REJ09B0430-0100

Everywhere you imagine. RENESAS

16

H8/38704 Group, H8/38702S Group Hardware Manual

Hardware Manual

Renesas 16-Bit Single-Chip Microcomputer
H8 Family / H8/300H Super Low Power Series

H8/38704 Group	H8/38704
	H8/38703
	H8/38702
H8/38702S Group	H8/38702S
	H8/38701S
	H8/38700S

Rev.1.00
Revision Date: Dec. 13, 2007

RenesasTechnology
www.renesas.com

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

How to Use This Manual

1. Objective and Target Users

   This manual was written to explain the hardware functions and electrical characteristics of this LSI to the target users, i.e. those who will be using this LSI in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logic circuits, and microcomputers.

   This manual is organized in the following items: an overview of the product, descriptions of the CPU, system control functions, and peripheral functions, electrical characteristics of the device, and usage notes.

   When designing an application system that includes this LSI, take all points to note into account. Points to note are given in their contexts and at the final part of each section, and in the section giving usage notes.

   The list of revisions is a summary of major points of revision or addition for earlier versions. It does not cover all revised items. For details on the revised points, see the actual locations in the manual.

   The following documents have been prepared for the H8/38704 Group and the H8/38702S Group. Before using any of the documents, please visit our web site to verify that you have the most up-to-date available version of the document.

Document Type	Contents	Document Title	Document No.
Data Sheet	Overview of hardware and electrical characteristics	—	—
Hardware Manual	Hardware specifications (pin assignments, memory maps, peripheral specifications, electrical characteristics, and timing charts) and descriptions of operation	H8/38704, H8/38702S Group Hardware Manual	This manual
Software Manual	Detailed descriptions of the CPU and instruction set	H8/300H Series Software Manual	REJ09B0213
Application Note	Examples of applications and sample programs	The latest versions are available from our web site.	
Renesas Technical Update	Preliminary report on the specifications of a product, document, etc.		

2. Description of Numbers and Symbols

Aspects of the notations for register names, bit names, numbers, and symbolic names in this manual are explained below.

(1) Overall notation

In descriptions involving the names of bits and bit fields within this manual, the modules and registers to which the bits belong may be clarified by giving the names in the forms "module name"."register name"."bit name" or "register name"."bit name".

(2) Register notation

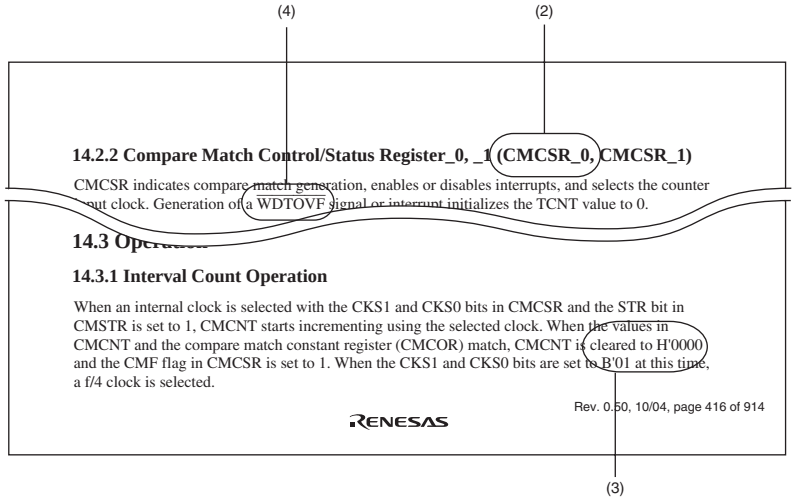
The style "register name"_"instance number" is used in cases where there is more than one instance of the same function or similar functions.
[Example] CMCSR_0: Indicates the CMCSR register for the compare-match timer of channel 0.

(3) Number notation

Binary numbers are given as B'nnnn (B' may be omitted if the number is obviously binary), hexadecimal numbers are given as H'nnnn or 0xnnnn, and decimal numbers are given as nnnn.
[Examples] Binary: B'11 or 11
Hexadecimal: H'EFA0 or 0xEFA0
Decimal: 1234

(4) Notation for active-low

An overbar on the name indicates that a signal or pin is active-low.
[Example] $\overline{\text{WDTOVF}}$



Note: The bit names and sentences in the above figure are examples and have nothing to do with the contents of this manual.

3. Description of Registers

Each register description includes a bit chart, illustrating the arrangement of bits, and a table of bits, describing the meanings of the bit settings. The standard format and notation for bit charts and tables are described below.

[Table of Bits]

Bit (1)	Bit Name (2)	Initial Value (3)	R/W (4)	Description (5)
15	–	0	R	Reserved
14	–	0	R	These bits are always read as 0.
13 to 11	ASID2 to ASID0	All 0	R/W	Address Identifier These bits enable or disable the pin function.
10	–	0	R	Reserved This bit is always read as 0.
9	–	1	R	Reserved This bit is always read as 1.
	–	0		

Note: The bit names and sentences in the above figure are examples, and have nothing to do with the contents of this manual.

(1) Bit
Indicates the bit number or numbers.
In the case of a 32-bit register, the bits are arranged in order from 31 to 0. In the case of a 16-bit register, the bits are arranged in order from 15 to 0.

(2) Bit name
Indicates the name of the bit or bit field.
When the number of bits has to be clearly indicated in the field, appropriate notation is included (e.g., ASID[3:0]).
A reserved bit is indicated by "–".
Certain kinds of bits, such as those of timer counters, are not assigned bit names. In such cases, the entry under Bit Name is blank.

(3) Initial value
Indicates the value of each bit immediately after a power-on reset, i.e., the initial value.
0: The initial value is 0
1: The initial value is 1
–: The initial value is undefined

(4) R/W
For each bit and bit field, this entry indicates whether the bit or field is readable or writable, or both writing to and reading from the bit or field are impossible.
The notation is as follows:

R/W: The bit or field is readable and writable.
R/(W): The bit or field is readable and writable.
However, writing is only performed to flag clearing.
R: The bit or field is readable.
"R" is indicated for all reserved bits. When writing to the register, write the value under Initial Value in the bit chart to reserved bits or fields.
W: The bit or field is writable.

(5) Description
Describes the function of the bit or field and specifies the values for writing.

4. Description of Abbreviations

The abbreviations used in this manual are listed below.

- Abbreviations used in this manual

Abbreviation	Description
ACIA	Asynchronous communication interface adapter
bps	Bits per second
CRC	Cyclic redundancy check
DMA	Direct memory access
DMAC	Direct memory access controller
GSM	Global System for Mobile Communications
Hi-Z	High impedance
IEBus	Inter Equipment Bus (IEBus is a trademark of NEC Electronics Corporation.)
I/O	Input/output
IrDA	Infrared Data Association
LSB	Least significant bit
MSB	Most significant bit
NC	No connection
PLL	Phase-locked loop
PWM	Pulse width modulation
SFR	Special function register
SIM	Subscriber Identity Module
UART	Universal asynchronous receiver/transmitter
VCO	Voltage-controlled oscillator

5. List of Product Specifications

Below is a table listing the product specifications for each group.

		H8/38704 Group		H8/38702S Group
Item		**Flash Memory**	**Mask ROM**	**Mask ROM**
Memory	ROM	16 k, 32 kbytes	16 k, 24 k, 32 kbytes	8 k, 12 k, 16 kbytes
	RAM	1 kbyte	1 kbyte	512 bytes
Operating voltage and operating frequency	4.5 to 5.5 V	—	16 MHz	—
	2.7 to 5.5 V	—	16 MHz	—
	1.8 to 5.5 V	—	—	—
	2.7 to 3.6 V	10 MHz	—	10 MHz
	1.8 to 3.6 V	4 MHz (2.2 V or more)	—	4 MHz
I/O ports	Input	9	9	9
	Output	6	5	6
	I/O	39	39	39
Timers	Clock (timer A)	1	1	1
	Compare (timer F)	1	1	1
	AEC	1	1	1
	WDT	1	1	1
	WDT (discrete)	—	—	—
SCI	UART/Clock frequency	1 ch	1 ch	1 ch
A-D (resolution × input channels)		10 bit × 4 ch	10 bit × 4 ch	10 bit × 4 ch
External interrupt (internal wakeup)		11(8)	11(8)	11(8)
Package		FP-64A	FP-64A	FP-64A
		FP-64E	FP-64E	FP-64K
		TNP-64B	TNP-64B	—
Operating temperature		Standard specifications: –20 to 75°C, WTR: –40 to 85°C		

Contents

Section 1 Overview

1.1 Features

Microcontrollers of the H8/38704 Group and the H8/38702S Group are CISC (complex instruction set computer) microcontrollers whose core is an H8/300H CPU, which has an internal 32-bit architecture. The H8/300H CPU provides upward compatibility with the H8/300 CPUs of other Renesas Technology-original microcontrollers.

As peripheral functions, each LSI of these Groups includes various timer functions that realize low-cost configurations for end systems. The power consumption of these modules can be kept down dynamically by power-down mode.

1.1.1 Application

Examples of the applications of this LSI include motor control, power meter, and health equipment.

1.1.2 Overview of Specifications

Table 1.1 lists the functions of H8/38704, H8/38702S Group products in outline.

Table 1.1 Overview of Functions

Classification	Module/ Function	Description
Memory	—	• ROM lineup: Flash memory version and mask Rom version • ROM capacity: 8 k, 12 k, 16 k, 24 k, and 32 kbytes
	RAM	• RAM capacity: 512 and 1024 bytes
CPU	CPU	• H8/300H CPU (CISC type) Upward compatibility for H8/300 CPU at object level • Sixteen 16-bit general registers • Eight addressing modes • 64-Kbyte address space Program: 64 Kbytes available Data: 64 Kbytes available • 62 basic instructions, classifiable as bit arithmetic and logic instructions, multiply and divide instructions, bit manipulation instructions, and others • Minimum instruction execution time: 400 ns (for an ADD instruction while system clock ϕ = 5 MHz and V_{CC} = 2.7 to 3.6 V) • On-chip multiplier (16 × 16 → 32 bits)
	Operating mode	• Normal mode
	MCU operating mode	Mode: Single-chip mode • Low power consumption state (transition driven by the SLEEP instruction)
Interrupt (source)	Interrupt controller (INTC)	• Eleven external interrupt pins (IRQAEC, $\overline{\text{IRQ1}}$, $\overline{\text{IRQ0}}$, $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$) • Seven internal interrupt sources • Independent vector addresses

Classification	Module/ Function	Description
Clock	Clock pulse generator (CPG)	• Two clock generation circuits available • Separate clock signals are provided for each of functional modules • Includes frequency division circuit, so the operating frequency is selectable • Seven low-power-consumption modes: Active (medium speed) mode, sleep (high speed or medium speed) mode, subactive mode, subsleep mode, standby mode, and watch mode
A/D converter	A/D converter (ADC)	• 10-bit resolution × four input channels • Sample and hold function included • Conversion time: 12.4 μs per channel (with φ at 5-MHz operation) • Method of starting A/D conversion: software
Timer	10-bit PWM	• 10 bits × two channels • Four conversion periods selectable • Pulse division method for less ripple
	Timer A	• 8-bit timer • Interval timer functionality: Eight interrupt periods are selectable • Clock time base functionality: Four overflow periods are selectable • Generates an interrupt upon overflow
	Timer F	• 16-bit timer (also can be used as two independent 8-bit timers) • Four counter input clocks • Output compare function supported • Toggle output function supported • two interrupt sources: Compare match and overflow
	Asynchron-ous event counter (AEC)	• 16-bit pulse timer (also can be used as two 8 bits × two channels) • Can count asynchronously-input external events
Watchdog timer	Watchdog timer (WDT)	8 bits × one channel (selectable from two counter input clocks)

Classification	Module/ Function	Description
Serial interface	Serial communi-cations interface 3 (SCI3)	• For both asynchronous and clock synchronous serial communications • Full-duplex communications capability • Select the desired bit rate • Six interrupt sources
I/O ports		• Five CMOS input-only pins • 39 CMOS input/output pins • Six large-current-drive pins (port 9) • 23 pull-up resistors • Six open drains
Package		• QFP-64: package code: FP-64A (package dimensions: 14 × 14 mm, pin pitch: 0.8 mm) • LQFP-64: package code: FP-64E (package dimensions: 10 × 10 mm, pin pitch: 0.5 mm) • LQFP-64: package code FP-64K (package dimensions: 10 × 10 mm, pin pitch: 0.5 mm) • P-VQFN-64: package code TNP-64B (package dimensions: 8 × 8 mm, pin pitch: 0.4 mm) Packages FP-64E and FP-64K have different package dimensions. For details, see appendix E, Package Dimensions.
Operating frequency/ Power supply voltage		• Operating frequency: 2 to 10 MHz • Power supply voltage: Flash memory version: Vcc = 2.2 to 3.6 V, AVcc = 2.2 to 3.6 V Mask ROM version: Vcc = 1.8 to 3.6 V, AVcc = 1.8 to 3.6 V • Supply current: Flash memory version: 3.6 mA (typ.) (Vcc = 3.0 V, AVcc = 3.0 V, ϕ = 10 MHz) Mask ROM version: 3.1 mA (typ.) (Vcc = 3.0 V, AVcc = 3.0 V, ϕ = 10 MHz)
Operating peripheral temperature (°C)		• –20 to +75°C (regular specifications) • –40 to +85°C (wide-range specifications)

1.2 List of Products

Table 1.2 and figure 1.1 show the list of products and the structure of a product number, respectively.

Table 1.2 List of Products

Group	Product Type	ROM Size	RAM Size	Package	Remarks
H8/38704 Group	HD64F38704	32 kbytes	1 kbyte	FP-64A, FP-64E*1, TNP-64B	Flash memory version
	HD64338704	32 kbytes	1 kbyte		Mask ROM version
	HD64338703	24 kbytes	1 kbyte		Mask ROM version
	HD64F38702	16 kbytes	1 kbyte		Flash memory version
	HD64338702	16 kbytes	1 kbyte		Mask ROM version
H8/38702S Group	HD64338702S	16 kbytes	512 bytes	FP-64A, FP-64K*2, TNP-64B	Mask ROM version
	HD64338701S	12 kbytes	512 bytes		Mask ROM version
	HD64338700S	8 kbytes	512 bytes		Mask ROM version

Notes: 1. FP-64E package is only available as an H8/38704 Group microcontroller.
2. FP-64K package is only available as an H8/38702S Group microcontroller.

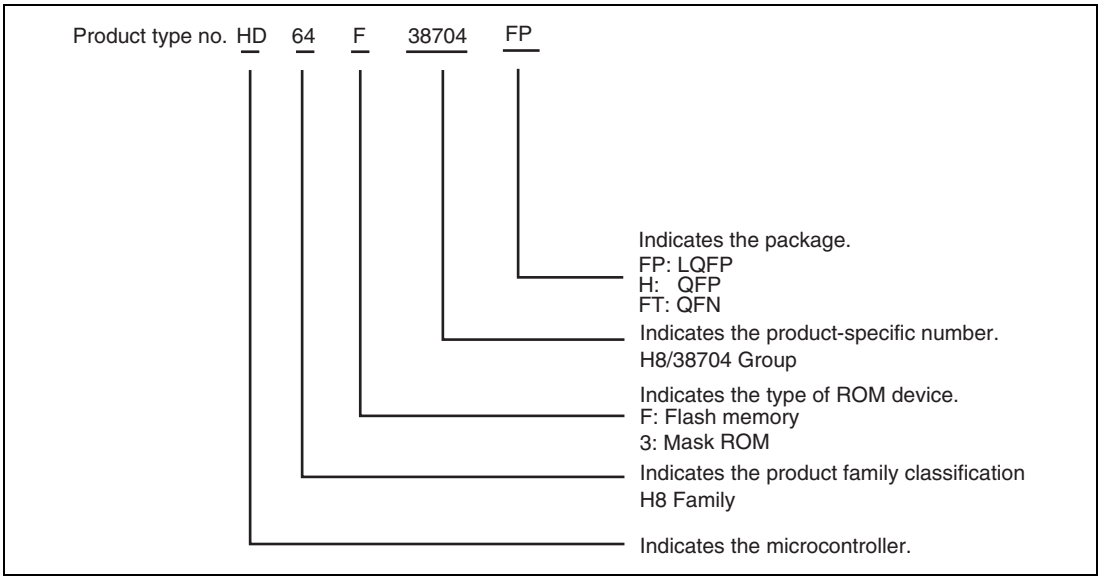
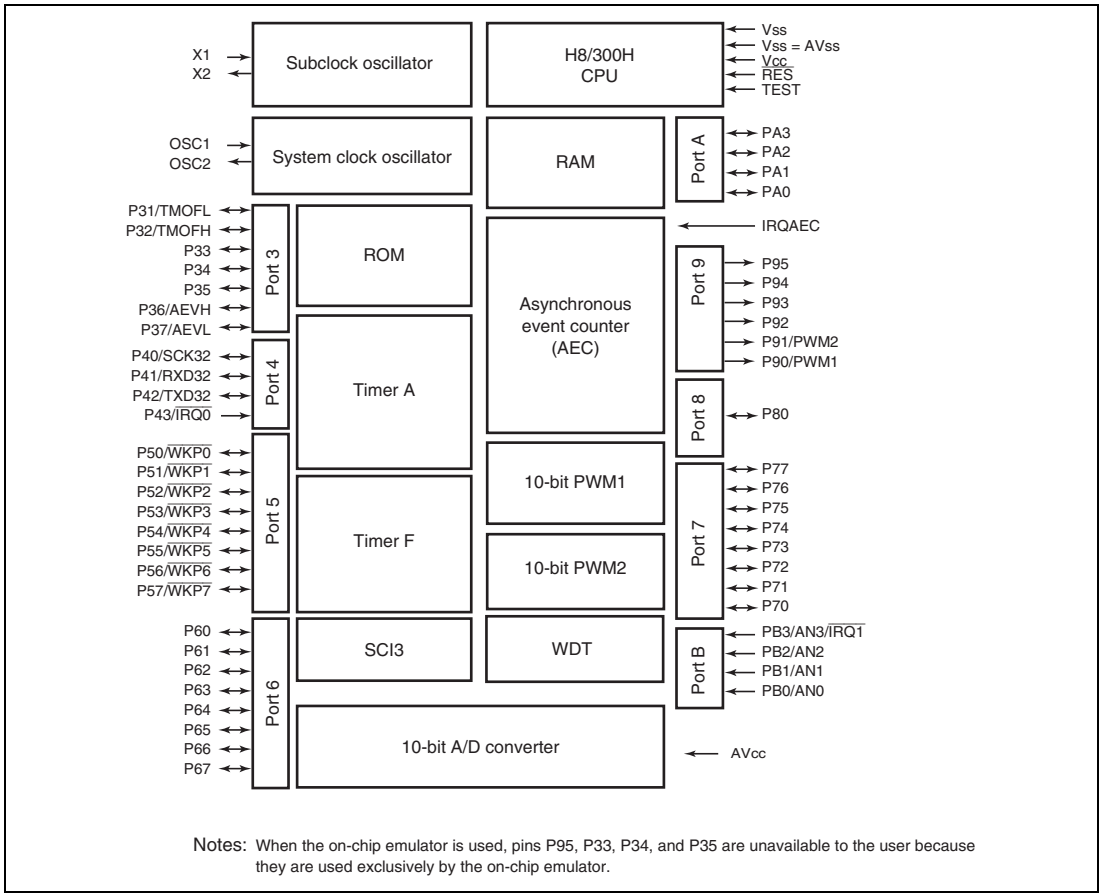


Figure 1.1 How to Read the Product Name Code

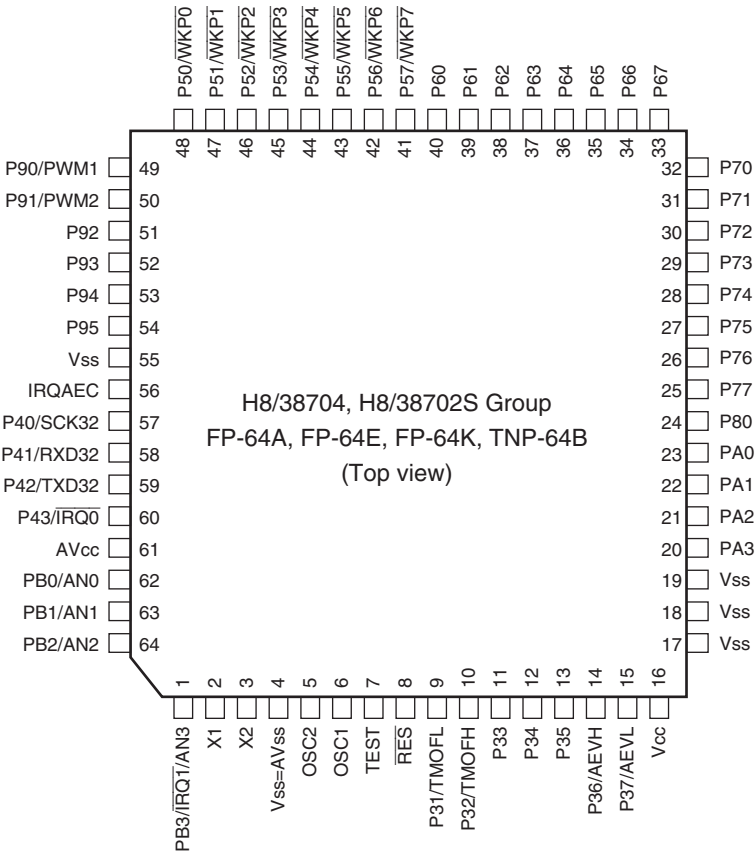
1.3 Block Diagram



Notes: When the on-chip emulator is used, pins P95, P33, P34, and P35 are unavailable to the user because they are used exclusively by the on-chip emulator.

Figure 1.2 Block Diagram of H8/38704, H8/38702S Group

1.4 Pin Assignment



Note: When the on-chip emulator is used, pins P95, P33, P34, and P35 are unavailable to the user because they are used exclusively by the on-chip emulator.

Figure 1.3 Pin Assignment of H8/38704, H8/38702S Group (FP-64A, FP-64E, FP-64K, and TNP-64B)

1.5 Pin Functions

Table 1.3 Pin Functions

Type	Symbol	Pin No. FP-64A, FP-64E*1, FP-64K*2, TNP-64B	I/O	Functions
Power source pins	V_{CC}	16	Input	Power supply pin. Connect this pin to the system power supply.
	V_{SS}	4 (= AV_{SS}) 17, 18, 19, 55	Input	Ground pin. Connect this pin to the system power supply (0V).
	AV_{CC}	61	Input	Analog power supply pin for the A/D converter. When the A/D converter is not used, connect this pin to the system power supply.
	AV_{SS}	4 (= V_{SS})	Input	Ground pin for the A/D converter. Connect this pin to the system power supply (0 V).
Clock pins	OSC1	6	Input	These pins connect to a crystal or ceramic resonator for system clocks, or can be used to input an external clock. See section 4, Clock Pulse Generators, for a typical connection.
	OSC2	5	Output	
	X1	2	Input	These pins connect to a 32.768- or 38.4-kHz crystal resonator for subclocks. See section 4, Clock Pulse Generators, for a typical connection.
	X2	3	Output	
System control	$\overline{RES}$	8	Input	Reset pin. When this driven low, the chip is reset.
	TEST	7	Input	Test pin. Connect this pin to V_{ss}. Users cannot use this pin.

Type	Symbol	Pin No. FP-64A, FP-64E*1, FP-64K*2, TNP-64B	I/O	Functions
Interrupt pins	$\overline{IRQ0}$	60	Input	External interrupt request input pins. Can select the rising or falling edge.
	$\overline{IRQ1}$	1		
	IRQAEC	56	Input	Asynchronous event counter interrupt input pin. Enables asynchronous event input.
	$\overline{WKP7}$ to $\overline{WKP0}$	41 to 48	Input	Wakeup interrupt request input pins. Can select the rising or falling edge.
Timer	AEVL AEVH	15 14	Input	This is an event input pin for input to the asynchronous event counter.
	TMOFL	9	Output	This is an output pin for waveforms generated by the timer FL output compare function.
	TMOFH	10	Output	This is an output pin for waveforms generated by the timer FH output compare function.
10-bit PWM	PWM1	49	Output	These are output pins for waveforms generated by the channel 1 and 2 10-bit PWMs.
	PWM2	50		
I/O ports	P37 to P31	15 to 9	I/O	7-bit I/O port. Input or output can be designated for each bit by means of the port control register 3 (PCR3). When the on-chip emulator is used, pins P33, P34, and P35 are unavailable to the user because they are used exclusively by the on-chip emulator.
	P43	60	Input	1-bit input port.
	P42 to P40	59 to 57	I/O	3-bit I/O port. Input or output can be designated for each bit by means of the port control register 4 (PCR4).
	P57 to P50	41 to 48	I/O	8-bit I/O port. Input or output can be designated for each bit by means of the port control register 5 (PCR5).

Type	Symbol	Pin No. FP-64A, FP-64E*1, FP-64K*2, TNP-64B	I/O	Functions
I/O ports	P67 to P60	33 to 40	I/O	8-bit I/O port. Input or output can be designated for each bit by means of the port control register 6 (PCR6).
	P77 to P70	25 to 32	I/O	8-bit I/O port. Input or output can be designated for each bit by means of the port control register 7 (PCR7).
	P80	24	I/O	1-bit I/O port. Input or output can be designated for each bit by means of the port control register 8 (PCR8).
	P95 to P90	54 to 49	Output	6-bit output port. When the on-chip emulator is used, pin P95 is unavailable to the user because it is used exclusively by the on-chip emulator. In the flash memory version, pin P95 should not be open but pulled up to go high in user mode.
	PA3 to PA0	20 to 23	I/O	4-bit I/O port. Input or output can be designated for each bit by means of the port control register A (PCRA).
	PB3 to PB0	1, 64 to 62	Input	4-bit input port.
Serial communications interface (SCI)	RXD32	58	Input	Receive data input pin.
	TXD32	59	Output	Transmit data output pin.
	SCK32	57	I/O	Clock I/O pin.
A/D converter	AN3 to AN0	1, 64 to 62	Input	Analog data input pins.

Notes: 1. FP-64E package is only available as an H8/38704 Group microcontroller.
2. FP-64K package is only available as an H8/38702S Group microcontroller.

Section 2 CPU

This LSI has an H8/300H CPU with an internal 32-bit architecture that is upward-compatible with the H8/300 CPU, and supports only normal mode, which has a 64-kbyte address space.

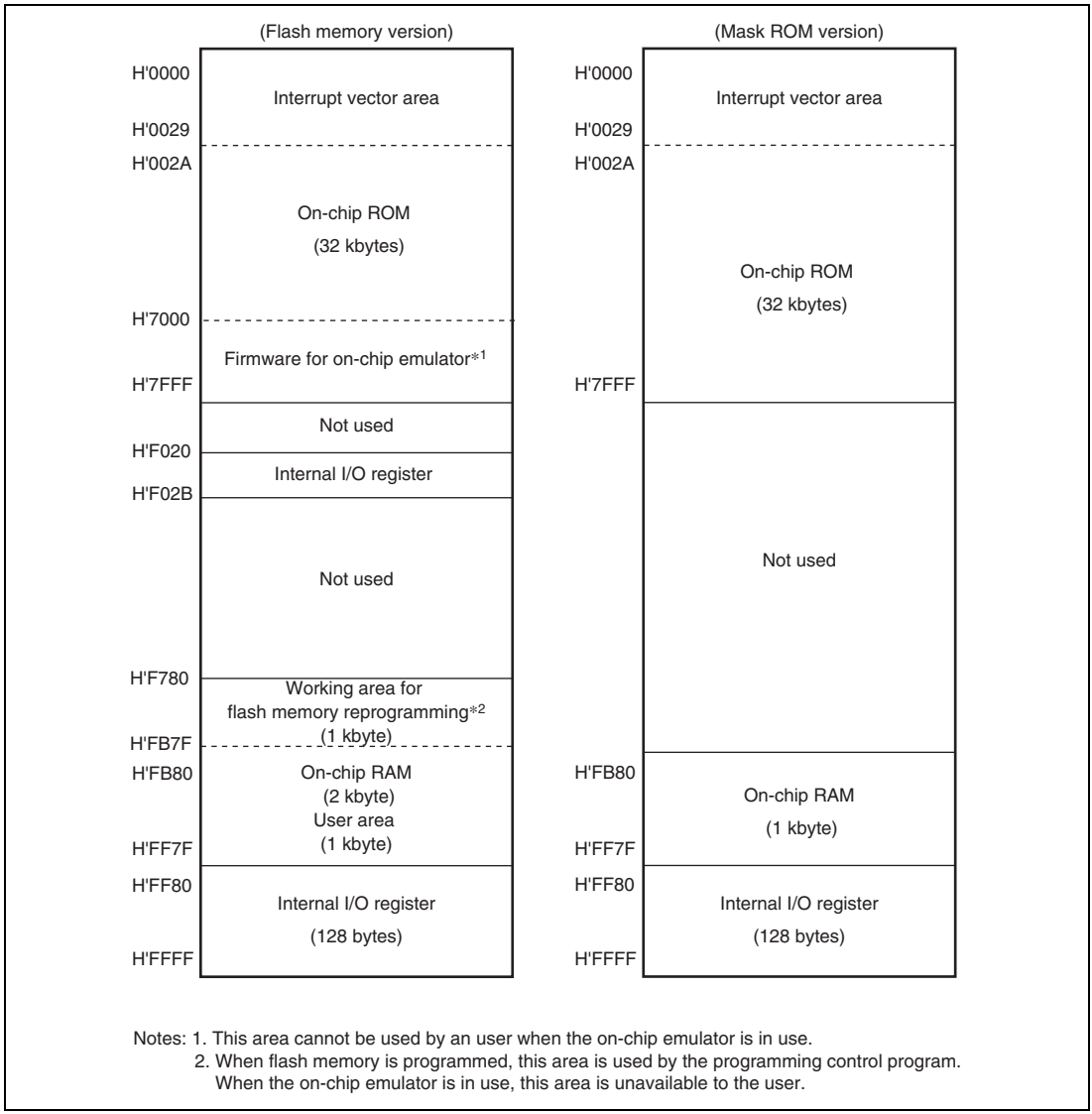
- Upward-compatible with H8/300 CPUs
  Can execute H8/300 CPUs object programs
  Additional eight 16-bit extended registers
  32-bit transfer and arithmetic and logic instructions are added
  Signed multiply and divide instructions are added.
- General-register architecture
  Sixteen 16-bit general registers also usable as sixteen 8-bit registers and eight 16-bit registers, or eight 32-bit registers
- Sixty-two basic instructions
  8/16/32-bit data transfer and arithmetic and logic instructions
  Multiply and divide instructions
  Powerful bit-manipulation instructions
- Eight addressing modes
  Register direct [Rn]
  Register indirect [@ERn]
  Register indirect with displacement [@(d:16,ERn) or @(d:24,ERn)]
  Register indirect with post-increment or pre-decrement [@ERn+ or @–ERn]
  Absolute address [@aa:8, @aa:16, @aa:24]
  Immediate [#xx:8, #xx:16, or #xx:32]
  Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  Memory indirect [@@aa:8]
- 64-kbyte address space

- High-speed operation
  All frequently-used instructions execute in one or two states
  8/16/32-bit register-register add/subtract : 2 states
  8 × 8-bit register-register multiply : 14 states
  16 ÷ 8-bit register-register divide : 14 states
  16 × 16-bit register-register multiply : 22 states
  32 ÷ 16-bit register-register divide : 22 states
- Power-down state
  Transition to power-down state by SLEEP instruction

2.1 Address Space and Memory Map

The address space of this LSI is 64 kbytes, which includes the program area and the data area.

Figures 2.1 show the memory map.



Notes: 1. This area cannot be used by an user when the on-chip emulator is in use.
2. When flash memory is programmed, this area is used by the programming control program. When the on-chip emulator is in use, this area is unavailable to the user.

Figure 2.1(1) H8/38704 Memory Map

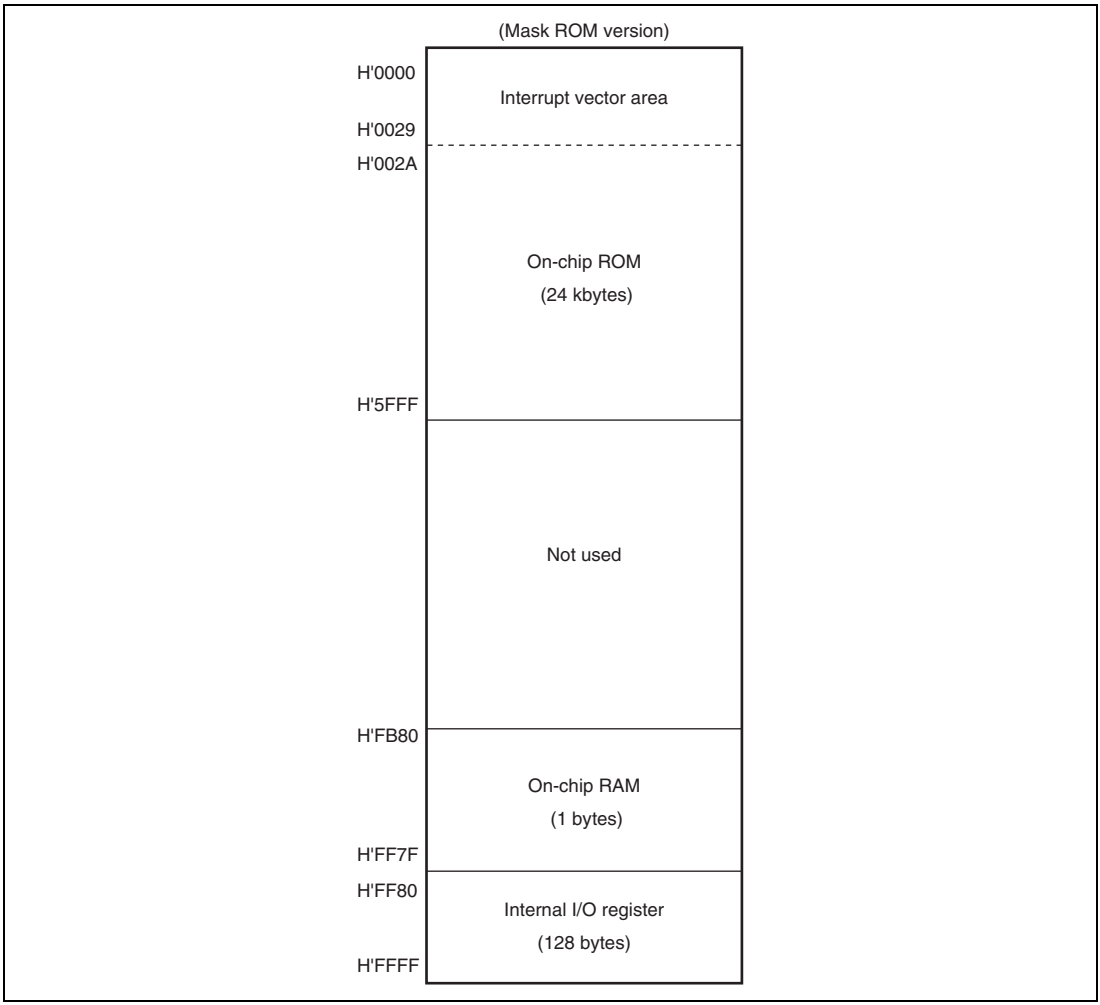
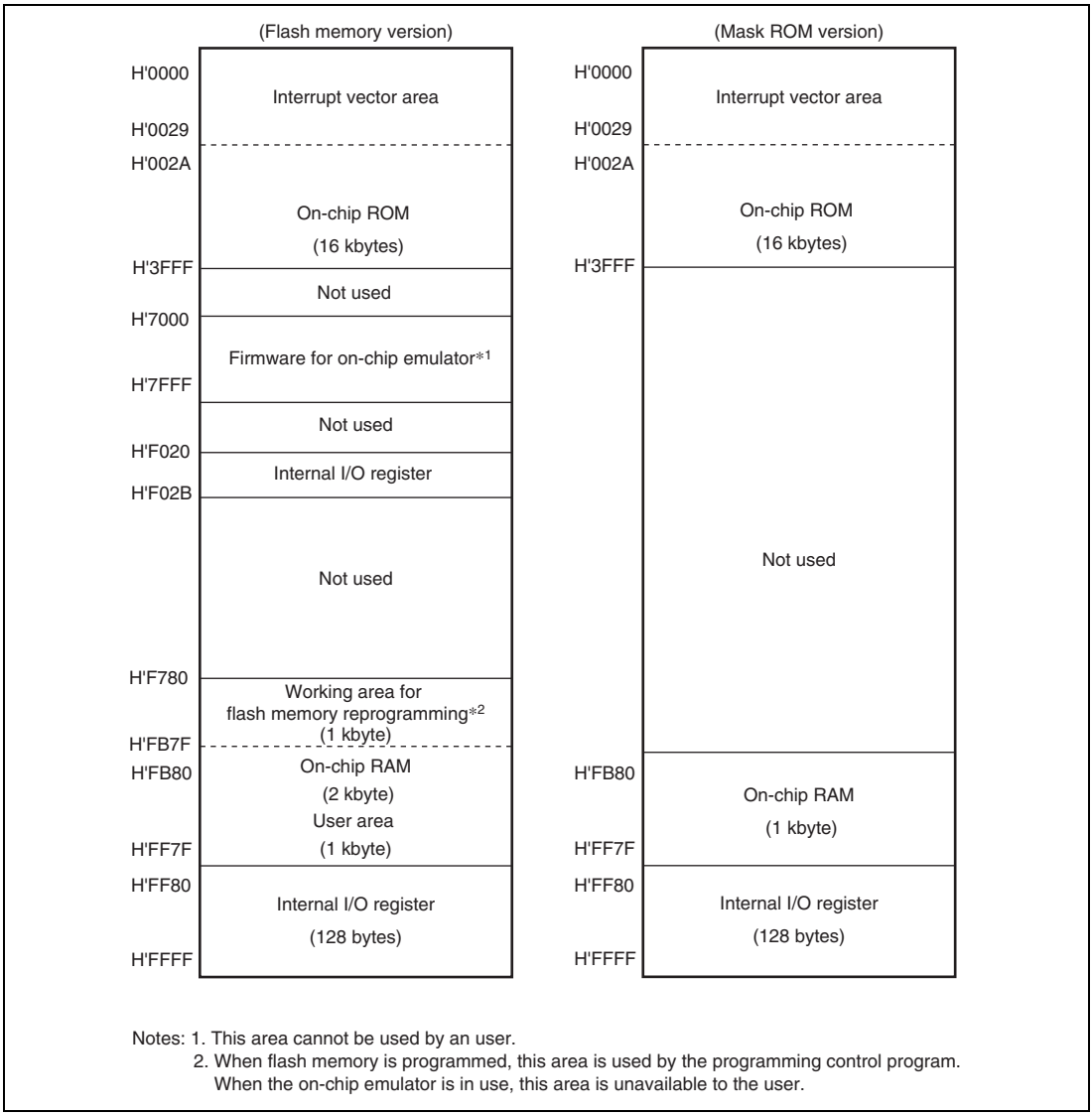


Figure 2.1(2) H8/38703 Memory Map



Notes: 1. This area cannot be used by an user.
2. When flash memory is programmed, this area is used by the programming control program. When the on-chip emulator is in use, this area is unavailable to the user.

Figure 2.1(3) H8/38702 Memory Map

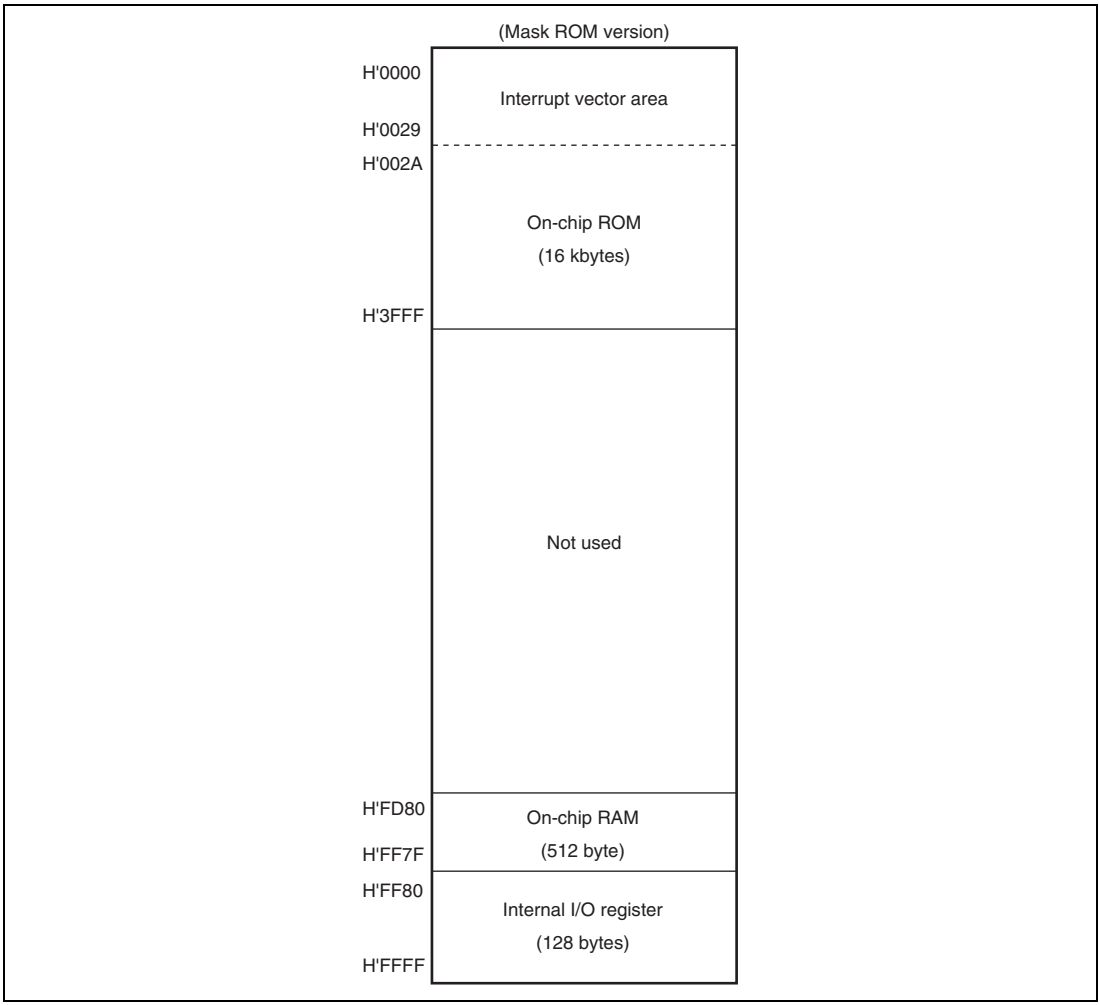


Figure 2.1(4) H8/38702S Memory Map

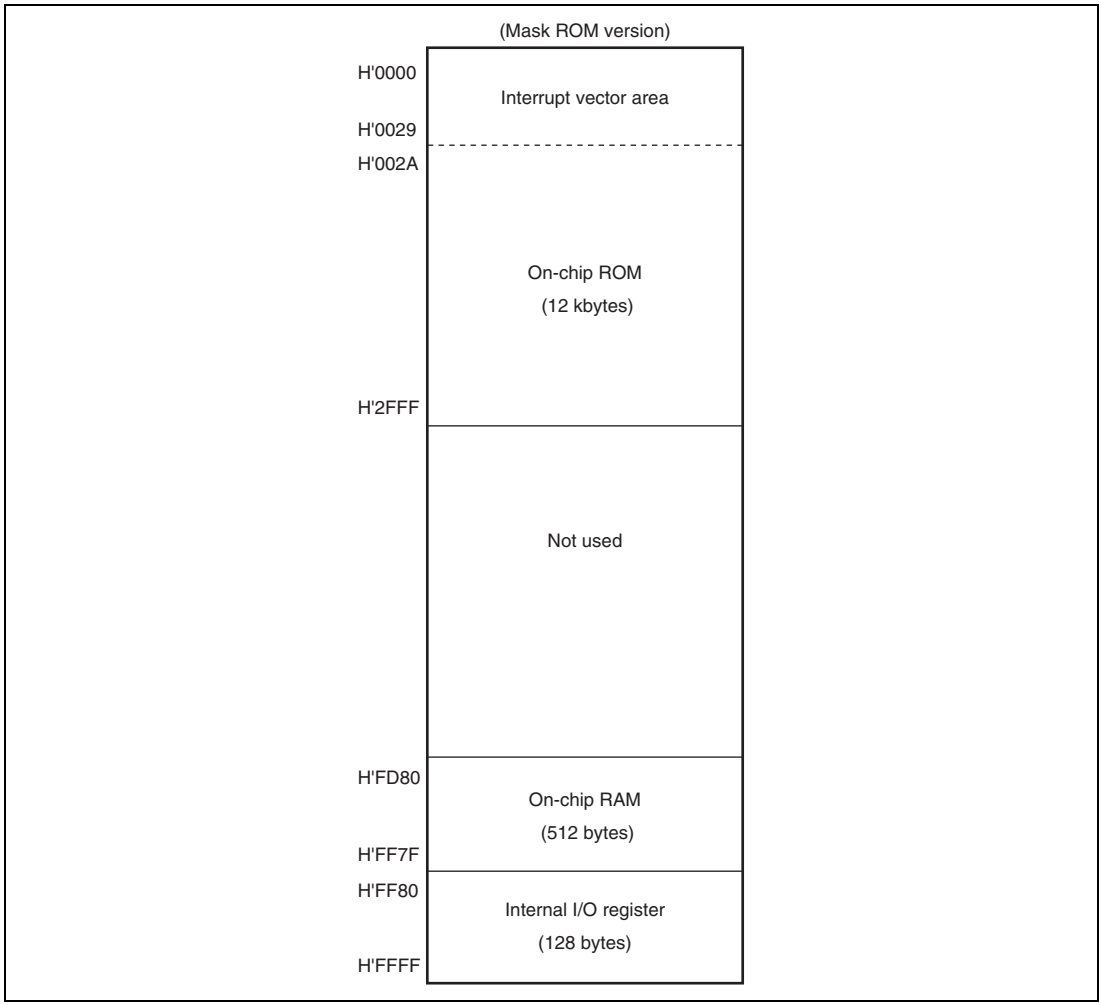


Figure 2.1(5) H8/38701S Memory Map

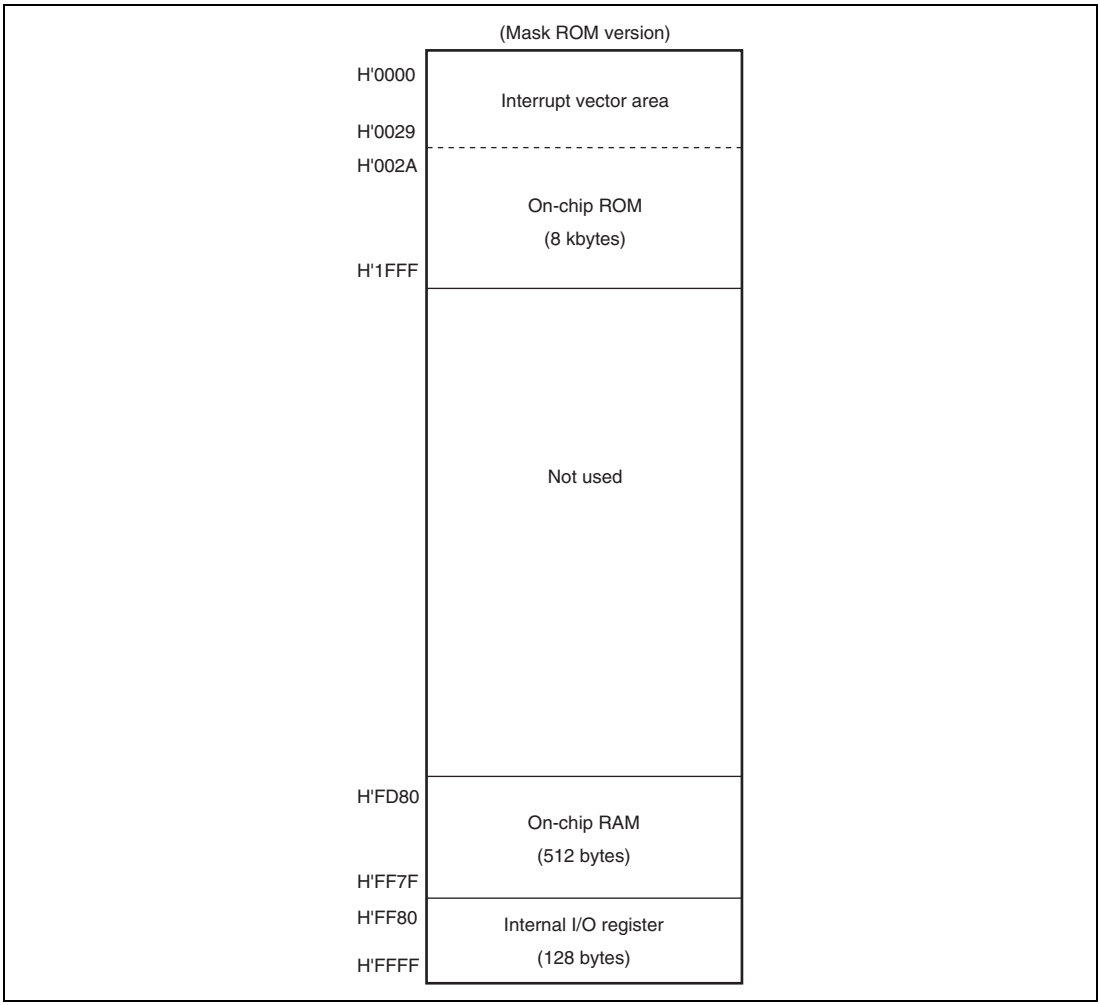


Figure 2.1(6) H8/38700S Memory Map

2.2 Register Configuration

The H8/300H CPU has the internal registers shown in figure 2.2. There are two types of registers; general registers and control registers. The control registers are a 24-bit program counter (PC), and an 8-bit condition-code register (CCR).

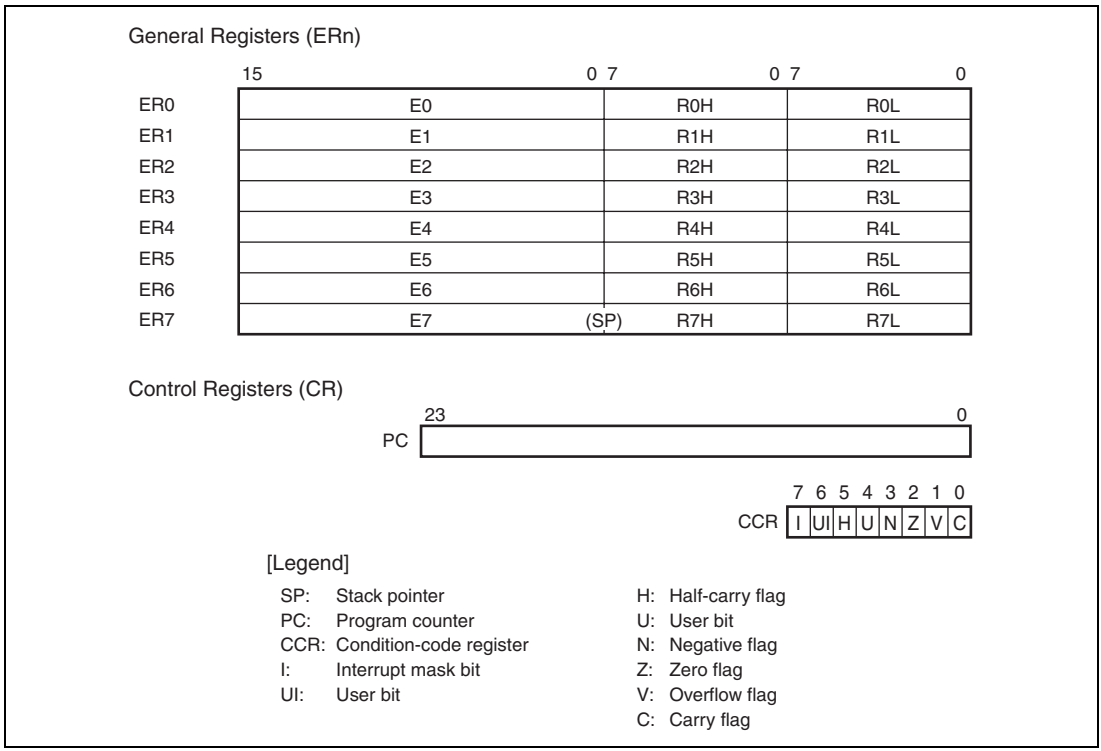


Figure 2.2 CPU Registers

2.2.1 General Registers

The H8/300H CPU has eight 32-bit general registers. These general registers are all functionally identical and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.3 illustrates the usage of the general registers. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

The ER registers divide into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum of sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

The R registers divide into 8-bit registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum of sixteen 8-bit registers.

The usage of each register can be selected independently.

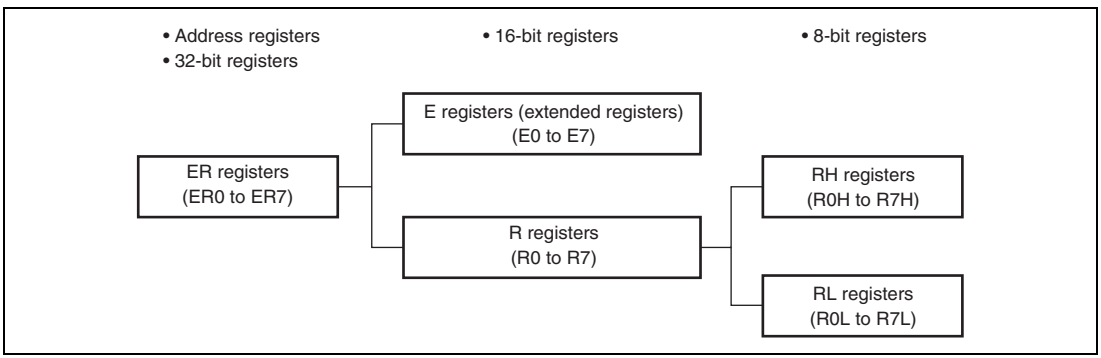


Figure 2.3 Usage of General Registers

General register ER7 has the function of the stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2.4 shows the relationship between the stack pointer and the stack area.

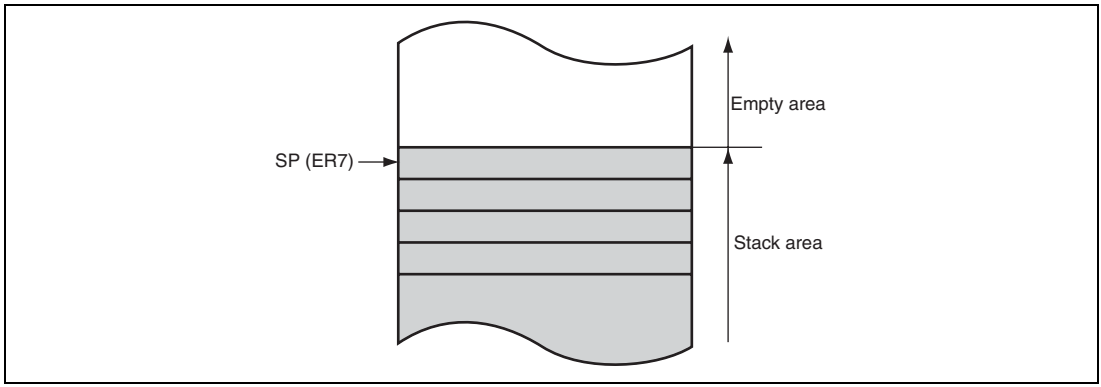


Figure 2.4 Relationship between Stack Pointer and Stack Area

2.2.2 Program Counter (PC)

This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 2 bytes (one word), so the least significant PC bit is ignored. (When an instruction is fetched, the least significant PC bit is regarded as 0). The PC is initialized when the start address is loaded by the vector address generated during reset exception-handling sequence.

2.2.3 Condition-Code Register (CCR)

This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags. The I bit is initialized to 1 by reset exception-handling sequence, but other bits are not initialized.

Some instructions leave flag bits unchanged. Operations can be performed on the CCR bits by the LDC, STC, ANDC, ORC, and XORC instructions. The N, Z, V, and C flags are used as branching conditions for conditional branch (Bcc) instructions.

For the action of each instruction on the flag bits, see appendix A.1, Instruction List.

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	Interrupt Mask Bit Masks interrupts other than NMI when set to 1. NMI is accepted regardless of the I bit setting. The I bit is set to 1 at the start of an exception-handling sequence.
6	UI	Undefined	R/W	User Bit Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions.
5	H	Undefined	R/W	Half-Carry Flag 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 cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.
4	U	Undefined	R/W	User Bit Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag Stores the value of the most significant bit of data as a sign bit.
2	Z	Undefined	R/W	Zero Flag Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.
0	C	Undefined	R/W	Carry Flag 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 indicate a carry The carry flag is also used as a bit accumulator by bit manipulation instructions.

2.3 Data Formats

The H8/300H CPU can process 1-bit, 4-bit (BCD), 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

2.3.1 General Register Data Formats

Figure 2.5 shows the data formats in general registers.

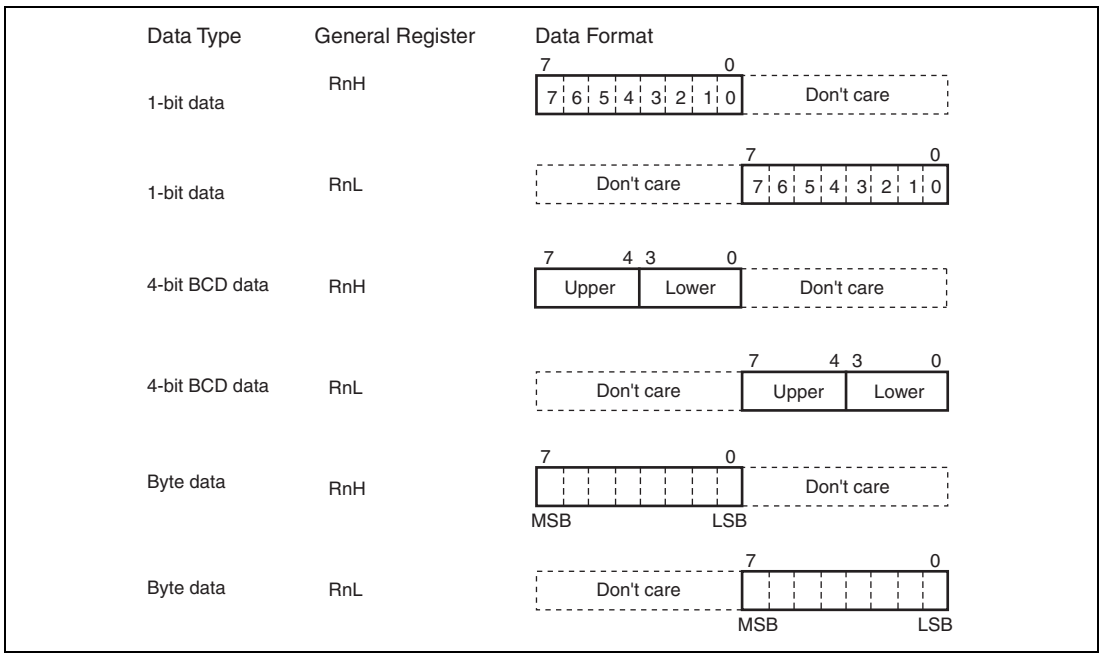


Figure 2.5 General Register Data Formats (1)

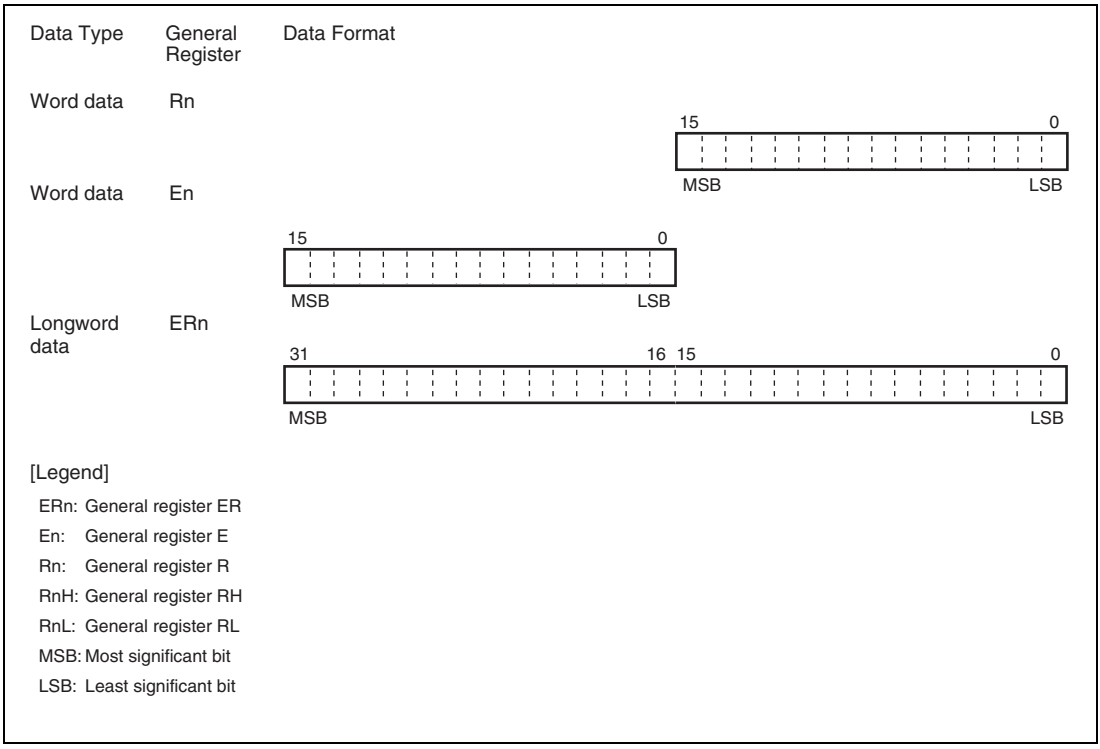


Figure 2.5 General Register Data Formats (2)

2.3.2 Memory Data Formats

Figure 2.6 shows the data formats in memory. The H8/300H CPU can access word data and longword data in memory, however word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, an address error does not occur, however the least significant bit of the address is regarded as 0, so access begins the preceding address. This also applies to instruction fetches.

When ER7 (SP) is used as an address register to access the stack area, the operand size should be word or longword.

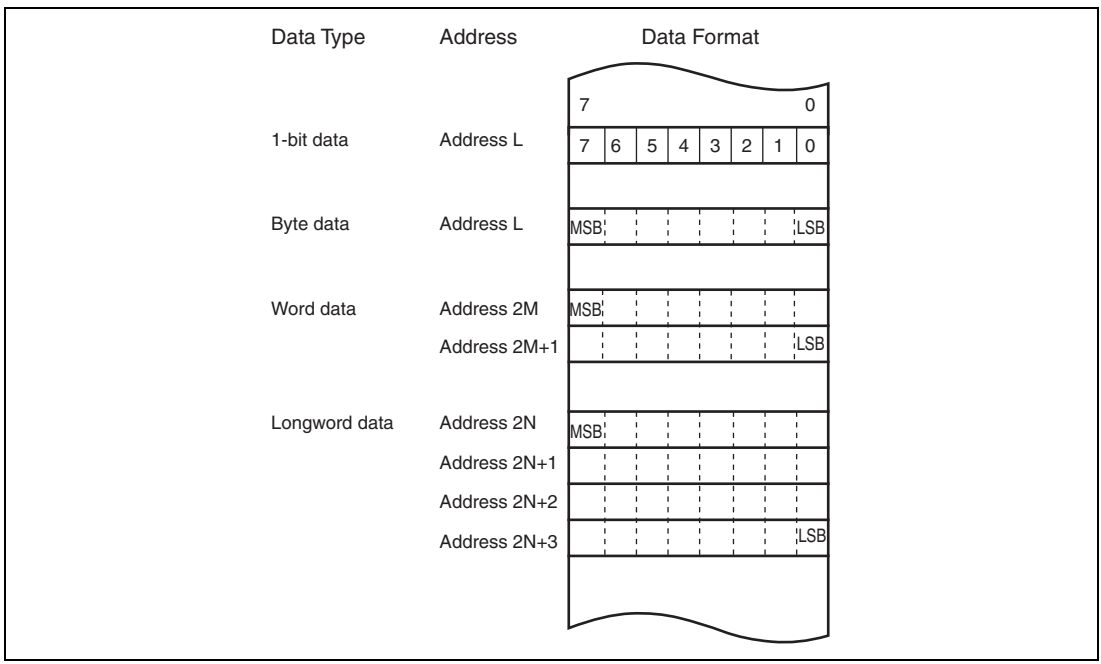


Figure 2.6 Memory Data Formats

2.4 Instruction Set

2.4.1 Table of Instructions Classified by Function

The H8/300H CPU has 62 instructions. Tables 2.2 to 2.9 summarize the instructions in each functional category. The notation used in tables 2.2 to 2.9 is defined in table 2.1.

Table 2.1 Operation Notation

Symbol	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register or address register)
(EAd)	Destination operand
(EAs)	Source operand
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
#IMM	Immediate data
disp	Displacement
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	Logical AND
∨	Logical OR
⊕	Logical XOR
→	Move
¬	NOT (logical complement)

Symbol	Description
:3/:8/:16/:24	3-, 8-, 16-, or 24-bit length

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers/address register (ER0 to ER7).

Table 2.2 Data Transfer Instructions

Instruction	Size*	Function
MOV	B/W/L	(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.
MOVFPE	B	(EAs) → Rd Cannot be used in this LSI.
MOVTPE	B	Rs → (EAs) Cannot be used in this LSI.
POP	W/L	@SP+ → Rn Pops a general register from the stack. POP.W Rn is identical to MOV.W @SP+, Rn. POP.L ERn is identical to MOV.L @SP+, ERn.
PUSH	W/L	Rn → @–SP Pushes a general register onto the stack. PUSH.W Rn is identical to MOV.W Rn, @–SP. PUSH.L ERn is identical to MOV.L ERn, @–SP.

Note: * Refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.3 Arithmetic Operations Instructions (1)

Instruction	Size*	Function
ADD SUB	B/W/L	Rd ± Rs → Rd, Rd ± #IMM → Rd Performs addition or subtraction on data in two general registers, or on immediate data and data in a general register (immediate byte data cannot be subtracted from byte data in a general register. Use the SUBX or ADD instruction.)
ADDX SUBX	B	Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd Performs addition or subtraction with carry on byte data in two general registers, or on immediate data and data in a general register.
INC DEC	B/W/L	Rd ± 1 → Rd, Rd ± 2 → Rd Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.)
ADDS SUBS	L	Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd Adds or subtracts the value 1, 2, or 4 to or from data in a 32-bit register.
DAA DAS	B	Rd (decimal adjust) → Rd Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 4-bit BCD data.
MULXU	B/W	Rd × Rs → Rd Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits.
MULXS	B/W	Rd × Rs → Rd Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits.
DIVXU	B/W	Rd ÷ Rs → Rd Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.

Note: * Refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.3 Arithmetic Operations Instructions (2)

Instruction	Size*	Function
DIVXS	B/W	Rd ÷ Rs → Rd Performs signed division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.
CMP	B/W/L	Rd – Rs, Rd – #IMM Compares data in a general register with data in another general register or with immediate data, and sets CCR bits according to the result.
NEG	B/W/L	0 – Rd → Rd Takes the two's complement (arithmetic complement) of data in a general register.
EXTU	W/L	Rd (zero extension) → Rd Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by padding with zeros on the left.
EXTS	W/L	Rd (sign extension) → Rd Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by extending the sign bit.

Note: * Refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.4 Logic Operations Instructions

Instruction	Size*	Function
AND	B/W/L	Rd ∧ Rs → Rd, Rd ∧ #IMM → Rd Performs a logical AND operation on a general register and another general register or immediate data.
OR	B/W/L	Rd ∨ Rs → Rd, Rd ∨ #IMM → Rd Performs a logical OR operation on a general register and another general register or immediate data.
XOR	B/W/L	Rd ⊕ Rs → Rd, Rd ⊕ #IMM → Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data.
NOT	B/W/L	¬ (Rd) → (Rd) Takes the one's complement (logical complement) of general register contents.

Note: * Refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.5 Shift Instructions

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

Note: * Refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.6 Bit Manipulation Instructions

Instruction	Size*	Function
BSET	B	1 → (<bit-No.> of <EAd>) Sets a specified bit in a general register or memory operand 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 operand 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 operand. 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 operand 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 carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIAND	B	C ∧ ¬ (<bit-No.> of <EAd>) → C ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BOR	B	C ∨ (<bit-No.> of <EAd>) → C ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIOR	B	C ∨ ¬ (<bit-No.> of <EAd>) → C ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → C XORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIXOR	B	C ⊕ ¬ (<bit-No.> of <EAd>) → C XORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.

Instruction	Size*	Function
BLD	B	(<bit-No.> of <EAd>) → C Transfers a specified bit in a general register or memory operand to the carry flag.
BILD	B	¬ (<bit-No.> of <EAd>) → C Transfers the inverse of a specified bit in a general register or memory operand to the carry flag. The bit number is specified by 3-bit immediate data.
BST	B	C → (<bit-No.> of <EAd>) Transfers the carry flag value to a specified bit in a general register or memory operand.
BIST	B	¬ C → (<bit-No.> of <EAd>) Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data.

Note: * Refers to the operand size.
B: Byte

Table 2.7 Branch Instructions

Instruction	Size	Function
Bcc*	—	Branches to a specified address if a specified condition is true. The branching conditions are listed below.

Mnemonic	Description	Condition
BRA(BT)	Always (true)	Always
BRN(BF)	Never (false)	Never
BHI	High	$C \vee Z = 0$
BLS	Low or same	$C \vee Z = 1$
BCC(BHS)	Carry clear (high or same)	$C = 0$
BCS(BLO)	Carry set (low)	$C = 1$
BNE	Not equal	$Z = 0$
BEQ	Equal	$Z = 1$
BVC	Overflow clear	$V = 0$
BVS	Overflow set	$V = 1$
BPL	Plus	$N = 0$
BMI	Minus	$N = 1$
BGE	Greater or equal	$N \oplus V = 0$
BLT	Less than	$N \oplus V = 1$
BGT	Greater than	$Z \vee (N \oplus V) = 0$
BLE	Less or equal	$Z \vee (N \oplus V) = 1$

JMP	—	Branches unconditionally to a specified address.
BSR	—	Branches to a subroutine at a specified address.
JSR	—	Branches to a subroutine at a specified address.
RTS	—	Returns from a subroutine

Note: * Bcc is the general name for conditional branch instructions.

Table 2.8 System Control Instructions

Instruction	Size*	Function
RTE	—	Returns from an exception-handling routine.
SLEEP	—	Causes a transition to a power-down state.
LDC	B/W	(EAs) → CCR Moves the source operand contents to the CCR. The CCR size is one byte, but in transfer from memory, data is read by word access.
STC	B/W	CCR → (EAd) Transfers the CCR contents to a destination location. The condition code register size is one byte, but in transfer to memory, data is written by word access.
ANDC	B	CCR ∧ #IMM → CCR Logically ANDs the CCR with immediate data.
ORC	B	CCR ∨ #IMM → CCR Logically ORs the CCR with immediate data.
XORC	B	CCR ⊕ #IMM → CCR Logically XORs the CCR with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

Note: * Refers to the operand size.
B: Byte
W: Word

Table 2.9 Block Data Transfer Instructions

Instruction	Size	Function
EEPMOV.B	—	if R4L ≠ 0 then Repeat @ER5+ → @ER6+, R4L–1 → R4L Until R4L = 0 else next;
EEPMOV.W	—	if R4 ≠ 0 then Repeat @ER5+ → @ER6+, R4–1 → R4 Until R4 = 0 else next; Transfers a data block. Starting from the address set in ER5, transfers data for the number of bytes set in R4L or R4 to the address location set in ER6. Execution of the next instruction begins as soon as the transfer is completed.

2.4.2 Basic Instruction Formats

H8/300H CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op), a register field (r), an effective address extension (EA), and a condition field (cc).

Figure 2.7 shows examples of instruction formats.

(1) Operation Field

Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.

(2) Register Field

Specifies a general register. Address registers are specified by 3 bits, and data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.

(3) Effective Address Extension

8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement. A24-bit address or displacement is treated as a 32-bit data in which the first 8 bits are 0 (H'00).

(4) Condition Field

Specifies the branching condition of Bcc instructions.

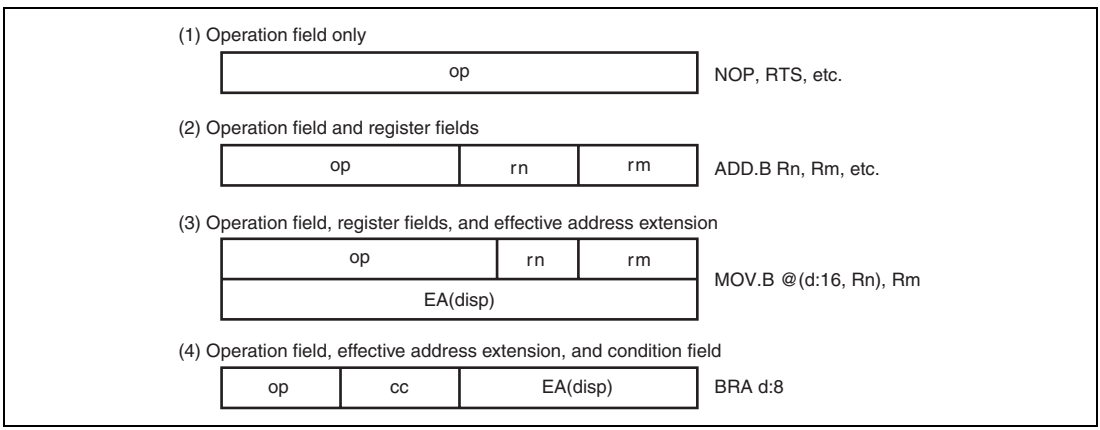


Figure 2.7 Instruction Formats

2.5 Addressing Modes and Effective Address Calculation

The following describes the H8/300H CPU. In this LSI, the upper eight bits are ignored in the generated 24-bit address, so the effective address is 16 bits.

2.5.1 Addressing Modes

The H8/300H CPU supports the eight addressing modes listed in table 2.10. Each instruction uses a subset of these addressing modes. Addressing modes that can be used differ depending on the instruction. For details, refer to appendix A.4, Combinations of Instructions and Addressing Modes.

Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit-manipulation instructions use register direct, register indirect, or the absolute addressing mode (@aa:8) to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

Table 2.10 Addressing Modes

No.	Addressing Mode	Symbol
1	Register direct	Rn
2	Register indirect	@ERn
3	Register indirect with displacement	@(d:16,ERn)/@(d:24,ERn)
4	Register indirect with post-increment Register indirect with pre-decrement	@ERn+ @–ERn
5	Absolute address	@aa:8/@aa:16/@aa:24
6	Immediate	#xx:8/#xx:16/#xx:32
7	Program-counter relative	@(d:8,PC)/@(d:16,PC)
8	Memory indirect	@@aa:8

(1) Register Direct—Rn

The register field of the instruction specifies an 8-, 16-, or 32-bit general register containing the operand. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

(2) Register Indirect—@ERn

The register field of the instruction code specifies an address register (ERn), the lower 24 bits of which contain the address of the operand on memory.

(3) Register Indirect with Displacement—@(d:16, ERn) or @(d:24, ERn)

A 16-bit or 24-bit displacement contained in the instruction is added to an address register (ERn) specified by the register field of the instruction, and the lower 24 bits of the sum the address of a memory operand. A 16-bit displacement is sign-extended when added.

(4) Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn

- Register indirect with post-increment—@ERn+

  The register field of the instruction code specifies an address register (ERn) the lower 24 bits of which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents (32 bits) and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access. For the word or longword access, the register value should be even.
- Register indirect with pre-decrement—@-ERn

  The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the lower 24 bits of the result is the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access. For the word or longword access, the register value should be even.

(5) Absolute Address—@aa:8, @aa:16, @aa:24

The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24)

For an 8-bit absolute address, the upper 16 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address the upper 8 bits are a sign extension. A 24-bit absolute address can access the entire address space.

The access ranges of absolute addresses for this LSI are those shown in table 2.11, because the upper 8 bits are ignored.

Table 2.11 Absolute Address Access Ranges

Absolute Address	Access Range
8 bits (@aa:8)	H'FF00 to H'FFFF
16 bits (@aa:16)	H'0000 to H'FFFF
24 bits (@aa:24)	H'0000 to H'FFFF

(6) Immediate—#xx:8, #xx:16, or #xx:32

The instruction contains 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data as an operand.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, specifying a bit number.

(7) Program-Counter Relative—@(d:8, PC) or @(d:16, PC)

This mode is used in the BSR instruction. An 8-bit or 16-bit displacement contained in the instruction is sign-extended and added to the 24-bit PC contents to generate a branch address. The PC value to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is –126 to +128 bytes (–63 to +64 words) or –32766 to +32768 bytes (–16383 to +16384 words) from the branch instruction. The resulting value should be an even number.

(8) Memory Indirect—@@aa:8

This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand. This memory operand contains a branch address. The memory operand is accessed in words, generating a 16-bit branch address. Figure 2.8 shows how to specify branch address for in memory indirect mode. The upper bits of the absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF).

Note that the first part of the address range is also the exception vector area.

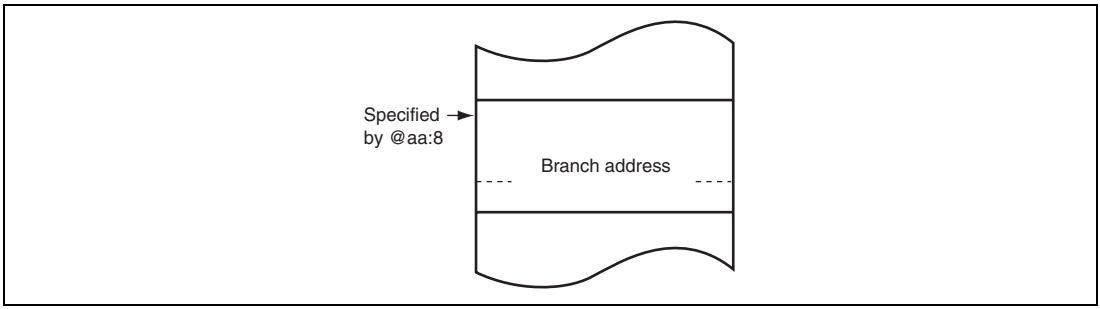


Figure 2.8 Branch Address Specification in Memory Indirect Mode

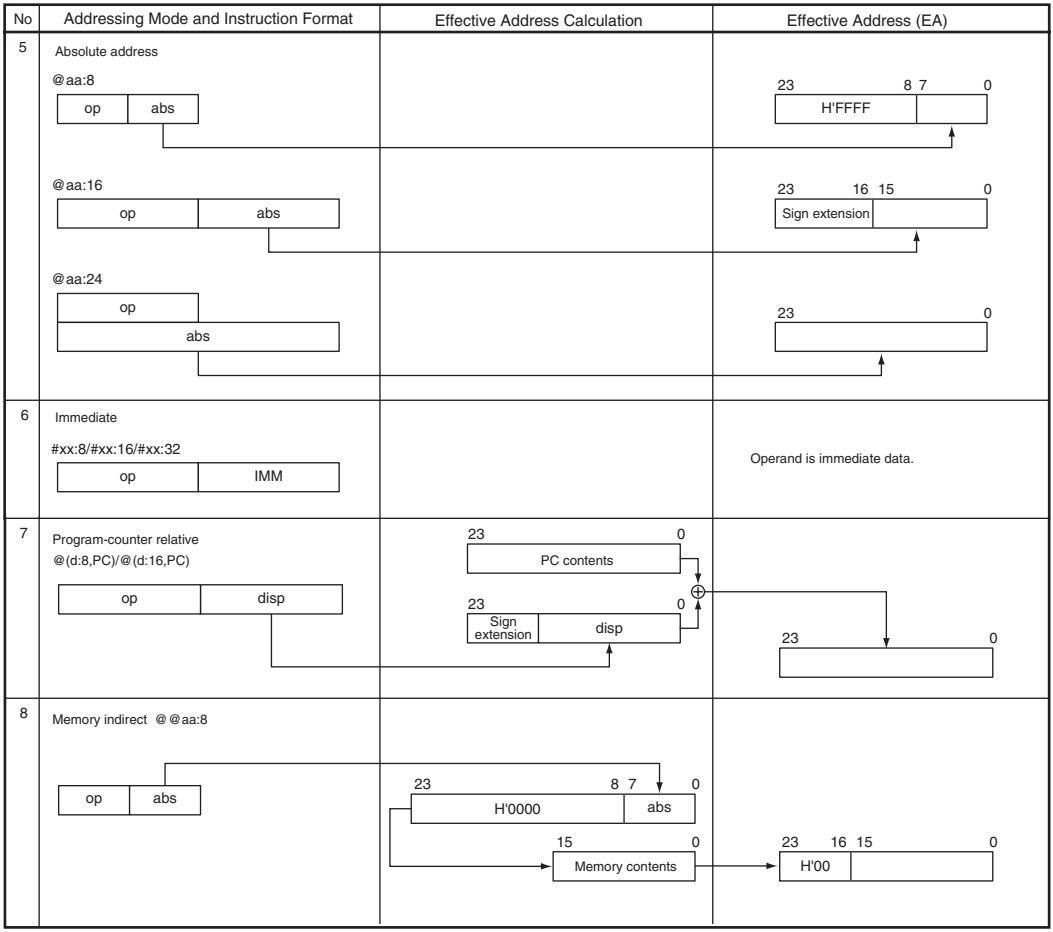
2.5.2 Effective Address Calculation

Table 2.12 indicates how effective addresses are calculated in each addressing mode. In this LSI, the upper 8 bits of the effective address are ignored in order to generate a 16-bit effective address.

Table 2.12 Effective Address Calculation (1)

No	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Register direct(Rn) op | rm | rn		Operand is general register contents.
2	Register indirect(@ERn) op | r	31 … 0: General register contents	23 … 0
3	Register indirect with displacement @(d:16,ERn) or @(d:24,ERn) op | r | disp	31 … 0: General register contents 31 … 0: Sign extension | disp	23 … 0
4	Register indirect with post-increment or pre-decrement •Register indirect with post-increment @ERn+ op | r •Register indirect with pre-decrement @-ERn op | r	31 … 0: General register contents 1, 2, or 4 The value to be added or subtracted is 1 when the operand is byte size, 2 for word size, and 4 for longword size.	23 … 0

Table 2.12 Effective Address Calculation (2)



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

2.6 Basic Bus Cycle

CPU operation is synchronized by a system clock (φ) or a subclock (ϕ_{SUB}). The period from a rising edge of φ or ϕ_{SUB} to the next rising edge is called one state. A bus cycle consists of two states or three states. The cycle differs depending on whether access is to on-chip memory or to on-chip peripheral modules.

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

Access to on-chip memory takes place in two states. The data bus width is 16 bits, allowing access in byte or word size. Figure 2.9 shows the on-chip memory access cycle.

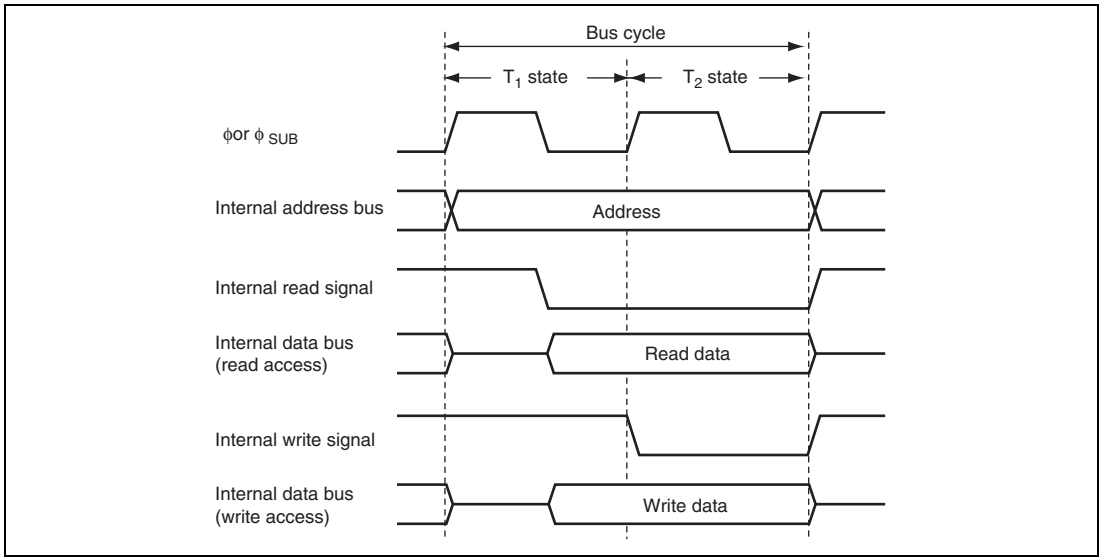


Figure 2.9 On-Chip Memory Access Cycle

2.6.2 On-Chip Peripheral Modules

On-chip peripheral modules are accessed in two states or three states. The data bus width is 8 bits or 16 bits depending on the register. For description on the data bus width and number of accessing states of each register, refer to section 13.1, Register Addresses (Address Order). Registers with 16-bit data bus width can be accessed by word size only. Registers with 8-bit data bus width can be accessed by byte or word size. When a register with 8-bit data bus width is accessed by word size, a bus cycle occurs twice. In two-state access, the operation timing is the same as that for on-chip memory.

Figure 2.10 shows the operation timing in the case of three-state access to an on-chip peripheral module.

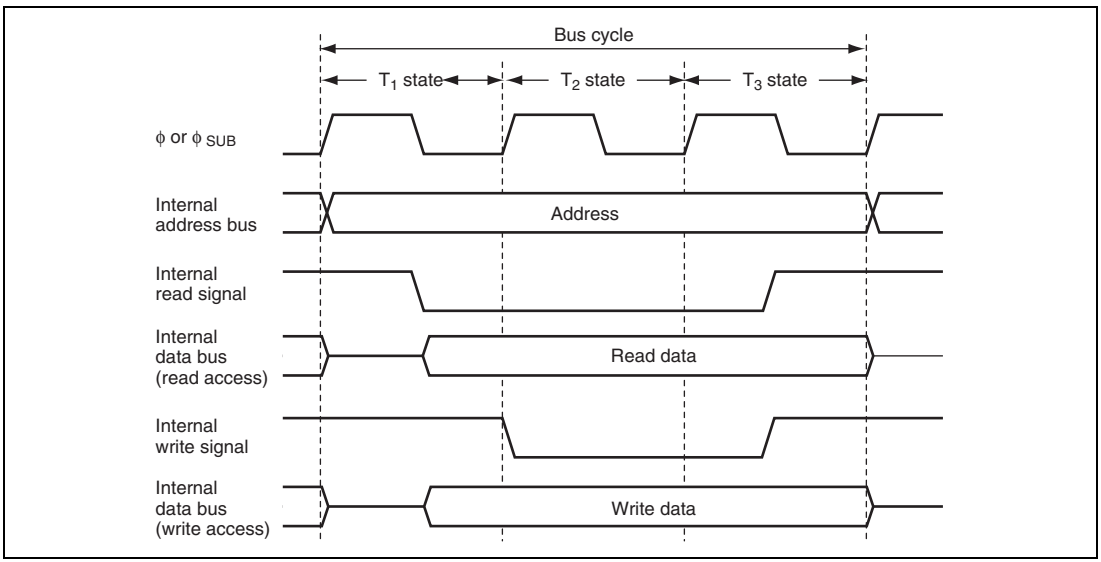


Figure 2.10 On-Chip Peripheral Module Access Cycle (3-State Access)

2.7 CPU States

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. For the program halt state, there are sleep (high-speed or medium-speed) mode, standby mode, watch mode, and subsleep mode. These states are shown in figure 2.11. Figure 2.12 shows the state transitions. For details on program execution state and program halt state, refer to section 5, Power-Down Modes. For details on exception handling, refer to section 3, Exception Handling.

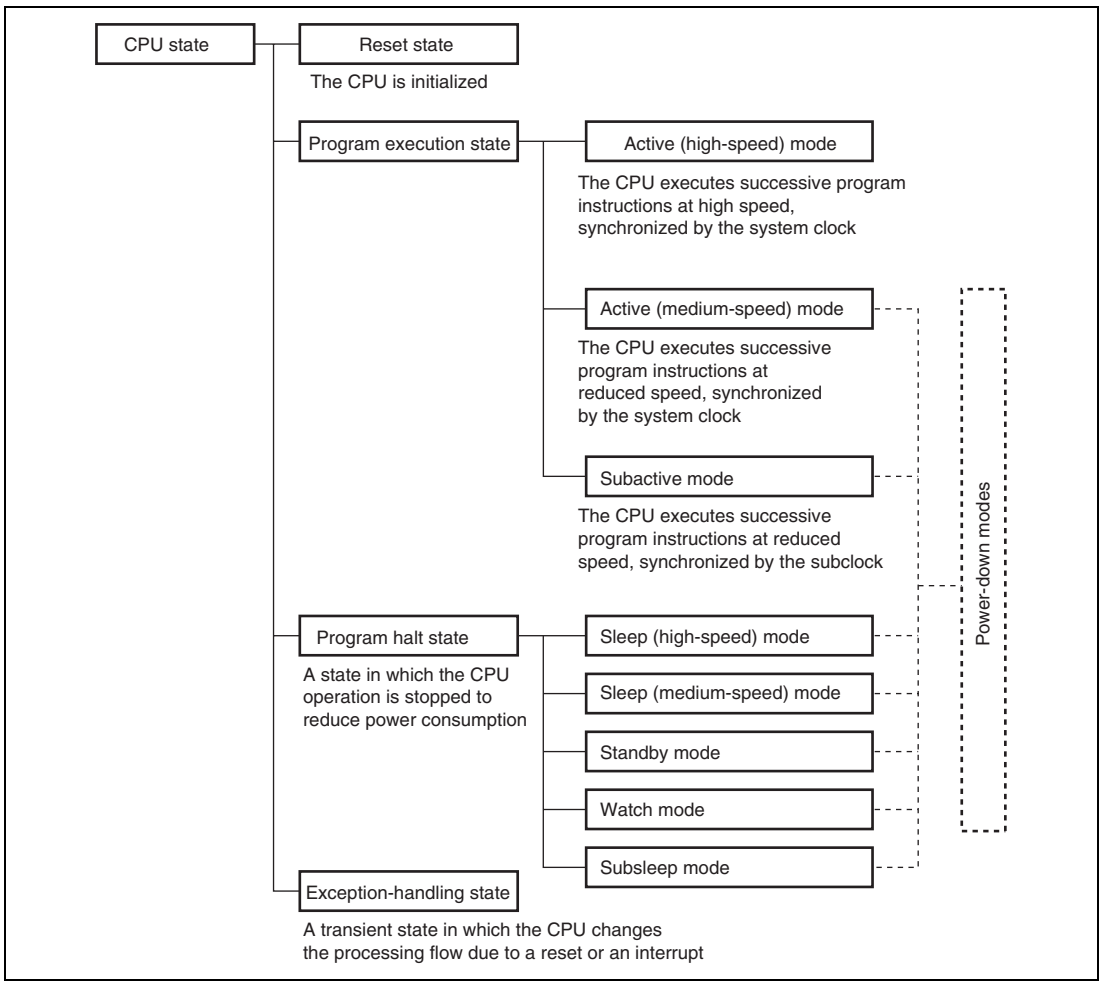


Figure 2.11 CPU Operating States

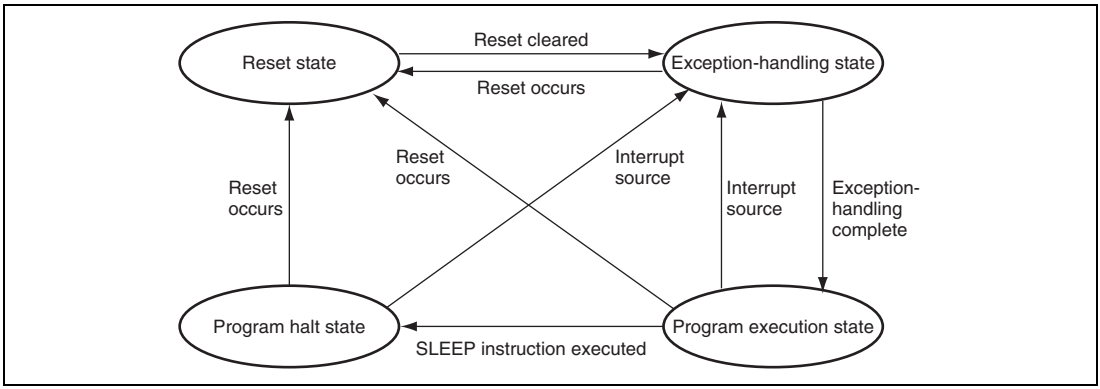


Figure 2.12 State Transitions

2.8 Usage Notes

2.8.1 Notes on Data Access to Empty Areas

The address space of this LSI includes empty areas in addition to the ROM, RAM, and on-chip I/O registers areas available to the user. When data is transferred from CPU to empty areas, the transferred data will be lost. This action may also cause the CPU to malfunction. When data is transferred from an empty area to CPU, the contents of the data cannot be guaranteed.

2.8.2 EEPMOV Instruction

EEPMOV is a block-transfer instruction and transfers the byte size of data indicated by R4L, which starts from the address indicated by R5, to the address indicated by R6. Set R4L and R6 so that the end address of the destination address (value of R6 + R4L) does not exceed H'FFFF (the value of R6 must not change from H'FFFF to H'0000 during execution).

2.8.3 Bit-Manipulation Instruction

The BSET, BCLR, BNOT, BST, and BIST instructions read data from the specified address in byte units, manipulate the data of the target bit, and write data to the same address again in byte units. Special care is required when using these instructions in cases where two registers are assigned to the same address, or when a bit is directly manipulated for a port or a register containing a write-only bit, because this may rewrite data of a bit other than the bit to be manipulated.

(1) Bit manipulation for two registers assigned to the same address

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

Figure 2.13 shows an example of a timer in which two timer registers are assigned to 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 takes place.

1. Data is read in byte units.
2. The CPU sets or resets the bit to be manipulated with the bit-manipulation instruction.
3. The written data is written again in byte units to the timer load register.

The timer 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 counter may be modified and the modified value may be written to the timer load register.

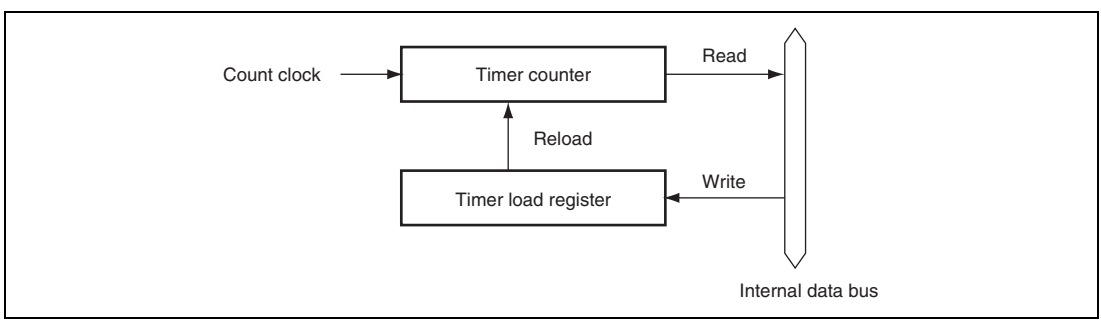


Figure 2.13 Example of Timer Configuration with Two Registers Allocated to Same Address

Example 2: When the BSET instruction is executed for port 5

P57 and P56 are input pins, with a low-level signal input at P57 and a high-level signal input at P56. P55 to P50 are output pins and output low-level signals. An example to output a high-level signal at P50 with a BSET instruction is shown below.

- Prior to executing BSET instruction

	P57	P56	P55	P54	P53	P52	P51	P50
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
PCR5	0	0	1	1	1	1	1	1
PDR5	1	0	0	0	0	0	0	0

- BSET instruction executed

```
BSET    #0,     @PDR5
```

The BSET instruction is executed for port 5.

- After executing BSET instruction

	P57	P56	P55	P54	P53	P52	P51	P50
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
PCR5	0	0	1	1	1	1	1	1
PDR5	0	1	0	0	0	0	0	1

- Description on operation

1. When the BSET instruction is executed, first the CPU reads port 5.
   Since P57 and P56 are input pins, the CPU reads the pin states (low-level and high-level input).
   P55 to P50 are output pins, so the CPU reads the value in PDR5. In this example PDR5 has a value of H'80, but the value read by the CPU is H'40.
2. Next, the CPU sets bit 0 of the read data to 1, changing the PDR5 data to H'41.

3. Finally, the CPU writes H'41 to PDR5, completing execution of BSET instruction.

   As a result of the BSET instruction, bit 0 in PDR5 becomes 1, and P50 outputs a high-level signal. However, bits 7 and 6 of PDR5 end up with different values. To prevent this problem, store a copy of the PDR5 data in a work area in memory. Perform the bit manipulation on the data in the work area, then write this data to PDR5.

- Prior to executing BSET instruction

```
MOV.B   #H'80, R0L
MOV.B   R0L,   @RAM0
MOV.B   R0L,   @PDR5
```

The PDR5 value (H'80) is written to a work area in memory (RAM0) as well as to PDR5.

	P57	**P56**	**P55**	**P54**	**P53**	**P52**	**P51**	**P50**
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
PCR5	0	0	1	1	1	1	1	1
PDR5	1	0	0	0	0	0	0	0
RAM0	1	0	0	0	0	0	0	0

- BSET instruction executed

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

The BSET instruction is executed designating the PDR5 work area (RAM0).

- After executing BSET instruction

```
MOV.B   @RAM0, R0L
MOV.B   R0L,  @PDR5
```

The work area (RAM0) value is written to PDR5.

	P57	**P56**	**P55**	**P54**	**P53**	**P52**	**P51**	**P50**
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
PCR5	0	0	1	1	1	1	1	1
PDR5	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: BCLR instruction executed designating port 5 control register PCR5

P57 and P56 are input pins, with a low-level signal input at P57 and a high-level signal input at P56. P55 to P50 are output pins that output low-level signals. An example of setting the P50 pin as an input pin by the BCLR instruction is shown below. It is assumed that a high-level signal will be input to this input pin.

- Prior to executing BCLR instruction

	P57	P56	P55	P54	P53	P52	P51	P50
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
PCR5	0	0	1	1	1	1	1	1
PDR5	1	0	0	0	0	0	0	0

- BCLR instruction executed

```
BCLR    #0,     @PCR5
```

The BCLR instruction is executed for PCR5.

- After executing BCLR instruction

	P57	P56	P55	P54	P53	P52	P51	P50
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
PCR5	1	1	1	1	1	1	1	0
PDR5	1	0	0	0	0	0	0	0

- Description on operation

1. When the BCLR instruction is executed, first the CPU reads PCR5. Since PCR5 is a write-only register, the CPU reads a value of H'FF, even though the PCR5 value is actually H'3F.
2. Next, the CPU clears bit 0 in the read data to 0, changing the data to H'FE.
3. Finally, H'FE is written to PCR5 and BCLR instruction execution ends.

   As a result of this operation, bit 0 in PCR5 becomes 0, making P50 an input port. However, bits 7 and 6 in PCR5 change to 1, so that P57 and P56 change from input pins to output pins. To prevent this problem, store a copy of the PDR5 data in a work area in memory and manipulate data of the bit in the work area, then write this data to PDR5.

- Prior to executing BCLR instruction

```
MOV.B   #H'3F, R0L
MOV.B   R0L,   @RAM0
MOV.B   R0L,   @PCR5
```

The PCR5 value (H'3F) is written to a work area in memory (RAM0) as well as to PCR5.

	P57	**P56**	**P55**	**P54**	**P53**	**P52**	**P51**	**P50**
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
PCR5	0	0	1	1	1	1	1	1
PDR5	1	0	0	0	0	0	0	0
RAM0	0	0	1	1	1	1	1	1

- BCLR instruction executed

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

The BCLR instructions executed for the PCR5 work area (RAM0).

- After executing BCLR instruction

```
MOV.B   @RAM0, R0L
MOV.B   R0L,  @PCR5
```

The work area (RAM0) value is written to PCR5.

	P57	**P56**	**P55**	**P54**	**P53**	**P52**	**P51**	**P50**
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
PCR5	0	0	1	1	1	1	1	0
PDR5	1	0	0	0	0	0	0	0
RAM0	0	0	1	1	1	1	1	0

Section 3 Exception Handling

Exception handling may be caused by a reset or interrupts.

- Reset

  A reset has the highest exception priority. Exception handling starts as soon as the reset is cleared by the $\overline{\text{RES}}$ pin. The chip is also reset when the watchdog timer overflows, and exception handling starts. Exception handling is the same as exception handling by the $\overline{\text{RES}}$ pin.
- Interrupts

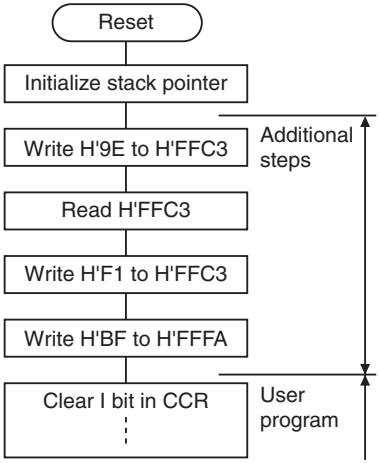
  External interrupts and internal interrupts are masked by the I bit in CCR, and kept masked while the I bit is set to 1. Exception handling starts when the current instruction or exception handling ends, if an interrupt request has been issued.

The following notes apply to the HD64F38704 and HD64F38702.

- Issue

  Depending on the circuitry status at power-on, a vector 17 (system reservation) interrupt request may be generated. If bit I in CCR is cleared to 0, this interrupt will be accepted just like any other internal interrupt. This can cause processing exceptions to occur, and program execution will eventually halt since there is no procedure for clearing the interrupt request flag in question.
- Countermeasure

  To prevent the above issue from occurring, it is recommended that the following steps be added to programs written for the product.


Reset
Initialize stack pointer
Write H'9E to H'FFC3
Read H'FFC3
Write H'F1 to H'FFC3
Write H'BF to H'FFFA
Clear I bit in CCR
Additional steps
User program

The following is an example in assembler.

```
  .ORG  H'0000
  .DATA.W      INIT
  .ORG  H'0100
INIT:
 MOV.W  #H'FF80:16,SP

 MOV.B  #H'9E:8,R0L
 MOV.B  R0L,@H'FFC3:8
 MOV.B  @H'FFC3:8,R0L
 MOV.B  #H'F1:8,R0L
 MOV.B  R0L,@H'FFC3:8
 MOV.B  #H'BF:8,R0L
 MOV.B  R0L,@H'FFFA:8
 ANDC.B #H'7F:8,CCR       ; user program
```

The following is an example in C.

```
   void powerON_Reset(void)
   {
   // ----------------------------------------------------
            unsigned char dummy;
            *((volatile unsigned char *)0xffc3)= 0x9e;
            dummy = *((volatile unsigned char *)0xffc3);
            *((volatile unsigned char *)0xffc3)= 0xf1;
            *((volatile unsigned char *)0xfffa)= 0xbf;
   // ----------------------------------------------------
            set_imask_ccr(0);       // clear I bit
                                    // user program
   }
```

On the mask ROM version of the product, user programs may be used as is (including the additional steps described above) or without the additional steps.

3.1 Exception Sources and Vector Address

Table 3.1 shows the vector addresses and priority of each exception handling. When more than one interrupt is requested, handling is performed from the interrupt with the highest priority.

Table 3.1 Exception Sources and Vector Address

Relative Module	Exception Sources	Vector Number	Vector Address	Priority
$\overline{\text{RES}}$ pin, WDT	Reset	0	H'0000 to H'0001	High
—	Reserved for system use	1 to 3	H'0002 to H'0007	
External interrupt pin	IRQ0	4	H'0008 to H'0009	
	IRQ1	5	H'000A to H'000B	
	IRQAEC	6	H'000C to H'000D	
—	Reserved for system use	7, 8	H'000E to H'0011	
External interrupt pin	WKP0 WKP1 WKP2 WKP3 WKP4 WKP5 WKP6 WKP7	9	H'0012 to H'0013	
—	Reserved for system use	10	H'0014 to H'0015	
Timer A	Timer A overflow	11	H'0016 to H'0017	
Asynchronous event counter	Asynchronous event counter overflow	12	H'0018 to H'0019	
—	Reserved for system use	13	H'001A to H'001B	
Timer F	Timer FL compare match Timer FL overflow	14	H'001C to H'001D	
	Timer FH compare match Timer FH overflow	15	H'001E to H'001F	
—	Reserved for system use	16, 17	H’0020 to H’0023	
SCI3	Transmit end Transmit data empty Transmit data full Receive error	18	H'0024 to H'0025	
A/D converter	A/D conversion end	19	H'0026 to H'0027	
CPU	Direct transition by SLEEP instruction	20	H'0028 to H'0029	Low

3.2 Register Descriptions

Interrupts are controlled by the following registers.

- Interrupt edge select register (IEGR)
- Interrupt enable register 1 (IENR1)
- Interrupt enable register 2 (IENR2)
- Interrupt request register 1 (IRR1)
- Interrupt request register 2 (IRR2)
- Wakeup interrupt request register (IWPR)
- Wakeup edge select register (WEGR)

3.2.1 Interrupt Edge Select Register (IEGR)

IEGR selects the direction of an edge that generates interrupt requests of pins and $\overline{\text{IRQ1}}$ and $\overline{\text{IRQ0}}$.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 1	—	Reserved These bits are always read as 1.
4 to 2	—	—	W	Reserved The write value should always be 0.
1	IEG1	0	R/W	IRQ1 and IRQ0 Edge Select
0	IEG0	0	R/W	0: Falling edge of $\overline{\text{IRQn}}$ pin input is detected 1: Rising edge of $\overline{\text{IRQn}}$ pin input is detected (n = 1 or 0)

3.2.2 Interrupt Enable Register 1 (IENR1)

IENR1 enables timers and external pin interrupts.

Bit	Bit Name	Initial Value	R/W	Description
7	IENTA	0	R/W	Timer A interrupt enable Enables or disables timer A overflow interrupt requests. 0: Disables timer A interrupt requests 1: Enables timer A interrupt requests
6	—	—	W	Reserved The write value should always be 0.
5	IENWP	0	R/W	Wakeup Interrupt Enable Enables or disables WKP7 to WKP0 interrupt requests. 0: Disables $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$ interrupt requests 1: Enables $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$ interrupt requests
4, 3	—	—	W	Reserved The write value should always be 0.
2	IENEC2	0	R/W	IRQAEC Interrupt Enable Enables or disables IRQAEC interrupt requests. 0: Disables IRQAEC interrupt requests 1: Enables IRQAEC interrupt requests
1	IEN1	0	R/W	IRQ1 and IRQ0 Interrupt Enable
0	IEN0	0	R/W	Enables or disables IRQ1 and IRQ0 interrupt requests. 0: Disables $\overline{\text{IRQn}}$ interrupt requests 1: Enables $\overline{\text{IRQn}}$ interrupt requests (n = 1, 0)

3.2.3 Interrupt Enable Register 2 (IENR2)

IENR2 enables direct transition, A/D converter, and timer interrupts.

Bit	Bit Name	Initial Value	R/W	Description
7	IENDT	0	R/W	Direct Transition Interrupt enable Enables or disables direct transition interrupt requests. 0: Disables direct transition interrupt requests 1: Enables direct transition interrupt requests
6	IENAD	0	R/W	A/D Converter Interrupt enable Enables or disables A/D conversion end interrupt requests. 0: Disables A/D converter interrupt requests 1: Enables A/D converter interrupt requests
5, 4	—	—	W	Reserved The write value should always be 0.
3	IENTFH	0	R/W	Timer FH Interrupt Enable Enables or disables timer FH compare match or overflow interrupt requests. 0: Disables timer FH interrupt requests 1: Enables timer FH interrupt requests
2	IENTFL	0	R/W	Timer FL Interrupt Enable Enables or disables timer FL compare match or overflow interrupt requests. 0: Disables timer FL interrupt requests 1: Enables timer FL interrupt requests
1	—	—	W	Reserved The write value should always be 0.
0	IENEC	0	R/W	Asynchronous Event Counter Interrupt Enable Enables or disables asynchronous event counter interrupt requests. 0: Disables asynchronous event counter interrupt requests 1: Enables asynchronous event counter interrupt requests

For details on SCI3 interrupt control, refer to section 10.3.6, Serial Control Register 3 (SCR3).

3.2.4 Interrupt Request Register 1 (IRR1)

IRR1 is a status flag register for timer A, IRQAEC, IRQ1, and IRQ0 interrupt requests. The corresponding flag is set to 1 when an interrupt request occurs. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit	Bit Name	Initial Value	R/W	Description
7	IRRTA	0	R/W*	Timer A Interrupt Request Flag [Setting condition] When the timer A counter value overflows [Clearing condition] When IRRTA = 1, it is cleared by writing 0
6, 4, 3	—	—	W	Reserved The write value should always be 0.
5	—	1	—	Reserved This bit is always read as 1 and cannot be modified.
2	IRREC2	0	R/W*	IRQAEC Interrupt Request Flag [Setting condition] When pin IRQAEC is designated for interrupt input and the designated signal edge is detected [Clearing condition] When IRREC2 = 1, it is cleared by writing 0
1	IRRI1	0	R/W*	IRQ1 and IRQ0 Interrupt Request Flag
0	IRRI0	0	R/W*	[Setting condition] When pin $\overline{\text{IRQn}}$ is designated for interrupt input and the designated signal edge is detected (n = 1, 0) [Clearing condition] When IRRI1 and IRRI0 = 1, they are cleared by writing 0

Note: * Only 0 can be written for flag clearing.

3.2.5 Interrupt Request Register 2 (IRR2)

IRR2 is a status flag register for direct transition, A/D converter, timer FH, timer FL, and asynchronous event counter interrupt requests. The corresponding flag is set to 1 when an interrupt request occurs. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit	Bit Name	Initial Value	R/W	Description
7	IRRDT	0	R/W*	Direct Transition Interrupt Request Flag [Setting condition] When a direct transition is made by executing a SLEEP instruction while the DTON bit = 1 [Clearing condition] When IRRDT = 1, it is cleared by writing 0
6	IRRAD	0	R/W*	A/D Converter Interrupt Request Flag [Setting condition] When A/D conversion is completed and the ADSF bit is cleared to 0 [Clearing condition] When IRRAD = 1, it is cleared by writing 0
5, 4	—	—	W	Reserved The write value should always be 0.
3	IRRTFH	0	R/W*	Timer FH Interrupt Request Flag [Setting condition] When TCFH and OCRFH match in 8-bit timer mode, or when TCF (TCFL, TCFH) and OCRF (OCRFL, OCRFH) match in 16-bit timer mode [Clearing condition] When IRRTFH = 1, it is cleared by writing 0
2	IRRTFL	0	R/W*	Timer FL Interrupt Request Flag [Setting condition] When TCFL and OCRFL match in 8-bit timer mode [Clearing condition] When IRRTFL = 1, it is cleared by writing 0
1	—	—	W	Reserved The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
0	IRREC	0	R/W*	Asynchronous Event Counter Interrupt Request Flag [Setting condition] When ECH overflows in 16-bit counter mode, or ECH or ECL overflows in 8-bit counter mode [Clearing condition] When IRREC = 1, it is cleared by writing 0

Note: * Only 0 can be written for flag clearing.

3.2.6 Wakeup Interrupt Request Register (IWPR)

IWPR is a status flag register for $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$ interrupt requests. The flags are not cleared automatically when an interrupt is accepted. It is necessary to write 0 to clear each flag.

Bit	Bit Name	Initial Value	R/W	Description
7	IWPF7	0	R/W*	Wakeup Interrupt Request Flag 7 to 0
6	IWPF6	0	R/W*	[Setting condition]
5	IWPF5	0	R/W*	When pin $\overline{\text{WKPn}}$ is designated for wakeup input and the designated edge is detected
4	IWPF4	0	R/W*	
3	IWPF3	0	R/W*	(n = 7 to 0)
2	IWPF2	0	R/W*	[Clearing condition]
1	IWPF1	0	R/W*	When IWPFn= 1, it is cleared by writing 0
0	IWPF0	0	R/W*	

Note: * Only 0 can be written for flag clearing.

3.2.7 Wakeup Edge Select Register (WEGR)

WEGR specifies rising or falling edge sensing for pins $\overline{\text{WKPn}}$.

Bit	Bit Name	Initial Value	R/W	Description
7	WKEGS7	0	R/W	WKPn Edge Select 7 to 0
6	WKEGS6	0	R/W	Selects $\overline{\text{WKPn}}$ pin input sensing.
5	WKEGS5	0	R/W	0: $\overline{\text{WKPn}}$ pin falling edge is detected
4	WKEGS4	0	R/W	1: $\overline{\text{WKPn}}$ pin rising edge is detected
3	WKEGS3	0	R/W	(n = 7 to 0)
2	WKEGS2	0	R/W	
1	WKEGS1	0	R/W	
0	WKEGS0	0	R/W	

3.3 Reset Exception Handling

When the $\overline{\text{RES}}$ pin goes low, all processing halts and this LSI enters the reset. The internal state of the CPU and the registers of the on-chip peripheral modules are initialized by the reset. To ensure that this LSI is reset at power-on, hold the $\overline{\text{RES}}$ pin low until the clock pulse generator output stabilizes. To reset the chip during operation, hold the $\overline{\text{RES}}$ pin low for at least 10 system clock cycles. When the $\overline{\text{RES}}$ pin goes high after being held low for the necessary time, this LSI starts reset exception handling. The reset exception handling sequence is shown in figure 3.1. The reset exception handling sequence is as follows.

1. Set the I bit in the condition code register (CCR) to 1.
2. The CPU generates a reset exception handling vector address (from H'0000 to H'0001), the data in that address is sent to the program counter (PC) as the start address, and program execution starts from that address.

3.4 Interrupt Exception Handling

3.4.1 External Interrupts

There are external interrupts, WKP7 to WKP0, IRQ1, IRQ0, and IRQAEC.

(1) WKP7 to WKP0 Interrupts

WKP7 to WKP0 interrupts are requested by input signals to pins $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$. These interrupts have the same vector addresses, and are detected individually by either rising edge sensing or falling edge sensing, depending on the settings of bits WKEGS7 to WKEGS0 in WEGR.

When pins $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$ are designated for interrupt input in PMR5 and the designated signal edge is input, the corresponding bit in IWPR is set to 1, requesting the CPU of an interrupt. These interrupts can be masked by setting bit IENWP in IENR1.

(2) IRQ1 and IRQ0 Interrupts

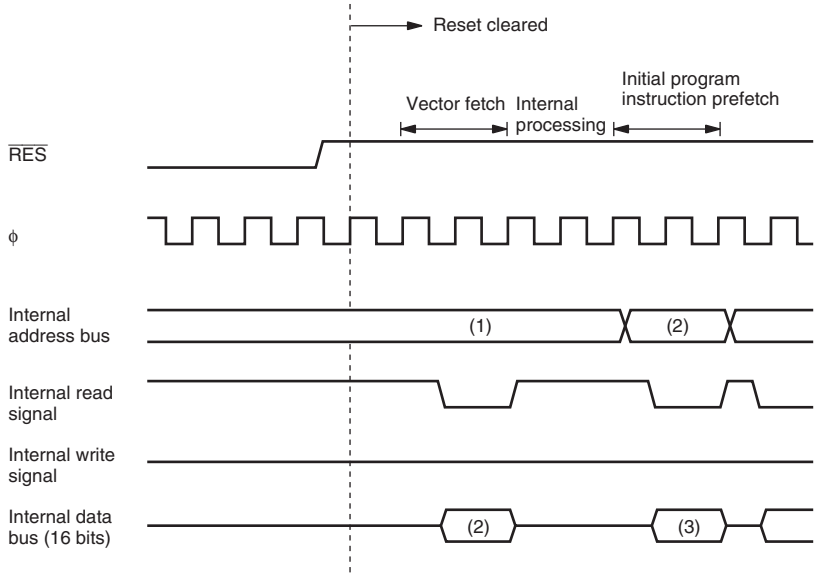
IRQ1 and IRQ0 interrupts are requested by input signals to pins $\overline{\text{IRQ1}}$ and $\overline{\text{IRQ0}}$. These interrupts are given different vector addresses, and are detected individually by either rising edge sensing or falling edge sensing, depending on the settings of bits IEG1 and IEG0 in IEGR.

When pins $\overline{\text{IRQ1}}$ and $\overline{\text{IRQ0}}$ are designated for interrupt input by PMRB and PMR2 and the designated signal edge is input, the corresponding bit in IRR1 is set to 1, requesting the CPU of an interrupt. These interrupts can be masked by setting bits IEN1 and IEN0 in IENR1.

(3) IRQAEC Interrupt

The IRQAEC interrupt is requested by an input signal to pin IRQAEC. This interrupt is detected by either rising edge sensing or falling edge sensing, depending on the settings of bits AIEGS1 and AIEGS0 in AEGSR.

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



(1) Reset exception handling vector address (H'0000)
(2) Program start address
(3) Initial program instruction

Figure 3.1 Reset Sequence

3.4.2 Internal Interrupts

Each on-chip peripheral module has a flag to show the interrupt request status and the enable bit to enable or disable the interrupt. For direct transition interrupt requests generated by execution of a SLEEP instruction, this function is included in IRR1 and IRR2.

When an on-chip peripheral module requests an interrupt, the corresponding interrupt request status flag is set to 1, requesting the CPU of an interrupt. When this interrupt is accepted, the I bit is set to 1 in CCR. These interrupts can be masked by writing 0 to clear the corresponding enable bit.

3.4.3 Interrupt Handling Sequence

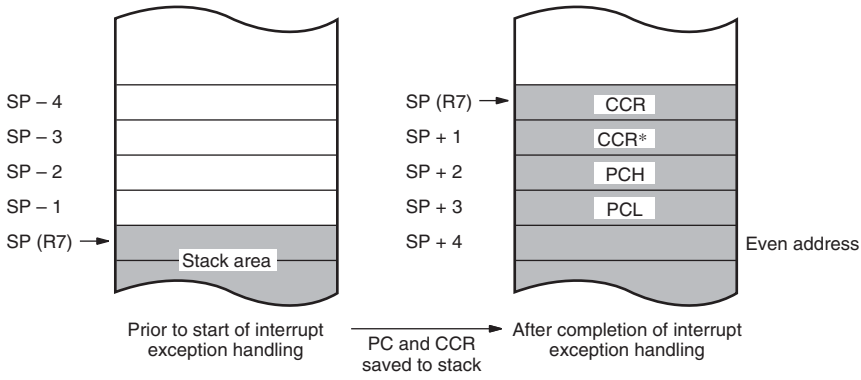
Interrupts are controlled by an interrupt controller.

Interrupt operation is described as follows.

1. If an interrupt occurs while the interrupt enable bit is set to 1, an interrupt request signal is sent to the interrupt controller.
2. When multiple interrupt requests are generated, the interrupt controller requests to the CPU for the interrupt handling with the highest priority at that time according to table 3.1. Other interrupt requests are held pending.
3. Interrupt requests are accepted, if the I bit is cleared to 0 in CCR; if the I bit is set to 1, the interrupt request is held pending.
4. If the CPU accepts the interrupt after processing of the current instruction is completed, interrupt exception handling will begin. First, both PC and CCR are pushed onto the stack. The state of the stack at this time is shown in figure 3.2. The PC value pushed onto the stack is the address of the first instruction to be executed upon return from interrupt handling.
5. Then, the I bit in CCR is set to 1, masking further interrupts. Upon return from interrupt handling, the values of I bit and other bits in CCR will be restored and returned to the values prior to the start of interrupt exception handling.
6. Next, the CPU generates the vector address corresponding to the accepted interrupt, and transfers the address to PC as a start address of the interrupt handling-routine. Then a program starts executing from the address indicated in PC.

Figure 3.3 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.

Notes: 1. When disabling interrupts by clearing bits in the interrupt enable register, or when clearing bits in the 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.



[Legend]
PCH: Upper 8 bits of program counter (PC)
PCL: Lower 8 bits of program counter (PC)
CCR: Condition code register
SP: Stack pointer

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

Figure 3.2 Stack Status after Exception Handling

3.4.4 Interrupt Response Time

Table 3.2 shows the number of wait states after an interrupt request flag is set until the first instruction of the interrupt handling-routine is executed.

Table 3.2 Interrupt Wait States

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

Note: * Not including EEPMOV instruction.

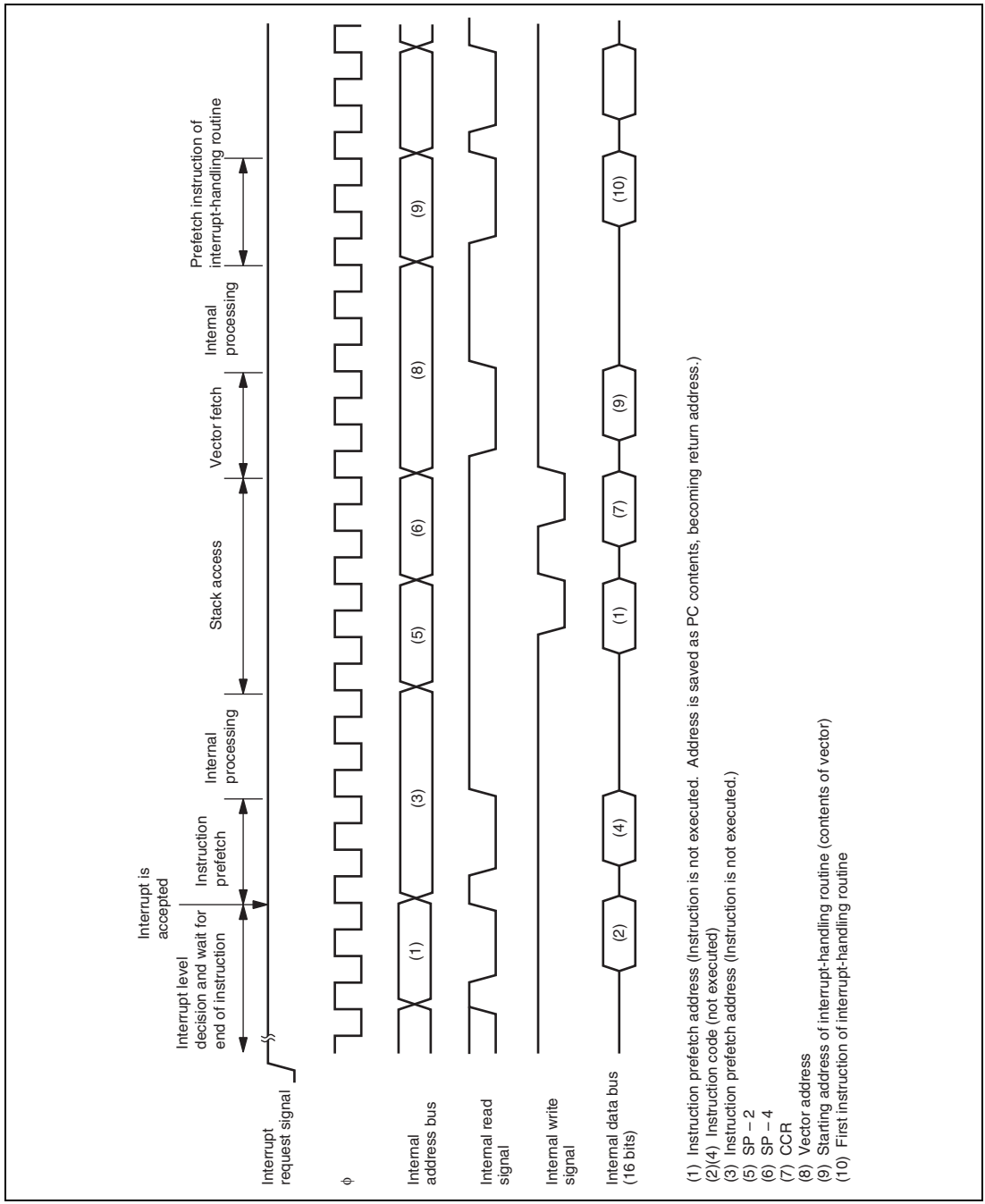


Figure 3.3 Interrupt Sequence

3.5 Usage Notes

3.5.1 Interrupts after Reset

If an interrupt is accepted after a reset and before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: MOV.W #xx: 16, SP).

3.5.2 Notes on Stack Area Use

When word data is accessed, 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.

3.5.3 Interrupt Request Flag Clearing Method

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

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

- BCR #1,@IRR1:8
- MOV.B R1L,@IRR1:8 (Set B′11111101 to R1L in advance)

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

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

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

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

3.5.4 Notes on Rewriting Port Mode Registers

When a port mode register is rewritten to switch the functions of external interrupt pins, IRQAEC, $\overline{\text{IRQ1}}$, $\overline{\text{IRQ0}}$, and $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$, the interrupt request flag may be set to 1.

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.

Table 3.3 lists the interrupt request flags which are set to 1 and the conditions.

Table 3.3 Conditions under which Interrupt Request Flag is Set to 1

Interrupt Request Flags Set to 1		Conditions
IRR1	IRREC2	When the edge designated by AIEGS1 and AIEGS0 in AEGSR is input while IENEC2 in IENRI is set to 1.
	IRRI1	When IRQ1 bit in PMRB is changed from 0 to 1 while pin $\overline{\text{IRQ1}}$ is low and IEG1 bit in IEGR = 0.
		When IRQ1 bit in PMRB is changed from 1 to 0 while pin $\overline{\text{IRQ1}}$ is low and IEG1 bit in IEGR = 1.
	IRRI0	When IRQ0 bit in PMR2 is changed from 0 to 1 while pin $\overline{\text{IRQ0}}$ is low and IEG0 bit in IEGR = 0.
		When IRQ0 bit in PMR2 is changed from 1 to 0 while pin $\overline{\text{IRQ0}}$ is low and IEG0 bit in IEGR = 1.

Interrupt Request Flags Set to 1		Conditions
IWPR	IWPF7	When PMR5 bit WKP7 is changed from 0 to 1 while pin $\overline{\text{WKP7}}$ is low and WEGR bit WKEGS7 = 0.
		When PMR5 bit WKP7 is changed from 1 to 0 while pin $\overline{\text{WKP7}}$ is low and WEGR bit WKEGS7 = 1.
	IWPF6	When PMR5 bit WKP6 is changed from 0 to 1 while pin $\overline{\text{WKP6}}$ is low and WEGR bit WKEGS6 = 0.
		When PMR5 bit WKP6 is changed from 1 to 0 while pin $\overline{\text{WKP6}}$ is low and WEGR bit WKEGS6 = 1.
	IWPF5	When PMR5 bit WKP5 is changed from 0 to 1 while pin $\overline{\text{WKP5}}$ is low and WEGR bit WKEGS5 = 0.
		When PMR5 bit WKP5 is changed from 1 to 0 while pin $\overline{\text{WKP5}}$ is low and WEGR bit WKEGS5 = 1.
	IWPF4	When PMR5 bit WKP4 is changed from 0 to 1 while pin $\overline{\text{WKP4}}$ is low and WEGR bit WKEGS4 = 0.
		When PMR5 bit WKP4 is changed from 1 to 0 while pin $\overline{\text{WKP4}}$ is low and WEGR bit WKEGS4 = 1.
	IWPF3	When PMR5 bit WKP3 is changed from 0 to 1 while pin $\overline{\text{WKP3}}$ is low and WEGR bit WKEGS3 = 0.
		When PMR5 bit WKP3 is changed from 1 to 0 while pin $\overline{\text{WKP3}}$ is low and WEGR bit WKEGS3 = 1.
	IWPF2	When PMR5 bit WKP2 is changed from 0 to 1 while pin $\overline{\text{WKP2}}$ is low and WEGR bit WKEGS2 = 0.
		When PMR5 bit WKP2 is changed from 1 to 0 while pin $\overline{\text{WKP2}}$ is low and WEGR bit WKEGS2 = 1.
	IWPF1	When PMR5 bit WKP1 is changed from 0 to 1 while pin $\overline{\text{WKP1}}$ is low and WEGR bit WKEGS1 = 0.
		When PMR5 bit WKP1 is changed from 1 to 0 while pin $\overline{\text{WKP1}}$ is low and WEGR bit WKEGS1 = 1.
	IWPF0	When PMR5 bit WKP0 is changed from 0 to 1 while pin $\overline{\text{WKP0}}$ is low and WEGR bit WKEGS0 = 0.
		When PMR5 bit WKP0 is changed from 1 to 0 while pin $\overline{\text{WKP0}}$ is low and WEGR bit WKEGS0 = 1.

Figure 3.4 shows a port mode register setting and interrupt request flag clearing procedure.

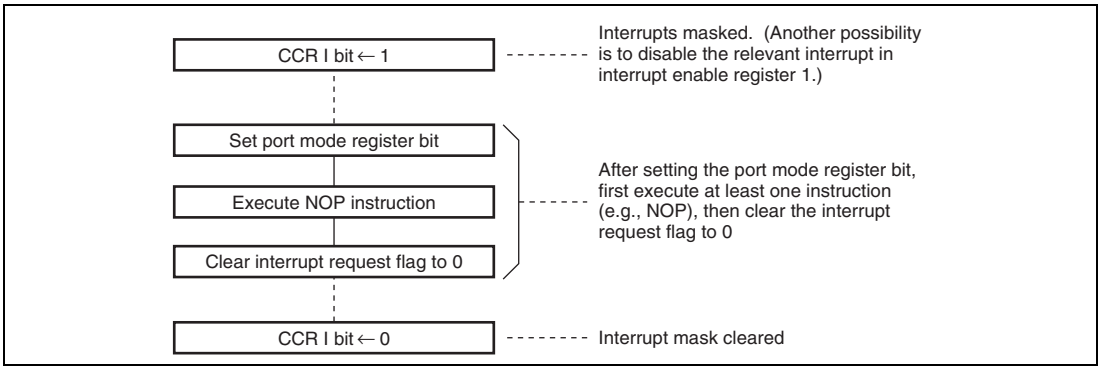


Figure 3.4 Port Mode Register Setting and Interrupt Request Flag Clearing Procedure

Section 4 Clock Pulse Generators

4.1 Features

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 and a subclock divider.

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

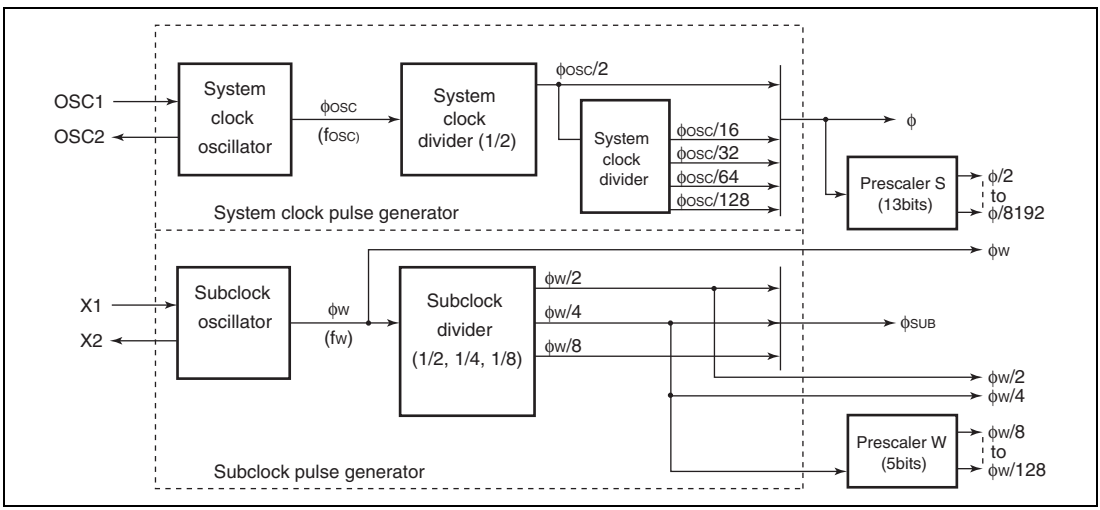
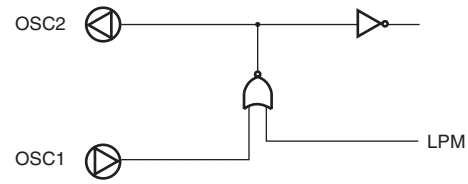


Figure 4.1 Block Diagram of Clock Pulse Generators

4.2 System Clock Generator

Clock pulses can be supplied to the system clock divider either by connecting a crystal or ceramic resonator, or by providing external clock input. Figure 4.2 shows a block diagram of the system clock generator.

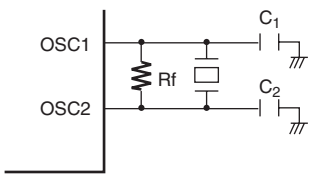


Note: LPM: Power-down mode (standby mode, subactive mode, subsleep mode, watch mode)

Figure 4.2 Block Diagram of System Clock Generator

4.2.1 Connecting Crystal Resonator

Figure 4.3 shows a typical method of connecting a crystal oscillator to the H8/38704, H8/38702S Group. Figure 4.4 shows the equivalent circuit of a crystal resonator. A resonator having the characteristics given in table 4.1 should be used.



Frequency	Manufacturer	Prodoct Name	C1, C2 Recommendation Value
4.0 MHz	KYOCERA KINSEKI Corp.	HC-49/U-S	12 pF ±20%

Rf = 1 MΩ ±20%

Note: Consult with the crystal resonator manufacturer to determine the circuit constants.

Figure 4.3 Typical Connection to Crystal Resonator (H8/38704, H8/38702S Group)

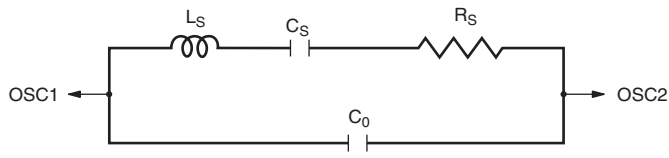


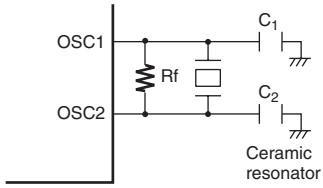
Figure 4.4 Equivalent Circuit of Crystal Resonator

Table 4.1 Crystal Resonator Parameters

Frequency (MHz)	4.10	4.193
R_S (max)	150 Ω	
C_0 (max)	1.4 pF	

4.2.2 Connecting Ceramic Resonator

Figure 4.5 shows a typical method of connecting a ceramic oscillator to the H8/38704, H8/38702S Group.



Frequency	Manufacturer	Product Name	C1, C2 Recommendation Value
2.0 MHz	Murata Manufacturing Co., Ltd.	CSTCC2M00G53-B0	15 pF ±20%
		CSTCC2M00G56-B0	47 pF ±20%
4.19 MHz		CSTLS4MAG53-B0	15 pF ±0.5%
		CSTLS4MAG56-B0	47 pF ±0.5%
10.0 MHz		CSTLS10M0G53-B0	15 pF ±20%
		CSTLS10M0G56-B0	47 pF ±20%

Rf = 1 MΩ ±20%

Notes: Consult with the ceramic resonator manufacturer to determine the circuit constants.

Figure 4.5 Typical Connection to Ceramic Resonator (H8/38704, H8/38702S Group)

4.2.3 External Clock Input Method

Connect an external clock signal to pin OSC1, and leave pin OSC2 open. Figure 4.6 shows a typical connection. The duty cycle of the external clock signal must be 45 to 55%.

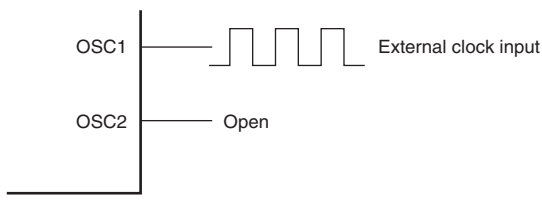
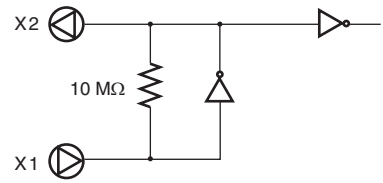


Figure 4.6 Example of External Clock Input

4.3 Subclock Generator

Figure 4.7 shows a block diagram of the subclock generator.

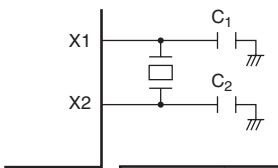


Note : Resistance is a reference value.

Figure 4.7 Block Diagram of Subclock Generator

4.3.1 Connecting 32.768-kHz/38.4-kHz Crystal Resonator

Clock pulses can be supplied to the subclock divider by connecting a 32.768-kHz or 38.4-kHz crystal resonator, as shown in figure 4.8. Figure 4.9 shows the equivalent circuit of the 32.768-kHz or 38.4-kHz crystal resonator.

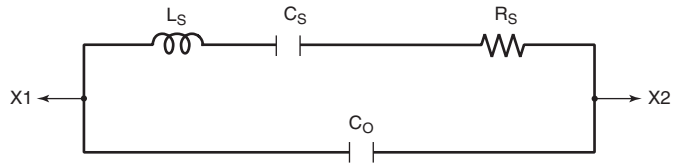


$C_1 = C_2 = 7$ pF (typ.)

Frequency	Manufacturer	Product Name	Equivalent Series Resistance
38.4 kHz	EPSON TOYOCOM Corp.	C-4-TYPE	30 kΩ (max.)
32.768 kHz	EPSON TOYOCOM Corp.	C-001R	35 kΩ (max.)

Notes: 1. When using a resonator other than the above, ensure optimal conditions by conducting sufficient evaluation of consistency in cooperation with the manufacturer of the resonator. Even if the above resonators or products equivalent to them are implemented, their oscillation characteristics are affected by the board design. Be sure to use the actual board to evaluate consistency as a system.
2. The consistency as a system has to be verified not only in a reset state (i.e., the $\overline{\text{RES}}$ pin is driven low) but also in a state where a reset state has been exited (i.e., the low-level $\overline{\text{RES}}$ signal has been driven high).

Figure 4.8 Typical Connection to 32.768-kHz/38.4-kHz Crystal Resonator



C_O = 0.9 pF (typ.)
R_S = 14 kΩ (typ.)
f_W = 32.768 kHz/38.4 kHz

Note: Constants are reference values.

Figure 4.9 Equivalent Circuit of 32.768-kHz/38.4-kHz Crystal Resonator

4.3.2 Pin Connection when Not Using Subclock

When the subclock is not used, connect pin X1 to GND and leave pin X2 open, as shown in figure 4.10.

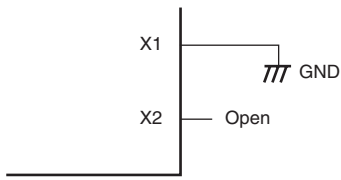


Figure 4.10 Pin Connection when Not Using Subclock

4.3.3 External Clock Input

Connect the external clock to pin X1 and leave pin X2 open, as shown in figure 4.11.

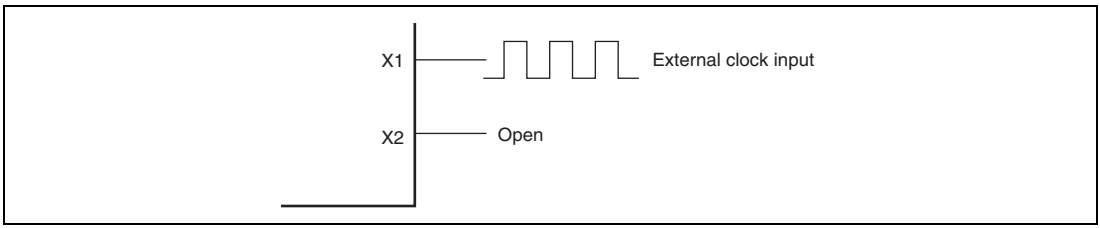


Figure 4.11 Pin Connection when Inputting External Clock

Frequency	Subclock (ϕ_w)
Duty	45% to 55%

4.4 Prescalers

4.4.1 Prescaler S

Prescaler S is a 13-bit counter using the system clock (ϕ) as its input clock. It is incremented once per clock period. Prescaler S is initialized to H'0000 by a reset, and starts counting on exit from the reset state. In standby mode, watch mode, subactive mode, and subsleep mode, the system clock pulse generator stops. Prescaler S also stops and is initialized to H'0000. The CPU cannot read or write prescaler S. The output from prescaler S is shared by the on-chip peripheral modules. The division ratio can be set separately for each on-chip peripheral function. In active (medium-speed) mode and sleep mode, the clock input to prescaler S is determined by the division ratio designated by the MA1 and MA0 bits in SYSCR2.

4.4.2 Prescaler W

Prescaler W is a 5-bit counter using a 32.768 kHz or 38.4 kHz signal divided by 4 (ϕ_w/4) as its input clock. The divided output is used for clock time base operation of timer A. 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. Prescaler W can be reset by setting 1s in bits TMA3 and TMA2 in TMA.

4.5 Usage Notes

4.5.1 Note on Resonators

Resonator characteristics are closely related to board design and should be carefully evaluated by the user, referring to the examples shown in this section. Resonator circuit constants will differ depending on the resonator element, stray capacitance in its interconnecting circuit, and other factors. Suitable constants should be determined in consultation with the resonator manufacturer. Design the circuit so that the resonator never receives voltages exceeding its maximum rating.

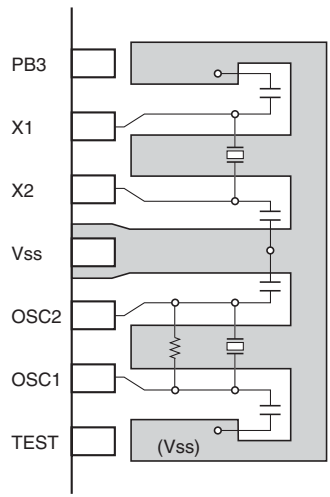


Figure 4.12 Example of Crystal and Ceramic Resonator Arrangement

Figure 4.13 (1) shows an example of the measurement circuit for the negative resistor which is recommended by the resonator manufacturer. Note that if the negative resistor in this circuit does not reach the level which is recommended by the resonator manufacturer, the main oscillator may be hard to start oscillation.

If the negative resistor does not reach the level which is recommended by the resonator manufacturer and oscillation is not started, changes as shown in figure 4.13 (2) to (4) should be made. The proposed change and capacitor size to be applied should be determined according to the evaluation result of the negative resistor and frequency deviation, etc.

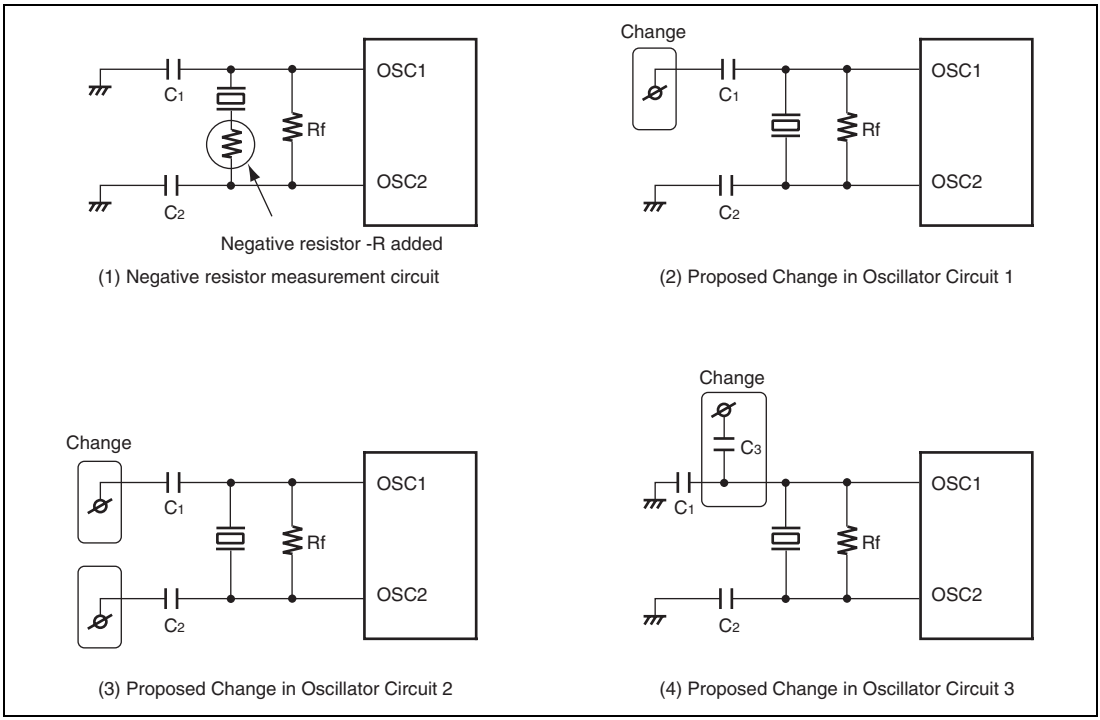


Figure 4.13 Negative Resistor Measurement and Proposed Changes in Circuit

4.5.2 Notes on Board Design

When using a crystal resonator (ceramic resonator), place the resonator and its load capacitors as close as possible to the OSC1 and OSC2 pins. Other signal lines should be routed away from the resonator circuit to prevent induction from interfering with correct oscillation (see figure 4.14).

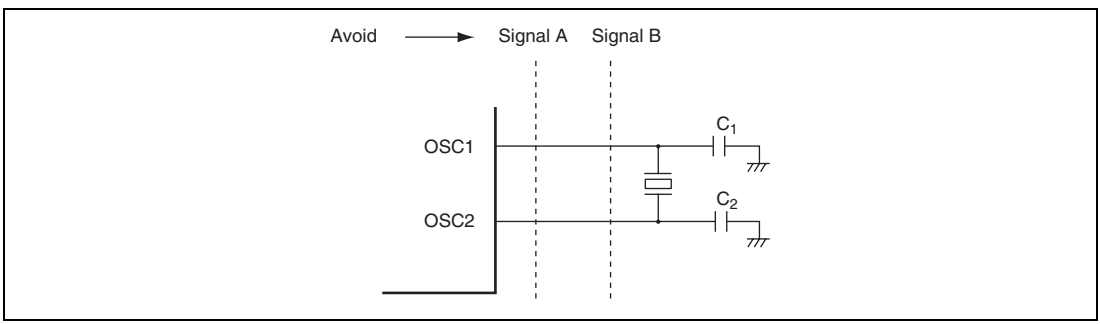


Figure 4.14 Example of Incorrect Board Design

4.5.3 Definition of Oscillation Stabilization Standby Time

Figure 4.15 shows the oscillation waveform (OSC2), system clock (ϕ), and microcomputer operating mode when a transition is made from standby mode, watch mode, or subactive mode, to active (high-speed/medium-speed) mode, with a resonator connected to the system clock oscillator.

As shown in figure 4.15, as the system clock oscillator is halted in standby mode, watch mode, and subactive mode, when a transition is made to active (high-speed/medium-speed) mode, the sum of the following two times (oscillation stabilization time and standby time) is required.

1. Oscillation start time

The time from the point at which the oscillation waveform of the system clock oscillator starts to change when an interrupt is generated, until generation of the system clock is started.

2. Standby time

The time required for the CPU and peripheral functions to begin operating after generation of the system clock has been started.

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

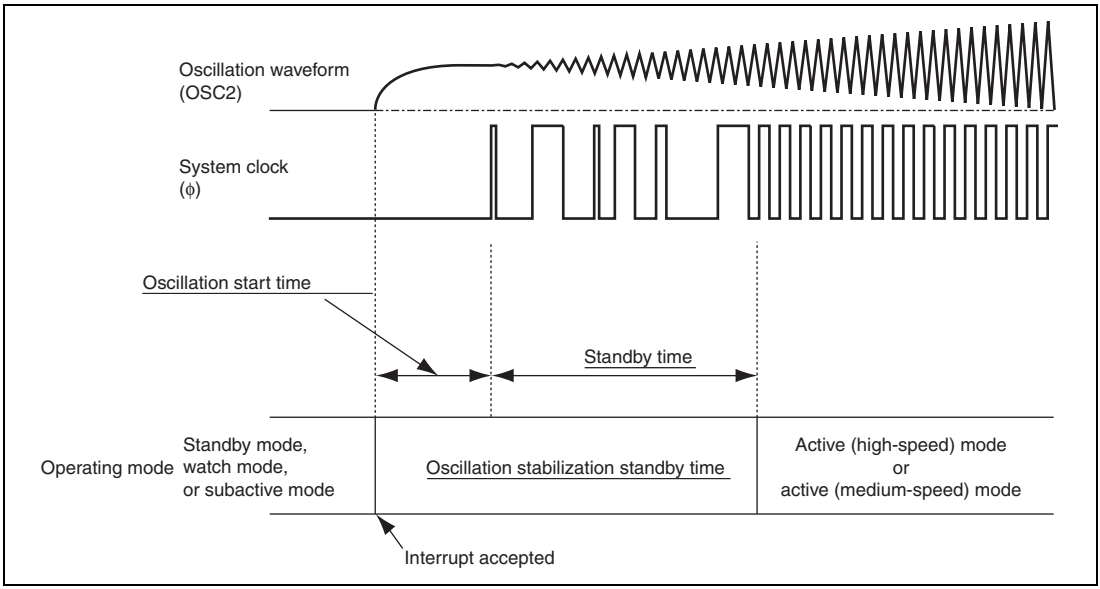


Figure 4.15 Oscillation Stabilization Standby Time

The required oscillation stabilization time is identical with the oscillation stabilization time (t_{rc}) when power as specified by the AC characteristics is supplied. The setting must be such that the time specified by the STS2 to STS0 bits in SYSCR is not less than t_{rc}. Consequently, when a resonator is connected as the system clock oscillator and a transition is made from the standby, watch, or subactive mode to the active (high- or medium-speed) mode, be sure to sufficiently test behavior on the actual circuit. Waiting time must be enough for the amplitudes of OSC1 and OSC2 to become sufficiently large.

Since the oscillation start time varies with the constant of the actual circuit and stray capacitance, determine the oscillation stabilization waiting time in close cooperation with the manufacturer of the resonator.

4.5.4 Notes on Use of Resonator

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

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

For example, if erroneous operation occurs with a standby time setting of 1,024 states, check the operation with a standby time setting of 2,048 states or more.

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

Section 5 Power-Down Modes

This LSI has eight modes of operation after a reset. These include a normal active (high-speed) mode and seven power-down modes, in which power consumption is significantly reduced. The module standby function reduces power consumption by selectively halting on-chip module functions.

- Active (medium-speed) mode

  The CPU and all on-chip peripheral modules are operable on the system clock. The system clock frequency can be selected from ϕosc/16, ϕosc/32, ϕosc/64, and ϕosc/128.
- Subactive mode

  The CPU and all on-chip peripheral modules are operable on the subclock. The subclock frequency can be selected from ϕw/2, ϕw/4, and ϕw/8.
- Sleep (high-speed) mode

  The CPU halts. On-chip peripheral modules are operable on the system clock.
- Sleep (medium-speed) mode

  The CPU halts. On-chip peripheral modules are operable on the system clock. The system clock frequency can be selected from ϕosc/16, ϕosc/32, ϕosc/64, and ϕosc/128.
- Subsleep mode

  The CPU halts. The timer A, timer F, SCI3, and AEC are operable on the subclock. The subclock frequency can be selected from ϕw/2, ϕw/4, and ϕw/8.
- Watch mode

  The CPU halts. Timer A's timekeeping function, timer F, and AEC are operable on the subclock.
- Standby mode

  The CPU and all on-chip peripheral modules halt.
- Module standby function

  Independent of the above modes, power consumption can be reduced by halting on-chip peripheral modules that are not used in module units.

Note: In this manual, active (high-speed) mode and active (medium-speed) mode are collectively called active mode.

5.1 Register Descriptions

The registers related to power-down modes are as follows.

- System control register 1 (SYSCR1)
- System control register 2 (SYSCR2)
- Clock halt registers 1 and 2 (CKSTPR1 and CKSTPR2)

5.1.1 System Control Register 1 (SYSCR1)

SYSCR1 controls the power-down modes, as well as SYSCR2.

Bit	Bit Name	Initial Value	R/W	Description
7	SSBY	0	R/W	Software Standby Selects the mode to transit after the execution of the SLEEP instruction. 0: A transition is made to sleep mode or subsleep mode. 1: A transition is made to standby mode or watch mode. For details, see table 5.2.
6	STS2	0	R/W	Standby Timer Select 2 to 0 Designate the time the CPU and peripheral modules wait for stable clock operation after exiting from standby mode, subactive mode, subsleep mode, or watch mode to active mode or sleep mode due to an interrupt. The designation should be made according to the operating frequency so that the waiting time is at least equal to the oscillation stabilization time. The relationship between the specified value and the number of wait states is shown in table 5.1. When an external clock is to be used, the minimum value (STS2 = 1, STS1 = 0, STS0 = 1) is recommended. If the setting other than the recommended value is made, operation may start before the end of the waiting time.
5	STS1	0	R/W	
4	STS0	0	R/W	
3	LSON	0	R/W	Selects the system clock (ϕ) or subclock (ϕ_{SUB}) as the CPU operating clock when watch mode is cleared. 0: The CPU operates on the system clock (ϕ) 1: The CPU operates on the subclock (ϕ_{SUB})

Bit	Bit Name	Initial Value	R/W	Description
2	—	1	—	Reserved This bit is always read as 1 and cannot be modified.
1	MA1	1	R/W	Active Mode Clock Select 1 and 0
0	MA0	1	R/W	Select ϕ_{osc}/16, ϕ_{osc}/32, ϕ_{osc}/64, or ϕ_{osc}/128 as the operating clock in active (medium-speed) mode and sleep (medium-speed) mode. The MA1 and MA0 bits should be written to in active (high-speed) mode or subactive mode. 00: ϕ_{osc}/16 01: ϕ_{osc}/32 10: ϕ_{osc}/64 11: ϕ_{osc}/128

Table 5.1 Operating Frequency and Waiting Time

Bit				Operating Frequency	
STS2	STS1	STS0	Waiting Time	5 MHz	2 MHz
0	0	0	8,192 states	1.638	4.1
		1	16,384 states	3.277	8.2
	1	0	1,024 states	0.205	0.512
		1	2,048 states	0.410	1.024
1	0	0	4,096 states	0.819	2.048
		1	2 states (external clock input)	0.0004	0.001
	1	0	8 states	0.002	0.004
		1	16 states	0.003	0.008

Note: The time unit is ms.

If external clock input is used, STS2 to STS0 should be set to the external clock input mode before the mode transition is executed. In addition, STS2 to STS0 should not be set to the external clock input mode if external clock input is not used.

5.1.2 System Control Register 2 (SYSCR2)

SYSCR2 controls the power-down modes, as well as SYSCR1.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
4	NESEL	1	R/W	Noise Elimination Sampling Frequency Select Selects the frequency at which the watch clock signal (ϕ_w) generated by the subclock pulse generator is sampled, in relation to the oscillator clock (ϕ_{osc}) generated by the system clock pulse generator. When ϕ_{osc} = 2 to 16 MHz, clear this bit to 0. 0: Sampling rate is ϕ_{osc}/16. 1: Sampling rate is ϕ_{osc}/4.
3	DTON	0	R/W	Direct Transfer on Flag Selects the mode to which the transition is made after the SLEEP instruction is executed with bits SSBY and LSON in SYSCR1, bit MSON in SYSCR2, and bit TMA3 in TMA. For details, see table 5.2.
2	MSON	0	R/W	Medium Speed on Flag After standby, watch, or sleep mode is cleared, this bit selects active (high-speed) or active (medium-speed) mode. 0: Operation in active (high-speed) mode 1: Operation in active (medium-speed) mode
1	SA1	0	R/W	Subactive Mode Clock Select 1 and 0
0	SA0	0	R/W	Select the operating clock frequency in subactive and subsleep modes. The operating clock frequency changes to the set frequency after the SLEEP instruction is executed. 00: ϕ_w/8 01: ϕ_w/4 1x: ϕ_w/2

[Legend] x: Don't care.

5.1.3 Clock Halt Registers 1 and 2 (CKSTPR1 and CKSTPR2)

CKSTPR1 and CKSTPR2 allow the on-chip peripheral modules to enter a standby state in module units.

- CKSTPR1

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved
5	S32CKSTP	1	R/W	SCI Module Standby SCI3 enters standby mode when this bit is cleared to 0.*1
4	ADCKSTP	1	R/W	A/D Converter Module Standby A/D converter enters standby mode when this bit is cleared to 0.
3	—	1	—	Reserved
2	TFCKSTP	1	R/W	Timer F Module Standby Timer F enters standby mode when this bit is cleared to 0.
1	—	1	—	Reserved
0	TACKSTP	1	R/W	Timer A Module Standby*2 Timer A enters standby mode when this bit is cleared to 0.

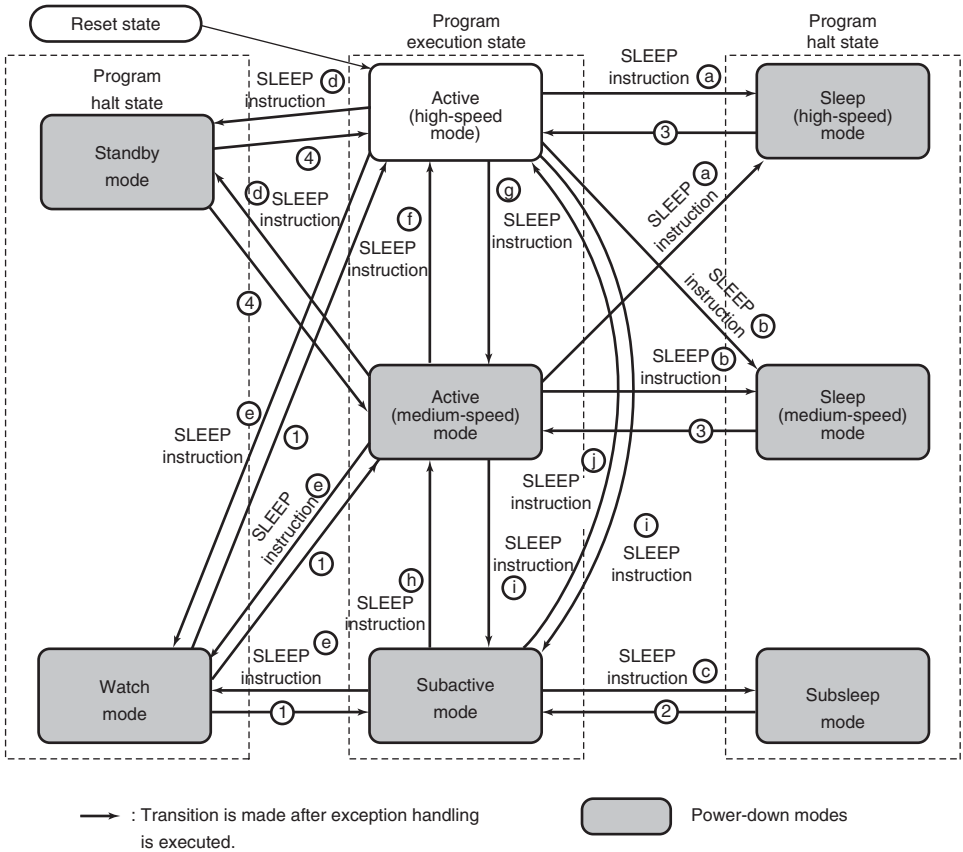
- CKSTPR2

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R/W	Reserved
6, 5	—	All 1	—	Reserved
4	PW2CKSTP	1	R/W*3	PWM2 Module Standby PWM2 enters standby mode when this bit is cleared to 0.
3	AECKSTP	1	R/W	Asynchronous Event Counter Module Standby Asynchronous event counter enters standby mode when this bit is cleared to 0
2	WDCKSTP	1	R/W*4	Watchdog Timer Module Standby Watchdog timer enters standby mode when this bit is cleared to 0
1	PW1CKSTP	1	R/W	PWM1 Module Standby PWM1 enters standby mode when this bit is cleared to 0
0	—	1	—	Reserved

Notes: 1. When the SCI module standby is set, all registers in the SCI3 enter the reset state.
2. When the timer A module standby is set, the TMA3 bit in TMA cannot be rewritten. When the TMA3 bit is rewritten, the TACKSTP bit in CKSTPR1 should be set to 1 in advance.

5.2 Mode Transitions and States of LSI

Figure 5.1 shows the possible transitions among these operating modes. A transition is made from the program execution state to the program halt state of the program by executing a SLEEP instruction. Interrupts allow for returning from the program halt state to the program execution state of the program. A direct transition between active mode and subactive mode, which are both program execution states, can be made without halting the program. The operating frequency can also be changed in the same modes by making a transition directly from active mode to active mode, and from subactive mode to subactive mode. $\overline{\text{RES}}$ input enables transitions from a mode to the reset state. Table 5.2 shows the transition conditions of each mode after the SLEEP instruction is executed and a mode to return by an interrupt. Table 5.3 shows the internal states of the LSI in each mode.



Mode Transition Conditions (1)

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

[Legend] * Don't care

Mode Transition Conditions (2)

	Interrupt Sources
①	Timer A, Timer F, IRQ0 interrupt, WKP7 to WKP0 interrupts
②	Timer A, Timer F, SCI3 interrupt, IRQ1 and IRQ0, IRQAEC interrupts, WKP7 o WKP0 interrupts, AEC
③	All interrupts
④	IRQ1 or IRQ0, WKP7 to WKP0 interrupts

Note: A transition between different modes cannot be made to occur simply because an interrupt request is generated. Make sure that interrupts are enabled.

Figure 5.1 Mode Transition Diagram

Table 5.2 Transition Mode after SLEEP Instruction Execution and Interrupt Handling

State Before Transition	LSON	MSON	SSBY	TMA3	DTON	Transition Mode after SLEEP Instruction Execution	Transition Mode due to Interrupt	Symbol in Figure 5.1
Active (high-speed) mode	0	0	0	x	0	Sleep (high-speed) mode	Active (high-speed) mode	a
	0	1	0	x	0	Sleep (medium-speed) mode	Active (medium-speed) mode	b
	0	0	1	0	0	Standby mode	Active (high-speed) mode	d
	0	1	1	0	0	Standby mode	Active (medium-speed) mode	d
	0	0	1	1	0	Watch mode	Active (high-speed) mode	e
	0	1	1	1	0	Watch mode	Active (medium-speed) mode	e
	1	x	1	1	0	Watch mode	Subactive mode	e
	0	0	0	x	1	Active (high-speed) mode (direct transition)	—	—
	0	1	0	x	1	Active (medium-speed) mode (direct transition)	—	g
	1	x	1	1	1	Subactive mode (direct transition)	—	i

State Before Transition	LSON	MSON	SSBY	TMA3	DTON	Transition Mode after SLEEP Instruction Execution	Transition Mode due to Interrupt	Symbol in Figure 6.1
Active (medium-speed) mode	0	0	0	x	0	Sleep (high-speed) mode	Active (high-speed) mode	a
	0	1	0	x	0	Sleep (medium-speed) mode	Active (medium-speed) mode	b
	0	0	1	0	0	Standby mode	Active (high-speed) mode	d
	0	1	1	0	0	Standby mode	Active (medium-speed) mode	d
	0	0	1	1	0	Watch mode	Active (high-speed) mode	e
	0	1	1	1	0	Watch mode	Active (medium-speed) mode	e
	1	1	1	1	0	Watch mode	Subactive mode	e
	0	0	0	x	1	Active (high-speed) mode (direct transition)	—	f
	0	1	0	x	1	Active (medium-speed) mode (direct transition)	—	—
	1	x	1	1	1	Subactive mode (direct transition)	—	i

State Before Transition	LSON	MSON	SSBY	TMA3	DTON	Transition Mode after SLEEP Instruction Execution	Transition Mode due to Interrupt	Symbol in Figure 6.1
Subactive mode	1	x	0	1	0	Subsleep mode	Subactive mode	c
	0	0	1	1	0	Watch mode	Active (high-speed) mode	e
	0	1	1	1	0	Watch mode	Active (medium-speed) mode	e
	1	x	1	1	0	Watch mode	Subactive mode	e
	0	0	1	1	1	Active (high-speed) mode (direct transition)	—	j
	0	1	1	1	1	Active (medium-speed) mode (direct transition)	—	h
	1	x	1	1	1	Subactive mode (direct transition)	—	—

[Legend]

x: Don't care

Table 5.3 Internal State in Each Operating Mode

Function		Active Mode: High-speed	Active Mode: Medium-speed	Sleep Mode: High-speed	Sleep Mode: Medium-speed	Watch Mode	Subactive Mode	Subsleep Mode	Stand-by Mode
System clock oscillator		Functioning	Functioning	Functioning	Functioning	Halted	Halted	Halted	Halted
Subclock oscillator		Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning
CPU	Instructions	Functioning	Functioning	Halted	Halted	Halted	Functioning	Halted	Halted
	RAM			Retained	Retained	Retained		Retained	Retained
	Registers								
	I/O								Retained[*1]
External interrupts	IRQ0	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning
	IRQ1					Retained[*4]			
	IRQAEC								Retained[*4]
	WKP7 to WKP0					Functioning			Functioning
Peripheral modules	Timer A	Functioning	Functioning	Functioning	Functioning	Functioning[*3]	Functioning[*3]	Functioning[*3]	Retained
	Asynchronous counter					Functioning[*5]	Functioning	Functioning	Functioning[*5]
	Timer F					Functioning/ retained[*6]	Functioning/ retained[*6]	Functioning/ retained[*6*7]	Retained
	WDT					Functioning/ retained[*8]	Functioning/ retained[*7]	Functioning/ retained[*8]	Functioning/ retained[*8]
	SCI3	Functioning	Functioning	Functioning	Functioning	Reset	Functioning/| retained[*2]	Functioning/ retained[*2]	Reset
	PWM	Functioning	Functioning	Functioning	Functioning	Retained	Retained	Retained	Retained
Peripheral modules	A/D converter	Functioning	Functioning	Functioning	Functioning	Retained	Retained	Retained	Retained

Notes: 1. Register contents are retained. Output is the high-impedance state.
2. Functioning if $\phi_W/2$ is selected as an internal clock, or halted and retained otherwise.
3. Functioning if the timekeeping time-base function is selected.
4. An external interrupt request is ignored. Contents of the interrupt request register are not affected.
5. The counter can be incremented. An interrupt cannot occur.
6. Functioning if ϕw/4 is selected as an internal clock. Halted and retained otherwise.
7. Operates when ϕw/32 is selected as the internal clock; otherwise stops and stands by.
8. Stops and stands by.

5.2.1 Sleep Mode

In sleep mode, CPU operation is halted but the system clock oscillator, subclock oscillator, and on-chip peripheral modules function. In sleep (medium-speed) mode, the on-chip peripheral modules function at the clock frequency set by the MA1 and MA0 bits in SYSCR1. CPU register contents are retained.

Sleep mode is cleared by an interrupt. When an interrupt is requested, sleep mode is cleared and interrupt exception handling starts. Sleep mode is not cleared if the I bit in CCR is set to 1 or the requested interrupt is disabled by the interrupt enable bit. After sleep mode is cleared, a transition is made from sleep (high-speed) mode to active (high-speed) mode or from sleep (medium-speed) mode to active (medium-speed) mode.

When the $\overline{\text{RES}}$ pin goes low, the CPU goes into the reset state and sleep mode is cleared. Since an interrupt request signal is synchronous with the system clock, the maximum time of $2/\phi$ (s) may be delayed from the point at which an interrupt request signal occurs until the interrupt exception handling is started.

Furthermore, it sometimes operates with half state early timing at the time of transition to sleep (medium-speed) mode.

5.2.2 Standby Mode

In standby mode, the clock pulse generator stops, so the CPU and on-chip peripheral modules stop functioning. However, 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 as long as the voltage set by the RAM data retention voltage is provided. The I/O ports go to the high-impedance state.

Standby mode is cleared by an interrupt. When an interrupt is requested, the system clock pulse generator starts. After the time set in bits STS2 to STS0 in SYSCR1 has elapsed, standby mode is cleared and interrupt exception handling starts. After standby mode is cleared, a transition is made to active (high-speed) or active (medium-speed) mode according to the MSON bit in SYSCR2. Standby mode is not cleared if the I bit in CCR is set to 1 or the requested interrupt is disabled by the interrupt enable bit.

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

5.2.3 Watch Mode

In watch mode, the system clock oscillator and CPU operation stop and on-chip peripheral modules stop functioning except for the timer A, timer F, and asynchronous event counter. However, as long as the rated voltage is supplied, the contents of CPU registers, some on-chip peripheral module registers, and on-chip RAM are retained. The I/O ports retain their state before the transition.

Watch mode is cleared by an interrupt. When an interrupt is requested, watch mode is cleared and interrupt exception handling starts. When watch mode is cleared by an interrupt, a transition is made to active (high-speed) mode, active (medium-speed) mode, or subactive mode depending on the settings of the LSON bit in SYSCR1 and the MSON bit in SYSCR2. When the transition is made to active mode, after the time set in bits STS2 to STS0 in SYSCR1 has elapsed, interrupt exception handling starts. Watch mode is not cleared if the I bit in CCR is set to 1 or the requested interrupt is disabled by the interrupt enable bit.

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

5.2.4 Subsleep Mode

In subsleep mode, the CPU operation stops but on-chip peripheral modules other than the A/D converter and PWM function. As long as a required voltage is applied, the contents of CPU registers, the on-chip RAM, and some registers of the on-chip peripheral modules are retained. I/O ports keep the same states as before the transition.

Subsleep mode is cleared by an interrupt. When an interrupt is requested, subsleep mode is cleared and interrupt exception handling starts. After subsleep mode is cleared, a transition is made to subactive mode. Subsleep mode is not cleared if the I bit in CCR is set to 1 or the requested interrupt is disabled in the interrupt enable register.

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

5.2.5 Subactive Mode

In subactive mode, the system clock oscillator stops but on-chip peripheral modules other than the A/D converter and PWM function. As long as a required voltage is applied, the contents of some registers of the on-chip peripheral modules are retained.

Subactive mode is cleared by the SLEEP instruction. When subacitve mode is cleared, a transition to subsleep mode, active mode, or watch mode is made, depending on the combination of bits SSBY and LSON in SYSCR1, bits MSON and DTON in SYSCR2, and bit TMA3 in TMA. Subactive mode is not cleared if the I bit in CCR is set to 1 or the requested interrupt is disabled in the interrupt enable register.

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

The operating frequency of subactive mode is selected from $\phi_w/2$, $\phi_w/4$, and $\phi_w/8$ by the SA1 and SA0 bits in SYSCR2. After the SLEEP instruction is executed, the operating frequency changes to the frequency which is set before the execution.

5.2.6 Active (Medium-Speed) Mode

In active (medium-speed) mode, the system clock oscillator, subclock oscillator, CPU, and on-chip peripheral modules function.

Active (medium-speed) mode is cleared by the SLEEP instruction. When active (medium-speed) mode is cleared, a transition to standby mode is made depending on the combination of bits SSBY and LSON in SYSCR1 and bit TMA3 in TMA, a transition to watch mode is made depending on the combination of bit SSBY in SYSCR1 and bit TMA3 in TMA, or a transition to sleep mode is made depending on the combination of bits SSBY and LSON in SYSCR1. Moreover, a transition to active (high-speed) mode or subactive mode is made by a direct transition. Active (medium-sleep) mode is not entered if the I bit in CCR is set to 1 or the requested interrupt is disabled in the interrupt enable register. When the $\overline{\text{RES}}$ pin goes low, the CPU goes into the reset state and active (medium-sleep) mode is cleared.

Furthermore, it sometimes operates with half state early timing at the time of transition to active (medium-speed) mode.

In active (medium-speed) mode, the on-chip peripheral modules function at the clock frequency set by the MA1 and MA0 bits in SYSCR1.

5.3 Direct Transition

The CPU can execute programs in two modes: active and subactive mode. A direct transition is a transition between these two modes without stopping program execution. A direct transition can be made by executing a SLEEP instruction while the DTON bit in SYSCR2 is set to 1. The direct transition also enables operating frequency modification in active or subactive mode. After the mode transition, direct transition interrupt exception handling starts.

If the direct transition interrupt is disabled in interrupt permission register 2, a transition is made instead to sleep or watch mode. Note that if a direct transition is attempted while the I bit in CCR is set to 1, sleep or watch mode will be entered, and the resulting mode cannot be cleared by means of an interrupt.

- Direct transfer from active (high-speed) mode to active (medium-speed) mode

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

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

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

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

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

- Direct transfer from subactive mode to active (medium-speed) mode

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

5.3.1 Direct Transition from Active (High-Speed) Mode to Active (Medium-Speed) Mode

The time from the start of SLEEP instruction execution to the end of interrupt exception handling (the direct transition time) is calculated by equation (1).

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

……(1)

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

Legend:
tosc: OSC clock cycle time
tcyc: System clock (ϕ) cycle time

5.3.2 Direct Transition from Active (Medium-Speed) Mode to Active (High-Speed) Mode

The time from the start of SLEEP instruction execution to the end of interrupt exception handling (the direct transition time) is calculated by equation (2).

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

……..(2)

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

Legend:
tosc: OSC clock cycle time
tcyc: System clock (ϕ) cycle time

5.3.3 Direct Transition from Subactive Mode to Active (High-Speed) Mode

The time from the start of SLEEP instruction execution to the end of interrupt exception handling (the direct transition time) is calculated by equation (3).

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

……..(3)

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

Legend:
tosc: OSC clock cycle time
tw: Watch clock cycle time
tcyc: System clock (ϕ) cycle time
tsubcyc: Subclock (ϕ_{SUB}) cycle time

5.3.4 Direct Transition from Subactive Mode to Active (Medium-Speed) Mode

The time from the start of SLEEP instruction execution to the end of interrupt exception handling (the direct transition time) is calculated by equation (4).

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

……..(4)

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

Legend:
tosc: OSC clock cycle time
tw: Watch clock cycle time
tcyc: System clock (ϕ) cycle time
tsubcyc: Subclock (ϕ_{SUB}) cycle time

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

- Direct transition from active (high-speed) mode to subactive mode
  Since the mode transition is performed via watch mode, see section 5.5.2, Notes on External Input Signal Changes before/after Standby Mode.
- Direct transition from active (medium-speed) mode to subactive mode
  Since the mode transition is performed via watch mode, see section 5.5.2, Notes on External Input Signal Changes before/after Standby Mode.
- Direct transition from subactive mode to active (high-speed) mode
  Since the mode transition is performed via watch mode, see section 5.5.2, Notes on External Input Signal Changes before/after Standby Mode.
- Direct transition from subactive mode to active (medium-speed) mode
  Since the mode transition is performed via watch mode, see section 5.5.2, Notes on External Input Signal Changes before/after Standby Mode.

5.4 Module Standby Function

The module-standby function can be set to any peripheral module. In module standby mode, the clock supply to modules stops to enter the power-down mode. Module standby mode enables each on-chip peripheral module to enter the standby state by clearing a bit that corresponds to each module in CKSTPR1 and CKSTPR2 to 0 and cancels the mode by setting the bit to 1. (See section 5.1.3, Clock Halt Registers 1 and 2 (CKSTPR1 and CKSTPR2).)

5.5 Usage Notes

5.5.1 Standby Mode Transition and Pin States

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

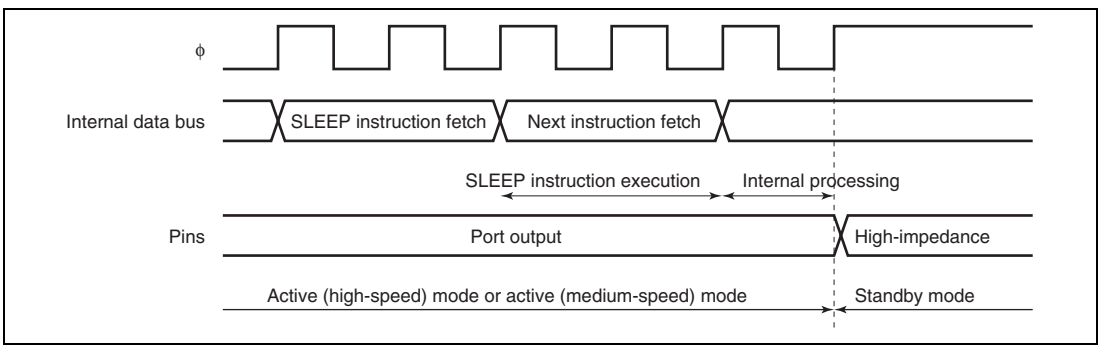


Figure 5.2 Standby Mode Transition and Pin States

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

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

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

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

   The case of falling edge capture is shown in figure 5.3.

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

3. Recommended timing of external input signals

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

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

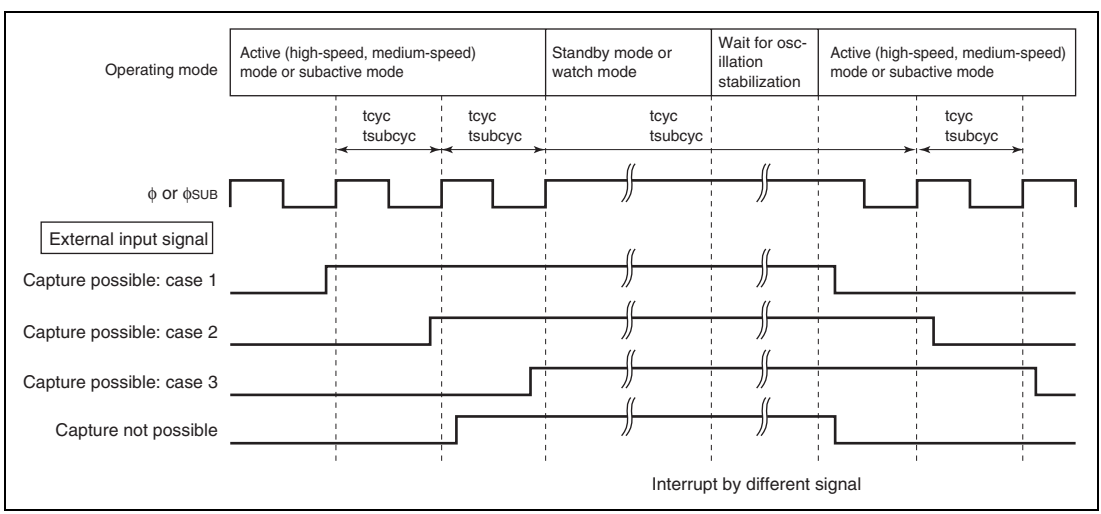


Figure 5.3 External Input Signal Capture when Signal Changes before/after Standby Mode or Watch Mode

4. Input pins to which these notes apply:
   $\overline{\text{IRQ1}}$, $\overline{\text{IRQ0}}$, $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$, and IRQAEC

Section 6 ROM

The H8/38704 has 32 kbytes of the on-chip mask ROM, the H8/38703 has 24 kbytes, the H8/38702 and H8/38702S have 16 kbytes, the H8/38701S has 12 kbytes, and the H8/38700S has 8 kbytes. The ROM is connected to the CPU by a 16-bit data bus, allowing high-speed two-state access for both byte data and word data. The H8/38704 and H8/38702 have flash ROM versions with 32-kbyte flash memory and 16-kbyte flash memory, respectively.

6.1 Block Diagram

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

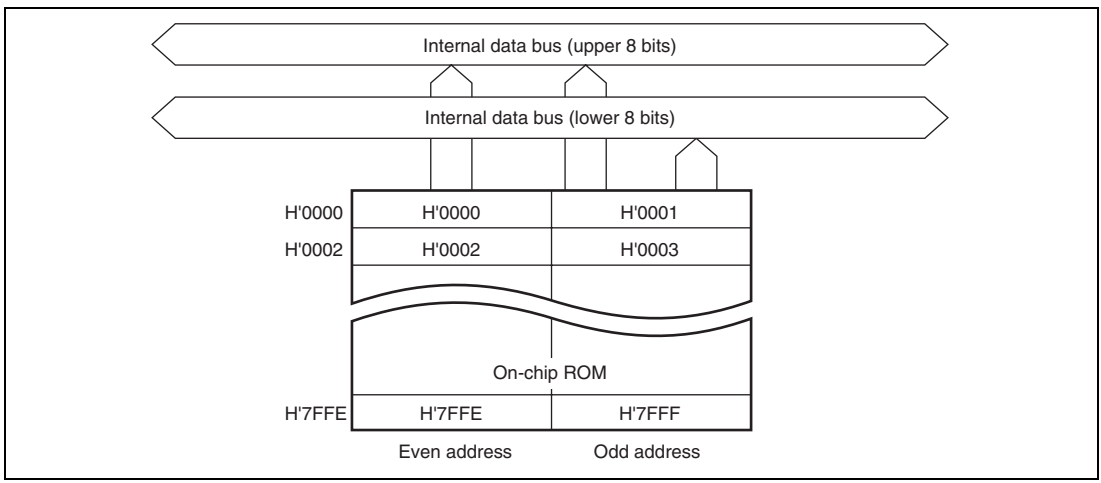


Figure 6.1 Block Diagram of ROM

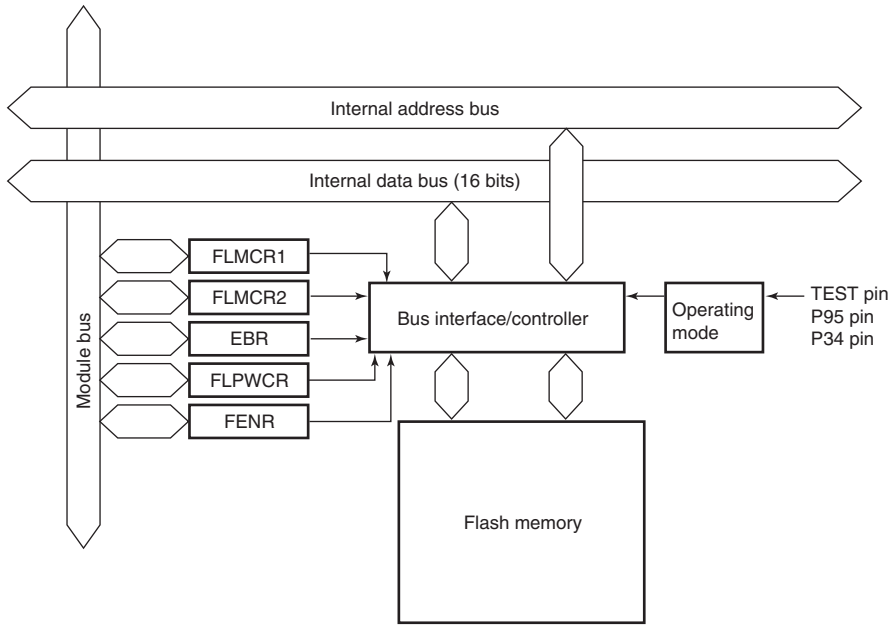
6.2 Overview of Flash Memory

6.2.1 Features

The features of the 32-kbyte or 16-kbyte flash memory built into the flash memory version are summarized below.

- Programming/erase methods
  - The flash memory is programmed 128 bytes at a time. Erase is performed in single-block units. The 32-kbyte flash memory are configured as 1 kbyte × 4 blocks and 28 kbytes × 1 block. The 16-kbyte flash memory is configured as 1 kbyte × 4 blocks and 12 kbytes × 1 block. To erase the entire flash memory, each block must be erased in turn.
- On-board programming
  - On-board programming/erasing can be done in boot mode, in which the boot program built into the chip is started to erase or program of the entire flash memory. In normal user program mode, individual blocks can be erased or programmed.
- Programmer mode
  - Flash memory can be programmed/erased in programmer mode using a PROM programmer, as well as in on-board programming mode.
- Automatic bit rate adjustment
  - For data transfer in boot mode, this LSI's bit rate can be automatically adjusted to match the transfer bit rate of the host.
- Programming/erasing protection
  - Sets software protection against flash memory programming/erasing.
- Power-down mode
  - Operation of the power supply circuit can be partly halted in subactive mode. As a result, flash memory can be read with low power consumption.

6.2.2 Block Diagram



[Legend]
FLMCR1: Flash memory control register 1
FLMCR2: Flash memory control register 2
EBR: Erase block register
FLPWCR: Flash memory power control register
FENR: Flash memory enable register

Figure 6.2 Block Diagram of Flash Memory

6.2.3 Block Configuration

Figure 6.3 shows the block configuration of 32-kbyte flash memory. The thick lines indicate erasing units, the narrow lines indicate programming units, and the values are addresses. The 32-kbyte flash memory is divided into 1 kbyte × 4 blocks and 28 kbytes × 1 block. Erasing is performed in these units. The 16-kbyte flash memory is divided into 1 kbyte × 4 blocks and 12 kbytes × 1 block. Programming is performed in 128-byte units starting from an address with lower eight bits H'00 or H'80.

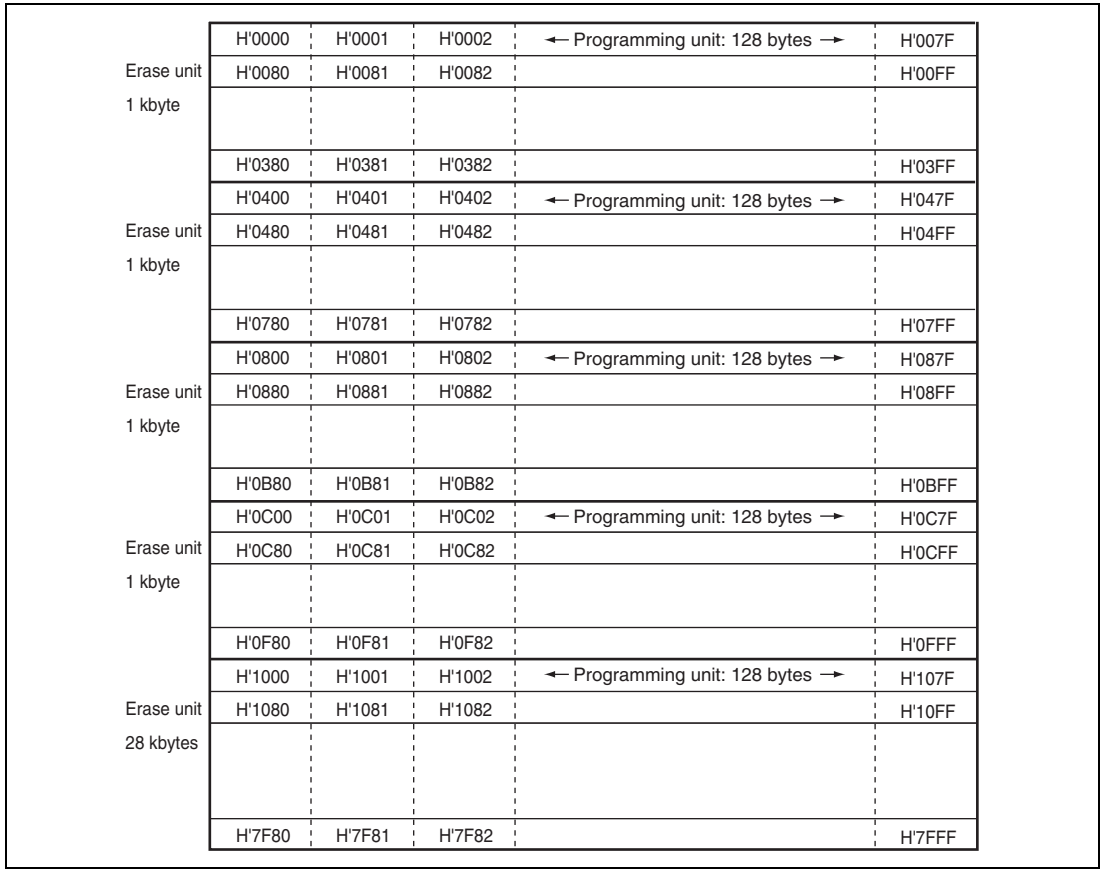


Figure 6.3(1) Block Configuration of 32-kbyte Flash Memory

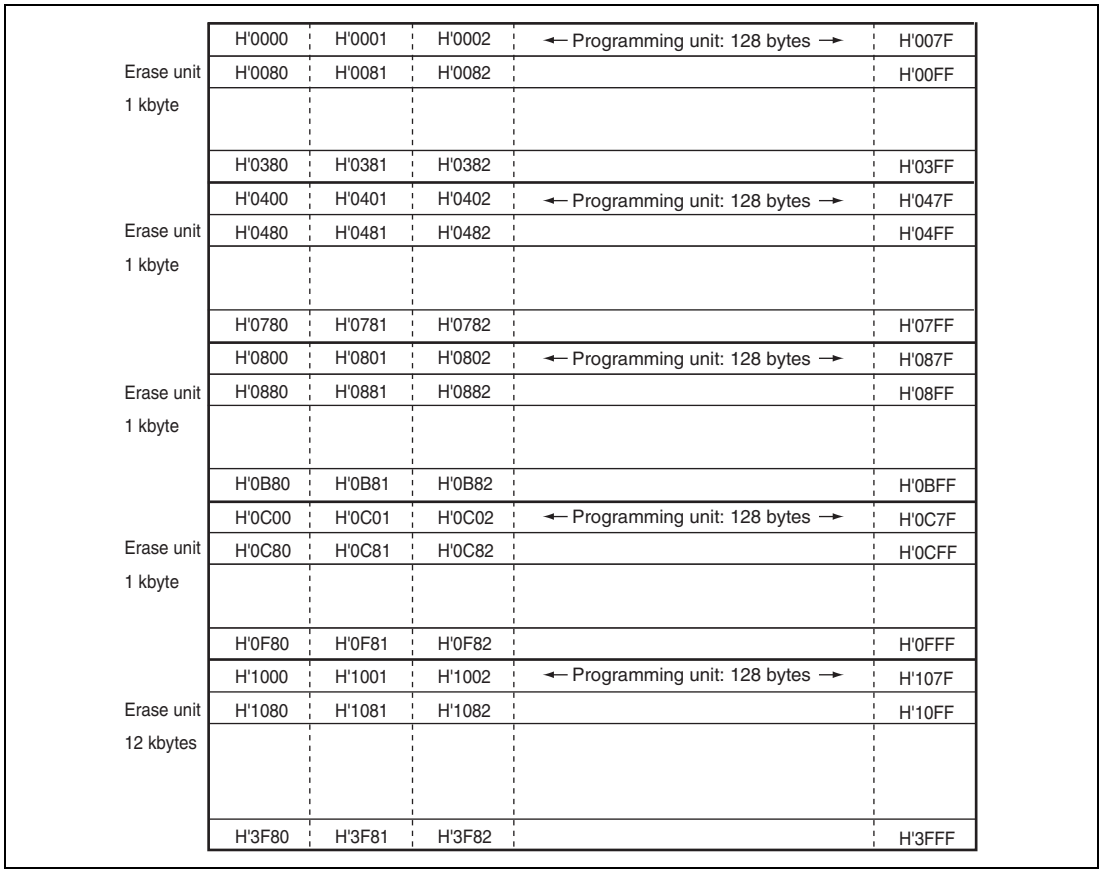


Figure 6.3(2) Block Configuration of 16-kbyte Flash Memory

6.3 Register Descriptions

The flash memory has the following registers.

- Flash memory control register 1 (FLMCR1)
- Flash memory control register 2 (FLMCR2)
- Erase block register (EBR)
- Flash memory power control register (FLPWCR)
- Flash memory enable register (FENR)

6.3.1 Flash Memory Control Register 1 (FLMCR1)

FLMCR1 is a register that makes the flash memory change to program mode, program-verify mode, erase mode, or erase-verify mode. For details on register setting, refer to section 6.5, Flash Memory Programming/Erasing.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	—	Reserved This bit is always read as 0.
6	SWE	0	R/W	Software Write Enable When this bit is set to 1, flash memory programming/erasing is enabled. When this bit is cleared to 0, flash memory programming/erasing is invalid. Other FLMCR1 bits and all EBR bits cannot be set.
5	ESU	0	R/W	Erase Setup When this bit is set to 1, the flash memory changes to the erase setup state. When it is cleared to 0, the erase setup state is cancelled. Set this bit to 1 before setting the E bit to 1 in FLMCR1.
4	PSU	0	R/W	Program Setup When this bit is set to 1, the flash memory changes to the program setup state. When it is cleared to 0, the program setup state is cancelled. Set this bit to 1 before setting the P bit in FLMCR1.
3	EV	0	R/W	Erase-Verify When this bit is set to 1, the flash memory changes to erase-verify mode. When it is cleared to 0, erase-verify mode is cancelled.
2	PV	0	R/W	Program-Verify When this bit is set to 1, the flash memory changes to program-verify mode. When it is cleared to 0, program-verify mode is cancelled.
1	E	0	R/W	Erase When this bit is set to 1, and while the SWE = 1 and ESU = 1 bits are 1, the flash memory changes to erase mode. When it is cleared to 0, erase mode is cancelled.
0	P	0	R/W	Program When this bit is set to 1, and while the SWE = 1 and PSU = 1 bits are 1, the flash memory changes to program mode. When it is cleared to 0, program mode is cancelled.

Note: Bits SWE, PSU, EV, PV, E, and P should not be set at the same time.

6.3.2 Flash Memory Control Register 2 (FLMCR2)

FLMCR2 is a register that displays the state of flash memory programming/erasing. FLMCR2 is a read-only register, and should not be written to.

Bit	Bit Name	Initial Value	R/W	Description
7	FLER	0	R	Flash Memory Error Indicates that an error has occurred during an operation on flash memory (programming or erasing). When flash memory goes to the error-protection state, this bit is set to 1. See section 6.6.3, Error Protection, for details.
6 to 0	—	All 0	—	Reserved These bits are always read as 0.

6.3.3 Erase Block Register (EBR)

EBR specifies the flash memory erase area block. EBR is initialized to H'00 when the SWE bit in FLMCR1 is 0. Do not set more than one bit at a time, as this will cause all the bits in EBR to be automatically cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	—	Reserved These bits are always read as 0.
4	EB4	0	R/W	H8/38704F: When this bit is set to 1, 28 kbytes of H'1000 to H'7FFF will be erased. H8/38702F: When this bit is set to 1, 12 kbytes of H'1000 to H'3FFF will be erased.
3	EB3	0	R/W	When this bit is set to 1, 1 kbyte of H'0C00 to H'0FFF will be erased.
2	EB2	0	R/W	When this bit is set to 1, 1 kbyte of H'0800 to H'0BFF will be erased.
1	EB1	0	R/W	When this bit is set to 1, 1 kbyte of H'0400 to H'07FF will be erased.
0	EB0	0	R/W	When this bit is set to 1, 1 kbyte of H'0000 to H'03FF will be erased.

6.3.4 Flash Memory Power Control Register (FLPWCR)

FLPWCR enables or disables a transition to the flash memory power-down mode when the LSI switches to subactive mode. There are two modes: mode in which operation of the power supply circuit of flash memory is partly halted in power-down mode and flash memory can be read, and mode in which even if a transition is made to subactive mode, operation of the power supply circuit of flash memory is retained and flash memory can be read.

Bit	Bit Name	Initial Value	R/W	Description
7	PDWND	0	R/W	Power-Down Disable When this bit is 0 and a transition is made to subactive mode, the flash memory enters the power-down mode. When this bit is 1, the flash memory remains in the normal mode even after a transition is made to subactive mode.
6 to 0	—	All 0	—	Reserved These bits are always read as 0.

6.3.5 Flash Memory Enable Register (FENR)

Bit 7 (FLSHE) in FENR enables or disables the CPU access to the flash memory control registers, FLMCR1, FLMCR2, EBR, and FLPWCR.

Bit	Bit Name	Initial Value	R/W	Description
7	FLSHE	0	R/W	Flash Memory Control Register Enable Flash memory control registers can be accessed when this bit is set to 1. Flash memory control registers cannot be accessed when this bit is set to 0.
6 to 0	—	All 0	—	Reserved These bits are always read as 0.

6.4 On-Board Programming Modes

There are two modes for programming/erasing of the flash memory; boot mode, which enables on-board programming/erasing, and programmer mode, in which programming/erasing is performed with a PROM programmer. On-board programming/erasing can also be performed in user program mode. At reset-start in reset mode, this LSI changes to a mode depending on the TEST pin settings, P95 pin settings, and input level of each port, as shown in table 6.1. The input level of each pin must be defined four states before the reset ends.

When changing to boot mode, the boot program built into this LSI is initiated. The boot program transfers the programming control program from the externally-connected host to on-chip RAM via SCI3. After erasing the entire flash memory, the programming control program is executed. This can be used for programming initial values in the on-board state or for a forcible return when programming/erasing can no longer be done in user program mode. In user program mode, individual blocks can be erased and programmed by branching to the user program/erase control program prepared by the user.

Table 6.1 Setting Programming Modes

TEST	P95	P34	PB0	PB1	PB2	LSI State after Reset End
0	1	x	x	x	x	User Mode
0	0	1	x	x	x	Boot Mode
1	x	x	0	0	0	Programmer Mode

[Legend]
x: Don't care.

6.4.1 Boot Mode

Table 6.2 shows the boot mode operations between reset end and branching to the programming control program.

1. When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. Prepare a programming control program in accordance with the description in section 6.5, Flash Memory Programming/Erasing.
2. The SCI3 should be set to asynchronous mode, and the transfer format as follows: 8-bit data, 1 stop bit, and no parity. Since the inversion function of SPCR is configured not to inverse data of the TXD pin and RXD pin, do not place an inversion circuit between the host and this LSI.

3. When the boot program is initiated, the chip measures the low-level period of asynchronous SCI communication data (H'00) transmitted continuously from the host. The chip then calculates the bit rate of transmission from the host, and adjusts the SCI3 bit rate to match that of the host. The reset should end with the RXD pin high. The RXD and TXD pins should be pulled up on the board if necessary. After the reset is complete, it takes approximately 100 states before the chip is ready to measure the low-level period.
4. After matching the bit rates, the chip transmits one H'00 byte to the host to indicate the completion of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the chip. If reception could not be performed normally, initiate boot mode again by a reset. Depending on the host's transfer bit rate and system clock frequency of this LSI, there will be a discrepancy between the bit rates of the host and the chip. To operate the SCI properly, set the host's transfer bit rate and system clock frequency of this LSI within the ranges listed in table 6.3.
5. In boot mode, a part of the on-chip RAM area is used by the boot program. The area H'F780 to H'FEEF is the area to which the programming control program is transferred from the host. The boot program area cannot be used until the execution state in boot mode switches to the programming control program.
6. Before branching to the programming control program, the chip terminates transfer operations by SCI3 (by clearing the RE and TE bits in SCR to 0), however the adjusted bit rate value remains set in BRR. Therefore, the programming control program can still use it for transfer of write data or verify data with the host. The TXD pin is high (PCR42 = 1, P42 = 1). The contents of the CPU general registers are undefined immediately after branching to the programming control program. These registers must be initialized at the beginning of the programming control program, as the stack pointer (SP), in particular, is used implicitly in subroutine calls, etc.
7. Boot mode can be cleared by a reset. End the reset after driving the reset pin low, waiting at least 20 states, and then setting the TEST pin and P95 pin. Boot mode is also cleared when a WDT overflow occurs.
8. Do not change the TEST pin and P95 pin input levels in boot mode.

Table 6.2 Boot Mode Operation

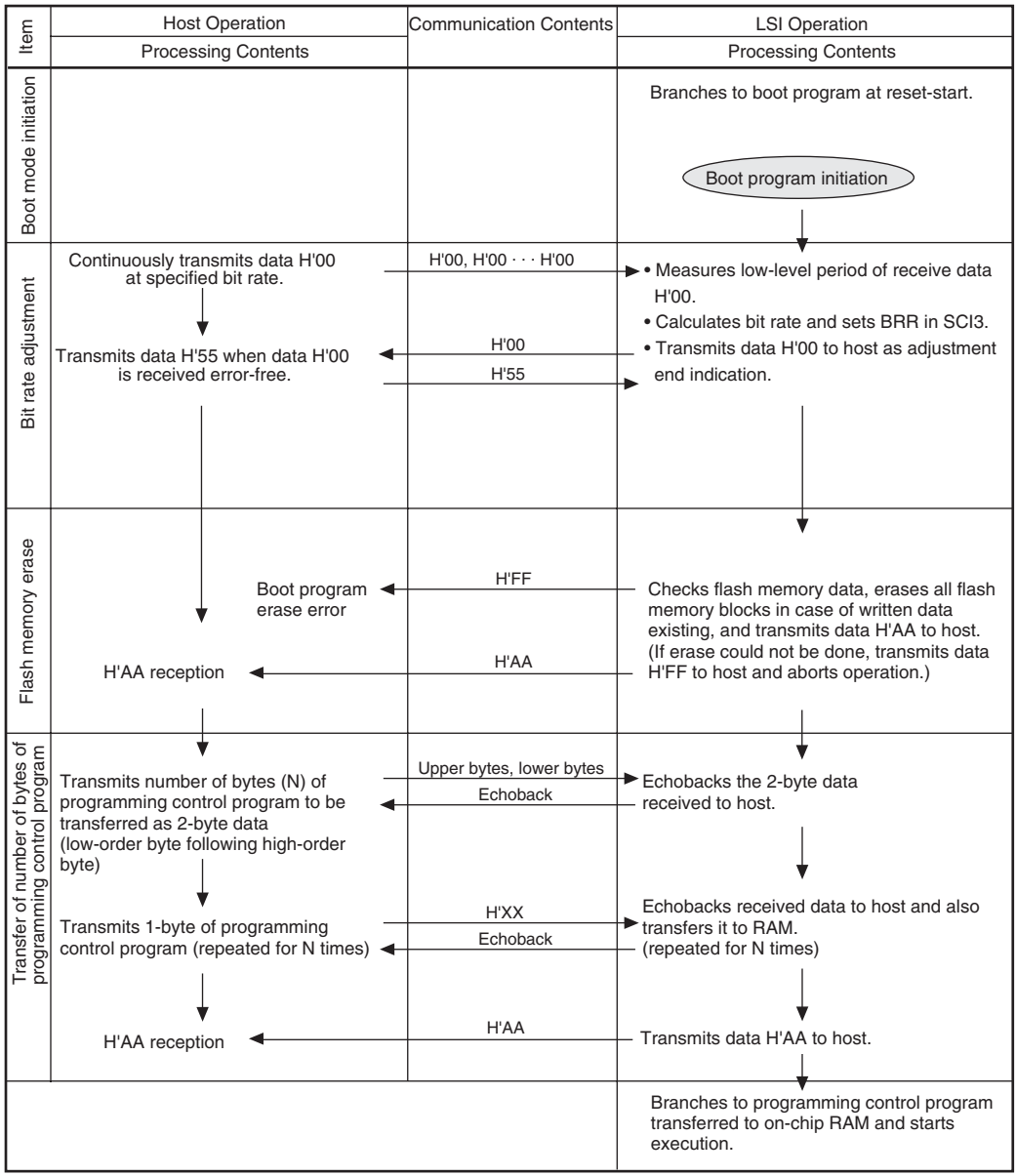


Table 6.3 Oscillation Frequencies for which Automatic Adjustment of LSI Bit Rate is Possible (f_{osc})

Host Bit Rate	Oscillation Frequency Range of LSI (f_{osc})
4,800 bps	8 to 10 MHz
2,400 bps	4 to 10 MHz
1,200 bps	2 to 10 MHz

6.4.2 Programming/Erasing in User Program Mode

User program mode means the execution state of the user program. On-board programming/erasing of an individual flash memory block can also be performed in user program mode by branching to a user program/erase control program. The user must set branching conditions and provide on-board means of supplying programming data. The flash memory must contain the user program/erase control program or a program that provides the user program/erase control program from external memory. As the flash memory itself cannot be read during programming/erasing, transfer the user program/erase control program to on-chip RAM, as in boot mode. Figure 6.4 shows a sample procedure for programming/erasing in user program mode. Prepare a user program/erase control program in accordance with the description in section 6.5, Flash Memory Programming/Erasing.

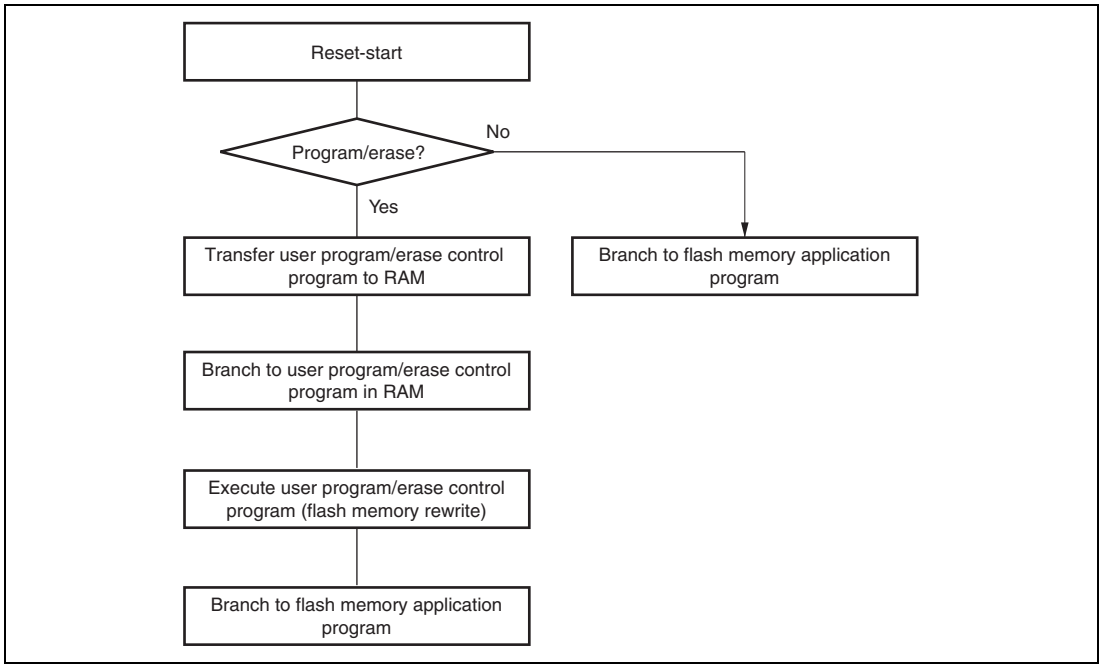


Figure 6.4 Programming/Erasing Flowchart Example in User Program Mode

6.5 Flash Memory Programming/Erasing

A software method using the CPU is employed to program and erase flash memory in the on-board programming modes. Depending on the FLMCR1 setting, the flash memory operates in one of the following four modes: Program mode, program-verify mode, erase mode, and erase-verify mode. The programming control program in boot mode and the user program/erase control program in user program mode use these operating modes in combination to perform programming/erasing. Flash memory programming and erasing should be performed in accordance with the descriptions in section 6.5.1, Program/Program-Verify and section 6.5.2, Erase/Erase-Verify, respectively.

6.5.1 Program/Program-Verify

When writing data or programs to the flash memory, the program/program-verify flowchart shown in figure 6.5 should be followed. Performing programming operations according to this flowchart will enable data or programs to be written to the flash memory without subjecting the chip to voltage stress or sacrificing program data reliability.

1. Programming must be done to an empty address. Do not reprogram an address to which programming has already been performed.
2. Programming should be carried out 128 bytes at a time. A 128-byte data transfer must be performed even if writing fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
3. Prepare the following data storage areas in RAM: A 128-byte programming data area, a 128-byte reprogramming data area, and a 128-byte additional-programming data area. Perform reprogramming data computation according to table 6.4, and additional programming data computation according to table 6.5.
4. Consecutively transfer 128 bytes of data in byte units from the reprogramming data area or additional-programming data area to the flash memory. The program address and 128-byte data are latched in the flash memory. The lower 8 bits of the start address in the flash memory destination area must be H'00 or H'80.
5. The time during which the P bit is set to 1 is the programming time. Table 6.6 shows the allowable programming times.
6. The watchdog timer (WDT) is set to prevent overprogramming due to program runaway, etc. An overflow cycle of approximately 6.6 ms is allowed.
7. For a dummy write to a verify address, write 1-byte data H'FF to an address whose lower one bit is B'0. Verify data can be read in word or longword units from the address to which a dummy write was performed.
8. The maximum number of repetitions of the program/program-verify sequence of the same bit is 1,000.

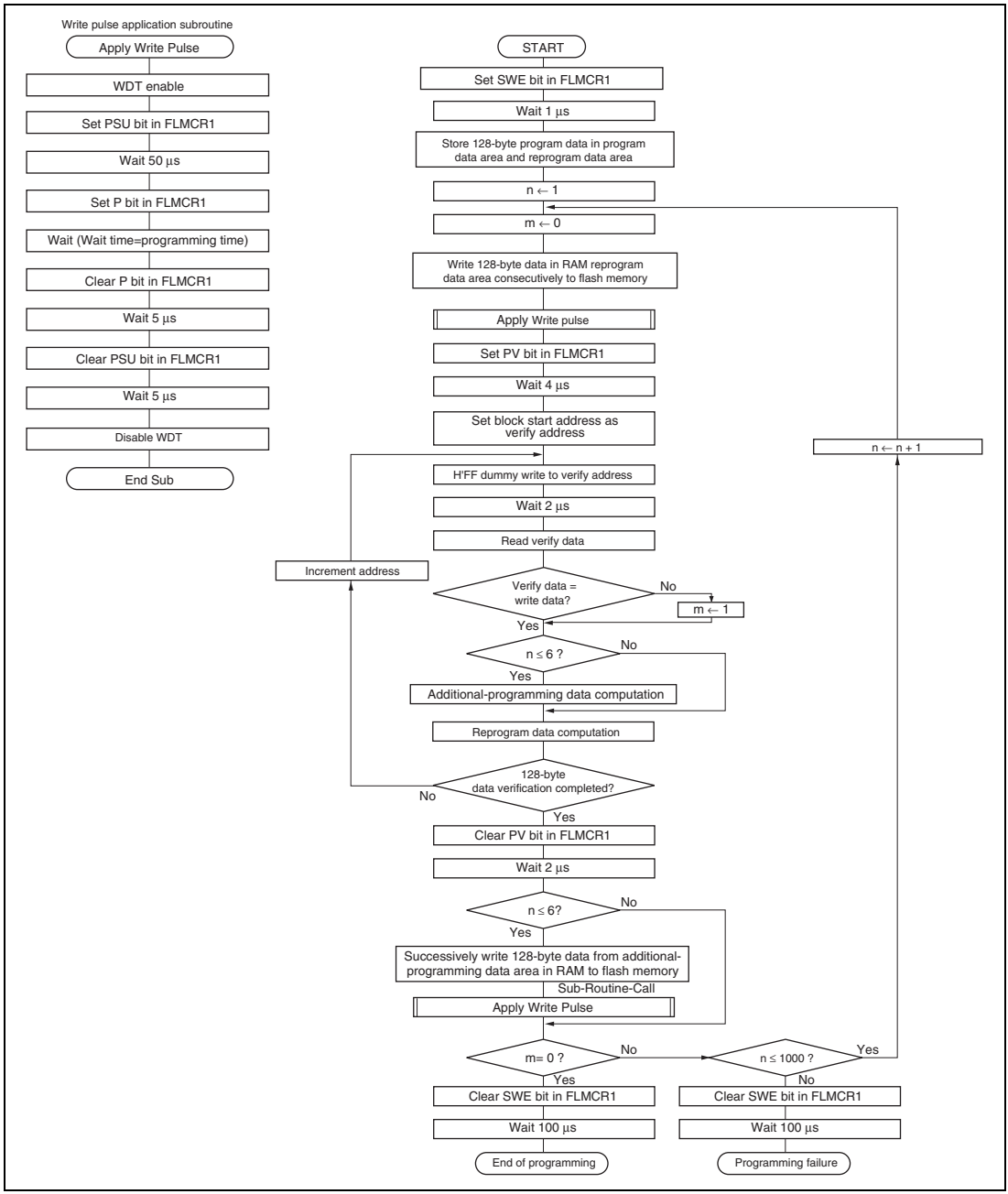


Figure 6.5 Program/Program-Verify Flowchart

Table 6.4 Reprogram Data Computation Table

Program Data	Verify Data	Reprogram Data	Comments
0	0	1	Programming completed
0	1	0	Reprogram bit
1	0	1	—
1	1	1	Remains in erased state

Table 6.5 Additional-Program Data Computation Table

Reprogram Data	Verify Data	Additional-Program Data	Comments
0	0	0	Additional-program bit
0	1	1	No additional programming
1	0	1	No additional programming
1	1	1	No additional programming

Table 6.6 Programming Time

n (Number of Writes)	Programming Time	In Additional Programming	Comments
1 to 6	30	10	
7 to 1,000	200	—	

Note: Time shown in μs.

6.5.2 Erase/Erase-Verify

When erasing flash memory, the erase/erase-verify flowchart shown in figure 6.6 should be followed.

1. Prewriting (setting erase block data to all 0s) is not necessary.
2. Erasing is performed in block units. Make only a single-bit specification in the erase block register (EBR). To erase multiple blocks, each block must be erased in turn.
3. The time during which the E bit is set to 1 is the flash memory erase time.
4. The watchdog timer (WDT) is set to prevent overerasing due to program runaway, etc. An overflow cycle of approximately 19.8 ms is allowed.
5. For a dummy write to a verify address, write 1-byte data H'FF to an address whose lower 1 bit is B'0. Verify data can be read in word or longword units from the address to which a dummy write was performed.
6. If the read data is not erased successfully, set erase mode again, and repeat the erase/erase-verify sequence as before. The maximum number of repetitions of the erase/erase-verify sequence is 100.

6.5.3 Interrupt Handling when Programming/Erasing Flash Memory

All interrupts are disabled while flash memory is being programmed or erased, or while the boot program is executing, for the following three reasons:

1. Interrupt during programming/erasing may cause a violation of the programming or erasing algorithm, with the result that normal operation cannot be assured.
2. If interrupt exception handling starts before the vector address is written or during programming/erasing, a correct vector cannot be fetched and the CPU malfunctions.
3. If an interrupt occurs during boot program execution, normal boot mode sequence cannot be carried out.

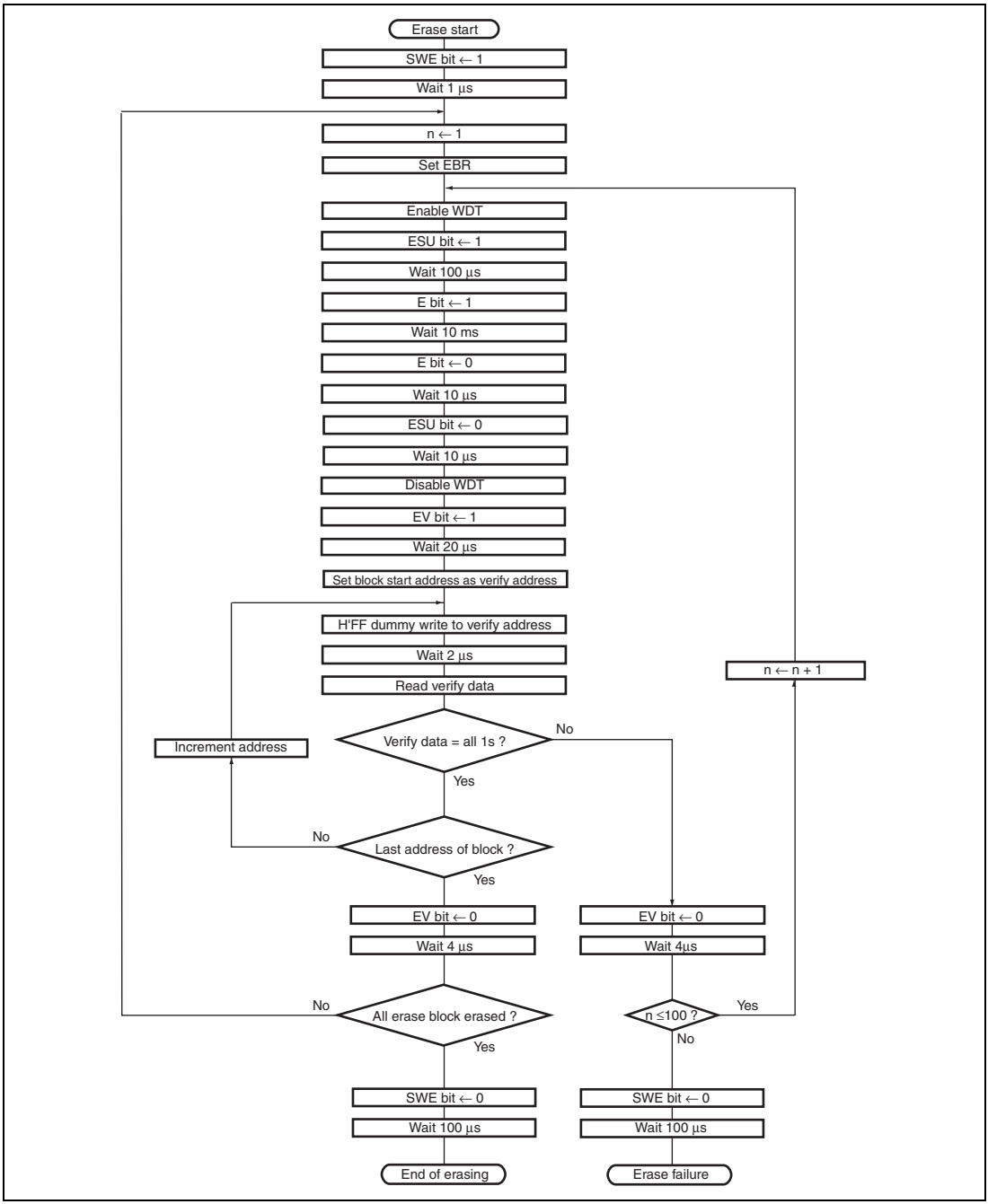


Figure 6.6 Erase/Erase-Verify Flowchart

6.6 Program/Erase Protection

There are three kinds of flash memory program/erase protection; hardware protection, software protection, and error protection.

6.6.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted because of a transition to reset, subactive mode, subsleep mode, watch mode, or standby mode. Flash memory control register 1 (FLMCR1), flash memory control register 2 (FLMCR2), and erase block register (EBR) are initialized. In a reset via the $\overline{\text{RES}}$ pin, the reset state is not entered unless the $\overline{\text{RES}}$ pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width specified in the AC Characteristics section.

6.6.2 Software Protection

Software protection can be implemented against programming/erasing of all flash memory blocks by clearing the SWE bit in FLMCR1. When software protection is in effect, setting the P or E bit in FLMCR1 does not cause a transition to program mode or erase mode. By setting the erase block register (EBR), erase protection can be set for individual blocks. When EBR is set to H'00, erase protection is set for all blocks.

6.6.3 Error Protection

In error protection, an error is detected when CPU runaway occurs during flash memory programming/erasing, or operation is not performed in accordance with the program/erase algorithm, and the program/erase operation is aborted. Aborting the program/erase operation prevents damage to the flash memory due to overprogramming or overerasing.

When the following errors are detected during programming/erasing of flash memory, the FLER bit in FLMCR2 is set to 1, and the error protection state is entered.

- When the flash memory of the relevant address area is read during programming/erasing (including vector read and instruction fetch)
- Immediately after exception handling excluding a reset during programming/erasing
- When a SLEEP instruction is executed during programming/erasing

The FLMCR1, FLMCR2, and EBR settings are retained, however program mode or erase mode is aborted at the point at which the error occurred. Program mode or erase mode cannot be re-entered by re-setting the P or E bit. However, PV and EV bit setting is enabled, and a transition can be made to verify mode. Error protection can be cleared only by a power-on reset.

6.7 Programmer Mode

In programmer mode, a PROM programmer can be used to perform programming/erasing via a socket adapter, just as a discrete flash memory. Use a PROM programmer that supports the MCU device type with the on-chip Renesas Technology (former Hitachi Ltd.) 64-kbyte flash memory (FZTAT64V3). A 10-MHz input clock is required. For the conditions for transition to programmer mode, see table 6.1.

6.7.1 Socket Adapter

The socket adapter converts the pin allocation of the HD64F38704 and HD64F38702 to that of the discrete flash memory HN28F101. The address of the on-chip flash memory is H'0000 to H'7FFF. Figure 6.7 shows a socket-adapter-pin correspondence diagram.

6.7.2 Programmer Mode Commands

The following commands are supported in programmer mode.

- Memory Read Mode
- Auto-Program Mode
- Auto-Erase Mode
- Status Read Mode

Status polling is used for auto-programming, auto-erasing, and status read modes. In status read mode, detailed internal information is output after the execution of auto-programming or auto-erasing. Table 6.7 shows the sequence of each command. In auto-programming mode, 129 cycles are required since 128 bytes are written at the same time. In memory read mode, the number of cycles depends on the number of address write cycles (n).

Table 6.7 Command Sequence in Programmer Mode

Command Name	Number of Cycles	1st Cycle			2nd Cycle		
		Mode	Address	Data	Mode	Address	Data
Memory read	1 + n	Write	X	H'00	Read	RA	Dout
Auto-program	129	Write	X	H'40	Write	WA	Din
Auto-erase	2	Write	X	H'20	Write	X	H'20
Status read	2	Write	X	H'71	Write	X	H'71

[Legend] n: Number of address write cycles

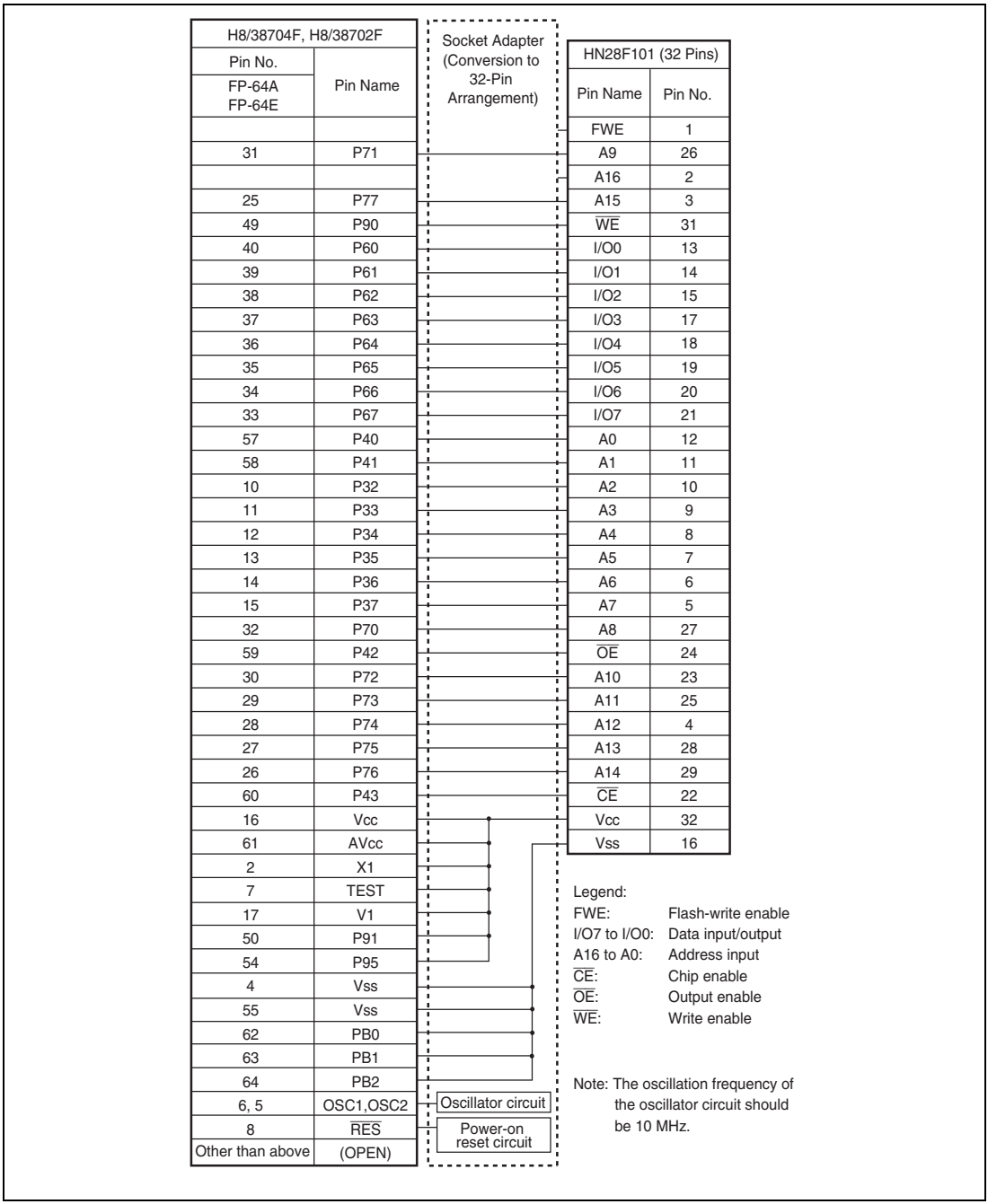


Figure 6.7 Socket Adapter Pin Correspondence Diagram (H8/38704F, H8/38702F)

6.7.3 Memory Read Mode

After completion of auto-program/auto-erase/status read operations, a transition is made to the command wait state. When reading memory contents, a transition to memory read mode must first be made with a command write, after which the memory contents are read. Once memory read mode has been entered, consecutive reads can be performed.

1. In memory read mode, command writes can be performed in the same way as in the command wait state.
2. After powering on, memory read mode is entered.
3. Tables 6.8 to 6.10 show the AC characteristics.

Table 6.8 AC Characteristics in Transition to Memory Read Mode

(Conditions: V_{CC} = 3.3 V ±0.3 V, V_{SS} = 0 V, Ta = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Test Condition
Command write cycle	t_{nxtc}	20	—	µs	Figure 6.8
$\overline{CE}$ hold time	t_{ceh}	0	—	ns	
$\overline{CE}$ setup time	t_{ces}	0	—	ns	
Data hold time	t_{dh}	50	—	ns	
Data setup time	t_{ds}	50	—	ns	
Write pulse width	t_{wep}	70	—	ns	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

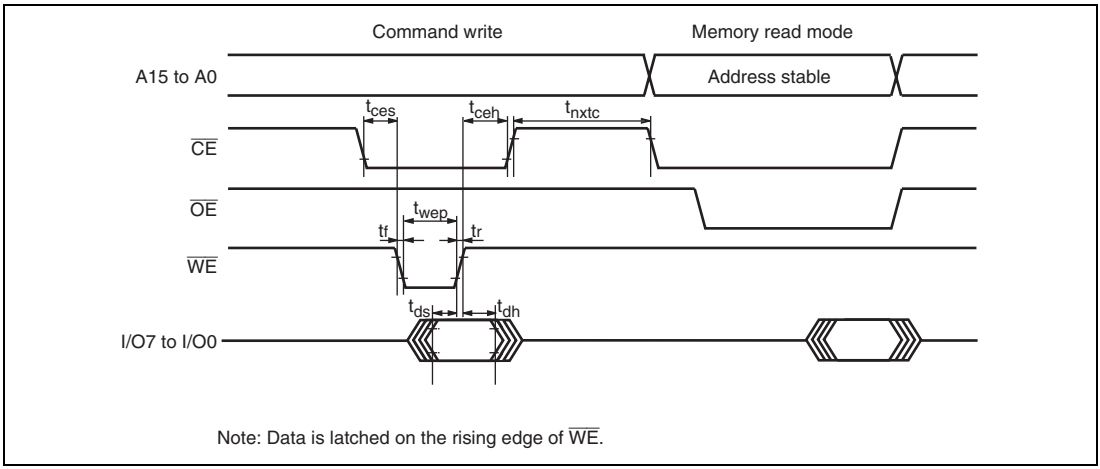


Figure 6.8 Timing Waveforms for Memory Read after Command Write

Table 6.9 AC Characteristics in Transition from Memory Read Mode to Another Mode

(Conditions: V_{CC} = 3.3 V ±0.3 V, V_{SS} = 0 V, Ta = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Test Condition
Command write cycle	t_{nxtc}	20	—	µs	Figure 6.9
$\overline{CE}$ hold time	t_{ceh}	0	—	ns	
$\overline{CE}$ setup time	t_{ces}	0	—	ns	
Data hold time	t_{dh}	50	—	ns	
Data setup time	t_{ds}	50	—	ns	
Write pulse width	t_{wep}	70	—	ns	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

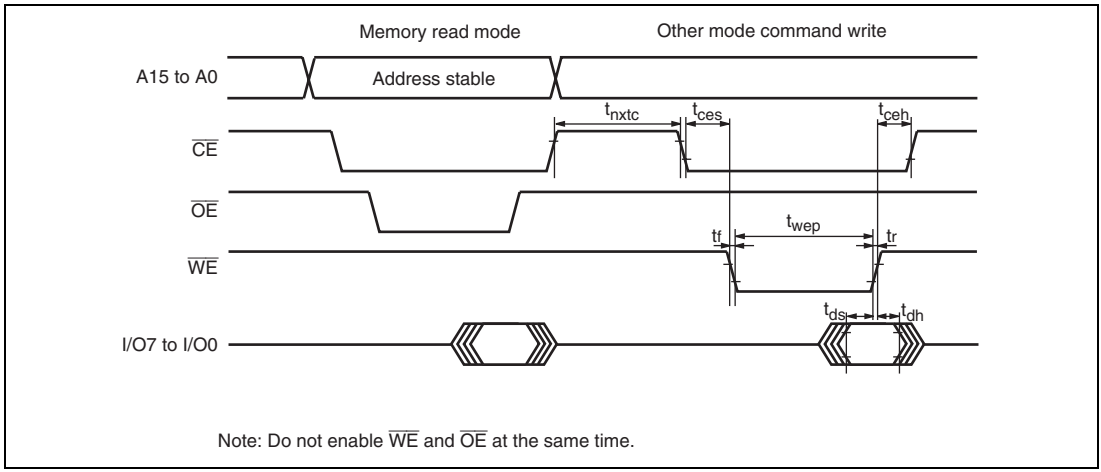


Figure 6.9 Timing Waveforms in Transition from Memory Read Mode to Another Mode

Table 6.10 AC Characteristics in Memory Read Mode

(Conditions: V_{CC} = 3.3 V ±0.3 V, V_{SS} = 0 V, Ta = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Test Condition
Access time	t_{acc}	—	20	µs	Figures 6.10 and 6.11
$\overline{CE}$ output delay time	t_{ce}	—	150	ns	
$\overline{OE}$ output delay time	t_{oe}	—	150	ns	
Output disable delay time	t_{df}	—	100	ns	
Data output hold time	t_{oh}	5	—	ns	

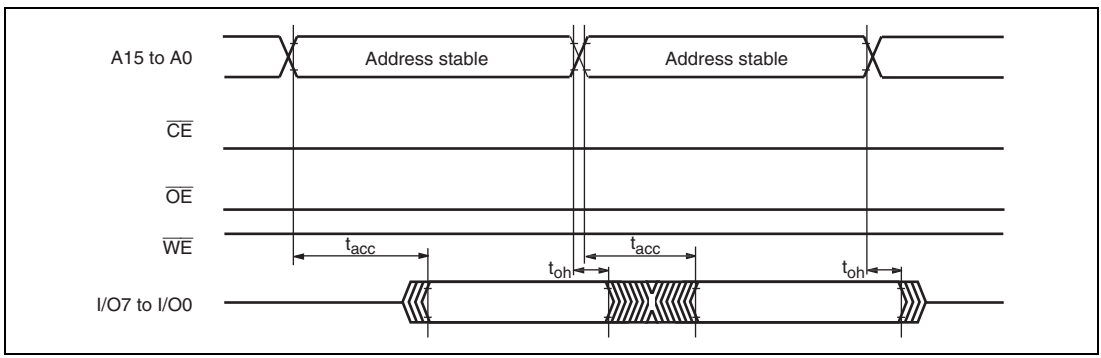


Figure 6.10 Timing Waveforms in $\overline{CE}$ and $\overline{OE}$ Enable State Read

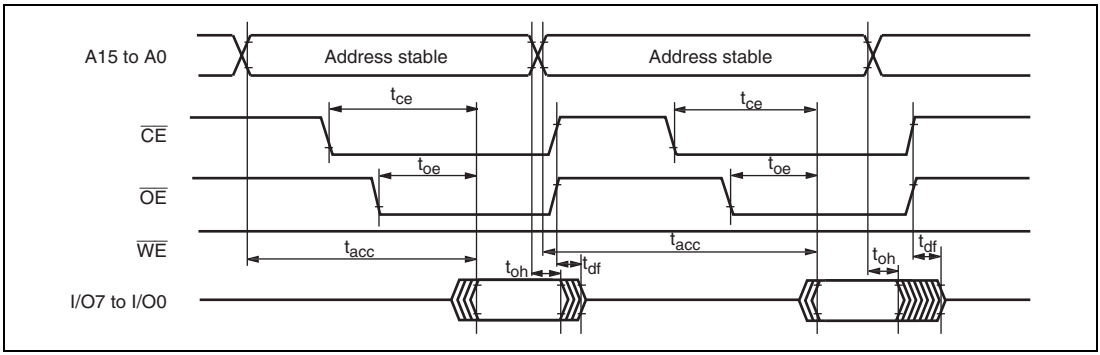


Figure 6.11 Timing Waveforms in $\overline{CE}$ and $\overline{OE}$ Clock System Read

6.7.4 Auto-Program Mode

1. When reprogramming previously programmed addresses, perform auto-erasing before auto-programming.
2. Perform auto-programming once only on the same address block. It is not possible to program an address block that has already been programmed.
3. In auto-program mode, 128 bytes are programmed simultaneously. This should be carried out by executing 128 consecutive byte transfers. A 128-byte data transfer is necessary even when programming fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
4. The lower 7 bits of the transfer address must be low. If a value other than an effective address is input, processing will switch to a memory write operation but a write error will be flagged.
5. Memory address transfer is performed in the second cycle (figure 6.12). Do not perform transfer after the third cycle.
6. Do not perform a command write during a programming operation.
7. Perform one auto-program operation for a 128-byte block for each address. Two or more additional programming operations cannot be performed on a previously programmed address block.
8. Confirm normal end of auto-programming by checking I/O6. Alternatively, status read mode can also be used for this purpose (I/O7 status polling uses the auto-program operation end decision pin).
9. Status polling I/O6 and I/O7 pin information is retained until the next command write. As long as the next command write has not been performed, reading is possible by enabling $\overline{CE}$ and $\overline{OE}$.
10. Table 6.11 shows the AC characteristics.

Table 6.11 AC Characteristics in Auto-Program Mode

(Conditions: V_{CC} = 3.3 V ±0.3 V, V_{SS} = 0 V, Ta = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Test Condition
Command write cycle	t_{nxtc}	20	—	µs	Figure 6.12
$\overline{CE}$ hold time	t_{ceh}	0	—	ns	
$\overline{CE}$ setup time	t_{ces}	0	—	ns	
Data hold time	t_{dh}	50	—	ns	
Data setup time	t_{ds}	50	—	ns	
Write pulse width	t_{wep}	70	—	ns	
Status polling start time	t_{wsts}	1	—	ms	
Status polling access time	t_{spa}	—	150	ns	
Address setup time	t_{as}	0	—	ns	
Address hold time	t_{ah}	60	—	ns	
Memory write time	t_{write}	1	3000	ms	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

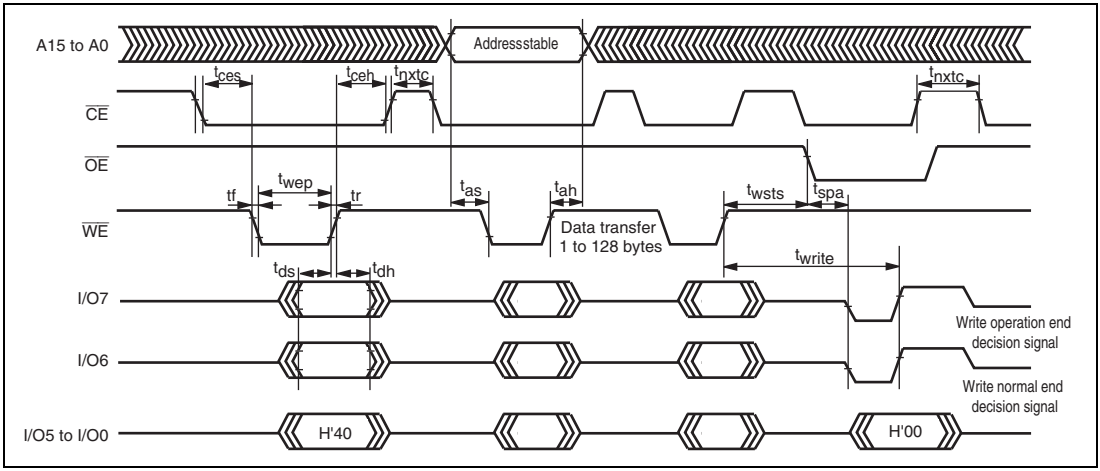


Figure 6.12 Timing Waveforms in Auto-Program Mode

6.7.5 Auto-Erase Mode

1. Auto-erase mode supports only entire memory erasing.
2. Do not perform a command write during auto-erasing.
3. Confirm normal end of auto-erasing by checking I/O6. Alternatively, status read mode can also be used for this purpose (I/O7 status polling uses the auto-erase operation end decision pin).
4. Status polling I/O6 and I/O7 pin information is retained until the next command write. As long as the next command write has not been performed, reading is possible by enabling $\overline{CE}$ and $\overline{OE}$.
5. Table 6.12 shows the AC characteristics.

Table 6.12 AC Characteristics in Auto-Erase Mode

(Conditions: V_{CC} = 3.3 V ±0.3 V, V_{SS} = 0 V, Ta = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Test Condition
Command write cycle	t_{nxtc}	20	—	µs	Figure 6.13
$\overline{CE}$ hold time	t_{ceh}	0	—	ns	
$\overline{CE}$ setup time	t_{ces}	0	—	ns	
Data hold time	t_{dh}	50	—	ns	
Data setup time	t_{ds}	50	—	ns	
Write pulse width	t_{wep}	70	—	ns	
Status polling start time	t_{ests}	1	—	ms	
Status polling access time	t_{spa}	—	150	ns	
Memory erase time	t_{erase}	100	40000	ms	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

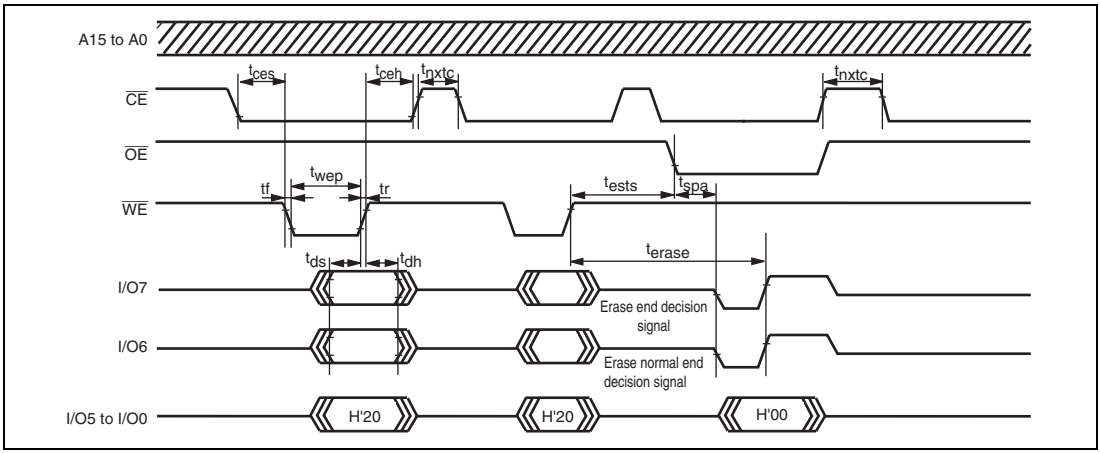


Figure 6.13 Timing Waveforms in Auto-Erase Mode

6.7.6 Status Read Mode

1. Status read mode is provided to identify the kind of abnormal end. Use this mode when an abnormal end occurs in auto-program mode or auto-erase mode.
2. The return code is retained until a command write other than command write in status read mode is executed.
3. Table 6.13 shows the AC characteristics and table 6.14 shows the return codes.

Table 6.13 AC Characteristics in Status Read Mode

(Conditions: V_{CC} = 3.3 V ±0.3 V, V_{SS} = 0 V, Ta = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Test Condition
Read time after command write	t_{nxtc}	20	—	µs	Figure 6.14
$\overline{CE}$ hold time	t_{ceh}	0	—	ns	
$\overline{CE}$ setup time	t_{ces}	0	—	ns	
Data hold time	t_{dh}	50	—	ns	
Data setup time	t_{ds}	50	—	ns	
Write pulse width	t_{wep}	70	—	ns	
$\overline{OE}$ output delay time	t_{oe}	—	150	ns	
Disable delay time	t_{df}	—	100	ns	
$\overline{CE}$ output delay time	t_{ce}	—	150	ns	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

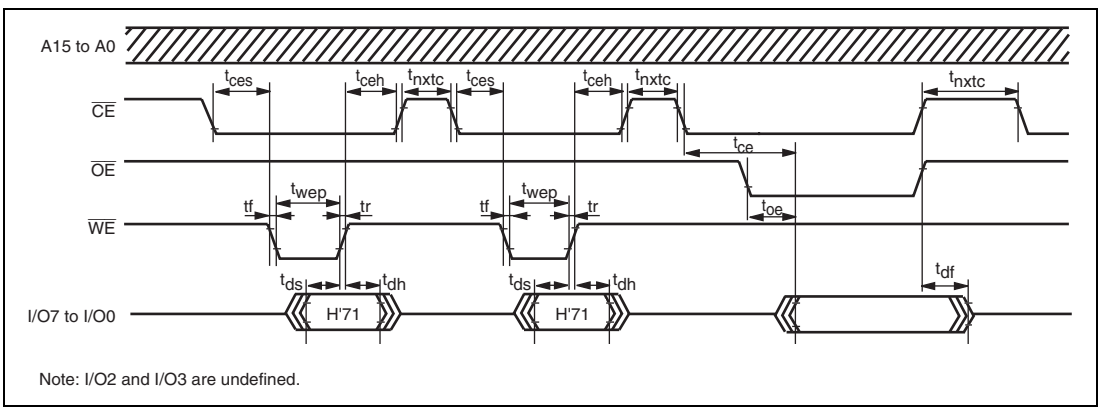


Figure 6.14 Timing Waveforms in Status Read Mode

Table 6.14 Return Codes in Status Read Mode

Pin Name	Initial Value	Description
I/O7	0	1: Abnormal end 0: Normal end
I/O6	0	1: Command error 0: Otherwise
I/O5	0	1: Programming error 0: Otherwise
I/O4	0	1: Erasing error 0: Otherwise
I/O3	0	Undefined
I/O2	0	Undefined
I/O1	0	1: Over counting of writing or erasing 0: Otherwise
I/O0	0	1: Effective address error 0: Otherwise

6.7.7 Status Polling

1. The I/O7 status polling flag indicates the operating status in auto-program/auto-erase mode.
2. The I/O6 status polling flag indicates a normal or abnormal end in auto-program/auto-erase mode.

Table 6.15 Status Polling Output

I/O7	I/O6	I/O0 to I/O5	Status
0	0	0	During internal operation
1	0	0	Abnormal end
1	1	0	Normal end
0	1	0	—

6.7.8 Programmer Mode Transition Time

Commands cannot be accepted during the oscillation stabilization period or the programmer mode setup period. After the programmer mode setup time, a transition is made to memory read mode.

Table 6.16 Stipulated Transition Times to Command Wait State

Item	Symbol	Min	Max	Unit	Test Condition
Oscillation stabilization time (crystal resonator)	t_{osc1}	10	—	ms	Figure 6.15
Oscillation stabilization time (ceramic resonator)		5	—	ms	
Programmer mode setup time	t_{bmv}	10	—	ms	
V_{CC} hold time	t_{dwn}	0	—	ms	

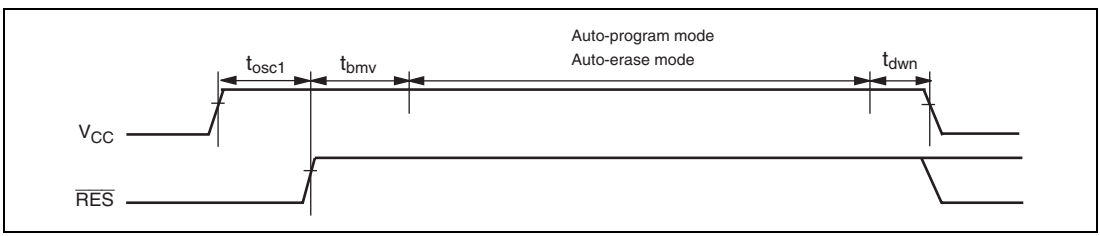


Figure 6.15 Oscillation Stabilization Time, Boot Program Transfer Time, and Power-Down Sequence

6.7.9 Notes on Memory Programming

1. When performing programming using programmer mode on a chip that has been programmed/erased in on-board programming mode, auto-erasing is recommended before carrying out auto-programming.
2. The flash memory is initially in the erased state when the device is shipped by Renesas. For other chips for which the erasure history is unknown, it is recommended that auto-erasing be executed to check and supplement the initialization (erase) level.

6.8 Power-Down States for Flash Memory

In user mode, the flash memory will operate in either of the following states:

- Normal operating mode
  The flash memory can be read and written to at high speed.
- Power-down operating mode
  The power supply circuit of flash memory can be partly halted. As a result, flash memory can be read with low power consumption.
- Standby mode
  All flash memory circuits are halted.

Table 6.17 shows the correspondence between the operating modes of this LSI and the flash memory. In subactive mode, the flash memory can be set to operate in power-down mode with the PDWND bit in FLPWCR. When the flash memory returns to its normal operating state from power-down mode or standby mode, a period to stabilize operation of the power supply circuits that were stopped is needed. When the flash memory returns to its normal operating state, bits STS2 to STS0 in SYSCR1 must be set to provide a wait time of at least 20 μs, even when the external clock is being used.

Table 6.17 Flash Memory Operating States

	Flash Memory Operating State	
LSI Operating State	**PDWND = 0 (Initial value)**	**PDWND = 1**
Active mode	Normal operating mode	Normal operating mode
Subactive mode	Power-down mode	Normal operating mode
Sleep mode	Normal operating mode	Normal operating mode
Subsleep mode	Standby mode	Standby mode
Standby mode	Standby mode	Standby mode
Watch mode	Standby mode	Standby mode

Section 7 RAM

This LSI has an on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling two-state access by the CPU to both byte data and word data.

Product Classification		RAM Size	RAM Address
Flash memory version	H8/38704F	1 kbyte	H'FB80 to H'FF7F
	H8/38702F	1 kbyte	H'FB80 to H'FF7F
Mask ROM version	H8/38704	1 kbyte	H'FB80 to H'FF7F
	H8/38703	1 kbyte	H'FB80 to H'FF7F
	H8/38702	1 kbyte	H'FB80 to H'FF7F
	H8/38702S	512 bytes	H'FD80 to H'FF7F
	H8/38701S	512 bytes	H'FD80 to H'FF7F
	H8/38700S	512 bytes	H'FD80 to H'FF7F

7.1 Block Diagram

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

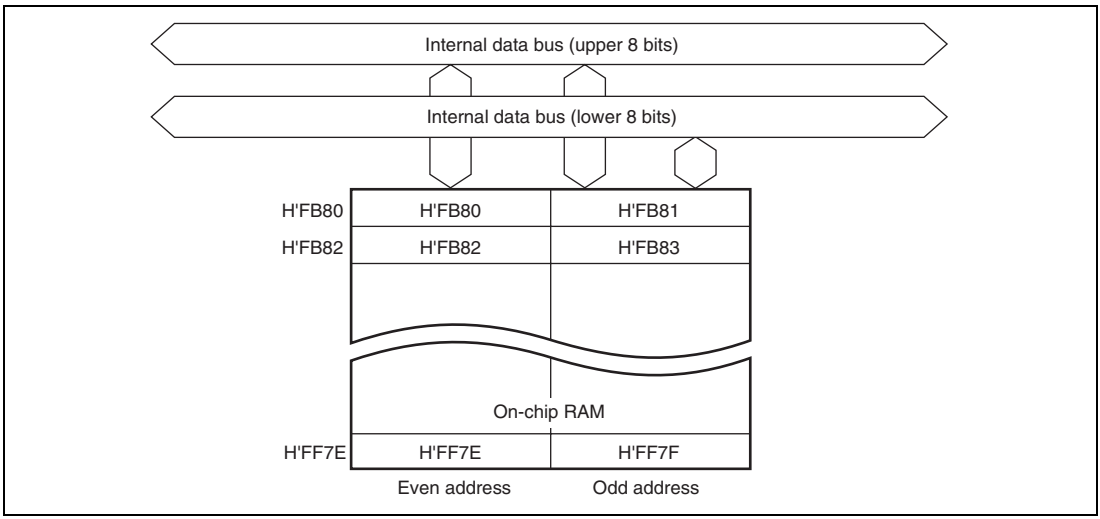


Figure 7.1 Block Diagram of RAM

Section 8 I/O Ports

This LSI is provided with three 8-bit I/O ports, one 7-bit I/O port, one 4-bit I/O port, one 3-bit I/O port, one 1-bit I/O port, one 4-bit input-only port, one 1-bit input-only port, and one 6-bit output-only port.

Each port is configured by the port control register (PCR) that controls input and output, and the port data register (PDR) that stores output data. Input or output can be assigned to individual bits.

See section 2.8.3, Bit-Manipulation Instructions, for information on executing bit-manipulation instructions to write data in PCR or PDR. Block diagrams of each port are given in appendix B, I/O Port Block Diagrams. Table 8.1 lists the functions of each port.

Table 8.1 Port Functions

Port	Description	Pins	Other Functions	Function Switching Registers
Port 3	• 7-bit I/O port • Input pull-up MOS option	P37/AEVL P36/AEVH P35 P34 P33	Asynchronous event counter event inputs AEVL, AEVH	PMR3
		P32/TMOFH P31/TMOFL	Timer F output compare output	PMR3
Port 4	• 1-bit input-only port	P43/$\overline{\text{IRQ0}}$	External interrupt 0	PMR2
	• 3-bit I/O port	P42/TXD32 P41/RXD32 P40/SCK32	SCI3 data output (TXD32), data input (RXD32), clock input/output (SCK32)	SCR3 SMR
Port 5	• 8-bit I/O port • Input pull-up MOS option	P57 to P50/ $\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$	Wakeup input ($\overline{\text{WKP7}}$ to $\overline{\text{WKP0}}$)	PMR5
Port 6	• 8-bit I/O port • Input pull-up MOS option	P67 to P60	None	—
Port 7	• 8-bit I/O port	P77 to P70	None	—
Port 8	• 1-bit I/O port	P80	None	—
Port 9	• 6-bit output-only port	P95 to P92	None	—
		P91, P90/ PWM2, PWM1	10-bit PWM output	PMR9
Port A	• 4-bit I/O port	PA3 to PA0	None	—
Port B	• 4-bit input-only port	PB3/AN3/ $\overline{\text{IRQ1}}$	A/D converter analog input External interrupt 1	AMR PMRB
		PB2/AN2	A/D converter analog input	AMR
		PB1/AN1 PB0/AN0	A/D converter analog input	AMR

8.1 Port 3

Port 3 is an I/O port also functioning as an asynchronous event counter input pin and timer F output pin. Figure 8.1 shows its pin configuration.

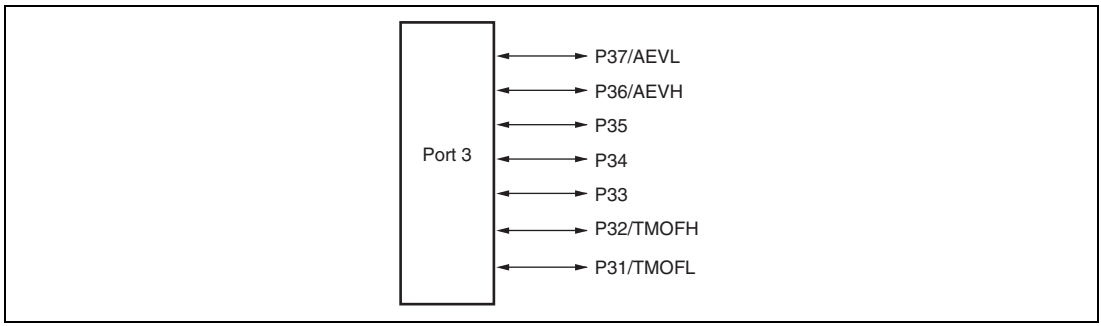


Figure 8.1 Port 3 Pin Configuration

Port 3 has the following registers.

- Port data register 3 (PDR3)
- Port control register 3 (PCR3)
- Port pull-up control register 3 (PUCR3)
- Port mode register 3 (PMR3)
- Port mode register 2 (PMR2)

8.1.1 Port Data Register 3 (PDR3)

PDR3 is a register that stores data of port 3.

Bit	Bit Name	Initial Value	R/W	Description
7	P37	0	R/W	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.
6	P36	0	R/W	
5	P35	0	R/W	
4	P34	0	R/W	
3	P33	0	R/W	
2	P32	0	R/W	
1	P31	0	R/W	
0	—	—	—	Reserved

8.1.2 Port Control Register 3 (PCR3)

PCR3 controls whether each of the port 3 pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7	PCR37	0	W	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.
6	PCR36	0	W	
5	PCR35	0	W	
4	PCR34	0	W	
3	PCR33	0	W	PCR3 is a write-only register. Bits 7 to 1 are always read as 1.
2	PCR32	0	W	
1	PCR31	0	W	
0	—	—	W	Reserved The write value should always be 0.

8.1.3 Port Pull-Up Control Register 3 (PUCR3)

PUCR3 controls whether the pull-up MOS of each of the port 3 pins is on or off.

Bit	Bit Name	Initial Value	R/W	Description
7	PUCR37	0	R/W	When a PCR3 bit is cleared to 0, setting the corresponding PUCR3 bit to 1 turns on the pull-up MOS for the corresponding pin, while clearing the bit to 0 turns off the pull-up MOS.
6	PUCR36	0	R/W	
5	PUCR35	0	R/W	
4	PUCR34	0	R/W	
3	PUCR33	0	R/W	
2	PUCR32	0	R/W	
1	PUCR31	0	R/W	
0	—	—	W	Reserved The write value should always be 0.

8.1.4 Port Mode Register 3 (PMR3)

PMR3 controls the selection of pin functions for port 3 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	AEVL	0	R/W	P37/AEVL Pin Function Switch This bit selects whether pin P37/AEVL is used as P37 or as AEVL. 0: P37 I/O pin 1: AEVL input pin
6	AEVH	0	R/W	P36/AEVH Pin Function Switch This bit selects whether pin P36/AEVH is used as P36 or as AEVH. 0: P36 I/O pin 1: AEVH input pin
5 to 3	—	—	W	Reserved The write value should always be 0.
2	TMOFH	0	R/W	P32/TMOFH Pin Function Switch This bit selects whether pin P32/TMOFH is used as P32 or as TMOFH. 0: P32 I/O pin 1: TMOFH output pin
1	TMOFL	0	R/W	P31/TMOFL Pin Function Switch This bit selects whether pin P31/TMOFL is used as P31 or as TMOFL. 0: P31 I/O pin 1: TMOFL output pin
0	—	—	W	Reserved The write value should always be 0.

8.1.5 Port Mode Register 2 (PMR2)

PMR2 controls the PMOS on/off state for the P35 pin, selects a pin function for the P43/$\overline{\text{IRQ0}}$ pin, and selects a clock of the watchdog timer.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
5	POF1	0	R/W	P35 Pin PMOS Control This bit controls the on/off state of the PMOS of the P35 pin output buffer. 0: CMOS output 1: NMOS open-drain output
4, 3	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
2	WDCKS	0	R/W	Watchdog Timer Source Clock Select This bit selects the input clock for the watchdog timer. 0: ϕ/8,192 1: ϕw/32
1	—	—	W	Reserved The write value should always be 0.
0	IRQ0	0	R/W	P43/$\overline{\text{IRQ0}}$ Pin Function Switch This bit selects whether pin P43/$\overline{\text{IRQ0}}$ is used as P43 or as $\overline{\text{IRQ0}}$. 0: P43 input pin 1: $\overline{\text{IRQ0}}$ input pin

8.1.6 Pin Functions

The port 3 pin functions are shown below.

- P37/AEVL pin

The pin function depends on the combination of bit AEVL in PMR3 and bit PCR37 in PCR3.

AEVL	0		1
PCR37	0	1	x
Pin Function	P37 input pin	P37 output pin	AEVL input pin

[Legend]
x: Don't care.

- P36/AEVH pin

The pin function depends on the combination of bit AEVH in PMR3 and bit PCR36 in PCR3.

AEVH	0		1
PCR36	0	1	x
Pin Function	P36 input pin	P36 output pin	AEVH input pin

[Legend]
x: Don't care.

- P35 to P33 pins

The pin function depends on the corresponding bit in PCR3.

(n = 5 to 3)

PCR3n	0	1
Pin Function	P3n input pin	P3n output pin

- P32/TMOFH pin

The pin function depends on the combination of bit TMOFH in PMR3 and bit PCR32 in PCR3.

TMOFH	0		1
PCR32	0	1	x
Pin Function	P32 input pin	P32 output pin	TMOFH output pin

[Legend]
x: Don't care.

- P31/TMOFL pin

The pin function depends on the combination of bit TMOFL in PMR3 and bit PCR31 in PCR3.

TMOFL	0		1
PCR31	0	1	x
Pin Function	P31 input pin	P31 output pin	TMOFL output pin

[Legend]
x: Don't care.

8.1.7 Input Pull-Up MOS

Port 3 has an on-chip input pull-up MOS function that can be controlled by software. When the PCR3 bit is cleared to 0, setting the corresponding PUCR3 bit to 1 turns on the input pull-up MOS for that pin. The input pull-up MOS function is in the off state after a reset.

(n = 7 to 1)

PCR3n	0		1
PUCR3n	0	1	x
Input Pull-Up MOS	Off	On	Off

[Legend]
x: Don't care.

8.2 Port 4

Port 4 is an I/O port also functioning as an interrupt input pin and SCI I/O pin. Figure 8.2 shows its pin configuration.

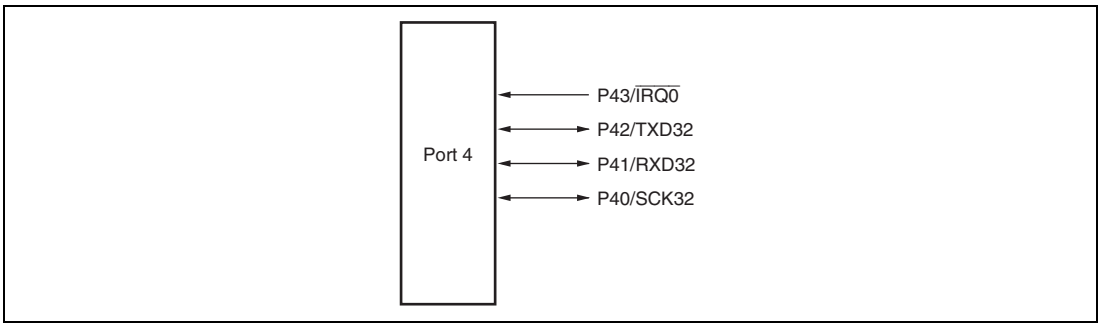


Figure 8.2 Port 4 Pin Configuration

Port 4 has the following registers.

- Port data register 4 (PDR4)
- Port control register 4 (PCR4)
- Serial port control register (SPCR)

8.2.1 Port Data Register 4 (PDR4)

PDR4 is a register that stores data of port 4.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	1	—	Reserved These bits are always read as 1.
3	P43	1	R	If port 4 is read while PCR4 bits are set to 1, the values stored in PDR4 are read, regardless of the actual pin states. If port 4 is read while PCR4 bits are cleared to 0, the pin states are read.
2	P42	0	R/W	
1	P41	0	R/W	
0	P40	0	R/W	

8.2.2 Port Control Register 4 (PCR4)

PCR4 controls whether each of the port 4 pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 1	—	Reserved These bits are always read as 1.
2	PCR42	0	W	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 and SCR2 as a general I/O pin. PCR4 is a write-only register. Bits 2 to 0 are always read as 1.
1	PCR41	0	W	
0	PCR40	0	W	

8.2.3 Serial Port Control Register (SPCR)

SPCR performs input/output data inversion switching of the RXD32 and TXD32 pins. Figure 8.3 shows the configuration.

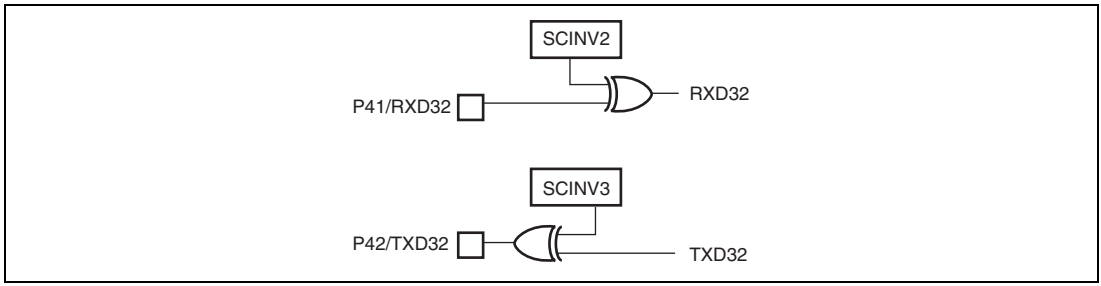


Figure 8.3 Input/Output Data Inversion Function

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
5	SPC32	0	R/W	P42/TXD32 Pin Function Switch This bit selects whether pin P42/TXD32 is used as P42 or as TXD32. 0: P42 I/O pin 1: TXD32 output pin* Note: * Set the TE bit in SCR3 after setting this bit to 1.
4	—	—	W	Reserved The write value should always be 0.
3	SCINV3	0	R/W	TXD32 Pin Output Data Inversion Switch This bit selects whether or not the logic level of the TXD32 pin output data is inverted. 0: TXD32 output data is not inverted 1: TXD32 output data is inverted
2	SCINV2	0	R/W	RXD32 Pin Input Data Inversion Switch This bit selects whether or not the logic level of the RXD32 pin input data is inverted. 0: RXD32 input data is not inverted 1: RXD32 input data is inverted
1, 0	—	—	W	Reserved The write value should always be 0.

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

8.2.4 Pin Functions

The port 4 pin functions are shown below.

- P43/$\overline{\text{IRQ0}}$ pin

The pin function depends on the IRQ0 bit in PMR2.

IRQ0	0	1
Pin Function	P43 input pin	$\overline{\text{IRQ0}}$ input pin

- P42/TXD32 pin

The pin function depends on the combination of bit TE in SCR3, bit SPC32 in SPCR, and bit PCR42 in PCR4.

SPC32	0		1
TE	0		x
PCR42	0	1	x
Pin Function	P42 input pin	P42 output pin	TXD32 output pin

[Legend]
x: Don't care.

- P41/RXD32 pin

The pin function depends on the combination of bit RE in SCR3 and bit PCR41 in PCR4.

RE	0		1
PCR41	0	1	x
Pin Function	P41 input pin	P41 output pin	RXD32 input pin

[Legend]
x: Don't care.

- P40/SCK32 pin

The pin function depends on the combination of bits CKE1 and CKE0 in SCR3, bit COM in SMR, and bit PCR40 in PCR4.

CKE1	0			1
CKE0	0		1	x
COM	0	1	x	x
PCR40	0	1	x	x
Pin Function	P40 input pin	P40 output pin	SCK32 output pin	SCK32 input pin

[Legend]
x: Don't care.

8.3 Port 5

Port 5 is an I/O port also functioning as a wakeup interrupt request input pin. Figure 8.4 shows its pin configuration.

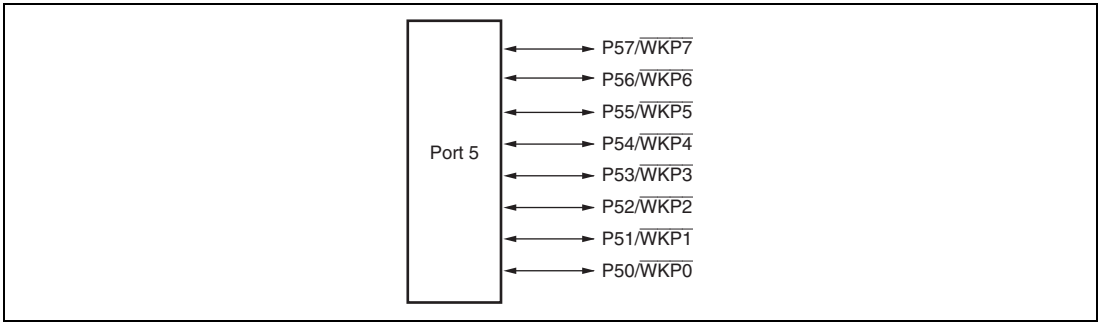


Figure 8.4 Port 5 Pin Configuration

Port 5 has the following registers.

- Port data register 5 (PDR5)
- Port control register 5 (PCR5)
- Port pull-up control register 5 (PUCR5)
- Port mode register 5 (PMR5)

8.3.1 Port Data Register 5 (PDR5)

PDR5 is a register that stores data of port 5.

Bit	Bit Name	Initial Value	R/W	Description
7	P57	0	R/W	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.
6	P56	0	R/W	
5	P55	0	R/W	
4	P54	0	R/W	
3	P53	0	R/W	
2	P52	0	R/W	
1	P51	0	R/W	
0	P50	0	R/W	

8.3.2 Port Control Register 5 (PCR5)

PCR5 controls whether each of the port 5 pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7	PCR57	0	W	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 by PMR5 and the SGS3 to SGS0 bits in LPCR as a general I/O pin.
6	PCR56	0	W	
5	PCR55	0	W	
4	PCR54	0	W	
3	PCR53	0	W	PCR5 is a write-only register. Bits 7 to 0 are always read as 1.
2	PCR52	0	W	
1	PCR51	0	W	
0	PCR50	0	W	

8.3.3 Port Pull-Up Control Register 5 (PUCR5)

PUCR5 controls whether the pull-up MOS of each of the port 5 pins is on or off.

Bit	Bit Name	Initial Value	R/W	Description
7	PUCR57	0	R/W	When a PCR5 bit is cleared to 0, setting the corresponding PUCR5 bit to 1 turns on the pull-up MOS for the corresponding pin, while clearing the bit to 0 turns off the pull-up MOS.
6	PUCR56	0	R/W	
5	PUCR55	0	R/W	
4	PUCR54	0	R/W	
3	PUCR53	0	R/W	
2	PUCR52	0	R/W	
1	PUCR51	0	R/W	
0	PUCR50	0	R/W	

8.3.4 Port Mode Register 5 (PMR5)

PMR5 controls the selection of pin functions for port 5 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	WKP7	0	R/W	P5n/$\overline{\text{WKPn}}$ Pin Function Switch
6	WKP6	0	R/W	These bits select whether pin P5n/$\overline{\text{WKPn}}$ is used as P5n or $\overline{\text{WKPn}}$. 0: P5n I/O pin 1: $\overline{\text{WKPn}}$ input pin (n = 7 to 0)
5	WKP5	0	R/W	
4	WKP4	0	R/W	
3	WKP3	0	R/W	
2	WKP2	0	R/W	
1	WKP1	0	R/W	
0	WKP0	0	R/W	

8.3.5 Pin Functions

The port 5 pin functions are shown below.

- P57/$\overline{\text{WKP7}}$ to P54/$\overline{\text{WKP4}}$ pins

The pin function depends on the combination of bit WKPn in PMR5 and bit PCR5n in PCR5.

(n = 7 to 4)

WKPn	0		1
PCR5n	0	1	x
Pin Function	P5n input pin	P5n output pin	$\overline{\text{WKPn}}$ input pin

[Legend]
x: Don't care.

- P53/$\overline{\text{WKP3}}$ to P50/$\overline{\text{WKP0}}$ pins

The pin function depends on the combination of bit WKPm in PMR5 and bit PCR5m in PCR5.

(m = 3 to 0)

WKPm	0		1
PCR5m	0	1	x
Pin Function	P5m input pin	P5m output pin	$\overline{\text{WKPm}}$ input pin

[Legend]
x: Don't care.

8.3.6 Input Pull-Up MOS

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

(n = 7 to 0)

PCR5n	0		1
PUCR5n	0	1	x
Input Pull-Up MOS	Off	On	Off

[Legend]
x: Don't care.

8.4 Port 6

Port 6 is an I/O port. Figure 8.5 shows its pin configuration.

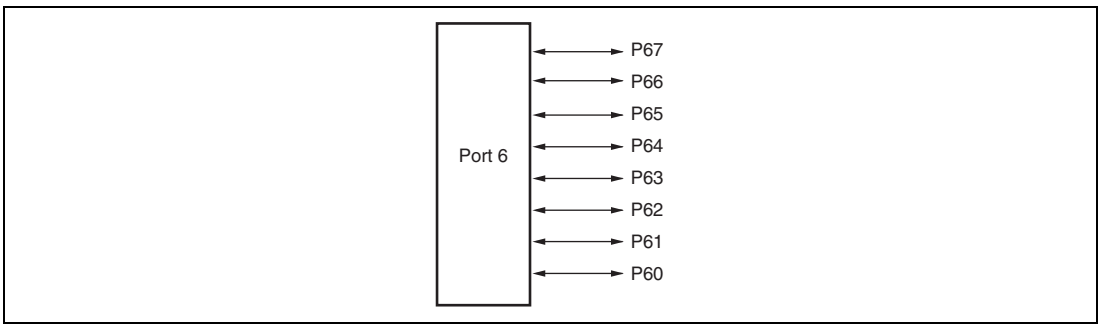


Figure 8.5 Port 6 Pin Configuration

Port 6 has the following registers.

- Port data register 6 (PDR6)
- Port control register 6 (PCR6)
- Port pull-up control register 6 (PUCR6)

8.4.1 Port Data Register 6 (PDR6)

PDR6 is a register that stores data of port 6.

Bit	Bit Name	Initial Value	R/W	Description
7	P67	0	R/W	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.
6	P66	0	R/W	
5	P65	0	R/W	
4	P64	0	R/W	
3	P63	0	R/W	
2	P62	0	R/W	
1	P61	0	R/W	
0	P60	0	R/W	

8.4.2 Port Control Register 6 (PCR6)

PCR6 controls whether each of the port 6 pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7	PCR67	0	W	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. PCR6 is a write-only register. Bits 7 to 0 are always read as 1.
6	PCR66	0	W	
5	PCR65	0	W	
4	PCR64	0	W	
3	PCR63	0	W	
2	PCR62	0	W	
1	PCR61	0	W	
0	PCR60	0	W	

8.4.3 Port Pull-Up Control Register 6 (PUCR6)

PUCR6 controls whether the pull-up MOS of each of the port 6 pins is on or off.

Bit	Bit Name	Initial Value	R/W	Description
7	PUCR67	0	R/W	When a PCR6 bit is cleared to 0, setting the corresponding PUCR6 bit to 1 turns on the pull-up MOS for the corresponding pin, while clearing the bit to 0 turns off the pull-up MOS.
6	PUCR66	0	R/W	
5	PUCR65	0	R/W	
4	PUCR64	0	R/W	
3	PUCR63	0	R/W	
2	PUCR62	0	R/W	
1	PUCR61	0	R/W	
0	PUCR60	0	R/W	

8.4.4 Pin Functions

The port 6 pin functions are shown below.

- P67 to P64 pins

The pin function depends on the setting of bit PCR6n in PCR6.

(n = 7 to 4)

PCR6n	0	1
Pin Function	P6n input pin	P6n output pin

- P63 to P60 pins

The pin function depends on the setting of bit PCR6m in PCR6.

(m = 3 to 0)

PCR6m	0	1
Pin Function	P6m input pin	P6m output pin

8.4.5 Input Pull-Up MOS

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

(n = 7 to 0)

PCR6n	0		1
PUCR6n	0	1	x
Input Pull-Up MOS	Off	On	Off

[Legend]
x: Don't care.

8.5 Port 7

Port 7 is an I/O port. Figure 8.6 shows its pin configuration.

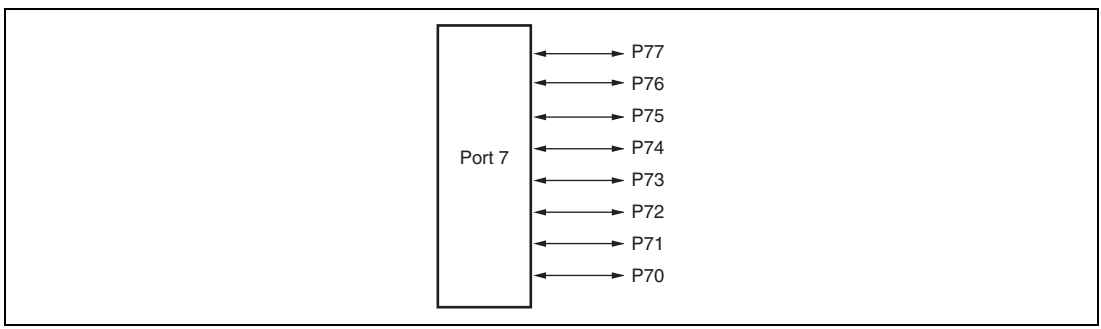


Figure 8.6 Port 7 Pin Configuration

Port 7 has the following registers.

- Port data register 7 (PDR7)
- Port control register 7 (PCR7)

8.5.1 Port Data Register 7 (PDR7)

PDR7 is a register that stores data of port 7.

Bit	Bit Name	Initial Value	R/W	Description
7	P77	0	R/W	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.
6	P76	0	R/W	
5	P75	0	R/W	
4	P74	0	R/W	
3	P73	0	R/W	
2	P72	0	R/W	
1	P71	0	R/W	
0	P70	0	R/W	

8.5.2 Port Control Register 7 (PCR7)

PCR7 controls whether each of the port 7 pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7	PCR77	0	W	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. The settings in PCR7 and in PDR7 are valid only when the corresponding pin is designated by the SGS3 to SGS0 bits in LPCR as a general I/O pin.
6	PCR76	0	W	
5	PCR75	0	W	
4	PCR74	0	W	
3	PCR73	0	W	PCR7 is a write-only register. Bits 7 to 0 are always read as 1.
2	PCR72	0	W	
1	PCR71	0	W	
0	PCR70	0	W	

8.5.3 Pin Functions

The port 7 pin functions are shown below.

- P77 to P74 pins

The pin function depends on the setting of bit PCR7n in PCR7.

(n = 7 to 4)

PCR7n	0	1
Pin Function	P7n input pin	P7n output pin

- P73 to P70 pins

The pin function depends on the setting of bit PCR7m in PCR7.

(m = 3 to 0)

PCR7m	0	1
Pin Function	P7m input pin	P7m output pin

8.6 Port 8

Port 8 is an I/O port. Figure 8.7 shows its pin configuration.

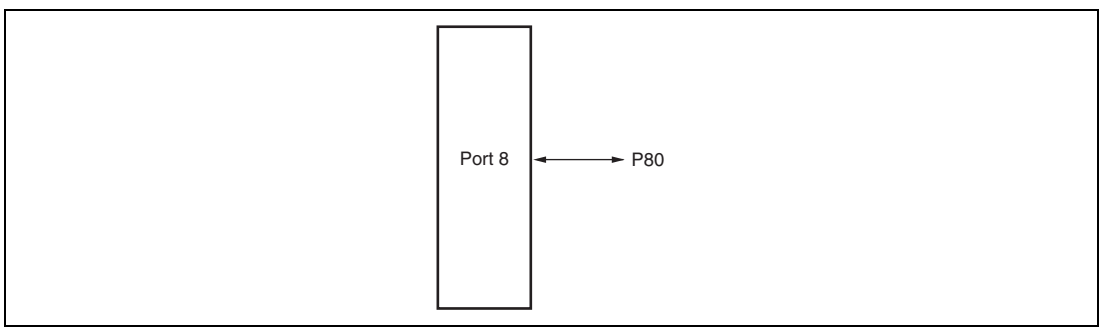


Figure 8.7 Port 8 Pin Configuration

Port 8 has the following registers.

- Port data register 8 (PDR8)
- Port control register 8 (PCR8)

8.6.1 Port Data Register 8 (PDR8)

PDR8 is a register that stores data of port 8.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	—	—	Reserved
0	P80	0	R/W	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.

8.6.2 Port Control Register 8 (PCR8)

PCR8 controls whether each of the port 8 pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	—	W	Reserved The write value should always be 0.
0	PCR80	0	W	Setting a PCR8 bit to 1 makes the corresponding pin an output pin, while clearing the bit to 0 makes the pin an input pin. PCR8 is a write-only register.

8.6.3 Pin Functions

The port 8 pin functions are shown below.

- P80

The pin function depends on the setting of bit PCR80 in PCR8.

PCR80	0	1
Pin Function	P80 input pin	P80 output pin

8.7 Port 9

Port 9 is a dedicated current port for NMOS output that also functions as a PWM output pin. Figure 8.8 shows its pin configuration.

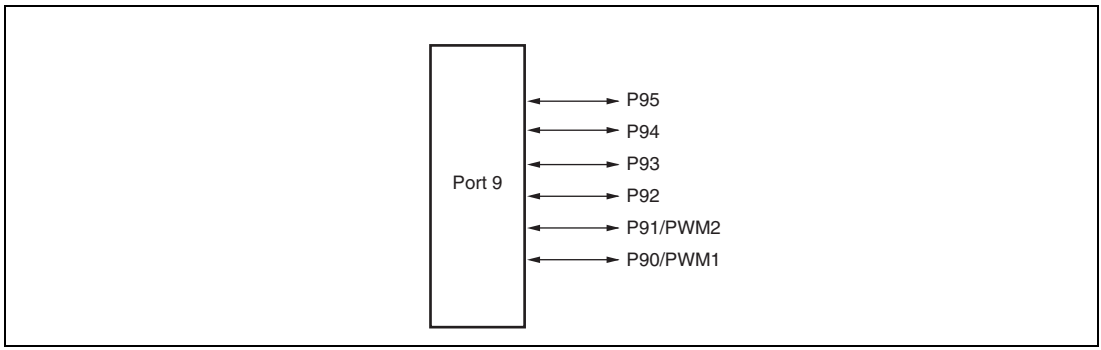


Figure 8.8 Port 9 Pin Configuration

Port 9 has the following registers.

- Port data register 9 (PDR9)
- Port mode register 9 (PMR9)

8.7.1 Port Data Register 9 (PDR9)

PDR9 is a register that stores data of port 9.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved The initial value should not be changed.
5	P95	1	R/W	If PDR9 is read, the values stored in PDR9 are read.
4	P94	1	R/W	
3	P93	1	R/W	
2	P92	1	R/W	
1	P91	1	R/W	
0	P90	1	R/W	

8.7.2 Port Mode Register 9 (PMR9)

PMR9 controls the selection of the P90 and P91 pin functions.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved The initial value should not be changed.
3	—	0	R/W	Reserved This bit can be read from or written to.
2	—	—	W	Reserved The write value should always be 0.
1	PWM2	0	R/W	P9n/PWMn+1 Pin Function Switch
0	PWM1	0	R/W	These bits select whether pin P9n/PWMn+1 is used as P9n or as PWMn+1. (n = 1, 0) 0: P9n output pin 1: PWMn+1 output pin

8.7.3 Pin Functions

The port 9 pin functions are shown below.

- P91/PWMn+1 to P90/PWMn+1 pins

(n = 1, 0)

PMR9n	0	1
Pin Function	P9n output pin	PWMn+1 output pin

8.8 Port A

Port A is an I/O port. Figure 8.9 shows its pin configuration.

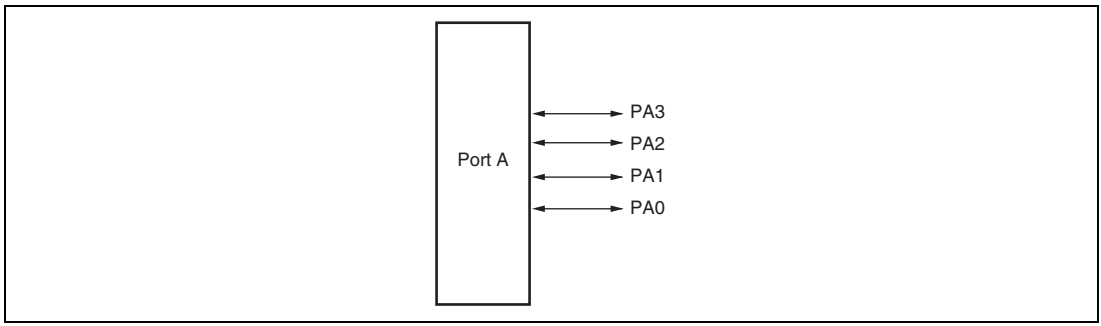


Figure 8.9 Port A Pin Configuration

Port A has the following registers.

- Port data register A (PDRA)
- Port control register A (PCRA)

8.8.1 Port Data Register A (PDRA)

PDRA is a register that stores data of port A.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved The initial value should not be changed.
3	PA3	0	R/W	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.
2	PA2	0	R/W	
1	PA1	0	R/W	
0	PA0	0	R/W	

8.8.2 Port Control Register A (PCRA)

PCRA controls whether each of the port A pins functions as an input pin or output pin.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved The initial value should not be changed.
3	PCRA3	0	W	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. The settings in PCRA and in PDRA are valid only when the corresponding pin is designated in LPCR as a general I/O pin. PCRA is a write-only register. Bits 3 to 0 are always read as 1.
2	PCRA2	0	W	
1	PCRA1	0	W	
0	PCRA0	0	W	

8.8.3 Pin Functions

The port A pin functions are shown below.

- PA3 pin

The pin function depends on the setting of bit PCRA3 in PCRA.

PCRA3	0	1
Pin Function	PA3 input pin	PA3 output pin

- PA2 pin

The pin function depends on the setting of bit PCRA2 in PCRA.

PCRA2	0	1
Pin Function	PA2 input pin	PA2 output pin

- PA1 pin

The pin function depends on the setting of bit PCRA1 in PCRA.

PCRA1	0	1
Pin Function	PA1 input pin	PA1 output pin

- PA0 pin

The pin function depends on the setting of bit PCRA0 in PCRA.

PCRA0	0	1
Pin Function	PA0 input pin	PA0 output pin

8.9 Port B

Port B is an input-only port also functioning as an analog input pin and interrupt input pin. Figure 8.10 shows its pin configuration.

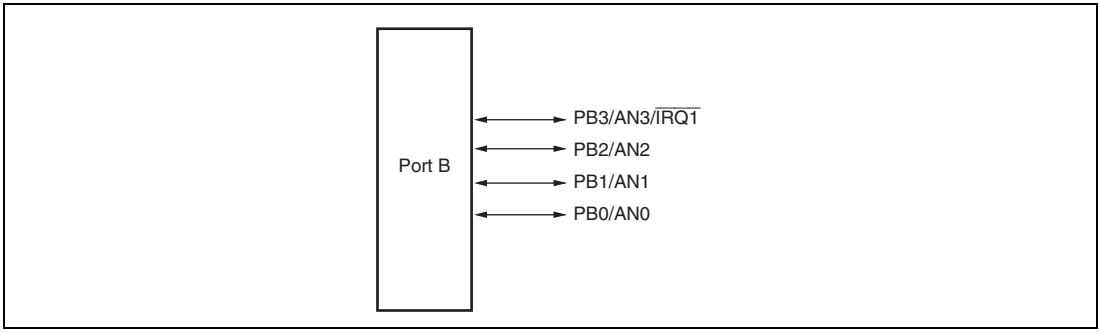


Figure 8.10 Port B Pin Configuration

Port B has the following registers.

- Port data register B (PDRB)
- Port mode register B (PMRB)

8.9.1 Port Data Register B (PDRB)

PDRB is a register that stores data of port B.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	Undefined	—	Reserved
3	PB3	Undefined	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 bits CH3 to CH0 in AMR, that pin reads 0 regardless of the input voltage.
2	PB2		R	
1	PB1		R	
0	PB0		R	

8.9.2 Port Mode Register B (PMRB)

PMRB controls the selection of the PB3 pin functions.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
3	IRQ1	0	R/W	PB3/AN3/$\overline{\text{IRQ1}}$ Pin Function Switch This bit selects whether pin PB3/AN3/$\overline{\text{IRQ1}}$ is used as PB3/AN3 or as $\overline{\text{IRQ1}}$. 0: PB3/AN3 input pin 1: $\overline{\text{IRQ1}}$ input pin
2 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.

8.9.3 Pin Functions

The port B pin functions are shown below.

- PB3/AN3/$\overline{\text{IRQ1}}$ pin

The pin function depends on the combination of bits CH3 to CH0 in AMR and bit IRQ1 in PMRB.

IRQ1	0		1
CH3 to CH0	Other than B'0111	B'0111	x
Pin Function	PB3 input pin	AN3 input pin	$\overline{\text{IRQ1}}$ input pin

[Legend]
x: Don't care.

- PB2/AN2 pin

The pin function depends on bits CH3 to CH0 in AMR.

CH3 to CH0	Other than B'0110	B'0110
Pin Function	PB2 input pin	AN2 input pin

- PB1/AN1 pin

Switching is accomplished by combining CH3 to CH0 in AMR as shown below.

CH3 to CH0	Other than B'0101	B'0101
Pin Function	PB1 input pin	AN1 input pin

- PB0/AN0 pin

Switching is accomplished by combining CH3 to CH0 in AMR as shown below.

CH3 to CH0	Other than B'0100	B'0100
Pin Function	PB0 input pin	AN0 input pin

8.10 Usage Notes

8.10.1 How to Handle Unused Pin

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

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

Section 9 Timers

9.1 Overview

This LSI has four timers: timer A, timer F, asynchronous event counter, and watchdog timer.

The functions of these timers are summarized in table 9.1.

Table 9.1 Timer Functions

Name	Functions	Internal Clock	Event Input Pin	Waveform Output Pin	Remarks
Timer A	• 8-bit timer • Interval function • Clock time base	ϕ/8 to ϕ/8192 (8 choices) ϕ_w/128 (choice of 4 overflow periods)	—	—	
Timer F	• 16-bit timer • Also usable as two independent 8-bit timers. • Output compare output function	ϕ/4 to ϕ/32, ϕ_w/4 (4 choices)	—	TMOFL TMOFH	
Asynchro-nous event counter (AEC)	• 16-bit counter • Also usable as two independent 8-bit counters • Counts events asynchronous to ϕ and ϕ_w • Can count asynchronous events (rising/falling/both edges) independ-ently of the MCU's internal clock	ϕ/2 to ϕ/8 (3 choices)	AEVL AEVH IRQAEC	—	
Watchdog timer	• Generates a reset signal by overflow of 8-bit counter	ϕ/8192, ϕ_w/32	—	—	

9.2 Timer A

The timer A is an 8-bit timer with interval timing and realtime clock time-base functions. The clock time-base function is available when a 32.768kHz crystal oscillator is connected. Figure 9.1 shows a block diagram of the timer A.

9.2.1 Features

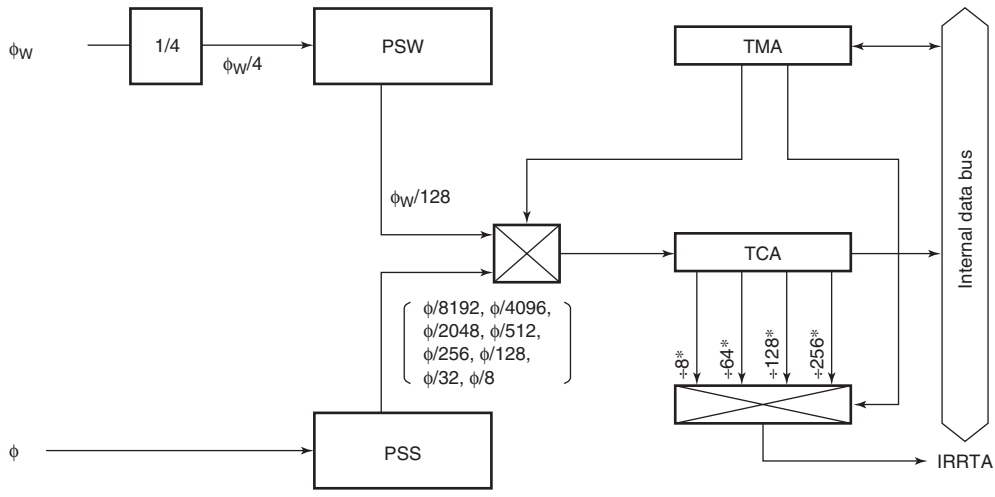
- The timer A can be used as an interval timer or a clock time base.
- An interrupt is requested when the counter overflows.
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (For details, refer to section 5.4, Module Standby Function.)

(1) Interval Timer

Choice of eight internal clock sources (ϕ/8192, ϕ/4096, ϕ/2048, ϕ/512, ϕ/256, ϕ/128, ϕ/32, and ϕ8)

(2) Clock Time Base

Choice of four overflow periods (1 s, 0.5 s, 0.25 s, and 31.25 ms) when timer A is used as a clock time base (using a 32.768 kHz crystal oscillator)



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

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

Figure 9.1 Block Diagram of Timer A

9.2.2 Register Descriptions

The timer A has the following registers.

- Timer mode register A (TMA)
- Timer counter A (TCA)

(1) Timer Mode Register A (TMA)

TMA selects the operating mode, the divided clock output, and the input clock.

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	W	Reserved
6	—	—	W	The write value should always be 0.
5	—	—	W	
4	—	1	—	Reserved This bit is always read as 1.
3	TMA3	0	R/W	Internal Clock Select 3 Selects the operating mode of the timer A. 0: Functions as an interval timer to count the outputs of prescaler S. 1: Functions as a clock-time base to count the outputs of prescaler W.
2	TMA2	0	R/W	Internal Clock Select 2 to 0
1	TMA1	0	R/W	Select the clock input to TCA when TMA3 = 0.
0	TMA0	0	R/W	000: ϕ/8192 001: ϕ/4096 010: ϕ/2048 011: ϕ/512 100: ϕ/256 101: ϕ/128 110: ϕ/32 111: ϕ/8 These bits select the overflow period when TMA3 = 1 (when a 32.768 kHz crystal oscillator is used as ϕw). 000: 1 s 001: 0.5 s 010: 0.25 s 011: 0.03125 s 1xx: Both PSW and TCA are reset

[Legend] x: Don't care.

(2) Timer Counter A (TCA)

TCA is an 8-bit readable 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 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 the interrupt request register 1 (IRR1) is set to 1. TCA is cleared by setting bits TMA3 and TMA2 in TMA to B'11. TCA is initialized to H'00.

9.2.3 Operation

(1) Interval Timer Operation

When bit TMA3 in TMA is cleared to 0, the 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 of the timer A resume immediately as an interval timer. 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 Flag Register 1 (IRR1). If IENTA = 1 in the 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 the timer A functions as an interval timer that generates an overflow output at intervals of 256 input clock pulses.

(2) Clock Time Base Operation

When bit TMA3 in TMA is set to 1, the timer A functions as a clock-timer 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 clock time base operation (TMA3 = 1), setting bit TMA2 to 1 clears both TCA and prescaler W to H'00.

9.2.4 Timer A Operating States

Table 9.2 summarizes the timer A operating states.

Table 9.2 Timer A Operating States

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

Note: * When the 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/ϕ (s) in the count cycle.

9.3 Timer F

The timer F has a 16-bit timer having an output compare function. The timer F also provides for counter resetting, interrupt request generation, toggle output, etc., using compare match signals. Thus, it can be applied to various systems. The timer F can also be used as two independent 8-bit timers (timer FH and timer FL). Figure 9.2 shows a block diagram of the timer F.

9.3.1 Features

- Choice of four internal clock sources (ϕ/32, ϕ/16, ϕ/4, and ϕ_w/4)
- Toggle output function

  Toggle output is performed to the TMOFH pin (TMOFL pin) using a single compare match signal.

  The initial value of toggle output can be set.
- Counter resetting by a compare match signal
- Two interrupt sources: One compare match, one overflow
- Choice of 16-bit or 8-bit mode by settings of bits CKSH2 to CKSH0 in TCRF
- Can operate in watch mode, subactive mode, and subsleep mode

  When ϕ_w/4 is selected as an internal clock, the timer F can operate in watch mode, subactive mode, and subsleep mode.
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (For details, refer to section 5.4, Module Standby Function.)

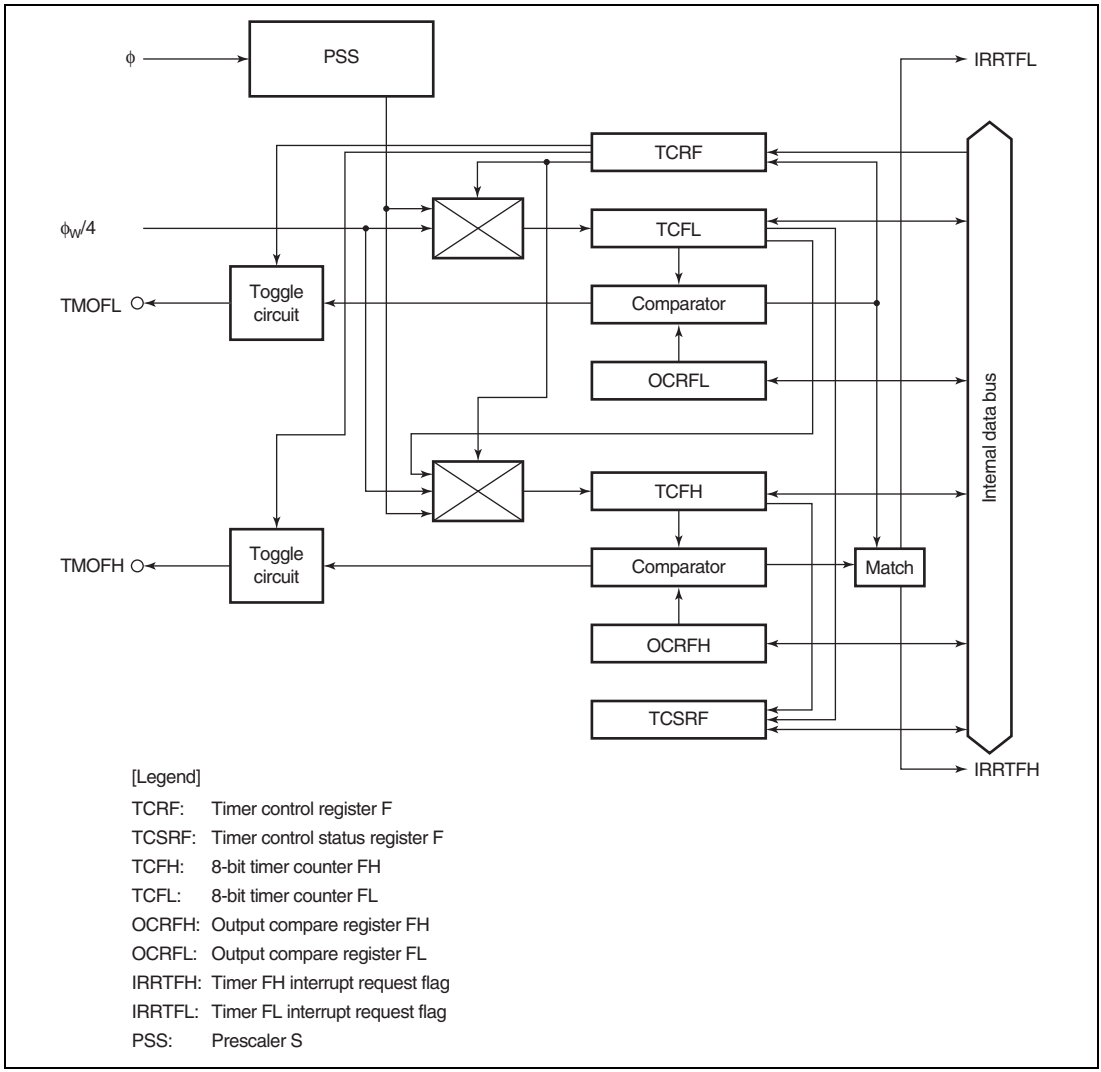


Figure 9.2 Block Diagram of Timer F

9.3.2 Input/Output Pins

Table 9.3 shows the pin configuration of the timer F.

Table 9.3 Pin Configuration

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

9.3.3 Register Descriptions

The timer F has the following registers.

- Timer counters FH and FL (TCFH,TCFL)
- Output compare registers FH and FL (OCRFH, OCRFL)
- Timer control register F (TCRF)
- Timer control status register F (TCSRF)

(1) Timer Counters FH and FL (TCFH, TCFL)

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

TCFH and TCFL can be read and written by the CPU, but when they are used in 16-bit mode, data transfer to and from the CPU is performed via a temporary register (TEMP). For details of TEMP, see section 9.3.4, CPU Interface. TCFH and TCFL are initialized to H'00 upon reset.

(a) 16-bit mode (TCF)

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

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

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

(b) 8-bit mode (TCFL/TCFH)

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

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

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

(2) Output Compare Registers FH and FL (OCRFH, OCRFL)

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

OCRFH and OCRFL can be read and written by the CPU, but when they are used in 16-bit mode, data transfer to and from the CPU is performed via a temporary register (TEMP). For details of TEMP, see section 9.3.4, CPU Interface. OCRFH and OCRFL are initialized to H'FF upon reset.

(a) 16-bit mode (OCRF)

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

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

(b) 8-bit mode (OCRFH/OCRFL)

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

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

(3) Timer Control Register F (TCRF)

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

Bit	Bit Name	Initial Value	R/W	Description
7	TOLH	0	W	Toggle Output Level H Sets the TMOFH pin output level. 0: Low level 1: High level
6	CKSH2	0	W	Clock Select H Select the clock input to TCFH from among four internal clock sources or TCFL overflow. 000: 16-bit mode, counting on TCFL overflow signal 001: 16-bit mode, counting on TCFL overflow signal 010: 16-bit mode, counting on TCFL overflow signal 011: Using prohibited 100: Internal clock: counting on ϕ/32 101: Internal clock: counting on ϕ/16 110: Internal clock: counting on ϕ/4 111: Internal clock: counting on ϕ_w/4
5	CKSH1	0	W	
4	CKSH0	0	W	
3	TOLL	0	W	Toggle Output Level L Sets the TMOFL pin output level. 0: Low level 1: High level
2	CKSL2	0	W	Clock Select L Select the clock input to TCFL from among four internal clock sources or external event input. 000: Non-operational 001: Using prohibited 010: Using prohibited 011: Using prohibited 100: Internal clock: counting on ϕ/32 101: Internal clock: counting on ϕ/16 110: Internal clock: counting on ϕ/4 111: Internal clock: counting on ϕ_w/4
1	CKSL1	0	W	
0	CKSL0	0	W	

(4) Timer Control Status Register F (TCSRF)

TCSRF performs counter clear selection, overflow flag setting, and compare match flag setting, and controls enabling of overflow interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7	OVFH	0	R/W*	Timer Overflow Flag H [Setting condition] When TCFH overflows from H'FF to H'00 [Clearing condition] When this bit is written to 0 after reading OVFH = 1
6	CMFH	0	R/W*	Compare Match Flag H This is a status flag indicating that TCFH has matched OCRFH. [Setting condition] When the TCFH value matches the OCRFH value [Clearing condition] When this bit is written to 0 after reading CMFH = 1
5	OVIEH	0	R/W	Timer Overflow Interrupt Enable H Selects enabling or disabling of interrupt generation when TCFH overflows. 0: TCFH overflow interrupt request is disabled 1: TCFH overflow interrupt request is enabled
4	CCLRH	0	R/W	Counter Clear H In 16-bit mode, this bit selects whether TCF is cleared when TCF and OCRF match. In 8-bit mode, this bit selects whether TCFH is cleared when TCFH and OCRFH match. In 16-bit mode: 0: TCF clearing by compare match is disabled 1: TCF clearing by compare match is enabled In 8-bit mode: 0: TCFH clearing by compare match is disabled 1: TCFH clearing by compare match is enabled

Bit	Bit Name	Initial Value	R/W	Description
3	OVFL	0	R/W*	Timer Overflow Flag L This is a status flag indicating that TCFL has overflowed. [Setting condition] When TCFL overflows from H'FF to H'00 [Clearing condition] When this bit is written to 0 after reading OVFL = 1
2	CMFL	0	R/W*	Compare Match Flag L This is a status flag indicating that TCFL has matched OCRFL. [Setting condition] When the TCFL value matches the OCRFL value [Clearing condition] When this bit is written to 0 after reading CMFL = 1
1	OVIEL	0	R/W	Timer Overflow Interrupt Enable L Selects enabling or disabling of interrupt generation when TCFL overflows. 0: TCFL overflow interrupt request is disabled 1: TCFL overflow interrupt request is enabled
0	CCLRL	0	R/W	Counter Clear L Selects whether TCFL is cleared when TCFL and OCRFL match. 0: TCFL clearing by compare match is disabled 1: TCFL clearing by compare match is enabled

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

9.3.4 CPU Interface

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

When performing TCF read/write access or OCRF write access in 16-bit mode, data will not be transferred correctly if only the upper byte or only the lower byte is accessed. Access must be performed for all 16 bits (using two consecutive byte-size MOV instructions), and the upper byte must be accessed before the lower byte.

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

(1) Write Access

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

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

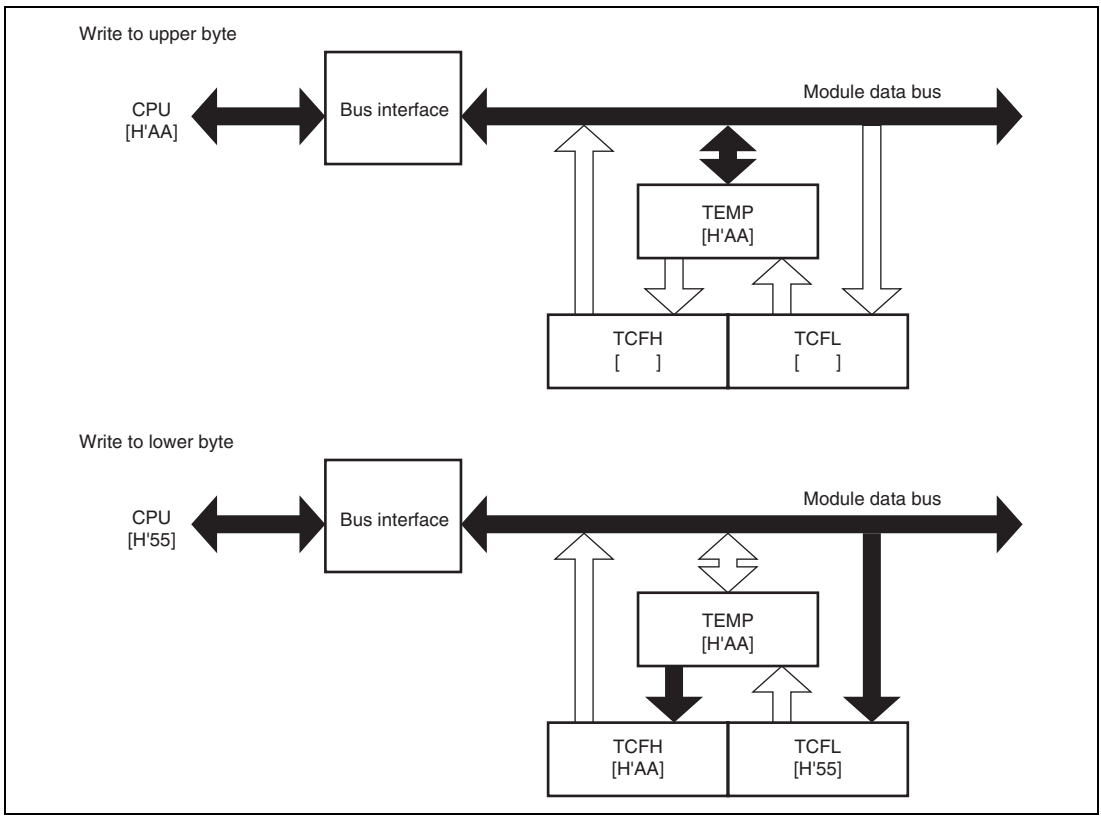


Figure 9.3 Write Access to TCF (CPU → TCF)

(2) Read Access

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

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

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

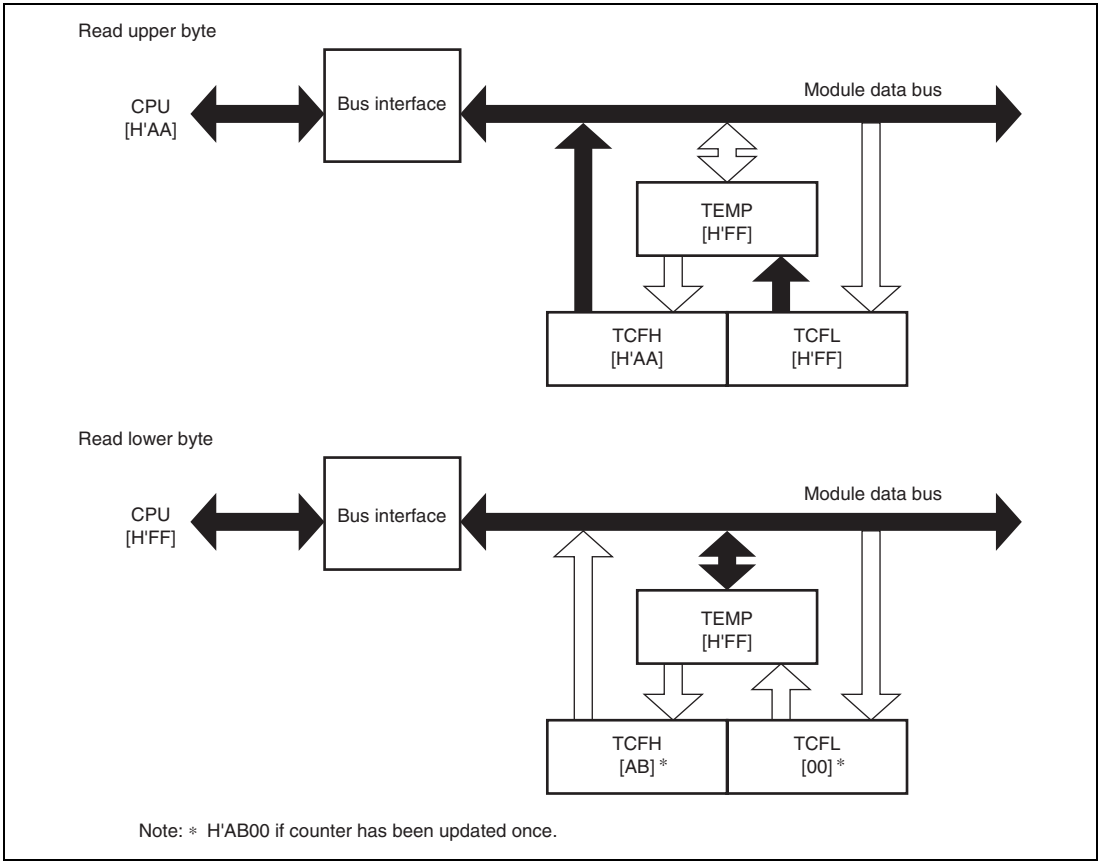


Figure 9.4 Read Access to TCF (TCF → CPU)

9.3.5 Operation

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

(1) Timer F Operation

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

(a) Operation in 16-bit timer mode

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

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

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

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

(b) Operation in 8-bit timer mode

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

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

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

(2) TCF Increment Timing

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

(3) TMOFH/TMOFL Output Timing

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

Figure 9.5 shows the output timing.

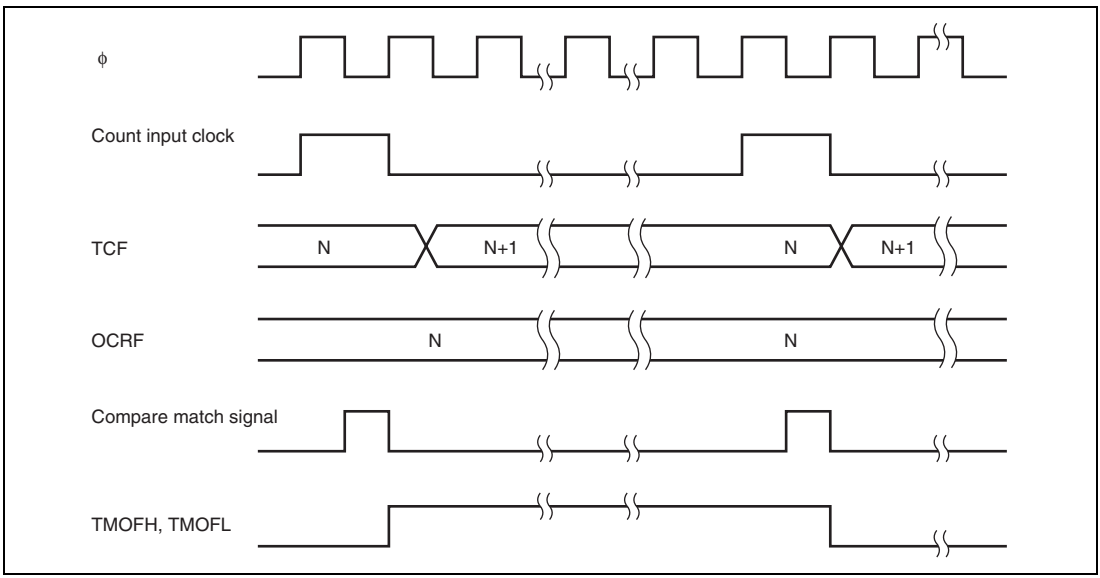


Figure 9.5 TMOFH/TMOFL Output Timing

(4) TCF Clear Timing

TCF can be cleared by a compare match with OCRF.

(5) Timer Overflow Flag (OVF) Set Timing

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

(6) Compare Match Flag Set Timing

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

9.3.6 Timer F Operating States

The timer F operating states are shown in table 9.4.

Table 9.4 Timer F Operating States

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

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

9.3.7 Usage Notes

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

(1) 16-Bit Timer Mode

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

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

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

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

(2) 8-Bit Timer Mode:

(a) TCFH, OCRFH

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

If an OCRFH write and compare match signal generation occur simultaneously, the compare match signal is invalid. However, even if the written data and the counter value match, a compare match signal is not necessarily generated at that point. The compare match signal is output in synchronization with the TCFH clock.

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

(b) TCFL, OCRFL

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

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

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

(3) Clear Timer FH, Timer FL Interrupt Request Flags (IRRTFH, IRRTFL), Timer Overflow Flags H, L (OVFH, OVFL), and Compare Match Flags H, L (CMFH, CMFL)

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

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

For interrupt request flag is set right after interrupt request is cleared, interrupt process to one time timer FH, timer FL interrupt might be repeated. (2 in figure 9.6) Therefore, to definitely clear interrupt request flag in active (high-speed, medium-speed) mode, clear should be processed after the time that calculated with below (1) formula. And, to definitely clear timer overflow flag and compare match flag, clear should be processed after read timer control status register F (TCSRF) after the time that calculated with below (1) formula.

For ST of (1) formula, please substitute the longest number of execution states in used instruction. (10 states of RTE instruction when MULXU, DIVXU instruction is not used, 14 states when MULXU, DIVXU instruction is used)

In subactive mode, there are not limitation for interrupt request flag, timer overflow flag, and compare match flag clear.

The term of validity of “Interrupt source generation signal”

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

ST: Executing number of execution states

Method 1 is recommended to operate for time efficiency.

Method 1

1. Prohibit interrupt in interrupt handling routine (set IENFH, IENFL to 0).
2. After program process returned normal handling, clear interrupt request flags (IRRTFH, IRRTFL) after more than that calculated with (1) formula.
3. After reading the timer control status register F (TCSRF), clear the timer overflow flags (OVFH, OVFL) and compare match flags (CMFH, CMFL).
4. Enable interrupts (set IENFH, IENFL to 1).

Method 2

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

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

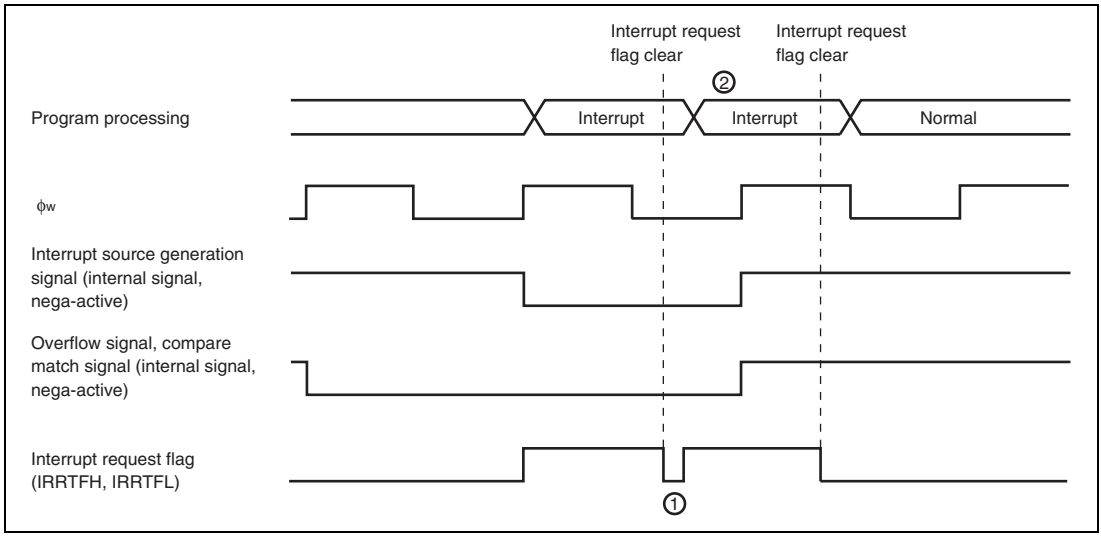


Figure 9.6 Clear Interrupt Request Flag when Interrupt Source Generation Signal is Valid

(4) Timer Counter (TCF) Read/Write

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

When reading or writing TCF in active (high-speed, medium-speed) mode is needed, please select the internal clock except for $\phi_w/4$ before read/write is performed.

In subactive mode, even if $\phi_w/4$ is selected as the internal clock, TCF can be read from or written to normally.

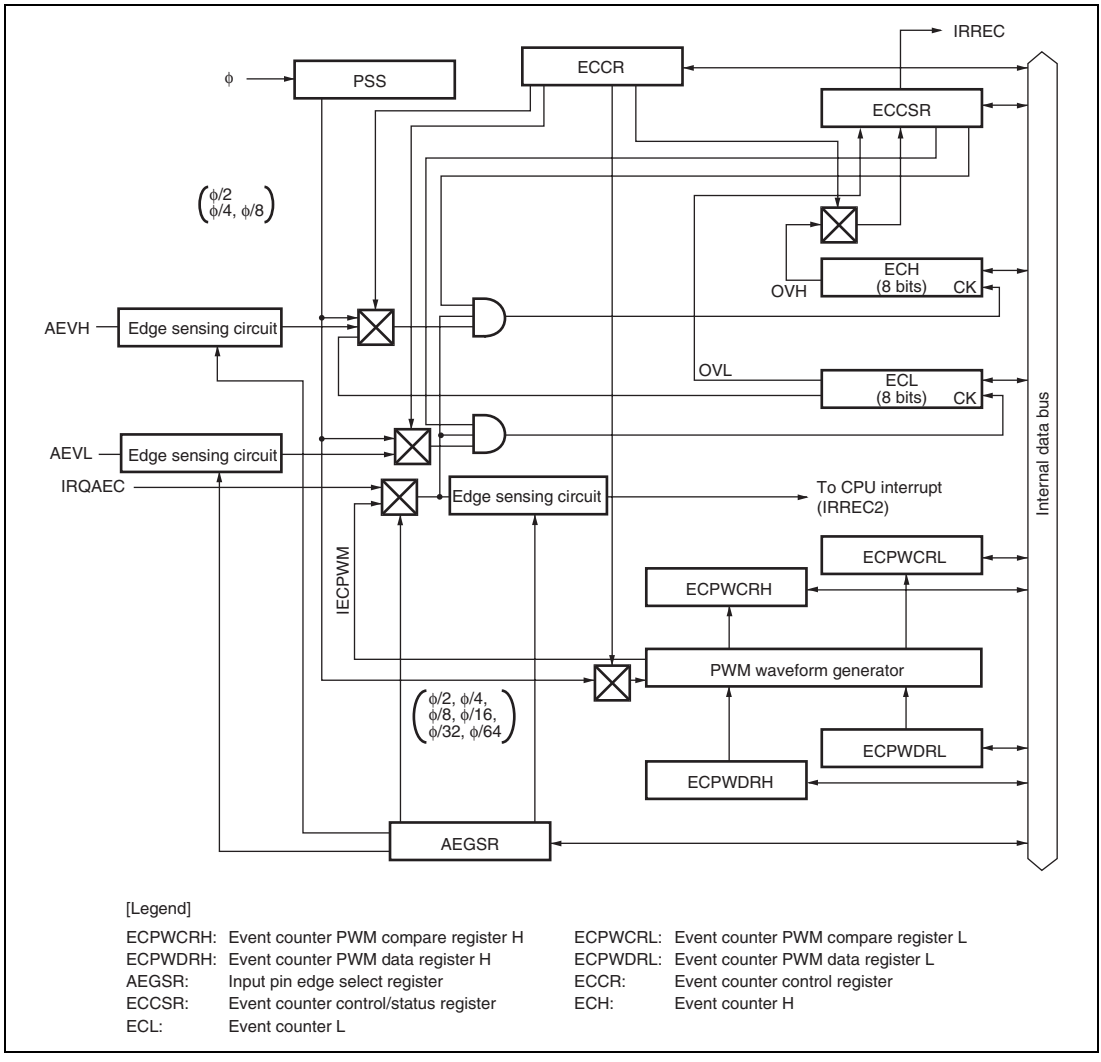
9.4 Asynchronous Event Counter (AEC)

The asynchronous event counter is incremented by external event clock or internal clock input. Figure 9.7 shows a block diagram of the asynchronous event counter.

9.4.1 Features

- Can count asynchronous events

  Can count external events input asynchronously without regard to the operation of system clocks ϕ and ϕ_{SUB}
- Can be used as two-channel independent 8-bit event counter or single-channel independent 16-bit event counter.
- Event/clock input is enabled only when IRQAEC is high or event counter PWM output (IECPWM) is high.
- Both edge sensing can be used for IRQAEC or event counter PWM output (IECPWM) interrupts. When the asynchronous counter is not used, they can be used as independent interrupts.
- When an event counter PWM is used, event clock input enabling/disabling can be controlled automatically in a fixed cycle.
- External event input or a prescaler output clock can be selected by software for the ECH and ECL clock sources. $\phi/2$, $\phi/4$, or $\phi/8$ can be selected as the prescaler output clock.
- Both edge counting is possible for AEVL and AEVH.
- Counter resetting and halting of the count-up function can be controlled by software
- Automatic interrupt generation on detection of an event counter overflow
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (For details, refer to section 5.4, Module Standby Function.)



[Legend]

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

Figure 9.7 Block Diagram of Asynchronous Event Counter

9.4.2 Input/Output Pins

Table 9.5 shows the pin configuration of the asynchronous event counter.

Table 9.5 Pin Configuration

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

9.4.3 Register Descriptions

The asynchronous event counter has the following registers.

- Event counter PWM compare register H (ECPWCRH)
- Event counter PWM compare register L (ECPWCRL)
- Event counter PWM data register H (ECPWDRH)
- Event counter PWM data register L (ECPWDRL)
- Input pin edge select register (AEGSR)
- Event counter control register (ECCR)
- Event counter control/status register (ECCSR)
- Event counter H (ECH)
- Event counter L (ECL)

(1) Event Counter PWM Compare Register H (ECPWCRH)

ECPWCRH sets the one conversion period of the event counter PWM waveform.

Bit	Bit Name	Initial Value	R/W	Description
7	ECPWCRH7	1	R/W	One conversion period of event counter PWM waveform
6	ECPWCRH6	1	R/W	
5	ECPWCRH5	1	R/W	
4	ECPWCRH4	1	R/W	
3	ECPWCRH3	1	R/W	
2	ECPWCRH2	1	R/W	
1	ECPWCRH1	1	R/W	
0	ECPWCRH0	1	R/W	

Notes: When ECPWME in AEGSR is 1, the event counter PWM is operating and therefore ECPWCRH should not be modified.

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

(2) Event Counter PWM Compare Register L (ECPWCRL)

ECPWCRL sets the one conversion period of the event counter PWM waveform.

Bit	Bit Name	Initial Value	R/W	Description
7	ECPWCRL7	1	R/W	One conversion period of event counter PWM waveform
6	ECPWCRL6	1	R/W	
5	ECPWCRL5	1	R/W	
4	ECPWCRL4	1	R/W	
3	ECPWCRL3	1	R/W	
2	ECPWCRL2	1	R/W	
1	ECPWCRL1	1	R/W	
0	ECPWCRL0	1	R/W	

Notes: When ECPWME in AEGSR is 1, the event counter PWM is operating and therefore ECPWCRL should not be modified.

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

(3) Event Counter PWM Data Register H (ECPWDRH)

ECPWDRH controls data of the event counter PWM waveform generator.

Bit	Bit Name	Initial Value	R/W	Description
7	ECPWDRH7	0	W	Data control of event counter PWM waveform generator
6	ECPWDRH6	0	W	
5	ECPWDRH5	0	W	
4	ECPWDRH4	0	W	
3	ECPWDRH3	0	W	
2	ECPWDRH2	0	W	
1	ECPWDRH1	0	W	
0	ECPWDRH0	0	W	

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

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

(4) Event Counter PWM Data Register L (ECPWDRL)

ECPWDRL controls data of the event counter PWM waveform generator.

Bit	Bit Name	Initial Value	R/W	Description
7	ECPWDRL7	0	W	Data control of event counter PWM waveform generator
6	ECPWDRL6	0	W	
5	ECPWDRL5	0	W	
4	ECPWDRL4	0	W	
3	ECPWDRL3	0	W	
2	ECPWDRL2	0	W	
1	ECPWDRL1	0	W	
0	ECPWDRL0	0	W	

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

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

(5) Input Pin Edge Select Register (AEGSR)

AEGSR selects rising, falling, or both edge sensing for the AEVH, AEVL, and IRQAEC pins.

Bit	Bit Name	Initial Value	R/W	Description
7	AHEGS1	0	R/W	AEC Edge Select H
6	AHEGS0	0	R/W	Select rising, falling, or both edge sensing for the AEVH pin. 00: Falling edge on AEVH pin is sensed 01: Rising edge on AEVH pin is sensed 10: Both edges on AEVH pin are sensed 11: Setting prohibited
5	ALEGS1	0	R/W	AEC Edge Select L
4	ALEGS0	0	R/W	Select rising, falling, or both edge sensing for the AEVL pin. 00: Falling edge on AEVL pin is sensed 01: Rising edge on AEVL pin is sensed 10: Both edges on AEVL pin are sensed 11: Setting prohibited
3	AIEGS1	0	R/W	IRQAEC Edge Select
2	AIEGS0	0	R/W	Select rising, falling, or both edge sensing for the IRQAEC pin. 00: Falling edge on IRQAEC pin is sensed 01: Rising edge on IRQAEC pin is sensed 10: Both edges on IRQAEC pin are sensed 11: Setting prohibited
1	ECPWME	0	R/W	Event Counter PWM Enable Controls operation of event counter PWM and selection of IRQAEC. 0: AEC PWM halted, IRQAEC selected 1: AEC PWM enabled, IRQAEC not selected
0	—	0	R/W	Reserved This bit can be read from or written to. However, this bit should not be set to 1.

(6) Event Counter Control Register (ECCR)

ECCR controls the counter input clock and IRQAEC/IECPWM.

Bit	Bit Name	Initial Value	R/W	Description
7	ACKH1	0	R/W	AEC Clock Select H
6	ACKH0	0	R/W	Select the clock used by ECH. 00: AEVH pin input 01: ϕ/2 10: ϕ/4 11: ϕ/8
5	ACKL1	0	R/W	AEC Clock Select L
4	ACKL0	0	R/W	Select the clock used by ECL. 00: AEVL pin input 01: ϕ/2 10: ϕ/4 11: ϕ/8
3	PWCK2	0	R/W	Event Counter PWM Clock Select
2	PWCK1	0	R/W	Select the event counter PWM clock.
1	PWCK0	0	R/W	000: ϕ/2 001: ϕ/4 010: ϕ/8 011: ϕ/16 1x0: ϕ/32 1x1 ϕ/64
0	—	0	R/W	Reserved This bit can be read from or written to. However, this bit should not be set to 1.

[Legend] x: Don't care.

(7) Event Counter Control/Status Register (ECCSR)

ECCSR controls counter overflow detection, counter clear resetting, and the count-up function.

Bit	Bit Name	Initial Value	R/W	Description
7	OVH	0	R/W*	Counter Overflow H This is a status flag indicating that ECH has overflowed. [Setting condition] When ECH overflows from H'FF to H'00 [Clearing condition] When this bit is written to 0 after reading OVH = 1
6	OVL	0	R/W*	Counter Overflow L This is a status flag indicating that ECL has overflowed. [Setting condition] When ECL overflows from H'FF to H'00 [Clearing condition] When this bit is written to 0 after reading OVL = 1
5	—	0	R/W	Reserved This bit can be read from or written to. However, the initial value should not be changed.
4	CH2	0	R/W	Channel Select Selects how ECH and ECL event counters are used 0: ECH and ECL are used together as a single-channel 16-bit event counter 1: ECH and ECL are used as two-channel 8-bit event counter
3	CUEH	0	R/W	Count-Up Enable H Enables event clock input to ECH. 0: ECH event clock input is disabled (ECH value is retained) 1: ECH event clock input is enabled
2	CUEL	0	R/W	Count-Up Enable L Enables event clock input to ECL. 0: ECL event clock input is disabled (ECL value is retained) 1: ECL event clock input is enabled

Bit	Bit Name	Initial Value	R/W	Description
1	CRCH	0	R/W	Counter Reset Control H Controls resetting of ECH. 0: ECH is reset 1: ECH reset is cleared and count-up function is enabled
0	CRCL	0	R/W	Counter Reset Control L Controls resetting of ECL. 0: ECL is reset 1: ECL reset is cleared and count-up function is enabled

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

(8) Event Counter H (ECH)

ECH is an 8-bit read-only up-counter that operates as an independent 8-bit event counter. ECH also operates as the upper 8-bit up-counter of a 16-bit event counter configured in combination with ECL.

Bit	Bit Name	Initial Value	R/W	Description
7	ECH7	0	R	Either the external asynchronous event AEVH pin, $\phi/2$, $\phi/4$, or $\phi/8$, or the overflow signal from lower 8-bit counter ECL can be selected as the input clock source. ECH can be cleared by clearing the CRC bits in ECCSR to 0.
6	ECH6	0	R	
5	ECH5	0	R	
4	ECH4	0	R	
3	ECH3	0	R	
2	ECH2	0	R	
1	ECH1	0	R	
0	ECH0	0	R	

(9) Event Counter L (ECL)

ECL is an 8-bit read-only up-counter that operates as an independent 8-bit event counter. ECL also operates as the lower 8-bit up-counter of a 16-bit event counter configured in combination with ECH.

Bit	Bit Name	Initial Value	R/W	Description
7	ECL7	0	R	Either the external asynchronous event AEVL pin, $\phi/2$, $\phi/4$, or $\phi/8$ can be selected as the input clock source. ECL can be cleared by clearing the CRCL bit in ECCSR to 0.
6	ECL6	0	R	
5	ECL5	0	R	
4	ECL4	0	R	
3	ECL3	0	R	
2	ECL2	0	R	
1	ECL1	0	R	
0	ECL0	0	R	

9.4.4 Operation

(1) 16-Bit Counter Operation

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

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

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

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

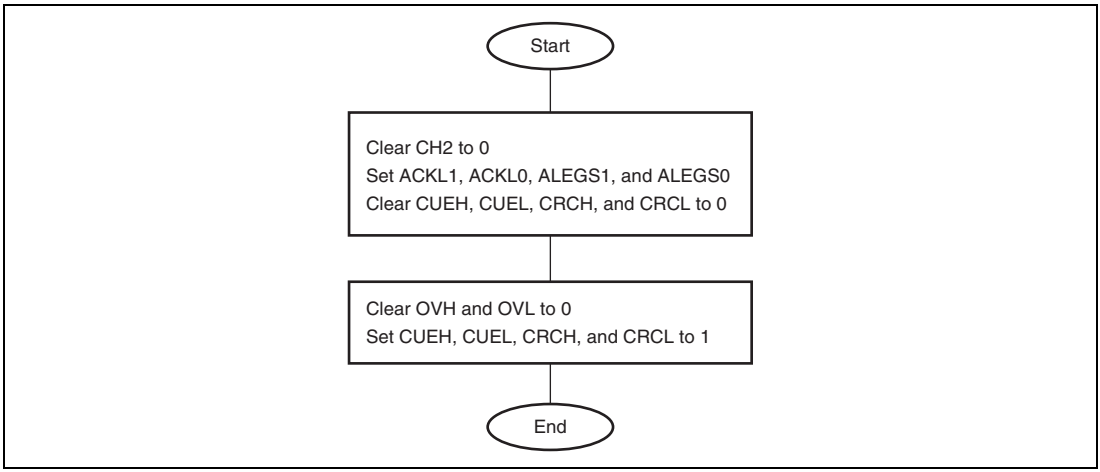


Figure 9.8 Example of Software Processing when Using ECH and ECL as 16-Bit Event Counter

As CH2 is cleared to 0 by a reset, ECH and ECL operate as a 16-bit event counter after a reset, and as ACKL1 and ACKL0 are cleared to B′00, the operating clock is asynchronous event input from the AEVL pin (using falling edge sensing).

When the next clock is input after the count value reaches H'FF in both ECH and ECL, ECH and ECL overflow from H'FFFF to H'0000, the OVH flag is set to 1 in ECCSR, the ECH and ECL count values each return to H'00, and counting up is restarted. When overflow occurs, the IRREC bit is set to 1 in IRR2. If the IENEC bit in IENR2 is 1 at this time, an interrupt request is sent to the CPU.

(2) 8-Bit Counter Operation

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

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

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

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

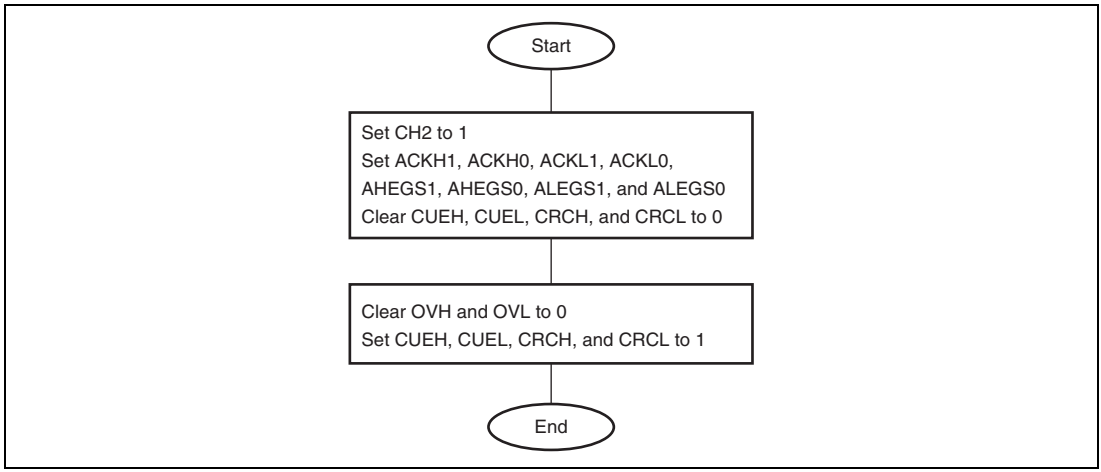


Figure 9.9 Example of Software Processing when Using ECH and ECL as 8-Bit Event Counters

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

(3) IRQAEC Operation

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

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

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

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

(4) Event Counter PWM Operation

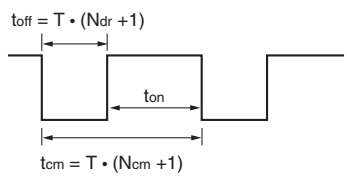
When ECPWME in AEGSR is 1, the ECH and ECL input clocks are enabled only when event counter PWM output (IECPWM) is high. When IECPWM is low, the input clocks are not input to the counters, and so ECH and ECL do not count. ECH and ECL count operations can therefore be controlled cyclically from outside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

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

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

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

Figure 9.10 and table 9.6 show examples of event counter PWM operation.



[Legend]
- t_{on}: Clock input enable time
- t_{off}: Clock input disable time
- t_{cm}: One conversion period
- T: ECPWM input clock cycle
- N_{dr}: Value of ECPWDRH and ECPWDRL
  Fixed low when Ndr = H'FFFF
- N_{cm}: Value of ECPWCRH and ECPWCRL

Figure 9.10 Event Counter Operation Waveform

Note: Ndr and Ncm above must be set so that Ndr < Ncm. If the settings do not satisfy this condition, do not set ECPWME to 1 in AEGSR.

Table 9.6 Examples of Event Counter PWM Operation

Conditions: fosc = 4 MHz, fϕ = 2 MHz, high-speed active mode, ECPWCR value (Ncm) = H'7A11, ECPWDR value (Ndr) = H'16E3

Clock Source Selection	Clock Source Cycle (T)*	ECPWMCR Value (Ncm)	ECPWMDR Value (Ndr)	toff = T × (Ndr + 1)	tcm = T × (Ncm + 1)	ton = tcm – toff
ϕ/2	1 µs	H'7A11	H'16E3	5.86 ms	31.25 ms	25.39 ms
ϕ/4	2 µs	D'31249	D'5859	11.72 ms	62.5 ms	50.78 ms
ϕ/8	4 µs			23.44 ms	125.0 ms	101.56 ms
ϕ/16	8 µs			46.88 ms	250.0 ms	203.12 ms
ϕ/32	16 µs			93.76 ms	500.0 ms	406.24 ms
ϕ/64	32 µs			187.52 ms	1000.0 ms	812.48 ms

Note: * toff minimum width

(5) Clock Input Enable/Disable Function Operation

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

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

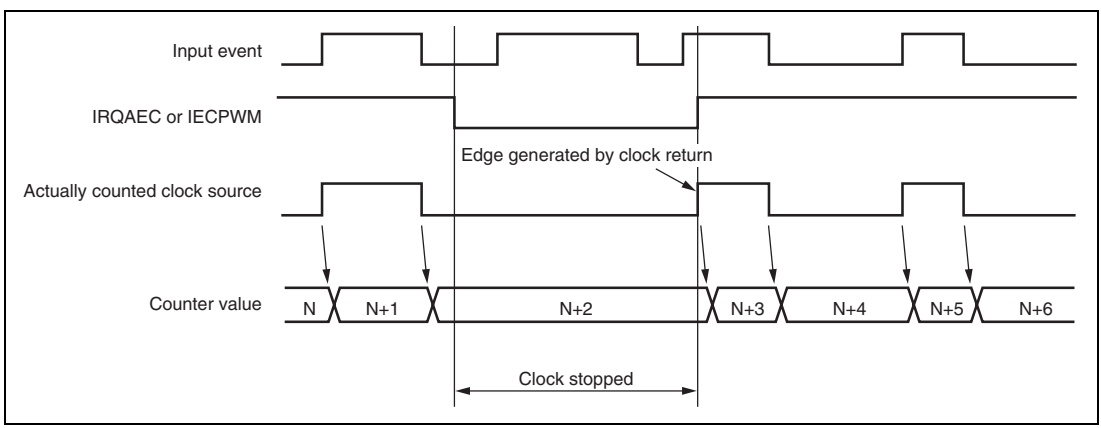


Figure 9.11 Example of Clock Control Operation

9.4.5 Operating States of Asynchronous Event Counter

The operating states of the asynchronous event counter are shown in table 9.7.

Table 9.7 Operating States of Asynchronous Event Counter

Operating Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby	Module Standby
AEGSR	Reset	Functions	Functions	Retained*1	Functions	Functions	Retained*1	Retained
ECCR	Reset	Functions	Functions	Retained*1	Functions	Functions	Retained*1	Retained
ECCSR	Reset	Functions	Functions	Retained*1	Functions	Functions	Retained*1	Retained
ECH	Reset	Functions	Functions	Functions*1*2	Functions*2	Functions*2	Functions*1*2	Halted
ECL	Reset	Functions	Functions	Functions*1*2	Functions*2	Functions*2	Functions*1*2	Halted
IRQAEC	Reset	Functions	Functions	Retained*3	Functions	Functions	Retained*3	Retained*4
Event counter PWM	Reset	Functions	Functions	Retained	Retained	Retained	Retained	Retained

Notes: 1. When an asynchronous external event is input, the counter increments but the counter overflow H/L flags are not affected.
2. Functions when asynchronous external events are selected; halted and retained otherwise.
3. Clock control by IRQAEC operates, but interrupts do not.
4. As the clock is stopped in module standby mode, IRQAEC has no effect.

9.4.6 Usage Notes

1. When reading the values in ECH and ECL, first clear bits CUEH and CUEL to 0 in ECCSR in 8-bit mode and clear bit CUEL to 0 in 16-bit mode to prevent asynchronous event input to the counter. The correct value will not be returned if the event counter increments while being read.
2. The maximum clock frequency that may be input to the AEVH and AEVL pins is either 4MHz with voltage range of 1.8 V to 3.6 V, or 10 MHz with voltage range of 7 V to 3.6 V. For information on the clock high width and low width, see section 14, Electrical Characteristics. The duty ratio does not matter as long as the high width and low width satisfy the minimum requirement.

Mode		Maximum Clock Frequency Input to AEVH/AEVL Pin
Active (high-speed), sleep (high-speed)		10 MHz
Active (medium-speed), sleep (medium-speed)	(ϕ/16)	2 • f_{osc}
	(ϕ/32)	f_{osc}
	(ϕ/64)	1/2 • f_{osc}
f_{osc} = 1 MHz to 4 MHz	(ϕ/128)	1/4 • f_{osc}
Watch, subactive, subsleep, standby	(ϕ_w/2)	1000 kHz
	(ϕ_w/4)	500 kHz
ϕ_w = 32.768 kHz or 38.4 kHz*2	(ϕ_w/8)	250 kHz

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

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

9.5 Watchdog Timer

The watchdog timer is an 8-bit timer that can generate an internal reset signal for this LSI if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow.

Figure 9.12 shows a block diagram of the watchdog timer.

9.5.1 Features

- Selectable from two counter input clocks
  Two clock sources (ϕ/8192 or ϕ_w/32) can be selected as the timer-counter clock.
- Reset signal generated on counter overflow
  An overflow period of 1 to 256 times the selected clock can be set.
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (For details, refer to section 5.4, Module Standby Function.)

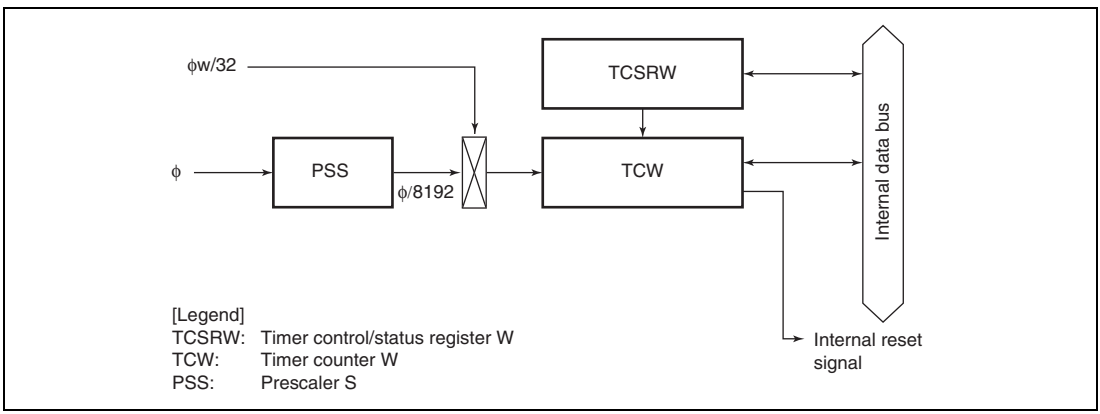


Figure 9.12 Block Diagram of Watchdog Timer

9.5.2 Register Descriptions

The watchdog timer has the following registers.

- Timer control/status register W (TCSRW)
- Timer counter W (TCW)

(1) Timer Control/Status Register W (TCSRW)

TCSRW performs the TCSRW and TCW write control. TCSRW also controls the watchdog timer operation and indicates the operating state. TCSRW must be rewritten by using the MOV instruction. The bit manipulation instruction cannot be used to change the setting value.

Bit	Bit Name	Initial Value	R/W	Description
7	B6WI	1	R	Bit 6 Write Inhibit The TCWE bit can be written only when the write value of the B6WI bit is 0. This bit is always read as 1.
6	TCWE	0	R/(W)*	Timer Counter W Write Enable TCW can be written when the TCWE bit is set to 1. When writing data to this bit, the value for bit 7 must be 0.
5	B4WI	1	R	Bit 4 Write Inhibit The TCSRWE bit can be written only when the write value of the B4WI bit is 0. This bit is always read as 1.
4	TCSRWE	0	R/(W)*	Timer Control/Status Register W Write Enable The WDON and WRST bits can be written when the TCSRWE bit is set to 1. When writing data to this bit, the value for bit 5 must be 0.
3	B2WI	1	R	Bit 2 Write Inhibit This bit can be written to the WDON bit only when the write value of the B2WI bit is 0. This bit is always read as 1.

Bit	Bit Name	Initial Value	R/W	Description
2	WDON	0	R/(W)*	Watchdog Timer On TCW starts counting up when WDON is set to 1 and halts when WDON is cleared to 0. [Setting condition] When 1 is written to the WDON bit while writing 0 to the B2WI bit when the TCSRWE bit=1 [Clearing condition] • Reset by $\overline{\text{RES}}$ pin • When 0 is written to the WDON bit while writing 0 to the B2WI when the TCSRWE bit=1
1	B0WI	1	R	Bit 0 Write Inhibit This bit can be written to the WRST bit only when the write value of the B0WI bit is 0. This bit is always read as 1.
0	WRST	0	R/(W)*	Watchdog Timer Reset [Setting condition] When TCW overflows and an internal reset signal is generated [Clearing condition] • Reset by $\overline{\text{RES}}$ pin • When 0 is written to the WRST bit while writing 0 to the B0WI bit when the TCSRWE bit = 1

Note: * These bits can be written only when the writing conditions are satisfied.

(2) Timer Counter W (TCW)

TCW is an 8-bit readable/writable up-counter. When TCW overflows from H'FF to H'00, the internal reset signal is generated and the WRST bit in TCSRW is set to 1. TCW is initialized to H'00.

9.5.3 Operation

The watchdog timer is provided with an 8-bit counter. The input clock is selected by the WDCKS bit in the port mode register 2 (PMR2)*: ϕ/8192 is selected when the WDCKS bit is cleared to 0, and ϕw/32 when set to 1.. If 1 is written to WDON while writing 0 to B2WI when the TCSRWE bit in TCSRW is set to 1, TCW begins counting up (to operate the watchdog timer, two write accesses to TCSRW are required). When a clock pulse is input after the TCW count value has reached H'FF, the watchdog timer overflows and an internal reset signal is generated. The internal reset signal is output for a period of 512 ϕ_{osc} clock cycles. TCW is a writable counter, and when a value is set in TCW, the count-up starts from that value. An overflow period in the range of 1 to 256 input clock cycles can therefore be set, according to the TCW set value.

Note: * For details, refer to section 8.1.5, Port Mode Register 2 (PMR2).

Figure 9.13 shows an example of watchdog timer operation.

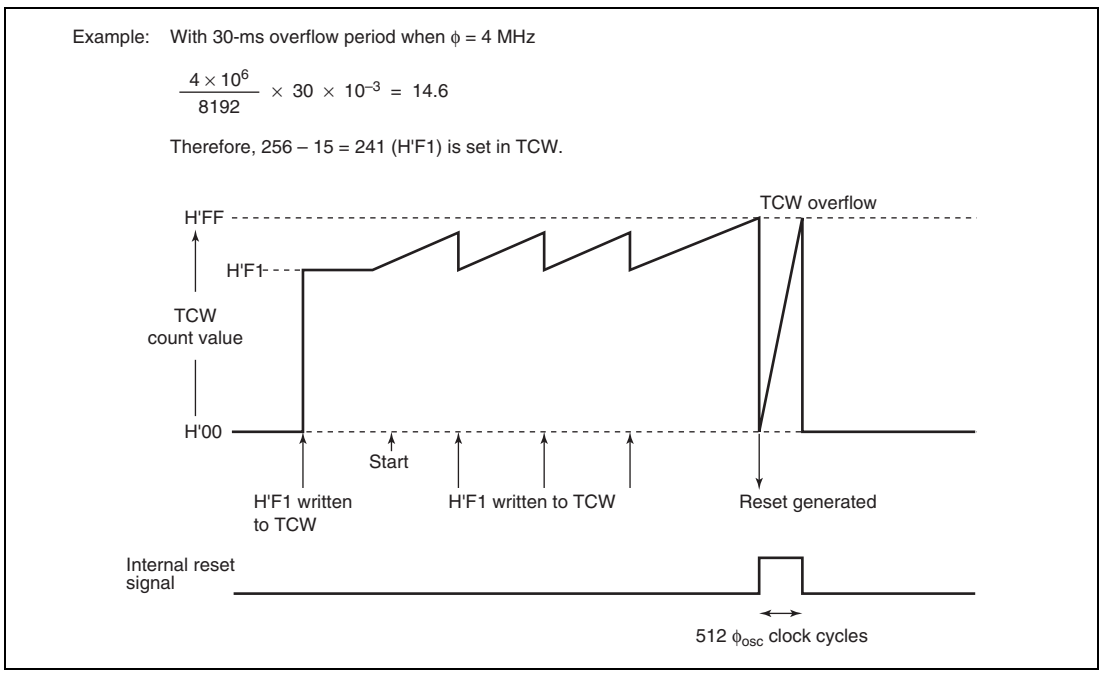


Figure 9.13 Example of Watchdog Timer Operation

9.5.4 Operating States of Watchdog Timer

Tables 9.8 summarizes the operating states of the watchdog timer.

Table 9.8 Operating States of Watchdog Timer

Operating Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby	Module Standby
TCW	Reset	Functions	Functions	Halted	Functions/ Halted*	Halted	Halted	Halted
TCSRW	Reset	Functions	Functions	Retained	Functions/ Halted*	Retained	Retained	Retained

Note: * Functions when $\phi_w/32$ is selected as the input clock.

Section 10 Serial Communication Interface 3 (SCI3)

Serial Communication Interface 3 (SCI3) can handle both asynchronous and clock synchronous serial communication. In the asynchronous method, serial data communication can be carried out using standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or an Asynchronous Communication Interface Adapter (ACIA).

Figure 10.1 shows a block diagram of the SCI3.

10.1 Features

- Choice of asynchronous or clock synchronous serial communication mode
- Full-duplex communication capability

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously.

  Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected
- On-chip baud rate generator, internal clock, or external clock can be selected as a transfer clock source.
- Six interrupt sources

  Transmit-end, transmit-data-empty, receive-data-full, overrun error, framing error, and parity error.
- Use of module standby mode enables this module to be placed in standby mode independently when it is not in use (for details, see section 5.4, Module Standby Function).

Asynchronous mode

- Data length: 7, 8, or 5 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the RXD32 pin level directly in the case of a framing error

Clocked synchronous mode

- Data length: 8 bits
- Receive error detection: Overrun errors detected

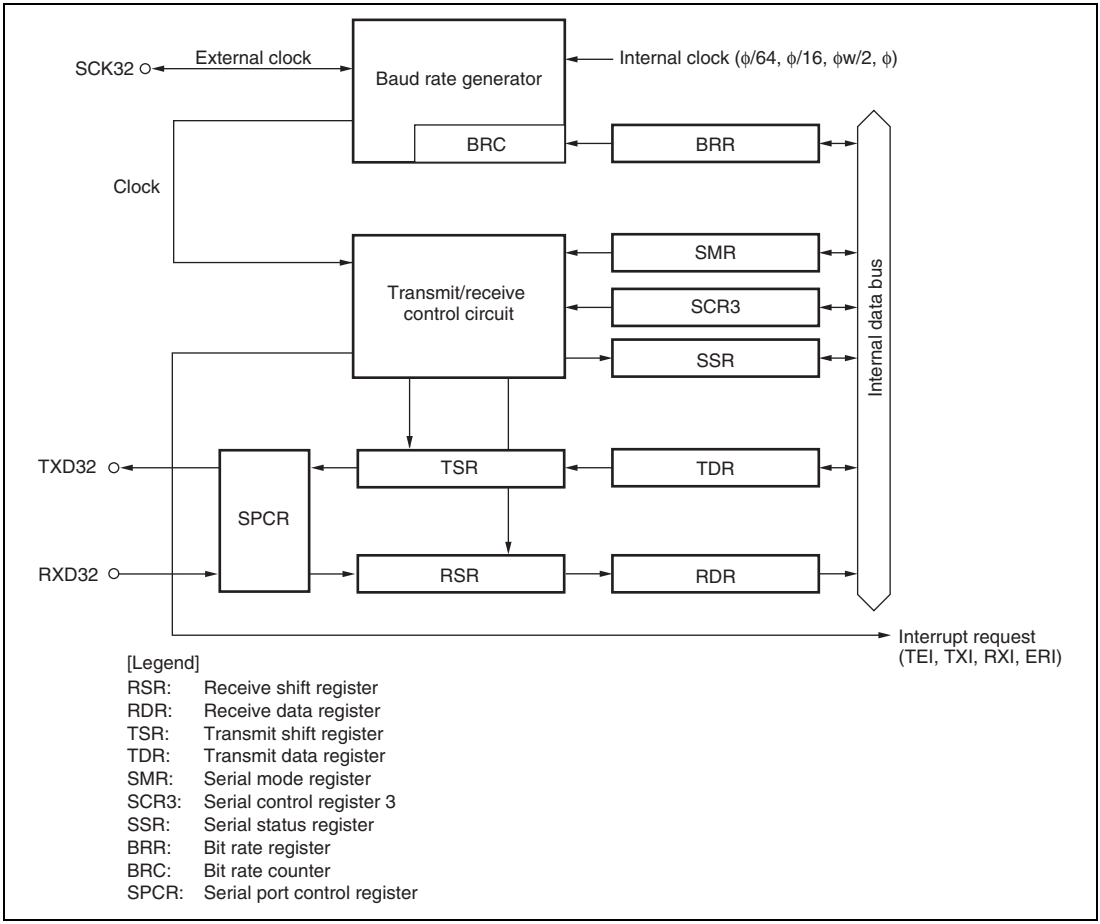


Figure 10.1 Block Diagram of SCI3

10.2 Input/Output Pins

Table 10.1 shows the SCI3 pin configuration.

Table 10.1 Pin Configuration

Pin Name	Abbreviation	I/O	Function
SCI3 clock	SCK32	I/O	SCI3 clock input/output
SCI3 receive data input	RXD32	Input	SCI3 receive data input
SCI3 transmit data output	TXD32	Output	SCI3 transmit data output

10.3 Register Descriptions

The SCI3 has the following registers.

- Receive shift register (RSR)
- Receive data register (RDR)
- Transmit shift register (TSR)
- Transmit data register (TDR)
- Serial mode register (SMR)
- Serial control register 3 (SCR3)
- Serial status register (SSR)
- Bit rate register (BRR)
- Serial port control register (SPCR)

10.3.1 Receive Shift Register (RSR)

RSR is a shift register that is used to receive serial data input from the RXD32 pin and convert it into parallel data. When one byte of data has been received, it is transferred to RDR automatically. RSR cannot be directly accessed by the CPU.

10.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores received data. When the SCI3 has received one byte of serial data, it transfers the received serial data from RSR to RDR, where it is stored. After this, RSR is receive-enabled. As RSR and RDR function as a double buffer in this way, continuous receive operations are possible. After confirming that the RDRF bit in SSR is set to 1, read RDR only once. RDR cannot be written to by the CPU. RDR is initialized to H'00 at a reset and in standby, watch, or module standby mode.

10.3.3 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI3 first transfers transmit data from TDR to TSR automatically, then sends the data that starts from the LSB to the TXD32 pin. Data transfer from TDR to TSR is not performed if no data has been written to TDR (if the TDRE bit in SSR is set to 1). TSR cannot be directly accessed by the CPU.

10.3.4 Transmit Data Register (TDR)

TDR is an 8-bit register that stores data for transmission. When the SCI3 detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structure of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR during transmission of one-frame data, the SCI3 transfers the written data to TSR to continue transmission. To achieve reliable serial transmission, write transmit data to TDR only once after confirming that the TDRE bit in SSR is set to 1. TDR is initialized to H'FF at a reset and in standby, watch, or module standby mode.

10.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI3's serial transfer format and select the on-chip baud rate generator clock source. SMR is initialized to H'00 at a reset and in standby, watch, or module standby mode.

Bit	Bit Name	Initial Value	R/W	Description
7	COM	0	R/W	Communication Mode 0: Asynchronous mode 1: Clocked synchronous mode
6	CHR	0	R/W	Character Length (enabled only in asynchronous mode) 0: Selects 8 or 5 bits as the data length. 1: Selects 7 or 5 bits as the data length. When 7-bit data is selected, the MSB (bit 7) in TDR is not transmitted. To select 5 bits as the data length, set 1 to both the PE and MP bits. The three most significant bits (bits 7, 6, and 5) in TDR are not transmitted. In clock synchronous mode, the data length is fixed to 8 bits regardless of the CHR bit setting.
5	PE	0	R/W	Parity Enable (enabled only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. In clock synchronous mode, parity bit addition and checking is not performed regardless of the PE bit setting.

Bit	Bit Name	Initial Value	R/W	Description
4	PM	0	R/W	Parity Mode (enabled only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity. When even parity is selected, a parity bit is added in transmission so that the total number of 1 bits in the transmit data plus the parity bit is an even number; in reception, a check is carried out to confirm that the number of 1 bits in the receive data plus the parity bit is an even number. When odd parity is selected, a parity bit is added in transmission so that the total number of 1 bits in the transmit data plus the parity bit is an odd number; in reception, a check is carried out to confirm that the number of 1 bits in the receive data plus the parity bit is an odd number. If parity bit addition and checking is disabled in clock synchronous mode and asynchronous mode, the PM bit setting is invalid.
3	STOP	0	R/W	Stop Bit Length (enabled only in asynchronous mode) Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits For reception, only the first stop bit is checked, regardless of the value in the bit. If the second stop bit is 0, it is treated as the start bit of the next transmit character.
2	MP	0	R/W	Five-Bit Communications When this bit is set to 1, the five-bit communications format is available. When writing 1 to this bit, be sure to write 1 to the PE bit (bit 5 of this register) simultaneously.

Bit	Bit Name	Initial Value	R/W	Description
1	CKS1	0	R/W	Clock Select 0 and 1
0	CKS0	0	R/W	These bits select the clock source for the on-chip baud rate generator. 00: ϕ clock (n = 0) 01: ϕw/2 or ϕw clock (n = 1) 10: ϕ/16 clock (n = 2) 11: ϕ/64 clock (n = 3) When the setting value is 01 in active mode and sleep mode, ϕw/2 clock is set. In subactive mode and subsleep mode, ϕw clock is set. The SCI3 is enabled only when ϕw /2 is selected for the CPU operating clock. For the relationship between the bit rate register setting and the baud rate, see section 10.3.8, Bit Rate Register (BRR). n is the decimal representation of the value of n in BRR (see section 10.3.8, Bit Rate Register (BRR).

10.3.6 Serial Control Register 3 (SCR3)

SCR3 is a register that enables or disables SCI3 transfer operations and interrupt requests, and is also used to select the transfer clock source. SCR3 is initialized to H'00 at a reset and in standby, watch, or module standby mode. For details on interrupt requests, refer to section 10.6, Interrupts.

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, the TXI interrupt request is enabled. TXI can be released by clearing the TDRE bit or TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled. RXI and ERI can be released by clearing bit RDRF or the FER, PER, or OER error flag to 0, or by clearing bit RIE to 0.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled. When this bit is 0, the TDRE bit in SSR is fixed at 1. When transmit data is written to TDR while this bit is 1, bit TDRE in SSR is cleared to 0 and serial data transmission is started. Be sure to carry out SMR settings, and setting of bit SPC32 in SPCR, to decide the transmission format before setting bit TE to 1.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled. In this state, serial data reception is started when a start bit is detected in asynchronous mode or serial clock input is detected in clock synchronous mode. Be sure to carry out the SMR settings to decide the reception format before setting bit RE to 1. Note that the RDRF, FER, PER, and OER flags in SSR are not affected when bit RE is cleared to 0, and retain their previous state.

Bit	Bit Name	Initial Value	R/W	Description
3	—	0	R/W	Reserved Only 0 should be written to this bit.
2	TEIE	0	R/W	Transmit End Interrupt Enable When this bit is set to 1, the TEI interrupt request is enabled. TEI can be released by clearing bit TDRE to 0 and clearing bit TEND to 0 in SSR, or by clearing bit TEIE to 0.
1	CKE1	0	R/W	Clock Enable 0 and 1
0	CKE0	0	R/W	Selects the clock source. Asynchronous mode: 00: Internal baud rate generator 01: Internal baud rate generator Outputs a clock of the same frequency as the bit rate from the SCK32 pin. 10: External clock Inputs a clock with a frequency 16 times the bit rate from the SCK32 pin. 11: Reserved Clocked synchronous mode: 00: Internal clock (SCK32 pin functions as clock output) 01: Reserved 10: External clock (SCK32 pin functions as clock input) 11: Reserved

10.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI3 and multiprocessor bits for transfer. 1 cannot be written to flags TDRE, RDRF, OER, PER, and FER; they can only be cleared. SSR is initialized to H'84 at a reset and in standby, watch, or module standby mode.

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates that transmit data is stored in TDR. [Setting conditions] • When the TE bit in SCR3 is 0 • When data is transferred from TDR to TSR [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When the transmit data is written to TDR
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates that the received data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing conditions] • When 0 is written to RDRF after reading RDRF = 1 • When data is read from RDR If an error is detected in reception, or if the RE bit in SCR3 has been cleared to 0, RDR and bit RDRF are not affected and retain their previous state. Note that if data reception is completed while bit RDRF is still set to 1, an overrun error (OER) will occur and the receive data will be lost.

Bit	Bit Name	Initial Value	R/W	Description
5	OER	0	R/(W)*	Overrun Error [Setting condition] • When an overrun error occurs in reception [Clearing condition] • When 0 is written to OER after reading OER = 1 When bit RE in SCR3 is cleared to 0, bit OER is not affected and retains its previous state. When an overrun error occurs, RDR retains the receive data it held before the overrun error occurred, and data received after the error is lost. Reception cannot be continued with bit OER set to 1, and in clock synchronous mode, transmission cannot be continued either.
4	FER	0	R/(W)*	Framing Error [Setting condition] • When a framing error occurs in reception [Clearing condition] • When 0 is written to FER after reading FER = 1 When bit RE in SCR3 is cleared to 0, bit FER is not affected and retains its previous state. Note that, in 2-stop-bit mode, only the first stop bit is checked for a value of 1, and the second stop bit is not checked. When a framing error occurs, the receive data is transferred to RDR but bit RDRF is not set. Reception cannot be continued with bit FER set to 1. In clock synchronous mode, neither transmission nor reception is possible when bit FER is set to 1.

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error [Setting condition] • When a parity error is generated during reception [Clearing condition] • When 0 is written to PER after reading PER = 1 When bit RE in SCR3 is cleared to 0, bit PER is not affected and retains its previous state. Receive data in which a parity error has occurred is still transferred to RDR, but bit RDRF is not set. Reception cannot be continued with bit PER set to 1. In clock synchronous mode, neither transmission nor reception is possible when bit PER is set to 1.
2	TEND	1	R	Transmit End [Setting conditions] • When the TE bit in SCR3 is 0 • When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When the transmit data is written to TDR
1	—	0	R	Reserved This is a read-only bit and cannot be modified.
0	—	0	R/W	Reserved The write value should always be 0.

Note: * Only 0 can be written for clearing a flag.

10.3.8 Bit Rate Register (BRR)

BRR is an 8-bit readable/writable register that adjusts the bit rate. BRR is initialized to H'FF at a reset and in standby, watch, or module standby mode. Table 10.2 shows the relationship between the N setting in BRR and the n setting in bits CKS1 and CKS0 of SMR in asynchronous mode. Table 10.4 shows the maximum bit rate for each frequency in asynchronous mode. The values shown in both tables 10.2 and 10.4 are values in active (high-speed) mode. Table 10.5 shows the relationship between the N setting in BRR and the n setting in bits CKS1 and CKS0 in SMR in clock synchronous mode. The values are shown in table 10.5. The N setting in BRR and error for other operating frequencies and bit rates can be obtained by the following formulas:

[Asynchronous Mode]

$$N = \frac{\phi}{32 \times 2^{2n} \times B} - 1$$

$$\text{Error (\%)} = \frac{\text{B (bit rate obtained from n, N, } \phi\text{)} - \text{R (bit rate in left-hand column in table 10.2)}}{\text{R (bit rate in left-hand column in table 10.2)}} \times 100$$

Legend:
- B: Bit rate (bit/s)
- N: BRR setting for baud rate generator ($0 \leq N \leq 255$)
- ϕ: Operating frequency (Hz)
- n: Baud rate generator input clock number (n = 0, 2, or 3)
  (The relation between n and the clock is shown in table 10.3.)

Table 10.2 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (1)

	φ											
	16.4 kHz			19.45 kHz			1 MHz			1.2288 MHz		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	—	—	—	—	—	—	2	17	–1.36	2	21	–0.83
150	—	—	—	0	3	0	2	12	0.16	3	3	0
200	—	—	—	0	2	0	2	9	–2.34	3	2	0
250	0	1	2.5	—	—	—	3	1	–2.34	0	153	–0.26
300	—	—	—	0	1	0	0	103	0.16	3	1	0
600	—	—	—	0	0	0	0	51	0.16	3	0	0
1200				—	—	—	0	25	0.16	2	1	0
2400							0	12	0.16	2	0	0
4800							—	—	—	0	7	0
9600							—	—	—	0	3	0
19200							—	—	—	0	1	0
31250							0	0	0	—	—	—
38400							—	—	—	0	0	0

Table 10.2 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (2)

	φ											
	2 MHz			5 MHz			8 MHz			10 MHz		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	8	–1.36	3	21	0.88	3	35	–1.36	3	43	0.88
150	2	25	0.16	3	15	1.73	3	25	0.16	3	32	–1.36
200	3	4	–2.34	3	11	1.73	3	19	–2.34	3	23	1.73
250	2	15	–2.34	3	9	–2.34	3	15	–2.34	3	19	–2.34
300	2	12	0.16	3	7	1.73	3	12	0.16	3	15	1.73
600	0	103	0.16	3	3	1.73	2	25	0.16	3	7	1.73
1200	0	51	0.16	3	1	1.73	2	12	0.16	3	3	1.73
2400	0	25	0.16	3	0	1.73	0	103	0.16	3	1	1.73
4800	0	12	0.16	2	1	1.73	0	51	0.16	3	0	1.73
9600	—	—	—	2	0	1.73	0	25	0.16	2	1	1.73
19200	—	—	—	0	7	1.73	0	12	0.16	2	0	1.73
31250	0	1	0	0	4	0	0	7	0	0	9	0
38400	—	—	—	0	3	1.73	—	—	—	0	7	1.73

[Legend]
No indication: Setting not possible.
— : A setting is available but error occurs

Table 10.3 Relation between n and Clock

		SMR Setting	
n	Clock	CKS1	CKS0
0	φ	0	0
0	$\phi_w/2$*1/ϕ_w*2	0	1
2	φ/16	1	0
3	φ/64	1	1

Notes: 1. $\phi_w/2$ clock in active (medium-speed/high-speed) mode and sleep (medium-speed/high-speed) mode

2. ϕ_w clock in subactive mode and subsleep mode
In subactive or subsleep mode, the SCI3 can be operated when CPU clock is $\phi_w/2$ only.

Table 10.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

			Setting	
OSC (MHz)	**ϕ (MHz)**	**Maximum Bit Rate (bit/s)**	**n**	**N**
0.0384*	0.0192	600	0	0
2	1	31250	0	0
2.4576	1.2288	38400	0	0
4	2	62500	0	0
10	5	156250	0	0
16	8	250000	0	0
20	10	312500	0	0

Note: * When CKS1 = 0 and CKS0 = 1 in SMR

Table 10.5 BRR Settings for Various Bit Rates (Clocked Synchronous Mode) (1)

	ϕ								
	19.2 kHz			**1 MHz**			**2 MHz**		
Bit Rate (bit/s)	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**
200	0	23	0	—	—	—	—	—	—
250	—	—	—	—	—	—	2	124	0
300	2	0	0	—	—	—	—	—	—
500				—	—	—	—	—	—
1k				0	249	0	—	—	—
2.5k				0	99	0	0	199	0
5k				0	49	0	0	99	0
10k				0	24	0	0	49	0
25k				0	9	0	0	19	0
50k				0	4	0	0	9	0
100k				—	—	—	0	4	0
250k				0	0	0	0	1	0
500k							0	0	0
1M									

Table 10.5 BRR Settings for Various Bit Rates (Clocked Synchronous Mode) (2)

Bit Rate (bit/s)	φ								
	5 MHz			8 MHz			10 MHz		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
200	—	—	—	—	—	—	0	12499	0
250	—	—	—	3	124	0	2	624	0
300	—	—	—	—	—	—	0	8332	0
500	—	—	—	2	249	0	0	4999	0
1k	—	—	—	2	124	0	0	2499	0
2.5k	—	—	—	2	49	0	0	999	0
5k	0	249	0	2	24	0	0	499	0
10k	0	124	0	0	199	0	0	249	0
25k	0	49	0	0	79	0	0	99	0
50k	0	24	0	0	39	0	0	49	0
100k	—	—	—	0	19	0	0	24	0
250k	0	4	0	0	7	0	0	9	0
500k	—	—	—	0	3	0	0	4	0
1M	—	—	—	0	1	0	—	—	—

[Legend]

Blankx: No setting is available.

—: A setting is available but error occurs.

Note: The value set in BRR is given by the following formula:

$$N = \frac{\phi}{8 \times 2^{2n} \times B} - 1$$

B: Bit rate (bit/s)

N: BRR setting for baud rate generator ($0 \leq N \leq 255$)

φ: Operating frequency (Hz)

n: Baud rate generator input clock number (n = 0, 2, or 3)
(The relation between n and the clock is shown in table 10.6.)

Table 10.6 Relation between n and Clock

		SMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	ϕ	0	0
0	$\phi_w/2$[*1]/ϕ_w[*2]	0	1
2	$\phi/16$	1	0
3	$\phi/64$	1	1

Notes: 1. $\phi_w/2$ clock in active (medium-speed/high-speed) mode and sleep (medium-speed/high-speed) mode

2. ϕ_w clock in subactive mode and subsleep mode
   In subactive or subsleep mode, the SCI3 can be operated when CPU clock is $\phi_w/2$ only.

10.3.9 Serial Port Control Register (SPCR)

SPCR selects whether input/output data of the RXD32 and TXD32 pins is inverted or not.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
5	SPC32	0	R/W	P42/TXD32 Pin Function Switch This bit selects whether pin P42/TXD32 is used as P42 or as TXD32. 0: P42 I/O pin 1: TXD32 output pin* Note: * Set the TE bit in SCR3 after setting this bit to 1.
4	—	—	W	Reserved The write value should always be 0.
3	SCINV3	0	R/W	TXD32 Pin Output Data Inversion Switch This bit selects whether or not the logic level of the TXD32 pin output data is inverted. 0: TXD32 output data is not inverted 1: TXD32 output data is inverted
2	SCINV2	0	R/W	RXD32 Pin Input Data Inversion Switch This bit selects whether or not the logic level of the RXD32 pin input data is inverted. 0: RXD32 input data is not inverted 1: RXD32 input data is inverted
1, 0	—	—	W	Reserved The write value should always be 0.

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

10.4 Operation in Asynchronous Mode

Figure 10.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally stop bits (high level). In asynchronous mode, synchronization is performed at the falling edge of the start bit during reception. The data is sampled on the 8th pulse of a clock with a frequency 16 times the bit period, so that the transfer data is latched at the center of each bit. Inside the SCI3, the transmitter and receiver are independent units, enabling full duplex. Both the transmitter and the receiver also have a double-buffered structure, so data can be read or written during transmission or reception, enabling continuous data transfer. Table 10.7 shows the 16 data transfer formats that can be set in asynchronous mode. The format is selected by the settings in SMR as shown in table 10.8.

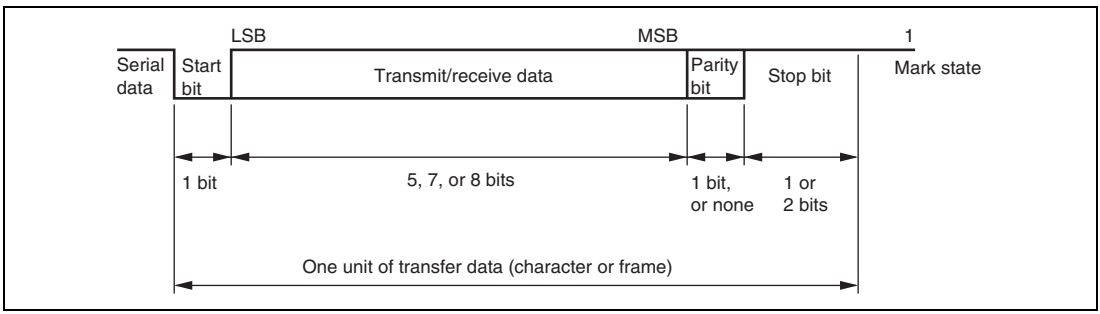


Figure 10.2 Data Format in Asynchronous Communication

10.4.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK32 pin can be selected as the SCI3's serial clock source, according to the setting of the COM bit in SMR and the CKE0 and CKE1 bits in SCR3. For details on selection of the clock source, see table 10.9. When an external clock is input at the SCK32 pin, the clock frequency should be 16 times the bit rate used. When the SCI3 is operated on an internal clock, the clock can be output from the SCK32 pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 10.3.

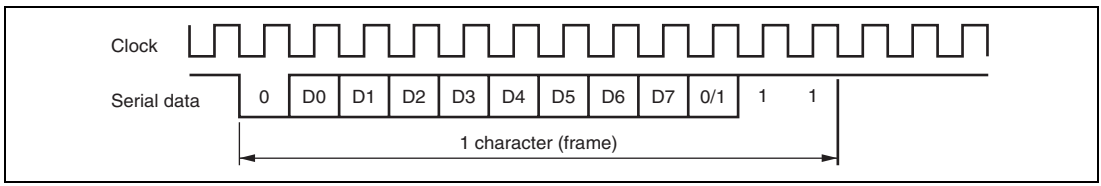


Figure 10.3 Relationship between Output Clock and Transfer Data Phase (Asynchronous Mode) (Example with 8-Bit Data, Parity, Two Stop Bits)

Table 10.7 Data Transfer Formats (Asynchronous Mode)

SMR				Serial Data Transfer Format and Frame Length
CHR	PE	MP	STOP	1 2 3 4 5 6 7 8 9 10 11 12
0	0	0	0	START 8-bit data STOP
0	0	0	1	START 8-bit data STOP STOP
0	0	1	0	Setting prohibited
0	0	1	1	Setting prohibited
0	1	0	0	START 8-bit data P STOP
0	1	0	1	START 8-bit data P STOP STOP
0	1	1	0	START 5-bit data STOP
0	1	1	1	START 5-bit data STOP STOP
1	0	0	0	START 7-bit data STOP
1	0	0	1	START 7-bit data STOP STOP
1	0	1	0	Setting prohibited
1	0	1	1	Setting prohibited
1	1	0	0	START 7-bit data P STOP
1	1	0	1	START 7-bit data P STOP STOP
1	1	1	0	START 5-bit data P STOP
1	1	1	1	START 5-bit data P STOP STOP

[Legend]
START: Start bit
STOP: Stop bit
P: Parity bit
MPB Multiprocessor bit

Table 10.8 SMR Settings and Corresponding Data Transfer Formats

SMR						Data Transfer Format			
Bit 7 COM	Bit 6 CHR	Bit 2 MP	Bit 5 PE	Bit 3 STOP	Mode	Data Length	Multiprocessor Bit	Parity Bit	Stop Bit Length
0	0	0	0	0	Asynchronous mode	8-bit data	No	No	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	1		0	0		7-bit data		No	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	0	1	0	0		Setting prohibited			
				1					
			1	0	Asynchronous mode	5-bit data	No	No	1 bit
				1					2 bits
	1		0	0		Setting prohibited			
				1					
			1	0	Asynchronous mode	5-bit data	No	Yes	1 bit
				1					2 bits
1	*	0	*	*	Clock synchronous mode	8-bit data	No	No	No

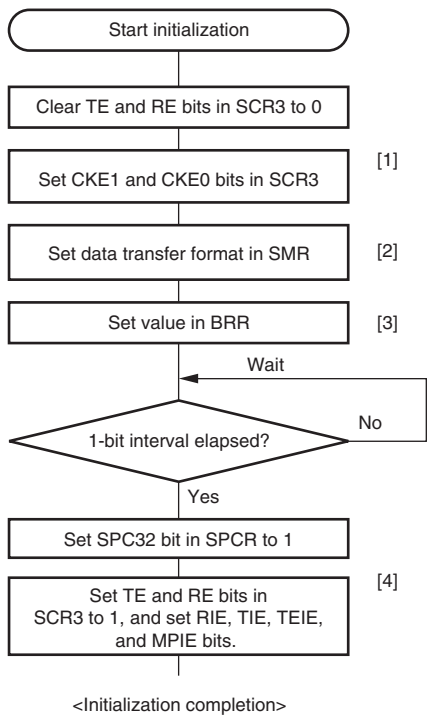
[Legend] *: Don’t care

Table 10.9 SMR and SCR3 Settings and Clock Source Selection

SMR	SCR3				
Bit 7	Bit 1	Bit 0		Transmit/Receive Clock	
COM	CKE1	CKE0	Mode	Clock Source	SCK32 Pin Function
0	0	0	Asynchronous mode	Internal	I/O port (SCK32 pin not used)
		1			Outputs clock with same frequency as bit rate
	1	0		External	Inputs clock with frequency 16 times bit rate
1	0	0	Clocked synchronous mode	Internal	Outputs serial clock
	1	0		External	Inputs serial clock
0	1	1	Reserved (Do not specify these combinations)		
1	0	1			
1	1	1			

10.4.2 SCI3 Initialization

Follow the flowchart as shown in figure 10.4 to initialize the SCI3. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the contents of the RDRF, PER, FER, and OER flags, or the contents of RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization. When the external clock is used in clock synchronous mode, the clock must not be supplied during initialization.



[1] Set the clock selection in SCR3. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock output is selected in asynchronous mode, clock is output immediately after CKE1 and CKE0 settings are made. When the clock output is selected at reception in clocked synchronous mode, clock is output immediately after CKE1, CKE0, and RE are set to 1.

[2] Set the data transfer format in SMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR3 to 1. Setting bits TE and RE enables the TXD32 and RXD32 pins to be used. Also set the RIE, TIE, TEIE, and MPIE bits, depending on whether interrupts are required. In asynchronous mode, the bits are marked at transmission and idled at reception to wait for the start bit.

Figure 10.4 Sample SCI3 Initialization Flowchart

10.4.3 Data Transmission

Figure 10.5 shows an example of operation for transmission in asynchronous mode. In transmission, the SCI3 operates as described below.

1. The SCI3 monitors the TDRE flag in SSR. If the flag is cleared to 0, the SCI3 recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI3 sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a TXI interrupt request is generated. Continuous transmission is possible because the TXI interrupt routine writes next transmit data to TDR before transmission of the current transmit data has been completed.
3. The SCI3 checks the TDRE flag at the timing for sending the stop bit.
4. If the TDRE flag is 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
5. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the "mark state" is entered, in which 1 is output. If the TEIE bit in SCR3 is set to 1 at this time, a TEI interrupt request is generated.
6. Figure 10.6 shows a sample flowchart for transmission in asynchronous mode.

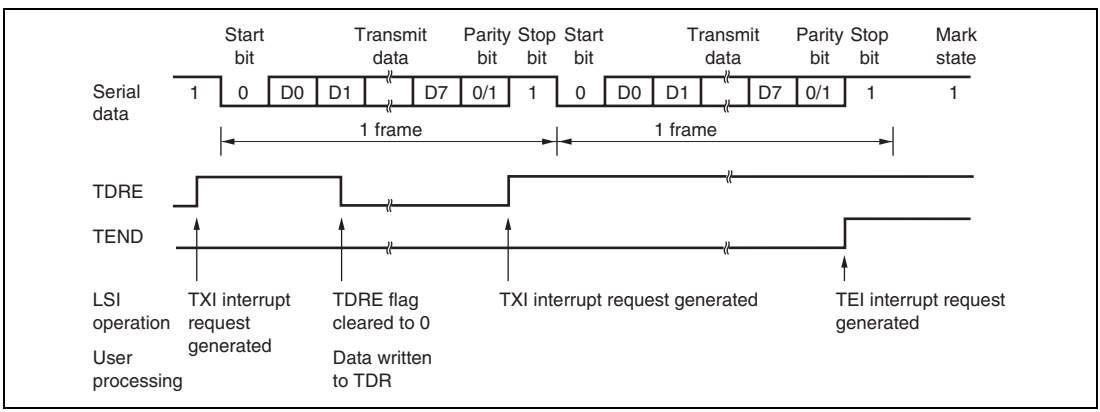
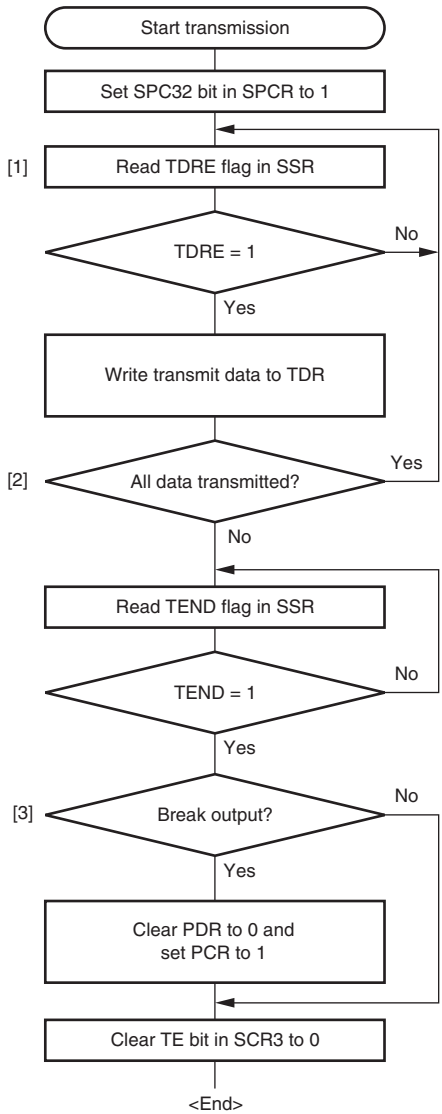


Figure 10.5 Example SCI3 Operation in Transmission in Asynchronous Mode (8-Bit Data, Parity, One Stop Bit)



[1] Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. When data is written to TDR, the TDRE flag is automaticaly cleared to 0. (After the TE bit is set to 1, one frame of 1 is output, then transmission is possible.)

[2] To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR. When data is written to TDR, the TDRE flag is automaticaly cleared to 0.

[3] To output a break in serial transmission, after setting PCR to 1 and PDR to 0, clear the TE bit in SCR3 to 0.

Figure 10.6 Sample Serial Transmission Flowchart (Asynchronous Mode)

10.4.4 Serial Data Reception

Figure 10.7 shows an example of operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI3 monitors the communication line. If a start bit is detected, the SCI3 performs internal synchronization, receives data in RSR, and checks the parity bit and stop bit.

- Parity check

  The SCI3 checks that the number of 1 bits in the receive data conforms to the parity (odd or even) set in bit PM in the serial mode register (SMR).
- Stop bit check

  The SCI3 checks that the stop bit is 1. If two stop bits are used, only the first is checked.
- Status check

  The SCI3 checks that bit RDRF is set to 0, indicating that the receive data can be transferred from RSR to RDR.

2. If an overrun error occurs (when reception of the next data is completed while the RDRF flag is still set to 1), the OER bit in SSR is set to 1. If the RIE bit in SCR3 is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR3 is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error is detected (when the stop bit is 0), the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR3 is set to 1 at this time, an ERI interrupt request is generated.
5. If reception is completed successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR3 is set to 1 at this time, an RXI interrupt request is generated. Continuous reception is possible because the RXI interrupt routine reads the receive data transferred to RDR before reception of the next receive data has been completed.

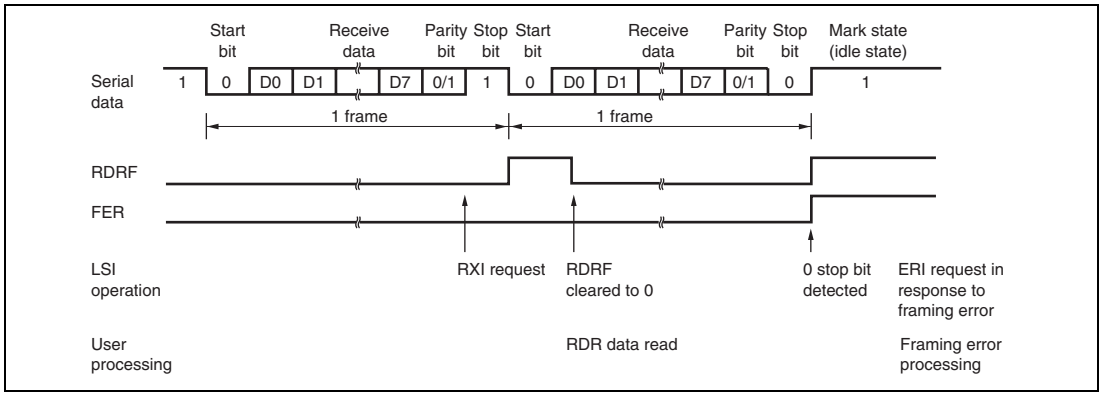


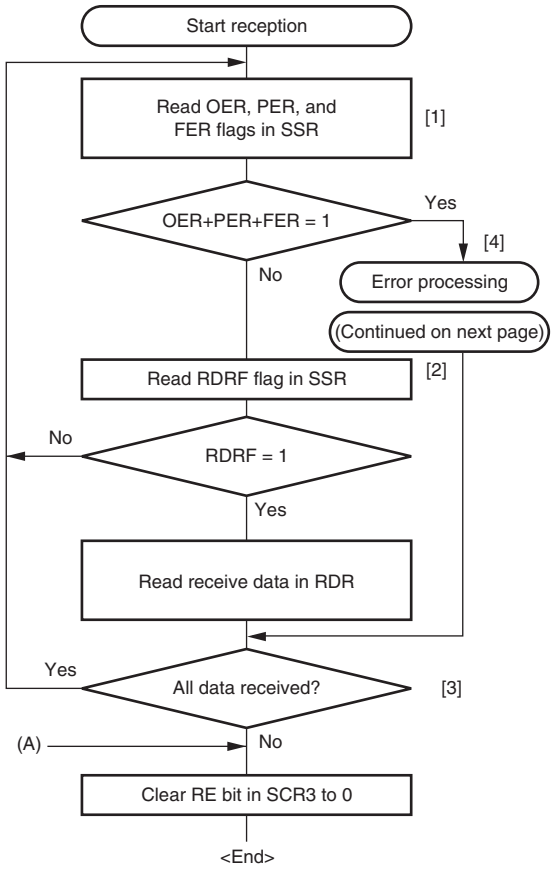
Figure 10.7 Example SCI3 Operation in Reception in Asynchronous Mode (8-Bit Data, Parity, One Stop Bit)

Table 10.10 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the OER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 10.8 shows a sample flowchart for serial data reception.

Table 10.10 SSR Status Flags and Receive Data Handling

SSR Status Flag					
RDRF*	**OER**	**FER**	**PER**	**Receive Data**	**Receive Error Type**
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

Note: * The RDRF flag retains the state it had before data reception. However, note that if RDR is read after an overrun error has occurred in a frame because reading of the receive data in the previous frame was delayed, the RDRF flag will be cleared to 0.



[1] Read the OER, PER, and FER flags in SSR to identify the error. If a receive error occurs, performs the appropriate error processing.

[2] Read SSR and check that RDRF = 1, then read the receive data in RDR. The RDRF flag is cleared automatically.

[3] To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag and read RDR.
The RDRF flag is cleared automatically.

[4] If a receive error occurs, read the OER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the OER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RXD32 pin.

Figure 10.8 Sample Serial Data Reception Flowchart (Asynchronous Mode) (1)

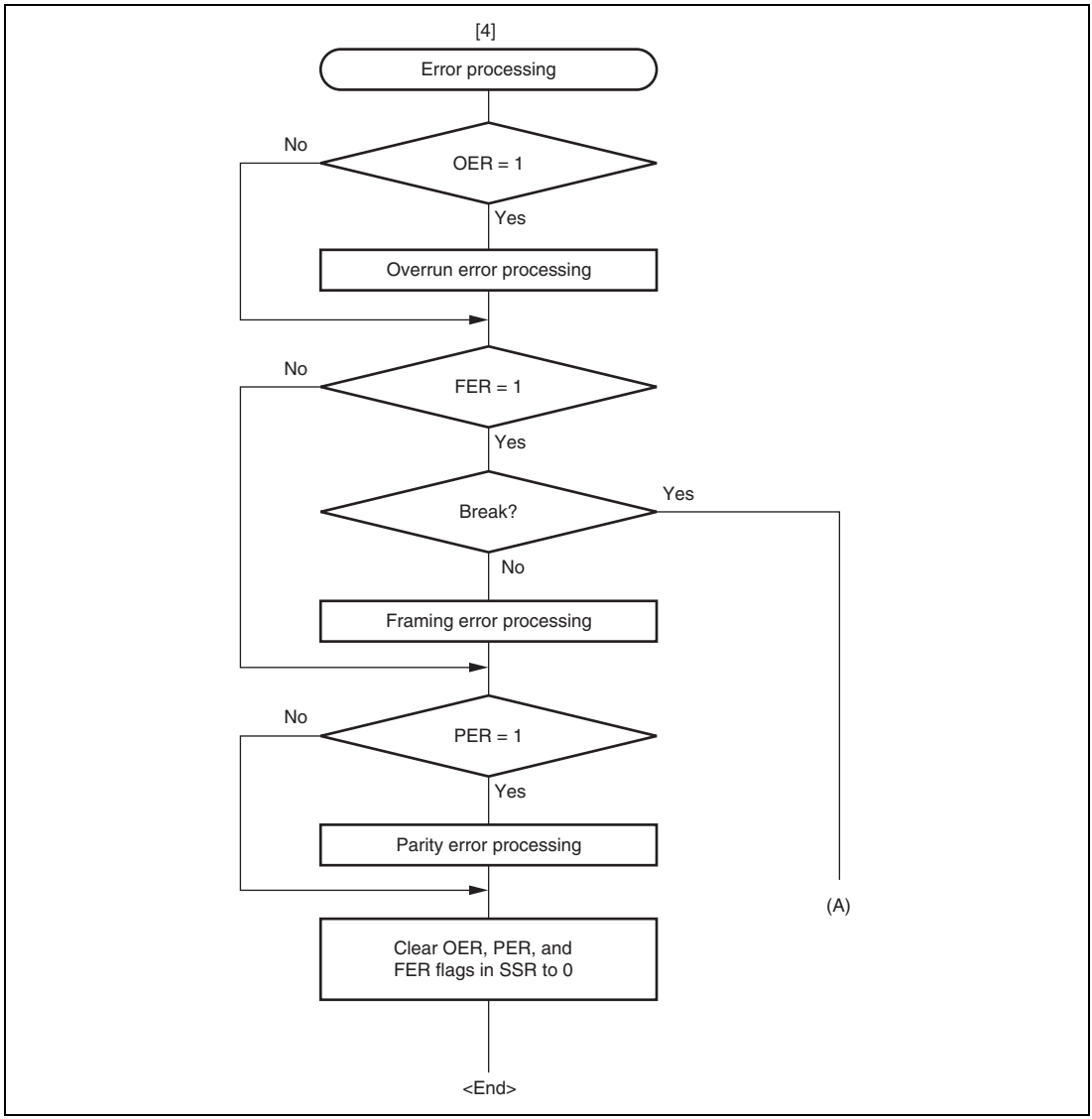


Figure 10.8 Sample Serial Data Reception Flowchart (Asynchronous Mode) (2)

10.5 Operation in Clocked Synchronous Mode

Figure 10.9 shows the general format for clock synchronous communication. In clock synchronous mode, data is transmitted or received synchronous with clock pulses. A single character in the transmit data consists of the 8-bit data starting from the LSB. In clock synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. In clock synchronous mode, the SCI3 receives data in synchronous with the rising edge of the serial clock. After 8-bit data is output, the transmission line holds the MSB state. In clock synchronous mode, no parity or multiprocessor bit is added. Inside the SCI3, the transmitter and receiver are independent units, enabling full-duplex communication through the use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so data can be read or written during transmission or reception, enabling continuous data transfer.

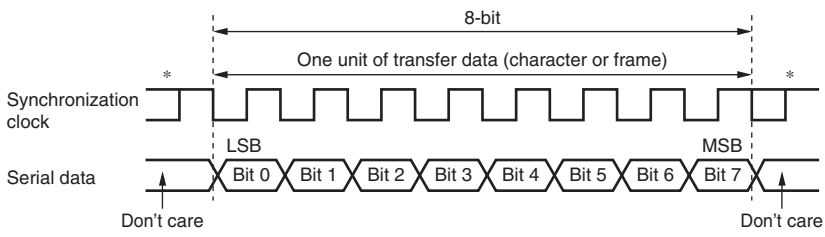


Figure 10.9 Data Format in Clocked Synchronous Communication

10.5.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK32 pin can be selected, according to the setting of the COM bit in SMR and CKE0 and CKE1 bits in SCR3. When the SCI3 is operated on an internal clock, the serial clock is output from the SCK32 pin. Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high.

10.5.2 SCI3 Initialization

Before transmitting and receiving data, the SCI3 should be initialized as described in a sample flowchart in figure 10.4.

10.5.3 Serial Data Transmission

Figure 10.10 shows an example of SCI3 operation for transmission in clock synchronous mode. In serial transmission, the SCI3 operates as described below.

1. The SCI3 monitors the TDRE flag in SSR, and if the flag is 0, the SCI recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. The SCI3 sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR3 is set to 1 at this time, a transmit data empty interrupt (TXI) is generated.
3. 8-bit data is sent from the TXD32 pin synchronized with the output clock when output clock mode has been specified, and synchronized with the input clock when use of an external clock has been specified. Serial data is transmitted sequentially from the LSB (bit 0), from the TXD32 pin.
4. The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).
5. If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TDRE flag maintains the output state of the last bit. If the TEIE bit in SCR3 is set to 1 at this time, a TEI interrupt request is generated.
7. The SCK32 pin is fixed high.

Figure 10.11 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (OER, FER, or PER) is set to 1. Make sure that the receive error flags are cleared to 0 before starting transmission.

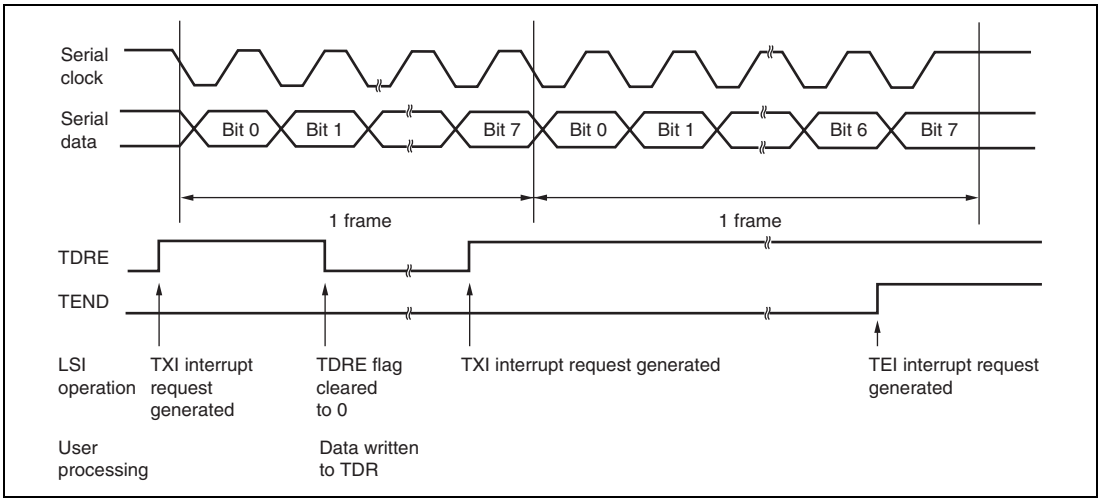
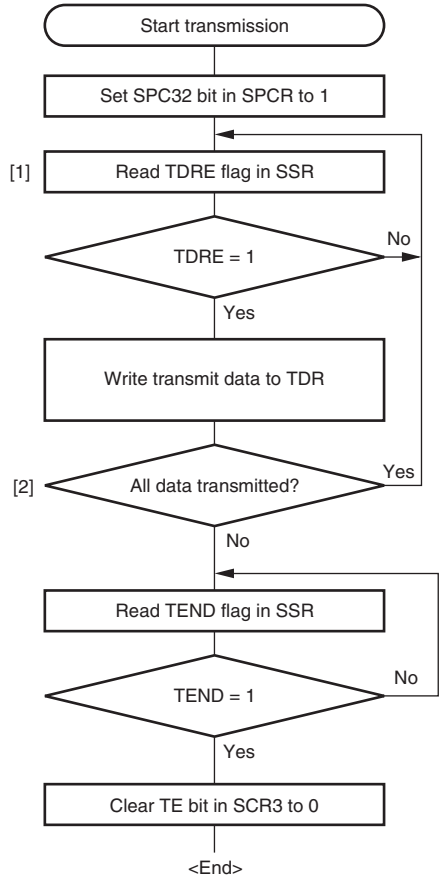


Figure 10.10 Example of SCI3 Operation in Transmission in Clocked Synchronous Mode



[1] Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. When data is written to TDR, the TDRE flag is automatically cleared to 0. When clock output is selected and data is written to TDR, clocks are output to start the data transmission.

[2] To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR. When data is written to TDR, the TDRE flag is automatically cleared to 0.

Figure 10.11 Sample Serial Transmission Flowchart (Clocked Synchronous Mode)

10.5.4 Serial Data Reception

Figure 10.12 shows an example of SCI3 operation for reception in clock synchronous mode. In serial reception, the SCI3 operates as described below.

1. The SCI3 performs internal initialization synchronous with a synchronous clock input or output, starts receiving data.
2. The SCI3 stores the received data in RSR.
3. If an overrun error occurs (when reception of the next data is completed while the RDRF flag in SSR is still set to 1), the OER bit in SSR is set to 1. If the RIE bit in SCR3 is set to 1 at this time, an ERI interrupt request is generated, receive data is not transferred to RDR, and the RDRF flag remains to be set to 1.
4. If reception is completed successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR3 is set to 1 at this time, an RXI interrupt request is generated.

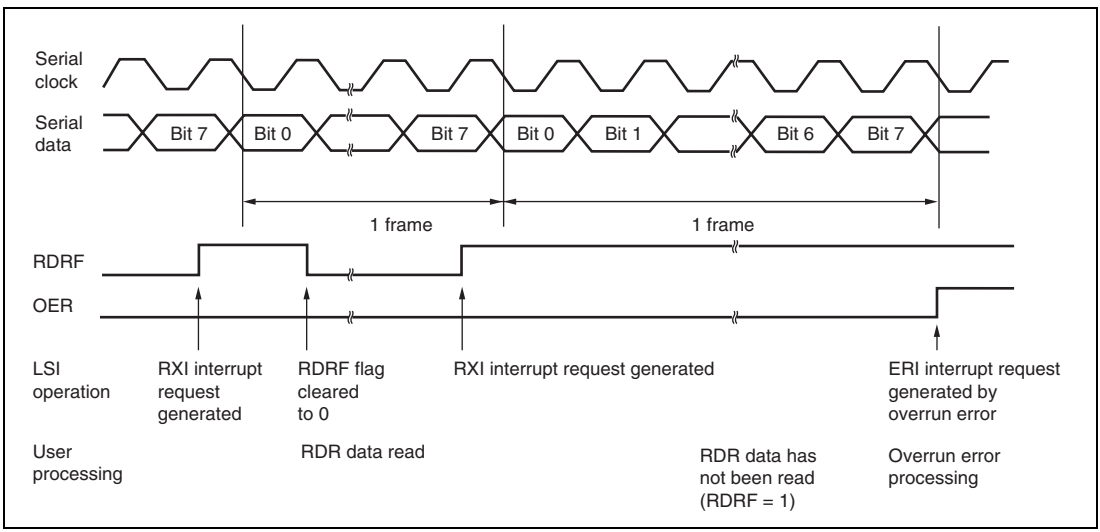
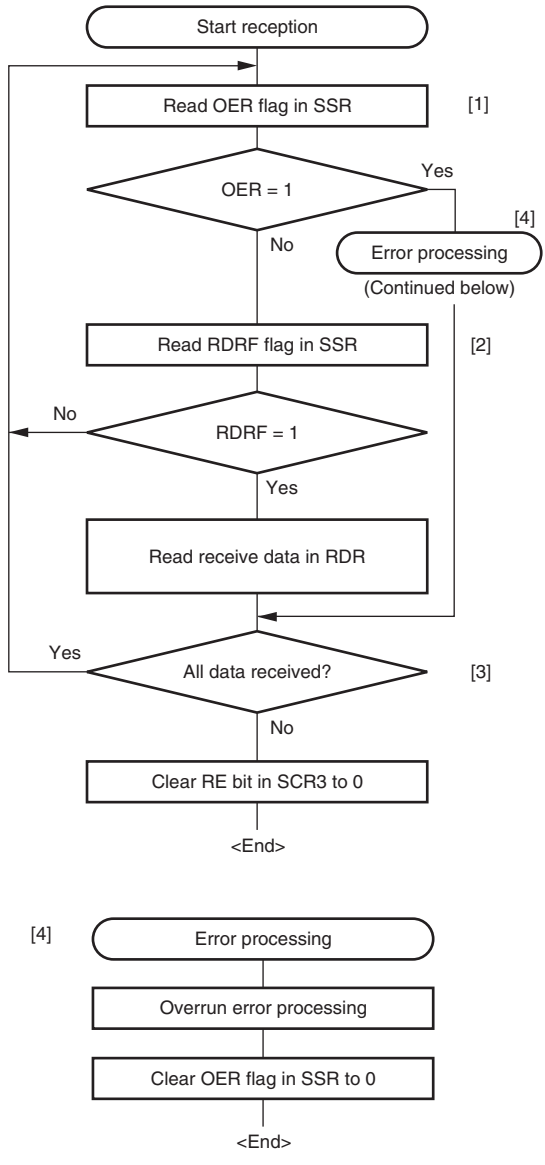


Figure 10.12 Example of SCI3 Reception Operation in Clocked Synchronous Mode

Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the OER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 10.13 shows a sample flowchart for serial data reception.



[1] Read the OER flag in SSR to determine if there is an error. If an overrun error has occurred, execute overrun error processing.

[2] Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR. When data is read from RDR, the RDRF flag is automatically cleared to 0.

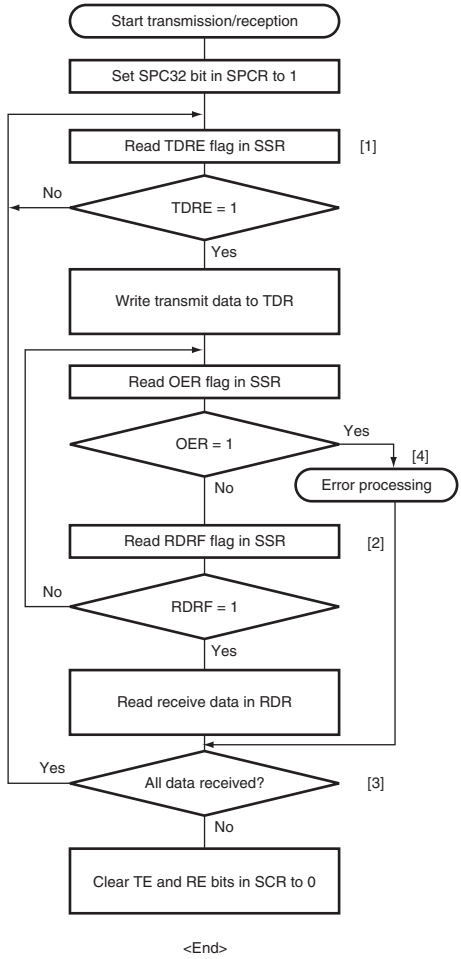
[3] To continue serial reception, before the MSB (bit 7) of the current frame is received, reading the RDRF flag and reading RDR should be finished. When data is read from RDR, the RDRF flag is automatically cleared to 0.

[4] If an overrun error occurs, read the OER flag in SSR, and after performing the appropriate error processing, clear the OER flag to 0. Reception cannot be resumed if the OER flag is set to 1.

Figure 10.13 Sample Serial Reception Flowchart (Clocked Synchronous Mode)

10.5.5 Simultaneous Serial Data Transmission and Reception

Figure 10.14 shows a sample flowchart for simultaneous serial transmit and receive operations. The following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI3 has finished transmission and the TDRE and TEND flags are set to 1, clear TE to 0. Then simultaneously set TE and RE to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI3 has finished reception, clear RE to 0. Then after checking that the RDRF and receive error flags (OER, FER, and PER) are cleared to 0, simultaneously set TE and RE to 1 with a single instruction.



[1] Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR.
When data is written to TDR, the TDRE flag is automatically cleared to 0.

[2] Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR.
When data is read from RDR, the RDRF flag is automatically cleared to 0.

[3] To continue serial transmission/ reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR.
When data is written to TDR, the TDRE flag is automatically cleared to 0. When data is read from RDR, the RDRF flag is automatically cleared to 0.

[4] If an overrun error occurs, read the OER flag in SSR, and after performing the appropriate error processing, clear the OER flag to 0. Transmission/reception cannot be resumed if the OER flag is set to 1.
For overrun error processing, see figure 10.13.

Figure 10.14 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations (Clocked Synchronous Mode)

10.6 Interrupts

The SCI3 creates the following six interrupt requests: transmission end, transmit data empty, receive data full, and receive errors (overrun error, framing error, and parity error). Table 10.11 shows the interrupt sources.

Table 10.11 SCI3 Interrupt Requests

Interrupt Requests	Abbreviation	Interrupt Sources	Enable Bit
Receive Data Full	RXI	Setting RDRF in SSR	RIE
Transmit Data Empty	TXI	Setting TDRE in SSR	TIE
Transmission End	TEI	Setting TEND in SSR	TEIE
Receive Error	ERI	Setting OER, FER, or PER in SSR	RIE

Each interrupt request can be enabled or disabled by means of bits TIE, RIE and TEIE in SCR3.

When bit TDRE is set to 1 in SSR, a TXI interrupt is requested. When bit TEND is set to 1 in SSR, a TEI interrupt is requested. These two interrupts are generated during transmission.

The initial value of the TDRE flag in SSR is 1. Thus, when the TIE bit in SCR3 is set to 1 before transferring the transmit data to TDR, a TXI interrupt request is generated even if the transmit data is not ready. The initial value of the TEND flag in SSR is 1. Thus, when the TEIE bit in SCR3 is set to 1 before transferring the transmit data to TDR, a TEI interrupt request is generated even if the transmit data has not been sent. It is possible to make use of the most of these interrupt requests efficiently by transferring the transmit data to TDR in the interrupt routine. To prevent the generation of these interrupt requests (TXI and TEI), set the enable bits (TIE and TEIE) that correspond to these interrupt requests to 1, after transferring the transmit data to TDR.

When bit RDRF is set to 1 in SSR, an RXI interrupt is requested, and if any of bits OER, PER, and FER is set to 1, an ERI interrupt is requested. These two interrupt requests are generated during reception.

For further details, see section 3, Exception Handling.

The SCI3 can carry out continuous reception using RXI and continuous transmission using TXI. These interrupts are shown in table 10.12.

Table 10.12 Transmit/Receive Interrupts

Interrupt	Flag and Enable Bit	Interrupt Request Conditions	Notes
RXI	RDRF RIE	When serial reception is performed normally and receive data is transferred from RSR to RDR, bit RDRF is set to 1, and if bit RIE is set to 1 at this time, RXI is enabled and an interrupt is requested. (See figure 10.15(a).)	The RXI interrupt routine reads the receive data transferred to RDR and clears bit RDRF to 0. Continuous reception can be performed by repeating the above operations until reception of the next RSR data is completed.
TXI	TDRE TIE	When TSR is found to be empty (on completion of the previous transmission) and the transmit data placed in TDR is transferred to TSR, bit TDRE is set to 1. If bit TIE is set to 1 at this time, TXI is enabled and an interrupt is requested. (See figure 10.15(b).)	The TXI interrupt routine writes the next transmit data to TDR and clears bit TDRE to 0. Continuous transmission can be performed by repeating the above operations until the data transferred to TSR has been transmitted.
TEI	TEND TEIE	When the last bit of the character in TSR is transmitted, if bit TDRE is set to 1, bit TEND is set to 1. If bit TEIE is set to 1 at this time, TEI is enabled and an interrupt is requested. (See figure 10.15(c).)	TEI indicates that the next transmit data has not been written to TDR when the last bit of the transmit character in TSR is transmitted.

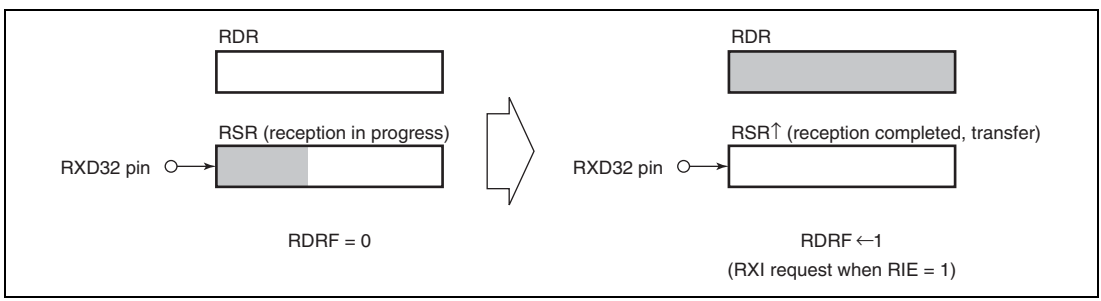


Figure 10.15(a) RDRF Setting and RXI Interrupt

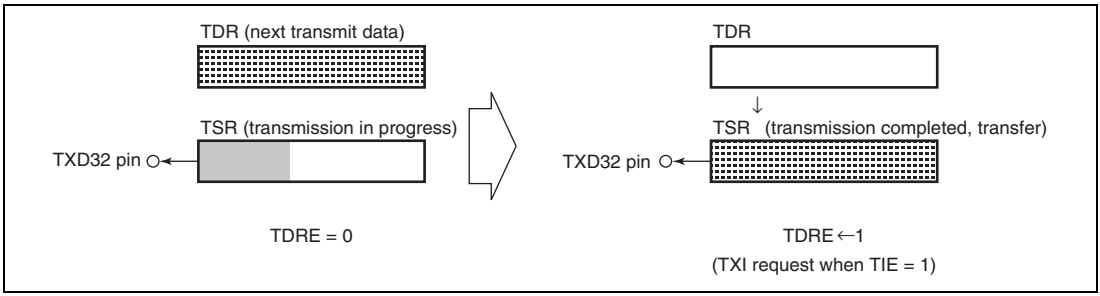


Figure 10.15(b) TDRE Setting and TXI Interrupt

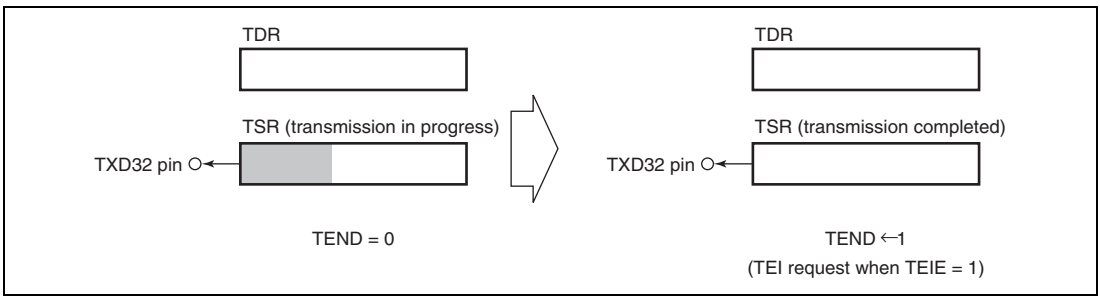


Figure 10.15(c) TEND Setting and TEI Interrupt

10.7 Usage Notes

10.7.1 Break Detection and Processing

When framing error detection is performed, a break can be detected by reading the RXD32 pin value directly. In a break, the input from the RXD32 pin becomes all 0, setting the FER flag, and possibly the PER flag. Note that as the SCI3 continues the receive operation after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

10.7.2 Mark State and Break Sending

When TE is 0, the TXD32 pin is used as an I/O port whose direction (input or output) and level are determined by PCR and PDR. This can be used to set the TXD32 pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line at mark state until TE is set to 1, set both PCR and PDR to 1. As TE is cleared to 0 at this point, the TXD32 pin becomes an I/O port, and 1 is output from the TXD32 pin. To send a break during serial transmission, first set PCR to 1 and PDR to 0, and then clear TE to 0. When TE is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TXD32 pin becomes an I/O port, and 0 is output from the TXD32 pin.

10.7.3 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only)

Transmission cannot be started when a receive error flag (OER, PER, or FER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

10.7.4 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI3 operates on a basic clock with a frequency of 16 times the transfer rate. In reception, the SCI3 samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the basic clock as shown in figure 10.16.

Thus, the reception margin in asynchronous mode is given by formula (1) below.

$$M = \left\{ (0.5 - \frac{1}{2N}) - \frac{D - 0.5}{N} - (L - 0.5)\,F \right\} \bullet 100(\%)$$

... Formula (1)

Where N : Ratio of bit rate to clock (N = 16)
D : Clock duty (D = 0.5 to 1.0)
L : Frame length (L = 9 to 12)
F : Absolute value of clock rate deviation

Assuming values of F (absolute value of clock rate deviation) = 0 and D (clock duty) = 0.5 in formula (1), the reception margin can be given by the formula.

M = {0.5 – 1/(2 16)} × 100 [%] = 46.875%

However, this is only the computed value, and a margin of 20% to 30% should be allowed for in system design.

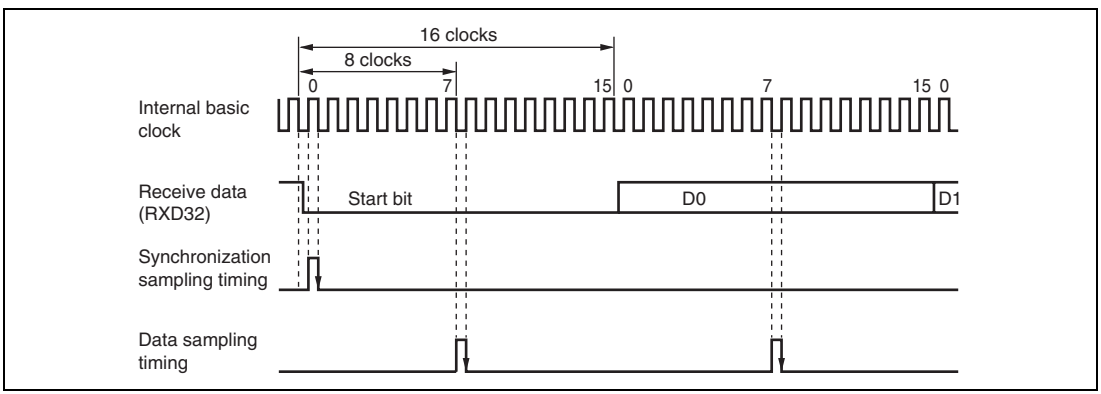


Figure 10.16 Receive Data Sampling Timing in Asynchronous Mode

10.7.5 Note on Switching SCK32 Function

If pin SCK32 is used as a clock output pin by the SCI3 in clock synchronous mode and is then switched to a general input/output pin (a pin with a different function), the pin outputs a low level signal for half a system clock (ϕ) cycle immediately after it is switched.

This can be prevented by either of the following methods according to the situation.

a. When an SCK32 function is switched from clock output to non clock-output

   When stopping data transfer, issue one instruction to clear bits TE and RE to 0 and to set bits CKE1 and CKE0 in SCR3 to 1 and 0, respectively.

   In this case, bit COM in SMR should be left 1. The above prevents SCK32 from being used as a general input/output pin. To avoid an intermediate level of voltage from being applied to SCK32, the line connected to SCK32 should be pulled up to the V_{CC} level via a resistor, or supplied with output from an external device.

b. When an SCK32 function is switched from clock output to general input/output

   When stopping data transfer,

   (i) Issue one instruction to clear bits TE and RE to 0 and to set bits CKE1 and CKE0 in SCR3 to 1 and 0, respectively.

   (ii) Clear bit COM in SMR to 0

   (iii) Clear bits CKE1 and CKE0 in SCR3 to 0

   Note that special care is also needed here to avoid an intermediate level of voltage from being applied to SCK32.

10.7.6 Relation between Writing to TDR and Bit TDRE

Bit TDRE in the serial status register (SSR) is a status flag that indicates that data for serial transmission has not been prepared in TDR. When data is written to TDR, bit TDRE is cleared to 0 automatically. When the SCI3 transfers data from TDR to TSR, bit TDRE is set to 1.

Data can be written to TDR irrespective of the state of bit TDRE, but if new data is written to TDR while bit TDRE is cleared to 0, the data previously stored in TDR will be lost if it has not yet been transferred to TSR. Accordingly, to ensure that serial transmission is performed dependably, you should first check that bit TDRE is set to 1, then write the transmit data to TDR only once (not two or more times).

10.7.7 Relation between RDR Reading and bit RDRF

In a receive operation, the SCI3 continually checks the RDRF flag. If bit RDRF is cleared to 0 when reception of one frame ends, normal data reception is completed. If bit RDRF is set to 1, this indicates that an overrun error has occurred.

When the contents of RDR are read, bit RDRF is cleared to 0 automatically. Therefore, if RDR is read more than once, the second and subsequent read operations will be performed while bit RDRF is cleared to 0. Note that, when an RDR read is performed while bit RDRF is cleared to 0, if the read operation coincides with completion of reception of a frame, the next frame of data may be read. This is shown in figure 10.17.

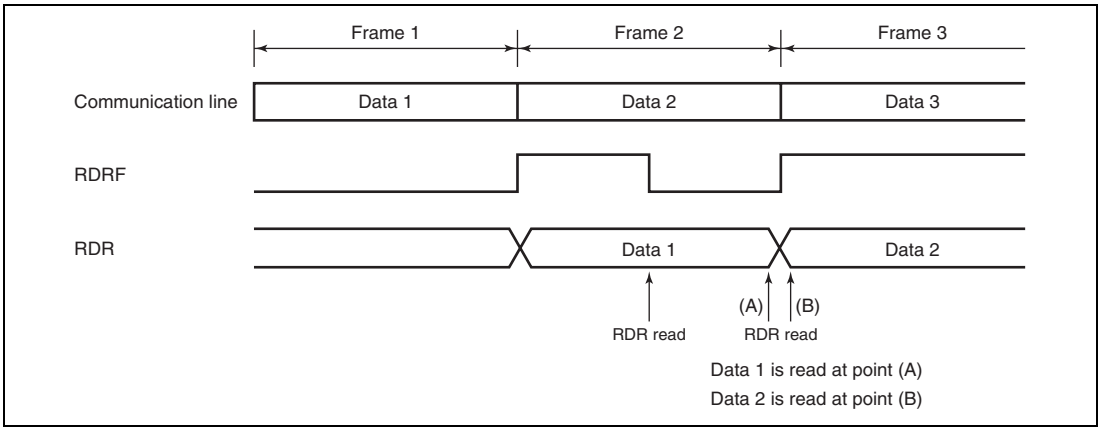


Figure 10.17 Relation between RDR Read Timing and Data

In this case, only a single RDR read operation (not two or more) should be performed after first checking that bit RDRF is set to 1. If two or more reads are performed, the data read the first time should be transferred to RAM, etc., and the RAM contents used. Also, ensure that there is sufficient margin in an RDR read operation before reception of the next frame is completed. To be precise in terms of timing, the RDR read should be completed before bit 7 is transferred in clock synchronous mode, or before the STOP bit is transferred in asynchronous mode.

10.7.8 Transmit and Receive Operations when Making State Transition

Make sure that transmit and receive operations have completely finished before carrying out state transition processing.

10.7.9 Setting in Subactive or Subsleep Mode

In subactive or subsleep mode, the SCI3 can operate only when the CPU clock is $\phi_w/2$. The SA1 bit in SYSCR2 should be set to 1.

Section 11 10-Bit PWM

This LSI has a two-channel 10-bit PWM. The PWM with a low-path filter connected can be used as a D/A converter. Figure 11.1 shows a block diagram of the 10-bit PWM.

11.1 Features

- Choice of four conversion periods

  A conversion period of 4096/ϕ with a minimum modulation width of 4/ϕ, a conversion period of 2048/ϕ with a minimum modulation width of 2/ϕ, a conversion period of 1024/ϕ with a minimum modulation width of 1/ϕ, or a conversion period of 512/ϕ with a minimum modulation width of 1/2ϕ can be selected.
- Pulse division method for less ripple
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (For details, refer to section 5.4, Module Standby Function.)

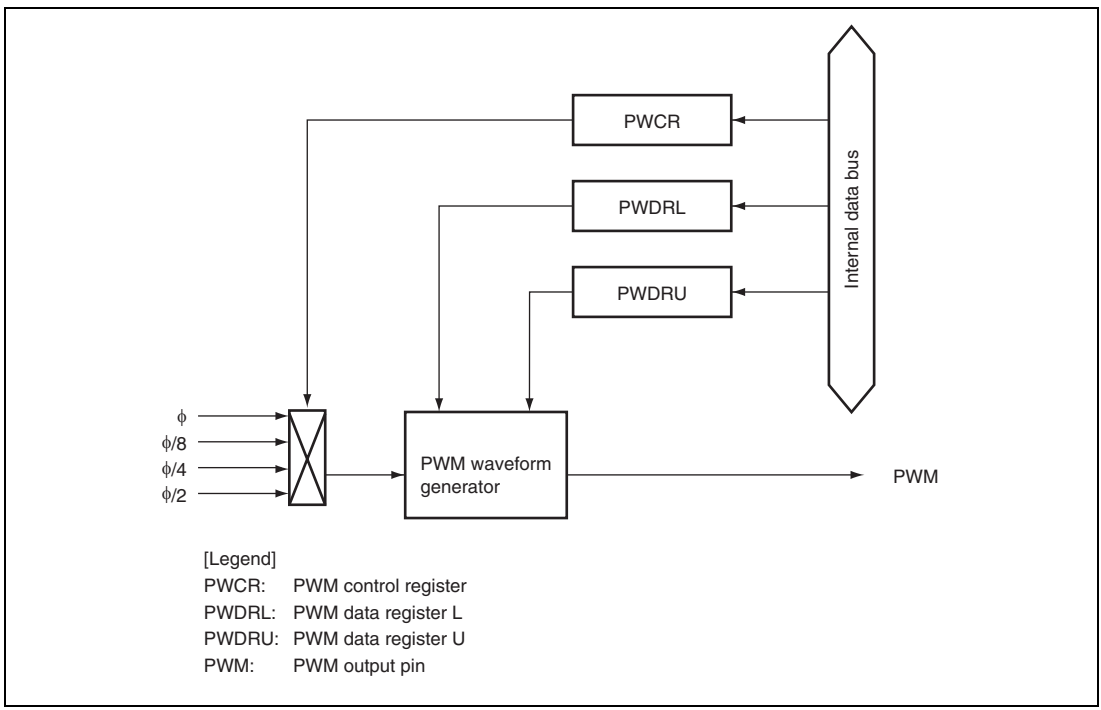


Figure 11.1 Block Diagram of 10-Bit PWM

11.2 Input/Output Pins

Table 11.1 shows the 10-bit PWM pin configuration.

Table 11.1 Pin Configuration

Name	Abbreviation	I/O	Function
10-bit PWM square-wave output 1	PWM1	Output	Channel 1: 10-bit PWM waveform output pin/event counter PWM output pin
10-bit PWM square-wave output 2	PWM2	Output	Channel 2: 10-bit PWM waveform output pin/event counter PWM output pin

11.3 Register Descriptions

The 10-bit PWM has the following registers.

- PWM control register (PWCR)
- PWM data register U (PWDRU)
- PWM data register L (PWDRL)

11.3.1 PWM Control Register (PWCR)

PWCR selects the conversion period.

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	—	Reserved
6	—	1	—	These bits are always read as 1, and cannot be modified.
5	—	1	—	
4	—	1	—	
3	—	1	—	
2	—	1	—	
1	PWCR1	0	W	Clock Select 1, 0
0	PWCR0	0	W	00: The input clock is ϕ ($t\phi = 1/\phi$) — The conversion period is $512/\phi$, with a minimum modulation width of $1/2\phi$ 01: The input clock is $\phi/2$ ($t\phi = 2/\phi$) — The conversion period is $1024/\phi$, with a minimum modulation width of $1/\phi$ 10: The input clock is $\phi/4$ ($t\phi = 4/\phi$) — The conversion period is $2048/\phi$, with a minimum modulation width of $2/\phi$ 11: The input clock is $\phi/8$ ($t\phi = 8/\phi$) — The conversion period is $4096/\phi$, with a minimum modulation width of $4/\phi$

[Legend] $t\phi$: Period of PWM clock input

11.3.2 PWM Data Registers U and L (PWDRU, PWDRL)

PWDRU and PWDRL indicate high level width in one PWM waveform cycle. PWDRU and PWDRL are 10-bit write-only registers, with the upper 2 bits assigned to PWDRU and the lower 8 bits to PWDRL. When read, all bits are always read as 1.

Both PWDRU and PWDRL are accessible only in bytes. Note that the operation is not guaranteed if word access is performed. When 10-bit data is written in PWDRU and PWDRL, the contents are latched in the PWM waveform generator and the PWM waveform generation data is updated. When writing the 10-bit data, the order is as follows: PWDRL to PWDRU.

PWDRU and PWDRL are initialized to H'FC00.

11.4 Operation

11.4.1 Operation

When using the 10-bit PWM, set the registers in this sequence:

1. Set the PWM2 and/or PWM1 bits in port mode register 9 (PMR9) to 1 to set the P91/PWM2 pin or P90/PWM1 pin, or both, to function as PWM output pins.
2. Set the PWCR0 and PWCR1 bits in PWCR to select one conversion period of either.
3. Set the output waveform data in PWDRU and PWDRL. Be sure to write byte data first to PWDRL and then to PWDRU. When the data is written in PWDRU, the contents of these registers are latched in the PWM waveform generator, and the PWM waveform generation data is updated in synchronization with internal signals.

One conversion period consists of four pulses, as shown in figure 11.2. The total high-level width during this period (T_H) corresponds to the data in PWDRU and PWDRL. This relation can be expressed as follows:

$$T_H = (\text{data value in PWDRU and PWDRL} + 4) \times t\phi/2$$

where tϕ is the period of PWM clock input: 1/ϕ (PWCR1 = 0, PWCR0 = 0), 2/ϕ (PWCR1 = 0, PWCR0 = 1), 4/ϕ (PWCR1 = 1, PWCR0 = 0), or 8/ϕ (PWCR1 = 1, PWCR0 = 1).
If the data value in PWDRU and PWDRL is from H'FFFC to H'FFFF, the PWM output stays high. When the data value is H'FC3C, T_H is calculated as follows:

$$T_H = 64 \times t\phi/2 = 32 \times t\phi$$

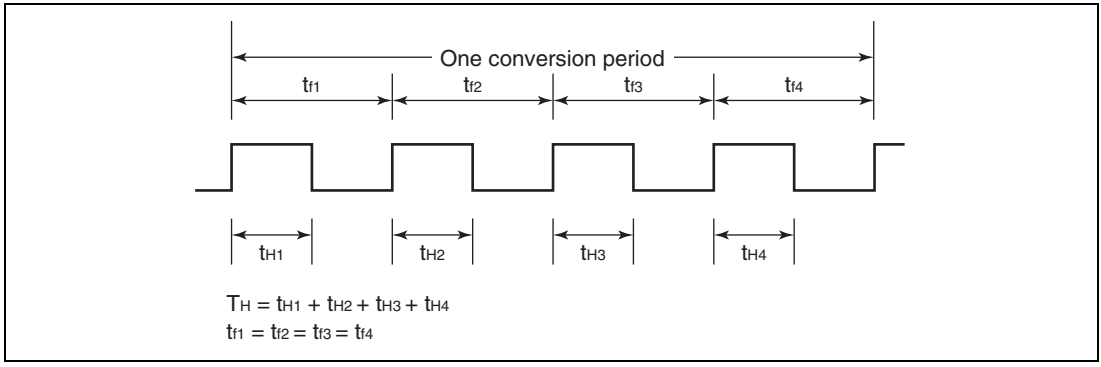


Figure 11.2 Waveform Output by 10-Bit PWM

11.4.2 PWM Operating States

Table 11.2 shows the PWM operating states.

Table 11.2 PWM Operating States

Operating Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby	Module Standby
PWCR	Reset	Functions	Functions	Retained	Retained	Retained	Retained	Retained
PWDRU	Reset	Functions	Functions	Retained	Retained	Retained	Retained	Retained
PWDRL	Reset	Functions	Functions	Retained	Retained	Retained	Retained	Retained

Section 12 A/D Converter

This LSI includes a successive approximation type 10-bit A/D converter that allows up to four analog input channels to be selected. The block diagram of the A/D converter is shown in figure 12.1.

12.1 Features

- 10-bit resolution
- Four input channels
- Conversion time: at least 12.4 μs per channel (ϕ = 5 MHz operation)
- Sample and hold function
- Conversion start method
  - Software
- Interrupt request
  - An A/D conversion end interrupt request (ADI) can be generated
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (For details, refer to section 5.4, Module Standby Function.)

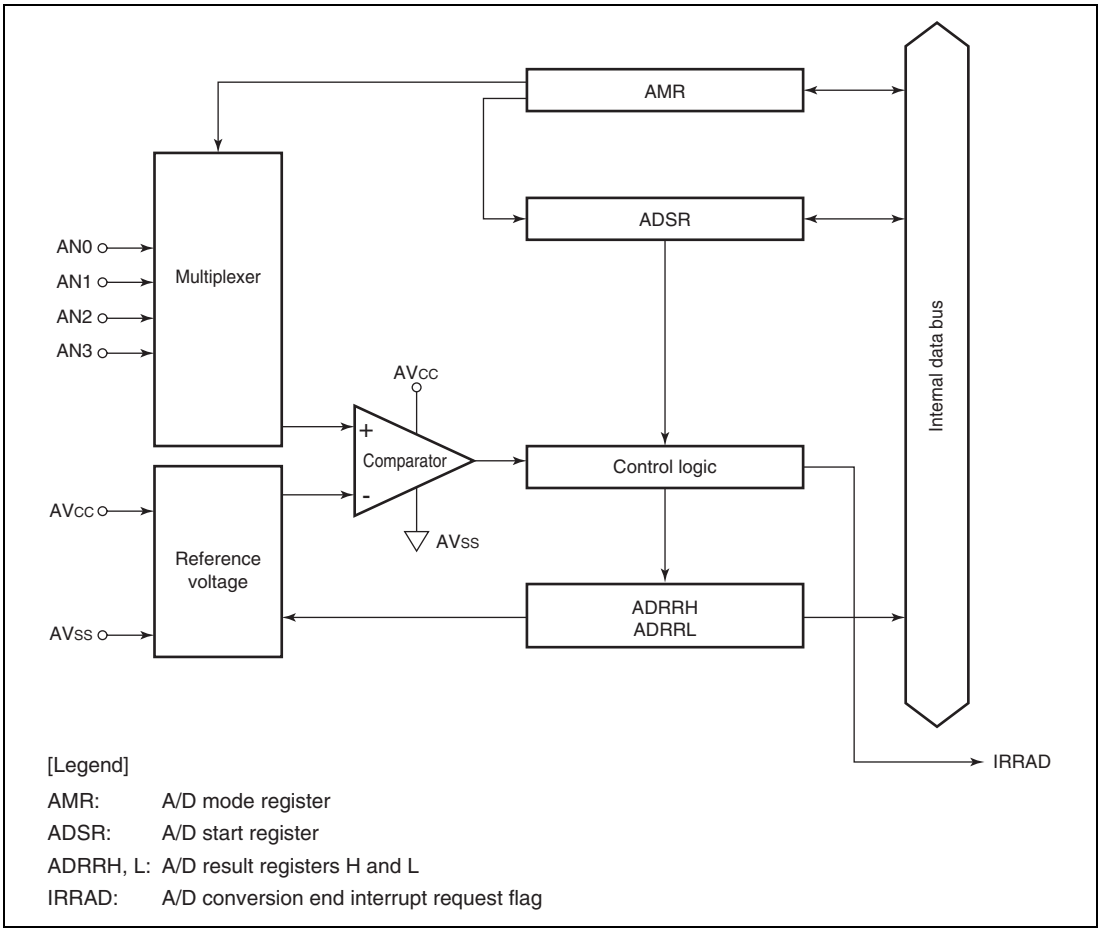


Figure 12.1 Block Diagram of A/D Converter

12.2 Input/Output Pins

Table 12.1 shows the input pins used by the A/D converter.

Table 12.1 Pin Configuration

Pin Name	Abbreviation	I/O	Function
Analog power supply pin	AVcc	Input	Power supply and reference voltage of analog part
Analog ground pin	AVss	Input	Ground and reference voltage of analog part
Analog input pin 0	AN0	Input	Analog input pins
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin 3	AN3	Input	

12.3 Register Descriptions

The A/D converter has the following registers.

- A/D result registers H and L (ADRRH and ADRRL)
- A/D mode register (AMR)
- A/D start register (ADSR)

12.3.1 A/D Result Registers H and L (ADRRH and ADRRL)

ADRRH and ADRRL are 16-bit read-only registers that store the results of A/D conversion.

The upper 8 bits of the data are stored in ADRRH, and the lower 2 bits in ADRRL.

ADRRH and ADRRL can be read by the CPU at any time, but the ADRRH and ADRRL values during A/D conversion are undefined. After A/D conversion is completed, the conversion result is stored as 10-bit data, and this data is retained until the next conversion operation starts.

The initial values of ADRRH and ADRRL are undefined.

12.3.2 A/D Mode Register (AMR)

AMR sets the A/D conversion time and analog input pins.

Bit	Bit Name	Initial Value	R/W	Description
7	CKS	0	R/W	Clock Select Sets the A/D conversion time. 0: Conversion time = 62 states 1: Conversion time = 31 states
6	—	0	R/W	Reserved Only 0 can be written to this bit.
5	—	1	—	Reserved
4	—	1	—	These bits are always read as 1 and cannot be modified.
3	CH3	0	R/W	Channel Select 3 to 0
2	CH2	0	R/W	Selects the analog input channel.
1	CH1	0	R/W	00XX: No channel selected
0	CH0	0	R/W	0100: AN0 0101: AN1 0110: AN2 0111: AN3 1XXX: Using prohibited The channel selection should be made while the ADSF bit is cleared to 0.

[Legend] X: Don't care.

12.3.3 A/D Start Register (ADSR)

ADSR starts and stops the A/D conversion.

Bit	Bit Name	Initial Value	R/W	Description
7	ADSF	0	R/W	When this bit is set to 1, A/D conversion is started. When conversion is completed, the converted data is set in ADRRH and ADRRL and at the same time this bit is cleared to 0. If this bit is written to 0, A/D conversion can be forcibly terminated.
6 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.

12.4 Operation

The A/D converter operates by successive approximation with 10-bit resolution. When changing the conversion time or analog input channel, in order to prevent incorrect operation, first clear the bit ADSF to 0 in ADSR.

12.4.1 A/D Conversion

1. A/D conversion is started from the selected channel when the ADSF bit in ADSR is set to 1, according to software.
2. When A/D conversion is completed, the result is transferred to the A/D result register.
3. On completion of conversion, the IRRAD flag in IRR2 is set to 1. If the IENAD bit in IENR2 is set to 1 at this time, an A/D conversion end interrupt request is generated.
4. The ADSF bit remains set to 1 during A/D conversion. When A/D conversion ends, the ADSF bit is automatically cleared to 0 and the A/D converter enters the wait state.

12.4.2 Operating States of A/D Converter

Table 12.2 shows the operating states of the A/D converter.

Table 12.2 Operating States of A/D Converter

Operating Mode	Reset	Active	Sleep	Watch	Sub-active	Sub-sleep	Standby	Module Standby
AMR	Reset	Functions	Functions	Retained	Retained	Retained	Retained	Retained
ADSR	Reset	Functions	Functions	Reset	Reset	Reset	Reset	Reset
ADRRH	Retained*	Functions	Functions	Retained	Retained	Retained	Retained	Retained
ADRRL	Retained*	Functions	Functions	Retained	Retained	Retained	Retained	Retained

Note: * Undefined in a power-on reset.

12.5 Example of Use

An example of how the A/D converter can be used is given below, using channel 1 (pin AN1) as the analog input channel. Figure 12.2 shows the operation timing.

1. Bits CH3 to CH0 in the A/D mode register (AMR) are set to 0101, making pin AN1 the analog input channel. A/D interrupts are enabled by setting bit IENAD to 1, and A/D conversion is started by setting bit ADSF to 1.
2. When A/D conversion is completed, bit IRRAD is set to 1, and the A/D conversion result is stored in ADRRH and ADRRL. At the same time bit ADSF is cleared to 0, and the A/D converter goes to the idle state.
3. Bit IENAD = 1, so an A/D conversion end interrupt is requested.
4. The A/D interrupt handling routine starts.
5. The A/D conversion result is read and processed.
6. The A/D interrupt handling routine ends.

If bit ADSF is set to 1 again afterward, A/D conversion starts and steps 2 through 6 take place.

Figures 12.3 and 12.4 show flowcharts of procedures for using the A/D converter.

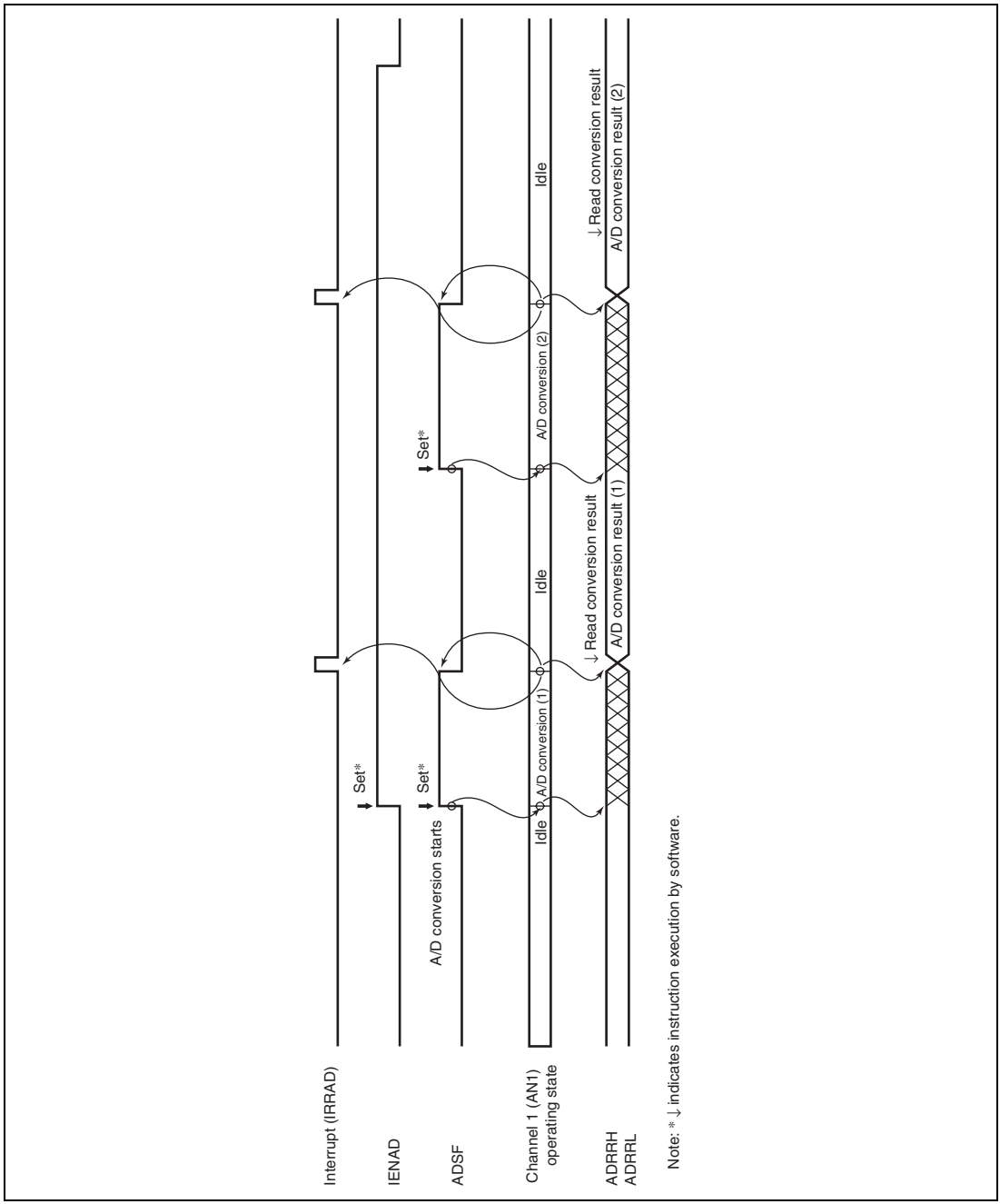


Figure 12.2 Example of A/D Conversion Operation

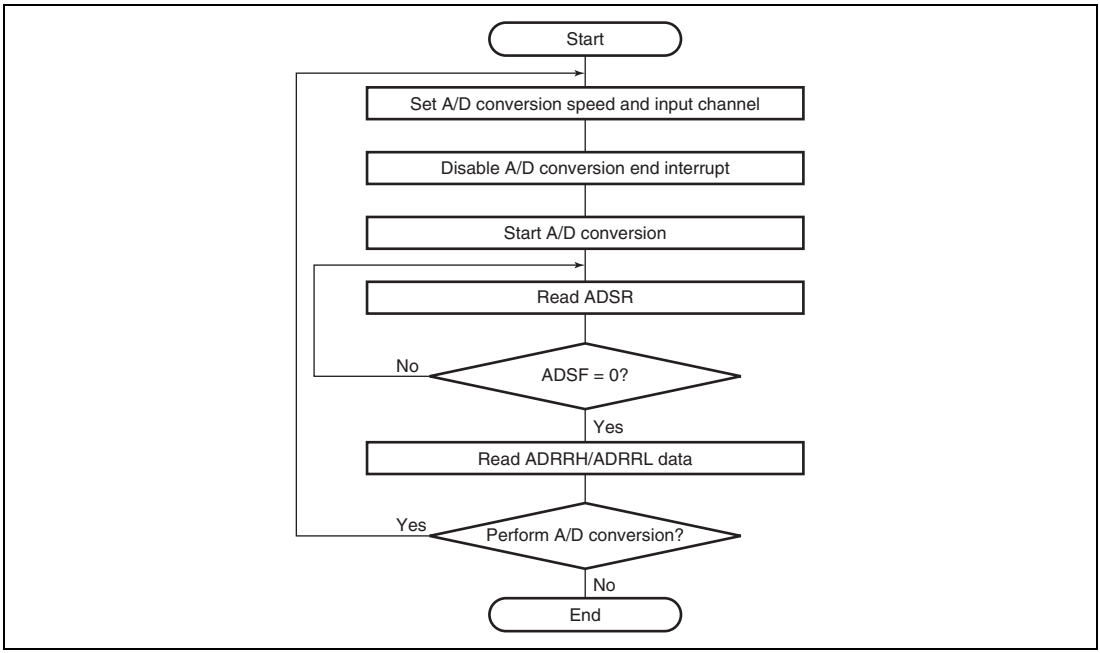


Figure 12.3 Flowchart of Procedure for Using A/D Converter (Polling by Software)

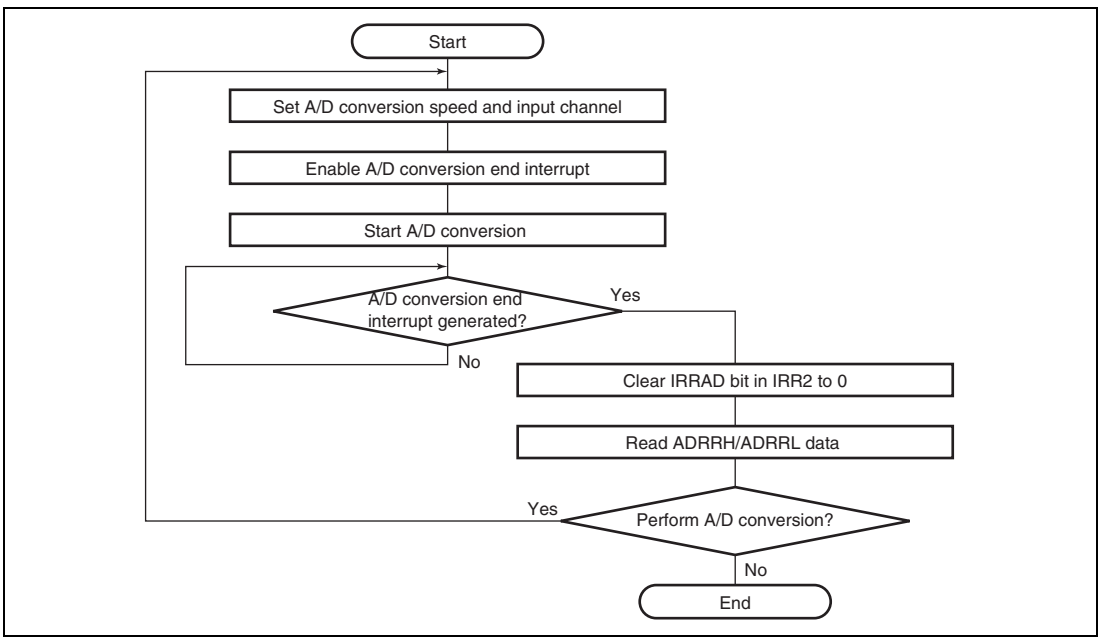


Figure 12.4 Flowchart of Procedure for Using A/D Converter (Interrupts Used)

12.6 A/D Conversion Accuracy Definitions

This LSI's A/D conversion accuracy definitions are given below.

- Resolution

  The number of A/D converter digital output codes
- Quantization error

  The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 12.5).
- Offset error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value 0000000000 to 0000000001 (see figure 12.6).
- Full-scale error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from 1111111110 to 1111111111 (see figure 12.6).
- Nonlinearity error

  The error with respect to the ideal A/D conversion characteristics between zero voltage and full-scale voltage. Does not include offset error, full-scale error, or quantization error.
- Absolute accuracy

  The deviation between the digital value and the analog input value. Includes offset error, full-scale error, quantization error, and nonlinearity error.

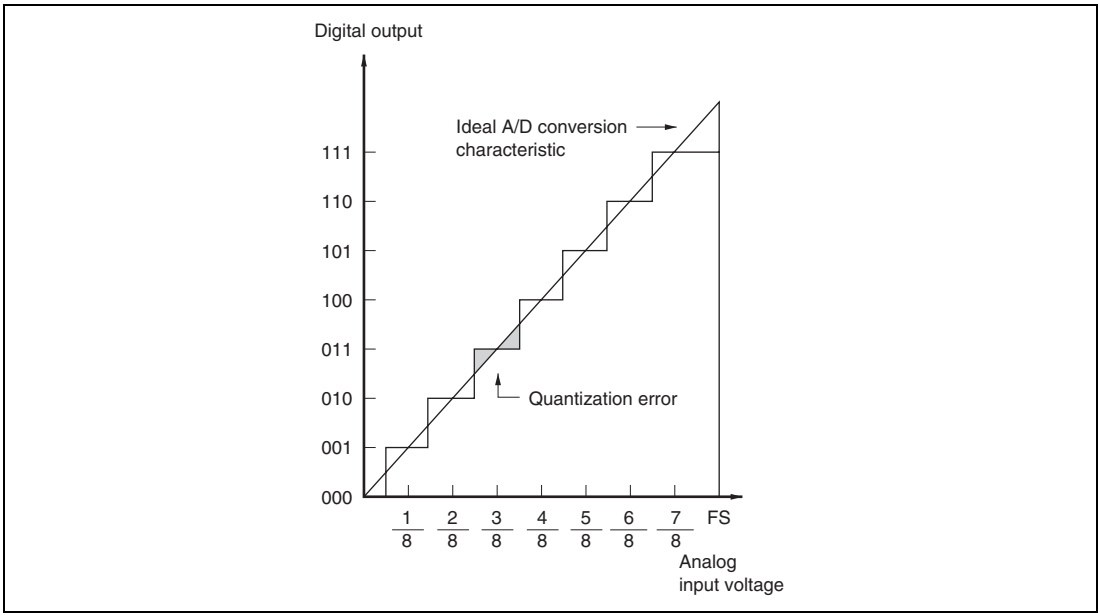


Figure 12.5 A/D Conversion Accuracy Definitions (1)

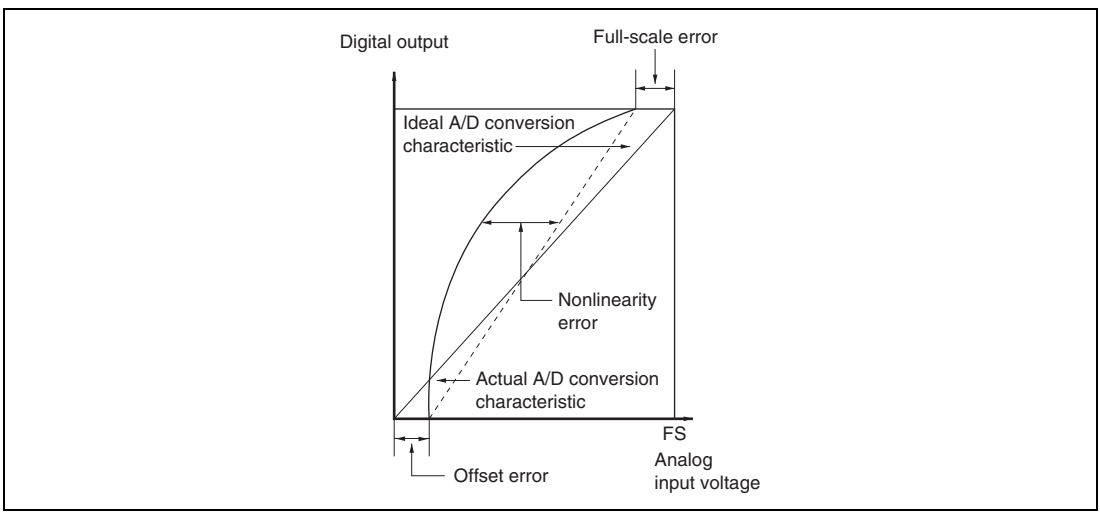


Figure 12.6 A/D Conversion Accuracy Definitions (2)

12.7 Usage Notes

12.7.1 Permissible Signal Source Impedance

This LSI's analog input is designed such that conversion accuracy is guaranteed for an input signal for which the signal source impedance is 10 kΩ or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 10 kΩ, charging may be insufficient and it may not be possible to guarantee A/D conversion accuracy.

As a countermeasure, a large capacitance can be provided externally to the analog input pin. This will cause the actual input resistance to comprise only the internal input resistance of 10 k , the signal source impedance does not need to be taken into consideration. This countermeasure has the disadvantage of creating a low-pass filter from the signal source impedance and capacitance, with the result that it may not be possible to follow analog signals having a large differential coefficient (e.g., 5 mV/μs or greater) (see figure 12.7). When converting a high-speed analog signal, a low-impedance buffer should be inserted.

12.7.2 Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute accuracy. Be sure to make the connection to an electrically stable GND.

Care is also required to ensure that filter circuits do not interfere with digital signals or act as antennas on the mounting board.

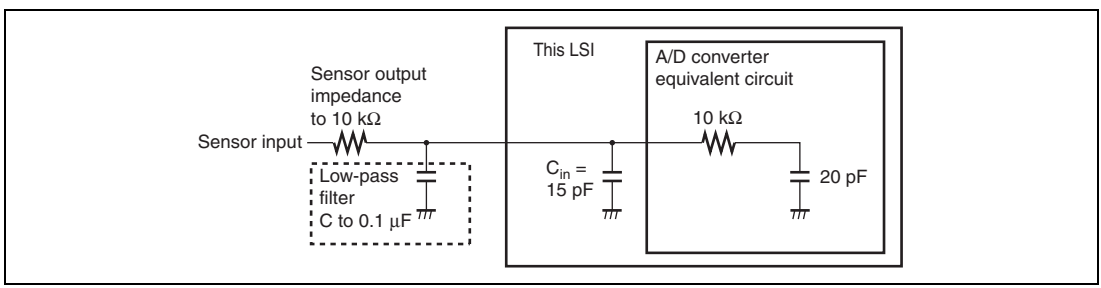


Figure 12.7 Example of Analog Input Circuit

12.7.3 Additional Usage Notes

1. ADRRH and ADRRL should be read only when the ADSF bit in ADSR is cleared to 0.
2. Changing the digital input signal at an adjacent pin during A/D conversion may adversely affect conversion accuracy.
3. When A/D conversion is started after clearing module standby mode, wait for 10ϕ clock cycles before starting A/D conversion.
4. In active mode and sleep mode, the analog power supply current flows in the ladder resistance even when the A/D converter is on standby. Therefore, if the A/D converter is not used, it is recommended that AVcc be connected to the system power supply and the ADCKSTP bit be cleared to 0 in CKSTPR1.

Section 13 List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register addresses (address order)
- Registers are listed from the lower allocation addresses.
- Registers are classified by functional modules.
- The data bus width is indicated.
- The number of access states is indicated.

2. Register bits
- Bit configurations of the registers are described in the same order as the register addresses.
- Reserved bits are indicated by — in the bit name column.
- When registers consist of 16 bits, bits are described from the MSB side.

3. Register states in each operating mode
- Register states are described in the same order as the register addresses.
- The register states described here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

13.1 Register Addresses (Address Order)

The data bus width indicates the numbers of bits by which the register is accessed.

The number of access states indicates the number of states based on the specified reference clock.

Register Name	Abbre-viation	Bit No	Address	Module Name	Data Bus Width	Access State
Flash memory control register 1	FLMCR1	8	H'F020	ROM	8	2
Flash memory control register 2	FLMCR2	8	H'F021	ROM	8	2
Flash memory power control register	FLPWCR	8	H'F022	ROM	8	2
Erase block register	EBR	8	H'F023	ROM	8	2
Flash memory enable register	FENR	8	H'F02B	ROM	8	2
Event counter PWM compare register H	ECPWCRH	8	H'FF8C	AEC*1	8	2
Event counter PWM compare register L	ECPWCRL	8	H'FF8D	AEC*1	8	2
Event counter PWM data register H	ECPWDRH	8	H'FF8E	AEC*1	8	2
Event counter PWM data register L	ECPWDRL	8	H'FF8F	AEC*1	8	2
Wakeup edge select register	WEGR	8	H'FF90	Interrupts	8	2
Serial port control register	SPCR	8	H'FF91	SCI3	8	2
Input pin edge select register	AEGSR	8	H'FF92	AEC*1	8	2
Event counter control register	ECCR	8	H'FF94	AEC*1	8	2
Event counter control/status register	ECCSR	8	H'FF95	AEC*1	8	2
Event counter H	ECH	8	H'FF96	AEC*1	8	2
Event counter L	ECL	8	H'FF97	AEC*1	8	2
Serial mode register	SMR	8	H'FFA8	SCI3	8	3
Bit rate register	BRR	8	H'FFA9	SCI3	8	3
Serial control register 3	SCR3	8	H'FFAA	SCI3	8	3
Transmit data register	TDR	8	H'FFAB	SCI3	8	3
Serial status register	SSR	8	H'FFAC	SCI3	8	3
Receive data register	RDR	8	H'FFAD	SCI3	8	3

Register Name	Abbre-viation	Bit No	Address	Module Name	Data Bus Width	Access State
Timer mode register A	TMA	8	H'FFB0	Timer A	8	2
Timer counter A	TCA	8	H'FFB1	Timer A	8	2
Timer control/status register W	TCSRW	8	H'FFB2	WDT*2	8	2
Timer counter W	TCW	8	H'FFB3	WDT*2	8	2
Timer control register F	TCRF	8	H'FFB6	Timer F	8	2
Timer control status register F	TCSRF	8	H'FFB7	Timer F	8	2
8-bit timer counter FH	TCFH	8	H'FFB8	Timer F	8	2
8-bit timer counter FL	TCFL	8	H'FFB9	Timer F	8	2
Output compare register FH	OCRFH	8	H'FFBA	Timer F	8	2
Output compare register FL	OCRFL	8	H'FFBB	Timer F	8	2
A/D result register H	ADRRH	8	H'FFC4	A/D converter	8	2
A/D result register L	ADRRL	8	H'FFC5	A/D converter	8	2
A/D mode register	AMR	8	H'FFC6	A/D converter	8	2
A/D start register	ADSR	8	H'FFC7	A/D converter	8	2
Port mode register 2	PMR2	8	H'FFC9	I/O port	8	2
Port mode register 3	PMR3	8	H'FFCA	I/O port	8	2
Port mode register 5	PMR5	8	H'FFCC	I/O port	8	2
PWM2 control register	PWCR2	8	H'FFCD	10-bit PWM	8	2
PWM2 data register U	PWDRU2	8	H'FFCE	10-bit PWM	8	2
PWM2 data register L	PWDRL2	8	H'FFCF	10-bit PWM	8	2
PWM1 control register	PWCR1	8	H'FFD0	10-bit PWM	8	2
PWM1 data register U	PWDRU1	8	H'FFD1	10-bit PWM	8	2
PWM1 data register L	PWDRL1	8	H'FFD2	10-bit PWM	8	2
Port data register 3	PDR3	8	H'FFD6	I/O port	8	2
Port data register 4	PDR4	8	H'FFD7	I/O port	8	2
Port data register 5	PDR5	8	H'FFD8	I/O port	8	2
Port data register 6	PDR6	8	H'FFD9	I/O port	8	2
Port data register 7	PDR7	8	H'FFDA	I/O port	8	2
Port data register 8	PDR8	8	H'FFDB	I/O port	8	2
Port data register 9	PDR9	8	H'FFDC	I/O port	8	2
Port data register A	PDRA	8	H'FFDD	I/O port	8	2

Register Name	Abbre-viation	Bit No	Address	Module Name	Data Bus Width	Access State
Port data register B	PDRB	8	H'FFDE	I/O port	8	2
Port pull-up control register 3	PUCR3	8	H'FFE1	I/O port	8	2
Port pull-up control register 5	PUCR5	8	H'FFE2	I/O port	8	2
Port pull-up control register 6	PUCR6	8	H'FFE3	I/O port	8	2
Port control register 3	PCR3	8	H'FFE6	I/O port	8	2
Port control register 4	PCR4	8	H'FFE7	I/O port	8	2
Port control register 5	PCR5	8	H'FFE8	I/O port	8	2
Port control register 6	PCR6	8	H'FFE9	I/O port	8	2
Port control register 7	PCR7	8	H'FFEA	I/O port	8	2
Port control register 8	PCR8	8	H'FFEB	I/O port	8	2
Port mode register 9	PMR9	8	H'FFEC	I/O port	8	2
Port control register A	PCRA	8	H'FFED	I/O port	8	2
Port mode register B	PMRB	8	H'FFEE	I/O port	8	2
System control register 1	SYSCR1	8	H'FFF0	SYSTEM	8	2
System control register 2	SYSCR2	8	H'FFF1	SYSTEM	8	2
IRQ edge select register	IEGR	8	H'FFF2	Interrupts	8	2
Interrupt enable register 1	IENR1	8	H'FFF3	Interrupts	8	2
Interrupt enable register 2	IENR2	8	H'FFF4	Interrupts	8	2
Interrupt request register 1	IRR1	8	H'FFF6	Interrupts	8	2
Interrupt request register 2	IRR2	8	H'FFF7	Interrupts	8	2
Wakeup interrupt request register	IWPR	8	H’FFF9	Interrupts	8	2
Clock stop register 1	CKSTPR1	8	H'FFFA	SYSTEM	8	2
Clock stop register 2	CKSTPR2	8	H'FFFB	SYSTEM	8	2

Notes: 1. AEC: Asynchronous event counter
2. WDT: Watchdog timer

13.2 Register Bits

Register bit names of the on-chip peripheral modules are described below.

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module Name
FLMCR1	—	SWE	ESU	PSU	EV	PV	E	P	ROM
FLMCR2	FLER	—	—	—	—	—	—	—	
FLPWCR	PDWND	—	—	—	—	—	—	—	
EBR	—	—	—	EB4	EB3	EB2	EB1	EB0	
FENR	FLSHE	—	—	—	—	—	—	—	
ECPWCRH	ECPWCRH7	ECPWCRH6	ECPWCRH5	ECPWCRH4	ECPWCRH3	ECPWCRH2	ECPWCRH1	ECPWCRH0	AEC*1
ECPWCRL	ECPWCRL7	ECPWCRL6	ECPWCRL5	ECPWCRL4	ECPWCRL3	ECPWCRL2	ECPWCRL1	ECPWCRL0	
ECPWDRH	ECPWDRH7	ECPWDRH6	ECPWDRH5	ECPWDRH4	ECPWDRH3	ECPWDRH2	ECPWDRH1	ECPWDRH0	
ECPWDRL	ECPWDRL7	ECPWDRL6	ECPWDRL5	ECPWDRL4	ECPWDRL3	ECPWDRL2	ECPWDRL1	ECPWDRL0	
WEGR	WKEGS7	WKEGS6	WKEGS5	WKEGS4	WKEGS3	WKEGS2	WKEGS1	WKEGS0	Interrupts
SPCR	—	—	SPC32	—	SCINV3	SCINV2	—	—	SCI3
AEGSR	AHEGS1	AHEGS0	ALEGS1	ALEGS0	AIEGS1	AIEGS0	ECPWME	—	AEC*1
ECCR	ACKH1	ACKH0	ACKL1	ACKL0	PWCK2	PWCK1	PWCK0	—	
ECCSR	OVH	OVL	—	CH2	CUEH	CUEL	CRCH	CRCL	
ECH	ECH7	ECH6	ECH5	ECH4	ECH3	ECH2	ECH1	ECH0	
ECL	ECL7	ECL6	ECL5	ECL4	ECL3	ECL2	ECL1	ECL0	
SMR	COM	CHR	PE	PM	STOP	MP	CKS1	CKS0	SCI3
BRR	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0	
SCR3	TIE	RIE	TE	RE	—	TEIE	CKE1	CKE0	
TDR	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0	
SSR	TDRE	RDRF	OER	FER	PER	TEND	MPBR	MPBT	
RDR	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0	
TMA	—	—	—	—	TMA3	TMA2	TMA1	TMA0	Timer A
TCA	TCA7	TCA6	TCA5	TCA4	TCA3	TCA2	TCA1	TCA0	
TCSRW	B6WI	TCWE	B4WI	TCSRWE	B2WI	WDON	BOWI	WRST	WDT*2
TCW	TCW7	TCW6	TCW5	TCW4	TCW3	TCW2	TCW1	TCW0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module Name
TCRF	TOLH	CKSH2	CKSH1	CKSH0	TOLL	CKSL2	CKSL1	CKSL0	Timer F
TCSRF	OVFH	CMFH	OVIEH	CCLRH	OVFL	CMFL	OVIEL	CCLRL	
TCFH	TCFH7	TCFH6	TCFH5	TCFH4	TCFH3	TCFH2	TCFH1	TCFH0	
TCFL	TCFL7	TCFL6	TCFL5	TCFL4	TCFL3	TCFL2	TCFL1	TCFL0	
OCRFH	OCRFH7	OCRFH6	OCRFH5	OCRFH4	OCRFH3	OCRFH2	OCRFH1	OCRFH0	
OCRFL	OCRFL7	OCRFL6	OCRFL5	OCRFL4	OCRFL3	OCRFL2	OCRFL1	OCRFL0	
ADRRH	ADR9	ADR8	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	A/D converter
ADRRL	ADR1	ADR0	—	—	—	—	—	—	
AMR	CKS	—	—	—	CH3	CH2	CH1	CH0	
ADSR	ADSF	—	—	—	—	—	—	—	
PMR2	—	—	POF1	—	—	WDCKS	—	IRQ0	I/O port
PMR3	AEVL	AEVH	—	—	—	TMOFH	TMOFL	—	
PMR5	WKP7	WKP6	WKP5	WKP4	WKP3	WKP2	WKP1	WKP0	
PWCR2	—	—	—	—	—	—	PWCR21	PWCR20	10-bit PWM
PWDRU2	—	—	—	—	—	—	PWDRU21	PWDRU20	
PWDRL2	PWDRL27	PWDRL26	PWDRL25	PWDRL24	PWDRL23	PWDRL22	PWDRL21	PWDRL20	
PWCR1	—	—	—	—	—	—	PWCR11	PWCR10	
PWDRU1	—	—	—	—	—	—	PWDRU11	PWDRU10	
PWDRL1	PWDRL17	PWDRL16	PWDRL15	PWDRL14	PWDRL13	PWDRL12	PWDRL11	PWDRL10	
PDR3	P37	P36	P35	P34	P33	P32	P31	—	I/O port
PDR4	—	—	—	—	P43	P42	P41	P40	
PDR5	P57	P56	P55	P54	P53	P52	P51	P50	
PDR6	P67	P66	P65	P64	P63	P62	P61	P60	
PDR7	P77	P76	P75	P74	P73	P72	P71	P70	
PDR8	—	—	—	—	—	—	—	P80	
PDR9	—	—	P95	P94	P93	P92	P91	P90	
PDRA	—	—	—	—	PA3	PA2	PA1	PA0	
PDRB	—	—	—	—	PB3	PB2	PB1	PB0	
PUCR3	PUCR37	PUCR36	PUCR35	PUCR34	PUCR33	PUCR32	PUCR31	—	
PUCR5	PUCR57	PUCR56	PUCR55	PUCR54	PUCR53	PUCR52	PUCR51	PUCR50	
PUCR6	PUCR67	PUCR66	PUCR65	PUCR64	PUCR63	PUCR62	PUCR61	PUCR60	
PCR3	PCR37	PCR36	PCR35	PCR34	PCR33	PCR32	PCR31	—	
PCR4	—	—	—	—	—	PCR42	PCR41	PCR40	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module Name
PCR5	PCR57	PCR56	PCR55	PCR54	PCR53	PCR52	PCR51	PCR50	I/O port
PCR6	PCR67	PCR66	PCR65	PCR64	PCR63	PCR62	PCR61	PCR60	
PCR7	PCR77	PCR76	PCR75	PCR74	PCR73	PCR72	PCR71	PCR70	
PCR8	—	—	—	—	—	—	—	PCR80	
PMR9	—	—	—	—	PIOFF	—	PWM2	PWM1	
PCRA	—	—	—	—	PCRA3	PCRA2	PCRA1	PCRA0	
PMRB	—	—	—	—	IRQ1	—	—	—	
SYSCR1	SSBY	STS2	STS1	STS0	LSON	—	MA1	MA0	SYSTEM
SYSCR2	—	—	—	NESEL	DTON	MSON	SA1	SA0	
IEGR	—	—	—	—	—	—	IEG1	IEG0	Interrupts
IENR1	IENTA	—	IENWP	—	—	IENEC2	IEN1	IEN0	
IENR2	IENDT	IENAD	—	—	IENTFH	IENTFL	—	IENEC	
IRR1	IRRTA	—	—	—	—	IRREC2	IRRI1	IRRI0	Interrupts
IRR2	IRRDT	IRRAD	—	—	IRRTFH	IRRTFL	—	IRREC	
IWPR	IWPF7	IWPF6	IWPF5	IWPF4	IWPF3	IWPF2	IWPF1	IWPF0	Interrupts
CKSTPR1	—	—	S32CKSTP	ADCKSTP	—	TFCKSTP	—	TACKSTP	SYSTEM
CKSTPR2	—	—	—	PW2CKSTP	AECKSTP	WDCKSTP	PW1CKSTP	—	

Notes: 1. AEC: Asynchronous event counter
2. WDT: Watchdog timer

13.3 Register States in Each Operating Mode

Register Abbreviation	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby	Module
FLMCR1	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	ROM
FLMCR2	Initialized	—	—	—	—	—	—	
FLPWCR	Initialized	—	—	—	—	—	—	
EBR	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
FENR	Initialized	—	—	—	—	—	—	
ECPWCRH	Initialized	—	—	—	—	—	—	AEC*1
ECPWCRL	Initialized	—	—	—	—	—	—	
ECPWDRH	Initialized	—	—	—	—	—	—	
ECPWDRL	Initialized	—	—	—	—	—	—	
WEGR	Initialized	—	—	—	—	—	—	Interrupts
SPCR	Initialized	—	—	—	—	—	—	SCI3
AEGSR	Initialized	—	—	—	—	—	—	AEC*1
ECCR	Initialized	—	—	—	—	—	—	
ECCSR	Initialized	—	—	—	—	—	—	
ECH	Initialized	—	—	—	—	—	—	
ECL	Initialized	—	—	—	—	—	—	
SMR	Initialized	—	—	Initialized	—	—	Initialized	SCI3
BRR	Initialized	—	—	Initialized	—	—	Initialized	
SCR3	Initialized	—	—	Initialized	—	—	Initialized	
TDR	Initialized	—	—	Initialized	—	—	Initialized	
SSR	Initialized	—	—	Initialized	—	—	Initialized	
RDR	Initialized	—	—	Initialized	—	—	Initialized	
TMA	Initialized	—	—	—	—	—	—	Timer A
TCA	Initialized	—	—	—	—	—	—	
TCSRW	Initialized	—	—	—	—	—	—	WDT*2
TCW	Initialized	—	—	—	—	—	—	

Register Abbreviation	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby	Module
TCRF	Initialized	—	—	—	—	—	—	Timer F
TCSRF	Initialized	—	—	—	—	—	—	
TCFH	Initialized	—	—	—	—	—	—	
TCFL	Initialized	—	—	—	—	—	—	
OCRFH	Initialized	—	—	—	—	—	—	
OCRFL	Initialized	—	—	—	—	—	—	
ADRRH	—	—	—	—	—	—	—	A/D converter
ADRRL	—	—	—	—	—	—	—	
AMR	Initialized	—	—	—	—	—	—	
ADSR	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
PMR2	Initialized	—	—	—	—	—	—	I/O port
PMR3	Initialized	—	—	—	—	—	—	
PMR5	Initialized	—	—	—	—	—	—	
PWCR2	Initialized	—	—	—	—	—	—	10-bit PWM
PWDRU2	Initialized	—	—	—	—	—	—	
PWDRL2	Initialized	—	—	—	—	—	—	
PWCR1	Initialized	—	—	—	—	—	—	
PWDRU1	Initialized	—	—	—	—	—	—	
PWDRL1	Initialized	—	—	—	—	—	—	
PDR3	Initialized	—	—	—	—	—	—	I/O port
PDR4	Initialized	—	—	—	—	—	—	
PDR5	Initialized	—	—	—	—	—	—	
PDR6	Initialized	—	—	—	—	—	—	
PDR7	Initialized	—	—	—	—	—	—	
PDR8	Initialized	—	—	—	—	—	—	
PDR9	Initialized	—	—	—	—	—	—	
PDRA	Initialized	—	—	—	—	—	—	
PDRB	Initialized	—	—	—	—	—	—	
PUCR3	Initialized	—	—	—	—	—	—	
PUCR5	Initialized	—	—	—	—	—	—	
PUCR6	Initialized	—	—	—	—	—	—	
PCR3	Initialized	—	—	—	—	—	—	
PCR4	Initialized	—	—	—	—	—	—	

Register Abbreviation	Reset	Active	Sleep	Watch	Subactive	Subsleep	Standby	Module
PCR5	Initialized	—	—	—	—	—	—	I/O port
PCR6	Initialized	—	—	—	—	—	—	
PCR7	Initialized	—	—	—	—	—	—	
PCR8	Initialized	—	—	—	—	—	—	
PMR9	Initialized	—	—	—	—	—	—	
PCRA	Initialized	—	—	—	—	—	—	
PMRB	Initialized	—	—	—	—	—	—	
SYSCR1	Initialized	—	—	—	—	—	—	SYSTEM
SYSCR2	Initialized	—	—	—	—	—	—	
IEGR	Initialized	—	—	—	—	—	—	Interrupts
IENR1	Initialized	—	—	—	—	—	—	
IENR2	Initialized	—	—	—	—	—	—	
IRR1	Initialized	—	—	—	—	—	—	
IRR2	Initialized	—	—	—	—	—	—	
IWPR	Initialized	—	—	—	—	—	—	
CKSTPR1	Initialized	—	—	—	—	—	—	SYSTEM
CKSTPR2	Initialized	—	—	—	—	—	—	

Notes: — is not initialized

1. AEC: Asynchronous event counter
2. WDT: Watchdog timer

Section 14 Electrical Characteristics

14.1 Absolute Maximum Ratings of H8/38704 Group (Flash Memory Version, Mask ROM Version), H8/38702S Group (Mask ROM Version)

Table 14.1 lists the absolute maximum ratings.

Table 14.1 Absolute Maximum Ratings

Item		Symbol	Value	Unit	Note
Power supply voltage		V_{CC}	–0.3 to +4.3	V	*1
Analog power supply voltage		AV_{CC}	–0.3 to +4.3	V	
Input voltage	Other than port B	V_{in}	–0.3 to V_{CC} +0.3	V	
	Port B	AV_{in}	–0.3 to AV_{CC} +0.3	V	
Port 9 pin voltage		V_{P9}	–0.3 to V_{CC} +0.3	V	
Operating temperature		T_{opr}	Regular specifications: –20 to +75*2	°C	
			Wide-range temperature specifications: –40 to +85*3		
Storage temperature		T_{stg}	–55 to +125	°C	

Notes: 1. Permanent damage may result if maximum ratings are exceeded. Normal operation should be under the conditions specified in Electrical Characteristics. Exceeding these values can result in incorrect operation and reduced reliability.

2. When the operating voltage is V_{CC} = 2.7 to 3.6 V during flash memory reading, the operating temperature ranges from –20°C to +75°C when programming or erasing the flash memory. When the operating voltage is V_{CC} = 2.2 to 3.6 V during flash memory reading, the operating temperature ranges from –20°C to +50°C when programming or erasing the flash memory.

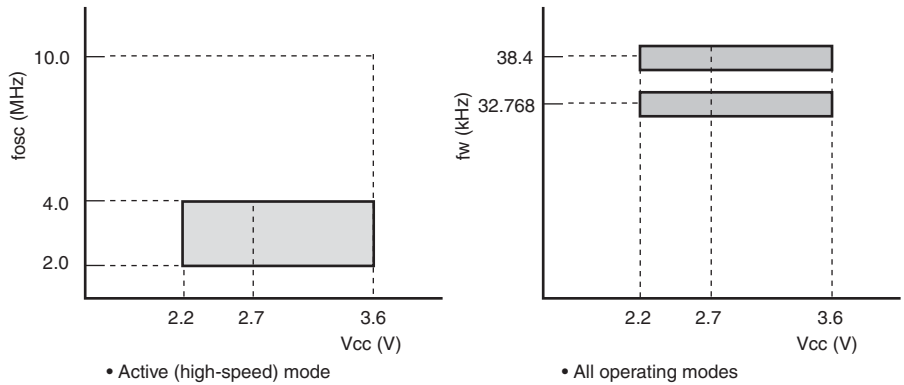
3. The operating temperature ranges from –20°C to +75°C when programming or erasing the flash memory.

14.2 Electrical Characteristics of H8/38704 Group (Flash Memory Version, Mask ROM Version), H8/38702S Group (Mask ROM Version)

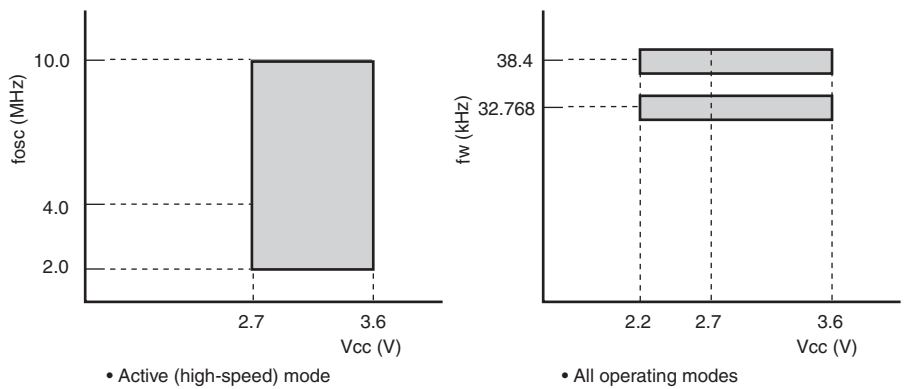
14.2.1 Power Supply Voltage and Operating Ranges

(1) Power Supply Voltage and Oscillation Frequency Range (Flash Memory Version)

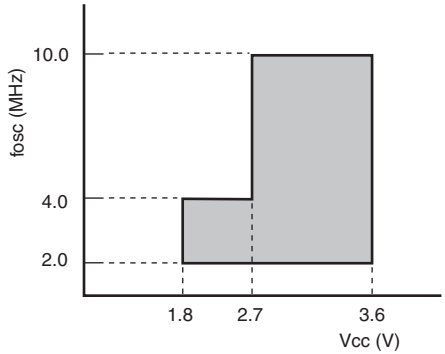
(a) 4-MHz Specification



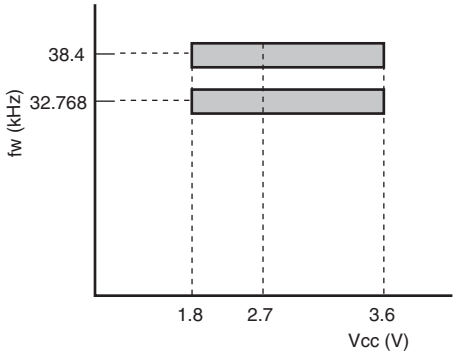
(b) 10-MHz Specification



(2) Power Supply Voltage and Oscillation Frequency Range (Mask ROM Version)

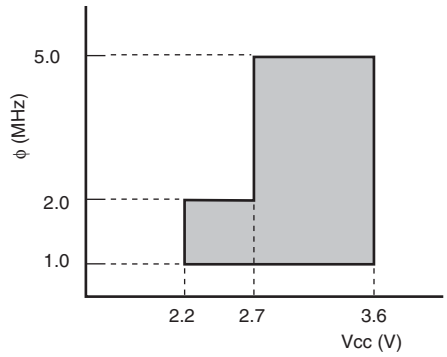


- Active (high-speed) mode
- Sleep (high-speed) mode

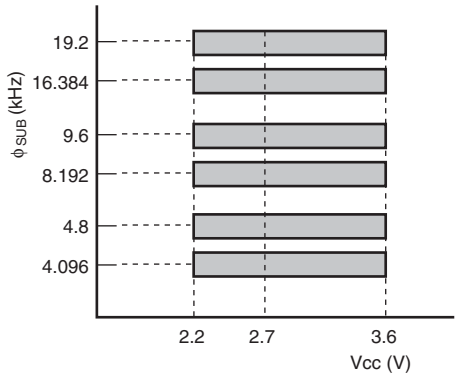


- All operating modes
- When a resonator is used, hold Vcc at 2.2 V to 3.6 V from power-on until the oscillation stabilization time has elapsed.

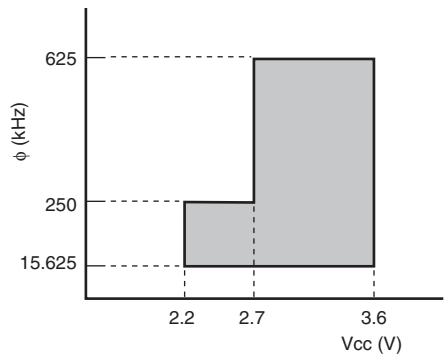
(3) Power Supply Voltage and Operating Frequency Range (Flash Memory Version)



- Active (high-speed) mode
- Sleep (high-speed) mode (except CPU)

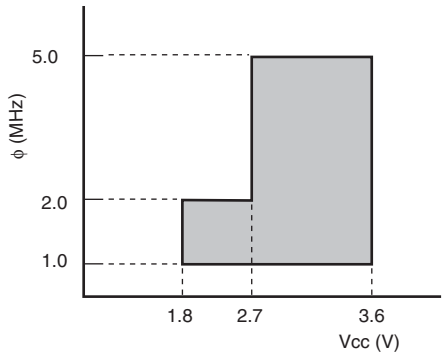


- Subactive mode
- Subsleep mode (except CPU)
- Watch mode (except CPU)

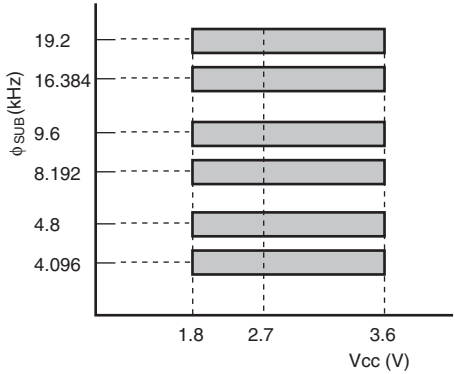


- Active (medium-speed) mode
- Sleep (medium-speed) mode (except A/D converter)

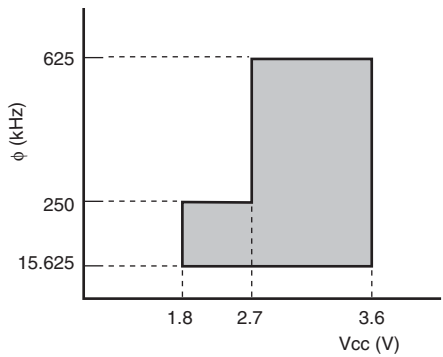
(4) Power Supply Voltage and Operating Frequency Range (Mask ROM Version)



• Active (high-speed) mode
• Sleep (high-speed) mode (except CPU)

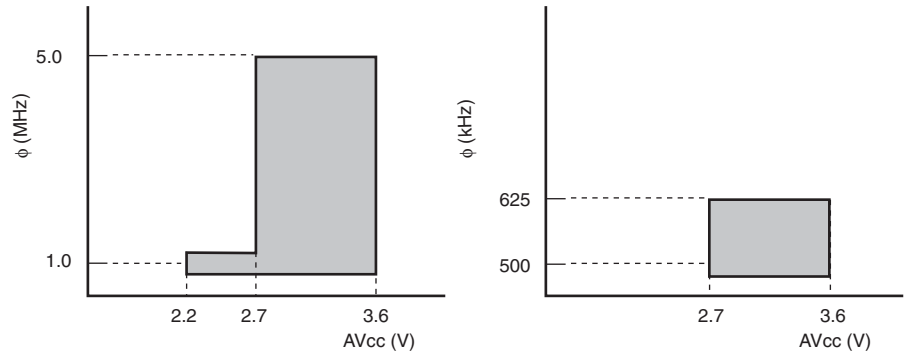


• Subactive mode
• Subsleep mode (except CPU)
• Watch mode (except CPU)



• Active (medium-speed) mode
• Sleep (medium-speed) mode (except A/D converter)

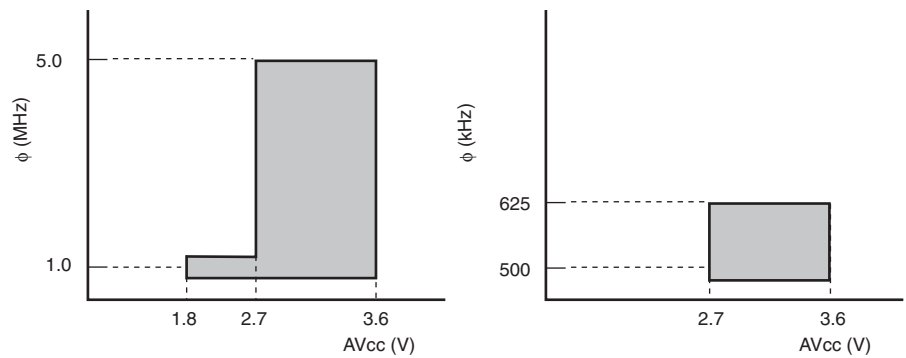
(5) Analog Power Supply Voltage and A/D Converter Operating Range (Flash Memory Version)



- Active (high-speed) mode
- Sleep (high-speed) mode
- Active (medium-speed) mode
- Sleep (medium-speed) mode

Note: When AVcc = 2.2 V to 2.7 V, the operating range is limited to φ = 1.0 MHz.

(6) Analog Power Supply Voltage and A/D Converter Operating Range (Mask ROM Version)



- Active (high-speed) mode
- Sleep (high-speed) mode
- Active (medium-speed) mode
- Sleep (medium-speed) mode

Note: When AVcc = 1.8 V to 2.7 V, the operating range is limited to φ = 1.0 MHz.

14.2.2 DC Characteristics

Table 14.2 lists the DC characteristics.

Table 14.2 DC Characteristics

One of following conditions is applied unless otherwise specified.

Condition A (Flash memory version): V_{CC} = 2.7 V to 3.6 V, AV_{CC} = 2.7 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Condition B (Flash memory version): V_{CC} = 2.2 V to 3.6 V, AV_{CC} = 2.2 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Condition C (Mask ROM version): V_{CC} = 1.8 V to 3.6 V, AV_{CC} = 1.8 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

				Values				
Item	**Symbol**	**Applicable Pins**	**Test Condition**	**Min.**	**Typ.**	**Max.**	**Unit**	**Notes**
Input high voltage	V_{IH}	$\overline{RES}$, $\overline{WKP0}$ to $\overline{WKP7}$, $\overline{IRQ0}$, AEVL, AEVH, SCK32		V_{CC} × 0.9	—	V_{CC} + 0.3	V	
		$\overline{IRQ1}$		V_{CC} × 0.9	—	AV_{CC} + 0.3	V	
		RXD32		V_{CC} × 0.8	—	V_{CC} + 0.3	V	
		OSC1		V_{CC} × 0.9	—	V_{CC} + 0.3	V	
		X1	V_{CC} = 1.8 V to 3.6 V	V_{CC} × 0.9	—	V_{CC} + 0.3	V	
		P31 to P37, P40 to P43, P50 to P57, P60 to P67, P70 to P77, P80, PA0 to PA3		V_{CC} × 0.8	—	V_{CC} + 0.3	V	
		PB0 to PB3		V_{CC} × 0.8	—	AV_{CC} + 0.3	V	
		IRQAEC, P95*5		V_{CC} × 0.9	—	V_{CC} + 0.3	V	

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Notes
Input low voltage	V_{IL}	$\overline{RES}$, $\overline{WKP0}$ to $\overline{WKP7}$, $\overline{IRQ0}$, $\overline{IRQ1}$, IRQAEC, P95*5, AEVL, AEVH, SCK32		– 0.3	—	$V_{CC} \times 0.1$	V	
		RXD32		– 0.3	—	$V_{CC} \times 0.2$	V	
		OSC1		– 0.3	—	$V_{CC} \times 0.1$	V	
		X1		– 0.3	—	$V_{CC} \times 0.1$	V	
		P31 to P37, P40 to P43, P50 to P57, P60 to P67, P70 to P77, P80, PA0 to PA3, PB0 to PB3		– 0.3	—	$V_{CC} \times 0.2$	V	
Output high voltage	V_{OH}	P31 to P37, P40 to P42, P50 to P57, P60 to P67, P70 to P77, P80, PA0 to PA3	V_{CC} = 2.7 V to 3.6 V $-I_{OH}$ = 1.0 mA	V_{CC} – 1.0	—	—	V	
			$-I_{OH}$ = 0.1 mA	V_{CC} – 0.3	—	—		
Output low voltage	V_{OL}	P40 to P42, P50 to P57, P60 to P67, P70 to P77, P80, PA0 to PA3, P31 to P37	I_{OL} = 0.4 mA	—	—	0.5	V	
		P90 to P95	V_{CC} = 2.2 V to 3.6 V I_{OL} = 10.0 mA	—	—	0.5		
			V_{CC} = 1.8 V to 3.6 V I_{OL} = 8.0 mA					

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Notes
Input/ output leakage current	$\lvert I_{IL} \rvert$	$\overline{RES}$, P43, OSC1, X1, P31 to P37, P40 to P42, P50 to P57, P60 to P67, P70 to P77, P80, IRQAEC, PA0 to PA3, P90 to P95	V_{IN} = 0.5 V to V_{CC} – 0.5 V	—	—	1.0	µA	
		PB0 to PB3	V_{IN} = 0.5 V to AV_{CC} – 0.5 V	—	—	1.0		
Pull-up MOS current	$-I_p$	P31 to P37, P50 to P57, P60 to P67	V_{CC} = 3.0 V, V_{IN} = 0.0 V	30	—	180	µA	
Input capacitance	C_{in}	All input pins except power supply pin	f = 1 MHz, V_{IN} = 0.0 V, T_a = 25°C	—	—	15.0	pF	
Vcc start voltage	Vcc_{START}	V_{CC}		0	—	0.1	V	*2
Vcc rising slope	SV_{CC}	V_{CC}		0.05	—	—	V/ms	*2

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Notes
Active mode supply current	I_{OPE1}	V_{CC}	Active (high-speed) mode V_{CC} = 1.8 V, f_{OSC} = 2 MHz	—	0.4	—	mA	*1*3*4 Approx. max. value = 1.1 × Typ.
			Active (high-speed) mode V_{CC} = 3 V, f_{OSC} = 2 MHz	—	0.6	—		*1*3*4 Approx. max. value = 1.1 × Typ.
				—	1.0	—		*2*3*4 Approx. max. value = 1.1 × Typ.
			Active (high-speed) mode V_{CC} = 3 V, f_{OSC} = 4 MHz	—	1.2	—		*1*3*4 Approx. max. value = 1.1 × Typ.
				—	1.6	2.8		*2*3*4 Condition B
			Active (high-speed) mode V_{CC} = 3 V, f_{OSC} = 10 MHz	—	3.1	6.0		*1*3*4
				—	3.6	6.0		*2*3*4 Condition A

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Notes
Active mode supply current	I_{OPE2}	V_{CC}	Active (medium-speed) mode V_{CC} = 1.8 V, f_{OSC} = 2 MHz, ϕ_{OSC}/128	—	0.06	—		*1*3*4 Approx. max. value = 1.1 × Typ.
			Active (medium-speed) mode V_{CC} = 3 V, f_{OSC} = 2 MHz, ϕ_{OSC}/128	—	0.1	—		*1*3*4 Approx. max. value = 1.1 × Typ.
				—	0.5	—		*2*3*4 Approx. max. value = 1.1 × Typ.
			Active (medium-speed) mode V_{CC} = 3 V, f_{OSC} = 4 MHz, ϕ_{OSC}/128	—	0.2	—		*1*3*4 Approx. max. value = 1.1 × Typ.
				—	0.7	1.3		*2*3*4 Condition B
			Active (medium-speed) mode V_{CC} = 3 V, f_{OSC} = 10 MHz, ϕ_{OSC}/128	—	0.6	1.8		*1*3*4
				—	1.0	1.8		*2*3*4 Condition A

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Notes
Sleep mode supply current	I_{SLEEP}	V_{CC}	V_{CC} = 1.8 V, f_{OSC} = 2 MHz	—	0.16	—	mA	*1*3*4 Approx. max. value = 1.1 × Typ.
			V_{CC} = 3 V, f_{OSC} = 2 MHz	—	0.3	—		*1*3*4 Approx. max. value = 1.1 × Typ.
				—	0.6	—		*2*3*4 Approx. max. value = 1.1 × Typ.
			V_{CC} = 3 V, f_{OSC} = 4 MHz	—	0.5	—		*1*3*4 Approx. max. value = 1.1 × Typ.
				—	0.9	2.2		*2*3*4 Condition B
			V_{CC} = 3 V, f_{OSC} = 10 MHz	—	1.3	4.8		*1*3*4
				—	1.7	4.8		*2*3*4 Condition A

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Values Typ.	Values Max.	Unit	Notes
Subactive mode supply current	I_{SUB}	V_{CC}	V_{CC} = 1.8 V, 32-kHz external clock input ($\phi_{SUB} = \phi_W/2$)	—	6.2	—	µA	*1*3*4 Reference value
			V_{CC} = 1.8 V, 32-kHz crystal resonator used ($\phi_{SUB} = \phi_W/2$)	—	5.4	—		
			V_{CC} = 2.7 V, 32-kHz external clock input ($\phi_{SUB} = \phi_W/2$)	—	10	40		*1*3*4
			V_{CC} = 2.7 V, 32-kHz crystal resonator used ($\phi_{SUB} = \phi_W/2$)	—	11	40		
			V_{CC} = 2.7 V, 32-kHz external clock input ($\phi_{SUB} = \phi_W/2$)	—	28	50		*2*3*4
			V_{CC} = 2.7 V, 32-kHz crystal resonator used ($\phi_{SUB} = \phi_W/2$)	—	25	50		
Subsleep mode supply current	I_{SUBSP}	V_{CC}	V_{CC} = 2.7 V, 32-kHz external clock input ($\phi_{SUB} = \phi_W/2$)	—	4.6	16	µA	*3*4
			V_{CC} = 2.7 V, 32-kHz crystal resonator used ($\phi_{SUB} = \phi_W/2$)	—	5.1	16		

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Notes
Watch mode supply current	I_{WATCH}	V_{CC}	V_{CC} = 1.8 V, Ta = 25°C, 32-kHz external clock input	—	1.2	—	µA	*1*3*4 Reference value
			V_{CC} = 1.8 V, Ta = 25°C, 32-kHz crystal resonator used	—	0.6	—		
			V_{CC} = 2.7 V, Ta = 25°C, 32-kHz external clock input	—	2.0	—		*3*4 Reference value
			V_{CC} = 2.7 V, Ta = 25°C, 32-kHz crystal resonator used	—	2.9	—		
			V_{CC} = 2.7 V, 32-kHz external clock input	—	2.0	6.0		*3*4
			V_{CC} = 2.7 V, 32-kHz crystal resonator used	—	2.9	6.0		
Standby mode supply current	I_{STBY}	V_{CC}	V_{CC} = 1.8 V, Ta = 25°C, 32-kHz crystal resonator not used	—	0.1	—	µA	*1*3*4 Reference value
			V_{CC} = 3.0 V, Ta = 25°C, 32-kHz crystal resonator not used	—	0.3	—		*3*4 Reference value
			32-kHz crystal resonator not used	—	1.0	5.0		*3*4
RAM data retaining voltage	V_{RAM}	V_{CC}		1.5	—	—	V	

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Item	Symbol
Allowable output low current (per pin)	I_{OL}	Output pins except port 9		—	—	0.5	mA	
		P90 to P95	V_{CC} = 2.2 V to 3.6 V	—	—	10.0		
			Other than above	—	—	8.0		
Allowable output low current (total)	$\sum I_{OL}$	Output pins except port 9		—	—	20.0	mA	
		Port 9		—	—	60.0		
Allowable output high current (per pin)	$-I_{OH}$	All output pins	V_{CC} = 2.7 V to 3.6 V	—	—	2.0	mA	
			Other than above	—	—	0.2		
Allowable output high current (total)	$\sum -I_{OH}$	All output pins		—	—	10.0	mA	

Notes: Connect the TEST pin to V_{SS}.

1. Applies to the mask-ROM version.
2. Applies to the flash memory version.
3. Pin states when supply current is measured

Mode	$\overline{RES}$ Pin	Internal State	Other Pins	Oscillator Pins
Active (high-speed) mode (I_{OPE1})	V_{CC}	Only CPU operates	V_{CC}	System clock: crystal resonator Subclock: Pin X1 = GND
Active (medium-speed) mode (I_{OPE2})				
Sleep mode	V_{CC}	Only all on-chip timers operate	V_{CC}	
Subactive mode	V_{CC}	Only CPU operates	V_{CC}	System clock: crystal resonator Subclock: crystal resonator
Subsleep mode	V_{CC}	Only all on-chip timers operate CPU stops	V_{CC}	
Watch mode	V_{CC}	Only clock time base operates CPU stops	V_{CC}	
Standby mode	V_{CC}	CPU and timers both stop	V_{CC}	System clock: crystal resonator Subclock: Pin X1 = GND

4. Except current which flows to the pull-up MOS or output buffer
5. Used when user mode or boot mode is determined after canceling a reset in the flash memory version

14.2.3 AC Characteristics

Table 14.3 lists the control signal timing and table 14.4 lists the serial interface timing.

Table 14.3 Control Signal Timing

One of following conditions is applied unless otherwise specified.

Condition A (Flash memory version): V_{CC} = 2.7 V to 3.6 V, AV_{CC} = 2.7 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Condition B (Flash memory version): V_{CC} = 2.2 V to 3.6 V, AV_{CC} = 2.2 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Condition C (Mask ROM version): V_{CC} = 1.8 V to 3.6 V, AV_{CC} = 1.8 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Item	Symbol	Applicable Pins	Test Condition	Values			Unit	Reference Figure
				Min.	Typ.	Max.		
System clock oscillation frequency	f_{OSC}	OSC1, OSC2	V_{CC} = 2.7 V to 3.6 V in conditions A and C	2.0	—	10.0	MHz	
			Other than above in condition C and condition B	2.0	—	4.0		
OSC clock (ϕ_{OSC}) cycle time	t_{OSC}	OSC1, OSC2	V_{CC} = 2.7 V to 3.6 V in conditions A and C	100	—	500	ns	Figure 14.1
			Other than above in condition C and condition B	250	—	500		
System clock (ϕ) cycle time	t_{cyc}			2	—	128	t_{OSC}	
				—	—	64	µs	
Subclock oscillation frequency	f_W	X1, X2		—	32.768 or 38.4	—	kHz	
Watch clock (ϕ_W) cycle time	t_W	X1, X2		—	30.5 or 26.0	—	µs	Figure 14.1
Subclock (ϕ_{SUB}) cycle time	t_{subcyc}			2	—	8	t_W	*
Instruction cycle time				2	—	—	t_{cyc} t_{subcyc}	

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Reference Figure
Oscillation stabilization time	t_{rc}	OSC1, OSC2	V_{CC} = 2.7 V to 3.6 V when using crystal resonator in figure 14.8	—	0.8	2.0	ms	Figure 14.8
			V_{CC} = 2.2 V to 3.6 V when using crystal resonator in figure 14.8 and in conditions B and C	—	1.2	3.0		
			Other than above in condition C and when using crystal resonator in figure 14.8	—	4.0	—		
			V_{CC} = 2.7 V to 3.6 V when using ceramic resonator in figure 14.8 and in conditions A and C	—	20	45	µs	
			V_{CC} = 2.2 V to 3.6 V when using ceramic resonator (1) in figure 14.8 and in conditions B and C	—	20	45		
			Other than above in condition C and when using ceramic resonator (1) in figure 14.8	—	80	—		
			Other than above	—	—	50	ms	
	t_{rc}	X1, X2	V_{CC} = 2.7 V to 3.6 V	—	—	2.0	s	
			V_{CC} = 2.2 V to 3.6 V and in conditions B and C	—	—	2.0		
			Other than above in condition C	—	4.0	—		

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Reference Figure
External clock high width	t_{CPH}	OSC1	V_{CC} = 2.7 V to 3.6 V in conditions A and C	40	—	—	ns	Figure 14.1
			Other than above in condition C and condition B	100	—	—		
		X1		—	15.26 or 13.02	—	µs	
External clock low width	t_{CPL}	OSC1	V_{CC} = 2.7 V to 3.6 V in conditions A and C	40	—	—	ns	Figure 14.1
			Other than above in condition C and condition B	100	—	—		
		X1		—	15.26 or 13.02	—	µs	
External clock rise time	t_{CPr}	OSC1	V_{CC} = 2.7 V to 3.6 V in conditions A and C	—	—	10	ns	Figure 14.1
			Other than above in condition C and condition B	—	—	25		
		X1		—	—	55.0	ns	
External clock fall time	t_{CPf}	OSC1	V_{CC} = 2.7 V to 3.6 V in conditions A and C	—	—	10	ns	Figure 14.1
			Other than above in condition C and condition B	—	—	25		
		X1		—	—	55.0	ns	
$\overline{RES}$ pin low width	t_{REL}	$\overline{RES}$		10	—	—	t_{cyc}	Figure 14.2
Input pin high width	t_{IH}	$\overline{IRQ0}$, $\overline{IRQ1}$, IRQAEC, $\overline{WKP0}$ to $\overline{WKP7}$,		2	—	—	t_{cyc} t_{subcyc}	Figure 14.3
		AEVL, AEVH		0.5	—	—	t_{OSC}	

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Reference Figure
Input pin low width	t_{IL}	$\overline{IRQ0}$, $\overline{IRQ1}$, IRQAEC, $\overline{WKP0}$ to $\overline{WKP7}$,		2	—	—	t_{cyc} t_{subcyc}	Figure 14.3
		AEVL, AEVH		0.5	—	—	t_{OSC}	

Note: * Determined by the SA1 and SA0 bits in the system control register 2 (SYSCR2).

Table 14.4 Serial Interface (SCI3) Timing

One of following conditions is applied unless otherwise specified.

Condition A (Flash memory version): V_{CC} = 2.7 V to 3.6 V, AV_{CC} = 2.7 V to 3.6 V, $V_{SS} = AV_{SS}$ = 0.0 V

Condition B (Flash memory version): V_{CC} = 2.2 V to 3.6 V, AV_{CC} = 2.2 V to 3.6 V, $V_{SS} = AV_{SS}$ = 0.0 V

Condition C (Mask ROM version): V_{CC} = 1.8 V to 3.6 V, AV_{CC} = 1.8 V to 3.6 V, $V_{SS} = AV_{SS}$ = 0.0 V

Item		Symbol	Test Condition	Values Min.	Typ.	Max.	Unit	Reference Figure
Input clock cycle	Asynchronous	t_{scyc}		4	—	—	t_{cyc} or t_{subcyc}	Figure 14.4
	Clocked synchronous			6	—	—		
Input clock pulse width		t_{SCKW}		0.4	—	0.6	t_{scyc}	Figure 14.4
Transmit data delay time (clocked synchronous)		t_{TXD}		—	—	1	t_{cyc} or t_{subcyc}	Figure 14.5
Receive data setup time (clocked synchronous)		t_{RXS}		400.0	—	—	ns	Figure 14.5
Receive data hold time (clocked synchronous)		t_{RXH}		400.0	—	—	ns	Figure 14.5

14.2.4 A/D Converter Characteristics

Table 14.5 shows the A/D converter characteristics.

Table 14.5 A/D Converter Characteristics

One of following conditions is applied unless otherwise specified.

Condition A (Flash memory version): V_{CC} = 2.7 V to 3.6 V, AV_{CC} = 2.7 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Condition B (Flash memory version): V_{CC} = 2.2 V to 3.6 V, AV_{CC} = 2.2 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Condition C (Mask ROM version): V_{CC} = 1.8 V to 3.6 V, AV_{CC} = 1.8 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V

Item	Symbol	Applicable Pins	Test Condition	Values			Unit	Reference Figure
				Min.	Typ.	Max.		
Analog power supply voltage	AV_{CC}	AV_{CC}	Condition A	2.7	—	3.6	V	*1
			Condition B	2.2	—	3.6		
			Condition C	1.8	—	3.6		
Analog input voltage	AV_{IN}	AN0 to AN3		– 0.3	—	AV_{CC} + 0.3	V	
Analog power supply current	AI_{OPE}	AV_{CC}	AV_{CC} = 3.0 V	—	—	1.0	mA	
	AI_{STOP1}	AV_{CC}		—	600	—	µA	*2 Reference value
	AI_{STOP2}	AV_{CC}		—	—	5.0	µA	*3
Analog input capacitance	C_{AIN}	AN0 to AN3		—	—	15.0	pF	
Allowable signal source impedance	R_{AIN}			—	—	10.0	kΩ	
Resolution (data length)				—	—	10	bit	

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Typ.	Max.	Unit	Reference Figure
Nonlinearity error			AV_{CC} = 2.7 V to 3.6 V	—	—	±3.5	LSB	
			AV_{CC} = 2.2 V to 3.6 V in condition B, AV_{CC} = 2.0 V to 3.6 V in condition C	—	—	±5.5		
			Other than above in condition C	—	—	±7.5		*4
Quantization error				—	—	±0.5	LSB	
Absolute accuracy			AV_{CC} = 2.7 V to 3.6 V	—	±2.0	±4.0	LSB	
			AV_{CC} = 2.2 V to 3.6 V in condition B, AV_{CC} = 2.0 V to 3.6 V in condition C	—	±2.5	±6.0		
			Other than above in condition C	—	±2.5	±8.0		*4
Conversion time			AV_{CC} = 2.7 V to 3.6 V	12.4	—	124	µs	
			Other than above	62	—	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.
4. The conversion time is 62 µs.

14.2.5 Flash Memory Characteristics

Table 14.6 shows the flash memory characteristics.

Table 14.6 Flash Memory Characteristics

Condition A: AV_{CC} = 2.7 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V, V_{CC} = 2.7 V to 3.6 V (range of operating voltage when reading), V_{CC} = 3.0 V to 3.6 V (range of operating voltage when programming/erasing), T_a = –20°C to +75°C (range of operating temperature when programming/erasing: product with regular specifications, product with wide-range temperature specifications)

Condition B: AV_{CC} = 2.2 V to 3.6 V, V_{SS} = AV_{SS} = 0.0 V, V_{CC} = 2.2 V to 3.6 V (range of operating voltage when reading), V_{CC} = 3.0 V to 3.6 V (range of operating voltage when programming/erasing), T_a = –20°C to +50°C (range of operating temperature when programming/erasing: product with regular specifications)

Item		Symbol	Test Conditions	Values Min.	Typ.	Max.	Unit
Programming time*1*2*4		t_P		—	7	200	ms/ 128 bytes
Erase time*1*3*5		t_E		—	100	1200	ms/ block
Reprogramming count		N_{WEC}		1000*8	10000*9	—	times
Data retain period		t_{DRP}		10*10	—	—	year
Programming	Wait time after SWE-bit setting*1	x		1	—	—	µs
	Wait time after PSU-bit setting*1	y		50	—	—	µs
	Wait time after P-bit setting*1*4	z1	1 ≤ n ≤ 6	28	30	32	µs
		z2	7 ≤ n ≤ 1000	198	200	202	µs
		z3	Additional programming	8	10	12	µs
	Wait time after P-bit clear*1	α		5	—	—	µs
	Wait time after PSU-bit clear*1	β		5	—	—	µs
	Wait time after PV-bit setting*1	γ		4	—	—	µs

Item		Symbol	Test Conditions	Values Min.	Typ.	Max.	Unit
Programming	Wait time after dummy write*1	ε		2	—	—	µs
	Wait time after PV-bit clear*1	η		2	—	—	µs
	Wait time after SWE-bit clear*1	θ		100	—	—	µs
	Maximum programming count*1*4*5	N		—	—	1000	times
Erase	Wait time after SWE-bit setting*1	x		1	—	—	µs
	Wait time after ESU-bit setting*1	y		100	—	—	µs
	Wait time after E-bit setting*1*6	z		10	—	100	ms
	Wait time after E-bit clear*1	α		10	—	—	µs
	Wait time after ESU-bit clear*1	β		10	—	—	µs
	Wait time after EV-bit setting*1	γ		20	—	—	µs
	Wait time after dummy write*1	ε		2	—	—	µs
	Wait time after EV-bit clear*1	η		4	—	—	µs
	Wait time after SWE-bit clear*1	θ		100	—	—	µs
	Maximum erase count*1*6*7	N		—	—	120	times

Notes:
1. Set the times according to the program/erase algorithms.
2. Programming time per 128 bytes (Shows the total period for which the P bit in FLMCR1 is set. It does not include the programming verification time.)
3. Block erase time (Shows the total period for which the E bit in FLMCR1 is set. It does not include the erase verification time.)
4. Maximum programming time (t_P (max))
   t_P (max) = Wait time after P-bit setting (z) • maximum number of writes (N)
5. The maximum number of writes (N) should be set according to the actual set value of z1, z2, and z3 to allow programming within the maximum programming time (t_P (max)).
   The wait time after P-bit setting (z1 and z2) should be alternated according to the number of writes (n) as follows:
   $1 \leq n \leq 6$ z1 = 30 µs
   $7 \leq n \leq 1000$ z2 = 200 µs

6. Maximum erase time (t_E (max))
   t_E (max) = Wait time after E-bit setting (z) • maximum erase count (N)
7. The maximum number of erases (N) should be set according to the actual set value of z to allow erasing within the maximum erase time (t_E (max)).
8. This minimum value guarantees all characteristics after reprogramming (the guaranteed range is from 1 to the minimum value).
9. Reference value when the temperature is 25°C (normally reprogramming will be performed by this count).
10. This is a data retain characteristic when reprogramming is performed within the specification range including this minimum value.

14.3 Operation Timing

Figures 14.1 to 14.5 show the operation timings.

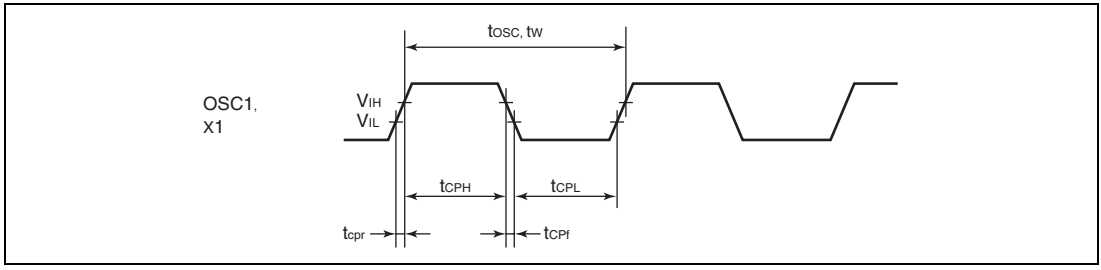


Figure 14.1 Clock Input Timing

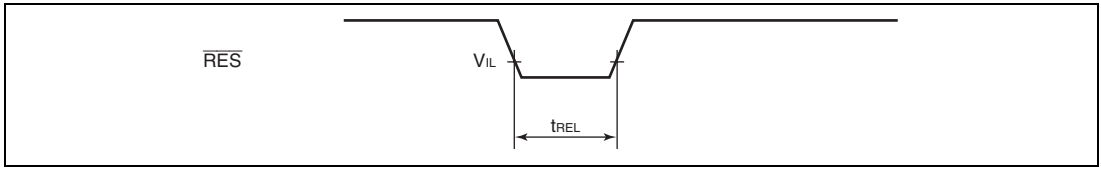


Figure 14.2 $\overline{\text{RES}}$ Low Width Timing

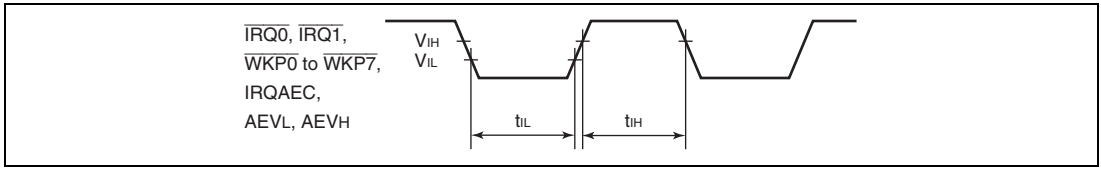


Figure 14.3 Input Timing

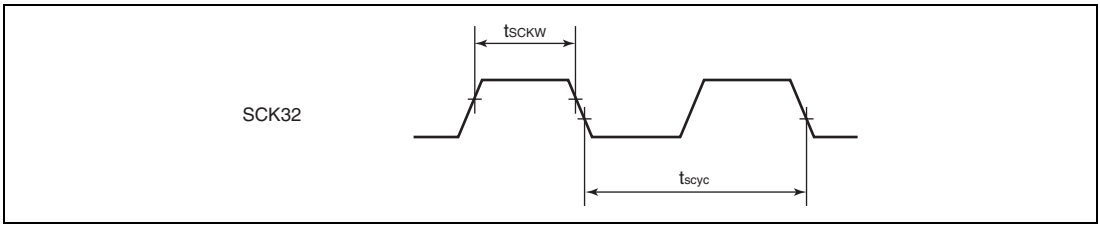
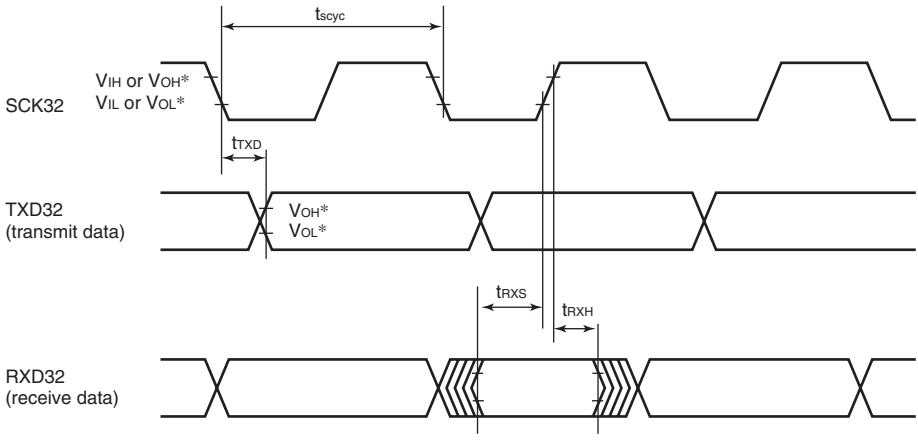


Figure 14.4 SCK3 Input Clock Timing



Note: * Output timing reference levels

Output high	V_{OH} = 1/2 V_{CC} + 0.2 V
Output low	V_{OL} = 0.8 V

Load conditions are shown in figure 14.6.

Figure 14.5 SCI3 Input/Output Timing in Clocked Synchronous Mode

14.4 Output Load Condition

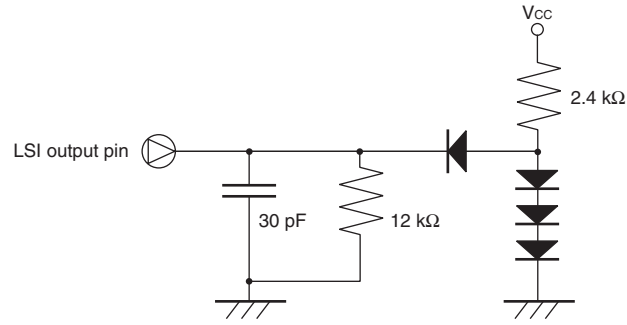
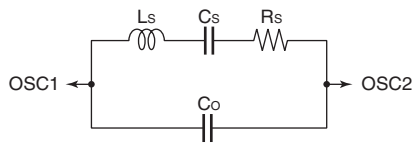


Figure 14.6 Output Load Circuit

14.5 Resonator Equivalent Circuit



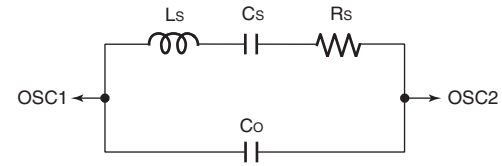
Crystal Resonator Parameter

Frequency (MHz)	4	4.193	10
Rs (max.)	100 Ω	100 Ω	30 Ω
Co (max.)	16 pF	16 pF	16 pF

Ceramic Resonator Parameter

Frequency (MHz)	2	4	10
Rs (max.)	18.3 Ω	6.8 Ω	4.6 Ω
Co (max.)	36.94 pF	36.72 pF	32.31 pF

Figure 14.7 Resonator Equivalent Circuit



Crystal Resonator Parameter
(Nominal Values by Manufacturer)

Frequency	4	Manufacturer
Rs (max)	150Ω	KYOCERA KINSEKI CORPORATION
Co (max)	12pF	

Ceramic Resonator Parameter (1)
(Nominal Values by Manufacturer)

Frequency	2	Manufacturer
Rs (max)	18.3Ω	Murata Manufacturing Co., Ltd.
Co (max)	36.94pF	

Ceramic Resonator Parameter (2)
(Nominal Values by Manufacturer)

Frequency	10	Manufacturer
Rs (max)	4.6Ω	Murata Manufacturing Co., Ltd.
Co (max)	32.31pF	

Figure 14.8 Resonator Equivalent Circuit

14.6 Usage Note

The flash memory and mask ROM versions satisfy the electrical characteristics shown in this manual, but actual electrical characteristic values, operating margins, noise margins, and other properties may vary due to differences in manufacturing process, on-chip ROM, layout patterns, and so on.

When system evaluation testing is carried out using the flash memory version, the same evaluation testing should also be conducted for the mask ROM version when changing over to that version.

Appendix

A. Instruction Set

A.1 Instruction List

Condition Code

Symbol	Description
Rd	General destination register
Rs	General source register
Rn	General register
ERd	General destination register (address register or 32-bit register)
ERs	General source register (address register or 32-bit register)
ERn	General register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
PC	Program counter
SP	Stack pointer
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
disp	Displacement
→	Transfer from the operand on the left to the operand on the right, or transition from the state on the left to the state on the right
+	Addition of the operands on both sides
–	Subtraction of the operand on the right from the operand on the left
×	Multiplication of the operands on both sides
÷	Division of the operand on the left by the operand on the right
∧	Logical AND of the operands on both sides
∨	Logical OR of the operands on both sides
⊕	Logical exclusive OR of the operands on both sides

Symbol	Description
¬	NOT (logical complement)
( ), < >	Contents of operand

Note: General registers include 8-bit registers (R0H to R7H and R0L to R7L) and 16-bit registers (R0 to R7 and E0 to E7).

Condition Code Notation (cont)

Symbol	Description
↕	Changed according to execution result
*	Undetermined (no guaranteed value)
0	Cleared to 0
1	Set to 1
—	Not affected by execution of the instruction
Δ	Varies depending on conditions, described in notes

Table A.1 Instruction Set

1. Data Transfer Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
MOV	MOV.B #xx:8, Rd	B	2									#xx:8 → Rd8	—	—	↕	↕	0	—	2	
	MOV.B Rs, Rd	B		2								Rs8 → Rd8	—	—	↕	↕	0	—	2	
	MOV.B @ERs, Rd	B			2							@ERs → Rd8	—	—	↕	↕	0	—	4	
	MOV.B @(d:16, ERs), Rd	B				4						@(d:16, ERs) → Rd8	—	—	↕	↕	0	—	6	
	MOV.B @(d:24, ERs), Rd	B				8						@(d:24, ERs) → Rd8	—	—	↕	↕	0	—	10	
	MOV.B @ERs+, Rd	B					2					@ERs → Rd8 ERs32+1 → ERs32	—	—	↕	↕	0	—	6	
	MOV.B @aa:8, Rd	B						2				@aa:8 → Rd8	—	—	↕	↕	0	—	4	
	MOV.B @aa:16, Rd	B						4				@aa:16 → Rd8	—	—	↕	↕	0	—	6	
	MOV.B @aa:24, Rd	B						6				@aa:24 → Rd8	—	—	↕	↕	0	—	8	
	MOV.B Rs, @ERd	B			2							Rs8 → @ERd	—	—	↕	↕	0	—	4	
	MOV.B Rs, @(d:16, ERd)	B				4						Rs8 → @(d:16, ERd)	—	—	↕	↕	0	—	6	
	MOV.B Rs, @(d:24, ERd)	B				8						Rs8 → @(d:24, ERd)	—	—	↕	↕	0	—	10	
	MOV.B Rs, @–ERd	B					2					ERd32–1 → ERd32 Rs8 → @ERd	—	—	↕	↕	0	—	6	
	MOV.B Rs, @aa:8	B						2				Rs8 → @aa:8	—	—	↕	↕	0	—	4	
	MOV.B Rs, @aa:16	B						4				Rs8 → @aa:16	—	—	↕	↕	0	—	6	
	MOV.B Rs, @aa:24	B						6				Rs8 → @aa:24	—	—	↕	↕	0	—	8	
	MOV.W #xx:16, Rd	W	4									#xx:16 → Rd16	—	—	↕	↕	0	—	4	
	MOV.W Rs, Rd	W		2								Rs16 → Rd16	—	—	↕	↕	0	—	2	
	MOV.W @ERs, Rd	W			2							@ERs → Rd16	—	—	↕	↕	0	—	4	
	MOV.W @(d:16, ERs), Rd	W				4						@(d:16, ERs) → Rd16	—	—	↕	↕	0	—	6	
	MOV.W @(d:24, ERs), Rd	W				8						@(d:24, ERs) → Rd16	—	—	↕	↕	0	—	10	
	MOV.W @ERs+, Rd	W					2					@ERs → Rd16 ERs32+2 → @ERd32	—	—	↕	↕	0	—	6	
	MOV.W @aa:16, Rd	W						4				@aa:16 → Rd16	—	—	↕	↕	0	—	6	
	MOV.W @aa:24, Rd	W						6				@aa:24 → Rd16	—	—	↕	↕	0	—	8	
	MOV.W Rs, @ERd	W			2							Rs16 → @ERd	—	—	↕	↕	0	—	4	
	MOV.W Rs, @(d:16, ERd)	W				4						Rs16 → @(d:16, ERd)	—	—	↕	↕	0	—	6	
	MOV.W Rs, @(d:24, ERd)	W				8						Rs16 → @(d:24, ERd)	—	—	↕	↕	0	—	10	

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
MOV	MOV.W Rs, @–ERd	W					2					ERd32–2 → ERd32 Rs16 → @ERd	—	—	↕	↕	0	—	6	
	MOV.W Rs, @aa:16	W						4				Rs16 → @aa:16	—	—	↕	↕	0	—	6	
	MOV.W Rs, @aa:24	W						6				Rs16 → @aa:24	—	—	↕	↕	0	—	8	
	MOV.L #xx:32, Rd	L	6									#xx:32 → Rd32	—	—	↕	↕	0	—	6	
	MOV.L ERs, ERd	L		2								ERs32 → ERd32	—	—	↕	↕	0	—	2	
	MOV.L @ERs, ERd	L			4							@ERs → ERd32	—	—	↕	↕	0	—	8	
	MOV.L @(d:16, ERs), ERd	L				6						@(d:16, ERs) → ERd32	—	—	↕	↕	0	—	10	
	MOV.L @(d:24, ERs), ERd	L				10						@(d:24, ERs) → ERd32	—	—	↕	↕	0	—	14	
	MOV.L @ERs+, ERd	L					4					@ERs → ERd32 ERs32+4 → ERs32	—	—	↕	↕	0	—	10	
	MOV.L @aa:16, ERd	L						6				@aa:16 → ERd32	—	—	↕	↕	0	—	10	
	MOV.L @aa:24, ERd	L						8				@aa:24 → ERd32	—	—	↕	↕	0	—	12	
	MOV.L ERs, @ERd	L			4							ERs32 → @ERd	—	—	↕	↕	0	—	8	
	MOV.L ERs, @(d:16, ERd)	L				6						ERs32 → @(d:16, ERd)	—	—	↕	↕	0	—	10	
	MOV.L ERs, @(d:24, ERd)	L				10						ERs32 → @(d:24, ERd)	—	—	↕	↕	0	—	14	
	MOV.L ERs, @–ERd	L					4					ERd32–4 → ERd32 ERs32 → @ERd	—	—	↕	↕	0	—	10	
	MOV.L ERs, @aa:16	L						6				ERs32 → @aa:16	—	—	↕	↕	0	—	10	
	MOV.L ERs, @aa:24	L						8				ERs32 → @aa:24	—	—	↕	↕	0	—	12	
POP	POP.W Rn	W									2	@SP → Rn16 SP+2 → SP	—	—	↕	↕	0	—	6	
	POP.L ERn	L									4	@SP → ERn32 SP+4 → SP	—	—	↕	↕	0	—	10	
PUSH	PUSH.W Rn	W									2	SP–2 → SP Rn16 → @SP	—	—	↕	↕	0	—	6	
	PUSH.L ERn	L									4	SP–4 → SP ERn32 → @SP	—	—	↕	↕	0	—	10	
MOVFPE	MOVFPE @aa:16, Rd	B						4				Cannot be used in this LSI	Cannot be used in this LSI							
MOVTPE	MOVTPE Rs, @aa:16	B						4				Cannot be used in this LSI	Cannot be used in this LSI							

2. Arithmetic Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
ADD	ADD.B #xx:8, Rd	B	2									Rd8+#xx:8 → Rd8	—	↕	↕	↕	↕	↕	2	
	ADD.B Rs, Rd	B		2								Rd8+Rs8 → Rd8	—	↕	↕	↕	↕	↕	2	
	ADD.W #xx:16, Rd	W	4									Rd16+#xx:16 → Rd16	—	(1)	↕	↕	↕	↕	4	
	ADD.W Rs, Rd	W		2								Rd16+Rs16 → Rd16	—	(1)	↕	↕	↕	↕	2	
	ADD.L #xx:32, ERd	L	6									ERd32+#xx:32 → ERd32	—	(2)	↕	↕	↕	↕	6	
	ADD.L ERs, ERd	L		2								ERd32+ERs32 → ERd32	—	(2)	↕	↕	↕	↕	2	
ADDX	ADDX.B #xx:8, Rd	B	2									Rd8+#xx:8 +C → Rd8	—	↕	↕	(3)	↕	↕	2	
	ADDX.B Rs, Rd	B		2								Rd8+Rs8 +C → Rd8	—	↕	↕	(3)	↕	↕	2	
ADDS	ADDS.L #1, ERd	L		2								ERd32+1 → ERd32	—	—	—	—	—	—	2	
	ADDS.L #2, ERd	L		2								ERd32+2 → ERd32	—	—	—	—	—	—	2	
	ADDS.L #4, ERd	L		2								ERd32+4 → ERd32	—	—	—	—	—	—	2	
INC	INC.B Rd	B		2								Rd8+1 → Rd8	—	—	↕	↕	↕	—	2	
	INC.W #1, Rd	W		2								Rd16+1 → Rd16	—	—	↕	↕	↕	—	2	
	INC.W #2, Rd	W		2								Rd16+2 → Rd16	—	—	↕	↕	↕	—	2	
	INC.L #1, ERd	L		2								ERd32+1 → ERd32	—	—	↕	↕	↕	—	2	
	INC.L #2, ERd	L		2								ERd32+2 → ERd32	—	—	↕	↕	↕	—	2	
DAA	DAA Rd	B		2								Rd8 decimal adjust → Rd8	—	*	↕	↕	*	—	2	
SUB	SUB.B Rs, Rd	B		2								Rd8–Rs8 → Rd8	—	↕	↕	↕	↕	↕	2	
	SUB.W #xx:16, Rd	W	4									Rd16–#xx:16 → Rd16	—	(1)	↕	↕	↕	↕	4	
	SUB.W Rs, Rd	W		2								Rd16–Rs16 → Rd16	—	(1)	↕	↕	↕	↕	2	
	SUB.L #xx:32, ERd	L	6									ERd32–#xx:32 → ERd32	—	(2)	↕	↕	↕	↕	6	
	SUB.L ERs, ERd	L		2								ERd32–ERs32 → ERd32	—	(2)	↕	↕	↕	↕	2	
SUBX	SUBX.B #xx:8, Rd	B	2									Rd8–#xx:8–C → Rd8	—	↕	↕	(3)	↕	↕	2	
	SUBX.B Rs, Rd	B		2								Rd8–Rs8–C → Rd8	—	↕	↕	(3)	↕	↕	2	
SUBS	SUBS.L #1, ERd	L		2								ERd32–1 → ERd32	—	—	—	—	—	—	2	
	SUBS.L #2, ERd	L		2								ERd32–2 → ERd32	—	—	—	—	—	—	2	
	SUBS.L #4, ERd	L		2								ERd32–4 → ERd32	—	—	—	—	—	—	2	
DEC	DEC.B Rd	B		2								Rd8–1 → Rd8	—	—	↕	↕	↕	—	2	
	DEC.W #1, Rd	W		2								Rd16–1 → Rd16	—	—	↕	↕	↕	—	2	
	DEC.W #2, Rd	W		2								Rd16–2 → Rd16	—	—	↕	↕	↕	—	2	

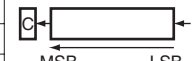
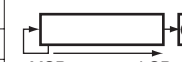
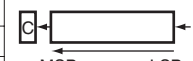
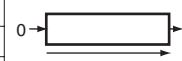
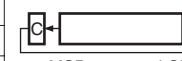
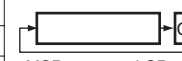
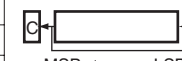
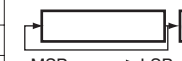
Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
DEC	DEC.L #1, ERd	L		2								ERd32–1 → ERd32	—	—	↕	↕	↕	—	2	
	DEC.L #2, ERd	L		2								ERd32–2 → ERd32	—	—	↕	↕	↕	—	2	
DAS	DAS.Rd	B		2								Rd8 decimal adjust → Rd8	—	*	↕	↕	*	—	2	
MULXU	MULXU. B Rs, Rd	B		2								Rd8 × Rs8 → Rd16 (unsigned multiplication)	—	—	—	—	—	—	14	
	MULXU. W Rs, ERd	W		2								Rd16 × Rs16 → ERd32 (unsigned multiplication)	—	—	—	—	—	—	22	
MULXS	MULXS. B Rs, Rd	B		4								Rd8 × Rs8 → Rd16 (signed multiplication)	—	—	↕	↕	—	—	16	
	MULXS. W Rs, ERd	W		4								Rd16 × Rs16 → ERd32 (signed multiplication)	—	—	↕	↕	—	—	24	
DIVXU	DIVXU. B Rs, Rd	B		2								Rd16 ÷ Rs8 → Rd16 (RdH: remainder, RdL: quotient) (unsigned division)	—	—	(6)	(7)	—	—	14	
	DIVXU. W Rs, ERd	W		2								ERd32 ÷ Rs16 → ERd32 (Ed: remainder, Rd: quotient) (unsigned division)	—	—	(6)	(7)	—	—	22	
DIVXS	DIVXS. B Rs, Rd	B		4								Rd16 ÷ Rs8 → Rd16 (RdH: remainder, RdL: quotient) (signed division)	—	—	(8)	(7)	—	—	16	
	DIVXS. W Rs, ERd	W		4								ERd32 ÷ Rs16 → ERd32 (Ed: remainder, Rd: quotient) (signed division)	—	—	(8)	(7)	—	—	24	
CMP	CMP.B #xx:8, Rd	B	2									Rd8–#xx:8	—	↕	↕	↕	↕	↕	2	
	CMP.B Rs, Rd	B		2								Rd8–Rs8	—	↕	↕	↕	↕	↕	2	
	CMP.W #xx:16, Rd	W	4									Rd16–#xx:16	—	(1)	↕	↕	↕	↕	4	
	CMP.W Rs, Rd	W		2								Rd16–Rs16	—	(1)	↕	↕	↕	↕	2	
	CMP.L #xx:32, ERd	L	6									ERd32–#xx:32	—	(2)	↕	↕	↕	↕	4	
	CMP.L ERs, ERd	L		2								ERd32–ERs32	—	(2)	↕	↕	↕	↕	2	

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
NEG	NEG.B Rd	B		2								0–Rd8 → Rd8	—	↕	↕	↕	↕	↕	2	
	NEG.W Rd	W		2								0–Rd16 → Rd16	—	↕	↕	↕	↕	↕	2	
	NEG.L ERd	L		2								0–ERd32 → ERd32	—	↕	↕	↕	↕	↕	2	
EXTU	EXTU.W Rd	W		2								0 → (<bits 15 to 8> of Rd16)	—	—	0	↕	0	—	2	
	EXTU.L ERd	L		2								0 → (<bits 31 to 16> of ERd32)	—	—	0	↕	0	—	2	
EXTS	EXTS.W Rd	W		2								(<bit 7> of Rd16) → (<bits 15 to 8> of Rd16)	—	—	↕	↕	0	—	2	
	EXTS.L ERd	L		2								(<bit 15> of ERd32) → (<bits 31 to 16> of ERd32)	—	—	↕	↕	0	—	2	

3. Logic Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
AND	AND.B #xx:8, Rd	B	2									Rd8∧#xx:8 → Rd8	—	—	↕	↕	0	—	2	
	AND.B Rs, Rd	B		2								Rd8∧Rs8 → Rd8	—	—	↕	↕	0	—	2	
	AND.W #xx:16, Rd	W	4									Rd16∧#xx:16 → Rd16	—	—	↕	↕	0	—	4	
	AND.W Rs, Rd	W		2								Rd16∧Rs16 → Rd16	—	—	↕	↕	0	—	2	
	AND.L #xx:32, ERd	L	6									ERd32∧#xx:32 → ERd32	—	—	↕	↕	0	—	6	
	AND.L ERs, ERd	L		4								ERd32∧ERs32 → ERd32	—	—	↕	↕	0	—	4	
OR	OR.B #xx:8, Rd	B	2									Rd8∨#xx:8 → Rd8	—	—	↕	↕	0	—	2	
	OR.B Rs, Rd	B		2								Rd8∨Rs8 → Rd8	—	—	↕	↕	0	—	2	
	OR.W #xx:16, Rd	W	4									Rd16∨#xx:16 → Rd16	—	—	↕	↕	0	—	4	
	OR.W Rs, Rd	W		2								Rd16∨Rs16 → Rd16	—	—	↕	↕	0	—	2	
	OR.L #xx:32, ERd	L	6									ERd32∨#xx:32 → ERd32	—	—	↕	↕	0	—	6	
	OR.L ERs, ERd	L		4								ERd32∨ERs32 → ERd32	—	—	↕	↕	0	—	4	
XOR	XOR.B #xx:8, Rd	B	2									Rd8⊕#xx:8 → Rd8	—	—	↕	↕	0	—	2	
	XOR.B Rs, Rd	B		2								Rd8⊕Rs8 → Rd8	—	—	↕	↕	0	—	2	
	XOR.W #xx:16, Rd	W	4									Rd16⊕#xx:16 → Rd16	—	—	↕	↕	0	—	4	
	XOR.W Rs, Rd	W		2								Rd16⊕Rs16 → Rd16	—	—	↕	↕	0	—	2	
	XOR.L #xx:32, ERd	L	6									ERd32⊕#xx:32 → ERd32	—	—	↕	↕	0	—	6	
	XOR.L ERs, ERd	L		4								ERd32⊕ERs32 → ERd32	—	—	↕	↕	0	—	4	
NOT	NOT.B Rd	B		2								¬ Rd8 → Rd8	—	—	↕	↕	0	—	2	
	NOT.W Rd	W		2								¬ Rd16 → Rd16	—	—	↕	↕	0	—	2	
	NOT.L ERd	L		2								¬ Rd32 → Rd32	—	—	↕	↕	0	—	2	

4. Shift Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
SHAL	SHAL.B Rd	B		2									—	—	↕	↕	↕	↕	2	
	SHAL.W Rd	W		2									—	—	↕	↕	↕	↕	2	
	SHAL.L ERd	L		2									—	—	↕	↕	↕	↕	2	
SHAR	SHAR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	SHAR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	SHAR.L ERd	L		2									—	—	↕	↕	0	↕	2	
SHLL	SHLL.B Rd	B		2									—	—	↕	↕	0	↕	2	
	SHLL.W Rd	W		2									—	—	↕	↕	0	↕	2	
	SHLL.L ERd	L		2									—	—	↕	↕	0	↕	2	
SHLR	SHLR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	SHLR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	SHLR.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTXL	ROTXL.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTXL.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTXL.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTXR	ROTXR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTXR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTXR.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTL	ROTL.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTL.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTL.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTR	ROTR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTR.L ERd	L		2									—	—	↕	↕	0	↕	2	

5. Bit-Manipulation Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	|		I	H	N	Z	V	C	Normal	Advanced
BSET	BSET #xx:3, Rd	B		2								(#xx:3 of Rd8) ← 1	—	—	—	—	—	—	2	
	BSET #xx:3, @ERd	B			4							(#xx:3 of @ERd) ← 1	—	—	—	—	—	—	8	
	BSET #xx:3, @aa:8	B						4				(#xx:3 of @aa:8) ← 1	—	—	—	—	—	—	8	
	BSET Rn, Rd	B		2								(Rn8 of Rd8) ← 1	—	—	—	—	—	—	2	
	BSET Rn, @ERd	B			4							(Rn8 of @ERd) ← 1	—	—	—	—	—	—	8	
	BSET Rn, @aa:8	B						4				(Rn8 of @aa:8) ← 1	—	—	—	—	—	—	8	
BCLR	BCLR #xx:3, Rd	B		2								(#xx:3 of Rd8) ← 0	—	—	—	—	—	—	2	
	BCLR #xx:3, @ERd	B			4							(#xx:3 of @ERd) ← 0	—	—	—	—	—	—	8	
	BCLR #xx:3, @aa:8	B						4				(#xx:3 of @aa:8) ← 0	—	—	—	—	—	—	8	
	BCLR Rn, Rd	B		2								(Rn8 of Rd8) ← 0	—	—	—	—	—	—	2	
	BCLR Rn, @ERd	B			4							(Rn8 of @ERd) ← 0	—	—	—	—	—	—	8	
	BCLR Rn, @aa:8	B						4				(Rn8 of @aa:8) ← 0	—	—	—	—	—	—	8	
BNOT	BNOT #xx:3, Rd	B		2								(#xx:3 of Rd8) ← ¬ (#xx:3 of Rd8)	—	—	—	—	—	—	2	
	BNOT #xx:3, @ERd	B			4							(#xx:3 of @ERd) ← ¬ (#xx:3 of @ERd)	—	—	—	—	—	—	8	
	BNOT #xx:3, @aa:8	B						4				(#xx:3 of @aa:8) ← ¬ (#xx:3 of @aa:8)	—	—	—	—	—	—	8	
	BNOT Rn, Rd	B		2								(Rn8 of Rd8) ← ¬ (Rn8 of Rd8)	—	—	—	—	—	—	2	
	BNOT Rn, @ERd	B			4							(Rn8 of @ERd) ← ¬ (Rn8 of @ERd)	—	—	—	—	—	—	8	
	BNOT Rn, @aa:8	B						4				(Rn8 of @aa:8) ← ¬ (Rn8 of @aa:8)	—	—	—	—	—	—	8	
BTST	BTST #xx:3, Rd	B		2								¬ (#xx:3 of Rd8) → Z	—	—	—	↕	—	—	2	
	BTST #xx:3, @ERd	B			4							¬ (#xx:3 of @ERd) → Z	—	—	—	↕	—	—	6	
	BTST #xx:3, @aa:8	B						4				¬ (#xx:3 of @aa:8) → Z	—	—	—	↕	—	—	6	
	BTST Rn, Rd	B		2								¬ (Rn8 of @Rd8) → Z	—	—	—	↕	—	—	2	
	BTST Rn, @ERd	B			4							¬ (Rn8 of @ERd) → Z	—	—	—	↕	—	—	6	
	BTST Rn, @aa:8	B						4				¬ (Rn8 of @aa:8) → Z	—	—	—	↕	—	—	6	
BLD	BLD #xx:3, Rd	B		2								(#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
BLD	BLD #xx:3, @ERd	B			4							(#xx:3 of @ERd) → C	—	—	—	—	—	↕	6	
	BLD #xx:3, @aa:8	B						4				(#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BILD	BILD #xx:3, Rd	B		2								¬ (#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BILD #xx:3, @ERd	B			4							¬ (#xx:3 of @ERd) → C	—	—	—	—	—	↕	6	
	BILD #xx:3, @aa:8	B						4				¬ (#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BST	BST #xx:3, Rd	B		2								C → (#xx:3 of Rd8)	—	—	—	—	—	—	2	
	BST #xx:3, @ERd	B			4							C → (#xx:3 of @ERd24)	—	—	—	—	—	—	8	
	BST #xx:3, @aa:8	B						4				C → (#xx:3 of @aa:8)	—	—	—	—	—	—	8	
BIST	BIST #xx:3, Rd	B		2								¬ C → (#xx:3 of Rd8)	—	—	—	—	—	—	2	
	BIST #xx:3, @ERd	B			4							¬ C → (#xx:3 of @ERd24)	—	—	—	—	—	—	8	
	BIST #xx:3, @aa:8	B						4				¬ C → (#xx:3 of @aa:8)	—	—	—	—	—	—	8	
BAND	BAND #xx:3, Rd	B		2								C∧(#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BAND #xx:3, @ERd	B			4							C∧(#xx:3 of @ERd24) → C	—	—	—	—	—	↕	6	
	BAND #xx:3, @aa:8	B						4				C∧(#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BIAND	BIAND #xx:3, Rd	B		2								C∧ ¬ (#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BIAND #xx:3, @ERd	B			4							C∧ ¬ (#xx:3 of @ERd24) → C	—	—	—	—	—	↕	6	
	BIAND #xx:3, @aa:8	B						4				C∧ ¬ (#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BOR	BOR #xx:3, Rd	B		2								C∨(#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BOR #xx:3, @ERd	B			4							C∨(#xx:3 of @ERd24) → C	—	—	—	—	—	↕	6	
	BOR #xx:3, @aa:8	B						4				C∨(#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BIOR	BIOR #xx:3, Rd	B		2								C∨ ¬ (#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BIOR #xx:3, @ERd	B			4							C∨ ¬ (#xx:3 of @ERd24) → C	—	—	—	—	—	↕	6	
	BIOR #xx:3, @aa:8	B						4				C∨ ¬ (#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BXOR	BXOR #xx:3, Rd	B		2								C⊕(#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BXOR #xx:3, @ERd	B			4							C⊕(#xx:3 of @ERd24) → C	—	—	—	—	—	↕	6	
	BXOR #xx:3, @aa:8	B						4				C⊕(#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	
BIXOR	BIXOR #xx:3, Rd	B		2								C⊕ ¬ (#xx:3 of Rd8) → C	—	—	—	—	—	↕	2	
	BIXOR #xx:3, @ERd	B			4							C⊕ ¬ (#xx:3 of @ERd24) → C	—	—	—	—	—	↕	6	
	BIXOR #xx:3, @aa:8	B						4				C⊕ ¬ (#xx:3 of @aa:8) → C	—	—	—	—	—	↕	6	

6. Branching Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	|	Branch Condition	I	H	N	Z	V	C	Normal	Advanced
Bcc	BRA d:8 (BT d:8)	—							2			If condition is true then PC ← PC+d else next; Always	—	—	—	—	—	—	4	
	BRA d:16 (BT d:16)	—							4				—	—	—	—	—	—	6	
	BRN d:8 (BF d:8)	—							2			Never	—	—	—	—	—	—	4	
	BRN d:16 (BF d:16)	—							4				—	—	—	—	—	—	6	
	BHI d:8	—							2			C∨ Z = 0	—	—	—	—	—	—	4	
	BHI d:16	—							4				—	—	—	—	—	—	6	
	BLS d:8	—							2			C∨ Z = 1	—	—	—	—	—	—	4	
	BLS d:16	—							4				—	—	—	—	—	—	6	
	BCC d:8 (BHS d:8)	—							2			C = 0	—	—	—	—	—	—	4	
	BCC d:16 (BHS d:16)	—							4				—	—	—	—	—	—	6	
	BCS d:8 (BLO d:8)	—							2			C = 1	—	—	—	—	—	—	4	
	BCS d:16 (BLO d:16)	—							4				—	—	—	—	—	—	6	
	BNE d:8	—							2			Z = 0	—	—	—	—	—	—	4	
	BNE d:16	—							4				—	—	—	—	—	—	6	
	BEQ d:8	—							2			Z = 1	—	—	—	—	—	—	4	
	BEQ d:16	—							4				—	—	—	—	—	—	6	
	BVC d:8	—							2			V = 0	—	—	—	—	—	—	4	
	BVC d:16	—							4				—	—	—	—	—	—	6	
	BVS d:8	—							2			V = 1	—	—	—	—	—	—	4	
	BVS d:16	—							4				—	—	—	—	—	—	6	
	BPL d:8	—							2			N = 0	—	—	—	—	—	—	4	
	BPL d:16	—							4				—	—	—	—	—	—	6	
	BMI d:8	—							2			N = 1	—	—	—	—	—	—	4	
	BMI d:16	—							4				—	—	—	—	—	—	6	
	BGE d:8	—							2			N⊕V = 0	—	—	—	—	—	—	4	
	BGE d:16	—							4				—	—	—	—	—	—	6	
	BLT d:8	—							2			N⊕V = 1	—	—	—	—	—	—	4	
	BLT d:16	—							4				—	—	—	—	—	—	6	
	BGT d:8	—							2			Z∨ (N⊕V) = 0	—	—	—	—	—	—	4	
	BGT d:16	—							4				—	—	—	—	—	—	6	
	BLE d:8	—							2			Z∨ (N⊕V) = 1	—	—	—	—	—	—	4	
	BLE d:16	—							4				—	—	—	—	—	—	6	

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
JMP	JMP @ERn	—			2							PC ← ERn	—	—	—	—	—	—	4	
	JMP @aa:24	—						4				PC ← aa:24	—	—	—	—	—	—	6	
	JMP @@aa:8	—								2		PC ← @aa:8	—	—	—	—	—	—	8	10
BSR	BSR d:8	—							2			PC → @–SP PC ← PC+d:8	—	—	—	—	—	—	6	8
	BSR d:16	—							4			PC → @–SP PC ← PC+d:16	—	—	—	—	—	—	8	10
JSR	JSR @ERn	—			2							PC → @–SP PC ← ERn	—	—	—	—	—	—	6	8
	JSR @aa:24	—						4				PC → @–SP PC ← aa:24	—	—	—	—	—	—	8	10
	JSR @@aa:8	—								2		PC → @–SP PC ← @aa:8	—	—	—	—	—	—	8	12
RTS	RTS	—									2	PC ← @SP+	—	—	—	—	—	—	8	10

7. System Control Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
RTE	RTE	—										CCR ← @SP+ PC ← @SP+	↕	↕	↕	↕	↕	↕	10	
SLEEP	SLEEP	—										Transition to power-down state							2	
LDC	LDC #xx:8, CCR	B	2									#xx:8 → CCR	↕	↕	↕	↕	↕	↕	2	
	LDC Rs, CCR	B		2								Rs8 → CCR	↕	↕	↕	↕	↕	↕	2	
	LDC @ERs, CCR	W			4							@ERs → CCR	↕	↕	↕	↕	↕	↕	6	
	LDC @(d:16, ERs), CCR	W				6						@(d:16, ERs) → CCR	↕	↕	↕	↕	↕	↕	8	
	LDC @(d:24, ERs), CCR	W				10						@(d:24, ERs) → CCR	↕	↕	↕	↕	↕	↕	12	
	LDC @ERs+, CCR	W					4					@ERs → CCR ERs32+2 → ERs32	↕	↕	↕	↕	↕	↕	8	
	LDC @aa:16, CCR	W						6				@aa:16 → CCR	↕	↕	↕	↕	↕	↕	8	
	LDC @aa:24, CCR	W						8				@aa:24 → CCR	↕	↕	↕	↕	↕	↕	10	
STC	STC CCR, Rd	B		2								CCR → Rd8							2	
	STC CCR, @ERd	W			4							CCR → @ERd							6	
	STC CCR, @(d:16, ERd)	W				6						CCR → @(d:16, ERd)							8	
	STC CCR, @(d:24, ERd)	W				10						CCR → @(d:24, ERd)							12	
	STC CCR, @–ERd	W					4					ERd32–2 → ERd32 CCR → @ERd							8	
	STC CCR, @aa:16	W						6				CCR → @aa:16							8	
	STC CCR, @aa:24	W						8				CCR → @aa:24							10	
ANDC	ANDC #xx:8, CCR	B	2									CCR∧#xx:8 → CCR	↕	↕	↕	↕	↕	↕	2	
ORC	ORC #xx:8, CCR	B	2									CCR∨#xx:8 → CCR	↕	↕	↕	↕	↕	↕	2	
XORC	XORC #xx:8, CCR	B	2									CCR⊕#xx:8 → CCR	↕	↕	↕	↕	↕	↕	2	
NOP	NOP	—									2	PC ← PC+2							2	

8. Block Transfer Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	|		I	H	N	Z	V	C	Normal	Advanced
EEPMOV	EEPMOV. B	—									4	if R4L ≠ 0 then repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4L–1 → R4L until R4L=0 else next	—	—	—	—	—	—	8+ 4n*2	
	EEPMOV. W	—									4	if R4 ≠ 0 then repeat @R5 → @R6 R5+1 → R5 R6+1 → R6 R4–1 → R4 until R4=0 else next	—	—	—	—	—	—	8+ 4n*2	

Notes: 1. The number of states in cases where the instruction code and its operands are located in on-chip memory is shown here. For other cases, see appendix A.3, Number of Execution States.

2. n is the value set in register R4L or R4.

(1) Set to 1 when a carry or borrow occurs at bit 11; otherwise cleared to 0.

(2) Set to 1 when a carry or borrow occurs at bit 27; otherwise cleared to 0.

(3) Retains its previous value when the result is zero; otherwise cleared to 0.

(4) Set to 1 when the adjustment produces a carry; otherwise retains its previous value.

(5) The number of states required for execution of an instruction that transfers data in synchronization with the E clock is variable.

(6) Set to 1 when the divisor is negative; otherwise cleared to 0.

(7) Set to 1 when the divisor is zero; otherwise cleared to 0.

(8) Set to 1 when the quotient is negative; otherwise cleared to 0.

A.2 Operation Code Map

Table A.2 Operation Code Map (1)

Instruction code:

1st byte		2nd byte	
AH	AL	BH	BL

Instruction when most significant bit of BH is 0.

Instruction when most significant bit of BH is 1.

AH \ AL	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	Table A-2 (2)	STC	LDC	ORC	XORC	ANDC	LDC	ADD		Table A-2 (2)	Table A-2 (2)	MOV		ADDX	Table A-2 (2)
1	Table A-2 (2)	Table A-2 (2)	Table A-2 (2)	Table A-2 (2)	OR.B	XOR.B	AND.B	Table A-2 (2)	SUB		Table A-2 (2)	Table A-2 (2)	CMP		SUBX	Table A-2 (2)
2	MOV.B															
3																
4	BRA	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU	MULXU	DIVXU	RTS	BSR	RTE		Table A-2 (2)	JMP		BSR	JSR			
6	BSET	BNOT	BCLR	BTST	OR	XOR	AND	BST/BIST	MOV							
7					BOR/BIOR	BXOR/BIXOR	BAND/BIAND	BLD/BILD		Table A-2 (2)	Table A-2 (2)	EEPMOV	Table A-2 (3)			
8	ADD															
9	ADDX															
A	CMP															
B	SUBX															
C	OR															
D	XOR															
E	AND															
F	MOV															

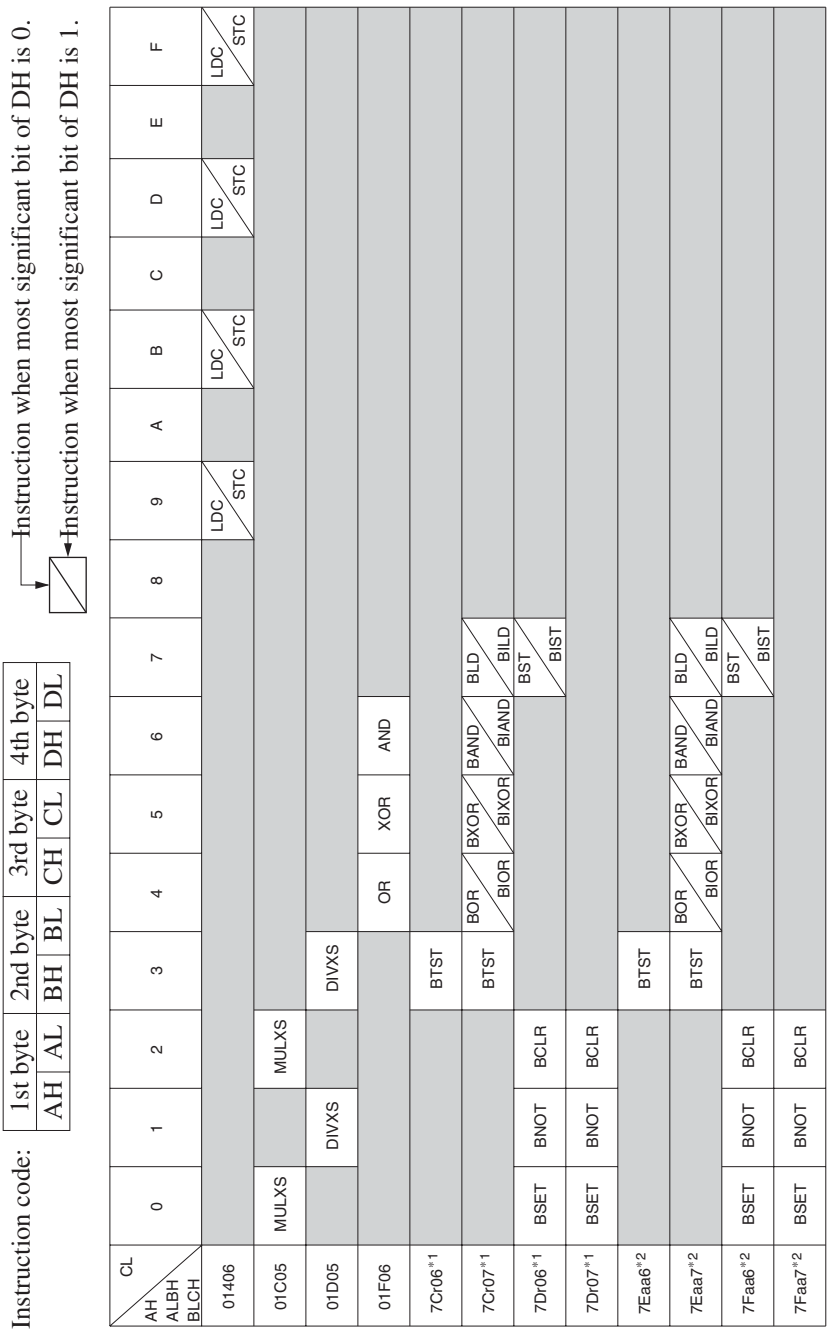
Table A.2 Operation Code Map (2)

Instruction code:

1st byte		2nd byte	
AH	AL	BH	BL

AH AL \ BH	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
01	MOV				LDC/STC				SLEEP				Table A-2 (3)	Table A-2 (3)		Table A-2 (3)
0A	INC												ADD			
0B	ADDS					INC		INC	ADDS					INC		INC
0F	DAA												MOV			
10	SHLL	SHLL		SHLL					SHAL	SHAL		SHAL				
11	SHLR	SHLR		SHLR					SHAR	SHAR		SHAR				
12	ROTXL	ROTXL		ROTXL					ROTL	ROTL		ROTL				
13	ROTXR	ROTXR		ROTXR					ROTR	ROTR		ROTR				
17	NOT	NOT		NOT		EXTU		EXTU	NEG	NEG		NEG		EXTS		EXTS
1A	DEC												SUB			
1B	SUBS					DEC		DEC	SUB	SUB				DEC		DEC
1F	DAS												CMP			
58	BRA	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
79	MOV	ADD	CMP	SUB	OR	XOR	AND									
7A	MOV	ADD	CMP	SUB	OR	XOR	AND									

Table A.2 Operation Code Map (3)



Notes: 1. r is the register designation field.
2. aa is the absolute address field.

A.3 Number of Execution States

The status of execution for each instruction of the H8/300H CPU and the method of calculating the number of states required for instruction execution are shown below. Table A.4 shows the number of cycles of each type occurring in each instruction, such as instruction fetch and data read/write. Table A.3 shows the number of states required for each cycle. The total number of states required for execution of an instruction can be calculated by the following expression:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

Examples: When instruction is fetched from on-chip ROM, and an on-chip RAM is accessed.

BSET #0, @FF00

From table A.4:
I = L = 2, J = K = M = N= 0

From table A.3:
$S_I = 2$, $S_L = 2$

Number of states required for execution = $2 \times 2 + 2 \times 2 = 8$

When instruction is fetched from on-chip ROM, branch address is read from on-chip ROM, and on-chip RAM is used for stack area.

JSR @@ 30

From table A.4:
I = 2, J = K = 1, L = M = N = 0

From table A.3:
$S_I = S_J = S_K = 2$

Number of states required for execution = $2 \times 2 + 1 \times 2 + 1 \times 2 = 8$

Table A.3 Number of Cycles in Each Instruction

Execution Status (Instruction Cycle)		Access Location	
		On-Chip Memory	On-Chip Peripheral Module
Instruction fetch	S_I	2	—
Branch address read	S_J		
Stack operation	S_K		
Byte data access	S_L		2 or 3*
Word data access	S_M		—
Internal operation	S_N	1	

Note: * Depends on which on-chip peripheral module is accessed. See section 13.1, Register Addresses (Address Order).

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 #xx:16, Rd	2					
	ADD.W Rs, Rd	1					
	ADD.L #xx:32, ERd	3					
	ADD.L ERs, ERd	1					
ADDS	ADDS #1/2/4, ERd	1					
ADDX	ADDX #xx:8, Rd	1					
	ADDX Rs, Rd	1					
AND	AND.B #xx:8, Rd	1					
	AND.B Rs, Rd	1					
	AND.W #xx:16, Rd	2					
	AND.W Rs, Rd	1					
	AND.L #xx:32, ERd	3					
	AND.L ERs, ERd	2					
ANDC	ANDC #xx:8, CCR	1					
BAND	BAND #xx:3, Rd	1					
	BAND #xx:3, @ERd	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					

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
Bcc	BLT d:8	2					
	BGT d:8	2					
	BLE d:8	2					
	BRA d:16(BT d:16)	2					2
	BRN d:16(BF d:16)	2					2
	BHI d:16	2					2
	BLS d:16	2					2
	BCC d:16(BHS d:16)	2					2
	BCS d:16(BLO d:16)	2					2
	BNE d:16	2					2
	BEQ d:16	2					2
	BVC d:16	2					2
	BVS d:16	2					2
	BPL d:16	2					2
	BMI d:16	2					2
	BGE d:16	2					2
	BLT d:16	2					2
	BGT d:16	2					2
	BLE d:16	2					2
BCLR	BCLR #xx:3, Rd	1					
	BCLR #xx:3, @ERd	2			2		
	BCLR #xx:3, @aa:8	2			2		
	BCLR Rn, Rd	1					
	BCLR Rn, @ERd	2			2		
	BCLR Rn, @aa:8	2			2		
BIAND	BIAND #xx:3, Rd	1					
	BIAND #xx:3, @ERd	2			1		
	BIAND #xx:3, @aa:8	2			1		
BILD	BILD #xx:3, Rd	1					
	BILD #xx:3, @ERd	2			1		
	BILD #xx:3, @aa:8	2			1		

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BIOR	BIOR #xx:3, Rd	1					
	BIOR #xx:3, @ERd	2			1		
	BIOR #xx:3, @aa:8	2			1		
BIST	BIST #xx:3, Rd	1					
	BIST #xx:3, @ERd	2			2		
	BIST #xx:3, @aa:8	2			2		
BIXOR	BIXOR #xx:3, Rd	1					
	BIXOR #xx:3, @ERd	2			1		
	BIXOR #xx:3, @aa:8	2			1		
BLD	BLD #xx:3, Rd	1					
	BLD #xx:3, @ERd	2			1		
	BLD #xx:3, @aa:8	2			1		
BNOT	BNOT #xx:3, Rd	1					
	BNOT #xx:3, @ERd	2			2		
	BNOT #xx:3, @aa:8	2			2		
	BNOT Rn, Rd	1					
	BNOT Rn, @ERd	2			2		
	BNOT Rn, @aa:8	2			2		
BOR	BOR #xx:3, Rd	1					
	BOR #xx:3, @ERd	2			1		
	BOR #xx:3, @aa:8	2			1		
BSET	BSET #xx:3, Rd	1					
	BSET #xx:3, @ERd	2			2		
	BSET #xx:3, @aa:8	2			2		
	BSET Rn, Rd	1					
	BSET Rn, @ERd	2			2		
	BSET Rn, @aa:8	2			2		
BSR	BSR d:8	2		1			
	BSR d:16	2		1			2
BST	BST #xx:3, Rd	1					
	BST #xx:3, @ERd	2			2		
	BST #xx:3, @aa:8	2			2		

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
BTST	BTST #xx:3, Rd	1					
	BTST #xx:3, @ERd	2			1		
	BTST #xx:3, @aa:8	2			1		
	BTST Rn, Rd	1					
	BTST Rn, @ERd	2			1		
	BTST Rn, @aa:8	2			1		
BXOR	BXOR #xx:3, Rd	1					
	BXOR #xx:3, @ERd	2			1		
	BXOR #xx:3, @aa:8	2			1		
CMP	CMP.B #xx:8, Rd	1					
	CMP.B Rs, Rd	1					
	CMP.W #xx:16, Rd	2					
	CMP.W Rs, Rd	1					
	CMP.L #xx:32, ERd	3					
	CMP.L ERs, ERd	1					
DAA	DAA Rd	1					
DAS	DAS Rd	1					
DEC	DEC.B Rd	1					
	DEC.W #1/2, Rd	1					
	DEC.L #1/2, ERd	1					
DUVXS	DIVXS.B Rs, Rd	2					12
	DIVXS.W Rs, ERd	2					20
DIVXU	DIVXU.B Rs, Rd	1					12
	DIVXU.W Rs, ERd	1					20
EEPMOV	EEPMOV.B	2			2n+2*[1]		
	EEPMOV.W	2			2n+2*[1]		
EXTS	EXTS.W Rd	1					
	EXTS.L ERd	1					
EXTU	EXTU.W Rd	1					
	EXTU.L ERd	1					

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
INC	INC.B Rd	1					
	INC.W #1/2, Rd	1					
	INC.L #1/2, ERd	1					
JMP	JMP @ERn	2					
	JMP @aa:24	2					2
	JMP @@aa:8	2	1				2
JSR	JSR @ERn	2		1			
	JSR @aa:24	2		1			2
	JSR @@aa:8	2	1	1			
LDC	LDC #xx:8, CCR	1					
	LDC Rs, CCR	1					
	LDC@ERs, CCR	2				1	
	LDC@(d:16, ERs), CCR	3				1	
	LDC@(d:24,ERs), CCR	5				1	
	LDC@ERs+, CCR	2				1	2
	LDC@aa:16, CCR	3				1	
	LDC@aa:24, CCR	4				1	
MOV	MOV.B #xx:8, Rd	1					
	MOV.B Rs, Rd	1					
	MOV.B @ERs, Rd	1			1		
	MOV.B @(d:16, ERs), Rd	2			1		
	MOV.B @(d:24, ERs), Rd	4			1		
	MOV.B @ERs+, Rd	1			1		2
	MOV.B @aa:8, Rd	1			1		
	MOV.B @aa:16, Rd	2			1		
	MOV.B @aa:24, Rd	3			1		
	MOV.B Rs, @Erd	1			1		
	MOV.B Rs, @(d:16, ERd)	2			1		
	MOV.B Rs, @(d:24, ERd)	4			1		
	MOV.B Rs, @-ERd	1			1		2
	MOV.B Rs, @aa:8	1			1		

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
MOV	MOV.B Rs, @aa:16	2			1		
	MOV.B Rs, @aa:24	3			1		
	MOV.W #xx:16, Rd	2					
	MOV.W Rs, Rd	1					
	MOV.W @ERs, Rd	1				1	
	MOV.W @(d:16,ERs), Rd	2				1	
	MOV.W @(d:24,ERs), Rd	4				1	
	MOV.W @ERs+, Rd	1				1	2
	MOV.W @aa:16, Rd	2				1	
	MOV.W @aa:24, Rd	3				1	
	MOV.W Rs, @ERd	1				1	
	MOV.W Rs, @(d:16,ERd)	2				1	
	MOV.W Rs, @(d:24,ERd)	4				1	
MOV	MOV.W Rs, @-ERd	1				1	2
	MOV.W Rs, @aa:16	2				1	
	MOV.W Rs, @aa:24	3				1	
	MOV.L #xx:32, ERd	3					
	MOV.L ERs, ERd	1					
	MOV.L @ERs, ERd	2				2	
	MOV.L @(d:16,ERs), ERd	3				2	
	MOV.L @(d:24,ERs), ERd	5				2	
	MOV.L @ERs+, ERd	2				2	2
	MOV.L @aa:16, ERd	3				2	
	MOV.L @aa:24, ERd	4				2	
	MOV.L ERs,@ERd	2				2	
	MOV.L ERs, @(d:16,ERd)	3				2	
	MOV.L ERs, @(d:24,ERd)	5				2	
	MOV.L ERs, @-ERd	2				2	2
	MOV.L ERs, @aa:16	3				2	
	MOV.L ERs, @aa:24	4				2	
MOVFPE	MOVFPE @aa:16, Rd*2	2			1		
MOVTPE	MOVTPE Rs,@aa:16*2	2			1		

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
MULXS	MULXS.B Rs, Rd	2					12
	MULXS.W Rs, ERd	2					20
MULXU	MULXU.B Rs, Rd	1					12
	MULXU.W Rs, ERd	1					20
NEG	NEG.B Rd	1					
	NEG.W Rd	1					
	NEG.L ERd	1					
NOP	NOP	1					
NOT	NOT.B Rd	1					
	NOT.W Rd	1					
	NOT.L ERd	1					
OR	OR.B #xx:8, Rd	1					
	OR.B Rs, Rd	1					
	OR.W #xx:16, Rd	2					
	OR.W Rs, Rd	1					
	OR.L #xx:32, ERd	3					
	OR.L ERs, ERd	2					
ORC	ORC #xx:8, CCR	1					
POP	POP.W Rn	1				1	2
	POP.L ERn	2				2	2
PUSH	PUSH.W Rn	1				1	2
	PUSH.L ERn	2				2	2
ROTL	ROTL.B Rd	1					
	ROTL.W Rd	1					
	ROTL.L ERd	1					
ROTR	ROTR.B Rd	1					
	ROTR.W Rd	1					
	ROTR.L ERd	1					
ROTXL	ROTXL.B Rd	1					
	ROTXL.W Rd	1					
	ROTXL.L ERd	1					

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
ROTXR	ROTXR.B Rd	1					
	ROTXR.W Rd	1					
	ROTXR.L ERd	1					
RTE	RTE	2		2			2
RTS	RTS	2		1			2
SHAL	SHAL.B Rd	1					
	SHAL.W Rd	1					
	SHAL.L ERd	1					
SHAR	SHAR.B Rd	1					
	SHAR.W Rd	1					
	SHAR.L ERd	1					
SHLL	SHLL.B Rd	1					
	SHLL.W Rd	1					
	SHLL.L ERd	1					
SHLR	SHLR.B Rd	1					
	SHLR.W Rd	1					
	SHLR.L ERd	1					
SLEEP	SLEEP	1					
STC	STC CCR, Rd	1					
	STC CCR, @ERd	2				1	
	STC CCR, @(d:16,ERd)	3				1	
	STC CCR, @(d:24,ERd)	5				1	
	STC CCR,@-ERd	2				1	2
	STC CCR, @aa:16	3				1	
	STC CCR, @aa:24	4				1	
SUB	SUB.B Rs, Rd	1					
	SUB.W #xx:16, Rd	2					
	SUB.W Rs, Rd	1					
	SUB.L #xx:32, ERd	3					
	SUB.L ERs, ERd	1					
SUBS	SUBS #1/2/4, ERd	1					

Instruction	Mnemonic	Instruction Fetch I	Branch Addr. Read J	Stack Operation K	Byte Data Access L	Word Data Access M	Internal Operation N
SUBX	SUBX #xx:8, Rd	1					
	SUBX. Rs, Rd	1					
XOR	XOR.B #xx:8, Rd	1					
	XOR.B Rs, Rd	1					
	XOR.W #xx:16, Rd	2					
	XOR.W Rs, Rd	1					
	XOR.L #xx:32, ERd	3					
	XOR.L ERs, ERd	2					
XORC	XORC #xx:8, CCR	1					

Notes: 1. n: Specified value in R4L. The source and destination operands are accessed n+1 times respectively.

2. It cannot be used in this LSI.

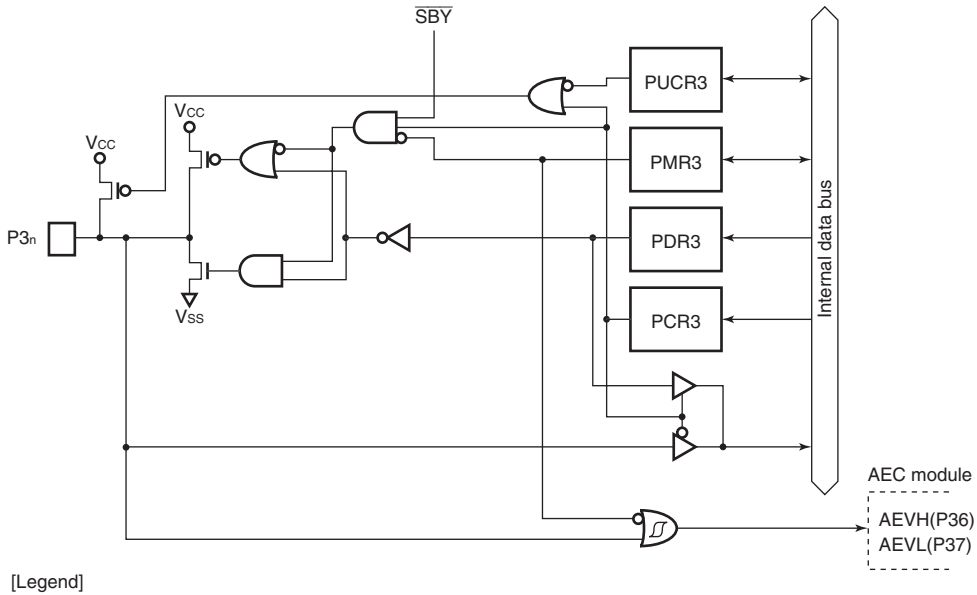
A.4 Combinations of Instructions and Addressing Modes

Table A.5 Combinations of Instructions and Addressing Modes

Functions	Instructions	Addressing Mode												
		#xx	Rn	@ERn	@(d:16.ERn)	@(d:24.ERn)	@ERn+/@ERn	@aa:8	@aa:16	@aa:24	@(d:8.PC)	@(d:16.PC)	@@aa:8	|
Data transfer instructions	MOV	BWL	BWL	BWL	BWL	BWL	BWL	B	BWL	BWL	—	—	—	—
	POP, PUSH	—	—	—	—	—	—	—	—	—	—	—	—	WL
	MOVFPE, MOVTPE	—	—	—	—	—	—	—	—	—	—	—	—	—
Arithmetic operations	ADD, CMP	BWL	BWL	—	—	—	—	—	—	—	—	—	—	—
	SUB	WL	BWL	—	—	—	—	—	—	—	—	—	—	—
	ADDX, SUBX	B	B	—	—	—	—	—	—	—	—	—	—	—
	ADDS, SUBS	—	L	—	—	—	—	—	—	—	—	—	—	—
	INC, DEC	—	BWL	—	—	—	—	—	—	—	—	—	—	—
	DAA, DAS	—	B	—	—	—	—	—	—	—	—	—	—	—
	MULXU, MULXS, DIVXU, DIVXS	—	BW	—	—	—	—	—	—	—	—	—	—	—
	NEG	—	BWL	—	—	—	—	—	—	—	—	—	—	—
	EXTU, EXTS	—	WL	—	—	—	—	—	—	—	—	—	—	—
Logical operations	AND, OR, XOR	—	BWL	—	—	—	—	—	—	—	—	—	—	—
	NOT	—	BWL	—	—	—	—	—	—	—	—	—	—	—
Shift operations		—	BWL	—	—	—	—	—	—	—	—	—	—	—
Bit manipulations		—	B	B	—	—	—	B	—	—	—	—	—	—
Branching instructions	BCC, BSR	—	—	—	—	—	—	—	—	—	—	—	—	—
	JMP, JSR	—	—	◯	—	—	—	—	—	—	◯	◯	—	—
	RTS	—	—	—	—	—	—	—	—	◯	—	—	◯	—
System control instructions	RTE	—	—	—	—	—	—	—	—	—	—	—	—	◯
	SLEEP	—	—	—	—	—	—	—	—	—	—	—	—	◯
	LDC	B	B	W	W	W	W	—	W	W	—	—	—	◯
	STC	—	B	W	W	W	W	—	W	W	—	—	—	—
	ANDC, ORC, XORC	B	—	—	—	—	—	—	—	—	—	—	—	—
	NOP	—	—	—	—	—	—	—	—	—	—	—	—	◯
Block data transfer instructions		—	—	—	—	—	—	—	—	—	—	—	—	BW

B. I/O Port Block Diagrams

B.1 Port 3 Block Diagrams



[Legend]
PDR3: Port data register 3
PCR3: Port control register 3
PMR3: Port mode register 3
PUCR3: Port pull-up control register 3
n = 7 or 6

Figure B.1(a) Port 3 Block Diagram (Pins P37 and P36)

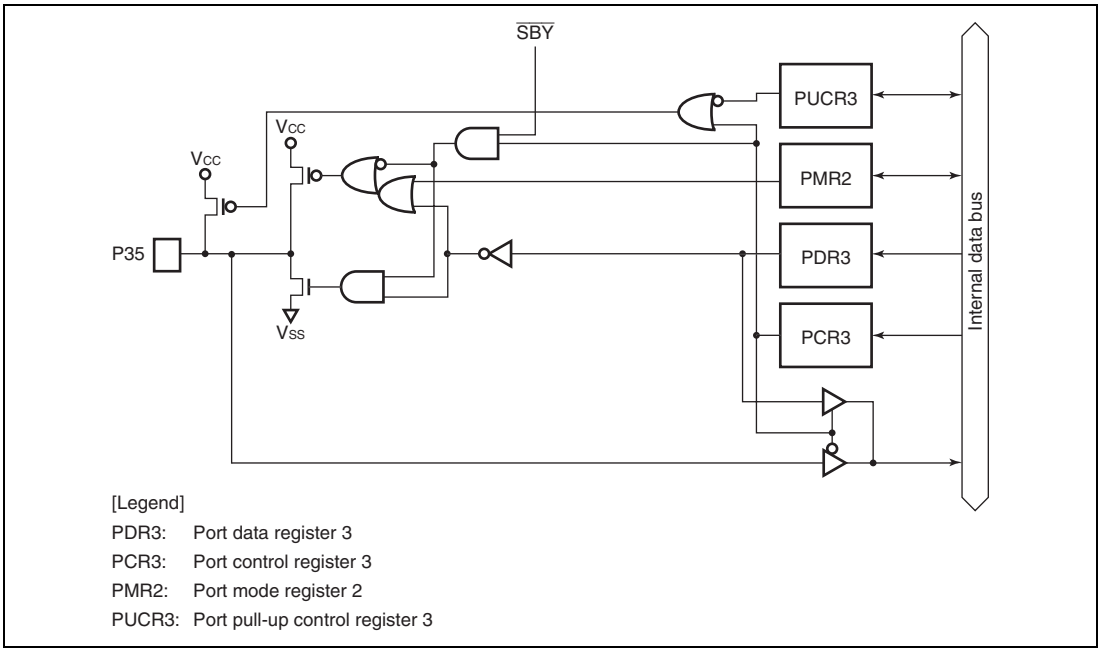


Figure B.1(b) Port 3 Block Diagram (Pin P35)

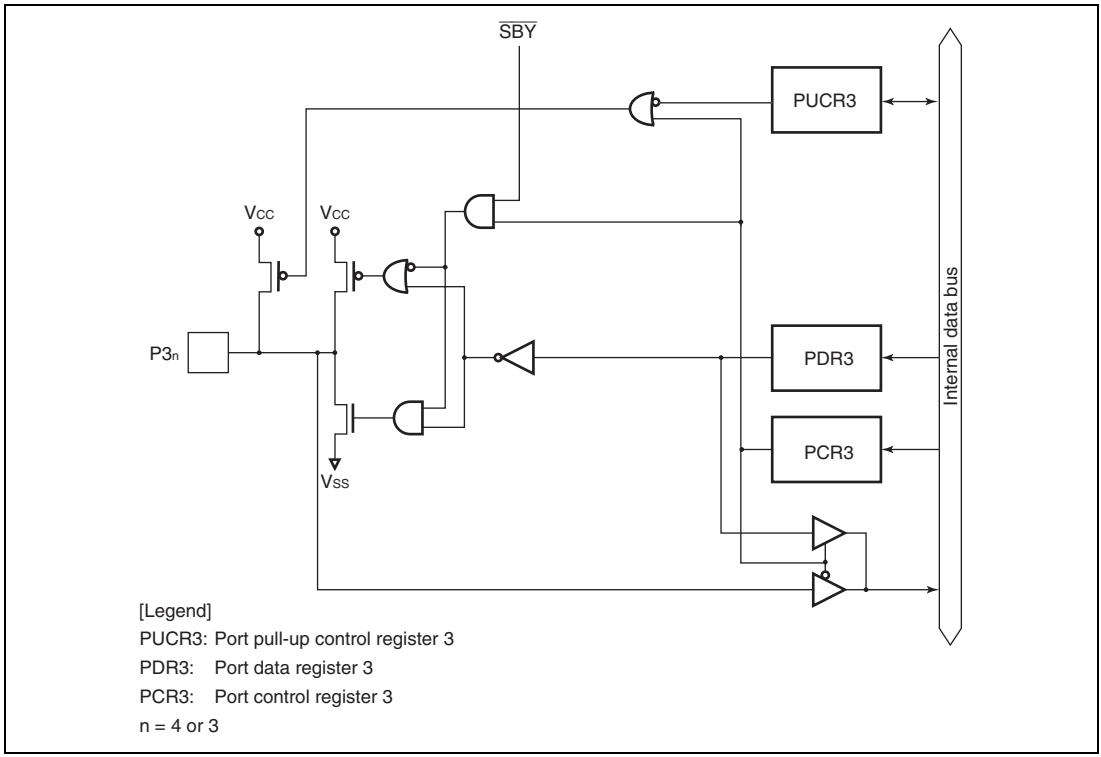
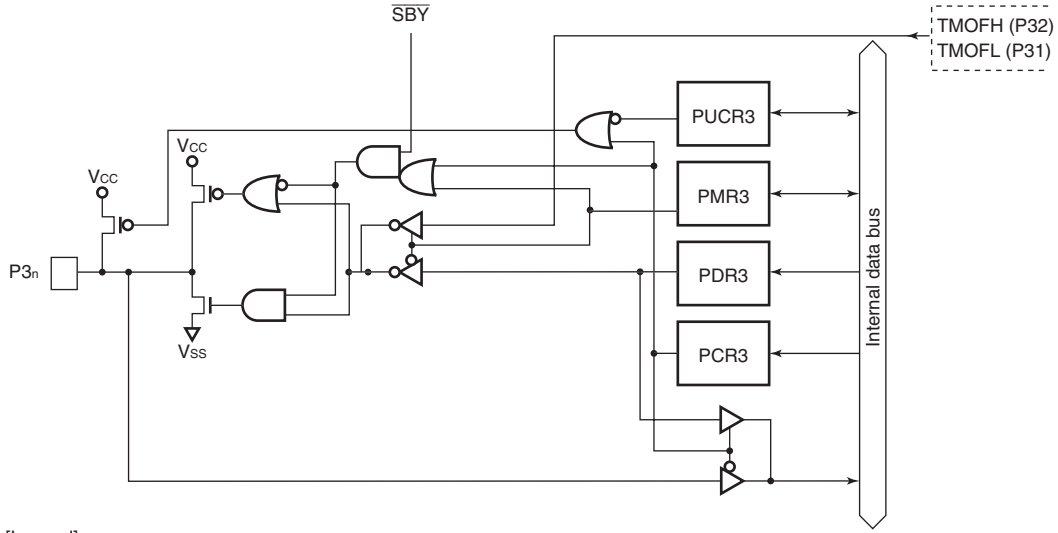


Figure B.1(c) Port 3 Block Diagram (Pins P34 and P33)



[Legend]

PDR3: Port data register 3

PCR3: Port control register 3

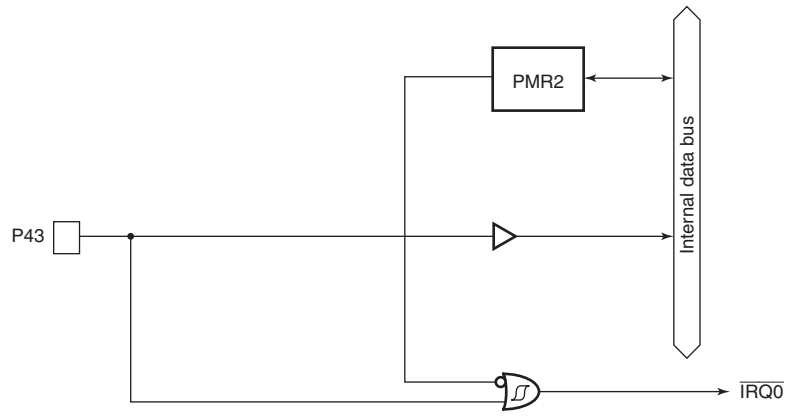
PMR3: Port mode register 3

PUCR3: Port pull-up control register 3

n = 2 or 1

Figure B.1(d) Port 3 Block Diagram (Pins P32 and P31)

B.2 Port 4 Block Diagrams



[Legend]
PMR2: Port mode register 2

Figure B.2(a) Port 4 Block Diagram (Pin P43)

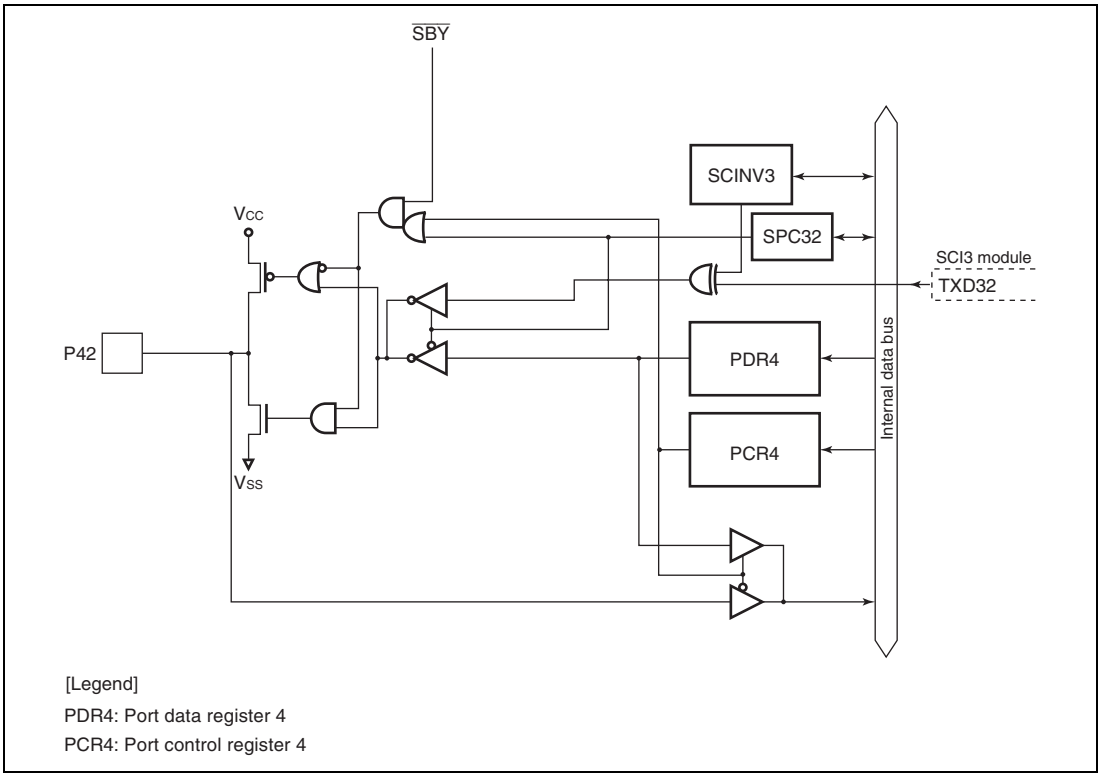


Figure B.2(b) Port 4 Block Diagram (Pin P42)

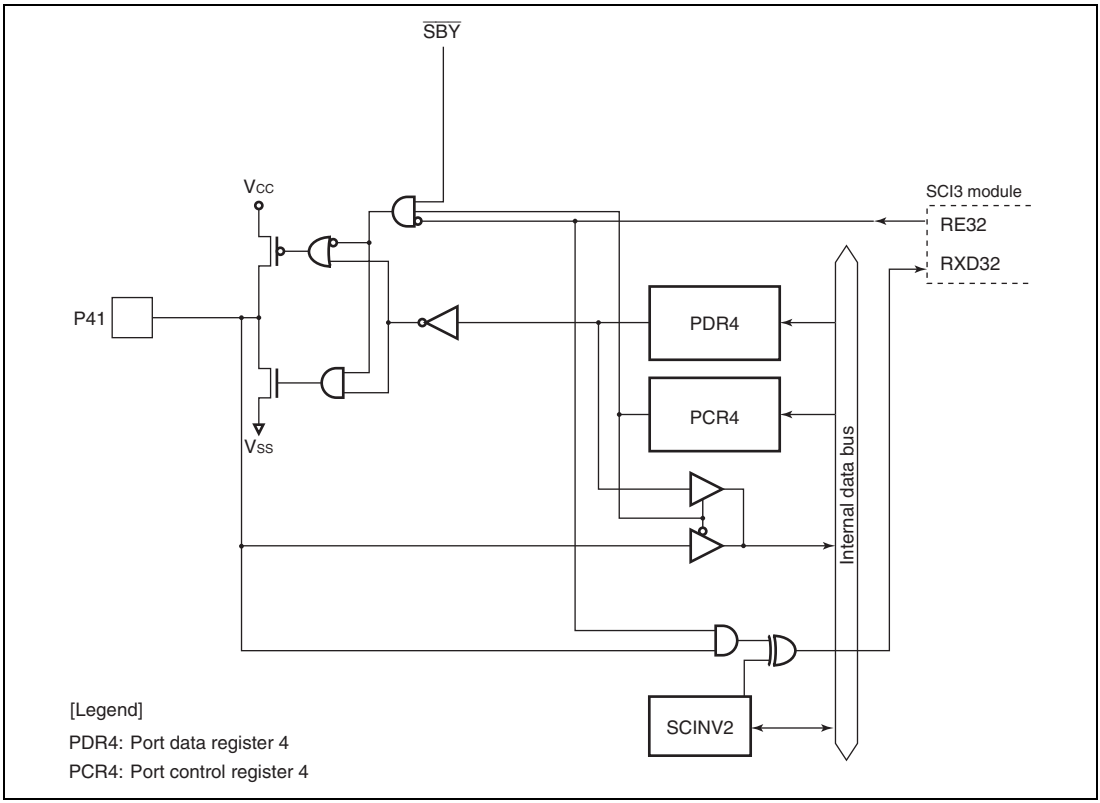


Figure B.2(c) Port 4 Block Diagram (Pin P41)

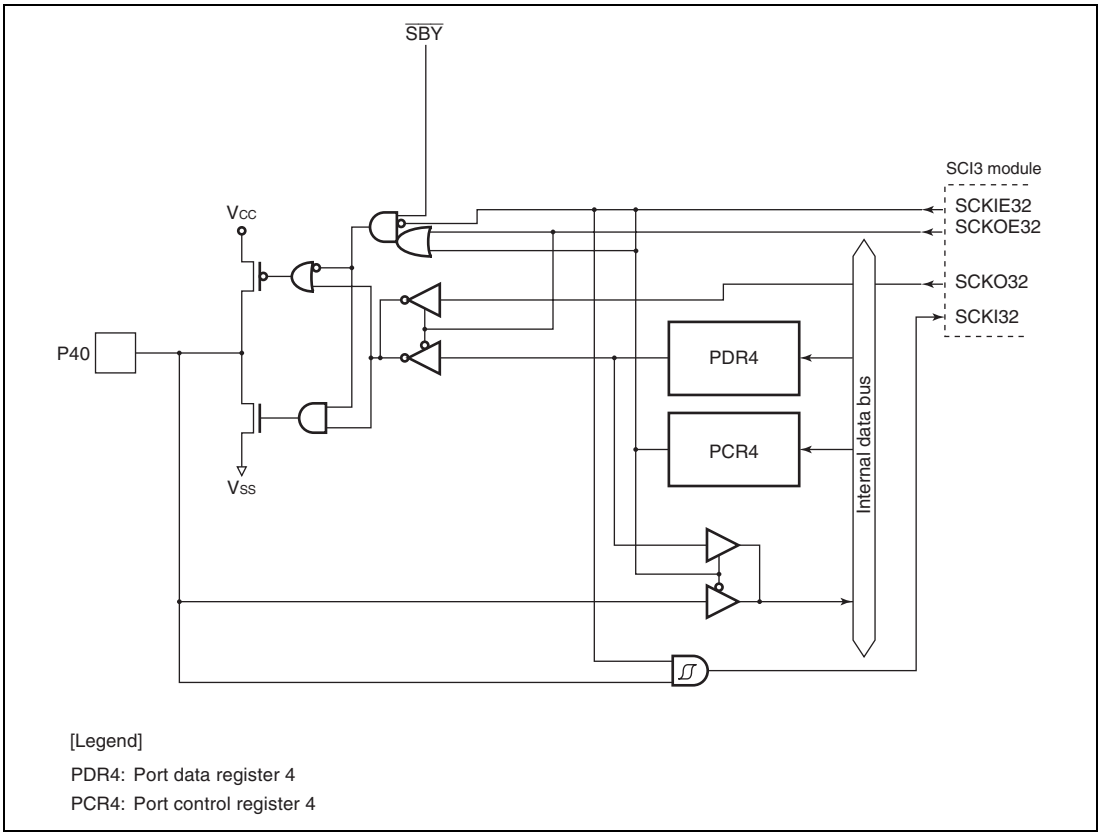


Figure B.2(d) Port 4 Block Diagram (Pin P40)

B.3 Port 5 Block Diagram

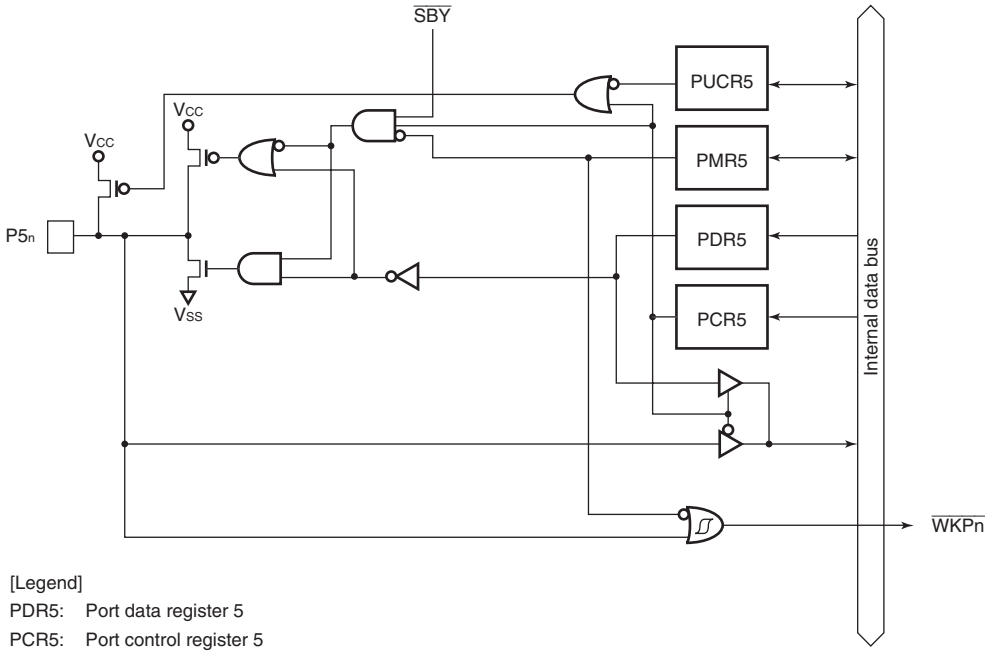
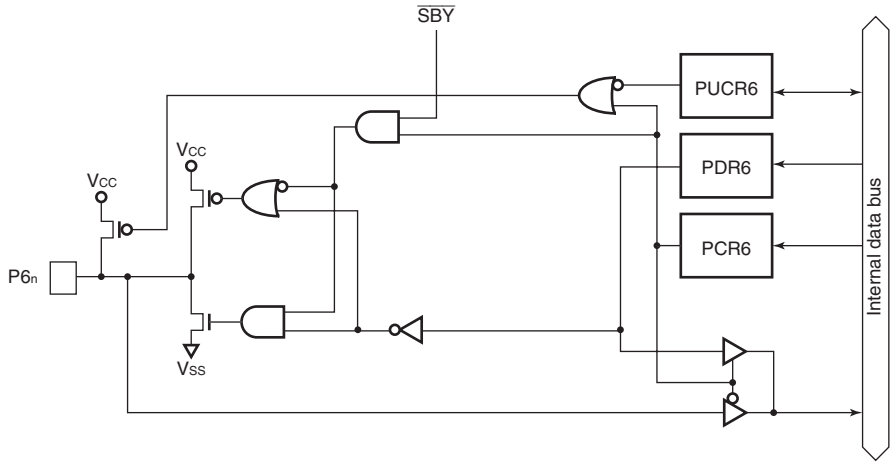


Figure B.3 Port 5 Block Diagram

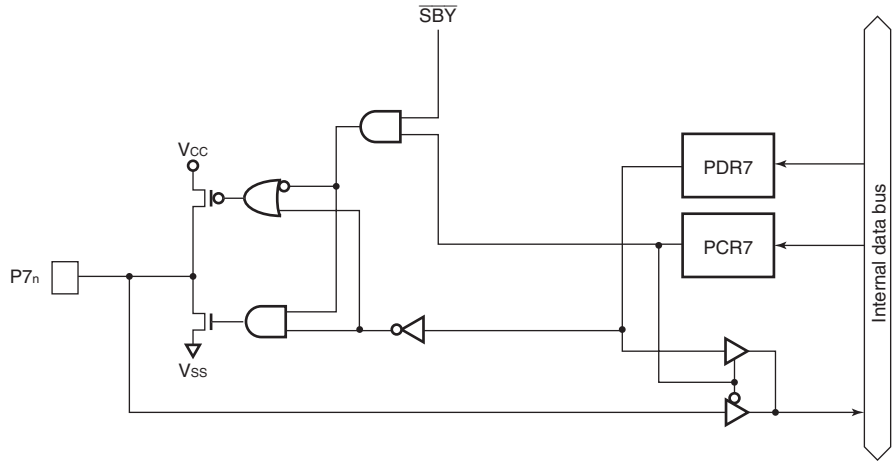
B.4 Port 6 Block Diagram



[Legend]
PDR6: Port data register 6
PCR6: Port control register 6
PUCR6: Port pull-up control register 6
n = 7 to 0

Figure B.4 Port 6 Block Diagram

B.5 Port 7 Block Diagram



[Legend]
PDR7: Port data register 7
PCR7: Port control register 7
n = 7 to 0

Figure B.5 Port 7 Block Diagram

B.6 Port 8 Block Diagram

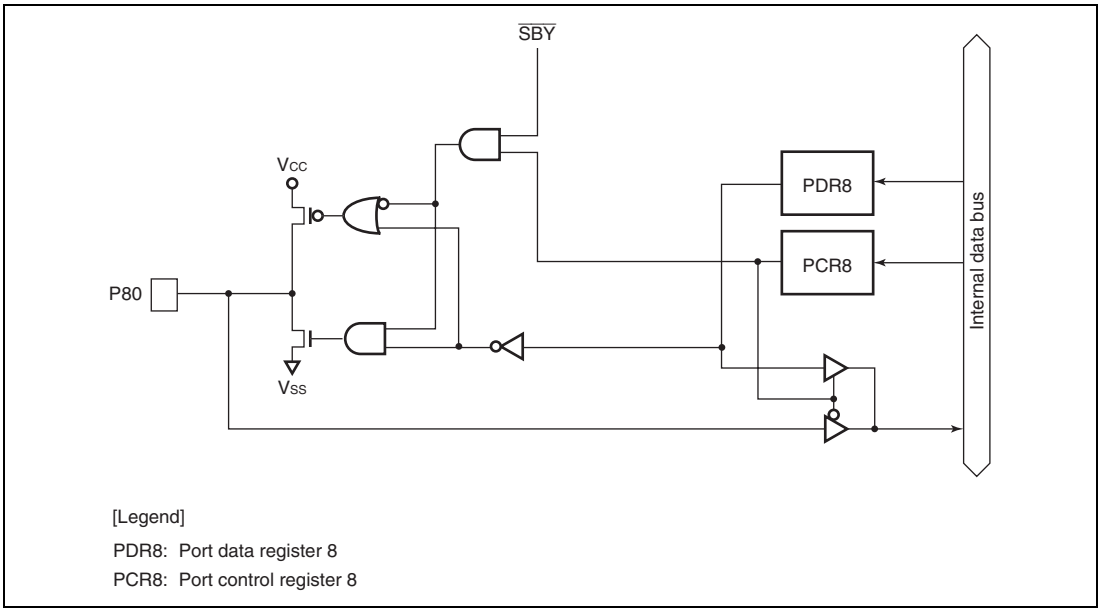


Figure B.6 Port 8 Block Diagram (Pin P80)

B.7 Port 9 Block Diagrams

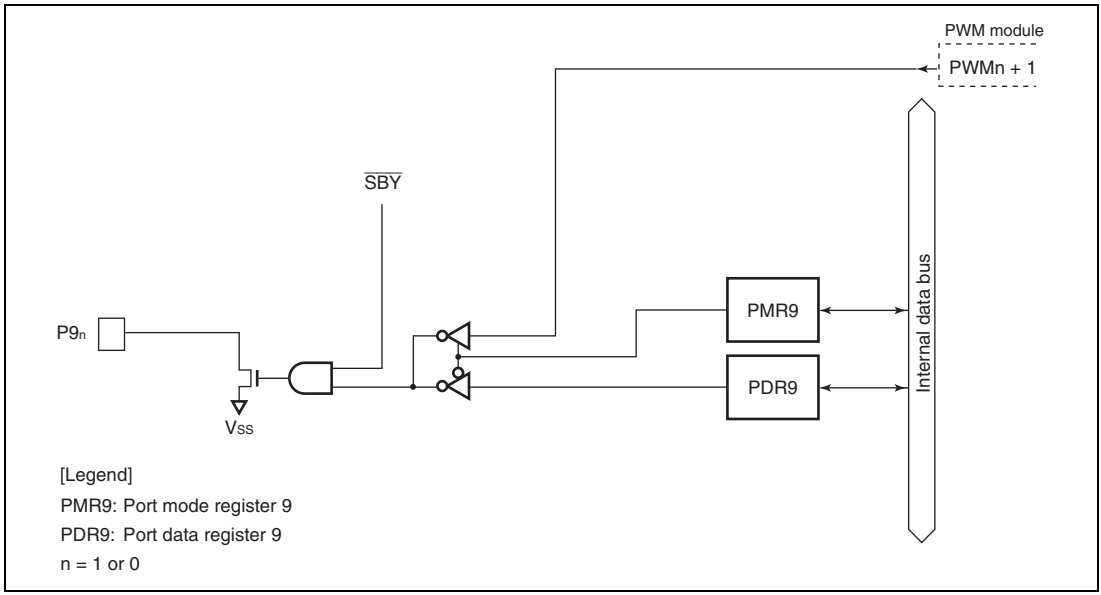


Figure B.7(a) Port 9 Block Diagram (Pins P91 and P90)

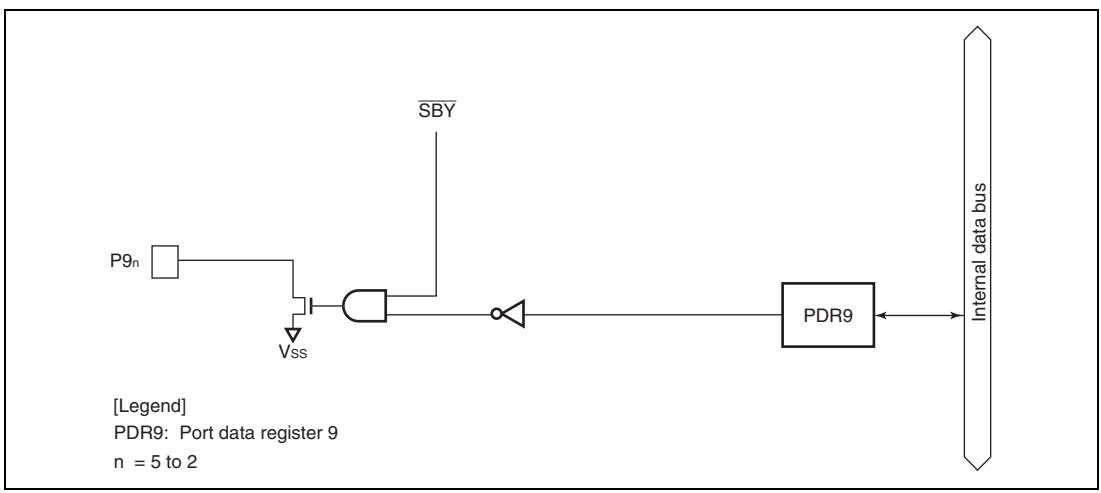
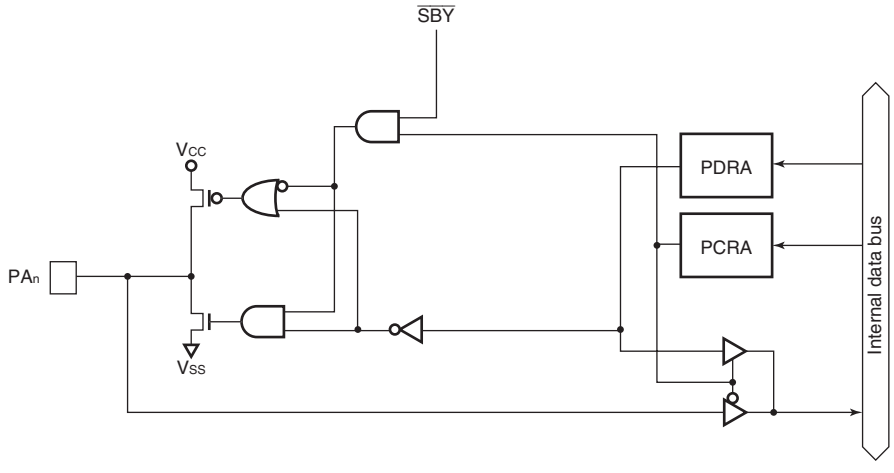


Figure B.7(b) Port 9 Block Diagram (Pins P95 to P92)

B.8 Port A Block Diagram



[Legend]
PDRA: Port data register A
PCRA: Port control register A
n = 3 to 0

Figure B.8 Port A Block Diagram

B.9 Port B Block Diagrams

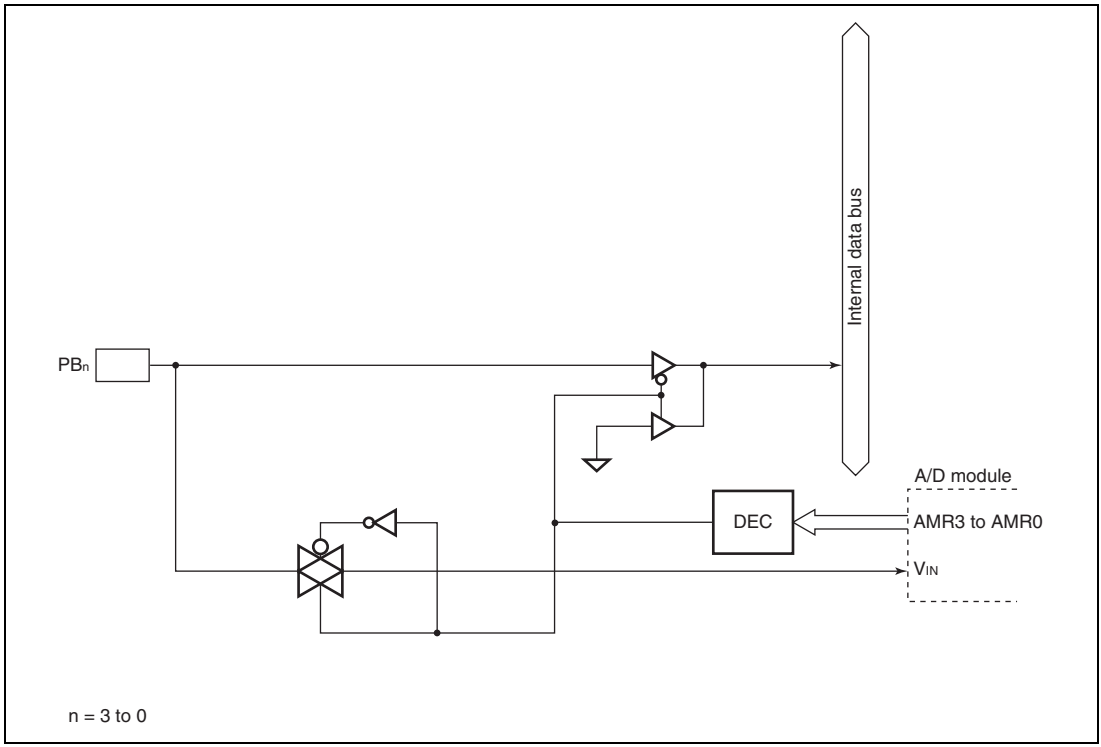


Figure B.9 Port B Block Diagram

C. Port States in Each Operating State

Table C.1 Port States

Port	Reset	Sleep	Subsleep	Standby	Watch	Subactive	Active
P37 to P31	High impedance	Retained	Retained	High impedance*	Retained	Functioning	Functioning
P43 to P40	High impedance	Retained	Retained	High impedance	Retained	Functioning	Functioning
P57 to P50	High impedance	Retained	Retained	High impedance*	Retained	Functioning	Functioning
P67 to P60	High impedance	Retained	Retained	High impedance*	Retained	Functioning	Functioning
P77 to P70	High impedance	Retained	Retained	High impedance	Retained	Functioning	Functioning
P80	High impedance	Retained	Retained	High impedance	Retained	Functioning	Functioning
P95 to P90	High impedance	Retained	Retained	High impedance	Retained	Functioning	Functioning
PA3 to PA0	High impedance	Retained	Retained	High impedance	Retained	Functioning	Functioning
PB3 to PB0	High impedance	High impedance	High impedance	High impedance	High impedance	High impedance	High impedance

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

D. Product Code Lineup

Table D.1 Product Code Lineup of H8/38704 Group

Product Type			Product Code	Model Marking	Package (Package Code)
H8/38704	Flash memory version	Regular product (2.7 V)	HD64F38704H10	64F38704H10	64-pin QFP (FP-64A)
			HD64F38704FP10	F38704FP10	64-pin LQFP (FP-64E)
			HD64F38704FT10	F38704FT10	64-pin QFN (TNP-64B)
		Regular product (2.2 V)	HD64F38704H4	64F38704H4	64-pin QFP (FP-64A)
			HD64F38704FP4	F38704FP4	64-pin LQFP (FP-64E)
			HD64F38704 FT4	F38704FT4	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications (2.7 V)	HD64F38704H10W	64F38704H10	64-pin QFP (FP-64A)
			HD64F38704FP10W	F38704FP10	64-pin LQFP (FP-64E)
			HD64F38704FT10W	38704FT10	64-pin QFN (TNP-64B)
	Mask ROM version	Regular product	HD64338704H	HD64338704H	64-pin QFP (FP-64A)
			HD64338704FP	38704 (***) FP	64-pin LQFP (FP-64E)
			HD64338704FT	38704 (***) FT	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications	HD64338704HW	HD64338704H	64-pin QFP (FP-64A)
			HD64338704FPW	38704 (***) FP	64-pin LQFP (FP-64E)
			HD64338704FTW	38704 (***) FT	64-pin QFN (TNP-64B)
H8/38703	Mask ROM version	Regular product	HD64338703H	HD64338703H	64-pin QFP (FP-64A)
			HD64338703FP	38703 (***) FP	64-pin LQFP (FP-64E)
			HD64338703FT	38703 (***) FT	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications	HD64338703HW	HD64338703H	64-pin QFP (FP-64A)
			HD64338703FPW	38703 (***) FP	64-pin LQFP (FP-64E)
			HD64338703FTW	38703 (***) FT	64-pin QFN (TNP-64B)

Product Type	Product Code	Model Marking	Package (Package Code)	Product Type	Product Code
H8/38702	Flash memory version	Regular product (2.7 V)	HD64F38702H10	64F38702H10	64-pin QFP (FP-64A)
			HD64F38702FP10	F38702FP10	64-pin LQFP (FP-64E)
			HD64F38702FT10	F38702FT10	64-pin QFN (TNP-64B)
		Regular product (2.2 V)	HD64F38702H4	64F38702H4	64-pin QFP (FP-64A)
			HD64F38702FP4	F38702FP4	64-pin LQFP (FP-64E)
			HD64F38702FT4	F38702FT4	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications (2.7 V)	HD64F38702H10W	64F38702H10	64-pin QFP (FP-64A)
			HD64F38702FP10W	F38702FP10	64-pin LQFP (FP-64E)
			HD64F38702FT10W	F38702FT10	64-pin QFN (TNP-64B)
	Mask ROM version	Regular product	HD64338702H	HD64338702H	64-pin QFP (FP-64A)
			HD64338702FP	38702 (***) FP	64-pin LQFP (FP-64E)
			HD64338702FT	38702 (***) FT	64-pin LQFN (TNP-64B)
		Product with wide-range temperature specifications	HD64338702HW	HD64338702H	64-pin QFP (FP-64A)
			HD64338702FPW	38702 (***) FP	64-pin LQFP (FP-64E)
			HD64338702FTW	38702 (***) FT	64-pin QFN (TNP-64B)

[Legend]
(***): ROM code

Table D.2 Product Code Lineup of H8/38702S Group

Product Type			Product Code	Model Marking	Package (Package Code)
H8/38702S	Mask ROM version	Regular product	HD64338702SH	38702 (***) H	64-pin QFP (FP-64A)
			HD64338702SFZ	38702 (***)	64-pin LQFP (FP-64K)
			HD64338702SFT	38702 (***) FT	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications	HD64338702SHW	38702 (***) H	64-pin QFP (FP-64A)
			HD64338702SFZW	38702 (***)	64-pin LQFP (FP-64K)
			HD64338702SFTW	38702 (***) FT	64-pin QFN (TNP-64B)
H8/38701S	Mask ROM version	Regular product	HD64338701SH	38701 (***) H	64-pin QFP (FP-64A)
			HD64338701SFZ	38701 (***)	64-pin LQFP (FP-64K)
			HD64338701SFT	38701 (***) FT	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications	HD64338701SHW	38701 (***) H	64-pin QFP (FP-64A)
			HD64338701SFZW	38701 (***)	64-pin LQFP (FP-64K)
			HD64338701SFTW	38701 (***) FT	64-pin QFN (TNP-64B)
H8/38700S	Mask ROM version	Regular product	HD64338700SH	38700 (***) H	64-pin QFP (FP-64A)
			HD64338700SFZ	38700 (***)	64-pin LQFP (FP-64K)
			HD64338700SFT	38700 (***) FT	64-pin QFN (TNP-64B)
		Product with wide-range temperature specifications	HD64338700SHW	38700 (***) H	64-pin QFP (FP-64A)
			HD64338700SFZW	38700 (***)	64-pin LQFP (FP-64K)
			HD64338700SFTW	38700 (***) FT	64-pin QFN (TNP-64B)

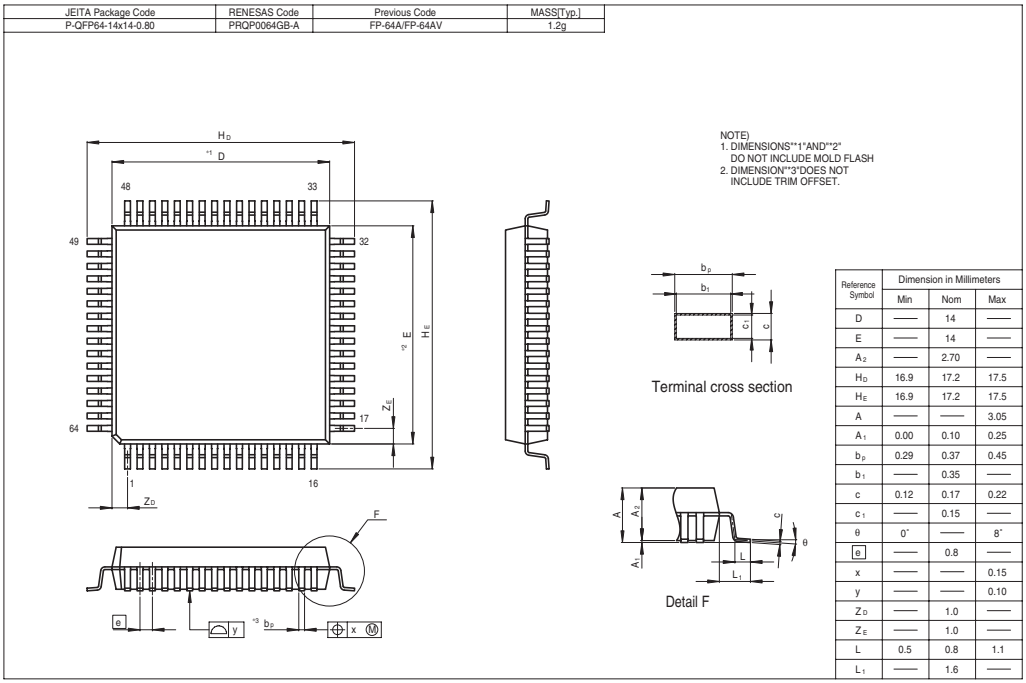
[Legend]

(***): ROM code

E. Package Dimensions

The package dimensions are shown in figure E.1 (FP-64A), figure E.2 (FP-64E), figure E.3 (FP-64K), and figure E.4 (TNP-64B).

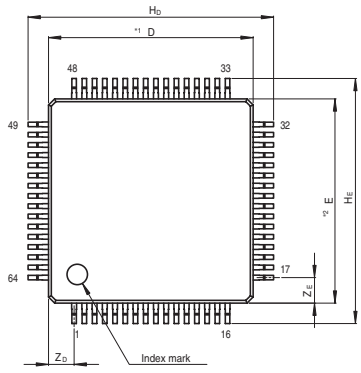
JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-QFP64-14x14-0.80	PRQP0064GB-A	FP-64A/FP-64AV	1.2g

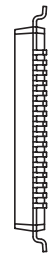


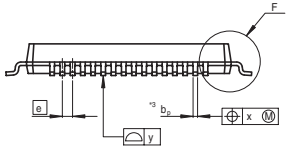
Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	—	14	—
E	—	14	—
A_2	—	2.70	—
H_D	16.9	17.2	17.5
H_E	16.9	17.2	17.5
A	—	—	3.05
A_1	0.00	0.10	0.25
b_p	0.29	0.37	0.45
b_1	—	0.35	—
c	0.12	0.17	0.22
c_1	—	0.15	—
θ	0°	—	8°
e	—	0.8	—
x	—	—	0.15
y	—	—	0.10
Z_D	—	1.0	—
Z_E	—	1.0	—
L	0.5	0.8	1.1
L_1	—	1.6	—

Figure E.1 Package Dimensions (FP-64A)

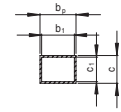
JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LQFP64-10x10-0.50	PLQP0064KC-A	FP-64E/FP-64EV	0.4g



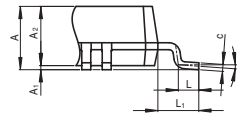




NOTE)
1. DIMENSIONS"*1"AND"*2" DO NOT INCLUDE MOLD FLASH
2. DIMENSION"*3"DOES NOT INCLUDE TRIM OFFSET.



Terminal cross section

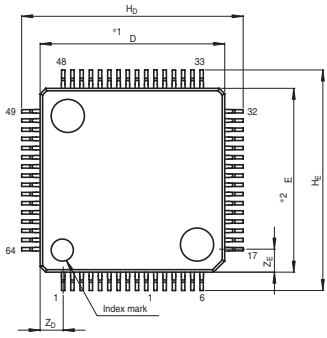


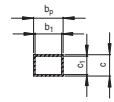
Detail F

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	——	10	——
E	——	10	——
A_2	——	1.45	——
H_D	11.8	12.0	12.2
H_E	11.8	12.0	12.2
A	——	——	1.70
A_1	0.00	0.10	0.20
b_p	0.17	0.22	0.27
b_1	——	0.20	——
c	0.12	0.17	0.22
c_1	——	0.15	——
θ	0°	——	8°
e	——	0.5	——
x	——	——	0.08
y	——	——	0.10
Z_D	——	1.25	——
Z_E	——	1.25	——
L	0.3	0.5	0.7
L_1	——	1.0	——

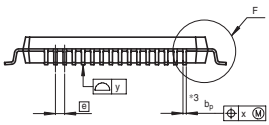
Figure E.2 Package Dimensions (FP-64E)

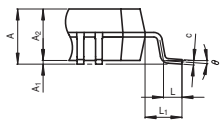
JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LQFP64-10x10-0.50	PLQP0064KB-A	64P6Q-A / FP-64K / FP-64KV	0.3g





Terminal cross section





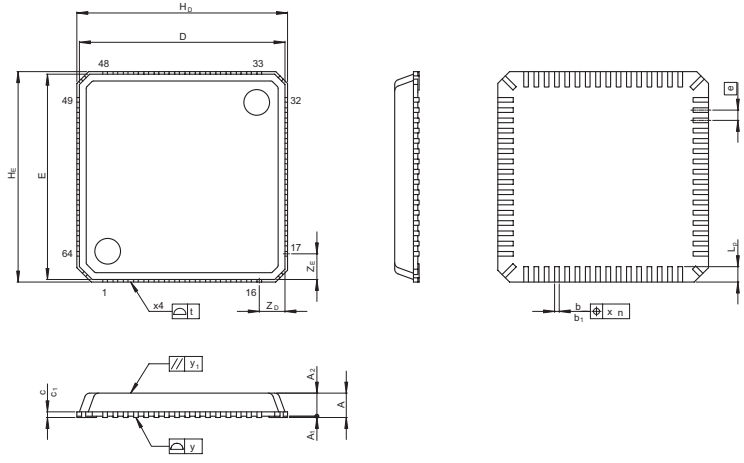
Detail F

NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	9.9	10.0	10.1
E	9.9	10.0	10.1
A_2	——	1.4	——
H_D	11.8	12.0	12.2
H_E	11.8	12.0	12.2
A	——	——	1.7
A_1	0.05	0.1	0.15
b_p	0.15	0.20	0.25
b_1	——	0.18	——
c	0.09	0.145	0.20
c_1	——	0.125	——
θ	0°	——	8°
e	——	0.5	——
x	——	——	0.08
y	——	——	0.08
Z_D	——	1.25	——
Z_E	——	1.25	——
L	0.35	0.5	0.65
L_1	——	1.0	——

Figure E.3 Package Dimensions (FP-64K)

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-VQFN64-8x8-0.40	PVQN0064LB-A	TNP-64B/TNP-64BV	0.12g



Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	—	8.0	—
E	—	8.0	—
A_2	—	0.89	—
A	—	—	0.95
A_1	0.005	0.02	0.04
b	0.13	0.18	0.23
b_1	—	0.16	—
e	—	0.4	—
L_p	0.50	0.60	0.70
x	—	—	0.05
y	—	—	0.05
y_1	—	—	0.2
t	—	—	0.2
H_D	—	8.2	—
H_E	—	8.2	—
Z_D	—	1.0	—
Z_E	—	1.0	—
c	0.17	0.22	0.25
c_1	—	0.20	—

Figure E.4 Package Dimensions (TNP-64B)

Index

M

O

P

R

S

T

V

W

Renesas 16-Bit Single-Chip Microcomputer
Hardware Manual
H8/38704 Group, H8/38702S Group

Publication Date:	Rev.1.00, Dec. 13, 2007
Published by:	Sales Strategic Planning Div. Renesas Technology Corp.
Edited by:	Customer Support Department Global Strategic Communication Div. Renesas Solutions Corp.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7858/7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2377-3473

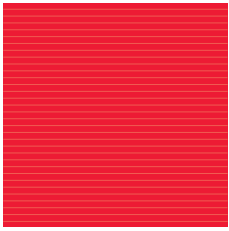
Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 3518-3399

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

H8/38704 Group, H8/38702S Group
Hardware Manual



Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan