OMC942723156

H8/3947 Series

H8/3945, H8/3946, H8/3947

Hardware Manual

Preface

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

On-chip peripheral functions of the H8/3947 Series include an I^2C bus interface, eight types of timers, an 8-bit pulse-width modulator, a serial communication interface, and an A/D converter. The H8/3947 Series can be advantageously used for embedded control in systems employing an I^2C bus interface for internal communication.

This manual describes the hardware of the H8/3947 Series. For details on the H8/3947 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 H8/3947 Series of microcomputers is equipped with two I^2C bus interface channels, as proposed by Philips. Other on-chip peripheral functions include eight timers, an 8-bit pulse width modulator (PWM) with eight channels, a serial communication interface, and an A/D converter. The H8/3947 Series is ideally suited for embedded applications in systems that use the I^2C bus interface for internal communication.

The H8/3947 has a ZTAT™* version with on-chip, user-programmable PROM.

Table 1-1 summarizes the features of the H8/3947 Series.

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

Table 1-1 Features

Item	Description
CPU	High-speed H8/300L CPU
	• General-register architecture
	General registers: Sixteen 8-bit registers (can be used as eight 16-bit registers)
	• Operating speed
	— Max. operating speed: 5 MHz
	— Add/subtract: 0.4 µs (operating at 5 MHz)
	— Multiply/divide: 2.8 µs (operating at 5 MHz)
	— Can run on 32.768 kHz subclock
	• Instruction set compatible with H8/300 CPU
	— Instruction length of 2 bytes or 4 bytes
	— Basic arithmetic operations between registers
	— MOV instruction for data transfer between memory and registers

Table 1-1 Features (cont)

Item	Description
CPU	Typical instructions
	• Multiply (8 bits × 8 bits)
	• Divide (16 bits ÷ 8 bits)
	• Bit accumulator
	• Register-indirect designation of bit position
Interrupts	38 interrupt sources
	• 14 external interrupt sources (NMI, IRQ_4 to IRQ_0, WKP_7 to WKP_0)
	• 24 internal interrupt sources
Clock pulse generators	Two on-chip clock pulse generators
	• System clock pulse generator: 1 to 10 MHz
	• Subclock pulse generator: 32.768 kHz
Power-down modes	Six power-down modes
	• Sleep mode
	• Standby mode
	• Watch mode
	• Subsleep mode
	• Subactive mode
	• Active (medium-speed) mode
Memory	Large on-chip memory
	• H8/3945: 40-kbyte ROM; 2-kbyte RAM
	• H8/3946: 48-kbyte ROM; 2-kbyte RAM
	• H8/3947: 60-kbyte ROM; 2-kbyte RAM
I/O ports	86 pins
	• 69 I/O pins (including 8 medium-voltage NMOS open-drain output pins (+15 V) and 7 high-current output pins)
	• 17 input pins

Table 1-1 Features (cont)

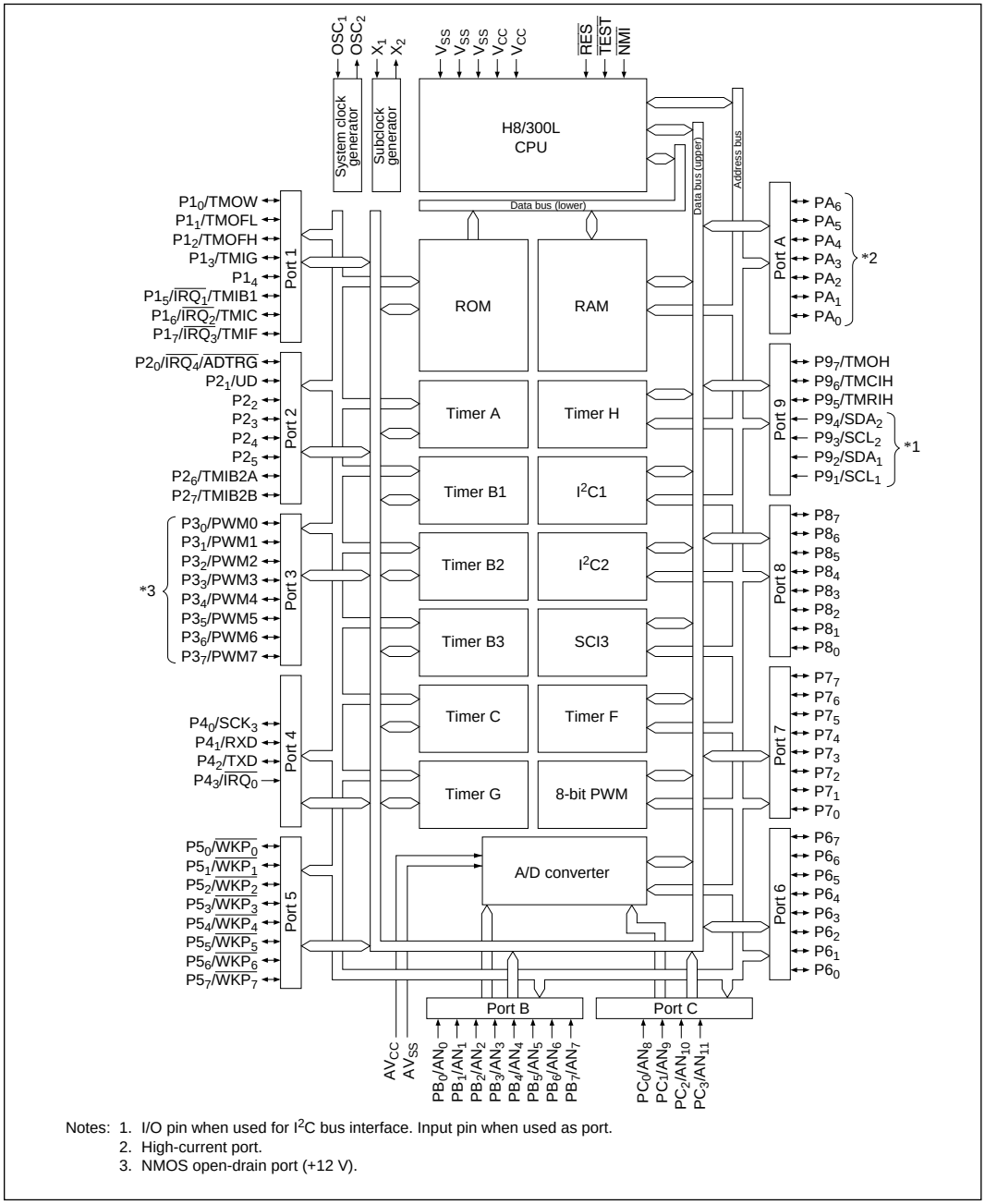
Item	Description
Timers	Eight on-chip timers
	• Timer A: 8-bit timer
	Count-up timer with selection of eight internal clock signals divided from the system clock (ø)* and four clock signals divided from the watch clock ($ø_W$)*
	• Timer B1: 8-bit timer
	— Count-up timer with selection of seven internal clock signals or event input from external pin
	— Auto-reloading
	• Timer B2: 8-bit timer
	— Count-up timer with selection of seven internal clock signals or event input from external pins
	— Selection of two event inputs
	— Auto-reloading
	• Timer B3: 8-bit timer
	— Count-up timer with selection of seven internal clock signals
	— Auto-reloading
	• Timer C: 8-bit timer
	— Count-up/count-down timer with selection of seven internal clock signals or event input from external pin
	— Auto-reloading
	• Timer F: 16-bit timer
	— Can be used as two independent 8-bit timers.
	— Count-up timer with selection of four internal clock signals or event input from external pin
	— Compare-match function with toggle output
	• Timer G: 8-bit timer
	— Count-up timer with selection of four internal clock signals
	— Input capture function with built-in noise canceller circuit
	• Timer H: 8-bit timer
	— Count-up timer with selection of three internal clock signals or event input from external pin
	— Compare-match waveform output
	Note: * ø and $ø_W$ are defined in section 4, Clock Pulse Generators.

Table 1-1 Features (cont)

Item	Specification			
Serial communication interface (including I^2C bus interface)	Three channels on chip • SCI3: 8-bit synchronous or asynchronous serial interface Built-in function for multiprocessor communication • I^2C1: I^2C bus interface 1 Includes single master mode and slave mode • I^2C2: I^2C bus interface 2 Includes single master mode and slave mode			
8-bit PWM	Eight channels on chip • All output pins withstand up to +12 V • Can be used as an 8-bit D/A converter by connecting an external low-pass filter			
A/D converter	• 8-bit successive approximations using a resistance ladder • 12-channel analog input port • Conversion time: 31/ø or 62/ø per channel			
Product lineup	**Product Code**			
	Mask ROM Version	**ZTAT™ Version**	**Package**	**ROM/RAM Size**
	HD6433945F	—	100-pin QFP (FP-100A)	ROM: 40 kbytes RAM: 2 kbytes
	HD6433946F	—	100-pin QFP (FP-100A)	ROM: 48 kbytes RAM: 2 kbytes
	HD6433947F	HD6473947F	100-pin QFP (FP-100A)	ROM: 60 kbytes RAM: 2 kbytes

1.2 Internal Block Diagram

Figure 1-1 shows a block diagram of the H8/3947 Series.



Notes: 1. I/O pin when used for I²C bus interface. Input pin when used as port.
2. High-current port.
3. NMOS open-drain port (+12 V).

Figure 1-1 Block Diagram

1.3 Pin Arrangement and Functions

1.3.1 Pin Arrangement

The H8/3947 Series pin arrangement is shown in figure 1-2.

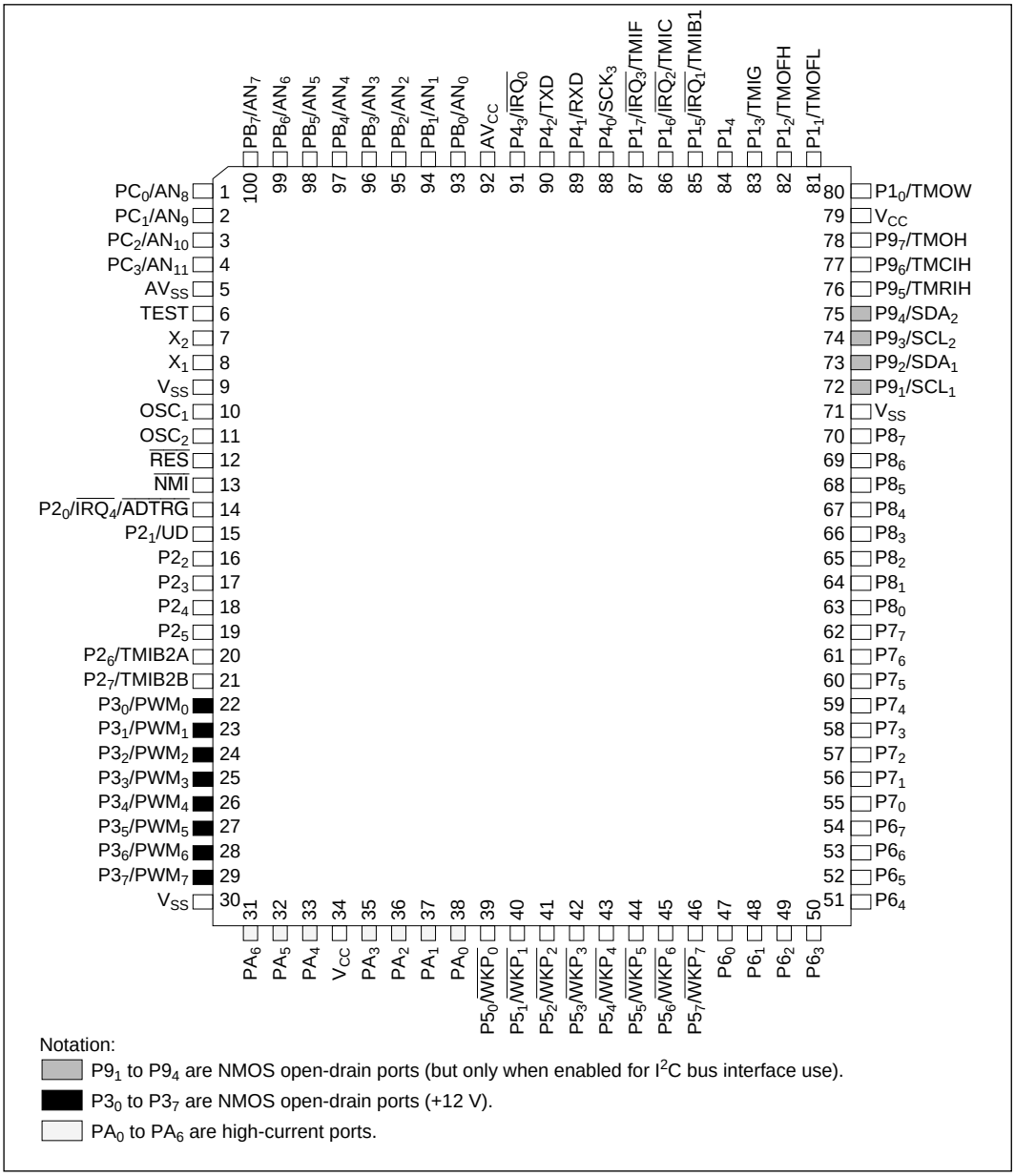


Figure 1-2 Pin Arrangement (FP-100A: Top View)

1.3.2 Pin Functions

Table 1-2 outlines the pin functions of the H8/3947 Series.

Table 1-2 Pin Functions

Type	Symbol	Pin No. FP-100A	I/O	Name and Functions
Power source pins	V_{CC}	34, 79	Input	**Power supply:** All V_{CC} pins should be connected to the system power supply (+5 V)
	V_{SS}	9, 30, 71	Input	**Ground:** All V_{SS} pins should be connected to the system power supply (0 V)
	AV_{CC}	92	Input	**Analog power supply:** This is the power supply pin for the A/D converter. When the A/D converter is not used, connect this pin to the system power supply (+5 V).
	AV_{SS}	5	Input	**Analog ground:** This is the A/D converter ground pin. It should be connected to the system power supply (0 V).
Clock pins	OSC_1	10	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	11	Output	
	X_1	8	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	7	Output	
System control	$\overline{RES}$	12	Input	**Reset:** When this pin is driven low, the chip is reset
	TEST	6	Input	**Test:** This is a test pin, not for use in application systems. It should be connected to V_{SS}.

Table 1-2 Pin Functions (cont)

Type	Symbol	Pin No. FP-100A	I/O	Name and Functions
Interrupt pins	$\overline{NMI}$	13	Input	**Nonmaskable interrupt:** This is an input pin for an edge-sensitive nonmaskable interrupt, with a selection of rising or falling edge
	$\overline{IRQ}_0$ $\overline{IRQ}_1$ $\overline{IRQ}_2$ $\overline{IRQ}_3$ $\overline{IRQ}_4$	91 85 86 87 14	Input	**External interrupt request 0 to 4:** These are input pins for external interrupts for which there is a choice between rising and falling edge sensing
	$\overline{WKP}_7$ to $\overline{WKP}_0$	46 to 39	Input	**Wakeup interrupt request 0 to 7:** These are input pins for external interrupts that are detected at the falling edge
Timer pins	TMOW	80	Output	**Clock output:** This is an output pin for waveforms generated by the timer A output circuit
	TMIB1	85	Input	**Timer B1 event input:** This is an event input pin for input to the timer B1 counter
	TMIB2A	20	Input	**Timer B2 event input:** This is an event input pin for input to the timer B2 counter
	TMIB2B	21	Input	**Timer B2 event input:** This is an event input pin for input to the timer B2 counter
	TMIC	86	Input	**Timer C event input:** This is an event input pin for input to the timer C counter
	UD	15	Input	**Timer C up/down select:** This pin selects whether the timer C counter is used for up- or down-counting. At high level it selects up-counting, and at low level down-counting.
	TMIF	87	Input	**Timer F event input:** This is an event input pin for input to the timer F counter

Table 1-2 Pin Functions (cont)

Type	Symbol	Pin No. FP-100A	I/O	Name and Functions
Timer pins	TMOFL	81	Output	**Timer FL output:** This is an output pin for waveforms generated by the timer FL output compare function
	TMOFH	82	Output	**Timer FH output:** This is an output pin for waveforms generated by the timer FH output compare function
	TMIG	83	Input	**Timer G capture input:** This is an input pin for the timer G input capture function
	TMOH	78	Output	**Timer H output:** This is an output pin for waveforms generated by the timer H output compare function
	TMCIH	77	Input	**Timer H event input:** This is an event input pin for input to the timer H counter
	TMRIH	76	Input	**Timer H reset input:** This is a reset input pin for the timer H counter
I/O ports	PB_7 to PB_0	100 to 93	Input	**Port B:** This is an 8-bit input port
	PC_3 to PC_0	4 to 1	Input	**Port C:** This is a 4-bit input port
	$P4_3$	91	Input	**Port 4 (bit 3):** This is a 1-bit input port
	$P4_2$ to $P4_0$	90 to 88	I/O	**Port 4 (bits 2 to 0):** This is a 3-bit I/O port. Input or output can be designated for each bit by means of port control register 4 (PCR4).
	$P1_7$ to $P1_0$	87 to 80	I/O	**Port 1:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 1 (PCR1).
	$P2_7$ to $P2_0$	21 to 14	I/O	**Port 2:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 2 (PCR2).

Table 1-2 Pin Functions (cont)

Type	Symbol	Pin No. FP-100B	Pin No. FP-100A	I/O	Name and Functions
I/O ports	$P3_7$ to $P3_0$	29 to 22	I/O		**Port 3:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 3 (PCR3).
	$P5_7$ to $P5_0$	46 to 39	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 $P6_0$	54 to 47	I/O		**Port 6:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 6 (PCR6).
	$P7_7$ to $P7_0$	62 to 55	I/O		**Port 7:** This is an 8-bit I/O port. Input or output can be designated for each bit by means of port control register 7 (PCR7).
	$P8_7$ to $P8_0$	70 to 63	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).
	$P9_7$ to $P9_5$	78 to 76	I/O		**Port 9 (bits 7 to 5):** This is a 3-bit I/O port. Input or output can be designated for each bit by means of port control register 9 (PCR9).
	$P9_4$ to $P9_1$	75 to 72	Input		**Port 9 (bits 4 to 1):** This is a 4-bit Input port
	PA_6 to PA_0	31 to 33, 35 to 38	I/O		**Port A:** This is a 7-bit I/O port. Input or output can be designated for each bit by means of port control register A (PCRA).

Table 1-2 Pin Functions (cont)

Type	Symbol	Pin No. FP-100A	I/O	Name and Functions
Serial communication interface (I^2C1, I^2C2)	SCL_1	72	I/O	**I^2C1 clock I/O:** This is the clock I/O pin for I^2C1
	SDA_1	73	I/O	**I^2C1 data I/O:** This is the data I/O pin for I^2C1
	SCL_2	74	I/O	**I^2C2 clock I/O:** This is the clock I/O pin for I^2C2
	SDA_2	75	I/O	**I^2C2 data I/O:** This is the data I/O pin for I^2C2
Serial communication interface (SCI3)	RXD	89	Input	**SCI3 receive data input:** This is the SCI3 data input pin
	TXD	90	Output	**SCI3 send data output:** This is the SCI3 data output pin
	SCK_3	88	I/O	**SCI3 clock I/O :** This is the SCI3 clock I/O pin
A/D converter	AN_{11} to AN_0	4 to 1 100 to 93	Input	**Analog input channels 11 to 0:** These are analog data input channels to the A/D converter
	$\overline{ADTRG}$	14	Input	**A/D converter trigger input:** This is the external trigger input pin to the A/D converter
8-bit PWM	PWM_7 to PWM_0	29 to 22	Output	**8-bit PWM output:** These are output pins for waveforms generated by the 8-bit PWM

Section 2 CPU

2.1 Overview

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

2.1.1 Features

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

- General-register architecture
  Sixteen 8-bit general registers, also usable as eight 16-bit general registers

- Instruction set with 55 basic instructions, including:
  — Multiply and divide instructions
  — Powerful bit-manipulation instructions

- Eight addressing modes
  — Register direct
  — Register indirect
  — Register indirect with displacement
  — Register indirect with post-increment or pre-decrement
  — Absolute address
  — Immediate
  — Program-counter relative
  — Memory indirect

- 64-kbyte address space

- High-speed operation
  — All frequently used instructions are executed in two to four states
  — High-speed arithmetic and logic operations
  — 8- or 16-bit register-register add or subtract: 0.4 µs*
  — 8 × 8-bit multiply: 2.8 µs*
  — 16 ÷ 8-bit divide: 2.8 µs*

- Low-power operation modes
  SLEEP instruction for 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.

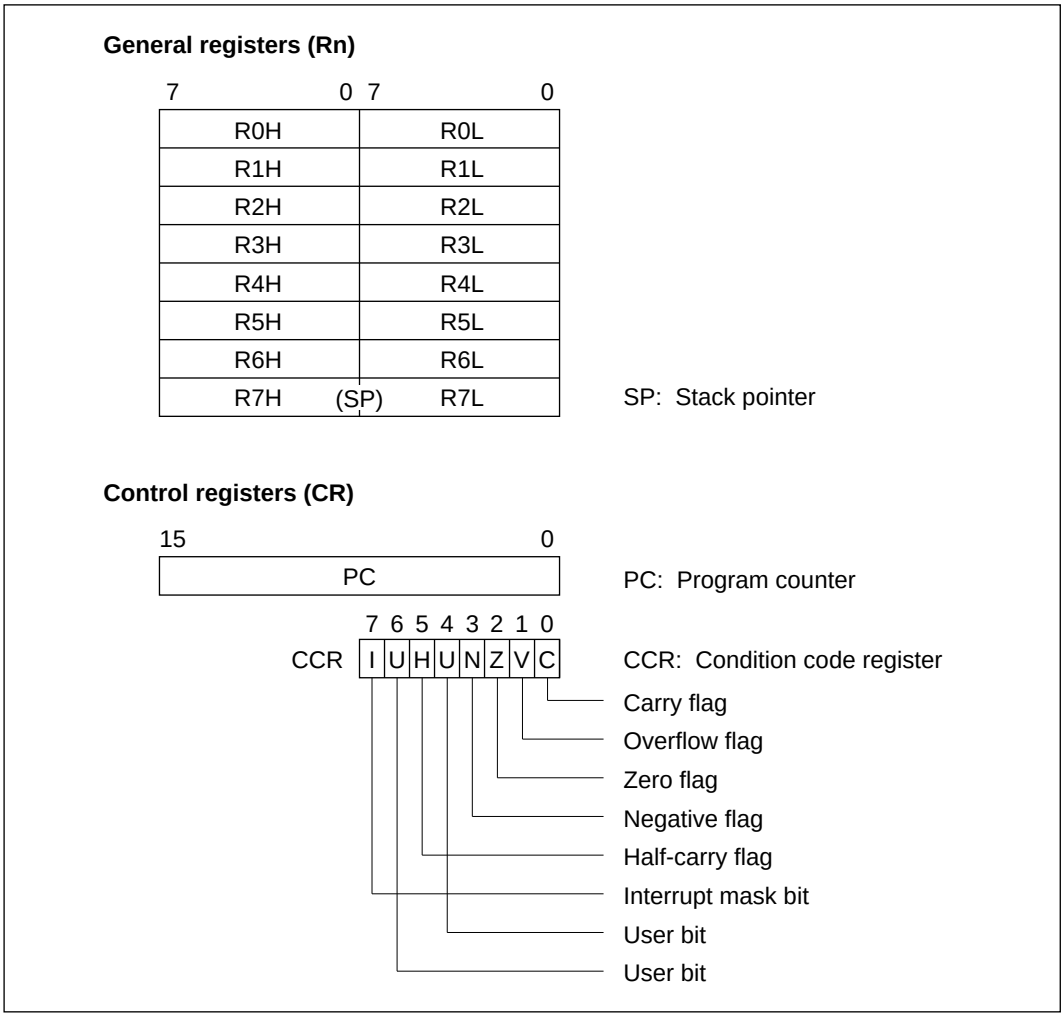


Figure 2-1 CPU Registers

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.

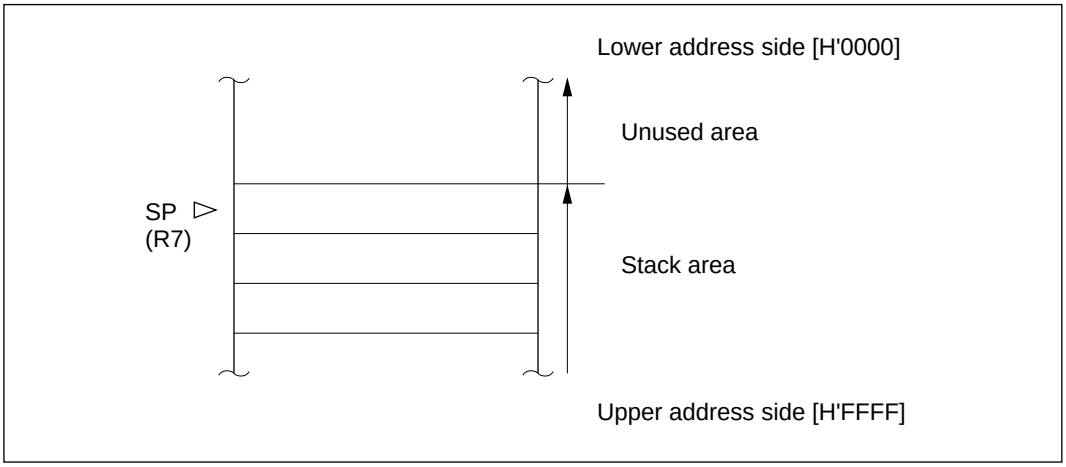


Figure 2-2 Stack Pointer

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, Interrupts.

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

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

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

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

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

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

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

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

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

- Add instructions, to indicate a carry
- Subtract instructions, to indicate a borrow
- Shift and rotate instructions, to store the value shifted out of the end bit

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

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

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

2.2.3 Initial Register Values

When the CPU is reset, the program counter (PC) is initialized to the value stored at address H'0000 in the vector table, and the I bit in the CCR is set to 1. The other CCR bits and the general registers are not initialized. In particular, the stack pointer (R7) is not initialized. To prevent program crashes the stack pointer should be initialized by software, by the first instruction executed after a reset.

2.3 Data Formats

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

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

2.3.1 Data Formats in General Registers

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

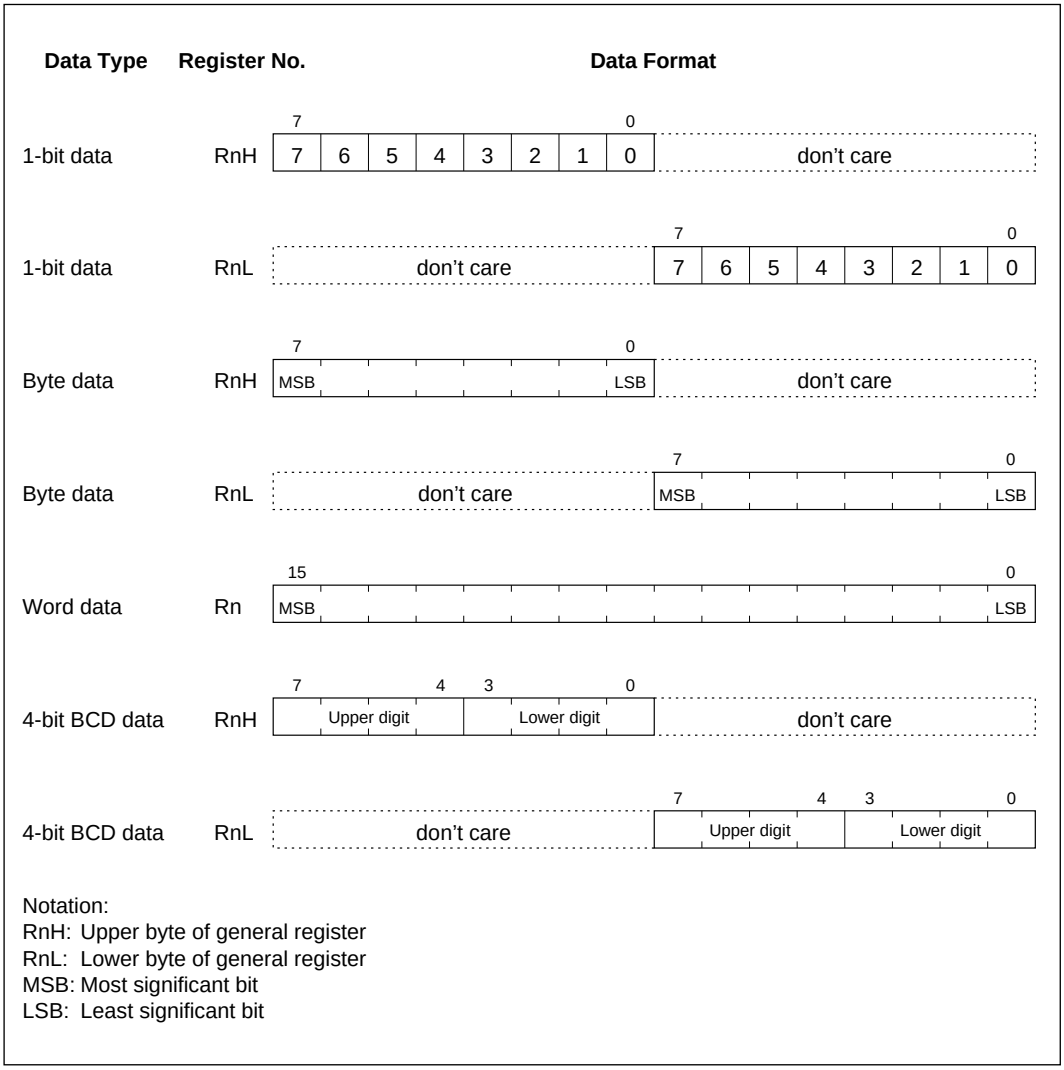


Figure 2-3 Register Data Formats

2.3.2 Memory Data Formats

Figure 2-4 indicates the data formats in memory. For access by the H8/300L CPU, word data stored in memory must always begin at an even address. In word access the least significant bit of the address is regarded as 0. If an odd address is specified, the access is performed at the preceding even address. This rule affects the MOV.W instruction, and also applies to instruction fetching.

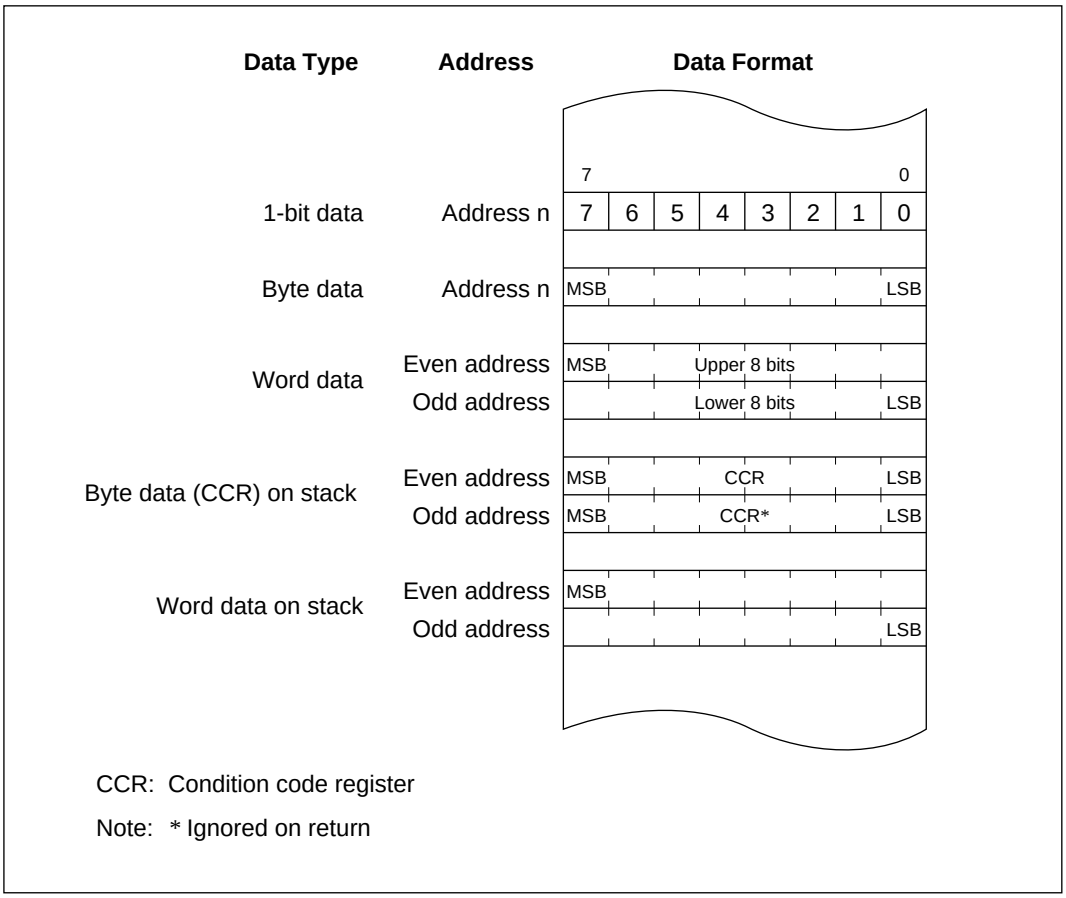


Figure 2-4 Memory Data Formats

When the stack is accessed using R7 as an address register, word access should always be performed. When the CCR is pushed on the stack, two identical copies of the CCR are pushed to make a complete word. When they are restored, the lower byte is ignored.

2.4 Addressing Modes

2.4.1 Addressing Modes

The H8/300L CPU supports the eight addressing modes listed in table 2-1. Each instruction uses a subset of these addressing modes.

Table 2-1 Addressing Modes

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

1. **Register Direct—Rn:** The register field of the instruction specifies an 8- or 16-bit general register containing the operand.

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

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

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

   This mode is used only in MOV instructions. For the MOV.W instruction, the resulting address must be even.

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

   - Register indirect with post-increment—@Rn+

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

     The register field of the instruction specifies a 16-bit general register containing the address of the operand. After the operand is accessed, the register is incremented by 1 for MOV.B or 2 for MOV.W. For MOV.W, the original contents of the 16-bit general register must be even.

   - Register indirect with pre-decrement—@–Rn

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

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

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

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

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

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

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

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

8. **Memory Indirect—@@aa:8:** This mode can be used by the JMP and JSR instructions. The second byte of the instruction code specifies an 8-bit absolute address. The word located at this address contains the branch destination address.

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

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

2.4.2 Effective Address Calculation

Table 2-2 shows how effective addresses are calculated in each of the addressing modes.

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

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

Bit manipulation instructions use register direct (1), register indirect (2), or absolute addressing (5) to specify a byte operand, and 3-bit immediate addressing (6) to specify a bit position in that byte. The BSET, BCLR, BNOT, and BTST instructions can also use register direct addressing (1) to specify the bit position.

Table 2-2 Effective Address Calculation

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

Table 2-2 Effective Address Calculation (cont)

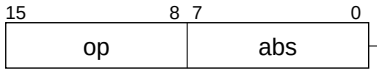
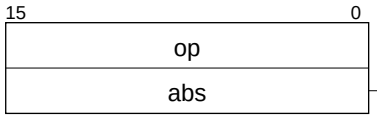
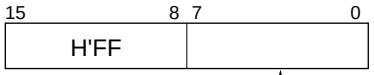
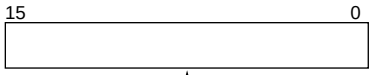
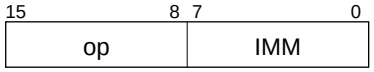
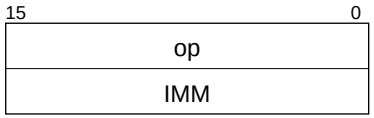
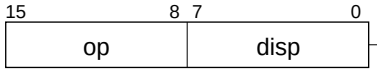
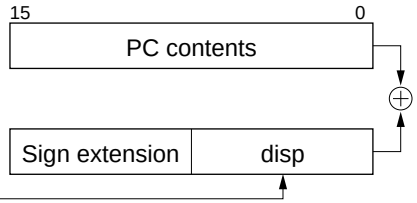
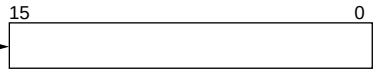
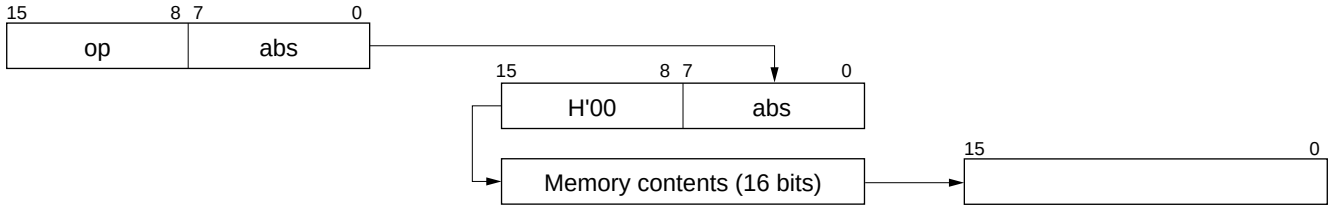
No.	Addressing Mode and Instruction Format	Effective Address Calculation Method	Effective Address (EA)
5	Absolute address @aa:8  @aa:16  		 
6	Immediate #xx:8  #xx:16  		Operand is 1- or 2-byte immediate data
7	Program-counter relative @(d:8, PC)  	 	

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		

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

2.5 Instruction Set

The H8/300L Series can use a total of 55 instructions, which are grouped by function in table 2-3.

Table 2-3 Instruction Set

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

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

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

Notation

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

2.5.1 Data Transfer Instructions

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

Table 2-4 Data Transfer Instructions

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

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.

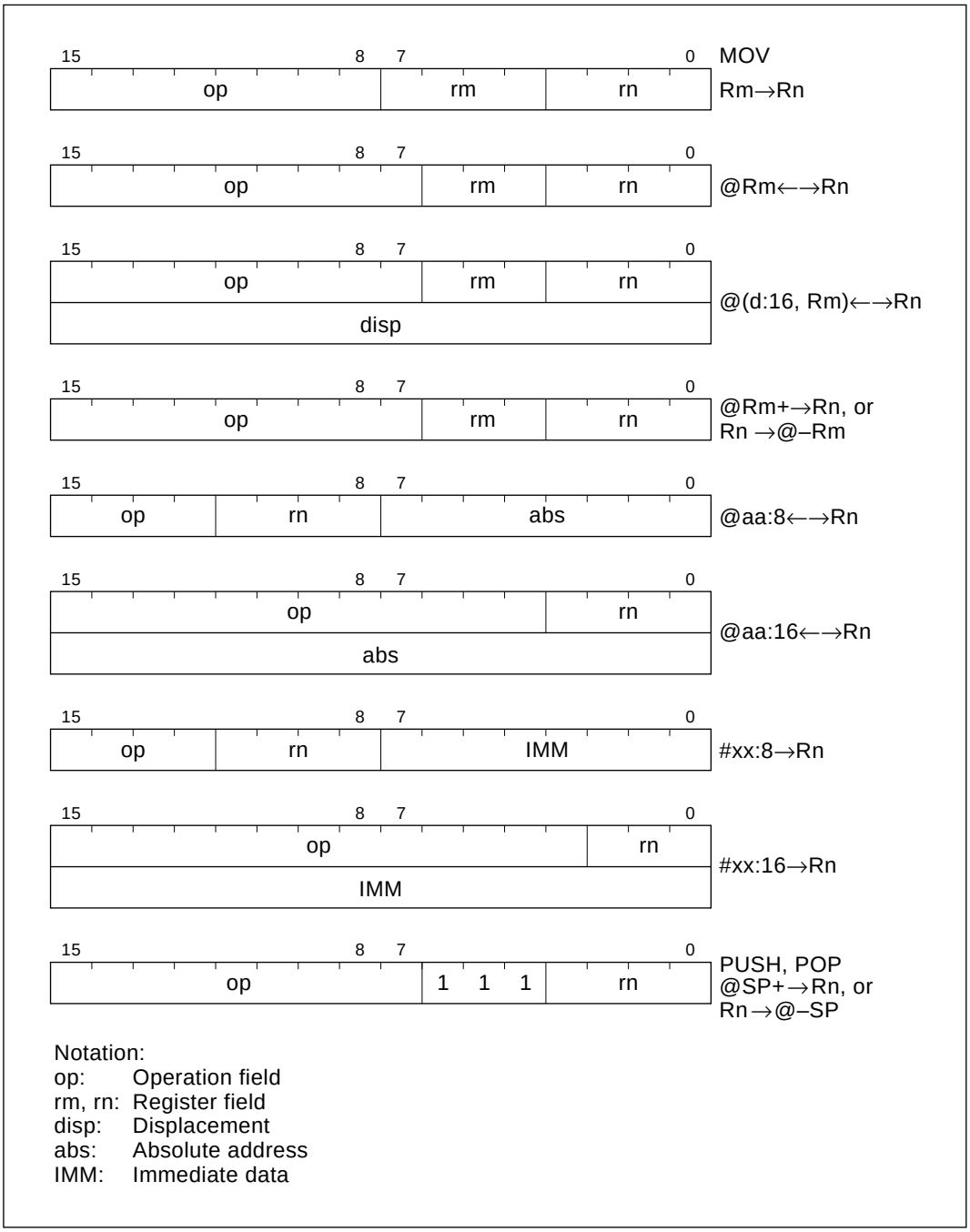


Figure 2-5 Data Transfer Instruction Codes

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 or borrow on byte data in two general registers, or addition or subtraction on immediate data and data in a general register.
INC DEC	B	Rd ± 1 → Rd Increments or decrements a general register
ADDS SUBS	W	Rd ± 1 → Rd, Rd ± 2 → Rd Adds or subtracts immediate data to or from data in a general register. The immediate data must be 1 or 2.
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 the result is stored in the CCR. Word data can be compared only between two general registers.
NEG	B	0 – Rd → Rd Obtains the two's complement (arithmetic complement) of data in a general register

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

2.5.3 Logic Operations

Table 2-6 describes the four instructions that perform logic operations.

Table 2-6 Logic Operation Instructions

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

Notes: * Size: Operand size
B: Byte

2.5.4 Shift Operations

Table 2-7 describes the eight shift instructions.

Table 2-7 Shift Instructions

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

Notes: * Size: Operand size
B: Byte

Figure 2-6 shows the instruction code format of arithmetic, logic, and shift instructions.

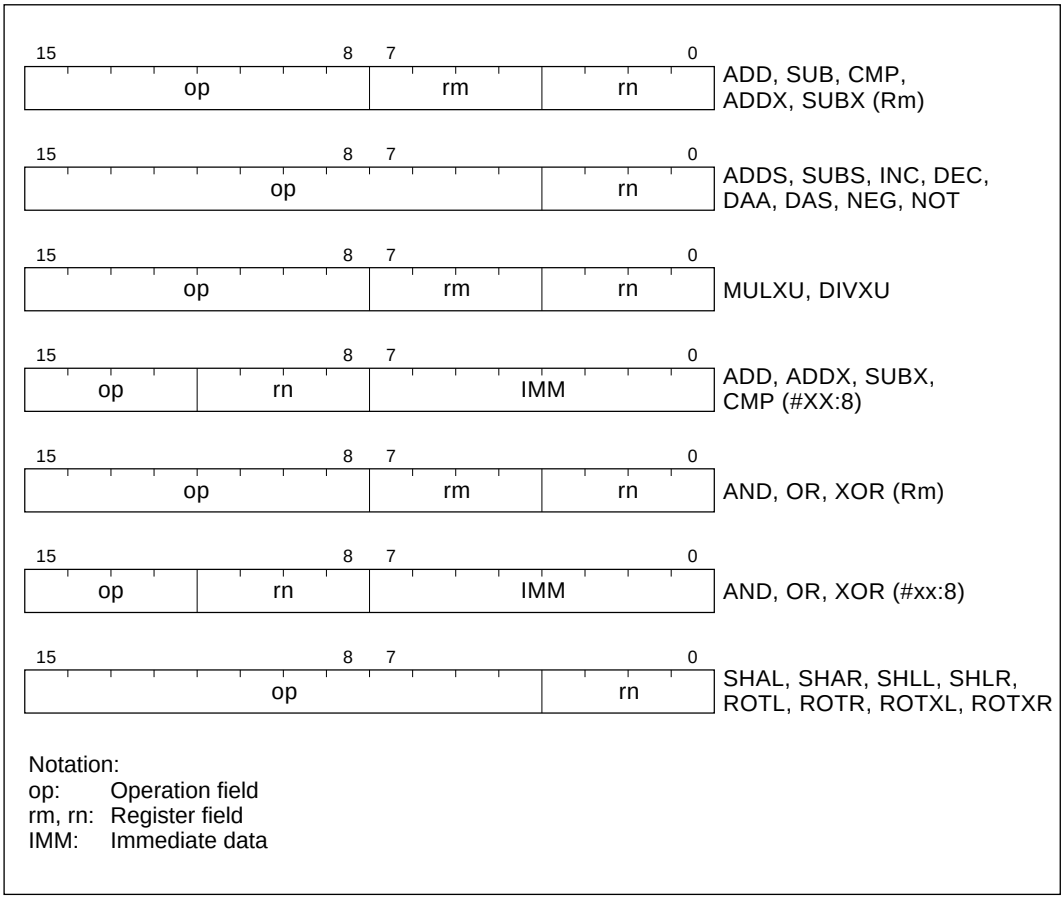


Figure 2-6 Arithmetic, Logic, and Shift Instruction Codes

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.

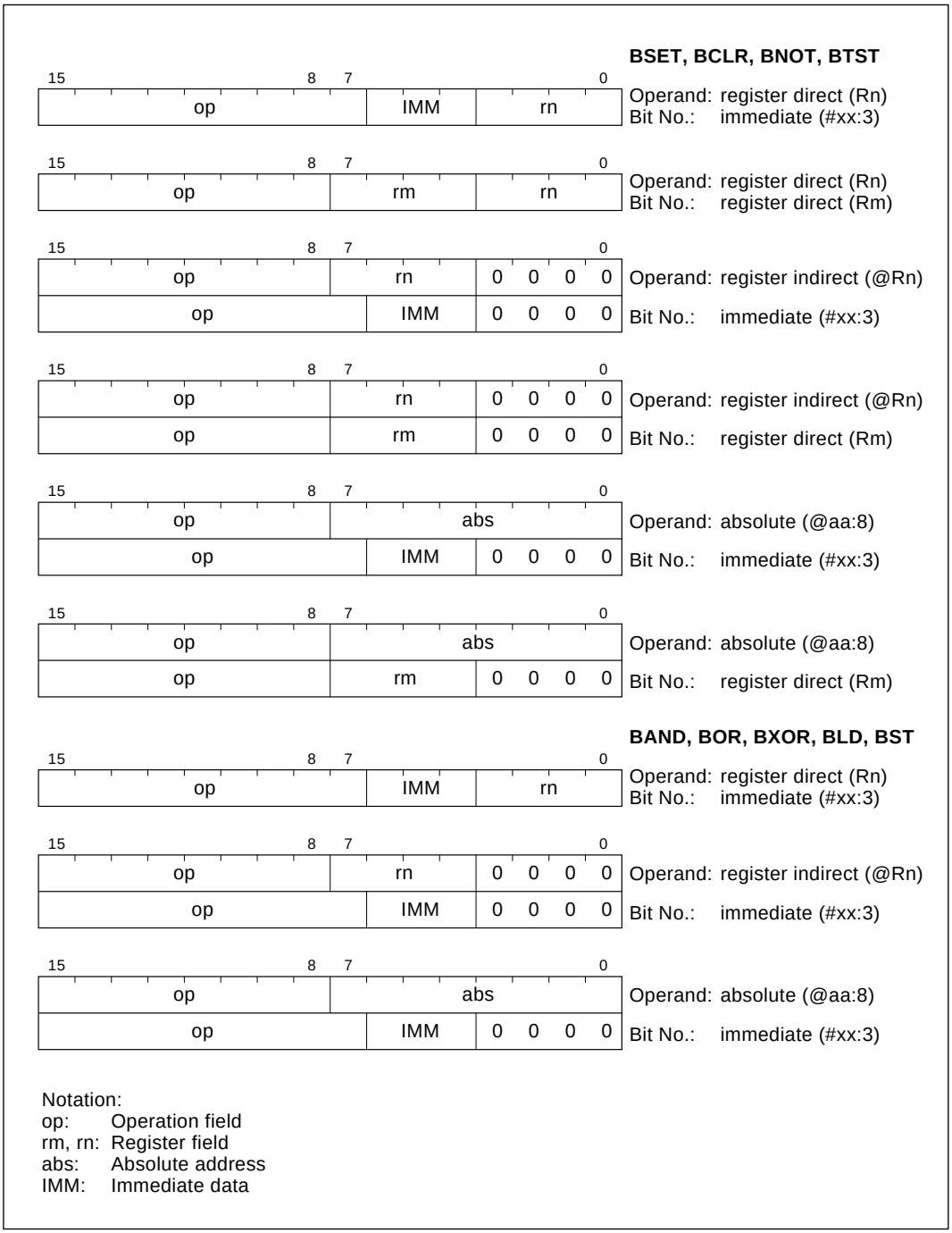


Figure 2-7 Bit Manipulation Instruction Codes

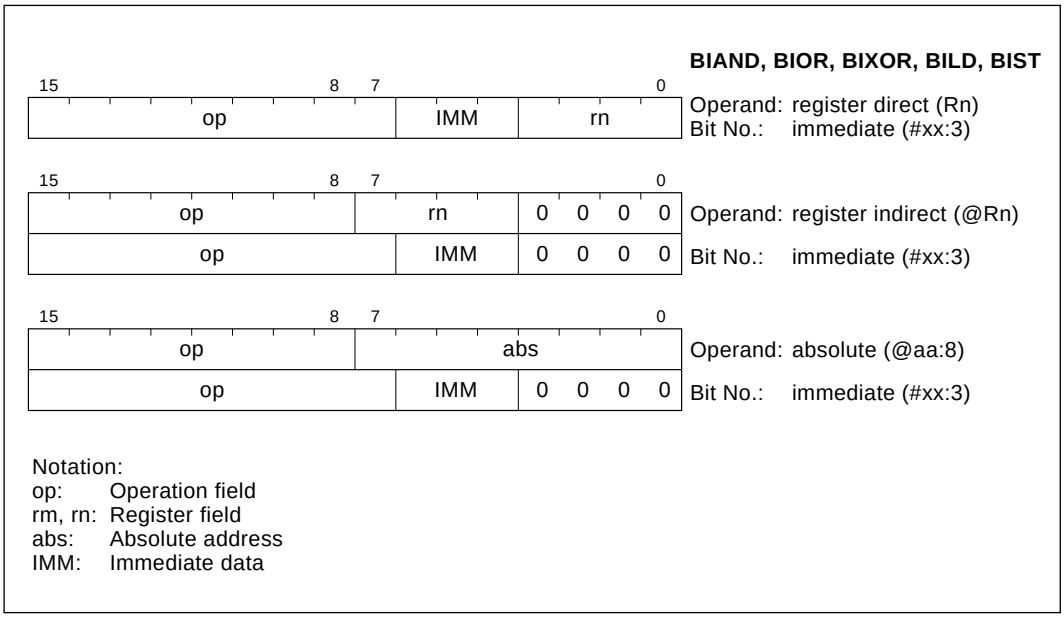


Figure 2-7 Bit Manipulation Instruction Codes (cont)

2.5.6 Branching Instructions

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

Table 2-9 Branching Instructions

Instruction	Size	Function
Bcc	—	Branches to the designated address if 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 displacement from the current address
JSR	—	Branches to a subroutine at a specified address
RTS	—	Returns from a subroutine

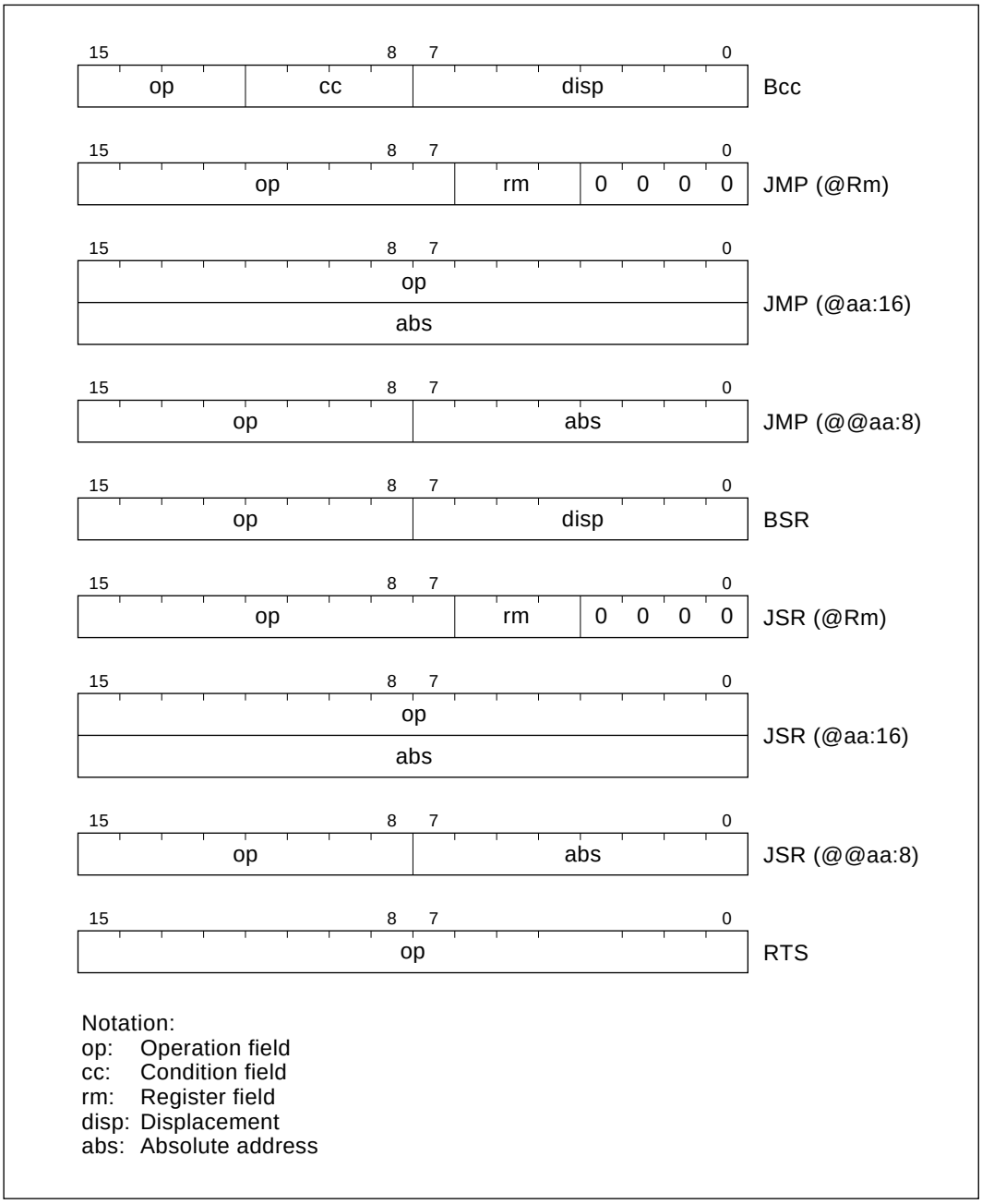


Figure 2-8 Branching Instruction Codes

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

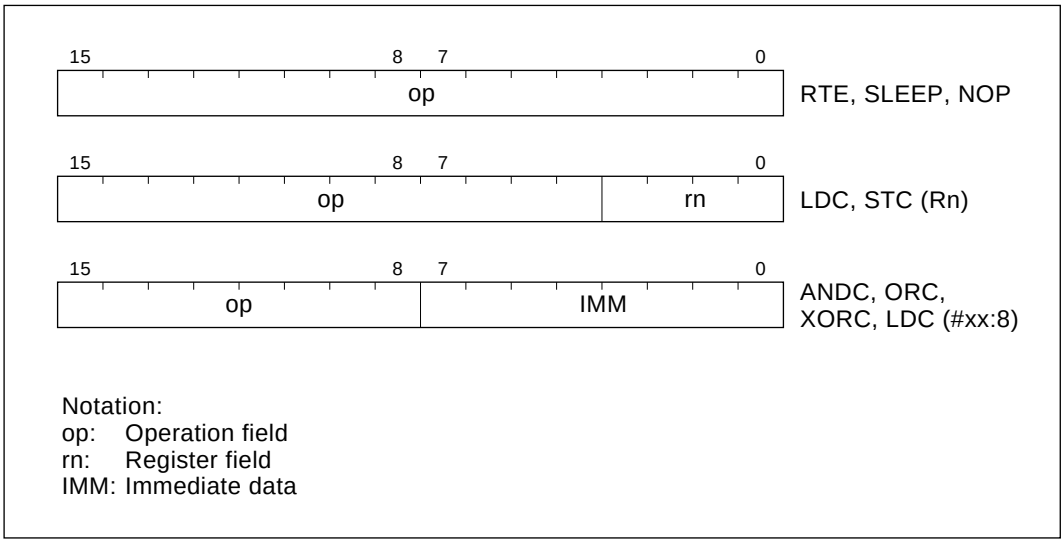


Figure 2-9 System Control Instruction Codes

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; Moves a data block according to parameters set in general registers R4L, R5, and R6. R4L: Size of block (bytes) R5: Starting source address R6: Starting destination address Execution of the next instruction starts as soon as the block transfer is completed.

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

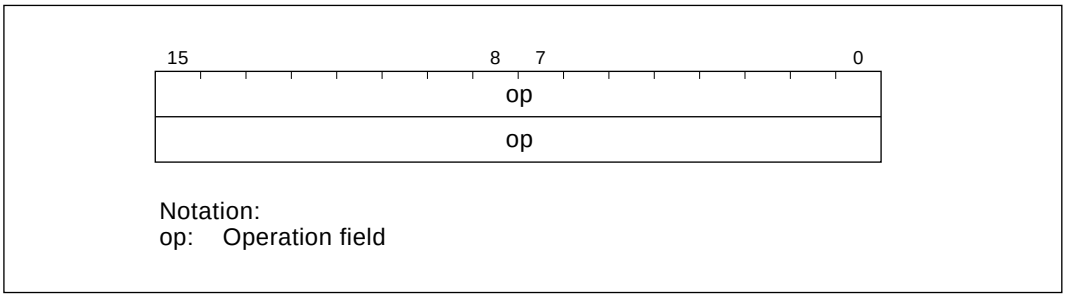


Figure 2-10 Block Data Transfer Instruction Code

2.6 Basic Operational Timing

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

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

Acess to on-chip memory takes place in two states. The data bus width is 16 bits, allowing access in byte or word size. Figure 2-11 shows the on-chip memory access cycle.

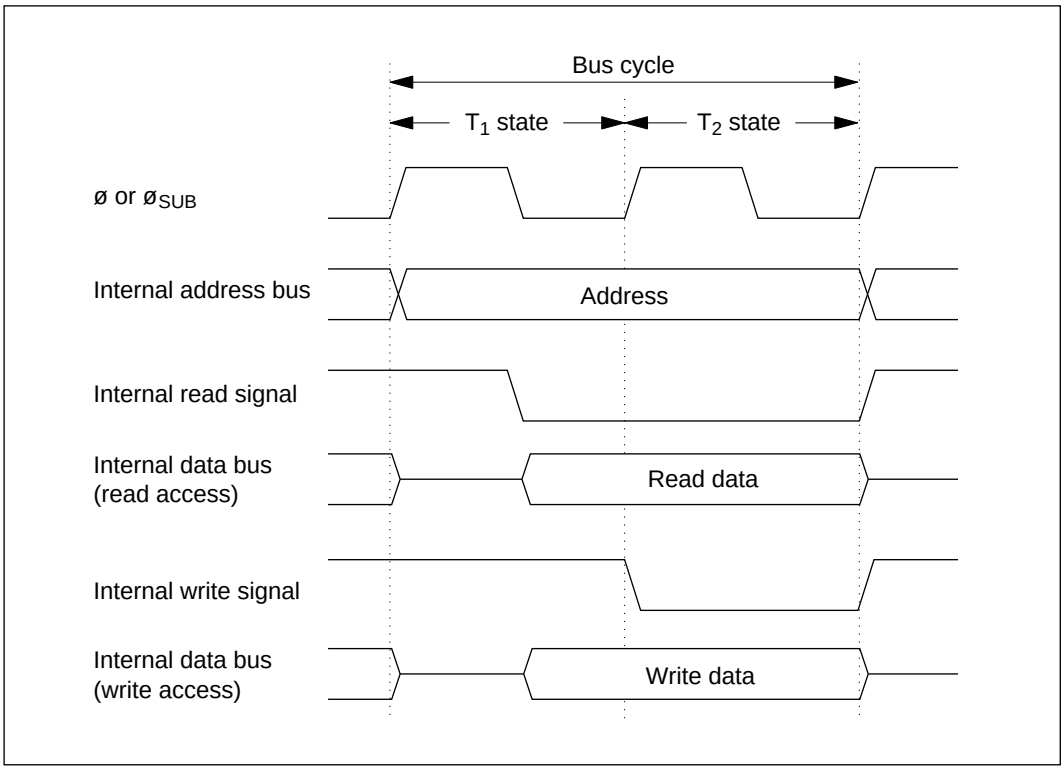


Figure 2-11 On-Chip Memory Access Cycle

2.6.2 Access to On-Chip Peripheral Modules

On-chip peripheral modules are accessed in two states or three states. The data bus width is 8 bits, so access is by byte size only. This means that for accessing word data, two instructions must be used. Figures 2-12 and 2-13 show the on-chip peripheral module access cycle.

Two-state access to on-chip peripheral modules

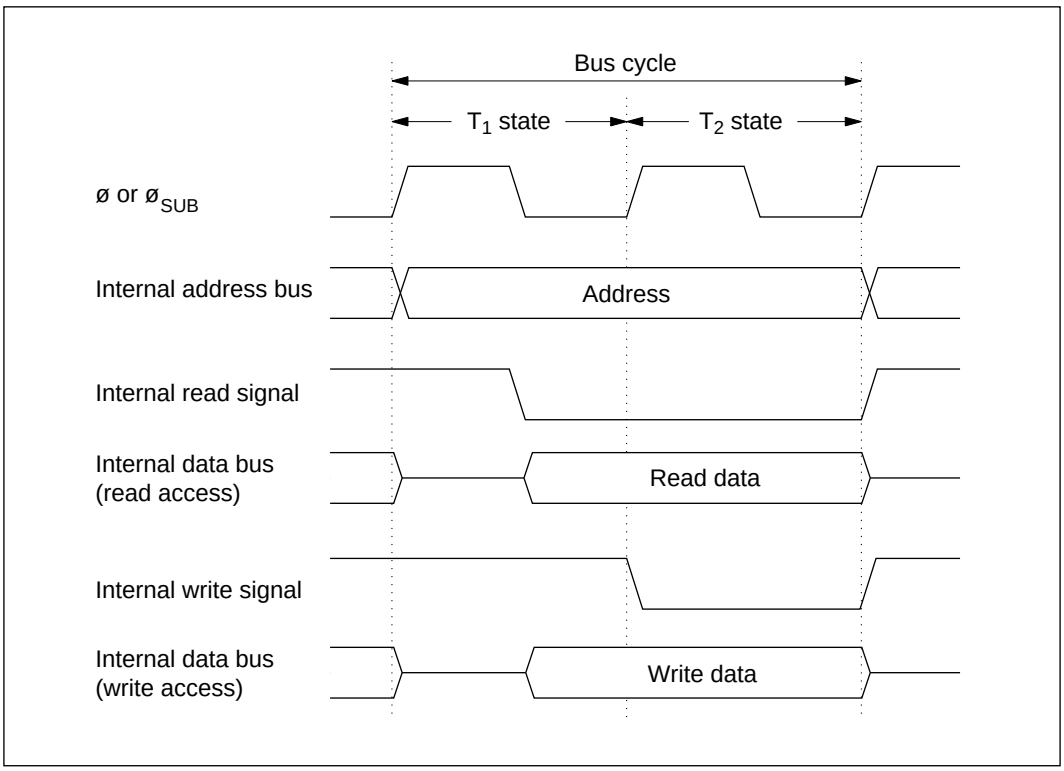


Figure 2-12 On-Chip Peripheral Module Access Cycle (2-State Access)

Three-state access to on-chip peripheral modules

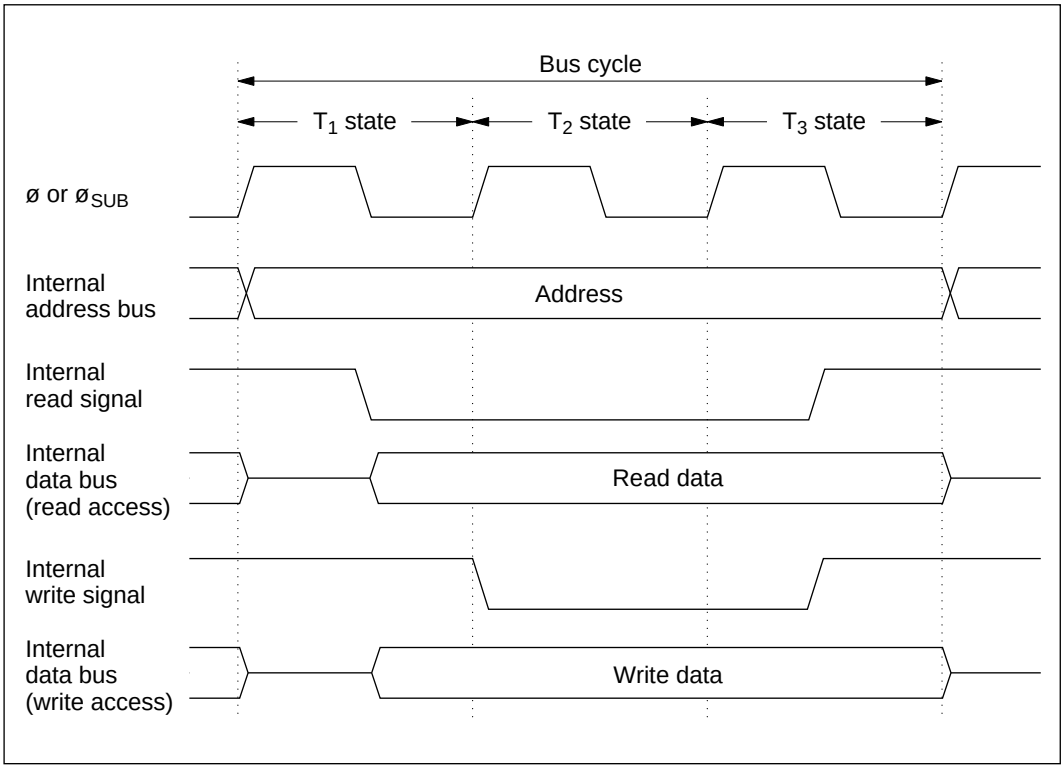


Figure 2-13 On-Chip Peripheral Module Access Cycle (3-State Access)

2.7 CPU States

2.7.1 Overview

There are four CPU states: the reset state, program execution state, program halt state, and exception-handling state. The program execution state includes active (high-speed or medium-speed) mode and subactive mode. In the program halt state there are a sleep mode, standby mode, watch mode, and subsleep mode. These states are shown in figure 2-14.
Figure 2-15 shows the state transitions.

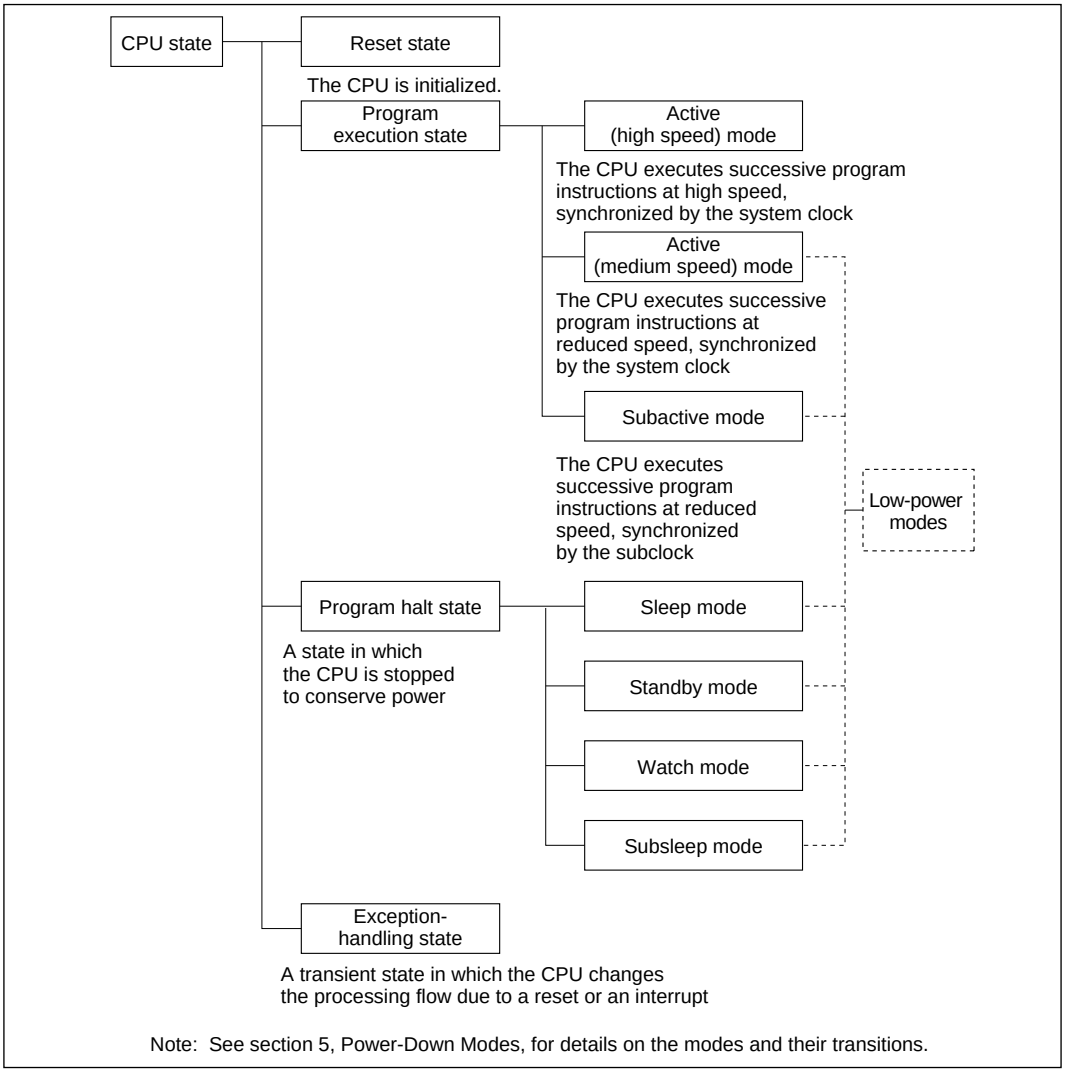


Figure 2-14 CPU Operation States

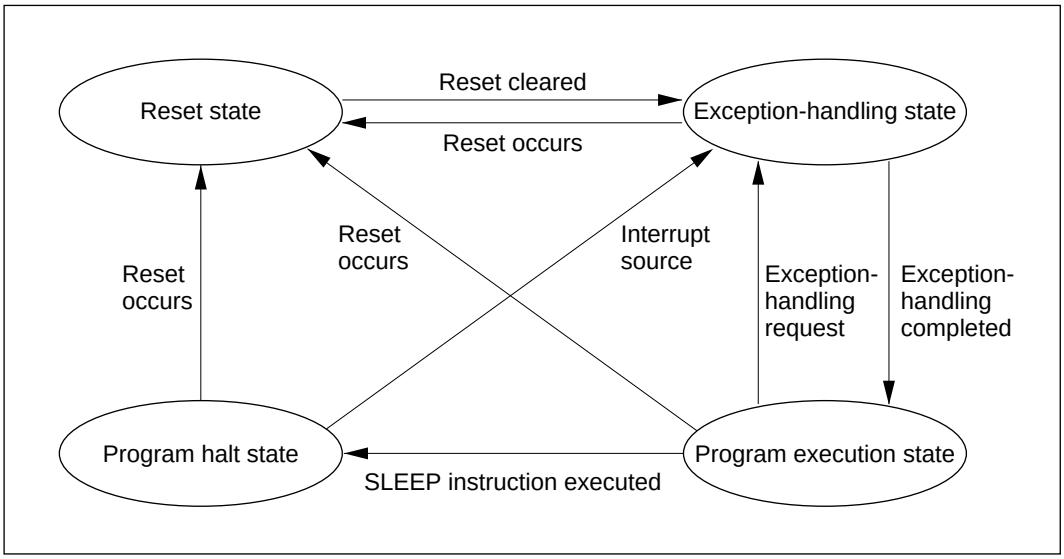


Figure 2-15 State Transitions

2.7.2 Program Execution State

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

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

2.7.3 Program Halt State

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

2.7.4 Exception-Handling State

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

For details on interrupt handling, see 3.3, Interrupts.

2.8 Memory Map

Figure 2-16 (a) shows the H8/3945 memory map. Figure 2-16 (b) shows the H8/3946 memory map. Figure 2-16 (c) shows the H8/3947 memory map.

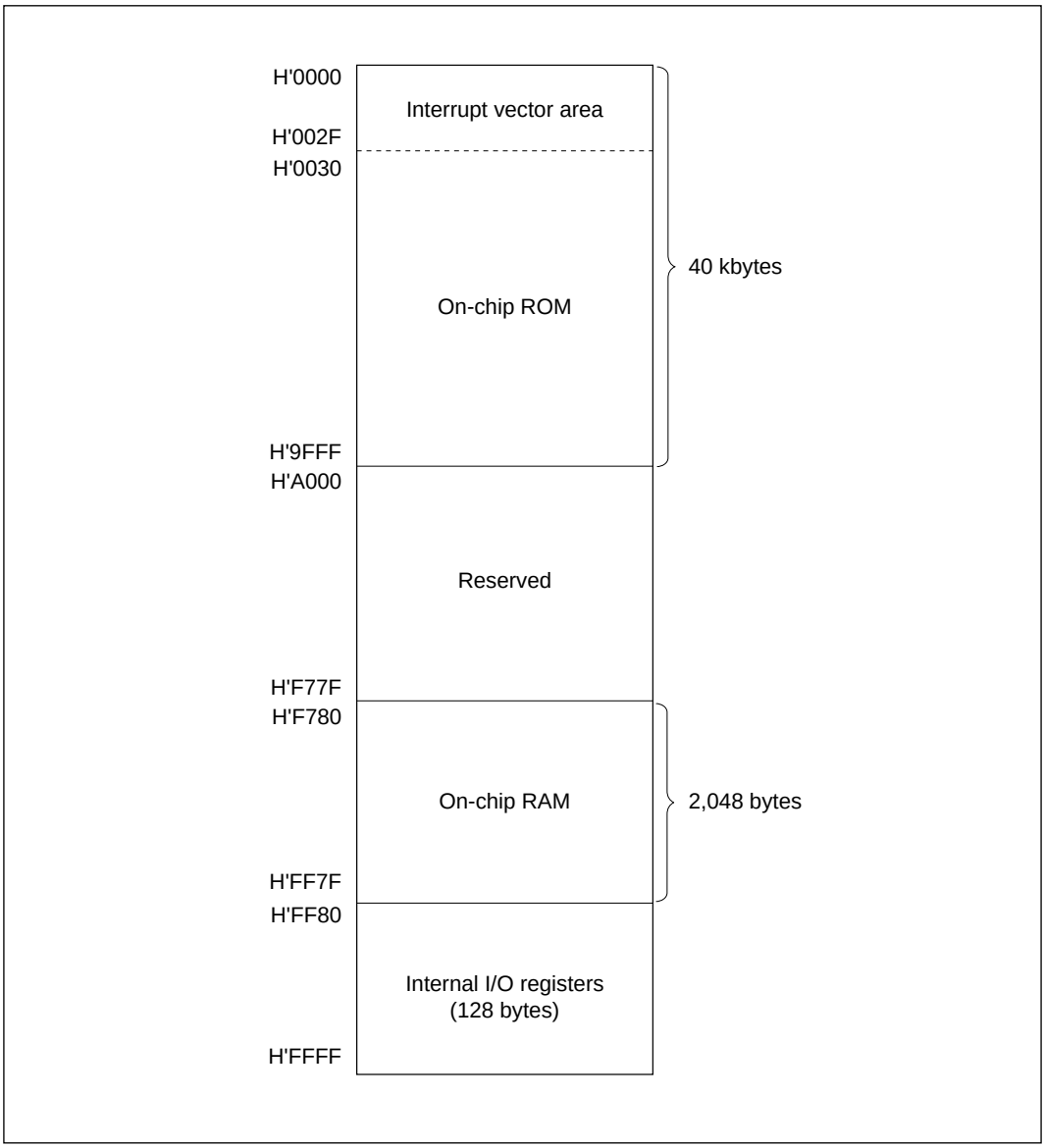


Figure 2-16 (a) H8/3945 Memory Map

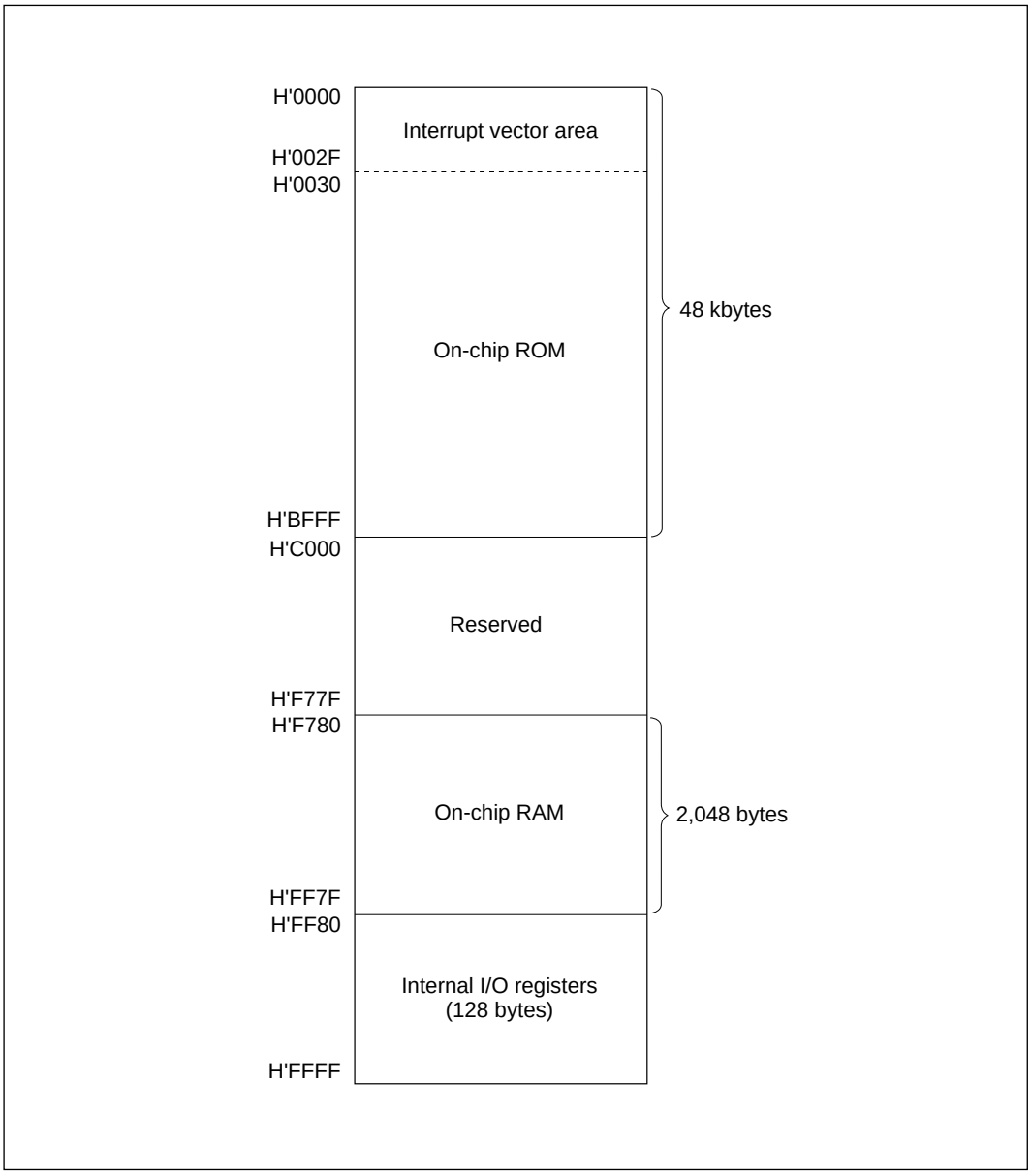


Figure 2-16 (b) H8/3946 Memory Map

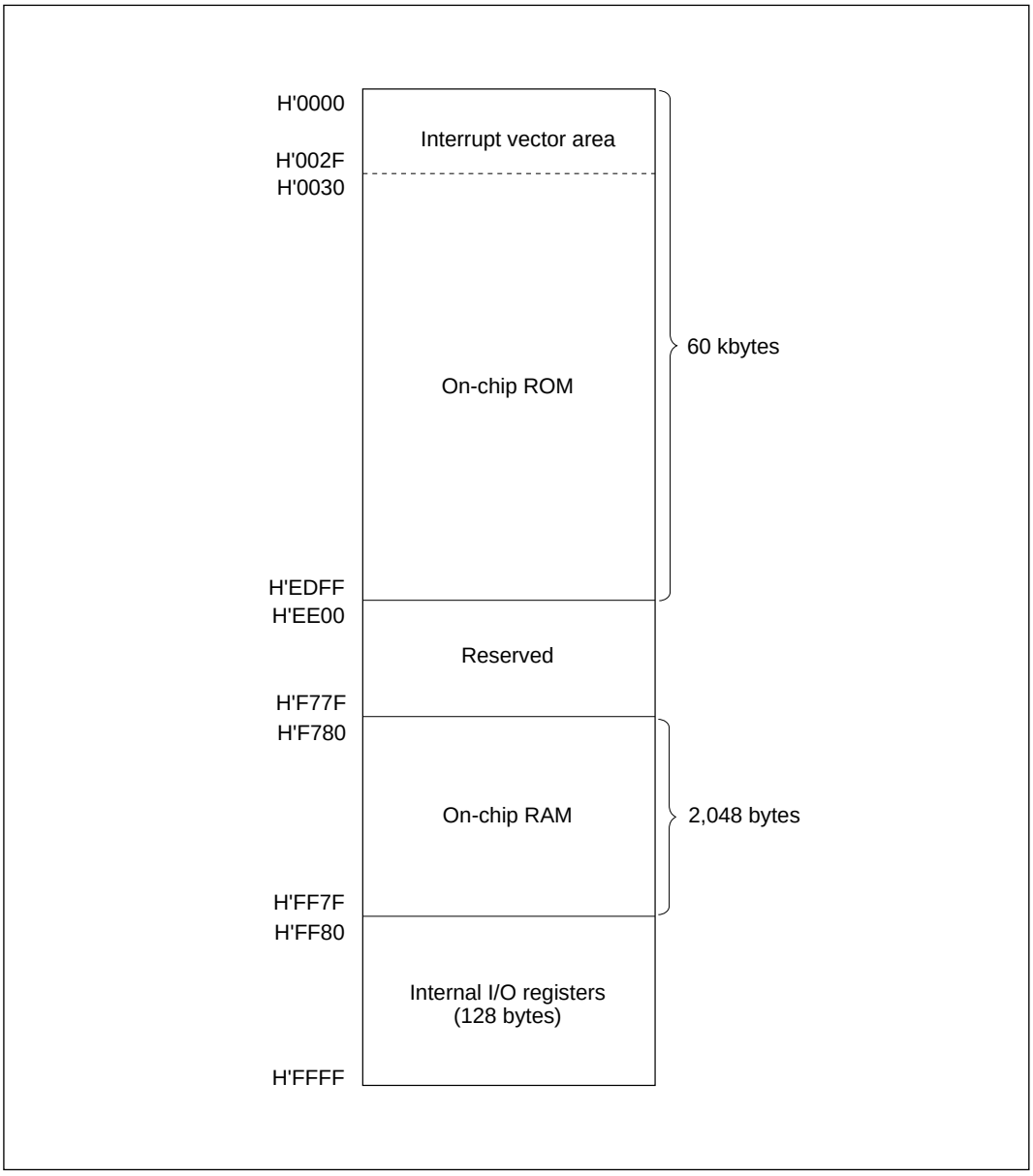


Figure 2-16 (c) H8/3947 Memory Map

2.9 Application Notes

2.9.1 Notes on Data Access

1. 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. 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.

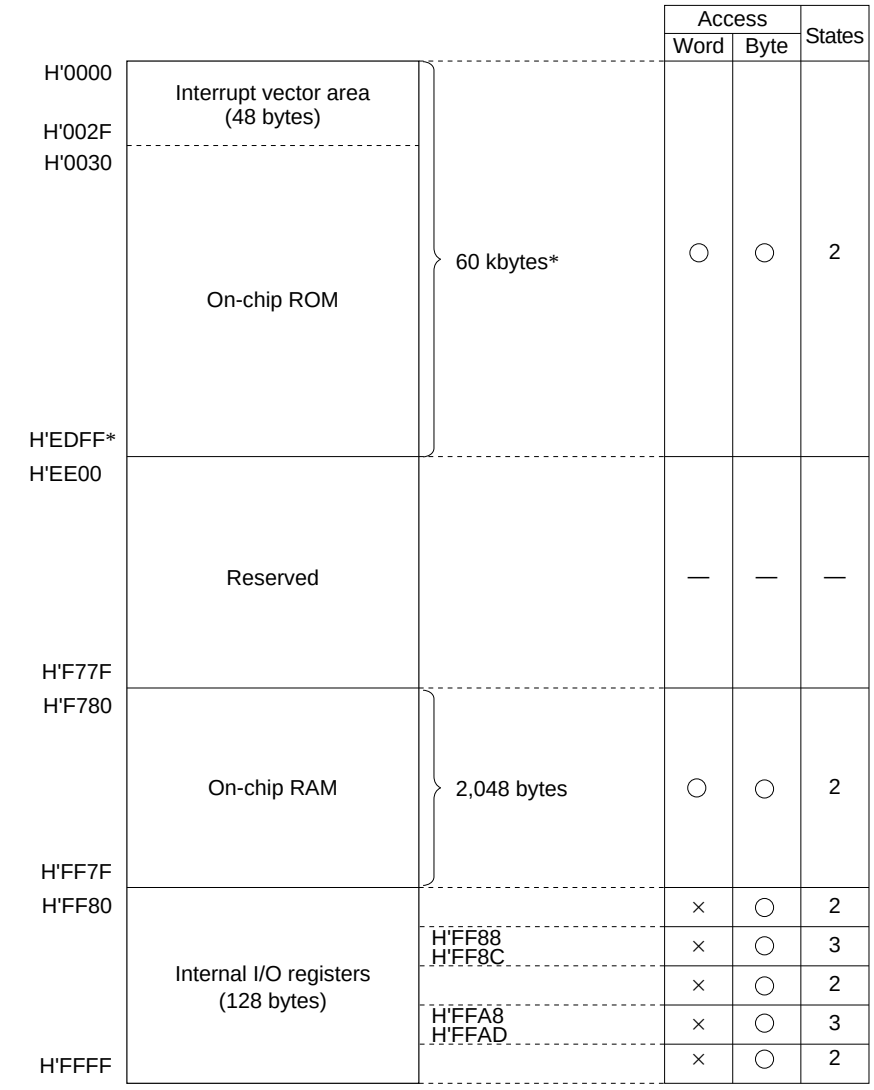
   Word access from CPU to I/O register area:

   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 H8/3947 is shown as an example.
* The H8/3945 ROM occupies 40 kbytes up to address H'9FFF. The H8/3946 ROM occupies 48 kbytes up to address H'BFFF.

Figure 2-17 Data Size and Number of States for Access to and from On-Chip Peripheral Modules

2.9.2 Notes on Bit Manipulation

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

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

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

Example 1

Figure 2-18 shows an example in which two timer registers share the same address. When a bit manipulation instruction accesses the timer load register and timer counter of a reloadable timer, since these two registers share the same address, the following operations take place.

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

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

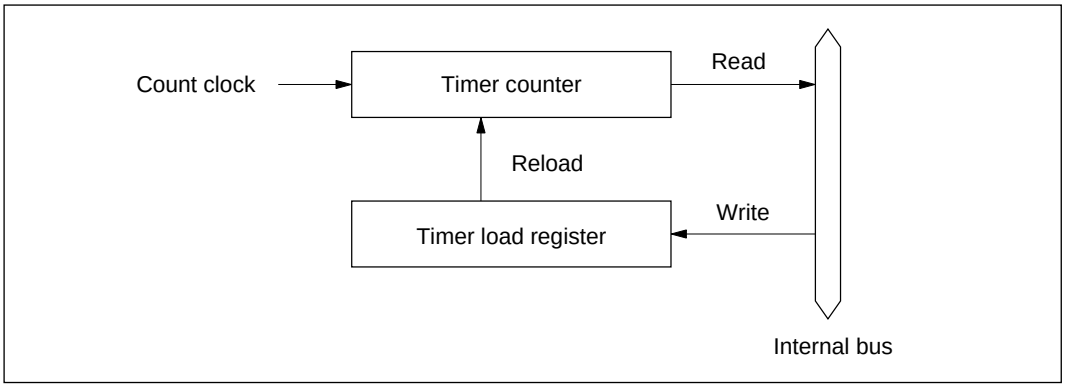


Figure 2-18 Timer Configuration Example

Example 2

Here a BSET instruction is executed designating port 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

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

Example 3

In this example, the port 3 control register PCR3 is accessed by a BCLR instruction.

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
Timer counter B1 and timer load register B1	TCB1/TLB1	H'FFB3
Timer counter B2 and timer load register B2	TCB2/TLB2	F'FF9D
Timer counter B3 and timer load register B3	TCB3/TLB3	H'FF9F
Timer counter C and timer load register C	TCC/TLC	H'FFB5
I^2C bus mode register 1 and slave address register 1	ICMR1/SAR1	H'FF83
I^2C bus mode register 2 and slave address register 2	ICMR2/SAR2	H'FF87
Port data register 1*	PDR1	H'FFD4
Port data register 2*	PDR2	H'FFD5
Port data register 3*	PDR3	H'FFD6
Port data register 4*	PDR4	H'FFD7
Port data register 5*	PDR5	H'FFD8
Port data register 6*	PDR6	H'FFD9
Port data register 7*	PDR7	H'FFDA
Port data register 8*	PDR8	H'FFDB
Port data register 9*	PDR9	H'FFDC
Port data register A*	PDRA	H'FFDD

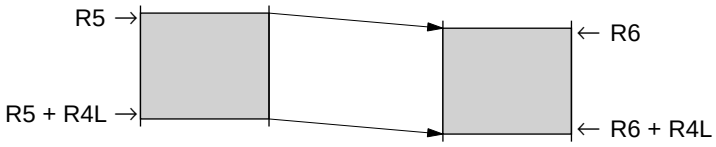
Note: * Port data registers have the same addresses as input pins.

Table 2-13 Registers with Write-Only Bits

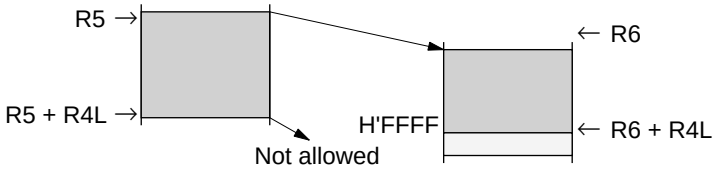
Register Name	Abbreviation	Address
Port control register 1	PCR1	H'FFE4
Port control register 2	PCR2	H'FFE5
Port control register 3	PCR3	H'FFE6
Port control register 4	PCR4	H'FFE7
Port control register 5	PCR5	H'FFE8
Port control register 6	PCR6	H'FFE9
Port control register 7	PCR7	H'FFEA
Port control register 8	PCR8	H'FFEB
Port control register 9	PCR9	H'FFEC
Port control register A	PCRA	H'FFED
Timer control register F	TCRF	H'FFB6

2.9.3 Notes on Use of the EEPMOV Instruction

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



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



Section 3 Exception Handling

3.1 Overview

Exception handling is performed in the H8/3947 Series when a reset or interrupt occurs. Table 3-1 shows the priorities of these two types of exception handling.

Table 3-1 Exception Handling Types and Priorities

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

3.2 Reset

3.2.1 Overview

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

3.2.2 Reset Sequence

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

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

- At power on: Hold the $\overline{\text{RES}}$ pin low until the clock pulse generator output stabilizes.
- Resetting during operation: Hold the $\overline{\text{RES}}$ pin low for at least 10 system clock cycles.

Reset exception handling takes place as follows.

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

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

Figure 3-1 shows the reset sequence.

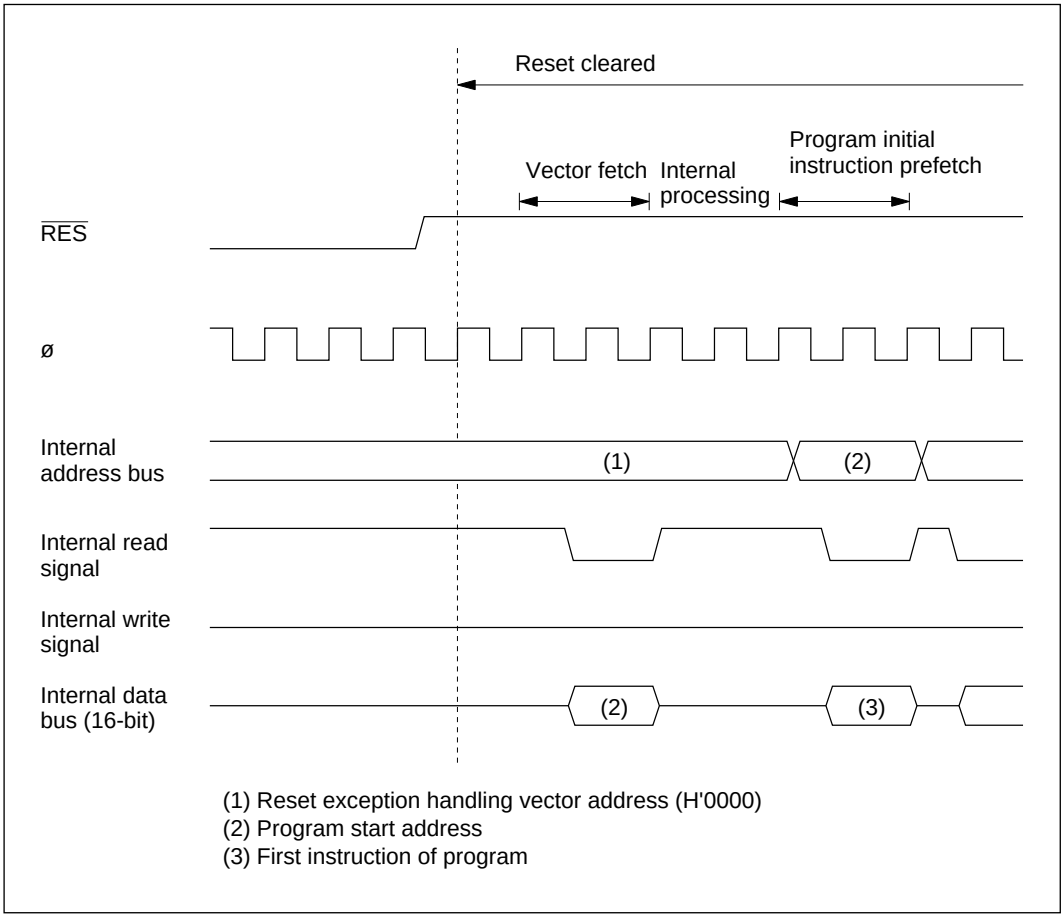


Figure 3-1 Reset Sequence

3.2.3 Interrupt Immediately after Reset

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

3.3 Interrupts

3.3.1 Overview

The interrupt sources include 14 external interrupts (NMI, IRQ_4 to IRQ_0, WKP_7 to WKP_0) and 24 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:

- Both 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 recognized.
- NMI and IRQ_4 to IRQ_0 can be set independently to either rising edge sensing or falling edge sensing.

Table 3-2 Interrupt Sources and Their Priorities

Interrupt Source		IInterrupt	Vector Number	Vector Address	Priority
External interrupts	$\overline{RES}$	Reset	0	H'0000 to H'0001	High
	$\overline{NMI}$	NMI	3	H'0006 to H'0007	
	$\overline{IRQ_0}$	IRQ_0	4	H'0008 to H'0009	
	$\overline{IRQ_1}$	IRQ_1	5	H'000A to H'000B	
	$\overline{IRQ_2}$	IRQ_2	6	H'000C to H'000D	
	$\overline{IRQ_3}$	IRQ_3	7	H'000E to H'000F	
	$\overline{IRQ_4}$	IRQ_4	8	H'0010 to H'0011	
	$\overline{WKP_0}$	Wakeup 0	9	H'0012 to H'0013	
	$\overline{WKP_1}$	Wakeup 1			
	$\overline{WKP_2}$	Wakeup 2			
	$\overline{WKP_3}$	Wakeup 3			
	$\overline{WKP_4}$	Wakeup 4			
	$\overline{WKP_5}$	Wakeup 5			
	$\overline{WKP_6}$	Wakeup 6			
	$\overline{WKP_7}$	Wakeup 7			Low

Table 3-2 Interrupt Sources and Priorities (cont)

Interrupt Source		IInterrupt	Vector Number	Vector Address	Priority
Internal interrupts	Timer B3	Timer B3 overflow	10	H'0014 to H'0015	High
	Timer A	Timer A overflow	11	H'0016 to H'0017	
	Timer B1	Timer B1 overflow	12	H'0018 to H'0019	
	Timer C	Timer C overflow or underflow	13	H'001A to H'001B	
	Timer FL	Timer FL compare match Timer FL overflow	14	H'001C to H'001D	
	Timer FH	Timer FH compare match Timer FH overflow	15	H'001E to H'001F	
	Timer G	Timer G input capture Timer G overflow	16	H'0020 to H'0021	
	Timer B2	Timer B2 overflow	17	H'0022 to H'0023	
	SCI3	SCI3 transmit end	18	H'0024 to H'0025	
		SCI3 transmit data empty			
		SCI3 receive data full			
		SCI3 overrun error			
		SCI3 framing error			
		SCI3 parity error			
	A/D converter	A/D conversion end	19	H'0026 to H'0027	
	(SLEEP instruction executed)	Direct transfer	20	H'0028 to H'0029	
	Timer H	Timer H compare match A	21	H'002A to H'002B	
		Timer H compare match B			
		Timer H overflow			
	I^2C1	Transfer end	22	H'002C to H'002D	
	I^2C2	Transfer end	23	H'002E to H'002F	Low

Note: * Vector addresses H'0002 to H'0005 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

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

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

1. IRQ edge select register (IEGR)

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

IEGR is an 8-bit read/write register, used to designate whether pins $\overline{\text{NMI}}$ and $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_4$ are set to rising edge sensing or falling edge sensing.

Bit 7: NMI edge select (NMIEG)

Bit 7 selects the input sensing of the $\overline{\text{NMI}}$ pin.

Bit 7 NMIEG	Description	
0	Falling edge of $\overline{\text{NMI}}$ pin input is detected	(initial value)
1	Rising edge of $\overline{\text{NMI}}$ pin input is detected	

Bits 6 and 5: Reserved bits

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

Bit 4: IRQ_4 edge select (IEG4)

Bit 4 selects the input sensing of pin $\overline{IRQ_4}/\overline{ADTRG}$.

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

Bit 3: IRQ_3 edge select (IEG3)

Bit 3 selects the input sensing of pin $\overline{IRQ_3}$/TMIF.

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

Bit 2: IRQ_2 edge select (IEG2)

Bit 2 selects the input sensing of pin $\overline{IRQ_2}$/TMIC.

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

Bit 1: IRQ_1 edge select (IEG1)

Bit 1 selects the input sensing of pin $\overline{IRQ_1}$/TMIB1.

Bit 1 IEG1	Description	
0	Falling edge of $\overline{IRQ_1}$/TMIB1 pin input is detected	(initial value)
1	Rising edge of $\overline{IRQ_1}$/TMIB1 pin input is detected	

Bit 0: IRQ_0 edge select (IEG0)

Bit 0 selects the input sensing of pin $\overline{IRQ_0}$.

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

2. Interrupt enable register 1 (IENR1)

Bit	7	6	5	4	3	2	1	0
	IENTA	IENTB3	IENWP	IEN4	IEN3	IEN2	IEN1	IEN0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

Bit 7: Timer A interrupt enable (IENTA)

Bit 7 enables or disables timer A overflow interrupt requests.

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

Bit 6: Timer B3 interrupt enable (IENTB3)

Bit 6 enables or disables timer B3 overflow interrupt requests.

Bit 6 IENTB3	**Description**	
0	Disables timer B3 interrupts	(initial value)
1	Enables timer B3 interrupts	

Bit 5: Wakeup interrupt enable (IENWP)

Bit 5 enables or disables WKP_7 to WKP_0 interrupt requests.

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

Bits 4 to 0: IRQ_4 to IRQ_0 interrupt enable (IEN4 to IEN0)

Bits 4 to 0 enable or disable IRQ_4 to IRQ_0 interrupt requests.

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

3. Interrupt enable register 2 (IENR2)

Bit	7	6	5	4	3	2	1	0
	IENDT	IENAD	IENTB2	IENTG	IENTFH	IENTFL	IENTC	IENTB1
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

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

Bit 7: Direct transfer interrupt enable (IENDT)

Bit 7 enables or disables direct transfer interrupt requests.

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

Bit 6: A/D converter interrupt enable (IENAD)

Bit 6 enables or disables A/D converter interrupt requests.

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

Bit 5: Timer B2 interrupt enable (IENTB2)

Bit 5 enables or disables timer B2 overflow interrupt requests.

Bit 5 IENTB2	Description	
0	Disables timer B2 interrupts	(initial value)
1	Enables timer B2 interrupts	

Bit 4: Timer G interrupt enable (IENTG)

Bit 4 enables or disables timer G input capture and overflow interrupt requests.

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

Bit 3: Timer FH interrupt enable (IENTFH)

Bit 3 enables or disables timer FH compare match and overflow interrupt requests.

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

Bit 2: Timer FL interrupt enable (IENTFL)

Bit 2 enables or disables timer FL compare match and overflow interrupt requests.

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

Bit 1: Timer C interrupt enable (IENTC)

Bit 1 enables or disables timer C overflow or underflow interrupt requests.

Bit 1 IENTC	Description	
0	Disables timer C interrupts	(initial value)
1	Enables timer C interrupts	

Bit 0: Timer B interrupt enable (IENTB1)

Bit 0 enables or disables timer B1 overflow interrupt requests.

Bit 0 IENTB1	Description	
0	Disables timer B1 interrupts	(initial value)
1	Enables timer B1 interrupts	

SCI3 interrupt control is covered in 10.2.2, in the description of serial control register 3 (SCR3). I^2C1 and I^2C2 interrupt control is covered in 10.3, I^2C Bus Interface. Timer H interrupt control is described in 9.9, Timer H.

4. Interrupt request register 1 (IRR1)

Bit	7	6	5	4	3	2	1	0
	IRRTA	IRRTB3	—	IRRI4	IRRI3	IRRI2	IRRI1	IRRI0
Initial value	0	0	1	0	0	0	0	0
Read/Write	R/W*	R/W*	—	R/W*	R/W*	R/W*	R/W*	R/W*

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

IRR1 is an 8-bit read/write register, in which the corresponding bit is set to 1 when a timer A, timer B3, or IRQ_4 to IRQ_0 interrupt is requested. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit 7: Timer A interrupt request flag (IRRTA)

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

Bit 6: Timer B3 interrupt request flag (IRRTB3)

Bit 6 IRRTB3	Description	
0	Clearing conditions: When IRRTB3 = 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When the timer B3 counter value overflows from H'FF to H'00	

Bit 5: Reserved bit

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

Bits 4 to 0: IRQ_4 to IRQ_0 interrupt request flags (IRRI4 to IRRI0)

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

(n = 4 to 0)

5. Interrupt request register 2 (IRR2)

Bit	7	6	5	4	3	2	1	0
	IRRDT	IRRAD	IRRTB2	IRRTG	IRRTFH	IRRTFL	IRRTC	IRRTB1
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.

IRR2 is an 8-bit read/write register, in which the corresponding bit is set to 1 when a direct transfer, A/D converter, timer B2, timer G, timer FH, timer FL, timer C, or timer B1 interrupt is requested. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit 7: Direct transfer interrupt request flag (IRRDT)

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

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

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

Bit 5: Timer B2 interrupt request flag (IRRTB2)

Bit 5 IRRTB2	Description	
0	Clearing conditions: When IRRTB2 = 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When the timer B2 counter value overflows from H'FF to H'00	

Bit 4: Timer G interrupt request flag (IRRTG)

Bit 4 IRRTG	Description	
0	Clearing conditions: When IRRTG = 1, it is cleared by writing 0	(initial value)
1	Setting conditions: When pin TMIG is set to TMIG input and the designated signal edge is detected	

Bit 3: Timer FH interrupt request flag (IRRTFH)

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

Bit 2: Timer FL interrupt request flag (IRRTFL)

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

Bit 1: Timer C interrupt request flag (IRRTC)

Bit 1 IRRTC	Description
0	Clearing conditions: (initial value) When IRRTC = 1, it is cleared by writing 0
1	Setting conditions: When the timer C counter value overflows (goes from H'FF to H'00) or underflows (goes from H'00 to H'FF)

Bit 0: Timer B1 interrupt request flag (IRRTB1)

Bit 0 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 (goes from H'FF to H'00)

6. Wakeup interrupt request register (IWPR)

Bit	7	6	5	4	3	2	1	0
	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*

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

IWPR is an 8-bit read/write register, in which the corresponding bit is set to 1 when pins WKP_7 to WKP_0 are set to wakeup input and a pin receives a falling edge input. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

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

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

(n = 7 to 0)

3.3.3 External Interrupts

There are 14 external interrupts: NMI, IRQ_4 to IRQ_0, and WKP_7 to WKP_0.

1. NMI interrupt

The NMI interrupt is requested by input at the $\overline{NMI}$ pin. This interrupt can be detected by either rising edge sensing or falling edge sensing, depending on the setting of bit NMIEG in IEGR. The NMI interrupt has highest priority and is always accepted, regardless of the setting of the I bit in CCR. The NMI interrupt has exception vector 3. When NMI exception handling is initiated, the I bit is set to 1 in CCR.

2. Interrupts WKP_0 to WKP_7

Interrupts WKP_0 to WKP_7 are requested by falling edge inputs at pins $\overline{WKP}_0$ to $\overline{WKP}_7$. When these pins are designated as $\overline{WKP}_0$ to $\overline{WKP}_7$ pins in port mode register 5 (PMR5) and falling edge input is detected, the corresponding bit in the wakeup interrupt request register (IWPR) is set to 1, requesting an interrupt. Wakeup interrupt requests can be disabled by clearing the IENWP bit in IENR1 to 0. It is also possible to mask all interrupts by setting the CCR I bit to 1.

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

3. Interrupts IRQ_0 to IRQ_4

Interrupts IRQ_0 to IRQ_4 are requested by into pins inputs to $\overline{IRQ}_0$ to $\overline{IRQ}_4$. These interrupts are detected by either rising edge sensing or falling edge sensing, depending on the settings of bits IEG0 to IEG4 in the edge select register (IEGR).

When these pins are designated as pins $\overline{IRQ}_0$ to $\overline{IRQ}_4$ in port mode registers 1 and 2 (PMR1 and PMR2) and the designated edge is input, the corresponding bit in IRR1 is set to 1, requesting an interrupt. Interrupts IRQ_0 to IRQ_4 can be disabled by clearing bits IEN0 to IEN4 in IENR1 to 0. All interrupts can be masked by setting the I bit in CCR to 1.

When IRQ_0 to IRQ_4 interrupt exception handling is initiated, the I bit is set to 1. Vector numbers 4 to 8 are assigned to interrupts IRQ_0 to IRQ_4. The order of priority is from IRQ_0 (high) to IRQ_4 (low). Table 3-2 gives details.

3.3.4 Internal Interrupts

There are 24 internal interrupts that can be requested by the on-chip peripheral modules. When a peripheral module requests an interrupt, the corresponding bit in IRR1 or IRR2 is set to 1. Individual interrupt requests can be disabled by clearing the corresponding bit in IENR1 or IENR2 to 0. All interrupts can be masked by setting the I bit in CCR to 1. When an internal interrupt request is accepted, the I bit is set to 1. Vector numbers 10 to 23 are assigned to these interrupts. Table 3-2 shows the order of priority of interrupts from on-chip peripheral modules.

3.3.5 Interrupt Operations

Interrupts are controlled by an interrupt controller. Figure 3-2 shows a block diagram of the interrupt controller. Figure 3-3 shows the flow up to interrupt acceptance.

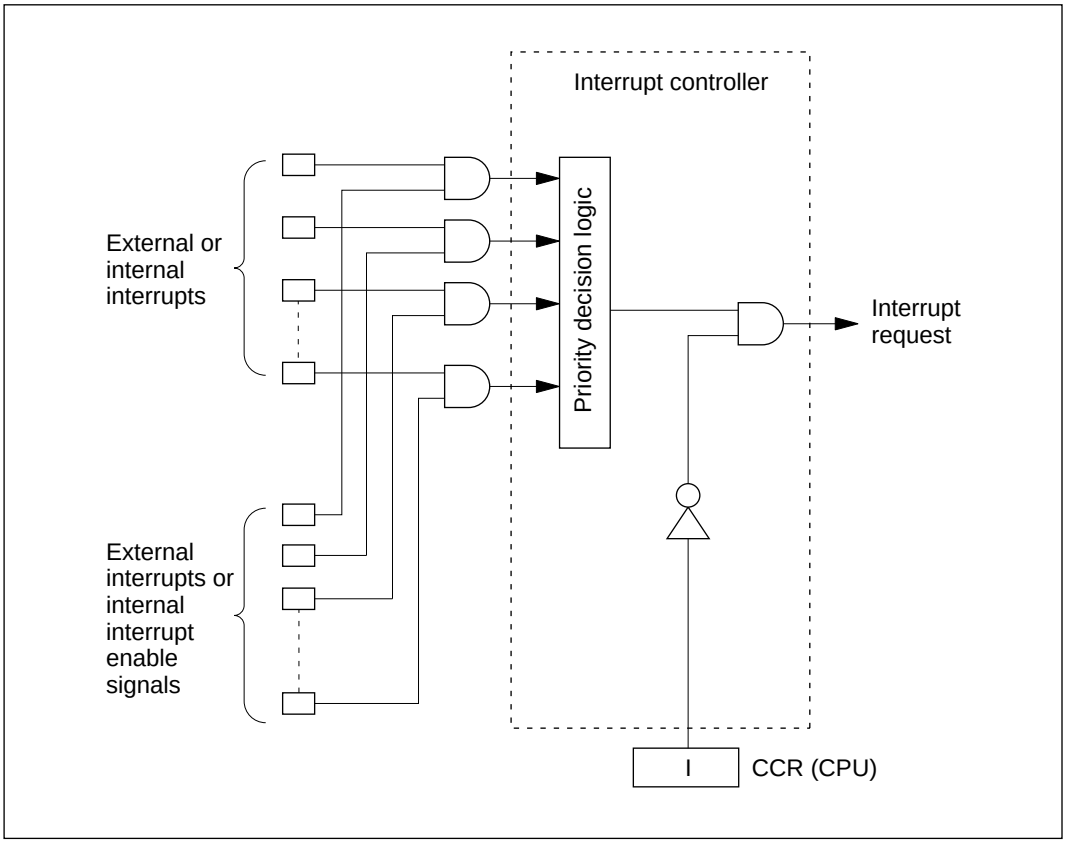


Figure 3-2 Block Diagram of Interrupt Controller

Interrupt operation is described as follows.

- When an interrupt condition is met while the interrupt enable register bit is set to 1, an interrupt request signal is sent to the interrupt controller.
- When the interrupt controller receives an interrupt request, it sets the interrupt request flag.
- From among the interrupts with interrupt request flags set to 1, the interrupt controller selects the interrupt request with the highest priority and holds the others pending. (Refer to table 3-2 for a list of interrupt priorities.)

- The interrupt controller checks the I bit of CCR. If the I bit is 0, the selected interrupt request is accepted; if the I bit is 1, the interrupt request is held pending.
- If the interrupt is accepted, after processing of the current instruction is completed, both PC and CCR are pushed onto the stack. The state of the stack at this time is shown in figure 3-4. The PC value pushed onto the stack is the address of the first instruction to be executed upon return from interrupt handling.
- The I bit of CCR is set to 1, masking all 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.

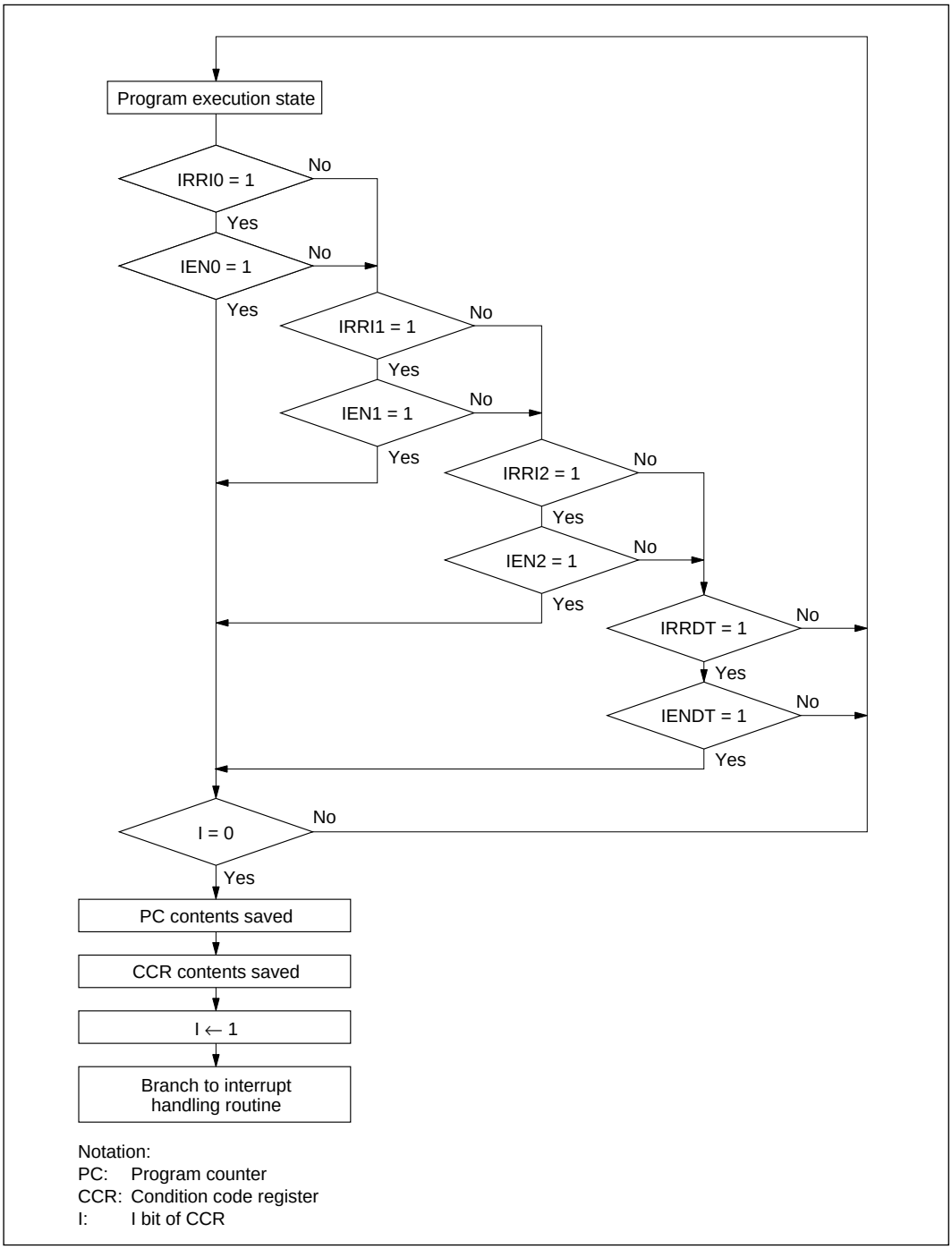


Figure 3-3 Flow up to Interrupt Acceptance

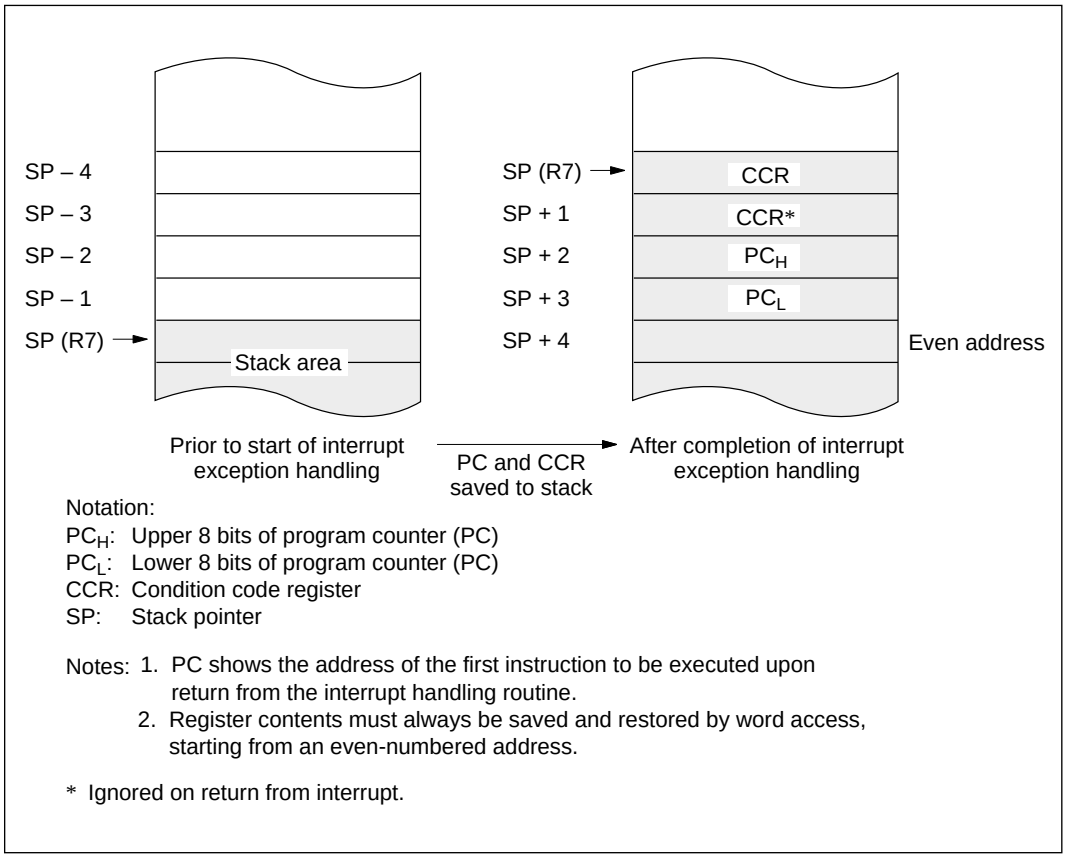


Figure 3-4 Stack State after Completion of Interrupt Exception Handling

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.

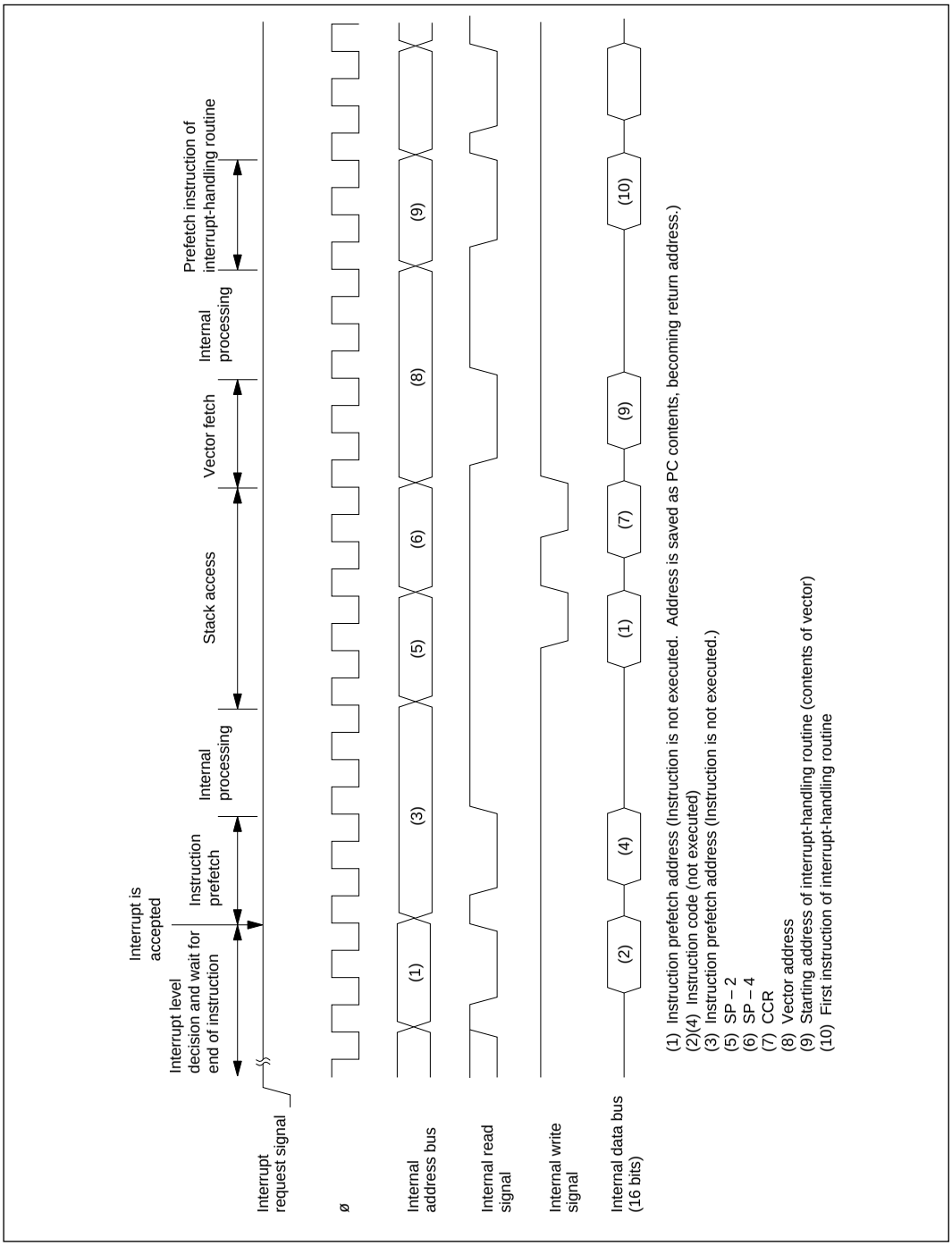


Figure 3-5 Interrupt Sequence

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 H8/3947 Series, the least significant bit of the address is regarded as 0. Access to the stack always takes place in word size, so the stack pointer (SP: R7) should never indicate an odd address. Use PUSH Rn (MOV.W Rn, @–SP) or POP Rn (MOV.W @SP+, Rn) to save or restore register values.

Setting an odd address in SP may cause a program to crash. An example is shown in figure 3-6.

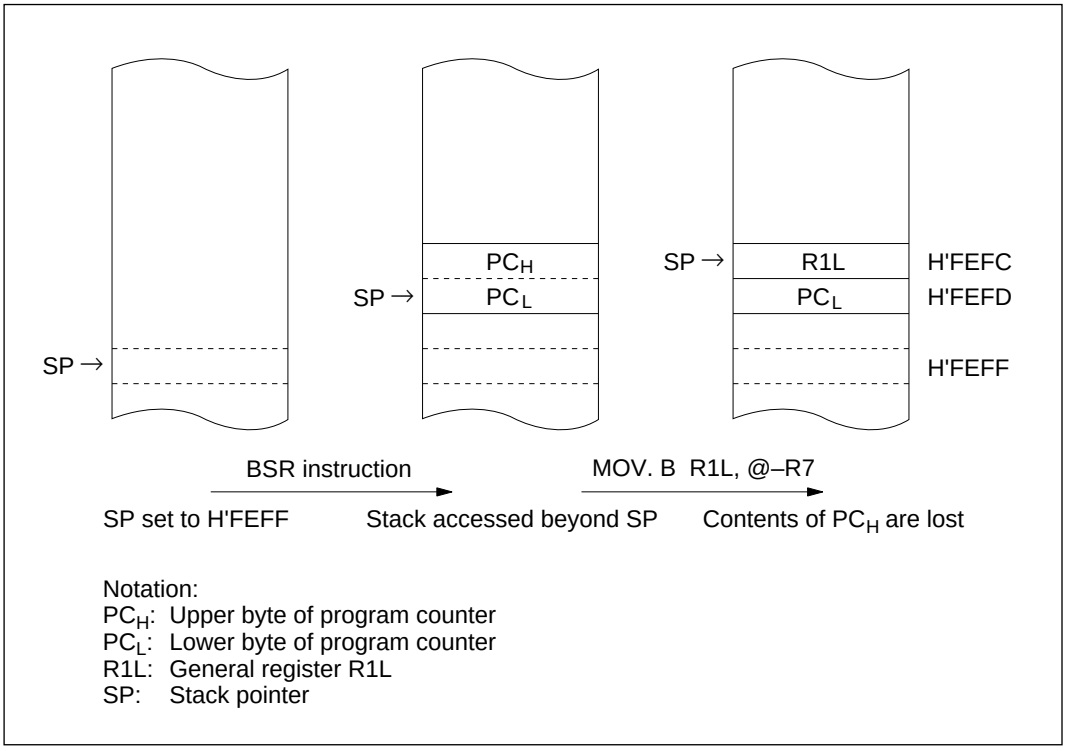


Figure 3-6 Operation when Odd Address is Set in SP

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

3.4.2 Notes on Rewriting Port Mode Registers

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

When an external interrupt pin function is switched by rewriting the port mode register that controls these pins ($\overline{IRQ}_4$ to $\overline{IRQ}_0$, and $\overline{WKP}_7$ to $\overline{WKP}_0$), the interrupt request flag may be set to 1 at the time the pin function is switched, even if no valid interrupt is input at the pin. Be sure to clear the interrupt request flag to 0 after switching pin functions. Table 3-5 shows the conditions under which interrupt request flags are set to 1 in this way.

Table 3-5 Conditions under which Interrupt Request Flag is Set to 1

Interrupt Request Flags Set to 1		Conditions
IRR1	IRRI4	When PMR2 bit IRQ4 is changed from 0 to 1 while pin $\overline{IRQ}_4$ is low and IEGR bit IEG4 = 0.
		When PMR2 bit IRQ4 is changed from 1 to 0 while pin $\overline{IRQ}_4$ is low and IEGR bit IEG4 = 1.
	IRRI3	When PMR1 bit IRQ3 is changed from 0 to 1 while pin $\overline{IRQ}_3$ is low and IEGR bit IEG3 = 0.
		When PMR1 bit IRQ3 is changed from 1 to 0 while pin $\overline{IRQ}_3$ is low and IEGR bit IEG3 = 1.
	IRRI2	When PMR1 bit IRQ2 is changed from 0 to 1 while pin $\overline{IRQ}_2$ is low and IEGR bit IEG2 = 0.
		When PMR1 bit IRQ2 is changed from 1 to 0 while pin $\overline{IRQ}_2$ is low and IEGR bit IEG2 = 1.
	IRRI1	When PMR1 bit IRQ1 is changed from 0 to 1 while pin $\overline{IRQ}_1$ is low and IEGR bit IEG1 = 0.
		When PMR1 bit IRQ1 is changed from 1 to 0 while pin $\overline{IRQ}_1$ is low and IEGR bit IEG1 = 1.
	IRRI0	When PMR2 bit IRQ0 is changed from 0 to 1 while pin $\overline{IRQ}_0$ is low and IEGR bit IEG0 = 0.
		When PMR2 bit IRQ0 is changed from 1 to 0 while pin $\overline{IRQ}_0$ is low and IEGR bit IEG0 = 1.

Table 3-5 Conditions under which Interrupt Request Flag is Set to 1 (cont)

Interrupt Request Flags Set to 1		Conditions
IWPR	IWPF7	When PMR5 bit WKP7 is changed from 0 to 1 while pin $\overline{WKP}_7$ is low
	IWPF6	When PMR5 bit WKP6 is changed from 0 to 1 while pin $\overline{WKP}_6$ is low
	IWPF5	When PMR5 bit WKP5 is changed from 0 to 1 while pin $\overline{WKP}_5$ is low
	IWPF4	When PMR5 bit WKP4 is changed from 0 to 1 while pin $\overline{WKP}_4$ is low
	IWPF3	When PMR5 bit WKP3 is changed from 0 to 1 while pin $\overline{WKP}_3$ is low
	IWPF2	When PMR5 bit WKP2 is changed from 0 to 1 while pin $\overline{WKP}_2$ is low
	IWPF1	When PMR5 bit WKP1 is changed from 0 to 1 while pin $\overline{WKP}_1$ is low
	IWPF0	When PMR5 bit WKP0 is changed from 0 to 1 while pin $\overline{WKP}_0$ is low

Figure 3-7 shows the procedure for setting a bit in a port mode register and clearing the interrupt request flag.

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

An alternative method is to avoid the setting of interrupt request flags when pin functions are switched by keeping the pins at the high level so that the conditions in table 3-5 do not occur.

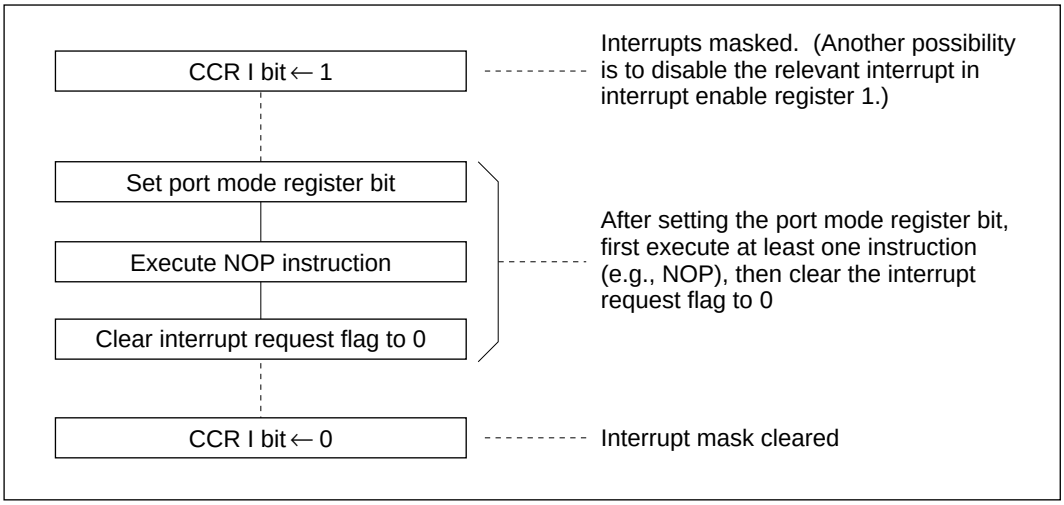


Figure 3-7 Port Mode Register Setting and Interrupt Request Flag Clearing Procedure

Section 4 Clock Pulse Generators

4.1 Overview

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

4.1.1 Block Diagram

Figure 4-1 shows a block diagram of the clock pulse generators.

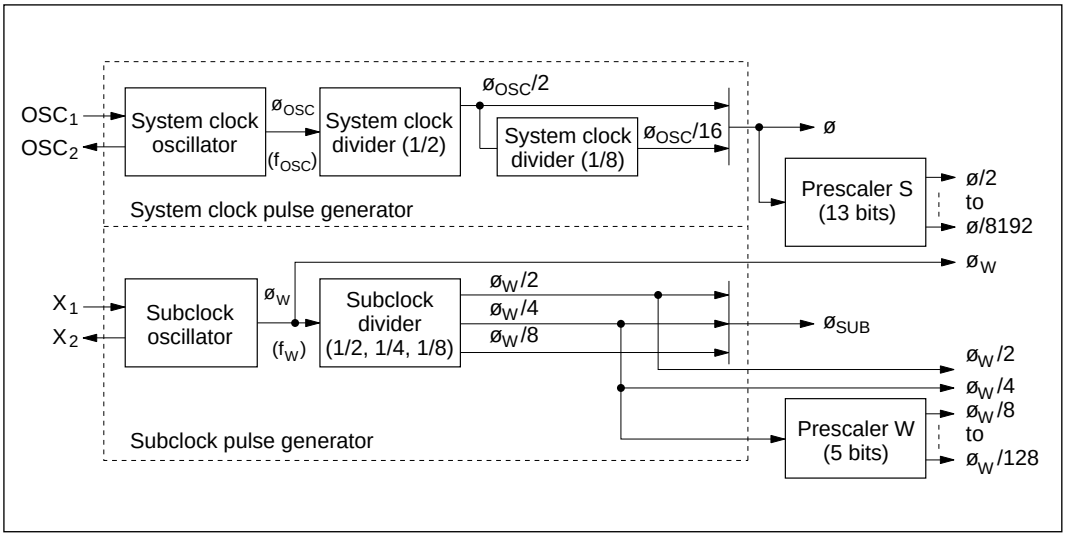


Figure 4-1 Block Diagram of Clock Pulse Generators

4.1.2 System Clock and Subclock

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

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

4.2 System Clock Generator

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

1. Connecting a crystal oscillator

Figure 4-2 shows a typical method of connecting a crystal oscillator.

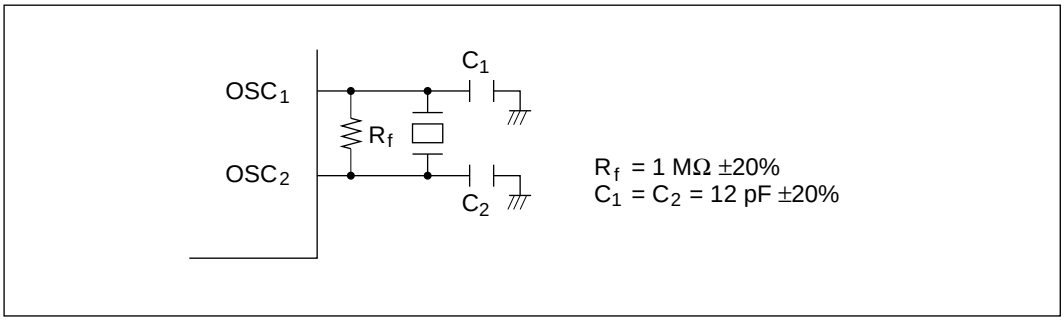


Figure 4-2 Typical Connection to Crystal Oscillator

Figure 4-3 shows the equivalent circuit of a crystal oscillator. An oscillator having the characteristics given in table 4-1 should be used.

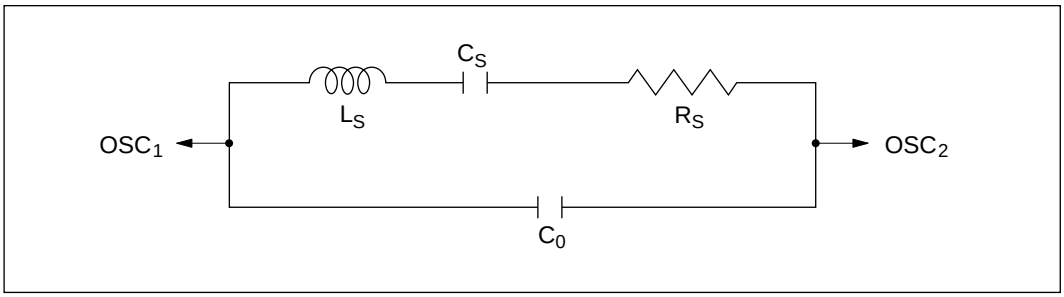


Figure 4-3 Equivalent Circuit of Crystal Oscillator

Table 4-1 Crystal Oscillator Parameters

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

2. Connecting a ceramic oscillator

Figure 4-4 shows a typical method of connecting a ceramic oscillator.

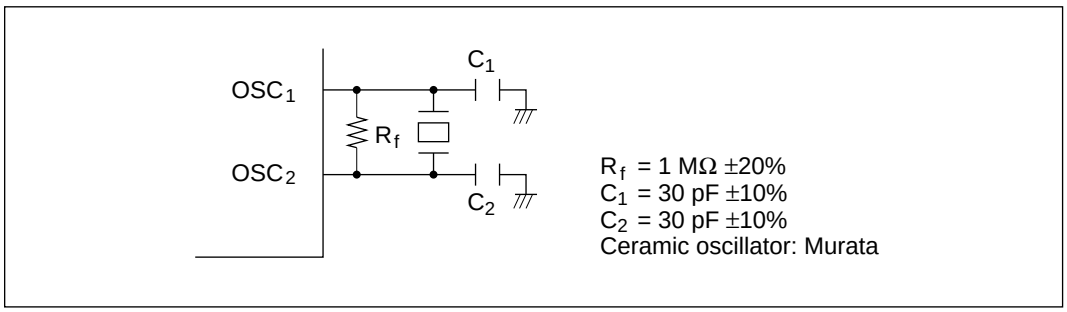


Figure 4-4 Typical Connection to Ceramic Oscillator

3. Notes on board design

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

Avoid running signal lines close to the oscillator circuit, since the oscillator may be adversely affected by induction currents. (See figure 4-5.)

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

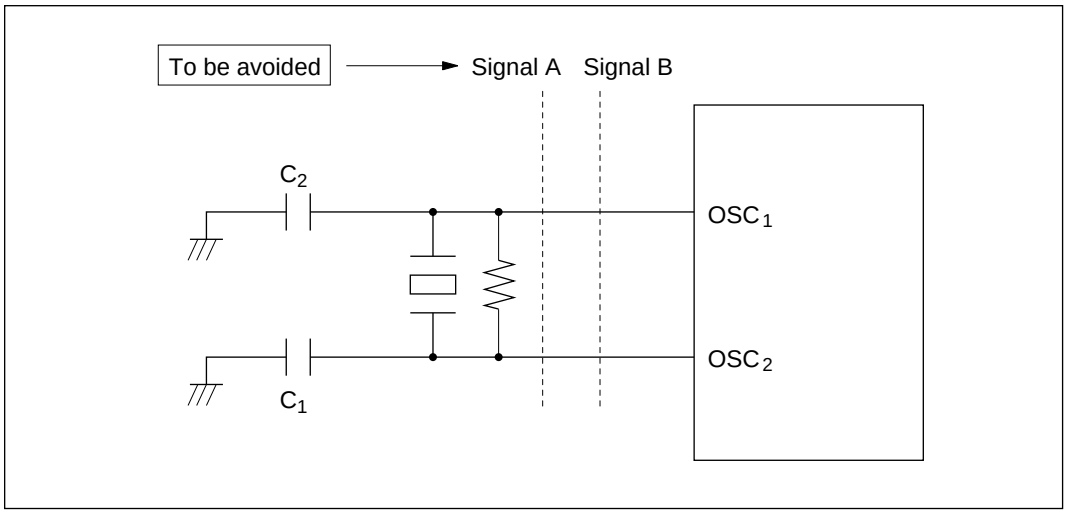


Figure 4-5 Board Design of Oscillator Circuit

4. External clock input method

Connect an external clock signal to pin OSC_1, and leave pin OSC_2 open. Figure 4-6 shows a typical connection.

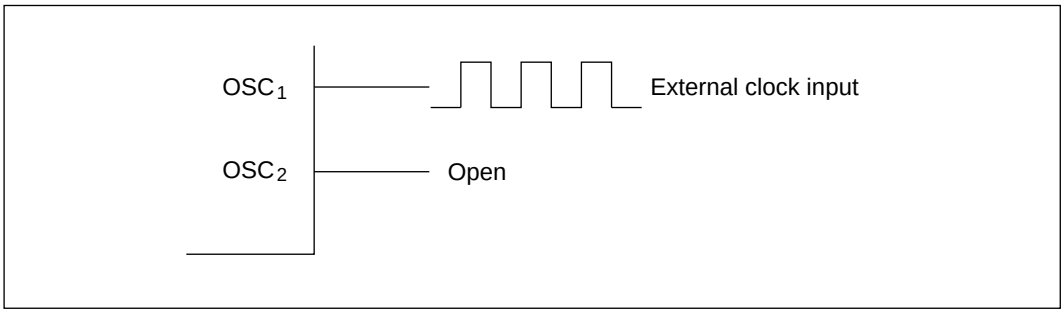


Figure 4-6 External Clock Input (Example)

Frequency	Duty Cycle
Oscillator clock frequency ($ø_{OSC}$)	45% to 55%

4.3 Subclock Generator

1. 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 (3) for the system clock.

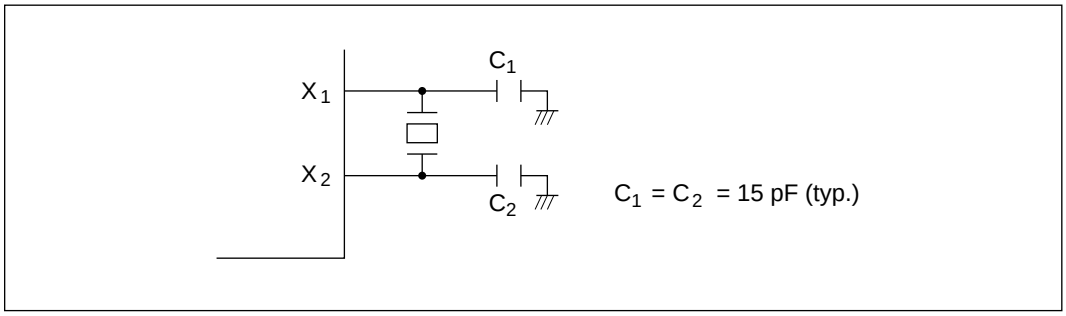


Figure 4-7 Typical Connection to 32.768-kHz Crystal Oscillator (Subclock)

Figure 4-8 shows the equivalent circuit of the 32.768-kHz crystal oscillator.

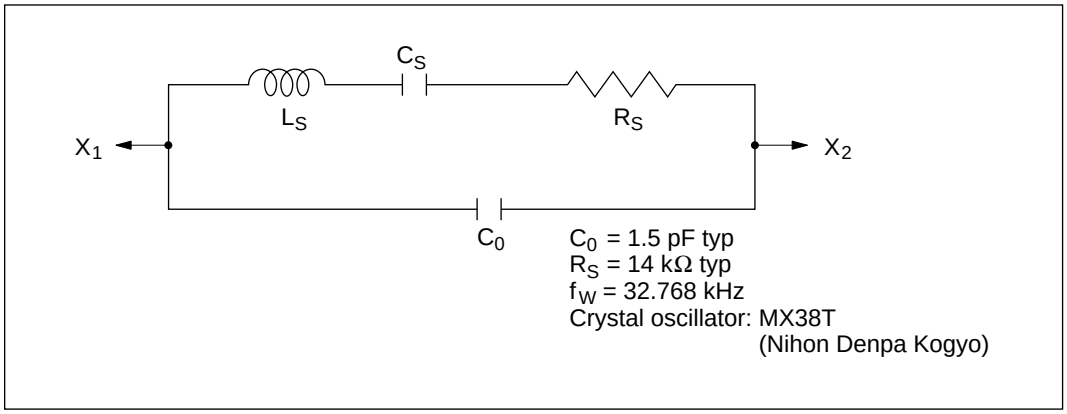


Figure 4-8 Equivalent Circuit of 32.768-kHz Crystal Oscillator

2. 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.

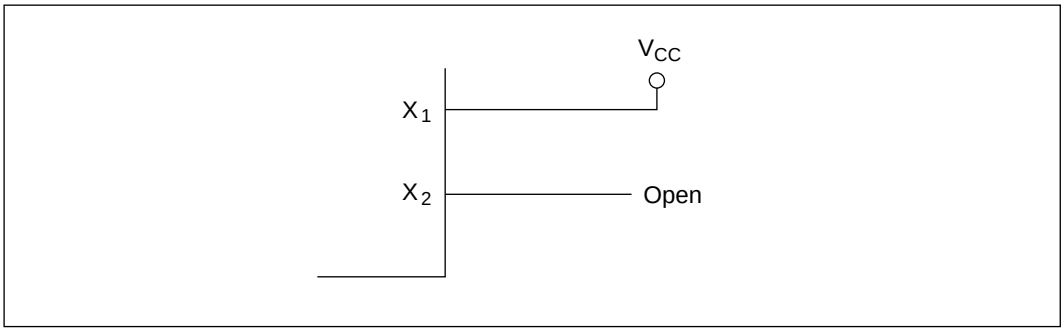


Figure 4-9 Pin Connection when not Using Subclock

4.4 Prescalers

The H8/3947 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.

1. Prescaler S (PSS)

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

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

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

The CPU cannot read or write prescaler S.

The outputs from prescaler S are shared by all on-chip peripheral functions. The divider ratio can be set separately for each on-chip peripheral function.

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

2. 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 H8/3947 Series has seven modes of operation after a reset. These include six power-down modes, in which power dissipation is significantly reduced.

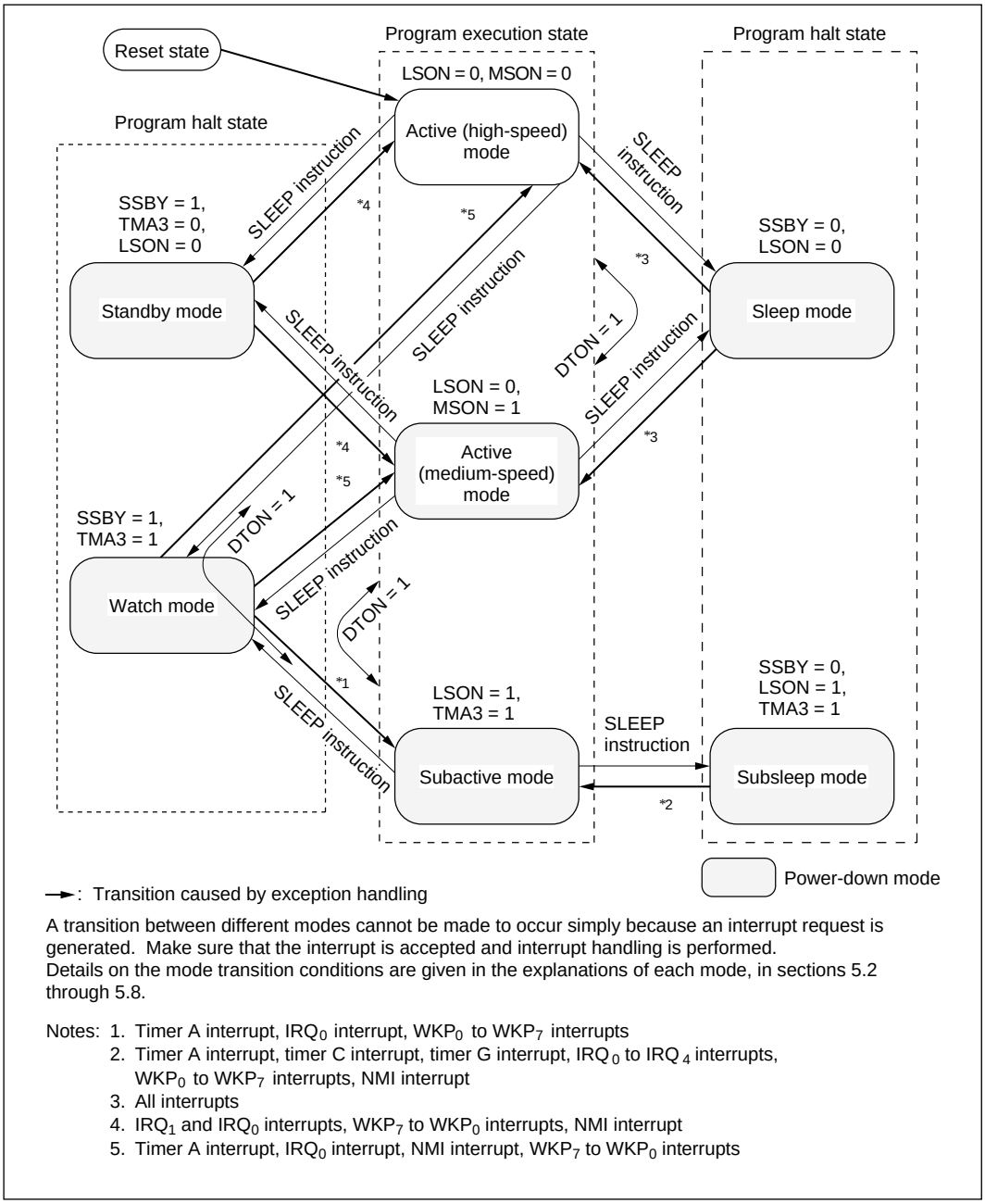
Table 5-1 gives a summary of the seven operation modes.

Table 5-1 Operation Modes

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

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

Figure 5-1 shows the transitions among these operation modes. Table 5-2 indicates the internal states in each mode.



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

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

Figure 5-1 Operation Mode Transition Diagram

Table 5-2 Internal State in Each Operation Mode

Function		Active Mode: High Speed	Active Mode: Medium Speed	Sleep Mode	Watch Mode	Subactive Mode	Subsleep Mode	Standby Mode
System clock oscillator		Functions	Functions	Functions	Halted	Halted	Halted	Halted
Subclock oscillator		Functions	Functions	Functions	Functions	Functions	Functions	Functions
CPU operation	Instructions	Functions	Functions	Halted	Halted	Functions	Halted	Halted
	RAM			Retained	Retained		Retained	Retained
	Registers							
	I/O							Retained*1
External interrupts	NMI	Functions	Functions	Functions	Functions	Functions	Functions	Functions
	IRQ_0							
	IRQ_1				Retained*5			
	IRQ_2							Retained*5
	IRQ_3							
	IRQ_4							
	WKP_0	Functions	Functions	Functions	Functions	Functions	Functions	Functions
	WKP_1							
	WKP_2							
	WKP_3							
	WKP_4							
	WKP_5							
	WKP_6							
	WKP_7							
Peripheral module functions	Timer A	Functions	Functions	Functions	Functions*4	Functions*4	Functions*4	Retained
	Timers B1 to B3				Retained	Retained	Retained	
	Timer C					Functions/ Retained*2	Functions/ Retained*2	
	Timer F					Retained	Retained	
	Timer G					Functions/ Retained*3	Functions/ Retained*3	
	Timer H					Retained	Retained	
	I^2C1, I^2C2	Functions	Functions	Functions	Retained	Retained	Retained	Retained
	SCI3				Reset	Reset	Reset	Reset
	8-bit PWM	Functions	Functions	Retained	Retained	Retained	Retained	Retained
	A/D	Functions	Functions	Functions	Retained	Retained	Retained	Retained

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

5.1.1 System Control Registers

The operation mode is selected using the system control registers described in table 5-3.

Table 5-3 System Control Register

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

1. System control register 1 (SYSCR1)

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

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

Bit 7: Software standby (SSBY)

This bit designates transition to standby mode or watch mode.

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

Bits 6 to 4: Standby timer select 2 to 0 (STS2 to STS0)

These bits designate the time the CPU and peripheral modules wait for stable clock operation after exiting from standby mode or watch mode to active mode due to an interrupt. The designation should be made according to the 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)
0	0	1	Wait time = 16,384 states	
0	1	0	Wait time = 32,768 states	
0	1	1	Wait time = 65,536 states	
1	*	*	Wait time = 131,072 states	

Note: * Don't care

Bit 3: Low speed on flag (LSON)

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

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

Bits 2 to 0: Reserved bits

These bits are reserved; they are always read as 1, and cannot be modified.

2. System control register 2 (SYSCR2)

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

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
1	Sampling rate is $ø_{OSC}$/4

Bit 3: Direct transfer on flag (DTON)

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

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

Bit 2: Medium speed on flag (MSON)

After standby, watch, or sleep mode is cleared, this bit selects active (high-speed) or active (medium-speed) mode.

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

Bits 1 and 0: Subactive mode clock select (SA1 and SA0)

These bits select the CPU clock rate ($ø_W/2$, $ø_W/4$, or $ø_W/8$) in subactive mode. SA1 and SA0 cannot be modified in subactive mode.

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

Note: * Don't care

5.2 Sleep Mode

5.2.1 Transition to Sleep Mode

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

5.2.2 Clearing Sleep Mode

Sleep mode is cleared by any interrupt (timer A, timer B1, timer B2, timer B3, timer C, timer F, timer G, timer H, NMI, IRQ_4 to IRQ_0, WKP_7 to WKP_0, SCI3, I^2C1, I^2C2, 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. Operation resumes in active (high-speed) mode if MSON = 0 in SYSCR2, or active (medium-speed) mode if MSON = 1. Sleep mode is not cleared if the I bit of the condition code register (CCR) is set to 1 or the particular interrupt is disabled in the interrupt enable register.

- Clearing by $\overline{RES}$ input

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

5.3 Standby Mode

5.3.1 Transition to Standby Mode

The system goes from active mode to standby mode when a SLEEP instruction is executed while the SSBY bit in SYSCR1 is set to 1, the LSON bit 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 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 (NMI, IRQ_1, IRQ_0, WKP_7 to WKP_0) or by input at the $\overline{RES}$ pin.

- Clearing by interrupt

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

- Clearing by $\overline{RES}$ input

When the $\overline{RES}$ pin goes low, the system clock pulse generator starts. After the pulse generator output has stabilized, if the $\overline{RES}$ pin is driven high, the CPU starts reset exception handling. Since system clock signals are supplied to the entire chip as soon as the system clock pulse generator starts functioning, the $\overline{RES}$ pin should be kept at the low level until the pulse generator output stabilizes.

5.3.3 Oscillator Settling Time after Standby Mode is Cleared

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

- When a crystal oscillator is used

Table 5-4 gives settings for various operating frequencies. Set bits STS2 to STS0 for a waiting time of at least 10 ms.

Table 5-4 Clock Frequency and Settling Time (times are in ms)

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

Note: * Don't care

- When an external clock is used

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

5.4 Watch Mode

5.4.1 Transition to Watch Mode

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

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

5.4.2 Clearing Watch Mode

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

- Clearing by interrupt

When watch mode is cleared by a timer A interrupt, IRQ_0 interrupt, or WKP_7 to WKP_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 watch mode is cleared by an NMI interrupt request, the transition is to active (high-speed) mode if MSON = 0 or to active (medium-speed) mode if MSON = 1. When the transition is to active mode, after the time set in SYSCR1 bits STS2–STS0 has elapsed, a stable clock signal is supplied to the entire chip, 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 $\overline{RES}$ input 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, timer C, and timer G is halted. As long as a minimum required voltage is applied, the contents of CPU registers and some registers of the on-chip peripheral modules, and the on-chip RAM contents, are retained. I/O ports keep the same states as before the transition.

5.5.2 Clearing Subsleep Mode

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

- Clearing by interrupt

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

- Clearing by $\overline{RES}$ input

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

5.6 Subactive Mode

5.6.1 Transition to Subactive Mode

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

5.6.2 Clearing Subactive Mode

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

- Clearing by SLEEP instruction

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

Clearing by $\overline{RES}$ input 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 NMI, IRQ_0, IRQ_1, or WKP_0 to WKP_7 interrupts in standby mode, timer A, NMI, IRQ_0, or WKP_0 to WKP_7 interrupts in watch mode, or any interrupt in sleep mode. A transition to active (medium-speed) mode does not take place if the I bit of CCR is set to 1 or the particular interrupt is disabled in the interrupt enable register.

5.7.2 Clearing Active (Medium-Speed) Mode

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

- Clearing by SLEEP instruction

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

- Clearing by $\overline{RES}$ 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

In active (medium-speed) mode, the CPU is clocked at 1/8 the frequency in active (high-speed) mode.

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 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 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) mode while the SSBY and LSON bits in SYSCR1 are set to 1, the DTON bit in SYSCR2 is set to 1, and 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 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 H8/3945 has 40 kbytes of on-chip ROM (mask ROM). The H8/3946 has 48 kbytes of on-chip ROM (mask ROM). The H8/3947 has 60 kbytes of on-chip ROM (PROM or mask 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.

When the PROM version of the H8/3947 is set to PROM mode, it can be programmed using a commercially available PROM programmer.

6.1.1 Block Diagram

Figure 6-1 shows a block diagram of the on-chip ROM.

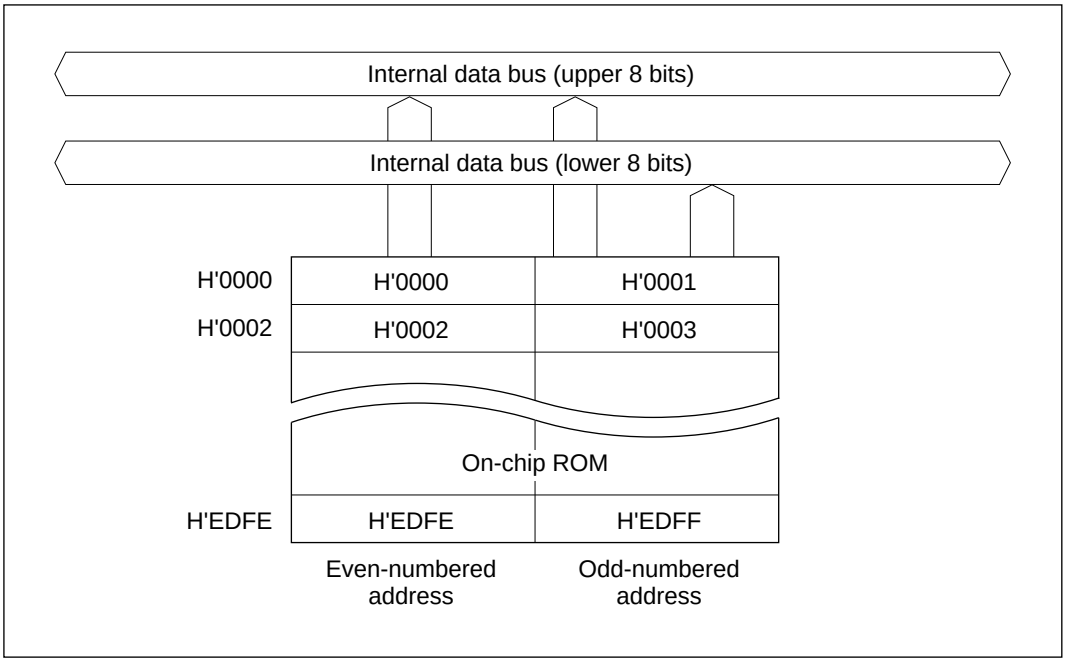


Figure 6-1 ROM Block Diagram (H8/3947)

6.2 PROM Mode

6.2.1 Setting to PROM Mode

Setting the PROM version of the H8/3947 to PROM mode stops its operation as a microcontroller and allows the on-chip PROM to be programmed in the same way as the HN27C101. Page programming, however, is not supported. Table 6-1 shows how to select 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

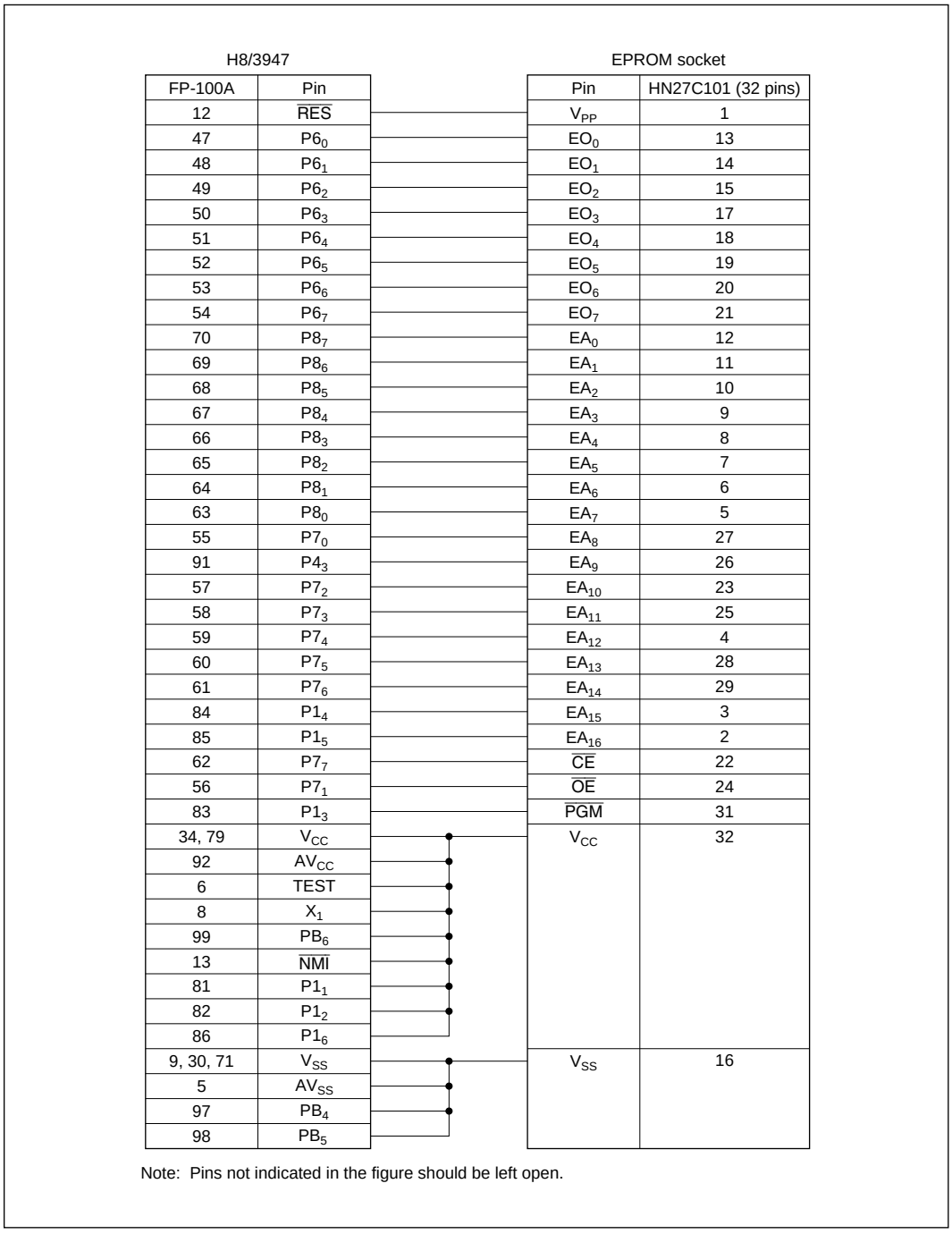
6.2.2 Socket Adapter Pin Arrangement and Memory Map

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

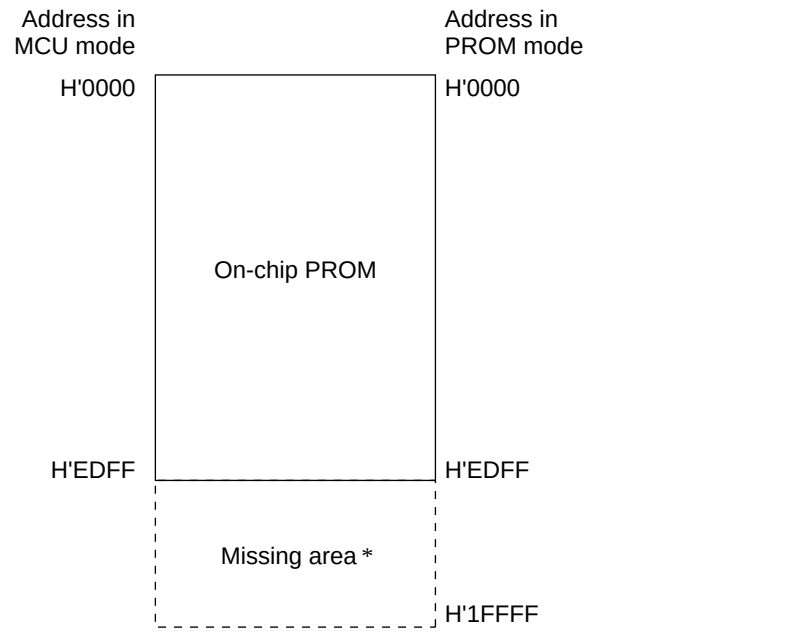
Table 6-2 Socket Adapter

Package	Socket Adapter
100-pin (FP100A)	HS3947ESF01H



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

Figure 6-2 Socket Adapter Pin Correspondence (with HN27C101)



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

Figure 6-3 H8/3947 Memory Map in PROM Mode

6.3 Programming

The write, verify, and other modes are selected as shown in table 6-3 in PROM mode.

Table 6-3 Mode Selection in PROM Mode

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

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

The specifications for writing and reading the on-chip PROM are identical to those for the standard HN27C101 EPROM. Page programming is not supported, however. The PROM writer must not be set to page mode. A PROM programmer that provides only page programming mode cannot be used. When selecting a PROM programer, check that it supports a byte-by-byte high-speed, high-reliability programming method. Be sure to set the address range to H'0000 to H'EDFF.

6.3.1 Writing and Verifying

An efficient, high-speed, high-reliability method is available for writing and verifying the PROM data. This method achieves high speed without voltage stress on the device and without lowering the reliability of written data. The basic flow of this high-speed, high-reliability programming method is shown in figure 6-4.

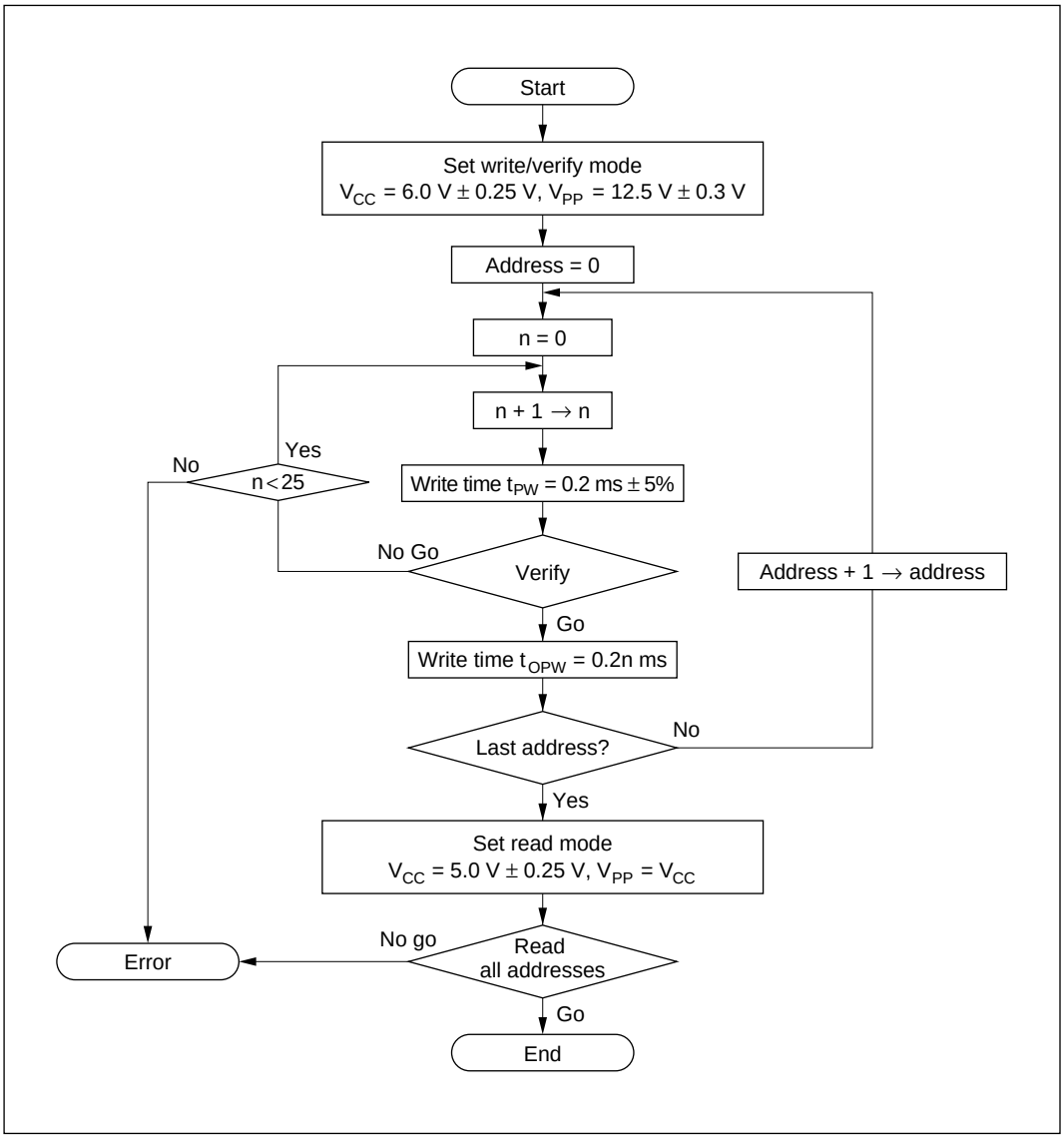


Figure 6-4 High-Speed, High-Reliability Programming Flow Chart

Table 6-4 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 Conditions
Input high-level voltage	EO_7 to EO_0, EA_{16} to EA_0 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	V_{IH}	2.4	—	V_{CC} + 0.3	V	
Input low-level voltage	EO_7 to EO_0, EA_{16} to EA_0 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	V_{IL}	–0.3	—	0.8	V	
Output high-level voltage	EO_7 to EO_0	V_{OH}	2.4	—	—	V	I_{OH} = –200 µA
Output low-level voltage	EO_7 to EO_0	V_{OL}	—	—	0.45	V	I_{OL} = 0.8 mA
Input leakage current	EO_7 to EO_0, EA_{16} to EA_0 $\overline{OE}$, $\overline{CE}$, $\overline{PGM}$	$\lvert I_{LI} \rvert$	—	—	2	µA	V_{in} = 5.25 V/ 0.5 V
V_{CC} current		I_{CC}	—	—	40	mA	
V_{PP} current		I_{PP}	—	—	40	mA	

Table 6-5 AC Characteristics

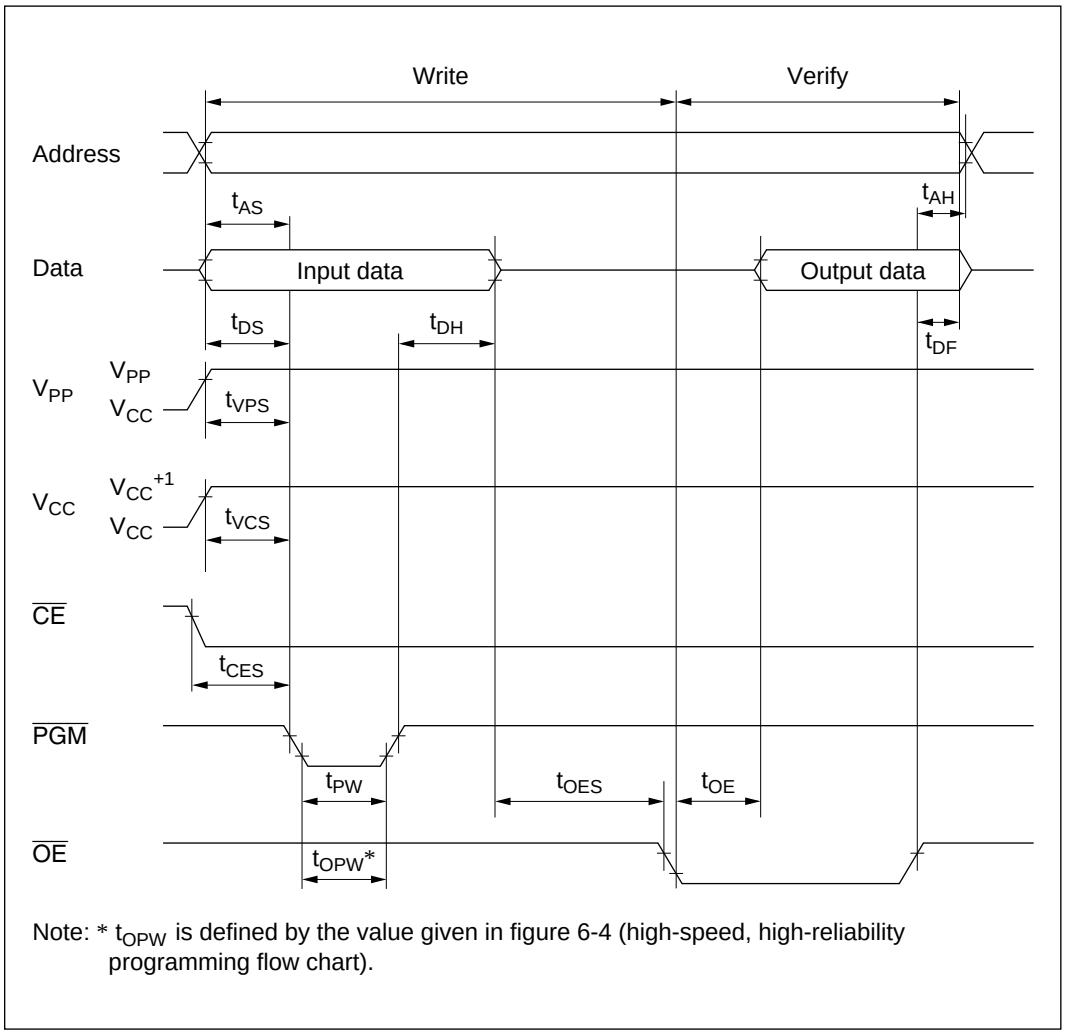
(Conditions: V_{CC} = 6.0 V ±0.25 V, V_{PP} = 12.5 V ±0.3 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Address setup time	t_{AS}	2	—	—	µs	Figure 6-5*1
$\overline{OE}$ setup time	t_{OES}	2	—	—	µs	
Data setup time	t_{DS}	2	—	—	µs	
Address hold time	t_{AH}	0	—	—	µs	
Data hold time	t_{DH}	2	—	—	µs	
Data output disable time	t_{DF}*2	—	—	130	ns	
V_{PP} setup time	t_{VPS}	2	—	—	µs	
Programming pulse width	t_{PW}	0.19	0.20	0.21	ms	
$\overline{PGM}$ pulse width for overwrite programming	t_{OPW}*3	0.19	—	5.25	ms	
V_{CC} setup time	t_{VCS}	2	—	—	µs	
$\overline{CE}$ setup time	t_{CES}	2	—	—	µs	
Data output delay time	t_{OE}	0	—	200	ns	

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

2. t_{DF} is defined at the point at which the output is floating and the output level cannot be read.
3. t_{OPW} is defined by the value given in figure 6-4 (high-speed, high-reliability programming flow chart).

Figure 6-5 shows a 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 HN27C101 will result in correct V_{PP} of 12.5 V.

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

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

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

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

6.4 Reliability of Programmed Data

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

Figure 6-6 shows a flowchart of this screening procedure.

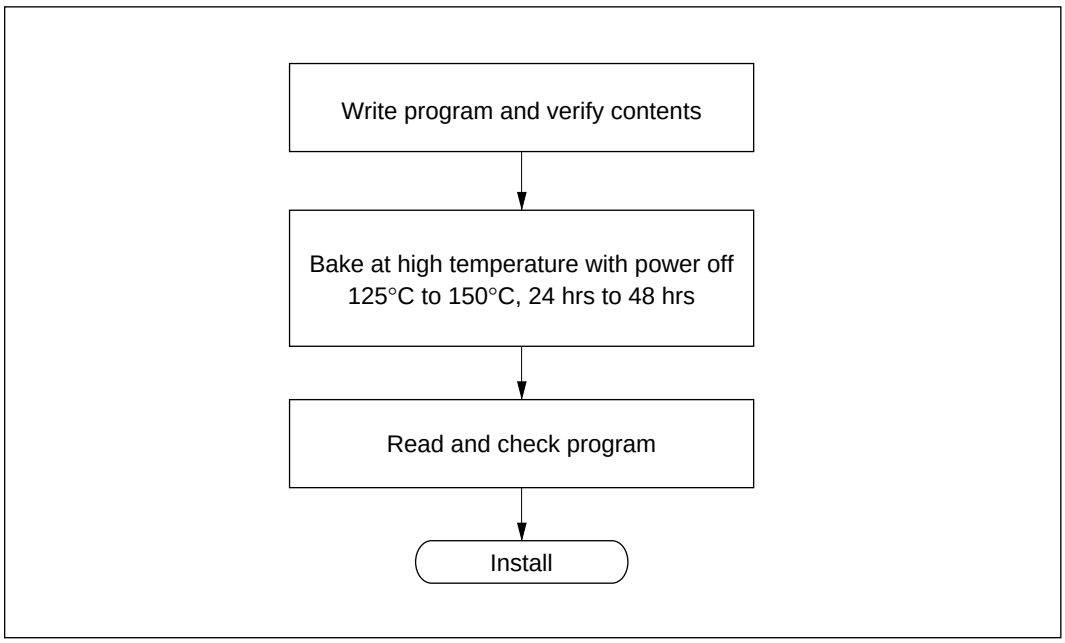


Figure 6-6 Recommended Screening Procedure

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

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

Section 7 RAM

7.1 Overview

The H8/3947 Series has 2 kbytes of high-speed static RAM on-chip. The RAM is connected to the CPU by a 16-bit data bus, allowing high-speed 2-state access for both byte data and word data.

7.1.1 Block Diagram

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

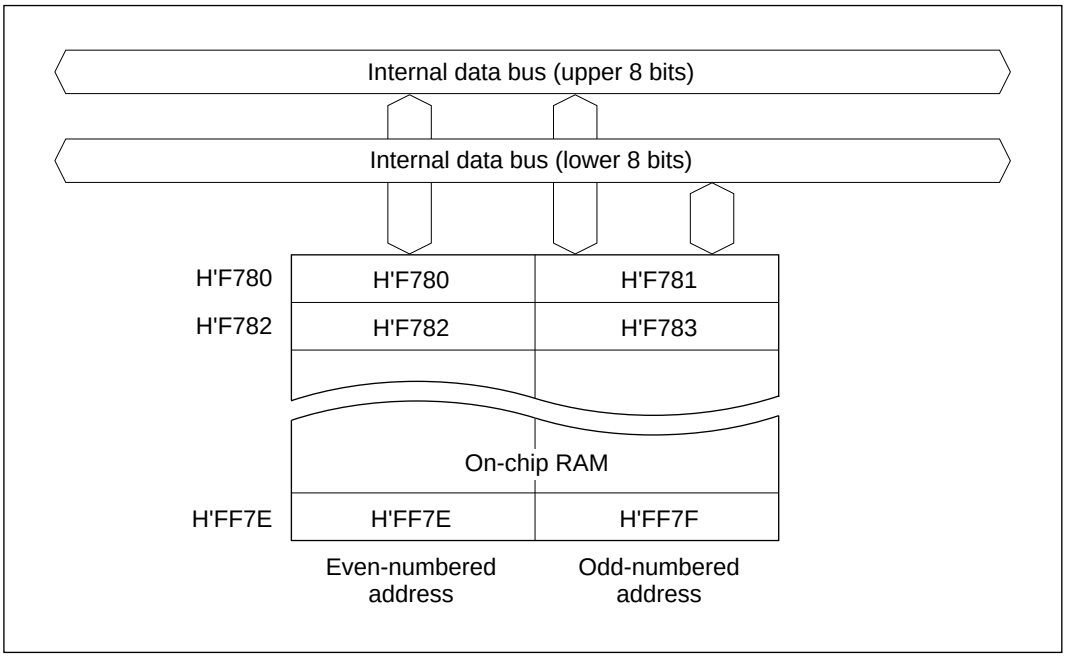


Figure 7-1 RAM Block Diagram

Section 8 I/O Ports

8.1 Overview

The H8/3947 Series is provided with seven 8-bit I/O ports, one 7-bit I/O port, two 3-bit I/O ports, one 8-bit input-only port, one 4-bit input-only port, and one 1-bit input-only port. Table 8-1 indicates the functions of each port.

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

Table 8-1 Port Functions

Port	Description	Pins	Other Functions	Function Switching Register
Port 1	• 8-bit I/O port • MOS input pull-up option	$P1_7$ to $P1_5$/ $\overline{IRQ_3}$ to $\overline{IRQ_1}$/ TMIF, TMIC, TMIB1	External interrupts 3 to 1 Timer event input TMIF, TMIC, TMIB1	PMR1 TCRF, TMC, TMB
		$P1_4$	None	
		$P1_3$/TMIG	Timer G input capture	PMR1
		$P1_2$, $P1_1$/ TMOFH, TMOFL	Timer F output compare	PMR1
		$P1_0$/TMOW	Timer A clock output	PMR1
Port 2	• 8-bit I/O port	$P2_7$, $P2_6$/ TMIB2B, TMIB2A	Timer B2 event inputs TMIB2A, TMIB2B	PMR2
		$P2_5$ to $P2_2$	None	
		$P2_1$/UD	Timer C count-up/down select	PMR2
		$P2_0$/$\overline{IRQ_4}$/ $\overline{ADTRG}$	External interrupt 4 and A/D converter external trigger	PMR2 AMR
Port 3	• 8-bit I/O port • Medium-voltage NMOS open-drain output	$P3_7$ to $P3_0$/ PWM_7 to PWM_0	PWM output (PWM_7 to PWM_0)	PMR3

Table 8-1 Port Functions (cont)

Port	Description	Pins	Other Functions	Function Switching Register
Port 4	• 1-bit input port • 3-bit I/O port	$P4_3/\overline{IRQ_0}$	External interrupt 0	PMR2
		$P4_2$/TXD $P4_1$/RXD $P4_0/SCK_3$	SCI3 data output (TXD), data input (RXD), and clock input/ output (SCK_3)	SCR3 SMR
Port 5	• 8-bit I/O port • MOS input pull-up option	$P5_7$ to $P5_0$/ $\overline{WKP_7}$ to $\overline{WKP_0}$	• Wakeup input ($\overline{WKP_7}$ to $\overline{WKP_0}$)	PMR5
Port 6	• 8-bit I/O port • MOS input pull-up option	$P6_7$ to $P6_0$	None	
Port 7	• 8-bit I/O port • MOS input pull-up option	$P7_7$ to $P7_0$	None	
Port 8	• 8-bit I/O port	$P8_7$ to $P8_0$	None	
Port 9	• 3-bit I/O port • 4-bit input port	$P9_7$ to $P9_5$ TMOH, TMCIH, TMRIH	• Timer H compare-match output (TMOH), event input (TMCIH), and reset input (TMRIH)	TCSRH
		$P9_4$, $P9_3$/ SDA_2, SCL_2	I^2C bus interface data input/output (SDA_2) and clock input/output (SCL_2)	ICCR2
		$P9_2$, $P9_1$/ SDA_1, SCL_1	I^2C bus interface data input/output (SDA_1) and clock input/output (SCL_1)	ICCR1
Port A	• 7-bit I/O port • High-current port	PA_6 to PA_0	None	
Port B	• 8-bit input port	PB_7 to PB_0/ AN_7 to AN_0	A/D converter analog input	AMR
Port C	• 4-bit input port	PC_3 to PC_0/ AN_{11} to AN_8	A/D converter analog input	AMR

8.2 Port 1

8.2.1 Overview

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

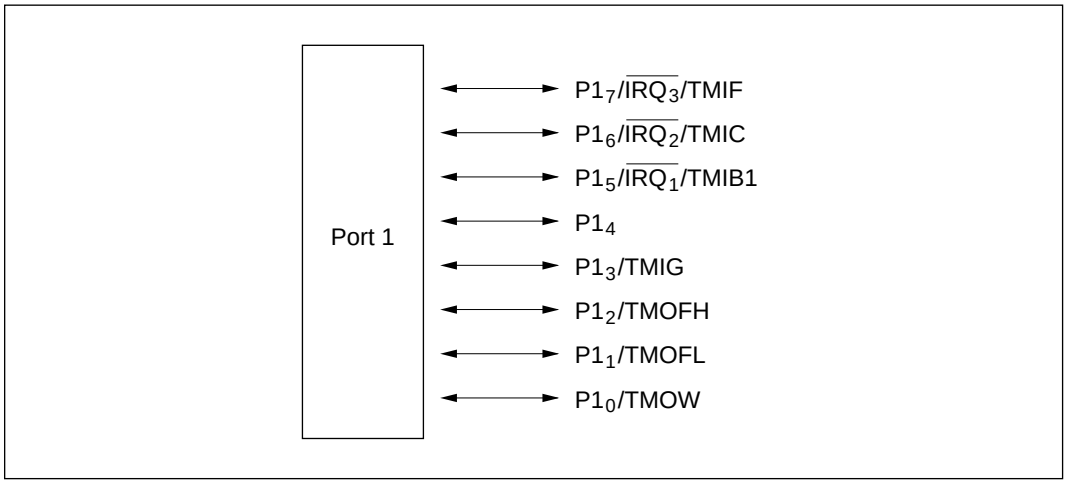


Figure 8-1 Port 1 Pin Configuration

8.2.2 Register Configuration and Description

Table 8-2 shows the port 1 register configuration.

Table 8-2 Port 1 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 1	PDR1	R/W	H'00	H'FFD4
Port control register 1	PCR1	W	H'00	H'FFE4
Port pull-up control register 1	PUCR1	R/W	H'00	H'FFE0
Port mode register 1	PMR1	R/W	H'10	H'FFC8

1. Port data register 1 (PDR1)

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

PDR1 is an 8-bit register that stores data for pins $P1_7$ through $P1_0$. 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.

2. Port control register 1 (PCR1)

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

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

Upon reset, PCR1 is initialized to H'00.

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

3. Port pull-up control register 1 (PUCR1)

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

PUCR1 controls whether the MOS pull-up of each port 1 pin 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.

4. Port mode register 1 (PMR1)

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

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

Upon reset, PMR1 is initialized to H'10.

Bit 7: $P1_7/\overline{IRQ}_3$/TMIF pin function switch (IRQ3)

This bit selects whether pin $P1_7/\overline{IRQ}_3$/TMIF is used as $P1_7$ or as $\overline{IRQ}_3$/TMIF.

Bit 7 IRQ3	**Description**	
0	Functions as $P1_7$ I/O pin	(initial value)
1	Functions as $\overline{IRQ}_3$/TMIF input pin	

Note: Rising or falling edge sensing can be designated for $\overline{IRQ}_3$/TMIF.
For details on TMIF pin settings, see 9.5.2 (3), timer control register F (TCRF).

Bit 6: $P1_6/\overline{IRQ}_2$/TMIC pin function switch (IRQ2)

This bit selects whether pin $P1_6/\overline{IRQ}_2$/TMIC is used as $P1_6$ or as $\overline{IRQ}_2$/TMIC.

Bit 6 IRQ2	**Description**	
0	Functions as $P1_6$ I/O pin	(initial value)
1	Functions as $\overline{IRQ}_2$/TMIC input pin	

Note: Rising or falling edge sensing can be designated for $\overline{IRQ}_2$/TMIC.
For details on TMIC pin settings, see 9.4.2 (1), timer mode register C (TMC).

Bit 5: $P1_5/\overline{IRQ}_1$/TMIB1 pin function switch (IRQ1)

This bit selects whether pin $P1_5/\overline{IRQ}_1$/TMIB1 is used as $P1_5$ or as $\overline{IRQ}_1$/TMIB1.

Bit 5 IRQ1	**Description**	
0	Functions as $P1_5$ I/O pin	(initial value)
1	Functions as $\overline{IRQ}_1$/TMIB1 input pin	

Note: Rising or falling edge sensing can be designated for $\overline{IRQ}_1$/TMIB1.
For details on TMIB1 pin settings, see 9.3.2 (1), timer mode register B1 (TMB1).

Bit 4: Reserved bit

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

Bit 3: $P1_3$/TMIG pin function switch (TMIG)

This bit selects whether pin $P1_3$/TMIG is used as $P1_3$ or as TMIG.

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

Bit 2: $P1_2$/TMOFH pin function switch (TMOFH)

This bit selects whether pin $P1_2$/TMOFH is used as $P1_2$ or as TMOFH.

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

Bit 1: $P1_1$/TMOFL pin function switch (TMOFL)

This bit selects whether pin $P1_1$/TMOFL is used as $P1_1$ or as TMOFL.

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

Bit 0: $P1_0$/TMOW pin function switch (TMOW)

This bit selects whether pin $P1_0$/TMOW is used as $P1_0$ or as TMOW.

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

8.2.3 Pin Functions

Table 8-3 shows the port 1 pin functions.

Table 8-3 Port 1 Pin Functions

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P1_7/\overline{IRQ}_3$/TMIF</td><td>The pin function depends on bit IRQ3 in PMR1, bits CKSL2 to CKSL0 in TCRF, and bit $PCR1_7$ in PCR1.
<table>
<tr><td>IRQ3</td><td colspan="2">0</td><td colspan="2">1</td></tr>
<tr><td>$PCR1_7$</td><td>0</td><td>1</td><td colspan="2">*</td></tr>
<tr><td>CKSL2 to CKSL0</td><td colspan="2">*</td><td>Not 0**</td><td>0**</td></tr>
<tr><td>Pin function</td><td>$P1_7$ input pin</td><td>$P1_7$ output pin</td><td>$\overline{IRQ}_3$ input pin</td><td>$\overline{IRQ}_3$/TMIF input pin</td></tr>
</table>
Note: When using as TMIF input pin, clear bit IEN3 in IENR1 to 0, disabling IRQ_3 interrupts.</td></tr>
<tr><td>$P1_6/\overline{IRQ}_2$/TMIC</td><td>The pin function depends on bit IRQ2 in PMR1, bits TMC2 to TMC0 in TMC, and bit $PCR1_6$ in PCR1.
<table>
<tr><td>IRQ2</td><td colspan="2">0</td><td colspan="2">1</td></tr>
<tr><td>$PCR1_6$</td><td>0</td><td>1</td><td colspan="2">*</td></tr>
<tr><td>TMC2 to TMC0</td><td colspan="2">*</td><td>Not 111</td><td>111</td></tr>
<tr><td>Pin function</td><td>$P1_6$ input pin</td><td>$P1_6$ output pin</td><td>$\overline{IRQ}_2$ input pin</td><td>$\overline{IRQ}_2$/TMIC input pin</td></tr>
</table>
Note: When using as TMIC input pin, clear bit IEN2 in IENR1 to 0, disabling IRQ_2 interrupts.</td></tr>
</table>

Note: * Don’t care

Table 8-3 Port 1 Pin Functions (cont)

Pin	Pin Functions and Selection Method
$P1_5/\overline{IRQ}_1$/ TMIB1	The pin function depends on bit IRQ1 in PMR1, bits TMB2 to TMB0 in TMB1, and bit $PCR1_5$ in PCR1.

IRQ1	0		1	
$PCR1_5$	0	1	*	
TMB2 to TMB0	*		Not 111	111
Pin function	$P1_5$ input pin	$P1_5$ output pin	$\overline{IRQ}_1$ input pin	$\overline{IRQ}_1$/TMIB1 input pin

Note: When using as TMIB1 input pin, clear bit IEN1 in IENR1 to 0, disabling IRQ_1 interrupts.

$P1_4$ The pin function depends on bit $PCR1_4$ in PCR1.

$PCR1_4$	0	1
Pin function	$P1_4$ input pin	$P1_4$ output pin

$P1_3$/TMIG The pin function depends on bit TMIG in PMR1 and bit $PCR1_3$ in PCR1.

TMIG	0		1
$PCR1_3$	0	1	*
Pin function	$P1_3$ input pin	$P1_3$ output pin	TMIG input pin

$P1_2$/TMOFH The pin function depends on bit TMOFH in PMR1 and bit $PCR1_2$ in PCR1.

TMOFH	0		1
$PCR1_2$	0	1	*
Pin function	$P1_2$ input pin	$P1_2$ output pin	TMOFH output pin

$P1_1$/TMOFL The pin function depends on bit TMOFL in PMR1 and bit $PCR1_1$ in PCR1.

TMOFL	0		1
$PCR1_1$	0	1	*
Pin function	$P1_1$ input pin	$P1_1$ output pin	TMOFL output pin

$P1_0$/TMOW The pin function depends on bit TMOW in PMR1 and bit $PCR1_0$ in PCR1.

TMOW	0		1
$PCR1_0$	0	1	*
Pin function	$P1_0$ input pin	$P1_0$ output pin	TMOW output pin

Note: * Don’t care

8.2.4 Pin States

Table 8-4 shows the port 1 pin states in each operating mode.

Table 8-4 Port 1 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P1_7/\overline{IRQ_3}$/TMIF $P1_6/\overline{IRQ_2}$/TMIC $P1_5/\overline{IRQ_1}$/TMIB1 $P1_4$ $P1_3$/TMIG $P1_2$/TMOFH $P1_1$/TMOFL $P1_0$/TMOW	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

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

8.2.5 MOS Input Pull-Up

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

8.3 Port 2

8.3.1 Overview

Port 2 is an 8-bit I/O port. Figure 8-2 shows its pin configuration.

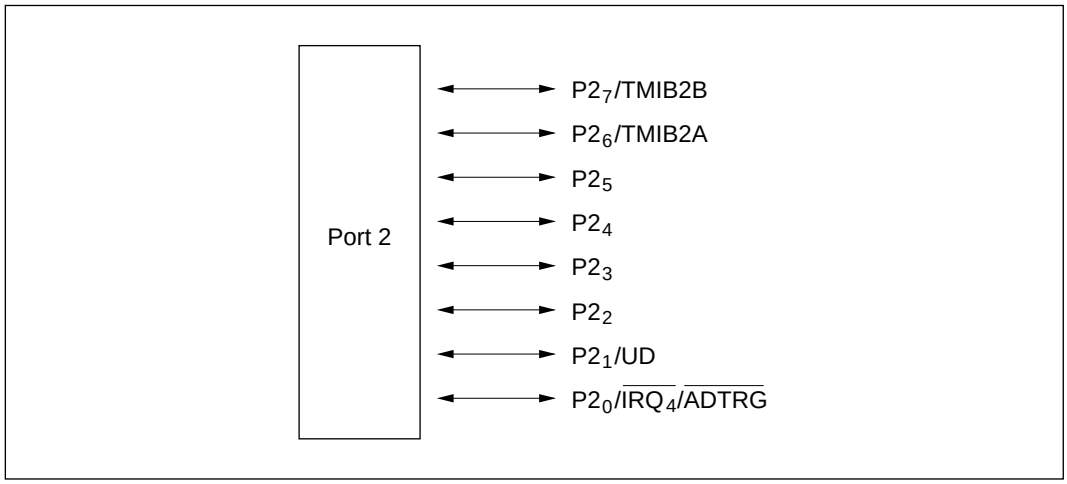


Figure 8-2 Port 2 Pin Configuration

8.3.2 Register Configuration and Description

Table 8-5 shows the port 2 register configuration.

Table 8-5 Port 2 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 2	PDR2	R/W	H'00	H'FFD5
Port control register 2	PCR2	W	H'00	H'FFE5
Port mode register 2	PMR2	R/W	H'04	H'FFC9

1. Port data register 2 (PDR2)

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

PDR2 is an 8-bit register that stores data for pins $P2_7$ to $P2_0$. 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.

2. Port control register 2 (PCR2)

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

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

Upon reset, PCR2 is initialized to H'00.

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

3. Port mode register 2 (PMR2)

Bit	7	6	5	4	3	2	1	0
	TMIB2B	TMIB2A	EDGB2	NCS	IRQ0	—	UD	IRQ4
Initial value	0	0	0	0	0	1	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	—	R/W	R/W

PMR2 is an 8-bit read/write register that controls the functions of pins $P2_0$, $P2_1$, $P2_6$, $P2_7$, and $P4_3$, selects rising or falling edge sensing for pins TMIB2A and TMIB2B, and controls the TMIG input noise canceler.

Upon reset, PMR2 is initialized to H'04.

Bit 7: $P2_7$/TMIB2B function switch (TMIB2B)

This bit selects whether pin $P2_7$/TMIB2B is used as $P2_7$ or as TMIB2B. Regardless of the setting of this bit, however, when bit TMIB2A of PMR2 is set to 1, event input to timer B2 is from the TMIB2A input pin.

Bit 7 TMIB2B	Description	
0	Functions as $P2_7$ I/O pin	(initial value)
1	Functions as TMIB2B input pin	

Bit 6: $P2_6$/TMIB2A function switch (TMIB2A)

This bit selects whether pin $P2_6$/TMIB2A is used as $P2_6$ or as TMIB2A.

Bit 6 TMIB2A	Description	
0	Functions as $P2_6$ I/O pin	(initial value)
1	Functions as TMIB2A input pin	

Bit 5: TMIB2 edge select (EDGB2)

This bit selects whether the rising or falling edge of the input at pins TMIB2A and TMIB2B is sensed.

Bit 5 EDGB2	Description	
0	Falling edge of TMIB2A and TMIB2B input is sensed	(initial value)
1	Rising edge of TMIB2A and TMIB2B input is sensed	

Bit 4: TMIG noise canceller select (NCS)

This bit controls the noise canceller circuit for input capture at pin TMIG.

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

Bit 3: $P4_3/\overline{IRQ_0}$ pin function switch (IRQ0)

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

Bit 3 IRQ0	Description	
0	Functions as $P4_3$ input pin	(initial value)
1	Functions as $\overline{IRQ_0}$ input pin	

Bit 2: Reserved bit

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

Bit 1: $P2_1$/UD pin function switch (UD)

This bit selects whether pin $P2_1$/UD is used as $P2_1$ or as UD.

Bit 1 UD	Description	
0	Functions as $P2_1$ I/O pin	(initial value)
1	Functions as UD input pin	

Bit 0: $P2_0/\overline{IRQ_4}/\overline{ADTRG}$ pin function switch (IRQ4)

This bit selects whether pin $P2_0/\overline{IRQ_4}/\overline{ADTRG}$ is used as $P2_0$ or as $\overline{IRQ_4}/\overline{ADTRG}$.

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

Note: See 12.3.2, Start of A/D Conversion by External Trigger Input, for the $\overline{ADTRG}$ pin setting.

8.3.3 Pin Functions

Table 8-6 shows the port 2 pin functions.

Table 8-6 Port 2 Pin Functions

Pin | **Pin Functions and Selection Method**

$P2_7$/TMIB2B — The pin function depends on bit TMIB2B in PMR2 and bit $PCR2_7$ in PCR2.

TMIB2B	0		1
$PCR2_7$	0	1	*
Pin function	$P2_7$ input	$P2_7$ output	TMIB2B input

$P2_6$/TMIB2A — The pin function depends on bit TMIB2A in PMR2 and bit $PCR2_6$ in PCR2.

TMIB2A	0		1
$PCR2_6$	0	1	*
Pin function	$P2_6$ input	$P2_6$ output	TMIB2A input

$P2_5$ to $P2_2$ — Input or output is selected as follows by the bit settings in PCR2.

(n = 2 to 5)

PCR2n	0	1
Pin function	$P2_n$ input pin	$P2_n$ output pin

$P2_1$/UD — The pin function depends on bit UD in PMR2 and bit $PCR2_1$ in PCR2.

UD	0		1
$PCR2_1$	0	1	*
Pin function	$P2_1$ input pin	$P2_1$ output pin	UD input pin

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

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

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

Note: * Don't care

8.3.4 Pin States

Table 8-7 shows the port 2 pin states in each operating mode.

Table 8-7 Port 2 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P2_7$/TMIB2B $P2_6$/TMIB2A $P2_5$ to $P2_2$ $P2_1$/UD $P2_0$/$\overline{IRQ_4}$/ $\overline{ADTRG}$	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 an 8-bit I/O port, configured as shown in figure 8-3.

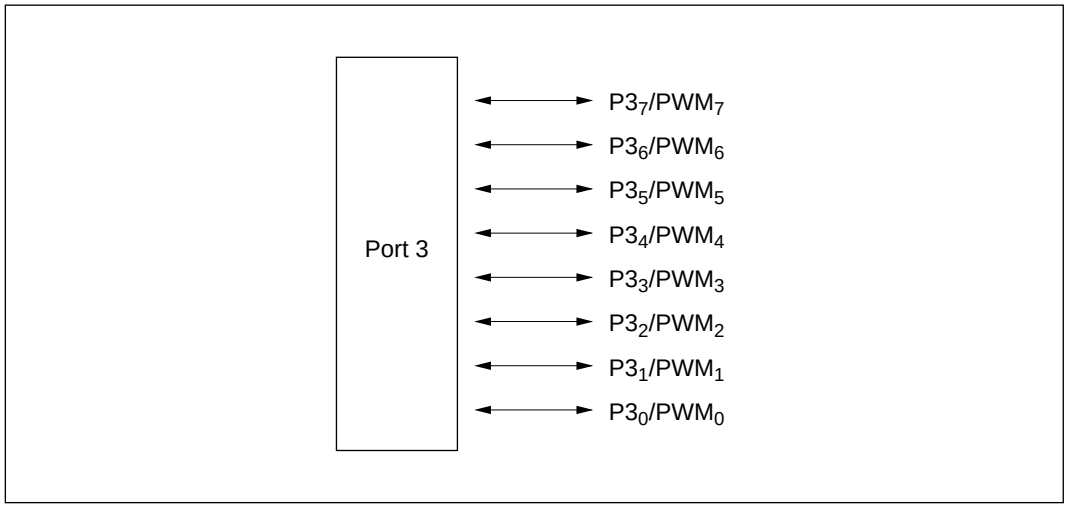


Figure 8-3 Port 3 Pin Configuration

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 mode register 3	PMR3	R/W	H'00	H'FFCA

1. Port data register 3 (PDR3)

Bit	7	6	5	4	3	2	1	0
	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PDR3 is an 8-bit register that stores data for port 3 pins $P3_7$ 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.

2. Port control register 3 (PCR3)

Bit	7	6	5	4	3	2	1	0
	$PCR3_7$	$PCR3_6$	$PCR3_5$	$PCR3_4$	$PCR3_3$	$PCR3_2$	$PCR3_1$	$PCR3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

PCR3 is an 8-bit register for controlling whether each of the port 3 pins $P3_7$ to $P3_0$ 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. All bits are read as 1.

3. Port mode register 3 (PMR3)

Bit	7	6	5	4	3	2	1	0
	PWM7	PWM6	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
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

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.

Bit n: $P3_n/PWM_n$ pin function switch (PWMn)

These bits select whether pin $P3_n/PWM_n$ is used as $P3_n$ or as PWM_n.

Bit n PWMn	Description	
0	Functions as $P3_n$ I/O pin	(initial value)
1	Functions as PWM_n output pin	

(n = 7 to 0)

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
$P3_7/PWM_7$ to $P3_0/PWM_0$	The pin function depends on bits PWMn in PMR3 and bit $PCR3_n$ in PCR3.

PWMn	0		1
$PCR3_n$	0	1	*
Pin function	$P3_n$ input pin	$P3_n$ output pin	PWM_n output pin

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

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
$P3_7/PWM_7$ to $P3_0/PWM_0$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.5 Port 4

8.5.1 Overview

Port 4 consists of a 3-bit I/O port and a 1-bit input port, and is configured as shown in figure 8-4.

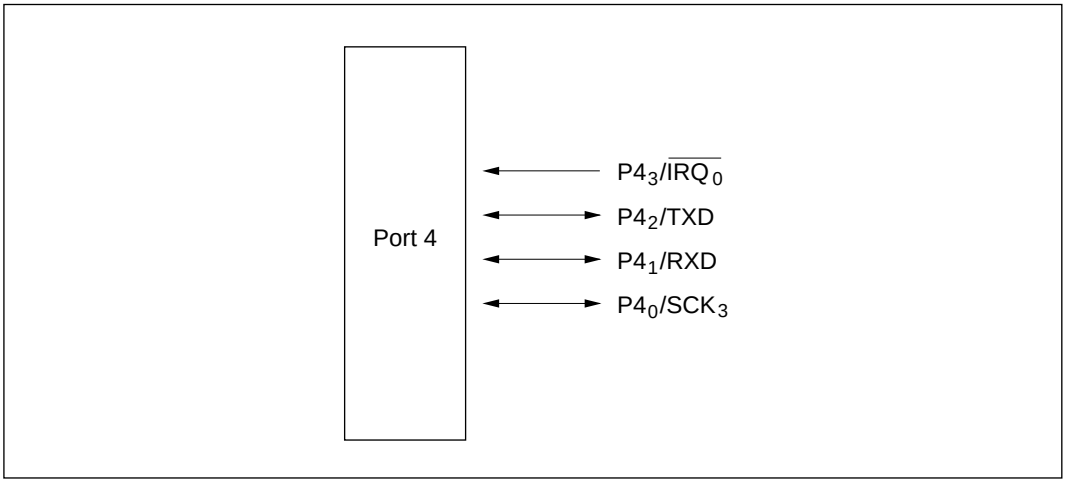


Figure 8-4 Port 4 Pin Configuration

8.5.2 Register Configuration and Description

Table 8-11 shows the port 4 register configuration.

Table 8-11 Port 4 Registers

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

1. Port data register 4 (PDR4)

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

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

Upon reset, PDR4 is initialized to H'F8.

2. Port control register 4 (PCR4)

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

PCR4 controls whether each of the port 4 pins $P4_2$ to $P4_0$ functions as an input pin or output pin. Setting a PCR4 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. The settings in PCR4 and in PDR4 are valid only when the corresponding pin is designated in SCR3 as a general I/O pin.

Upon reset, PCR4 is initialized to H'F8.

PCR4 is a write-only register. All bits are read as 1.

8.5.3 Pin Functions

Table 8-12 shows the port 4 pin functions.

Table 8-12 Port 4 Pin Functions

<table>
<tr><th>Pin</th><th>Pin Functions and Selection Method</th></tr>
<tr><td>$P4_3/\overline{IRQ}_0$</td><td>The pin function depends on the IRQ0 bit setting in PMR2.
<table>
<tr><td>IRQ0</td><td>0</td><td>1</td></tr>
<tr><td>Pin function</td><td>$P4_3$ input pin</td><td>$\overline{IRQ}_0$ input pin</td></tr>
</table></td></tr>
<tr><td>$P4_2$/TXD</td><td>The pin function depends on bit TE in SCR3 and bit $PCR4_2$ in PCR4.
<table>
<tr><td>UD</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR4_2$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P4_2$ input pin</td><td>$P4_2$ output pin</td><td>TXD output pin</td></tr>
</table></td></tr>
<tr><td>$P4_1$/RXD</td><td>The pin function depends on bit RE in SCR3 and bit $PCR4_1$ in PCR4.
<table>
<tr><td>RE</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>$PCR4_1$</td><td>0</td><td>1</td><td>*</td></tr>
<tr><td>Pin function</td><td>$P4_1$ input pin</td><td>$P4_1$ output pin</td><td>RXD input pin</td></tr>
</table></td></tr>
<tr><td>$P4_0/SCK_3$</td><td>The pin function depends on bits CKE1 and CKE0 in SCR3, bit COM in SMR, and bit $PCR4_0$ in PCR4.
<table>
<tr><td>CKE1</td><td colspan="3">0</td><td>1</td></tr>
<tr><td>CKE0</td><td colspan="2">0</td><td>1</td><td>*</td></tr>
<tr><td>COM</td><td colspan="2">0</td><td>1</td><td>*</td></tr>
<tr><td>$PCR4_0$</td><td>0</td><td>1</td><td colspan="2">*</td></tr>
<tr><td>Pin function</td><td>$P4_0$ input pin</td><td>$P4_0$ output pin</td><td>SCK_3 output pin</td><td>SCK_3 input pin</td></tr>
</table></td></tr>
</table>

Note: * Don’t care

8.5.4 Pin States

Table 8-13 shows the port 4 pin states in each operating mode.

Table 8-13 Port 4 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P4_3/\overline{IRQ_0}$ $P4_2$/TXD $P4_1$/RXD $P4_0/SCK_3$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.6 Port 5

8.6.1 Overview

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

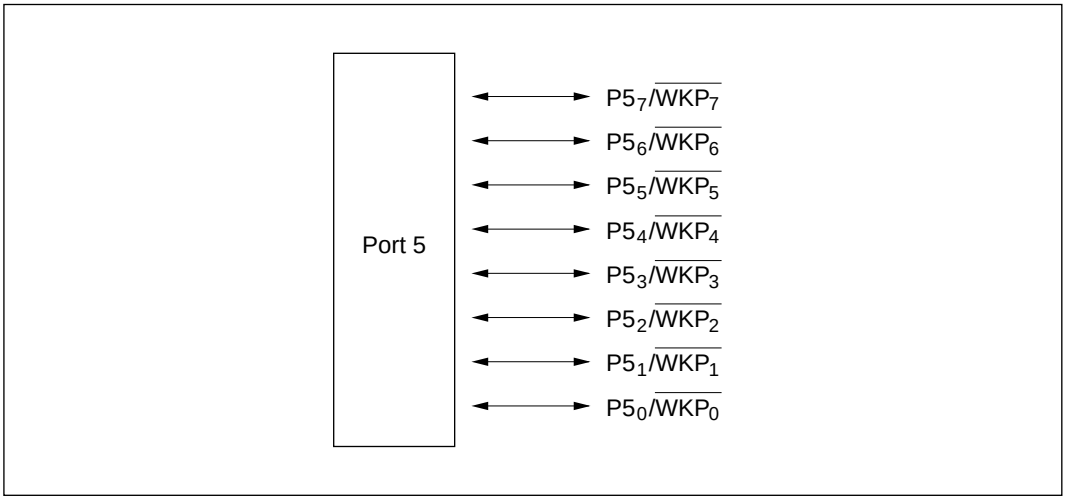


Figure 8-5 Port 5 Pin Configuration

8.6.2 Register Configuration and Description

Table 8-14 shows the port 5 register configuration.

Table 8-14 Port 5 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 5	PDR5	R/W	H'00	H'FFD8
Port control register 5	PCR5	W	H'00	H'FFE8
Port pull-up control register 5	PUCR5	R/W	H'00	H'FFE2
Port mode register 5	PMR5	R/W	H'00	H'FFCC

1. Port data register 5 (PDR5)

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

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

Upon reset, PDR5 is initialized to H'00.

2. Port control register 5 (PCR5)

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

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

Upon reset, PCR5 is initialized to H'00.

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

3. Port pull-up control register 5 (PUCR5)

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

PUCR5 controls whether the MOS pull-up of each port 5 pin ($P5_7$ to $P5_0$) is on or off. When a PCR5 bit is cleared to 0, setting the corresponding PUCR5 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR5 is initialized to H'00.

4. Port mode register 5 (PMR5)

Bit	7	6	5	4	3	2	1	0
	WKP7	WKP6	WKP5	WKP4	WKP3	WKP2	WKP1	WKP0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

Upon reset, PMR5 is initialized to H'00.

Bit n: $P5_n/\overline{WKP}_n$ pin function switch (WKPn)

These bits select whether pin $P5_n/\overline{WKP}_n$ is used as $P5_n$ or as $\overline{WKP}_n$.

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

(n = 7 to 0)

8.6.3 Pin Functions

Table 8-15 shows the port 5 pin functions.

Table 8-15 Port 5 Pin Functions

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

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

Note: * Don’t care

8.6.4 Pin States

Table 8-16 shows the port 5 pin states in each operating mode.

Table 8-16 Port 5 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P5_7/\overline{WKP_7}$ to $P5_0/\overline{WKP_0}$	High-impedance	Retains previous state	Retains previous state	High-impedance*	Retains previous state	Functional	Functional

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

8.6.5 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)

8.7 Port 6

8.7.1 Overview

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

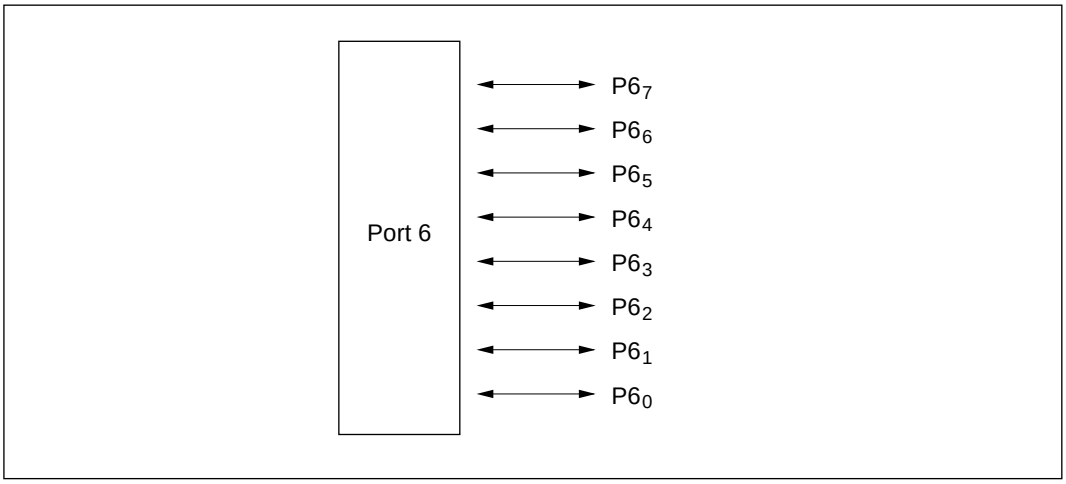


Figure 8-6 Port 6 Pin Configuration

8.7.2 Register Configuration and Description

Table 8-17 shows the port 6 register configuration.

Table 8-17 Port 6 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 6	PDR6	R/W	H'00	H'FFD9
Port control register 6	PCR6	W	H'00	H'FFE9
Port pull-up control register 6	PUCR6	R/W	H'00	H'FFE3

1. Port data register 6 (PDR6)

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

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

Upon reset, PDR6 is initialized to H'00.

2. Port control register 6 (PCR6)

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

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

Upon reset, PCR6 is initialized to H'00.

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

3. Port pull-up control register 6 (PUCR6)

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

PUCR6 controls whether the MOS pull-up of each port 6 pin ($P6_7$ to $P6_0$) is on or off. When a PCR6 bit is cleared to 0, setting the corresponding PUCR6 bit to 1 turns on the MOS pull-up for the corresponding pin, while clearing the bit to 0 turns off the MOS pull-up.

Upon reset, PUCR6 is initialized to H'00.

8.7.3 Pin Functions

Table 8-18 shows the port 6 pin functions.

Table 8-18 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

8.7.4 Pin States

Table 8-19 shows the port 6 pin states in each operating mode.

Table 8-19 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.

8.7.5 MOS Input Pull-Up

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

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

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

8.8 Port 7

8.8.1 Overview

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

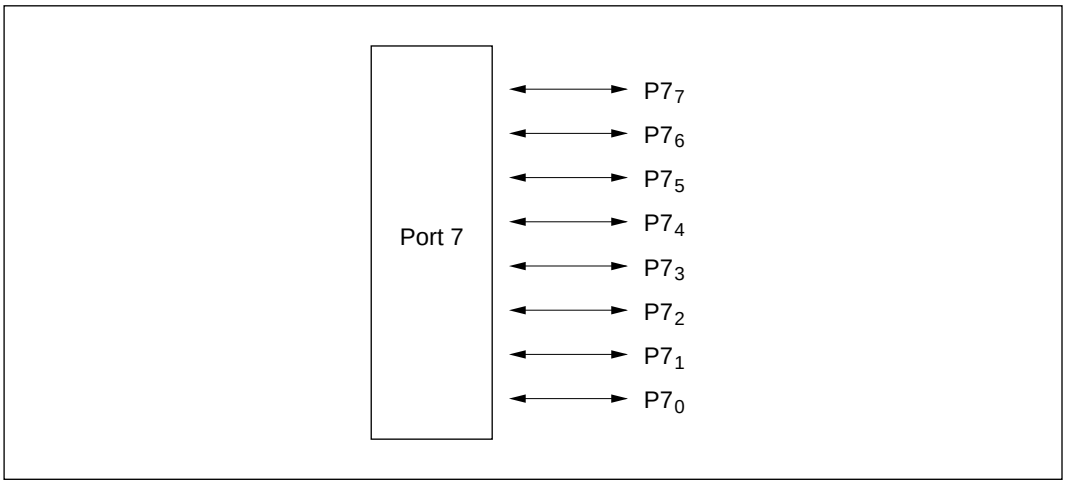


Figure 8-7 Port 7 Pin Configuration

8.8.2 Register Configuration and Description

Table 8-20 shows the port 7 register configuration.

Table 8-20 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 pull-up control register 7	PUCR7	R/W	H'00	H'FFE1

1. Port data register 7 (PDR7)

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

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

Upon reset, PDR7 is initialized to H'00.

2. Port control register 7 (PCR7)

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

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

Upon reset, PCR7 is initialized to H'00.

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

3. Port pull-up control register 7 (PUCR7)

Bit	7	6	5	4	3	2	1	0
	$PUCR7_7$	$PUCR7_6$	$PUCR7_5$	$PUCR7_4$	$PUCR7_3$	$PUCR7_2$	$PUCR7_1$	$PUCR7_0$
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

PUCR7 controls whether the MOS pull-up of each port 7 pin ($P7_7$ to $P7_0$) is on or off. When a PCR7 bit is cleared to 0, setting the corresponding PUCR7 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, PUCR7 is initialized to H'00.

8.8.3 Pin Functions

Table 8-21 shows the port 7 pin functions.

Table 8-21 Port 7 Pin Functions

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

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

Note: * Don't care

8.8.4 Pin States

Table 8-22 shows the port 7 pin states in each operating mode.

Table 8-22 Port 7 Pin States

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

8.8.5 MOS Input Pull-Up

Port 7 has a built-in MOS input pull-up function that can be controlled by software. When a PCR7 bit is cleared to 0, setting the corresponding PUCR7 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.

PCR7n	0		1
PUCR7n	0	1	*
MOS input pull-up	Off	On	Off

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

8.9 Port 8

8.9.1 Overview

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

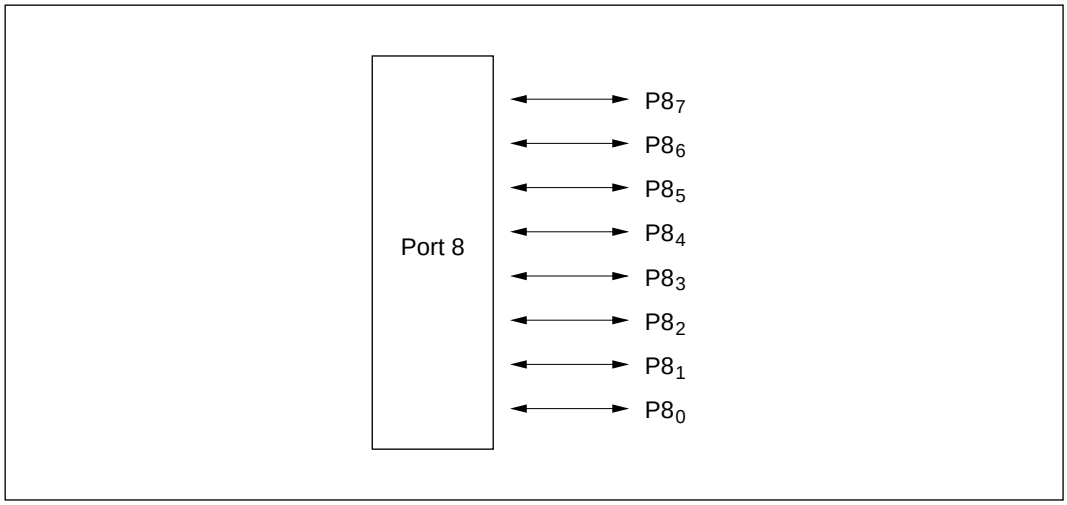


Figure 8-8 Port 8 Pin Configuration

8.9.2 Register Configuration and Description

Table 8-23 shows the port 8 register configuration.

Table 8-23 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

1. 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.

2. Port control register 8 (PCR8)

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

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

Upon reset, PCR8 is initialized to H'00.

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

8.9.3 Pin Functions

Table 8-24 shows the port 8 pin functions.

Table 8-24 Port 8 Pin Functions

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

$PCR8_n$	0	1
Piv bn function	$P8_n$ input pin	$P8_n$ output pin

8.9.4 Pin States

Table 8-25 shows the port 8 pin states in each operating mode.

Table 8-25 Port 8 Pin States

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

8.10 Port 9

8.10.1 Overview

Port 9 consists of a 3-bit I/O port and a 4-bit input port, and is configured as shown in figure 8-9.

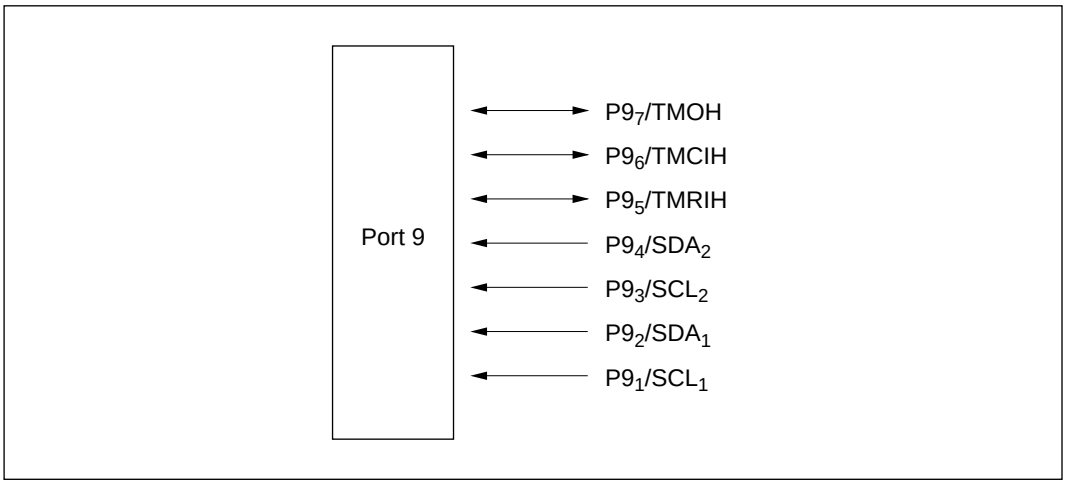


Figure 8-9 Port 9 Pin Configuration

8.10.2 Register Configuration and Description

Table 8-26 shows the port 9 register configuration.

Table 8-26 Port 9 Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register 9	PDR9	R/W	H'1F	H'FFDC
Port control register 9	PCR9	W	H'1F	H'FFEC

1. Port data register 9 (PDR9)

Bit	7	6	5	4	3	2	1	0
	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	—
Initial value	0	0	0	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R	R	R	R	—

PDR9 is an 8-bit register that stores data for port 9 pins $P9_7$ to $P9_1$. 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'1F.

2. Port control register 9 (PCR9)

Bit	7	6	5	4	3	2	1	0
	$PCR9_7$	$PCR9_6$	$PCR9_5$	—	—	—	—	—
Initial value	0	0	0	1	1	1	1	1
Read/Write	W	W	W	—	—	—	—	—

PCR9 is an 8-bit register for controlling whether each of the port 9 pins $P9_7$ to $P9_5$ functions as an input 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'1F.

PCR9 is a write-only register. All bits are read as 1.

8.10.3 Pin Functions

Table 8-27 shows the port 9 pin functions.

Table 8-27 Port 9 Pin Functions

Pin	Pin Functions and Selection Method
$P9_7$/TMOH	The pin function depends on bit $PCR9_7$ in PCR9 and bits OS3 to OS0 in TCSRH.

0S3 to 0S0	0000		Not 0000
$PCR9_7$	0	1	*
Pin function	$P9_7$ input pin	$P9_7$ output pin	TMOH output pin

$P9_6$/TMCIH	The pin function depends on bit $PCR9_6$ in PCR9.

$PCR9_6$	0	1
Pin function	$P9_6$ input pin	$P9_6$ output pin
	TMCIH input pin	

$P9_5$/TMRIH	The pin function depends on bit $PCR9_5$ in PCR9.

$PCR9_5$	0	1
Pin function	$P9_5$ input pin	$P9_5$ output pin
	TMRIH input pin	

$P9_4$/SDA_2	The pin function depends on bit ICE in ICCR2.

ICE	0	1
Pin function	$P9_4$ input pin	SDA_2 input/output pin

$P9_3$/SCL_2	The pin function depends on bit ICE in ICCR2.

ICE	0	1
Pin function	$P9_3$ input pin	SCL_2 input/output pin

Note: * Don’t care

Table 8-27 Port 9 Pin Functions (cont)

Pin	Pin Functions and Selection Method
$P9_2/SDA_1$	The pin function depends on bit ICE in ICCR1. ICE: 0 → Pin function: $P9_2$ input pin; 1 → Pin function: SDA_1 input/output pin
$P9_1/SCL_1$	The pin function depends on bit ICE in ICCR1. ICE: 0 → Pin function: $P9_1$ input pin; 1 → Pin function: SCL_1 input/output pin

8.10.4 Pin States

Table 8-28 shows the port 9 pin states in each operating mode.

Table 8-28 Port 9 Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P9_7$/TMOH $P9_6$/TMCIH $P9_5$/TMRIH $P9_4/SDA_2$ $P9_3/SCL_2$ $P9_2/SDA_1$ $P9_1/SCL_1$	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.11 Port A

8.11.1 Overview

Port A is a 7-bit I/O port, configured as shown in figure 8-10.

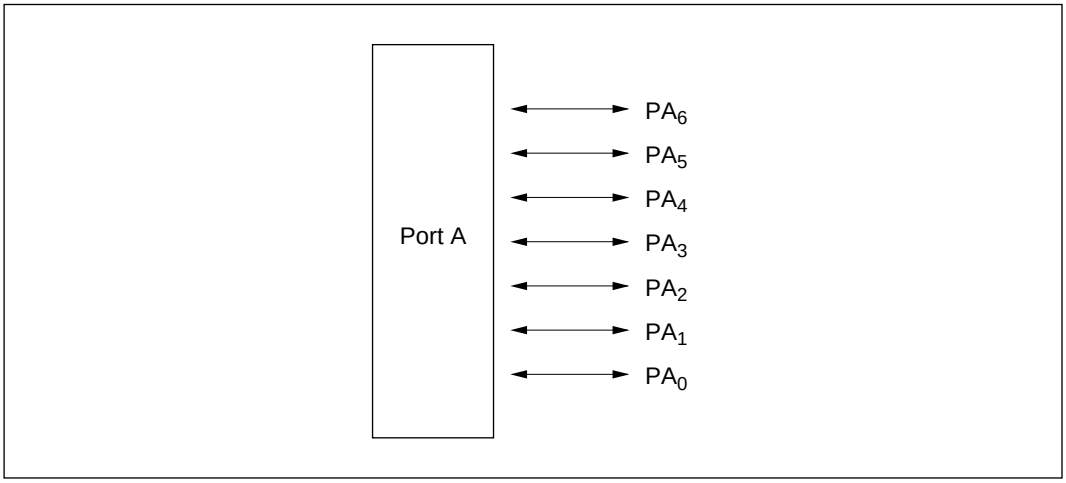


Figure 8-10 Port A Pin Configuration

8.11.2 Register Configuration and Description

Table 8-29 shows the port A register configuration.

Table 8-29 Port A Registers

Name	Abbrev.	R/W	Initial Value	Address
Port data register A	PDRA	R/W	H'80	H'FFDD
Port control register A	PCRA	W	H'80	H'FFED

1. Port data register A (PDRA)

Bit	7	6	5	4	3	2	1	0
	—	PA_6	PA_5	PA_4	PA_3	PA_2	PA_1	PA_0
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PDRA is an 8-bit register that stores data for port A pins PA_6 to PA_0. If port A is read while PCRA bits are set to 1, the values stored in PDRA are read, regardless of the actual pin states. If port A is read while PCRA bits are cleared to 0, the pin states are read.

Upon reset, PDRA is initialized to H'80.

2. Port control register A (PCRA)

Bit	7	6	5	4	3	2	1	0
	—	$PCRA_6$	$PCRA_5$	$PCRA_4$	$PCRA_3$	$PCRA_2$	$PCRA_1$	$PCRA_0$
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

PCRA is an 8-bit register for controlling whether each of the port A pins PA_6 to PA_0 functions as an input or output pin. Setting a PCRA bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin.

Upon reset, PCRA is initialized to H'80.

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

8.11.3 Pin Functions

Table 8-30 gives the port A pin functions.

Table 8-30 Port A Pin Functions

Pin	Pin Functions and Selection Method
PA_6 to PA_0	The pin function depends on bit $PCRA_n$ in PCRA. (n = 6 to 0)

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

8.11.4 Pin States

Table 8-31 shows the port A pin states in each operating mode.

Table 8-31 Port A Pin States

Pins	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
PA_6 to PA_0	High-impedance	Retains previous state	Retains previous state	High-impedance	Retains previous state	Functional	Functional

8.12 Port B

8.12.1 Overview

Port B is an 8-bit input-only port, configured as shown in figure 8-11.

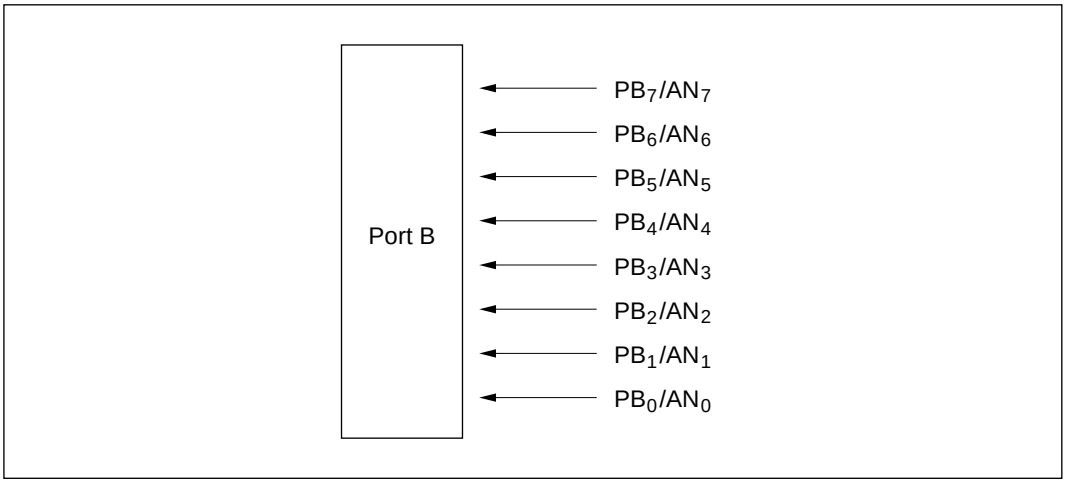


Figure 8-11 Port B Pin Configuration

8.12.2 Register Configuration and Description

Table 8-32 shows the port B register configuration.

Table 8-32 Port B Register

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

Port Data Register B (PDRB)

Bit	7	6	5	4	3	2	1	0
	PB_7	PB_6	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.

8.12.3 Pin Functions

Table 8-33 shows the port B pin functions.

Table 8-33 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

8.12.4 Pin States

Table 8-34 shows the port B pin states in each operating mode.

Table 8-34 Port B Pin States

Pin	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)

8.13 Port C

8.13.1 Overview

Port C is a 4-bit input-only port, configured as shown in figure 8-12.

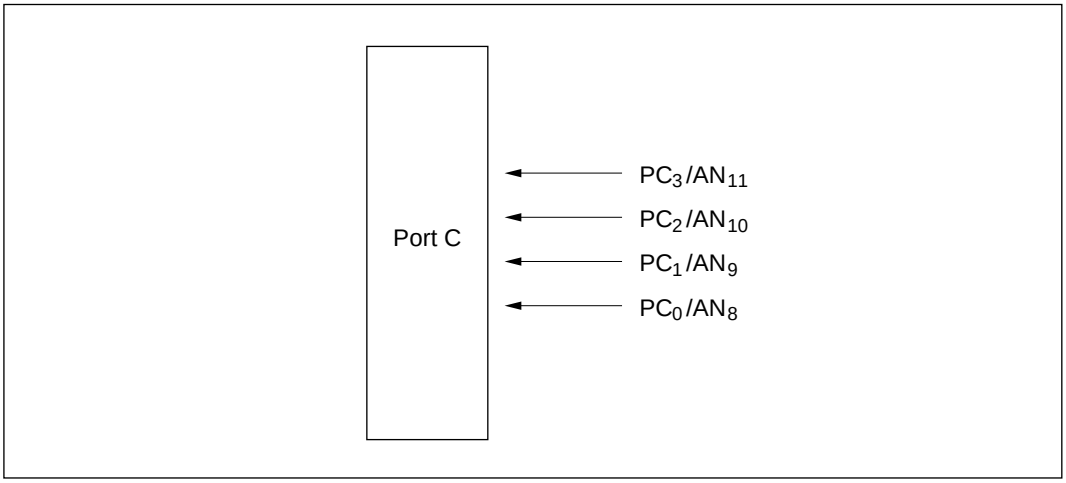


Figure 8-12 Port C Pin Configuration

8.13.2 Register Configuration and Description

Table 8-35 shows the port C register configuration.

Table 8-35 Port C Register

Name	Abbrev.	R/W	Address
Port data register C	PDRC	R	H'FFDF

Port Data Register C (PDRC)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PC_3	PC_2	PC_1	PC_0
Read/Write	—	—	—	—	R	R	R	R

Reading PDRC always gives the pin states. However, if a port C 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.

8.13.3 Pin Functions

Table 8-36 shows the port C pin functions.

Table 8-36 Port C Pin Functions

Pin	Pin Functions and Selection Method	
PC_3/AN_{11} to PC_0/AN_8	Always as below.	
	Pin function	PC_3 to PC_0 input pin or AN_{11} to AN_8 input pin

8.13.4 Pin States

Table 8-37 shows the port C pin states in each operating mode.

Table 8-37 Port C Pin States

Pin	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
PC_3/AN_{11} to PC_0/AN_8	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance	High-impedance

Section 9 Timers

9.1 Overview

The H8/3947 Series provides eight on-chip timers: timers A, B1, B2, B3, C, F, G, and H. 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 timer	ø/8 to ø/8192 (8 choices)	—	—	
		• Time base	$ø_W$/128 (choice of 4 overflow periods)			
		• Clock output	ø/4 to ø/32 $ø_W$/4 to $ø_W$/32 (8 choices)	—	TMOW	
Timer B1, B2, B3	• 8-bit timer • Interval timer • Event counter		ø/4 to ø/8192 (7 choices)	TMIB1 TMIB2A TMIB2B	—	Timer B3 has no event counter function
Timer C	• 8-bit timer • Interval timer • Event counter • Choice of up- or down-counting		ø/4 to ø/8192 $ø_W$/4 (7 choices)	TMIC	—	Counting direction can be controlled by software or hardware
Timer F	• 16-bit timer • Event counter • Can be used as two independent 8-bit timers • Output compare		ø/2 to ø/32 (4 choices)	TMIF	TMOFL TMOFH	
Timer G	• 8-bit timer • Input capture • Interval timer		ø/2 to ø/64 $ø_W$/2 (4 choices)	TMIG	—	• Counter clearing option • Built-in noise canceler for input capture signal

Table 9-1 Timer Functions (cont)

Name	Functions	Internal Clock	Event Input Pin	Waveform Output Pin	Remarks
Timer H	• 8-bit timer • Event counter • Output control by dual compare match • Counter clearing option	ø/8 to ø/1024 (3 choices)	TMCIH	TMOH	External counter clearing pin (TMRIH)

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.

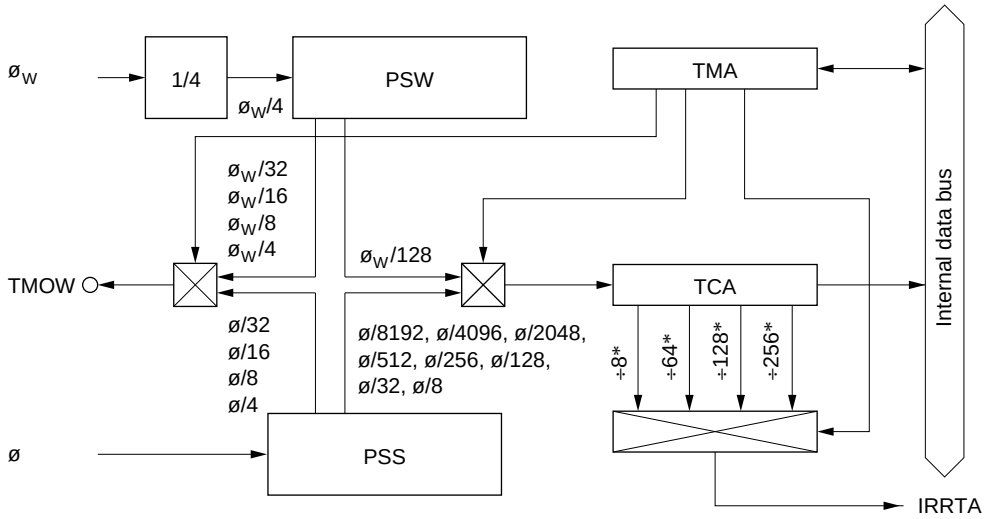
1. Features

Features of timer A are given below.

- Choice of eight internal clock sources (ø/8192, ø/4096, ø/2048, ø/512, ø/256, ø/128, ø/32, ø/8).
- Choice of four overflow periods (1 s, 0.5 s, 0.25 s, 31.25 ms) when timer A is used as a clock time base (using a 32.768 kHz crystal oscillator).
- An interrupt is requested when the counter overflows.
- 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.

2. Block diagram

Figure 9-2-1 shows a block diagram of timer A.



Notation:

TMA: Timer mode register A
TCA: Timer counter A
IRRTA: Timer A overflow interrupt request flag (interrupt request register 1)
PSW: Prescaler W
PSS: Prescaler S

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

Figure 9-2-1 Block Diagram of Timer A

3. Pin configuration

Table 9-2-1 shows the timer A pin configuration.

Table 9-2-1 Pin Configuration

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

4. Register configuration

Table 9-2-2 shows the register configuration of timer A.

Table 9-2-2 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

1. 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			

2. Timer counter A (TCA)

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

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

TCA is cleared by setting bits TMA3 and TMA2 of TMA both to 1.

Upon reset, TCA is initialized to H'00.

9.2.3 Timer Operation

1. Interval timer operation

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

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

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

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

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

2. Real-time clock time base operation

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

3. 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 9-2-3 summarizes the timer A operation states.

Table 9-2-3 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.

1. 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.

2. Block diagram

Figure 9-3-1 shows a block diagram of timer B1.

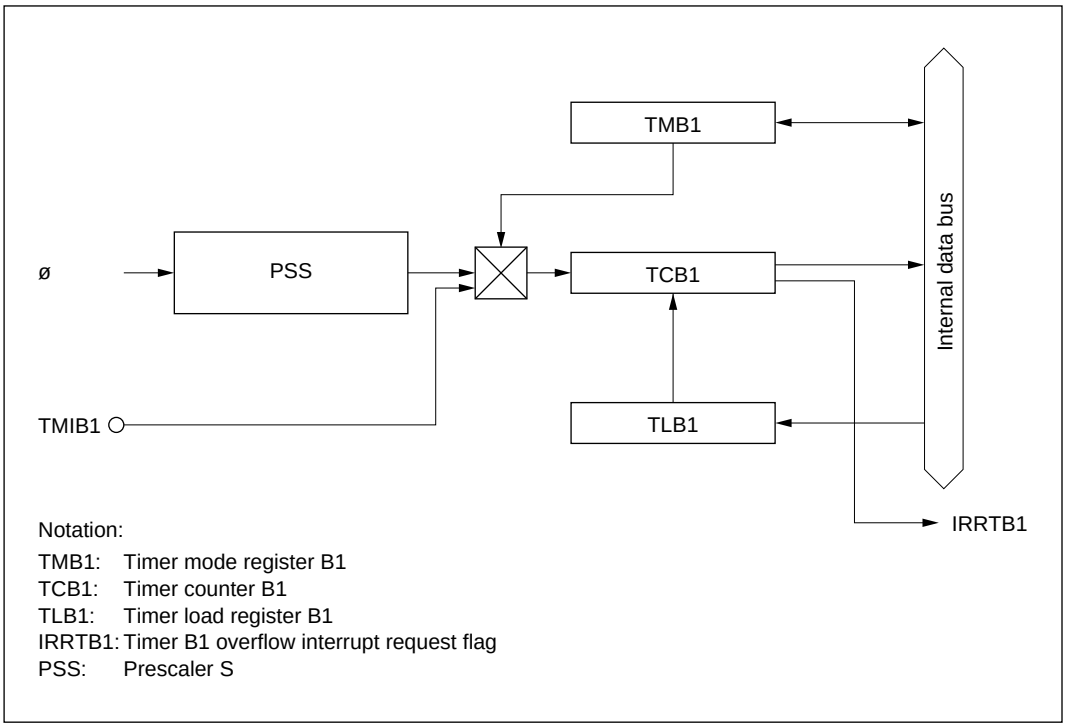


Figure 9-3-1 Block Diagram of Timer B1

3. Pin configuration

Table 9-3-1 shows the timer B1 pin configuration.

Table 9-3-1 Pin Configuration

Name	Abbrev.	I/O	Function
Timer B1 event input	TMIB1	Input	Event input to TCB1

4. Register configuration

Table 9-3-2 shows the register configuration of timer B1.

Table 9-3-2 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

1. 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)
0	0	1	Internal clock: ø/2048	
0	1	0	Internal clock: ø/512	
0	1	1	Internal clock: ø/256	
1	0	0	Internal clock: ø/64	
1	0	1	Internal clock: ø/16	
1	1	0	Internal clock: ø/4	
1	1	1	External event (TMIB1): rising or falling edge*	

Note: * The edge of the external event signal is selected by bit IEG1 in the IRQ edge select register (IEGR). See 3.3.2, Interrupt Control Registers, for details on the IRQ edge select register. Be sure to set bit IRQ1 in port mode register 1 (PMR1) to 1 before setting bits TMB12 to TMB10 to 111.

2. 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 interrupt request register 2 (IRR2) is set to 1.

TCB1 is allocated to the same address as timer load register B1 (TLB1).

Upon reset, TCB1 is initialized to H'00.

3. 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.

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

1. 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 TMIB1. 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 2 (IRR2). If IENTB1 = 1 in interrupt enable register 2 (IENR2), 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.

2. 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.

3. Event counter operation

Timer B1 can operate as an event counter, counting rising or falling edges of an external event signal input at pin TMIB1. External event counting is selected by setting bits TMB12 to TMB10 in timer mode register B1 to all 1s (111).

When timer B1 is used to count external event input, bit IRQ1 in port mode register 1 (PMR1) should be set to 1, and bit IEN1 in interrupt enable register 1 (IENR1) should be cleared to 0 to disable IRQ_1 interrupt requests.

9.3.4 Timer B1 Operation States

Table 9-3-3 summarizes the timer B1 operation states.

Table 9-3-3 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 B2

9.4.1 Overview

Timer B2 is an 8-bit timer that increments each time a clock pulse is input. This timer has two operation modes, interval and auto reload.

1. Features

Features of timer B2 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.
- Choice of two external event input pins.

2. Block diagram

Figure 9-4-1 shows a block diagram of timer B2.

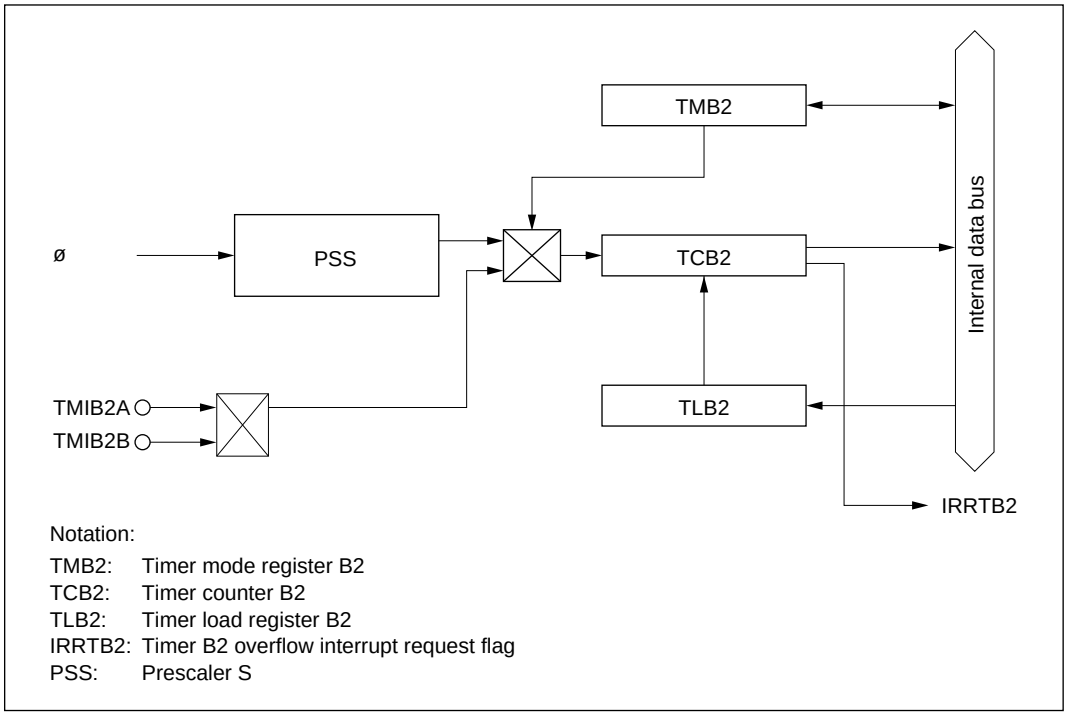


Figure 9-4-1 Block Diagram of Timer B2

3. Pin configuration

Table 9-4-1 shows the timer B2 pin configuration.

Table 9-4-1 Pin Configuration

Name	Abbrev.	I/O	Function
Timer B2 event input A	TMIB2A	Input	Event input to TCB2
Timer B2 event input B	TMIB2B	Input	Event input to TCB2

4. Register configuration

Table 9-4-2 shows the register configuration of timer B2.

Table 9-4-2 Timer B2 Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register B2	TMB2	R/W	H'78	H'FF9C
Timer counter B2	TCB2	R	H'00	H'FF9D
Timer load register B2	TLB2	W	H'00	H'FF9D

9.4.2 Register Descriptions

1. Timer mode register B2 (TMB2)

Bit	7	6	5	4	3	2	1	0
	TMB27	—	—	—	—	TMB22	TMB21	TMB20
Initial value	0	1	1	1	1	0	0	0
Read/Write	R/W	—	—	—	—	R/W	R/W	R/W

TMB2 is an 8-bit read/write register for selecting the auto-reload function and input clock.

Upon reset, TMB2 is initialized to H'78.

Bit 7: Auto-reload function select (TMB27)

Bit 7 selects whether timer B2 is used as an interval timer or auto-reload timer.

Bit 7 TMB27	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 (TMB22 to TMB20)

Bits 2 to 0 select the clock input to TCB2. For external event input there is a choice between counting rising edges or falling edges.

Bit 2 TMB22	Bit 1 TMB21	Bit 0 TMB20	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 (TMIB2A or TMIB2B), rising or falling edge*	

Note: * The edge selection of the external event signal is made by bit EDGB2 in port mode register 2 (PMR2). For details, see 8.3.2 (3), Port Mode Register 2 (PMR2). Be sure to set either bit TMIB2A or TMIB2B to 1 in port mode register 2 (PMR2) before setting bits TMB22 to TMB20 to 111.

2. Timer counter B2 (TCB2)

Bit	7	6	5	4	3	2	1	0
	TCB27	TCB26	TCB25	TCB24	TCB23	TCB22	TCB21	TCB20
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TCB2 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 TMB22 to TMB20 in timer mode register B2 (TMB2). TCB2 values can be read by the CPU at any time.

When TCB2 overflows from H'FF to H'00 or to the value set in TLB2, the IRRTB2 flag in IRR2 is set to 1.

TCB2 is allocated to the same address as TLB2.

Upon reset, TCB2 is initialized to H'00.

3. Timer load register B2 (TLB2)

Bit	7	6	5	4	3	2	1	0
	TLB27	TLB26	TLB25	TLB24	TLB23	TLB22	TLB21	TLB20
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

TLB2 is an 8-bit write-only register for setting the reload value of timer counter B2.

When a reload value is set in TLB2, the same value is loaded into timer counter B2 (TCB2) as well, and TCB2 starts counting up from that value. When TCB2 overflows during operation in auto-reload mode, the TLB2 value is loaded into TCB2. Accordingly, overflow periods can be set within the range of 1 to 256 input clocks.

The same address is allocated to TLB2 as to TCB2.

Upon reset, TLB2 is initialized to H'00.

9.4.3 Timer Operation

1. Interval timer operation

When bit TMB27 in timer mode register B2 (TMB2) is cleared to 0, timer B2 functions as an 8-bit interval timer.

Upon reset, TCB2 is cleared to H'00 and bit TMB27 is cleared to 0, so up-counting and interval timing resume immediately. The clock input to timer B2 is selected from seven internal clock signals output by prescaler S, and an external clock signal from pin TMIB2A or TMIB2B. The selection is made by bits TMB22 to TMB20 of TMB2.

After the count value in TCB2 reaches H'FF, the next clock signal input causes timer B2 to overflow, setting the IRRTB2 flag to 1 in interrupt request register 2 (IRR2). If IENTB2 = 1 in interrupt enable register 2 (IENR2), a CPU interrupt is requested.*

At overflow, TCB2 returns to H'00 and starts counting up again.

During interval timer operation (TMB27 = 0), when a value is set in timer load register B2 (TLB2), the same value is set in TCB2.

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

2. Auto-reload timer operation

Setting bit TMB27 in TMB2 to 1 causes timer B2 to function as an 8-bit auto-reload timer. When a reload value is set in TLB2, the same value is loaded into TCB2, becoming the value from which TCB2 starts its count.

After the count value in TCB2 reaches H'FF, the next clock signal input causes timer B2 to overflow. The TLB2 value is then loaded into TCB2, 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 TLB2 value.

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

In auto-reload mode (TMB27 = 1), when a new value is set in TLB2, the TLB2 value is also set in TCB2.

3. Event counter operation

Timer B2 can operate as an event counter, counting an event signal input at pin TMIB2A or TMIB2B. External event counting is selected by setting bits TMB22 to TMB20 in timer mode register B2 (TMB2) to all 1s (111). TCB2 is incremented at the rising or falling edge of the input at pin TMIB2A or TMIB2B.

The event input pin is selected by setting either bit TMIB2A or TMIB2B in port mode register 2 (PMR2) to 1. Switching event input pins while TCB2 is counting may cause the count value to change. Be sure to disable the interrupt before making the switch, and reset TCB2 (or TLB2) after the switch.

9.4.4 Timer B2 Operation States

Table 9-4-3 summarizes the timer B2 operation states.

Table 9-4-3 Timer B2 Operation States

Operation Mode		Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby
TCB2	Interval	Reset	Functions	Functions	Halted	Halted	Halted	Halted
	Auto reload	Reset	Functions	Functions	Halted	Halted	Halted	Halted
TMB2		Reset	Functions	Retained	Retained	Retained	Retained	Retained

9.5 Timer B3

9.5.1 Overview

Timer B3 is an 8-bit timer that increments each time a clock pulse is input. This timer has two operation modes, interval and auto reload.

1. Features

Features of timer B3 are given below.

- Choice of seven internal clock sources (ø/8192, ø/2048, ø/512, ø/256, ø/64, ø/16, ø/4).
- An interrupt is requested when the counter overflows.

2. Block diagram

Figure 9-5-1 shows a block diagram of timer B3.

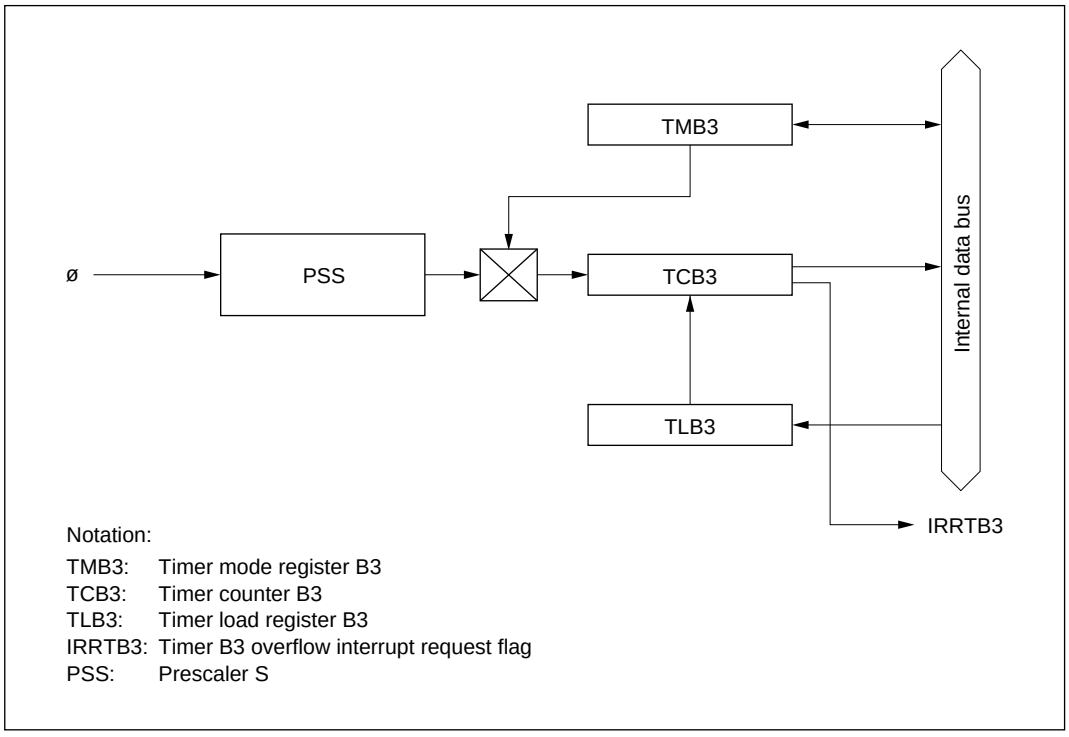


Figure 9-5-1 Block Diagram of Timer B3

3. Register configuration

Table 9-5-1 shows the register configuration of timer B3.

Table 9-5-1 Timer B3 Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register B3	TMB3	R/W	H'78	H'FF9E
Timer counter B3	TCB3	R	H'00	H'FF9F
Timer load register B3	TLB3	W	H'00	H'FF9F

9.5.2 Register Descriptions

1. Timer mode register B3 (TMB3)

Bit	7	6	5	4	3	2	1	0
	TMB37	—	—	—	—	TMB32	TMB31	TMB30
Initial value	0	1	1	1	1	0	0	0
Read/Write	R/W	—	—	—	—	R/W	R/W	R/W

TMB3 is an 8-bit read/write register for selecting the auto-reload function and input clock.

Upon reset, TMB3 is initialized to H'78.

Bit 7: Auto-reload function select (TMB37)

Bit 7 selects whether timer B3 is used as an interval timer or auto-reload timer.

Bit 7 TMB37	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 (TMB32 to TMB30)

Bits 2 to 0 select the clock input to TCB3.

Bit 2 TMB32	Bit 1 TMB31	Bit 0 TMB30	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	Reserved	

2. Timer counter B3 (TCB3)

Bit	7	6	5	4	3	2	1	0
	TCB37	TCB36	TCB35	TCB34	TCB33	TCB32	TCB31	TCB30
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TCB3 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 TMB32 to TMB30 in timer mode register B3 (TMB3). TCB3 values can be read by the CPU at any time.

When TCB3 overflows from H'FF to H'00 or to the value set in TLB3, the IRRTB3 bit in IRR2 is set to 1.

TCB3 is allocated to the same address as TLB3.

Upon reset, TCB3 is initialized to H'00.

3. Timer load register B3 (TLB3)

Bit	7	6	5	4	3	2	1	0
	TLB37	TLB36	TLB35	TLB34	TLB33	TLB32	TLB31	TLB30
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

TLB3 is an 8-bit write-only register for setting the reload value of timer counter B3.

When a reload value is set in TLB3, the same value is loaded into timer counter B3 (TCB3) as well, and TCB3 starts counting up from that value. When TCB3 overflows during operation in auto-reload mode, the TLB3 value is loaded into TCB3. Accordingly, overflow periods can be set within the range of 1 to 256 input clocks.

The same address is allocated to TLB3 as to TCB3.

Upon reset, TLB3 is initialized to H'00.

9.5.3 Timer Operation

1. Interval timer operation

When bit TMB37 in timer mode register B3 (TMB3) is cleared to 0, timer B3 functions as an 8-bit interval timer.

Upon reset, TCB3 is cleared to H'00 and bit TMB37 is cleared to 0, so up-counting and interval timing resume immediately. The clock input to timer B3 is selected from seven internal clock signals output by prescaler S. The selection is made by bits TMB32 to TMB30 of TMB3.

After the count value in TCB3 reaches H'FF, the next clock signal input causes timer B3 to overflow, setting bit IRRTB3 to 1 in interrupt request register 1 (IRR1). If IENTB3 = 1 in interrupt enable register 2 (IENR2), a CPU interrupt is requested.*

At overflow, TCB3 returns to H'00 and starts counting up again.

During interval timer operation (TMB37 = 0), when a value is set in timer load register B3 (TLB3), the same value is set in TCB3.

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

2. Auto-reload timer operation

Setting bit TMB37 in TMB3 to 1 causes timer B3 to function as an 8-bit auto-reload timer. When a reload value is set in TLB3, the same value is loaded into TCB3, becoming the value from which TCB3 starts its count.

After the count value in TCB3 reaches H'FF, the next clock signal input causes timer B3 to overflow. The TLB3 value is then loaded into TCB3, 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 TLB3 value.

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

In auto-reload mode (TMB37 = 1), when a new value is set in TLB3, the TLB3 value is also set in TCB3.

9.5.4 Timer B3 Operation States

Table 9-5-2 summarizes the timer B3 operation states.

Table 9-5-2 Timer B3 Operation States

Operation Mode		Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby
TCB3	Interval	Reset	Functions	Functions	Halted	Halted	Halted	Halted
	Auto reload	Reset	Functions	Functions	Halted	Halted	Halted	Halted
TMB3		Reset	Functions	Retained	Retained	Retained	Retained	Retained

9.6 Timer C

9.6.1 Overview

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

1. Features

The main features of timer C are given below.

- Choice of seven internal clock sources (ø/8192, ø/2048, ø/512, ø/64, ø/16, ø/4, $ø_W$/4) or an external clock (can be used to count external events).
- An interrupt is requested when the counter overflows or underflows.
- Can be switched between up- and down-counting by software or hardware.
- When $ø_W$/4 is selected as the internal clock source, or when an external clock is selected, timer C can function in subactive mode and subsleep mode.

2. Block diagram

Figure 9-6-1 shows a block diagram of timer C.

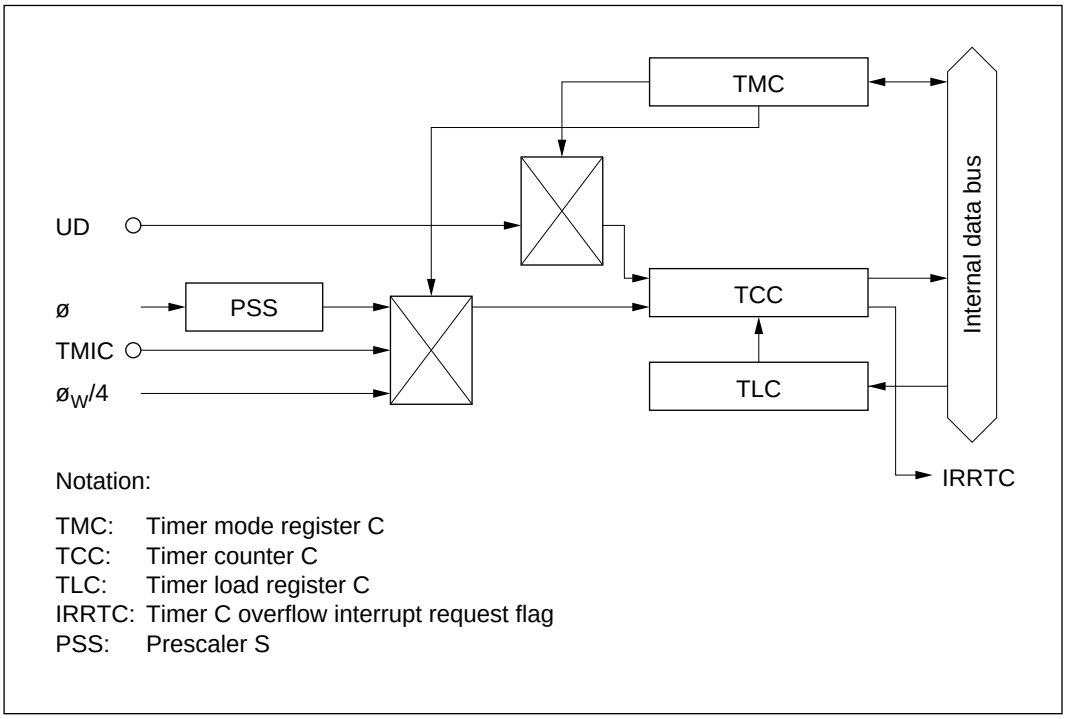


Figure 9-6-1 Block Diagram of Timer C

3. Pin configuration

Table 9-6-1 shows the timer C pin configuration.

Table 9-6-1 Pin Configuration

Name	Abbrev.	I/O	Function
Timer C event input	TMIC	Input	Event input to TCC
Timer C up/down select	UD	Input	Selection of counting direction

4. Register configuration

Table 9-6-2 shows the register configuration of timer C.

Table 9-6-2 Timer C Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer mode register C	TMC	R/W	H'18	H'FFB4
Timer counter C	TCC	R	H'00	H'FFB5
Timer load register C	TLC	W	H'00	H'FFB5

9.6.2 Register Descriptions

1. Timer mode register C (TMC)

Bit	7	6	5	4	3	2	1	0
	TMC7	TMC6	TMC5	—	—	TMC2	TMC1	TMC0
Initial value	0	0	0	1	1	0	0	0
Read/Write	R/W	R/W	R/W	—	—	R/W	R/W	R/W

TMC is an 8-bit read/write register for selecting the auto-reload function, counting direction, and input clock.

Upon reset, TMC is initialized to H'18.

Bit 7: Auto-reload function select (TMC7)

Bit 7 selects whether timer C is used as an interval timer or auto-reload timer.

Bit 7 TMC7	Description	
0	Interval timer function selected	(initial value)
1	Auto-reload function selected	

Bits 6 and 5: Counter up/down control (TMC6 and TMC5)

These bits select the counting direction of timer counter C (TCC), or allow hardware to control the counting direction using pin UD.

Bit 6 TMC6	Bit 5 TMC5	Description	
0	0	TCC is an up-counter	(initial value)
0	1	TCC is a down-counter	
1	*	TCC up/down control is determined by input at pin UD. TCC is a down-counter if the UD input is high, and an up-counter if the UD input is low.	

Note: * Don't care

Bits 4 and 3: Reserved bits

Bits 4 and 3 are reserved; they are always read as 1, and cannot be modified.

Bits 2 to 0: Clock select (TMC2 to TMC0)

Bits 2 to 0 select the clock input to TCC. For external clock counting, either the rising or falling edge can be selected.

Bit 2 TMC2	Bit 1 TMC1	Bit 0 TMC0	Description	
0	0	0	Internal clock: ø/8192	(initial value)
0	0	1	Internal clock: ø/2048	
0	1	0	Internal clock: ø/512	
0	1	1	Internal clock: ø/64	
1	0	0	Internal clock: ø/16	
1	0	1	Internal clock: ø/4	
1	1	0	Internal clock: $ø_W/4$	
1	1	1	External event (TMIC): rising or falling edge*	

Note: * The edge of the external event signal is selected by bit IEG2 in the IRQ edge select register (IEGR). See 3.3.2, Interrupt Control Registers, for details on the IRQ edge select register. Be sure to set bit IRQ2 in port mode register 1 (PMR1) to 1 before setting bits TMC2 to TMC0 to 111.

2. Timer counter C (TCC)

Bit	7	6	5	4	3	2	1	0
	TCC7	TCC6	TCC5	TCC4	TCC3	TCC2	TCC1	TCC0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TCC is an 8-bit read-only up-/down-counter, which is incremented or decremented by internal or external clock input. The clock source for input to this counter is selected by bits TMC2 to TMC0 in timer mode register C (TMC). TCC values can be read by the CPU at any time.

When TCC overflows (from H'FF to H'00 or to the value set in TLC) or underflows (from H'00 to H'FF or to the value set in TLC), the IRRTC bit in interrupt request register 2 (IRR2) is set to 1.

TCC is allocated to the same address as timer load register C (TLC).

Upon reset, TCC is initialized to H'00.

3. Timer load register C (TLC)

Bit	7	6	5	4	3	2	1	0
	TLC7	TLC6	TLC5	TLC4	TLC3	TLC2	TLC1	TLC0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

TLC is an 8-bit write-only register for setting the reload value of TCC. When a reload value is set in TLC, the same value is loaded into timer counter C (TCC) as well, and TCC starts counting up or down from that value. When TCC overflows or underflows during operation in auto-reload mode, the TLC value is loaded into TCC. Accordingly, overflow and underflow periods can be set within the range of 1 to 256 input clocks.

The same address is allocated to TLC as to TCC.

Upon reset, TLC is initialized to H'00.

9.6.3 Timer Operation

1. Interval timer operation

When bit TMC7 in timer mode register C (TMC) is cleared to 0, timer C functions as an 8-bit interval timer.

Upon reset, timer counter C (TCC) is initialized to H'00 and TMC to H'18, so counting and interval timing resume immediately. The clock input to timer C is selected from seven internal clock signals, or an external clock input at pin TMIC. The selection is made by bits TMC2 to TMC0 in TMC.

Either software or hardware can control whether TCC counts up or down. The selection is made by TMC bits TMC6 and TMC5.

After the count value in TCC reaches H'FF (H'00), the next clock signal input causes timer C to overflow (underflow), setting bit IRRTC to 1 in interrupt request register 2 (IRR2). If IENTC = 1 in interrupt enable register 2 (IENR2), a CPU interrupt is requested.*

At overflow or underflow, TCC returns to H'00 or H'FF and starts counting up or down again.

During interval timer operation (TMC7 = 0), when a value is set in timer load register C (TLC), the same value is set in TCC.

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

2. Auto-reload timer operation

Setting bit TMC7 in TMC to 1 causes timer C to function as an 8-bit auto-reload timer. When a reload value is set in TLC, the same value is loaded into TCC, becoming the value from which TCC starts its count.

After the count value in TCC reaches H'FF (H'00), the next clock signal input causes timer C to overflow (underflow). The TLC value is then loaded TCC, and the count continues from that value. The overflow (underflow) period can be set within a range from 1 to 256 input clocks, depending on the TLC value.

The clock sources, up/down control, and interrupts in auto-reload mode are the same as in interval mode.

In auto-reload mode (TMC7 = 1), when a new value is set in TLC, the TLC value is also set in TCC.

3. Event counter operation

Timer C can operate as an event counter, counting an event signal input at pin TMIC. External event counting is selected by setting TMC bits TMC2 to TMC0 to all 1s (111). TCC counts up or down at the rising or falling edge of the input at pin TMIC.

When timer C is used to count external event inputs, bit IRQ2 in port mode register 1 (PMR1) should be set to 1, and bit IEN2 in interrupt enable register 1 (IENR1) should be cleared to 0 to disable IRQ_2 interrupt requests.

4. TCC up/down control by hardware

The counting direction of timer C can be controlled by input at pin UD. When bit TMC6 in TMC is set to 1, high-level input at the UD pin selects down-counting, while low-level input selects up-counting.

When using input at pin UD for this control function, set the UD bit in port mode register 2 (PMR2) to 1.

9.6.4 Timer C Operation States

Table 9-6-3 summarizes the timer C operation states.

Table 9-6-3 Timer C Operation States

Operation Mode		Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCC	Interval	Reset	Functions	Functions	Halted	Functions/ Halted*	Functions/ Halted*	Halted
TCC	Auto reload	Reset	Functions	Functions	Halted	Functions/ Halted*	Functions/ Halted*	Halted
TMC		Reset	Functions	Retained	Retained	Functions	Retained	Retained

Note: When $\phi_W/4$ is selected as the internal clock of TCC 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.

* When timer C is operated in subactive mode or subsleep mode, either an external clock or the $\phi_W/4$ internal clock must be selected. The counter will not operate in these modes if another clock is selected. If the internal $\phi_W/4$ clock is selected when $\phi_W/8$ is being used as the subclock ϕ_{SUB}, the lower 2 bits of the counter will operate on the same cycle, with the least significant bit not being counted.

9.7 Timer F

9.7.1 Overview

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

1. Features

Features of timer F are given below.

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

Timer FH

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

Timer FL

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

2. Block diagram

Figure 9-7-1 shows a block diagram of timer F.

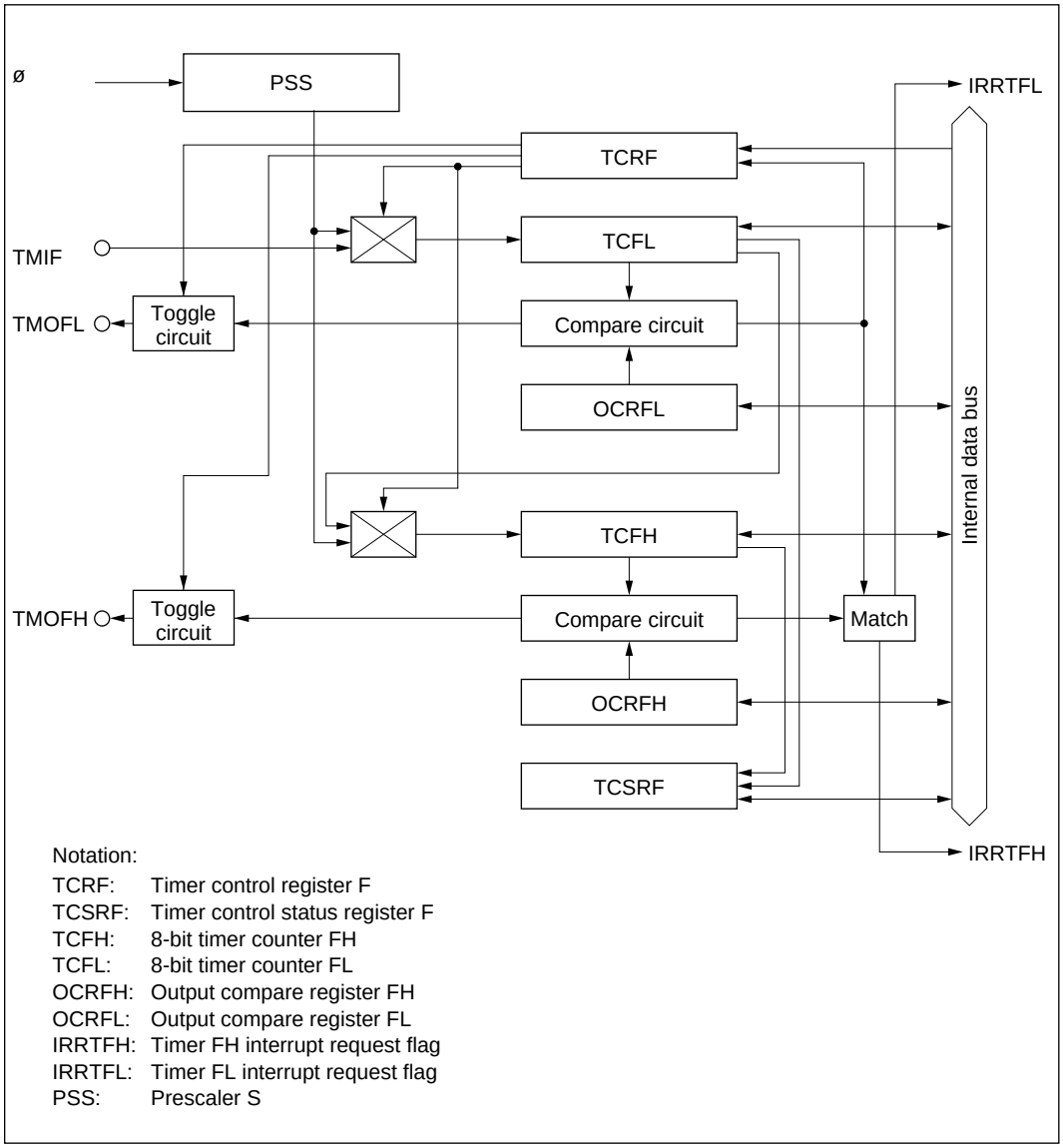


Figure 9-7-1 Block Diagram of Timer F

3. Pin configuration

Table 9-7-1 shows the timer F pin configuration.

Table 9-7-1 Pin Configuration

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

4. Register configuration

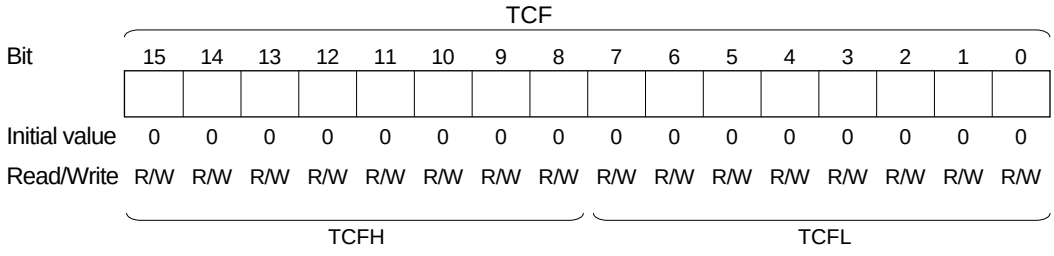
Table 9-7-2 shows the register configuration of timer F.

Table 9-7-2 Timer F Registers

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

9.7.2 Register Descriptions

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



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

TCFH and TCFL can be read and written by the CPU, but in 16-bit mode, data transfer with the CPU takes place via a temporary register (TEMP). For details see 9.7.3, Interface with the CPU.

- 16-bit mode (TCF)

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

Bit CCLRH in TCSRF can be set so that counter TCF will be cleared by compare match.

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

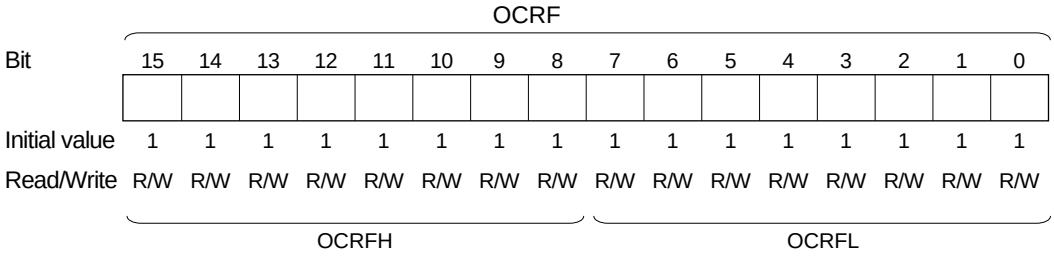
- 8-bit mode (TCFH, TCFL)

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

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

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

2. 16-bit output compare register F (OCRF)
   8-bit output compare register FH (OCRFH)
   8-bit output compare register FL (OCRFL)



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

OCRFH and OCRFL can be read and written by the CPU, but in 16-bit mode, data transfer with the CPU takes place via a temporary register (TEMP). For details see 9.7.3, Interface with the CPU.

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

- 16-bit mode (OCRF)

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

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

- 8-bit mode (OCRFH, OCRFL)

Setting bit CKSH2 in TCRF to 1 results in two independent output compare registers, OCRFH and OCRFL.

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

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

3. Timer control register F (TCRF)

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

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

Upon reset, TCRF is initialized to H'00.

Bit 7: Toggle output level H (TOLH)

Bit 7 sets the output level at pin TMOFH. The setting goes into effect immediately after this bit is written.

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

Bits 6 to 4: Clock select H (CKSH2 to CKSH0)

Bits 6 to 4 select the input to TCFH from four internal clock signals or the overflow of TCFL.

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

Note: * Don’t care

Bit 3: Toggle output level L (TOLL)

Bit 3 sets the output level at pin TMOFL. The setting goes into effect immediately after this bit is written.

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

Bits 2 to 0: Clock select L (CKSL2 to CKSL0)

Bits 2 to 0 select the input to TCFH from four internal clock signals or external event input.

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

* Don't care

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

4. Timer control/status register F (TCSRF)

Bit	7	6	5	4	3	2	1	0
	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/W	R/W	R/(W)*	R/(W)*	R/W	R/W

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

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

Upon reset or in standby mode, TCSRF is initialized to H'00.

Bit 7: Timer overflow flag H (OVFH)

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

Bit 7 OVFH	Description	
0	Clearing conditions: After reading OVFH = 1, cleared by writing 0 to OVFH	(initial value)
1	Setting conditions: 16-bit mode: set when the value of TCF goes from H'FFFF to H'0000 8-bit mode: set when the value of TCFH goes from H'FF to H'00	

Bit 6: Compare match flag H (CMFH)

Bit 6 is a status flag indicating a compare match between TCFH and OCRFH. This flag is set by hardware and cleared by software. It cannot be set by software.

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

Bit 5: Timer overflow interrupt enable H (OVIEH)

Bit 5 enables or disables TCFH overflow interrupts.

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

Bit 4: Counter clear H (CCLRH)

In 16-bit mode, bit 4 selects whether or not TCF is cleared when a compare match occurs between TCF and OCRF.

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

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

Bit 3: Timer overflow flag L (OVFL)

Bit 3 is a status flag indicating TCFL overflow (H'FF to H'00). This flag is set by hardware and cleared by software. It cannot be set by software.

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

Bit 2: Compare match flag L (CMFL)

Bit 2 is a status flag indicating a compare match between TCFL and OCRFL. This flag is set by hardware and cleared by software. It cannot be set by software.

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

Bit 1: Timer overflow interrupt enable L (OVIEL)

Bit 1 enables or disables TCFL overflow interrupts.

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

Bit 0: Counter clear L (CCLRL)

Bit 0 selects whether or not TCFL is cleared when a compare match occurs between TCFL and OCRFL.

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

9.7.3 Interface with the CPU

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

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

- Write access

When the upper byte is written, the upper-byte data is loaded into the TEMP register. Next when the lower byte is written, the data in TEMP goes to the upper byte of the register, and the lower-byte data goes directly to the lower byte of the register. Figure 9-7-2 shows a TCF write operation when H'AA55 is written to TCF.

- Read access

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

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

Figure 9-7-3 shows a TCF read operation when H'AAFF is read from TCF.

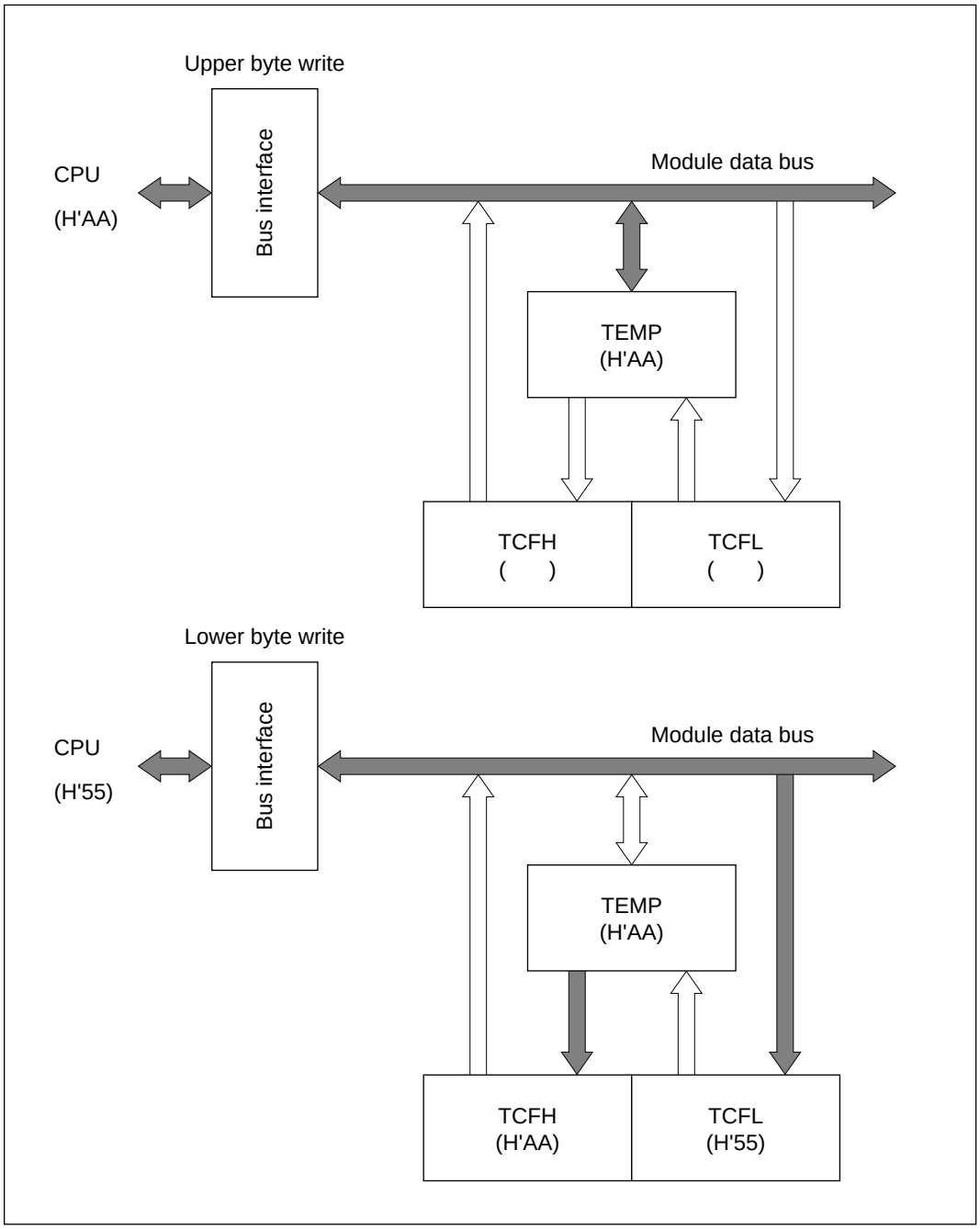


Figure 9-7-2 TCF Write Operation (CPU → TCF)

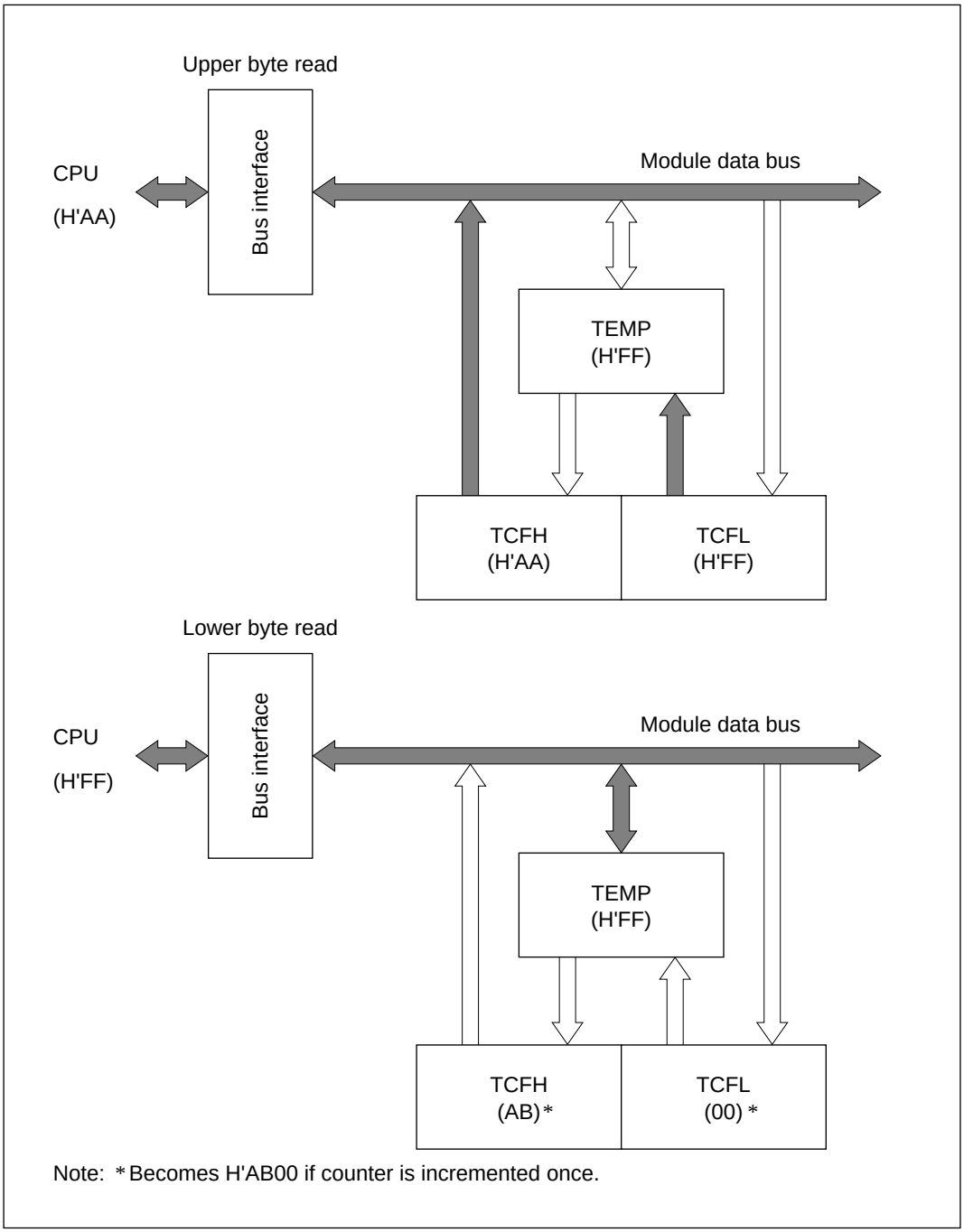


Figure 9-7-3 TCF Read Operation (TCF → CPU)

9.7.4 Timer Operation

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

1. Timer F operation

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

- 16-bit timer mode

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

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

Instead of counting external events, timer F can be switched by bits CKSL2 to CKSL0 in TCRF to count one of four internal clocks output by prescaler S.

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

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

- 8-bit timer mode

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

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

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

2. TCF count timing

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

- Internal clock

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

- External event signal

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

3. TMOFH and TMOFL output timing

The outputs at pins TMOFH and TMOFL are the values set in bits TOLH and TOLL in TCRF. When a compare match occurs, the output value is inverted. Figure 9-7-4 shows the output timing.

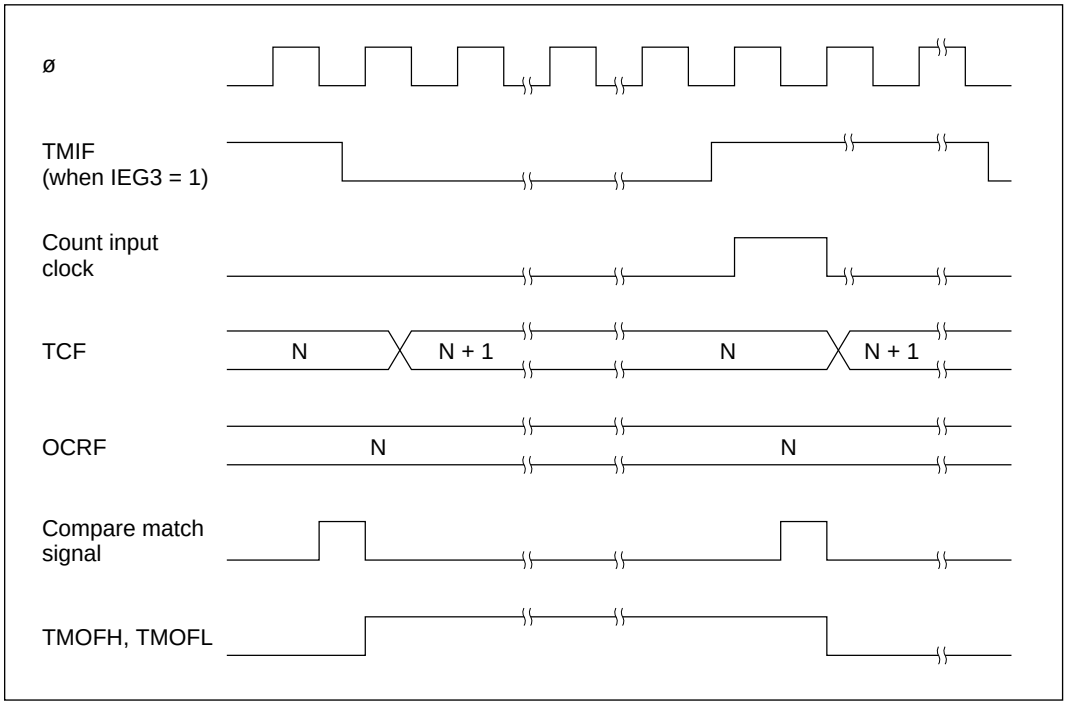


Figure 9-7-4 TMOFH, TMOFL Output Timing

4. Compare match flag set timing

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

7. Timer F operation states

Table 9-7-3 summarizes the timer F operation states.

Table 9-7-3 Timer F Operation States

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

9.7.5 Application Notes

The following conflicts can arise in timer F operation.

1. 16-bit timer mode

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

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

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

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

2. 8-bit timer mode

TCFH and OCRFH

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

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

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

TCFL and OCRFL

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

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

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

9.8 Timer G

9.8.1 Overview

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

1. Features

Features of timer G are given below.

- Choice of four internal clock sources (ø/64, ø/32, ø/2, $ø_W$/2)
- Input capture function

  Separate input capture registers are provided for the rising and falling edges.
- Counter overflow level detection

  Can detect whether overflow occurred when the input capture signal was high or low.
- Choice of counter clear triggers

  The counter can be cleared at the rising edge, falling edge, both edges, or neither edge of the input capture signal.
- Two interrupt sources

  Interrupts can be requested by input capture and by overflow. For input capture, the rising or falling edge can be selected.
- Built-in noise-canceller circuit

  The noise canceller circuit can eliminate high-frequency noise in the input capture signal.
- Operates in subactive and subsleep modes

  When $ø_W$/2 is selected as the internal clock source, timer G can operate in the subactive and subsleep modes.

2. Block diagram

Figure 9-8-1 shows a block diagram of timer G.

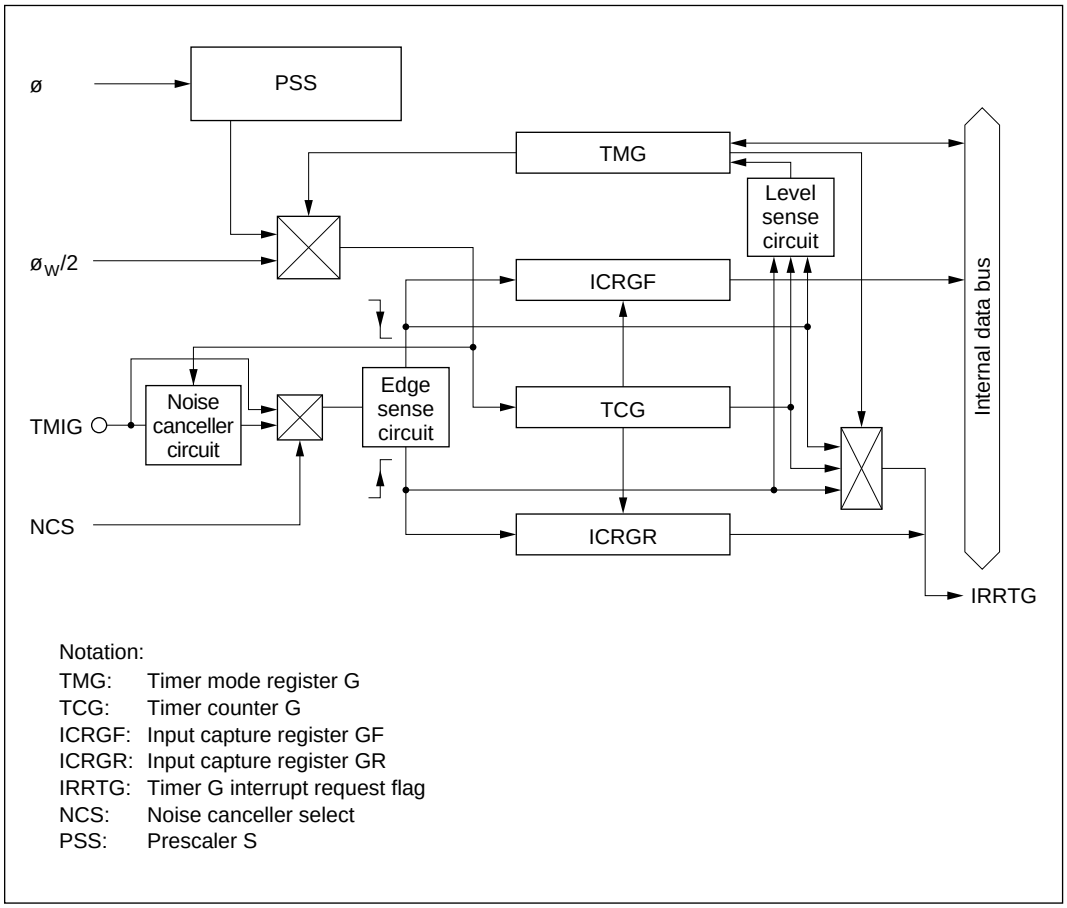


Figure 9-8-1 Block Diagram of Timer G

3. Pin configuration

Table 9-8-1 shows the timer G pin configuration.

Table 9-8-1 Pin Configuration

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

4. Register configuration

Table 9-8-2 shows the register configuration of timer G.

Table 9-8-2 Timer G Registers

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

9.8.2 Register Descriptions

1. Timer counter G (TCG)

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

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

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

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

TCG cannot be read or written by the CPU.

Upon reset, TCG is initialized to H'00.

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

2. Input capture register GF (ICRGF)

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

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

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

Upon reset, ICRGF is initialized to H'00.

3. Input capture register GR (ICRGR)

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

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

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

Upon reset, ICRGR is initialized to H'00.

4. Timer mode register G (TMG)

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

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

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

Upon reset, TMG is initialized to H'00.

Bit 7: Timer overflow flag H (OVFH)

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

Bit 7 OVFH	**Description**
0	Clearing conditions: (initial value) After reading OVFH = 1, cleared by writing 0 to OVFH
1	Setting conditions: Set when the value of TCG overflows from H'FF to H'00 while the input capture signal is high

Bit 6: Timer overflow flag L (OVFL)

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

Bit 6 OVFL	**Description**
0	Clearing conditions: (initial value) After reading OVFL = 1, cleared by writing 0 to OVFL
1	Setting conditions: Set when the value of TCG overflows from H'FF to H'00 while the input capture signal is low

Bit 5: Timer overflow interrupt enable (OVIE)

Bit 5 enables or disables TCG overflow interrupts.

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

Bit 4: Input capture interrupt edge select (IIEGS)

Bit 4 selects the input signal edge at which input capture interrupts are requested.

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

Bits 3, 2: Counter clear 1, 0 (CCLR1, CCLR0)

Bits 3 and 2 designate whether TCG is cleared at the rising, falling, or both edges of the input capture signal, or is not cleared.

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

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

Bits 1 and 0 select the clock input to TCG from four internal clock signals.

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	Internal clock: ø/64	(initial value)
0	1	Internal clock: ø/32	
1	0	Internal clock: ø/2	
1	1	Internal clock: $ø_W$/2	

9.8.3 Noise Canceller Circuit

The noise canceller circuit built into the H8/3947 Series is a digital low-pass filter that rejects high-frequency pulse noise in the input at the input capture pin. The noise canceller circuit is enabled by the noise canceller select (NCS) bit in port mode register 2 (PMR2)*.

Figure 9-8-2 shows a block diagram of the noise canceller circuit.

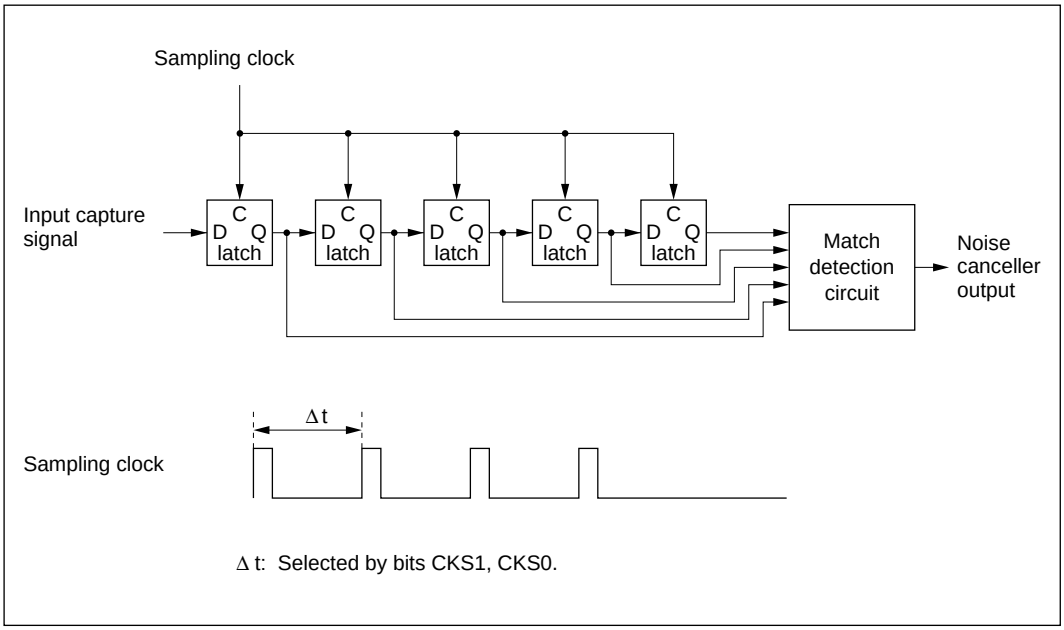


Figure 9-8-2 Block Diagram of Noise Canceller Circuit

The noise canceller consists of five latch circuits connected in series, and a match detection circuit. When the noise canceller function is disabled (NCS = 0), the system clock is selected as the sampling clock. When the noise canceller is enabled (NCS = 1), the internal clock selected by bits CKS1 and CKS0 in TMG becomes the sampling clock. The input signal is sampled at the rising edge of this clock pulse. Data is considered correct when the outputs of all five latch circuits match. If they do not match, the previous value is retained. Upon reset, the noise canceller output is initialized after the falling edge of the input capture signal has been sampled five times. Accordingly, after the noise canceller function is enabled, pulses that have a pulse width five times greater than the sampling clock will be recognized as input capture signals. If the noise canceller circuit is not used, the input capture signal pulse width must be at least 2ø or $2ø_{SUB}$ in order to ensure proper input capture operation.

Note: * Rewriting the NCS bit may cause an internal input capture signal to be generated.

Figure 9-8-3 shows a typical timing diagram for the noise canceller circuit. In this example, a high-level input at the input capture pin is rejected as noise because its pulse width is less than five sampling clock cycles.

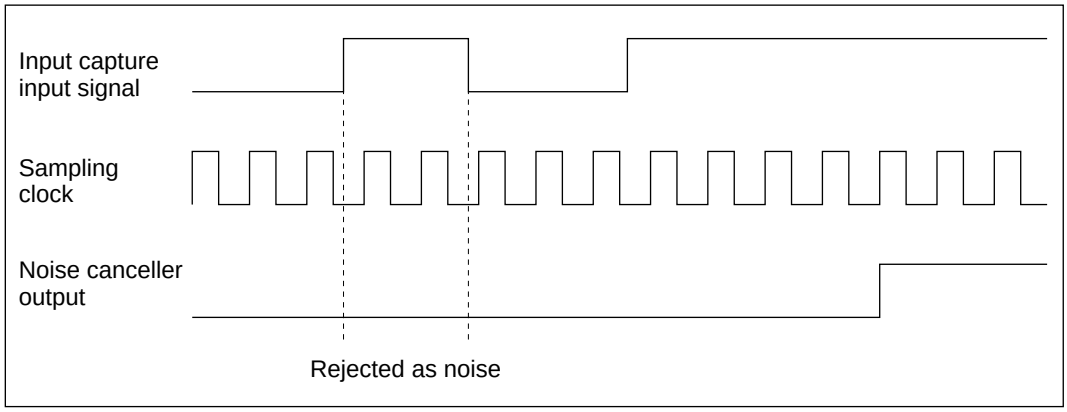


Figure 9-8-3 Noise Canceller Circuit Timing (Example)

9.8.4 Timer Operation

Timer G is an 8-bit timer with input capture and interval timer functions.

1. Timer G functions

Timer G is an 8-bit timer/counter that functions as an input capture timer or an interval timer. These two functions are described below.

- Input capture timer operation

Timer G functions as an input capture timer when bit TMIG of port mode register 1 (PMR1) is set to 1.*

At reset, timer mode register G (TMG), timer counter G (TCG), input capture register GF (ICRGF), and input capture register GR (ICRGR) are all initialized to H'00.

Immediately after reset, TCG begins counting an internal clock with a frequency of ø divided by 64 (ø/64). Three other internal clocks can be selected using bits CKS1 and CKS0 of TMG.

At the rising edge/falling edge of the input capture signal input to pin TMIG, the value of TCG is copied into ICRGR/ICRGF. If the input edge is the same as the edge selected by the IIEGS bit of TMG, then bit IRRTG is set to 1 in IRR2. If bit IENTG is also set to 1 in IENR2, a CPU interrupt is requested. For details on interrupts, see section 3.3, Interrupts.

TCG can be cleared to 0 at the rising edge, falling edge, or both edges of the input capture signal as determined with bits CCLR1 and CCLR0 of TMG. If TCG overflows while the input capture

signal is high, bit OVFH of TMG is set. If TCG overflows while the input capture signal is low, bit OVFL of TMG is set. When either of these bits is set, if bit OVIE of TMG is currently set to 1, then bit IRRTG is set to 1 in IRR2. If bit IENTG is also set to 1 in IENR2, then timer G requests a CPU interrupt. For further details see 3.3, Interrupts.

Timer G has a noise canceller circuit that rejects high-frequency pulse noise in the input to pin TMIG. See 9.8.3, Noise Canceller Circuit, for details.

Note: * Rewriting the TMIG bit may cause an internal input capture signal to be generated.

- Interval timer operation

Timer G functions as an interval timer when bit TMIG is cleared to 0 in PMR1. Following a reset, TCG starts counting cycles of the ø/64 internal clock. This is one of four internal clock sources that can be selected by bits CKS1 and CKS0 of TMG. TCG counts up according to the selected clock source. When it overflows from H'FF to H'00, bit OVFL of TMG is set to 1. If bit OVIE of TMG is currently set to 1, then bit IRRTG is set to 1 in IRR2. If bit IENTG is also set to 1 in IENR2, then timer G requests a CPU interrupt. For further details see 3.3, Interrupts.

2. Count timing

TCG is incremented by input pulses from an internal clock. TMG bits CKS1 and CKS0 select one of four internal clocks (ø/64, ø/32, ø/2, $ø_W$/2) derived by dividing the system clock (ø) or the watch clock ($ø_W$).

3. Timing of internal input capture signals

- Timing with noise canceller function disabled

Separate internal input capture signals are generated from the rising and falling edges of the external input signal.

Figure 9-8-4 shows the timing of these signals.

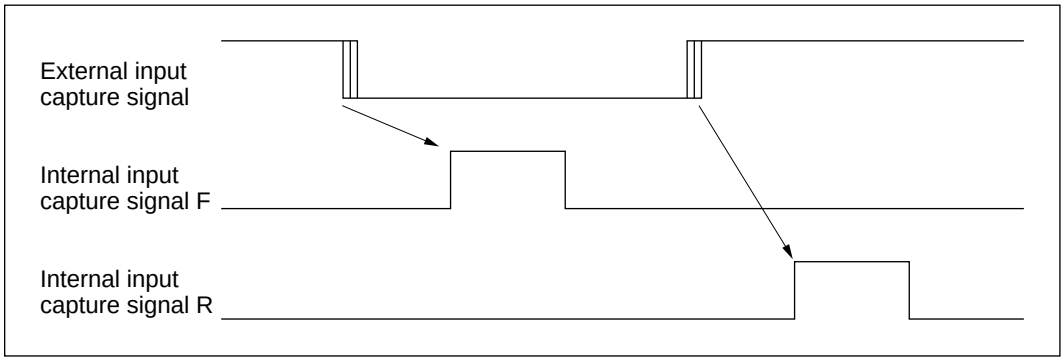


Figure 9-8-4 Input Capture Signal Timing (Noise Canceller Function Disabled)

- Timing with noise canceller function enabled

When input capture noise cancelling is enabled, the external input capture signal is routed via the noise canceller circuit, so the internal signals are delayed from the input edge by five sampling clock cycles. Figure 9-8-5 shows the timing.

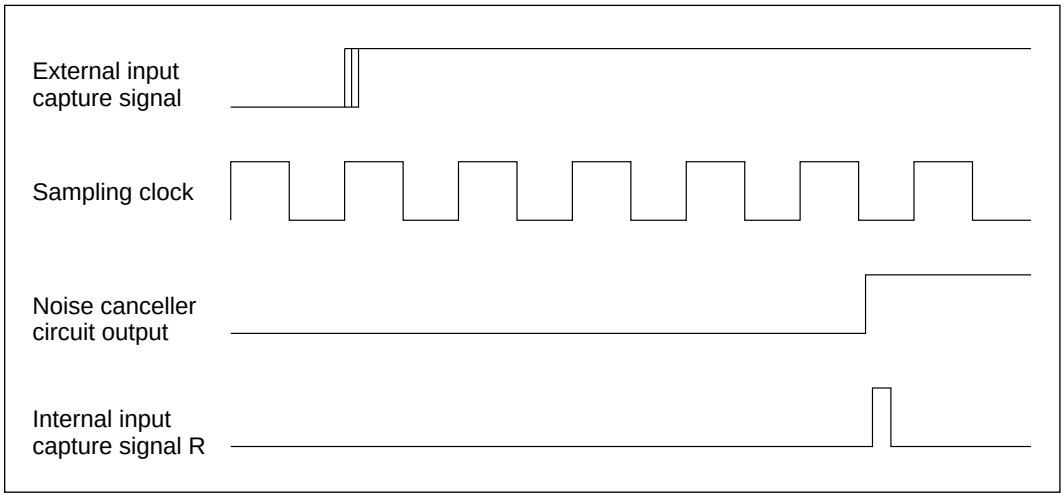


Figure 9-8-5 Input Capture Signal Timing (Noise Canceller Function Enabled)

4. Timing of input capture

Figure 9-8-6 shows the input capture timing in relation to the internal input capture signal.

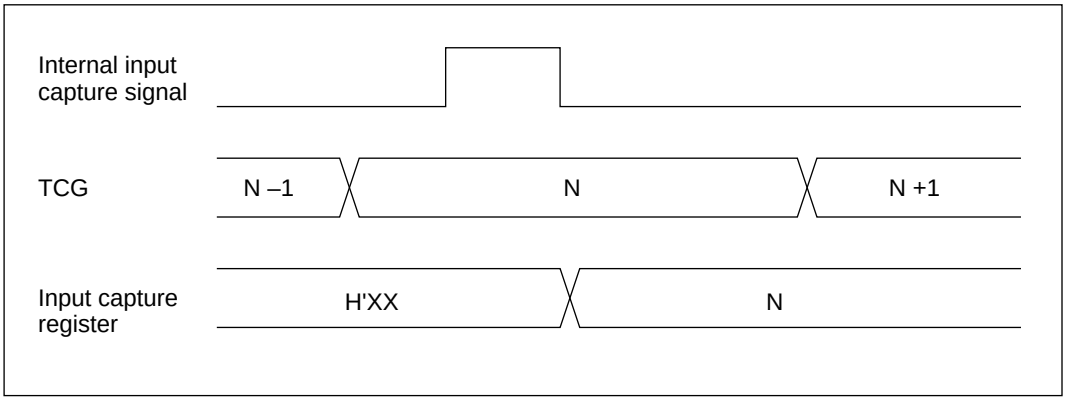


Figure 9-8-6 Input Capture Timing

5. TCG clear timing

TCG can be cleared at the rising edge, falling edge, or both edges of the external input capture signal. Figure 9-8-7 shows the timing for clearing at both edges.

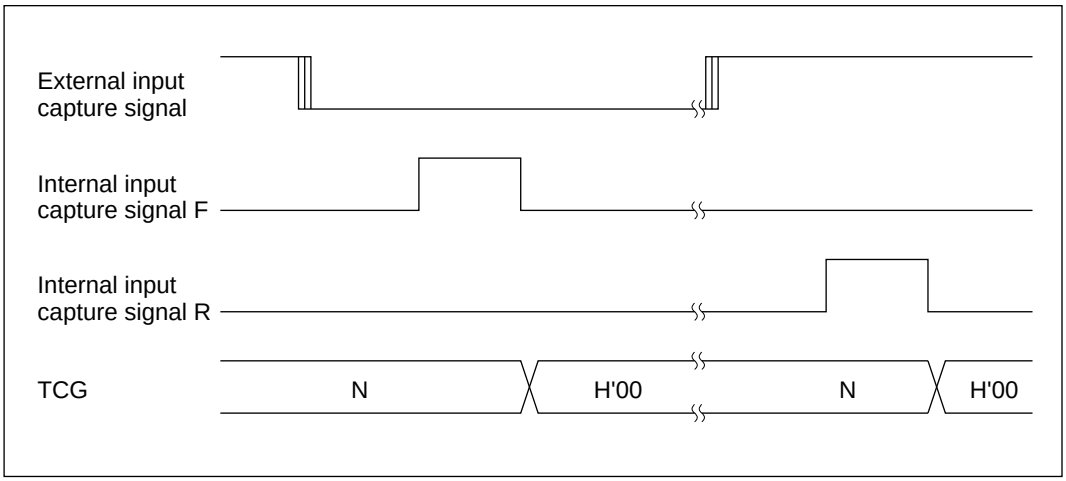


Figure 9-8-7 TCG Clear Timing

6. Timer G operation states

Table 9-8-3 summarizes the timer G operation states.

Table 9-8-3 Timer G Operation States

Operation Mode		Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby
TCG	Input capture	Reset	Functions*	Functions*	Halted	Functions/ Halted*	Functions/ Halted*	Halted
	Interval	Reset	Functions*	Functions*	Retained	Functions/ Halted*	Functions/ Halted*	Halted
ICRGF		Reset	Functions*	Functions*	Retained	Functions/ Halted*	Functions/ Halted*	Retained
ICRGR		Reset	Functions*	Functions*	Retained	Functions/ Halted*	Functions/ Halted*	Retained
TMG		Reset	Functions	Retained	Retained	Functions	Retained	Retained

Note: * In active mode and sleep mode, if $ø_W/2$ is selected as the TCG internal clock, since the system clock and internal clock are not synchronized with each other, a synchronization circuit is used. This may result in a count cycle error of up to 1/ø (s). In subactive mode and subsleep mode, if $ø_W/2$ is selected as the TCG internal clock, regardless of the subclock ($ø_W/2$, $ø_W/4$, $ø_W/8$) TCG and the noise canceller circuit run on an internal clock of $ø_W/2$. If any other internal clock is chosen, TCG and the noise canceller circuit will not run, and the input capture function will not operate.

9.8.5 Sample Timer G Application

The absolute values of the high and low widths of the input capture signal can be measured by using timer G. The CCLR1 and CCLR0 bits of TMG should be set to 1. Figure 9-8-8 shows an example of this operation.

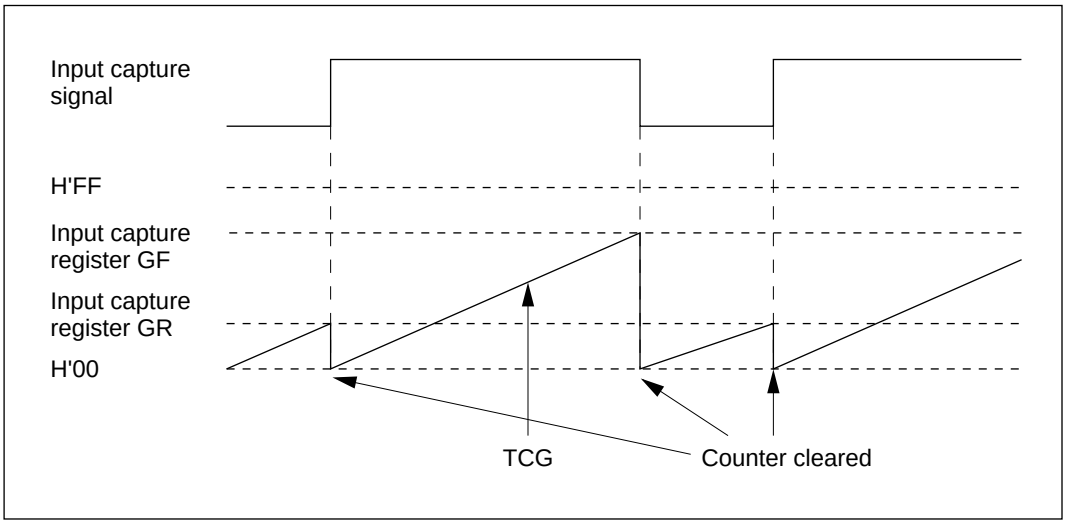


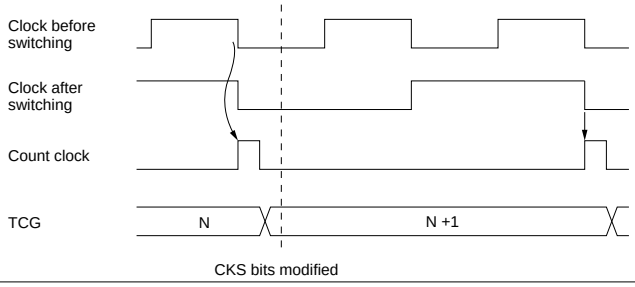
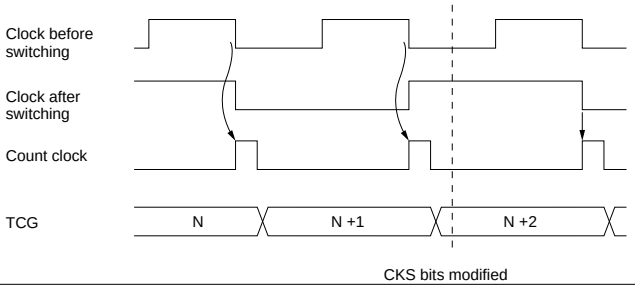
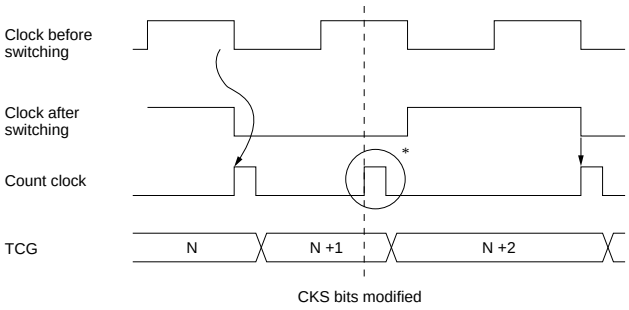
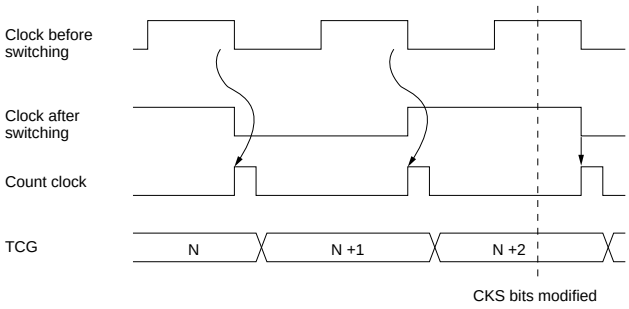
Figure 9-8-8 Sample Timer G Application

9.8.6 Application Notes

1. Input clock switching and TCG operation

Depending on when the input clock is switched, there will be cases in which TCG is incremented in the process. Table 9-8-4 shows the relation between internal clock switchover timing (selected in bits CKS1 and CKS0) and TCG operation. If an internal clock (derived from the system clock ø or subclock $ø_{SUB}$) is used, an increment pulse is generated when a falling edge of the internal clock is detected. For this reason, in a case like No. 3 in table 9-8-4, where the clock is switched at a time such that the clock signal goes from high level before switching to low level after switching, the switchover is seen as a falling edge of the clock pulse, causing TCG to be incremented.

Table 9-8-4 Internal Clock Switching and TCG Operation

No.	Clock Level Before and After Modifying Bits CKS1 and CKS0	TCG Operation
1	Goes from low level to low level	
2	Goes from low level to high level	
3	Goes from high level to low level	
4	Goes from high level to high level	

Note: * The switchover is seen as a falling edge of the clock pulse, and TCG is incremented.

2. Note on rewriting port mode registers

When a port mode register setting is modified to enable or disable the input capture function or input capture noise canceling function, note the following points.

- Switching the function of the input capture pin

When the function of the input capture pin is switched by modifying port mode register 1 bit 3 (the TMIG bit), an input capture edge may be recognized even though no valid signal edge has been input. This occurs under the conditions listed in table 9-8-5.

Table 9-8-5 False Input Capture Edges Generating by Switching of Input Capture Pin Function

Input Capture Edge	Conditions
Rising edge recognized	TMIG pin level is high, and TMIG bit is changed from 0 to 1
	TMIG pin level is high and NCS bit is changed from 0 to 1, then TMIG bit is changed from 0 to 1 before noise canceller circuit completes five samples
Falling edge recognized	TMIG pin level is high, and TMIG bit is changed from 1 to 0
	TMIG pin level is low and NCS bit is changed from 0 to 1, then TMIG bit is changed from 0 to 1 before noise canceller circuit completes five samples
	TMIG pin level is high and NCS bit is changed from 0 to 1, then TMIG bit is changed from 1 to 0 before noise canceller circuit completes five samples

Note: When pin $P1_3$ is not used for input capture, the input capture signal input to timer G is low.

- Switching the input capture noise canceling function

When modifying port mode register 2 bit 4 (the NCS bit) to enable or disable the input capture noise canceling function, first clear the TMIG bit to 0. Otherwise an input capture edge may be recognized even though no valid signal edge has been input. This occurs under the conditions listed in table 9-8-6.

Table 9-8-6 False Input Capture Edges Generating by Switching of Noise Canceling Function

Input Capture Edge	Conditions
Rising edge recognized	TMIG bit is set to 1 and TMIG pin level changes from low to high, then NCS bit is changed from 1 to 0 before noise canceller circuit completes five samples
Falling edge recognized	TMIG bit is set to 1 and TMIG pin level changes from high to low, then NCS bit is changed from 1 to 0 before noise canceller circuit completes five samples

If switching of the pin function generates a false input capture edge matching the edge selected by the input capture interrupt edge select bit (IIEGS), the interrupt request flag will be set to 1, making it necessary to clear this flag to 0 before using the interrupt function. Figure 9-8-9 shows the procedure for modifying port mode register settings and clearing the interrupt request flag. The first step is to mask interrupts before modifying the port mode register. After modifying the port mode register setting, wait long enough for an input capture edge to be recognized (at least two system clocks when noise canceling is disabled; at least five sampling clocks when noise canceling is enabled), then clear the interrupt request flag to 0 (assuming it has been set to 1). An alternative procedure is to avoid having the interrupt request flag set when the pin function is switched, either by controlling the level of the input capture pin so that it does not satisfy the conditions in tables 9-8-5 and 9-8-6, or by setting the IIEGS bit of TMG to select the edge opposite to the falsely generated edge.

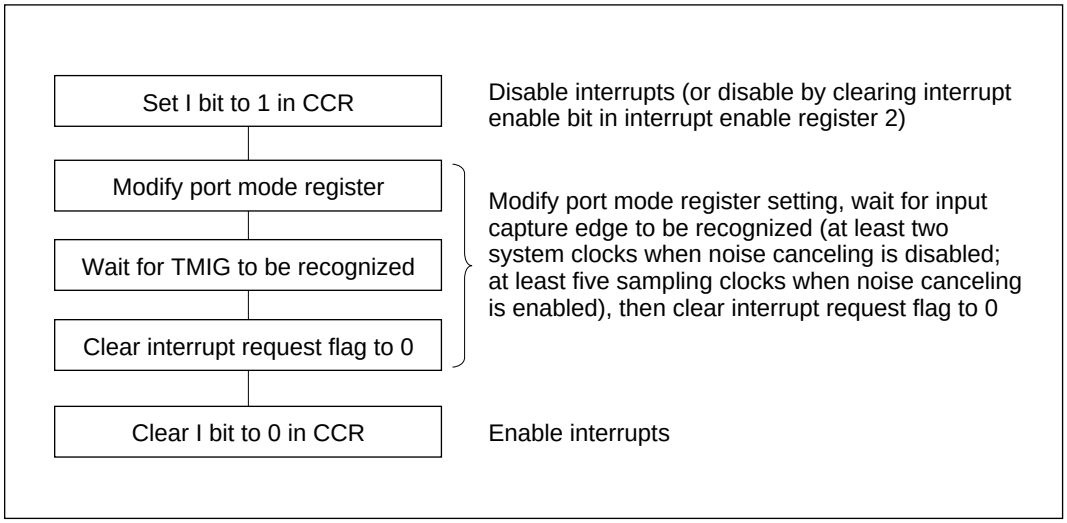


Figure 9-8-9 Procedure for Modifying Port Mode Register and Clearing Interrupt Request Flag

9.9 Timer H

9.9.1 Overview

Built-in timer H is based on an 8-bit counter. In addition to a timer counter (TCNTH), timer H has two time constant registers (TCORAH and TCORBH) that support a variety of functions. For example, a pulse signal with an arbitrary duty cycle can be output by use of the internal compare match signals generated when TCNTH matches TCORAH and TCORBH.

1. Features

- Choice of three internal clock sources (ø/1024, ø/64, ø/8) 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.
- 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

2. Block diagram

Figure 9-9-1 shows a block diagram of timer H.

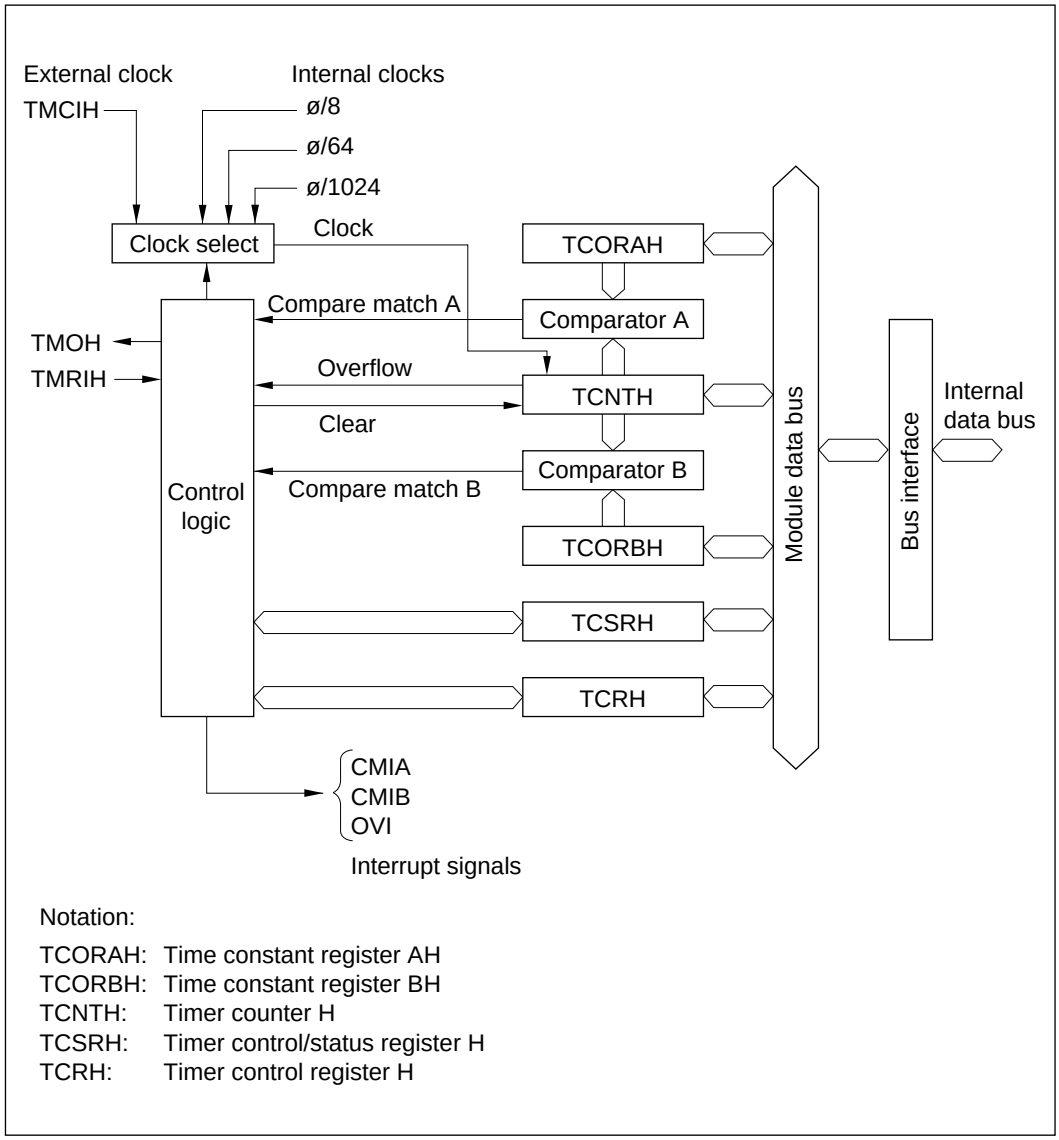


Figure 9-9-1 Block Diagram of Timer H

3. Pin configuration

Table 9-9-1 shows the timer H pin configuration.

Table 9-9-1 Pin Configuration

Name	Abbrev.	I/O	Function
Timer H output	TMOH	Output	Compare match output
Timer H clock input	TMCIH	Input	External clock input to counter
Timer H reset input	TMRIH	Input	External input to reset counter

4. Register configuration

Table 9-9-2 shows the register configuration of timer H.

Table 9-9-2 Timer H Registers

Name	Abbrev.	R/W	Initial Value	Address
Timer control register H	TCRH	R/W	H'00	H'FF88
Timer control/status register H	TCSRH	R/(W)*	H'10	H'FF89
Time constant register AH	TCORAH	R/W	H'FF	H'FF8A
Time constant register BH	TCORBH	R/W	H'FF	H'FF8B
Timer counter H	TCNTH	R/W	H'00	H'FF8C

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

9.9.2 Register Descriptions

1. Timer counter (TCNTH)

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

TCNTH 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 TCRH. The TCNTH value can be read by the CPU at any time. TCNTH 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 TCRH.

When TCNTH overflows from H'FF to H'00, the OVF flag is set to 1 in TCSRH.

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

2. Time constant registers AH and BH (TCORAH, TCORBH)

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

TCORAH and TCORBH are 8-bit read/write registers.

TCORAH and TCORBH are compared with TCNTH at all times. When they match TCNTH, CMFA or CMFB is set to 1 in TCSRH. The comparison is disabled, however, during the T_3 state of a write cycle to TCORAH or TCORBH.

Timer output can be controlled by internal signals (compare match signals) generated from compare match, according to the settings of bits OS3 to OS0 in TCSRH.

TCORAH and TCORBH are initialized to H'FF upon reset and in standby mode, watch mode, subsleep mode, and subactive mode.

3. Timer control register H (TCRH)

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

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

TCRH 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 TCSRH.

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 TCSRH.

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 TCSRH.

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 TCNTH, and select compare match A or B or an external reset signal.

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 select the internal or external clock input to TCNTH.

Three internal clock sources divided from the system clock (ø) can be selected (ø/8, ø/64, ø/1024). The counter increments on the falling edge.

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

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	0	Clock input disabled	(initial value)
		1	Internal clock: ø/8, falling edge	
	1	0	Internal clock: ø/64, falling edge	
		1	Internal clock: ø/1024, falling edge	
1	0	0	Clock input disabled	
		1	External clock: rising edge	
	1	0	External clock: falling edge	
		1	External clock: both edges	

4. Timer control/status register H (TCSRH)

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: * These bits can only be written with 0, for flag clearing.

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

TCSRH 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 TCNTH has matched TCORBH. 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 TCNTH value matches the TCORBH value	

Bit 6: Compare match flag A (CMFA)

Bit 6 is a status flag indicating that TCNTH has matched TCORAH. 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 TCNTH value matches the TCORAH value	

Bit 5: Timer overflow flag (OVF)

Bit 5 is a status flag indicating that TCNTH 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 TCNTH 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 changes in response to compare match between TCNTH and TCORAH or TCORBH.

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	1	1 output at compare match A	
	1	Output toggles at compare match A	

9.9.3 Timer Operation

1. TCNTH increment timing

TCNTH is incremented by an input (internal or external) clock.

- Internal clock

  One of three clocks (ø/1024, ø/64, ø/8) divided from the system clock (ø) can be selected by bits CKS2 to CKS0 in TCRH. Figure 9-9-2 shows the timing.

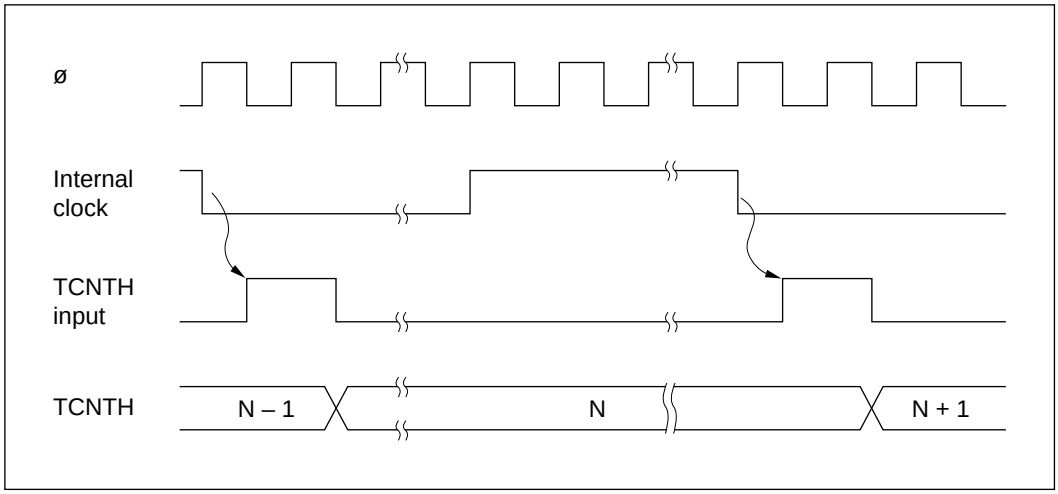


Figure 9-9-2 Increment Timing with Internal Clock

- 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 TCRH.

  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 9-9-3 shows the timing when both the rising and falling edges of the external clock are selected.

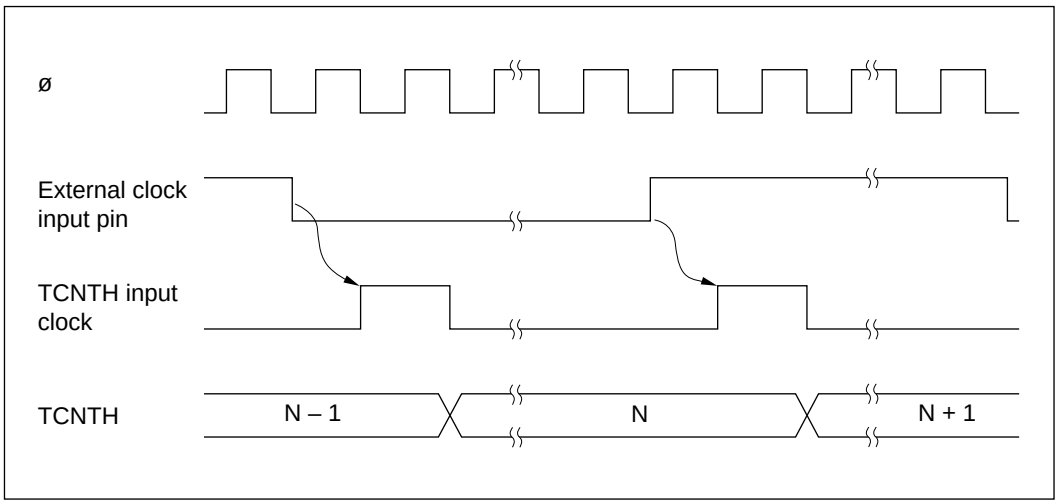


Figure 9-9-3 Increment Timing with External Clock

2. Compare match timing

- Compare match flag A or B (CMFA or CMFB) set timing

  CMFA or CMFB is set to 1 in TCSRH by an internal compare match signal generated when TCNTH matches TCORAH or TCORBH. The compare match signal is generated in the last state in which the values match (when TCNTH changes from the matching value to a new value). Accordingly, when TCNTH matches TCORAH or TCORBH, the compare match signal is not generated until the next clock input to TCNTH. Figure 9-9-4 shows the timing.

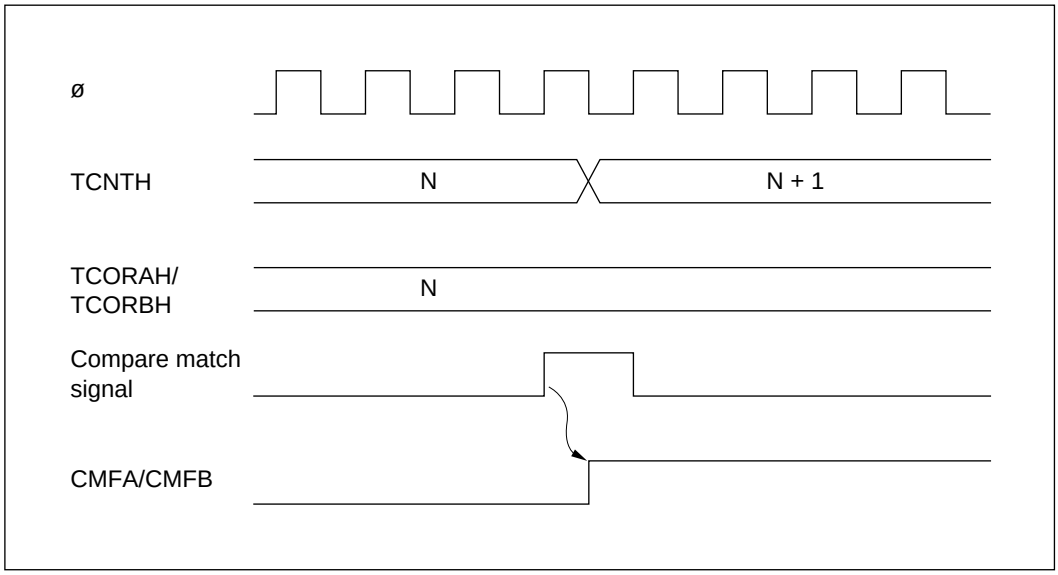


Figure 9-9-4 CMFA and CMFB Set Timing

- Timer output timing

  The timer 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 TCSRH. Figure 9-9-5 shows the timing when the output is toggled by compare match A.

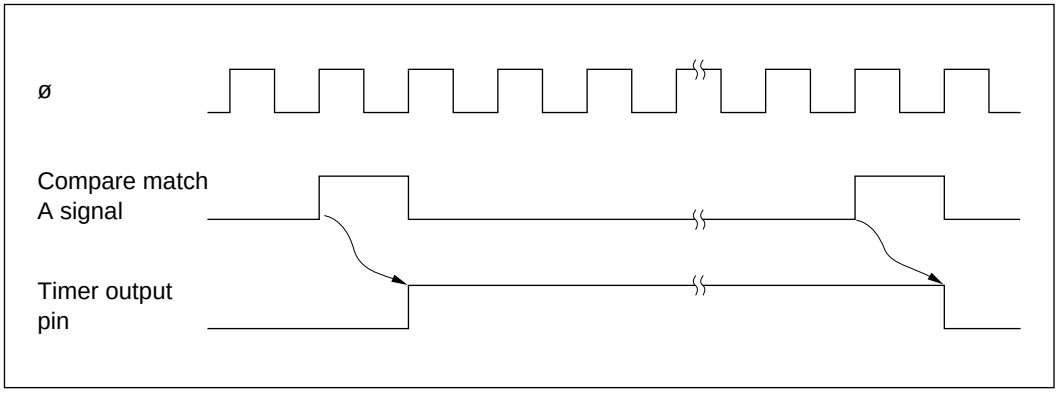


Figure 9-9-5 Timer Output Timing

- Clearing by compare match

  TCNTH can be cleared by compare match A or B, as selected by bits CCLR1 and CCLR0 in TCRH. Figure 9-9-6 shows the timing.

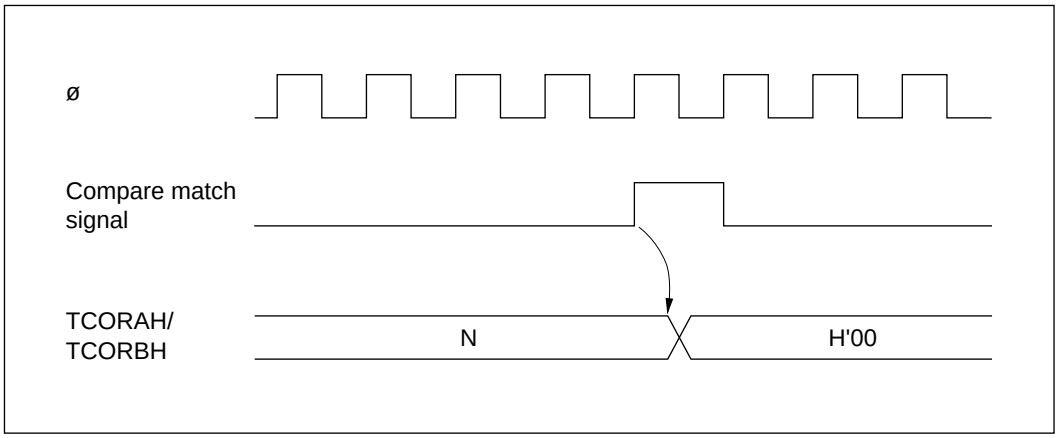


Figure 9-9-6 Clear Timing by Compare Match

3. TCNTH external reset timing

TCNTH can be cleared by the rising edge of an external reset signal, as selected by bits CCLR1 and CCLR0 in TCRH. The pulse width of the external reset signal must be at least 1.5 system clocks. Figure 9-9-7 shows the timing.

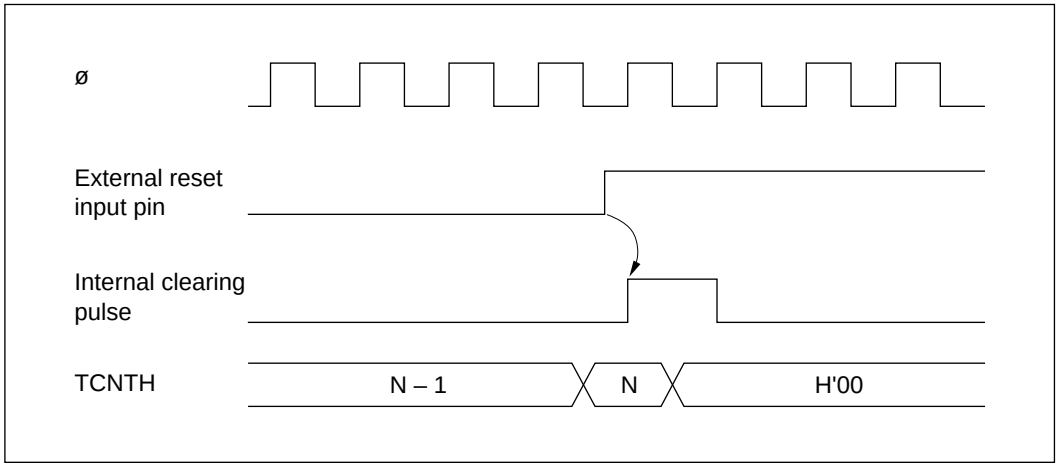


Figure 9-9-7 Clear Timing by External Reset Input

4. Overflow flag (OVF) set timing

The overflow flag (OVF) is set to 1 when TCNTH overflows from H'FF to H'00. Figure 9-9-8 shows the timing.

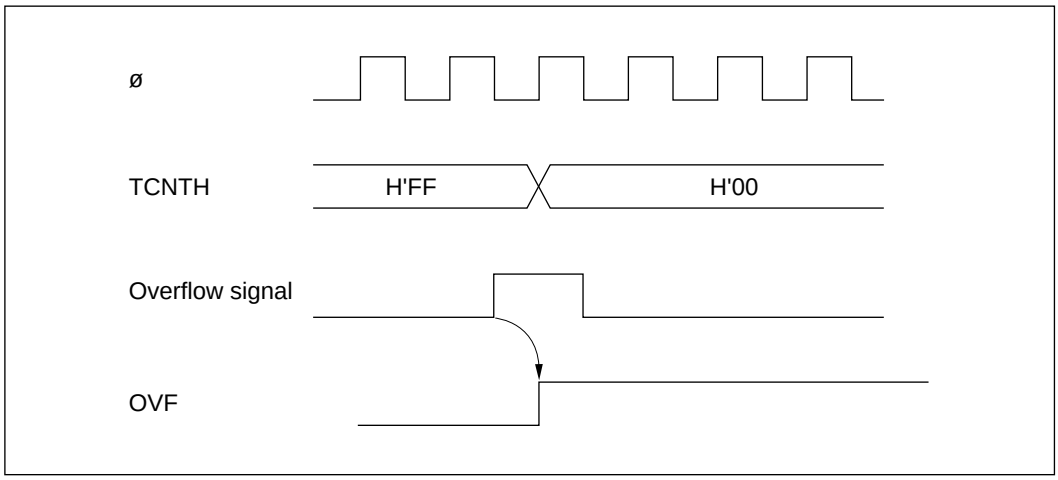


Figure 9-9-8 OVF Set Timing

9.9.4 Interrupt Sources

Timer H has three interrupt sources: CMIA, CMIB, and OVI. Table 9-9-3 lists the interrupt sources and their vector address. Each interrupt source can be enabled or disabled by an interrupt enable bit in TCRH. Although all three interrupts share the same vector, they have individual interrupt flags, so software can discriminate the interrupt source.

Table 9-9-3 Timer H Interrupt Sources

Interrupt	Description	Vector Address
CMIA	Interrupt generated from CMFA	H'002A
CMIB	Interrupt generated from CMFB	
OVI	Interrupt generated from OVF	

9.9.5 Timer H Application Examples

Figure 9-9-9 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 TCRH so that TCNTH will be cleared by compare match with TCORAH.
- Set bits OS3 to OS0 to 0110 in TCSRH so that the output will go to 1 at compare match with TCORAH and to 0 at compare match with TCORBH.

With these settings, a waveform is output without further software intervention, with a period determined by TCORAH and a pulse width determined by TCORBH.

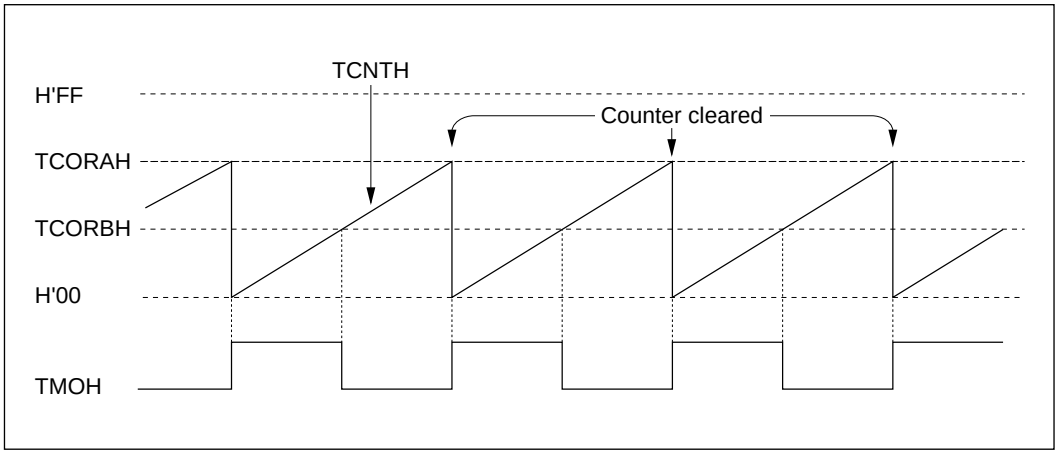


Figure 9-9-9 Pulse Output Example

9.9.6 Application Notes

The following types of contention can occur in timer H operation.

1. Contention between TCNTH write and counter clear

If a TCNTH clear signal is generated in the T_3 state of a TCNTH write cycle, clearing takes precedence and the write to the counter is not carried out. Figure 9-9-10 shows the timing.

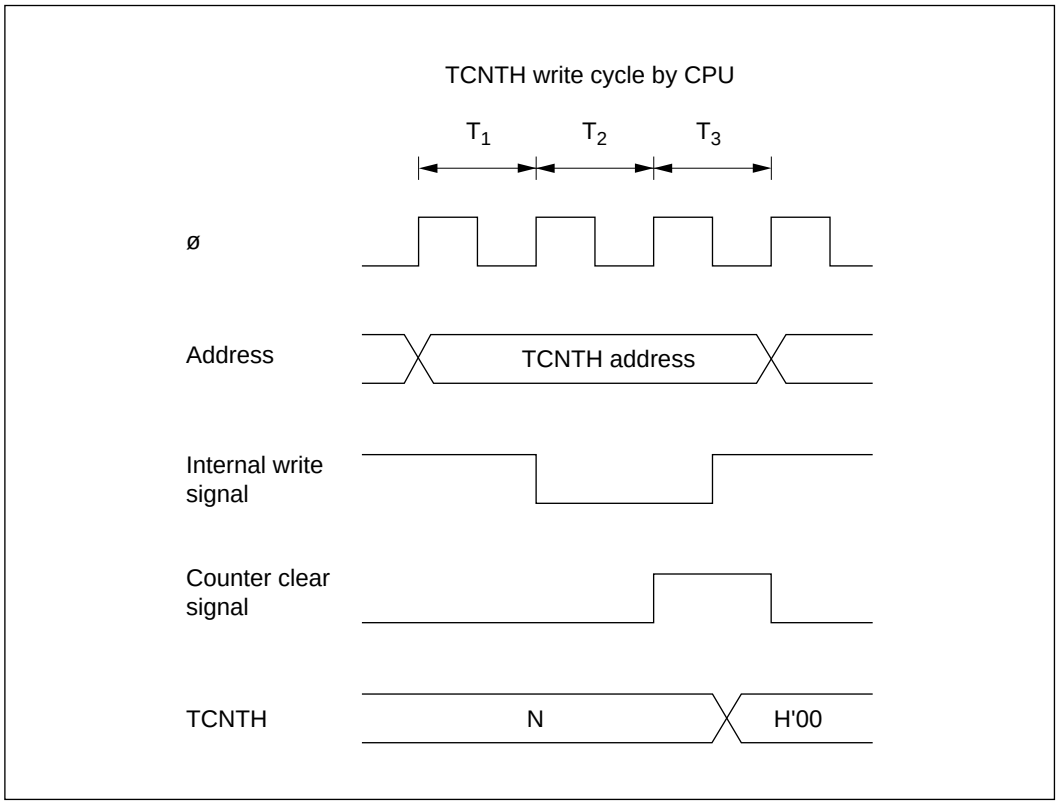


Figure 9-9-10 Contention between TCNTH Write and Clear

2. Contention between TCNTH write and increment

If a TCNTH increment clock signal is generated in the T_3 state of a TCNTH write cycle, the write takes precedence and the counter is not incremented. Figure 9-9-11 shows the timing.

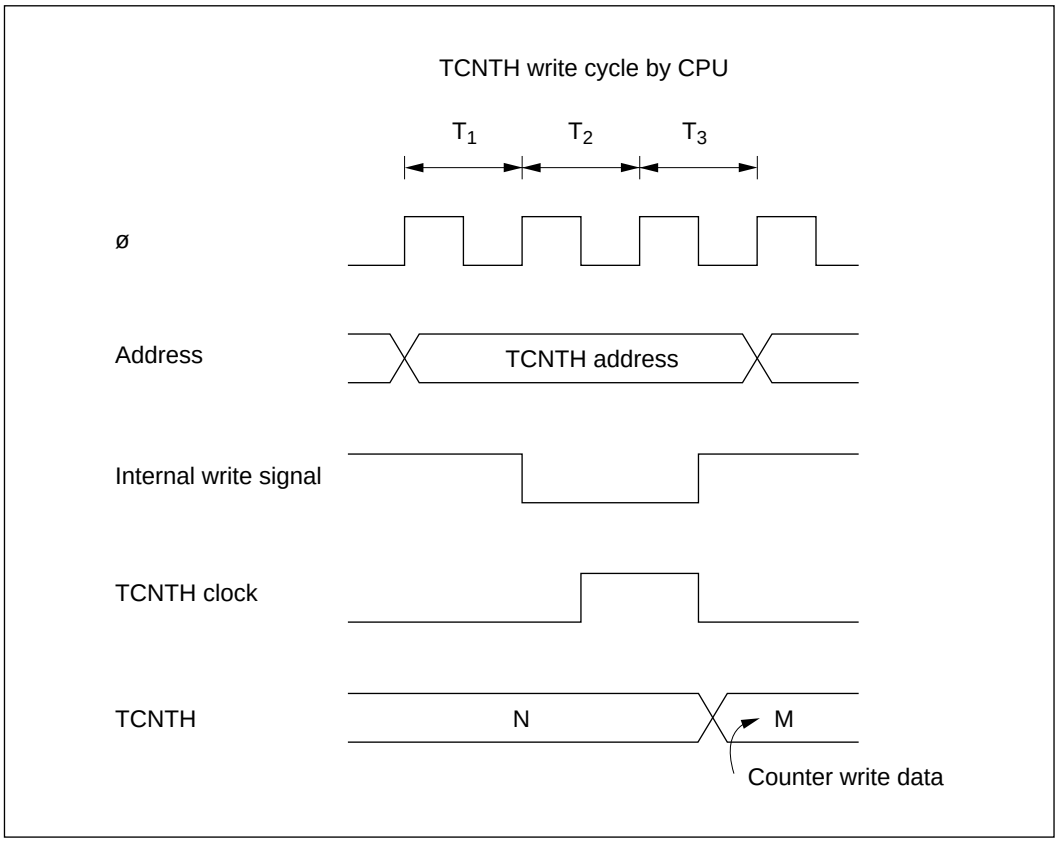


Figure 9-9-11 Contention between TCNTH Write and Increment

3. Contention between TCORAH/TCORBH write and compare match

If a compare match is generated in the T_3 state of a TCORAH or TCORBH write cycle, the write to TCORAH or TCORBH takes precedence and the compare match signal is inhibited. Figure 9-9-12 shows the timing.

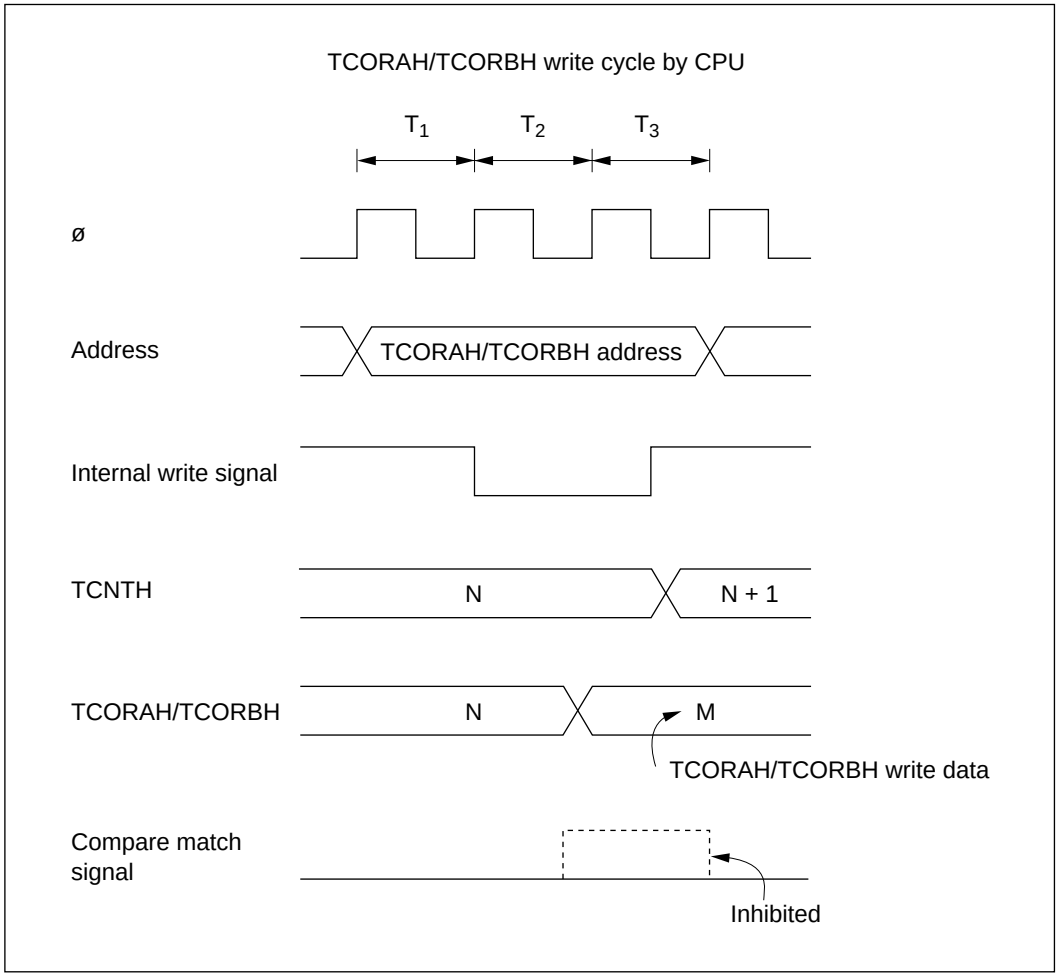


Figure 9-9-12 Contention between TCORAH/TCORBH Write and Compare Match

4. 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 9-9-4.

Table 9-9-4 Timer Output Priority Order

Output Setting	Priority
Toggle output	High
1 output	↕
0 output	
No change	Low

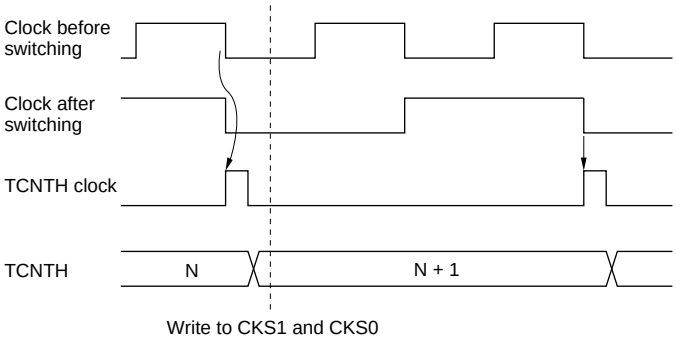
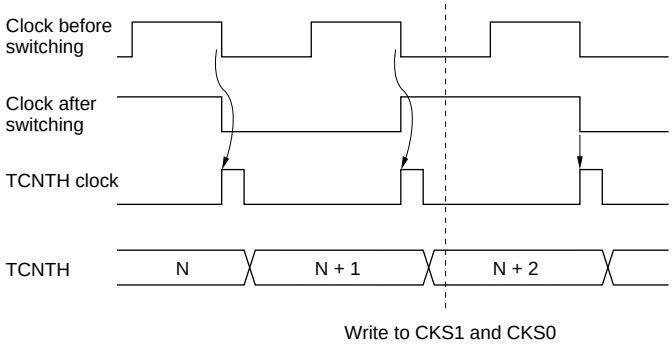
5. Internal clock switching and TCNTH operation

Depending on the timing, TCNTH may be incremented by a switch between different internal clock sources. Table 9-9-5 shows the relation between internal clock switchover timing (by writing to bits CKS1 and CKS0) and TCNTH operation.

When TCNTH 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 9-9-5 where the switch is from a high clock signal to a low clock signal, the switchover is seen as a falling edge, causing TCNTH to increment.

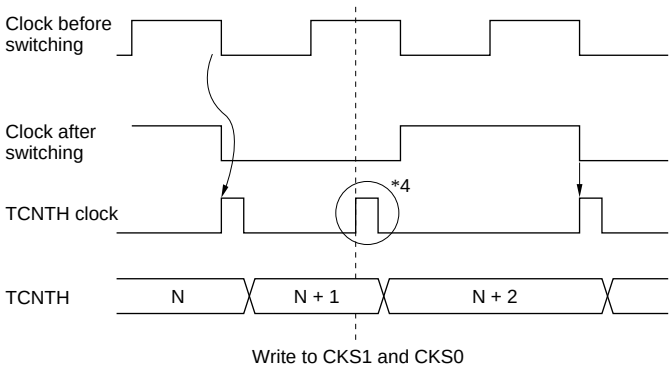
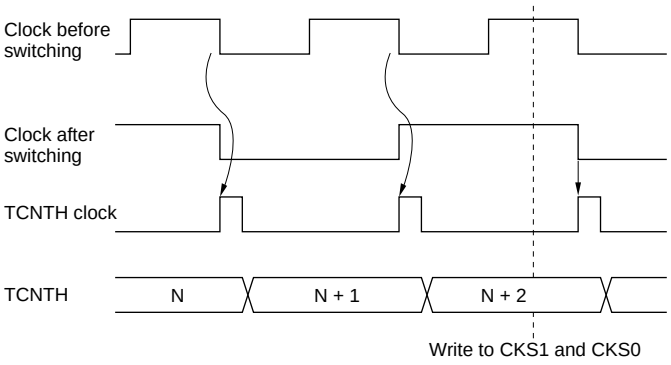
TCNTH can also be incremented by a switch between internal and external clocks.

Table 9-9-5 Internal Clock Switching and TCNTH Operation

No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	TCNTH 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 9-9-5 Internal Clock Switching and TCNTH Operation

No.	Clock Levels Before and After Modifying Bits CKS1 and CKS0	TCNTH Operation
3	Goes from high level to low level*3	
4	Goes from high to high	

Notes: 3. Including a transition from the high level to the stopped state.
4. The switchover is seen as a falling edge, and TCNTH is incremented.

Section 10 Serial Communication Interface

10.1 Overview

The H8/3947 Series is provided with a three-channel serial communication interface (SCI). Two of the three channels conform to the I²C bus interface (inter IC bus interface). Table 10-1-1 summarizes the functions and features of the three SCI channels.

Table 10-1-1 Serial Communication Interface Functions

Channel	Functions	Features
SCI3	Synchronous serial transfer • 8-bit data transfer • Send, receive, or simultaneous send/receive Asynchronous serial transfer • Multiprocessor communication function • Choice of 7-bit or 8-bit data length • Choice of 1-bit or 2-bit stop bit length • Odd or even parity	• Built-in baud rate generator • Receive error detection • Break detection • Interrupt requested at completion of transfer or error
I²C1 I²C2	I²C bus interface functions • Selectable data bit length (1 to 8 bits) • Master/slave and transmit/receive selection • Selection of serial clocks (ø/64, ø/56, ø/50, ø/40, ø/32, ø/24, ø/20, ø/14) • Selection of slave address and data format • Optional wait after data transfer • Three interrupt sources (data transfer end, slave address detect, bus arbitration lost)	• Conforms to Philips I²C bus interface specifications • Data transfer controlled by bus arbitration check • Data transfer can be controlled by acknowledge bit

10.2 SCI3

10.2.1 Overview

Serial communication interface 3 (SCI3) has both synchronous and asynchronous serial data communication capabilities. It also has a multiprocessor communication function for serial data communication among two or more processors.

1. Features

SCI3 features are listed below.

- Selection of asynchronous or synchronous mode

a. Asynchronous mode

SCI3 can communicate with a UART (universal asynchronous receiver/transmitter), ACIA (asynchronous communication interface adapter), or other chip that employs standard asynchronous serial communication. It can also communicate with two or more other processors using the multiprocessor communication function. There are twelve selectable serial data communication formats.

Data length	7 or 8 bits
Stop bit length	1 or 2 bits
Parity	Even, odd, or none
Multiprocessor bit	1 or none
Receive error detection	Parity, overrun, and framing errors
Break detection	By reading the RXD level directly when a framing error occurs

b. Synchronous mode

Serial data communication is synchronized with a clock signal. SCI3 can communicate with other chips having a clocked synchronous communication function.

Data length	8 bits
Receive error detection	Overrun errors

- Full duplex communication

The transmitting and receiving sections are independent, so SCI3 can transmit and receive simultaneously. Both sections use double buffering, so continuous data transfer is possible in both the transmit and receive directions.

- Built-in baud rate generator with selectable bit rates.
- Internal or external clock may be selected as the transfer clock source.
- There are six interrupt sources: transmit end, transmit data empty, receive data full, overrun error, framing error, and parity error.

2. Block diagram

Figure 10-2-1 shows a block diagram of SCI3.

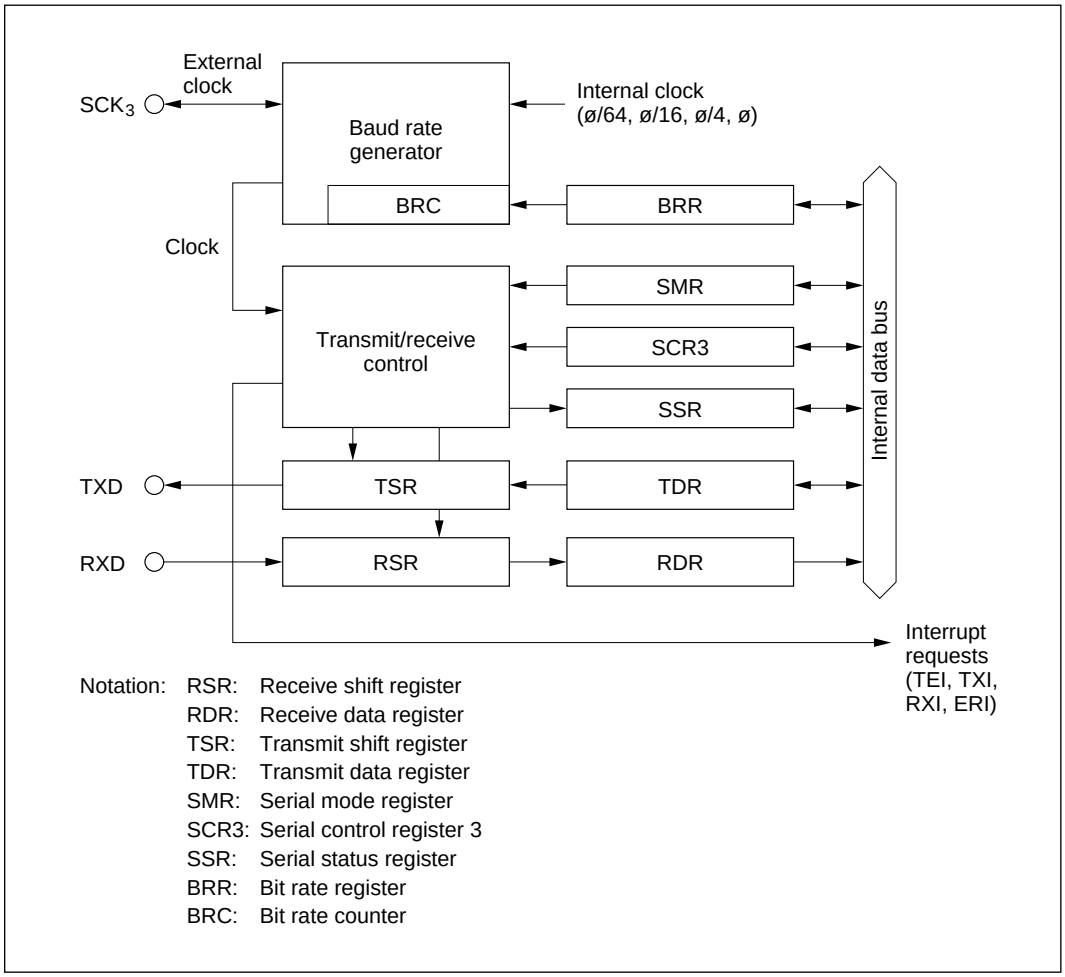


Figure 10-2-1 SCI3 Block Diagram

3. Pin configuration

Table 10-2-1 shows the SCI3 pin configuration.

Table 10-2-1 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

4. Register configuration

Table 10-2-2 shows the SCI3 internal register configuration.

Table 10-2-2 SCI3 Registers

Name	Abbrev.	R/W	Initial Value	Address
Serial mode register	SMR	R/W	H'00	H'FFA8
Bit rate register	BRR	R/W	H'FF	H'FFA9
Serial control register 3	SCR3	R/W	H'00	H'FFAA
Transmit data register	TDR	R/W	H'FF	H'FFAB
Serial status register	SSR	R/(W)*	H'84	H'FFAC
Receive data register	RDR	R	H'00	H'FFAD
Transmit shift register	TSR	—	—	—
Receive shift register	RSR	—	—	—
Bit rate counter	BRC	—	—	—

Note: * Only 0 can be written in bits 7 to 3, to clear flags. Bits 2 and 1 are read-only bits.

10.2.2 Register Descriptions

1. Receive shift register (RSR)

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

The receive shift register (RSR) is for receiving serial data.

Serial data is input in LSB-first order (bit 0 first) into RSR from pin RXD, converting it to parallel data. After each byte of data has been received, the byte is automatically transferred to the receive data register (RDR).

RSR cannot be read or written directly by the CPU.

2. Receive data register (RDR)

Bit	7	6	5	4	3	2	1	0
	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

The receive data register (RDR) is an 8-bit register for storing received serial data.

Each time a byte of data is received, the received data is transferred from the receive shift register (RSR) to RDR, completing a receive operation. Thereafter RSR again becomes ready to receive new data. RSR and RDR form a double buffer mechanism that allows data to be received continuously.

RDR is exclusively for receiving data and cannot be written by the CPU.

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

3. Transmit shift register (TSR)

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

The transmit shift register (TSR) is for transmitting serial data.

Transmit data is first transferred from the transmit data register (TDR) to TSR, then is transmitted from pin TXD, starting from the LSB (bit 0). After one byte of data has been sent, the next byte is automatically transferred from TDR to TSR, and the next transmission begins. If no data has been written to TDR (1 is set in TDRE), there is no data transfer from TDR to TSR.

TSR cannot be read or written directly by the CPU.

4. Transmit data register (TDR)

Bit	7	6	5	4	3	2	1	0
	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The transmit data register (TDR) is an 8-bit register for holding transmit data. When SCI3 detects that the transmit shift register (TSR) is empty, it shifts transmit data written in TDR to TSR and starts serial data transmission. While TSR is transmitting serial data, the next byte to be transmitted can be written to TDR, realizing continuous transmission.

TDR can be read or written by the CPU at all times.

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

5. Serial mode register (SMR)

Bit	7	6	5	4	3	2	1	0
	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The serial mode register (SMR) is an 8-bit register for setting the serial data communication format and for selecting the clock source of the baud rate generator. SMR can be read and written by the CPU at any time.

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

Bit 7: Communication mode (COM)

Bit 7 selects asynchronous mode or synchronous mode for SCI3.

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

Bit 6: Character length (CHR)

Bit 6 selects either 7 bits or 8 bits as the data length in asynchronous mode. In synchronous mode the data length is always 8 bits regardless of the CHR 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) in the transmit data register is not transmitted.

Bit 5: Parity enable (PE)

In asynchronous mode, bit 5 selects whether or not a parity bit is to be added to transmitted data and checked in received data. In synchronous mode there is no adding or checking of parity regardless of the PE setting.

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

Note: * When PE is set to 1, then either odd or even parity is added to transmit data, depending on the setting of the parity mode bit (PM). When data is received, it is checked for odd or even parity as designated in bit PM.

Bit 4: Parity mode (PM)

In asynchronous mode, bit 4 selects whether odd or even parity is to be added to transmitted data and checked in received data. The setting here is valid only if parity adding/checking is enabled in bit PE. In synchronous mode, or if parity adding/checking is disabled in bit PE, bit PM is ignored.

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

Notes: 1. When even parity is designated, a parity bit is added to the transmitted data so that the sum of 1s in the resulting data is an even number. When data is received, the sum of 1s in the data plus parity bit is checked to see if the result is an even number.
2. When odd parity is designated, a parity bit is added to the transmitted data so that the sum of 1s in the resulting data is an odd number. When data is received, the sum of 1s in the data plus parity bit is checked to see if the result is an odd number.

Bit 3: Stop bit length (STOP)

Bit 3 selects 1 bit or 2 bits as the stop bit length in asynchronous mode. This setting is valid only in asynchronous mode. In synchronous mode a stop bit is not added, so this bit is ignored.

When data is received, only the first stop bit is checked regardless of the stop bit length. If the second stop bit value is 1 it is treated as a stop bit; if it is 0, it is treated as the start bit of the next character.

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

Notes: 1. When data is transmitted, one “1” bit is added at the end of each transmitted character as the stop bit.
2. When data is transmitted, two “1” bits are added at the end of each transmitted character as the stop bits.

Bit 2: Multiprocessor mode (MP)

Bit 2 enables or disables the multiprocessor communication function. When the multiprocessor communication function is enabled, the parity enable (PE) and parity mode (PM) settings are ignored. The MP bit is valid only in asynchronous mode; it should be cleared to 0 in synchronous mode.

See 10.2.6, for details on the multiprocessor communication function.

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

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

Bits 1 and 0 select the clock source for the built-in baud rate generator. A choice of ø/64, ø/16, ø/4, or ø is made in these bits.

See 8, Bit rate register, below for information on the clock source and bit rate register settings, and their relation to the baud rate.

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	ø clock	(initial value)
0	1	ø/4 clock	
1	0	ø/16 clock	
1	1	ø/64 clock	

6. Serial control register 3 (SCR3)

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Serial control register 3 (SCR3) is an 8-bit register that controls SCI3 transmit and receive operations, enables or disables serial clock output in asynchronous mode, enables or disables interrupts, and selects the serial clock source. SCR3 can be read and written by the CPU at any time.

SCR3 is initialized to H'00 upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Bit 7: Transmit interrupt enable (TIE)

Bit 7 enables or disables the transmit data empty interrupt (TXI) request when data is transferred from TDR to TSR and the transmit data register empty bit (TDRE) in the serial status register (SSR) is set to 1. The TXI interrupt can be cleared by clearing bit TDRE to 0, or by clearing bit TIE to 0.

Bit 7 TIE	Description	
0	Transmit data empty interrupt request (TXI) disabled	(initial value)
1	Transmit data empty interrupt request (TXI) enabled	

Bit 6: Receive interrupt enable (RIE)

Bit 6 enables or disables the receive error interrupt (ERI), and the receive data full interrupt (RXI) requested when data is transferred from RSR to RDR and the receive data register full bit (RDRF) in the serial status register (SSR) is set to 1. RXI and ERI interrupts can be cleared by clearing SSR flag RDRF, or flags FER, PER, and OER to 0, or by clearing bit RIE to 0.

Bit 6 RIE	Description	
0	Receive data full interrupt request (RXI) and receive error interrupt request (ERI) disabled	(initial value)
1	Receive data full interrupt request (RXI) and receive error interrupt request (ERI) enabled	

Bit 5: Transmit enable (TE)

Bit 5 enables or disables the start of a transmit operation.

Bit 5 TE	Description	
0	Transmit operation disabled*1 (TXD is a general I/O port)	(initial value)
1	Transmit operation enabled*2 (TXD is the transmit data pin)	

Notes: 1. The transmit data register empty bit (TDRE) in the serial status register (SSR) is fixed at 1.
2. In this state, if transmit data is written in TDR, bit TDRE is cleared to 0 in SSR and serial transmission begins. Be sure to select the transmit format in SMR before setting TE to 1.

Bit 4: Receive enable (RE)

Bit 4 enables or disables the start of a receive operation.

Bit 4 RE	Description	
0	Receive operation disabled*1 (RXD is a general I/O port)	(initial value)
1	Receive operation enabled*2 (RXD is the receive data pin)	

Notes: 1. When RE is cleared to 0, this has no effect on the SSR flags RDRF, FER, PER, and OER, which retain their states.
2. Serial data receiving begins when, in this state, a start bit is detected in asynchronous mode, or serial clock input is detected in synchronous mode.
Before setting RE to 1 it is necessary to set the receive format in SMR.

Bit 3: Multiprocessor interrupt enable (MPIE)

Bit 3 enables or disables multiprocessor interrupt requests. This setting is valid only in asynchronous mode, and only when the multiprocessor mode bit (MP) in the serial mode register (SMR) is set to 1. It applies only to data receiving. This bit is ignored when COM is set to 1 or when bit MP is cleared to 0.

Bit 3 MPIE	Description	
0	Multiprocessor interrupt request disabled (ordinary receive operation)	(initial value)
	Clearing condition: Multiprocessor bit receives a data value of 1	
1	Multiprocessor interrupt request enabled*	

Note: * SCI3 does not transfer receive data from RSR to RDR, does not detect receive errors, and does not set status flags RDRF, FER, and OER in SSR. Until a multiprocessor bit value of 1 is received, the receive data full interrupt (RXI) and receive error interrupt (ERI) are disabled and serial status register (SSR) flags RDRF, FER, and OER are not set. When the multiprocessor bit receives a “1”, the MPBR bit of SSR is set to 1, MPIE is automatically cleared to 0, RXI and ERI interrupts are enabled (provided bits TIE and RIE in SCR3 are set to 1), and setting of the RDRF, FER, and OER flags is enabled.

Bit 2: Transmit end interrupt enable (TEIE)

Bit 2 enables or disables the transmit end interrupt (TEI) requested if there is no valid transmit data in TDR when the MSB is transmitted.

Bit 2 TEIE	Description	
0	Transmit end interrupt (TEI) disabled	(initial value)
1	Transmit end interrupt (TEI) enabled*	

Note: * A TEI interrupt can be cleared by clearing the SSR bit TDRE to 0, thereby clearing the transmit end bit (TEND) to 0, or by clearing bit TEIE to 0.

Bits 1 and 0: Clock enable 1, 0 (CKE1, CKE0)

Bits 1 and 0 select the clock source and enable or disable clock output at pin SCK_3. The combination of bits CKE1 and CKE0 determines whether pin SCK_3 is a general I/O port, a clock output pin, or a clock input pin.

The CKE0 setting is valid only in asynchronous mode and only when using an internal clock (CKE1 = 0). CKE0 should be cleared to 0 in synchronous mode, or when using an external clock (CKE1 = 1).

CKE1 and CKE0 settings should be made before selecting the operating mode in SMR. For details on clock source selection, see table 10-2-9 in 10.2.3, Operation.

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
0	1	Asynchronous	Internal clock	Clock output*2
		Synchronous	Reserved	Reserved
1	0	Asynchronous	External clock	Clock input*3
		Synchronous	External clock	Serial clock input
1	1	Asynchronous	Reserved	Reserved
		Synchronous	Reserved	Reserved

Notes: 1. Initial value
2. A clock is output with the same frequency as the bit rate.
3. Input a clock with a frequency 16 times the bit rate.

7. Serial status register (SSR)

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	OER	FER	PER	TEND	MPBR	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: *Only 0 can be written for flag clearing.

The serial status register (SSR) is an 8-bit register containing status flags for indicating SCI3 states, and containing the multiprocessor bits.

SSR can be read and written by the CPU at any time, but the CPU cannot write a 1 to the status flags TDRE, RDRF, OER, PER, and FER. To clear these flags to 0 it is first necessary to read a 1. Bit 2 (TEND) and bit 1 (MPBR) are read-only bits and cannot be modified.

SSR is initialized to H'84 upon reset or in standby mode, watch mode, subactive mode, or subsleep mode.

Bit 7: Transmit data register empty (TDRE)

Bit 7 is a status flag indicating that data has been transferred from TDR to TSR.

Bit 7 TDRE	**Description**	
0	Indicates that transmit data written to TDR has not been transferred to TSR	
	Clearing conditions: After reading TDRE = 1, cleared by writing 0 to TDRE. When data is written to TDR by an instruction.	
1	Indicates that no transmit data has been written to TDR, or the transmit data written to TDR has been transferred to TSR	(initial value)
	Setting conditions: When bit TE in SCR3 is cleared to 0. When data is transferred from TDR to TSR.	

Bit 6: Receive data register full (RDRF)

Bit 6 is a status flag indicating whether there is receive data in RDR.

Bit 6 RDRF	Description	
0	Indicates there is no receive data in RDR	(initial value)
	Clearing conditions: After reading RDRF = 1, cleared by writing 0 to RDRF. When data is read from RDR by an instruction.	
1	Indicates that there is receive data in RDR	
	Setting condition: When receiving ends normally, with receive data transferred from RSR to RDR	

Note: If a receive error is detected at the end of receiving, or if bit RE in serial control register 3 (SCR3) is cleared to 0, RDR and RDRF are unaffected and keep their previous states. An overrun error (OER) occurs if receiving of data is completed while bit RDRF remains set to 1. If this happens, receive data will be lost.

Bit 5: Overrun error (OER)

Bit 5 is a status flag indicating that an overrun error has occurred during data receiving.

Bit 5 OER	Description	
0	Indicates that data receiving is in progress or has been completed*1	(initial value)
	Clearing condition: After reading OER = 1, cleared by writing 0 to OER	
1	Indicates that an overrun error occurred in data receiving*2	
	Setting condition: When data receiving is completed while RDRF is set to 1	

Notes: 1. When bit RE in serial control register 3 (SCR3) is cleared to 0, OER is unaffected and keeps its previous state.
2. RDR keeps the data received prior to the overrun; data received after that is lost. While OER is set to 1, data receiving cannot be continued. In synchronous mode, data transmitting cannot be continued either.

Bit 4: Framing error (FER)

Bit 4 is a status flag indicating that a framing error has occurred during asynchronous receiving.

Bit 4 FER	Description	
0	Indicates that data receiving is in progress or has been completed*1	(initial value)
	Clearing condition: After reading FER = 1, cleared by writing 0 to FER	
1	Indicates that a framing error occurred in data receiving	
	Setting condition: The stop bit at the end of receive data is checked and found to be 0*2	

Notes: 1. When bit RE in serial control register 3 (SCR3) is cleared to 0, FER is unaffected and keeps its previous state.
2. When two stop bits are used only the first stop bit is checked, not the second. When a framing error occurs, receive data is transferred to RDR but RDRF is not set. While FER is set to 1, data receiving cannot be continued. In synchronous mode, data transmitting cannot be continued either.

Bit 3: Parity error (PER)

Bit 3 is a status flag indicating that a parity error has occurred during asynchronous receiving.

Bit 3 PER	Description	
0	Indicates that data receiving is in progress or has been completed*1	(initial value)
	Clearing condition: After reading PER = 1, cleared by writing 0 to PER	
1	Indicates that a parity error occurred in data receiving*2	
	Setting condition: When the sum of 1s in received data plus the parity bit does not match the parity mode bit (PM) setting in the serial mode register (SMR)	

Notes: 1. When bit RE in serial control register 3 (SCR3) is cleared to 0, PER is unaffected and keeps its previous state.
2. When a parity error occurs, receive data is transferred to RDR but RDRF is not set. While PER is set to 1, data receiving cannot be continued. In synchronous mode, data transmitting cannot be continued either.

Bit 2: Transmit end (TEND)

Bit 2 is a status flag indicating that TDRE was set to 1 when the last bit of a transmitted character was sent. TEND is a read-only bit and cannot be modified directly.

Bit 2 TEND	Description	
0	Indicates that transmission is in progress	
	Clearing conditions: After reading TDRE = 1, cleared by writing 0 to TDRE. When data is written to TDR by an instruction.	
1	Indicates that a transmission has ended	(initial value)
	Setting conditions: When bit TE in SCR3 is cleared to 0. If TDRE is set to 1 when the last bit of a transmitted character is sent.	

Bit 1: Multiprocessor bit receive (MPBR)

Bit 1 holds the multiprocessor bit in data received in asynchronous mode using a multiprocessor format. MPBR is a read-only bit and cannot be modified.

Bit 1 MPBR	Description	
0	Indicates reception of data in which the multiprocessor bit is 0*	(initial value)
1	Indicates reception of data in which the multiprocessor bit is 1	

Note: * If bit RE is cleared to 0 while a multiprocessor format is in use, MPBR retains its previous state.

Bit 0: Multiprocessor bit transmit (MPBT)

Bit 0 holds the multiprocessor bit to be added to transmitted data when a multiprocessor format is used in asynchronous mode. Bit MPBT is ignored when synchronous mode is chosen, when the multiprocessor communication function is disabled, or when data transmission is disabled.

Bit 0 MPBT	Description	
0	The multiprocessor bit in transmit data is 0	(initial value)
1	The multiprocessor bit in transmit data is 1	

8. Bit rate register (BRR)

Bit	7	6	5	4	3	2	1	0
	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The bit rate register (BRR) is an 8-bit register which, together with the baud rate generator clock selected by bits CKS1 and CKS0 in the serial mode register (SMR), sets the transmit/receive bit rate.

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

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

Table 10-2-3 gives examples of how BRR is set in asynchronous mode. The values in table 10-2-3 are for active (high-speed) mode.

Table 10-2-3 BRR Settings and Bit Rates in Asynchronous Mode (1)

	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 10-2-3 BRR Settings and Bit Rates in Asynchronous Mode (2)

	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 10-2-3 BRR Settings and Bit Rates in Asynchronous Mode (3)

	OSC (MHz)					
	9.8304			10		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)
110	2	86	+0.31	2	88	–0.25
150	1	255	0	2	64	+0.16
300	1	127	0	1	129	+0.16
600	0	255	0	1	64	+0.16
1200	0	127	0	0	129	+0.16
2400	0	63	0	0	64	+0.16
4800	0	31	0	0	32	–1.36
9600	0	15	0	0	15	+1.73
19200	0	7	0	0	7	+1.73
31250	0	4	–1.70	0	4	0
38400	0	3	0	0	3	+1.73

Notes: 1. Settings should be made so that error is within 1%.

2. BRR setting values are derived by the following equation.

$$N = \frac{OSC}{64 \times 2^{2n} \times B} \times 10^6 - 1$$

B: Bit rate (bits/s)
N: BRR baud rate generator setting ($0 \leq N \leq 255$)
OSC: Value of $ø_{OSC}$ (MHz)
n: Baud rate generator input clock number (n = 0, 1, 2, 3)

3. The error values in table 10-2-3 were derived by performing the following calculation and rounding off to two decimal places.

$$\text{Error (\%)} = \frac{B - R}{R} \times 100$$

B: Bit rate found from n, N, and OSC
R: Bit rate listed in left column of table 10-2-3

The meaning of n is shown in table 10-2-4.

Table 10-2-4 Relation between n and Clock

		SMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	ø	0	0
1	ø/4	0	1
2	ø/16	1	0
3	ø/64	1	1

Table 10-2-5 shows the maximum bit rate for selected frequencies in asynchronous mode. Values in table 10-2-5 are for active (high-speed) mode.

Table 10-2-5 Maximum Bit Rate at Selected Frequencies (Asynchronous Mode)

		Setting	
OSC (MHz)	**Maximum Bit Rate (bits/s)**	**n**	**N**
2	31250	0	0
2.4576	38400	0	0
4	62500	0	0
4.194304	65536	0	0
4.9152	76800	0	0
6	93750	0	0
7.3728	115200	0	0
8	125000	0	0
9.8304	153600	0	0
10	156250	0	0

Table 10-2-6 shows typical BRR settings in synchronous mode. Values in table 10-2-6 are for active (high-speed) mode.

Table 10-2-6 Typical BRR Settings and Bit Rates (Synchronous Mode)

Bit Rate (bits/s)	OSC (MHz)							
	2		4		8		10	
	n	N	n	N	n	N	n	N
110	—	—	—	—	—	—	—	—
250	1	249	2	124	2	249	—	—
500	1	124	1	249	2	124	—	—
1K	0	249	1	124	1	249	—	—
2.5K	0	99	0	199	1	99	1	124
5K	0	49	0	99	0	199	0	249
10K	0	24	0	49	0	99	0	124
25K	0	9	0	19	0	39	0	49
50K	0	4	0	9	0	19	0	24
100K	—	—	0	4	0	9	—	—
250K	0	0*	0	1	0	3	0	4
500K			0	0*	0	1	—	—
1M					0	0*	—	—
2.5M								

Notes: Blank: Cannot be set
—: Can be set, but error will result
*: Continuous transfer not possible at this setting
BRR setting values are derived by the following equation.

$$N = \frac{OSC}{8 \times 2^{2n} \times B} \times 10^6 - 1$$

B: Bit rate (bits/s)
N: BRR baud rate generator setting ($0 \leq N \leq 255$)
OSC: Value of ϕ_{OSC} (MHz)
n: Baud rate generator input clock number (n = 0, 1, 2, 3)

The meaning of n is shown in table 10-2-7.

Table 10-2-7 Relation between n and Clock

		SMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	ø	0	0
1	ø/4	0	1
2	ø/16	1	0
3	ø/64	1	1

10.2.3 Operation

SCI3 supports serial data communication in both asynchronous mode, where each character transferred is synchronized separately, and synchronous mode, where transfer is synchronized by clock pulses.

The choice of asynchronous mode or synchronous mode, and the communication format, is made in the serial mode register (SMR), as shown in table 10-2-8. The SCI3 clock source is determined by bit COM in SMR and bits CKE1 and CKE0 in serial control register 3 (SCR3), as shown in table 10-2-9.

1. Asynchronous mode

   — Data length: choice of 7 bits or 8 bits

   — Transmit/receive format options include addition of parity bit, multiprocessor bit, and one or two stop bits (character length depends on this combination of options).

   — Framing error (FER), parity error (PER), overrun error (OER), and line breaks can be detected when data is received.

   — Clock source: Choice of internal clocks or an external clock

   When an internal clock is selected: Operates on baud rate generator clock. A clock can be output with the same frequency as the bit rate.

   When an external clock is selected: A clock input with a frequency 16 times the bit rate is required (internal baud rate generator is not used).

2. Synchronous mode

— Transfer format: 8 bits

— Overrun error can be detected when data is received.

— Clock source: Choice of internal clocks or an external clock

When an internal clock is selected: Operates on baud rate generator clock, and outputs a serial clock.

When an external clock is selected: The internal baud rate generator is not used. Operation is synchronous with the input clock.

Table 10-2-8 SMR Settings and Communication Format

SMR Setting						Communication Format			
Bit7 COM	**Bit6 CHR**	**Bit2 MP**	**Bit5 PE**	**Bit3 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		None

Note: * Don’t care

Table 10-2-9 SMR and SCR3 Settings and Clock Source Selection

SMR	SCR3			Transmit/Receive Clock	
Bit7 COM	Bit1 CKE1	Bit0 CKE0	Mode	Clock Source	Pin SCK_3 Function
0	0	0	Asynchronous mode	Internal	I/O port (SCK_3 function not used)
		1			Outputs clock with same frequency as bit rate
	1	0		External	Clock should be input with frequency 16 times the desired bit rate
1	0	0	Synchronous mode	Internal	Outputs a serial clock
	1	0		External	Inputs a serial clock
0	1	1	Reserved	(illegal settings)	
1	0	1			
1	1	1			

3. Continuous transmit/receive operation using interrupts

Continuous transmit and receive operations are possible with SCI3, using the RXI or TXI interrupts. Table 10-2-10 explains this use of these interrupts.

Table 10-2-10 Transmit/Receive Interrupts

Interrupt	Flag	Interrupt Conditions	Remarks
RXI	RDRF RIE	When serial data is received normally and receive data is transferred from RSR to RDR, RDRF is set to 1. If RIE is 1 at this time, RXI is enabled and an interrupt occurs. (See figure 10-2-2 (a).)	The RXI interrupt handler routine should read the receive data from RDR and clear RDRF to 0. Continuous receiving is possible if these operations are completed before the next data has been completely received in RSR.
TXI	TDRE TIE	When TSR empty (previous transmission complete) is detected and the transmit data set in TDR is transferred to TSR, TDRE is set to 1. If TIE is 1 at this time, TXI is enabled and an interrupt occurs. (See figure 10-2-2 (b).)	The TXI interrupt handler routine should write the next transmit data to TDR, clearing TDRE to 0. Continuous transmission is possible if these operations are completed before the data transferred to TSR has been completely transmitted.
TEI	TEND TEIE	When the last bit of the TSR transmit character has been sent, if TDRE is 1, then 1 is set in TEND. If TEIE is 1 at this time, TEI is enabled and an interrupt occurs. (See figure 10-2-2 (c).)	TEI indicates that, when the last bit of the TSR transmit character was sent, the next transmit data had not been written to TDR.

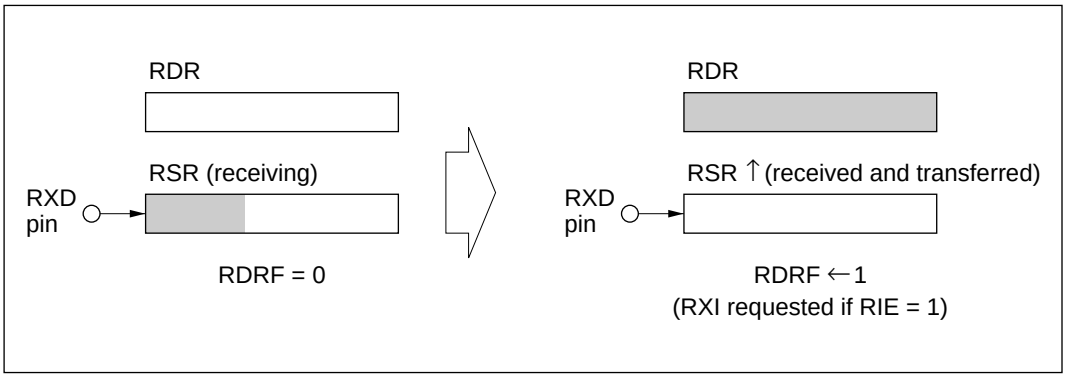


Figure 10-2-2 (a) RDRF Setting and RXI Interrupt

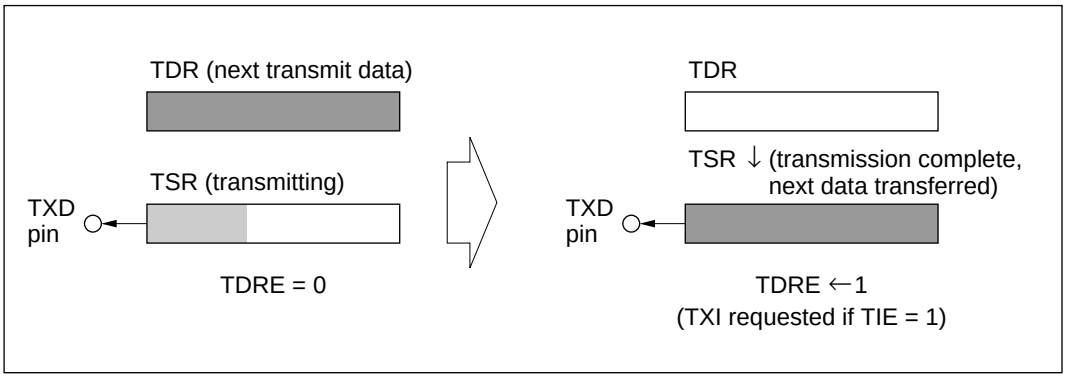


Figure 10-2-2 (b) TDRE Setting and TXI Interrupt

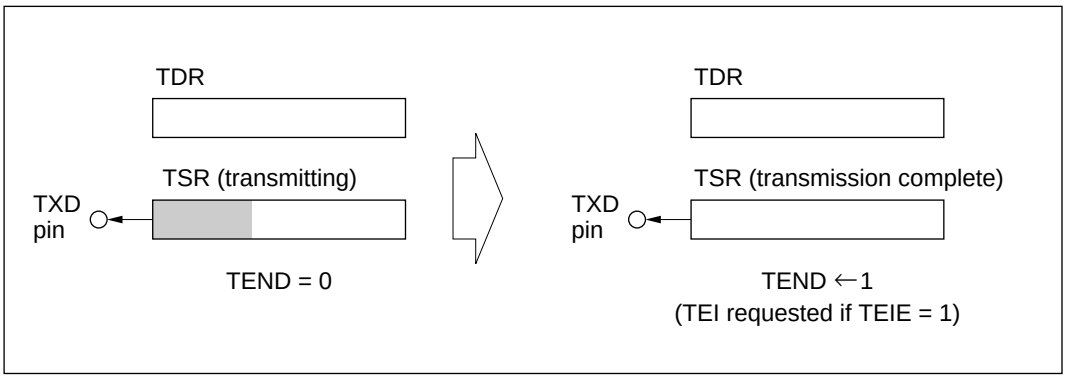


Figure 10-2-2 (c) TEND Setting and TEI Interrupt

10.2.4 Operation in Asynchronous Mode

In asynchronous communication mode, a start bit indicating the start of communication and a stop bit (1 or 2 bits) indicating the end of communication are added to each character that is sent. In this way synchronization is achieved for each character as a self-contained unit.

SCI3 consists of independent transmit and receive modules, giving it the capability of full duplex communication. Both the transmit and receive modules have a double-buffer configuration, allowing data to be read or written during communication operations so that data can be transmitted and received continuously.

1. Transmit/receive formats

Figure 10-2-3 shows the general format for asynchronous serial communication.

The communication line in asynchronous communication mode normally stays at the high level, in the “mark” state. SCI3 monitors the communication line, and begins serial data communication when it detects a “space” (low-level signal), which is regarded as a start bit.

One character consists of a start bit (low level), transmit/receive data (in LSB-first order), a parity bit (high or low level), and finally a stop bit (high level), in this order.

In asynchronous data receiving, synchronization is with the falling edge of the start bit. SCI3 samples data on the 8th pulse of a clock that has 16 times the frequency of the bit rate, so each bit of data is latched at its center.

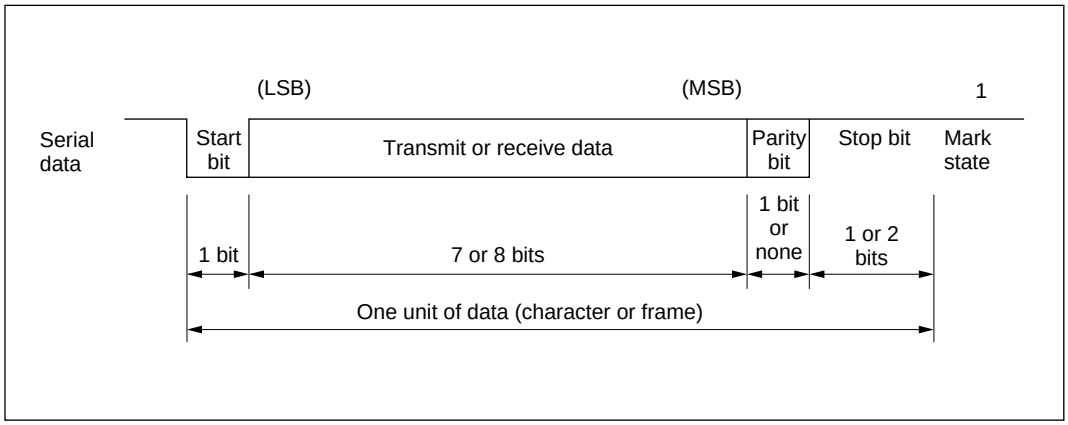


Figure 10-2-3 Data Format in Asynchronous Serial Communication Mode

Table 10-2-11 shows the 12 formats that can be selected in asynchronous mode. The format is selected in the serial mode register (SMR).

Table 10-2-11 Serial Communication Formats in Asynchronous Mode

SMR Setting				Serial Communication Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
0	*	1	0	S	8-bit data								MPB	STOP	
0	*	1	1	S	8-bit data								MPB	STOP	STOP
1	*	1	0	S	7-bit data							MPB	STOP		
1	*	1	1	S	7-bit data							MPB	STOP	STOP	

Notation: S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

Note: * Don't care

2. Clock

The clock source is determined by bit COM in SMR and bits CKE1 and CKE0 in serial control register 3 (SCR3). See table 10-2-9 for the settings. Either an internal clock source can be used to run the built-in baud rate generator, or an external clock source can be input at pin SCK_3.

When an external clock source is input, it should have a frequency 16 times the desired bit rate.

When an internal clock source is used, SCK_3 is used as the clock output pin. The clock output has the same frequency as the serial bit rate, and is synchronized as in figure 10-2-4 so that the rising edge of the clock occurs in the center of each bit of transmit/receive data.

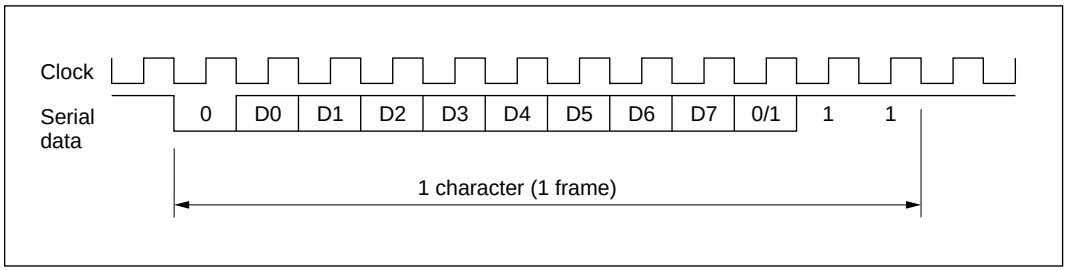


Figure 10-2-4 Phase Relation of Output Clock and Communication Data in Asynchronous Mode (8-Bit Data, Parity Bit Added, and 2 Stop Bits)

3. Data transmit/receive operations

• SCI3 initialization

Before data is sent or received, bits TE and RE in serial control register 3 (SCR3) must be cleared to 0, after which initialization can be performed using the following procedure.

Note:

When modifying the operation mode, transfer format or other settings, always be sure to clear bits TE and RE first. When TE is cleared to 0, bit TDRE will be set to 1. Clearing RE does not clear the status flags RDRF, PER, FER, or OER, or alter the contents of the receive data register (RDR).

When an external clock is used in asynchronous mode, do not stop the clock during operation, including during initialization. When an external clock is used in synchronous mode, do not supply the clock during initialization.

Figure 10-2-5 shows a typical flow chart for SCI3 initialization.

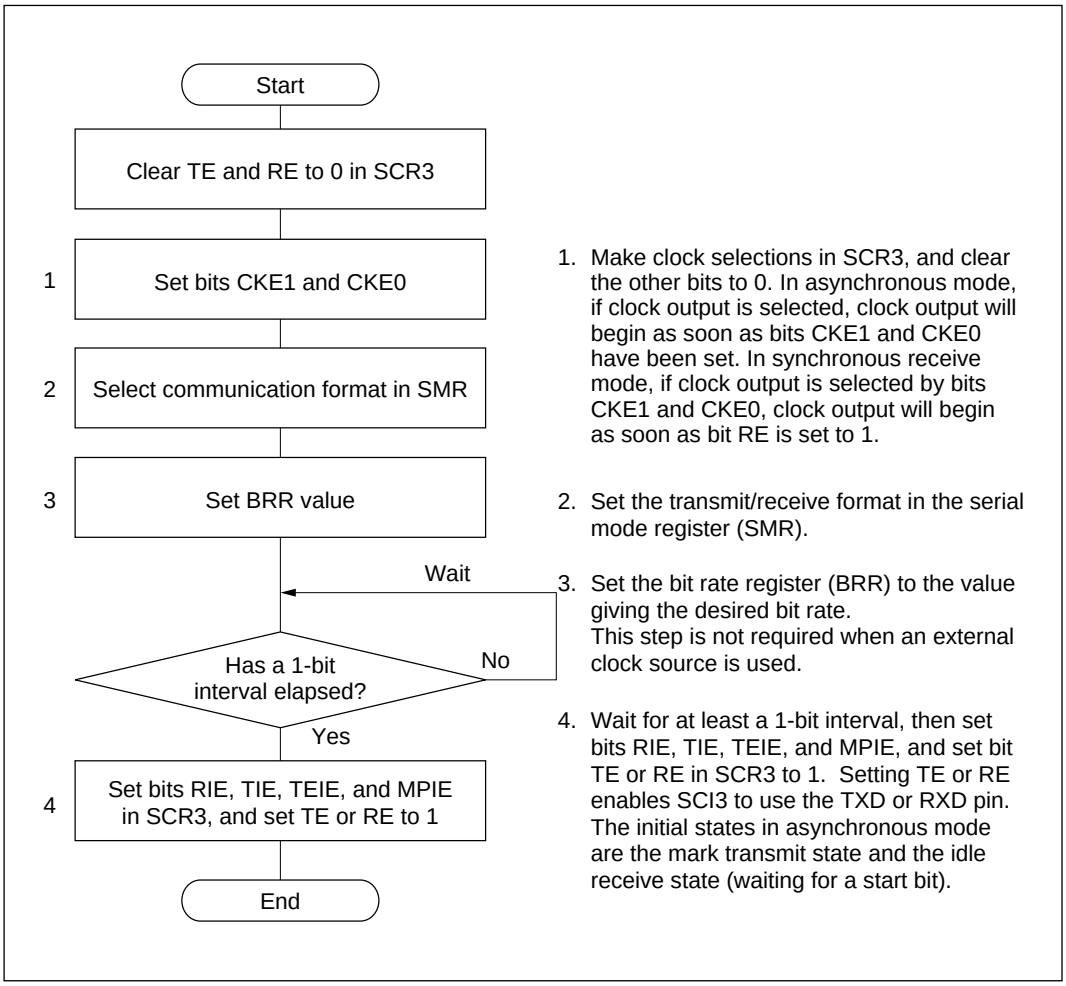


Figure 10-2-5 Typical Flow Chart when SCI3 Is Initialized

• Transmitting

Figure 10-2-6 shows a typical flow chart for data transmission. After SCI3 initialization, follow the procedure below.

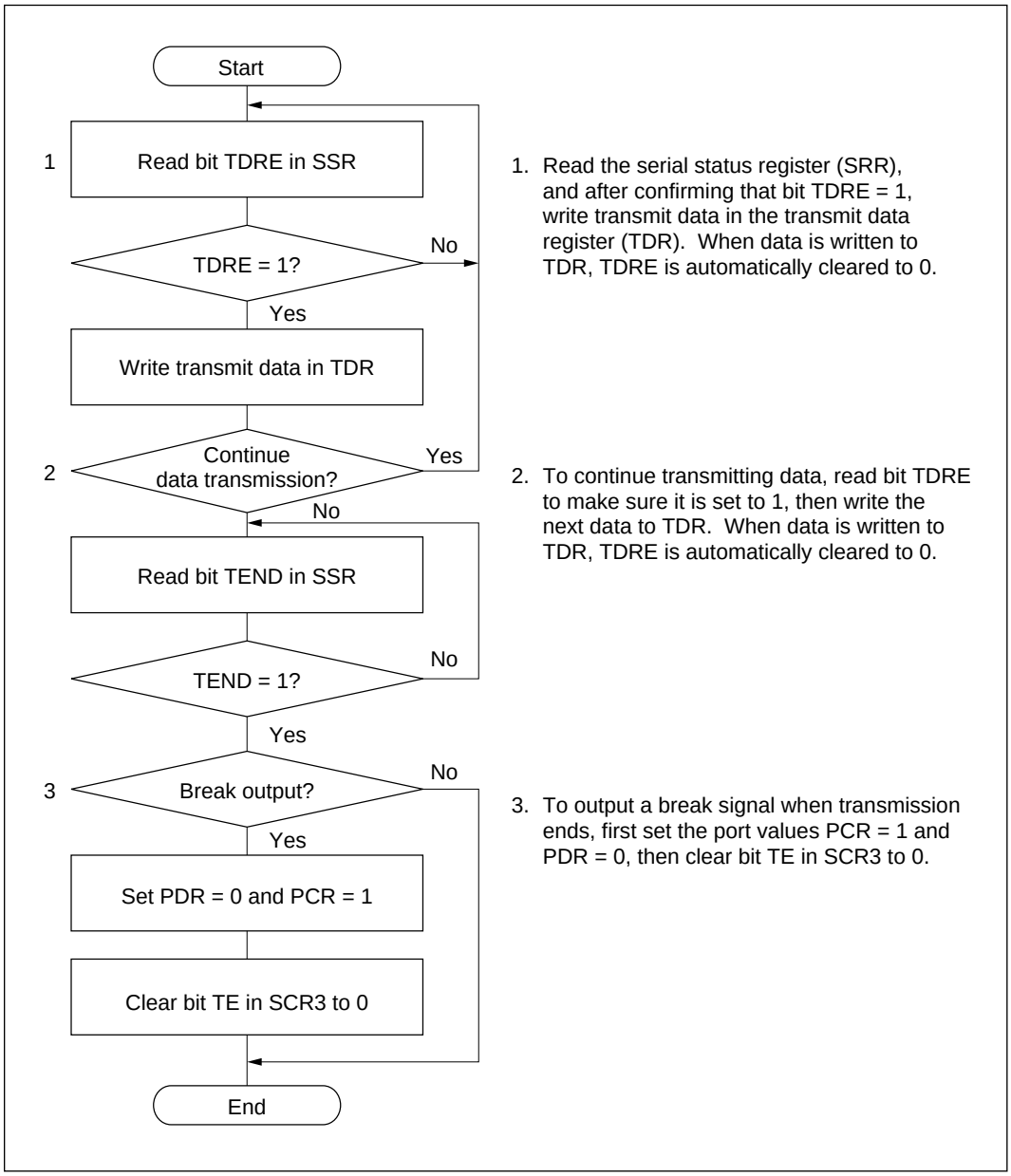


Figure 10-2-6 Typical Data Transmission Flow Chart (Asynchronous Mode)

SCI3 operates as follows during data transmission in asynchronous mode.

SCI3 monitors bit TDRE in SSR. When this bit is cleared to 0, SCI3 recognizes that there is data written in the transmit data register (TDR), which it transfers to the transmit shift register (TSR). Then TDRE is set to 1 and transmission starts. If bit TIE in SCR3 is set to 1, a TXI interrupt is requested.

Serial data is transmitted from pin TXD using the communication format outlined in table 10-2-11. Next, TDRE is checked as the stop bit is being transmitted.

If TDRE is 0, data is transferred from TDR to TSR, and after the stop bit is sent, transmission of the next frame starts. If TDRE is 1, the TEND bit in SSR is set to 1, and after the stop bit is sent the output remains at 1 (mark state). A TEI interrupt is requested in this state if bit TEIE in SCR3 is set to 1.

Figure 10-2-7 shows a typical operation in asynchronous transmission mode.

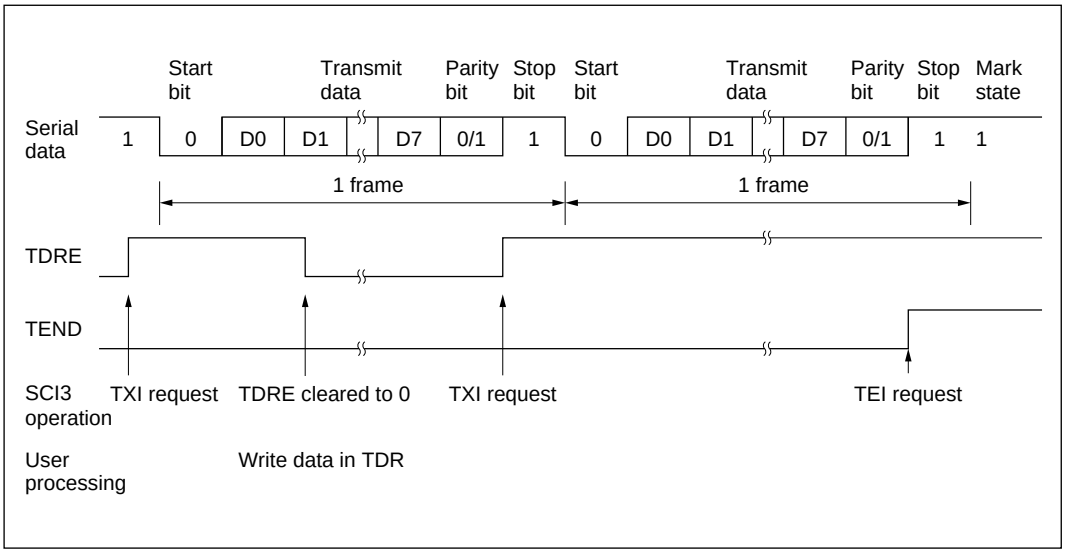


Figure 10-2-7 Typical Transmit Operation in Asynchronous Mode (8-Bit Data, Parity Bit Added, and 1 Stop Bit)

• Receiving

Figure 10-2-8 shows a typical flow chart for receiving serial data. After SCI3 initialization, follow the procedure below.

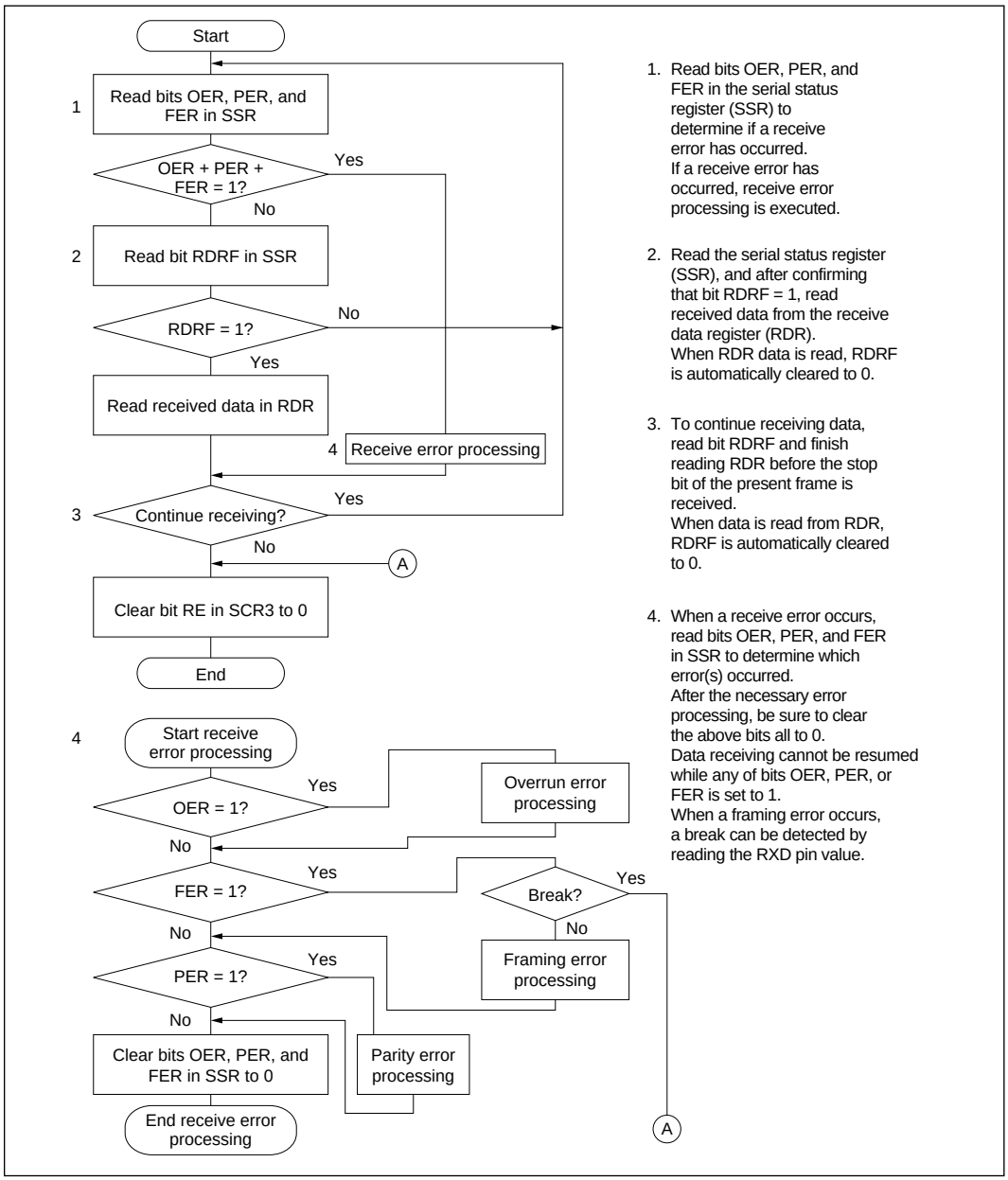


Figure 10-2-8 Typical Serial Data Receiving Flow Chart in Asynchronous Mode

SCI3 operates as follows when receiving serial data in asynchronous mode.

SCI3 monitors the communication line, and when a start bit (0) is detected it performs internal synchronization and starts receiving. The communication format for data receiving is as outlined in table 10-2-11. Received data is set in RSR from LSB to MSB, then the parity bit and stop bit(s) are received. After receiving the data, SCI3 performs the following checks:

- Parity check: The number of 1s received is checked to see if it matches the odd or even parity selected in bit PM of SMR.
- Stop bit check: The stop bit is checked for a value of 1. If there are two stop bits, only the first bit is checked.
- Status check: The RDRF bit is checked for a value of 0 to make sure received data can be transferred from RSR to RDR.

If no receive error is detected by the above checks, bit RDRF is set to 1 and the received data is stored in RDR. At that time, if bit RIE in SCR3 is set to 1, an RXI interrupt is requested. If the error check detects a receive error, the appropriate error flag (OER, PER, or FER) is set to 1. RDRF retains the same value as before the data was received. If at this time bit RIE in SCR3 is set to 1, an ERI interrupt is requested.

Table 10-2-12 gives the receive error detection conditions and the processing of received data in each case.

Note: Data receiving cannot be continued while a receive error flag is set. Before continuing the receive operation it is necessary to clear the OER, FER, PER, and RDRF flags to 0.

Table 10-2-12 Receive Error Conditions and Received Data Processing

Receive Error	Abbrev.	Detection Conditions	Received Data Processing
Overrun error	OER	Receiving of the next data ends while bit RDRF in SSR is still set to 1	Received data is not transferred from RSR to RDR
Framing error	FER	Stop bit is 0	Received data is transferred from RSR to RDR
Parity error	PER	Received data does not match the parity (odd/even) set in SMR	Received data is not transferred from RSR to RDR

Figure 10-2-9 shows a typical SCI3 data receive operation in asynchronous mode.

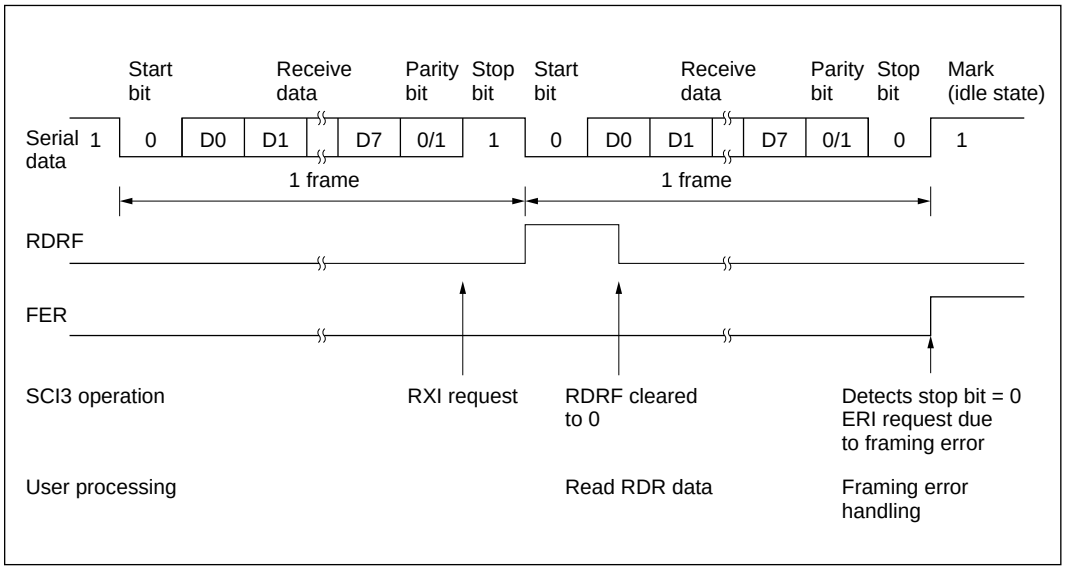


Figure 10-2-9 Typical Receive Operation in Asynchronous Mode (8-Bit Data, Parity Bit Added, and 1 Stop Bit)

10.2.5 Operation in Synchronous Mode

In synchronous mode, data is sent or received in synchronization with clock pulses. This mode is suited to high-speed serial communication.

SCI3 consists of independent transmit and receive modules, so full duplex communication is possible, sharing the same clock between both modules. Both the transmit and receive modules have a double-buffer configuration. This allows data to be written during a transmit operation so that data can be transmitted continuously, and enables data to be read during a receive operation so that data can be received continuously.

1. Transmit/receive format

Figure 10-2-10 shows the general communication data format for synchronous communication.

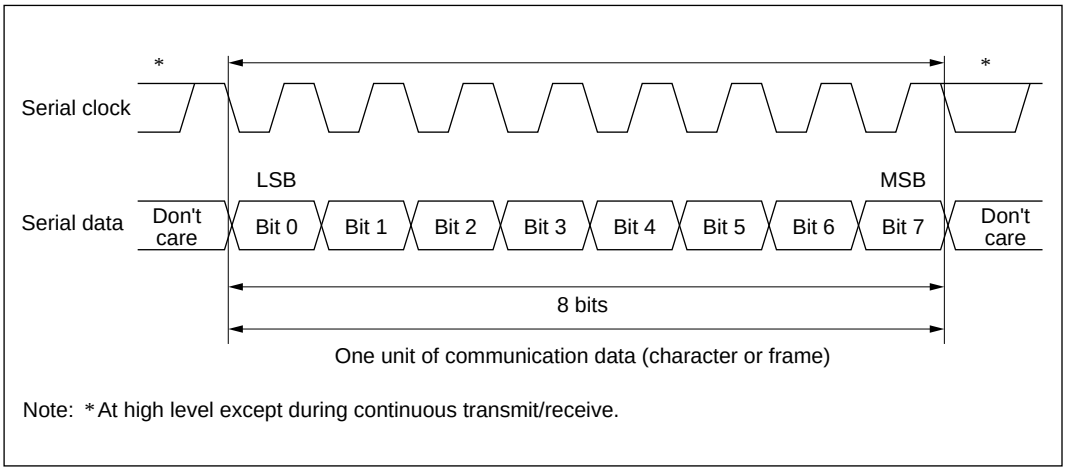


Figure 10-2-10 Data Format in Synchronous Communication Mode

In synchronous communication, data on the communication line is output from one falling edge of the serial clock until the next falling edge. Data is guaranteed valid at the rising edge of the serial clock.

One character of data starts from the LSB and ends with the MSB. The communication line retains the MSB state after the MSB is output.

In synchronous receive mode, SCI3 latches receive data in synchronization with the rising edge of the serial clock.

The transmit/receive format is fixed at 8-bit data. No parity bit or multiprocessor bit is added in this mode.

2. Clock

Either an internal clock from the built-in baud rate generator is used, or an external clock is input at pin SCK_3. The choice of clock sources is designated by bit COM in SMR and bits CKE1 and CKE0 in serial control register 3 (SCR3). See table 10-2-9 for details on selecting the clock source.

When operation is based on an internal clock, a serial clock is output at pin SCK_3. Eight clock pulses are output per character of transmit/receive data. When no transmit or receive operation is being performed, the pin is held at the high level.

3. Data transmit/receive operations

• SCI3 initialization

Before transmitting or receiving data, follow the SCI3 initialization procedure explained under 10.2.4, SCI3 Initialization, and illustrated in figure 10-2-5.

• Transmitting

Figure 10-2-11 shows a typical flow chart for data transmission. After SCI3 initialization, follow the procedure below.

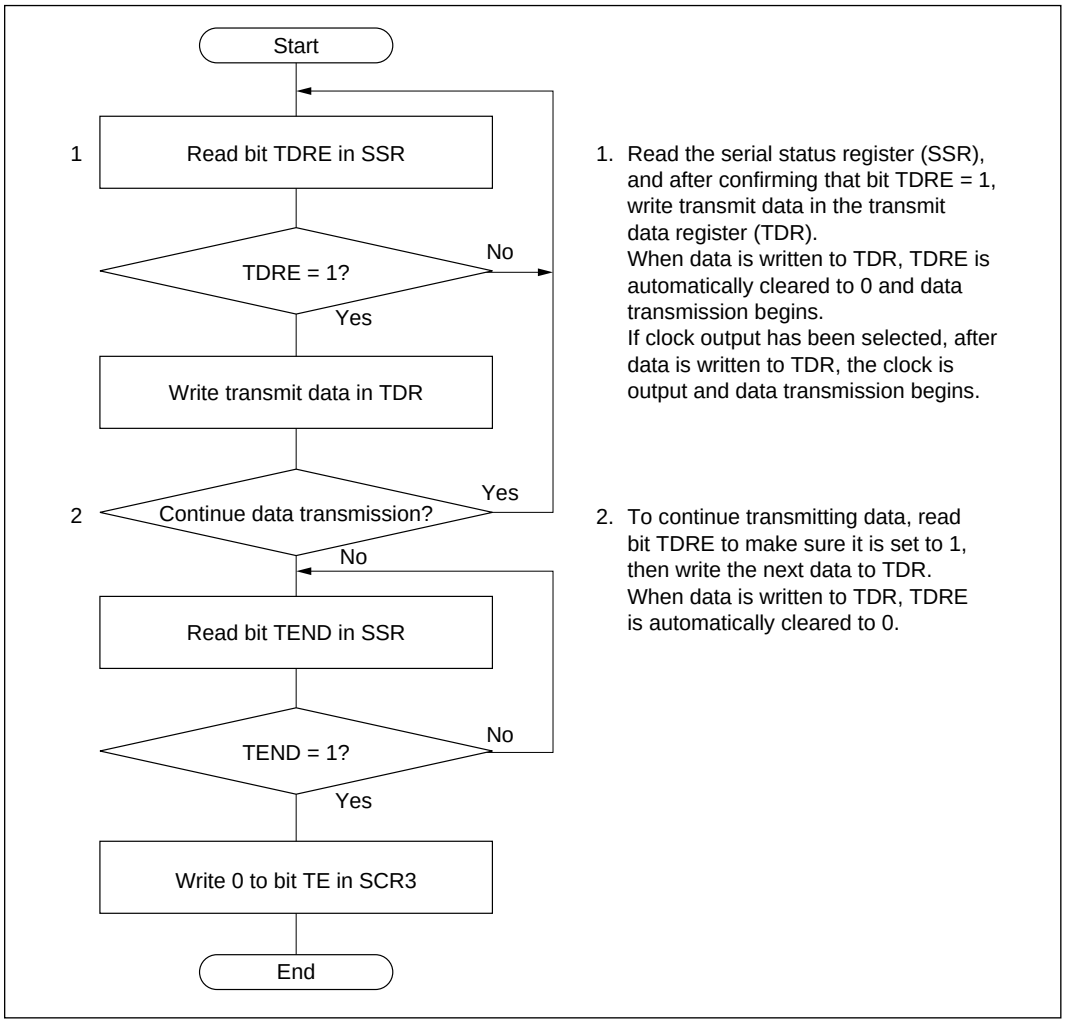


Figure 10-2-11 Typical Data Transmission Flow Chart in Synchronous Mode

SCI3 operates as follows during data transmission in synchronous mode.

SCI3 monitors bit TDRE in SSR. When this bit is cleared to 0, SCI3 recognizes that there is data written in the transmit data register (TDR), which it transfers to the transmit shift register (TSR). Then TDRE is set to 1 and transmission starts. If bit TIE in SCR3 is set to 1, a TXI interrupt is requested.

If clock output is selected, SCI3 outputs eight serial clock pulses. If an external clock is used, data is output in synchronization with the clock input.

Serial data is transmitted from pin TXD in order from LSB (bit 0) to MSB (bit 7).

Then TDRE is checked as the MSB (bit 7) is being transmitted. If TDRE is 0, data is transferred from TDR to TSR, and after the MSB (bit 7) is sent, transmission of the next frame starts. If TDRE is 1, the TEND bit in SSR is set to 1, and after the MSB (bit 7) has been sent, the MSB state is maintained. A TEI interrupt is requested in this state if bit TEIE in SCR3 is set to 1.

After data transmission ends, pin SCK_3 is held at the high level.

Note: Data transmission cannot take place while any of the receive error flags (OER, FER, PER) is set to 1. Be sure to confirm that these error flags are cleared to 0 before starting transmission.

Figure 10-2-12 shows a typical SCI3 transmit operation in synchronous mode.

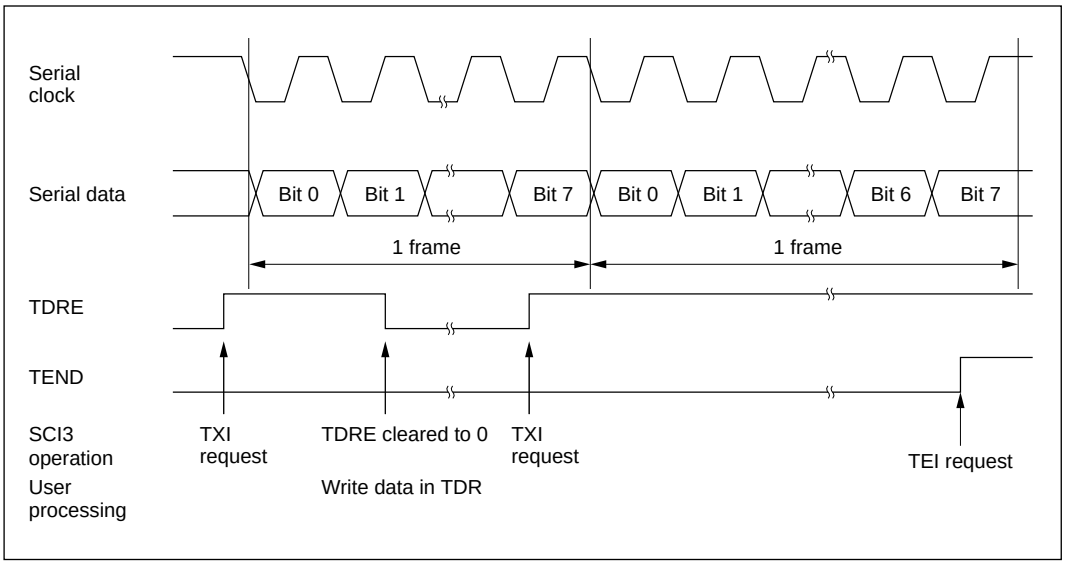


Figure 10-2-12 Typical SCI3 Transmit Operation in Synchronous Mode

• Receiving

Figure 10-2-13 shows a typical flow chart for receiving data. After SCI3 initialization, follow the procedure below.

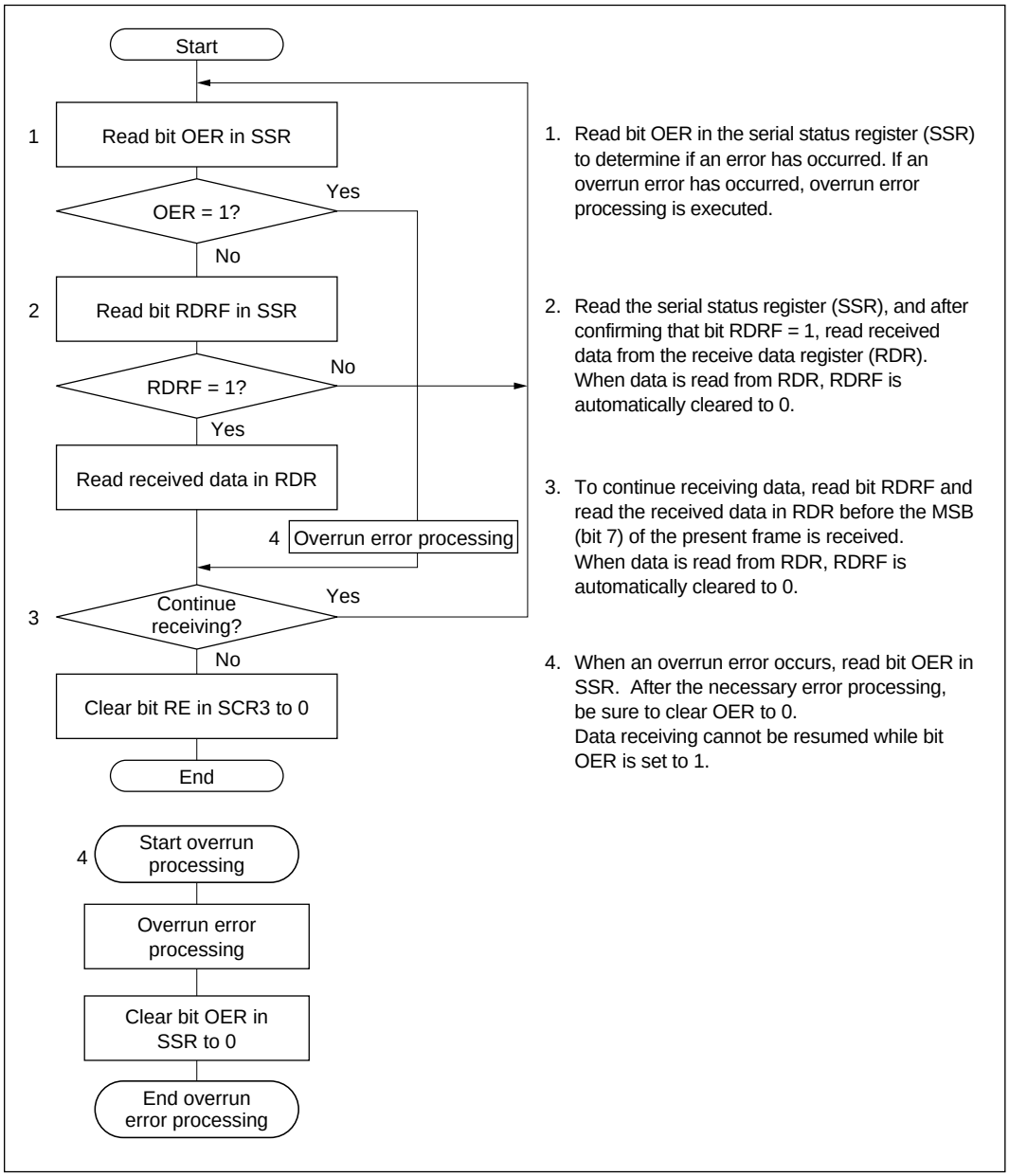


Figure 10-2-13 Typical Data Receiving Flow Chart in Synchronous Mode

SCI3 operates as follows when receiving serial data in synchronous mode.

SCI3 synchronizes internally with the input or output of the serial clock and starts receiving. Received data is set in RSR from LSB to MSB.

After data has been received, SCI3 checks to confirm that the value of bit RDRF is 0 indicating that received data can be transferred from RSR to RDR. If this check passes, RDRF is set to 1 and the received data is stored in RDR. At this time, if bit RIE in SCR3 is set to 1, an RXI interrupt is requested. If an overrun error is detected, OER is set to 1 and RDRF remains set to 1. Then if bit RIE in SCR3 is set to 1, an ERI interrupt is requested.

For the overrun error detection conditions and receive data processing, see table 10-2-12.

Note: Data receiving cannot be continued while a receive error flag is set. Before continuing the receive operation it is necessary to clear the OER, FER, PER, and RDRF flags to 0.

Figure 10-2-14 shows a typical receive operation in synchronous mode.

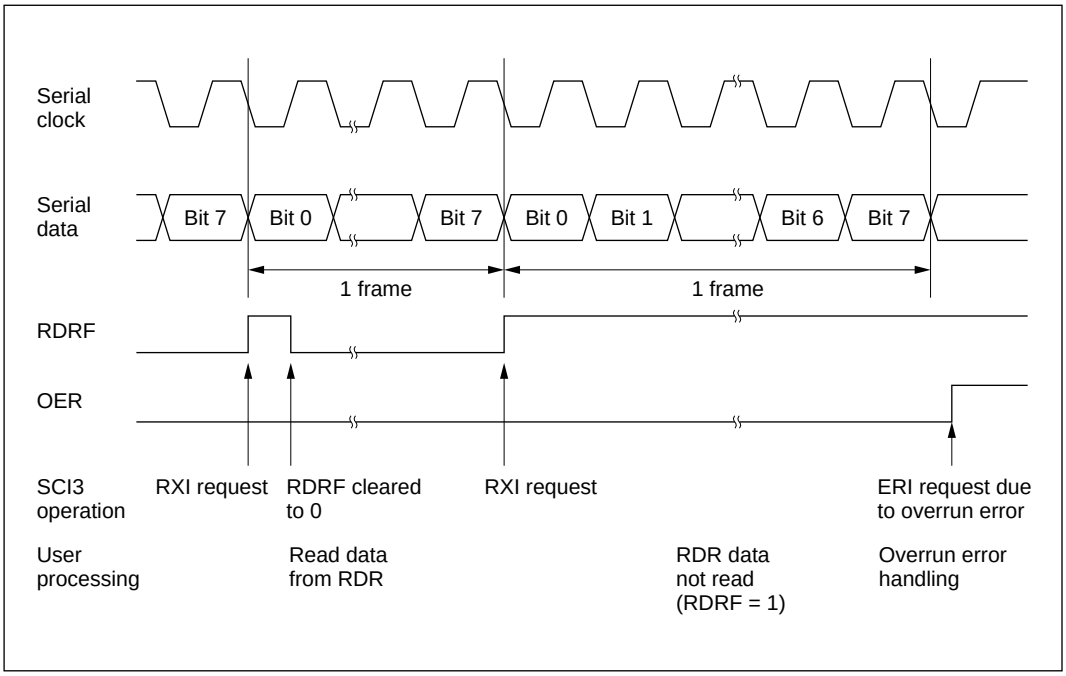


Figure 10-2-14 Typical Receive Operation in Synchronous Mode

- Simultaneous transmit/receive

Figure 10-2-15 shows a typical flow chart for transmitting and receiving simultaneously. After SCI3 synchronization, follow the procedure below.

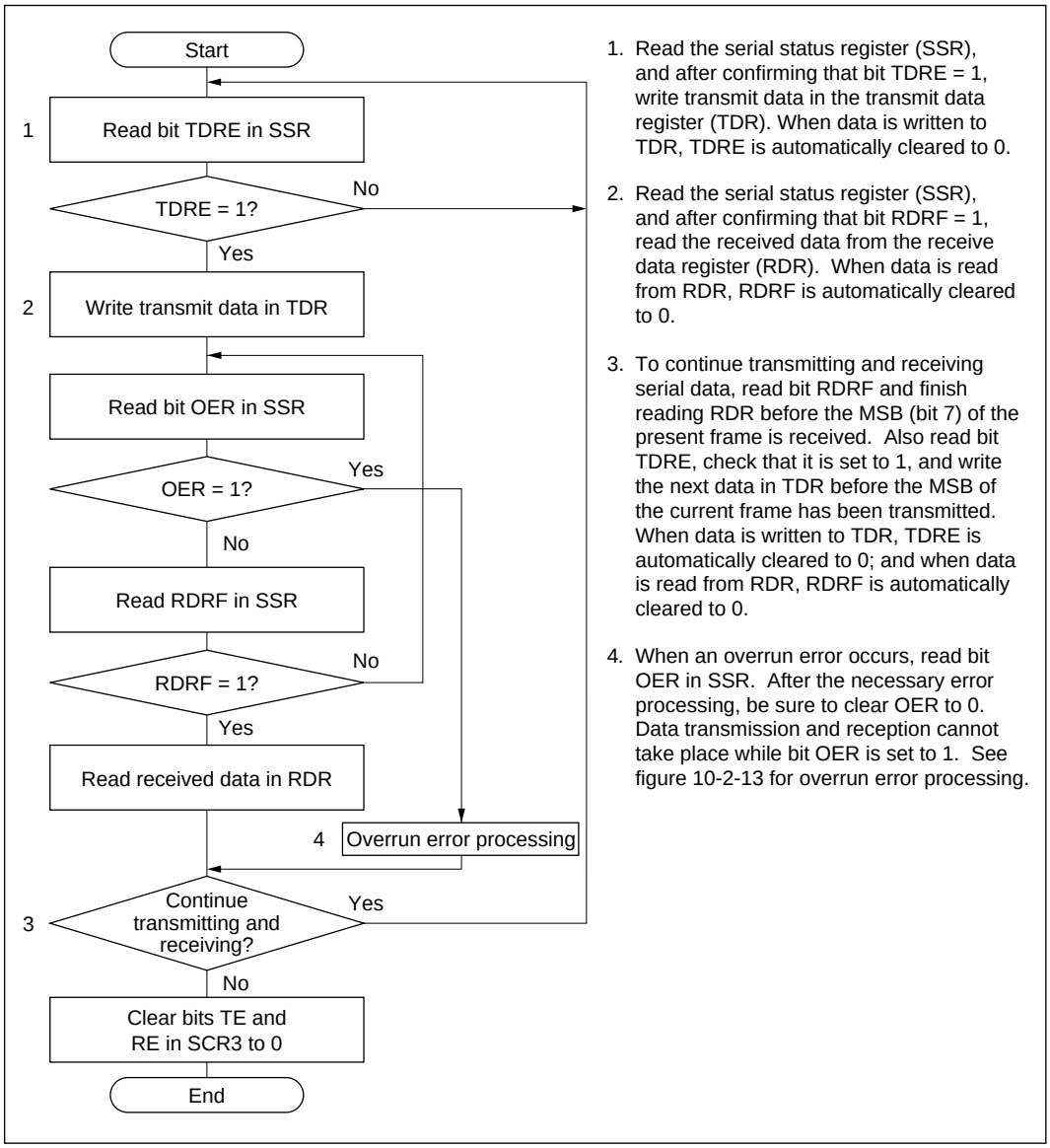


Figure 10-2-15 Simultaneous Transmit/Receive Flow Chart in Synchronous Mode

Notes: 1. To switch from transmitting to simultaneous transmitting and receiving, use the following procedure.
- First confirm that TDRE and TEND are both set to 1 and that SCI3 has finished transmitting. Next clear TE to 0. Then set both TE and RE to 1.

2. To switch from receiving to simultaneous transmitting and rceiving, use the following procedure.
- After confirming that SCI3 has finished receiving, clear RE to 0. Next, after confirming that RDRF and the error flags (OER FER, PER) are all 0, set both TE and RE to 1.

10.2.6 Multiprocessor Communication Function

The multiprocessor communication function enables several processors to share a single serial communication line. The processors communicate in asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by an ID code. A serial communication cycle consists of two cycles: an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit is 1 in an ID-sending cycle, and 0 in a data-sending cycle.

The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0. When a receiving processor receives data with the multiprocessor bit set to 1, it compares the data with its own ID. If the data matches its ID, the receiving processor continues to receive incoming data. If the data does not match its ID, the receiving processor skips further incoming data until it again receives data with the multiprocessor bit set to 1. Multiple processors can send and receive data in this way.

Figure 10-2-16 shows an example of communication among different processors using a multiprocessor format.

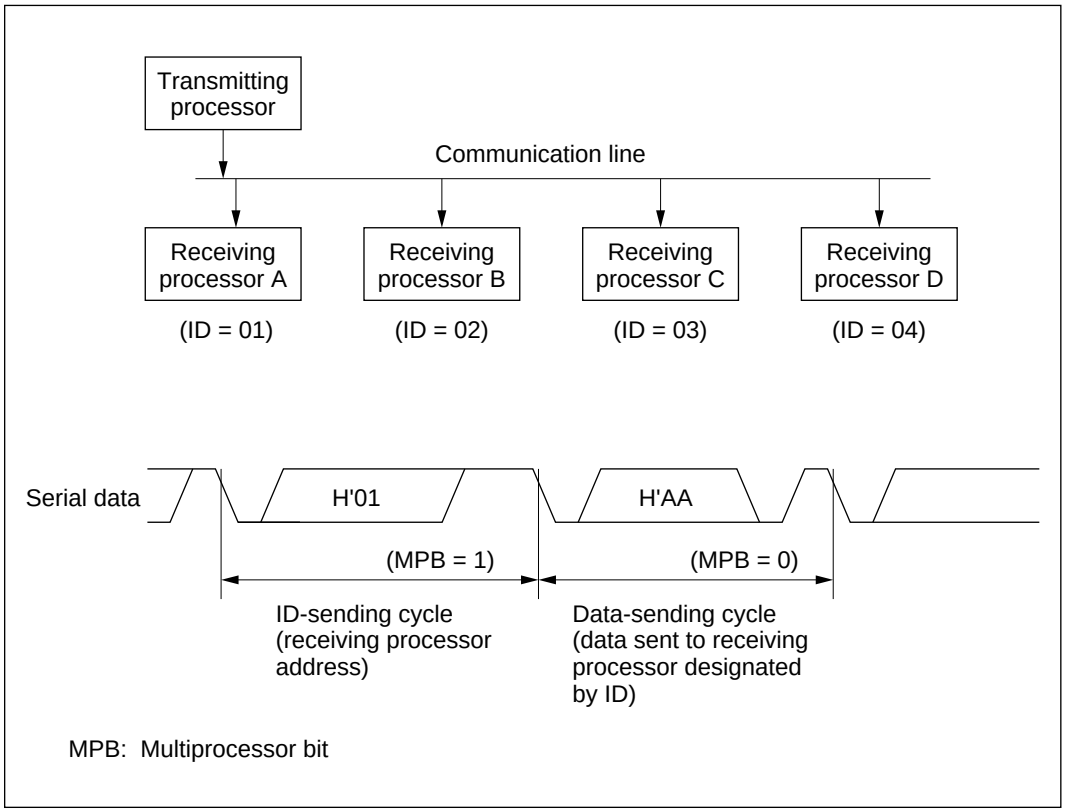


Figure 10-2-16 Example of Interprocessor Communication Using Multiprocessor Format (Data H'AA Sent to Receiving Processor A)

Four communication formats are available. Parity-bit settings are ignored when a multiprocessor format is selected. For details see table 10-2-11.

For a description of the clock used in multiprocessor communication, see 10.2.4, Operation in Asynchronous Mode.

- Transmitting multiprocessor data

Figure 10-2-17 shows a typical flow chart for multiprocessor serial data transmission. After SCI3 initialization, follow the procedure below.

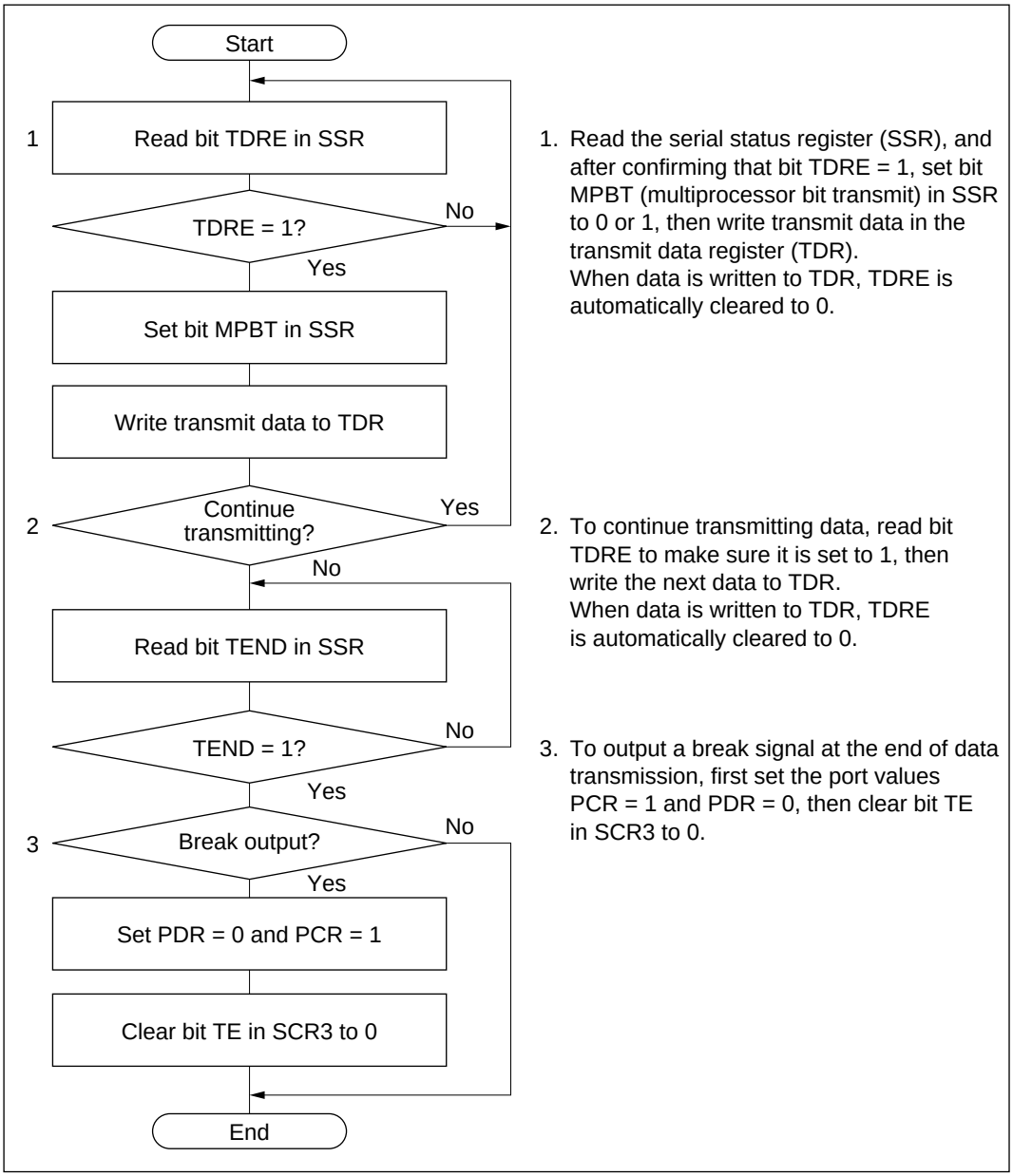


Figure 10-2-17 Typical Multiprocessor Data Transmission Flow Chart

SCI3 operates as follows during data transmission using a multiprocessor format.

SCI3 monitors bit TDRE in SSR. When this bit is cleared to 0, SCI3 recognizes that there is data written in the transmit data register (TDR), which it transfers to the transmit shift register (TSR). Then TDRE is set to 1 and transmission starts. If bit TIE in SCR3 is set to 1, a TXI interrupt is requested.

Serial data is transmitted from pin TXD using the communication format outlined in table 10-2-11.

Next, TDRE is checked as the stop bit is being transmitted. If TDRE is 0, data is transferred from TDR to TSR, and after the stop bit is sent, transmission of the next frame starts. If TDRE is 1, the TEND bit in SSR is set to 1, and after the stop bit is sent the output remains at 1 (mark state). A TEI interrupt is requested in this state if bit TEIE (transmit end interrupt enable) in SCR3 is set to 1.

Figure 10-2-18 shows a typical SCI3 operation in multiprocessor communication mode.

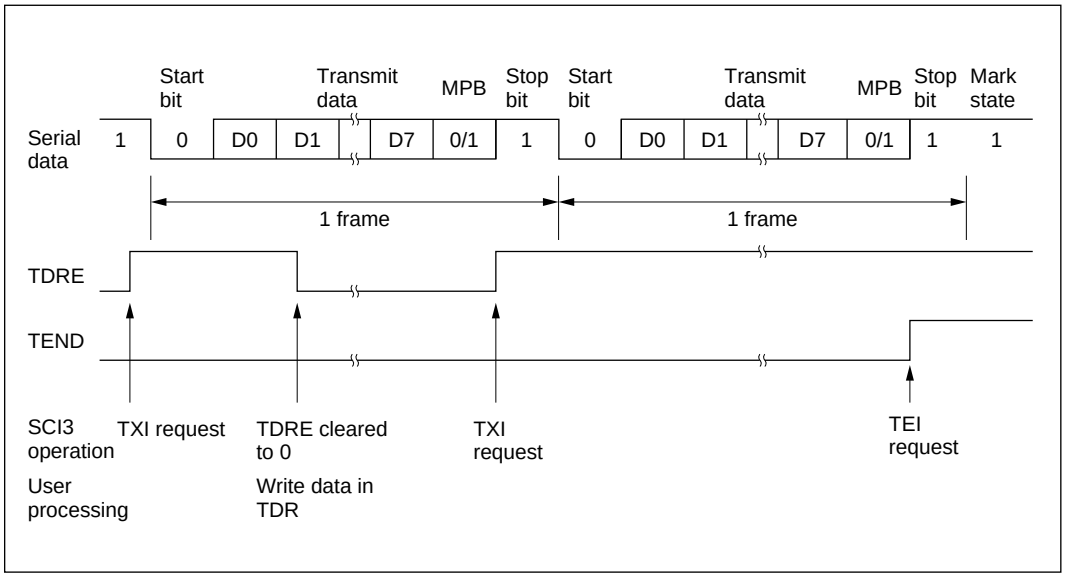


Figure 10-2-18 Typical Multiprocessor Format Transmit Operation (8-Bit Data, Multiprocessor Bit Added, and 1 Stop Bit)

• Receiving multiprocessor data

Figure 10-2-19 shows a typical flow chart for receiving data using a multiprocessor format. After SCI3 initialization, follow the procedure below.

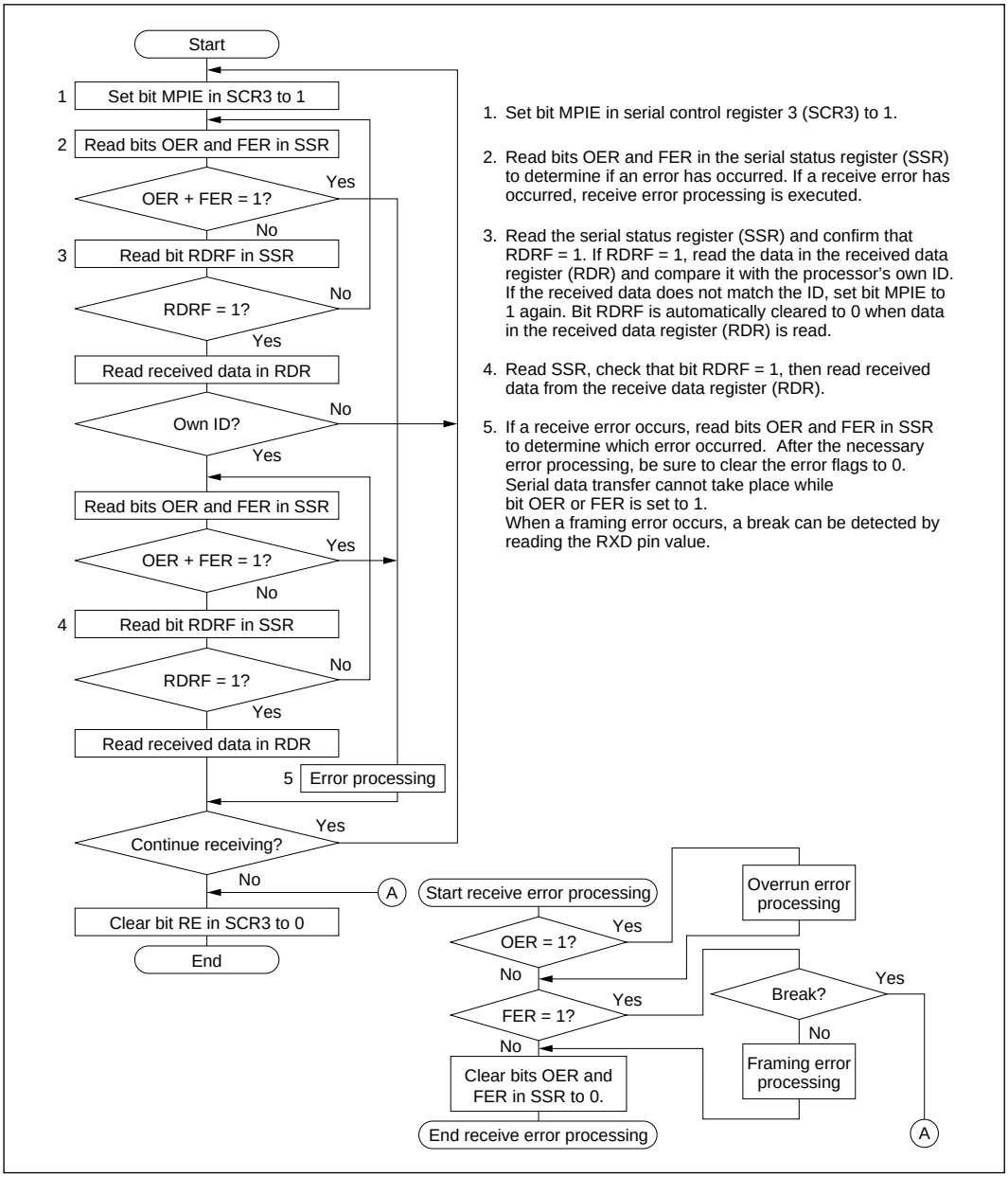


Figure 10-2-19 Typical Flow Chart for Receiving Serial Data Using Multiprocessor Format

Figure 10-2-20 gives an example of data reception using a multiprocessor format.

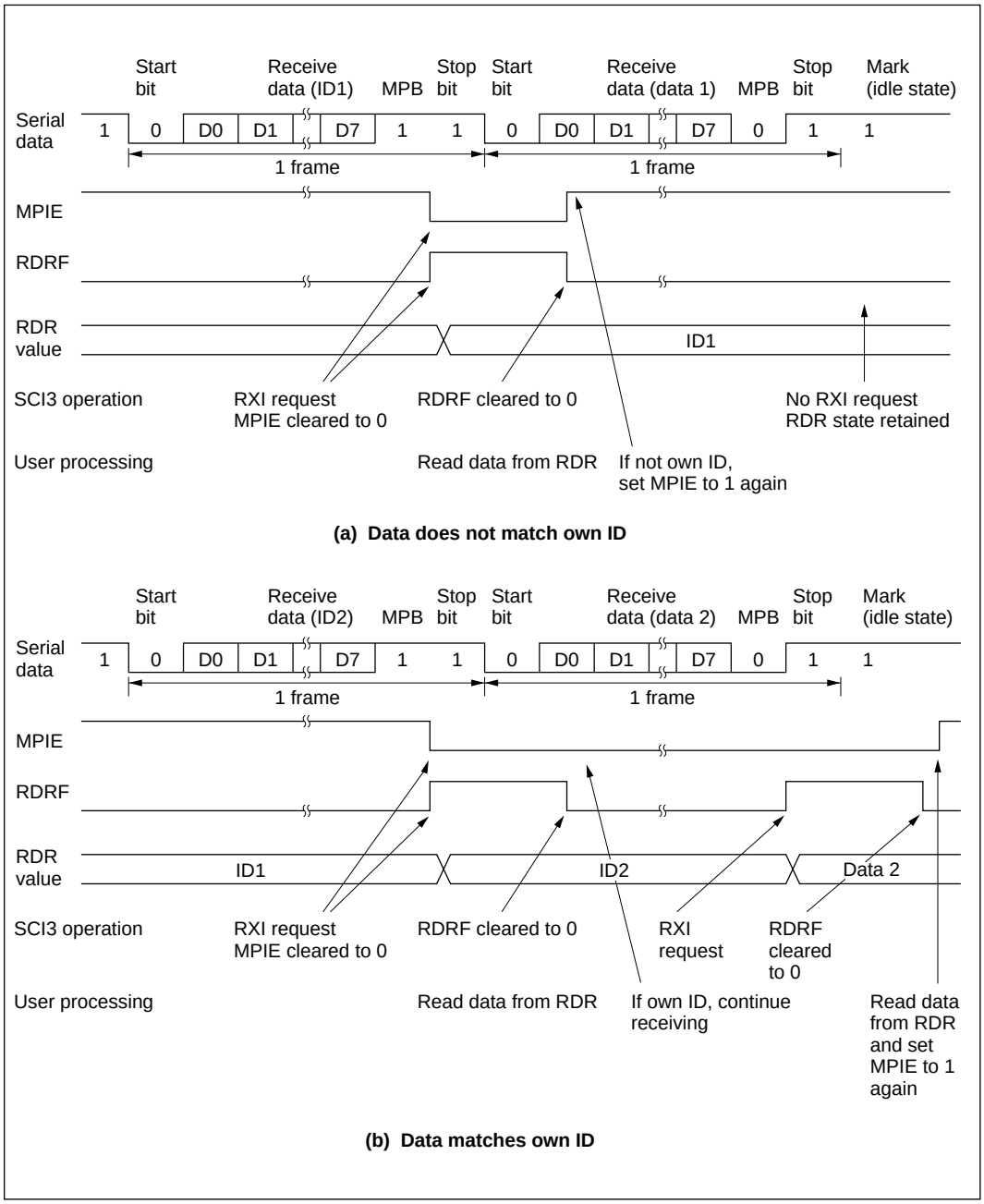


Figure 10-2-20 Example of Multiprocessor Format Receive Operation (8-Bit Data, Multiprocessor Bit Added, and 1 Stop Bit)

10.2.7 Interrupts

SCI3 has six interrupt sources: transmit end, transmit data empty, receive data full, and the three receive error interrupts (overrun error, framing error, and parity error). All share a common interrupt vector. Table 10-2-13 describes each interrupt.

Table 10-2-13 SCI3 Interrupts

Interrupt	Description	Vector Address
RXI	Interrupt request due to receive data register full (RDRF)	H'0024
TXI	Interrupt request due to transmit data register empty (TDRE)	
TEI	Interrupt request due to transmit end (TEND)	
ERI	Interrupt request due to receive error (OER, FER, or PER)	

The interrupt requests are enabled and disabled by bits TIE and RIE of SCR3.

When bit TDRE in SSR is set to 1, TXI is requested. When bit TEND in SSR is set to 1, TEI is requested. These two interrupt requests occur during data transmission.

The initial value of bit TDRE is 1. Accordingly, if the transmit data empty interrupt request (TXI) is enabled by setting bit TIE to 1 in SCR3 before placing transmit data in TDR, TXI will be requested even though no transmit data has been readied.

Likewise, the initial value of bit TEND is 1. Accordingly, if the transmit end interrupt request (TEI) is enabled by setting bit TEIE to 1 in SCR3 before placing transmit data in TDR, TEI will be requested even though no data has been transmitted.

These interrupt features can be used to advantage by programming the interrupt handler to move the transmit data into TDR. When this technique is not used, the interrupt enable bits (TIE and TEIE) should not be set to 1 until after TDR has been loaded with transmit data, to avoid unwanted TXI and TEI interrupts.

When bit RDRF in SSR is set to 1, RXI is requested. When any of SSR bits OER, FER, or PER is set to 1, ERI is requested. These two interrupt requests occur during the receiving of data.

Details on interrupts are given in 3.3, Interrupts.

10.2.8 Application Notes

When using SCI3, attention should be paid to the following matters.

1. Relation between bit TDRE and writing data to TDR

Bit TDRE in the serial status register (SSR) is a status flag indicating that TDR does not contain new transmit data. TDRE is automatically cleared to 0 when data is written to TDR. When SCI3 transfers data from TDR to TSR, bit TDRE is set to 1.

Data can be written to TDR regardless of the status of bit TDRE. However, if new data is written to TDR while TDRE is cleared to 0, assuming the data held in TDR has not yet been shifted to TSR, it will be lost. For this reason it is advisable to confirm that bit TDRE is set to 1 before each write to TDR and not write to TDR more than once without checking TDRE in between.

2. Operation when multiple receive errors occur at the same time

When two or more receive errors occur at the same time, the status flags in SSR are set as shown in table 10-2-14. If an overrun error occurs, data is not transferred from RSR to RDR, and receive data is lost.

Table 10-2-14 SSR Status Flag States and Transfer of Receive Data

SSR Status Flags				Receive Data Transfer	
RDRF*	OER	FER	PER	(RSR → RDR)	Receive Error Status
1	1	0	0	×	Overrun error
0	0	1	0	○	Framing error
0	0	0	1	○	Parity error
1	1	1	0	×	Overrun error + framing error
1	1	0	1	×	Overrun error + parity error
0	0	1	1	○	Framing error + parity error
1	1	1	1	×	Overrun error + framing error + parity error

Notation: ○: Receive data transferred from RSR to RDR
×: Receive data not transferred from RSR to RDR

Note: *RDRF keeps the same state as before the data was received. However, if due to a late read of received data in one frame an overrun error occurs in the next frame, RDRF is cleared to 0 when RDR is read.

3. Break detection and processing

Break signals can be detected by reading the RXD pin directly when a framing error (FER) is detected. In the break state the input from the RXD pin consists of all 0s, so FER is set and the parity error flag (PER) may also be set. In the break state SCI3 continues to receive, so if the FER bit is cleared to 0 it will be set to 1 again.

4. Sending a mark or break signal

When TE is cleared to 0 the TXD pin becomes an I/O port, set by the PDR and PCR bits. This feature can be used to place the TXD pin in the mark state or send a break signal.

To place the serial communication line in the mark (1) state before TE is set to 1, set the PDR and PCR bits both to 1. Since TE is cleared to 0, TXD becomes a general output port outputting the value 1.

To send a break signal during data transmission, set the PCR bit to 1 and clear the PDR bit to 0, then clear TE to 0. When TE is cleared to 0 the transmitter is initialized, regardless of its current state, so the TXD pin becomes an output port outputting the value 0.

5. Receive error flags and transmit operation (sysnchronous mode only)

When a receive error flag (OER, PER, or FER) is set to 1, SCI3 will not start transmitting even if TDRE is cleared to 0. Be sure to clear the receive error flags to 0 when starting to transmit. Note that clearing RE to 0 does not clear the receive error flags.

6. Receive data sampling timing and receive margin in asynchronous mode

In asynchronous mode SCI3 operates on a base clock with 16 times the bit rate frequency. In receiving, SCI3 synchronizes internally with the falling edge of the start bit, which it samples on the base clock. Receive data is latched at the rising edge of the eighth base clock pulse. See figure 10-2-21.

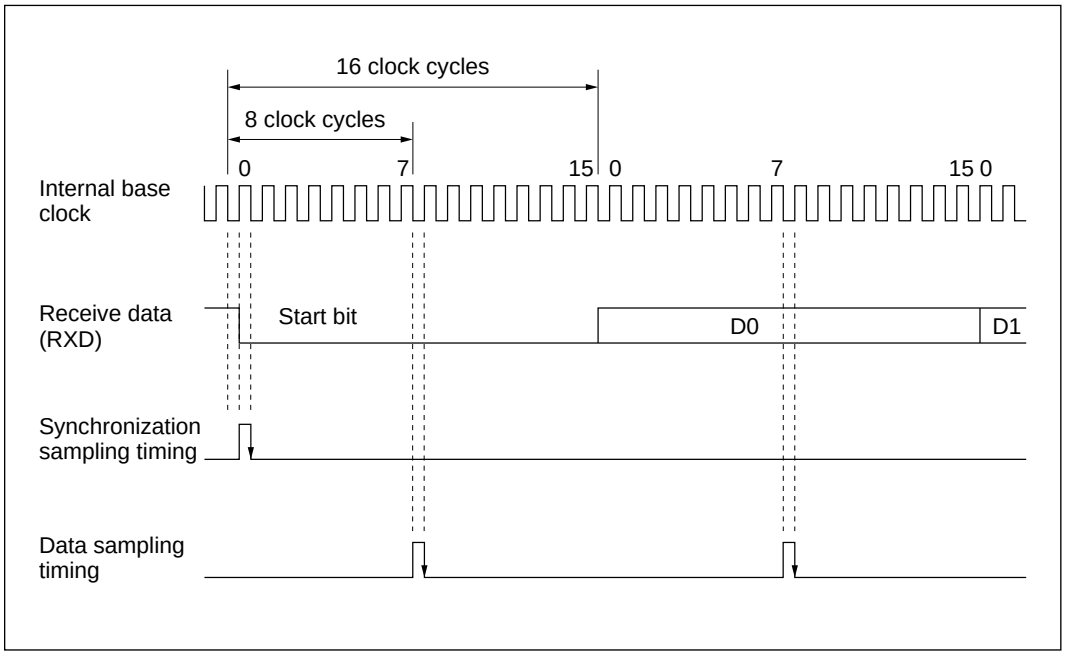


Figure 10-2-21 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be derived from the following equation.

$M = \{(0.5 - 1/2N) - (D - 0.5) / N - (L - 0.5) F\} \times 100\%$.....Equation (1)

M: Receive margin (%)
N: Ratio of clock frequency to bit rate (N = 16)
D: Clock duty cycle (D = 0.5 to 1)
L: Frame length (L = 9 to 12)
F: Absolute value of clock frequency error

In equation (1), if F (absolute value of clock frequency error) = 0 and D (clock duty cycle) = 0.5, the receive margin is 46.875% as given by equation (2) below.

When D = 0.5 and F = 0,

$M = \{0.5 - 1/(2 \times 16)\} \times 100\% = 46.875\%$.....Equation (2)

This value is theoretical. In actual system designs a margin of from 20 to 30 percent should be allowed.

7. Relationship between bit RDRF and reading RDR

While SCI3 is receiving, it checks the RDRF flag. When a frame of data has been received, if the RDRF flag is cleared to 0, data receiving ends normally. If RDRF is set to 1, an overrun error occurs.

RDRF is automatically cleared to 0 when the contents of RDR are read. If RDR is read more than once, the second and later reads will be performed with RDRF cleared to 0. While RDRF is 0, if RDR is read when reception of the next frame is just ending, data from the next frame may be read. This is illustrated in figure 10-2-22.

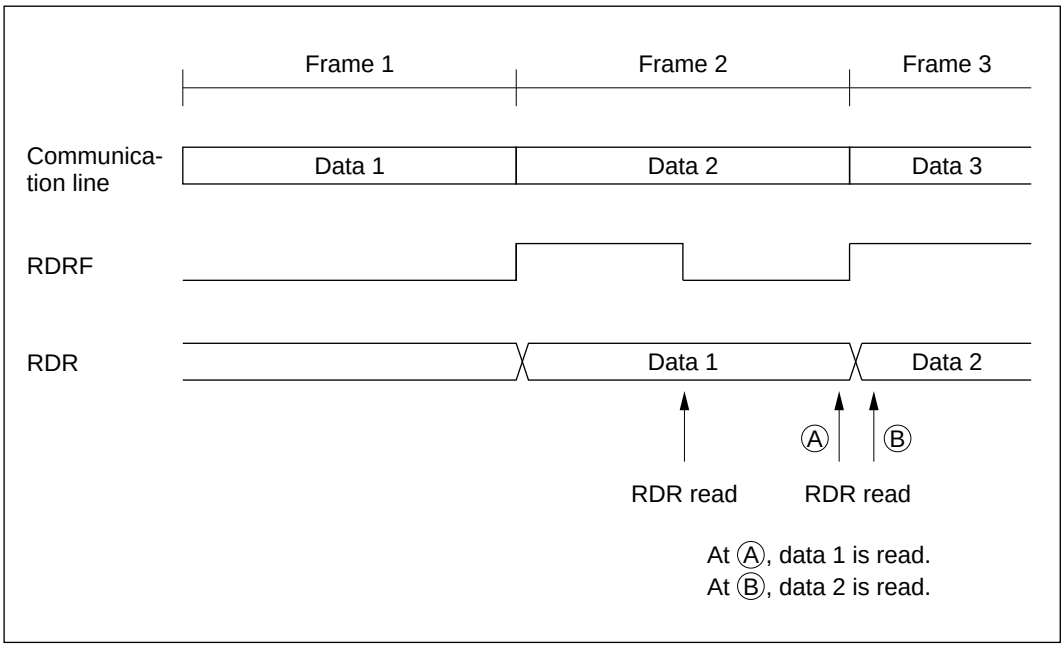


Figure 10-2-22 Relationship between Data and RDR Read Timing

To avoid the situation described above, after RDRF is confirmed to be 1, RDR should only be read once and should not be read twice or more. When the same data must be read more than once, the data read the first time should be copied to RAM, for example, and the copied data should be used. An alternative is to read RDR but leave a safe margin of time before reception of the next frame is completed. In synchronous mode, all reads of RDR should be completed before bit 7 is received. In asynchronous mode, all reads of RDR should be completed before the stop bit is received.

8. Notes on switching functions of pin SCK_3

If pin SCK_3 is switched from clock output in SCI3 synchronous mode to usage as an input port, when the function of pin SCK_3 is switched over, a brief low output will occur. The duration of this output is one-half of a system clock period ($0.5 \times ø$). This brief low output can be avoided by the following methods.

- When switching pin SCK_3 from clock output to non-output

  To terminate transmitting and receiving, use one instruction to clear the TE and RE bits to 0 in SCR3, and simultaneously set bit CKE1 to 1 and clear bit CKE0 to 0. Leave the COM bit in SMR set to 1. This will make the pin unavailable for general input/output port usage. To avoid ambiguous voltage input at pin SCK_3, pull the connected line up to V_{CC} through a resistor, or provide output from another device.

- When switching pin SCK_3 from clock output to input/output port usage

  Terminate receiving as follows:

  1. First use one instruction to clear the TE and RE bits to 0 in SCR3, and simultaneously set bit CKE1 to 1 and clear bit CKE0 to 0.
  2. Next clear the COM bit to 0 in SMR.
  3. Finally, clear bits CKE1 and CKE0 both to 0 in SCR3.

  Avoid ambiguous voltage input at pin SCK_3, as above.

9. Switching TXD function

If pin TXD is used as a data output pin by SCI3 in synchronous mode and is then switched to a general input/output pin (a pin with a different function), the pin outputs a high level signal for one system clock (ø) cycle immediately after it is switched.

10.3 I²C Bus Interface

10.3.1 Overview

The I²C bus interface conforms to and provides a subset of the Philips I²C bus (inter-IC bus) interface functions. The register configuration that controls the I²C bus differs partly from the Philips configuration, however.

The I²C bus interface uses only one data line (SDA) and one clock line (SCL) to transfer data, so it can save board and connector space. Figure 10-3-1 shows typical I²C bus interface connections.

1. Features

- Conforms to Philips I²C bus interface
- Start and stop conditions generated automatically
- Selectable acknowledge output level when receiving
- Auto-loading of acknowledge bit when transmitting
- Selection of eight internal clocks (in master mode)
- Selection of acknowledgement mode*1, or serial mode*1 without acknowledge bit
- Wait function: a wait can be inserted in acknowledgement mode by holding the SCL pin low after a data transfer, before acknowledgement of the transfer.
- Three interrupt sources
  - — Data transfer end
  - — In slave receive mode*2: slave address matched, or general call address received
  - — In master transmit mode*2: bus arbitration lost

Notes: 1. See the description of bits 5 and 4 in 10.3.2, I²C Bus Control Register (ICCR).
2. See the description of bit 3 in 10.3.2, I²C Bus Control Register (ICCR).

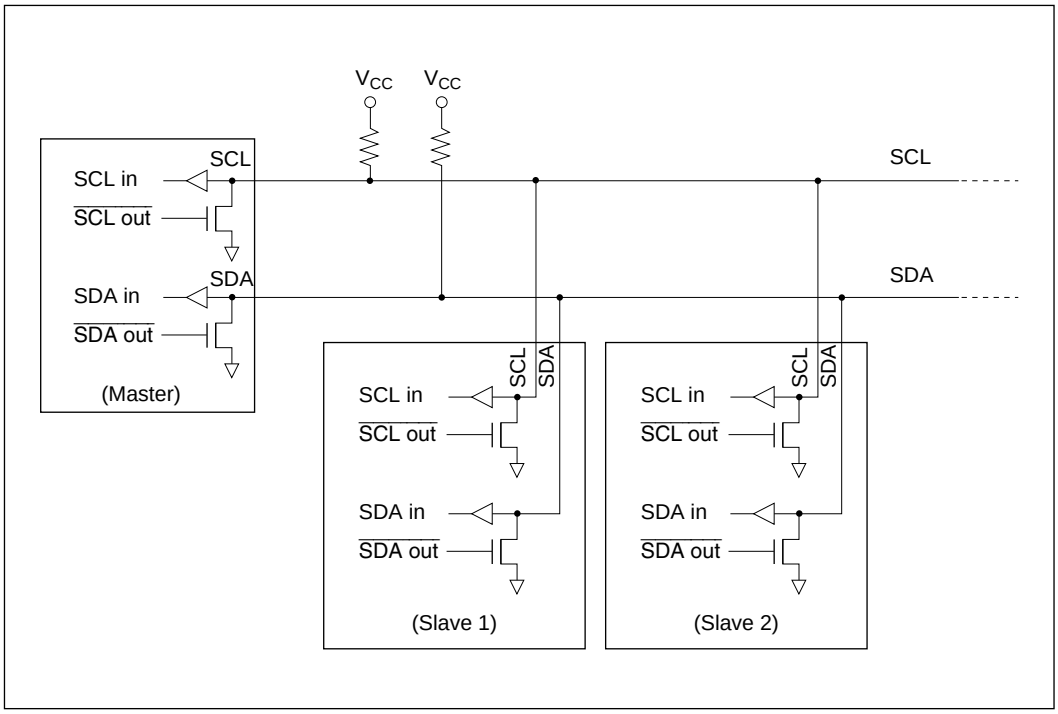


Figure 10-3-1 I²C Bus Interface Connections (Example)

2. Block diagram

Figure 10-3-2 shows a block diagram of the I2C bus interface.

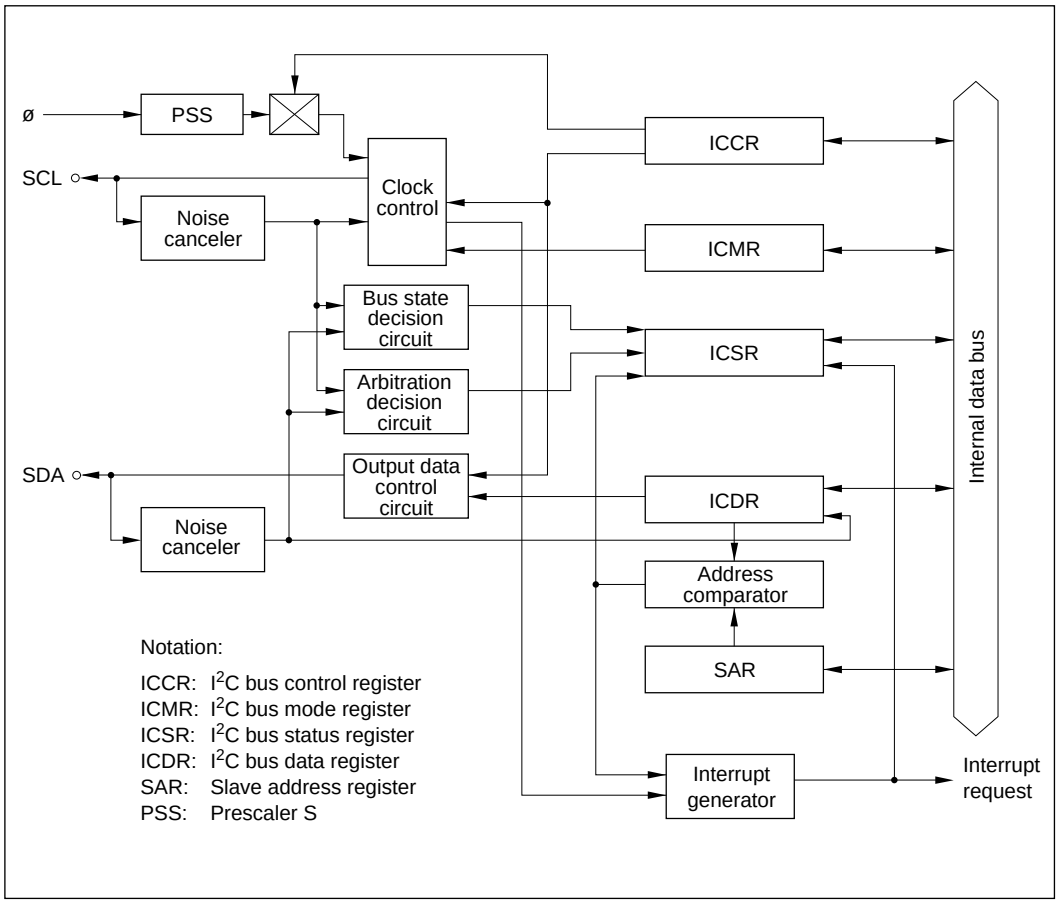


Figure 10-3-2 Block Diagram of I2C Bus Interface

3. Input/output pins

Table 10-3-1 summarizes the input/output pins used by the I²C bus interface.

Table 10-3-1 I²C Bus Interface Pins

Channel	Name	Abbrev.	I/O	Function
1	Serial clock 1	SCL_1	Input/output	Serial clock input/output
	Serial data 1	SDA_1	Input/output	Serial data input/output
2	Serial clock 2	SCL_2	Input/output	Serial clock input/output
	Serial data 2	SDA_2	Input/output	Serial data input/output

4. Register configuration

Table 10-3-2 summarizes the registers of the I²C bus interface.

Table 10-3-2 Register Configuration

Channe	Name	Abbrev.	R/W	Initial Value	Address
1	I²C bus control register 1	ICCR1	R/W	H'00	H'FF80
	I²C bus status register 1	ICSR1	R/W	H'30	H'FF81
	I²C bus data register 1	ICDR1	R/W	—	H'FF82
	I²C bus mode register 1	ICMR1	R/W	H'38	H'FF83*
	Slave address register 1	SAR1	R/W	H'00	H'FF83*
2	I²C bus control register 2	ICCR2	R/W	H'00	H'FF84
	I²C bus status register 2	ICSR2	R/W	H'30	H'FF85
	I²C bus data register 2	ICDR2	R/W	—	H'FF86
	I²C bus mode register 2	ICMR2	R/W	H'38	H'FF87*
	Slave address register 2	SAR2	R/W	H'00	H'FF87*

Note: * The register that can be written or read depends on the ICE bit in the I²C bus control register. The slave address register can be accessed when ICE = 0. The I²C bus mode register can be accessed when ICE = 1.

10.3.2 Register Descriptions

1. I²C bus data register (ICDR)

Bit	7	6	5	4	3	2	1	0
	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0
Initial value	—	—	—	—	—	—	—	—
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ICDR is an 8-bit read/write register that is used as a transmit data register when transmitting and a receive data register when receiving. Transmitting is started by writing data in ICDR. Receiving is started by reading data from ICDR.

ICDR is also used as a shift register, so it must not be written or read until data has been completely transmitted or received. If this register is written or read while a data transfer is in progress, the data contents are not guaranteed.

The ICDR value upon reset is not fixed.

2. Slave address register (SAR)

Bit	7	6	5	4	3	2	1	0
	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SAR is an 8-bit read/write register that stores the slave address and selects the communication format. When the chip is in slave mode (and the addressing format is selected), if the upper 7 bits of SAR match the upper 7 bits of the first byte received after a start condition, the chip operates as the slave device specified by the master device. SAR is assigned to the same address as ICMR. SAR can be written and read only when the ICE bit is cleared to 0 in ICCR.

SAR is initialized to H'00 upon reset.

Bits 7 to 1: Slave address (SVA6 to SVA0)

Set a unique address in bits SVA6 to SVA0, differing from the addresses of other slave devices connected to the I²C bus.

Bit 0: Format select (FS)

Bit 0 selects whether to use the addressing format or non-addressing format in slave mode. The addressing format is used to recognize slave addresses.

Bit 0 FS	Description	
0	Addressing format, slave addresses recognized	(initial value)
1	Non-addressing format	

3. I^2C bus mode register (ICMR)

Bit	7	6	5	4	3	2	1	0
	MLS	WAIT	—	—	—	BC2	BC1	BC0
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

ICMR is an 8-bit read/write register that selects whether the MSB or LSB is transferred first, performs wait control, and selects the transfer bit count. ICMR is assigned to the same address as SAR. ICMR can be written and read only when the ICE bit is set to 1 in ICCR.

ICMR is initialized to H'38 upon reset.

Bit 7: MSB-first/LSB-first select (MLS)

Bit 7 selects whether data is transferred MSB-first or LSB-first.

Bit 7 MLS	Description	
0	MSB-first	(initial value)
1	LSB-first	

Bit 6: Wait insertion bit (WAIT)

Bit 6 selects whether to insert a wait between the transfer of data and the acknowledge bit, in acknowledgement mode. When WAIT is set to 1, after the fall of the clock for the final data bit, a wait state begins (with SCL staying at the low level). When bit IRIC is cleared in ICSR, the wait ends and the acknowledge bit is transferred. If WAIT is cleared to 0, data and acknowledge bits are transferred consecutively with no wait inserted.

Bit 6 WAIT	Description	
0	Data and acknowledge transferred consecutively	(initial value)
1	Wait inserted between data and acknowledge	

Bits 5 to 3: Reserved bits

Bits 5 to 3 are reserved; they are always read as 1, and cannot be modified.

Bits 2 to 0: Bit counter (BC2 to BC0)

BC2 to BC0 specify the number of bits to be transferred next. When the ACK bit is cleared to 0 in ICCR (acknowledgement mode), the data is transferred with one additional acknowledge bit. BC2 to BC0 settings should be made during an interval between transfer frames. If BC2 to BC0 are set to a value other than 000, the setting should be made while the SCL line is low.

The bit counter is initialized to 000 by a reset and when a start condition is detected. The value returns to 000 at the end of a data transfer, including the acknowledge.

Bit 2 BC2	Bit 1 BC1	Bit 0 BC0	Bits/Frame Serial Mode	 Acknowledgement Mode	
0	0	0	8	9	(initial value)
		1	1	2	
	1	0	2	3	
		1	3	4	
1	0	0	4	5	
		1	5	6	
	1	0	6	7	
		1	7	8	

4. I²C bus control register (ICCR)

Bit	7	6	5	4	3	2	1	0
	ICE	IEIC	MST	TRS	ACK	CKS2	CKS1	CKS0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ICCR is an 8-bit read/write register that enables or disables the I²C bus interface, enables or disables interrupts, and selects master or slave mode, transmit or receive, acknowledgement or serial mode, and the clock frequency.

ICCR is initialized to H'00 upon reset.

Bit 7: I²C bus interface enable (ICE)

Bit 7 selects whether or not to use the I²C bus interface. When ICE is set to 1, the corresponding port pins become SCL and SDA input/output pins and transfer operations are enabled. When ICE is cleared to 0, the port pins are placed in the high-impedance state and the interface module is disabled.

The SAR register can be accessed when ICE is 0. The ICMR register can be accessed when ICE is 1.

Bit 7 ICE	**Description**	
0	Interface module disabled, with SCL and SDA signals in high-impedance state	(initial value)
1	Interface module enabled for transfer operations	

Bit 6: I²C bus interface interrupt enable (IEIC)

Bit 6 enables or disables interrupts from the I²C bus interface to the CPU.

Bit 6 IEIC	**Description**	
0	Interrupts disabled	(initial value)
1	Interrupts enabled	

Bit 5: Master/slave select (MST)

Bit 4: Transmit/receive select (TRS)

MST selects whether the I^2C bus interface operates in master mode or slave mode.

TRS selects whether the I^2C bus interface operates in transmit mode or receive mode.

In master mode, when arbitration is lost, MST and TRS are both reset by hardware, causing a transition to slave receive mode. In slave receive mode with the addressing format (FS = 0), hardware automatically selects transmit or receive mode according to the R/W bit in the first byte after a start condition.

MST and TRS select the operating mode as follows.

Bit 5 MST	Bit 4 TRS	Operating Mode	
0	0	Slave receive mode	(initial value)
	1	Slave transmit mode	
1	0	Master receive mode	
	1	Master transmit mode	

Bit 3: Acknowledgement mode select (ACK)

Bit 3 selects acknowledgement mode or serial mode. In acknowledgement mode (ACK = 0), data is transferred in frames consisting of the number of data bits selected by BC2 to BC0 in ICMR, plus an extra acknowledge bit. In serial mode (ACK = 1), the number of data bits selected by BC2 to BC0 in ICMR is transferred as one frame.

Bit 3 ACK	Description	
0	Acknowledgement mode	(initial value)
1	Serial mode	

Bits 2 to 0: Serial clock select (CKS2 to CKS0)

Bits 2 to 0 select the serial clock frequency in master mode. They should be set according to the required transfer rate.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Clock	Transfer Rate ø = 2 MHZ	ø = 4 MHz	ø = 5 MHz
0	0	0	ø/14	143 kHz	286 kHz	357 kHz
		1	ø/20	100 kHz	200 kHz	250 kHz
	1	0	ø/24	83.3 kHz	167 kHz	208 kHz
		1	ø/32	62.5 kHz	125 kHz	156 kHz
1	0	0	ø/40	50.0 kHz	100 kHz	125 kHz
		1	ø/50	40.0 kHz	80.0 kHz	100 kHz
	1	0	ø/56	35.7 kHz	71.4 kHz	89.3 kHz
		1	ø/64	31.3 kHz	62.5 kHz	78.1 kHz

5. I^2C bus status register (ICSR)

Bit	7	6	5	4	3	2	1	0
	BBSY	IRIC	SCP	—	AL	AAS	ADZ	ACKB
Initial value	0	0	1	1	0	0	0	0
Read/Write	R/W	R/(W)*	W	—	R/(W)*	R/(W)*	R/(W)*	R/W

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

ICSR is an 8-bit read/write register with flags that indicate the status of the I^2C bus interface. It is also used for issuing start and stop conditions, and recognizing and controlling acknowledge bits.

ICSR is initialized to H'30 by a reset.

Bit 7: Bus busy (BBSY)

Bit 7 can be read to check whether the I2C bus (SCL and SDA) is busy or free. In master mode this bit is also used in issuing start and stop conditions.

A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0.

To issue a start condition, use a MOV instruction to write 1 in BBSY and 0 in SCP. A retransmitted start condition is issued in the same way. To issue a stop condition, use a MOV instruction to write 0 in BBSY and 0 in SCP. It is not possible to write to BBSY in slave mode.

Bit 7 BBSY	Description	
0	Bus is free	(initial value)
	Clearing condition: Cleared when a stop condition is detected	
1	Bus is busy	
	Setting condition: Set when a start condition is detected	

Bit 6: I2C bus interface interrupt request flag (IRIC)

Bit 6 indicates that the I2C bus interface has issued an interrupt request to the CPU. IRIC is set to 1 at the end of a data transfer, when a slave address or general call address is detected in slave receive mode, and when bus arbitration is lost in master transmit mode. IRIC is set at different timings depending on the ACK bit in ICCR and WAIT bit in ICMR. See 10.3.3 (6), IRIC Set Timing and SCL Control.

IRIC is cleared by reading IRIC after it has been set to 1, then writing 0 in IRIC.

Bit 6 IRIC	Description	
0	Waiting for transfer, or transfer in progress	(initial value)
	Clearing condition: Cleared by reading IRIC = 1, then writing 0 in IRIC	
1	Interrupt requested	
	Setting conditions:	
	Master mode • End of data transfer • When bus arbitration is lost	
	Slave mode (when FS = 0) • When the slave address is matched, and whenever a data transfer ends after that, until a retransmitted start condition or a stop condition is detected • When a general call address is detected, and whenever a data transfer ends after that, until a retransmitted start condition or a stop condition is detected	
	Slave mode (when FS = 1) • End of data transfer	

Bit 5: Start condition/stop condition prohibit (SCP)

Bit 5 controls the issuing of start and stop conditions in master mode. To issue a start condition, write 1 in BBSY and 0 in SCP. A retransmitted start condition is issued in the same way. To issue a stop condition, write 0 in BBSY and 0 in SCP. This bit always reads 1. Written data is not stored.

Bit 5 SCP	Description		
0	Write	Issues a start or stop condition, in combination with BBSY	
1	Read	Always results in 1	(initial value)
	Write	Ignored	

Bit 4: Reserved bit

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

Bit 3: Arbitration lost (AL)

Bit 3 indicates that arbitration was lost in master mode. The I^2C bus interface monitors the bus. When two or more master devices attempt to seize the bus at nearly the same time, if the I^2C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master. At the same time, it sets the IRIC bit in ICSR to generate an interrupt request.

AL is cleared by reading AL after it has been set to 1, then writing 0 in AL. In addition, AL is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

Bit 3 AL	Description	
0	Bus arbitration won	(initial value)
	Clearing conditions: • When ICDR data is written (transmit mode) or read (receive mode) • After reading AL = 1, cleared by writing 0 to AL	
1	Arbitration lost	
	Setting conditions: • When the internal SDA signal and bus line disagree at the rise of SCL in master transmit mode • When the internal SCL is high at the fall of SCL in master transmit mode	

Bit 2: Slave address recognition flag (AAS)

When the addressing format is selected (FS = 0) in slave receive mode, this flag is set to 1 if the first byte following a start condition matches bits SVA6 to SVA0 in SAR, or if the general call address (H'00) is detected.

AAS is cleared by reading AAS after it has been set to 1, then writing 0 in AAS. In addition, AAS is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

Bit 2 AAS	Description	
0	Slave address or general call address not recognized	(initial value)
	Clearing conditions: • When ICDR data is written (transmit mode) or read (receive mode) • After reading AAS = 1, cleared by writing 0 to AAS	
1	Slave address or general call address recognized	
	Setting conditions: • When the slave address or general call address is detected in slave receive mode	

Bit 1: General call address recognition flag (ADZ)

When the addressing format is selected (FS = 0) in slave receive mode, this flag is set to 1 if the first byte following a start condition is the general call address (H'00).

ADZ is cleared by reading ADZ after it has been set to 1, then writing 0 in ADZ. In addition, ADZ is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

Bit 1 ADZ	Description	
0	General call address not recognized	(initial value)
	Clearing conditions: • When ICDR data is written (transmit mode) or read (receive mode) • After reading ADZ = 1, cleared by writing 0 to ADZ	
1	General call address recognized	
	Setting condition: When the general call address is detected in slave receive mode	

Bit 0: Acknowledge bit (ACKB)

Bit 0 stores acknowledge data in acknowledgement mode. In transmit mode, after the receiving device receives data, it returns acknowledge data, and this data is loaded into ACKB. In receive mode, after data has been received, the acknowledge data set in this bit is sent to the transmitting device.

When this bit is read, if TRS = 1, the value loaded from the bus line is read. If TRS = 0, the value set by internal software is read.

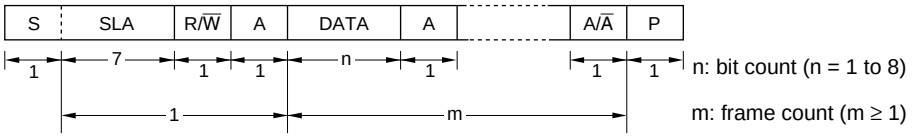
Bit 0 ACKB	Description	
0	Receive mode: 0 is output at acknowledge output timing Transmit mode: indicates that the receiving device has acknowledged the data	(initial value)
1	Receive mode: 1 is output at acknowledge output timing Transmit mode: indicates that the receiving device has not acknowledged the data	

10.3.3 Operation

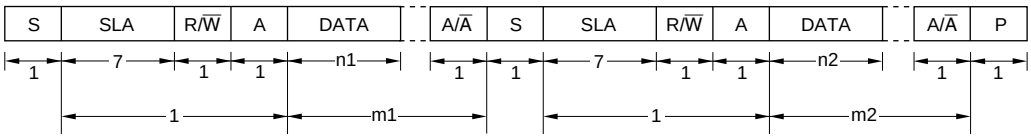
1. I^2C bus data format

The I^2C bus interface has three data formats: two addressing formats, shown as (a) and (b) in figure 10-3-3 (1), and a non-addressing format, shown as (c) in figure 10-3-3 (2). The first byte following a start condition always consists of 8 bits. Figure 10-3-4 shows the I^2C bus timing.

(a) Addressing format (FS = 0)



(b) Addressing format (retransmit start condition, FS = 0)



Notation:

S: Start condition. The master device drives SDA from high to low while SCL is high.

SLA: Slave address, by which the master device selects a slave device.

R/$\overline{W}$: Indicates the direction of data transfer: from the slave device to the master device when R/$\overline{W}$ is 1, or from the master device to the slave device when R/$\overline{W}$ is 0.

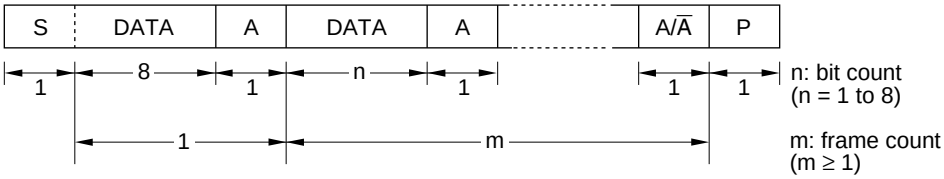
A: Acknowledge. The receiving device (the slave in master transmit mode, or the master in master receive mode) drives SDA low to acknowledge a transfer. If transfers need not be acknowledged, set the ACK bit to 1 in ICCR to keep the interface from generating the acknowledge signal and its clock pulse.

DATA: Transferred data. The bit length is set by bits BC2 to BC0 in ICMR. The MSB-first or LSB-first format is selected by bit MLS in ICMR.

P: Stop condition. The master device drives SDA from low to high while SCL is high.

Figure 10-3-3 (1) I^2C Bus Data Formats (Addressing Formats)

(c) Non-addressing format (FS = 1)



Notation:

S: Start condition. The master device drives SDA from high to low while SCL is high.

A: Acknowledge. The receiving device (the slave in master transmit mode, or the master in master receive mode) drives SDA low to acknowledge a transfer. If transfers need not be acknowledged, set the ACK bit to 1 in ICCR to keep the interface from generating the acknowledge signal and its clock pulse.

DATA: Transferred data. The bit length is set by bits BC2 to BC0 in ICMR. The MSB-first or LSB-first format is selected by bit MLS in ICMR.

P: Stop condition. The master device drives SDA from low to high while SCL is high.

Figure 10-3-3 (2) I²C Bus Data Format (Non-Addressing Format)

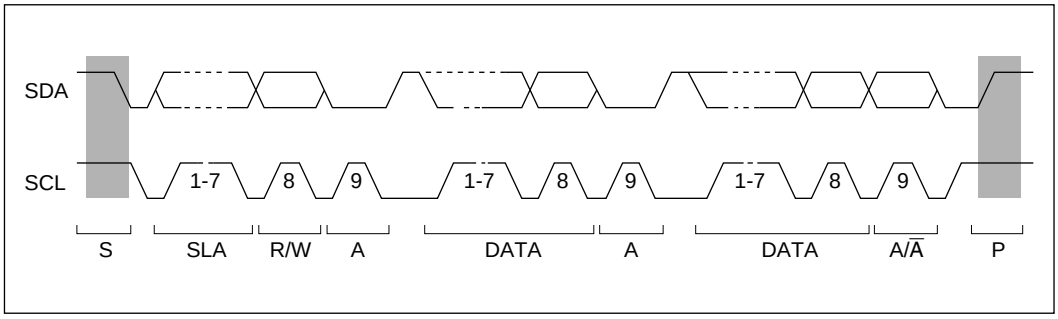


Figure 10-3-4 I²C Bus Timing

2. Master transmit operation

In master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The transmit procedure and operations in master transmit mode are described below.

1. Set bits MLS and WAIT in ICMR and bits ACK and CKS2 to CKS0 in ICCR according to the operating mode. Set bit ICE in ICCR to 1.

2. Read BBSY in ICSR, check that the bus is free, then set MST and TRS to 1 in ICCR to select master transmit mode. After that, write 1 in BBSY and 0 in SCP. This generates a start condition by causing a high-to-low transition of SDA while SCL is high.

3. Write data in ICDR. The master device outputs the written data together with a sequence of transmit clock pulses at the timing shown in figure 10-3-5. If FS is 0 in SAR, the first byte following the start condition contains a 7-bit slave address and indicates the transmit/receive direction. The selected slave device (the device with the matching slave address) drives SDA low at the ninth transmit clock pulse to acknowledge the data.

4. When 1 byte of data has been transmitted, IRIC is set to 1 in ICSR at the rise of the ninth transmit clock pulse. If IEIC is set to 1 in ICCR, a CPU interrupt is requested. After one frame has been transferred, SCL is automatically brought to the low level in synchronization with the internal clock and held low.

5. Software clears IRIC to 0 in ICSR.

6. To continue transmitting, write the next transmit data in ICDR. Transmission of the next byte will begin in synchronization with the internal clock.

Steps 4 to 6 can be repeated to transmit data continuously. To end the transmission, write 0 in BBSY and 0 in SCP in ICSR. This generates a stop condition by causing a low-to-high transition of SDA while SCL is high.

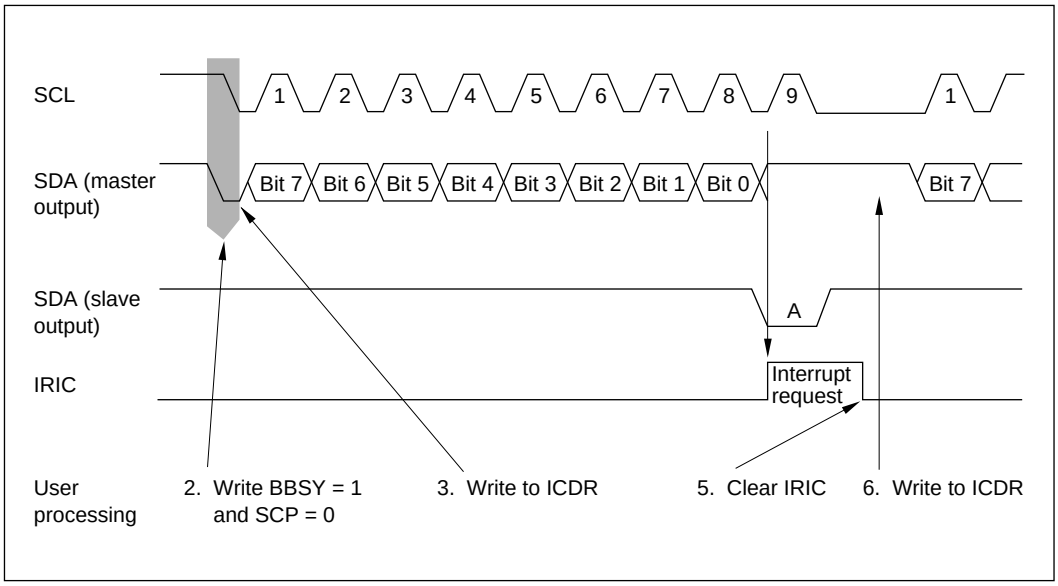


Figure 10-3-5 Timing in Master Transmit Mode (MLS = WAIT = ACK = 0)

3. Master receive operation

In master receive mode, the master device outputs the receive clock, receives data, and returns an acknowledge signal. The slave device transmits the data. The receive procedure and operations in master receive mode are described below. See also figure 10-3-6.

1. Clear TRS to 0 in ICCR to switch from transmit mode to receive mode.

2. Read ICDR to start receiving. When ICDR is read, a receive clock is output in synchronization with the internal clock, and data is received. At the ninth clock pulse the master device drives SDA low to acknowledge the data.

3. When 1 byte of data has been received, IRIC is set to 1 in ICSR at the rise of the ninth receive clock pulse. If IEIC is set to 1 in ICCR, a CPU interrupt is requested. After one frame has been transferred, SCL is automatically brought to the low level in synchronization with the internal clock and held low.

4. Software clears IRIC to 0 in ICSR.

5. When ICDR is read, receiving of the next data starts in synchronization with the internal clock.

Steps 3 to 5 can be repeated to receive data continuously. To stop receiving, set TRS to 1, read ICDR, then write 0 in BBSY and 0 in SCP in ICSR. This generates a stop condition by causing a low-to-high transition of SDA while SCL is high. If it is not necessary to acknowledge each bye of data, set ACKB to 1 in ICSR before receiving starts.

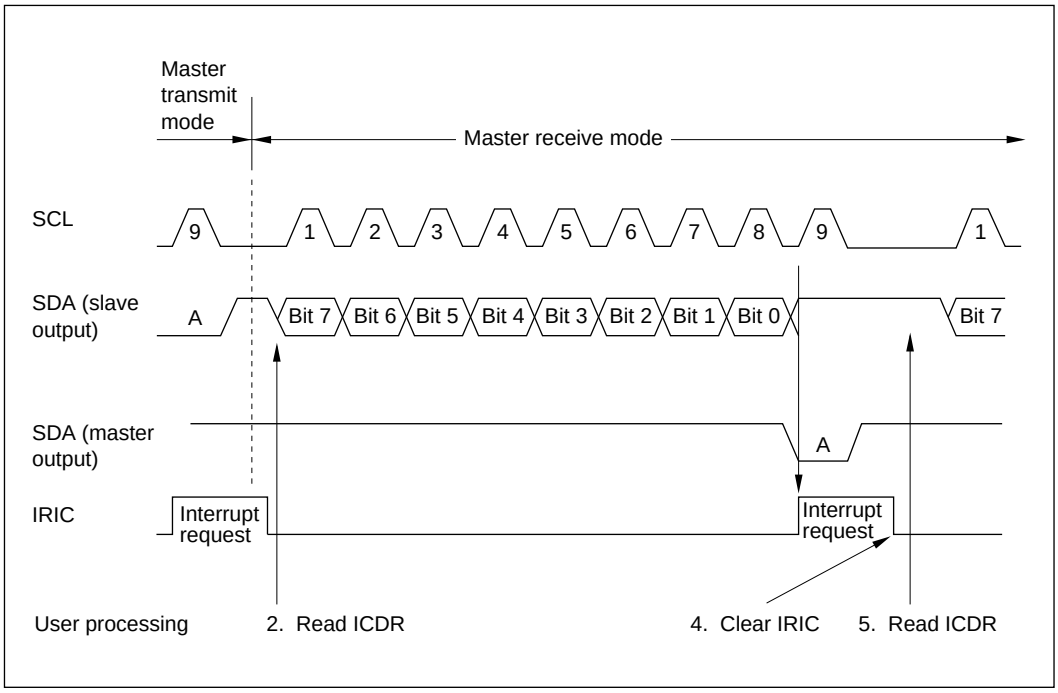


Figure 10-3-6 Timing in Master Receive Mode (MLS = WAIT = ACK = 0)

4. Slave transmit operation

In slave transmit mode, the slave device outputs the transmit data, and the master device outputs a receive clock and returns an acknowledge signal. The transmit procedure and operations in slave transmit mode are described below.

1. Set bits MLS and WAIT in ICMR and bits MST, TRS, ACK, and CKS2 to CKS0 in ICCR according to the operating mode. Set bit ICE in ICCR to 1.

2. After the slave device detects a start condition, if the first byte matches its slave address, at the ninth clock pulse the slave device drives SDA low to acknowledge the transfer. At the same time, IRIC is set to 1 in ICSR, generating an interrupt. If the eighth data bit (R/$\overline{W}$) is 1, the TRS bit is set to 1 in ICCR, automatically causing a transition to slave transmit mode. The slave device holds SCL low from the fall of the transmit clock until data is written in ICDR.

3. Software clears IRIC to 0 in ICSR.

4. Write data in ICDR. The slave device outputs the written data serially in step with the clock output by the master device, with the timing shown in figure 10-3-7.

5. When 1 byte of data has been transmitted, at the rise of the ninth transmit clock pulse IRIC is set to 1 in ICSR. If IEIC is set to 1 in ICCR, a CPU interrupt is requested. The slave device holds SCL low from the fall of the transmit clock until data is written in ICDR. The master device drives SDA low at the ninth clock pulse to acknowledge the data. The acknowledge signal is stored in ACKB in ICSR, and can be used to check whether the transfer was carried out normally.

6. Software clears IRIC to 0 in ICSR.

7. To continue transmitting, write the next transmit data in ICDR.

Steps 5 to 7 can be repeated to transmit continuously. To end the transmission, write H'FF in ICDR. When a stop condition is detected (a low-to-high transition of SDA while SCL is high), BBSY will be cleared to 0 in ICSR.

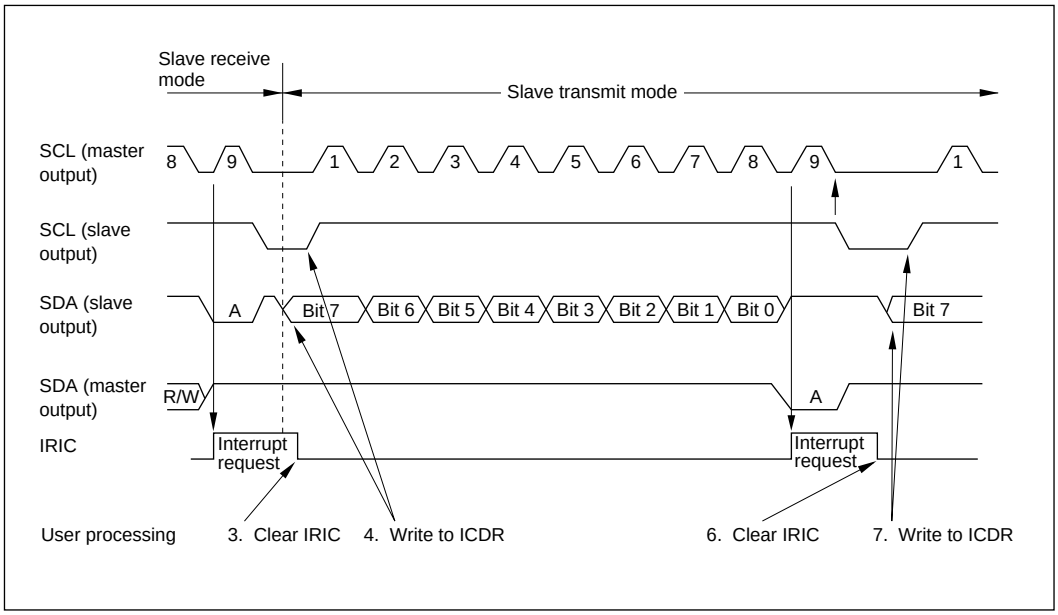


Figure 10-3-7 Timing in Slave Transmit Mode (MLS = WAIT = ACK = 0)

5. Slave receive operation

In slave receive mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The receive procedure and operations in slave receive mode are described below. See also figure 10-3-8.

1. Set bits MLS and WAIT in ICMR and bits MST, TRS, and ACK in ICCR according to the operating mode. Set bit ICE in ICCR to 1.

2. A start condition output by the master device sets BBSY to 1 in ICSR.

3. After the slave device detects the start condition, if the first byte matches its slave address, at the ninth clock pulse the slave device drives SDA low to acknowledge the transfer. At the same time, IRIC is set to 1 in ICSR. If IEIC is 1 in ICCR, a CPU interrupt is requested. The slave device holds SCL low from the fall of the receive clock until it has read the data in ICDR.

4. Software clears IRIC to 0 in ICSR.

5. When ICDR is read, receiving of the next data starts.

Steps 4 and 5 can be repeated to receive data continuously. When a stop condition is detected (a low-to-high transition of SDA while SCL is high), BBSY is cleared to 0 in ICSR.

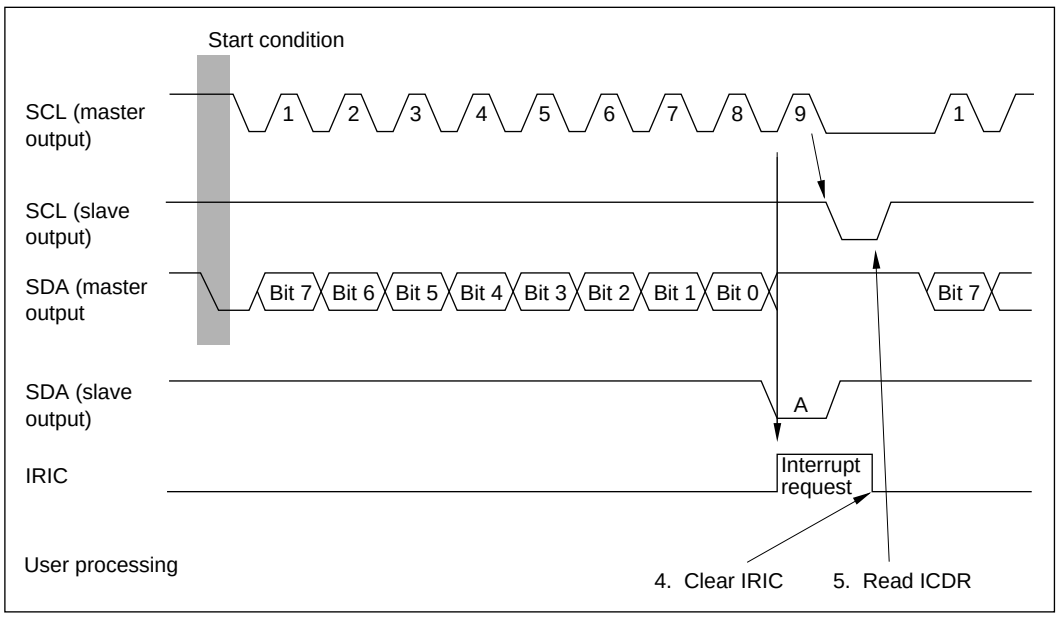


Figure 10-3-8 Timing in Slave Receive Mode (MLS = WAIT = ACK = 0)

6. IRIC set timing and SCL control

The interrupt request flag (IRIC) is set at different times depending on the WAIT bit in ICMR and ACK bit in ICCR. SCL is automatically held low after one frame has been transferred; this timing is synchronized with the internal clock. Figure 10-3-9 shows the IRIC set timing and SCL control.

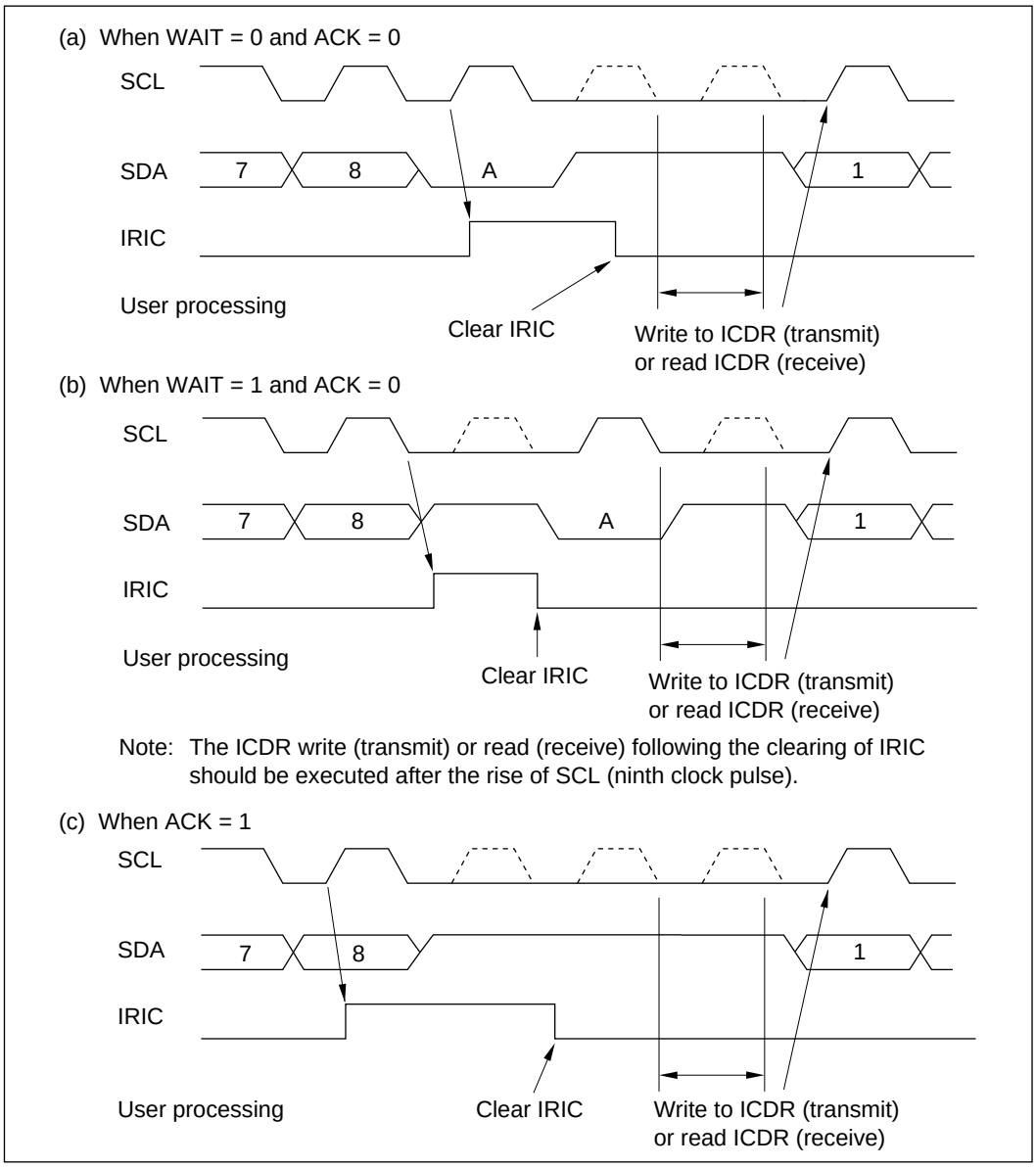


Figure 10-3-9 IRIC Set Timing and SCL Control

7. Noise canceler

The logic levels at the SCL and SDA pins are routed through noise cancelers before being latched internally. Figure 10-3-10 shows a block diagram of the noise canceler circuit.

The noise canceler consists of two cascaded latches and a match detector. The SCL (or SDA) input signal is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.

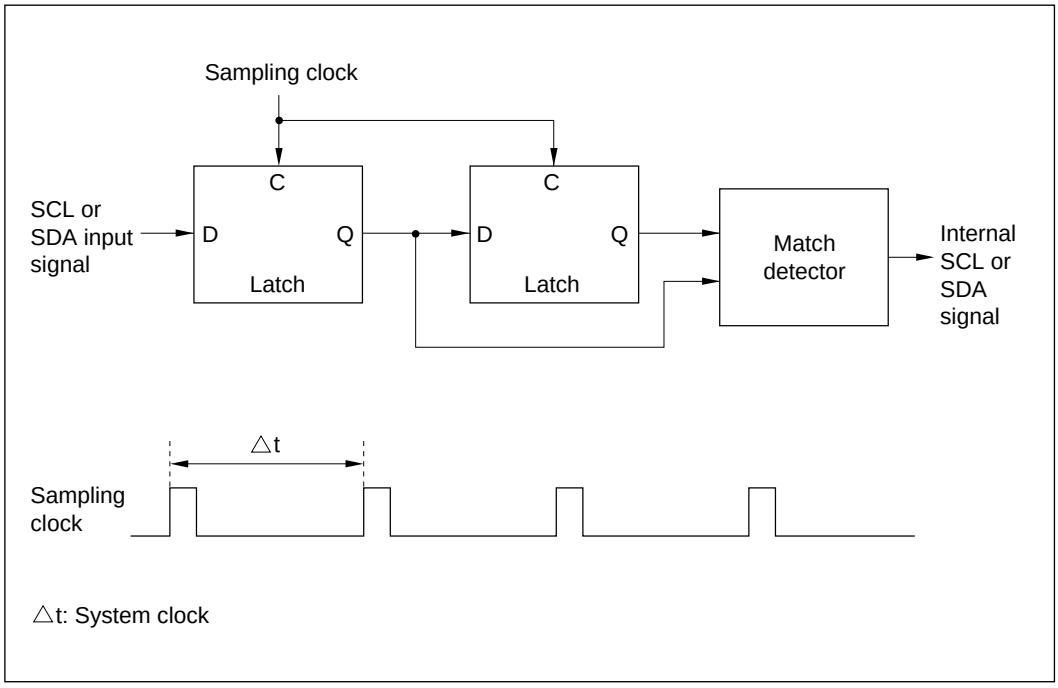


Figure 10-3-10 Block Diagram of Noise Canceler

8. Sample flowcharts

Figures 10-3-11 to 10-3-14 show typical flowcharts for using the I²C bus interface in each mode.

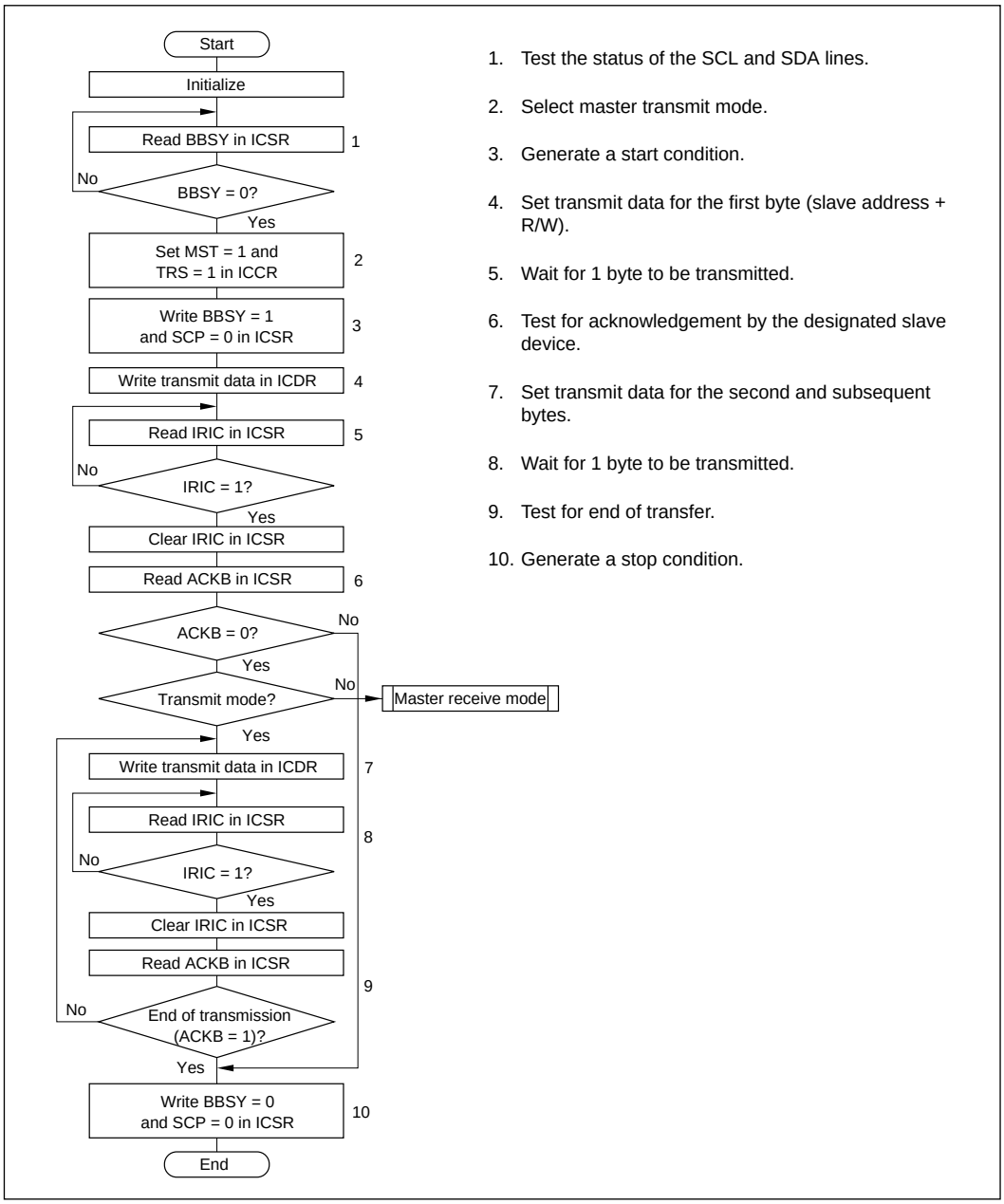
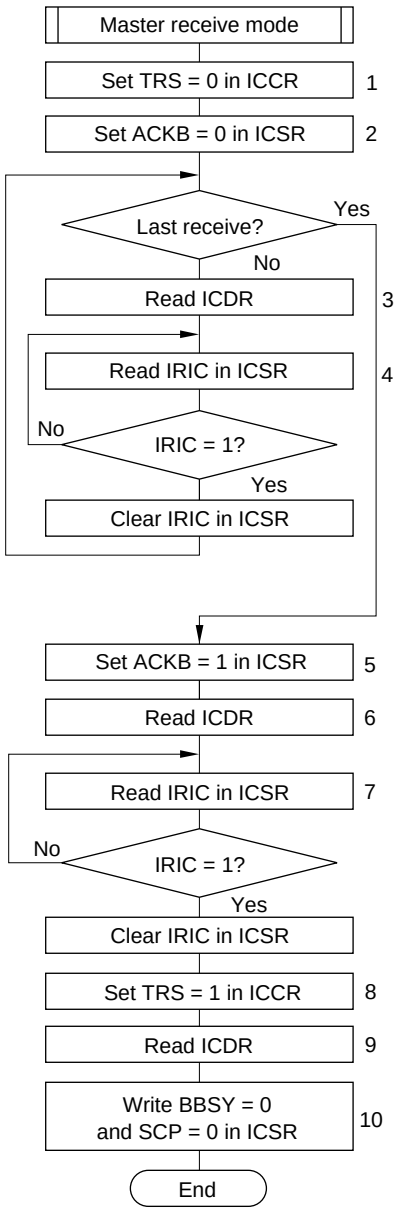


Figure 10-3-11 Flowchart for Master Transmit Mode (Example)



1. Select receive mode.
2. Set acknowledgement data.
3. Start receiving. The first read is a dummy read.
4. Wait for 1 byte to be received.
5. Set acknowledgement data for the last receive.
6. Start the last receive.
7. Wait for 1 byte to be received.
8. Select transmit mode.
9. Read the last receive data (if ICDR is read without selecting transmit mode, receive operations will resume).
10. Generate a stop condition.

Figure 10-3-12 Flowchart for Master Receive Mode (Example)

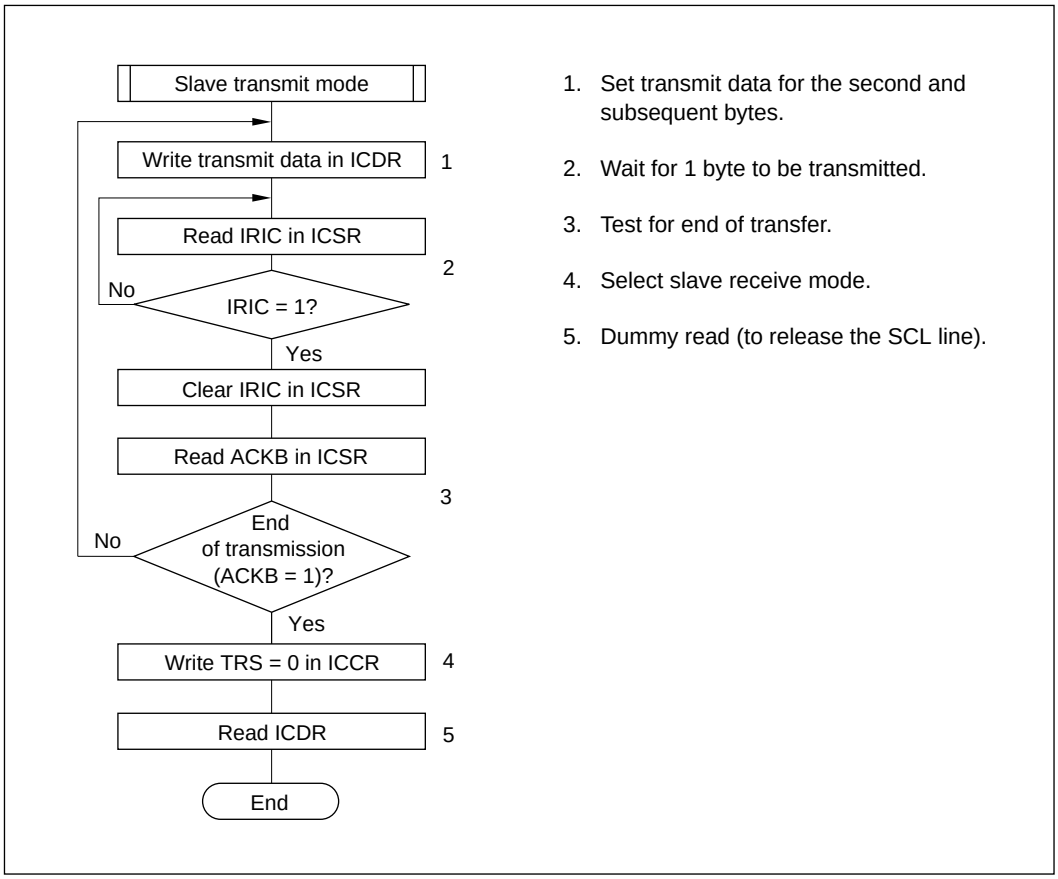


Figure 10-3-13 Flowchart for Slave Transmit Mode (Example)

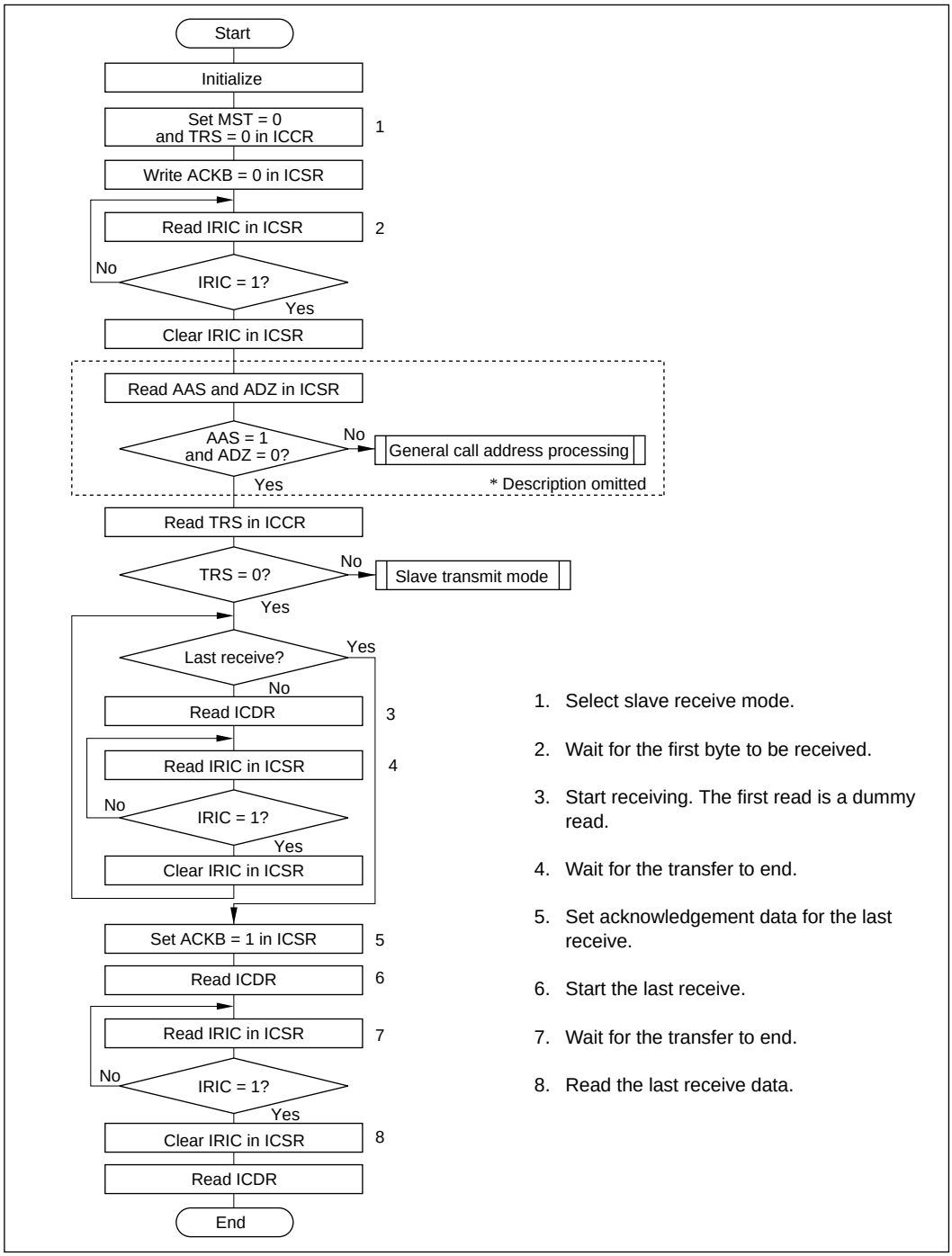


Figure 10-3-14 Flowchart for Slave Receive Mode (Example)

10.3.4 Application Notes

- In master mode, if an instruction to generate a start condition is immediately followed by an instruction to generate a stop condition, neither condition will be output correctly. To output consecutive start and stop conditions, after issuing the instruction that generates the start condition, read the relevant ports, check that SCL and SDA are both low, then issue the instruction that generates the stop condition.

- Either of the following two conditions will start the next transfer. Pay attention to these conditions when reading or writing to ICDR.

  1. Write access to ICDR when ICE = 1 and TRS = 1
  2. Read access to ICDR when ICE = 1 and TRS = 0

- In master mode, the I^2C bus interface synchronizes each bit by monitoring the SCL line. If the SCL rise time t_{sr} (the time required for a transition from the low level to V_{IH}) exceeds 2.5 t_{cyc}, the high period of SCL will be extended. The SCL rise time depends on the pull-up resistance and load capacitance of the SCL line. To operate at the desired transfer rate, design the pull-up resistor and load capacitance so that t_{sr} will be within 2.5 t_{cyc}.

Section 11 8-Bit PWM

11.1 Overview

The H8/3947 Series has on-chip an 8-bit pulse-width modulator (PWM) with eight channels. The PWM can be used as a D/A converter by connecting a low-pass filter.

11.1.1 Features

Features of the 8-bit PWM are given below.

- PWM output can be set independently on eight channels
- Selection of 409.6 µs or 819.2 µs period (with 5-MHz clock)
- Master-slave PWM register configuration in all eight channels
- Medium-voltage output pins with NMOS open-drain output circuits

11.1.2 Block Diagram

Figure 11-1 shows a block diagram of the PWM.

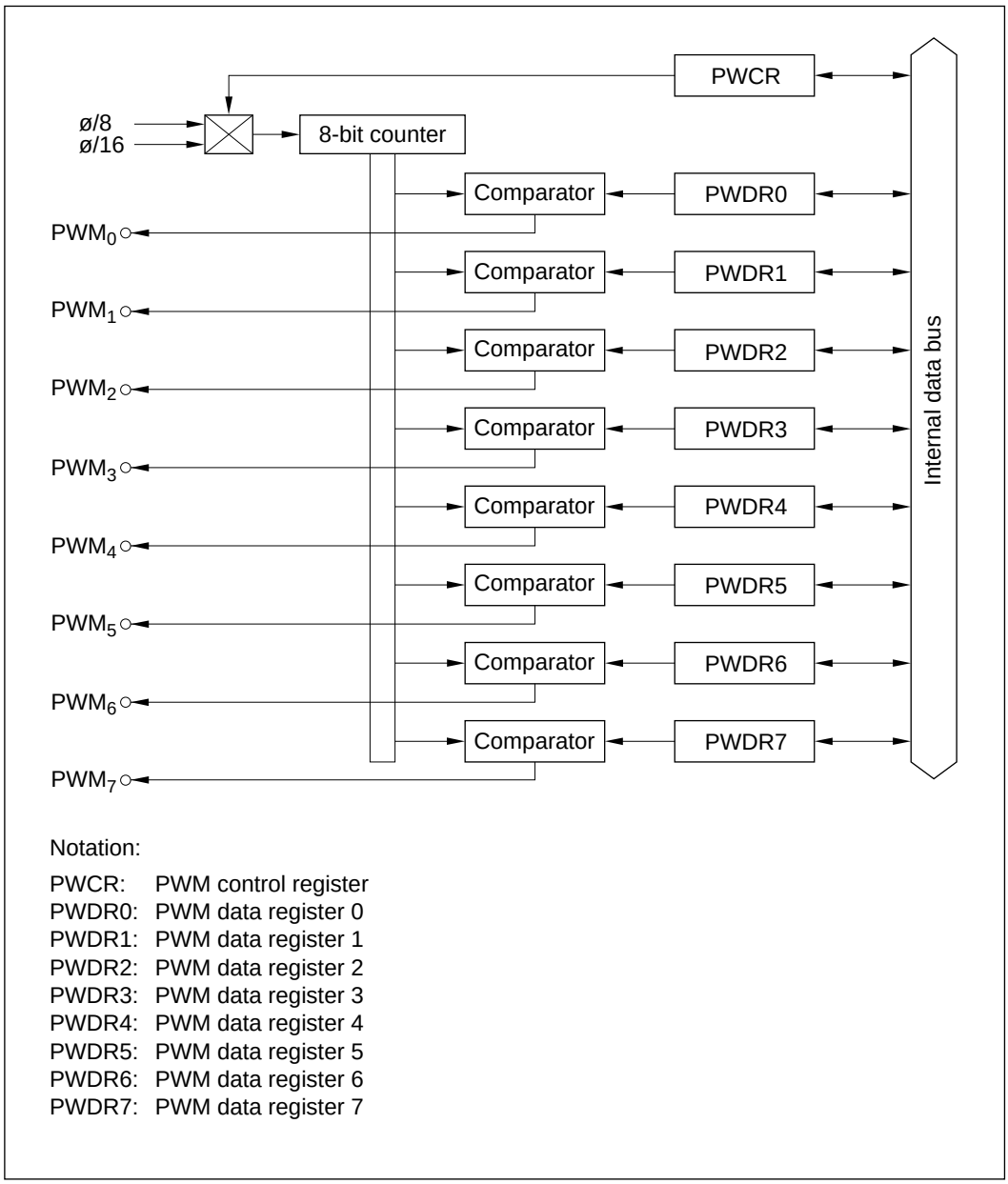


Figure 11-1 Block Diagram of the 8-Bit PWM

11.1.3 Pin Configuration

Table 11-1 shows the PWM pin configuration.

Table 11-1 Pin Configuration

Name	Abbrev.	I/O	Function
PWM_0 output pin	PWM_0	Output	PWM pulse output 0
PWM_1 output pin	PWM_1	Output	PWM pulse output 1
PWM_2 output pin	PWM_2	Output	PWM pulse output 2
PWM_3 output pin	PWM_3	Output	PWM pulse output 3
PWM_4 output pin	PWM_4	Output	PWM pulse output 4
PWM_5 output pin	PWM_5	Output	PWM pulse output 5
PWM_6 output pin	PWM_6	Output	PWM pulse output 6
PWM_7 output pin	PWM_7	Output	PWM pulse output 7

11.1.4 Register Configuration

Table 11-2 shows the PWM register configuration.

Table 11-2 Register Configuration

Name	Abbrev.	R/W	Initial Value	Address
PWM control register	PWCR	R/W	H'FE	H'FF90
PWM data register 0	PWDR0	R/W	H'00	H'FF91
PWM data register 1	PWDR1	R/W	H'00	H'FF92
PWM data register 2	PWDR2	R/W	H'00	H'FF93
PWM data register 3	PWDR3	R/W	H'00	H'FF94
PWM data register 4	PWDR4	R/W	H'00	H'FF95
PWM data register 5	PWDR5	R/W	H'00	H'FF96
PWM data register 6	PWDR6	R/W	H'00	H'FF97
PWM data register 7	PWDR7	R/W	H'00	H'FF98

11.2 Register Descriptions

11.2.1 PWM Control Register (PWCR)

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

PWCR is an 8-bit read/write register that selects the input clock for the 8-bit counter. 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 (CKS)

Bit 0 selects the input clock for the 8-bit counter.

Bit 0 CKS	Description	
0	Input clock is ø/16, with cycle time of 4096/ø	(initial value)
1	Input clock is ø/8, with cycle time of 2048/ø	

11.2.2 PWM Data Register 0 (PWDR0)

Bit	7	6	5	4	3	2	1	0
	$PWDR0_7$	$PWDR0_6$	$PWDR0_5$	$PWDR0_4$	$PWDR0_3$	$PWDR0_2$	$PWDR0_1$	$PWDR0_0$
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

PWDR0 is an 8-bit read/write register that controls the high width of the PWM_0 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR0 is initialized to H'00.

11.2.3 PWM Data Register 1 (PWDR1)

Bit	7	6	5	4	3	2	1	0
	$PWDR1_7$	$PWDR1_6$	$PWDR1_5$	$PWDR1_4$	$PWDR1_3$	$PWDR1_2$	$PWDR1_1$	$PWDR1_0$
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

PWDR1 is an 8-bit read/write register that controls the high width of the PWM_1 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR1 is initialized to H'00.

11.2.4 PWM Data Register 2 (PWDR2)

Bit	7	6	5	4	3	2	1	0
	$PWDR2_7$	$PWDR2_6$	$PWDR2_5$	$PWDR2_4$	$PWDR2_3$	$PWDR2_2$	$PWDR2_1$	$PWDR2_0$
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

PWDR2 is an 8-bit read/write register that controls the high width of the PWM_2 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR2 is initialized to H'00.

11.2.5 PWM Data Register 3 (PWDR3)

Bit	7	6	5	4	3	2	1	0
	$PWDR3_7$	$PWDR3_6$	$PWDR3_5$	$PWDR3_4$	$PWDR3_3$	$PWDR3_2$	$PWDR3_1$	$PWDR3_0$
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

PWDR3 is an 8-bit read/write register that controls the high width of the PWM_3 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR3 is initialized to H'00.

11.2.6 PWM Data Register 4 (PWDR4)

Bit	7	6	5	4	3	2	1	0
	$PWDR4_7$	$PWDR4_6$	$PWDR4_5$	$PWDR4_4$	$PWDR4_3$	$PWDR4_2$	$PWDR4_1$	$PWDR4_0$
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

PWDR4 is an 8-bit read/write register that controls the high width of the PWM_4 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR4 is initialized to H'00.

11.2.7 PWM Data Register 5 (PWDR5)

Bit	7	6	5	4	3	2	1	0
	$PWDR5_7$	$PWDR5_6$	$PWDR5_5$	$PWDR5_4$	$PWDR5_3$	$PWDR5_2$	$PWDR5_1$	$PWDR5_0$
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

PWDR5 is an 8-bit read/write register that controls the high width of the PWM_5 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR5 is initialized to H'00.

11.2.8 PWM Data Register 6 (PWDR6)

Bit	7	6	5	4	3	2	1	0
	$PWDR6_7$	$PWDR6_6$	$PWDR6_5$	$PWDR6_4$	$PWDR6_3$	$PWDR6_2$	$PWDR6_1$	$PWDR6_0$
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

PWDR6 is an 8-bit read/write register that controls the high width of the PWM_6 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR6 is initialized to H'00.

11.2.9 PWM Data Register 7 (PWDR7)

Bit	7	6	5	4	3	2	1	0
	$PWDR7_7$	$PWDR7_6$	$PWDR7_5$	$PWDR7_4$	$PWDR7_3$	$PWDR7_2$	$PWDR7_1$	$PWDR7_0$
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

PWDR7 is an 8-bit read/write register that controls the high width of the PWM_7 output. This register is actually a master-slave register pair, with data being loaded from the master to the slave register when the counter overflows (see figure 11-2). Read access is to the slave register.

Upon reset, PWDR7 is initialized to H'00.

11.3 Operation

1. Master-slave register configuration

PWDR0 to PWDR7 have a master-slave configuration, each consisting of a master register and slave register as shown in figure 11-2.

- Write access is carried out from the CPU to the master register.
- Read access is carried out from the slave register.
- When the counter overflows from H'FF to H'00, the master register contents are copied to the slave register.

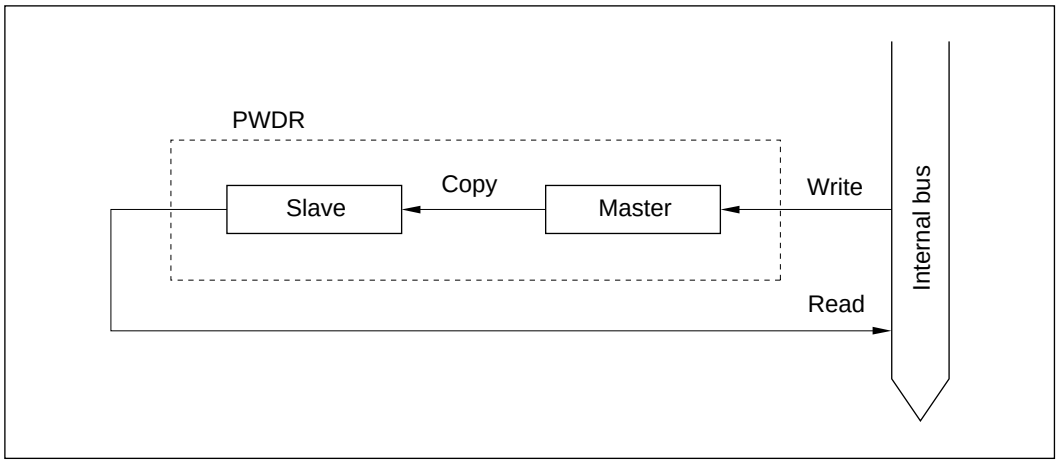


Figure 11-2 PWM Data Register Configuration

2. Update timing of PWDR data and PWM output

Figure 11-3 shows the update timing of the PWDR data and PWM output. When data is written in a PWM data register, first the master register is updated. When the counter overflows, the master register contents are transferred to the slave register. The PWM output goes high when the counter overflows, and goes low when the counter value matches the value in the slave register.

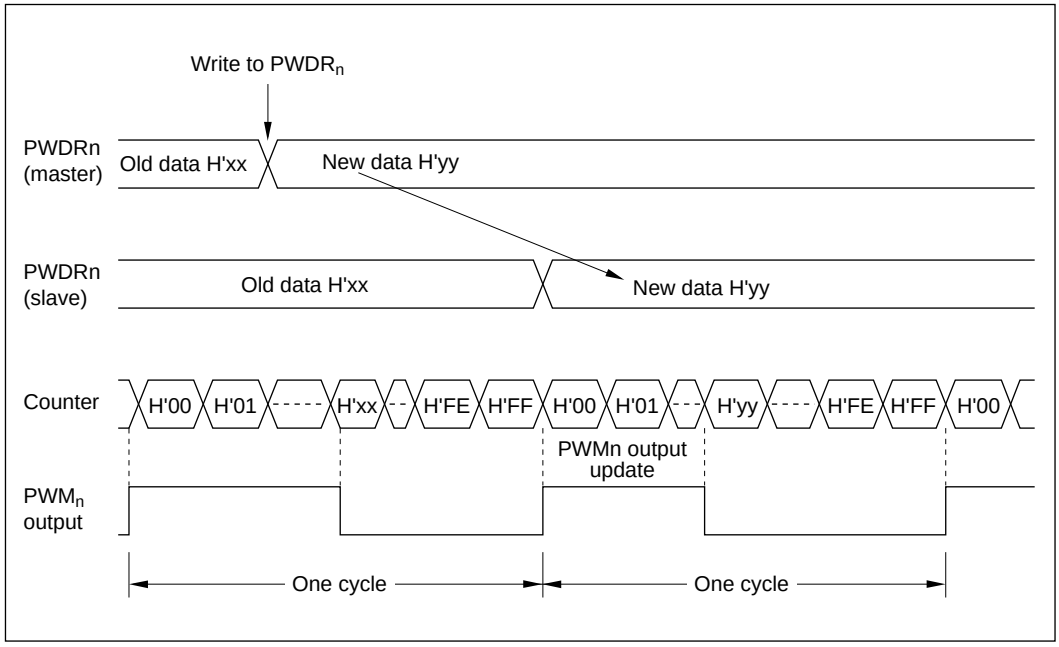


Figure 11-3 Update Timing of PWDRn and PWM$_n$ Output

3. PWM output waveform

The PWM consists of an 8-bit free-running counter, PWM data registers, and comparators. Figure 11-4 shows the PWM output waveform. The high width and cycle length can be expressed as follows.

High width = (value in PWM data register) × tø
Cycle length = 256 × tø

The quantity tø is the value selected by CKS in PWCR: 16/ø when CKS = 0, or 8/ø when CKS = 1.

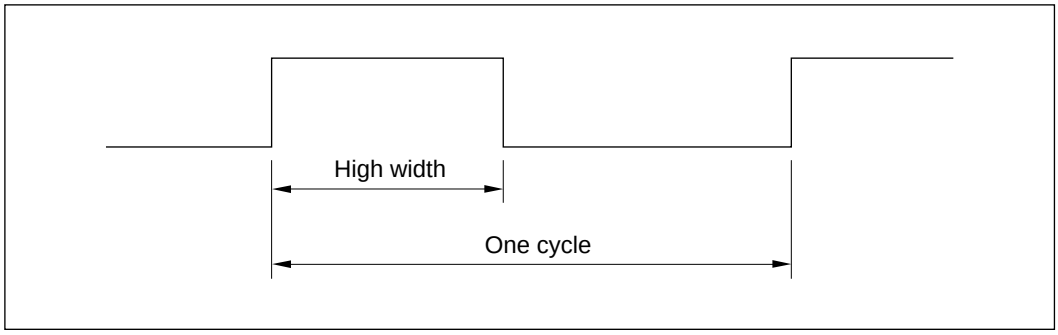


Figure 11-4 PWM_n Output Waveform

4. PWM usage

A typical register setup procedure for using the PWM is given next.

1. Set bit PWMn to 1 in port mode register 3 (PMR3) to select the PWM_n output pin function (n = 7 to 0).
2. Select the input clock for the 8-bit counter with the CKS bit in PWCR.
3. Set the output waveform data in the PWM data register (PWDRn). The output waveform will be updated when the counter overflows.

The output waveform can be controlled by repeatedly executing step 3 to update the waveform.

5. Operation modes

Table 11-3 indicates the PWM operation modes.

Table 11-3 PWM Operation Modes

Operation Mode	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby
PWM_n output	High impedance	Functions	Functions	Low	Low	Low	High impedance
PWCR	Reset	Functions	Retained	Retained	Retained	Retained	Retained
PWDRn	Reset	Functions	Retained	Retained	Retained	Retained	Retained

(n = 7 to 0)

11.4 Application Notes

When using the PWM, note the following points.

1. Changing the clock select bit (CKS) during PWM operation

   If CKS is changed during PWM operation when PWDRn ≠ H'00, the PWMn output waveform will be incorrect. To change CKS during operation, follow the procedure given below.

   1. Before changing CKS during operation, write H'00 in PWDRn in all channels.
   2. After writing to the last PWDRn, read this PWDRn and check that its value is H'00.
   3. If PWDRn ≠ H'00, repeat step 2.
   4. If PWDRn = H'00, change CKS in PWCR.
   5. Write waveform data in the PWDRn registers.

2. PWM_n output is held at the low level in subactive mode, subsleep mode, watch mode, and standby mode. PWM_n output starts immediately on exit from these modes.

3. Holding PWM_n output high

   To hold the PWM_n output at the high level, set $P3_n$ to 1 in port data register 3 (PDR3), set $PCR3_n$ to 1 in port control register 3 (PCR3), and set PWMn to 0 in port mode register 3 (PMR3).

   Note that when H'FF is set in PWDRn, the output is high for a duration of $255 \times t\text{ø}$ and low for a duration of tø.

4. The PWM_n output cycle is the same in all eight channels. It is not possible to set the cycle length independently in each channel. Low-to-high transitions of the PWM_n outputs occur simultaneously in all eight channels.

5. If the PWM is operating in active mode or sleep mode and the chip is reset or the mode is changed to watch mode, subactive mode, subsleep mode, or standby mode, hazard pulses may be output from the PWM_n pins.

Section 12 A/D Converter

12.1 Overview

The H8/3947 Series includes on-chip a resistance-ladder-based successive-approximation analog-to-digital converter, and can convert up to 12 channels of analog input.

12.1.1 Features

The A/D converter has the following features.

- 8-bit resolution
- 12 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.

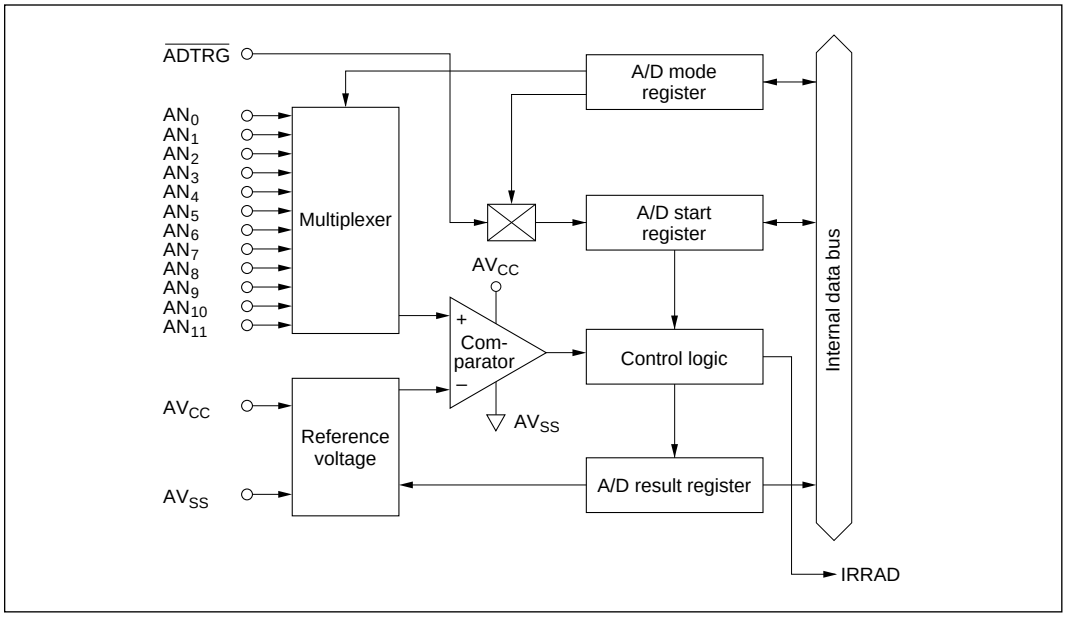


Figure 12-1 Block Diagram of the A/D Converter

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 pin	AV_{CC}	Input	Power supply and reference voltage of analog part
Analog ground pin	AV_{SS}	Input	Ground and reference voltage of analog part
Analog input pin 0	AN_0	Input	Analog input channel 0
Analog input pin 1	AN_1	Input	Analog input channel 1
Analog input pin 2	AN_2	Input	Analog input channel 2
Analog input pin 3	AN_3	Input	Analog input channel 3
Analog input pin 4	AN_4	Input	Analog input channel 4
Analog input pin 5	AN_5	Input	Analog input channel 5
Analog input pin 6	AN_6	Input	Analog input channel 6
Analog input pin 7	AN_7	Input	Analog input channel 7
Analog input pin 8	AN_8	Input	Analog input channel 8
Analog input pin 9	AN_9	Input	Analog input channel 9
Analog input pin 10	AN_{10}	Input	Analog input channel 10
Analog input pin 11	AN_{11}	Input	Analog input channel 11
External trigger input pin	$\overline{\text{ADTRG}}$	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	—	—	—	—	—	—	—	—
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.

Bit 7 CKS	**Conversion Period**	**Conversion Time ø = 2 MHz**	**Conversion Time ø = 5 MHz**
0	62/ø (initial value)	31 µs	12.4 µs
1	31/ø	15.5 µs	*

Note: * Operation is not guaranteed if the conversion time is less than 12.4 µs. Set 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 IEG4 of the IRQ edge select register (IEGR). See 3.3.2 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)
0	1	0	0	AN_0	
0	1	0	1	AN_1	
0	1	1	0	AN_2	
0	1	1	1	AN_3	
1	0	0	0	AN_4	
1	0	0	1	AN_5	
1	0	1	0	AN_6	
1	0	1	1	AN_7	
1	1	0	0	AN_8	
1	1	0	1	AN_9	
1	1	1	0	AN_{10}	
1	1	1	1	AN_{11}	

Note: * Don’t care

12.2.3 A/D Start Register (ADSR)

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

The A/D start register (ADSR) is an 8-bit read/write register for starting and stopping A/D conversion.

A/D conversion is started by writing 1 to the A/D start flag (ADSF) or by input of the designated edge of the external trigger signal, which also sets ADSF to 1. When conversion is complete, the converted data is set in the A/D result register (ADRR), and at the same time ADSF is cleared to 0.

Bit 7: A/D start flag (ADSF)

Bit 7 controls and indicates the start and end of A/D conversion.

Bit 7 ADSF	Description	
0	[Read access] Indicates the completion of A/D conversion.	(initial value)
	[Write access] Stops A/D conversion.	
1	[Read access] Indicates A/D conversion in progress.	
	[Write access] Starts A/D conversion.	

Bits 6 to 0: Reserved bits

Bits 6 to 0 are reserved; they are always read as 1, and cannot be modified.

12.3 Operation

12.3.1 A/D Conversion Operation

The A/D converter operates by successive approximations, and yields its conversion result as 8-bit data.

A/D conversion begins when software sets the A/D start flag (bit ADSF) to 1. Bit ADSF keeps a value of 1 during A/D conversion, and is cleared to 0 automatically when conversion is complete.

The completion of conversion also sets bit IRRAD in interrupt request register 2 (IRR2) to 1. An A/D conversion end interrupt is requested if bit IENAD in interrupt enable register 2 (IENR2) is set to 1.

If the conversion time or input channel needs to be changed in the A/D mode register (AMR) during A/D conversion, bit ADSF should first be cleared to 0, stopping the conversion operation, in order to avoid malfunction.

12.3.2 Start of A/D Conversion by External Trigger Input

The A/D converter can be made to start A/D conversion by input of an external trigger signal. External trigger input is enabled at pin $\overline{\text{ADTRG}}$ when bit IRQ4 in port mode register 2 (PMR2) is set to 1, and bit TRGE in AMR is set to 1. Then when the input signal edge designated in bit IEG4 of the IRQ edge select register (IEGR) is detected at pin $\overline{\text{ADTRG}}$, bit ADSF in ADSR will be set to 1, starting A/D conversion.

Figure 12-2 shows the timing.

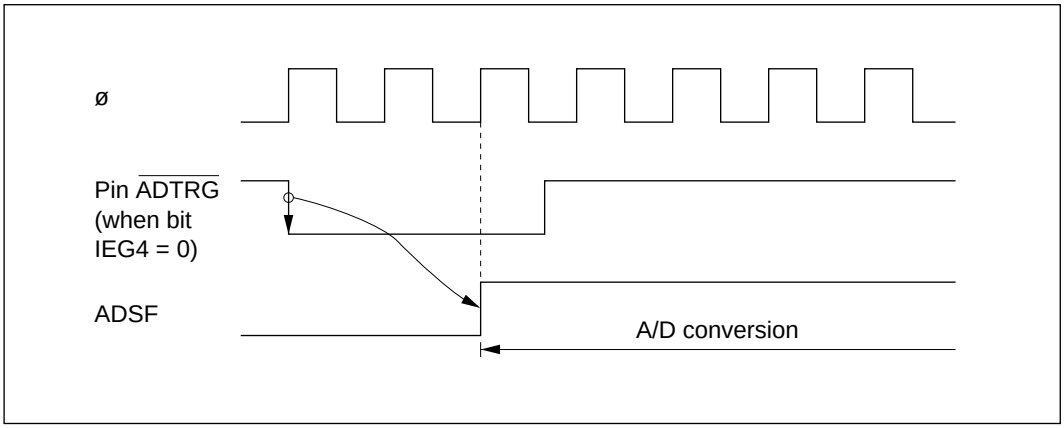


Figure 12-2 External Trigger Input Timing

12.4 Interrupts

When A/D conversion ends (ADSF changes from 1 to 0), bit IRRAD in interrupt request register 2 (IRR2) is set to 1.

A/D conversion end interrupts can be enabled or disabled by means of bit IENAD in interrupt enable register 2 (IENR2).

For further details see 3.3, Interrupts.

12.5 Typical Use

An example of how the A/D converter can be used is given below, using channel 1 (pin AN1) as the analog input channel. Figure 12-3 shows the operation timing.

- 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.
- 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.
- Bit IENAD = 1, so an A/D conversion end interrupt is requested.
- The A/D interrupt handling routine starts.
- The A/D conversion result is read and processed.
- 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.

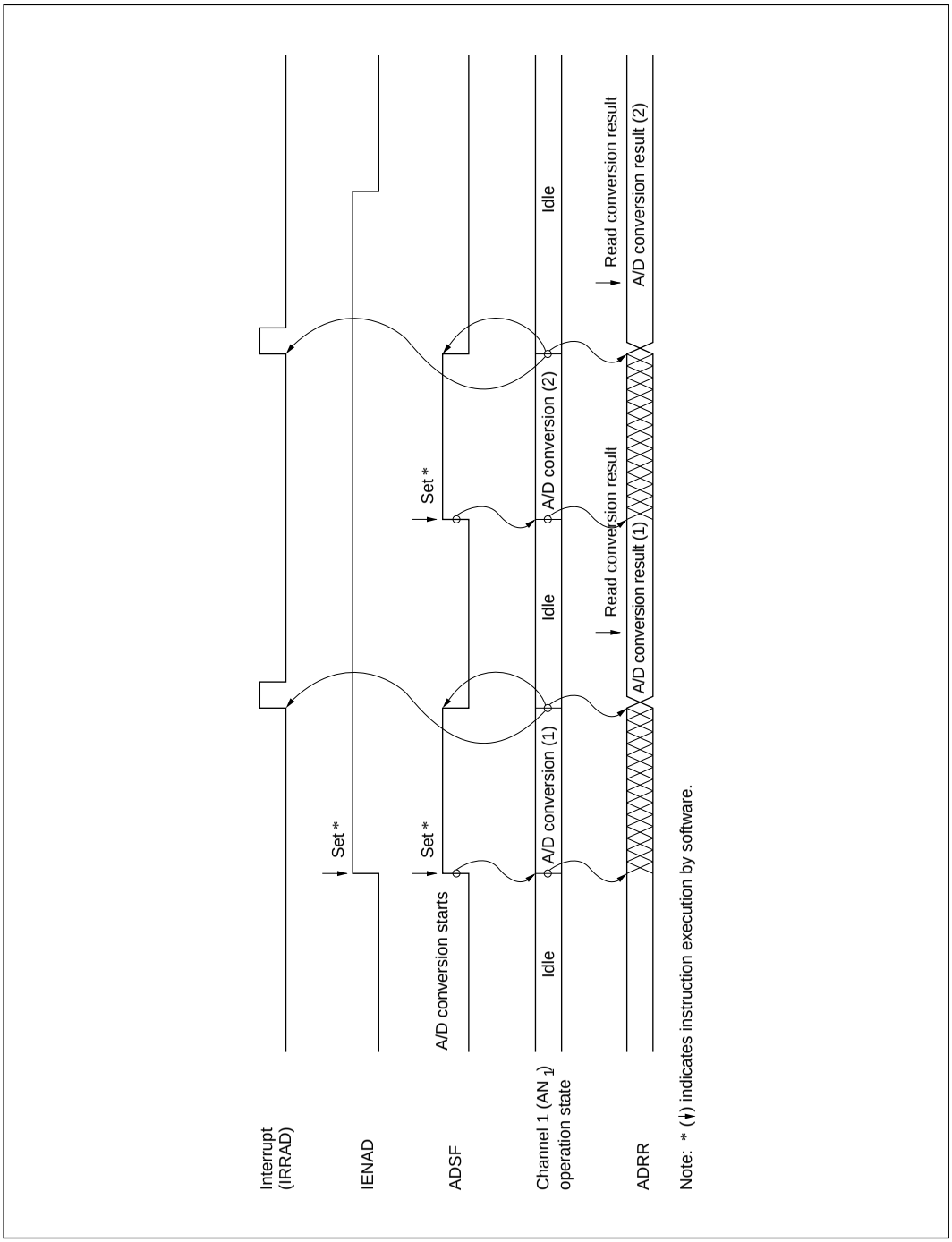


Figure 12-3 Typical A/D Converter Operation Timing

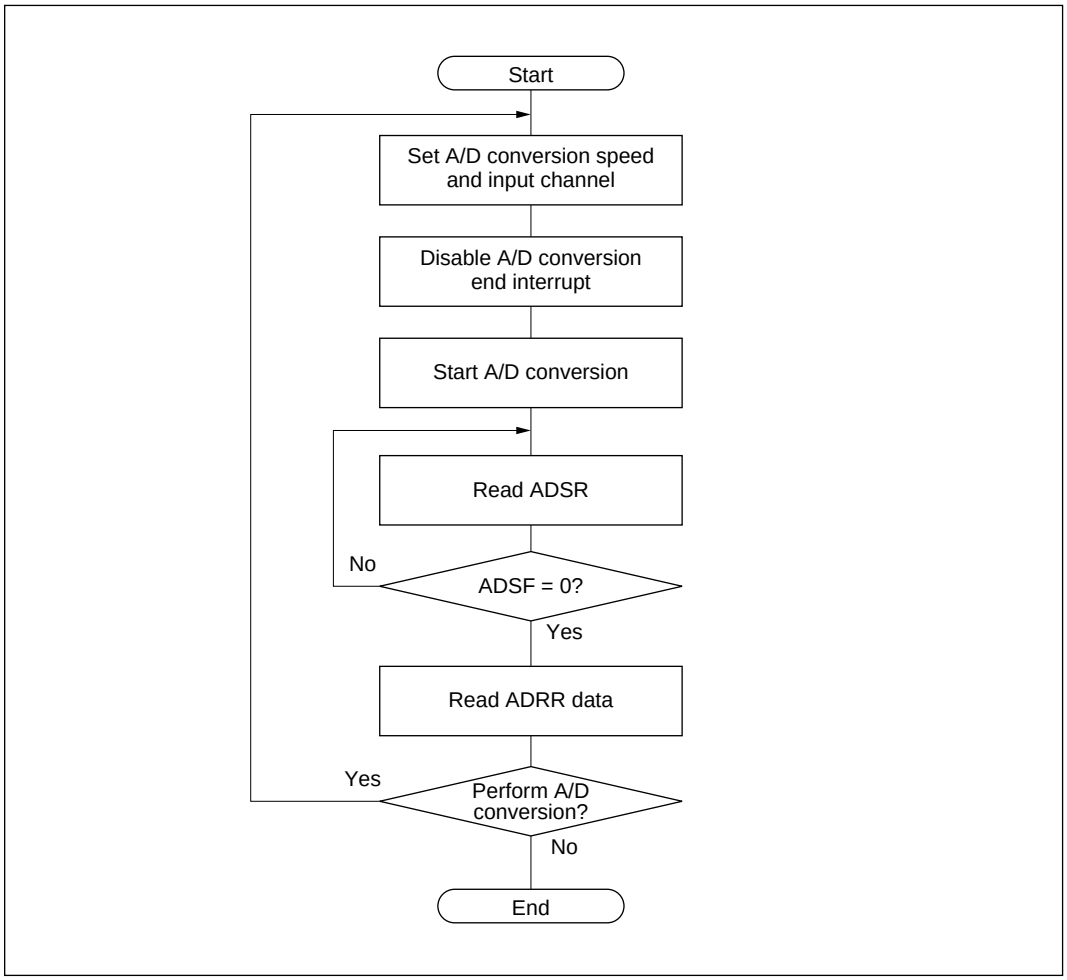


Figure 12-4 Flow Chart of Procedure for Using A/D Converter (1) (Polling by Software)

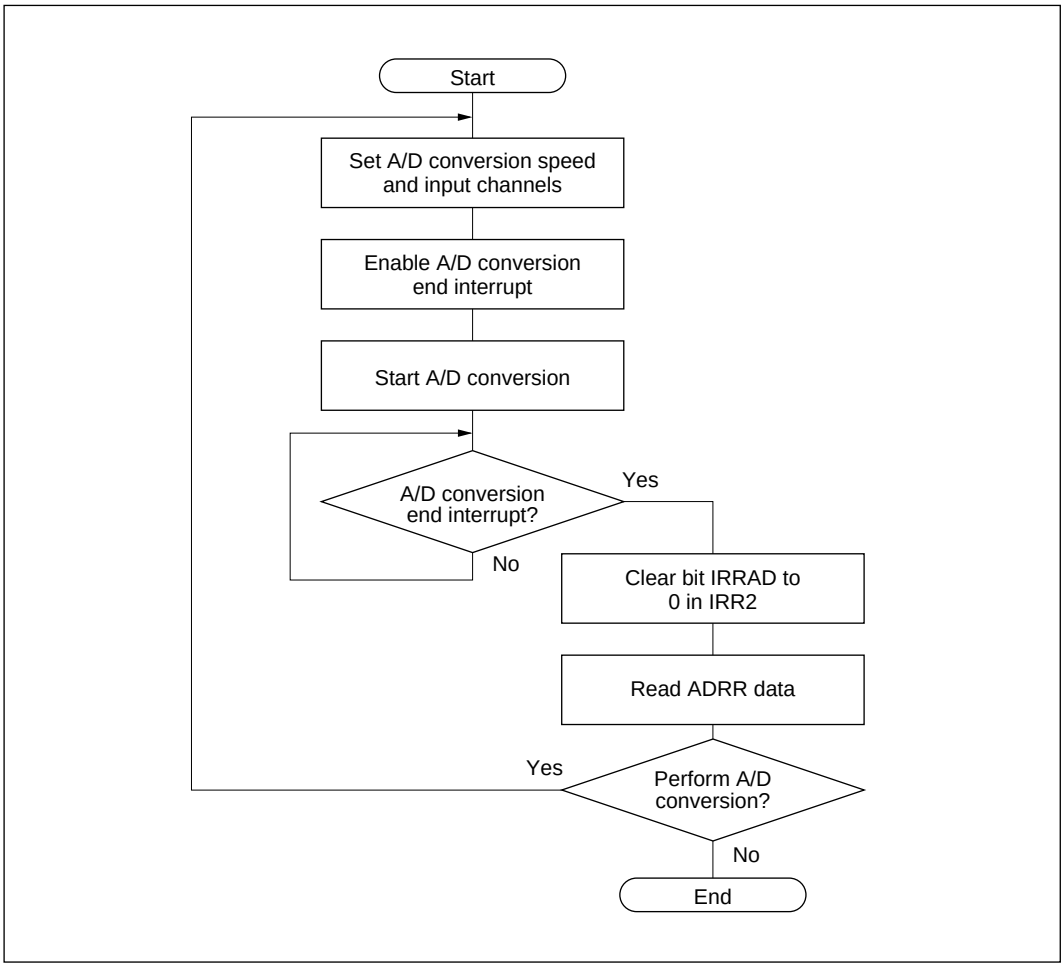


Figure 12-5 Flow Chart of Procedure for Using A/D Converter (2) (Interrupts Used)

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

Item		Symbol	Value	Unit
Power supply voltage		V_{CC}	–0.3 to +7.0	V
Analog power supply voltage		AV_{CC}	–0.3 to +7.0	V
Programming voltage		V_{PP}	–0.3 to +13.0	V
Input voltage	Ports other than ports 3, B and C	V_{in}	–0.3 to V_{CC} +0.3	V
	Port 3	V_{in}	–0.3 to 15.0	V
	Ports B and C	AV_{in}	–0.3 to AV_{CC} +0.3	V
Operating temperature		T_{opr}	–20 to +75	°C
Storage temperature		T_{stg}	–55 to +125	°C

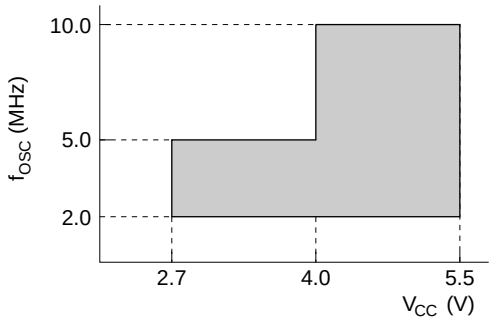
Note: Permanent damage may occur to the chip if maximum ratings are exceeded. Normal operation should be under the conditions specified in Electrical Characteristics. Exceeding these values can result in incorrect operation and reduced reliability.

13.2 Electrical Characteristics

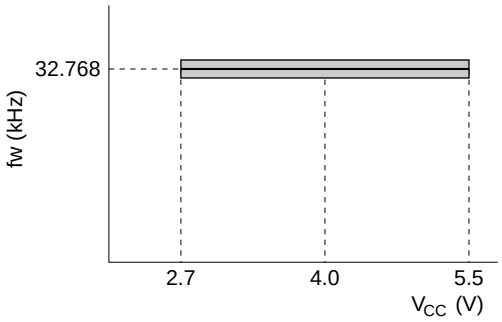
13.2.1 Power Supply Voltage and Operating Range

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

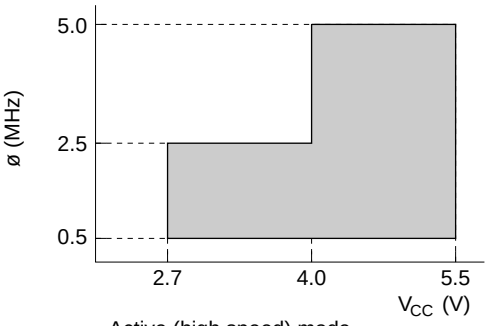


- Active (high speed) mode
- Sleep mode

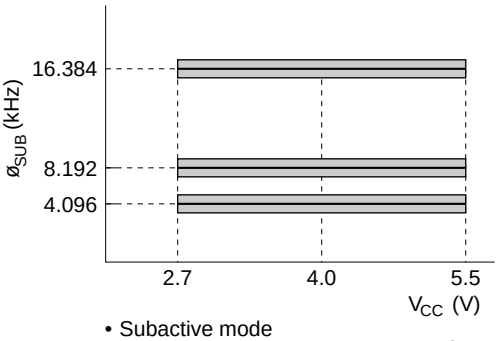


- All operating modes

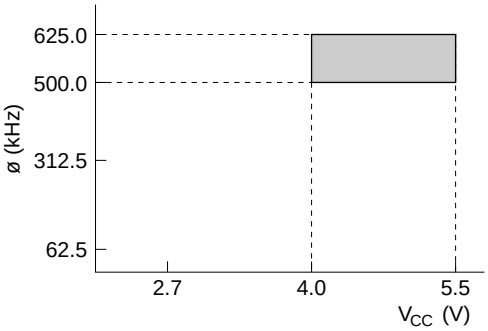
2. Power supply voltage vs. clock frequency range



- Active (high speed) mode
- Sleep mode (except CPU and I^2C bus interface)

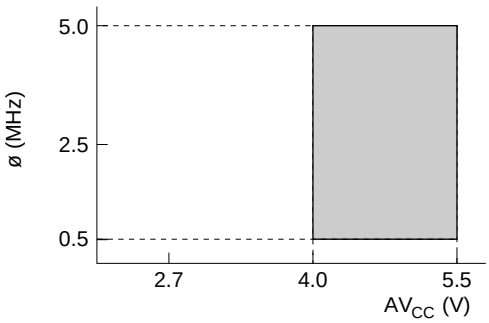


- Subactive mode
- Subsleep mode (except CPU and I^2C bus interface)
- Watch mode (except CPU and I^2C bus interface)

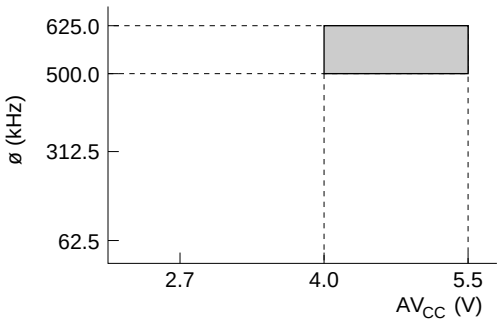


- Active (medium speed) mode

3. Analog power supply voltage vs. A/D converter operating range

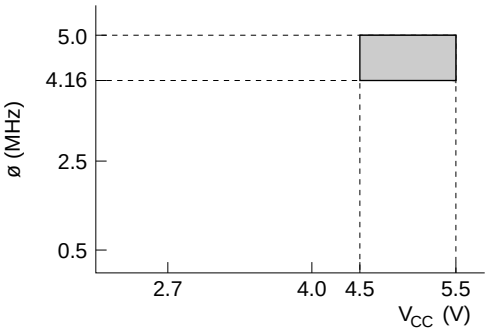


• Active (high speed) mode
• Sleep mode



• Active (medium speed) mode

4. Power supply voltage vs. I²C bus interface operating range



• Active (high speed) mode
• Sleep mode

Do not use in the following modes:

• Active (medium speed) mode
• Subactive mode
• Subsleep mode
• Watch mode

13.2.2 DC Characteristics

Table 13-2 lists the DC characteristics.

Table 13-2 DC Characteristics

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Note
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{NMI}$, $\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{IRQ_0}$ to $\overline{IRQ_4}$, TMIB1, TMIB2A, TMIB2B, TMIC, TMIF, TMIG, SCK_3, $\overline{ADTRG}$, TMCIH, TMRIH	V_{CC} = 4.0 V to 5.5 V	0.8 V_{CC}	—	V_{CC} + 0.3	V	
				0.9 V_{CC}	—	V_{CC} + 0.3		
		UD, RXD, SCL_1, SDA_1, SCL_2, SDA_2	V_{CC} = 4.0 V to 5.5 V	0.7 V_{CC}	—	V_{CC} + 0.3	V	
				0.8 V_{CC}	—	V_{CC} + 0.3		
		OSC_1	V_{CC} = 4.0 V to 5.5 V	V_{CC} – 0.5	—	V_{CC} + 0.3	V	
				V_{CC} – 0.3	—	V_{CC} + 0.3		
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P4_0$ to $P4_3$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_7$, PA_0 to PA_6	V_{CC} = 4.0 V to 5.5 V	0.7 V_{CC}	—	V_{CC} + 0.3	V	
				0.8 V_{CC}	—	V_{CC} + 0.3		
		$P3_0$ to $P3_7$	V_{CC} = 4.0 V to 5.5 V	0.7 V_{CC}	—	12.0	V	
				0.8 V_{CC}	—	12.0		
		PB_0 to PB_7, PC_0 to PC_3	V_{CC} = 4.0 V to 5.5 V	0.7 V_{CC}	—	AV_{CC} + 0.3	V	
				0.8 V_{CC}	—	AV_{CC} + 0.3		
Input low voltage	V_{IL}	$\overline{RES}$, $\overline{NMI}$, $\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{IRQ_0}$ to $\overline{IRQ_4}$, TMIB1, TMIB2A, TMIB2B, TMIC, TMIF, TMIG, SCK_3, $\overline{ADTRG}$, TMCIH, TMRIH	V_{CC} = 4.0 V to 5.5 V	–0.3	—	0.2 V_{CC}	V	
				–0.3	—	0.1 V_{CC}		
		UD, RXD, SCL_1, SDA_1, SCL_2, SDA_2	V_{CC} = 4.0 V to 5.5 V	–0.3	—	0.3 V_{CC}	V	
				–0.3	—	0.2 V_{CC}		

Note: Connect pin TEST to V_{SS}.

Table 13-2 DC Characteristics (cont)

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Note
Input low voltage	V_{IL}	OSC_1	V_{CC} = 4.0 V to 5.5 V	–0.3	—	0.5	V	
				–0.3	—	0.3		
		$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_3$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_7$, PA_0 to PA_6, PB_0 to PB_7, PC_0 to PC_3	V_{CC} = 4.0 V to 5.5 V	–0.3	—	0.3 V_{CC}	V	
				–0.3	—	0.2 V_{CC}		
Output high voltage	V_{OH}	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P4_0$ to $P4_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_5$ to $P9_7$, PA_0 to PA_6	V_{CC} = 4.0 V to 5.5 V, $–I_{OH}$ = 1.0 mA	V_{CC} – 1.0	—	—	V	
			V_{CC} = 4.0 V to 5.5 V, $–I_{OH}$ = 0.5 mA	V_{CC} – 0.5	—	—		
			$–I_{OH}$ = 0.1 mA	V_{CC} – 0.5	—	—		
Output low voltage	V_{OL}	$P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P4_0$ to $P4_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_5$ to $P9_7$	V_{CC} = 4.0 V to 5.5 V, I_{OL} = 1.6 mA	—	—	0.6	V	
			I_{OL} = 0.4 mA	—	—	0.5		
		$P3_0$ to $P3_7$, PA_0 to PA_6	V_{CC} = 4.0 V to 5.5 V, I_{OL} = 10 mA	—	—	1.5		
			V_{CC} = 4.0 V to 5.5 V, I_{OL} = 1.6 mA	—	—	0.6		
			I_{OL} = 0.4 mA	—	—	0.5		
		SCL_1, SDA_1, SCL_2, SDA_2	V_{CC} = 4.0 V to 5.5 V, I_{OL} = 6 mA	—	—	0.6		
			I_{OL} = 0.4 mA	—	—	0.5		

Note: Connect pin TEST to V_{SS}.

Table 13-2 DC Characteristics (cont)

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Note
Input leakage current	$\|I_{IL}\|$	$\overline{RES}$, $P4_3$	V_{IN} = 0.5 V to V_{CC} – 0.5 V	—	—	20.0	µA	2
				—	—	1.0		1
		OSC_1, $\overline{NMI}$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P4_0$ to $P4_2$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_1$ to $P9_7$, PA_0 to PA_3	V_{IN} = 0.5 V to V_{CC} – 0.5 V	—	—	1.0	µA	
		$P3_0$ to $P3_7$	V_{IN} = 0.5 V to 12.0 V	—	—	20.0		
		PB_0 to PB_7, PC_0 to PC_3	V_{IN} = 0.5 V to AV_{CC} – 0.5 V	—	—	1.0		
Pull-up MOS current	$-I_P$	$P1_0$ to $P1_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$	V_{CC} = 5 V, V_{IN} = 0 V	50.0	—	300.0	µA	
			V_{CC} = 2.7 V, V_{IN} = 0 V	—	30.0	—		Reference value
Input capacitance	C_{IN}	All input pins except power supply, $\overline{RES}$, $P4_3$, and $P9_1$ to $P9_4$	f = 1 MHz, V_{IN} = 0 V, T_a = 25°C	—	—	15.0	pF	
		$\overline{RES}$		—	—	60.0		2
				—	—	15.0		1
		$P4_3$		—	—	30.0		2
				—	—	15.0		1
		$P9_1$ to $P9_4$		—	—	20.0		
Active mode current dissipation	I_{OPE1}	V_{CC}	Active mode (high speed), V_{CC} = 5 V, f_{osc} = 10 MHz	—	14.0	24.0	mA	3, 4
	I_{OPE2}	V_{CC}	Active mode (medium speed), V_{CC} = 5 V, f_{osc} = 10 MHz	—	2.5	5.0	mA	3, 4
Sleep mode current dissipation	I_{SLEEP}	V_{CC}	V_{CC} = 5 V, f_{osc} = 10 MHz	—	6.5	10.0	mA	3, 4

Notes:
1. Applies to HD6433947.
2. Applies to HD6473947.
3. Pin states during current measurement are given in the table below.
4. Excludes current in pull-up MOS transistors and output buffers.

Table 13-2 DC Characteristics (cont)

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Note
Subactive mode current dissipation	I_{SUB}	V_{CC}	V_{CC} = 2.7 V, 32-kHz crystal oscillator ($ø_{SUB}$ = ø w/2)	—	30.0	70.0	µA	1, 2
			V_{CC} = 2.7 V, 32-kHz crystal oscillator ($ø_{SUB}$ = ø w/8)	—	15.0	—	µA	Reference value 1, 2
Subsleep mode current dissipation	I_{SUBSP}	V_{CC}	V_{CC} = 2.7 V, 32-kHz crystal oscillator ($ø_{SUB}$ = ø w/2)	—	20.0	50.0	µA	1, 2
Watch mode current dissipation	I_{WATCH}	V_{CC}	V_{CC} = 2.7 V, 32-kHz crystal oscillator	—	—	6.0	µA	1, 2
Standby mode current dissipation	I_{STBY}	V_{CC}	32-kHz crystal oscillator not used	—	—	5.0	µA	1, 2
RAM data retaining voltage	V_{RAM}	V_{CC}		2.0	—	—	V	1, 2

Notes: 1. Pin states during current measurement are given in the table below.
2. Excludes current in pull-up MOS transistors and output buffers.

Mode	Internal State	Other Pins	Oscillator Pins
Active mode (high and medium speed)	Operates	V_{CC}	System clock oscillator: Crystal Subclock oscillator: Pin X_1 = V_{CC}
Sleep mode	Only timers operate	V_{CC}	
Subactive mode	Operates	V_{CC}	System clock oscillator: Crystal Subclock oscillator: Crystal
Subsleep mode	Only timers operate, CPU stops	V_{CC}	
Watch mode	Only time-base clock operates, CPU stops	V_{CC}	
Standby mode	CPU and timers all stop	V_{CC}	System clock oscillator: Crystal Subclock oscillator: Pin X_1 = V_{CC}

Note: Excludes current in pull-up MOS transistors and output buffers.

Table 13-2 DC Characteristics (cont)

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise indicated.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit
Allowable output low current (per pin)	I_{OL}	Output pins except in ports 3 and A	V_{CC} = 4.0 V to 5.5 V	—	—	2.0	mA
		Ports 3 and A, SCL_1, SDA_1, SCL_2, SDA_2	V_{CC} = 4.0 V to 5.5 V	—	—	10.0	
		All output pins		—	—	0.5	
Allowable output low current (total)	ΣI_{OL}	Output pins except in ports 3 and A	V_{CC} = 4.0 V to 5.5 V	—	—	40.0	mA
		Ports 3 and A, SCL_1, SDA_1, SCL_2, SDA_2	V_{CC} = 4.0 V to 5.5 V	—	—	80.0	
		All output pins		—	—	20.0	
Allowable output high current (per pin)	$-I_{OH}$	All output pins	V_{CC} = 4.0 V to 5.5 V	—	—	2.0	mA
				—	—	0.2	
Allowable output high current (total)	$\Sigma-I_{OH}$	All output pins	V_{CC} = 4.0 V to 5.5 V	—	—	15.0	mA
				—	—	10.0	

13.2.3 AC Characteristics

Table 13-3 lists the control signal timing, and tables 13-4 and 13-5 list the serial interface timing.

Table 13-3 Control Signal Timing

V_{CC} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise specified.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Reference Figure
System clock oscillation frequency	f_{OSC}	OSC_1, OSC_2	V_{CC} = 4.0 V to 5.5 V	2.0	—	10.0	MHz	
				2.0	—	5.0		
OSC clock ($ø_{OSC}$) cycle time	t_{OSC}	OSC_1, OSC_2	V_{CC} = 4.0 V to 5.5 V	100.0	—	1000.0	ns	1 Figure 13-1
				200.0	—	1000.0		
System clock (ø) cycle time	t_{cyc}			2	—	16	t_{OSC}	1
				—	—	2000.0	ns	
Subclock oscillation frequency	f_W	X_1, X_2		—	32.678	—	kHz	
Watch clock ($ø_W$) cycle time	t_W	X_1, X_2		—	30.5	—	µs	
Subclock ($ø_{SUB}$) cycle time	t_{subcyc}			2	—	8	t_W	2
Instruction cycle time				2	—	—	t_{cyc} t_{subcyc}	
Oscillation stabilization time (crystal oscillator)	t_{rc}	OSC_1, OSC_2	V_{CC} = 4.0 V to 5.5 V	—	—	40.0	ms	
				—	—	60.0		
Oscillation stabilization time	t_{rc}	X_1, X_2	—	—	—	2.0	s	
External clock high width	t_{CPH}	OSC_1	V_{CC} = 4.0 V to 5.5 V	40.0	—	—	ns	Figure 13-1
				80.0	—	—		
External clock low width	t_{CPL}	OSC_1	V_{CC} = 4.0 V to 5.5 V	40.0	—	—	ns	Figure 13-1
				80.0	—	—		
External clock rise time	t_{CPr}		V_{CC} = 4.0 V to 5.5 V	—	—	15.0	ns	Figure 13-1
				—	—	20.0		
External clock fall time	t_{CPf}		V_{CC} = 4.0 V to 5.5 V	—	—	15.0	ns	Figure 13-1
				—	—	20.0		
Pin $\overline{RES}$ low width	t_{REL}	$\overline{RES}$		10	—	—	t_{cyc}	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} = 2.7 V to 5.5 V, AV_{CC} = 2.7 V to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, T_a = –20°C to +75°C, including subactive mode, unless otherwise specified.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Reference Figure
Input pin high width	t_{IH}	$\overline{IRQ_0}$ to $\overline{IRQ_4}$ $\overline{WKP_0}$ to $\overline{WKP_7}$ $\overline{ADTRG}$, $\overline{NMI}$, TMIB1, TMIB2A, TMIB2B, TMIC, TMIF, TMIG, TMCIH, TMRIH		2	—	—	t_{cyc} t_{subcyc}	Figure 13-3
Input pin low width	t_{IL}	$\overline{IRQ_0}$ to $\overline{IRQ_4}$, $\overline{WKP_0}$ to $\overline{WKP_7}$, $\overline{ADTRG}$, $\overline{NMI}$, TMIB1, TMIB2A, TMIB2B, TMIC, TMIF, TMIG, TMCIH, TMRIH		2	—	—	t_{cyc} t_{subcyc}	Figure 13-3
Pin UD minimum modulation width	t_{UDH} t_{UDL}	UD		4	—	—	t_{cyc} t_{subcyc}	Figure 13-4

Table 13-4 Serial Interface Timing (SCI3)

V_{CC} = 2.7 V to 5.5 V, AV_{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	Test Conditions	Min	Typ	Max	Unit	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 mode)		t_{TXD}	V_{CC} = 4.0 V to 5.5 V	—	—	1	t_{cyc}	Figure 13-6
				—	—	1		
Receive data setup time (synchronous mode)		t_{RXS}	V_{CC} = 4.0 V to 5.5 V	200.0	—	—	ns	
				400.0	—	—		
Receive data hold time (synchronous mode)		t_{RXH}	V_{CC} = 4.0 V to 5.5 V	200.0	—	—	ns	
				400.0	—	—		

Table 13-5 Serial Interface Timing (I^2C1, I^2C2)

V_{CC} = 4.5 V to 5.5 V, AV_{CC} = 4.5 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	Test Conditions	Min	Typ	Max	Unit	Reference Figure
SCL clock cycle time	t_{SCL}	SCL_1, SCL_2		$12t_{cyc}$	—	—	ns	Figure 13-7
SCL clock high pulse width	t_{SCLH}	SCL_1, SCL_2		$3t_{cyc}$	—	—	ns	
SCL clock low pulse width	t_{SCLL}	SCL_1, SCL_2		$5t_{cyc}$	—	—	ns	
SCL and SDA rise time	t_{Sr}	SCL_1, SDA_1 SCL_2, SDA_2	Standard mode, 100 kbits/s (max)	—	—	1000	ns	Figure 13-7 *1
			High-speed mode, 400 kbits/s (max)	$20 + 0.1C_b$	—	300		
SCL and SDA fall time	t_{Sf}	SCL_1, SDA_1, SCL_2, SDA_2	Standard mode, 100 kbits/s (max)	—	—	300	ns	Figure 13-7 *2
			High-speed mode, 400 kbits/s (max)	$(20 + 0.1C_b)/2$	—	300		
Bus free time	t_{BUF}	SDA_1, SDA_2		$7t_{cyc} – 300$	—	—	ns	Figure 13-7
Start condition hold time	t_{STAH}	SCL_1, SCL_2		$3t_{cyc}$	—	—	ns	
Retransmitted start condition setup time	t_{STAS}	SCL_1, SCL_2		$3t_{cyc}$	—	—	ns	
Stop condition setup time	t_{STOS}	SDA_1, SDA_2		$3t_{cyc}$	—	—	ns	
SDA data setup time	t_{SDAS}	SDA_1, SDA_2		$1t_{cyc} + 10$	—	—	ns	Figure 13-7 *3
SDA data hold time	t_{SDAH}	SDA_1, SDA_2		0	—	—	ns	Figure 13-7
Capacitive load on bus lines	C_b	SDA_1, SDA_2		—	—	400	pF	

Notes: 1. In master mode, if the rise time t_{Sr} is greater than 2.5 t_{cyc}, in order to secure the clock high pulse time, the transfer rate will decrease accordingly.
2. Refer to figure 13-9 regarding characteristics of SCL and SDA fall time t_{Sf}.
3. t_{SDAS} is less than 250 ns when t_{cyc} is between 200 ns and 240 ns (f_{osc} = 10 MHz to 8.33 MHz).

13.2.4 A/D Converter Characteristics

Table 13-6 shows the A/D converter characteristics.

Table 13-6 A/D Converter Characteristics

V_{CC} = 2.7 V to 5.5 V, AV_{SS} = V_{SS} = 0.0 V, T_a = –20°C to +75°C, unless otherwise specified.

Item	Symbol	Applicable Pins	Test Conditions	Min	Typ	Max	Unit	Note
Analog power supply voltage	AV_{CC}	AV_{CC}		4.0	—	5.5	V	1
Analog input voltage	AV_{IN}	AN_0 to AN_{11}		–0.3	—	AV_{CC} + 0.3	V	
Analog power supply current	AI_{OPE}	AV_{CC}	AV_{CC} = 5.0 V	—	—	1.5	mA	
	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_{11}		—	—	30	pF	
Allowable signal source impedance	R_{AIN}			—	—	10	kΩ	
Resolution (data length)				—	—	8	bit	
Non-linearity error				—	—	±2.0	LSB	
Quantization error				—	—	±0.5	LSB	
Absolute accuracy				—	—	±2.5	LSB	
Conversion time			AV_{CC} = 4.5 V to 5.5 V	12.4	—	124	µs	
			AV_{CC} = 4.0 V to 5.5 V	24.8	—	124		

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 Operation Timing

Figures 13-1 to 13-7 show timing diagrams.

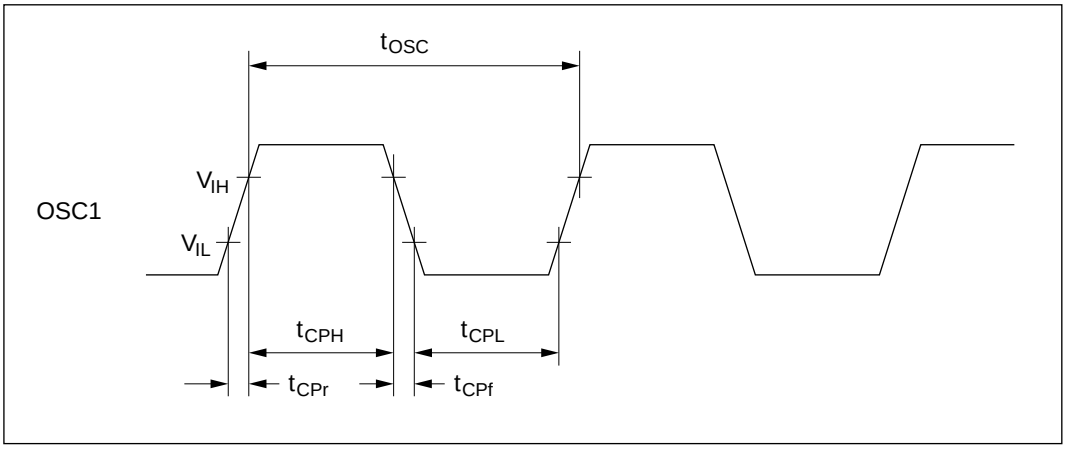


Figure 13-1 System Clock Input Timing

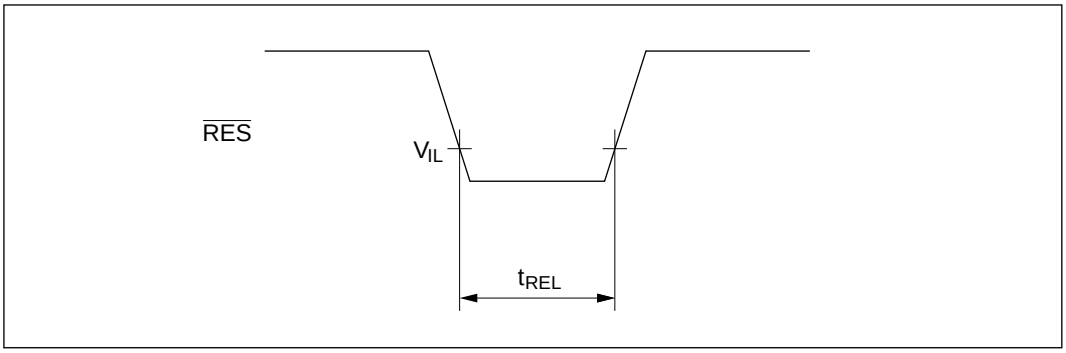


Figure 13-2 RES Low Width

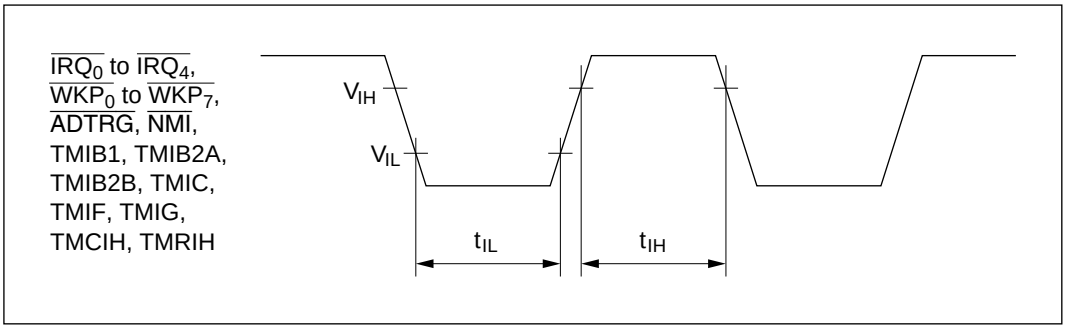


Figure 13-3 Input Timing

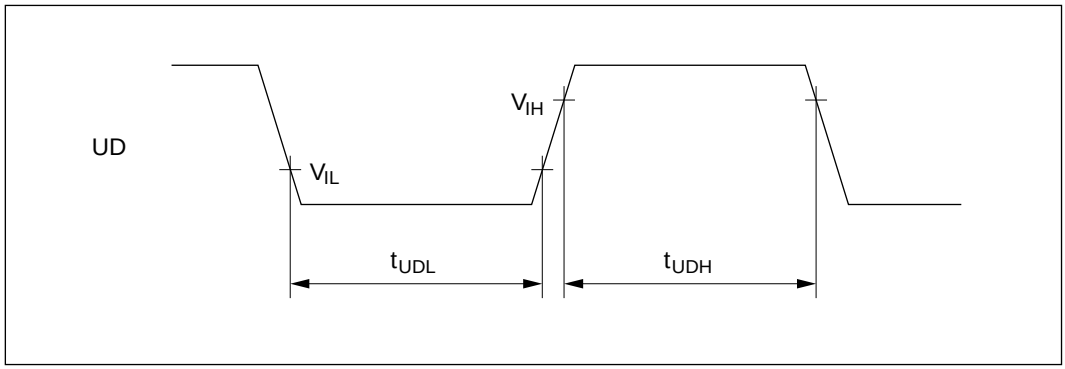


Figure 13-4 Minimum UD High and Low Width

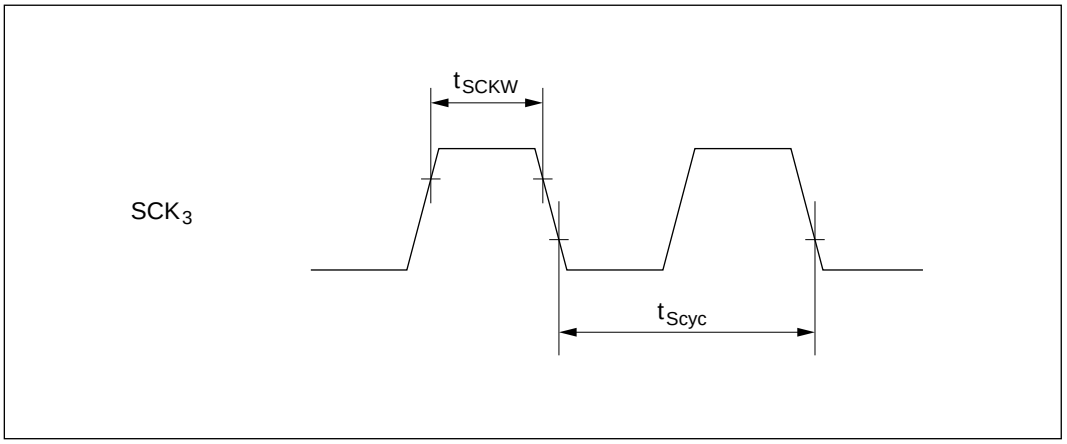
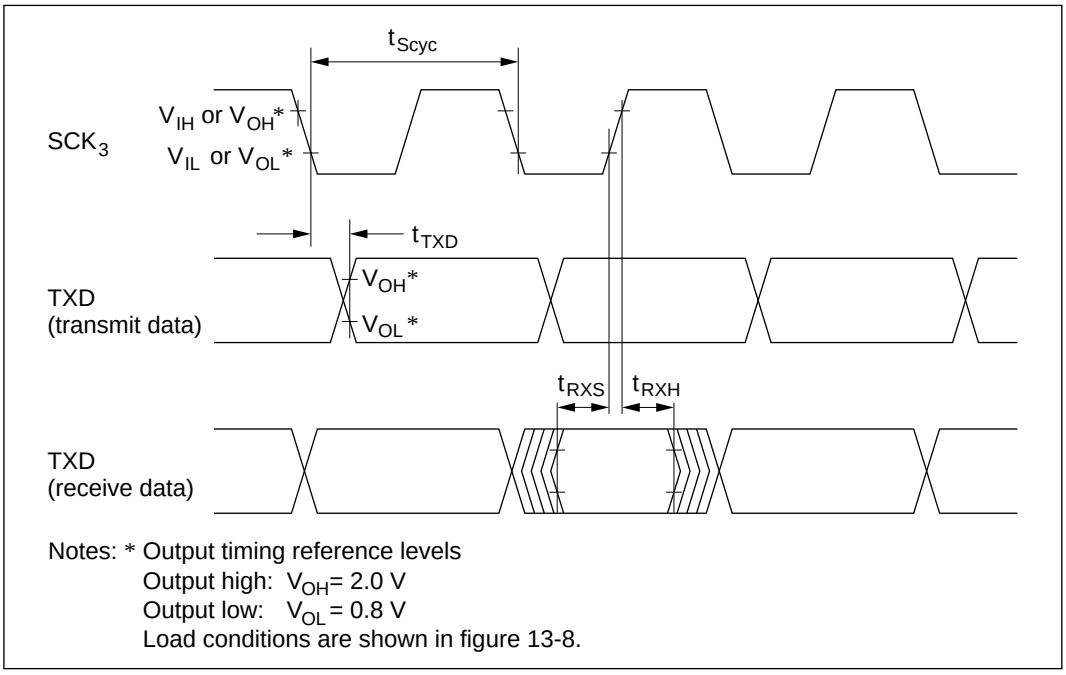
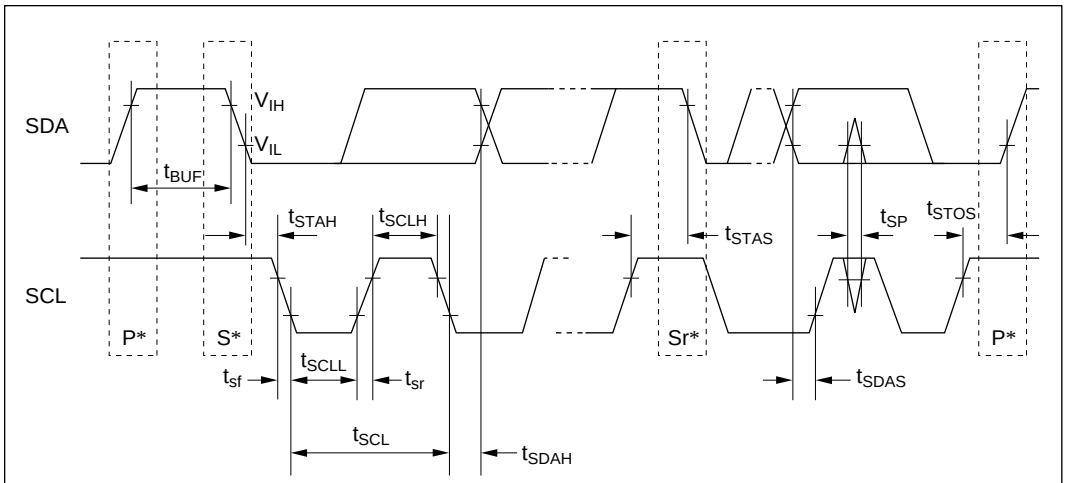


Figure 13-5 SCK_3 Input Clock Timing



Notes: * Output timing reference levels
Output high: V_{OH}= 2.0 V
Output low: V_{OL} = 0.8 V
Load conditions are shown in figure 13-8.

Figure 13-6 Input/Output Timing of Serial Interface 3 in Synchronous Mode



Note: * S, P, and Sr have the following meanings:

S: Start condition
P: Stop condition
Sr: Retransmitted start condition

Figure 13-7 I^2C1 and I^2C2 Bus Interface Input/Output Timing

13.4 Output Load Circuit

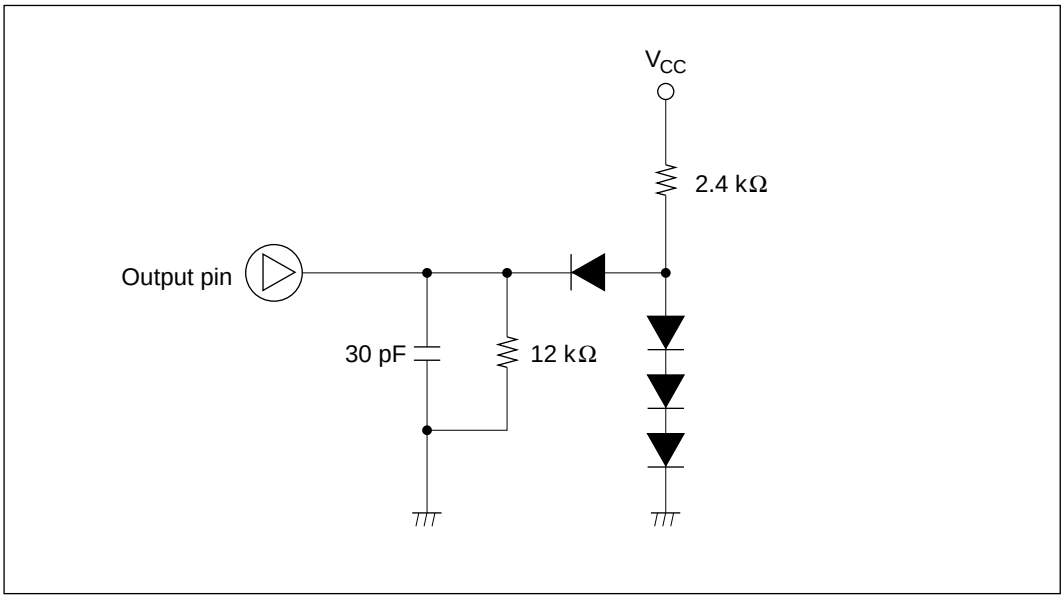


Figure 13-8 Output Load Condition

13.4 Electrical Characteristics Diagram

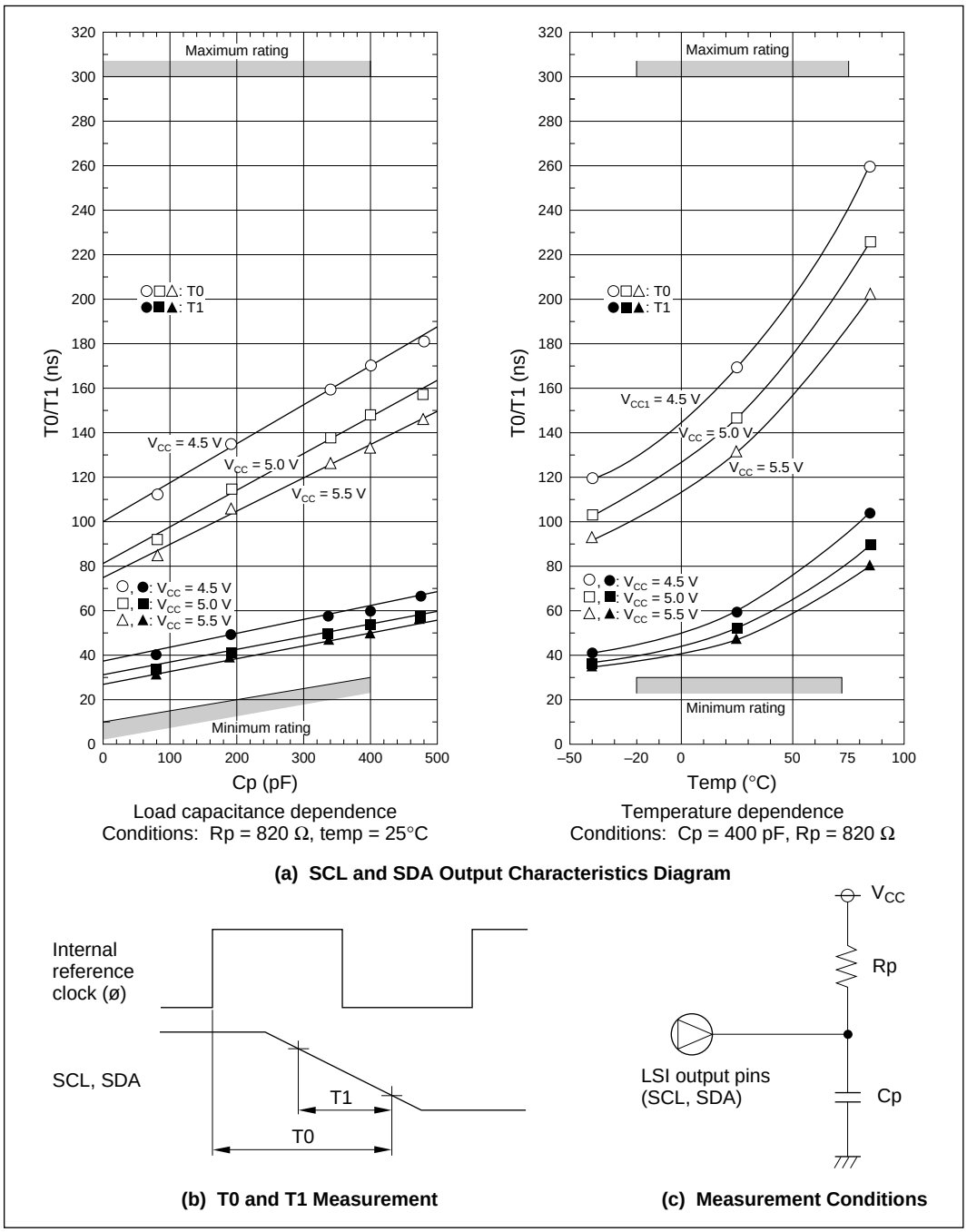


Figure 13-9 I²C Bus Interface Electrical Characteristics Diagram

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	—	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
PUSH Rs	W	SP–2 → SP Rs16 → @SP					2					—	—	↕	↕	0	—	6

Table A-1 Instruction Set (cont)

Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	|	I	H	N	Z	V	C	
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

Table A-1 Instruction Set (cont)

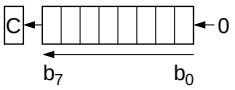
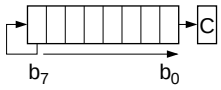
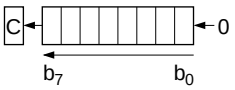
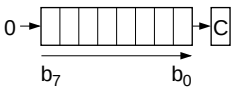
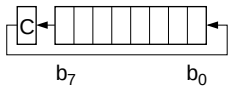
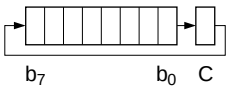
Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	—	I	H	N	Z	V	C	
MULXU.B Rs, Rd	B	Rd8 × Rs8 → Rd16		2								—	—	—	—	—	—	14
DIVXU.B Rs, Rd	B	Rd16÷Rs8 → Rd16 (RdH: remainder, RdL: quotient)		2								—	—	[5]	[6]	—	—	14
AND.B #xx:8, Rd	B	Rd8∧#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
AND.B Rs, Rd	B	Rd8∧Rs8 → Rd8		2								—	—	↕	↕	0	—	2
OR.B #xx:8, Rd	B	Rd8∨#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
OR.B Rs, Rd	B	Rd8∨Rs8 → Rd8		2								—	—	↕	↕	0	—	2
XOR.B #xx:8, Rd	B	Rd8⊕#xx:8 → Rd8	2									—	—	↕	↕	0	—	2
XOR.B Rs, Rd	B	Rd8⊕Rs8 → Rd8		2								—	—	↕	↕	0	—	2
NOT.B Rd	B	$\overline{Rd}$ → Rd		2								—	—	↕	↕	0	—	2
SHAL.B Rd	B			2								—	—	↕	↕	↕	↕	2
SHAR.B Rd	B			2								—	—	↕	↕	0	↕	2
SHLL.B Rd	B			2								—	—	↕	↕	0	↕	2
SHLR.B Rd	B			2								—	—	0	↕	0	↕	2
ROTXL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTXR.B Rd	B			2								—	—	↕	↕	0	↕	2

Table A-1 Instruction Set (cont)

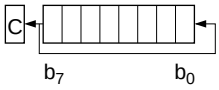
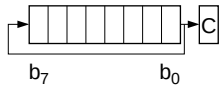
Mnemonic	Operand Size	Operation	Addressing Mode/ Instruction Length (bytes)									Condition Code						No. of States
			#xx: 8/16	Rn	@Rn	@(d:16, Rn)	@–Rn/@Rn+	@aa: 8/16	@(d:8, PC)	@@aa	|	I	H	N	Z	V	C	
ROTL.B Rd	B			2								—	—	↕	↕	0	↕	2
ROTR.B Rd	B			2								—	—	↕	↕	0	↕	2
BSET #xx:3, Rd	B	(#xx:3 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BSET Rn, Rd	B	(Rn8 of Rd8) ← 1		2								—	—	—	—	—	—	2
BSET Rn, @Rd	B	(Rn8 of @Rd16) ← 1			4							—	—	—	—	—	—	8
BSET Rn, @aa:8	B	(Rn8 of @aa:8) ← 1						4				—	—	—	—	—	—	8
BCLR #xx:3, Rd	B	(#xx:3 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR #xx:3, @Rd	B	(#xx:3 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BCLR Rn, Rd	B	(Rn8 of Rd8) ← 0		2								—	—	—	—	—	—	2
BCLR Rn, @Rd	B	(Rn8 of @Rd16) ← 0			4							—	—	—	—	—	—	8
BCLR Rn, @aa:8	B	(Rn8 of @aa:8) ← 0						4				—	—	—	—	—	—	8
BNOT #xx:3, Rd	B	(#xx:3 of Rd8) ← $\overline{(\#xx:3 \text{ of } Rd8)}$		2								—	—	—	—	—	—	2
BNOT #xx:3, @Rd	B	(#xx:3 of @Rd16) ← $\overline{(\#xx:3 \text{ of } @Rd16)}$			4							—	—	—	—	—	—	8
BNOT #xx:3, @aa:8	B	(#xx:3 of @aa:8) ← $\overline{(\#xx:3 \text{ of } @aa:8)}$						4				—	—	—	—	—	—	8
BNOT Rn, Rd	B	(Rn8 of Rd8) ← $\overline{(Rn8 \text{ of } Rd8)}$		2								—	—	—	—	—	—	2
BNOT Rn, @Rd	B	(Rn8 of @Rd16) ← $\overline{(Rn8 \text{ of } @Rd16)}$			4							—	—	—	—	—	—	8
BNOT Rn, @aa:8	B	(Rn8 of @aa:8) ← $\overline{(Rn8 \text{ of } @aa:8)}$						4				—	—	—	—	—	—	8

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	—	I	H	N	Z	V	C	
BTST #xx:3, Rd	B	$(\overline{\#xx:3 \text{ of } Rd8}) \rightarrow Z$		2								—	—	—	↕	—	—	2
BTST #xx:3, @Rd	B	$(\overline{\#xx:3 \text{ of } @Rd16}) \rightarrow Z$			4							—	—	—	↕	—	—	6
BTST #xx:3, @aa:8	B	$(\overline{\#xx:3 \text{ of } @aa:8}) \rightarrow Z$						4				—	—	—	↕	—	—	6
BTST Rn, Rd	B	$(\overline{Rn8 \text{ of } Rd8}) \rightarrow Z$		2								—	—	—	↕	—	—	2
BTST Rn, @Rd	B	$(\overline{Rn8 \text{ of } @Rd16}) \rightarrow Z$			4							—	—	—	↕	—	—	6
BTST Rn, @aa:8	B	$(\overline{Rn8 \text{ of } @aa:8}) \rightarrow Z$						4				—	—	—	↕	—	—	6
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	$(\overline{\#xx:3 \text{ of } Rd8}) \rightarrow C$		2								—	—	—	—	—	↕	2
BILD #xx:3, @Rd	B	$(\overline{\#xx:3 \text{ of } @Rd16}) \rightarrow C$			4							—	—	—	—	—	↕	6
BILD #xx:3, @aa:8	B	$(\overline{\#xx:3 \text{ of } @aa:8}) \rightarrow C$						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	$\overline{C} \rightarrow$ (#xx:3 of Rd8)		2								—	—	—	—	—	—	2
BIST #xx:3, @Rd	B	$\overline{C} \rightarrow$ (#xx:3 of @Rd16)			4							—	—	—	—	—	—	8
BIST #xx:3, @aa:8	B	$\overline{C} \rightarrow$ (#xx:3 of @aa:8)						4				—	—	—	—	—	—	8
BAND #xx:3, Rd	B	C∧(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BAND #xx:3, @Rd	B	C∧(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BAND #xx:3, @aa:8	B	C∧(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIAND #xx:3, Rd	B	$C \wedge (\overline{\#xx:3 \text{ of } Rd8}) \rightarrow C$		2								—	—	—	—	—	↕	2
BIAND #xx:3, @Rd	B	$C \wedge (\overline{\#xx:3 \text{ of } @Rd16}) \rightarrow C$			4							—	—	—	—	—	↕	6
BIAND #xx:3, @aa:8	B	$C \wedge (\overline{\#xx:3 \text{ of } @aa:8}) \rightarrow C$						4				—	—	—	—	—	↕	6
BOR #xx:3, Rd	B	C∨(#xx:3 of Rd8) → C		2								—	—	—	—	—	↕	2
BOR #xx:3, @Rd	B	C∨(#xx:3 of @Rd16) → C			4							—	—	—	—	—	↕	6
BOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C						4				—	—	—	—	—	↕	6
BIOR #xx:3, Rd	B	$C \vee (\overline{\#xx:3 \text{ of } Rd8}) \rightarrow C$		2								—	—	—	—	—	↕	2
BIOR #xx:3, @Rd	B	$C \vee (\overline{\#xx:3 \text{ of } @Rd16}) \rightarrow C$			4							—	—	—	—	—	↕	6

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	|	I	H	N	Z	V	C	
BIOR #xx:3, @aa:8	B	C∨(#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⊕(#xx:3 of Rd8) → C			2								—	—	—	—	—	↕	2
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

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	—	I	H	N	Z	V	C	
JSR @Rn	—	SP–2 → SP PC → @SP PC ← Rn16			2							—	—	—	—	—	—	6
JSR @aa:16	—	SP–2 → SP PC → @SP PC ← aa:16						4				—	—	—	—	—	—	8
JSR @@aa:8		SP–2 → SP PC → @SP PC ← @aa:8								2		—	—	—	—	—	—	8
RTS	—	PC ← @SP SP+2 → SP									2	—	—	—	—	—	—	8
RTE	—	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
SLEEP	—	Transit to sleep mode.									2	—	—	—	—	—	—	2
LDC #xx:8, CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
LDC Rs, CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
STC CCR, Rd	B	CCR → Rd8		2								—	—	—	—	—	—	2
ANDC #xx:8, CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
ORC #xx:8, CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
XORC #xx:8, CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
NOP	—	PC ← PC+2									2	—	—	—	—	—	—	2

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).

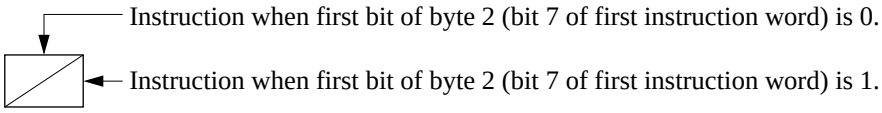


Table A-2 Operation Code Map

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, data read/write, etc.) 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:

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 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 = $2 \times 2 + 2 \times 2 = 8$
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 States in Each Cycle

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					
	MOV.B @Rs, Rd	1			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 @(d:16, Rs), Rd	2			1		
	MOV.B @Rs+, Rd	1			1		2
	MOV.B @aa:8, Rd	1			1		
	MOV.B @aa:16, Rd	2			1		
	MOV.B Rs, @Rd	1			1		
	MOV.B Rs, @(d:16, Rd)	2			1		
	MOV.B Rs, @–Rd	1			1		2
	MOV.B Rs, @aa:8	1			1		
	MOV.B Rs, @aa:16	2			1		
	MOV.W #xx:16, Rd	2					
	MOV.W Rs, Rd	1					
	MOV.W @Rs, Rd	1				1	
	MOV.W @(d:16, Rs), Rd	2				1	
	MOV.W @Rs+, Rd	1				1	2
	MOV.W @aa:16, Rd	2				1	
	MOV.W Rs, @Rd	1				1	
	MOV.W Rs, @(d:16, Rd)	2				1	
	MOV.W Rs, @–Rd	1				1	2
	MOV.W Rs, @aa:16	2				1	
MULXU	MULXU.B Rs, Rd	1					12
NEG	NEG.B Rd	1					
NOP	NOP	1					
NOT	NOT.B Rd	1					
OR	OR.B #xx:8, Rd	1					
	OR.B Rs, Rd	1					
ORC	ORC #xx:8, CCR	1					
POP	POP Rd	1		1			2
PUSH	PUSH Rs	1		1			2
ROTL	ROTL.B Rd	1					
ROTR	ROTR.B Rd	1					
ROTXL	ROTXL.B Rd	1					
ROTXR	ROTXR.B Rd	1					
RTE	RTE	2		2			2
RTS	RTS	2		1			2

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
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					
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 On-Chip Registers

B.1 I/O Registers (1)

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'80	ICCR1	ICE1	IEIC1	MST1	TRS1	ACK1	CKS12	CKS11	CKS10	I²C
H'81	ICSR1	BBSY1	IRIC1	SCP1	—	AL1	AAS1	ADZ1	ACKB1	
H'82	ICDR1	ICDR17	ICDR16	ICDR15	ICDR14	ICDR13	ICDR12	ICDR11	ICDR10	
H'83	ICMR1/ SAR1	MLS1/ SVA16	WAIT1/ SVA15	—/ SVA14	—/ SVA13	—/ SVA12	BC12/ SVA11	BC11/ SVA10	BC10/ FS1	
H'84	ICCR2	ICE2	IEIC2	MST2	TRS2	ACK2	CKS22	CKS21	CKS20	
H'85	ICSR2	BBSY2	IRIC2	SCP2	—	AL2	AAS2	ADZ2	ACKB2	
H'86	ICDR2	ICDR27	ICDR26	ICDR25	ICDR24	ICDR23	ICDR22	ICDR21	ICDR20	
H'87	ICMR2/ SAR2	MLS2/ SVA26	WAIT2/ SVA25	—/ SVA24	—/ SVA23	—/ SVA22	BC22/ SVA21	BC21/ SVA20	BC20/ FS2	
H'88	TCRH	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	Timer H
H'89	TCSRH	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	
H'8A	TCORAH	TCORAH7	TCORAH6	TCORAH5	TCORAH4	TCORAH3	TCORAH2	TCORAH1	TCORAH0	
H'8B	TCORBH	TCORBH7	TCORBH6	TCORBH5	TCORBH4	TCORBH3	TCORBH2	TCORBH1	TCORBH0	
H'8C	TCNTH	TCNTH7	TCNTH6	TCNTH5	TCNTH4	TCNTH3	TCNTH2	TCNTH1	TCNTH0	
H'8D										
H'8E										
H'8F										
H'90	PWCR	—	—	—	—	—	—	—	CKS	8-bit PWM
H'91	PWDR0	$PWDR0_7$	$PWDR0_6$	$PWDR0_5$	$PWDR0_4$	$PWDR0_3$	$PWDR0_2$	$PWDR0_1$	$PWDR0_0$	
H'92	PWDR1	$PWDR1_7$	$PWDR1_6$	$PWDR1_5$	$PWDR1_4$	$PWDR1_3$	$PWDR1_2$	$PWDR1_1$	$PWDR1_0$	
H'93	PWDR2	$PWDR2_7$	$PWDR2_6$	$PWDR2_5$	$PWDR2_4$	$PWDR2_3$	$PWDR2_2$	$PWDR2_1$	$PWDR2_0$	
H'94	PWDR3	$PWDR3_7$	$PWDR3_6$	$PWDR3_5$	$PWDR3_4$	$PWDR3_3$	$PWDR3_2$	$PWDR3_1$	$PWDR3_0$	
H'95	PWDR4	$PWDR4_7$	$PWDR4_6$	$PWDR4_5$	$PWDR4_4$	$PWDR4_3$	$PWDR4_2$	$PWDR4_1$	$PWDR4_0$	
H'96	PWDR5	$PWDR5_7$	$PWDR5_6$	$PWDR5_5$	$PWDR5_4$	$PWDR5_3$	$PWDR5_2$	$PWDR5_1$	$PWDR5_0$	
H'97	PWDR6	$PWDR6_7$	$PWDR6_6$	$PWDR6_5$	$PWDR6_4$	$PWDR6_3$	$PWDR6_2$	$PWDR6_1$	$PWDR6_0$	
H'98	PWDR7	$PWDR7_7$	$PWDR7_6$	$PWDR7_5$	$PWDR7_4$	$PWDR7_3$	$PWDR7_2$	$PWDR7_1$	$PWDR7_0$	
H'99										
H'9A										
H'9B										

Notation:
I²C: I²C bus interface

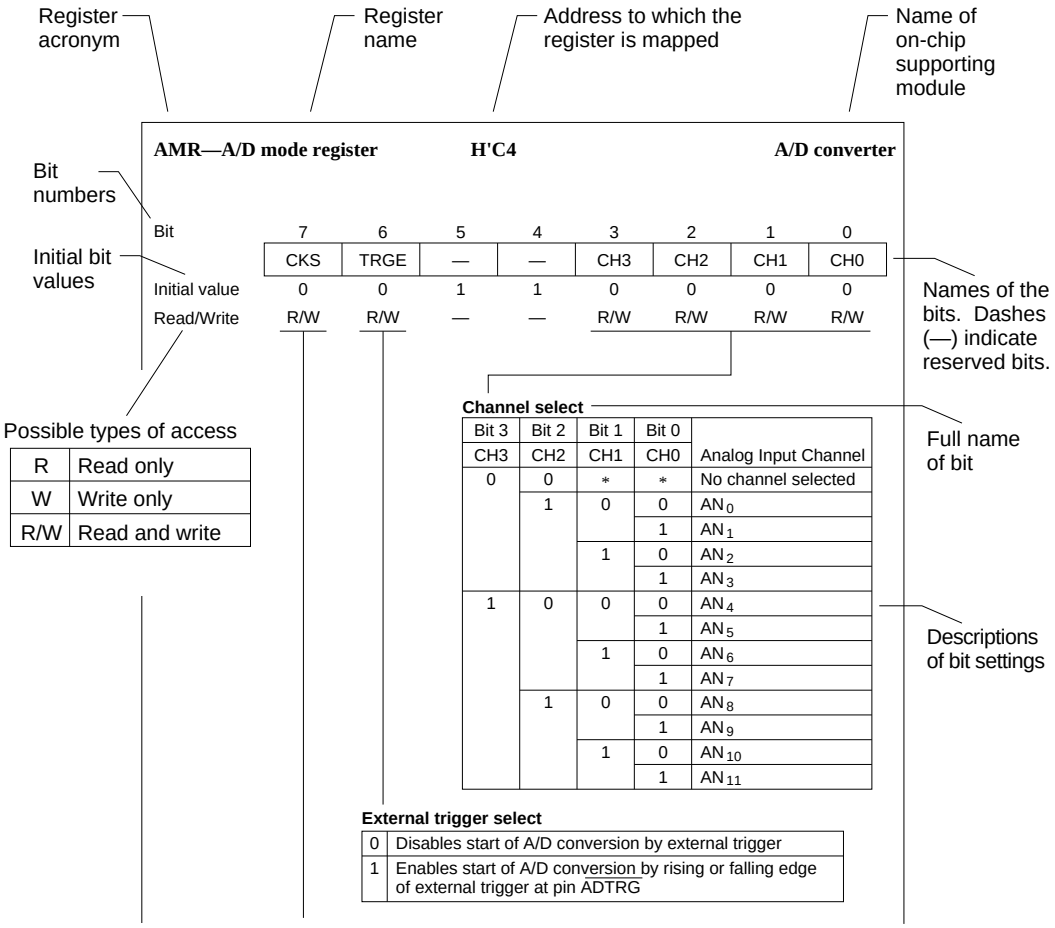
Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'9C	TMB2	TMB27	—	—	—	—	TMB22	TMB21	TMB20	Timer B2
H'9D	TCB2/ TLB2	TCB27/ TLB27	TCB26/ TLB26	TCB25/ TLB25	TCB24/ TLB24	TCB23/ TLB23	TCB22/ TLB22	TCB21/ TLB21	TCB20/ TLB20	
H'9E	TMB3	TMB37	—	—	—	—	TMB32	TMB31	TMB30	Timer B3
H'9F	TCB3/ TLB3	TCB37/ TLB37	TCB36/ TLB36	TCB35/ TLB35	TCB34/ TLB34	TCB33/ TLB33	TCB32/ TLB32	TCB31/ TLB31	TCB320/ TLB30	
H'A0										
H'A1										
H'A2										
H'A3										
H'A4										
H'A5										
H'A6										
H'A7										
H'A8	SMR	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0	SCI3
H'A9	BRR	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0	
H'AA	SCR3	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
H'AB	TDR	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0	
H'AC	SSR	TDRE	RDRF	OER	FER	PER	TEND	MPBR	MPBT	
H'AD	RDR	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0	
H'AE										
H'AF										
H'B0	TMA	TMA7	TMA6	TMA5	—	TMA3	TMA2	TMA1	TMA0	Timer A
H'B1	TCA	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0	
H'B2	TMB1	TMB17	—	—	—	—	TMB12	TMB11	TMB10	Timer B1
H'B3	TCB1/ TLB1	TCB17/ TLB17	TCB16/ TLB16	TCB15/ TLB15	TCB14/ TLB14	TCB13/ TLB13	TCB12/ TLB12	TCB11/ TLB11	TCB10/ TLB10	
H'B4	TMC	TMC7	TMC6	TMC5	—	—	TMC2	TMC1	TMC0	Timer C
H'B5	TCC/TLC	TCC7/ TLC7	TCC6/ TLC6	TCC5/ TLC5	TCC4/ TLC4	TCC3/ TLC3	TCC2/ TLC2	TCC1/ TLC1	TCC0/ TLC0	
H'B6	TCRF	TOLH	CKSH2	CKSH1	CKSH0	TOLL	CKSL2	CKSL1	CKSL0	Timer F
H'B7	TCSRF	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL	
H'B8	TCFH	TCFH7	TCFH6	TCFH5	TCFH4	TCFH3	TCFH2	TCFH1	TCFH0	
H'B9	TCFL	TCFL7	TCFL6	TCFL5	TCFL4	TCFL3	TCFL2	TCFL1	TCFL0	
H'BA	OCRFH	OCRFH7	OCRFH6	OCRFH5	OCRFH4	OCRFH3	OCRFH2	OCRFH1	OCRFH0	

Notation:
SCI3: Serial communication interface 3

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'BB	OCRFL	OCRFL7	OCRFL6	OCRFL5	OCRFL4	OCRFL3	OCRFL2	OCRFL1	OCRFL0	Timer F
H'BC	TMG	OVFH	OVFL	OVIE	IIEGS	CCLR1	CCLR0	CKS1	CKS0	Timer G
H'BD	ICRGF	ICRGF7	ICRGF6	ICRGF5	ICRGF4	ICRGF3	ICRGF2	ICRGF1	ICRGF0	
H'BE	ICRGR	ICRGR7	ICRGR6	ICRGR5	ICRGR4	ICRGR3	ICRGR2	ICRGR1	ICRGR0	
H'BF										
H'C0										
H'C1										
H'C2										
H'C3										
H'C4	AMR	CKS	TRGE	—	—	CH3	CH2	CH1	CH0	A/D converter
H'C5	ADRR	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	
H'C6	ADSR	ADSF	—	—	—	—	—	—	—	
H'C7										
H'C8	PMR1	IRQ3	IRQ2	IRQ1	—	TMIG	TMOFH	TMOFL	TMOW	I/O ports
H'C9	PMR2	TMIB2B	TMIB2A	EDGB2	NCS	IRQ0	—	UD	IRQ4	
H'CA	PMR3	PWM7	PWM6	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0	
H'CB										
H'CC	PMR5	WKP_7	WKP_6	WKP_5	WKP_4	WKP_3	WKP_2	WKP_1	WKP_0	
H'CD										
H'CE										
H'CF										
H'D0										
H'D1										
H'D2										
H'D3										
H'D4	PDR1	$P1_7$	$P1_6$	$P1_5$	$P1_4$	$P1_3$	$P1_2$	$P1_1$	$P1_0$	I/O ports
H'D5	PDR2	$P2_7$	$P2_6$	$P2_5$	$P2_4$	$P2_3$	$P2_2$	$P2_1$	$P2_0$	
H'D6	PDR3	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$	
H'D7	PDR4	—	—	—	—	$P4_3$	$P4_2$	$P4_1$	$P4_0$	
H'D8	PDR5	$P5_7$	$P5_6$	$P5_5$	$P5_4$	$P5_3$	$P5_2$	$P5_1$	$P5_0$	
H'D9	PDR6	$P6_7$	$P6_6$	$P6_5$	$P6_4$	$P6_3$	$P6_2$	$P6_1$	$P6_0$	
H'DA	PDR7	$P7_7$	$P7_6$	$P7_5$	$P7_4$	$P7_3$	$P7_2$	$P7_1$	$P7_0$	
H'DB	PDR8	$P8_7$	$P8_6$	$P8_5$	$P8_4$	$P8_3$	$P8_2$	$P8_1$	$P8_0$	
H'DC	PDR9	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	$P9_0$	
H'DD	PDRA	—	PA_6	PA_5	PA_4	PA_3	PA_2	PA_1	PA_0	

Address (Low)	Register Name	Bit Names								Module Name
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'DE	PDRB	PB_7	PB_6	PB_5	PB_4	PB_3	PB_2	PB_1	PB_0	I/O ports
H'DF	PDRC	—	—	—	—	PC_3	PC_2	PC_1	PC_0	
H'E0	PUCR1	$PUCR1_7$	$PUCR1_6$	$PUCR1_5$	$PUCR1_4$	$PUCR1_3$	$PUCR1_2$	$PUCR1_1$	$PUCR1_0$	
H'E1	PUCR7	$PUCR7_7$	$PUCR7_6$	$PUCR7_5$	$PUCR7_4$	$PUCR7_3$	$PUCR7_2$	$PUCR7_1$	$PUCR7_0$	
H'E2	PUCR5	$PUCR5_7$	$PUCR5_6$	$PUCR5_5$	$PUCR5_4$	$PUCR5_3$	$PUCR5_2$	$PUCR5_1$	$PUCR5_0$	
H'E3	PUCR6	$PUCR6_7$	$PUCR6_6$	$PUCR6_5$	$PUCR6_4$	$PUCR6_3$	$PUCR6_2$	$PUCR6_1$	$PUCR6_0$	
H'E4	PCR1	$PCR1_7$	$PCR1_6$	$PCR1_5$	$PCR1_4$	$PCR1_3$	$PCR1_2$	$PCR1_1$	$PCR1_0$	
H'E5	PCR2	$PCR2_7$	$PCR2_6$	$PCR2_5$	$PCR2_4$	$PCR2_3$	$PCR2_2$	$PCR2_1$	$PCR2_0$	
H'E6	PCR3	$PCR3_7$	$PCR3_6$	$PCR3_5$	$PCR3_4$	$PCR3_3$	$PCR3_2$	$PCR3_1$	$PCR3_0$	
H'E7	PCR4	—	—	—	—	—	$PCR4_2$	$PCR4_1$	$PCR4_0$	
H'E8	PCR5	$PCR5_7$	$PCR5_6$	$PCR5_5$	$PCR5_4$	$PCR5_3$	$PCR5_2$	$PCR5_1$	$PCR5_0$	
H'E9	PCR6	$PCR6_7$	$PCR6_6$	$PCR6_5$	$PCR6_4$	$PCR6_3$	$PCR6_2$	$PCR6_1$	$PCR6_0$	
H'EA	PCR7	$PCR7_7$	$PCR7_6$	$PCR7_5$	$PCR7_4$	$PCR7_3$	$PCR7_2$	$PCR7_1$	$PCR7_0$	
H'EB	PCR8	$PCR8_7$	$PCR8_6$	$PCR8_5$	$PCR8_4$	$PCR8_3$	$PCR8_2$	$PCR8_1$	$PCR8_0$	
H'EC	PCR9	$PCR9_7$	$PCR9_6$	$PCR9_5$	—	—	—	—	—	
H'ED	PCRA	—	$PCRA_6$	$PCRA_5$	$PCRA_4$	$PCRA_3$	$PCRA_2$	$PCRA_1$	$PCRA_0$	
H'EE										
H'EF										
H'F0	SYSCR1	SSBY	STS2	STS1	STS0	LSON	—	—	—	System control
H'F1	SYSCR2	—	—	—	NESEL	DTON	MSON	SA1	SA0	
H'F2	IEGR	NMIEG	—	—	IEG4	IEG3	IEG2	IEG1	IEG0	
H'F3	IENR1	IENTA	IENTB3	IENWP	IEN4	IEN3	IEN2	IEN1	IEN0	
H'F4	IENR2	IENDT	IENAD	IENTB2	IENTG	IENTFH	IENTFL	IENTC	IENTB1	
H'F5										
H'F6	IRR1	IRRTA	IRRTB3	—	IRRI4	IRRI3	IRRI2	IRRI1	IRRI0	System control
H'F7	IRR2	IRRDT	IRRAD	IRRTB2	IRRTG	IRRTFH	IRRTFL	IRRTC	IRRTB1	
H'F8										
H'F9	IWPR	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0	System control
H'FA										
H'FB										
H'FC										
H'FD										
H'FE										
H'FF										
H'FF										

B.2 I/O Registers (2)



ICCR1—I²C bus control register 1 **H'80** **I²C**

Bit	7	6	5	4	3	2	1	0
	ICE1	IEIC1	MST1	TRS1	ACK1	CKS12	CKS11	CKS10
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

Transfer clock select

CKS12	CKS11	CKS10	Clock	Transfer Rate		
				ø = 2 MHz	ø = 4 MHz	ø = 5 MHz
0	0	0	ø/14	143 kHz	286 kHz	357 kHz
		1	ø/20	100 kHz	200 kHz	250 kHz
	1	0	ø/24	83.3 kHz	167 kHz	208 kHz
		1	ø/32	62.5 kHz	125 kHz	156 kHz
1	0	0	ø/40	50.0 kHz	100 kHz	125 kHz
		1	ø/50	40.0 kHz	80.0 kHz	100 kHz
	1	0	ø/56	35.7 kHz	71.4 kHz	89.3 kHz
		1	ø/64	31.3 kHz	62.5 kHz	78.1 kHz

Acknowledgement mode select

0	Acknowledgement mode
1	Serial mode

Master/slave select and transmit/receive select

0	0	Slave receive mode
	1	Slave transmit mode
1	0	Master receive mode
	1	Master transmit mode

I²C bus interface interrupt enable

0	Interrupts disabled
1	Interrupts enabled

I²C bus interface enable

0	Interface module disabled, with SCL_1 and SDA_1 signals in high-impedance state
1	Interface module enabled for transfer operations

Bit	7	6	5	4	3	2	1	0
	BBSY1	IRIC1	SCP1	—	AL1	AAS1	ADZ1	ACKB1
Initial value	0	0	1	1	0	0	0	0
Read/Write	R/W	R/(W)*	W	—	R/(W)*	R/(W)*	R/(W)*	R/W

Acknowledge bit

0	Receive mode: 0 is output at acknowledge output timing Transmit mode: indicates that the receiving device has acknowledged the data
1	Receive mode: 1 is output at acknowledge output timing Transmit mode: indicates that the receiving device has not acknowledged the data

General call address recognition flag

0	General call address not recognized [Clearing conditions] Cleared when ICDR1 data is written (transmit mode) or read (receive mode) After reading ADZ1 = 1, cleared by writing 0 to ADZ1
1	General call address recognized [Setting conditions] When the general call address is detected in slave receive mode

Slave address recognition flag

0	Slave address or general call address not recognized [Clearing conditions] Cleared when ICDR1 data is written (transmit mode) or read (receive mode) After reading AAS1 = 1, cleared by writing 0 to AAS1
1	Slave address or general call address recognized [Setting conditions] When the slave address or general call address is detected in slave receive mode

Arbitration lost flag

0	Bus arbitration won [Clearing conditions] Cleared when ICDR1 data is written (transmit mode) or read (receive mode) After reading AL1 = 1, cleared by writing 0 to AL1
1	Arbitration lost [Setting conditions] When the internal SDA_1 and bus line disagree at the rise of SCL_1 in master transmit mode When the internal SCL_1 is high at the fall of SCL_1 in master transmit mode

Start condition/stop condition prohibit

0	Writing 0 issues a start or stop condition, in combination with BBSY1
1	Reading always results in 1 Writing is ignored

I²C bus interface interrupt request flag

0	Waiting for transfer, or transfer in progress [Clearing conditions] After reading IRIC1 = 1, cleared by writing 0 to IRIC1
1	Interrupt requested [Setting conditions] Master mode • End of data transfer • Bus arbitration lost Slave mode (when FS1 = 0) • When the slave address is matched, and whenever a data transfer ends after that, until a retransmitted start condition or a stop condition is detected • When a general call address is detected, and whenever a data transfer ends after that, until a retransmitted start condition or a stop condition is detected Slave mode (when FS1 = 1) • End of data transfer

Bus busy

0	Bus is free Cleared by detection of a stop condition
1	Bus is busy Set by detection of a start condition

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

ICDR1—I²C bus data register 1 **H'82** **I²C**

Bit	7	6	5	4	3	2	1	0
	ICDR17	ICDR16	ICDR15	ICDR14	ICDR13	ICDR12	ICDR11	ICDR10
Initial value	—	—	—	—	—	—	—	—
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Transmit/receive data

SAR1—Slave address register 1 **H'83** **I²C**

Bit	7	6	5	4	3	2	1	0
	SVA16	SVA15	SVA14	SVA13	SVA12	SVA11	SVA10	FS1
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

Slave address

Format Select

0	Addressing format, slave address recognized
1	Non-addressing format

ICMR1—I²C bus mode register 1 **H'83** **I²C**

Bit	7	6	5	4	3	2	1	0
	MLS1	WAIT1	—	—	—	BC12	BC11	BC10
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

Bit counter

BC12	BC11	BC10	Bits/Frame	
			Serial Mode	Acknowledgement Mode
0	0	0	8	9
		1	1	2
	1	0	2	3
		1	3	4
1	0	0	4	5
		1	5	6
	1	0	6	7
		1	7	8

Wait insertion bit

0	Data and acknowledge transferred consecutively
1	Wait inserted between data and acknowledge

MSB-first/LSB-first

0	MSB-first
1	LSB-first

ICCR2—I²C bus control register 2 **H'84** **I²C**

Bit	7	6	5	4	3	2	1	0
	ICE2	IEIC2	MST2	TRS2	ACK2	CKS22	CKS21	CKS20
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

Transfer clock select

CKS22	CKS21	CKS20	Clock	Transfer Rate		
				ø = 2 MHz	ø = 4 MHz	ø = 5 MHz
0	0	0	ø/14	143 kHz	286 kHz	357 kHz
		1	ø/20	100 kHz	200 kHz	250 kHz
	1	0	ø/24	83.3 kHz	167 kHz	208 kHz
		1	ø/32	62.5 kHz	125 kHz	156 kHz
1	0	0	ø/40	50.0 kHz	100 kHz	125 kHz
		1	ø/50	40.0 kHz	80.0 kHz	100 kHz
	1	0	ø/56	35.7 kHz	71.4 kHz	89.3 kHz
		1	ø/64	31.3 kHz	62.5 kHz	78.1 kHz

Acknowledgement mode select

0	Acknowledgement mode
1	Serial mode

Master/slave select and transmit/receive select

0	0	Slave receive mode
	1	Slave transmit mode
1	0	Master receive mode
	1	Master transmit mode

I²C bus interface interrupt enable

0	Interrupts disabled
1	Interrupts enabled

I²C bus interface enable

0	Interface module disabled, with SCL_2 and SDA_2 signals in high-impedance state
1	Interface module enabled for transfer operations

Bit	7	6	5	4	3	2	1	0
	BBSY2	IRIC2	SCP2	—	AL2	AAS2	ADZ2	ACKB2
Initial value	0	0	1	1	0	0	0	0
Read/Write	R/W	R/(W)*	W	—	R/(W)*	R/(W)*	R/(W)*	R/W

Acknowledge bit

0	Receive mode: 0 is output at acknowledge output timing Transmit mode: indicates that the receiving device has acknowledged the data
1	Receive mode: 1 is output at acknowledge output timing Transmit mode: indicates that the receiving device has not acknowledged the data

General call address recognition flag

0	General call address not recognized [Clearing conditions] Cleared when ICDR2 data is written (transmit mode) or read (receive mode) After reading ADZ2 = 1, cleared by writing 0 to ADZ2
1	General call address recognized [Setting conditions] When the general call address is detected in slave receive mode

Slave address recognition flag

0	Slave address or general call address not recognized [Clearing conditions] Cleared when ICDR2 data is written (transmit mode) or read (receive mode) After reading AAS2 = 1, cleared by writing 0 to AAS2
1	Slave address or general call address recognized [Setting conditions] When the slave address or general call address is detected in slave receive mode

Arbitration lost flag

0	Bus arbitration won [Clearing conditions] Cleared when ICDR2 data is written (transmit mode) or read (receive mode) After reading AL2 = 1, cleared by writing 0 to AL2
1	Arbitration lost [Setting conditions] When the internal SDA_2 and bus line disagree at the rise of SCL_2 in master transmit mode When the internal SCL_2 is high at the fall of SCL_2 in master transmit mode

Start condition/stop condition prohibit

0	Writing 0 issues a start or stop condition, in combination with BBSY1
1	Reading always results in 1 Writing is ignored

I²C bus interface interrupt request flag

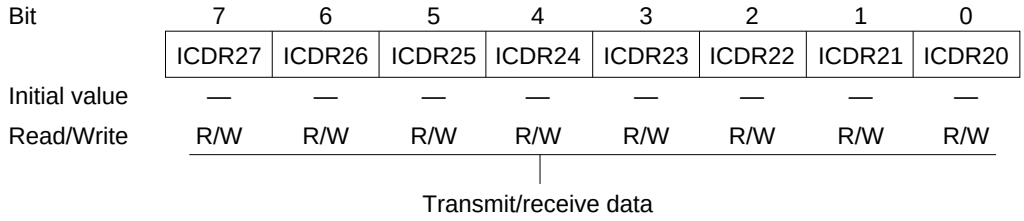
0	Waiting for transfer, or transfer in progress [Clearing conditions] After reading IRIC2 = 1, cleared by writing 0 to IRIC2
1	Interrupt requested [Setting conditions] Master mode • End of data transfer • Bus arbitration lost Slave mode (when FS2 = 0) • When the slave address is matched, and whenever a data transfer ends after that, until a retransmitted start condition or a stop condition is detected • When a general call address is detected, and whenever a data transfer ends after that, until a retransmitted start condition or a stop condition is detected Slave mode (when FS2 = 1) • End of data transfer

Bus busy

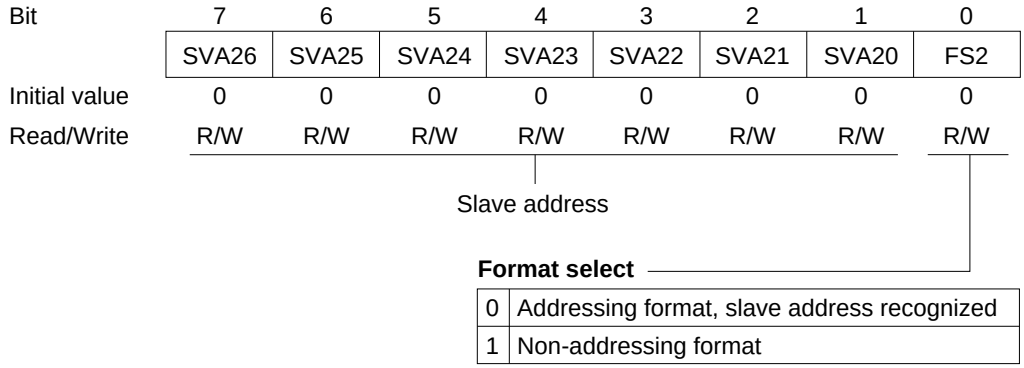
0	Bus is free Cleared by detection of a stop condition
1	Bus is busy Set by detection of a start condition

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

ICDR2—I²C bus data register 2 **H'86** **I²C**



SAR2—Slave address register 2 **H'87** **I²C**



0	Addressing format, slave address recognized
1	Non-addressing format

ICMR2—I²C bus mode register 2 **H'87** **I²C**

Bit	7	6	5	4	3	2	1	0
	MLS2	WAIT2	—	—	—	BC22	BC21	BC20
Initial value	0	0	1	1	1	0	0	0
Read/Write	R/W	R/W	—	—	—	R/W	R/W	R/W

Bit counter

BC22	BC21	BC20	Bits/Frame	
			Serial Mode	Acknowledgement Mode
0	0	0	8	9
		1	1	2
	1	0	2	3
		1	3	4
1	0	0	4	5
		1	5	6
	1	0	6	7
		1	7	8

Wait insertion bit

0	Data and acknowledge transferred consecutively
1	Wait inserted between data and acknowledge

MSB-first/LSB-first

0	MSB-first
1	LSB-first

TCRH—Timer control register H **H'88** **Timer H**

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

Clock select

0	0	0	Clock input disabled
		1	Internal clock: ø/8, falling edge
	1	0	Internal clock: ø/64, falling edge
		1	Internal clock: ø/1024, falling edge
1	0	0	Clock input disabled
		1	External clock: rising edge
	1	0	External clock: falling edge
		1	External clock: both edges

Counter clear

0	0	Clearing disabled
	1	Cleared by compare match A
1	0	Cleared by compare match B
	1	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

TCSRH—Timer control/status register H **H'89** **Timer H**

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 1 and 0

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 3 and 2

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] When TCNTH changes 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] When TCNTH = TCORAH

Compare match flag B

0	[Clearing condition] After reading CMFB = 1, cleared by writing 0 in CMFB
1	[Setting condition] When TCNTH = TCORBH

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

TCORAH—Time constant register AH **H'8A** **Timer H**

Bit	7	6	5	4	3	2	1	0
	TCORAH7	TCORAH6	TCORAH5	TCORAH4	TCORAH3	TCORAH2	TCORAH1	TCORAH0
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

TCORBH—Time constant register BH **H'8B** **Timer H**

Bit	7	6	5	4	3	2	1	0
	TCORBH7	TCORBH6	TCORBH5	TCORBH4	TCORBH3	TCORBH2	TCORBH1	TCORBH0
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

TCNTH—Timer counter H **H'8C** **Timer H**

Bit	7	6	5	4	3	2	1	0
	TCNTH7	TCNTH6	TCNTH5	TCNTH4	TCNTH3	TCNTH2	TCNTH1	TCNTH0
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

PWCR—PWM control register **H'90** **8-bit PWM**

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

Clock select

0	Input clock is ø/16, cycle time is 4096/ø
1	Input clock is ø/8, cycle time is 2048/ø

PWDR0—PWM data register 0 **H'91** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR0_7$	$PWDR0_6$	$PWDR0_5$	$PWDR0_4$	$PWDR0_3$	$PWDR0_2$	$PWDR0_1$	$PWDR0_0$
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

Controls high-level width of PWM_0 output

PWDR1—PWM data register 1 **H'92** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR1_7$	$PWDR1_6$	$PWDR1_5$	$PWDR1_4$	$PWDR1_3$	$PWDR1_2$	$PWDR1_1$	$PWDR1_0$
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

Controls high-level width of PWM_1 output

PWDR2—PWM data register 2 **H'93** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR2_7$	$PWDR2_6$	$PWDR2_5$	$PWDR2_4$	$PWDR2_3$	$PWDR2_2$	$PWDR2_1$	$PWDR2_0$
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

Controls high-level width of PWM_2 output

PWDR3—PWM data register 3 **H'94** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR3_7$	$PWDR3_6$	$PWDR3_5$	$PWDR3_4$	$PWDR3_3$	$PWDR3_2$	$PWDR3_1$	$PWDR3_0$
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

Controls high-level width of PWM_3 output

PWDR4—PWM data register 4 **H'95** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR4_7$	$PWDR4_6$	$PWDR4_5$	$PWDR4_4$	$PWDR4_3$	$PWDR4_2$	$PWDR4_1$	$PWDR4_0$
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

Controls high-level width of PWM_4 output

PWDR5—PWM data register 5 **H'96** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR5_7$	$PWDR5_6$	$PWDR5_5$	$PWDR5_4$	$PWDR5_3$	$PWDR5_2$	$PWDR5_1$	$PWDR5_0$
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

Controls high-level width of PWM_5 output

PWDR6—PWM data register 6 **H'97** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR6_7$	$PWDR6_6$	$PWDR6_5$	$PWDR6_4$	$PWDR6_3$	$PWDR6_2$	$PWDR6_1$	$PWDR6_0$
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

Controls high-level width of PWM_6 output

PWDR7—PWM data register 7 **H'98** **8-bit PWM**

Bit	7	6	5	4	3	2	1	0
	$PWDR7_7$	$PWDR7_6$	$PWDR7_5$	$PWDR7_4$	$PWDR7_3$	$PWDR7_2$	$PWDR7_1$	$PWDR7_0$
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

Controls high-level width of PWM_7 output

TMB2—Timer mode register B2 **H'9C** **Timer B2**

Bit	7	6	5	4	3	2	1	0
	TMB27	—	—	—	—	TMB22	TMB21	TMB20
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 (TMIB2A or TMIB2B), rising or falling edge

---

TCB2—Timer counter B2 **H'9D** **Timer B2**

Bit	7	6	5	4	3	2	1	0
	TCB27	TCB26	TCB25	TCB24	TCB23	TCB22	TCB21	TCB20
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Count value

TLB2—Timer load register B2 **H'9D** **Timer B2**

Bit	7	6	5	4	3	2	1	0
	TLB27	TLB26	TLB25	TLB24	TLB23	TLB22	TLB21	TLB20
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Reload value

TMB3—Timer mode register B3 **H'9E** **Timer B3**

Bit	7	6	5	4	3	2	1	0
	TMB37	—	—	—	—	TMB32	TMB31	TMB30
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	Reserved

TCB3—Timer counter B3 **H'9F** **Timer B3**

Bit	7	6	5	4	3	2	1	0
	TCB37	TCB36	TCB35	TCB34	TCB33	TCB32	TCB31	TCB30
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

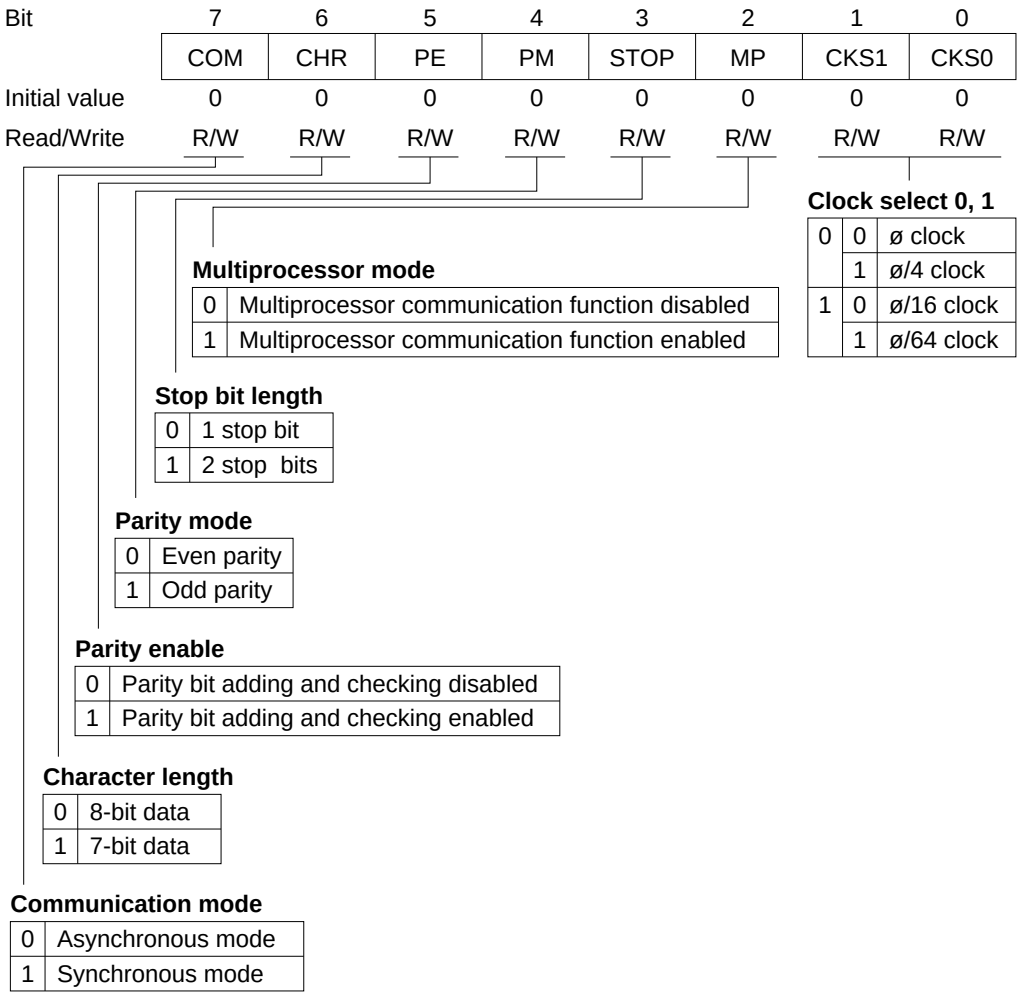
Count value

TLB3—Timer load register B3 **H'9F** **Timer B3**

Bit	7	6	5	4	3	2	1	0
	TLB37	TLB36	TLB35	TLB34	TLB33	TLB32	TLB31	TLB30
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Reload value

SMR—Serial mode register **H'A8** **SCI3**



BRR—Bit rate register **H'A9** **SCI3**

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

SCR3—Serial control register 3 **H'AA** **SCI3**

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Clock enable

Bit 1	Bit 0	Description		
CKE1	CKE0	Communication Mode	Clock Source	SCK_3 Pin Function
0	0	Asynchronous	Internal clock	I/O port
		Synchronous	Internal clock	Serial clock output
	1	Asynchronous	Internal clock	Clock output
		Synchronous	Reserved	Reserved
1	0	Asynchronous	External clock	Clock input
		Synchronous	External clock	Serial clock input
	1	Asynchronous	Reserved	Reserved
		Synchronous	Reserved	Reserved

Transmit end interrupt enable

0	Transmit end interrupt (TEI) disabled
1	Transmit end interrupt (TEI) enabled

Multiprocessor interrupt enable

0	Multiprocessor interrupt request disabled (ordinary receive operation) [Clearing condition] Multiprocessor bit receives a data value of 1
1	Multiprocessor interrupt request enabled Until a multiprocessor bit value of 1 is received, the receive data full interrupt (RXI) and receive error interrupt (ERI) are disabled, and serial status register (SSR) flags RDRF, FER, and OER are not set.

Receive enable

0	Receive operation disabled (RXD is a general I/O port)
1	Receive operation enabled (RXD is the receive data pin)

Transmit enable

0	Transmit operation disabled (TXD is a general I/O port)
1	Transmit operation enabled (TXD is the transmit data pin)

Receive interrupt enable

0	Receive data interrupt request (RXI) and receive error interrupt request (ERI) disabled
1	Receive data interrupt request (RXI) and receive error interrupt request (ERI) enabled

Transmit interrupt enable (TIE)

0	Transmit data empty interrupt request (TXI) disabled
1	Transmit data empty interrupt request (TXI) enabled

TDR—Transmit data register **H'AB** **SCI3**

Bit	7	6	5	4	3	2	1	0
	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Data to be transferred to TSR

SSR—Serial status register **H'AC** **SCI3**

Bit	7	6	5	4	3	2	1	0
	TDRE	RDRF	OER	FER	PER	TEND	MPBR	MPBT
Initial value	1	0	0	0	0	1	0	0
Read/Write	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Multiprocessor bit receive

0	Indicates reception of data in which the multiprocessor bit is 0
1	Indicates reception of data in which the multiprocessor bit is 1

Multiprocessor bit transmit

0	The multiprocessor bit in transmit data is 0
1	The multiprocessor bit in transmit data is 1

Transmit end

0	Indicates that transmission is in progress [Clearing conditions] After reading TDRE = 1, cleared by writing 0 to TDRE. When data is written to TDR by an instruction.
1	Indicates that a transmission has ended [Setting conditions] When bit TE in serial control register 3 (SCR3) is 0. If TDRE is set to 1 when the last bit of a transmitted character is sent.

Parity error

0	Indicates that data receiving is in progress or has been completed [Clearing conditions] After reading PER = 1, cleared by writing 0
1	Indicates that a parity error occurred in data receiving [Setting conditions] When the sum of 1s in received data plus the parity bit does not match the parity mode bit (PM) setting in the serial mode register (SMR)

Framing error

0	Indicates that data receiving is in progress or has been completed [Clearing conditions] After reading FER = 1, cleared by writing 0
1	Indicates that a framing error occurred in data receiving [Setting conditions] The stop bit at the end of receive data is checked and found to be 0

Overrun error

0	Indicates that data receiving is in progress or has been completed [Clearing conditions] After reading OER = 1, cleared by writing 0
1	Indicates that an overrun error occurred in data receiving [Setting conditions] When data receiving is completed while RDRF is set to 1

Receive data register full

0	Indicates there is no receive data in RDR [Clearing conditions] After reading RDRF = 1, cleared by writing 0. When data is read from RDR by an instruction.
1	Indicates that there is receive data in RDR [Setting conditions] When receiving ends normally, with receive data transferred from RSR to RDR

Transmit data register empty

0	Indicates that transmit data written to TDR has not been transferred to TSR [Clearing conditions] After reading TDRE = 1, cleared by writing 0. When data is written to TDR by an instruction.
1	Indicates that no transmit data has been written to TDR, or the transmit data written to TDR has been transferred to TSR [Setting conditions] When bit TE in serial control register 3 (SCR3) is 0. When data is transferred from TDR to TSR.

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

RDR—Receive data register **H'AD** **SCI3**

Bit	7	6	5	4	3	2	1	0
	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

TMA—Timer mode register A **H'B0** **Timer A**

Bit	7	6	5	4	3	2	1	0
	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'B1** **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'B2** **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 (TMIB1): Rising or falling edge

TCB1—Timer counter B1 **H'B3** **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'B3** **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

TMC—Timer mode register C **H'B4** **Timer C**

Bit	7	6	5	4	3	2	1	0
	TMC7	TMC6	TMC5	—	—	TMC2	TMC1	TMC0
Initial value	0	0	0	1	1	0	0	0
Read/Write	R/W	R/W	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/64
1	0	0	Internal clock: ø/16
		1	Internal clock: ø/4
	1	0	Internal clock: $ø_W$/4
		1	External event (TMIC): Rising or falling edge

Counter up/down control

0	0	TCC is an up-counter
	1	TCC is a down-counter
1	*	TCC up/down control is determined by input at pin UD. TCC is a down-counter if the UD input is high, and an up-counter if the UD input is low.

Note: * Don't care

TCC—Timer counter C **H'B5** **Timer C**

Bit	7	6	5	4	3	2	1	0
	TCC7	TCC6	TCC5	TCC4	TCC3	TCC2	TCC1	TCC0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R	R	R	R	R	R	R	R

Count value

TLC—Timer load register C **H'B5** **Timer C**

Bit	7	6	5	4	3	2	1	0
	TLC7	TLC6	TLC5	TLC4	TLC3	TLC2	TLC1	TLC0
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Reload value

TCRF—Timer control register F **H'B6** **Timer F**

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

Toggle output level H

0	Low level
1	High level

Clock select L

0	*	*	External event (TMIF): Rising or falling edge
1	0	0	Internal clock: ø/32
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	Internal clock: ø/2

Toggle output level L

0	Low level
1	High level

Clock select H

0	*	*	16-bit mode selected. TCFL overflow signals are counted.
1	0	0	Internal clock: ø/32
		1	Internal clock: ø/16
	1	0	Internal clock: ø/4
		1	Internal clock: ø/2

Note: * Don’t care

TCSRF—Timer control/status register F **H'B7** **Timer F**

Bit	7	6	5	4	3	2	1	0
	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/(W)*	R/(W)*	R/W	R/W	R/(W)*	R/(W)*	R/W	R/W

Timer overflow interrupt enable L

0	TCFL overflow interrupt disabled
1	TCFL overflow interrupt enabled

Compare match flag L

0	[Clearing condition] After reading CMFL = 1, cleared by writing 0 to CMFL
1	[Setting condition] When the TCFL value matches the OCRFL value

Timer overflow flag L

0	[Clearing condition] After reading OVFL = 1, cleared by writing 0 to OVFL
1	[Setting condition] When the value of TCFL goes from H'FF to H'00

Counter clear H

0	16-bit mode: TCF clearing by compare match disabled 8-bit mode: TCFH clearing by compare match disabled
1	16-bit mode: TCF clearing by compare match enabled 8-bit mode: TCFH clearing by compare match enabled

Timer overflow interrupt enable H

0	TCFH overflow interrupt disabled
1	TCFH overflow interrupt enabled

Counter clear L

0	TCFL clearing by compare match disabled
1	TCFL clearing by compare match enabled

Compare match flag H

0	[Clearing condition] After reading CMFH = 1, cleared by writing 0 to CMFH
1	[Setting condition] When the TCFH value matches the OCRFH value

Timer overflow flag H

0	[Clearing condition] After reading OVFH = 1, cleared by writing 0 to OVFH
1	[Setting condition] 16-bit mode: When the value of TCF goes from H'FFFF to H'0000 8-bit mode: When the value of TCFH goes from H'FF to H'00

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

TCFH—8-bit timer counter FH **H'B8** **Timer F**

Bit	7	6	5	4	3	2	1	0
	TCFH7	TCFH6	TCFH5	TCFH4	TCFH3	TCFH2	TCFH1	TCFH0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

TCFL—8-bit timer counter FL **H'B9** **Timer F**

Bit	7	6	5	4	3	2	1	0
	TCFL7	TCFL6	TCFL5	TCFL4	TCFL3	TCFL2	TCFL1	TCFL0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Count value

OCRFH—Output compare register FH **H'BA** **Timer F**

Bit	7	6	5	4	3	2	1	0
	OCRFH7	OCRFH6	OCRFH5	OCRFH4	OCRFH3	OCRFH2	OCRFH1	OCRFH0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

OCRFL—Output compare register FL **H'BB** **Timer F**

Bit	7	6	5	4	3	2	1	0
	OCRFL7	OCRFL6	OCRFL5	OCRFL4	OCRFL3	OCRFL2	OCRFL1	OCRFL0
Initial value	1	1	1	1	1	1	1	1
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

Clock select

0	0	Internal clock: ø/64
	1	Internal clock: ø/32
1	0	Internal clock: ø/2
	1	Internal clock: $ø_W$/2

Counter clear

0	0	TCG is not cleared
	1	TCG is cleared at the falling edge of the input capture signal
1	0	TCG is cleared at the rising edge of the input capture signal
	1	TCG is cleared at both edges of the input capture signal

Input capture interrupt edge select

0	Interrupts are requested at the rising edge of the input capture signal
1	Interrupts are requested at the falling edge of the input capture signal

Timer overflow interrupt enable

0	TCG overflow interrupt disabled
1	TCG overflow interrupt enabled

Timer overflow flag L

0	[Clearing condition] After reading OVFL = 1, cleared by writing 0 to OVFL
1	[Setting condition] When the value of TCG goes from H'FF to H'00

Timer overflow flag H

0	[Clearing condition] After reading OVFH = 1, cleared by writing 0 to OVFH
1	[Setting condition] When the value of TCG goes from H'FF to H'00

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

ICRGF—Input capture register GF **H'BD** **Timer G**

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

ICRGR—Input capture register GR **H'BE** **Timer G**

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

AMR—A/D mode register **H'C4** **A/D converter**

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	AN_8
			1	AN_9
		1	0	AN_{10}
			1	AN_{11}

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'C5** **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

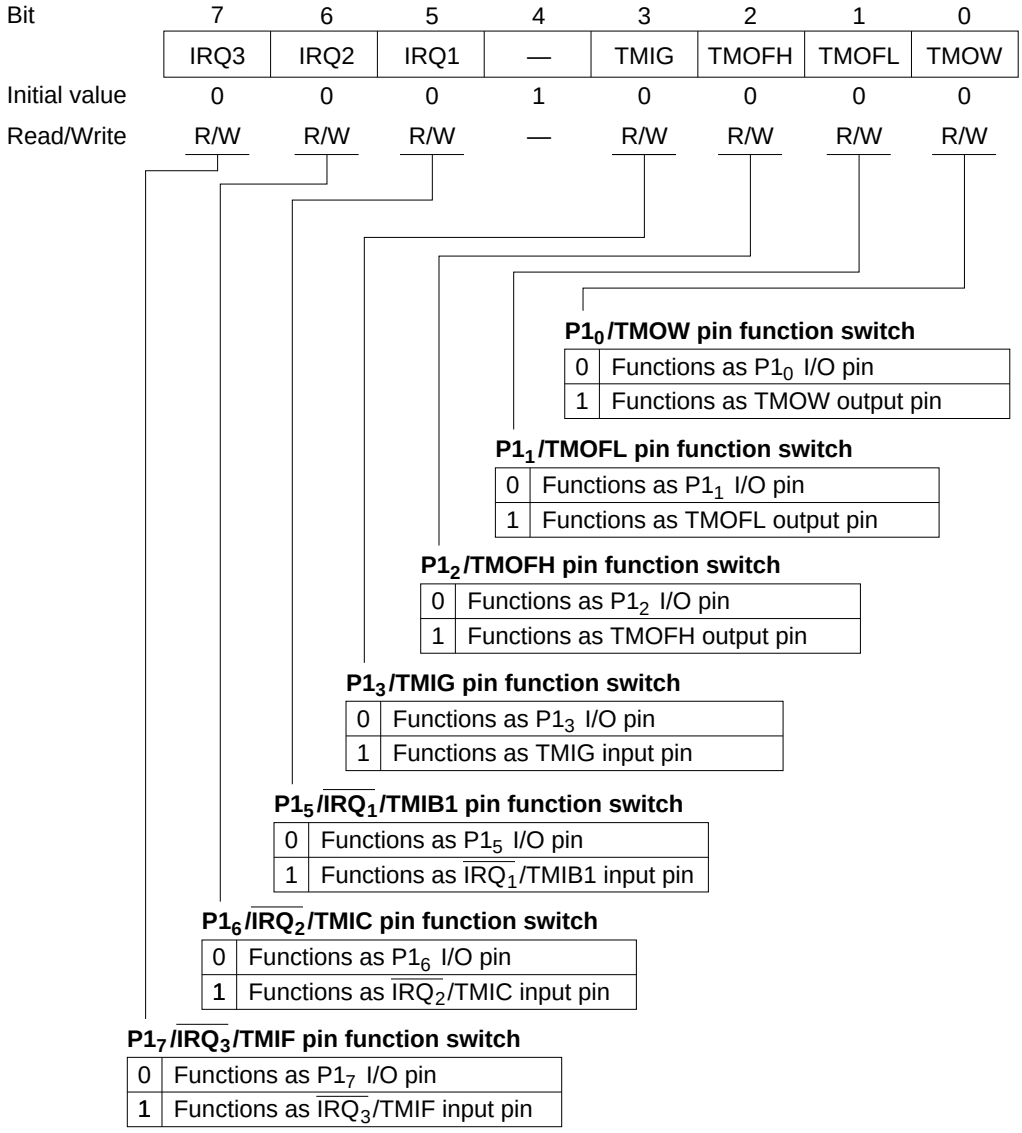
ADSR—A/D start register **H'C6** **A/D converter**

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

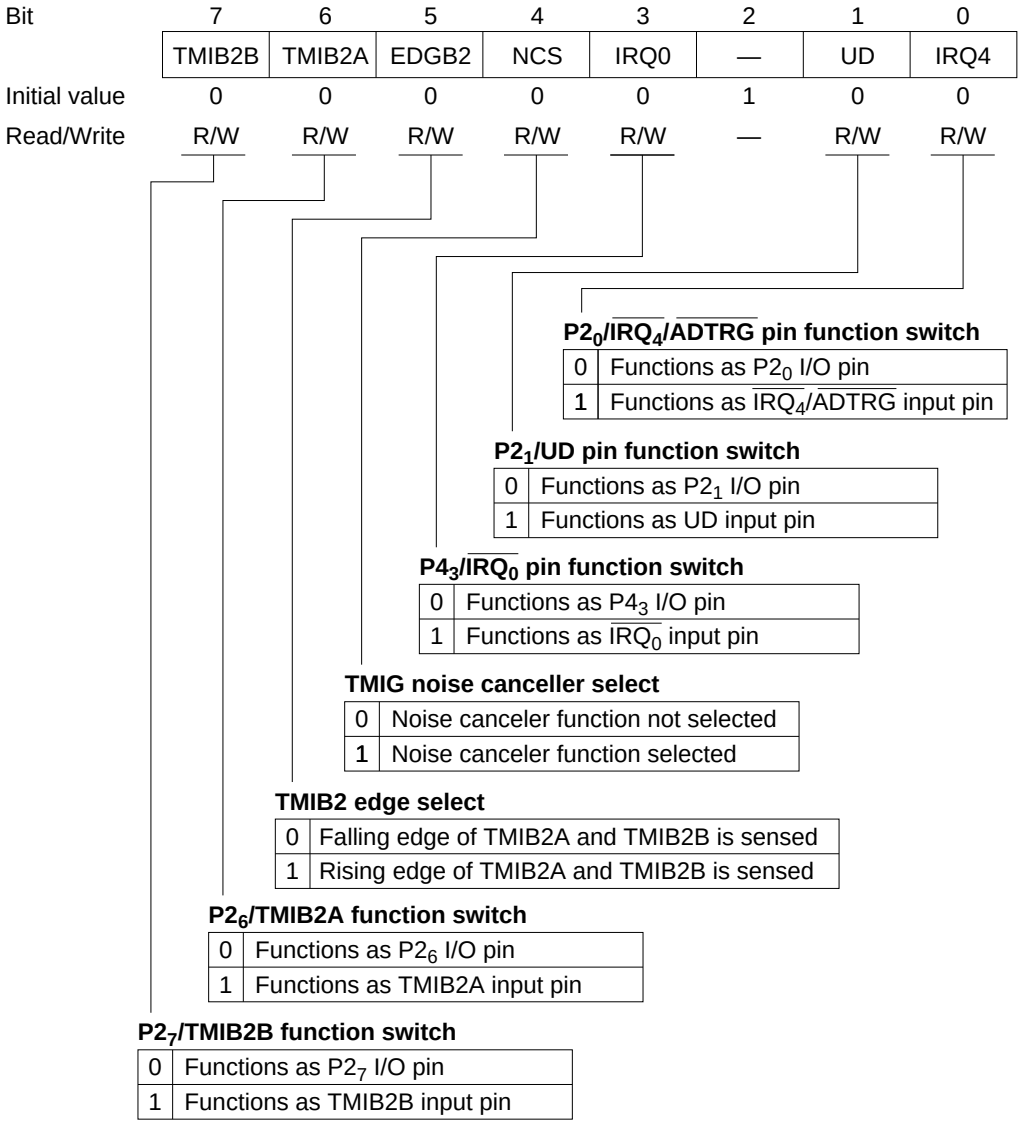
A/D status flag

0	[Read] Indicates the completion of A/D conversion [Write] Stops A/D conversion
1	[Read] Indicates A/D conversion in progress [Write] Starts A/D conversion

PMR1—Port mode register 1 **H'C8** **I/O ports**



PMR2—Port mode register 2 **H'C9** **I/O ports**



PMR3—Port mode register 3 **H'CA** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	PWM7	PWM6	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0
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

$P3_n/PWM_n$ pin function switch

0	Functions as $P3_n$ I/O pin
1	Functions as PWM_n output pin

(n = 7 to 0)

PMR5—Port mode register 5 **H'CC** **I/O ports**

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

$P5_n/\overline{WKP_n}$ pin function switch

0	Functions as $P5_n$ I/O pin
1	Functions as $\overline{WKP_n}$ input pin

(n = 7 to 0)

PDR1—Port data register 1 **H'D4** **I/O ports**

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

PDR2—Port data register 2 **H'D5** **I/O ports**

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

PDR3—Port data register 3 **H'D6** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$P3_7$	$P3_6$	$P3_5$	$P3_4$	$P3_3$	$P3_2$	$P3_1$	$P3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PDR4—Port data register 4 **H'D7** **I/O ports**

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

PDR5—Port data register 5 **H'D8** **I/O ports**

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

PDR6—Port data register 6 **H'D9** **I/O ports**

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

PDR7—Port data register 7 **H'DA** **I/O ports**

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

PDR8—Port data register 8 **H'DB** **I/O ports**

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

PDR9—Port data register 9 **H'DC** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$P9_7$	$P9_6$	$P9_5$	$P9_4$	$P9_3$	$P9_2$	$P9_1$	—
Initial value	0	0	0	1	1	1	1	1
Read/Write	R/W	R/W	R/W	—	—	—	—	—

PDRA—Port data register A **H'DD** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	PA_6	PA_5	PA_4	PA_3	PA_2	PA_1	PA_0
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PDRB—Port data register B **H'DE** **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

PDRC—Port data register C **H'DF** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	PC_3	PC_2	PC_1	PC_0
Initial value								
Read/Write	—	—	—	—	R	R	R	R

PUCR1—Port pull-up control register 1 **H'E0** **I/O ports**

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

PUCR7—Port pull-up control register 7 **H'E1** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PUCR7_7$	$PUCR7_6$	$PUCR7_5$	$PUCR7_4$	$PUCR7_3$	$PUCR7_2$	$PUCR7_1$	$PUCR7_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PUCR5—Port pull-up control register 5 **H'E2** **I/O ports**

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

PUCR6—Port pull-up control register 6 **H'E3** **I/O ports**

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

PCR1—Port control register 1 **H'E4** **I/O ports**

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

Port 1 input/output select

0	Input pin
1	Output pin

PCR2—Port control register 2 **H'E5** **I/O ports**

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

Port 2 input/output select

0	Input pin
1	Output pin

PCR3—Port control register 3 **H'E6** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR3_7$	$PCR3_6$	$PCR3_5$	$PCR3_4$	$PCR3_3$	$PCR3_2$	$PCR3_1$	$PCR3_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 3 input/output select

0	Input pin
1	Output pin

PCR4—Port control register 4 **H'E7** **I/O ports**

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

Port 4 input/output select

0	Input pin
1	Output pin

PCR5—Port control register 5 **H'E8** **I/O ports**

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

Port 5 input/output select

0	Input pin
1	Output pin

PCR6—Port control register 6 **H'E9** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	PCR6$_7$	PCR6$_6$	PCR6$_5$	PCR6$_4$	PCR6$_3$	PCR6$_2$	PCR6$_1$	PCR6$_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 6 input/output select

0	Input pin
1	Output pin

PCR7—Port control register 7 **H'EA** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	PCR7$_7$	PCR7$_6$	PCR7$_5$	PCR7$_4$	PCR7$_3$	PCR7$_2$	PCR7$_1$	PCR7$_0$
Initial value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W

Port 7 input/output select

0	Input pin
1	Output pin

PCR8—Port control register 8 **H'EB** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	PCR8$_7$	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'EC** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	$PCR9_7$	$PCR9_6$	$PCR9_5$	—	—	—	—	—
Initial value	0	0	0	1	1	1	1	1
Read/Write	W	W	W	—	—	—	—	—

Port 9 input/output select

0	Input pin
1	Output pin

PCRA—Port control register A **H'ED** **I/O ports**

Bit	7	6	5	4	3	2	1	0
	—	$PCRA_6$	$PCRA_5$	$PCRA_4$	$PCRA_3$	$PCRA_2$	$PCRA_1$	$PCRA_0$
Initial value	1	0	0	0	0	0	0	0
Read/Write	—	W	W	W	W	W	W	W

Port A input/output select

0	Input pin
1	Output pin

SYSCR1—System control register 1 **H'F0** **System control**

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

Low speed on flag

0	The CPU operates on the system clock (ø)
1	The CPU operates on the subclock ($ø_{SUB}$)

Standby timer select 2 to 0

0	0	0	Wait time = 8,192 states
		1	Wait time = 16,384 states
	1	0	Wait time = 32,768 states
		1	Wait time = 65,536 states
1	*	*	Wait time = 131,072 states

Software standby

0	When a SLEEP instruction is executed in active mode, a transition is made to sleep mode. When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode.
1	When a SLEEP instruction is executed in active mode, a transition is made to standby mode or watch mode. When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode.

Note: * Don’t care

SYSCR2—System control register 2 **H'F1** **System control**

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

Medium speed on flag

0	Operates in active (high-speed) mode
1	Operates in active (medium-speed) mode

Subactive mode clock select

0	0	$ø_W/8$
	1	$ø_W/4$
1	*	$ø_W/2$

Direct transfer on flag

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

Noise elimination sampling frequency select

0	Sampling rate is $ø_{OSC}/16$
1	Sampling rate is $ø_{OSC}/4$

Note: * Don’t care

IEGR—IRQ edge select register **H'F2** **System control**

Bit	7	6	5	4	3	2	1	0
	NMIEG	—	—	IEG4	IEG3	IEG2	IEG1	IEG0
Initial value	0	1	1	0	0	0	0	0
Read/Write	R/W	—	—	R/W	R/W	R/W	R/W	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 $\overline{IRQ_1}$/TMIB pin input is detected
1	Rising edge of $\overline{IRQ_1}$/TMIB pin input is detected

IRQ_2 edge select

0	Falling edge of $\overline{IRQ_2}$/TMIC pin input is detected
1	Rising edge of $\overline{IRQ_2}$/TMIC pin input is detected

IRQ_3 edge select

0	Falling edge of $\overline{IRQ_3}$/TMIF pin input is detected
1	Rising edge of $\overline{IRQ_3}$/TMIF pin input is detected

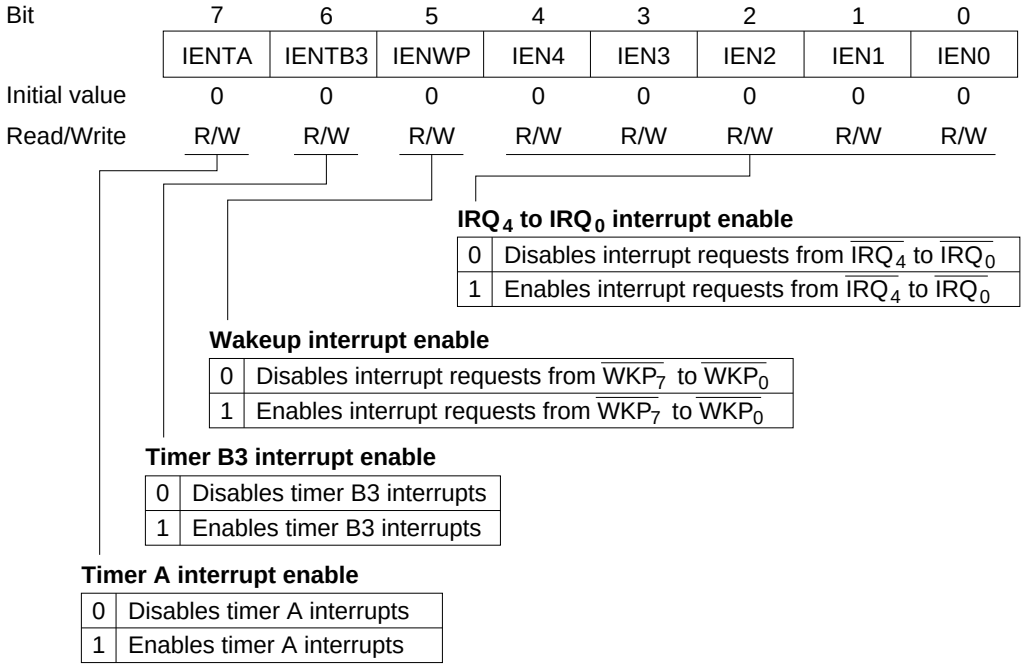
IRQ_4 edge select

0	Falling edge of $\overline{IRQ_4}$/$\overline{ADTRG}$ pin input is detected
1	Rising edge of $\overline{IRQ_4}$/$\overline{ADTRG}$ pin input is detected

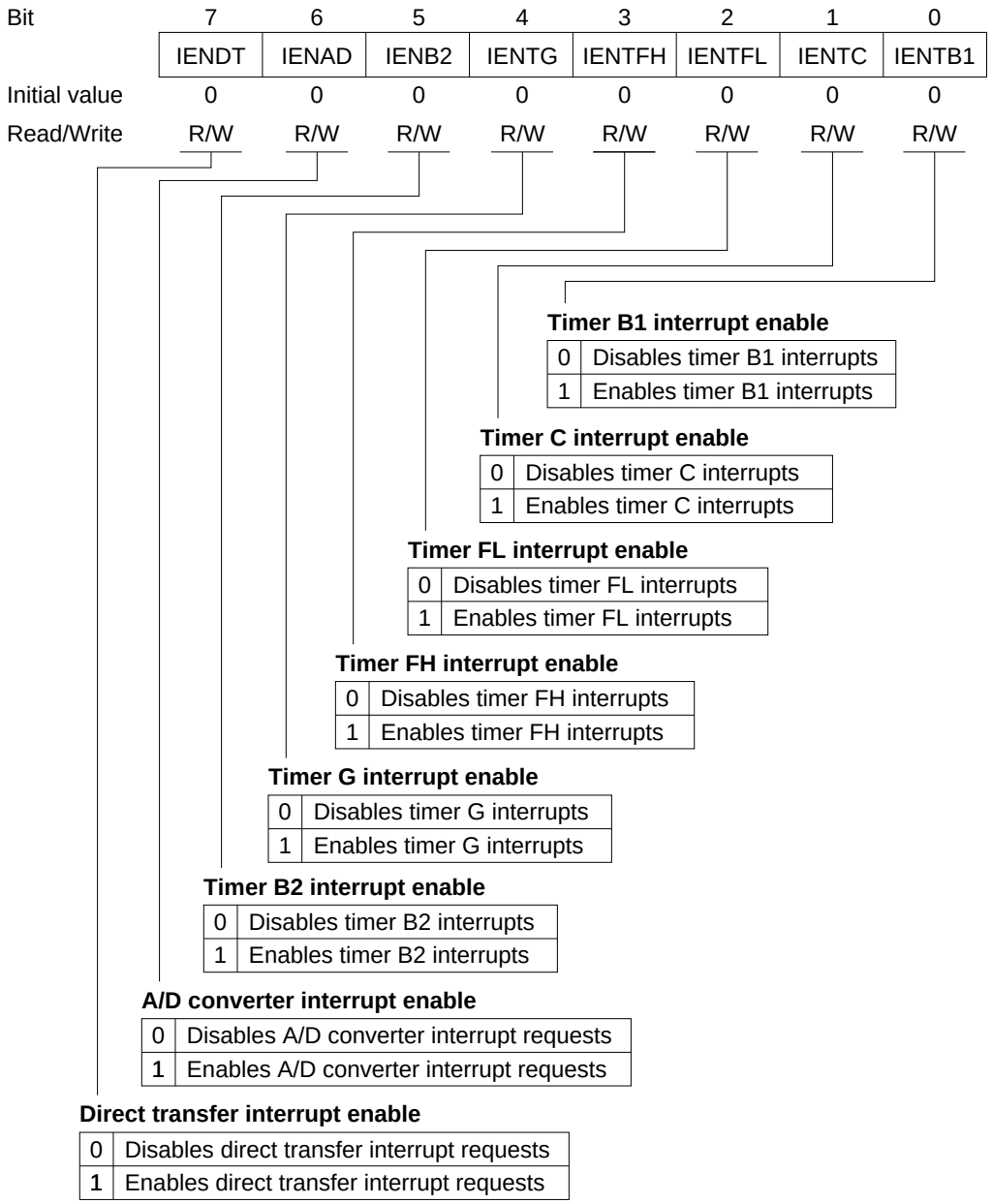
NMI edge select

0	Falling edge of $\overline{NMI}$ pin input is detected
1	Rising edge of $\overline{NMI}$ pin input is detected

IENR1—Interrupt enable register 1 **H'F3** **System control**



IENR2—Interrupt enable register 2 **H'F4** **System control**



IRR1—Interrupt request register 1 **H'F6** **System control**

Bit	7	6	5	4	3	2	1	0
	IRRTA	IRRTB3	—	IRRI4	IRRI3	IRRI2	IRRI1	IRRI0
Initial value	0	0	1	0	0	0	0	0
Read/Write	R/W*	R/W*	—	R/W*	R/W*	R/W*	R/W*	R/W*

IRQ_4 to IRQ_0 interrupt request flag

0	[Clearing condition] When IRRI4 = 1, it is cleared by writing 0; the same applies to IRRI3 to IRRI0
1	[Setting condition] When pin $\overline{IRQ_4}$ is set to interrupt input and the designated signal edge is detected; the same applies to IRRI3 to IRRI0

Timer B3 interrupt request flag

0	[Clearing condition] When IRRTB3 = 1, it is cleared by writing 0
1	[Setting condition] When timer counter B3 overflows from H'FF to H'00

Timer A interrupt request flag

0	[Clearing condition] When IRRTA = 1, it is cleared by writing 0
1	[Setting condition] When the timer A counter overflows from H'FF to H'00

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

IRR2—Interrupt request register 2 **H'F7** **System control**

Bit	7	6	5	4	3	2	1	0
	IRRDT	IRRAD	IRRTB2	IRRTG	IRRTFH	IRRTFL	IRRTC	IRRTB1
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*

Timer B1 interrupt request flag

0	[Clearing condition]	When IRRTB1 = 1, it is cleared by writing 0
1	[Setting condition]	When the timer B1 counter overflows from H'FF to H'00

Timer C interrupt request flag

0	[Clearing condition]	When IRRTC = 1, it is cleared by writing 0
1	[Setting condition]	When the timer C counter overflows from H'FF to H'00 or underflows from H'00 to H'FF

Timer FL interrupt request flag

0	[Clearing condition]	When IRRTFL = 1, it is cleared by writing 0
1	[Setting condition]	When counter FL matches output compare register FL in 8-bit mode

Timer FH interrupt request flag

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

Timer G interrupt request flag

0	[Clearing condition]	When IRRTG = 1, it is cleared by writing 0
1	[Setting condition]	When pin TMIG is set to TMIG input and the designated signal edge is detected

Timer B2 interrupt request flag

0	[Clearing condition]	When IRRTB2 = 1, it is cleared by writing 0
1	[Setting condition]	When the timer B2 counter value overflows from H'FF to H'00

A/D converter interrupt request flag

0	[Clearing condition]	When IRRAD = 1, it is cleared by writing 0
1	[Setting condition]	When A/D conversion is completed and ADSF is reset

Direct transfer interrupt request flag

0	[Clearing condition]	When IRRDT = 1, it is cleared by writing 0
1	[Setting condition]	A SLEEP instruction is executed when DTON = 1 and a direct transfer is made

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

IWPR—Wakeup interrupt request register **H'F9** **System control**

Bit	7	6	5	4	3	2	1	0
	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0
Initial value	0	0	0	0	0	0	0	0
Read/Write	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*

Wakeup interrupt request flag

0	[Clearing condition] When IWPFn = 1, it is cleared by writing 0
1	[Setting condition] When pin $\overline{WKP}_n$ is designed for wakeup input and a falling edge is input

(n = 7 to 0)

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

Appendix C I/O Port Block Diagrams

C.1 Schematic Diagram of Port 1

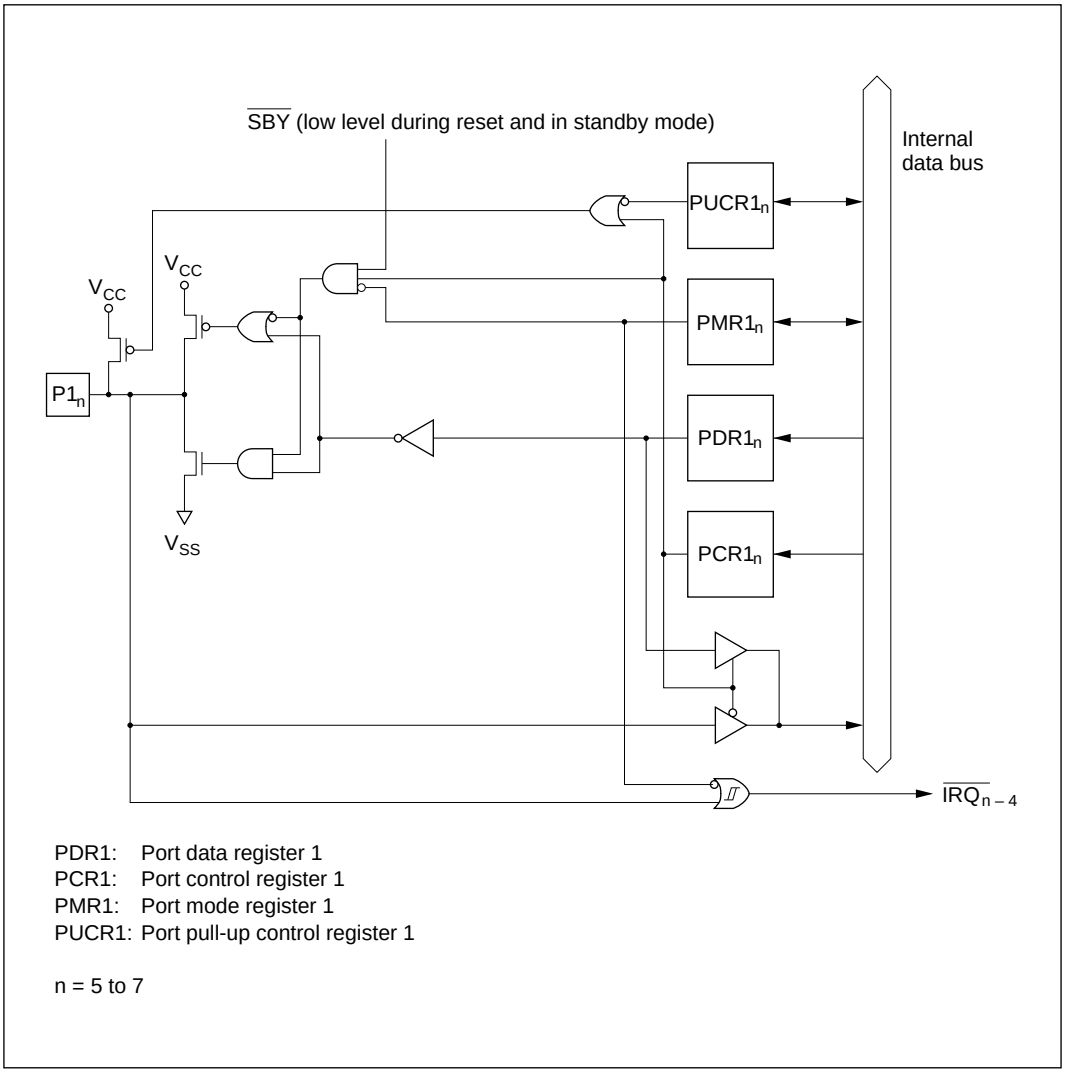


Figure C-1 (a) Port 1 Block Diagram (Pins $P1_7$ to $P1_5$)

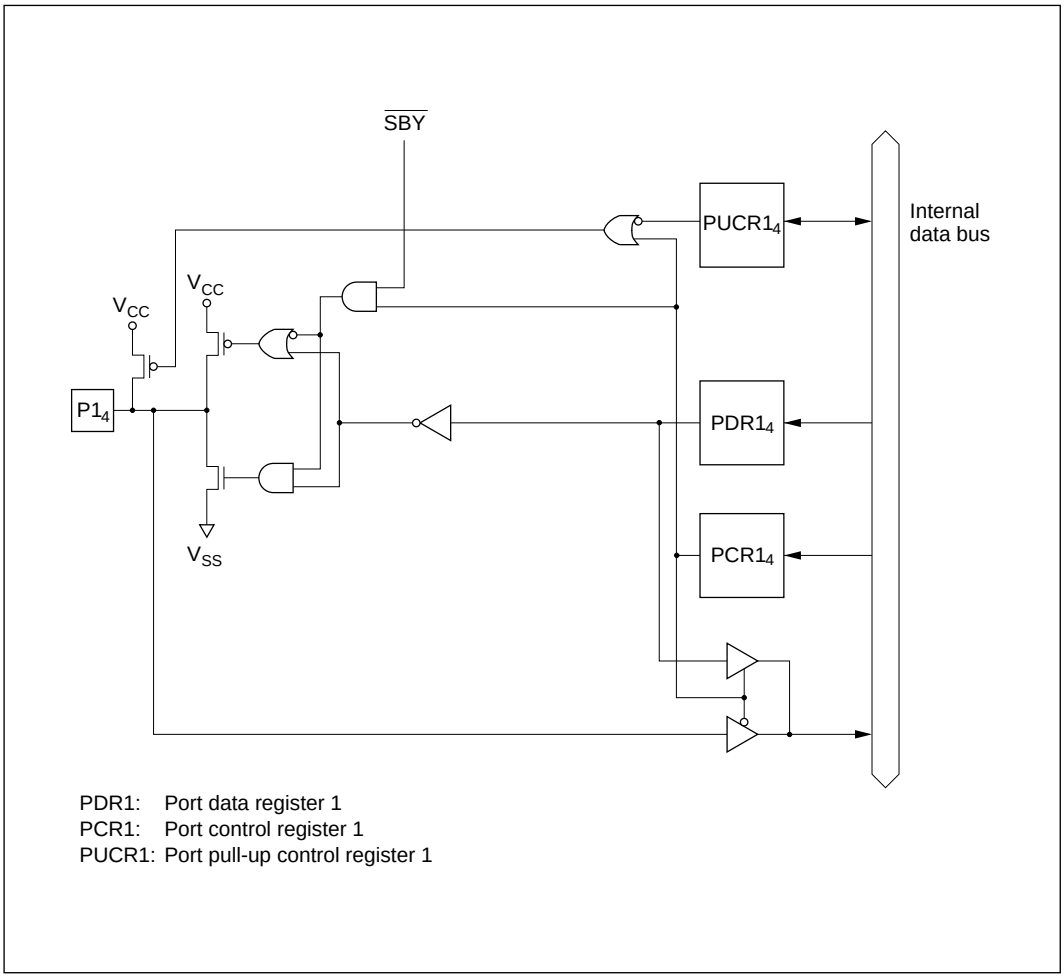


Figure C-1 (b) Port 1 Block Diagram (Pin $P1_4$)

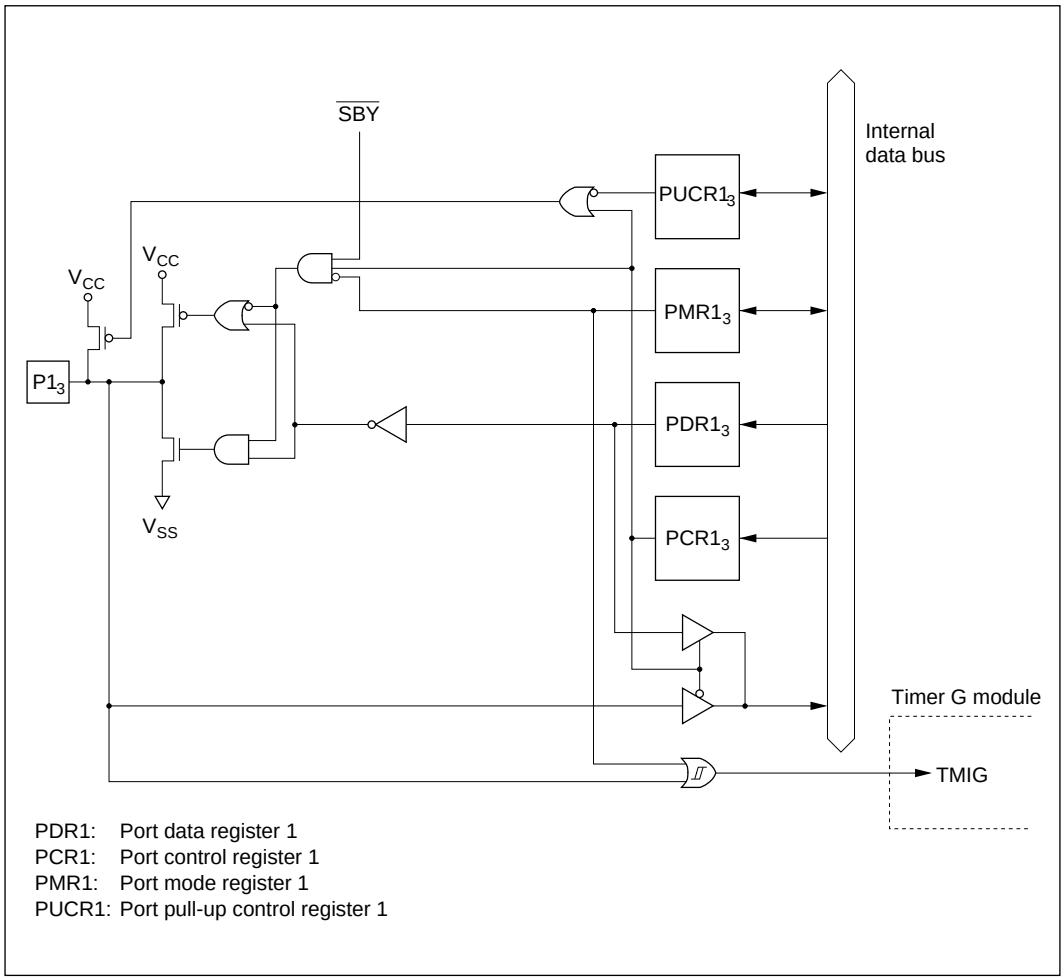


Figure C-1 (c) Port 1 Block Diagram (Pin $P1_3$)

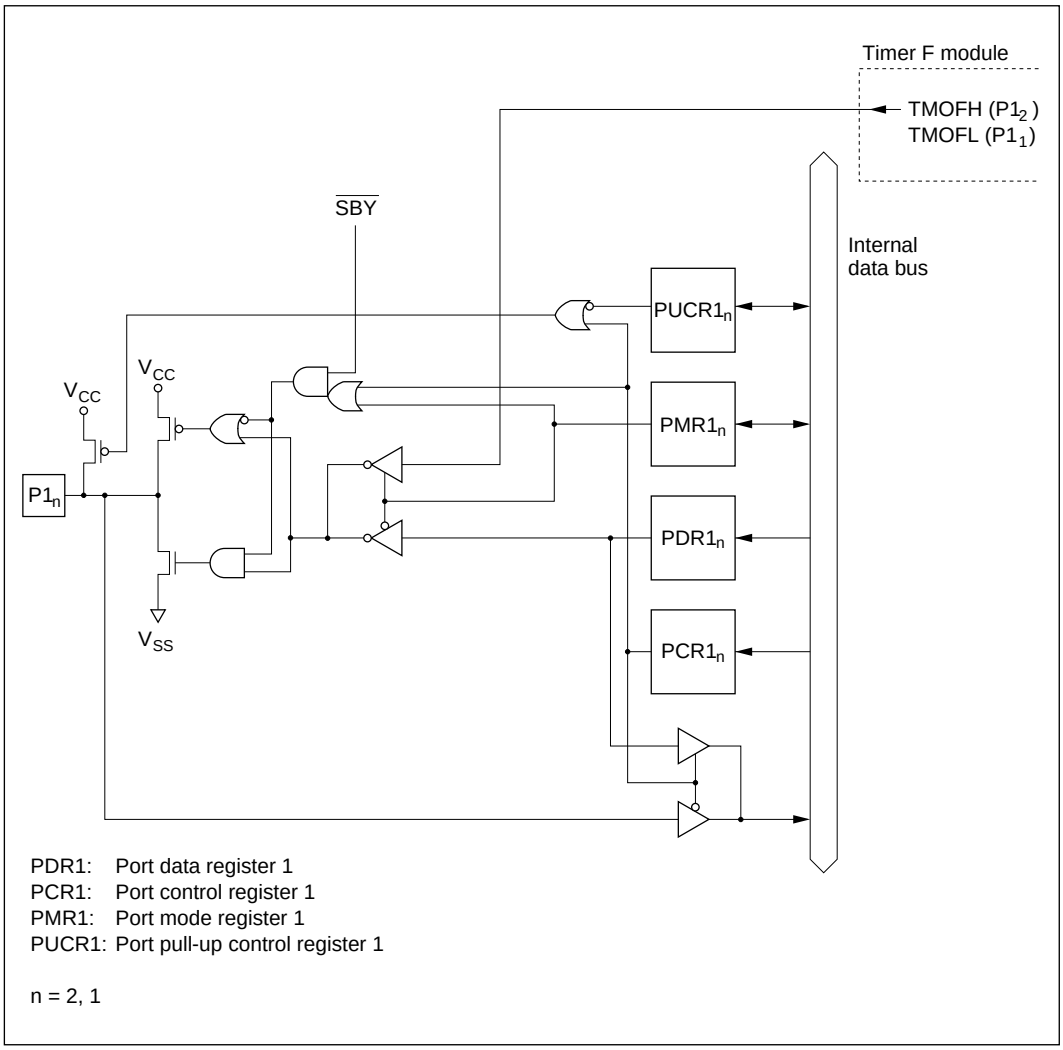


Figure C-1 (d) Port 1 Block Diagram (Pins $P1_2$ and $P1_1$)

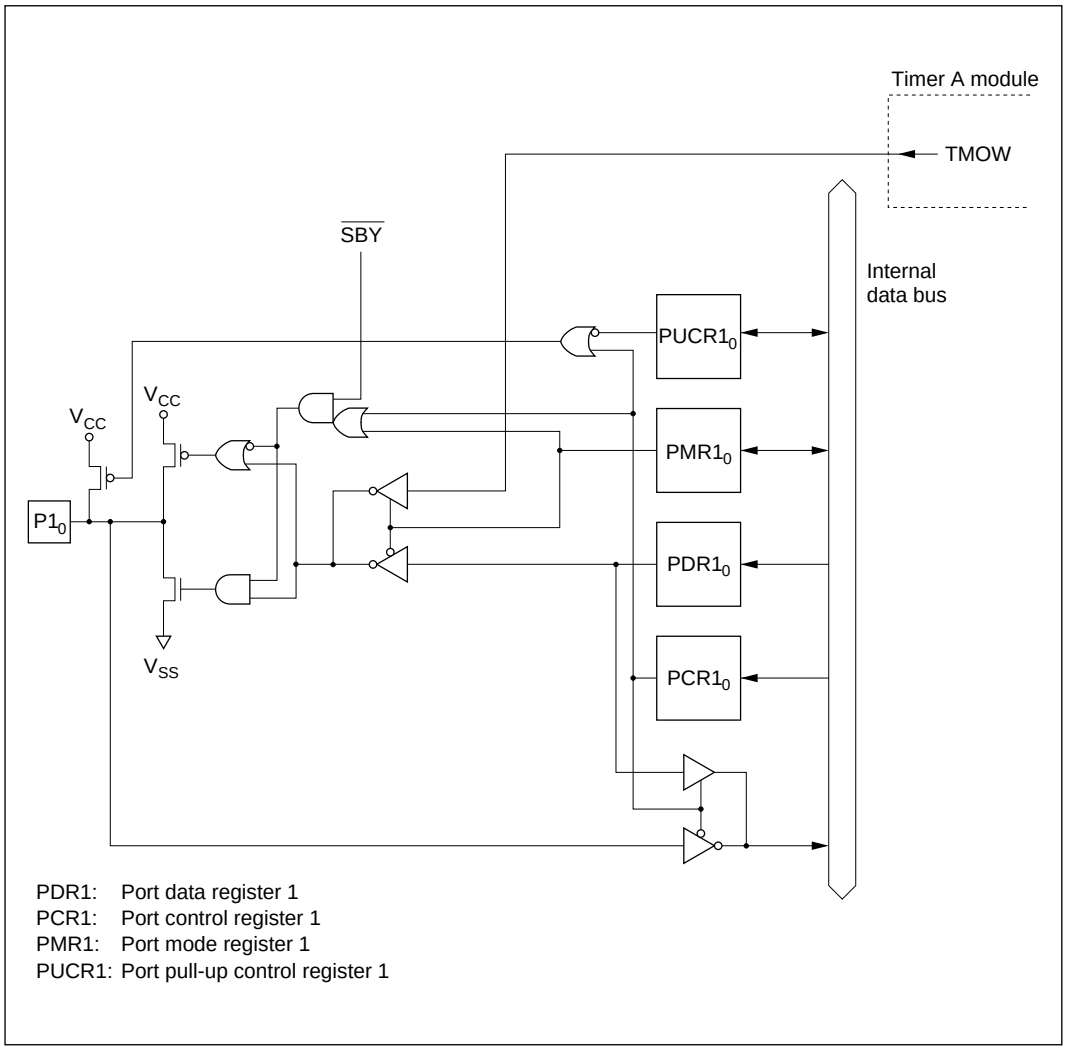


Figure C-1 (e) Port 1 Block Diagram (Pin $P1_0$)

C.2 Schematic Diagrams of Port 2

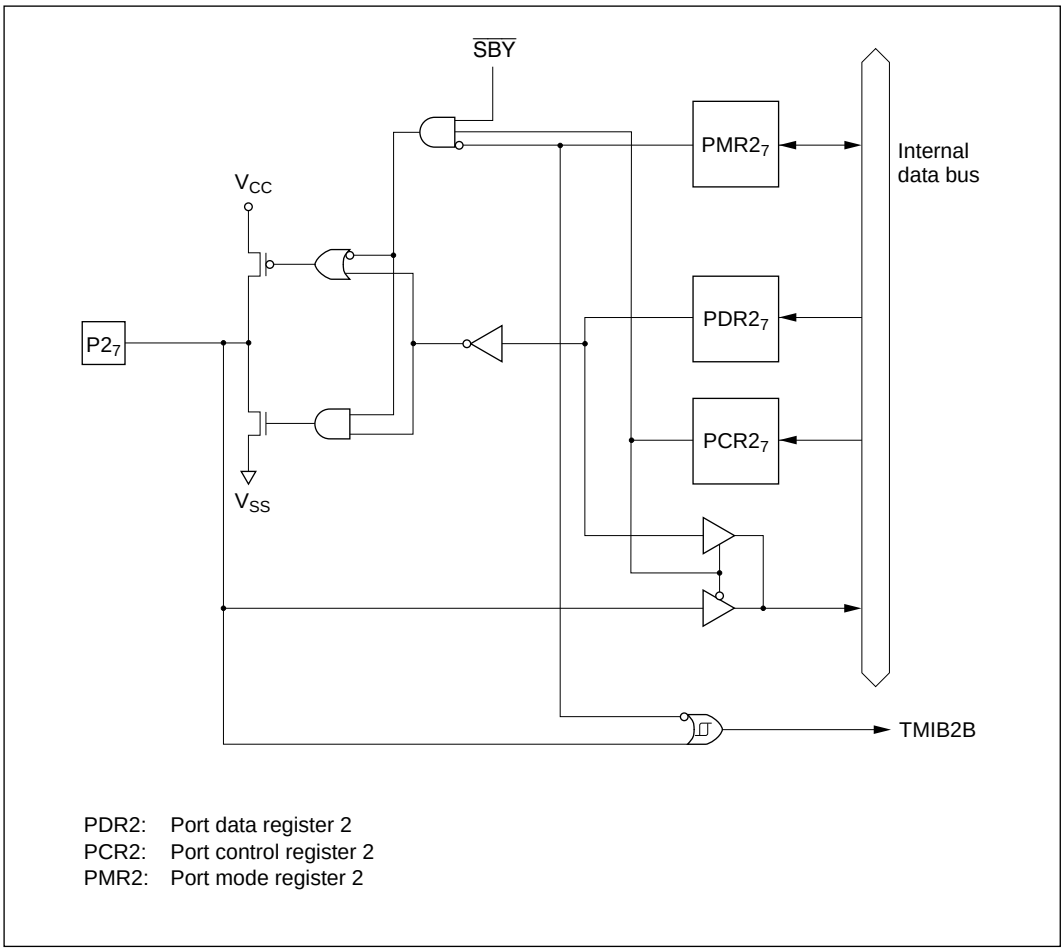
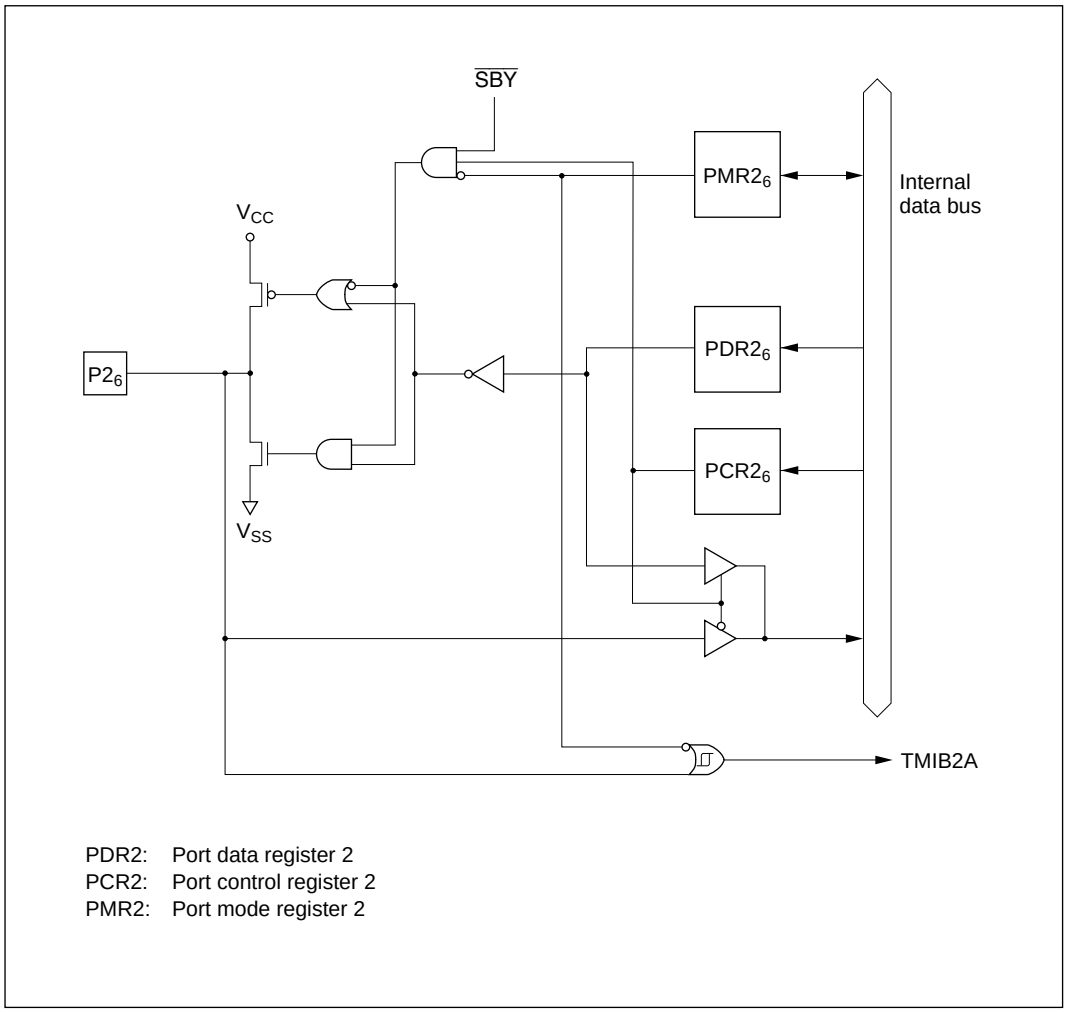


Figure C-2 (a) Port 2 Block Diagram (Pin $P2_7$)



PDR2: Port data register 2
PCR2: Port control register 2
PMR2: Port mode register 2

Figure C-2 (b) Port 2 Block Diagram (Pin $P2_6$)

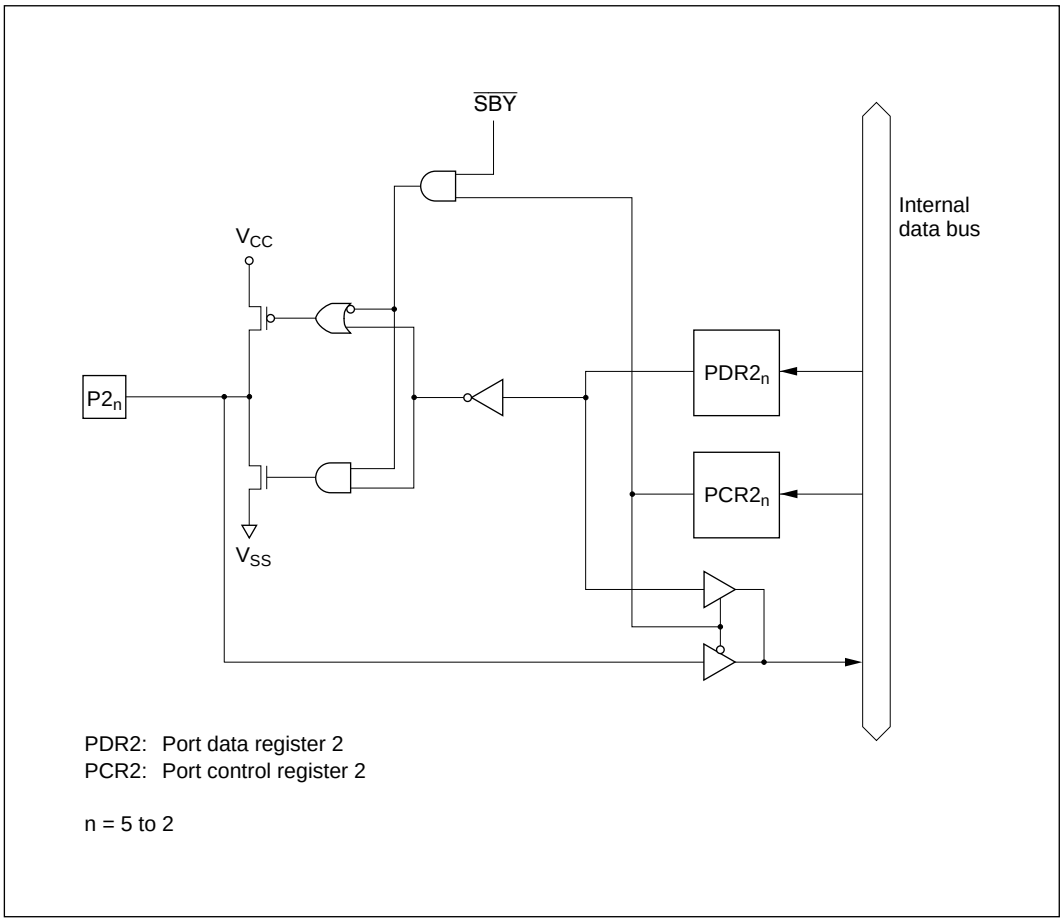


Figure C-2 (c) Port 2 Block Diagram (Pins $P2_5$ to $P2_2$)

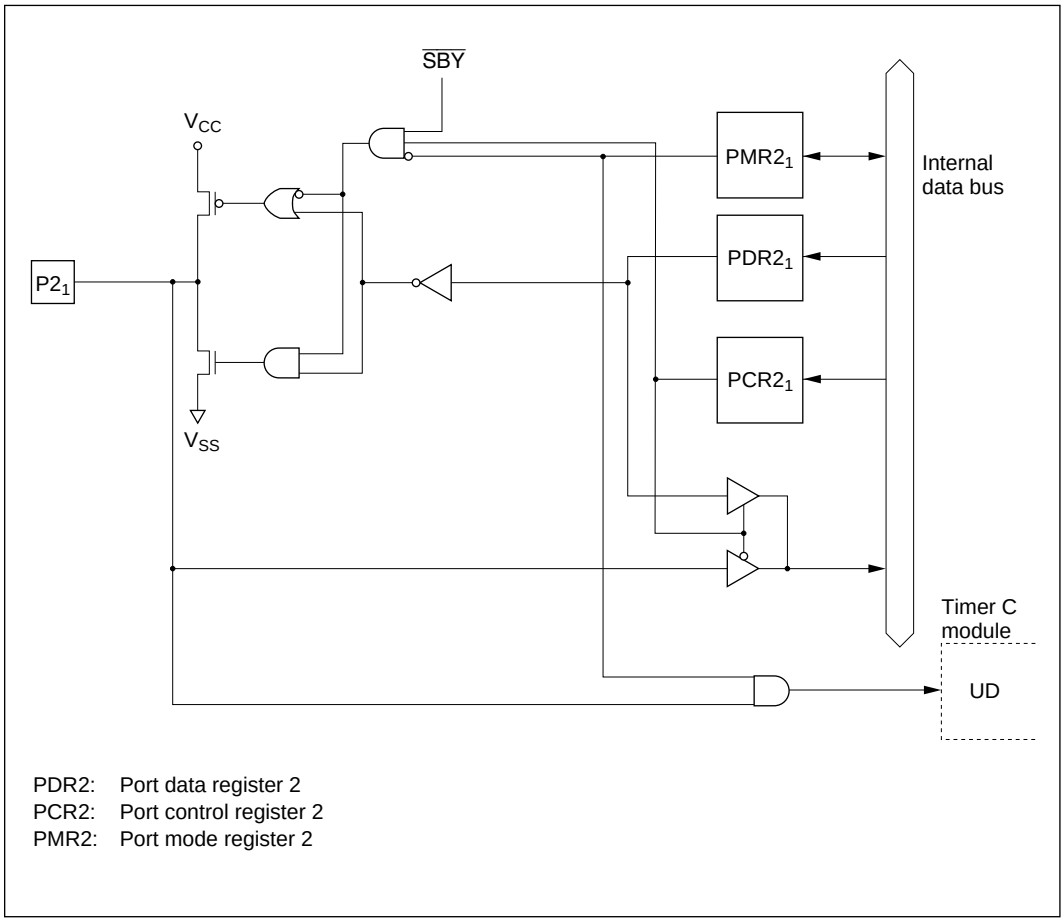
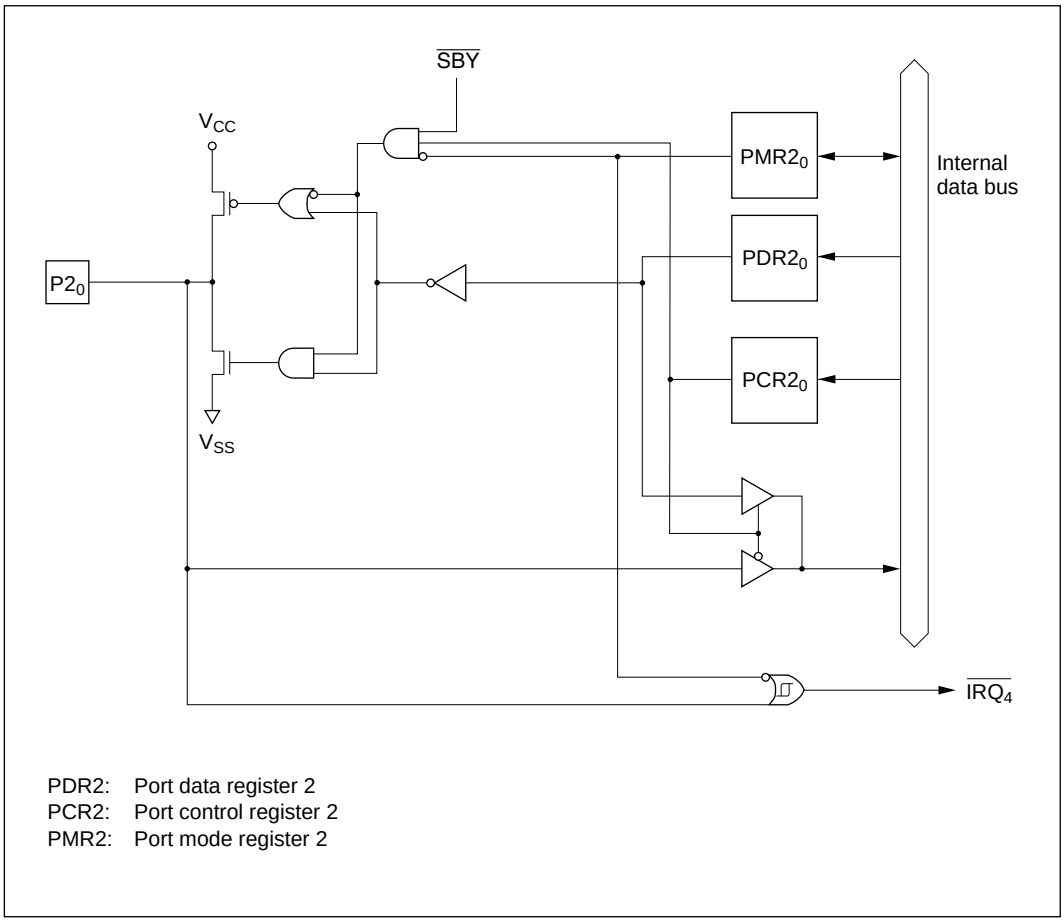


Figure C-2 (d) Port 2 Block Diagram (Pin $P2_1$)



PDR2: Port data register 2
PCR2: Port control register 2
PMR2: Port mode register 2

Figure C-2 (e) Port 2 Block Diagram (Pin $P2_0$)

C.3 Schematic Diagram of Port 3

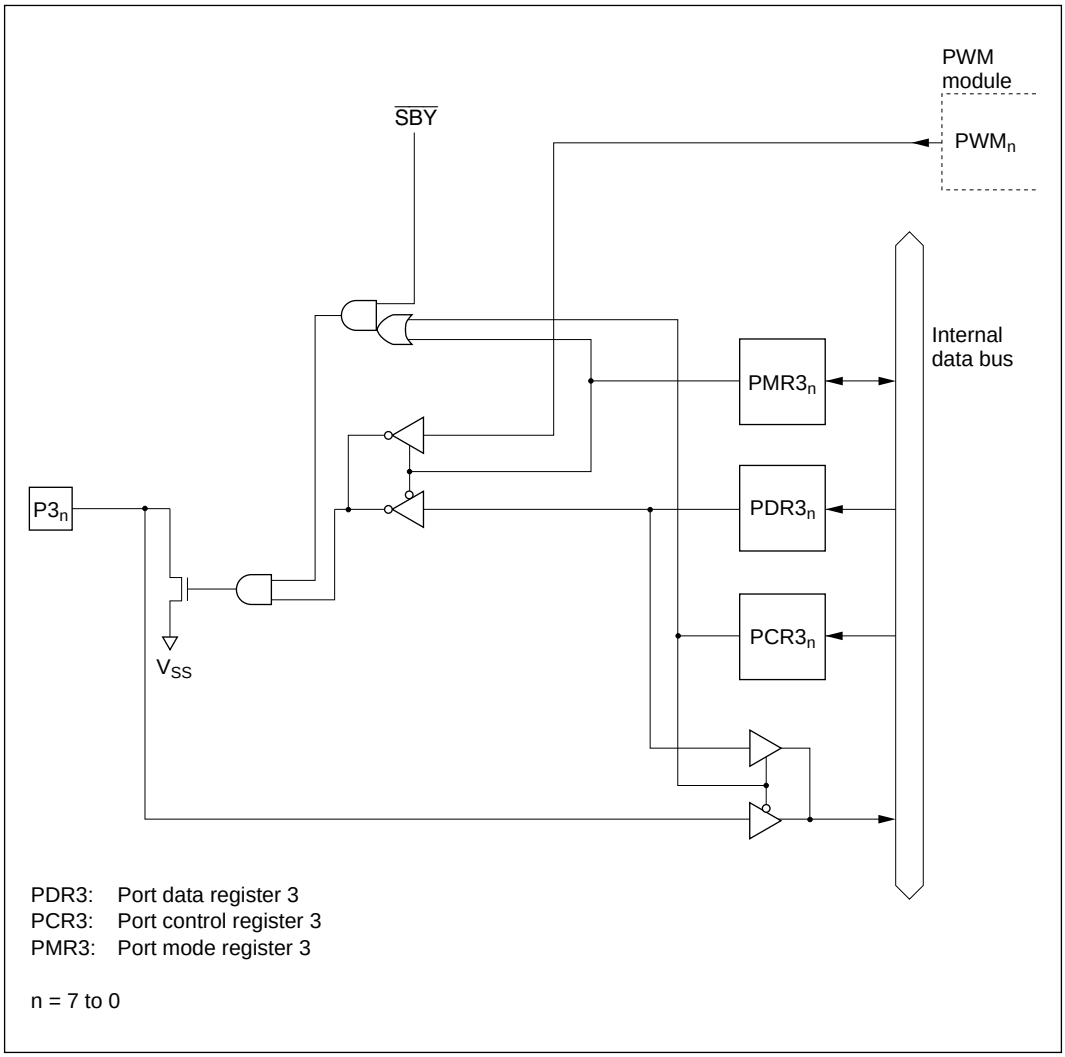


Figure C-3 Port 3 Block Diagram

C.4 Schematic Diagram of Port 4

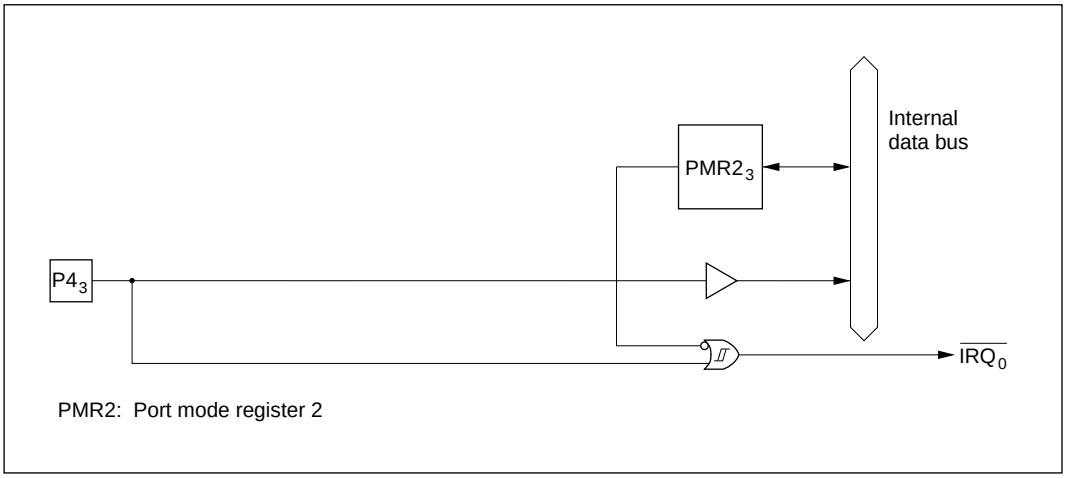


Figure C-4 (a) Port 4 Block Diagram (Pin $P4_3$)

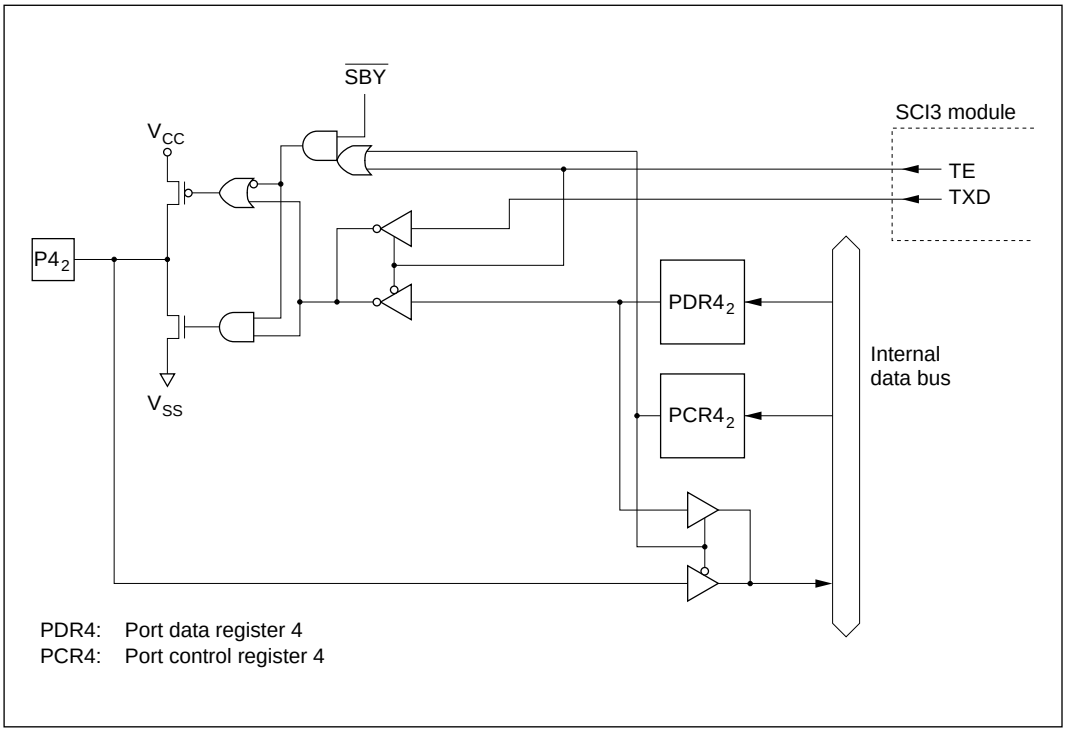


Figure C-4 (b) Port 4 Block Diagram (Pin $P4_2$)

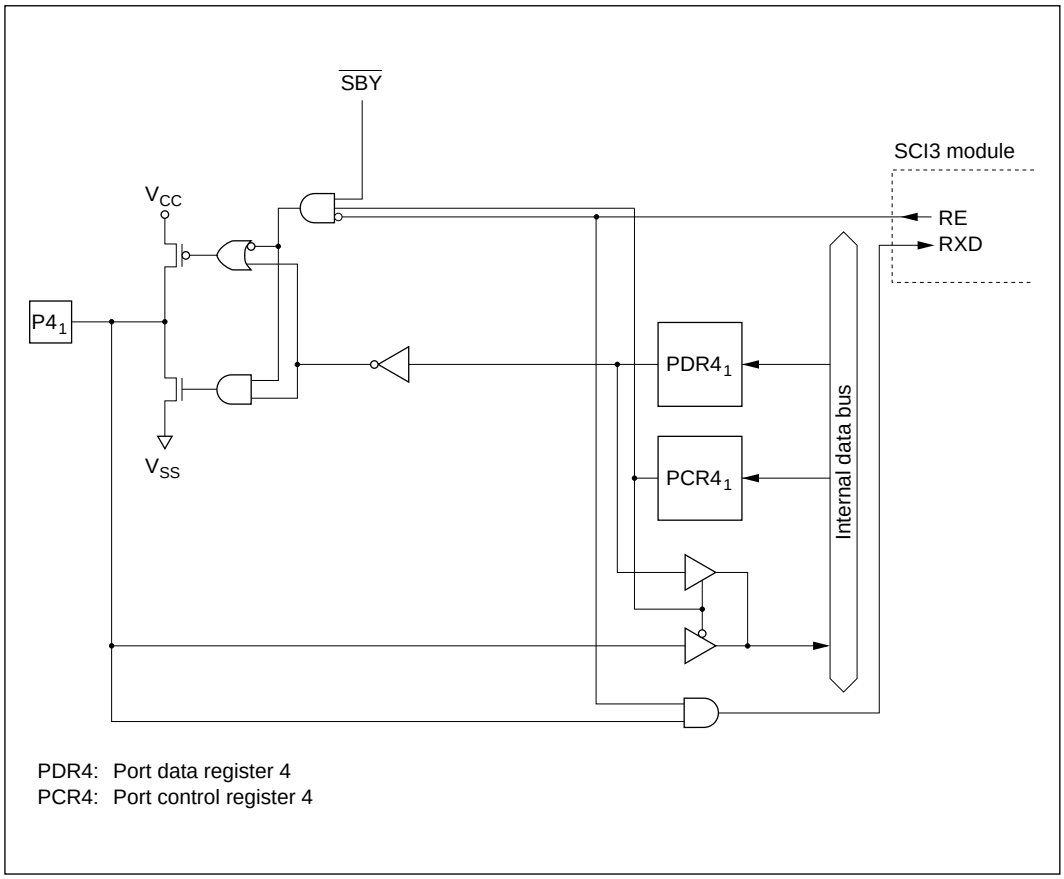


Figure C-4 (c) Port 4 Block Diagram (Pin $P4_1$)

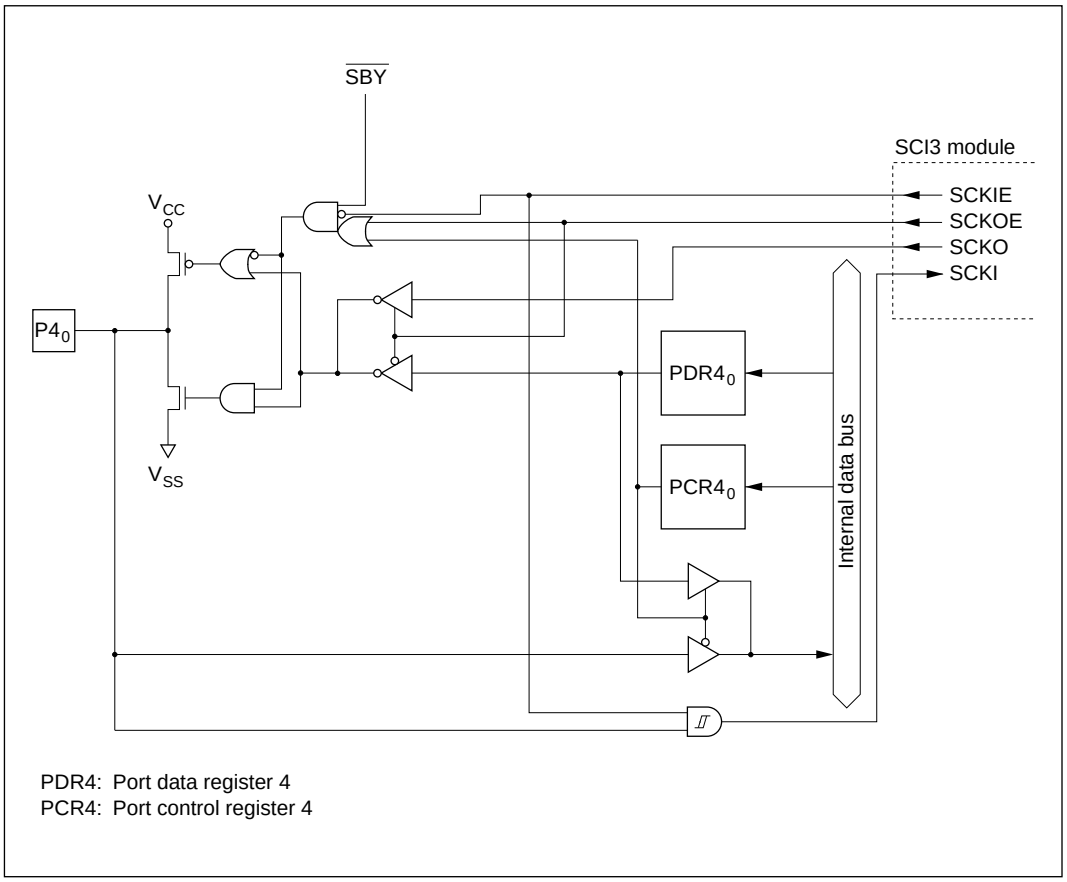
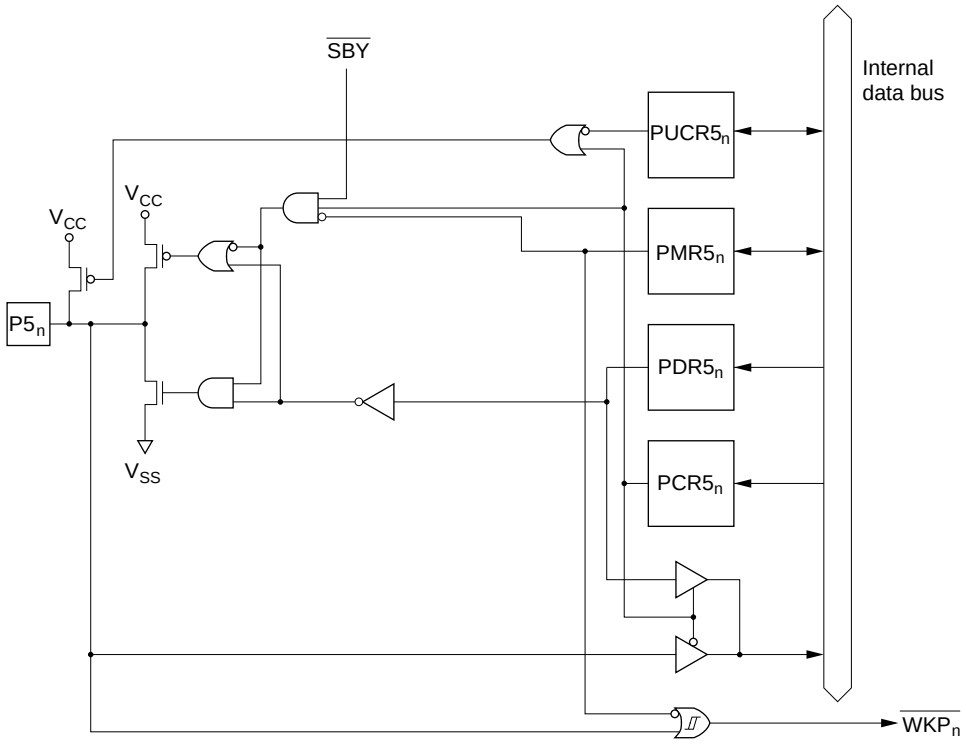


Figure C-4 (d) Port 4 Block Diagram (Pin $P4_0$)

C.5 Schematic Diagram of Port 5

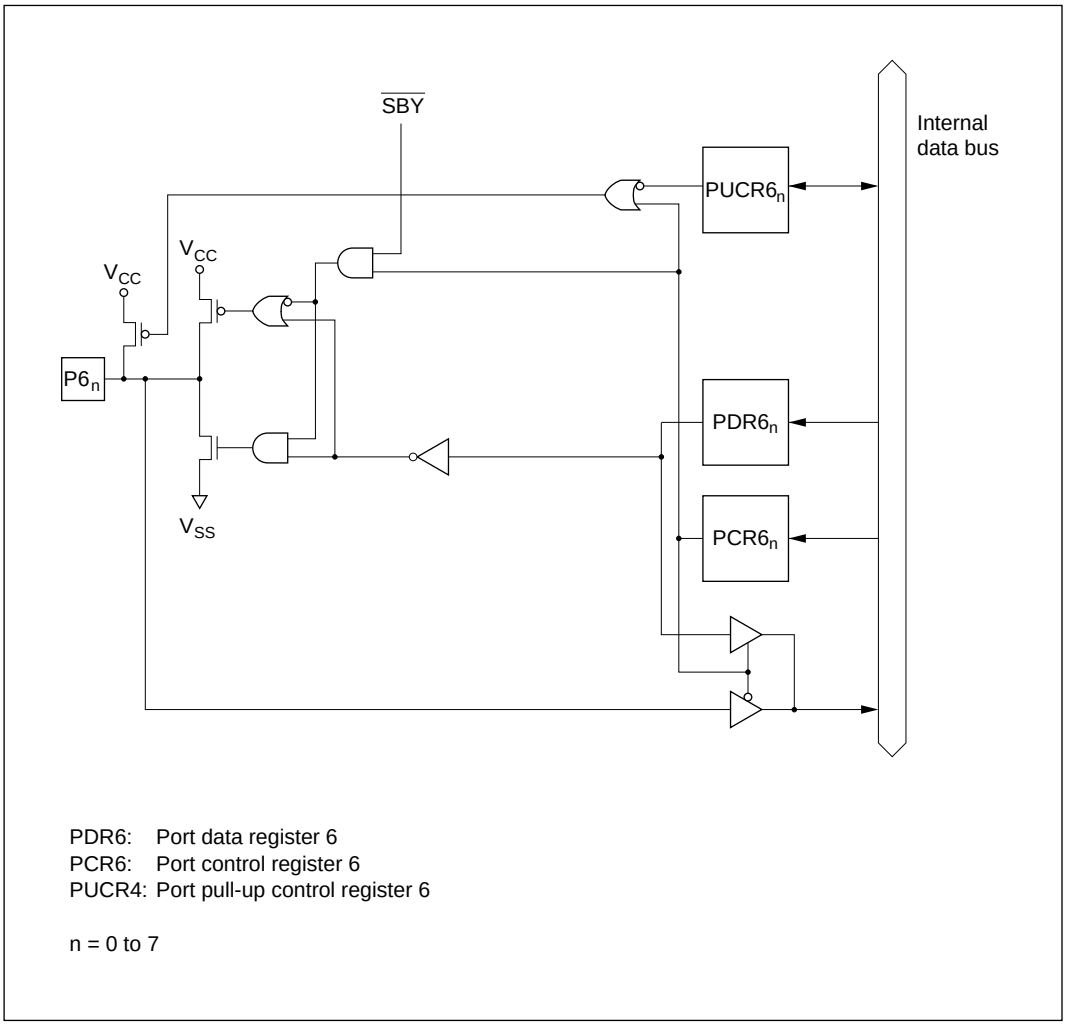


PDR5: Port data register 5
PCR5: Port control register 5
PMR5: Port mode register 5
PUCR5: Port pull-up control register 5

n = 0 to 7

Figure C-5 Port 5 Block Diagram

C.6 Schematic Diagram of Port 6



PDR6: Port data register 6
PCR6: Port control register 6
PUCR4: Port pull-up control register 6

n = 0 to 7

Figure C-6 Port 6 Block Diagram

C.7 Schematic Diagram of Port 7

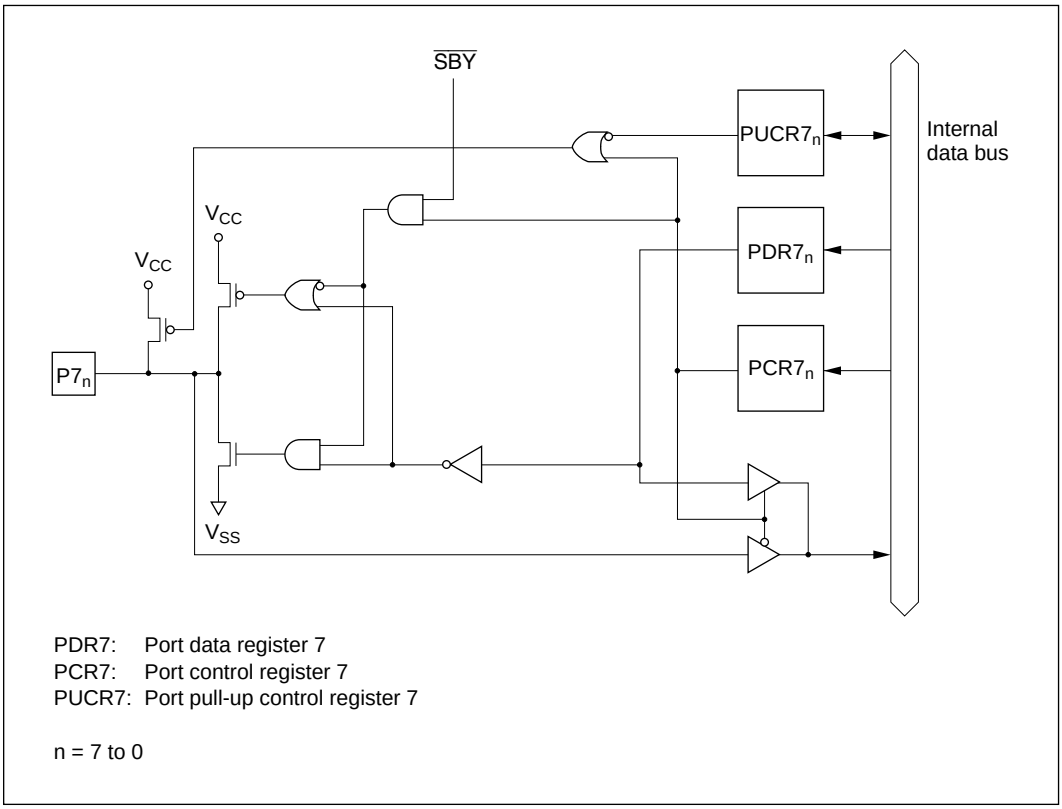


Figure C-7 Port 7 Block Diagram

C.8 Schematic Diagram of Port 8

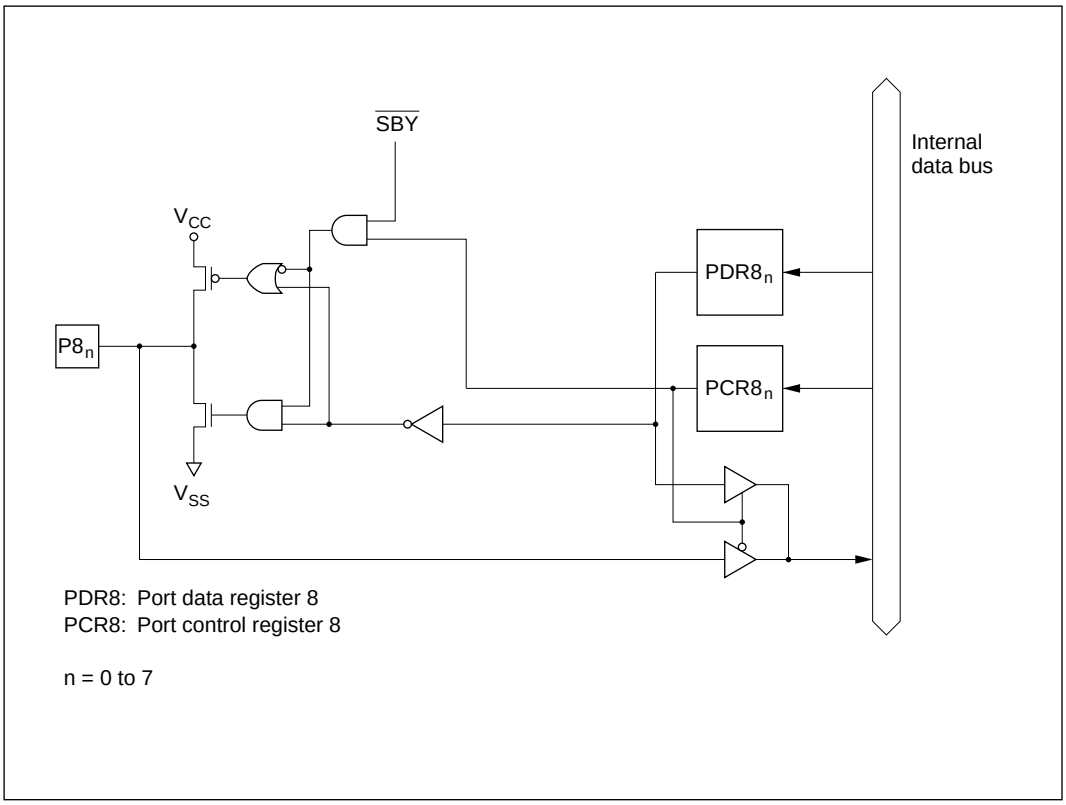


Figure C-8 Port 8 Block Diagram

C.9 Schematic Diagram of Port 9

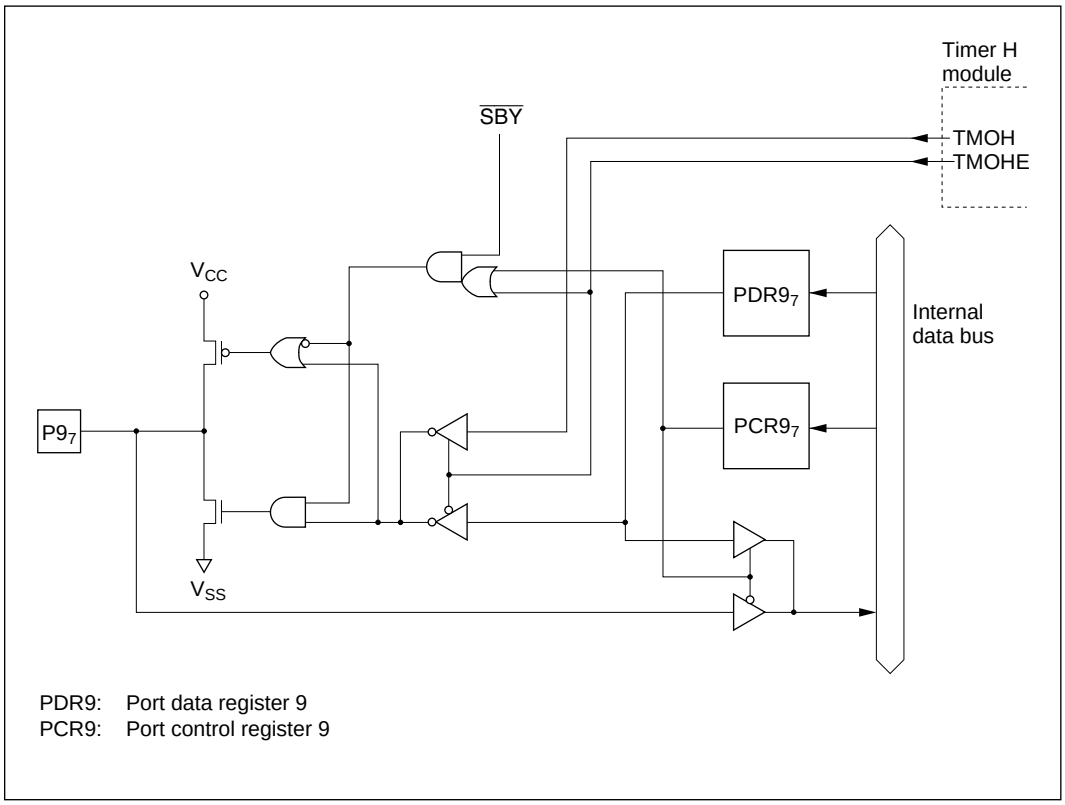


Figure C-9 (a) Port 9 Block Diagram (Pin $P9_7$)

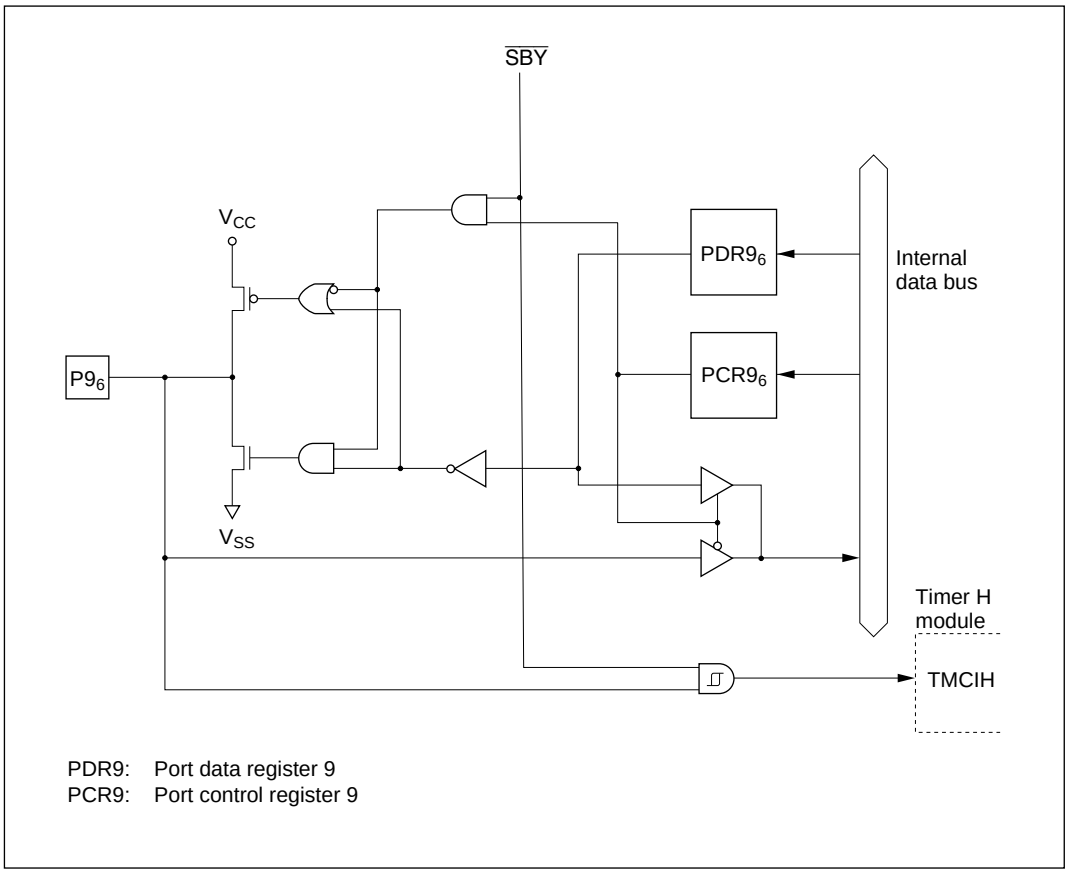


Figure C-9 (b) Port 9 Block Diagram (Pin $P9_6$)

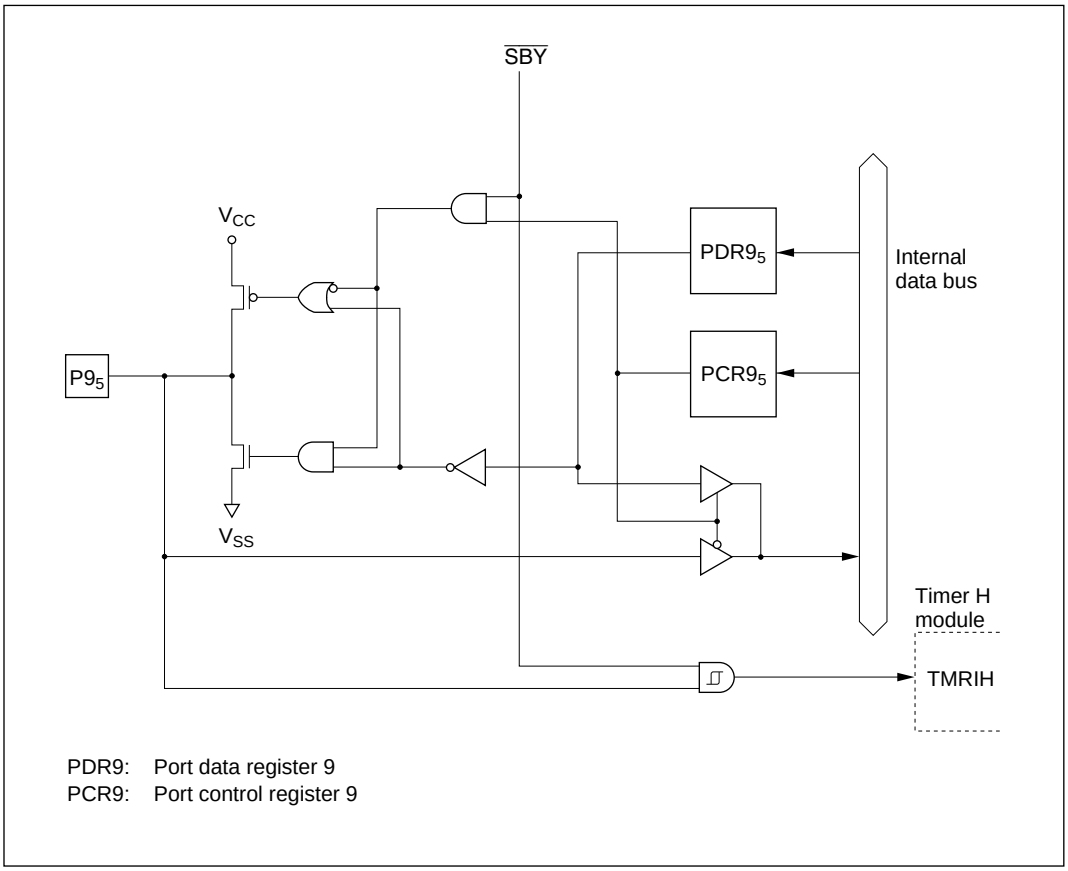


Figure C-9 (c) Port 9 Block Diagram (Pin $P9_5$)

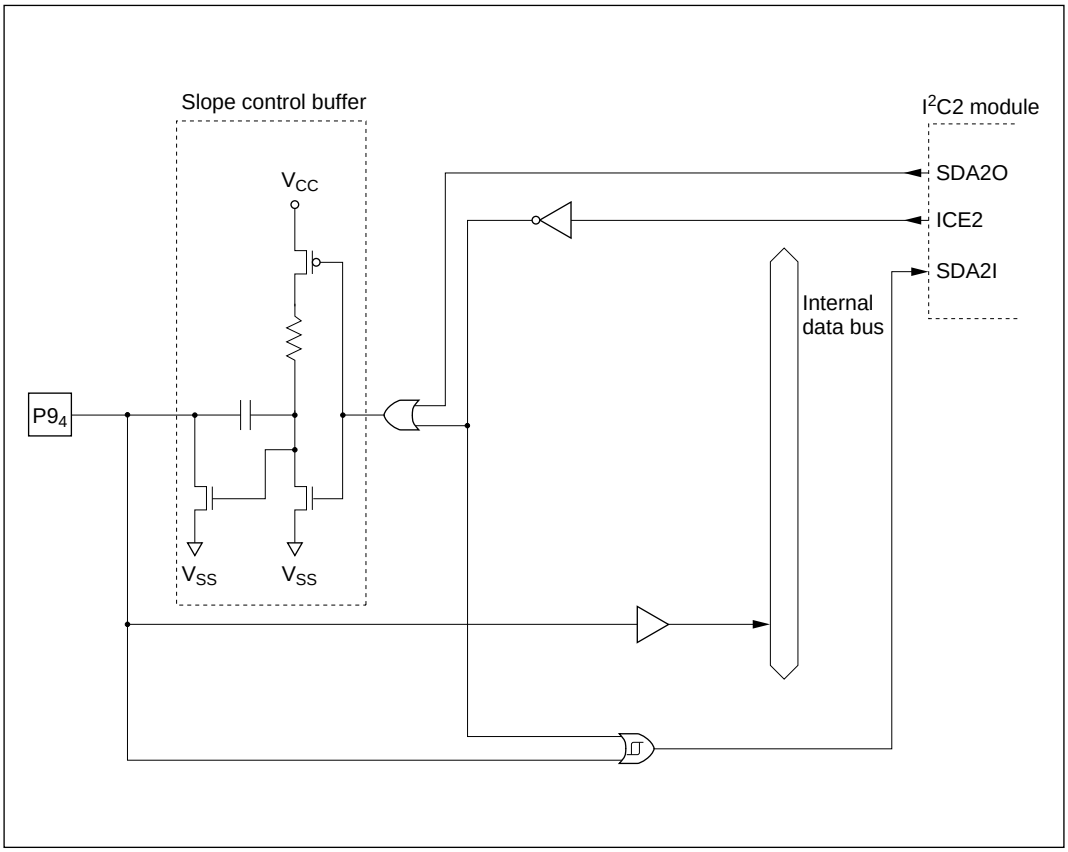


Figure C-9 (d) Port 9 Block Diagram (Pin $P9_4$)

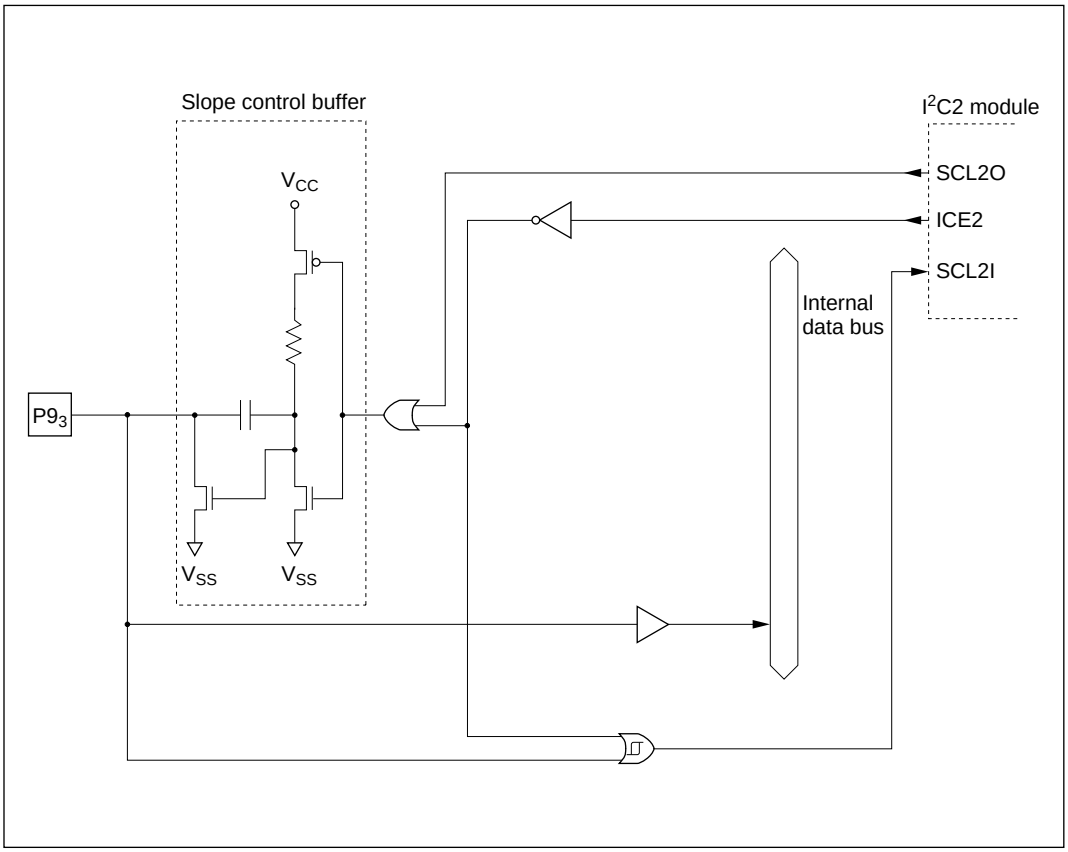


Figure C-9 (e) Port 9 Block Diagram (Pin $P9_3$)

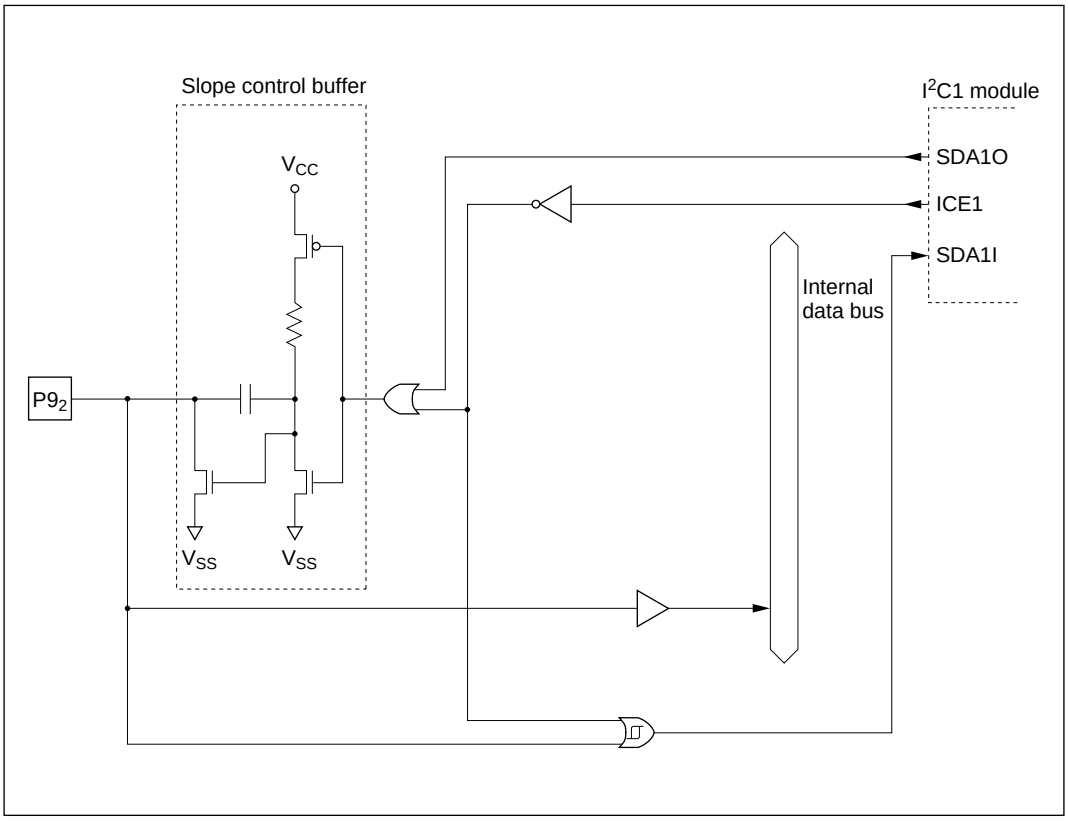


Figure C-9 (f) Port 9 Block Diagram (Pin $P9_2$)

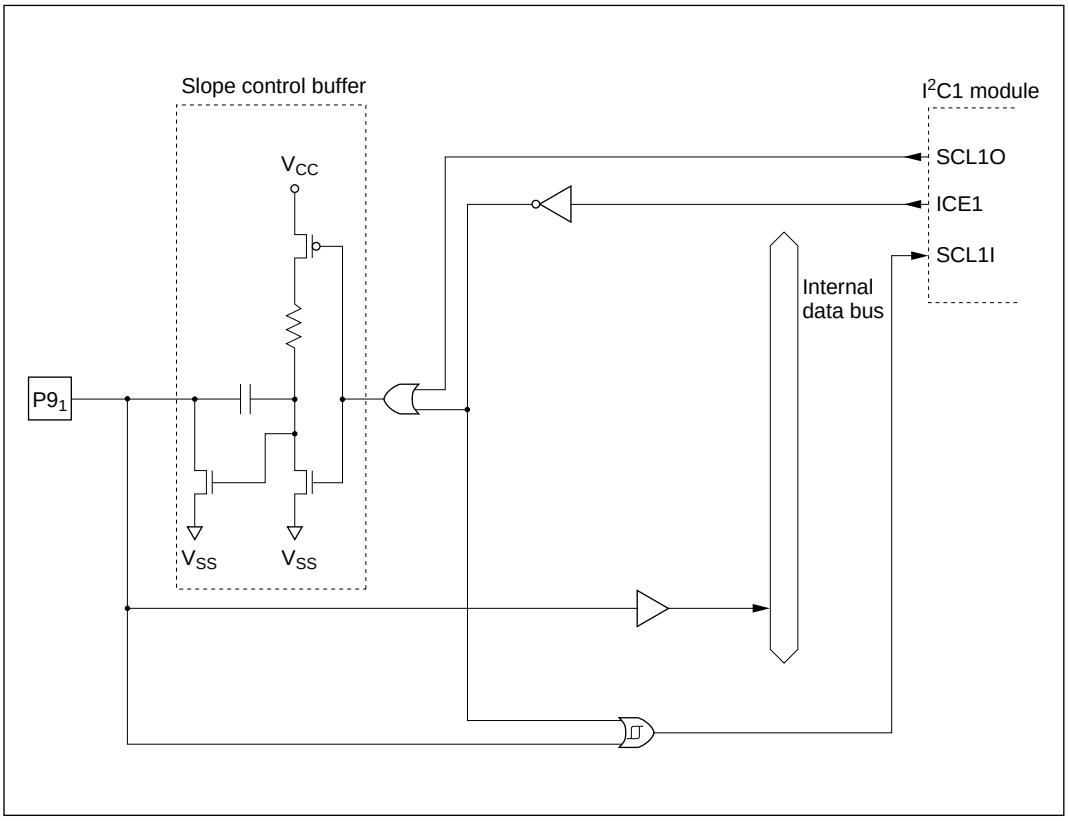


Figure C-9 (g) Port 9 Block Diagram (Pin $P9_1$)

C.10 Schematic Diagram of Port A

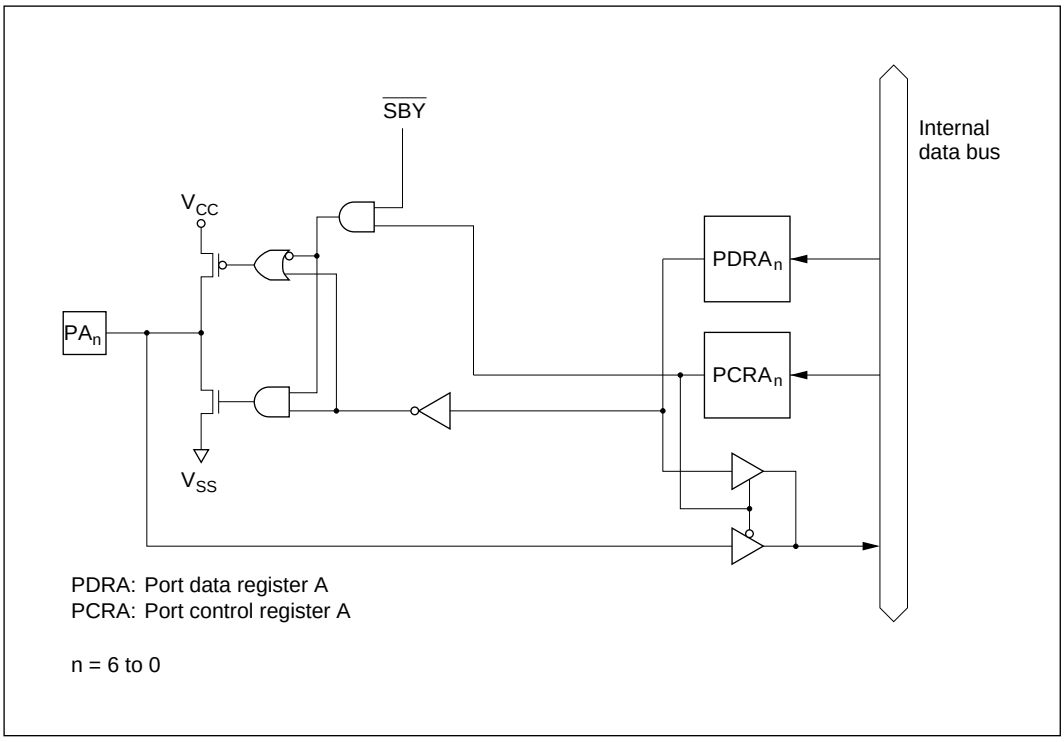


Figure C-10 Port A Block Diagram

C.11 Schematic Diagram of Port B

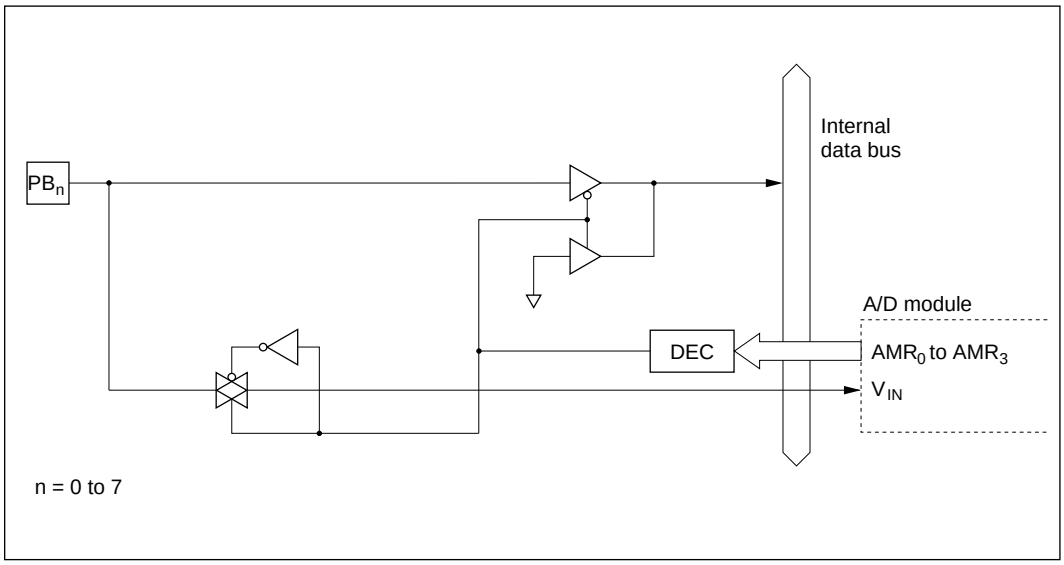


Figure C-11 Port B Block Diagram

C.12 Schematic Diagram of Port C

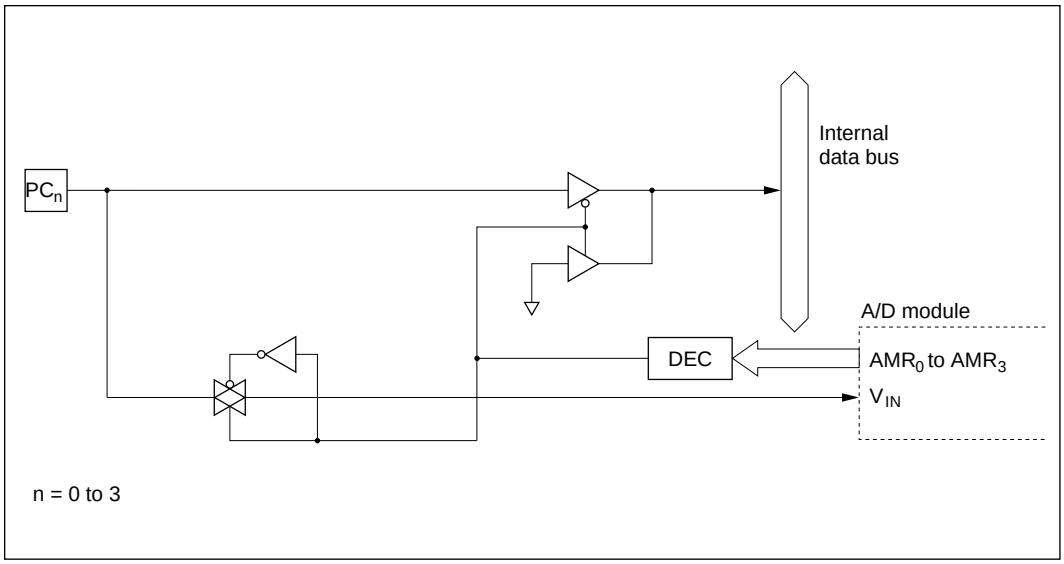


Figure C-12 Port C Block Diagram

Appendix D Port States in the Different Processing States

Table D-1 Port States Overview

Port	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
$P1_7$ to $P1_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P2_7$ to $P2_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P3_7$ to $P3_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P4_3$ to $P4_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P5_7$ to $P5_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P6_7$ to $P6_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P7_7$ to $P7_0$	High impedance	Retained	Retained	High impedance*	Retained	Functions	Functions
$P8_7$ to $P8_0$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
$P9_7$ to $P9_1$	High impedance	Retained	Retained	High impedance	Retained	Functions	Functions
PA_6 to PA_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
PC_3 to PC_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 Package Dimensions

Figure E-1 shows the dimensions of the FP-100A package of the H8/3947 Series.

unit: mm

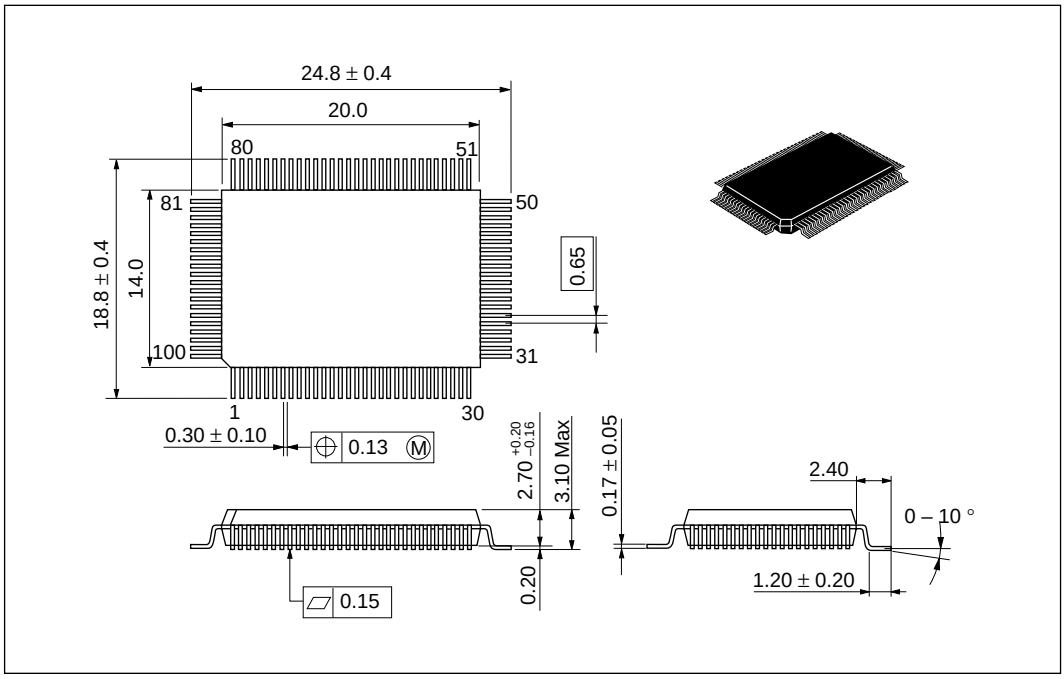


Figure E-1 FP-100A Package Dimensions

Note: In case of inconsistencies arising within figures, dimensional drawings listed in the Hitachi Semiconductor Packages Manual take precedence and are considered correct.