_{SUB}$.

The counter has a 16-bit configuration, enabling it to count up to 65536 (2^{16}) events.

- Can also be used as two independent 8-bit event counter channels.
- Can be used as single-channel independent 16-bit event counter.
- Event/clock input is enabled only when IRQAEC is high or event counter PWM output (IECPWM) is high.
- Both edge sensing can be used for IRQAEC or event counter PWM output (IECPWM) interrupts. When the asynchronous counter is not used, independent interrupt function use is possible.
- When an event counter PWM is used, event clock input enabling/disabling can be performed automatically in a fixed cycle.
- External event input or a prescaler output clock can be selected by software for the ECH and ECL clock sources. ø/2, ø/4, or ø/8 can be selected as the prescaler output clock.
- Both edge counting is possible for AEVL and AEVH.
- Counter resetting and halting of the count-up function controllable by software
- Automatic interrupt generation on detection of event counter overflow
- Use of module standby mode enables this module to be placed in standby mode independently when not used.

2. Block diagram

Figure 9.7 shows a block diagram of the asynchronous event counter.



Notation

ECPWCRH: Event counter PWM compare register H
ECPWDRH: Event counter PWM data register H
AEGSR: Input pin edge selection register
ECCSR: Event counter control/status register
ECH: Event counter H
ECL: Event counter L
ECPWCRL: Event counter PWM compare register L
ECPWDRL: Event counter PWM data register L
ECCR: Event counter control register

Figure 9.7 Block Diagram of Asynchronous Event Counter

3. Pin configuration

Table 9.7 shows the asynchronous event counter pin configuration.

Table 9.7 Pin Configuration

Name	Abbrev.	I/O	Function
Asynchronous event input H	AEVH	Input	Event input pin for input to event counter H
Asynchronous event input L	AEVL	Input	Event input pin for input to event counter L
Event input enable interrupt input	IRQAEC	Input	Input pin for interrupt enabling event input

4. Register configuration

Table 9.8 shows the register configuration of the asynchronous event counter.

Table 9.8 Asynchronous Event Counter Registers

Name	Abbrev.	R/W	Initial Value	Address
Event counter PWM compare register H	ECPWCRH	R/W	H'FF	H'FF8C
Event counter PWM compare register L	ECPWCRL	R/W	H'FF	H'FF8D
Event counter PWM data register H	ECPWDRH	W	H'00	H'FF8E
Event counter PWM data register L	ECPWDRL	W	H'00	H'FF8F
Input pin edge selection register	AEGSR	R/W	H'00	H'FF92
Event counter control register	ECCR	R/W	H'00	H'FF94
Event counter control/status register	ECCSR	R/W	H'00	H'FF95
Event counter H	ECH	R	H'00	H'FF96
Event counter L	ECL	R	H'00	H'FF97
Clock stop register 2	CKSTP2	R/W	H'FF	H'FFFB

9.4.2 Register Configurations

1. Event counter PWM compare register H (ECPWCRH)

Bit	7	6	5	4	3	2	1	0
	ECPWCRH7	ECPWCRH6	ECPWCRH5	ECPWCRH4	ECPWCRH3	ECPWCRH2	ECPWCRH1	ECPWCRH0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: When ECPWME in AEGSR is 1, event counter PWM is operating and therefore ECPWCRH should not be modified.
When changing the conversion period, event counter PWM must be halted by clearing ECPWME to 0 in AEGSR before modifying ECPWCRH.

ECPWCRH is an 8-bit read/write register that sets the event counter PWM waveform conversion period.

2. Event counter PWM compare register L (ECPWCRL)

Bit	7	6	5	4	3	2	1	0
	ECPWCRL7	ECPWCRL6	ECPWCRL5	ECPWCRL4	ECPWCRL3	ECPWCRL2	ECPWCRL1	ECPWCRL0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: When ECPWME in AEGSR is 1, event counter PWM is operating and therefore ECPWCRL should not be modified.
When changing the conversion period, event counter PWM must be halted by clearing ECPWME to 0 in AEGSR before modifying ECPWCRL.

ECPWCRL is an 8-bit read/write register that sets the event counter PWM waveform conversion period.

3. Event counter PWM data register H (ECPWDRH)

Bit	7	6	5	4	3	2	1	0
	ECPWDRH7	ECPWDRH6	ECPWDRH5	ECPWDRH4	ECPWDRH3	ECPWDRH2	ECPWDRH1	ECPWDRH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Note: When ECPWME in AEGSR is 1, event counter PWM is operating and therefore ECPWDRH should not be modified.

When changing the data, event counter PWM must be halted by clearing ECPWME to 0 in AEGSR before modifying ECPWDRH.

ECPWDRH is an 8-bit write-only register that controls event counter PWM waveform generator data.

4. Event counter PWM data register L (ECPWDRL)

Bit	7	6	5	4	3	2	1	0
	ECPWDRL7	ECPWDRL6	ECPWDRL5	ECPWDRL4	ECPWDRL3	ECPWDRL2	ECPWDRL1	ECPWDRL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Note: When ECPWME in AEGSR is 1, event counter PWM is operating and therefore ECPWDRL should not be modified.

When changing the data, event counter PWM must be halted by clearing ECPWME to 0 in AEGSR before modifying ECPWDRL.

ECPWDRL is an 8-bit write-only register that controls event counter PWM waveform generator data.

5. Input pin edge selection register (AEGSR)

Bit	7	6	5	4	3	2	1	0
	AHEGS1	AHEGS0	ALEGS1	ALEGS0	AIEGS1	AIEGS0	ECPWME	—
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

AEGSR is an 8-bit read/write register that selects rising, falling, or both edge sensing for the AEVH, AEVL, and IRQAEC pins.

Bits 7 and 6: AEC edge select H

Bits 7 and 6 select rising, falling, or both edge sensing for the AEVH pin.

Bit 7 AHEGS1	Bit 6 AHEGS0	Description	
0	0	Falling edge on AEVH pin is sensed	(initial value)
	1	Rising edge on AEVH pin is sensed	
1	0	Both edges on AEVH pin are sensed	
	1	Use prohibited	

Bits 5 and 4: AEC edge select L

Bits 5 and 4 select rising, falling, or both edge sensing for the AEVL pin.

Bit 5 ALEGS1	Bit 4 ALEGS0	Description	
0	0	Falling edge on AEVL pin is sensed	(initial value)
	1	Rising edge on AEVL pin is sensed	
1	0	Both edges on AEVL pin are sensed	
	1	Use prohibited	

Bits 3 and 2: IRQAEC edge select

Bits 3 and 2 select rising, falling, or both edge sensing for the IRQAEC pin.

Bit 3 AIEGS1	Bit 2 AIEGS0	Description	
0	0	Falling edge on IRQAEC pin is sensed	(initial value)
	1	Rising edge on IRQAEC pin is sensed	
1	0	Both edges on IRQAEC pin are sensed	
	1	Use prohibited	

Bit 1: Event counter PWM enable

Bit 1 controls enabling/disabling of event counter PWM and selection/deselection of IRQAEC.

Bit 1 ECPWME	Description	
0	AEC PWM halted, IRQAEC selected	(initial value)
1	AEC PWM operation enabled, IRQAEC deselected	

Bit 0: Reserved bit

Bit 0 is a readable/writable reserved bit. It is initialized to 0 by a reset.

Note: Do not set this bit to 1.

6. Event counter control register (ECCR)

Bit	7	6	5	4	3	2	1	0
	ACKH1	ACKH0	ACKL1	ACKL0	PWCK2	PWCK1	PWCK0	—
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

ECCR performs counter input clock and IRQAEC/IECPWM control.

Bits 7 and 6: AEC clock select H (ACKH1, ACKH0)

Bits 7 and 6 select the clock used by ECH.

Bit 7 ACKH1	Bit 6 ACKH0	Description	
0	0	AEVH pin input	(initial value)
	1	ø/2	
1	0	ø/4	
	1	ø/8	

Bits 5 and 4: AEC clock select L (ACKL1, ACKL0)

Bits 5 and 4 select the clock used by ECL.

Bit 5 ACKL1	Bit 4 ACKL0	Description	
0	0	AEVL pin input	(initial value)
	1	ø/2	
1	0	ø/4	
	1	ø/8	

Bits 3 to 1: Event counter PWM clock select (PWCK2, PWCK1, PWCK0)

Bits 3 to 1 select the event counter PWM clock.

Bit 3 PWCK2	Bit 2 PWCK1	Bit 1 PWCK0	Description	
0	0	0	ø/2	(initial value)
		1	ø/4	
	1	0	ø/8	
		1	ø/16	
1	*	0	ø/32	
		1	ø/64	

*: Don’t care

Bit 0: Reserved bit

Bit 0 is a readable/writable reserved bit. It is initialized to 0 by a reset.

Note: Do not set this bit to 1.

7. Event counter control/status register (ECCSR)

Bit	7	6	5	4	3	2	1	0
	OVH	OVL	—	CH2	CUEH	CUEL	CRCH	CRCL
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: * Bits 7 and 6 can only be written with 0, for flag clearing.

ECCSR is an 8-bit read/write register that controls counter overflow detection, counter resetting, and halting of the count-up function.

ECCSR is initialized to H'00 upon reset.

Bit 7: Counter overflow flag H (OVH)

Bit 7 is a status flag indicating that ECH has overflowed from H'FF to H'00. This flag is set when ECH overflows. It is cleared by software but cannot be set by software. OVH is cleared by reading it when set to 1, then writing 0.

When ECH and ECL are used as a 16-bit event counter with CH2 cleared to 0, OVH functions as a status flag indicating that the 16-bit event counter has overflowed from H'FFFF to H'0000.

Bit 7 OVH	Description	
0	ECH has not overflowed Clearing conditions: After reading OVH = 1, cleared by writing 0 to OVH	(initial value)
1	ECH has overflowed Setting conditions: Set when ECH overflows from H’FF to H’00	

Bit 6: Counter overflow flag L (OVL)

Bit 6 is a status flag indicating that ECL has overflowed from H'FF to H'00. This flag is set when ECL overflows. It is cleared by software but cannot be set by software. OVL is cleared by reading it when set to 1, then writing 0.

Bit 6 OVL	Description	
0	ECL has not overflowed Clearing conditions: After reading OVL = 1, cleared by writing 0 to OVL	(initial value)
1	ECL has overflowed Setting conditions: Set when ECL overflows from H'FF to H'00 while CH2 is set to 1	

Bit 5: Reserved bit

Bit 5 is reserved; it can be read and written, and is initialized to 0 upon reset.

Bit 4: Channel select (CH2)

Bit 4 selects whether ECH and ECL are used as a single-channel 16-bit event counter or as two independent 8-bit event counter channels. When CH2 is cleared to 0, ECH and ECL function as a 16-bit event counter which is incremented each time an event clock is input to the AEVL pin. In this case, the overflow signal from ECL is selected as the ECH input clock. When CH2 is set to 1, ECH and ECL function as independent 8-bit event counters which are incremented each time an event clock is input to the AEVH or AEVL pin, respectively.

Bit 4 CH2	Description
0	ECH and ECL are used together as a single-channel 16-bit event counter (initial value)
1	ECH and ECL are used as two independent 8-bit event counter channels

Bit 3: Count-up enable H (CUEH)

Bit 3 enables event clock input to ECH. When 1 is written to this bit, event clock input is enabled and increments the counter. When 0 is written to this bit, event clock input is disabled and the ECH value is held. The AEVH pin or the ECL overflow signal can be selected as the event clock source by bit CH2.

Bit 3 CUEH	Description
0	ECH event clock input is disabled (initial value) ECH value is held
1	ECH event clock input is enabled

Bit 2: Count-up enable L (CUEL)

Bit 3 enables event clock input to ECL. When 1 is written to this bit, event clock input is enabled and increments the counter. When 0 is written to this bit, event clock input is disabled and the ECL value is held.

Bit 2 CUEL	Description
0	ECL event clock input is disabled (initial value) ECL value is held
1	ECL event clock input is enabled

Bit 1: Counter reset control H (CRCH)

Bit 1 controls resetting of ECH. When this bit is cleared to 0, ECH is reset. When 1 is written to this bit, the counter reset is cleared and the ECH count-up function is enabled.

Bit 1 CRCH	Description	
0	ECH is reset	(initial value)
1	ECH reset is cleared and count-up function is enabled	

Bit 0: Counter reset control L (CRCL)

Bit 0 controls resetting of ECL. When this bit is cleared to 0, ECL is reset. When 1 is written to this bit, the counter reset is cleared and the ECL count-up function is enabled.

Bit 0 CRCL	Description	
0	ECL is reset	(initial value)
1	ECL reset is cleared and count-up function is enabled	

8. Event counter H (ECH)

Bit	7	6	5	4	3	2	1	0
	ECH7	ECH6	ECH5	ECH4	ECH3	ECH2	ECH1	ECH0
Initial Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ECH is an 8-bit read-only up-counter that operates either as an independent 8-bit event counter or as the upper 8-bit up-counter of a 16-bit event counter configured in combination with ECL. Either the external asynchronous event AEVH pin or the overflow signal from lower 8-bit counter ECL can be selected as the input clock source. ECH can be cleared to H'00 by software, and is also initialized to H'00 upon reset.

9. Event counter L (ECL)

ECL is an 8-bit read-only up-counter that operates either as an independent 8-bit event counter or as the lower 8-bit up-counter of a 16-bit event counter configured in combination with ECH. The event clock from the external asynchronous event AEVL pin, or ø/2, ø/4, or ø/8, is used as the input clock source. ECL can be cleared to H'00 by software, and is also initialized to H'00 upon reset.

Bit	7	6	5	4	3	2	1	0
	ECL7	ECL6	ECL5	ECL4	ECL3	ECL2	ECL1	ECL0
Initial Value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

10. Clock stop register 2 (CKSTPR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	PW2CKSTP	AECKSTP	—	PW1CKSTP	LDCKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	R/W	R/W	—	R/W	R/W

CKSTPR2 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bit relating to the asynchronous event counter is described here. For details of the other bits, see the sections on the relevant modules.

Bit 3: Asynchronous event counter module standby mode control (AECKSTP)

Bit 3 controls setting and clearing of module standby mode for the asynchronous event counter.

AECKSTP	Description	
0	Asynchronous event counter is set to module standby mode	
1	Asynchronous event counter module standby mode is cleared	(initial value)

9.4.3 Operation

1. 16-bit event counter operation

When bit CH2 is cleared to 0 in ECCSR, ECH and ECL operate as a 16-bit event counter.

Any of four input clock sources—ø/2, ø/4, ø/8, or AEVL pin input—can be selected by means of bits ACKL1 and ACKL0 in ECCR.

When AEVL pin input is selected, input sensing is selected with bits ALEGS1 and ALEGS0.

The input clock is enabled only when IRQAEC is high or IECPWM is high. When IRQAEC is low or IECPWM is low, the input clock is not input to the counter, which therefore does not operate. Figure 9.8 shows an example of the software processing when ECH and ECL are used as a 16-bit event counter.

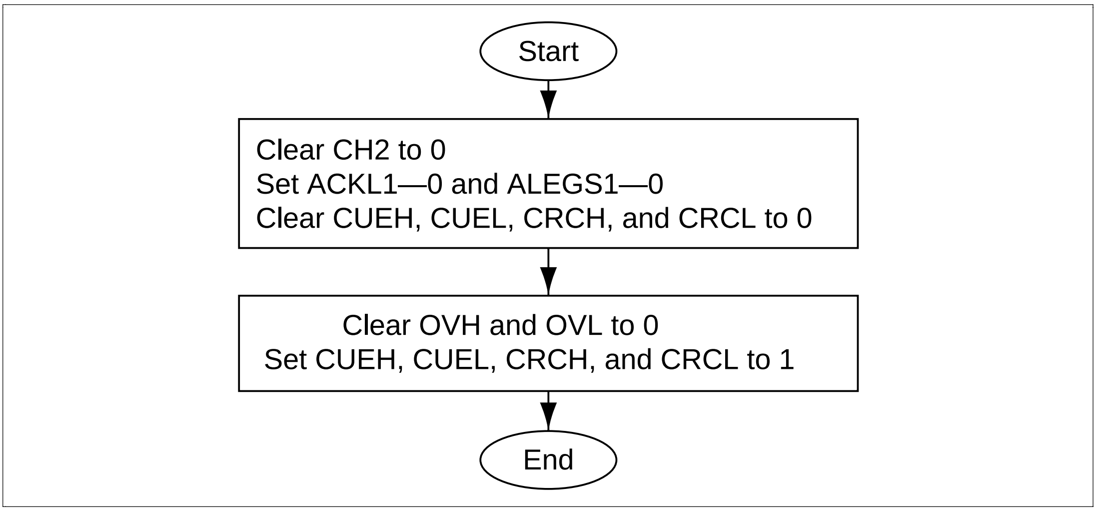


Figure 9.8 Example of Software Processing when Using ECH and ECL as 16-Bit Event Counter

As CH2 is cleared to 0 by a reset, ECH and ECL operate as a 16-bit event counter after a reset, and as ACKL1 and ACKL0 are cleared to 00, the operating clock is asynchronous event input from the AEVL pin (using falling edge sensing). When the next clock is input after the count value reaches H'FF in both ECH and ECL, ECH and ECL overflow from H'FFFF to H'0000, the OVH flag is set to 1 in ECCSR, the ECH and ECL count values each return to H'00, and counting up is restarted. When overflow occurs, the IRREC bit is set to 1 in IRR2. If the IENEC bit in IENR2 is 1 at this time, an interrupt request is sent to the CPU.

2. 8-bit event counter operation

When bit CH2 is set to 1 in ECCSR, ECH and ECL operate as independent 8-bit event counters.

ø/2, ø/4, ø/8, or AEVH pin input can be selected as the input clock source for ECH by means of bits ACKH1 and ACKH0 in ECCR, and ø/2, ø/4, ø/8, or AEVL pin input can be selected as the input clock source for ECL by means of bits ACKL1 and ACKL0 in ECCR.

Input sensing is selected with bits AHEGS1 and AHEGS0 when AEVH pin input is selected, and with bits ALEGS1 and ALEGS0 when AEVL pin input is selected.

The input clock is enabled only when IRQAEC is high or IECPWM is high. When IRQAEC is low or IECPWM is low, the input clock is not input to the counter, which therefore does not operate. Figure 9.9 shows an example of the software processing when ECH and ECL are used as 8-bit event counters.

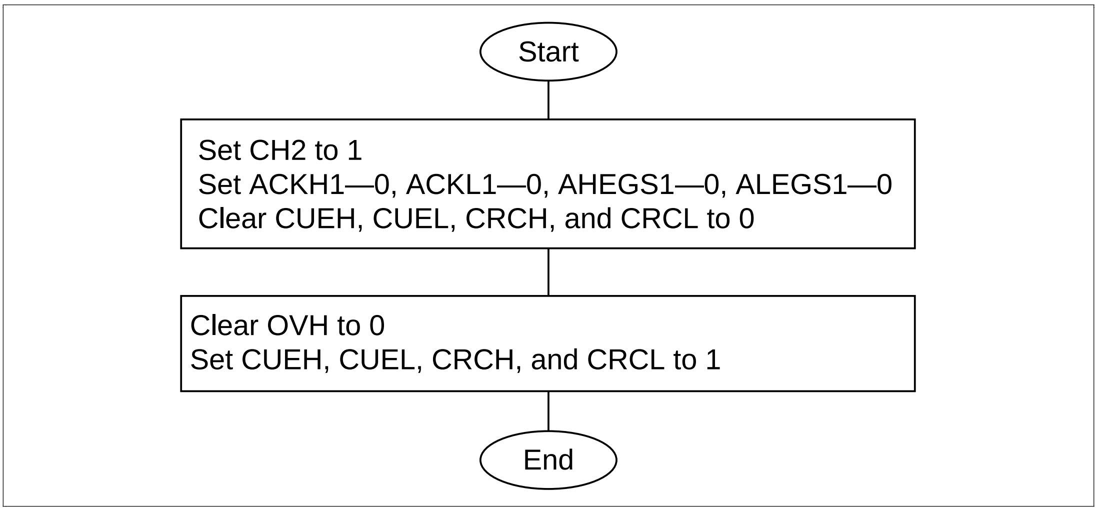


Figure 9.9 Example of Software Processing when Using ECH and ECL as 8-Bit Event Counters

ECH and ECL can be used as 8-bit event counters by carrying out the software processing shown in the example in figure 9.9. When the next clock is input after the ECH count value reaches H'FF, ECH overflows, the OVH flag is set to 1 in ECCSR, the ECH count value returns to H'00, and counting up is restarted. Similarly, when the next clock is input after the ECL count value reaches H'FF, ECL overflows, the OVL flag is set to 1 in ECCSR, the ECL count value returns to H'00, and counting up is restarted. When overflow occurs, the IRREC bit is set to 1 in IRR2. If the IENEC bit in IENR2 is 1 at this time, an interrupt request is sent to the CPU.

3. IRQAEC operation

When ECPWME in AEGSR is 0, the ECH and ECL input clocks are enabled only when IRQAEC is high. When IRQAEC is low, the input clocks are not input to the counters, and so ECH and ECL do not count. ECH and ECL count operations can therefore be controlled from outside by controlling IRQAEC. In this case, ECH and ECL cannot be controlled individually.

IRQAEC can also operate as an interrupt source. In this case the vector number is 6 and the vector addresses are H'000C and H'000D.

Interrupt enabling is controlled by IENEC2 in IENR1. When an IRQAEC interrupt is generated, IRR1 interrupt request flag IRREC2 is set to 1. If IENEC2 in IENR1 is set to 1 at this time, an interrupt request is sent to the CPU.

Rising, falling, or both edge sensing can be selected for the IRQAEC input pin with bits AIAGS1 and AIAGS0 in AEGSR.

4. Event counter PWM operation

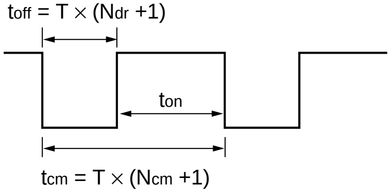
When ECPWME in AEGSR is 1, the ECH and ECL input clocks are enabled only when event counter PWM output (IECPWM) is high. When IECPWM is low, the input clocks are not input to the counters, and so ECH and ECL do not count. ECH and ECL count operations can therefore be controlled cyclically from outside by controlling event counter PWM. In this case, ECH and ECL cannot be controlled individually.

IECPWM can also operate as an interrupt source. In this case the vector number is 6 and the vector addresses are H'000C and H'000D.

Interrupt enabling is controlled by IENEC2 in IENR1. When an IECPWM interrupt is generated, IRR1 interrupt request flag IRREC2 is set to 1. If IENEC2 in IENR1 is set to 1 at this time, an interrupt request is sent to the CPU.

Rising, falling, or both edge detection can be selected for IECPWM interrupt sensing with bits AIAGS1 and AIAGS0 in AEGSR.

Figure 9.10 and table 9.9 show examples of event counter PWM operation.



T_{on} : Clock input enabled time
T_{off} : Clock input disabled time
T_{cm} : One conversion period
T : ECPWM input clock cycle
N_{dr} : Value of ECPWDRH and ECPWDRL
Fixed high when Ndr = H'FFFF
N_{cm} : Value of ECPWCRH and ECPWCRL

Figure 9.10 Event Counter Operation Waveform

Note: N_{dr} and N_{cm} above must be set so that $N_{dr} < N_{cm}$. If the settings do not satisfy this condition, do not set ECPWME to 1 in AEGSR.

Table 9.9 Examples of Event Counter PWM Operation

Conditions: f_{osc} = 4 MHz, $f_ø$ = 2 MHz, high-speed active mode, ECPWCR value (N_{cm}) = H'7A11, ECPWDR value (N_{dr}) = H'16E3

Clock Source Selection	Clock Source Cycle (T)*	ECPWMCR Value (N_{cm})	ECPWMDR Value (N_{dr})	t_{off} = T × (N_{dr} + 1)	t_{cm} = T × (N_{cm} + 1)	$t_{on} = t_{cm} - t_{off}$
ø/2	1 µs	H'7A11 D'31249	H'16E3 D'5859	5.86 ms	31.25 ms	25.39 ms
ø/4	2 µs			11.72 ms	62.5 ms	50.78 ms
ø/8	4 µs			23.44 ms	125.0 ms	101.56 ms
ø/16	8 µs			46.88 ms	250.0 ms	203.12 ms
ø/32	16 µs			93.76 ms	500.0 ms	406.24 ms
ø/64	32 µs			187.52 ms	1000.0 ms	812.48 ms

Note: * t_{off} minimum width

5. Clock Input Enable/Disable Function Operation

The clock input to the event counter can be controlled by the IRQAEC pin when ECPWME in AEGSR is 0, and by event counter PWM output IECPWM when ECPWME in AEGSR is 1. As this function forcibly terminates the clock input by each signal, a maximum error of one count will occur depending the IRQAEC or IECPWM timing.

Figure 9.11 shows an example of the operation of this function.

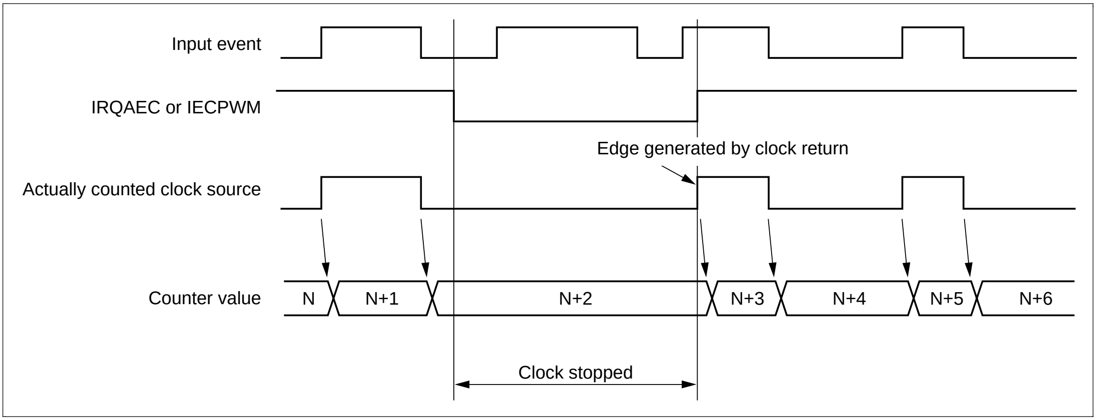


Figure 9.11 Example of Clock Control Operation

9.4.4 Asynchronous Event Counter Operation Modes

Asynchronous event counter operation modes are shown in table 9.10.

Table 9.10 Asynchronous Event Counter Operation Modes

Operation Mode	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby	Module Standby
AEGSR	Reset	Functions	Functions	Held*1	Functions	Functions	Held*1	Held
ECCR	Reset	Functions	Functions	Held*1	Functions	Functions	Held*1	Held
ECCSR	Reset	Functions	Functions	Held*1	Functions	Functions	Held*1	Held
ECH	Reset	Functions	Functions	Functions*1*2	Functions*2	Functions*2	Functions*1*2	Halted
ECL	Reset	Functions	Functions	Functions*1*2	Functions*2	Functions*2	Functions*1*2	Halted
IEQAEC	Reset	Functions	Functions	Held*3	Functions	Functions	Held*3	Held*4
Event counter PWM	Reset	Functions	Functions	Held	Held	Held	Held	Held

Notes: 1. When an asynchronous external event is input, the counter increments but the counter overflow H/L flags are not affected.

2. Operates when asynchronous mode external events are selected; halted and retained otherwise.
3. Clock control by IRQAEC operates, but interrupts do not.
4. As the clock is stopped in module standby mode, IRQAEC has no effect.

9.4.5 Application Notes

1. When reading the values in ECH and ECL, first clear bits CUEH and CUEL to 0 in ECCSR to prevent asynchronous event input to the counter. The correct value will not be returned if the event counter increments while being read.
2. Use a clock with a frequency of up to 16 MHz for input to the AEVH and AEVL pins, and ensure that the high and low widths of the clock are at least 30 ns. The duty cycle is immaterial.

Mode		Maximum AEVH/AEVL Pin Input Clock Frequency
Active (high-speed), sleep (high-speed)		16 MHz
Active (medium-speed), sleep (medium-speed)	(ø/16)	$2 \cdot f_{OSC}$
	(ø/32)	f_{OSC}
	(ø/64)	$1/2 \cdot f_{OSC}$
f_{OSC} = 1 MHz to 4 MHz	(ø/128)	$1/4 \cdot f_{OSC}$
Watch, subactive, subsleep, standby	(øw/2)	1000 kHz
	(øw/4)	500 kHz
øw = 32.768 kHz or 38.4 kHz	(øw/8)	250 kHz

3. When AEC uses with 16-bit mode, set CUEH in ECCSR to “1” first, set CRCH in ECCSR to “1” second, or set both CUEH and CRCH to “1” at same time before clock entry. While AEC is operating on 16-bit mode, do not change CUEH. Otherwise, ECH will be miscounted up.
4. When ECPWME in AEGSR is 1, event counter PWM is operating and therefore ECPWCRH, ECPWCRL, ECPWDRH, and ECPWDRL should not be modified.

   When changing the data, event counter PWM must be halted by clearing ECPWME to 0 in AEGSR before modifying these registers.
5. The event counter PWM data register and event counter PWM compare register must be set so that event counter PWM data register < event counter PWM compare register. If the settings do not satisfy this condition, do not set ECPWME to 1 in AEGSR.
6. As synchronization is established internally when an IRQAEC interrupt is generated, a maximum error of 1 tcyc will occur between clock halting and interrupt acceptance.

Section 10 Serial Communication Interface

10.1 Overview

The H8/3802 Series is provided with one serial communication interface, SCI3.

Serial communication interface 3 (SCI3) can carry out serial data communication in either asynchronous or synchronous mode. It is also provided with a multiprocessor communication function that enables serial data to be transferred among processors.

10.1.1 Features

Features of SCI3 are listed below.

- Choice of asynchronous or synchronous mode for serial data communication
  - Asynchronous mode

    Serial data communication is performed asynchronously, with synchronization provided character by character. In this mode, serial data can be exchanged with standard asynchronous communication LSIs such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A multiprocessor communication function is also provided, enabling serial data communication among processors.

    There is a choice of 16 data transfer formats.

Data length	7, 8, 5 bits
Stop bit length	1 or 2 bits
Parity	Even, odd, or none
Multiprocessor bit	1 or 0
Receive error detection	Parity, overrun, and framing errors
Break detection	Break detected by reading the RXD_{32} pin level directly when a framing error occurs

— Synchronous mode

Serial data communication is synchronized with a clock. In his mode, serial data can be exchanged with another LSI that has a synchronous communication function.

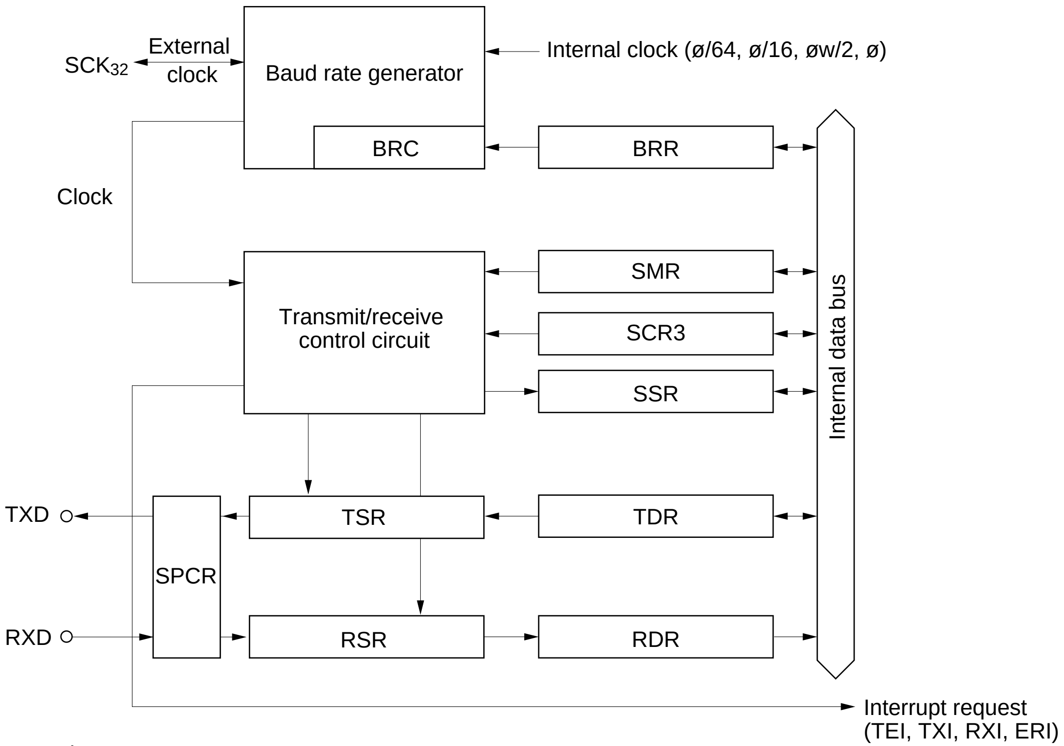
Data length	8 bits
Receive error detection	Overrun errors

- Full-duplex communication

  Separate transmission and reception units are provided, enabling transmission and reception to be carried out simultaneously. The transmission and reception units are both double-buffered, allowing continuous transmission and reception.
- On-chip baud rate generator, allowing any desired bit rate to be selected
- Choice of an internal or external clock as the transmit/receive clock source
- Six interrupt sources: transmit end, transmit data empty, receive data full, overrun error, framing error, and parity error

10.1.2 Block diagram

Figure 10.1 shows a block diagram of SCI3.



Notation:

RSR: Receive shift register
RDR: Receive data register
TSR: Transmit shift register
TDR: Transmit data register
SMR: Serial mode register
SCR3: Serial control register 3
SSR: Serial status register
BRR: Bit rate register
BRC: Bit rate counter
SPCR: Serial port control register

Figure 10.1 SCI3 Block Diagram

10.1.3 Pin configuration

Table 10.1 shows the SCI3 pin configuration.

Table 10.1 Pin Configuration

Name	Abbrev.	I/O	Function
SCI3 clock	SCK_{32}	I/O	SCI3 clock input/output
SCI3 receive data input	RXD_{32}	Input	SCI3 receive data input
SCI3 transmit data output	TXD_{32}	Output	SCI3 transmit data output

10.1.4 Register configuration

Table 10.2 shows the SCI3 register configuration.

Table 10.2 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 data register	SSR	R/W	H'84	H'FFAC
Receive data register	RDR	R	H'00	H'FFAD
Transmit shift register	TSR	Protected	—	—
Receive shift register	RSR	Protected	—	—
Bit rate counter	BRC	Protected	—	—
Clock stop register 1	CKSTPR1	R/W	H'FF	H'FFFA
Serial port control register	SPCR	R/W	—	H'FF91

10.2 Register Descriptions

10.2.1 Receive shift register (RSR)

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

RSR is a register used to receive serial data. Serial data input to RSR from the RXD_{32} pin is set in the order in which it is received, starting from the LSB (bit 0), and converted to parallel data. When one byte of data is received, it is transferred to RDR automatically.

RSR cannot be read or written directly by the CPU.

10.2.2 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

RDR is an 8-bit register that stores received serial data.

When reception of one byte of data is finished, the received data is transferred from RSR to RDR, and the receive operation is completed. RSR is then able to receive data. RSR and RDR are double-buffered, allowing consecutive receive operations.

RDR is a read-only register, and cannot be written by the CPU.

RDR is initialized to H'00 upon reset, and in standby, module standby or watch mode.

10.2.3 Transmit shift register (TSR)

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

TSR is a register used to transmit serial data. Transmit data is first transferred from TDR to TSR, and serial data transmission is carried out by sending the data to the TXD_{32} pin in order, starting from the LSB (bit 0). When one byte of data is transmitted, the next byte of transmit data is transferred to TDR, and transmission started, automatically. Data transfer from TDR to TSR is not performed if no data has been written to TDR (if bit TDRE is set to 1 in the serial status register (SSR)).

TSR cannot be read or written directly by the CPU.

10.2.4 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

TDR is an 8-bit register that stores transmit data. When TSR is found to be empty, the transmit data written in TDR is transferred to TSR, and serial data transmission is started. Continuous transmission is possible by writing the next transmit data to TDR during TSR serial data transmission.

TDR can be read or written by the CPU at any time.

TDR is initialized to H'FF upon reset, and in standby, module standby, or watch mode.

10.2.5 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

SMR is an 8-bit register used to set the serial data transfer format and to select the clock source for the baud rate generator.

SMR can be read or written by the CPU at any time.

SMR is initialized to H'00 upon reset, and in standby, module standby, or watch mode.

Bit 7: Communication mode (COM)

Bit 7 selects whether SCI3 operates in asynchronous mode or synchronous mode.

Bit 7 COM	Description	
0	Asynchronous mode	(initial value)
1	Synchronous mode	

Bit 6: Character length (CHR)

Bit 6 selects either 7 or 8 bits as the data length to be used in asynchronous mode. In synchronous mode the data length is always 8 bits, irrespective of the bit 6 setting.

Bit 6 CHR	Description	
0	8-bit data/5-bit data*2	(initial value)
1	7-bit data*1/5-bit data*2	

Notes: 1. When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted.

2. When 5-bit data is selected, set both PE and MP to 1. The three most significant bits (bits 7, 6, and 5) of TDR are not transmitted.

Bit 5: Parity enable (PE)

Bit 5 selects whether a parity bit is to be added during transmission and checked during reception in asynchronous mode. In synchronous mode parity bit addition and checking is not performed, irrespective of the bit 5 setting.

Bit 5 PE	Description	
0	Parity bit addition and checking disabled*2	(initial value)
1	Parity bit addition and checking enabled*1/*2	

Notes: 1. When PE is set to 1, even or odd parity, as designated by bit PM, is added to transmit data before it is sent, and the received parity bit is checked against the parity designated by bit PM.

2. For the case where 5-bit data is selected, see table 10.11.

Bit 4: Parity mode (PM)

Bit 4 selects whether even or odd parity is to be used for parity addition and checking. The PM bit setting is only valid in asynchronous mode when bit PE is set to 1, enabling parity bit addition and checking. The PM bit setting is invalid in synchronous mode, and in asynchronous mode if parity bit addition and checking is disabled.

Bit 4 PM	Description	
0	Even parity*1	(initial value)
1	Odd parity*2	

Notes: 1. When even parity is selected, a parity bit is added in transmission so that the total number of 1 bits in the transmit data plus the parity bit is an even number; in reception, a check is carried out to confirm that the number of 1 bits in the receive data plus the parity bit is an even number.

2. When odd parity is selected, a parity bit is added in transmission so that the total number of 1 bits in the transmit data plus the parity bit is an odd number; in reception, a check is carried out to confirm that the number of 1 bits in the receive data plus the parity bit 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. The STOP bit setting is only valid in asynchronous mode. When synchronous mode is selected the STOP bit setting is invalid since stop bits are not added.

Bit 3 STOP	Description	
0	1 stop bit*1	(initial value)
1	2 stop bits*2	

Notes: 1. In transmission, a single 1 bit (stop bit) is added at the end of a transmit character.
2. In transmission, two 1 bits (stop bits) are added at the end of a transmit character.

In reception, only the first of the received stop bits is checked, irrespective of the STOP bit setting. If the second stop bit is 1 it is treated as a stop bit, but if 0, it is treated as the start bit of the next transmit character.

Bit 2: Multiprocessor mode (MP)

Bit 2 enables or disables the multiprocessor communication function. When the multiprocessor communication function is enabled, the parity settings in the PE and PM bits are invalid. The MP bit setting is only valid in asynchronous mode. When synchronous mode is selected the MP bit should be set to 0. For details on the multiprocessor communication function, see 10.3.4, Multiprocessor Communication Function.

Bit 2 MP	Description	
0	Multiprocessor communication function disabled*	(initial value)
1	Multiprocessor communication function enabled*	

Note: * For the case where 5-bit data is selected, see table 10.11.

Bits 1 and 0: Clock select 1, 0 (CKS1, CKS0)

Bits 1 and 0 choose ø/64, ø/16, øw/2, or ø as the clock source for the baud rate generator.

For the relation between the clock source, bit rate register setting, and baud rate, see 8, Bit rate register (BRR).

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	ø clock	(initial value)
0	1	ø w/2 clock*1/ø w clock*2	
1	0	ø/16 clock	
1	1	ø/64 clock	

Notes: 1. ø w/2 clock in active (medium-speed/high-speed) mode and sleep mode
2. ø w clock in subactive mode and subsleep mode
3. In subactive or subsleep mode, SCI3 can be operated when CPU clock is øw/2 only.

10.2.6 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

SCR3 is an 8-bit register for selecting transmit or receive operation, the asynchronous mode clock output, interrupt request enabling or disabling, and the transmit/receive clock source.

SCR3 can be read or written by the CPU at any time.

SCR3 is initialized to H'00 upon reset, and in standby, module standby or watch mode.

Bit 7: Transmit interrupt enable (TIE)

Bit 7 selects enabling or disabling of the transmit data empty interrupt request (TXI) when transmit data is transferred from the transmit data register (TDR) to the transmit shift register (TSR), and bit TDRE in the serial status register (SSR) is set to 1.

TXI can be released by clearing bit TDRE or 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 selects enabling or disabling of the receive data full interrupt request (RXI) and the receive error interrupt request (ERI) when receive data is transferred from the receive shift register (RSR) to the receive data register (RDR), and bit RDRF in the serial status register (SSR) is set to 1. There are three kinds of receive error: overrun, framing, and parity.

RXI and ERI can be released by clearing bit RDRF or the FER, PER, or OER error flag 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 selects enabling or disabling of the start of transmit operation.

Bit 5 TE	Description	
0	Transmit operation disabled*1 (TXD_{32} pin is transmit data pin)	(initial value)
1	Transmit operation enabled*2 (TXD_{32} pin is transmit data pin)	

Notes: 1. Bit TDRE in SSR is fixed at 1.

2. When transmit data is written to TDR in this state, bit TDR in SSR is cleared to 0 and serial data transmission is started. Be sure to carry out serial mode register (SMR) settings, and setting of bit SPC32 in SPCR, to decide the transmission format before setting bit TE to 1.

Bit 4: Receive enable (RE)

Bit 4 selects enabling or disabling of the start of receive operation.

Bit 4 RE	Description	
0	Receive operation disabled*1 (RXD pin is I/O port)	(initial value)
1	Receive operation enabled*2 (RXD pin is receive data pin)	

Notes: 1. Note that the RDRF, FER, PER, and OER flags in SSR are not affected when bit RE is cleared to 0, and retain their previous state.

2. In this state, serial data reception is started when a start bit is detected in asynchronous mode or serial clock input is detected in synchronous mode. Be sure to carry out serial mode register (SMR) settings to decide the reception format before setting bit RE to 1.

Bit 3: Multiprocessor interrupt enable (MPIE)

Bit 3 selects enabling or disabling of the multiprocessor interrupt request. The MPIE bit setting is only valid when asynchronous mode is selected and reception is carried out with bit MP in SMR set to 1. The MPIE bit setting is invalid when bit COM is set to 1 or bit MP is cleared to 0.

Bit 3 MPIE	Description	
0	Multiprocessor interrupt request disabled (normal receive operation) Clearing conditions: When data is received in which the multiprocessor bit is set to 1	(initial value)
1	Multiprocessor interrupt request enabled*	

Note: * Receive data transfer from RSR to RDR, receive error detection, and setting of the RDRF, FER, and OER status flags in SSR is not performed. RXI, ERI, and setting of the RDRF, FER, and OER flags in SSR, are disabled until data with the multiprocessor bit set to 1 is received. When a receive character with the multiprocessor bit set to 1 is received, bit MPBR in SSR is set to 1, bit MPIE is automatically cleared to 0, and RXI and ERI requests (when bits TIE and RIE in serial control register 3 (SCR3) are set to 1) and setting of the RDRF, FER, and OER flags are enabled.

Bit 2: Transmit end interrupt enable (TEIE)

Bit 2 selects enabling or disabling of the transmit end interrupt request (TEI) if there is no valid transmit data in TDR when MSB data is to be sent.

Bit 2 TEIE	Description	
0	Transmit end interrupt request (TEI) disabled	(initial value)
1	Transmit end interrupt request (TEI) enabled*	

Note: * TEI can be released by clearing bit TDRE to 0 and clearing bit TEND to 0 in SSR, or by clearing bit TEIE to 0.

Bits 1 and 0: Clock enable 1 and 0 (CKE1, CKE0)

Bits 1 and 0 select the clock source and enabling or disabling of clock output from the SCK_{32} pin. The combination of CKE1 and CKE0 determines whether the SCK_{32} pin functions as an I/O port, a clock output pin, or a clock input pin.

The CKE0 bit setting is only valid in case of internal clock operation (CKE1 = 0) in asynchronous mode. In synchronous mode, or when external clock operation is used (CKE1 = 1), bit CKE0 should be cleared to 0.

After setting bits CKE1 and CKE0, set the operating mode in the serial mode register (SMR).

For details on clock source selection, see table 10.9 in 10.3.1.

Bit 1	Bit 0	Description		
CKE1	CKE0	Communication Mode	Clock Source	SCK_{32} Pin Function
0	0	Asynchronous	Internal clock	I/O port*1
		Synchronous	Internal clock	Serial clock output*1
0	1	Asynchronous	Internal clock	Clock output*2
		Synchronous	Reserved	
1	0	Asynchronous	External clock	Clock input*3
		Synchronous	External clock	Serial clock input
1	1	Asynchronous	Reserved	
		Synchronous	Reserved	

Notes: 1. Initial value

2. A clock with the same frequency as the bit rate is output.

3. Input a clock with a frequency 16 times the bit rate.

10.2.7 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.

SSR is an 8-bit register containing status flags that indicate the operational status of SCI3, and multiprocessor bits.

SSR can be read or written to by the CPU at any time, but 1 cannot be written to bits TDRE, RDRF, OER, PER, and FER.

Bits TEND and MPBR are read-only bits, and cannot be modified.

SSR is initialized to H'84 upon reset, and in standby, module standby, or watch mode.

Bit 7: Transmit data register empty (TDRE)

Bit 7 indicates that transmit data has been transferred from TDR to TSR.

Bit 7 TDRE	Description	
0	Transmit data written in 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	Transmit data has not been written to TDR, or transmit data written in TDR has been transferred to TSR Setting conditions: When bit TE in SCR3 is cleared to 0 When data is transferred from TDR to TSR	(initial value)

Bit 6: Receive data register full (RDRF)

Bit 6 indicates that received data is stored in RDR.

Bit 6 RDRF	Description	
0	There is no receive data in RDR Clearing conditions: After reading RDRF = 1, cleared by writing 0 to RDRF When RDR data is read by an instruction	(initial value)
1	There is receive data in RDR Setting conditions: When reception ends normally and receive data is transferred from RSR to RDR	

Note: If an error is detected in the receive data, or if the RE bit in SCR3 has been cleared to 0, RDR and bit RDRF are not affected and retain their previous state.
Note that if data reception is completed while bit RDRF is still set to 1, an overrun error (OER) will result and the receive data will be lost.

Bit 5: Overrun error (OER)

Bit 5 indicates that an overrun error has occurred during reception.

Bit 5 OER	Description	
0	Reception in progress or completed*1 Clearing conditions: After reading OER = 1, cleared by writing 0 to OER	(initial value)
1	An overrun error has occurred during reception*2 Setting conditions: When reception is completed with RDRF set to 1	

Notes: 1. When bit RE in SCR3 is cleared to 0, bit OER is not affected and retains its previous state.
2. RDR retains the receive data it held before the overrun error occurred, and data received after the error is lost. Reception cannot be continued with bit OER set to 1, and in synchronous mode, transmission cannot be continued either.

Bit 4: Framing error (FER)

Bit 4 indicates that a framing error has occurred during reception in asynchronous mode.

Bit 4 FER	Description	
0	Reception in progress or completed*1 Clearing conditions: After reading FER = 1, cleared by writing 0 to FER	(initial value)
1	A framing error has occurred during reception Setting conditions: When the stop bit at the end of the receive data is checked for a value of 1 at the end of reception, and the stop bit is 0*2	

Notes: 1. When bit RE in SCR3 is cleared to 0, bit FER is not affected and retains its previous state.

2. Note that, in 2-stop-bit mode, only the first stop bit is checked for a value of 1, and the second stop bit is not checked. When a framing error occurs the receive data is transferred to RDR but bit RDRF is not set. Reception cannot be continued with bit FER set to 1. In synchronous mode, neither transmission nor reception is possible when bit FER is set to 1.

Bit 3: Parity error (PER)

Bit 3 indicates that a parity error has occurred during reception with parity added in asynchronous mode.

Bit 3 PER	Description	
0	Reception in progress or completed*1 Clearing conditions: After reading PER = 1, cleared by writing 0 to PER	(initial value)
1	A parity error has occurred during reception*2 Setting conditions: When the number of 1 bits in the receive data plus parity bit does not match the parity designated by bit PM in the serial mode register (SMR)	

Notes: 1. When bit RE in SCR3 is cleared to 0, bit PER is not affected and retains its previous state.

2. Receive data in which it a parity error has occurred is still transferred to RDR, but bit RDRF is not set. Reception cannot be continued with bit PER set to 1. In synchronous mode, neither transmission nor reception is possible when bit FER is set to 1.

Bit 2: Transmit end (TEND)

Bit 2 indicates that bit TDRE is set to 1 when the last bit of a transmit character is sent.

Bit 2 is a read-only bit and cannot be modified.

Bit 2 TEND	Description	
0	Transmission in progress Clearing conditions: After reading TDRE = 1, cleared by writing 0 to TDRE When data is written to TDR by an instruction	
1	Transmission ended Setting conditions: When bit TE in SCR3 is cleared to 0 When bit TDRE is set to 1 when the last bit of a transmit character is sent	(initial value)

Bit 1: Multiprocessor bit receive (MPBR)

Bit 1 stores the multiprocessor bit in a receive character during multiprocessor format reception in asynchronous mode.

Bit 1 is a read-only bit and cannot be modified.

Bit 1 MPBR	Description	
0	Data in which the multiprocessor bit is 0 has been received*	(initial value)
1	Data in which the multiprocessor bit is 1 has been received	

Note: * When bit RE is cleared to 0 in SCR3 with the multiprocessor format, bit MPBR is not affected and retains its previous state.

Bit 0: Multiprocessor bit transfer (MPBT)

Bit 0 stores the multiprocessor bit added to transmit data when transmitting in asynchronous mode. The bit MPBT setting is invalid when synchronous mode is selected, when the multiprocessor communication function is disabled, and when not transmitting.

Bit 0 MPBT	Description	
0	A 0 multiprocessor bit is transmitted	(initial value)
1	A 1 multiprocessor bit is transmitted	

10.2.8 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

BRR is an 8-bit register that designates the transmit/receive bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 of the serial mode register (SMR).

BRR can be read or written by the CPU at any time.

BRR is initialized to H'FF upon reset, and in standby, module standby, or watch mode.

Table 10.3 shows examples of BRR settings in asynchronous mode. The values shown are for active (high-speed) mode.

Table 10.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (1)

	OSC														
	32.8 kHz			38.4 kHz			2 MHz			2.4576 MHz			4 MHz		
B Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	Cannot be used,			—	—	—	—	—	—	2	21	–0.83	—	—	—
150	as error			0	3	0	2	12	0.16	3	3	0	2	25	0.16
200	exceeds 3%			0	2	0	0	155	0.16	3	2	0	—	—	—
250				—	—	—	0	124	0	0	153	–0.26	0	249	0
300				0	1	0	0	103	0.16	3	1	0	2	12	0.16
600				0	0	0	0	51	0.16	3	0	0	0	103	0.16
1200				—	—	—	0	25	0.16	2	1	0	0	51	0.16
2400				—	—	—	0	12	0.16	2	0	0	0	25	0.16
4800				—	—	—	—	—	—	0	7	0	0	12	0.16
9600				—	—	—	—	—	—	0	3	0	—	—	—
19200				—	—	—	—	—	—	0	1	0	—	—	—
31250				—	—	—	0	0	0	—	—	—	0	1	0
38400				—	—	—	—	—	—	0	0	0	—	—	—

Table 10.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (2)

	OSC					
	10 MHz			16 MHz		
B Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)
110	2	88	–0.25	2	141	–0.02
150	2	64	0.16	2	103	0.16
200	2	48	–0.35	2	77	0.16
250	2	38	0.16	2	62	–0.79
300	—	—	—	2	51	0.16
600	—	—	—	2	25	0.16
1200	0	129	0.16	0	207	0.16
2400	0	64	0.16	0	103	0.16
4800	—	—	—	0	51	0.16
9600	—	—	—	0	25	0.16
19200	—	—	—	0	12	0.16
31250	0	4	0	0	7	0
38400	—	—	—	—	—	—

Notes: 1. The setting should be made so that the error is not more than 1%.

2. The value set in BRR is given by the following equation:

$$N = \frac{OSC}{(64 \times 2^{2n} \times B)} - 1$$

where

B: Bit rate (bit/s)

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

OSC: Value of $ø_{OSC}$ (Hz)

n: Baud rate generator input clock number (n = 0, 2, or 3)

(The relation between n and the clock is shown in table 10.4.)

3. The error in table 10.3 is the value obtained from the following equation, rounded to two decimal places.

$$\text{Error (\%)} = \frac{\text{B (rate obtained from n, N, OSC)} - \text{R(bit rate in left-hand column in table 10.3.)}}{\text{R (bit rate in left-hand column in table 10.3.)}} \times 100$$

Table 10.4 Relation between n and Clock

n	Clock	SMR Setting CKS1	SMR Setting CKS0
0	ø	0	0
0	$ø_W/2^{*1}/ø_W^{*2}$	0	1
2	ø/16	1	0
3	ø/64	1	1

Notes: 1. ø w/2 clock in active (medium-speed/high-speed) mode and sleep mode

2. ø w clock in subactive mode and subsleep mode

In subactive or subsleep mode, SCI3 can be operated when CPU clock is øw/2 only.

Table 10.5 shows the maximum bit rate for each frequency. The values shown are for active (high-speed) mode.

Table 10.5 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

OSC (MHz)	Maximum Bit Rate (bit/s)	Setting n	Setting N
0.0384*	600	0	0
2	31250	0	0
2.4576	38400	0	0
4	62500	0	0
10	156250	0	0
16	250000	0	0

Note: * When SMR is set up to CKS1 = 0, CKS0 = 1.

Table 10.6 shows examples of BRR settings in synchronous mode. The values shown are for active (high-speed) mode.

Table 10.6 Examples of BRR Settings for Various Bit Rates (Synchronous Mode) (1)

	OSC								
B Bit Rate	**38.4 kHz**			**2 MHz**			**4 MHz**		
(bit/s)	**n**	**N**	**Error**	**n**	**N**	**Error**	**n**	**N**	**Error**
200	0	23	0	—	—	—	—	—	—
250	—	—	—	—	—	—	2	124	0
300	2	0	0	—	—	—	—	—	—
500				—	—	—	—	—	—
1k				0	249	0	—	—	—
2.5k				0	99	0	0	199	0
5k				0	49	0	0	99	0
10k				0	24	0	0	49	0
25k				0	9	0	0	19	0
50k				0	4	0	0	9	0
100k				—	—	—	0	4	0
250k				0	0	0	0	1	0
500k							0	0	0
1M									

Table 10.6 Examples of BRR Settings for Various Bit Rates (Synchronous Mode) (2)

B Bit Rate (bit/s)	OSC 10 MHz			OSC 16 MHz		
	n	N	Error	n	N	Error
200	—	—	—	—	—	—
250	—	—	—	3	124	0
300	—	—	—	—	—	—
500	—	—	—	2	249	0
1k	—	—	—	2	124	0
2.5k	—	—	—	2	49	0
5k	0	249	0	2	24	0
10k	0	124	0	0	199	0
25k	0	49	0	0	79	0
50k	0	24	0	0	39	0
100k	—	—	—	0	19	0
250k	0	4	0	0	7	0
500k	—	—	—	0	3	0
1M	—	—	—	0	1	0

Blank: Cannot be set.

— : A setting can be made, but an error will result.

Notes: The value set in BRR is given by the following equation:

$$N = \frac{OSC}{(8 \times 2^{2n} \times B)} - 1$$

where

B: Bit rate (bit/s)

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

OSC: Value of ϕ_{OSC} (Hz)

n: Baud rate generator input clock number (n = 0, 2, or 3)
(The relation between n and the clock is shown in table 10.7.)

Table 10.7 Relation between n and Clock

n	Clock	SMR Setting CKS1	SMR Setting CKS0
0	ø	0	0
0	$ø_W/2^{*1}/ø_W^{*2}$	0	1
2	ø/16	1	0
3	ø/64	1	1

Notes: 1. ø w/2 clock in active (medium-speed/high-speed) mode and sleep mode

2. ø w clock in subactive mode and subsleep mode

In subactive or subsleep mode, SCI3 can be operated when CPU clock is øw/2 only.

10.2.9 Clock stop register 1 (CKSTPR1)

Bit	7	6	5	4	3	2	1	0
	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	R/W	R/W	—	R/W	—	R/W

CKSTPR1 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bits relating to SCI3 are described here. For details of the other bits, see the sections on the relevant modules.

Bit 5: SCI3 module standby mode control (S32CKSTP)

Bit 5 controls setting and clearing of module standby mode for SCI3.

S32CKSTP	Description	
0	SCI3 is set to module standby mode	
1	SCI3 module standby mode is cleared	(initial value)

Note: All SCI3 register is initialized in module standby mode.

10.2.10 Serial Port Control Register (SPCR)

Bit	7	6	5	4	3	2	1	0
	—	—	SPC32	—	SCINV3	SCINV2	—	—
Initial value	1	1	0	—	0	0	—	—
Read/Write	—	—	R/W	W	R/W	R/W	W	W

SPCR is an 8-bit readable/writable register that performs RXD_{32} and TXD_{32} pin input/output data inversion switching.

Bits 7 and 6: Reserved bits

Bits 7 and 6 are reserved; they are always read as 1 and cannot be modified.

Bit 5: $P4_2$/TXD_{32} pin function switch (SPC32)

This bit selects whether pin $P4_2$/TXD_{32} is used as $P4_2$ or as TXD_{32}.

Bit 5 SPC32	Description	
0	Functions as $P4_2$ I/O pin	(initial value)
1	Functions as TXD_{32} output pin*	

Note: * Set the TE bit in SCR3 after setting this bit to 1.

Bits 4, 1, and 0: Reserved bits

Bits 4, 1, and 0 are reserved; only 0 can be written to these bits.

Bit 3 specifies whether or not TXD_{32} pin output data is to be inverted.

Bit 3 SCINV3	Description	
0	TXD_{32} output data is not inverted	(initial value)
1	TXD_{32} output data is inverted	

Bit 2: RXD_{32} pin input data inversion switch

Bit 2 specifies whether or not RXD_{32} pin input data is to be inverted.

Bit 2 SCINV2	Description	
0	RXD_{32} input data is not inverted	(initial value)
1	RXD_{32} input data is inverted	

10.3 Operation

10.3.1 Overview

SCI3 can perform serial communication in two modes: asynchronous mode in which synchronization is provided character by character, and synchronous mode in which synchronization is provided by clock pulses. The serial mode register (SMR) is used to select asynchronous or synchronous mode and the data transfer format, as shown in table 10.8.

The clock source for SCI3 is determined by bit COM in SMR and bits CKE1 and CKE0 in SCR3, as shown in table 10.9.

1. Asynchronous mode

- Choice of 5-, 7-, or 8-bit data length
- Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits. (The combination of these parameters determines the data transfer format and the character length.)
- Framing error (FER), parity error (PER), overrun error (OER), and break detection during reception
- Choice of internal or external clock as the clock source

  When internal clock is selected: SCI3 operates on the baud rate generator clock, and a clock with the same frequency as the bit rate can be output.

  When external clock is selected: A clock with a frequency 16 times the bit rate must be input. (The on-chip baud rate generator is not used.)

2. Synchronous mode

- Data transfer format: Fixed 8-bit data length
- Overrun error (OER) detection during reception
- Choice of internal or external clock as the clock source

  When internal clock is selected: SCI3 operates on the baud rate generator clock, and a serial clock is output.

  When external clock is selected: The on-chip baud rate generator is not used, and SCI3 operates on the input serial clock.

Table 10.8 SMR Settings and Corresponding Data Transfer Formats

SMR						Data Transfer Format			
bit 7 COM	**bit 6 CHR**	**bit 2 MP**	**bit 5 PE**	**bit 3 STOP**	**Mode**	**Data Length**	**Multiprocessor Bit**	**Parity Bit**	**Stop Bit Length**
0	0	0	0	0	Asynchronous mode	8-bit data	No	No	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	1		0	0		7-bit data		No	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	0	1	0	0		8-bit data	Yes	No	1 bit
				1					2 bits
			1	0		5-bit data	No		1 bit
				1					2 bits
	1		0	0		7-bit data	Yes		1 bit
				1					2 bits
			1	0		5-bit data	No	Yes	1 bit
				1					2 bits
1	*	0	*	*	Synchronous mode	8-bit data	No	No	No

*: Don't care

Table 10.9 SMR and SCR3 Settings and Clock Source Selection

SMR	SCR3			Transmit/Receive Clock	
bit 7	bit 1	bit 0			
COM	CKE1	CKE0	Mode	Clock Source	SCK_{32} Pin Function
0	0	0	Asynchronous mode	Internal	I/O port (SCK_{32} pin not used)
		1			Outputs clock with same frequency as bit rate
	1	0		External	Inputs clock with frequency 16 times bit rate
1	0	0	Synchronous mode	Internal	Outputs serial clock
	1	0		External	Inputs serial clock
0	1	1	Reserved (Do not specify these combinations)		
1	0	1			
1	1	1			

3. Interrupts and continuous transmission/reception

SCI3 can carry out continuous reception using RXI and continuous transmission using TXI. These interrupts are shown in table 10.10.

Table 10.10 Transmit/Receive Interrupts

Interrupt	Flags	Interrupt Request Conditions	Notes
RXI	RDRF RIE	When serial reception is performed normally and receive data is transferred from RSR to RDR, bit RDRF is set to 1, and if bit RIE is set to 1 at this time, RXI is enabled and an interrupt is requested. (See figure 10.2 (a).)	The RXI interrupt routine reads the receive data transferred to RDR and clears bit RDRF to 0. Continuous reception can be performed by repeating the above operations until reception of the next RSR data is completed.
TXI	TDRE TIE	When TSR is found to be empty (on completion of the previous transmission) and the transmit data placed in TDR is transferred to TSR, bit TDRE is set to 1. If bit TIE is set to 1 at this time, TXI is enabled and an interrupt is requested. (See figure 10.2 (b).)	The TXI interrupt routine writes the next transmit data to TDR and clears bit TDRE to 0. Continuous transmission can be performed by repeating the above operations until the data transferred to TSR has been transmitted.
TEI	TEND TEIE	When the last bit of the character in TSR is transmitted, if bit TDRE is set to 1, bit TEND is set to 1. If bit TEIE is set to 1 at this time, TEI is enabled and an interrupt is requested. (See figure 10.2 (c).)	TEI indicates that the next transmit data has not been written to TDR when the last bit of the transmit character in TSR is sent.

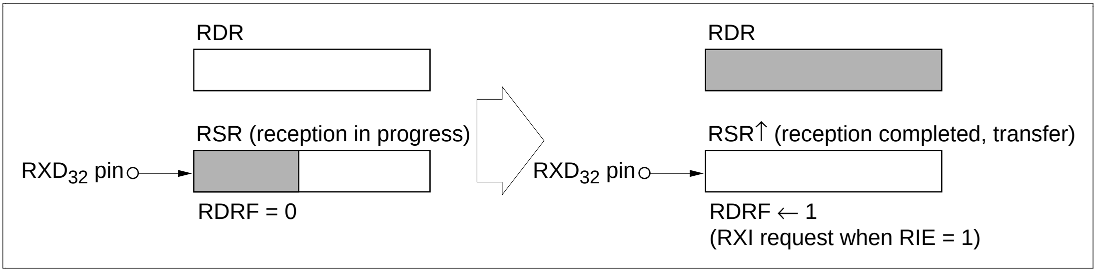


Figure 10.2 (a) RDRF Setting and RXI Interrupt

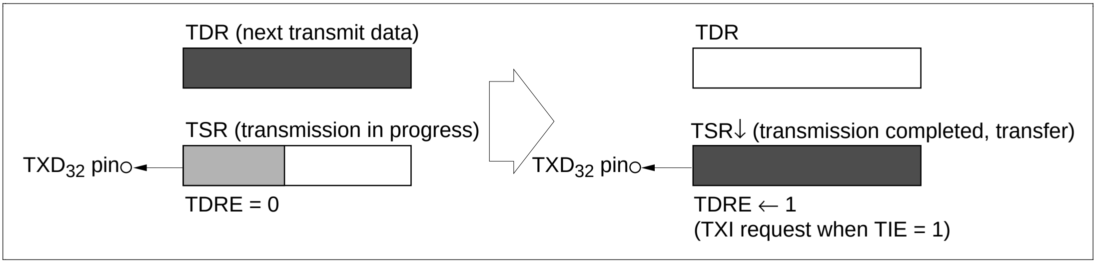


Figure 10.2 (b) TDRE Setting and TXI Interrupt

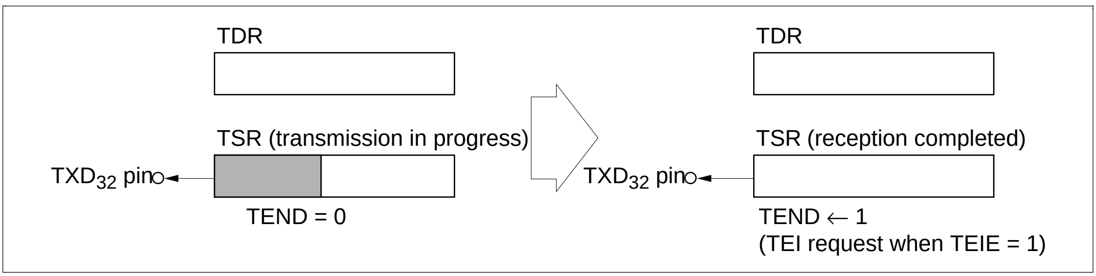


Figure 10.2 (c) TEND Setting and TEI Interrupt

10.3.2 Operation in Asynchronous Mode

In asynchronous mode, serial communication is performed with synchronization provided character by character. A start bit indicating the start of communication and one or two stop bits indicating the end of communication are added to each character before it is sent.

SCI3 has separate transmission and reception units, allowing full-duplex communication. As the transmission and reception units are both double-buffered, data can be written during transmission and read during reception, making possible continuous transmission and reception.

1. Data transfer format

The general data transfer format in asynchronous communication is shown in figure 10.3.

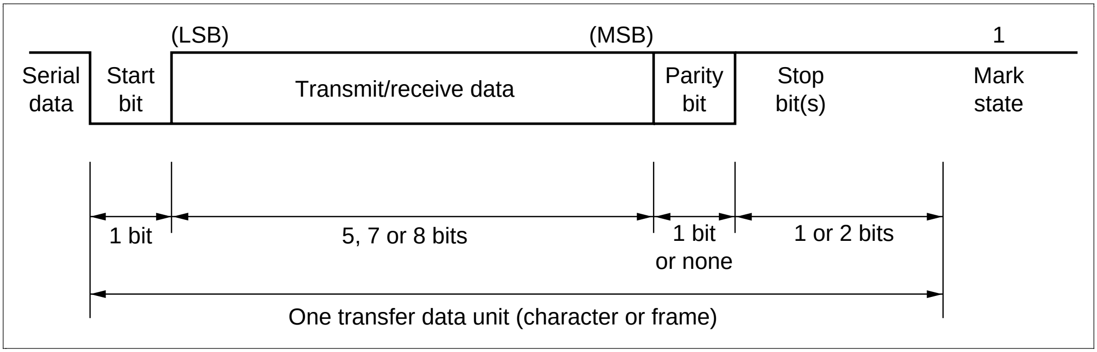


Figure 10.3 Data Format in Asynchronous Communication

In asynchronous communication, the communication line is normally in the mark state (high level). SCI3 monitors the communication line and when it detects a space (low level), identifies this as a start bit and begins serial data communication.

One transfer data character consists of a start bit (low level), followed by transmit/receive data (LSB-first format, starting from the least significant bit), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, synchronization is performed by the falling edge of the start bit during reception. The data is sampled on the 8th pulse of a clock with a frequency 16 times the bit period, so that the transfer data is latched at the center of each bit.

Table 10.11 shows the 16 data transfer formats that can be set in asynchronous mode. The format is selected by the settings in the serial mode register (SMR).

Table 10.11 Data Transfer Formats (Asynchronous Mode)

SMR				Serial Data Transfer 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	0	1	0	S	8-bit data								MPB	STOP	
0	0	1	1	S	8-bit data								MPB	STOP	STOP
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
0	1	1	0	S	5-bit data					STOP					
0	1	1	1	S	5-bit data					STOP	STOP				
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	0	1	0	S	7-bit data							MPB	STOP		
1	0	1	1	S	7-bit data							MPB	STOP	STOP	
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
1	1	1	0	S	5-bit data					P	STOP				
1	1	1	1	S	5-bit data					P	STOP	STOP			

Notation:
S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

2. Clock

Either an internal clock generated by the baud rate generator or an external clock input at the SCK_{32} pin can be selected as the SCI3 transmit/receive clock. The selection is made by means of bit COM in SMR and bits SCE1 and CKE0 in SCR3. See table 10.9 for details on clock source selection.

When an external clock is input at the SCK_{32} pin, the clock frequency should be 16 times the bit rate.

When SCI3 operates on an internal clock, the clock can be output at the SCK_{32} pin. In this case the frequency of the output clock is the same as the bit rate, and the phase is such that the clock rises at the center of each bit of transmit/receive data, as shown in figure 10.4.

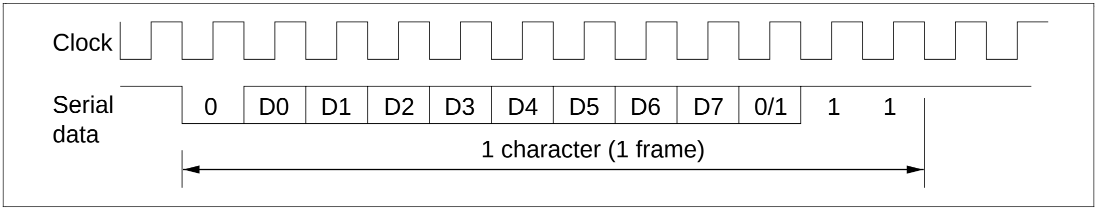


Figure 10.4 Phase Relationship between Output Clock and Transfer Data (Asynchronous Mode) (8-bit data, parity, 2 stop bits)

3. Data transfer operations

- SCI3 initialization

Before data is transferred on SCI3, bits TE and RE in SCR3 must first be cleared to 0, and then SCI3 must be initialized as follows.

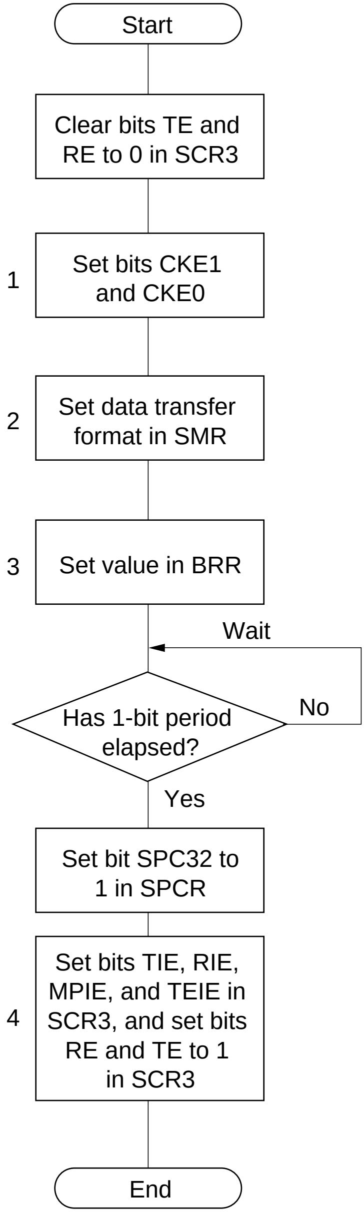
Note: If the operation mode or data transfer format is changed, bits TE and RE must first be cleared to 0.

When bit TE is cleared to 0, bit TDRE is set to 1.

Note that the RDRF, PER, FER, and OER flags and the contents of RDR are retained when RE is cleared to 0.

When an external clock is used in asynchronous mode, the clock should not be stopped during operation, including initialization. When an external clock is used in synchronous mode, the clock should not be supplied during operation, including initialization.

Figure 10.5 shows an example of a flowchart for initializing SCI3.



1. Set clock selection in SCR3. Be sure to clear the other bits to 0. If clock output is selected in asynchronous mode, the clock is output immediately after setting bits CKE1 and CKE0. If clock output is selected for reception in synchronous mode, the clock is output immediately after bits CKE1, CKE0, and RE are set to 1.

2. Set the data transfer format in the serial mode register (SMR).

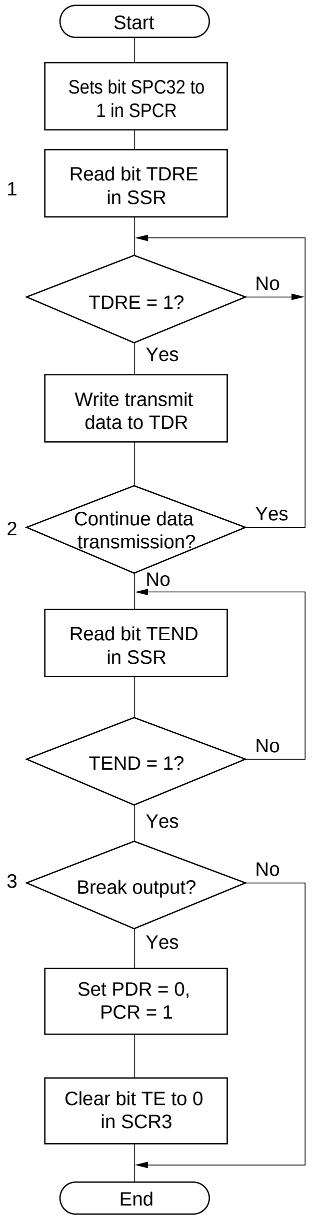
3. Write the value corresponding to the transfer rate in BRR. This operation is not necessary when an external clock is selected.

4. Wait for at least one bit period, then set bits TIE, RIE, MPIE, and TEIE in SCR3, and set bits RE and TE to 1 in PMR7. Setting bits TE and RE enables the TXD_{32} and RXD_{32} pins to be used. In asynchronous mode the mark state is established when transmitting, and the idle state waiting for a start bit when receiving.

Figure 10.5 Example of SCI3 Initialization Flowchart

• Transmitting

Figure 10.6 shows an example of a flowchart for data transmission. This procedure should be followed for data transmission after initializing SCI3.



1. Read the serial status register (SSR) and check that bit TDRE is set to 1, then write transmit data to the transmit data register (TDR). When data is written to TDR, bit TDRE is cleared to 0 automatically.
(After the TE bit is set to 1, one frame of 1s is output, then transmission is possible.)

2. When continuing data transmission, be sure to read TDRE = 1 to confirm that a write can be performed before writing data to TDR. When data is written to TDR, bit TDRE is cleared to 0 automatically.

3. If a break is to be output when data transmission ends, set the port PCR to 1 and clear the port PDR to 0, then clear bit TE in SCR3 to 0.

Figure 10.6 Example of Data Transmission Flowchart (Asynchronous Mode)

SCI3 operates as follows when transmitting data.

SCI3 monitors bit TDRE in SSR, and when it is cleared to 0, recognizes that data has been written to TDR and transfers data from TDR to TSR. It then sets bit TDRE to 1 and starts transmitting. If bit TIE in SCR3 is set to 1 at this time, a TXI request is made.

Serial data is transmitted from the TXD_{32} pin using the relevant data transfer format in table 10.11. When the stop bit is sent, SCI3 checks bit TDRE. If bit TDRE is cleared to 0, SCI3 transfers data from TDR to TSR, and when the stop bit has been sent, starts transmission of the next frame. If bit TDRE is set to 1, bit TEND in SSR bit is set to 1the mark state, in which 1s are transmitted, is established after the stop bit has been sent. If bit TEIE in SCR3 is set to 1 at this time, a TEI request is made.

Figure 10.7 shows an example of the operation when transmitting in asynchronous mode.

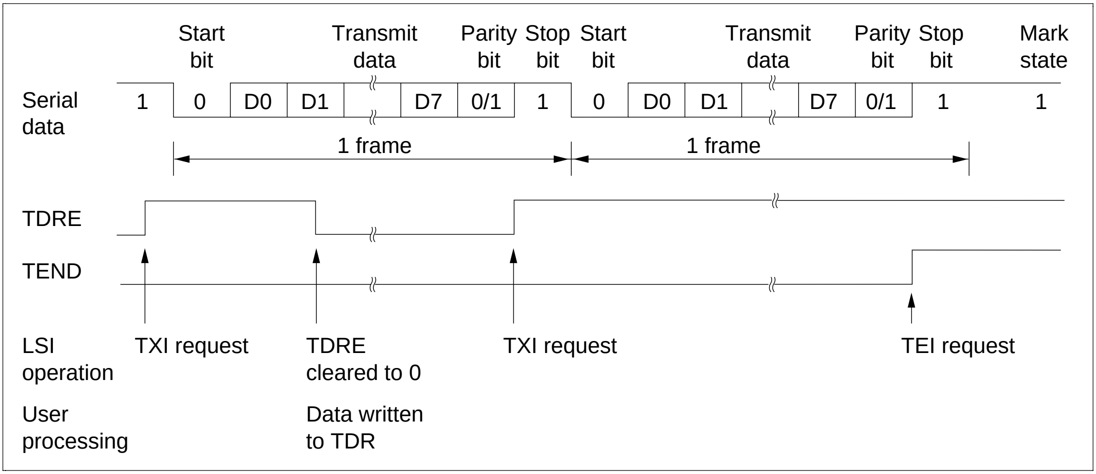
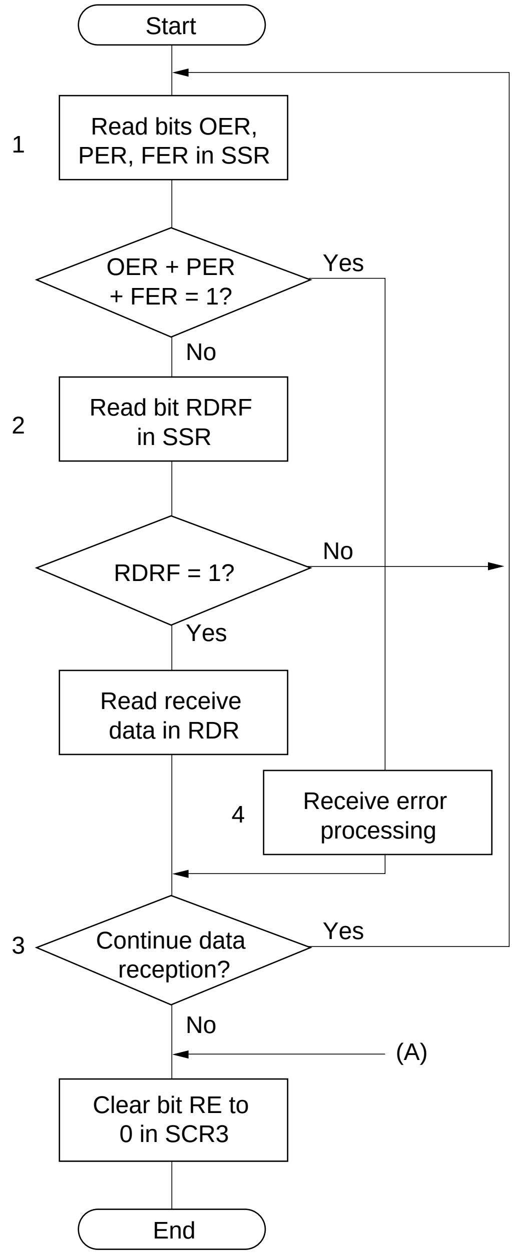


Figure 10.7 Example of Operation when Transmitting in Asynchronous Mode (8-bit data, parity, 1 stop bit)

• Receiving

Figure 10.8 shows an example of a flowchart for data reception. This procedure should be followed for data reception after initializing SCI3.

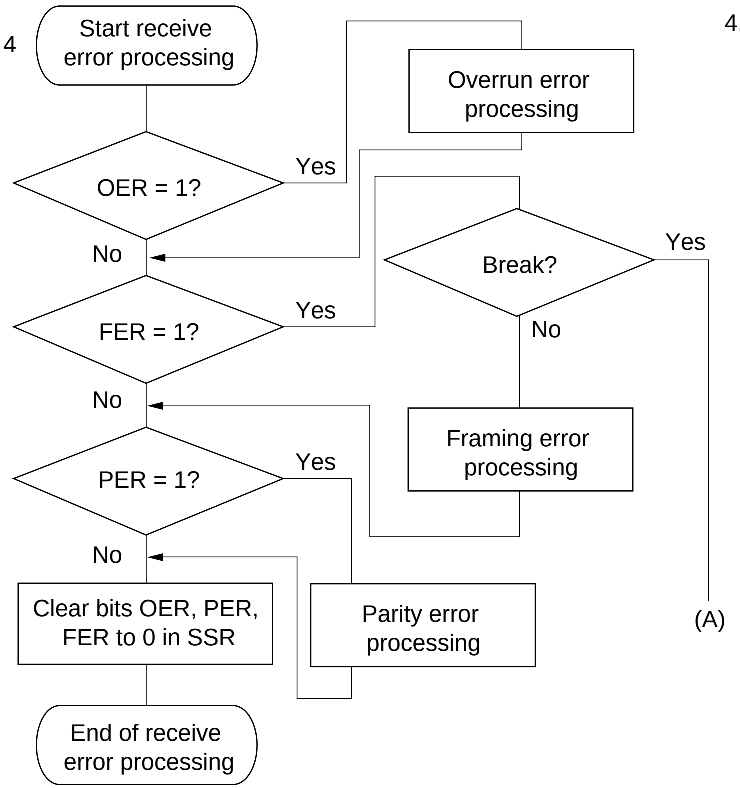


1. Read bits OER, PER, and FER in the serial status register (SSR) to determine if there is an error. If a receive error has occurred, execute receive error processing.

2. Read SSR and check that bit RDRF is set to 1. If it is, read the receive data in RDR. When the RDR data is read, bit RDRF is cleared to 0 automatically.

3. When continuing data reception, finish reading of bit RDRF and RDR before receiving the stop bit of the current frame. When the data in RDR is read, bit RDRF is cleared to 0 automatically.

Figure 10.8 Example of Data Reception Flowchart (Asynchronous Mode)



4. If a receive error has occurred, read bits OER, PER, and FER in SSR to identify the error, and after carrying out the necessary error processing, ensure that bits OER, PER, and FER are all cleared to 0. Reception cannot be resumed if any of these bits is set to 1. In the case of a framing error, a break can be detected by reading the value of the RXD_{32} pin.

Figure 10.8 Example of Data Reception Flowchart (Asynchronous Mode) (cont)

SCI3 operates as follows when receiving data.

SCI3 monitors the communication line, and when it detects a 0 start bit, performs internal synchronization and begins reception. Reception is carried out in accordance with the relevant data transfer format in table 10.11. The received data is first placed in RSR in LSB-to-MSB order, and then the parity bit and stop bit(s) are received. SCI3 then carries out the following checks.

- Parity check

  SCI3 checks that the number of 1 bits in the receive data conforms to the parity (odd or even) set in bit PM in the serial mode register (SMR).
- Stop bit check

  SCI3 checks that the stop bit is 1. If two stop bits are used, only the first is checked.
- Status check

  SCI3 checks that bit RDRF is set to 0, indicating that the receive data can be transferred from RSR to RDR.

If no receive error is found in the above checks, bit RDRF is set to 1, and the receive data is stored in RDR. If bit RIE is set to 1 in SCR3, an RXI interrupt is requested. If the error checks identify a receive error, bit OER, PER, or FER is set to 1 depending on the kind of error. Bit RDRF retains its state prior to receiving the data. If bit RIE is set to 1 in SCR3, an ERI interrupt is requested.

Table 10.12 shows the conditions for detecting a receive error, and receive data processing.

Note: No further receive operations are possible while a receive error flag is set. Bits OER, FER, PER, and RDRF must therefore be cleared to 0 before resuming reception.

Table 10.12 Receive Error Detection Conditions and Receive Data Processing

Receive Error	Abbreviation	Detection Conditions	Receive Data Processing
Overrun error	OER	When the next date receive operation is completed while bit RDRF is still set to 1 in SSR	Receive data is not transferred from RSR to RDR
Framing error	FER	When the stop bit is 0	Receive data is transferred from RSR to RDR
Parity error	PER	When the parity (odd or even) set in SMR is different from that of the received data	Receive data is transferred from RSR to RDR

Figure 10.9 shows an example of the operation when receiving in asynchronous mode.

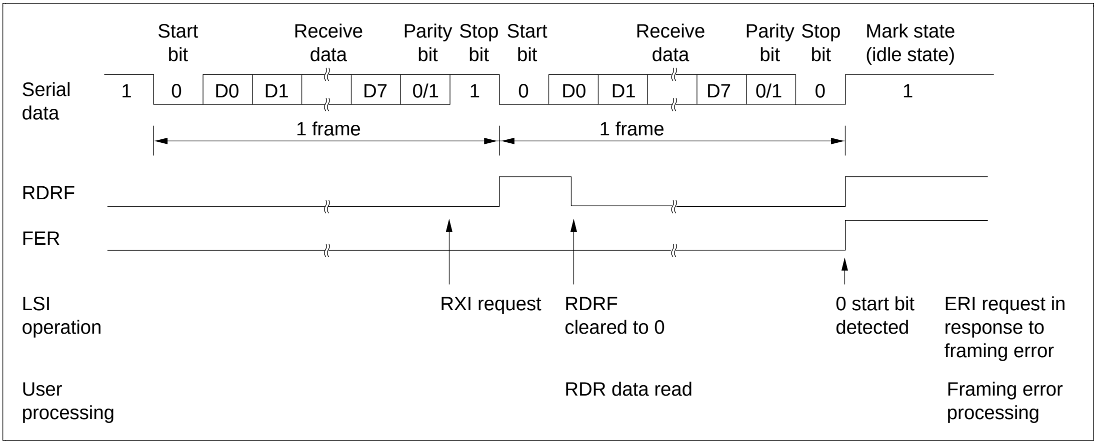


Figure 10.9 Example of Operation when Receiving in Asynchronous Mode (8-bit data, parity, 1 stop bit)

10.3.3 Operation in Synchronous Mode

In synchronous mode, SCI3 transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

SCI3 has separate transmission and reception units, allowing full-duplex communication with a shared clock.

As the transmission and reception units are both double-buffered, data can be written during transmission and read during reception, making possible continuous transmission and reception.

1. Data transfer format

The general data transfer format in asynchronous communication is shown in figure 10.10.

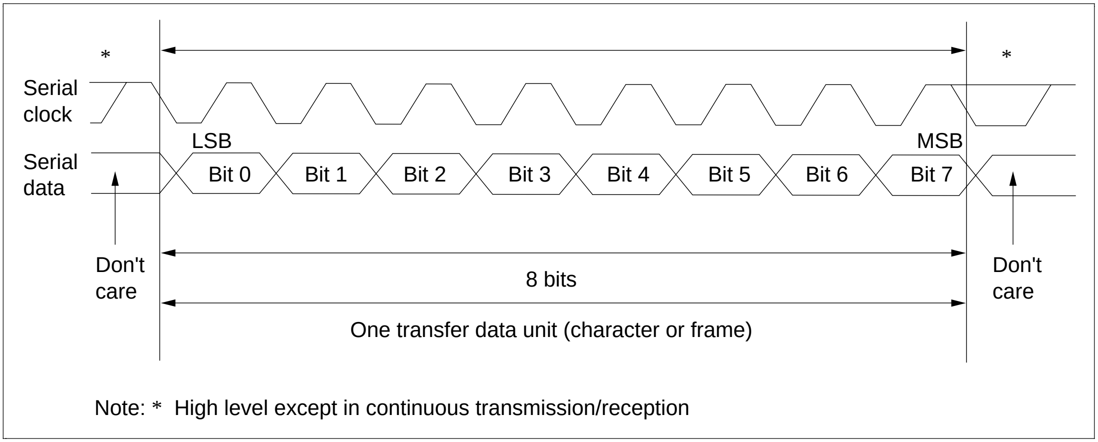


Figure 10.10 Data Format in Synchronous Communication

In synchronous communication, data on the communication line is output from one falling edge of the serial clock until the next falling edge. Data confirmation is guaranteed at the rising edge of the serial clock.

One transfer data character begins with the LSB and ends with the MSB. After output of the MSB, the communication line retains the MSB state.

When receiving in synchronous mode, SCI3 latches receive data at the rising edge of the serial clock.

The data transfer format uses a fixed 8-bit data length.

Parity and multiprocessor bits cannot be added.

2. Clock

Either an internal clock generated by the baud rate generator or an external clock input at the SCK_{32} pin can be selected as the SCI3 serial clock. The selection is made by means of bit COM in SMR and bits CKE1 and CKE0 in SCR3. See table 10.9 for details on clock source selection.

When SCI3 operates on an internal clock, the serial clock is output at the SCK_{32} pin. Eight pulses of the serial clock are output in transmission or reception of one character, and when SCI3 is not transmitting or receiving, the clock is fixed at the high level.

3. Data transfer operations

• SCI3 initialization

Data transfer on SCI3 first of all requires that SCI3 be initialized as described in 10.3.2 3. SCI3 initialization, and shown in figure 10.5.

• Transmitting

Figure 10.11 shows an example of a flowchart for data transmission. This procedure should be followed for data transmission after initializing SCI3.

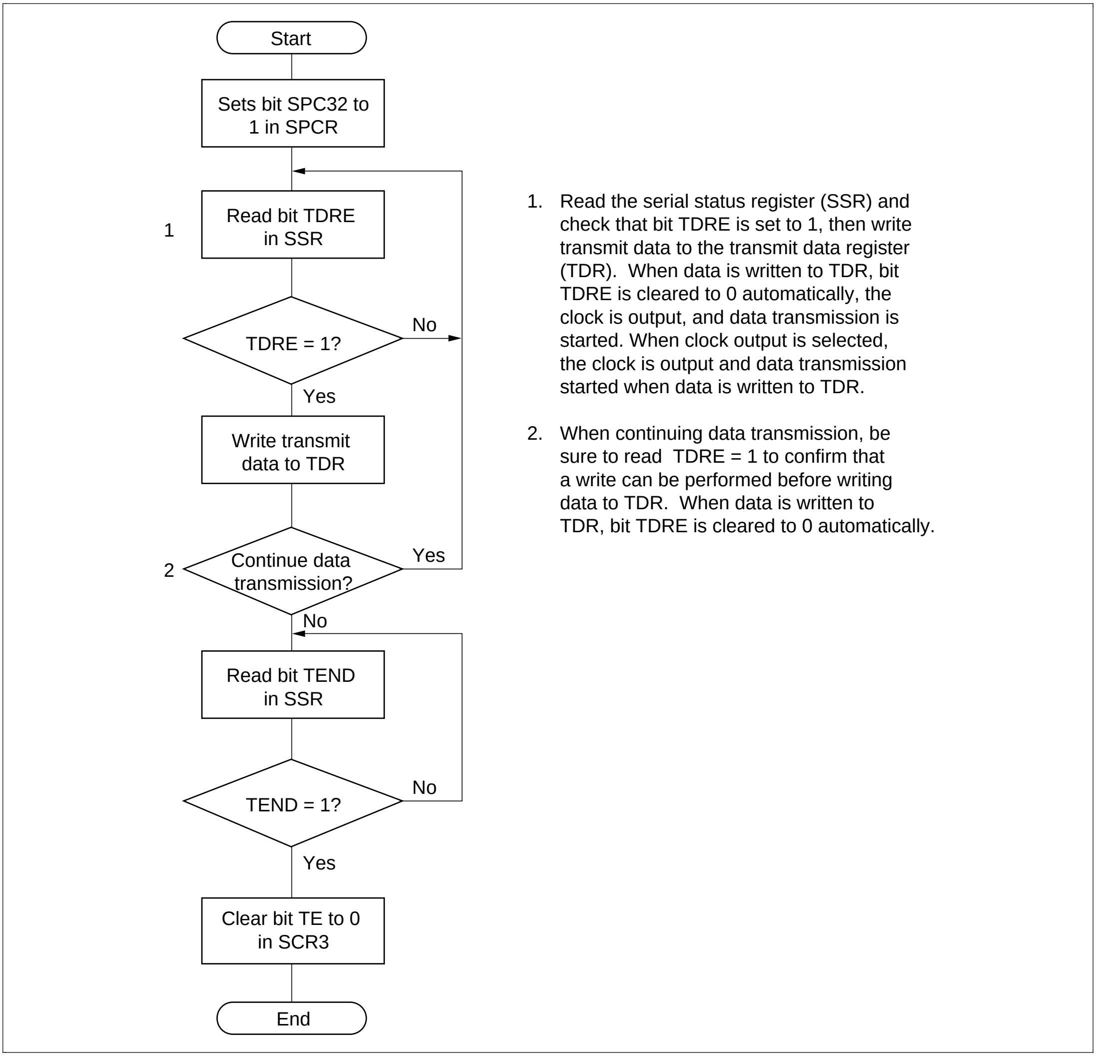


Figure 10.11 Example of Data Transmission Flowchart (Synchronous Mode)

SCI3 operates as follows when transmitting data.

SCI3 monitors bit TDRE in SSR, and when it is cleared to 0, recognizes that data has been written to TDR and transfers data from TDR to TSR. It then sets bit TDRE to 1 and starts transmitting. If bit TIE in SCR3 is set to 1 at this time, a TXI request is made.

When clock output mode is selected, SCI3 outputs 8 serial clock pulses. When an external clock is selected, data is output in synchronization with the input clock.

Serial data is transmitted from the TXD32 pin in order from the LSB (bit 0) to the MSB (bit 7). When the MSB (bit 7) is sent, checks bit TDRE. If bit TDRE is cleared to 0, SCI3 transfers data from TDR to TSR, and starts transmission of the next frame. If bit TDRE is set to 1, SCI3 sets bit TEND to 1 in SSR, and after sending the MSB (bit 7), retains the MSB state. If bit TEIE in SCR3 is set to 1 at this time, a TEI request is made.

After transmission ends, the SCK pin is fixed at the high level.

Note: Transmission is not possible if an error flag (OER, FER, or PER) that indicates the data reception status is set to 1. Check that these error flags are all cleared to 0 before a transmit operation.

Figure 10.12 shows an example of the operation when transmitting in synchronous mode.

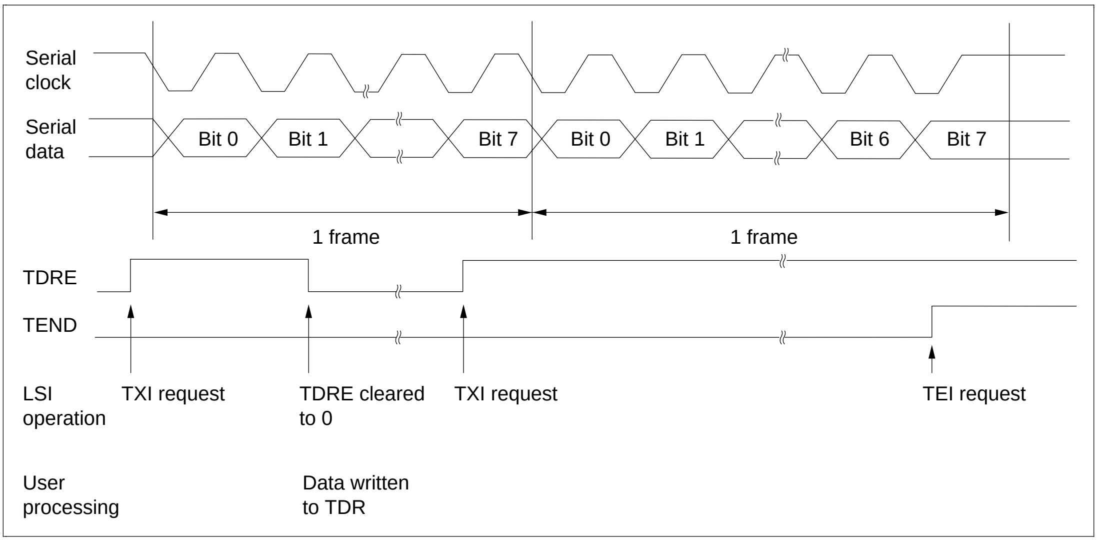


Figure 10.12 Example of Operation when Transmitting in Synchronous Mode

• Receiving

Figure 10.13 shows an example of a flowchart for data reception. This procedure should be followed for data reception after initializing SCI3.

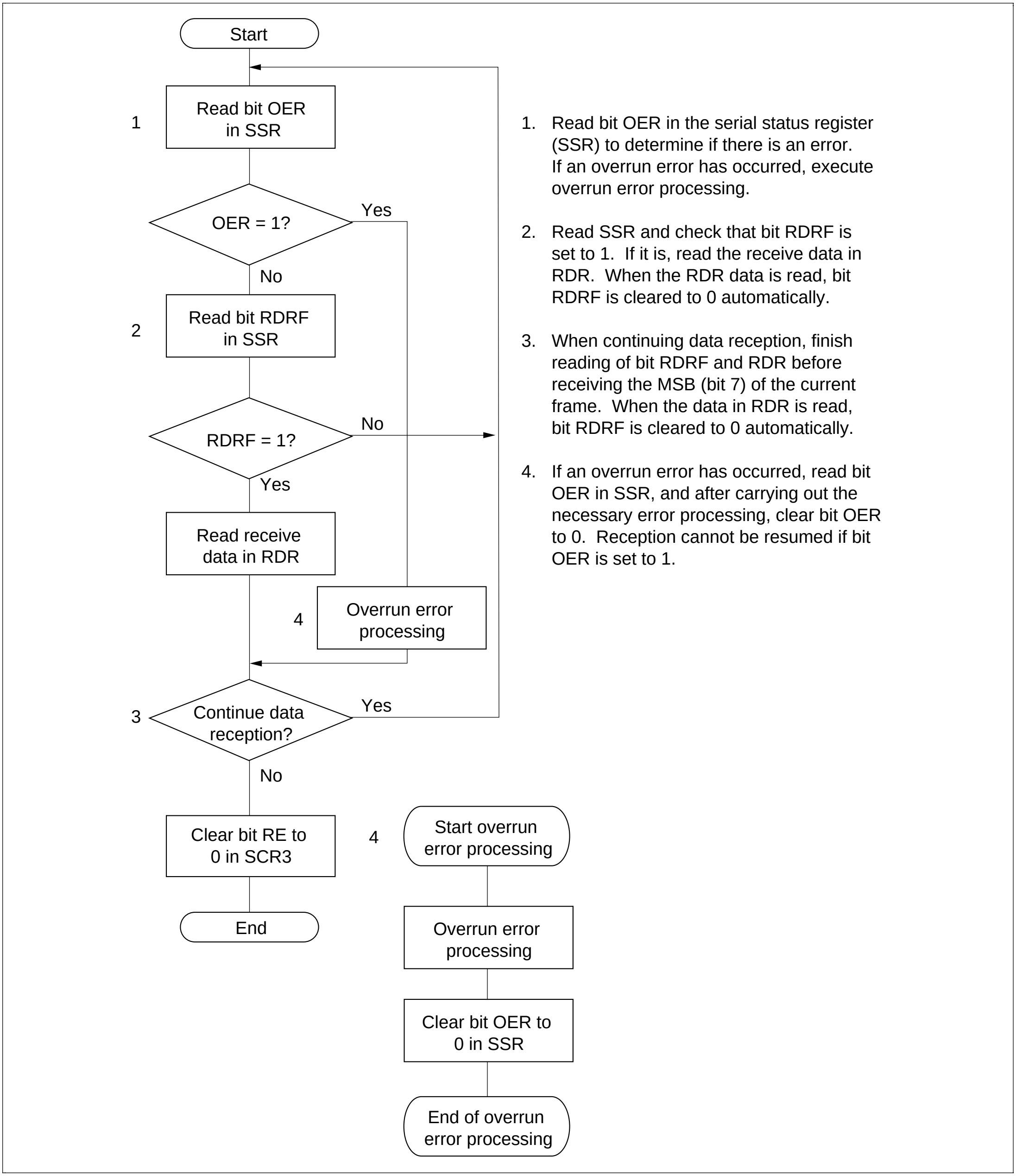


Figure 10.13 Example of Data Reception Flowchart (Synchronous Mode)

SCI3 operates as follows when receiving data.

SCI3 performs internal synchronization and begins reception in synchronization with the serial clock input or output.

The received data is placed in RSR in LSB-to-MSB order.

After the data has been received, SCI3 checks that bit RDRF is set to 0, indicating that the receive data can be transferred from RSR to RDR.

If this check shows that there is no overrun error, bit RDRF is set to 1, and the receive data is stored in RDR. If bit RIE is set to 1 in SCR3, an RXI interrupt is requested. If the check identifies an overrun error, bit OER is set to 1.

Bit RDRF remains set to 1. If bit RIE is set to 1 in SCR3, an ERI interrupt is requested.

See table 10.12 for the conditions for detecting a receive error, and receive data processing.

Note: No further receive operations are possible while a receive error flag is set. Bits OER, FER, PER, and RDRF must therefore be cleared to 0 before resuming reception.

Figure 10.14 shows an example of the operation when receiving in synchronous mode.

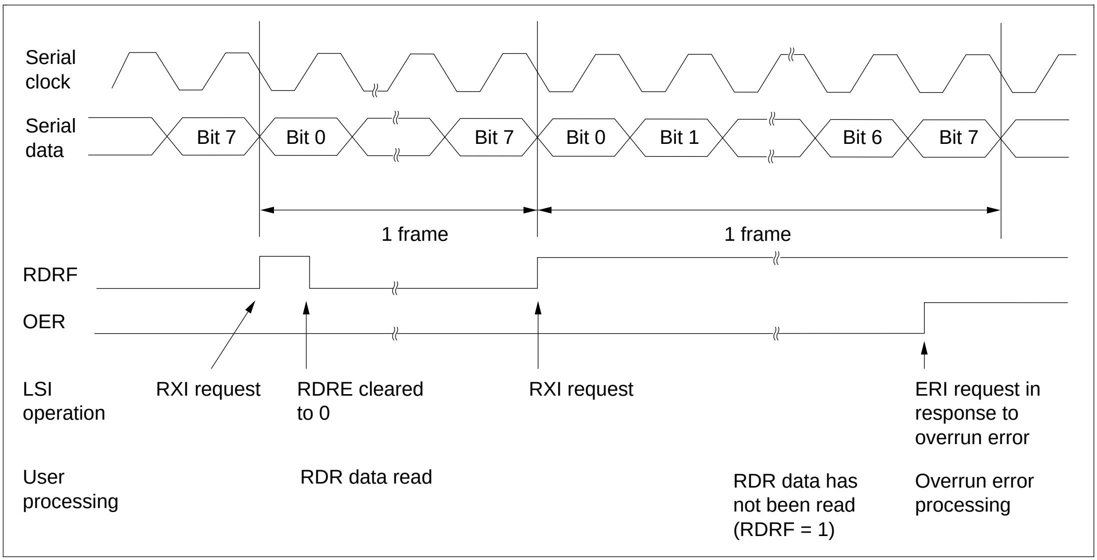
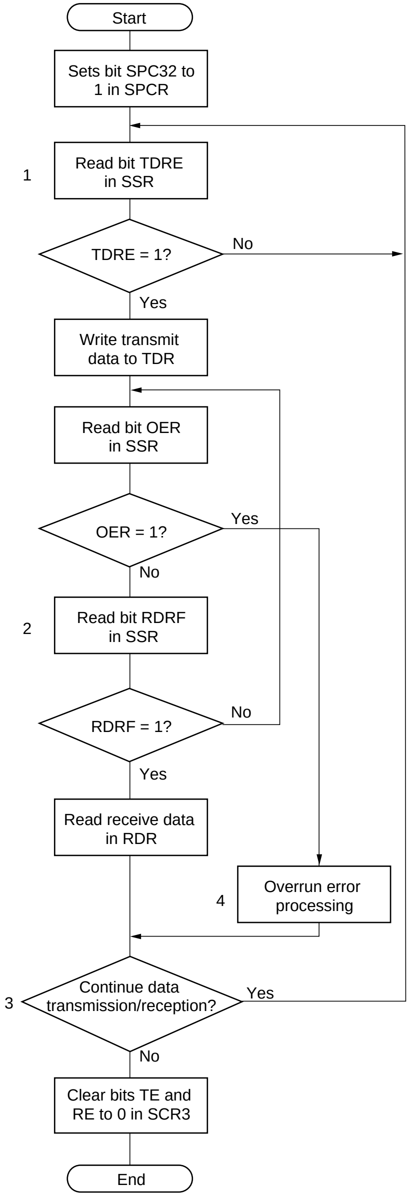


Figure 10.14 Example of Operation when Receiving in Synchronous Mode

• Simultaneous transmit/receive

Figure 10.15 shows an example of a flowchart for a simultaneous transmit/receive operation. This procedure should be followed for simultaneous transmission/reception after initializing SCI3.



1. Read the serial status register (SSR) and check that bit TDRE is set to 1, then write transmit data to the transmit data register (TDR). When data is written to TDR, bit TDRE is cleared to 0 automatically.

2. Read SSR and check that bit RDRF is set to 1. If it is, read the receive data in RDR. When the RDR data is read, bit RDRF is cleared to 0 automatically.

3. When continuing data transmission/reception, finish reading of bit RDRF and RDR before receiving the MSB (bit 7) of the current frame. Before receiving the MSB (bit 7) of the current frame, also read TDRE = 1 to confirm that a write can be performed, then write data to TDR. When data is written to TDR, bit TDRE is cleared to 0 automatically, and when the data in RDR is read, bit RDRF is cleared to 0 automatically.

4. If an overrun error has occurred, read bit OER in SSR, and after carrying out the necessary error processing, clear bit OER to 0. Transmission and reception cannot be resumed if bit OER is set to 1.
See figure 10-13 for details on overrun error processing.

Figure 10.15 Example of Simultaneous Data Transmission/Reception Flowchart (Synchronous Mode)

Notes: 1. When switching from transmission to simultaneous transmission/reception, check that SCI3 has finished transmitting and that bits TDRE and TEND are set to 1, clear bit TE to 0, and then set bits TE and RE to 1 simultaneously.

2. When switching from reception to simultaneous transmission/reception, check that SCI3 has finished receiving, clear bit RE to 0, then check that bit RDRF and the error flags (OER, FER, and PER) are cleared to 0, and finally set bits TE and RE to 1 simultaneously.

10.3.4 Multiprocessor Communication Function

The multiprocessor communication function enables data to be exchanged among a number of processors on a shared communication line. Serial data communication is performed in asynchronous mode using the multiprocessor format (in which a multiprocessor bit is added to the transfer data).

In multiprocessor communication, each receiver is assigned its own ID code. The serial communication cycle consists of two cycles, an ID transmission cycle in which the receiver is specified, and a data transmission cycle in which the transfer data is sent to the specified receiver. These two cycles are differentiated by means of the multiprocessor bit, 1 indicating an ID transmission cycle, and 0, a data transmission cycle.

The sender first sends transfer data with a 1 multiprocessor bit added to the ID code of the receiver it wants to communicate with, and then sends transfer data with a 0 multiprocessor bit added to the transmit data. When a receiver receives transfer data with the multiprocessor bit set to 1, it compares the ID code with its own ID code, and if they are the same, receives the transfer data sent next. If the ID codes do not match, it skips the transfer data until data with the multiprocessor bit set to 1 is sent again.

In this way, a number of processors can exchange data among themselves.

Figure 10.16 shows an example of communication between processors using the multiprocessor format.

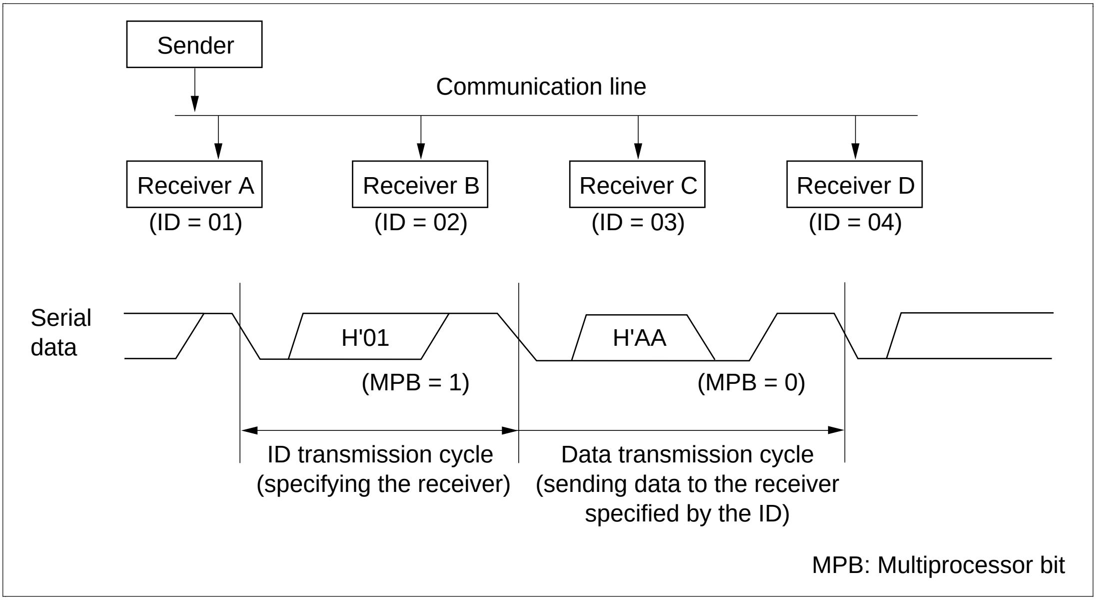


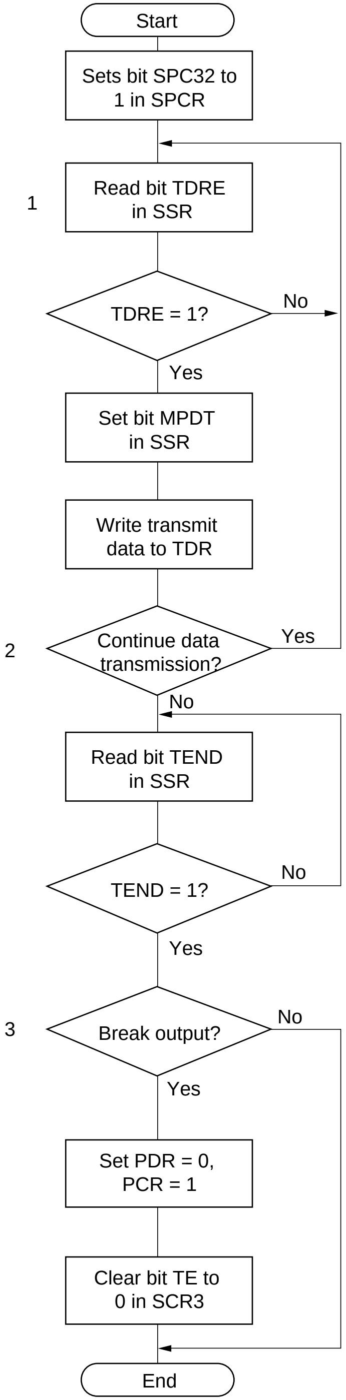
Figure 10.16 Example of Inter-Processor Communication Using Multiprocessor Format (Sending data H'AA to receiver A)

There is a choice of four data transfer formats. If a multiprocessor format is specified, the parity bit specification is invalid. See table 10.11 for details.

For details on the clock used in multiprocessor communication, see 10.3.2, Operation in Synchronous Mode.

- Multiprocessor transmitting

Figure 10.17 shows an example of a flowchart for multiprocessor data transmission. This procedure should be followed for multiprocessor data transmission after initializing SCI3.



1. Read the serial status register (SSR) and check that bit TDRE is set to 1, then set bit MPBT in SSR to 0 or 1 and write transmit data to the transmit data register (TDR). When data is written to TDR, bit TDRE is cleared to 0 automatically.

2. When continuing data transmission, be sure to read TDRE = 1 to confirm that a write can be performed before writing data to TDR. When data is written to TDR, bit TDRE is cleared to 0 automatically.

3. If a break is to be output when data transmission ends, set the port PCR to 1 and clear the port PDR to 0, then clear bit TE in SCR3 to 0.

Figure 10.17 Example of Multiprocessor Data Transmission Flowchart

SCI3 operates as follows when transmitting data.

SCI3 monitors bit TDRE in SSR, and when it is cleared to 0, recognizes that data has been written to TDR and transfers data from TDR to TSR. It then sets bit TDRE to 1 and starts transmitting. If bit TIE in SCR3 is set to 1 at this time, a TXI request is made.

Serial data is transmitted from the TXD pin using the relevant data transfer format in table 10.11. When the stop bit is sent, SCI3 checks bit TDRE. If bit TDRE is cleared to 0, SCI3 transfers data from TDR to TSR, and when the stop bit has been sent, starts transmission of the next frame. If bit TDRE is set to 1 bit TEND in SSR bit is set to 1, the mark state, in which 1s are transmitted, is established after the stop bit has been sent. If bit TEIE in SCR3 is set to 1 at this time, a TEI request is made.

Figure 10.18 shows an example of the operation when transmitting using the multiprocessor format.

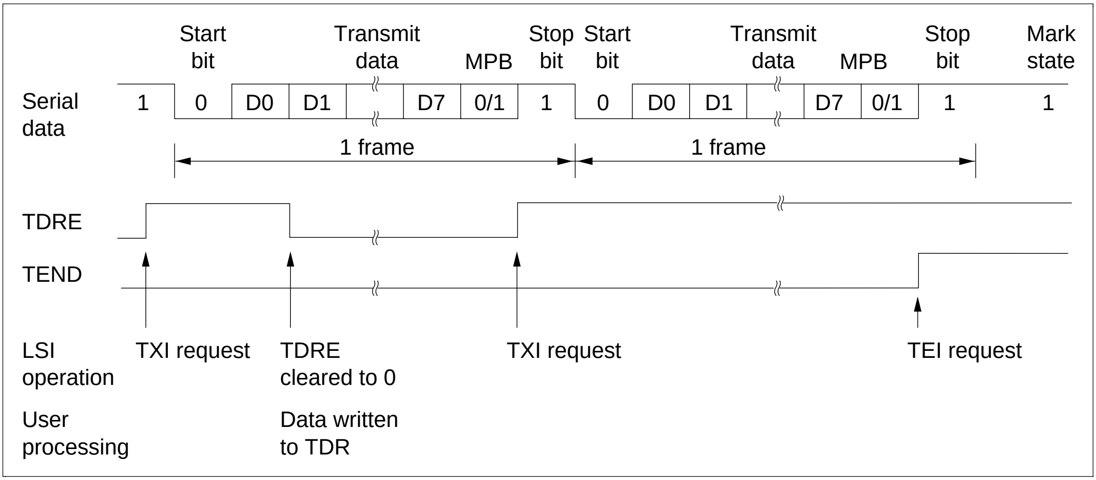
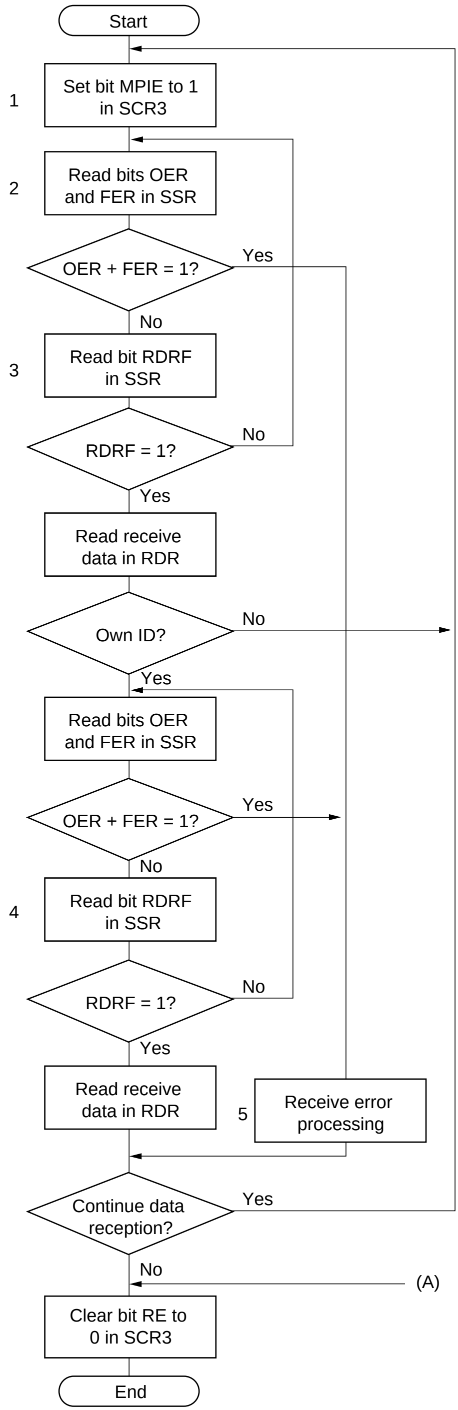


Figure 10.18 Example of Operation when Transmitting using Multiprocessor Format (8-bit data, multiprocessor bit, 1 stop bit)

- Multiprocessor receiving

Figure 10.19 shows an example of a flowchart for multiprocessor data reception. This procedure should be followed for multiprocessor data reception after initializing SCI3.



1. Set bit MPIE to 1 in SCR3.

2. Read bits OER and FER in the serial status register (SSR) to determine if there is an error. If a receive error has occurred, execute receive error processing.

3. Read SSR and check that bit RDRF is set to 1. If it is, read the receive data in RDR and compare it with this receiver's own ID. If the ID is not this receiver's, set bit MPIE to 1 again. When the RDR data is read, bit RDRF is cleared to 0 automatically.

4. Read SSR and check that bit RDRF is set to 1, then read the data in RDR.

5. If a receive error has occurred, read bits OER and FER in SSR to identify the error, and after carrying out the necessary error processing, ensure that bits OER and FER are both cleared to 0. Reception cannot be resumed if either of these bits is set to 1. In the case of a framing error, a break can be detected by reading the value of the RXD_{32} pin.

Figure 10.19 Example of Multiprocessor Data Reception Flowchart

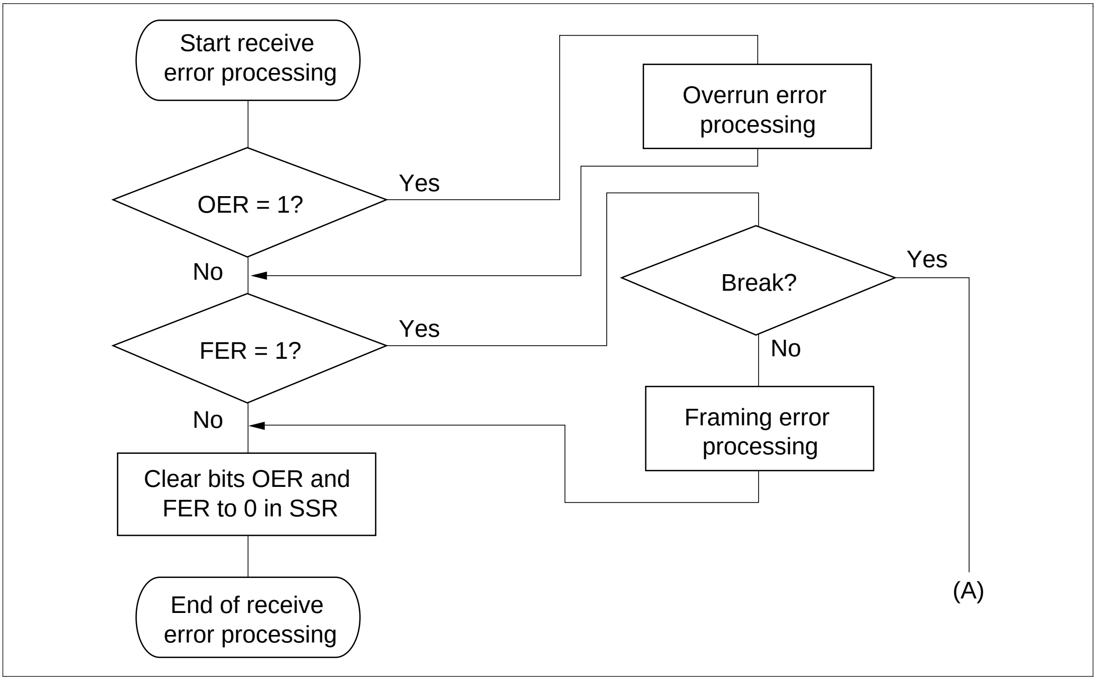


Figure 10.19 Example of Multiprocessor Data Reception Flowchart (cont)

Figure 10.20 shows an example of the operation when receiving using the multiprocessor format.

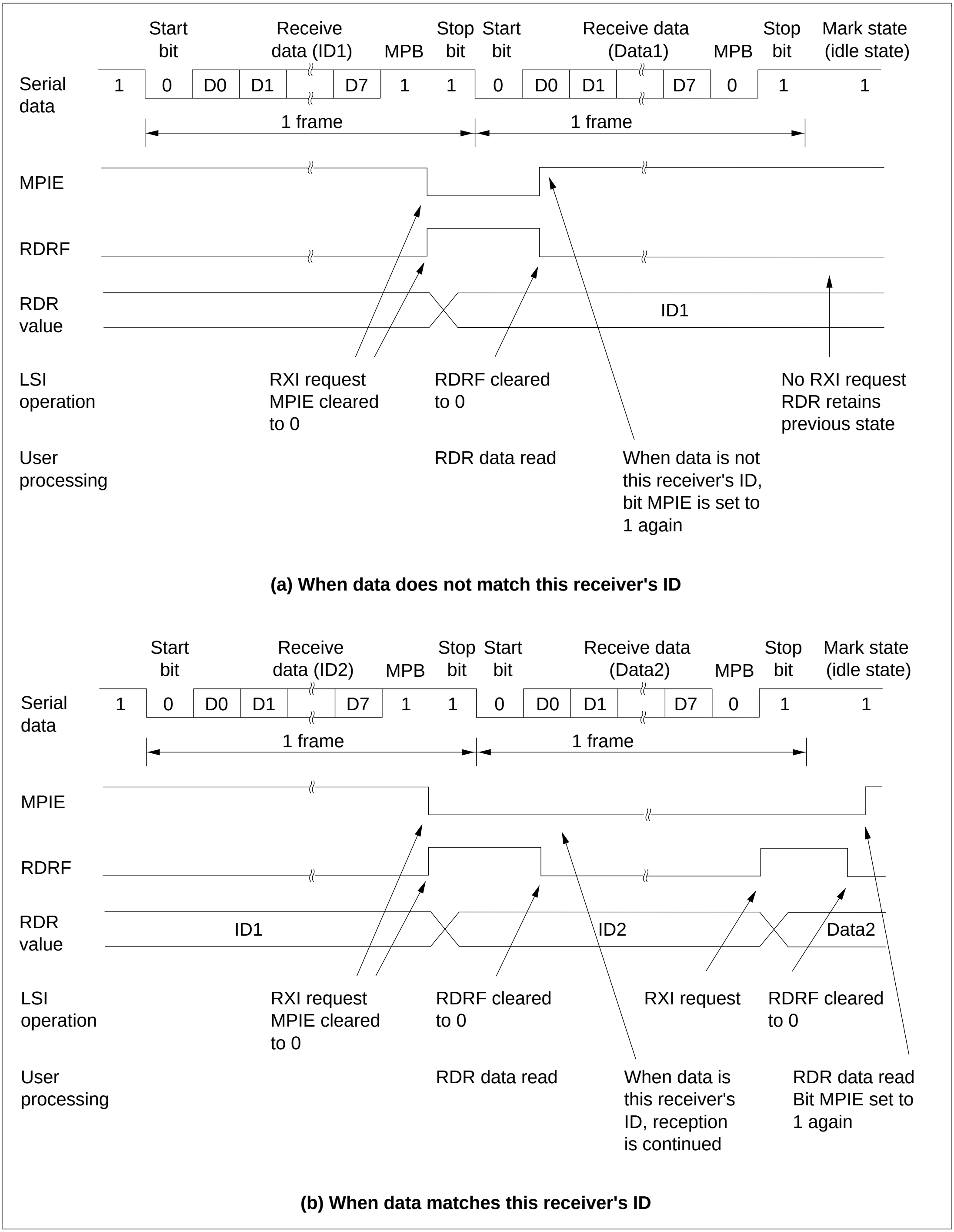


Figure 10.20 Example of Operation when Receiving using Multiprocessor Format (8-bit data, multiprocessor bit, 1 stop bit)

10.4 Interrupts

SCI3 can generate six kinds of interrupts: transmit end, transmit data empty, receive data full, and three receive error interrupts (overrun error, framing error, and parity error). These interrupts have the same vector address.

The various interrupt requests are shown in table 10.13.

Table 10.13 SCI3 Interrupt Requests

Interrupt Abbreviation	Interrupt Request	Vector Address
RXI	Interrupt request initiated by receive data full flag (RDRF)	H'0024
TXI	Interrupt request initiated by transmit data empty flag (TDRE)	
TEI	Interrupt request initiated by transmit end flag (TEND)	
ERI	Interrupt request initiated by receive error flag (OER, FER, PER)	

Each interrupt request can be enabled or disabled by means of bits TIE and RIE in SCR3.

When bit TDRE is set to 1 in SSR, a TXI interrupt is requested. When bit TEND is set to 1 in SSR, a TEI interrupt is requested. These two interrupts are generated during transmission.

The initial value of bit TDRE in SSR is 1. Therefore, if the transmit data empty interrupt request (TXI) is enabled by setting bit TIE to 1 in SCR3 before transmit data is transferred to TDR, a TXI interrupt will be requested even if the transmit data is not ready.

Also, the initial value of bit TEND in SSR is 1. Therefore, if the transmit end interrupt request (TEI) is enabled by setting bit TEIE to 1 in SCR3 before transmit data is transferred to TDR, a TEI interrupt will be requested even if the transmit data has not been sent.

Effective use of these interrupt requests can be made by having processing that transfers transmit data to TDR carried out in the interrupt service routine.

To prevent the generation of these interrupt requests (TXI and TEI), on the other hand, the enable bits for these interrupt requests (bits TIE and TEIE) should be set to 1 after transmit data has been transferred to TDR.

When bit RDRF is set to 1 in SSR, an RXI interrupt is requested, and if any of bits OER, PER, and FER is set to 1, an ERI interrupt is requested. These two interrupt requests are generated during reception.

For further details, see 3.3, Interrupts.

10.5 Application Notes

The following points should be noted when using SCI3.

1. Relation between writes to TDR and bit TDRE

Bit TDRE in the serial status register (SSR) is a status flag that indicates that data for serial transmission has not been prepared in TDR. When data is written to TDR, bit TDRE is cleared to 0 automatically. When SCI3 transfers data from TDR to TSR, bit TDRE is set to 1.

Data can be written to TDR irrespective of the state of bit TDRE, but if new data is written to TDR while bit TDRE is cleared to 0, the data previously stored in TDR will be lost of it has not yet been transferred to TSR. Accordingly, to ensure that serial transmission is performed dependably, you should first check that bit TDRE is set to 1, then write the transmit data to TDR once only (not two or more times).

2. Operation when a number of receive errors occur simultaneously

If a number of receive errors are detected simultaneously, the status flags in SSR will be set to the states shown in table 10.14. If an overrun error is detected, data transfer from RSR to RDR will not be performed, and the receive data will be lost.

Table 10.14 SSR Status Flag States and Receive Data Transfer

SSR Status Flags				Receive Data Transfer	
RDRF*	OER	FER	PER	RSR → RDR	Receive Error Status
1	1	0	0	X	Overrun error
0	0	1	0	O	Framing error
0	0	0	1	O	Parity error
1	1	1	0	X	Overrun error + framing error
1	1	0	1	X	Overrun error + parity error
0	0	1	1	O	Framing error + parity error
1	1	1	1	X	Overrun error + framing error + parity error

O : Receive data is transferred from RSR to RDR.

X : Receive data is not transferred from RSR to RDR.

Note: * Bit RDRF retains its state prior to data reception. However, note that if RDR is read after an overrun error has occurred in a frame because reading of the receive data in the previous frame was delayed, RDRF will be cleared to 0.

3. Break detection and processing

When a framing error is detected, a break can be detected by reading the value of the RXD_{32} pin directly. In a break, the input from the RXD_{32} pin becomes all 0s, with the result that bit FER is set and bit PER may also be set.

SCI3 continues the receive operation even after receiving a break. Note, therefore, that even though bit FER is cleared to 0 it will be set to 1 again.

4. Mark state and break detection

When bit TE is cleared to 0, the TXD_{32} pin functions as an I/O port whose input/output direction and level are determined by PDR and PCR. This fact can be used to set the TXD_{32} pin to the mark state, or to detect a break during transmission.

To keep the communication line in the mark state (1 state) until bit TE is set to 1, set PCR = 1 and PDR = 1. Since bit TE is cleared to 0 at this time, the TXD_{32} pin functions as an I/O port and 1 is output.

To detect a break, clear bit TE to 0 after setting PCR = 1 and PDR = 0.

When bit TE is cleared to 0, the transmission unit is initialized regardless of the current transmission state, the TXD_{32} pin functions as an I/O port, and 0 is output from the TXD_{32} pin.

5. Receive error flags and transmit operation (synchronous mode only)

When a receive error flag (OER, PER, or FER) is set to 1, transmission cannot be started even if bit TDRE is cleared to 0. The receive error flags must be cleared to 0 before starting transmission.

Note also that receive error flags cannot be cleared to 0 even if bit RE is cleared to 0.

6. Receive data sampling timing and receive margin in asynchronous mode

In asynchronous mode, SCI3 operates on a basic clock with a frequency 16 times the transfer rate. When receiving, SCI3 performs internal synchronization by sampling the falling edge of the start bit with the basic clock. Receive data is latched internally at the 8th rising edge of the basic clock. This is illustrated in figure 10.21.

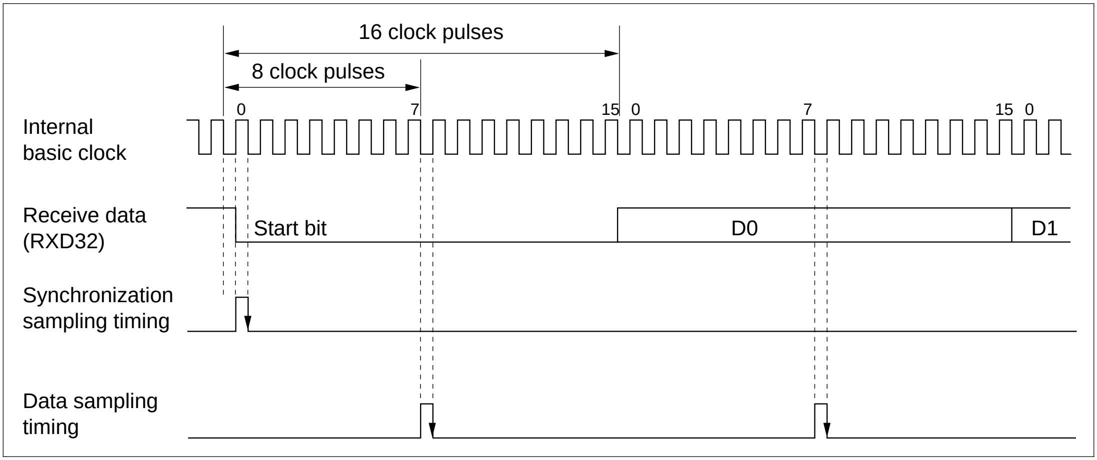


Figure 10.21 Receive Data Sampling Timing in Asynchronous Mode

Consequently, the receive margin in asynchronous mode can be expressed as shown in equation (1).

$$M = \{(0.5 - \frac{1}{2N}) - \frac{D - 0.5}{N} - (L - 0.5) F\} \times 100\ [\%] \qquad \text{..... Equation (1)}$$

where

M: Receive margin (%)
N: Ratio of bit rate to clock (N = 16)
D: Clock duty (D = 0.5 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute value of clock frequency deviation

Substituting 0 for F (absolute value of clock frequency deviation) and 0.5 for D (clock duty) in equation (1), a receive margin of 46.875% is given by equation (2).

When D = 0.5 and F = 0,

$$M = \{0.5 - 1/(2 \times 16)\} \times 100\ [\%]$$
$$= 46.875\% \qquad \text{..... Equation (2)}$$

However, this is only a computed value, and a margin of 20% to 30% should be allowed when carrying out system design.

7. Relation between RDR reads and bit RDRF

In a receive operation, SCI3 continually checks the RDRF flag. If bit RDRF is cleared to 0 when reception of one frame ends, normal data reception is completed. If bit RDRF is set to 1, this indicates that an overrun error has occurred.

When the contents of RDR are read, bit RDRF is cleared to 0 automatically. Therefore, if bit RDR is read more than once, the second and subsequent read operations will be performed while bit RDRF is cleared to 0. Note that, when an RDR read is performed while bit RDRF is cleared to 0, if the read operation coincides with completion of reception of a frame, the next frame of data may be read. This is illustrated in figure 10.22.

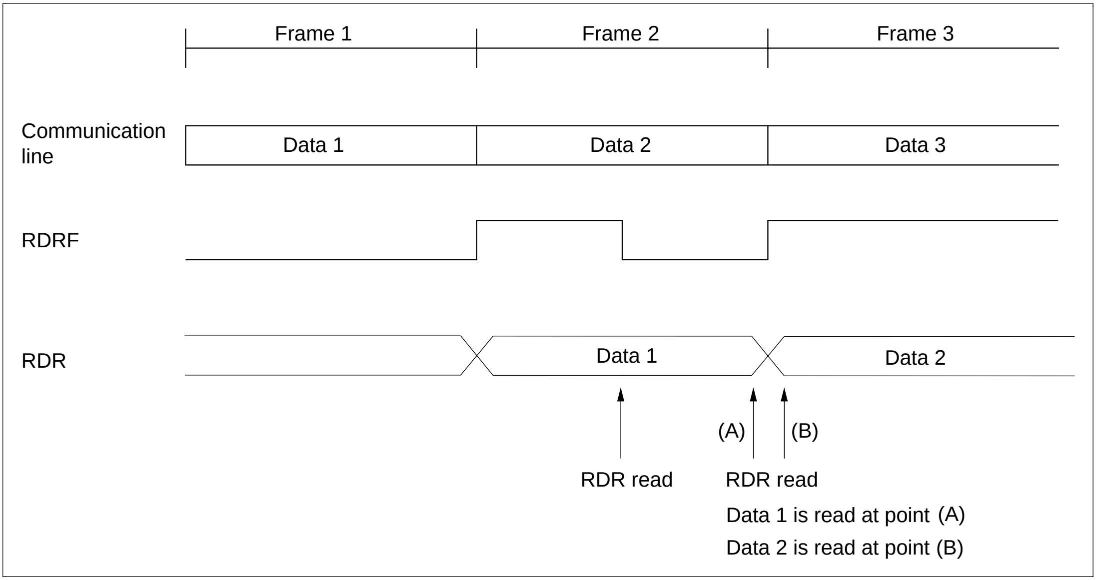


Figure 10.22 Relation between RDR Read Timing and Data

In this case, only a single RDR read operation (not two or more) should be performed after first checking that bit RDRF is set to 1. If two or more reads are performed, the data read the first time should be transferred to RAM, etc., and the RAM contents used. Also, ensure that there is sufficient margin in an RDR read operation before reception of the next frame is completed. To be precise in terms of timing, the RDR read should be completed before bit 7 is transferred in synchronous mode, or before the STOP bit is transferred in asynchronous mode.

8. Transmit and receive operations when making a state transition

Make sure that transmit and receive operations have completely finished before carrying out state transition processing.

9. Switching SCK_{32} function

If pin SCK_{32} is used as a clock output pin by SCI3 in synchronous mode and is then switched to a general input/output pin (a pin with a different function), the pin outputs a low level signal for half a system clock (ø) cycle immediately after it is switched.

This can be prevented by either of the following methods according to the situation.

a. When an SCK_{32} function is switched from clock output to non clock-output

When stopping data transfer, issue one instruction to clear bits TE and RE to 0 and to set bits CKE1 and CKE0 in SCR3 to 1 and 0, respectively. In this case, bit COM in SMR should be left 1. The above prevents SCK_{32} from being used as a general input/output pin. To avoid an intermediate level of voltage from being applied to SCK_{32}, the line connected to SCK_{32} should be pulled up to the V_{CC} level via a resistor, or supplied with output from an external device.

b. When an SCK_{32} function is switched from clock output to general input/output

When stopping data transfer,

(i) Issue one instruction to clear bits TE and RE to 0 and to set bits CKE1 and CKE0 in SCR3 to 1 and 0, respectively.

(ii) Clear bit COM in SMR to 0

(iii) Clear bits CKE1 and CKE0 in SCR3 to 0

Note that special care is also needed here to avoid an intermediate level of voltage from being applied to SCK_{32}.

10. Set up at subactive or subsleep mode

At subactive or subsleep mode, SCI3 becomes possible use only at CPU clock is øw/2.

Section 11 10-Bit PWM

11.1 Overview

The H8/3802 Series is provided with two on-chip 10-bit PWMs (pulse width modulators), designated PWM1 and PWM2, with identical functions. The PWMs can be used as D/A converters by connecting a low-pass filter. In this section the suffix m (m = 1 or 2) is used with register names, etc., as in PWDRLm, which denotes the PWDRL registers for each PWM.

11.1.1 Features

Features of the 10-bit PWMs are as follows.

- Choice of four conversion periods

  Any of the following conversion periods can be chosen:

  4,096/ø, with a minimum modulation width of 4/ø

  2,048/ø, with a minimum modulation width of 2/ø

  1,024/ø, with a minimum modulation width of 1/ø

  512/ø, with a minimum modulation width of 1/2 ø
- Pulse division method for less ripple
- Use of module standby mode enables this module to be placed in standby mode independently when not used.

11.1.2 Block Diagram

Figure 11.1 shows a block diagram of the 10-bit PWM.

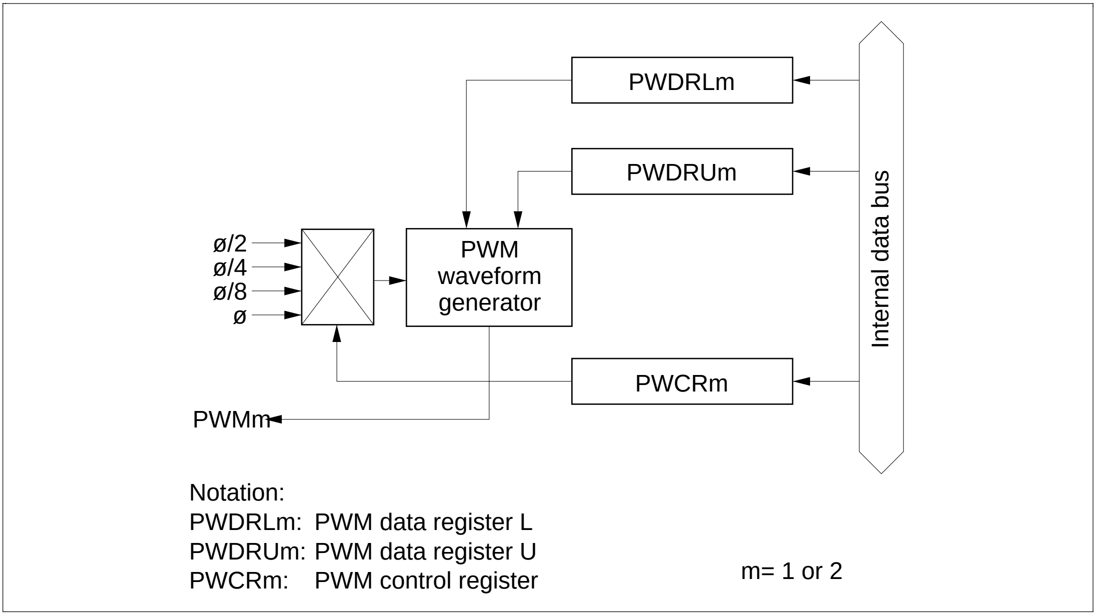


Figure 11.1 Block Diagram of the 10 bit PWM

11.1.3 Pin Configuration

Table 11.1 shows the output pin assigned to the 10-bit PWM.

Table 11.1 Pin Configuration

Name	Abbrev.	I/O	Function
PWM1 output pin	PWM1	Output	Pulse-division PWM waveform output (PWM1)
PWM2 output pin	PWM2	Output	Pulse-division PWM waveform output (PWM2)

11.1.4 Register Configuration

Table 11.2 shows the register configuration of the 10-bit PWM.

Table 11.2 Register Configuration

Name	Abbrev.	R/W	Initial Value	Address
PWM1 control register	PWCR1	W	H'FC	H'FFD0
PWM1 data register U	PWDRU1	W	H'FC	H'FFD1
PWM1 data register L	PWDRL1	W	H'00	H'FFD2
PWM2 control register	PWCR2	W	H'FC	H'FFCD
PWM2 data register U	PWDRU2	W	H'FC	H'FFCE
PWM2 data register L	PWDRL2	W	H'00	H'FFCF
Clock stop register 2	CKSTPR2	R/W	H'FF	H'FFFB

11.2 Register Descriptions

11.2.1 PWM Control Register (PWCRm)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	PWCRm1	PWCRm0
Initial value	1	1	1	1	1	1	0	0
Read/Write	—	—	—	—	—	—	W	W

PWCRm is an 8-bit write-only register for input clock selection.

Upon reset, PWCRm is initialized to H'FC.

Bits 7 to 2: Reserved bits

Bits 7 to 2 are reserved; they are always read as 1, and cannot be modified.

Bits 1 and 0: Clock select 1 and 0 (PWCRm1, PWCRm0)

Bits 1 and 0 select the clock supplied to the 10-bit PWM. These bits are write-only bits; they are always read as 1.

Bit 1 PWCRm1	Bit 0 PWCRm0	Description	
0	0	The input clock is ø (tø* = 1/ø) The conversion period is 512/ø, with a minimum modulation width of 1/2ø	(initial value)
0	1	The input clock is ø/2 (tø* = 2/ø) The conversion period is 1,024/ø, with a minimum modulation width of 1/ø	
1	0	The input clock is ø/4 (tø* = 4/ø) The conversion period is 2,048/ø, with a minimum modulation width of 2/ø	
1	1	The input clock is ø/8 (tø* = 8/ø) The conversion period is 4,096/ø, with a minimum modulation width of 4/ø	

Note: * Period of PWM input clock.

11.2.2 PWM Data Registers U and L (PWDRUm, PWDRLm)

PWDRUm

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	$PWDRUm_1$	$PWDRUm_0$
Initial value	1	1	1	1	1	1	0	0
Read/Write	—	—	—	—	—	—	W	W

PWDRLm

Bit	7	6	5	4	3	2	1	0
	$PWDRLm_7$	$PWDRLm_6$	$PWDRLm_5$	$PWDRLm_4$	$PWDRLm_3$	$PWDRLm_2$	$PWDRLm_1$	$PWDRLm_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PWDRUm and PWDRLm form a 10-bit write-only register, with the upper 2 bits assigned to PWDRUm and the lower 8 bits to PWDRLm. The value written to PWDRUm and PWDRLm gives the total high-level width of one PWM waveform cycle.

When 10-bit data is written to PWDRUm and PWDRLm, the register contents are latched in the PWM waveform generator, updating the PWM waveform generation data. The 10-bit data should always be written in the following sequence:

1. Write the lower 8 bits to PWDRLm.

2. Write the upper 2 bits to PWDRUm for the same channel.

PWDRUm and PWDRLm are write-only registers. If they are read, all bits are read as 1.

Upon reset, PWDRUm is initialized to H'FC, and PWDRLm to H'00.

11.2.3 Clock Stop Register 2 (CKSTPR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	PW2CKSTP	AECKSTP	—	PW1CKSTP	LDCKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	R/W	R/W	—	R/W	R/W

CKSTPR2 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bit relating to the PWM is described here. For details of the other bits, see the sections on the relevant modules.

Bits 4 and 1: PWM module standby mode control (PWmCKSTP)

Bits 4 and 1 control setting and clearing of module standby mode for the PWMm.

PWmCKSTP	Description
0	PWMm is set to module standby mode
1	PWMm module standby mode is cleared (initial value)

11.3 Operation

11.3.1 Operation

When using the 10-bit PWM, set the registers in the following sequence.

1. Set PWM1 or PWM2 in PMR9 to 1 for the PWM channel to be used, so that pin $P9_0$/PWM1 or $P9_1$/PWM2 is designated as the PWM output pin.
2. Set bits PWCRm1 and PWCRm0 in the PWM control register (PWCRm) to select a conversion period of 4,096/ø (PWCRm1 = 1, PWCRm0 = 1), 2,048/ø (PWCRm1 = 1, PWCRm0 = 0), 1,024/ø (PWCRm1 = 0, PWCRm0 = 1), or 512/ø (PWCRm1 = 0, PWCRm0 = 0).
3. Set the output waveform data in PWDRUm and PWDRLm. Be sure to write in the correct sequence, first PWDRLm then PWDRUm for the same channel. When data is written to PWDRUm, the data will be latched in the PWM waveform generator, updating the PWM waveform generation in synchronization with internal signals.

   One conversion period consists of 4 pulses, as shown in figure 11.2. The total of the high-level pulse widths during this period (T_H) corresponds to the data in PWDRUm and PWDRLm. This relation can be represented as follows.

$$T_H = (\text{data value in PWDRUm and PWDRLm} + 4) \times t_ø/2$$

where tø is the PWM input clock period: 1/ø (PWCRm = H'0), 2/ø (PWCRm = H'1), 4/ø (PWCRm = H'2), or 8/ø (PWCRm = H'3).

Example: Settings in order to obtain a conversion period of 1,024 µs:
When PWCRm1 = 0 and PWCRm0 = 0, the conversion period is 512/ø, so ø must be 0.5 MHz. In this case, tfn = 256 µs, with 1/2ø (resolution) = 1.0 µs.
When PWCRm1 = 0 and PWCRm0 = 1, the conversion period is 1,024/ø, so ø must be 1 MHz. In this case, tfn = 256 µs, with 1/ø (resolution) = 1.0 µs.
When PWCRm1 = 1 and PWCRm0 = 0, the conversion period is 2,048/ø , so ø must be 2 MHz. In this case, tfn = 256 µs, with 2/ø (resolution) = 1.0 µs.
When PWCRm1 = 1 and PWCRm0 = 1, the conversion period is 4,096/ø, so ø must be 4 MHz.
In this case, t_{fn} = 256 µs, with 4/ø (resolution) = 1.0 µs
Accordingly, for a conversion period of 1,024 µs, the system clock frequency (ø) must be 0.5 MHz, 1 MHz, 2 Mhz, or 4MHz.

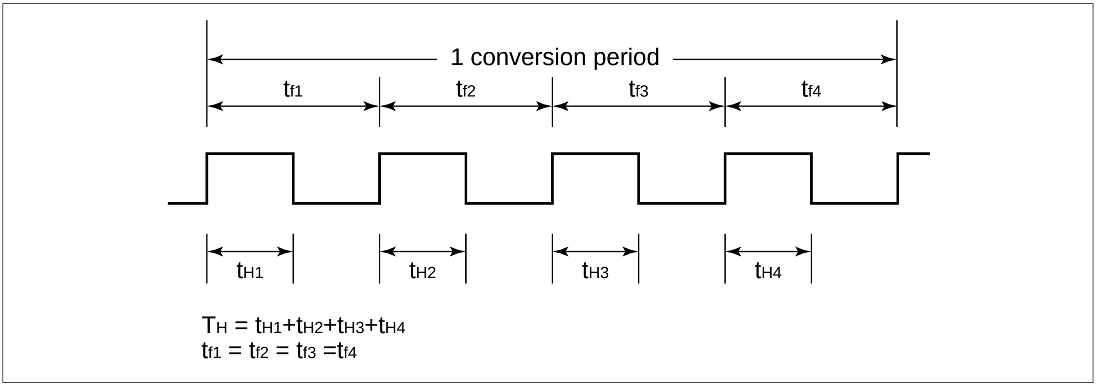


Figure 11.2 PWM Output Waveform

11.3.2 PWM Operation Modes

PWM operation modes are shown in table 11.3.

Table 11.3 PWM Operation Modes

Operation Mode	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby	Module Standby
PWCRm	Reset	Functions	Functions	Held	Held	Held	Held	Held
PWDRUm	Reset	Functions	Functions	Held	Held	Held	Held	Held
PWDRLm	Reset	Functions	Functions	Held	Held	Held	Held	Held

Section 12 A/D Converter

12.1 Overview

The H8/3802 Series includes on-chip a resistance-ladder-based successive-approximation analog-to-digital converter, and can convert up to 4 channels of analog input.

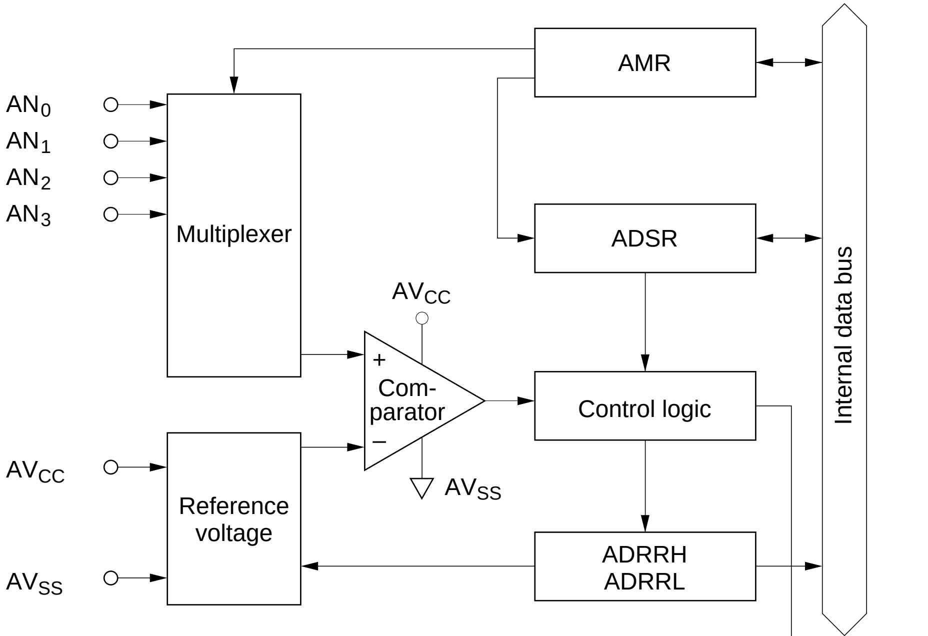
12.1.1 Features

The A/D converter has the following features.

- 10-bit resolution
- Four 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
- Use of module standby mode enables this module to be placed in standby mode independently when not used.

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the A/D converter.



Notation:
AMR: A/D mode register
ADSR: A/D start register
ADRR: A/D result register
IRRAD: A/D conversion end interrupt request flag

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
Analog power supply	AV_{CC}	Input	Power supply and reference voltage of analog part
Analog ground	AV_{SS}	Input	Ground and reference voltage of analog part
Analog input 0	AN_0	Input	Analog input channel 0
Analog input 1	AN_1	Input	Analog input channel 1
Analog input 2	AN_2	Input	Analog input channel 2
Analog input 3	AN_3	Input	Analog input channel 3

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'30	H'FFC6
A/D start register	ADSR	R/W	H'7F	H'FFC7
A/D result register H	ADRRH	R	Not fixed	H'FFC4
A/D result register L	ADRRL	R	Not fixed	H'FFC5
Clock stop register 1	CKSTPR1	R/W	H'FF	H'FFFA

12.2 Register Descriptions

12.2.1 A/D Result Registers (ADRRH, ADRRL)

Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
	ADR9	ADR8	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	—	—	—	—	—	—
Initial value	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	—	—	—	—	—	—
Read/Write	R	R	R	R	R	R	R	R	R	R	—	—	—	—	—	—
	ADRRH								ADRRL							

ADRRH and ADRRL together comprise a 16-bit read-only register for holding the results of analog-to-digital conversion. The upper 8 bits of the data are held in ADRRH, and the lower 2 bits in ADRRL.

ADRRH and ADRRL can be read by the CPU at any time, but the ADRRH and ADRRL values during A/D conversion are not fixed. After A/D conversion is complete, the conversion result is stored as 10-bit data, and this data is held until the next conversion operation starts.

ADRRH and ADRRL are not cleared on reset.

12.2.2 A/D Mode Register (AMR)

Bit	7	6	5	4	3	2	1	0
	CKS	—	—	—	CH3	CH2	CH1	CH0
Initial value	0	0	1	1	0	0	0	0
Read/Write	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'30.

Bit 7: Clock select (CKS)

Bit 7 sets the A/D conversion speed.

Bit 7		Conversion Time	
CKS	**Conversion Period**	**ø = 1 MHz**	**ø = 5 MHz**
0	62/ø (initial value)	62 µs	12.4 µs
1	31/ø	31 µs	—*

Note: * Operation is not guaranteed if the conversion time is less than 12.4 µs. Set bit 7 for a value of at least 12.4 µs.

Bit 6: Reserved bit

Bit 6 is reserved; only 0 can be written to this bit.

Bits 5 and 4: Reserved bits

Bits 5 and 4 are reserved; they are always read as 1, and cannot be modified.

Bits 3 to 0: Channel select (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)
0	1	0	0	AN_0
0	1	0	1	AN_1
0	1	1	0	AN_2
0	1	1	1	AN_3
1	0	0	0	Setting prohibited
1	0	0	1	
1	0	1	0	
1	0	1	1	

*: 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), which also sets ADSF to 1. When conversion is complete, the converted data is set in ADRRH and ADRRL, 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: Indicates the completion of A/D conversion	(initial value)
	Write: Stops A/D conversion	
1	Read: Indicates A/D conversion in progress	
	Write: 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.

12.2.4 Clock Stop Register 1 (CKSTPR1)

Bit	7	6	5	4	3	2	1	0
	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	R/W	R/W	—	R/W	—	R/W

CKSTPR1 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bit relating to the A/D converter is described here. For details of the other bits, see the sections on the relevant modules.

Bit 4: A/D converter module standby mode control (ADCKSTP)

Bit 4 controls setting and clearing of module standby mode for the A/D converter.

ADCKSTP	Description	
0	A/D converter is set to module standby mode	
1	A/D converter module standby mode is cleared	(initial value)

12.3 Operation

12.3.1 A/D Conversion Operation

The A/D converter operates by successive approximations, and yields its conversion result as 10-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 A/D Converter Operation Modes

A/D converter operation modes are shown in table 12.3.

Table 12.3 A/D Converter Operation Modes

Operation Mode	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby	Module Standby
AMR	Reset	Functions	Functions	Held	Held	Held	Held	Held
ADSR	Reset	Functions	Functions	Held	Held	Held	Held	Held
ADRRH	Held*	Functions	Functions	Held	Held	Held	Held	Held
ADRRL	Held*	Functions	Functions	Held	Held	Held	Held	Held

Note: * Undefined in a power-on reset.

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 1 (pin AN1) as the analog input channel. Figure 12.2 shows the operation timing.

1. Bits CH3 to CH0 of the A/D mode register (AMR) are set to 0101, making pin AN_1 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 is stored in ADRRH and ADRRL. 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 12.3 and 12.4 show flow charts of procedures for using the A/D converter.

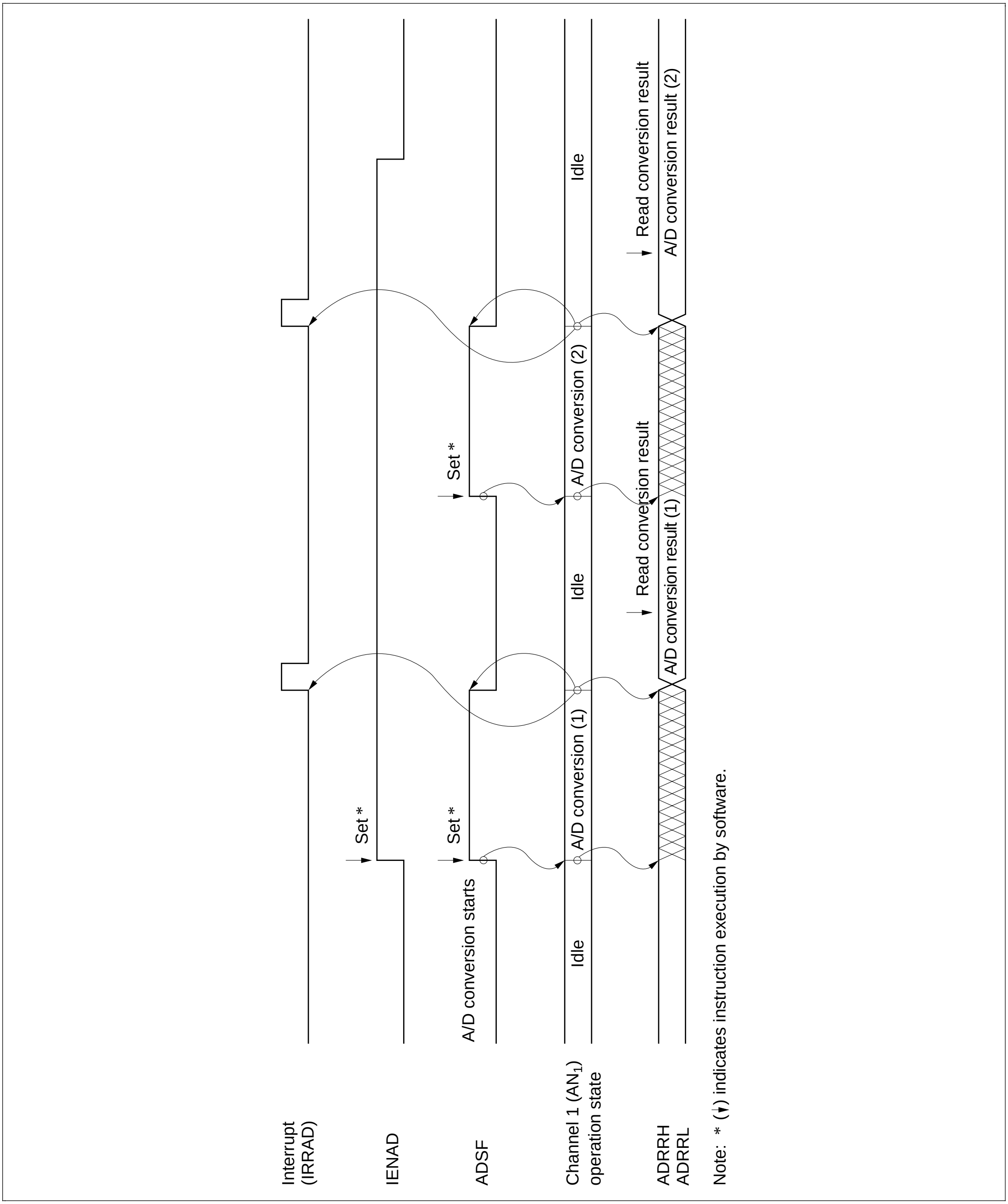


Figure 12.2 Typical A/D Converter Operation Timing

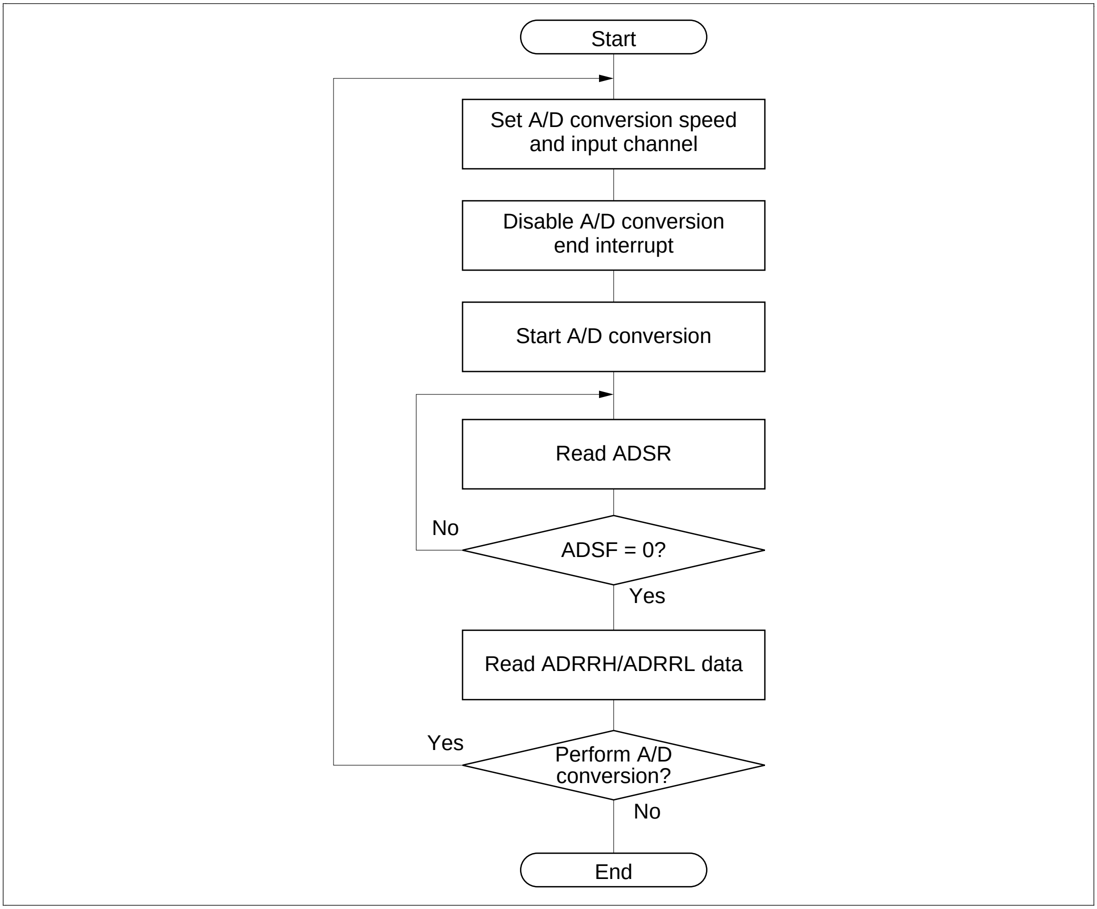


Figure 12.3 Flow Chart of Procedure for Using A/D Converter (Polling by Software)

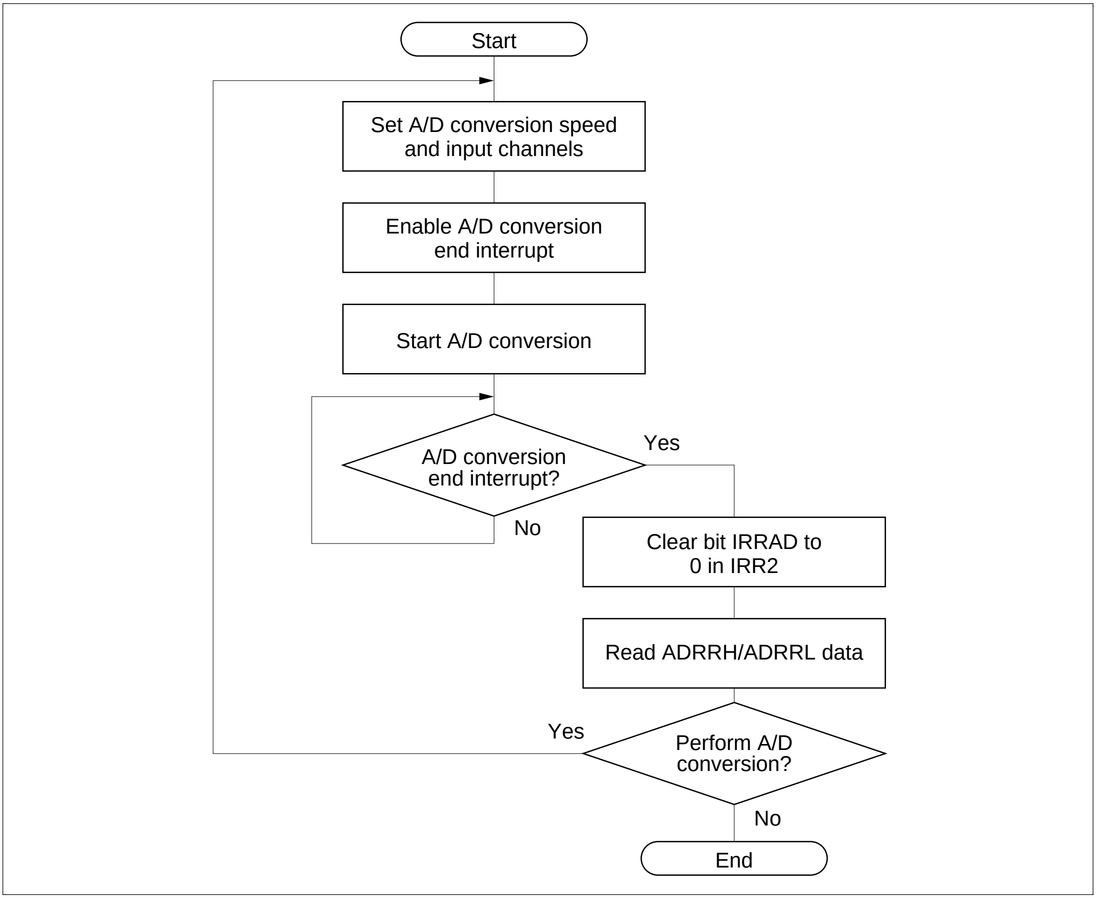


Figure 12.4 Flow Chart of Procedure for Using A/D Converter (Interrupts Used)

12.6 Application Notes

- Data in ADRRH and ADRRL 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.
- When A/D conversion is started after clearing module standby mode, wait for 10 ø clock cycles before starting.
- In active mode and sleep mode, the analog power supply current (AI_{STOP1}) flows in the ladder resistance even when the A/D converter is on standby. Therefore, if the A/D converter is not used, it is recommended that AV_{CC} be connected to the system power supply and the ADCKSTP (A/D converter module standby mode control) bit be cleared to 0 in clock stop register 1 (CKSTPR1).

Section 13 LCD Controller/Driver

13.1 Overview

The H8/3802 Series has an on-chip segment type LCD control circuit, LCD driver, and power supply circuit, enabling it to directly drive an LCD panel.

13.1.1 Features

1. Features

Features of the LCD controller/driver are given below.

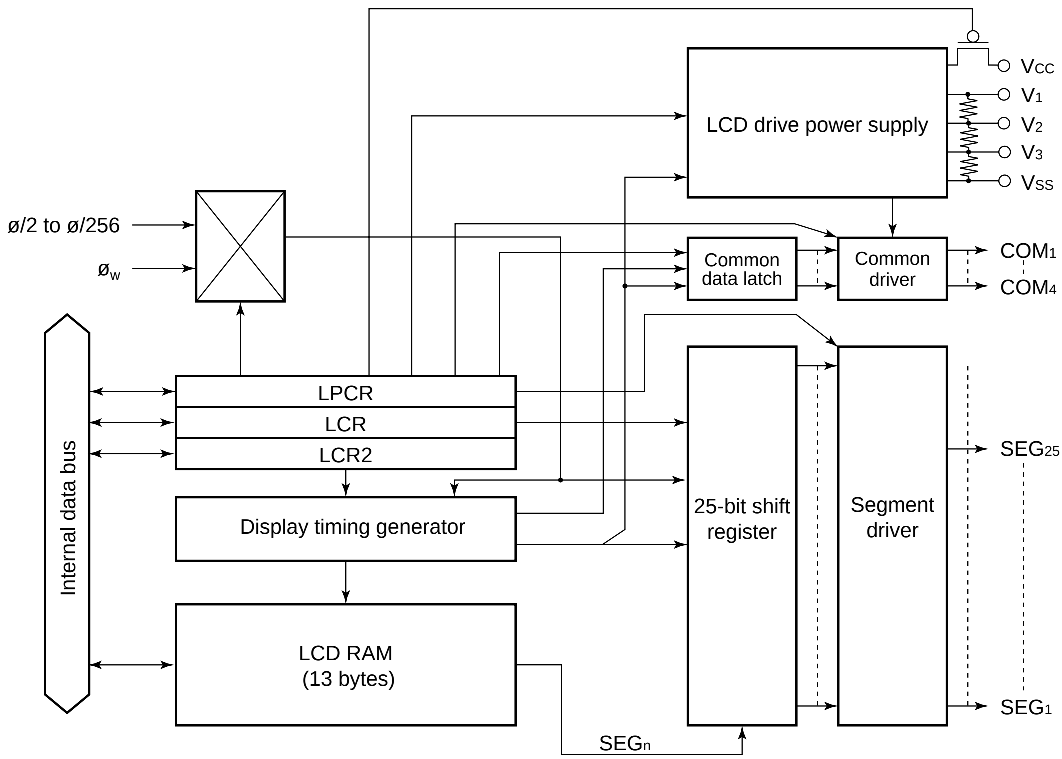
- Display capacity

Duty Cycle	Internal Driver
Static	25 seg
1/2	25 seg
1/3	25 seg
1/4	25 seg

- LCD RAM capacity
  8 bits × 13 bytes (104 bits)
- Word access to LCD RAM
- All four segment output pins can be used individually as port pins.
- Common output pins not used because of the duty cycle can be used for common double-buffering (parallel connection).
- Display possible in operating modes other than standby mode
- Choice of 11 frame frequencies
- Built-in power supply split-resistance, supplying LCD drive power
- Use of module standby mode enables this module to be placed in standby mode independently when not used.
- A or B waveform selectable by software

13.1.2 Block Diagram

Figure 13.1 shows a block diagram of the LCD controller/driver.



Notation:
LPCR: LCD port control register
LCR: LCD control register
LCR2: LCD control register 2

Figure 13.1 Block Diagram of LCD Controller/Driver

13.1.3 Pin Configuration

Table 13.1 shows the LCD controller/driver pin configuration.

Table 13.1 Pin Configuration

Name	Abbrev.	I/O	Function
Segment output pins	SEG_{25} to SEG_1	Output	LCD segment drive pins All pins are multiplexed as port pins (setting programmable)
Common output pins	COM_4 to COM_1	Output	LCD common drive pins Pins can be used in parallel with static or 1/2 duty
LCD power supply pins	V_1, V_2, V_3	—	Used when a bypass capacitor is connected externally, and when an external power supply circuit is used

13.1.4 Register Configuration

Table 13.2 shows the register configuration of the LCD controller/driver.

Table 13.2 LCD Controller/Driver Registers

Name	Abbrev.	R/W	Initial Value	Address
LCD port control register	LPCR	R/W	—	H'FFC0
LCD control register	LCR	R/W	H'80	H'FFC1
LCD control register 2	LCR2	R/W	—	H'FFC2
LCD RAM	—	R/W	Undefined	H'F740 to H'F74C
Clock stop register 2	CKSTPR2	R/W	H'FF	H'FFFB

13.2 Register Descriptions

13.2.1 LCD Port Control Register (LPCR)

Bit	7	6	5	4	3	2	1	0
	DTS1	DTS0	CMX	—	SGS3	SGS2	SGS1	SGS0
Initial value	0	0	0	—	0	0	0	0
Read/Write	R/W	R/W	R/W	W	R/W	R/W	R/W	R/W

LPCR is an 8-bit read/write register which selects the duty cycle and LCD driver pin functions.

Bits 7 to 5: Duty cycle select 1 and 0 (DTS1, DTS0), common function select (CMX)

The combination of DTS1 and DTS0 selects static, 1/2, 1/3, or 1/4 duty. CMX specifies whether or not the same waveform is to be output from multiple pins to increase the common drive power when not all common pins are used because of the duty setting.

Bit 7 DTS1	Bit 6 DTS0	Bit 5 CMX	Duty Cycle	Common Drivers	Notes
0	0	0	Static	COM_1 (initial value)	Do not use COM_4, COM_3, and COM_2.
		1		COM_4 to COM_1	COM_4, COM_3, and COM_2 output the same waveform as COM_1.
0	1	0	1/2 duty	COM_2 to COM_1	Do not use COM_4 and COM_3.
		1		COM_4 to COM_1	COM_4 outputs the same waveform as COM_3, and COM_2 outputs the same waveform as COM_1.
1	0	0	1/3 duty	COM_3 to COM_1	Do not use COM_4.
		1		COM_4 to COM_1	Do not use COM_4.
1	1	0	1/4 duty	COM_4 to COM_1	—
		1			

Bit 4: Reserved bit

Bit 4 is reserved; only 0 can be written to this bit.

Bits 3 to 0: Segment driver select 3 to 0 (SGS3 to SGS0)

Bits 3 to 0 select the segment drivers to be used.

Bit 3 SGS3	Bit 2 SGS2	Bit 1 SGS1	Bit 0 SGS0	Function of Pins SEG_{25} to SEG_1: SEG_{25}	SEG_{24} to SEG_{21}	SEG_{20} to SEG_{17}	SEG_{16} to SEG_{13}	SEG_{12} to SEG_9	SEG_8 to SEG_5	SEG_4 to SEG_1	Notes
0	0	0	0	Port	Port	Port	Port	Port	Port	Port	(Initial value)
			1	Port	Port	Port	Port	Port	Port	SEG	
		1	0	Port	Port	Port	Port	Port	SEG	SEG	
			1	Port	Port	Port	Port	SEG	SEG	SEG	
	1	0	0	Port	Port	Port	SEG	SEG	SEG	SEG	
			1	Port	Port	SEG	SEG	SEG	SEG	SEG	
		1	0	Port	SEG	SEG	SEG	SEG	SEG	SEG	
			1	SEG	SEG	SEG	SEG	SEG	SEG	SEG	
1	0	0	0	SEG	SEG	SEG	SEG	SEG	SEG	SEG	
			1	SEG	SEG	SEG	SEG	SEG	SEG	Port	
		1	0	SEG	SEG	SEG	SEG	SEG	Port	Port	
			1	SEG	SEG	SEG	SEG	Port	Port	Port	
	1	0	0	SEG	SEG	SEG	Port	Port	Port	Port	
			1	SEG	SEG	Port	Port	Port	Port	Port	
		1	0	SEG	Port	Port	Port	Port	Port	Port	
			1	Port	Port	Port	Port	Port	Port	Port	

13.2.2 LCD Control Register (LCR)

Bit	7	6	5	4	3	2	1	0
	—	PSW	ACT	DISP	CKS3	CKS2	CKS1	CKS0
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

LCR is an 8-bit read/write register which performs LCD drive power supply on/off control and display data control, and selects the frame frequency.

LCR is initialized to H'80 upon reset.

Bit 7: Reserved bit

Bit 7 is reserved; it is always read as 1 and cannot be modified.

Bit 6: LCD drive power supply on/off control (PSW)

Bit 6 can be used to turn the LCD drive power supply off when LCD display is not required in a power-down mode, or when an external power supply is used. When the ACT bit is cleared to 0, or in standby mode, the LCD drive power supply is turned off regardless of the setting of this bit.

Bit 6 PSW	Description	
0	LCD drive power supply off	(initial value)
1	LCD drive power supply on	

Bit 5: Display function activate (ACT)

Bit 5 specifies whether or not the LCD controller/driver is used. Clearing this bit to 0 halts operation of the LCD controller/driver. The LCD drive power supply is also turned off, regardless of the setting of the PSW bit. However, register contents are retained.

Bit 5 ACT	Description	
0	LCD controller/driver operation halted	(initial value)
1	LCD controller/driver operates	

Bit 4: Display data control (DISP)

Bit 4 specifies whether the LCD RAM contents are displayed or blank data is displayed regardless of the LCD RAM contents.

Bit 4 DISP	Description	
0	Blank data is displayed	(initial value)
1	LCD RAM data is display	

Bits 3 to 0: Frame frequency select 3 to 0 (CKS3 to CKS0)

Bits 3 to 0 select the operating clock and the frame frequency. In subactive mode, watch mode, and subsleep mode, the system clock (ø) is halted, and therefore display operations are not performed if one of the clocks from ø/2 to ø/256 is selected. If LCD display is required in these modes, øw, øw/2, or øw/4 must be selected as the operating clock.

Bit 3	Bit 2	Bit 1	Bit 0		Frame Frequency*2	
CKS3	**CKS2**	**CKS1**	**CKS0**	**Operating Clock**	**ø = 2 MHz**	**ø = 250 kHz*1**
0	*	0	0	øw	128 Hz*3 (initial value)	
0	*	0	1	øw/2	64 Hz*3	
0	*	1	*	øw/4	32 Hz*3	
1	0	0	0	ø/2	—	244 Hz
1	0	0	1	ø/4	977 Hz	122 Hz
1	0	1	0	ø/8	488 Hz	61 Hz
1	0	1	1	ø/16	244 Hz	30.5 Hz
1	1	0	0	ø/32	122 Hz	—
1	1	0	1	ø/64	61 Hz	—
1	1	1	0	ø/128	30.5 Hz	—
1	1	1	1	ø/256	—	—

*: Don't care

Notes: 1. This is the frame frequency in active (medium-speed, øosc/16) mode when ø = 2 MHz.
2. When 1/3 duty is selected, the frame frequency is 4/3 times the value shown.
3. This is the frame frequency when øw = 32.768 kHz.

13.2.3 LCD Control Register 2 (LCR2)

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

LCR2 is an 8-bit read/write register which controls switching between the A waveform and B waveform.

Bit 7: A waveform/B waveform switching control (LCDAB)

Bit 7 specifies whether the A waveform or B waveform is used as the LCD drive waveform.

Bit 7 LCDAB	Description	
0	Drive using A waveform	(initial value)
1	Drive using B waveform	

Bits 6 and 5: Reserved bits

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

Bits 4 to 0: Reserved bits

Bits 4 to 0 are reserved; only 0 can be written to these bits.

13.2.4 Clock Stop Register 2 (CKSTPR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	PW2CKSTP	AECKSTP	—	PW1CKSTP	LDCKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	R/W	R/W	—	R/W	R/W

CKSTPR2 is an 8-bit read/write register that performs module standby mode control for peripheral modules. Only the bit relating to the LCD controller/driver is described here. For details of the other bits, see the sections on the relevant modules.

Bit 0: LCD controller/driver module standby mode control (LDCKSTP)

Bit 0 controls setting and clearing of module standby mode for the LCD controller/driver.

Bit 0 LDCKSTP	Description	
0	LCD controller/driver is set to module standby mode	
1	LCD controller/driver module standby mode is cleared	(initial value)

13.3 Operation

13.3.1 Settings up to LCD Display

To perform LCD display, the hardware and software related items described below must first be determined.

1. Hardware settings
   a. Using 1/2 duty
   When 1/2 duty is used, interconnect pins V_2 and V_3 as shown in figure 13.2.

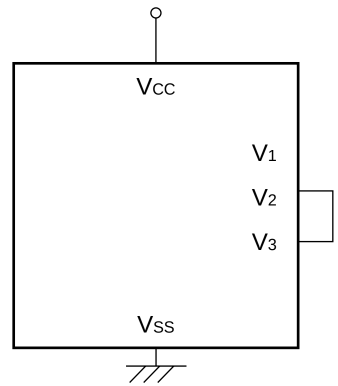


Figure 13.2 Handling of LCD Drive Power Supply when Using 1/2 Duty

   b. Large-panel display
   As the impedance of the built-in power supply split-resistance is large, it may not be suitable for driving a large panel. If the display lacks sharpness when using a large panel, refer to section 13.3.4, Boosting the LCD Drive Power Supply. When static or 1/2 duty is selected, the common output drive capability can be increased. Set CMX to 1 when selecting the duty cycle. In this mode, with a static duty cycle pins COM_4 to COM_1 output the same waveform, and with 1/2 duty the COM_1 waveform is output from pins COM_2 and COM_1, and the COM_2 waveform is output from pins COM_4 and COM_3.

2. Software settings

a. Duty selection

Any of four duty cycles—static, 1/2 duty, 1/3 duty, or 1/4 duty—can be selected with bits DTS1 and DTS0.

b. Segment selection

The segment drivers to be used can be selected with bits SGS_3 to SGS_0.

c. Frame frequency selection

The frame frequency can be selected by setting bits CKS_3 to CKS_0. The frame frequency should be selected in accordance with the LCD panel specification. For the clock selection method in watch mode, subactive mode, and subsleep mode, see 13.3.3, Operation in Power-Down Modes.

d. A or B waveform selection

Either the A or B waveform can be selected as the LCD waveform to be used by means of LCDAB.

13.3.2 Relationship between LCD RAM and Display

The relationship between the LCD RAM and the display segments differs according to the duty cycle. LCD RAM maps for the different duty cycles are shown in figures 13.3 to 13.6.

After setting the registers required for display, data is written to the part corresponding to the duty using the same kind of instruction as for ordinary RAM, and display is started automatically when turned on. Word- or byte-access instructions can be used for RAM setting.

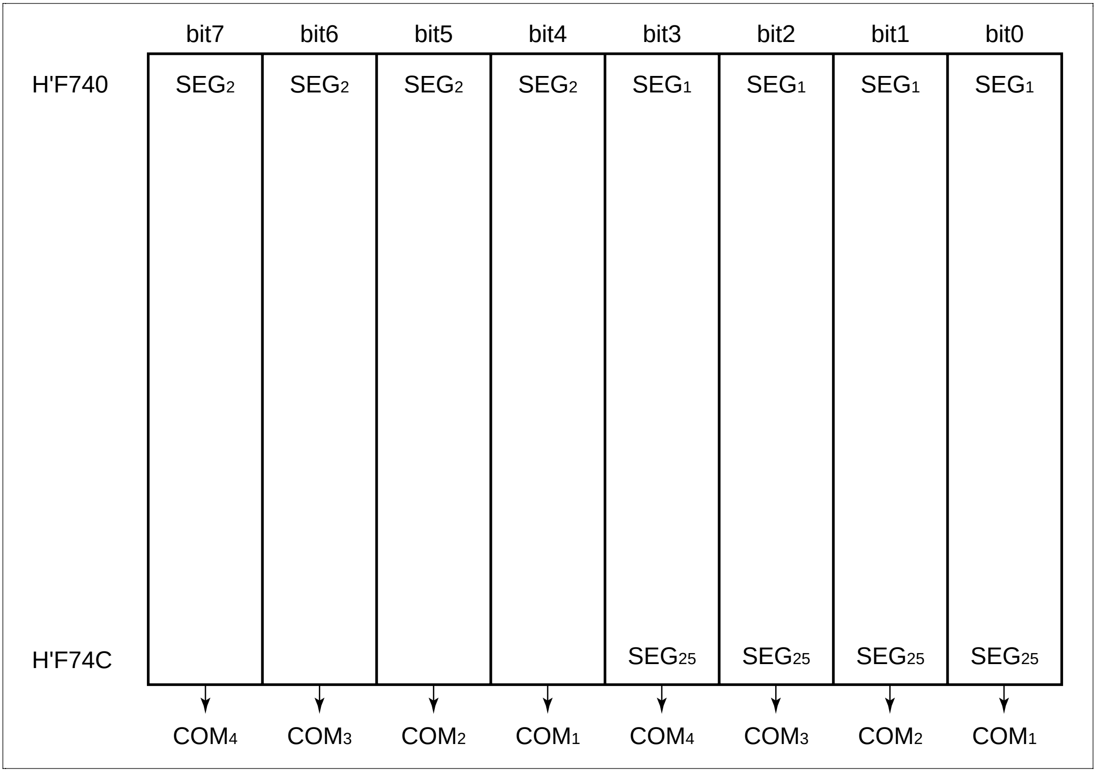


Figure 13.3 LCD RAM Map (1/4 Duty)

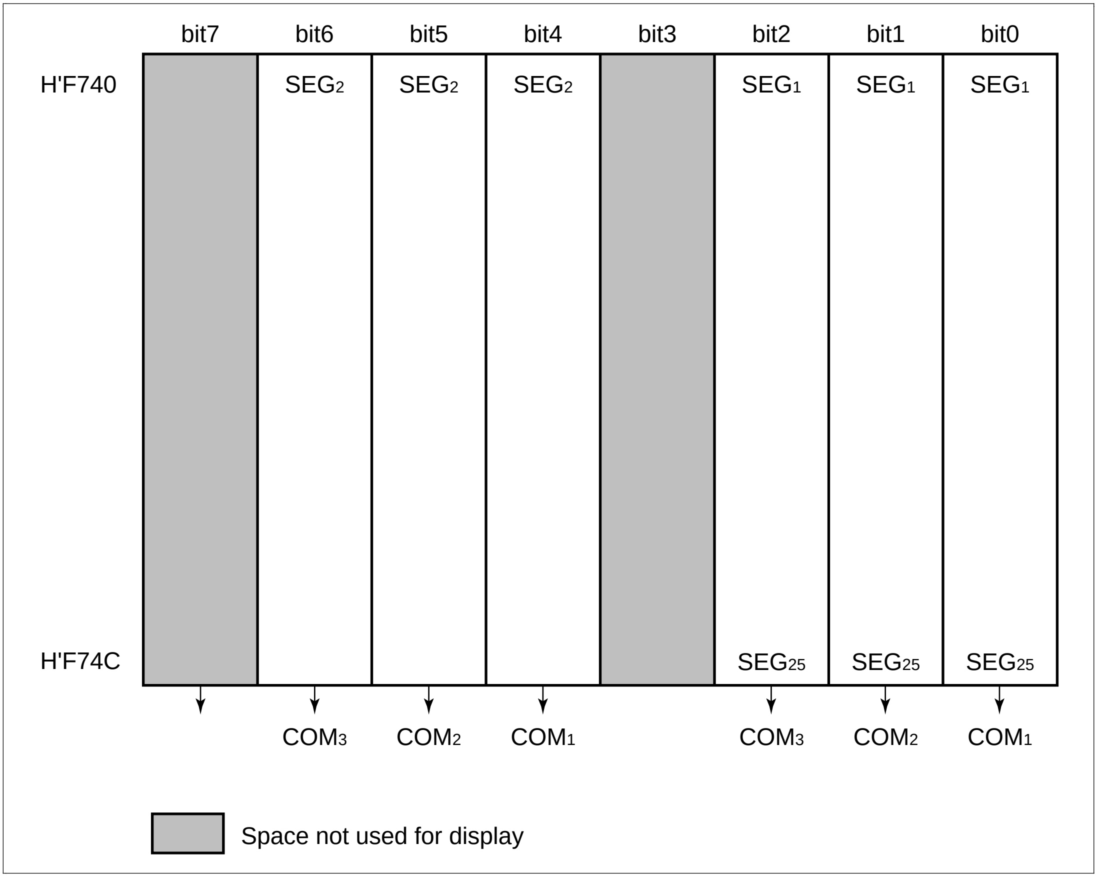


Figure 13.4 LCD RAM Map (1/3 Duty)

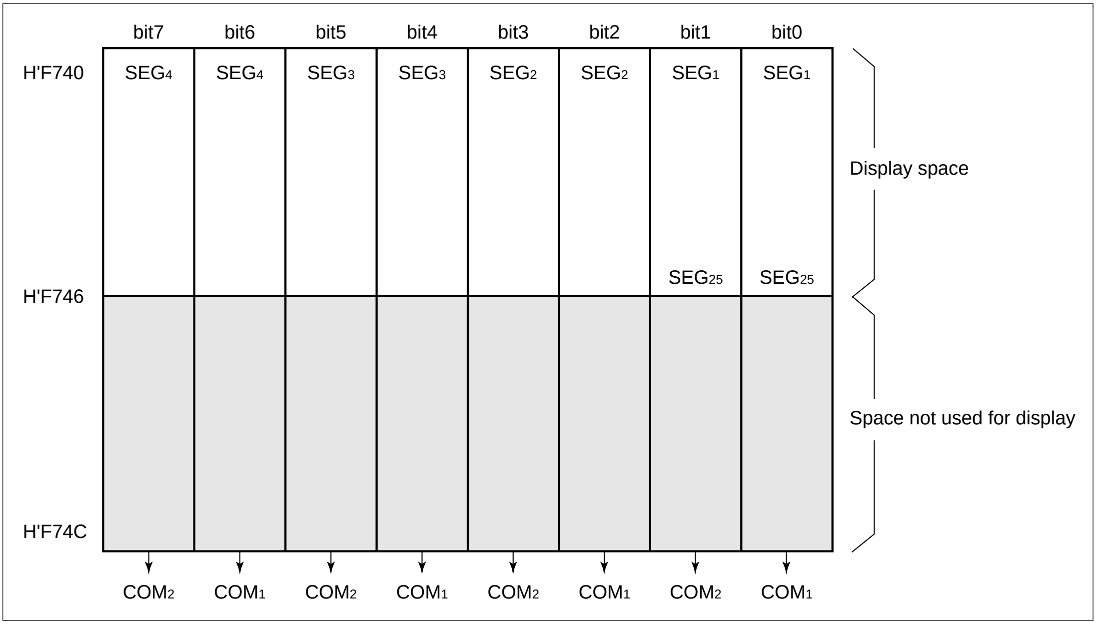


Figure 13.5 LCD RAM Map (1/2 Duty)

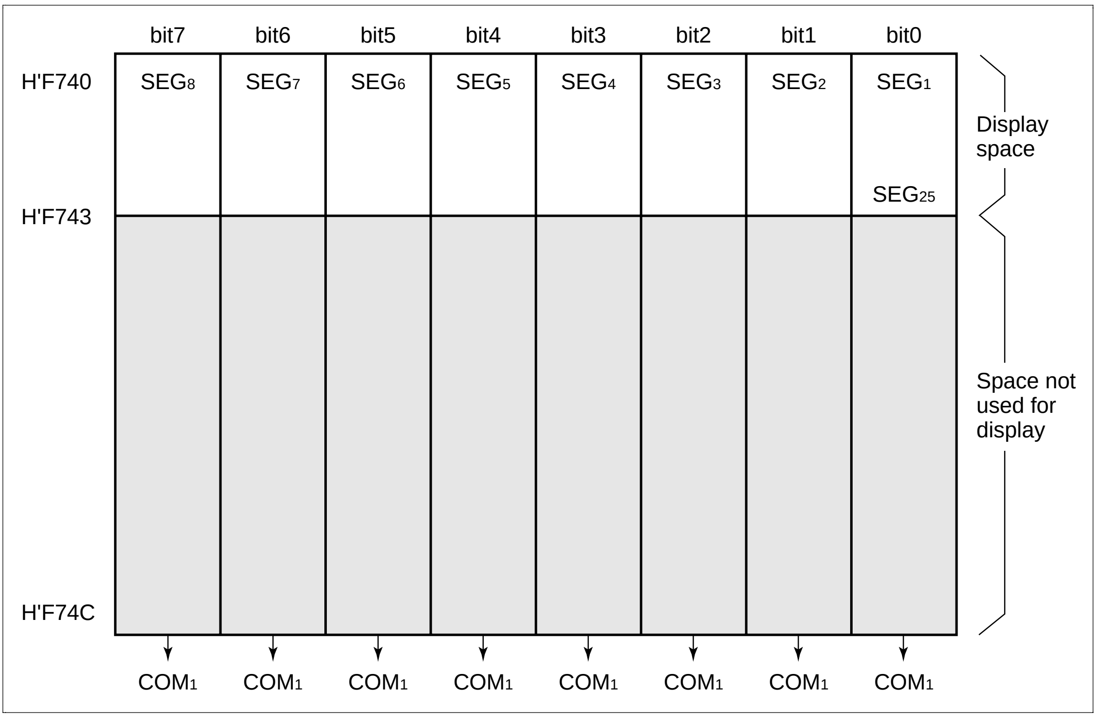


Figure 13.6 LCD RAM Map (Static Mode)

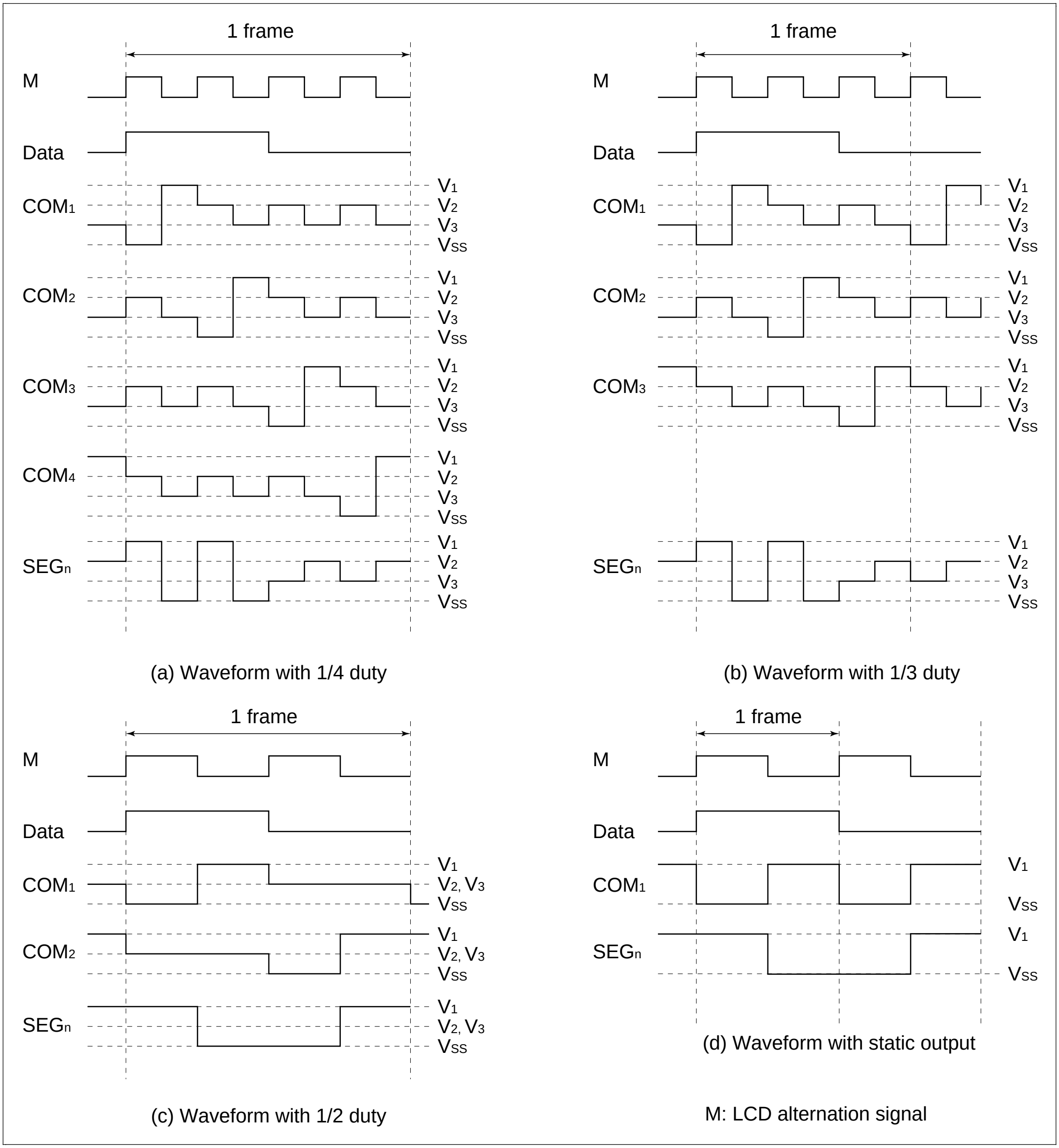


Figure 13.7 Output Waveforms for Each Duty Cycle (A Waveform)

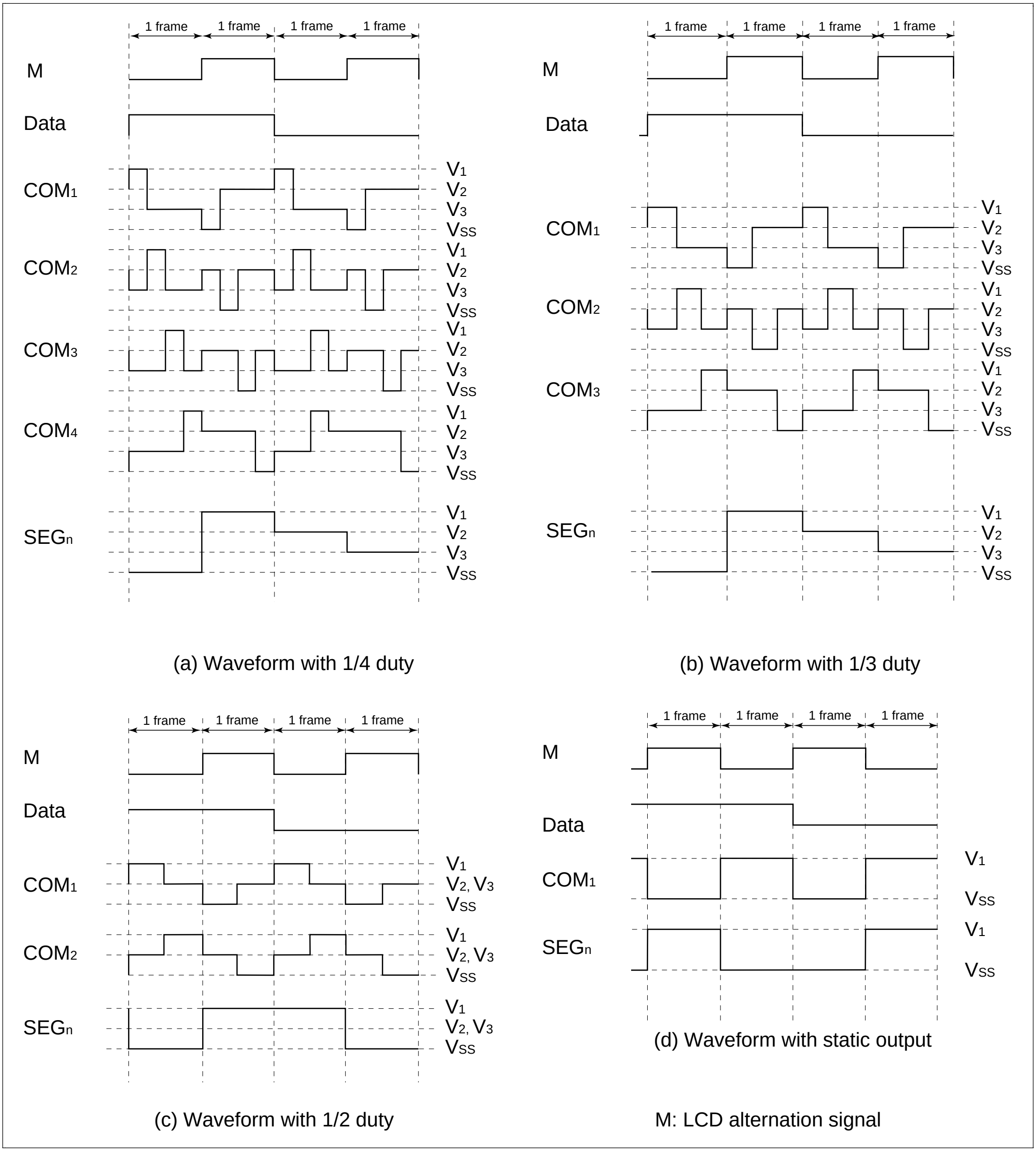


Figure 13.8 Output Waveforms for Each Duty Cycle (B Waveform)

Table 13.3 Output Levels

Data		0	0	1	1
M		**0**	**1**	**0**	**1**
Static	Common output	V_1	V_{SS}	V_1	V_{SS}
	Segment output	V_1	V_{SS}	V_{SS}	V_1
1/2 duty	Common output	V_2, V_3	V_2, V_3	V_1	V_{SS}
	Segment output	V_1	V_{SS}	V_{SS}	V_1
1/3 duty	Common output	V_3	V_2	V_1	V_{SS}
	Segment output	V_2	V_3	V_{SS}	V_1
1/4 duty	Common output	V_3	V_2	V_1	V_{SS}
	Segment output	V_2	V_3	V_{SS}	V_1

M: LCD alternation signal

13.3.3 Operation in Power-Down Modes

In the H8/3802 Series, the LCD controller/driver can be operated even in the power-down modes. The operating state of the LCD controller/driver in the power-down modes is summarized in table 13.4.

In subactive mode, watch mode, and subsleep mode, the system clock oscillator stops, and therefore, unless øw, øw/2, or øw/4 has been selected by bits CKS3 to CKS0, the clock will not be supplied and display will halt. Since there is a possibility that a direct current will be applied to the LCD panel in this case, it is essential to ensure that øw, øw/2, or øw/4 is selected. In active (medium-speed) mode, the system clock is switched, and therefore CKS3 to CKS0 must be modified to ensure that the frame frequency does not change.

Table 13.4 Power-Down Modes and Display Operation

Mode		Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby	Module Standby
Clock	ø	Runs	Runs	Runs	Stops	Stops	Stops	Stops	Stops*4
	øw	Runs	Runs	Runs	Runs	Runs	Runs	Stops*1	Stops*4
Display	ACT = 0	Stops	Stops	Stops	Stops	Stops	Stops	Stops*2	Stops
operation	ACT = 1	Stops	Functions	Functions	Functions*3	Functions*3	Functions*3	Stops*2	Stops

Notes:
1. The subclock oscillator does not stop, but clock supply is halted.
2. The LCD drive power supply is turned off regardless of the setting of the PSW bit.
3. Display operation is performed only if øw, øw/2, or øw/4 is selected as the operating clock.
4. The clock supplied to the LCD stops.

13.3.4 Boosting the LCD Drive Power Supply

When a large panel is driven, the on-chip power supply capacity may be insufficient. If the power supply capacity is insufficient when V_{CC} is used as the power supply, the power supply impedance must be reduced. This can be done by connecting bypass capacitors of around 0.1 to 0.3 µF to pins V_1 to V_3, as shown in figure 13.9, or by adding a split-resistance externally.

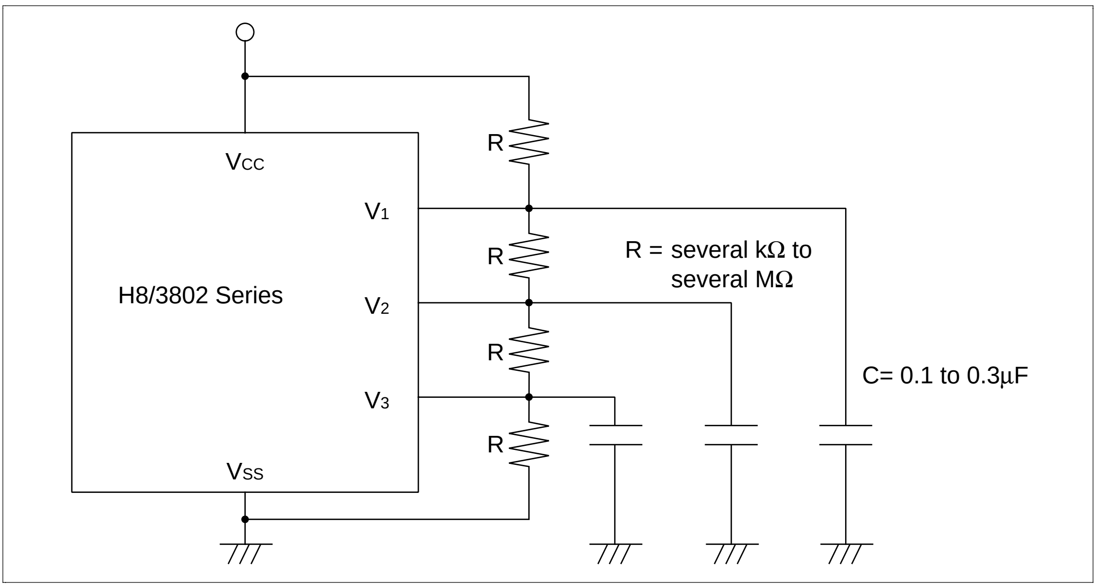


Figure 13.9 Connection of External Split-Resistance

Section 14 Electrical Characteristics

14.1 H8/3802 Series Absolute Maximum Ratings

Table 14.1 lists the absolute maximum ratings.

Table 14.1 Absolute Maximum Ratings

Item		Symbol	Value	Unit
Power supply voltage		V_{CC}	–0.3 to +7.0	V
Analog power supply voltage		AV_{CC}	–0.3 to +7.0	V
Programming voltage		V_{PP}	–0.3 to +13.0	V
Input voltage	Ports other than Port B, IRQAEC	V_{in}	–0.3 to V_{CC} +0.3	V
	Port B	AV_{in}	–0.3 to AV_{CC} +0.3	V
	IRQAEC	HV_{in}	–0.3 to +7.3	V
Operating temperature		T_{opr}	–20 to +75	°C
Storage temperature		T_{stg}	–55 to +125	°C

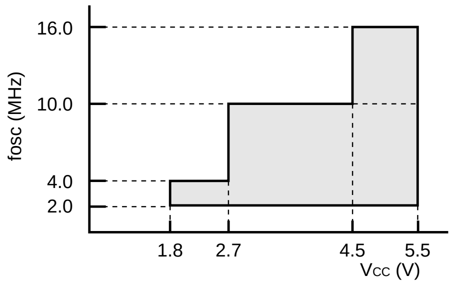
Note: Permanent damage may occur to the chip if maximum ratings are exceeded. Normal operation should be under the conditions specified in Electrical Characteristics. Exceeding these values can result in incorrect operation and reduced reliability.

14.2 H8/3802 Series Electrical Characteristics

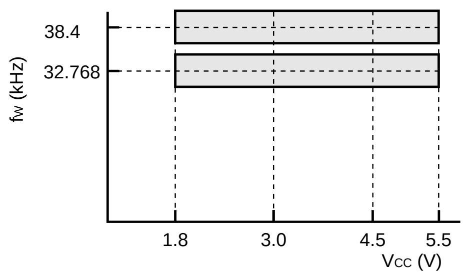
14.2.1 Power Supply Voltage and Operating Range

The power supply voltage and operating range are indicated by the shaded region in the figures.

1. Power supply voltage and oscillator frequency range

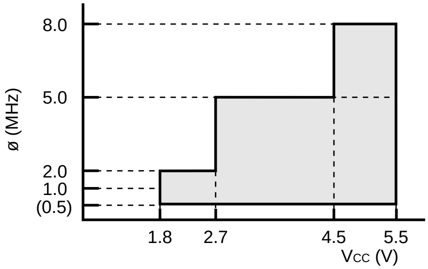


- Active (high-speed) mode
- Sleep (high-speed) mode
- Note 1: The fosc values are those when an oscillator is used; when an external clock is used the minimum value of fosc is 1 MHz.



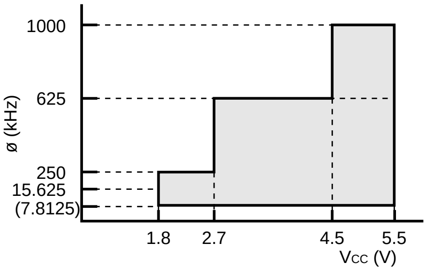
- All operating
- Note 2: When an oscillator is used for the subclock, hold V_{CC} at 2.2 V to 5.5 V from power-on until the oscillation settling time has elapsed.

2. Power supply voltage and operating frequency range



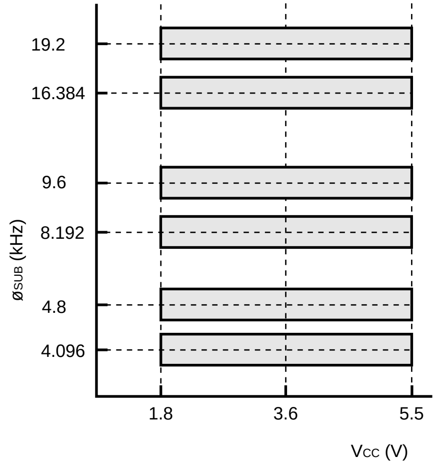
- Active (high-speed) mode
- Sleep (high-speed) mode (except CPU)

Note 1. The figure in parentheses is the minimum operating frequency when an external clock is input. When using an oscillator, the minimum operating frequency (ø) is 1 MHz.



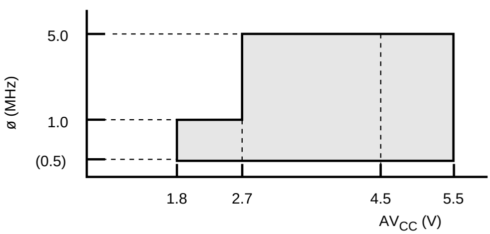
- Active (medium-speed) mode
- Sleep (medium-speed) mode (except A/D converter)

Note 2. The figure in parentheses is the minimum operating frequency when an external clock is input. When using an oscillator, the minimum operating frequency (ø) is 15.625 kHz.

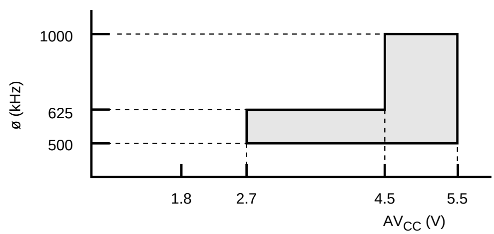


- Subactive mode
- Subsleep mode (except CPU)
- Watch mode (except CPU)

3. Analog power supply voltage and A/D converter operating range



- Active (high-speed) mode
- Sleep (high-speed) mode



- Active (medium-speed) mode
- Sleep (medium-speed) mode

Note 3: When AV_{CC} = 1.8 V to 2.7 V, the operating range is limited to ø = 1.0 MHz when using an oscillator, and is ø = 0.5 MHz to 1.0 MHz when using an external clock.

14.2.2 DC Characteristics

Table 14.2 lists the DC characteristics of the H8/3802.

Table 14.2 DC Characteristics

V_{CC} = 1.8 V to 5.5 V, AV_{CC} = 1.8 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C (including subactive mode) unless otherwise indicated.

Item	Symbol	Applicable Pins	Values Min	Values Typ	Values Max	Unit	Test Condition	Notes
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{IRQ_0}$, $\overline{IRQ_1}$, AEVL, AEVH, SCK_{32}	0.8 V_{CC}	—	V_{CC} + 0.3	V	V_{CC} = 4.0 V to 5.5 V	
			0.9 V_{CC}	—	V_{CC} + 0.3		Except the above	
		RXD_{32}	0.7 V_{CC}	—	V_{CC} + 0.3	V	V_{CC} = 4.0 V to 5.5 V	
			0.8 V_{CC}	—	V_{CC} + 0.3		Except the above	
		OSC_1	0.8 V_{CC}	—	V_{CC} + 0.3	V	V_{CC} = 4.0 V to 5.5 V	
			0.9 V_{CC}	—	V_{CC} + 0.3		Except the above	
		X_1	0.9 V_{CC}	—	V_{CC} + 0.3	V	V_{CC} = 1.8 V to 5.5 V	
		$P3_1$ to $P3_7$, $P4_0$ to $P4_3$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$, PA_0 to PA_3	0.7 V_{CC}	—	V_{CC} + 0.3	V	V_{CC} = 4.0 V to 5.5 V	
			0.8 V_{CC}	—	V_{CC} + 0.3		Except the above	
		PB_0 to PB_3	0.7 V_{CC}	—	AV_{CC} + 0.3		V_{CC} = 4.0 V to 5.5 V	
			0.8 V_{CC}	—	AV_{CC} + 0.3		Except the above	
		IRQAEC	0.8 V_{CC}	—	7.3	V	V_{CC} = 4.0 V to 5.5 V	
			0.9 V_{CC}	—	7.3		Except the above	

Note: Connect the TEST pin to V_{SS}.

Item	Symbol	Applicable Pins	Values			Unit	Test Condition	Notes
			Min	Typ	Max			
Input low voltage	V_{IL}	$\overline{RES}$,	–0.3	—	0.2 V_{CC}	V	V_{CC} = 4.0 V to 5.5 V	
		$\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{IRQ_0}$, $\overline{IRQ_1}$, IRQAEC, AEVL, AEVH, SCK_{32}	–0.3	—	0.1 V_{CC}		Except the above	
		RXD_{32},	–0.3	—	0.3 V_{CC}	V	V_{CC} = 4.0 V to 5.5 V	
			–0.3	—	0.2 V_{CC}		Except the above	
		OSC_1	–0.3	—	0.2 V_{CC}	V	V_{CC} = 4.0 V to 5.5 V	
			–0.3	—	0.1 V_{CC}		Except the above	
		X_1	–0.3	—	0.1 V_{CC}	V	V_{CC} = 1.8 V to 5.5 V	
		$P3_1$ to $P3_7$, $P4_0$ to $P4_3$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$, PA_0 to PA_3, PB_0 to PB_3	–0.3	—	0.3 V_{CC}	V	V_{CC} = 4.0 V to 5.5 V	
			–0.3	—	0.2 V_{CC}		Except the above	
Output high voltage	V_{OH}	$P3_1$ to $P3_7$, $P4_0$ to $P4_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$, PA_0 to PA_3	V_{CC} – 1.0	—	—	V	V_{CC} = 4.0 V to 5.5 V $–I_{OH}$ = 1.0 mA	
			V_{CC} – 0.5	—	—		V_{CC} = 4.0 V to 5.5 V $–I_{OH}$ = 0.5 mA	
			V_{CC} – 0.3	—	—		$–I_{OH}$ = 0.1 mA	
Output low voltage	V_{OL}	$P4_0$ to $P4_2$	—	—	0.6	V	V_{CC} = 4.0 V to 5.5 V I_{OL} = 1.6 mA	
			—	—	0.5		I_{OL} = 0.4 mA	
		$P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$, PA_0 to PA_3	—	—	0.5		I_{OL} = 0.4 mA	
		$P3_1$ to $P3_7$	—	—	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	

Item	Symbol	Applicable Pins	Values Min	Values Typ	Values Max	Unit	Test Condition	Notes
Output low voltage	V_{OL}	$P9_0$ to $P9_2$	—	—	0.5	V	V_{CC} = 2.2 V to 5.5 V, I_{OL} = 25 mA	
							I_{OL} = 15 mA	
							I_{OL} = 10 mA	*6
		$P9_3$ to $P9_5$	—	—	0.5		I_{OL} = 10 mA	
Input/output leakage current	$\mid I_{IL} \mid$	$\overline{RES}$, $P4_3$	—	—	20.0	μA	V_{IN} = 0.5 V to V_{CC} – 0.5 V	*2
			—	—	1.0			*1
		OSC_1, X_1, $P3_1$ to $P3_7$, $P4_0$ to $P4_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$, IRQAEC, $P9_0$ to $P9_5$, PA_0 to PA_3	—	—	1.0	μA	V_{IN} = 0.5 V to V_{CC} – 0.5 V	
		PB_0 to PB_3	—	—	1.0		V_{IN} = 0.5 V to AV_{CC} – 0.5 V	
Pull-up MOS current	$–I_p$	$P3_1$ to $P3_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$	50.0	—	300.0	μA	V_{CC} = 5 V, V_{IN} = 0 V	
			—	35.0	—		V_{CC} = 2.7 V, V_{IN} = 0 V	Reference value
Input capacitance	C_{IN}	All input pins except power supply, $\overline{RES}$, IRQAEC, $P4_3$, PB_0 to PB_3	—	—	15.0	pF	f = 1 MHz, V_{IN} =0 V, T_a = 25°C	
		IRQAEC	—	—	30.0			
		$\overline{RES}$	—	—	80.0			*2
			—	—	15.0			*1
		$P4_3$	—	—	50.0			*2
			—	—	15.0			*1
		PB_0 to PB_3	—	—	15.0			

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Notes
Active mode current dissipation	I_{OPE1}	V_{CC}	—	7.0	10.0	mA	Active (high-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	*3 *4
	I_{OPE2}	V_{CC}	—	2.2	3.0	mA	Active (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz $ø_{osc}$/128	*3 *4
Sleep mode current dissipation	I_{SLEEP}	V_{CC}	—	3.8	5.0	mA	V_{CC}=5 V, f_{OSC}= 10 MHz	*3 *4
Subactive mode current dissipation	I_{SUB}	V_{CC}	—	15.0	30.0	µA	V_{CC} = 2.7 V, LCD on 32 kHz crystal oscillator ($ø_{SUB}$=$ø_W$/2)	*3 *4
			—	8.0	—	µA	V_{CC} = 2.7 V, LCD on 32 kHz crystal oscillator ($ø_{SUB}$=$ø_W$/8)	*3 *4 Reference value
Subsleep mode current dissipation	I_{SUBSP}	V_{CC}	—	7.5	16.0	µA	V_{CC} = 2.7 V, LCD on 32 kHz crystal oscillator ($ø_{SUB}$=$ø_W$/2)	*3 *4
Watch mode current dissipation	I_{WATCH}	V_{CC}	—	3.8	6.0	µA	V_{CC} = 2.7 V 3 2 kHz crystal oscillator LCD not used	*2 *3 *4
				2.8				*1 *3 *4
Standby mode current dissipation	I_{STBY}	V_{CC}	—	1.0	5.0	µA	32 kHz crystal oscillator not used	*3 *4
RAM data retaining voltage	V_{RAM}	V_{CC}	1.5	—	—	V		

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Notes
Allowable output low current (per pin)	I_{OL}	Output pins except port 3 and 9	—	—	2.0	mA	V_{CC} = 4.0 V to 5.5 V	
		Port 3	—	—	10.0		V_{CC} = 4.0 V to 5.5 V	
		Output pins except port 9	—	—	0.5			
Allowable output low current (per pin)	I_{OL}	$P9_0$ to $P9_2$	—	—	25.0	mA	V_{CC} = 2.2 V to 5.5 V	*5
			—	—	15.0			
			—	—	10.0			
		$P9_3$ to $P9_5$	—	—	10.0			
Allowable output low current (total)	ΣI_{OL}	Output pins except ports 3 and 9	—	—	40.0	mA	V_{CC} = 4.0 V to 5.5 V	
		Port 3	—	—	80.0		V_{CC} = 4.0 V to 5.5 V	
		Output pins except port 9	—	—	20.0			
		Port 9	—	—	80.0			
Allowable output high current (per pin)	$-I_{OH}$	All output pins	—	—	2.0	mA	V_{CC} = 4.0 V to 5.5 V	
			—	—	0.2		Except the above	
Allowable output high	$\Sigma - I_{OH}$	All output pins	—	—	15.0	mA	V_{CC} = 4.0 V to 5.5 V	
			—	—	10.0		Except the above	

Notes: 1. Applies to the Mask ROM products.

2. Applies to the HD6473802.

3. Pin states during current measurement.

Mode	$\overline{\text{RES}}$ Pin	Internal State	Other Pins	LCD Power Supply	Oscillator Pins
Active (high-speed) mode (I_{OPE1})	V_{CC}	Operates	V_{CC}	Halted	System clock oscillator: crystal Subclock oscillator: Pin X_1 = GND
Active (medium-speed) mode (I_{OPE2})					
Sleep mode	V_{CC}	Only timers operate	V_{CC}	Halted	
Subactive mode	V_{CC}	Operates	V_{CC}	Halted	System clock oscillator: crystal Subclock oscillator: crystal
Subsleep mode	V_{CC}	Only timers operate, CPU stops	V_{CC}	Halted	
Watch mode	V_{CC}	Only time base operates, CPU stops	V_{CC}	Halted	
Standby mode	V_{CC}	CPU and timers both stop	V_{CC}	Halted	System clock oscillator: crystal Subclock oscillator: Pin X_1 = GND

4. Excludes current in pull-up MOS transistors and output buffers.
5. When the PIOFF bit in the port mode register 9 is 0.
6. When the PIOFF bit in the port mode register 9 is 1.

14.2.3 AC Characteristics

Table 14.3 lists the control signal timing, and tables 14.4 lists the serial interface timing of the H8/3802.

Table 14.3 Control Signal Timing

V_{CC} = 1.8 V to 5.5 V, AV_{CC} = 1.8 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C (including subactive mode) unless otherwise indicated.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
System clock oscillation frequency	f_{OSC}	OSC_1, OSC_2	2.0	—	16.0	MHz	V_{CC} = 4.5 V to 5.5 V	
			2.0	—	10.0		V_{CC} = 2.7 V to 5.5 V	
			2.0	—	4.0		Except the above	
OSC clock ($ø_{OSC}$) cycle time	t_{OSC}	OSC_1, OSC_2	62.5	—	500 (1000)	ns	V_{CC} = 4.5 V to 5.5 V	Figure 14.1
			100	—	500 (1000)		V_{CC} = 2.7 V to 5.5 V	*2
			250	—	500 (1000)		Except the above	
System clock (ø) cycle time	t_{cyc}		2	—	128	t_{OSC}		
			—	—	128	µs		
Subclock oscillation frequency	f_W	X_1, X_2	—	32.768 or 38.4	—	kHz		
Watch clock ($ø_W$) cycle time	t_W	X_1, X_2	—	30.5 or 26.0	—	µs		Figure 14.1
Subclock ($ø_{SUB}$) cycle time	t_{subcyc}		2	—	8	t_W		*1
Instruction cycle time			2	—	—	t_{cyc} t_{subcyc}		
Oscillation stabilization time	t_{rc}	OSC_1, OSC_2	—	20	45	µs	Figure 14.7 V_{CC} = 2.2 V to 5.5 V	Figure 14.7
			—	—	50	ms	Except the above	Figure 14.7

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Oscillation stabilization time	t_{rc}	X_1, X_2	—	—	2.0	s	V_{CC} = 2.7 V to 5.5 V	*3
			—	—	10.0		V_{CC} = 2.7 V to 5.5 V	
External clock high width	t_{CPH}	OSC_1	25	—	—	ns	V_{CC} = 4.5 V to 5.5 V	Figure 14.1
			40	—	—		V_{CC} = 2.7 V to 5.5 V	
			100	—	—		Except the above	Figure 14.1
		X_1	—	15.26 or 13.02	—	µs		
External clock low width	t_{CPL}	OSC_1	25	—	—	ns	V_{CC} = 4.5 V to 5.5 V	Figure 14.1
			40	—	—		V_{CC} = 2.7 V to 5.5 V	
			100	—	—		Except the above	Figure 14.1
		X_1	—	15.26 or 13.02	—	µs		
External clock rise time	t_{CPr}	OSC_1	—	—	6	ns	V_{CC} = 4.5 V to 5.5 V	Figure 14.1
			—	—	10		V_{CC} = 2.7 V to 5.5 V	
			—	—	25		Except the above	Figure 14.1
		X_1	—	—	55.0	ns		
External clock fall time	t_{CPf}	OSC_1	—	—	6	ns	V_{CC} = 4.5 V to 5.5 V	Figure 14.1
			—	—	10		V_{CC} = 2.7 V to 5.5 V	
			—	—	25		Except the above	Figure 14.1
		X_1	—	—	55.0	ns		
Pin $\overline{RES}$ low width	t_{REL}	$\overline{RES}$	10	—	—	t_{cyc}		Figure 14.2
Input pin high width	t_{IH}	$\overline{IRQ_0}$, $\overline{IRQ_1}$, IRQAEC, $\overline{WKP_0}$ to $\overline{WKP_7}$	2	—	—	t_{cyc} t_{subcyc}		Figure 14.3
		AEVL, AEVH	0.5	—	—	t_{OSC}		
Input pin low width	t_{IL}	$\overline{IRQ_0}$ to $\overline{IRQ_1}$, IRQAEC, $\overline{WKP_0}$ to $\overline{WKP_7}$	2	—	—	t_{cyc} t_{subcyc}		Figure 14.3
		AEVL, AEVH	0.5	—	—	t_{OSC}		

Notes: 1. Selected with SA1 and SA0 of system clock control register 2 (SYSCR2).
2. The figure in parentheses applies when an external clock is used.
3. After powering on, hold V_{cc} at 2.2 V to 5.5 V until the chip's oscillation settling time has elapsed.

Table 14.4 Serial Interface (SCI3) Timing

V_{CC} = 1.8 V to 5.5 V, AV_{CC} = 1.8 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C (including subactive mode) unless otherwise indicated.

Item		Symbol	Values Min	Typ	Max	Unit	Test Conditions	Reference Figure
Input clock cycle	Asynchronous	t_{scyc}	4	—	—	t_{cyc} or t_{subcyc}		Figure 14.4
	Synchronous		6	—	—			
Input clock pulse width		t_{SCKW}	0.4	—	0.6	t_{scyc}		Figure 14.4
Transmit data delay time (synchronous)		t_{TXD}	—	—	1	t_{cyc} or t_{subcyc}	V_{CC} = 4.0 V to 5.5 V	Figure 14.5
			—	—	1		Except the above	
Receive data setup time (synchronous)		t_{RXS}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.5
			400.0	—	—		Except the above	Figure 14.5
Receive data hold time (synchronous)		t_{RXH}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	Figure 14.5
			400.0	—	—		Except the above	Figure 14.5

14.2.4 A/D Converter Characteristics

Table 14.5 shows the A/D converter characteristics of the H8/3802.

Table 14.5 A/D Converter Characteristics

V_{CC} = 1.8 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C unless otherwise indicated.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Analog power supply voltage	AV_{CC}	AV_{CC}	1.8	—	5.5	V		*1
Analog input voltage	AV_{IN}	AN_0 to AN_3	– 0.3	—	AV_{CC} + 0.3	V		
Analog power supply current	AI_{OPE}	AV_{CC}	—	—	1.5	mA	AV_{CC} = 5 V	
	AI_{STOP1}	AV_{CC}	—	600	—	μA		*2 Reference value
	AI_{STOP2}	AV_{CC}	—	—	5	μA		*3
Analog input capacitance	C_{AIN}	AN_0 to AN_3	—	—	15.0	pF		
Allowable signal source impedance	R_{AIN}		—	—	10.0	kΩ		
Resolution (data length)			—	—	10	bit		
Nonlinearity error			—	—	±2.5	LSB	AV_{CC} = 2.7 V to 5.5 V V_{CC} = 2.7 V to 5.5 V	
			—	—	±5.5		AV_{CC} = 2.0 V to 5.5 V V_{CC} = 2.0 V to 5.5 V	
			—	—	±7.5		Except the above	*4
Quantization error			—	—	±0.5	LSB		

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Absolute accuracy			—	—	±3.0	LSB	AV_{CC} = 2.7 V to 5.5 V V_{CC} = 2.7 V to 5.5 V	
			—	—	±6.0		AV_{CC} = 2.0 V to 5.5 V V_{CC} = 2.0 V to 5.5 V	
			—	—	±8.0		Except the above	*4
Conversion time			12.4	—	124	µs	AV_{CC} = 2.7 V to 5.5 V V_{CC} = 2.7 V to 5.5 V	
			62	—	124		Except the above	

Notes: 1. Set $AV_{CC} = V_{CC}$ when the A/D converter is not used.
2. AI_{STOP1} is the current in active and sleep modes while the A/D converter is idle.
3. AI_{STOP2} is the current at reset and in standby, watch, subactive, and subsleep modes while the A/D converter is idle.
4. Conversion time 62 µs

14.2.5 LCD Characteristics

Table 14.6 shows the LCD characteristics.

Table 14.6 LCD Characteristics

V_{CC} = 1.8 V to 5.5 V, AV_{CC} = 1.8 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C (including subactive mode) unless otherwise specified.

Item	Symbol	Applicable Pins	Test Conditions	Values Min	Typ	Max	Unit	Reference Figure
Segment driver drop voltage	V_{DS}	SEG_1 to SEG_{25}	I_D = 2 µA V_1 = 2.7 V to 5.5 V	—	—	0.6	V	*1
Common driver drop voltage	V_{DC}	COM_1 to COM_4	I_D = 2 µA V_1 = 2.7 V to 5.5 V	—	—	0.3	V	*1
LCD power supply split-resistance	R_{LCD}		Between V_1 and V_{SS}	0.5	3.0	9.0	MΩ	
Liquid crystal display voltage	V_{LCD}	V_1		2.2	—	5.5	V	*2

Notes: 1. The voltage drop from power supply pins V_1, V_2, V_3, and VSS to each segment pin or common pin.

2. When the liquid crystal display voltage is supplied from an external power source, ensure that the following relationship is maintained: $V_{CC} \geq V_1 \geq V_2 \geq V_3 \geq V_{SS}$.

14.3 Operation Timing

Figures 14.1 to 14.5 show timing diagrams.

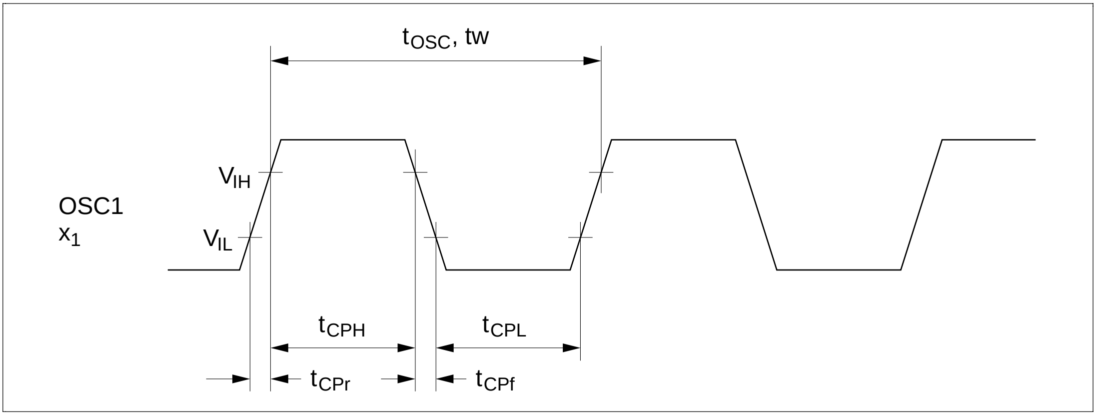


Figure 14.1 Clock Input Timing

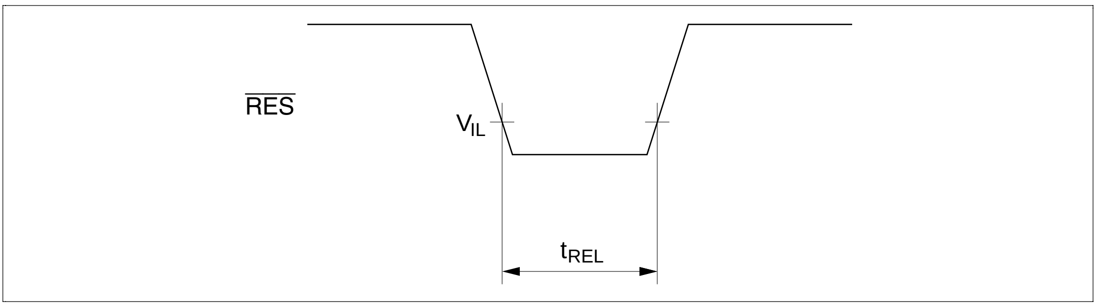


Figure 14.2 $\overline{RES}$ Low Width

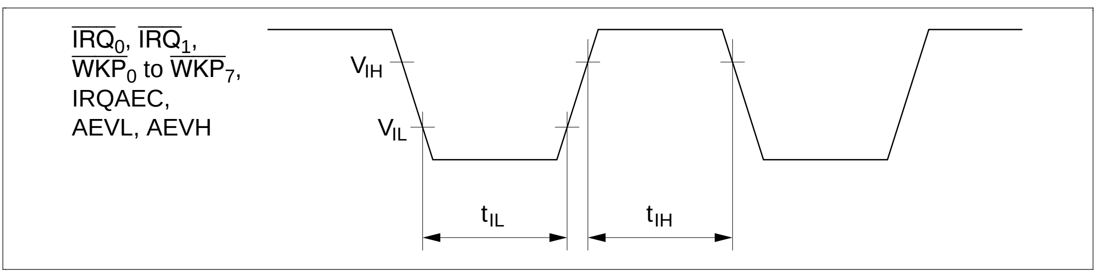


Figure 14.3 Input Timing

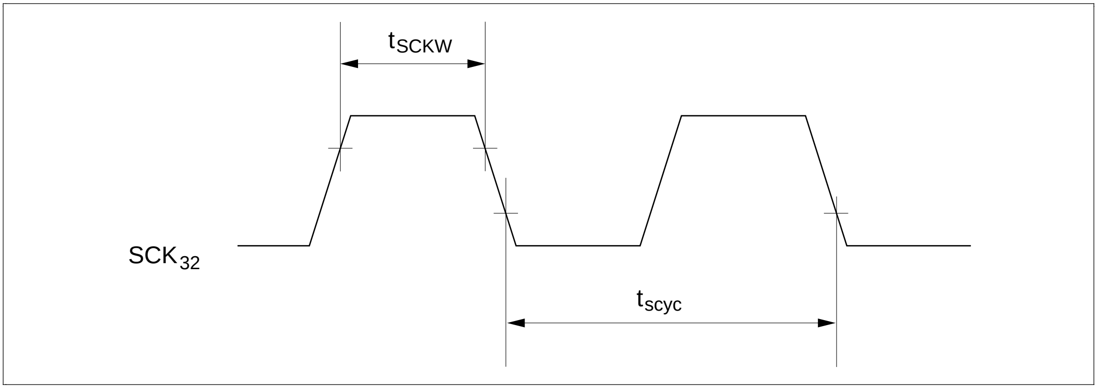


Figure 14.4 SCK3 Input Clock Timing

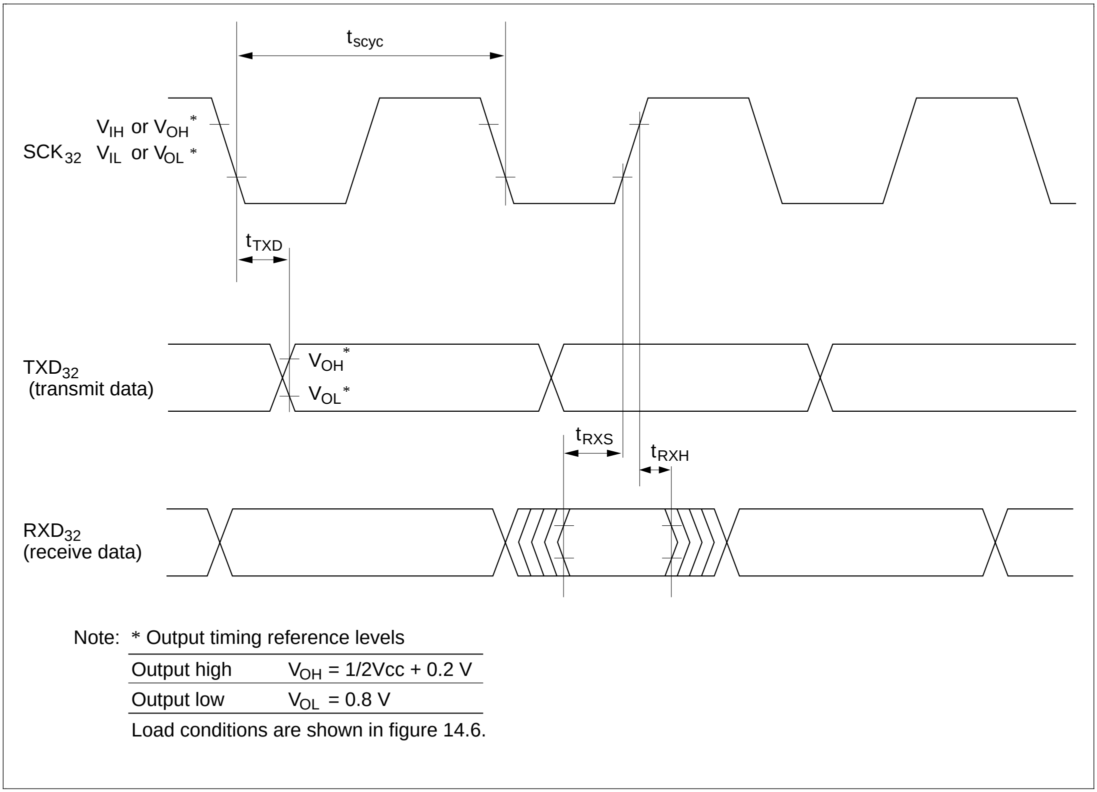


Figure 14.5 SCI3 Synchronous Mode Input/Output Timing

14.4 Output Load Circuit

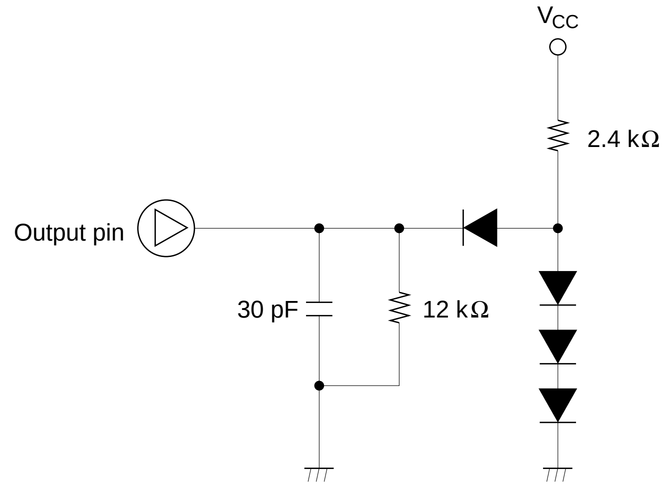
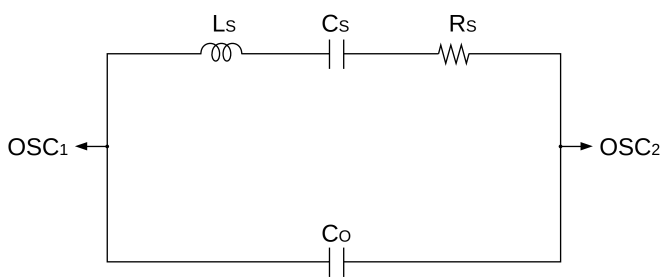


Figure 14.6 Output Load Condition

14.5 Resonator Equivalent Circuit



Crystal Resonator Parameter

Frequency (MHz)	4.193
R_S (max)	100 Ω
C_O (max)	16 pF

Ceramic Resonator Parameters

Frequency (MHz)	4
R_S (max)	8.8 Ω
C_O (max)	36 pF

Figure 14.7 Resonator Equivalent Circuit

14.6 Usage Note

The ZTAT and mask ROM versions both satisfy the electrical characteristics shown in this manual, but actual electrical characteristic values, operating margins, noise margins, and other properties may vary due to differences in manufacturing process, on-chip ROM, layout patterns, and so on.

When system evaluation testing is carried out using the ZTAT version, the same evaluation testing should also be conducted for the mask ROM version when changing over to that version.

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

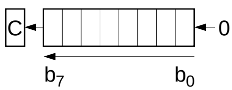
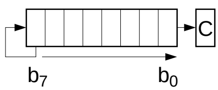
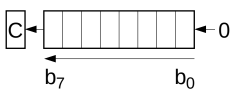
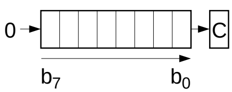
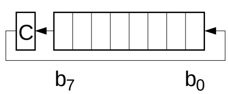
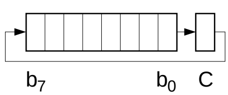
Symbol	
↕	Modified according to the instruction result
*	Not fixed (value not guaranteed)
0	Always cleared to 0
—	Not affected by the instruction execution result

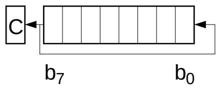
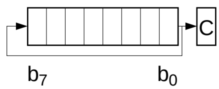
Table A.1 lists the H8/300L CPU instruction set.

Table A.1 Instruction Set

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (bytes): #xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	Condition Code: I	H	N	Z	V	C	No. of States
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
PUSH Rs	W	SP–2 → SP Rs16 → @SP					2					—	—	↕	↕	0	—	6

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	
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

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	
MULXU.B Rs, Rd	B	Rd8 × Rs8 → Rd16		2								—	—	—	—	—	—	14
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

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (bytes) #xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	Condition Code I	H	N	Z	V	C	No. of States
ROTL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTR.B Rd	B			2								—	—	↕	↕	0	↕	2
BSET #xx:3, Rd	B	(#xx:3 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BSET Rn, Rd	B	(Rn8 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET Rn, @Rd	B	(Rn8 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET Rn, @aa:8	B	(Rn8 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BCLR #xx:3, Rd	B	(#xx:3 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BCLR Rn, Rd	B	(Rn8 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR Rn, @Rd	B	(Rn8 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR Rn, @aa:8	B	(Rn8 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BNOT #xx:3, Rd	B	(#xx:3 of Rd8) ← $\overline{\text{(\#xx:3 of Rd8)}}$		2								—	—	—	—	—	—	2
BNOT #xx:3, @Rd	B	(#xx:3 of @Rd16) ← $\overline{\text{(\#xx:3 of @Rd16)}}$			4							—	—	—	—	—	—	8
BNOT #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← $\overline{\text{(\#xx:3 of @aa:8)}}$						4				—	—	—	—	—	—	8
BNOT Rn, Rd	B	(Rn8 of Rd8) ← $\overline{\text{(Rn8 of Rd8)}}$		2								—	—	—	—	—	—	2
BNOT Rn, @Rd	B	(Rn8 of @Rd16) ← $\overline{\text{(Rn8 of @Rd16)}}$			4							—	—	—	—	—	—	8
BNOT Rn, @aa:8	B	(Rn8 of @aa:8) ← $\overline{\text{(Rn8 of @aa:8)}}$						4				—	—	—	—	—	—	8

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	
BTST #xx:3, Rd	B	($\overline{\text{\#xx:3 of Rd8}}$) → Z		2								—	—	—	↕	—	—	2
BTST #xx:3, @Rd	B	($\overline{\text{\#xx:3 of @Rd16}}$) → Z			4							—	—	—	↕	—	—	6
BTST #xx:3, @aa:8	B	($\overline{\text{\#xx:3 of @aa:8}}$) → Z						4				—	—	—	↕	—	—	6
BTST Rn, Rd	B	($\overline{\text{Rn8 of Rd8}}$) → Z		2								—	—	—	↕	—	—	2
BTST Rn, @Rd	B	($\overline{\text{Rn8 of @Rd16}}$) → Z			4							—	—	—	↕	—	—	6
BTST Rn, @aa:8	B	($\overline{\text{Rn8 of @aa:8}}$) → Z						4				—	—	—	↕	—	—	6
BLD #xx:3, Rd	B	(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BLD #xx:3, @Rd	B	(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BLD #xx:3, @aa:8	B	(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BILD #xx:3, Rd	B	($\overline{\text{\#xx:3 of Rd8}}$) → C		2								—	—	—	—	—	↕	2
BILD #xx:3, @Rd	B	($\overline{\text{\#xx:3 of @Rd16}}$) → C			4							—	—	—	—	—	↕	6
BILD #xx:3, @aa:8	B	($\overline{\text{\#xx:3 of @aa:8}}$) → C						4				—	—	—	—	—	↕	6
BST #xx:3, Rd	B	C → (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BST #xx:3, @Rd	B	C → (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BST #xx:3, @aa:8	B	C → (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BIST #xx:3, Rd	B	$\overline{\text{C}}$ → (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BIST #xx:3, @Rd	B	$\overline{\text{C}}$ → (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BIST #xx:3, @aa:8	B	$\overline{\text{C}}$ → (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BAND #xx:3, Rd	B	C∧(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BAND #xx:3, @Rd	B	C∧(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BAND #xx:3, @aa:8	B	C∧(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIAND #xx:3, Rd	B	C∧($\overline{\text{\#xx:3 of Rd8}}$) → C		2								—	—	—	—	—	↕	2
BIAND #xx:3, @Rd	B	C∧($\overline{\text{\#xx:3 of @Rd16}}$) → C			4							—	—	—	—	—	↕	6
BIAND #xx:3, @aa:8	B	C∧($\overline{\text{\#xx:3 of @aa:8}}$) → C						4				—	—	—	—	—	↕	6
BOR #xx:3, Rd	B	C∨(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BOR #xx:3, @Rd	B	C∨(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIOR #xx:3, Rd	B	C∨($\overline{\text{\#xx:3 of Rd8}}$) → C		2								—	—	—	—	—	↕	2
BIOR #xx:3, @Rd	B	C∨($\overline{\text{\#xx:3 of @Rd16}}$) → C			4							—	—	—	—	—	↕	6

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	
BIOR #xx:3, @aa:8	B	$C \vee (\overline{\#xx:3 \text{ of } @aa:8}) \rightarrow C$							4				—	—	—	—	—	↕	6
BXOR #xx:3, Rd	B	$C \oplus (\#xx:3 \text{ of } Rd8) \rightarrow C$			2								—	—	—	—	—	↕	2
BXOR #xx:3, @Rd	B	$C \oplus (\#xx:3 \text{ of } @Rd16) \rightarrow C$				4							—	—	—	—	—	↕	6
BXOR #xx:3, @aa:8	B	$C \oplus (\#xx:3 \text{ of } @aa:8) \rightarrow C$							4				—	—	—	—	—	↕	6
BIXOR #xx:3, Rd	B	$C \oplus (\overline{\#xx:3 \text{ of } Rd8}) \rightarrow C$			2								—	—	—	—	—	↕	2
BIXOR #xx:3, @Rd	B	$C \oplus (\overline{\#xx:3 \text{ of } @Rd16}) \rightarrow C$				4							—	—	—	—	—	↕	6
BIXOR #xx:3, @aa:8	B	$C \oplus (\overline{\#xx:3 \text{ of } @aa:8}) \rightarrow C$							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

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 @Rn	—	SP–2 → SP PC → @SP PC ← Rn16			2							—	—	—	—	—	—	6
JSR @aa:16	—	SP–2 → SP PC → @SP PC ← aa:16						4				—	—	—	—	—	—	8
JSR @@aa:8	—	SP–2 → SP PC → @SP PC ← @aa:8								2		—	—	—	—	—	—	8
RTS	—	PC ← @SP SP+2 → SP									2	—	—	—	—	—	—	8
RTE	—	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
SLEEP	—	Transit to sleep mode.									2	—	—	—	—	—	—	2
LDC #xx:8, CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
LDC Rs, CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
STC CCR, Rd	B	CCR → Rd8		2								—	—	—	—	—	—	2
ANDC #xx:8, CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
ORC #xx:8, CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
XORC #xx:8, CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
NOP	—	PC ← PC+2									2	—	—	—	—	—	—	2
EEPMOV	—	if R4L≠0 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.

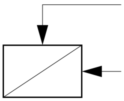
(4) The number of states required for execution is 4n + 9 (n = value of R4L).

(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).



Instruction when first bit of byte 2 (bit 7 of first instruction word) is 0.

Instruction when first bit of byte 2 (bit 7 of first instruction word) is 1.

Table A.2 Operation Code Map

<table>
<tr><th>Low
High</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th></tr>
<tr><td>0</td><td>NOP</td><td>SLEEP</td><td>STC</td><td>LDC</td><td>ORC</td><td>XORC</td><td>ANDC</td><td>LDC</td><td colspan="2">ADD</td><td>INC</td><td>ADDS</td><td colspan="2">MOV</td><td>ADDX</td><td>DAA</td></tr>
<tr><td>1</td><td>SHLL
SHAL</td><td>SHLR
SHAR</td><td>ROTXL
ROTL</td><td>ROTXR
ROTR</td><td>OR</td><td>XOR</td><td>AND</td><td>NOT
NEG</td><td colspan="2">SUB</td><td>DEC</td><td>SUBS</td><td colspan="2">CMP</td><td>SUBX</td><td>DAS</td></tr>
<tr><td>2</td><td colspan="16" rowspan="2">MOV</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td><td>BRA</td><td>BRN</td><td>BHI</td><td>BLS</td><td>BCC</td><td>BCS</td><td>BNE</td><td>BEQ</td><td>BVC</td><td>BVS</td><td>BPL</td><td>BMI</td><td>BGE</td><td>BLT</td><td>BGT</td><td>BLE</td></tr>
<tr><td>5</td><td>MULXU</td><td>DIVXU</td><td colspan="2"></td><td>RTS</td><td>BSR</td><td>RTE</td><td colspan="2"></td><td colspan="3">JMP</td><td></td><td colspan="3">JSR</td></tr>
<tr><td>6</td><td rowspan="2">BSET</td><td rowspan="2">BNOT</td><td rowspan="2">BCLR</td><td rowspan="2">BTST</td><td colspan="3"></td><td>BST
BIST</td><td colspan="8">MOV*</td></tr>
<tr><td>7</td><td>BOR
BIOR</td><td>BXOR
BIXOR</td><td>BAND
BIAND</td><td>BLD
BILD</td><td></td><td>MOV</td><td></td><td>EEPMOV</td><td colspan="4">Bit-manipulation instructions</td></tr>
<tr><td>8</td><td colspan="16">ADD</td></tr>
<tr><td>9</td><td colspan="16">ADDX</td></tr>
<tr><td>A</td><td colspan="16">CMP</td></tr>
<tr><td>B</td><td colspan="16">SUBX</td></tr>
<tr><td>C</td><td colspan="16">OR</td></tr>
<tr><td>D</td><td colspan="16">XOR</td></tr>
<tr><td>E</td><td colspan="16">AND</td></tr>
<tr><td>F</td><td colspan="16">MOV</td></tr>
</table>

Note: * The PUSH and POP instructions are identical in machine language to MOV instructions.

A.3 Number of Execution States

The tables here can be used to calculate the number of states required for instruction execution. Table A.4 indicates the number of states required for each cycle (instruction fetch, read/write, etc.), and table A.3 indicates the number of cycles of each type occurring in each instruction. The total number of states required for execution of an instruction can be calculated from these two tables as follows:

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.

BSET #0, @FF00
From table A.4:
$I = L = 2, \quad J = K = M = N= 0$
From table A.3:
$S_I = 2, \quad S_L = 2$
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.

JSR @@ 30
From table A.4:
$I = 2, \quad J = K = 1, \quad L = M = N = 0$
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	Access Location: 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	

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					
	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		

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BILD	BILD #xx:3, Rd	1					
	BILD #xx:3, @Rd	2			1		
	BILD #xx:3, @aa:8	2			1		
BIOR	BIOR #xx:3, Rd	1					
	BIOR #xx:3, @Rd	2			1		
	BIOR #xx:3, @aa:8	2			1		
BIST	BIST #xx:3, Rd	1					
	BIST #xx:3, @Rd	2			2		
	BIST #xx:3, @aa:8	2			2		
BIXOR	BIXOR #xx:3, Rd	1					
	BIXOR #xx:3, @Rd	2			1		
	BIXOR #xx:3, @aa:8	2			1		
BLD	BLD #xx:3, Rd	1					
	BLD #xx:3, @Rd	2			1		
	BLD #xx:3, @aa:8	2			1		
BNOT	BNOT #xx:3, Rd	1					
	BNOT #xx:3, @Rd	2			2		
	BNOT #xx:3, @aa:8	2			2		
	BNOT Rn, Rd	1					
	BNOT Rn, @Rd	2			2		
	BNOT Rn, @aa:8	2			2		
BOR	BOR #xx:3, Rd	1					
	BOR #xx:3, @Rd	2			1		
	BOR #xx:3, @aa:8	2			1		
BSET	BSET #xx:3, Rd	1					
	BSET #xx:3, @Rd	2			2		
	BSET #xx:3, @aa:8	2			2		
	BSET Rn, Rd	1					
	BSET Rn, @Rd	2			2		
	BSET Rn, @aa:8	2			2		
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		

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BTST	BTST Rn, @aa:8	2			1		
BXOR	BXOR #xx:3, Rd	1					
	BXOR #xx:3, @Rd	2			1		
	BXOR #xx:3, @aa:8	2			1		
CMP	CMP. B #xx:8, Rd	1					
	CMP. B Rs, Rd	1					
	CMP.W Rs, Rd	1					
DAA	DAA.B Rd	1					
DAS	DAS.B Rd	1					
DEC	DEC.B Rd	1					
DIVXU	DIVXU.B Rs, Rd	1					12
EEPMOV	EEPMOV	2			2n+2*		1
INC	INC.B Rd	1					
JMP	JMP @Rn	2					
	JMP @aa:16	2					2
	JMP @@aa:8	2	1				2
JSR	JSR @Rn	2		1			
	JSR @aa:16	2		1			2
	JSR @@aa:8	2	1	1			
LDC	LDC #xx:8, CCR	1					
	LDC Rs, CCR	1					
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					
	MOV.B @Rs, Rd	1			1		
	MOV.B @(d:16, Rs), Rd	2			1		
	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	

Note: * n: Initial value in R4L. The source and destination operands are accessed n + 1 times each.

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.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					
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 Addresses

Lower Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module Name
H'80										
H'81										
H'82										
H'83										
H'84										
H'85										
H'86										
H'87										
H'88										
H'89										
H'8A										
H'8B										
H'8C	ECPWCRH	ECPWCRH7	ECPWCRH6	ECPWCRH5	ECPWCRH4	ECPWCRH3	ECPWCRH2	ECPWCRH1	ECPWCRH0	Asynchronous event counter
H'8D	ECPWCRL	ECPWCRL7	ECPWCRL6	ECPWCRL5	ECPWCRL4	ECPWCRL3	ECPWCRL2	ECPWCRL1	ECPWCRL0	
H'8E	ECPWDRH	ECPWDRH7	ECPWDRH6	ECPWDRH5	ECPWDRH4	ECPWDRH3	ECPWDRH2	ECPWDRH1	ECPWDRH0	
H'8F	ECPWDRL	ECPWDRL7	ECPWDRL6	ECPWDRL5	ECPWDRL4	ECPWDRL3	ECPWDRL2	ECPWDRL1	ECPWDRL0	
H'90	WEGR	WKEGS7	WKEGS6	WKEGS5	WKEGS4	WKEGS3	WKEGS2	WKEGS1	WKEGS0	System control
H'91	SPCR	—	—	SPC32	—	SCINV3	SCINV2	—	—	SCI
H'92	AEGSR	AHEGS1	AHEGS0	ALEGS1	ALEGS0	AIEGS1	AIEGS0	ECPWME	—	Asynchronous event counter
H'93										
H'94	ECCR	ACKH1	ACKH0	ACKL1	ACKL0	PWCK2	PWCK1	PWCK0	—	
H'95	ECCSR	OVH	OVL	—	CH2	CUEH	CUEL	CRCH	CRCL	
H'96	ECH	ECH7	ECH6	ECH5	ECH4	ECH3	ECH2	ECH1	ECH0	
H'97	ECL	ECL7	ECL6	ECL5	ECL4	ECL3	ECL2	ECL1	ECL0	
H'98										
H'99										
H'9A										
H'9B										
H'9C										
H'9D										
H'9E										
H'9F										
H'A0										
H'A1										

Lower Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module Name
H'A2										
H'A3										
H'A4										
H'A5										
H'A6										
H'A7										
H'A8	SMR	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0	SCI
H'A9	BRR	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0	
H'AA	SCR3	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'AB	TDR	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0	
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	—	—	—	—	TMA3	TMA2	TMA1	TMA0	Timer A
H'B1	TCA	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0	
H'B2										
H'B3										
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										
H'BD										
H'BE										
H'BF										
H'C0	LPCR	DTS1	DTS0	CMX	—	SGS3	SGS2	SGS1	SGS0	LCD controller/ driver
H'C1	LCR	—	PSW	ACT	DISP	CKS3	CKS2	CKS1	CKS0	
H'C2	LCR2	LCDAB	—	—	—	—	—	—	—	
H'C3										
H'C4	ADRRH	ADR9	ADR8	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	A/D converter
H'C5	ADRRL	ADR1	ADR0	—	—	—	—	—	—	
H'C6	AMR	CKS	—	—	—	CH3	CH2	CH1	CH0	
H'C7	ADSR	ADSF	—	—	—	—	—	—	—	
H'C8										I/O port
H'C9	PMR2	—	—	POF1	—	—	—	—	IRQ0	

Lower Address	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'CA	PMR3	AEVL	AEVH	—	—	—	TMOFH	TMOFL	—	I/O port
H'CB										
H'CC	PMR5	WKP7	WKP6	WKP5	WKP4	WKP3	WKP2	WKP1	WKP0	
H'CD	PWCR2	—	—	—	—	—	—	PWCR21	PWCR20	10 bit PWM2
H'CE	PWDRU2	—	—	—	—	—	—	PWDRU21	PWDRU20	
H'CF	PWDRL2	PWDRL27	PWDRL26	PWDRL25	PWDRL24	PWDRL23	PWDRL22	PWDRL21	PWDRL20	
H'D0	PWCR1	—	—	—	—	—	—	PWCR11	PWCR10	10 bit PWM1
H'D1	PWDRU1	—	—	—	—	—	—	PWDRU11	PWDRU10	
H'D2	PWDRL1	PWDRL17	PWDRL16	PWDRL15	PWDRL14	PWDRL13	PWDRL12	PWDRL11	PWDRL10	
H'D3										I/O port
H'D4										
H'D5										
H'D6	PDR3	P37	P36	P35	P34	P33	P32	P31	—	
H'D7	PDR4	—	—	—	—	P43	P42	P41	P40	
H'D8	PDR5	P57	P56	P55	P54	P53	P52	P51	P50	
H'D9	PDR6	P67	P66	P65	P64	P63	P62	P61	P60	
H'DA	PDR7	P77	P76	P75	P74	P73	P72	P71	P70	
H'DB	PDR8	—	—	—	—	—	—	—	P80	
H'DC	PDR9	—	—	P95	P94	P93	P92	P91	P90	
H'DD	PDRA	—	—	—	—	PA3	PA2	PA1	PA0	
H'DE	PDRB	—	—	—	—	PB3	PB2	PB1	PB0	
H'DF										
H'E0										
H'E1	PUCR3	PUCR37	PUCR36	PUCR35	PUCR34	PUCR33	PUCR32	PUCR31	—	
H'E2	PUCR5	PUCR57	PUCR56	PUCR55	PUCR54	PUCR53	PUCR52	PUCR51	PUCR50	
H'E3	PUCR6	PUCR67	PUCR66	PUCR65	PUCR64	PUCR63	PUCR62	PUCR61	PUCR60	
H'E4										
H'E5										
H'E6	PCR3	PCR37	PCR36	PCR35	PCR34	PCR33	PCR32	PCR31	—	
H'E7	PCR4	—	—	—	—	—	PCR42	PCR41	PCR40	
H'E8	PCR5	PCR57	PCR56	PCR55	PCR54	PCR53	PCR52	PCR51	PCR50	
H'E9	PCR6	PCR67	PCR66	PCR65	PCR64	PCR63	PCR62	PCR61	PCR60	
H'EA	PCR7	PCR77	PCR76	PCR75	PCR74	PCR73	PCR72	PCR71	PCR70	
H'EB	PCR8	—	—	—	—	—	—	—	PCR80	
H'EC	PMR9	—	—	—	—	PIOFF	—	PWM2	PWM1	
H'ED	PCRA	—	—	—	—	PCRA3	PCRA2	PCRA1	PCRA0	
H'EE	PMRB	—	—	—	—	IRQ1	—	—	—	
H'EF										
H'F0	SYSCR1	SSBY	STS2	STS1	STS0	LSON	—	MA1	MA0	System control
H'F1	SYSCR2	—	—	—	NESEL	DTON	MSON	SA1	SA0	

Lower Address	Register Name	Bit Names Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module Name
H'F2	IEGR	—	—	—	—	—	—	IEG1	IEG0	System control
H'F3	IENR1	IENTA	—	IENWP	—	—	IENEC2	IEN1	IEN0	
H'F4	IENR2	IENDT	IENAD	—	—	IENTFH	IENTFL	—	IENEC	
H'F5										
H'F6	IRR1	IRRTA	—	—	—	—	IRREC2	IRRI1	IRRI0	
H'F7	IRR2	IRRDT	IRRAD	—	—	IRRTFH	IRRTFL	—	IRREC	
H'F8										
H'F9	IWPR	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0	System control
H'FA	CKSTPR1	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP	
H'FB	CKSTPR2	—	—	—	PW2CKSTP	AECKSTP	—	PW1CKSTP	LDCKSTP	
H'FC										
H'FD										
H'FE										
H'FF										

Legend

SCI: Serial Communication Interface

B.2 Functions

Register name

Register acronym

Address to which the register is mapped

Name of on-chip supporting module

Bit numbers

Initial bit values
Dashes (—) indicate undefined bits.

Possible types of access

R	Read only
W	Write only
R/W	Read and write
—	See relevant register description

Names of the bits. Dashes (—) indicate reserved bits.

Full name of bit

Descriptions of bit settings

TCRF—Timer Control Register F H'B6 **Timer F**

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
R/W	W	W	W	W	W	W	W	W

Clock select L

0	*	*	Counts on external event (TMIF) rising/ falling edge
1	0	0	Internal clock: ø/32
1	0	1	Internal clock: ø/16
1	1	0	Internal clock: ø/4
1	1	1	Internal clock: øw/4

Toggle output level L

0	Set to low level
1	Set to high level

Clock select H

0	*	*	16-bit mode, counts on TCFL overflow signal
1	0	0	Internal clock: ø/32
1	0	1	Internal clock: ø/16
1	1	0	Internal clock: ø/4
1	1	1	Internal clock: øw/4

* Don't care

Toggle output level H

0	Set to low level
1	Set to high level

ECPWCRH—Event Counter PWM Compare Register H H'8C AEC

Bit	7	6	5	4	3	2	1	0
	ECPWCRH7	ECPWCRH6	ECPWCRH5	ECPWCRH4	ECPWCRH3	ECPWCRH2	ECPWCRH1	ECPWCRH0
Initial value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets event counter PWM waveform conversion period

ECPWCRL—Event Counter PWM Compare Register L H'8D AEC

Bit	7	6	5	4	3	2	1	0
	ECPWCRL7	ECPWCRL6	ECPWCRL5	ECPWCRL4	ECPWCRL3	ECPWCRL2	ECPWCRL1	ECPWCRL0
Initial value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Sets event counter PWM waveform conversion period

ECPWDRH—Event Counter PWM Data Register H H'8E AEC

Bit	7	6	5	4	3	2	1	0
	ECPWDRH7	ECPWDRH6	ECPWDRH5	ECPWDRH4	ECPWDRH3	ECPWDRH2	ECPWDRH1	ECPWDRH0
Initial value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Controls event counter PWM waveform generator data

ECPWDRL—Event Counter PWM Data Register L H'8F AEC

Bit	7	6	5	4	3	2	1	0
	ECPWDRL7	ECPWDRL6	ECPWDRL5	ECPWDRL4	ECPWDRL3	ECPWDRL2	ECPWDRL1	ECPWDRL0
Initial value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Controls event counter PWM waveform generator data

Bit	7	6	5	4	3	2	1	0
	WKEGS7	WKEGS6	WKEGS5	WKEGS4	WKEGS3	WKEGS2	WKEGS1	WKEGS0
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

$\overline{\text{WKPn}}$ edge selected

0	$\overline{\text{WKPn}}$ pin falling edge detected
1	$\overline{\text{WKPn}}$ pin rising edge detected

(n = 7 to 0)

Bit	7	6	5	4	3	2	1	0
	—	—	SPC32	—	SCINV3	SCINV2	—	—
Initial value	1	1	0	—	0	0	—	—
Read/Write	—	—	R/W	W	R/W	R/W	W	W

RXD_{32} pin input data inversion switch

0	RXD_{32} input data is not inverted
1	RXD_{32} input data is inverted

TXD_{32} pin output data inversion switch

0	TXD_{32} output data is not inverted
1	TXD_{32} output data is inverted

$P4_2$/TXD_{32}pin function switch

0	Function as $P4_2$ I/O pin
1	Function as TXD_{32} output pin

Bit	7	6	5	4	3	2	1	0
	AHEGS1	AHEGS0	ALEGS1	ALEGS0	AIEGS1	AIEGS0	ECPWME	—
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

Reserved bit

Event counter PWM enable/disable, IRQAEC select/deselect

0	AEC PWM halted, IRQAEC selected
1	AEC PWM operation enabled, IRQAEC deselected

IRQAEC edge select

Bit 3 AIEGS1	Bit 2 AIEGS0	Description
0	0	Falling edge on IRQAEC pin is sensed
0	1	Rising edge on IRQAEC pin is sensed
1	0	Both edges on IRQAEC pin are sensed
1	1	Use prohibited

AEC edge select L

Bit 5 ALEGS1	Bit 4 ALEGS0	Description
0	0	Falling edge on AEVL pin is sensed
0	1	Rising edge on AEVL pin is sensed
1	0	Both edges on AEVL pin are sensed
1	1	Use prohibited

AEC edge select H

Bit 7 AHEGS1	Bit 6 AHEGS0	Description
0	0	Falling edge on AEVH pin is sensed
0	1	Rising edge on AEVH pin is sensed
1	0	Both edges on AEVH pin are sensed
1	1	Use prohibited

Bit	7	6	5	4	3	2	1	0
	ACKH1	ACKH0	ACKL1	ACKL0	PWCK2	PWCK1	PWCK0	—
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

Reserved bit

Event counter PWM clock select

Bit 3 PWCK2	Bit 2 PWCK1	Bit 1 PWCK0	Description
0	0	0	ø/2
0	0	1	ø/4
0	1	0	ø/8
0	1	1	ø/16
0	*	0	ø/32
0	*	1	ø/64

AEC clock select L

Bit 5 ACKL1	Bit 4 ACKL0	Description
0	0	AEVL pin input
0	1	ø/2
1	0	ø/4
1	1	ø/8

AEC clock select H

Bit 7 ACKH1	Bit 6 ACKH0	Description
0	0	AEVH pin input
0	1	ø/2
1	0	ø/4
1	1	ø/8

Bit	7	6	5	4	3	2	1	0
	OVH	OVL	—	CH2	CUEH	CUEL	CRCH	CRCL
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Counter reset control L (bit 0, CRCL)

0	ECL is reset
1	ECL reset is cleared and count-up function is enabled

Counter reset control H (bit 1, CRCH)

0	ECH is reset
1	ECH reset is cleared and count-up function is enabled

Count-up enable L (bit 2, CUEL)

0	ECL event clock input is disabled. ECL value is held
1	ECL event clock input is enabled

Count-up enable H (bit 3, CUEH)

0	ECH event clock input is disabled. ECH value is held
1	ECH event clock input is enabled

Channel select (bit 4, CH2)

0	ECH and ECL are used together as a single-channel 16-bit event counter
1	ECH and ECL are used as two independent 8-bit event counter channels

Counter overflow L (bit 6, OVL)

0	ECL has not overflowed
1	ECL has overflowed

Counter overflow H (bit 7, OVH)

0	ECH has not overflowed
1	ECH has overflowed

ECH—Event counter H H'96 AEC

Bit	7	6	5	4	3	2	1	0
	ECH7	ECH6	ECH5	ECH4	ECH3	ECH2	ECH1	ECH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ECL—Event counter L H'97 AEC

Bit	7	6	5	4	3	2	1	0
	ECL7	ECL6	ECL5	ECL4	ECL3	ECL2	ECL1	ECL0
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
	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	0	ø clock
0	1	øw/2 clock
1	0	ø/16 clock
1	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 addition and checking disabled
1	Parity bit addition and checking enabled

Character length

0	8-bit data/5-bit data
1	7-bit data/5-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 Pin Function
0	0	Asynchronous	Internal clock	I/O port
		Synchronous	Internal clock	Serial clock output
0	1	Asynchronous	Internal clock	Clock output
		Synchronous	Reserved (Do not specify this combination)	
1	0	Asynchronous	External clock	Clock input
		Synchronous	External clock	Serial clock input
1	1	Asynchronous	Reserved (Do not specify this combination)	
		Synchronous	Reserved (Do not specify this combination)	

Transmit end interrupt enable

0	Transmit end interrupt request (TEI) disabled
1	Transmit end interrupt request (TEI) enabled

Multiprocessor interrupt enable

0	Multiprocessor interrupt request disabled (normal receive operation) [Clearing conditions] When data is received in which the multiprocessor bit is set to 1
1	Multiprocessor interrupt request enabled The receive interrupt request (RXI), receive error interrupt request (ERI), and setting of the RDRF, FER, and OER flags in the serial status register (SSR), are disabled until data with the multiprocessor bit set to 1 is received.

Receive enable

0	Receive operation disabled (RXD pin is I/O port)
1	Receive operation enabled (RXD pin is receive data pin)

Transmit enable

0	Transmit operation disabled (TXD pin is transmit data pin)
1	Transmit operation enabled (TXD pin is 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 for transfer 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 transfer (bit 0, MPBT)

0	A 0 multiprocessor bit is transmitted
1	A 1 multiprocessor bit is transmitted

Multiprocessor bit receive (bit 1, MPBR)

0	Data in which the multiprocessor bit is 0 has been received
1	Data in which the multiprocessor bit is 1 has been received

Transmit end (bit 2, TEND)

0	Transmission in progress [Clearing conditions] • After reading TDRE = 1, cleared by writing 0 to TDRE • When data is written to TDR by an instruction
1	Transmission ended [Setting conditions] • When bit TE in serial control register3 (SCR3) is cleared to 0 • When bit TDRE is set to 1 when the last bit of a transmit character is sent

Parity error (bit 3, PER)

0	Reception in progress or completed normally [Clearing conditions] After reading PER = 1, cleared by writing 0 to PER
1	A parity error has occurred during reception [Setting conditions] When the number of 1 bits in the receive data plus parity bit does not match the parity designated by the parity mode bit (PM) in the serial mode register (SMR)

Framing error (bit 4, FER)

0	Reception in progress or completed normally [Clearing conditions] After reading FER = 1, cleared by writing 0 to FER
1	A framing error has occurred during reception [Setting conditions] When the stop bit at the end of the receive data is checked for a value of 1 at completion of reception, and the stop bit is 0

Overrun error (bit 5, OER)

0	Reception in progress or completed [Clearing conditions] After reading OER = 1, cleared by writing 0 to OER
1	An overrun error has occurred during reception [Setting conditions] When the next serial reception is completed with RDRF set to 1

Receive data register full (bit 6, RDRF)

0	There is no receive data in RDR [Clearing conditions] • After reading RDRF = 1, cleared by writing 0 to RDRF • When RDR data is read by an instruction
1	There is receive data in RDR [Setting conditions] When reception ends normally and receive data is transferred from RSR to RDR

Transmit data register empty (bit 7, TDRE)

0	Transmit data written in 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	Transmit data has not been written to TDR, or transmit data written in TDR has been transferred to TSR [Setting conditions] • When bit TE in serial control register3 (SCR3) is cleared to 0 • When data is transferred from TDR to TSR

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

RDR—Receive data register H'AD SCI3

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
	—	—	—	—	TMA3	TMA2	TMA1	TMA0
Initial value	—	—	—	1	0	0	0	0
Read/Write	W	W	W	—	R/W	R/W	R/W	R/W

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 (when using 32.768 kHz)
			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
	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

Clock select L

0	0	0	Non-operationl
0	0	1	Use prohibited
0	1	0	Use prohibited
0	1	1	Use prohibited
1	0	0	Internal clock ø/32
1	0	1	Internal clock ø/16
1	1	0	Internal clock ø/4
1	1	1	Internal clock øw/4

Toggle output level L

0	Low level
1	High level

Clock select H

0	0	0	16-bit mode, counting on TCFL overflow signal
0	0	1	
0	1	0	
0	1	1	Use prohibited
1	0	0	Internal clock ø/32
1	0	1	Internal clock ø/16
1	1	0	Internal clock ø/4
1	1	1	Internal clock øw/4

Toggle output level H

0	Low level
1	High level

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

Counter clear L

0	TCFL clearing by compare match is disabled
1	TCFL clearing by compare match is enabled

Timer overflow interrupt enable L

0	TCFL overflow interrupt request is disabled
1	TCFL overflow interrupt request is enabled

Compare match flag L

0	Clearing conditions: After reading CMFL = 1, cleared by writing 0 to CMFL
1	Setting conditions: Set when the TCFL value matches the OCRFL value

Timer overflow flag L

0	Clearing conditions: After reading OVFL = 1, cleared by writing 0 to OVFL
1	Setting conditions: Set when TCFL overflows from H'FF to H'00

Counter clear H

0	16-bit mode: TCF clearing by compare match is disabled 8-bit mode: TCFH clearing by compare match is disabled
1	16-bit mode: TCF clearing by compare match is enabled 8-bit mode: TCFH clearing by compare match is enabled

Timer overflow interrupt enable H

0	TCFH overflow interrupt request is disabled
1	TCFH overflow interrupt request is enabled

Compare match flag H

0	Clearing conditions: After reading CMFH = 1, cleared by writing 0 to CMFH
1	Setting conditions: Set when the TCFH value matches the OCRFH value

Timer overflow flag H

0	Clearing conditions: After reading OVFH = 1, cleared by writing 0 to OVFH
1	Setting conditions: Set when TCFH overflows from H'FF to H'00

Note: * Bits 7, 6, 3, and 2 can only be written with 0, for flag clearing.

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
	DTS1	DTS0	CMX	—	SGS3	SGS2	SGS1	SGS0
Initial value	0	0	0	—	0	0	0	0
Read/Write	R/W	R/W	R/W	W	R/W	R/W	R/W	R/W

Clock enable

Bit 3	Bit 2	Bit 1	Bit 0	Function of Pins SEG_{25} to SEG_1							Notes
SGS3	SGS2	SGS1	SGS0	SEG_{25}	SEG_{24} to SEG_{21}	SEG_{20} to SEG_{17}	SEG_{16} to SEG_{13}	SEG_{12} to SEG_9	SEG_8 to SEG_5	SEG_4 to SEG_1	
0	0	0	0	Port	Port	Port	Port	Port	Port	Port	(Initial value)
0	0	0	1	Port	Port	Port	Port	Port	Port	SEG	
0	0	1	0	Port	Port	Port	Port	Port	SEG	SEG	
0	0	1	1	Port	Port	Port	Port	SEG	SEG	SEG	
0	1	0	0	Port	Port	Port	SEG	SEG	SEG	SEG	
0	1	0	1	Port	Port	SEG	SEG	SEG	SEG	SEG	
0	1	1	0	Port	SEG	SEG	SEG	SEG	SEG	SEG	
0	1	1	1	SEG	SEG	SEG	SEG	SEG	SEG	SEG	
1	0	0	0	SEG	SEG	SEG	SEG	SEG	SEG	SEG	
1	0	0	1	SEG	SEG	SEG	SEG	SEG	SEG	Port	
1	0	1	0	SEG	SEG	SEG	SEG	SEG	Port	Port	
1	0	1	1	SEG	SEG	SEG	SEG	Port	Port	Port	
1	1	0	0	SEG	SEG	SEG	Port	Port	Port	Port	
1	1	0	1	SEG	SEG	Port	Port	Port	Port	Port	
1	1	1	0	SEG	Port	Port	Port	Port	Port	Port	
1	1	1	1	Port	Port	Port	Port	Port	Port	Port	

Duty select, common function select

Bit 7 DTS1	Bit 6 DTS0	Bit 5 CMX	Duty Cycle	Common Drivers	Notes
0	0	0	Static	COM_1	
		1		COM_4 to COM_1	COM_4 to COM_2 output the same waveform as COM_1
0	1	0	1/2 duty	COM_2 to COM_1	
		1		COM_4 to COM_1	COM_4 outputs the same waveform as COM_3 and COM_2 outputs the same waveform as COM_1
1	0	0	1/3 duty	COM_3 to COM_1	
		1		COM_4 to COM_1	COM_4 outputs a non-selected waveform
1	1	0	1/4 duty	COM_4 to COM_1	—
		1			

Bit	7	6	5	4	3	2	1	0
	—	PSW	ACT	DISP	CKS3	CKS2	CKS1	CKS0
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Frame frequency select

Bit 3	Bit 2	Bit 1	Bit 1	Operating Clock
CKS3	CKS2	CKS1	CKS0	
0	*	0	0	øw
0	*	0	1	øw/2
0	*	1	*	øw/4
1	0	0	0	ø/2
1	0	0	1	ø/4
1	0	1	0	ø/8
1	0	1	1	ø/16
1	1	0	0	ø/32
1	1	0	1	ø/64
1	1	1	0	ø/128
1	1	1	1	ø/256

* : Don’t care

Display data control

0	Blank data is displayed
1	LCD RAM data is displayed

Display function activate

0	LCD controller/driver operation halted
1	LCD controller/driver operates

LCD drive power supply on/off control

0	LCD drive power supply off
1	LCD drive power supply on

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

A waveform/B waveform switching control

0	Drive using A waveform
1	Drive using B waveform

Bit	7	6	5	4	3	2	1	0
	CKS	—	—	—	CH3	CH2	CH1	CH0
Initial value	0	0	1	1	0	0	0	0
Read/Write	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	0	0	AN_0
			1	AN_1
		1	0	AN_2
			1	AN_3
1	0	0	0	Use prohibited
			1	
		1	0	
			1	

*: Don't care

Clock select

Bit 7		Conversion Time	
CKS	Conversion Period	ø = 1 MHz	ø = 5 MHz
0	62/ø	62 µs	12.4 µs
1	31/ø	31 µs	*

*: Operation is not guaranteed with a conversion time of less than 12.4 µs.
Select a setting that gives a conversion time of at least 12.4 µs.

ADRRH—A/D result register H H'C4 A/D converter
ADRRL—A/D result register L H'C5

ADRRH

Bit	7	6	5	4	3	2	1	0
	ADR9	ADR8	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2
Initial value	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed
Read/Write	R	R	R	R	R	R	R	R

A/D conversion result

ADRRL

Bit	7	6	5	4	3	2	1	0
	ADR1	ADR0	—	—	—	—	—	—
Initial value	Not fixed	Not fixed	—	—	—	—	—	—
Read/Write	R	R	—	—	—	—	—	—

A/D conversion result

ADSR—A/D start register H'C7 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 status flag

0	Read	Indicates completion of A/D conversion
	Write	Stops A/D conversion
1	Read	Indicates A/D conversion in progress
	Write	Starts A/D conversion

Bit	7	6	5	4	3	2	1	0
	—	—	POF1	—	—	—	—	IRQ0
Initial value	1	1	0	1	1	—	—	0
Read/Write	—	—	R/W	—	—	W	W	R/W

$P4_3/\overline{IRQ0}$ pin function switch

0	Functions as $P4_3$ I/O pin
1	Functions as $\overline{IRQ_0}$ input pin

$P3_5$ pin output buffer PMOS on/off control

0	CMOS output
1	NMOS open-drain output

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

$P3_1$/TMOFL pin function switch

0	Functions as $P3_1$ I/O pin
1	Functions as TMOFL output pin

$P3_2$/TMOFH pin function switch

0	Functions as $P3_2$ I/O pin
1	Functions as TMOFH output pin

$P3_6$/AEVH pin function switch

0	Functions as $P3_6$ I/O pin
1	Functions as AEVH input pin

$P3_7$/AEVL pin function switch

0	Functions as $P3_7$ I/O pin
1	Functions as AEVL input pin

PMR5—Port mode register 5 H'CC I/O port

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

$P5_n/\overline{WKP}_n/SEG_{n+1}$ pin function switch

0	Functions as $P5_n$ I/O pin
1	Functions as $\overline{WKP}_n$ input pin

PWCR2—PWM2 Control Register H'CD 10-bit PWM

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	PWCR21	PWCR20
Initial value	1	1	1	1	1	1	0	0
Read/Write	—	—	—	—	—	—	W	W

Cloxk select

0	0	The input clock is ø (tø* = 1/ø) The conversion period is 512/ø, with a minimum modulation width of 1/2 ø
	1	The input clock is ø/2 (tø* = 2/ø) The conversion period is 1,024/ø, with a minimum modulation width of 1/ø
1	0	The input clock is ø/4 (tø* = 4/ø) The conversion period is 2,048/ø, with a minimum modulation width of 2/ø
	1	The input clock is ø/8 (tø* = 8/ø) The conversion period is 4,096/ø, with a minimum modulation width of 4/ø

Note: * tø: Period of PWM2 input clock

PWDRU2—PWM2 Data Register U H'CE 10-bit PWM

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	PWDRU21	PWDRU20
Initial value	1	1	1	1	1	1	0	0
Read/Write	—	—	—	—	—	—	W	W

Upper 2 bits of PWM2 waveform generation data

PWDRL2—PWM2 Data Register L H'CF **10-bit PWM**

Bit	7	6	5	4	3	2	1	0
	PWDRL27	PWDRL26	PWDRL25	PWDRL24	PWDRL23	PWDRL22	PWDRL21	PWDRL20
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Lower 8 bits of PWM2 waveform generation data

PWCR1—PWM1 control register H'D0 **10-bit PWM**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	$PWCR1_1$	$PWCR1_0$
Initial value	1	1	1	1	1	1	0	0
Read/Write	—	—	—	—	—	—	W	W

Clock select

0	The input clock is ø (tø* = 1/ø) The conversion period is 512/ø, with a minimum modulation width of 1/2ø
	The input clock is ø/2 (tø* = 2/ø) The conversion period is 1,024/ø, with a minimum modulation width of 1/ø
1	The input clock is ø/4 (tø* = 4/ø) The conversion period is 2,048/ø, with a minimum modulation width of 2/ø
	The input clock is ø/8 (tø* = 8/ø) The conversion period is 4,096/ø, with a minimum modulation width of 4/ø

Note: * tø: Period of PWM input clock

PWDRU1—PWM1 data register U — H'D1 — 10-bit PWM

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	$PWDUR1_1$	$PWDRU1_0$
Initial value	1	1	1	1	1	1	0	0
Read/Write	—	—	—	—	—	—	W	W

Bits 1–0: Upper 2 bits of data for generating PWM1 waveform

PWDRL1—PWM1 data register L — H'D2 — 10-bit PWM

Bit	7	6	5	4	3	2	1	0
	$PWDRL1_7$	$PWDRL1_6$	$PWDRL1_5$	$PWDRL1_4$	$PWDRL1_3$	$PWDRL1_2$	$PWDRL1_1$	$PWDRL1_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Bits 7–0: Lower 8 bits of data for generating PWM1 waveform

PDR3—Port data register 3 — H'D6 — I/O ports

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

Bits 7–1: Stores data of port 3 pins

PDR4—Port data register 4 — H'D7 — I/O ports

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

Bits 2–0: Stores data of port 4 pins

Bit 3: Reads $P4_3$ pin state

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

Stores data of port 5 pins

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

Stores data of port 6 pins

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

Stores data of port 7 pins

PDR8—Port data register 8 H'DB I/O ports

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	$P8_0$
Initial value	—	—	—	—	—	—	—	0
Read/Write	—	—	—	—	—	—	—	R/W

Stores data of $P8_0$ pin

PDR9—Port data register 9 **H'DC** **I/O ports**

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

Stores data of port 9 pins

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	PA_0
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

Stores data of port A pins

PDRB—Port data register B **H'DE** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PB_3	PB_2	PB_1	PB_0
Initial value								
Read/Write	—	—	—	—	R	R	R	R

Reads states of port B pins

PUCR3—Port pull-up control register 3 **H'E1** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PUCR3_7$	$PUCR3_6$	$PUCR3_5$	$PUCR3_4$	$PUCR3_3$	$PUCR3_2$	$PUCR3_1$	—
Initial value	0	0	0	0	0	0	0	—
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	W

PUCR5—Port pull-up control register 5 **H'E2** **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'E3** **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

PCR3—Port control register 3 **H'E6** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR3_7$	$PCR3_6$	$PCR3_5$	$PCR3_4$	$PCR3_3$	$PCR3_2$	$PCR3_1$	—
Initial value	0	0	0	0	0	0	0	—
Read/Write	W	W	W	W	W	W	W	W

Port 3 input/output select

0	Input pin
1	Output pin

PCR4—Port control register 4 **H'E7** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PCR4_2$	$PCR4_1$	$PCR4_0$
Initial value	1	1	1	1	1	0	0	0
Read/Write	—	—	—	—	—	W	W	W

Port 4 input/output select

0	Input pin
1	Output pin

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
	—	—	—	—	—	—	—	$PCR8_0$
Initial value	—	—	—	—	—	—	—	0
Read/Write	W	W	W	W	W	W	W	W

Port 8 input/output select

0	Input pin
1	Output pin

PMR9—Port mode register 9 H'EC I/O ports

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PIOFF	—	PWM2	PWM1
Initial value	1	1	1	1	0	—	0	0
Read/Write	—	—	—	—	R/W	W	R/W	R/W

P90/PWM1 pin function switch

0	Functions as P90 output pin
1	Functions as PWM1 output pin

P91/PWM2 pin function switch

0	Functions as P91 output pin
1	Functions as PWM2 output pin

P92 to P90 step-up circuit control

0	Large-current port step-up circuit is turned on
1	Large-current port step-up circuit is turned off

PCRA—Port control register A **H'ED** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	$PCRA_3$	$PCRA_2$	$PCRA_1$	$PCRA_0$
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	W	W	W	W

Port A input/output select

0	Input pin
1	Output pin

PMRB—Port mode register B **H'EE** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	IRQ1	—	—	—
Initial value	1	1	1	1	0	1	1	1
Read/Write	—	—	—	—	R/W	—	—	—

$PB_3/AN_3/\overline{IRQ_1}$ pin function switch

0	Functions as PB_3/AN_3 input pin
1	Functions as $\overline{IRQ_1}$ input pin

Bit	7	6	5	4	3	2	1	0
	SSBY	STS2	STS1	STS0	LSON	—	MA1	MA0
Initial value	0	0	0	0	0	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	—	R/W	R/W

Active (medium-speed) mode clock select

0	0	$ø_{OSC}$/16
	1	$ø_{OSC}$/32
1	0	$ø_{OSC}$/64
	1	$ø_{OSC}$/128

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 = 1,024 states
		1	Wait time = 2,048 states
1	0	0	Wait time = 4,096 states
		1	Wait time = 2 states
	1	0	Wait time = 8 states
		1	Wait time = 16 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

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

Subactive mode clock select

0	0	$ø_W/8$
	1	$ø_W/4$
1	*	$ø_W/2$

*: Don't care

Medium speed on flag

0	Operates in active (high-speed) mode
1	Operates in active (medium-speed) mode

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 $ø_{OSC}/4$

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

$\overline{IRQ_0}$ edge select

0	Falling edge of $\overline{IRQ_0}$ pin input is detected
1	Rising edge of $\overline{IRQ_0}$ pin input is detected

$\overline{IRQ_1}$ edge select

0	Falling edge of $\overline{IRQ_1}$ pin input is detected
1	Rising edge of $\overline{IRQ_1}$ pin input is detected

Bit	7	6	5	4	3	2	1	0
	IENTA	—	IENWP	—	—	IENEC2	IEN1	IEN0
Initial value	0	—	0	—	—	0	0	0
Read/Write	R/W	W	R/W	W	W	R/W	R/W	R/W

$\overline{IRQ_1}$ to $\overline{IRQ_0}$ interrupt enable

0	Disables $\overline{IRQ_1}$ to $\overline{IRQ_0}$ interrupt, requests
1	Enables $\overline{IRQ_1}$ to $\overline{IRQ_0}$ interrupt requests

IRQAEC interrupt enable

0	Disables IRQAEC interrupt requests
1	Enables IRQAEC interrupt requests

Wakeup interrupt enable

0	Disables $\overline{WKP_7}$ to $\overline{WKP_0}$ interrupt requests
1	Enables $\overline{WKP_7}$ to $\overline{WKP_0}$ interrupt requests

Timer A interrupt enable

0	Disables timer A interrupt requests
1	Enables timer A interrupt requests

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

Asynchronous event counter interrupt enable

0	Disables asynchronous event counter interrupt requests
1	Enables asynchronous event counter interrupt requests

Timer FL interrupt enable

0	Disables timer FL interrupt requests
1	Enables timer FL interrupt requests

Timer FH interrupt enable

0	Disables timer FH interrupt requests
1	Enables timer FH interrupt requests

A/D converter interrupt enable

0	Disables A/D converter interrupt requests
1	Enables A/D converter interrupt requests

Direct transition interrupt enable

0	Disables direct transition interrupt requests
1	Enables direct transition interrupt requests

Bit	7	6	5	4	3	2	1	0
	IRRTA	—	—	—	—	IRREC2	IRRI1	IRRI0
Initial value	0	—	1	—	—	0	0	0
Read/Write	R/W*	W	—	W	W	R/W*	R/W*	R/W*

IRQ1 to IRQ0 interrupt request flags

0	Clearing conditions: When IRRIn = 1, it is cleared by writing 0
1	Setting conditions: When pin $\overline{\text{IRQn}}$ is designated for interrupt input and the designated signal edge is input

(n = 1 or 0)

IRQAEC interrupt request flag

0	Clearing conditions: When IRREC2 = 1, it is cleared by writing 0
1	Setting conditions: When pin IRQAEC is designated for interrupt input and the designated signal edge is input

Timer A interrupt request flag

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

Note: * Bits 7 and 2 to 0 can only be written with 0, for flag clearing.

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

Asynchronous event counter interrupt request flag

0	Clearing conditions: When IRREC = 1, it is cleared by writing 0
1	Setting conditions: When the asynchronous event counter value overflows

Timer FL interrupt request flag

0	Clearing conditions: When IRRTFL = 1, it is cleared by writing 0
1	Setting conditions: When counter FL and output compare register FL match in 8-bit timer mode

Timer FH interrupt request flag

0	Clearing conditions: When IRRTFH = 1, it is cleared by writing 0
1	Setting conditions: When counter FH and output compare register FH match in 8-bit timer mode, or when 16-bit counters FL and FH and output compare registers FL and FH match in 16-bit timer mode

A/D converter interrupt request flag

0	Clearing conditions: When IRRAD = 1, it is cleared by writing 0
1	Setting conditions: When the A/D converter completes conversion and ADSF is reset

Direct transition interrupt request flag

0	Clearing conditions: When IRRDT = 1, it is cleared by writing 0
1	Setting conditions: When a SLEEP instruction is executed while DTON is set to 1, and a direct transition is made

Note: * Bits 7, 6, 3, 2, and 0 can only be written with 0, for flag clearing.

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 register

0	Clearing conditions: When IWPFn = 1, it is cleared by writing 0
1	Setting conditions: When pin $\overline{\text{WKPn}}$ is designated for wakeup input and a falling edge is input at that pin

(n = 7 to 0)

Note: * All bits can only be written with 0, for flag clearing.

Bit	7	6	5	4	3	2	1	0
	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	R/W	R/W	—	R/W	—	R/W

Timer A module standby mode control (bit 0, TACKSTP)

0	Timer A is set to module standby mode
1	Timer A module standby mode is cleared

Timer F module standby mode control (bit 2, TFCKSTP)

0	Timer F is set to module standby mode
1	Timer F module standby mode is cleared

A/D converter module standby mode control (bit 4, ADCKSTP)

0	A/D converter is set to module standby mode
1	A/D converter module standby mode is cleared

SCI3 module standby mode control (bit 5, S32CKSTP)

0	SCI3 is set to module standby mode
1	SCI3 module standby mode is cleared

Bit	7	6	5	4	3	2	1	0
	—	—	—	PW2CKSTP	AECKSTP	—	PW1CKSTP	LDCKSTP
Initial value	1	1	1	1	1	1	1	1
Read/Write	—	—	—	R/W	R/W	—	R/W	R/W

LCD module standby mode control

0	LCD is set to module standby mode
1	LCD module standby mode is cleared

PWM1 module standby mode control

0	PWM1 is set to module standby mode
1	PWM1 module standby mode is cleared

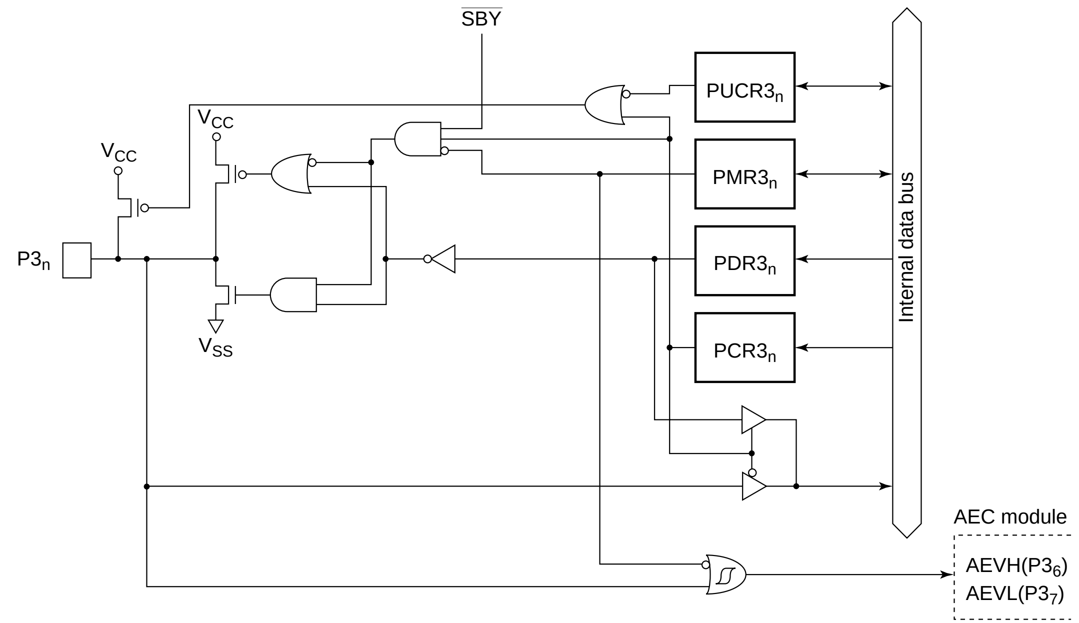
Asynchronous event counter module standby mode control

0	Asynchronous event counter is set to module standby mode
1	Asynchronous event counter module standby mode is cleared

PWM2 module standby mode control

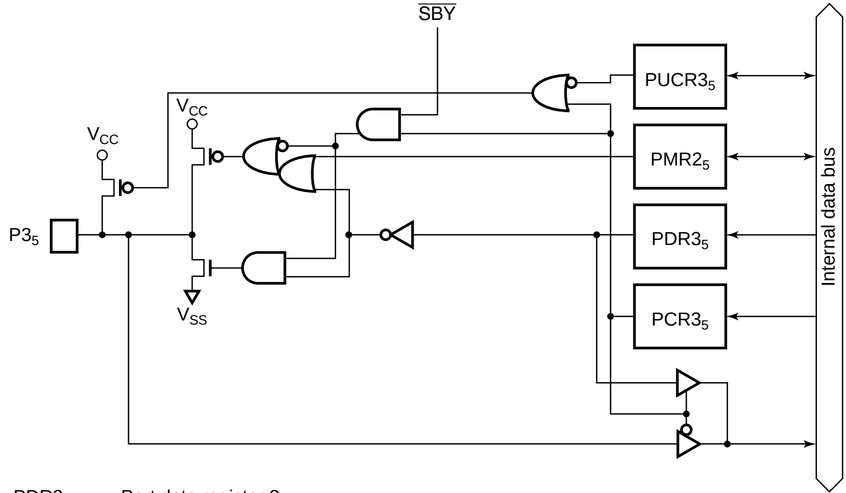
0	PWM2 is set to module standby mode
1	PWM2 module standby mode is cleared

C.1 Block Diagrams of Port 3



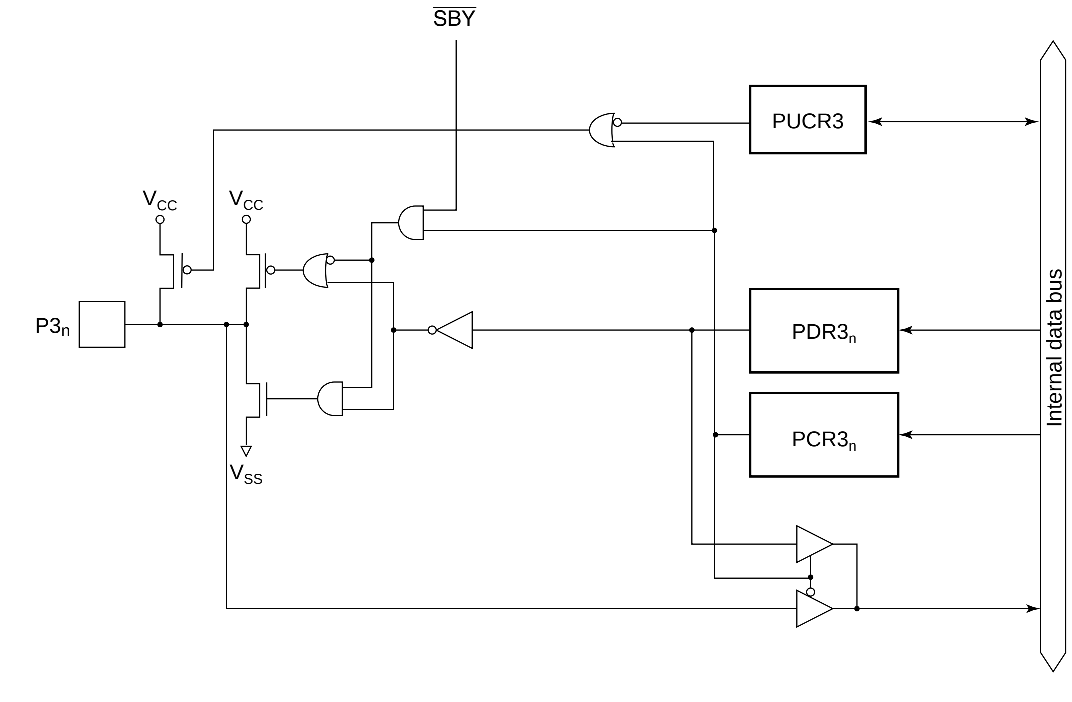
PDR3: Port data register 3
PCR3: Port control register 3
PMR3: Port mode register 3
PUCR3: Port pull-up control register 3
n=7 or 6

Figure C.1 (a) Port 3 Block Diagram (Pins $P3_7$ and $P3_6$)



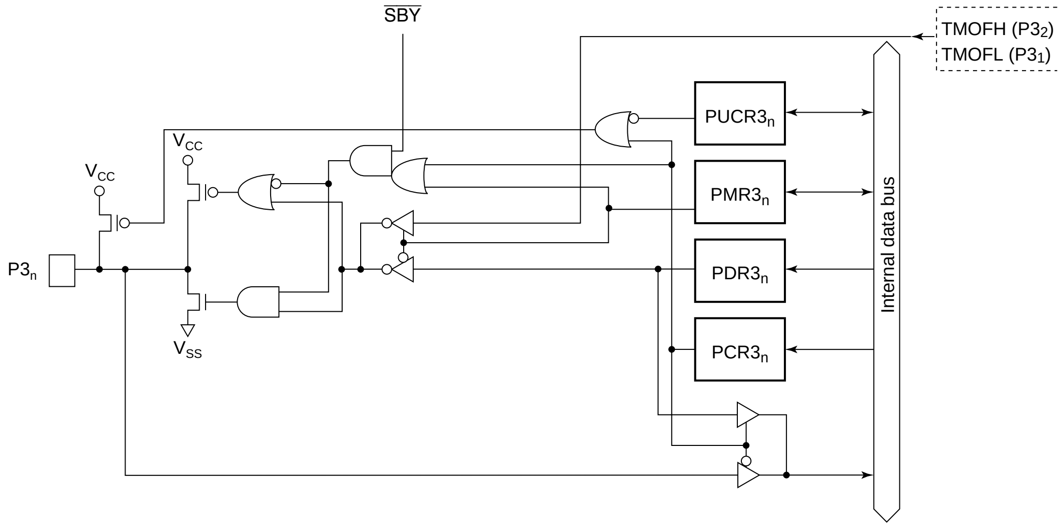
PDR3: Port data register 3
PCR3: Port control register 3
PUCR3: Port pull-up control register 3
PMR2 Port mode register 2

Figure C.1 (b) Port 3 Block Diagram (Pin P3$_5$)



PDR3: Port data register 3
PCR3: Port control register 3
n = 4 or 3

Figure C.1 (c) Port 3 Block Diagram (Pins $P3_4$ and $P3_3$)



PDR3: Port data register 3

PCR3: Port control register 3

PMR3: Port mode register 3

PUCR3: Port pull-up control register 3

n = 2 or 1

Figure C.1 (d) Port 3 Block Diagram (Pins $P3_2$ and $P3_1$)

C.2 Block Diagrams of Port 4

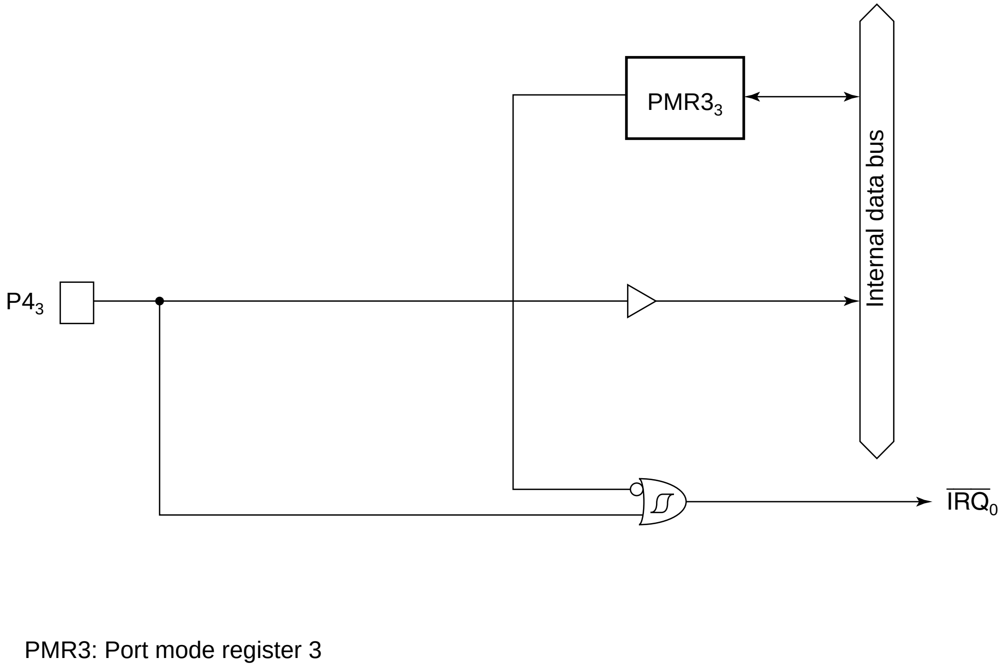


Figure C.2 (a) Port 4 Block Diagram (Pin $P4_3$)

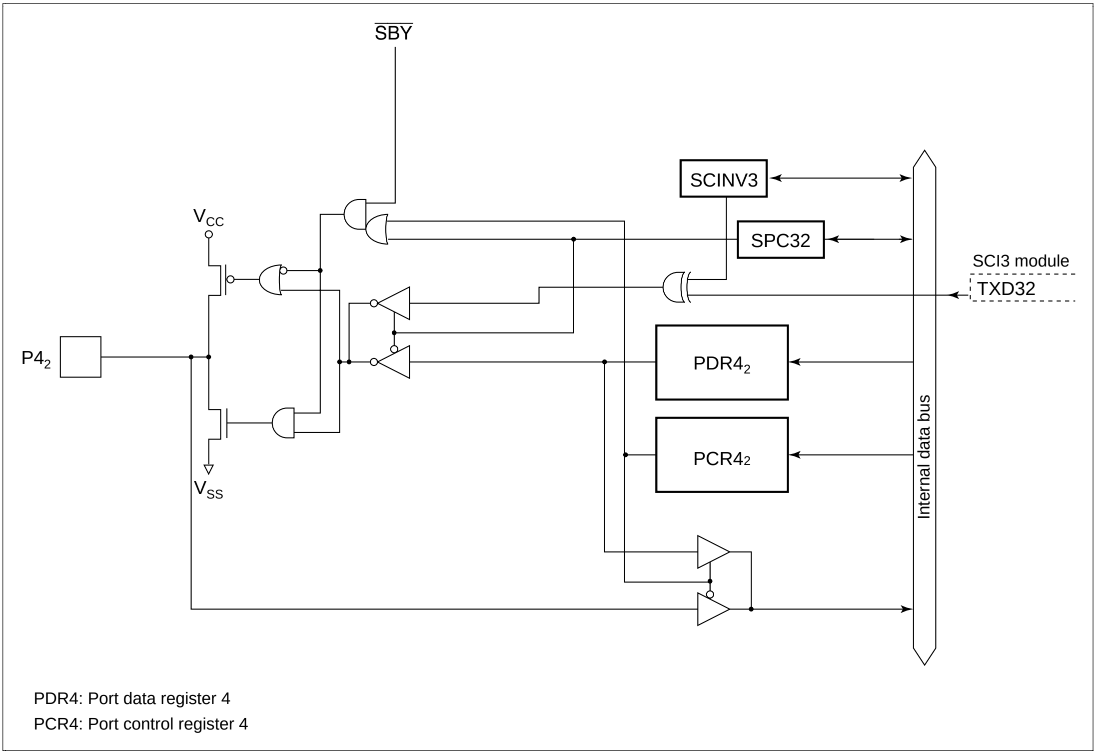


Figure C.2 (b) Port 4 Block Diagram (Pin $P4_2$)

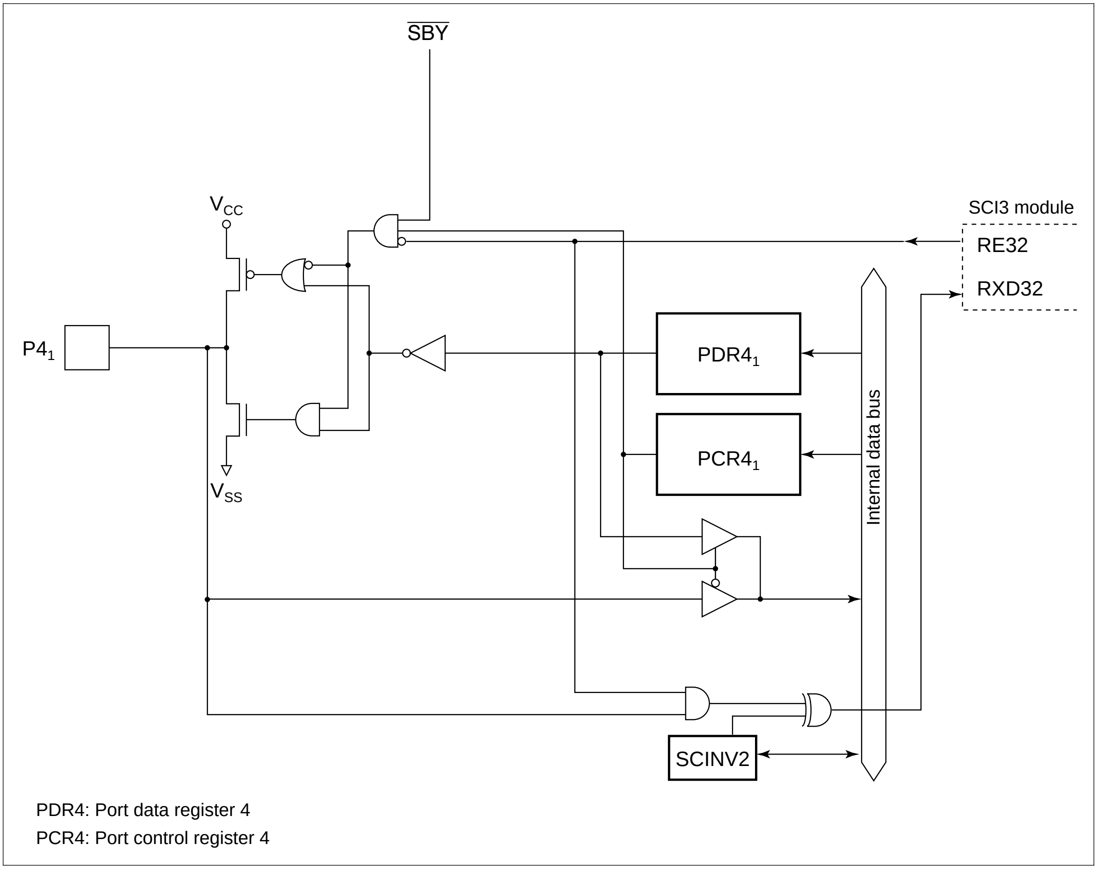


Figure C.2 (c) Port 4 Block Diagram (Pin $P4_1$)

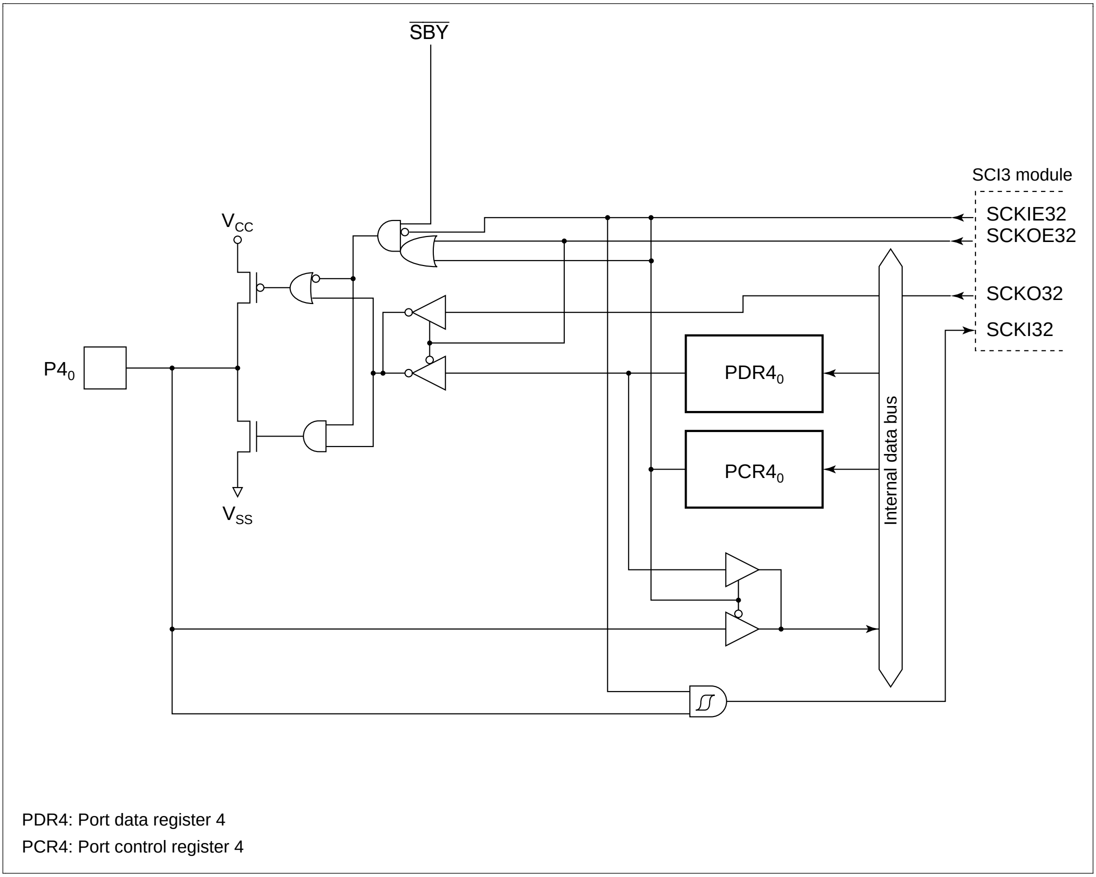
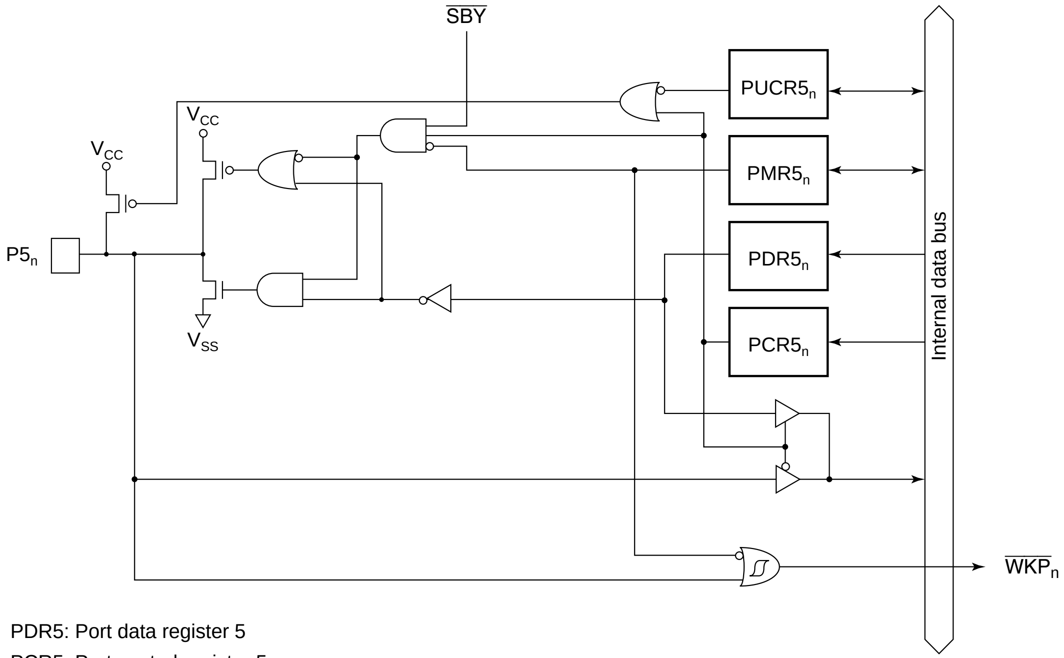


Figure C.2 (d) Port 4 Block Diagram (Pin $P4_0$)

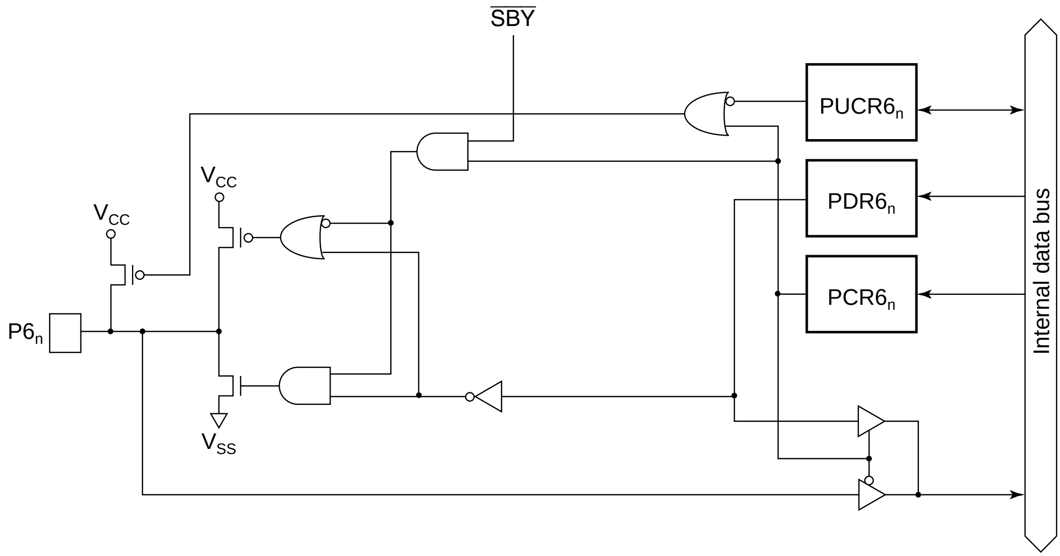
C.3 Block Diagram of Port 5



PDR5: Port data register 5
PCR5: Port control register 5
PMR5: Port mode register 5
PUCR5: Port pull-up control register 5
n = 7 to 0

Figure C.3 Port 5 Block Diagram

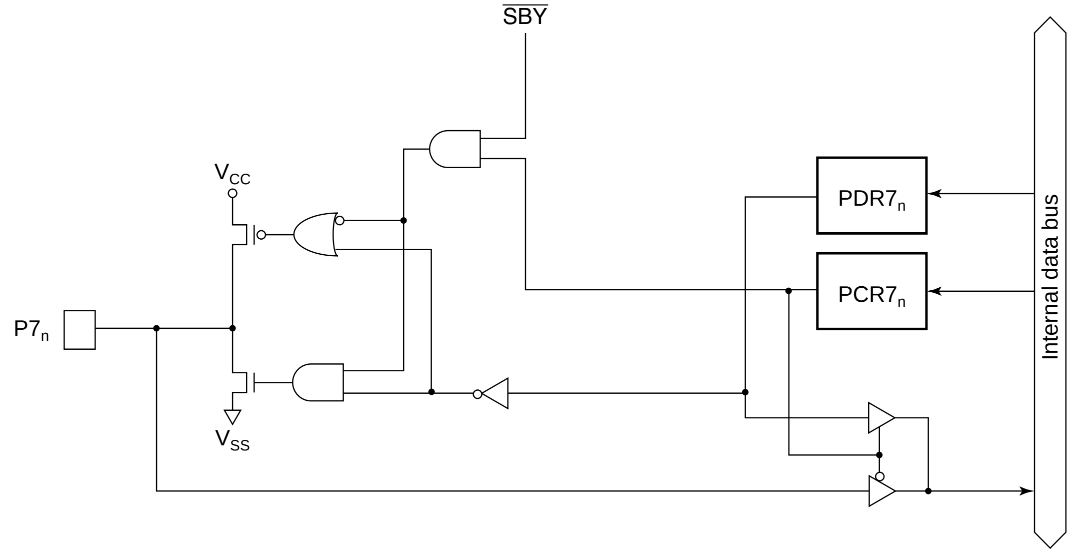
C.4 Block Diagram of Port 6



PDR6: Port data register 6
PCR6: Port control register 6
PUCR6: Port pull-up control register 6
n = 7 to 0

Figure C.4 Port 6 Block Diagram

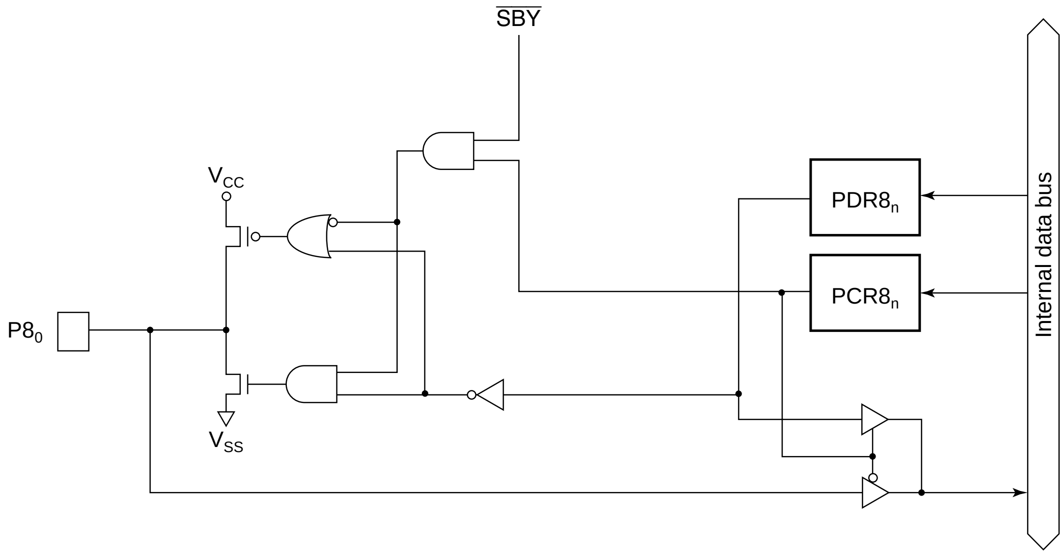
C.5 Block Diagram of Port 7



PDR7: Port data register 7
PCR7: Port control register 7
n = 7 to 0

Figure C.5 Port 7 Block Diagram

C.6 Block Diagrams of Port 8



PDR8: Port data register 8
PCR8: Port control register 8

Figure C.6 Port 8 Block Diagram (Pin P8$_0$)

C.7 Block Diagrams of Port 9

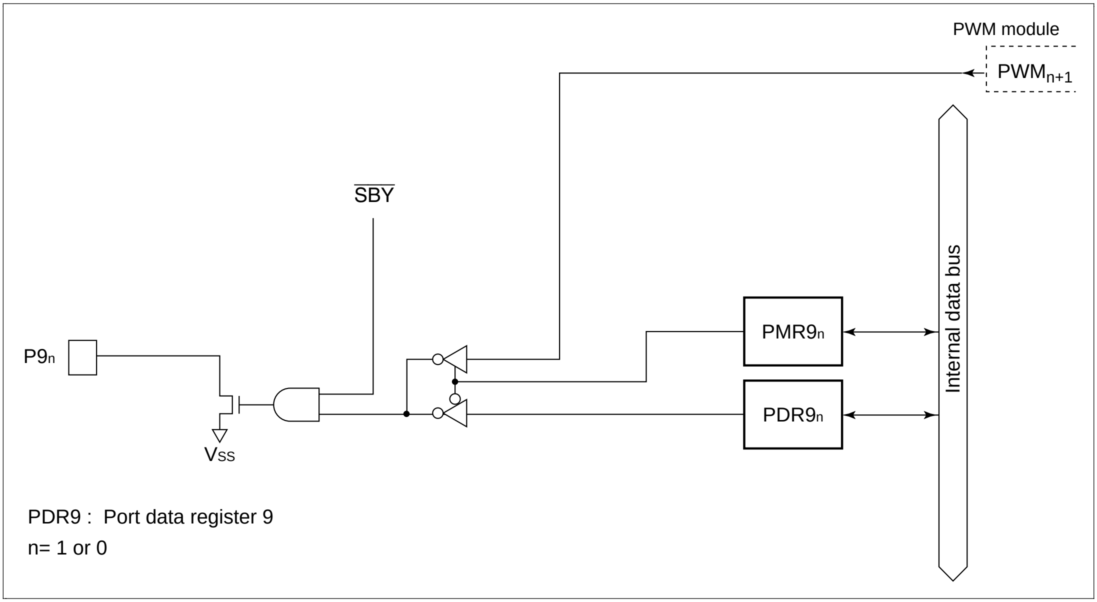


Figure C.7 (a) Port 9 Block Diagram (Pins $P9_1$ and $P9_0$)

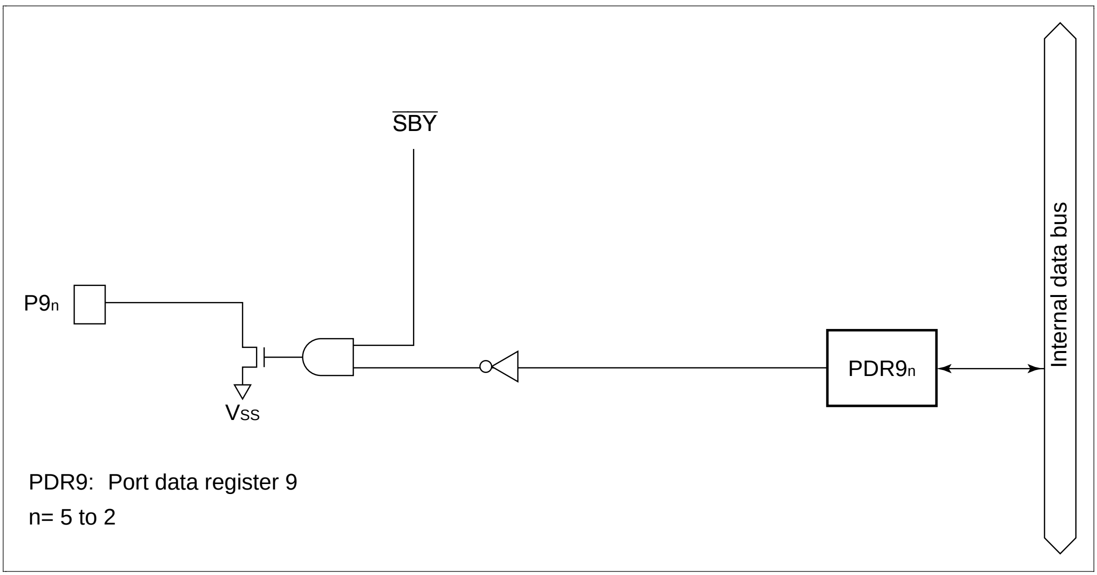
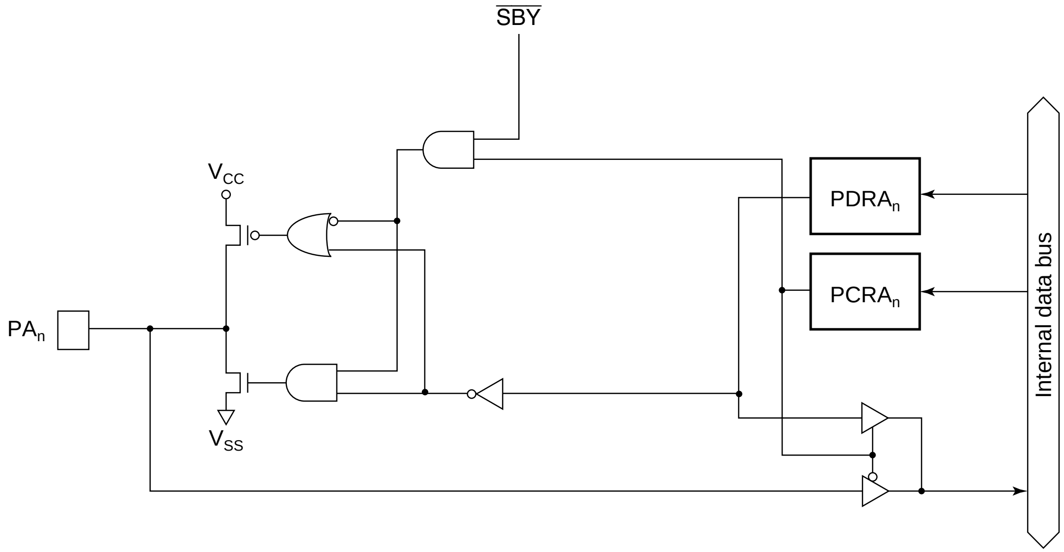


Figure C.7 (b) Port 9 Block Diagram (Pins $P9_5$ to $P9_2$)

C.8 Block Diagram of Port A



PDRA: Port data register A
PCRA: Port control register A
n = 3 to 0

Figure C.8 Port A Block Diagram

C.9 Block Diagram of Port B

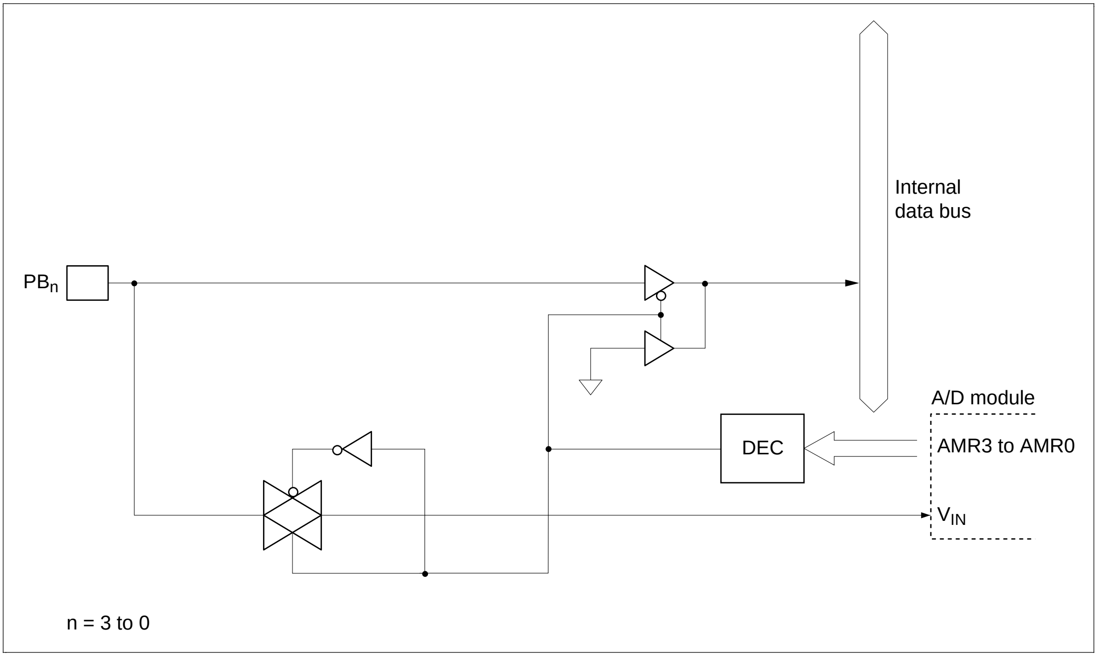


Figure C.9 Port B Block Diagram

Appendix D Port States in the Different Processing States

Table D.1 Port States Overview

Port	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P3_7$ to $P3_1$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P4_3$ to $P4_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P5_7$ to $P5_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P6_7$ to $P6_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P7_7$ to $P7_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P8_7$ to $P8_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P9_5$ to $P9_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
PA_3 to PA_0	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
PB_3 to PB_0	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 List of Product Codes

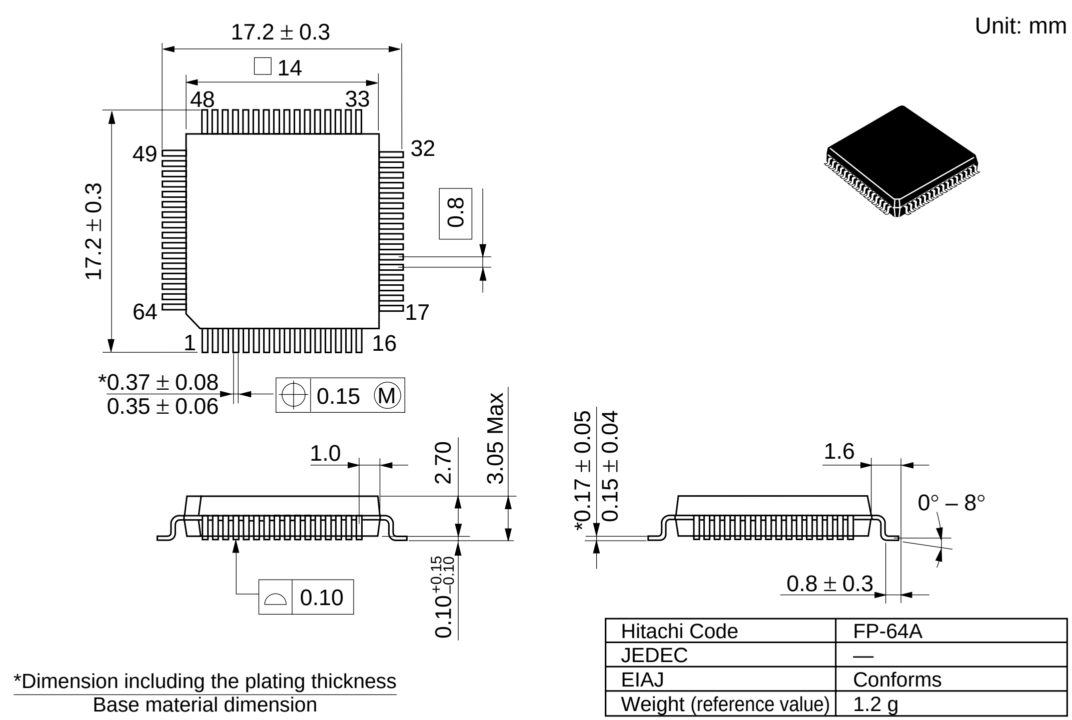
Table E.1 H8/3802 Series Product Code Lineup

	Product Type		Product Code	Mark Code	Package(Hitachi Package Code)
H8/3802 series	H8/3802	Mask ROM versions	HD6433802H	HD6433802 (***) H	64-pin QFP (FP-64A)
			HD64433802FP	HD6433802 (***) FP	64-pin LQFP (FP-64E)
			HD6433802P	HD6433802 (***) P	64-pin DILP (DP-64S)
		ZTAT versions	HD6473802H	HD6473802H	64-pin QFP (FP-64A)
			HD6473802FP	HD6473802FP	64-pin LQFP (FP-64E)
			HD6473802P	HD6473802P	64-pin DILP (DP-64S)
	H8/3801	Mask ROM versions	HD6433801H	HD6433801 (***) H	64-pin QFP (FP-64A)
			HD6433801FP	HD6433801 (***) FP	64-pin LQFP (FP-64E)
			HD6433801P	HD6433801 (***) P	64-pin DILP (DP-64S)
	H8/3800	Mask ROM versions	HD6433800H	HD6433800 (***) H	64-pin QFP (FP-64A)
			HD6433800FP	HD6433800 (***) FP	64-pin LQFP (FP-64E)
			HD6433800P	HD6433800 (***) P	64-pin DILP (DP-64S)

Note: For mask ROM versions, (***) is the ROM code.

Appendix F Package Dimensions

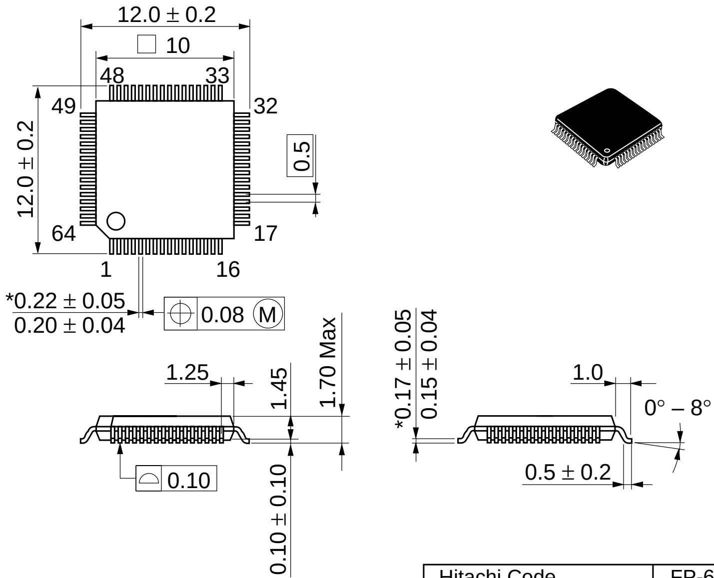
Dimensional drawings of H8/3802 Series packages FP-64A, FP-64E, and DP-64S are shown in figures F.1, F.2, and F.3 below.



Hitachi Code	FP-64A
JEDEC	—
EIAJ	Conforms
Weight (reference value)	1.2 g

Figure F.1 FP-64A Package Dimensions

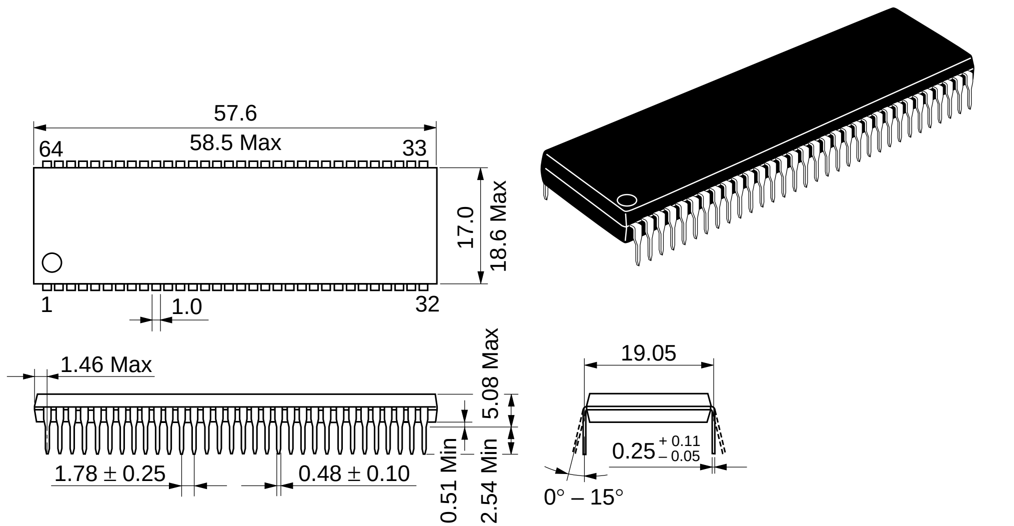
Unit: mm



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



Hitachi Code	DP-64S
JEDEC	—
EIAJ	Conforms
Weight (reference value)	8.8 g

Figure F.3   DP-64S Package Dimensions

H8/3802 Series Hardware Manual

Publication Date:	1st Edition, November 1999
	2nd Edition, January 2001
Published by:	Electronic Devices Sales & Marketing Group
	Semiconductor & Integrated Circuits
	Hitachi, Ltd.
Edited by:	Technical Documentation Group
	Hitachi Kodaira Semiconductor Co., Ltd.