H8/3644 Series

H8/3644

HD6473644, HD6433644

H8/3643

HD6433643

H8/3642

HD6433642

H8/3641

HD6433641

H8/3640

HD6433640

H8/3644F-ZTAT™

HD64F3644

H8/3643F-ZTAT™

HD64F3643

H8/3642AF-ZTAT™

HD64F3642A

Hardware Manual

HITACHI

ADE-602-087C
Rev. 4.0
08/08/98
Hitachi, Ltd.
MC-Setsu

Hitachi semiconductor

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

Preface

The H8/300L Series of single-chip microcomputers has 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 H8/3644 Series has a system-on-a-chip architecture that includes such peripheral functions as a D/A converter, five timers, a 14-bit PWM, a two-channel serial communication interface, and an A/D converter. This makes it ideal for use in advanced control systems.

This manual describes the hardware of the H8/3644 Series. For details on the H8/3644 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 H8/3644 Series of microcomputers are equipped with a UART (Universal Asynchronous Receiver/Transmitter). Other on-chip peripheral functions include five timers, a 14-bit pulse width modulator (PWM), two serial communication interface channels, and an A/D converter, providing an ideal configuration as a microcomputer for embedding in high-level control systems. In addition to the mask ROM version, the H8/3644 is also available in a ZTAT™*[1] version with on-chip user-programmable PROM, and an F-ZTAT™*[2] version with on-chip flash memory that can be programmed on-board. Table 1 summarizes the features of the H8/3644 Series.

Notes: 1. ZTAT is a trademark of Hitachi, Ltd.
2. F-ZTAT is a registered trademark of Hitachi, Ltd.

Table 1.1 Features

Item	Description
CPU	High-speed H8/300L CPU • General-register architecture General registers: Sixteen 8-bit registers (can be used as eight 16-bit registers) • Operating speed — Max. operation speed: 5 MHz (mask ROM and ZTAT versions) 8 MHz (F-ZTAT version) — Add/subtract: 0.4 µs (operating at ø = 5 MHz) 0.25 µs (operating at ø = 8 MHz) — Multiply/divide: 2.8 µs (operating at ø = 5 MHz) 1.75 µs (operating at ø = 8 MHz) — Can run on 32.768 kHz subclock • Instruction set compatible with H8/300 CPU — Instruction length of 2 bytes or 4 bytes — Basic arithmetic operations between registers — MOV instruction for data transfer between memory and registers • Typical instructions — Multiply (8 bits × 8 bits) — Divide (16 bits ÷ 8 bits) — Bit accumulator — Register-indirect designation of bit position
Interrupts	33 interrupt sources • 12 external interrupt sources (IRQ_3 to IRQ_0, INT_7 to INT_0) • 21 internal interrupt sources
Clock pulse generators	Two on-chip clock pulse generators • System clock pulse generator: 1 to 10 MHz (1 to 16 MHz)* • Crystal or ceramic resonator: 2 to 10 MHz (2 to 16 MHz)* • External clock input: 1 to 10 MHz (1 to 16 MHz)*
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

Note: * Values in parentheses are for the F-ZTAT version.

Table 1.1 Features (cont)

Item	Description
Memory	Large on-chip memory • H8/3644: 32-kbyte ROM, 1-kbyte RAM • H8/3643: 24-kbyte ROM, 1-kbyte RAM • H8/3642: 16-kbyte ROM, 512 byte RAM (1-kbyte RAM F-ZTAT version) • H8/3641: 12-kbyte ROM, 512 byte RAM • H8/3640: 8-kbyte ROM, 512 byte RAM
I/O ports	53 pins • 45 I/O pins • 8 input pins
Timers	Five 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 ($ø_w$)* • Timer B1: 8-bit timer — Count-up timer with selection of seven internal clock signals or event input from external pin — Auto-reloading • Timer V: 8-bit timer — Count-up timer with selection of six internal clock signals or event input from external pin — Compare-match waveform output — Externally triggerable • Timer X: 16-bit timer — Count-up timer with selection of three internal clock signals or event input from external pin — Output compare (2 output pins) — Input capture (4 input pins) • Watchdog timer — Reset signal generated by 8-bit counter overflow Note: * ø and $ø_w$ are defined in section 4, Clock Pulse Generators.

Table 1.1 Features (cont)

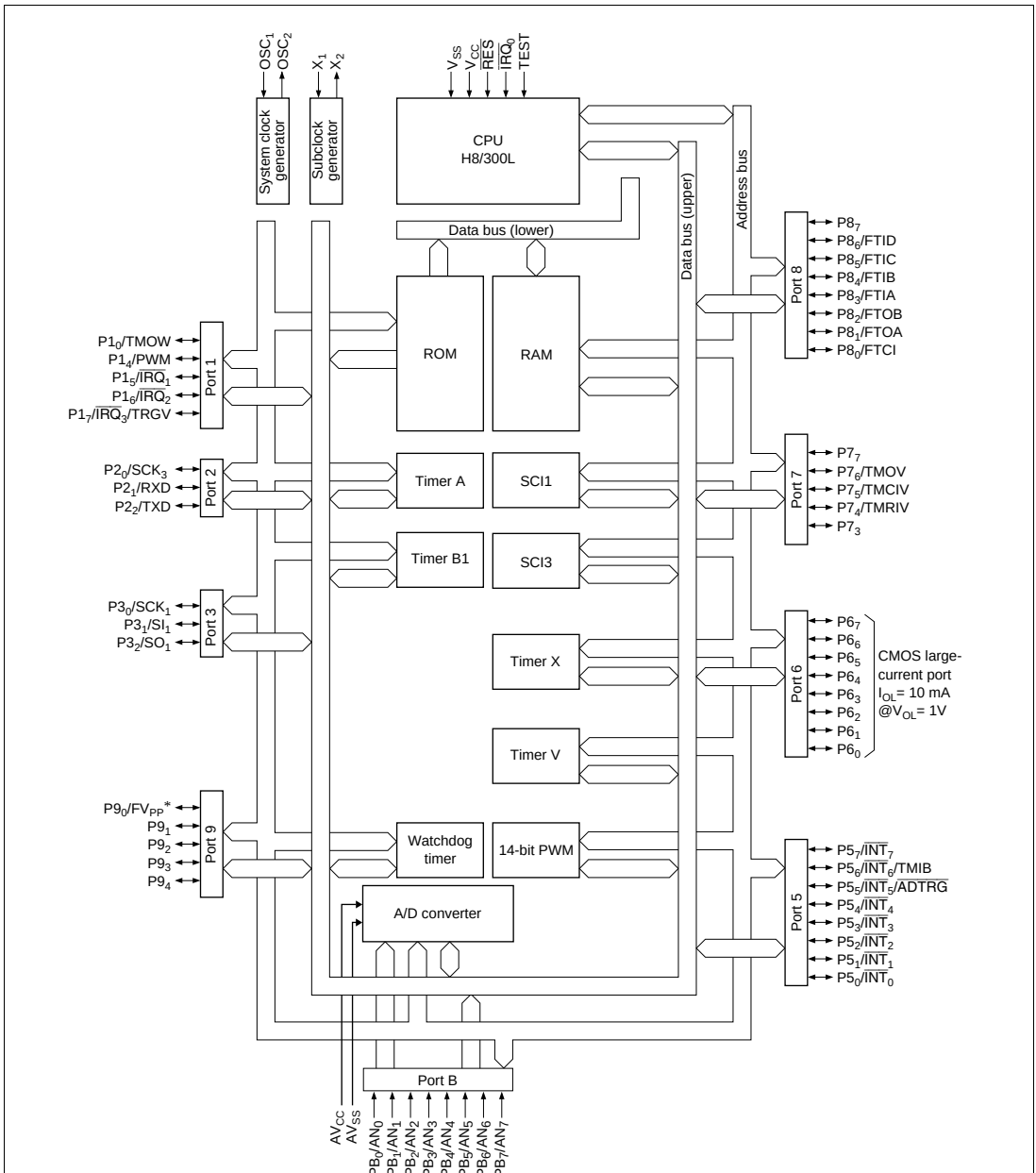
Item	Description
Serial communication interface	Two on-chip serial communication interface channels • SCI1: synchronous serial interface Choice of 8-bit or 16-bit data transfer • SCI3: 8-bit synchronous/asynchronous serial interface Incorporates multiprocessor communication function
14-bit PWM	Pulse-division PWM output for reduced ripple • Can be used as a 14-bit D/A converter by connecting to an external low-pass filter.
A/D converter	Successive approximations using a resistance ladder • 8-channel analog input pins • Conversion time: 31/ø or 62/ø per channel

Product lineup

Product Code				
Mask ROM Version	**ZTAT™ Version**	**F-ZTAT™ Version**	**Package**	**ROM/RAM Size**
HD6433644H	HD6473644H	HD64F3644H	64-pin QFP (FP-64A)	ROM: 32 kbytes RAM: 1 kbyte
HD6433644P	HD6473644P	HD64F3644P	64-pin SDIP (DP-64S)	
HD6433644W	HD6473644W	HD64F3644W	80-pin TQFP (TFP-80C)	
HD6433643H	—	HD64F3643H	64-pin QFP (FP-64A)	ROM: 24 kbytes RAM: 1 kbyte
HD6433643P	—	HD64F3643P	64-pin SDIP (DP-64S)	
HD6433643W	—	HD64F3643W	80-pin TQFP (TFP-80C)	
HD6433642H	—	HD64F3642AH	64-pin QFP (FP-64A)	ROM: 16 kbytes RAM: 512 kbytes RAM: 1 kbyte (F-ZTAT version)
HD6433642P	—	HD64F3642AP	64-pin SDIP (DP-64S)	
HD6433642W	—	HD64F3642AW	80-pin TQFP (TFP-80C)	
HD6433641H	—	—	64-pin QFP (FP-64A)	ROM: 12 kbytes RAM: 512 bytes
HD6433641P	—	—	64-pin SDIP (DP-64S)	
HD6433641W	—	—	80-pin TQFP (TFP-80C)	
HD6433640H	—	—	64-pin QFP (FP-64A)	ROM: 8 kbytes RAM: 512 bytes
HD6433640P	—	—	64-pin SDIP (DP-64S)	
HD6433640W	—	—	80-pin TQFP (TFP-80C)	

1.2 Internal Block Diagram

Figure 1.1 shows a block diagram of the H8/3644 Series.



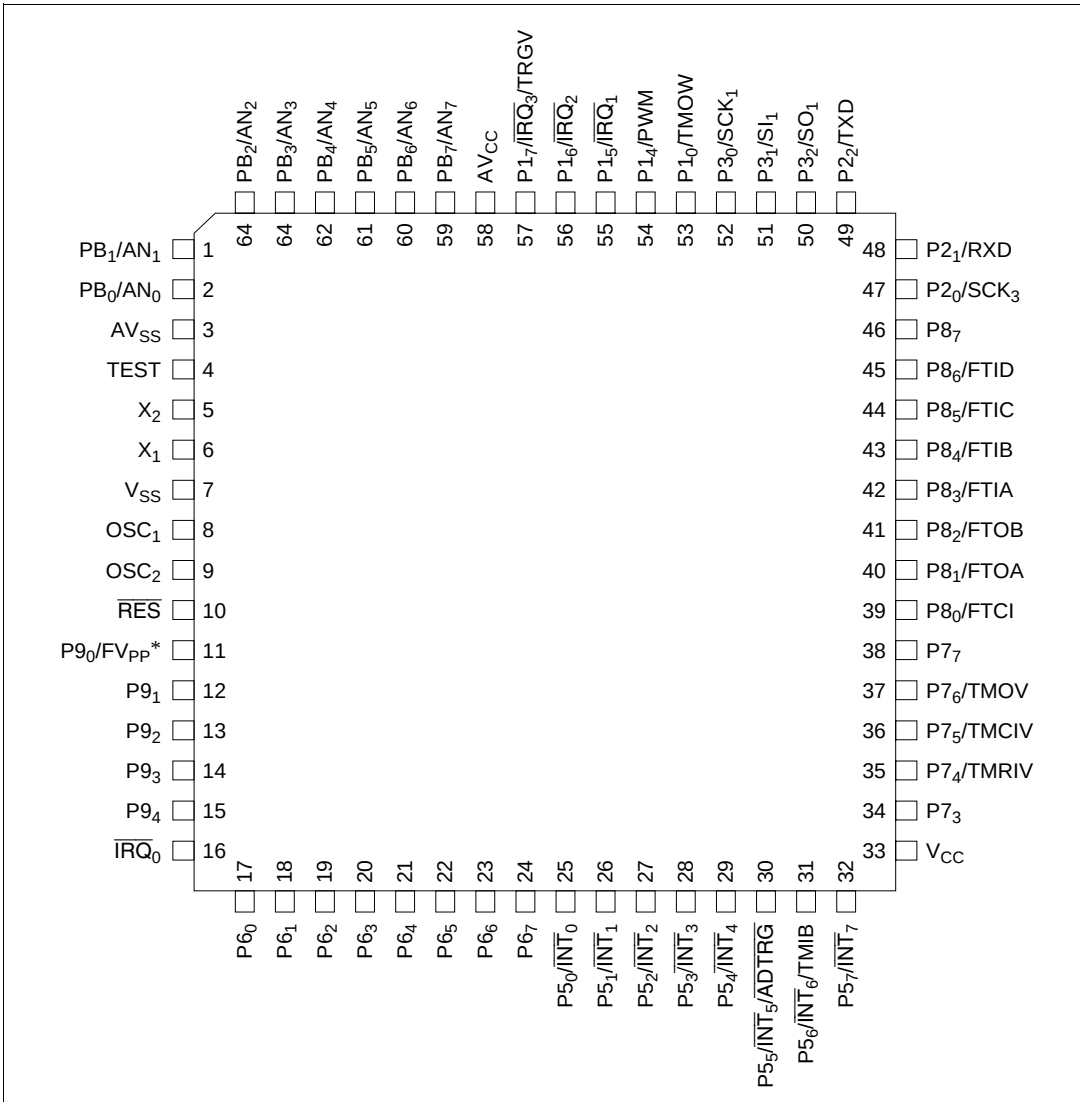
Note: * There is no $P9_0$ function in the flash memory version.

Figure 1.1 Block Diagram

1.3 Pin Arrangement and Functions

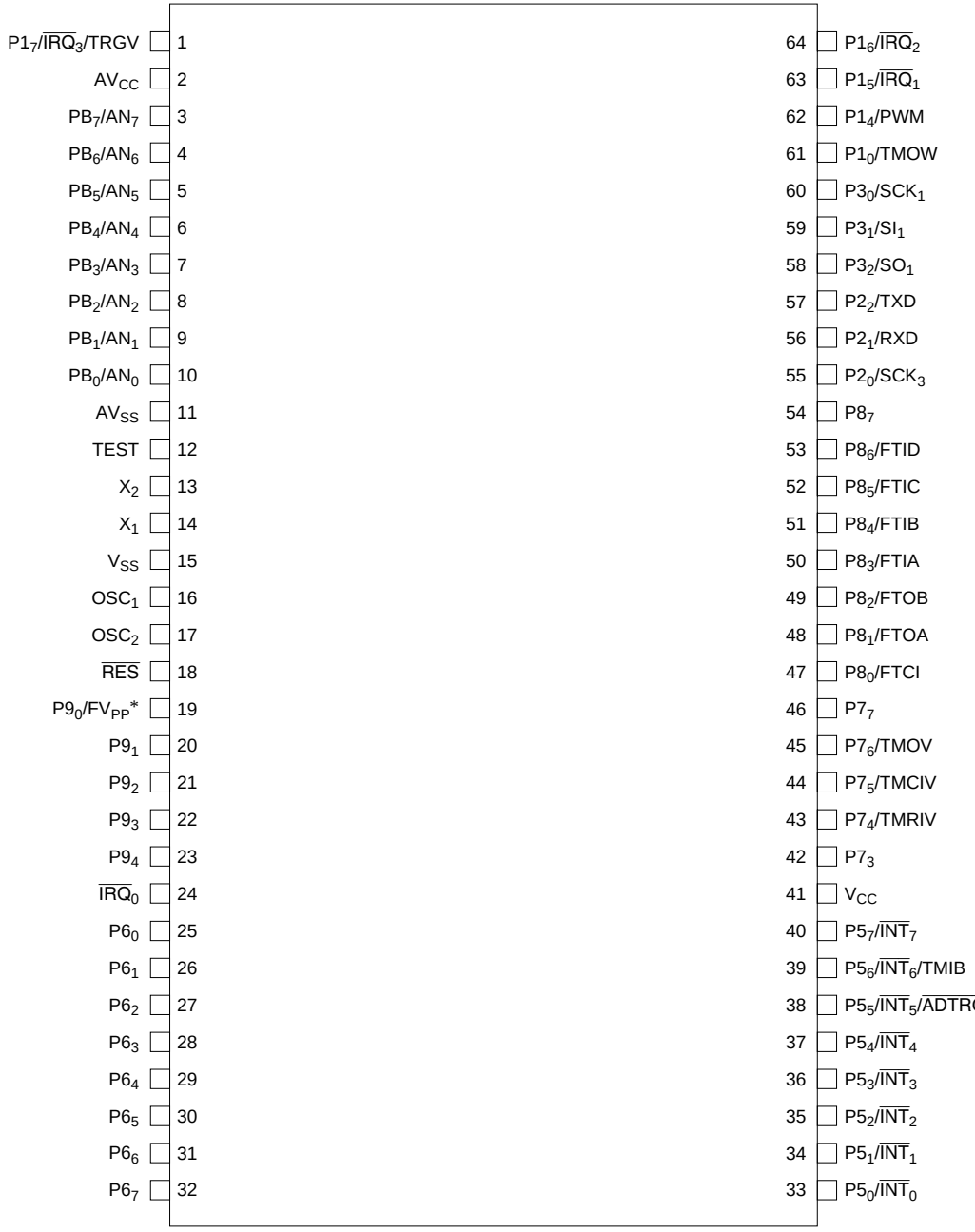
1.3.1 Pin Arrangement

The H8/3644 Series pin arrangement is shown in figures 1.2 (FP-64A), 1.3 (DP-64S), and 1.4 (TFP-80C).



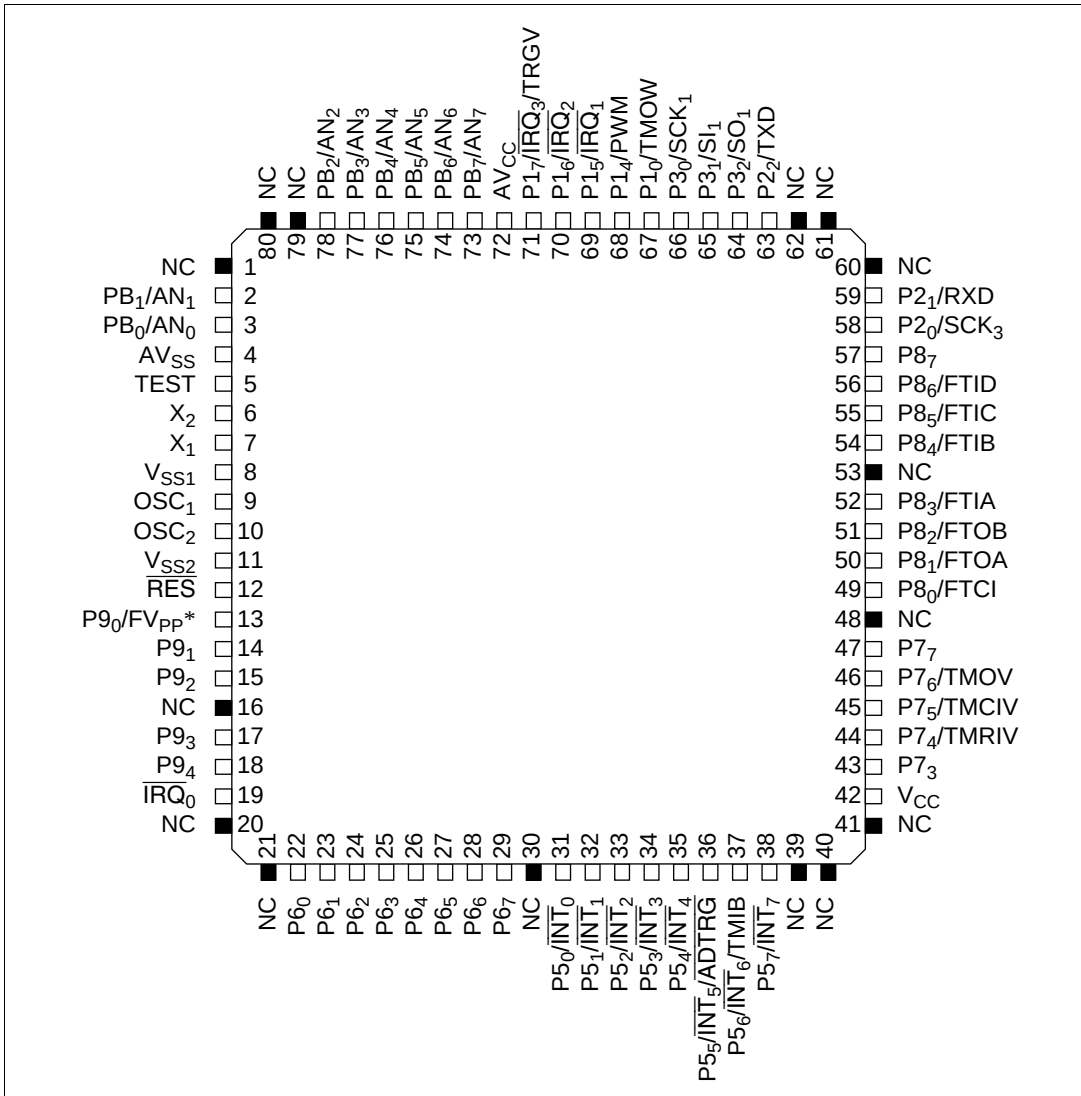
Note: * There is no $P9_0$ function in the flash memory version.

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



Note: * There is no $P9_0$ function in the flash memory version.

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



Note: * There is no $P9_0$ function in the flash memory version.

Figure 1.4 Pin Arrangement (TFP-80C: Top View)

1.3.2 Pin Functions

Table 1.2 outlines the pin functions of the H8/3644 Series.

Table 1.2 Pin Functions

		Pin No.				
Type	**Symbol**	**FP-64A**	**DP-64S**	**TFP-80C**	**I/O**	**Name and Functions**
Power source pins	V_{CC}	33	41	42	Input	**Power supply:** All V_{CC} pins should be connected to the user system V_{CC}.
	V_{SS}	7	15	8, 11	Input	**Ground:** All V_{SS} pins should be connected to the user system GND.
	AV_{CC}	58	2	72	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 user system V_{CC}.
	AV_{SS}	3	11	4	Input	**Analog ground:** This is the A/D converter ground pin. It should be connected to the user system GND.
Clock pins	OSC_1	8	16	9	Input	**System clock:** 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	9	17	10	Output	
	X_1	6	14	7	Input	**Subclock:** These pins connect to a 32.768-kHz crystal oscillator. See section 4, Clock Pulse Generators, for a typical connection diagram.
	X_2	5	13	6	Output	
System control	$\overline{RES}$	10	18	12	Input	**Reset:** When this pin is driven low, the chip is reset
	TEST	4	12	5	Input	**Test:** This is a test pin, not for use in application systems. It should be connected to V_{SS}.

Table 1.2 Pin Functions (cont)

Type	Symbol	Pin No. FP-64A	DP-64S	TFP-80C	I/O	Name and Functions
Interrupt pins	$\overline{IRQ}_0$ $\overline{IRQ}_1$ $\overline{IRQ}_2$ $\overline{IRQ}_3$	16 55 56 57	24 63 64 1	19 69 70 71	Input	**IRQ interrupt request 0 to 3:** These are input pins for edge-sensitive external interrupts, with a selection of rising or falling edge
	$\overline{INT}_7$ to $\overline{INT}_0$	32 to 25	40 to 33	38 to 31	Input	**INT interrupt request 0 to 7:** These are input pins for edge-sensitive external interrupts, with a selection of rising or falling edge
Timer pins	TMOW	53	61	67	Output	**Clock output:** This is an output pin for waveforms generated by the timer A output circuit
	TMIB	31	39	37	Input	**Timer B1 event counter input:** This is an event input pin for input to the timer B1 counter
	TMOV	37	45	46	Output	**Timer V output:** This is an output pin for waveforms generated by the timer V output compare function
	TMCIV	36	44	45	Input	**Timer V event input:** This is an event input pin for input to the timer V counter
	TMRIV	35	43	44	Input	**Timer V counter reset:** This is a counter reset input pin for timer V
	TRGV	57	1	71	Input	**Timer V counter trigger input:** This is a trigger input pin for the timer V counter and realtime output port
	FTCI	39	47	49	Input	**Timer X clock input:** This is an external clock input pin for input to the timer X counter
	FTOA	40	48	50	Output	**Timer X output compare A output:** This is an output pin for timer X output compare A
	FTOB	41	49	51	Output	**Timer X output compare B output:** This is an output pin for timer X output compare B

Table 1.2 Pin Functions (cont)

Type	Symbol	Pin No. FP-64A	DP-64S	TFP-80C	I/O	Name and Functions
Timer pins	FTIA	42	50	52	Input	**Timer X input capture A input:** This is an input pin for timer X input capture A
	FTIB	43	51	54	Input	**Timer X input capture B input:** This is an input pin for timer X input capture B
	FTIC	44	52	55	Input	**Timer X input capture C input:** This is an input pin for timer X input capture C
	FTID	45	53	56	Input	**Timer X input capture D input:** This is an input pin for timer X input capture D
14-bit PWM pin	PWM	54	62	68	Output	**14-bit PWM output:** This is an output pin for waveforms generated by the 14-bit PWM
I/O ports	PB_7 to PB_0	59 to 64, 1 to 2	3 to 10	73 to 78 2, 3	Input	**Port B:** This is an 8-bit input port
	$P1_7$ to $P1_4$, $P1_0$	57 to 53	1, 64 to 61	71 to 67	I/O	**Port 1:** This is a 5-bit I/O port. Input or output can be designated for each bit by means of port control register 1 (PCR1)
	$P2_2$ to $P2_0$	49 to 47	57 to 55	63, 59 58	I/O	**Port 2:** This is a 3-bit I/O port. Input or output can be designated for each bit by means of port control register 2 (PCR2)
	$P3_2$ to $P3_0$	50 to 52	58 to 60	64 to 66	I/O	**Port 3:** This is a 3-bit I/O port. Input or output can be designated for each bit by means of port control register 3 (PCR3)
	$P5_7$ to $P5_0$	32 to 25	40 to 33	38 to 31	I/O	**Port 5:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 5 (PCR5)
	$P6_7$ to $P6_0$	24 to 17	32 to 25	29 to 22	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)

Table 1.2 Pin Functions (cont)

Type	Symbol	Pin No. FP-64A	DP-64S	TFP-80C	I/O	Name and Functions
I/O ports	$P7_7$ to $P7_3$	38 to 34	46 to 42	47 to 43	I/O	**Port 7:** This is a 5-bit I/O port. Input or output can be designated for each bit by means of port control register 7 (PCR7)
	$P8_7$ to $P8_0$	46 to 39	54 to 47	57 to 54, 52 to 49	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)
	$P9_4$ to $P9_0$	15 to 11	23 to 19	18, 17 15 to 13	I/O	**Port 9:** This is a 5-bit I/O port. Input or output can be designated for each bit by means of port control register 9 (PCR9) Note: There is no $P9_0$ function in the flash memory version since $P9_0$ is used as the FV_{PP} pin.
Serial communication interface (SCI)	SI_1	51	59	65	Input	**SCI1 receive data input:** This is the SCI1 data input pin
	SO_1	50	58	64	Output	**SCI1 transmit data output:** This is the SCI1 data output pin
	SCK_1	52	60	66	I/O	**SCI1 clock I/O:** This is the SCI1 clock I/O pin
	RXD	48	56	59	Input	**SCI3 receive data input:** This is the SCI3 data input pin
	TXD	49	57	63	Output	**SCI3 transmit data output:** This is the SCI3 data output pin
	SCK_3	47	55	58	I/O	**SCI3 clock I/O:** This is the SCI3 clock I/O pin
A/D converter	AN_7 to AN_0	59 to 64, 1 to 2	3 to 10	73 to 78 2, 3	Input	**Analog input channels 11 to 0:** These are analog data input channels to the A/D converter
	$\overline{ADTRG}$	30	38	36	Input	**A/D converter trigger input:** This is the external trigger input pin to the A/D converter

Table 1.2 Pin Functions (cont)

Type	Symbol	Pin No. FP-64A	Pin No. DP-64S	Pin No. TFP-80C	I/O	Name and Functions
Flash memory	FV_{PP}	11	19	13	Input	**On-board-programmable flash memory power supply:** Connected to the flash memory programming power supply (+12 V). When the flash memory is not being programmed, connect to the user system V_{CC}. In versions other than the on-chip flash memory version, this pin is $P9_0$
Other	NC	—	—	1, 16, 20, 21, 30, 39, 40, 41, 48, 53, 60 to 62, 79, 80	—	**Non-connected pins:** These pins must be left unconnected

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:	0.4 µs (operating at ø = 5 MHz)
	0.25 µs (operating at ø = 8 MHz)*
8 × 8-bit multiply:	2.8 µs (operating at ø = 5 MHz)
	1.75 µs (operating at ø = 8 MHz)*
16 ÷ 8-bit divide:	2.8 µs (operating at ø = 5 MHz)
	1.75 µs (operating at ø = 8 MHz)*

Note: * F-ZTAT version only.

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

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

2.1.2 Address Space

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

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

2.1.3 Register Configuration

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

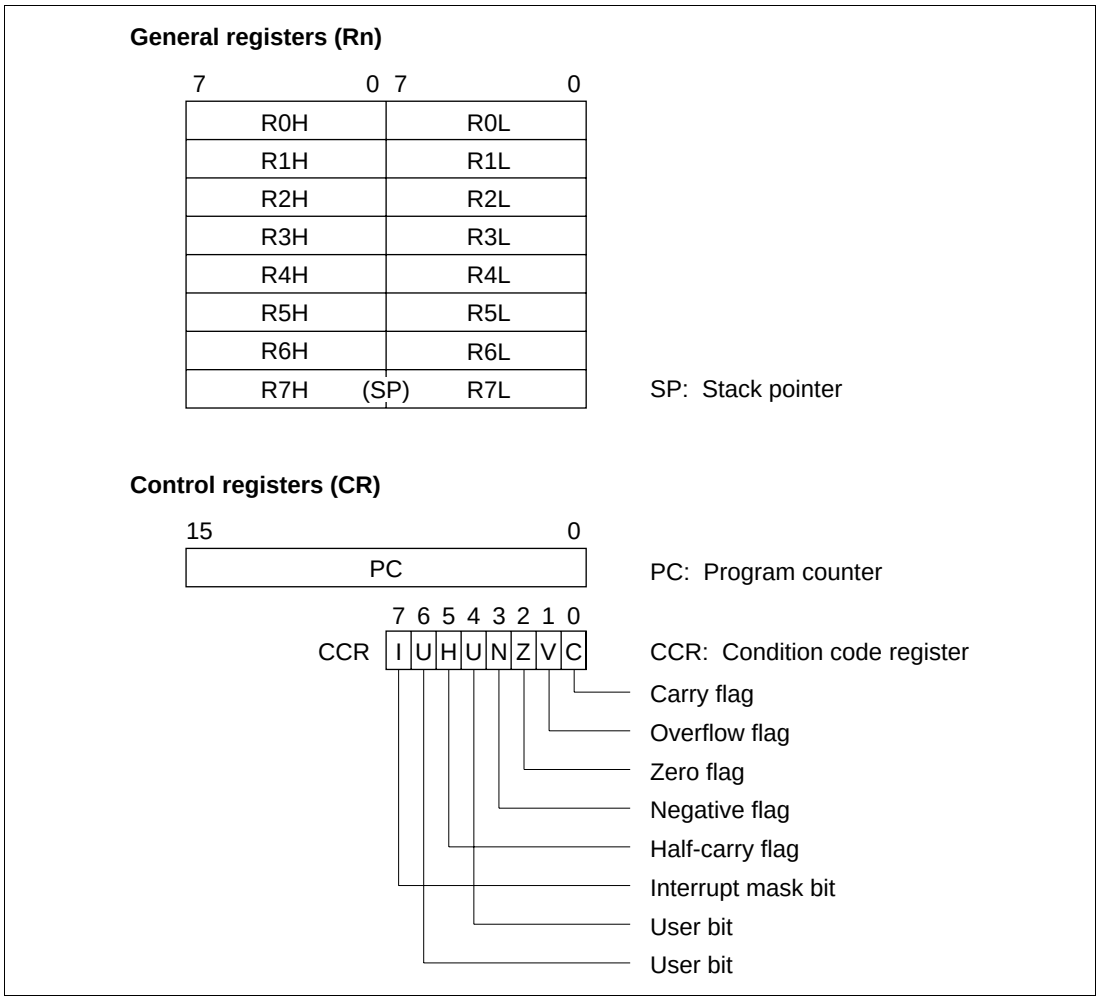


Figure 2.1 CPU Registers

2.2 Register Descriptions

2.2.1 General Registers

All the general registers 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.

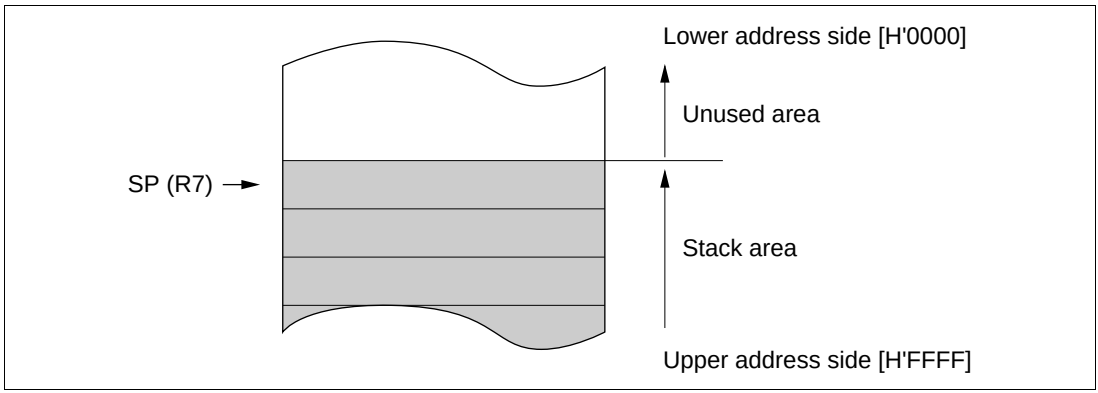


Figure 2.2 Stack Pointer

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/rotate carry

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

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

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

2.2.3 Initial Register Values

In reset exception handling, the program counter (PC) is initialized by a vector address (H'0000) load, 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.

The H8/300L CPU can process 1-bit, 4-bit BCD, 8-bit (byte), and 16-bit (word) data. 1-bit data is handled by bit manipulation instructions, and is accessed by being specified as bit n (n = 0, 1, 2, ... 7) in the operand data (byte).

Byte data is handled by all arithmetic and logic instructions except ADDS and SUBS. Word data is handled by the MOV.W, ADD.W, SUB.W, CMP.W, ADDS, SUBS, MULXU (b bits $\times$ 8 bits), and DIVXU (16 bits $\div$ 8 bits) instructions.

With the DAA and DAS decimal adjustment instructions, byte data is handled as two 4-bit BCD data units.

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.

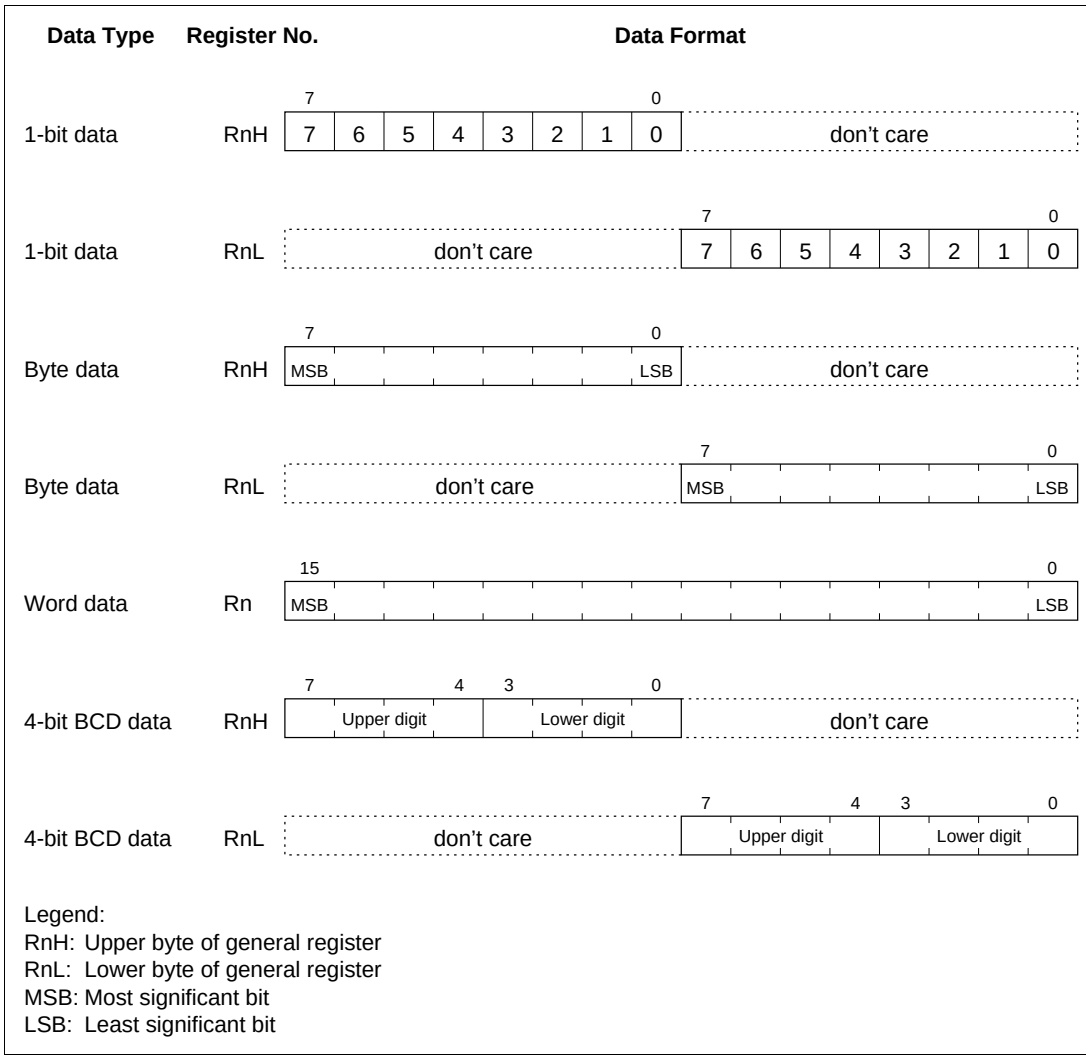


Figure 2.3 General Register Data Formats

2.3.2 Memory Data Formats

Figure 2.4 indicates the data formats in memory. For access by the H8/300L CPU, word data stored in memory must always begin at an even address. When word data beginning at an odd address is accessed, the least significant bit is regarded as 0, and the word data beginning at the preceding address is accessed. The same applies to instruction codes.

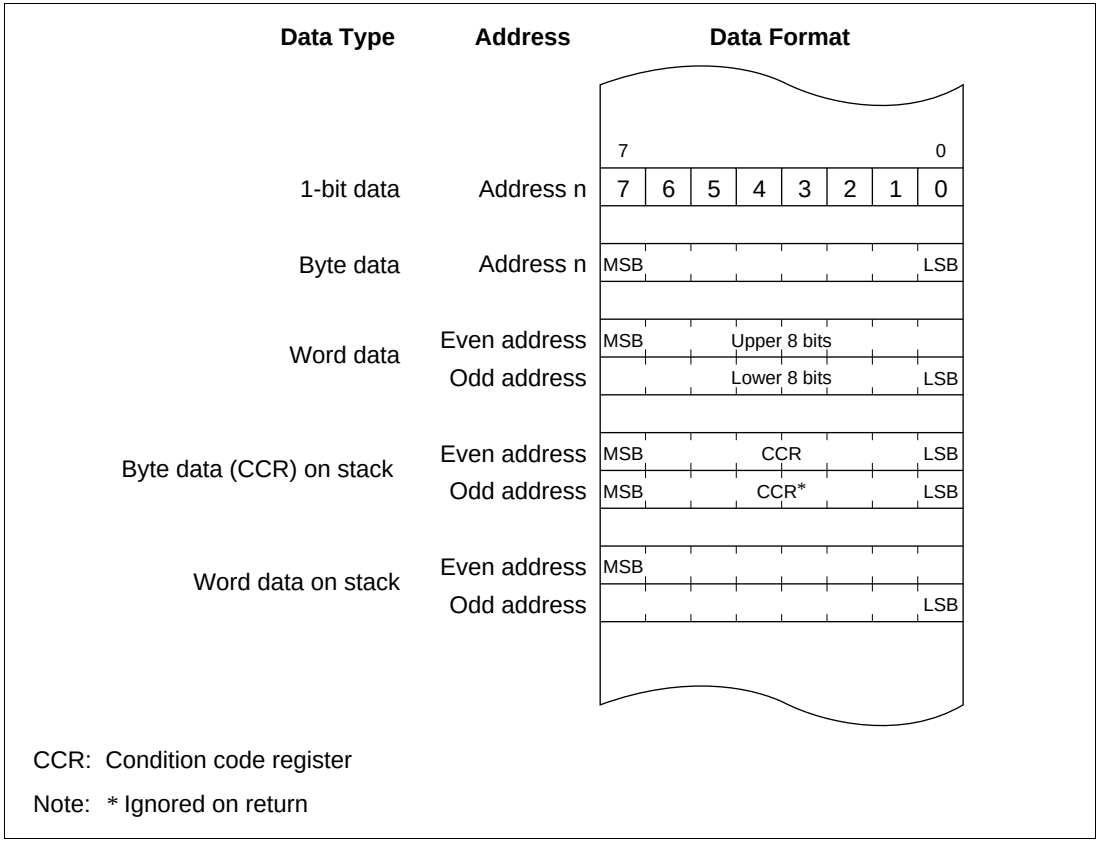


Figure 2.4 Memory Data Formats

When the stack is accessed using R7 as an address register, word access should always be performed. The CCR is stored as word data with the same value in the upper 8 bits and the lower 8 bits. On return, the lower 8 bits are 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, and the result of the addition is stored in the register. 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 second byte (#xx:8) or the third and fourth bytes (#xx:16) of the instruction code are used directly as the operand. Only MOV.W instructions can be used with #xx:16.

   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, and the PC contents to be added are the start address of the next instruction, so that the possible branching range is –126 to +128 bytes (–63 to +64 words) from the branch instruction. 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. This specifies an operand in memory, and a branch is performed with the contents of this operand as the branch 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 use register direct (1), register indirect (2), or 8-bit absolute addressing (5) to specify a byte operand, and 3-bit immediate addressing (6) to specify a bit position in that byte. The BSET, BCLR, BNOT, and BTST instructions can also use register direct addressing (1) to specify the bit position.

Table 2.2 Effective Address Calculation

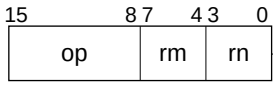
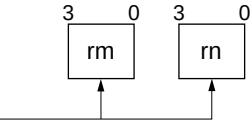
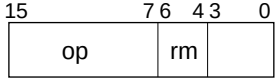
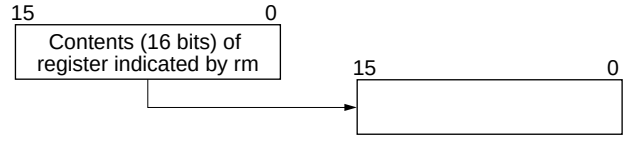
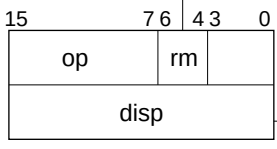
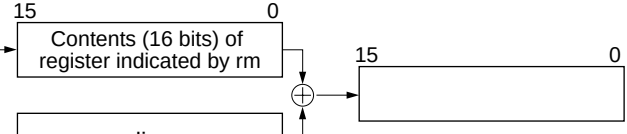
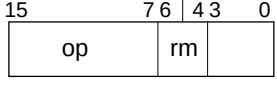
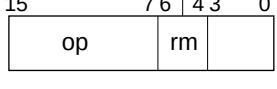
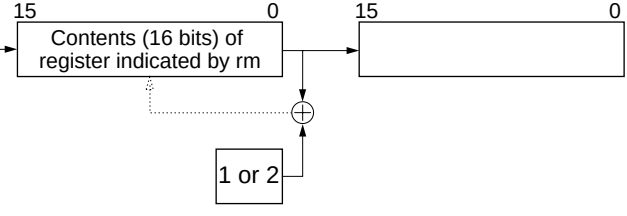
No.	Addressing Mode and Instruction Format	Effective Address Calculation Method	Effective Address (EA)
1	Register indirect, Rn		 Operand is contents of registers indicated by rm/rn
2	Register indirect, @Rn		
3	Register indirect with displacement, @(d:16, Rn)		
4	Register indirect with post-increment,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

Table 2.2 Effective Address Calculation (cont)

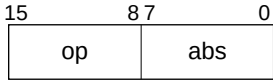
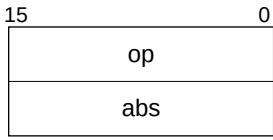
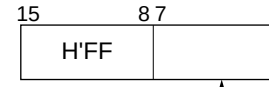
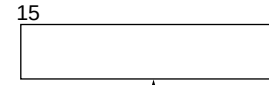
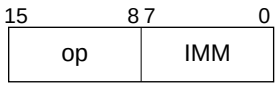
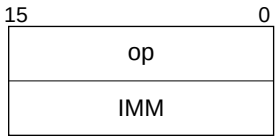
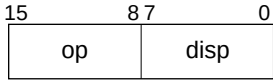
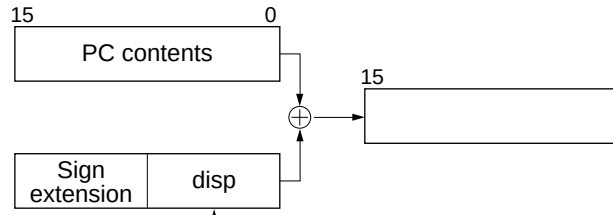
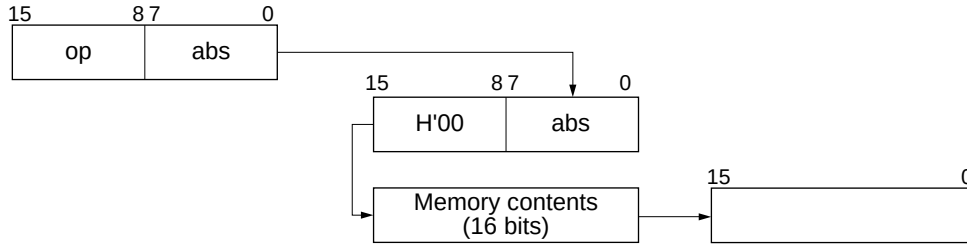
No.	Addressing Mode and Instruction Format	Effective Address Calculation Method	Effective Address (EA)
5	Absolute address @aa:8  @aa:16 	 	
6	Immediate #xx:8  #xx:16 		Operand is 1- or 2-byte immediate data
7	Program-counter relative @(d:8, PC) 		

Table 2.2 Effective Address Calculation (cont)

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

Legend:

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

2.5 Instruction Set

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

Table 2.3 Instruction Set

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

Notes: 1. PUSH Rn is equivalent to MOV.W Rn, @–SP.
POP Rn is equivalent to MOV.W @SP+, Rn.
2. Bcc is a conditional branch instruction. The same applies to machine language.

Tables 2.4 to 2.11 show the function of each instruction. 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 a word-size specification.
POP	W	@SP+ → Rn Pops a general register from the stack. Equivalent to MOV.W @SP+, Rn.
PUSH	W	Rn → @–SP Pushes 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.

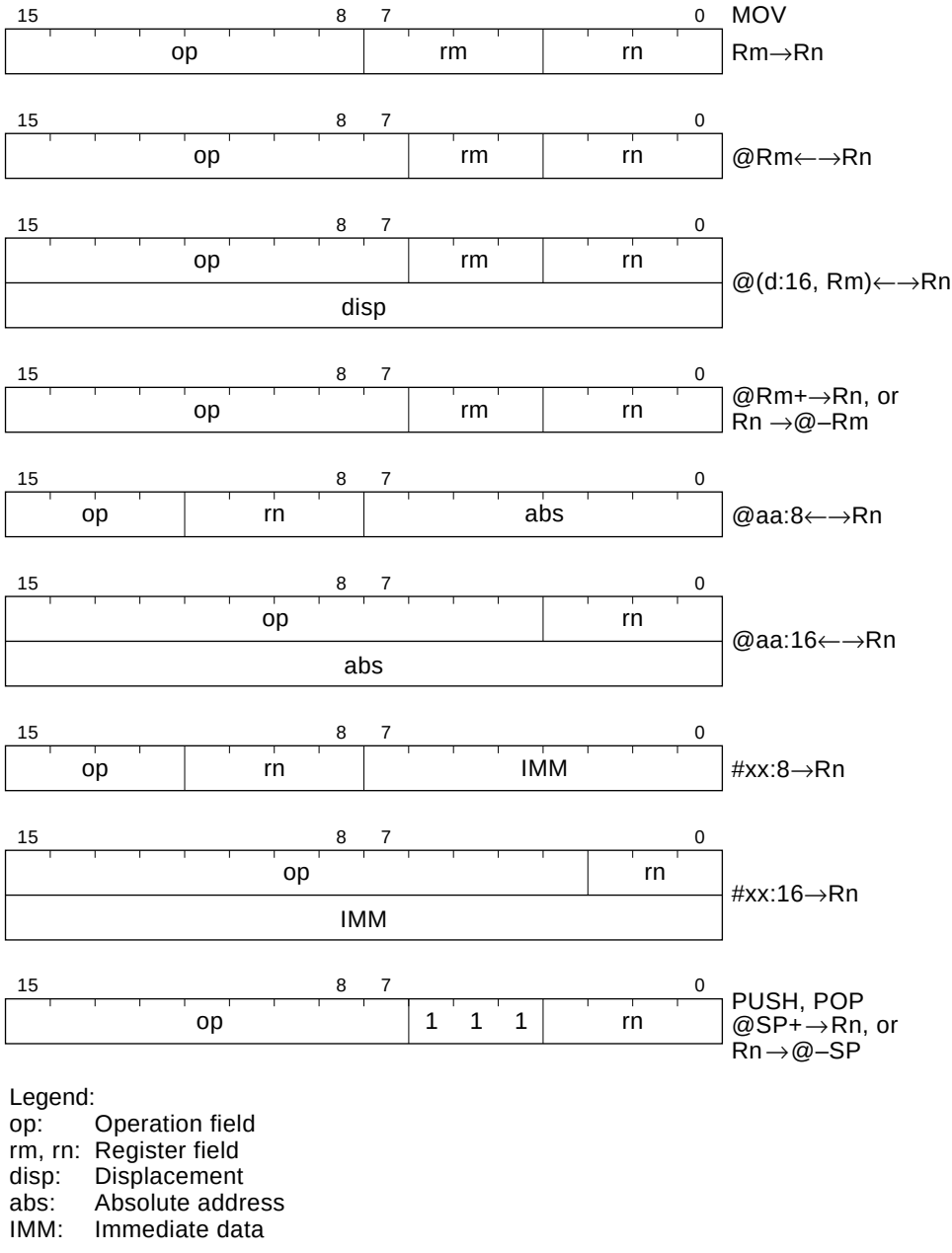


Figure 2.5 Data Transfer Instruction Codes

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 on data in two general registers, or addition or subtraction with carry on immediate data and data in a general register.
INC DEC	B	Rd ± 1 → Rd Increments or decrements a general register
ADDS SUBS	W	Rd ± 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 packed BCD) an addition or subtraction result in a general register by referring to the CCR
MULXU	B	Rd × Rs → Rd Performs 8-bit × 8-bit unsigned multiplication on data in two general registers, providing a 16-bit result
DIVXU	B	Rd ÷ Rs → Rd Performs 16-bit ÷ 8-bit unsigned division on data in two general registers, providing an 8-bit quotient and 8-bit remainder
CMP	B/W	Rd – Rs, Rd – #IMM Compares data in a general register with data in another general register or with immediate data, 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 ∧ Rs → Rd, Rd ∧ #IMM → Rd Performs a logical AND operation on a general register and another general register or immediate data
OR	B	Rd ∨ Rs → Rd, Rd ∨ #IMM → Rd Performs a logical OR operation on a general register and another general register or immediate data
XOR	B	Rd ⊕ Rs → Rd, Rd ⊕ #IMM → Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data
NOT	B	~ Rd → Rd Obtains the one's complement (logical complement) of general register contents

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	B	Rd shift → Rd
SHAR		Performs an arithmetic shift operation on general register contents
SHLL	B	Rd shift → Rd
SHLR		Performs a logical shift operation on general register contents
ROTL	B	Rd rotate → Rd
ROTR		Rotates general register contents
ROTXL	B	Rd rotate → Rd
ROTXR		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.

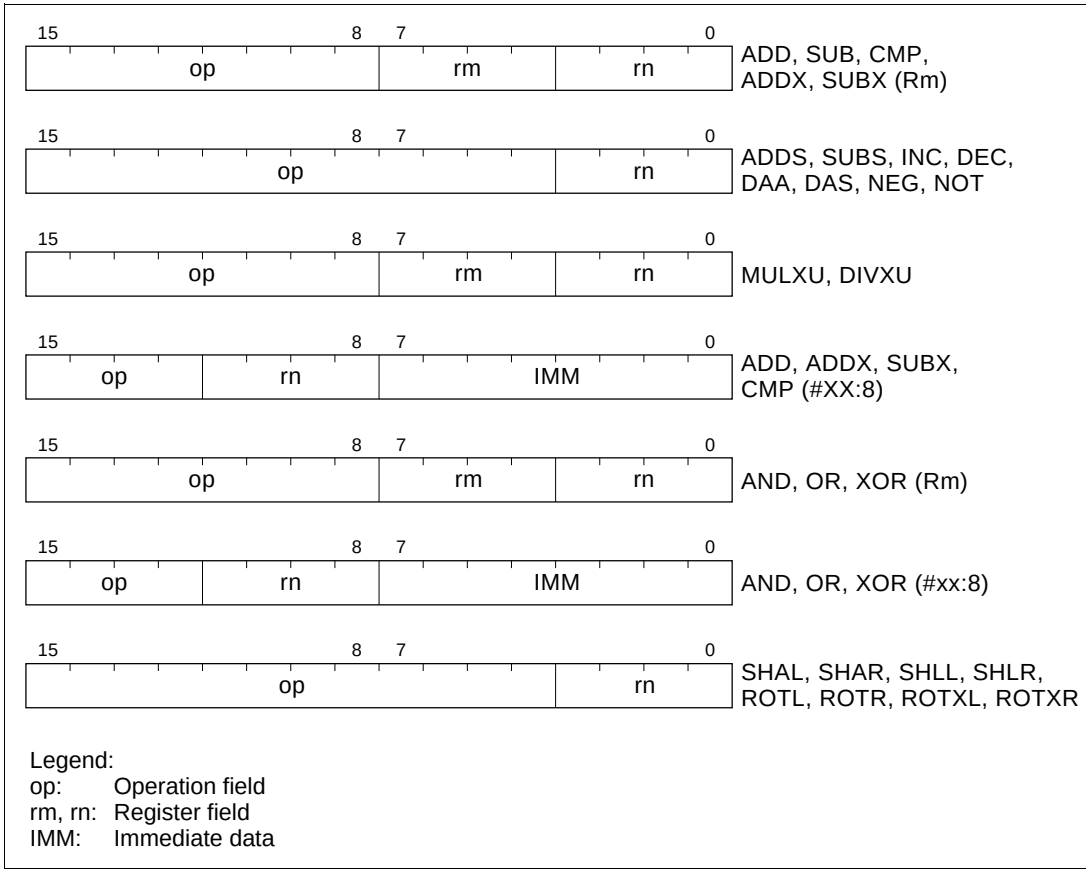


Figure 2.6 Arithmetic, Logic, and Shift Instruction Codes

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

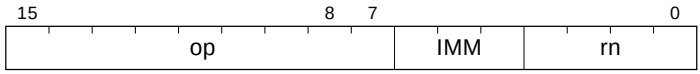
Table 2.8 Bit-Manipulation Instructions (cont)

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

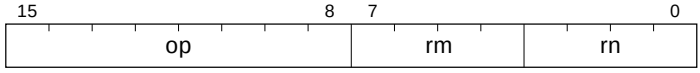
Notes: * Size: Operand size
B: Byte

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

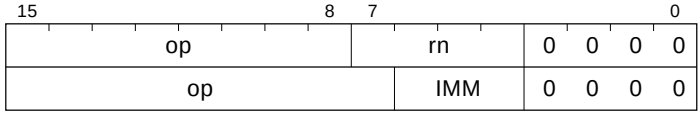
BSET, BCLR, BNOT, BTST



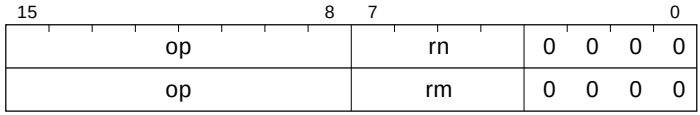
Operand: register direct (Rn)
Bit No.: immediate (#xx:3)



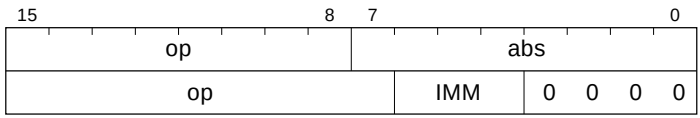
Operand: register direct (Rn)
Bit No.: register direct (Rm)



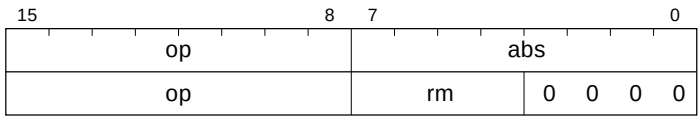
Operand: register indirect (@Rn)
Bit No.: immediate (#xx:3)



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

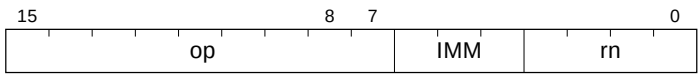


Operand: absolute (@aa:8)
Bit No.: immediate (#xx:3)

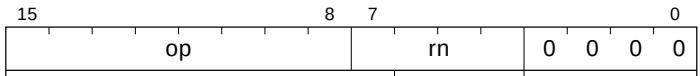


Operand: absolute (@aa:8)
Bit No.: register direct (Rm)

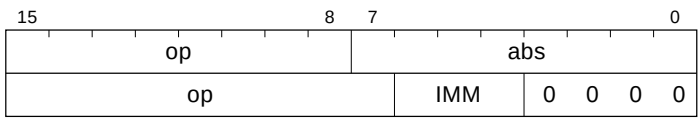
BAND, BOR, BXOR, BLD, BST



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



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

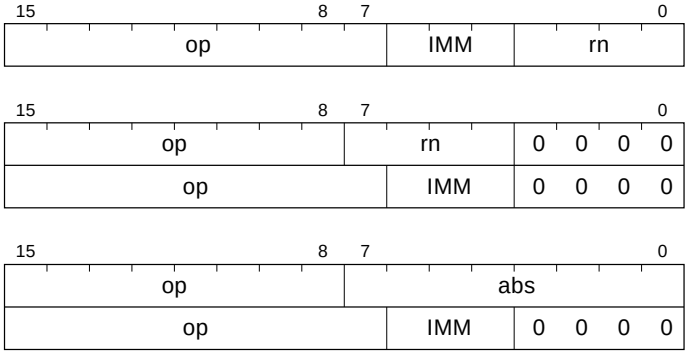


Operand: absolute (@aa:8)
Bit No.: immediate (#xx:3)

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

Figure 2.7 Bit Manipulation Instruction Codes

BIAND, BIOR, BIXOR, BILD, BIST



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

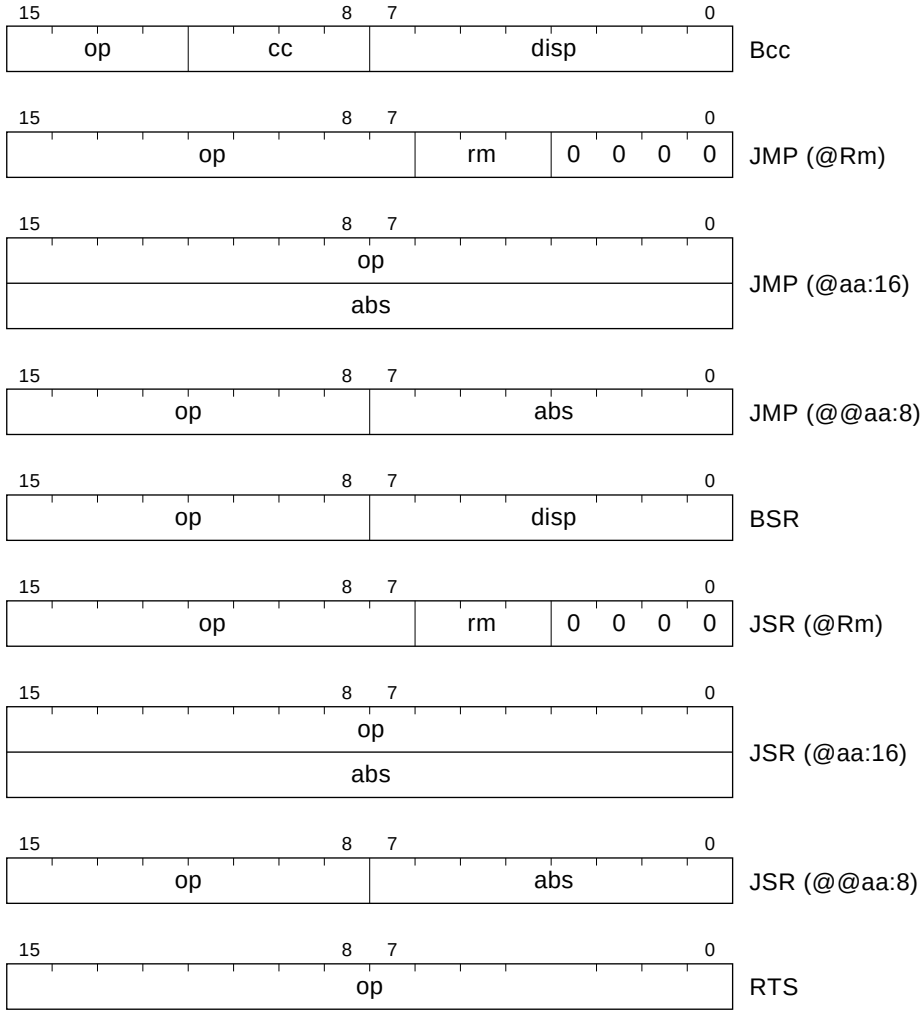
Figure 2.7 Bit Manipulation Instruction Codes (cont)

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



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

Figure 2.8 Branching Instruction Codes

2.5.7 System Control Instructions

Table 2.10 describes the system control instructions. Figure 2.9 shows their object code formats.

Table 2.10 System Control Instructions

Instruction	Size*	Function
RTE	—	Returns from an exception-handling routine
SLEEP	—	Causes a transition from active mode to a power-down mode. See section 5, Power-Down Modes, for details.
LDC	B	Rs → CCR, #IMM → CCR Moves immediate data or general register contents to the condition code register
STC	B	CCR → Rd Copies the condition code register to a specified general register
ANDC	B	CCR ∧ #IMM → CCR Logically ANDs the condition code register with immediate data
ORC	B	CCR ∨ #IMM → CCR Logically ORs the condition code register with immediate data
XORC	B	CCR ⊕ #IMM → CCR Logically exclusive-ORs the condition code register with immediate data
NOP	—	PC + 2 → PC Only increments the program counter

Notes: * Size: Operand size
B: Byte

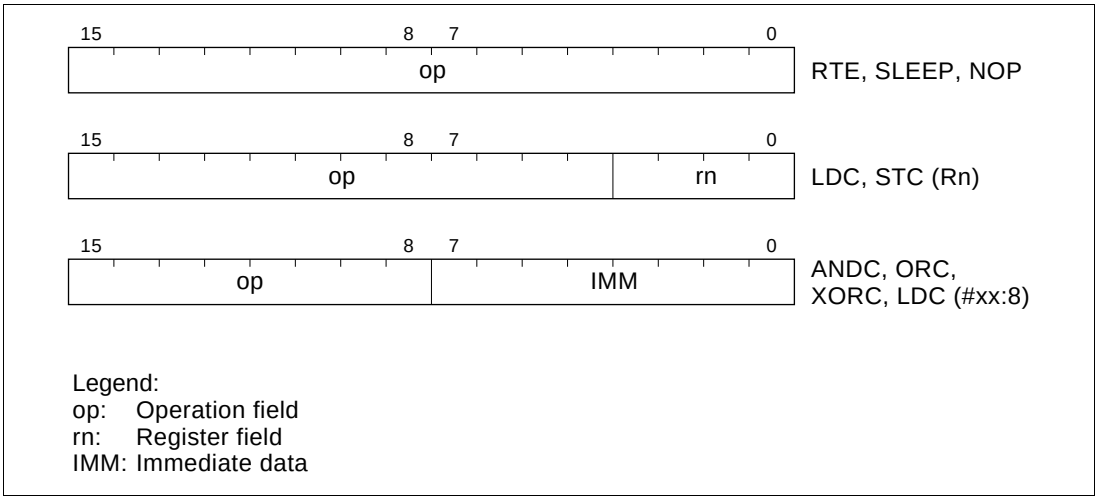


Figure 2.9 System Control Instruction Codes

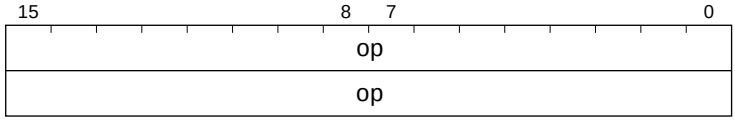
2.5.8 Block Data Transfer Instruction

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

Table 2.11 Block Data Transfer Instruction

Instruction	Size	Function
EEPMOV	—	If R4L ≠ 0 then repeat @R5+ → @R6+ R4L – 1 → R4L until R4L = 0 else next; Block transfer instruction. Transfers the number of 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.



Legend:
op: Operation field

Figure 2.10 Block Data Transfer Instruction Code

2.6 Basic Operational Timing

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

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

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.

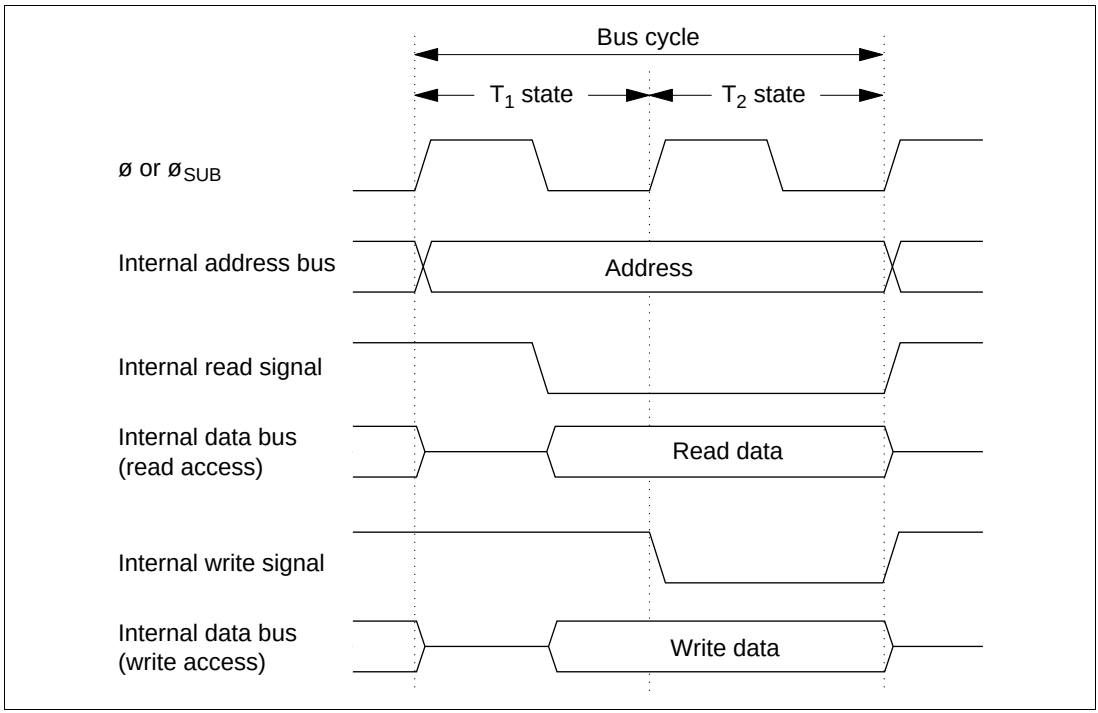


Figure 2.11 On-Chip Memory Access Cycle

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.

Two-State Access to On-Chip Peripheral Modules: Figure 2.12 shows the operation timing in the case of two-state access to an on-chip peripheral module.

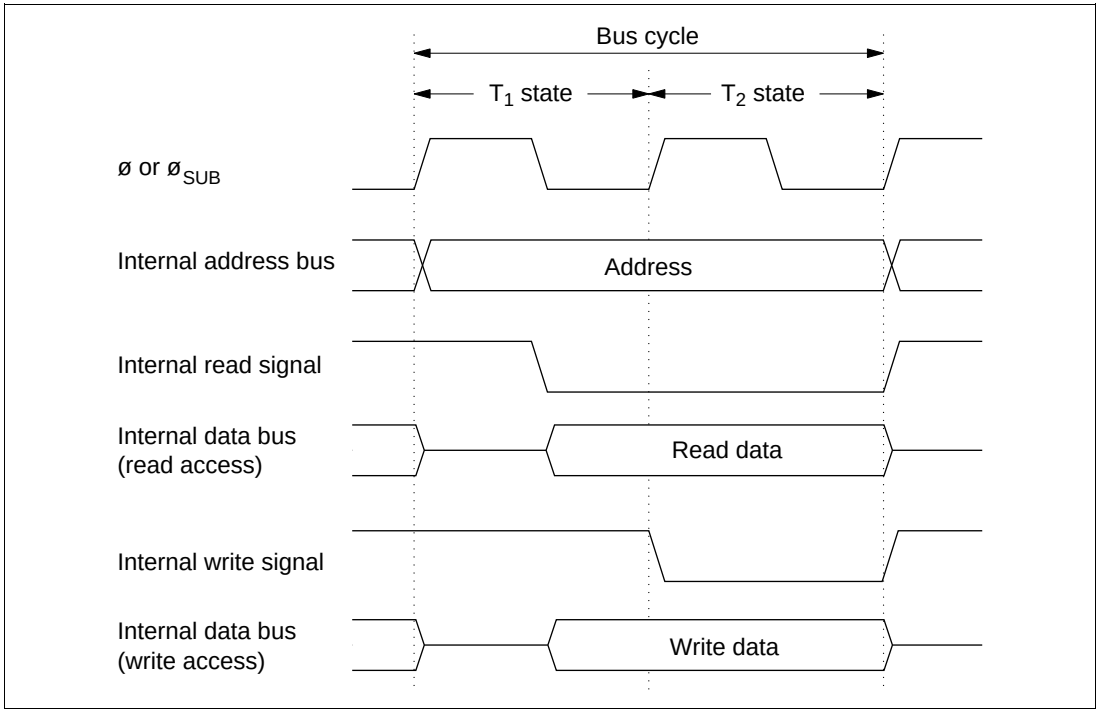


Figure 2.12 On-Chip Peripheral Module Access Cycle (2-State Access)

Three-State Access to On-Chip Peripheral Modules: Figure 2.13 shows the operation timing in the case of three-state access to an on-chip peripheral module.

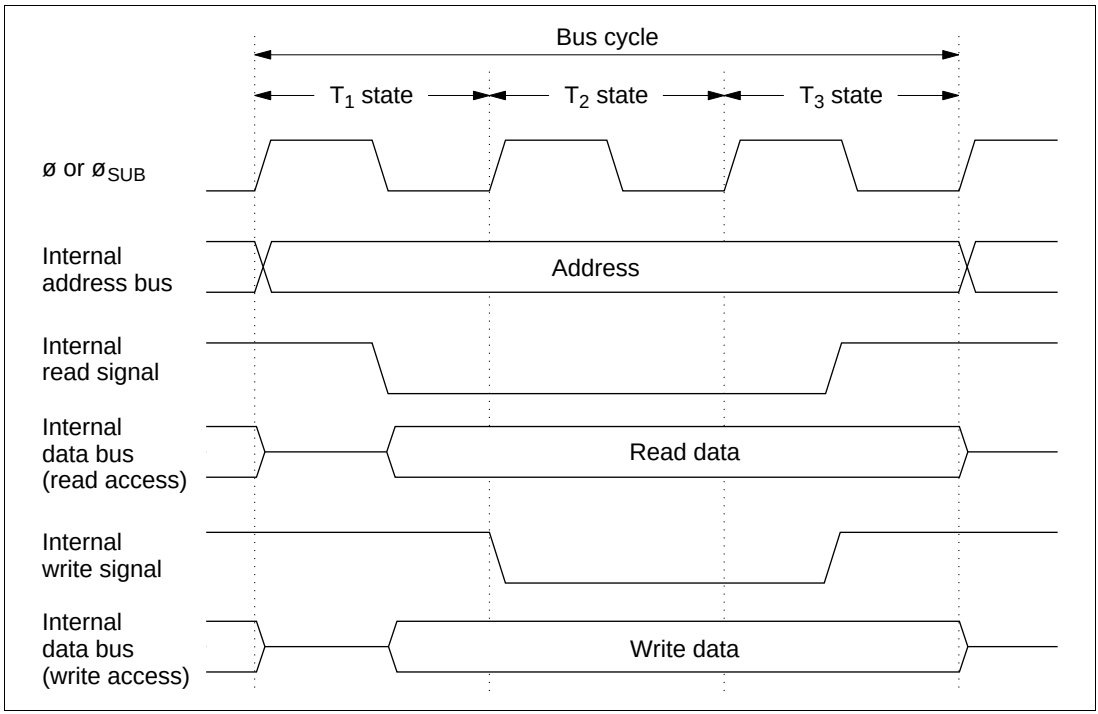
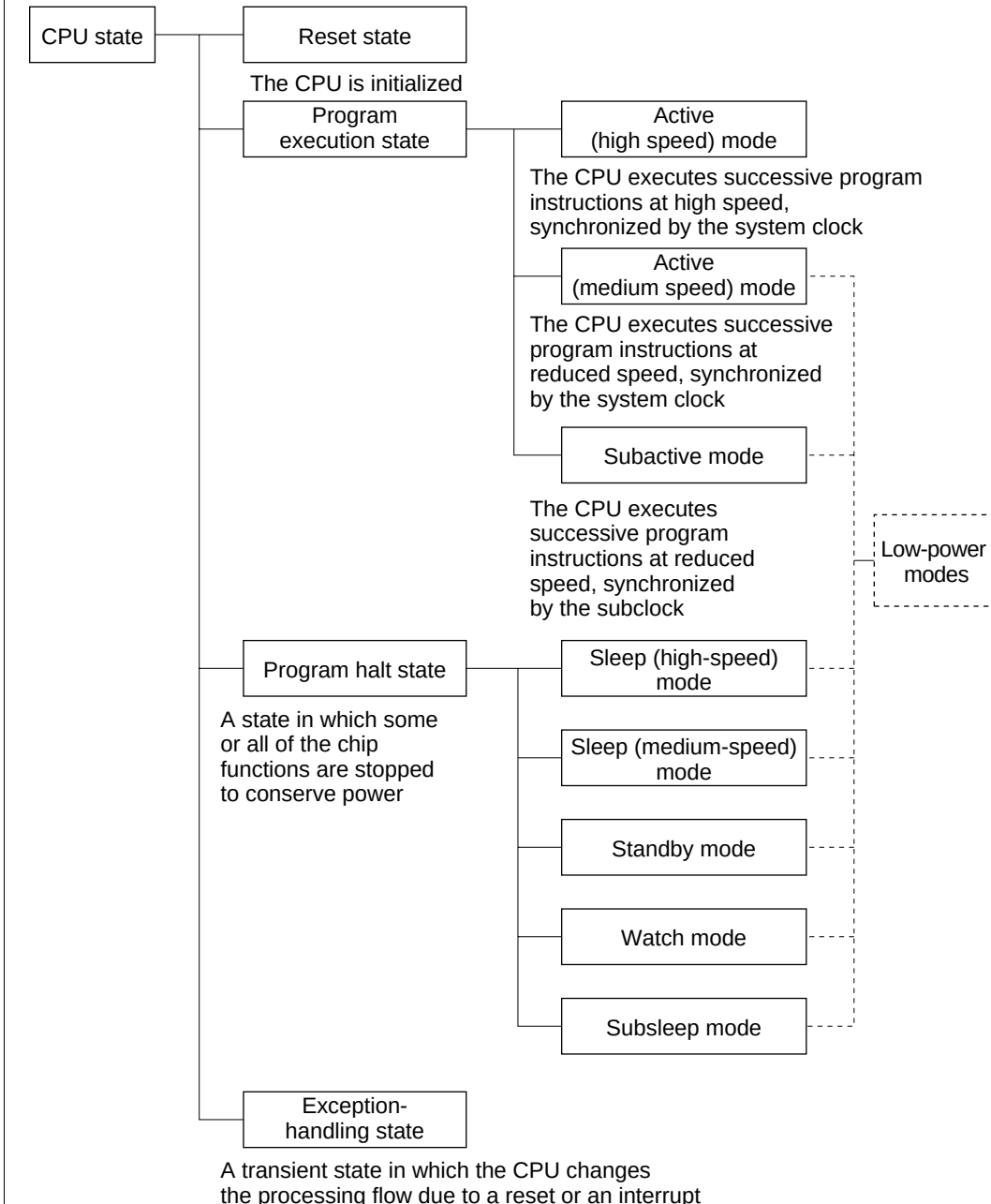


Figure 2.13 On-Chip Peripheral Module Access Cycle (3-State Access)

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

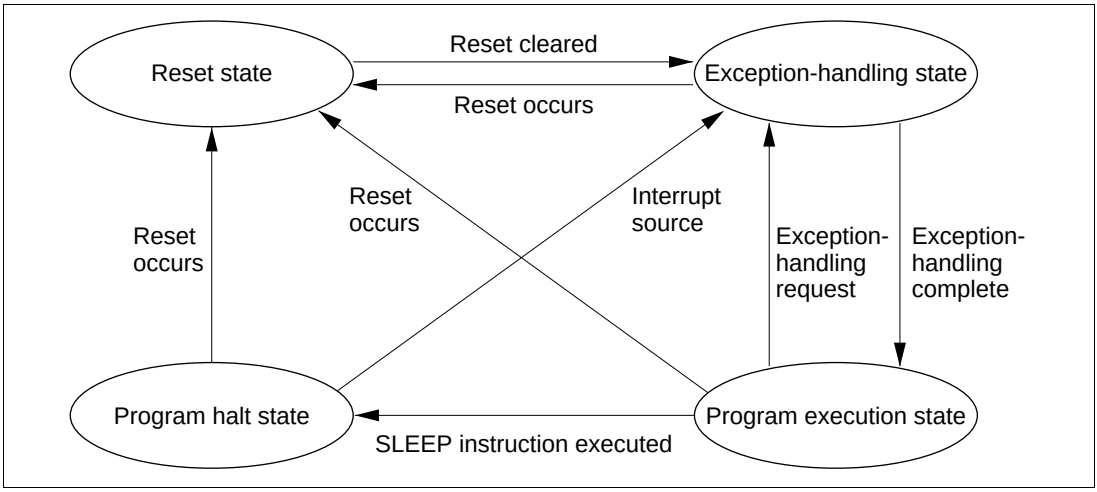


Figure 2.15 State Transitions

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

Figure 2.16 shows a memory map of the H8/3644 Series.

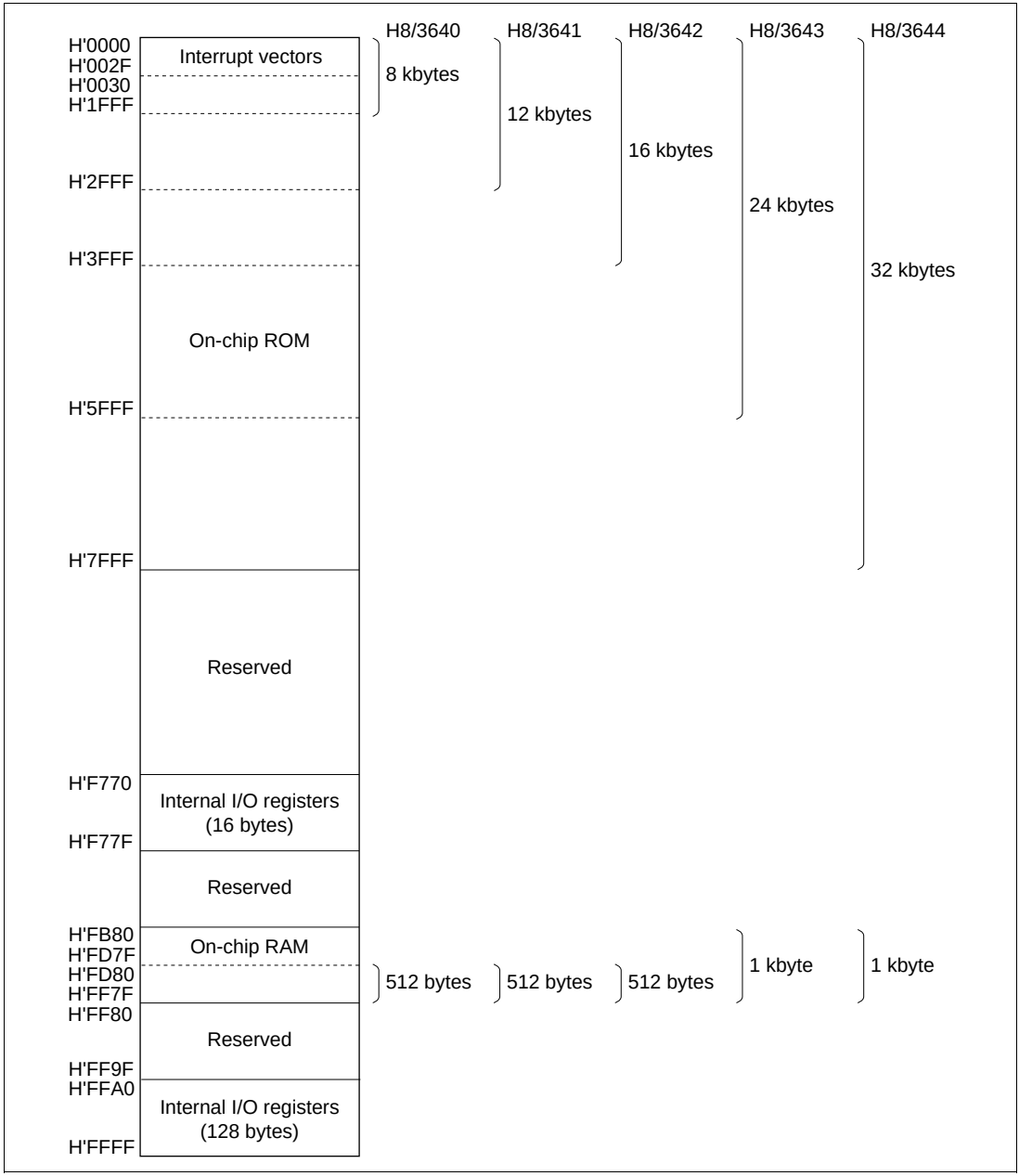


Figure 2.16 H8/3644 Series Memory Map

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.

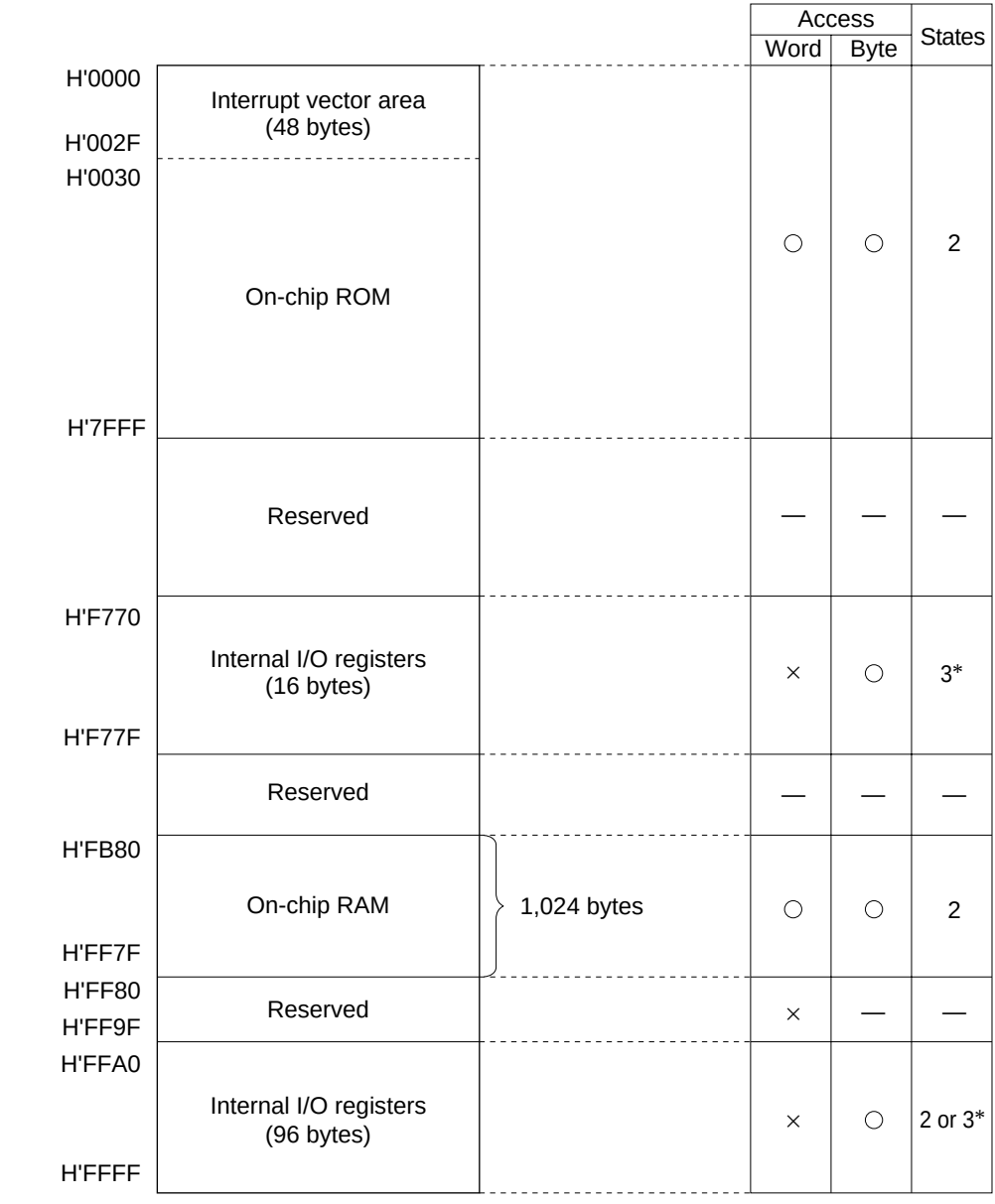
   Word access from CPU to I/O register area:

   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 H8/3644 is shown as an example.

* Internal I/O registers in areas assigned to timer X (H'F770 to H'F77F), SCI3 (H'FFA8 to H'FFAD), and timer V (H'FFB8 to H'FFBD) are accessed in three states.

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

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.

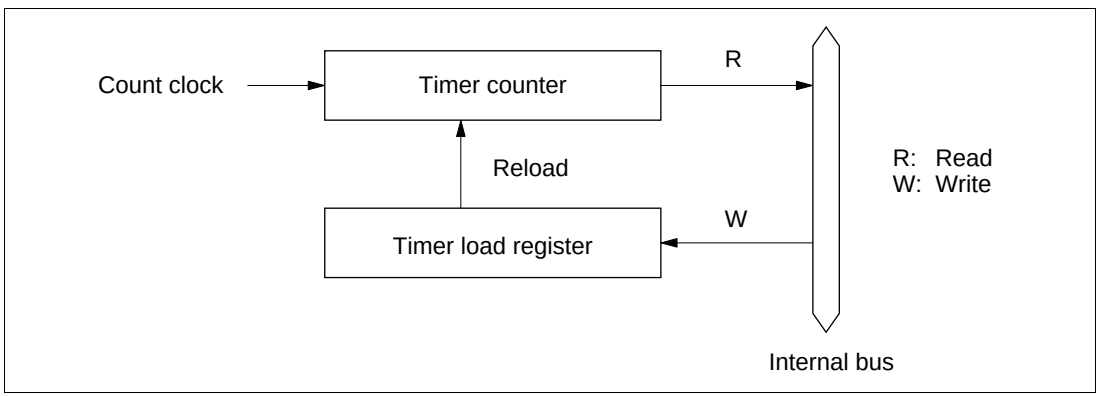


Figure 2.18 Timer Configuration Example

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 $P3_0$, are output pins and output low-level signals. In this example, the BSET instruction is used to change pin $P3_0$ to high-level output.

[A: Prior to executing BSET]

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	0

[B: BSET instruction executed]

```
BSET   #0  ,  @PDR3
```

The BSET instruction is executed designating port 3.

[C: After executing BSET]

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR3	0	0	1	1	1	1	1	1
PDR3	0	1	0	0	0	0	0	1

[D: Explanation of how BSET operates]

When the BSET instruction is executed, first the CPU reads port 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 $P3_0$ are output pins, so the CPU reads the value in PDR3. In this example PDR3 has a value of H'80, but the value read by the CPU is H'40.

Next, the CPU sets bit 0 of the read data to 1, changing the PDR3 data to H'41. Finally, the CPU writes this value (H'41) to PDR3, completing execution of BSET.

As a result of this operation, bit 0 in PDR3 becomes 1, and $P3_0$ 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  #80,  R0L
MOV. B  R0L,  @RAM0
MOV. B  R0L,  @PDR3
```

The PDR3 value (H'80) is written to a work area in memory (RAM0) as well as to PDR3.

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	0
RAM0	1	0	0	0	0	0	0	0

[B: BSET instruction executed]

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

The BSET instruction is executed designating the 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.

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
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

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 $P3_0$, are output pins that output low-level signals. In this example, the BCLR instruction is used to change pin $P3_0$ to an input port. It is assumed that a high-level signal will be input to this input pin.

[A: Prior to executing BCLR]

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	0

[B: BCLR instruction executed]

```
BCLR   #0  ,  @PCR3
```

The BCLR instruction is executed designating PCR3.

[C: After executing BCLR]

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Output	Output	Output	Output	Output	Output	Output	Input
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR3	1	1	1	1	1	1	1	0
PDR3	1	0	0	0	0	0	0	0

[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 0 in the read data to 0, changing the data to H'FE. Finally, this value (H'FE) is written to PCR3 and BCLR instruction execution ends.

As a result of this operation, bit 0 in PCR3 becomes 0, making $P3_0$ 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.

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
PCR3	0	0	1	1	1	1	1	1
PDR3	1	0	0	0	0	0	0	0
RAM0	0	0	1	1	1	1	1	1

[B: BCLR instruction executed]

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

The BCLR instruction is executed designating the 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.

	$P3_7$	**$P3_6$**	**$P3_5$**	**$P3_4$**	**$P3_3$**	**$P3_2$**	**$P3_1$**	**$P3_0$**
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
PCR3	0	0	1	1	1	1	1	0
PDR3	1	0	0	0	0	0	0	0
RAM0	0	0	1	1	1	1	1	0

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
Output compare register AH and output compare register BH (timer X)	OCRAH/OCRBH	H'F774
Output compare register AL and output compare register BL (timer X)	OCRAL/OCRBL	H'F775
Timer counter B1 and timer load register B1 (timer B1)	TCB1/TLB1	H'FFB3
Port data register 1*	PDR1	H'FFD4
Port data register 2*	PDR2	H'FFD5
Port data register 3*	PDR3	H'FFD6
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 9*	PDR9	H'FFDC

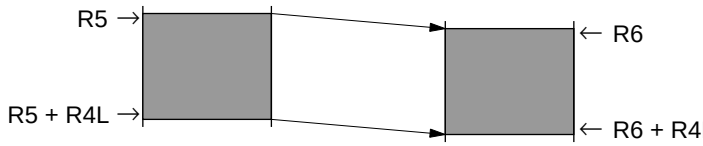
Note: * Port data registers have the same addresses as input pins.

Table 2.13 Registers with Write-Only Bits

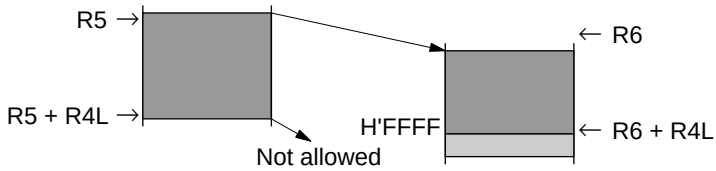
Register Name	Abbreviation	Address
Port control register 1	PCR1	H'FFE4
Port control register 2	PCR2	H'FFE5
Port control register 3	PCR3	H'FFE6
Port control register 5	PCR5	H'FFE8
Port control register 6	PCR6	H'FFE9
Port control register 7	PCR7	H'FFEA
Port control register 8	PCR8	H'FFEB
Port control register 9	PCR9	H'FFEC
PWM control register	PWCR	H'FFD0
PWM data register U	PWDRU	H'FFD1
PWM data register L	PWDRL	H'FFD2

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 H8/3644 Series when a reset or interrupt occurs. Table 3.1 shows the priorities of these two types of exception handling.

Table 3.1 Exception Handling Types and Priorities

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

3.2 Reset

3.2.1 Overview

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

3.2.2 Reset Sequence

Reset by $\overline{\text{RES}}$ Pin: As soon as the 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 begins when the $\overline{\text{RES}}$ pin is held low for a given period, then returned to the high level.

Reset exception handling takes place as follows.

- The CPU internal state and the registers of on-chip peripheral modules are initialized, with the I bit of the condition code register (CCR) set to 1.
- The PC is loaded from the reset exception handling vector address (H'0000 to H'0001), after which the program starts executing from the address indicated in PC.

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

Figure 3.1 shows the reset sequence starting from $\overline{RES}$ input.

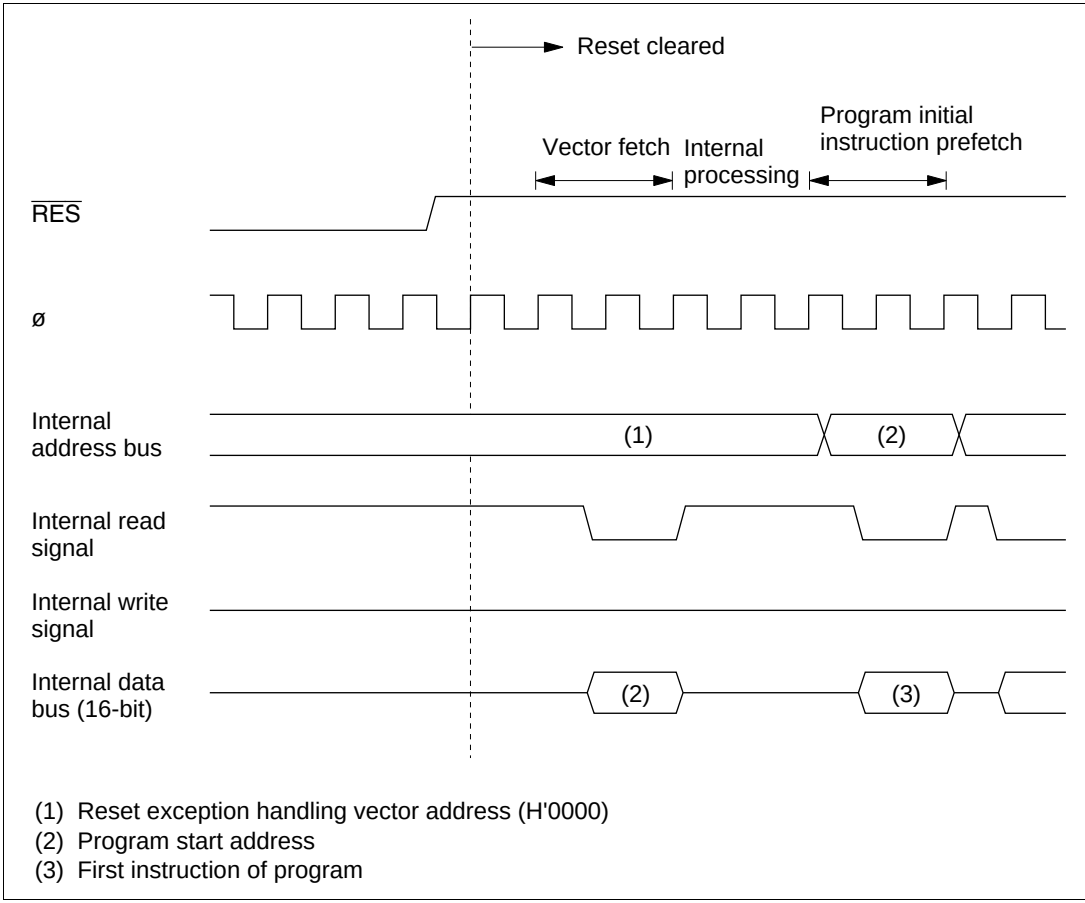


Figure 3.1 Reset Sequence

Reset by Watchdog Timer: The watchdog timer counter (TCW) starts counting up when the WDON bit is set to 1 in the watchdog timer control/status register (TCSRW). If TCW overflows, the WRST bit is set to 1 in TCSRW and the chip enters the reset state. While the WRST bit is set to 1 in TCSRW, when TCW overflows the reset state is cleared and reset exception handling begins. The same reset exception handling is carried out as for input at the $\overline{RES}$ pin. For details on the watchdog timer, see 9.1.1, Watchdog Timer.

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 12 external interrupts (IRQ_3 to IRQ_0, INT_7 to INT_0) and 21 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.
- IRQ_3 to IRQ_0 and INT_7 to INT_0 can be set independently 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	
$\overline{IRQ_2}$	IRQ_2	6	H'000C to H'000D	
$\overline{IRQ_3}$	IRQ_3	7	H'000E to H'000F	
$\overline{INT_0}$	INT_0	8	H'0010 to H'0011	
$\overline{INT_1}$	INT_1			
$\overline{INT_2}$	INT_2			
$\overline{INT_3}$	INT_3			
$\overline{INT_4}$	INT_4			
$\overline{INT_5}$	INT_5			
$\overline{INT_6}$	INT_6			
$\overline{INT_7}$	INT_7			
Timer A	Timer A overflow	10	H'0014 to H'0015	
Timer B1	Timer B1 overflow	11	H'0016 to H'0017	
Timer X	Timer X input capture A	16	H'0020 to H'0021	
	Timer X input capture B			
	Timer X input capture C			
	Timer X input capture D			
	Timer X compare match A			
	Timer X compare match B			
	Timer X overflow			
Timer V	Timer V compare match A	17	H'0022 to H'0023	
	Timer V compare match B			
	Timer V overflow			
SCI1	SCI1 transfer complete	19	H'0026 to H'0027	
SCI3	SCI3 transmit end	21	H'002A to H'002B	
	SCI3 transmit data empty			
	SCI3 receive data full			
	SCI3 overrun error			
	SCI3 framing error			
	SCI3 parity error			
A/D	A/D conversion end	22	H'002C to H'002D	
(SLEEP instruction executed)	Direct transfer	23	H'002E to H'002F	Low

Note: Vector addresses H'0002 to H'0007, H'0012 to H'0013, H'0018 to H'001F, H'0024 to H'0025, H'0028 to H'0029 are reserved and cannot be used.

3.3.2 Interrupt Control Registers

Table 3.3 lists the registers that control interrupts.

Table 3.3 Interrupt Control Registers

Name	Abbreviation	R/W	Initial Value	Address
Interrupt edge select register 1	IEGR1	R/W	H'70	H'FFF2
Interrupt edge select register 2	IEGR2	R/W	H'00	H'FFF3
Interrupt enable register 1	IENR1	R/W	H'10	H'FFF4
Interrupt enable register 2	IENR2	R/W	H'00	H'FFF5
Interrupt enable register 3	IENR3	R/W	H'00	H'FFF6
Interrupt request register 1	IRR1	R/W*	H'10	H'FFF7
Interrupt request register 2	IRR2	R/W*	H'00	H'FFF8
Interrupt request register 3	IRR3	R/W*	H'00	H'FFF9

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

Interrupt Edge Select Register 1 (IEGR1)

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

IEGR1 is an 8-bit read/write register used to designate whether pins $\overline{IRQ}_3$ to $\overline{IRQ}_0$ are set to rising edge sensing or falling edge sensing. Upon reset, IEGR1 is initialized to H'70.

Bit 7—Reserved Bit: Bit 7 is reserved: it is always read as 0 and cannot be modified.

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

Bit 3—$\overline{IRQ}_3$ Edge Select (IEG3): Bit 3 selects the input sensing of pin $\overline{IRQ}_3$.

Bit 3: IEG3	Description	
0	Falling edge of $\overline{IRQ}_3$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ}_3$ pin input is detected	

Bit 2—$\overline{IRQ}_2$ Edge Select (IEG2): Bit 2 selects the input sensing of pin $\overline{IRQ}_2$.

Bit 2: IEG2	Description	
0	Falling edge of $\overline{IRQ}_2$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ}_2$ pin input is detected	

Bit 1—$\overline{IRQ}_1$ Edge Select (IEG1): Bit 1 selects the input sensing of pin $\overline{IRQ}_1$.

Bit 1: IEG1	Description	
0	Falling edge of $\overline{IRQ}_1$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ}_1$ pin input is detected	

Bit 0—$\overline{IRQ}_0$ Edge Select (IEG0): Bit 0 selects the input sensing of pin $\overline{IRQ}_0$.

Bit 0: IEG0	Description	
0	Falling edge of $\overline{IRQ}_0$ pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ}_0$ pin input is detected	

Interrupt Edge Select Register 2 (IEGR2)

Bit	7	6	5	4	3	2	1	0
	INTEG7	INTEG6	INTEG5	INTEG4	INTEG3	INTEG2	INTEG1	INTEG0
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

IEGR2 is an 8-bit read/write register, used to designate whether pins $\overline{INT}_7$ to $\overline{INT}_0$, TMIY, and TMIB are set to rising edge sensing or falling edge sensing. Upon reset, IEGR2 is initialized to H'00.

Bit 7—$\overline{INT}_7$ Edge Select (INTEG7): Bit 7 selects the input sensing of the $\overline{INT}_7$ pin and TMIY pin.

Bit 7: INTEG7	Description	
0	Falling edge of $\overline{INT}_7$ and TMIY pin input is detected	(initial value)
1	Rising edge of $\overline{INT}_7$ and TMIY pin input is detected	

Bit 6—$\overline{INT}_6$ Edge Select (INTEG6): Bit 6 selects the input sensing of the $\overline{INT}_6$ pin and TMIB pin.

Bit 6: INTEG6	Description	
0	Falling edge of $\overline{INT}_6$ and TMIB pin input is detected	(initial value)
1	Rising edge of $\overline{INT}_6$ and TMIB pin input is detected	

Bit 5—$\overline{INT}_5$ Edge Select (INTEG5): Bit 5 selects the input sensing of the $\overline{INT}_5$ pin and $\overline{ADTRG}$ pin.

Bit 5: INTEG5	Description	
0	Falling edge of $\overline{INT}_5$ and $\overline{ADTRG}$ pin input is detected	(initial value)
1	Rising edge of $\overline{INT}_5$ and $\overline{ADTRG}$ pin input is detected	

Bits 4 to 0—$\overline{INT}_4$ to $\overline{INT}_0$ Edge Select (INTEG4 to INTEG0): Bits 4 to 0 select the input sensing of pins $\overline{INT}_4$ to $\overline{INT}_0$.

Bit n: INTEGn	Description	
0	Falling edge of $\overline{INT}_n$ pin input is detected	(initial value)
1	Rising edge of $\overline{INT}_n$ pin input is detected	

(n = 4 to 0)

Interrupt Enable Register 1 (IENR1)

Bit	7	6	5	4	3	2	1	0
	IENTB1	IENTA	—	—	IEN3	IEN2	IEN1	IEN0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/W	R/W	—	—	R/W	R/W	R/W	R/W

IENR1 is an 8-bit read/write register that enables or disables interrupt requests. Upon reset, IENR1 is initialized to H'10.

Bit 7—Timer B1 Interrupt Enable (IENTB1): Bit 7 enables or disables timer B1 overflow interrupt requests.

Bit 7: IENTB1	Description	
0	Disables timer B1 interrupt requests	(initial value)
1	Enables timer B1 interrupt requests	

Bit 6—Timer A Interrupt Enable (IENTA): Bit 6 enables or disables timer A overflow interrupt requests.

Bit 6: IENTA	Description	
0	Disables timer A interrupt requests	(initial value)
1	Enables timer A interrupt requests	

Bit 5—Reserved Bit: Bit 5 is reserved: it is always read as 0 and cannot be modified.

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

Bits 3 to 0—IRQ_3 to IRQ_0 Interrupt Enable (IEN3 to IEN0): Bits 3 to 0 enable or disable IRQ_3 to IRQ_0 interrupt requests.

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

(n = 3 to 0)

Interrupt Enable Register 2 (IENR2)

Bit	7	6	5	4	3	2	1	0
	IENDT	IENAD	—	IENS1	—	—	—	—
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	—	R/W	—	—	—	—

IENR2 is an 8-bit read/write register that enables or disables interrupt requests. Upon reset, IENR2 is initialized to H'00.

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	

Bit 5—Reserved Bit: Bit 5 is reserved: it is always read as 0 and cannot be modified.

Bit 4—SCI1 Interrupt Enable (IENS1): Bit 4 enables or disables SCI1 transfer complete interrupt requests.

Bit 4: IENS1	Description	
0	Disables SCI1 interrupt requests	(initial value)
1	Enables SCI1 interrupt requests	

Bits 3 to 0—Reserved Bits: Bits 3 to 0 are reserved: they are always read as 0 and cannot be modified.

Interrupt Enable Register 3 (IENR3)

Bit	7	6	5	4	3	2	1	0
	INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0
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

IENR3 is an 8-bit read/write register that enables or disables INT_7 to INT_0 interrupt requests. Upon reset, IENR3 is initialized to H'00.

Bits 7 to 0—INT_7 to INT_0 Interrupt Enable (INTEN7 to INTEN0): Bits 7 to 0 enable or disable INT_7 to INT_0 interrupt requests.

Bit n: INTENn	Description	
0	Disables interrupt requests from pin $\overline{INT}_n$	(initial value)
1	Enables interrupt requests from pin $\overline{INT}_n$	

(n = 7 to 0)

Interrupt Request Register 1 (IRR1)

Bit	7	6	5	4	3	2	1	0
	IRRTB1	IRRTA	—	—	IRRI3	IRRI2	IRRI1	IRRI0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/W*	R/W*	—	—	R/W*	R/W*	R/W*	R/W*

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

IRR1 is an 8-bit read/write register, in which a corresponding flag is set to 1 when a timer B1, timer A, timer Y, or IRQ_3 to IRQ_0 interrupt is requested. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag. Upon reset, IRR1 is initialized to H'10.

Bit 7—Timer B1 Interrupt Request Flag (IRRTB1)

Bit 7: IRRTB1	Description
0	Clearing conditions: (initial value) When IRRTB1 = 1, it is cleared by writing 0
1	Setting conditions: When the timer B1 counter value overflows from H'FF to H'00

Bit 6—Timer A Interrupt Request Flag (IRRTA)

Bit 6: IRRTA	Description
0	Clearing conditions: (initial value) 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

Bit 5—Reserved Bit: Bit 5 is reserved: it is always read as 0 and cannot be modified.

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

Bits 3 to 0—IRQ_3 to IRQ_0 Interrupt Request Flags (IRRI3 to IRRI0)

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

(n = 3 to 0)

Interrupt Request Register 2 (IRR2)

Bit	7	6	5	4	3	2	1	0
	IRRDT	IRRAD	—	IRRS1	—	—	—	—
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/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, or SCI1 interrupt is requested. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag. Upon reset, IRR2 is initialized to H'00.

Bit 7—Direct Transfer Interrupt Request Flag (IRRDT)

Bit 7: IRRDT	Description
0	Clearing conditions: (initial value) When IRRDT = 1, it is cleared by writing 0
1	Setting conditions: When 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: (initial value) When IRRAD = 1, it is cleared by writing 0
1	Setting conditions: When A/D conversion is completed and ADSF is cleared to 0 in ADSR

Bit 5—Reserved bit: Bit 5 is reserved: it is always read as 0 and cannot be modified.

Bit 4—SCI1 Interrupt Request Flag (IRRS1)

Bit 4: IRRS1	Description
0	Clearing conditions: (initial value) When IRRS1 = 1, it is cleared by writing 0
1	Setting conditions: When an SCI1 transfer is completed

Bits 3 to 0—Reserved Bits: Bits 3 to 0 are reserved: they are always read as 0 and cannot be modified.

Interrupt Request Register 3 (IRR3)

Bit	7	6	5	4	3	2	1	0
	INTF7	INTF6	INTF5	INTF4	INTF3	INTF2	INTF1	INTF0
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.

IRR3 is an 8-bit read/write register, in which a corresponding flag is set to 1 by a transition at pin $\overline{INT}_7$ to $\overline{INT}_0$. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag. Upon reset, IRR3 is initialized to H'00.

Bits 7 to 0—INT_7 to INT_0 Interrupt Request Flags (INTF7 to INTF0)

Bit n: INTFn	Description
0	Clearing conditions: (initial value) When INTFn = 1, it is cleared by writing 0
1	Setting conditions: When the designated signal edge is input at pin $\overline{INT}_n$

(n = 7 to 0)

3.3.3 External Interrupts

There are 12 external interrupts: IRQ_3 to IRQ_0 and INT_7 to INT_0.

Interrupts IRQ_3 to IRQ_0: Interrupts IRQ_3 to IRQ_0 are requested by input signals to pins $\overline{IRQ_3}$ to $\overline{IRQ_0}$. These interrupts are detected by either rising edge sensing or falling edge sensing, depending on the settings of bits IEG3 to IEG0 in IEGR1.

When these pins are designated as pins $\overline{IRQ_3}$ to $\overline{IRQ_0}$ in port mode register 1 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 IEN3 to IEN0 to 0 in IENR1. These interrupts can all be masked by setting the I bit to 1 in CCR.

When IRQ_3 to IRQ_0 interrupt exception handling is initiated, the I bit is set to 1 in CCR. Vector numbers 7 to 4 are assigned to interrupts IRQ_3 to IRQ_0. The order of priority is from IRQ_0 (high) to IRQ_3 (low). Table 3.2 gives details.

INT Interrupts: INT interrupts are requested by input signals to pins $\overline{INT_7}$ to $\overline{INT_0}$. These interrupts are detected by either rising edge sensing or falling edge sensing, depending on the settings of bits INTEG7 to INTEG0 in IEGR2.

When the designated edge is input at pins $\overline{INT_7}$ to $\overline{INT_0}$, the corresponding bit in IRR3 is set to 1, requesting an interrupt. Recognition of these interrupt requests can be disabled individually by clearing bits INTEN7 to INTEN0 to 0 in IENR3. These interrupts can all be masked by setting the I bit to 1 in CCR.

When INT interrupt exception handling is initiated, the I bit is set to 1 in CCR. Vector number 8 is assigned to the INT interrupts. All eight interrupts have the same vector number, so the interrupt-handling routine must discriminate the interrupt source.

Note: Pins $\overline{INT_7}$ to $\overline{INT_0}$ are multiplexed with port 5. Even in port usage of these pins, whenever the designated edge is input or output, the corresponding bit INTFn is set to 1.

3.3.4 Internal Interrupts

There are 21 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 to 0. 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 23 to 9 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.

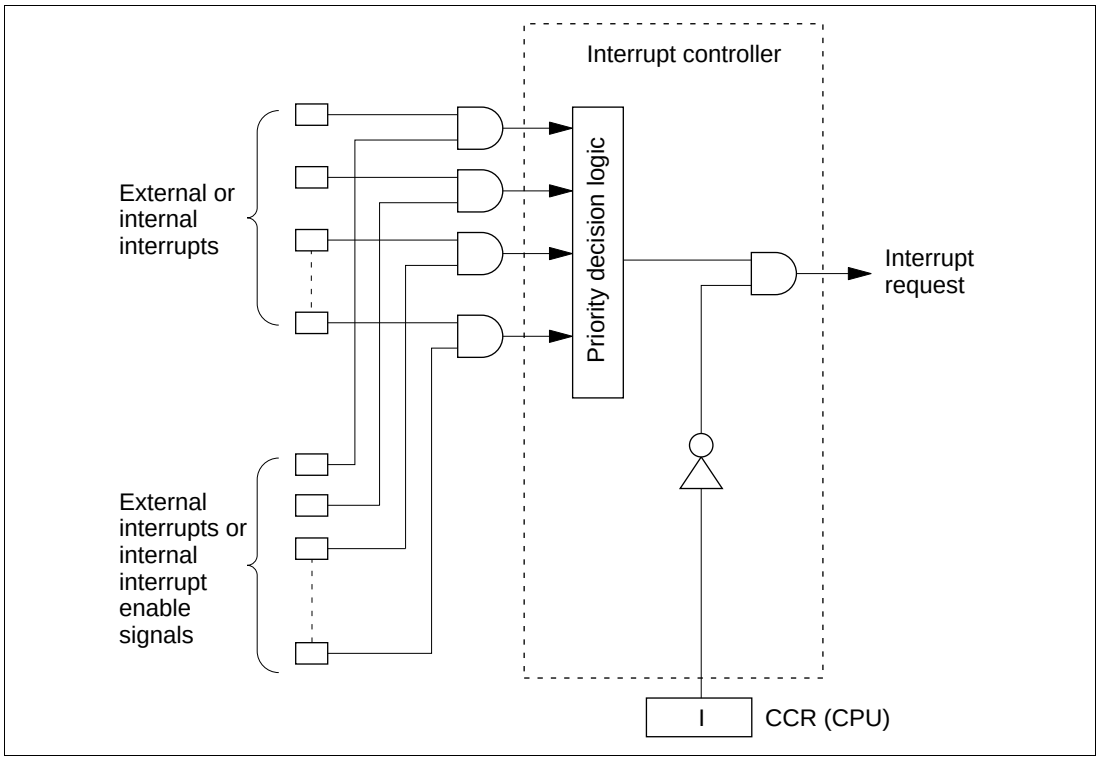


Figure 3.2 Block Diagram of Interrupt Controller

Interrupt operation is described as follows.

- If an interrupt occurs 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.

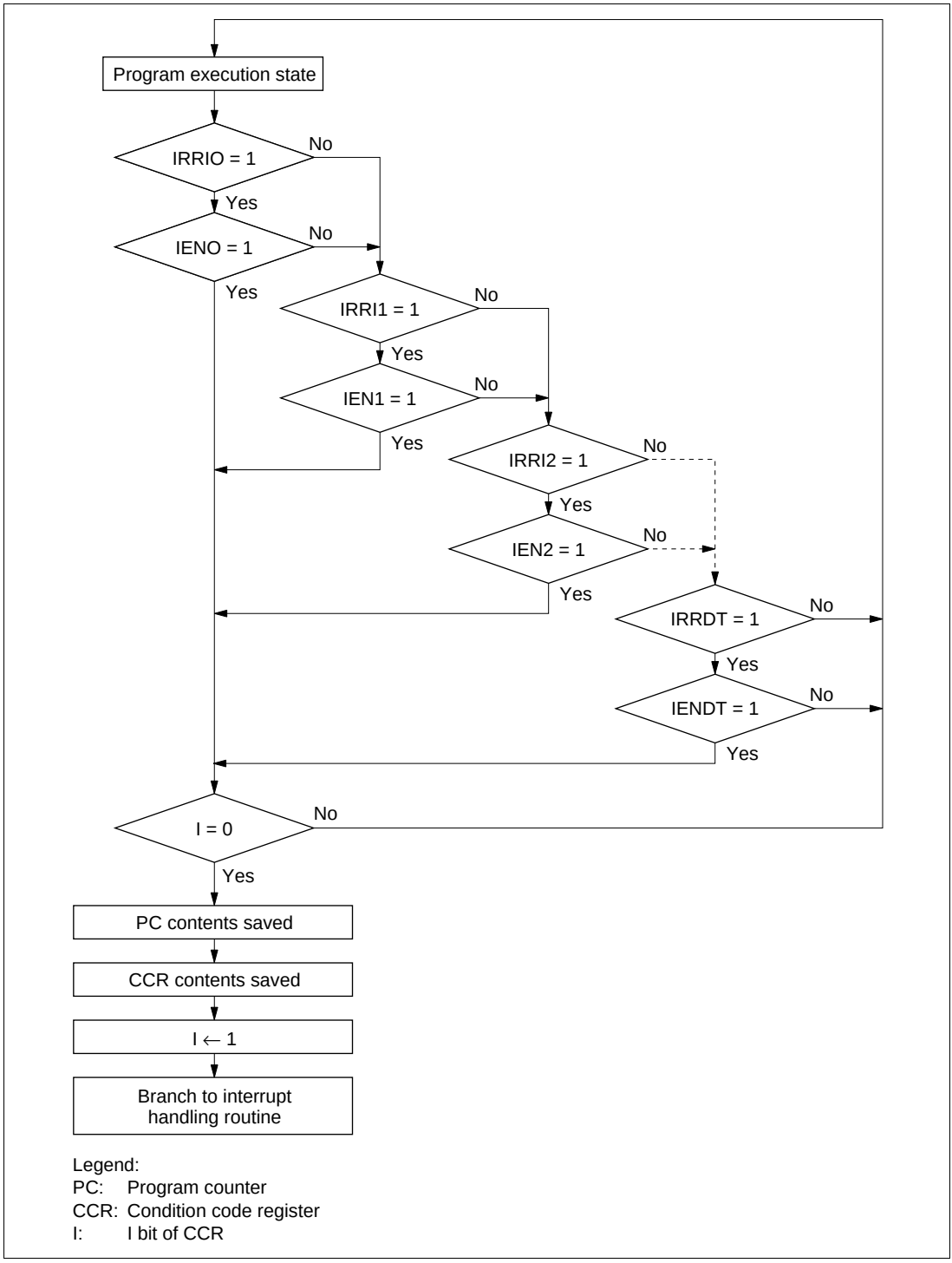
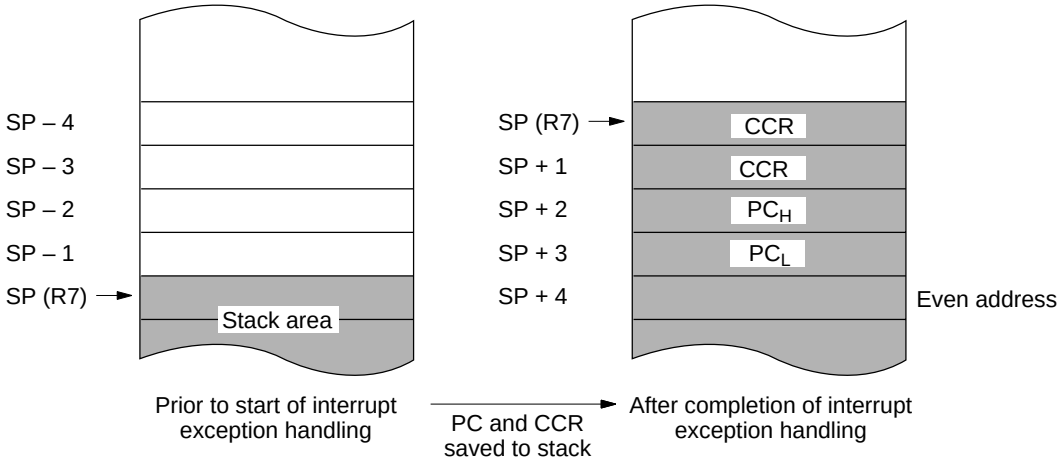


Figure 3.3 Flow up to Interrupt Acceptance



Legend:
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.

Figure 3.4 Stack State after Completion of Interrupt Exception Handling

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

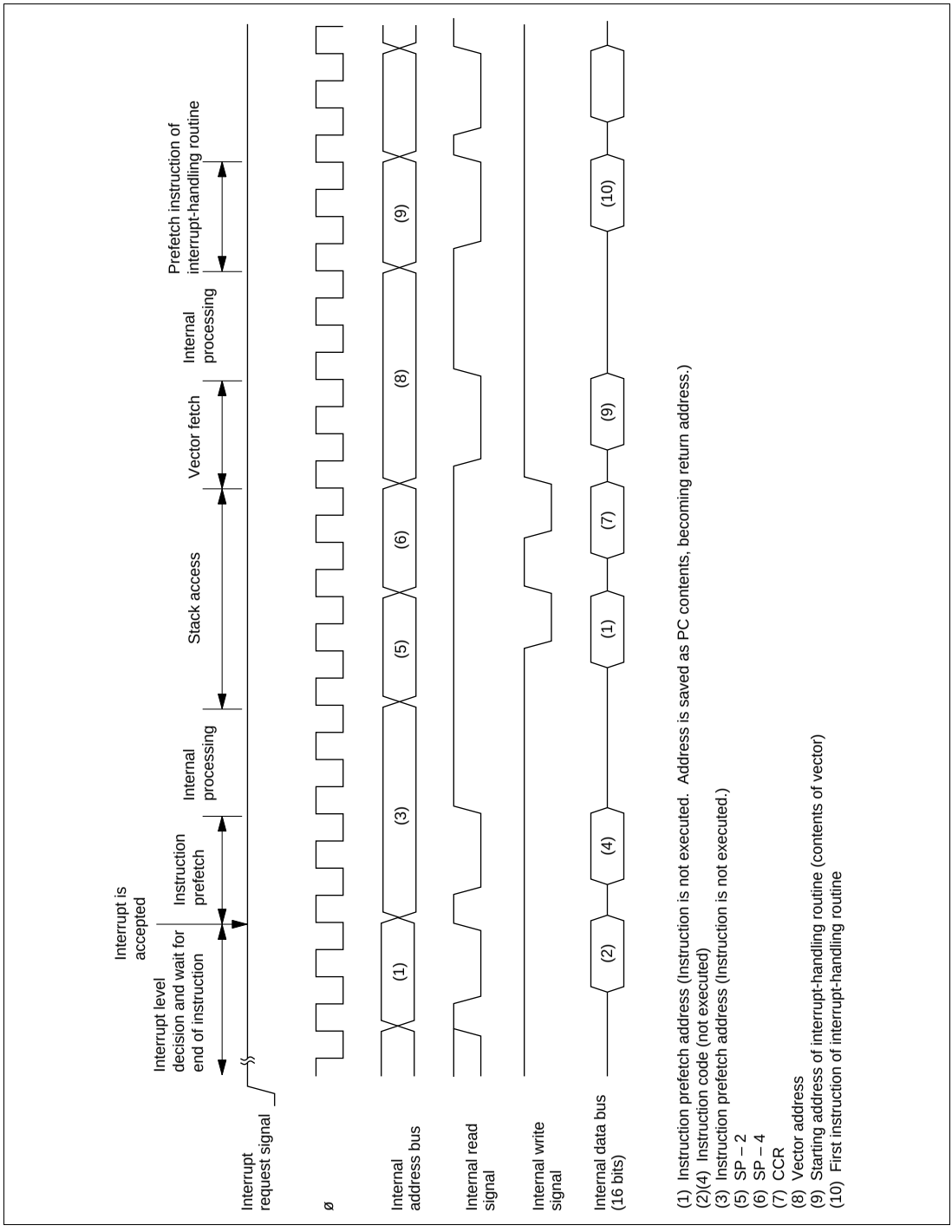


Figure 3.5 Interrupt Sequence

3.3.6 Interrupt Response Time

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

Table 3.4 Interrupt Wait States

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

Note: * Not including EEPMOV instruction.

3.4 Application Notes

3.4.1 Notes on Stack Area Use

When word data is accessed in the H8/3644 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.

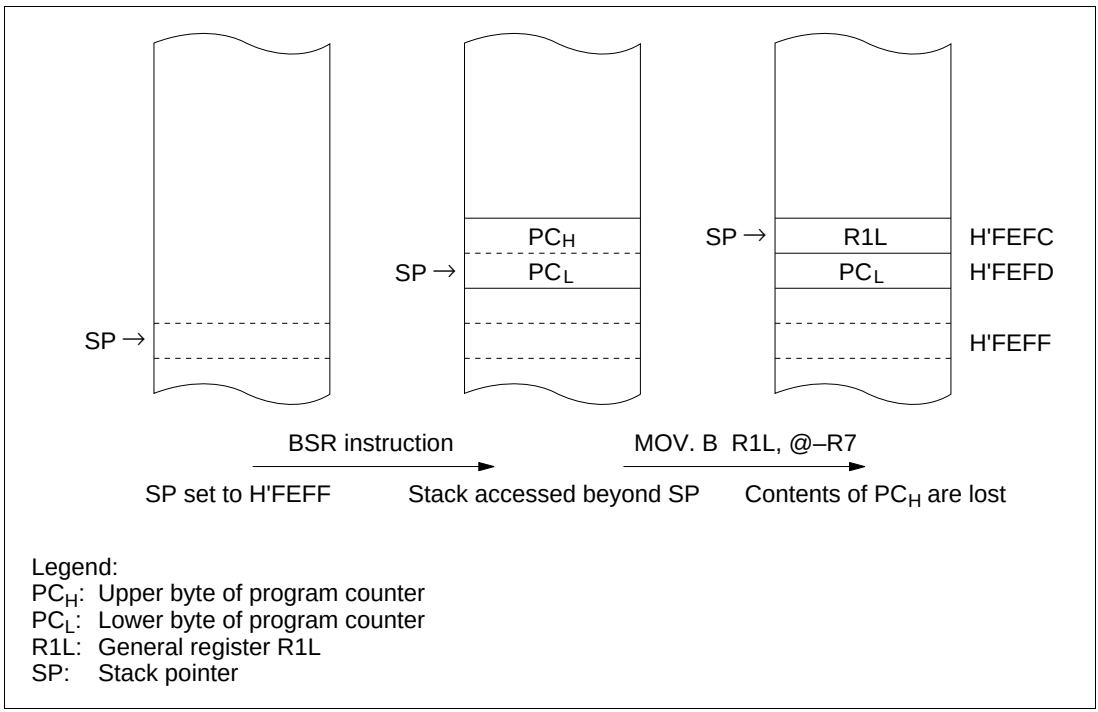


Figure 3.6 Operation when Odd Address is Set in SP

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 $\overline{IRQ}_3$ to $\overline{IRQ}_1$, 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. 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	IRRI3	When PMR1 bit IRQ3 is changed from 0 to 1 while pin $\overline{IRQ}_3$ is low and IEGR bit IEG3 = 0.
		When PMR1 bit IRQ3 is changed from 1 to 0 while pin $\overline{IRQ}_3$ is low and IEGR bit IEG3 = 1.
	IRRI2	When PMR1 bit IRQ2 is changed from 0 to 1 while pin $\overline{IRQ}_2$ is low and IEGR bit IEG2 = 0.
		When PMR1 bit IRQ2 is changed from 1 to 0 while pin $\overline{IRQ}_2$ is low and IEGR bit IEG2 = 1.
	IRRI1	When PMR1 bit IRQ1 is changed from 0 to 1 while pin $\overline{IRQ}_1$ is low and IEGR bit IEG1 = 0.
		When PMR1 bit IRQ1 is changed from 1 to 0 while pin $\overline{IRQ}_1$ is low and IEGR bit IEG1 = 1.

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

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

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

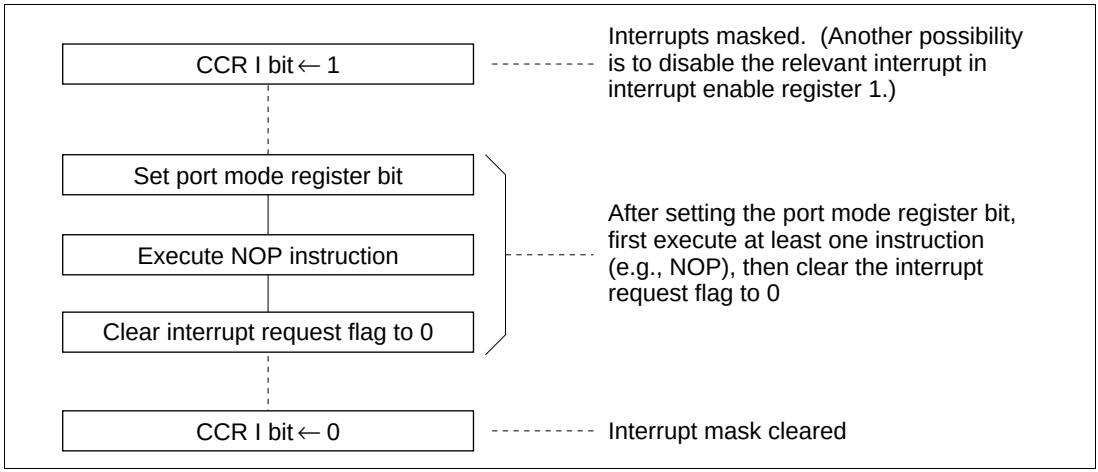


Figure 3.7 Port Mode Register Setting and Interrupt Request Flag Clearing Procedure

Section 4 Clock Pulse Generators

4.1 Overview

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

4.1.1 Block Diagram

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

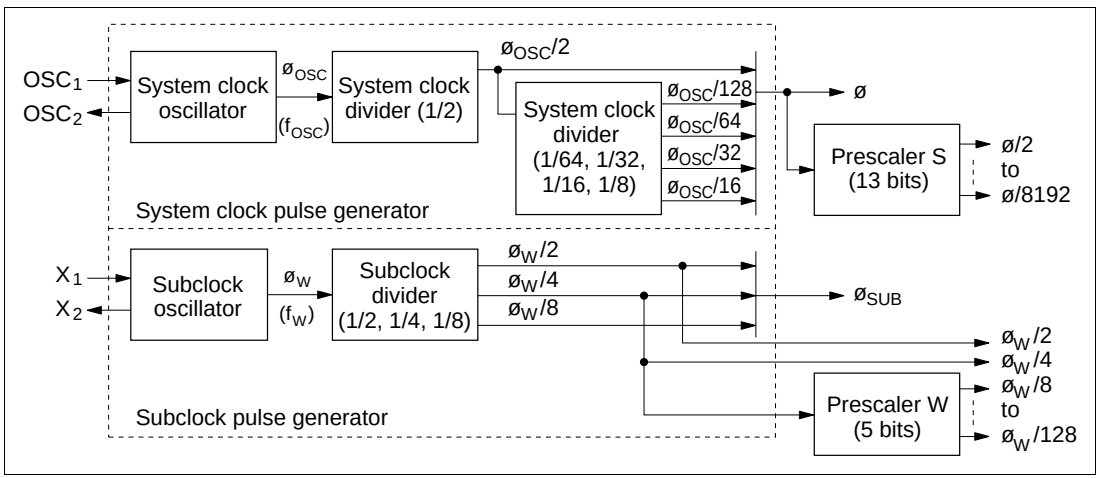


Figure 4.1 Block Diagram of Clock Pulse Generators

4.1.2 System Clock and Subclock

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

The clock signals available for use by peripheral modules are ø/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, $ø_W/2$, $ø_W/4$, $ø_W/8$, $ø_W/16$, $ø_W/32$, $ø_W/64$, 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.

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

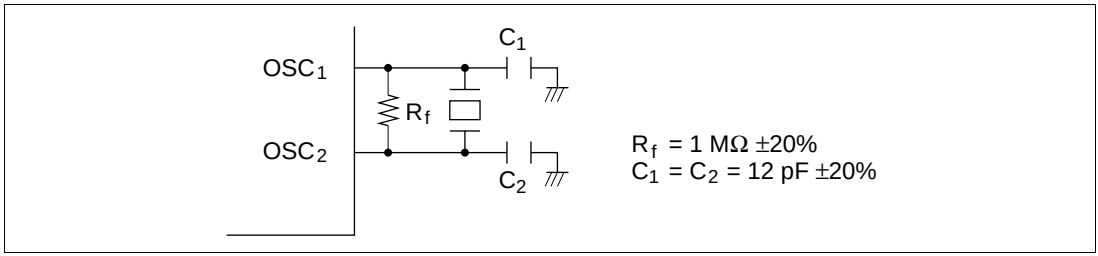


Figure 4.2 Typical Connection to Crystal Oscillator

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

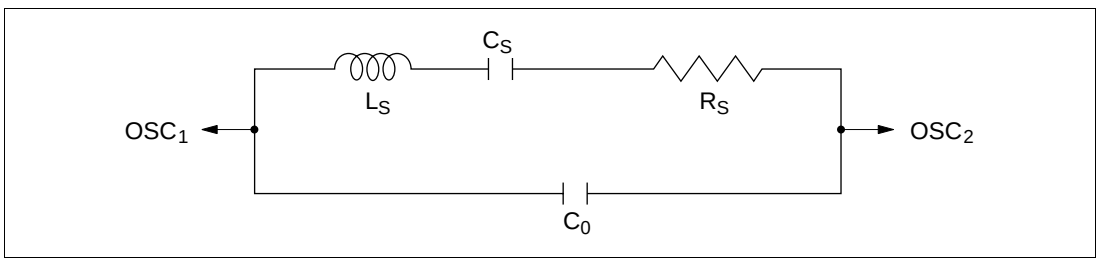


Figure 4.3 Equivalent Circuit of Crystal Oscillator

Table 4.1 Crystal Oscillator Parameters

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

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

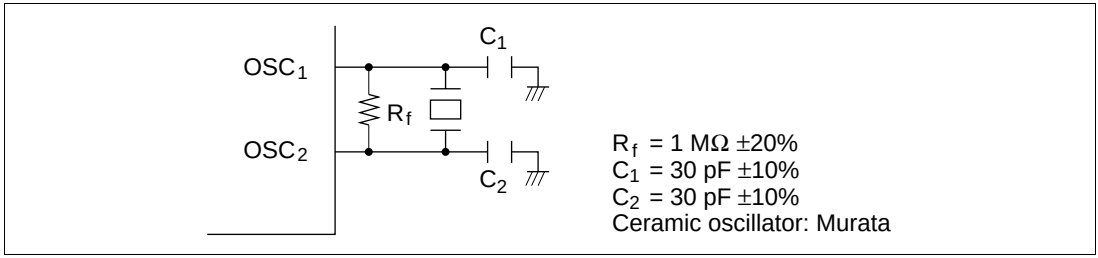


Figure 4.4 Typical Connection to Ceramic Oscillator

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

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

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

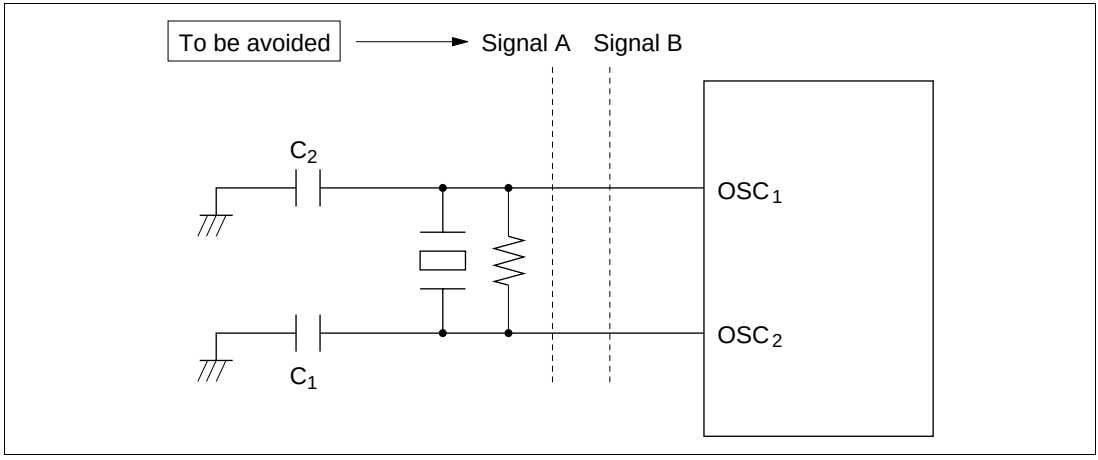


Figure 4.5 Board Design of Oscillator Circuit

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

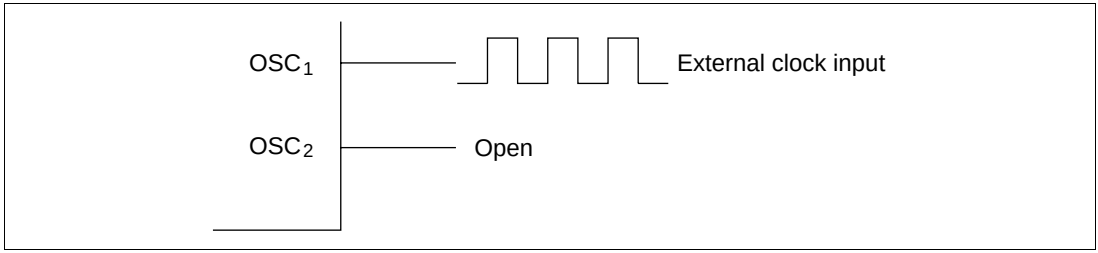


Figure 4.6 External Clock Input (Example)

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

4.3 Subclock Generator

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

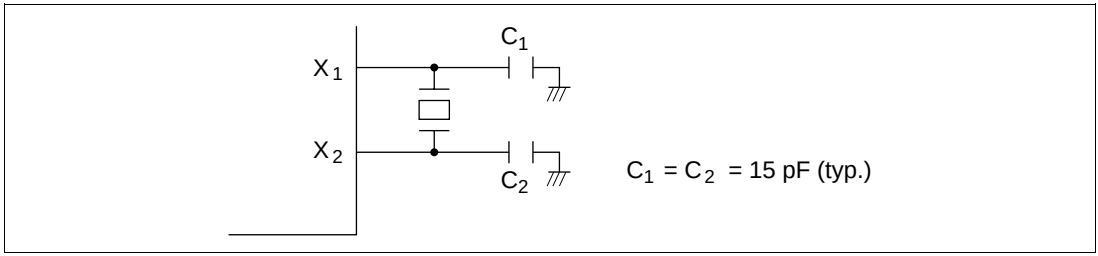


Figure 4.7 Typical Connection to 32.768-kHz Crystal Oscillator

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

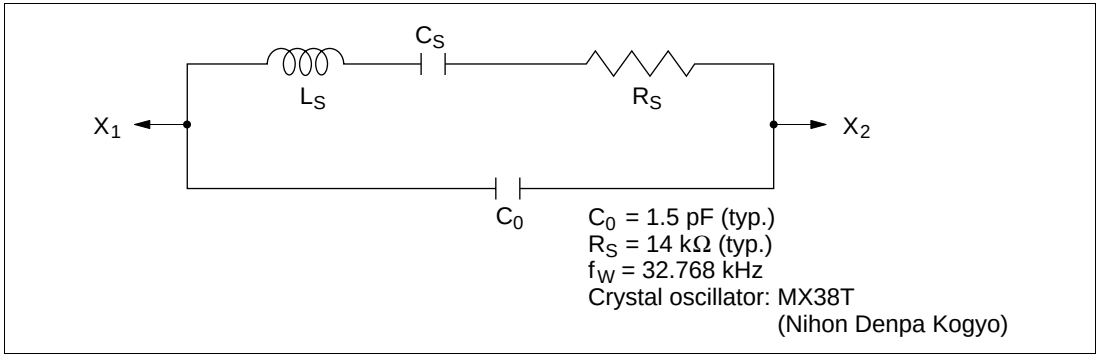


Figure 4.8 Equivalent Circuit of 32.768-kHz Crystal Oscillator

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

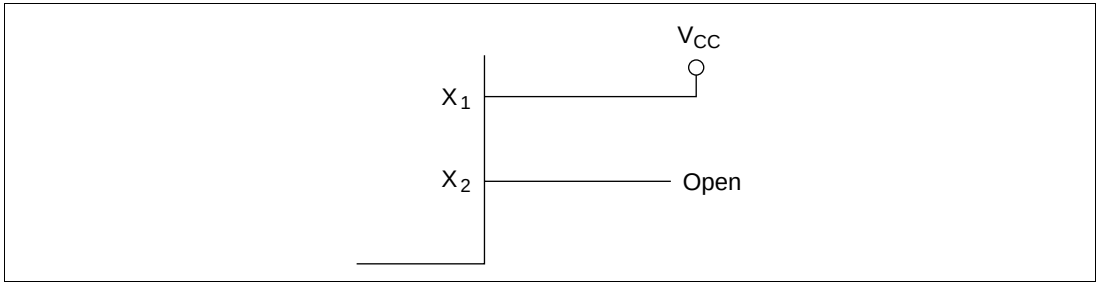


Figure 4.9 Pin Connection when not Using Subclock

4.4 Prescalers

The H8/3644 Series is equipped with two on-chip prescalers having different input clocks (prescaler S and prescaler W). Prescaler S is a 13-bit counter using the system clock (ø) as its input clock. Its prescaled outputs provide internal clock signals for on-chip peripheral modules. Prescaler W is a 5-bit counter using a 32.768-kHz signal divided by 4 ($ø_W/4$) as its input clock. Its prescaled outputs are used by timer A as a time base for timekeeping.

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

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

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

The CPU cannot read or write prescaler S.

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

In active (medium-speed) mode the clock input to prescaler S is determined by the division factor designated by MA1 and MA0 in SYSCR1.

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

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

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

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

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

4.5 Note on Oscillators

Oscillator characteristics are closely related to board design and should be carefully evaluated by the user, 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.

Section 5 Power-Down Modes

5.1 Overview

The H8/3644 Series has eight modes of operation after a reset. These include seven power-down modes, in which power dissipation is significantly reduced. Table 5.1 gives a summary of the eight operating modes.

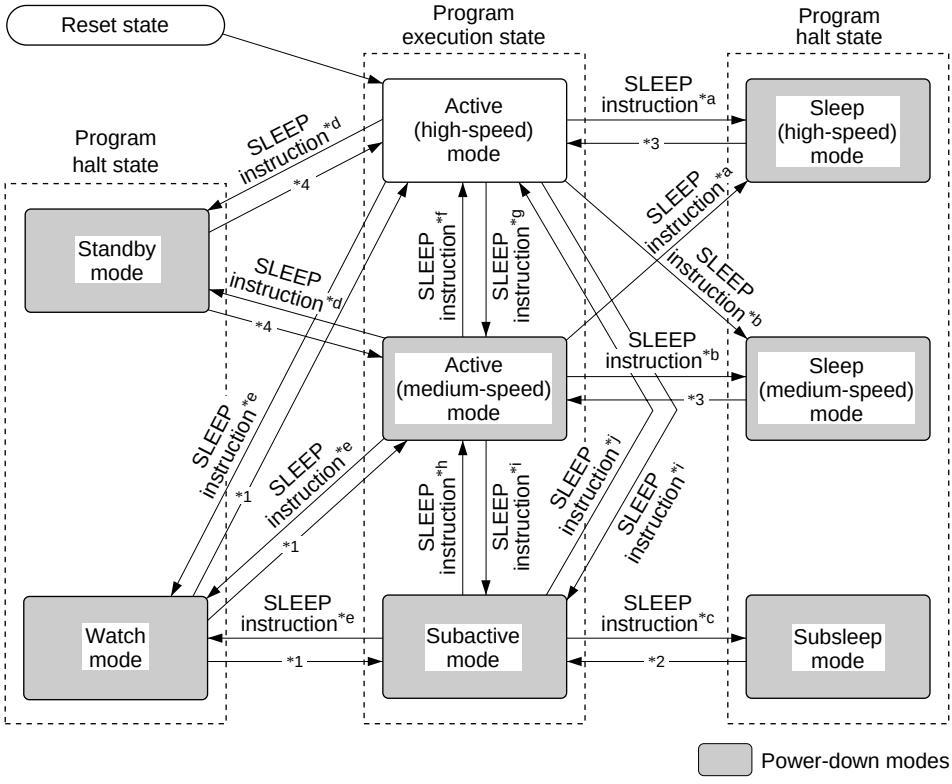
Table 5.1 Operating Modes

Operating Mode	Description
Active (high-speed) mode	The CPU and all on-chip peripheral functions are operable on the system clock
Active (medium-speed) mode	The CPU and all on-chip peripheral functions are operable on the system clock, but at 1/64, 1/32, 1/6, or 1/8* the speed in active (high-speed) mode
Subactive mode	The CPU, and the time-base function of timer A are operable on the subclock
Sleep (high-speed) mode	The CPU halts. On-chip peripheral functions except PWM are operable on the system clock
Sleep (medium-speed) mode	The CPU halts. On-chip peripheral functions except PWM are operable on the system clock, but at 1/64, 1/32, 1/6, or 1/8* the speed in active (high-speed) mode
Subsleep mode	The CPU halts. The time-base function of timer A are operable on the subclock
Watch mode	The CPU halts. The time-base function of timer A is operable on the subclock
Standby mode	The CPU and all on-chip peripheral functions halt

Note: * Determined by the value set in bits MA1 and MA0 of system control register 1 (SYSCR1).

Of these eight 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, and the two sleep modes (high-speed and medium speed) will be referred to collectively as sleep 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 interrupt, IRQ_0 interrupt
2	Timer a interrupt, IRQ_3 to IRQ_0 interrupts, INT interrupt
3	All interrupts
4	IRQ_1 or IRQ_0 interrupt

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: High-Speed	Active Mode: Medium-Speed	Sleep Mode: High-Speed	Sleep Mode: 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
	Registers			Retained	Retained	Retained		Retained	Retained
	RAM								
	I/O ports								Retained*1
External interrupts	IRQ_0	Functions	Functions	Functions	Functions	Functions	Functions	Functions	Functions
	IRQ_1					Retained*2			
	IRQ_2								Retained*2
	IRQ_3								
	INT_0	Functions	Functions	Functions	Functions	Retained*2	Functions	Functions	Retained*2
	INT_1								
	INT_2								
	INT_3								
	INT_4								
	INT_5								
	INT_6								
	INT_7								
Peripheral functions	Timer A	Functions	Functions	Functions	Functions	Functions*3	Functions*3	Functions*3	Retained
	Timer B1					Retained	Retained	Retained	
	Timer V					Reset	Reset	Reset	Reset
	Timer X								
	Watchdog timer					Retained	Retained	Retained	Retained
	SCI1								
	SCI3					Reset	Reset	Reset	Reset
	PWM			Retained	Retained	Retained	Retained	Retained	Retained
	A/D converter			Functions	Functions				

Notes: 1. Register contents are retained, but output is high-impedance state.
2. External interrupt requests are ignored. Interrupt request register contents are not altered.
3. Functions if timekeeping time-base function is selected.

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'E0	H'FFF1

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 • When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode (initial value)
1	• When a SLEEP instruction is executed in active mode, a transition is made to standby mode or watch mode • When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode

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

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

Note: * Don't care

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

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

Bits 2—Reserved Bits: 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	$ø_{osc}/16$	
	1	$ø_{osc}/32$	
1	0	$ø_{osc}/64$	
	1	$ø_{osc}/128$	(initial value)

System Control Register 2 (SYSCR2)

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

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

Upon reset, SYSCR2 is initialized to H'E0.

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

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

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

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

Bit 3: DTON	Description
0	• When a SLEEP instruction is executed in active mode, a transition is made to standby mode, watch mode, or sleep mode • When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode or subsleep mode (initial value)
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), active (medium-speed), or sleep (medium-speed) mode.

Bit 2: MSON	Description
0	• After standby, watch, or sleep mode is cleared, operation is in active (high-speed) mode • When a SLEEP instruction is executed in active mode, a transition is made to sleep (high-speed) mode (initial value)
1	• After standby, watch, or sleep mode is cleared, operation is in active (medium-speed) mode • When a SLEEP instruction is executed in active mode, a transition is made to sleep (medium-speed) mode

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

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

Note: * Don't care

5.2 Sleep Mode

5.2.1 Transition to Sleep Mode

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 and the MSON and DTON bits in SYSCR2 are all cleared to 0. In sleep (high-speed) mode CPU operation is halted but the on-chip peripheral functions other than PWM are operational. CPU register contents are retained.

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 other than PWM 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.

5.2.2 Clearing Sleep Mode

Sleep mode is cleared by any interrupt (timer A, timer B1, timer X, timer V, IRQ_3 to IRQ_0, INT_7 to INT_0, SCI_3, SCI_1, or 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.

- 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) 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–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 5.4 gives settings for various operating frequencies. Set bits STS2 to STS0 for a waiting time of at least 10 ms.

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

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

Note: * Don't care

- When an external clock is used

  Any values may be set. Normally the minimum time (STS2 = STS1 = STS0 = 0) should be set.

5.4 Watch Mode

5.4.1 Transition to Watch Mode

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

In watch mode, operation of on-chip peripheral modules other than timer A is halted. As long as a minimum required voltage is applied, the contents of CPU registers, 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 or IRQ_0) or by input at the $\overline{RES}$ pin.

- Clearing by interrupt

  When watch mode is cleared by a timer A interrupt or IRQ_0 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 5.3.2, Clearing Standby Mode.

5.4.3 Oscillator Settling Time after Watch Mode is Cleared

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

5.5 Subsleep Mode

5.5.1 Transition to Subsleep Mode

The system goes from subactive mode to subsleep mode when a SLEEP instruction is executed while the SSBY bit in SYSCR1 is cleared to 0, LSON 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 timer A 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, IRQ_3 to IRQ_0, INT_7 to INT_0) or by input at the $\overline{RES}$ pin.

- Clearing by interrupt

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

- Clearing by $\overline{RES}$ input

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

5.6 Subactive Mode

5.6.1 Transition to Subactive Mode

Subactive mode is entered from watch mode if a timer A or IRQ_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, IRQ_3 to IRQ_0, or INT_7 to INT_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 input at the $\overline{RES}$ pin.

- Clearing by SLEEP instruction

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

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

5.6.3 Operating Frequency in Subactive Mode

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

5.7 Active (Medium-Speed) Mode

5.7.1 Transition to Active (Medium-Speed) Mode

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

5.7.2 Clearing Active (Medium-Speed) Mode

Active (medium-speed) mode is cleared by a SLEEP instruction or by input at the $\overline{RES}$ pin.

- 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 (high-speed) mode is entered if MSON is cleared to 0 in SYSCR2, and sleep (medium-speed) mode is entered if MSON is set to 1. 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 goes low, the CPU enters 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

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.

Section 6 ROM

6.1 Overview

The H8/3644 has 32 kbytes of on-chip mask ROM, PROM or flash memory. The H8/3643 has 24 kbytes of mask ROM or flash memory. The H8/3642 has 16 kbytes of mask ROM or flash memory. The H8/3641 has 12 kbytes of on-chip ROM. H8/3640 has 8 kbytes of ROM. 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.

In the PROM version (H8/3644 ZTAT) and flash memory versions (H8/3644 F-ZTAT, H8/3643 F-ZTAT, H8/3642 AF-ZTAT), programs can be written and erased with a general-purpose PROM programmer. In the on-chip flash memory versions, programs can be written and erased on-board.

6.1.1 Block Diagram

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

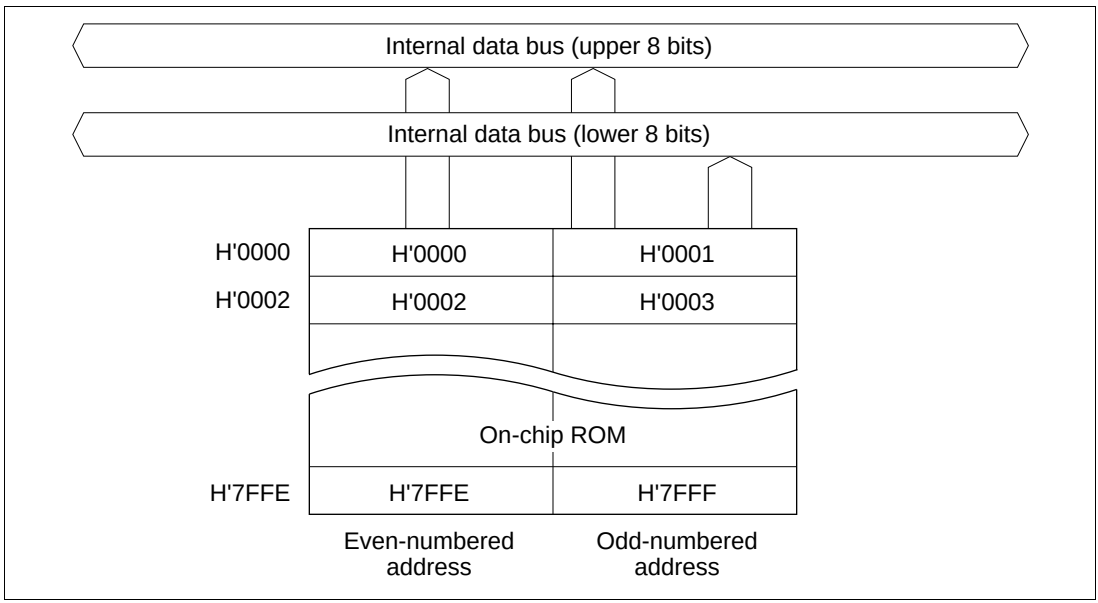


Figure 6.1 ROM Block Diagram (H8/3644)

6.2 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 HN27C256 EPROM. 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_4/AN_4	Low level
PB_5/AN_5	
PB_6/AN_6	High level

6.2.2 Socket Adapter Pin Arrangement and Memory Map

A general-purpose PROM programmer can be used to program the PROM. A socket adapter is required for conversion to 28 pins, as listed in table 6.2.

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

Table 6.2 Socket Adapter

Package	Socket Adapter
64-pin QFP (FP-64A)	Under development
64-pin SDIP (DP-64S)	Under development
80-pin TQFP (TFP-80C)	Under development

H8/3644				HN27C256 (28-pin)	
Pin #			Pin name	Pin name	Pin #
FP-64A	DP-64S	TFP-80C			
10	18	12	$\overline{RES}$	V_{PP}	1
17	25	22	$P6_0$	EO_0	11
18	26	23	$P6_1$	EO_1	12
19	27	24	$P6_2$	EO_2	13
20	28	25	$P6_3$	EO_3	15
21	29	26	$P6_4$	EO_4	16
22	30	27	$P6_5$	EO_5	17
23	31	28	$P6_6$	EO_6	18
24	32	29	$P6_7$	EO_7	19
46	54	57	$P8_7$	EA_0	10
45	53	56	$P8_6$	EA_1	9
44	52	55	$P8_5$	EA_2	8
43	51	54	$P8_4$	EA_3	7
42	50	52	$P8_3$	EA_4	6
41	49	51	$P8_2$	EA_5	5
40	48	50	$P8_1$	EA_6	4
39	47	49	$P8_0$	EA_7	3
55	63	69	$P1_5$	EA_8	25
16	24	19	$\overline{IRQ}_0$	EA_9	24
57	1	71	$P1_7$	EA_{10}	21
34	42	43	$P7_3$	EA_{11}	23
35	43	44	$P7_4$	EA_{12}	2
36	44	45	$P7_5$	EA_{13}	26
37	45	46	$P7_6$	EA_{14}	27
38	46	47	$P7_7$	$\overline{CE}$	20
56	64	70	$P1_6$	$\overline{OE}$	22
33	41	42	V_{CC}	V_{CC}	28
58	2	72	AV_{CC}		
4	12	5	TEST		
6	14	7	X_1		
60	4	74	PB_6		
47	55	58	$P2_0$		
48	56	59	$P2_1$		
7	15	8, 11	V_{SS}	V_{SS}	14
3	11	4	AV_{SS}		
62	6	76	PB_4		
61	5	75	PB_5		
52	60	66	$P3_0$		

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

Figure 6.2 Socket Adapter Pin Correspondence (ZTAT)

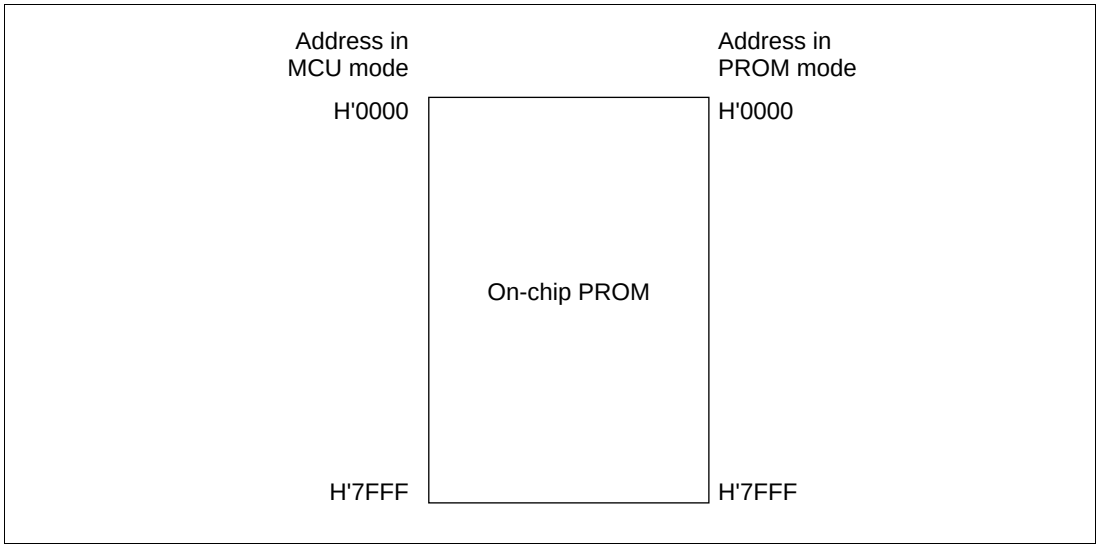


Figure 6.3 H8/3644 Memory Map in PROM Mode

When programming with a PROM programmer, be sure to specify addresses from H'0000 to H'7FFF.

6.3 Programming

The H8/3644 write, verify, and other modes are selected as shown in table 6.3 in PROM mode.

Table 6.3 Mode Selection in PROM Mode (H8/3644)

	Pin					
Mode	$\overline{\textbf{CE}}$	$\overline{\textbf{OE}}$	$\textbf{V}_{\textbf{PP}}$	$\textbf{V}_{\textbf{CC}}$	$\textbf{EO}_7$ **to** $\textbf{EO}_0$	$\textbf{EA}_{14}$ **to** $\textbf{EA}_0$
Write	L	H	V_{PP}	V_{CC}	Data input	Address input
Verify	H	L	V_{PP}	V_{CC}	Data output	Address input
Programming disabled	H	H	V_{PP}	V_{CC}	High impedance	Address input

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 HN27C256 EPROM.

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. Data in unused address areas has a value of H'FF. The basic flow of this high-speed, high-reliability programming method is shown in figure 6.4.

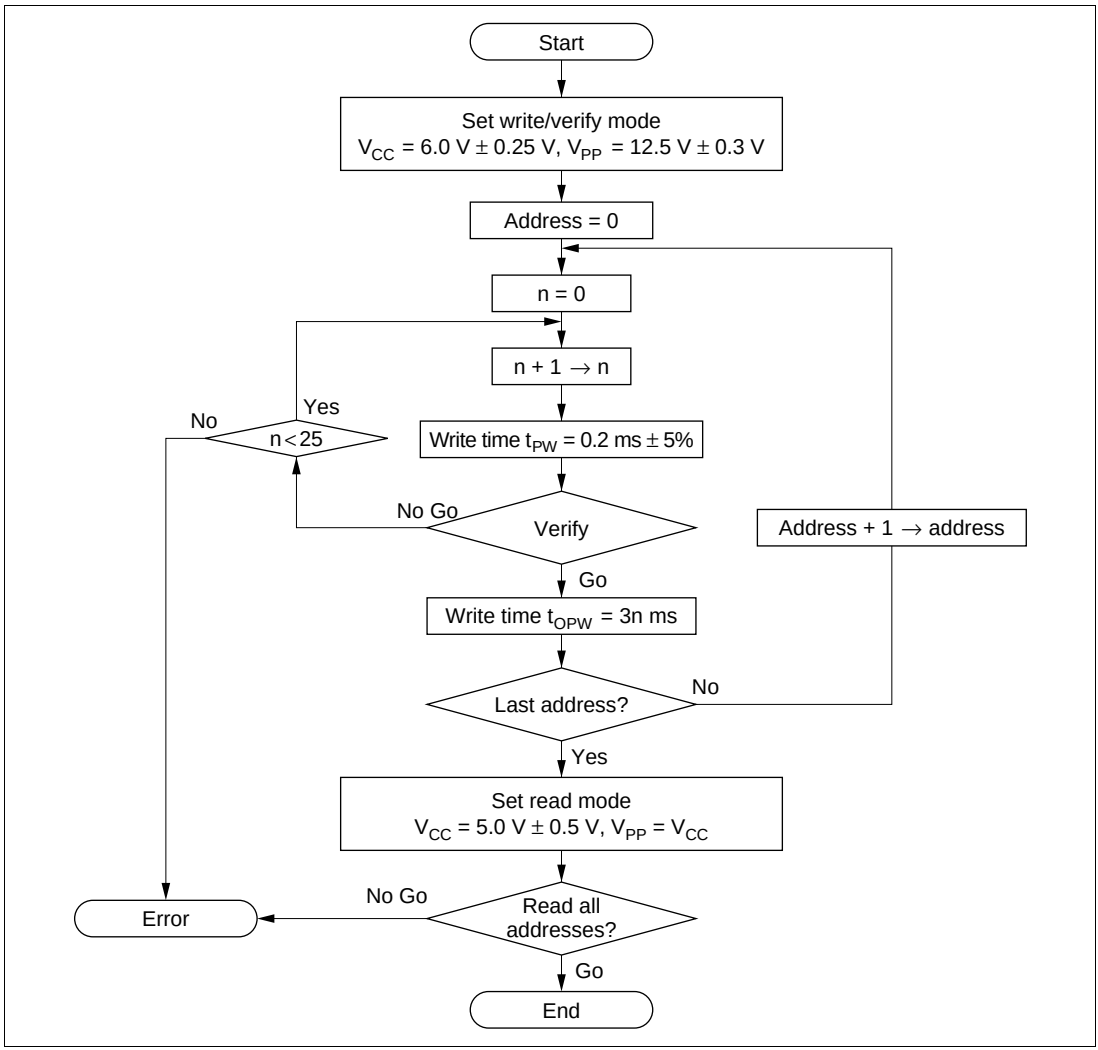


Figure 6.4 High-Speed, High-Reliability Programming Flow Chart

Table 6.4 and table 6.5 give the electrical characteristics in programming mode.

Table 6.4 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_{14} to EA_0 $\overline{OE}$, $\overline{CE}$	V_{IH}	2.4	—	V_{CC} + 0.3	V	
Input low-level voltage	EO_7 to EO_0, EA_{14} to EA_0 $\overline{OE}$, $\overline{CE}$	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_{14} to EA_0 $\overline{OE}$, $\overline{CE}$	$\|I_{LI}\|$	—	—	2	µA	V_{in} = 5.25 V/ 0.5 V
V_{CC} current		I_{CC}	—	—	40	mA	
V_{PP} current		I_{PP}	—	—	40	mA	

Table 6.5 AC Characteristics

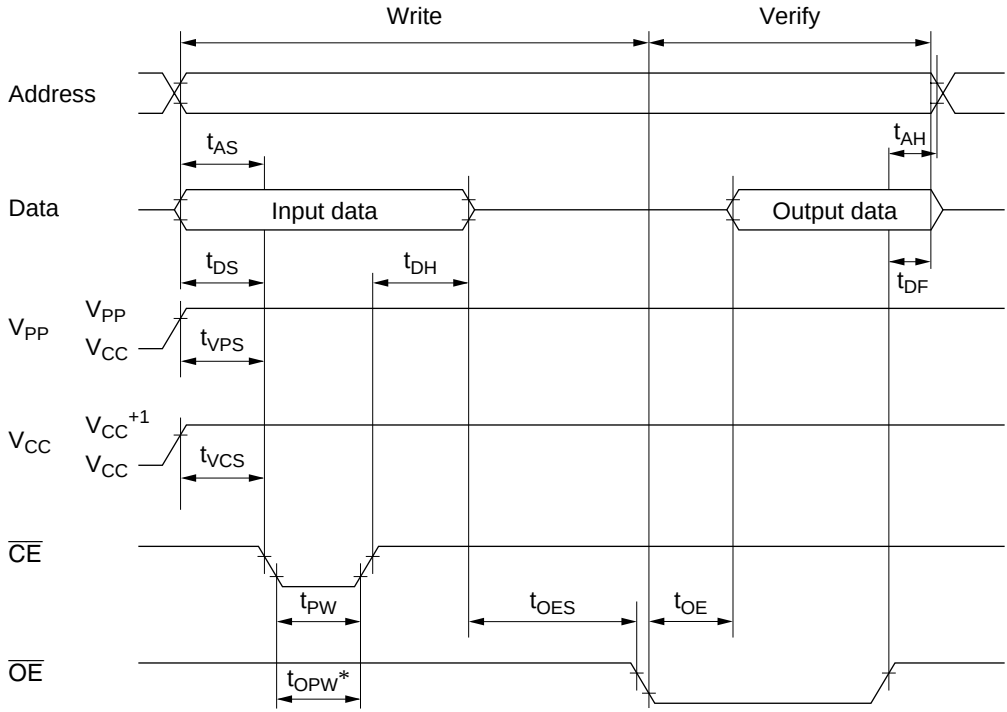
(Conditions: V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Typ	Max	Unit	Test Condition
Address setup time	t_{AS}	2	—	—	µs	Figure 6.5*[1]
$\overline{OE}$ 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]	0	—	130	ns	
V_{PP} setup time	t_{VPS}	2	—	—	µs	
Programming pulse width	t_{PW}	0.95	1.0	1.05	ms	
$\overline{CE}$ pulse width for overwrite programming	t_{OPW}*[3]	2.85	—	78.7	ms	
V_{CC} setup time	t_{VCS}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	500	ns	

Notes: 1. Input pulse level: 0.8 V to 2.2 V
Input rise time/fall time ≤ 20 ns
Timing reference levels: Input: 1.0 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 given in figure 6.4, High-Speed, High-Reliability Programming Flow Chart.

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 HN27C256 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.

6.3.3 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.

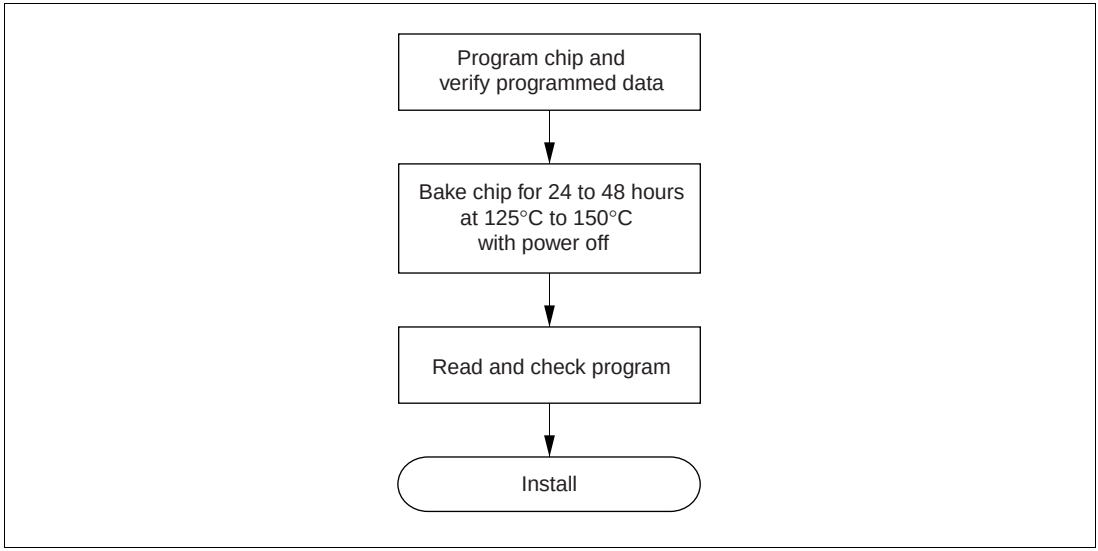


Figure 6.6 Recommended Screening Procedure

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, using a microcomputer with on-chip EPROM in a windowed package, etc. Please inform Hitachi of any abnormal conditions noted during or after programming or in screening of program data after high-temperature baking.

6.4 Flash Memory Overview

6.4.1 Principle of Flash Memory Operation

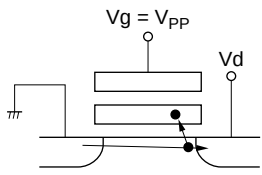
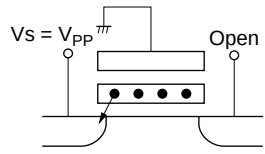
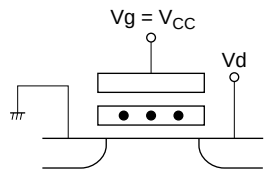
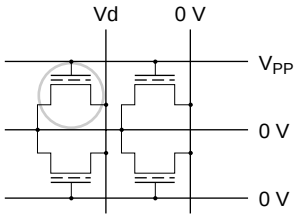
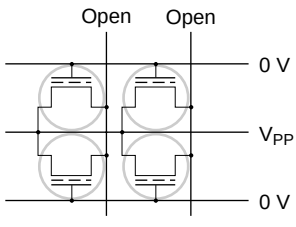
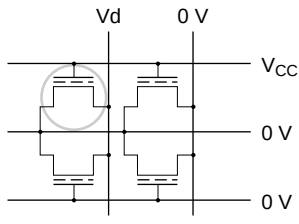
Table 6.6 illustrates the principle of operation of the on-chip flash memory in the H8/3644F, H8/3643F, and H8/3642AF.

Like EPROM, flash memory is programmed by applying a high gate-to-drain voltage that draws hot electrons generated in the vicinity of the drain into a floating gate. The threshold voltage of a programmed memory cell is therefore higher than that of an erased cell. Cells are erased by grounding the gate and applying a high voltage to the source, causing the electrons stored in the

floating gate to tunnel out. After erasure, the threshold voltage drops. A memory cell is read like an EPROM cell, by driving the gate to a high level and detecting the drain current, which depends on the threshold voltage. Erasing must be done carefully, because if a memory cell is overerased, its threshold voltage may become negative, causing the cell to operate incorrectly.

Section 6.7.6, Erase Flowcharts and Sample Programs, shows optimal erase control flowcharts and sample programs.

Table 6.6 Principle of Memory Cell Operation

	Program	Erase	Read
Memory cell			
Memory array			

6.4.2 Mode Pin Settings and ROM Space

The H8/3644F has 32 kbytes of on-chip flash memory, the H8/3643F has 24 kbytes, and the H8/3642AF has 16 kbytes. The ROM is connected to the CPU by a 16-bit data bus. The CPU accesses flash memory in two states for both byte-size and word-size instructions.

The flash memory is allocated to addresses H'0000 to H'7FFF in the H8/3644F, to addresses H'0000 to H'5FFF in the H8/3643F, and to addresses H'0000 to H'3FFF in the H8/3642AF.

6.4.3 Features

The features of the flash memory are summarized below.

- Five flash memory operating modes

  There are five flash memory operating modes: program mode, program-verify mode, erase mode, erase-verify mode, and prewrite-verify mode.
- Erase block specification

  Blocks to be erased in the flash memory space can be specified by setting the corresponding register bits. The address space includes a large block area (four blocks with sizes from 4 kbytes to 8 kbytes) and a small block area (eight blocks with sizes from 128 bytes to 1 kbyte).
- Programming/erase times

  The flash memory programming time is 50 μs (typ.) per byte, and the erase time is 1 s (typ.).
- Erase-program cycles

  Flash memory contents can be erased and reprogrammed up to 100 times.
- On-board programming modes

  There are two modes in which flash memory can be programmed, erased, and verified on-board: boot mode and user program mode.
- Automatic bit rate adjustment

  For data transfer in boot mode, the chip's bit rate can be automatically adjusted to match the transfer bit rate of the host (max. 9600 bps).
- PROM mode

  Flash memory can be programmed and erased in PROM mode, using a general-purpose PROM programmer, as well as in on-board programming mode. The specifications for programming, erasing, verifying, etc., are the same as for standard HN28F101 flash memory.

6.4.4 Block Diagram

Figure 6.7 shows a block diagram of the flash memory.

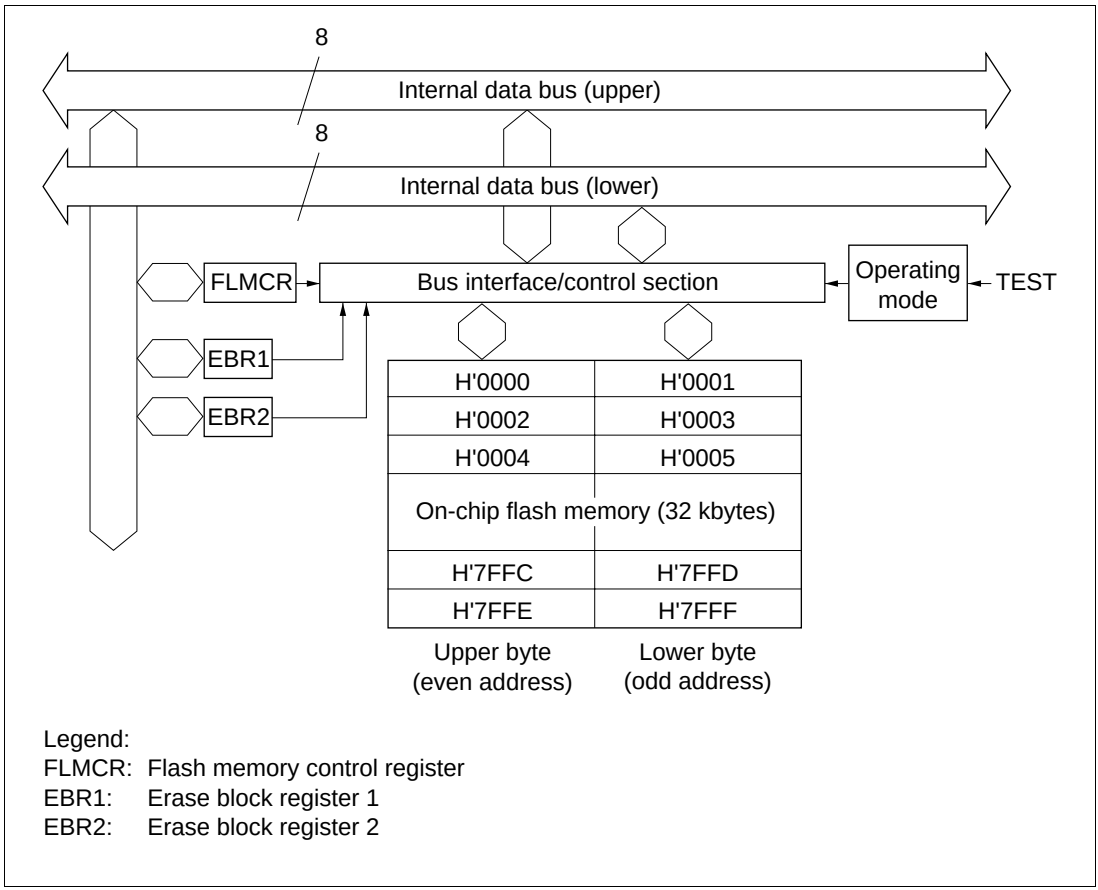


Figure 6.7 Block Diagram of Flash Memory (Example of the H8/3644F)

6.4.5 Pin Configuration

The flash memory is controlled by means of the pins shown in table 6.7.

Table 6.7 Flash Memory Pins

Pin Name	Abbreviation	Input/Output	Function
Programming power	FV_{PP}	Power supply	Apply 12.0 V
Mode pin	TEST	Input	Sets H8/3644F operating mode
Transmit data	TXD	Output	SCI3 transmit data output
Receive data	RXD	Input	SCI3 receive data input

The transmit data pin and receive data pin are used in boot mode.

6.4.6 Register Configuration

The registers used to control the on-chip flash memory are shown in table 6.8.

Table 6.8 Flash Memory Registers

Register Name	Abbreviation	R/W	Initial Value	Address
Flash memory control register	FLMCR	R/W	H'00	H'FF80
Erase block register 1	EBR1	R/W	H'F0	H'FF82
Erase block register 2	EBR2	R/W	H'00	H'FF83

The FLMCR, EBR1, and EBR2 registers are valid only when programming and erasing flash memory, and can only be accessed when 12 V is applied to the FV_{PP} pin. When 12 V is not applied to the FV_{PP} pin, addresses H'FF80 to H'FF83 cannot be modified and are always read as H'FF.

6.5 Flash Memory Register Descriptions

6.5.1 Flash Memory Control Register (FLMCR)

FLMCR is an 8-bit register used for flash memory operating mode control. Transitions to program mode, erase mode, program-verify mode, and erase-verify mode are made by setting bits in this register. FLMCR is initialized to H'00 upon reset, in sleep mode, subsleep mode, watch mode, and standby mode, and when 12 V is not applied to FV_{PP}. When 12 V is applied to FV_{PP}, a reset initializes FLMCR to H'80.

Bit	7	6	5	4	3	2	1	0
	V_{PP}	—	—	—	EV	PV	E	P
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	—	—	—	R/W*	R/W*	R/W*	R/W*

Note: * For information on access to this register, see note 11 in section 6.9, Flash Memory Programming and Erasing Precautions.

Bit 7—Programming Power (V_{PP}): Bit 7 is a status flag that indicates that 12 V is applied to the FV_{PP} pin. For further information, see note 5 in section 6.9, Flash Memory Programming and Erasing Precautions.

Bit 7: V_{PP}	Description	
0	Clearing conditions: When 12 V is not applied to the FV_{PP} pin	(initial value)
1	Setting conditions: When 12 V is applied to the FV_{PP} pin	

Bit 3—Erase-Verify Mode (EV)*[1]: Bit 3 selects transition to or exit from erase-verify mode.

Bit 3: EV	Description	
0	Exit from erase-verify mode	(initial value)
1	Transition to erase-verify mode	

Bit 2—Program-Verify Mode (PV)*[1]: Bit 2 selects transition to or exit from program-verify mode.

Bit 2: PV	Description	
0	Exit from program-verify mode	(initial value)
1	Transition to program-verify mode	

Bit 1—Erase Mode (E)*[1] *[2]: Bit 1 selects transition to or exit from erase mode.

Bit 1: E	Description	
0	Exit from erase mode	(initial value)
1	Transition to erase mode	

Bit 0—Program Mode (P)*[1] *[2]: Bit 0 selects transition to or exit from program mode.

Bit 0: P	Description	
0	Exit from program mode	(initial value)
1	Transition to program mode	

Notes: 1. Do not set multiple bits simultaneously.

Do not release or cut the V_{CC} or V_{PP} power supply while a bit is set.

2. P bit and E bit setting should be carried out in accordance with the program/erase algorithms shown in section 6.7, Programming and Erasing Flash Memory.

A watchdog timer setting should be made beforehand to prevent the P or E bit from being set for longer than the specified time.

See section 6.9, Flash Memory Programming and Erasing Precautions, for more information on the use of these bits.

6.5.2 Erase Block Register 1 (EBR1)

EBR1 is an 8-bit register that specifies large flash-memory blocks for programming or erasure. EBR1 is initialized to H'F0 upon reset, in sleep mode, subsleep mode, watch mode, and standby mode, and when 12 V is not applied to FV_{PP}. When a bit in EBR1 is set to 1, the corresponding block is selected and can be programmed and erased. The erase block map is shown in figure 6.8, and the correspondence between bits and erase blocks is shown in table 6.9.

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	LB3	LB2	LB1	LB0
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W*	R/W*	R/W*	R/W*

Note: * Word access cannot be used on this register; byte access must be used. For information on access to this register, see note 11 in section 6.9, Flash Memory Programming and Erasing Precautions. LB3 is invalid in the H8/3643F, and LB3 and LB2 are invalid in the H8/3642AF.

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

Bits 3 to 0—Large Block 3 to 0 (LB3 to LB0): These bits select large blocks (LB3 to LB0) to be programmed and erased.

Bits 3 to 0: LB3 to LB0	Description	
0	Block LB3 to LB0 is not selected	(initial value)
1	Block LB3 to LB0 is selected	

6.5.3 Erase Block Register 2 (EBR2)

EBR2 is an 8-bit register that specifies small flash-memory blocks for programming or erasure. EBR2 is initialized to H'00 upon reset, in sleep mode, subsleep mode, watch mode, and standby mode, and when 12 V is not applied to FV_{pp}. When a bit in EBR2 is set to 1, the corresponding block is selected and can be programmed and erased. The erase block map is shown in figure 6.8, and the correspondence between bits and erase blocks is shown in table 6.9.

Bit	7	6	5	4	3	2	1	0
	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*

Note: * Word access cannot be used on this register; byte access must be used. For information on access to this register, see note 11 in section 6.9, Flash Memory Programming and Erasing Precautions. LB3 is invalid in the H8/3643F, and LB3 and LB2 are invalid in the H8/3642AF.

Bits 7 to 0—Small Block 7 to 0 (SB7 to SB0): These bits select small blocks (SB7 to SB0) to be programmed and erased.

Bits 7 to 0: SB7 to SB0	Description	
0	Block SB7 to SB0 is not selected	(initial value)
1	Block SB7 to SB0 is selected	

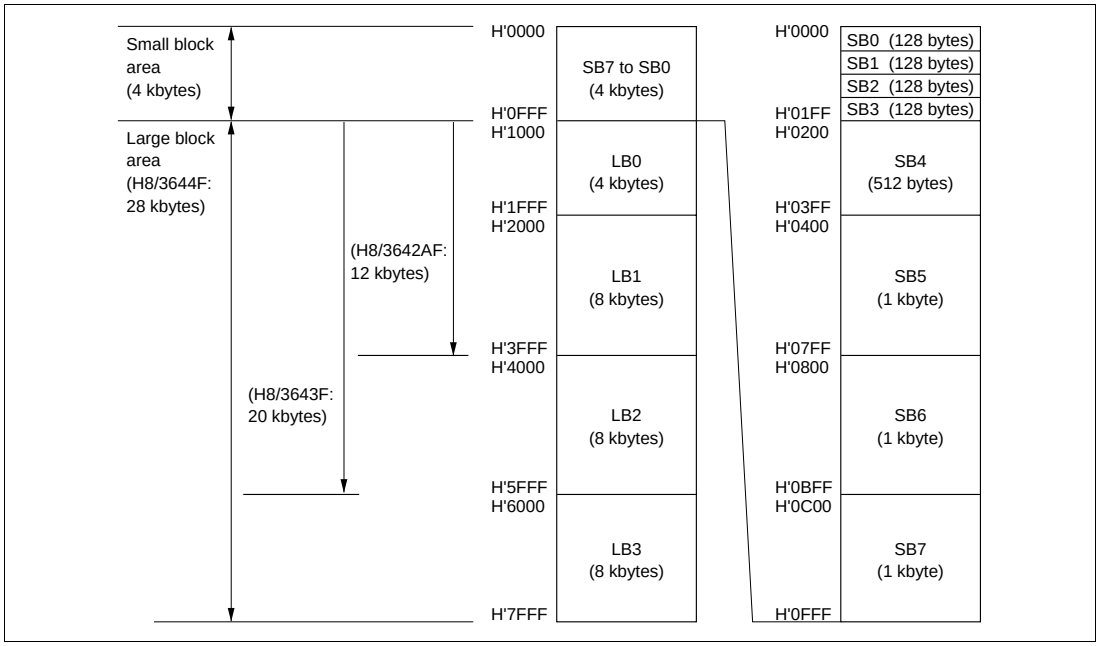


Figure 6.8 Erase Block Map

Table 6.9 Correspondence between Erase Blocks and EBR1/EBR2 Bits

Register	Bit	Block	Addresses	Size
EBR1	0	LB0	H'1000 to H'1FFF	4 kbytes
	1	LB1	H'2000 to H'3FFF	8 kbytes
	2	LB2	H'4000 to H'5FFF	8 kbytes
	3	LB3	H'6000 to H'7FFF	8 kbytes

Register	Bit	Block	Addresses	Size
EBR2	0	SB0	H'0000 to H'007F	128 bytes
	1	SB1	H'0080 to H'00FF	128 bytes
	2	SB2	H'0100 to H'017F	128 bytes
	3	SB3	H'0180 to H'01FF	128 bytes
	4	SB4	H'0200 to H'03FF	512 bytes
	5	SB5	H'0400 to H'07FF	1 kbyte
	6	SB6	H'0800 to H'0BFF	1 kbyte
	7	SB7	H'0C00 to H'0FFF	1 kbyte

6.6 On-Board Programming Modes

When an on-board programming mode is selected, on-chip flash memory programming, erasing, and verifying can be carried out. There are two on-board programming modes—boot mode and user program mode—set by the mode pin (TEST) and the FV_{PP} pin. Table 6.10 shows how to select the on-board programming modes. For information on turning V_{PP} on and off, see note 5 in section 6.9, Flash Memory Programming and Erasing Precautions.

Table 6.10 On-Board Programming Mode Selection

Mode Setting	FV_{PP}	TEST	Notes
Boot mode	12 V*	12 V*	
User program mode		V_{SS}	

Note: * See notes 6 to 8 in section 6.6.1, Notes on Use of Boot Mode, for the timing of 12 V application.

6.6.1 Boot Mode

When boot mode is used, a user program for flash memory programming and erasing must be prepared beforehand in the host machine (which may be a personal computer). SCI3 is used in asynchronous mode (see figure 6.8). When the H8/3644F, H8/3643F, or H8/3642AF is set to boot mode, after reset release a built-in boot program is activated, the low period of the data sent from the host is first measured, and the bit rate register (BRR) value determined. The chip's on-chip serial communication interface (SCI3) can then be used to download the user program from the host machine. The downloaded user program is written into RAM.

After the program has been stored, execution branches to the start address (H'FBE0) of the on-chip RAM, the program stored in RAM is executed, and flash memory programming/erasing can be carried out. Figure 6.10 shows the boot mode execution procedure.

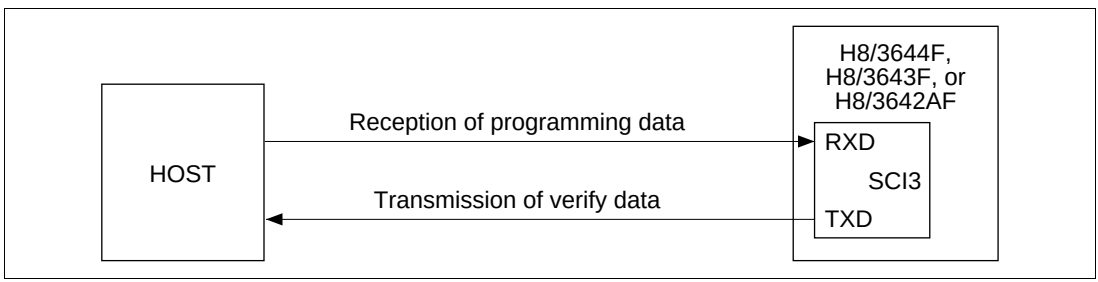
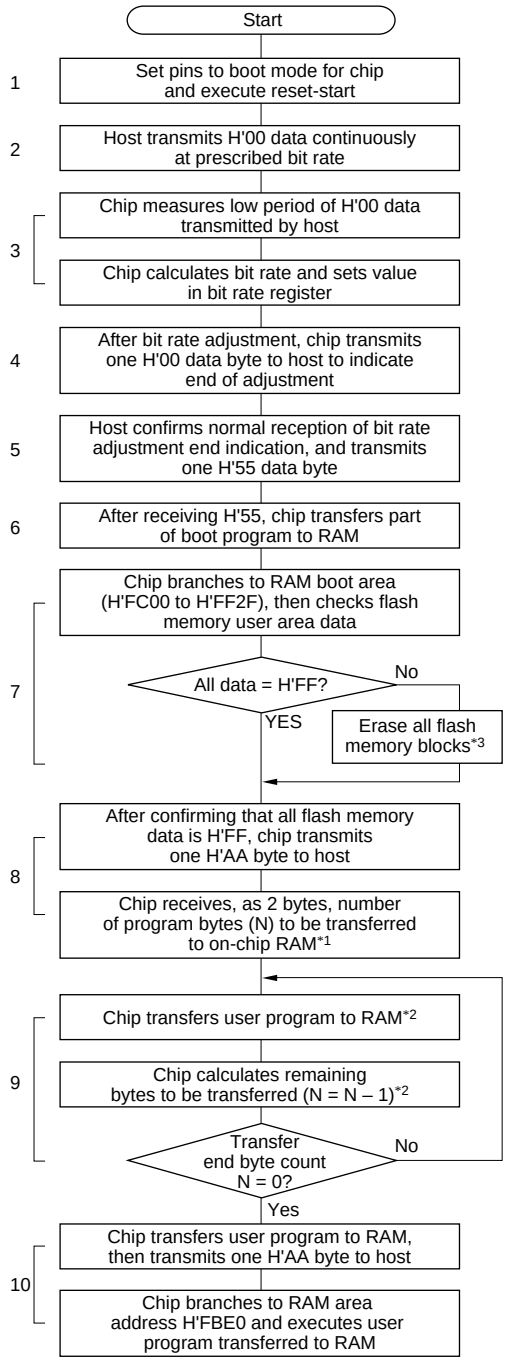


Figure 6.9 Boot Mode System Configuration

Boot Mode Execution Procedure: The boot mode execution procedure is shown below.



1. Set the chip to boot mode and execute a reset-start.
2. Set the host to the prescribed bit rate (2400/4800/9600) and have it transmit H'00 data continuously using a transfer data format of 8-bit data plus 1 stop bit.
3. The chip repeatedly measures the low period at the RXD pin and calculates the asynchronous communication bit rate used by the host.
4. After SCI3 bit rate adjustment is completed, the chip transmits one H'00 data byte to indicate the end of adjustment.
5. On receiving the one-byte data indicating completion of bit rate adjustment, the host should confirm normal reception of this indication and transmit one H'55 data byte.
6. After receiving H'55, the chip transfers part of the boot program to RAM areas H'FB80 to H'FBDF and H'FC00 to H'FF2F.
7. The chip branches to the RAM boot program area (H'FC00–H'FF2F) and checks for the presence of data written in the flash memory. If data has been written in the flash memory, the chip erases all blocks.
8. The chip transmits one H'AA byte. The host then transmits the number of user program bytes to be transferred to the chip. The number of bytes should be sent as two bytes, upper byte followed by lower byte. The host should then transmit sequentially the program set by the user.
   The chip transmits the received byte count and user program sequentially to the host, one byte at a time, as verify data (echo-back).
9. The chip writes the received user program sequentially to on-chip RAM area H'FBE0 to H'FF6D (910 bytes).
10. The chip transmits one H'AA byte, then branches to on -chip RAM address H'FBE0 and executes the user program written in area H'FBE0 to H'FF6D.

Notes:
1. The size of the RAM area available to the user is 910 bytes. The number of bytes to be transferred must not exceed 910 bytes. The transfer byte count must be sent as two bytes, upper byte followed by lower byte.
   Example of transfer byte count: for 256 bytes (H'0100), upper byte = H'01, lower byte = H'00
2. The part of the user program that controls the flash memory should be set in the program in accordance with the flash memory program/ erase algorithms described later in this section.
3. If a memory cell does not operate normally and cannot be erased, the chip transmits one H'FF byte as an erase error indication and halts the erase operation and subsequent operations.

Figure 6.10 Boot Mode Operation Flowchart

Automatic SCI Bit Rate Adjustment: When boot mode is initiated, the H8/3644F, H8/3643F, or H8/3642AF measures the low period of the asynchronous SCI communication data transmitted continuously from the host (figure 6.11). The data format should be set as 8-bit data, 1 stop bit, no parity. The chip calculates the bit rate of the transmission from the host from the measured low period (9 bits), and transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication has been received normally, and transmit one H'55 byte to the chip. If reception cannot be performed normally, initiate boot mode again (reset), and repeat the above operations. Depending on the host's transmission bit rate and the chip's system clock oscillation frequency (f_{OSC}), there will be a discrepancy between the bit rates of the host and the chip. To insure correct SCI operation, the host's transfer bit rate should be set to 2400, 4800, or 9600 bps*[1]. Table 6.11 shows typical host transfer bit rates and system clock oscillation frequency for which automatic adjustment of the chip's bit rate is possible. Boot mode should be used within this system clock oscillation frequency range*[2].

Notes: 1. Only use a host bit rate setting of 2400, 4800, or 9600 bps. No other bit rate setting should be used.

2. Although the chip may also perform automatic bit rate adjustment with bit rate and system clock oscillation frequency combinations other than those shown in table 6.11, a degree of error will arise between the bit rates of the host and the chip, and subsequent transfer will not be performed normally. Therefore, only a combination of bit rate and system clock oscillation frequency within one of the ranges shown in table 6.11 can be used for boot mode execution.

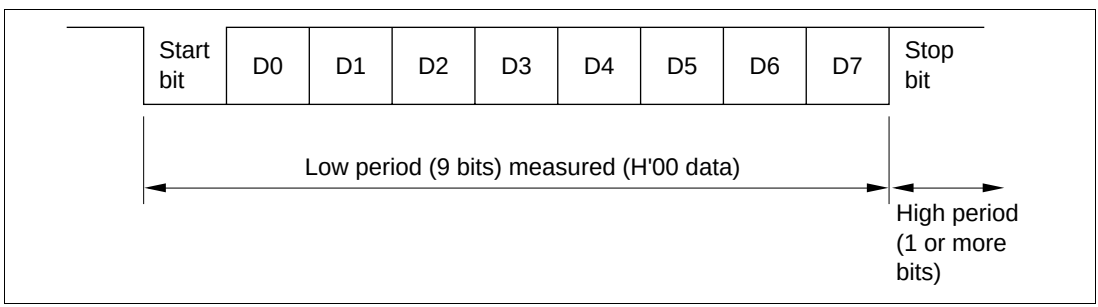


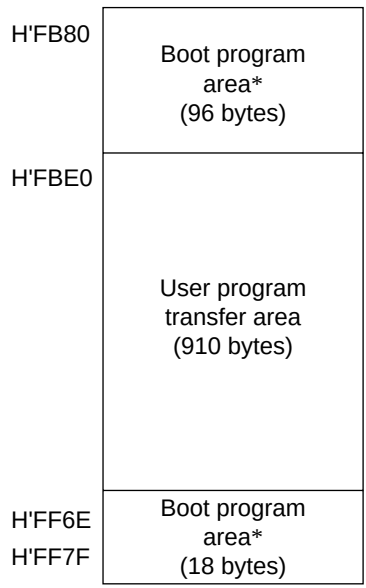
Figure 6.11 Measurement of Low Period in Transmit Data from Host

Table 6.11 System Clock Oscillation Frequencies Permitting Automatic Adjustment of Chip (H8/3644F, H8/3643F, H8/3642AF) Bit Rate

Host Bit Rate*	System Clock Oscillation Frequencies (f_{osc}) Permitting Automatic Adjustment of Chip (H8/3644F, H8/3643F, H8/3642AF) Bit Rate
9600 bps	8 MHz to 16 MHz
4800 bps	4 MHz to 16 MHz
2400 bps	2 MHz to 16 MHz

Note: * Use a host bit rate setting of 2400, 4800, or 9600 bps only. No other setting should be used.

RAM Area Allocation in Boot Mode: In boot mode, the 96-byte area from H'FB80 to H'FBDF and the 18-byte area from H'FF6E to H'FF7F are reserved for boot program use, as shown in figure 6.12. The area to which the user program is transferred is H'FBE0 to H'FF6D (910 bytes). The boot program area becomes available when a transition is made to the execution state for the user program transferred to RAM. A stack area should be set within the user program as required.



Note: * These areas cannot be used until a transition is made to the execution state for the user program transferred to RAM (i.e. a branch is made to RAM address H'FBE0). Note also that the boot program remains in the boot program area in RAM (H'FB80 to H'FBDF, H'FF6E to H'FF7F) even after control branches to the user program. When an interrupt handling routine is executed in the boot program, the 16 bytes from H'FB80 to H'FB8F in this area cannot be used. For details see section 6.7.9, Interrupt Handling during Flash Memory Programming/Erasing.

Figure 6.12 RAM Areas in Boot Mode

Notes on Use of Boot Mode:

1. When the chip (H8/3644F, H8/3643F, or H8/3642AF) comes out of reset in boot mode, it measures the low period of the input at the SCI3's RXD pin. The reset should end with RXD high. After the reset ends, it takes about 100 states for the chip to get ready to measure the low period of the RXD input.

2. In boot mode, if any data has been programmed into the flash memory (if all data is not H'FF), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, such as the first time on-board programming is performed, or if the program activated in user program mode is accidentally erased.

3. Interrupts cannot be used while the flash memory is being programmed or erased.

4. The RXD and TXD lines should be pulled up on the board.

5. Before branching to the user program (RAM address H'FBE0), the chip terminates transmit and receive operations by its on-chip SCI3 (by clearing the RE and TE bits to 0 in the serial control register (SCR)), but the adjusted bit rate value remains set in the bit rate register (BRR). The transmit data output pin, TXD, goes to the high-level output state ($PCR2_2$ = 1 in the port 2 control register, $P2_2$ = 1 in the port 2 data register).

   The contents of the CPU's internal general registers are undefined at this time, so these registers must be initialized immediately after branching to the user program. In particular, since the stack pointer (SP) is used implicitly in subroutine calls, etc., a stack area must be specified for use by the user program.

   The initial values of other on-chip registers are not changed.

6. Boot mode can be entered by applying 12 V to the TEST pin and FV_{PP} pin in accordance with the mode setting conditions shown in table 6.10, and then executing a reset-start. Care must be taken with turn-on of the V_{PP} power supply at this time.

   On reset release (a low-to-high transition), the chip determines whether 12 V is being applied to the TEST pin and FV_{PP} pin, and on detecting that boot mode has been set, retains that state internally. As the applied voltage criterion level (threshold level) at this time is the range of approximately V_{CC} +2 V to 11.4 V, a transition will be made to boot mode even if a voltage sufficient for executing programming and erasing (11.4 V to 12.6 V) is not being applied. Therefore, when executing the boot program, the V_{PP} power supply must be stabilized within the range of 11.4 V to 12.6 V before a branch is made to the RAM area, as shown in figure 6.24.

   Insure that the program voltage V_{PP} does not exceed 12.6 V when a transition is made to boot mode (when reset is released), and does not exceed the range 12 V ±0.6 V during boot mode operation. If these values are exceeded, boot mode execution will not be performed correctly.

Also, do not release or cut V_{PP} during boot mode execution or when programming or erasing flash memory*.

Boot mode can be exited by driving the reset pin low, then releasing 12 V application to the TEST pin and FV_{PP} pin at least 10 system clock cycles later, and setting the TEST pin to V_{SS} to release the reset.

However, external pin settings must not be changed during boot mode execution.

Note that the boot mode state is not maintained if 12 V application to the TEST pin is released while in boot mode.

Also, if a watchdog timer reset occurs in this boot mode state, the built-in boot program will be restarted without clearing the MCU's internal mode state.

7. If the TEST pin input level is changed (e.g. from 0 V to 5 V to 12 V) during a reset (while a low level is being input at the $\overline{RES}$ pin), port states will change as a result of the change of MCU operating mode. Therefore, care must be taken to make pin settings to prevent these pins from becoming output signal pins during a reset, and to prevent collision with signals outside the MCU.

8. Regarding 12 V application to the FV_{PP} and TEST pins, insure that peak overshoot does not exceed the maximum rating of 13 V.

   Also, be sure to connect bypass capacitors to the FV_{PP} and TEST pins.

Note: * For further information on V_{PP} application, release, and cut-off, see note 5 in section 6.9, Flash Memory Programming and Erasing Precautions.

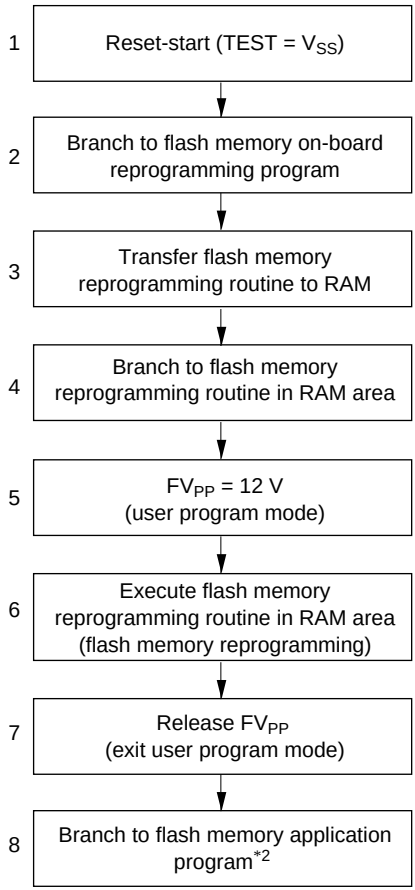
6.6.2 User Program Mode

When set to user program mode, the H8/3644F, H8/3643F, or H8/3642AF can program and erase its flash memory by executing a user program. Therefore, on-board reprogramming of the on-chip flash memory can be carried out by providing on-board means of supplying V_{PP} and programming data, and storing an on-board reprogramming program in part of the program area.

User program mode is selected by applying 12 V to the FV_{PP} pin when flash memory is not being accessed, during a reset or after confirming that a reset has been performed properly (after the reset is released).

The flash memory cannot be read while being programmed or erased, so the on-board reprogramming program or flash memory reprogramming routine should be transferred to the RAM area, and on-board reprogramming executed in that area.

User Program Mode Execution Procedure*[1]: The procedure for user program execution in RAM is shown below.



Procedure:
An on-board reprogramming program must be written into flash memory by the user beforehand.

1. Set the TEST pin to V_{SS} and execute a reset-start.
2. Branch to the on-board reprogramming program written to flash memory.
3. Transfer the flash memory reprogramming routine to the RAM area.
4. Branch to the flash memory reprogramming routine transferred to the RAM area.
5. Apply 12 V to the FV_{PP} pin. (Transition to user program mode)
6. Execute the flash memory reprogramming routine in the RAM area, an perform on-board reprogramming of the flash memory.
7. Switch the FV_{PP} pin from 12 V to V_{CC}, and exit user program mode.
8. After on-board reprogramming of the flash memory ends, branch to the flash memory application program.

Notes: 1. Do not apply 12 V to the FV_{PP} pin during normal operation. To prevent inadvertent programming or erasing due to program runaway, etc., apply 12 V to the FV_{PP} pin only when the flash memory is being programmed or erased . Memory cells may not operate normally if overprogrammed or overerased due to program runaway, etc. Also, while 12 V is applied to the FV_{PP} pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc. For further information on FV_{PP} application, release, and cut-off, see note 5 in section 6.9, Flash Memory Programming and Erasing Precautions.

2. When the application of 12 V to the FV_{PP} pin is released after programming is completed, the flash memory read setup time (t_{FRS}) must elapse before executing a program in flash memory. This specifies the setup time from the point at which the FV_{PP} voltage reaches the V_{CC} + 2 V level after 12 V application is released until the flash memory is read.

Figure 6.13 Example of User Program Mode Operation

6.7 Programming and Erasing Flash Memory

The on-chip flash memory of the H8/3644F, H8/3643F, and H8/3642AF is programmed and erased by software, using the CPU. There are five flash memory operating modes: program mode, erase mode, program-verify mode, erase-verify mode, and prewrite-verify mode. Transitions to these modes can be made by setting the P, E, PV, and EV bits in the flash memory control register (FLMCR).

The flash memory cannot be read while being programmed or erased. Therefore, the program that controls flash memory programming and erasing should be located and executed in on-chip RAM or external memory. A description of each mode is given below, with recommended flowcharts and sample programs for programming and erasing.

See section 6.9, Flash Memory Programming and Erasing Precautions, for additional notes on programming and erasing.

6.7.1 Program Mode

To write data into the flash memory, follow the programming algorithm shown in figure 6.14. This programming algorithm enables data to be written without subjecting the device to voltage stress or impairing the reliability of the programmed data.

To write data, first set the blocks to be programmed with erase block registers 1 and 2 (EBR1, EBR2), and write the data to the address to be programmed, as in writing to RAM. The flash memory latches the programming address and programming data in an address latch and data latch. Next set the P bit in FLMCR, selecting program mode. The programming time is the time during which the P bit is set. Make a setting so that the total programming time does not exceed 1 ms. Programming for too long a time, due to program runaway for example, can damage the device. Before selecting program mode, set up the watchdog timer so as to prevent overprogramming.

For details of the programming procedure, see section 6.7.3, Programming Flowchart and Sample Program.

6.7.2 Program-Verify Mode

In program-verify mode, the data written in program mode is read to check whether it has been correctly written in the flash memory.

After the elapse of the programming time, exit programming mode (clear the P bit to 0) and select program-verify mode (set the PV bit to 1). In program-verify mode, a program-verify voltage is applied to the memory cells at the latched address. If the flash memory is read in this state, the data at the latched address will be read. After selecting program-verify mode, wait at least 4 µs before reading, then compare the programmed data with the verify data. If they agree, exit program-verify mode and program the next address. If they do not agree, select program mode again and repeat the same program and program-verify sequence. Do not repeat the program and program-verify sequence more than six times* for the same bit.

Note: * When a bit is programmed repeatedly, set a loop counter so that the total programming time will not exceed 1 ms.

6.7.3 Programming Flowchart and Sample Program

Flowchart for Programming One Byte

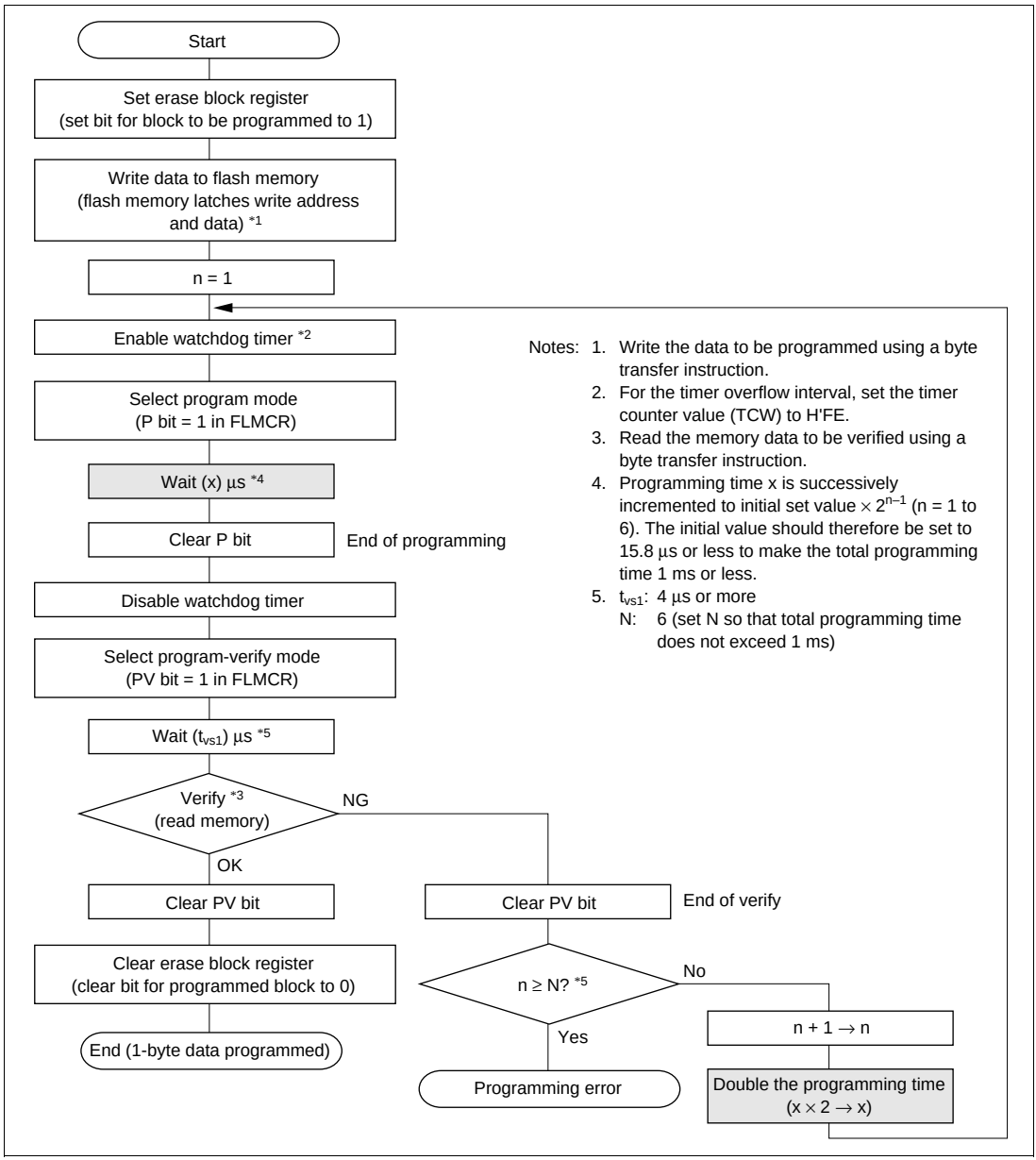


Figure 6.14 Programming Flowchart

Sample Program for Programming One Byte

This program uses the following registers:

R0H: Used for erase block specification.
R1H: Stores programming data.
R1L: Stores read data.
R3: Stores the programming address. Valid address specifications are H'0000 to H'EF7F.
R4: Used for program and program-verify loop counter value setting. Also stores register set values.
R5: Used for program loop counter value setting.
R6L: Used for the program-verify fail count.

Arbitrary data can be programmed at an arbitrary address by setting the R3 (programming address) and R1H (programming data) values.

The values of #a and #b depend on the operating frequency. They should be set as indicated in table 6.12.

```
FLMCR:   .EQU   H'FF80
EBR1:    .EQU   H'FF82
EBR2:    .EQU   H'FF83
TCSRW:   .EQU   H'FFBE
TCW:     .EQU   H'FFBF
         .ALIGN 2
PRGM:    MOV.B  #H'**,     R0H       ;
         MOV.B  R0H,       @EBR*:8   ; Set EBR*

         MOV.B  #H'00,     R6L       ; Program-verify fail count
         MOV.W  #H'a,      R5        ; Set program loop counter
         MOV.B  R1H,       @R3       ; Dummy write
PRGMS:   INC    R6L                 ; Program-verify fail counter + 1 → R6L
         MOV.W  #H'FE5A,   R4        ;
         MOV.B  R4L,       @TCSRW:8  ;
         MOV.B  R4H,       @TCW:8    ;
         MOV.B  #H'36,     R4L       ;
         MOV.B  R4L,       @TCSRW:8  ; Start watchdog timer
         MOV.W  R5,        R4        ; Set program loop counter
         BSET   #0,        @FLMCR:8  ; Set P bit
LOOP1:   SUBS   #1,        R4        ;
         MOV.W  R4,        R4        ;
         BNE    LOOP1                ; Wait loop
         BCLR   #0,        @FLMCR:8  ; Clear P bit
         MOV.B  #H'50,     R4L       ;
         MOV.B  R4L,       @TCSRW:8  ; Stop watchdog timer

         MOV.B  #H'b,      R4H       ; Set program-verify fail counter
         BSET   #2,        @FLMCR:8  ; Set PV bit
```

```
LOOP2:  DEC     R4H                 ;
        BNE     LOOP2               ; Wait loop
        MOV.B   @R3,        R1L     ; Read programmed data
        CMP.B   R1H,        R1L     ; Compare programmed data with read data
        BEQ     PVOK                ; Program-verify decision
        BCLR    #2,         @FLMCR:8 ; Clear PV bit

        CMP.B   #H'06,      R6L     ; Program-verify executed 6 times?
        BEQ     NGEND               ; If program-verify executed 6 times, branch to NGEND
        ADD.W   R5,         R5      ; Double programming time
        BRA     PRGMS               ; Program again

PVOK:   BCLR    #2,         @FLMCR:8 ; Clear PV bit
        MOV.B   #H'00,      R6L     ;
        MOV.B   R6L,        @EBR*:8 ; Clear EBR*

        One byte programmed

NGEND:  Programming error
```

6.7.4 Erase Mode

To erase the flash memory, follow the erasing algorithm shown in figure 6.15. This erasing algorithm enables data to be erased without subjecting the device to voltage stress or impairing the reliability of the programmed data.

To erase flash memory, before starting to erase, first place all memory data in all blocks to be erased in the programmed state (program all memory data to H'00). If all memory data is not in the programmed state, follow the sequence described later to program the memory data to zero. Select the flash memory areas to be programmed with erase block registers 1 and 2 (EBR1, EBR2). Next set the E bit in FLMCR, selecting erase mode. The erase time is the time during which the E bit is set. To prevent overerasing, use a software timer to divide the time for one erasure, and insure that the total time does not exceed 30 s. See section 6.7.6, Erase Flowcharts and Sample Programs, for the time for one erasure. Overerasing, due to program runaway for example, can give memory cells a negative threshold voltage and cause them to operate incorrectly. Before selecting erase mode, set up the watchdog timer so as to prevent overerasing.

6.7.5 Erase-Verify Mode

In erase-verify mode, after data has been erased, it is read to check that it has been erased correctly. After the erase time has elapsed, exit erase mode (clear the E bit to 0), and select erase-verify mode (set the EV bit to 1). Before reading data in erase-verify mode, write H'FF dummy data to the address to be read. This dummy write applies an erase-verify voltage to the memory cells at the latched address. If the flash memory is read in this state, the data at the latched address will be read. After the dummy write, wait at least 2 µs before reading. Also, wait at least 4 µs before performing the first dummy write after selecting erase-verify mode. If the read data has been successfully erased, perform the erase-verify sequence (dummy write, wait of at least 2 µs, read) on the next address. If the read data has not been erased, select erase mode again and repeat the same erase and erase-verify sequence through the last address, until all memory data has been erased to 1. Do not repeat the erase and erase-verify sequence more than 602 times, however.

6.7.6 Erase Flowcharts and Sample Programs

Flowchart for Erasing One Block

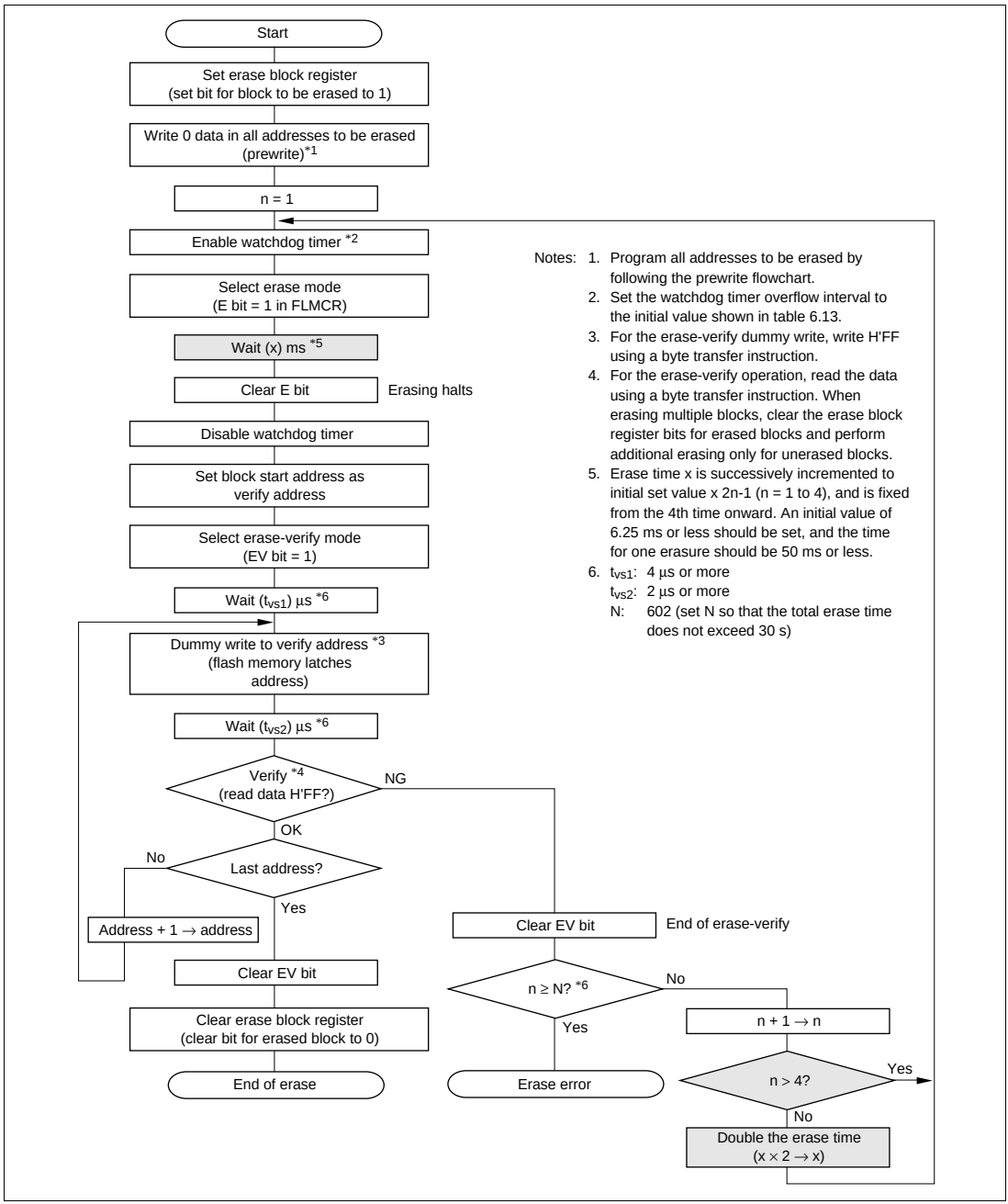


Figure 6.15 Erase Flowchart

Prewrite Flowchart

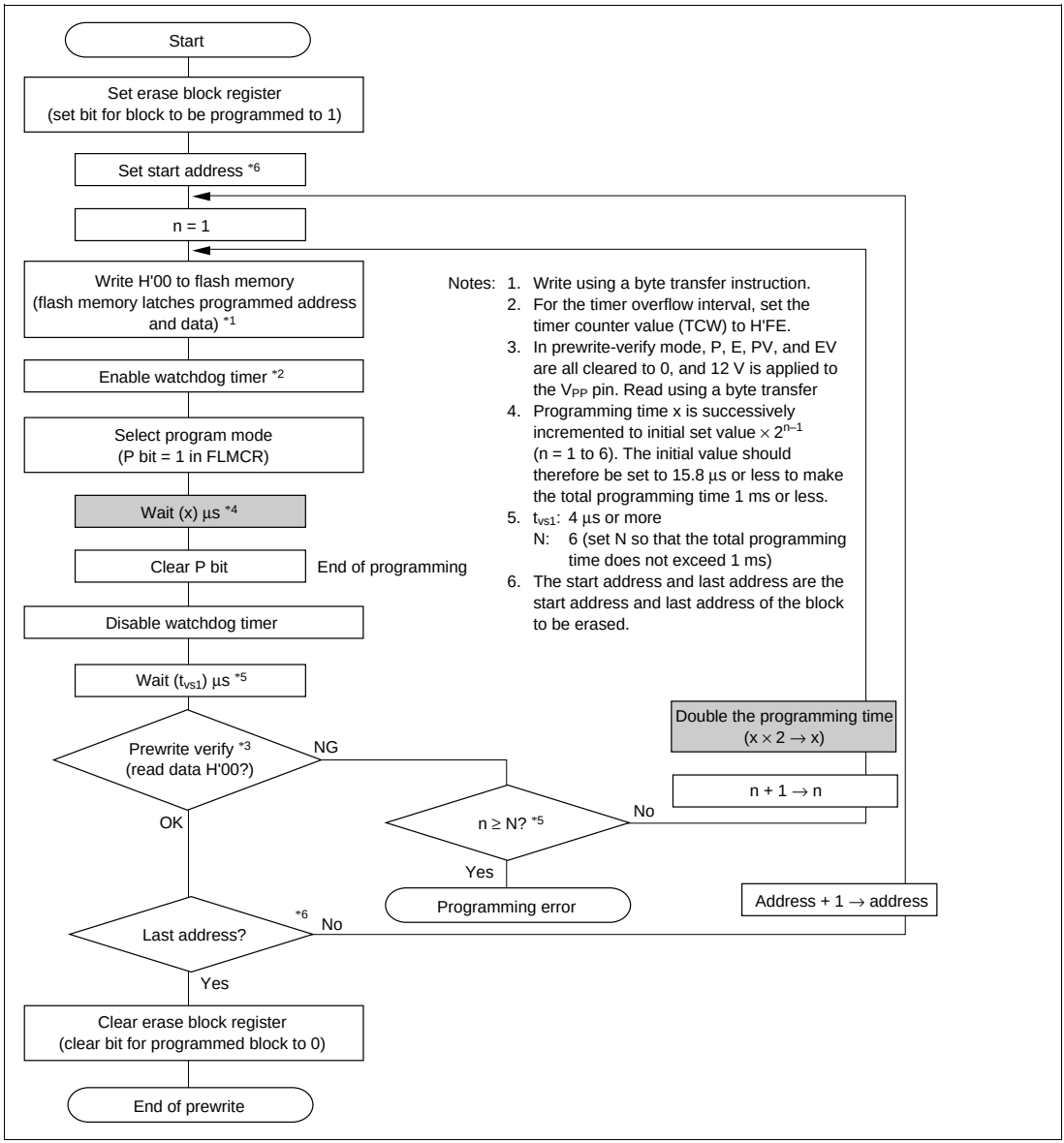


Figure 6.16 Prewrite Flowchart

Sample Program for Erasing One Block

This program uses the following registers:

R0: Used for erase block specification. Also stores address used in prewrite and erase-verify.
R1H: Stores read data. Also used in dummy write.
R2: Stores last address of block to be erased.
R3: Stores address used in prewrite and erase-verify.
R4: Used for prewrite, prewrite-verify, erase, and erase-verify loop counter value setting. Also stores register set values.
R5: Used for prewrite and erase loop counter value setting.
R6L: Used for prewrite-verify and erase-verify fail count.

The values of #a, #b, #c, #d, and #e in the program depend on the operating frequency. They should be set as indicated in tables 6.12 and 6.13. Erase block register (EBR1, EBR2) settings should be made as indicated in sections 6.5.2 and 6.5.3 in section 6.5, Flash Memory Register Descriptions. For #BLKSTR and #BLKEND, the start address and end address corresponding to the set erase block register should be set as indicated in table 6.8.

```
FLMCR:    .EQU    H'FF80
EBR1:     .EQU    H'FF82
EBR2:     .EQU    H'FF83
TCSRW:    .EQU    H'FFBE
TCW:      .EQU    H'FFBF

          .ALIGN            2
          MOV.B   #H'**,    R0H       ;
          MOV.B   R0H,      @EBR*:8   ; Set EBR*

; #BLKSTR is start address of block to be erased
; #BLKEND is last address of block to be erased
          MOV.W   #BLKSTR,  R0        ; Start address of block to be erased
          MOV.W   #BLKEND,  R2        ; Last address of block to be erased
          ADDS    #1,       R2        ; Last address of block to be erased + 1 → R2


; Execute prewrite
          MOV.W   R0,       R3        ; Start address of block to be erased
PREWRT:   MOV.B   #H'00,    R6L       ; Prewrite verify fail counter
          MOV.W   #H'a,     R5        ; Set prewrite loop counter
PREWRS:   INC     R6L                 ; Prewrite-vector fail counter + 1 → R6L
          MOV.B   #H'00,    R1H       ;
          MOV.B   R1H,      @R3       ; Write H'00
          MOV.W   #H'FE5A,  R4        ;
          MOV.B   R4L,      @TCSRW:8  ;
          MOV.B   R4H,      @TCW:8    ;
          MOV.B   #H'36,    R4L       ;
```

```
          MOV.B   R4L,        @TCSRW:8 ; Start watchdog timer
          MOV.W   R5,         R4       ; Set prewrite loop counter
          BSET    #0,         @FLMCR:8 ; Set P bit
LOOPR1:   SUBS    #1,         R4       ;
          MOV.W   R4,         R4       ;
          BNE     LOOPR1               ; Wait loop
          BCLR    #0,         @FLMCR:8 ; Clear P bit
          MOV.B   #H'50,      R4L      ;
          MOV.B   R4L,        @TCSRW:8 ; Stop watchdog timer
          MOV.B   #H'c,       R4H      ; Set prewrite-verify fail counter
LOOPR2:   DEC     R4H                  ;
          BNE     LOOPR2               ; Wait loop
          MOV.B   @R3,        R1H      ; Read data = H'00?
          BEQ     PWVFOK               ; If read data = H'00, branch to PWVFOK
          CMP.B   #H'06,      R6L      ; Prewrite-verify executed 6 times?
          BEQ     ABEND1               ; If prewrite-verify executed 6 times, branch to ABEND1
          ADD.W   R5,         R5       ; Double the programming time
          BRA     PREWRS               ; Prewrite again

ABEND1:   Write error

PWVFOK:   ADDS    #1,         R3       ; Address + 1 → R3
          CMP.W   R2,         R3       ; Last address?
          BNE     PREWRT               ; If not last address, prewrite next address

; Execute erase
ERASES:   MOV.W   #H'0000,    R6       ; Erase-verify fail counter
          MOV.W   #H'd,       R5       ; Set erase loop counter
ERASE:    ADDS    #1,         R6       ; Erase-verify fail counter + 1 → R6
          MOV.W   #H'e5A,     R4       ;
          MOV.B   R4L,        @TCSRW:8 ;
          MOV.B   R4H,        @TCW:8   ;
          MOV.B   #H'36,      R4L      ;
          MOV.B   R4L,        @TCSRW:8 ; Start watchdog timer
          MOV.W   R5,         R4       ; Set erase loop counter
          BSET    #1,         @FLMCR:8 ; Set E bit
LOOPE:    NOP
          NOP
          NOP
          NOP
          SUBS    #1,         R4       ;
          MOV.W   R4,         R4       ;
          BNE     LOOPE                ; Wait loop
          BCLR    #1,         @FLMCR:8 ; Clear E bit
          MOV.B   #H'50,      R4L      ;
          MOV.B   R4L,        @TCSRW:8 ; Stop watchdog timer

; Execute erase-verify
          MOV.W   R0,         R3       ; Start address of block to be erased
          MOV.B   #H'b,       R4H      ; Set erase-verify loop counter
          BSET    #3,         @FLMCR:8 ; Set EV bit
```

```
LOOPEV:  DEC     R4H                ;
         BNE     LOOPEV             ; Wait loop
EVR2:    MOV.B   #H'FF,     R1H     ;
         MOV.B   R1H,       @R3     ; Dummy write
         MOV.B   #H'c,      R4H     ; Set erase-verify loop counter
LOOPDW:  DEC     R4H                ;
         BNE     LOOPDW             ; Wait loop
         MOV.B   @R3+,      R1H     ; Read
         CMP.B   #H'FF,     R1H     ; Read data = H'FF?
         BNE     RERASE             ; If read data ≠ H'FF, branch to RERASE
         CMP.W   R2,        R3      ; Last address in block?
         BNE     EVR2               ;
         BRA     OKEND              ;

RERASE:  BCLR    #3,       @FLMCR:8 ; Clear EV bit
         SUBS    #1,       R3       ; Erase-verify address – 1 → R3
         MOV.W   #H'0004,  R4       ;
         CMP.W   R4,       R6       ; Erase-verify fail count = 4?
         BPL     BRER               ; If R6 ≥ 4. branch to BRER (branch until R6 = 4–602)
         ADD.W   R5,       R5       ; If R6 < 4, double erase time (executed only for R6 = 1, 2, 3)

BRER:    MOV.W   #H'025A,  R4       ;
         CMP.W   R4,       R6       ; Erase-verify executed 602 times?
         BNE     ERASE              ; If erase-verify not executed 602 times, erase again
         BRA     ABEND2             ; If erase-verify executed 602 times, branch to ABEND2

OKEND:   BCLR    #3,       @FLMCR:8 ; Clear EV bit
         MOV.B   #H'00,    R6L      ;
         MOV.B   R6L,      @EBR*:8  ; Clear EBR*

         One block erased

ABEND2:  Erase error
```

Flowchart for Erasing Multiple Blocks

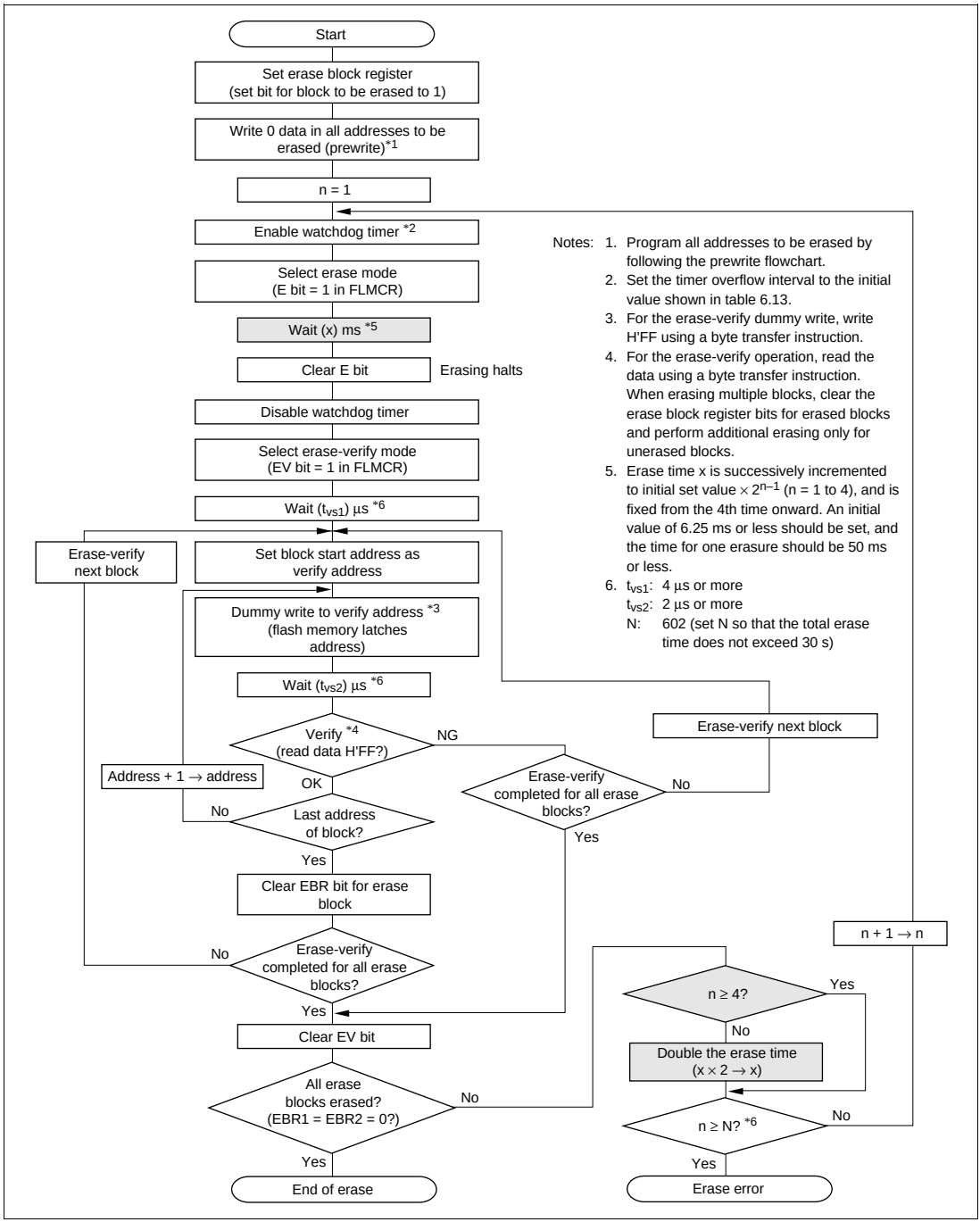


Figure 6.17 Multiple-Block Erase Flowchart

Sample Program for Erasing Multiple Blocks

This program uses the following registers:

R0: Used for erase block specification (set as explained below). Also stores address used in prewrite and erase-verify.
R1H: Used to test bits 8 to 11 of R0. Stores read data; used in dummy write.
R1L: Used to test bits 0 to 11 of R0.
R2: Specifies address where address used in prewrite and erase-verify is stored.
R3: Stores address used in prewrite and erase-verify.
R4: Stores last address of block to be erased.
R5: Used for prewrite and erase loop counter value setting.
R6L: Used for prewrite-verify and erase-verify fail count.

Arbitrary blocks can be erased by setting bits in R0. R0 settings should be made by writing with a word transfer instruction.

A bit map of R0 and a sample setting for erasing specific blocks are shown below.

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	—	—	—	—	LB3	LB2	LB1	LB0	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
	Corresponds to EBR1								Corresponds to EBR2							

Note: Bits 15 to 12 should be cleared to 0.

Example: To erase blocks LB2, SB7, and SB0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	—	—	—	—	LB3	LB2	LB1	LB0	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
	Corresponds to EBR1								Corresponds to EBR2							
	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1

R0 is set as follows:

```
MOV.W    #H'0481, R0
MOV.B    R0H,     @EBR1
MOV.B    R0L,     @EBR2
```

The values of #a, #b, #c, #d, and #e in the program depend on the operating frequency. They should be set as indicated in tables 6.12 and 6.13.

Notes: 1. In this sample program, the stack pointer (SP) is set to address H'FF80. On-chip RAM addresses H'FF7E and H'FF7F are used as a stack area. Therefore addresses H'FF7E and H'FF7F should not be used when this program is executed, and on-chip RAM should not be disabled.

2. It is assumed that this program, written in the ROM area, is transferred to the RAM area and executed there. For #RAMSTR in the program, substitute the start address of the RAM area to which the program is transferred. The value set for #RAMSTR must be an even number.

```
FLMCR:    .EQU    H'FF80
EBR1:     .EQU    H'FF82
EBR2:     .EQU    H'FF83
TCSRW:    .EQU    H'FFBE
TCW:      .EQU    H'FFBF
STACK:    .EQU    H'FF80

          .ALIGN  2
START:    MOV.W   #STACK,     SP          ; Set stack pointer

; Set R0 value as explained on previous page. This sample program erases
; all blocks.
          MOV.W   #H'0FFF,    R0          ; Select blocks to be erased (R0: EBR1/EBR2)
          MOV.B   R0H,        @EBR1       ; Set EBR1
          MOV.B   R0L,        @EBR2       ; Set EBR2

; #RAMSTR is start address of RAM area to which program is transferred
; Set #RAMSTR to even number
          MOV.W   #RAMSTR, R2             ; Transfer destination start address (RAM)
          MOV.W   #ERVADR, R3             ;
          ADD.W   R3,         R2          ; #RAMSTR + #ERVADR → R2
          MOV.W   #START,     R3          ;
          SUB.W   R3,         R2          ; Address of data area used in RAM

          MOV.B   #H'00,      R1L         ; Used to test bit R1L in R0
PRETST:   CMP.B   #H'0C,      R1L         ; R1L = H'0C?
          BEQ     ERASES                  ; If finished checking all R0 bits, branch to ERASES
          CMP.B   #H'08,      R1L         ;
          BMI     EBR2PW                  ; If R1L ≥ 8, EBR1 test; if R1L < 8, EBR2 test
          MOV.B   R1L,        R1H         ;
          SUBX    #H'08,      R1H         ; R1L – 8 → R1H
          BTST    R1H,        R0H         ; Test bit R1H in EBR1 (R0H)
          BNE     PREWRT                  ; If bit R1H in EBR1 (R0H) is 1, branch to PREWRT
          BRA                 PWADD1      ; If bit R1H in EBR1 (R0H) is 0, branch to PWADD1
EBR2PW:   BTST    R1L,        R0L         ; Test bit R1L in EBR2 (R0L)
          BNE     PREWRT                  ; If bit R1L in EBR2 (R0L) is 1, branch to PREWRT
PWADD1:   INC     R1L                     ; R1L + 1 → R1L
          MOV.W   @R2+,       R3          ; Dummy-increment R2
          BRA     PRETST                  ;
```

```
; Execute prewrite
PREWRT:  MOV.W  @R2+,      R3         ; Prewrite start address
PREW:    MOV.B  #H'00,     R6L        ; Prewrite-verify fail counter
         MOV.W  #H'a,      R5         ; Set prewrite loop counter
PREWRS:  INC    R6L                  ; Prewrite-verify fail counter + 1 → R6H
         MOV.B  #H'00      R1H        ;
         MOV.B  R1H,       @R3        ; Write H'00
         MOV.W  #H'FE5A,   R4         ;
         MOV.B  R4L,       @TCSRW:8   ;
         MOV.B  R4H,       @TCW:8     ;
         MOV.B  #H'36,     R4L        ;
         MOV.B  R4L,       @TCSRW:8   ; Start watchdog timer
         MOV.W  R5,        R4         ; Set prewrite loop counter
         BSET   #0,        @FLMCR:8   ; Set P bit

LOOPR1:  SUBS   #1,        R4         ;
         MOV.W  R4,        R4         ;
         BNE    LOOPR1               ; Wait loop
         BCLR   #0,        @FLMCR:8   ; Clear P bit
         MOV.B  #H'50,     R4L        ;
         MOV.B  R4L,       @TCSRW:8   ; Stop watchdog timer
         MOV.B  #H'c,      R4H        ; Set prewrite-verify loop counter

LOOPR2:  DEC    R4H                  ;
         BNE    LOOPR2               ; Wait loop
         MOV.B  @R3,       R1H        ; Read data = H'00?
         BEQ    PWVFOK               ; If read data = H'00, branch to PWVFOK
         CMP.B  #H'06,     R6L        ; Prewrite-verify executed 6 times?
         BEQ    ABEND1               ; If prewrite-verify executed 6 times, branch to ABEND1
         ADD.W  R5,        R5         ; Double the programming time
         BRA    PREWRS               ; Prewrite again

ABEND1:  Write error

PWVFOK:  ADDS   #1,        R3         ; Address + 1 → R3
         MOV.W  @R2,       R4         ; Start address of next block
         CMP.W  R4,        R3         ; Last address?
         BNE    PREW                 ; If not last address, prewrite next address
PWADD2:  INC    R1L                  ; Used to test bit R1L+1 in R0
         BRA    PRETST               ; Branch to PRETST

; Execute erase
ERASES:  MOV.W  #H'0000,   R6         ; Erase-verify fail counter
         MOV.W  #H'd,      R5         ; Set erase loop counter
ERASE:   ADDS   #1,        R6         ; Erase-verify fail counter + 1 → R6
         MOV.W  #H'e5A,    R4         ;
         MOV.B  R4L,       @TCSRW:8   ;
         MOV.B  R4H,       @TCW:8     ;
         MOV.B  #H'36,     R4L        ;
         MOV.B  R4L,       @TCSRW:8   ; Start watchdog timer
         MOV.W  R5,        R4         ; Set erase loop counter
```

```
         BSET    #1,         @FLMCR:8 ; Set E bit
LOOPE:   NOP
         NOP
         NOP
         NOP
         SUBS    #1,         R4       ;
         MOV.W   R4,         R4       ;
         BNE     LOOPE                ; Wait loop
         BCLR    #1,         @FLMCR:8 ; Clear E bit
         MOV.B   #H'50,      R4L      ;
         MOV.B   R4L,        @TCSRW:8 ; Stop watchdog timer

; Execute erase-verify
EVR:     MOV.W   #RAMSTR, R2          ; Transfer destination start address (RAM)
         MOV.W   #ERVADR, R3          ;
         ADD.W   R3,         R2       ; #RAMSTR + #ERVADR → R2
         MOV.W   #START,     R3       ;
         SUB.W   R3,         R2       ; Address of data area used in RAM

         MOV.B   #H'00,      R1L      ; Used to test bit R1L in R0
         MOV.B   #H'b,       R4H      ; Set erase-verify loop counter
         BSET    #3,         @FLMCR:8 ; Set EV bit
LOOPEV:  DEC     R4H                  ;
         BNE     LOOPEV               ; Wait loop
EBRTST:  CMP.B   #H'0C,      R1L      ; R1L = H'0C?
         BEQ     HANTEI               ; If finished checking all R0 bits, branch to HANTEI
         CMP.B   #H'08,      R1L      ;
         BMI     EBR2EV               ; If R1L ≥ 8, EBR1 test; if R1L < 8, EBR2 test
         MOV.B   R1L,        R1H      ;
         SUBX    #H'08,      R1H      ; R1L – 8 → R1H
         BTST    R1H,        R0H      ; Test bit R1H in EBR1 (R0H)
         BNE     ERSEVF               ; If bit R1H in EBR1 (R0H) is 1, branch to ERSEVF
         BRA     ADD01                ; If bit R1H in EBR1 (R0H) is 0, branch to ADD01
EBR2EV:  BTST    R1L,        R0L      ; Test bit R1L in EBR2 (R0L)
         BNE     ERSEVF               ; If bit R1L in EBR2 (R0L) is 1, branch to ERSEVF
ADD01:   INC     R1L                  ; R1L + 1 → R1L
         MOV.W   @R2+,       R3       ; Dummy-increment R2
         BRA     EBRTST               ;

ERASE1:  BRA     ERASE                ; Branch to ERASE via ERASE1

ERSEVF:  MOV.W   @R2+,       R3       ; Start address of block to be erase-verified
EVR2:    MOV.B   #H'FF,      R1H      ;
         MOV.B   R1H,        @R3      ; Dummy write
         MOV.B   #H'c,       R4H      ; Set erase-verify loop counter
LOOPEP:  DEC     R4H                  ;
         BNE     LOOPEP               ; Wait loop
         MOV.B   @R3+,       R1H      ; Read
         CMP.B   #H'FF,      R1H      ; Read data = H'FF?
         BNE     BLKAD                ; If read data ≠ H'FF, branch to BLKAD
         MOV.W   @R2,        R4       ; Start address of next block
```

```
          CMP.W   R4,         R3        ; Last address in block?
          BNE     EVR2                  ;

          CMP.B   #H'08,      R1L       ;
          BMI     SBCLR                 ; If R1L ≥ 8, EBR1 test; if R1L < 8, EBR2 test
          MOV.B   R1L,        R1H       ;
          SUBX    #H'08,      R1H       ; R1L – 8 → R1H
          BCLR    R1H,        R0H       ; Clear bit R1H in EBR1 (R0H)
          BRA     BLKAD                 ;
SBCLR:    BCLR    R1L,        R0L       ; Clear bit R1L in EBR2 (R0L)
BLKAD:    INC     R1L                   ; R1L + 1 → R1L
          BRA     EBRTST                ;

HANTEI:   BCLR    #3,         @FLMCR:8  ; Clear EV bit
          MOV.B   R0H,        @EBR1:8   ;
          MOV.B   R0L,        @EBR2:8   ;
          MOV.W   R0,         R4        ;
          BEQ     EOWARI                ; If EBR1/EBR2 = all 0s, normal end of erase
          MOV.W   #H'0004,    R4        ;
          CMP.W   R4,         R6        ; Erase-verify fail count = 4?
          BPL     BRER                  ; If R6 ≥ 4. branch to BRER (branch until R6 = 4–602)
          ADD.W   R5,         R5        ; If R6 < 4, double erase time (executed only for R6 = 1, 2, 3)

BRER:     MOV.W   #H'025A,    R4        ;
          CMP.W   R4,         R6        ; Erase-verify executed 602 times?
          BNE     ERASE1                ; If erase-verify not executed 602 times, erase again
          BRA     ABEND2                ; If erase-verify executed 602 times, branch to ABEND2

;****  < Block address table used in erase-verify >  ****
          .ALIGN              2
ERVADR:   .DATA.WH'0000       ;         SB0
          .DATA.WH'0080       ;         SB1
          .DATA.WH'0100       ;         SB2
          .DATA.WH'0180       ;         SB3
          .DATA.WH'0200       ;         SB4
          .DATA.WH'0400       ;         SB5
          .DATA.WH'0800       ;         SB6
          .DATA.WH'0C00       ;         SB7
          .DATA.WH'1000       ;         LB0
          .DATA.WH'2000       ;         LB1
          .DATA.WH'4000       ;         LB2
          .DATA.WH'6000       ;         LB3
          .DATA.WH'8000       ;         FLASH END

EOWARI:                               ; End of erase
ABEND2:                               ; Erase error
```

Loop Counter and Watchdog Timer Overflow Interval Settings in Programs: The settings of #a, #b, #c, #d, and #e in the program examples depend on the clock frequency. Sample loop counter settings for typical operating frequencies are shown in table 6.12. The value of #e should be set as indicated in table 6.13.

As software loops are used, there is intrinsic error, and the calculated value and actual time may not be the same. Therefore, initial values should be set so that the total write time does not exceed 1 ms, and the total erase time does not exceed 30 s.

The maximum number of writes in the program examples is set as N = 6.

Write and erase operations as shown in the flowcharts are achieved by setting the values of #a, #b, #c, and #d in the program examples as indicated in table 6.12. Use the settings shown in table 6.13 for the value of #e.

In these sample programs, wait state insertion is disabled. If wait states are used, the setting should be made after the end of the program.

The set value for the watchdog timer (WDT) overflow time is calculated on the basis of the number of instructions including the write time and erase time from the time the watchdog timer is started until it stops. Therefore, no other instructions should be added between starting and stopping of the watchdog timer in these programs.

Table 6.12 Set Values of #a, #b, #c, and #d for Typical Operating Frequencies when Sample Program is Executed in On-Chip Memory (RAM)

			Oscillation Frequency			
			f_{osc} = 16 MHz	f_{osc} = 10 MHz	f_{osc} = 8 MHz	f_{osc} = 2 MHz
			Operating Frequency			
			ø = 8 MHz	ø = 5 MHz	ø = 4 MHz	ø = 1 MHz
Meaning of Variable		Set Time	Counter Set Value	Counter Set Value	Counter Set Value	Counter Set Value
a (ø)	Programming time (initial set value)	15.8 µs	H'000F	H'0009	H'0007	H'0001
b (ø)	tvs1	4 µs	H'06	H'04	H'03	H'01
c (ø)	tvs2	2 µs	H'03	H'02	H'01	H'01
d (ø)	Erase time (initial set value)	6.25 ms	H'0C34	H'07A1	H'061A	H'0186

Formula:

If an operating frequency other than those shown in table 6.12 is used, the values can be calculated using the formula shown below. The calculation is based on an operating frequency (ø) of 5 MHz.

For a (ø) and d (ø), after decimal calculation, round down the first decimal place and convert to hexadecimal so that a (ø) and d (ø) are 15.8 µs or less and 6.25 ms or less, respectively.

For b (ø) and c (ø), after decimal calculation, round up the first decimal place and convert to hexadecimal so that b (ø) and c (ø) are 4 µs or more and 2 µs or more, respectively.

$$\text{a (ø) to d (ø)} = \frac{\text{Operating frequency ø [MHz]}}{5} \times \text{a (ø = 5) to d (ø = 5)}$$

Examples:

Sample calculations when executing a program in on-chip memory (RAM) at an operating frequency of 6 MHz

$$\text{a (ø)} = \frac{6}{5} \times 9 = 10.8 \approx 10 = \text{H'000A}$$

$$\text{b (ø)} = \frac{6}{5} \times 4 = 4.8 \approx 5 = \text{H'05}$$

$$\text{c (ø)} = \frac{6}{5} \times 2 = 2.4 \approx 3 = \text{H'03}$$

$$\text{d (ø)} = \frac{6}{5} \times 1953 = 2343.6 \approx 2343 = \text{H'0927}$$

Table 6.13 Watchdog Timer Overflow Interval Settings (Set Value of #e for Operating Frequencies)

	Oscillation Frequency			
	f_{osc} = 16 MHz	**f_{osc} = 10 MHz**	**f_{osc} = 8 MHz**	**f_{osc} = 2 MHz**
	Operating Frequency			
Variable	**ø = 8 MHz**	**ø = 5 MHz**	**ø = 4 MHz**	**ø = 1 MHz**
e (ø)	H'9B	H'DF	H'E5	H'F7

6.7.7 Prewrite-Verify Mode

Prewrite-verify mode is a verify mode used to all bits to equalize their threshold voltages before erasure.

To program all bits, write H'00 in accordance with the prewrite algorithm shown in figure 6.16. Use this procedure to set all data in the flash memory to H'00 after programming. After the necessary programming time has elapsed, exit program mode (by clearing the P bit to 0) and select prewrite-verify mode (leave the P, E, PV, and EV bits all cleared to 0). In prewrite-verify mode, a prewrite-verify voltage is applied to the memory cells at the read address. If the flash memory is read in this state, the data at the read address will be read. After selecting prewrite-verify mode, wait at least 4 µs before reading.

Note: For a sample prewriting program, see the prewrite subroutine in the sample erasing program.

6.7.8 Protect Modes

There are two modes for flash memory program/erase protection: hardware protection and software protection. These two protection modes are described below.

Software Protection: With software protection, setting the P or E bit in the flash memory control register (FLMCR) does not cause a transition to program mode or erase mode.

Details of software protection are given below.

		Functions		
Item	**Description**	**Program**	**Erase**	**Verify***
Block protect	Programming and erase protection can be set for individual blocks by settings in the erase block registers (EBR1 and EBR2). Setting EBR1 to H'F0 and EBR2 to H'00 places all blocks in the program/erase-protected state.	Disabled	Disabled	Enabled

Note: * Three modes: program-verify, erase-verify, and prewrite-verify.

Hardware Protection: Hardware protection refers to a state in which programming/erasing of flash memory is forcibly suspended or disabled. At this time, the flash memory control register (FLMCR) and erase block register (EBR1 and EBR2) settings are cleared.

Details of the hardware protection states are given below.

		Functions		
Item	**Description**	**Program**	**Erase**	**Verify*[1]**
Programming voltage (FV_{PP}) protect	When 12 V is not being applied to the FV_{PP} pin, FLMCR, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered. To obtain this protection, the V_{PP} voltage should not exceed the V_{CC} power supply voltage.*[3]	Disabled	Disabled*[2]	Disabled
Reset/standby protect	In a reset, (including a watchdog timer reset), and in sleep, subsleep, watch, and standby mode, FLMCR, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered. In a reset via the $\overline{RES}$ pin, the reset state is not reliably entered unless the $\overline{RES}$ pin is held low for at least 20 ms (oscillation settling time)*[4] after powering on. In the case of a reset during operation, the $\overline{RES}$ pin must be held low for a minimum of 10 system clock cycles (10ø).	Disabled	Disabled*[2]	Disabled

Notes: 1. Three modes: program-verify, erase-verify, and prewrite-verify.
2. All blocks are erase-disabled, and individual block specification is not possible.
3. For details, see section 6.9, Flash Memory Programming and Erasing Precautions.
4. For details, see AC Characteristics in section 13, Electrical Characteristics.

6.7.9 Interrupt Handling during Flash Memory Programming/Erasing

If an interrupt is generated while the flash memory is being programmed or erased (while the P or E bit is set in FLMCR), an operating state may be entered in which the vector will not be read correctly in the exception handling sequence, resulting in program runaway. All interrupt sources should therefore be masked to prevent interrupt generation while programming or erasing the flash memory.

6.8 Flash Memory PROM Mode (H8/3644F, H8/3643F, and H8/3642AF)

6.8.1 PROM Mode Setting

The H8/3644F, H8/3643F, and H8/3642AF, in which the on-chip ROM is flash memory, have a PROM mode as well as the on-board programming modes for programming and erasing flash memory. In PROM mode, the on-chip ROM can be freely programmed using a general-purpose PROM programmer.

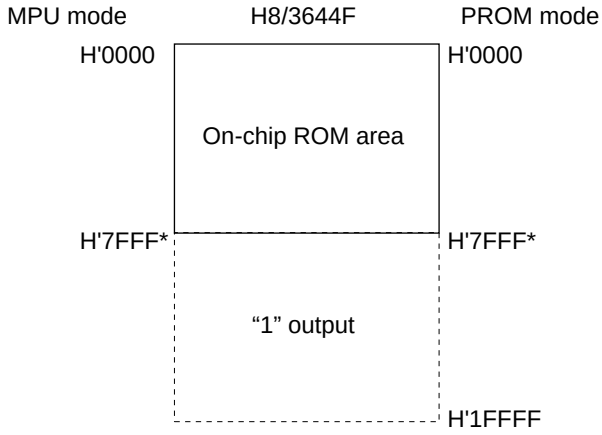
6.8.2 Socket Adapter and Memory Map

For program writing and verification, a special-purpose 100-to-32-pin adapter is mounted on the PROM programmer. Socket adapter product codes are listed in table 6.14.

Figure 6.18 shows the memory map in PROM mode. Figure 6.19 shows the socket adapter pin interconnections.

Table 6.14 Socket Adapter Product Codes

MCU Product Code	Package	Socket Adapter Product Code
HD64F3644H	64-pin QFP (FP-64A)	Under development
HD64F3644P	64-pin SDIP (DP-64S)	Under development
HD64F3644W	80-pin TQFP (TFP-80C)	Under development
HD64F3643H	64-pin QFP (FP-64A)	Under development
HD64F3643P	64-pin SDIP (DP-64S)	Under development
HD64F3643W	80-pin TQFP (TFP-80C)	Under development
HD64F3642AH	64-pin QFP (FP-64A)	Under development
HD64F3642AP	64-pin SDIP (DP-64S)	Under development
HD64F3642AW	80-pin TQFP (TFP-80C)	Under development



Note: * This example applies to the H8/3644F. This address is H'5FFF in the H8/3643F, and H'3FFF in the H8/3642AF.

Figure 6.18 Memory Map in PROM Mode

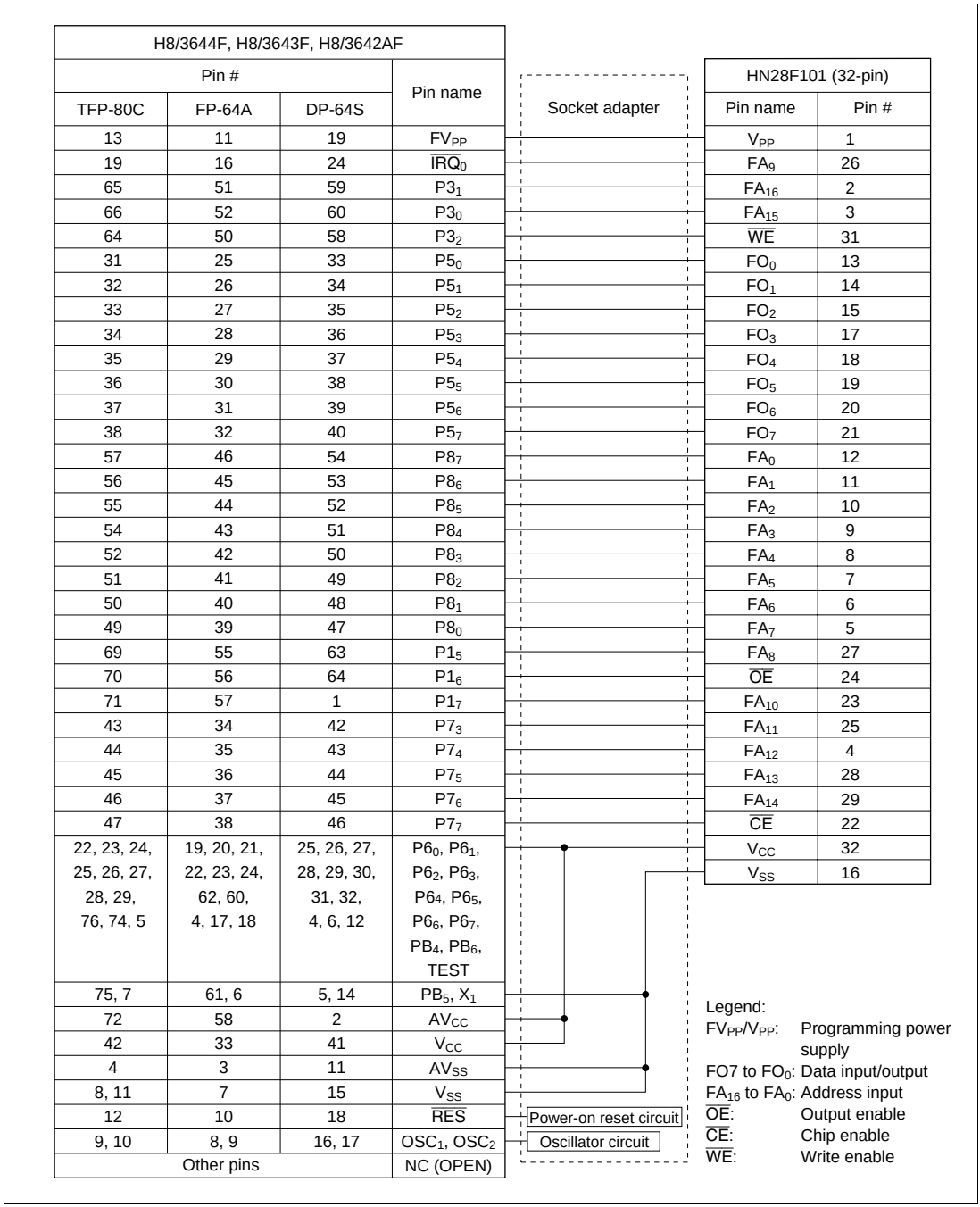


Figure 6.19 Socket Adapter Pin Correspondence (F-ZTAT)

6.8.3 Operation in PROM Mode

The program/erase/verify specifications in PROM mode are the same as for the standard HN28F101 flash memory. The H8/3644F, H8/3643F, and H8/3642AF do not have a device recognition code, so the programmer cannot read the device name automatically. Table 6.15 shows how the different operating modes are selected when using PROM mode.

Table 6.15 Operating Mode Selection In PROM Mode

		Pins						
Mode		**FV_{PP}**	**V_{CC}**	**$\overline{CE}$**	**$\overline{OE}$**	**$\overline{WE}$**	**D_7 to D_0**	**A_{16} to A_0**
Read	Read	V_{CC}*	V_{CC}	L	L	H	Data output	Address input
	Output disable	V_{CC}*	V_{CC}	L	H	H	High impedance	
	Standby	V_{CC}*	V_{CC}	H	X	X	High impedance	
Command write	Read	V_{PP}	V_{CC}	L	L	H	Data output	
	Output disable	V_{PP}	V_{CC}	L	H	H	High impedance	
	Standby	V_{PP}	V_{CC}	H	X	X	High impedance	
	Write	V_{PP}	V_{CC}	L	H	L	Data input	

Note: * In these states, the FV_{PP} pin must be set to V_{CC}.

Legend:

L: Low level

H: High level

V_{PP}: V_{PP} level

V_{CC}: V_{CC} level

X: Don't care

V_H: 11, 5 V ≤ V_H ≤ 12.5 V

Table 6.16 PROM Mode Commands

		1st Cycle			2nd Cycle		
Command	**Cycles**	**Mode**	**Address**	**Data**	**Mode**	**Address**	**Data**
Memory read	1	Write	X	H'00	Read	RA	Dout
Erase setup/erase	2	Write	X	H'20	Write	X	H'20
Erase-verify	2	Write	EA	H'A0	Read	X	EVD
Auto-erase setup/ auto-erase	2	Write	X	H'30	Write	X	H'30
Program setup/ program	2	Write	X	H'40	Write	PA	PD
Program-verify	2	Write	X	H'C0	Read	X	PVD
Reset	2	Write	X	H'FF	Write	X	H'FF

PA: Program address
EA: Erase-verify address
RA: Read address
PD: Program data
PVD: Program-verify output data
EVD: Erase-verify output data

High-Speed, High-Reliability Programming: Unused areas of the flash memory in the H8/3644F, H8/3643F, or H8/3642AF contain H'FF data (initial value). The flash memory uses a high-speed, high-reliability programming procedure. This procedure provides higher programming speed without subjecting the device to voltage stress and without sacrificing the reliability of the programmed data.

Figure 6.20 shows the basic high-speed, high-reliability programming flowchart. Tables 6.17 and 6.18 list the electrical characteristics during programming.

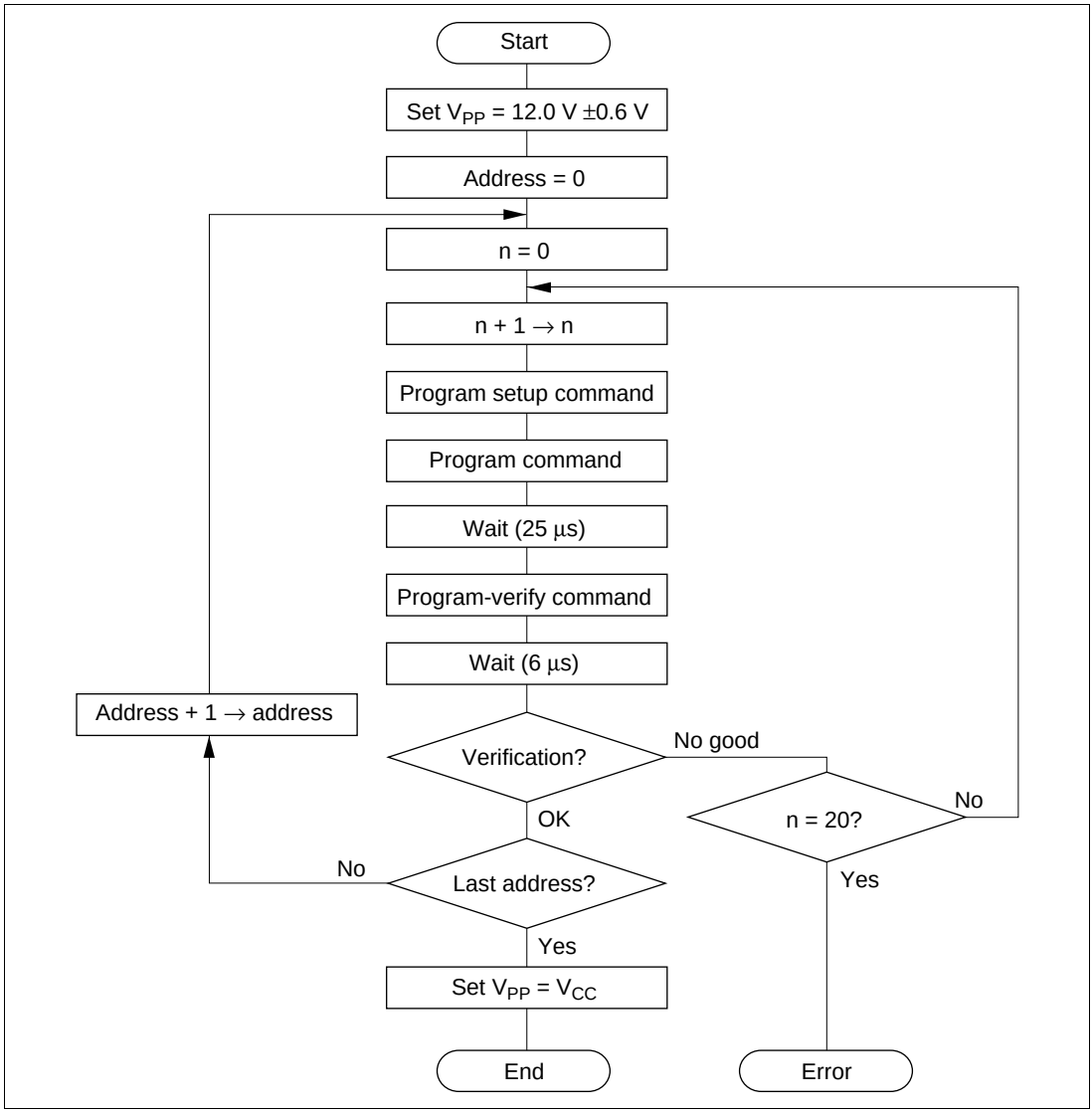
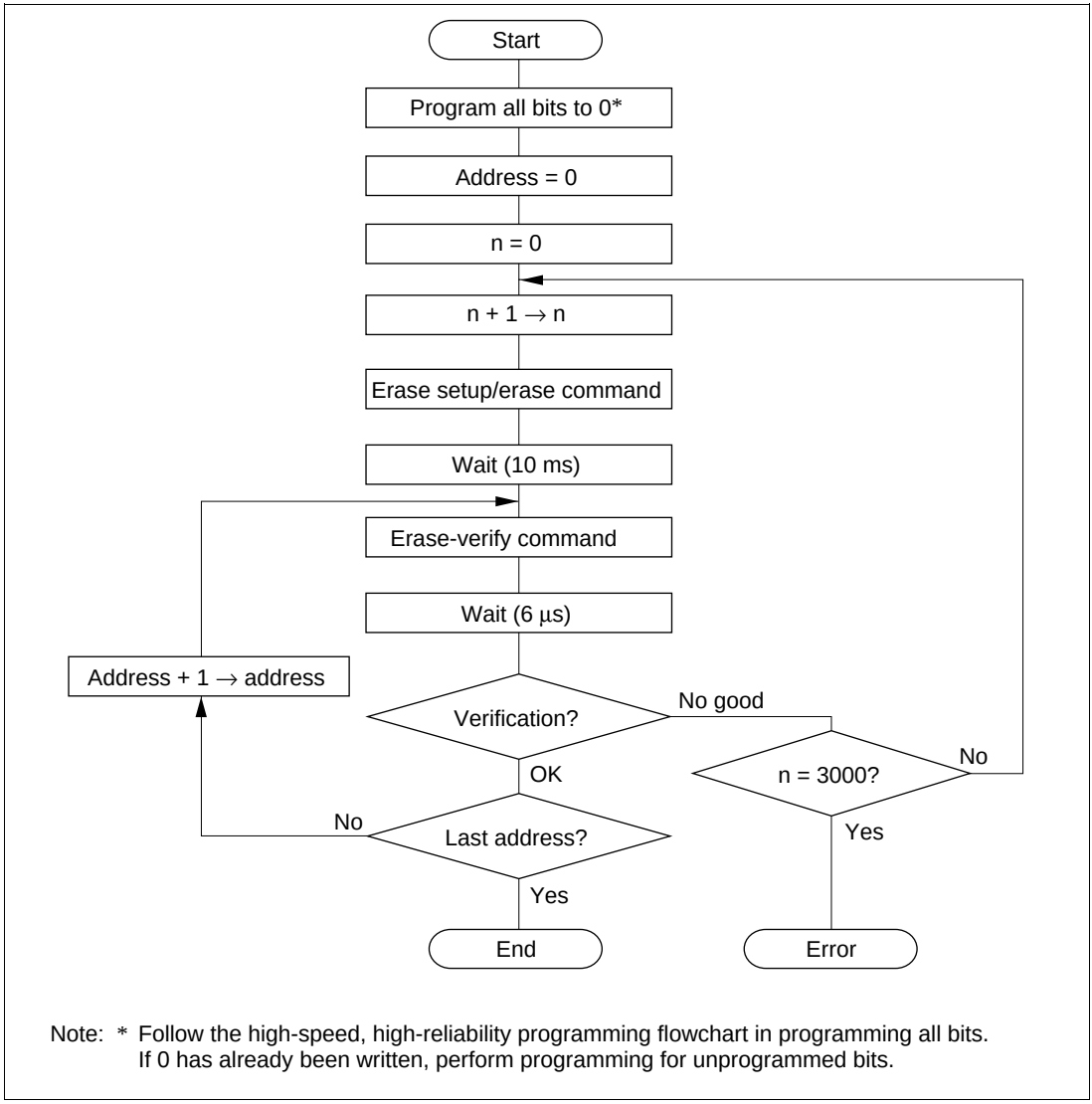


Figure 6.20 High-Speed, High-Reliability Programming

High-Speed, High-Reliability Erasing: The flash memory in the H8/3644F, H8/3643F, and H8/3642AF uses a high-speed, high-reliability erasing procedure. This procedure provides higher erasing speed without subjecting the device to voltage stress and without sacrificing the reliability of data reliability.

Figure 6.21 shows the basic high-speed, high-reliability erasing flowchart. Tables 6.17 and 6.18 list the electrical characteristics during erasing.



Note: * Follow the high-speed, high-reliability programming flowchart in programming all bits. If 0 has already been written, perform programming for unprogrammed bits.

Figure 6.21 High-Speed, High-Reliability Erasing

Table 6.17 DC Characteristics in PROM Mode

(Conditions: V_{CC} = 5.0 V ±10%, V_{PP} = 12.0 V ±0.6 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input high voltage	FO_7 to FO_0, FA_{16} to FA_0, $\overline{OE}$, $\overline{CE}$, $\overline{WE}$	V_{IH}	2.2	—	V_{CC} + 0.3	V	
Input low voltage	FO_7 to FO_0, FA_{16} to FA_0, $\overline{OE}$, $\overline{CE}$, $\overline{WE}$	V_{IL}	–0.3	—	0.8	V	
Output high voltage	FO_7 to FO_0	V_{OH}	2.4	—	—	V	I_{OH} = –200 µA
Output low voltage	FO_7 to FO_0	V_{OL}	—	—	0.45	V	I_{OL} = 1.6 mA
Input leakage current	FO_7 to FO_0, FA_{16} to FA_0, $\overline{OE}$, $\overline{CE}$, $\overline{WE}$	$\lvert I_{LI} \rvert$	—	—	2	µA	V_{in} = 0 to V_{CC}
V_{CC} current	Read	I_{CC}	—	40	80	mA	
	Program	I_{CC}	—	40	80	mA	
	Erase	I_{CC}	—	40	80	mA	
FV_{PP} current	Read	I_{PP}	—	—	10	µA	V_{PP} = 2.7 to 5.5 V
			—	10	20	mA	V_{PP} = 12.6 V
	Program	I_{PP}	—	20	40	mA	V_{PP} = 12.6 V
	Erase	I_{PP}	—	20	40	mA	V_{PP} = 12.6 V

Table 6.18 AC Characteristics in PROM Mode

(Conditions: V_{CC} = 5.0 V ±10%, V_{PP} = 12.0 V ±0.6 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Command write cycle	t_{CWC}	120	—	—	ns	Figure 6.21
Address setup time	t_{AS}	0	—	—	ns	Figure 6.22*
Address hold time	t_{AH}	60	—	—	ns	Figure 6.23
Data setup time	t_{DS}	50	—	—	ns	
Data hold time	t_{DH}	10	—	—	ns	
$\overline{CE}$ setup time	t_{CES}	0	—	—	ns	
$\overline{CE}$ hold time	t_{CEH}	0	—	—	ns	
V_{PP} setup time	t_{VPS}	100	—	—	ns	
V_{PP} hold time	t_{VPH}	100	—	—	ns	
$\overline{WE}$ programming pulse width	t_{WEP}	70	—	—	ns	
$\overline{WE}$ programming pulse high time	t_{WEH}	40	—	—	ns	
$\overline{OE}$ setup time before command write	t_{OEWS}	0	—	—	ns	
$\overline{OE}$ setup time before verify	t_{OERS}	6	—	—	µs	
Verify access time	t_{VA}	—	—	500	ns	
$\overline{OE}$ setup time before status polling	t_{OEPS}	120	—	—	ns	
Status polling access time	t_{SPA}	—	—	120	ns	
Program wait time	t_{PPW}	25	—	—	µs	
Erase wait time	t_{ET}	9	—	11	ms	
Output disable time	t_{DF}	0	—	40	ns	
Total auto-erase time	t_{AET}	0.5	—	30	s	

Note: The $\overline{CE}$, $\overline{OE}$, and $\overline{WE}$ pins should be driven high during transitions of V_{PP} from 5 V to 12 V and from 12 V to 5 V.

* Input pulse level: 0.45 V to 2.4 V
Input rise time and fall time ≤ 10 ns
Timing reference levels: 0.8 V and 2.0 V for input; 0.8 V and 2.0 V for output

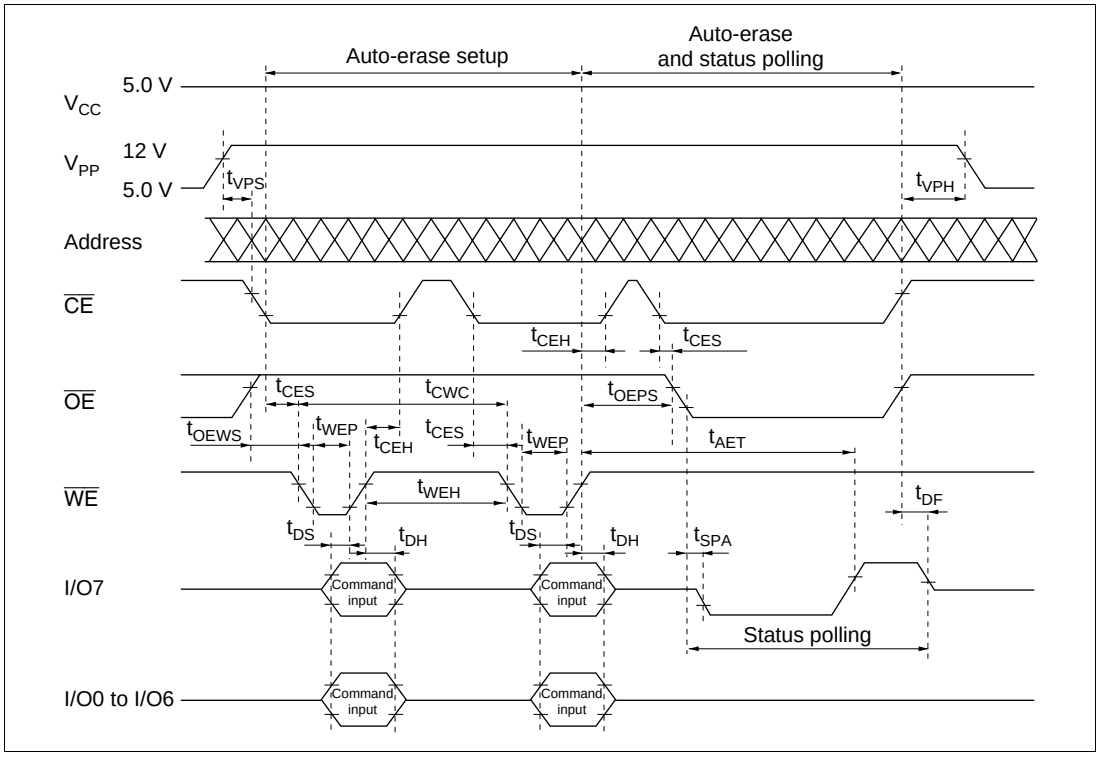
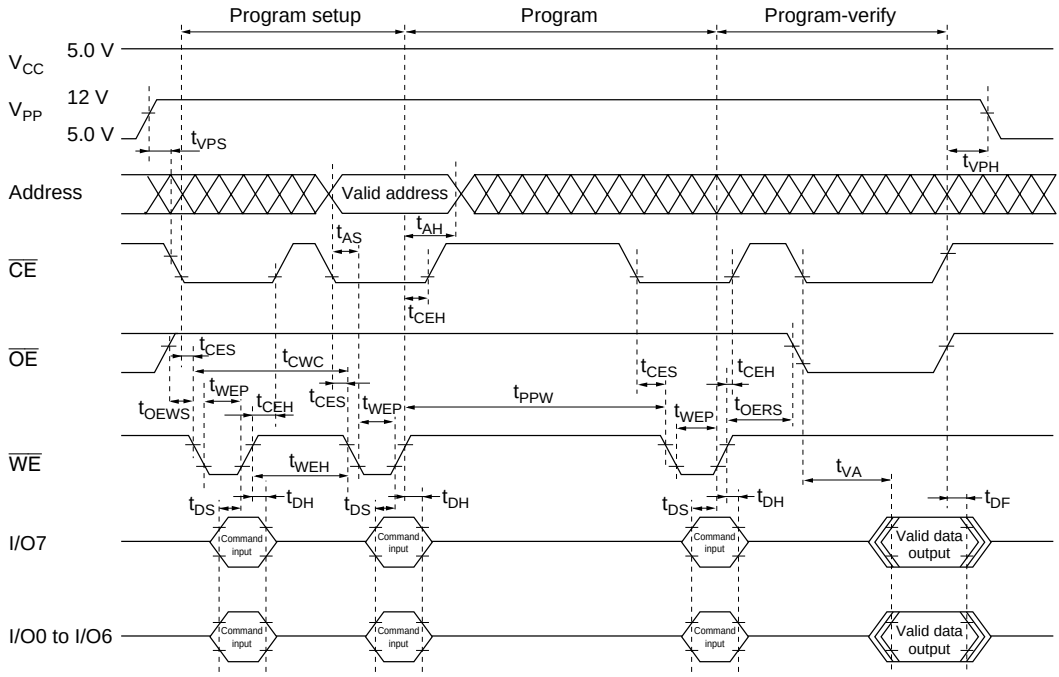
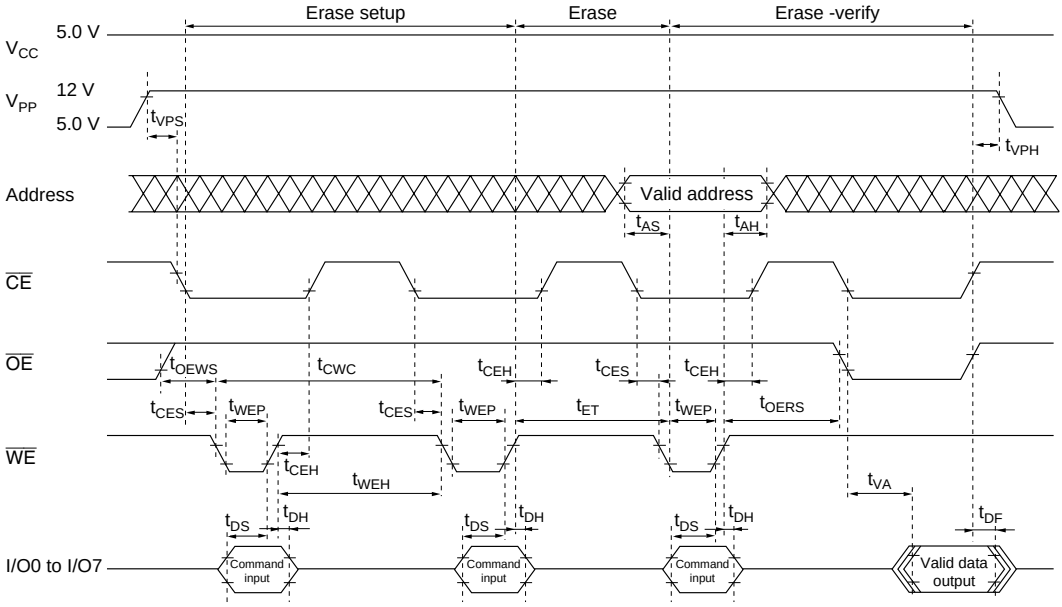


Figure 6.22 Auto-Erase Timing



Note: Program-verify data output values maybe intermediate between 1 and 0 if programming is insufficient.

Figure 6.23 High-Speed, High-Reliability Programming Timing



Note: Erase -verify data output values maybe intermediate between 1 and 0 if erasing is insufficient.

Figure 6.24 Erase Timing

6.9 Flash Memory Programming and Erasing Precautions

Precautions concerning the use of on-board programming modes and PROM mode are summarized below.

1. Program with the specified voltages and timing.
   The rated programming voltage (V_{PP}) of the flash memory is 12.0 V.
   If the PROM programmer is set to Hitachi HN28F101 specifications, V_{PP} will be 12.0 V. Applied voltages in excess of the rating can permanently damage the device. In particular, insure that the peak overshoot of the PROM programmer does not exceed the maximum rating of 13 V.

2. Before programming, check that the chip is correctly mounted in the PROM programmer. Overcurrent damage to the device can result if the index marks on the PROM programmer socket, socket adapter, and chip are not correctly aligned.

3. Do not touch the socket adapter or chip while programming. Touching either of these can cause contact faults and write errors.

4. Set H'FF as the PROM programmer buffer data for the following addresses:
   H8/3644F: H'8000 to H'1FFFF
   H8/3643F: H'6000 to H'1FFFF
   H8/3642AF: H'4000 to H'1FFFF
   The size of the PROM area is 32 kbytes in the H8/3644F, 24 kbytes in the H8/3643F, and 16 kbytes in the H8/3642AF. The addresses shown above always read H'FF, so if H'FF is not specified as programmer data, a block error will occur.

5. Precautions in applying, releasing, and cutting*[1] the programming voltage (V_{PP})

   a. Apply the programming voltage (V_{PP}) after V_{CC} has stabilized, and release V_{PP} before cutting V_{CC}.
      To avoid programming or erasing flash memory by mistake, V_{PP} should only be applied, released, and cut when the MCU is in a “stable operating condition” as described below.

      - MCU stable operating condition

        — The V_{CC} voltage must be within the rated voltage range (V_{CC} = 2.7 V to 5.5 V).
        If the V_{PP} voltage is applied, released, or cut while V_{CC} is not within its rated voltage range (V_{CC} = 2.7 V to 5.5 V), since the MCU is unstable, the flash memory may be programmed or erased by mistake. This can occur even if V_{CC} = 0 V. Adequate power supply measures should be taken, such as the insertion of a bypass capacitor, to prevent fluctuation of the V_{CC} power supply when V_{PP} is applied.

— Oscillation must have stabilized (following the elapse of the oscillation settling time) or be stopped.

When the V_{CC} power is turned on, hold the $\overline{RES}$ pin low for the duration of the oscillation settling time*[2] (t_{rc} = 20 ms) before applying V_{PP}.

— The MCU must be in the reset state, or in a state in which reset has ended normally (reset has been released) and flash memory is not being accessed.

Apply or release V_{PP} either in the reset state, or when the CPU is not accessing flash memory (when a program in on-chip RAM or external memory is executing). Flash memory data cannot be read normally at the instant when V_{PP} is applied or released, so do not read flash memory while V_{PP} is being applied or released.

For a reset during operation, apply or release V_{PP} only after the $\overline{RES}$ pin has been held low for at least 10 system clock cycles (10ø).

— The P and E bits must be cleared in the flash memory control register (FLMCR).

When applying or releasing V_{PP}, make sure that the P or E bit is not set by mistake.

— There must be no program runaway.

When V_{PP} is applied, program execution must be supervised, e.g. by the watchdog timer.

These power-on and power-off timing requirements for V_{CC} and V_{PP} should also be satisfied in the event of a power failure and in recovery from a power failure. If these requirements are not satisfied, overprogramming or overerasing may occur due to program runaway, etc., which could cause memory cells to malfunction.

b. The V_{PP} flag is set and cleared by a threshold decision on the voltage applied to the FV_{PP} pin. The threshold level is approximately in the range from V_{CC} +2 V to 11.4 V.

When this flag is set, it becomes possible to write to the flash memory control register (FLMCR) and the erase block registers (EBR1 and EBR2), even though the V_{PP} voltage may not yet have reached the programming voltage range of 12.0 V ±0.6 V.

Do not actually program or erase the flash memory until V_{PP} has reached the programming voltage range.

The programming voltage range for programming and erasing flash memory is 12.0 V ±0.6 V (11.4 V to 12.6 V). Programming and erasing cannot be performed correctly outside this range. When not programming or erasing the flash memory, insure that the V_{PP} voltage does not exceed the V_{CC} voltage. This will prevent unintentional programming and erasing.

Notes: 1. Definitions of V_{PP} application, release, and cut-off are as follows:

Application: Raising the voltage from V_{CC} to 12.0 V ±0.6 V

Release: Dropping the voltage from 12.0 V ±0.6 V to V_{CC}

Cut-off: Halting voltage application (floating state)

2. The time depends on the resonator used; refer to the electrical characteristics.

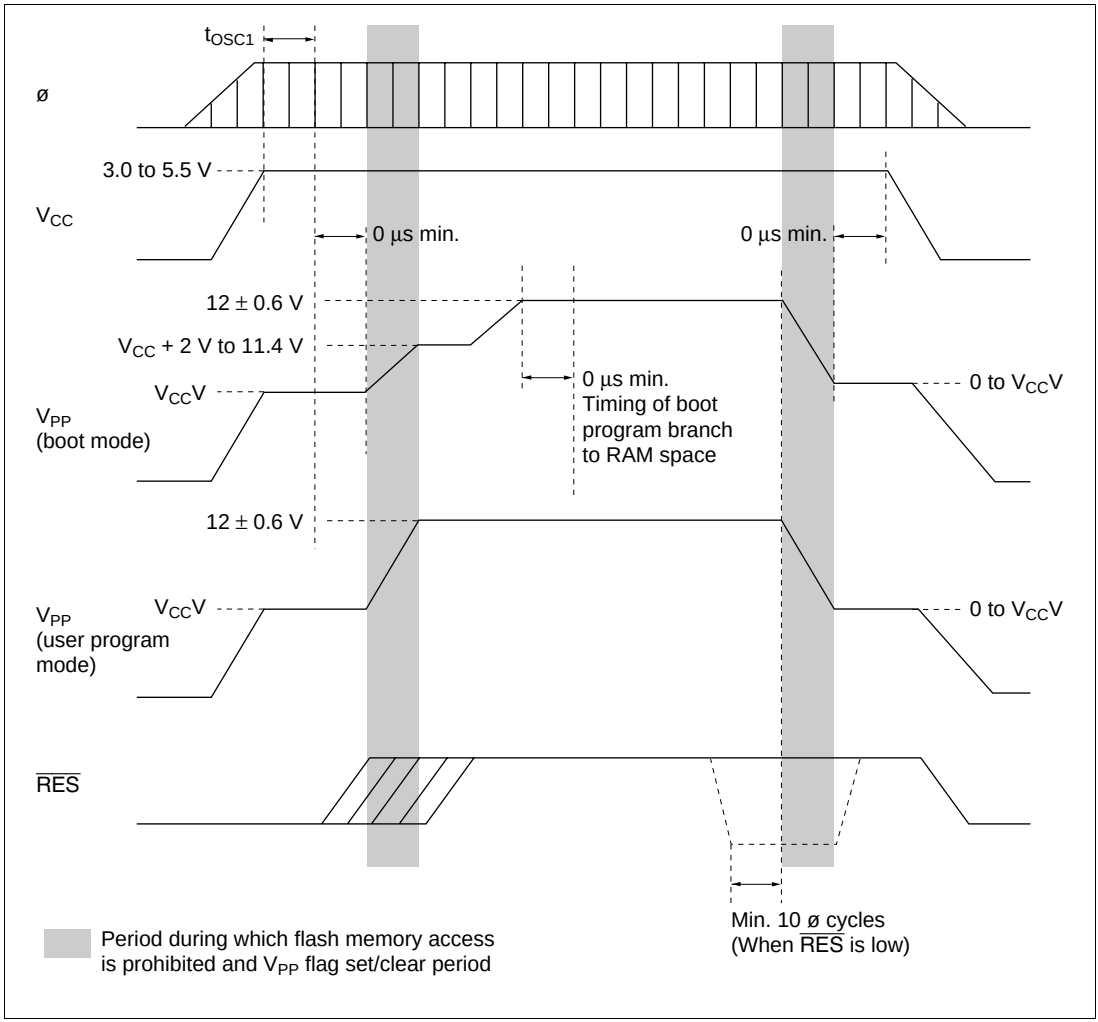


Figure 6.25 V_{PP} Power-On and Cut-Off Timing

6. Do not apply 12 V to the FV_{PP} pin during normal operation.

   To prevent erroneous programming or erasing due to program runaway, etc., apply 12 V to the FV_{PP} pin only when programming or erasing flash memory. If overprogramming or overerasing occurs due to program runaway, etc., the memory cells may not operate normally. A system configuration in which a high level is constantly applied to the FV_{PP} pin should be avoided. Also, while a high level is applied to the FV_{PP} pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

7. Design a current margin into the programming voltage (V_{PP}) power supply.
   Insure that V_{PP} remains within the range 12.0 V ±0.6 V (11.4 V to 12.6 V) during programming and erasing. Programming and erasing may become impossible outside this range.

8. Insure that peak overshoot at the FV_{PP} and TEST pins does not exceed the maximum rating. Connect bypass capacitors as close as possible to the FV_{PP} and TEST pins.
   In boot mode start-up, also, bypass capacitors should be connected to the TEST pin in the same way.

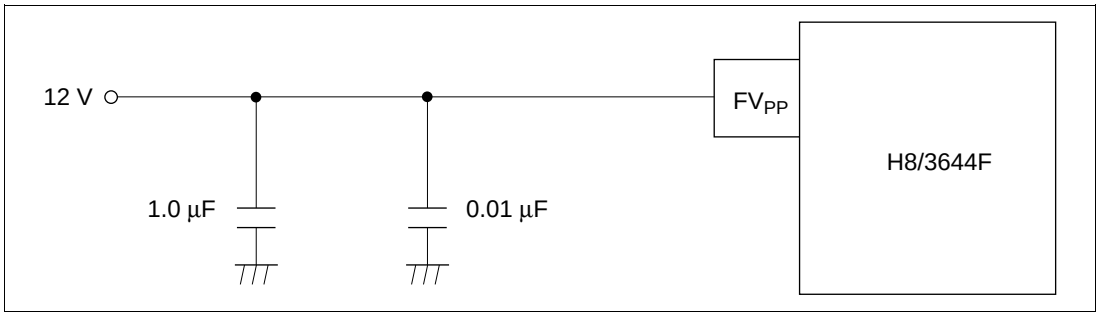


Figure 6.26 Example of V_{PP} Power Supply Circuit Design

9. Use the recommended algorithms when programming and erasing flash memory.
   The recommended algorithms enable programming and erasing to be carried out without subjecting the device to voltage stress or sacrificing program data reliability. When setting the program (P) or erase (E) bit in the flash memory control register (FLMCR), the watchdog timer should be set beforehand to prevent the specified time from being exceeded.

10. For comments on interrupt handling while flash memory is being programmed or erased, see section 6.7.9, Interrupt Handling during Flash Memory Programming/Erasing.

11. Notes on accessing flash memory control registers

    a. Flash memory control register access state in each operating mode
       The H8/3644F, H8/3643F, and H8/3642AF have flash memory control registers located at addresses H'FF80 (FLMCR), H'FF82 (EBR1), and H'FF83 (EBR2). These registers can only be accessed when 12 V is applied to the flash memory programming power supply pin, FV_{PP}.

b. To check for 12 V application/non-application in user mode

When address H'FF80 is accessed in user mode, if 12 V is being applied to FV_{PP}, FLMCR is read/written to, and its initial value after reset is H'80. When 12 V is not being applied to FV_{PP}, FLMCR is a reserved area that cannot be modified and always reads H'FF. Since bit 7 (corresponding to the V_{PP} bit) is set to 1 at this time regardless of whether or not 12 V is applied to FV_{PP}, application or release of 12 V to FV_{PP} cannot be determined simply from the 0 or 1 status of this bit. A byte data comparison is necessary to check whether 12V is being applied. The relevant coding is shown below.

```
            .
            .
LABEL1:   MOV.B  @H'FF80, R1L
          CMP.B  #H'FF, R1L
          BEQ    LABEL1
            .
            .
            .
```

Sample program for detection of 12 V application to FV_{PP} (user mode)

Table 6.19 Flash Memory DC Characteristics

V_{CC} = 3.0 V to 5.5 V, AV_{CC} = 3.0 V to 5.5 V, AV_{REF} = 3.0 V to AV_{CC}, $V_{SS} = AV_{SS}$ = 0 V, V_{PP} = 12.0 V ±0.6 V

T_a = –20°C to +75°C (regular specifications), T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
High voltage (12 V) application criterion level*	FV_{PP}, TEST	V_H	V_{CC} + 2	—	11.4	V	
FV_{PP} current	Read	I_{PP}	—	—	10	µA	V_{PP} = 2.7 to 5.5 V
			—	10	20	mA	V_{PP} = 12.6 V
	Program		—	20	40	mA	
	Erase		—	20	40	mA	

Note: * The high voltage application criterion level is as shown in the table above, but a setting of 12.0 V ±0.6 V should be made in boot mode and when programming and erasing flash memory.

Table 6.20 Flash Memory AC Characteristics

V_{CC} = 3.0 V to 5.5 V, AV_{CC} = 3.0 V to 5.5 V, AV_{REF} = 3.0 V to AV_{CC}, V_{SS} = AV_{SS} = 0 V, V_{PP} = 12.0 V ±0.6 V

T_a = –20°C to +75°C (regular specifications), T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Programming time*[1, 2]	t_P	—	50	1000	µs	
Erase time*[1, 3]	t_E	—	1	30	s	
Reprogramming capability	N_{WEC}	—	—	100	Times	
Verify setup time 1*[1]	t_{VS1}	4	—	—	µs	
Verify setup time 2*[1]	t_{VS2}	2	—	—	µs	
Flash memory read setup time*[4]	t_{FRS}	50	—	—	µs	$V_{CC} \geq 4.5$ V
		100	—	—		$V_{CC} \geq 4.5$ V

Notes: 1. Follow the program/erase algorithms shown in section 6 when making the settings.

2. Indicates the programming time per byte (the time during which the P bit is set in the flash memory control register (FLMCR)). Does not include the program-verify time.

3. Indicates the time to erase all blocks (32 kB) (the time during which the E bit is set in FLMCR). Does not include the prewrite time before erasing of the erase-verify time.

4. After powering on when using an external clock, when the programming voltage (V_{PP}) is switched from 12 V to V_{CC}, an interval at least equal to the read setup time must be allowed to elapse before reading the flash memory.

   When V_{PP} is released, this specifies the setup time from the point at which the V_{PP} voltage reaches the V_{CC} + 2 V level until the flash memory is read.

Section 7 RAM

7.1 Overview

The H8/3644 Series has 1 kbyte and 512 byte of high-speed static RAM on-chip. The RAM is connected to the CPU by a 16-bit data bus, allowing high-speed 2-state access for both byte data and word data.

7.1.1 Block Diagram

Figure 7-1 shows a block diagram of the on-chip RAM.

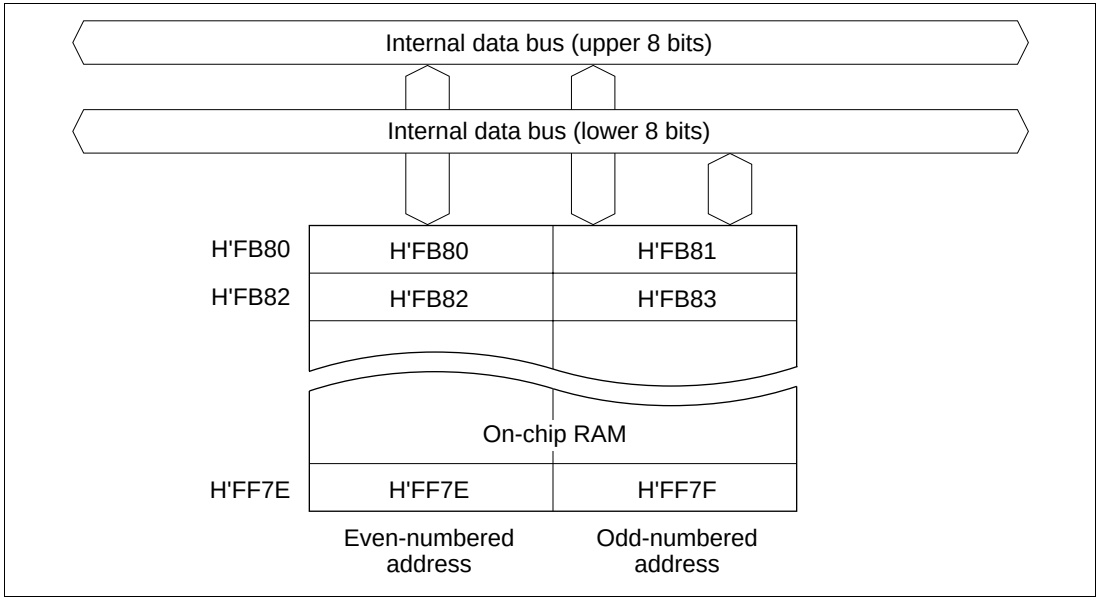


Figure 7-1 RAM Block Diagram

Section 8 I/O Ports

8.1 Overview

The H8/3644 Series is provided with three 8-bit I/O ports, three 5-bit I/O ports, two 3-bit I/O ports, and one 8-bit input-only port. Table 8.1 indicates the functions of each port.

Each port has of a port control register (PCR) that controls input and output, and a port data register (PDR) for storing output data. Input or output can be 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.

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

Table 8.1 Port Functions

Port	Description	Pins	Other Functions	Function Switching Register
Port 1	• 5-bit I/O port • Input pull-up MOS selectable	$P1_7$/$\overline{IRQ_3}$/TRGV	External interrupt 3, timer V trigger input	PMR1
		$P1_6$ to $P1_5$/ $\overline{IRQ_2}$ to $\overline{IRQ_1}$	External interrupts 2 and 1	
		$P1_4$/PWM	14-bit PWM output	PMR1
		$P1_0$/TMOW	Timer A clock output	PMR1
Port 2	• 3-bit I/O port	$P2_2$/TxD	SCI3 data output	PMR7
		$P2_1$/RxD	SCI3 data input	SCR3
		$P2_0$/SCK_1	SCI3 clock input/output	SCR3, SMR
Port 3	• 3-bit I/O port • Input pull-up MOS selectable	$P3_2$/SO_1 $P3_1$/SI_1 $P3_0$/SCK_1	SCI1 data output (SO_1), data input (SI_1), clock input/output (SCK_1)	PMR3
Port 5	• 8-bit I/O port • Input pull-up MOS	$P5_7$ /$\overline{INT_7}$	INT interrupt 7	
		$P5_6$ /$\overline{INT_6}$	INT interrupt 6	
		TMIB	Timer B event input	
		$P5_5$/$\overline{INT_5}$/ $\overline{ADTRG}$	INT interrupt 5 A/D converter external trigger input	
		$P5_4$ to $P5_0$/ $\overline{INT_4}$ to $\overline{INT_0}$	INT interrupts 4 to 0	

Table 8.1 Port Functions (cont)

Port	Description	Pins	Other Functions	Function Switching Register
Port 6	• 8-bit I/O port • High-current port	$P6_7$ to $P6_0$		
Port 7	• 5-bit I/O port	$P7_7$		
		$P7_6$/TMOV	Timer V compare-match output	TCSRV
		$P7_5$/TMCIV	Timer V clock input	
		$P7_4$/TMRIV	Timer V reset input	
		$P7_3$		
Port 8	• 8-bit I/O port	$P8_7$		
		$P8_6$/FTID	Timer X input capture D input	
		$P8_5$/FTIC	Timer X input capture C input	
		$P8_4$/FTIB	Timer X input capture B input	
		$P8_3$/FTIA	Timer X input capture A input	
		$P8_2$/FTOB	Timer X output compare B output	TOCR
		$P8_1$/FTOA	Timer X output compare A output	TOCR
		$P8_0$/FTCI	Timer X clock input	
Port 9	• 5-bit I/O port	$P9_0$* to $P9_4$		
Port B	• 8-bit input port	PB_7 to PB_0/ AN_7 to AN_0	A/D converter analog input (AN_7 to AN_0)	

Note: * There is no $P9_0$ function in the flash memory version since $P9_0$ is used as the FV_{PP} pin.

8.2 Port 1

8.2.1 Overview

Port 1 is a 5-bit I/O port. Figure 8.1 shows its pin configuration.

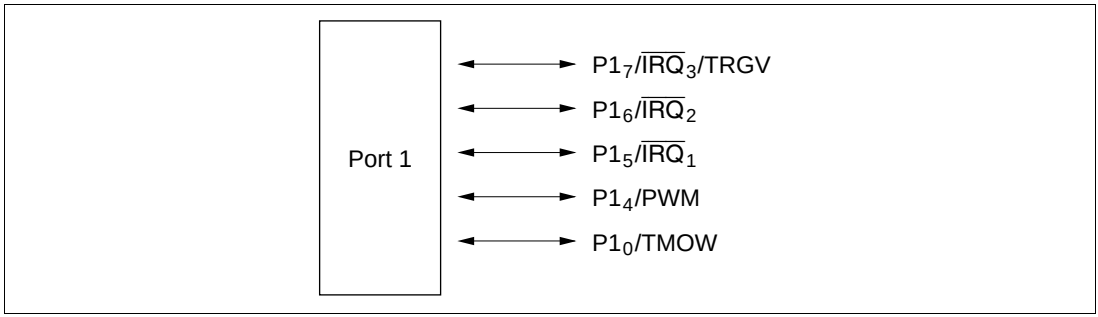


Figure 8.1 Port 1 Pin Configuration

8.2.2 Register Configuration and Description

Table 8.2 shows the port 1 register configuration.

Table 8.2 Port 1 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 1	PDR1	R/W	H'00	H'FFD4
Port control register 1	PCR1	W	H'00	H'FFE4
Port pull-up control register 1	PUCR1	R/W	H'00	H'FFED
Port mode register 1	PMR1	R/W	H'04	H'FFFC

Port Data Register 1 (PDR1)

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

Note: * Bits 3 to 1 are reserved; they are always read as 0 and cannot be modified.

PDR1 is an 8-bit register that stores data for port 1 pins $P1_7$ through $P1_4$ and $P1_0$. If port 1 is read while PCR1 bits are set to 1, the values stored in PDR1 are read, regardless of the actual pin states. If port 1 is read while PCR1 bits are cleared to 0, the pin states are read.

Upon reset, PDR1 is initialized to H'00.

Port Control Register 1 (PCR1)

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

PCR1 is an 8-bit register for controlling whether each of the port 1 pins $P1_7$ through $P1_4$ and $P1_0$ functions as an input pin or output pin. Setting a PCR1 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. The settings in PCR1 and in PDR1 are valid only when the corresponding pin is designated in PMR1 as a general I/O pin.

Upon reset, PCR1 is initialized to H'00.

PCR1 is a write-only register, which is always read as all 1s.

Port Pull-Up Control Register 1 (PUCR1)

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

Note: * Bits 3 to 1 are reserved; they are always read as 0 and cannot be modified.

PUCR1 controls whether the MOS pull-up of each of the port 1 pins $P1_7$ through $P1_4$ and $P1_0$ is on or off. When a PCR1 bit is cleared to 0, setting the corresponding PUCR1 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR1 is initialized to H'00.

Port Mode Register 1 (PMR1)

Bit	7	6	5	4	3	2	1	0
	IRQ3	IRQ2	IRQ1	PWM	—	—	—	TMOW
Initial value	0	0	0	0	0	1	0	0
Read/Write	R/W	R/W	R/W	R/W	—	—	—	R/W

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

Upon reset, PMR1 is initialized to H'04.

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

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

Note: Rising or falling edge sensing can be designated for $\overline{IRQ_3}$. Rising, falling, or both edge sensing can be designated for TRGV. For details on TRGV settings, see 9.4.2, Register Descriptions.

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

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

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

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

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

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

Bit 4—$P1_4$/PWM Pin Function Switch (PWM): This bit selects whether pin $P1_4$/PWM is used as $P1_4$ or as PWM.

Bit 4: PWM	Description	
0	Functions as $P1_4$ I/O pin	(initial value)
1	Functions as PWM output pin	

Bit 3—Reserved Bit: Bit 3 is reserved: it is always read as 0 and cannot be modified.

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

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

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

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

8.2.3 Pin Functions

Table 8.3 shows the port 1 pin functions.

Table 8.3 Port 1 Pin Functions

Pin	Pin Functions and Selection Method			
$P1_7$/$\overline{IRQ_3}$/TRGV	The pin function depends on bit IRQ3 in PMR1 and bit $PCR1_7$ in PCR1.			
	IRQ3	0		1
	$PCR1_7$	0	1	*
	Pin function	$P1_7$ input pin	$P1_7$ output pin	$\overline{IRQ_3}$/TRGV input pin
$P1_6$/$\overline{IRQ_2}$/ $P1_5$/$\overline{IRQ_1}$	The pin function depends on bits IRQ2 and IRQ1 in PMR1 and bit $PCR1_n$ in PCR1. (m = n – 4, n = 6, 5)			
	IRQ_m	0		1
	$PCR1_n$	0	1	*
	Pin function	$P1_n$ input pin	$P1_n$ output pin	IRQ_m input pin
$P1_4$/PWM	The pin function depends on bit PWM in PMR1 and bit $PCR1_4$ in PCR1.			
	PWM	0		1
	$PCR1_4$	0	1	*
	Pin function	$P1_4$ input pin	$P1_4$ output pin	PWM output pin
$P1_0$/TMOW	The pin function depends on bit TMOW in PMR1 and bit $PCR1_0$ in PCR1.			
	TMOW	0		1
	$PCR1_0$	0	1	*
	Pin function	$P1_0$ input pin	$P1_0$ output pin	TMOW output pin

Note: * Don't care

8.2.4 Pin States

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

Table 8.4 Port 1 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P1_7/\overline{IRQ_3}$/TRGV	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional
$P1_6/\overline{IRQ_2}$							
$P1_5/\overline{IRQ_1}$							
$P1_4$/PWM							
$P1_0$/TMOW							

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

8.2.5 MOS Input Pull-Up

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

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

Note: * Don't care
n = 7 to 4, 0

8.3 Port 2

8.3.1 Overview

Port 2 is a 3-bit I/O port, configured as shown in figure 8.2.

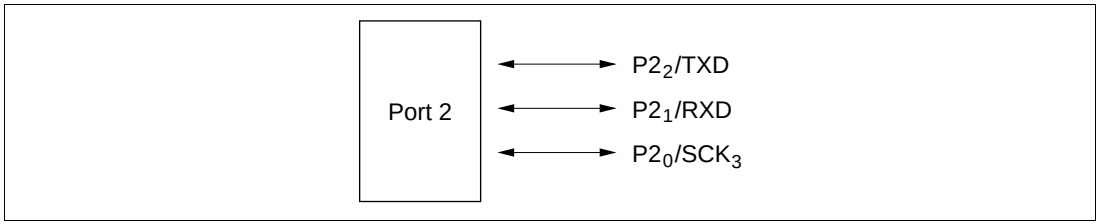


Figure 8.2 Port 2 Pin Configuration

8.3.2 Register Configuration and Description

Table 8.5 shows the port 2 register configuration.

Table 8.5 Port 2 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 2	PDR2	R/W	H'00	H'FFD5
Port control register 2	PCR2	W	H'00	H'FFE5

Port Data Register 2 (PDR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	P2$_2$	P2$_1$	P2$_0$
Initial value	0*	0*	0*	0*	0*	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

Note: * Bits 7 to 3 are reserved; they are always read as 0 and cannot be modified.

PDR2 is an 8-bit register that stores data for port 2 pins P2$_2$ to P2$_0$. If port 2 is read while PCR2 bits are set to 1, the values stored in PDR2 are read, regardless of the actual pin states. If port 2 is read while PCR2 bits are cleared to 0, the pin states are read.

Upon reset, PDR2 is initialized to H'00.

Port Control Register 2 (PCR2)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PCR2_2$	$PCR2_1$	$PCR2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	W	W	W

PCR2 is an 8-bit register for controlling whether each of the port 1 pins $P2_2$ to $P2_0$ functions as an input pin or output pin. Setting a PCR2 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. The settings in PCR2 and PDR2 are valid only when the corresponding pin is designated in SCR3 as a general I/O pin.

Upon reset, PCR2 is initialized to H'00.

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

8.3.3 Pin Functions

Table 8.6 shows the port 2 pin functions.

Table 8.6 Port 2 Pin Functions

Pin	Pin Functions and Selection Method			
$P2_2$/TXD	The pin function depends on bit TXD in PMR7 and bit $PCR2_2$ in PCR2.			
	TXD	0		1
	$PCR2_2$	0	1	*
	Pin function	$P2_2$ input pin	$P2_2$ output pin	TXD output pin
$P2_1$/RXD	The pin function depends on bit RE in SCR3 and bit $PCR2_1$ in PCR2.			
	RE	0		1
	$PCR2_1$	0	1	*
	Pin function	$P2_1$ input pin	$P2_1$ output pin	RXD input pin
$P2_0$/SCK_3	The pin function depends on bits CKE1 and CKE0 in SCR3, bit COM in SMR, and bit $PCR2_0$ in PCR2.			
	CKE1	0		1
	CKE0	0	1	*
	COM	0	1 *	*
	$PCR2_0$	0	1	* *
	Pin function	$P2_0$ input pin	$P2_0$ output pin	SCK_3 output pin / SCK_3 input pin

Note: * Don't care

8.3.4 Pin States

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

Table 8.7 Port 2 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P2_2$/TXD $P2_1$/RXD $P2_0$/SCK_3	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.4 Port 3

8.4.1 Overview

Port 3 is a 8-bit I/O port, configured as shown in figure 8.3.

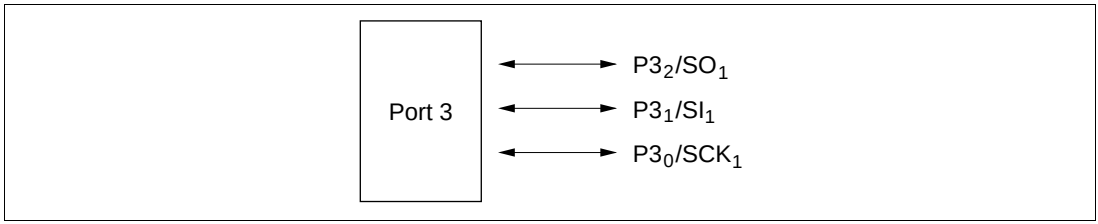


Figure 8.3 Port 3 Pin Configuration

8.4.2 Register Configuration and Description

Table 8.8 shows the port 3 register configuration.

Table 8.8 Port 3 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 3	PDR3	R/W	H'00	H'FFD6
Port control register 3	PCR3	W	H'00	H'FFE6
Port pull-up control register 3	PUCR3	R/W	H'00	H'FFEE
Port mode register 3	PMR3	R/W	H'00	H'FFFD
Port mode register 7	PMR7	R/W	H'F8	H'FFFF

Port Data Register 3 (PDR3)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	P3$_2$	P3$_1$	P3$_0$
Initial value	0*	0*	0*	0*	0*	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

Note: * Bits 7 to 3 are reserved; they are always read as 0 and cannot be modified.

PDR3 is an 8-bit register that stores data for port 3 pins $P3_2$ to $P3_0$. 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.

Upon reset, PDR3 is initialized to H'00.

Port Control Register 3 (PCR3)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PCR3_2$	$PCR3_1$	$PCR3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	W	W	W

PCR3 is an 8-bit register for controlling whether each of the port 3 pins $P3_2$ to $P3_0$ 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.

Upon reset, PCR3 is initialized to H'00.

PCR3 is a write-only register, which is always read as all 1s.

Port Pull-Up Control Register 3 (PUCR3)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PUCR3_2$	$PUCR3_1$	$PUCR3_0$
Initial value	0*	0*	0*	0*	0*	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

Note: * Bits 7 to 3 are reserved; they are always read as 0 and cannot be modified.

PUCR3 controls whether the MOS pull-up of each of the port 3 pins $P3_2$ to $P3_0$ 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.

Upon reset, PUCR3 is initialized to H'00.

Port Mode Register 3 (PMR3)

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

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

Upon reset, PMR3 is initialized to H'00.

Bits 7 to 3—Reserved Bits: Bits 7 to 3 are reserved: they are always read as 0 and cannot be modified.

Bit 2—$P3_2/SO_1$ Pin Function Switch (SO1): This bit selects whether pin $P3_2/SO_1$ is used as $P3_2$ or as SO_1.

Bit 2: SO1	Description	
0	Functions as $P3_2$ I/O pin	(initial value)
1	Functions as SO_1 output pin	

Bit 1—$P3_1/SI_1$ Pin Function Switch (SI1): This bit selects whether pin $P3_1/SI_1$ is used as $P3_1$ or as SI_1.

Bit 1: SI1	Description	
0	Functions as $P3_1$ I/O pin	(initial value)
1	Functions as SI_1 input pin	

Bit 0—$P3_0/SCK_1$ Pin Function Switch (SCK1): This bit selects whether pin $P3_0/SCK_1$ is used as $P3_0$ or as SCK_1.

Bit 0: SCK1	Description	
0	Functions as $P3_0$ I/O pin	(initial value)
1	Functions as SCK_1 I/O pin	

Port Mode Register 7 (PMR7)

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

PMR7 is an 8-bit read/write register that turns the PMOS transistors of pins and $P3_2/SO_1$ on and off.

Upon reset, PMR7 is initialized to H'F8.

Bits 7 to 3—Reserved Bits: Bits 7 to 3 are reserved; they are always read as 1, and cannot be modified.

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

Bit 2: TXD	Description	
0	Functions as $P2_2$ I/O pin	(initial value)
1	Functions as TXD output pin	

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

Bit 0—$P3_2/SO_1$ pin PMOS control (POF1): This bit controls the PMOS transistor in the $P3_2/SO_1$ pin output buffer.

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

8.4.3 Pin Functions

Table 8.9 shows the port 3 pin functions.

Table 8.9 Port 3 Pin Functions

Pin	Pin Functions and Selection Method			
$P3_2/SO_1$	The pin function depends on bit SO1 in PMR3 and bit $PCR3_2$ in PCR3.			
	SO1	0		1
	$PCR3_2$	0	1	*
	Pin function	$P3_2$ input pin	$P3_2$ output pin	SO_1 output pin
$P3_1/SI_1$	The pin function depends on bit SI1 in PMR3 and bit $PCR3_1$ in PCR3.			
	SI1	0		1
	$PCR3_1$	0	1	*
	Pin function	$P3_1$ input pin	$P3_1$ output pin	SI_1 input pin
$P3_0/SCK_1$	The pin function depends on bit SCK1 in PMR3, bit CKS3 in SCR1, and bit $PCR3_0$ in PCR3.			
	SCK1	0		1	
	CKS3	*		0	1
	$PCR3_0$	0	1	*	*
	Pin function	$P3_0$ input pin	$P3_0$ output pin	SCK_1 output pin	SCK_1 input pin

Note: * Don't care

8.4.4 Pin States

Table 8.10 shows the port 3 pin states in each operating mode.

Table 8.10 Port 3 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P3_2/SO_1$ $P3_1/SI_1$ $P3_0/SCK_1$	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 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		1
$PUCR3_n$	0	1	*
MOS input pull-up	Off	On	Off

Note: * Don't care
(n = 2 to 0)

8.5 Port 5

8.5.1 Overview

Port 5 is an 8-bit I/O port, configured as shown in figure 8.4.

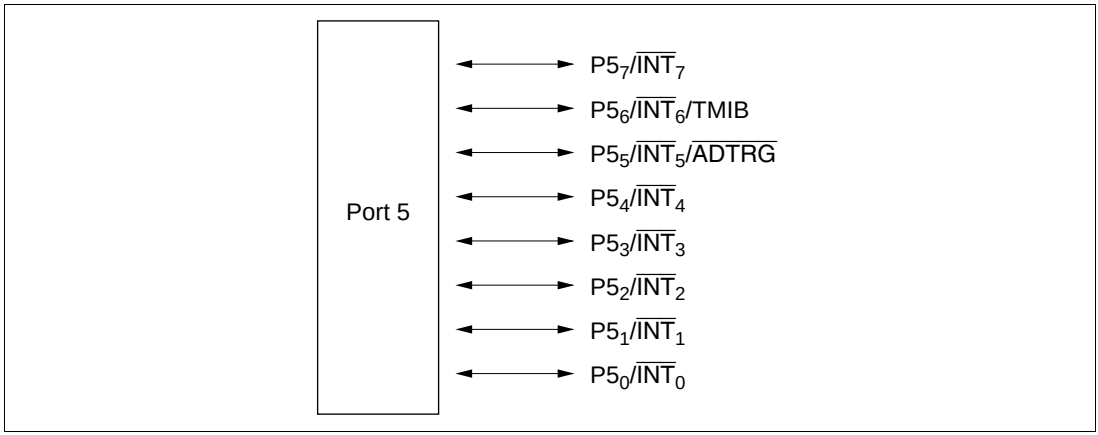


Figure 8.4 Port 5 Pin Configuration

8.5.2 Register Configuration and Description

Table 8.11 shows the port 5 register configuration.

Table 8.11 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'FFEF

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.

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.

Upon reset, PCR5 is initialized to H'00.

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

Port Pull-Up Control Register 5 (PUCR5)

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

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

Upon reset, PUCR5 is initialized to H'00.

8.5.3 Pin Functions

Table 8.12 shows the port 5 pin functions.

Table 8.12 Port 5 Pin Functions

Pin	Pin Functions and Selection Method		
$P5_7/\overline{INT}_7$	The pin function depends on bit $PCR5_7$ in PCR5.		
	$PCR5_7$	0	1
	Pin function	$P5_7$ input pin	$P5_7$ output pin
		$\overline{INT}_7$ input pin	
$P5_6/\overline{INT}_6$/TMIB	The pin function depends on bit $PCR5_6$ in PCR5.		
	$PCR5_6$	0	1
	Pin function	$P5_6$ input pin	$P5_6$ output pin
		$\overline{INT}_6$ input pin and TMIB input pin	
$P5_5/\overline{INT}_5/\overline{ADTRG}$	The pin function depends on bit $PCR5_5$ in PCR5.		
	$PCR5_5$	0	1
	Pin function	$P5_5$ input pin	$P5_5$ output pin
		$\overline{INT}_5$ input pin and $\overline{ADTRG}$ input pin	
$P5_4/\overline{INT}_4$ to $P5_0/\overline{INT}_0$	The pin function depends on bit $PCR5_n$ in PCR5. (n = 4 to 0)		
	$PCR5_n$	0	1
	Pin function	$P5_n$ input pin	$P5_n$ output pin
		$\overline{INT}_n$ input pin	

8.5.4 Pin States

Table 8.13 shows the port 5 pin states in each operating mode.

Table 8.13 Port 5 Pin States

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

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

8.5.5 MOS Input Pull-Up

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

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

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

8.6 Port 6

8.6.1 Overview

Port 6 is an 8-bit large-current I/O port, with a maximum sink current of 10 mA. The port 6 pin configuration is shown in figure 8.5.

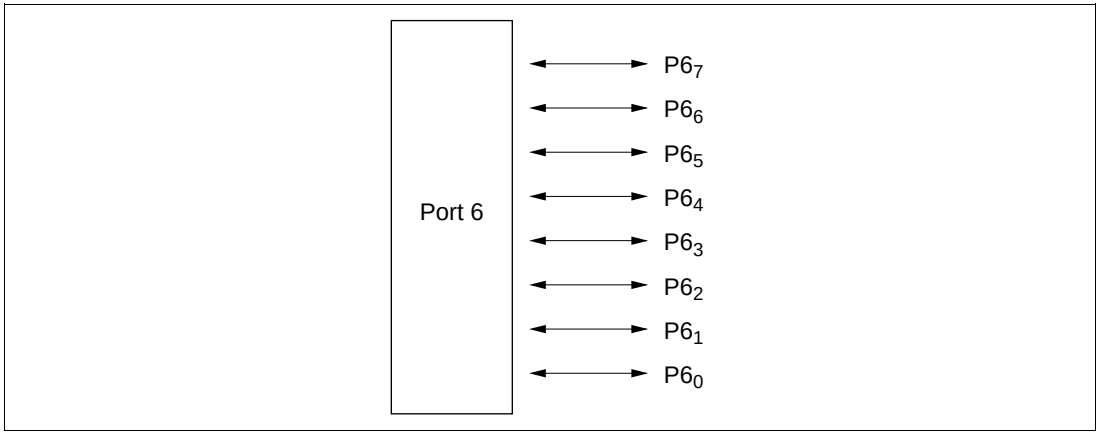


Figure 8.5 Port 6 Pin Configuration

8.6.2 Register Configuration and Description

Table 8.14 shows the port 6 register configuration.

Table 8.14 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 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$_0$.

When a bit in PCR6 is set to 1, if port 6 is read the value of the corresponding PDR6 bit is returned directly regardless of the pin state. When a bit in PCR6 is cleared to 0, if port 6 is read the corresponding pin state is read.

Upon reset, PDR6 is initialized to H'00.

Port Control Register 6 (PCR6)

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

PCR6 is an 8-bit register for controlling whether each of the port 6 pins $P6_7$ to $P6_0$ functions as an input pin or output pin.

When a bit in PCR6 is set to 1, the corresponding pin of $P6_7$ to $P6_0$ becomes an output pin.

Upon reset, PCR6 is initialized to H'00.

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

8.6.3 Pin Functions

Table 8.15 shows the port 6 pin functions.

Table 8.15 Port 6 Pin Functions

Pin	Pin Functions and Selection Method		
$P6_7$ to $P6_0$	The pin function depends on bit $PCR6_n$ in PCR6 (n = 7 to 0)		
	$PCR6_n$	0	1
	Pin function	$P6_n$ input pin	$P6_n$ output pin

8.6.4 Pin States

Table 8.16 shows the port 6 pin states in each operating mode.

Table 8.16 Port 6 Pin States

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

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

8.7 Port 7

8.7.1 Overview

Port 7 is a 8-bit I/O port, configured as shown in figure 8.6.

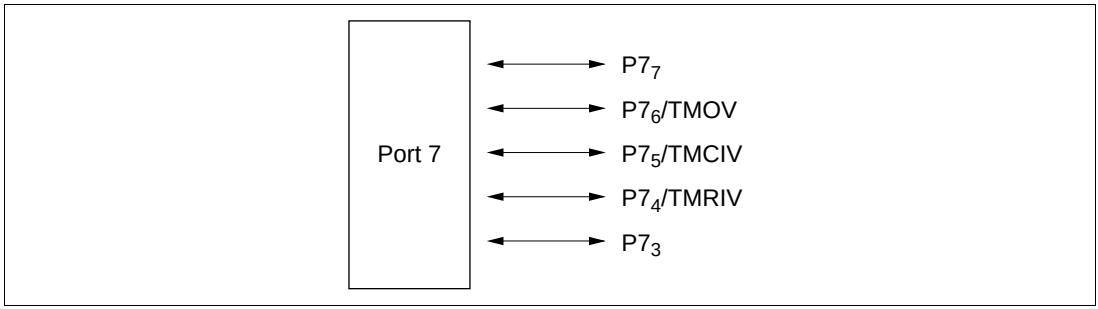


Figure 8.6 Port 7 Pin Configuration

8.7.2 Register Configuration and Description

Table 8.17 shows the port 7 register configuration.

Table 8.17 Port 7 Registers

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

Port Data Register 7 (PDR7)

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

Note: * Bits 2 to 0 are reserved; they are always read as 0 and cannot be modified.

PDR7 is an 8-bit register that stores data for port 7 pins $P7_7$ to $P7_3$. 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.

Port Control Register 7 (PCR7)

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

PCR7 is an 8-bit register for controlling whether each of the port 7 pins $P7_7$ to $P7_3$ functions as an input pin or output pin. Setting a PCR7 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCR7 is initialized to H'00.

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

8.7.3 Pin Functions

Table 8.18 shows the port 7 pin functions.

Table 8.18 Port 7 Pin Functions

Pin	Pin Functions and Selection Method		
$P7_7$, $P7_3$	The pin function depends on bit $PCR7_n$ in PCR7. (n = 7 or 3)		
	$PCR7_n$	0	1
	Pin function	$P7_n$ input pin	$P7_n$ output pin
$P7_6$/TMOV	The pin function depends on bit $PCR7_6$ in PCR7 and bits OS3 to OS0 in TCSRV.		
	OS3 to OS0	0000	Not 0000
	$PCR7_6$	0 / 1	*
	Pin function	$P7_6$ input pin / $P7_6$ output pin	TMOV output pin
$P7_5$/TMCIV	The pin function depends on bit $PCR7_5$ in PCR7.		
	$PCR7_5$	0	1
	Pin function	$P7_5$ input pin	$P7_5$ output pin
		TMCIV input pin	
$P7_4$/TMRIV	The pin function depends on bit $PCR7_4$ in PCR7.		
	$PCR7_4$	0	1
	Pin function	$P7_4$ input pin	$P7_4$ output pin
		TMRIV input pin	

Note: * Don't care

8.7.4 Pin States

Table 8.19 shows the port 7 pin states in each operating mode.

Table 8.19 Port 7 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P7_7$ to $P7_3$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.8 Port 8

8.8.1 Overview

Port 8 is an 8-bit I/O port configured as shown in figure 8.7.

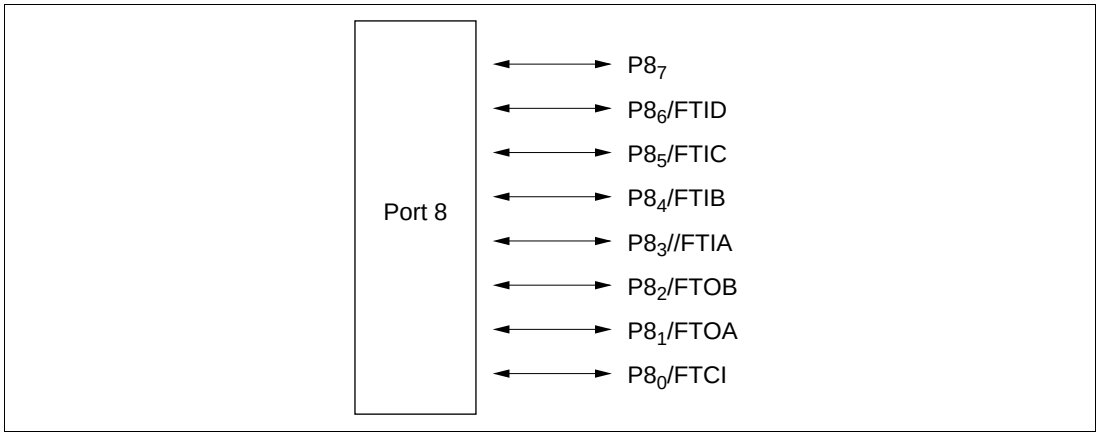


Figure 8.7 Port 8 Pin Configuration

8.8.2 Register Configuration and Description

Table 8.20 shows the port 8 register configuration.

Table 8.20 Port 8 Registers

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

Port Data Register 8 (PDR8)

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

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

Upon reset, PDR8 is initialized to H'00.

Port Control Register 8 (PCR8)

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

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

Upon reset, PCR8 is initialized to H'00.

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

8.8.3 Pin Functions

Table 8.21 shows the port 8 pin functions.

Table 8.21 Port 8 Pin Functions

Pin	Pin Functions and Selection Method		
$P8_7$	The pin function depends on bit $PCR8_7$ in PCR8.		
	$PCR8_7$	0	1
	Pin function	$P8_7$ input pin	$P8_7$ output pin
$P8_6$/FTID	The pin function depends on bit $PCR8_6$ in PCR8.		
	$PCR8_6$	0	1
	Pin function	$P8_6$ input pin	$P8_6$ output pin
		FTID input pin	
$P8_5$/FTIC	The pin function depends on bit $PCR8_5$ in PCR8.		
	$PCR8_5$	0	1
	Pin function	$P8_5$ input pin	$P8_5$ output pin
		FTIC input pin	
$P8_4$/FTIB	The pin function depends on bit $PCR8_4$ in PCR8.		
	$PCR8_4$	0	1
	Pin function	$P8_4$ input pin	$P8_4$ output pin
		FTIB input pin	
$P8_3$/FTIA	The pin function depends on bit $PCR8_3$ in PCR8.		
	$PCR8_3$	0	1
	Pin function	$P8_3$ input pin	$P8_3$ output pin
		FTIA input pin	
$P8_2$/FTOB	The pin function depends on bit $PCR8_2$ in PCR8 and bit OEB in TOCR.		
	OEB	0	1
	$PCR8_2$	0	1	*
	Pin function	$P8_2$ input pin	$P8_2$ output pin	FTOB output pin

Note: * Don't care

Table 8.21 Port 8 Pin Functions (cont)

Pin	Pin Functions and Selection Method		
$P8_1$/FTOA	The pin function depends on bit $PCR8_1$ in PCR8 and bit OEA in TOCR.		
	OEA	0	1
	$PCR8_1$	0	1	*
	Pin function	$P8_1$ input pin	$P8_1$ output pin	FTOA output pin
$P8_0$/FTCI	The pin function depends on bit $PCR8_0$ in PCR8.		
	$PCR8_0$	0	1
	Pin function	$P8_0$ input pin	$P8_0$ output pin
		FTCI input pin	

Note: * Don't care

8.8.4 Pin States

Table 8.22 shows the port 8 pin states in each operating mode.

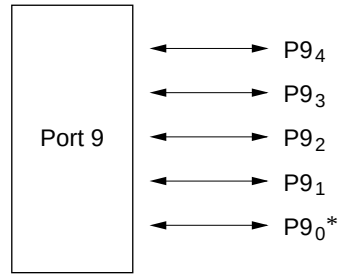
Table 8.22 Port 8 Pin States

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

8.9 Port 9

8.9.1 Overview

Port 9 is a 5-bit I/O port, configured as shown in figure 8.8.



Note: * There is no $P9_0$ function in the flash memory version since $P9_0$ is used as the FV_{PP} pin.

Figure 8.8 Port 9 Pin Configuration

8.9.2 Register Configuration and Description

Table 8.23 shows the port 9 register configuration.

Table 8.23 Port 9 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 9	PDR9	R/W	H'C0	H'FFDC
Port control register 9	PCR9	W	H'C0	H'FFEC

Port Data Register 9 (PDR9)

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

Notes: 1. Bits 7 to 6 are reserved; they are always read as 1 and cannot be modified.
2. Bit 5 is reserved; it is always read as 0 and cannot be modified.
3. In the on-chip flash memory version, this bit is always read as 0 and cannot be modified.

PDR9 is an 8-bit register that stores data for port 9 pins $P9_4$ to $P9_0$. If port 9 is read while PCR9 bits are set to 1, the values stored in PDR9 are read, regardless of the actual pin states. If port 9 is read while PCR9 bits are cleared to 0, the pin states are read.

Upon reset, PDR9 is initialized to H'C0.

Port Control Register 9 (PCR9)

Bit	7	6	5	4	3	2	1	0
	—	—	—	$PCR9_4$	$PCR9_3$	$PCR9_2$	$PCR9_1$	$PCR9_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	W	W	W	W	W

PCR9 controls whether each of the port 9 pins $P9_4$ to $P9_0$ functions as an input pin or output pin. Setting a PCR9 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCR9 is initialized to H'C0.

PCR9 is a write-only register, which is always reads as all 1.

8.9.3 Pin Functions

Table 8.24 shows the port 9 pin functions.

Table 8.24 Port 9 Pin Functions

Pin	Pin Functions and Selection Method
$P9_n$	The pin function depends on bit $PCR9_n$ in PCR9. (n = 4 to 0)

$PCR9_n$	0	1
Pin function	$P9_n$ input pin	$P9_n$ output pin

8.9.4 Pin States

Table 8.25 shows the port 9 pin states in each operating mode.

Table 8.25 Port 9 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P9_4$ to $P9_0$	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 an 8-bit input-only port, configured as shown in figure 8.9.

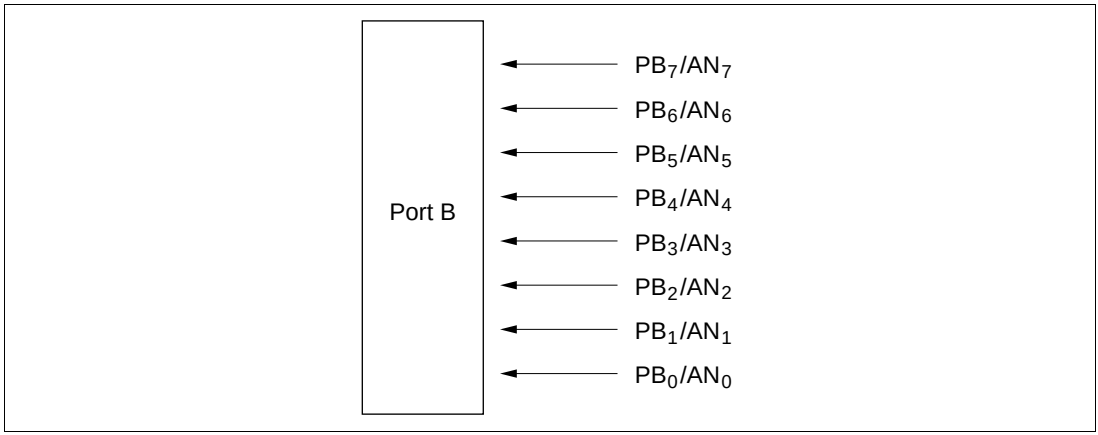


Figure 8.9 Port B Pin Configuration

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	Address
Port data register B	PDRB	R	H'FFDD

Port Data Register B (PDRB)

Bit	7	6	5	4	3	2	1	0
	PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0
Read/Write	R	R	R	R	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.

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_n/AN_n	Always as below.	
		(n = 7 to 0)
	Pin function	PB_n input pin or AN_n input pin

8.10.4 Pin States

Table 8.28 shows the port B pin states in each operating mode.

Table 8.28 Port B Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
PB_n/AN_n	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance

(n = 7 to 0)

Section 9 Timers

9.1 Overview

The H8/3644 Series provides five timers: timers A, B1, V, X, and a watchdog timer. 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 timer				
	• Interval function	ø/8 to ø/8192 (8 choices)	—	—	
	• Time base	$ø_w$/128 (choice of 4 overflow periods)			
	• Clock output	ø/4 to ø/32 $ø_w$/4 to $ø_w$/32 (8 choices)	—	TMOW	
Timer B1	• 8-bit timer • Interval timer • Event counter	ø/4 to ø/8192 (7 choices)	TMIB	—	
Timer V	• 8-bit timer • Event counter • Output control by dual compare match • Counter clearing option • Count-up start by external trigger input can be specified	ø/4 to ø/128 (6 choices)	TMCIV	TMOV	
Timer X	• 16-bit free-running timer • 2 output compare channels • 4 input capture channels • Counter clearing option • Event counter	ø/2 to ø/32 (3 choices)	FTCI FTIA FTIB FTIC FTID	FTOA FTOB	
Watchdog timer	• Reset signal generated when 8-bit counter overflows	ø/8192	—	—	

9.2 Timer A

9.2.1 Overview

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

Features

Features of timer A are given below.

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

Block Diagram

Figure 9.1 shows a block diagram of timer A.

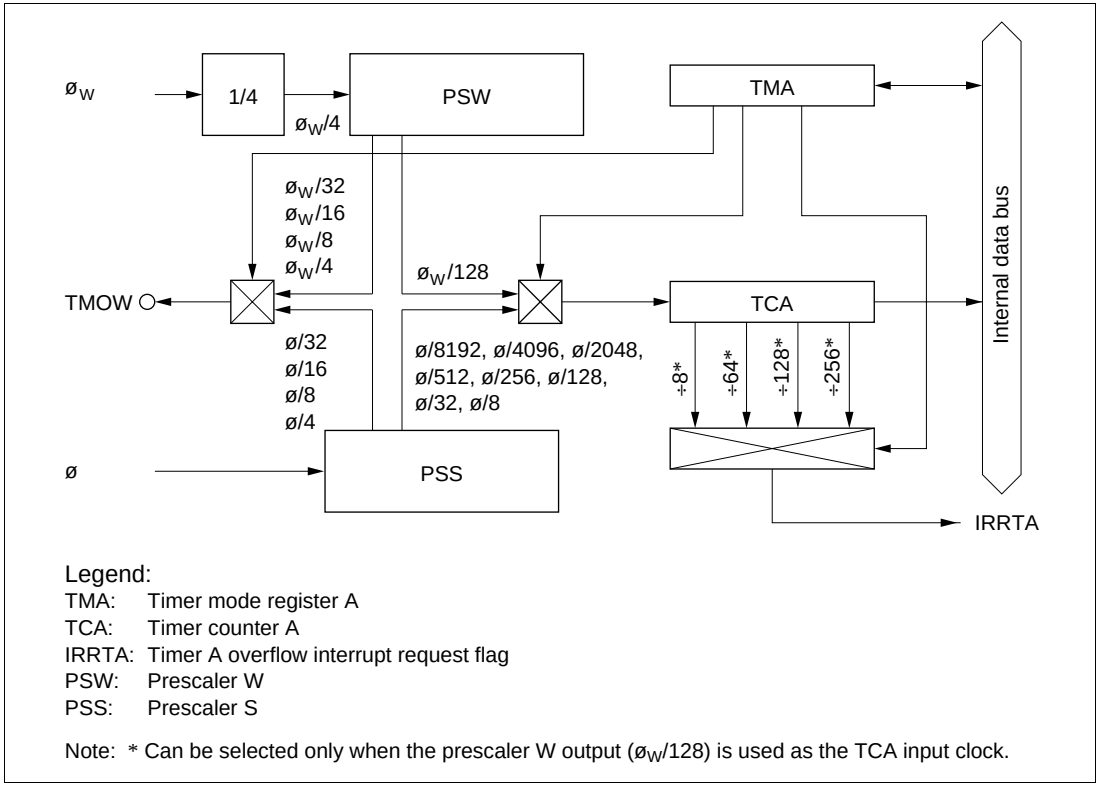


Figure 9.1 Block Diagram of Timer A

Pin Configuration

Table 9.2 shows the timer A pin configuration.

Table 9.2 Pin Configuration

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

Register Configuration

Table 9.3 shows the register configuration of timer A.

Table 9.3 Timer A Registers

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

9.2.2 Register Descriptions

Timer Mode Register A (TMA)

Bit	7	6	5	4	3	2	1	0
	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W

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

Upon reset, TMA is initialized to H'10.

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

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

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

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

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

Timer Counter A (TCA)

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

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

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

Upon reset, TCA is initialized to H'00.

9.2.3 Timer Operation

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

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

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

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

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

Real-Time Clock Time Base Operation: When bit TMA3 in TMA is set to 1, timer A functions as a real-time clock time base by counting clock signals output by prescaler W. The overflow period of timer A is set by bits TMA1 and TMA0 in TMA. A choice of four periods is available. In time base operation (TMA3 = 1), setting bit TMA2 to 1 clears both TCA and prescaler W to their initial values of H'00.

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

9.2.4 Timer A Operation States

Table 9.4 summarizes the timer A operation states.

Table 9.4 Timer A Operation States

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

Note: When 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.

9.3 Timer B1

9.3.1 Overview

Timer B1 is an 8-bit timer that increments each time a clock pulse is input. This timer has two operation modes, interval and auto reload.

Features

Features of timer B1 are given below.

- Choice of seven internal clock sources (ø/8192, ø/2048, ø/512, ø/256, ø/64, ø/16, ø/4) or an external clock (can be used to count external events).
- An interrupt is requested when the counter overflows.

Block Diagram

Figure 9.2 shows a block diagram of timer B1.

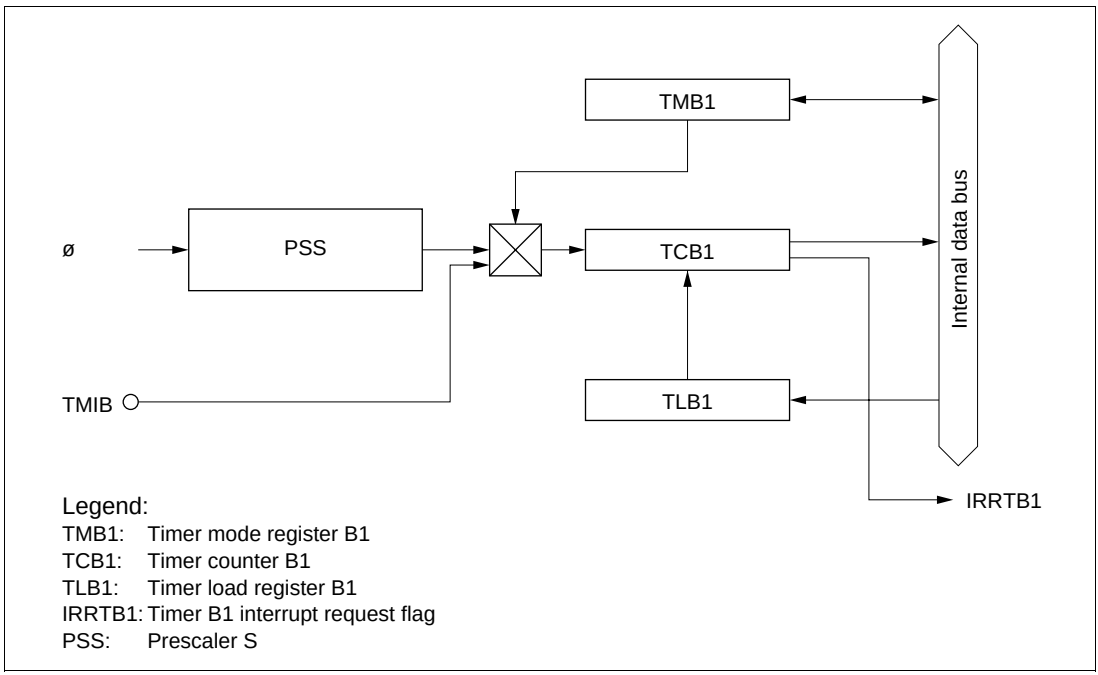


Figure 9.2 Block Diagram of Timer B1

Pin Configuration

Table 9.5 shows the timer B1 pin configuration.

Table 9.5 Pin Configuration

Name	Abbrev.	I/O	Function
Timer B1 event input	TMIB	Input	Event input to TCB1

Register Configuration

Table 9.6 shows the register configuration of timer B1.

Table 9.6 Timer B1 Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register B1	TMB1	R/W	H'78	H'FFB2
Timer counter B1	TCB1	R	H'00	H'FFB3
Timer load register B1	TLB1	W	H'00	H'FFB3

9.3.2 Register Descriptions

Timer Mode Register B1 (TMB1)

Bit	7	6	5	4	3	2	1	0
	TMB17	—	—	—	—	TMB12	TMB11	TMB10
Initial value	0	1	1	1	1	0	0	0
Read/Write	R/W	—	—	—	—	R/W	R/W	R/W

TMB1 is an 8-bit read/write register for selecting the auto-reload function and input clock.

Upon reset, TMB1 is initialized to H'78.

Bit 7—Auto-Reload Function Select (TMB17): Bit 7 selects whether timer B1 is used as an interval timer or auto-reload timer.

Bit 7: TMB17	Description	
0	Interval timer function selected	(initial value)
1	Auto-reload function selected	

Bits 6 to 3—Reserved Bits: Bits 6 to 3 are reserved; they are always read as 1, and cannot be modified.

Bits 2 to 0—Clock Select (TMB12 to TMB10): Bits 2 to 0 select the clock input to TCB1. For external event counting, either the rising or falling edge can be selected.

Bit 2: TMB12	Bit 1: TMB11	Bit 0: TMB10	Description
0	0	0	Internal clock: ø/8192 (initial value)
		1	Internal clock: ø/2048
	1	0	Internal clock: ø/512
		1	Internal clock: ø/256
1	0	0	Internal clock: ø/64
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	External event (TMIB): rising or falling edge*

Note: * The edge of the external event signal is selected by bit INTEG6 in interrupt edge select register 2 (IEGR2). See 3.3.2, Interrupt Control Registers, for details.

Timer Counter B1 (TCB1)

Bit	7	6	5	4	3	2	1	0
	TCB17	TCB16	TCB15	TCB14	TCB13	TCB12	TCB11	TCB10
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TCB1 is an 8-bit read-only up-counter, which is incremented by internal clock or external event input. The clock source for input to this counter is selected by bits TMB12 to TMB10 in timer mode register B1 (TMB1). TCB1 values can be read by the CPU at any time.

When TCB1 overflows from H'FF to H'00 or to the value set in TLB1, the IRRTB1 bit in IRR1 is set to 1.

TCB1 is allocated to the same address as TLB1.

Upon reset, TCB1 is initialized to H'00.

Timer Load Register B1 (TLB1)

Bit	7	6	5	4	3	2	1	0
	TLB17	TLB16	TLB15	TLB14	TLB13	TLB12	TLB11	TLB10
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

TLB1 is an 8-bit write-only register for setting the reload value of timer counter B1 (TCB1).

When a reload value is set in TLB1, the same value is loaded into timer counter B1 (TCB1) as well, and TCB1 starts counting up from that value. When TCB1 overflows during operation in auto-reload mode, the TLB1 value is loaded into TCB1. Accordingly, overflow periods can be set within the range of 1 to 256 input clocks.

The same address is allocated to TLB1 as to TCB1.

Upon reset, TLB1 is initialized to H'00.

9.3.3 Timer Operation

Interval Timer Operation: When bit TMB17 in timer mode register B1 (TMB1) is cleared to 0, timer B1 functions as an 8-bit interval timer.

Upon reset, TCB1 is cleared to H'00 and bit TMB17 is cleared to 0, so up-counting and interval timing resume immediately. The clock input to timer B1 is selected from seven internal clock signals output by prescaler S, or an external clock input at pin TMIB. The selection is made by bits TMB12 to TMB10 of TMB1.

After the count value in TCB1 reaches H'FF, the next clock signal input causes timer B1 to overflow, setting bit IRRTB1 to 1 in interrupt request register 1 (IRR1). If IENTB1 = 1 in interrupt enable register 1 (IENR1), a CPU interrupt is requested.*

At overflow, TCB1 returns to H'00 and starts counting up again.

During interval timer operation (TMB17 = 0), when a value is set in timer load register B1 (TLB1), the same value is set in TCB1.

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

Auto-Reload Timer Operation: Setting bit TMB17 in TMB1 to 1 causes timer B1 to function as an 8-bit auto-reload timer. When a reload value is set in TLB1, the same value is loaded into TCB1, becoming the value from which TCB1 starts its count.

After the count value in TCB1 reaches H'FF, the next clock signal input causes timer B1 to overflow. The TLB1 value is then loaded into TCB1, and the count continues from that value. The overflow period can be set within a range from 1 to 256 input clocks, depending on the TLB1 value.

The clock sources and interrupts in auto-reload mode are the same as in interval mode.

In auto-reload mode (TMB17 = 1), when a new value is set in TLB1, the TLB1 value is also set in TCB1.

Event Counter Operation: Timer B1 can operate as an event counter, counting rising or falling edges of an external event signal input at pin TMIB. External event counting is selected by setting bits TMB12 to TMB10 in timer mode register B1 (TMB1) to all 1s (111).

When timer B1 is used to count external event input, bit INTEN6 in IENR3 should be cleared to 0 to disable INT_6 interrupt requests.

9.3.4 Timer B1 Operation States

Table 9.7 summarizes the timer B1 operation states.

Table 9.7 Timer B1 Operation States

Operation Mode		Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCB1	Interval	Reset	Functions	Functions	Halted	Halted	Halted	Halted
	Auto reload	Reset	Functions	Functions	Halted	Halted	Halted	Halted
TMB1		Reset	Functions	Retained	Retained	Retained	Retained	Retained

9.4 Timer V

9.4.1 Overview

Timer V is an 8-bit timer based on an 8-bit counter. Timer V counts external events. Also compare match signals can be used to reset the counter, request an interrupt, or output a pulse signal with an arbitrary duty cycle. Counting can be initiated by a trigger input at the TRGV pin, enabling pulse output control to be synchronized to the trigger, with an arbitrary delay from the trigger input. The trigger input signal is shared with the realtime port.

Features

Features of timer V are given below.

- Choice of six internal clock sources (ø/128, ø/64, ø/32, ø/16, ø/8, ø/4) or an external clock (can be used as an external event counter).
- Counter can be cleared by compare match A or B, or by an external reset signal. If the count stop function is selected, the counter can be halted when cleared.
- Timer output is controlled by two independent compare match signals, enabling pulse output with an arbitrary duty cycle, PWM output, and other applications.
- Three interrupt sources: two compare match, one overflow
- Counting can be initiated by trigger input at the TRGV pin. The rising edge, falling edge, or both edges of the TRGV input can be selected.

Block Diagram

Figure 9.3 shows a block diagram of timer V.

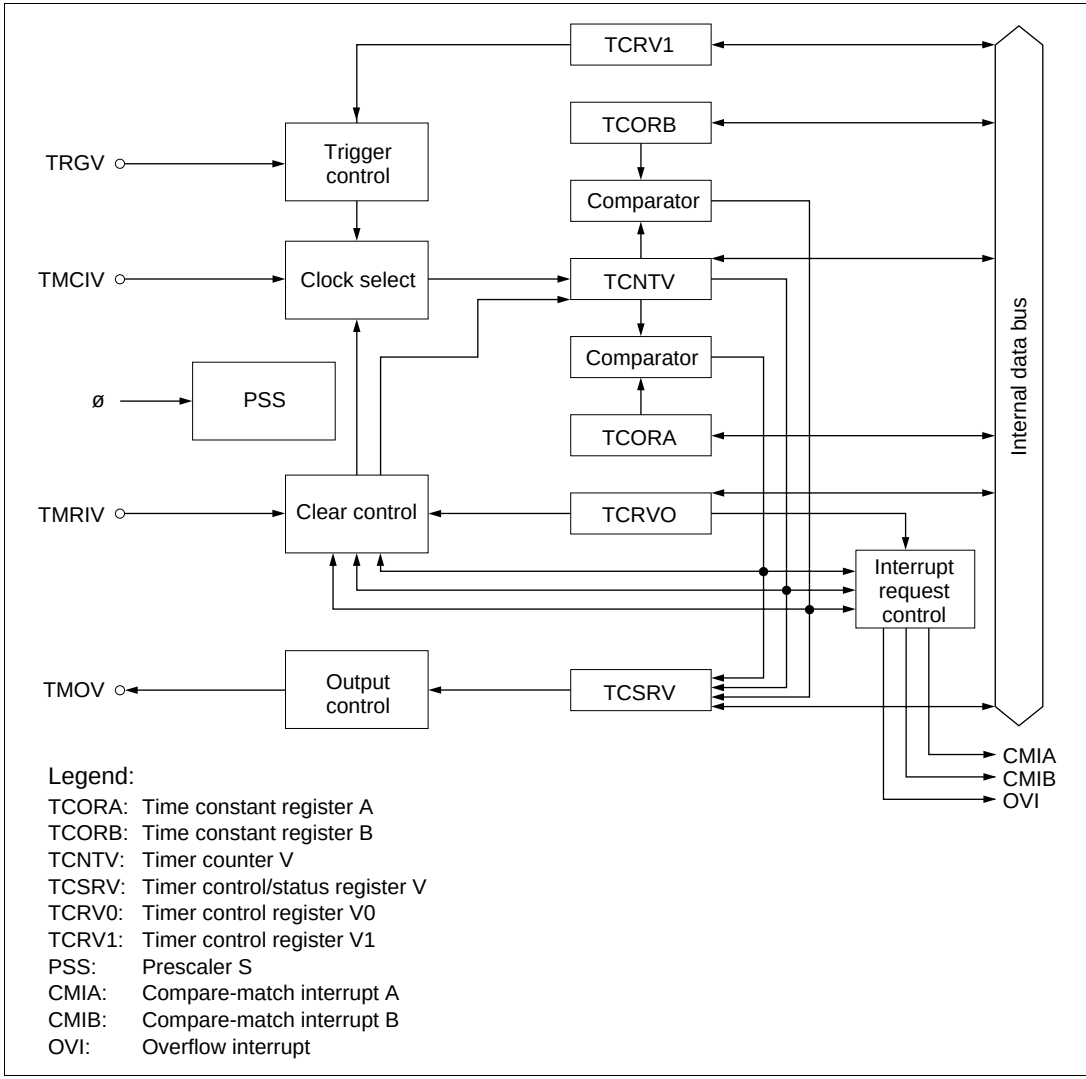


Figure 9.3 Block Diagram of Timer V

Pin Configuration

Table 9.8 shows the timer V pin configuration.

Table 9.8 Pin Configuration

Name	Abbrev.	I/O	Function
Timer V output	TMOV	Output	Timer V waveform output
Timer V clock input	TMCIV	Input	Clock input to TCNTV
Timer V reset input	TMRIV	Input	External input to reset TCNTV
Trigger input	TRGV	Input	Trigger input to initiate counting

Register Configuration

Table 9.9 shows the register configuration of timer V.

Table 9.9 Timer V Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer control register V0	TCRV0	R/W	H'00	H'FFB8
Timer control/status register V	TCSRV	R/(W)*	H'10	H'FFB9
Time constant register A	TCORA	R/W	H'FF	H'FFBA
Time constant register B	TCORB	R/W	H'FF	H'FFBB
Timer counter V	TCNTV	R/W	H'00	H'FFBC
Timer control register V1	TCRV1	R/W	H'E2	H'FFBD

Note: * Bits 7 to 5 can only be written with 0, for flag clearing.

9.4.2 Register Descriptions

Timer Counter V (TCNTV)

Bit	7	6	5	4	3	2	1	0
	$TCNTV_7$	$TCNTV_6$	$TCNTV_5$	$TCNTV_4$	$TCNTV_3$	$TCNTV_2$	$TCNTV_1$	$TCNTV_0$
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

TCNTV is an 8-bit read/write up-counter which is incremented by internal or external clock input. The clock source is selected by bits CKS2 to CKS0 in TCRV0. The TCNTV value can be read and written by the CPU at any time. TCNTV can be cleared by an external reset signal, or by compare match A or B. The clearing signal is selected by bits CCLR1 and CCLR0 in TCRV0.

When TCNTV overflows from H'FF to H'00, OVF is set to 1 in TCSRV.

TCNTV is initialized to H'00 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Time Constant Registers A and B (TCORA, TCORB)

Bit	7	6	5	4	3	2	1	0
	$TCORn_7$	$TCORn_6$	$TCORn_5$	$TCORn_4$	$TCORn_3$	$TCORn_2$	$TCORn_1$	$TCORn_0$
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

n = A or B

TCORA and TCORB are 8-bit read/write registers.

TCORA and TCNTV are compared at all times, except during the T_3 state of a TCORA write cycle. When the TCORA and TCNTV contents match, CMFA is set to 1 in TCSRV. If CMIEA is also set to 1 in TCRV0, a CPU interrupt is requested.

Timer output from the TMOV pin can be controlled by a signal resulting from compare match, according to the settings of bits OS3 to OS0 in TCSRV.

TCORA is initialized to H'FF upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

TCORB is similar to TCORA.

Timer Control Register V0 (TCRV0)

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

TCRV0 is an 8-bit read/write register that selects the TCNTV input clock, controls the clearing of TCNTV, and enables interrupts.

TCRV0 is initialized to H'00 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Bit 7—Compare Match Interrupt Enable B (CMIEB): Bit 7 enables or disables the interrupt request (CMIB) generated from CMFB when CMFB is set to 1 in TCSRV.

Bit 7: CMIEB	Description	
0	Interrupt request (CMIB) from CMFB disabled	(initial value)
1	Interrupt request (CMIB) from CMFB enabled	

Bit 6—Compare Match Interrupt Enable A (CMIEA): Bit 6 enables or disables the interrupt request (CMIA) generated from CMFA when CMFA is set to 1 in TCSRV.

Bit 6: CMIEA	Description	
0	Interrupt request (CMIA) from CMFA disabled	(initial value)
1	Interrupt request (CMIA) from CMFA enabled	

Bit 5—Timer Overflow Interrupt Enable (OVIE): Bit 5 enables or disables the interrupt request (OVI) generated from OVF when OVF is set to 1 in TCSRV.

Bit 5: OVIE	Description	
0	Interrupt request (OVI) from OVF disabled	(initial value)
1	Interrupt request (OVI) from OVF enabled	

Bits 4 and 3—Counter Clear 1 and 0 (CCLR1, CCLR0): Bits 4 and 3 specify whether or not to clear TCNTV, and select compare match A or B or an external reset input.

When clearing is specified, if TRGE is set to 1 in TCRV1, then when TCNTV is cleared it is also halted. Counting resumes when a trigger edge is input at the TRGV pin.

If TRGE is cleared to 0, after TCNTV is cleared it continues counting up.

Bit 4: CCLR1	Bit 3: CCLR0	Description	
0	0	Clearing is disabled	(initial value)
	1	Cleared by compare match A	
1	0	Cleared by compare match B	
	1	Cleared by rising edge of external reset input	

Bits 2 to 0—Clock Select 2 to 0 (CKS2 to CKS0): Bits 2 to 0 and bit ICKS0 in TCRV1 select the clock input to TCNTV.

Six internal clock sources divided from the system clock (ø) can be selected. The counter increments on the falling edge.

If the external clock is selected, there is a further selection of incrementing on the rising edge, falling edge, or both edges.

If TRGE is cleared to 0, after TCNTV is cleared it continues counting up.

TCRV0			TCRV1		
Bit 2: CKS2	Bit 1: CKS1	Bit 0: CKS0	Bit 0: ICKS0	Description	
0	0	0	—	Clock input disabled	(initial value)
		1	0	Internal clock: ø/4, falling edge	
			1	Internal clock: ø/8, falling edge	
	1	0	0	Internal clock: ø/16, falling edge	
			1	Internal clock: ø/32, falling edge	
		1	0	Internal clock: ø/64, falling edge	
			1	Internal clock: ø/128, falling edge	
1	0	0	—	Clock input disabled	
		1	—	External clock: rising edge	
	1	0	—	External clock: falling edge	
		1	—	External clock: rising and falling edges	

Timer Control/Status Register V (TCSRV)

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

Note: * Bits 7 to 5 can be only written with 0, for flag clearing.

TCSRV is an 8-bit register that sets compare match flags and the timer overflow flag, and controls compare match output.

TCSRV is initialized to H'10 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Bit 7—Compare Match Flag B (CMFB): Bit 7 is a status flag indicating that TCNTV has matched TCORB. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 7: CMFB	Description	
0	Clearing conditions: After reading CMFB = 1, cleared by writing 0 to CMFB	(initial value)
1	Setting conditions: Set when the TCNTV value matches the TCORB value	

Bit 6—Compare Match Flag A (CMFA): Bit 6 is a status flag indicating that TCNTV has matched TCORA. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 6: CMFA	Description	
0	Clearing conditions: After reading CMFA = 1, cleared by writing 0 to CMFA	(initial value)
1	Setting conditions: Set when the TCNTV value matches the TCORA value	

Bit 5—Timer Overflow Flag (OVF): Bit 5 is a status flag indicating that TCNTV 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 5: OVF	Description	
0	Clearing conditions: After reading OVF = 1, cleared by writing 0 to OVF	(initial value)
1	Setting conditions: Set when TCNTV overflows from H'FF to H'00	

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

Bits 3 to 0—Output Select 3 to 0 (OS3 to OS0): Bits 3 to 0 select the way in which the output level at the TMOV pin changes in response to compare match between TCNTV and TCORA or TCORB.

OS3 and OS2 select the output level for compare match B. OS1 and OS0 select the output level for compare match A. The two levels can be controlled independently.

If two compare matches occur simultaneously, any conflict between the settings is resolved according to the following priority order: toggle output > 1 output > 0 output.

When OS3 to OS0 are all cleared to 0, timer output is disabled.

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

Bit 3: OS3	Bit 2: OS2	Description	
0	0	No change at compare match B	(initial value)
	1	0 output at compare match B	
1	0	1 output at compare match B	
	1	Output toggles at compare match B	

Bit 1: OS1	Bit 0: OS0	Description	
0	0	No change at compare match A	(initial value)
	1	0 output at compare match A	
1	0	1 output at compare match A	
	1	Output toggles at compare match A	

Timer Control Register V1 (TCRV1)

Bit	7	6	5	4	3	2	1	0
	—	—	—	TVEG1	TVEG0	TRGE	—	ICKS0
Initial value	1	1	1	0	0	0	1	0
Read/Write	—	—	—	R/W	R/W	R/W	—	R/W

TCRV1 is an 8-bit read/write register that selects the valid edge at the TRGV pin, enables TRGV input, and selects the clock input to TCNTV.

TCRV1 is initialized to H'E2 upon reset and in watch mode, subsleep mode, and subactive mode.

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

Bits 4 and 3—TRGV Input Edge Select (TVEG1, TVEG0): Bits 4 and 3 select the TRGV input edge.

Bit 4: TVEG1	Bit 3: TVEG0	Description
0	0	TRGV trigger input is disabled (initial value)
	1	Rising edge is selected
1	0	Falling edge is selected
	1	Rising and falling edges are both selected

Bit 2—TRGV Input Enable (TRGE): Bit 2 enables TCNTV counting to be triggered by input at the TRGV pin, and enables TCNTV counting to be halted when TCNTV is cleared by compare match. TCNTV stops counting when TRGE is set to 1, then starts counting when the edge selected by bits TVEG1 and TVEG0 is input at the TRGV pin.

Bit 2: TRGE	Description
0	TCNTV counting is not triggered by input at the TRGV pin, and does not stop when TCNTV is cleared by compare match (initial value)
1	TCNTV counting is triggered by input at the TRGV pin, and stops when TCNTV is cleared by compare match

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

Bit 0—Internal Clock Select 0 (ICKS0): Bit 0 and bits CKS2 to CKS0 in TCRV0 select the TCNTV clock source. For details see 9.4.2 Register Descriptions.

9.4.3 Timer Operation

Timer V Operation: A reset initializes TCNTV to H'00, TCORA and TCORB to H'FF, TCRV0 to H'00, TCSRV to H'10, and TCRV1 to H'E2.

Timer V can be clocked by one of six internal clocks output from prescaler S, or an external clock, as selected by bits CKS2 to CKS0 in TCRV0 and bit ICKS0 in TCRV1. The valid edge or edges of the external clock can also be selected by CKS2 to CKS0. When the clock source is selected, TCNTV starts counting the selected clock input.

The TCNTV contents are always compared with TCORA and TCORB. When a match occurs, the CMFA or CMFB bit is set to 1 in TCSRV. If CMIEA or CMIEB is set to 1 in TCRV0, a CPU interrupt is requested. At the same time, the output level selected by bits OS3 to OS0 in TCSRV is output from the TMOV pin.

When TCNT overflows from H'FF to H'00, if OVIE is 1 in TCRV0, a CPU interrupt is requested.

If bits CCLR1 and CCLR0 in TCRV0 are set to 01 (clear by compare match A) or 10 (clear by compare match B), TCNTV is cleared by the corresponding compare match. If these bits are set to 11, TCNTV is cleared by input of a rising edge at the TMRIV pin.

When the counter clear event selected by bits CCLR1 and CCLR0 in TCRV0 occurs, TCNTV is cleared and the count-up is halted. TCNTV starts counting when the signal edge selected by bits TVEG1 and TVEG0 in TCRV1 is input at the TRGV pin.

TCNTV Increment Timing: TCNTV is incremented by an input (internal or external) clock.

- Internal clock

  One of six clocks (ø/128, ø/64, ø/32, ø/16, ø/8, ø/4) divided from the system clock (ø) can be selected by bits CKS2 to CKS0 in TCRV0 and bit ICKS0 in TCRV1. Figure 9.4 shows the timing.

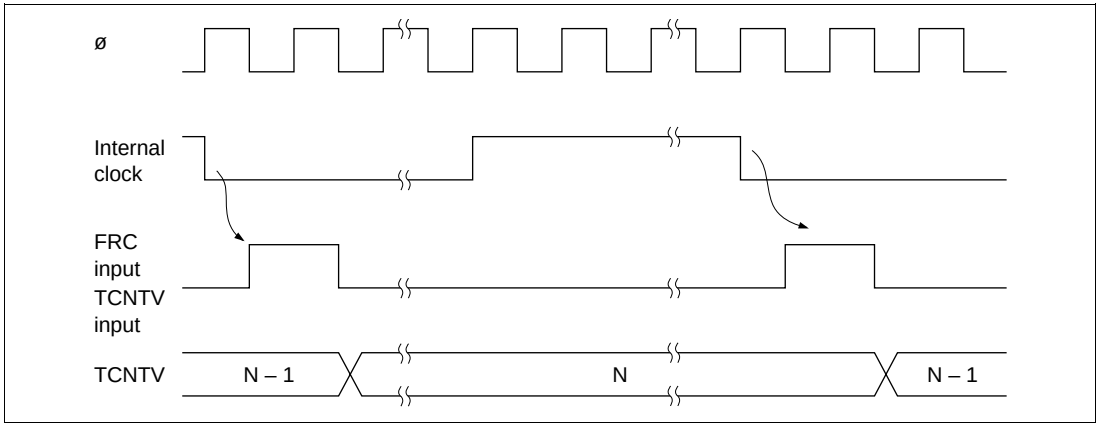


Figure 9.4 Increment Timing with Internal Clock

- External clock

  Incrementation on the rising edge, falling edge, or both edges of the external clock can be selected by bits CKS2 to CKS0 in TCRV0.

  The external clock pulse width should be at least 1.5 system clocks (ø) when a single edge is counted, and at least 2.5 system clocks when both edges are counted. Shorter pulses will not be counted correctly.

  Figure 9.5 shows the timing when both the rising and falling edges of the external clock are selected.

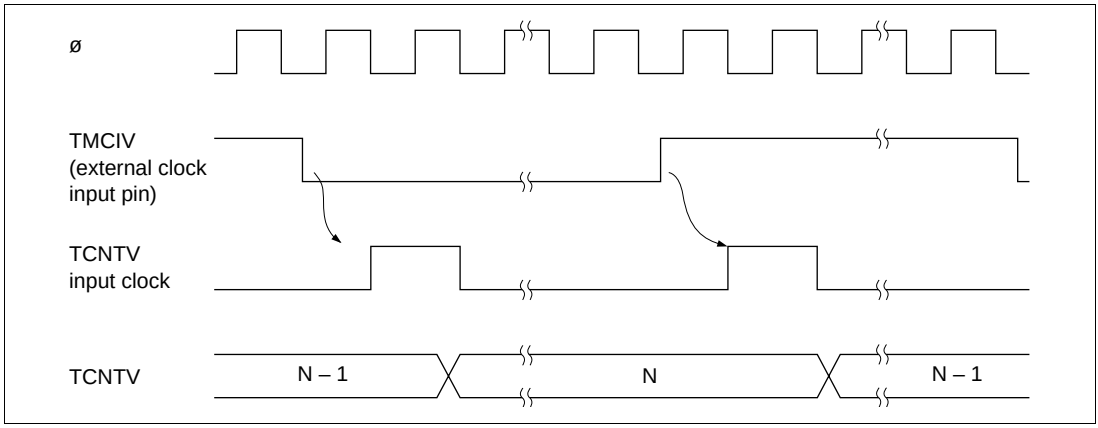


Figure 9.5 Increment Timing with External Clock

Overflow flag Set Timing: The overflow flag (OVF) is set to 1 when TCNTV overflows from H'FF to H'00. Figure 9.6 shows the timing.

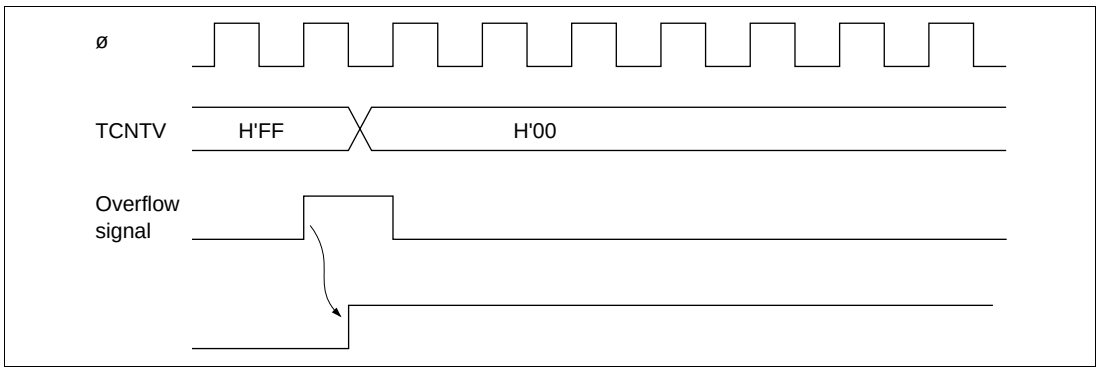


Figure 9.6 OVF Set Timing

Compare Match Flag set Timing: Compare match flag A or B (CMFA or CMFB) is set to 1 when TCNTV matches TCORA or TCORB. The internal compare-match signal is generated in the last state in which the values match (when TCNTV changes from the matching value to a new value). Accordingly, when TCNTV matches TCORA or TCORB, the compare match signal is not generated until the next clock input to TCNTV. Figure 9.7 shows the timing.

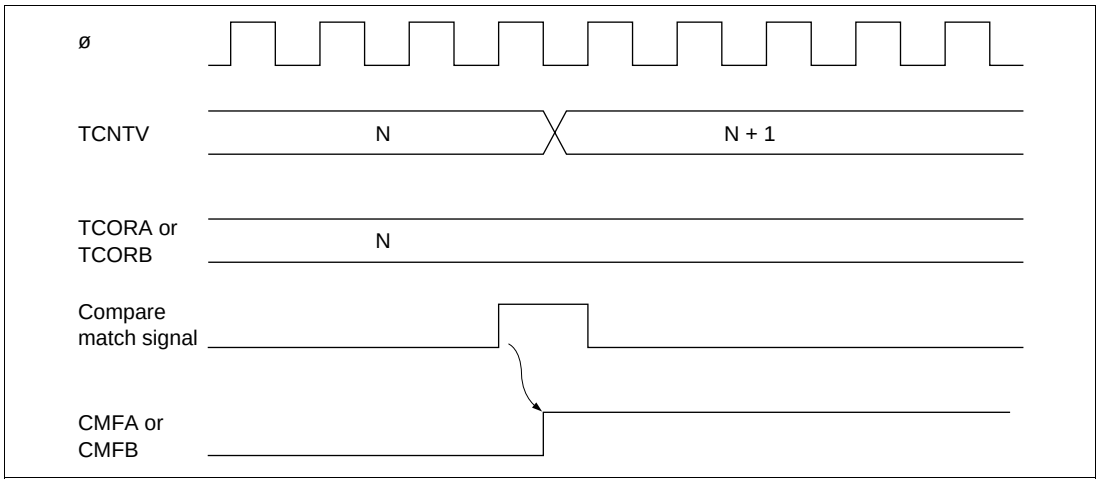


Figure 9.7 CMFA and CMFB Set Timing

TMOV Output Timing: The TMOV output responds to compare match A or B by remaining unchanged, changing to 0, changing to 1, or toggling, as selected by bits OS3 to OS0 in TCSRV. Figure 9.8 shows the timing when the output is toggled by compare match A.

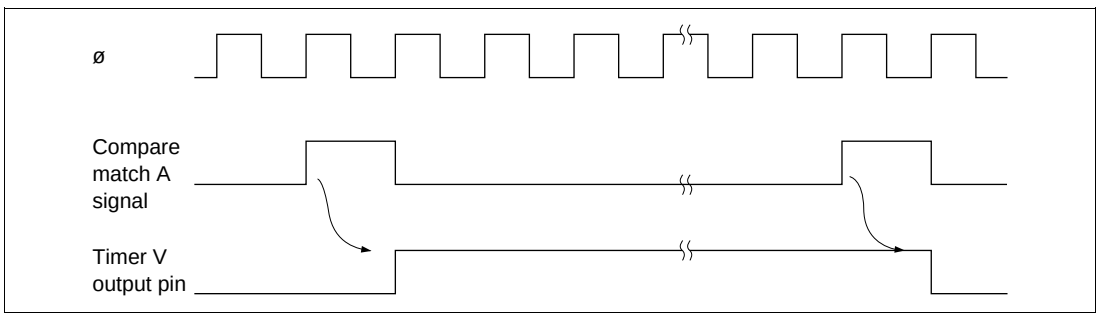


Figure 9.8 TMOV Output Timing

TCNTV Clear Timing by Compare Match: TCNTV can be cleared by compare match A or B, as selected by bits CCLR1 and CCLR0 in TCRV0. Figure 9.9 shows the timing.

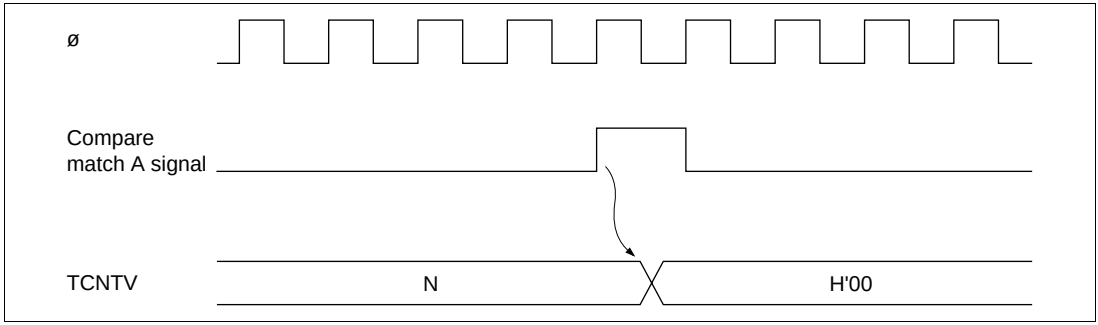


Figure 9.9 Clear Timing by Compare Match

TCNTV Clear Timing by TMRIV: TCNTV can be cleared by a rising edge at the TMRIV pin, as selected by bits CCLR1 and CCLR0 in TCRV0. A TMRIV input pulse width of at least 1.5 system clocks is necessary. Figure 9.10 shows the timing.

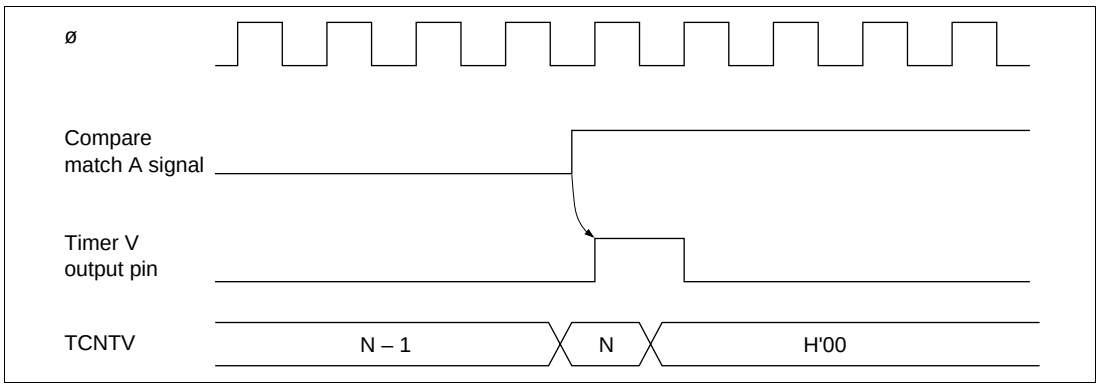


Figure 9.10 Clear Timing by TMRIV Input

9.4.4 Timer V Operation Modes

Table 9.10 summarizes the timer V operation states.

Table 9.10 Timer V Operation States

Operation Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCNTV	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TCRV0, TCRV1	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TCORA, TCORB	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TCSRV	Reset	Functions	Functions	Reset	Reset	Reset	Reset

9.4.5 Interrupt Sources

Timer V has three interrupt sources: CMIA, CMIB, and OVI. Table 9.11 lists the interrupt sources and their vector address. Each interrupt source can be enabled or disabled by an interrupt enable bit in TCRV0. Although all three interrupts share the same vector, they have individual interrupt flags, so software can discriminate the interrupt source.

Table 9.11 Timer V Interrupt Sources

Interrupt	Description	Vector Address
CMIA	Generated from CMFA	H'0022
CMIB	Generated from CMFB	
OVI	Generated from OVF	

9.4.6 Application Examples

Pulse Output with Arbitrary Duty Cycle: Figure 9.11 shows an example of output of pulses with an arbitrary duty cycle. To set up this output:

- Clear bit CCLR1 to 0 and set bit CCLR0 to 1 in TCRV0 so that TCNTV will be cleared by compare match with TCORA.
- Set bits OS3 to OS0 to 0110 in TCSRV so that the output will go to 1 at compare match with TCORA and to 0 at compare match with TCORB.
- Set bits CKS2 to CKS0 in TCRV0 and bit ICKS0 in TCRV1 to select the desired clock source.

With these settings, a waveform is output without further software intervention, with a period determined by TCORA and a pulse width determined by TCORB.

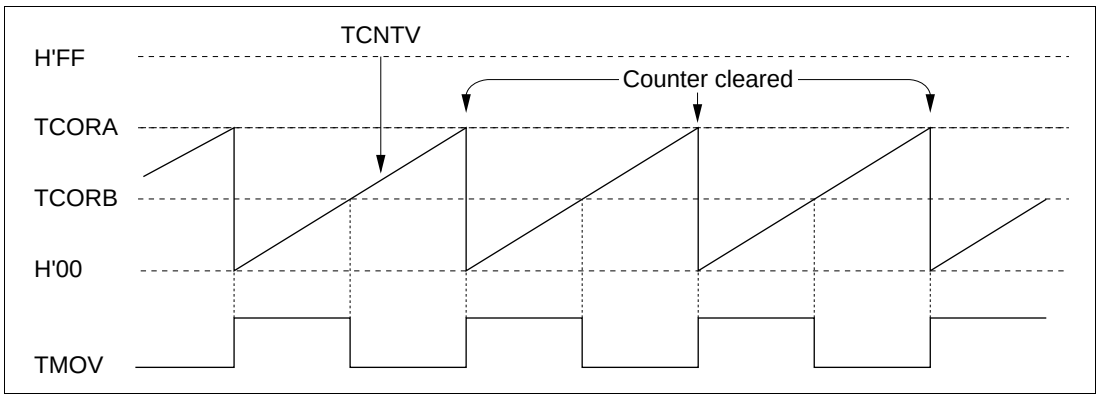


Figure 9.11 Pulse Output Example

Single-Shot Output with Arbitrary Pulse Width and Delay from TRGV Input: The trigger function can be used to output a pulse with an arbitrary pulse width at an arbitrary delay from the TRGV input, as shown in figure 9.12. To set up this output:

- Set bit CCLR1 to 1 and clear bit CCLR0 to 0 in TCRV0 so that TCNTV will be cleared by compare match with TCORB.
- Set bits OS3 to OS0 to 0110 in TCSRV so that the output will go to 1 at compare match with TCORA and to 0 at compare match with TCORB.
- Set bits TVEG1 and TVEG0 to 10 in TCRV1 and set TRGE to 1 to select the falling edge of the TRGV input.
- Set bits CKS2 to CKS0 in TCRV0 and bit ICKS0 in TCRV1 to select the desired clock source.

After these settings, a pulse waveform will be output without further software intervention, with a delay determined by TCORA from the TRGV input, and a pulse width determined by (TCORB – TCORA).

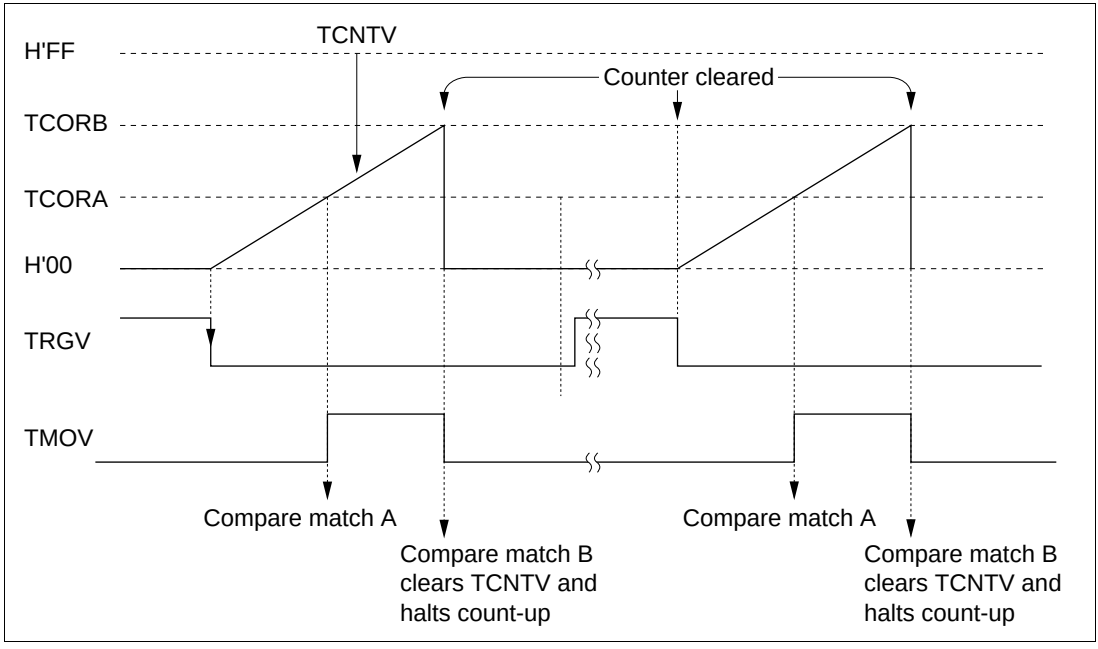


Figure 9.12 Pulse Output Synchronized to TRGV Input

9.4.7 Application Notes

The following types of contention can occur in timer V operation.

Contention between TCNTV Write and Counter Clear: If a TCNTV clear signal is generated in the T_3 state of a TCNTV write cycle, clearing takes precedence and the write to the counter is not carried out. Figure 9.13 shows the timing.

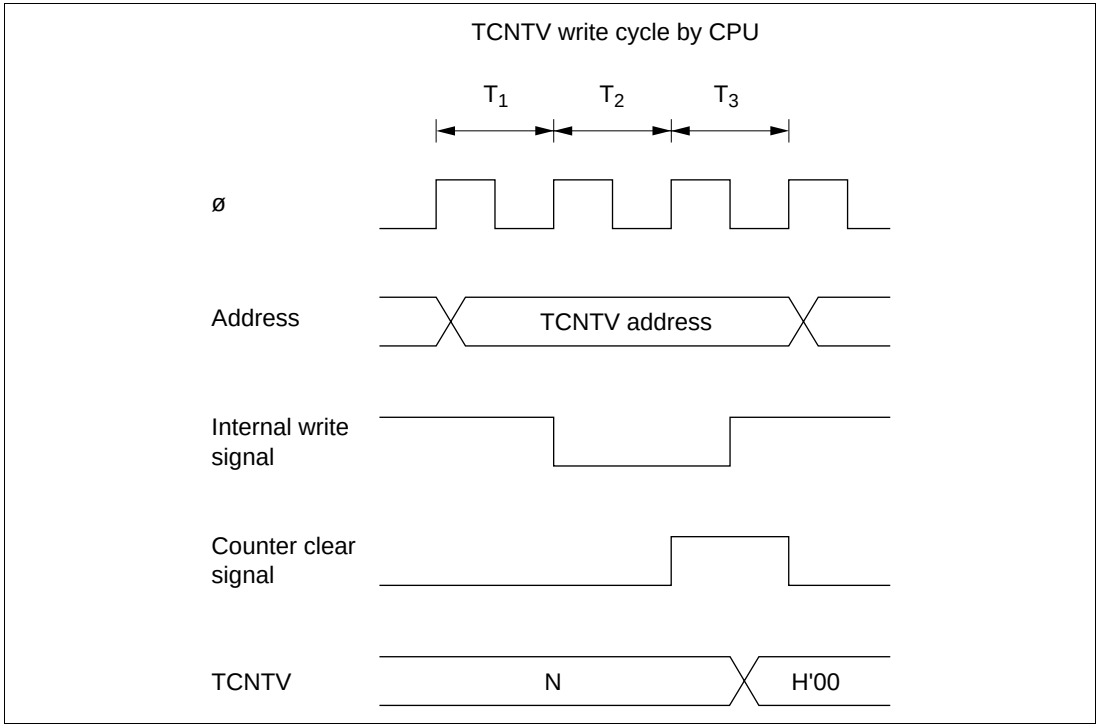


Figure 9.13 Contention between TCNTV Write and Clear

Contention between TCNTV Write and Increment: If a TCNTV increment clock signal is generated in the T_3 state of a TCNTV write cycle, the write takes precedence and the counter is not incremented. Figure 9.14 shows the timing.

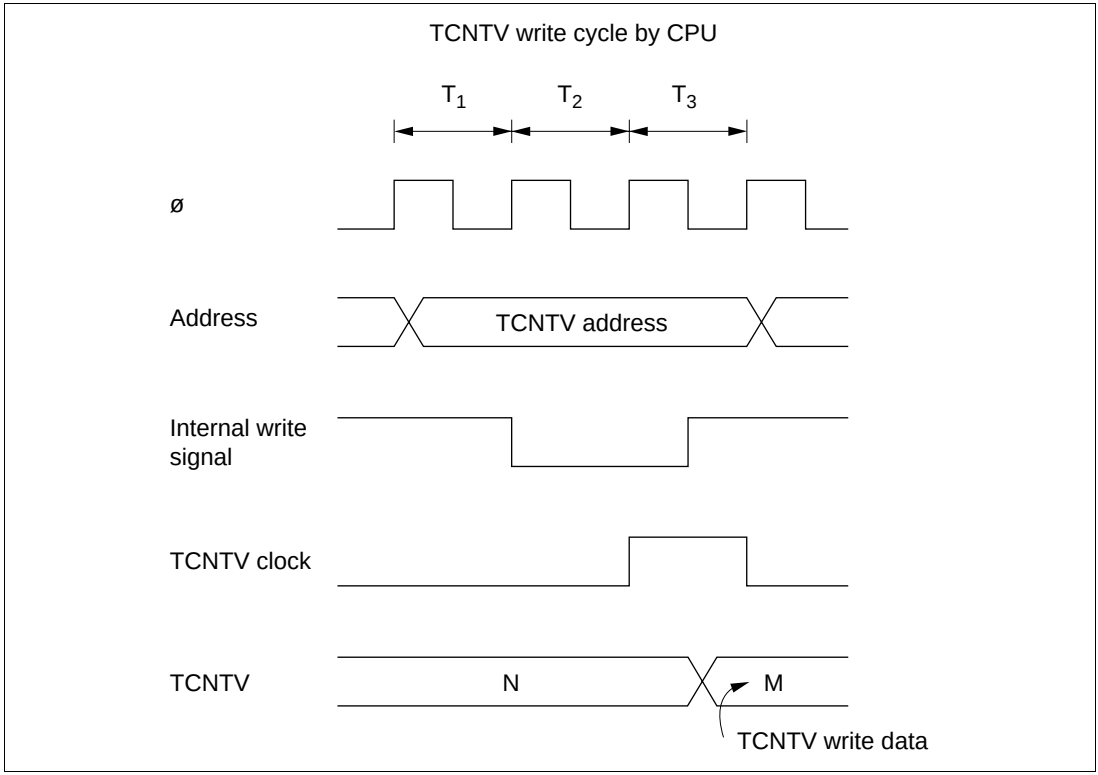


Figure 9.14 Contention between TCNTV Write and Increment

Contention between TCOR Write and Compare Match: If a compare match is generated in the T_3 state of a TCORA or TCORB write cycle, the write to TCORA or TCORB takes precedence and the compare match signal is inhibited. Figure 9.15 shows the timing.

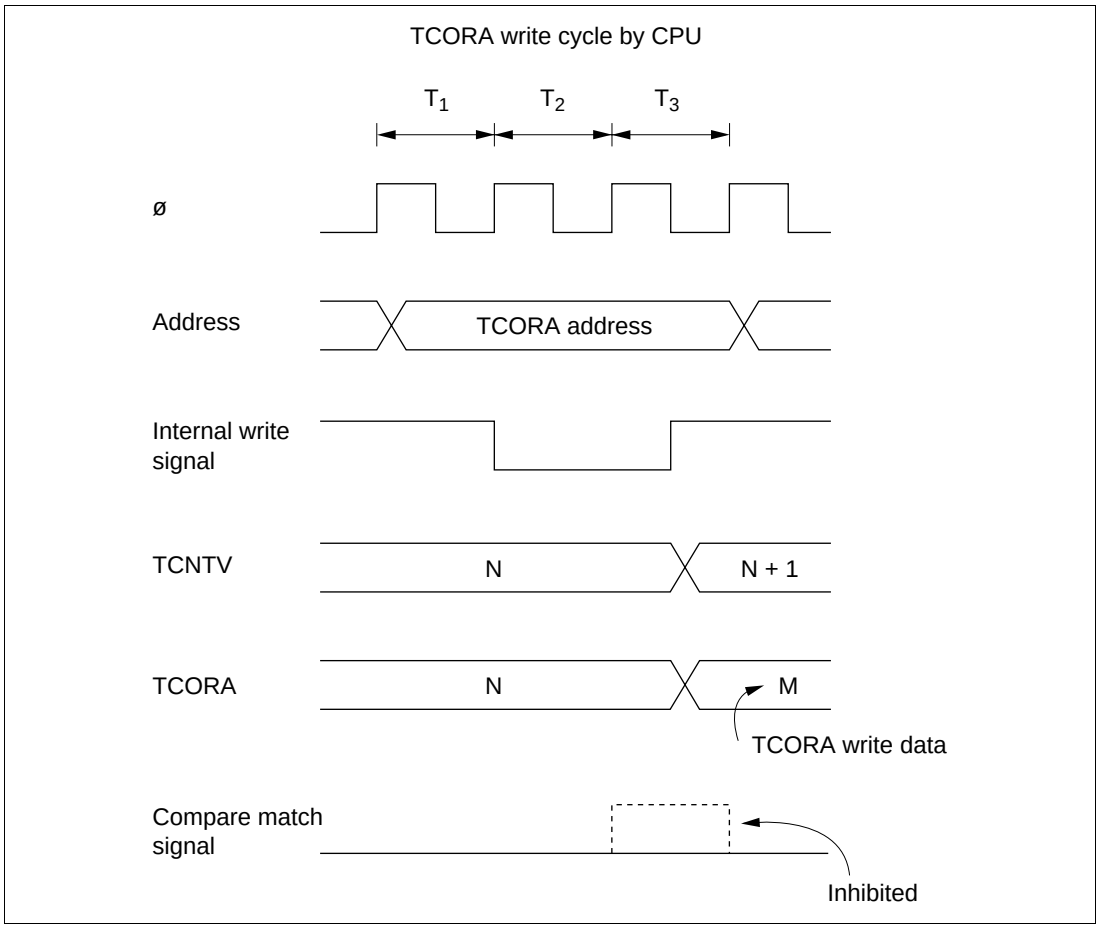


Figure 9.15 Contention between TCORA Write and Compare Match

Contention between Compare Match A and B: If compare match A and B occur simultaneously, any conflict between the output selections for compare match A and compare match B is resolved by following the priority order in table 9.12.

Table 9.12 Timer Output Priority Order

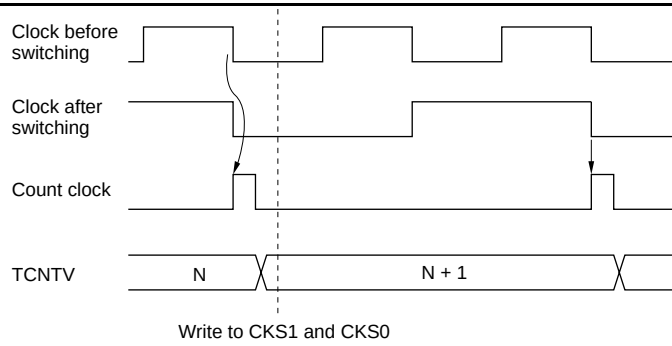
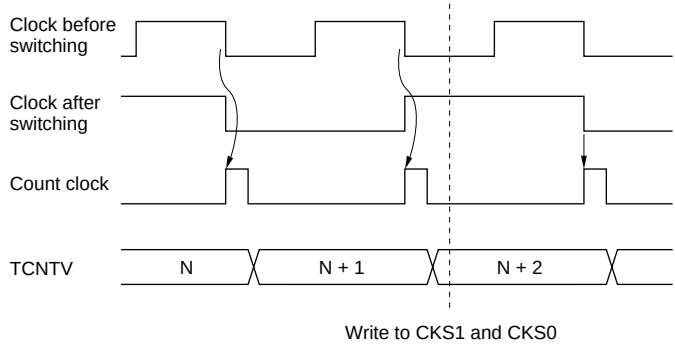
Output Setting	Priority
Toggle output	High
1 output	↑
0 output	
No change	Low

Internal Clock Switching and Counter Operation: Depending on the timing, TCNTV may be incremented by a switch between different internal clock sources. Table 9.13 shows the relation between internal clock switchover timing (by writing to bits CKS1 and CKS0) and TCNTV operation.

When TCNTV is internally clocked, an increment pulse is generated from the falling edge of an internal clock signal, which is divided from the system clock (ø). For this reason, in a case like No. 3 in table 9.13 where the switch is from a high clock signal to a low clock signal, the switchover is seen as a falling edge, causing TCNTV to increment.

TCNTV can also be incremented by a switch between internal and external clocks.

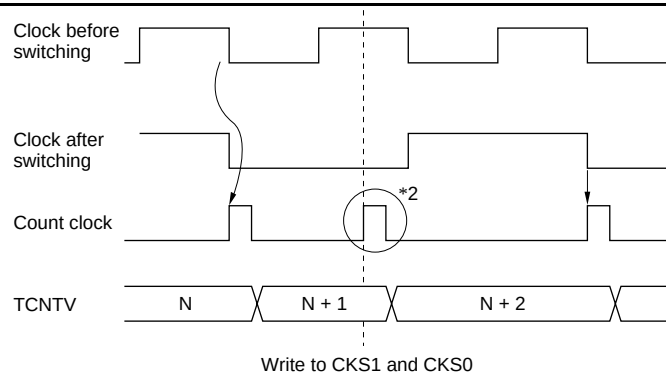
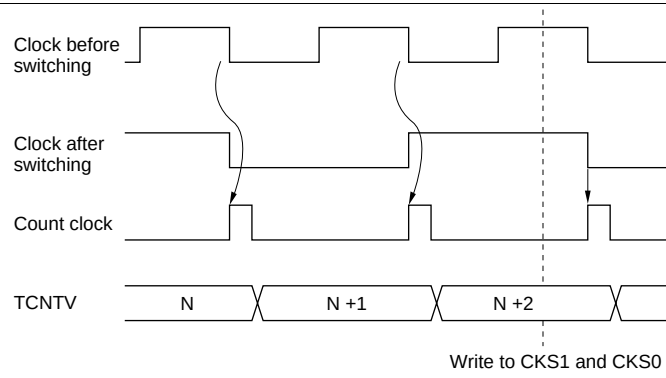
Table 9.13 Internal Clock Switching and TCNTV Operation

No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	TCNTV Operation
1	Goes from low level to low level*[1]	
2	Goes from low to high*[2]	

Notes: 1. Including a transition from the low level to the stopped state, or from the stopped state to the low level.

2. Including a transition from the stopped state to the high level.

Table 9.13 Internal Clock Switching and TCNTV Operation (cont)

No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	TCNTV Operation
3	Goes from high level to low level*[1]	
4	Goes from high to high	

Notes: 1. Including a transition from the high level to the stopped state.
2. The switchover is seen as a falling edge, and TCNTV is incremented.

9.5 Timer X

9.5.1 Overview

Timer X is based on a 16-bit free-running counter (FRC). It can output two independent waveforms, or measure input pulse widths and external clock periods.

Features

Features of timer X are given below.

- Choice of three internal clock sources (ø/2, ø/8, ø/32) or an external clock (can be used as an external event counter).
- Two independent output compare waveforms.
- Four independent input capture channels, with selection of rising or falling edge and buffering option.
- Counter can be cleared by compare match A.
- Seven independent interrupt sources: two compare match, four input capture, one overflow

Block Diagram

Figure 9.16 shows a block diagram of timer X.

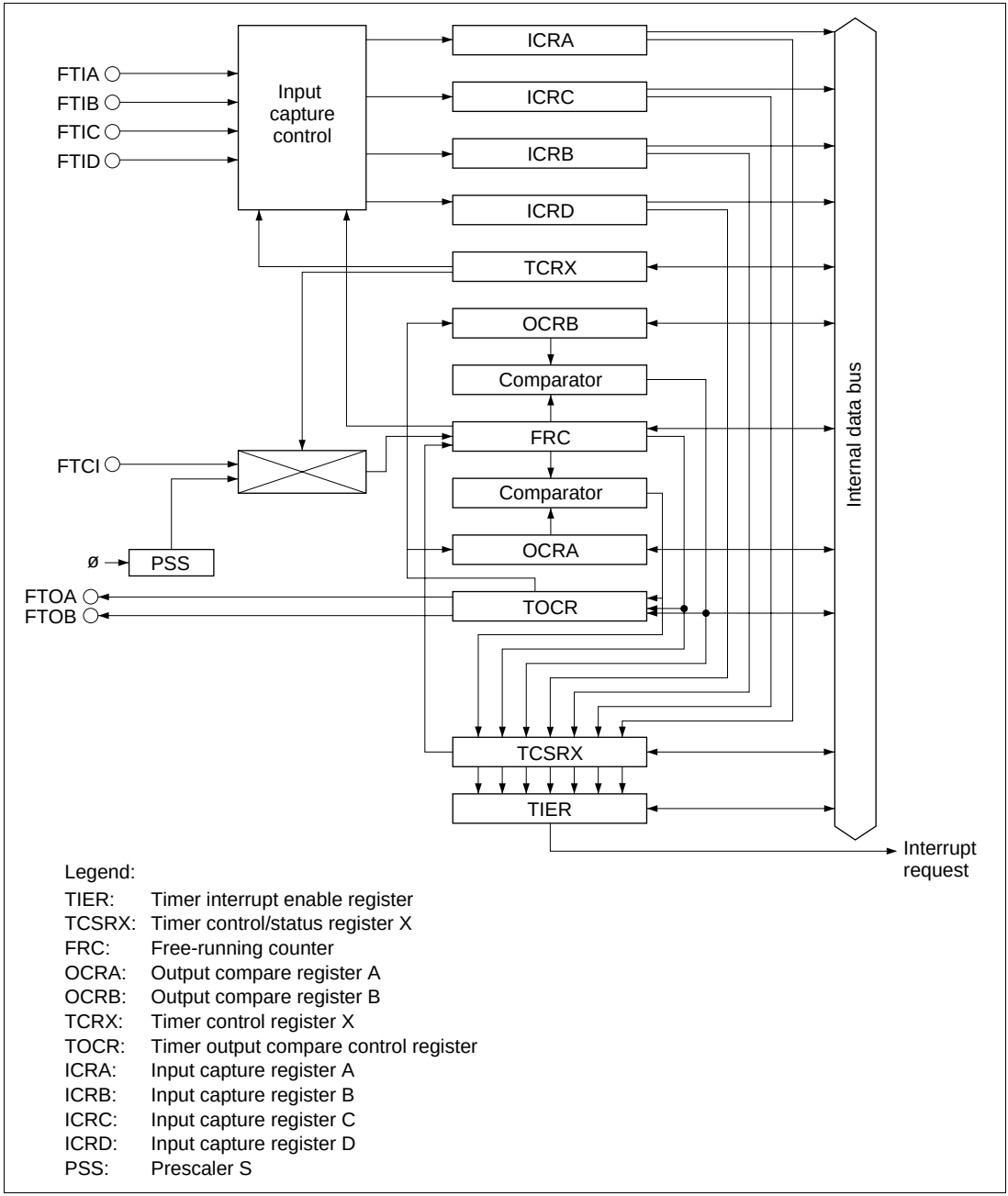


Figure 9.16 Block Diagram of Timer X

Pin Configuration

Table 9.14 shows the timer X pin configuration.

Table 9.14 Pin Configuration

Name	Abbrev.	I/O	Function
Counter clock input	FTCI	Input	Clock input to FRC
Output compare A	FTOA	Output	Output pin for output compare A
Output compare B	FTOB	Output	Output pin for output compare B
Input capture A	FTIA	Input	Input pin for input capture A
Input capture B	FTIB	Input	Input pin for input capture B
Input capture C	FTIC	Input	Input pin for input capture C
Input capture D	FTID	Input	Input pin for input capture D

Register Configuration

Table 9.15 shows the register configuration of timer X.

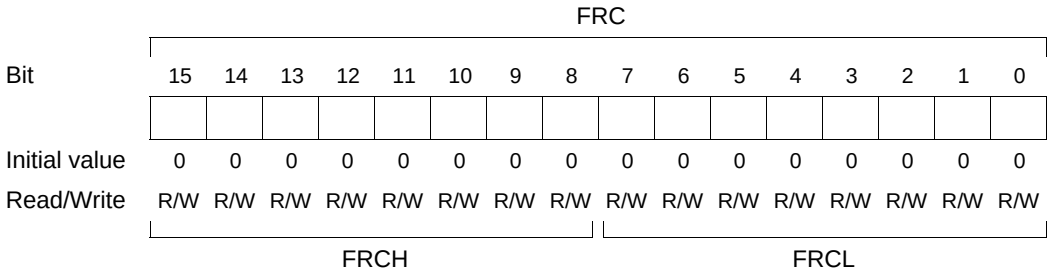
Table 9.15 Timer X Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer interrupt enable register	TIER	R/W	H'01	H'F770
Timer control/status register X	TCSRX	R/(W)*[1]	H'00	H'F771
Free-running counter H	FRCH	R/W	H'00	H'F772
Free-running counter L	FRCL	R/W	H'00	H'F773
Output compare register AH	OCRAH	R/W	H'FF	H'F774*[2]
Output compare register AL	OCRAL	R/W	H'FF	H'F775*[2]
Output compare register BH	OCRBH	R/W	H'FF	H'F774*[2]
Output compare register BL	OCRBL	R/W	H'FF	H'F775*[2]
Timer control register X	TCRX	R/W	H'00	H'F776
Timer output compare control register	TOCR	R/W	H'E0	H'F777
Input capture register AH	ICRAH	R	H'00	H'F778
Input capture register AL	ICRAL	R	H'00	H'F779
Input capture register BH	ICRBH	R	H'00	H'F77A
Input capture register BL	ICRBL	R	H'00	H'F77B
Input capture register CH	ICRCH	R	H'00	H'F77C
Input capture register CL	ICRCL	R	H'00	H'F77D
Input capture register DH	ICRDH	R	H'00	H'F77E
Input capture register DL	ICRDL	R	H'00	H'F77F

Notes: 1. Bits 7 to 1 can only be written with 0 for flag clearing. Bit 0 is a read/write bit.
2. OCRA and OCRB share the same address. They are selected by the OCRS bit in TOCR.

9.5.2 Register Descriptions

Free-Running Counter (FRC)
Free-Running Counter H (FRCH)
Free-Running Counter L (FRCL)



FRC is a 16-bit read/write up-counter, which is incremented by internal or external clock input. The clock source is selected by bits CKS1 and CKS0 in TCRX.

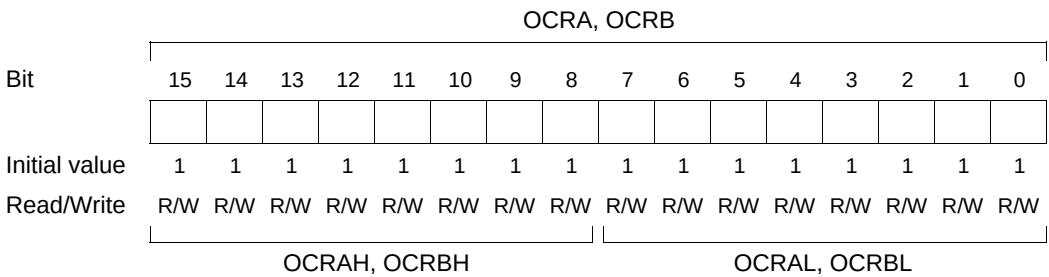
FRC can be cleared by compare match A, depending on the setting of CCLRA in TCSRX.

When FRC overflows from H'FFFF to H'0000, OVF is set to 1 in TCSRX. If OVIE = 1 in TIER, a CPU interrupt is requested.

FRC can be written and read by the CPU. Since FRC has 16 bits, data is transferred between the CPU and FRC via a temporary register (TEMP). For details see 9.5.3, CPU Interface.

FRC is initialized to H'0000 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Output Compare Registers A and B (OCRA, OCRB)
Output Compare Registers AH and BH (OCRAH, OCRBH)
Output Compare Registers AL and BL (OCRAL, OCRBL)



There are two 16-bit read/write output compare registers, OCRA and OCRB, the contents of which are always compared with FRC. When the values match, OCFA or OCFB is set to 1 in

TCSRX. If OCIAE = 1 or OCIBE = 1 in TIER, a CPU interrupt is requested.

When a compare match with OCRA or OCRB occurs, if OEA = 1 or OEB = 1 in TOCR, the value selected by OLVLA or OLVLB in TOCR is output at the FTOA or FTOB pin. After a reset, the output from the FTOA or FTOB pin is 0 until the first compare match occurs.

OCRA and OCRB can be written and read by the CPU. Since they are 16-bit registers, data is transferred between them and the CPU via a temporary register (TEMP). For details see 9.5.3, CPU Interface.

OCRA and OCRB are initialized to H'FFFF upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Input Capture Registers A to D (ICRA to ICRD)
Input Capture Registers AH to DH (ICRAH to ICRDH)
Input Capture Registers AL to DL (ICRAL to ICRDL)

ICRA, ICRB, ICRC, ICRD

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

ICRAH, ICRBH, ICRCH, ICRDH | ICRAL, ICRBL, ICRCL, ICRDL

There are four 16-bit read only input capture registers, ICRA to ICRD.

When the falling edge of an input capture signal is input, the FRC value is transferred to the corresponding input capture register, and the corresponding input capture flag (ICFA to ICFD) is set to 1 in TCSRX. If the corresponding input capture interrupt enable bit (ICIAE to ICIDE) is 1 in TCRX, a CPU interrupt is requested. The valid edge of the input signal can be selected by bits IEDGA to IEDGD in TCRX.

ICRC and ICRD can also be used as buffer registers for ICRA and ICRB. Buffering is enabled by bits BUFEA and BUFEB in TCRX.

Figure 9.17 shows the interconnections when ICRC operates as a buffer register of ICRA (when BUFEA = 1). When ICRC is used as the ICRA buffer, both the rising and falling edges of the external input signal can be selected simultaneously, by setting IEDGA ≠ IEDGC. If IEDGA = IEDGC, then only one edge is selected (either the rising edge or falling edge). See table 9.16.

Note: The FRC value is transferred to the input capture register (ICR) regardless of the value of the input capture flag (ICF).

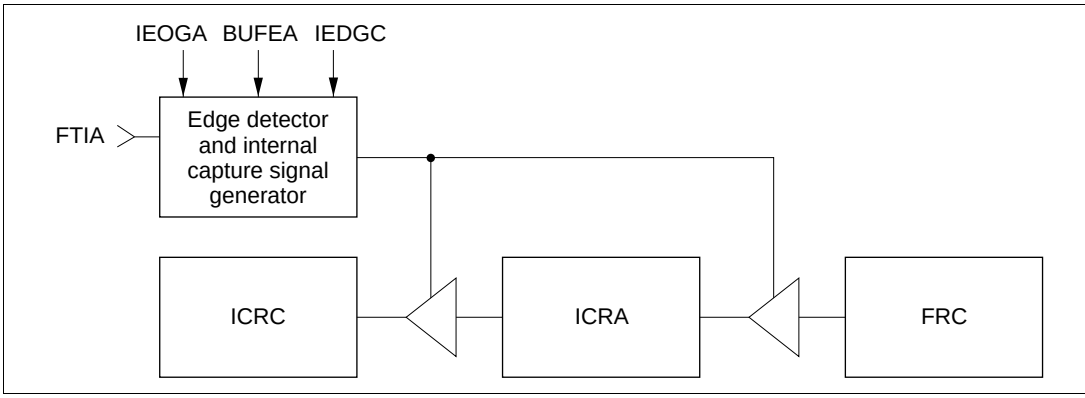


Figure 9.17 Buffer Operation (Example)

Table 9.16 Input Edge Selection during Buffer Operation

IEDGA	IEDGC	Input Edge Selection
0	0	Falling edge of input capture A input signal is captured (initial value)
	1	Rising and falling edge of input capture A input signal are both captured
1	0	
	1	Rising edge of input capture A input signal is captured

ICRA to ICRD can be written and read by the CPU. Since they are 16-bit registers, data is transferred from them to the CPU via a temporary register (TEMP). For details see 9.5.3, CPU Interface.

To assure input capture, the pulse width of the input capture input signal must be at least 1.5 system clocks (ø) when a single edge is selected, or at least 2.5 system clocks (ø) when both edges are selected.

ICRA to ICRD are initialized to H'0000 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Timer Interrupt Enable Register (TIER)

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

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

TIER is initialized to H'01 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

Bit 7—Input Capture Interrupt A Enable (ICIAE): Bit 7 enables or disables the ICIA interrupt requested when ICFA is set to 1 in TCSRX.

Bit 7: ICIAE	Description	
0	Interrupt request by ICFA (ICIA) is disabled	(initial value)
1	Interrupt request by ICFA (ICIA) is enabled	

Bit 6—Input Capture Interrupt B Enable (ICIBE): Bit 6 enables or disables the ICIB interrupt requested when ICFB is set to 1 in TCSRX.

Bit 6: ICIBE	Description	
0	Interrupt request by ICFB (ICIB) is disabled	(initial value)
1	Interrupt request by ICFB (ICIB) is enabled	

Bit 5—Input Capture Interrupt C Enable (ICICE): Bit 5 enables or disables the ICIC interrupt requested when ICFC is set to 1 in TCSRX.

Bit 5: ICICE	Description	
0	Interrupt request by ICFC (ICIC) is disabled	(initial value)
1	Interrupt request by ICFC (ICIC) is enabled	

Bit 4—Input Capture Interrupt D Enable (ICIDE): Bit 4 enables or disables the ICID interrupt requested when ICFD is set to 1 in TCSRX.

Bit 4: ICIDE	Description	
0	Interrupt request by ICFD (ICID) is disabled	(initial value)
1	Interrupt request by ICFD (ICID) is enabled	

Bit 3—Output Compare Interrupt A Enable (OCIAE): Bit 3 enables or disables the OCIA interrupt requested when OCFA is set to 1 in TCSRX.

Bit 3: OCIAE	Description	
0	Interrupt request by OCFA (OCIA) is disabled	(initial value)
1	Interrupt request by OCFA (OCIA) is enabled	

Bit 2—Output Compare Interrupt B Enable (OCIBE): Bit 2 enables or disables the OCIB interrupt requested when OCFB is set to 1 in TCSRX.

Bit 2: OCIBE	Description	
0	Interrupt request by OCFB (OCIB) is disabled	(initial value)
1	Interrupt request by OCFB (OCIB) is enabled	

Bit 1—Timer Overflow Interrupt Enable (OVIE): Bit 1 enables or disables the FOVI interrupt requested when OVF is set to 1 in TCSRX.

Bit 1: OVIE	Description	
0	Interrupt request by OVF (FOVI) is disabled	(initial value)
1	Interrupt request by OVF (FOVI) is enabled	

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

Timer Control/Status Register X (TCSRX)

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

Note: * Bits 7 to 1 can only be written with 0 for flag clearing.

TCSRX is an 8-bit register that selects clearing of the counter and controls interrupt request signals.

TCSRX is initialized to H'00 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode. Other timing is described in section 9.6.3, Timer Operation.

Bit 7—Input Capture Flag A (ICFA): Bit 7 is a status flag that indicates that the FRC value has been transferred to ICRA by an input capture signal. If BUFEA is set to 1 in TCRX, ICFA indicates that the FRC value has been transferred to ICRA by an input capture signal and that the ICRA value before this update has been transferred to ICRC.

This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 7: ICFA	Description
0	Clearing conditions: After reading ICFA = 1, cleared by writing 0 to ICFA (initial value)
1	Setting conditions: Set when the FRC value is transferred to ICRA by an input capture signal

Bit 6—Input Capture Flag B (ICFB): Bit 6 is a status flag that indicates that the FRC value has been transferred to ICRB by an input capture signal. If BUFEB is set to 1 in TCRX, ICFB indicates that the FRC value has been transferred to ICRB by an input capture signal and that the ICRB value before this update has been transferred to ICRC.

This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 6: ICFB	Description
0	Clearing conditions: After reading ICFB = 1, cleared by writing 0 to ICFB (initial value)
1	Setting conditions: Set when the FRC value is transferred to ICRB by an input capture signal

Bit 5—Input Capture Flag C (ICFC): Bit 5 is a status flag that indicates that the FRC value has been transferred to ICRC by an input capture signal. If BUFEA is set to 1 in TCRX, ICFC is set by the input capture signal even though the FRC value is not transferred to ICRC. In buffered operation, ICFC can accordingly be used as an external interrupt, by setting the ICICE bit to 1.

This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 5: ICFC	Description
0	Clearing conditions: After reading ICFC = 1, cleared by writing 0 to ICFC (initial value)
1	Setting conditions: Set by input capture signal

Bit 4—Input Capture Flag D (ICFD): Bit 4 is a status flag that indicates that the FRC value has been transferred to ICRD by an input capture signal. If BUFEB is set to 1 in TCRX, ICFD is set by the input capture signal even though the FRC value is not transferred to ICRD. In buffered operation, ICFD can accordingly be used as an external interrupt, by setting the ICIDE bit to 1.

This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 4: ICFD	Description	
0	Clearing conditions: After reading ICFD = 1, cleared by writing 0 to ICFD	(initial value)
1	Setting conditions: Set by input capture signal	

Bit 3—Output Compare Flag A (OCFA): Bit 3 is a status flag that indicates that the FRC value has matched OCRA. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 3: OCFA	Description	
0	Clearing conditions: After reading OCFA = 1, cleared by writing 0 to OCFA	(initial value)
1	Setting conditions: Set when FRC matches OCRA	

Bit 2—Output Compare Flag B (OCFB): Bit 2 is a status flag that indicates that the FRC value has matched OCRB. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 2: OCFB	Description	
0	Clearing conditions: After reading OCFB = 1, cleared by writing 0 to OCFB	(initial value)
1	Setting conditions: Set when FRC matches OCRB	

Bit 1—Timer Overflow Flag (OVF): Bit 1 is a status flag that indicates that FRC has overflowed from H'FFFF to H'0000. This flag is set by hardware and cleared by software. It cannot be set by software.

Bit 1: OVF	Description	
0	Clearing conditions: After reading OVF = 1, cleared by writing 0 to OVF	(initial value)
1	Setting conditions: Set when the FRC value overflows from H'FFFF to H'0000	

Bit 0—Counter Clear A (CCLRA): Bit 0 selects whether or not to clear FRC by compare match A (when FRC matches OCRA).

Bit 0: CCLRA	Description	
0	FRC is not cleared by compare match A	(initial value)
1	FRC is cleared by compare match A	

Timer Control Register X (TCRX)

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

TCRX is an 8-bit read/write register that selects the valid edges of the input capture signals, enables buffering, and selects the FRC clock source.

TCRX is initialized to H'00 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

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

Bit 7: IEDGA	Description	
0	Falling edge of input capture A is captured	(initial value)
1	Rising edge of input capture A is captured	

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

Bit 6: IEDGB	Description	
0	Falling edge of input capture B is captured	(initial value)
1	Rising edge of input capture B is captured	

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

Bit 5: IEDGC	Description	
0	Falling edge of input capture C is captured	(initial value)
1	Rising edge of input capture C is captured	

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

Bit 4: IEDGD	Description	
0	Falling edge of input capture D is captured	(initial value)
1	Rising edge of input capture D is captured	

Bit 3—Buffer Enable A (BUFEA): Bit 3 selects whether or not to use ICRC as a buffer register for ICRA.

Bit 3: BUFEA	Description	
0	ICRC is not used as a buffer register for ICRA	(initial value)
1	ICRC is used as a buffer register for ICRA	

Bit 2—Buffer Enable B (BUFEB): Bit 2 selects whether or not to use ICRD as a buffer register for ICRB.

Bit 2: BUFEB	Description	
0	ICRD is not used as a buffer register for ICRB	(initial value)
1	ICRD is used as a buffer register for ICRB	

Bits 1 and 0—Clock Select (CKS1, CKS0): Bits 1 and 0 select one of three internal clock sources or an external clock for input to FRC. The external clock is counted on the rising edge.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	Internal clock: ø/2	(initial value)
	1	Internal clock: ø/8	
1	0	Internal clock: ø/32	
	1	External clock: rising edge	

Timer Output Compare Control Register (TOCR)

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

TOCR is an 8-bit read/write register that selects the output compare output levels, enables output compare output, and controls access to OCRA and OCRB.

TOCR is initialized to H'E0 upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

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

Bit 4—Output Compare Register Select (OCRS): OCRA and OCRB share the same address. OCRS selects which register is accessed when this address is written or read. It does not affect the operation of OCRA and OCRB.

Bit 4: OCRS	Description	
0	OCRA is selected	(initial value)
1	OCRB is selected	

Bit 3—Output Enable A (OEA): Bit 3 enables or disables the timer output controlled by output compare A.

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

Bit 2—Output Enable B (OEB): Bit 2 enables or disables the timer output controlled by output compare B.

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

Bit 1—Output Level A (OLVLA): Bit 1 selects the output level that is output at pin FTOA by compare match A (when FRC matches OCRA).

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

Bit 0—Output Level B (OLVLB): Bit 0 selects the output level that is output at pin FTOB by compare match B (when FRC matches OCRB).

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

9.5.3 CPU Interface

FRC, OCRA, OCRB, and ICRA to ICRD are 16-bit 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).

These registers should always be accessed 16 bits at a time. If two consecutive byte-size MOV instructions are used, the upper byte must be accessed first and the lower byte second. Data will not be transferred correctly if only the upper byte or only the lower byte is accessed.

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 9.18 shows an example of the writing of H'AA55 to FRC.

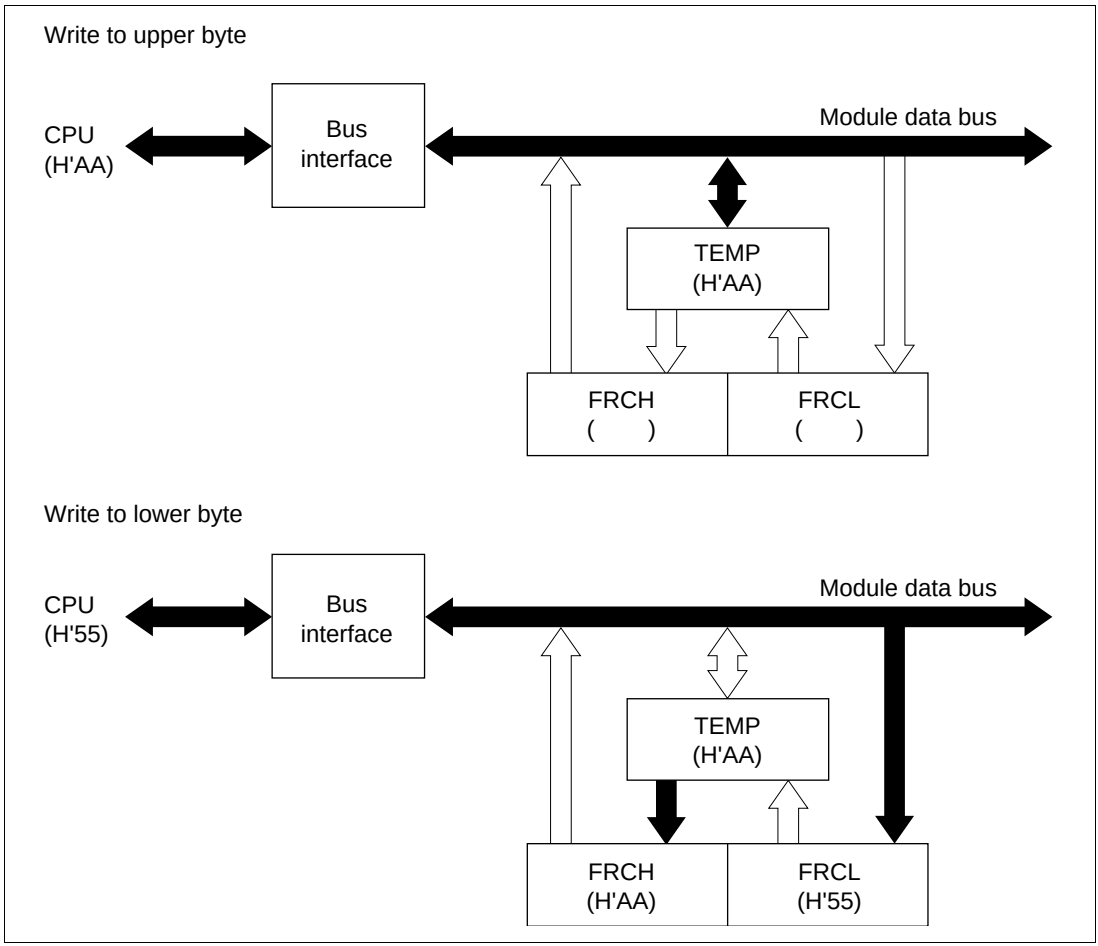


Figure 9.18 Write Access to FRC (CPU → FRC)

Read Access: In access to FRC and ICRA to ICRD, 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 OCRA or OCRB, when the upper byte is read the upper-byte data is transferred directly to the CPU, and when the lower byte is read the lower-byte data is transferred directly to the CPU.

Figure 9.19 shows an example of the reading of FRC when FRC contains H'AAFF.

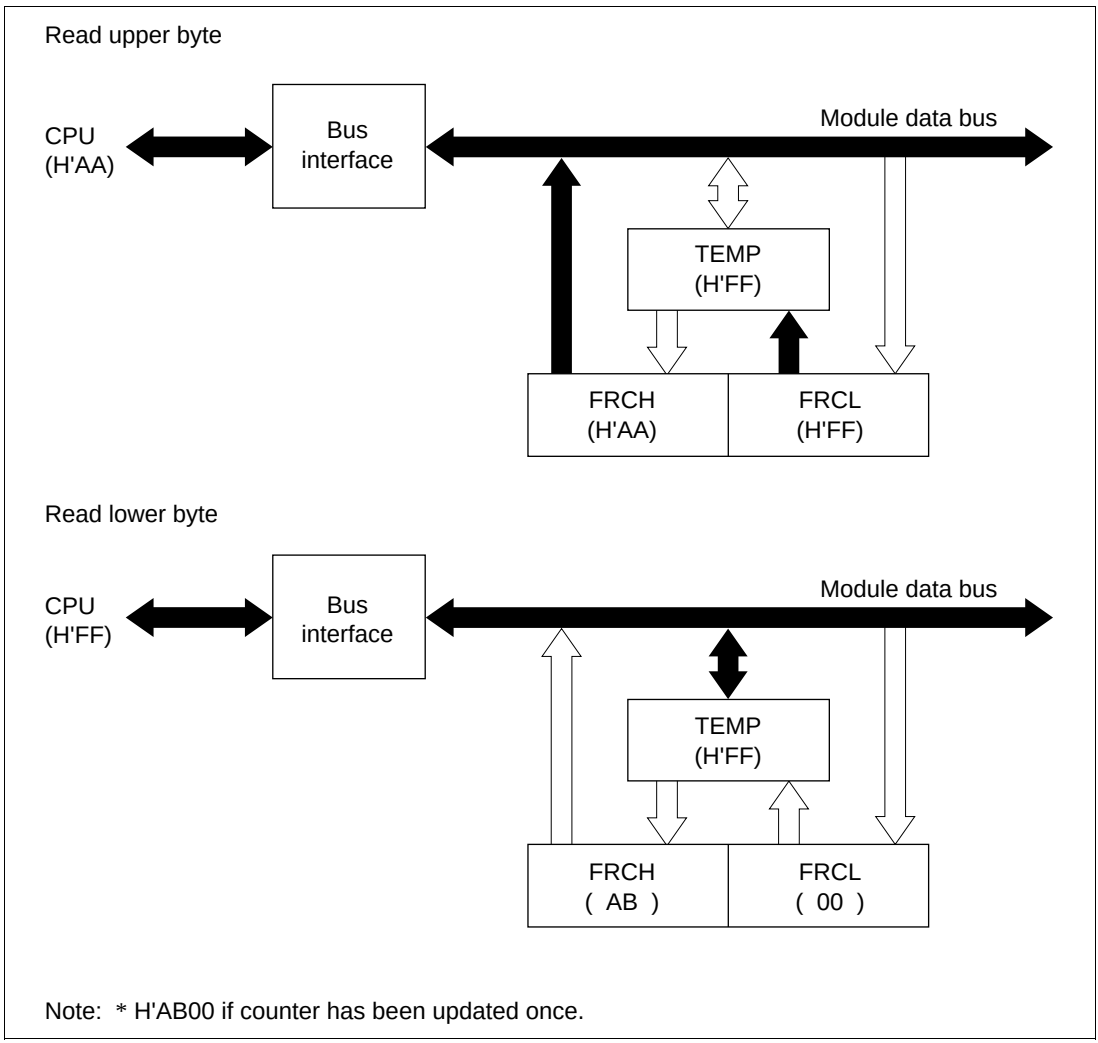


Figure 9.19 Read Access to FRC (FRC → CPU)

9.5.4 Timer Operation

Timer X Operation

- Output compare operation

  Following a reset, FRC is initialized to H'0000 and starts counting up. Bits CKS1 and CKS0 in TCRX can select one of three internal clock sources or an external clock for input to FRC. The FRC contents are compared constantly with OCRA and OCRB. When a match occurs, the output at pin FTOA or FTOB goes to the level selected by OLVLA or OLVLB in TOCR. Following a reset, the output at both FTOA and FTOB is 0 until the first compare match. If CCLRA is set to 1 in TCSRX, compare match A clears FRC to H'0000.

- Input capture operation

  Following a reset, FRC is initialized to H'0000 and starts counting up. Bits CKS1 and CKS0 in TCRX can select one of three internal clock sources or an external clock for input to FRC. When the edges selected by bits IEDGA to IEDGD in TCRX are input at pins FTIA to FTID, the FRC value is transferred to ICRA to ICRD, and ICFA to ICFD are set to 1 in TCSRX. If bits ICIAE to ICIDE are set to 1 in TIER, a CPU interrupt is requested.

  If bits BUFEA and BUFEB are set to 1 in TCRX, ICRC and ICRD operate as buffer registers for ICRA or ICRB. When the edges selected by bits IEDGA to IEDGD in TCRX are input at pins FTIA and FTIB, the FRC value is transferred to ICRA or ICRB, and the previous value in ICRA or ICRB is transferred to ICRC or ICRD. Simultaneously, ICFA or ICFB is set to 1. If bit ICIAE or ICIBE is set to 1 in TIER, a CPU interrupt is requested.

FRC Count Timing: FRC is incremented by clock input. Bits CKS1 and CKS0 in TCRX can select one of three internal clock sources (ø/2, ø/8, ø/32) or an external clock.

- Internal clock

  Bits CKS1 and CKS0 in TCRX select one of three internal clock sources (ø/2, ø/8, ø/32) created by dividing the system clock (ø). Figure 9.20 shows the increment timing.

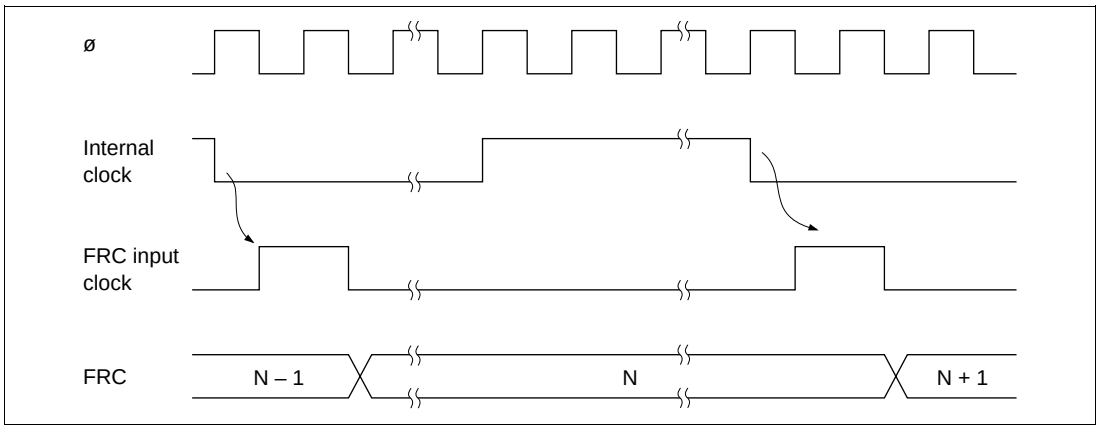


Figure 9.20 Increment Timing with Internal Clock

- External clock

  External clock input is selected when bits CKS1 and CKS0 are both set to 1 in TCRX. FRC increments on the rising edge of the external clock. An external pulse width of at least 1.5 system clocks (ø) is necessary. Shorter pulses will not be counted correctly. Figure 9.21 shows the timing.

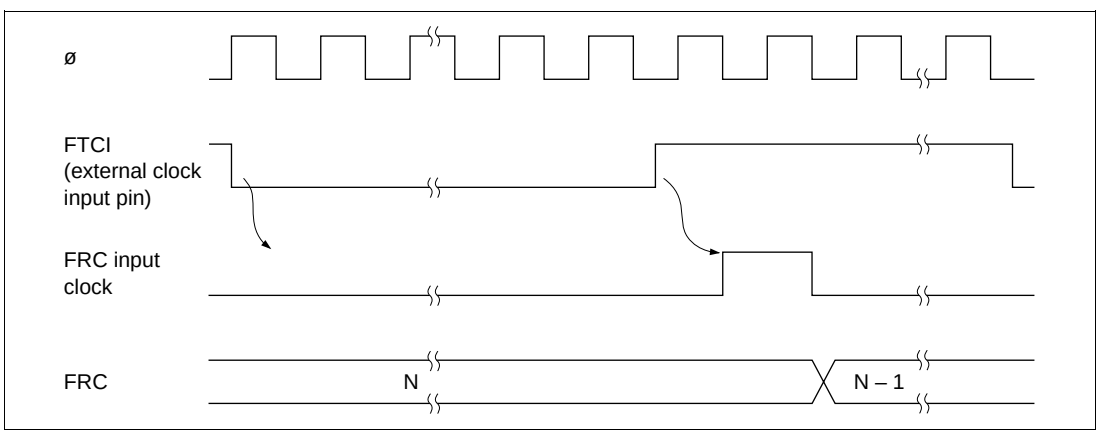


Figure 9.21 Increment Timing with External Clock

Output Compare Timing: When a compare match occurs, the output level selected by the OLVL bit in TOCR is output at pin FTOA or FTOB. Figure 9.22 shows the output timing for output compare A.

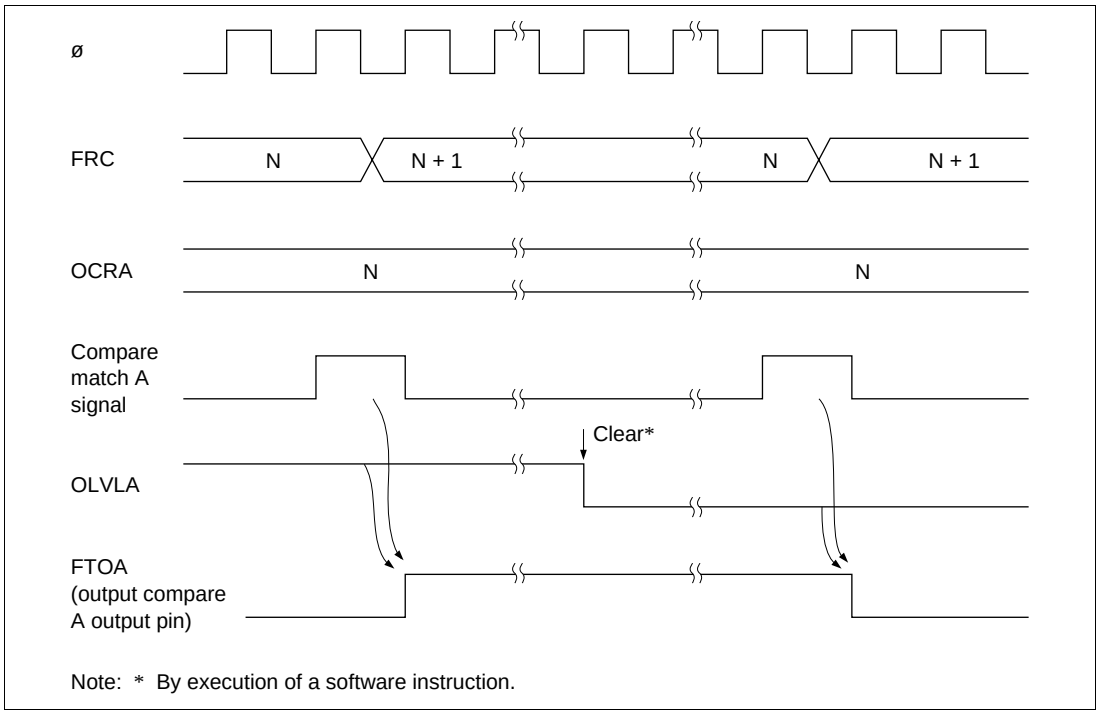


Figure 9.22 Output Compare A Output Timing

FRC Clear Timing: FRC can be cleared by compare match A. Figure 9.23 shows the timing.

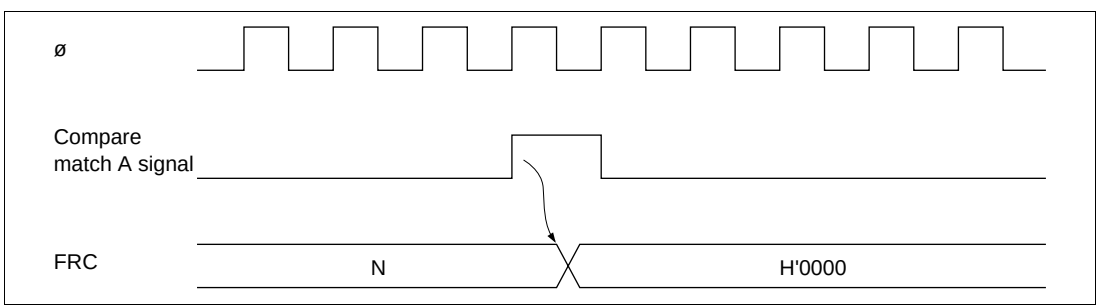


Figure 9.23 Clear Timing by Compare Match A

Input Capture Timing

- Input capture timing

The rising or falling edge is selected for input capture by bits IEDGA to IEDGD in TCRX. Figure 9.24 shows the timing when the rising edge is selected (IEDGA/B/C/D = 1).

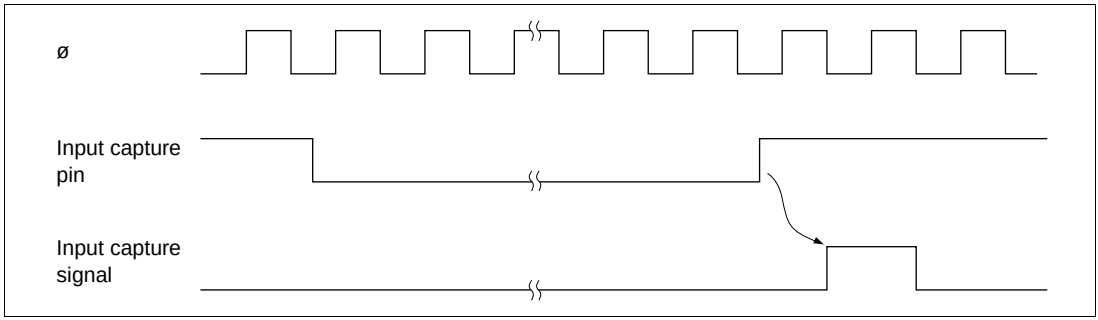


Figure 9.24 Input Capture Signal Timing (Normal Case)

If the input at the input capture pin occurs while the upper byte of the corresponding input capture register (ICRA to ICRD) is being read, the internal input capture signal is delayed by one system clock (ø). Figure 9.25 shows the timing.

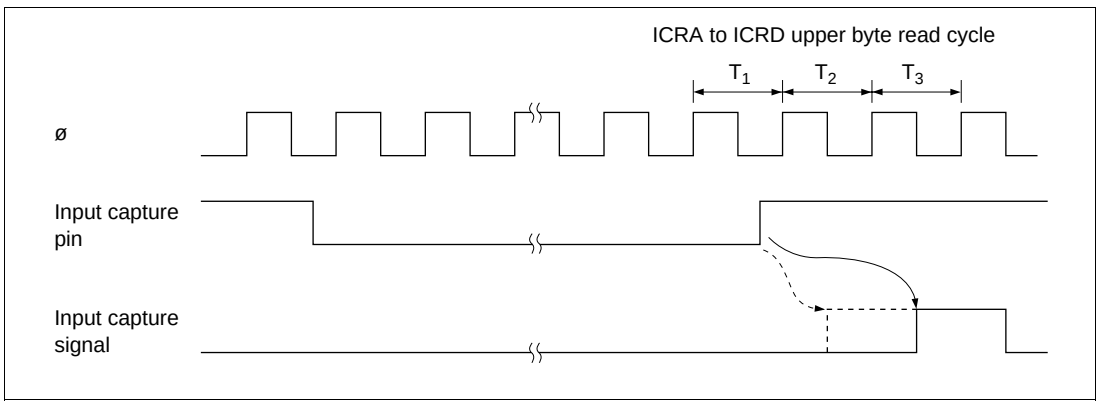


Figure 9.25 Input Capture Signal Timing (during ICRA to ICRD Read)

- Buffered input capture timing

  Input capture can be buffered by using ICRC or ICRD as a buffer for ICRA or ICRB. Figure 9.26 shows the timing when ICRA is buffered by ICRC (BUFEA = 1) and both the rising and falling edges are selected (IEDGA = 1 and IEDGC = 0, or IEDGA = 0 and IEDGC = 1).

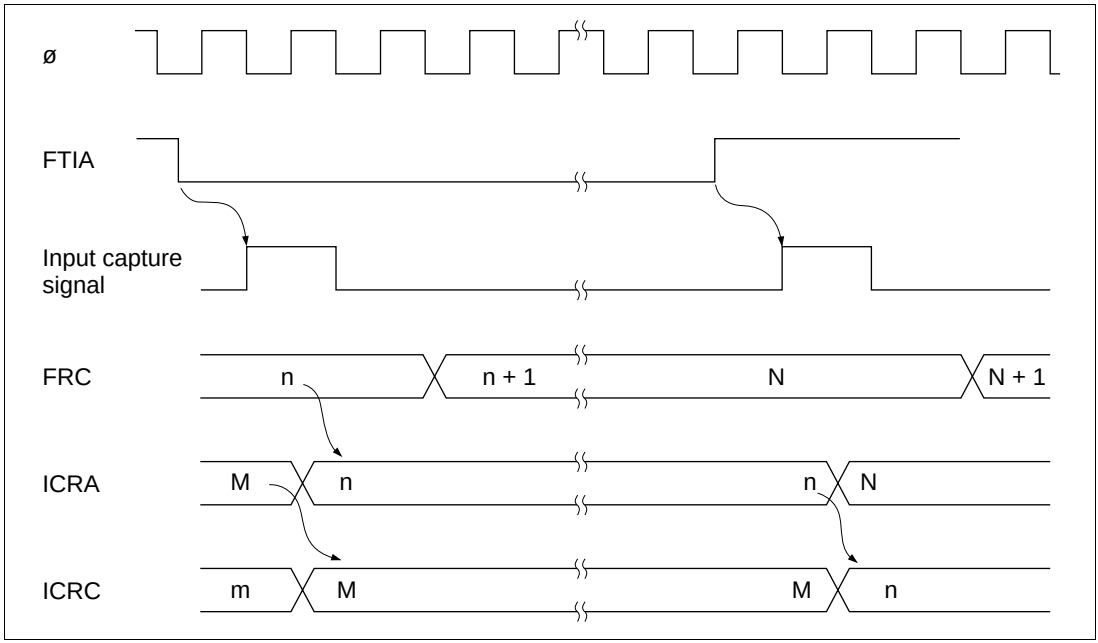


Figure 9.26 Buffered Input Capture Timing (Normal Case)

When ICRC or ICRD is used as a buffer register, the input capture flag is still set by the selected edge of the input capture input signal. For example, if ICRC is used to buffer ICRA, when the edge transition selected by the IEDGC bit occurs at the input capture pin, ICFC will be set, and if the ICIEC bit is set, an interrupt will be requested. The FRC value will not be transferred to ICRC, however.

In buffered operation, if the upper byte of one of the two registers that receives a data transfer (ICRA and ICRC, or ICRB and ICRD) is being read when an input capture signal would normally occur, the input capture signal will be delayed by one system clock (ø). Figure 9.27 shows the case when BUFEA = 1.

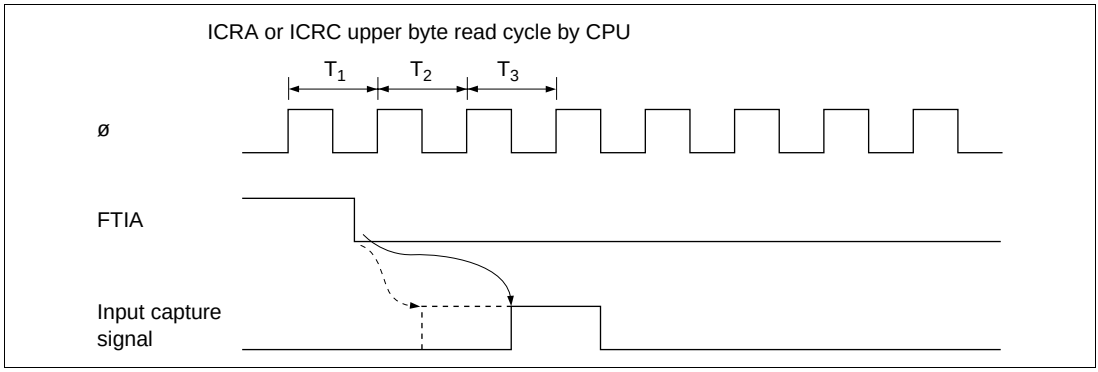


Figure 9.27 Buffered Input Capture Signal Timing (during ICRA or ICRD Read)

Input Capture Flag (ICFA to ICFD) Set Timing: Figure 9.28 shows the timing when an input capture flag (ICFA to ICFD) is set to 1 and the FRC value is transferred to the corresponding input capture register (ICRA to ICRD).

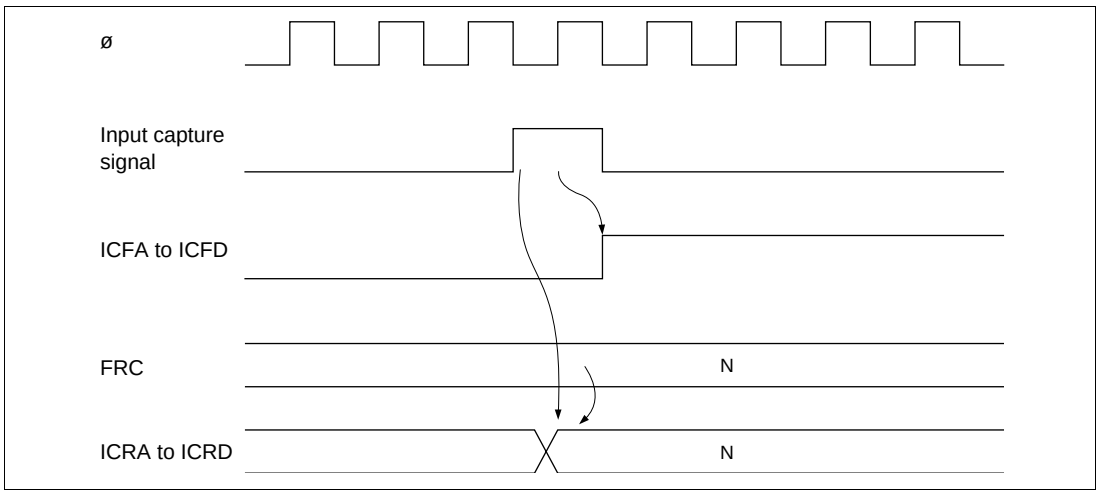


Figure 9.28 ICFA to ICFD Set Timing

Output Compare Flag (OCFA or OCFB) Set Timing: OCFA and OCFB are set to 1 by internal compare match signals that are output when FRC matches OCRA or OCRB. The compare match signal is generated in the last state during which the values match (when FRC is updated from the matching value to a new value). When FRC matches OCRA or OCRB, the compare match signal is not generated until the next counter clock. Figure 9.29 shows the OCFA and OCFB set timing.

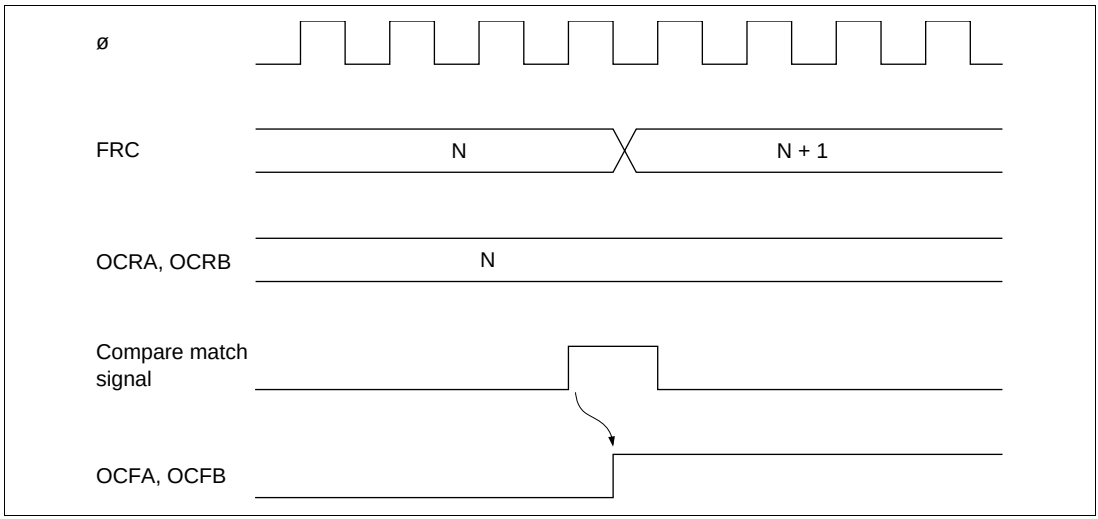


Figure 9.29 OCFA and OCFB Set Timing

Overflow Flag (OVF) Set Timing: OVF is set to 1 when FRC overflows from H'FFFF to H'0000. Figure 9.30 shows the timing.

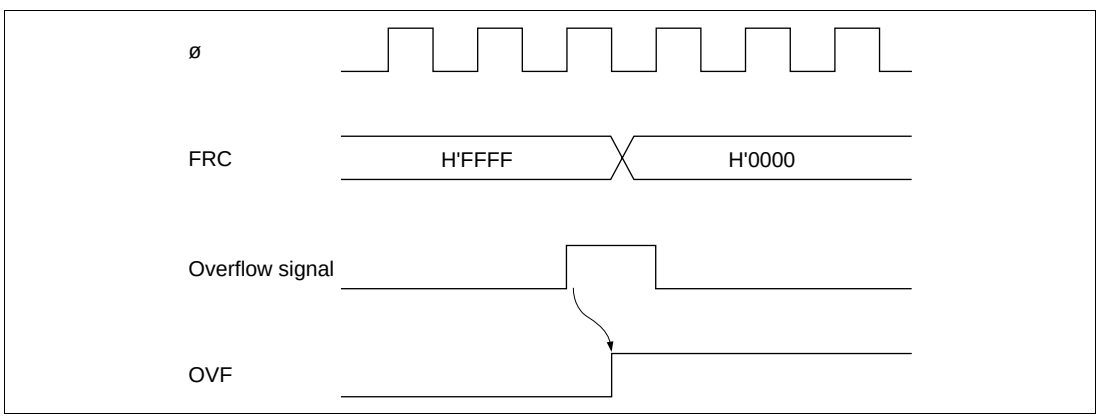


Figure 9.30 OVF Set Timing

9.5.5 Timer X Operation Modes

Figure 9.17 shows the timer X operation modes.

Table 9.17 Timer X Operation Modes

Operation Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
FRC	Reset	Functions	Functions	Reset	Reset	Reset	Reset
OCRA, OCRB	Reset	Functions	Functions	Reset	Reset	Reset	Reset
ICRA to ICRD	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TIER	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TCRX	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TOCR	Reset	Functions	Functions	Reset	Reset	Reset	Reset
TCSRX	Reset	Functions	Functions	Reset	Reset	Reset	Reset

9.5.6 Interrupt Sources

Timer X has three types of interrupts and seven interrupt sources: ICIA to ICID, OCIA, OCIB, and FOVI. Table 9.18 lists the sources of interrupt requests. Each interrupt source can be enabled or disabled by an interrupt enable bit in TIER. Although all seven interrupts share the same vector, they have individual interrupt flags, so software can discriminate the interrupt source.

Table 9.18 Timer X Interrupt Sources

Interrupt	Description	Vector Address
ICIA	Interrupt requested by ICFA	H'0020
ICIB	Interrupt requested by ICFB	
ICIC	Interrupt requested by ICFC	
ICID	Interrupt requested by ICFD	
OCIA	Interrupt requested by OCFA	
OCIB	Interrupt requested by OCFB	
FOVI	Interrupt requested by OVF	

9.5.7 Timer X Application Example

Figure 9.31 shows an example of the output of pulse signals with a 50% duty cycle and arbitrary phase offset. To set up this output:

- Set bit CCLRA to 1 in TCSRX.
- Have software invert the OLVLA and OLVLB bits at each corresponding compare match.

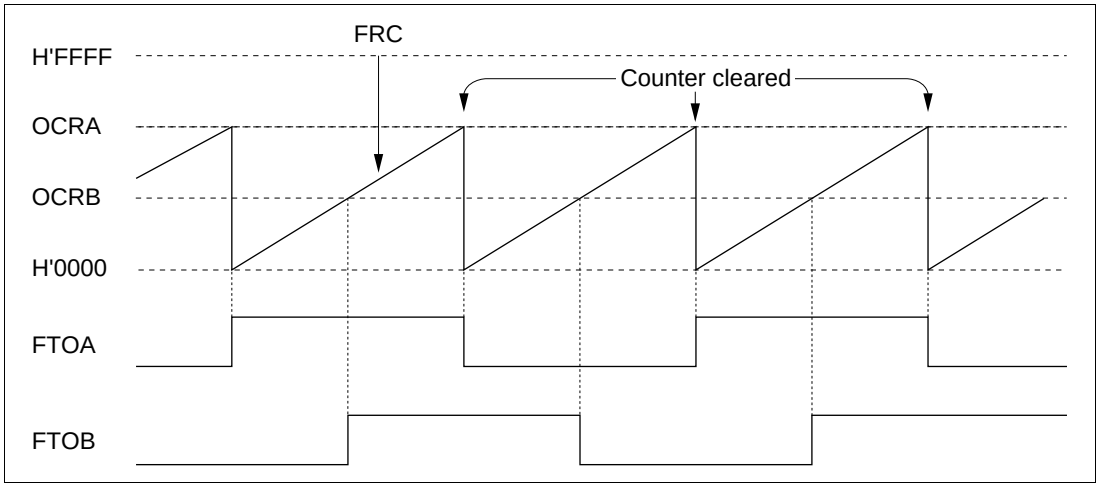


Figure 9.31 Pulse Output Example

9.5.8 Application Notes

The following types of contention can occur in timer X operation.

1. Contention between FRC write and counter clear

   If an FRC clear signal is generated in the T_3 state of a write cycle to the lower byte of FRC, clearing takes precedence and the write to the counter is not carried out. Figure 9.32 shows the timing.

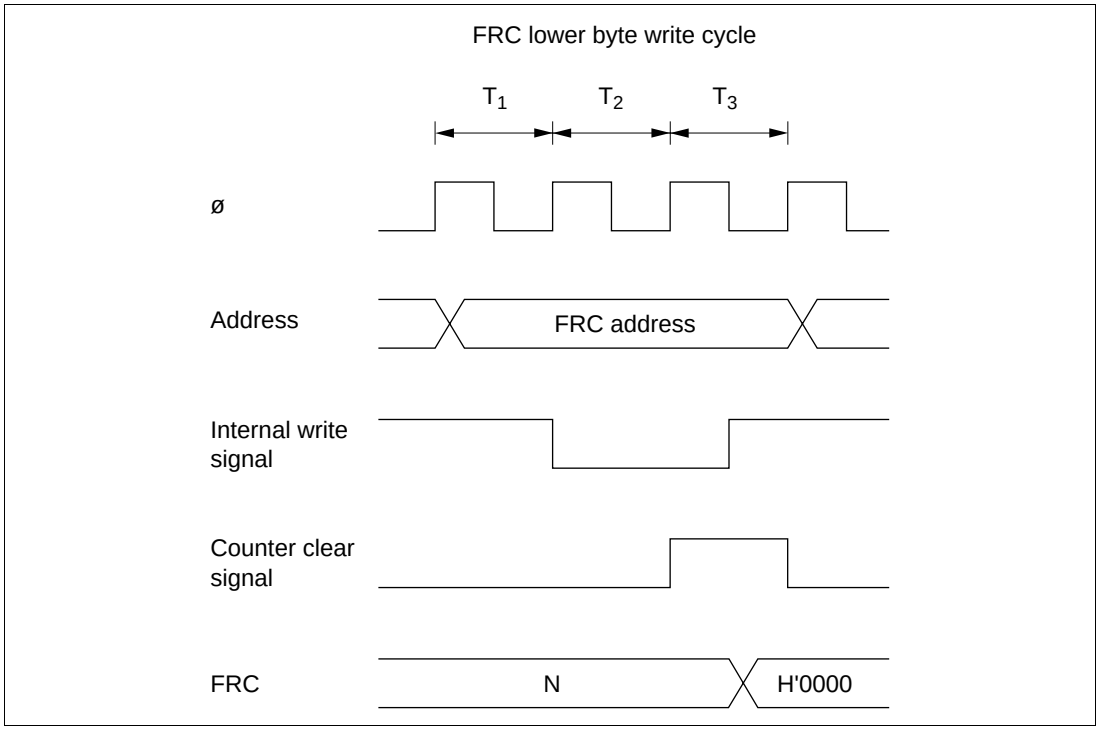


Figure 9.32 Contention between FRC Write and Clear

2. Contention between FRC write and increment

If an FRC increment clock signal is generated in the T_3 state of a write cycle to the lower byte of FRC, the write takes precedence and the counter is not incremented. Figure 9.33 shows the timing.

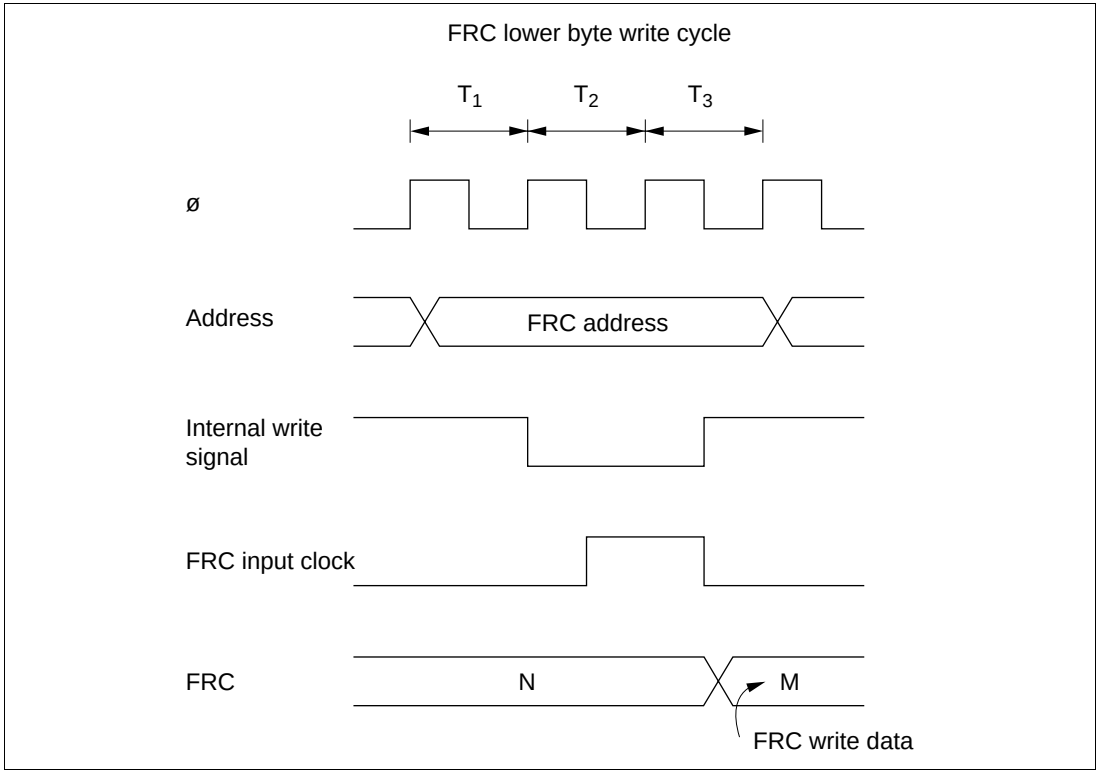


Figure 9.33 Contention between FRC Write and Increment

3. Contention between OCR write and compare match

If a compare match is generated in the T_3 state of a write cycle to the lower byte of OCRA or OCRB, the write to OCRA or OCRB takes precedence and the compare match signal is inhibited. Figure 9.34 shows the timing.

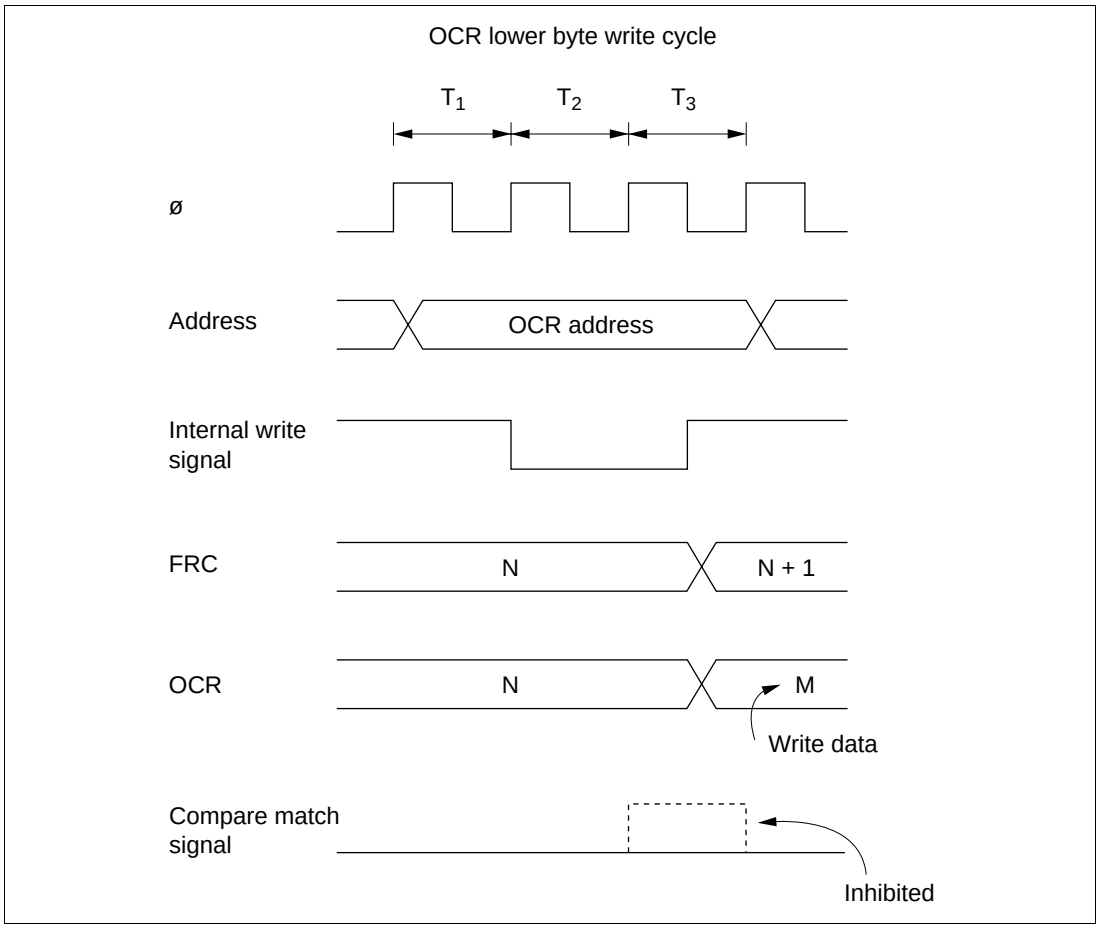


Figure 9.34 Contention between OCR Write and Compare Match

4. Internal clock switching and counter operation

Depending on the timing, FRC may be incremented by a switch between different internal clock sources. Table 9.19 shows the relation between internal clock switchover timing (by writing to bits CKS1 and CKS0) and FRC operation.

When FRC is internally clocked, an increment pulse is generated from the falling edge of an internal clock signal, which is divided from the system clock (ø). For this reason, in a case like No. 3 in table 9.19 where the switch is from a high clock signal to a low clock signal, the switchover is seen as a falling edge, causing FRC to increment.

FRC can also be incremented by a switch between internal and external clocks.

Table 9.19 Internal Clock Switching and FRC Operation

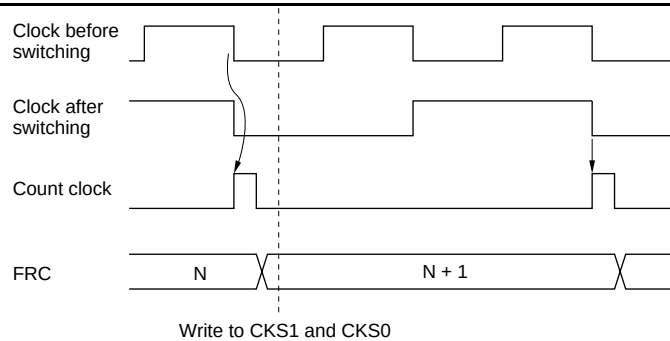
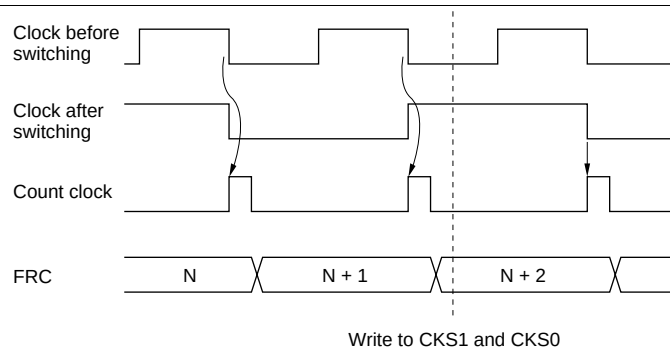
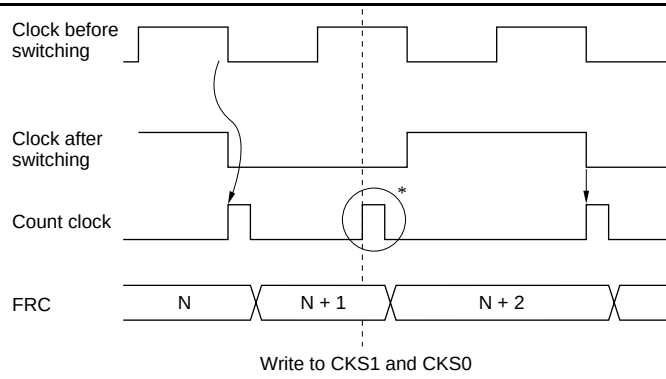
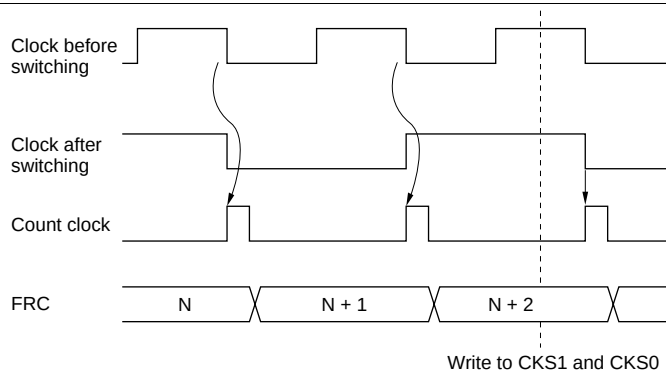
No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	FRC Operation
1	Goes from low level to low level	
2	Goes from low to high	

Table 9.19 Internal Clock Switching and FRC Operation (cont)

No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	FRC Operation
3	Goes from high level to low level	
4	Goes from high to high	

Note: * The switchover is seen as a falling edge, and FRC is incremented.

9.6 Watchdog Timer

9.6.1 Overview

The watchdog timer has an 8-bit counter that is incremented by an input clock. If a system runaway allows the counter value to overflow before being rewritten, the watchdog timer can reset the chip internally.

Features

Features of the watchdog timer are given below.

- Incremented by internal clock source (ø/8192).
- A reset signal is generated when the counter overflows. The overflow period can be set from 1 to 256 times 8192/ø (from approximately 2 ms to 500 ms when ø = 4.19 MHz).

Block Diagram

Figure 9.35 shows a block diagram of the watchdog timer.

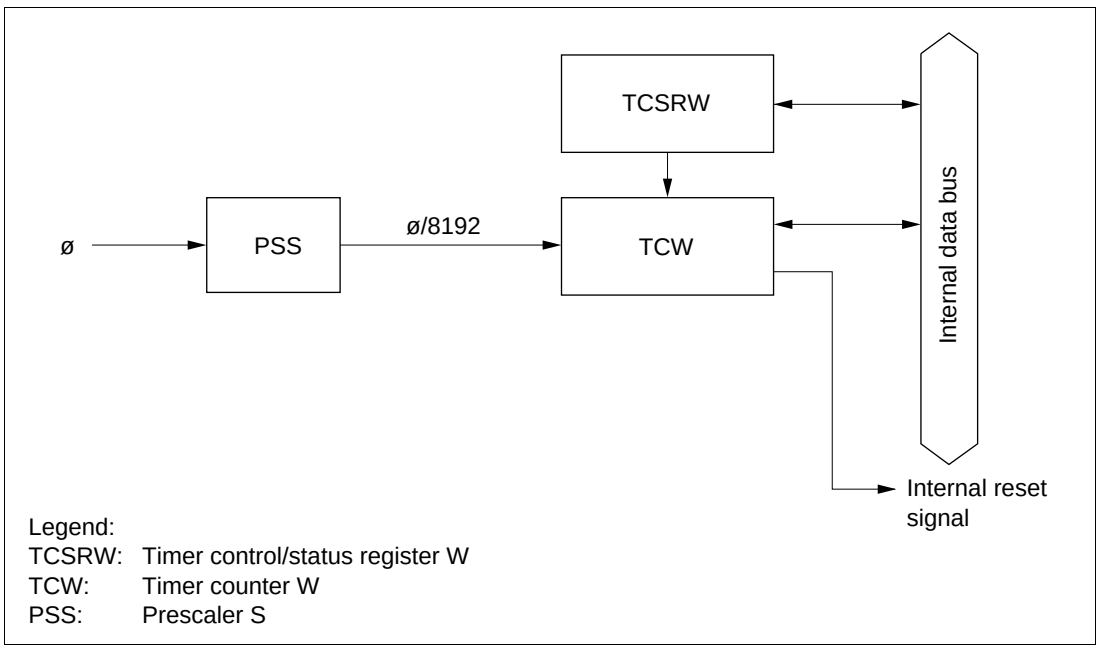


Figure 9.35 Block Diagram of Watchdog Timer

Register Configuration

Table 9.20 shows the register configuration of the watchdog timer.

Table 9.20 Watchdog Timer Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer control/status register W	TCSRW	R/W	H'AA	H'FFBE
Timer counter W	TCW	R/W	H'00	H'FFBF

9.6.2 Register Descriptions

Timer Control/Status Register W (TCSRW)

Bit	7	6	5	4	3	2	1	0
	B6WI	TCWE	B4WI	TCSRWE	B2WI	WDON	B0WI	WRST
Initial value	1	0	1	0	1	0	1	0
Read/Write	R	R/(W)*	R	R/(W)*	R	R/(W)*	R	R/(W)*

Note: * Write is permitted only under certain conditions, which are given in the descriptions of the individual bits.

TCSRW is an 8-bit read/write register that controls write access to TCW and TCSRW itself, controls watchdog timer operations, and indicates operating status.

Bit 7—Bit 6 Write Inhibit (B6WI): Bit 7 controls the writing of data to bit 6 in TCSRW.

This bit is always read as 1. Data written to this bit is not stored.

Bit 7: B6WI	Description	
0	Bit 6 is write-enabled	
1	Bit 6 is write-protected	(initial value)

Bit 6—Timer Counter W Write Enable (TCWE): Bit 6 controls the writing of data to bit 8 to TCW.

Bit 6: TCWE	Description	
0	Data cannot be written to TCW	(initial value)
1	Data can be written to TCW	

Bit 5—Bit 4 Write Inhibit (B4WI): Bit 5 controls the writing of data to bit 4 in TCSRW.

This bit is always read as 1. Data written to this bit is not stored.

Bit 5: B4WI	Description	
0	Bit 4 is write-enabled	
1	Bit 4 is write-protected	(initial value)

Bit 4—Timer Control/Status Register W Write Enable (TCSRWE): Bit 4 controls the writing of data to TCSRW bits 2 and 0.

Bit 4: TCSRWE	Description	
0	Data cannot be written to bits 2 and 0	(initial value)
1	Data can be written to bits 2 and 0	

Bit 3—Bit 2 Write Inhibit (B2WI): Bit 3 controls the writing of data to bit 2 in TCSRW.

This bit is always read as 1. Data written to this bit is not stored.

Bit 3: B2WI	Description	
0	Bit 2 is write-enabled	
1	Bit 2 is write-protected	(initial value)

Bit 2—Watchdog Timer On (WDON): Bit 2 enables watchdog timer operation.

Counting starts when this bit is set to 1, and stops when this bit is cleared to 0.

Bit 2: WDON	Description	
0	Watchdog timer operation is disabled	(initial value)
	Clearing conditions: Reset, or when TCSRWE = 1 and 0 is written in both B2WI and WDON	
1	Watchdog timer operation is enabled	
	Setting conditions: When TCSRWE = 1 and 0 is written in B2WI and 1 is written in WDON	

Bit 1—Bit 0 Write Inhibit (B0WI): Bit 1 controls the writing of data to bit 0 in TCSRW.

This bit is always read as 1. Data written to this bit is not stored.

Bit 1: B0WI	Description
0	Bit 0 is write-enabled
1	Bit 0 is write-protected (initial value)

Bit 0—Watchdog Timer Reset (WRST): Bit 0 indicates that TCW has overflowed, generating an internal reset signal. The internal reset signal generated by the overflow resets the entire chip. WRST is cleared to 0 by a reset from the $\overline{\text{RES}}$ pin, or when software writes 0.

Bit 0: WRST	Description
0	Clearing conditions: (initial value) • Reset by $\overline{\text{RES}}$ pin • When TCSRWE = 1, and 0 is written in both B0WI and WRST
1	Setting conditions: When TCW overflows and an internal reset signal is generated

Timer Counter W (TCW)

Bit	7	6	5	4	3	2	1	0
	TCW7	TCW6	TCW5	TCW4	TCW3	TCW2	TCW1	TCW0
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

TCW is an 8-bit read/write up-counter, which is incremented by internal clock input. The input clock is ø/8192. The TCW value can always be written or read by the CPU.

When TCW overflows from H'FF to H'00, an internal reset signal is generated and WRST is set to 1 in TCSRW. Upon reset, TCW is initialized to H'00.

9.6.3 Timer Operation

The watchdog timer has an 8-bit counter (TCW) that is incremented by clock input (ø/8192). When TCSRWE = 1 in TCSRW, if 0 is written in B2WI and 1 is simultaneously written in WDON, TCW starts counting up (two write accesses to TCSRW are necessary in order to operate the watchdog timer). When the TCW count value reaches H'FF, the next clock input causes the watchdog timer to overflow and generates an internal reset signal. The internal reset signal is output for 512 clock cycles of the $ø_{OSC}$ clock. It is possible to write to TCW, causing TCW to count up from the written value. The overflow period can be set in the range from 1 to 256 input clocks, depending on the value written in TCW.

Figure 9.36 shows an example of watchdog timer operations.

Example: ø = 4 MHz and the desired overflow period is 30 ms.

$$\frac{4 \times 10^6}{8192} \times 30 \times 10^{-3} = 14.6$$

The value set in TCW should therefore be 256 – 15 = 241 (H'F1).

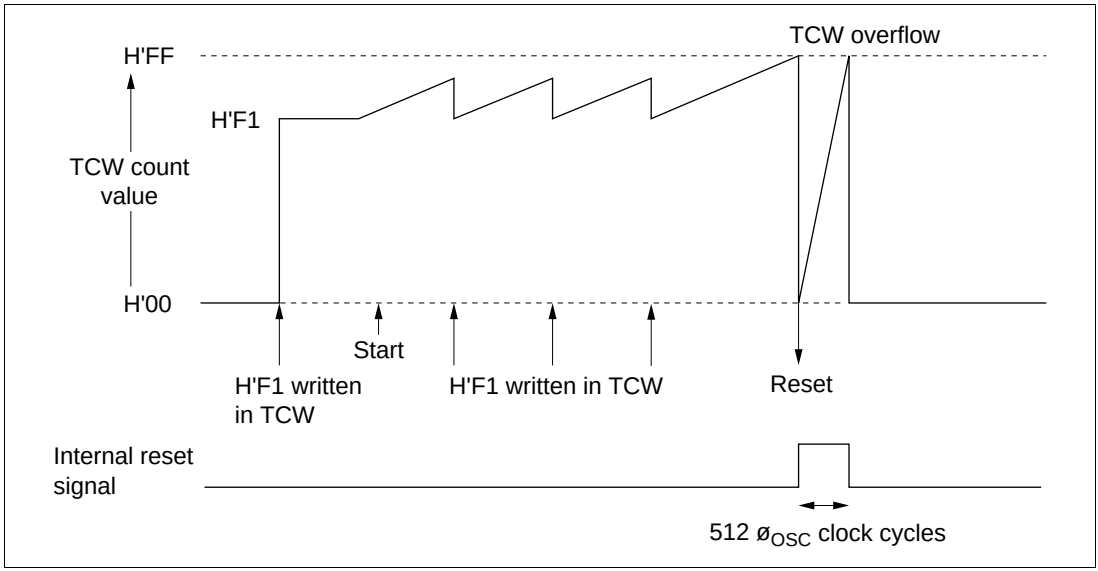


Figure 9.36 Typical Watchdog Timer Operations (Example)

9.6.4 Watchdog Timer Operation States

Table 9.21 summarizes the watchdog timer operation states.

Table 9.21 Watchdog Timer Operation States

Operation Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCW	Reset	Functions	Functions	Halted	Halted	Halted	Halted
TCSRW	Reset	Functions	Functions	Retained	Retained	Retained	Retained

Section 10 Serial Communication Interface

10.1 Overview

The H8/3644 Series is provided with a two-channel serial communication interface (SCI). Table 10.1 summarizes the functions and features of the two SCI channels.

Table 10.1 Serial Communication Interface Functions

Channel	Functions	Features
SCI1	Synchronous serial transfer • Choice of 8-bit or 16-bit data length • Continuous clock output	• Choice of 8 internal clocks (ø/1024 to ø/2) or external clock • Open drain output possible • Interrupt requested at completion of transfer
SCI3	Synchronous serial transfer • 8-bit data length • Send, receive, or simultaneous send/receive Asynchronous serial transfer • Multiprocessor communication • Choice of 7-bit or 8-bit data length • Choice of 1 or 2 stop bits • Parity addition	• On-chip baud rate generator • Receive error detection • Break detection • Interrupt requested at completion of transfer or error

10.2 SCI1

10.2.1 Overview

Serial communication interface 1 (SCI1) performs synchronous serial transfer of 8-bit or 16-bit data. SSB (Synchronized Serial Bus) communication is also provided, enabling multiple ICs to be controlled.

Features

- Choice of 8-bit or 16-bit data length
- Choice of eight internal clock sources (ø/1024, ø/256, ø/64, ø/32, ø/16, ø/8, ø/4, ø/2) or an external clock
- Interrupt requested at completion of transfer
- Choice of HOLD mode or LATCH mode in SSB mode.

Block Diagram

Figure 10.1 shows a block diagram of SCI1.

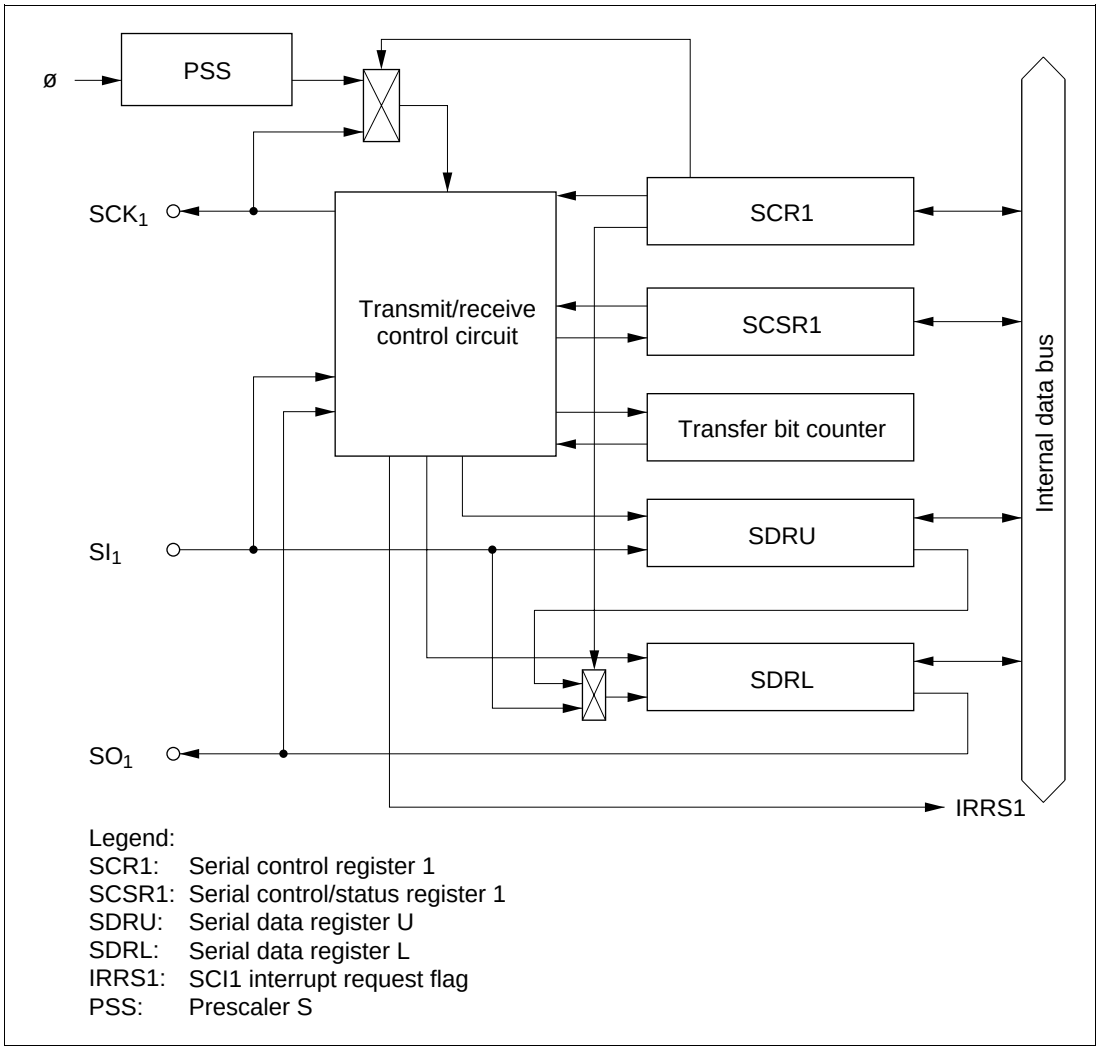


Figure 10.1 SCI1 Block Diagram

Pin Configuration

Table 10.2 shows the SCI1 pin configuration.

Table 10.2 Pin Configuration

Name	Abbrev.	I/O	Function
SCI1 clock pin	SCK_1	I/O	SCI1 clock input or output
SCI1 data input pin	SI_1	Input	SCI1 receive data input
SCI1 data output pin	SO_1	Output	SCI1 transmit data output

Register Configuration

Table 10.3 shows the SCI1 register configuration.

Table 10.3 SCI1 Registers

Name	Abbrev.	R/W	Initial Value	Address
Serial control register 1	SCR1	R/W	H'00	H'FFA0
Serial control status register 1	SCSR1	R/W	H'9C	H'FFA1
Serial data register U	SDRU	R/W	Not fixed	H'FFA2
Serial data register L	SDRL	R/W	Not fixed	H'FFA3

10.2.2 Register Descriptions

Serial Control Register 1 (SCR1)

Bit	7	6	5	4	3	2	1	0
	SNC1	SNC0	MRKON	LTCH	CKS3	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCR1 is an 8-bit read/write register for selecting the operation mode, the transfer clock source, and the prescaler division ratio.

Upon reset, SCR1 is initialized to H'00. Writing to this register during a transfer stops the transfer.

Bits 7 and 6—Operation Mode Select 1, 0 (SNC1, SNC0): Bits 7 and 6 select the operation mode.

Bit 7: SNC1	Bit 6: SNC0	Description	
0	0	8-bit synchronous transfer mode	(initial value)
	1	16-bit synchronous transfer mode	
1	0	Continuous clock output mode*[1]	
	1	Reserved*[2]	

Notes: 1. Pins SI_1 and SO_1 should be used as general input or output ports.
2. Don't set bits SNC1 and SNC0 to 11.

Bits 5—TAIL MARK Control (MRKON): Bit 5 controls TAIL MARK output after an 8- or 16-bit data transfer.

Bit 5: MRKON	Description	
0	TAIL MARK is not output (synchronous mode)	(initial value)
1	TAIL MARK is output (SSB mode)	

Bits 4—LATCH TAIL Select (LTCH): Bit 4 selects whether LATCH TAIL or HOLD TAIL is output as TAIL MARK when bit MRKON is set to 1 (SSB mode).

Bit 4: LTCH	Description	
0	HOLD TAIL is output	(initial value)
1	LATCH TAIL is output	

Bit 3—Clock Source Select (CKS3): Bit 3 selects the clock source and sets pin SCK_1 as an input or output pin.

Bit 3: CKS3	Description	
0	Clock source is prescaler S, and pin SCK_1 is output pin	(initial value)
1	Clock source is external clock, and pin SCK_1 is input pin	

Bits 2 to 0—Clock Select (CKS2 to CKS 0): When CKS3 = 0, bits 2 to 0 select the prescaler division ratio and the serial clock cycle.

				Serial Clock Cycle	
Bit 2: CKS2	**Bit 1: CKS1**	**Bit 0: CKS0**	**Prescaler Division**	**ø = 5 MHz**	**ø = 2.5 MHz**
0	0	0	ø/1024 (initial value)	204.8 µs	409.6 µs
		1	ø/256	51.2 µs	102.4 µs
	1	0	ø/64	12.8 µs	25.6 µs
		1	ø/32	6.4 µs	12.8 µs
1	0	0	ø/16	3.2 µs	6.4 µs
		1	ø/8	1.6 µs	3.2 µs
	1	0	ø/4	0.8 µs	1.6 µs
		1	ø/2	—	0.8 µs

Serial Control/Status Register 1 (SCSR1)

Bit	7	6	5	4	3	2	1	0
	—	SOL	ORER	—	—	—	MTRF	STF
Initial value	1	0	0	1	1	1	0	0
Read/Write	—	R/W	R/(W)*	—	—	—	R	R/W

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

SCSR1 is an 8-bit read/write register indicating operation status and error status.

Upon reset, SCSR1 is initialized to H'9C.

Bit 7—Reserved Bit: Bit 7 is reserved; it is always read as 1, and cannot be modified.

Bit 6—Extended Data Bit (SOL): Bit 6 sets the SO_1 output level. When read, SOL returns the output level at the SO_1 pin. After completion of a transmission, SO_1 continues to output the value of the last bit of transmitted data. The SO_1 output can be changed by writing to SOL before or after a transmission. The SOL bit setting remains valid only until the start of the next transmission. SSB mode settings also become invalid. To control the level of the SO_1 pin after transmission ends, it is necessary to write to the SOL bit at the end of each transmission. Do not write to this register while transmission is in progress, because that may cause a malfunction.

Bit 6: SOL	Description	
0	Read: SO_1 pin output level is low	(initial value)
	Write: SO_1 pin output level changes to low	
1	Read: SO_1 pin output level is high	
	Write: SO_1 pin output level changes to high	

Bit 5—Overrun Error Flag (ORER): When an external clock is used, bit 5 indicates the occurrence of an overrun error. If noise occurs during a transfer, causing an extraneous pulse to be superimposed on the normal serial clock, incorrect data may be transferred. If a clock pulse is input after transfer completion, this bit is set to 1 indicating an overrun.

Bit 5: ORER	Description	
0	Clearing conditions: After reading ORER = 1, cleared by writing 0 to ORER	(initial value)
1	Setting conditions: Set if a clock pulse is input after transfer is complete, when an external clock is used	

Bits 4 to 2—Reserved Bits: Bits 4 to 2 are reserved. They are always read as 0, and cannot be modified.

Bit 1—TAIL MARK Transmit Flag (MTRF): When bit MRKON is set to 1, bit 1 indicates that TAIL MARK is being sent. Bit 1 is a read-only bit and cannot be modified.

Bit 1: MTRF	Description	
0	Idle state, or 8- or 16-bit data is being transferred	(initial value)
1	TAIL MARK is being sent	

Bit 0—Start Flag (STF): Bit 0 controls the start of a transfer. Setting this bit to 1 causes SCI1 to start transferring data.

During the transfer or while waiting for the first clock pulse, this bit remains set to 1. It is cleared to 0 upon completion of the transfer. It can therefore be used as a busy flag.

Bit 0: STF	Description	
0	Read: Indicates that transfer is stopped	(initial value)
	Write: Invalid	
1	Read: Indicates transfer in progress	
	Write: Starts a transfer operation	

Serial Data Register U (SDRU)

Bit	7	6	5	4	3	2	1	0
	SDRU7	SDRU6	SDRU5	SDRU4	SDRU3	SDRU2	SDRU1	SDRU0
Initial value	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SDRU is an 8-bit read/write register. It is used as the data register for the upper 8 bits in 16-bit transfer (SDRL is used for the lower 8 bits).

Data written to SDRU is output to SDRL starting from the least significant bit (LSB). This data is then replaced by LSB-first data input at pin SI1, which is shifted in the direction from the most significant bit (MSB) toward the LSB.

SDRU must be written or read only after data transmission or reception is complete. If this register is written or read while a data transfer is in progress, the data contents are not guaranteed.

The SDRU value upon reset is not fixed.

Serial Data Register L (SDRL)

Bit	7	6	5	4	3	2	1	0
	SDRL7	SDRL6	SDRL5	SDRL4	SDRL3	SDRL2	SDRL1	SDRL0
Initial value	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SDRL is an 8-bit read/write register. It is used as the data register in 8-bit transfer, and as the data register for the lower 8 bits in 16-bit transfer (SDRU is used for the upper 8 bits).

In 8-bit transfer, data written to SDRL is output from pin SO_1 starting from the least significant bit (LSB). This data is then replaced by LSB-first data input at pin SI_1, which is shifted in the direction from the most significant bit (MSB) toward the LSB.

In 16-bit transfer, operation is the same as for 8-bit transfer, except that input data is fed in via SDRU.

SDRL must be written or read only after data transmission or reception is complete. If this register is read or written while a data transfer is in progress, the data contents are not guaranteed.

The SDRL value upon reset is not fixed.

10.2.3 Operation in Synchronous Mode

Data can be sent and received in an 8-bit or 16-bit format, with an internal or external clock selected as the clock source. Overrun errors can be detected when an external clock is used.

Clock: The serial clock can be selected from a choice of eight internal clocks and an external clock. When an internal clock source is selected, pin SCK_1 becomes the clock output pin. When continuous clock output mode is selected (SCR1 bits SNC1 and SNC0 are set to 10), the clock signal (ø/1024 to ø/2) selected in bits CKS2 to CKS0 is output continuously from pin SCK_1. When an external clock is used, pin SCK_1 is the clock input pin.

Data Transfer Format: Figure 10.2 shows the data transfer format. Data is sent and received starting from the least significant bit, in LSB-first format. Transmit data is output from one falling edge of the serial clock until the next rising edge. Receive data is latched at the rising edge of the serial clock.

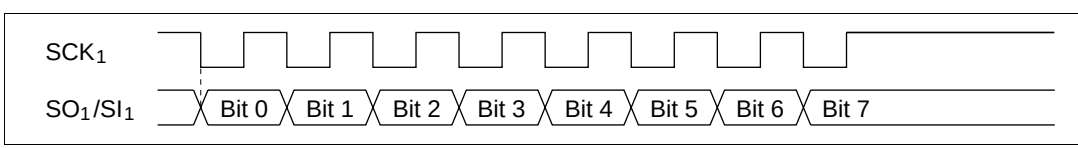


Figure 10.2 Transfer Format

Data Transfer Operations

Transmitting: A transmit operation is carried out as follows.

1. Set bits SO1 and SCK1 to 1 in PMR3 to select the SO_1 and SCK_1 pin functions. If necessary, set bit POF1 in PMR7 for NMOS open-drain output at pin SO_1.
2. Clear bit SNC1 in SCR1 to 0, set bit SNC0 to 0 or 1, and clear bit MRKON to 0, designating 8- or 16-bit synchronous transfer mode. Select the serial clock in bits CKS3 to CKS0. Writing data to SCR1 when bit MRKON in SCR1 is cleared to 0 initializes the internal state of SCI1.
3. Write transmit data in SDRL and SDRU, as follows.

   8-bit transfer mode: SDRL

   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL
4. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating and outputs transmit data at pin SO_1.
5. After data transmission is complete, bit IRRS1 in interrupt request register 2 (IRR2) is set to 1.

When an internal clock is used, a serial clock is output from pin SCK_1 in synchronization with the transmit data. After data transmission is complete, the serial clock is not output until the next time the start flag is set to 1. During this time, pin SO_1 continues to output the value of the last bit transmitted.

When an external clock is used, data is transmitted in synchronization with the serial clock input at pin SCK_1. After data transmission is complete, an overrun occurs if the serial clock continues to be input; no data is transmitted and the SCSR1 overrun error flag (bit ORER) is set to 1.

While transmission is stopped, the output value of pin SO_1 can be changed by rewriting bit SOL in SCSR1.

Receiving: A receive operation is carried out as follows.

1. Set bits SI1 and SCK1 to 1 in PMR3 to select the SI_1 and SCK_1 pin functions.
2. Clear bit SNC1 in SCR1 to 0, set bit SNC0 to 0 or 1, and clear bit MRKON to 0, designating 8- or 16-bit synchronous transfer mode. Select the serial clock in bits CKS3 to CKS0. Writing data to SCR1 when bit MRKON in SCR1 is cleared to 0 initializes the internal state of SCI1.
3. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating and receives data at pin SI_1.
4. After data reception is complete, bit IRRS1 in interrupt request register 2 (IRR2) is set to 1.
5. Read the received data from SDRL and SDRU, as follows.

   8-bit transfer mode: SDRL

   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL
6. After data reception is complete, an overrun occurs if the serial clock continues to be input; no data is received and the SCSR1 overrun error flag (bit ORER) is set to 1.

Simultaneous Transmit/Receive: A simultaneous transmit/receive operation is carried out as follows.

1. Set bits SO1, SI1, and SCK1 to 1 in PMR3 to select the SO_1, SI_1, and SCK_1 pin functions. If necessary, set bit POF1 in PMR7 for NMOS open-drain output at pin SO_1.
2. Clear bit SNC1 in SCR1 to 0, set bit SNC0 to 0 or 1, and clear bit MRKON to 0, designating 8- or 16-bit synchronous transfer mode. Select the serial clock in bits CKS3 to CKS0. Writing data to SCR1 when bit MRKON in SCR1 is cleared to 0 initializes the internal state of SCI1.
3. Write transmit data in SDRL and SDRU, as follows.
   8-bit transfer mode: SDRL
   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL
4. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating. Transmit data is output at pin SO_1. Receive data is input at pin SI_1.
5. After data transmission and reception are complete, bit IRRS1 in IRR2 is set to 1.
6. Read the received data from SDRL and SDRU, as follows.
   8-bit transfer mode: SDRL
   16-bit transfer mode: Upper byte in SDRU, lower byte in SDRL

When an internal clock is used, a serial clock is output from pin SCK_1 in synchronization with the transmit data. After data transmission is complete, the serial clock is not output until the next time the start flag is set to 1. During this time, pin SO_1 continues to output the value of the last bit transmitted.

When an external clock is used, data is transmitted and received in synchronization with the serial clock input at pin SCK_1. After data transmission and reception are complete, an overrun occurs if the serial clock continues to be input; no data is transmitted or received and the SCSR1 overrun error flag (bit ORER) is set to 1.

While transmission is stopped, the output value of pin SO_1 can be changed by rewriting bit SOL in SCSR1.

10.2.4 Operation in SSB Mode

SSB communication uses two lines, SCL (Serial Clock) and SDA (Serial Data), and enables multiple ICs to be connected as shown in figure 10.3.

In SSB mode, TAIL MARK is sent after an 8- or 16-bit data transfer. HOLD TAIL or LATCH TAIL can be selected as TAIL MARK.

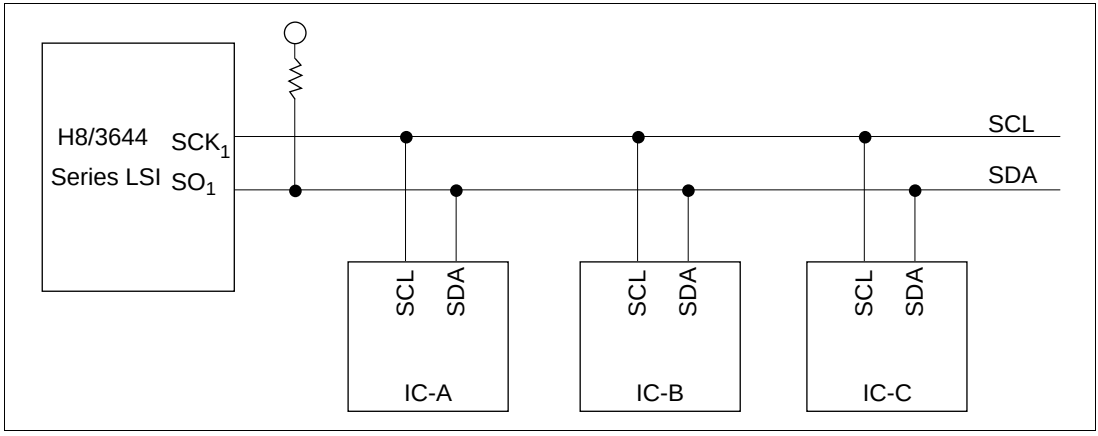


Figure 10.3 Example of SSB Connection

Clock: The transfer clock can be selected from eight internal clocks or an external clock, but since the H8/3644 Series uses clock output, an external clock should not be selected. The transfer rate can be selected by bits CKS2 to CKS0 in SCR1. Since this is also the TAIL MARK transfer rate, the setting should be made to give a transfer clock cycle of at least 2 µs.

Data Transfer Format: Figure 10.4 shows the SCI1 transfer format. Data is sent starting from the least significant bit, in LSB-first format. TAIL MARK is sent after an 8- or 16-bit data transfer.

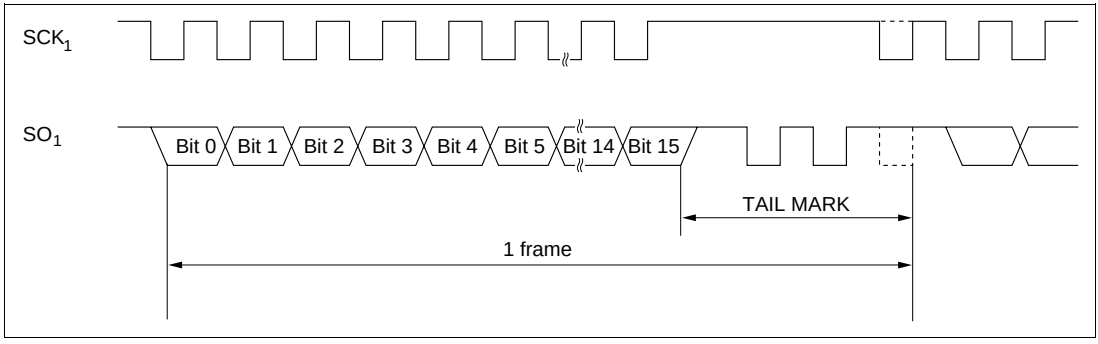


Figure 10.4 Transfer Format (When SNC1 = 0, SNC0 = 1, MRKON = 1)

TAIL MARK: TAIL MARK can be either HOLD TAIL or LATCH TAIL. The output waveforms of HOLD TAIL and LATCH TAIL are shown in figure 10.5. Time t in the figure is determined by the transfer clock cycle set in bits CKS2 to CKS0 in SCR1.

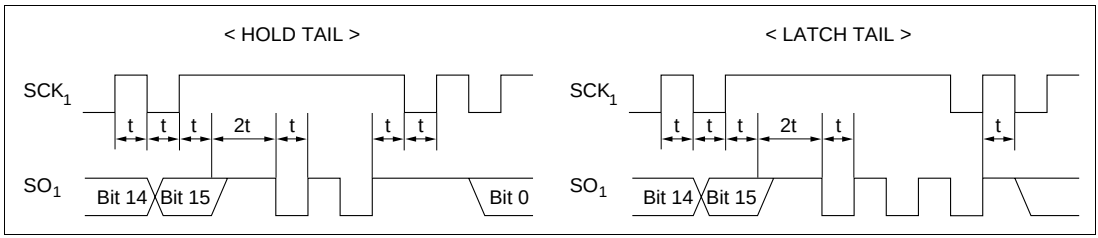


Figure 10.5 HOLD TAIL and LATCH TAIL Waveforms

Transmitting: A transmit operation is carried out as follows.

1. Set bit SOL in SCSR1 to 1.
2. Set bits SO1 and SCK1 to 1 in PMR3 to select the $S0_1$ and SCK_1 pin functions. Set bit POF1 in PMR7 to 1 for NMOS open-drain output at pin SO_1.
3. Clear bit SNC1 in SCR1 to 0 and set bit SNC0 to 0 or 1, designating 8-bit mode or 16-bit mode. Set bit MRKON in SCR1 to 1, selecting SSB mode.
4. Write transmit data in SDRL and SDRU as follows, and select TAIL MARK with bit LTCH in SCR1.

   8-bit mode: SDRL

   16-bit mode: Upper byte in SDRU, lower byte in SDRL
5. Set the SCSR1 start flag (STF) to 1. SCI1 starts operating and outputs transmit data at pin $S0_1$.
6. After 8- or 16-bit data transmission is complete, bit STF in SCSR1 is cleared to 0 and bit IRRS1 in interrupt request register 2 (IRRS2) is set to 1. The selected TAIL MARK is output after the data transmission. During TAIL MARK output, bit MTRF in SCSR1 is set to 1.

Data can be sent continuously by repeating steps 4 to 6. Check that SCI1 is in the idle state before rewriting bit MRKON in SCR1.

10.2.5 Interrupts

SCI1 can generate an interrupt at the end of a data transfer.

When an SCI1 transfer is complete, bit IRRS1 in interrupt request register 2 (IRR2) is set to 1. SCI1 interrupt requests can be enabled or disabled by bit IENS1 of interrupt enable register 2 (IENR2).

For further details, see 3.3, Interrupts.

10.3 SCI3

10.3.1 Overview

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.

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 12 data transfer formats.

Data length	7 or 8 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 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

Block Diagram

Figure 10.6 shows a block diagram of SCI3.

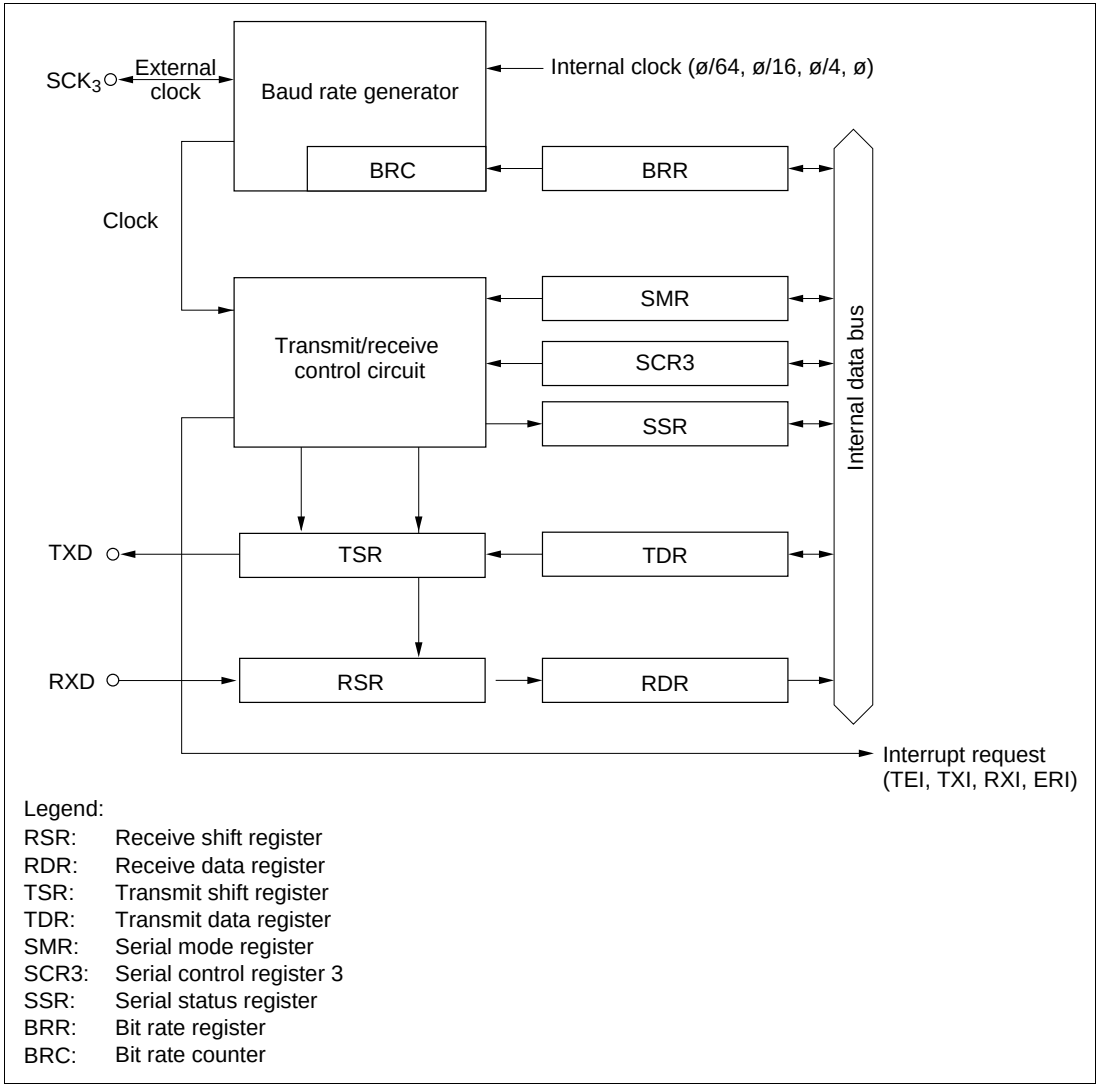


Figure 10.6 SCI3 Block Diagram

Pin Configuration

Table 10.4 shows the SCI3 pin configuration.

Table 10.4 Pin Configuration

Name	Abbrev.	I/O	Function
SCI3 clock	SCK_3	I/O	SCI3 clock input/output
SCI3 receive data input	RXD	Input	SCI3 receive data input
SCI3 transmit data output	TXD	Output	SCI3 transmit data output

Register Configuration

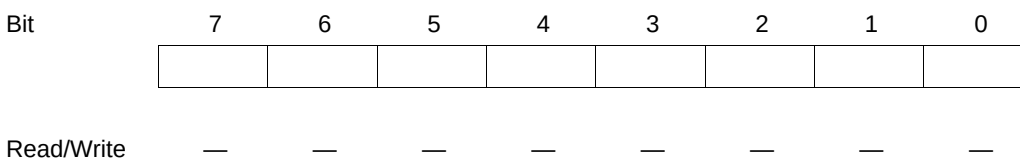
Table 10.5 shows the SCI3 register configuration.

Table 10.5 Registers

Name	Abbrev.	R/W	Initial Value	Address
Serial mode register	SMR	R/W	H'00	H'FFA8
Bit rate register	BRR	R/W	H'FF	H'FFA9
Serial control register 3	SCR3	R/W	H'00	H'FFAA
Transmit data register	TDR	R/W	H'FF	H'FFAB
Serial status register	SSR	R/W	H'84	H'FFAC
Receive data register	RDR	R	H'00	H'FFAD
Transmit shift register	TSR	Protected	—	—
Receive shift register	RSR	Protected	—	—
Bit rate counter	BRC	Protected	—	—

10.3.2 Register Descriptions

Receive Shift Register (RSR)



RSR is a register used to receive serial data. Serial data input to RSR from the RXD 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.

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 enabled for reception. 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, watch, subactive, or subsleep mode.

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 pin in order, starting from the LSB (bit 0). When one byte of data is transmitted, the next byte of transmit data is transferred from TDR to TSR, 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.

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, watch, subactive, or subsleep mode.

Serial Mode Register (SMR)

Bit	7	6	5	4	3	2	1	0
	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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, watch, subactive, or subsleep 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	(initial value)
1	7-bit data*	

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is 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	(initial value)
1	Parity bit addition and checking enabled*	

Note: * 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.

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 is 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 10.3.6.

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

Bits 1 and 0—Clock Select 1, 0 (CKS1, CKS0): Bits 1 and 0 choose ø/64, ø/16, ø/4, 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 Bit Rate Register (BRR).

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	ø clock	(initial value)
	1	ø/4 clock	
1	0	ø/16 clock	
	1	ø/16 clock	

Serial Control Register 3 (SCR3)

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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, watch, subactive, or subsleep 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 pin is transmit data pin)*3	(initial value)
1	Transmit operation enabled*2 (TXD pin is transmit data pin) *3	

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 to decide the transmission format before setting bit TE to 1.
3. When bit TXD in PMR7 is set to 1. When bit TXD is cleared to 0, the TXD pin functions as an I/O port regardless of the TE bit setting.

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) (initial value) Clearing conditions: When data is received in which the multiprocessor bit is set to 1
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 (SCR) 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_3 pin. These bits determine whether the SCK_3 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 10.10 in 10.3.3, Operation.

		Description		
Bit 1: CKE1	**Bit 0: CKE0**	**Communication Mode**	**Clock Source**	**SCK_3 Pin Function**
0	0	Asynchronous	Internal clock	I/O port*1
		Synchronous	Internal clock	Serial clock output*1
	1	Asynchronous	Internal clock	Clock output*2
		Synchronous	Reserved	
1	0	Asynchronous	External clock	Clock input*3
		Synchronous	External clock	Serial clock input
	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.

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 by the CPU at any time, but only a write of 1 is possible to bits TDRE, RDRF, OER, PER, and FER. In order to clear these bits by writing 0, 1 must first be read.

Bits TEND and MPBR are read-only bits, and cannot be modified.

SSR is initialized to H'84 upon reset, and in standby, watch, subactive, or subsleep 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	(initial value)
	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	

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	(initial value)
	Clearing conditions: After reading OER = 1, cleared by writing 0 to OER	
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]	(initial value)
	Clearing conditions: After reading FER = 1, cleared by writing 0 to FER	
1	A framing error has occurred during reception*[2]	
	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]	(initial value)
	Clearing conditions: After reading PER = 1, cleared by writing 0 to PER	
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 PER 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	

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, watch, subactive, or subsleep mode.

Table 10.6 shows examples of BRR settings in asynchronous mode. The values shown are for active (high-speed) mode.

Table 10.6 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode)

	OSC (MHz)											
	2			2.4576			4			4.194304		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	70	+0.03	1	86	+0.31	1	141	+0.03	1	148	–0.04
150	0	207	+0.16	0	255	0	1	103	+0.16	1	108	+0.21
300	0	103	+0.16	0	127	0	0	207	+0.16	0	217	+0.21
600	0	51	+0.16	0	63	0	0	103	+0.16	0	108	+0.21
1200	0	25	+0.16	0	31	0	0	51	+0.16	0	54	–0.70
2400	0	12	+0.16	0	15	0	0	25	+0.16	0	26	+1.14
4800	—	—	—	0	7	0	0	12	+0.16	0	13	–2.48
9600	—	—	—	0	3	0	—	—	—	0	6	–2.48
19200	—	—	—	0	1	0	—	—	—	—	—	—
31250	0	0	0	—	—	—	0	1	0	—	—	—
38400	—	—	—	0	0	0	—	—	—	—	—	—

Table 10.6 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)

	OSC (MHz)											
	4.9152			6			7.3728			8		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	174	–0.26	1	212	+0.03	2	64	+0.70	2	70	+0.03
150	1	127	0	1	155	+0.16	1	191	0	1	207	+0.16
300	0	255	0	1	77	+0.16	1	95	0	1	103	+0.16
600	0	127	0	0	155	+0.16	0	191	0	0	207	+0.16
1200	0	63	0	0	77	+0.16	0	95	0	0	103	+0.16
2400	0	31	0	0	38	+0.16	0	47	0	0	51	+0.16
4800	0	15	0	0	19	–2.34	0	23	0	0	25	+0.16
9600	0	7	0	0	9	–2.34	0	11	0	0	12	+0.16
19200	0	3	0	0	4	–2.34	0	5	0	—	—	—
31250	—	—	—	0	2	0	—	—	—	0	3	0
38400	0	1	0	—	—	—	0	2	0	—	—	—

Table 10.6 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)

	OSC (MHz)					
	9.8304			10		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)
110	2	86	+0.31	2	88	–0.25
150	1	255	0	2	64	+0.16
300	1	127	0	1	129	+0.16
600	0	255	0	1	64	+0.16
1200	0	127	0	0	129	+0.16
2400	0	63	0	0	64	+0.16
4800	0	31	0	0	32	–1.36
9600	0	15	0	0	15	+1.73
19200	0	7	0	0	7	+1.73
31250	0	4	–1.70	0	4	0
38400	0	3	0	0	3	+1.73

Notes: 1. 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:

$$N = \frac{OSC}{(64 \times 2^{2n} \times B)} \times 10^6 - 1$$

where

B: Bit rate (bit/s)
N: Baud rate generator BRR setting ($0 \leq N \leq 255$)
OSC: Value of ϕ_{OSC} (MHz)
n: Baud rate generator input clock number (n = 0, 1, 2, or 3)
(The relation between n and the clock is shown in table 10.7.)

Table 10.7 Relation between n and Clock

		SMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	ø	0	0
1	ø/4	0	1
2	ø16	1	0
3	ø/64	1	1

3. The error in table 10.6 is the value obtained from the following equation, rounded to two decimal places.

$$\text{Error (\%)} = \frac{\text{B (rate obtained from n, N, OSC)} - \text{R (bit rate in left-hand column in table 10.6)}}{\text{R (bit rate in left-hand column in table 10.6)}} \times 100$$

Table 10.8 shows the maximum bit rate for each frequency. The values shown are for active (high-speed) mode.

Table 10.8 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

		Setting	
OSC (MHz)	**Maximum Bit Rate (bits/s)**	**n**	**N**
2	31250	0	0
2.4576	38400	0	0
4	62500	0	0
4.194304	65536	0	0
4.9152	76800	0	0
6	93750	0	0
7.3728	115200	0	0
8	125000	0	0
9.8304	153600	0	0
10	156250	0	0

Table 10.9 shows examples of BRR settings in synchronous mode. The values shown are for active (high-speed) mode.

Table 10.9 Examples of BRR Settings for Various Bit Rates (Synchronous Mode)

Bit Rate (bits/s)	OSC (MHz)							
	2		4		8		10	
	n	N	n	N	n	N	n	N
110	—	—	—	—	—	—	—	—
250	1	249	2	124	2	249	—	—
500	1	124	1	249	2	124	—	—
1k	0	249	1	124	1	249	—	—
2.5k	0	99	0	199	1	99	1	124
5k	0	49	0	99	0	199	0	249
10k	0	24	0	49	0	99	0	124
25k	0	9	0	19	0	39	0	49
50k	0	4	0	9	0	19	0	24
100k	—	—	0	4	0	9	—	—
250k	0	0*	0	1	0	3	0	4
500k			0	0*	0	1	—	—
1M					0	0*	—	—
2.5M								

Blank: Cannot be set.

— : A setting can be made, but an error will result.

* : Continuous transmission/reception is not possible.

Note: The value set in BRR is given by the following equation:

$$N = \frac{OSC}{(8 \times 2^{2n} \times B)} \times 10^6 - 1$$

where

B: Bit rate (bit/s)

N: Baud rate generator BRR setting ($0 \leq N \leq 255$)

OSC: Value of $ø_{OSC}$ (MHz)

n: Baud rate generator input clock number (n = 0, 1, 2, or 3)
(The relation between n and the clock is shown in table 10.10.)

Table 10.10 Relation between n and Clock

		SMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	ø	0	0
1	ø/4	0	1
2	ø16	1	0
3	ø/64	1	1

10.3.3 Operation

SCI3 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 10.11.

The clock source for SCI3 is determined by bit COM in SMR and bits CKE1 and CKE0 in SCR3, as shown in table 10.12.

Asynchronous Mode

- Choice of 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.)

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 10.11 SMR Settings and Corresponding Data Transfer Formats

SMR Setting						Communication Format			
Bit 7: COM	Bit 6: CHR	Bit 2: MP	Bit 5: PE	Bit 3: STOP	Mode	Data Length	Multipro-cessor Bit	Parity Bit	Stop Bit Length
0	0	0	0	0	Asynchronous mode	8-bit data	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	Asynchronous mode (multiprocessor format)	8-bit data	Yes	No	1 bit
			*	1					2 bits
	1		*	0		7-bit data			1 bit
			*	1					2 bits
1	*	0	*	*	Synchronous mode	8-bit data	No	No	No

Note: * Don't care

Table 10.12 SMR and SCR3 Settings and Clock Source Selection

SMR	SCR3			Transmit/Receive Clock	
Bit 7: COM	Bit 1: CKE1	Bit 0: CKE0	Mode	Clock Source	SCK_3 Pin Function
0	0	0	Asynchronous mode	Internal	I/O port (SCK_3 pin not used)
		1			Outputs clock with same frequency as bit rate
	1	0		External	Outputs 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			

Interrupts and Continuous Transmission/Reception: SCI3 can carry out continuous reception using RXI and continuous transmission using TXI. These interrupts are shown in table 10.13.

Table 10.13 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 10.7 (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 10.7 (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 10.7 (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.

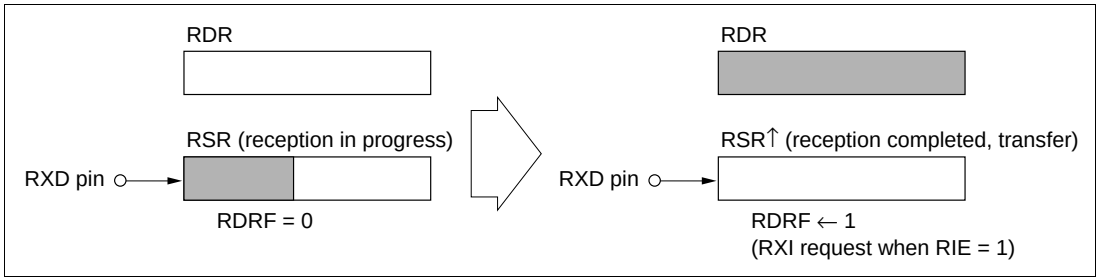


Figure 10.7 (a) RDRF Setting and RXI Interrupt

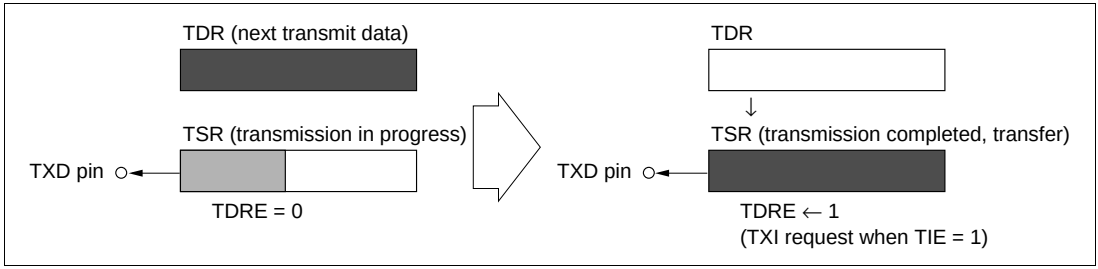


Figure 10.7 (b) TDRE Setting and TXI Interrupt

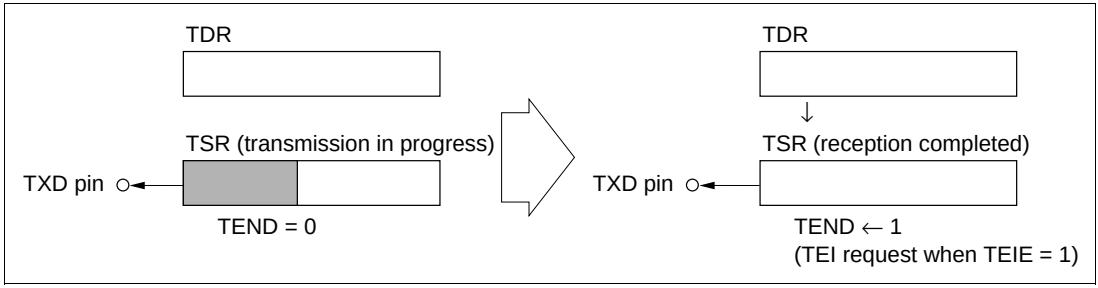


Figure 10.7 (c) TEND Setting and TEI Interrupt

10.3.4 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.

Data Transfer Format: The general data transfer format in asynchronous communication is shown in figure 10.8.

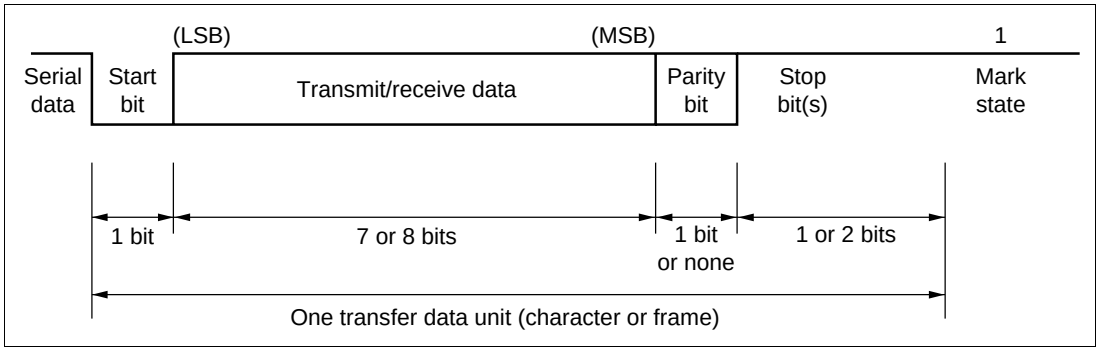


Figure 10.8 Data Format in Asynchronous Communication

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 10.14 shows the 12 data transfer formats that can be set in asynchronous mode. The format is selected by the settings in the serial mode register (SMR).

Table 10.14 Data Transfer Formats (Asynchronous Mode)

SMR Settings				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	1	0	0	S 8-bit data P STOP
0	1	0	1	S 8-bit data P STOP STOP
1	0	0	0	S 7-bit data STOP
1	0	0	1	S 7-bit data STOP STOP
1	1	0	0	S 7-bit data P STOP
1	1	0	1	S 7-bit data P STOP STOP
0	*	1	0	S 8-bit data MPB STOP
0	*	1	1	S 8-bit data MPB STOP STOP
1	*	1	0	S 7-bit data MPB STOP
1	*	1	1	S 7-bit data MPB STOP STOP

* Don't care

Legend:

S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

Clock: Either an internal clock generated by the baud rate generator or an external clock input at the SCK_3 pin can be selected as the SCI3 transmit/receive clock. The selection is made by means of bit COM in SMR and bits CKE1 and CKE0 in SCR3. See table 10.12 for details on clock source selection.

When an external clock is input at the SCK_3 pin, a clock with a frequency of 16 times the bit rate used should be input.

When SCI3 operates on an internal clock, the clock can be output at the SCK_3 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 10.9.

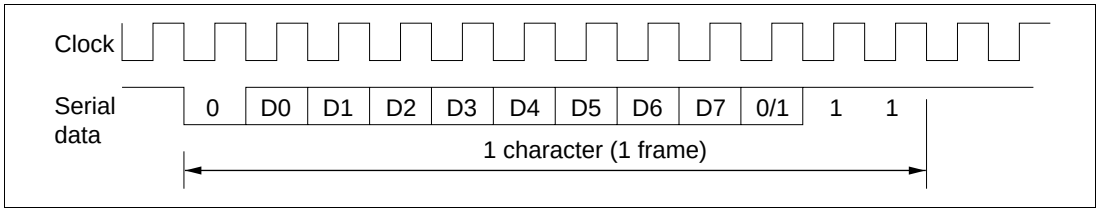


Figure 10.9 Phase Relationship between Output Clock and Transfer Data (Asynchronous Mode) (8-Bit Data, Parity, 2 Stop Bits)

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.

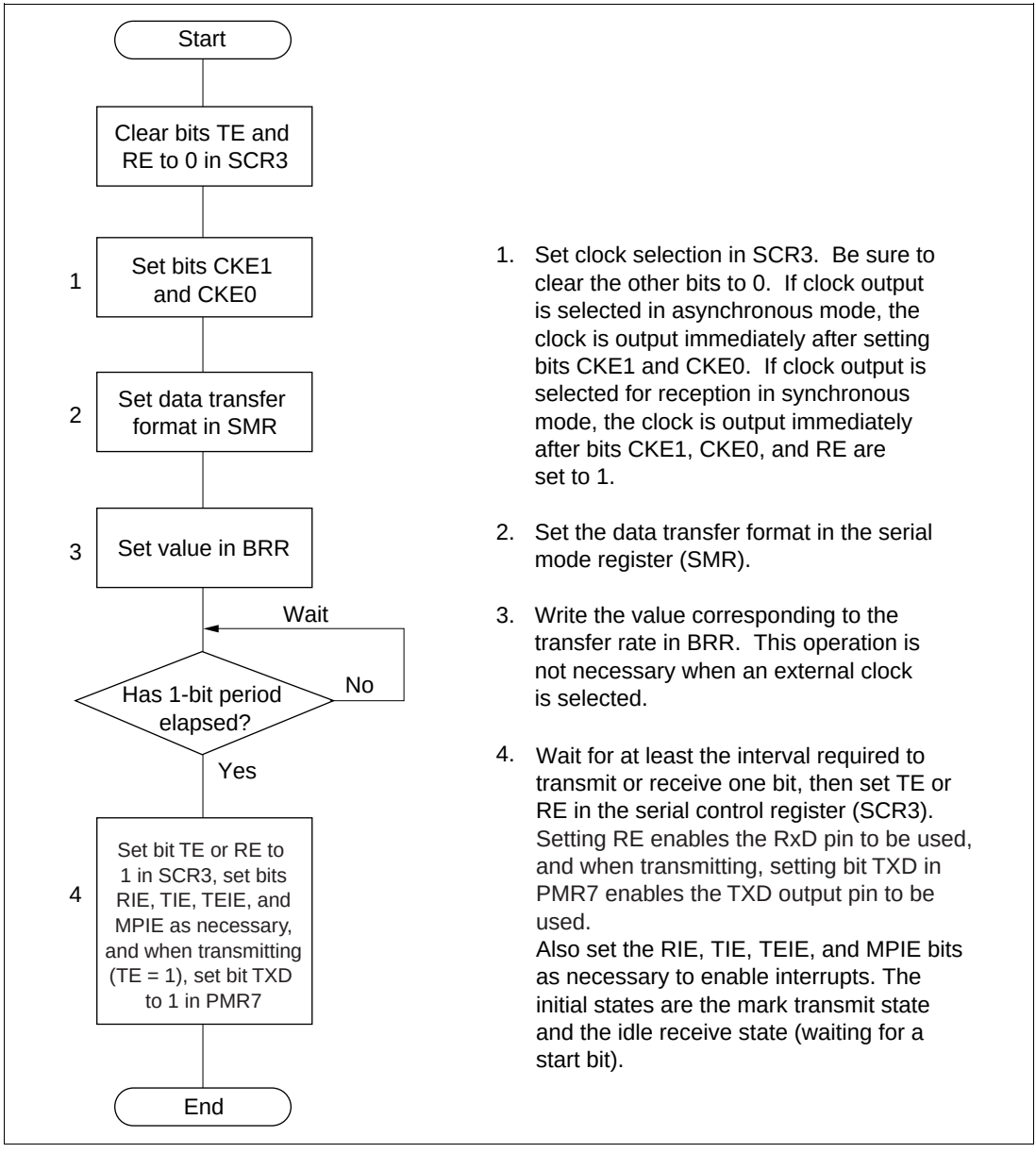
Note: If the operation mode or data transfer format is changed, bits TE and RE must first be cleared to 0.

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 10.10 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.

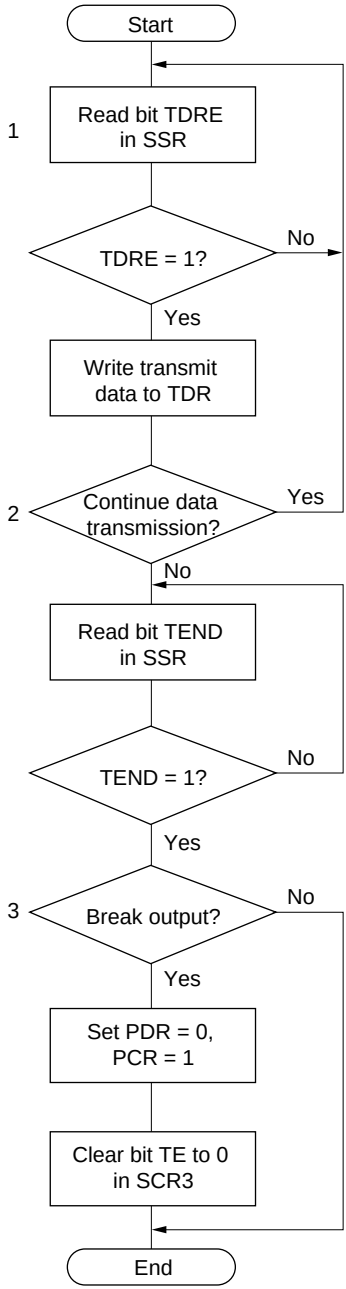
2. Set the data transfer format in the serial mode register (SMR).

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 the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR3). Setting RE enables the RxD pin to be used, and when transmitting, setting bit TXD in PMR7 enables the TXD output pin to be used.
   Also set the RIE, TIE, TEIE, and MPIE bits as necessary to enable interrupts. The initial states are the mark transmit state and the idle receive state (waiting for a start bit).

Figure 10.10 Example of SCI3 Initialization Flowchart

Transmitting: Figure 10.11 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.

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 TXD in PMR7 and bit TE in SCR3 to 0.

Figure 10.11 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 pin using the relevant data transfer format in table 10.14. 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 is set to 1, and 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 10.12 shows an example of the operation when transmitting in asynchronous mode.

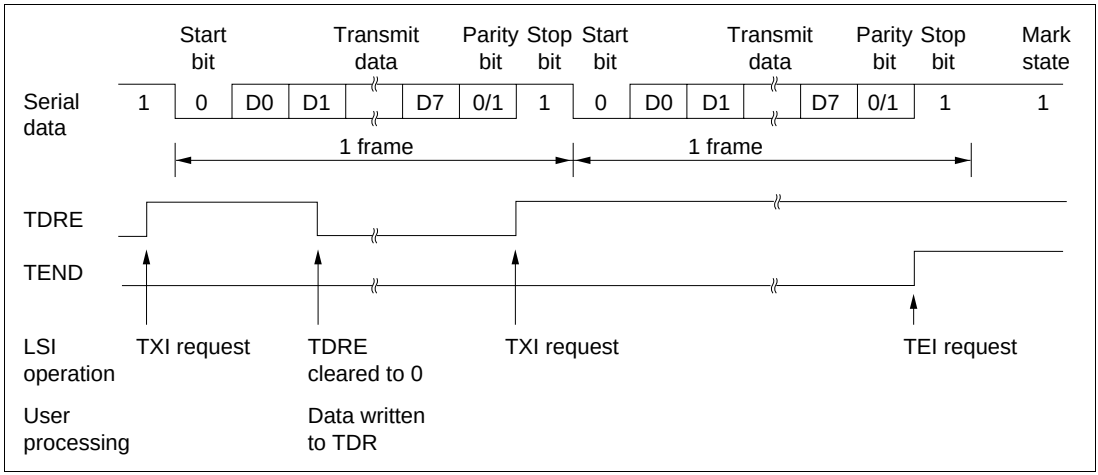
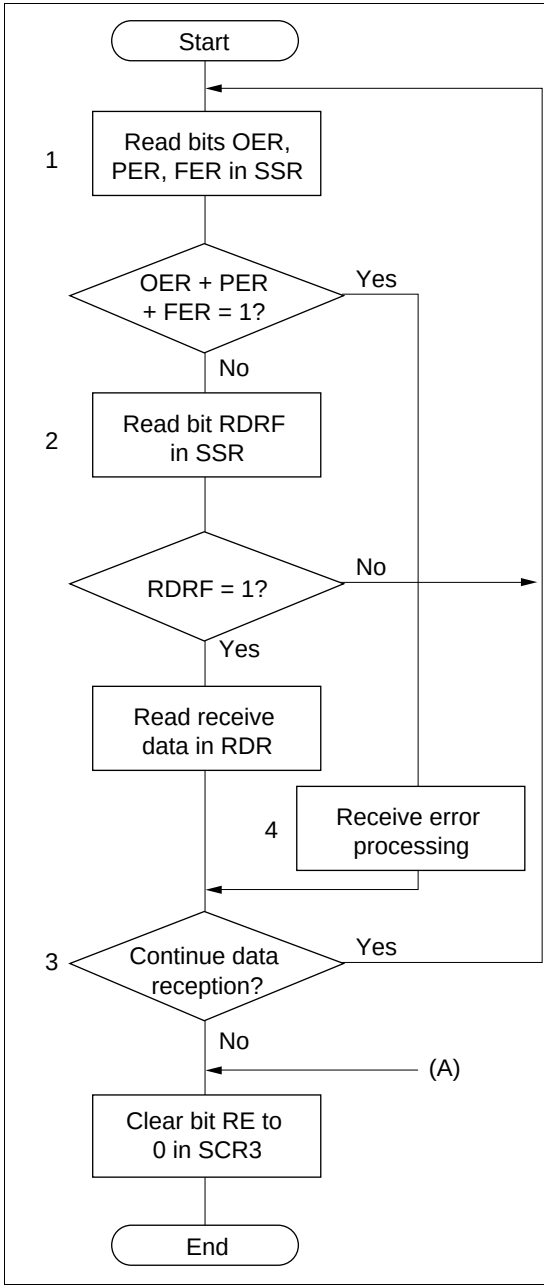


Figure 10.12 Example of Operation when Transmitting in Asynchronous Mode (8-Bit Data, Parity, 1 Stop Bit)

Receiving: Figure 10.13 shows an example of a flowchart for data reception. This procedure should be followed for data reception after initializing SCI3.

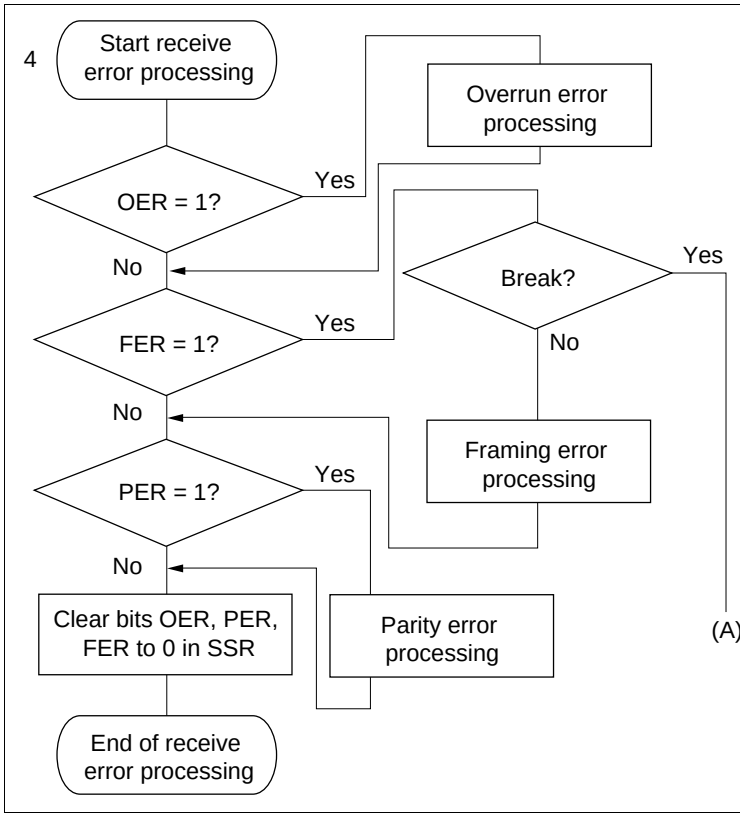


1. Read bits OER, PER, and FER in the serial status register (SSR) to determine if there is an error. If a receive error has occurred, execute receive error processing.

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 10.13 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 pin.

Figure 10.13 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 10.14. 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 1, 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 10.15 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 10.15 Receive Error Detection Conditions and Receive Data Processing

Receive Error	Abbreviation	Detection Conditions	Received 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 10.14 shows an example of the operation when receiving in asynchronous mode.

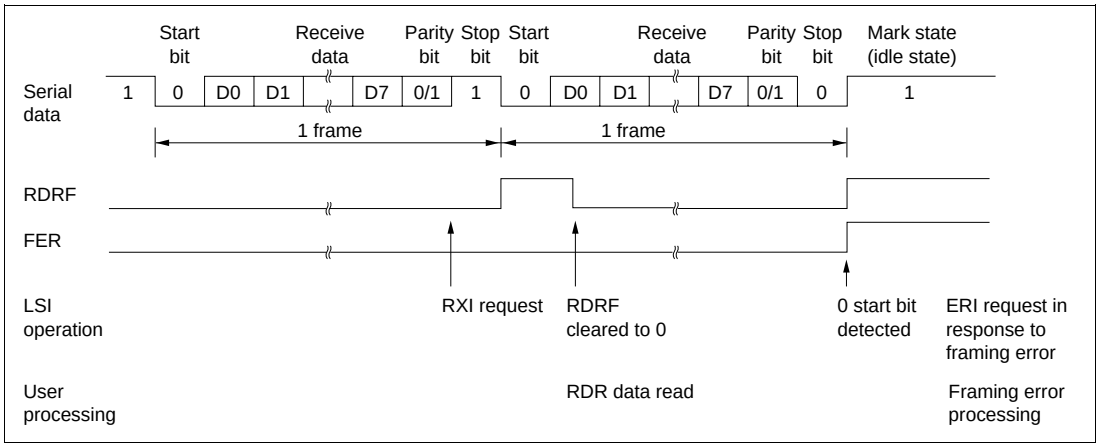


Figure 10.14 Example of Operation when Receiving in Asynchronous Mode (8-Bit Data, Parity, 1 Stop Bit)

10.3.5 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.

Data Transfer Format: The general data transfer format in synchronous communication is shown in figure 10.15.

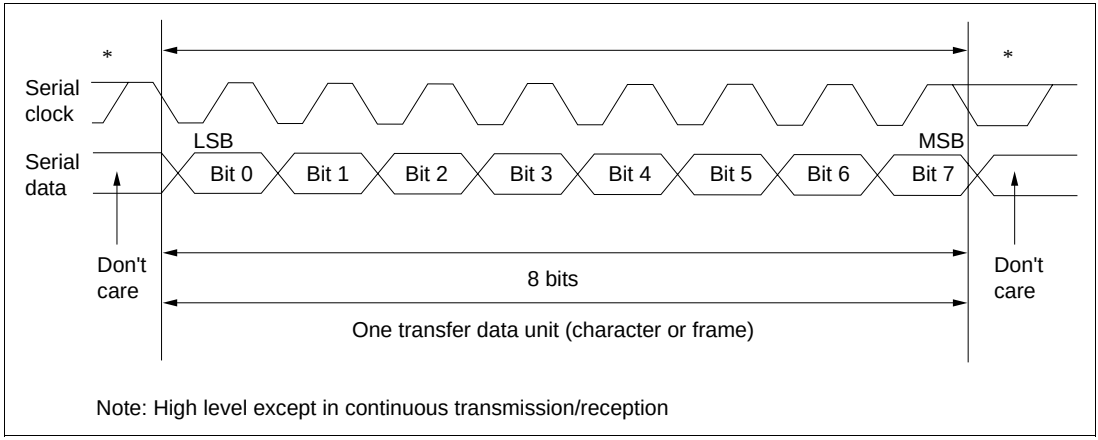


Figure 10.15 Data Format in Synchronous Communication

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.

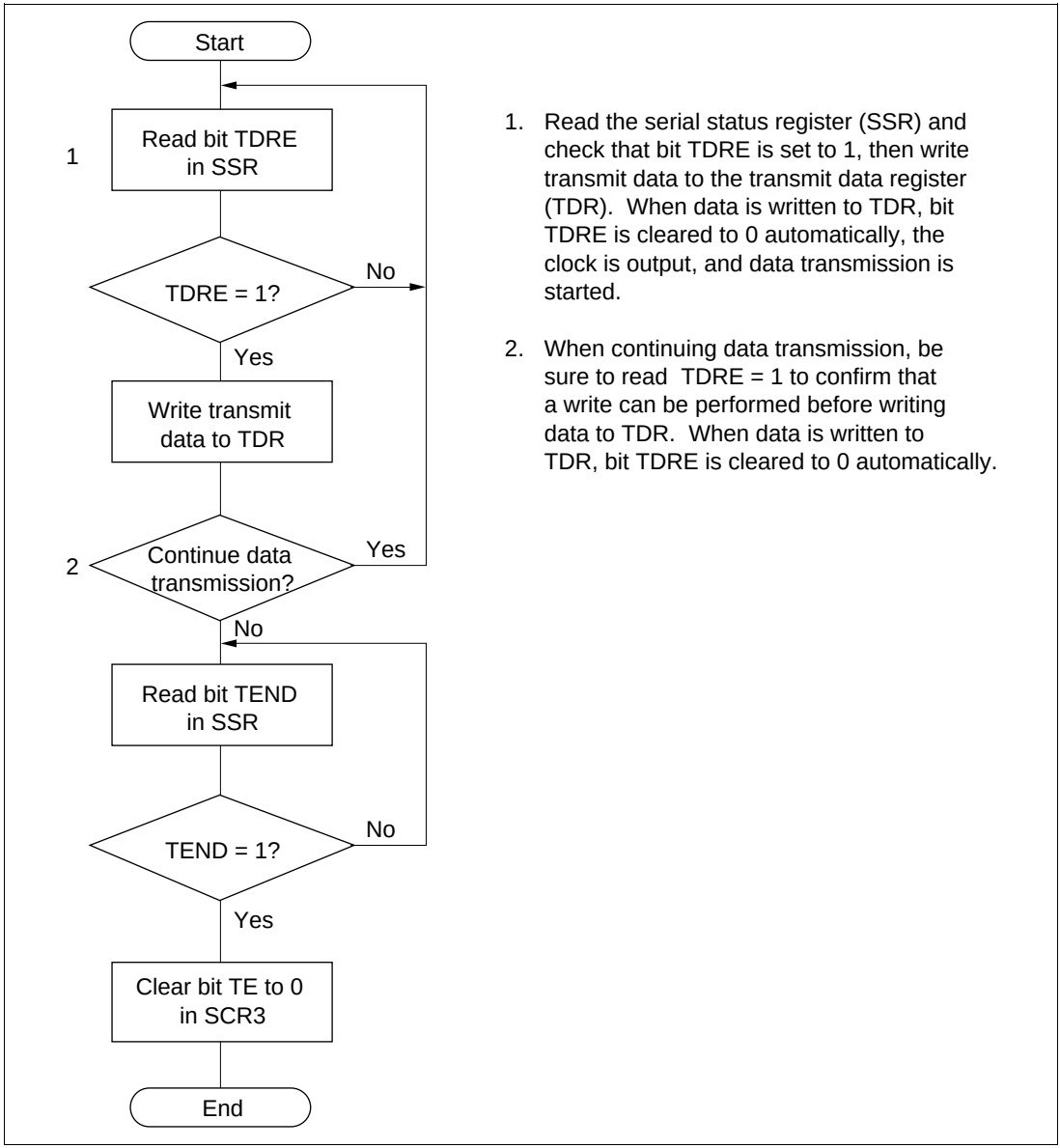
Clock: Either an internal clock generated by the baud rate generator or an external clock input at the SCK_3 pin can be selected as the SCI3 serial clock. The selection is made by means of bit COM in SMR and bits CKE1 and CKE0 in SCR3. See table 10.12 for details on clock source selection.

When SCI3 operates on an internal clock, the serial clock is output at the SCK_3 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.

Data Transfer Operations

SCI3 Initialization: Data transfer on SCI3 first of all requires that SCI3 be initialized as described in 10.3.4, SCI3 Initialization, and shown in figure 10.10.

Transmitting: Figure 10.16 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, the clock is output, and data transmission is started.

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.

Figure 10.16 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 TXD 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_3 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 (OER, FER, and PER) are all cleared to 0 before a transmit operation.

Figure 10.17 shows an example of the operation when transmitting in synchronous mode.

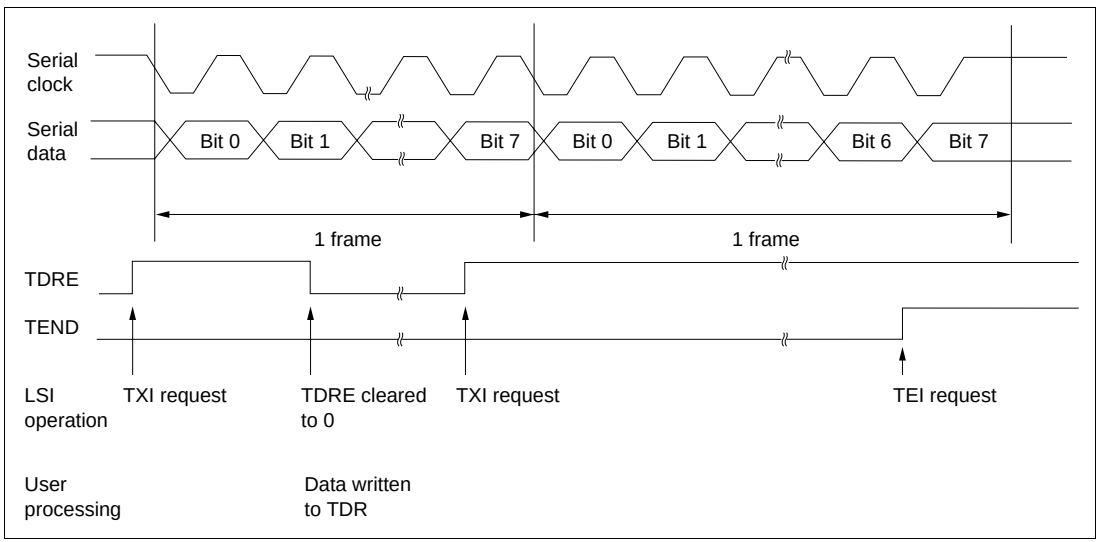


Figure 10.17 Example of Operation when Transmitting in Synchronous Mode

Receiving: Figure 10.18 shows an example of a flowchart for data reception. This procedure should be followed for data reception after initializing SCI3.

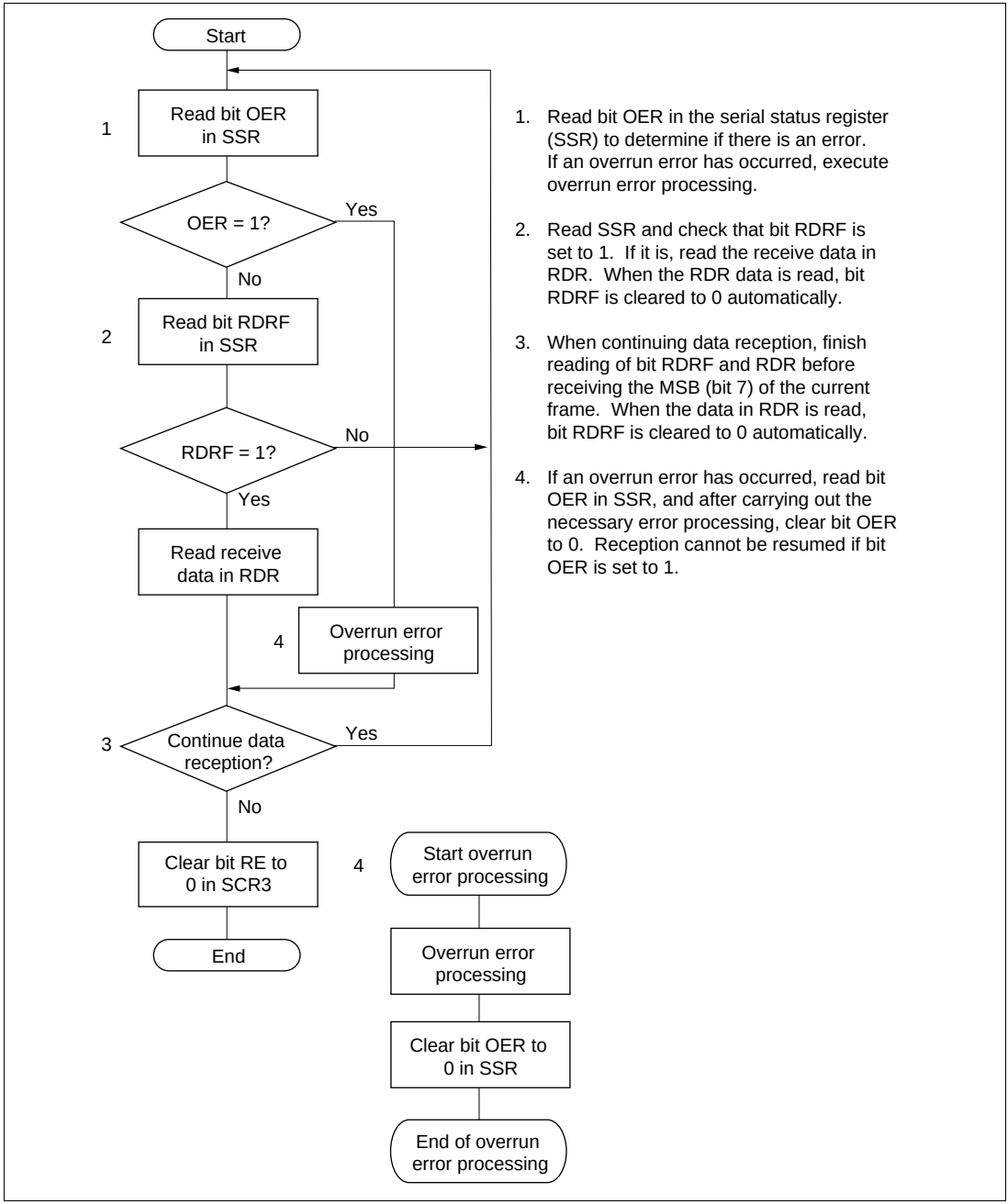


Figure 10.18 Example of Data Reception Flowchart (Synchronous Mode)

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 10.15 for the conditions for detecting an overrun 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 10.19 shows an example of the operation when receiving in synchronous mode.

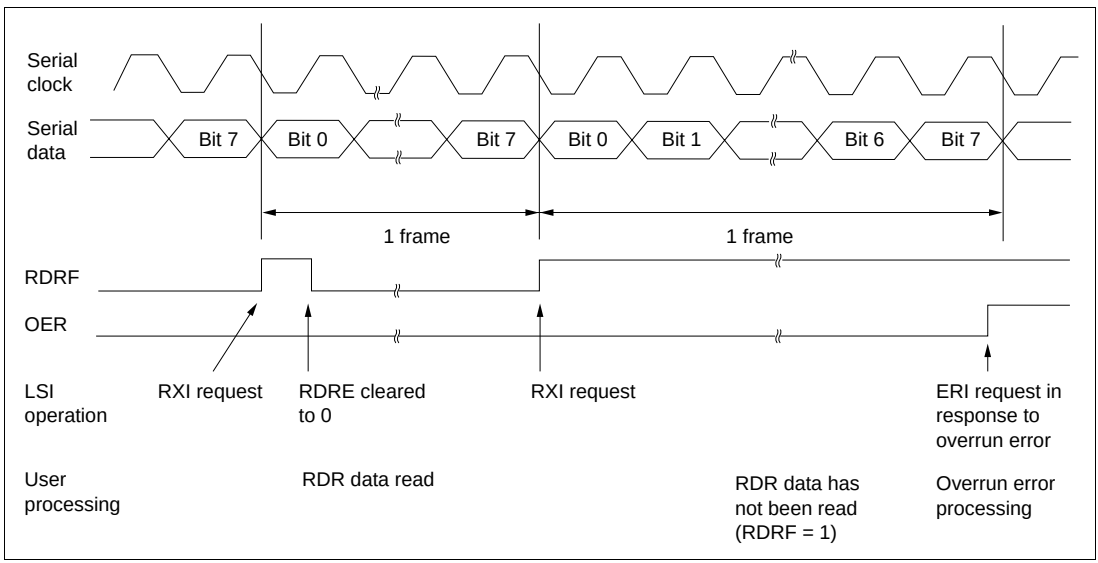


Figure 10.19 Example of Operation when Receiving in Synchronous Mode

Simultaneous Transmit/Receive: Figure 10.20 shows an example of a flowchart for a simultaneous transmit/receive operation. This procedure should be followed for simultaneous transmission/reception after initializing SCI3.

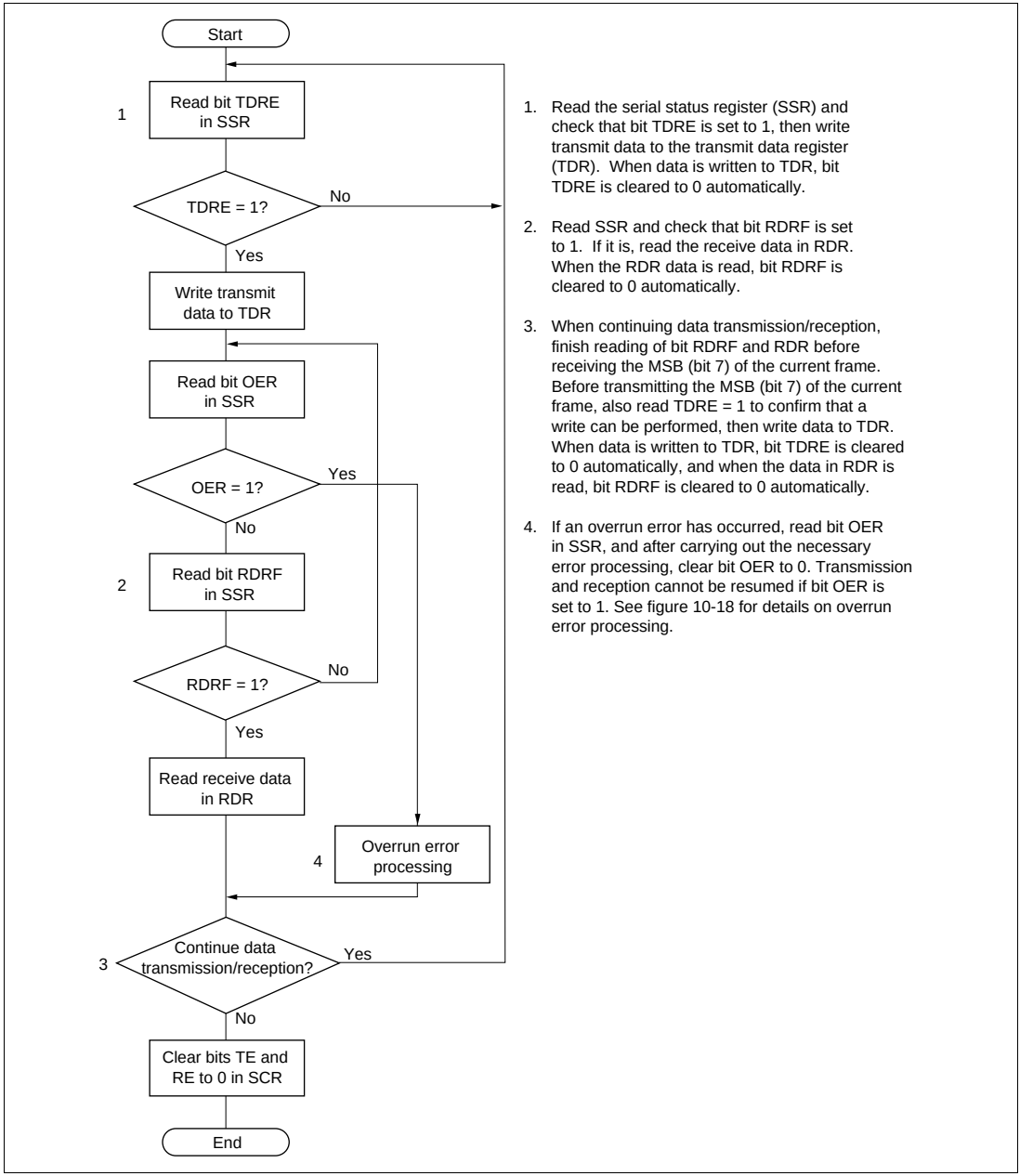


Figure 10.20 Example of Simultaneous Data Transmission/Reception Flowchart (Synchronous Mode)

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 with a single instruction.

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 with a single instruction.

10.3.6 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 10.21 shows an example of communication between processors using the multiprocessor format.

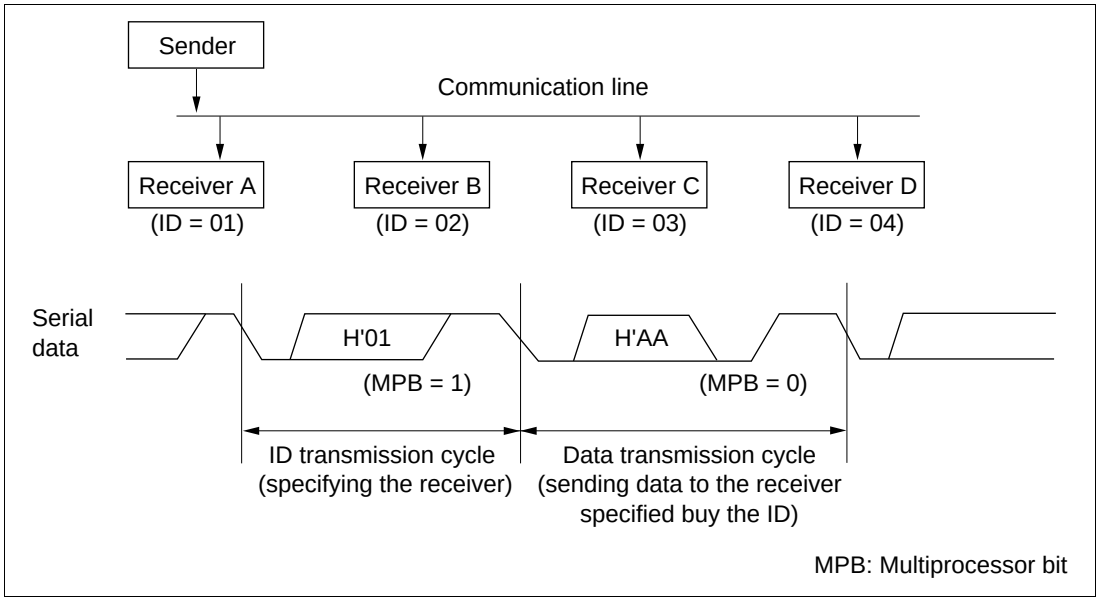
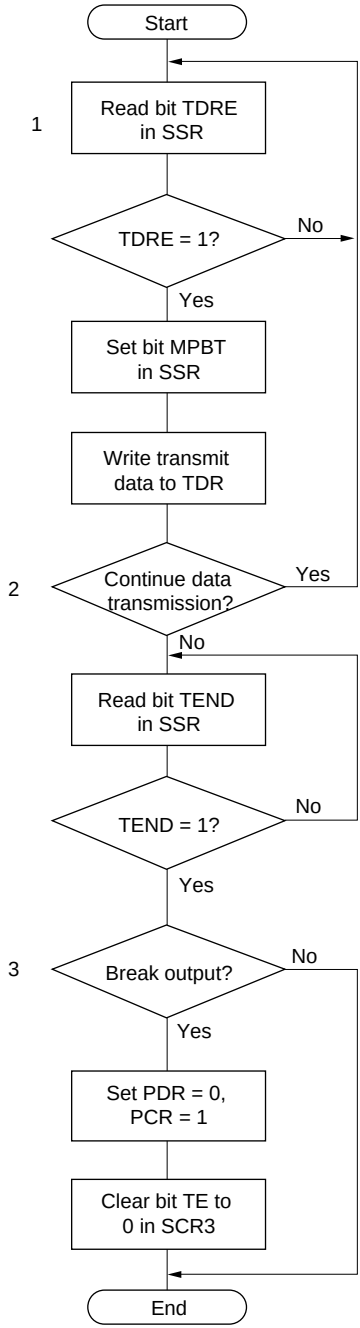


Figure 10.21 Example of Inter-Processor Communication Using Multiprocessor Format (Sending Data H'AA to Receiver A)

There is a choice of four data transfer formats. If a multiprocessor format is specified, the parity bit specification is invalid. See table 10.14 for details.

For details on the clock used in multiprocessor communication, see 10.3.4, Operation in Synchronous Mode.

Multiprocessor Transmitting: Figure 10.22 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 10.22 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 10.14. 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 is set to 1, and 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 10.23 shows an example of the operation when transmitting using the multiprocessor format.

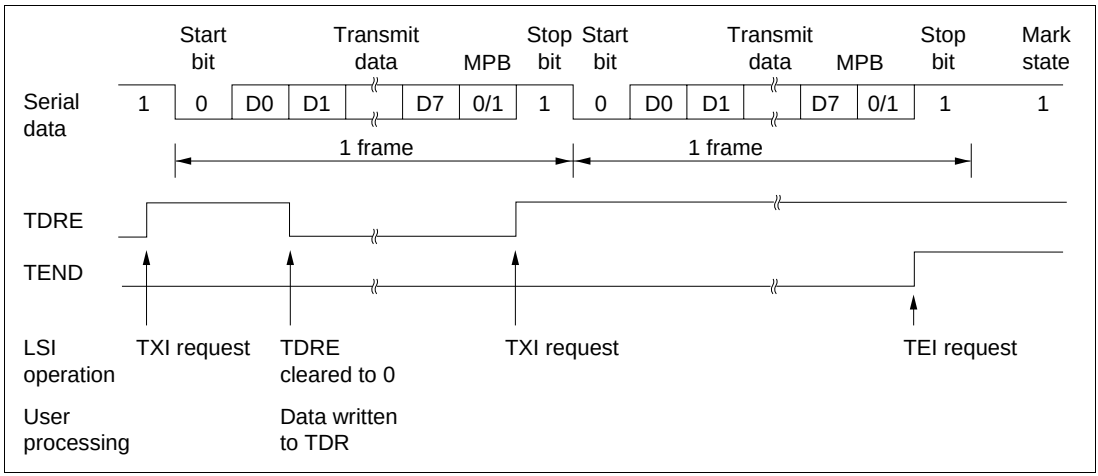
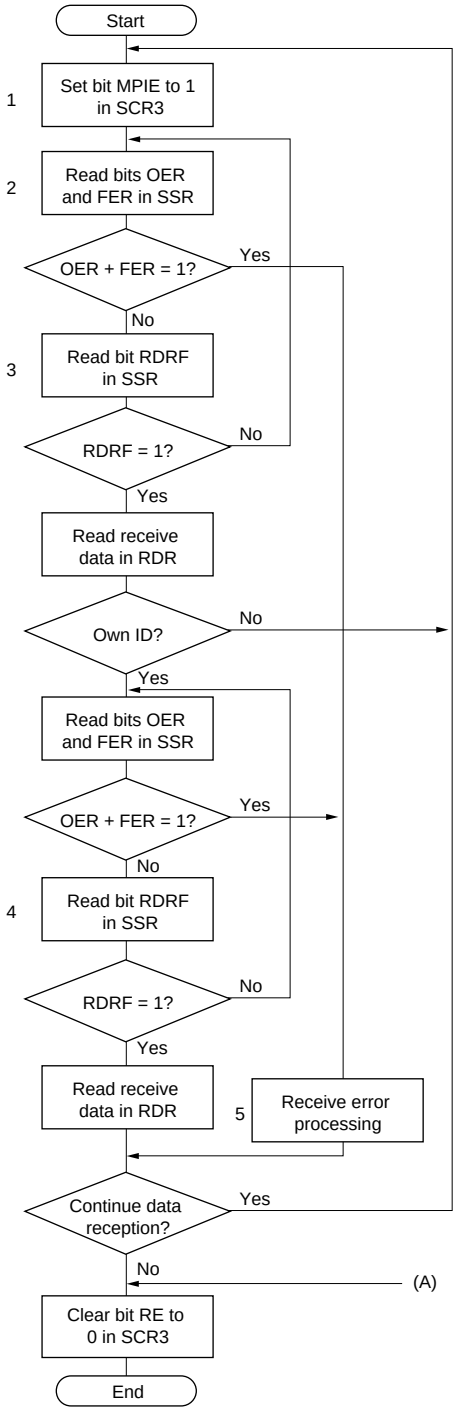


Figure 10.23 Example of Operation when Transmitting using Multiprocessor Format (8-Bit Data, Multiprocessor Bit, 1 Stop Bit)

Multiprocessor Receiving: Figure 10.24 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 pin.

Figure 10.24 Example of Multiprocessor Data Reception Flowchart

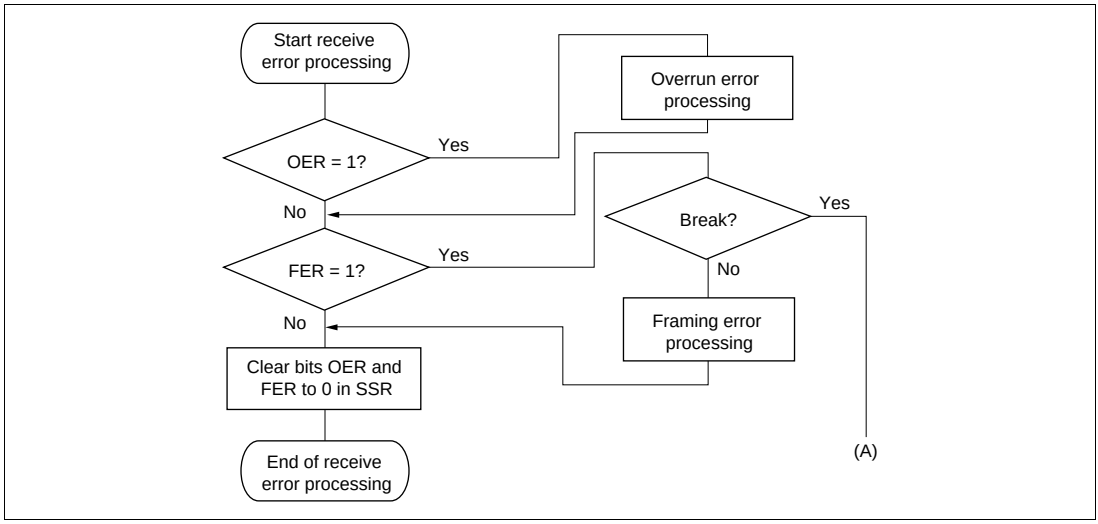
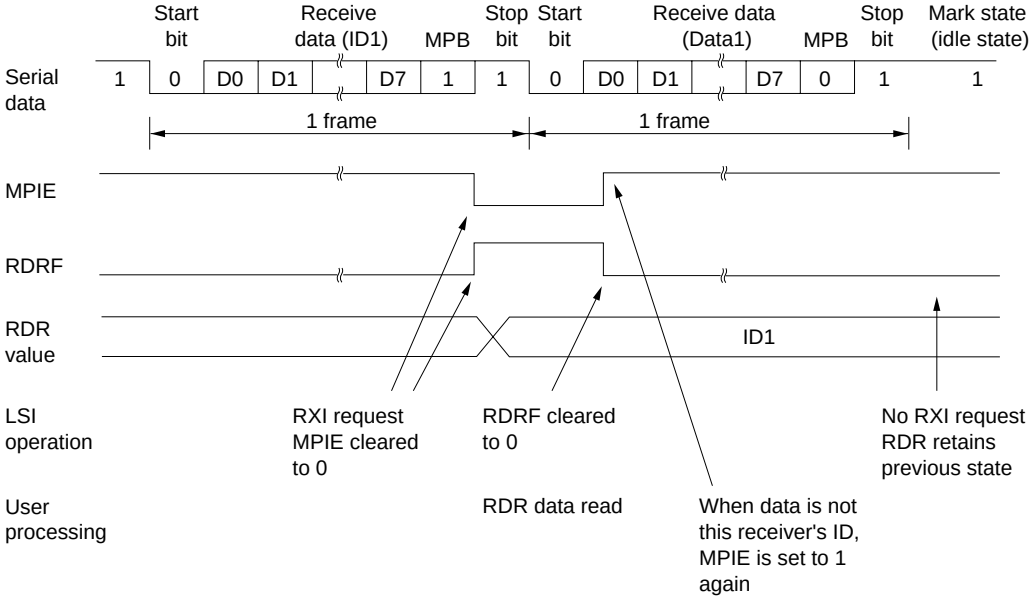
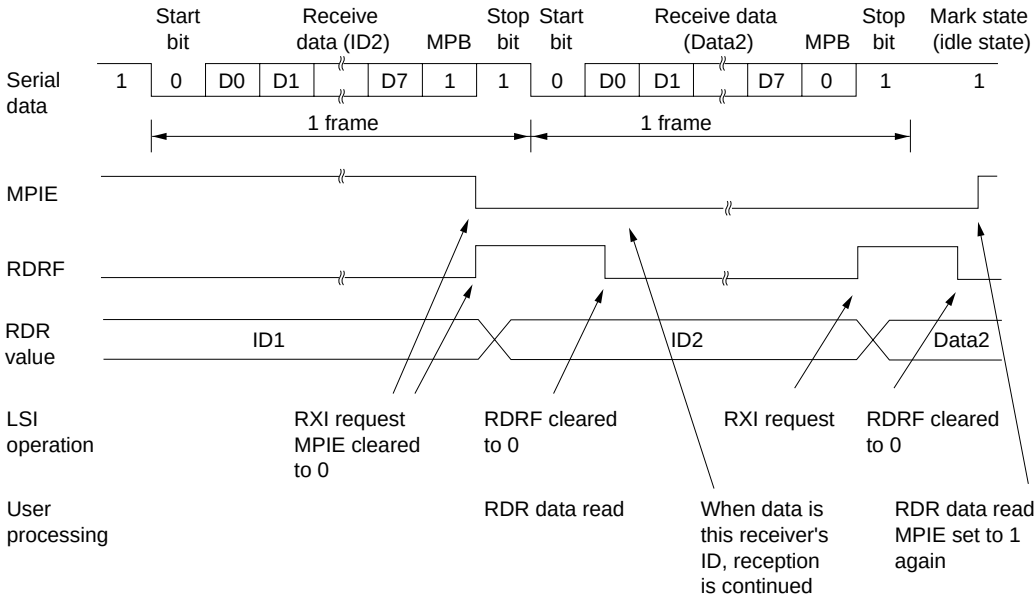


Figure 10.24 Example of Multiprocessor Data Reception Flowchart (cont)

Figure 10.25 shows an example of the operation when receiving using the multiprocessor format.



(a) When data does not match this receiver's ID



(b) When data matches this receiver's ID

Figure 10.25 Example of Operation when Receiving using Multiprocessor Format (8-Bit Data, Multiprocessor Bit, 1 Stop Bit)

10.3.7 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 10.16.

Table 10.16 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.

10.3.8 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 10.17. If an overrun error is detected, data transfer from RSR to RDR will not be performed, and the receive data will be lost.

Table 10.17 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	×	Overrun error
0	0	1	0	O	Framing error
0	0	0	1	O	Parity error
1	1	1	0	×	Overrun error + framing error
1	1	0	1	×	Overrun error + parity error
0	0	1	1	O	Framing error + parity error
1	1	1	1	×	Overrun error + framing error + parity error

O: Receive data is transferred from RSR to RDR.

×: Receive data is not transferred from RSR to RDR.

Note: * Bit RDRF retains its state prior to data reception.

3. Break detection and processing

   When a framing error is detected, a break can be detected by reading the value of the RXD pin directly. In a break, the input from the RXD 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 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 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 pin functions as an I/O port and 1 is output.

   To detect a break during transmission, 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 pin functions as an I/O port, and 0 is output from the TXD 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 10.26.

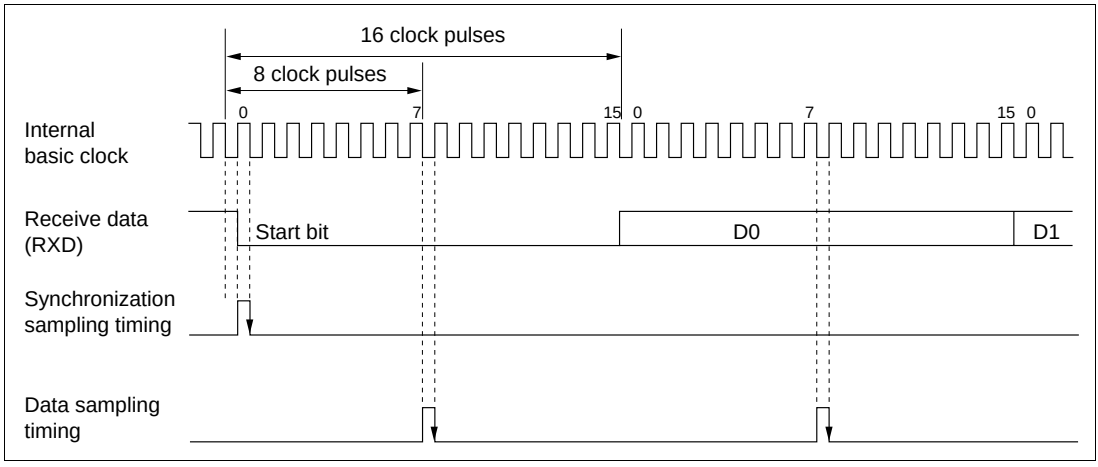


Figure 10.26 Receive Data Sampling Timing in Asynchronous Mode

Consequently, the receive margin in asynchronous mode can be expressed as shown in equation (1).

$$M = \left\{(0.5 - \frac{1}{2N}) - \frac{D - 0.5}{N} - (L - 0.5)\,F\right\} \times 100 \quad \text{. . . . . Equation (1)}$$

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,

$$M = \{0.5 - 1/(2 \times 16)\} \times 100\ [\%] = 46.875\% \quad \text{. . . . . Equation (2)}$$

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 10.27.

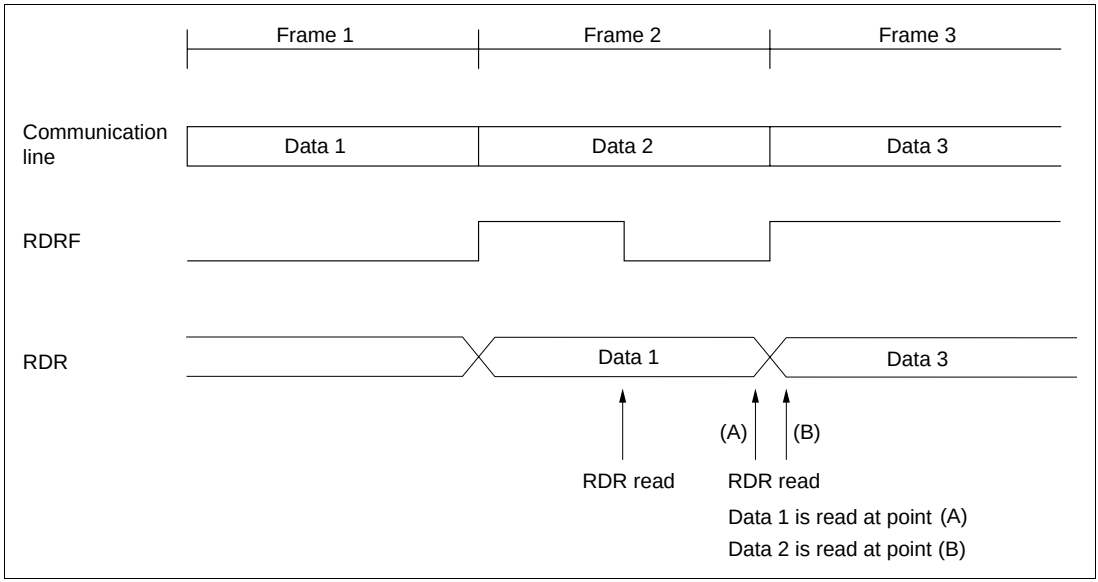


Figure 10.27 Relation between RDR Read Timing and Data

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.

Section 11 14-Bit PWM

11.1 Overview

The H8/3644 Series is provided with a 14-bit PWM (pulse width modulator) on-chip, which can be used as a D/A converter by connecting a low-pass filter.

11.1.1 Features

Features of the 14-bit PWM are as follows.

- Choice of two conversion periods
  A conversion period of 32,768/ø, with a minimum modulation width of 2/ø or a conversion period of 16,384/ø, with a minimum modulation width of 1/ø can be chosen.
- Pulse division method for less ripple

11.1.2 Block Diagram

Figure 11.1 shows a block diagram of the 14-bit PWM.

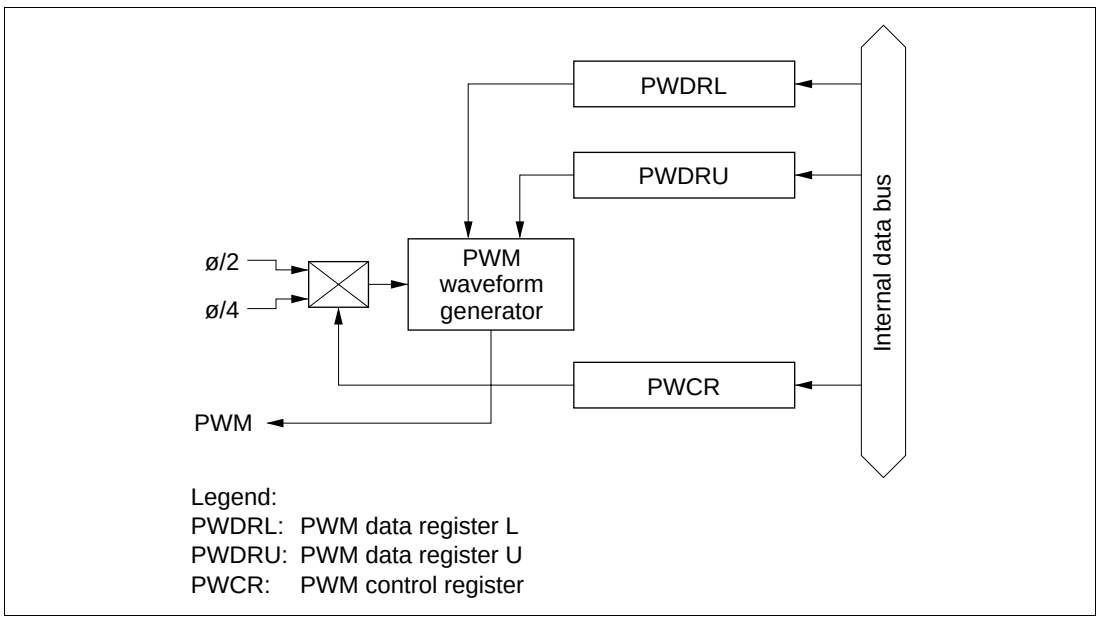


Figure 11.1 Block Diagram of the 14-Bit PWM

11.1.3 Pin Configuration

Table 11.1 shows the output pin assigned to the 14-bit PWM.

Table 11.1 Pin Configuration

Name	Abbrev.	I/O	Function
PWM output pin	PWM	Output	Pulse-division PWM waveform output

11.1.4 Register Configuration

Table 11.2 shows the register configuration of the 14-bit PWM.

Table 11.2 Register Configuration

Name	Abbrev.	R/W	Initial Value	Address
PWM control register	PWCR	W	H'FE	H'FFD0
PWM data register U	PWDRU	W	H'C0	H'FFD1
PWM data register L	PWDRL	W	H'00	H'FFD2

11.2 Register Descriptions

11.2.1 PWM Control Register (PWCR)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	PWCR0
Initial value	1	1	1	1	1	1	1	0
Read/Write	—	—	—	—	—	—	—	W

PWCR is an 8-bit write-only register for input clock selection.

Upon reset, PWCR is initialized to H'FE.

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

Bit 0—Clock Select 0 (PWCR0): Bit 0 selects the clock supplied to the 14-bit PWM. This bit is a write-only bit; it is always read as 1.

Bit 0: PWCR0	Description
0	The input clock is ø/2 ($t_ø$* = 2/ø). The conversion period is 16,384/ø, with a minimum modulation width of 1/ø (initial value)
1	The input clock is ø/4 ($t_ø$* = 4/ø). The conversion period is 32,768/ø, with a minimum modulation width of 2/ø.

Note: * $t_ø$: Period of PWM input clock

11.2.2 PWM Data Registers U and L (PWDRU, PWDRL)

PWDRU

Bit	7	6	5	4	3	2	1	0
	—	—	PWDRU5	PWDRU4	PWDRU3	PWDRU2	PWDRU1	PWDRU0
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	W	W	W	W	W	W

PWDRL

Bit	7	6	5	4	3	2	1	0
	PWDRL7	PWDRL6	PWDRL5	PWDRL4	PWDRL3	PWDRL2	PWDRL1	PWDRL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PWDRU and PWDRL form a 14-bit write-only register, with the upper 6 bits assigned to PWDRU and the lower 8 bits to PWDRL. The value written to PWDRU and PWDRL gives the total high-level width of one PWM waveform cycle.

When 14-bit data is written to PWDRU and PWDRL, the register contents are latched in the PWM waveform generator, updating the PWM waveform generation data. The 14-bit data should always be written in the following sequence:

1. Write the lower 8 bits to PWDRL.
2. Write the upper 6 bits to PWDRU.

PWDRU and PWDRL are write-only registers. If they are read, all bits are read as 1.

Upon reset, PWDRU and PWDRL are initialized to H'C000.

11.3 Operation

When using the 14-bit PWM, set the registers in the following sequence.

1. Set bit PWM in port mode register 1 (PMR1) to 1 so that pin $P1_4$/PWM is designated for PWM output.

2. Set bit PWCR0 in the PWM control register (PWCR) to select a conversion period of either 32,768/ø (PWCR0 = 1) or 16,384/ø (PWCR0 = 0).

3. Set the output waveform data in PWM data registers U and L (PWDRU/L). Be sure to write in the correct sequence, first PWDRL then PWDRU. When data is written to PWDRU, the data in these registers will be latched in the PWM waveform generator, updating the PWM waveform generation in synchronization with internal signals.

   One conversion period consists of 64 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 PWDRU and PWDRL. This relation can be represented as follows.

   $$T_H = (\text{data value in PWDRU and PWDRL} + 64) \times t_ø/2$$

   where $t_ø$ is the PWM input clock period, either 2/ø (bit PWCR0 = 0) or 4/ø (bit PWCR0 = 1).

   Example: Settings in order to obtain a conversion period of 8,192 µs:

   When bit PWCR0 = 0, the conversion period is 16,384/ø, so ø must be 2 MHz. In this case t_{fn} = 128 µs, with 1/ø (resolution) = 0.5 µs.
   When bit PWCR0 = 1, the conversion period is 32,768/ø, so ø must be 4 MHz. In this case t_{fn} = 128 µs, with 2/ø (resolution) = 0.5 µs.
   Accordingly, for a conversion period of 8,192 µs, the system clock frequency (ø) must be 2 MHz or 4 MHz.

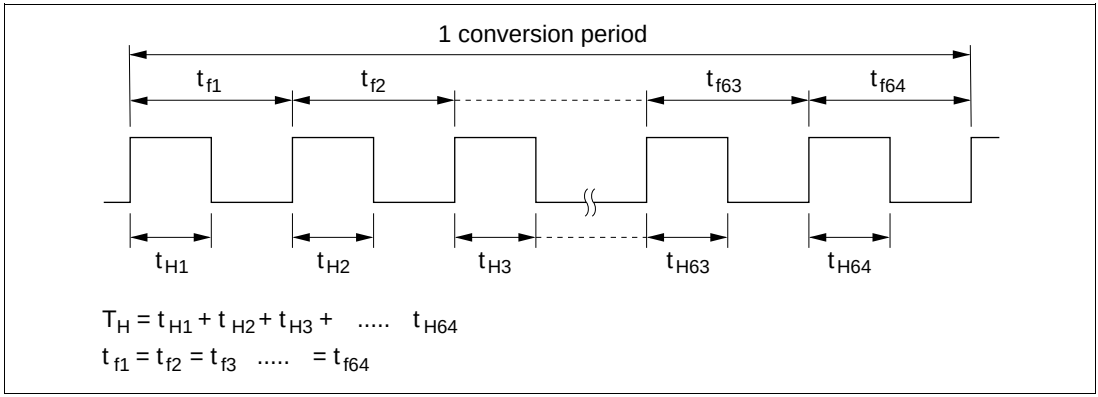


Figure 11.2 PWM Output Waveform

Section 12 A/D Converter

12.1 Overview

The H8/3644 Series includes on-chip a resistance-ladder-based successive-approximation analog-to-digital converter, and can convert up to 8 channels of analog input.

12.1.1 Features

The A/D converter has the following features.

- 8-bit resolution
- Eight input channels
- Conversion time: approx. 12.4 μs per channel (at 5 MHz operation)
- Built-in sample-and-hold function
- Interrupt requested on completion of A/D conversion
- A/D conversion can be started by external trigger input

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the A/D converter.

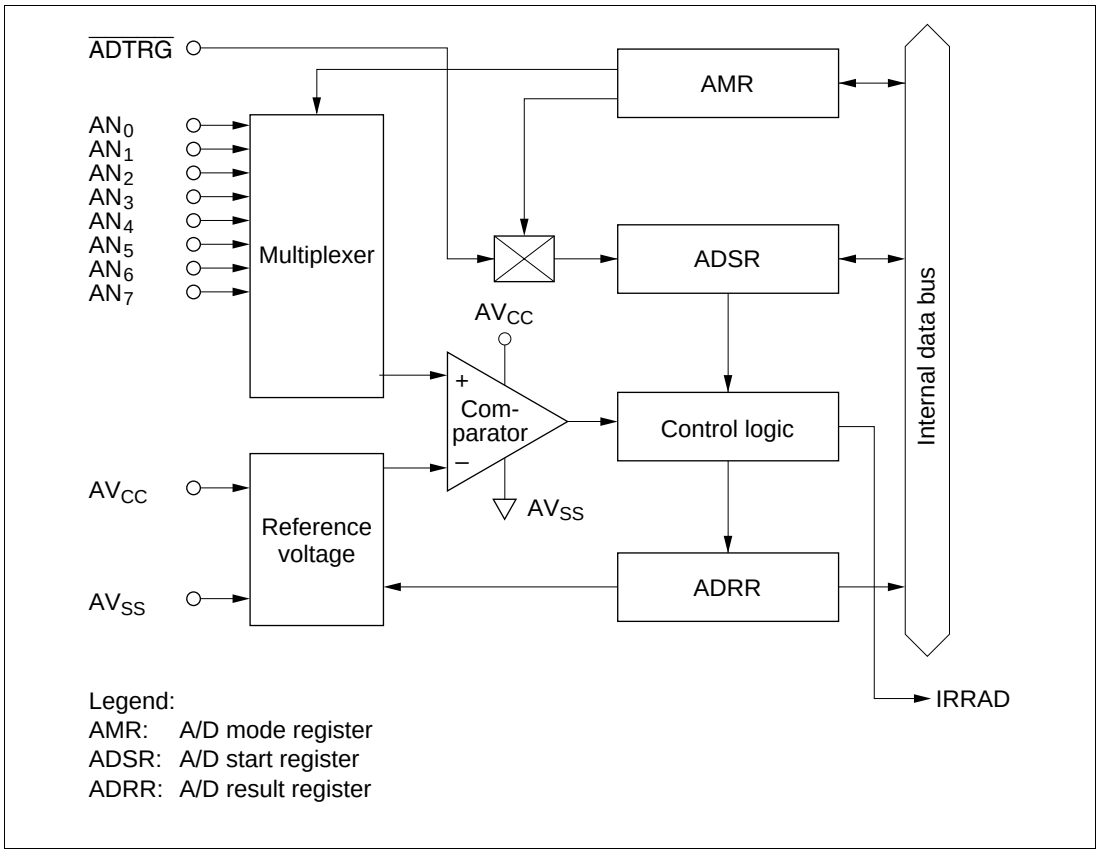


Figure 12.1 Block Diagram of the A/D Converter

12.1.3 Pin Configuration

Table 12.1 shows the A/D converter pin configuration.

Table 12.1 Pin Configuration

Name	Abbrev.	I/O	Function
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
Analog input 4	AN_4	Input	Analog input channel 4
Analog input 5	AN_5	Input	Analog input channel 5
Analog input 6	AN_6	Input	Analog input channel 6
Analog input 7	AN_7	Input	Analog input channel 7
External trigger input	$\overline{\text{ADTRG}}$	Input	External trigger input for starting A/D conversion

12.1.4 Register Configuration

Table 12.2 shows the A/D converter register configuration.

Table 12.2 Register Configuration

Name	Abbrev.	R/W	Initial Value	Address
A/D mode register	AMR	R/W	H'30	H'FFC4
A/D start register	ADSR	R/W	H'7F	H'FFC6
A/D result register	ADRR	R	Not fixed	H'FFC5

12.2 Register Descriptions

12.2.1 A/D Result Register (ADRR)

Bit	7	6	5	4	3	2	1	0
	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
Initial value	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

The A/D result register (ADRR) is an 8-bit read-only register for holding the results of analog-to-digital conversion.

ADRR can be read by the CPU at any time, but the ADRR values during A/D conversion are not fixed.

After A/D conversion is complete, the conversion result is stored in ADRR as 8-bit data; this data is held in ADRR until the next conversion operation starts.

ADRR is not cleared on reset.

12.2.2 A/D Mode Register (AMR)

Bit	7	6	5	4	3	2	1	0
	CKS	TRGE	—	—	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.

		Conversion Time		
Bit 7: CKS	**Conversion Period**	**ø = 2 MHz**	**ø = 5 MHz**	**ø = 8 MHz*[1]**
0	62/ø (initial value)	31 µs	12.4 µs	7.75 µs
1	31/ø	15.5 µs	—*[2]	—

Notes: 1. F-ZTAT version only.
2. 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—External Trigger Select (TRGE): Bit 6 enables or disables the start of A/D conversion by external trigger input.

Bit 6: TRGE	Description	
0	Disables start of A/D conversion by external trigger	(initial value)
1	Enables start of A/D conversion by rising or falling edge of external trigger at pin $\overline{\text{ADTRG}}$*	

Note: * The external trigger ($\overline{\text{ADTRG}}$) edge is selected by bit INTEG5 of IEGR2. See 3.3.2 Interrupt Edge Select Register 2 (IEGR2) for details.

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)
	1	0	0	AN_0	
			1	AN_1	
		1	0	AN_2	
			1	AN_3	
1	0	0	0	AN_4	
			1	AN_5	
		1	0	AN_6	
			1	AN_7	
	1	0	0	Reserved	
			1	Reserved	
		1	0	Reserved	
			1	Reserved	

Note: * Don't care

12.2.3 A/D Start Register (ADSR)

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

The A/D start register (ADSR) is an 8-bit read/write register for starting and stopping A/D conversion.

A/D conversion is started by writing 1 to the A/D start flag (ADSF) or by input of the designated edge of the external trigger signal, which also sets ADSF to 1. When conversion is complete, the converted data is set in the A/D result register (ADRR), and at the same time ADSF is cleared to 0.

Bit 7—A/D Start Flag (ADSF): Bit 7 controls and indicates the start and end of A/D conversion.

Bit 7: ADSF	Description	
0	Read: 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.

12.3 Operation

12.3.1 A/D Conversion Operation

The A/D converter operates by successive approximations, and yields its conversion result as 8-bit data.

A/D conversion begins when software sets the A/D start flag (bit ADSF) to 1. Bit ADSF keeps a value of 1 during A/D conversion, and is cleared to 0 automatically when conversion is complete.

The completion of conversion also sets bit IRRAD in interrupt request register 2 (IRR2) to 1. An A/D conversion end interrupt is requested if bit IENAD in interrupt enable register 2 (IENR2) is set to 1.

If the conversion time or input channel needs to be changed in the A/D mode register (AMR) during A/D conversion, bit ADSF should first be cleared to 0, stopping the conversion operation, in order to avoid malfunction.

12.3.2 Start of A/D Conversion by External Trigger Input

The A/D converter can be made to start A/D conversion by input of an external trigger signal. External trigger input is enabled at pin $\overline{\text{ADTRG}}$ when bit TRGE in AMR is set to 1. Then when the input signal edge designated in bit INTEG5 of interrupt edge select register 2 (IEGR2) is detected at pin $\overline{\text{ADTRG}}$, bit ADSF in ADSR will be set to 1, starting A/D conversion.

Figure 12.2 shows the timing.

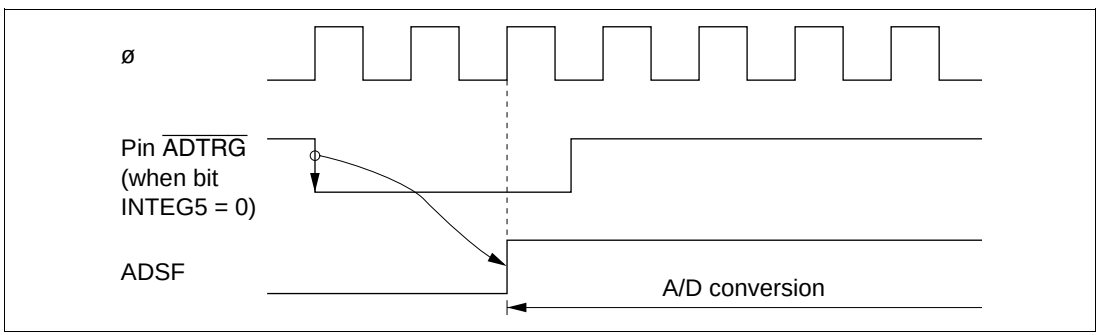


Figure 12.2 External Trigger Input Timing

12.4 Interrupts

When A/D conversion ends (ADSF changes from 1 to 0), bit IRRAD in interrupt request register 2 (IRR2) is set to 1.

A/D conversion end interrupts can be enabled or disabled by means of bit IENAD in interrupt enable register 2 (IENR2).

For further details see 3.3, Interrupts.

12.5 Typical Use

An example of how the A/D converter can be used is given below, using channel 1 (pin AN1) as the analog input channel. Figure 12.3 shows the operation timing.

1. Bits CH3 to CH0 of the A/D mode register (AMR) are set to 0101, making pin AN1 the analog input channel. A/D interrupts are enabled by setting bit IENAD to 1, and A/D conversion is started by setting bit ADSF to 1.
2. When A/D conversion is complete, bit IRRAD is set to 1, and the A/D conversion result is stored in the A/D result register (ADRR). At the same time ADSF is cleared to 0, and the A/D converter goes to the idle state.
3. Bit IENAD = 1, so an A/D conversion end interrupt is requested.
4. The A/D interrupt handling routine starts.
5. The A/D conversion result is read and processed.
6. The A/D interrupt handling routine ends.

If ADSF is set to 1 again afterward, A/D conversion starts and steps 2 through 6 take place.

Figures 12.4 and 12.5 show flow charts of procedures for using the A/D converter.

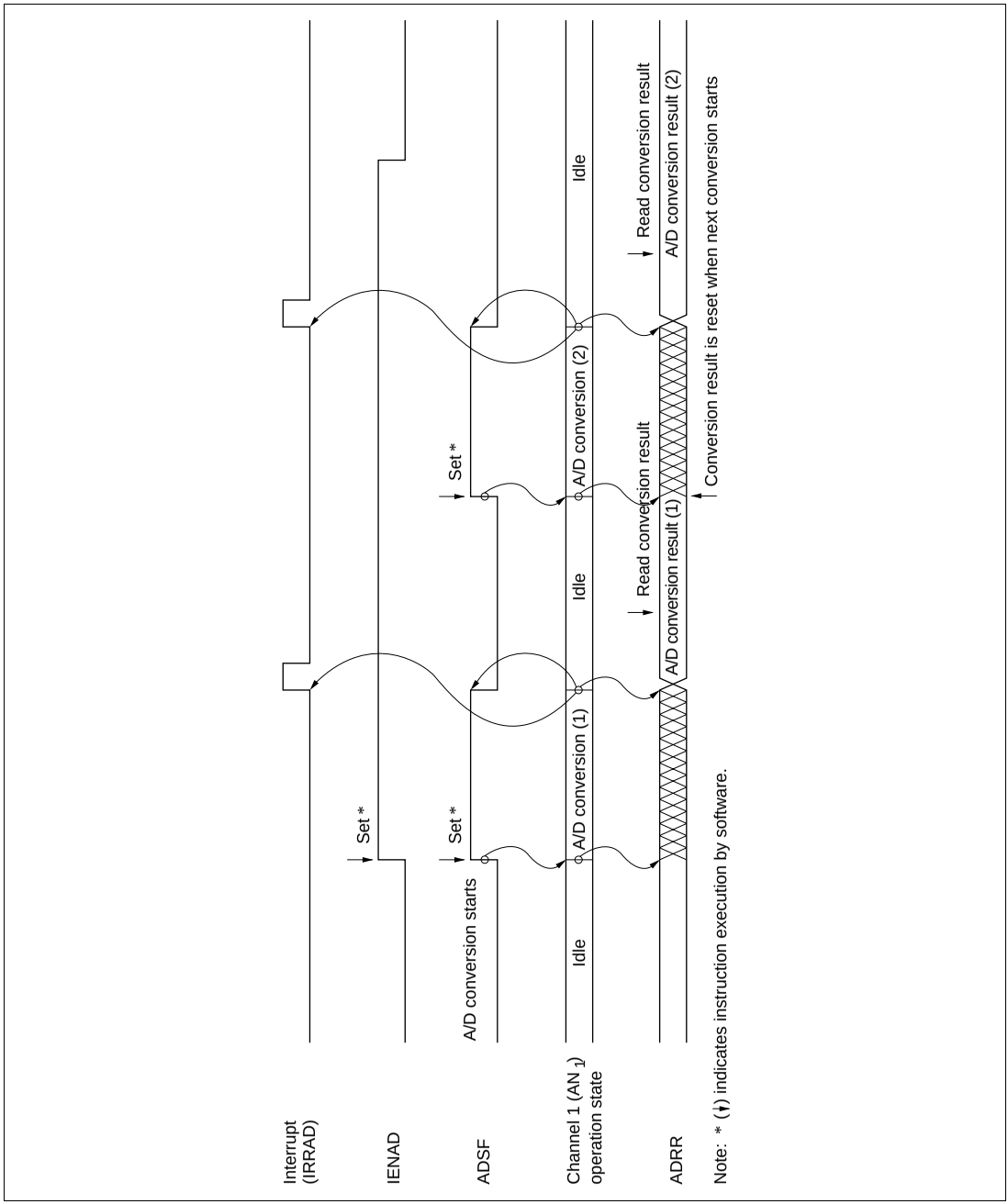


Figure 12.3 Typical A/D Converter Operation Timing

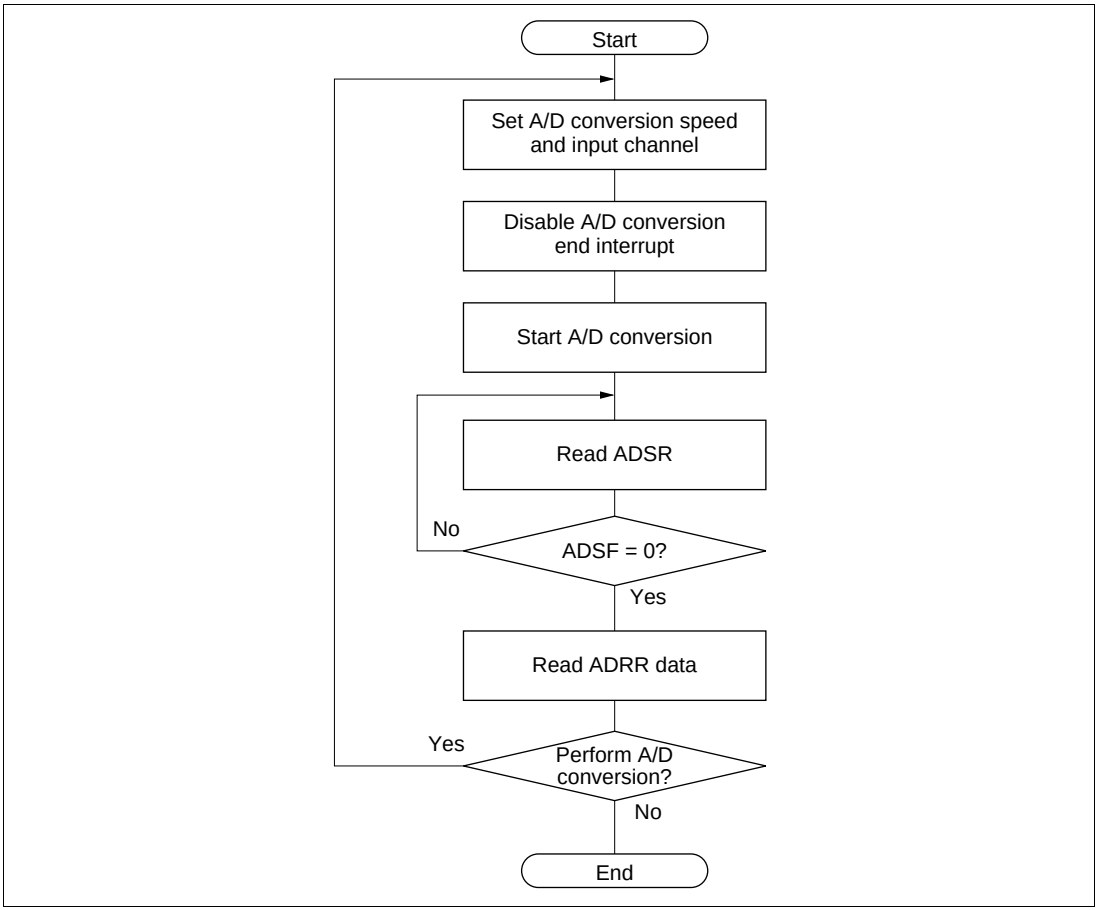


Figure 12.4 Flow Chart of Procedure for Using A/D Converter (1) (Polling by Software)

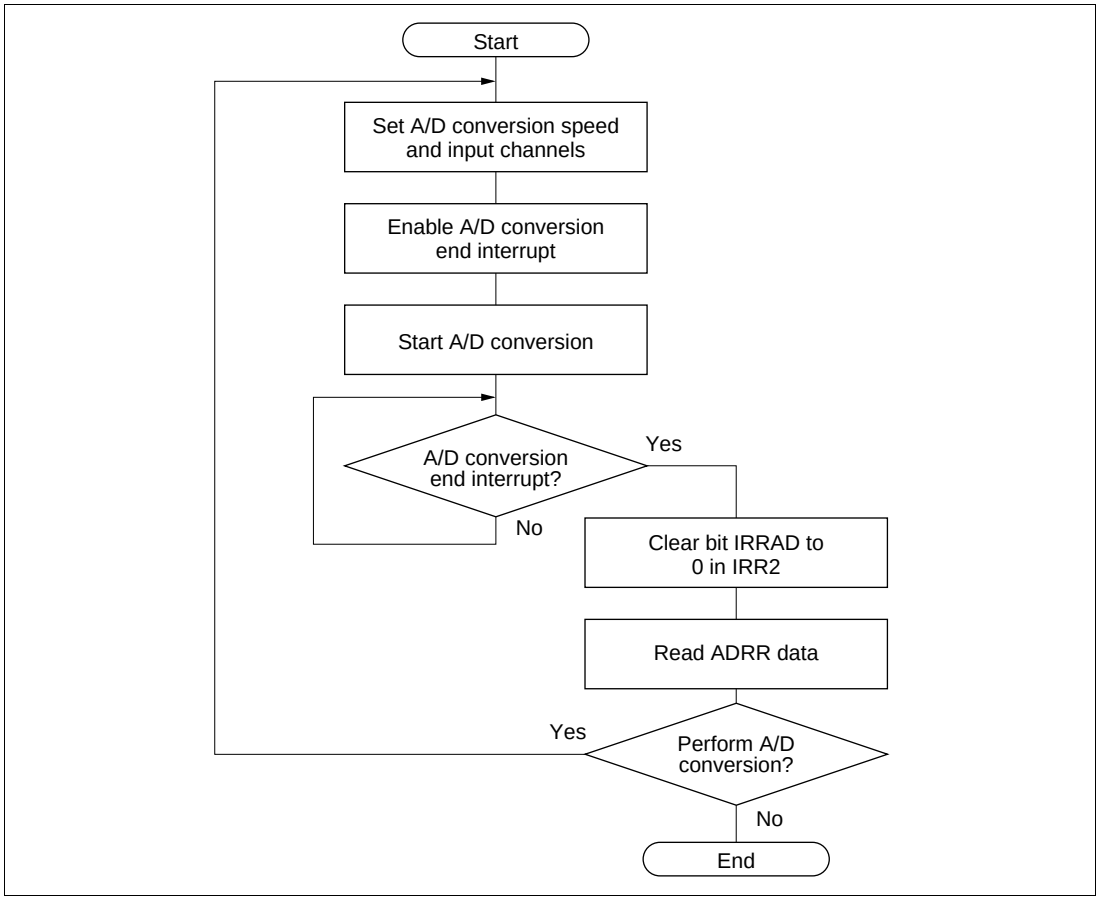


Figure 12.5 Flow Chart of Procedure for Using A/D Converter (2) (Interrupts Used)

12.6 Application Notes

- Data in the A/D result register (ADRR) should be read only when the A/D start flag (ADSF) in the A/D start register (ADSR) is cleared to 0.
- Changing the digital input signal at an adjacent pin during A/D conversion may adversely affect conversion accuracy.

Section 13 Electrical Characteristics

13.1 Absolute Maximum Ratings

Table 13.1 lists the absolute maximum ratings.

Table 13.1 Absolute Maximum Ratings*[1]

Item		Symbol	Value	Unit	Note
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	HD6473644	V_{PP}	–0.3 to +13.0	V	
	HD64F3644, HD64F3643, HD64F3642A	FV_{PP}	–0.3 to +13.0	V	2
Input voltage	Ports other than Port B	V_{in}	–0.3 to V_{CC} +0.3	V	
	Port B		–0.3 to AV_{CC} +0.3	V	
	TEST (HD64F3644, HD64F3643,HD64F3642A)		–0.3 to +13.0	V	2
Operating temperature		T_{opr}	–20 to +75	°C	
Storage temperature		T_{stg}	–55 to +125	°C	

Notes: 1. Permanent damage may occur to the chip if maximum ratings are exceeded. Normal operation should be under the conditions specified in Electrical Characteristics. Exceeding these values can result in incorrect operation and reduced reliability.

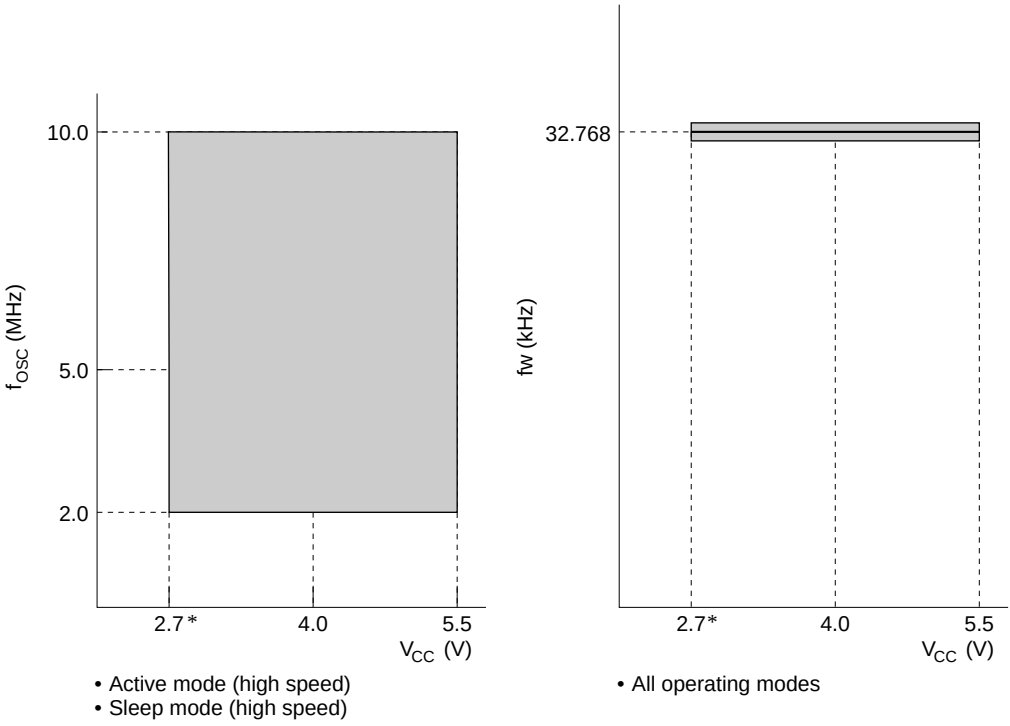
2. The voltage at the FV_{PP} and TEST pins should not exceed 13 V, including peak overshoot.

13.2 Electrical Characteristics (ZTAT™, Mask ROM Version)

13.2.1 Power Supply Voltage and Operating Range

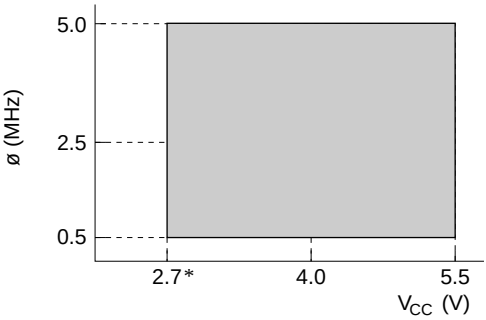
The power supply voltage and operating range are indicated by the shaded region in the figures below.

1. Power supply voltage vs. oscillator frequency range

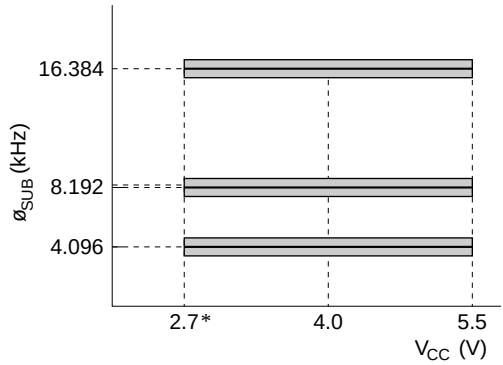


Note: * 2.5 V for the HD6433644, HD6433643, HD6433642, HD6433641 and HD6433640.

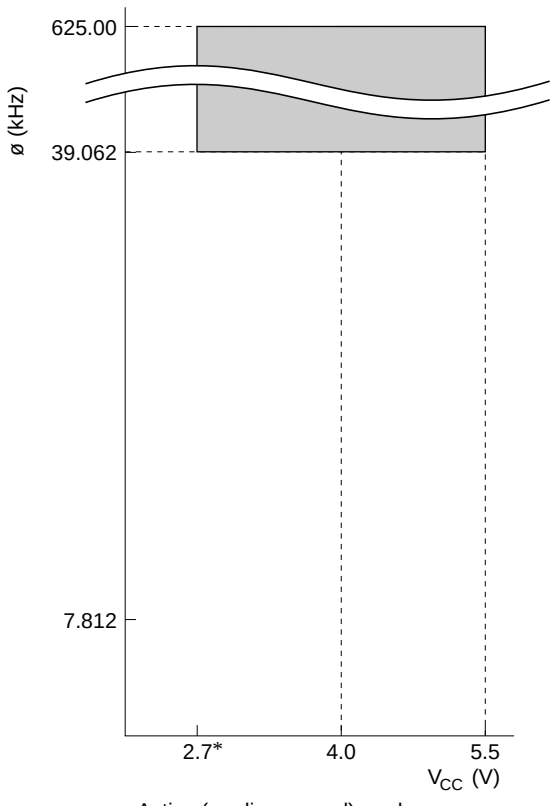
2. Power supply voltage vs. clock frequency range



- Active (high speed) mode
- Sleep (high speed) mode (except CPU)



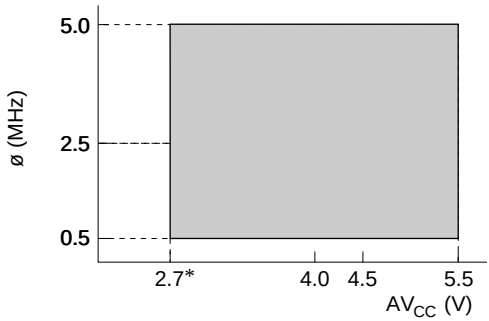
- Subactive mode
- Subsleep mode (except CPU)
- Watch mode (except CPU)



- Active (medium speed) mode
- Sleep (medium speed) mode (except CPU)

Note: * 2.5 V for the HD6433644, HD6433643, HD6433642, HD6433641 and HD6433640.

3. Analog power supply voltage vs. A/D converter guaranteed accuracy range



Do not exceed the maximum conversion time value.

- Active (high speed) mode
- Sleep (high speed) mode

- Active (medium speed) mode
- Sleep (medium speed) mode

Note: * The voltage for guaranteed A/D conversion operation is 2.5 (V).

13.2.2 DC Characteristics (HD6473644)

Table 13.2 lists the DC characteristics of the HD6473644.

Table 13.2 DC Characteristics

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{INT}_0$ to $\overline{INT}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_3$, $\overline{ADTRG}$, TMIB, TMRIV, TMCIV, FTCI, FTIA, FTIB, FTIC, FTID, SCK_1, SCK_3, TRGV	0.8 V_{CC}	—	V_{CC} + 0.3	V		
			0.9 V_{CC}	—	V_{CC} + 0.3		V_{CC} = 2.7 V to 5.5 V including subactive mode	
		SI_1, RXD, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	0.7 V_{CC}	—	V_{CC} + 0.3	V		
			0.8 V_{CC}	—	V_{CC} + 0.3		V_{CC} = 2.7 V to 5.5 V including subactive mode	
		PB_0 to PB_7	0.7 V_{CC}	—	AV_{CC} + 0.3	V		
			0.8 V_{CC}	—	AV_{CC} + 0.3		V_{CC} = 2.7 V to 5.5 V including subactive mode	
		OSC_1	V_{CC} – 0.5	—	V_{CC} + 0.3	V		
			V_{CC} – 0.3	—	V_{CC} + 0.3		V_{CC} = 2.7 V to 5.5 V including subactive mode	

Note: Connect the TEST pin to V_{SS}.

Table 13.2 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input low voltage	V_{IL}	$\overline{RES}$, $\overline{INT}_0$ to $\overline{INT}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_3$, $\overline{ADTRG}$, TMIB, TMRIV, TMCIV, FTCI, FTIA, FTIB, FTIC, FTID, SCK_1, SCK_3, TRGV	–0.3	—	0.2 V_{CC}	V		
			–0.3	—	0.1 V_{CC}		V_{CC} = 2.7 V to 5.5 V including subactive mode	
		SI_1, RXD, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$, PB_0 to PB_7	–0.3	—	0.3 V_{CC}	V		
			–0.3	—	0.2 V_{CC}		V_{CC} = 2.7 V to 5.5 V including subactive mode	
		OSC_1	–0.3	—	0.5	V		
			–0.3	—	0.3		V_{CC} = 2.7 V to 5.5 V including subactive mode	

Table 13.2 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Output high voltage	V_{OH}	$P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$,	V_{CC} – 1.0	—	—	V	$-I_{OH}$ = 1.5 mA	
		$P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	V_{CC} – 0.5	—	—		V_{CC} = 2.7 V to 5.5 V $-I_{OH}$ = 0.1 mA	
Output low voltage	V_{OL}	$P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$,	—	—	0.6	V	I_{OL} = 1.6 mA	
		$P5_0$ to $P5_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	—	—	0.4		V_{CC} = 2.7 V to 5.5 V I_{OL} = 0.4 mA	
		$P6_0$ to $P6_7$	—	—	1.0	V	I_{OL} = 10.0 mA	
			—	—	0.4		I_{OL} = 1.6 mA	
			—	—	0.4		V_{CC} = 2.7 V to 5.5 V I_{OL} = 0.4 mA	
Input/ output leakage current	$\lvert I_{IL} \rvert$	OSC_1, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	—	—	1.0	µA	V_{in} = 0.5 V to (V_{CC} – 0.5 V)	
		PB_0 to PB_7	—	—	1.0	µA	V_{in} = 0.5 V to (AV_{CC} – 0.5 V)	
Input leakage current	$\lvert I_{IL} \rvert$	$\overline{RES}$, $\overline{IRQ_0}$	—	—	20	µA	V_{in} = 0.5 V to (V_{CC} – 0.5 V)	
Pull-up MOS current	$-I_p$	$P1_0$, $P1_4$ to $P1_7$, $P3_0$ to $P3_2$,	50	—	300	µA	V_{CC} = 5 V, V_{in} = 0 V	
		$P5_0$ to $P5_7$	—	25	—		V_{CC} = 2.7 V, V_{in} = 0 V	Reference value

Table 13.2 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input capacitance	C_{in}	All input pins except $\overline{RES}$	—	—	15.0	pF	f = 1 MHz, V_{in} = 0 V, T_a = 25°C	
		$\overline{RES}$	—	—	60.0			
		$\overline{IRQ_0}$	—	—	30.0			
Active mode current dissipation	I_{OPE1}	V_{CC}	—	10	15	mA	Active (high-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	5	—		V_{CC} = 2.7 V, f_{OSC} = 10 MHz	1, 2 Reference value
	I_{OPE2}	V_{CC}	—	2	3	mA	Active (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	1	—		V_{CC} = 2.7 V, f_{OSC} = 10 MHz	1, 2 Reference value
Sleep mode current dissipation	I_{SLEEP1}	V_{CC}	—	5	7	mA	Sleep (high-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	2	—		V_{CC} = 2.7 V, f_{OSC} = 10 MHz	1, 2 Reference value
	I_{SLEEP2}	V_{CC}	—	2	3	mA	Sleep (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	1	—		V_{CC} = 2.7 V, f_{OSC} = 10 MHz	1, 2 Reference value
Subactive mode current dissipation	I_{SUB}	V_{CC}	—	10	20	µA	V_{CC} = 2.7 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/2$)	1, 2
			—	10	—		V_{CC} = 2.7 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/8$)	1, 2 Reference value

Table 13.2 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Subsleep mode current dissipation	I_{SUBSP}	V_{CC}	—	5	10	µA	V_{CC} = 2.7 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/2$)	1, 2
Watch mode current dissipation	I_{WATCH}	V_{CC}	—	—	6	µA	V_{CC} = 2.7 V 32-kHz crystal oscillator	1, 2
Standby mode current dissipation	I_{STBY}	V_{CC}	—	—	5	µA	32-kHz crystal oscillator not used	1, 2
RAM data retaining voltage	V_{RAM}	V_{CC}	2	—	—	V		

Notes: 1. Pin states during current measurement are given below.

Mode	$\overline{RES}$ Pin	Internal State	Other Pins	Oscillator Pins
Active (high-speed) mode	V_{CC}	Operates	V_{CC}	System clock oscillator: ceramic or crystal
Active (medium-speed) mode		Operates ($ø_{OSC}$/128)		Subclock oscillator: Pin $X_1 = V_{CC}$
Sleep (high-speed) mode	V_{CC}	Only timers operate	V_{CC}	
Sleep (medium-speed) mode		Only timers operate ($ø_{OSC}$/128)		
Subactive mode	V_{CC}	Operates	V_{CC}	System clock oscillator: ceramic or crystal
Subsleep mode	V_{CC}	Only timers operate, CPU stops	V_{CC}	Subclock oscillator: crystal
Watch mode	V_{CC}	Only time base operates, CPU stops	V_{CC}	
Standby mode	V_{CC}	CPU and timers both stop	V_{CC}	System clock oscillator: ceramic or crystal Subclock oscillator: Pin $X_1 = V_{CC}$

2. Excludes current in pull-up MOS transistors and output buffers.

Table 13.2 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise indicated.

Item		Symbol	Values			Unit
			Min	Typ	Max	
Allowable output low current (per pin)	Output pins except port 6	I_{OL}	—	—	2	mA
	Port 6		—	—	10	
Allowable output low current (total)	Output pins except port 6	ΣI_{OL}	—	—	40	mA
	Port 6		—	—	80	
Allowable output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	mA
Allowable output high current (total)	All output pins	$\Sigma(-I_{OH})$	—	—	30	mA

13.2.3 AC Characteristics (HD6473644)

Table 13.3 lists the control signal timing, and tables 13.4 and 13.5 list the serial interface timing of the HD6473644.

Table 13.3 Control Signal Timing

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
System clock oscillation frequency	f_{OSC}	OSC_1, OSC_2	2	—	10	MHz	V_{CC} = 2.7 V to 5.5 V	
OSC clock ($ø_{OSC}$) cycle time	t_{OSC}	OSC_1, OSC_2	100	—	1000	ns	V_{CC} = 2.7 V to 5.5 V	*1 Figure 13.1
System clock (ø) cycle time	t_{cyc}		2	—	128	t_{OSC}	V_{CC} = 2.7 V to 5.5 V	*1
			—	—	25.6	µs		
Subclock oscillation frequency	f_W	X_1, X_2	—	32.768	—	kHz	V_{CC} = 2.7 V to 5.5 V	
Watch clock ($ø_W$) cycle time	t_W	X_1, X_2	—	30.5	—	µs	V_{CC} = 2.7 V to 5.5 V	
Subclock ($ø_{SUB}$) cycle time	t_{subcyc}		2	—	8	t_W	V_{CC} = 2.7 V to 5.5 V	*2
Instruction cycle time			2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 2.7 V to 5.5 V	
Oscillation stabilization time (crystal oscillator)	t_{rc}	OSC_1, OSC_2	—	—	40	ms		
			—	—	60		V_{CC} = 2.7 V to 5.5 V	
Oscillation stabilization time (ceramic oscillator)	t_{rc}	OSC_1, OSC_2	—	—	20	ms		
			—	—	40		V_{CC} = 2.7 V to 5.5 V	
Oscillation stabilization time	t_{rc}	X_1, X_2	—	—	2	s	V_{CC} = 2.7 V to 5.5 V	
External clock high width	t_{CPH}	OSC_1	40	—	—	ns	V_{CC} = 2.7 V to 5.5 V	Figure 13.1
External clock low width	t_{CPL}	OSC_1	40	—	—	ns	V_{CC} = 2.7 V to 5.5 V	
External clock rise time	t_{CPr}		—	—	15	ns	V_{CC} = 2.7 V to 5.5 V	
External clock fall time	t_{CPf}		—	—	15	ns	V_{CC} = 2.7 V to 5.5 V	
Pin $\overline{RES}$ low width	t_{REL}	$\overline{RES}$	10	—	—	t_{cyc}	V_{CC} = 2.7 V to 5.5 V	Figure 13.2

Notes: 1. A frequency between 1 MHz to 10 MHz is required when an external clock is input.
2. Selected with SA1 and SA0 of system clock control register 2 (SYSCR2).

Table 13.3 Control Signal Timing (cont)

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input pin high width	t_{IH}	$\overline{IRQ_0}$ to $\overline{IRQ_3}$, $\overline{INT_0}$ to $\overline{INT_7}$, $\overline{ADTRG}$, TMIB, TMCIV, TMRIV, FTCI, FTIA, FTIB, FTIC, FTID, TRGV	2	—	—	t_{cyc} t_{subcyc}		Figure 13.3
Input pin low width	t_{IL}	$\overline{IRQ_0}$ to $\overline{IRQ_3}$, $\overline{INT_6}$, $\overline{INT_7}$, $\overline{ADTRG}$, TMIB, TMCIV, TMRIV, FTCI, FTIA, FTIB, FTIC, FTID, TRGV	2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 2.7 V to 5.5 V	

Table 13.4 Serial Interface (SCI1) Timing

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input serial clock cycle time	t_{Scyc}	SCK_1	2	—	—	t_{cyc}	V_{CC} = 2.7 V to 5.5 V	Figure 13.4
Input serial clock high width	t_{SCKH}	SCK_1	0.4	—	—	t_{Scyc}	V_{CC} = 2.7 V to 5.5 V	
Input serial clock low width	t_{SCKL}	SCK_1	0.4	—	—	t_{Scyc}	V_{CC} = 2.7 V to 5.5 V	
Input serial clock rise time	t_{SCKr}	SCK_1	—	—	60	ns		
			—	—	80		V_{CC} = 2.7 V to 5.5 V	
Input serial clock fall time	t_{SCKf}	SCK_1	—	—	60	ns		
			—	—	80		V_{CC} = 2.7 V to 5.5 V	
Serial output data delay time	t_{SOD}	SO_1	—	—	200	ns		
			—	—	350		V_{CC} = 2.7 V to 5.5 V	
Serial input data setup time	t_{SIS}	SI_1	180	—	—	ns		
			360	—	—		V_{CC} = 2.7 V to 5.5 V	
Serial input data hold time	t_{SIH}	SI_1	180	—	—	ns		
			360	—	—		V_{CC} = 2.7 V to 5.5 V	

Table 13.5 Serial Interface (SCI3) Timing

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item		Symbol	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input clock cycle	Asynchronous	t_{Scyc}	4	—	—	t_{cyc}		Figure 13.5
	Synchronous		6	—	—			
Input clock pulse width		t_{SCKW}	0.4	—	0.6	t_{Scyc}		
Transmit data delay time (synchronous)		t_{TXD}	—	—	1	t_{cyc}	V_{CC} = 4.0 V to 5.5 V	Figure 13.6
			—	—	1			
Receive data setup time (synchronous)		t_{RXS}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	
			400.0	—	—			
Receive data hold time (synchronous)		t_{RXH}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	
			400.0	—	—			

13.2.4 DC Characteristics (HD6433644, HD6433643, HD6433642, HD6433641, HD6433640)

Table 13.6 lists the DC characteristics of the HD6433644, the HD6433643, the HD6433642, the HD6433641 and the HD6433640.

Table 13.6 DC Characteristics

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C unless otherwise indicated.

			Values					
Item	**Symbol**	**Applicable Pins**	**Min**	**Typ**	**Max**	**Unit**	**Test Condition**	**Notes**
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{INT}_0$ to $\overline{INT}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_3$, $\overline{ADTRG}$, TMIB, TMRIV, TMCIV, FTCI, FTIA, FTIB, FTIC, FTID, SCK_1, SCK_3, TRGV	0.8 V_{CC}	—	V_{CC} + 0.3	V		
			0.9 V_{CC}	—	V_{CC} + 0.3		V_{CC} = 2.5 V to 5.5 V including subactive mode	
		SI_1, RXD, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	0.7 V_{CC}	—	V_{CC} + 0.3	V		
			0.8 V_{CC}	—	V_{CC} + 0.3		V_{CC} = 2.5 V to 5.5 V including subactive mode	
		PB_0 to PB_7	0.7 V_{CC}	—	AV_{CC} + 0.3	V		
			0.8 V_{CC}	—	AV_{CC} + 0.3		V_{CC} = 2.5 V to 5.5 V including subactive mode	
		OSC_1	V_{CC} – 0.5	—	V_{CC} + 0.3	V		
			V_{CC} – 0.3	—	V_{CC} + 0.3		V_{CC} = 2.5 V to 5.5 V including subactive mode	

Note: Connect the TEST pin to V_{SS}.

Table 13.6 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input low voltage	V_{IL}	$\overline{RES}$, $\overline{INT}_0$ to $\overline{INT}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_3$, $\overline{ADTRG}$, TMIB,	–0.3	—	0.2 V_{CC}	V		
			–0.3	—	0.1 V_{CC}		V_{CC} = 2.5 V to 5.5 V including subactive mode	
		TMRIV, TMCIV, FTCI, FTIA, FTIB, FTIC, FTID, SCK_1, SCK_3, TRGV						
		SI_1, RXD, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$,	–0.3	—	0.3 V_{CC}	V		
		$P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$, PB_0 to PB_7	–0.3	—	0.2 V_{CC}		V_{CC} = 2.5 V to 5.5 V including subactive mode	
		OSC_1	–0.3	—	0.5	V		
			–0.3	—	0.3		V_{CC} = 2.5 V to 5.5 V including subactive mode	

Table 13.6 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Output high voltage	V_{OH}	$P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$,	V_{CC} – 1.0	—	—	V	$–I_{OH}$ = 1.5 mA	
		$P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	V_{CC} – 0.5	—	—		V_{CC} = 2.5 V to 5.5 V $–I_{OH}$ = 0.1 mA	
Output low voltage	V_{OL}	$P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$,	—	—	0.6	V	I_{OL} = 1.6 mA	
		$P5_0$ to $P5_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	—	—	0.4		V_{CC} = 2.5 V to 5.5 V I_{OL} = 0.4 mA	
		$P6_0$ to $P6_7$	—	—	1.0	V	I_{OL} = 10.0 mA	
			—	—	0.4		I_{OL} = 1.6 mA	
			—	—	0.4		V_{CC} = 2.5 V to 5.5 V I_{OL} = 0.4 mA	
Input/ output leakage current	$\lvert I_{IL} \rvert$	OSC_1, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_4$	—	—	1.0	µA	V_{in} = 0.5 V to (V_{CC} – 0.5 V)	
		PB_0 to PB_7	—	—	1.0	µA	V_{in} = 0.5 V to (AV_{CC} – 0.5 V)	
Input leakage current	$\lvert I_{IL} \rvert$	$\overline{RES}$, $\overline{IRQ_0}$	—	—	1	µA	V_{in} = 0.5 V to (V_{CC} – 0.5 V)	
Pull-up MOS current	$–I_p$	$P1_0$, $P1_4$ to $P1_7$, $P3_0$ to $P3_2$,	50	—	300	µA	V_{CC} = 5 V, V_{in} = 0 V	
		$P5_0$ to $P5_7$	—	25	—		V_{CC} = 2.7 V, V_{in} = 0 V	Reference value

Table 13.6 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input capacitance	C_{in}	All input pins except $\overline{RES}$	—	—	15.0	pF	f = 1 MHz, V_{in} = 0 V, T_a = 25°C	
		$\overline{RES}$	—	—	15.0			
		$\overline{IRQ_0}$	—	—	15.0			
Active mode current dissipation	I_{OPE1}	V_{CC}	—	10	15	mA	Active (high-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	5	—		V_{CC} = 2.5 V, f_{OSC} = 10 MHz	1, 2 Reference value
	I_{OPE2}	V_{CC}	—	2	3	mA	Active (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	1	—		V_{CC} = 2.5 V, f_{OSC} = 10 MHz	1, 2 Reference value
Sleep mode current dissipation	I_{SLEEP1}	V_{CC}	—	5	7	mA	Sleep (high-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	2	—		V_{CC} = 2.5 V, f_{OSC} = 10 MHz	1, 2 Reference value
	I_{SLEEP2}	V_{CC}	—	2	3	mA	Sleep (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 10 MHz	1, 2
			—	1	—		V_{CC} = 2.5 V, f_{OSC} = 10 MHz	1, 2 Reference value
Subactive mode current dissipation	I_{SUB}	V_{CC}	—	10	20	µA	V_{CC} = 2.5 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/2$)	1, 2
			—	10	—		V_{CC} = 2.5 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/8$)	1, 2 Reference value

Table 13.6 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Subsleep mode current dissipation	I_{SUBSP}	V_{CC}	—	5	10	µA	V_{CC} = 2.5 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/2$)	1, 2
Watch mode current dissipation	I_{WATCH}	V_{CC}	—	—	6	µA	V_{CC} = 2.5 V 32-kHz crystal oscillator	1, 2
Standby mode current dissipation	I_{STBY}	V_{CC}	—	—	5	µA	32-kHz crystal oscillator not used	1, 2
RAM data retaining voltage	V_{RAM}	V_{CC}	2	—	—	V		

Notes: 1. Pin states during current measurement are given below.

Mode	$\overline{RES}$ Pin	Internal State	Other Pins	Oscillator Pins
Active (high-speed) mode	V_{CC}	Operates	V_{CC}	System clock oscillator: ceramic or crystal
Active (medium-speed) mode		Operates ($ø_{OSC}$/128)		Subclock oscillator: Pin X_1 = V_{CC}
Sleep (high-speed) mode	V_{CC}	Only timers operate	V_{CC}	
Sleep (medium-speed) mode		Only timers operate ($ø_{OSC}$/128)		
Subactive mode	V_{CC}	Operates	V_{CC}	System clock oscillator: ceramic or crystal
Subsleep mode	V_{CC}	Only timers operate, CPU stops	V_{CC}	Subclock oscillator: crystal
Watch mode	V_{CC}	Only time base operates, CPU stops	V_{CC}	
Standby mode	V_{CC}	CPU and timers both stop	V_{CC}	System clock oscillator: ceramic or crystal Subclock oscillator: Pin X_1 = V_{CC}

2. Excludes current in pull-up MOS transistors and output buffers.

Table 13.6 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise indicated.

Item		Symbol	Values			Unit
			Min	Typ	Max	
Allowable output low current (per pin)	Output pins except port 6	I_{OL}	—	—	2	mA
	Port 6		—	—	10	
Allowable output low current (total)	Output pins except port 6	ΣI_{OL}	—	—	40	mA
	Port 6		—	—	80	
Allowable output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	mA
Allowable output high current (total)	All output pins	$\Sigma(-I_{OH})$	—	—	30	mA

13.2.5 AC Characteristics (HD6433644, HD6433643, HD6433642, HD6433641, HD6433640)

Table 13.7 lists the control signal timing, and tables 13.8 and 13.9 list the serial interface timing of the HD6433644, the HD6433643, the HD6433642, the HD6433641 and the HD6433640.

Table 13.7 Control Signal Timing

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
System clock oscillation frequency	f_{OSC}	OSC_1, OSC_2	2	—	10	MHz		
			2	—	5		V_{CC} = 2.5 V to 5.5 V	
OSC clock ($ø_{OSC}$) cycle time	t_{OSC}	OSC_1, OSC_2	100	—	1000	ns		*1
			200	—	1000		V_{CC} = 2.5 V to 5.5 V	Figure 13.1
System clock (ø) cycle time	t_{cyc}		2	—	128	t_{OSC}	V_{CC} = 2.5 V to 5.5 V	*1
			—	—	25.6	µs		
Subclock oscillation frequency	f_W	X_1, X_2	—	32.768	—	kHz	V_{CC} = 2.5 V to 5.5 V	
Watch clock ($ø_W$) cycle time	t_W	X_1, X_2	—	30.5	—	µs	V_{CC} = 2.5 V to 5.5 V	
Subclock ($ø_{SUB}$) cycle time	t_{subcyc}		2	—	8	t_W	V_{CC} = 2.5 V to 5.5 V	*2
Instruction cycle time			2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 2.5 V to 5.5 V	
Oscillation stabilization time (crystal oscillator)	t_{rc}	OSC_1, OSC_2	—	—	40	ms		
			—	—	60		V_{CC} = 2.5 V to 5.5 V	
Oscillation stabilization time (ceramic oscillator)	t_{rc}	OSC_1, OSC_2	—	—	20	ms		
			—	—	40		V_{CC} = 2.5 V to 5.5 V	
Oscillation stabilization time	t_{rc}	X_1, X_2	—	—	2	s	V_{CC} = 2.5 V to 5.5 V	
External clock high width	t_{CPH}	OSC_1	40	—	—	ns		Figure 13.1
			80	—	—		V_{CC} = 2.5 V to 5.5 V	
External clock low width	t_{CPL}	OSC_1	40	—	—	ns		
			80	—	—		V_{CC} = 2.5 V to 5.5 V	
External clock rise time	t_{CPr}		—	—	15	ns		
			—	—	20		V_{CC} = 2.5 V to 5.5 V	
External clock fall time	t_{CPf}		—	—	15	ns		
			—	—	20		V_{CC} = 2.5 V to 5.5 V	
Pin $\overline{RES}$ low width	t_{REL}	$\overline{RES}$	10	—	—	t_{cyc}	V_{CC} = 2.5 V to 5.5 V	Figure 13.2

Notes: 1. A frequency between 1 MHz to 10 MHz is required when an external clock is input.
2. Selected with SA1 and SA0 of system clock control register 2 (SYSCR2).

Table 13.7 Control Signal Timing (cont)

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input pin high width	t_{IH}	$\overline{IRQ_0}$ to $\overline{IRQ_3}$, $\overline{INT_0}$ to $\overline{INT_7}$, $\overline{ADTRG}$, TMIB, TMCIV, TMRIV, FTCI, FTIA, FTIB, FTIC, FTID, TRGV	2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 2.5 V to 5.5 V	Figure 13.3
Input pin low width	t_{IL}	$\overline{IRQ_0}$ to $\overline{IRQ_3}$, $\overline{INT_6}$, $\overline{INT_7}$, $\overline{ADTRG}$, TMIB, TMCIV, TMRIV, FTCI, FTIA, FTIB, FTIC, FTID, TRGV	2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 2.5 V to 5.5 V	

Table 13.8 Serial Interface (SCI1) Timing

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input serial clock cycle time	t_{Scyc}	SCK_1	2	—	—	t_{cyc}	V_{CC} = 2.5 V to 5.5 V	Figure 13.4
Input serial clock high width	t_{SCKH}	SCK_1	0.4	—	—	t_{Scyc}	V_{CC} = 2.5 V to 5.5 V	
Input serial clock low width	t_{SCKL}	SCK_1	0.4	—	—	t_{Scyc}	V_{CC} = 2.5 V to 5.5 V	
Input serial clock rise time	t_{SCKr}	SCK_1	—	—	60	ns		
			—	—	80		V_{CC} = 2.5 V to 5.5 V	
Input serial clock fall time	t_{SCKf}	SCK_1	—	—	60	ns		
			—	—	80		V_{CC} = 2.5 V to 5.5 V	
Serial output data delay time	t_{SOD}	SO_1	—	—	200	ns		
			—	—	350		V_{CC} = 2.5 V to 5.5 V	
Serial input data setup time	t_{SIS}	SI_1	180	—	—	ns		
			360	—	—		V_{CC} = 2.5 V to 5.5 V	
Serial input data hold time	t_{SIH}	SI_1	180	—	—	ns		
			360	—	—		V_{CC} = 2.5 V to 5.5 V	

Table 13.9 Serial Interface (SCI3) Timing

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.5 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item		Symbol	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input clock cycle	Asynchronous	t_{Scyc}	4	—	—	t_{cyc}		Figure 13.5
	Synchronous		6	—	—			
Input clock pulse width		t_{SCKW}	0.4	—	0.6	t_{Scyc}		
Transmit data delay time (synchronous)		t_{TXD}	—	—	1	t_{cyc}	V_{CC} = 4.0 V to 5.5 V	Figure 13.6
			—	—	1			
Receive data setup time (synchronous)		t_{RXS}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	
			400.0	—	—			
Receive data hold time (synchronous)		t_{RXH}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	
			400.0	—	—			

13.2.6 A/D Converter Characteristics

Table 13.10 shows the A/D converter characteristics of the HD6473644, the HD6433644, the HD6433643, the HD6433642, the HD6433641 and the HD6433640.

Table 13.10 A/D Converter Characteristics

V_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Values Typ	Values Max	Unit	Test Condition	Reference Figure
Analog power supply voltage	AV_{CC}	AV_{CC}	2.7	—	5.5	V		*1
Analog input voltage	AV_{in}	AN_0 to AN_7	AV_{SS} – 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}	—	150	—	µA		*2 Reference value
	AI_{STOP2}	AV_{CC}	—	—	5	µA		*3
Analog input capacitance	C_{Ain}	AN_0 to AN_7	—	—	30	pF		
Allowable signal source impedance	R_{Ain}		—	—	5.0	kΩ		
Resolution			—	—	8	bit		
Nonlinearity error			—	—	±2.0	LSB		
Quantization error			—	—	±0.5	LSB		
Absolute accuracy			—	—	±2.5	LSB		
Conversion time			12.4	—	124	µs		

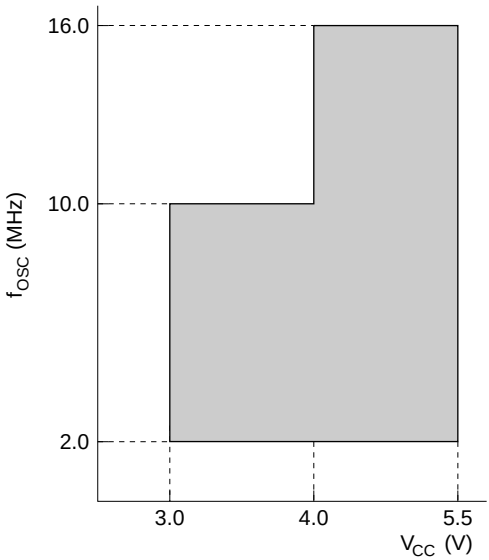
Notes: 1. Set AV_{CC} = V_{CC} when the A/D converter is not used.
2. AI_{STOP1} is the current in active and sleep modes while the A/D converter is idle.
3. AI_{STOP2} is the current at reset and in standby, watch, subactive, and subsleep modes while the A/D converter is idle.

13.3 Electrical Characteristics (F-ZTAT™ Version)

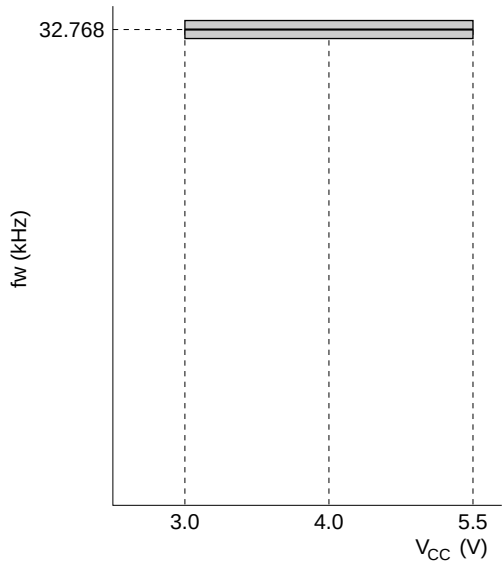
13.3.1 Power Supply Voltage and Operating Range

The power supply voltage and operating range are indicated by the shaded region in the figures below.

1. Power supply voltage vs. oscillator frequency range

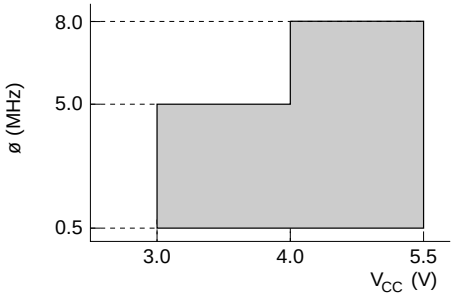


• Active mode (high speed)
• Sleep mode (high speed)

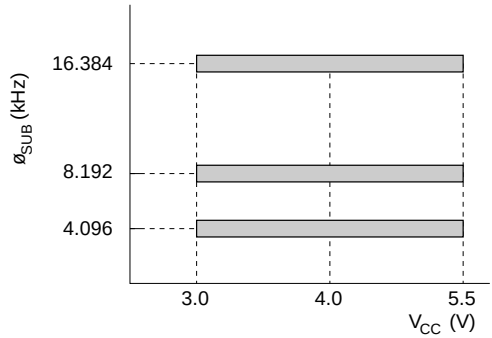


• All operating modes

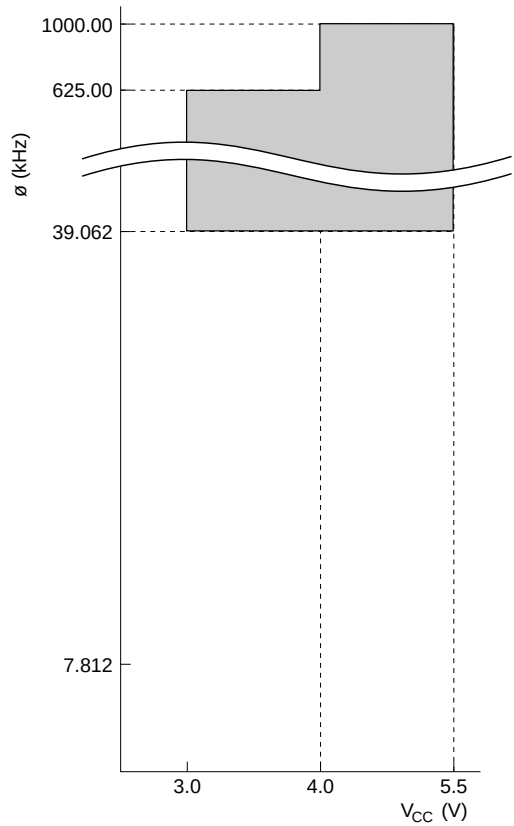
2. Power supply voltage vs. clock frequency range



- Active (high speed) mode
- Sleep (high speed) mode (except CPU)

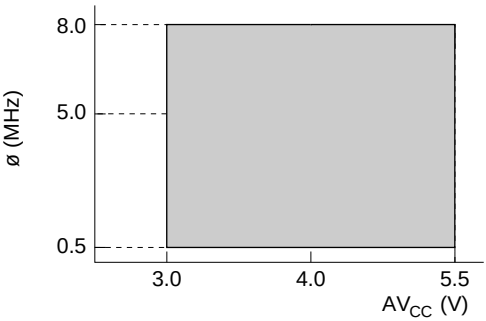


- Subactive mode
- Subsleep mode (except CPU)
- Watch mode (except CPU)



- Active (medium speed) mode
- Sleep (medium speed) mode (except CPU)

3. Analog power supply voltage vs. A/D converter operating range



Do not exceed the maximum conversion time value.

- Active (high speed) mode
- Sleep (high speed) mode

- Active (medium speed) mode
- Sleep (medium speed) mode

13.3.2 DC Characteristics (HD64F3644, HD64F3643, HD64F3642A)

Table 13.11 lists the DC characteristics of the HD64F3644, HD64F3643, and HD64F3642A.

Table 13.11 DC Characteristics

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C unless otherwise indicated.

			Values					
Item	**Symbol**	**Applicable Pins**	**Min**	**Typ**	**Max**	**Unit**	**Test Condition**	**Notes**
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{INT}_0$ to $\overline{INT}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_3$, $\overline{ADTRG}$, TMIB, TMRIV, TMCIV, FTCI, FTIA, FTIB, FTIC, FTID, SCK_1, SCK_3, TRGV	0.8 V_{CC}	—	V_{CC} + 0.3	V		
			0.9 V_{CC}	—	V_{CC} + 0.3		V_{CC} = 3.0 V to 5.5 V including subactive mode	
		SI_1, RXD, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_4$	0.7 V_{CC}	—	V_{CC} + 0.3	V		
			0.8 V_{CC}	—	V_{CC} + 0.3		V_{CC} = 3.0 V to 5.5 V including subactive mode	
		PB_0 to PB_7	0.7 V_{CC}	—	AV_{CC} + 0.3	V		
			0.8 V_{CC}	—	AV_{CC} + 0.3		V_{CC} = 3.0 V to 5.5 V including subactive mode	
		OSC_1	V_{CC} – 0.5	—	V_{CC} + 0.3	V		
			V_{CC} – 0.3	—	V_{CC} + 0.3		V_{CC} = 3.0 V to 5.5 V including subactive mode	

Note: Except in boot mode, connect the TEST pin to V_{SS}.

Table 13.11 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input low voltage	V_{IL}	$\overline{RES}$, $\overline{INT}_0$ to $\overline{INT}_7$, $\overline{IRQ}_0$ to $\overline{IRQ}_3$, $\overline{ADTRG}$, TMIB,	–0.3	—	0.2 V_{CC}	V		
		TMRIV, TMCIV, FTCI, FTIA, FTIB, FTIC, FTID, SCK_1, SCK_3, TRGV	–0.3	—	0.1 V_{CC}		V_{CC} = 3.0 V to 5.5 V including subactive mode	
		SI_1, RXD, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$,	–0.3	—	0.3 V_{CC}	V		
		$P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_4$, PB_0 to PB_7	–0.3	—	0.2 V_{CC}		V_{CC} = 3.0 V to 5.5 V including subactive mode	
		OSC_1	–0.3	—	0.5	V		
			–0.3	—	0.3		V_{CC} = 3.0 V to 5.5 V including subactive mode	

Table 13.11 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Output high voltage	V_{OH}	$P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$,	V_{CC} – 1.0	—	—	V	$-I_{OH}$ = 1.5 mA	
		$P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_4$	V_{CC} – 0.5	—	—		V_{CC} = 3.0 V to 5.5 V $-I_{OH}$ = 0.1 mA	
Output low voltage	V_{OL}	$P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$,	—	—	0.6	V	I_{OL} = 1.6 mA	
		$P5_0$ to $P5_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_4$	—	—	0.4		V_{CC} = 3.0 V to 5.5 V I_{OL} = 0.4 mA	
		$P6_0$ to $P6_7$	—	—	1.0	V	I_{OL} = 10.0 mA	
			—	—	0.4		I_{OL} = 1.6 mA	
			—	—	0.4		V_{CC} = 2.7 V to 5.5 V I_{OL} = 0.4 mA	
Input/ output leakage current	$\lvert I_{IL} \rvert$	OSC_1, $\overline{RES}$, $P1_0$, $P1_4$ to $P1_7$, $P2_0$ to $P2_2$, $P3_0$ to $P3_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_3$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_4$	—	—	1.0	µA	V_{in} = 0.5 V to (V_{CC} – 0.5 V)	
		PB_0 to PB_7	—	—	1.0	µA	V_{in} = 0.5 V to (AV_{CC} – 0.5 V)	
Input leakage current	$\lvert I_{IL} \rvert$	$\overline{IRQ_0}$, TEST	—	—	20	µA	V_{in} = 0.5 V to (V_{CC} – 0.5 V)	
Pull-up MOS current	$-I_p$	$P1_0$, $P1_4$ to $P1_7$, $P3_0$ to $P3_2$,	50	—	300	µA	V_{CC} = 5 V, V_{in} = 0 V	
		$P5_0$ to $P5_7$	—	35	—		V_{CC} = 3.0 V, V_{in} = 0 V	Reference value

Table 13.11 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Input capacitance	C_{in}	All input pins except TEST	—	—	15.0	pF	f = 1 MHz, V_{in} = 0 V, T_a = 25°C	
		$\overline{IRQ_0}$, TEST	—	—	30.0			
Active mode current dissipation	I_{OPE1}	V_{CC}	—	15	25	mA	Active (high-speed) mode V_{CC} = 5 V, f_{OSC} = 16 MHz	1, 2
			—	8.5	—		V_{CC} = 3.0 V, f_{OSC} = 10 MHz	1, 2 Reference value
	I_{OPE2}	V_{CC}	—	3	5	mA	Active (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 16 MHz	1, 2
			—	2	—		V_{CC} = 3.0 V, f_{OSC} = 10 MHz	1, 2 Reference value
Sleep mode current dissipation	I_{SLEEP1}	V_{CC}	—	6	10	mA	Sleep (high-speed) mode V_{CC} = 5 V, f_{OSC} = 16 MHz	1, 2
			—	3.5	—		V_{CC} = 3.0 V, f_{OSC} = 10 MHz	1, 2 Reference value
	I_{SLEEP2}	V_{CC}	—	2	4	mA	Sleep (medium-speed) mode V_{CC} = 5 V, f_{OSC} = 16 MHz	1, 2
			—	1	—		V_{CC} = 3.0 V, f_{OSC} = 10 MHz	1, 2 Reference value
Subactive mode current dissipation	I_{SUB}	V_{CC}	—	1	2	mA	V_{CC} = 3.0 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/2$)	1, 2
			—	1	—	mA	V_{CC} = 3.0 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/8$)	1, 2 Reference value

Table 13.11 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{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	Notes
Subsleep mode current dissipation	I_{SUBSP}	V_{CC}	—	5	10	µA	V_{CC} = 3.0 V 32-kHz crystal oscillator ($ø_{SUB} = ø_W/2$)	1, 2
Watch mode current dissipation	I_{WATCH}	V_{CC}	—	—	8	µA	V_{CC} = 3.0 V 32-kHz crystal oscillator	1, 2
Standby mode current dissipation	I_{STBY}	V_{CC}	—	—	5	µA	32-kHz crystal oscillator not used	1, 2
RAM data retaining voltage	V_{RAM}	V_{CC}	2	—	—	V		

Notes: 1. Pin states during current measurement are given below.

Mode	$\overline{RES}$ Pin	Internal State	Other Pins	Oscillator Pins
Active (high-speed) mode	V_{CC}	Operates	V_{CC}	System clock oscillator: ceramic or crystal
Active (medium-speed) mode		Operates ($ø_{OSC}$/128)		Subclock oscillator: Pin $X_1 = V_{CC}$
Sleep (high-speed) mode	V_{CC}	Only timers operate	V_{CC}	
Sleep (medium-speed) mode		Only timers operate ($ø_{OSC}$/128)		
Subactive mode	V_{CC}	Operates	V_{CC}	System clock oscillator: ceramic or crystal
Subsleep mode	V_{CC}	Only timers operate, CPU stops	V_{CC}	Subclock oscillator: crystal
Watch mode	V_{CC}	Only time base operates, CPU stops	V_{CC}	
Standby mode	V_{CC}	CPU and timers both stop	V_{CC}	System clock oscillator: ceramic or crystal Subclock oscillator: Pin $X_1 = V_{CC}$

2. Excludes current in pull-up MOS transistors and output buffers.

Table 13.11 DC Characteristics (cont)

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise indicated.

Item		Symbol	Values			Unit
			Min	Typ	Max	
Allowable output low current (per pin)	Output pins except port 6	I_{OL}	—	—	2	mA
	Port 6		—	—	10	
Allowable output low current (total)	Output pins except port 6	ΣI_{OL}	—	—	40	mA
	Port 6		—	—	80	
Allowable output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	mA
Allowable output high current (total)	All output pins	$\Sigma(-I_{OH})$	—	—	30	mA

13.3.3 AC Characteristics (HD64F3644, HD64F3643, HD64F3642A)

Table 13.12 lists the control signal timing, and tables 13.13 and 13.14 list the serial interface timing of the HD64F3644, HD64F3643, and HD64F3642A.

Table 13.12 Control Signal Timing

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
System clock oscillation frequency	f_{OSC}	OSC_1, OSC_2	2	—	16	MHz		
			2	—	10		V_{CC} = 3.0 V to 5.5 V	
OSC clock ($ø_{OSC}$) cycle time	t_{OSC}	OSC_1, OSC_2	62.5	—	1000	ns		*1 Figure 13.1
			100	—	1000		V_{CC} = 3.0 V to 5.5 V	
System clock (ø) cycle time	t_{cyc}		2	—	128	t_{OSC}	V_{CC} = 3.0 V to 5.5 V	*1
			—	—	25.6	µs		
Subclock oscillation frequency	f_W	X_1, X_2	—	32.768	—	kHz	V_{CC} = 3.0 V to 5.5 V	
Watch clock ($ø_W$) cycle time	t_W	X_1, X_2	—	30.5	—	µs	V_{CC} = 3.0 V to 5.5 V	
Subclock ($ø_{SUB}$) cycle time	t_{subcyc}		2	—	8	t_W	V_{CC} = 3.0 V to 5.5 V	*2
Instruction cycle time			2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 3.0 V to 5.5 V	
Oscillation stabilization time (crystal oscillator)	t_{rc}	OSC_1, OSC_2	—	—	40	ms		
			—	—	60		V_{CC} = 3.0 V to 5.5 V	
Oscillation stabilization time (ceramic oscillator)	t_{rc}	OSC_1, OSC_2	—	—	20	ms		
			—	—	40		V_{CC} = 3.0 V to 5.5 V	
Oscillation stabilization time	t_{rc}	X_1, X_2	—	—	2	s	V_{CC} = 3.0 V to 5.5 V	
External clock high width	t_{CPH}	OSC_1	20	—	—	ns		Figure 13.1
			40	—	—		V_{CC} = 3.0 V to 5.5 V	
External clock low width	t_{CPL}	OSC_1	20	—	—	ns		
			40	—	—		V_{CC} = 3.0 V to 5.5 V	
External clock rise time	t_{CPr}		—	—	15	ns	V_{CC} = 3.0 V to 5.5 V	
External clock fall time	t_{CPf}		—	—	15	ns	V_{CC} = 3.0 V to 5.5 V	

Notes: 1. A frequency between 1 MHz to 10 MHz is required when an external clock is input.
2. Selected with SA1 and SA0 of system clock control register 2 (SYSCR2).

Table 13.12 Control Signal Timing (cont)

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Pin $\overline{RES}$ low width	t_{REL}	$\overline{RES}$	10	—	—	t_{cyc}	V_{CC} = 3.0 V to 5.5 V	Figure 13.2
Input pin high level width	t_{IH}	$\overline{IRQ_0}$ to $\overline{IRQ_3}$, $\overline{INT_0}$ to $\overline{INT_7}$, $\overline{ADTRG}$, TMIB, TMCIV, TMRIV, FTCI, FTIA, FTIB, FTIC, FTID, TRGV	2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 3.0 V to 5.5 V	Figure 13.3
Input pin low level width	t_{IL}	$\overline{IRQ_0}$ to $\overline{IRQ_3}$, $\overline{INT_6}$, $\overline{INT_7}$, $\overline{ADTRG}$, TMIB, TMCIV, TMRIV, FTCI, FTIA, FTIB, FTIC, FTID, TRGV	2	—	—	t_{cyc} t_{subcyc}	V_{CC} = 3.0 V to 5.5 V	

Table 13.13 Serial Interface (SCI1) Timing

V_{CC} = 4.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input serial clock cycle time	t_{Scyc}	SCK_1	2	—	—	t_{cyc}	V_{CC} = 3.0 V to 5.5 V	Figure 13.4
Input serial clock high width	t_{SCKH}	SCK_1	0.4	—	—	t_{Scyc}	V_{CC} = 3.0 V to 5.5 V	
Input serial clock low width	t_{SCKL}	SCK_1	0.4	—	—	t_{Scyc}	V_{CC} = 3.0 V to 5.5 V	
Input serial clock rise time	t_{SCKr}	SCK_1	—	—	60	ns		
			—	—	80		V_{CC} = 3.0 V to 5.5 V	
Input serial clock fall time	t_{SCKf}	SCK_1	—	—	60	ns		
			—	—	80		V_{CC} = 3.0 V to 5.5 V	
Serial output data delay time	t_{SOD}	SO_1	—	—	200	ns		
			—	—	350		V_{CC} = 3.0 V to 5.5 V	
Serial input data setup time	t_{SIS}	SI_1	180	—	—	ns		
			360	—	—		V_{CC} = 3.0 V to 5.5 V	
Serial input data hold time	t_{SIH}	SI_1	180	—	—	ns		
			360	—	—		V_{CC} = 3.0 V to 5.5 V	

Table 13.14 Serial Interface (SCI3) Timing

V_{CC} = 3.0 V to 5.5 V, AV_{CC} = 3.0 V to 5.5 V, V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item		Symbol	Values Min	Typ	Max	Unit	Test Condition	Reference Figure
Input clock cycle	Asynchronous	t_{Scyc}	4	—	—	t_{cyc}		Figure 13.5
	Synchronous		6	—	—			
Input clock pulse width		t_{SCKW}	0.4	—	0.6	t_{Scyc}		
Transmit data delay time (synchronous)		t_{TXD}	—	—	1	t_{cyc}	V_{CC} = 4.0 V to 5.5 V	Figure 13.6
			—	—	1			
Receive data setup time (synchronous)		t_{RXS}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	
			400.0	—	—			
Receive data hold time (synchronous)		t_{RXH}	200.0	—	—	ns	V_{CC} = 4.0 V to 5.5 V	
			400.0	—	—			

13.3.4 A/D Converter Characteristics

Table 13.15 shows the A/D converter characteristics of the HD64F3644, HD64F3643, and HD64F3642A.

Table 13.15 A/D Converter Characteristics

V_{CC} = 3.0 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Values Min	Values Typ	Values Max	Unit	Test Condition	Reference Figure
Analog power supply voltage	AV_{CC}	AV_{CC}	3.0	—	5.5	V		*1
Analog input voltage	AV_{in}	AN_0 to AN_7	AV_{SS} – 0.3	—	AV_{SS} + 0.3	V		
Analog power supply current	AI_{OPE}	AV_{CC}	—	—	1.5	mA	AV_{CC} = 5.0 V	
	AI_{STOP1}	AV_{CC}	—	150	—	µA		*2 Reference value
	AI_{STOP2}	AV_{CC}	—	—	5.0	µA		*3
Analog input capacitance	C_{Ain}	AN_0 to AN_7	—	—	30.0	pF		
Allowable signal source impedance	R_{Ain}		—	—	5.0	kΩ		
Resolution			—	—	8	bit		
Nonlinearity error			—	—	±2.0	LSB		
Quantization error			—	—	±0.5	LSB		
Absolute accuracy			—	—	±2.5	LSB		
Conversion time			7.75	—	124	µs		

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.

13.4 Operation Timing

Figures 13.1 to 13.6 show timing diagrams.

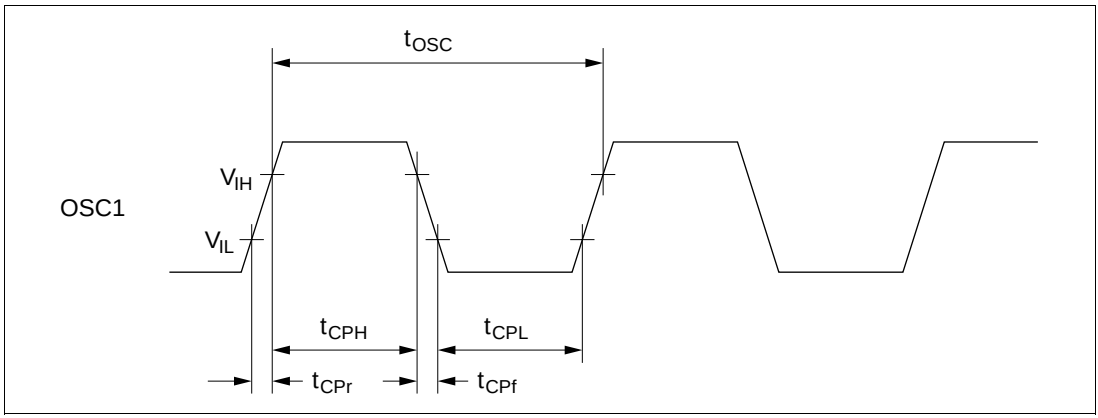


Figure 13.1 System Clock Input Timing

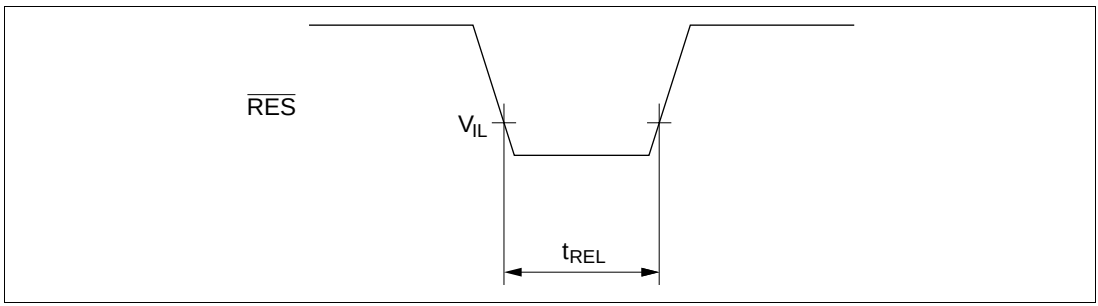


Figure 13.2 $\overline{RES}$ Low Width Timing

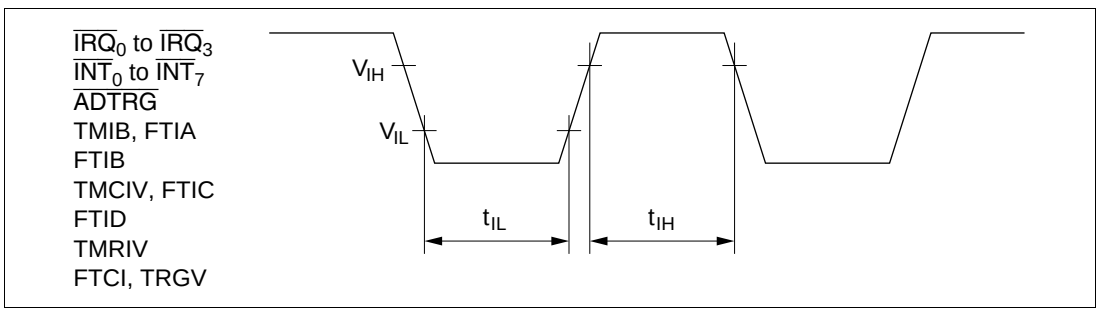
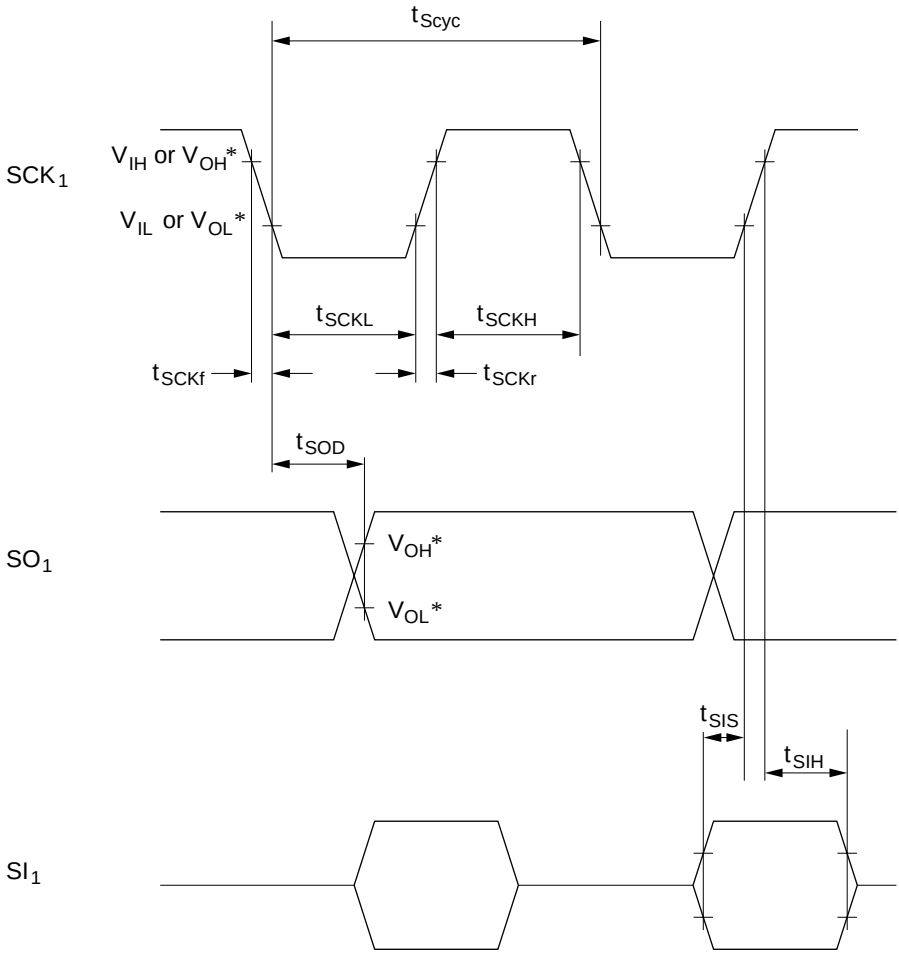


Figure 13.3 Input Timing



Note: * Output timing reference levels
Output high: V_{OH} = 2.0 V
Output low: V_{OL} = 0.8 V
Load conditions are shown in figure 13.7.

Figure 13.4 Serial Interface 1, 2 Input/Output Timing

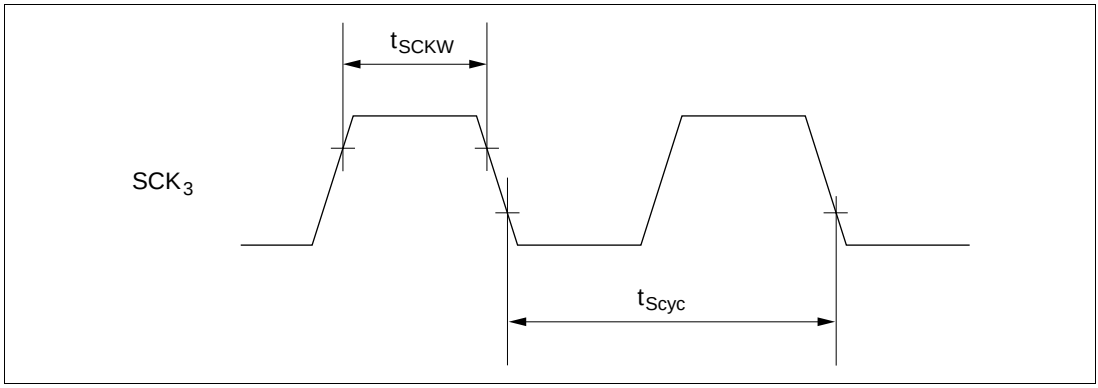
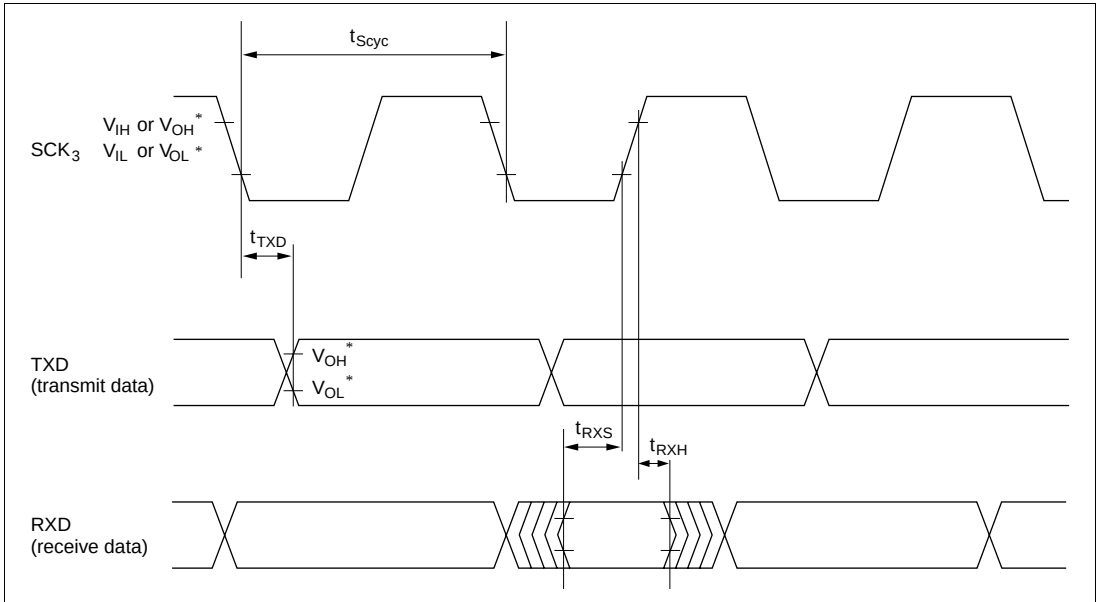


Figure 13.5 SCK_3 Input Clock Timing



Note: * Output timing reference levels
Output high: V_{OH} = 2.0 V
Output low: V_{OL} = 0.8 V
Load conditions are shown in figure 13.7.

Figure 13.6 Serial Interface 3 Synchronous Mode Input/Output Timing

13.5 Output Load Circuit

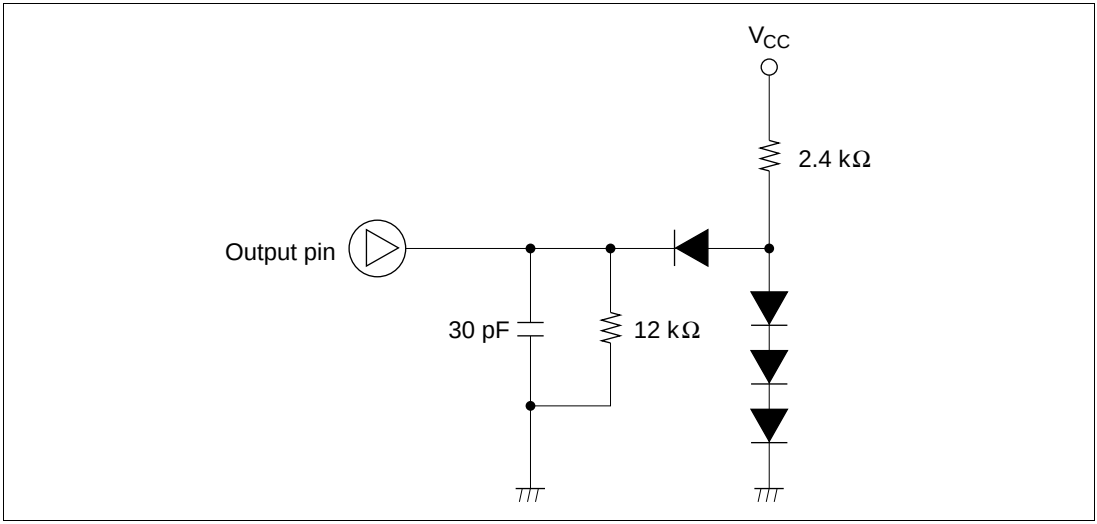


Figure 13.7 Output Load Condition

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)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
MOV.B #xx:8, Rd	B	#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
MOV.B Rs, Rd	B	Rs8 → Rd8		2								—	—	↕	↕	0	—	2
MOV.B @Rs, Rd	B	@Rs16 → Rd8			2							—	—	↕	↕	0	—	4
MOV.B @(d:16, Rs), Rd	B	@(d:16, Rs16) → Rd8				4						—	—	↕	↕	0	—	6
MOV.B @Rs+, Rd	B	@Rs16 → Rd8 Rs16+1 → Rs16					2					—	—	↕	↕	0	—	6
MOV.B @aa:8, Rd	B	@aa:8 → Rd8						2				—	—	↕	↕	0	—	4
MOV.B @aa:16, Rd	B	@aa:16 → Rd8						4				—	—	↕	↕	0	—	6
MOV.B Rs, @Rd	B	Rs8 → @Rd16			2							—	—	↕	↕	0	—	4
MOV.B Rs, @(d:16, Rd)	B	Rs8 → @(d:16, Rd16)				4						—	—	↕	↕	0	—	6
MOV.B Rs, @–Rd	B	Rd16–1 → Rd16 Rs8 → @Rd16					2					—	—	↕	↕	0	—	6
MOV.B Rs, @aa:8	B	Rs8 → @aa:8						2				—	—	↕	↕	0	—	4
MOV.B Rs, @aa:16	B	Rs8 → @aa:16						4				—	—	↕	↕	0	—	6
MOV.W #xx:16, Rd	W	#xx:16 → Rd	4									—	—	↕	↕	0	—	4
MOV.W Rs, Rd	W	Rs16 → Rd16		2								—	—	↕	↕	0	—	2
MOV.W @Rs, Rd	W	@Rs16 → Rd16			2							—	—	↕	↕	0	—	4
MOV.W @(d:16, Rs), Rd	W	@(d:16, Rs16) → Rd16				4						—	—	↕	↕	0	—	6
MOV.W @Rs+, Rd	W	@Rs16 → Rd16 Rs16+2 → Rs16					2					—	—	↕	↕	0	—	6
MOV.W @aa:16, Rd	W	@aa:16 → Rd16						4				—	—	↕	↕	0	—	6
MOV.W Rs, @Rd	W	Rs16 → @Rd16			2							—	—	↕	↕	0	—	4
MOV.W Rs, @(d:16, Rd)	W	Rs16 → @(d:16, Rd16)				4						—	—	↕	↕	0	—	6
MOV.W Rs, @–Rd	W	Rd16–2 → Rd16 Rs16 → @Rd16					2					—	—	↕	↕	0	—	6
MOV.W Rs, @aa:16	W	Rs16 → @aa:16						4				—	—	↕	↕	0	—	6
POP Rd	W	@SP → Rd16 SP+2 → SP					2					—	—	↕	↕	0	—	6

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
PUSH Rs	W	SP–2 → SP Rs16 → @SP					2					—	—	↕	↕	0	—	6
EEPMOV	—	if R4L≠0 then Repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4L–1 → R4L Until R4L=0 else next;									4			↕	↕	—	—	(4)
ADD.B #xx:8, Rd	B	Rd8+#xx:8 → Rd8	2									—	↕	↕	↕	↕	↕	2
ADD.B Rs, Rd	B	Rd8+Rs8 → Rd8		2								—	↕	↕	↕	↕	↕	2
ADD.W Rs, Rd	W	Rd16+Rs16 → Rd16		2								—	(1)	↕	↕	↕	↕	2
ADDX.B #xx:8, Rd	B	Rd8+#xx:8 +C → Rd8	2									—	↕	↕	(2)	↕	↕	2
ADDX.B Rs, Rd	B	Rd8+Rs8 +C → Rd8		2								—	↕	↕	(2)	↕	↕	2
ADDS.W #1, Rd	W	Rd16+1 → Rd16		2								—	—	—	—	—	—	2
ADDS.W #2, Rd	W	Rd16+2 → Rd16		2								—	—	—	—	—	—	2
INC.B Rd	B	Rd8+1 → Rd8		2								—	—	↕	↕	↕	—	2
DAA.B Rd	B	Rd8 decimal adjust → Rd8		2								—	*	↕	↕	*	(3)	2
SUB.B Rs, Rd	B	Rd8–Rs8 → Rd8		2								—	↕	↕	↕	↕	↕	2
SUB.W Rs, Rd	W	Rd16–Rs16 → Rd16		2								—	(1)	↕	↕	↕	↕	2
SUBX.B #xx:8, Rd	B	Rd8–#xx:8 –C → Rd8	2									—	↕	↕	(2)	↕	↕	2
SUBX.B Rs, Rd	B	Rd8–Rs8 –C → Rd8		2								—	↕	↕	(2)	↕	↕	2
SUBS.W #1, Rd	W	Rd16–1 → Rd16		2								—	—	—	—	—	—	2
SUBS.W #2, Rd	W	Rd16–2 → Rd16		2								—	—	—	—	—	—	2
DEC.B Rd	B	Rd8–1 → Rd8		2								—	—	↕	↕	↕	—	2
DAS.B Rd	B	Rd8 decimal adjust → Rd8		2								—	*	↕	↕	*	—	2
NEG.B Rd	B	0–Rd → Rd		2								—	↕	↕	↕	↕	↕	2
CMP.B #xx:8, Rd	B	Rd8–#xx:8	2									—	↕	↕	↕	↕	↕	2
CMP.B Rs, Rd	B	Rd8–Rs8		2								—	↕	↕	↕	↕	↕	2
CMP.W Rs, Rd	W	Rd16–Rs16		2								—	(1)	↕	↕	↕	↕	2
MULXU.B Rs, Rd	B	Rd8 × Rs8 → Rd16		2								—	—	—	—	—	—	14

Table A.1 Instruction Set (cont)

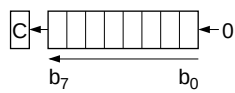
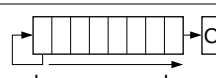
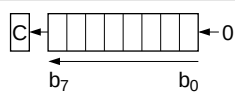
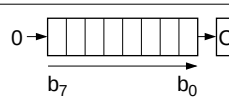
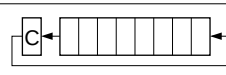
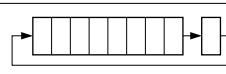
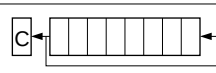
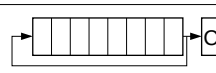
Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
DIVXU.B Rs, Rd	B	Rd16÷Rs8 → Rd16 (RdH: remainder, RdL: quotient)		2								—	—	(5)	(6)	—	—	14
AND.B #xx:8, Rd	B	Rd8∧#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
AND.B Rs, Rd	B	Rd8∧Rs8 → Rd8		2								—	—	↕	↕	0	—	2
OR.B #xx:8, Rd	B	Rd8∨#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
OR.B Rs, Rd	B	Rd8∨Rs8 → Rd8		2								—	—	↕	↕	0	—	2
XOR.B #xx:8, Rd	B	Rd8⊕#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
XOR.B Rs, Rd	B	Rd8⊕Rs8 → Rd8		2								—	—	↕	↕	0	—	2
NOT.B Rd	B	$\overline{\text{Rd}}$ → 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	↕	2
ROTXL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTXR.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTR.B Rd	B			2								—	—	↕	↕	0	↕	2

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
BSET #xx:3, Rd	B	(#xx:3 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BSET Rn, Rd	B	(Rn8 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET Rn, @Rd	B	(Rn8 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET Rn, @aa:8	B	(Rn8 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BCLR #xx:3, Rd	B	(#xx:3 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BCLR Rn, Rd	B	(Rn8 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR Rn, @Rd	B	(Rn8 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR Rn, @aa:8	B	(Rn8 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BNOT #xx:3, Rd	B	(#xx:3 of Rd8) ← ($\overline{\text{#xx:3 of Rd8}}$)		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
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

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
BLD #xx:3, Rd	B	(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BLD #xx:3, @Rd	B	(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BLD #xx:3, @aa:8	B	(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BILD #xx:3, Rd	B	$(\overline{\text{#xx:3 of Rd8}}) \rightarrow$ C		2								—	—	—	—	—	↕	2
BILD #xx:3, @Rd	B	$(\overline{\text{#xx:3 of @Rd16}}) \rightarrow$ C			4							—	—	—	—	—	↕	6
BILD #xx:3, @aa:8	B	$(\overline{\text{#xx:3 of @aa:8}}) \rightarrow$ C						4				—	—	—	—	—	↕	6
BST #xx:3, Rd	B	C → (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BST #xx:3, @Rd	B	C → (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BST #xx:3, @aa:8	B	C → (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BIST #xx:3, Rd	B	$\overline{\text{C}}$ → (#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
BIOR #xx:3, @aa:8	B	C∨$(\overline{\text{#xx:3 of @aa:8}})$ → C						4				—	—	—	—	—	↕	6
BXOR #xx:3, Rd	B	C⊕(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BXOR #xx:3, @Rd	B	C⊕(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIXOR #xx:3, Rd	B	C⊕$(\overline{\text{#xx:3 of Rd8}})$ → C		2								—	—	—	—	—	↕	2

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Branching Condition	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
				#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
BIXOR #xx:3, @Rd	B	C⊕(#xx:3 of @Rd16) → C				4							—	—	—	—	—	↕	6
BIXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C							4				—	—	—	—	—	↕	6
BRA d:8 (BT d:8)	—	PC ← PC+d:8								2			—	—	—	—	—	—	4
BRN d:8 (BF d:8)	—	PC ← PC+2								2			—	—	—	—	—	—	4
BHI d:8	—	If condition is true then PC ← PC+d:8 else next;	C ∨ Z = 0							2			—	—	—	—	—	—	4
BLS d:8	—		C ∨ Z = 1							2			—	—	—	—	—	—	4
BCC d:8 (BHS d:8)	—		C = 0							2			—	—	—	—	—	—	4
BCS d:8 (BLO d:8)	—		C = 1							2			—	—	—	—	—	—	4
BNE d:8	—		Z = 0							2			—	—	—	—	—	—	4
BEQ d:8	—		Z = 1							2			—	—	—	—	—	—	4
BVC d:8	—		V = 0							2			—	—	—	—	—	—	4
BVS d:8	—		V = 1							2			—	—	—	—	—	—	4
BPL d:8	—		N = 0							2			—	—	—	—	—	—	4
BMI d:8	—		N = 1							2			—	—	—	—	—	—	4
BGE d:8	—		N⊕V = 0							2			—	—	—	—	—	—	4
BLT d:8	—		N⊕V = 1							2			—	—	—	—	—	—	4
BGT d:8	—		Z ∨ (N⊕V) = 0							2			—	—	—	—	—	—	4
BLE d:8	—		Z ∨ (N⊕V) = 1							2			—	—	—	—	—	—	4
JMP @Rn	—	PC ← Rn16				2							—	—	—	—	—	—	4
JMP @aa:16	—	PC ← aa:16							4				—	—	—	—	—	—	6
JMP @@aa:8	—	PC ← @aa:8									2		—	—	—	—	—	—	8
BSR d:8	—	SP–2 → SP PC → @SP PC ← PC+d:8								2			—	—	—	—	—	—	6
JSR @Rn	—	SP–2 → SP PC → @SP PC ← Rn16				2							—	—	—	—	—	—	6
JSR @aa:16	—	SP–2 → SP PC → @SP PC ← aa:16							4				—	—	—	—	—	—	8

Table A.1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (Bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
JSR @@aa:8		SP–2 → SP PC → @SP PC ← @aa:8								2		—	—	—	—	—	—	8
RTS	—	PC ← @SP SP+2 → SP									2	—	—	—	—	—	—	8
RTE	—	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
SLEEP	—	Transit to sleep mode.									2	—	—	—	—	—	—	2
LDC #xx:8, CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
LDC Rs, CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
STC CCR, Rd	B	CCR → Rd8		2								—	—	—	—	—	—	2
ANDC #xx:8, CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
ORC #xx:8, CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
XORC #xx:8, CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
NOP	—	PC ← PC+2									2	—	—	—	—	—	—	2
EEPMOV	—	if R4L≠0 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).

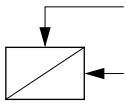

Instruction when first bit of byte 2 (bit 7 of first instruction word) is 0.
Instruction when first bit of byte 2 (bit 7 of first instruction word) is 1.

Table A.2 Operation Code Map

High \ Low	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	SLEEP	STC	LDC	ORC	XORC	ANDC	LDC	ADD		INC	ADDS	MOV		ADDX	DAA
1	SHLL / SHAL	SHLR / SHAR	ROTXL / ROTL	ROTXR / ROTR	OR	XOR	AND	NOT / NEG	SUB		DEC	SUBS	CMP		SUBX	DAS
2	MOV															
3																
4	BRA	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU			RTS	BSR	RTE		JMP					JSR		
6	BSET	BNOT	BCLR	BTST				BST / BIST	MOV*							
7					BOR / BIOR	BXOR / BIXOR	BAND / BIAND	BLD / BILD		MOV		EEPMOV	Bit-manipulation instructions			
8	ADD															
9	ADDX															
A	CMP															
B	SUBX															
C	OR															
D	XOR															
E	AND															
F	MOV															

Note: * The PUSH and POP instructions are identical in machine language to MOV instructions.

A.3 Number of Execution States

The tables here can be used to calculate the number of states required for instruction execution. Table A.3 indicates the number of states required for each cycle (instruction fetch, branch address read, stack operation, byte data access, word data access, internal operation).
Table A.4 indicates the number of cycles of each type occurring in each instruction. The total number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

Examples: 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, J = K = M = N= 0

From table A.3:
$S_I = 2$, $S_L = 2$

Number of states required for execution = $2 \times 2 + 2 \times 2 = 8$

When instruction is fetched from on-chip ROM, branch address is read from on-chip ROM, and on-chip RAM is used for stack area.

JSR @@ 30

From table A.4:
I = 2, J = K = 1, L = M = N = 0

From table A.3:
$S_I = S_J = S_K = 2$

Number of states required for execution = $2 \times 2 + 1 \times 2 + 1 \times 2 = 8$

Table A.3 Number of Cycles in Each Instruction

Execution Status (Instruction Cycle)		Access Location	
		On-Chip Memory	On-Chip Peripheral Module
Instruction fetch	S_I	2	—
Branch address read	S_J		
Stack operation	S_K		
Byte data access	S_L		2 or 3*
Word data access	S_M		—
Internal operation	S_N	1	

Note: * Depends on which on-chip module is accessed. See 2.9.1, Notes on Data Access for details.

Table A.4 Number of Cycles in Each Instruction

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
ADD	ADD.B #xx:8, Rd	1					
	ADD.B Rs, Rd	1					
	ADD.W Rs, Rd	1					
ADDS	ADDS.W #1, Rd	1					
	ADDS.W #2, Rd	1					
ADDX	ADDX.B #xx:8, Rd	1					
	ADDX.B Rs, Rd	1					
AND	AND.B #xx:8, Rd	1					
	AND.B Rs, Rd	1					
ANDC	ANDC #xx:8, CCR	1					
BAND	BAND #xx:3, Rd	1					
	BAND #xx:3, @Rd	2			1		
	BAND #xx:3, @aa:8	2			1		
Bcc	BRA d:8 (BT d:8)	2					
	BRN d:8 (BF d:8)	2					
	BHI d:8	2					
	BLS d:8	2					
	BCC d:8 (BHS d:8)	2					
	BCS d:8 (BLO d:8)	2					
	BNE d:8	2					
	BEQ d:8	2					
	BVC d:8	2					
	BVS d:8	2					
	BPL d:8	2					
	BMI d:8	2					
	BGE d:8	2					
	BLT d:8	2					
	BGT d:8	2					
	BLE d:8	2					
BCLR	BCLR #xx:3, Rd	1					
	BCLR #xx:3, @Rd	2			2		
	BCLR #xx:3, @aa:8	2			2		
	BCLR Rn, Rd	1					

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BCLR	BCLR Rn, @Rd	2			2		
	BCLR Rn, @aa:8	2			2		
BIAND	BIAND #xx:3, Rd	1					
	BIAND #xx:3, @Rd	2			1		
	BIAND #xx:3, @aa:8	2			1		
BILD	BILD #xx:3, Rd	1					
	BILD #xx:3, @Rd	2			1		
	BILD #xx:3, @aa:8	2			1		
BIOR	BIOR #xx:3, Rd	1					
	BIOR #xx:3, @Rd	2			1		
	BIOR #xx:3, @aa:8	2			1		
BIST	BIST #xx:3, Rd	1					
	BIST #xx:3, @Rd	2			2		
	BIST #xx:3, @aa:8	2			2		
BIXOR	BIXOR #xx:3, Rd	1					
	BIXOR #xx:3, @Rd	2			1		
	BIXOR #xx:3, @aa:8	2			1		
BLD	BLD #xx:3, Rd	1					
	BLD #xx:3, @Rd	2			1		
	BLD #xx:3, @aa:8	2			1		
BNOT	BNOT #xx:3, Rd	1					
	BNOT #xx:3, @Rd	2			2		
	BNOT #xx:3, @aa:8	2			2		
	BNOT Rn, Rd	1					
	BNOT Rn, @Rd	2			2		
	BNOT Rn, @aa:8	2			2		
BOR	BOR #xx:3, Rd	1					
	BOR #xx:3, @Rd	2			1		
	BOR #xx:3, @aa:8	2			1		
BSET	BSET #xx:3, Rd	1					
	BSET #xx:3, @Rd	2			2		
	BSET #xx:3, @aa:8	2			2		
	BSET Rn, Rd	1					
	BSET Rn, @Rd	2			2		

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BSET	BSET Rn, @aa:8	2			2		
BSR	BSR d:8	2		1			
BST	BST #xx:3, Rd	1					
	BST #xx:3, @Rd	2			2		
	BST #xx:3, @aa:8	2			2		
BTST	BTST #xx:3, Rd	1					
	BTST #xx:3, @Rd	2			1		
	BTST #xx:3, @aa:8	2			1		
	BTST Rn, Rd	1					
	BTST Rn, @Rd	2			1		
	BTST Rn, @aa:8	2			1		
BXOR	BXOR #xx:3, Rd	1					
	BXOR #xx:3, @Rd	2			1		
	BXOR #xx:3, @aa:8	2			1		
CMP	CMP. B #xx:8, Rd	1					
	CMP. B Rs, Rd	1					
	CMP.W Rs, Rd	1					
DAA	DAA.B Rd	1					
DAS	DAS.B Rd	1					
DEC	DEC.B Rd	1					
DIVXU	DIVXU.B Rs, Rd	1					12
EEPMOV	EEPMOV	2			2n+2*		1
INC	INC.B Rd	1					
JMP	JMP @Rn	2					
	JMP @aa:16	2					2
	JMP @@aa:8	2	1				2
JSR	JSR @Rn	2		1			
	JSR @aa:16	2		1			2
	JSR @@aa:8	2	1	1			
LDC	LDC #xx:8, CCR	1					
	LDC Rs, CCR	1					
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					

Note: n: Initial value in R4L. The source and destination operands are accessed n + 1 times each.

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
MOV	MOV.B @Rs, Rd	1		1			
	MOV.B @(d:16, Rs), Rd	2		1			
	MOV.B @Rs+, Rd	1		1			2
	MOV.B @aa:8, Rd	1		1			
	MOV.B @aa:16, Rd	2		1			
	MOV.B Rs, @Rd	1			1		
	MOV.B Rs, @(d:16, Rd)	2			1		
	MOV.B Rs, @–Rd	1			1		2
	MOV.B Rs, @aa:8	1			1		
	MOV.B Rs, @aa:16	2			1		
	MOV.W #xx:16, Rd	2					
	MOV.W Rs, Rd	1					
	MOV.W @Rs, Rd	1				1	
	MOV.W @(d:16, Rs), Rd	2				1	
	MOV.W @Rs+, Rd	1				1	2
	MOV.W @aa:16, Rd	2				1	
	MOV.W Rs, @Rd	1				1	
	MOV.W Rs, @(d:16d)	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					

Table A.4 Number of Cycles in Each Instruction (cont)

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
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					

Appendix B Internal I/O Registers

B.1 Addresses

Address	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'F740										
H'F741										
H'F742										
H'F743										
H'F744										
H'F770	TIER	ICIAE	ICIBE	ICICE	ICIDE	OCIAE	OCIBE	OVIE	—	Timer X
H'F771	TCSRX	ICFA	ICFB	ICFC	ICFD	OCFA	OCFB	OVF	CCLRA	
H'F772	FRCH	FRCH7	FRCH6	FRCH5	FRCH4	FRCH3	FRCH2	FRCH1	FRCH0	
H'F773	FRCL	FRCL7	FRCL6	FRCL5	FRCL4	FRCL3	FRCL2	FRCL1	FRCL0	
H'F774	OCRAH/ OCRBH	OCRAH7/ OCRBH7	OCRAH6/ OCRBH6	OCRAH5/ OCRBH5	OCRAH4/ OCRBH4	OCRAH3/ OCRBH3	OCRAH2/ OCRBH2	OCRAH1/ OCRBH1	OCRAH0/O CRBH0	
H'F775	OCRAL/ OCRBL	OCRAL7/ OCRBL7	OCRAL6/ OCRBL6	OCRAL5/ OCRBL5	OCRAL4/ OCRBL4	OCRAL3/ OCRBL3	OCRAL2/ OCRBL2	OCRAL1/ OCRBL1	OCRAL0/ OCRBL0	
H'F776	TCRX	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0	
H'F777	TOCR	—	—	—	OCRS	OEA	OEB	OLVLA	OLVLB	
H'F778	ICRAH	ICRAH7	ICRAH6	ICRAH5	ICRAH4	ICRAH3	ICRAH2	ICRAH1	ICRAH0	
H'F779	ICRAL	ICRAL7	ICRAL6	ICRAL5	ICRAL4	ICRAL3	ICRAL2	ICRAL1	ICRAL0	
F'F77A	ICRBH	ICRBH7	ICRBH6	ICRBH5	ICRBH4	ICRBH3	ICRBH2	ICRBH1	ICRBH0	
F'F77B	ICRBL	ICRBL7	ICRBL6	ICRBL5	ICRBL4	ICRBL3	ICRBL2	ICRBL1	ICRBL0	
H'F77C	ICRCH	ICRCH7	ICRCH6	ICRCH5	ICRCH4	ICRCH3	ICRCH2	ICRCH1	ICRCH0	
H'F77D	ICRCL	ICRCL7	ICRCL6	ICRCL5	ICRCL4	ICRCL3	ICRCL2	ICRCL1	ICRCL0	
H'F77E	ICRDH	ICRDH7	ICRDH6	ICRDH5	ICRDH4	ICRDH3	ICRDH2	ICRDH1	ICRDH0	
H'F77F	ICRDL	ICRDL7	ICRDL6	ICRDL5	ICRDL4	ICRDL3	ICRDL2	ICRDL1	ICRDL0	
H'FF80	FLMCR	V_{PP}	—	—	—	EV	PV	E	P	Flash memory (flash memory version only)
H'FF81										
H'FF82	EBR1	—	—	—	—	LB3	LB2	LB1	LB0	
H'FF83	EBR2	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0	

Address	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFA0	SCR1	SNC1	SNC0	MRKON	LTCH	CKS3	CKS2	CKS1	CKS0	SCI1
H'FFA1	SCSR1	—	SOL	ORER	—	—	—	MTRF	STF	
H'FFA2	SDRU	SDRU7	SDRU6	SDRU5	SDRU4	SDRU3	SDRU2	SDRU1	SDRU0	
H'FFA3	SDRL	SDRL7	SDRL6	SDRL5	SDRL4	SDRL3	SDRL2	SDRL1	SDRL0	
H'FFA4										
H'FFA5										
H'FFA6										
H'FFA7										
H'FFA8	SMR	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0	SCI3
H'FFA9	BRR	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0	
H'FFAA	SCR3	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'FFAB	TDR	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0	
H'FFAC	SSR	TDRE	RDRF	OER	FER	PER	TEND	MPBR	MPBT	
H'FFAD	RDR	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0	
H'FFAE										
H'FFAF										
H'FFB0	TMA	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0	Timer A
H'FFB1	TCA	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0	
H'FFB2	TMB1	TMB17	—	—	—	—	TMB12	TMB11	TMB10	Timer B1
H'FFB3	TCB1/ TLB1	TCB17/ TLB17	TCB16/ TLB16	TCB15/ TLB15	TCB14/ TLB14	TCB13/ TLB13	TCB12/ TLB12	TCB11/ TLB11	TCB10/ TLB10	
H'FFB4										
H'FFB5										
H'FFB6										
H'FFB7										
H'FFB8	TCRV0	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	Timer V
H'FFB9	TCSRV	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	
H'FFBA	TCORA	TCORA7	TCORA6	TCORA5	TCORA4	TCORA3	TCORA2	TCORA1	TCORA0	
H'FFBB	TCORB	TCORB7	TCORB6	TCORB5	TCORB4	TCORB3	TCORB2	TCORB1	TCORB0	
H'FFBC	TCNTV	TCNTV7	TCNTV6	TCNTV5	TCNTV4	TCNTV3	TCNTV2	TCNTV1	TCNTV0	
H'FFBD	TCRV1	—	—	—	TVEG1	TVEG0	TRGE	—	ICKS0	

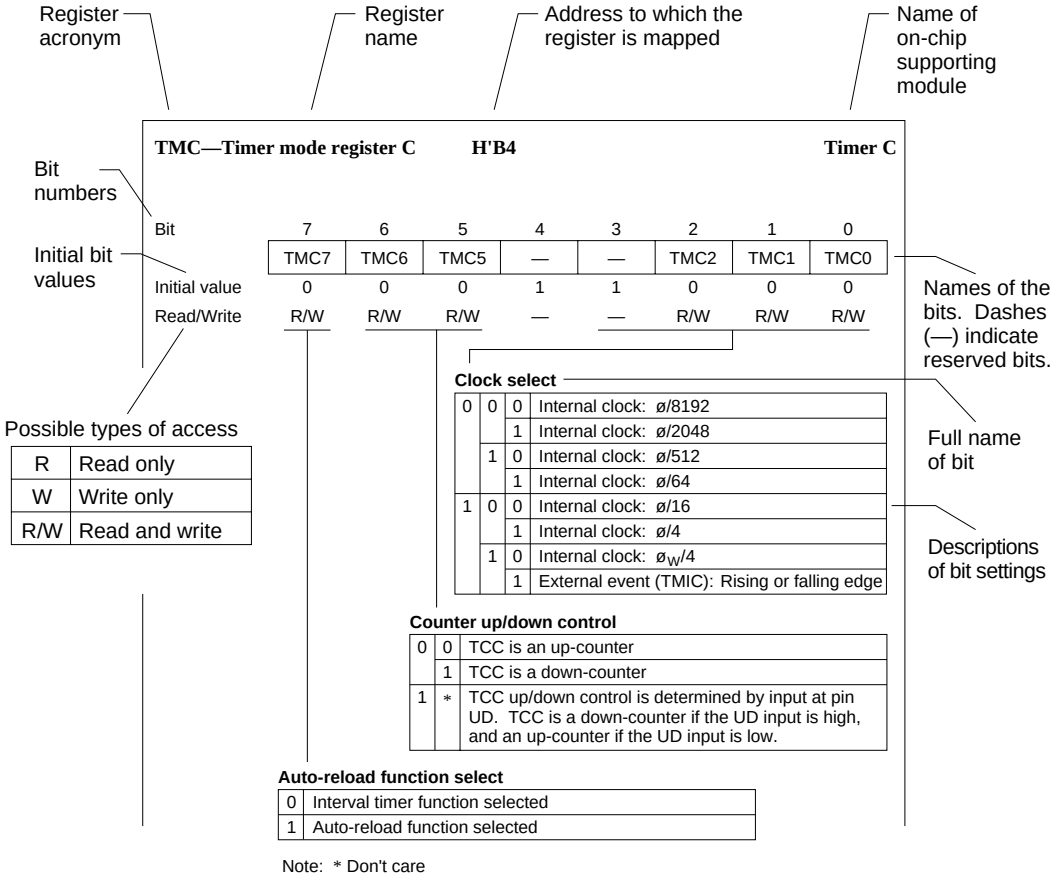
Legend:

SCI1: Serial communication interface 1

Address	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFBE	TCSRW	B6WI	TCWE	B4WI	TCSRWE	B2WI	WDON	BOWI	WRST	Watchdog timer
H'FFBF	TCW	TCW7	TCW6	TCW5	TCW4	TCW3	TCW2	TCW1	TCW0	
H'FFC0										
H'FFC1										
H'FFC2										
H'FFC3										
H'FFC4	AMR	CKS	TRGE	—	—	CH3	CH2	CH1	CH0	A/D converter
H'FFC5	ADRR	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	
H'FFC6	ADSR	ADSF	—	—	—	—	—	—	—	
H'FFC7										
H'FFC8										
H'FFC9										
H'FFCA										
H'FFCB										
H'FFCC										
H'FFCD										
H'FFCE										
H'FFCF										
H'FFD0	PWCR	—	—	—	—	—	—	—	$PWCR_0$	14-bit PWM
H'FFD1	PWDRU	—	—	$PWDRU_5$	$PWDRU_4$	$PWDRU_3$	$PWDRU_2$	$PWDRU_1$	$PWDRU_0$	
H'FFD2	PWDRL	$PWDRL_7$	$PWDRL_6$	$PWDRL_5$	$PWDRL_4$	$PWDRL_3$	$PWDRL_2$	$PWDRL_1$	$PWDRL_0$	
H'FFD3										
H'FFD4	PDR1	$P1_7$	$P1_6$	$P1_5$	$P1_4$	—	—	—	$P1_0$	I/O ports
H'FFD5	PDR2	—	—	—	—	—	$P2_2$	$P2_1$	$P2_0$	
H'FFD6	PDR3	—	—	—	—	—	$P3_2$	$P3_1$	$P3_0$	
H'FFD7										
H'FFD8	PDR5	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$	
H'FFD9	PDR6	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$	
H'FFDA	PDR7	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	—	—	—	
H'FFDB	PDR8	$P8_7$	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$	
H'FFDC	PDR9	—	—	—	$P9_4$	$P9_3$	$P9_2$	$P9_1$	$P9_0$	
H'FFDD	PDRB	PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0	
H'FFDE										
H'FFDF										

Address	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFE0										I/O ports
H'FFE1										
H'FFE2										
H'FFE3										
H'FFE4	PCR1	$PCR1_7$	$PCR1_6$	$PCR1_5$	$PCR1_4$	—	—	—	$PCR1_0$	
H'FFE5	PCR2	—	—	—	—	—	$PCR2_2$	$PCR2_1$	$PCR2_0$	I/O ports
H'FFE6	PCR3	—	—	—	—	—	$PCR3_2$	$PCR3_1$	$PCR3_0$	
H'FFE7										
H'FFE8	PCR5	$PCR5_7$	$PCR5_6$	$PCR5_5$	$PCR5_4$	$PCR5_3$	$PCR5_2$	$PCR5_1$	$PCR5_0$	
H'FFE9	PCR6	$PCR6_7$	$PCR6_6$	$PCR6_5$	$PCR6_4$	$PCR6_3$	$PCR6_2$	$PCR6_1$	$PCR6_0$	
H'FFEA	PCR7	$PCR7_7$	$PCR7_6$	$PCR7_5$	$PCR7_4$	$PCR7_3$	—	—	—	
H'FFEB	PCR8	$PCR8_7$	$PCR8_6$	$PCR8_5$	$PCR8_4$	$PCR8_3$	$PCR8_2$	$PCR8_1$	$PCR8_0$	
H'FFEC	PCR9	—	—	—	$PCR9_4$	$PCR9_3$	$PCR9_2$	$PCR9_1$	$PCR9_0$	
H'FFED	PUCR1	$PUCR1_7$	$PUCR1_6$	$PUCR1_5$	$PUCR1_4$	—	—	—	$PUCR1_0$	
H'FFEE	PUCR3	—	—	—	—	—	$PUCR3_2$	$PUCR3_1$	$PUCR3_0$	
H'FFEF	PUCR5	$PUCR5_7$	$PUCR5_6$	$PUCR5_5$	$PUCR5_4$	$PUCR5_3$	$PUCR5_2$	$PUCR5_1$	$PUCR5_0$	
H'FFF0	SYSCR1	SSBY	STS2	STS1	STS0	LSON	—	MA1	MA0	System control
H'FFF1	SYSCR2	—	—	—	NESEL	DTON	MSON	SA1	SA0	
H'FFF2	IEGR1	—	—	—	—	IEG3	IEG2	IEG1	IEG0	
H'FFF3	IEGR2	$INTEG_7$	$INTEG_6$	$INTEG_5$	$INTEG_4$	$INTEG_3$	$INTEG_2$	$INTEG_1$	$INTEG_0$	
H'FFF4	IENR1	IENTB1	IENTA	—	—	IEN3	IEN2	IEN1	IEN0	
H'FFF5	IENR2	IENDT	IENAD	—	IENSI	—	—	—	—	
H'FFF6	IENR3	INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0	
H'FFF7	IRR1	IRRTB1	IRRTA	—	—	IRRI3	IRRI2	IRRI1	IRRI0	
H'FFF8	IRR2	IRRDT	IRRAD	—	IRRS1	—	—	—	—	
H'FFF9	IRR3	$INTF_7$	$INTF_6$	$INTF_5$	$INTF_4$	$INTF_3$	$INTF_2$	$INTF_1$	$INTF_0$	
H'FFFA										
H'FFFB										
H'FFFC	PMR1	IRQ3	IRQ2	IRQ1	PWM	—	—	—	TMOW	I/O ports
H'FFFD	PMR3	—	—	—	—	—	SO1	SI1	SCK1	
H'FFFE										
H'FFFF	PMR7	—	—	—	—	—	TXD	—	POF1	I/O ports

B.2 Functions



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

Timer overflow interrupt enable

0	Interrupt request (FOVI) by OVF is disabled
1	Interrupt request (FOVI) by OVF is enabled

Output compare interrupt B enable

0	Interrupt request (OCIB) by OCFB is disabled
1	Interrupt request (OCIB) by OCFB is enabled

Output compare interrupt A enable

0	Interrupt request (OCIA) by OCFA is disabled
1	Interrupt request (OCIA) by OCFA is enabled

Input capture interrupt D enable

0	Interrupt request (ICID) by ICFD is disabled
1	Interrupt request (ICID) by ICFD is enabled

Input capture interrupt C enable

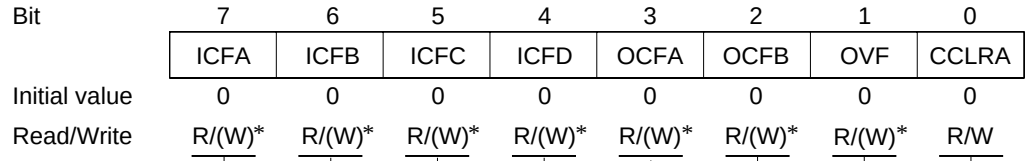
0	Interrupt request (ICIC) by ICFC is disabled
1	Interrupt request (ICIC) by ICFC is enabled

Input capture interrupt B enable

0	Interrupt request (ICIB) by ICFB is disabled
1	Interrupt request (ICIB) by ICFB is enabled

Input capture interrupt A enable

0	Interrupt request (ICIA) by ICFA is disabled
1	Interrupt request (ICIA) by ICFA is enabled



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

Counter clear A

0	FRC is not cleared by compare match A
1	FRC is cleared by compare match A

Timer overflow

0	[Clearing condition] After reading OVF = 1, cleared by writing 0 to OVF
1	[Setting condition] Set when the FRC value goes from H'FFFF to H'0000

Output compare flag B

0	[Clearing condition] After reading OCFB = 1, cleared by writing 0 to OCFB
1	[Setting condition] Set when FRC matches OCRB

Output compare flag A

0	[Clearing condition] After reading OCFA = 1, cleared by writing 0 to OCFA
1	[Setting condition] Set when FRC matches OCRA

Input capture flag D

0	[Clearing condition] After reading ICFD = 1, cleared by writing 0 to ICFD
1	[Setting condition] Set by input capture signal

Input capture flag C

0	[Clearing condition] After reading ICFC = 1, cleared by writing 0 to ICFC
1	[Setting condition] Set by input capture signal

Input capture flag B

0	[Clearing condition] After reading ICFB = 1, cleared by writing 0 to ICFB
1	[Setting condition] When the value of FRC is transferred to ICRB by the input capture signal

Input capture flag A

0	[Clearing condition] After reading ICFA = 1, cleared by writing 0 to ICFA
1	[Setting condition] When the value of FRC is transferred to ICRA by the input capture signal

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

FRCH—Free-running counter H **H'F772** **Timer X**

Bit	7	6	5	4	3	2	1	0
	FRCH7	FRCH6	FRCH5	FRCH4	FRCH3	FRCH2	FRCH1	FRCH0
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

FRCL—Free-running counter L **H'F773** **Timer X**

Bit	7	6	5	4	3	2	1	0
	FRCL7	FRCL6	FRCL5	FRCL4	FRCL3	FRCL2	FRCL1	FRCL0
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

OCRAH—Output compare register AH **H'F774** **Timer X**

Bit	7	6	5	4	3	2	1	0
	OCRAH7	OCRAH6	OCRAH5	OCRAH4	OCRAH3	OCRAH2	OCRAH1	OCRAH0
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

OCRBH—Output compare register BH **H'F774** **Timer X**

Bit	7	6	5	4	3	2	1	0
	OCRBH7	OCRBH6	OCRBH5	OCRBH4	OCRBH3	OCRBH2	OCRBH1	OCRBH0
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

OCRAL—Output compare register AL **H'F775** **Timer X**

Bit	7	6	5	4	3	2	1	0
	OCRAL7	OCRAL6	OCRAL5	OCRAL4	OCRAL3	OCRAL2	OCRAL1	OCRAL0
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

OCRBL—Output compare register BL **H'F775** **Timer X**

Bit	7	6	5	4	3	2	1	0
	OCRBL7	OCRBL6	OCRBL5	OCRBL4	OCRBL3	OCRBL2	OCRBL1	OCRBL0
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
	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock select

0	0	Internal clock: ø/2
	1	Internal clock: ø/8
1	0	Internal clock: ø/32
	1	Internal clock: rising edge

Buffer enable B

0	ICRD is not used as a buffer register for ICRB
1	ICRD is used as a buffer register for OCRB

Buffer enable A

0	ICRC is not used as a buffer register for ICRA
1	ICRC is used as a buffer register for OCRA

Input edge select D

0	Rising edge of input D is captured
1	Falling edge of input D is captured

Input edge select C

0	Rising edge of input C is captured
1	Falling edge of input C is captured

Input edge select B

0	Rising edge of input B is captured
1	Falling edge of input B is captured

Input edge select A

0	Rising edge of input A is captured
1	Falling edge of input A is captured

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

Output level B

0	Low level
1	High level

Output level A

0	Low level
1	High level

Output enable B

0	Output compare B output is disabled
1	Output compare B output is enabled

Output enable A

0	Output compare A output is disabled
1	Output compare A output is enabled

Output compare register select

0	OCRA is selected
1	OCRB is selected

ICRAH—Input capture register AH **H'F778** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRAH7	ICRAH6	ICRAH5	ICRAH4	ICRAH3	ICRAH2	ICRAH1	ICRAH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRAL—Input capture register AL **H'F779** **Timer X**

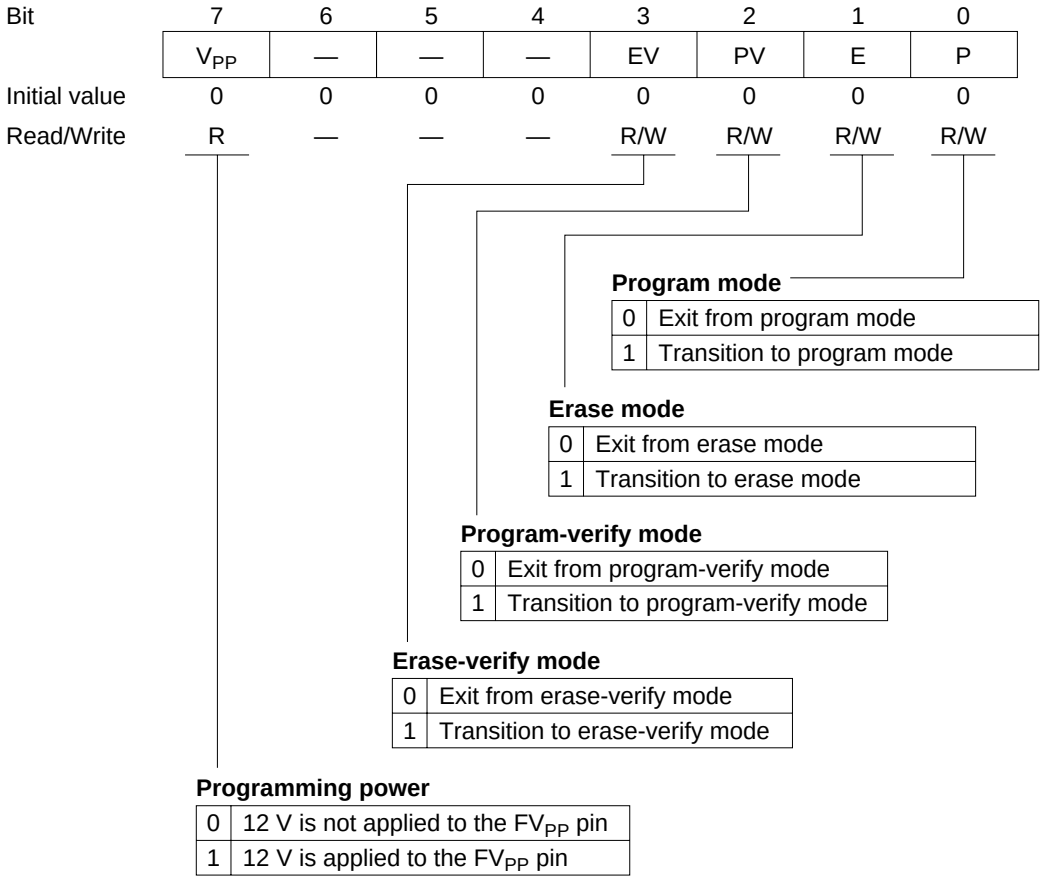
Bit	7	6	5	4	3	2	1	0
	ICRAL7	ICRAL6	ICRAL5	ICRAL4	ICRAL3	ICRAL2	ICRAL1	ICRAL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRBH—Input capture register BH **H'F77A** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRBH7	ICRBH6	ICRBH5	ICRBH4	ICRBH3	ICRBH2	ICRBH1	ICRBH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRBL—Input capture register BL **H'F77B** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRBL7	ICRBL6	ICRBL5	ICRBL4	ICRBL3	ICRBL2	ICRBL1	ICRBL0
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
V_{PP}
—
—
—
EV
PV
E
P
Initial value
0
0
0
0
0
0
0
0
Read/Write
R
—
—
—
R/W
R/W
R/W
R/W
Program mode
0 Exit from program mode
1 Transition to program mode
Erase mode
0 Exit from erase mode
1 Transition to erase mode
Program-verify mode
0 Exit from program-verify mode
1 Transition to program-verify mode
Erase-verify mode
0 Exit from erase-verify mode
1 Transition to erase-verify mode
Programming power
0 12 V is not applied to the FV_{PP} pin
1 12 V is applied to the FV_{PP} pin

EBR1—Erase block register 1 **H'FF82** **Flash memory (Flash memory version only)**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	LB3	LB2	LB1	LB0
Initial value	1	1	1	1	0	0	0	0
Read/Write	—	—	—	—	R/W	R/W	R/W	R/W

Large block 3 to 0

0	Not selected
1	Selected

EBR2—Erase block register 2 **H'FF83** **Flash memory (Flash memory version only)**

Bit	7	6	5	4	3	2	1	0
	SB7	SB6	SB5	SB4	SB3	SB2	SB1	SB0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Small block 7 to 0

0	Not selected
1	Selected

ICRCH—Input capture register CH **H'F77C** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRCH7	ICRCH6	ICRCH5	ICRCH4	ICRCH3	ICRCH2	ICRCH1	ICRCH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRCL—Input capture register CL **H'F77D** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRCL7	ICRCL6	ICRCL5	ICRCL4	ICRCL3	ICRCL2	ICRCL1	ICRCL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRDH—Input capture register DH **H'F77E** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRDH7	ICRDH6	ICRDH5	ICRDH4	ICRDH3	ICRDH2	ICRDH1	ICRDH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

ICRDL—Input capture register DL **H'F77F** **Timer X**

Bit	7	6	5	4	3	2	1	0
	ICRDL7	ICRDL6	ICRDL5	ICRDL4	ICRDL3	ICRDL2	ICRDL1	ICRDL0
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
	SNC1	SNC0	MRKON	LTCH	CKS3	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock select (CKS2 to CKS0)

Bit 2	Bit 1	Bit 0	Prescaler Division	Serial Clock Cycle: Synchronous ø = 5 MHz	Serial Clock Cycle: Synchronous ø = 2.5 MHz
CKS2	CKS1	CKS0			
0	0	0	ø/1024	204.8 µs	409.6 µs
		1	ø/256	51.2 µs	102.4 µs
	1	0	ø/64	12.8 µs	25.6 µs
		1	ø/32	6.4 µs	12.8 µs
1	0	0	ø/16	3.2 µs	6.4 µs
		1	ø/8	1.6 µs	3.2 µs
	1	0	ø/4	0.8 µs	1.6 µs
		1	ø/2	—	0.8 µs

Clock source select (CKS3)

0	Clock source is prescaler S, and pin SCK_1 is output pin
1	Clock source is external clock, and pin SCK_1 is input pin

LATCH TAIL select

0	HOLD TAIL is output
1	LATCH TAIL is output

TAIL MARK control

0	TAIL MARK is not output (synchronous mode)
1	TAIL MARK is output (SSB mode)

Operation mode select

0	0	8-bit synchronous transfer mode
	1	16-bit synchronous transfer mode
1	0	Continuous clock output mode
	1	Reserved

Bit	7	6	5	4	3	2	1	0
	—	SOL	ORER	—	—	—	MTRF	STF
Initial value	1	0	0	1	1	1	0	0
Read/Write	—	R/W	R/(W)*	—	—	—	R	R/W

Start flag

0	Read	Indicates that transfer is stopped
	Write	Invalid
1	Read	Indicates transfer in progress
	Write	Starts a transfer operation

TAIL MARK transmit flag

0	Idle state and 8- or -16-bit data transfer in progress
1	TAIL MARK transmission in progress

Overrun error flag

0	[Clearing condition] After reading 1, cleared by writing 0
1	[Setting condition] Set if a clock pulse is input after transfer is complete, when an external clock is used

Extended data bit

0	Read	SO_1 pin output level is low
	Write	SO_1 pin output level changes to low
1	Read	SO_1 pin output level is high
	Write	SO_1 pin output level changes to high

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

SDRU—Serial data register U **H'FFA2** **SCI1**

Bit	7	6	5	4	3	2	1	0
	SDRU7	SDRU6	SDRU5	SDRU4	SDRU3	SDRU2	SDRU1	SDRU0
Initial value	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Stores transmit and receive data

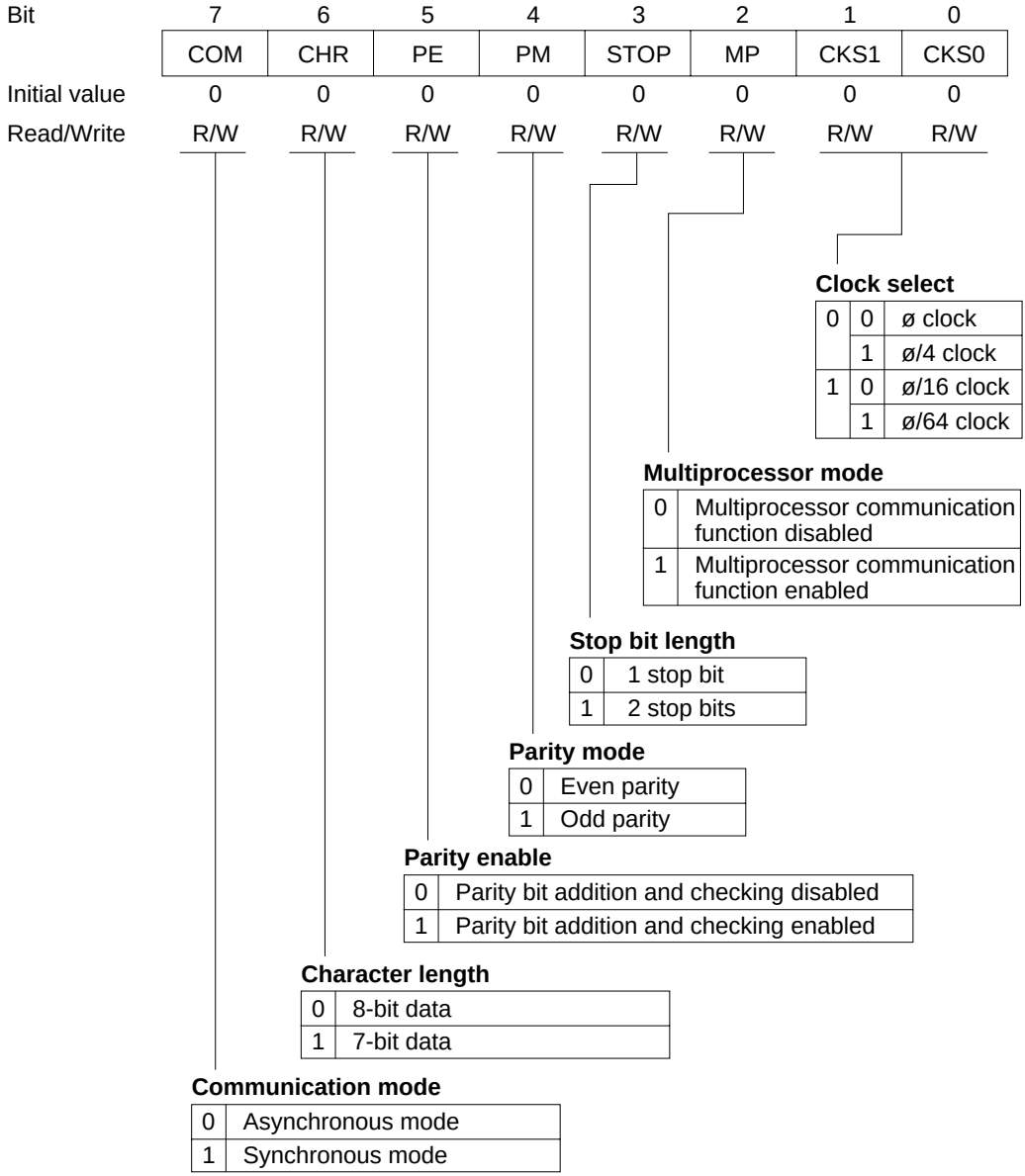
8-bit transfer mode: Not used
16-bit transfer mode: Upper 8 bits of data

SDRL—Serial data register L **H'FFA3** **SCI1**

Bit	7	6	5	4	3	2	1	0
	SDRL7	SDRL6	SDRL5	SDRL4	SDRL3	SDRL2	SDRL1	SDRL0
Initial value	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed	Not fixed
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Stores transmit and receive data

8-bit transfer mode: 8-bit data
16-bit transfer mode: Lower 8 bits of data


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
1 ø/4 clock
1 0 ø/16 clock
1 ø/64 clock
Multiprocessor mode
0 Multiprocessor communication function disabled
1 Multiprocessor communication function enabled
Stop bit length
0 1 stop bit
1 2 stop bits
Parity mode
0 Even parity
1 Odd parity
Parity enable
0 Parity bit addition and checking disabled
1 Parity bit addition and checking enabled
Character length
0 8-bit data
1 7-bit data
Communication mode
0 Asynchronous mode
1 Synchronous mode

Bit	7	6	5	4	3	2	1	0
	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock enable

Bit 1	Bit 0	Description		
CKE1	CKE0	Communication Mode	Clock Source	SCK_3 Pin Function
0	0	Asynchronous	Internal clock	I/O port
		Synchronous	Internal clock	Serial clock output
	1	Asynchronous	Internal clock	Clock output
		Synchronous	Reserved (Do not specify this combination)	
1	0	Asynchronous	External clock	Clock input
		Synchronous	External clock	Serial clock input
	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
1	Transmit operation enabled (TXD pin is transmit data pin)*1

Note: 1. When bit TXD is set to 1 in PMR7

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

0	A 0 multiprocessor bit is transmitted
1	A 1 multiprocessor bit is transmitted

Multiprocessor bit receive

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

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 register 3 (SCR3) is cleared to 0 • When bit TDRE is set to 1 when the last bit of a transmit character is sent

Parity error

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

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

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

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

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 register 3 (SCR3) is cleared to 0 • When data is transferred from TDR to TSR

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

Bit	7	6	5	4	3	2	1	0
	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TMA—Timer mode register A H'FFB0 Timer A

Bit	7	6	5	4	3	2	1	0
	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W

Clock output select

0	0	0	ø/32
		1	ø/16
	1	0	ø/8
		1	ø/4
1	0	0	$ø_W$/32
		1	$ø_W$/16
	1	0	$ø_W$/8
		1	$ø_W$/4

Internal clock select

TMA3	TMA2	TMA1	TMA0	Prescaler and Divider Ratio or Overflow Period		Function
0	0	0	0	PSS	ø/8192	Interval timer
			1	PSS	ø/4096	
		1	0	PSS	ø/2048	
			1	PSS	ø/512	
	1	0	0	PSS	ø/256	
			1	PSS	ø/128	
		1	0	PSS	ø/32	
			1	PSS	ø/8	
1	0	0	0	PSW	1 s	Time base
			1	PSW	0.5 s	
		1	0	PSW	0.25 s	
			1	PSW	0.03125 s	
	1	0	0	PSW and TCA are reset		
			1			
		1	0			
			1			

TCA—Timer counter A **H'FFB1** **Timer A**

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

TMB1—Timer mode register B1 **H'FFB2** **Timer B1**

Bit	7	6	5	4	3	2	1	0
	TMB17	—	—	—	—	TMB12	TMB11	TMB10
Initial value	0	1	1	1	1	0	0	0
Read/Write	R/W	—	—	—	—	R/W	R/W	R/W

Auto-reload function select

0	Interval timer function selected
1	Auto-reload function selected

Clock select

0	0	0	Internal clock: ø/8192
		1	Internal clock: ø/2048
	1	0	Internal clock: ø/512
		1	Internal clock: ø/256
1	0	0	Internal clock: ø/64
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	External event (TMIB): Rising or falling edge

TCB1—Timer counter B1 **H'FFB3** **Timer B1**

Bit	7	6	5	4	3	2	1	0
	TCB17	TCB16	TCB15	TCB14	TCB13	TCB12	TCB11	TCB10
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Count value

TLB1—Timer load register B1 **H'FFB3** **Timer B1**

Bit	7	6	5	4	3	2	1	0
	TLB17	TLB16	TLB15	TLB14	TLB13	TLB12	TLB11	TLB10
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Reload value

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

Clock select

TCRV0			TCRV1	Description
Bit 2	Bit 1	Bit 0	Bit 0	
CKS2	CKS1	CKS0	ICKS0	
0	0	0	—	Clock input disabled
		1	0	Internal clock: ø/4, falling edge
			1	Internal clock: ø/8, falling edge
	1	0	0	Internal clock: ø/16, falling edge
			1	Internal clock: ø/32, falling edge
		1	0	Internal clock: ø/64, falling edge
			1	Internal clock: ø/128, falling edge
1	0	0	—	Clock input disabled
		1	—	External clock: rising edge
	1	0	—	External clock: falling edge
		1	—	External clock: rising and falling edges

Counter clear 1 and 0

0	Clearing is disabled
	Cleared by compare match A
1	Cleared by compare match B
	Cleared by rising edge of external reset input

Timer overflow interrupt enable

0	Interrupt request (OVI) from OVF disabled
1	Interrupt request (OVI) from OVF enabled

Compare match interrupt enable A

0	Interrupt request (CMIA) from CMFA disabled
1	Interrupt request (CMIA) from CMFA enabled

Compare match interrupt enable B

0	Interrupt request (CMIB) from CMFB disabled
1	Interrupt request (CMIB) from CMFB enabled

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

Output select

0	0	No change at compare match A
	1	0 output at compare match A
1	0	1 output at compare match A
	1	Output toggles at compare match A

Output select

0	0	No change at compare match B
	1	0 output at compare match B
1	0	1 output at compare match B
	1	Output toggles at compare match B

Timer overflow flag

0	[Clearing condition] After reading OVF = 1, cleared by writing 0 to OVF
1	[Setting condition] Set when TCNTV overflows from H'FF to H'00

Compare match flag A

0	[Clearing condition] After reading CMFA = 1, cleared by writing 0 to CMFA
1	[Setting condition] Set when the TCNTV value matches the TCORA value

Compare match flag B

0	[Clearing condition] After reading CMFB = 1, cleared by writing 0 to CMFB
1	[Setting condition] Set when the TCNTV value matches the TCORB value

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

TCORA—Time constant register A **H'FFBA** **Timer V**

Bit	7	6	5	4	3	2	1	0
	TCORA7	TCORA6	TCORA5	TCORA4	TCORA3	TCORA2	TCORA1	TCORA0
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

TCORB—Time constant register B **H'FFBB** **Timer V**

Bit	7	6	5	4	3	2	1	0
	TCORB7	TCORB6	TCORB5	TCORB4	TCORB3	TCORB2	TCORB1	TCORB0
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

TCNTV—Timer counter V **H'FFBC** **Timer V**

Bit	7	6	5	4	3	2	1	0
	TCNTV7	TCNTV6	TCNTV5	TCNTV4	TCNTV3	TCNTV2	TCNTV1	TCNTV0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
	—	—	—	TVEG1	TVEG0	TRGE	—	ICKS0
Initial value	1	1	1	0	0	0	1	0
Read/Write	—	—	—	R/W	R/W	R/W	—	R/W

Internal clock select

Selects the TCNTV clock source, with bits CKS2 to CKS0 in TCRV0

TRGV input enable

0	TCNTV counting is not triggered by input at the TRGV pin, and does not stop when TCNTV is cleared by compare match
1	TCNTV counting is triggered by input at the TRGV pin, and stops when TCNTV is cleared by compare match

TRGV input edge select

0	0	TRGV trigger input is disabled
	1	Rising edge is selected
1	0	Falling edge is selected
	1	Rising and falling edges are both selected

Bit	7	6	5	4	3	2	1	0
	B6WI	TCWE	B4WI	TCSRWE	B2WI	WDON	B0WI	WRST
Initial value	1	0	1	0	1	0	1	0
Read/Write	R	R/(W)*	R	R/(W)*	R	R/(W)*	R	R/(W)*

Watchdog timer reset

0	[Clearing conditions] • Reset by $\overline{\text{RES}}$ pin • When TCSRWE = 1, and 0 is written in both B0WI and WRST
1	[Setting condition] When TCW overflows and a reset signal is generated

Bit 0 write inhibit

0	Bit 0 is write-enabled
1	Bit 0 is write-protected

Watchdog timer on

0	Watchdog timer operation is disabled
1	Watchdog timer operation is enabled

Bit 2 write inhibit

0	Bit 2 is write-enabled
1	Bit 2 is write-protected

Timer control/status register W write enable

0	Data cannot be written to TCSRW bits 2 and 0
1	Data can be written to TCSRW bits 2 and 0

Bit 4 write inhibit

0	Bit 4 is write-enabled
1	Bit 4 is write-protected

Timer counter W write enable

0	Data cannot be written to TCW
1	Data can be written to TCW

Bit 6 write inhibit

0	Bit 6 is write-enabled
1	Bit 6 is write-protected

Note: * Write is permitted only under certain conditions.

Bit	7	6	5	4	3	2	1	0
	TCW7	TCW6	TCW5	TCW4	TCW3	TCW2	TCW1	TCW0
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

Bit	7	6	5	4	3	2	1	0
	CKS	TRGE	—	—	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	AN_4
			1	AN_5
		1	0	AN_6
			1	AN_7
	1	0	0	Reserved
		0	1	Reserved
		1	0	Reserved
		1	1	Reserved

External trigger select

0	Disables start of A/D conversion by external trigger
1	Enables start of A/D conversion by rising or falling edge of external trigger at pin $\overline{\text{ADTRG}}$

Clock select

Bit 7		Conversion Time	
CKS	Conversion Period	ø = 2 MHz	ø = 5 MHz
0	62/ø	31 µs	12.4 µs
1	31/ø	15.5 µs	—*1

Notes: * Don't care
1. 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.

ADRR—A/D result register **H'FFC5** **A/D converter**

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

A/D conversion result

ADSR—A/D start register **H'FFC6** **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

PWCR—PWM control register **H'FFD0** **14-bit PWM**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	PWCR0
Initial value	1	1	1	1	1	1	1	0
Read/Write	—	—	—	—	—	—	—	W

Clock select

0	The input clock is ø/2 (tø* = 2/ø). The conversion period is 16,384/ø, with a minimum modulation width of 1/ø.
1	The input clock is ø/4 (tø* = 4/ø). The conversion period is 32,768/ø, with a minimum modulation width of 2/ø.

Note: *tø: Period of PWM input clock

PWDRU—PWM data register U **H'FFD1** **14-bit PWM**

Bit	7	6	5	4	3	2	1	0
	—	—	PWDRU5	PWDRU4	PWDRU3	PWDRU2	PWDUR1	PWDRU0
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	W	W	W	W	W	W

Upper 6 bits of data for generating PWM waveform

PWDRL—PWM data register L **H'FFD2** **14-bit PWM**

Bit	7	6	5	4	3	2	1	0
	PWDRL7	PWDRL6	PWDRL5	PWDRL4	PWDRL3	PWDRL2	PWDRL1	PWDRL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Lower 8 bits of data for generating PWM waveform

PDR1—Port data register 1 **H'FFD4** **I/O ports**

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

PDR2—Port data register 2 **H'FFD5** **I/O ports**

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

PDR3—Port data register 3 **H'FFD6** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$P3_2$	$P3_1$	$P3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

PDR5—Port data register 5 **H'FFD8** **I/O ports**

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

PDR6—Port data register 6 **H'FFD9** **I/O ports**

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

PDR7—Port data register 7 **H'FFDA** **I/O ports**

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

PDR8—Port data register 8 **H'FFDB** **I/O ports**

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

PDR9—Port data register 9 **H'FFDC** **I/O ports**

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

PDRB—Port data register B **H'FFDD** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0
Initial value								
Read/Write	R	R	R	R	R	R	R	R

PCR1—Port control register 1 **H'FFE4** **I/O ports**

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

Port 1 input/output select

0	Input pin
1	Output pin

PCR2—Port control register 2 **H'FFE5** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PCR2_2$	$PCR2_1$	$PCR2_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	W	W	W

Port 2 input/output select

0	Input pin
1	Output pin

PCR3—Port control register 3 **H'FFE6** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PCR3_2$	$PCR3_1$	$PCR3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	W	W	W

Port 3 input/output select

0	Input pin
1	Output pin

PCR5—Port control register 5 **H'FFE8** **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'FFE9** **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'FFEA** **I/O ports**

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

Port 7 input/output select

0	Input pin
1	Output pin

PCR8—Port control register 8 **H'FFEB** **I/O ports**

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

Port 8 input/output select

0	Input pin
1	Output pin

PCR9—Port control register 9 **H'FFEC** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	$PCR9_4$	$PCR9_3$	$PCR9_2$	$PCR9_1$	$PCR9_0$
Initial value	1	1	0	0	0	0	0	0
Read/Write	—	—	—	W	W	W	W	W

Port 9 input/output select

0	Input pin
1	Output pin

PUCR1—Port pull-up control register 1 **H'FFED** **I/O ports**

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

PUCR3—Port pull-up control register 3 **H'FFEE** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	—	$PUCR3_2$	$PUCR3_1$	$PUCR3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	—	—	—	—	—	R/W	R/W	R/W

PUCR5—Port pull-up control register 5 **H'FFEF** **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

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 = 32,768 states
		1	Wait time = 65,536 states
1	*	*	Wait time = 131,072 states

Software standby

0	• When a SLEEP instruction is executed in active mode, a transition is made to sleep mode • When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode
1	• When a SLEEP instruction is executed in active mode, a transition is made to standby mode or watch mode • When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode

Note: * Don't care

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

Subactive mode clock select

0	0	$ø_W/8$
	1	$ø_W/4$
1	*	$ø_W/2$

Medium speed on flag

0	• Operates in active (high-speed) mode after exit from standby, watch, or sleep mode • Operates in sleep (high-speed) mode if a SLEEP instruction is executed in active mode
1	• Operates in active (medium-speed) mode after exit from standby, watch, or sleep mode • Operates in sleep (medium-speed) mode if a SLEEP instruction is executed in active 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 $ø_{OSC}/4$

Note: * Don’t care

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

IRQ_0 edge select

0	Falling edge of $\overline{IRQ_0}$ pin input is detected
1	Rising edge of $\overline{IRQ_0}$ pin input is detected

IRQ_1 edge select

0	Falling edge of $\overline{IRQ_1}$ pin input is detected
1	Rising edge of $\overline{IRQ_1}$ pin input is detected

IRQ_2 edge select

0	Falling edge of $\overline{IRQ_2}$ pin input is detected
1	Rising edge of $\overline{IRQ_2}$ pin input is detected

IRQ_3 edge select

0	Falling edge of $\overline{IRQ_3}$ pin input is detected
1	Rising edge of $\overline{IRQ_3}$ pin input is detected

Bit	7	6	5	4	3	2	1	0
	INTEG7	INTEG6	INTEG5	INTEG4	INTEG3	INTEG2	INTEG1	INTEG0
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

INT_4 to INT_0 edge select

0	Falling edge of $\overline{INT}_n$ pin input is detected
1	Rising edge of $\overline{INT}_n$ pin input is detected

(n = 4 to 0)

INT_5 edge select

0	Falling edge of $\overline{INT}_5$ and $\overline{ADTRG}$ pin input is detected
1	Rising edge of $\overline{INT}_5$ and $\overline{ADTRG}$ pin input is detected

INT_6 edge select

0	Falling edge of $\overline{INT}_6$ and TMIB pin input is detected
1	Rising edge of $\overline{INT}_6$ and TMIB pin input is detected

INT_7 edge select

0	Falling edge of $\overline{INT}_7$ and TMIY pin input is detected
1	Rising edge of $\overline{INT}_7$ and TMIY pin input is detected

Bit	7	6	5	4	3	2	1	0
	IENTB1	IENTA	—	—	IEN3	IEN2	IEN1	IEN0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/W	R/W	—	—	R/W	R/W	R/W	R/W

IRQ_3 to IRQ_0 interrupt enable

0	Disables IRQ3 to IRQ_0 interrupt requests
1	Enables IRQ3 to IRQ_0 interrupt requests

Timer A interrupt enable

0	Disables timer A interrupt requests
1	Enables timer A interrupt requests

Timer B1 interrupt enable

0	Disables timer B1 interrupt requests
1	Enables timer B1 interrupt requests

IENR2—Interrupt enable register 2 **H'FFF5** **System control**

Bit	7	6	5	4	3	2	1	0
	IENDT	IENAD	—	IENS1	—	—	—	—
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	—	R/W	—	—	—	—

SCI1 interrupt enable

0	Disables SCI1 interrupt requests
1	Enables SCI1 interrupt requests

A/D converter interrupt enable

0	Disables A/D converter interrupt requests
1	Enables A/D converter interrupt requests

Direct transfer interrupt enable

0	Disables direct transfer interrupt requests
1	Enables direct transfer interrupt requests

IENR3—Interrupt enable register 3 **H'FFF6** **System control**

Bit	7	6	5	4	3	2	1	0
	INTEN7	INTEN6	INTEN5	INTEN4	INTEN3	INTEN2	INTEN1	INTEN0
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

INT_7 to INT_0 interrupt enable

0	Disables INT_7 to INT_0 interrupt requests
1	Enables INT_7 to INT_0 interrupt requests

Bit	7	6	5	4	3	2	1	0
	IRRTB1	IRRTA	—	—	IRRI3	IRRI2	IRRI1	IRRI0
Initial value	0	0	0	1	0	0	0	0
Read/Write	R/W*	R/W*	—	—	R/W*	R/W*	R/W*	R/W*

IRQ_3 to IRQ_0 interrupt request flag

0	[Clearing condition] When IRRIn = 1, it is cleared by writing 0
1	[Setting condition] When pin $\overline{IRQn}$ is set for interrupt input and the designated signal edge is input

(n = 3 to 0)

Timer A interrupt request flag

0	[Clearing condition] When IRRTA = 1, it is cleared by writing 0
1	[Setting condition] When timer counter A overflows from H'FF to H'00

Timer B1 interrupt request flag

0	[Clearing condition] When IRRTB1 = 1, it is cleared by writing 0
1	[Setting condition] When timer counter B1 overflows from H'FF to H'00

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

Bit	7	6	5	4	3	2	1	0
	IRRDT	IRRAD	—	IRRS1	—	—	—	—
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/W*	—	R/W*	—	—	—	—

SCI1 interrupt request flag

0	[Clearing condition] When IRRS1 = 1, it is cleared by writing 0
1	[Setting condition] When an SCI1 transfer is completed

A/D converter interrupt request flag

0	[Clearing condition] When IRRAD = 1, it is cleared by writing 0
1	[Setting condition] When A/D conversion is completed and ADSF is cleared to 0 in ADSR

Direct transfer interrupt request flag

0	[Clearing condition] When IRRDT = 1, it is cleared by writing 0
1	[Setting condition] A SLEEP instruction is executed when DTON = 1 and a direct transfer is made

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

Bit	7	6	5	4	3	2	1	0
	INTF7	INTF6	INTF5	INTF4	INTF3	INTF2	INTF1	INTF0
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*

INT_7 to INT_0 interrupt request flag

0	[Clearing condition] When $\overline{INTF}_n$ = 1, it is cleared by writing 0
1	[Setting condition] When the designated signal edge is input at pin $\overline{INT}_n$

(n = 7 to 0)

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

Bit	7	6	5	4	3	2	1	0
	IRQ3	IRQ2	IRQ1	PWM	—	—	—	TMOW
Initial value	0	0	0	0	0	1	0	0
Read/Write	R/W	R/W	R/W	R/W	—	—	—	R/W

$P1_0$/TMOW pin function switch

0	Functions as $P1_0$ I/O pin
1	Functions as TMOW output pin

$P1_4$/PWM pin function switch

0	Functions as $P1_4$ I/O pin
1	Functions as PWM output pin

$P1_5/\overline{IRQ_1}$ pin function switch

0	Functions as $P1_5$ I/O pin
1	Functions as $\overline{IRQ_1}$ input pin

$P1_6/\overline{IRQ_2}$ pin function switch

0	Functions as $P1_6$ I/O pin
1	Functions as $\overline{IRQ_2}$ input pin

$P1_7/\overline{IRQ_3}$ pin function switch

0	Functions as $P1_7$ I/O pin
1	Functions as $\overline{IRQ_3}$/TRGV input pin

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

$P3_0/SCK_1$ pin function switch

0	Functions as $P3_0$ I/O pin
1	Functions as SCK_1 I/O pin

$P3_1/SI_1$ pin function switch

0	Functions as $P3_1$ I/O pin
1	Functions as SI_1 input pin

$P3_2/SO_1$ pin function switch

0	Functions as $P3_2$ I/O pin
1	Functions as SO_1 output pin

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

$P3_2/SO_1$ pin PMOS control

0	CMOS output
1	NMOS open-drain output

$P2_2$/TXD pin function switch (TXD)

0	Functions as $P2_2$ I/O pin
1	Functions as TXD output pin

Appendix C I/O Port Block Diagrams

C.1 Block Diagrams of Port 1

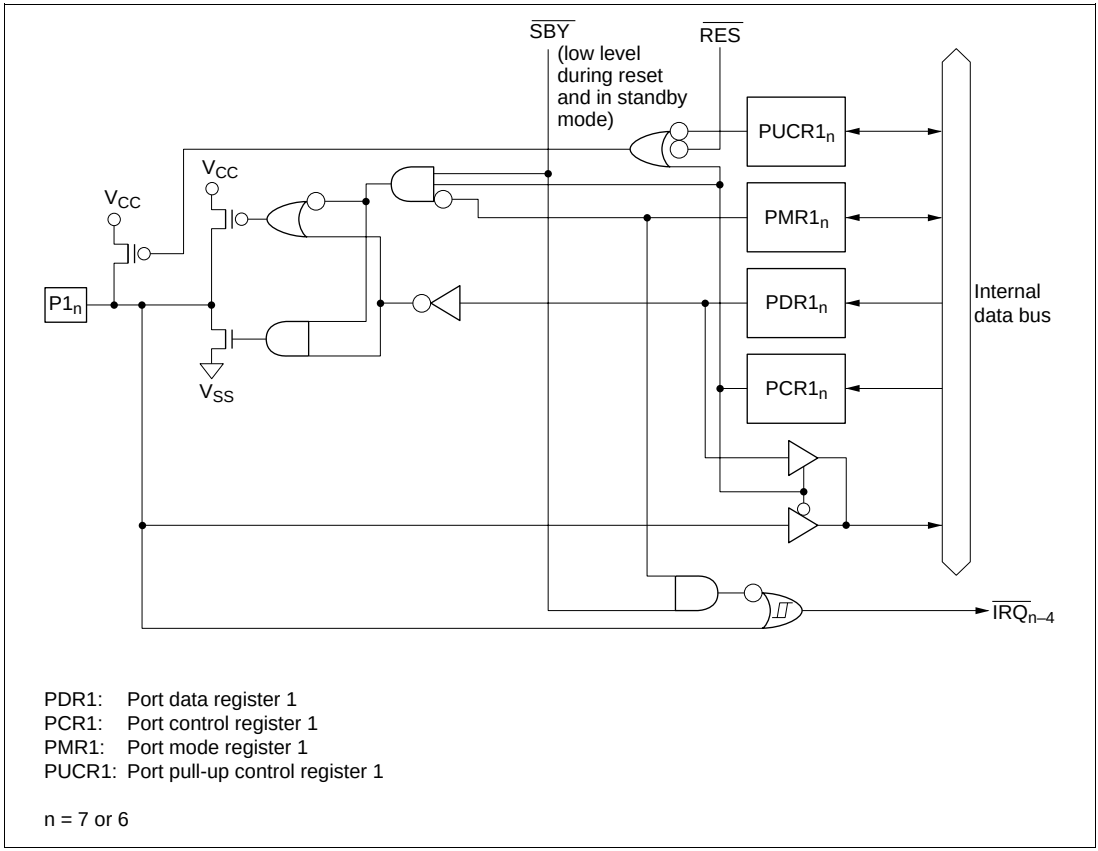
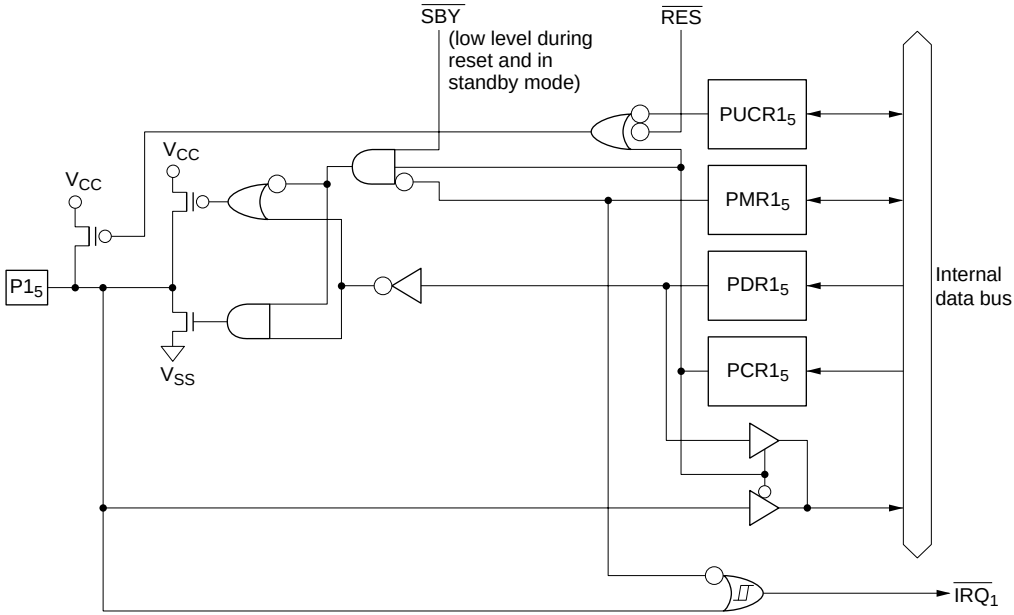


Figure C.1 (a) Port 1 Block Diagram (Pins $P1_7$ and $P1_6$)



PDR1: Port data register 1
PCR1: Port control register 1
PMR1: Port mode register 1
PUCR1: Port pull-up control register 1

Figure C.1 (b) Port 1 Block Diagram (Pin $P1_5$)

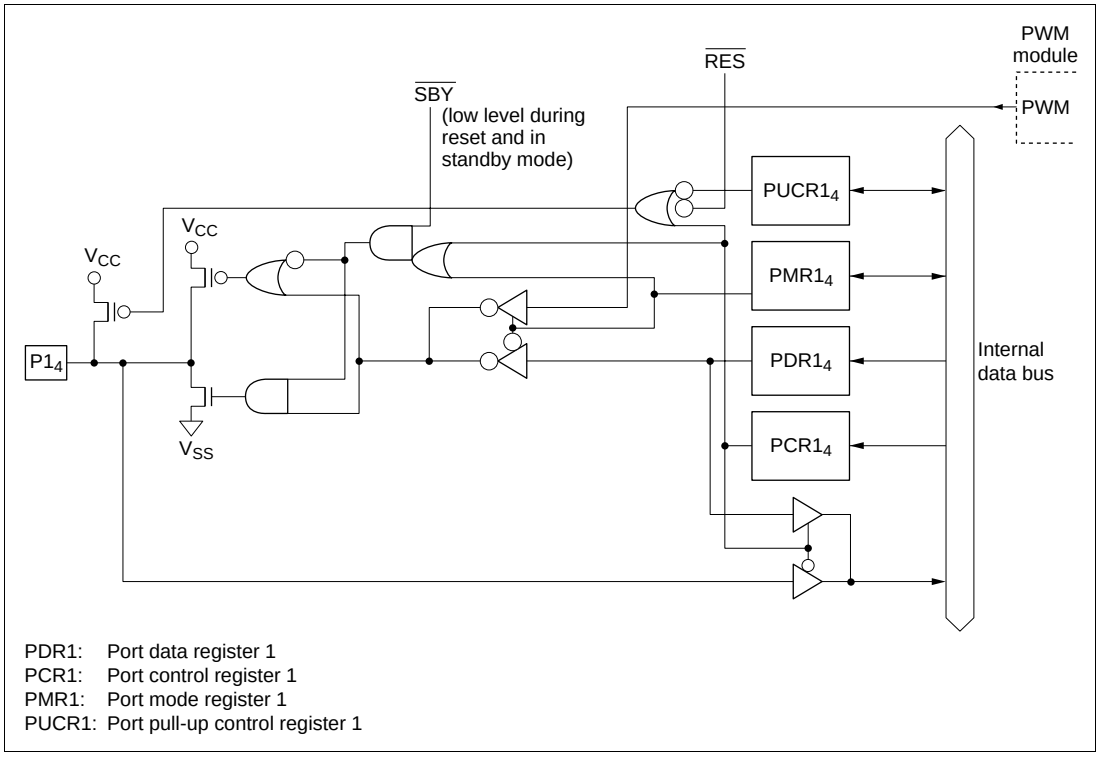


Figure C.1 (c) Port 1 Block Diagram (Pin $P1_4$)

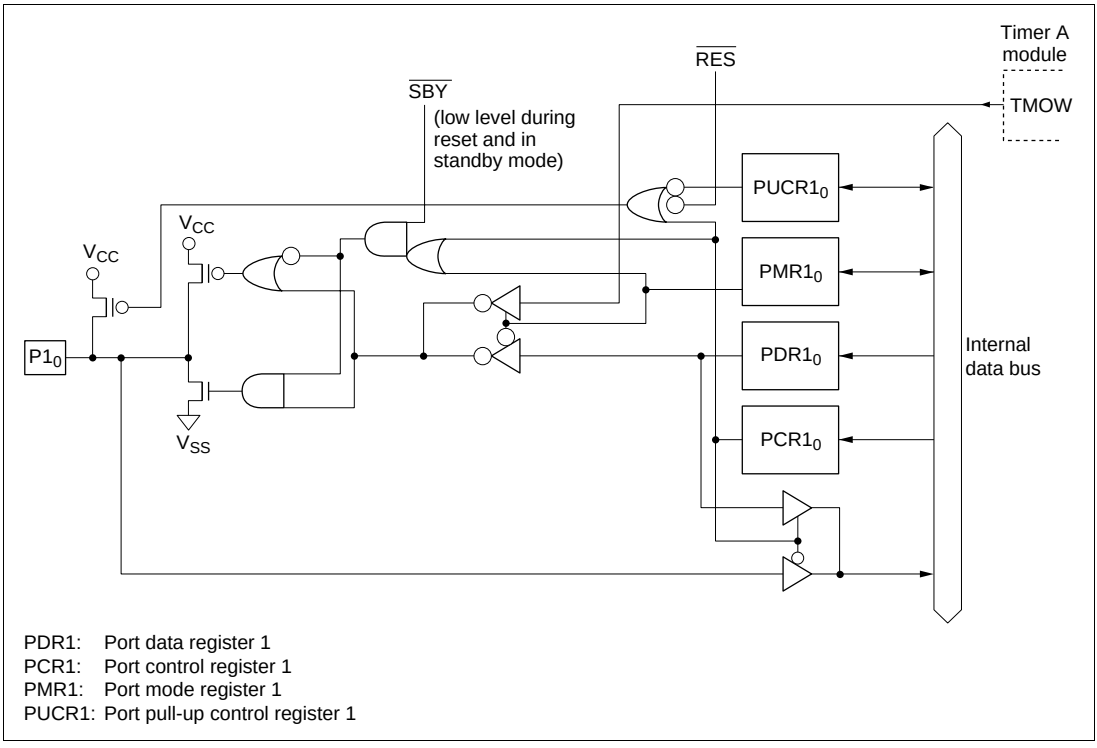


Figure C.1 (d) Port 1 Block Diagram (Pin $P1_0$)

C.2 Block Diagrams of Port 2

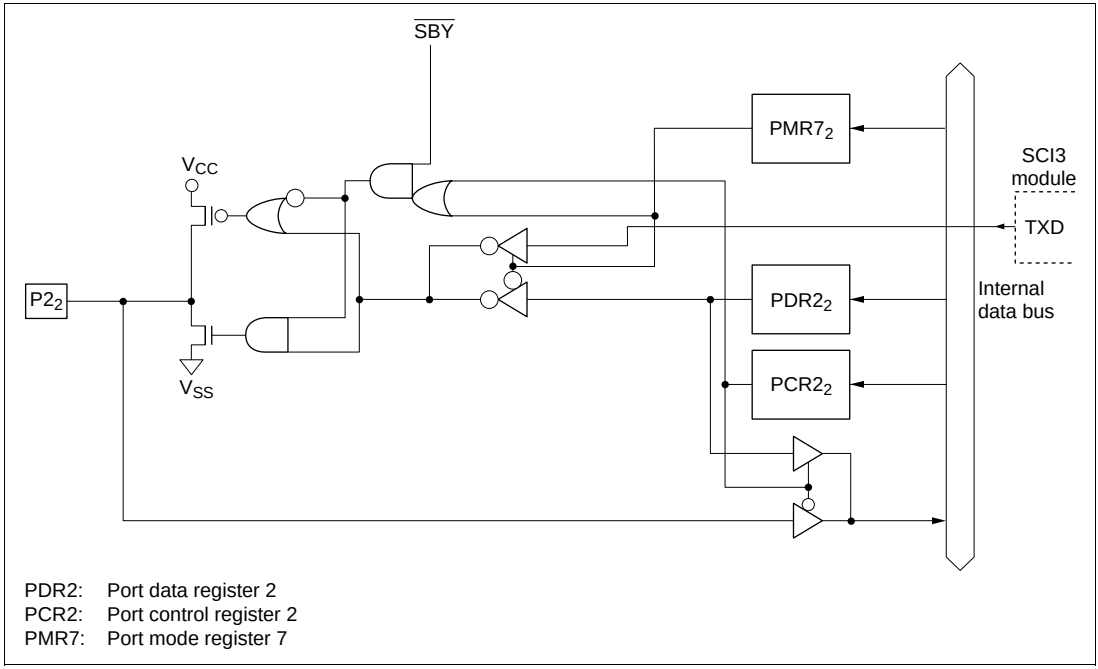
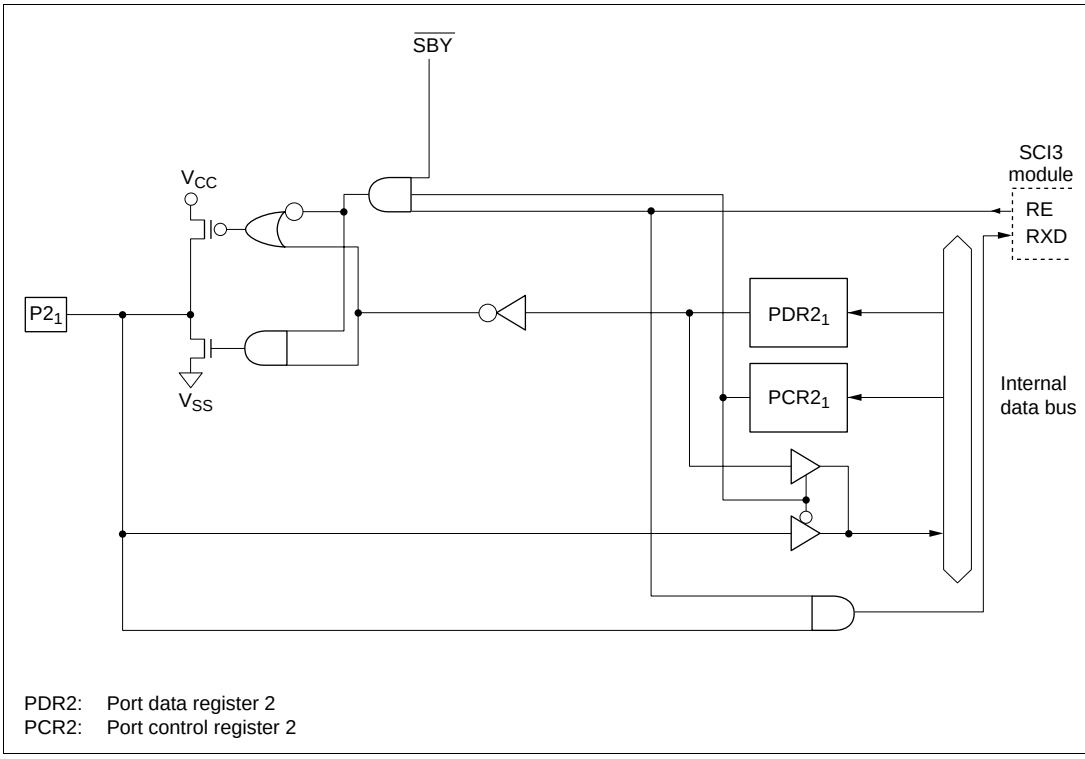
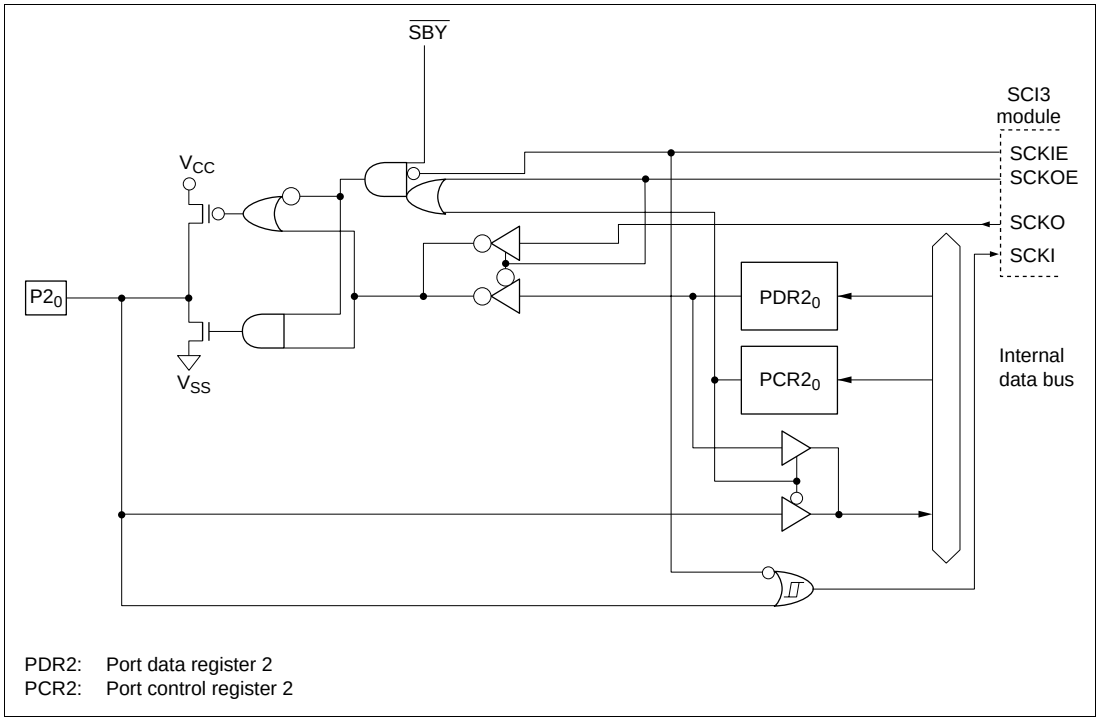


Figure C.2 (a) Port 2 Block Diagram (Pin $P2_2$)



PDR2: Port data register 2
PCR2: Port control register 2

Figure C.2 (b) Port 2 Block Diagram (Pin $P2_1$)



PDR2: Port data register 2
PCR2: Port control register 2

Figure C.2 (c) Port 2 Block Diagram (Pin $P2_0$)

C.3 Block Diagrams of Port 3

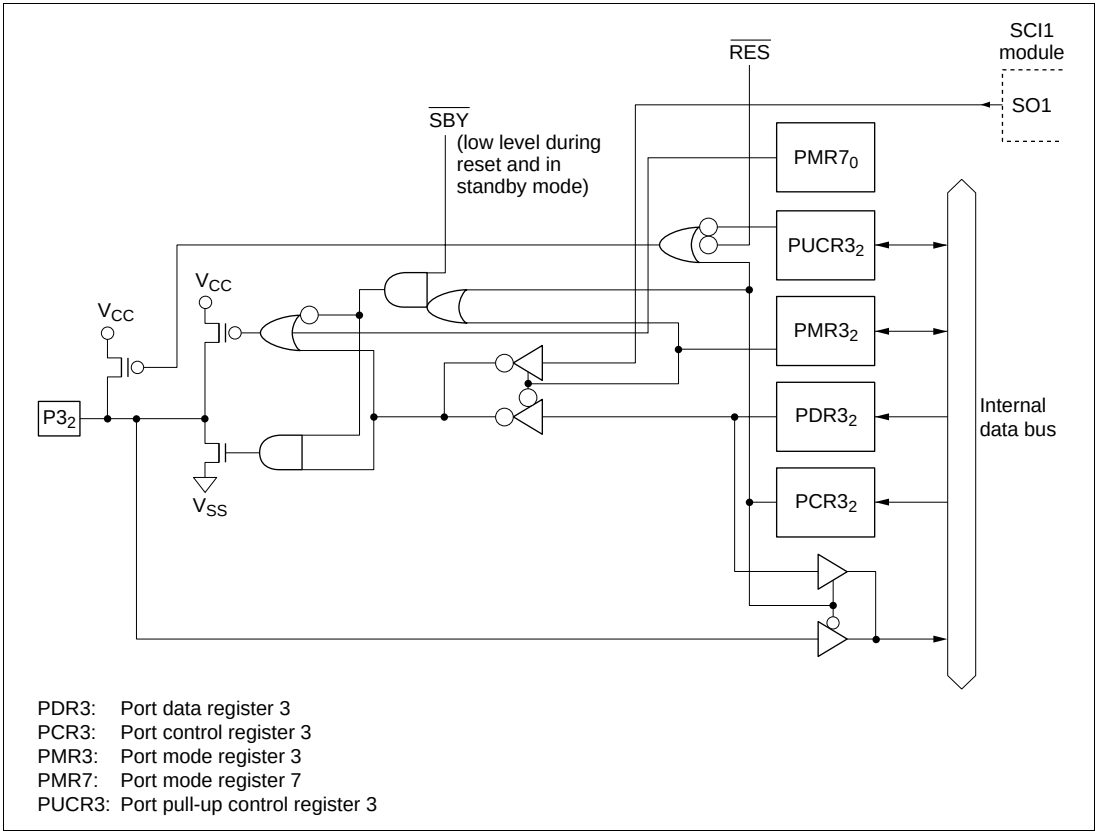
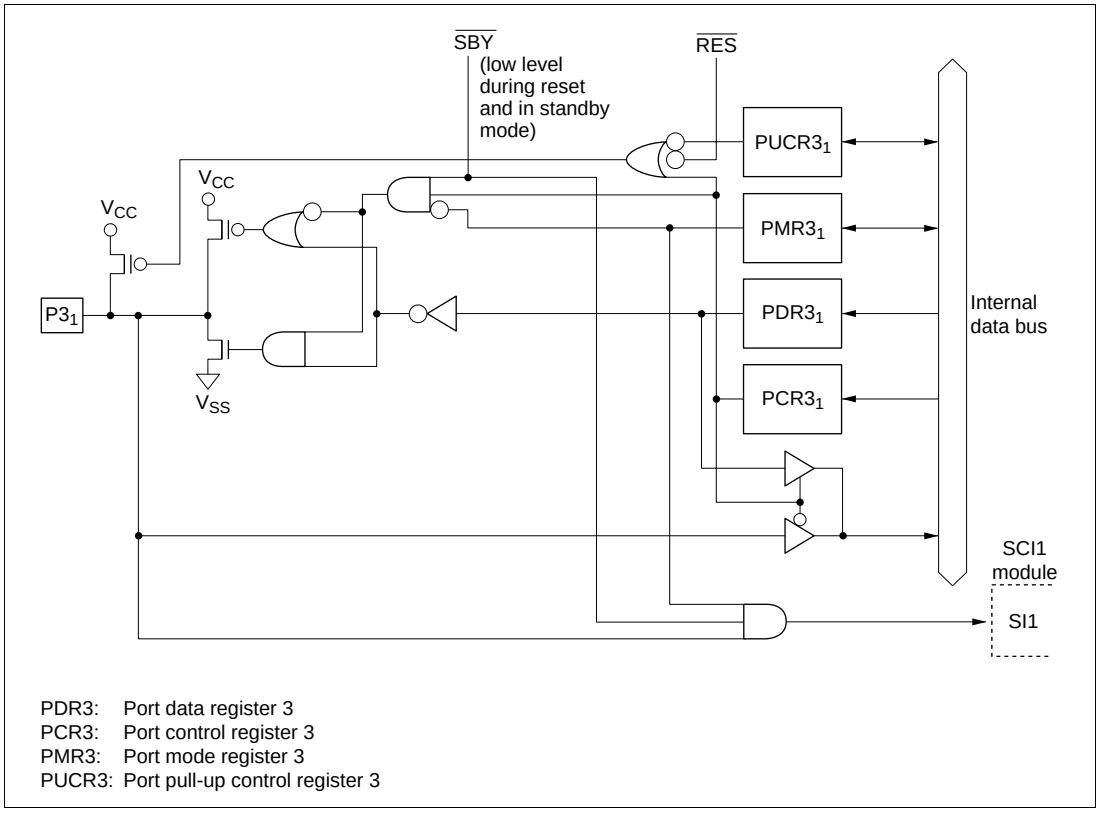


Figure C.3 (a) Port 3 Block Diagram (Pin P3$_2$)



PDR3: Port data register 3
PCR3: Port control register 3
PMR3: Port mode register 3
PUCR3: Port pull-up control register 3

Figure C.3 (b) Port 3 Block Diagram (Pin P3$_1$)

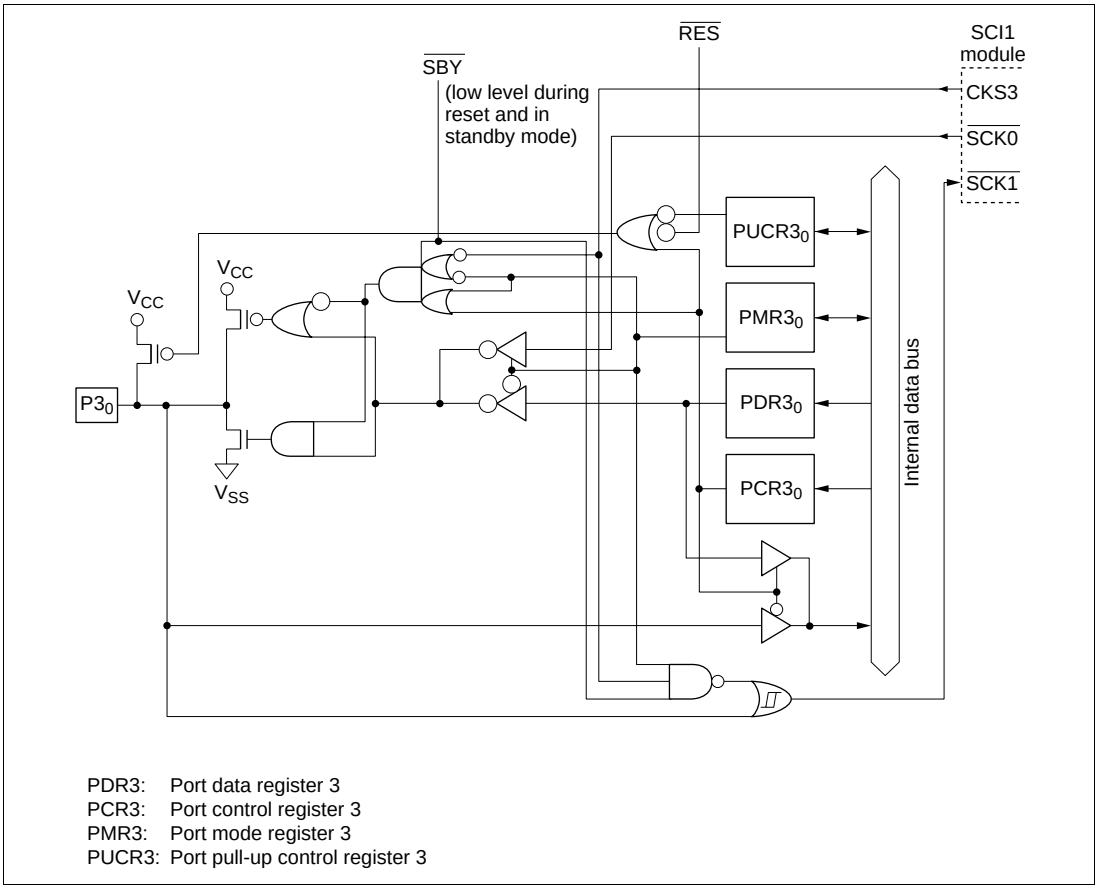


Figure C.3 (c) Port 3 Block Diagram (Pin $P3_0$)

C.4 Block Diagrams of Port 5

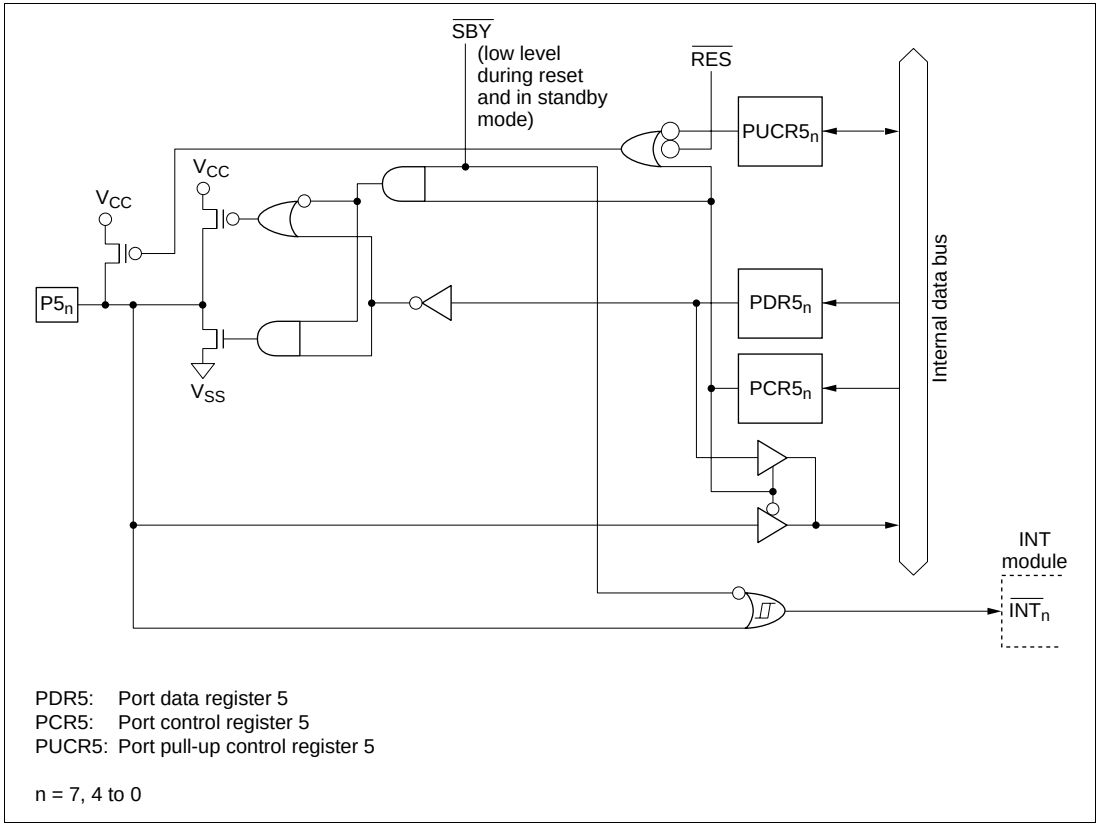


Figure C.4 (a) Port 5 Block Diagram (Pins $P5_7$ and $P5_4$ to $P5_0$)

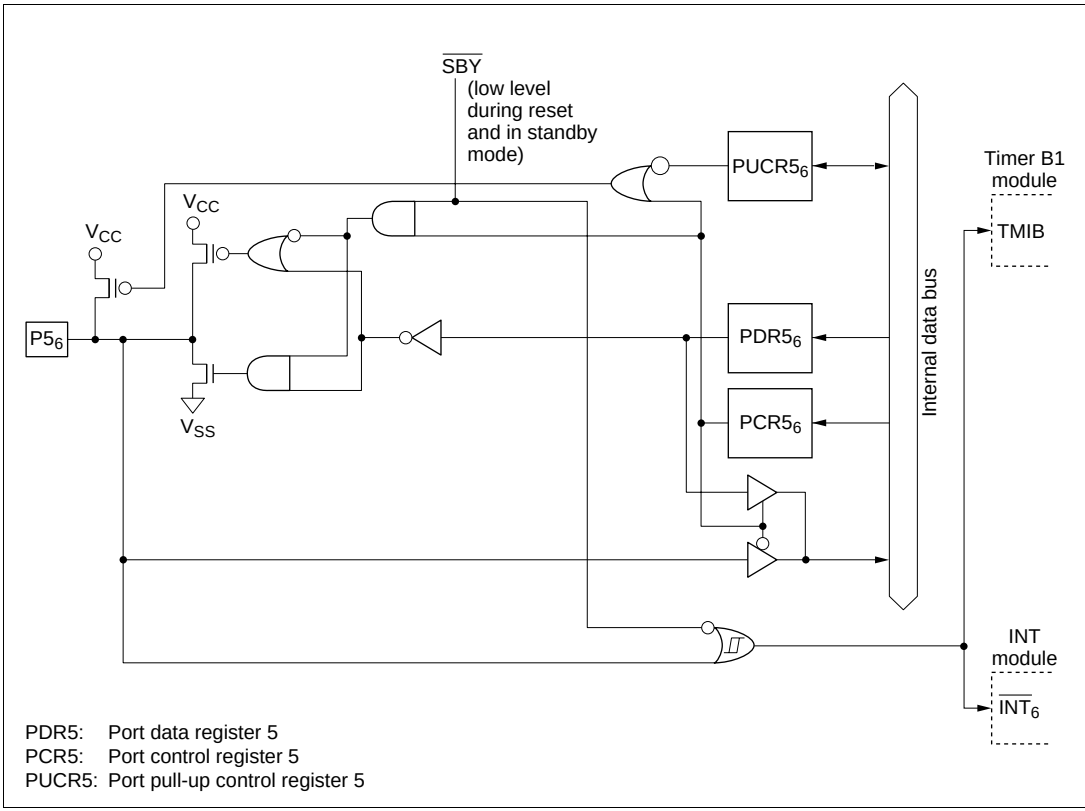


Figure C.4 (b) Port 5 Block Diagram (Pin $P5_6$)

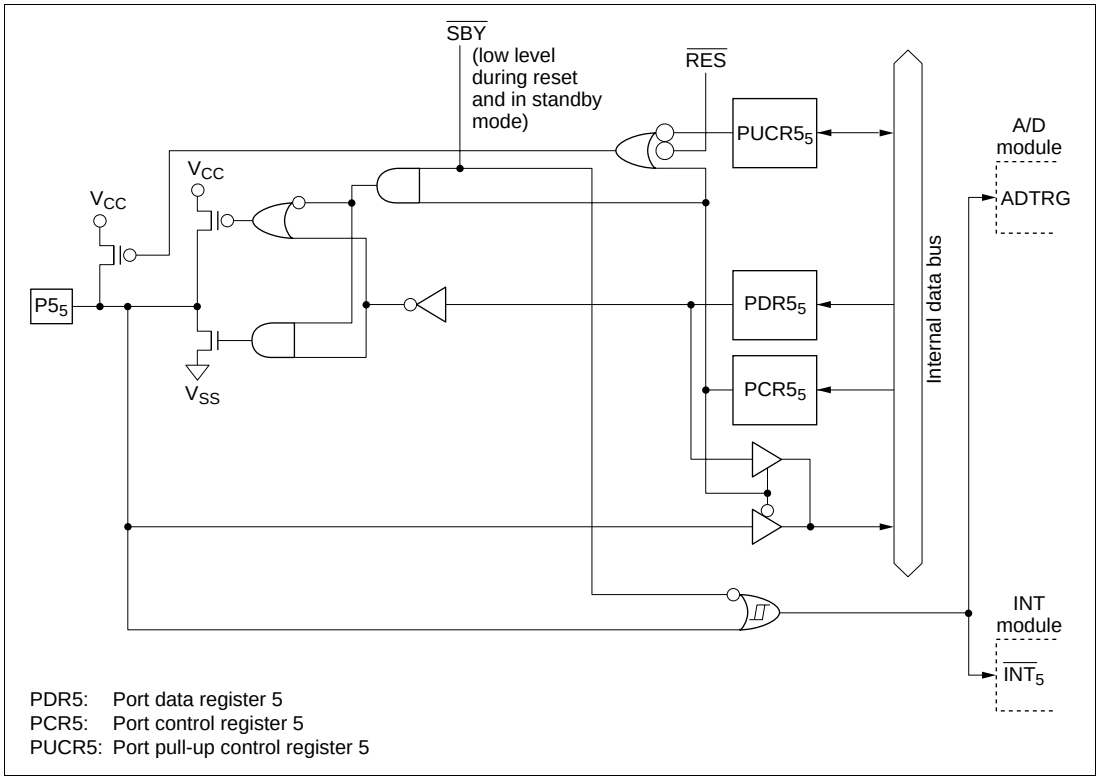
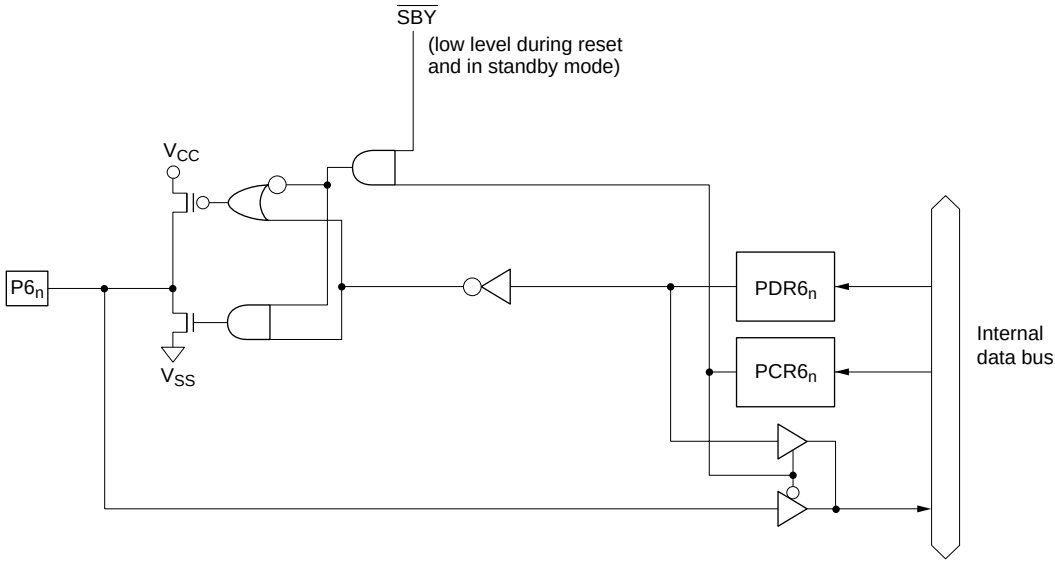


Figure C.4 (c) Port 5 Block Diagram (Pin $P5_5$)

C.5 Block Diagram of Port 6



PDR6: Port data register 6
PCR6: Port control register 6

n = 7 to 0

Figure C.5 Port 6 Block Diagram (Pins $P6_7$ to $P6_0$)

C.6 Block Diagrams of Port 7

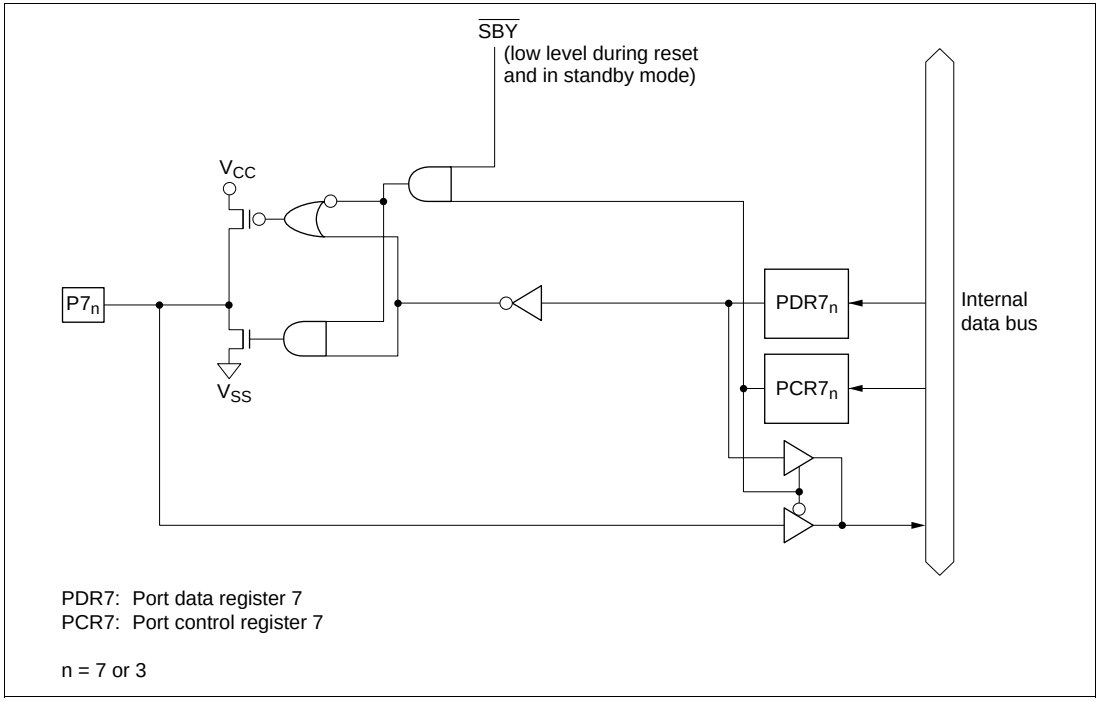


Figure C.6 (a) Port 7 Block Diagram (Pins $P7_7$ and $P7_3$)

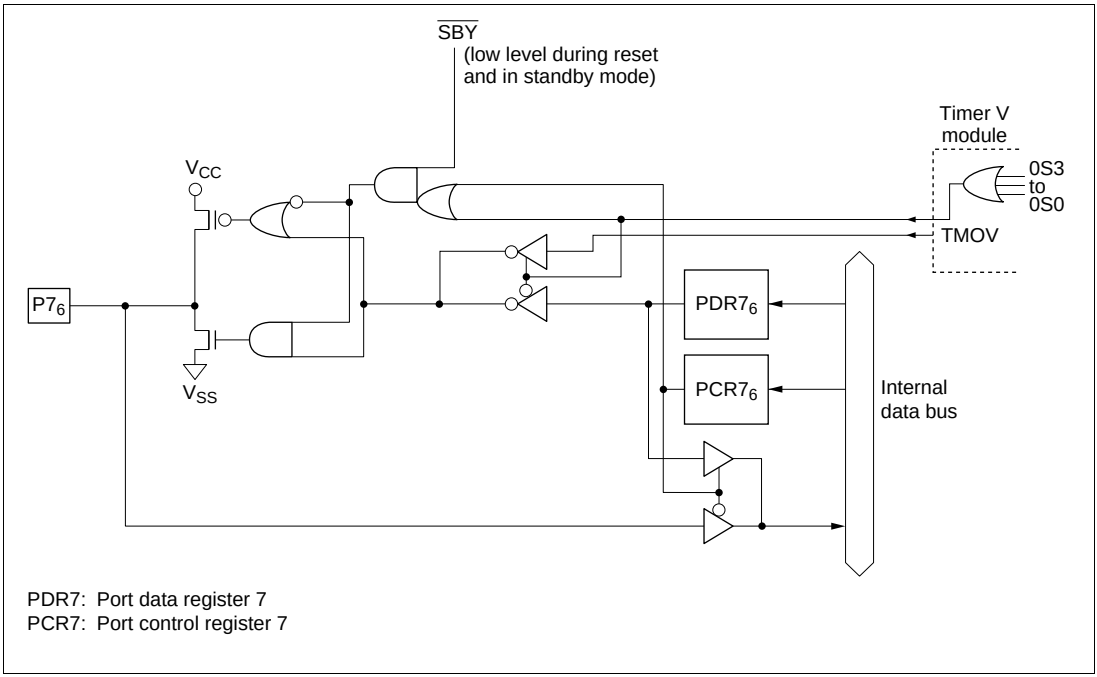


Figure C.6 (b) Port 7 Block Diagram (Pin $P7_6$)

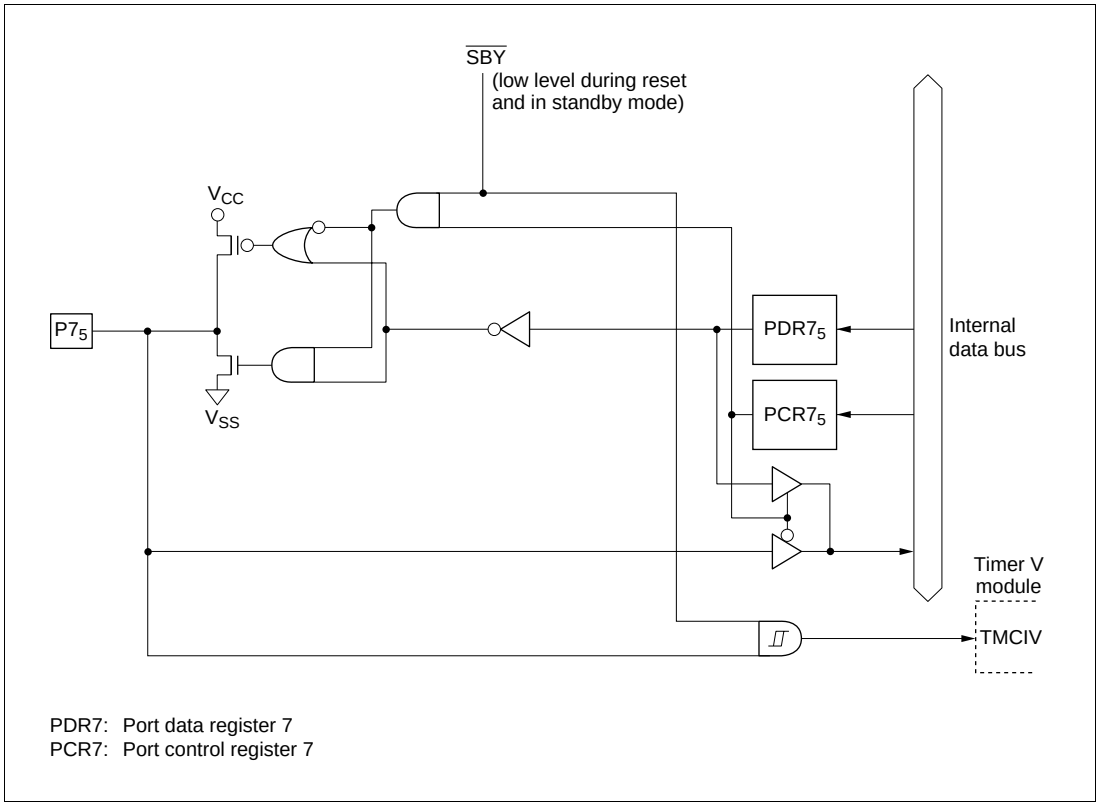


Figure C.6 (c) Port 7 Block Diagram (Pin $P7_5$)

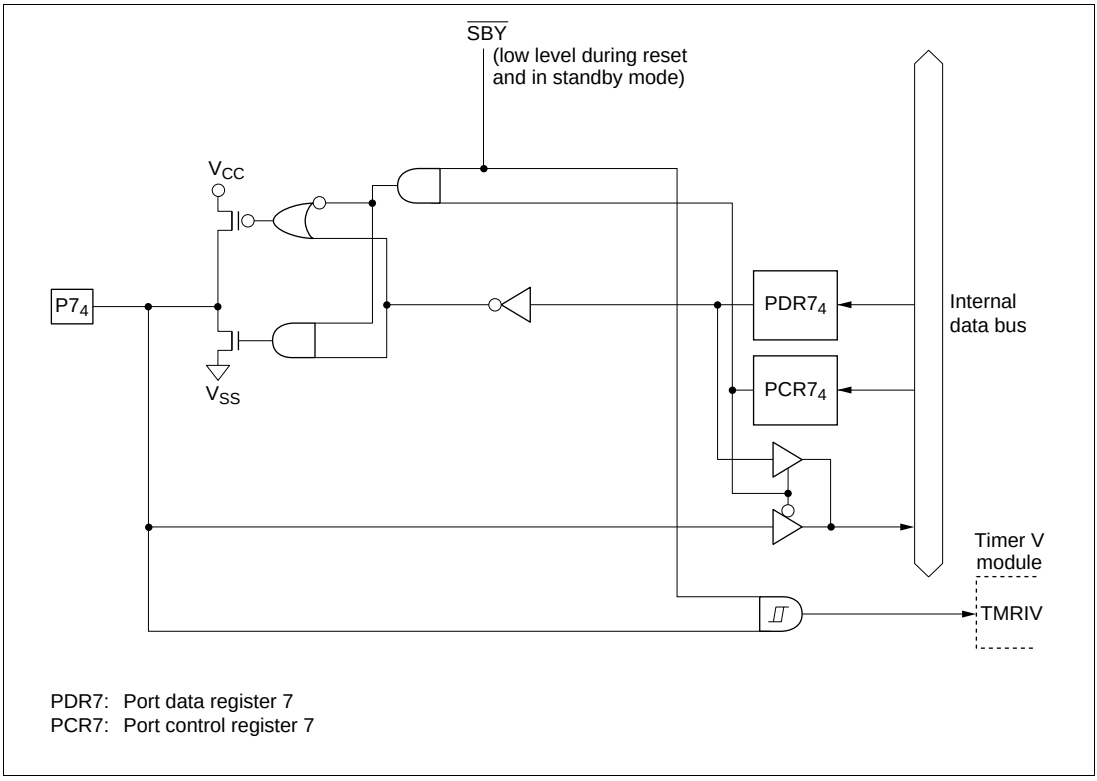


Figure C.6 (d) Port 7 Block Diagram (Pin $P7_4$)

C.7 Block Diagrams of Port 8

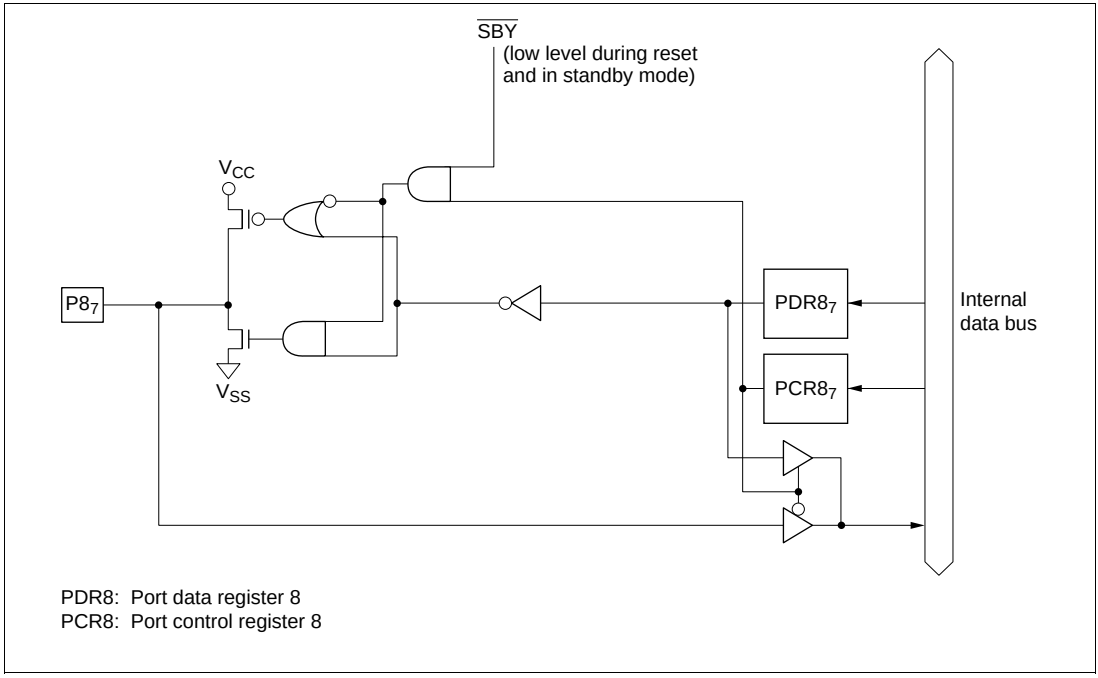


Figure C.7 (a) Port 8 Block Diagram (Pin $P8_7$)

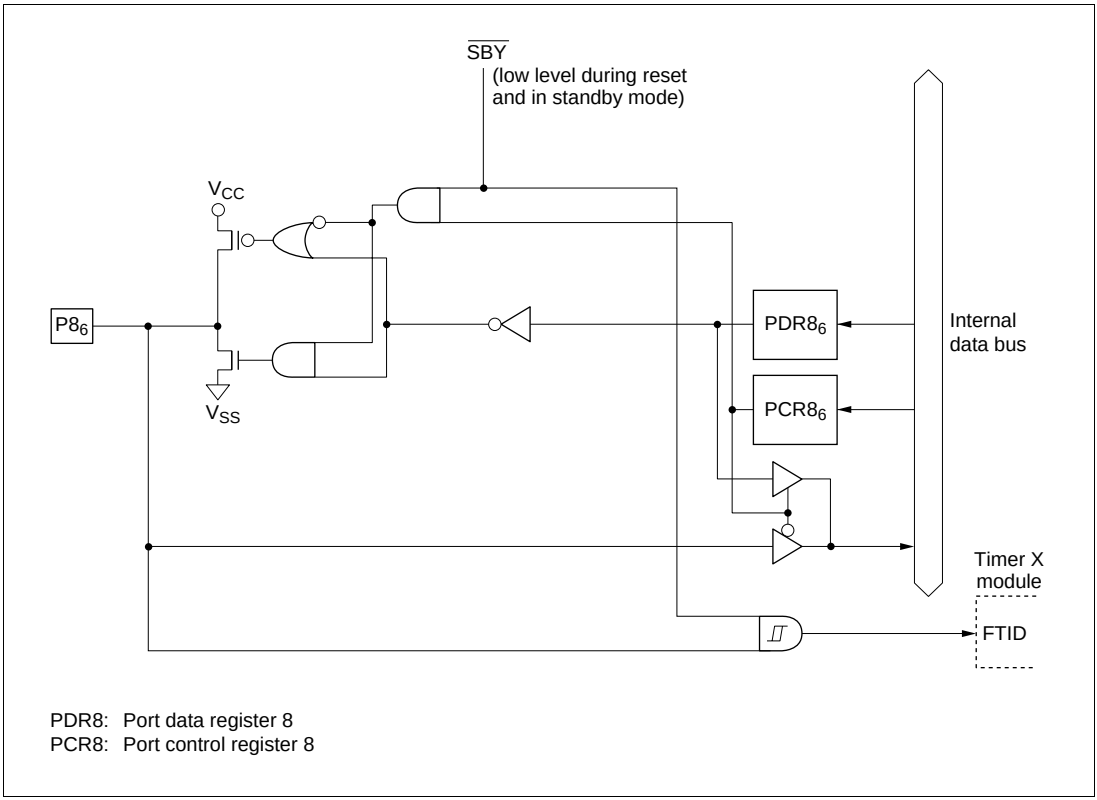


Figure C.7 (b) Port 8 Block Diagram (Pin $P8_6$)

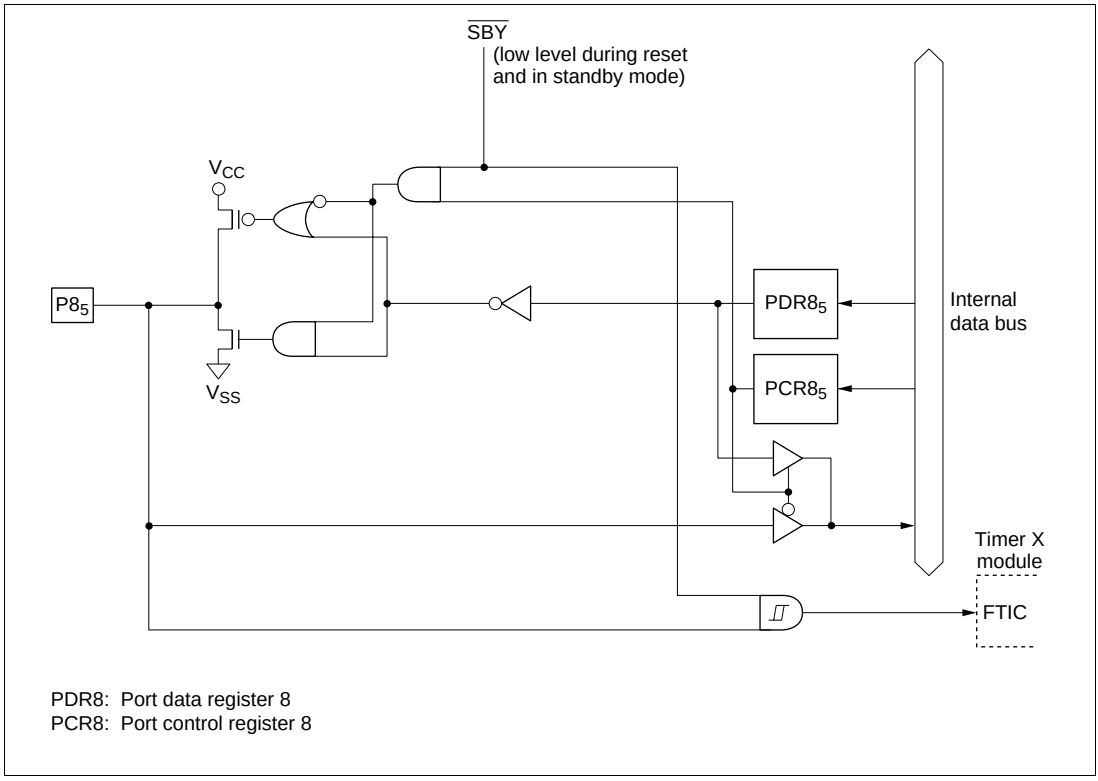


Figure C.7 (c) Port 8 Block Diagram (Pin $P8_5$)

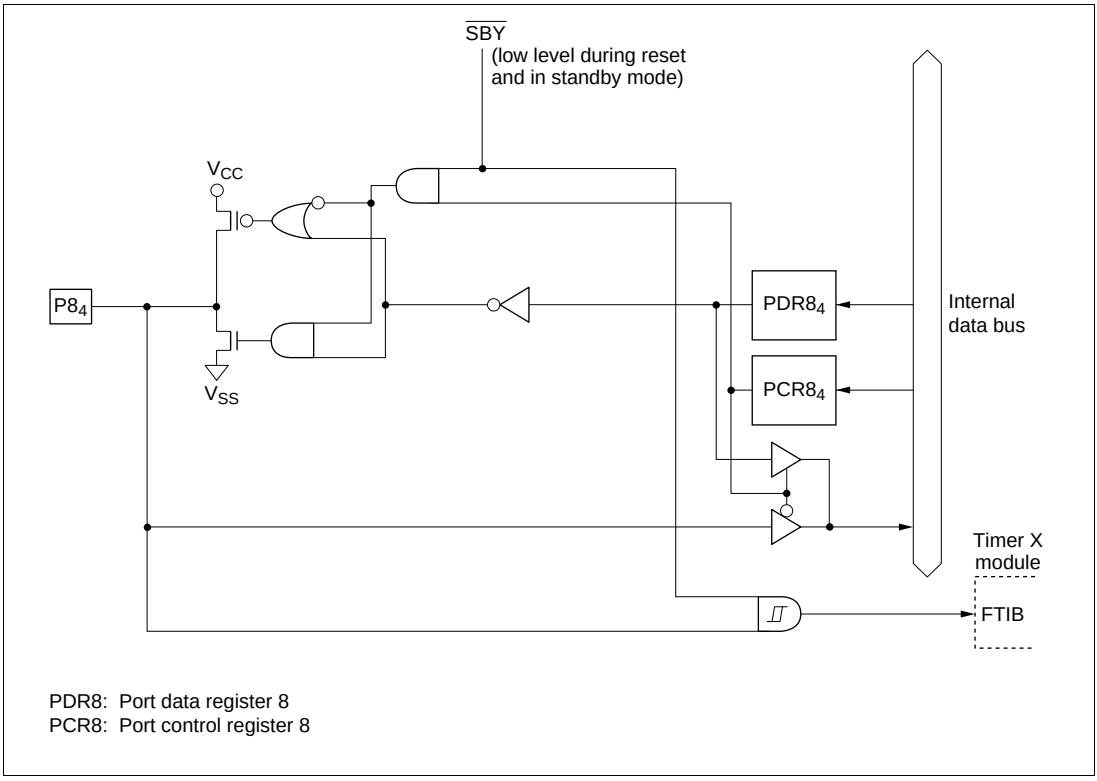


Figure C.7 (d) Port 8 Block Diagram (Pin $P8_4$)

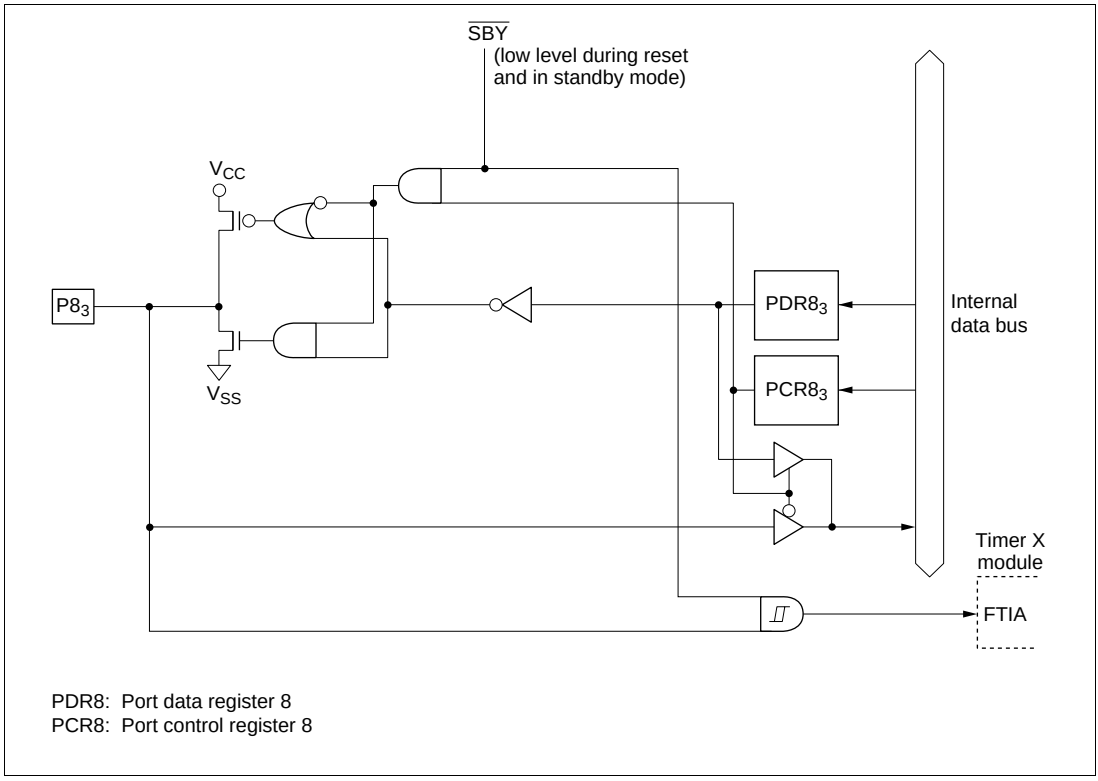


Figure C.7 (e) Port 8 Block Diagram (Pin $P8_3$)

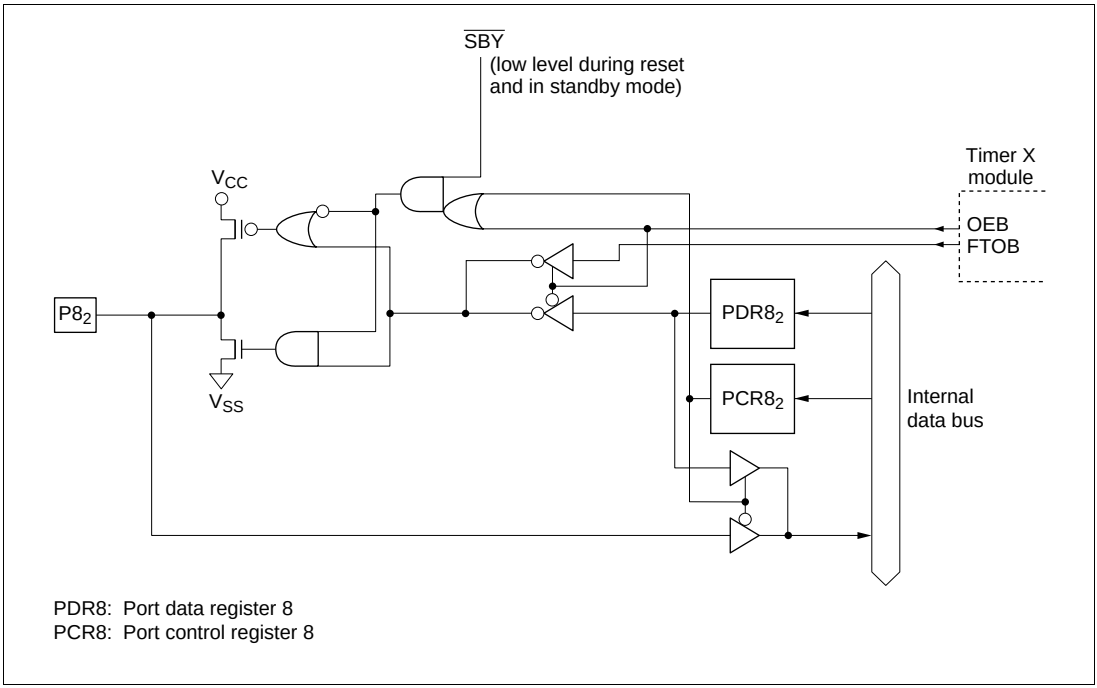


Figure C.7 (f) Port 8 Block Diagram (Pin $P8_2$)

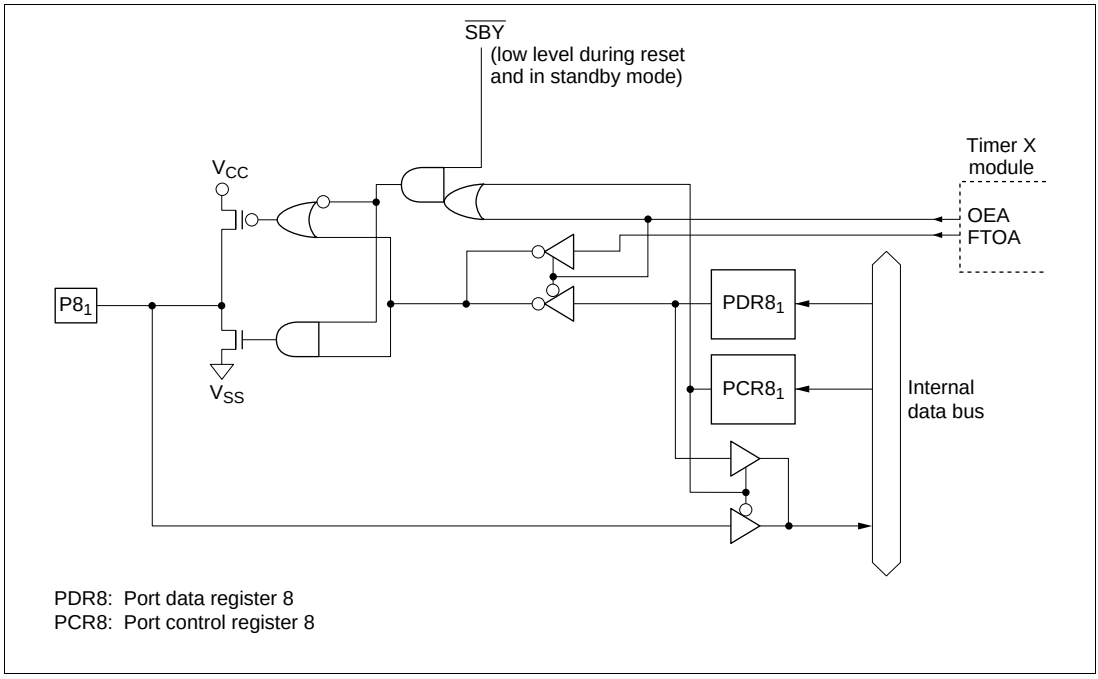


Figure C.7 (g) Port 8 Block Diagram (Pin $P8_1$)

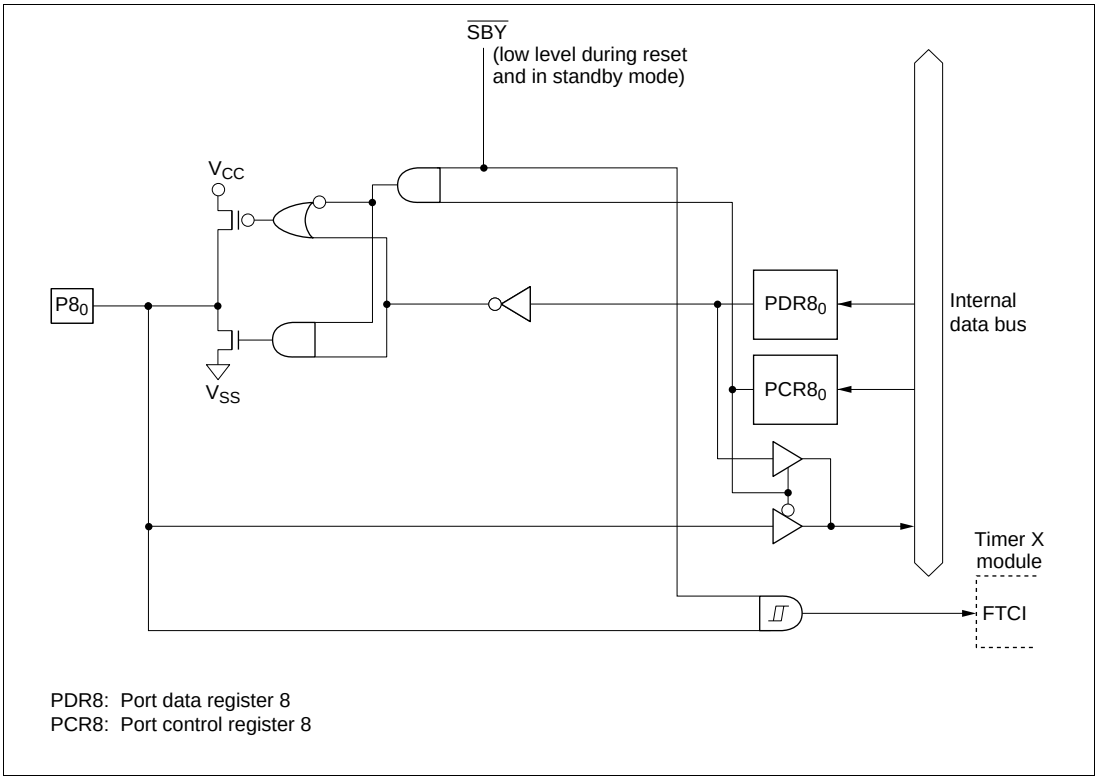


Figure C.7 (h) Port 8 Block Diagram (Pin $P8_0$)

C.8 Block Diagram of Port 9

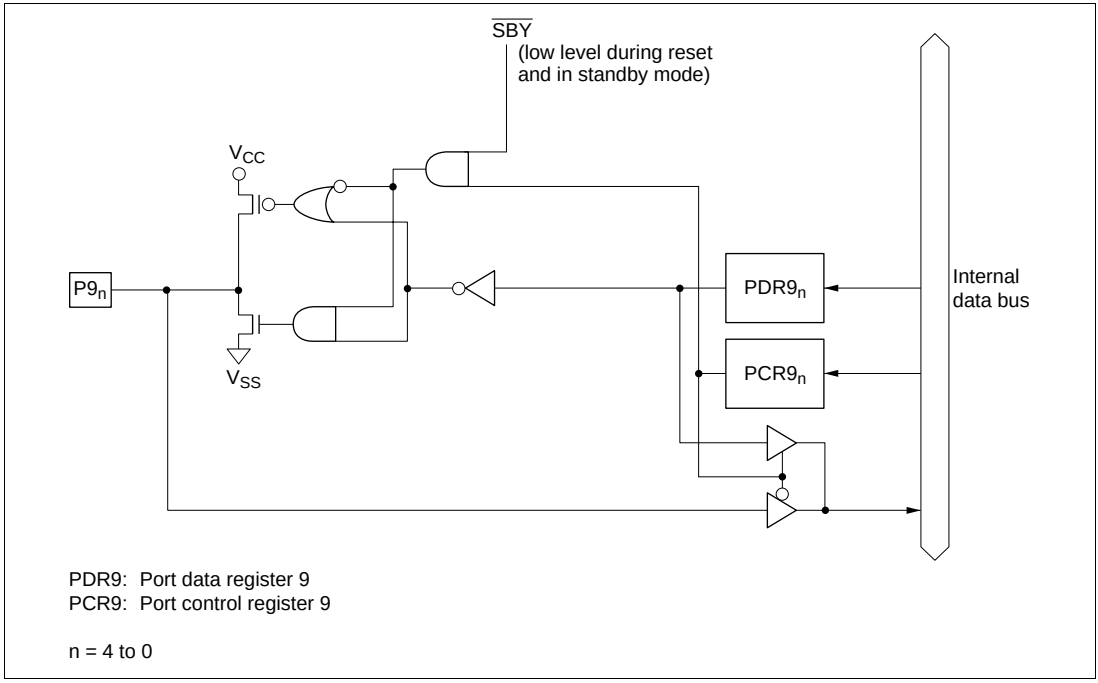


Figure C.8 Port 9 Block Diagram (Pins $P9_4$ to $P9_0$)

C.9 Block Diagram of Port B

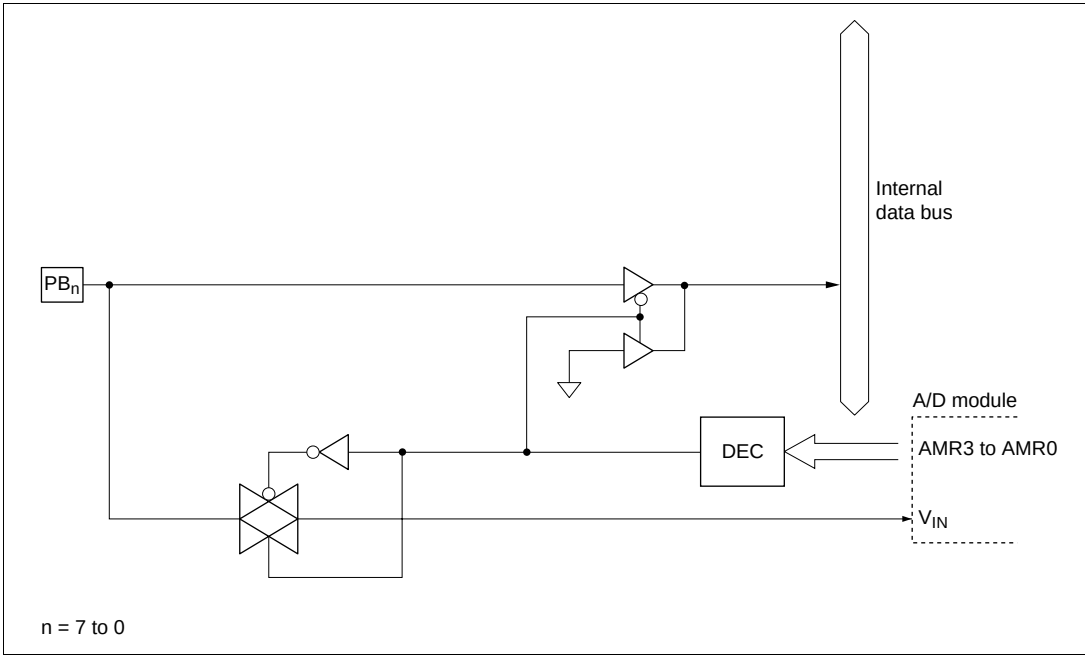


Figure C.9 Port B Block Diagram (Pins PB_7 to PB_0)

Appendix D Port States in the Different Processing States

Table D.1 Port States Overview

Port	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P1_7$ to $P1_4$, $P1_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P2_2$ to $P2_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P3_2$ to $P3_0$	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 $P7_3$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P8_7$ to $P8_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P9_4$ to $P9_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
PB_7 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 Product Code Lineup

Table E.1 Product Lineup

Product Type			Product Code	Mark Code	Package (Hitachi Package Code)
H8/3644	ZTAT™ version	Standard products	HD6473644H	HD6473644H	64-pin QFP (FP-64A)
			HD6473644P	HD6473644P	64-pin SDIP (DP-64S)
			HD6473644W	HD6473644W	80-pin TQFP (TFP-80C)
	F-ZTAT™ version		HD64F3644H	HD64F3644H	64-pin QFP (FP-64A)
			HD64F3644P	HD64F3644P	64-pin SDIP (DP-64S)
			HD64F3644W	HD64F3644W	80-pin TQFP (TFP-80C)
	Mask ROM version		HD6433644H	HD6433644(***)H	64-pin QFP (FP-64A)
			HD6433644P	HD6433644(***)P	64-pin SDIP (DP-64S)
			HD6433644W	HD6433644(***)W	80-pin TQFP (TFP-80C)
H8/3643	FLASH	Standard products	HD64F3643H	HD64F3643H	64-pin QFP (FP-64A)
			HD64F3643P	HD64F3643P	64-pin SDIP (DP-64S)
			HD64F3643W	HD64F3643W	80-pin TQFP (TFP-80C)
	Mask ROM version		HD6433643H	HD6433643(***)H	64-pin QFP (FP-64A)
			HD6433643P	HD6433643(***)P	64-pin SDIP (DP-64S)
			HD6433643W	HD6433643(***)W	80-pin TQFP (TFP-80C)
H8/3642	FLASH	Standard products	HD64F3642AH	HD64F3642AH	64-pin QFP (FP-64A)
			HD64F3642AP	HD64F3642AP	64-pin SDIP (DP-64S)
			HD64F3642AW	HD64F3642AW	80-pin TQFP (TFP-80C)
	Mask ROM version		HD6433642H	HD6433642(***)H	64-pin QFP (FP-64A)
			HD6433642P	HD6433642(***)P	64-pin SDIP (DP-64S)
			HD6433642W	HD6433642(***)W	80-pin TQFP (TFP-80C)
H8/3641	Mask ROM version	Standard products	HD6433641H	HD6433641(***)H	64-pin QFP (FP-64A)
			HD6433641P	HD6433641(***)P	64-pin SDIP (DP-64S)
			HD6433641W	HD6433641(***)W	80-pin TQFP (TFP-80C)
H8/3640	Mask ROM version	Standard products	HD6433640H	HD6433640(***)H	64-pin QFP (FP-64A)
			HD6433640P	HD6433640(***)P	64-pin SDIP (DP-64S)
			HD6433640W	HD6433640(***)W	80-pin TQFP (TFP-80C)

Note: For mask ROM versions, (***) is the ROM code.

Appendix F Package Dimensions

Dimensional drawings of H8/3644 packages FP-64A, DP-64S and TFP-80C are shown in figures F.1 to F.3 below.

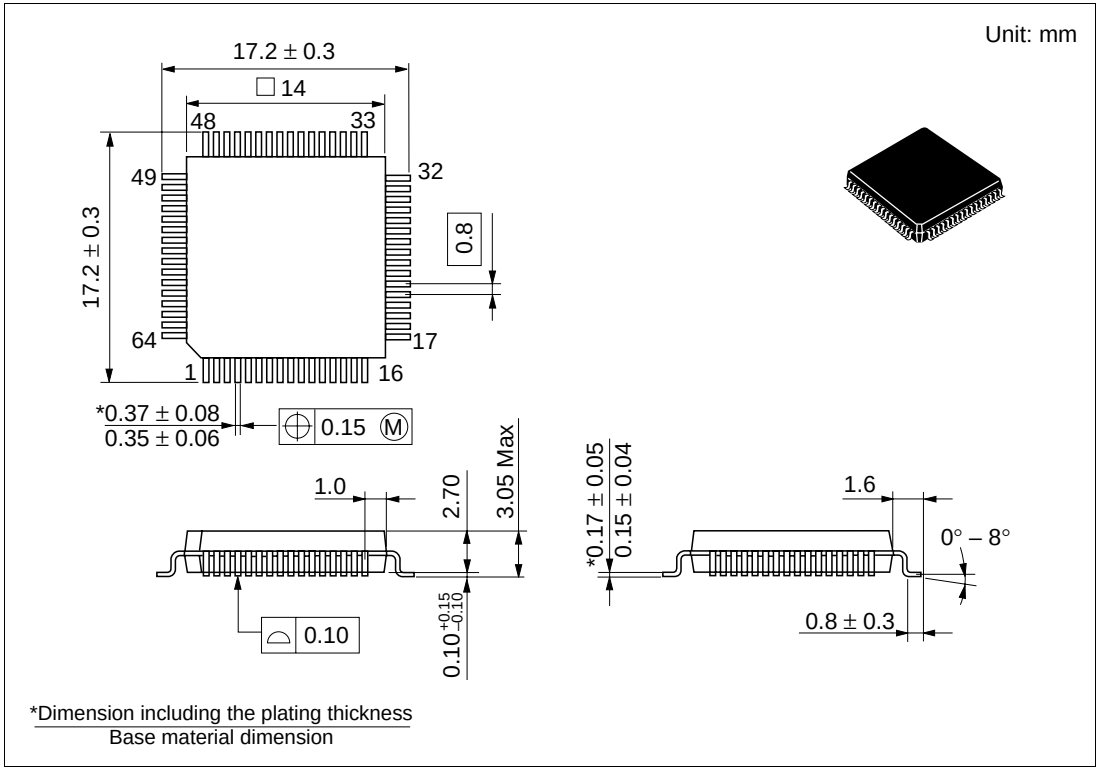


Figure F.1 FP-64A Package Dimensions

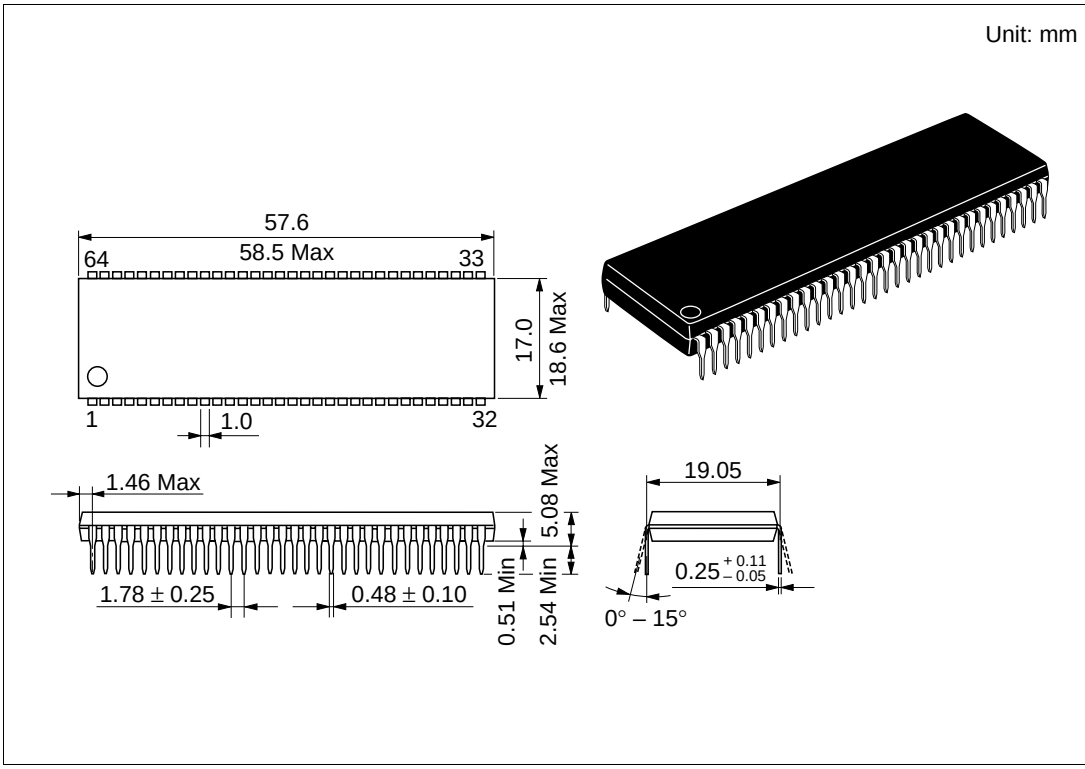


Figure F.2 DP-64S Package Dimensions

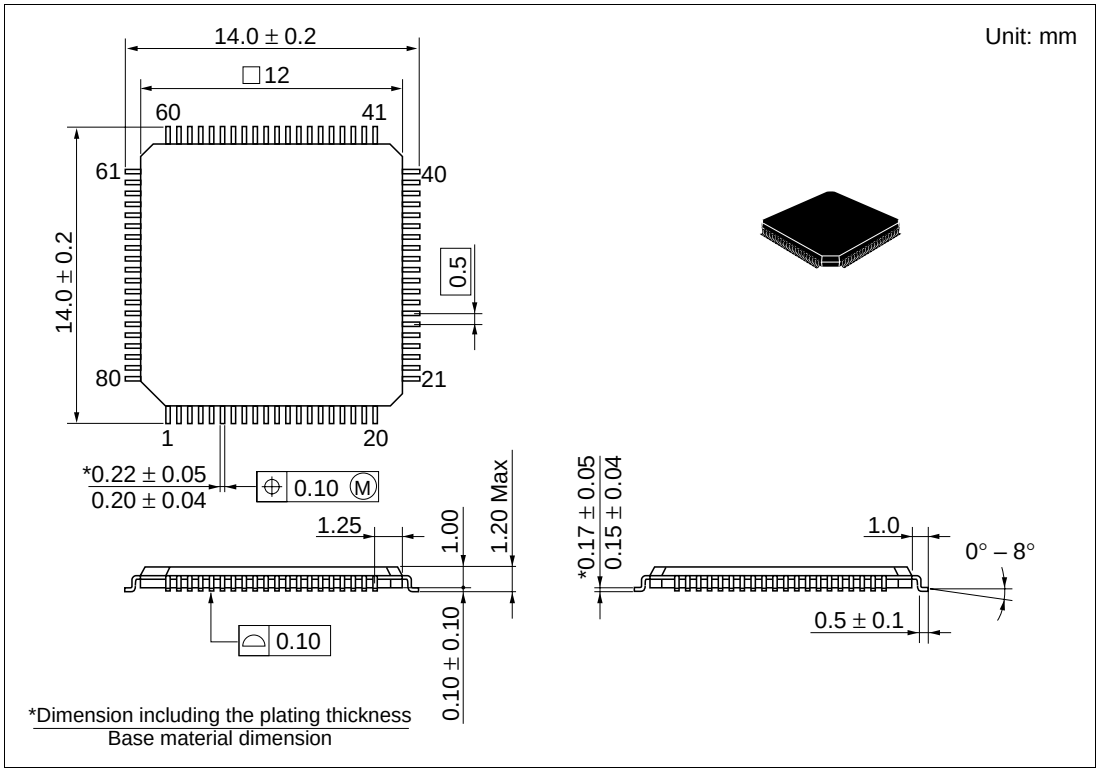


Figure F.3 TFP-80C Package Dimensions

Note: In case of inconsistencies arising within figures, dimensional drawings listed in the Hitachi Semiconductor Packages Manual take precedence and are considered correct.

H8/3644 Series, H8/3644F-ZTAT™,
H8/3643 F-ZTAT™, H8/3642 AF-ZTAT™,
Hardware Manual

Publication Date: 1st Edition, September 1996
4th Edition, August 1998
Published by: Electronic Devices Sales & Marketing Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.