To all our customers

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

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

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

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

Cautions

Keep safety first in your circuit designs!

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

Notes regarding these materials

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

Hitachi SuperH™ RISC engine

SH7018

Hardware Manual

ADE-602-186
Rev. 1.0
3/4/03
Hitachi, Ltd.

Cautions

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

Preface

The SH7018 is a single-chip RISC (reduced instruction set computer) microcomputer with a CPU based on Hitachi-original RISC-type SuperH™* architecture as its core. It also provides peripheral functions essential to system composition.

The CPU incorporated into the chip supports the RISC instruction set. Basic instructions execute in one cycle (one system clock cycle), so instruction execution times are extremely fast. The chip has an internal 32-bit architecture for efficient data processing. The CPU is capable of supporting applications employing real-time control. Such applications were not practical using previous microcomputers due to the extremely fast processing speeds required. It makes possible the development of systems providing high performance and excellent functionality at low cost.

The chip incorporates several peripheral functions essential to system composition, such as ROM, RAM, timers, serial communication interface (SCI), A/D converter, interrupt controller (INTC), and I/O ports. It also supports external memory access, which allows for efficient connections to external memory and LSI devices. These features help to reduce system costs substantially.

The SH7018 employs on-chip flash memory with F-ZTAT™* (flexible zero turnaround time). In addition to allowing programming of the chip by writing to it using a program writer for LSI devices, software can be written to the flash memory and erased as necessary.

This Hardware Manual describes the hardware features of the SH7018. For detailed information on the supported instructions, please refer to the Programming Manual.

Note: * SuperH™ and F-ZTAT™ are trademarks of Hitachi, Ltd.

Related Manuals

SH-1/SH-2/SH-DSP Programming Manual.

For information on development environment systems, please contact a Hitachi sales office.

Contents

Section 1 SH7018 Overview

1.1 SH7018 Features

The SH7018 is a CMOS single-chip microcomputer with a CPU based on Hitachi-original RISC-type SuperH™* architecture as its core. It also provides peripheral functions essential to system configuration.

The CPU incorporated into the chip supports the RISC (reduced instruction set computer) instruction set. Basic instructions execute in one cycle, so instruction execution times are extremely fast. The chip has an internal 32-bit architecture for efficient data processing. The CPU is capable of supporting applications employing real-time control. Such applications were not practical using previous microcomputers due to the extremely fast processing speeds required. It makes possible the development of systems providing high performance and excellent functionality at low cost.

The chip incorporates several peripheral functions essential to system configuration, such as ROM, RAM, timers, serial communication interface (SCI), A/D converter, interrupt controller (INTC), and I/O ports. It also supports external memory access, which allows for efficient connections to external memory and LSI devices. These features help to reduce system costs substantially.

The SH7018 employs on-chip flash memory with F-ZTAT™* (flexible zero turnaround time). In addition to allowing programming of the chip by writing to it using a program writer for LSI devices, software can be written to the flash memory and erased as necessary. This allows the firmware to be overwritten at the user site while the chip is mounted on the circuit board.

Note: * SuperH™ and F-ZTAT™ are trademarks of Hitachi, Ltd.

Table 1.1 Features

Item	Specification
CPU	• Original Hitachi architecture • Internal 32-bit configuration • General register machine — General registers: 32-bit × 16 — Control registers: 32-bit × 3 — System registers: 32-bit × 4 • RISC (reduced instruction set computer) instruction set — Instruction length: 16-bit fixed for efficient coding — Load-store architecture (basic operations executed between registers) — Extended branching instructions to minimize pipeline disturbance when branching — Instruction set based on C language • Instruction execution time of one cycle per instruction (50 ns per instruction when operating at 20 MHz) • Maximum address space of 4 GB supported by architecture • On-chip multiplier The on-chip multiplier handles 32 × 32 → 64 multiplication operations in two to four cycles and 32 × 32 + 64 → 64 multiplication/accumulation operations in two to four cycles. • Pipeline 5-stage pipeline
Interrupt controller (INTC)	• Seven external interrupt pins (NMI, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$) • 16-level priority setting supported
Bus state controller (BSC)	• Bus access to external memory and external devices supported — 8-bit fixed external data bus • Address space divided into four areas (SRAM space × 4 areas) Wait cycles may be specified (0 to 3 cycles) separately for each area. • Chip select signals corresponding to memory areas are output. • Wait cycles may be inserted using external $\overline{\text{WAIT}}$ signal.

Table 1.1 Features (cont)

Item	Specification
Multifunction timer pulse unit (MTU) × 3 channels	• 16-bit free running counter × 3 channels • Eight compare match registers • Interrupt requests are generated by compare match and overflow operations.
Compare match timer (CMT) × 2 channels	• 16-bit free running counter × 2 channels • Compare registers: 1 per channel • Interrupt requests are generated by compare match operations.
Watchdog timer (WDT)	• Can be switched between watchdog timer and interval timer functions. • Internal reset or interrupt generated by counter overflow.
5 V I/O pins	• By specifying the power supply for the input/output circuitry, PV_{CC}, the input/output voltage level for the following pins can be set to either 3.3 V or 5 V: $\overline{RES}$, NMI, $\overline{IRQ0}$, $\overline{WAIT}$, D0 to D7, SCK, TxD, RxD, TIOC0A, TIOC0C (total 17 pins).
8-bit timer (TIM2)	• 8-bit interval timer function • Interrupt generated by compare match operations.
Serial communication interface (SCI)	• Asynchronous or clock-synchronous mode is selectable (full duplex) • On-chip dedicated baud rate generator • Multi-processor communication function
I/O ports	• 62 inputs and outputs • 8 inputs
A/D converter	• 10 bits × 8 channels • Built-in sample and hold function
On-chip memory	• RAM: 4 kB • ROM: 160 kB (F-ZTAT)

Table 1.1 Features (cont)

Item	Specification
Operating modes	• Processing modes — Program execution mode — Exception processing mode • Operating modes — Extended ROM enabled mode — Boot mode — User program mode — Program mode • Low-power-consumption modes — Sleep mode — Standby mode
Clock pulse generator (CPG)	On-chip clock pulse generator (1 : 1 oscillation using duty correction circuit)
Product lineup	**Product Name** / **Voltage** / **Operating Frequency** / **Product Code** / **Package** SH7018 / 3.3 V / 20 MHz / HD64F7018X20 / TFP-100B

1.2 Block Diagram

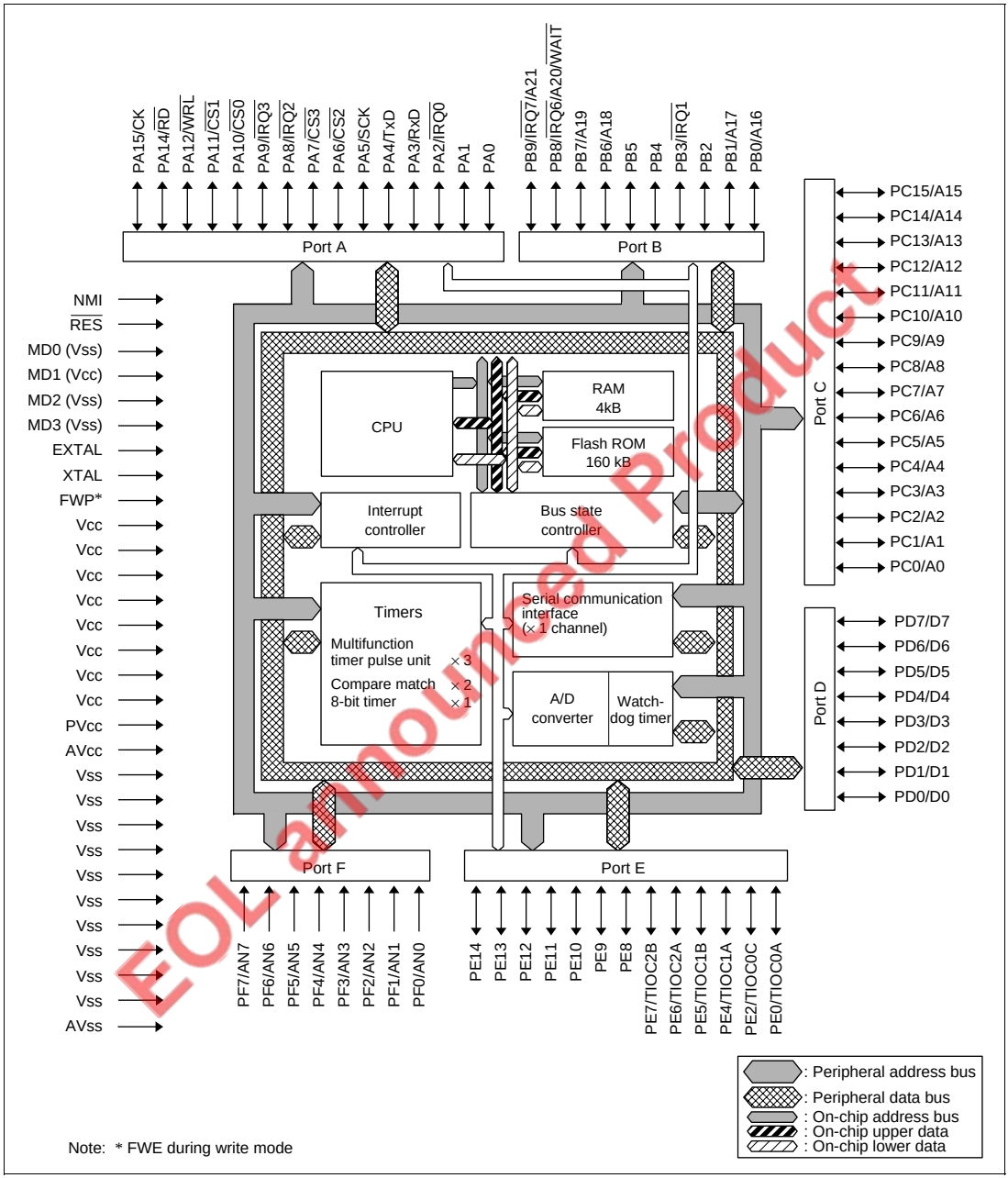
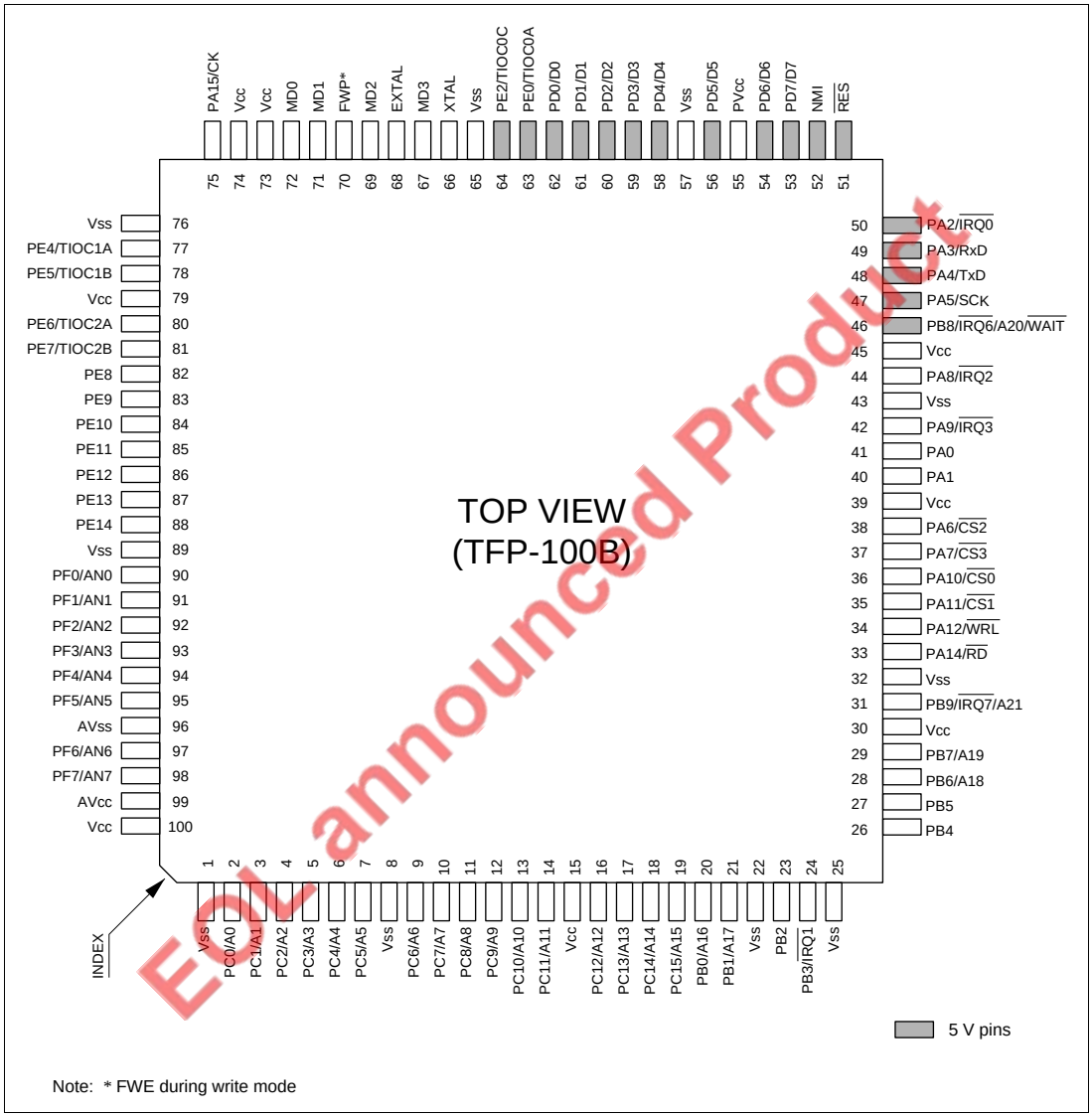


Figure 1.1 Block Diagram of SH7018 (TFP-100B Pins)

1.3 Pin Description

1.3.1 Pin Layout



Note: * FWE during write mode

Figure 1.2 SH7018 Pin Layout (TFP-100B: Top View)

1.3.2 Pin Functions

The pin functions are listed in table 1.2.

Table 1.2 Pin Functions

Type	Abbreviation	I/O	Name	Function
Power supply	V_{CC}	Input	Power supply	Connect the power supply for the entire system to the V_{CC} pin. The chip will not function if this pin is left open.
	V_{SS}	Input	Ground	Connect this pin to a ground. Connect the ground for the entire system to the V_{SS} pin. The chip will not function if this pin is left open.
	PV_{cc}	Input	Power supply for input/output circuit	This pin is the power supply for an input/output circuit.
Clock	EXTAL	Input	External clock	Connect this pin to a crystal oscillator. Alternately, the EXTAL pin may be used for external clock input.
	XTAL	Input	Crystal	Connect this pin to a crystal oscillator.
	CK	Output	System clock	Supplies the system clock signal to peripheral devices.
System control	$\overline{RES}$	Input	Power-on reset	Applying a low-level signal to this pin triggers a power-on reset.
Operating mode control	MD0 to MD3	Input	Mode setting	This pin determines the operating mode. Do not change the input level while the system is operating.
	FWP	Input	Flash memory write prevent	This pin is used to prevent programming or erasing of the flash memory.
Interrupts	NMI	Input	Non-maskable interrupt	This is a non-maskable interrupt request pin. The user can select whether requests are accepted at the rising or the falling edge.
	$\overline{IRQ0}$ to $\overline{IRQ3}$, $\overline{IRQ6}$, $\overline{IRQ7}$	Input	Interrupt request 0 to 3, 6, and 7	These are maskable interrupt request pins. Level input and edge input selection are supported.

Table 1.2 Pin Functions (cont)

Type	Abbreviation	I/O	Name	Function
Address bus	A0 to A21	Output	Address bus	These pins are used for address output.
Data bus	D0 to D7	Input/ output	Data bus	This is an 8-bit bi-directional data bus.
Bus control	$\overline{CS0}$ to $\overline{CS3}$	Output	Chip select 0 to 3	These pins are used for chip select signals for external memory or devices.
	$\overline{RD}$	Output	Read	Indicates that data is being read from an external device.
	$\overline{WRL}$	Output	Write	Indicates that the lower eight external data bits (bits 7 to 0) are being written to.
	$\overline{WAIT}$	Input	Wait	This input is used to insert a wait cycle when accessing the external area.
Multifunction timer pulse unit (MTU)	TIOC0A TIOC0C	Input/ output	MTU input capture/output compare (channel 0)	Channel 0 input capture input/output compare output/PWM output pin.
	TIOC1A TIOC1B	Input/ output	MTU input capture/output compare (channel 1)	Channel 1 input capture input/output compare output/PWM output pin.
	TIOC2A TIOC2B	Input/ output	MTU input capture/output compare (channel 2)	Channel 2 input capture input/output compare output/PWM output pin.
Serial communication interface (SCI)	TxD	Output	Transmit data	Transmit data output pin.
	RxD	Input	Receive data	Receive data input pin.
	SCK	Input/ output	Serial clock	Clock input/output pin.

Table 1.2 Pin Functions (cont)

Type	Abbreviation	I/O	Name	Function
A/D converter	AV_{CC}	Input	Analog power supply	Connect to V_{CC} potential when using an analog power supply.
	AV_{SS}	Input	Analog ground	Connect to V_{ss} potential when using an analog power supply.
	AN0 to AN7	Input	Analog inputs	Analog signal input terminals.
I/O ports	PA15, 14, 12 to 0	Input/ output	General port	General I/O port pins. I/O can be specified one bit at a time.
	PB9 to 0	Input/ output	General port	General I/O port pins. I/O can be specified one bit at a time.
	PC15 to 0	Input/ output	General port	General I/O port pins. I/O can be specified one bit at a time.
	PD7 to 0	Input/ output	General port	General I/O port pins. I/O can be specified one bit at a time.
	PE14 to 4, 2, 0	Input/ output	General port	General I/O port pins. I/O can be specified one bit at a time.
	PF7 to 0	Input	General port	General I/O port pins.

Note: The following power-on/power-off order is recommended.

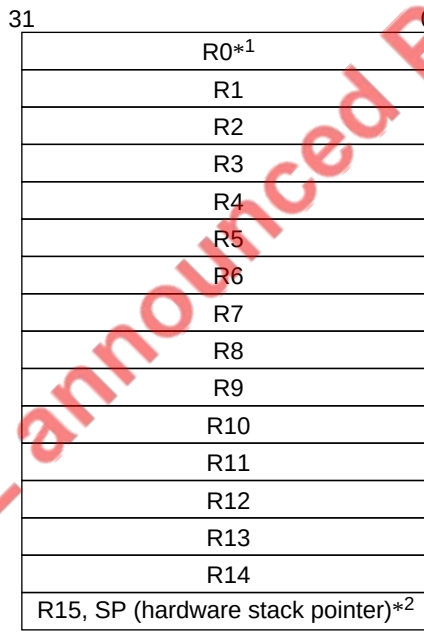
1. Powering on
   (1) Turn on the 5 V power (PV_{CC}) first, then the 3 V power (V_{CC}, $PLLV_{CC}$, AV_{CC}).
   (2) Pin states are undefined while only 5 V power is on, as reset input is invalid.
2. Powering off
   (1) Power off in the reverse order to powering on: turn off the 3 V power first, then the 5 V power.
   (2) Pin states are undefined while only 5 V power is being supplied.
3. Power-on/off interval
   To minimize the length of time during which pin states are undefined, the power-on/off interval should be kept as short as possible. Also, the system design should ensure that erroneous system operation will not result from pin states becoming undefined.

2.1 Register Configuration

The register set consists of sixteen 32-bit general registers, three 32-bit control registers and four 32-bit system registers.

2.1.1 General Registers (Rn)

The sixteen 32-bit general registers (Rn) are numbered R0 to R15. General registers are used for data processing and address calculation. R0 is also used as an index register. Several instructions have R0 fixed as their only usable register. R15 is used as the hardware stack pointer (SP). Saving and recovering the status register (SR) and program counter (PC) in exception processing is accomplished by referencing the stack using R15. Figure 2.1 shows the general registers.



Notes: 1. R0 functions as an index register in the indirect indexed register addressing mode and indirect indexed GBR addressing mode. In some instructions, R0 functions as a fixed source register or destination register.
2. R15 functions as a hardware stack pointer (SP) during exception processing.

Figure 2.1 General Registers

2.1.2 Control Registers

The 32-bit control registers consist of the 32-bit status register (SR), global base register (GBR), and vector base register (VBR). The status register indicates processing states. The global base register functions as a base address for the indirect GBR addressing mode to transfer data to the registers of on-chip peripheral modules. The vector base register functions as the base address of the exception processing vector area (including interrupts). Figure 2.2 shows a control register.

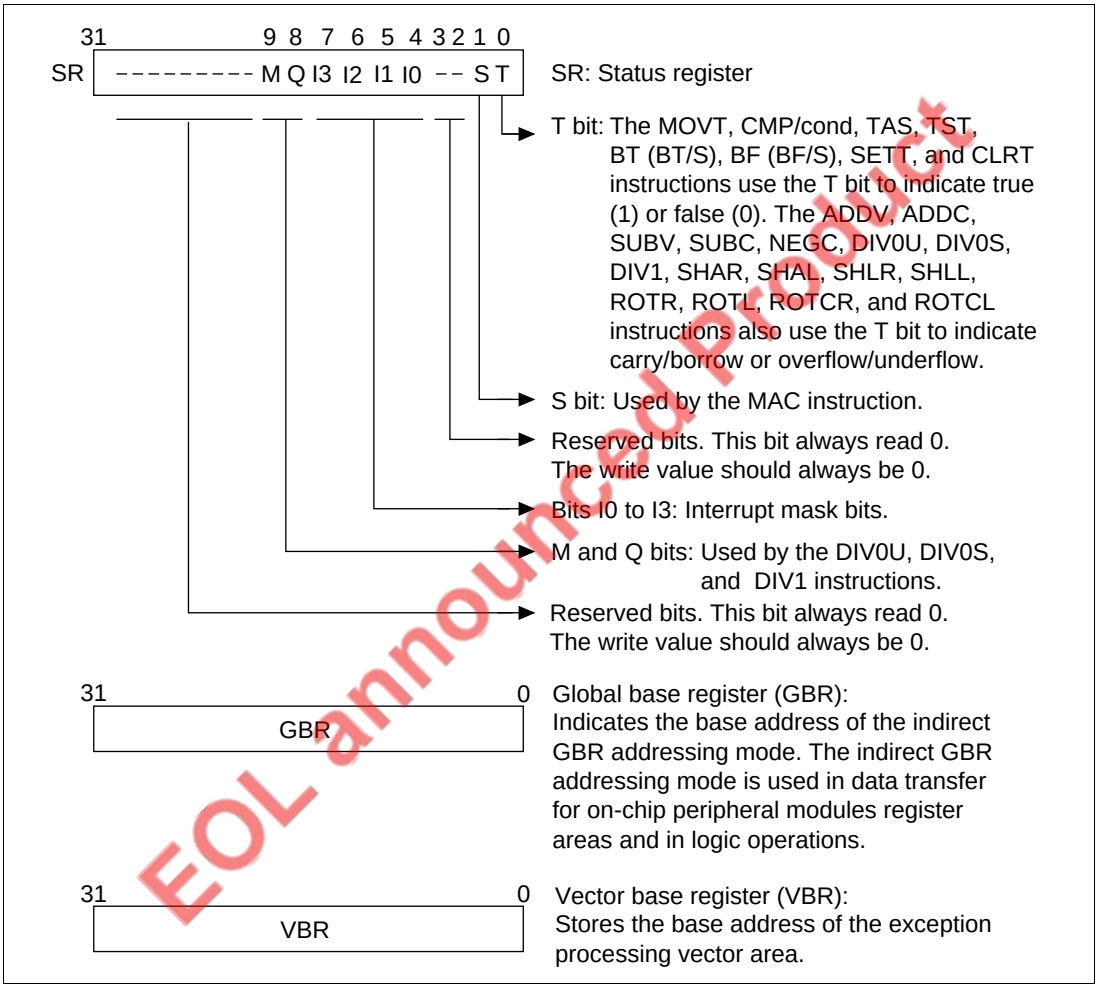


Figure 2.2 Control Registers

2.1.3 System Registers

System registers consist of four 32-bit registers: high and low multiply and accumulate registers (MACH and MACL), the procedure register (PR), and the program counter (PC). The multiply and accumulate registers store the results of multiply and accumulate operations. The procedure register stores the return address from the subroutine procedure. The program counter stores program addresses to control the flow of the processing. Figure 2.3 shows a system register.

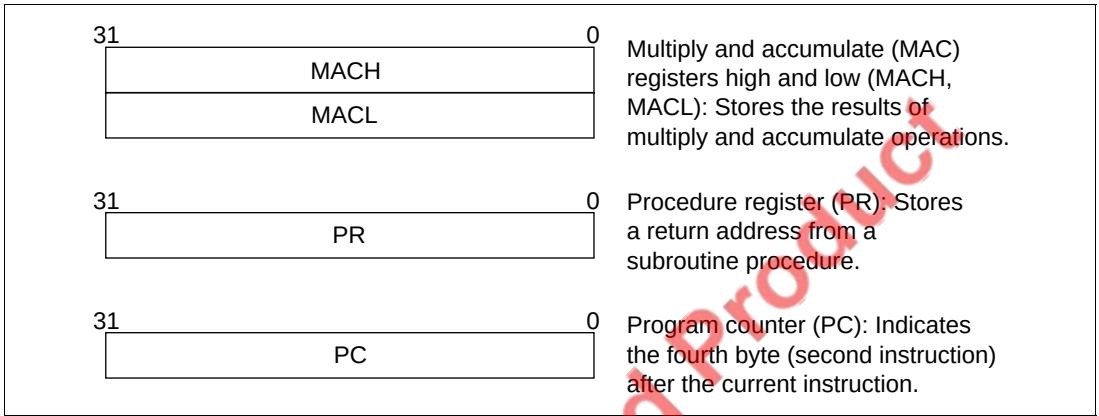


Figure 2.3 System Registers

2.1.4 Initial Values of Registers

Table 2.1 lists the values of the registers after reset.

Table 2.1 Initial Values of Registers

Classification	Register	Initial Value
General registers	R0 to R14	Undefined
	R15 (SP)	Value of the stack pointer in the vector address table
Control registers	SR	Bits I3 to I0 are 1111 (H'F), reserved bits are 0, and other bits are undefined
	GBR	Undefined
	VBR	H'00000000
System registers	MACH, MACL, PR	Undefined
	PC	Value of the program counter in the vector address table

2.2 Data Formats

2.2.1 Data Format in Registers

Register operands are always longwords (32 bits). When the memory operand is only a byte (8 bits) or a word (16 bits), it is sign-extended into a longword when loaded into a register (figure 2.4).

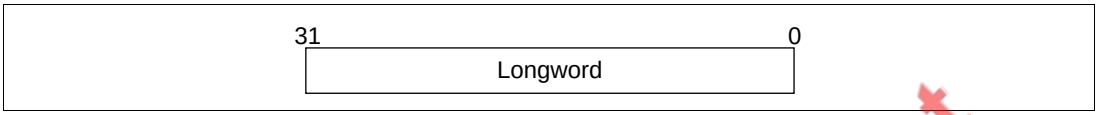


Figure 2.4 Longword Operand

2.2.2 Data Format in Memory

Memory data formats are classified into bytes, words, and longwords. Byte data can be accessed from any address, but an address error will occur if you try to access word data starting from an address other than 2n or longword data starting from an address other than 4n. In such cases, the data accessed cannot be guaranteed. The hardware stack area, referred to by the hardware stack pointer (SP, R15), uses only longword data starting from address 4n because this area holds the program counter and status register (figure 2.5).

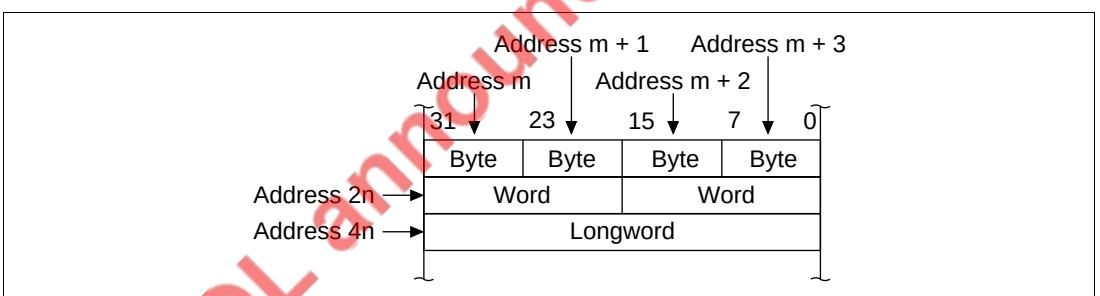


Figure 2.5 Byte, Word, and Longword Alignment

2.2.3 Immediate Data Format

Byte (8-bit) immediate data resides in an instruction code. Immediate data accessed by the MOV, ADD, and CMP/EQ instructions is sign-extended and handled in registers as longword data. Immediate data accessed by the TST, AND, OR, and XOR instructions is zero-extended and handled as longword data. Consequently, AND instructions with immediate data always clear the upper 24 bits of the destination register.

Word or longword immediate data is not located in the instruction code, but instead is stored in a memory table. An immediate data transfer instruction (MOV) accesses the memory table using the PC relative addressing mode with displacement.

2.3 Instruction Features

2.3.1 RISC-Type Instruction Set

All instructions are RISC type. This section details their functions.

16-Bit Fixed Length: All instructions are 16 bits long, increasing program code efficiency.

One Instruction per Cycle: The microprocessor can execute basic instructions in one cycle using the pipeline system. Instructions are executed in 35 ns at 28.7 MHz.

Data Length: Longword is the standard data length for all operations. Memory can be accessed in bytes, words, or longwords. Byte or word data accessed from memory is sign-extended and handled as longword data. Immediate data is sign-extended for arithmetic operations or zero-extended for logic operations. It also is handled as longword data (table 2.2).

Table 2.2 Sign Extension of Word Data

SH7018 CPU		Description	Example of Conventional CPU	
MOV.W	@(disp,PC),R1	Data is sign-extended to 32 bits, and R1 becomes H'00001234. It is next operated upon by an ADD instruction.	ADD.W	#H'1234,R0
ADD	R1,R0			
	.....			
.DATA.W	H'1234			

Note: @(disp, PC) accesses the immediate data.

Load-Store Architecture: Basic operations are executed between registers. For operations that involve memory access, data is loaded to the registers and executed (load-store architecture). Instructions such as AND that manipulate bits, however, are executed directly in memory.

Delayed Branch Instructions: Unconditional branch instructions are delayed. Executing the instruction that follows the branch instruction and then branching reduces pipeline disruption during branching (table 2.3). There are two types of conditional branch instructions: delayed branch instructions and ordinary branch instructions.

Table 2.3 Delayed Branch Instructions

SH7018 CPU		Description	Example of Conventional CPU	
BRA	TRGET	Executes an ADD before branching to TRGET	ADD.W	R1,R0
ADD	R1,R0		BRA	TRGET

Multiplication/Accumulation Operation: 16-bit × 16-bit → 32-bit multiplication operations are executed in one to two cycles. 16-bit × 16-bit + 64-bit → 64-bit multiplication/accumulation operations are executed in two to three cycles. 32-bit × 32-bit → 64-bit and 32-bit × 32-bit + 64-bit → 64-bit multiplication/accumulation operations are executed in two to four cycles.

T Bit: The T bit in the status register changes according to the result of the comparison, and in turn is the condition (true/false) that determines if the program will branch. The number of instructions that change the T bit is kept to a minimum to improve the processing speed (table 2.4).

Table 2.4 T Bit

SH7018 CPU		Description	Example of Conventional CPU	
CMP/GE	R1,R0	T bit is set when R0 ≥ R1. The program branches to TRGET0 when R0 ≥ R1 and to TRGET1 when R0 < R1.	CMP.W	R1,R0
BT	TRGET0		BGE	TRGET0
BF	TRGET1		BLT	TRGET1
ADD	#1,R0	T bit is not changed by ADD. T bit is set when R0 = 0. The program branches if R0 = 0.	SUB.W	#1,R0
CMP/EQ	#0,R0		BEQ	TRGET
BT	TRGET			

Immediate Data: Byte (8-bit) immediate data resides in instruction code. Word or longword immediate data is not input via instruction codes but is stored in a memory table. An immediate data transfer instruction (MOV) accesses the memory table using the PC relative addressing mode with displacement (table 2.5).

Table 2.5 Immediate Data Accessing

Classification	SH7018 CPU		Example of Conventional CPU	
8-bit immediate	MOV	#H'12,R0	MOV.B	#H'12,R0
16-bit immediate	MOV.W	@(disp,PC),R0	MOV.W	#H'1234,R0
		.....		
	.DATA.W	H'1234		
32-bit immediate	MOV.L	@(disp,PC),R0	MOV.L	#H'12345678,R0
		.....		
	.DATA.L	H'12345678		

Note: @(disp, PC) accesses the immediate data.

Absolute Address: When data is accessed by absolute address, the value already in the absolute address is placed in the memory table. Loading the immediate data when the instruction is executed transfers that value to the register and the data is accessed in the indirect register addressing mode (table 2.6).

Table 2.6 Absolute Address Accessing

Classification	SH7018 CPU		Example of Conventional CPU	
Absolute address	MOV.L	@(disp,PC),R1	MOV.B	@H'12345678,R0
	MOV.B	@R1,R0		
		.....		
	.DATA.L	H'12345678		

Note: @(disp,PC) accesses the immediate data.

16-Bit/32-Bit Displacement: When data is accessed by 16-bit or 32-bit displacement, the pre-existing displacement value is placed in the memory table. Loading the immediate data when the instruction is executed transfers that value to the register and the data is accessed in the indirect indexed register addressing mode (table 2.7).

Table 2.7 Displacement Accessing

Classification	SH7018 CPU		Example of Conventional CPU	
16-bit displacement	MOV.W	@(disp,PC),R0	MOV.W	@(H'1234,R1),R2
	MOV.W	@(R0,R1),R2		
		.....		
	.DATA.W	H'1234		

Note: @(disp,PC) accesses the immediate data.

2.3.2 Addressing Modes

Table 2.8 describes addressing modes and effective address calculation.

Table 2.8 Addressing Modes and Effective Addresses

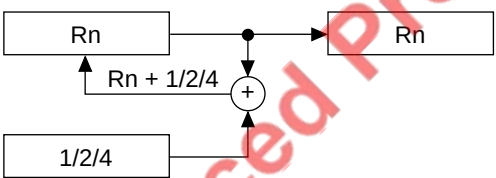
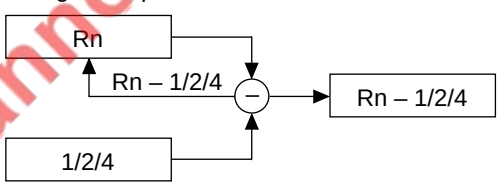
Addressing Mode	Instruction Format	Effective Addresses Calculation	Equation
Direct register addressing	Rn	The effective address is register Rn. (The operand is the contents of register Rn.)	—
Indirect register addressing	@Rn	The effective address is the content of register Rn. 	Rn
Post-increment indirect register addressing	@Rn+	The effective address is the content of register Rn. A constant is added to the content of Rn after the instruction is executed. 1 is added for a byte operation, 2 for a word operation, and 4 for a longword operation. 	Rn (After the instruction executes) Byte: Rn + 1 → Rn Word: Rn + 2 → Rn Longword: Rn + 4 → Rn
Pre-decrement indirect register addressing	@–Rn	The effective address is the value obtained by subtracting a constant from Rn. 1 is subtracted for a byte operation, 2 for a word operation, and 4 for a longword operation. 	Byte: Rn – 1 → Rn Word: Rn – 2 → Rn Longword: Rn – 4 → Rn (Instruction executed with Rn after calculation)

Table 2.8 Addressing Modes and Effective Addresses (cont)

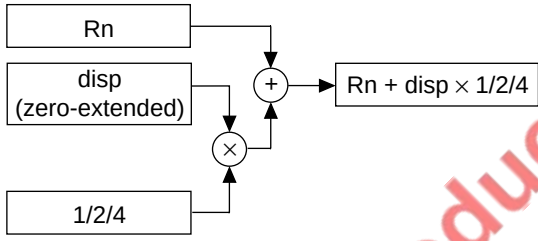
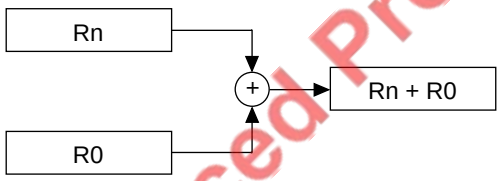
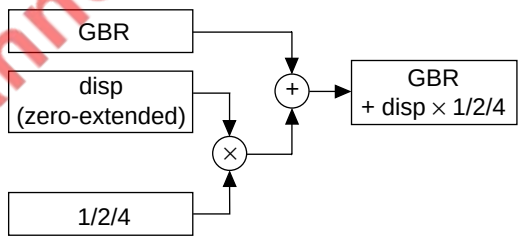
Addressing Mode	Instruction Format	Effective Addresses Calculation	Equation
Indirect register addressing with displacement	`@(disp:4, Rn)`	The effective address is Rn plus a 4-bit displacement (disp). The value of disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation. 	Byte: Rn + disp Word: Rn + disp × 2 Longword: Rn + disp × 4
Indirect indexed register addressing	`@(R0, Rn)`	The effective address is the Rn value plus R0. 	Rn + R0
Indirect GBR addressing with displacement	`@(disp:8, GBR)`	The effective address is the GBR value plus an 8-bit displacement (disp). The value of disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation. 	Byte: GBR + disp Word: GBR + disp × 2 Longword: GBR + disp × 4

Table 2.8 Addressing Modes and Effective Addresses (cont)

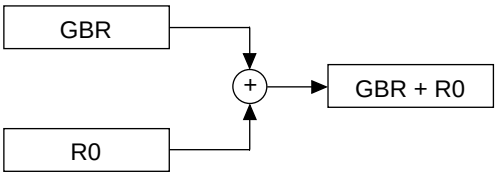
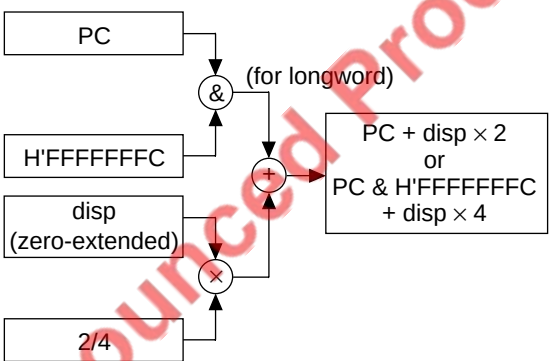
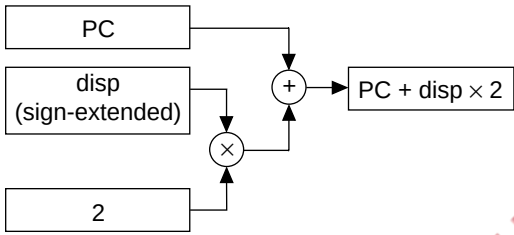
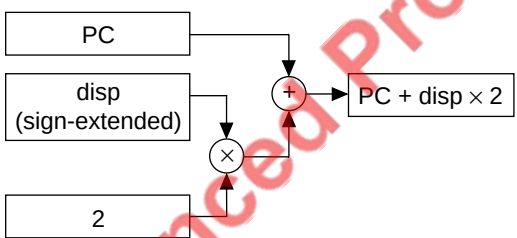
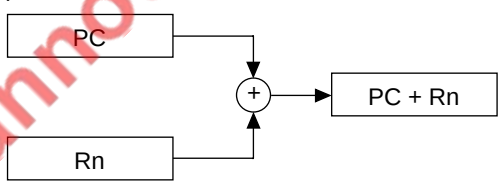
Addressing Mode	Instruction Format	Effective Addresses Calculation	Equation
Indirect indexed GBR addressing	`@(R0, GBR)`	The effective address is the GBR value plus the R0. 	GBR + R0
PC relative addressing with displacement	`@(disp:8, PC)`	The effective address is the PC value plus an 8-bit displacement (disp). The value of disp is zero-extended, and is doubled for a word operation, and quadrupled for a longword operation. For a longword operation, the lowest two bits of the PC value are masked. 	Word: PC + disp × 2 Longword: PC & H'FFFFFFFC + disp × 4

Table 2.8 Addressing Modes and Effective Addresses (cont)

Addressing Mode	Instruction Format	Effective Addresses Calculation	Equation
PC relative addressing	`disp:8`	The effective address is the PC value sign-extended with an 8-bit displacement (disp), doubled, and added to the PC value. 	PC + disp × 2
	`disp:12`	The effective address is the PC value sign-extended with a 12-bit displacement (disp), doubled, and added to the PC value. 	PC + disp × 2
	`Rn`	The effective address is the register PC value plus Rn. 	PC + Rn
Immediate addressing	`#imm:8`	The 8-bit immediate data (imm) for the TST, AND, OR, and XOR instructions are zero-extended.	—
	`#imm:8`	The 8-bit immediate data (imm) for the MOV, ADD, and CMP/EQ instructions are sign-extended.	—
	`#imm:8`	The 8-bit immediate data (imm) for the TRAPA instruction is zero-extended and is quadrupled.	—

2.3.3 Instruction Format

Table 2.9 lists the instruction formats for the source operand and the destination operand. The meaning of the operand depends on the instruction code. The symbols are used as follows:

- xxxx: Instruction code
- mmmm: Source register
- nnnn: Destination register
- iiii: Immediate data
- dddd: Displacement

Table 2.9 Instruction Formats

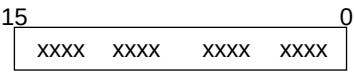
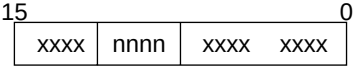
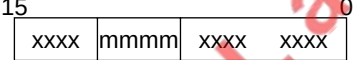
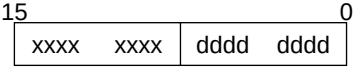
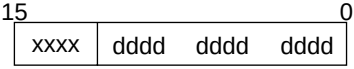
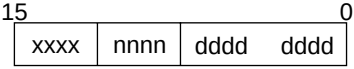
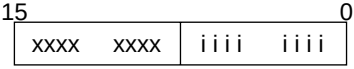
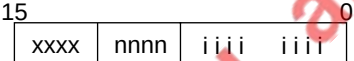
Instruction Formats	Source Operand	Destination Operand	Example
0 format 15 0 	—	—	`NOP`
n format 15 0 	—	nnnn: Direct register	`MOVT Rn`
	Control register or system register	nnnn: Direct register	`STS MACH,Rn`
	Control register or system register	nnnn: Indirect pre-decrement register	`STC.L SR,@-Rn`
m format 15 0 	mmmm: Direct register	Control register or system register	`LDC Rm,SR`
	mmmm: Indirect post-increment register	Control register or system register	`LDC.L @Rm+,SR`
	mmmm: Direct register	—	`JMP @Rm`
	mmmm: PC relative using Rm	—	`BRAF Rm`

Table 2.9 Instruction Formats (cont)

Instruction Formats	Source Operand	Destination Operand	Example
nm format 15 0 xxxx nnnn mmmm xxxx	mmmm: Direct register	nnnn: Direct register	ADD Rm,Rn
	mmmm: Direct register	nnnn: Indirect register	MOV.L Rm,@Rn
	mmmm: Indirect post-increment register (multiply/ accumulate) nnnn*: Indirect post-increment register (multiply/ accumulate)	MACH, MACL	MAC.W @Rm+,@Rn+
	mmmm: Indirect post-increment register	nnnn: Direct register	MOV.L @Rm+,Rn
	mmmm: Direct register	nnnn: Indirect pre-decrement register	MOV.L Rm,@-Rn
	mmmm: Direct register	nnnn: Indirect indexed register	MOV.L Rm,@(R0,Rn)
md format 15 0 xxxx xxxx mmmm dddd	mmmmdddd: indirect register with displacement	R0 (Direct register)	MOV.B @(disp,Rm),R0
nd4 format 15 0 xxxx xxxx nnnn dddd	R0 (Direct register)	nnnndddd: Indirect register with displacement	MOV.B R0,@(disp,Rn)
nmd format 15 0 xxxx nnnn mmmm dddd	mmmm: Direct register	nnnndddd: Indirect register with displacement	MOV.L Rm,@(disp,Rn)
	mmmmdddd: Indirect register with displacement	nnnn: Direct register	MOV.L @(disp,Rm),Rn

Note: * In multiply/accumulate instructions, nnnn is the source register.

Table 2.9 Instruction Formats (cont)

Instruction Formats	Source Operand	Destination Operand	Example
d format 15 0 	dddddddd: Indirect GBR with displacement	R0 (Direct register)	MOV.L @(disp,GBR),R0
	R0(Direct register)	dddddddd: Indirect GBR with displacement	MOV.L R0,@(disp,GBR)
	dddddddd: PC relative with displacement	R0 (Direct register)	MOVA @(disp,PC),R0
	dddddddd: PC relative	—	BF label
d12 format 15 0 	ddddddddddddd: PC relative	—	BRA label (label = disp + PC)
nd8 format 15 0 	dddddddd: PC relative with displacement	nnnn: Direct register	MOV.L @(disp,PC),Rn
i format 15 0 	iiiiiiii: Immediate	Indirect indexed GBR	AND.B #imm,@(R0,GBR)
	iiiiiiii: Immediate	R0 (Direct register)	AND #imm,R0
	iiiiiiii: Immediate	—	TRAPA #imm
ni format 15 0 	iiiiiiii: Immediate	nnnn: Direct register	ADD #imm,Rn

2.4 Instruction Set by Classification

Table 2.10 Classification of Instructions

Classification	Types	Operation Code	Function	No. of Instructions
Data transfer	5	MOV	Data transfer, immediate data transfer, peripheral module data transfer, structure data transfer	39
		MOVA	Effective address transfer	
		MOVT	T bit transfer	
		SWAP	Swap of upper and lower bytes	
		XTRCT	Extraction of the middle of registers connected	
Arithmetic operations	21	ADD	Binary addition	33
		ADDC	Binary addition with carry	
		ADDV	Binary addition with overflow check	
		CMP/cond	Comparison	
		DIV1	Division	
		DIV0S	Initialization of signed division	
		DIV0U	Initialization of unsigned division	
		DMULS	Signed double-length multiplication	
		DMULU	Unsigned double-length multiplication	
		DT	Decrement and test	
		EXTS	Sign extension	
		EXTU	Zero extension	
		MAC	Multiply/accumulate, double-length multiply/accumulate operation	
		MUL	Double-length multiply operation	
		MULS	Signed multiplication	
		MULU	Unsigned multiplication	
		NEG	Negation	
		NEGC	Negation with borrow	
		SUB	Binary subtraction	
		SUBC	Binary subtraction with borrow	
		SUBV	Binary subtraction with underflow	

Table 2.10 Classification of Instructions (cont)

Classification	Types	Operation Code	Function	No. of Instructions
Logic operations	6	AND	Logical AND	14
		NOT	Bit inversion	
		OR	Logical OR	
		TAS	Memory test and bit set	
		TST	Logical AND and T bit set	
		XOR	Exclusive OR	
Shift	10	ROTL	One-bit left rotation	14
		ROTR	One-bit right rotation	
		ROTCL	One-bit left rotation with T bit	
		ROTCR	One-bit right rotation with T bit	
		SHAL	One-bit arithmetic left shift	
		SHAR	One-bit arithmetic right shift	
		SHLL	One-bit logical left shift	
		SHLLn	n-bit logical left shift	
		SHLR	One-bit logical right shift	
		SHLRn	n-bit logical right shift	
Branch	9	BF	Conditional branch, conditional branch with delay (Branch when T = 0)	11
		BT	Conditional branch, conditional branch with delay (Branch when T = 1)	
		BRA	Unconditional branch	
		BRAF	Unconditional branch	
		BSR	Branch to subroutine procedure	
		BSRF	Branch to subroutine procedure	
		JMP	Unconditional branch	
		JSR	Branch to subroutine procedure	
		RTS	Return from subroutine procedure	

Table 2.10 Classification of Instructions (cont)

Classification	Types	Operation Code	Function	No. of Instructions
System control	11	CLRT	T bit clear	31
		CLRMAC	MAC register clear	
		LDC	Load to control register	
		LDS	Load to system register	
		NOP	No operation	
		RTE	Return from exception processing	
		SETT	T bit set	
		SLEEP	Shift into power-down mode	
		STC	Storing control register data	
		STS	Storing system register data	
		TRAPA	Trap exception processing	
Total:	62			142

Table 2.11 shows the format used in tables 2.12 to 2.17, which list instruction codes, operation, and execution states in order by classification.

Table 2.11 Instruction Code Format

Item	Format	Explanation
Instruction	`OP.Sz SRC,DEST`	OP: Operation code Sz: Size (B: byte, W: word, or L: longword) SRC: Source DEST: Destination Rm: Source register Rn: Destination register imm: Immediate data disp: Displacement*1
Instruction code	MSB ↔ LSB	mmmm: Source register nnnn: Destination register 0000: R0 0001: R1 . . . 1111: R15 iiii: Immediate data dddd: Displacement
Operation	→, ←	Direction of transfer
	(xx)	Memory operand
	M/Q/T	Flag bits in the SR
	&	Logical AND of each bit
	|	Logical OR of each bit
	^	Exclusive OR of each bit
	~	Logical NOT of each bit
	<<n	n-bit left shift
	>>n	n-bit right shift
Execution cycles	—	Value when no wait states are inserted*2
T bit	—	Value of T bit after instruction is executed. An em-dash (—) in the column means no change.

Notes: 1. Depending on the operand size, displacement is scaled ×1, ×2, or ×4. For details, see the *SH-1/SH-2/SH-DSP Programming Manual*.

2. Instruction execution cycles: The execution cycles shown in the table are minimums. The actual number of cycles may be increased when (1) contention occurs between instruction fetches and data access, or (2) when the destination register of the load instruction (memory → register) and the register used by the next instruction are the same.

Table 2.12 Data Transfer Instructions

Instruction	Instruction Code	Operation	Execution Cycles	T Bit
MOV #imm,Rn	1110nnnniiiiiiii	#imm → Sign extension → Rn	1	—
MOV.W @(disp,PC),Rn	1001nnnndddddddd	(disp × 2 + PC) → Sign extension → Rn	1	—
MOV.L @(disp,PC),Rn	1101nnnndddddddd	(disp × 4 + PC) → Rn	1	—
MOV Rm,Rn	0110nnnnmmmm0011	Rm → Rn	1	—
MOV.B Rm,@Rn	0010nnnnmmmm0000	Rm → (Rn)	1	—
MOV.W Rm,@Rn	0010nnnnmmmm0001	Rm → (Rn)	1	—
MOV.L Rm,@Rn	0010nnnnmmmm0010	Rm → (Rn)	1	—
MOV.B @Rm,Rn	0110nnnnmmmm0000	(Rm) → Sign extension → Rn	1	—
MOV.W @Rm,Rn	0110nnnnmmmm0001	(Rm) → Sign extension → Rn	1	—
MOV.L @Rm,Rn	0110nnnnmmmm0010	(Rm) → Rn	1	—
MOV.B Rm,@–Rn	0010nnnnmmmm0100	Rn–1 → Rn, Rm → (Rn)	1	—
MOV.W Rm,@–Rn	0010nnnnmmmm0101	Rn–2 → Rn, Rm → (Rn)	1	—
MOV.L Rm,@–Rn	0010nnnnmmmm0110	Rn–4 → Rn, Rm → (Rn)	1	—
MOV.B @Rm+,Rn	0110nnnnmmmm0100	(Rm) → Sign extension → Rn,Rm + 1 → Rm	1	—
MOV.W @Rm+,Rn	0110nnnnmmmm0101	(Rm) → Sign extension → Rn,Rm + 2 → Rm	1	—
MOV.L @Rm+,Rn	0110nnnnmmmm0110	(Rm) → Rn,Rm + 4 → Rm	1	—
MOV.B R0,@(disp,Rn)	10000000nnnndddd	R0 → (disp + Rn)	1	—
MOV.W R0,@(disp,Rn)	10000001nnnndddd	R0 → (disp × 2 + Rn)	1	—
MOV.L Rm,@(disp,Rn)	0001nnnnmmmmdddd	Rm → (disp × 4 + Rn)	1	—
MOV.B @(disp,Rm),R0	10000100mmmmdddd	(disp + Rm) → Sign extension → R0	1	—
MOV.W @(disp,Rm),R0	10000101mmmmdddd	(disp × 2 + Rm) → Sign extension → R0	1	—
MOV.L @(disp,Rm),Rn	0101nnnnmmmmdddd	(disp × 4 + Rm) → Rn	1	—
MOV.B Rm,@(R0,Rn)	0000nnnnmmmm0100	Rm → (R0 + Rn)	1	—

Table 2.12 Data Transfer Instructions (cont)

Instruction		Instruction Code	Operation	Execution Cycles	T Bit
MOV.W	Rm,@(R0,Rn)	0000nnnnmmmm0101	Rm → (R0 + Rn)	1	—
MOV.L	Rm,@(R0,Rn)	0000nnnnmmmm0110	Rm → (R0 + Rn)	1	—
MOV.B	@(R0,Rm),Rn	0000nnnnmmmm1100	(R0 + Rm) → Sign extension → Rn	1	—
MOV.W	@(R0,Rm),Rn	0000nnnnmmmm1101	(R0 + Rm) → Sign extension → Rn	1	—
MOV.L	@(R0,Rm),Rn	0000nnnnmmmm1110	(R0 + Rm) → Rn	1	—
MOV.B	R0,@(disp,GBR)	11000000dddddddd	R0 → (disp + GBR)	1	—
MOV.W	R0,@(disp,GBR)	11000001dddddddd	R0 → (disp × 2 + GBR)	1	—
MOV.L	R0,@(disp,GBR)	11000010dddddddd	R0 → (disp × 4 + GBR)	1	—
MOV.B	@(disp,GBR),R0	11000100dddddddd	(disp + GBR) → Sign extension → R0	1	—
MOV.W	@(disp,GBR),R0	11000101dddddddd	(disp × 2 + GBR) → Sign extension → R0	1	—
MOV.L	@(disp,GBR),R0	11000110dddddddd	(disp × 4 + GBR) → R0	1	—
MOVA	@(disp,PC),R0	11000111dddddddd	disp × 4 + PC → R0	1	—
MOVT	Rn	0000nnnn00101001	T → Rn	1	—
SWAP.B	Rm,Rn	0110nnnnmmmm1000	Rm → Swap the bottom two bytes → Rn	1	—
SWAP.W	Rm,Rn	0110nnnnmmmm1001	Rm → Swap two consecutive words → Rn	1	—
XTRCT	Rm,Rn	0010nnnnmmmm1101	Rm: Middle 32 bits of Rn → Rn	1	—

Table 2.13 Arithmetic Operation Instructions

Instruction		Instruction Code	Operation	Execution Cycles	T Bit
ADD	Rm, Rn	0011nnnnmmmm1100	Rn + Rm → Rn	1	—
ADD	#imm, Rn	0111nnnniiiiiiii	Rn + imm → Rn	1	—
ADDC	Rm, Rn	0011nnnnmmmm1110	Rn + Rm + T → Rn, Carry → T	1	Carry
ADDV	Rm, Rn	0011nnnnmmmm1111	Rn + Rm → Rn, Overflow → T	1	Overflow
CMP/EQ	#imm, R0	10001000iiiiiiii	If R0 = imm, 1 → T	1	Comparison result
CMP/EQ	Rm, Rn	0011nnnnmmmm0000	If Rn = Rm, 1 → T	1	Comparison result
CMP/HS	Rm, Rn	0011nnnnmmmm0010	If Rn ≥ Rm with unsigned data, 1 → T	1	Comparison result
CMP/GE	Rm, Rn	0011nnnnmmmm0011	If Rn ≥ Rm with signed data, 1 → T	1	Comparison result
CMP/HI	Rm, Rn	0011nnnnmmmm0110	If Rn > Rm with unsigned data, 1 → T	1	Comparison result
CMP/GT	Rm, Rn	0011nnnnmmmm0111	If Rn > Rm with signed data, 1 → T	1	Comparison result
CMP/PL	Rn	0100nnnn00010101	If Rn > 0, 1 → T	1	Comparison result
CMP/PZ	Rn	0100nnnn00010001	If Rn ≥ 0, 1 → T	1	Comparison result
CMP/STR	Rm, Rn	0010nnnnmmmm1100	If Rn and Rm have an equivalent byte, 1 → T	1	Comparison result
DIV1	Rm, Rn	0011nnnnmmmm0100	Single-step division (Rn/Rm)	1	Calculation result
DIV0S	Rm, Rn	0010nnnnmmmm0111	MSB of Rn → Q, MSB of Rm → M, M ^ Q → T	1	Calculation result
DIV0U		0000000000011001	0 → M/Q/T	1	0

Table 2.13 Arithmetic Operation Instructions (cont)

Instruction		Instruction Code	Operation	Execution Cycles	T Bit
DMULS.L	Rm,Rn	0011nnnnmmmm1101	Signed operation of Rn × Rm → MACH, MACL 32 × 32 → 64 bit	2 to 4*	—
DMULU.L	Rm,Rn	0011nnnnmmmm0101	Unsigned operation of Rn × Rm → MACH, MACL 32 × 32 → 64 bit	2 to 4*	—
DT	Rn	0100nnnn00010000	Rn – 1 → Rn, when Rn is 0, 1 → T. When Rn is nonzero, 0 → T	1	Comparison result
EXTS.B	Rm,Rn	0110nnnnmmmm1110	A byte in Rm is sign-extended → Rn	1	—
EXTS.W	Rm,Rn	0110nnnnmmmm1111	A word in Rm is sign-extended → Rn	1	—
EXTU.B	Rm,Rn	0110nnnnmmmm1100	A byte in Rm is zero-extended → Rn	1	—
EXTU.W	Rm,Rn	0110nnnnmmmm1101	A word in Rm is zero-extended → Rn	1	—
MAC.L	@Rm+,@Rn+	0000nnnnmmmm1111	Signed operation of (Rn) × (Rm) + MAC → MAC 32 × 32 → 64 bit	3/(2 to 4)*	—
MAC.W	@Rm+,@Rn+	0100nnnnmmmm1111	Signed operation of (Rn) × (Rm) + MAC → MAC 16 × 16 + 64 → 64 bit	3/(2)*	—
MUL.L	Rm,Rn	0000nnnnmmmm0111	Rn × Rm → MACL, 32 × 32 → 32 bit	2 to 4*	—
MULS.W	Rm,Rn	0010nnnnmmmm1111	Signed operation of Rn × Rm → MAC 16 × 16 → 32 bit	1 to 3*	—
MULU.W	Rm,Rn	0010nnnnmmmm1110	Unsigned operation of Rn × Rm → MAC 16 × 16 → 32 bit	1 to 3*	—
NEG	Rm,Rn	0110nnnnmmmm1011	0–Rm → Rn	1	—
NEGC	Rm,Rn	0110nnnnmmmm1010	0–Rm–T → Rn, Borrow → T	1	Borrow

Table 2.13 Arithmetic Operation Instructions (cont)

Instruction		Instruction Code	Operation	Execution Cycles	T Bit
SUB	Rm,Rn	0011nnnnmmmm1000	Rn–Rm → Rn	1	—
SUBC	Rm,Rn	0011nnnnmmmm1010	Rn–Rm–T → Rn, Borrow → T	1	Borrow
SUBV	Rm,Rn	0011nnnnmmmm1011	Rn–Rm → Rn, Underflow → T	1	Overflow

Note: * The normal minimum number of execution cycles. (The number in parentheses is the number of cycles when there is contention with following instructions.)

Table 2.14 Logic Operation Instructions

Instruction		Instruction Code	Operation	Execution Cycles	T Bit
AND	Rm,Rn	0010nnnnmmmm1001	Rn & Rm → Rn	1	—
AND	#imm,R0	11001001iiiiiiii	R0 & imm → R0	1	—
AND.B	#imm,@(R0,GBR)	11001101iiiiiiii	(R0 + GBR) & imm → (R0 + GBR)	3	—
NOT	Rm,Rn	0110nnnnmmmm0111	~Rm → Rn	1	—
OR	Rm,Rn	0010nnnnmmmm1011	Rn | Rm → Rn	1	—
OR	#imm,R0	11001011iiiiiiii	R0 | imm → R0	1	—
OR.B	#imm,@(R0,GBR)	11001111iiiiiiii	(R0 + GBR) | imm → (R0 + GBR)	3	—
TAS.B	@Rn	0100nnnn00011011	If (Rn) is 0, 1 → T; 1 → MSB of (Rn)	4	Test result
TST	Rm,Rn	0010nnnnmmmm1000	Rn & Rm; if the result is 0, 1 → T	1	Test result
TST	#imm,R0	11001000iiiiiiii	R0 & imm; if the result is 0, 1 → T	1	Test result
TST.B	#imm,@(R0,GBR)	11001100iiiiiiii	(R0 + GBR) & imm; if the result is 0, 1 → T	3	Test result
XOR	Rm,Rn	0010nnnnmmmm1010	Rn ^ Rm → Rn	1	—
XOR	#imm,R0	11001010iiiiiiii	R0 ^ imm → R0	1	—
XOR.B	#imm,@(R0,GBR)	11001110iiiiiiii	(R0 + GBR) ^ imm → (R0 + GBR)	3	—

Table 2.15 Shift Instructions

Instruction		Instruction Code	Operation	Execution Cycles	T Bit
ROTL	Rn	0100nnnn00000100	T ← Rn ← MSB	1	MSB
ROTR	Rn	0100nnnn00000101	LSB → Rn → T	1	LSB
ROTCL	Rn	0100nnnn00100100	T ← Rn ← T	1	MSB
ROTCR	Rn	0100nnnn00100101	T → Rn → T	1	LSB
SHAL	Rn	0100nnnn00100000	T ← Rn ← 0	1	MSB
SHAR	Rn	0100nnnn00100001	MSB → Rn → T	1	LSB
SHLL	Rn	0100nnnn00000000	T ← Rn ← 0	1	MSB
SHLR	Rn	0100nnnn00000001	0 → Rn → T	1	LSB
SHLL2	Rn	0100nnnn00001000	Rn<<2 → Rn	1	—
SHLR2	Rn	0100nnnn00001001	Rn>>2 → Rn	1	—
SHLL8	Rn	0100nnnn00011000	Rn<<8 → Rn	1	—
SHLR8	Rn	0100nnnn00011001	Rn>>8 → Rn	1	—
SHLL16	Rn	0100nnnn00101000	Rn<<16 → Rn	1	—
SHLR16	Rn	0100nnnn00101001	Rn>>16 → Rn	1	—

Table 2.16 Branch Instructions

Instruction		Instruction Code	Operation	Exec. Cycles	T Bit
BF	label	10001011dddddddd	If T = 0, disp × 2 + PC → PC; if T = 1, nop	3/1*	—
BF/S	label	10001111dddddddd	Delayed branch, if T = 0, disp × 2 + PC → PC; if T = 1, nop	3/1*	—
BT	label	10001001dddddddd	If T = 1, disp × 2 + PC → PC; if T = 0, nop	3/1*	—
BT/S	label	10001101dddddddd	Delayed branch, if T = 1, disp × 2 + PC → PC; if T = 0, nop	2/1*	—
BRA	label	1010dddddddddddd	Delayed branch, disp × 2 + PC → PC	2	—
BRAF	Rm	0000mmmm00100011	Delayed branch, Rm + PC → PC	2	—
BSR	label	1011dddddddddddd	Delayed branch, PC → PR, disp × 2 + PC → PC	2	—
BSRF	Rm	0000mmmm00000011	Delayed branch, PC → PR, Rm + PC → PC	2	—
JMP	@Rm	0100mmmm00101011	Delayed branch, Rm → PC	2	—
JSR	@Rm	0100mmmm00001011	Delayed branch, PC → PR, Rm → PC	2	—
RTS		0000000000001011	Delayed branch, PR → PC	2	—

Note: * One state when it does not branch.

Table 2.17 System Control Instructions

Instruction		Instruction Code	Operation	Exec. Cycles	T Bit
CLRT		0000000000001000	0 → T	1	0
CLRMAC		0000000000101000	0 → MACH, MACL	1	—
LDC	Rm,SR	0100mmmm00001110	Rm → SR	1	LSB
LDC	Rm,GBR	0100mmmm00011110	Rm → GBR	1	—
LDC	Rm,VBR	0100mmmm00101110	Rm → VBR	1	—
LDC.L	@Rm+,SR	0100mmmm00000111	(Rm) → SR, Rm + 4 → Rm	3	LSB
LDC.L	@Rm+,GBR	0100mmmm00010111	(Rm) → GBR, Rm + 4 → Rm	3	—
LDC.L	@Rm+,VBR	0100mmmm00100111	(Rm) → VBR, Rm + 4 → Rm	3	—
LDS	Rm,MACH	0100mmmm00001010	Rm → MACH	1	—
LDS	Rm,MACL	0100mmmm00011010	Rm → MACL	1	—
LDS	Rm,PR	0100mmmm00101010	Rm → PR	1	—
LDS.L	@Rm+,MACH	0100mmmm00000110	(Rm) → MACH, Rm + 4 → Rm	1	—
LDS.L	@Rm+,MACL	0100mmmm00010110	(Rm) → MACL, Rm + 4 → Rm	1	—
LDS.L	@Rm+,PR	0100mmmm00100110	(Rm) → PR, Rm + 4 → Rm	1	—
NOP		0000000000001001	No operation	1	—
RTE		0000000000101011	Delayed branch, stack area → PC/SR	4	—
SETT		0000000000011000	1 → T	1	1
SLEEP		0000000000011011	Sleep	3*	—
STC	SR,Rn	0000nnnn00000010	SR → Rn	1	—
STC	GBR,Rn	0000nnnn00010010	GBR → Rn	1	—
STC	VBR,Rn	0000nnnn00100010	VBR → Rn	1	—
STC.L	SR,@–Rn	0100nnnn00000011	Rn–4 → Rn, SR → (Rn)	2	—
STC.L	GBR,@–Rn	0100nnnn00010011	Rn–4 → Rn, GBR → (Rn)	2	—
STC.L	VBR,@–Rn	0100nnnn00100011	Rn–4 → Rn, BR → (Rn)	2	—
STS	MACH,Rn	0000nnnn00001010	MACH → Rn	1	—
STS	MACL,Rn	0000nnnn00011010	MACL → Rn	1	—
STS	PR,Rn	0000nnnn00101010	PR → Rn	1	—

Table 2.17 System Control Instructions (cont)

Instruction		Instruction Code	Operation	Exec. Cycles	T Bit
STS.L	MACH,@–Rn	0100nnnn00000010	Rn–4 → Rn, MACH → (Rn)	1	—
STS.L	MACL,@–Rn	0100nnnn00010010	Rn–4 → Rn, MACL → (Rn)	1	—
STS.L	PR,@–Rn	0100nnnn00100010	Rn–4 → Rn, PR → (Rn)	1	—
TRAPA	#imm	11000011iiiiiiii	PC/SR → stack area, (imm) → PC	8	—

Note: * The number of execution cycles before the chip enters sleep mode: The execution cycles shown in the table are minimums. The actual number of cycles may be increased when (1) contention occurs between instruction fetches and data access, or (2) when the destination register of the load instruction (memory → register) and the register used by the next instruction are the same.

2.5 Processing States

2.5.1 State Transitions

The CPU has for processing states: reset, exception processing, program execution, and power-down. Figure 2.6 shows the transitions between the states.

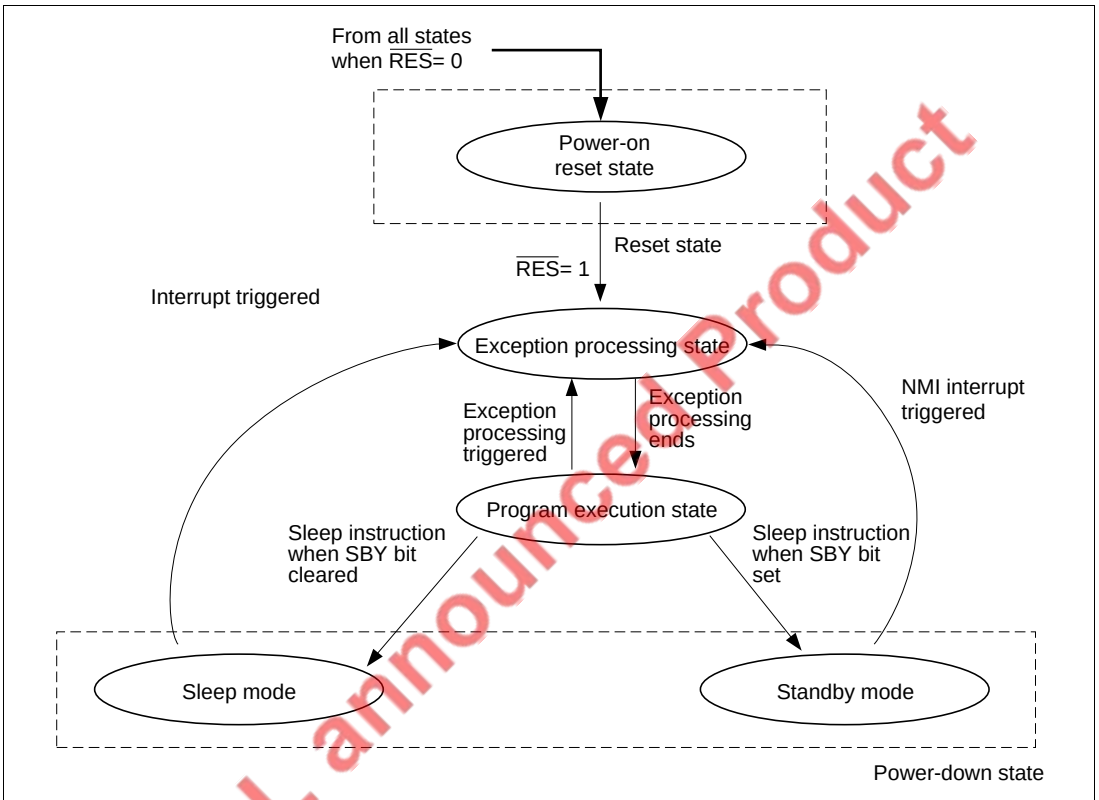


Figure 2.6 Transitions Between Processing States

Reset State: The CPU resets in this state. When the $\overline{\text{RES}}$ pin goes low, a power-on reset results.

Exception Processing State: This is a transient state that occurs when the CPU's processing state flow is altered due to the triggering of exception processing.

In the case of a reset, the execution start address and stack pointer (SP) initial value are fetched from the exception processing vector table as the initial values of the program counter (PC) and stored. The CPU then branches to the execution start address and execution of the program begins.

In the case of an interrupt or the like, the SP is accessed and the PC and status register (SR) are saved to the stack area. The exception service routine start address is fetched from the exception processing vector table. The CPU then branches to that address and execution of the program begins.

The processing state which follows is the program execution state.

Program Execution State: In this state the CPU executes programs sequentially.

Power-Down State: In this state CPU operation halts in order to consume less power. The SLEEP instruction causes the CPU to enter the power-down state. This state has two modes, the sleep mode and the standby mode.

2.5.2 Power-Down State

The power-down state is one of the CPU's processing states. In this state CPU operation halts, in addition to the execution of programs, and power consumption is reduced. It has two modes, the sleep mode and the standby mode.

Sleep Mode: Issuing the SLEEP instruction when the standby bit (SBY) of standby control register (SBYCR) is cleared to 0 causes the CPU to enter the sleep mode. In the sleep mode CPU operation is halted, but data stored in the CPU's internal registers and in on-chip RAM is retained. The functions of on-chip peripheral modules other than the CPU do not halt.

A power-on reset or any interrupt causes the CPU to recover from the sleep mode. Exception processing then takes place, after which the CPU enters the normal program execution mode.

Standby Mode: Issuing the SLEEP instruction when SBY of SBYCR is set to 1 causes the CPU to enter the standby mode. In the standby mode the functioning of the CPU, on-chip peripheral modules, and oscillator are halted. If the chip enters the standby mode while a multiplier instruction is executing, the MACH and MACL values become uncertain.

A power-on reset or NMI interrupt causes the chip to recover from the sleep mode. After a reset the oscillator stabilization time elapses, after which exception processing takes place and then the chip enters the normal program execution mode. In the case of an NMI interrupt, the oscillator stabilization time must elapse, after which exception processing takes place and then the chip enters the normal program execution mode.

This mode halts the operation of the oscillator, so power consumption is decreased substantially.

Table 2.18 Power-Down State

Mode	Transition Conditions	State						Recovery Method
		Clock	CPU	On-Chip Peripheral Modules	Contents of CPU Registers	Contents of On-Chip RAM	I/O Port Pin States	
Sleep mode	SLEEP command issued when SBY of SBYCR is cleared to 0	Operating	Halted	Operating	Retained	Retained	Retained	(1) Interrupt (2) Power-on reset
Standby mode	SLEEP command issued when SBY of SBYCR is set to 1	Halted	Halted	Halted or initialized*	Retained	Retained	Retained or Hi-Z (can be specified by user)	(1) NMI interrupt (2) Power-on reset

Note: * Differs depending on the peripheral module and pin.

Section 3 Operating Modes

3.1 Operating Mode Selection

The SH7018 has four operating modes: MCU mode, boot mode, user program mode, and programmer mode. The settings of the mode pins (MD3 to MD0) determine the mode in which the chip operates. The mode pin settings must not be changed while the chip is operating (while power is being supplied).

The method of selecting the operating mode is shown in table 3.1.

Table 3.1 Operating Mode Selection

Pin Settings							
FWP	**MD3*[1]**	**MD2*[1]**	**MD1**	**MD0**	**Operating Mode**	**On-Chip ROM**	**CS0 Bus Width**
1	0	0	1	0	MCU mode	Enabled	8 bits*[2]
0	0	0	0	0	Boot mode*[1]	Enabled	8 bits*[2]
0	0	0	1	0	User program mode*[1]	Enabled	8 bits*[2]
1	1	1	0	1	Programmer mode*[1]	Enabled	—

Notes: 1. F-ZTAT version only.
2. Set using BCR1 of BSC.

3.2 Description of Operating Modes

MCU Mode: The on-chip ROM is enabled in the MCU mode. The bus width for the on-chip ROM space is 32 bits.

Boot Mode: Refer to section 16.6.1 Boot Mode for information on the boot mode.

User Program Mode: Refer to section 16.6.2 User Program Mode for information on the user program mode.

Programmer Mode: Refer to section 16.11 Flash Memory Programmer Mode for information on the programmer mode.

3.3 Pin Configuration

The functions of the pins relating to the operating modes are shown in table 3.2.

Table 3.2 Pin Functions

Pin Name	I/O	Function
XTAL	Input	Connected to the crystal resonator.
EXTAL	Input	Connected to the crystal resonator, or used as input pin for external clock.
MD0	Input	The level at this pin is used in the operating mode specification.
MD1	Input	The level at this pin is used in the operating mode specification.
MD2	Input	The level at this pin is used in the operating mode specification.
MD3	Input	The level at this pin is used in the operating mode specification.

Section 4 Clock Pulse Generator (CPG)

4.1 Overview

The clock pulse generator (CPG) supplies clock pulses within the SH7018 and to external devices. The SH7018's CPG operates the SH7018 at a frequency equal to the oscillation frequency of the crystal resonator. The CPG is composed of an oscillator and a duty adjustment circuit (figure 4.1). There are two ways of generating a clock with the CPG: by connecting a crystal resonator, or by inputting an external clock.

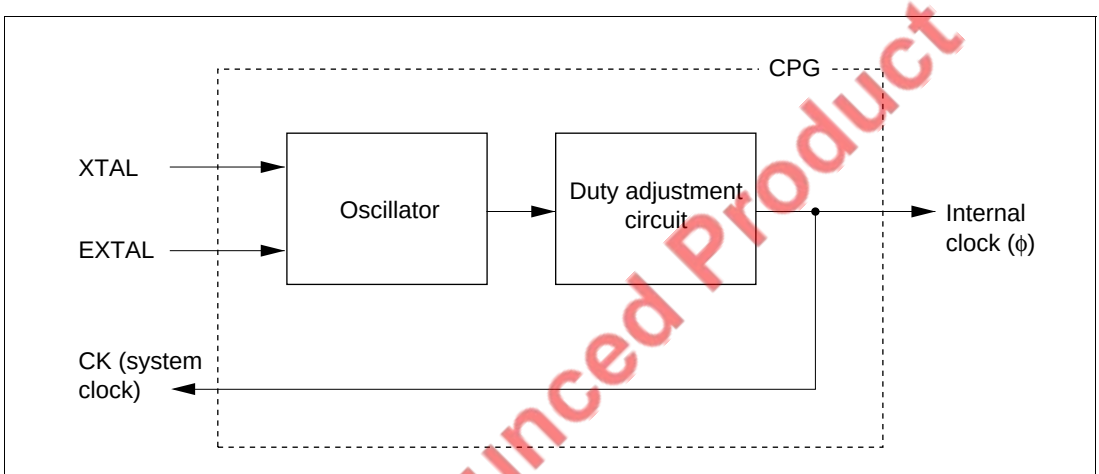


Figure 4.1 CPG Block Diagram

4.2 Clock Source

Either a crystal resonator or an external clock can be selected as the clock pulse source.

4.2.1 Crystal Resonator Connection

Circuit Configuration: Figure 4.2 shows the method of connecting a crystal resonator. Use the damping resistance (Rd) shown in table 4.1. An AT-cut parallel-resonance type crystal resonator with the same frequency as the system clock (CK) should be used. Load capacitors (C_{L1}, C_{L2}) must be connected as shown in the figure.

The clock pulses generated by the crystal resonator and internal oscillator are sent to the duty adjustment circuit. After the duty has been adjusted, the pulses are supplied within the SH7018 chip and to external devices.

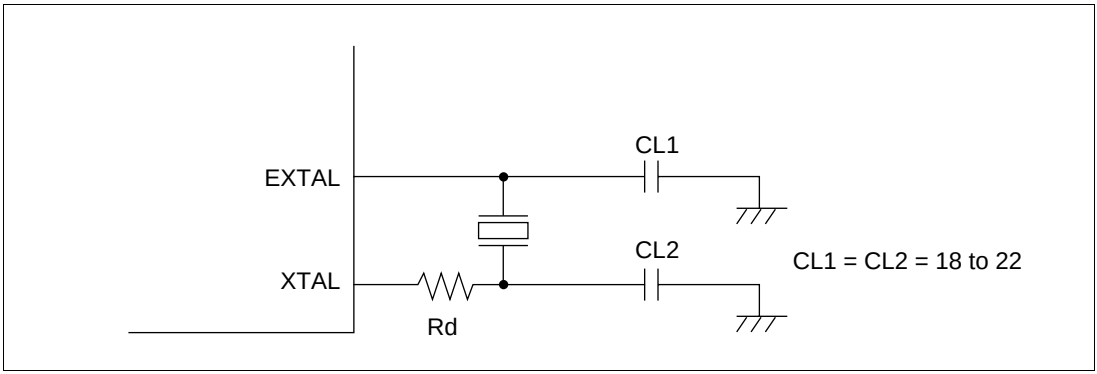


Figure 4.2 Example of Crystal Resonator Connection

Table 4.1 Damping Resistance Value

Frequency (MHz)	20
Rd (Ω)	0

Crystal Resonator: Figure 4.3 shows an equivalent circuit for the crystal resonator. Use a crystal resonator with the characteristics shown in table 4.2.

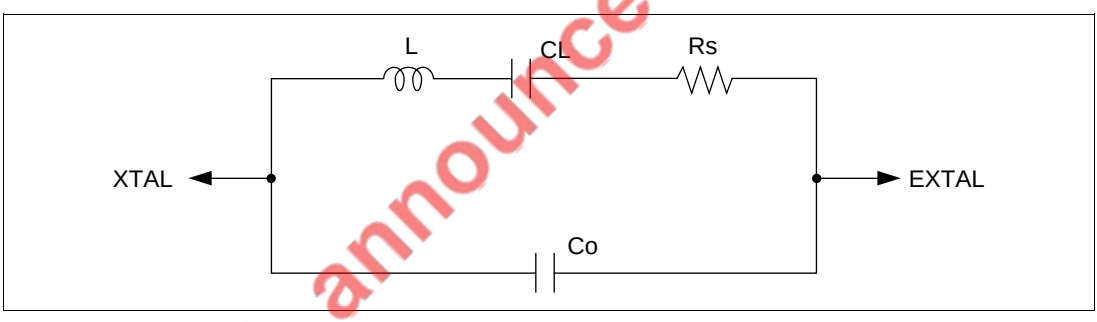


Figure 4.3 Crystal Resonator Equivalent Circuit

Table 4.2 Crystal Resonator Characteristics

	Frequency (MHz)
Parameter	**20**
Rs max (Ω)	60
Co max (pF)	7

4.2.2 External Clock Input

Input the external clock to the EXTAL pin and leave the XTAL pin open (figure 4.4.). The external clock frequency should be the same as that of the system clock (CK).

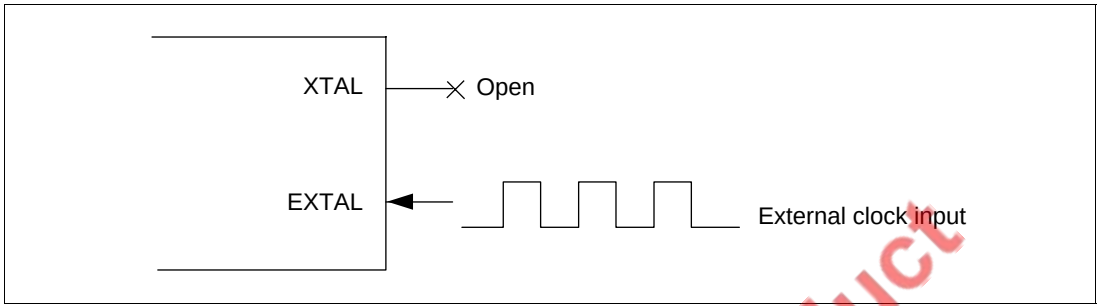


Figure 4.4 External Clock Input

4.3 Usage Notes

Note on Board Design: Place the crystal resonator and load capacitors as close as possible to the EXTAL and XTAL pins. To prevent induction from interfering with correct oscillation, ensure that other signal lines to not cross the EXTAL and XTAL pin signal lines.

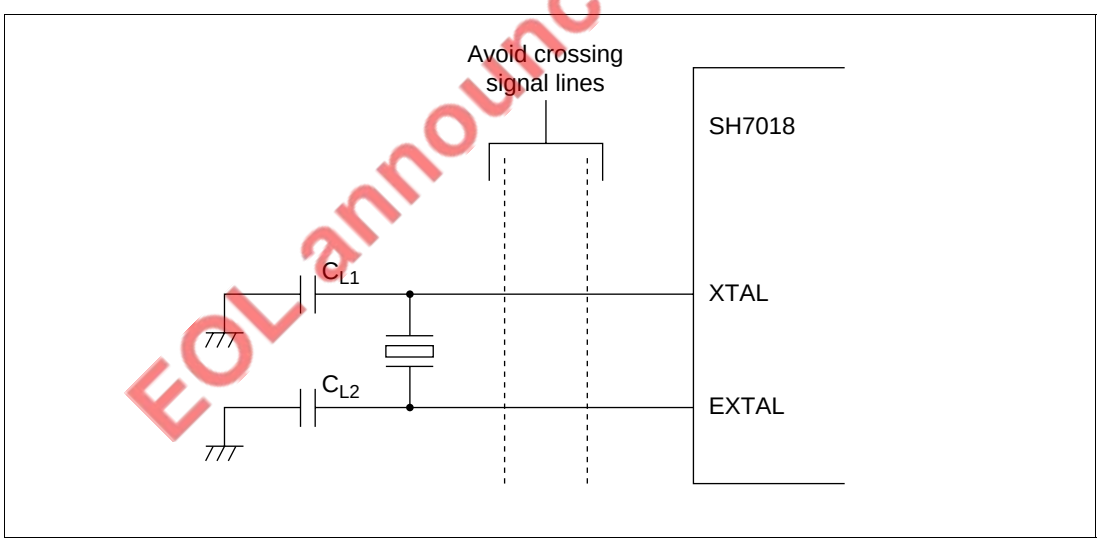


Figure 4.5 Note on Board Design

Notes on Duty Adjustment: Duty adjustment circuit is performed on an input clock of 5 MHz or higher. With a frequency of less than 5 MHz, duty adjustment may not be performed, but AC characteristics t_{CH} (clock high-level width) and t_{CL} (clock low-level width) are satisfied, and there is no problem with SH7018 internal operation. Figure 4.6 shows the basic characteristics of the duty adjustment circuit.

The duty adjustment circuit does not correct for transient fluctuations or jitter in the input clock. Thus, several tens of µs are required until duty adjustment is performed and a stable clock is obtained.

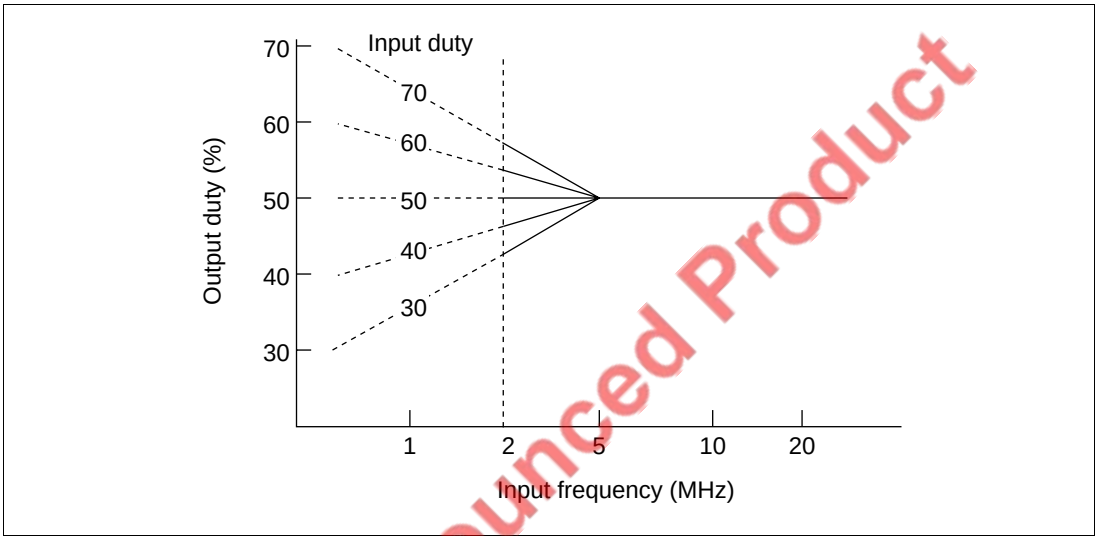


Figure 4.6 Duty Adjustment Circuit Characteristics

Section 5 Exception Processing

5.1 Overview

5.1.1 Types of Exception Processing and Priority

Exception processing is started by four sources: resets, address errors, interrupts and instructions and have the priority shown in table 5.1. When several exception processing sources occur at once, they are processed according to the priority shown.

Table 5.1 Types of Exception Processing and Priority Order

Exception	Source	Priority
Reset	Power-on reset	High
Address error	CPU address error	
Interrupt	NMI	
	IRQ	
	On-chip peripheral modules: • Multifunction timer/pulse unit (MTU) • Serial communications interface (SCI) • A/D converter (A/D) • Compare match timer (CMT) • Watchdog timer (WDT) • 8-bit timer 2 (TIM2)	
Instructions	Trap instruction (TRAPA instruction)	
	General illegal instructions (undefined code)	
	Illegal slot instructions (undefined code placed directly after a delay branch instruction*1 or instructions that rewrite the PC*2)	Low

Notes: 1. Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE, BF/S, BT/S, BSRF, BRAF.
2. Instructions that rewrite the PC: JMP, JSR, BRA, BSR, RTS, RTE, BT, BF, TRAPA, BF/S, BT/S, BSRF, BRAF.

5.1.2 Exception Processing Operations

The exception processing sources are detected and begin processing according to the timing shown in table 5.2.

Table 5.2 Timing of Exception Source Detection and the Start of Exception Processing

Exception	Source	Timing of Source Detection and Start of Processing
Power-on reset		Starts when the $\overline{\text{RES}}$ pin changes from low to high.
Address error		Detected when instruction is decoded and starts when the previous executing instruction finishes executing.
Interrupts		Detected when instruction is decoded and starts when the previous executing instruction finishes executing.
Instructions	Trap instruction	Starts from the execution of a TRAPA instruction.
	General illegal instructions	Starts from the decoding of undefined code anytime except after a delayed branch instruction (delay slot).
	Illegal slot instructions	Starts from the decoding of undefined code placed in a delayed branch instruction (delay slot) or of instructions that rewrite the PC.

When exception processing starts, the CPU operates as follows:

1. Exception processing triggered by reset:

   The initial values of the program counter (PC) and stack pointer (SP) are fetched from the exception processing vector table (PC and SP are respectively the H'00000000 and H'00000004 addresses). See section 5.1.3, Exception Processing Vector Table, for more information. 0 is then written to the vector base register (VBR) and 1111 is written to the interrupt mask bits (I3 to I0) of the status register (SR). The program begins running from the PC address fetched from the exception processing vector table.
2. Exception processing triggered by address errors, interrupts and instructions:

   SR and PC are saved to the stack indicated by R15. For interrupt exception processing, the interrupt priority level is written to the SR's interrupt mask bits (I3 to I0). For address error and instruction exception processing, the I3 to I0 bits are not affected. The start address is then fetched from the exception processing vector table and the program begins running from that address.

5.1.3 Exception Processing Vector Table

Before exception processing begins running, the exception processing vector table must be set in memory. The exception processing vector table stores the start addresses of exception service routines. (The reset exception processing table holds the initial values of PC and SP.)

All exception sources are given different vector numbers and vector table address offsets, from which the vector table addresses are calculated. During exception processing, the start addresses of the exception service routines are fetched from the exception processing vector table, which indicated by this vector table address.

Table 5.3 shows the vector numbers and vector table address offsets. Table 5.4 shows how vector table addresses are calculated.

Table 5.3 Exception Processing Vector Table

Exception Sources		Vector Numbers	Vector Table Address Offset
Power-on reset	PC	0	H'00000000 to H'00000003
	SP	1	H'00000004 to H'00000007
(Reserved by system)		2	H'00000008 to H'0000000F
		3	
General illegal instruction		4	H'00000010 to H'00000013
(Reserved by system)		5	H'00000014 to H'00000017
Slot illegal instruction		6	H'00000018 to H'0000001B
(Reserved by system)		7	H'0000001C to H'0000001F
(Reserved by system)		8	H'00000020 to H'00000023
CPU address error		9	H'00000024 to H'00000027
(Reserved by system)		10	H'00000028 to H'0000002B
Interrupts	NMI	11	H'0000002C to H'0000002F
(Reserved by system)		12	H'00000030 to H'00000033
		:	:
		31	H'0000007C to H'0000007F
Trap instruction (user vector)		32	H'00000080 to H'00000083
		:	:
		63	H'000000FC to H'000000FF

Table 5.3 Exception Processing Vector Table (cont)

Exception Sources		Vector Numbers	Vector Table Address Offset
Interrupts	IRQ0	64	H'00000100 to H'00000103
	IRQ1	65	H'00000104 to H'00000107
	IRQ2	66	H'00000108 to H'0000010B
	IRQ3	67	H'0000010C to H'0000010F
(Reserved by system)		68	H'00000110 to H'00000113
		69	H'00000114 to H'00000117
Interrupts	IRQ6	70	H'00000118 to H'0000011B
	IRQ7	71	H'0000011C to H'0000011F
On-chip peripheral module*		72	H'00000120 to H'00000124
		:	:
		255	H'000003FC to H'000003FF

Note: * The vector numbers and vector table address offsets for each on-chip peripheral module interrupt are given in section 6, Interrupt Controller, table 6.3, Interrupt Exception Processing Vectors and Priorities.

Table 5.4 Calculating Exception Processing Vector Table Addresses

Exception Source	Vector Table Address Calculation
Resets	Vector table address = (vector table address offset) = (vector number) × 4
Address errors, interrupts, instructions	Vector table address = VBR + (vector table address offset) = VBR + (vector number) × 4

Notes: 1. VBR: Vector base register
2. Vector table address offset: See table 5.3.
3. Vector number: See table 5.3.

5.2 Resets

5.2.1 Reset

A reset has the highest priority of any exception source. As shown in table 5.5, a power-on reset initializes the internal state of the CPU and the on-chip peripheral module registers.

Table 5.5 Types of Resets

	Conditions for Transition to Reset Status	Internal Status	
Type	**$\overline{\text{RES}}$**	**CPU**	**On-Chip Peripheral Module**
Power-on reset	Low	Initialized	Initialized

5.2.2 Power-On Reset

When the $\overline{\text{RES}}$ pin is driven low, the LSI does a power-on reset. To reliably reset the LSI, the $\overline{\text{RES}}$ pin should be kept at low for at least the duration of the oscillation settling time when applying power or when in standby mode (when the clock circuit is halted) or at least 20 t_{cyc} (when the clock circuit is running). During power-on reset, CPU internal status and all registers of on-chip peripheral modules are initialized. See Appendix B, Pin Status, for the status of individual pins during the power-on reset status.

In the power-on reset status, power-on reset exception processing starts when the $\overline{\text{RES}}$ pin is first driven low for a set period of time and then returned to high. The CPU will then operate as follows:

1. The initial value (execution start address) of the program counter (PC) is fetched from the exception processing vector table.
2. The initial value of the stack pointer (SP) is fetched from the exception processing vector table.
3. The vector base register (VBR) is cleared to H'00000000 and the interrupt mask bits (I3 to I0) of the status register (SR) are set to H'F (1111).
4. The values fetched from the exception processing vector table are set in the program counter (PC) and SP and the program begins executing.

Be certain to always perform power-on reset processing when turning the system power on.

5.3 Address Errors

Address errors occur when instructions are fetched or data read or written, as shown in table 5.6.

Table 5.6 Bus Cycles and Address Errors

Bus Cycle Type	Bus Cycle Description	Address Errors
Instruction fetch	Instruction fetched from even address	None (normal)
	Instruction fetched from odd address	Address error occurs
	Instruction fetched from other than on-chip peripheral module space*	None (normal)
	Instruction fetched from on-chip peripheral module space*	Address error occurs
Data read/write	Word data accessed from even address	None (normal)
	Word data accessed from odd address	Address error occurs
	Longword data accessed from other than a longword boundary	Address error occurs
	Byte or word data accessed in on-chip peripheral module space*	None (normal)
	Longword data accessed in 16-bit on-chip peripheral module space*	None (normal)
	Longword data accessed in 8-bit on-chip peripheral module space*	Address error occurs

Note: * See section 7, Bus State Controller.

5.3.1 Address Error Exception Processing

When an address error occurs, the bus cycle in which the address error occurred ends. When the executing instruction then finishes, address error exception processing starts up. The CPU operates as follows:

1. The status register (SR) is saved to the stack.
2. The program counter (PC) is saved to the stack. The PC value saved is the start address of the instruction to be executed after the last executed instruction.
3. The exception service routine start address is fetched from the exception processing vector table that corresponds to the address error that occurred and the program starts executing from that address. The jump that occurs is not a delayed branch.

5.4 Interrupts

Table 5.7 shows the sources that start up interrupt exception processing. These are divided into NMI, IRQ and on-chip peripheral modules.

Table 5.7 Interrupt Sources

Type	Request Source	Number of Sources
NMI	NMI pin (external input)	1
IRQ	$\overline{\text{IRQ0}}$ to $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$ (external input)	6
On-chip peripheral module	Multifunction timer/pulse unit (MTU)	11
	Serial communications interface (SCI)	4
	A/D converter	1
	Compare match timer (CMT)	2
	Watchdog Timer (WDT)	1
	8-bit timer 2 (TIM2)	1

Each interrupt source is allocated a different vector number and vector table offset. See section 6, Interrupt Controller, table 6.3, Interrupt Exception Processing Vectors and Priorities, for more information on vector numbers and vector table address offsets.

5.4.1 Interrupt Priority Level

The interrupt priority order is predetermined. When multiple interrupts occur simultaneously (overlap), the interrupt controller (INTC) determines their relative priorities and starts up processing according to the results.

The priority order of interrupts is expressed as priority levels 0 to 16, with priority 0 the lowest and priority 16 the highest. The NMI interrupt has priority 16 and cannot be masked, so it is always accepted. The user break interrupt priority level is 15. IRQ interrupts and on-chip peripheral module interrupt priority levels can be set freely using the INTC's interrupt priority level setting registers A through H (IPRA to IPRH) as shown in table 5.8. The priority levels that can be set are 0 to 15. Level 16 cannot be set. See section 6.3.1, Interrupt Priority Registers A to H (IPRA to IPRH), for more information on IPRA to IPRH.

Table 5.8 Interrupt Priority Order

Type	Priority Level	Comment
NMI	16	Fixed priority level. Cannot be masked.
IRQ	0 to 15	Set with interrupt priority level setting registers A through H (IPRA to IPRH).
On-chip peripheral module	0 to 15	Set with interrupt priority level setting registers A through H (IPRA to IPRH).

5.4.2 Interrupt Exception Processing

When an interrupt occurs, its priority level is ascertained by the interrupt controller (INTC). NMI is always accepted, but other interrupts are only accepted if they have a priority level higher than the priority level set in the interrupt mask bits (I3 to I0) of the status register (SR).

When an interrupt is accepted, exception processing begins. In interrupt exception processing, the CPU saves SR and the program counter (PC) to the stack. The priority level value of the accepted interrupt is written to SR bits I3 to I0. For NMI, however, the priority level is 16, but the value set in I3 to I0 is H'F (level 15). Next, the start address of the exception service routine is fetched from the exception processing vector table for the accepted interrupt, that address is jumped to and execution begins. See section 6.4, Interrupt Operation, for more information on the interrupt exception processing.

5.5 Exceptions Triggered by Instructions

Exception processing can be triggered by trap instructions, general illegal instructions, and illegal slot instructions, as shown in table 5.9.

Table 5.9 Types of Exceptions Triggered by Instructions

Type	Source Instruction	Comment
Trap instructions	TRAPA	—
Illegal slot instructions	Undefined code placed immediately after a delayed branch instruction (delay slot) and instructions that rewrite the PC	Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE, BF/S, BT/S, BSRF, BRAF Instructions that rewrite the PC: JMP, JSR, BRA, BSR, RTS, RTE, BT, BF, TRAPA, BF/S, BT/S, BSRF, BRAF
General illegal instructions	Undefined code anywhere besides in a delay slot	—

5.5.1 Trap Instructions

When a TRAPA instruction is executed, trap instruction exception processing starts up. The CPU operates as follows:

1. The status register (SR) is saved to the stack.
2. The program counter (PC) is saved to the stack. The PC value saved is the start address of the instruction to be executed after the TRAPA instruction.
3. The exception service routine start address is fetched from the exception processing vector table that corresponds to the vector number specified in the TRAPA instruction. That address is jumped to and the program starts executing. The jump that occurs is not a delayed branch.

5.5.2 Illegal Slot Instructions

An instruction placed immediately after a delayed branch instruction is said to be placed in a delay slot. When the instruction placed in the delay slot is undefined code, illegal slot exception processing starts up when that undefined code is decoded. Illegal slot exception processing also starts up when an instruction that rewrites the program counter (PC) is placed in a delay slot. The processing starts when the instruction is decoded. The CPU handles an illegal slot instruction as follows:

1. The status register (SR) is saved to the stack.
2. The program counter (PC) is saved to the stack. The PC value saved is the jump address of the delayed branch instruction immediately before the undefined code or the instruction that rewrites the PC.
3. The exception service routine start address is fetched from the exception processing vector table that corresponds to the exception that occurred. That address is jumped to and the program starts executing. The jump that occurs is not a delayed branch.

5.5.3 General Illegal Instructions

When undefined code placed anywhere other than immediately after a delayed branch instruction (i.e., in a delay slot) is decoded, general illegal instruction exception processing starts up. The CPU handles general illegal instructions the same as illegal slot instructions. Unlike processing of illegal slot instructions, however, the program counter value stored is the start address of the undefined code.

5.6 When Exception Sources Are Not Accepted

When an address error or interrupt is generated after a delayed branch instruction or interrupt-disabled instruction, it is sometimes not accepted immediately but stored instead, as shown in table 5.10. When this happens, it will be accepted when an instruction that can accept the exception is decoded.

Table 5.10 Generation of Exception Sources Immediately after a Delayed Branch Instruction or Interrupt-Disabled Instruction

	Exception Source	
Point of Occurrence	**Address Error**	**Interrupt**
Immediately after a delayed branch instruction*1	Not accepted	Not accepted
Immediately after an interrupt-disabled instruction*2	Accepted	Not accepted

Notes: 1. Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE, BF/S, BT/S, BSRF, BRAF
2. Interrupt-disabled instructions: LDC, LDC.L, STC, STC.L, LDS, LDS.L, STS, STS.L

5.6.1 Immediately after a Delayed Branch Instruction

When an instruction placed immediately after a delayed branch instruction (delay slot) is decoded, neither address errors nor interrupts are accepted. The delayed branch instruction and the instruction located immediately after it (delay slot) are always executed consecutively, so no exception processing occurs during this period.

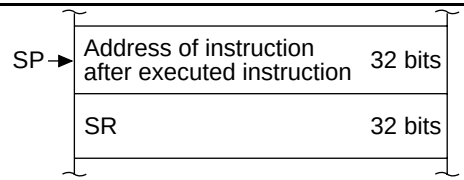
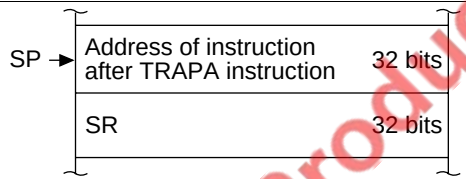
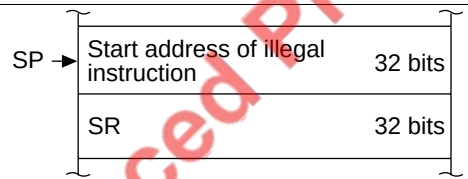
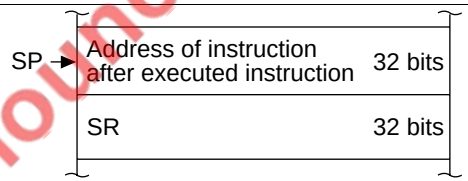
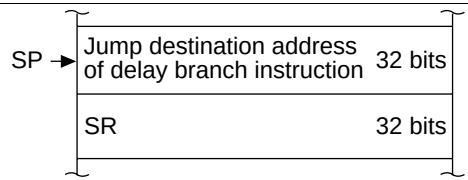
5.6.2 Immediately after an Interrupt-Disabled Instruction

When an instruction immediately following an interrupt-disabled instruction is decoded, interrupts are not accepted. Address errors are accepted.

5.7 Stack Status after Exception Processing Ends

The status of the stack after exception processing ends is as shown in table 5.11.

Table 5.11 Types of Stack Status After Exception Processing Ends

Types	Stack Status
Address error	
Trap instruction	
General illegal instruction	
Interrupt	
Illegal slot instruction	

5.8 Notes on Use

5.8.1 Value of Stack Pointer (SP)

The value of the stack pointer must always be a multiple of four. If it is not, an address error will occur when the stack is accessed during exception processing.

5.8.2 Value of Vector Base Register (VBR)

The value of the vector base register must always be a multiple of four. If it is not, an address error will occur when the stack is accessed during exception processing.

5.8.3 Address Errors Caused by Stacking of Address Error Exception Processing

When the stack pointer is not a multiple of four, an address error will occur during stacking of the exception processing (interrupts, etc.) and address error exception processing will start up as soon as the first exception processing is ended. Address errors will then also occur in the stacking for this address error exception processing. To ensure that address error exception processing does not go into an endless loop, no address errors are accepted at that point. This allows program control to be shifted to the address error exception service routine and enables error processing.

When an address error occurs during exception processing stacking, the stacking bus cycle (write) is executed. During stacking of the status register (SR) and program counter (PC), the SP is –4 for both, so the value of SP will not be a multiple of four after the stacking either. The address value output during stacking is the SP value, so the address where the error occurred is itself output. This means the write data stacked will be undefined.

Section 6 Interrupt Controller (INTC)

6.1 Overview

The interrupt controller (INTC) ascertains the priority of interrupt sources and controls interrupt requests to the CPU. The INTC has registers for setting the priority of each interrupt which can be used by the user to order the priorities in which the interrupt requests are processed.

6.1.1 Features

The INTC has the following features:

- 16 levels of interrupt priority: By setting the eight interrupt-priority level registers, the priorities of IRQ interrupts and on-chip peripheral module interrupts can be set in 16 levels for different request sources.
- NMI noise canceler function: NMI input level bits indicate the NMI pin status. By reading these bits with the interrupt exception service routine, the pin status can be confirmed, enabling it to be used as a noise canceler.

6.1.2 Block Diagram

Figure 6.1 is a block diagram of the INTC.

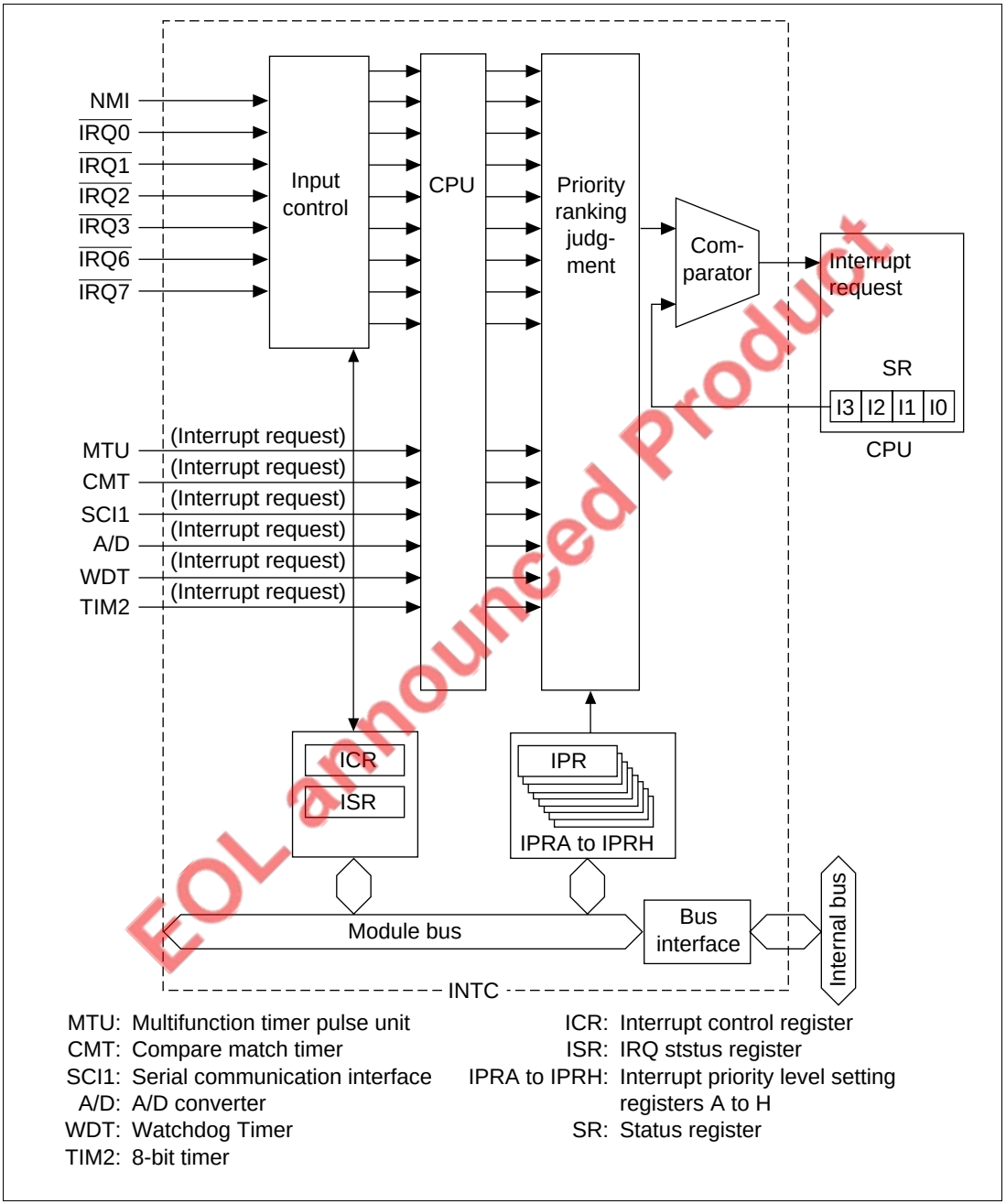


Figure 6.1 INTC Block Diagram

6.1.3 Pin Configuration

Table 6.1 shows the INTC pin configuration.

Table 6.1 Pin Configuration

Name	Abbreviation	I/O	Function
Non-maskable interrupt input pin	NMI	I	Input of non-maskable interrupt request signal
Interrupt request input pins	$\overline{\text{IRQ0}}$ to $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$	I	Input of maskable interrupt request signals

6.1.4 Register Configuration

The INTC has the 10 registers shown in table 6.2. These registers set the priority of the interrupts and control external interrupt input signal detection.

Table 6.2 Register Configuration

Name	Abbr.	R/W	Initial Value	Address	Access Sizes
Interrupt priority register A	IPRA	R/W	H'0000	H'FFFF8348	8, 16, 32
Interrupt priority register B	IPRB	R/W	H'0000	H'FFFF834A	8, 16, 32
Interrupt priority register C	IPRC	R/W	H'0000	H'FFFF834C	8, 16, 32
Interrupt priority register D	IPRD	R/W	H'0000	H'FFFF834E	8, 16, 32
Interrupt priority register E	IPRE	R/W	H'0000	H'FFFF8350	8, 16, 32
Interrupt priority register F	IPRF	R/W	H'0000	H'FFFF8352	8, 16, 32
Interrupt priority register G	IPRG	R/W	H'0000	H'FFFF8354	8, 16, 32
Interrupt priority register H	IPRH	R/W	H'0000	H'FFFF8356	8, 16, 32
Interrupt control register	ICR	R/W	*1	H'FFFF8358	8, 16, 32
IRQ status register	ISR	R(W)*2	H'0000	H'FFFF835A	8, 16, 32

Notes: 1. The value when the NMI pin is high is H'8000; when the NMI pin is low, it is H'0000.
2. Only 0 can be written, in order to clear flags.

6.2 Interrupt Sources

There are three types of interrupt sources: NMI, IRQ, and on-chip peripheral modules. Each interrupt has a priority expressed as a priority level (0 to 16, with 0 the lowest and 16 the highest). Giving an interrupt a priority level of 0 masks it.

6.2.1 NMI Interrupts

The NMI interrupt has priority 16 and is always accepted. Input at the NMI pin is detected by edge. Use the NMI edge select bit (NMIE) in the interrupt control register (ICR) to select either the rising or falling edge. NMI interrupt exception processing sets the interrupt mask level bits (I3 to I0) in the status register (SR) to level 15.

6.2.2 IRQ Interrupts

IRQ interrupts are requested by input from pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ7}}$. Set the IRQ sense select bits (IRQ0S to IRQ3S, IRQ6S, and IRQ7S) of the interrupt control register (ICR) to select low level detection or falling edge detection for each pin. The priority level can be set from 0 to 15 for each pin using the interrupt priority registers A and B (IPRA, IPRB).

When IRQ interrupts are set to low level detection, an interrupt request signal is sent to the INTC during the period the IRQ pin is low level. Interrupt request signals are not sent to the INTC when the IRQ pin becomes high level. Interrupt request levels can be confirmed by reading the IRQ flags (IRQ0F to IRQ3F, IRQ6F, and IRQ7F) of the IRQ status register (ISR).

When IRQ interrupts are set to falling edge detection, interrupt request signals are sent to the INTC upon detecting a change on the IRQ pin from high to low level. IRQ interrupt request detection results are maintained until the interrupt request is accepted. Confirmation that IRQ interrupt requests have been detected is possible by reading the IRQ flags (IRQ0F to IRQ3F, IRQ6F, and IRQ7F) of the IRQ status register (ISR), and by writing a 0 after reading a 1, IRQ interrupt request detection results can be withdrawn.

In IRQ interrupt exception processing, the interrupt mask bits (I3 to I0) of the status register (SR) are set to the priority level value of the accepted IRQ interrupt.

6.2.3 On-Chip Peripheral Module Interrupts

On-chip peripheral module interrupts are interrupts generated by the following on-chip peripheral modules:

- Multifunction timer pulse unit (MTU)
- Compare match timer (CMT)
- Serial communications interface (SCI1)
- A/D converter (A/D)
- Watchdog timer (WDT)
- 8-bit timer 2 (TIM2)

A different interrupt vector is assigned to each interrupt source, so the exception service routine does not have to decide which interrupt has occurred. Priority levels between 0 and 15 can be assigned to individual on-chip peripheral modules in interrupt priority registers C to H (IPRC to IPRH).

On-chip peripheral module interrupt exception processing sets the interrupt mask level bits (I3 to I0) in the status register (SR) to the priority level value of the on-chip peripheral module interrupt that was accepted.

6.2.4 Interrupt Exception Vectors and Priority Rankings

Table 6.3 lists interrupt sources and their vector numbers, vector table address offsets and interrupt priorities.

Each interrupt source is allocated a different vector number and vector table address offset. Vector table addresses are calculated from vector numbers and address offsets. In interrupt exception processing, the exception service routine start address is fetched from the vector table indicated by the vector table address. See section 5 Exception Processing, table 5.4, Calculating Exception Processing Vector Table Addresses.

IRQ interrupts and on-chip peripheral module interrupt priorities can be set freely between 0 and 15 for each pin or module by setting interrupt priority registers A to H (IPRA to IPRH). The ranking of interrupt sources for IPRC to IPRH, however, must be the order listed under Priority Order Within IPR Setting Range in table 6.3 and cannot be changed. A power-on reset assigns priority level 0 to IRQ interrupts and on-chip peripheral module interrupts. If the same priority level is assigned to two or more interrupt sources and interrupts from those sources occur simultaneously, their priority order is the default priority order indicated at the right in table 6.3.

Table 6.3 Interrupt Exception Processing Vectors and Priorities

Interrupt Source		Interrupt Vector		Interrupt Priority (Initial Value)	Corre-sponding IPR (Bits)	Priority within IPR Setting Range	Default Priority
		Vector No.	Vector Table Address Offset				
NMI		11	H'0000002C to H'0000002F	16	—	—	High
IRQ0		64	H'00000100 to H'00000103	0 to 15 (0)	IPRA (15 to 12)	—	
IRQ1		65	H'00000104 to H'00000107	0 to 15 (0)	IPRA (11 to 8)	—	
IRQ2		66	H'00000108 to H'0000010B	0 to 15 (0)	IPRA (7 to 4)	—	
IRQ3		67	H'0000010C to H'0000010F	0 to 15 (0)	IPRA (3 to 0)	—	
IRQ6		70	H'00000118 to H'0000011B	0 to 15 (0)	IPRB (7 to 4)	—	
IRQ7		71	H'0000011C to H'0000011F	0 to 15 (0)	IPRB (3 to 0)	—	
MTU0	TGI0A	88	H'00000160 to H'00000163	0 to 15 (0)	IPRD (15 to 12)	High	
	TGI0B	89	H'00000164 to H'00000167	0 to 15 (0)			
	TGI0C	90	H'00000168 to H'0000016B	0 to 15 (0)			
	TGI0D	91	H'0000016C to H'0000016F	0 to 15 (0)		Low	
	TCI0V	92	H'00000170 to H'00000173	0 to 15 (0)	IPRD (11 to 8)	—	
MTU1	TGI1A	96	H'00000180 to H'00000183	0 to 15 (0)	IPRD (7 to 4)	High	
	TGI1B	97	H'00000184 to H'00000187	0 to 15 (0)		Low	
	TCI1V	100	H'00000190 to H'00000193	0 to 15 (0)	IPRD (3 to 0)	—	Low

Table 6.3 Interrupt Exception Processing Vectors and Priorities (cont)

Interrupt Source		Interrupt Vector		Interrupt Priority (Initial Value)	Corre-sponding IPR (Bits)	Priority within IPR Setting Range	Default Priority
		Vector No.	Vector Table Address Offset				
MTU2	TGI2A	104	H'000001A0 to H'000001A3	0 to 15 (0)	IPRE (15 to 12)	High	High
	TGI2B	105	H'000001A4 to H'000001A7	0 to 15 (0)		Low	
	TCI2V	108	H'000001B0 to H'000001B3	0 to 15 (0)	IPRE (11 to 8)	—	
SCI1	ERI1	132	H'00000210 to H'00000213	0 to 15 (0)	IPRF (3 to 0)	High	
	RXI1	133	H'00000214 to H'00000217				
	TXI1	134	H'00000218 to H'0000021B				
	TEI1	135	H'0000021C to H'0000021F			Low	
A/D	ADI	138	H'00000228 to H'0000022B	0 to 15 (0)	IPRG (15 to 12)	—	
CMT0	CMI0	144	H'00000240 to H'00000243	0 to 15 (0)	IPRG (7 to 4)	—	
CMT1	CMI1	148	H'00000250 to H'00000253	0 to 15 (0)	IPRG (3 to 0)	—	
WDT	ITI	152	H'00000260 to H'00000263	0 to 15 (0)	IPRH (15 to 2)	High	
TIM2	CMI	153	H'00000264 to H'00000267			Low	Low

6.3 Description of Registers

6.3.1 Interrupt Priority Registers A to H (IPRA to IPRH)

Interrupt priority registers A to H (IPRA to IPRH) are 16-bit readable/writable registers that set priority levels from 0 to 15 for IRQ interrupts and on-chip peripheral module interrupts. Correspondence between interrupt request sources and each of the IPRA to IPRH bits is shown in table 6.4.

Bit:	15	14	13	12	11	10	9	8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6.4 Interrupt Request Sources and IPRA to IPRH

	Bits			
Register	**15 to 12**	**11 to 8**	**7 to 4**	**3 to 0**
Interrupt priority register A	IRQ0	IRQ1	IRQ2	IRQ3
Interrupt priority register B	Reserved	Reserved	IRQ6	IRQ7
Interrupt priority register C	Reserved	Reserved	Reserved	Reserved
Interrupt priority register D	MTU0	MTU0	MTU1	MTU1
Interrupt priority register E	MTU2	MTTU2	Reserved	Reserved
Interrupt priority register F	Reserved	Reserved	Reserved	SCI1
Interrupt priority register G	A/D	Reserved	CMT0	CMT1
Interrupt priority register H	WDT, TIM2	Reserved	Reserved	Reserved

As indicated in table 6.4, four $\overline{\text{IRQ}}$ pins or groups of 4 on-chip peripheral modules are allocated to each register. Each of the corresponding interrupt priority ranks are established by setting a value from H'0 (0000) to H'F (1111) in each of the four-bit groups 15 to 12, 11 to 8, 7 to 4 and 3 to 0. Interrupt priority rank becomes level 0 (lowest) by setting H'0, and level 15 (highest) by setting H'F. 8-bit timers 1 and 2 are set to the same priority rank.

IPRA to IPRH are initialized to H'0000 by a power-on reset. Reserved bits are always read as 0. The write value should always be 0. It is not initialized in the standby mode.

6.3.2 Interrupt Control Register (ICR)

The ICR is a 16-bit register that sets the input signal detection mode of the external interrupt input pin NMI and $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$ and indicates the input signal level to the NMI pin.
A power-on reset initializes ICR. It is not initialized in the standby mode.

Bit:	15	14	13	12	11	10	9	8
	NMIL	—	—	—	—	—	—	NMIE
Initial value:	*	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

Bit:	7	6	5	4	3	2	1	0
	IRQ0S	IRQ1S	IRQ2S	IRQ3S	—	—	IRQ6S	IRQ7S
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

Note: * When NMI input is high: 1; when NMI input is low: 0

- Bit 15—NMI Input Level (NMIL): Sets the level of the signal input at the NMI pin. This bit can be read to determine the NMI pin level. This bit cannot be modified.

Bit 15: NMIL	Description
0	NMI input level is low
1	NMI input level is high

- Bits 14 to 9—Reserved: These bits are always read as 0. The write value should always be 0.
- Bit 8—NMI Edge Select (NMIE)

Bit 8: NMIE	Description
0	Interrupt request is detected on falling edge of NMI input (Initial value)
1	Interrupt request is detected on rising edge of NMI input

- Bits 7 to 4, 1, and 0—IRQ0 to IRQ3, IRQ6, and IRQ7 Sense Select (IRQ0S to IRQ3S, IRQ6S, IRQ7S): These bits set the IRQ0 to IRQ7 interrupt request detection mode.

Bits 7 to 4, 1, 0: IRQ0S to IRQ3S, IRQ6S, IRQ7S	Description	
0	Interrupt request is detected on low level of IRQ input	(Initial value)
1	Interrupt request is detected on falling edge of IRQ input	

- Bits 3 and 2—Reserved: These bits always read as 0. The write value should always be 0.

6.3.3 IRQ Status Register (ISR)

The ISR is a 16-bit register that indicates the interrupt request status of the external interrupt input pins $\overline{IRQ0}$ to $\overline{IRQ3}$, $\overline{IRQ6}$, and $\overline{IRQ7}$. When IRQ interrupts are set to edge detection, held interrupt requests can be withdrawn by writing a 0 to IRQnF after reading an IRQnF = 1.

A power-on reset initializes ISR. It is not initialized in the standby mode.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	IRQ0F	IRQ1F	IRQ2F	IRQ3F	—	—	IRQ6F	IRQ7F
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

- Bits 15 to 8, 3, and 2—Reserved: These bits are always read as 0. The write value should always be 0.

- Bits 7 to 4, 1, and 0—IRQ0 to IRQ3, IRQ6, and IRQ7 Flags (IRQ0F to IRQ3F, IRQ6F, IRQ7F): These bits display the IRQ0 to IRQ3, IRQ6, IRQ7 interrupt request status.

Bits 7 to 4, 1, 0: IRQ0F to IRQ3F, IRQ6F, IRQ7F	Detection Setting	Description
0	Level detection	No IRQn interrupt request exists. Clear conditions: When $\overline{\text{IRQn}}$ input is high level
	Edge detection	No IRQn interrupt request was detected. (Initial value) Clear conditions: 1. When a 0 is written after reading IRQnF = 1 status 2. When IRQn interrupt exception processing has been executed
1	Level detection	An IRQn interrupt request exists. Set conditions: When $\overline{\text{IRQn}}$ input is low level
	Edge detection	An IRQn interrupt request was detected. Set conditions: When a falling edge occurs at an $\overline{\text{IRQn}}$ input

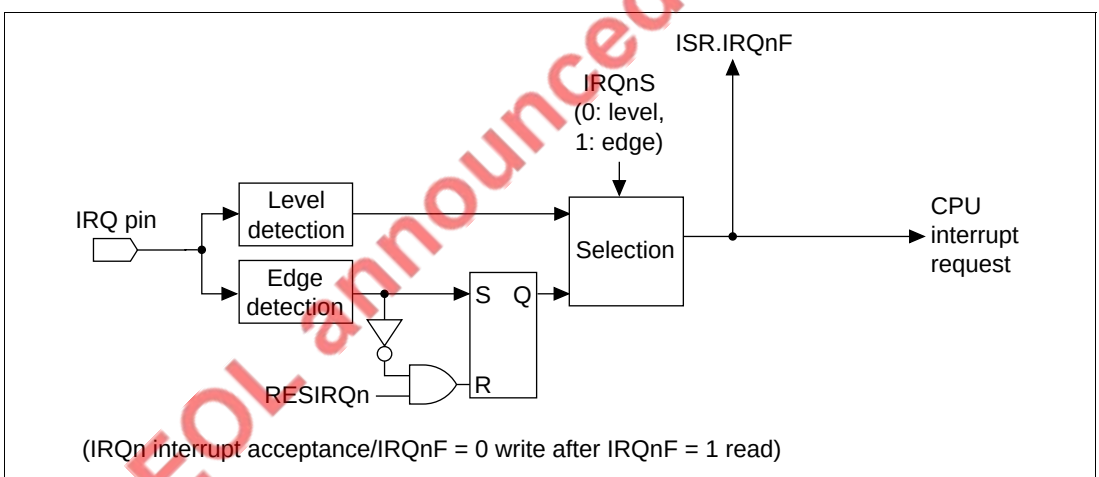


Figure 6.2 External Interrupt Process

6.4 Interrupt Operation

6.4.1 Interrupt Sequence

The sequence of interrupt operations is explained below. Figure 6.3 is a flowchart of the operations.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest priority interrupt in the interrupt requests sent, following the priority levels set in interrupt priority level setting registers A to H (IPRA to IPRH). Lower-priority interrupts are ignored*. If a number of interrupts with the same priority level occur, or if multiple interrupts occur within a single module, the interrupt with the highest default priority or the highest priority within its IPR setting range (as indicated in table 6.3) is selected.
3. The interrupt controller compares the priority level of the selected interrupt request with the interrupt mask bits (I3 to I0) in the CPU's status register (SR). If the request priority level is equal to or less than the level set in I3 to I0, the request is ignored. If the request priority level is higher than the level in bits I3 to I0, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. The interrupt controller detects the interrupt request sent from the interrupt controller when it decodes the next instruction to be executed. Instead of executing the decoded instruction, the CPU starts interrupt exception processing (figure 6.5).
5. The status register (SR) and program counter (PC) are saved onto the stack.
6. The priority level of the accepted interrupt is written to bits I3 to I0 in SR.
7. The CPU reads the start address of the exception service routine from the exception vector table for the accepted interrupt, jumps to that address, and starts executing the program there. This jump is not a delay branch.

Note: An interrupt request for which edge detection has been set is held pending until it is accepted. However, an IRQ interrupt can be cleared by an IRQ status register (ISR) access. For details see section 6.2.2, IRQ Interrupts.
Pending edge-detected interrupts are cleared by a power-on reset.

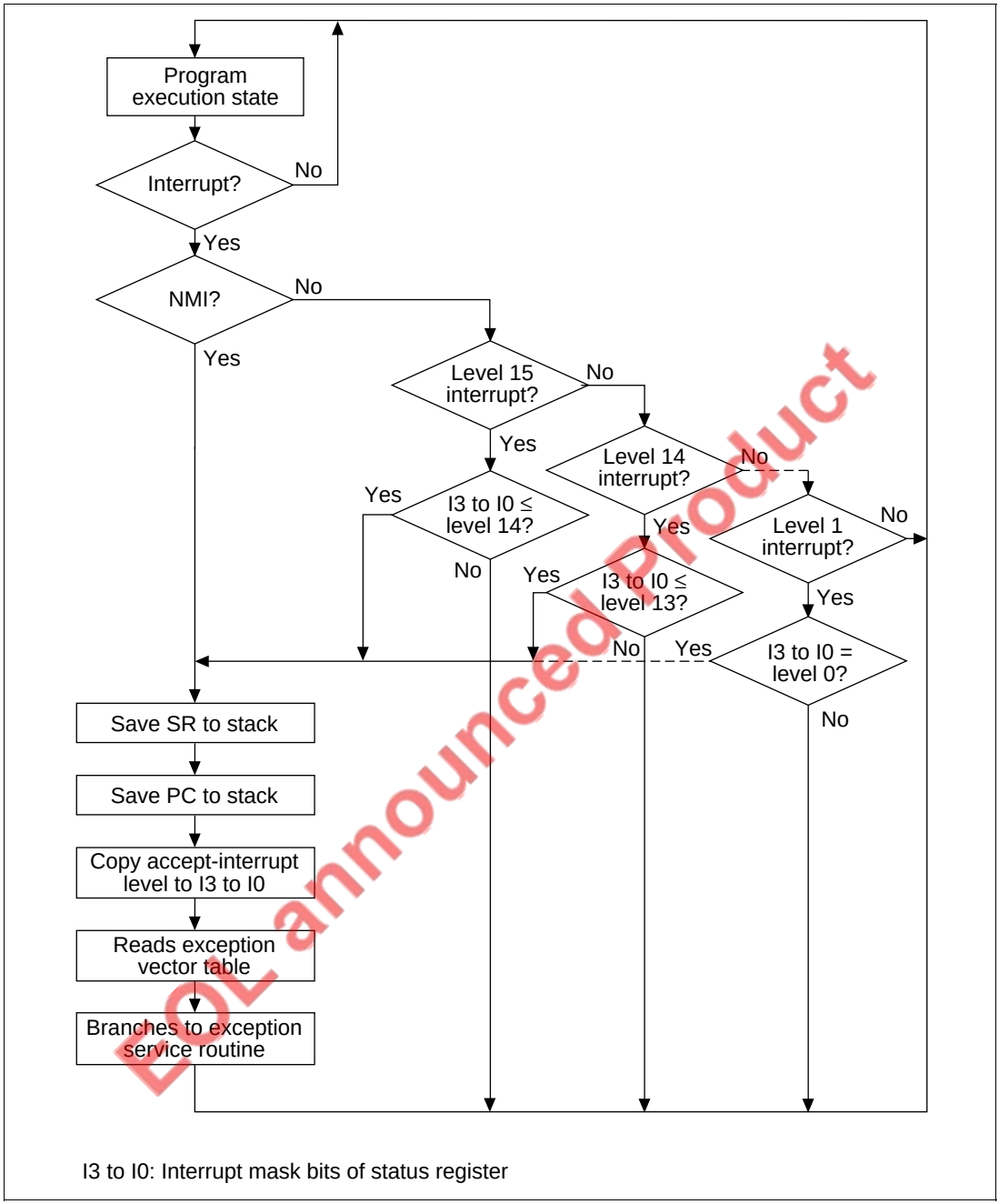
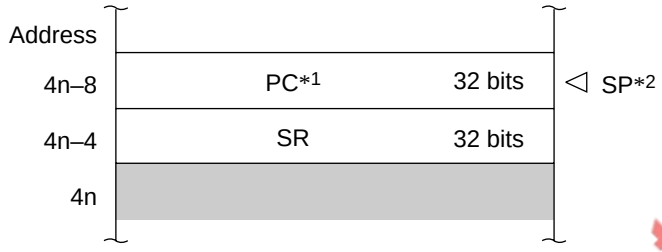


Figure 6.3 Interrupt Sequence Flowchart

6.4.2 Stack after Interrupt Exception Processing

Figure 6.4 shows the stack after interrupt exception processing.



Notes: 1. PC: Start address of the next instruction (return destination instruction) after the executing instruction
2. Always be certain that SP is a multiple of 4

Figure 6.4 Stack after Interrupt Exception Processing

6.5 Interrupt Response Time

Table 6.5 indicates the interrupt response time, which is the time from the occurrence of an interrupt request until the interrupt exception processing starts and fetching of the first instruction of the interrupt service routine begins. Figure 6.5 shows the pipeline when an IRQ interrupt is accepted.

Table 6.5 Interrupt Response Time

Item		Number of States: NMI, Peripheral Module	Number of States: IRQ	Notes
Compare identified interrupt priority with SR mask level		2	3	
Wait for completion of sequence currently being executed by CPU		X (≥ 0)		The longest sequence is for interrupt or address-error exception processing (X = 4 + m1 + m2 + m3 + m4). If an interrupt-masking instruction follows, however, the time may be even longer.
Time from start of interrupt exception processing until fetch of first instruction of exception service routine starts		5 + m1 + m2 + m3		Performs the PC and SR saves and vector address fetch.
Interrupt response time	Total	7 + m1 + m2 + m3	9 + m1 + m2 + m3	
	Minimum	10	12	20 MHz operation: 0.5 to 0.6 µs
	Maximum	12 + 2 (m1 + m2 + m3) + m4	13 + 2 (m1 + m2 + m3) + m4	20 MHz operation: 0.95 to 1.0 µs*

Note: * When m1 = m2 = m3 = m4 = 1

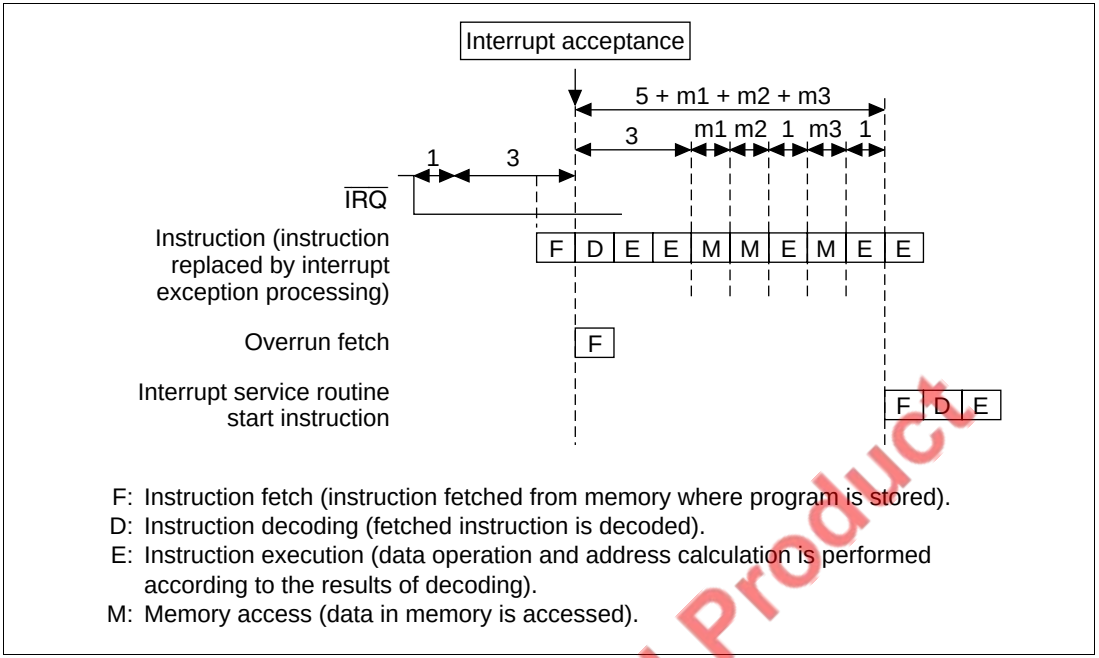
m1 to m4 are the number of states needed for the following memory accesses.

m1: SR save (longword write)

m2: PC save (longword write)

m3: Vector address read (longword read)

m4: Fetch first instruction of interrupt service routine



F: Instruction fetch (instruction fetched from memory where program is stored).
D: Instruction decoding (fetched instruction is decoded).
E: Instruction execution (data operation and address calculation is performed according to the results of decoding).
M: Memory access (data in memory is accessed).

Figure 6.5 Pipeline when an IRQ Interrupt is Accepted

Section 7 Bus State Controller (BSC)

7.1 Overview

The bus state controller (BSC) divides up the address spaces and outputs control for various types of memory. This enables memories like SRAM and ROM to be linked directly to the LSI without external circuitry.

7.1.1 Features

- Address space is divided into four spaces
  A maximum linear 2 Mbytes for address space CS0
  — A maximum linear 4 Mbytes for each of address spaces CS1 to CS3
  — 8-bit bus width
  — Wait states can be inserted by software for each space (0 to 3 waits)
  — In external memory space access, wait states can be inserted by the $\overline{\text{WAIT}}$ pin
  — Outputs control signals for each space according to the type of memory connected
- RAM interface
  — On-chip RAM access of 32 bits in 1 state

7.1.2 Block Diagram

Figure 7.1 shows the BSC block diagram.

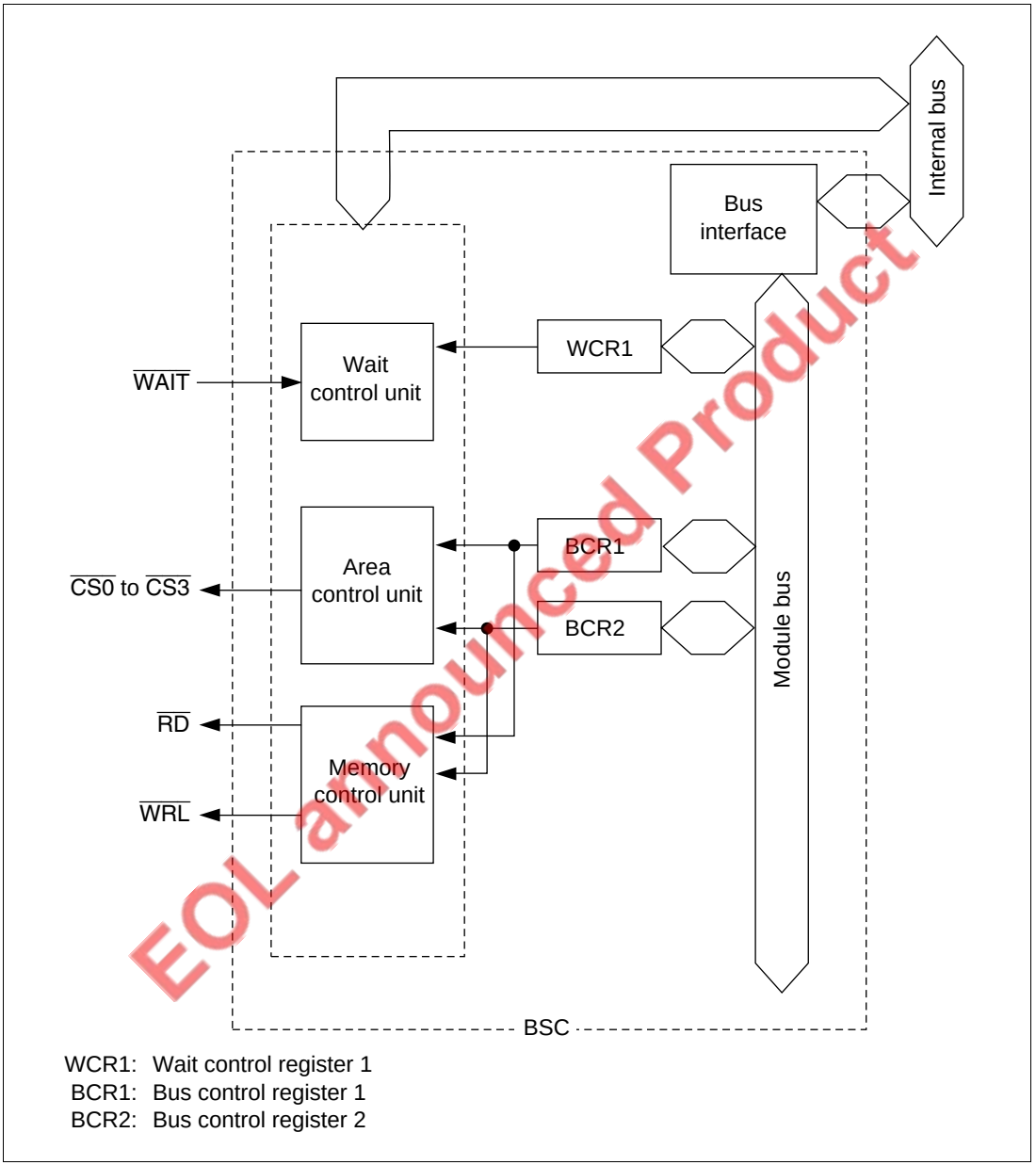


Figure 7.1 BSC Block Diagram

7.1.3 Pin Configuration

Table 7.1 shows the bus state controller pin configuration.

Table 7.1 Pin Configuration

Pin Name	I/O	Function
A21 to A0	Output	Address output
D7 to D0	I/O	8-bit data bus
$\overline{CS0}$ to $\overline{CS3}$	Output	Chip select
$\overline{RD}$	Output	Strobe that indicates a read cycle for ordinary space/multiplex I/O
$\overline{WRL}$	Output	Strobe that indicates a write cycle
$\overline{WAIT}$	Input	Wait state request signal

7.1.4 Register Configuration

The bus state controller has three registers. The functions of these registers include control of wait states and interfaces with memories such as ROM and SRAM. The registers are summarized in table 7.2.

Both registers are 16 bits in size, and are initialized by a power-on reset.

Table 7.2 Register Configuration

Name	Abbr.	R/W	Initial Value	Address	Access Size
Bus control register 1	BCR1	R/W	H'200F	H'FFFF8620	8, 16
Bus control register 2	BCR2	R/W	H'FFFF	H'FFFF8622	8, 16
Wait state control register 1	WCR1	R/W	H'FFFF	H'FFFF8624	8, 16

7.1.5 Address Map

Figure 7.2 shows the address format used by this LSI.

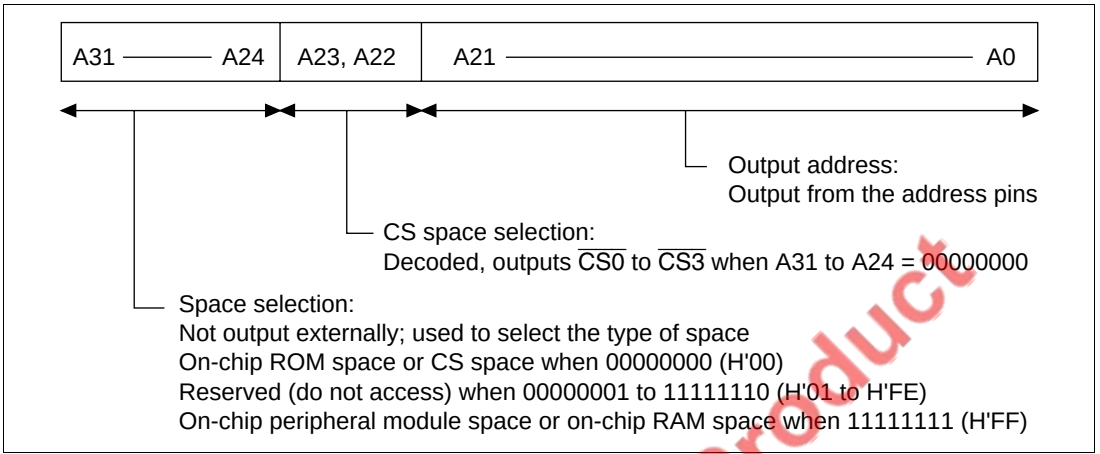


Figure 7.2 Address Format

This LSI uses 32-bit addresses:

- A31 to A24 are used to select the type of space and are not output externally.
- Bits A23 and A22 are decoded and output as chip select signals ($\overline{CS0}$ to $\overline{CS3}$) for the corresponding areas when bits A31 to A24 are 00000000.
- A21 to A0 are output externally.

Table 7.3 shows an address map.

Table 7.3 Address Map

Address	Space	Memory	Size	Bus Width
H'00000000 to H'00027FFF	On-chip ROM	On-chip ROM	160 kbytes	32 bits
H'00028000 to H'001FFFFF	Reserved	Reserved		
H'00200000 to H'003FFFFF	CS0 space	Ordinary space	2 Mbytes	8 bits
H'00400000 to H'007FFFFF	CS1 space	Ordinary space	4 Mbytes	8 bits
H'00800000 to H'00BFFFFF	CS2 space	Ordinary space	4 Mbytes	8 bits
H'00C00000 to H'00FFFFFF	CS3 space	Ordinary space	4 Mbytes	8 bits
H'01000000 to H'FFFF7FFF	Reserved	Reserved		
H'FFFF8000 to H'FFFF87FF	On-chip peripheral module	On-chip peripheral module	2 kbytes	8/16 bits
H'FFFF8800 to H'FFFFEFFF	Reserved	Reserved		
H'FFFFF000 to H'FFFFFFFF	On-chip RAM	On-chip RAM	4 kbytes	32 bits

Note: Do not access reserved spaces. Operation cannot be guaranteed if they are accessed.

7.2 Description of Registers

7.2.1 Bus Control Register 1 (BCR1)

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	1	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	A3SZ	A2SZ	A1SZ	A0SZ
Initial value:	0	0	0	0	1	1	1	1
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

Note: Never write 1 to bits 4 to 7; doing so can result in unstable operation.

Bus control register 1 (BCR1) is a 16-bit readable/writeable register that specifies the bus size for each CS space. Note that this chip requires that byte bus size (8 bits).

Initial settings should be written to bits 8 to 0 of BCR1 following a power-on reset, and the values should then be left unchanged. Also, the CS spaces should not be accessed until initial setting of the registers has been completed.

BCR1 is initialized to H'200F by a power-on reset. It is not initialized in the standby mode.

- Bits 15, 14, and 12 to 4—Reserved: These bits are always read as 0. The write value should always be 0.
- Bit 13—Reserved: This bit is always read as 1. The write value should always be 1.
- Bit 3—CS3 Space Size Specification (A3SZ): Specifies the bus size for CS3. The SH7018 requires a byte bus size, so this bit must always be set to 0.

Bit 3: A3SZ	Description	
0	Byte (8-bit) size	
1	Word (16-bit) size	(Initial value)

- Bit 2—CS2 Space Size Specification (A2SZ): Specifies the bus size for CS2. The SH7018 requires a byte bus size, so this bit must always be set to 0.

Bit 2: A2SZ	Description	
0	Byte (8-bit) size	
1	Word (16-bit) size	(Initial value)

- Bit 1—CS1 Space Size Specification (A1SZ): Specifies the bus size for CS1. The SH7018 requires a byte bus size, so this bit must always be set to 0.

Bit 1: A1SZ	Description	
0	Byte (8-bit) size	
1	Word (16-bit) size	(Initial value)

- Bit 0—CS0 Space Size Specification (A0SZ): Specifies the bus size for CS0. The SH7018 requires a byte bus size, so this bit must always be set to 0.

Bit 0: A0SZ	Description	
0	Byte (8-bit) size	
1	Word (16-bit) size	(Initial value)

7.2.2 Bus Control Register 2 (BCR2)

BCR2 is a 16-bit read/write register that specifies the number of idle cycles and $\overline{CS}$ signal assert extension of each CS space.

BCR2 is initialized by power-on resets to H'FFFF.

Bit:	15	14	13	12	11	10	9	8
	IW31	IW30	IW21	IW20	IW11	IW10	IW01	IW00
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	CW3	CW2	CW1	CW0	SW3	SW2	SW1	SW0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- Bits 15 to 8—Idles between Cycles (IW31, IW30, IW21, IW20, IW11, IW10, IW01, IW00): These bits specify idle cycles inserted between consecutive accesses when the second one is to a different CS area after a read. Idles are used to prevent data conflict between ROM (and other memories, which are slow to turn the read data buffer off), fast memories, and I/O interfaces. Even when access is to the same area, idle cycles must be inserted when a read access is followed immediately by a write access. The idle cycles to be inserted comply with the area specification of the previous access. Refer to section 7.4, Waits between Access Cycles, for details.

  IW31 and IW30 specify the idle between cycles for the CS3 space; IW21 and IW20 specify the idle between cycles for the CS2 space; IW11 and IW10 specify the idle between cycles for the CS1 space, and IW01 and IW00 specify the idle between cycles for the CS0 space.

Bit 15 (IW31)	Bit 14 (IW30)	Description
0	0	No idle cycle after accessing CS3 space
	1	Inserts one idle cycle after accessing CS3 space
1	0	Inserts two idle cycles after accessing CS3 space
	1	Inserts three idle cycles after accessing CS3 space (Initial value)

Bit 13 (IW21)	Bit 12 (IW20)	Description
0	0	No idle cycle after accessing CS2 space
	1	Inserts one idle cycle after accessing CS2 space
1	0	Inserts two idle cycles after accessing CS2 space
	1	Inserts three idle cycles after accessing CS2 space (Initial value)

Bit 11 (IW11)	Bit 10 (IW10)	Description
0	0	No idle cycle after accessing CS1 space
	1	Inserts one idle cycle after accessing CS1 space
1	0	Inserts two idle cycles after accessing CS1 space
	1	Inserts three idle cycles after accessing CS1 space (Initial value)

Bit 9 (IW01)	Bit 8 (IW00)	Description
0	0	No idle cycle after accessing CS0 space
	1	Inserts one idle cycle after accessing CS0 space
1	0	Inserts two idle cycles after accessing CS0 space
	1	Inserts three idle cycles after accessing CS0 space (Initial value)

- Bits 7 to 4—Idle Specification for Continuous Access (CW3, CW2, CW1, CW0): The continuous access idle specification makes insertions to clearly delineate the bus intervals by once negating the $\overline{CSn}$ signal when doing consecutive accesses of the same CS space. When a write immediately follows a read, the number of idle cycles inserted is the larger of the two values specified by IW and CW. Refer to section 7.4, Waits between Access Cycles, for details.

  CW3 specifies the continuous access idles for the CS3 space; CW2 specifies the continuous access idles for the CS2 space; CW1 specifies the continuous access idles for the CS1 space and CW0 specifies the continuous access idles for the CS0 space.

Bit 7 (CW3)	Description	
0	No CS3 space continuous access idle cycles	
1	One CS3 space continuous access idle cycle	(Initial value)

Bit 6 (CW2)	Description	
0	No CS2 space continuous access idle cycles	
1	One CS2 space continuous access idle cycle	(Initial value)

Bit 5 (CW1)	Description	
0	No CS1 space continuous access idle cycles	
1	One CS1 space continuous access idle cycle	(Initial value)

Bit 4 (CW0)	Description	
0	No CS0 space continuous access idle cycles	
1	One CS0 space continuous access idle cycle	(Initial value)

- Bits 3 to 0—$\overline{\text{CS}}$ Assert Extension Specification (SW3, SW2, SW1, SW0): The CS assert cycle extension specification is for making insertions to prevent extension of the $\overline{\text{RD}}$ signal or $\overline{\text{WRL}}$ signal assert period beyond the length of the $\overline{\text{CS}}$n signal assert period. Extended cycles insert one cycle before and after each bus cycle, which simplifies interfaces with external devices and also has the effect of extending write data hold time. Refer to section 7.3.3, $\overline{\text{CS}}$ Assert Period Extension, for details.

  SW3 specifies the $\overline{\text{CS}}$ assert extension for CS3 space access; SW2 specifies the $\overline{\text{CS}}$ assert extension for CS2 space access; SW1 specifies the $\overline{\text{CS}}$ assert extension for CS1 space access and SW0 specifies the $\overline{\text{CS}}$ assert extension for CS0 space access.

Bit 3 (SW3)	Description	
0	No CS3 space $\overline{\text{CS}}$ assert extension	
1	CS3 space $\overline{\text{CS}}$ assert extension	(Initial value)

Bit 2 (SW2)	Description	
0	No CS2 space $\overline{\text{CS}}$ assert extension	
1	CS2 space $\overline{\text{CS}}$ assert extension	(Initial value)

Bit 1 (SW1)	Description	
0	No CS1 space $\overline{CS}$ assert extension	
1	CS1 space $\overline{CS}$ assert extension	(Initial value)

Bit 0 (SW0)	Description	
0	No CS0 space $\overline{CS}$ assert extension	
1	CS0 space $\overline{CS}$ assert extension	(Initial value)

7.2.3 Wait Control Register 1 (WCR1)

Wait control register 1 (WCR1) is a 16-bit read/write register that specifies the number of wait cycles (0 to 3) for each CS space.

WCR1 is initialized to H'FFFF by power-on resets.

Bit:	15	14	13	12	11	10	9	8
	—	—	W31	W30	—	—	W21	W20
Initial value:	1	1	1	1	1	1	1	1
R/W:	R	R	R/W	R/W	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	—	—	W11	W10	—	—	W01	W00
Initial value:	1	1	1	1	1	1	1	1
R/W:	R	R	R/W	R/W	R	R	R/W	R/W

- Bits 15 and 14—Reserved: These bits are always read as 1. The write value should always be 1.

- Bits 13 and 12—CS3 Space Wait Specification (W31, W30): These bits specify the number of waits for CS3 space accesses.

Bit 13 (W31)	Bit 12 (W30)	Description	
0	0	No wait (external wait input disabled)	
	1	1-wait external wait input enabled	
1	0	2-wait external wait input enabled	
	1	3-wait external wait input enabled	(Initial value)

- Bits 11 and 10—Reserved: These bits are always read as 1. The write value should always be 1.

- Bits 9 and 8—CS2 Space Wait Specification (W21, W20): These bits specify the number of waits for CS2 space accesses.

Bit 9 (W21)	Bit 8 (W20)	Description	
0	0	No wait (external wait input disabled)	
	1	1-wait external wait input enabled	
1	0	2-wait external wait input enabled	
	1	3-wait external wait input enabled	(Initial value)

- Bits 7 and 6—Reserved. Either 0 or 1 can be written. These bits read the written value.
- Bits 5 and 4—CS1 Space Wait Specification (W11, W10): These bits specify the number of waits for CS1 space accesses.

Bit 5 (W11)	Bit 4 (W10)	Description	
0	0	No wait (external wait input disabled)	
	1	1-wait external wait input enabled	
1	0	2-wait external wait input enabled	
	1	3-wait external wait input enabled	(Initial value)

- Bits 3 and 2—Reserved. Either 0 or 1 can be written. These bits read the written value.
- Bits 1 and 0—CS0 Space Wait Specification (W01, W00): These bits specify the number of waits for CS0 space accesses.

Bit 1 (W01)	Bit 0 (W00)	Description	
0	0	No wait (external wait input disabled)	
	1	1-wait external wait input enabled	
1	0	2-wait external wait input enabled	
	1	3-wait external wait input enabled	(Initial value)

7.3 Accessing Ordinary Space

A strobe signal is output by ordinary space accesses to provide primarily for SRAM or ROM direct connections.

7.3.1 Basic Timing

Figure 7.3 shows the basic timing of ordinary space accesses. Ordinary access bus cycles are performed in 2 states.

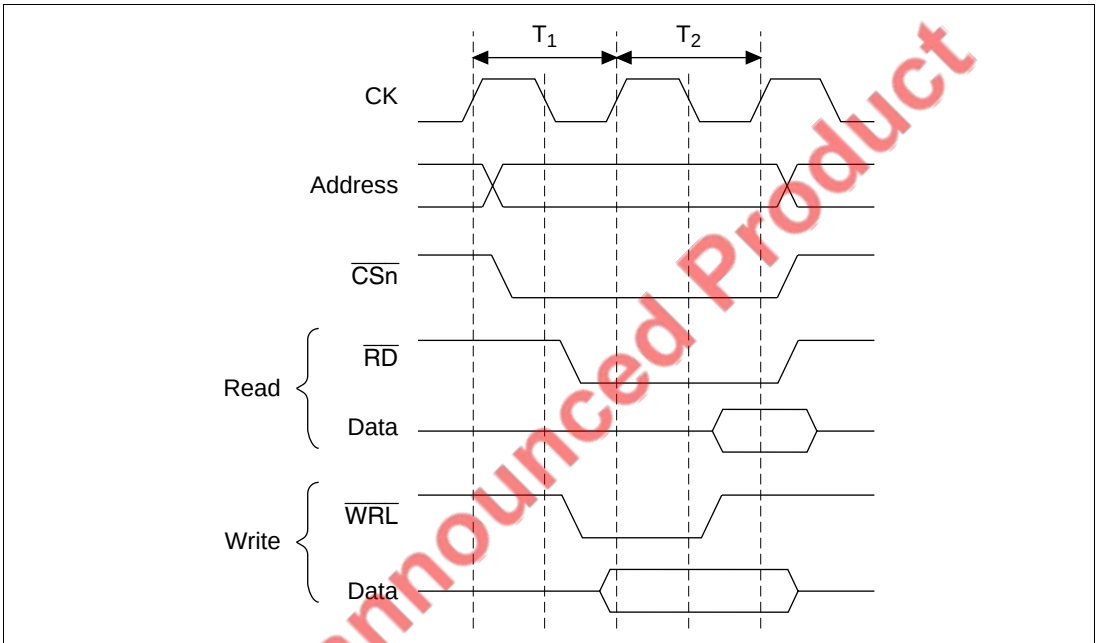


Figure 7.3 Basic Timing of Ordinary Space Access

During a read, irrespective of operand size, all bits in the data bus width for the access space (address) are fetched by the LSI on $\overline{RD}$, using the required byte locations.

During a write, the following signals are associated with transfer of these actual byte locations: $\overline{WRL}$ (bits 7 to 0).

7.3.2 Wait State Control

The number of wait states inserted into ordinary space access states can be controlled using the WCR settings. The specified number of T_w cycles (0 to 3 waits) are inserted as software wait cycles with the timing shown in figure 7.4.

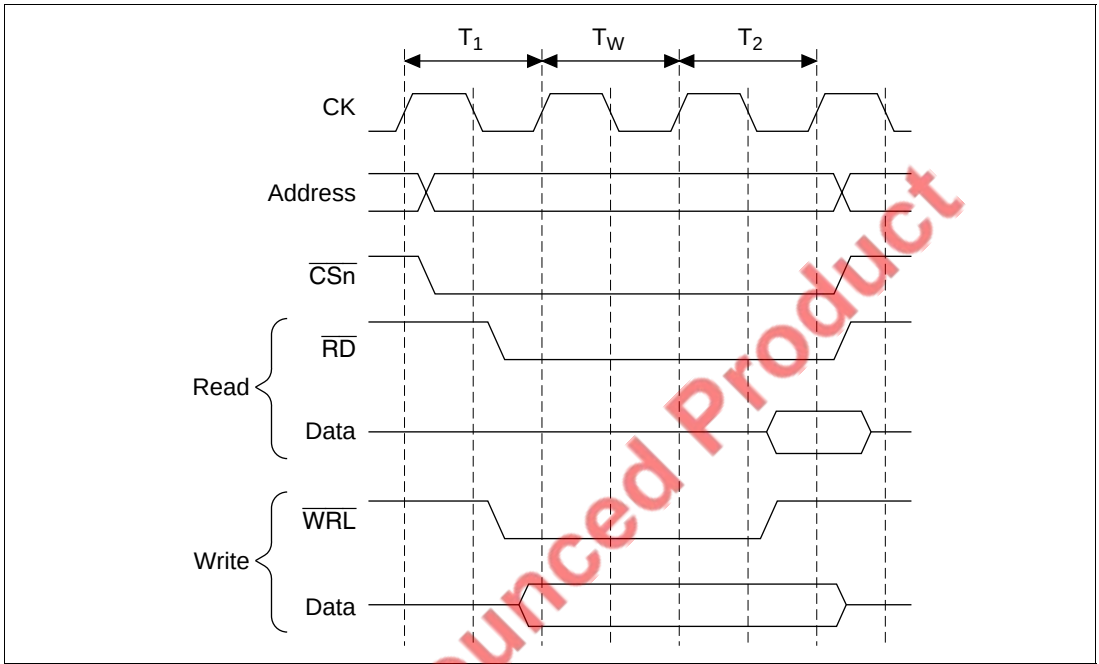


Figure 7.4 Wait Timing of Ordinary Space Access (Software Wait Only)

When the wait is specified by software using WCR, the wait input $\overline{\text{WAIT}}$ signal from outside is sampled. Figure 7.5 shows the $\overline{\text{WAIT}}$ signal sampling. The $\overline{\text{WAIT}}$ signal is sampled at the clock rise one cycle before the clock rise when T_w state shifts to T_2 state.

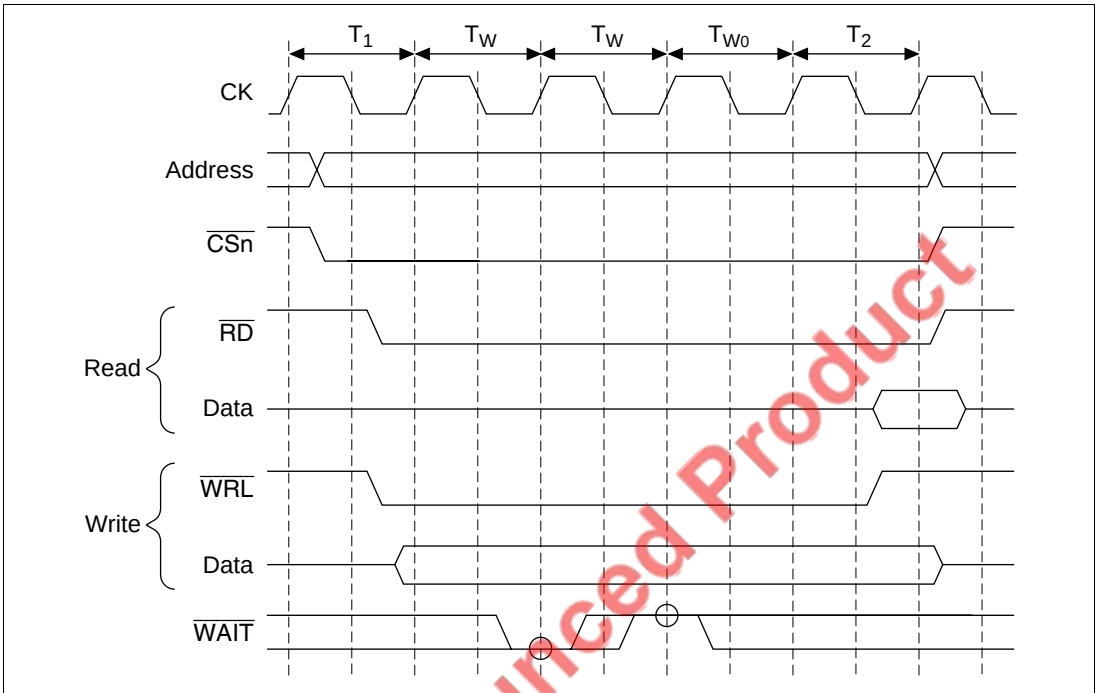


Figure 7.5 Wait State Timing of Ordinary Space Access (Wait States from Software Wait 2 State + $\overline{\text{WAIT}}$ Signal)

7.3.3 $\overline{CS}$ Assert Period Extension

Idle cycles can be inserted to prevent extension of the $\overline{RD}$ signal or $\overline{WRL}$ signal assert period beyond the length of the $\overline{CSn}$ signal assert period by setting the SW3 to SW0 bits of BCR2. This allows for flexible interfaces with external circuitry. The timing is shown in figure 7.6. T_h and T_f cycles are added respectively before and after the ordinary cycle. Only $\overline{CSn}$ is asserted in these cycles; $\overline{RD}$ and $\overline{WRL}$ signals are not. Further, data is extended up to the T_f cycle, which is effective for gate arrays and the like, which have slower write operations.

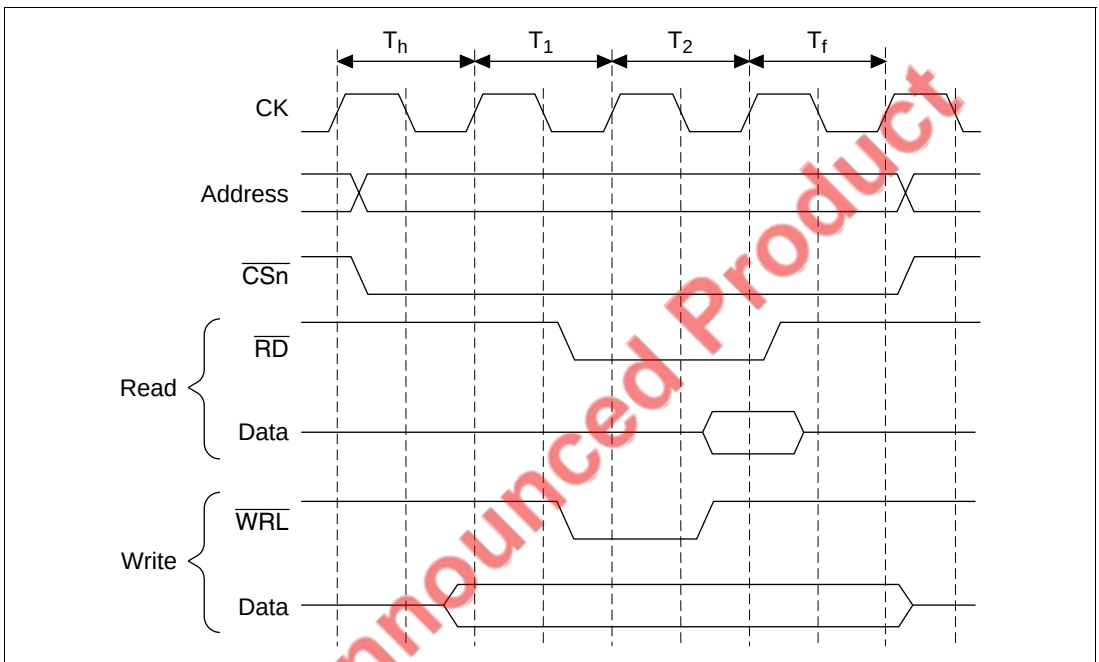


Figure 7.6 $\overline{CS}$ Assert Period Extension Function

7.4 Waits between Access Cycles

When a read from a slow device is completed, data buffers may not go off in time to prevent data conflicts with the next access. If there is a data conflict during memory access, the problem can be solved by inserting a wait in the access cycle.

To enable detection of bus cycle starts, waits can be inserted between access cycles during continuous accesses of the same CS space by negating the $\overline{\text{CSn}}$ signal once.

7.4.1 Prevention of Data Bus Conflicts

For the two cases of write cycles after read cycles, and read cycles for a different area after read cycles, waits are inserted so that the number of idle cycles specified by the IW31 to IW00 bits of the BCR2 is inserted. When idle cycles already exist between access cycles, only the number of empty cycles remaining beyond the specified number of idle cycles is inserted.

Figure 7.7 shows an example of idles between cycles. In this example, 1 idle between CSn space cycles has been specified, so when a CSm space write immediately follows a CSn space read cycle, 1 idle cycle is inserted.

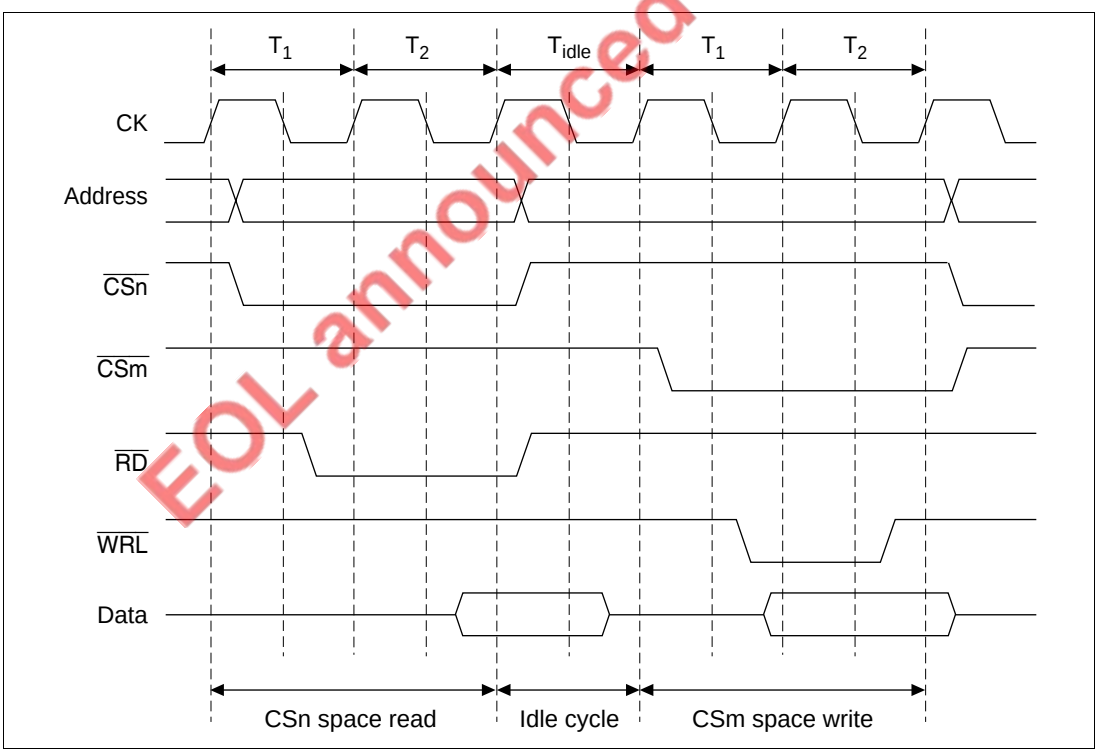


Figure 7.7 Idle Cycle Insertion Example

IW31 and IW30 specify the number of idle cycles required after a CS3 space read either to read other external spaces, or for this LSI, to do write accesses. In the same manner, IW21 and IW20 specify the number of idle cycles after a CS2 space read, IW11 and IW10, the number after a CS1 space read, and IW01 and IW00, the number after a CS0 space read.

0 to 3 cycles can be specified for CS space.

7.4.2 Simplification of Bus Cycle Start Detection

For consecutive accesses of the same CS space, waits are inserted so that the number of idle cycles designated by the CW3 to CW0 bits of the BCR2 occur. However, for write cycles after reads, the number of idle cycles inserted will be the larger of the two values defined by the IW and CW bits. When idle cycles already exist between access cycles, waits are not inserted. Figure 7.8 shows an example. A continuous access idle is specified for CSn space, and CSn space is consecutively write accessed.

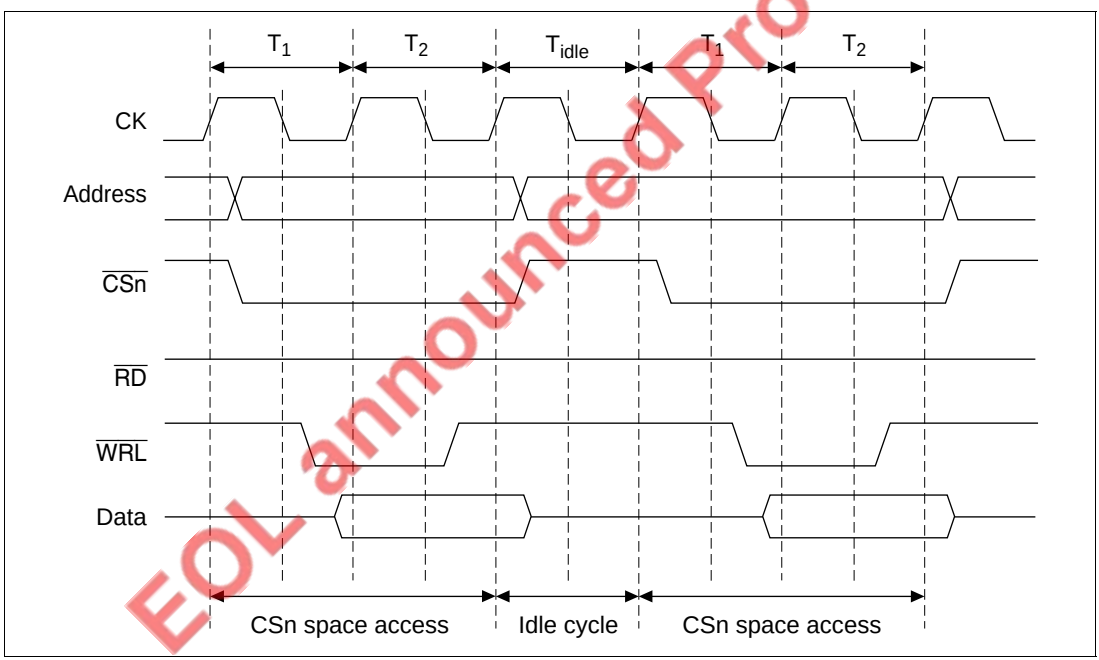


Figure 7.8 Same Space Consecutive Access Idle Cycle Insertion Example

7.5 Memory Connection Examples

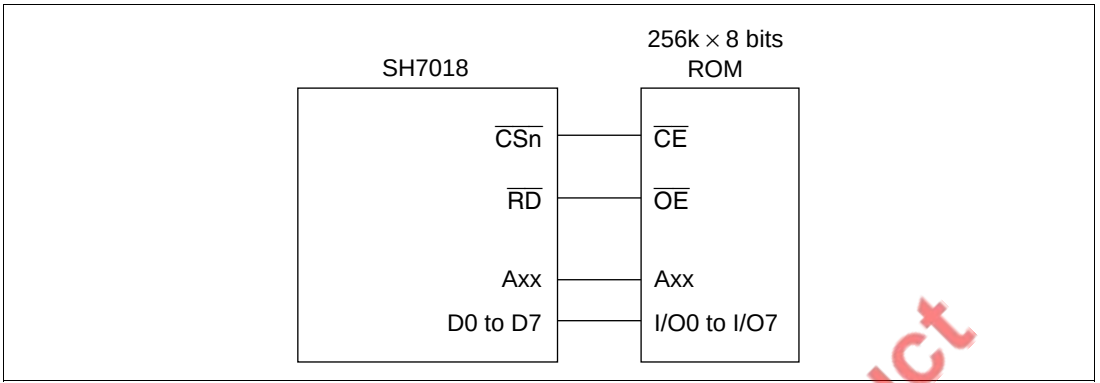


Figure 7.9 8-Bit Data Bus Width ROM Connection

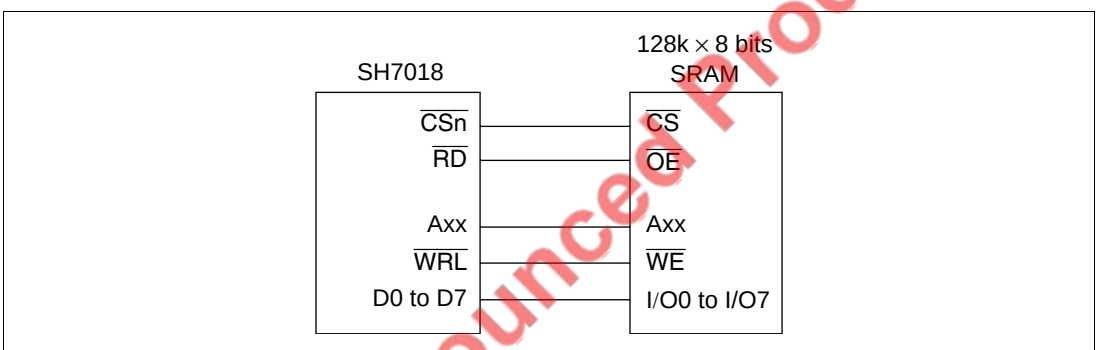


Figure 7.10 8-Bit Data Bus Width SRAM Connection

Section 8 Multifunction Timer Pulse Unit (MTU)

8.1 Overview

The SH7018 has an on-chip 16-bit multifunction timer pulse unit (MTU) with three channels of 16-bit timers.

8.1.1 Features

- Can process a maximum of six different pulse outputs and inputs.
- Has eight timer general registers (TGR), four for channel 0 and two each for channels 1 and 2, that can be set to function independently as output compare registers or (except for TGR0B and TGR0D of channel 0) as input capture registers. The channel 0 TGRC and TGRD registers can be used as buffer registers.
- Can select six counter input clock sources for all channels
- All channels can be set for the following operating modes:
  - Compare match waveform output: 0 output/1 output/toggle output selectable.
  - Input capture function: Selectable rising edge, falling edge, or both rising and falling edge detection.
  - Counter clearing function: Counters can be cleared by a compare-match or input capture.
  - Synchronizing mode: Two or more timer counters (TCNT) can be written to simultaneously. Two or more timer counters can be simultaneously cleared by a compare-match or input capture. Counter synchronization functions enable synchronized register input/output.
  - PWM mode: PWM output can be provided with any duty cycle. When combined with the counter synchronizing function, enables up to four-phase* PWM output.

Note: * When channels 0 to 2 are set to PWM mode 1

- Channel 0 can be set for buffer operation
  - Input capture register double buffer configuration possible
  - Output compare register automatic re-write possible
- Cascade connection operation
  - Can be operated as a 32-bit counter by using the channel 2 input clock for channel 1 overflow/underflow
- High speed access via internal 16-bit bus
- Eleven interrupt sources
  - Channel 0 has two dual-function compare-match/input capture interrupts, two compare-match interrupts, and one overflow interrupt, which can be requested independently.

— Channels 1 and 2 have two compare-match/input capture interrupts, one overflow interrupt, and one underflow interrupt which can be requested independently.

- A/D converter conversion start trigger can be generated
  — Channel 0 to 2 compare-match/input capture signals can be used as A/D converter conversion start triggers.

Table 8.1 summarizes the MTU functions.

Table 8.1 MTU Functions

Item		Channel 0	Channel 1	Channel 2
Counter clocks		Internal: φ/1, φ/4, φ/16, φ/64, φ/256, φ/1024 Six to each channel		
General registers		TGR0A	TGR1A	TGR2A
		TGR0B	TGR1B	TGR2B
General registers/buffer registers		TGR0C TGR0D	No	No
Input/output pins		TIOC0A	TIOC1A	TIOC2A
		TIOC0C	TIOC1B	TIOC2B
Counter clear function		TGR compare-match or input capture	TGR compare-match or input capture	TGR compare-match or input capture
Compare match output	0	Yes	Yes	Yes
	1	Yes	Yes	Yes
	Toggle	Yes	Yes	Yes
Input capture function		Yes	Yes	Yes
Synchronization		Yes	Yes	Yes
Buffer operation		Yes	No	No
PWM mode 1		Yes	Yes	Yes
PWM mode 2		Yes	Yes	Yes
A/D conversion start trigger		TGR0A compare match or input capture	TGR1A compare match or input capture	TGR2A compare match or input capture
Interrupt sources		Compare match/input capture 0A	Compare match/input capture 1A	Compare match/input capture 2A
		Compare match 0B	Compare match/input capture 1B	Compare match/input capture 2B
		Compare match/input capture 0C	Overflow	Overflow
		Compare match 0D	—	—
		Overflow	—	—

8.1.2 Block Diagram

Figure 8.1 is the block diagram of the MTU.

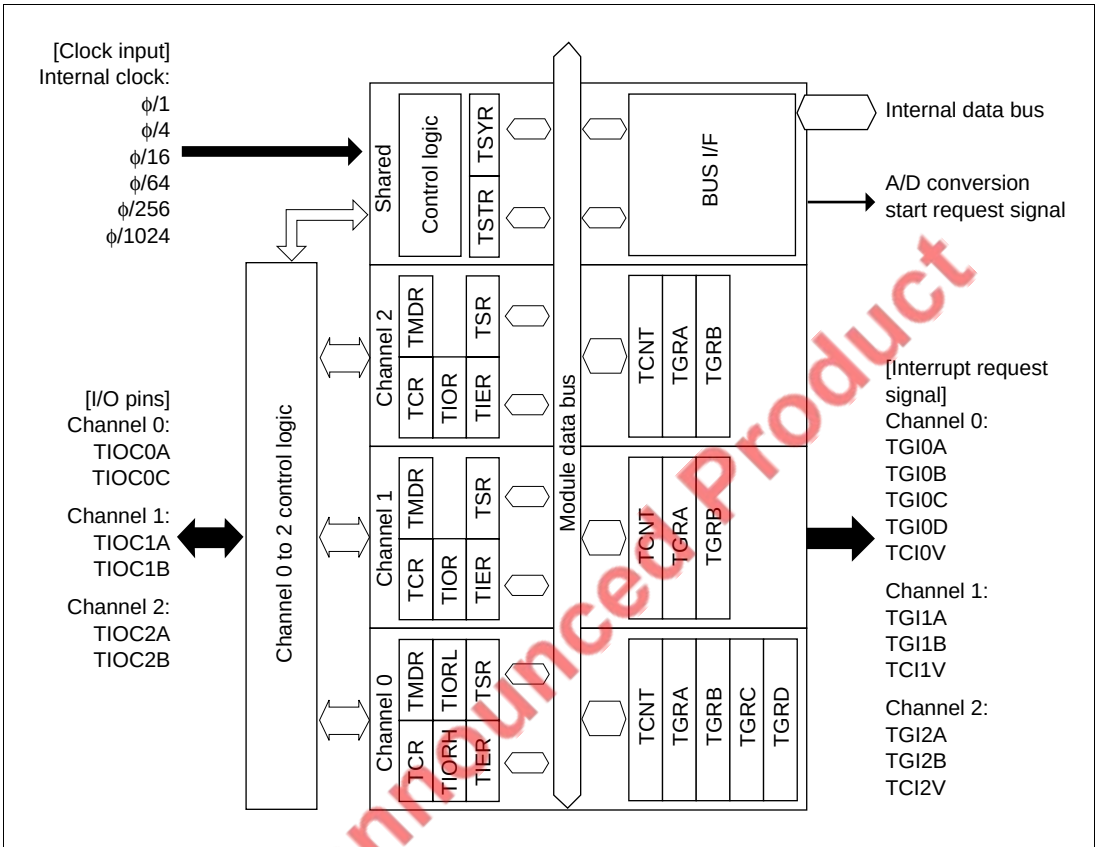


Figure 8.1 MTU Block Diagram

8.1.3 Pin Configuration

Table 8.2 summarizes the MTU pins.

Table 8.2 Pin Configuration

Channel	Name	Pin Name	I/O	Function
0	Input capture/output compare-match 0A	TIOC0A	I/O	TGR0A input capture input/output compare output/PWM output pin
	Input capture/output compare-match 0C	TIOC0C	I/O	TGR0C input capture input/output compare output/PWM output pin
1	Input capture/output compare-match 1A	TIOC1A	I/O	TGR1A input capture input/output compare output/PWM output pin
	Input capture/output compare-match 1B	TIOC1B	I/O	TGR1B input capture input/output compare output/PWM output pin
2	Input capture/output compare-match 2A	TIOC2A	I/O	TGR2A input capture input/output compare output/PWM output pin
	Input capture/output compare-match 2B	TIOC2B	I/O	TGR2B input capture input/output compare output/PWM output pin

Note: The TIOC pins output undefined values when they are set to input capture and timer output by the pin function controller (PFC).

8.1.4 Register Configuration

Table 8.3 summarizes the MTU register configuration.

Table 8.3 Register Configuration

Channel	Name	Abbreviation	R/W	Initial Value	Address	Access Size (Bits)*[1]
Shared	Timer start register	TSTR	R/W	H'00	H'FFFF8240	8, 16
	Timer synchro register	TSYR	R/W	H'00	H'FFFF8241	
0	Timer control register 0	TCR0	R/W	H'00	H'FFFF8260	8, 16, 32
	Timer mode register 0	TMDR0	R/W	H'C0	H'FFFF8261	
	Timer I/O control register 0H	TIOR0H	R/W	H'00	H'FFFF8262	
	Timer I/O control register 0L	TIOR0L	R/W	H'00	H'FFFF8263	
	Timer interrupt enable register 0	TIER0	R/W	H'40	H'FFFF8264	
	Timer status register 0	TSR0	R/(W)*[2]	H'C0	H'FFFF8265	
	Timer counter 0	TCNT0	R/W	H'0000	H'FFFF8266	16, 32
	General register 0A	TGR0A	R/W	H'FFFF	H'FFFF8268	
	General register 0B	TGR0B	R/W	H'FFFF	H'FFFF826A	
	General register 0C	TGR0C	R/W	H'FFFF	H'FFFF826C	
	General register 0D	TGR0D	R/W	H'FFFF	H'FFFF826E	
1	Timer control register 1	TCR1	R/W	H'00	H'FFFF8280	8, 16
	Timer mode register 1	TMDR1	R/W	H'C0	H'FFFF8281	
	Timer I/O control register 1	TIOR1	R/W	H'00	H'FFFF8282	8
	Timer interrupt enable register 1	TIER1	R/W	H'40	H'FFFF8284	8, 16, 32
	Timer status register 1	TSR1	R/(W)*[2]	H'C0	H'FFFF8285	
	Timer counter 1	TCNT1	R/W	H'0000	H'FFFF8286	16, 32
	General register 1A	TGR1A	R/W	H'FFFF	H'FFFF8288	
	General register 1B	TGR1B	R/W	H'FFFF	H'FFFF828A	

Table 8.3 Register Configuration (cont)

Channel	Name	Abbreviation	R/W	Initial Value	Address	Access Size (Bits)*[1]
2	Timer control register 2	TCR2	R/W	H'00	H'FFFF82A0	8, 16
	Timer mode register 2	TMDR2	R/W	H'C0	H'FFFF82A1	
	Timer I/O control register 2	TIOR2	R/W	H'00	H'FFFF82A2	8
	Timer interrupt enable register 2	TIER2	R/W	H'40	H'FFFF82A4	8, 16, 32
	Timer status register 2	TSR2	R/(W)*[2]	H'C0	H'FFFF82A5	
	Timer counter 2	TCNT2	R/W	H'0000	H'FFFF82A6	16, 32
	General register 2A	TGR2A	R/W	H'FFFF	H'FFFF82A8	
	General register 2B	TGR2B	R/W	H'FFFF	H'FFFF82AA	

Notes: Do not access empty addresses.
1. 16-bit registers (TCNT, TGR) cannot be read or written in 8-bit units.
2. Write 0 to clear flags.

8.2 MTU Register Descriptions

8.2.1 Timer Control Register (TCR)

The TCR is an 8-bit read/write register for controlling the TCNT counter for each channel. The MTU has three TCR registers, one for each of the channels 0 to 2. TCR is initialized to H'00 by a power-on reset or the standby mode.

Channel 0: TCR0

Bit:	7	6	5	4	3	2	1	0
	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Channels 1 and 2: TCR1, TCR2

Bit:	7	6	5	4	3	2	1	0
	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- Bits 7 to 5—Counter Clear 2, 1, and 0 (CCLR2, CCLR1, CCLR0): Select the counter clear source for the TCNT counter.

Channel 0:

Bit 7: CCLR2	Bit 6: CCLR1	Bit 5: CCLR0	Description
0	0	0	TCNT clear disabled (Initial value)
		1	TCNT is cleared by TGRA compare-match or input capture
	1	0	TCNT is cleared by TGRB compare-match
		1	Synchronizing clear: TCNT is cleared in synchronization with clear of other channel counters operating in sync*[1]
1	0	0	TCNT clear disabled
		1	TCNT is cleared by TGRC compare-match or input capture*[2]
	1	0	TCNT is cleared by TGRD compare-match*[2]
		1	Synchronizing clear: TCNT is cleared in synchronization with clear of other channel counters operating in sync*[1]

Notes: 1. Setting the SYNC bit of the TSYR to 1 sets the synchronization.
2. When TGRC or TGRD are functioning as buffer registers, TCNT is not cleared because the buffer registers have priority and compare-match/input captures do not occur.

Channels 1 and 2:

Bit 7: Reserved*[1]	Bit 6: CCLR1	Bit 5: CCLR0	Description
0	0	0	TCNT clear disabled (Initial value)
		1	TCNT is cleared by TGRA compare-match or input capture
	1	0	TCNT is cleared by TGRB compare-match or input capture
		1	Synchronizing clear: TCNT is cleared in synchronization with clear of other channel counters operating in sync*[2]

Notes: 1. The bit 7 of channels 1 and 2 is reserved. This bit is always read as 0. The write value should always be 0.
2. Setting the SYNC bit of the TSYR to 1 sets the synchronization.

- Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0): CKEG1 and CKEG0 select the input clock edges. When counting is done on both edges of the internal clock the input clock frequency becomes 1/2 (Example: both edges of $\phi/4$ = rising edge of $\phi/2$).

Bit 4: CKEG1	Bit 3: CKEG0	Description	
0	0	Count on rising edges	(Initial value)
	1	Count on falling edges	
1	X	Count on both rising and falling edges	

Notes: 1. X: 0 or 1, don't care.
2. Internal clock edge selection is effective when the input clock is ϕ/4 or slower. These settings are ignored when ϕ/1, or the overflow of another channel is selected for the input clock.

- Bits 2 to 0—Timer Prescaler 2 to 0 (TPSC2 to TPSC0): TPSC2 to TPSC0 select the counter clock source for the TCNT. An independent clock source can be selected for each channel. Table 8.4 shows the possible settings for each channel.

Table 8.4 MTU Clock Sources

	Internal Clock						
Channel	ϕ/1	ϕ/4	ϕ/16	ϕ/64	ϕ/256	ϕ/1024	Other Channel Overflow
0	O	O	O	O	X	X	X
1	O	O	O	O	O	X	O
2	O	O	O	O	X	O	X

Note: Symbols: O: Setting possible X: Setting not possible

Channel 0:

Bit 2: TPSC2	Bit 1: TPSC1	Bit 0: TPSC0	Description	
0	0	0	Internal clock: count with ϕ/1	(Initial value)
		1	Internal clock: count with ϕ/4	
	1	0	Internal clock: count with ϕ/16	
		1	Internal clock: count with ϕ/64	
1	0	0	Reserved (Do not set)	
		1	Reserved (Do not set)	
	1	0	Reserved (Do not set)	
		1	Reserved (Do not set)	

Channel 1:

Bit 2: TPSC2	Bit 1: TPSC1	Bit 0: TPSC0	Description	
0	0	0	Internal clock: count with ϕ/1	(Initial value)
		1	Internal clock: count with ϕ/4	
	1	0	Internal clock: count with ϕ/16	
		1	Internal clock: count with ϕ/64	
1	0	0	Reserved (Do not set)	
		1	Reserved (Do not set)	
	1	0	Internal clock: count with ϕ/256	
		1	Count with the TCNT2 overflow	

Channel 2:

Bit 2: TPSC2	Bit 1: TPSC1	Bit 0: TPSC0	Description	
0	0	0	Internal clock: count with ϕ/1	(Initial value)
		1	Internal clock: count with ϕ/4	
	1	0	Internal clock: count with ϕ/16	
		1	Internal clock: count with ϕ/64	
1	0	0	Reserved (Do not set)	
		1	Reserved (Do not set)	
	1	0	Reserved (Do not set)	
		1	Internal clock: count with ϕ/1024	

8.2.2 Timer Mode Register (TMDR)

The TMDR is an 8-bit read/write register that sets the operating mode for each channel. The MTU has three TMDR registers, one for each channel. TMDR is initialized to H'C0 by a power-on reset.

Channel 0: TMDR0

Bit:	7	6	5	4	3	2	1	0
	—	—	BFB	BFA	MD3	MD2	MD1	MD0
Initial value:	1	1	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Channels 1 and 2: TMDR1, TMDR2

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	MD3	MD2	MD1	MD0
Initial value:	1	1	0	0	0	0	0	0
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

- Bits 7 and 6—Reserved: These bits are always read as 1. The write value should always be 1.
- Bit 5—Buffer Operation B (BFB): Designates whether to use the TGRB register for normal operation, or buffer operation in combination with the TGRD register. When using TGRD as a buffer register, no TGRD register input capture/output compares are generated.

  This bit is reserved in channels 1 and 2, which have no TGRD registers. It is always read as 0. The write value should always be 0.

Bit 5: BFB	Description	
0	TGRB operates normally	(Initial value)
1	TGRB and TGRD buffer operation	

- Bit 4—Buffer Operation A (BFA): Designates whether to use the TGRA register for normal operation, or buffer operation in combination with the TGRC register. When using TGRC as a buffer register, no TGRC register input capture/output compares are generated.

  This bit is reserved in channels 1 and 2, which have no TGRC registers. It is always read as 0. The write value should always be 0.

Bit 4: BFA	Description	
0	TGRA operates normally	(Initial value)
1	TGRA and TGRC buffer operation	

- Bits 3 to 0—Modes 3 to 0 (MD3 to MD0): These bits set the timer operation mode.

Bit 3: MD3	Bit 2: MD2	Bit 1: MD1	Bit 0: MD0	Description	
0	0	0	0	Normal operation	(Initial value)
			1	Reserved (do not set)	
		1	0	PWM mode 1	
			1	PWM mode 2	
0	1	*	*	Reserved (Do not set)	
1	*	*	*	Reserved (Do not set)	

*: Don't care

8.2.3 Timer I/O Control Register (TIOR)

The TIOR is a register that controls the TGR. The MTU has four TIOR registers, two for channel 0, and one each for channels 1 and 2. TIOR is initialized to H'00 by a power-on reset.

Channel 0: TIOR0H

Bit:	7	6	5	4	3	2	1	0
	—	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Channels 1 and 2: TIOR1, TIOR2

Bit:	7	6	5	4	3	2	1	0
	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 7 to 4—I/O Control B3 to B0 (IOB3 to IOB0): These bits set the TGRB register function.

Bit 7 of TIOR0H is reserved bit. It is always read as 0. The write value should always be 0.

Bits 3 to 0—I/O Control A3 to A0 (IOA3 to IOA0): These bits set the TGRA register function.

Channel 0: TIOR0L

Bit:	7	6	5	4	3	2	1	0
	—	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: When the TGRC or TGRD registers are set for buffer operation, these settings become ineffective and the operation is as a buffer register.

- Bit 7—Reserved. This bit is always read as 0. The write value should always be 0.
- Bits 6 to 4—I/O Control D2 to D0 (IOD2 to IOD0): These bits set the TGRD register function.
- Bits 3 to 0—I/O Control C3 to C0 (IOC3 to IOC0): These bits set the TGRC register function.

Channel 0 (TIOR0H Register):

Bit 6: IOB2	Bit 5: IOB1	Bit 4: IOB0	Description		
0	0	0	TGR0B is an output compare register	Output disabled	(Initial value)
		1		Initial output is 0	Output 0 on compare-match
	1	0			Output 1 on compare-match
		1			Toggle output on compare-match
1	0	0		Output disabled	
		1		Initial output is 1	Output 0 on compare-match
	1	0			Output 1 on compare-match
		1			Toggle output on compare-match

Bit 3: IOA3	Bit 2: IOA2	Bit 1: IOA1	Bit 0: IOA0	Description		
0	0	0	0	TGR0A is an output compare register	Output disabled	(Initial value)
			1		Initial output is 0	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
	1	0	0		Output disabled	
			1		Initial output is 1	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
1	0	0	0	TGR0A is an input capture register	Capture input source is the TIOC0A pin	Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			
	1	0	0		Capture input source is channel 1/ count clock	Input capture on TCNT1 count up/count down
			1			
		1	0			
			1			

Channel 0 (TIOR0L Register):

Bit 6: IOD2	Bit 5: IOD1	Bit 4: IOD0	Description		
0	0	0	TGR0D is an output compare register	Output disabled	(Initial value)
		1		Initial output is 0	Output 0 on compare-match
	1	0			Output 1 on compare-match
		1			Toggle output on compare-match
1	0	0		Output disabled	
		1		Initial output is 1	Output 0 on compare-match
	1	0			Output 1 on compare-match
		1			Toggle output on compare-match

Bit 3: IOC3	Bit 2: IOC2	Bit 1: IOC1	Bit 0: IOC0	Description		
0	0	0	0	TGR0C is an output compare register	Output disabled	(Initial value)
			1		Initial output is 0	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
	1	0	0		Output disabled	
			1		Initial output is 1	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
1	0	0	0	TGR0C is an input capture register	Capture input source is the TIOC0C pin	Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			
	1	0	0		Capture input source is channel 1/ count clock	Input capture on TCNT1 count up/count down
			1			
		1	0			
			1			

Note: When the BFA bit of TMDR0 is set to 1 and TGR0C is being used as a buffer register, these settings become ineffective and input capture/output compares do not occur.

Channel 1 (TIOR1 Register):

Bit 7: IOB3	Bit 6: IOB2	Bit 5: IOB1	Bit 4: IOB0	Description		
0	0	0	0	TGR1B is an output compare register	Output disabled	(Initial value)
			1		Initial output is 0	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
	1	0	0		Output disabled	
			1		Initial output is 1	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
1	0	0	0	TGR1B is an input capture register	Capture input source is the TIOC1B pin	Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			
	1	0	0		Capture input source TGR0C compare/match input capture	Input capture on channel TGR0C compare-match/input capture generation
			1			
		1	0			
			1			

Bit 3: IOA3	Bit 2: IOA2	Bit 1: IOA1	Bit 0: IOA0	Description		
0	0	0	0	TGR1A is an output compare register	Output disabled	(Initial value)
			1		Initial output is 0	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
	1	0	0		Output disabled	
			1		Initial output is 1	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
1	0	0	0	TGR1A is an input capture register	Capture input source is the TIOC1A pin	Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			
	1	0	0		Capture input source is TGR0A compare-match/input capture	Input capture on channel 0/TGR0A compare-match/input capture generation
			1			
		1	0			
			1			

Channel 2 (TIOR2 Register):

Bit 7: IOB3	Bit 6: IOB2	Bit 5: IOB1	Bit 4: IOB0	Description		
0	0	0	0	TGR2B is an output compare register	Output disabled	(Initial value)
			1		Initial output is 0	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
	1	0	0		Output disabled	
			1		Initial output is 1	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
1	0	0	0	TGR2B is an input capture register	Capture input source is the TIOC2B pin	Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			
	1	0	0			Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			

Bit 3: IOA3	Bit 2: IOA2	Bit 1: IOA1	Bit 0: IOA0	Description		
0	0	0	0	TGR2A is an output compare register	Output disabled	(Initial value)
			1		Initial output is 0	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
	1	0	0		Output disabled	
			1		Initial output is 1	Output 0 on compare-match
		1	0			Output 1 on compare-match
			1			Toggle output on compare-match
1	0	0	0	TGR2A is an input capture register	Capture input source is the TIOC2A pin	Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			
	1	0	0			Input capture on rising edge
			1			Input capture on falling edge
		1	0			Input capture on both edges
			1			

8.2.4 Timer Interrupt Enable Register (TIER)

The TIER is an 8-bit register that controls the enable/disable of interrupt requests for each channel. The MTU has three TIER registers, one each for channel. TIER is initialized to H'40 by a power-on reset.

Channel 0: TIER0

Bit:	7	6	5	4	3	2	1	0
	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA
Initial value:	0	1	0	0	0	0	0	0
R/W:	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Channels 1 and 2: TIER1, TIER2

Bit:	7	6	5	4	3	2	1	0
	TTGE	—	—	TCIEV	—	—	TGIEB	TGIEA
Initial value:	0	1	0	0	0	0	0	0
R/W:	R/W	R	R	R/W	R	R	R/W	R/W

- Bit 7—A/D Conversion Start Request Enable (TTGE): Enables or disables generation of an A/D conversion start request by a TGRA register input capture/compare-match.

Bit 7: TTGE	Description	
0	Disable A/D conversion start requests generation	(Initial value)
1	Enable A/D conversion start request generation	

- Bit 6—Reserved: This bit is always read as 1. The write value should always be 1.
- Bit 5—Reserved. This bit is always read as 0. The write value should always be 0.
- Bit 4—Overflow Interrupt Enable (TCIEV): Enables or disables interrupt requests when the overflow flag TCFV of the timer status register (TSR) is set to 1.

Bit 4: TCIEV	Description	
0	Disable TCFV interrupt requests (TCIV)	(Initial value)
1	Enable TCFV interrupt requests (TCIV)	

- Bit 3—TGR Interrupt Enable D (TGIED): Enables or disables interrupt TGFD requests when the TGFD bit of the channel 0 of the TSR register is set to 0.

  This bit is reserved for channels 1 and 2. It is always read as 0. The write value should always be 0.

Bit 3: TGIED	Description	
0	Disable interrupt requests (TGID) due to the TGFD bit	(Initial value)
1	Enable interrupt requests (TGID) due to the TGFD bit	

- Bit 2—TGR Interrupt Enable C (TGIEC): Enables or disables TGFC interrupt requests when the TGFC bit of the Channel 0 of the TSR register is set to 1.

  This bit is reserved for channels 1 and 2. It is always read as 0. The write value should always be 0.

Bit 2: TGIEC	Description	
0	Disable interrupt requests (TGIC) due to the TGFC bit	(Initial value)
1	Enable interrupt requests (TGIC) due to the TGFC bit	

- Bit 1—TGR Interrupt Enable B (TGIEB): Enables or disables TGFB interrupt requests when the TGFB bit of the TSR register is set to 1.

Bit 1: TGIEB	Description	
0	Disable interrupt requests (TGIB) due to the TGFB bit	(Initial value)
1	Enable interrupt requests (TGIB) due to the TGFB bit	

- Bit 0—TGR Interrupt Enable A (TGIEA): Enables or disables TGFA interrupt requests when the TGFA bit of the TSR register is set to 1.

Bit 0: TGIEA	Description	
0	Disable interrupt requests (TGIA) due to the TGFA bit	(Initial value)
1	Enable interrupt requests (TGIA) due to the TGFA bit	

8.2.5 Timer Status Register (TSR)

The timer status register (TSR) is an 8-bit register that indicates the status of each channel. The MTU has three TSR registers, one each for channel. TSR is initialized to H'C0 by a power-on reset.

Channel 0: TSR0

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA
Initial value:	1	1	0	0	0	0	0	0
R/W:	R	R	R	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: * Only 0 writes to clear the flags are possible.

Channels 1 and 2: TSR1, TSR2

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TCFV	—	—	TGFB	TGFA
Initial value:	1	1	0	0	0	0	0	0
R/W:	R	R	R	R/(W)*	R	R	R/(W)*	R/(W)*

Note: * Only 0 writes to clear the flags are possible.

- Bits 7 and 6—Reserved: These bits are always read as 1. The write value should always be 1.
- Bit 5—Reserved: This bit is always read as 0. The write value should always be 0.
- Bit 4—Overflow Flag (TCFV): This status flag indicates the occurrence of a TCNT counter overflow.

Bit 4: TCFV	Description
0	Clear condition: With TCFV =1, a 0 write to TCFV after reading it (Initial value)
1	Set condition: When the TCNT value overflows (H'FFFF → H'0000)

- Bit 3—Output Compare Flag D (TGFD): This status flag indicates the occurrence of a channel 0 TGRD register compare-match.

  This bit is reserved in channels 1 and 2. It is always read as 0. The write value should always be 0.

Bit 3: TGFD	Description
0	Clear condition: With TGFD = 1, a 0 write to TGFD following a read (Initial value)
1	Set condition: When TCNT = TGRD while TGRD is functioning as an output compare register

- Bit 2—Input Capture/Output Compare Flag C (TGFC): This status flag indicates the occurrence of a Channel 0 TGRC register input capture or compare-match.

  This bit is reserved for channels 1 and 2. It is always read as 0. The write value should always be 0.

Bit 2: TGFC	Description
0	Clear condition: With TGFC = 1, a 0 write to TGFC following a read (Initial value)
1	Set conditions: • When TGRC is functioning as an output compare register (TCNT = TGRC) • When TGRC is functioning as input capture (the TCNT value is sent to TGRC by the input capture signal)

- Bit 1—Output Compare Flag B (TGFB): This status flag indicates the occurrence of a TGRB register compare-match.

Bit 1: TGFB	Description
0	Clear condition: With TGFB = 1, a 0 write to TGFB following a read (Initial value)
1	Set conditions: When TGRB is functioning as an output compare register (TCNT = TGRB)

- Bit 0—Input Capture/Output Compare Flag A (TGFA): This status flag indicates the occurrence of a TGRA register input capture or compare-match.

Bit 0: TGFA	Description
0	Clear condition: With TGFA = 1, a 0 write to TGFA following a read (Initial value)
1	Set conditions: • When TGRA is functioning as an output compare register (TCNT = TGRA) • When TGRA is functioning as input capture (the TCNT value is sent to TGRA by the input capture signal)

8.2.6 Timer Counters (TCNT)

The timer counters (TCNT) are 16-bit counters, with one for each channel, for a total of three.

The TCNT are initialized to H'0000 by a power-on reset. Accessing the TCNT counters in 8-bit units is prohibited. Always access in 16-bit units.

Channel 0: TCNT0
Channel 1: TCNT1
Channel 2: TCNT2

Bit:	15	14	13	12	11	10	9	8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

8.2.7 Timer General Register (TGR)

Each timer general register (TGR) is a 16-bit register that can function as either an output compare register or an input capture register. There are a total of eight TGR, four each for channels 0 and two each for channels 1 and 2. The TGRC and TGRD of channels 0 can be set to operate as buffer registers. The TGR register and buffer register combinations are TGRA with TGRC, and TGRB with TGRD.

The TGRs are initialized to H'FFFF by a power-on reset. Accessing of the TGRs in 8-bit units is disabled; they may only be accessed in 16-bit units.

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

8.2.8 Timer Start Register (TSTR)

The timer start register (TSTR) is an 8-bit read/write register that starts and stops the timer counters (TCNT) of channels 0 to 2. TSTR is initialized to H'00 upon power-on reset.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	CST2	CST1	CST0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

- Bits 7 to 3—Reserved: These bits are always read as 0. The write value should always be 0.
- Bits 2 to 0—Counter Start 2 to 0 (CST2 to CST0): Select starting and stopping of the timer counters (TCNT). The corresponding between bits and channels is as follows:

  CST2: Channel 2 (TCNT2)
  CST1: Channel 1 (TCNT1)
  CST0: Channel 0 (TCNT0)

Bit n: CSTn	Description	
0	TCNTn count is halted	(Initial value)
1	TCNTn counts	

Note: n = 2 to 0.

If 0 is written to a CST bit during operation with the TIOC pin in the output state, the counter stops, but the TIOC pin output compare output level is maintained. If a write is performed on the TIOR register while a CST bit is 0, the pin output level is updated to the set initial output value.

8.2.9 Timer Synchro Register (TSYR)

The timer synchro register (TSYR) is an 8-bit read/write register that selects independent or synchronous TCNT counter operation for channels 0 to 2. Channels for which 1 is set in the corresponding bit will be synchronized. TSYR is initialized to H'00 upon power-on reset.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	SYNC2	SYNC1	SYNC0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

- Bits 7 to 3—Reserved: These bits are always read as 0. The write value should always be 0.
- Bits 2 to 0—Timer Synchronization 2 to 0 (SYNC2 to SYNC0): Selects operation independent of, or synchronized to, other channels. Synchronous operation allows synchronous clears due to multiple TCNT synchronous presets and other channel counter clears. A minimum of two channels must have SYNC bits set to 1 for synchronous operation. For synchronization clearing, it is necessary to set the TCNT counter clear sources (the CCLR2 to CCLR0 bits of the TCR register), in addition to the SYNC bit. The counter start to channel and bit-to-channel correspondence are indicated in the tables below.

  SYNC2: Channel 2 (TCNT2)
  SYNC1: Channel 1 (TCNT1)
  SYNC0: Channel 0 (TCNT0)

Bit n: SYNCn	Description
0	Timer counter (TCNTn) independent operation (Initial value) (TCNTn preset/clear unrelated to other channels)
1	Timer counter synchronous operation*[1] TCNTn synchronous preset/ synchronous clear*[2] possible

Note: 1. Minimum of two channel SYNC bits must be set to 1 for synchronous operation.
2. TCNT counter clear sources (CCLR2 to CCLR0 bits of the TCR register) must be set in addition to the SYNC bit in order to have clear synchronization.
n = 2 to 0.

8.3 Bus Master Interface

8.3.1 16-Bit Registers

The timer counters (TCNT) and general registers (TGR) are 16-bit registers. A 16-bit data bus to the bus master enables 16-bit read/writes. 8-bit read/write is not possible. Always access in 16-bit units. Figure 8.2 shows an example of 16-bit register access operation.

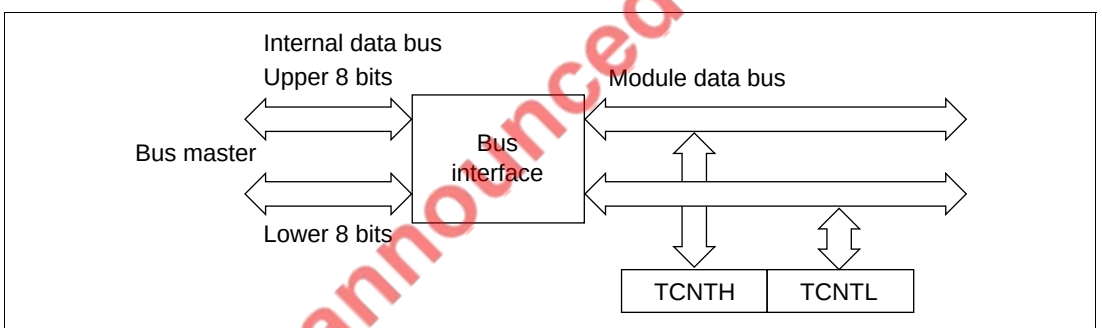


Figure 8.2 16-Bit Register Access Operation (Bus Master ↔ TCNT (16 Bits))

8.3.2 8-Bit Registers

All registers other than the TCNT and general registers (TGR) are 8-bit registers. These are connected to the CPU by a 16-bit data bus, so 16-bit read/writes and as 8-bit read/writes are both possible (figure 8.3 to figure 8.5).

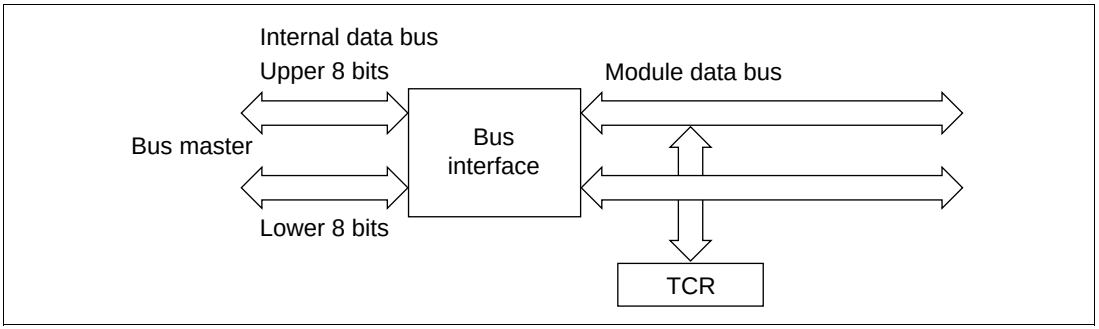


Figure 8.3 8-Bit Register Access Operation (Bus Master ↔ TCR (Upper 8 Bits))

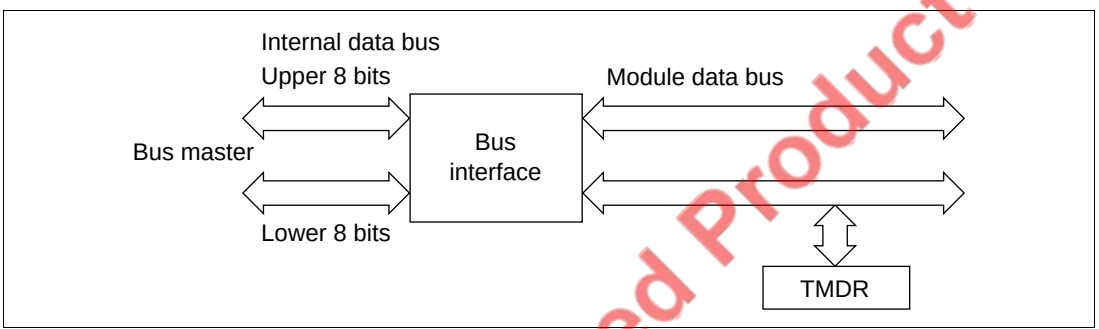


Figure 8.4 8-Bit Register Access Operation (Bus Master ↔ TMDR (Lower 8 Bits))

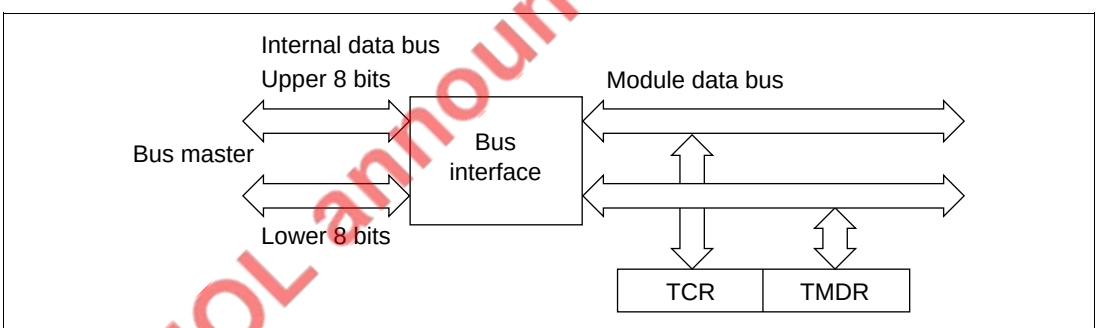


Figure 8.5 8-Bit Register Access Operation (Bus Master ↔ TCR, TMDR (16 Bits))

8.4 Operation

8.4.1 Overview

The operation modes are described below.

Ordinary Operation: Each channel has a timer counter (TCNT) and general register (TGR). The TCNT is an upcounter and can also operate as a free-running counter, periodic counter or external event counter. General registers (TGR) can be used as output compare registers or input capture registers.

Synchronized Operation: The TCNT of a channel set for synchronized operation does a synchronized preset. When any TCNT of a channel operating in the synchronized mode is rewritten, the TCNTs of other channels are simultaneously rewritten as well. The timer synchronization bits of the TSYR registers of multiple channels set for synchronous operation can be set to clear the TCNTs simultaneously.

Buffer Operation: When TGR is an output compare register, the buffer register value of the corresponding channel is transferred to the TGR when a compare-match occurs. When TGR is an input capture register, the TCNT counter value is transferred to the TGR when an input capture occur simultaneously the value previously stored in the TGR is transferred to the buffer register.

Cascade Connection Operation: The channel 1 and channel 2 counters (TCNT1 and TCNT2) can be connected together to operate as a 32-bit counter.

PWM Mode: In PWM mode, a PWM waveform is output. The output level can be set by the TIOR register. Each TGR can be set for PWM waveform output with a duty cycle between 0% and 100%.

8.4.2 Basic Functions

Always select MTU external pin set function using the pin function controller (PFC).

Counter Operation: When a start bit (CST0 to CST2) in the timer start register (TSTR) is set to 1, the corresponding timer counter (TCNT) starts counting. There are two counting modes: a free-running mode and a periodic mode.

To select the counting operation (figure 8.6):

1. Set bits TPSC2 to TPSC0 in the TCR to select the counter clock. At the same time, set bits CKEG1 and CKEG0 in the TCR to select the desired edge of the input clock.
2. To operate as a periodic counter, set the CCLR2 to CCLR0 bits in the TCR to select TGR as a clearing source for the TCNT.

3. Set the TGR selected in step 2 as an output compare register using the timer I/O control register (TIOR).
4. Write the desired cycle value in the TGR selected in step 2.
5. Set the CST bit in the TSTR to 1 to start counting.

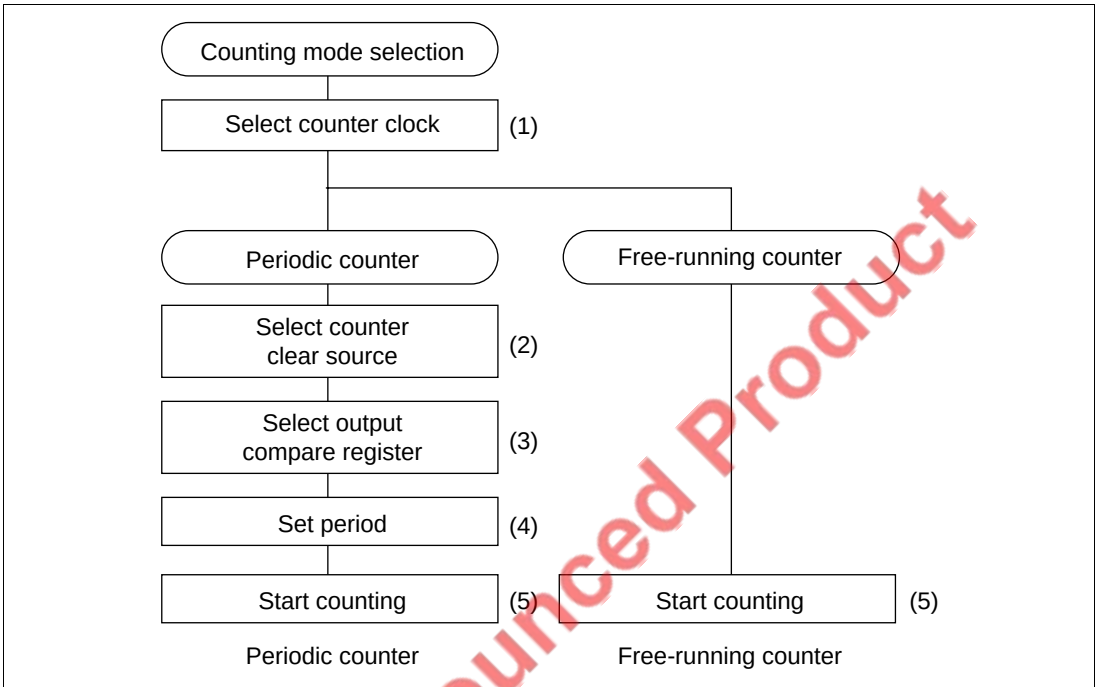


Figure 8.6 Procedure for Selecting the Counting Operation

Free-Running Counter Operation Example: A reset of the MTU timer counters (TCNT) leaves them all in the free-running mode. When a bit in the TSTR is set to 1, the corresponding timer counter operates as a free-running counter and begins to increment. When the count overflows from H'FFFF to H'0000, the TCFV bit in the timer status register (TSR) is set to 1. If the TCIEV bit in the timer's corresponding timer interrupt enable register (TIER) is set to 1, the MTU will make an interrupt request to the interrupt controller. After the TCNT overflows, counting continues from H'0000. Figure 8.7 shows an example of free-running counter operation.

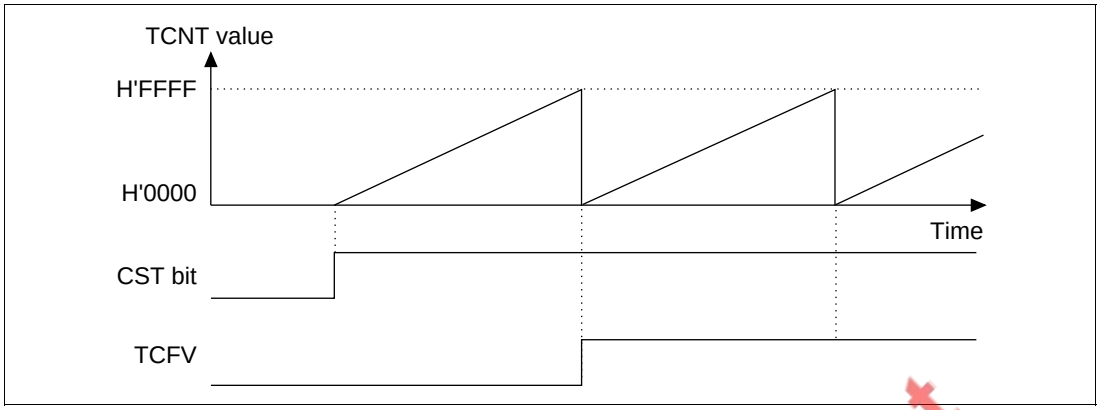


Figure 8.7 Free-Running Counter Operation

Periodic Counter Operation Example: Periodic counter operation is obtained for a given channel's TCNT by selecting compare-match as a TCNT clear source. Set the TGR register for period setting to output compare register and select counter clear upon compare-match using the CCLR2 to CCLR0 bits of the timer control register (TCR). After these settings, the TCNT begins incrementing as a periodic counter when the corresponding bit of TSTR is set to 1. When the count matches the TGR register value, the TGF bit in the TSR is set to 1 and the counter is cleared to H'0000. If the TGIE bit of the corresponding TIER is set to 1 at this point, the MTU will make an interrupt request to the interrupt controller. After the compare-match, TCNT continues counting from H'0000. Figure 8.8 shows an example of periodic counting.

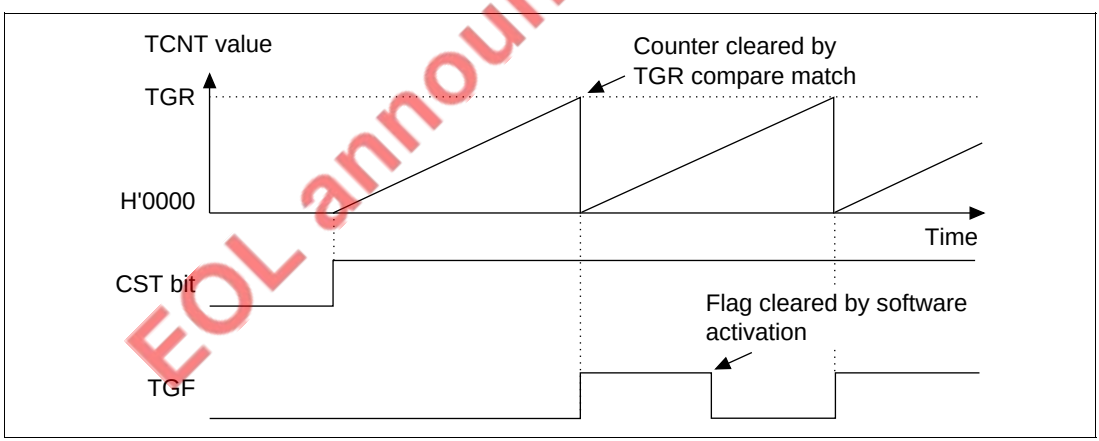


Figure 8.8 Periodic Counter Operation

Compare-Match Waveform Output Function: The MTU can output 0 level, 1 level, or toggle output from the corresponding output pins upon compare-matches.

Procedure for selecting the compare-match waveform output operation (figure 8.9):

1. Set the TIOR to select 0 output or 1 output for the initial value, and 0 output, 1 output, or toggle output for compare-match output. The TIOC pin will output the set initial value until the first compare-match occurs.
2. Set a value in the TGR to select the compare-match timing.
3. Set the CST bit in the TSTR to 1 to start counting.

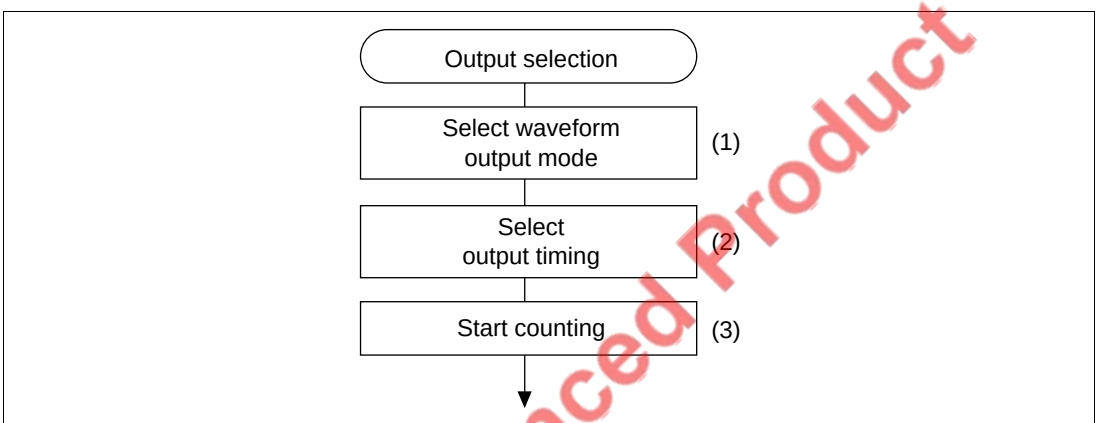


Figure 8.9 Procedure for Selecting Compare Match Waveform Output Operation

Waveform Output Operation (0 Output/1 Output): Figure 8.10 shows 0 output/1 output. In the example, TCNT is a free-running counter, 1 is output upon compare-match A and 0 is output upon compare-match B. When the pin level matches the set level, the pin level does not change.

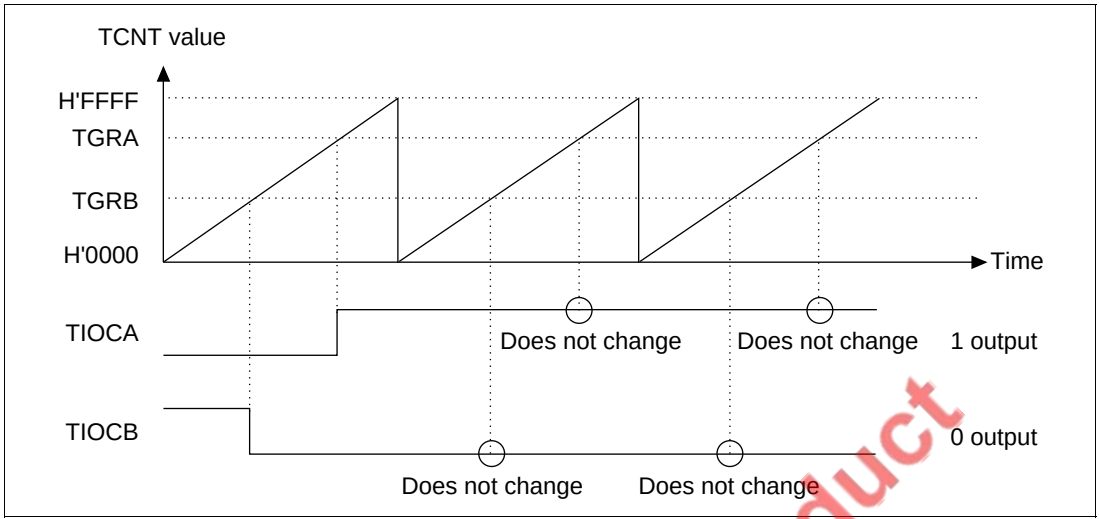


Figure 8.10 Example of 0 Output/1 Output

Waveform Output Operation (Toggle Output): Figure 8.11 shows the toggle output. In the example, the TCNT operates as a periodic counter cleared by compare-match B, with toggle output at both compare-match A and compare-match B.

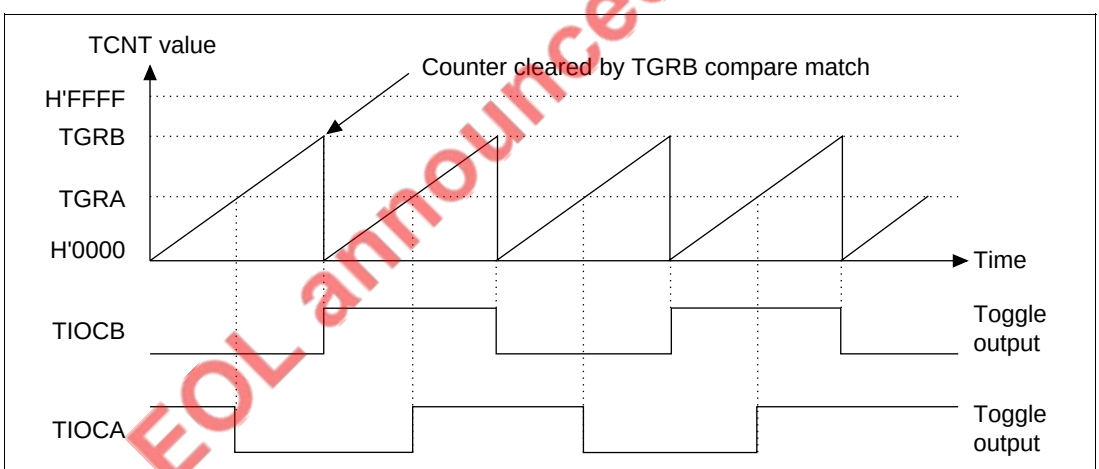


Figure 8.11 Example of Toggle Output

Input Capture Function: In the input capture mode, the TCNT value is transferred into the TGR register when the input edge is detected at the input capture/output compare pin (TIOC).

Detection can take place on the rising edge, falling edge, or both edges. Channels 0 and 1 can use other channel counter input clocks or compare-match signals as input capture sources.

The procedure for selecting the input capture operation (figure 8.12) is:

1. Set the TIOR to select the input capture function of the TGR, then select the input capture source, and rising edge, falling edge, or both edges as the input edge.
2. Set the CST bit in the TSTR to 1 to start the TCNT counting.

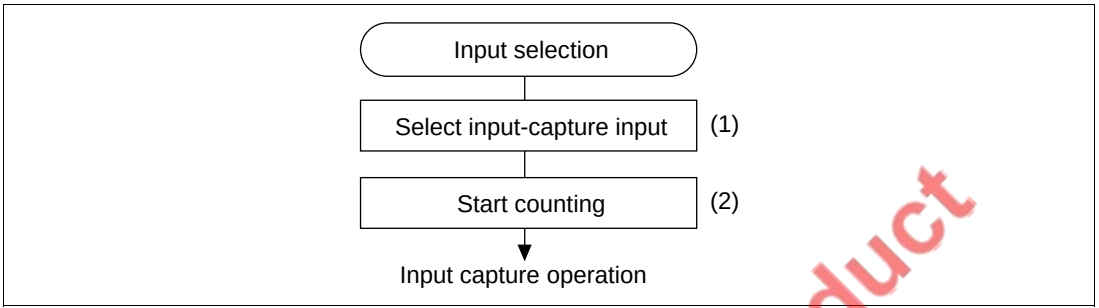


Figure 8.12 Procedure for Selecting Input Capture Operation

Input Capture Operation: Figure 8.13 shows input capture. The falling edge of TIOCB and both edges of TIOCA are selected as input capture input edges. In the example, TCNT is set to clear at the input capture of the TGRB register.

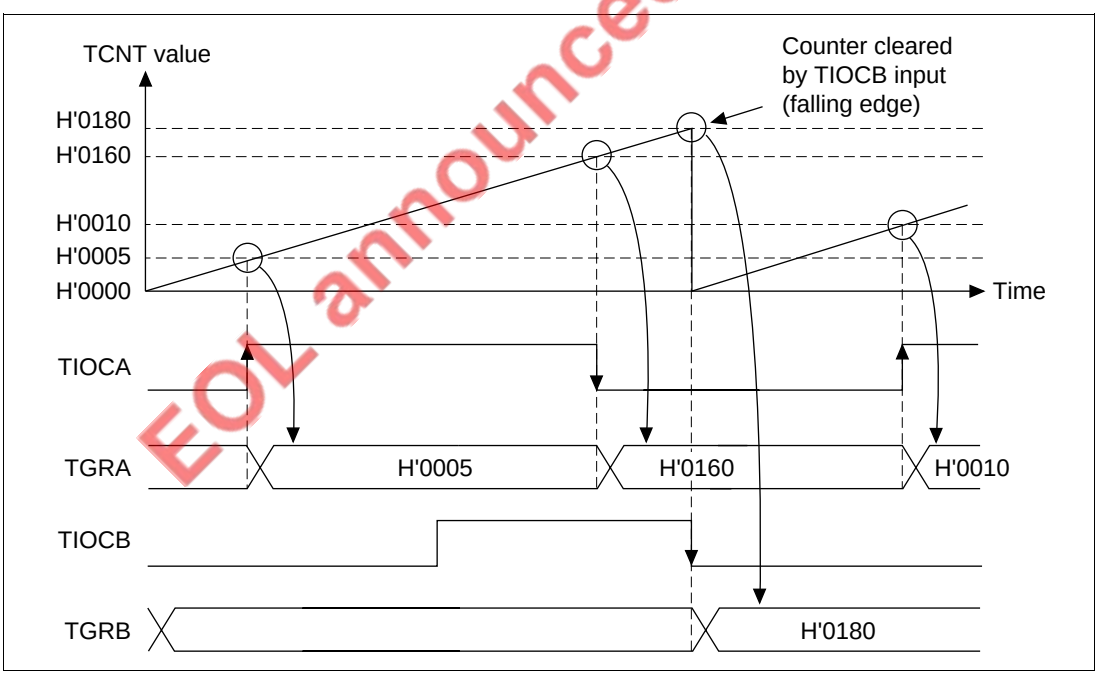


Figure 8.13 Input Capture Operation

8.4.3 Synchronous Operation

There are two kinds of synchronous operation, synchronized preset and synchronized clear. The synchronized preset operation allows multiple timer counters (TCNT) to be rewritten simultaneously, while the synchronized clear operation allows multiple TCNT counters to be cleared simultaneously using timer control register (TCR) settings.

The synchronizing mode can increase the number of TGR registers for a single time base. All five channels can be set for synchronous operation.

Procedure for Selecting the Synchronizing Mode (Figure 8.14):

1. Set 1 in the SYNC bit of the timer synchro register (TSYR) to use the corresponding channel in the synchronizing mode.
2. When a value is written in the TCNT in any of the synchronized channels, the same value is simultaneously written in the TCNT in the other channels.
3. Set the counter to clear with output compare/input capture using bits CCLR2 to CCLR0 in the TCR.
4. Set the counter clear source to synchronized clear using the CCLR2 to CCLR0 bits of the TCR.
5. Set the CST bits for the corresponding channels in the TSTR to 1 to start counting in the TCNT.

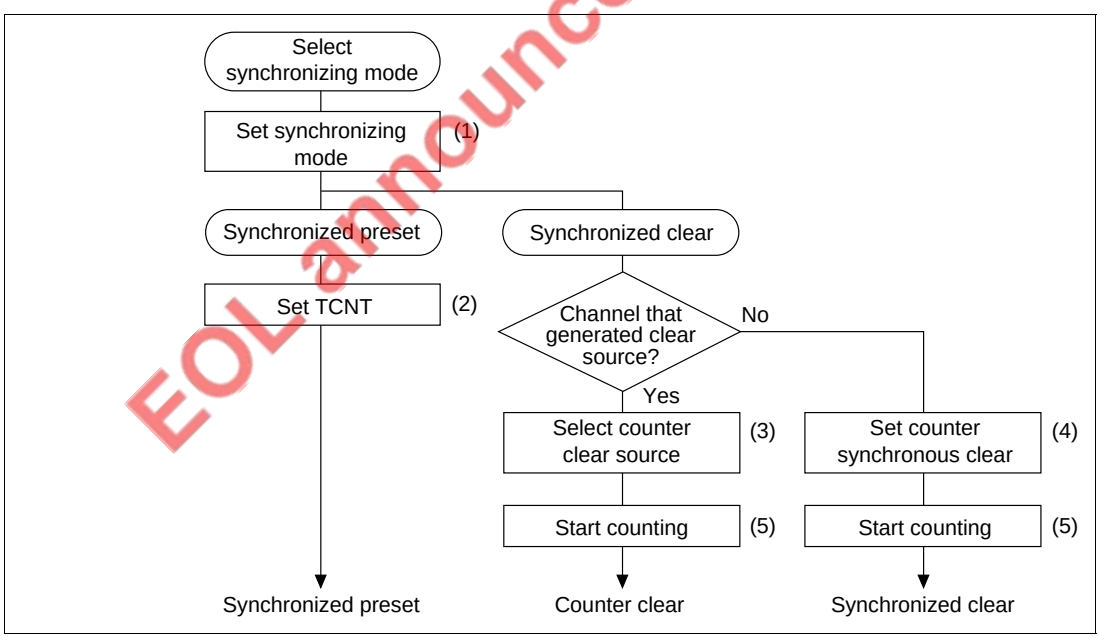


Figure 8.14 Procedure for Selecting Synchronizing Operation

Synchronized Operation: Figure 8.15 shows an example of synchronized operation. Channels 0, 1, and 2 are set to synchronized operation and PWM mode 1. Channel 0 is set for a counter clear upon compare-match with TGR0B. Channels 1 and 2 are set for synchronous counter clears by synchronous presets and TGR0B register compare-matches. Accordingly, a three-phase PWM waveform with the data set in the TGR0B register as its PWM period is output from the TIOC0A, TIOC1A, and TIOC2A pins.

See section 8.4.6, PWM Mode, for details on the PWM mode.

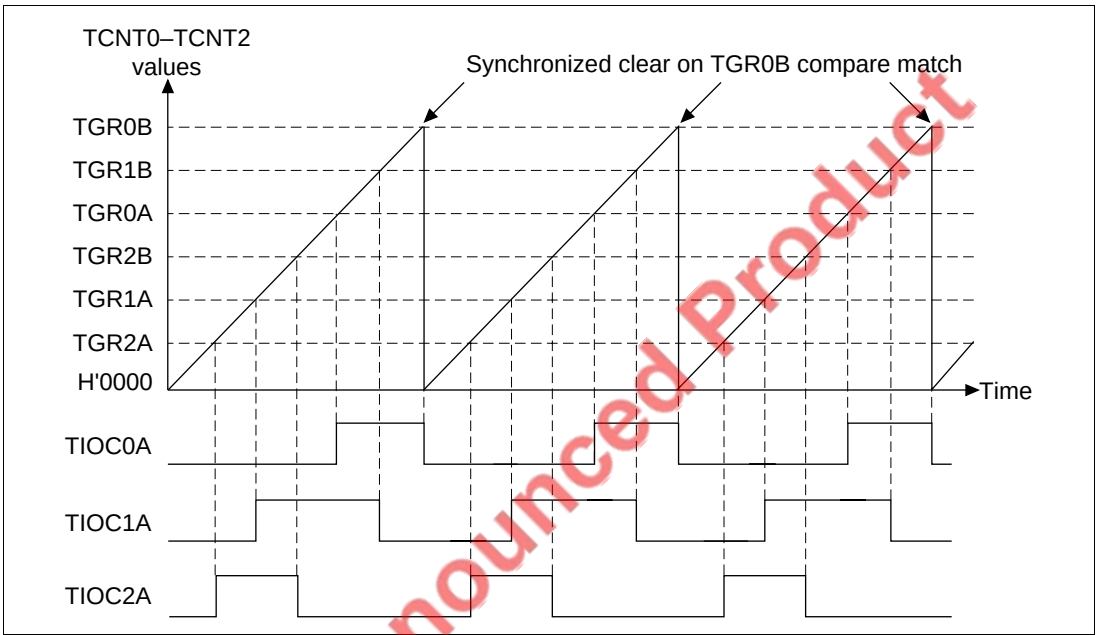


Figure 8.15 Synchronized Operation Example

8.4.4 Buffer Operation

Buffer operation is a function of channel 0. TGRC and TGRD can be used as buffer registers. Table 8.5 shows the register combinations for buffer operation.

Table 8.5 Register Combinations

Channel	General Register	Buffer Register
0	TGR0A	TGR0C
	TGR0B	TGR0D

The buffer operation differs, depending on whether the TGR has been set as an input capture register or an output compare register.

When TGR Is an Output Compare Register: When a compare-match occurs, the corresponding channel buffer register value is transferred to the general register. Figure 8.16 shows an example.

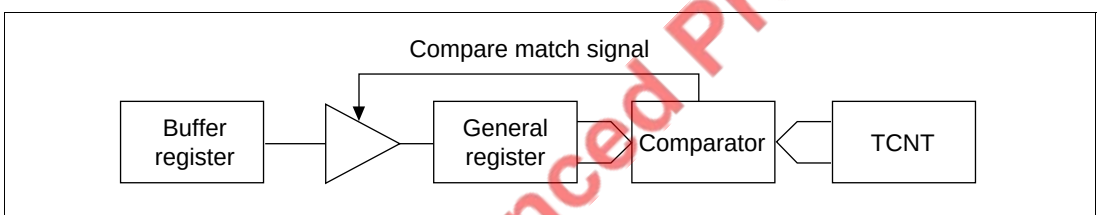


Figure 8.16 Compare Match Buffer Operation

When TGR Is an Input Capture Register: When an input capture occurs, the timer counter (TCNT) value is transferred to the general register (TGR), and the value that had been held up to that time in the TGR is transferred to the buffer register (figure 8.17).

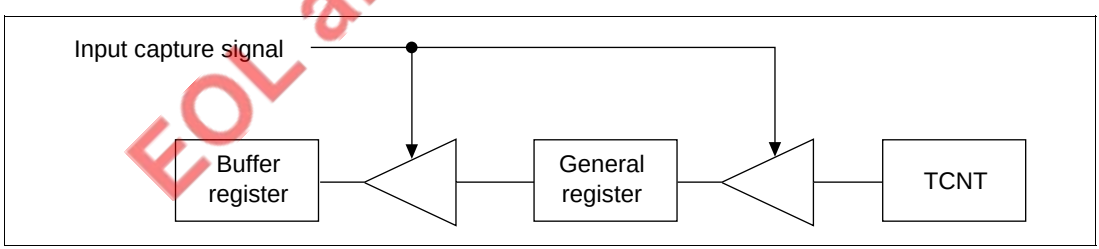


Figure 8.17 Input Capture Buffer Operation

Procedure for Setting Buffer Mode (Figure 8.18):

1. Use the timer I/O control register (TIOR) to set the TGR as either an input capture or output compare register.
2. Use the timer mode register (TMDR) BFA, and BFB bits to set the TGR for buffer mode.
3. Set the CST bit in the TSTR to 1 to start the count operation.

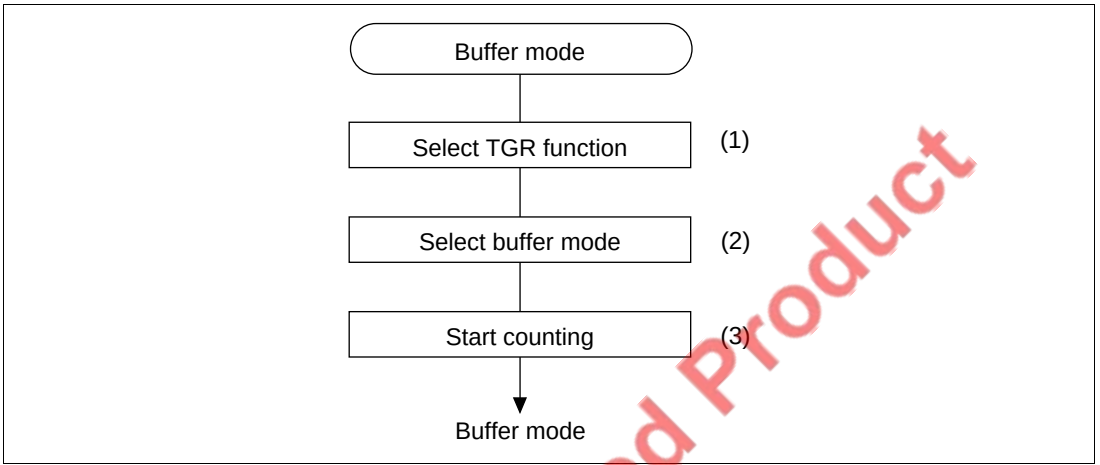


Figure 8.18 Buffer Operation Setting Procedure

Buffer Operation Examples—when TGR Is an Output Compare Register: Figure 8.19 shows an example of channel 0 set to PWM mode 1, and the TGRA and TGRC registers set for buffer operation.

The TCNT counter is cleared by a compare-match B, and the output is a 1 upon compare-match A and 0 output upon compare-match B. Because buffer mode is selected, a compare-match A changes the output, and the buffer register TGRC value is simultaneously transferred to the general register TGRA. This operation is repeated with each occurrence of a compare-match A.

See section 8.4.6, PWM Mode, for details on the PWM mode.

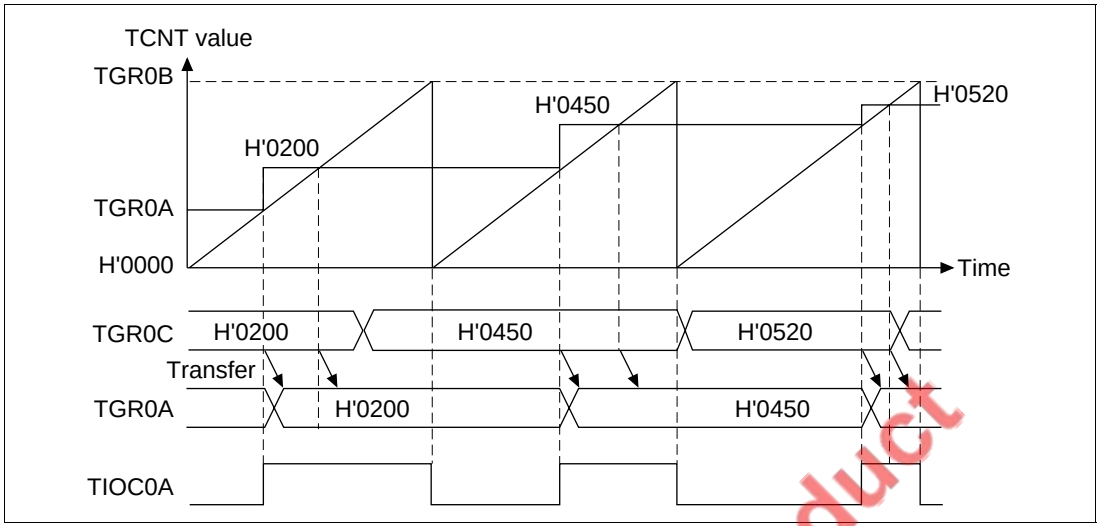


Figure 8.19 Buffer Operation Example (Output Compare Register)

Buffer Operation Examples—when TGR Is an Input Capture Register: Figure 8.20 shows an example of TGRA set as an input capture register with the TGRA and TGRB registers set for buffer operation.

The TCNT counter is cleared by a TGRA register input capture, and the TIOCA pin input capture input edge is selected as both rising and falling edge. Because buffer mode is selected, an input capture A causes the TCNT counter value to be stored in the TGRA register, and the value that was stored in the TGRA up until that time is simultaneously transferred to the TGRC register.

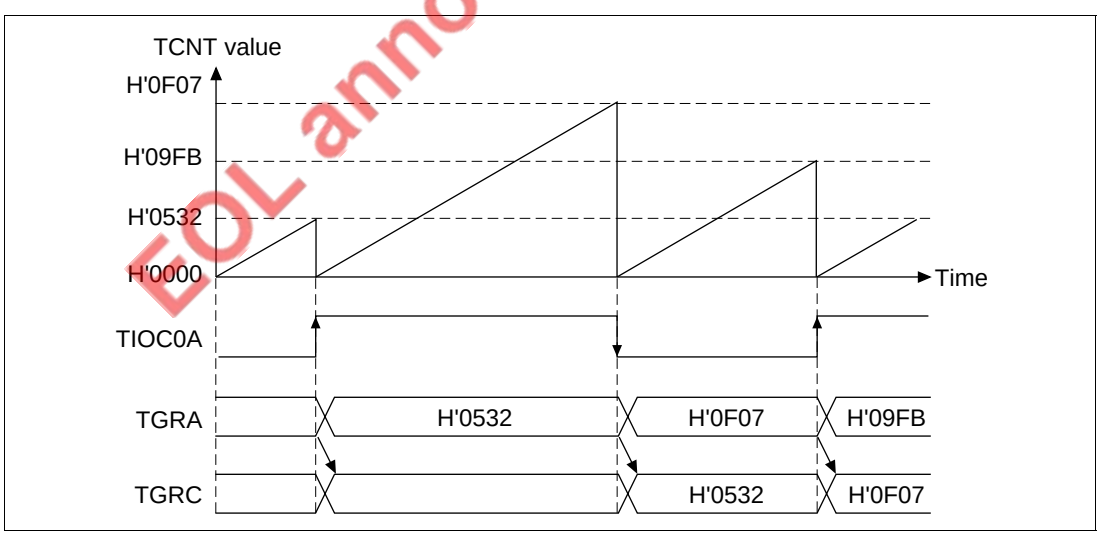


Figure 8.20 Buffer Operation Example (Input Capture Register)

8.4.5 Cascade Connection Mode

Cascade connection mode is a function that connects the 16-bit counters of two channels together to act as a 32-bit counter.

This function operates by using the TPSC2 to TPSC0 bits of the TCR register to set the channel 1 counter clock to count by TCNT2 counter overflow.

Table 8.6 shows the cascade connection combinations.

Table 8.6 Cascade Connection Combinations

Combination	Upper 16 Bits	Lower 16 Bits
Channel 1, channel 2	TCNT1	TCNT2

Procedure for Setting Cascade Connection Mode (Figure 8.21):

1. Set the TPSC2 to TPSC 0 bits of the channel 1 timer control register (TCR) to B'111 to select "count by TCNT2 overflow."
2. Set the CST bits corresponding to the upper and lower 16 bits in the TSTR to 1 to start the count operation.

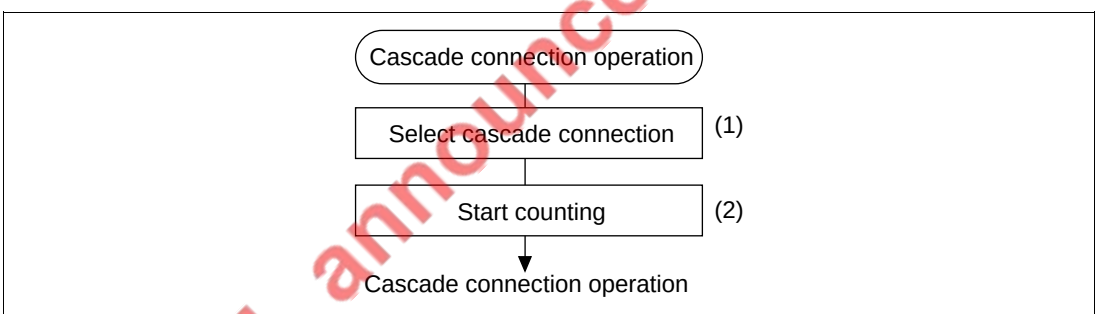


Figure 8.21 Procedure for Selecting Cascade Connection Mode

Cascade Connection Mode Examples—Input Capture: Figure 8.22 shows an example of operation when the TCNT1 counter is set to count on TCNT2 overflow, the TGR1A and TGR2A registers are set as input capture registers, and the TIOC pin rising edge is selected.

Through simultaneous input of the rising edge to the TIOC1A and TIOC2A pins, 32-bit data is transferred, with the upper 16 bits to the TGR1A register and the lower 16 bits to the TGR2A register.

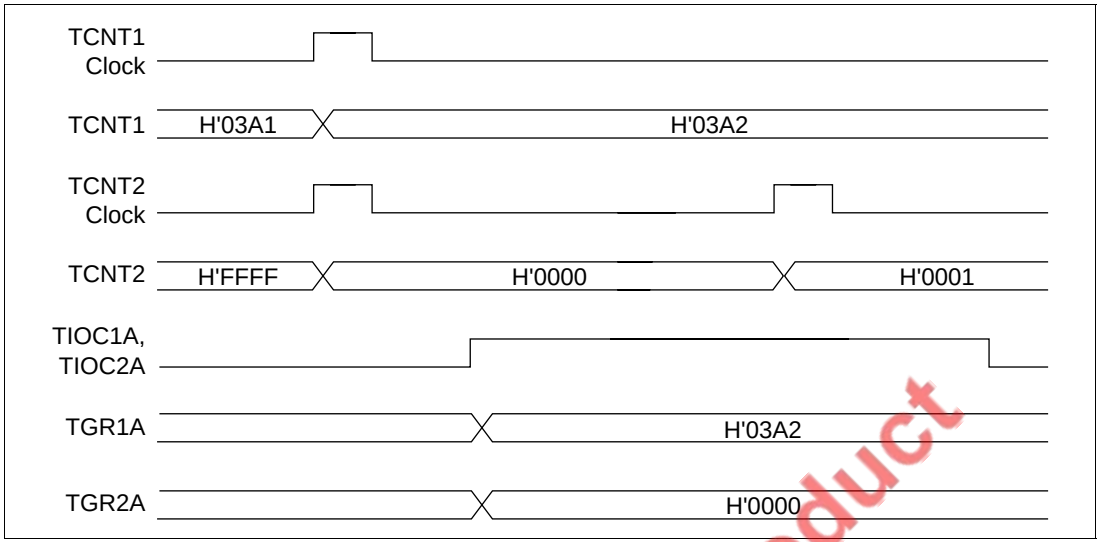


Figure 8.22 Cascade Connection Operation Example (Input Capture)

8.4.6 PWM Mode

PWM mode outputs the various PWM waveforms from output pins. Output levels of 0 output, 1 output, or toggle output can be selected as the output level for the compare-match of each TGR.

A period can be set for a register by using the TGR compare-match as a counter clear source. All five channels can be independently set to PWM mode. Synchronous operation is also possible.

There are two PWM modes:

- PWM mode 1

  Generates PWM output using the TGRA and TGRB registers, and TGRC and TGRD registers as pairs. The initial output values are those established in the TGRA and TGRC registers. When the values set in TGR registers being used as a pair are equal, output values will not change even if a compare-match occurs.

  A maximum of 4-phase PWM output is possible for PWM mode 1.
- PWM mode 2

  Generates PWM output using one TGR register as a period register and another as a duty cycle register. The output value of each pin upon a counter clear is the initial value established by the TIOR register. When the values set in the period register and duty register are equal, output values will not change even if a compare-match occurs.

Table 8.7 lists the combinations of PWM output pins and registers.

Table 8.7 Combinations of PWM Output Pins and Registers

		Output Pin	
Channel	**Register**	**PWM Mode 1**	**PWM Mode 2**
0 (AB pair)	TGR0A TGR0B	TIOC0A	TIOC 0A
0 (CD pair)	TGR0C TGR0D	TIOC0C	TIOC 0C
1	TGR1A TGR1B	TIOC1A	TIOC 1A TIOC 1B
2	TGR2A TGR2B	TIOC2A	TIOC 2A TIOC 2B

Note: PWM output of the period setting TGR is not possible in PWM mode 2.

Procedure for Selecting the PWM Mode (Figure 8.23):

1. Set bits TPSC2 to TPSC0 in the TCR to select the counter clock source. At the same time, set bits CKEG1 and CKEG0 in the TCR to select the desired edge of the input clock.
2. Set bits CCLR2 to CCLR0 in the TCR to select the TGR to be used as a counter clear source.
3. Set the period in the TGR selected in step 2, and the duty cycle in another TGR.
4. Using the timer I/O control register (TIOR), set the TGR selected in step 3 to act as an output compare register, and select the initial value and output value.
5. Set the MD3 to MD 0 bits in TMDR to select the PWM mode.
6. Set the CST bit in the TSTR to 1 to let the TCNT start counting.

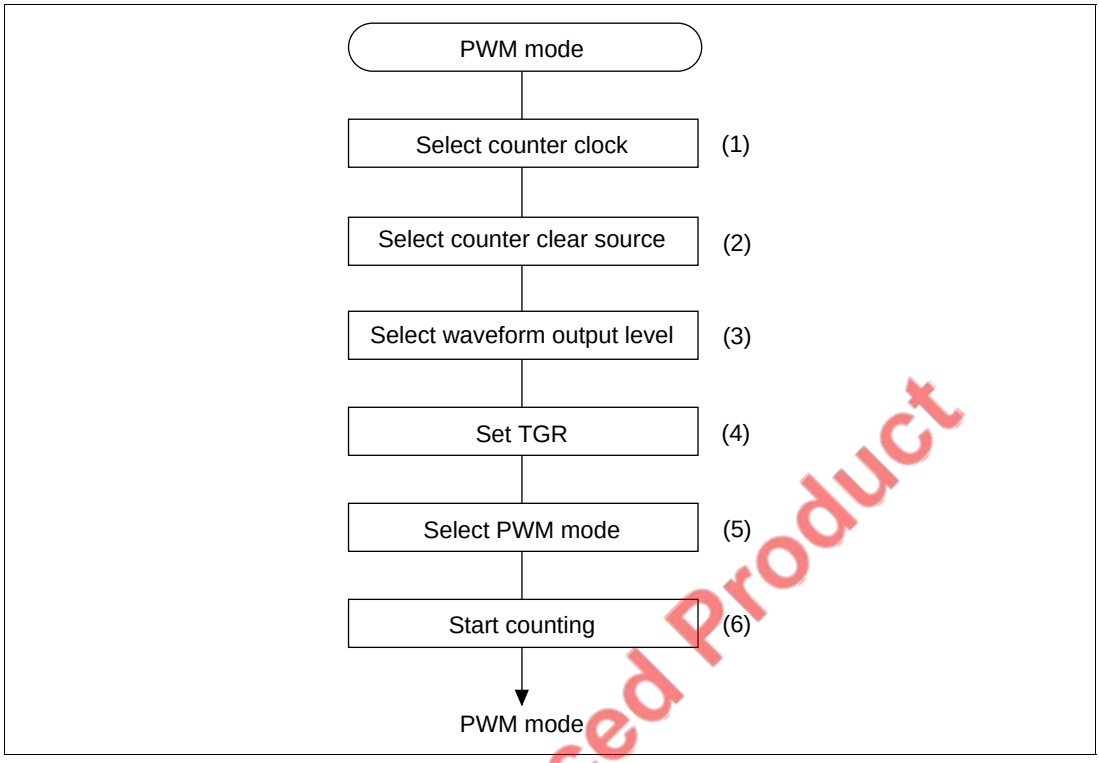


Figure 8.23 Procedure for Selecting the PWM Mode

PWM Mode Operation Examples—PWM Mode 1 (Figure 8.24): A TGRA register compare-match is used as a TCNT counter clear source, the TGRA register initial output value and output compare output value are both 0, and the TGRB register output compare output value is a 1. In this example, the value established in the TGRA register becomes the period and the value established in the TGRB register becomes the duty cycle.

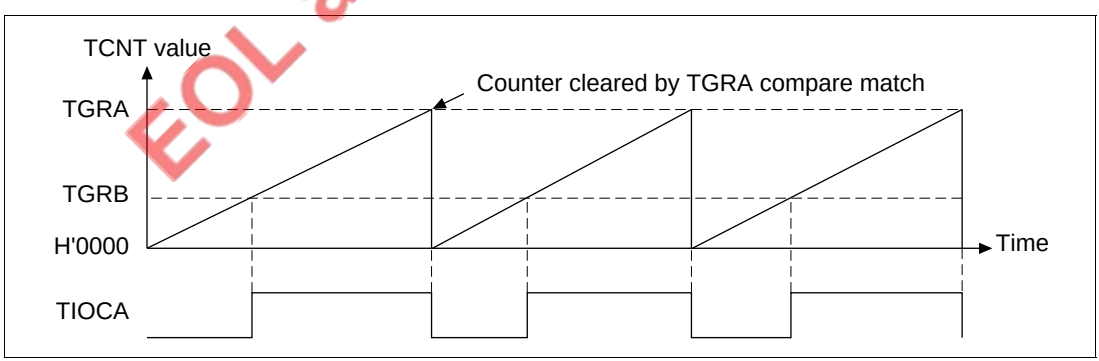


Figure 8.24 PWM Mode Operation Example (Mode 1)

PWM Mode Operation Examples—PWM Mode 2 (Figure 8.25): Channels 0 and 1 are set for synchronous operation, TGR1B register compare-match is used as a TCNT counter clear source, the other TGR register initial output value is 0 and output compare output value is 1, and a 3-phase PWM waveform is output. In this example, the value established in the TGR1B register becomes the period and the value established in the other TGR register becomes the duty cycle.

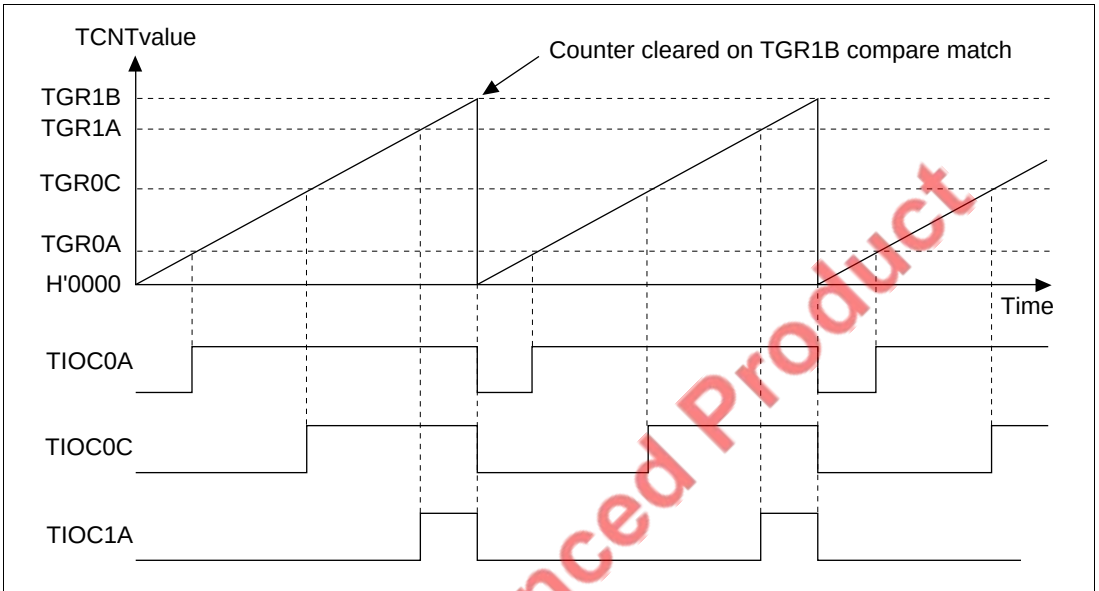


Figure 8.25 PWM Mode Operation Example (Mode 2)

0% Duty Cycle: Figure 8.26 shows an example of a 0% duty cycle PWM waveform output in PWM mode.

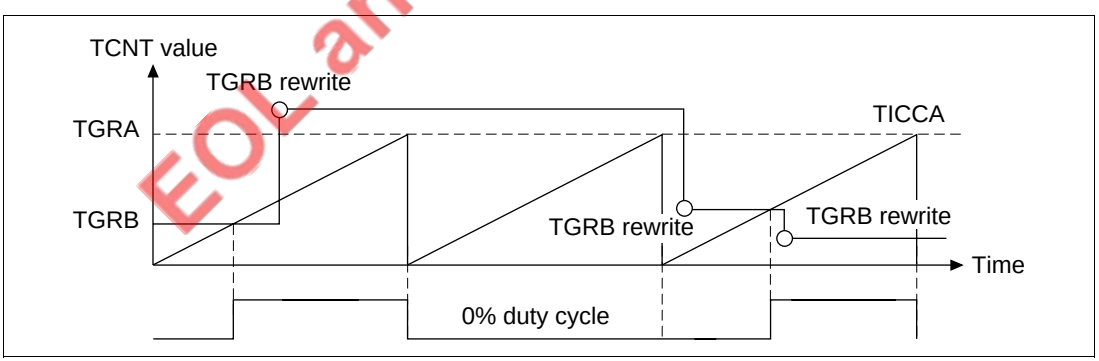


Figure 8.26 PWM Mode Operation Example (0% Duty Cycle)

100% Duty Cycle: Figure 8.27 shows an example of a 100% duty cycle PWM waveform output in PWM mode.

In PWM mode, when setting cycle = duty cycle the output waveform does not change, nor is there a change of waveform for the first pulse immediately after clearing the counter.

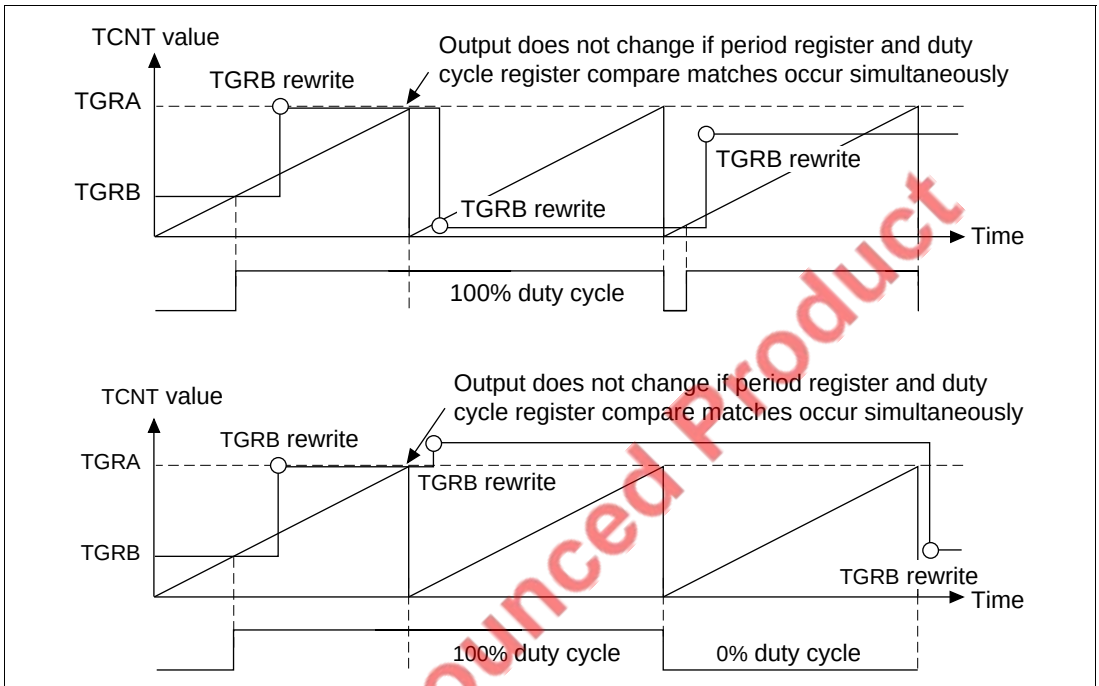


Figure 8.27 PWM Mode Operation Example (100% Duty Cycle)

8.5 Interrupts

8.5.1 Interrupt Sources and Priority Ranking

The MTU has two interrupt sources: TGR register compare-match/input captures, TCNT counter overflows. Because each of these three types of interrupts are allocated its own dedicated status flag and enable/disable bit, the issuing of interrupt request signals to the interrupt controller can be independently enabled or disabled.

When an interrupt source is generated, the corresponding status flag in the timer status register (TSR) is set to 1. If the corresponding enable/disable bit in the timer input enable register (TIER) is set to 1 at this time, the MTU makes an interrupt request of the interrupt controller. The interrupt request is canceled by clearing the status flag to 0.

The channel priority order can be changed with the interrupt controller. The priority ranking within a channel is fixed. For more information, see section 6, Interrupt Controller.

Table 8.8 lists the MTU interrupt sources.

Input Capture/Compare Match Interrupts: If the TGIE bit of the timer input enable register (TIER) is already set to 1 when the TGF flag in the timer status register (TSR) is set to 1 by a TGR register input capture/compare-match of any channel, an interrupt request is sent to the interrupt controller. The interrupt request is canceled by clearing the TGF flag to 0. The MTU has 8 input capture/compare-match interrupts; four each for channel 0, and two each for channels 1 and 2.

Overflow Interrupts: If the TCIEV bit of the TIER is already set to 1 when the TCFV flag in the TSR is set to 1 by a TCNT counter overflow of any channel, an interrupt request is sent to the interrupt controller. The interrupt request is canceled by clearing the TCFV flag to 0. The MTU has three overflow interrupts, one for each channel.

Table 8.8 MTU Interrupt Sources

Channel	Interrupt Source	Description	Priority*
0	TGI0A	TGR0A input capture/compare-match	High
	TGI0B	TGR0B input capture/compare-match	
	TGI0C	TGR0C input capture/compare-match	
	TGI0D	TGR0D input capture/compare-match	
	TCI0V	TCNT0 overflow	
1	TGI1A	TGR1A input capture/compare-match	
	TGI1B	TGR1B input capture/compare-match	
	TCI1V	TCNT1 overflow	
2	TGI2A	TGR2A input capture/compare-match	
	TGI2B	TGR2B input capture/compare-match	
	TCI2V	TCNT2 overflow	Low

Note: * Indicates the initial status following reset. The ranking of channels can be altered using the interrupt controller.

8.5.2 A/D Converter Activation

The TGRA register input capture/compare-match of any channel can be used to activate the on-chip A/D converter.

If the TTGE bit of the TIER is already set to 1 when the TGFA flag in the TSR is set to 1 by a TGRA register input capture/compare-match of any of the channels, an A/D conversion start request is sent to the A/D converter. If the MTU conversion start trigger is selected at such a time on the A/D converter side when this happens, the A/D conversion starts.

The MTU has 3 TGRA register input capture/compare-match interrupts, one for each channel, that can be used as A/D converter activation sources.

8.6 Operation Timing

8.6.1 Input/Output Timing

TCNT Count Timing: Count timing for the TCNT counter with internal clock operation is shown in figure 8.28.

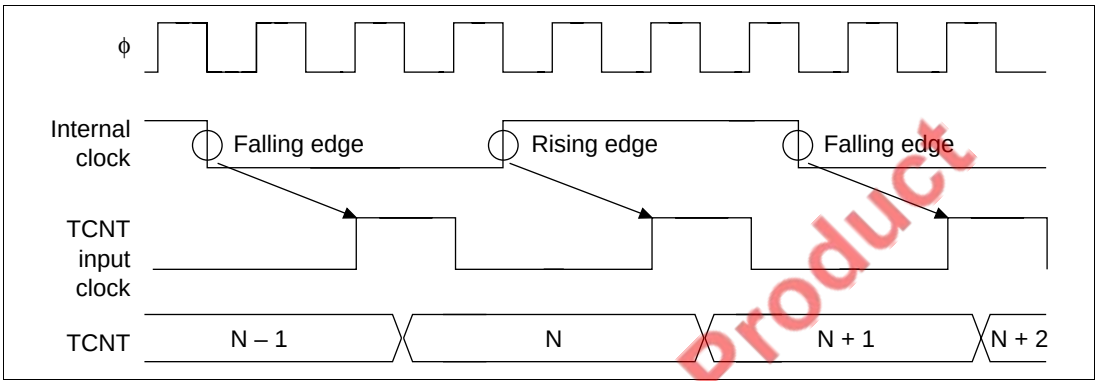


Figure 8.28 TCNT Count Timing during Internal Clock Operation

Output Compare Output Timing: The compare-match signal is generated at the final state of TCNT and TGR matching. When a compare-match signal is issued, the output value set in TIOR or TOCR is output to the output compare output pin (TIOC pin). After TCNT and TGR matching, a compare-match signal is not issued until immediately before the TCNT input clock.

Output compare output timing (normal mode and PWM mode) is shown in figure 8.29.

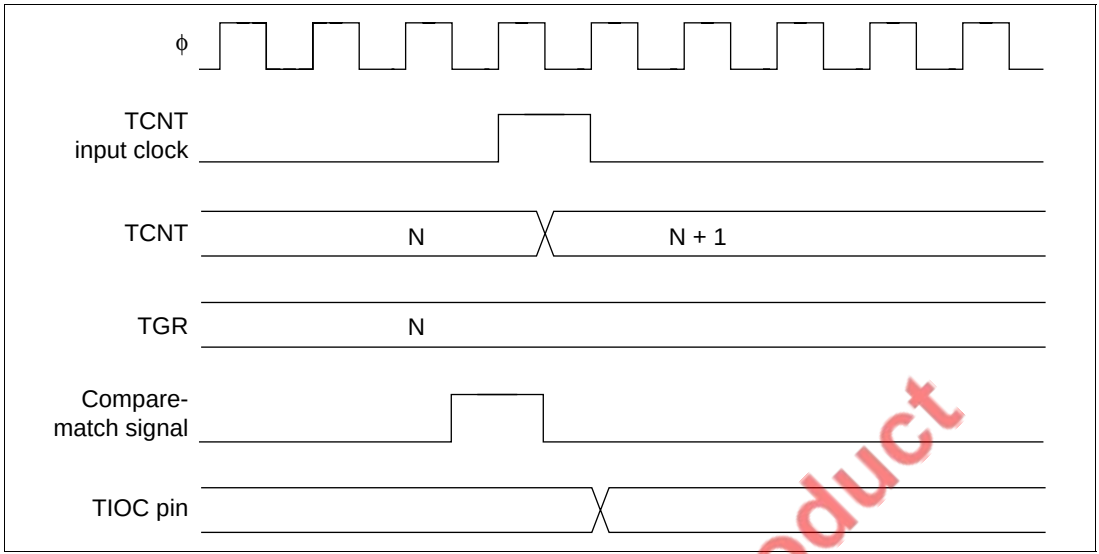


Figure 8.29 Output Compare Output Timing (Normal Mode/PWM Mode)

Input Capture Signal Timing: Figure 8.30 illustrates input capture timing.

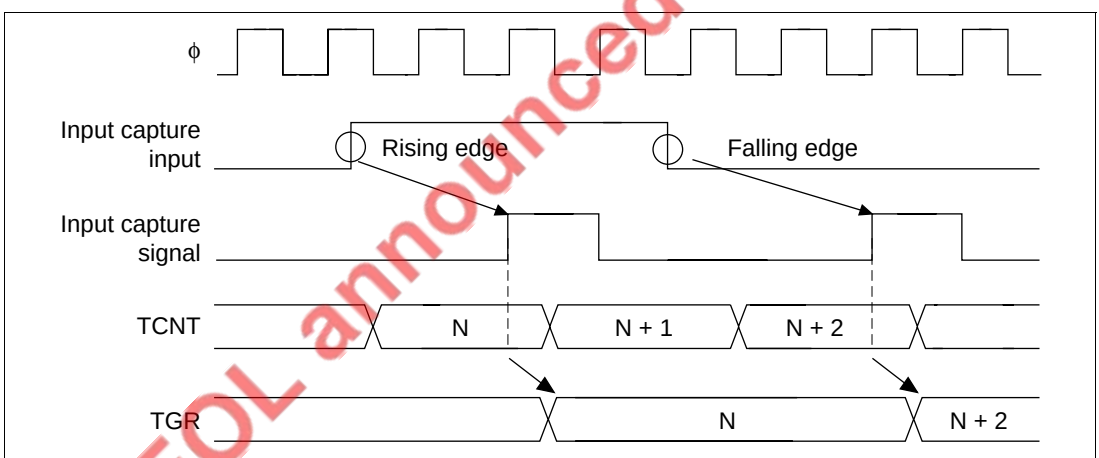


Figure 8.30 Input Capture Input Signal Timing

Counter Clearing Timing Due to Compare-Match/Input Capture: Timing for counter clearing due to compare-match is shown in figure 8.31. Figure 8.32 shows the timing for counter clearing due to input capture.

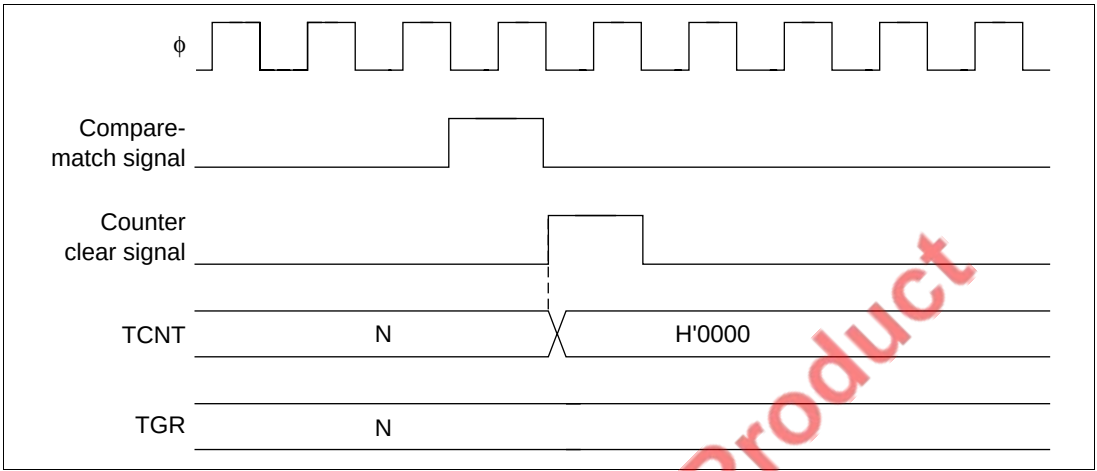


Figure 8.31 Counter Clearing Timing (Compare-Match)

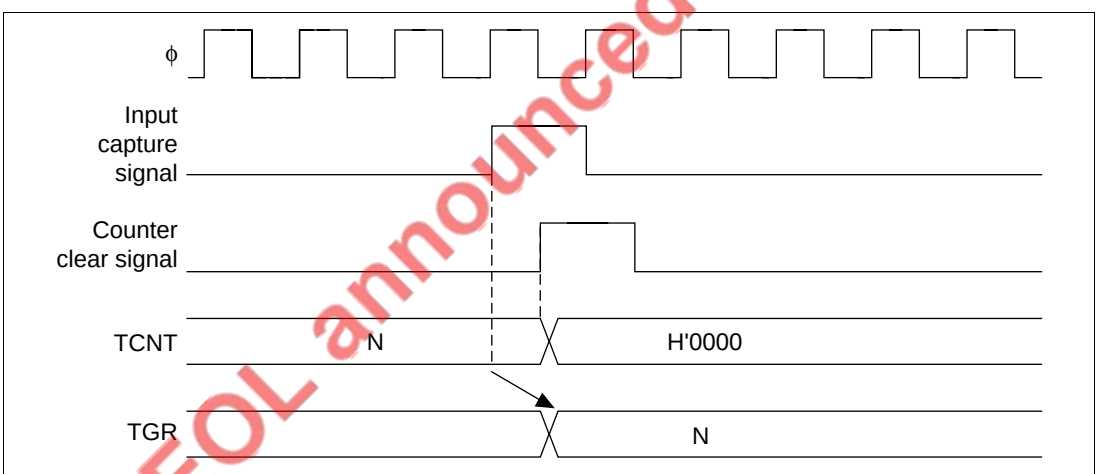


Figure 8.32 Counter Clearing Timing (Input Capture)

Buffer Operation Timing: Compare-match buffer operation timing is shown in figure 8.33. Figure 8.34 shows input capture buffer operation timing.

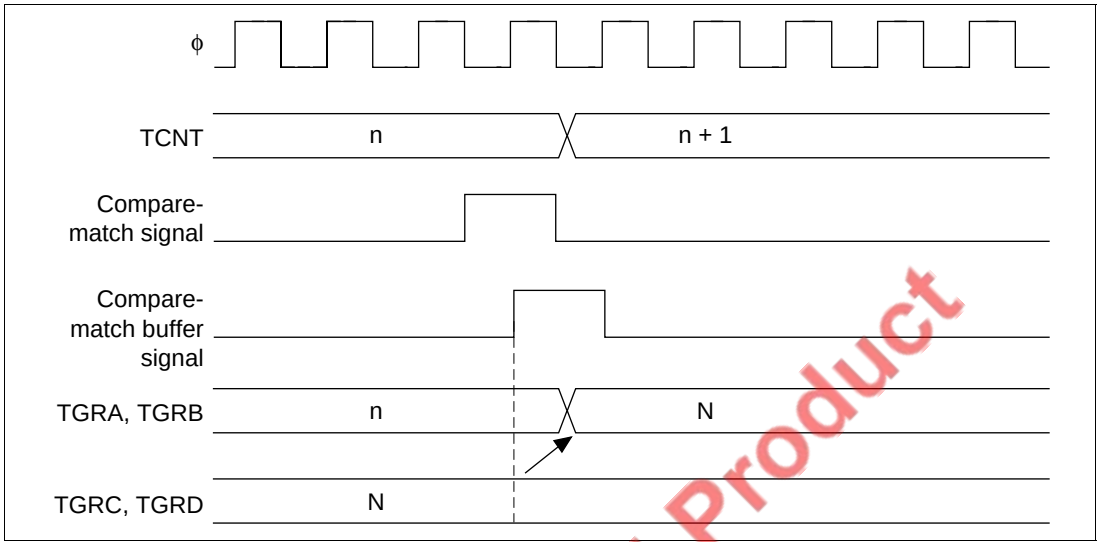


Figure 8.33 Buffer Operation Timing (Compare-Match)

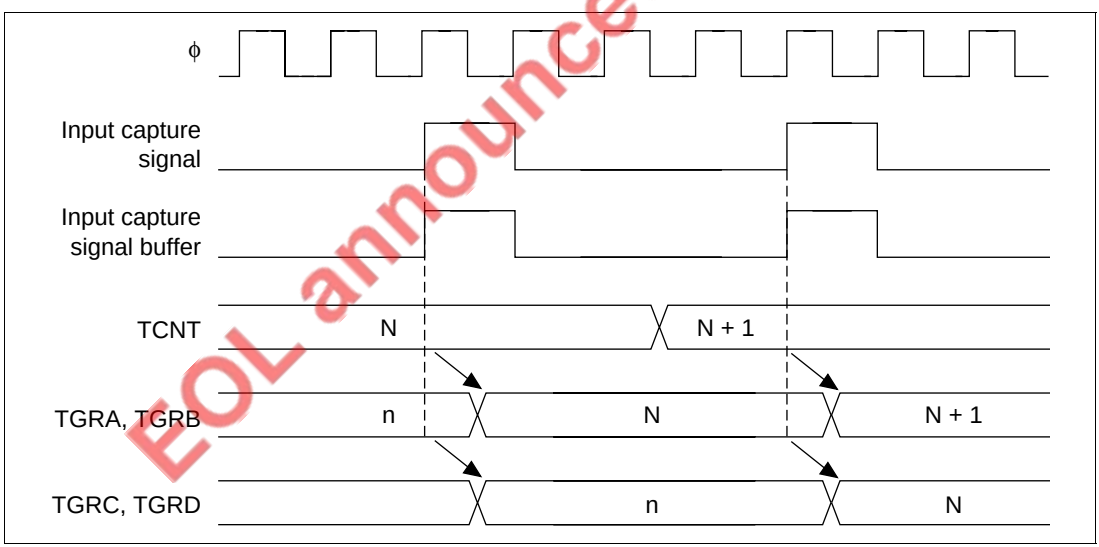


Figure 8.34 Buffer Operation Timing (Input Capture)

8.6.2 Interrupt Signal Timing

Setting TGF Flag Timing during Compare-Match: Figure 8.35 shows timing for the TGF flag of the timer status register (TSR) due to compare-match, as well as TGI interrupt request signal timing.

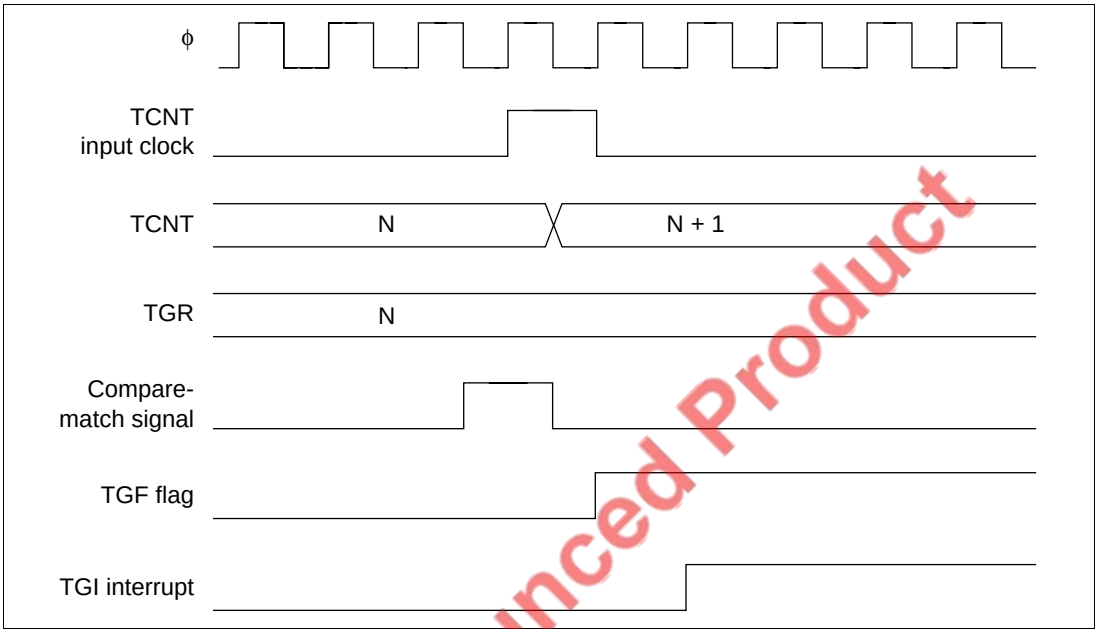


Figure 8.35 TGI Interrupt Timing (Compare Match)

Setting TGF Flag Timing during Input Capture: Figure 8.36 shows timing for the TGF flag of the timer status register (TSR) due to input capture, as well as TGI interrupt request signal timing.

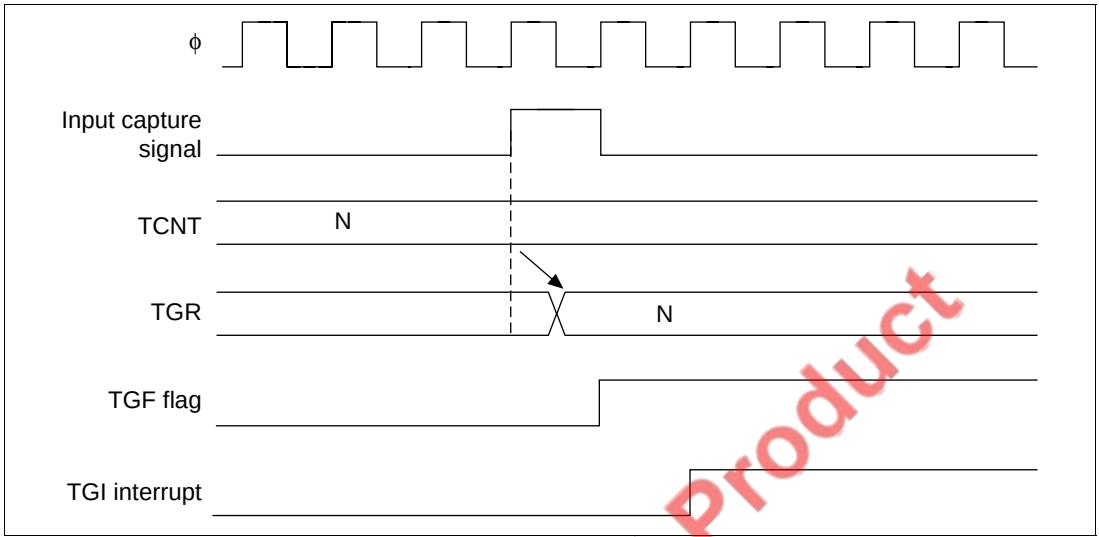


Figure 8.36 TGI Interrupt Timing (Input Capture)

Setting Timing for Overflow Flag (TCFV): Figure 8.37 shows timing for the TCFV flag of the timer status register (TSR) due to overflow, as well as TCIV interrupt request signal timing.

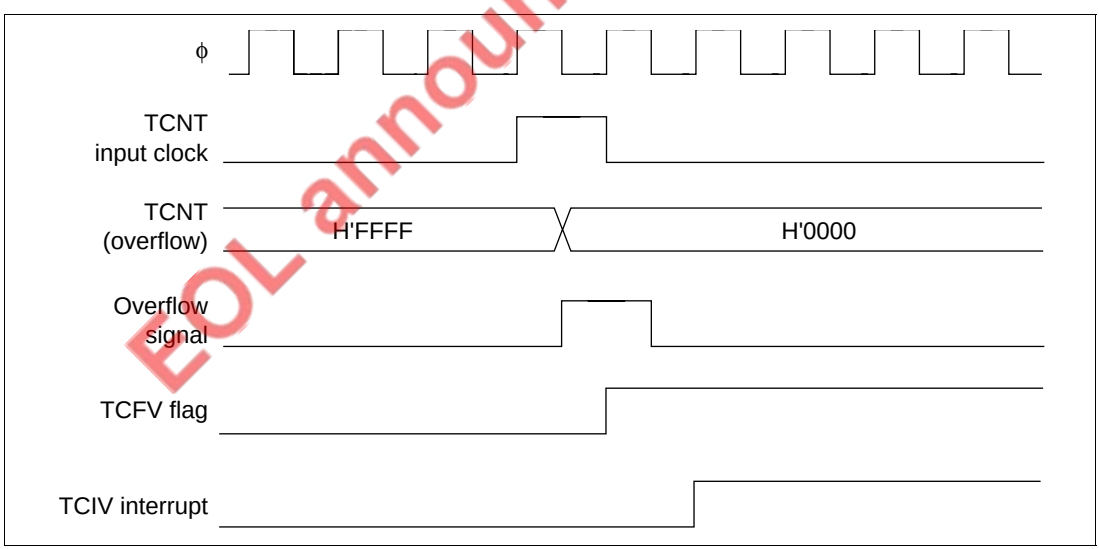


Figure 8.37 TCIV Interrupt Setting Timing

Status Flag Clearing Timing: The status flag is cleared when the CPU reads a 1 status followed by a 0 write. Figure 8.38 shows the timing for status flag clearing by the CPU.

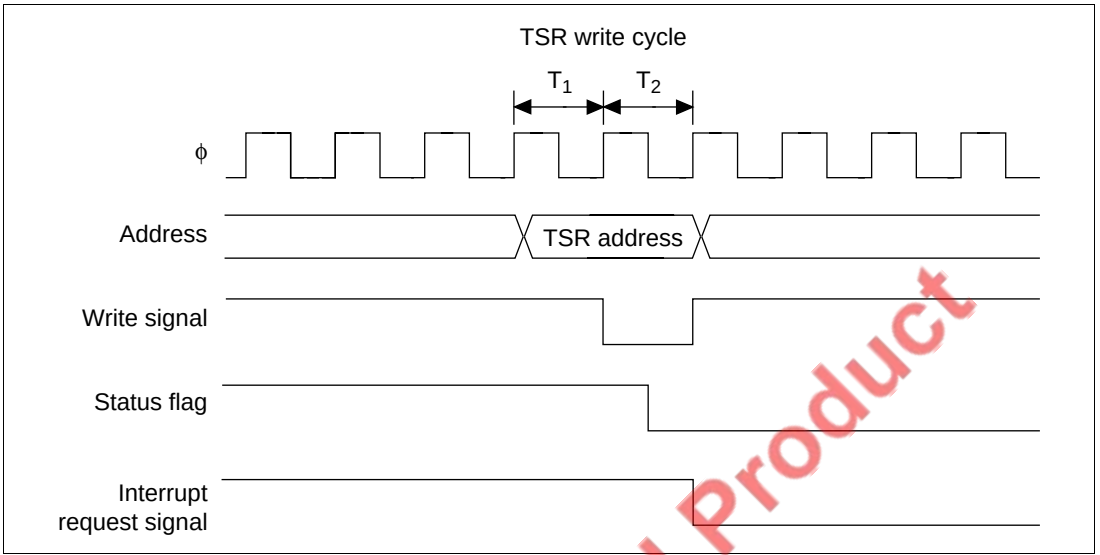


Figure 8.38 Timing of Status Flag Clearing by the CPU

8.7 Notes and Precautions

This section describes contention and other matters requiring special attention during MTU operations.

Note on Cycle Setting: When setting a counter clearing by compare-match, clearing is done in the final state when TCNT matches the TGR value (update timing for count value on TCNT match). The actual number of states set in the counter is given by the following equation:

$$f = \frac{\phi}{(N + 1)}$$

(f: counter frequency, ϕ: operating frequency, N: value set in the TGR)

Contention between TCNT Write and Clear: If a counter clear signal is issued in the T_2 state during the TCNT write cycle, TCNT clearing has priority, and TCNT write is not conducted (figure 8.39).

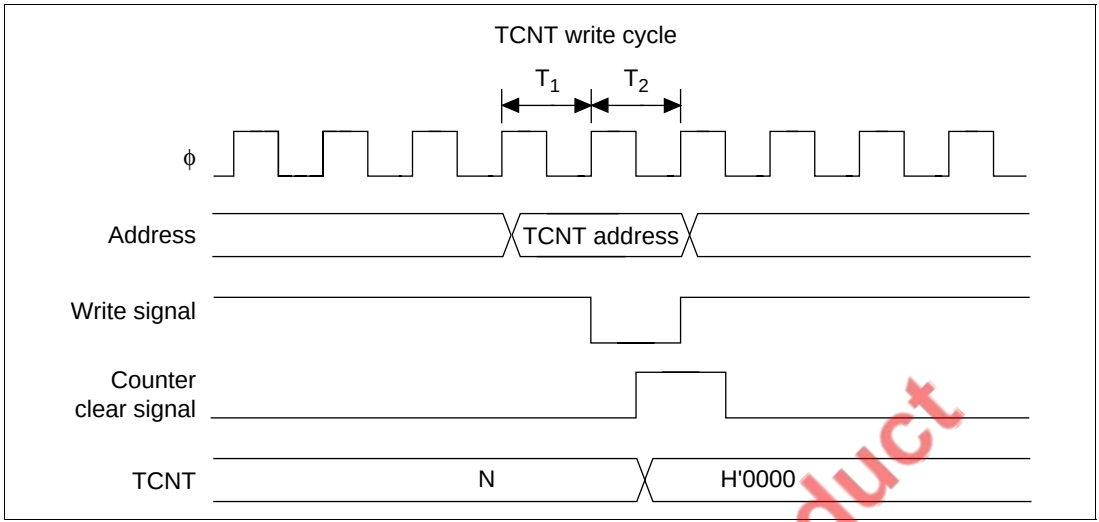


Figure 8.39 TCNT Write and Clear Contention

Contention between TCNT Write and Increment: If a count-up signal is issued in the T_2 state during the TCNT write cycle, TCNT write has priority, and the counter is not incremented (figure 8.40).

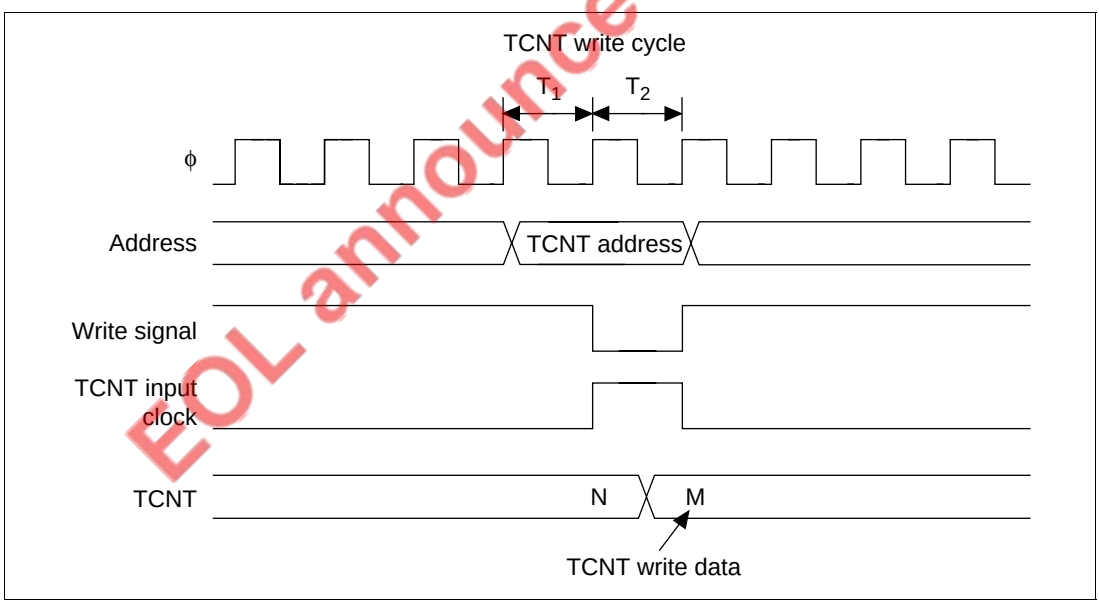


Figure 8.40 TCNT Write and Increment Contention

Contention between Buffer Register Write and Compare Match: If a compare-match occurs in the T_2 state of the TGR write cycle, data is transferred by the buffer operation from the buffer register to the TGR. On channel 0, the data to be transferred is that after the write (figure 8.41).

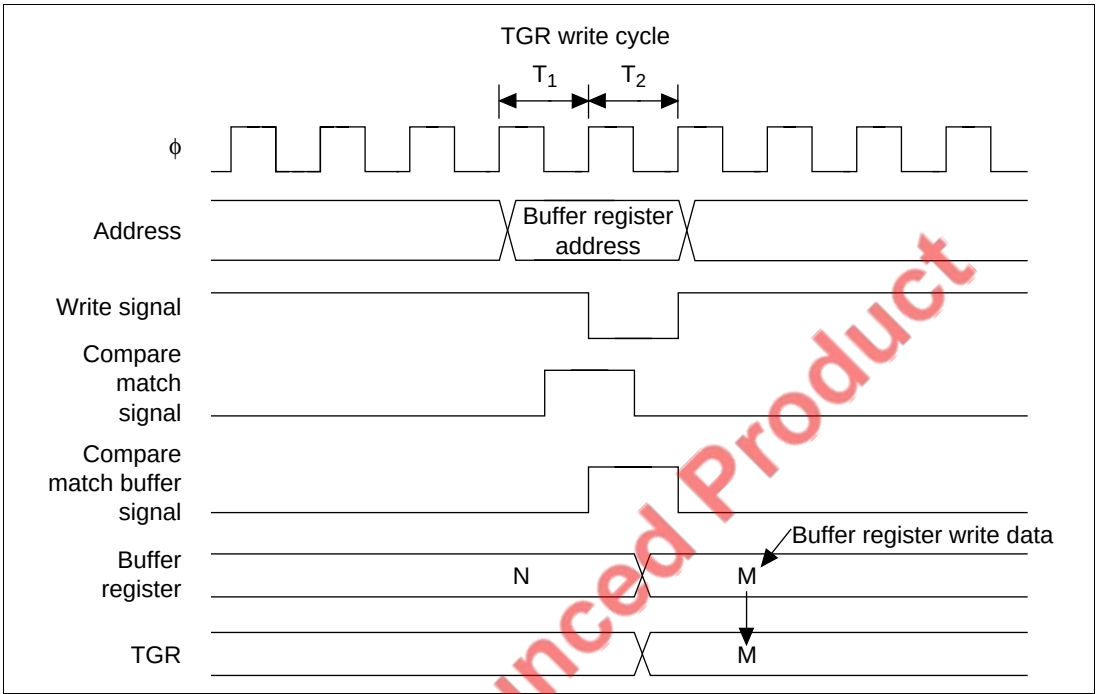


Figure 8.41 TGR Write and Compare-Match Contention (Channel 0)

Contention between TGR Read and Input Capture: If an input capture signal is issued in the T_1 state of the TGR read cycle, the read data is that after input capture transfer (figure 8.42).

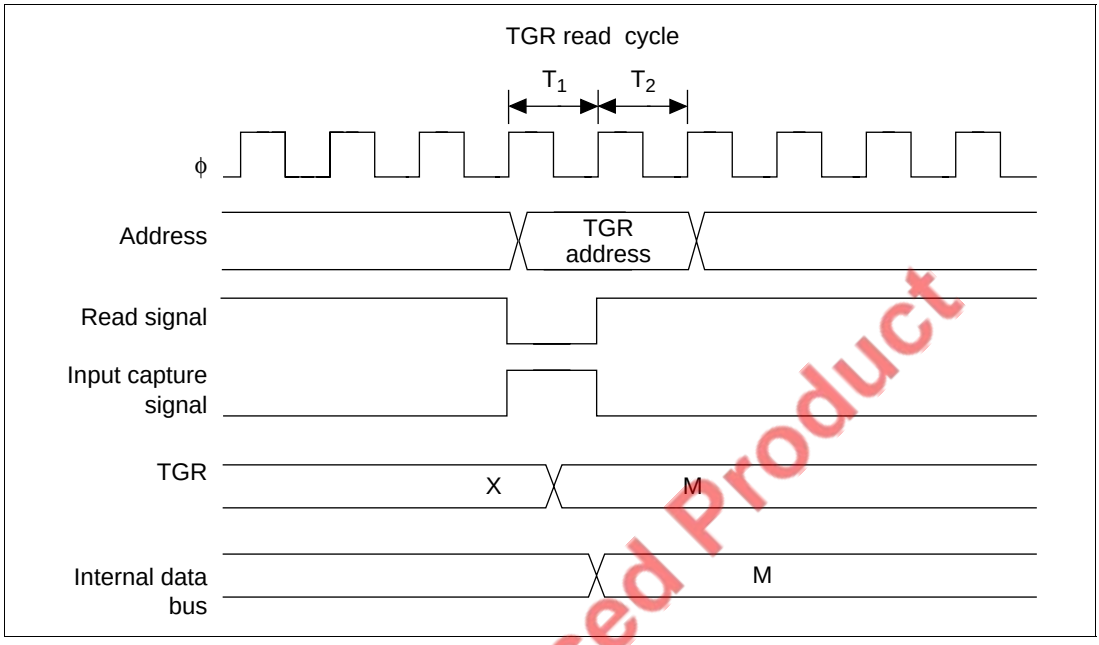


Figure 8.42 TGR Read and Input Capture Contention

Contention between TGR Write and Input Capture: If an input capture signal is issued in the T_2 state of the TGR read cycle, input capture has priority, and TGR write does not occur (figure 8.43).

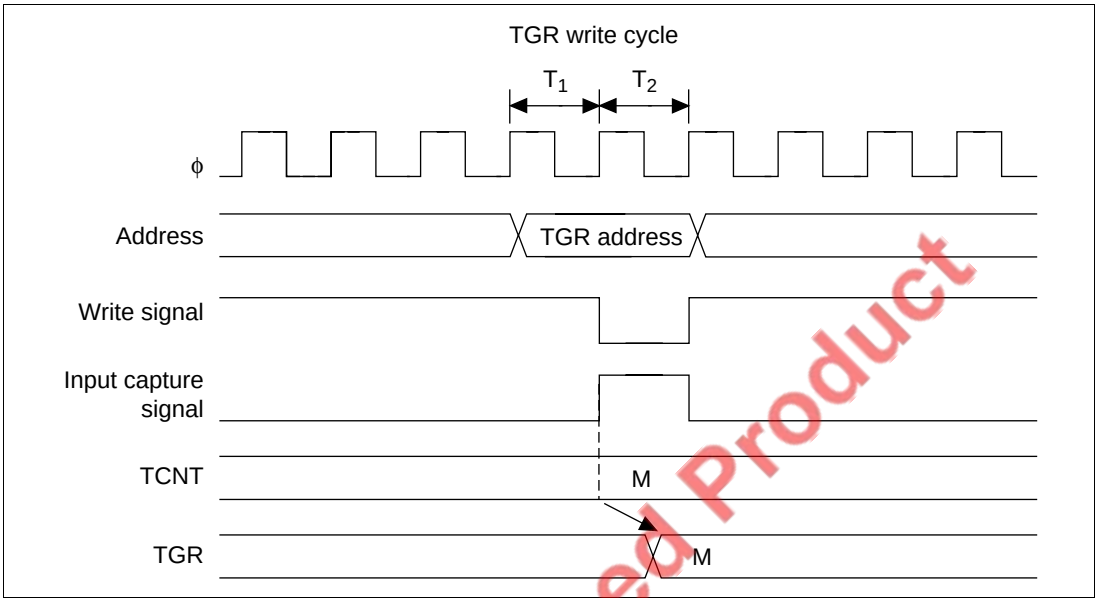


Figure 8.43 TGR Write and Input Capture Contention

Contention between Buffer Register Write and Input Capture: If an input capture signal is issued in the T_2 state of the buffer write cycle, write to the buffer register does not occur, and buffer operation takes priority (figure 8.44).

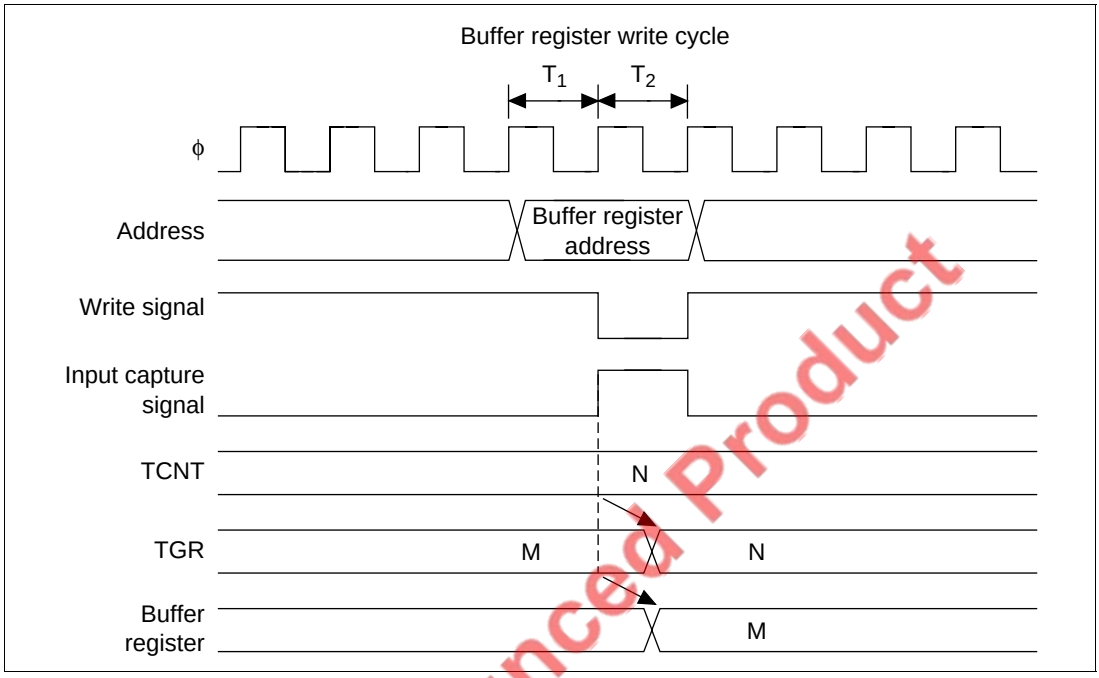


Figure 8.44 Buffer Register Write and Input Capture Contention

Contention Between TGR Write and Compare Match: If a compare-match occurs in the T_2 state of the TGR write cycle, data is written to the TGR and a compare-match signal is issued (figure 8.45).

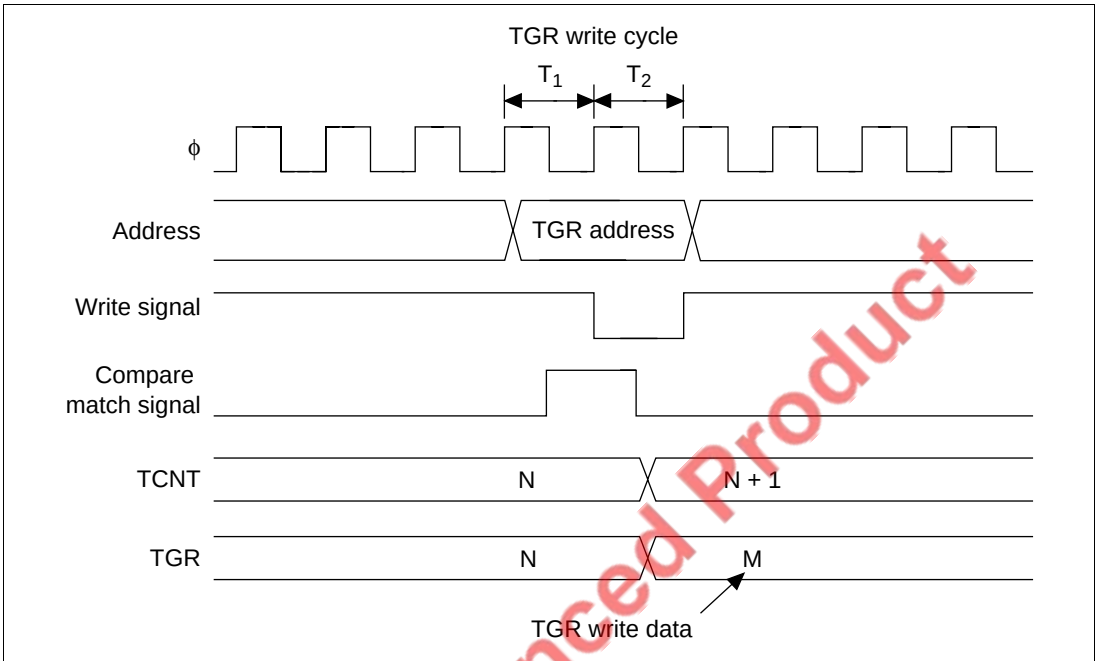


Figure 8.45 TGR Write and Compare Match Contention

TCNT2 Write and Overflow Contention in Cascade Connection: With timer counters TCNT1 and TCNT2 in a cascade connection, when a contention occurs during TCNT1 count (during a TCNT2 overflow) in the T_2 state of the TCNT2 write cycle, the write to TCNT2 is conducted, and the TCNT1 count signal is prohibited. At this point, if there is match with TGR1A and the TCNT1 value, a compare signal is issued. When the TCNT1 count clock is selected as the channel 0 input capture source, TGR0A and TGR0C operate as input capture registers. When TGR0C compare-match/input capture is selected as the TGR1B input capture source, TGR1B operates as an input capture register. The timing is shown in figure 8.46.

For cascade connections, be sure to synchronize settings for channels 1 and 2 when setting TCNT clearing.

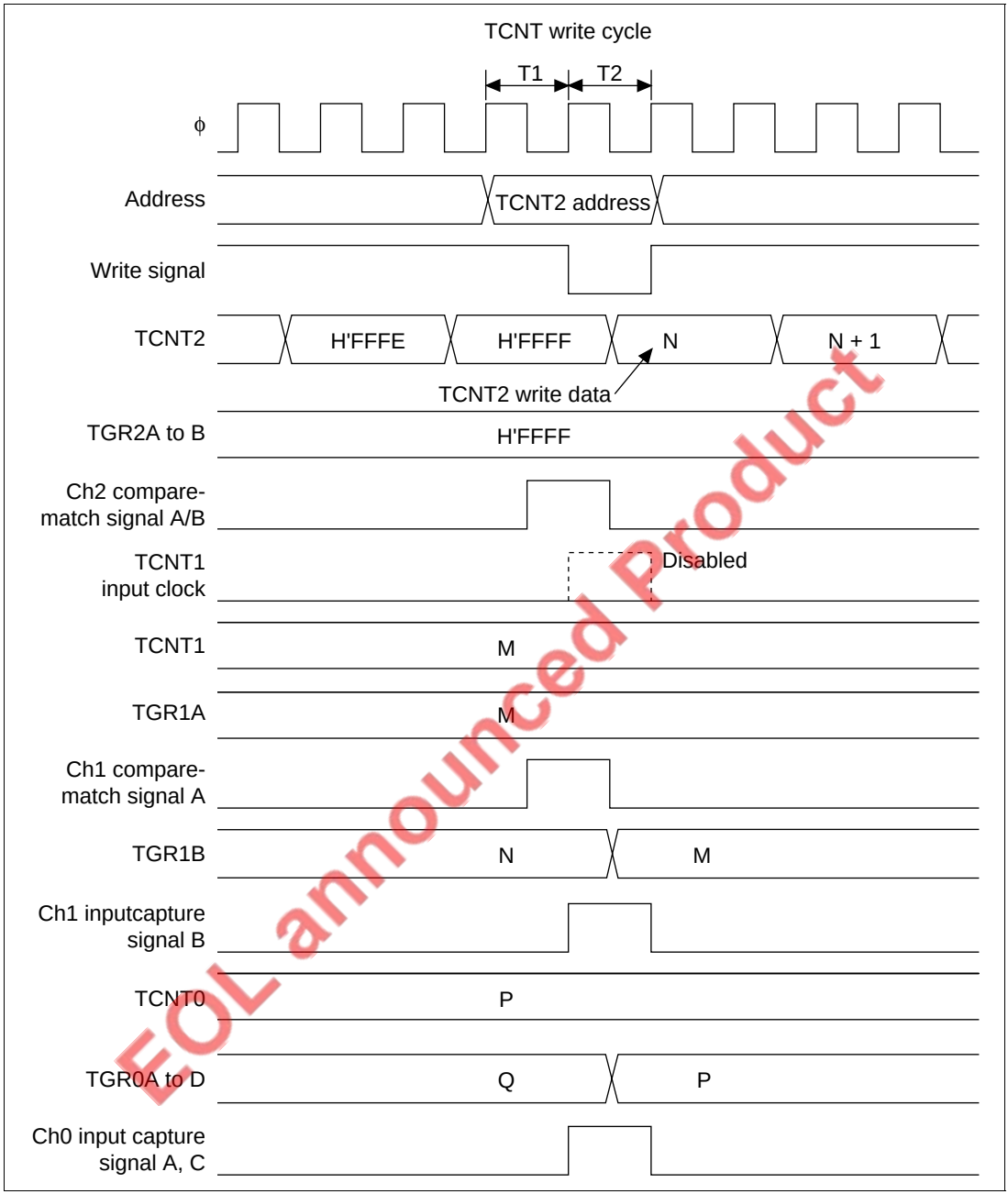


Figure 8.46 TCNT2 Write and Overflow Contention with Cascade Connection

Contention between Overflow and Counter Clearing: If overflow and counter clearing occur simultaneously, the TCFV flag in TSR is not set and TCNT clearing takes precedence.

Figure 8.47 shows the operation timing when a TGR compare-match is specified as the clearing source, and H'FFFF is set in TGR.

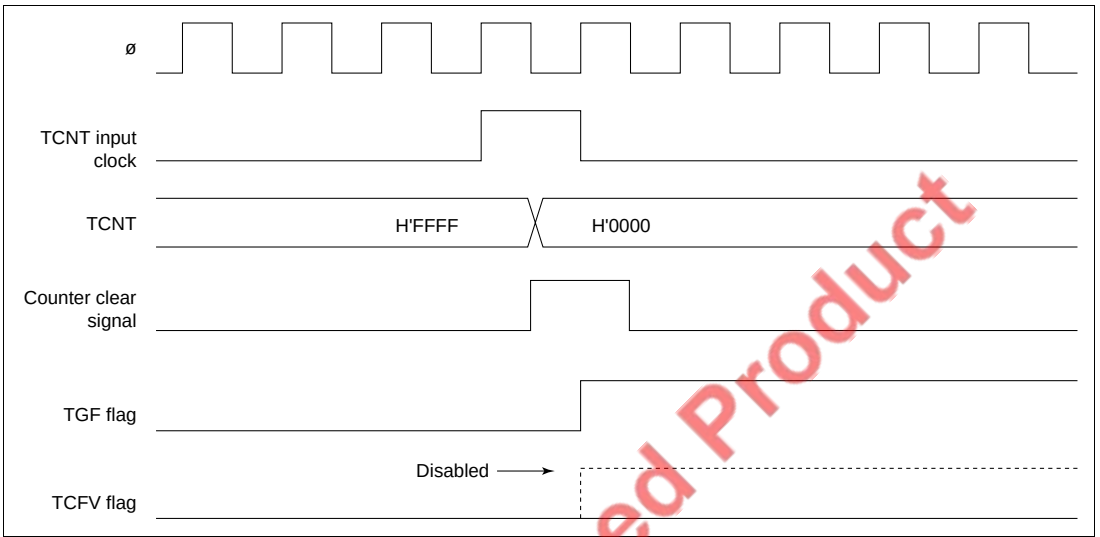


Figure 8.47 Contention between Overflow and Counter Clearing

Contention between TCNT Write and Overflow: If there is an up-count in the T2 state of a TCNT write cycle, and overflow occurs, the TCNT write takes precedence and the TCFV flag in TSR is not set .

Figure 8.48 shows the operation timing in this case.

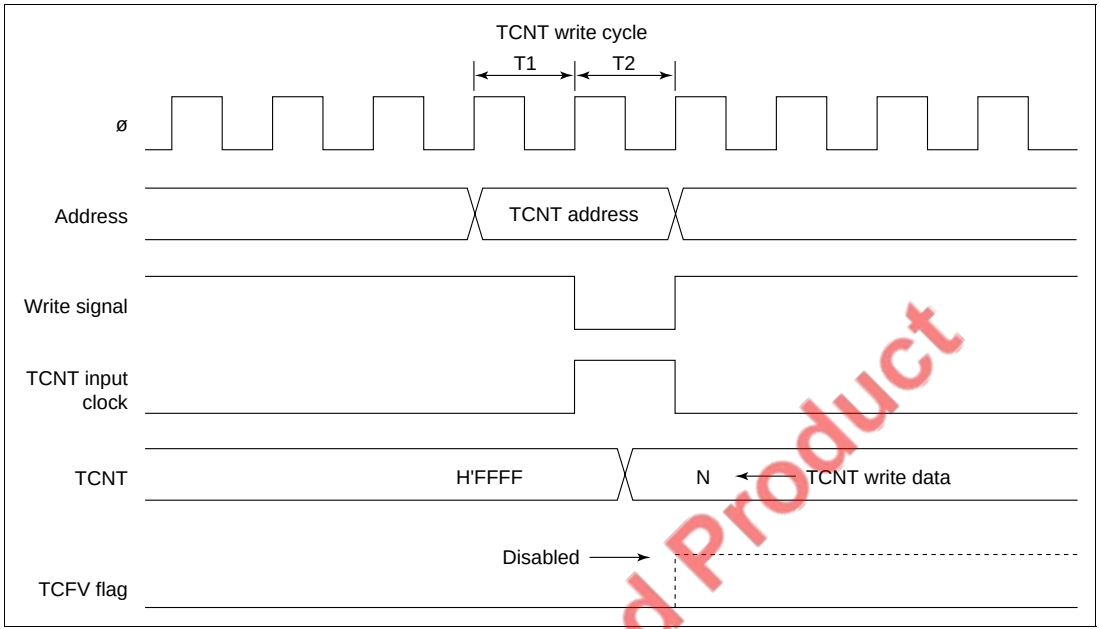


Figure 8.48 Contention between TCNT Write and Overflow

8.8 MTU Output Pin Initialization

8.8.1 Operating Modes

The MTU has the following three operating modes. Waveform output is possible in all of these modes.

- Normal mode (channels 0 to 2)
- PWM mode 1 (channels 0 to 2)
- PWM mode 2 (channels 0 to 2)

The MTU output pin initialization method for each of these modes is described in this section.

8.8.2 Reset Start Operation

The MTU output pins (TIOC*) are initialized low by a reset and in standby mode. Since MTU pin function selection is performed by the pin function controller (PFC), when the PFC is set, the MTU pin states at that point are output to the ports. When MTU output is selected by the PFC immediately after a reset, the MTU output initial level, low, is output directly at the port. When the active level is low, the system will operate at this point, and therefore the PFC setting should be made after initialization of the MTU output pins is completed.

Note: * Represents the channel number and port symbol.

8.8.3 Operation in Case of Re-Setting Due to Error During Operation, Etc.

If an error occurs during MTU operation, MTU output should be cut by the system. Cutoff is performed by switching the pin output to port output with the PFC and outputting the inverse of the active level. The pin initialization procedures for re-setting due to an error during operation, etc., and the procedures for restarting in a different mode after re-setting, are shown below.

The MTU has three operating modes, as stated above. There are thus 9 mode transition combinations. Possible mode transition combinations are shown in table 8.9.

Table 8.9 Mode Transition Combinations

	After		
Before	**Normal**	**PWM1**	**PWM2**
Normal	(1)	(2)	(3)
PWM1	(4)	(5)	(6)
PWM2	(7)	(8)	(9)

Legend:
Normal: Normal mode
PWM1: PWM1 mode
PWM2: PWM2 mode
The above abbreviations are used in some places in the following descriptions.

8.8.4 Overview of Initialization Procedures and Mode Transitions in Case of Error During Operation, Etc.

- When making a transition to a mode (Normal, PWM1, PWM2) in which the pin output level is selected by the timer I/O control register (TIOR) setting, initialize the pins by means of a TIOR setting.
- In PWM mode 1, since a waveform is not output to the TIOC*B pin, setting TIOR will not initialize the pins. If initialization is required, carry it out in normal mode, then switch to PWM mode 1.
- In PWM mode 2, since a waveform is not output to the cycle register pin, setting TIOR will not initialize the pins. If initialization is required, carry it out in normal mode, then switch to PWM mode 2.
- In normal mode or PWM mode 2, if TGRC and TGRD operate as buffer registers, setting TIOR will not initialize the buffer register pins. If initialization is required, clear buffer mode, carry out initialization, then set buffer mode again.
- In PWM mode 1, if either TGRC or TGRD operates as a buffer register, setting TIOR will not initialize the TGRC pin. To initialize the TGRC pin, clear buffer mode, carry out initialization, then set buffer mode again.

Note: An asterisk in this section represents the channel number.

Pin initialization procedures are described below for the numbered combinations in table 8.9. The active level is assumed to be low.

(1) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in Normal Mode: Figure 8.49 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in normal mode after re-setting.

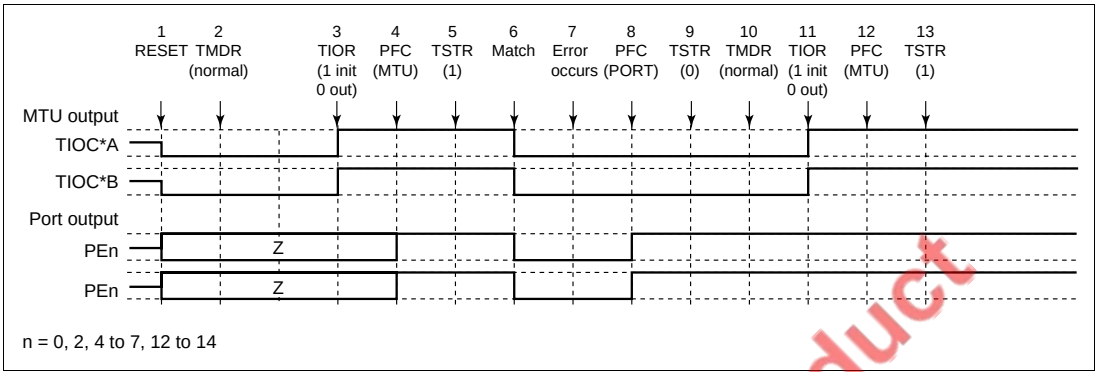


Figure 8.49 Error Occurrence in Normal Mode, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. After a reset, the TMDR setting is for normal mode.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Not necessary when restarting in normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(2) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in PWM Mode 1: Figure 8.50 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in PWM mode 1 after re-setting.

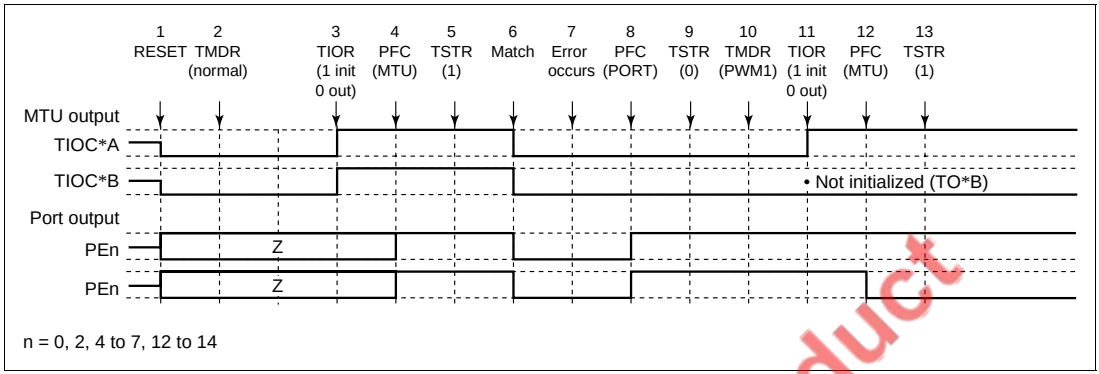


Figure 8.50 Error Occurrence in Normal Mode, Recovery in PWM Mode 1

1 to 9 are the same as in figure 8.49.

10. Set PWM mode 1.
11. Initialize the pins with TIOR. (In PWM mode 1, the TO*B side is not initialized. If initialization is required, initialize in normal mode, then switch to PWM mode 1.)
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(3) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in PWM Mode 2: Figure 8.51 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in PWM mode 2 after re-setting.

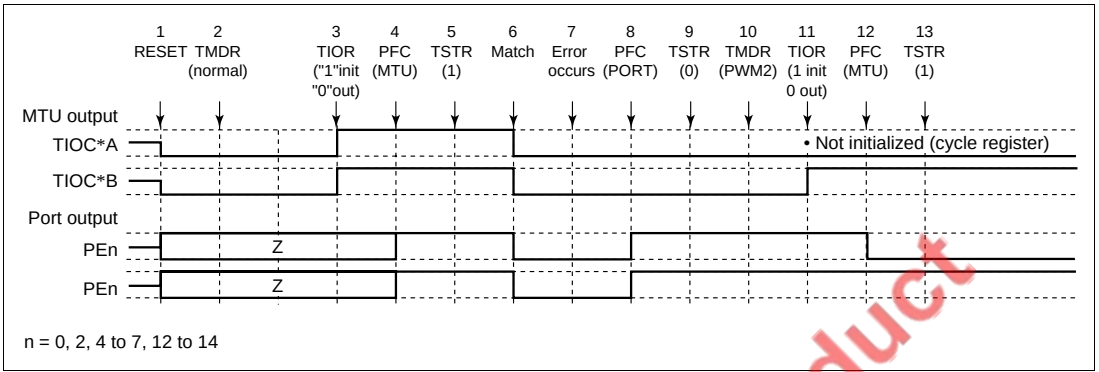


Figure 8.51 Error Occurrence in Normal Mode, Recovery in PWM Mode 2

1 to 9 are the same as in figure 8.49.

10. Set PWM mode 2.
11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized. If initialization is required, initialize in normal mode, then switch to PWM mode 2.)
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(4) Operation when Error Occurs during PWM Mode 1 Operation, and Operation is Restarted in Normal Mode: Figure 8.52 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in normal mode after re-setting.

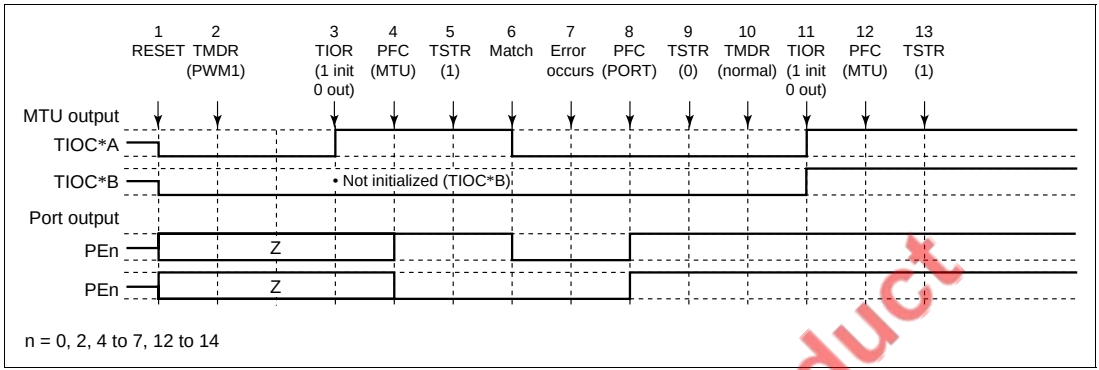


Figure 8.52 Error Occurrence in PWM Mode 1, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. Set PWM mode 1.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence. In PWM mode 1, the TIOC*B side is not initialized.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Set normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(5) Operation when Error Occurs during PWM Mode 1 Operation, and Operation is Restarted in PWM Mode 1: Figure 8.53 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in PWM mode 1 after re-setting.

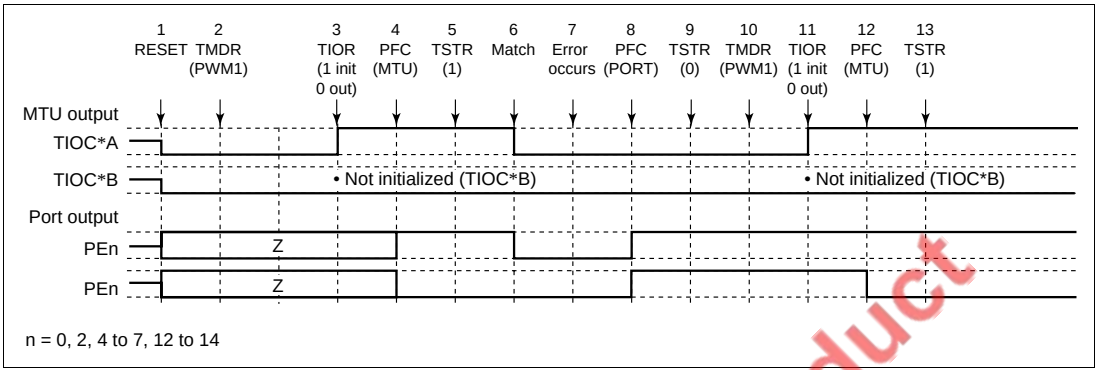


Figure 8.53 Error Occurrence in PWM Mode 1, Recovery in PWM Mode 1

1 to 9 are the same as in figure 8.52.

10. Not necessary when restarting in PWM mode 1.
11. Initialize the pins with TIOR. (In PWM mode 1, the TIOC*B side is not initialized.)
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(6) Operation when Rrror Occurs during PWM Mode 1 Operation, and Operation is Restarted in PWM Mode 2: Figure 8.54 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in PWM mode 2 after re-setting.

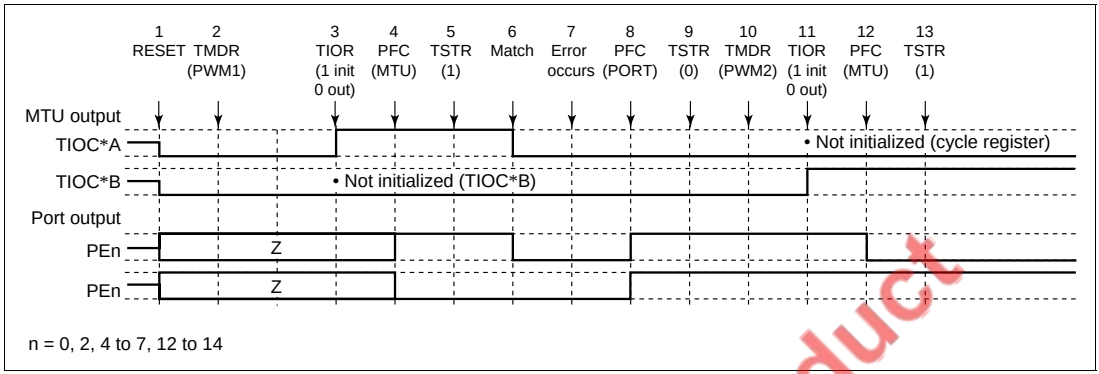


Figure 8.54 Error Occurrence in PWM Mode 1, Recovery in PWM Mode 2

1 to 9 are the same as in figure 8.52.

10. Set PWM mode 2.
11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized.)
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(7) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in Normal Mode: Figure 8.55 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in normal mode after re-setting.

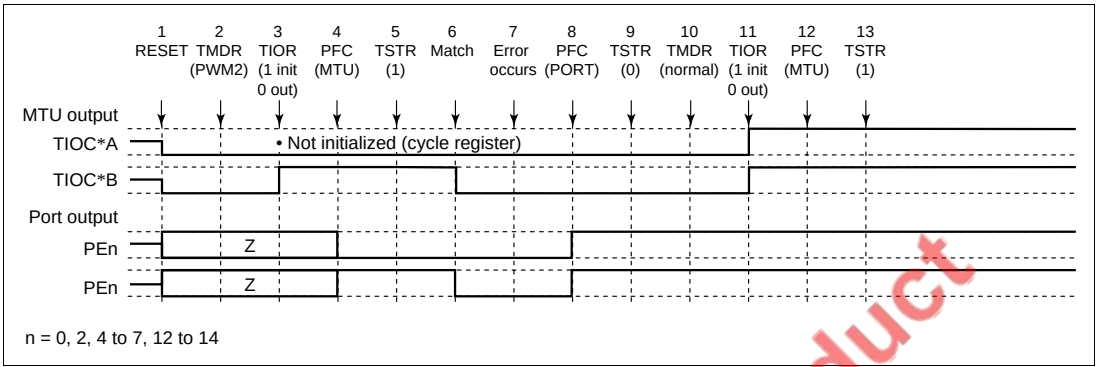


Figure 8.55 Error Occurrence in PWM Mode 2, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. Set PWM mode 2.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence. In PWM mode 2, the cycle register pins are not initialized. In the example, TIOC*A is the cycle register.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Set normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(8) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in PWM Mode 1: Figure 8.56 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in PWM mode 1 after re-setting.

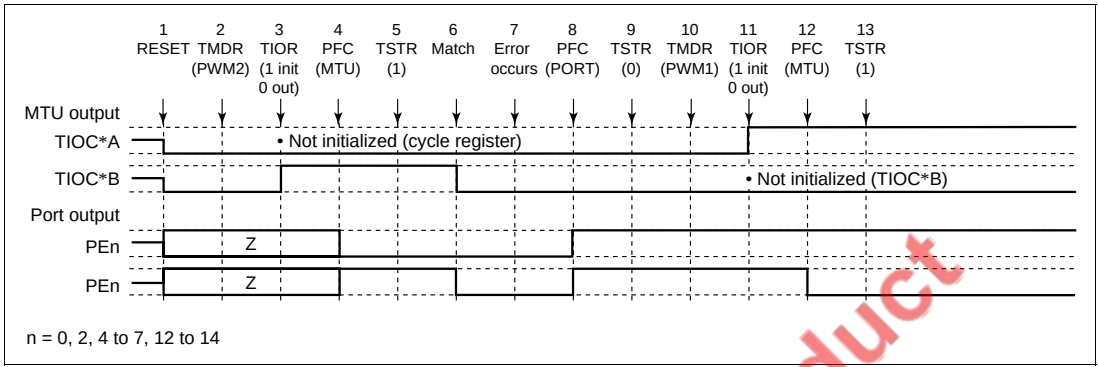


Figure 8.56 Error Occurrence in PWM Mode 2, Recovery in PWM Mode 1

1 to 9 are the same as in figure 8.55.

10. Set PWM mode 1.
11. Initialize the pins with TIOR. (In PWM mode 1, the TIOC*B side is not initialized.)
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(9) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in PWM Mode 2: Figure 8.57 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in PWM mode 2 after re-setting.

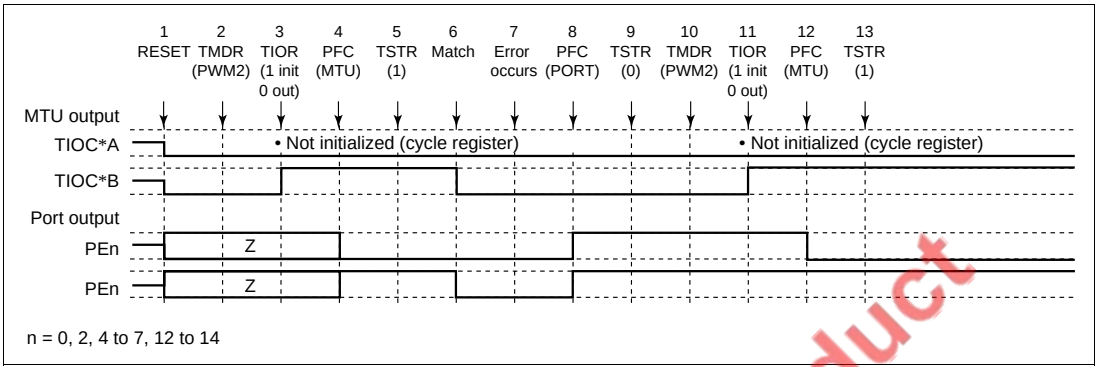


Figure 8.57 Error Occurrence in PWM Mode 2, Recovery in PWM Mode 2

1 to 9 are the same as in figure 8.55.

10. Not necessary when restarting in PWM mode 2.
11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized.)
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

Section 9 8-Bit Timer 2 (TIM2)

9.1 Overview

8-bit timer 2 (TIM2) is a single-channel interval timer that generates compare match interrupts.

9.1.1 Features

- 8-bit interval timer
- Generates compare match interrupts
  A compare match interrupt is generated by a counter compare match.
- Selection of seven counter input clock sources

9.1.2 Block Diagram

Figure 9.1 shows a block diagram of 8-bit timer 2 (TIM2).

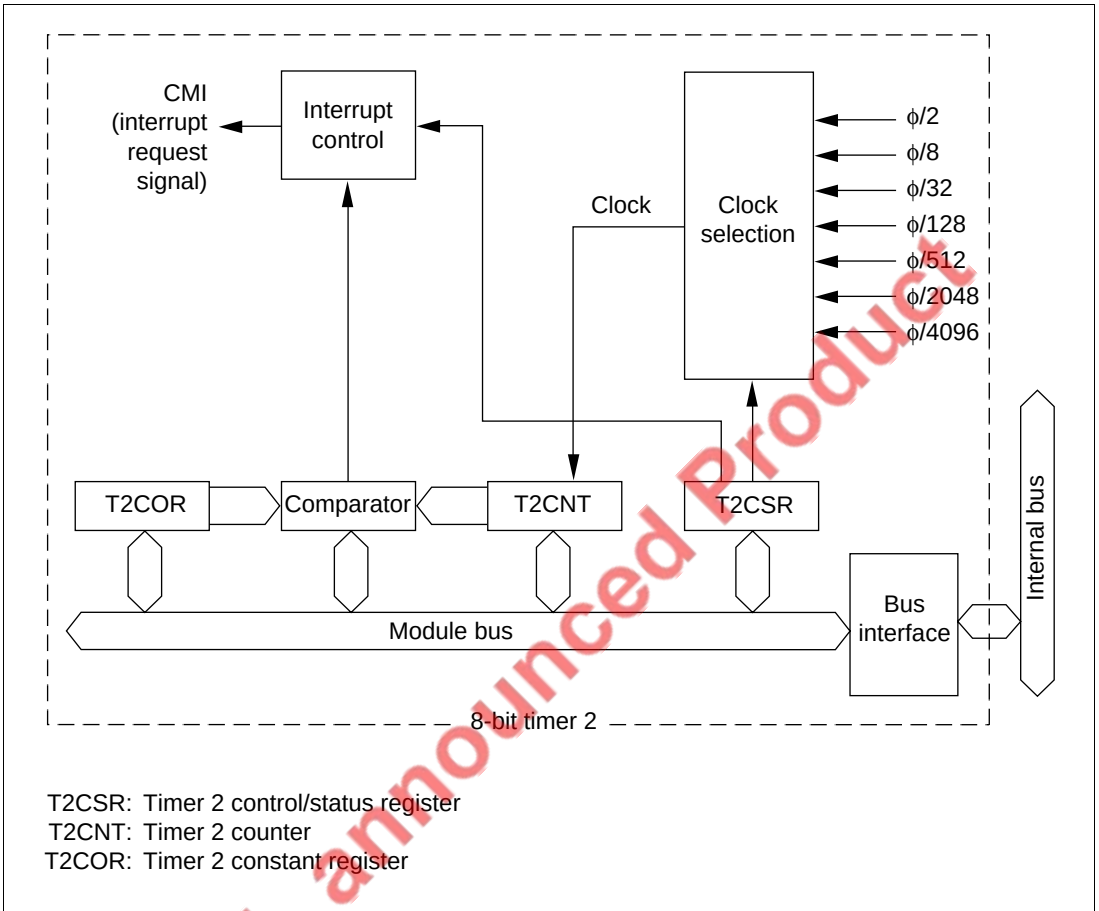


Figure 9.1 Block Diagram of 8-Bit Timer 2

9.1.3 Register Configuration

8-bit timer 2 (TIM2) has three registers for compare match cycle setting, clock selection, and other functions. The register configuration is shown in table 9.1.

All the registers are 16 bits in size, and are initialized by a power-on reset.

Table 9.1 8-Bit Timer 2 Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Timer 2 control/status register	T2CSR	R/W	H'0000	H'FFFF862C	8, 16, 32
Timer 2 counter	T2CNT	R/W	H'0000	H'FFFF862E	8, 16, 32
Timer 2 constant register	T2COR	R/W	H'0000	H'FFFF8630	8, 16

9.2 Register Descriptions

9.2.1 Timer 2 Control/Status Register (T2CSR)

The timer 2 control/status register (T2CSR) is a 16-bit readable/writable* register that selects the clock to be input to the timer 2 counter (T2CNT) and controls compare match interrupts (CMI).

T2CSR is initialized to H'0000 by a power-on reset.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	CMF	CMIE	CKS2	CKS1	CKS0	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R	R

- Bits 15 to 7—Reserved: These bits are always read as 0. The write value should always be 0.
- Bit 6—Compare Match Flag (CMF): Status flag that indicates a match between the values of T2CNT and T2COR. The setting and clearing conditions for this flag are shown below.

Bit 6: CMF	Description
0	[Clearing condition] Cleared by reading T2CSR when CMF = 1, then writing 0 in CMF (Initial value)
1	[Setting condition] Set when T2CNT = T2COR*

Note: * When T2CNT and T2COR still contain their initial values (when the initial values have not been changed or when the T2CNT value has not been incremented), CMF is not set even though the T2CNT and T2COR values are the same (H'0000).

- Bit 5—Compare Match Interrupt Enable (CMIE): Enables or disables interrupt requests initiated by the CMF flag when set to 1 in T2CSR.

Bit 5: CMIE	Description	
0	Interrupt request by CMF flag disabled	(Initial value)
1	Interrupt request by CMF flag enabled	

- Bits 4 to 2—Clock Select 2 to 0 (CKS2 to CKS0): These bits select one of seven internal clock sources, obtained by dividing the system clock (ϕ), for input to T2CNT.

Bit 4: CKS2	Bit 3: CKS1	Bit 2: CKS0	Description	
0	0	0	Up-count stopped	(Initial value)
		1	ϕ/2	
	1	0	ϕ/8	
		1	ϕ/32	
1	0	0	ϕ/128	
		1	ϕ/512	
	1	0	ϕ/2048	
		1	ϕ/4096	

- Bits 1 and 0—Reserved: These bits are always read as 0. The write value should always be 0.

9.2.2 Timer 2 Counter (T2CNT)

The timer 2 counter (T2CNT) is a 16-bit readable/writable register used as an 8-bit up-counter.

T2CNT increments on the internal clock selected by bits CKS2 to CKS0 in T2CSR. The T2CNT value can be read or written by the CPU at all times. When the T2CNT value matches the value in the timer 2 constant register (T2COR), T2CNT is cleared to H'0000 and the CMF flag is set to 1 in T2CSR. If the CMIE bit in T2CSR is set to 1 at this time, a compare match interrupt (CMI) is generated.

Bits 15 to 8 are reserved and have no counter function. These bits are always read as 0. The write value should always be 0.

T2CNT is initialized to H'0000 by a power-on reset.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

9.2.3 Timer 2 Constant Register (T2COR)

The timer 2 constant register (T2COR) is a 16-bit readable/writable register that is used to set the T2CNT compare match cycle. The values in T2COR and T2CNT are continually compared, and when the values match the CMF flag is set in T2CSR and T2CNT is cleared to 0. If the CMIE bit in T2CSR is set to 1, an interrupt request is sent to the interrupt controller in response to the match signal. The interrupt request is output continuously until the CMF flag in T2CSR is cleared.

Bits 15 to 8 are reserved and are not used in the cycle setting. These bits are always read as 0. The write value should always be 0.

T2COR is initialized to H'0000 by a power-on reset.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

9.3 Operation

9.3.1 Cyclic Count Operation

When a clock is selected with bits CKS2 to CKS0 in the T2CSR register, the T2CNT counter starts incrementing on the selected clock. When the T2CNT counter value matches the value in the timer 2 constant register (T2COR), the T2CNT counter is cleared to H'00, and the CMF flag is set to 1 in the T2CSR register. If the CMIE bit in T2CSR is set to 1 at this time, a compare match interrupt (CMI) is requested. The T2CNT counter then starts incrementing again from H'00.

The compare match counter operation is shown in figure 9.2.

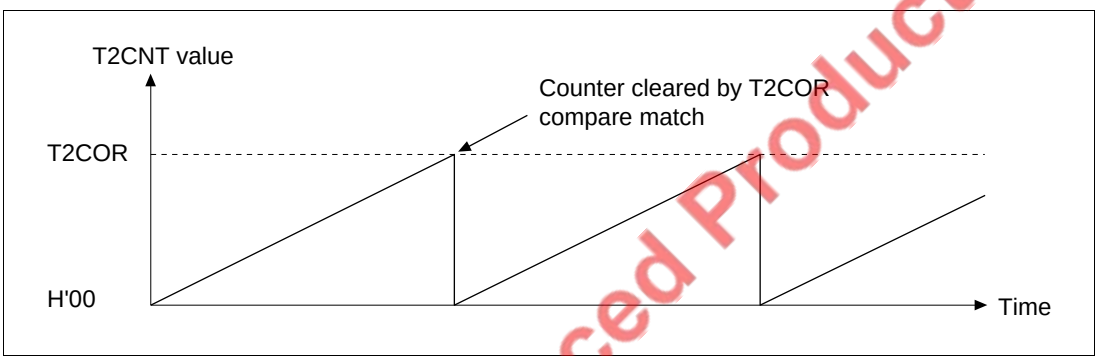


Figure 9.2 Counter Operation

9.3.2 T2CNT Count Timing

Any of seven internal clocks (ϕ/2, ϕ/8, ϕ/32, ϕ/128, ϕ/512, ϕ/2048, or ϕ/4096) divided from the system clock (CK) can be selected with bits CKS2 to CKS0 in T2CSR. The count timing is shown in figure 9.3.

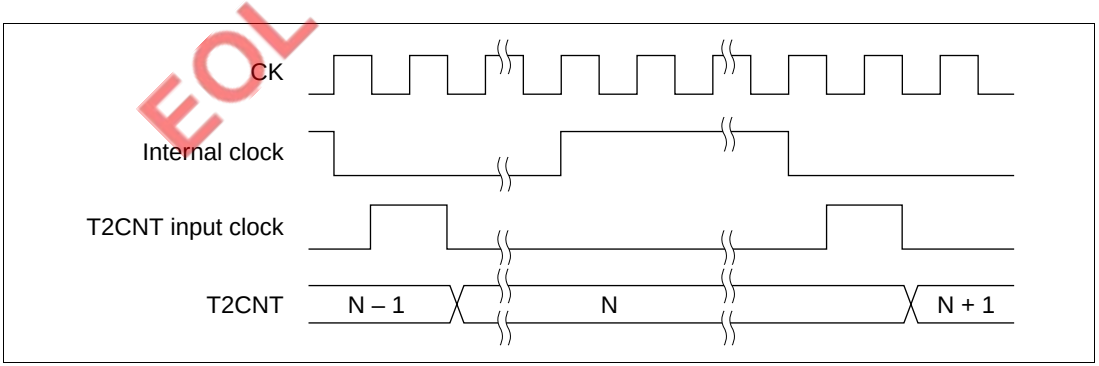


Figure 9.3 Count Timing

9.4 Interrupts

9.4.1 Interrupt Source

When interrupt request flag CMF is set to 1, and interrupt enable bit CMIE is also 1, the corresponding interrupt request is output.

9.4.2 Timing of Compare Match Flag Setting

The CMF bit in the T2CSR register is set to 1 by the compare match signal generated when the T2COR register and T2CNT counter values match. The compare match signal is generated in the last state in which the match is true (when the value at which the T2CNT counter match occurred is about to be updated). Therefore, after a match between the T2CNT counter and the T2COR register, the compare match signal is not generated until the next T2CNT counter input clock pulse. Figure 9.4 shows the timing of CMF bit setting.

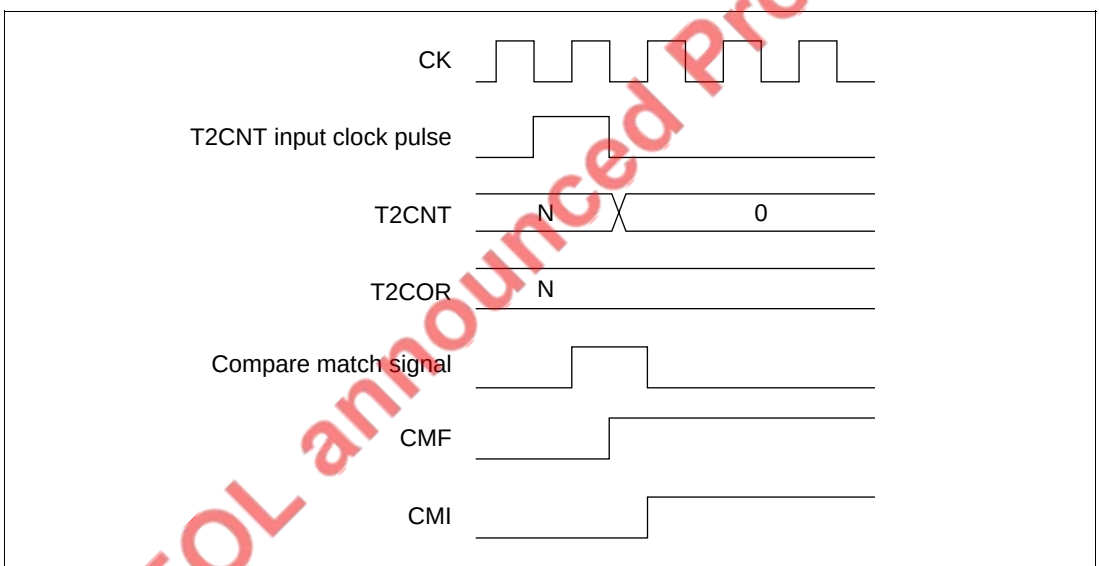


Figure 9.4 Timing of CMF Setting

9.4.3 Timing of Compare Match Flag Clearing

The CMF bit in the T2CSR register is cleared by reading the bit when it is set to 1, then writing 0 in it. Figure 9.5 shows the timing of CMF bit clearing by the CPU.

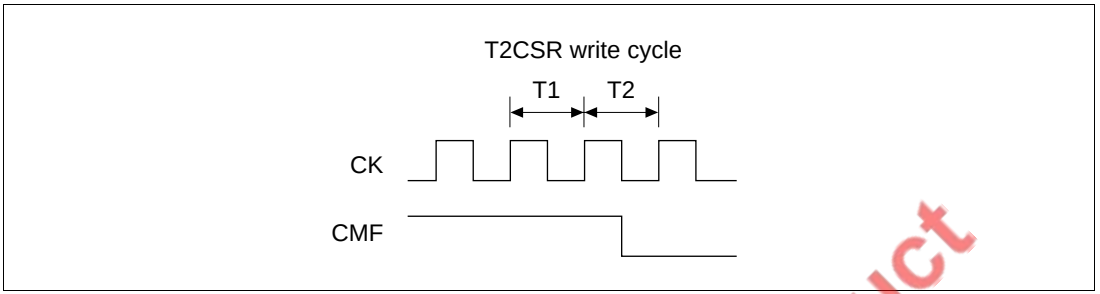


Figure 9.5 Timing of CMF Clearing by CPU

Section 10 Compare Match Timer (CMT)

10.1 Overview

The SH7018 has an on-chip compare match timer (CMT) configured of 16-bit timers for two channels. The CMT has 16-bit counters and can generate interrupts at set intervals.

10.1.1 Features

The CMT has the following features:

- Four types of counter input clock can be selected
  - One of four internal clocks (ϕ/8, ϕ/32, ϕ/128, ϕ/512) can be selected independently for each channel.
- Interrupt sources
  - A compare match interrupt can be requested independently for each channel.

10.1.2 Block Diagram

Figure 10.1 shows a block diagram of the CMT.

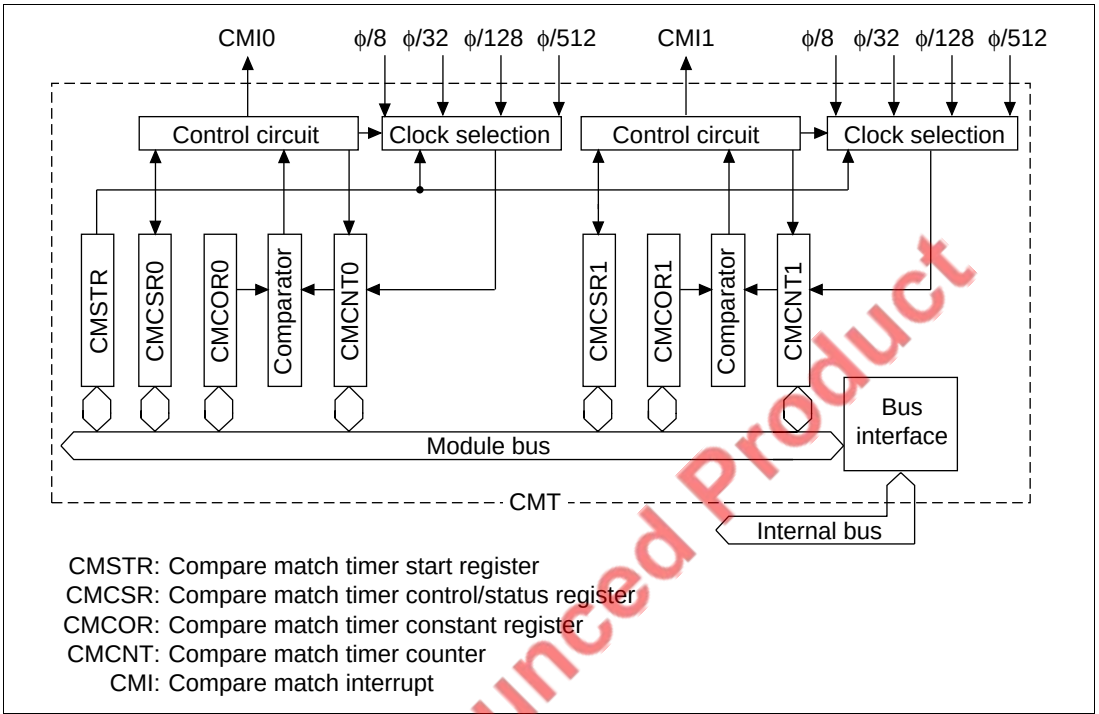


Figure 10.1 CMT Block Diagram

10.1.3 Register Configuration

Table 10.1 summarizes the CMT register configuration.

Table 10.1 Register Configuration

Channel	Name	Abbreviation	R/W	Initial Value	Address	Access Size (Bits)
Shared	Compare match timer start register	CMSTR	R/W	H'0000	H'FFFF83D0	8, 16, 32
0	Compare match timer control/status register 0	CMCSR0	R/(W)*	H'0000	H'FFFF83D2	8, 16, 32
	Compare match timer counter 0	CMCNT0	R/W	H'0000	H'FFFF83D4	8, 16, 32
	Compare match timer constant register 0	CMCOR0	R/W	H'FFFF	H'FFFF83D6	8, 16, 32
1	Compare match timer control/status register 1	CMCSR1	R/(W)*	H'0000	H'FFFF83D8	8, 16, 32
	Compare match timer counter 1	CMCNT1	R/W	H'0000	H'FFFF83DA	8, 16, 32
	Compare match timer constant register 1	CMCOR1	R/W	H'FFFF	H'FFFF83DC	8, 16

Note: * The only value that can be written to the CMCSR0 and CMCSR1 CMF bits is a 0 to clear the flags.

10.2 Register Descriptions

10.2.1 Compare Match Timer Start Register (CMSTR)

The compare match timer start register (CMSTR) is a 16-bit register that selects whether to operate or halt the channel 0 and channel 1 counters (CMCNT). It is initialized to H'0000 by power-on resets.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	STR1	STR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

- Bits 15 to 2—Reserved: These bits are always read as 0. The write value should always be 0.
- Bit 1—Count Start 1 (STR1): Selects whether to operate or halt compare match timer counter 1 (CMCNT1).

Bit 1: STR1	Description	
0	CMCNT1 count operation halted	(Initial value)
1	CMCNT1 count operation	

- Bit 0—Count Start 0 (STR0): Selects whether to operate or halt compare match timer counter 0 (CMCNT0).

Bit 0: STR0	Description	
0	CMCNT0 count operation halted	(Initial value)
1	CMCNT0 count operation	

10.2.2 Compare Match Timer Control/Status Register (CMCSR)

The compare match timer control/status register (CMCSR) is a 16-bit register that indicates the occurrence of compare matches, sets the enable/disable of interrupts, and establishes the clock used for incrementation. It is initialized to H'0000 by power-on resets.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	CMF	CMIE	—	—	—	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/W	R	R	R	R	R/W	R/W

Note: * The only value that can be written is a 0 to clear the flag.

- Bits 15 to 8 and 5 to 2—Reserved: These bits are always read as 0. The write value should always be 0.
- Bit 7—Compare Match Flag (CMF): This flag indicates whether or not the CMCNT and CMCOR values have matched.

Bit 7: CMF	Description	
0	CMCNT and CMCOR values have not matched	(Initial value)
	Clear condition: Write a 0 to CMF after reading a 1 from it	
1	CMCNT and CMCOR values have matched	

- Bit 6—Compare Match Interrupt Enable (CMIE): Selects whether to enable or disable a compare match interrupt (CMI) when the CMCNT and CMCOR values have matched (CMF = 1).

Bit 6: CMIE	Description	
0	Compare match interrupts (CMI) disabled	(Initial value)
1	Compare match interrupts (CMI) enabled	

- Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock input to the CMCNT from among the four internal clocks obtained by dividing the system clock (ϕ). When the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with the clock selected by CKS1 and CKS0.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	ϕ/8	(Initial value)
	1	ϕ/32	
1	0	ϕ/128	
	1	ϕ/512	

10.2.3 Compare Match Timer Counter (CMCNT)

The compare match timer counter (CMCNT) is a 16-bit register used as an upcounter for generating interrupt requests.

When an internal clock is selected with the CKS1, CKS0 bits of the CMCSR register and the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with that clock. When the CMCNT value matches that of the compare match timer constant register (CMCOR), the CMCNT is cleared to H'0000 and the CMF flag of the CMCSR is set to 1. If the CMIE bit of the CMCSR is set to 1 at this time, a compare match interrupt (CMI) is requested.

The CMCNT is initialized to H'0000 by power-on resets.

Bit:	15	14	13	12	11	10	9	8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

10.2.4 Compare Match Timer Constant Register (CMCOR)

The compare match timer constant register (CMCOR) is a 16-bit register that sets the compare match period with the CMCNT.

The CMCOR is initialized to H'FFFF by power-on resets.

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

10.3 Operation

10.3.1 Period Count Operation

When an internal clock is selected with the CKS1, CKS0 bits of the CMCSR register and the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with the selected clock. When the CMCNT counter value matches that of the compare match constant register (CMCOR), the CMCNT counter is cleared to H'0000 and the CMF flag of the CMCSR register is set to 1. If the CMIE bit of the CMCSR register is set to 1 at this time, a compare match interrupt (CMI) is requested. The CMCNT counter begins counting up again from H'0000.

Figure 10.2 shows the compare match counter operation.

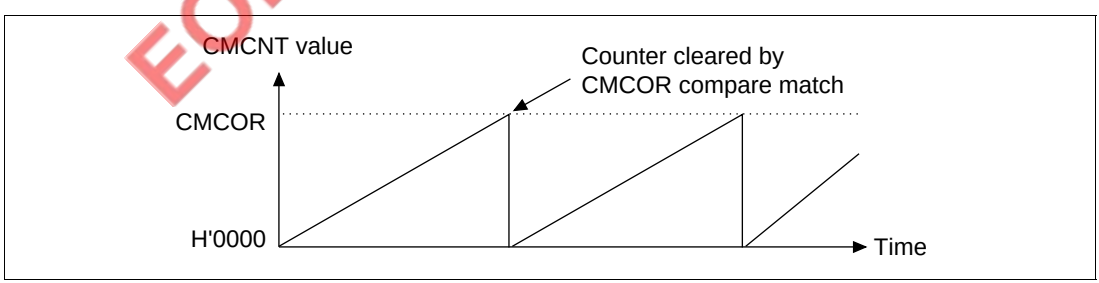


Figure 10.2 Counter Operation

10.3.2 CMCNT Count Timing

One of four clocks (ϕ/8, ϕ/32, ϕ/128, ϕ/512) obtained by dividing the system clock (CK) can be selected by the CKS1 and CKS0 bits of the CMCSR. Figure 10.3 shows the timing.

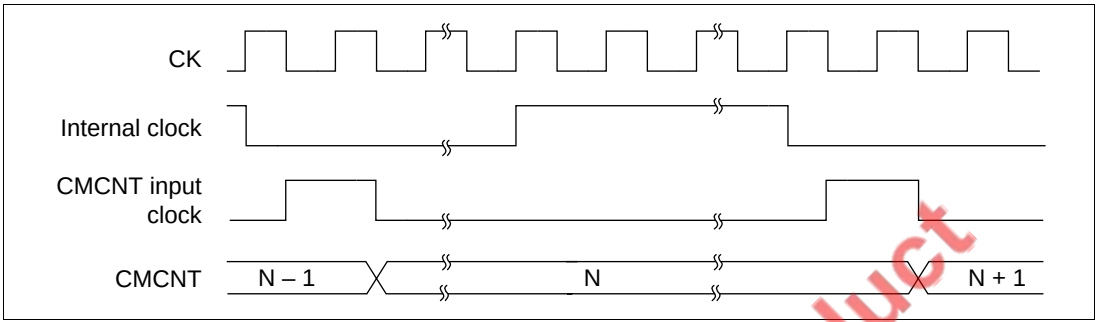


Figure 10.3 Count Timing

10.4 Interrupts

10.4.1 Interrupt Sources

The CMT has a compare match interrupt for each channel, with independent vector addresses allocated to each of them. The corresponding interrupt request is output when the interrupt request flag CMF is set to 1 and the interrupt enable bit CMIE has also been set to 1.

When activating CPU interrupts by interrupt request, the priority between the channels can be changed by using the interrupt controller settings. See section 6, Interrupt Controller, for details.

10.4.2 Compare Match Flag Set Timing

The CMF bit of the CMCSR register is set to 1 by the compare match signal generated when the CMCOR register and the CMCNT counter match. The compare match signal is generated upon the final state of the match (timing at which the CMCNT counter matching count value is updated). Consequently, after the CMCOR register and the CMCNT counter match, a compare match signal will not be generated until a CMCNT counter input clock occurs. Figure 10.4 shows the CMF bit set timing.

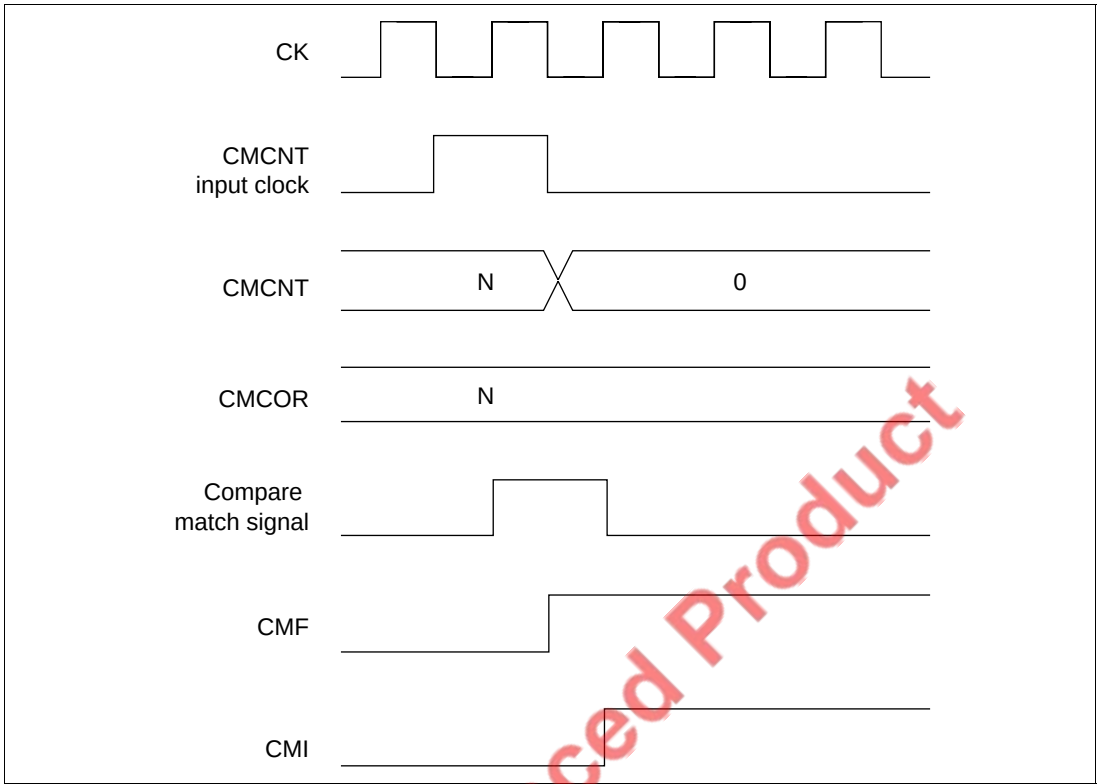


Figure 10.4 CMF Set Timing

10.4.3 Compare Match Flag Clear Timing

The CMF bit of the CMCSR register is cleared either by writing a 0 to it after reading a 1. Figure 10.5 shows the timing when the CMF bit is cleared by the CPU.

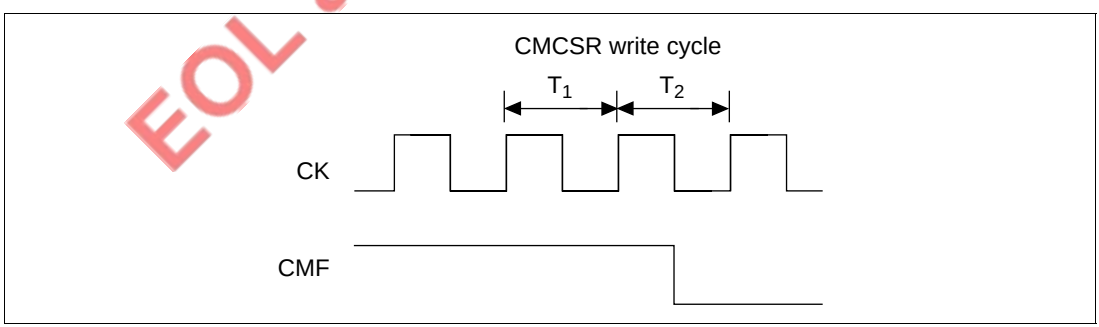


Figure 10.5 Timing of CMF Clear by the CPU

10.5 Notes on Use

Take care that the contentions described in sections 10.5.1 to 10.5.3 do not arise during CMT operation.

10.5.1 Contention between CMCNT Write and Compare Match

If a compare match signal is generated during the T_2 state of the CMCNT counter write cycle, the CMCNT counter clear has priority, so the write to the CMCNT counter is not performed. Figure 10.6 shows the timing.

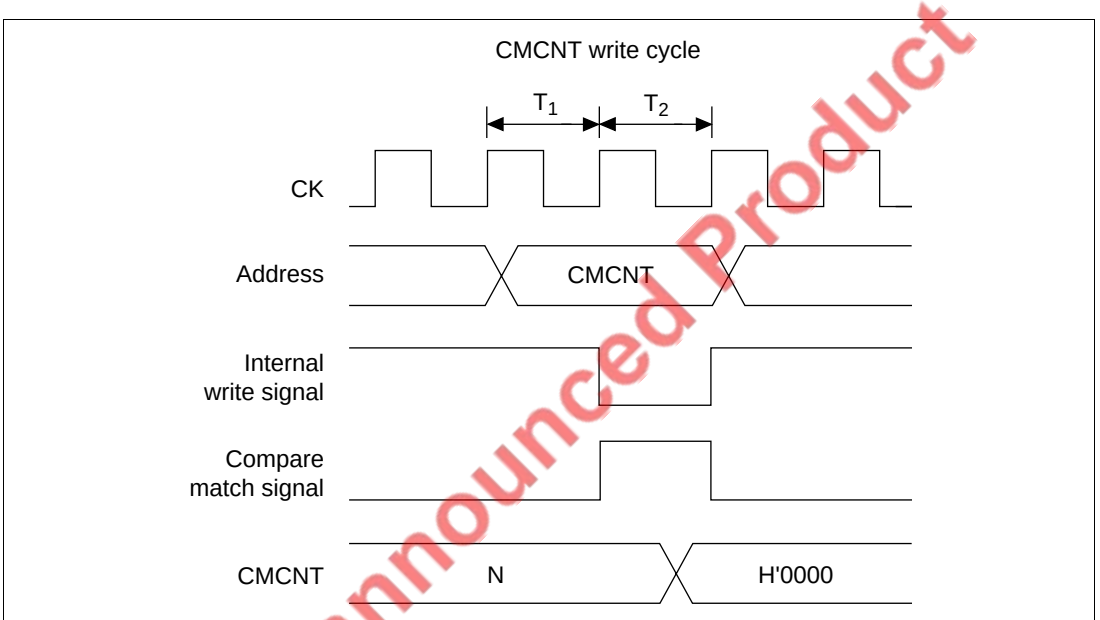


Figure 10.6 CMCNT Write and Compare Match Contention

10.5.2 Contention between CMCNT Word Write and Incrementation

If an increment occurs during the T_2 state of the CMCNT counter word write cycle, the counter write has priority, so no increment occurs. Figure 10.7 shows the timing.

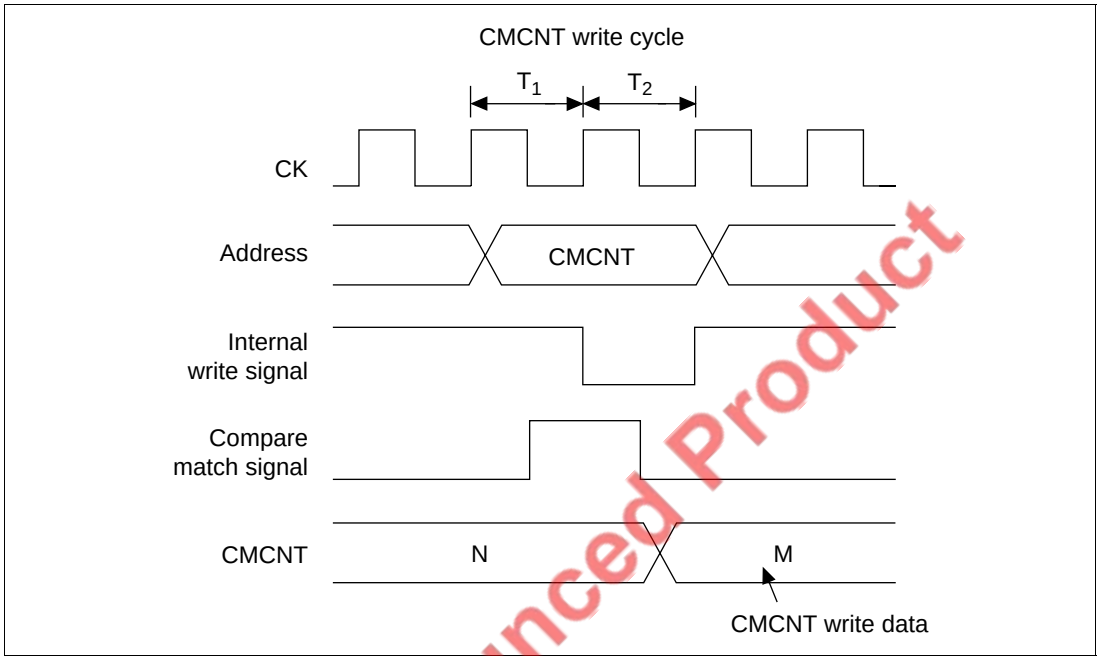


Figure 10.7 CMCNT Word Write and Increment Contention

10.5.3 Contention between CMCNT Byte Write and Incrementation

If an increment occurs during the T_2 state of the CMCNT byte write cycle, the counter write has priority, so no increment of the write data results on the writing side. The byte data on the side not performing the writing is also not incremented, so the contents are those before the write.

Figure 10.8 shows the timing when an increment occurs during the T_2 state of the CMCNTH write cycle.

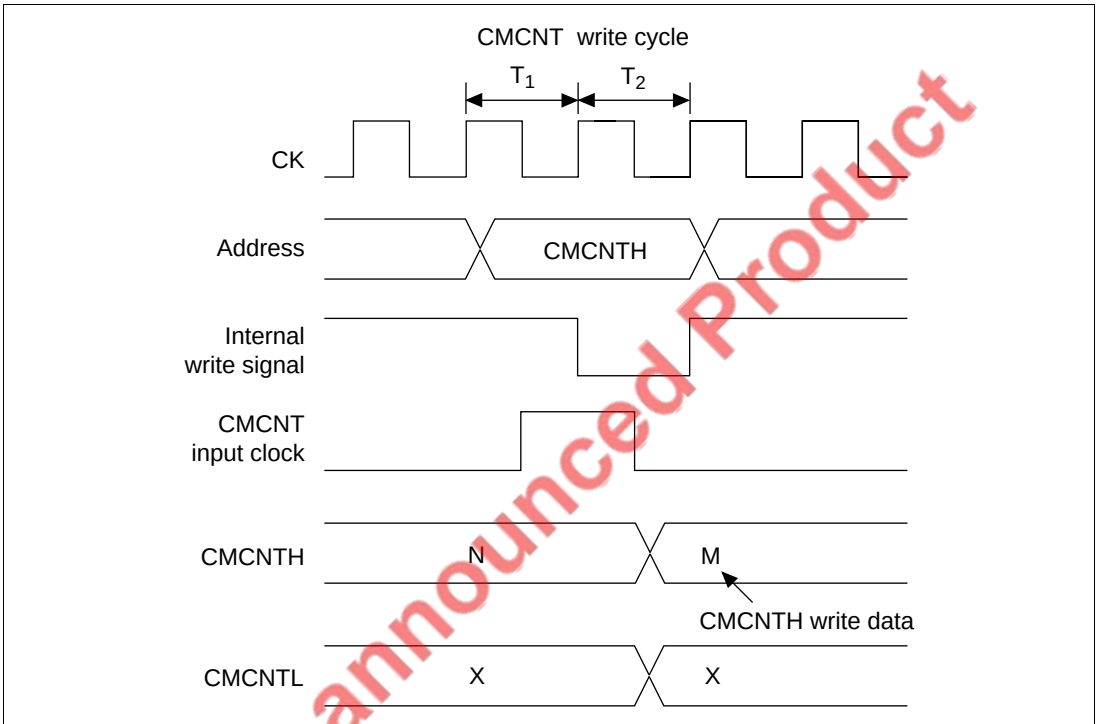


Figure 10.8 CMCNT Byte Write and Increment Contention

Section 11 Watchdog Timer (WDT)

11.1 Overview

The watchdog timer (WDT) is a 1-channel timer for monitoring system operations. If the WDT overflows without being rewritten correctly by the CPU due to system runaway or the like, it can generate an internal reset signal for the chip.

When the watchdog function is not needed, the WDT can be used as an interval timer. In the interval timer operation, an interval timer interrupt is generated at each counter overflow. The WDT is also used in recovering from the standby mode.

11.1.1 Features

- Works in watchdog timer mode or interval timer mode.
- If the counter overflows in the watchdog timer mode, WDT can generate an internal reset signal for the chip.
- Generates interrupts in the interval timer mode. When the counter overflows, it generates an interval timer interrupt.
- Clears standby mode.
- Works with eight counter input clocks.

11.1.2 Block Diagram

Figure 11.1 is the block diagram of the WDT.

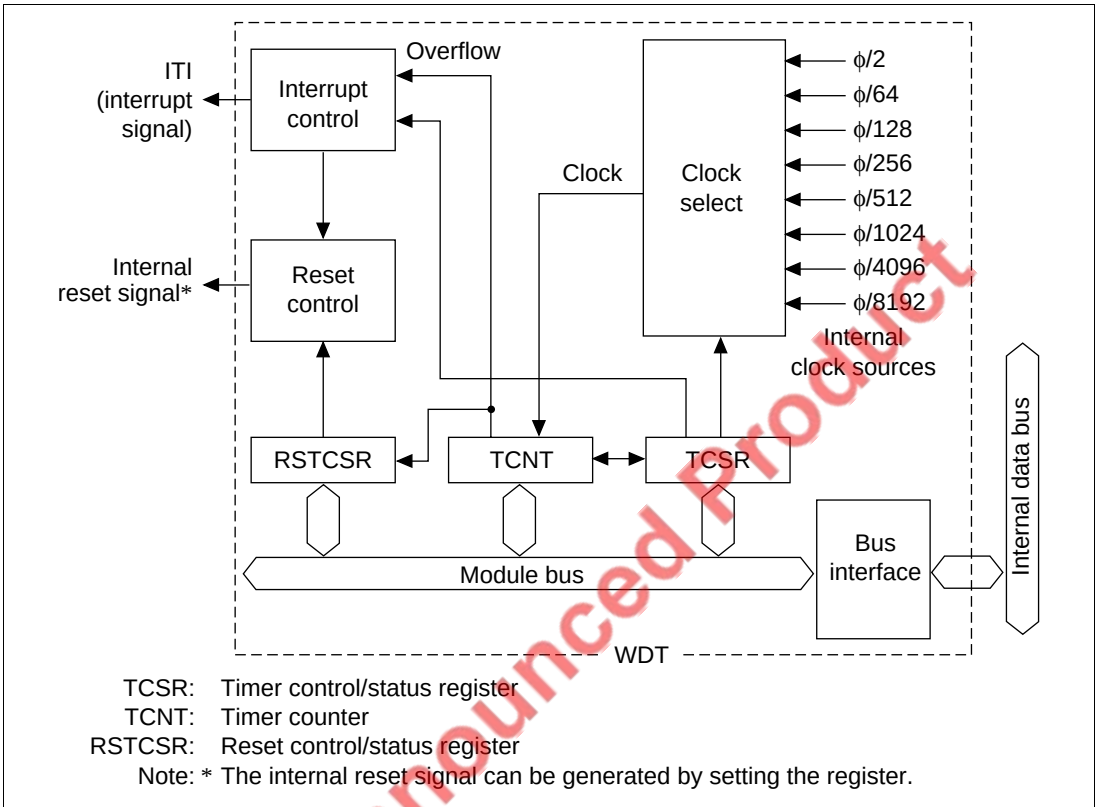


Figure 11.1 WDT Block Diagram

11.1.3 Register Configuration

Table 11.1 summarizes the three WDT registers. They are used to select the clock, switch the WDT mode, and control the reset signal.

Table 11.1 WDT Registers

Name	Abbreviation	R/W	Initial Value	Address	
				Write*[1]	Read*[2]
Timer control/status register	TCSR	R/(W)*[3]	H'18	H'FFFF8610	H'FFFF8610
Timer counter	TCNT	R/W	H'00		H'FFFF8611
Reset control/status register	RSTCSR	R/(W)*[3]	H'1F	H'FFFF8612	H'FFFF8613

Notes: 1. Write by word transfer. It cannot be written in byte or longword.
2. Read by byte transfer. It cannot be read in word or longword.
3. Only 0 can be written in bit 7 to clear the flag.

11.2 Register Descriptions

11.2.1 Timer Counter (TCNT)

The TCNT is an 8-bit read/write upcounter. (The TCNT differs from other registers in that it is more difficult to write to. See section 11.2.4, Register Access, for details.) When the timer enable bit (TME) in the timer control/status register (TCSR) is set to 1, the watchdog timer counter starts counting pulses of an internal clock selected by clock select bits 2 to 0 (CKS2 to CKS0) in the TCSR. When the value of the TCNT overflows (changes from H'FF to H'00), an internal reset signal from the watchdog timer* or interval timer interrupt (ITI) is generated, depending on the mode selected in the WT/$\overline{IT}$ bit of the TCSR.

The TCNT is initialized to H'00 by a power-on reset and when the TME bit is cleared to 0. It is not initialized in the standby mode.

Note: * If RSTE of RSTCSR is set to 1.

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

11.2.2 Timer Control/Status Register (TCSR)

The timer control/status register (TCSR) is an 8-bit read/write register. (The TCSR differs from other registers in that it is more difficult to write to. See section 11.2.4, Register Access, for details.) Its functions include selecting the clock input source and mode of the timer counter (TCNT).

Bits 7 to 5 are initialized to 000 by a power-on reset or in standby mode. Bits 2 to 0 are initialized to 000 by a power-on reset, but retain their values in the standby mode.

Bit:	7	6	5	4	3	2	1	0
	OVF	WT/$\overline{IT}$	TME	—	—	CKS2	CKS1	CKS0
Initial value:	0	0	0	1	1	0	0	0
R/W:	R/(W)	R/W	R/W	R	R	R/W	R/W	R/W

- Bit 7—Overflow Flag (OVF): Indicates that the TCNT has overflowed from H'FF to H'00 in the interval timer mode. It is not set in the watchdog timer mode.

Bit 7: OVF	Description
0	No overflow of TCNT in interval timer mode (Initial value) Cleared by reading OVF, then writing 0 in OVF
1	TCNT overflow in the interval timer mode

- Bit 6—Timer Mode Select (WT/$\overline{IT}$): Selects whether to use the WDT as a watchdog timer or interval timer.

Bit 6: WT/$\overline{IT}$	Description
0	Interval timer mode (Initial value)
1	Watchdog timer mode: $\overline{WDTOVF}$ signal output externally when TCNT overflows. (Section 11.2.3, Reset Control/Status Register (RSTCSR), describes in detail what happens when TCNT overflows in the watchdog timer mode.)

- Bit 5—Timer Enable (TME): Enables or disables the timer.

Bit 5: TME	Description
0	Timer disabled: TCNT is initialized to H'00 and count-up stops (Initial value)
1	Timer enabled: TCNT starts counting.

- Bits 4 and 3—Reserved: These bits are always read as 1. The write value should always be 1.

- Bits 2 to 0: Clock Select 2 to 0 (CKS2 to CKS0): These bits select one of eight internal clock sources for input to the TCNT. The clock signals are obtained by dividing the frequency of the system clock (ϕ).

			Description	
Bit 2: CKS2	**Bit 1: CKS1**	**Bit 0: CKS0**	**Clock Source**	**Overflow Interval* (ϕ = 20.0 MHz)**
0	0	0	ϕ/2 (Initial value)	25.6 µs
		1	ϕ/64	819.2 µs
	1	0	ϕ/128	1.6384 ms
		1	ϕ/256	3.2768 ms
1	0	0	ϕ/512	6.5536 ms
		1	ϕ/1024	13.1072 ms
	1	0	ϕ/4096	52.4288 ms
		1	ϕ/8192	104.8576 ms

Note: * The overflow interval listed is the time from when the TCNT begins counting at H'00 until an overflow occurs.

11.2.3 Reset Control/Status Register (RSTCSR)

The RSTCSR is an 8-bit readable and writable register. (The RSTCSR differs from other registers in that it is more difficult to write. See section 11.2.4, Register Access, for details.) It controls output of the internal reset signal generated by timer counter (TCNT) overflow and selects the internal reset signal type. RSTCR is initialized to H'1F by input of a reset signal from the $\overline{\text{RES}}$ pin, but is not initialized by the internal reset signal generated by the overflow of the WDT. It is initialized to H'1F in standby mode.

Bit:	7	6	5	4	3	2	1	0
	WOVF	RSTE	—	—	—	—	—	—
Initial value:	0	0	0	1	1	1	1	1
R/W:	R/(W)*	R/W	R	R	R	R	R	R

Note: * Only 0 can be written in bit 7 to clear the flag.

- Bit 7—Watchdog Timer Overflow Flag (WOVF): Indicates that the TCNT has overflowed (H'FF to H'00) in the watchdog timer mode. It is not set in the interval timer mode.

Bit 7: WOVF	Description
0	No TCNT overflow in watchdog timer mode (Initial value) Cleared when software reads WOVF, then writes 0 in WOVF
1	Set by TCNT overflow in watchdog timer mode

- Bit 6—Reset Enable (RSTE): Selects whether to reset the chip internally if the TCNT overflows in the watchdog timer mode.

Bit 6: RSTE	Description
0	Not reset when TCNT overflows (Initial value) LSI not reset internally, but TCNT and TCSR reset within WDT.
1	Reset when TCNT overflows

- Bit 5—Reserved: This bit is always read as 0. The write value should always be 0.
- Bits 4 to 0—Reserved: These bits are always read as 1. The write value should always be 1.

11.2.4 Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in that they are more difficult to write to. The procedures for writing and reading these registers are given below.

Writing to the TCNT and TCSR: These registers must be written by a word transfer instruction. They cannot be written by byte transfer instructions.

The TCNT and TCSR both have the same write address. The write data must be contained in the lower byte of the written word. The upper byte must be H'5A (for the TCNT) or H'A5 (for the TCSR) (figure 11.2). This transfers the write data from the lower byte to the TCNT or TCSR.

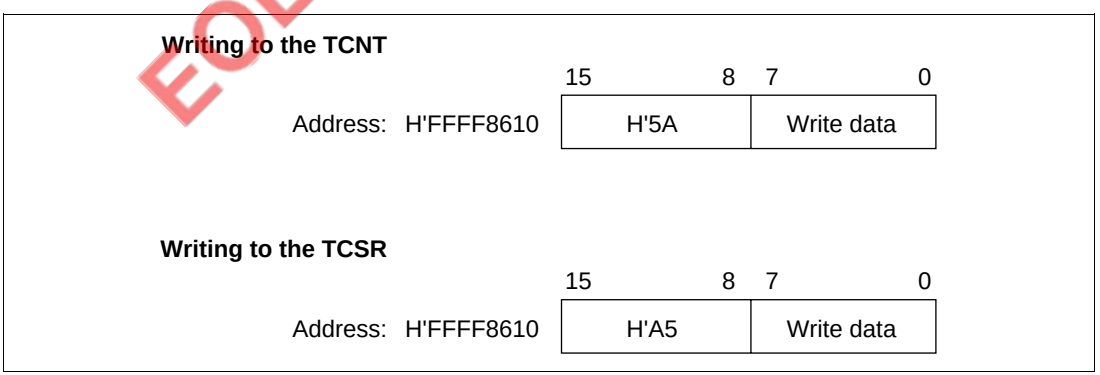


Figure 11.2 Writing to the TCNT and TCSR

Writing to the RSTCSR: The RSTCSR must be written by a word access to address H'FFFF8612. It cannot be written by byte transfer instructions.

Procedures for writing 0 in WOVF (bit 7) and for writing to RSTE (bit 6) and RSTS (bit 5) are different, as shown in figure 11.3.

To write 0 in the WOVF bit, the write data must be H'A5 in the upper byte and H'00 in the lower byte. This clears the WOVF bit to 0. The RSTE and RSTS bits are not affected. To write to the RSTE and RSTS bits, the upper byte must be H'5A and the lower byte must be the write data. The values of bits 6 and 5 of the lower byte are transferred to the RSTE and RSTS bits, respectively. The WOVF bit is not affected.

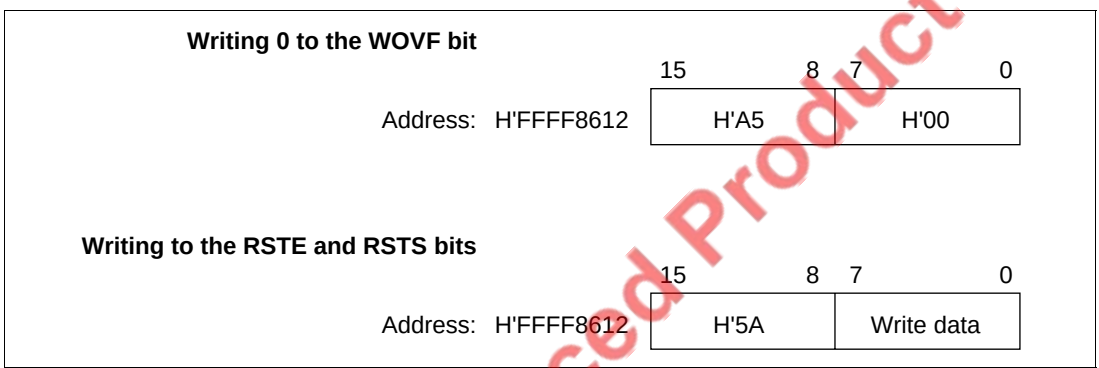


Figure 11.3 Writing to the RSTCSR

Reading from the TCNT, TCSR, and RSTCSR: TCNT, TCSR, and RSTCSR are read like other registers. Use byte transfer instructions. The read addresses are H'FFFF8610 for the TCSR, H'FFFF8611 for the TCNT, and H'FFFF8613 for the RSTCSR.

11.3 Operation

11.3.1 Watchdog Timer Mode

To use the WDT as a watchdog timer, set the WT/$\overline{IT}$ and TME bits of the TCSR to 1. Software must prevent TCNT overflow by rewriting the TCNT value (normally by writing H'00) before overflow occurs. No TCNT overflows will occur while the system is operating normally, but if RSTE of RSTCSR is set to 1 and a problem such as system runaway occurs, the value of TCNT is not overwritten and an overflow results. This causes WDT to generate an internal reset signal for the chip. The internal reset signal is output for 512 ϕ clock cycles. Figure 11.4 shows the timing.

When a watchdog overflow reset is generated simultaneously with a reset input at the $\overline{RES}$ pin, the $\overline{RES}$ reset takes priority, and the WOVF bit is cleared to 0.

The following are not initialized a WDT reset signal:

- PFC (Pin Function Controller) function register
- I/O port register

Initializing is only possible by external power-on reset.

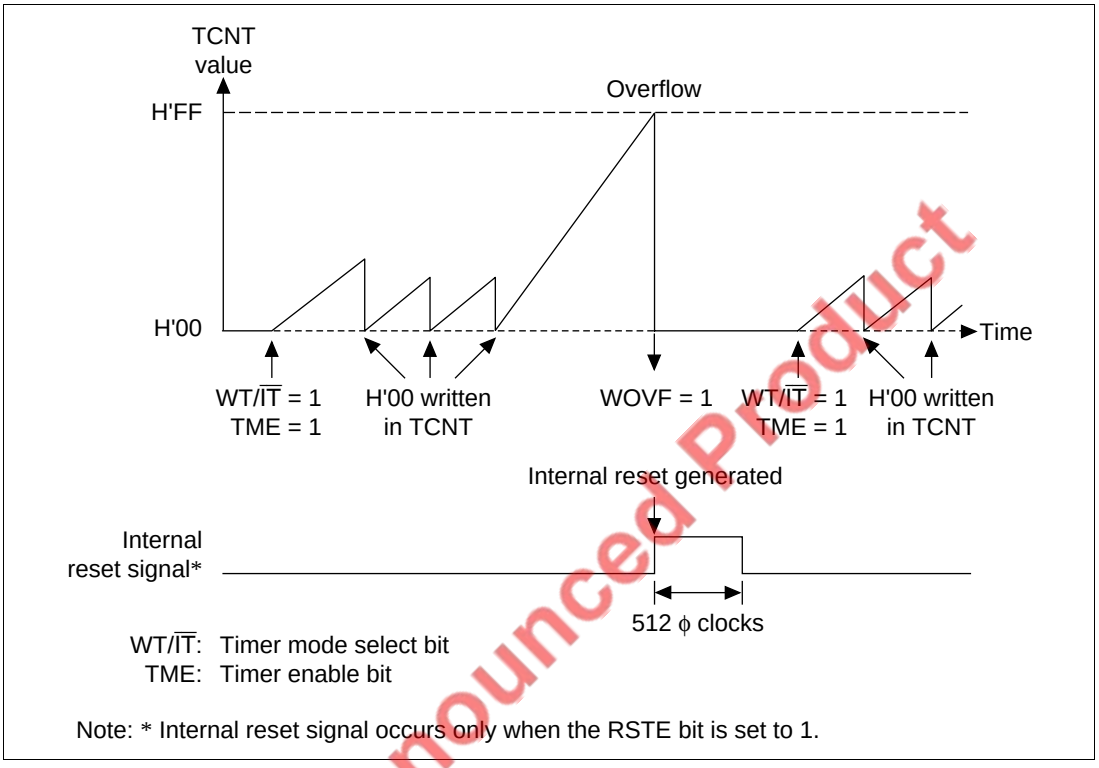


Figure 11.4 Operation in the Watchdog Timer Mode

11.3.2 Interval Timer Mode

To use the WDT as an interval timer, clear WT/$\overline{IT}$ to 0 and set TME to 1. An interval timer interrupt (ITI) is generated each time the timer counter overflows. This function can be used to generate interval timer interrupts at regular intervals (figure 11.5).

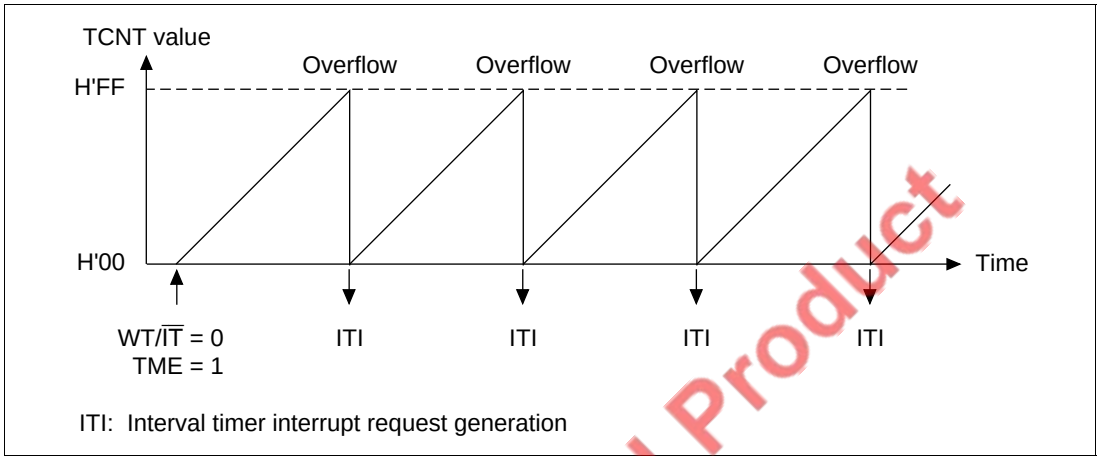


Figure 11.5 Operation in the Interval Timer Mode

11.3.3 Clearing the Standby Mode

The watchdog timer has a special function to clear the standby mode with an NMI interrupt. When using the standby mode, set the WDT as described below.

Before Transition to the Standby Mode: The TME bit in the TCSR must be cleared to 0 to stop the watchdog timer counter before it enters the standby mode. The chip cannot enter the standby mode while the TME bit is set to 1. Set bits CKS2 to CKS0 so that the counter overflow interval is equal to or longer than the oscillation settling time. See section 19.3, AC Characteristics, for the oscillation settling time.

Recovery from the Standby Mode: When an NMI request signal is received in standby mode, the clock oscillator starts running and the watchdog timer starts incrementing at the rate selected by bits CKS2 to CKS0 before the standby mode was entered. When the TCNT overflows (changes from H'FF to H'00), the clock is presumed to be stable and usable; clock signals are supplied to the entire chip and the standby mode ends.

For details on the standby mode, see section 18, Power Down State.

11.3.4 Timing of Setting the Overflow Flag (OVF)

In the interval timer mode, when the TCNT overflows, the OVF flag of the TCSR is set to 1 and an interval timer interrupt is simultaneously requested (figure 11.6).

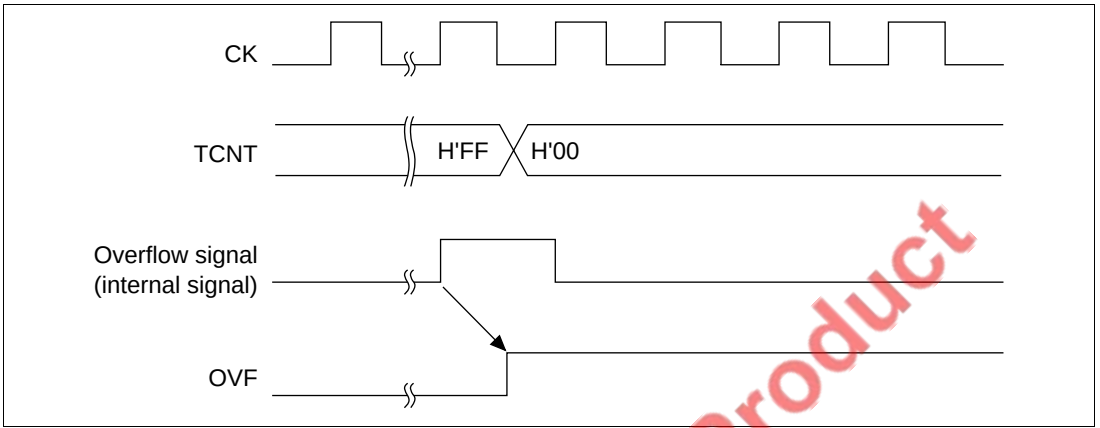


Figure 11.6 Timing of Setting the OVF

11.3.5 Timing of Setting the Watchdog Timer Overflow Flag (WOVF)

If the timer counter (TCNT) overflows in the watchdog timer mode, the WOVF bit of the reset control/status register (RSTCSR) is set to 1. If the RSTE bit of RSTCSR is set to 1, an internal reset for the entire chip is generated when TCNT overflows. Figure 11.7 shows the timing.

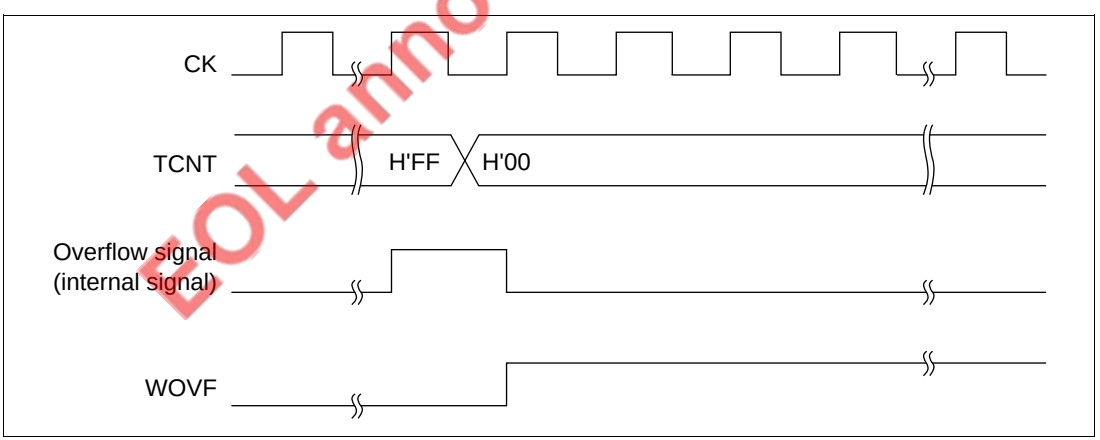


Figure 11.7 Timing of Setting the WOVF Bit

11.4 Notes on Use

11.4.1 TCNT Write and Increment Contention

If a timer counter (TCNT) increment clock pulse is generated during the T_3 state of a write cycle to the TCNT, the write takes priority and the timer counter is not incremented (figure 11.8).

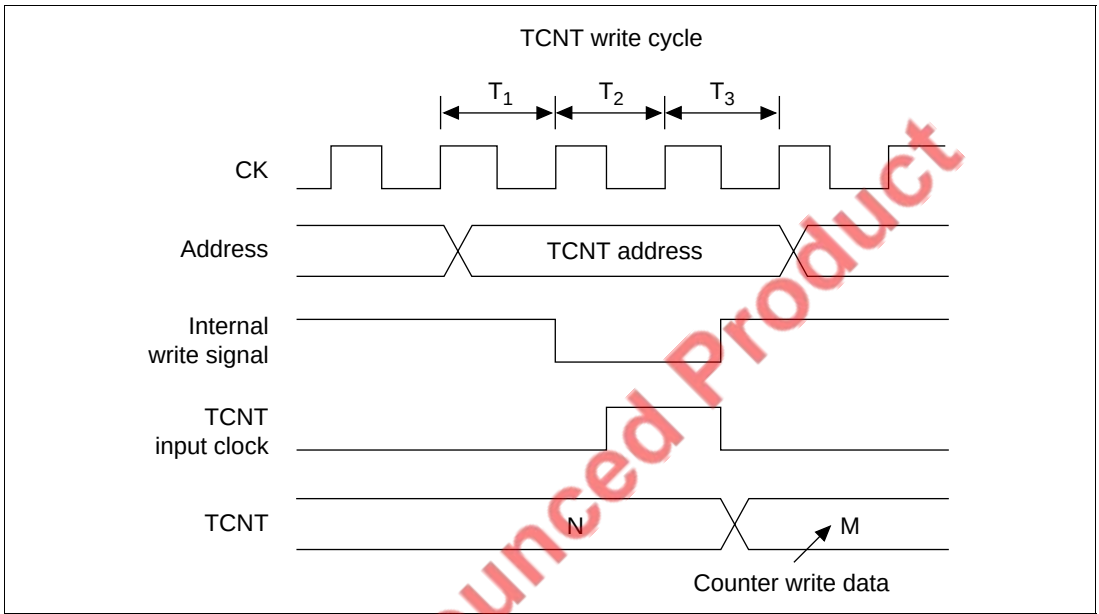


Figure 11.8 Contention between TCNT Write and Increment

11.4.2 Changing CKS2 to CKS0 Bit Values

If the values of bits CKS2 to CKS0 are altered while the WDT is running, the count may increment incorrectly. Always stop the watchdog timer (by clearing the TME bit to 0) before changing the values of bits CKS2 to CKS0.

11.4.3 Changing between Watchdog Timer/Interval Timer Modes

To prevent incorrect operation, always stop the watchdog timer (by clearing the TME bit to 0) before switching between interval timer mode and watchdog timer mode.

11.4.4 Internal Reset With the Watchdog Timer

If the RSTE bit is cleared to 0 in the watchdog timer mode, the LSI will not reset internally when a TCNT overflow occurs, but the TCNT and TCSR in the WDT will reset.

Section 12 Serial Communication Interface (SCI1)

12.1 Overview

The SH7018 has a serial communication interface (SCI1) with one channel.

The SCI supports asynchronous serial communication. It also has a multiprocessor communication function for serial communication among two or more processors.

12.1.1 Features

- Select asynchronous or clock synchronous as the serial communications mode.
  - — Asynchronous mode: Serial data communications are synched by start-stop in character units. The SCI1 can communicate with a universal asynchronous receiver/transmitter (UART), an asynchronous communication interface adapter (ACIA), or any other chip that employs a standard asynchronous serial communication. It can also communicate with two or more other processors using the multiprocessor communication function. There are twelve selectable serial data communication formats.
    - Data length: seven or eight bits
    - Stop bit length: one or two bits
    - Parity: even, odd, or none
    - Multiprocessor bit: one or none
    - Receive error detection: parity, overrun, and framing errors
    - Break detection: by reading the RxD level directly when a framing error occurs
  - — Clock synchronous mode: Serial data transfer is synchronized with the clock. This permits serial data communications with other LSI devices equipped with a clock synchronous communications function. One serial data communications format is supported.
    - Number of data bits: 8
    - Receive error detection: Over-line error detection
- Full duplex communication: The transmitting and receiving sections are independent, so the SCI can transmit and receive simultaneously. Both sections use double buffering, so continuous data transfer is possible in both the transmit and receive directions.
- On-chip baud rate generator with selectable bit rates.
- Internal transmit/receive clock source: baud rate generator (internal).
- Four types of interrupts: Transmit-data-empty, transmit-end, receive-data-full, and receive-error interrupts are requested independently.

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the SCI1.

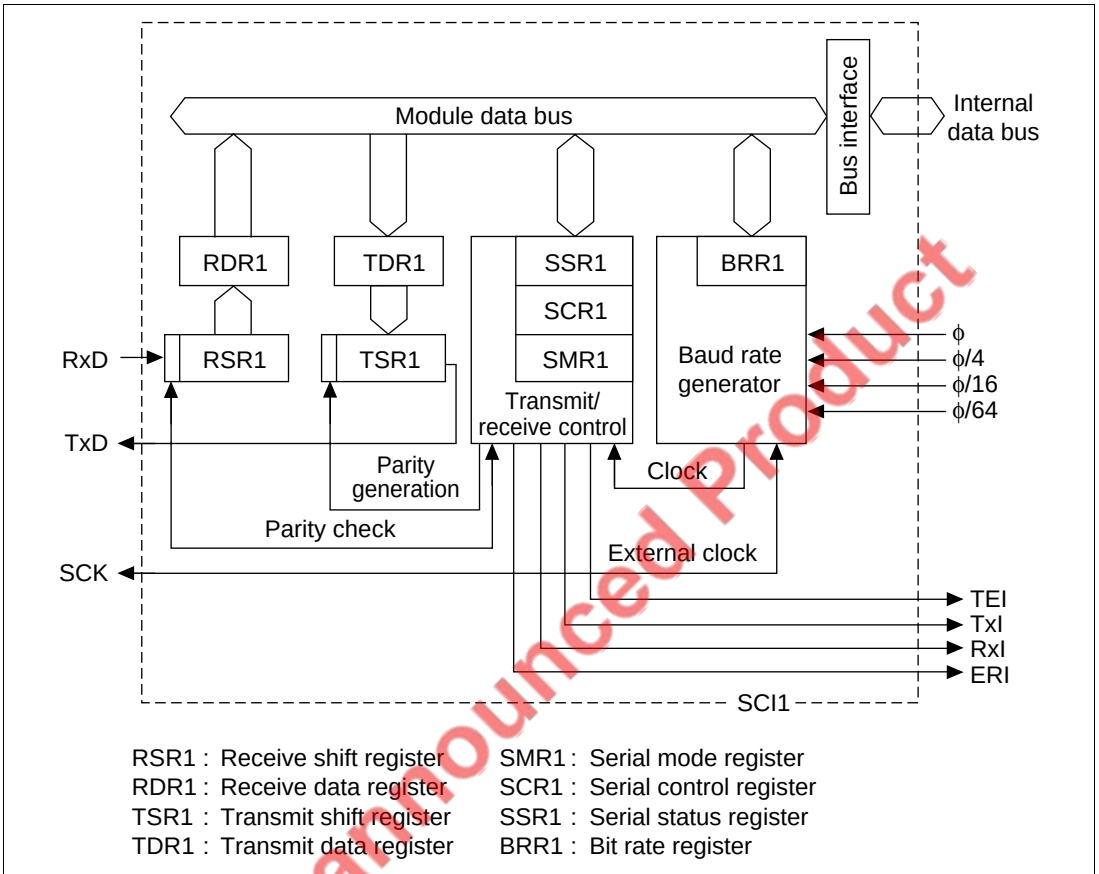


Figure 12.1 SCI1 Block Diagram

12.1.3 Pin Configuration

Table 12.1 summarizes the SCI1 pins by channel.

Table 12.1 SCI1 Pins

Pin Name	Abbreviation	Input/Output	Function
Serial clock pin	SCK	Input/output	SCI1 clock input/output
Receive data pin	RxD	Input	SCI1 receive data input
Transmit data pin	TxD	Output	SCI1 transmit data output

12.1.4 Register Configuration

Table 12.2 summarizes the SCI1 internal registers. These registers specify the data format and bit rate, and control the transmitter and receiver sections.

Table 12.2 Registers

Name	Abbreviation	R/W	Initial Value	Address*[1]	Access Size
Serial mode register	SMR1	R/W	H'00	H'FFFF81B0	8, 16
Bit rate register	BRR1	R/W	H'FF	H'FFFF81B1	8, 16
Serial control register	SCR1	R/W	H'00	H'FFFF81B2	8, 16
Transmit data register	TDR1	R/W	H'FF	H'FFFF81B3	8, 16
Serial status register	SSR1	R/(W)*[2]	H'84	H'FFFF81B4	8, 16
Receive data register	RDR1	R	H'00	H'FFFF81B5	8, 16

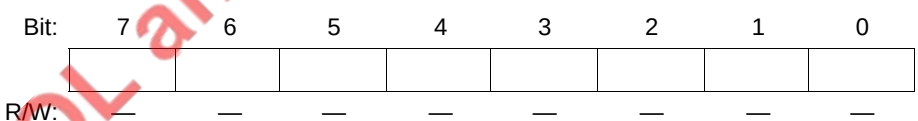
Notes: 1. Do not attempt to access empty addresses.
2. The only value that can be written is a 0 to clear the flags.

12.2 Register Descriptions

12.2.1 Receive Shift Register (RSR1)

The receive shift register (RSR1) receives serial data. Data input at the RxD pin is loaded into the RSR1 in the order received, LSB (bit 0) first, converting the data to parallel form. When one byte has been received, it is automatically transferred to the RDR1.

The CPU cannot read or write the RSR1 directly.



12.2.2 Receive Data Register (RDR1)

The receive data register (RDR1) stores serial receive data. The SCI1 completes the reception of one byte of serial data by moving the received data from the receive shift register (RSR1) into the RDR1 for storage. The RSR1 is then ready to receive the next data. This double buffering allows the SCI1 to receive data continuously.

The CPU can read but not write the RDR1. The RDR is initialized to H'00 by a power-on reset or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

12.2.3 Transmit Shift Register (TSR1)

The transmit shift register (TSR1) transmits serial data. The SCI1 loads transmit data from the transmit data register (TDR1) into the TSR1, then transmits the data serially from the TxD pin, LSB (bit 0) first. After transmitting one data byte, the SCI1 automatically loads the next transmit data from the TDR1 into the TSR1 and starts transmitting again. If the TDRE bit of the SSR1 is 1, however, the SCI1 does not load the TDR1 contents into the TSR1.

The CPU cannot read or write the TSR1 directly.

Bit:	7	6	5	4	3	2	1	0
R/W:	—	—	—	—	—	—	—	—

12.2.4 Transmit Data Register (TDR1)

The transmit data register (TDR1) is an 8-bit register that stores data for serial transmission. When the SCI1 detects that the transmit shift register (TSR1) is empty, it moves transmit data written in the TDR1 into the TSR1 and starts serial transmission. Continuous serial transmission is possible by writing the next transmit data in the TDR1 during serial transmission from the TSR1.

The CPU can always read and write the TDR1. The TDR1 is initialized to H'FF by a power-on reset or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12.2.5 Serial Mode Register (SMR1)

The serial mode register (SMR1) is an 8-bit register that specifies the SCI1 serial communication format and selects the clock source for the baud rate generator.

The CPU can always read and write the SMR1. The SMR1 is initialized to H'00 by a power-on reset.

Bit:	7	6	5	4	3	2	1	0
	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	MP	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- Bit 7—Communication Mode (C/$\overline{A}$): Sets the SCI operation mode to either start-stop synchronous mode or clock synchronous mode.

Bit 7: C/$\overline{A}$	Description	
0	Start-stop synchronous mode	(Initial value)
1	Clock synchronous mode	

- Bit 6—Character Length (CHR): Selects 7-bit or 8-bit data in the asynchronous mode. The number of data bits is fixed at eight in the clock synchronous mode, regardless of the CHR setting.

Bit 6: CHR	Description	
0	Eight-bit data	(Initial value)
1	Seven-bit data When 7-bit data is selected, the MSB (bit 7) of the transmit data register is not transmitted.	

- Bit 5—Parity Enable (PE): Selects whether to add a parity bit to transmit data and to check the parity of receive data. A parity bit is added in the clock synchronous mode regardless of the setting of the PE bit, and no checking is performed.

Bit 5: PE	Description	
0	Parity bit not added or checked	(Initial value)
1	Parity bit added and checked When PE is set to 1, an even or odd parity bit is added to transmit data, depending on the parity mode (O/$\overline{E}$) setting. Receive data parity is checked according to the even/odd (O/$\overline{E}$) mode setting.	

- Bit 4—Parity Mode (O/$\overline{E}$): Selects even or odd parity when parity bits are added and checked. The O/$\overline{E}$ setting is used only when the parity enable bit (PE) is set to 1 to enable parity addition and check. The O/$\overline{E}$ setting is ignored when parity addition and check is disabled.

Bit 4: O/$\overline{E}$	Description
0	Even parity (Initial value) If even parity is selected, the parity bit is added to transmit data to make an even number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an even number of 1s in the received character and parity bit combined.
1	Odd parity If odd parity is selected, the parity bit is added to transmit data to make an odd number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an odd number of 1s in the received character and parity bit combined.

- Bit 3—Stop Bit Length (STOP): Selects one or two bits as the stop bit length.

  In receiving, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit, but if the second stop bit is 0, it is treated as the start bit of the next incoming character.

Bit 3: STOP	Description
0	One stop bit (Initial value) In transmitting, a single bit of 1 is added at the end of each transmitted character.
1	Two stop bits In transmitting, two bits of 1 are added at the end of each transmitted character.

In receiving, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, is it treated as a stop bit, but if the second stop bit is 0, it is treated as the start bit of the next character to be transmitted.

- Bit 2—Multiprocessor Mode (MP): Selects multiprocessor format. When multiprocessor format is selected, settings of the parity enable (PE) and parity mode (O/$\overline{E}$) bits are ignored. Also, the setting of the MP bit is valid only in the start-stop synchronous mode. The setting of the MP bit is ignored in the clock synchronous mode. For the multiprocessor communication function, see section 12.3.3, Multiprocessor Communication.

Bit 2: MP	Description
0	Multiprocessor function disabled (Initial value)
1	Multiprocessor format selected

- Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the internal clock source of the on-chip baud rate generator. Four clock sources are available; ϕ, $\phi/4$, $\phi/16$, or $\phi/64$. For

further information on the clock source, bit rate register settings, and baud rate, see section 12.2.8, Bit Rate Register.

Bit 1: CKS1	Bit 0: CKS0	Description
0	0	φ (Initial value)
	1	φ/4
1	0	φ/16
	1	φ/64

12.2.6 Serial Control Register (SCR1)

The serial control register (SCR1) operates the SCI1 transmitter/receiver, enables/disables interrupt requests. The CPU can always read and write the SCR1. The SCR1 is initialized to H'00 by a power-on reset.

Bit:	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- Bit 7—Transmit Interrupt Enable (TIE): Enables or disables the transmit-data-empty interrupt (TxI) requested when the transmit data register empty bit (TDRE) in the serial status register (SSR1) is set to 1 by transfer of serial transmit data from the TDR1 to the TSR1.

Bit 7: TIE	Description
0	Transmit-data-empty interrupt request (TxI) is disabled (Initial value) The TxI interrupt request can be cleared by reading TDRE after it has been set to 1, then clearing TDRE to 0, or by clearing TIE to 0.
1	Transmit-data-empty interrupt request (TxI) is enabled

- Bit 6—Receive Interrupt Enable (RIE): Enables or disables the receive-data-full interrupt (RxI) requested when the receive data register full bit (RDRF) in the serial status register (SSR1) is set to 1 by transfer of serial receive data from the RSR1 to the RDR1. It also enables or disables receive-error interrupt (ERI) requests.

Bit 6: RIE	Description
0	Receive-data-full interrupt (RxI) and receive-error interrupt (ERI) requests are disabled. (Initial value) RxI and ERI interrupt requests can be cleared by reading the RDRF flag or error flag (FER, PER, or ORER) after it has been set to 1, then clearing the flag to 0, or by clearing RIE to 0.
1	Receive-data-full interrupt (RxI) and receive-error interrupt (ERI) requests are enabled.

- Bit 5—Transmit Enable (TE): Enables or disables the SCI1 serial transmitter.

Bit 5: TE	Description
0	Transmitter disabled (Initial value) The transmit data register empty bit (TDRE) in the serial status register (SSR1) is locked at 1.
1	Transmitter enabled Serial transmission starts when the transmit data register empty (TDRE) bit in the serial status register (SSR1) is cleared to 0 after writing of transmit data into the TDR1. Select the transmit format in the SMR1 before setting TE to 1.

- Bit 4—Receive Enable (RE): Enables or disables the SCI1 serial receiver.

Bit 4: RE	Description
0	Receiver disabled (Initial value) Clearing RE to 0 does not affect the receive flags (RDRF, FER, PER, ORER). These flags retain their previous values.
1	Receiver enabled Serial reception starts when a start bit is detected in the asynchronous mode, or synchronous clock input is detected in the clock synchronous mode. Select the receive format in the SMR1 before setting RE to 1.

- Bit 3—Multiprocessor Interrupt Enable (MPIE): Enables or disables multiprocessor interrupts. The MPIE setting is used only if the multiprocessor mode bit (MP) in the serial mode register (SMR1) is set to 1 during reception.

Bit 3: MPIE	Description
0	Multiprocessor interrupts are disabled (normal receive operation) (Initial value) MPIE is cleared when the MPIE bit is cleared to 0, or the multiprocessor bit (MPB) is set to 1 in receive data.
1	Multiprocessor interrupts are enabled Receive-data-full interrupt requests (RxI), receive-error interrupt requests (ERI), and setting of the RDRF, FER, and ORER status flags in the serial status register (SSR1) are disabled until data with the multiprocessor bit set to 1 is received. The SCI does not transfer receive data from the RSR1 to the RDR1, does not detect receive errors, and does not set the RDRF, FER, and ORER flags in the serial status register (SSR1). When it receives data that includes MPB = 1, MPB is set to 1, and the SCI1 automatically clears MPIE to 0, generates RxI and ERI interrupts (if the TIE and RIE bits in the SCR1 are set to 1), and allows the FER and ORER bits to be set.

- Bit 2—Transmit-End Interrupt Enable (TEIE): Enables or disables the transmit-end interrupt (TEI) requested if TDR does not contain valid transmit data when the MSB is transmitted.

Bit 2: TEIE	Description
0	Transmit-end interrupt (TEI) requests are disabled* (Initial value)
1	Transmit-end interrupt (TEI) requests are enabled.*

Note: * The TEI request can be cleared by reading the TDRE bit in the serial status register (SSR1) after it has been set to 1, then clearing TDRE to 0 and clearing the transmit end (TEND) bit to 0; or by clearing the TEIE bit to 0.

- Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits select the SCI1 clock source and enable or disable clock output from the SCK pin. Depending on the combination of CKE1 and CKE0, the SCK pin can be used for serial clock output, or serial clock input. Select the SCK pin function by using the pin function controller (PFC).

  The CKE0 setting is valid only in the asynchronous mode, and only when the SCI1 is internally clocked (CKE1 = 0). The CKE0 setting is ignored in the clock synchronous mode, or when an external clock source is selected (CKE1 = 1). Select the SCI1 operating mode in the serial mode register (SMR) before setting CKE1 and CKE0. For further details on selection of the SCI1 clock source, see table 12.9 in section 12.3, Operation.

Bit 1: CKE1	Bit 0: CKE0	Description[*1]	
0	0	Asynchronous mode	Internal clock, SCK pin used for input pin (input signal is ignored) or output pin (output level is undefined)[*2]
		Clock synchronous mode	Internal clock, SCK pin used for synchronous clock output[*2]
0	1	Asynchronous mode	Internal clock, SCK pin used for clock output[*3]
		Clock synchronous mode	Internal clock, SCK pin used for synchronous clock output
1	0	Asynchronous mode	External clock, SCK pin used for clock input[*4]
		Clock synchronous mode	External clock, SCK pin used for synchronous clock input
1	1	Asynchronous mode	External clock, SCK pin used for clock input[*4]
		Clock synchronous mode	External clock, SCK pin used for synchronous clock input

Notes: 1. The SCK pin is multiplexed with other functions. Use the pin function controller (PFC) to select the SCK function for this pin, as well as the I/O direction.
2. Initial value.
3. The output clock frequency is the same as the bit rate.
4. The input clock frequency is 16 times the bit rate.

12.2.7 Serial Status Register (SSR1)

The serial status register (SSR1) is an 8-bit register containing multiprocessor bit values, and status flags that indicate SCI1 operating status.

The CPU can always read and write the SSR1, but cannot write 1 in the status flags (TDRE, RDRF, ORER, PER, and FER). These flags can be cleared to 0 only if they have first been read (after being set to 1). Bits 2 (TEND) and 1 (MPB) are read-only bits that cannot be written. The SSR1 is initialized to H'84 by a power-on reset or in standby mode.

Bit:	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value:	1	0	0	0	0	1	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: * The only value that can be written is a 0 to clear the flag.

- Bit 7—Transmit Data Register Empty (TDRE): Indicates that the SCI1 has loaded transmit data from the TDR1 into the TSR1 and new serial transmit data can be written in the TDR1.

Bit 7: TDRE	Description
0	TDR1 contains valid transmit data TDRE is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE.
1	TDR1 does not contain valid transmit data (Initial value) TDRE is set to 1 when the chip is power-on reset or in standby mode, the TE bit in the serial control register (SCR1) is cleared to 0, or TDR1 contents are loaded into TSR1, so new data can be written in TDR1.

- Bit 6—Receive Data Register Full (RDRF): Indicates that RDR1 contains received data.

Bit 6: RDRF	Description
0	RDR1 does not contain valid received data (Initial value) RDRF is cleared to 0 when the chip is power-on reset or in standby mode, software reads RDRF after it has been set to 1, then writes 0 in RDRF.
1	RDR1 contains valid received data RDRF is set to 1 when serial data is received normally and transferred from RSR1 to RDR1.

Note: The RDR1 and RDRF are not affected by detection of receive errors or by clearing of the RE bit to 0 in the serial control register (SCR1). They retain their previous contents. If RDRF is still set to 1 when reception of the next data ends, an overrun error (ORER) occurs and the received data is lost.

- Bit 5—Overrun Error (ORER): Indicates that data reception ended abnormally due to an overrun error.

Bit 5: ORER	Description
0	Receiving is in progress or has ended normally (Initial value) Clearing the RE bit to 0 in the serial control register (SCR1) does not affect the ORER bit, which retains its previous value. ORER is cleared to 0 when the chip is power-on reset, in standby mode, or software reads ORER after it has been set to 1, then writes 0 in ORER.
1	A receive overrun error occurred RDR1 continues to hold the data received before the overrun error, so subsequent receive data is lost. Serial receiving cannot continue while ORER is set to 1. ORER is set to 1 if reception of the next serial data ends when RDRF is set to 1.

- Bit 4—Framing Error (FER): Indicates that data reception ended abnormally due to a framing error.

Bit 4: FER	Description
0	Receiving is in progress or has ended normally (Initial value) Clearing the RE bit to 0 in the serial control register (SCR1) does not affect the FER bit, which retains its previous value. FER is cleared to 0 when the chip is power-on reset, in standby mode, or software reads FER after it has been set to 1, then writes 0 in FER.
1	A receive framing error occurred When the stop bit length is two bits, only the first bit is checked to see if it is a 1. The second stop bit is not checked. When a framing error occurs, the SCI1 transfers the receive data into the RDR but does not set RDRF. Serial receiving cannot continue while FER is set to 1. FER is set to 1 if the stop bit at the end of receive data is checked and found to be 0.

- Bit 3—Parity Error (PER): Indicates that data reception (with parity) ended abnormally due to a parity error.

Bit 3: PER	Description
0	Receiving is in progress or has ended normally (Initial value) Clearing the RE bit to 0 in the serial control register (SCR1) does not affect the PER bit, which retains its previous value. PER is cleared to 0 when the chip is power-on reset, in standby mode, or software reads PER after it has been set to 1, then writes 0 in PER.
1	A receive parity error occurred When a parity error occurs, the SCI1 transfers the receive data into the RDR1 but does not set RDRF. Serial receiving cannot continue while PER is set to 1. PER is set to 1 if the number of 1s in receive data, including the parity bit, does not match the even or odd parity setting of the parity mode bit (O/$\overline{E}$) in the serial mode register (SMR1).

- Bit 2—Transmit End (TEND): Indicates that when the last bit of a serial character was transmitted, the TDR1 did not contain valid data, so transmission has ended. TEND is a read-only bit and cannot be written.

Bit 2: TEND	Description
0	Transmission is in progress TEND is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE.
1	End of transmission (Initial value) TEND is set to 1 when the chip is power-on reset or in standby mode, TE is cleared to 0 in the serial control register (SCR1), or TDRE is 1 when the last bit of a one-byte serial character is transmitted.

- Bit 1—Multiprocessor Bit (MPB): Stores the value of the multiprocessor bit in receive data when a multiprocessor format is selected for receiving. The MPB is a read-only bit and cannot be written.

Bit 1: MPB	Description
0	Multiprocessor bit value in receive data is 0 (Initial value) If RE is cleared to 0 when a multiprocessor format is selected, the MPB retains its previous value.
1	Multiprocessor bit value in receive data is 1

- Bit 0—Multiprocessor Bit Transfer (MPBT): Stores the value of the multiprocessor bit added to transmit data when a multiprocessor format is selected for transmitting. The setting of the MPBT bit is ignored in the clock synchronous mode, when the multiprocessor format is not being used, and when transmission is not taking place.

Bit 0: MPBT	Description
0	Multiprocessor bit value in transmit data is 0 (Initial value)
1	Multiprocessor bit value in transmit data is 1

12.2.8 Bit Rate Register (BRR1)

The bit rate register (BRR1) is an 8-bit register that, together with the baud rate generator clock source selected by the CKS1 and CKS0 bits in the serial mode register (SMR1), determines the serial transmit/receive bit rate.

The CPU can always read and write the BRR1. The BRR1 is initialized to H'FF by a power-on reset.

Bit:	7	6	5	4	3	2	1	0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 12.3 lists examples of BRR1 settings in the asynchronous mode. Table 12.4 lists examples of BRR1 settings in the clock synchronous mode.

Table 12.3 Bit Rates and BRR1 Settings (Asynchronous Mode)

Bit Rate (Bits/s)	ϕ (MHz)								
	4			4.9152			6		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	70	0.03	2	86	0.31	2	106	–0.44
150	1	207	0.16	1	255	0.00	2	77	0.16
300	1	103	0.16	1	127	0.00	1	155	0.16
600	0	207	0.16	0	255	0.00	1	77	0.16
1200	0	103	0.16	0	127	0.00	0	155	0.16
2400	0	51	0.16	0	63	0.00	0	77	0.16
4800	0	25	0.16	0	31	0.00	0	38	0.16
9600	0	12	0.16	0	15	0.00	0	19	–2.34
14400	0	8	–3.55	0	10	–3.03	0	12	0.16
19200	0	6	–6.99	0	7	0.00	0	9	–2.34
28800	0	3	8.51	0	4	6.67	0	6	–6.99
31250	0	3	0.00	0	4	–1.70	0	5	0.00
38400	0	2	8.51	0	3	0.00	0	4	–2.34

Table 12.3 Bit Rates and BRR1 Settings (Asynchronous Mode) (cont)

Bit Rate (Bits/s)	ϕ (MHz)								
	7.3728			8			9.8304		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	130	–0.07	2	141	0.03	2	174	–0.26
150	2	95	0.00	2	103	0.16	2	127	0.00
300	1	191	0.00	1	207	0.16	1	255	0.00
600	1	95	0.00	1	103	0.16	1	127	0.00
1200	0	191	0.00	0	207	0.16	0	255	0.00
2400	0	95	0.00	0	103	0.16	0	127	0.00
4800	0	47	0.00	0	51	0.16	0	63	0.00
9600	0	23	0.00	0	25	0.16	0	31	0.00
14400	0	15	0.00	0	16	2.12	0	20	1.59
19200	0	11	0.00	0	12	0.16	0	15	0.00
28800	0	7	0.00	0	8	–3.55	0	10	–3.03
31250	0	6	5.33	0	7	0.00	0	9	–1.70
38400	0	5	0.00	0	6	–6.99	0	7	0.00

Table 12.3 Bit Rates and BRR1 Settings (Asynchronous Mode) (cont)

Bit Rate (Bits/s)	ϕ (MHz)								
	10			11.0592			12		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	177	–0.25	2	195	0.19	2	212	0.03
150	2	129	0.16	2	143	0.00	2	155	0.16
300	2	64	0.16	2	71	0.00	2	77	0.16
600	1	129	0.16	1	143	0.00	1	155	0.16
1200	1	64	0.16	1	71	0.00	1	77	0.16
2400	0	129	0.16	0	143	0.00	0	155	0.16
4800	0	64	0.16	0	71	0.00	0	77	0.16
9600	0	32	–1.36	0	35	0.00	0	38	0.16
14400	0	21	–1.36	0	23	0.00	0	25	0.16
19200	0	15	1.73	0	17	0.00	0	19	–2.34
28800	0	10	–1.36	0	11	0.00	0	12	0.16
31250	0	9	0.00	0	10	0.54	0	11	0.00
38400	0	7	1.73	0	8	0.00	0	9	–2.34

Table 12.3 Bit Rates and BRR1 Settings (Asynchronous Mode) (cont)

Bit Rate (Bits/s)	ϕ (MHz)								
	12.288			14			14.7456		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	217	0.08	2	248	–0.17	3	64	0.70
150	2	159	0.00	2	181	0.16	2	191	0.00
300	2	79	0.00	2	90	0.16	2	95	0.00
600	1	159	0.00	1	181	0.16	1	191	0.00
1200	1	79	0.00	1	90	0.16	1	95	0.00
2400	0	159	0.00	0	181	0.16	0	191	0.00
4800	0	79	0.00	0	90	0.16	0	95	0.00
9600	0	39	0.00	0	45	–0.93	0	47	0.00
14400	0	26	–1.23	0	29	1.27	0	31	0.00
19200	0	19	0.00	0	22	–0.93	0	23	0.00
28800	0	12	2.56	0	14	1.27	0	15	0.00
31250	0	11	2.40	0	13	0.00	0	14	–1.70
38400	0	9	0.00	0	10	3.57	0	11	0.00

Table 12.3 Bit Rates and BRR1 Settings (Asynchronous Mode) (cont)

Bit Rate (Bits/s)	ϕ (MHz)								
	16			17.2032			18		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	70	0.03	3	75	0.48	3	79	–0.12
150	2	207	0.16	2	223	0.00	2	233	0.16
300	2	103	0.16	2	111	0.00	2	116	0.16
600	1	207	0.16	1	223	0.00	1	233	0.16
1200	1	103	0.16	1	111	0.00	1	116	0.16
2400	0	207	0.16	0	223	0.00	0	233	0.16
4800	0	103	0.16	0	111	0.00	0	116	0.16
9600	0	51	0.16	0	55	0.00	0	58	–0.69
14400	0	34	–0.79	0	36	0.90	0	38	0.16
19200	0	25	0.16	0	27	0.00	0	28	1.02
28800	0	16	2.12	0	18	–1.75	0	19	–2.34
31250	0	15	0.00	0	16	1.20	0	17	0.00
38400	0	12	0.16	0	13	0.00	0	14	–2.34

Table 12.3 Bit Rates and BRR1 Settings (Asynchronous Mode) (cont)

Bit Rate (Bits/s)	ϕ (MHz)								
	18.432			19.6608			20		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	81	–0.22	3	86	0.31	3	88	–0.25
150	2	239	0.00	2	255	0.00	3	64	0.16
300	2	119	0.00	2	127	0.00	2	129	0.16
600	1	239	0.00	1	255	0.00	2	64	0.16
1200	1	119	0.00	1	127	0.00	1	129	0.16
2400	0	239	0.00	0	255	0.00	1	64	0.16
4800	0	119	0.00	0	127	0.00	0	129	0.16
9600	0	59	0.00	0	63	0.00	0	64	0.16
14400	0	39	0.00	0	42	–0.78	0	42	0.94
19200	0	29	0.00	0	31	0.00	0	32	–1.36
28800	0	19	0.00	0	20	1.59	0	21	–1.36
31250	0	17	2.40	0	19	–1.70	0	19	0.00
38400	0	14	0.00	0	15	0.00	0	15	1.73

Table 12.4 Bit Rates and BRR Settings in Clocked Synchronous Mode

Bit Rate (Bits/s)	ϕ (MHz)							
	4		8		10		12	
	n	N	n	N	n	N	n	N
110	3	141	3	212	3	212	3	212
250	2	249	3	124	3	155	3	187
500	2	124	2	249	3	77	3	93
1k	1	249	2	124	2	155	2	187
2.5k	1	99	1	199	1	249	2	74
5k	0	199	1	99	1	124	1	149
10k	0	99	0	199	0	249	1	74
25k	0	39	0	79	0	99	0	119
50k	0	19	0	39	0	49	0	59
100k	0	9	0	19	0	24	0	29
250k	0	3	0	7	0	9	0	11
500k	0	1	0	3	0	4	0	5
1M			0	1	—	—	0	2
2.5M					0	0*	0	0*
5M								

Table 12.4 Bit Rates and BRR Settings in Clocked Synchronous Mode (cont)

Bit Rate (Bits/s)	ϕ (MHz)			
	16		20	
	n	N	n	N
110	3	212	3	212
250	3	249	3	249
500	3	124	3	155
1k	2	249	3	77
2.5k	2	99	2	124
5k	1	199	1	249
10k	1	99	1	124
25k	0	159	0	199
50k	0	79	0	99
100k	0	39	0	49
250k	0	15	0	19
500k	0	7	0	9
1M	0	3	0	4
2.5M	—	—	0	1
5M			0	0*

Note: Settings with an error of 1% or less are recommended.

Legend

Blank: No setting available

—: Setting possible, but error occurs

*: Continuous transmission/reception is not possible.

The BRR1 setting is calculated as follows:

Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Clock synchronous mode:

$$N = \frac{\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

B: Bit rate (bit/s)
N: Baud rate generator BRR1 setting ($0 \leq N \leq 255$)
ϕ: Operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the following table for the clock sources and value of n.)

		SMR1 Settings	
n	**Clock Source**	**CKS1**	**CKS2**
0	ϕ	0	0
1	$\phi/4$	0	1
2	$\phi/16$	1	0
3	$\phi/64$	1	1

The bit rate error in asynchronous mode is calculated as follows:

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 12.5 indicates the maximum bit rates in the asynchronous mode when the baud rate generator is being used for various frequencies. Tables 12.6 and 12.7 show the maximum rates for external clock input..

Table 12.5 Maximum Bit Rates for Various Frequencies with Baud Rate Generator (Asynchronous Mode)

		Settings	
ϕ (MHz)	**Maximum Bit Rate (Bits/s)**	**n**	**N**
4	125000	0	0
4.9152	153600	0	0
6	187500	0	0
7.3728	230400	0	0
8	250000	0	0
9.8304	307200	0	0
10	312500	0	0
11.0592	345600	0	0
12	375000	0	0
12.288	384000	0	0
14	437500	0	0
14.7456	460800	0	0
16	500000	0	0
17.2032	537600	0	0
18	562500	0	0
18.432	576000	0	0
19.6608	614400	0	0
20	625000	0	0

Table 12.6 Maximum Bit Rates during External Clock Input (Asynchronous Mode)

ϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (Bits/s)
4	1.0000	62500
4.9152	1.2288	76800
6	1.5000	93750
7.3728	1.8432	115200
8	2.0000	125000
9.8304	2.4576	153600
10	2.5000	156250
11.0592	2.7648	172800
12	3.0000	187500
12.288	3.0720	192000
14	3.5000	218750
14.7456	3.6864	230400
16	4.0000	250000
17.2032	4.3008	268800
18	4.5000	281250
18.432	4.6080	288000
19.6608	4.9152	307200
20	5.0000	312500

Table 12.7 Maximum Bit Rates during External Clock Input (Clock Synchronous Mode)

ϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (Bits/s)
4	0.6667	666666.7
6	1.0000	1000000.0
8	1.3333	1333333.3
10	1.6667	1666666.7
12	2.0000	2000000.0
14	2.3333	2333333.3
16	2.6667	2666666.7
18	3.0000	3000000.0
20	3.3333	3333333.3

12.3 Operation

12.3.1 Overview

For serial communication, the SCI1 has an asynchronous mode in which characters are synchronized individually, and a clock synchronous mode in which communication is synchronized with clock pulses. Asynchronous/clock synchronous mode and the transmission format are selected in the serial mode register (SMR1), as shown in table 12.8. The SCI1 clock source is selected by the C/$\overline{A}$ bit in the serial mode register (SMR1) and the CKE1 and CKE0 bits in the serial control register (SCR), as shown in table 12.9.

Asynchronous Mode:

- Data length is selectable: seven or eight bits.
- Parity and multiprocessor bits are selectable, as well as the stop bit length (one or two bits). These selections determine the transmit/receive format and character length.
- In receiving, it is possible to detect framing errors (FER), parity errors (PER), overrun errors (ORER), and the break state.
- An internal or external clock can be selected as the SCI1 clock source.
  - — When an internal clock is selected, the SCI1 operates using the on-chip baud rate generator clock, and can output a clock with a frequency matching the bit rate.
  - — When an external clock is selected, the external clock input must have a frequency 16 times the bit rate. (The on-chip baud rate generator is not used.)

Clock Synchronous Mode:

- The communication format has a fixed 8-bit data length.
- In receiving, it is possible to detect overrun errors (ORER).
- An internal or external clock can be selected as the SCI1 clock source.
  - — When an internal clock is selected, the SCI1 operates using the on-chip baud rate generator clock, and outputs a synchronous clock signal to external devices.
  - — When an external clock is selected, the SCI1 operates on the input synchronous clock. The on-chip baud rate generator is not used.

Table 12.8 SMR1 Settings and SCI1 Communication Formats

	SMR Settings					SCI1 Communication Format			
Mode	Bit 7 C/$\overline{A}$	Bit 6 CHR	Bit 5 PE	Bit 2 MP	Bit 3 STOP	Data Length	Parity Bit	Multipro-cessor Bit	Stop Bit Length
Asynchronous	0	0	0	0	0	8-bit	Not set	Not set	1 bit
					1				2 bits
			1		0		Set		1 bit
					1				2 bits
		1	0		0	7-bit	Not set		1 bit
					1				2 bits
			1		0		Set		1 bit
					1				2 bits
Asynchronous (multiprocessor format)		0	*	1	0	8-bit	Not set	Set	1 bit
			*		1				2 bits
		1	*		0	7-bit			1 bit
			*		1				2 bits
Clock synchronous	1	*	*	*	*	8-bit		Not set	None

Note: Asterisks (*) in the table indicate don't-care bits.

Table 12.9 SMR1 and SCR1 Settings and SCI1 Clock Source Selection

	SMR1	SCR1 Settings		SCI1 Transmit/Receive Clock	
Mode	Bit 7 C/$\overline{A}$	Bit 1 CKE1	Bit 0 CKE0	Clock Source	SCK Pin Function*
Asynchronous	0	0	0	Internal	SCI1 does not use the SCK pin
			1		Outputs a clock with frequency matching the bit rate
		1	0	External	Inputs a clock with frequency 16 times the bit rate
			1		
Clock synch-ronous	1	0	0	Internal	Outputs the synchronous clock
			1		
		1	0	External	Inputs the synchronous clock
			1		

Note: * Select the function in combination with the pin function controller (PFC).

12.3.2 Operation in Asynchronous Mode

In the asynchronous mode, each transmitted or received character begins with a start bit and ends with a stop bit. Serial communication is synchronized one character at a time.

The transmitting and receiving sections of the SCI1 are independent, so full duplex communication is possible. The transmitter and receiver are both double buffered, so data can be written and read while transmitting and receiving are in progress, enabling continuous transmitting and receiving.

Figure 12.2 shows the general format of asynchronous serial communication. In asynchronous serial communication, the communication line is normally held in the marking (high) state. The SCI1 monitors the line and starts serial communication when the line goes to the space (low) state, indicating a start bit. One serial character consists of a start bit (low), data (LSB first), parity bit (high or low), and stop bit (high), in that order.

When receiving in the asynchronous mode, the SCI1 synchronizes on the falling edge of the start bit. The SCI1 samples each data bit on the eighth pulse of a clock with a frequency 16 times the bit rate. Receive data is latched at the center of each bit.

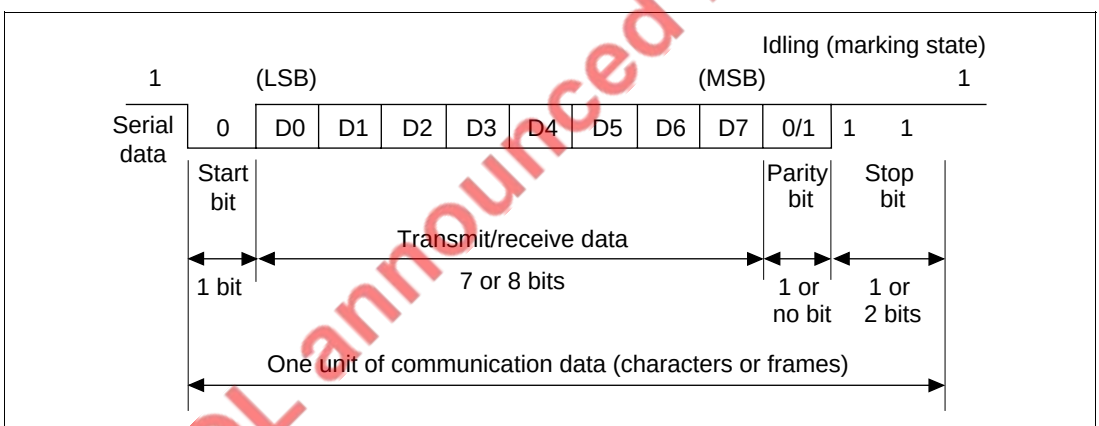


Figure 12.2 Data Format in Asynchronous Communication (Example: 8-bit Data with Parity and Two Stop Bits)

Transmit/Receive Formats: Table 12.10 shows the 12 communication formats that can be selected in the asynchronous mode. The format is selected by settings in the serial mode register (SMR1).

Table 12.10 Serial Communication Formats (Asynchronous Mode)

SMR1 Bits				Serial Transmit/Receive Format and Frame Length											
Bit 6: CHR	Bit 5: PE	Bit 2: MP	Bit 3: 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	1	0	0	START	8-bit data								P	STOP	
0	1	0	1	START	8-bit data								P	STOP	STOP
1	0	0	0	START	7-bit data							STOP			
1	0	0	1	START	7-bit data							STOP	STOP		
1	1	0	0	START	7-bit data							P	STOP		
1	1	0	1	START	7-bit data							P	STOP	STOP	
0	—	1	0	START	8-bit data								MPB	STOP	
0	—	1	1	START	8-bit data								MPB	STOP	STOP
1	—	1	0	START	7-bit data							MPB	STOP		
1	—	1	1	START	7-bit data							MPB	STOP	STOP	

—: Don't care bits.

Note: START: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI1 transmit/receive clock. The clock source is selected by the C/$\overline{A}$ bit in the serial mode register (SMR1) and bits CKE1 and CKE0 in the serial control register (SCR1) (table 12.9).

When an external clock is input at the SCK pin, it must have a frequency equal to 16 times the desired bit rate.

When the SCI1 operates on an internal clock, it can output a clock signal at the SCK pin. The frequency of this output clock is equal to the bit rate. The phase is aligned as in figure 12.3 so that the rising edge of the clock occurs at the center of each transmit data bit.

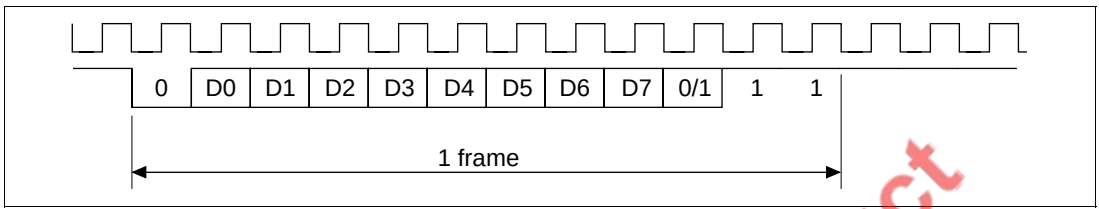


Figure 12.3 Output Clock and Communication Data Phase Relationship (Asynchronous Mode)

SCI1 Initialization (Asynchronous Mode): Before transmitting or receiving, clear the TE and RE bits to 0 in the serial control register (SCR1), then initialize the SCI1 as follows.

When changing the operation mode or communication format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR1). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR1), which retain their previous contents.

Figure 12.4 is a sample flowchart for initializing the SCI1. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. Select the clock source in the serial control register (SCR1). Leave RIE, TIE, TEIE, MPIE, TE and RE cleared to 0.
2. Select the communication format in the serial mode register (SMR1).
3. Write the value corresponding to the bit rate in the bit rate register (BRR1).
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR1) to 1. Also set RIE, TIE, TEIE and MPIE as necessary. Setting TE or RE enables the SCI1 to use the TxD or RxD pin. The initial states are the marking transmit state, and the idle receive state (waiting for a start bit).

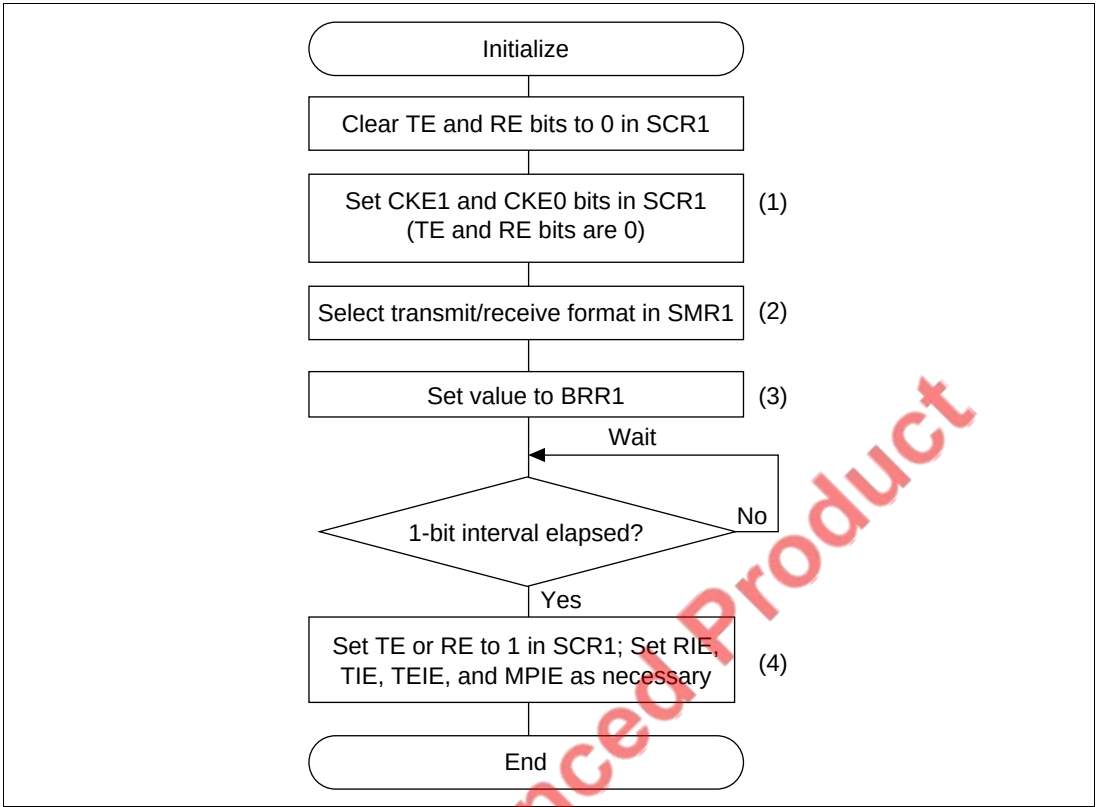


Figure 12.4 Sample Flowchart for SCI1 Initialization

Transmitting Serial Data (Asynchronous Mode): Figure 12.5 shows a sample flowchart for transmitting serial data. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. SCI1 initialization: Set the TxD pin using the PFC.
2. SCI1 status check and transmit data write: Read the serial status register (SSR1), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR1) and clear TDRE to 0.
3. Continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in TDR1, then clear TDRE to 0.
4. To output a break at the end of serial transmission, first clear the port data register (DR) to 0, then clear the TE to 0 in SCR1 and use the PFC to establish the TxD pin as an output port.

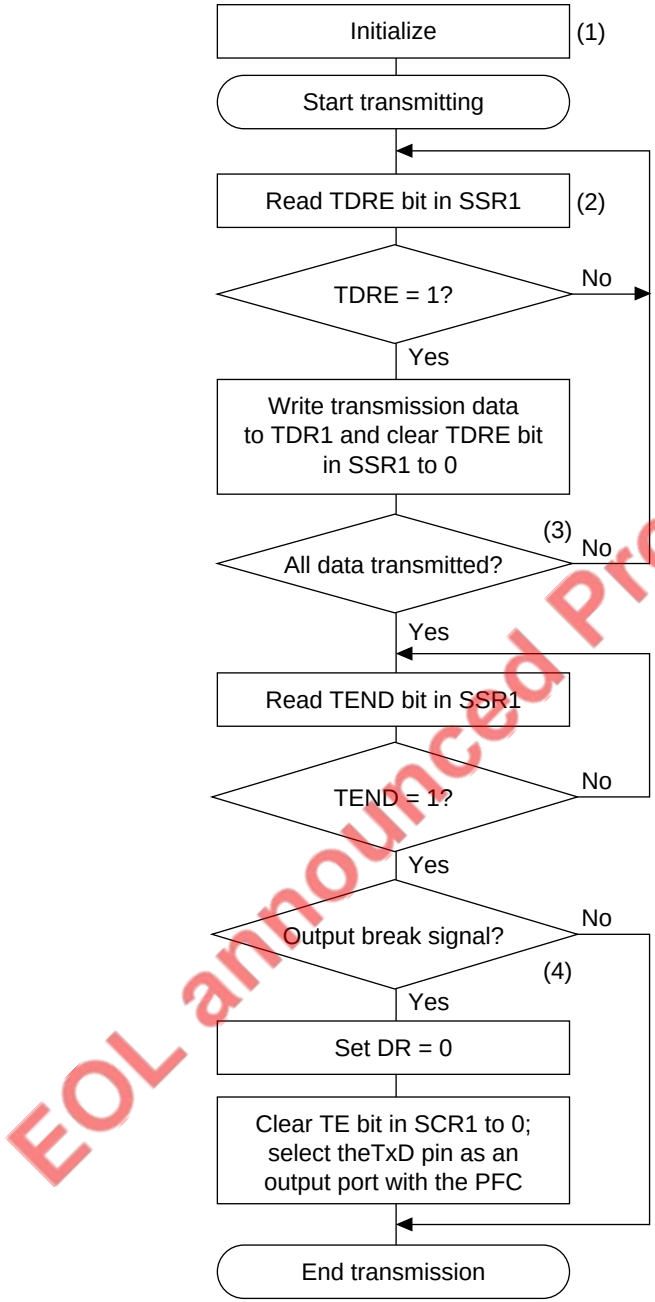


Figure 12.5 Sample Flowchart for Transmitting Serial Data

In transmitting serial data, the SCI1 operates as follows:

1. The SCI1 monitors the TDRE bit in the SSR1. When TDRE is cleared to 0, the SCI1 recognizes that the transmit data register (TDR1) contains new data, and loads this data from the TDR1 into the transmit shift register (TSR1).
2. After loading the data from the TDR1 into the TSR1, the SCI1 sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) is set to 1 in the SCR1, the SCI1 requests a transmit-data-empty interrupt (TxI) at this time.

   Serial transmit data is transmitted in the following order from the TxD pin:

   a. Start bit: one 0 bit is output.
   b. Transmit data: seven or eight bits of data are output, LSB first.
   c. Parity bit or multiprocessor bit: one parity bit (even or odd parity) or one multiprocessor bit is output. Formats in which neither a parity bit nor a multiprocessor bit is output can also be selected.
   d. Stop bit: one or two 1 bits (stop bits) are output.
   e. Marking: output of 1 bits continues until the start bit of the next transmit data.

3. The SCI1 checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI1 loads new data from the TDR1 into the TSR1, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI1 sets the TEND bit to 1 in the SSR1, outputs the stop bit, then continues output of 1 bits (marking). If the transmit-end interrupt enable bit (TEIE) in the SCR1 is set to 1, a transmit-end interrupt (TEI) is requested.

Figure 12.6 shows an example of SCI transmit operation in asynchronous mode.

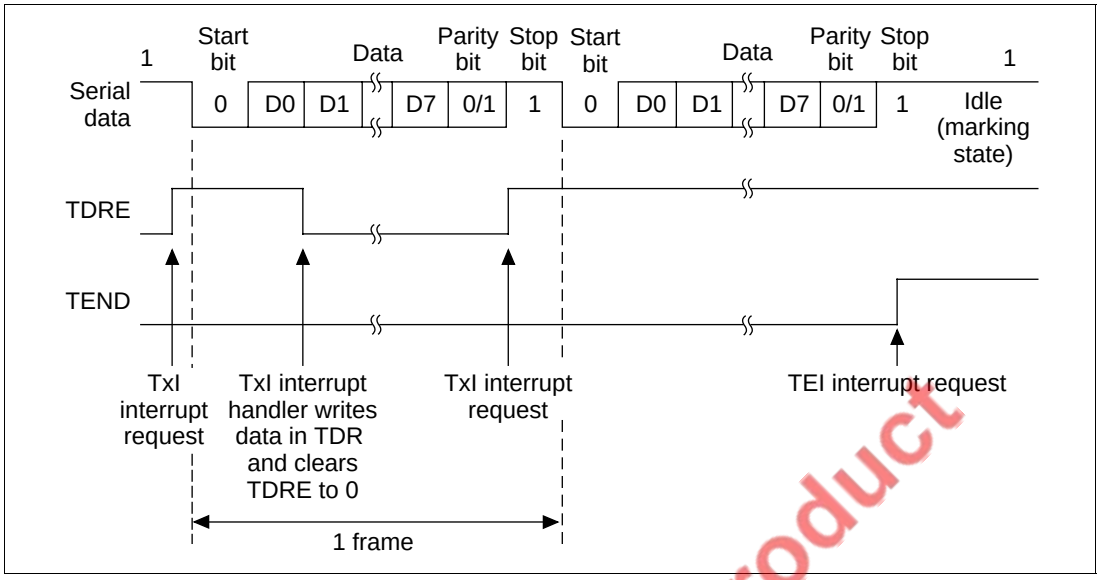


Figure 12.6 SCI1 Transmit Operation in Asynchronous Mode (8-Bit Data with Parity and One Stop Bit)

Receiving Serial Data: Figures 12.7 show a sample flowchart for receiving serial data. The procedure is as follows (the steps correspond to the numbers in the flowchart).

1. SCI1 initialization: Set the RxD pin using the PFC.
2. Receive error handling and break detection: If a receive error occurs, read the ORER, PER, and FER bits of the SSR1 to identify the error. After executing the necessary error handling, clear ORER, PER, and FER all to 0. Receiving cannot resume if ORER, PER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
3. SCI1 status check and receive-data read: Read the serial status register (SSR1), check that RDRF is set to 1, then read receive data from the receive data register (RDR1) and clear RDRF to 0. The RxI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. Continue receiving serial data: Read the RDR1 and RDRF bit and clear RDRF to 0 before the stop bit of the current frame is received.

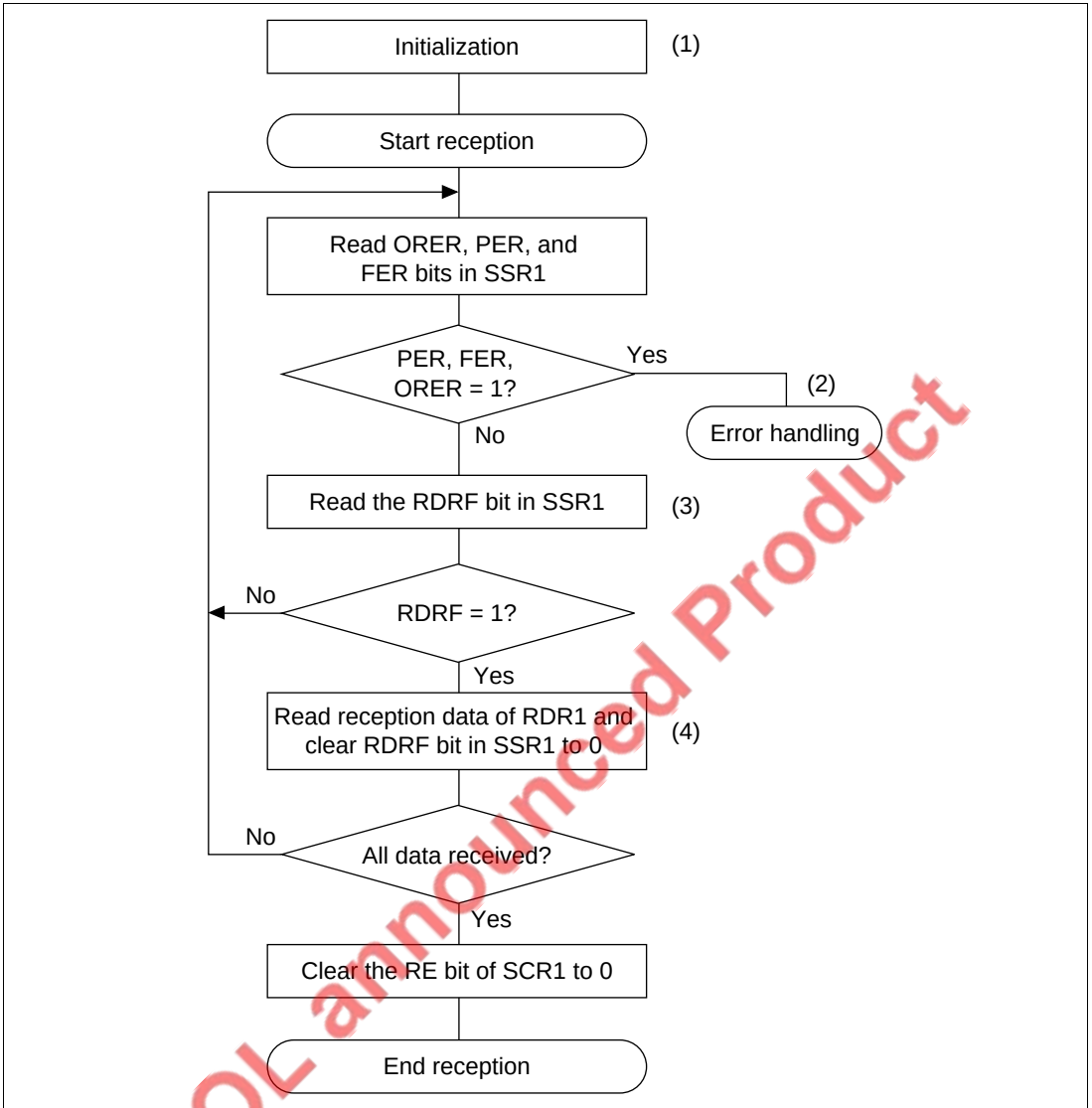


Figure 12.7 Sample Flowchart for Receiving Serial Data

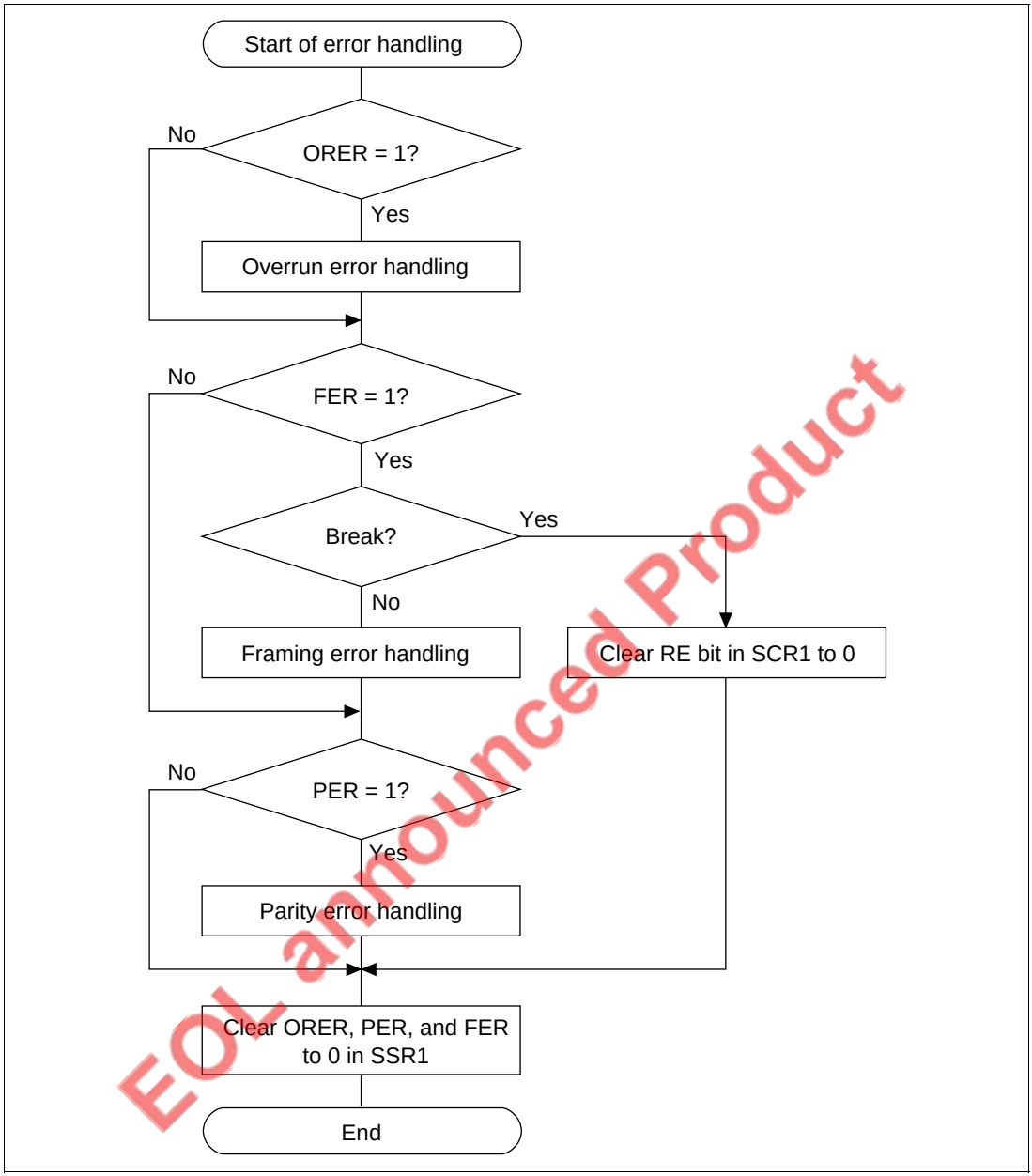


Figure 12.7 Sample Flowchart for Receiving Serial Data (cont)

In receiving, the SCI1 operates as follows:

1. The SCI1 monitors the communication line. When it detects a start bit (0), the SCI1 synchronizes internally and starts receiving.
2. Receive data is shifted into the RSR1 in order from the LSB to the MSB.
3. The parity bit and stop bit are received. After receiving these bits, the SCI1 makes the following checks:
   a. Parity check. The number of 1s in the receive data must match the even or odd parity setting of the O/$\overline{E}$ bit in the SMR1.
   b. Stop bit check. The stop bit value must be 1. If there are two stop bits, only the first stop bit is checked.
   c. Status check. RDRF must be 0 so that receive data can be loaded from the RSR1 into the RDR1.

   If the data passes these checks, the SCI1 sets RDRF to 1 and stores the received data in the RDR1. If one of the checks fails (receive error), the SCI1 operates as indicated in table 12.11.

   Note: When a receive error occurs, further receiving is disabled. While receiving, the RDRF bit is not set to 1, so be sure to clear the error flags.
4. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCR1, the SCI1 requests a receive-data-full interrupt (RxI). If one of the error flags (ORER, PER, or FER) is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCR is also set to 1, the SCI1 requests a receive-error interrupt (ERI).

Figure 12.8 shows an example of SCI1 receive operation in the asynchronous mode.

Table 12.11 Receive Error Conditions and SCI1 Operation

Receive Error	Abbreviation	Condition	Data Transfer
Overrun error	ORER	Receiving of next data ends while RDRF is still set to 1 in SSR1	Receive data not loaded from RSR1 into RDR1
Framing error	FER	Stop bit is 0	Receive data loaded from RSR1 into RDR1
Parity error	PER	Parity of receive data differs from even/odd parity setting in SMR1	Receive data loaded from RSR1 into RDR1

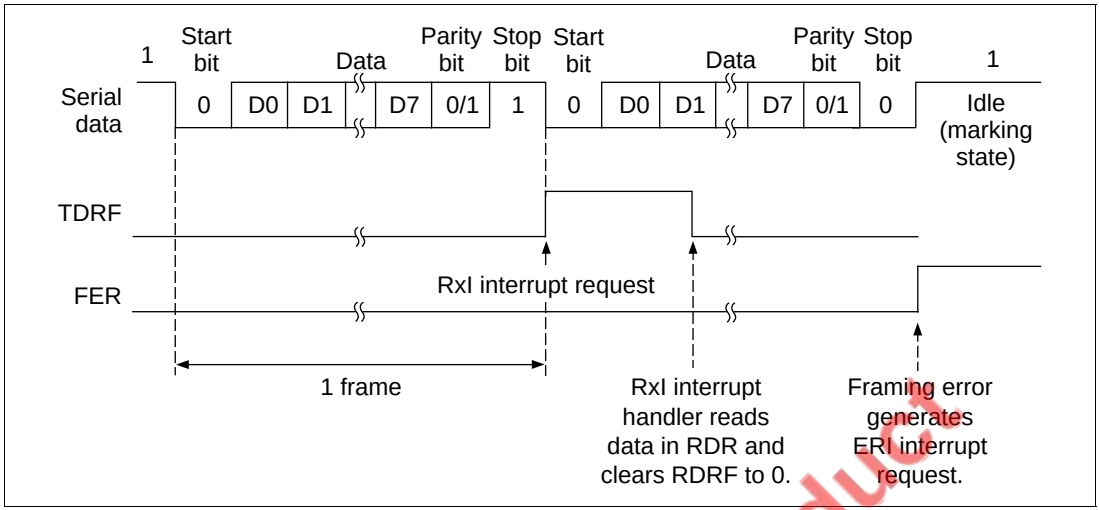


Figure 12.8 SCI1 Receive Operation (8-Bit Data with Parity and One Stop Bit)

12.3.3 Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line for sending and receiving data. The processors communicate in the asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by a unique ID. A serial communication cycle consists of an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit distinguishes ID-sending cycles from data-sending cycles. The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0.

Receiving processors skip incoming data until they receive data with the multiprocessor bit set to 1. When they receive data with the multiprocessor bit set to 1, receiving processors compare the data with their IDs. The receiving processor with a matching ID continues to receive further incoming data. Processors with IDs not matching the received data skip further incoming data until they again receive data with the multiprocessor bit set to 1. Multiple processors can send and receive data in this way.

Figure 12.9 shows the example of communication among processors using the multiprocessor format.

Communication Formats: Four formats are available. Parity-bit settings are ignored when the multiprocessor format is selected. For details see table 12.8.

Clock: See the description in the asynchronous mode section.

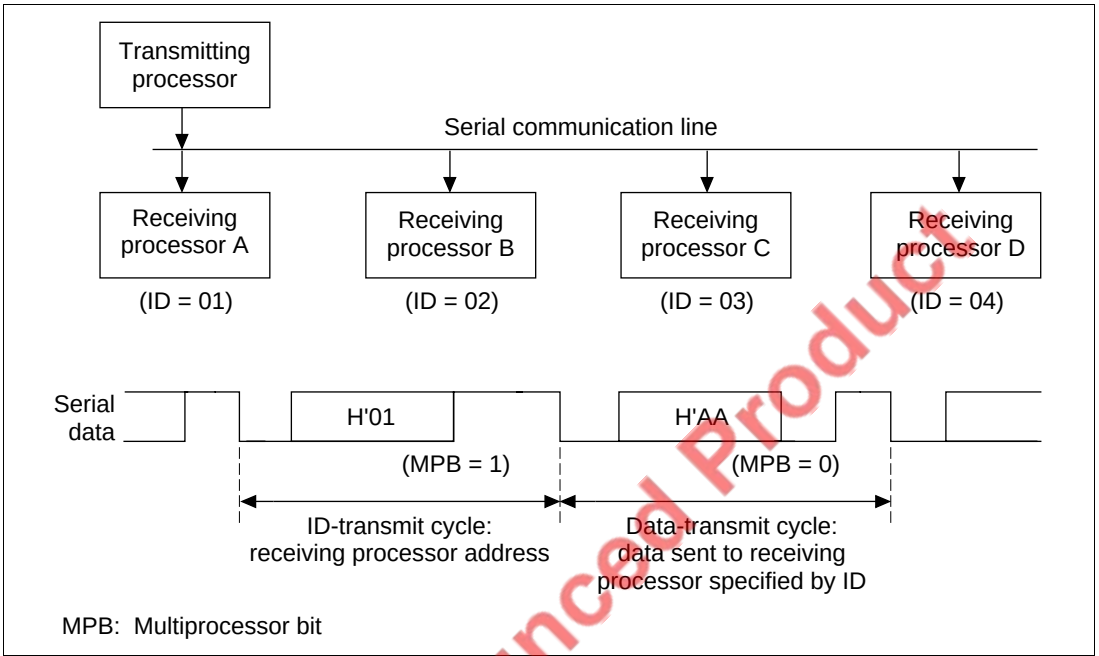


Figure 12.9 Communication Among Processors Using Multiprocessor Format (Sending Data H'AA to Receiving Processor A)

Transmitting Multiprocessor Serial Data: Figure 12.10 shows a sample flowchart for transmitting multiprocessor serial data. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. SCI1 initialization: Set the TxD pin using the PFC.
2. SCI1 status check and transmit data write: Read the serial status register (SSR1), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR1). Also set MPBT (multiprocessor bit transfer) to 0 or 1 in SSR1. Finally, clear TDRE to 0.
3. Continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in TDR1, then clear TDRE to 0.
4. Output a break at the end of serial transmission: Set the data register (DR) of the port to 0, then clear TE to 0 in SCR1 and set the TxD pin function as output port with the PFC.

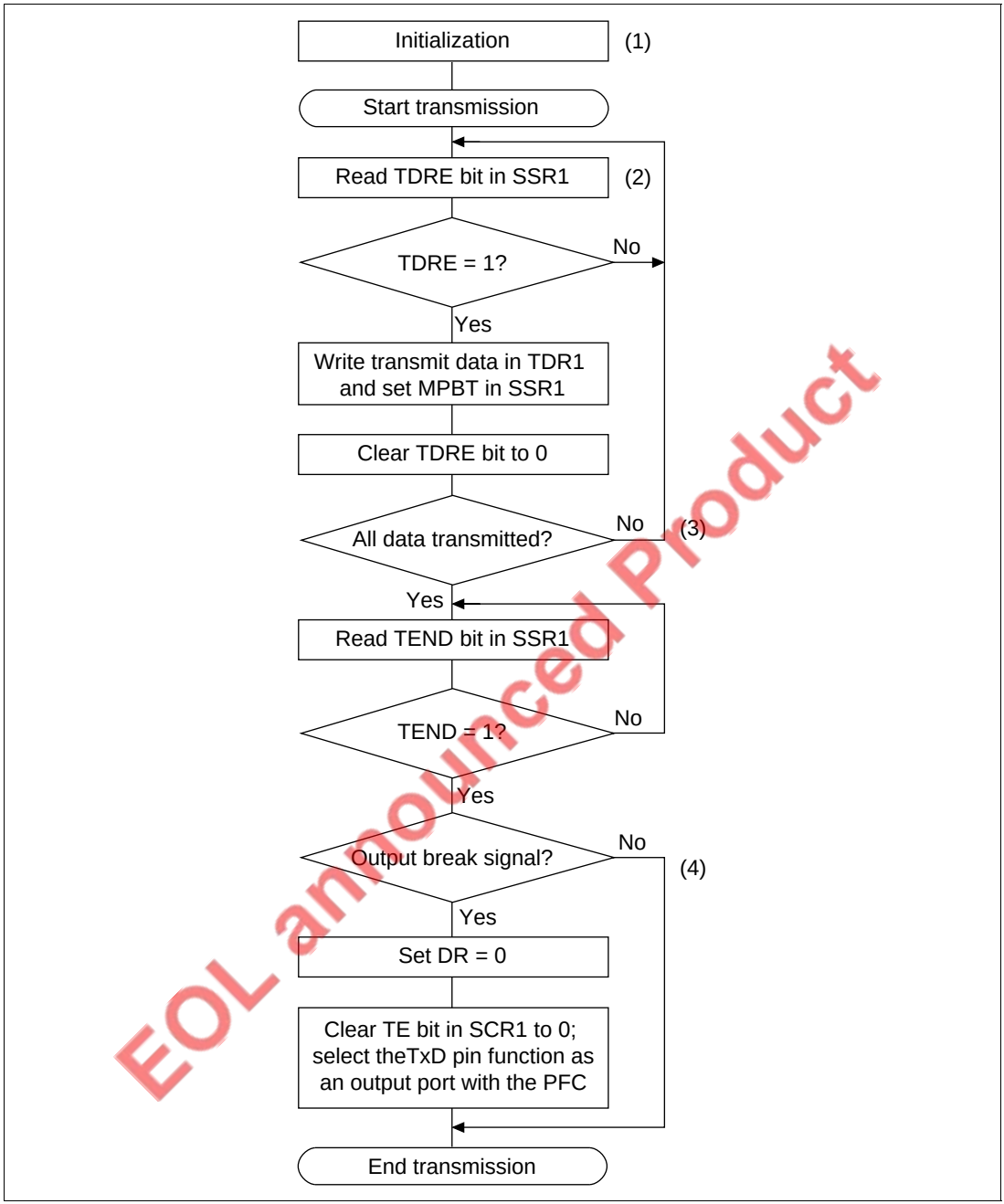


Figure 12.10 Sample Flowchart for Transmitting Multiprocessor Serial Data

In transmitting serial data, the SCI1 operates as follows:

1. The SCI1 monitors the TDRE bit in the SSR1. When TDRE is cleared to 0 the SCI1 recognizes that the transmit data register (TDR1) contains new data, and loads this data from the TDR1 into the transmit shift register (TSR1).
2. After loading the data from the TDR1 into the TSR1, the SCI1 sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCR1 is set to 1, the SCI1 requests a transmit-data-empty interrupt (TxI) at this time.

   Serial transmit data is transmitted in the following order from the TxD pin:

   a. Start bit: one 0 bit is output.
   b. Transmit data: seven or eight bits are output, LSB first.
   c. Multiprocessor bit: one multiprocessor bit (MPBT value) is output.
   d. Stop bit: one or two 1 bits (stop bits) are output.
   e. Marking: output of 1 bits continues until the start bit of the next transmit data.

3. The SCI1 checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI1 loads data from the TDR1 into the TSR1, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI1 sets the TEND bit in the SSR1 to 1, outputs the stop bit, then continues output of 1 bits in the marking state. If the transmit-end interrupt enable bit (TEIE) in the SCR1 is set to 1, a transmit-end interrupt (TEI) is requested at this time.

Figure 12.11 shows an example of SCI1 receive operation in the multiprocessor format.

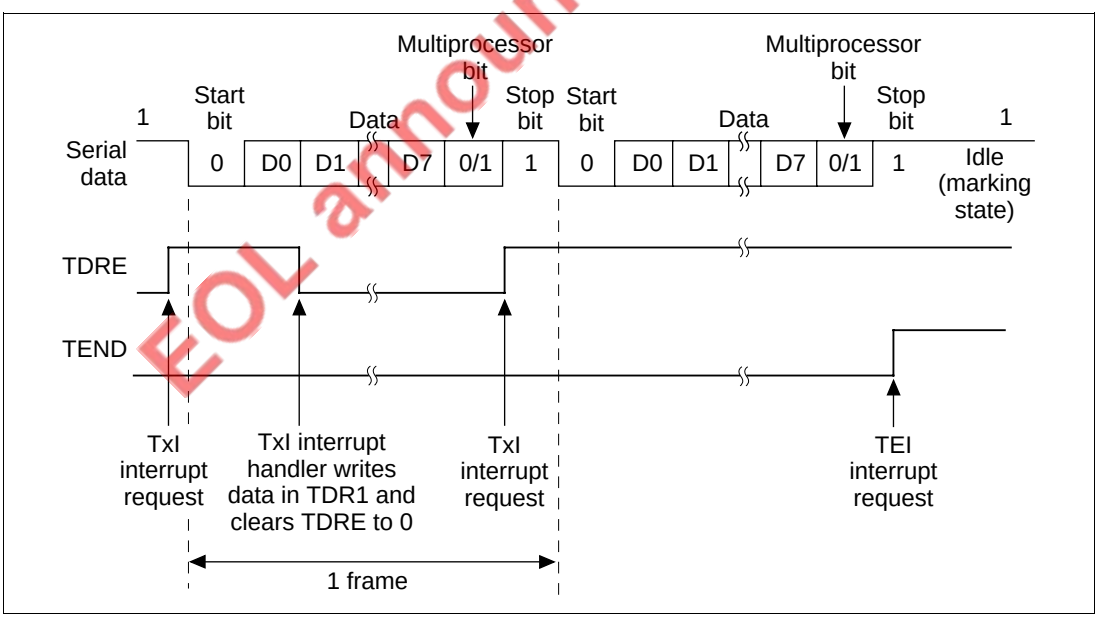


Figure 12.11 SCI1 Multiprocessor Transmit Operation (8-Bit Data with Multiprocessor Bit and One Stop Bit)

Receiving Multiprocessor Serial Data: Figure 12.12 shows a sample flowchart for receiving multiprocessor serial data. The procedure for receiving multiprocessor serial data is listed below.

1. SCI1 initialization: Set the RxD pin using the PFC.
2. ID receive cycle: Set the MPIE bit in the serial control register (SCR1) to 1.
3. SCI1 status check and compare to ID reception: Read the serial status register (SSR1), check that RDRF is set to 1, then read data from the receive data register (RDR1) and compare with the processor's own ID. If the ID does not match the receive data, set MPIE to 1 again and clear RDRF to 0. If the ID matches the receive data, clear RDRF to 0.
4. Receive error handling and break detection: If a receive error occurs, read the ORER and FER bits in SSR1 to identify the error. After executing the necessary error processing, clear both ORER and FER to 0. Receiving cannot resume if ORER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
5. SCI1 status check and data receiving: Read SSR1, check that RDRF is set to 1, then read data from the receive data register (RDR1).

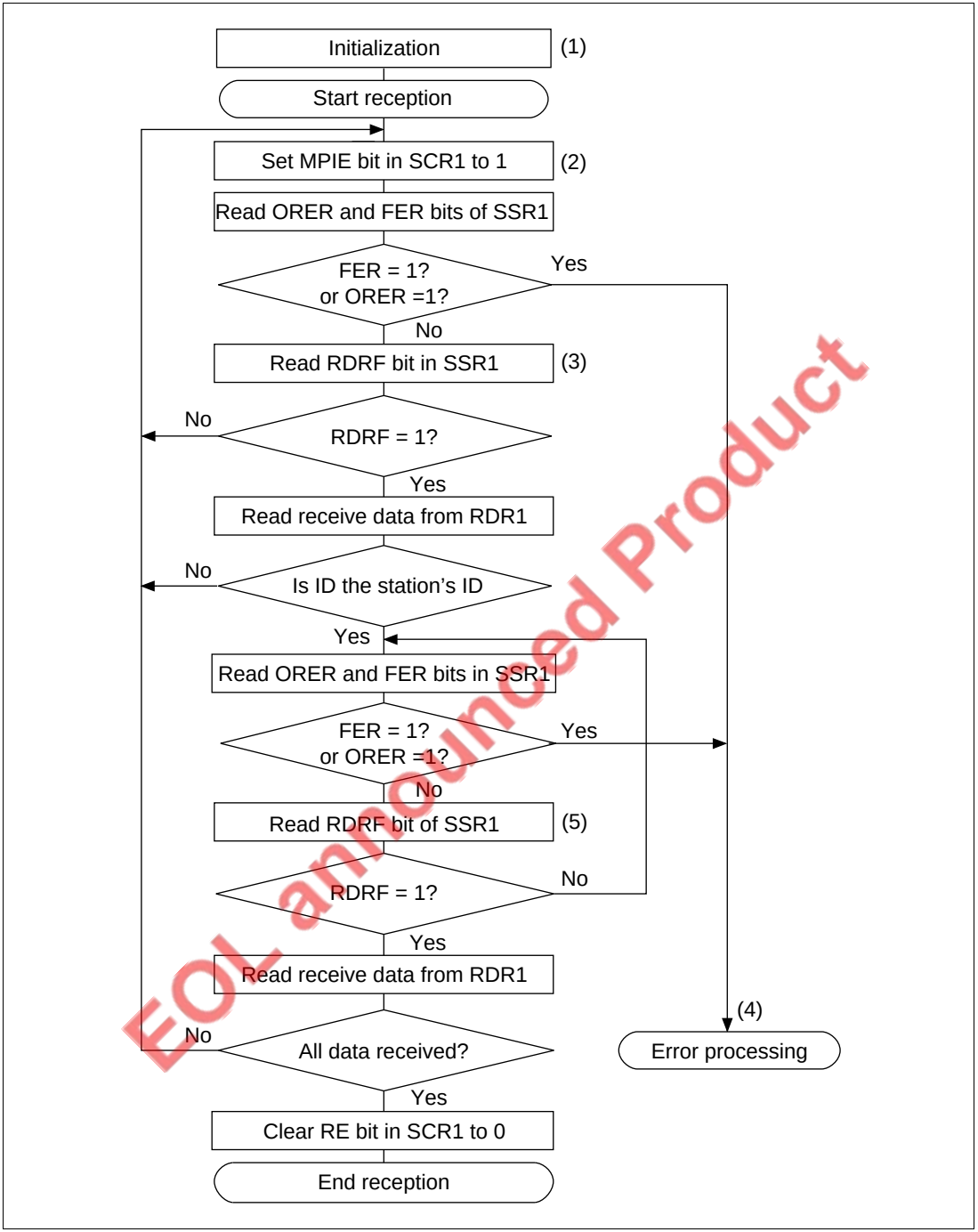


Figure 12.12 Sample Flowchart for Receiving Multiprocessor Serial Data

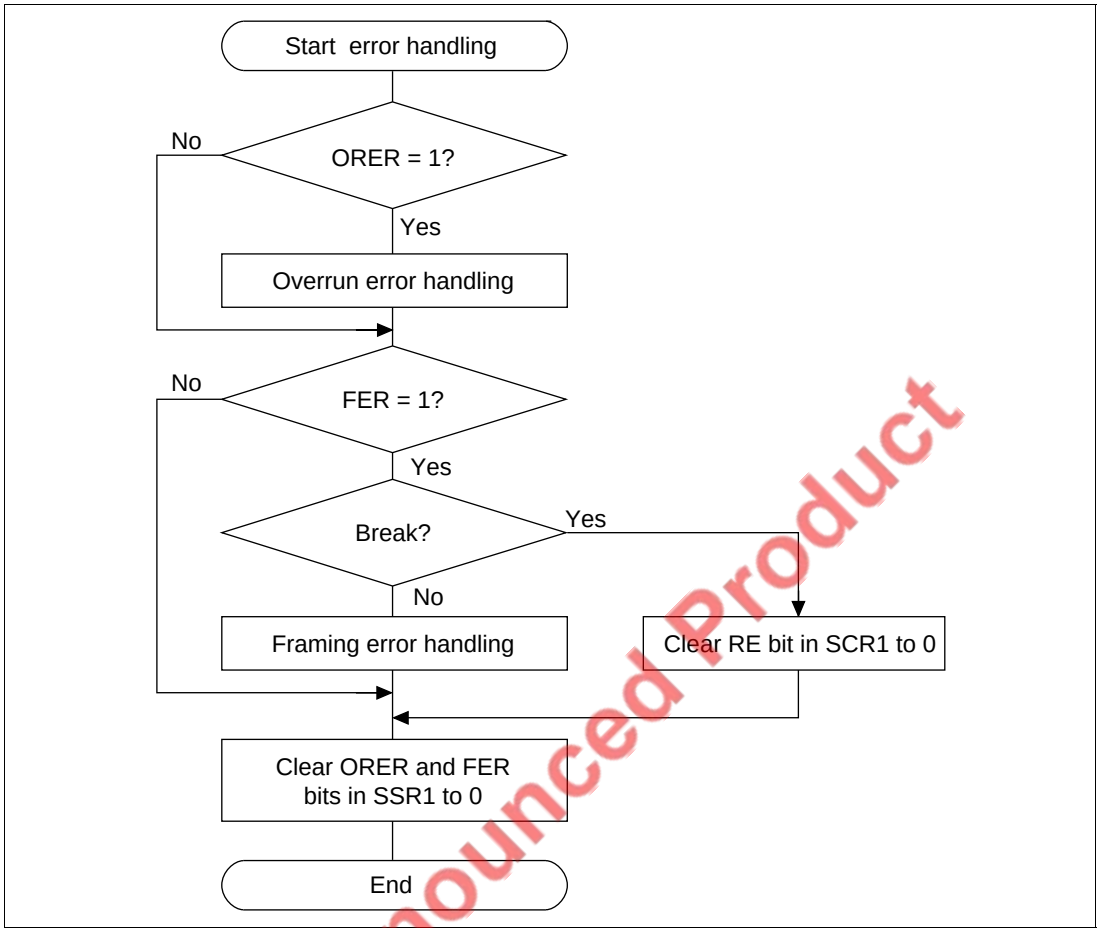
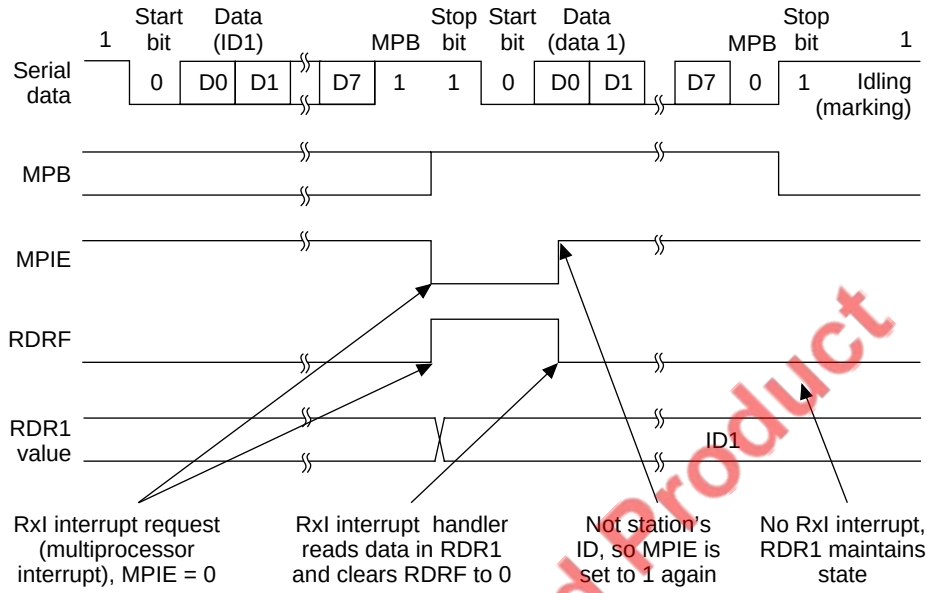
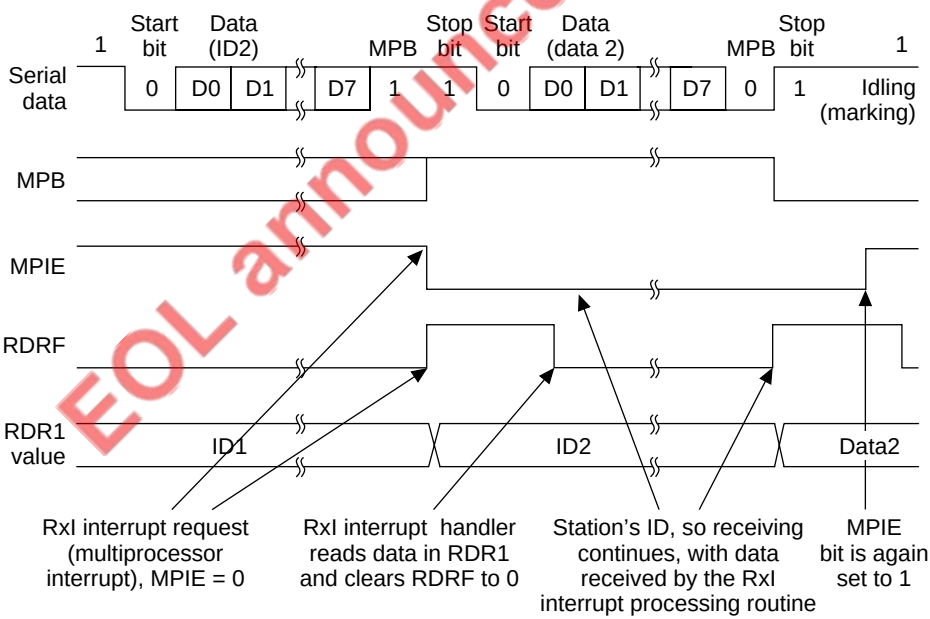


Figure 12.12 Sample Flowchart for Receiving Multiprocessor Serial Data (cont)

Figures 12.13 show examples of SCI1 receive operation using a multiprocessor format.



(a) ID Does Not Match



(b) ID Matches

Figure 12.13 SCI1 Receive Operation (8-Bit Data with Multiprocessor Bit and One Stop Bit)

12.3.4 Clock Synchronous Operation

In the clock synchronous mode, the SCI1 transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

The SCI1 transmitter and receiver are independent, so full duplex communication is possible while sharing the same clock. The transmitter and receiver are also double buffered, so continuous transmitting or receiving is possible by reading or writing data while transmitting or receiving is in progress.

Figure 12.14 shows the general format in clock synchronous serial communication.

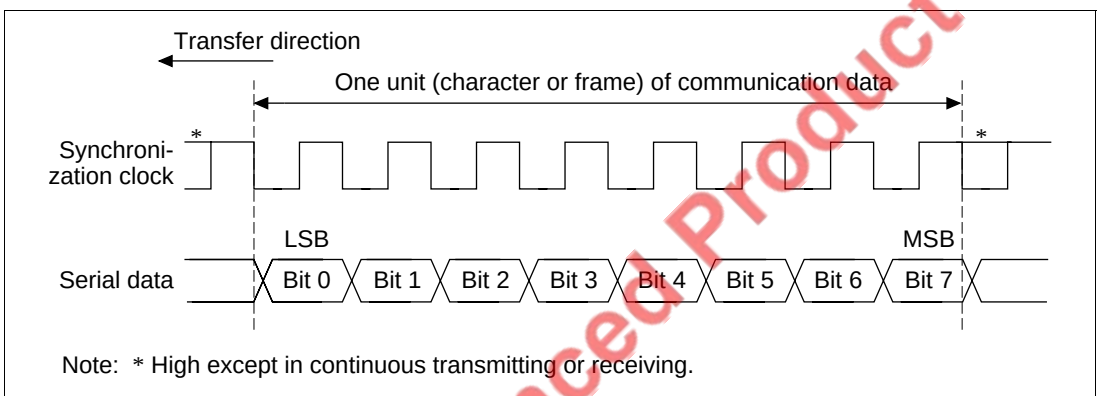


Figure 12.14 Data Format in Clock Synchronous Communication

In clock synchronous serial communication, each data bit is output on the communication line from one falling edge of the serial clock to the next. Data are guaranteed valid at the rising edge of the serial clock. In each character, the serial data bits are transmitted in order from the LSB (first) to the MSB (last). After output of the MSB, the communication line remains in the state of the MSB. In the clock synchronous mode, the SCI1 transmits or receives data by synchronizing with the falling edge of the synchronization clock.

Communication Format: The data length is fixed at eight bits. No parity bit or multiprocessor bit can be added.

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI1 transmit/receive clock. The clock source is selected by the C/$\overline{A}$ bit in the serial mode register (SMR1) and bits CKE1 and CKE0 in the serial control register (SCR1). See table 12.9.

When the SCI1 operates on an internal clock, it outputs the clock signal at the SCK pin. Eight clock pulses are output per transmitted or received character. When the SCI1 is not transmitting or receiving, the clock signal remains in the high state.

Note: An overrun error occurs only during the receive operation, and the sync clock is output until the RE bit is cleared to 0. When you want to perform a receive operation in one-character units, select external clock for the clock source.

SCI1 Initialization (Clock Synchronous Mode): Before transmitting or receiving, software must clear the TE and RE bits to 0 in the serial control register (SCR1), then initialize the SCI1 as follows.

When changing the mode or communication format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR1). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR1), which retain their previous contents.

Figure 12.15 is a sample flowchart for initializing the SCI1.

1. Select the clock source in the serial control register (SCR1). Leave RIE, TIE, TEIE, MPIE, TE, and RE cleared to 0.
2. Select the communication format in the serial mode register (SMR1).
3. Write the value corresponding to the bit rate in the bit rate register (BRR1) unless an external clock is used.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR1) to 1. Also set RIE, TIE, TEIE, and MPIE. The TxD, RxD pins becomes usable in response to the PFC corresponding bits and the TE, RE bit settings.

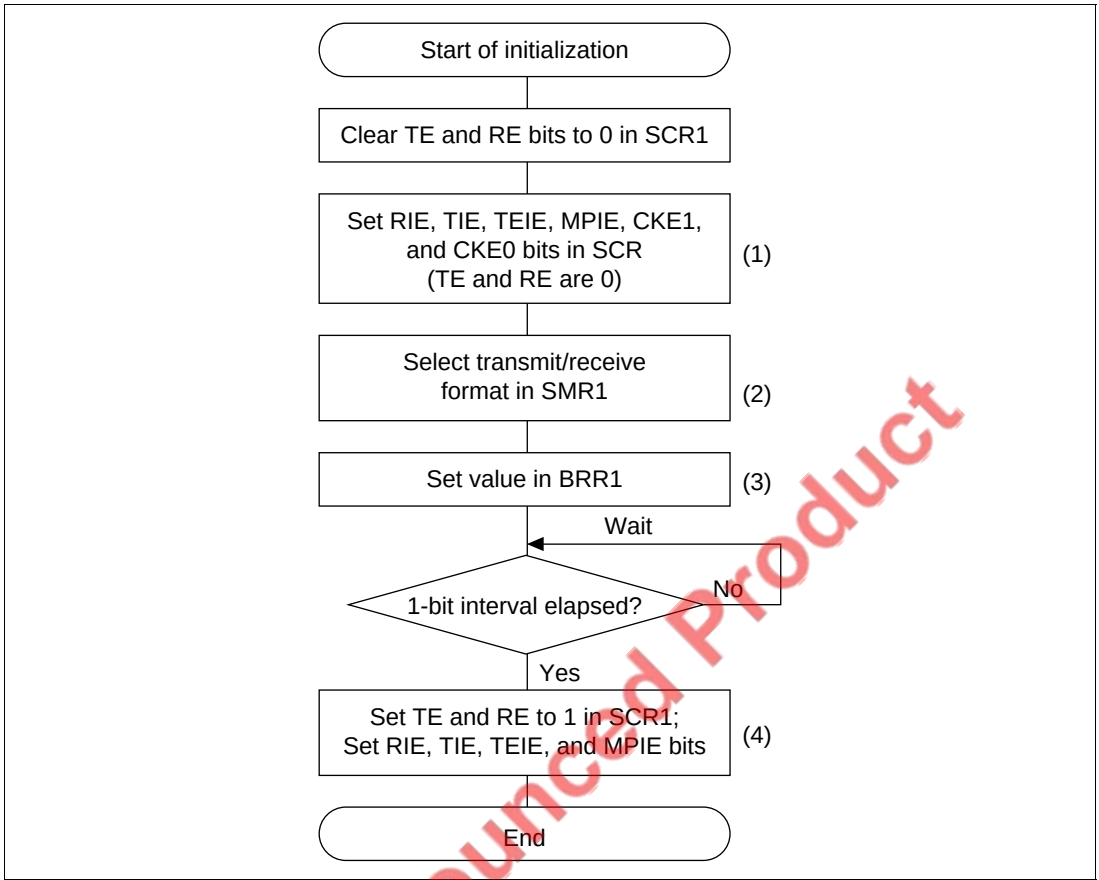


Figure 12.15 Sample Flowchart for SCI Initialization

Transmitting Serial Data (Clock Synchronous Mode): Figure 12.16 shows a sample flowchart for transmitting serial data and indicates the procedure to follow.

1. SCI1 initialization: Set the TxD pin function with the PFC.
2. SCI1 status check and transmit data write: Read SSR1, check that the TDRE flag is 1, then write transmit data in TDR1 and clear the TDRE flag to 0.
3. To continue transmitting serial data: After checking that the TDRE flag is 1, indicating that data can be written, write data in TDR1, then clear the TDRE flag to 0. When the DMAC or DTC is activated by a transmit-data-empty interrupt request (TxI) to write data in TDR1, the TDRE flag is checked and cleared automatically.

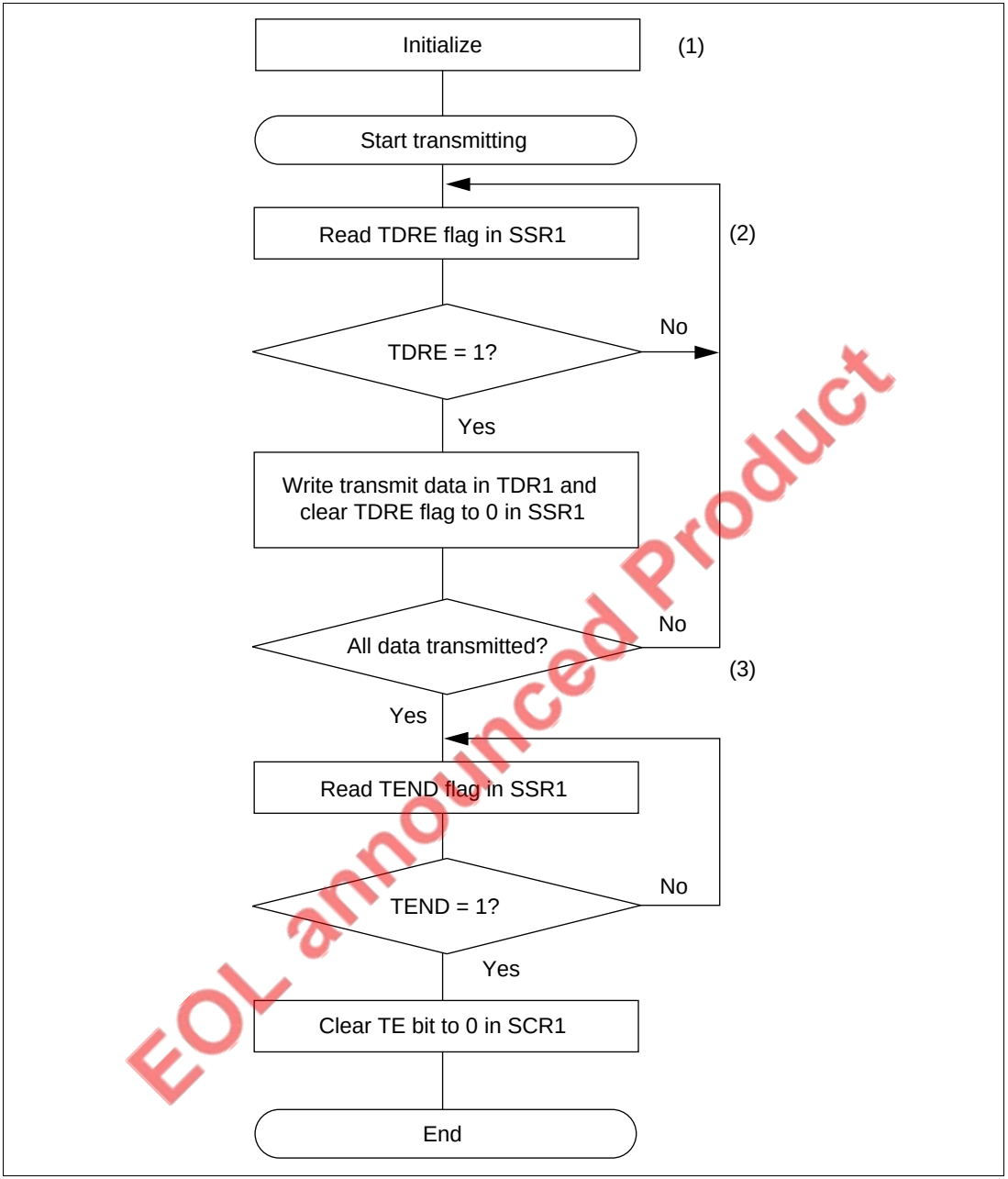


Figure 12.16 Sample Flowchart for Serial Transmitting

Figure 12.17 shows an example of SCI1 transmit operation.

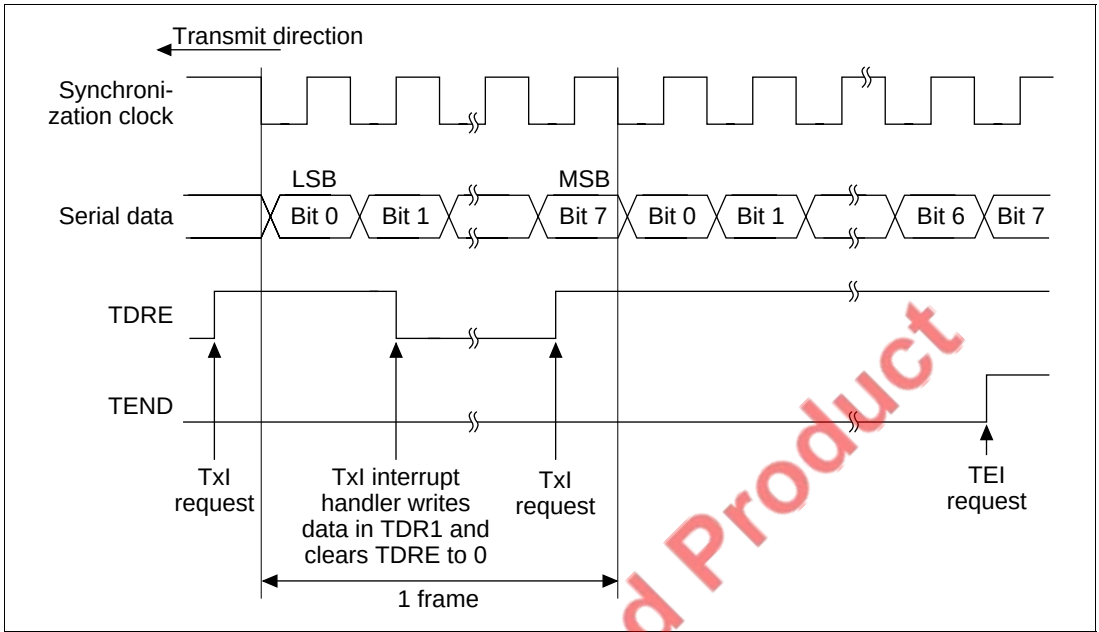


Figure 12.17 Example of SCI1 Transmit Operation

SCI1 serial transmission operates as follows.

1. The SCI1 monitors the TDRE bit in the SSR1. When TDRE is cleared to 0 the SCI1 recognizes that the transmit data register (TDR1) contains new data and loads this data from the TDR1 into the transmit shift register (TSR1).
2. After loading the data from the TDR1 into the TSR1, the SCI1 sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCR1 is set to 1, the SCI1 requests a transmit-data-empty interrupt (TxI) at this time.

   If clock output mode is selected, the SCI1 outputs eight synchronous clock pulses. If an external clock source is selected, the SCI1 outputs data in synchronization with the input clock. Data are output from the TxD pin in order from the LSB (bit 0) to the MSB (bit 7).
3. The SCI1 checks the TDRE bit when it outputs the MSB (bit 7). If TDRE is 0, the SCI1 loads data from the TDR1 into the TSR1, then begins serial transmission of the next frame. If TDRE is 1, the SCI1 sets the TEND bit in the SSR1 to 1, transmits the MSB, then holds the transmit data pin (TxD) in the MSB state. If the transmit-end interrupt enable bit (TEIE) in the SCR1 is set to 1, a transmit-end interrupt (TEI) is requested at this time.
4. After the end of serial transmission, the SCK pin is held in the high state.

Receiving Serial Data (Clock Synchronous Mode): Figure 12.18 shows a sample flowchart for receiving serial data. When switching from the asynchronous mode to the clock synchronous mode, make sure that ORER, PER, and FER are cleared to 0. If PER or FER is set to 1, the RDRF bit will not be set and both transmitting and receiving will be disabled.

The procedure for receiving serial data is listed below:

1. SCI1 initialization: Set the RxD pin using the PFC.
2. Receive error handling: If a receive error occurs, read the ORER bit in SSR1 to identify the error. After executing the necessary error handling, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
3. SCI1 status check and receive data read: Read the serial status register (SSR1), check that RDRF is set to 1, then read receive data from the receive data register (RDR1) and clear RDRF to 0. The RxI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. Continue receiving serial data: Read RDR1, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. If the DMAC is started by a receive-data-full interrupt (RxI) to read RDR1, the RDRF bit is cleared automatically so this step is unnecessary.

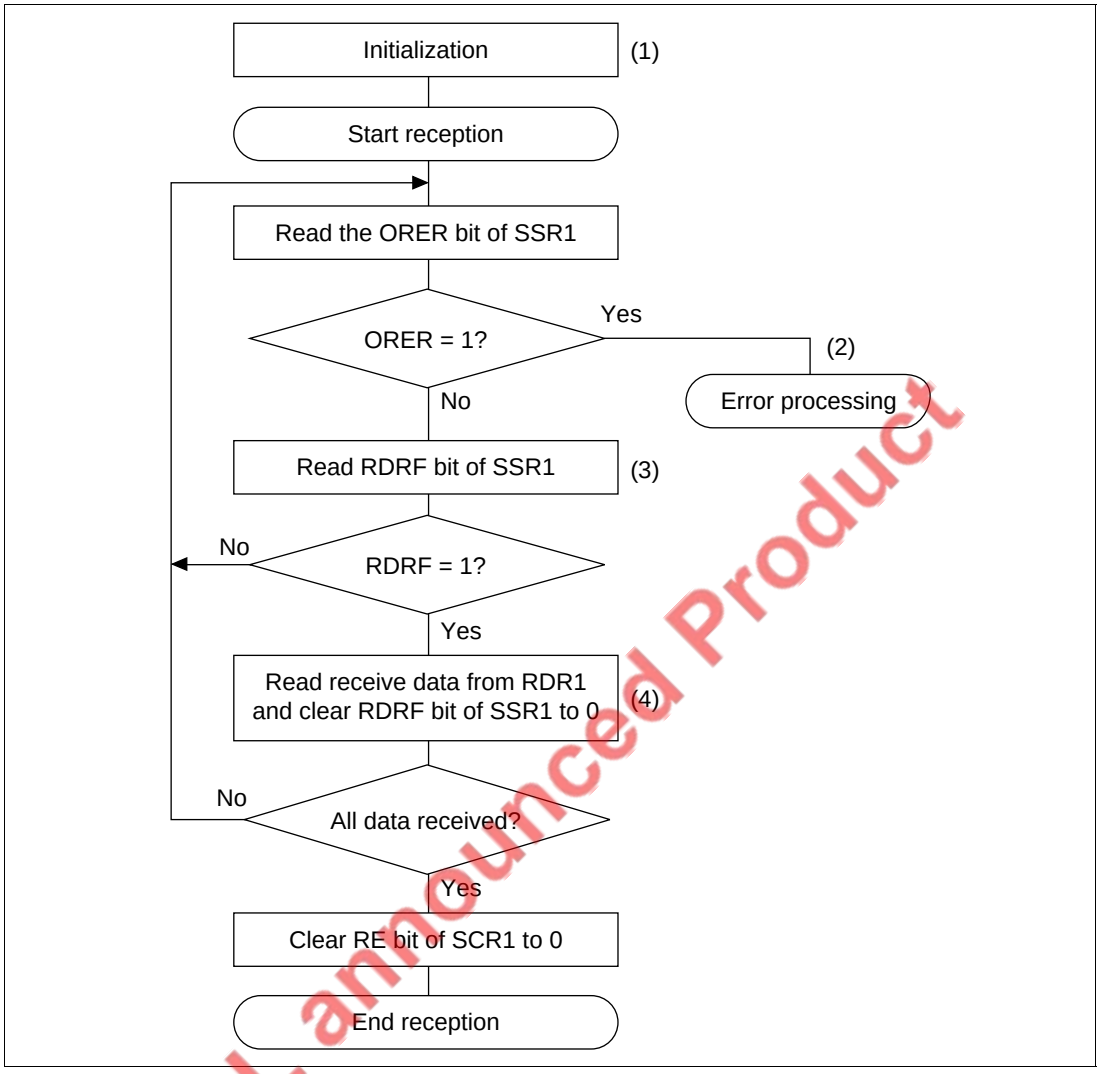


Figure 12.18 Sample Flowchart for Serial Receiving

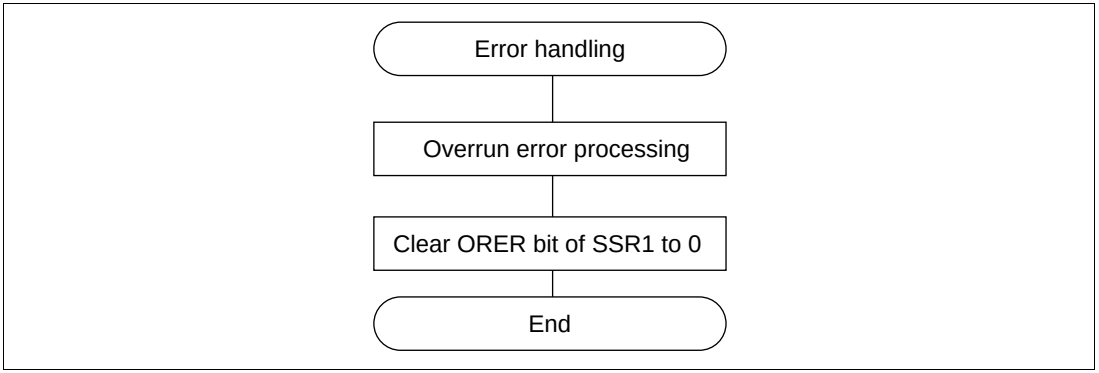


Figure 12.18 Sample Flowchart for Serial Receiving (cont)

Figure 12.19 shows an example of the SCI1 receive operation.

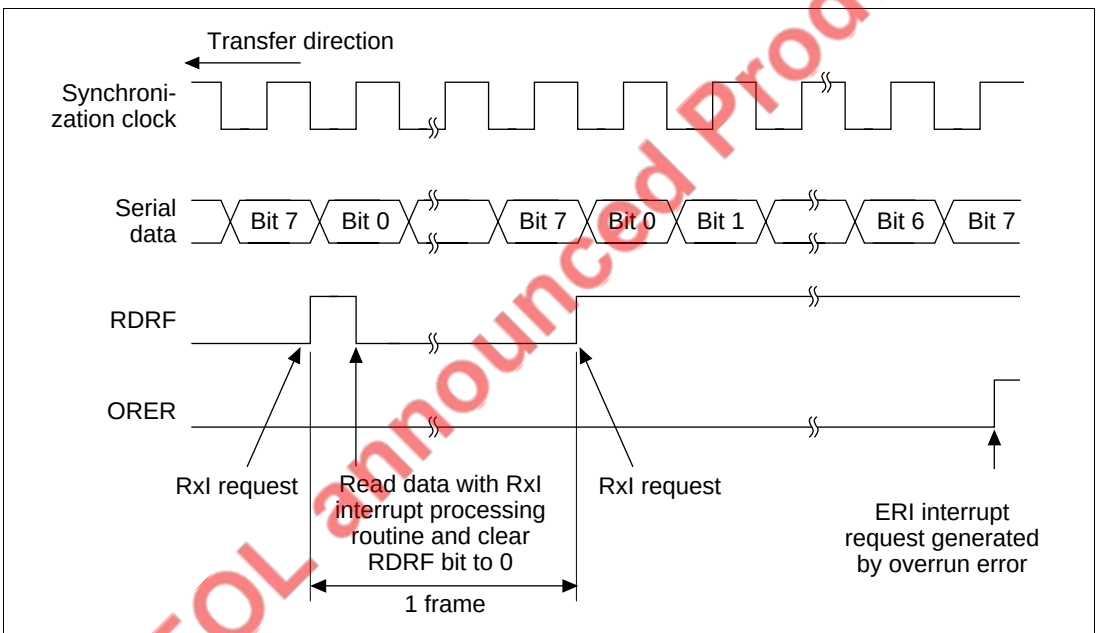


Figure 12.19 Example of SCI1 Receive Operation

In receiving, the SCI1 operates as follows:

1. The SCI1 synchronizes with serial clock input or output and initializes internally.
2. Receive data is shifted into the RSR1 in order from the LSB to the MSB. After receiving the data, the SCI1 checks that RDRF is 0 so that receive data can be loaded from the RSR1 into the RDR1. If this check passes, the SCI1 sets RDRF to 1 and stores the received data in the RDR1. If the check does not pass (receive error), the SCI1 operates as indicated in table 12.11 and no further transmission or reception is possible. If the error flag is set to 1, the RDRF bit is

not set to 1 during reception, even if the RDRF bit is 0 cleared. When restarting reception, be sure to clear the error flag.

3. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCR1, the SCI1 requests a receive-data-full interrupt (RxI). If the ORER bit is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCR1 is also set to 1, the SCI1 requests a receive-error interrupt (ERI).

Transmitting and Receiving Serial Data Simultaneously (Clock Synchronous Mode): Figure 12.20 shows a sample flowchart for transmitting and receiving serial data simultaneously. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. SCI1 initialization: Set the TxD and RxD pins using the PFC.
2. SCI1 status check and transmit data write: Read the serial status register (SSR1), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR1) and clear TDRE to 0. The TxI interrupt can also be used to determine if the TDRE bit has changed from 0 to 1.
3. Receive error handling: If a receive error occurs, read the ORER bit in SSR1 to identify the error. After executing the necessary error processing, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
4. SCI1 status check and receive data read: Read the serial status register (SSR1), check that RDRF is set to 1, then read receive data from the receive data register (RDR1) and clear RDRF to 0. The RxI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
5. Continue transmitting and receiving serial data: Read the RDRF bit and RDR1, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. Before the MSB (bit 7) of the current frame is received, read the TDRE bit and check that it is safe to write (if it reads 1); if so, write data in TDR1, then clear TDRE to 0.

Note: In switching from transmitting or receiving to simultaneous transmitting and receiving, simultaneously clear the TE bit and RE bit to 0, then simultaneously set the TE bit and RE bit to 1.

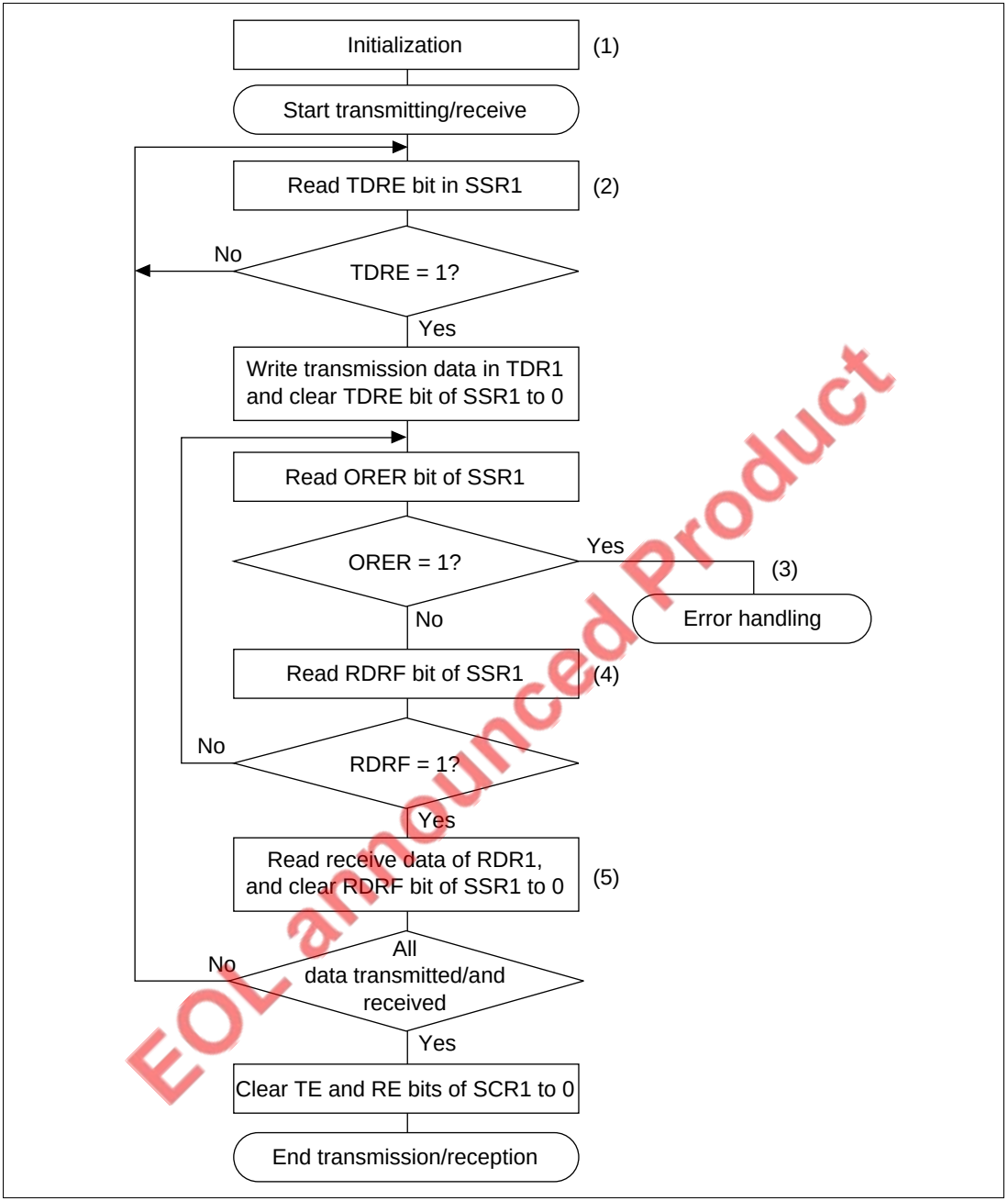


Figure 12.20 Sample Flowchart for Serial Transmission

12.4 Interrupt

The SCI1 has four interrupt sources: transmit-end (TEI), receive-error (ERI), receive-data-full (RxI), and transmit-data-empty (TxI). Table 12.12 lists the interrupt sources and indicates their priority. These interrupts can be enabled and disabled by the TIE, RIE, and TEIE bits in the serial control register (SCR1). Each interrupt request is sent separately to the interrupt controller.

TxI is requested when the TDRE bit in the serial status register (SSR1) is set to 1.

RxI is requested when the RDRF bit in the SSR1 is set to 1.

ERI is requested when the ORER, FER, or PER bit in the SSR1 is set to 1.

TEI is requested when the TEND bit in the SSR1 is set to 1.

Where the TxI interrupt indicates that transmit data writing is enabled, the TEI interrupt indicates that the transmit operation has ended.

Table 12.12 SCI1 Interrupt Sources

Interrupt Source	Description	Priority
ERI	Receive error (ORER, PER, or FER)	High
RxI	Receive data full (RDRF)	
TxI	Transmit data empty (TDRE)	
TEI	Transmit end (TEND)	Low

12.5 Notes on Use

The following points should be noted when using the SCI1.

TDR1 Write and TDRE Flags: The TDRE bit in the serial status register (SSR1) is a status flag indicating loading of transmit data from TDR1 into TSR1. The SCI1 sets TDRE to 1 when it transfers data from TDR1 to TSR1. Data can be written to TDR1 regardless of the TDRE bit status. If new data is written in TDR1 when TDRE is 0, however, the old data stored in TDR1 will be lost because the data has not yet been transferred to the TSR1. Before writing transmit data to the TDR1, be sure to check that TDRE is set to 1.

Simultaneous Multiple Receive Errors: Table 12.13 indicates the state of the SSR1 status flags when multiple receive errors occur simultaneously. When an overrun error occurs, the RSR1 contents cannot be transferred to the RDR1, so receive data is lost.

Table 12.13 SSR1 Status Flags and Transfer of Receive Data

	SSR1 Status Flags				Receive Data Transfer
Receive Error Status	**RDRF**	**ORER**	**FER**	**PER**	**RSR1 → RDR1**
Overrun error	1	1	0	0	X
Framing error	0	0	1	0	O
Parity error	0	0	0	1	O
Overrun error + framing error	1	1	1	0	X
Overrun error + parity error	1	1	0	1	X
Framing error + parity error	0	0	1	1	O
Overrun error + framing error + parity error	1	1	1	1	X

Note: O = Receive data is transferred from RSR1 to RDR1.
X = Receive data is not transferred from RSR1 to RDR1.

Break Detection and Processing: Break signals can be detected by reading the RxD pin directly when a framing error (FER) is detected. In the break state, the input from the RxD pin consists of all 0s, so FER is set and the parity error flag (PER) may also be set. In the break state, the SCI1 receiver continues to operate, so if the FER bit is cleared to 0, it will be set to 1 again.

Sending a Break Signal: The TxD pin becomes a general I/O pin with the I/O direction and level determined by the I/O port data register (DR) and pin function controller (PFC) control register (CR). These conditions allow break signals to be sent. The DR value is substituted for the marking status until the PFC is set. Consequently, the output port is set to initially output a 1. To send a break in serial transmission, first clear the DR to 0, then establish the TxD pin as an output port

using the PFC. When TE is cleared to 0, the transmission section is initialized regardless of the present transmission status.

Receive Error Flags and Transmitter Operation (Clock Synchronous Mode Only): When a receive error flag (ORER, PER, or FER) is set to 1, the SCI will not start transmitting even if TDRE is set to 1. Be sure to clear the receive error flags to 0 before starting to transmit. Note that clearing RE to 0 does not clear the receive error flags.

Receive Data Sampling Timing and Receive Margin in the Asynchronous Mode: The SCI1 operates on a base clock of 16 times the bit rate frequency in the asynchronous mode. In receiving, the SCI1 synchronizes internally with the falling edge of the start bit, which it samples on the base clock. Receive data is latched on the rising edge of the eighth base clock pulse (figure 12.21).

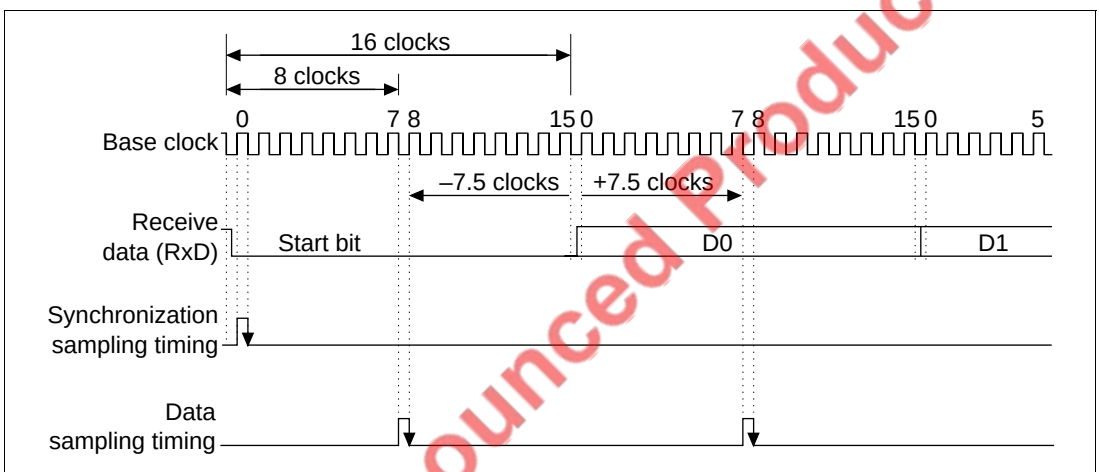


Figure 12.21 Receive Data Sampling Timing in the Asynchronous Mode

The receive margin in the asynchronous mode can therefore be expressed as:

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

M : Receive margin (%)
N : Ratio of clock frequency to bit rate (N = 16)
D : Clock duty cycle (D = 0 to 1.0)
L : Frame length (L = 9 to 12)
F : Absolute deviation of clock frequency

From the equation above, if F = 0 and D = 0.5 the receive margin is 46.875%:

D = 0.5, F = 0
M = $(0.5 - 1/(2 \times 16)) \times 100\%$
= 46.875%

This is a theoretical value. A reasonable margin to allow in system designs is 20 to 30%.

Cautions for Clock Synchronous External Clock Mode

- Set TE = RE = 1 only when the external clock SCK is 1.
- Do not set TE = RE = 1 until at least four clocks after the external clock SCK has changed from 0 to 1.
- When receiving, RDRF is 1 when RE is set to zero 2.5 to 3.5 clocks after the rising edge of the RxD D7 bit SCK input, but it cannot be copied to RDR.

Caution for Clock Synchronous Internal Clock Mode: When receiving, RDRF is 1 when RE is set to zero 1.5 clocks after the rising edge of the RxD D7 bit SCK output, but it cannot be copied to RDR.

Caution for SCI Register Initialization in Standby Mode: The SCR1, SMR1, and BRR1 registers incorporated into the serial communication interface (SCI) of the SH7018F user chip are not initialized in standby mode. Consequently, if a transition is made to standby mode while the TIE bit in SCR1 is set to 1, the TDRE bit in the serial status register (SSR1) will be set to 1, and after recovery from standby mode a transmit-data-empty interrupt (TXI) will be generated. When switching to standby mode, therefore, coding that initializes the SCR1, SMR1, and BRR1 registers must be inserted immediately before the relevant SLEEP instruction.

Section 13 A/D Converter (A/D)

13.1 Overview

The A/D converter has 10-bit resolution, and can select from a maximum of eight channels of analog inputs.

13.1.1 Features

The A/D converter has the following features:

- 10-bit resolution
- Eight input channels
- High-speed conversion
  - Minimum conversion time: 6.7 µs per channel (for 20-MHz operation)
- Two operating modes: single mode or scan mode
  - Single mode: A/D conversion on one channel
  - Scan mode: Continuous A/D conversion on one to four channels
- Four 16-bit data registers
  Conversion results transferred to and stored in data registers corresponding to each channel.
- Sample and hold function
- A/D conversion end interrupt generation
  An A/D conversion end interrupt (ADI) request can be generated on completion of A/D conversion.
- A/D conversion can be started by MTU trigger input.

13.1.2 Block Diagram

Figure 13.1 is the block diagram of the A/D converter.

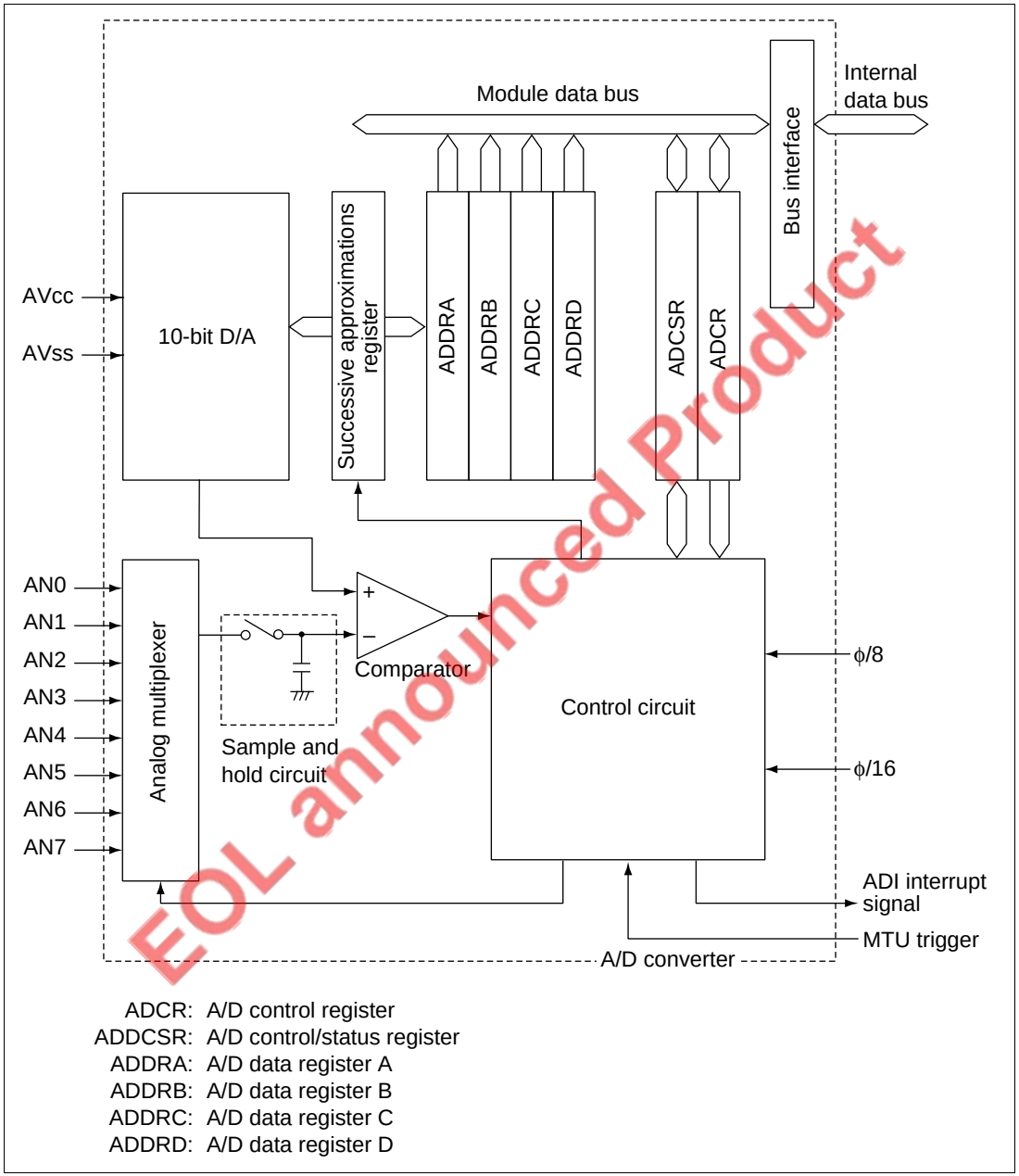


Figure 13.1 A/D Converter Block Diagram

13.1.3 Pin Configuration

Table 13.1 shows the input pins used by the A/D converter.

The seven analog input pins are divided into two groups: group 0, comprising analog input pins 0 to 3 (AN0 to AN3), and group 1, comprising analog input pins 4 to 7 (AN4 to AN7).

The AV_{CC} and AV_{SS} pins are for the A/D converter internal analog section power supply.

Table 13.1 Pin Configuration

Pin	Abbreviation	I/O	Function
Analog supply	AV_{CC}	I	Analog section power supply
Analog ground	AV_{SS}	I	Analog section ground and A/D conversion reference voltage
Analog input 0	AN0	I	Analog input group 0
Analog input 1	AN1	I	
Analog input 2	AN2	I	
Analog input 3	AN3	I	
Analog input 4	AN4	I	Analog input group 1
Analog input 5	AN5	I	
Analog input 6	AN6	I	
Analog input 7	AN7	I	

13.1.4 Register Configuration

Table 13.2 shows the configuration of the A/D converter registers.

Table 13.2 Register Configuration

Name	Abbreviation	R/W	Initial Value	Address	Access Size
A/D data register AH	ADDRAH	R	H'00	H'FFFF8420	8,16
A/D data register AL	ADDRAL	R	H'00	H'FFFF8421	16
A/D data register BH	ADDRBH	R	H'00	H'FFFF8422	8,16
A/D data register BL	ADDRBL	R	H'00	H'FFFF8423	16
A/D data register CH	ADDRCH	R	H'00	H'FFFF8424	8,16
A/D data register CL	ADDRCL	R	H'00	H'FFFF8425	16
A/D data register DH	ADDRDH	R	H'00	H'FFFF8426	8,16
A/D data register DL	ADDRDL	R	H'00	H'FFFF8427	16
A/D control/status register	ADCSR	R/(W)*	H'00	H'FFFF8428	8,16
A/D control register	ADCR	R/W	H'7F	H'FFFF8429	8,16

Note: * Only 0 can be written to bit 7 to clear the flag.

13.2 Register Descriptions

13.2.1 A/D Data Registers A to D (ADDRA to ADDRD)

The A/D data registers (ADDR) are 16-bit read-only registers for storing A/D conversion results. There are four of these registers, ADDRA through ADDRD.

The A/D-converted data is 10-bit data which is transferred to the ADDR for the selected channel for storage. The upper 8 bits of the converted data correspond to the upper byte of the ADDR, and the lower 2 bits correspond to the lower byte. Bits 5 to 0 of the lower byte of the ADDR are reserved. These bits are always read as 0. The write value should always be 0. Table 13.3 shows the correspondence between the analog input channels and the ADDR registers.

The ADDR registers can be read by the CPU at all times. The upper byte is read directly, but the lower byte data is transferred via a temporary register (TEMP). For details, see section 13.3, CPU Interface.

The ADDR registers are initialized to H'0000 by a power-on reset.

Bit:	15	14	13	12	11	10	9	8
ADDRn:	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
ADDRn:	AD1	AD0	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

(n = A to D)

Table 13.3 Correspondence between Analog Input Channels and ADDRA-ADDRD

Analog Input Channel		
Group 0	**Group 1**	**A/D Data Register**
AN0	AN4	ADDRA
AN1	AN5	ADDRB
AN2	AN6	ADDRC
AN3	AN7	ADDRD

13.2.2 A/D Control/Status Register (ADCSR)

The ADCSR is an 8-bit read/write register used for A/D conversion operation control and to indicate status.

The ADCSR is initialized to H'00 by power-on reset.

Bit:	7	6	5	4	3	2	1	0
	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * The only value that can be written is a 0 to clear the flag.

- Bit 7—A/D End Flag (ADF): This status flag indicates that A/D conversion has ended.

Bit 7: ADF	Description
0	Clear conditions (Initial value) With ADF = 1, by reading the ADF flag then writing 0 in ADF
1	Set conditions • Single mode: When A/D conversion ends after conversion for all designated channels • Scan mode: After one round of A/D conversion for all specified channels

- Bit 6—A/D Interrupt Enable (ADIE): Enables or disables interrupt requests (ADI) after A/D conversion ends.

Bit 6: ADIE	Description
0	Disables interrupt requests (ADI) after A/D conversion ends (Initial value)
1	Enables interrupt requests (ADI) after A/D conversion ends

- Bit 5—A/D Start (ADST): Selects start or stop for A/D conversion.

  The ADST bit remains set to 1 during A/D conversion. It can also be set to 1 by MTU trigger input.

Bit 5: ADST	Description
0	A/D conversion halted (Initial value)
1	Single mode: Start A/D conversion. Automatically cleared to 0 after conversion for the designated channel ends. Scan mode: Start A/D conversion. Continuous conversion until 0 cleared by software, and by power-on reset.

- Bit 4—Scan Mode (SCAN): Selects single mode or scan mode for A/D conversion. For details of the operation in single mode and scan mode, see section 13.4, Operation. Change the mode only when ADST = 0.

Bit 4: SCAN	Description
0	Single mode (Initial value)
1	Scan mode

- Bit 3—Clock Select (CKS): Sets the A/D conversion time. Change the conversion time only when ADST = 0.

Bit 3: CKS	Description	
0	Conversion time = 266 states (max.)	(Initial value)
1	Conversion time = 134 states (max.)	

- Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0): These bits, along with the SCAN bit, select the analog input channel.

  Change the channel selection only when ADST = 0.

			Description	
Bit 2: CH2	Bit 1: CH1	Bit 0: CH0	Single Mode	Scan Mode
0	0	0	AN0 (Initial value)	AN0 (Initial value)
0	0	1	AN1	AN0, AN1
0	1	0	AN2	AN0 to AN2
0	1	1	AN3	AN0 to AN3
1	0	0	AN4	AN4
1	0	1	AN5	AN4, AN5
1	1	0	AN6	AN4 to AN6
1	1	1	AN7	AN4 to AN7

13.2.3 A/D Control Register (ADCR)

The A/D control register (ADCR) is an 8-bit read/write register that enables or disables starting of A/D conversion by MTU trigger input. The ADCR is initialized to H'7F by a power-on reset.

Bit:	7	6	5	4	3	2	1	0
	TRGE	—	—	—	—	—	—	—
Initial value:	0	1	1	1	1	1	1	1
R/W:	R/W	R	R	R	R	R	R	R

- Bit 7—Trigger Enable (TRGE): Enables or disables starting of A/D conversion by MTU trigger input.

Bit 7: TRGE	Description	
0	Disables A/D conversion start by MTU trigger input	(Initial value)
1	A/D conversion is started by MTU trigger	

- Bits 6 to 0—Reserved: These bits are always read as 1. The write value should always be 1.

13.3 CPU Interface

ADDRA to ADDRD are 16-bit registers, but they are connected to the CPU by an 8-bit data bus. Therefore, while the upper byte is accessed directly by the CPU, the lower byte is accessed via an 8-bit temporary register (TEMP).

Data is read from an ADDR register as follows. When the upper byte is read, the upper-byte value is transferred directly to the CPU and the lower-byte value is transferred into TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the CPU.

When reading an ADDR register, always read the upper byte before the lower byte. This operation can be performed by reading ADDR from the upper byte address using a word transfer instruction (such as MOV.W). It is possible to read only the upper byte, but if only the lower byte is read, incorrect data may be obtained.

Figure 13.2 shows the data flow for access to an ADDR register.

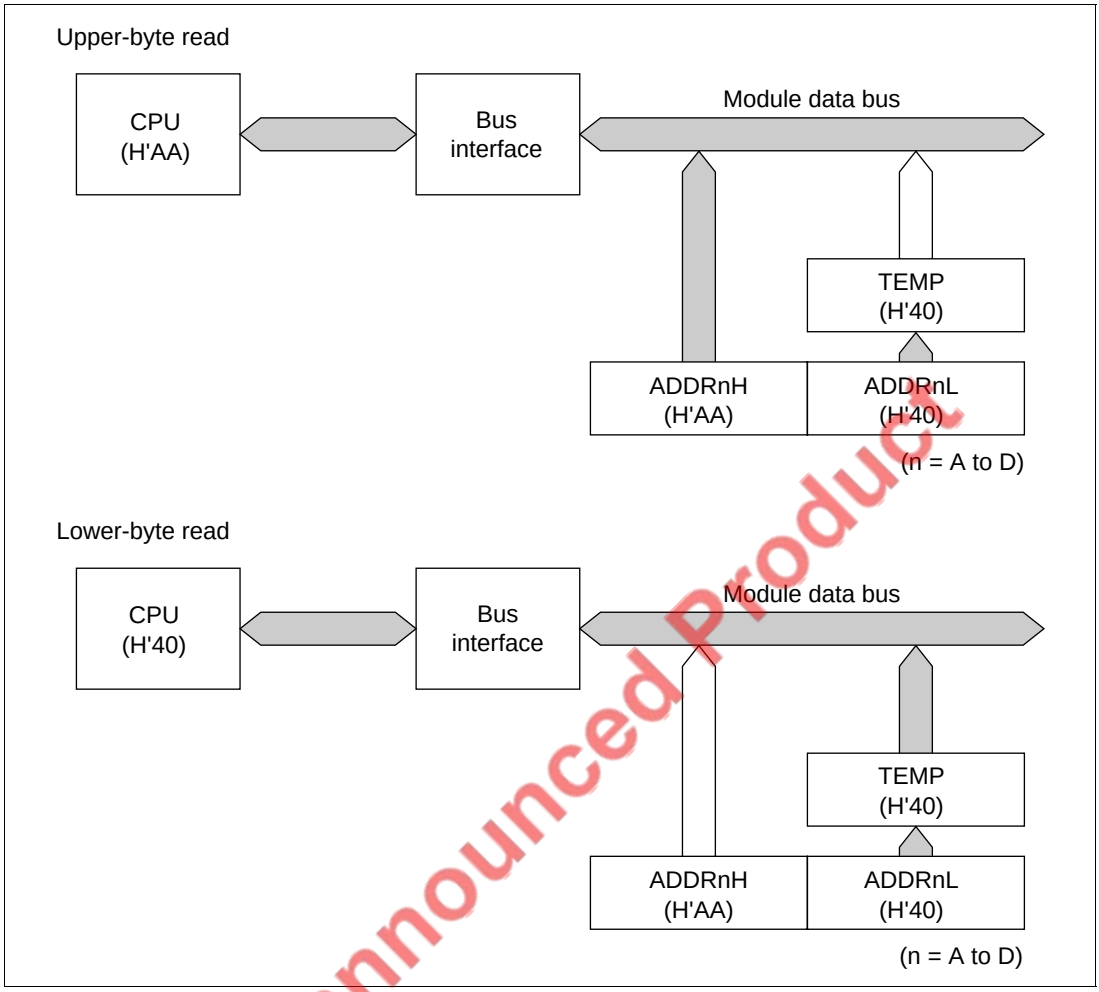


Figure 13.2 ADDR Access Operation (Reading H'AA40)

13.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution. It has two operating modes: single mode and scan mode.. The operation in these two modes is described below.

13.4.1 Single Mode (SCAN = 0)

Single mode should be selected for A/D conversion on only one channel. A/D conversion starts when the ADST bit in the A/D control/status register (ADCSR) is set to 1 by software or MTU trigger input. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends.

When conversion ends, the ADF bit in ADCSR is set to 1. If the ADIE bit in ADCSR is also 1, an ADI interrupt is requested. To clear the ADF bit, first read ADF when set to 1, then write 0 in ADF.

To prevent incorrect operation, A/D conversion should be halted by clearing the ADST bit to 0 before changing the mode or analog input channel. After the change is made, A/D conversion is restarted by setting the ADST bit to 1 (the mode or channel change and setting of the ADST bit can be carried out simultaneously).

An example of the operation when analog input channel 1 (AN1) is selected and A/D conversion is performed in single mode is described below. Figure 13.3 shows a timing diagram for this example.

1. Single mode is selected (SCAN = 0), input channel AN1 is selected (CH2 = CH1 = 0, CH0 = 1), the A/D interrupt request is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).
2. When A/D conversion is completed, the result is transferred to ADDRB. At the same time ADF is set to 1, ADST is cleared to 0, and the A/D converter becomes idle.
3. Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.
4. The A/D interrupt service routine is started.
5. The routine reads ADF set to 1, then writes 0 in ADF.
6. The routine reads and processes the conversion result (ADDRB).
7. Execution of the A/D interrupt service routine ends. After this, if the ADST bit is set to 1, A/D conversion starts again and steps 2 to 7 are repeated.

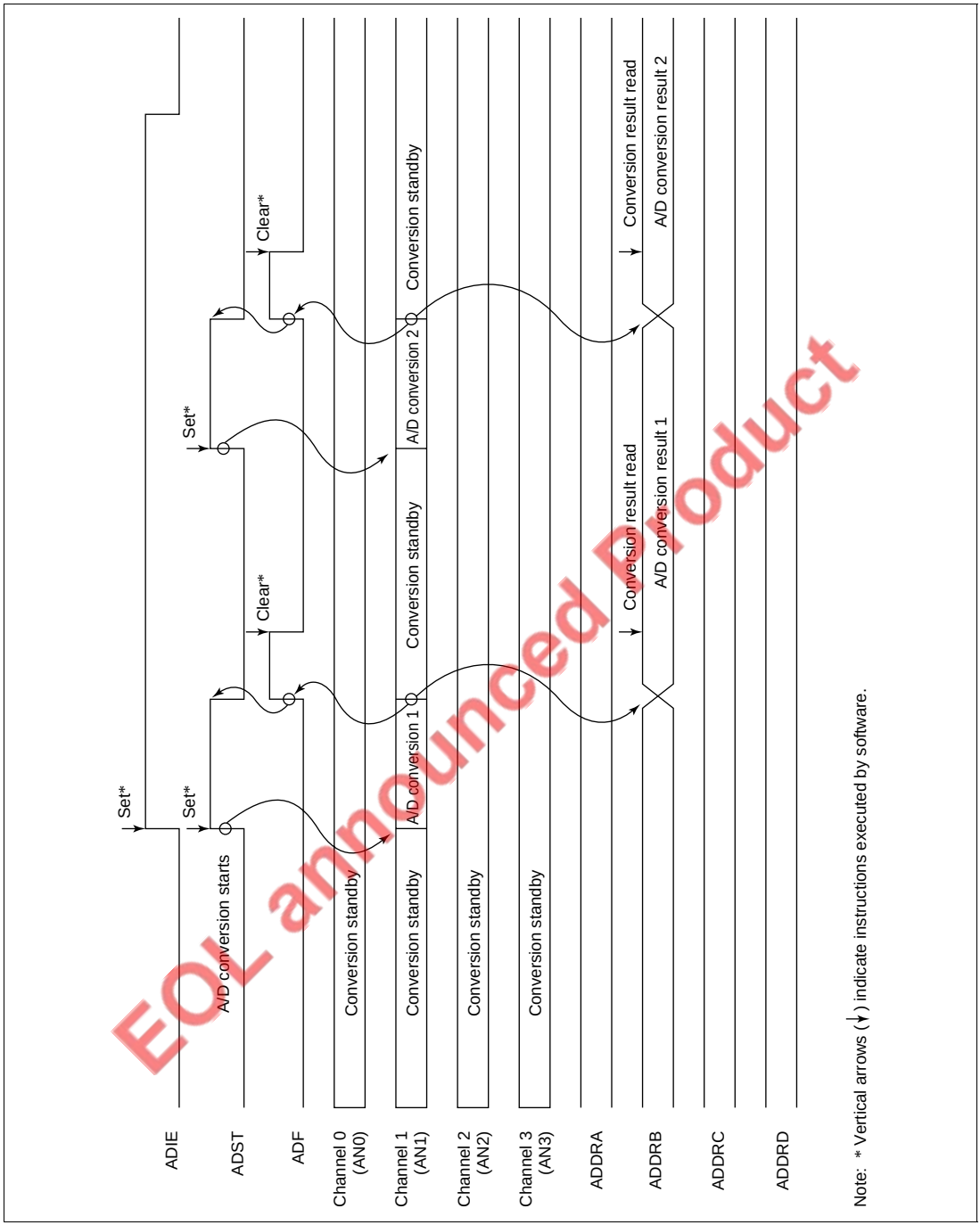


Figure 13.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)

13.4.2 Scan Mode (SCAN = 1)

Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit in the A/D control/status register (ADCSR) is set to 1 by software or MTU trigger input, A/D conversion starts on the first channel in the group (AN0 when CH2 = 0; AN4 when CH1 = 1).

When more than one channel has been selected, A/D conversion starts on the second channel (AN1 or AN5) as soon as conversion ends on the first channel.

A/D conversion is performed repeatedly on all the selected channels until the ADST bit is cleared to 0. The conversion results are transferred to and stored in the ADDR register for each channel.

To prevent incorrect operation, A/D conversion should be halted by clearing the ADST bit to 0 before changing the mode or analog input channels. After the change is made, the first channel is selected and A/D conversion is restarted by setting the ADST bit to 1 (the mode or channel change and setting of the ADST bit can be carried out simultaneously).

An example of the A/D conversion operation in scan mode when three channels (AN0 to AN2) in group 0 are selected is described below. Figure 13.4 shows a timing diagram for this example.

1. Scan mode is selected (SCAN = 1), group 0 is selected as the scan group (CH2 = 0), analog input channels AN0-AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1).
2. A/D conversion starts on the first channel (AN0), and when completed, the result is transferred to ADDRA. Next, conversion of the second channel (AN1) starts automatically.
3. Conversion proceeds in the same way through the third channel (AN2).
4. When conversion is completed for all the selected channels (AN0 to AN2), ADF is set to 1, the first channel (AN0) is selected again, and conversion is performed on that channel. If the ADIE bit is also 1, an ADI interrupt is requested when conversion is completed.
5. Steps 2 to 4 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After this, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).

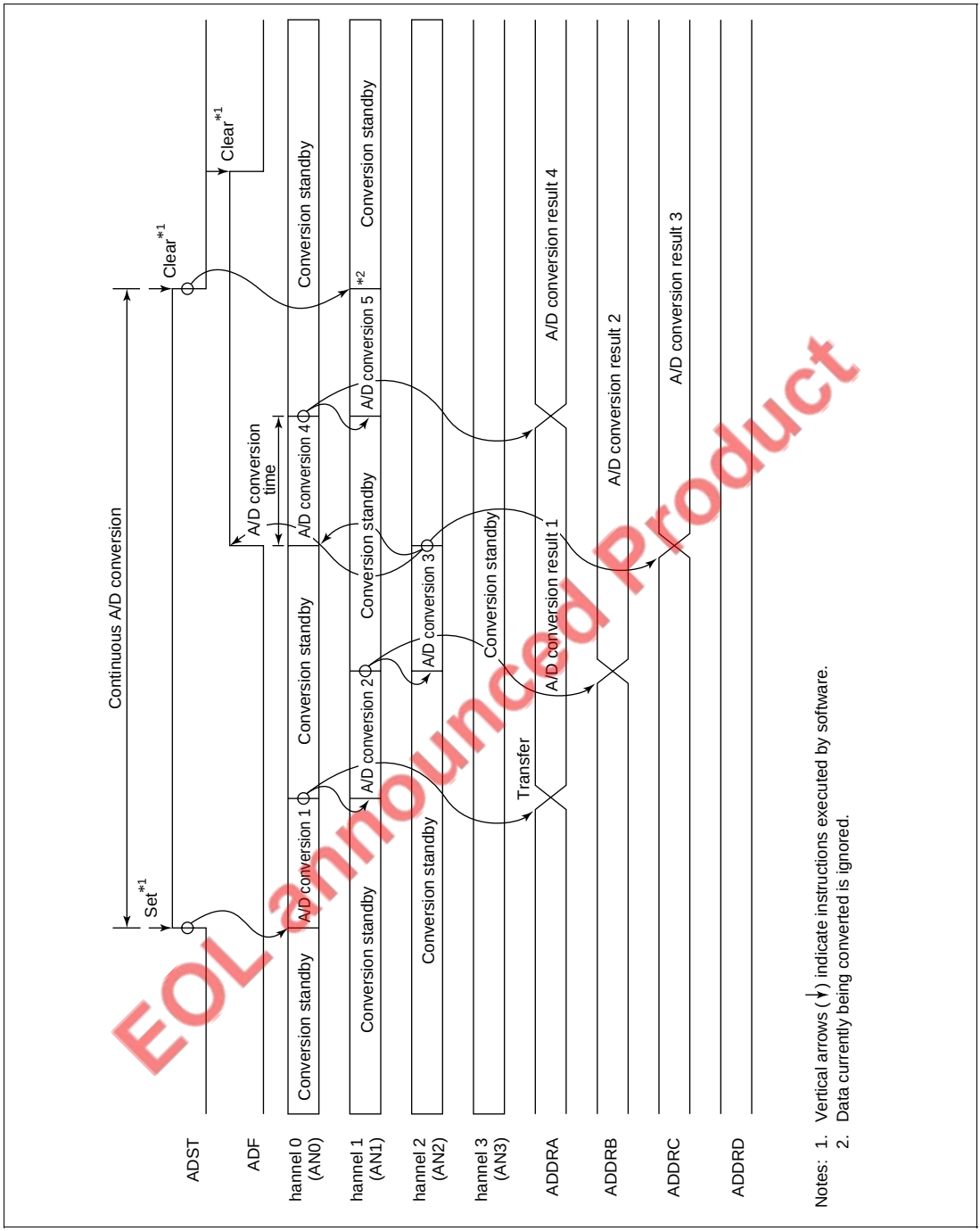


Figure 13.4 Example of A/D Converter Operation (Scan Mode, Channels AN0 to AN2 Selected)

13.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample and hold circuit. The A/D converter samples the analog input at time t_D after the ADST bit is set to 1 in the A/D control/status register (ADCSR), then starts conversion. Figure 13.5 shows the A/D conversion timing, and table 13.4 shows A/D conversion times.

As shown in figure 13.5, A/D conversion time tCONV consists of A/D conversion start delay time t_D and analog input sampling time t_{SPL}. The length of tD is not fixed, but is determined by the timing of the write to ADSCR. The total conversion time therefore varies within the ranges shown in table 13.4.

In scan mode, the t_{CONV} values given in table 13.4 apply to the first conversion. In the second and subsequent conversions, t_{CONV} is fixed at 256 states when CKS = 0 or 128 states when CKS = 1.

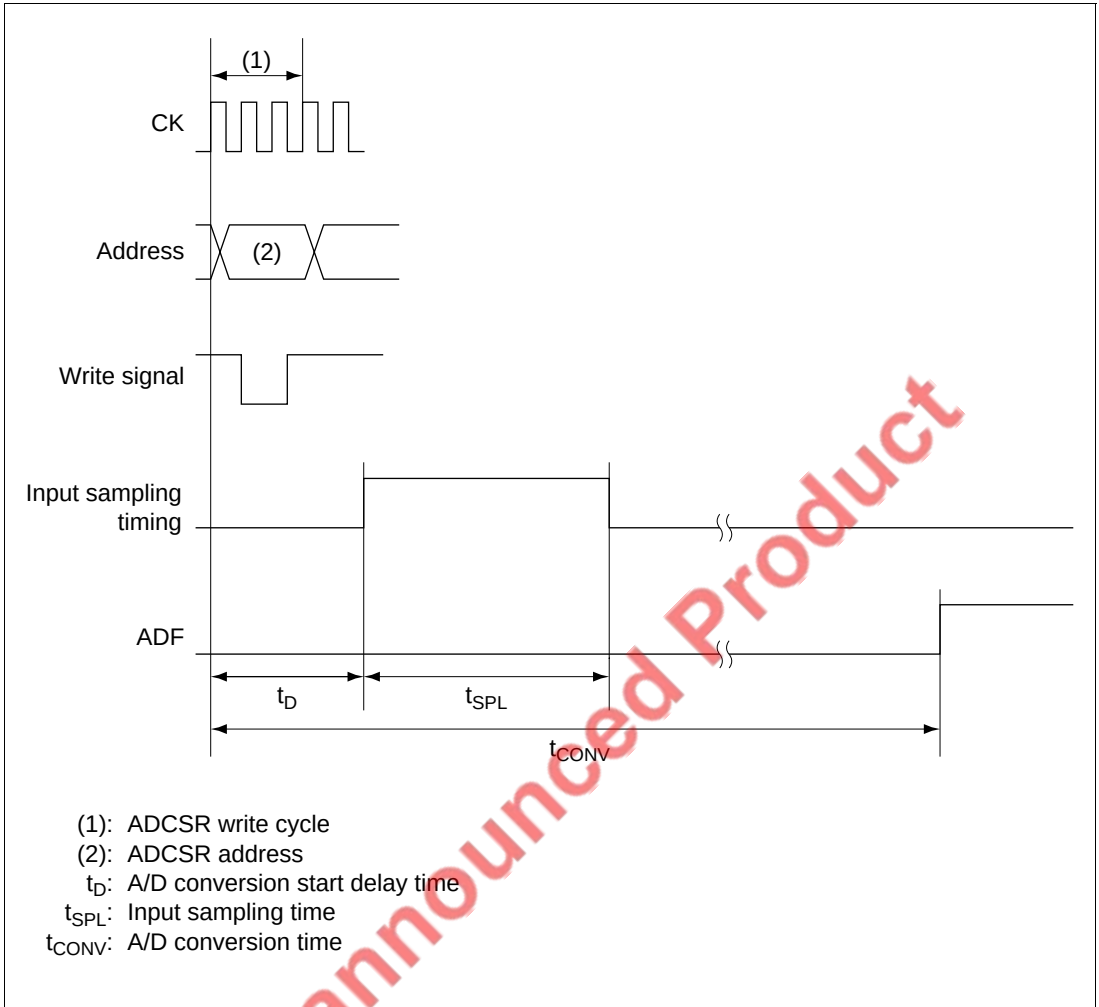


Figure 13.5 A/D Conversion Timing

Table 13.4 A/D Conversion Times (Single Mode)

		CKS = 0			CKS = 1		
	Symbol	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**
A/D conversion start delay time	t_D	10	—	17	6	—	9
Input sampling time	t_{SPL}	—	64	—	—	32	—
A/D conversion time	t_{CCNV}	259	—	266	131	—	134

Note: Unit: states (t_{cyc})

13.4.4 MTU Trigger Input Timing

A/D conversion can also be started by MTU trigger input. When the TRGE bit is set to 1 in the A/D control register (ADCR), input from the MTU functions as trigger input. When an MTU trigger is detected, the ADST bit is set to 1 in the A/D control/status register (ADST), and the A/D converter is started.

Other operations, for both single mode and scan mode, are the same as when the ADST bit is set to 1 by software, .

Figure 13.6 shows the timing for MTU trigger input.

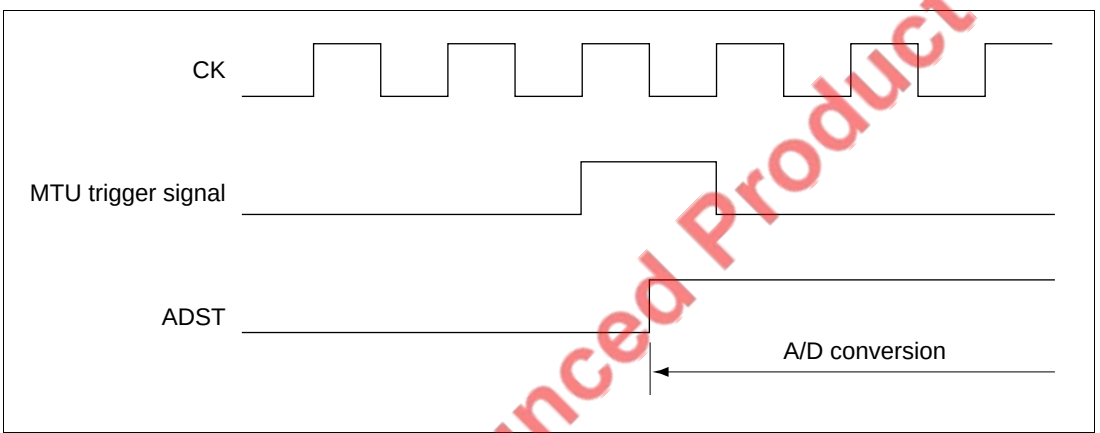


Figure 13.6 External Trigger Input Timing

13.5 A/D Conversion Precision Definitions

The A/D converter converts analog values input from analog input channels to 10-bit digital values by comparing them with an analog reference voltage. In this operation, the absolute precision of the A/D conversion (i.e. the deviation between the input analog value and the output digital value) includes the following kinds of error.

1. Offset error
2. Full-scale error
3. Quantization error
4. Nonlinearity error

The above four kinds of error are described below with reference to figure 13.7. For the sake of clarity, this figure shows 3-bit A/D conversion rather than 10-bit A/D conversion. Offset error (see figure 13.7 (1)) is the deviation between the actual A/D conversion characteristic and the ideal A/D conversion characteristic when the digital output value changes from the minimum value (zero voltage) of 0000000000 (000 in the figure) to 0000000001 (001 in the figure). Full-scale error (see figure 13.7 (2)) is the deviation between the actual A/D conversion characteristic and the ideal A/D conversion characteristic when the digital output value changes from 1111111110 (110 in the figure) to the maximum value (full-scale voltage) of 111111111 (111 in the figure). Quantization error is the deviation inherent in the A/D converter, given by 1/2 LSB (see figure 13.7 (3)). Nonlinearity error is the deviation between the actual A/D conversion characteristic and the ideal A/D conversion characteristic from zero voltage to full-scale voltage (see figure 13.7 (4)). This does not include offset error, full-scale error, and quantization error.

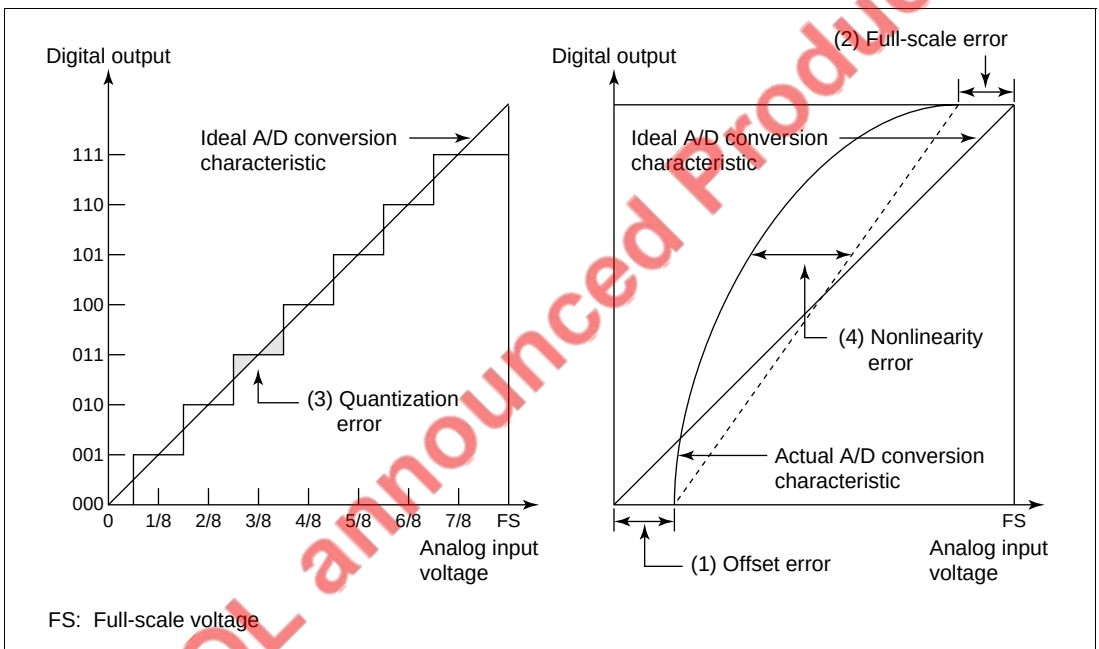


Figure 13.7 A/D Conversion Precision Definitions

13.6 Notes on Use

The following points should be noted when using the A/D converter.

13.6.1 Analog Voltage Settings

Analog Input Voltage Range: The voltage applied to analog input pins during A/D conversion should be in the range $AV_{SS} \leq ANn \leq AV_{CC}$ (n = 0 to 7).

AV_{CC} and AV_{SS} input voltages: For the AV_{CC} and AV_{SS} input voltages, set AV_{CC} = 3.3 V ± 10%, and $AV_{SS} = V_{SS}$. When the A/D converter is not used, set $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$.

13.6.2 Handling of Analog Input Pins

To prevent damage from surges and other abnormal voltages at the analog input pins (AN0 to AN7), connect a protection circuit such as that shown in figure 13.8. This circuit also includes a CR filter function that suppresses error due to noise. The circuit shown here is only a design example; circuit constants must be decided on the basis of the actual operating conditions.

Figure 13.9 shows an equivalent circuit for the analog input pins, and table 13.5 summarizes the analog input pin specifications.

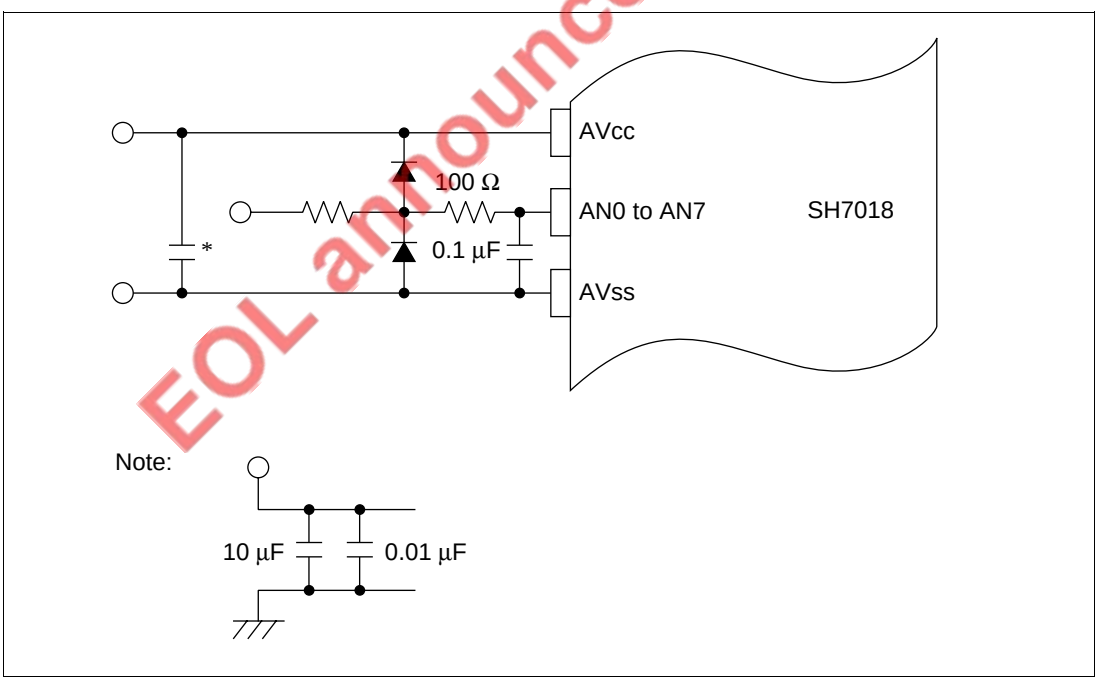
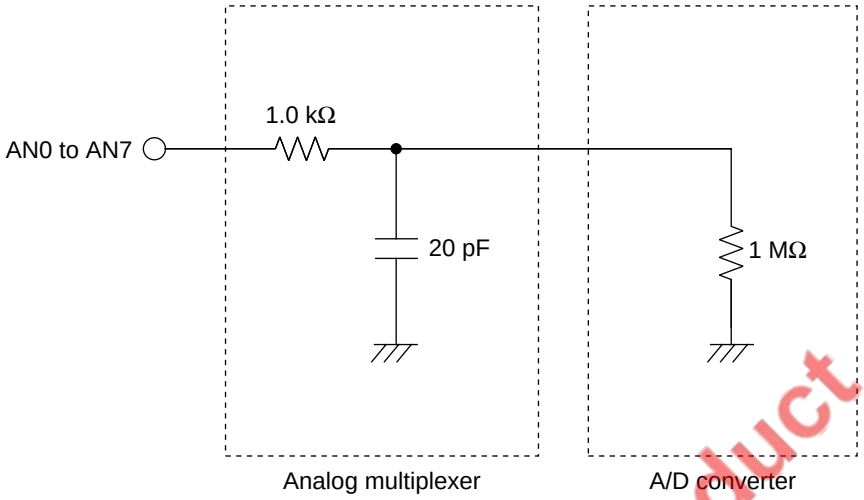


Figure 13.8 Example of Analog Input Pin Protection Circuit



Note: Values are reference values.

Figure 13.9 Analog Input Pin Equivalent Circuit

Table 13.5 Analog Input Pin Specifications

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Permitted signal source impedance	—	3	kΩ

Section 14 Pin Function Controller (PFC)

14.1 Overview

The pin function controller (PFC) consists of registers for selecting multiplex pin functions and their input/output direction. Table 14.1 shows the SH7018's multiplex pins. The functions of the multiplex pins are determined by the operating mode. Table 14.2 and table 14.3 show the pin functions in each operating mode and the initial values.

Table 14.1 Multiplex Pins

Port	Function 1 (Related Module)	Function 2 (Related Module)	Function 3 (Related Module)	Function 4 (Related Module)	Pin No.
A	PA15 I/O (port)	CK output (CPG)			75
A	PA14 I/O (port)	$\overline{RD}$ output (BSC)			33
A	PA12 I/O (port)	$\overline{WRL}$ output (BSC)			34
A	PA11 I/O (port)	$\overline{CS1}$ output (BSC)			35
A	PA10 I/O (port)	$\overline{CS0}$ output (BSC)			36
A	PA9 I/O (port)		$\overline{IRQ3}$ (INTC)		42
A	PA8 I/O (port)		$\overline{IRQ2}$ (INTC)		44
A	PA7 I/O (port)		$\overline{CS3}$ output (BSC)		37
A	PA6 I/O (port)		$\overline{CS2}$ output (BSC)		38
A	PA5 I/O (port)	SCK I/O (SCI)			47
A	PA4 I/O (port)	TXD output (SCI)			48
A	PA3 I/O (port)	RXD input (SCI)			49
A	PA2 I/O (port)			$\overline{IRQ0}$ input (INTC)	50
A	PA1 I/O (port)				40
A	PA0 I/O (port)				41
B	PB9 I/O (port)	$\overline{IRQ7}$ input (INTC)	A21 output (BSC)		31
B	PB8 I/O (port)	$\overline{IRQ6}$ input (INTC)	A20 output (BSC)	$\overline{WAIT}$ input (BSC)	46
B	PB7 I/O (port)		A19 output (BSC)		29
B	PB6 I/O (port)	A18 output (BSC)			28
B	PB5 I/O (port)				27
B	PB4 I/O (port)				26
B	PB3 I/O (port)	$\overline{IRQ1}$ input (INTC)			24
B	PB2 I/O (port)				23

Table 14.1 Multiplex Pins (cont)

Port	Function 1 (Related Module)	Function 2 (Related Module)	Function 3 (Related Module)	Function 4 (Related Module)	Pin No.
B	PB1 I/O (port)	A17 output (BSC)			21
B	PB0 I/O (port)	A16 output (BSC)			20
C	PC15 I/O (port)	A15 output (BSC)			19
C	PC14 I/O (port)	A14 output (BSC)			18
C	PC13 I/O (port)	A13 output (BSC)			17
C	PC12 I/O (port)	A12 output (BSC)			16
C	PC11 I/O (port)	A11 output (BSC)			14
C	PC10 I/O (port)	A10 output (BSC)			13
C	PC9 I/O (port)	A9 output (BSC)			12
C	PC8 I/O (port)	A8 output (BSC)			11
C	PC7 I/O (port)	A7 output (BSC)			10
C	PC6 I/O (port)	A6 output (BSC)			9
C	PC5 I/O (port)	A5 output (BSC)			7
C	PC4 I/O (port)	A4 output (BSC)			6
C	PC3 I/O (port)	A3 output (BSC)			5
C	PC2 I/O (port)	A2 output (BSC)			4
C	PC1 I/O (port)	A1 output (BSC)			3
C	PC0 I/O (port)	A0 output (BSC)			2
D	PD7 I/O (port)	D7 I/O (BSC)			53
D	PD6 I/O (port)	D6 I/O (BSC)			54
D	PD5 I/O (port)	D5 I/O (BSC)			56
D	PD4 I/O (port)	D4 I/O (BSC)			58
D	PD3 I/O (port)	D3 I/O (BSC)			59
D	PD2 I/O (port)	D2 I/O (BSC)			60
D	PD1 I/O (port)	D1 I/O (BSC)			61
D	PD0 I/O (port)	D0 I/O (BSC)			62
E	PE14 I/O (port)				88
E	PE13 I/O (port)				87
E	PE12 I/O (port)				86
E	PE11 I/O (port)				85
E	PE10 I/O (port)				84

Table 14.1 Multiplex Pins (cont)

Port	Function 1 (Related Module)	Function 2 (Related Module)	Function 3 (Related Module)	Function 4 (Related Module)	Pin No.
E	PE9 I/O (port)				83
E	PE8 I/O (port)				82
E	PE7 I/O (port)	TIOC2B I/O (MTU)			81
E	PE6 I/O (port)	TIOC2A I/O (MTU)			80
E	PE5 I/O (port)	TIOC1B I/O (MTU)			78
E	PE4 I/O (port)	TIOC1A I/O (MTU)			77
E	PE2 I/O (port)	TIOC0C I/O (MTU)			64
E	PE0 I/O (port)	TIOC0A I/O (MTU)			63
F	PF7 input (port)	AN7 input (A/D)			98
F	PF6 input (port)	AN6 input (A/D)			97
F	PF5 input (port)	AN5 input (A/D)			95
F	PF4 input (port)	AN4 input (A/D)			94
F	PF3 input (port)	AN3 input (A/D)			93
F	PF2 input (port)	AN2 input (A/D)			92
F	PF1 input (port)	AN1 input (A/D)			91
F	PF0 input (port)	AN0 input (A/D)			90

14.2 Register Configuration

The PFC registers are listed in table 14.3.

Table 14.3 PFC Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port A IO register L	PAIORL	R/W	H'0000	H'FFFF8386 H'FFFF8387	8, 16, 32
Port A control register L1	PACRL1	R/W	H'4000	H'FFFF838C H'FFFF838D	8, 16, 32
Port A control register L2	PACRL2	R/W	H'0000	H'FFFF838E H'FFFF838F	8, 16, 32
Port B IO register	PBIOR	R/W	H'0000	H'FFFF8394 H'FFFF8395	8, 16, 32
Port B control register 1	PBCR1	R/W	H'0000	H'FFFF8398 H'FFFF8399	8, 16, 32
Port B control register 2	PBCR2	R/W	H'0000	H'FFFF839A H'FFFF839B	8, 16, 32
Port C IO register	PCIOR	R/W	H'0000	H'FFFF8396 H'FFFF8397	8, 16, 32
Port C control register	PCCR	R/W	H'0000	H'FFFF839C H'FFFF839D	8, 16, 32
Port D IO register L	PDIORL	R/W	H'0000	H'FFFF83A6 H'FFFF83A7	8, 16, 32
Port D control register L	PDCRL	R/W	H'0000	H'FFFF83AC H'FFFF83AD	8, 16, 32
Port E IO register	PEIOR	R/W	H'0000	H'FFFF83B4 H'FFFF83B5	8, 16, 32
Port E control register 2	PECR2	R/W	H'0000	H'FFFF83BA H'FFFF83BB	8, 16, 32

14.3 Register Descriptions

14.3.1 Port A IO Register L (PAIORL)

Port A IO register L (PAIORL) is a 16-bit readable/writable register that selects the input/output direction of the pins in port A. The bits of this register correspond to the various pins. PAIORL is enabled when the port A pins function as general input/output (PA15 to PA0) or serial clock (SCK) pins, and disabled otherwise.

When the port A pins function as PA15 to PA0 or SCK, a pin becomes an output when the corresponding bit in PAIORL is set to 1, and an input when the bit is cleared to 0.

PAIORL is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases it retains its previous data.

Bit:	15	14	13	12	11	10	9	8
	PA15IOR	PA14IOR	—	PA12IOR	PA11IOR	PA10IOR	PA9IOR	PA8IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PA7IOR	PA6IOR	PA5IOR	PA4IOR	PA3IOR	PA2IOR	PA1IOR	PA0IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

14.3.2 Port A Control Registers L1 and L2 (PACRL1, PACRL2)

Port A control registers L1 and L2 (PACRL1, PACRL2) are 16-bit readable/writable registers that select the functions of the pins in port A.

PACRL1 is initialized to H'4000 by an external power-on reset. PACRL2 is initialized to H'0000 by an external power-on reset. However, they are not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases they retain their previous data.

Port A Control Register L1 (PACRL1)

Bit:	15	14	13	12	11	10	9	8
	—	PA15MD	—	PA14MD	—	—	—	PA12MD
Initial value:	0	1	0	0	0	0	0	0
R/W:	R	R/W	R	R/W	R	R	R	R/W

Bit:	7	6	5	4	3	2	1	0
	—	PA11MD0	—	PA10MD	PA9MD1	—	PA8MD1	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R	R/W	R/W	R	R/W	R

Bit 15—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 14—PA15 Mode (PA15MD): Selects the function of the PA15/CK pin.

Bit 14: PA15MD	Description	
0	General input/output (PA15)	
1	Clock output (CK)	(Initial value)

Bit 13—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 12—PA14 Mode (PA14MD): Selects the function of the PA14/$\overline{RD}$ pin.

Bit 12: PA14MD	Description	
0	General input/output (PA14)	(Initial value)
1	Read output ($\overline{RD}$)	

Bits 11 to 9—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 8—PA12 Mode (PA12MD): Selects the function of the PA12/$\overline{WRL}$ pin.

Bit 8: PA12MD	Description	
0	General input/output (PA12)	(Initial value)
1	Chip select output ($\overline{WRL}$)	

Bit 7—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 6—PA11 Mode (PA11MD): Selects the function of the PA11/$\overline{CS1}$ pin.

Bit 6: PA11MD	Description	
0	General input/output (PA11)	(Initial value)
1	Chip select output ($\overline{CS1}$)	

Bit 5—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 4—PA10 Mode (PA10MD): Selects the function of the PA10/$\overline{CS0}$ pin.

Bit 4: PA10MD	Description	
0	General input/output (PA10)	(Initial value)
1	Chip select output ($\overline{CS0}$)	

Bit 3—PA9 Mode 1 (PA9MD1): This bit selects the function of the PA9/$\overline{IRQ3}$ pin.

Bit 3: PA9MD1	Description	
0	General input/output (PA9)	(Initial value)
1	Interrupt request input ($\overline{IRQ3}$)	

Bit 2—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 1—PA8 Mode 1 (PA8MD1): This bit selects the function of the PA8/$\overline{IRQ2}$ pin.

Bit 1: PA8MD1	Description	
0	General input/output (PA8)	(Initial value)
1	Interrupt request input ($\overline{IRQ2}$)	

Bit 0—Reserved: This bit is always read as 0. The write value should always be 0.

Port A Control Register L2 (PACRL2)

Bit:	15	14	13	12	11	10	9	8
	PA7MD1	—	PA6MD1	—	—	PA5MD0	—	PA4MD
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R/W	R	R	R/W	R	R/W

Bit:	7	6	5	4	3	2	1	0
	—	PA3MD	PA2MD1	PA2MD0	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R

Bit 15—PA7 Mode 1 (PA7MD1): This bit selects the function of the PA7/$\overline{CS3}$ pin.

Bit 15: PA7MD1	Description	
0	General input/output (PA7)	(Initial value)
1	Chip select output ($\overline{CS3}$)	

Bit 14—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 13—PA6 Mode 1 (PA6MD1): This bit selects the function of the PA6/$\overline{CS2}$ pin.

Bit 13: PA6MD1	Description	
0	General input/output (PA6)	(Initial value)
1	Chip select output ($\overline{CS2}$)	

Bit 12—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 10—PA5 Mode 0 (PA5MD0): Selects the function of the PA5/SCK pin.

Bit 10: PA5MD0	Description	
0	General input/output (PA5)	(Initial value)
1	Serial clock input/output (SCK1)	

Bits 11 and 9—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 8—PA4 Mode (PA4MD): Selects the function of the PA4/TXD pin.

Bit 8: PA4MD	Description	
0	General input/output (PA4)	(Initial value)
1	Transmit data output (TXD)	

Bit 7—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 6—PA3 Mode (PA3MD): Selects the function of the PA3/RXD pin.

Bit 6: PA3MD	Description	
0	General input/output (PA3)	(Initial value)
1	Receive data input (RXD)	

Bits 5 and 4—PA2 Mode 1 and 0 (PA2MD1, PA2MD0): These bits select the function of the PA2/$\overline{\text{IRQ0}}$ pin.

Bit 5: PA2MD1	Bit 4: PA2MD0	Description	
0	0	General input/output (PA2)	(Initial value)
	1	Reserved	
1	0	Reserved	
	1	Interrupt request input ($\overline{\text{IRQ0}}$)	

Bits 3 to 0—Reserved: These bits are always read as 0. The write value should always be 0.

14.3.3 Port B IO Register (PBIOR)

The port B IO register (PBIOR) is a 16-bit readable/writable register that selects the input/output direction of the pins in port B. The bits of this register correspond to the various pins. PBIOR is enabled when the port B pins function as general input/output (PB9 to PB0) pins, and disabled otherwise.

When the port B pins function as PB9 to PB0, a pin becomes an output when the corresponding bit in PBIOR is set to 1, and an input when the bit is cleared to 0.

PBIOR is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases it retains its previous data.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	PB9IOR	PB8IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PB7IOR	PB6IOR	PB5IOR	PB4IOR	PB3IOR	PB2IOR	PB1IOR	PB0IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

14.3.4 Port B Control Registers 1 and 2 (PVCR1, PBCR2)

Port B control registers 1 and 2 (PVCR1, PBCR2) are 16-bit readable/writable registers that select the functions of the pins in port B.

PVCR1 and PBCR2 are each initialized to H'0000 by an external power-on reset. However, they are not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases they retain their previous data.

Port B Control Register 1 (PBCR1)

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	PB9MD1	PB9MD0	PB8MD1	PB8MD0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

Bits 15 to 4—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 3 and 2—PB9 Mode 1 and 0 (PB9MD1, PB9MD0): These bits select the function of the PB9/$\overline{\text{IRQ7}}$/A21 pin.

Bit 3: PB9MD1	Bit 2: PB9MD0	Description	
0	0	General input/output (PB9)	(Initial value)
	1	Interrupt request input ($\overline{\text{IRQ7}}$)	
1	0	Address output (A21)	
	1	Reserved	

Bits 1 and 0—PB8 Mode 1 and 0 (PB8MD1, PB8MD0): These bits select the function of the PB8/$\overline{\text{IRQ6}}$/A20/$\overline{\text{WAIT}}$ pin.

Bit 1: PB8MD1	Bit 0: PB8MD0	Description	
0	0	General input/output (PB8)	(Initial value)
	1	Interrupt request input ($\overline{\text{IRQ6}}$)	
1	0	Address output (A20)	
	1	Wait state request input ($\overline{\text{WAIT}}$)	

Port B Control Register 2 (PBCR2)

Bit:	15	14	13	12	11	10	9	8
	PB7MD1	PB7MD0	PB6MD1	PB6MD0	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	PB3MD0	—	—	—	PB1MD	—	PB0MD
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R	R	R	R/W	R	R/W

Bits 15 and 14—PB7 Mode 1 and 0 (PB7MD1, PB7MD0): These bits select the function of the PB7/A19 pin.

Bit 15: PB7MD1	Bit 14: PB7MD0	Description	
0	0	General input/output (PB7)	(Initial value)
	1	Reserved	
1	0	Address output (A19)	
	1	Reserved	

Bits 13 and 12—PB6 Mode 1 and 0 (PB6MD1, PB6MD0): These bits select the function of the PB6/A18 pin.

Bit 13: PB6MD1	Bit 12: PB6MD0	Description	
0	0	General input/output (PB6)	(Initial value)
	1	Reserved	
1	0	Address output (A18)	
	1	Reserved	

Bits 11 to 7—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 6—PB3 Mode 0 (PB3MD0): Selects the function of the PB3/$\overline{\text{IRQ1}}$ pin.

Bit 6: PB3MD0	Description	
0	General input/output (PB3)	(Initial value)
1	Interrupt request input ($\overline{\text{IRQ1}}$)	

Bits 5 to 3—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 2—PB1 Mode (PB1MD): Selects the function of the PB1/A17 pin.

Bit 2: PB1MD	Description	
0	General input/output (PB1)	(Initial value)
1	Address output (A17)	

Bit 1—Reserved: This bit is always read as 0 and should only be written with 0.

Bit 0—PB0 Mode (PB0MD): Selects the function of the PB0/A16 pin.

Bit 0: PB0MD	Description	
0	General input/output (PB0)	(Initial value)
1	Address output (A16)	

14.3.5 Port C IO Register (PCIOR)

The port C IO register (PCIOR) is a 16-bit readable/writable register that selects the input/output direction of the pins in port C. The bits of this register correspond to the various pins. PCIOR is enabled when the port C pins function as general input/output (PC15 to PC0) pins, and disabled otherwise.

When the port C pins function as PC15 to PC0, a pin becomes an output when the corresponding bit in PCIOR is set to 1, and an input when the bit is cleared to 0.

PCIOR is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases it retains its previous data.

Bit:	15	14	13	12	11	10	9	8
	PC15IOR	PC14IOR	PC13IOR	PC12IOR	PC11IOR	PC10IOR	PC9IOR	PC8IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PC7IOR	PC6IOR	PC5IOR	PC4IOR	PC3IOR	PC2IOR	PC1IOR	PC0IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

14.3.6 Port C Control Register (PCCR)

The port C control register (PCCR) is a 16-bit readable/writable register that selects the functions of the pins in port C.

PCCR is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases they retain their previous data.

Bit:	15	14	13	12	11	10	9	8
	PC15MD	PC14MD	PC13MD	PC12MD	PC11MD	PC10MD	PC9MD	PC8MD
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PC7MD	PC6MD	PC5MD	PC4MD	PC3MD	PC2MD	PC1MD	PC0MD
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 15—PC15 Mode (PC15MD): Selects the function of the PC15/A15 pin.

Bit 15: PC15MD	Description	
0	General input/output (PC15)	(Initial value)
1	Address output (A15)	

Bit 14—PC14 Mode (PC14MD): Selects the function of the PC14/A14 pin.

Bit 14: PC14MD	Description	
0	General input/output (PC14)	(Initial value)
1	Address output (A14)	

Bit 13—PC13 Mode (PC13MD): Selects the function of the PC13/A13 pin.

Bit 13: PC13MD	Description	
0	General input/output (PC13)	(Initial value)
1	Address output (A13)	

Bit 12—PC12 Mode (PC12MD): Selects the function of the PC12/A12 pin.

Bit 12: PC12MD	Description	
0	General input/output (PC12)	(Initial value)
1	Address output (A12)	

Bit 11—PC11 Mode (PC11MD): Selects the function of the PC11/A11 pin.

Bit 11: PC11MD	Description	
0	General input/output (PC11)	(Initial value)
1	Address output (A11)	

Bit 10—PC10 Mode (PC10MD): Selects the function of the PC10/A10 pin.

Bit 10: PC10MD	Description	
0	General input/output (PC10)	(Initial value)
1	Address output (A10)	

Bit 9—PC9 Mode (PC9MD): Selects the function of the PC9/A9 pin.

Bit 9: PC9MD	Description	
0	General input/output (PC9)	(Initial value)
1	Address output (A9)	

Bit 8—PC8 Mode (PC8MD): Selects the function of the PC8/A8 pin.

Bit 8: PC8MD	Description	
0	General input/output (PC8)	(Initial value)
1	Address output (A8)	

Bit 7—PC7 Mode (PC7MD): Selects the function of the PC7/A7 pin.

Bit 7: PC7MD	Description	
0	General input/output (PC7)	(Initial value)
1	Address output (A7)	

Bit 6—PC6 Mode (PC6MD): Selects the function of the PC6/A6 pin.

Bit 6: PC6MD	Description	
0	General input/output (PC6)	(Initial value)
1	Address output (A6)	

Bit 5—PC5 Mode (PC5MD): Selects the function of the PC5/A5 pin.

Bit 5: PC5MD	Description	
0	General input/output (PC5)	(Initial value)
1	Address output (A5)	

Bit 4—PC4 Mode (PC4MD): Selects the function of the PC4/A4 pin.

Bit 4: PC4MD	Description	
0	General input/output (PC4)	(Initial value)
1	Address output (A4)	

Bit 3—PC3 Mode (PC3MD): Selects the function of the PC3/A3 pin.

Bit 3: PC3MD	Description	
0	General input/output (PC3)	(Initial value)
1	Address output (A3)	

Bit 2—PC2 Mode (PC2MD): Selects the function of the PC2/A2 pin.

Bit 2: PC2MD	Description	
0	General input/output (PC2)	(Initial value)
1	Address output (A2)	

Bit 1—PC1 Mode (PC1MD): Selects the function of the PC1/A1 pin.

Bit 1: PC1MD	Description	
0	General input/output (PC1)	(Initial value)
1	Address output (A1)	

Bit 0—PC0 Mode (PC0MD): Selects the function of the PC0/A0 pin.

Bit 0: PC0MD	Description	
0	General input/output (PC0)	(Initial value)
1	Address output (A0)	

14.3.7 Port D IO Register L (PDIORL)

Port D IO register L (PDIORL) is a 16-bit readable/writable register that selects the input/output direction of the pins in port D. The bits of this register correspond to the various pins. PDIORL is enabled when the port D pins function as general input/output (PD7 to PD0), and disabled otherwise.

When the port D pins function as PD7 to PD0, a pin becomes an output when the corresponding bit in PDIORL is set to 1, and an input when the bit is cleared to 0.

PDIORL is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases it retains its previous data.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	PD7IOR	PD6IOR	PD5IOR	PD4IOR	PD3IOR	PD2IOR	PD1IOR	PD0IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

14.3.8 Port D Control Register L (PDCRL)

Port D control register L (PDCRL) is a 16-bit readable/writable register that selects the functions of the pins in port D.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	PD7MD	PD6MD	PD5MD	PD4MD	PD3MD	PD2MD	PD1MD	PD0MD
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- On-Chip ROM Enabled Extended Mode
  Port D pins function as both data I/O pins and general I/O pins. PDCRL settings are enabled.

PDCRL is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases they retain their previous data.

Bits 15 to 8—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 7—PD7 Mode (PD7MD): Selects the function of the PD7/D7 pin.

Bit 7: PD7MD	Description	
0	General input/output (PD7)	(Initial value)
1	Data input/output (D7)	

Bit 6—PD6 Mode (PD6MD): Selects the function of the PD6/D6 pin.

Bit 6: PD6MD	Description	
0	General input/output (PD6)	(Initial value)
1	Data input/output (D6)	

Bit 5—PD5 Mode (PD5MD): Selects the function of the PD5/D5 pin.

Bit 5: PD5MD	Description	
0	General input/output (PD5)	(Initial value)
1	Data input/output (D5)	

Bit 4—PD4 Mode (PD4MD): Selects the function of the PD4/D4 pin.

Bit 4: PD4MD	Description	
0	General input/output (PD4)	(Initial value)
1	Data input/output (D4)	

Bit 3—PD3 Mode (PD3MD): Selects the function of the PD3/D3 pin.

Bit 3: PD3MD	Description	
0	General input/output (PD3)	(Initial value)
1	Data input/output (D3)	

Bit 2—PD2 Mode (PD2MD): Selects the function of the PD2/D2 pin.

Bit 2: PD2MD	Description	
0	General input/output (PD2)	(Initial value)
1	Data input/output (D2)	

Bit 1—PD1 Mode (PD1MD): Selects the function of the PD1/D1 pin.

Bit 1: PD1MD	Description	
0	General input/output (PD1)	(Initial value)
1	Data input/output (D1)	

Bit 0—PD0 Mode (PD0MD): Selects the function of the PD0/D0 pin.

Bit 0: PD0MD	Description	
0	General input/output (PD0)	(Initial value)
1	Data input/output (D0)	

14.3.9 Port E IO Register (PEIOR)

The port E IO register (PEIOR) is a 16-bit readable/writable register that selects the input/output direction of the pins in port E. The bits of this register correspond to the various pins. PEIOR is enabled when the port E pins function as general input/output (PE14 to PE4, PE2, and PE0) pins or TIOC pins for the MTU, and disabled otherwise.

When the port E pins function as PE14 to PE4, PE2, and PE0 pins or TIOC pins for the MTU, a pin becomes an output when the corresponding bit in PEIOR is set to 1, and an input when the bit is cleared to 0.

PEIOR is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases it retains its previous data.

Bit:	15	14	13	12	11	10	9	8
	—	PE14IOR	PE13IOR	PE12IOR	PE11IOR	PE10IOR	PE9IOR	PE8IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PE7IOR	PE6IOR	PE5IOR	PE4IOR	—	PE2IOR	—	PE0IOR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R/W	R	R/W

14.3.10 Port E Control Register 2 (PECR2)

Port E control register 2 (PECR2) is a 16-bit readable/writable register that selects the functions of the pins in port E.

PECR2 is initialized to H'0000 by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode. In these cases it retains its previous data.

Bit:	15	14	13	12	11	10	9	8
	—	PE7MD	—	PE6MD	—	PE5MD	—	PE4MD
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R	R/W	R	R/W	R	R/W

Bit:	7	6	5	4	3	2	1	0
	—	—	—	PE2MD0	—	—	—	PE0MD0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R/W

Bit 15—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 14—PE7 Mode (PE7MD): Selects the function of the PE7/TIOC2B pin.

Bit 14: PE7MD	Description	
0	General input/output (PE7)	(Initial value)
1	MTU input capture input/output compare output (TIOC2B)	

Bit 13—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 12—PE6 Mode (PE6MD): Selects the function of the PE6/TIOC2A pin.

Bit 12: PE6MD	Description	
0	General input/output (PE6)	(Initial value)
1	MTU input capture input/output compare output (TIOC2A)	

Bit 11—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 10—PE5 Mode (PE5MD): Selects the function of the PE5/TIOC1B pin.

Bit 10: PE5MD	Description	
0	General input/output (PE5)	(Initial value)
1	MTU input capture input/output compare output (TIOC1B)	

Bit 9—Reserved: This bit is always read as 0. The write value should always be 0.

Bit 8—PE4 Mode (PE4MD): Selects the function of the PE4/TIOC1A pin.

Bit 8: PE4MD	Description	
0	General input/output (PE4)	(Initial value)
1	MTU input capture input/output compare output (TIOC1A)	

Bits 7 to 5—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 4—PE2 Mode 0 (PE2MD0): Selects the function of the PE2/TIOC0C pin.

Bit 4: PE2MD0	Description	
0	General input/output (PE2)	(Initial value)
1	MTU input capture input/output compare output (TIOC0C)	

Bits 3 to 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 0—PE0 Mode 0 (PE0MD0): Selects the function of the PE0/TIOC0A pin.

Bit 0: PE0MD0	Description	
0	General input/output (PE0)	(Initial value)
1	MTU input capture input/output compare output (TIOC0A)	

Section 15 I/O Ports (I/O)

15.1 Overview

All the port pins are multiplexed as general input/output pins (general input pins in the case of port F) and special function pins. The functions of the multiplex pins are selected by means of the pin function controller (PFC). Each port is provided with a data register for storing the pin data.

15.2 Port A

Port A is an input/output port with the 15 pins shown in figure 15.1.

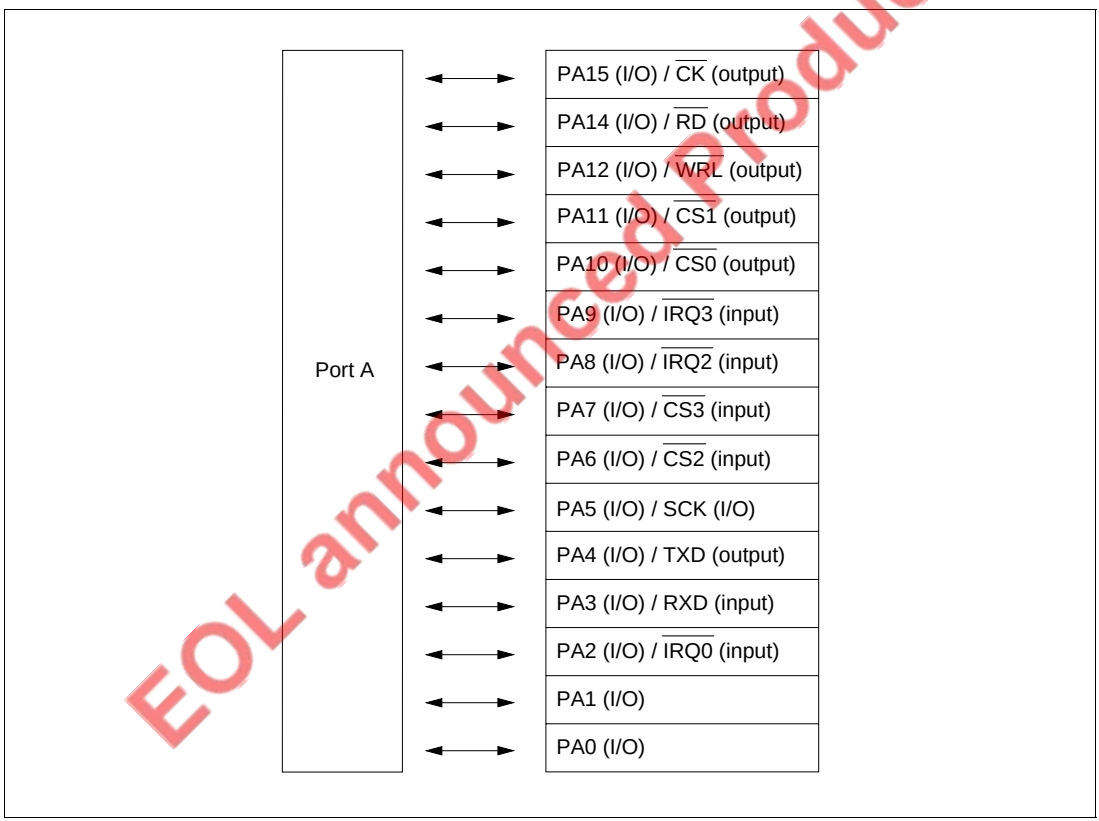


Figure 15.1 Port A

15.2.1 Register Configuration

The port A registers are shown in table 15.1.

Table 15.1 Port A Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port A Data Register L	PADRL	R/W	H'0000	H'FFFF8382 H'FFFF8383	8, 16, 32

15.2.2 Port A Data Register L (PADRL)

Port A data register L (PADRL) is a 16-bit readable/writable register that stores port A data. The bits of this register correspond to the various pins.

When a pin functions as a general output, if a value is written to PADRL, that value is output directly from the pin, and if PADRL is read, the register value is returned directly regardless of the pin state.

When a pin functions as a general input, if PADRL is read the pin state, not the register value, is returned directly. If a value is written to PADRL, that value is written to PADRL but it does not affect the pin state. Table 15.2 summarizes the port A data register read/write operations.

PADRL is initialized by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode.

Bit:	15	14	13	12	11	10	9	8
	PA15DR	PA14DR	—	PA12DR	PA11DR	PA10DR	PA9DR	PA8DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15.2 Port A Data Register (PADR) Read/Write Operations

PAIOR	Pin Function	Read	Write
0	General input	Pin state	Value is written to PADR, but does not affect pin state
	Other than general input	Pin state	Value is written to PADR, but does not affect pin state
1	General output	PADR value	Write value is output from pin
	Other than general output	PADR value	Value is written to PADR, but does not affect pin state

15.3 Port B

Port A is an input/output port with the 10 pins shown in figure 15.2.

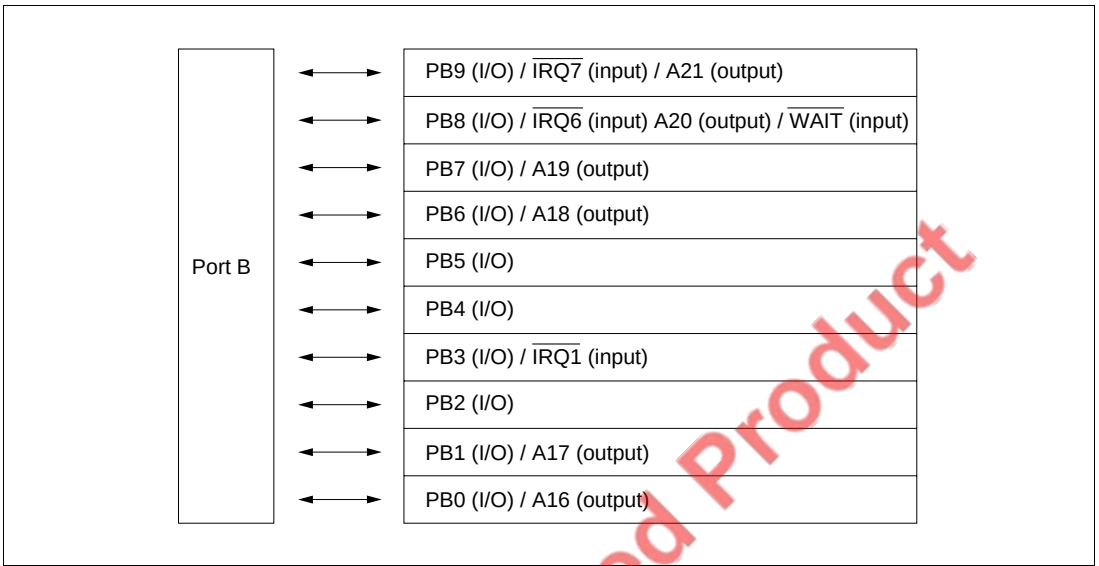


Figure 15.2 Port B

15.3.1 Register Configuration

The port B registers are shown in table 15.3.

Table 15.3 Port B Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port B Data Register	PBDR	R/W	H'0000	H'FFFF8390 H'FFFF8391	8, 16, 32

15.3.2 Port B Data Register (PBDR)

The port B data register (PBDR) is a 16-bit readable/writable register that stores port B data. The bits of this register correspond to the various pins.

When a pin functions as a general output, if a value is written to PBDR, that value is output directly from the pin, and if PBDR is read, the register value is returned directly regardless of the pin state.

When a pin functions as a general input, if PBDR is read the pin state, not the register value, is returned directly. If a value is written to PBDR, that value is written to PBDR but it does not affect the pin state. Table 15.4 summarizes the port B data register read/write operations.

PADR is initialized by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	PB9DR	PB8DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PB7DR	PB6DR	PB5DR	PB4DR	PB3DR	PB2DR	PB1DR	PB0DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15.4 Port B Data Register (PBDR) Read/Write Operations

PBIOR	Pin Function	Read	Write
0	General input	Pin state	Value is written to PBDR, but does not affect pin state
	Other than general input	Pin state	Value is written to PBDR, but does not affect pin state
1	General output	PBDR value	Write value is output from pin
	Other than general output	PBDR value	Value is written to PBDR, but does not affect pin state

15.4 Port C

Port C is an input/output port with the 16 pins shown in figure 15.3.

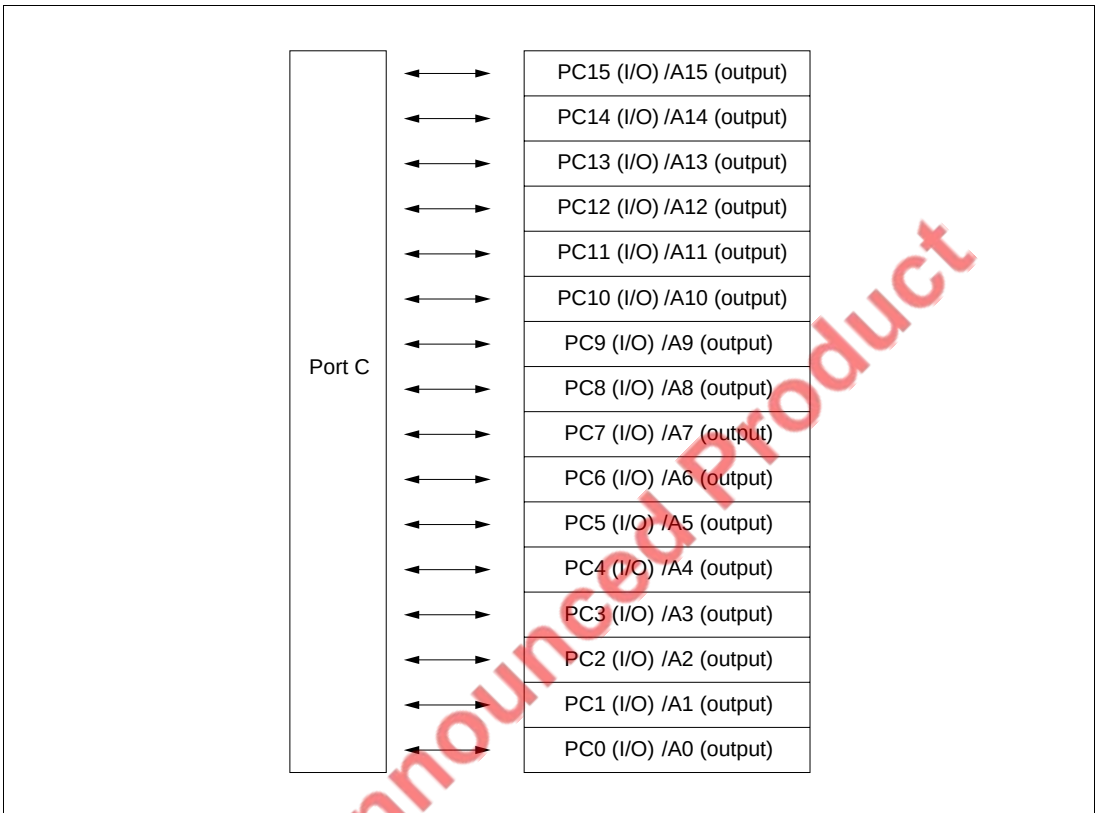


Figure 15.3 Port C

15.4.1 Register Configuration

The port C registers are shown in table 15.5.

Table 15.5 Port C Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port C Data Register	PCDR	R/W	H'0000	H'FFFF8392 H'FFFF8393	8, 16, 32

15.4.2 Port C Data Register (PCDR)

The port C data register (PCDR) is a 16-bit readable/writable register that stores port C data. The bits of this register correspond to the various pins.

When a pin functions as a general output, if a value is written to PCDR, that value is output directly from the pin, and if PCDR is read, the register value is returned directly regardless of the pin state.

When a pin functions as a general input, if PCDR is read the pin state, not the register value, is returned directly. If a value is written to PCDR, that value is written to PCDR but it does not affect the pin state. Table 15.6 summarizes the port C data register read/write operations.

PCDR is initialized by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode.

Bit:	15	14	13	12	11	10	9	8
	PC15DR	PC14DR	PC13DR	PC12DR	PC11DR	PC10DR	PC9DR	PC8DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PC7DR	PC6DR	PC5DR	PC4DR	PC3DR	PC2DR	PC1DR	PC0DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15.6 Port C Data Register (PCDR) Read/Write Operations

PCIOR	Pin Function	Read	Write
0	General input	Pin state	Value is written to PCDR, but does not affect pin state
	Other than general input	Pin state	Value is written to PCDR, but does not affect pin state
1	General output	PCDR value	Write value is output from pin
	Other than general output	PCDR value	Value is written to PCDR, but does not affect pin state

15.5 Port D

Port D is an input/output port with the eight pins shown in figure 15.4.

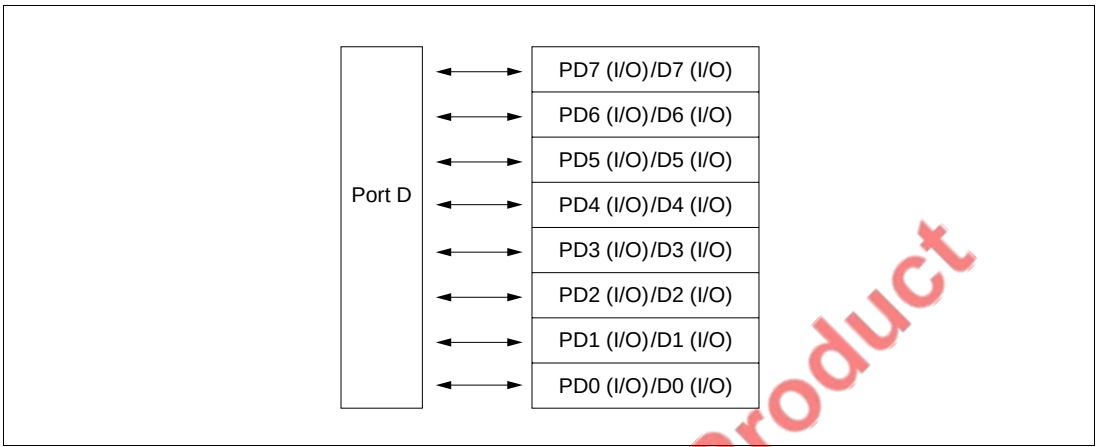


Figure 15.4 Port D

15.5.1 Register Configuration

The port D registers are shown in table 15.7.

Table 15.7 Port D Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port D Data Register L	PDDRL	R/W	H'0000	H'FFFF83A2 H'FFFF83A3	8, 16, 32

15.5.2 Port D Data Register L (PDDRL)

Port D data register L (PDDRL) is a 16-bit readable/writable register that stores port D data. The bits of this register correspond to the various pins.

When a pin functions as a general output, if a value is written to PDDRL, that value is output directly from the pin, and if PDDRL is read, the register value is returned directly regardless of the pin state.

When a pin functions as a general input, if PDDRL is read the pin state, not the register value, is returned directly. If a value is written to PDDRL, that value is written to PDDRL but it does not affect the pin state. Table 15.8 summarizes the port D data register read/write operations.

PDDRL is initialized by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15.8 Port D Data Register (PDDR) Read/Write Operations

PDIOR	Pin Function	Read	Write
0	General input	Pin state	Value is written to PDDR, but does not affect pin state
	Other than general input	Pin state	Value is written to PDDR, but does not affect pin state
1	General output	PDDR value	Write value is output from pin
	Other than general output	PDDR value	Value is written to PDDR, but does not affect pin state

15.6 Port E

Port E is an input/output port with the 13 pins shown in figure 15.5.

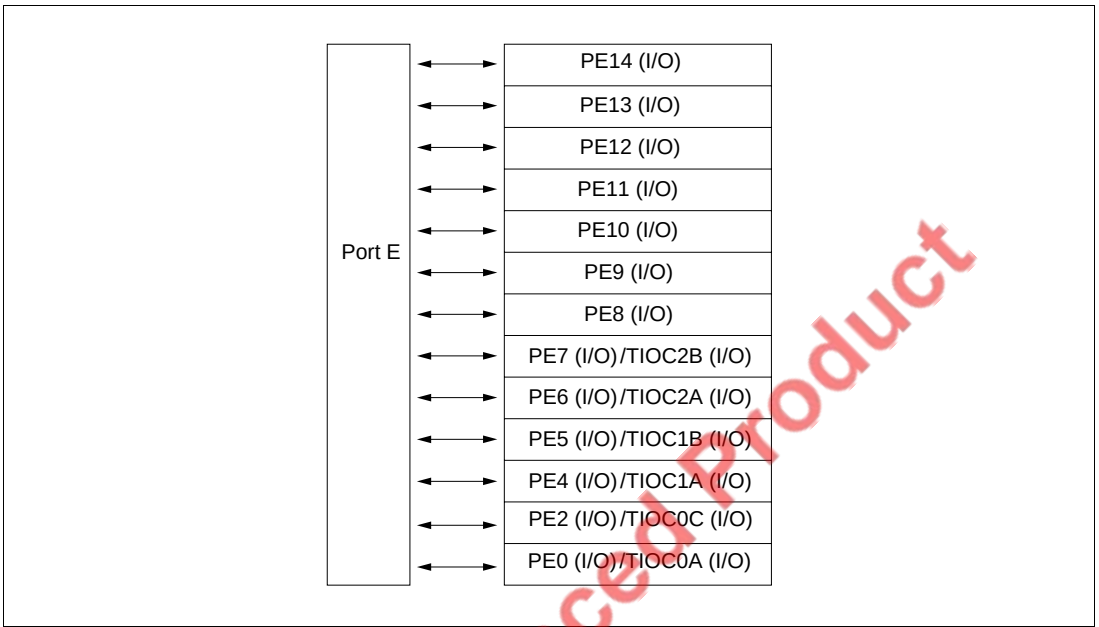


Figure 15.5 Port E

15.6.1 Register Configuration

The port E registers are shown in table 15.9.

Table 15.9 Port E Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port E Data Register	PEDR	R/W	H'0000	H'FFFF83B0 H'FFFF83B1	8, 16, 32

15.6.2 Port E Data Register (PEDR)

The port E data register (PEDR) is a 16-bit readable/writable register that stores port E data. The bits of this register correspond to the various pins. When a pin functions as a general output, if a value is written to PEDR, that value is output directly from the pin, and if PEDR is read, the register value is returned directly regardless of the pin state.

When a pin functions as a general input, if PEDR is read the pin state, not the register value, is returned directly. If a value is written to PEDR, that value is written to PEDR but it does not affect the pin state. Table 15.10 summarizes the port E data register read/write operations.

PEDR is initialized by an external power-on reset. However, it is not initialized by a WDT reset, in standby mode, or in sleep mode.

Bit:	15	14	13	12	11	10	9	8
	—	PE14DR	PE13DR	PE12DR	PE11DR	PE10DR	PE9DR	PE8DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PE7DR	PE6DR	PE5DR	PE4DR	—	PE2DR	—	PE0DR
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R/W	R	R/W

Table 15.10 Port E Data Register (PEDR) Read/Write Operations

PEIOR	Pin Function	Read	Write
0	General input	Pin state	Value is written to PEDR, but does not affect pin state
	Other than general input	Pin state	Value is written to PEDR, but does not affect pin state
1	General output	PEDR value	Write value is output from pin
	Other than general output	PEDR value	Value is written to PEDR, but does not affect pin state

15.7 Port F

Port F is an input port with the eight pins shown in figure 15.6.

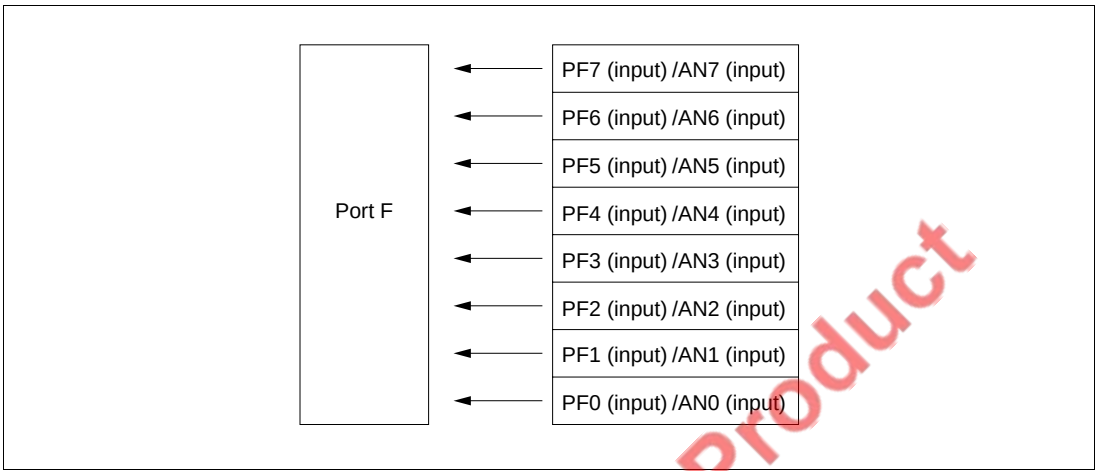


Figure 15.6 Port F

15.7.1 Register Configuration

The port F registers are shown in table 15.11.

Table 15.11 Port F Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port F Data Register	PFDR	R/W	Depends on external pins	H'FFFF83B3	8

15.7.2 Port E Data Register (PFDR)

The port F data register (PFDR) is an 8-bit read-only register that stores port F data. The bits of this register correspond to the various pins.

Writes to these bits are ignored, and do not affect the pin states. When these bits are read the pin state, not the register value, is returned directly. However, 1 is returned while A/D converter analog input is being sampled. Table 15.12 summarizes the port E data register read/write operations.

PFDR is not initialized by a power-on reset, in standby mode, or in sleep mode. (The bits always reflect the pin states.)

Bit:	7	6	5	4	3	2	1	0
	PF7DR	PF6DR	PF5DR	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Note: * These values depend on the pin state when the initial value is read.

Table 15.12 Port F Data Register (PFDR) Read/Write Operations

Pin Function	Pin state	Read	Write
Input	General input	Pin state is read	Ignored (does not affect pin state)
	ANn	1 is read	Ignored (does not affect pin state)

ANn: Analog input

Section 16 160 kB Flash Memory (F-ZTAT)

16.1 Features

The SH7018 has 160 kbytes of on-chip flash memory. The features of the flash memory are summarized below.

- Four flash memory operating modes
  - Program mode
  - Erase mode
  - Program-verify mode
  - Erase-verify mode
- Programming/erase methods

  The flash memory is programmed 128 bytes at a time. Block erase (in single-block units) can be performed. Erasing the entire memory requires erasure of each block in turn. Block erasing can be performed as required on 4 kB, 32 kB, and 64 kB blocks.
- Programming/erase times

  The flash memory programming time is 25 ms (typ.) for simultaneous 128-byte programming, equivalent to 195 µs (typ.) per byte, and the erase time is 10 ms (typ.).
- Reprogramming capability

  The flash memory can be reprogrammed up to 100 times.
- On-board programming modes

  There are two modes in which flash memory can be programmed/erased/verified on-board:
  - Boot mode
  - User program mode
- Automatic bit rate adjustment

  With data transfer in boot mode, the SH7018's bit rate can be automatically adjusted to match the transfer bit rate of the host.
- Flash memory emulation in RAM

  Flash memory programming can be emulated in real time by overlapping a part of RAM onto flash memory.
- Protect modes

  There are two protect modes, hardware and software, which allow protected status to be designated for flash memory program/erase/verify operations.
- Programmer mode

  Flash memory can be programmed/erased in programmer mode, using a PROM programmer, as well as in on-board programming mode.

16.2 Overview

16.2.1 Block Diagram

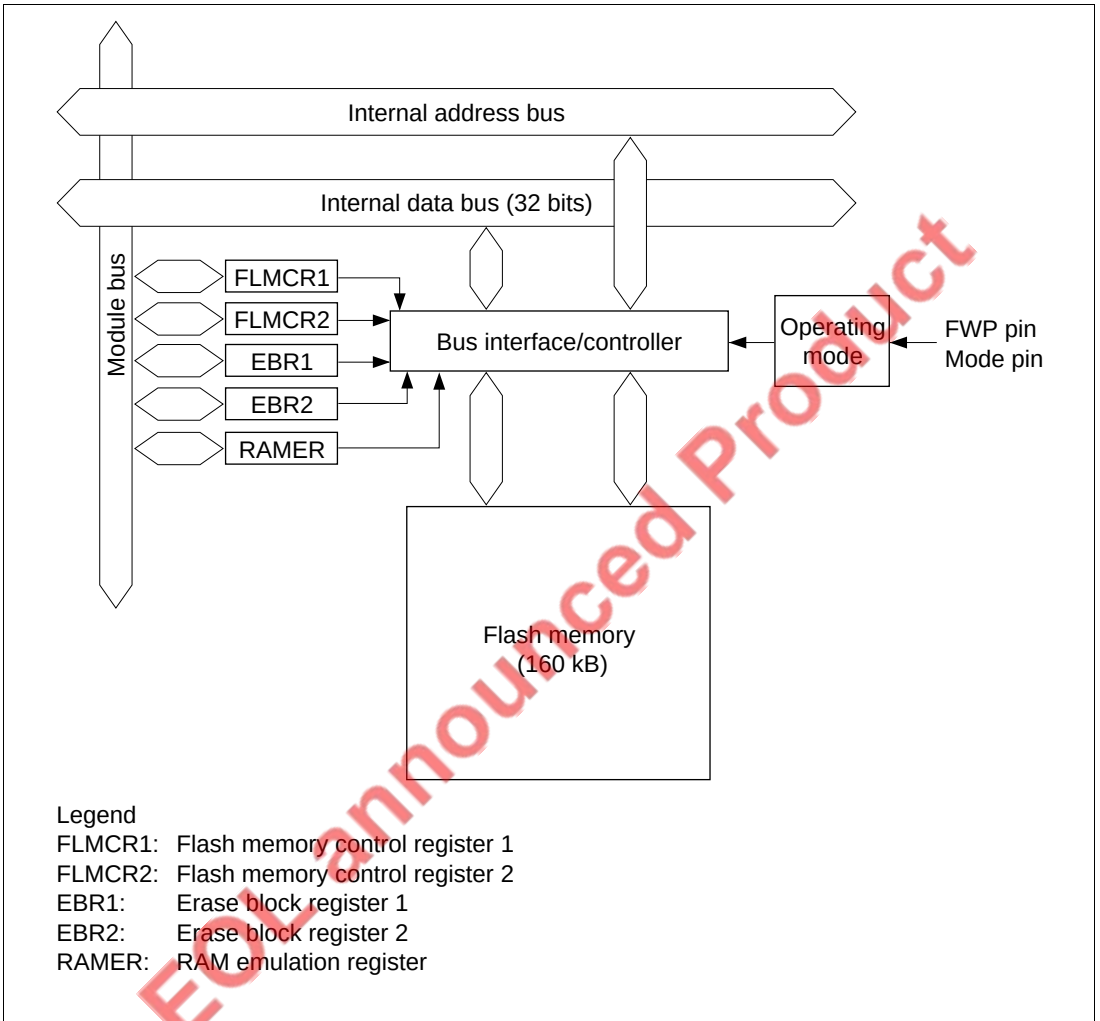
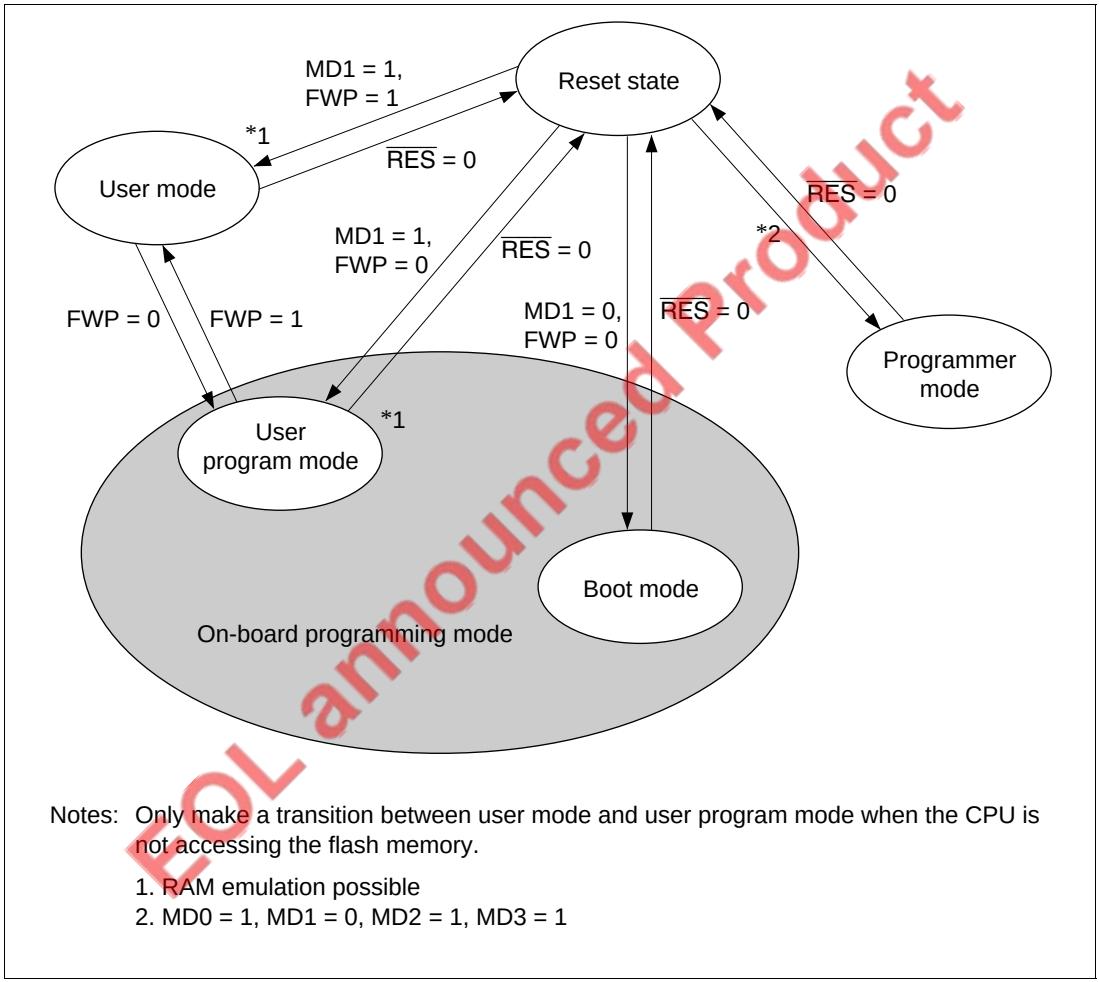


Figure 16.1 Block Diagram of Flash Memory

16.2.2 Mode Transitions

When the mode pins and the FWP pin are set in the reset state and a reset-start is executed, the SH7018 enters one of the operating modes shown in figure 16.2. In user mode, flash memory can be read but not programmed or erased.

Flash memory can be programmed and erased in boot mode, user program mode, and programmer mode.



Notes: Only make a transition between user mode and user program mode when the CPU is not accessing the flash memory.

1. RAM emulation possible
2. MD0 = 1, MD1 = 0, MD2 = 1, MD3 = 1

Figure 16.2 Flash Memory Mode Transitions

16.2.3 On-Board Programming Modes

Boot Mode

Figure 16.3 illustrates overwrite operation in the boot mode. For more information on the boot mode, see 16.6.1, Boot Mode.

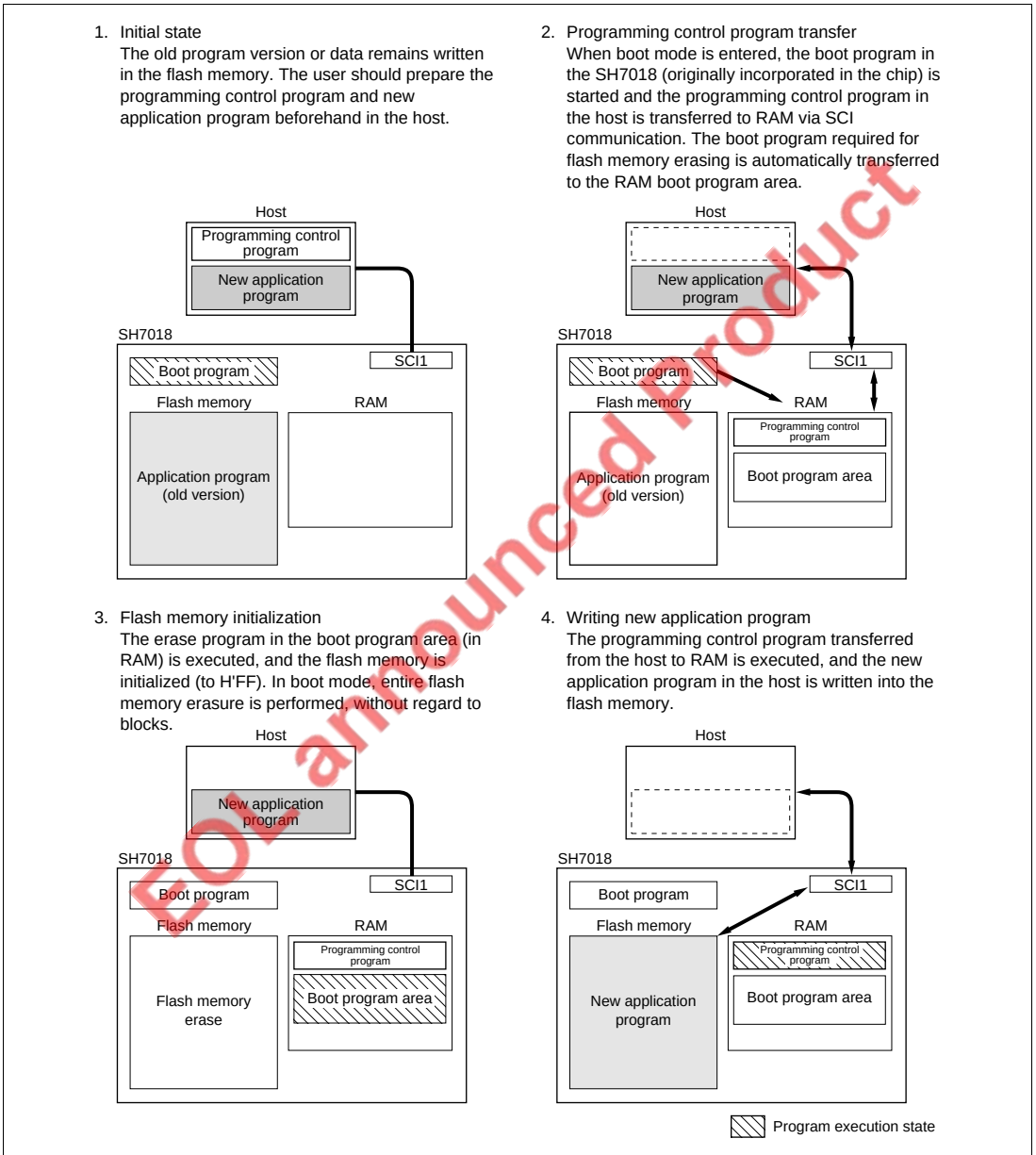


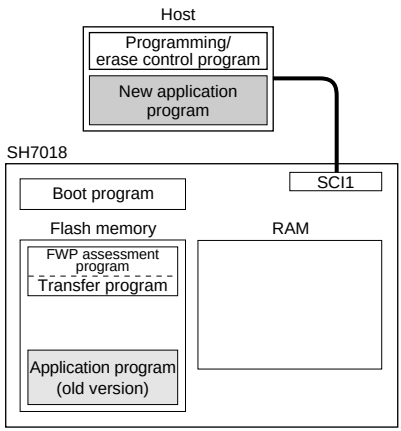
Figure 16.3 Overwrite Operation in the Boot Mode

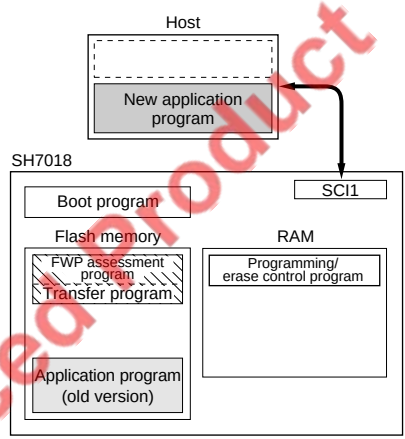
User Program Mode

Figure 16.4 illustrates overwrite operation in the user program mode. For more information on the user program mode, see 16.6.2, User Program Mode.

1. Initial state
   The FWP assessment program that confirms that user program mode has been entered, and the program that will transfer the programming/erase control program from flash memory to on-chip RAM should be written into the flash memory by the user beforehand. The programming/erase control program should be prepared in the host or in the flash memory.

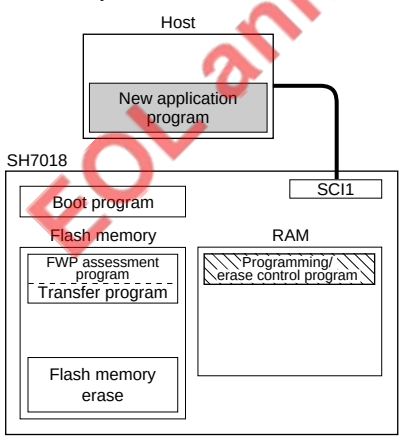
2. Programming/erase control program transfer
   When user program mode is entered, user software confirms this fact, executes transfer program in the flash memory, and transfers the programming/erase control program to RAM.





3. Flash memory initialization
   The programming/erase program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.

4. Writing new application program
   Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.



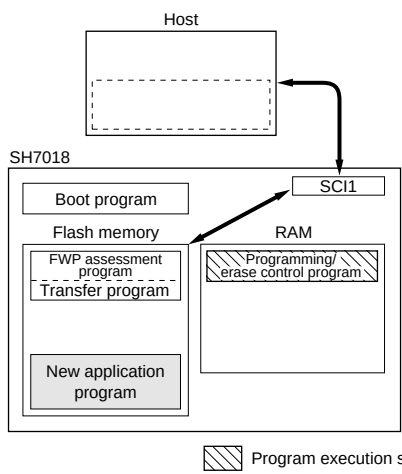


Figure 16.4 Example of Overwrite Operation in the User Program Mode

16.2.4 Flash Memory Emulation in RAM

Emulation should be performed in user mode or user program mode. When the emulation block set in RAMER is accessed while the emulation function is being executed, data written in the overlap RAM is read.

- User Mode
- User Program Mode

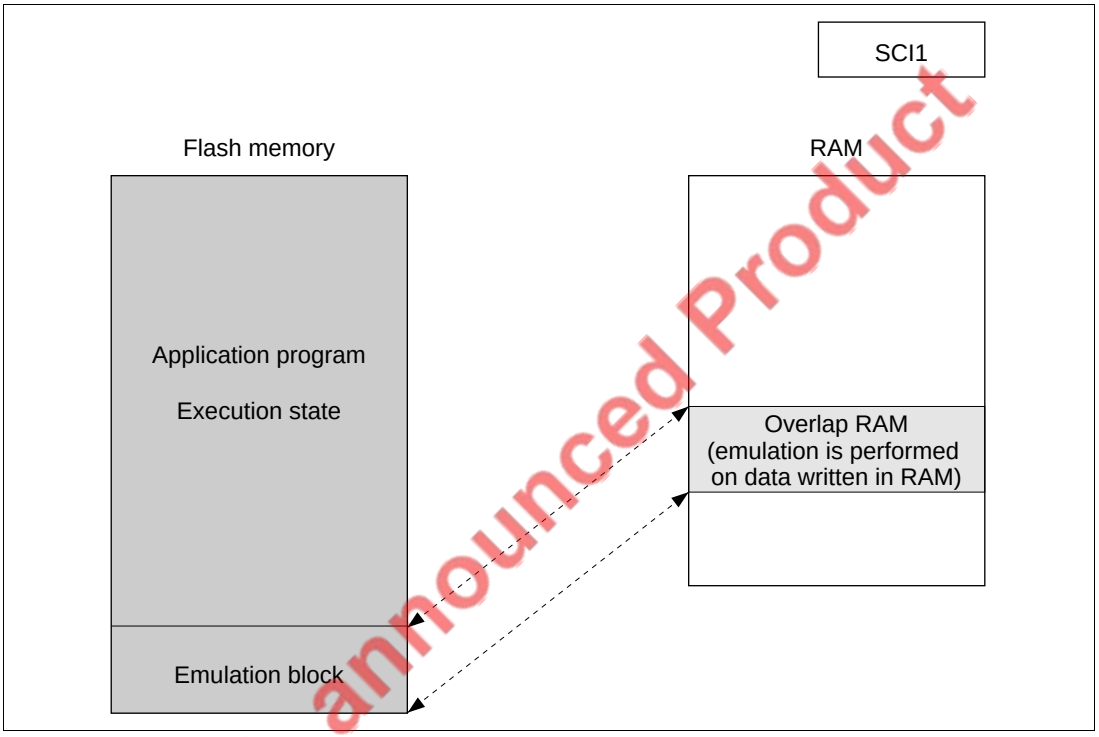


Figure 16.5 RAM Emulation (RAM Overlap)

When overlap RAM data is confirmed, the RAMS bit is cleared, RAM overlap is released, and writes should actually be performed to the flash memory.

When the programming control program is transferred to RAM, ensure that the transfer destination and the overlap RAM do not overlap, as this will cause data in the overlap RAM to be rewritten.

- User Program Mode

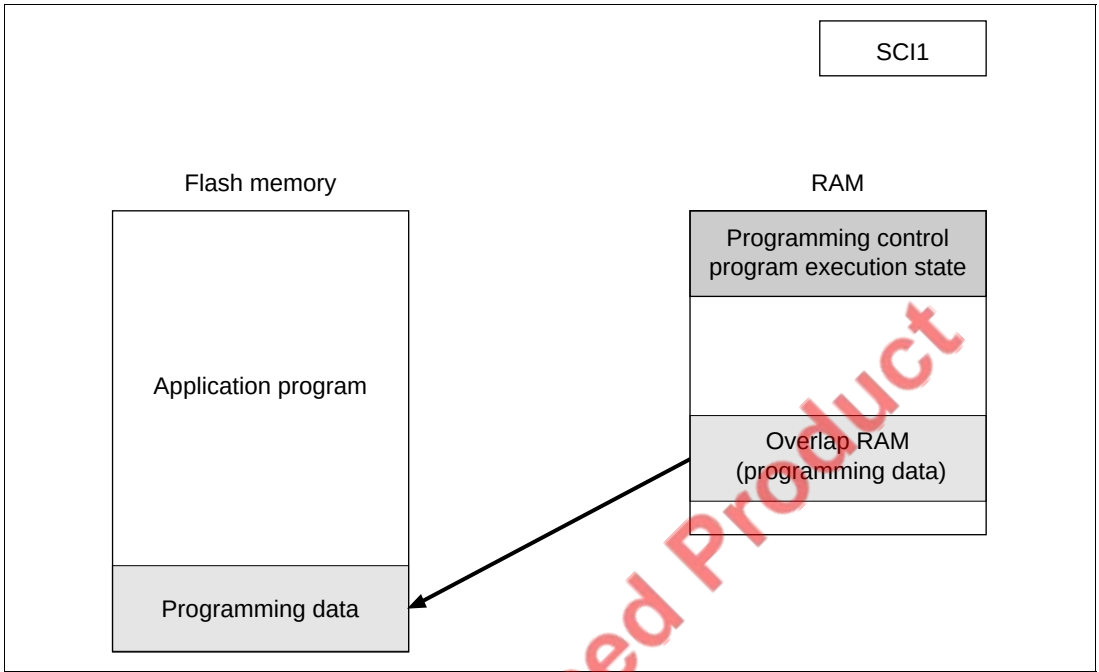


Figure 16.6 RAM Emulation (Flash Memory Overwrite)

16.2.5 Differences between Boot Mode and User Program Mode

	Boot Mode	User Program Mode
Entire memory erase	Yes	Yes
Block erase	No	Yes
Programming control program*	(2)	(1) (2) (3)

(1) Erase/erase-verify
(2) Program/program-verify
(3) Emulation

Note: * To be provided by the user, in accordance with the recommended algorithm.

16.2.6 Block Configuration

The flash memory is divided into eight 4 kB blocks, two 32 kB blocks, and one 64 kB block. The data in flash memory can be deleted one block at a time in the user program mode.

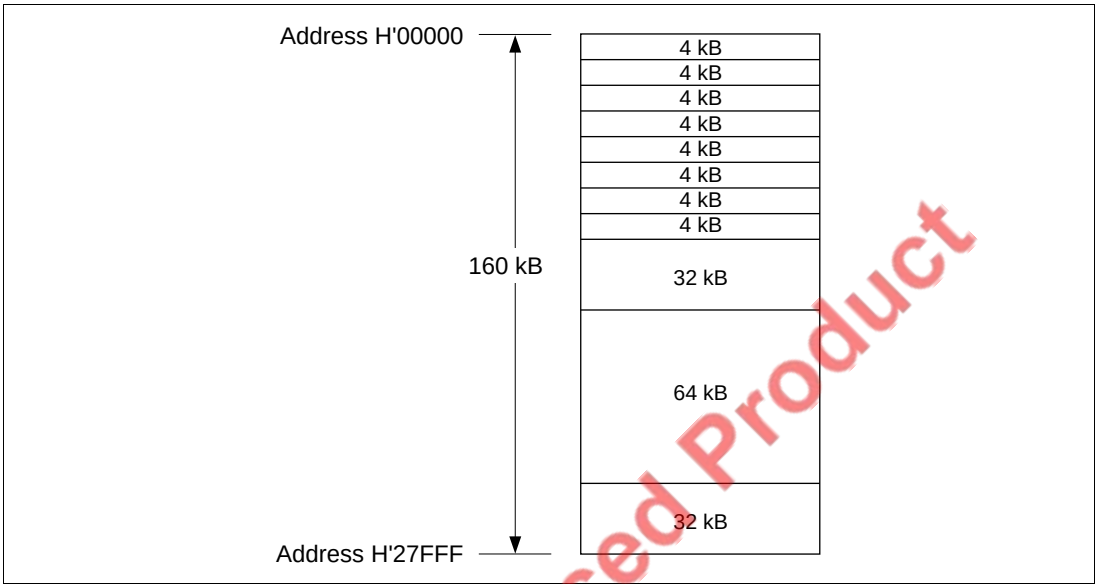


Figure 16.7 Block Configuration of Area to Be Deleted

16.3 Pin Configuration

The flash memory is controlled by means of the pins shown in table 16.1.

Table 16.1 Flash Memory Pins

Pin Name	Abbreviation	I/O	Function
Power-on reset	$\overline{\text{RES}}$	Input	Power-on reset
Flash memory write protect	FWP	Input	Flash program/erase protection by hardware
Mode 3	MD3	Input	Sets SH7018 operating mode
Mode 2	MD2	Input	Sets SH7018 operating mode
Mode 1	MD1	Input	Sets SH7018 operating mode
Mode 0	MD0	Input	Sets SH7018 operating mode
Transmit data	TxD	Output	Serial transmit data output
Receive data	RxD	Input	Serial receive data input

16.4 Register Configuration

The registers used to control the on-chip flash memory when enabled are shown in table 16.2.

Table 16.2 Flash Memory Registers

Register Name	Abbreviation	R/W	Initial Value	Address	Access Size
Flash memory control register 1	FLMCR1	R/W*[1]	H'00*[2]	H'FFFF8580	8
Flash memory control register 2	FLMCR2	R	H'00	H'FFFF8581	8
Erase block register 1	EBR1	R/W*[1]	H'00*[3]	H'FFFF8582	8
Erase block register 2	EBR2	R/W*[1]	H'00*[3]	H'FFFF8583	8
RAM emulation register	RAMER	R/W	H'0000	H'FFFF8628	8, 16, 32

Notes:
1. Writes are disabled when the FWE bit is set to 1 in FLMCR1.
2. When a low level is input to the FWP pin, the initial value is H'80.
3. When a high level is input to the FWP pin, or if a low level is input and the SWE bit in FLMCR1 is not set, these registers are initialized to H'00.
4. FLMCR1, FLMCR2, EBR1, and EBR2 are 8-bit registers, and RAMER is a 16-bit register.
5. Only byte accesses are valid for FLMCR1, FLMCR2, EBR1, and EBR2, the access requiring 3 cycles. Three cycles are required for a byte or word access to RAMER, and 6 cycles for a longword access.
6. When a longword write is performed on RAMER, 0 must always be written to the lower word (address H'FFFF8630). Operation is not guaranteed if any other value is written.

16.5 Register Descriptions

16.5.1 Flash Memory Control Register 1 (FLMCR1)

FLMCR1 is an 8-bit register used for flash memory operating mode control. Program-verify mode or erase-verify mode is entered by setting SWE to 1 when FWE = 1. Program mode is entered by setting SWE to 1 when FWE = 1, then setting the PSU bit, and finally setting the P bit. Erase mode is entered by setting SWE to 1 when FWE = 1, then setting the ESU bit, and finally setting the E bit. FLMCR1 is initialized by a power-on reset, and in standby mode. Its initial value is H'80 when a low level is input to the FWP pin, and H'00 when a high level is input. When on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

Writes to bits SWE, ESU, PSU, EV, and PV are enabled only when FWE = 1 and SWE = 1; writes to the E bit only when FWE = 1, SWE = 1, and ESU = 1; and writes to the P bit only when FWE = 1, SWE = 1, and PSU = 1.

Bit:	7	6	5	4	3	2	1	0
	FWE	SWE	ESU	PSU	EV	PV	E	P
Initial value:	1/0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- Bit 7—Flash Write Enable Bit (FWE): Sets hardware protection against flash memory programming/erasing.

Bit 7: FWE	Description	
0	When a high level is input to the FWP pin (hardware-protected state)	(Initial value)
1	When a low level is input to the FWP pin	

- Bit 6—Software Write Enable Bit (SWE): Enables or disables the flash memory. This bit should be set before setting bits 5 to 0, EBR1 bits 7 to 0, and EBR2 bits 2 to 0. The flash memory cannot be read when SWE is set to 1, except in the program verify/erase verify mode.

Bit 6: SWE	Description	
0	Writes disabled	(Initial value)
1	Writes enabled [Setting condition] When FWE = 1	

- Bit 5—Erase Setup Bit (ESU): Prepares for a transition to erase mode. Do not set the SWE, PSU, EV, PV, E, or P bit at the same time.

Bit 5: ESU	Description	
0	Erase setup cleared	(Initial value)
1	Erase setup [Setting condition] When FWE = 1 and SWE = 1	

- Bit 4—Program Setup Bit (PSU): Prepares for a transition to program mode. Do not set the SWE, ESU, EV, PV, E, or P bit at the same time.

Bit 4: PSU	Description	
0	Program setup cleared	(Initial value)
1	Program setup [Setting condition] When FWE = 1 and SWE = 1	

- Bit 3—Erase-Verify (EV): Selects erase-verify mode transition or clearing. Do not set the SWE, ESU, PSU, PV, E, or P bit at the same time.

Bit 3: EV	Description	
0	Erase-verify mode cleared	(Initial value)
1	Transition to erase-verify mode [Setting condition] When FWE = 1 and SWE = 1	

- Bit 2—Program-Verify (PV): Selects program-verify mode transition or clearing. Do not set the SWE, ESU, PSU, EV, E, or P bit at the same time.

Bit 2: PV	Description	
0	Program-verify mode cleared	(Initial value)
1	Transition to program-verify mode [Setting condition] When FWE = 1 and SWE = 1	

- Bit 1—Erase (E): Selects erase mode transition or clearing. Do not set the SWE, ESU, PSU, EV, PV, or P bit at the same time.

Bit 1:

E	Description	
0	Erase mode cleared	(Initial value)
1	Transition to erase mode [Setting condition] When FWE = 1, SWE = 1, and ESU1 = 1	

- Bit 0—Program (P): Selects program mode transition or clearing. Do not set the SWE, PSU, ESU, EV, PV, or E bit at the same time.

Bit 0:

P	Description	
0	Program mode cleared	(Initial value)
1	Transition to program mode [Setting condition] When FWE = 1, SWE = 1, and PSU = 1	

16.5.2 Flash Memory Control Register 2 (FLMCR2)

FLMCR2 is an 8-bit register that monitors the presence or absence of flash memory program/erase protection (error protection). FLMCR2 is initialized to H'00 by a power-on reset.

When on-chip flash memory is disabled, a read will return H'00.

Bit:	7	6	5	4	3	2	1	0
	FLER	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

- Bit 7—Flash Memory Error (FLER): Indicates that an error has occurred during an operation on flash memory (programming or erasing). When FLER is set to 1, flash memory goes to the error-protection state.

Bit 7: FLER	Description	
0	Flash memory is operating normally.	(Initial value)
	Flash memory program/erase protection (error protection) is disabled.	
	[Clearing condition]	
	Power-on reset	
1	An error has occurred during flash memory programming/erasing.	
	Flash memory program/erase protection (error protection) is enabled.	
	[Setting condition]	
	See section 16.8.3, Error Protection.	

- Bits 6 to 0—Reserved: These bits are always read as 0. The write value should always be 0.

16.5.3 Erase Block Register 1 (EBR1)

EBR1 is an 8-bit readable/writable register that specifies the flash memory erase area block by block. EBR1 is initialized to H'00 by a power-on reset, in standby mode, when a high level is input to the FWP pin, and when a low level is input to the FWP pin and the SWE bit in FLMCR1 is not set. When a bit in EBR1 is set to 1, the corresponding block can be erased. Other blocks are erase-protected. As with EBR2, set only one bit of EBR1 (more than one bit cannot be set). If two or more bits are set, writes to the ESU and E bits are invalid. When on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

The flash memory block configuration is shown in table 16.3.

Bit:	7	6	5	4	3	2	1	0
	EB7	EB6	EB5	EB4	EB3	EB2	EB1	EB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

16.5.4 Erase Block Register 2 (EBR2)

EBR2 is an 8-bit register that specifies the flash memory erase area block by block. EBR2 is initialized to H'00 by a power-on reset and in standby mode, when a high level is input to the FWP pin, and when a low level is input to the FWP pin and the SWE bit of FLMCR1 is not set. When a bit in EBR2 is set to 1, the corresponding block can be erased. The other blocks are erase-protected. If the on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

The flash memory block configuration is shown in table 16.3.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	EB10	EB9	EB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

- Bits 7 to 3—Reserved: These bits are always read as 0. The write value should always be 0.

Table 16.3 Flash Memory Erase Blocks

Block (Size)	Address
EB0 (4 kB)	H'000000 to H'000FFF
EB1 (4 kB)	H'001000 to H'001FFF
EB2 (4 kB)	H'002000 to H'002FFF
EB3 (4 kB)	H'003000 to H'003FFF
EB4 (4 kB)	H'004000 to H'004FFF
EB5 (4 kB)	H'005000 to H'005FFF
EB6 (4 kB)	H'006000 to H'006FFF
EB7 (4 kB)	H'007000 to H'007FFF
EB8 (32 kB)	H'008000 to H'00FFFF
EB9 (64 kB)	H'010000 to H'01FFFF
EB10 (32 kB)	H'020000 to H'027FFF

16.5.5 RAM Emulation Register (RAMER)

RAMER specifies the area of flash memory to be overlapped with part of RAM when emulating real-time flash memory programming. RAMER is initialized to H'0000 by a power-on reset. It is not initialized in standby mode. RAMER settings should be made in user mode or user program mode.

Flash memory area divisions are shown in table 16.4. To ensure correct operation of the emulation function, the ROM for which RAM emulation is performed should not be accessed immediately after this register has been modified. Normal execution of an access immediately after register modification is not guaranteed.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	RAMS	RAM1	RAM0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

- Bits 15 to 3—Reserved: These bits are always read as 0. The write value should always be 0.

- Bit 2—RAM Select (RAMS): Specifies selection or non-selection of flash memory emulation in RAM. When RAMS = 1, all flash memory block are program/erase-protected.

Bit 2: RAMS	Description	
0	Emulation not selected	(Initial value)
	Program/erase-protection of all flash memory blocks is disabled	
1	Emulation selected	
	Program/erase-protection of all flash memory blocks is enabled	

- Bits 1 and 0—Flash Memory Area Selection (RAM1, RAM0): These bits are used together with bit 2 to select the flash memory area to be overlapped with RAM. For each block, only the first 1 kB of addresses can be overlapped. (See table 16.4.)

Table 16.4 Flash Memory Area Divisions

Addresses	Block Name	RAMS	RAM1	RAM0
H'FFF800 to H'FFFBFF	RAM area 1 kB	0	*	*
H'004000 to H'0043FF	EB4 (1 kB)	1	0	0
H'005000 to H'0053FF	EB5 (1 kB)	1	0	1
H'006000 to H'0063FF	EB6 (1 kB)	1	1	0
H'007000 to H'0073FF	EB7 (1 kB)	1	1	1

16.6 On-Board Programming Modes

When pins are set to on-board programming mode and a power-on reset-start is executed, a transition is made to the on-board programming state in which program/erase/verify operations can be performed on the on-chip flash memory. There are two on-board programming modes: boot mode and user program mode. The pin settings for transition to each of these modes are shown in table 16.5. For a diagram of the transitions to the various flash memory modes, see figure 16.2.

Table 16.5 Setting On-Board Programming Modes

Mode	FWP	MD3	MD2	MD1	MD0
Boot mode	0	0	0	0	0
User program mode	0	0	0	1	0

16.6.1 Boot Mode

When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. The SCI1 to be used is set to channel asynchronous mode.

When a reset-start is executed after the SH7018 pins have been set to boot mode in the power-on reset state, the boot program built into the SH7018 is started and the programming control program prepared in the host is serially transmitted to the SH7018 via SCI1 (RxD, TxD). In the SH7018, the programming control program received via SCI1 is written into the programming control program area in on-chip RAM. After the transfer is completed, control branches to the start address of the programming control program area and the programming control program execution state is entered (flash memory programming is performed).

The transferred programming control program must therefore include coding that follows the programming algorithm given later.

The system configuration in boot mode is shown in figure 16.8, and the boot mode execution procedure in figure 16.9.

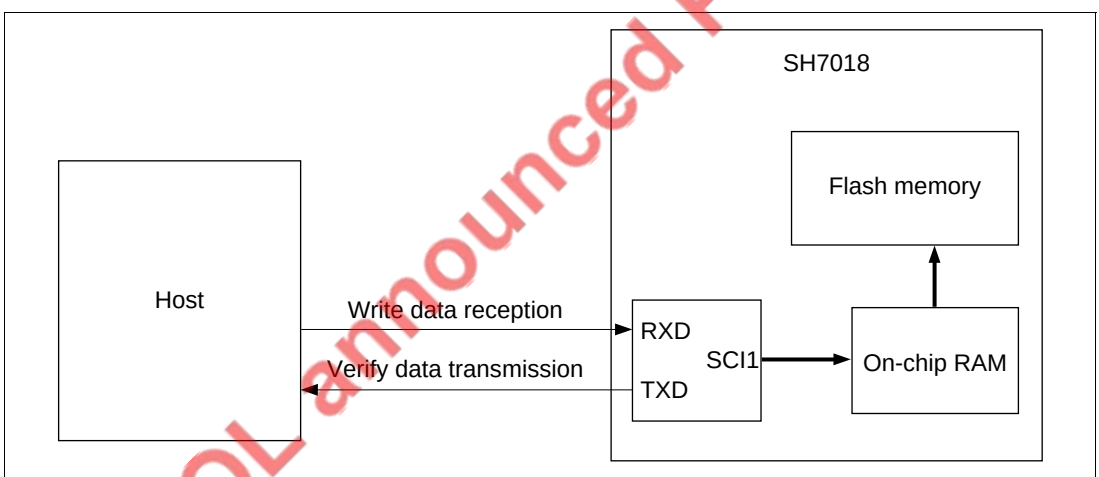
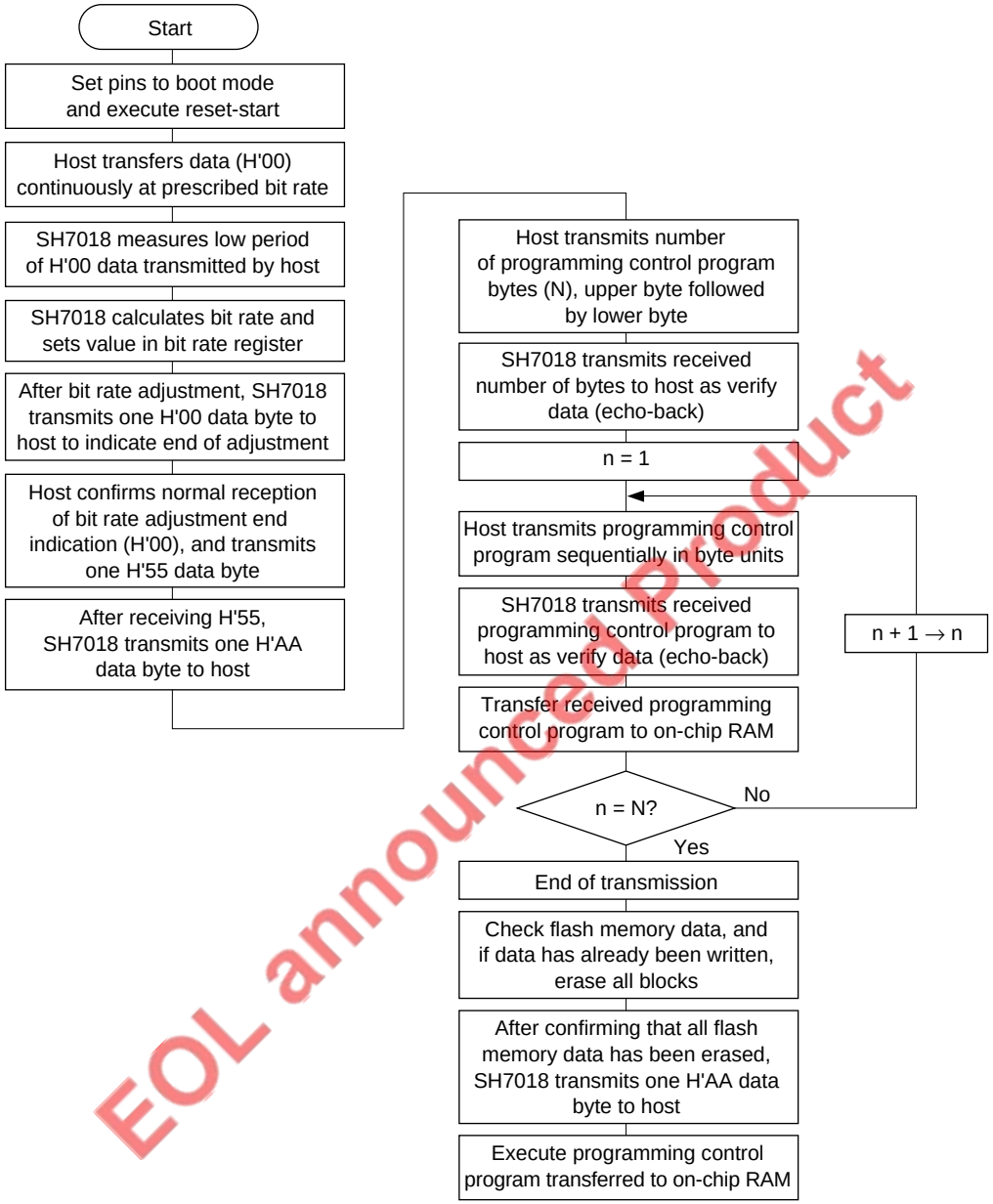


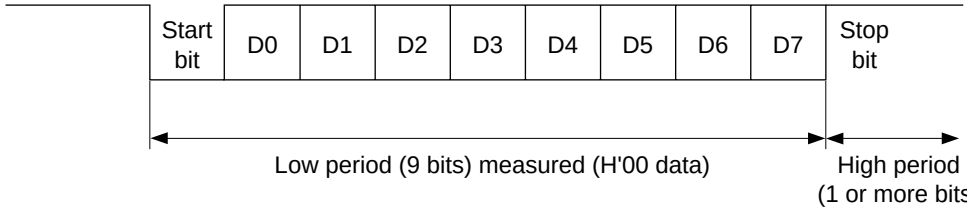
Figure 16.8 System Configuration in Boot Mode



Note: If a memory cell does not operate normally and cannot be erased, one H'FF byte is transmitted as an erase error, and the erase operation and subsequent operations are halted.

Figure 16.9 Boot Mode Execution Procedure

Automatic SCI Bit Rate Adjustment



When boot mode is initiated, the SH7018 measures the low period of the asynchronous SCI communication data (H'00) transmitted continuously from the host. The SCI transmit/receive format should be set as follows: 8-bit data, 1 stop bit, no parity. The SH7018 calculates the bit rate of the transmission from the host from the measured low period, and transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the SH7018. If reception cannot be performed normally, initiate boot mode again (power-on reset), and repeat the above operations. Depending on the host's transmission bit rate and the SH7018's system clock frequency, there will be a discrepancy between the bit rates of the host and the SH7018. To ensure correct SCI operation, the host's transfer bit rate should be set to 4800bps, 9600bps.

Table 16.6 shows host transfer bit rates and system clock frequencies for which automatic adjustment of the SH7018 bit rate is possible. The boot program should be executed within this system clock range.

Table 16.6 System Clock Frequencies for which Automatic Adjustment of SH7018 Bit Rate is Possible

Host Bit Rate	System Clock Frequency for which Automatic Adjustment of SH7018 Bit Rate is Possible
19200 bps	16 to 20 MHz
9600 bps	8 to 20 MHz
4800 bps	4 to 20 MHz

On-Chip RAM Area Divisions in Boot Mode: In boot mode, the RAM area is divided into an area used by the boot program and an area to which the programming control program is transferred via the SCI, as shown in figure 16.10. The boot program area cannot be used until the execution state in boot mode switches to the programming control program transferred from the host.

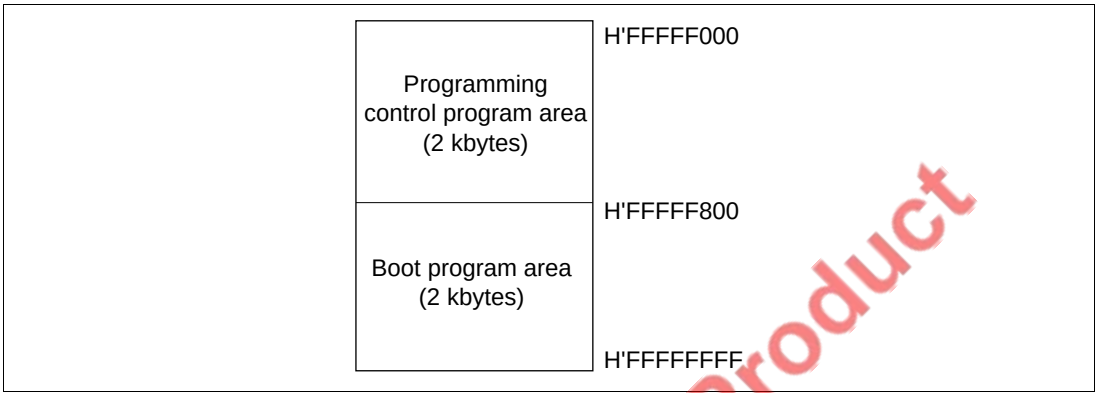


Figure 16.10 RAM Areas in Boot Mode

Note: The boot program area cannot be used until a transition is made to the execution state for the programming control program transferred to RAM. Note also that the boot program remains in this area of the on-chip RAM even after control branches to the programming control program.

16.6.2 User Program Mode

After setting FWP, the user should branch to, and execute, the previously prepared programming/erase control program.

As the flash memory itself cannot be read while flash memory programming/erasing is being executed, the control program that performs programming and erasing should be run in on-chip RAM or external memory.

Use the following procedure (figure 16.11) to execute the programming control program that writes to flash memory (when transferred to RAM).

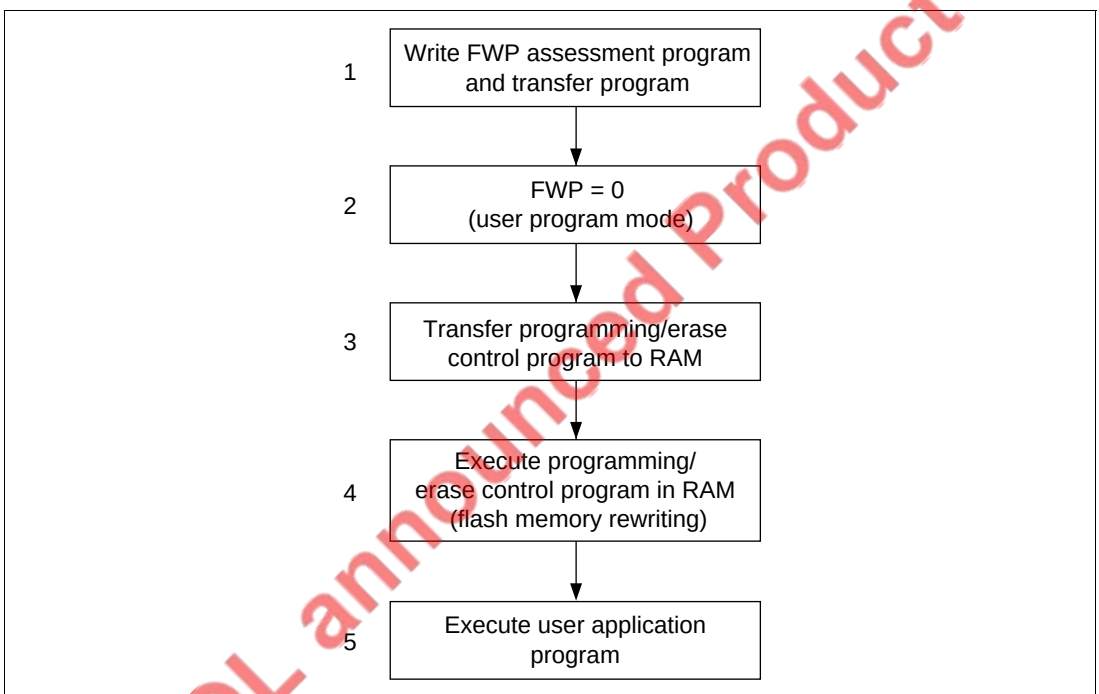


Figure 16.11 User Program Mode Execution Procedure

Note: When programming and erasing, start the watchdog timer so that measures can be taken to prevent program runaway, etc. Memory cells may not operate normally if overprogrammed or overerased due to program runaway.

16.7 Programming/Erasing Flash Memory

A software method, using the CPU, is employed to program and erase flash memory in the on-board programming modes. There are four flash memory operating modes: program mode, erase mode, program-verify mode, and erase-verify mode. Transitions to these modes can be made by setting the PSU, ESU, P, E, PV, and EV bits in FLMCR1.

The flash memory cannot be read while being programmed or erased. Therefore, the program (programming control program) that controls flash memory programming/erasing should be located and executed in on-chip RAM or external memory.

Notes: 1. Operation is not guaranteed if setting/resetting of the SWE, ESU, PSU, EV, PV, E, and P bits in FLMCR1 is executed by a program in flash memory.
2. When programming or erasing, a low level is input to the FWP pin (programming/erasing will not be executed if a high level is input to the FWP pin).
3. Programming should be performed in the erased state. Do not perform additional programming on previously programmed addresses.

16.7.1 Program Mode

Follow the procedure shown in the program/program-verify flowchart in figure 16.12 to write data or programs to flash memory. Performing program operations according to this flowchart will enable data or programs to be written to flash memory without subjecting the device to voltage stress or sacrificing program data reliability. Programming should be carried out 128 bytes at a time.

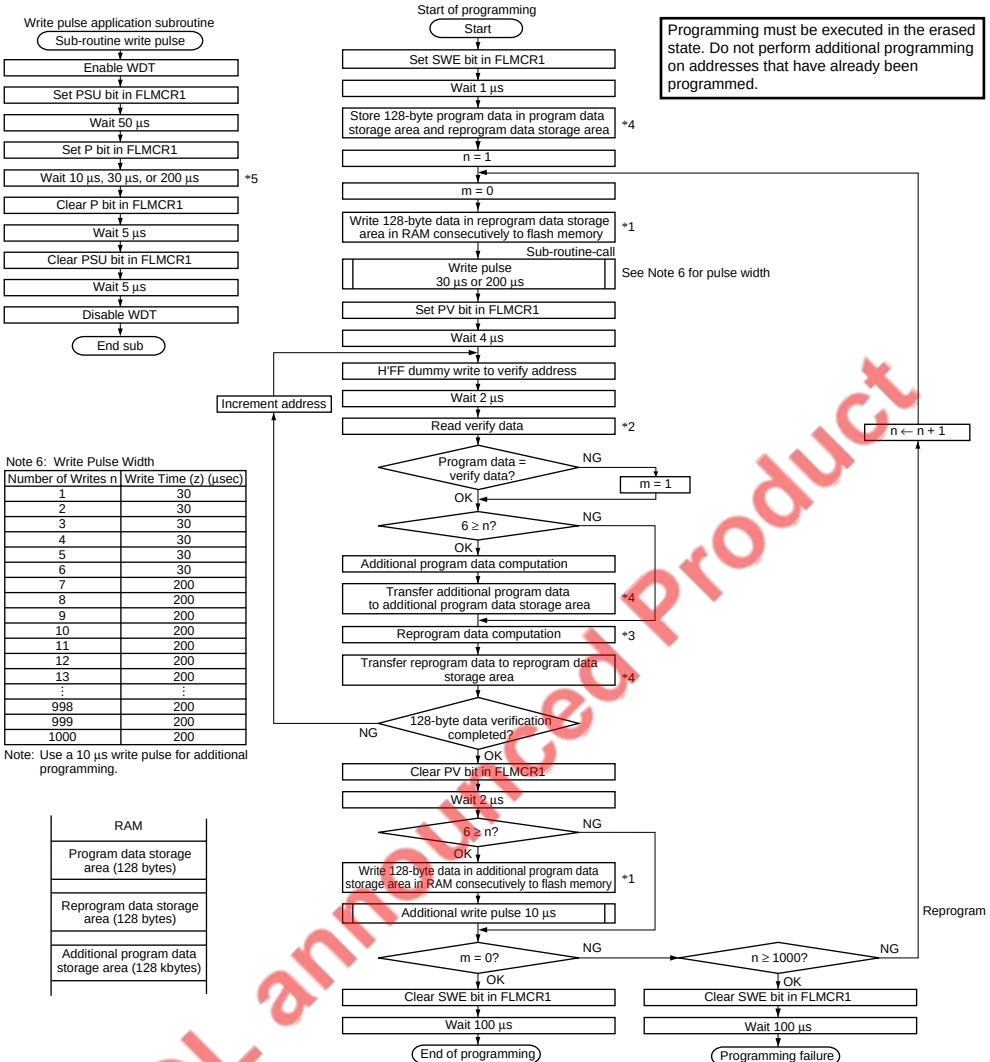
Following the elapse of 1 µs or more after the SWE bit is set to 1 in flash memory control register 1 (FLMCR1), 128-byte program data is stored in the program address (the lower 8 bits of the first address written to must be H'00, or H'80). The program address and program data are latched in the flash memory. 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.

Next, the watchdog timer is set to prevent overprogramming in the event of program runaway, etc. After this, preparation for program mode (program setup) is carried out by setting the PSU bit in FLMCR1, and after the elapse of 50 µs or more, the operating mode is switched to program mode by setting the P bit in FLMCR1. The time during which the P bit is set is the flash memory programming time. For the write time, refer to the table accompanying the Program/Program-Verify Flowchart.

16.7.2 Program-Verify Mode

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

After the elapse of a given programming time, the programming mode is exited (the P bit in FLMCR1 is cleared, then the PSUn bit is cleared at least 5 µs later). The watchdog timer is cleared after the elapse of 5 µs or more, and the operating mode is switched to program-verify mode by setting the PV bit in FLMCR1. Before reading in program-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of 4 µs or more. When the flash memory is read in this state (verify data is read in 32-bit units), the data at the latched address is read. Wait at least 2 µs after the dummy write before performing this read operation. Next, the written data is compared with the verify data, and reprogram data is computed (see figure 16.12) and transferred to the reprogram data area. After 128 bytes of data have been verified, exit program verify mode, wait for at least 2 µs, clear the SWE bit in FLMCR1, then wait for at least 100 µs. If reprogramming is necessary, set program mode again, and repeat the program/program-verify sequence as before. However, ensure that the program/program-verify sequence is not repeated more than 1,000 times on the same bits.



Note 6: Write Pulse Width

Number of Writes n	Write Time (z) (µsec)
1	30
2	30
3	30
4	30
5	30
6	30
7	200
8	200
9	200
10	200
11	200
12	200
13	200
:	:
998	200
999	200
1000	200

Note: Use a 10 µs write pulse for additional programming.

Notes: 1. Data transfer is performed by byte transfer. The lower 8 bits of the first address written to must be H'00 or H'80. 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.
2. Verify data is read in 32-bit (longword) units.
3. Even bits for which programming has been completed in the 128-byte programming loop will be subjected to additional programming if they fail the subsequent verify operation.
4. A 128-byte area for storing program data, a 128-byte area for storing reprogram data, and a 128-byte area for storing additional program data must be provided in RAM. The reprogram and additional program data contents are modified as programming proceeds.
5. The write pulse of 30 µs or 200 µs is applied according to the progress of the programming operation. See Note 6 for the pulse widths. When writing of additional program data is executed, a 10 µs write pulse should be applied. Reprogram data X means reprogram data when the write pulse is applied.

Reprogram Data Computation Chart

Original Data (D)	Verify Data (V)	Reprogram Data (X)	Comments
0	0	1	Programming completed
	1	0	Programming incomplete; reprogram
1	0	1	
	1	1	Still in erased state; no action

Additional Program Data Computation Chart

Reprogram Data (X')	Verify Data (V)	Additional Program Data (Y)	Comments
0	0	0	Additional programming executed
	1	1	Additional programming not executed
1	0	1	Additional programming not executed
	1	1	Additional programming not executed

Figure 16.12 Program/Program-Verify Flowchart

Sample 128-Byte Programming Program: The wait time set values (number of loops) are for the case where f = 20 MHz. For other frequencies, the set value is given by {wait time (µs) × f (MHz) ÷ 4}.

Registers Used
R11 (input): Program data storage address
R12 (input): Programming destination address
R13 (output): OK (normal) or NG (error)
R0–10, 14: Work registers

```
FLMCR1          .EQU  H'80
OK              .EQU  H'0
NG              .EQU  H'1
WAIT_X          .EQU  5             ; 1 µs
WAIT_Y          .EQU  250           ; 50 µs
WAIT_Z1         .EQU  150           ; 30 µs (1st to 6th time)
WAIT_Z5         .EQU  1000          ; 200 µs (7th to 1000th time)
WAIT_ZA         .EQU  50            ; 10 µs (Additional write)
WAIT_A          .EQU  25            ; 5 µs
WAIT_B          .EQU  25            ; 5 µs
WAIT_C          .EQU  20            ; 4 µs
WAIT_D          .EQU  10            ; 2 µs
WAIT_E          .EQU  10            ; 2 µs
WAIT_F          .EQU  500           ; 100 µs
WDT_TCSR.EQU    H'FFFF8610
WDT_1m          .EQU  H'A57A
SWESET          .EQU  H'40          ; B'01000000
PSUSET          .EQU  H'50          ; B'01010000
PSET            .EQU  H'51          ; B'01010001
PCLEAR          .EQU  H'50          ; B'01010000
PSUCLEAR.EQU    H'40  ; B'01000000
PVSET           .EQU  H'44          ; B'01000100
PVCLEAR         .EQU  H'40          ; B'01000000
SWECLEAR.EQU    H'00  ; B'00000000
MAXVERIFY       .EQU  1000
;
FLASHPROGRAM    .EQU  $
        MOV     R11,R3              ; Save program data to
        MOV.L   #RDATABUFF,R0       ; work area
```

```
        MOV.L   #ADATABUFF,R2
        MOV     #32,R6
COPY_LOOP1      .EQU  $
        MOV.L   @R3+,R1
        MOV.L   R1,@R0
        MOV.L   R1,@R2
        ADD     #4,R0
        ADD     #4,R2
        DT      R6
        BF      COPY_LOOP1
        MOV.L   #H'FFFF8500,R0      ; Initialize GBR
        LDC     R0,GBR
;
        MOV.L   #WAIT_X,R2
        MOV     #SWESET,R0
        MOV.B   R0,@(FLMCR1,GBR)    ; Set SWE
WAIT_1  DT      R2                  ; Wait 1 µs
        BF      WAIT_1
;
        MOV     #1,R14              ; Initialize N (R14) to 1
PROGRAM_LOOP    .EQU  $
        MOV     #0,R5               ; Initialize M (R5) to 0
        MOV.L   #128,R2             ; Write 128-byte data consecutively
        MOV.L   #RDATABUFF,R3
        MOV     R12,R6
WRITE_LOOP1     .EQU  $
        MOV.B   @R3+,R1
        MOV.B   R1,@R6
        ADD     #1,R6
        DT      R2
        BF      WRITE_LOOP1
;
        MOV.L   #WDT_TCSR,R0        ; Enable WDT
        MOV.L   #WDT_1m,R1          ; 1.6 ms cycle
        MOV.W   R1,@R0
;
        MOV.L   #WAIT_Y,R2
```

```
        MOV     #PSUSET,R0          ; Set PSU
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_2  DT      R2                  ; Wait 50 µs
        BF      WAIT_2
;
        MOV.L   #WAIT_Z1,R2         ; 1st to 6th time
        MOV     #6,R3
        CMP/GE  R14,R3
        BT      UNDER7
        MOV.L   #WAIT_Z5,R2         ; 7th to 1000th time
UNDER7  MOV     #PSET,R0            ; Set P
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_3  DT      R2                  ; Wait 30 µs or 200 µs
        BF      WAIT_3
;
        MOV.L   #WAIT_A,R2
        MOV     #PCLEAR,R0          ; Clear P
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_4  DT      R2                  ; Wait 5 µs
        BF      WAIT_4
;
        MOV.L   #WAIT_B,R2
        MOV     #PSUCLEAR,R0        ; Clear PSU
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_5  DT      R2                  ; Wait 5 µs
        BF      WAIT_5
;
        MOV.L   #WDT_TCSR,R0        ; Disable WDT
        MOV.W   #H'A55F,R1
        MOV.W   R1,@R0
;
        MOV.L   #WAIT_C,R2
        MOV     #PVSET,R0           ; Set PV
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_6  DT      R2                  ; Wait 4 µs
        BF      WAIT_6
;
```

```
        MOV.L   #ADATABUFF,R9
        MOV.L   #RDATABUFF,R7
        MOV     R11,R1
        MOV     R12,R3
        MOV     #32,R6
        MOV.L   #H'FFFFFFFF,R4
;
VERIFYLOOP      .EQU  $
        MOV.L   R4,@R3          ; Write H'FF to verify address
        MOV.L   R4,@R9          ; Additional program data RAM (ADATABUFF) initialization
        MOV.L   #WAIT_D,R2
WAIT_7  DT      R2              ; Wait 2 µs
        BF      WAIT_7
;
        MOV.L   @R3+,R2         ; Read verify data
        MOV.L   @R1+,R0         ; Read program data (source data)
        CMP/EQ  R2,R0           ; Verify check
        BT      VERIFY_OK
        MOV     #1,R5           ; If verify NG, assign 1 to M
;
VERIFY_OK       .EQU  $
        MOV     #6,R8           ; 6 or more writes?
        CMP/GE  R14,R8
        BF      NO_ADWRT
        MOV.L   @R7,R10         ; Read reprogram data
        OR      R2,R10          ; Additional program data operation
        MOV.L   R10,@R9         ; Store in additional program data RAM (ADATABUFF)
;
NO_ADWRT .EQU   $
        MOV.L   R4,@R7          ; Reprogram data RAM (RDATABUFF) initialization
        NOT     R2,R2           ; Reprogram data computation
        OR      R2,R0
        MOV.L   R0,@R7          ; Store in reprogram data RAM (RDATABUFF)
;
        ADD     #4,R7
        ADD     #4,R9
        DT      R6
```

```
        BF      VERIFYLOOP
;
        MOV.L   #WAIT_E,R2
        MOV     #PVCLEAR,R0         ; Clear PV
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_8  DT      R2                  ; Wait 2 µs
        BF      WAIT_8
;
        MOV     #6,R8               ; 6 or more writes?
        CMP/GE  R14,R8
        BF      NO_ADWRT2
;
        MOV.L   #128,R2             ; Consecutively write 128-byte date to
        MOV.L   #ADATABUFF,R3       ; additional program data RAM (ADATABUFF)
        MOV     R12,R6
WRITE_LOOP2     .EQU  $
        MOV.B   @R3+,R1
        MOV.B   R1,@R6
        ADD     #1,R6
        DT      R2
        BF      WRITE_LOOP2
;
        MOV.L   #WDT_TCSR,R0        ; Enable WDT
        MOV.L   #WDT_1m,R1          ; 1.6 ms cycle
        MOV.W   R1,@R0
;
        MOV.L   #WAIT_Y,R2
        MOV     #PSUSET,R0          ; Set PSU
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_9  DT      R2                  ; Wait 50 µs
        BF      WAIT_9
;
        MOV.L   #WAIT_ZA,R2         ; 10 µs additional write
        MOV     #PSET,R0            ; Set P
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_10 DT      R2                  ; Wait 10 µs
        BF      WAIT_10
```

```
;
        MOV.L   #WAIT_A,R2
        MOV     #PCLEAR,R0          ; Clear P
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_11 DT      R2                  ; Wait 5 µs
        BF      WAIT_11
;
        MOV.L   #WAIT_B,R2
        MOV     #PSUCLEAR,R0        ; Clear PSU
        MOV.B   R0,@(FLMCR1,GBR)
WAIT_12 DT      R2                  ; Wait 5 µs
        BF      WAIT_12
;
        MOV.L   #WDT_TCSR,R0        ; Disable WDT
        MOV.W   #H'A55F,R1
        MOV.W   R1,@R0
;
NO_ADWRT2       .EQU  $
        CMP/PL  R5                  ; If M = 0, end of programming
        BF      PROGRAM_OK
        ADD     #1,R14
        MOV     #NG,R13             ; Move NG (return value) to R13
        MOV.L   #MAXVERIFY,R3       ; If N ≥ 1000, programming error
        CMP/GT  R14,R3
        BF      PROGRAM_END
;
        BRA     PROGRAM_LOOP
        NOP
;
PROGRAM_OK      .EQU  $
        MOV     #OK,R13             ; Move OK (return value) to R13
PROGRAM_END     .EQU  $
        MOV     #SWECLEAR,R0        ; Clear SWE
        MOV.B   R0,@(FLMCR1,GBR)
;
        MOV.L   #WAIT_F,R2
WAIT_13 DT      R2                  ; Wait 100 µs
```

```
        BF      WAIT_13
;
        RTS
        NOP
;
ADATABUFF       .RES.B 128          ; Additional programming RAM area
RDATABUFF       .RES.B 128          ; Reprogramming RAM area
```

16.7.3 Erase Mode

Flash memory is erased one block at a time using the method shown in figure 16.13, Erase/Erase-Verify Flowchart (Single-Block Erasure).

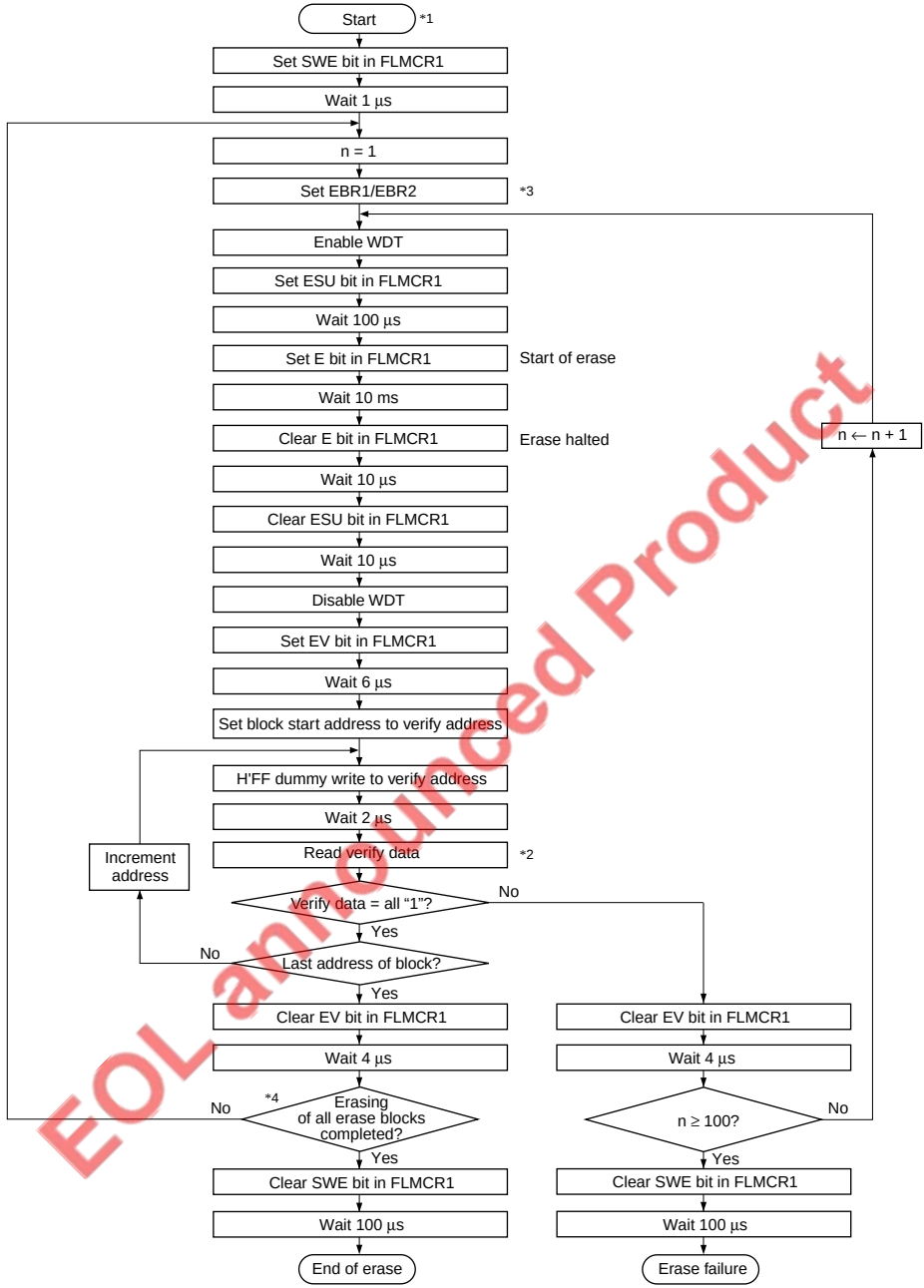
To perform data or program erasure, set the 1-bit flash memory area to be erased in erase block register 1 (EBR1) and erase block register 2 (EBR2) at least 1 µs after setting the SWE bit to 1 in flash memory control register 1 (FLMCR1). Next, the watchdog timer is set to prevent overprogramming in the event of program runaway, etc. After this, preparation for erase mode (erase setup) is carried out by setting the ESU bit in FLMCR1, and after the elapse of 100 µs or more, the operating mode is switched to erase mode by setting the E bit in FLMCR1. The time during which the E bit is set is the flash memory erase time. Ensure that the erase time does not exceed 10 ms.

Note: With flash memory erasing, preprogramming (setting all memory data in the memory to be erased to all “0”) is not necessary before starting the erase procedure.

16.7.4 Erase-Verify Mode

In erase-verify mode, data is read after memory has been erased to check whether it has been correctly erased.

After the elapse of the erase time, erase mode is exited (the E bit in FLMCR1 is cleared, then the ESU bit is cleared at least 10 µs later), the watchdog timer is cleared after the elapse of 10 µs or more, and the operating mode is switched to erase-verify mode by setting the EV bit in FLMCR1. Before reading in erase-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of 6 µs or more. When the flash memory is read in this state (verify data is read in 32-bit units), the data at the latched address is read. Wait at least 2 µs after the dummy write before performing this read operation. If the read data has been erased (all “1”), a dummy write is performed to the next address, and erase-verify is performed. If the read data has not been erased, set erase mode again, and repeat the erase/erase-verify sequence in the same way. However, ensure that the erase/erase-verify sequence is not repeated more than 100 times. When verification is completed, exit erase-verify mode, and wait for at least 4 µs. If erasure has been completed on all the erase blocks, clear the SWE bit in FLMCR1. If there are any unerased blocks, set 1 bit for the flash memory area to be erased, and repeat the erase/erase-verify sequence in the same way.



Notes: 1. Preprogramming (setting erase block data to all "0") is not necessary.
2. Verify data is read in 32-bit (longword) units.
3. Set only one bit in the erase block register (EBR). Two or more bits must not be set.
4. Erasing is performed in block units. For a multiple-block erase, the individual blocks must be erased sequentially.

Figure 16.13 Erase/Erase-Verify Flowchart (Single-Block Erase)

Sample Single-Block Erase Program: The wait time set values (number of loops) are for the case where f = 20 MHz. For other frequencies, the set value is given by {wait time (µs) × f (MHz) ÷ 4}.

Registers Used
R5 (input): Memory block table pointer
R7 (output): OK (normal) or NG (error)
R0–3, 6, 8–9: Work registers

```
FLMCR1          .EQU        H'80
EBR1            .EQU        H'02
OK              .EQU        H'0
NG              .EQU        H'1
EWait_X         .EQU        5
EWait_Y         .EQU        500
EWait_Z         .EQU        50000
EWait_a         .EQU        50
EWait_b         .EQU        50
EWait_c         .EQU        30
EWait_d         .EQU        10
EWait_e         .EQU        20
EWait_f         .EQU        500
WDT_TCSR         EQU        H'FFFF8610
WDT_6m           EQU        H'A57C
SWESET          .EQU        B'01000000
ESUSET          .EQU        B'00100000
ESET            .EQU        B'00000010
ECLEAR          .EQU        B'11111101
ESUCLEAR        .EQU        B'11011111
EVSET           .EQU        B'00001000
EVCLEAR         .EQU        B'11110111
SWECLEAR        .EQU        B'10111111
MAXErase        .EQU        100
;
FlashErase      .EQU        $
        MOV.L   #H'FFFF8500,R0
        LDC     R0,GBR              ; Initialize GBR
        MOV.L   #1,R2
;
```

```
                MOV,L     #EWait_X,R3
                MOV.L     #FLMCR1,R0
                OR.B      #SWESET,@(R0,GBR)          ; Set SWE
EWait_1         SUBC      R2,R3                      ; Wait 1 µs
                BF        EWait_1
;
                MOV.L     #0,R9                      ; Initialize n (R9) to 0
;
                MOV.W     @(6,R5),R0
                MOV.W     R0,@(EBR1,GBR)             ; Erase memory block (EBR1/2) setting
                MOV.L     @R5,R6                     ; Erase memory block start address →
                                                     ; R6 setting
;
EraseLoop                 .EQU              $
                MOV.L     #WDT_TCSR,R1               ; Enable WDT
                MOV.W     #WDT_6m,R3                 ; 6.5 ms cycle
                MOV.W     R3,@R1
                MOV.L     #EWait_Y,R3
                MOV.L     #FLMCR1,R0
                OR.B      #ESUSET,@(R0,GBR)          ; Set ESU
EWait_2         SUBC      R2,R3                      ; Wait 100 µs
                BF        EWait_2
;
                MOV.L     #EWait_Z,R3
                OR.B      #ESET,@(R0,GBR)            ; Set E
EWait_3         SUBC      R2,R3                      ; Wait 10 ms
                BF        EWait_3
;
                MOV.L     #EWait_a,R3
                AND.B     #ECLEAR,@(R0,GBR)          ; Clear E
EWait_4         SUBC      R2,R3                      ; Wait 10 µs
                BF        EWait_4
;
                MOV.L     #EWait_b,R3
                AND.B     #ESUCLEAR,@(R0,GBR)        ; Clear ESU
EWait_5         SUBC      R2,R3                      ; Wait 10 µs
                BF        EWait_5
```

```
;
                MOV.L   #WDT_TCSR,R1            ; Disable WDT
                MOV.W   #H'A55F,R3
                MOV.W   R3,@R1
;
                MOV.L   #EWait_c,R3
                OR.B    #EVSET,@(R0,GBR)        ; Set EV
EWait_6         SUBC    R2,R3                   ; Wait 6 µs
                BF      EWait_6
;
BlockVerify_1           .EQU            $       ; Erase-verify
                MOV.L   #H'FFFFFFFF,R8
                MOV.L   R8,@R6                  ; H'FF dummy write
                MOV.L   #EWait_d,R3
EWait_7         SUBC    R2,R3                   ; Wait 2 µs
                BF      EWait_7
;
                MOV.L   @R6+,R1                 ; Read verify data
                CMP/EQ  R8,R1
                BF      BlockVerify_NG
                MOV.L   @(8,R5),R7
                CMP/EQ  R6,R7                   ; Check for last address of memory block
                BF      BlockVerify_1
                MOV.L   #EWait_e,R3
                AND.B   #EVCLEAR,@(R0,GBR)      ; Clear EV
EWait_8         SUBC    R2,R3                   ; Wait 4 µs
                BF      EWait_8
;
                MOV.L   #OK,R7                  ; Move OK (return value) to R7
                BRA     FlashErase_end          ; Verify OK
                NOP
;
BlockVerify_NG          .EQU            $
                ADD.L   #1,R9                   ; If verify NG, assign n+1 to n
                ADD.L   #-4,R6                  ; Next verify address
                MOV.L   #EWait_e,R3
                AND.B   #EVCLEAR,@(R0,GBR)      ; Clear EV
```

```
EWait_9         SUBC        R2,R3                   ; Wait 4 µs
                BF          EWait_9
;
                MOV.L       #MAXErase,R7            ; If N > 100, erase error
                CMP/EQ      R7,R9
                BF          EraseLoop
                MOV.L       #NG,R7                  ; Move NG (return value) to R7
FlashErase_end              .EQU            $
                MOV.L       #FLMCR1,R0
                AND.B       #SWECLEAR,@(R0,GBR)     ; Clear SWE
                MOV.L       #Ewait_f,R3
Ewait_10        SUBC        R2,R3                   ; Wait 100 µs
                BF          Ewait_10
;
                RTS
                NOP
;
;Memory block table          Memory block start address: EBR value
                .ALIGN      4
Flash_BlockData             .EQU            $
EB0             .DATA.L     H'00000000,H'00000100
EB1             .DATA.L     H'00001000,H'00000200
EB2             .DATA.L     H'00002000,H'00000400
EB3             .DATA.L     H'00003000,H'00000800
EB4             .DATA.L     H'00004000,H'00001000
EB5             .DATA.L     H'00005000,H'00002000
EB6             .DATA.L     H'00006000,H'00004000
EB7             .DATA.L     H'00007000,H'00008000
EB8             .DATA.L     H'00008000,H'00000001
EB9             .DATA.L     H'00010000,H'00000002
EB10            .DATA.L     H'00020000,H'00000004
Dummy           .DATA.L     H'00028000
```

16.7.5 Wait Time Widths in Programming/Erasing

Various wait time widths in the program/erase control program provided by the user should be within the specifications shown below.

Table 16.7 Programming/Erasing-Related Wait Width Specifications

Flow Section	Item	Symbol	Min	Typ	Max	Unit	Notes
Programming-related	Wait time after PSU bit setting	tspsu	50	50	—	µs	
	Wait time after P bit setting (10 µs)	tsp10	8	10	12	µs	Additional-programming time wait
	Wait time after P bit setting (30 µs)	tsp30	28	30	32	µs	Programming time wait
	Wait time after P bit setting (200 µs)	tsp200	198	200	202	µs	Programming time wait
	Wait time after P bit clearing	tcp	5	5	—	µs	
	Wait time after PSU bit clearing	tcpsu	5	5	—	µs	
	Wait time after PV bit setting	tspv	4	4	—	µs	
	Wait time after dummy write	tspvr	2	2	—	µs	
	Wait time after PV bit clearing	tcpv	2	2	—	µs	
Erase-related	Wait time after ESU bit setting	tsesu	100	100	—	µs	
	Wait time after E bit setting	tse	10	10	100	ms	Erase time wait
	Wait time after E bit clearing	tce	10	10	—	µs	
	Wait time after ESU bit clearing	tcesu	10	10	—	µs	
	Wait time after EV bit setting	tsev	6	6	—	µs	
	Wait time after dummy write	tsevr	2	2	—	µs	
	Wait time after EV bit clearing	tcev	4	4	—	µs	
Other (common)	Wait time after SWE bit setting	tsswe	1	1	—	µs	
	Wait time after SWE bit clearing	tcswe	100	100	—	µs	

16.8 Protection

There are two kinds of flash memory program/erase protection, hardware protection and software protection.

16.8.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted. Hardware protection is reset by settings in flash memory control register 1 (FLMCR1), erase block register 1 (EBR1), and erase block register 2 (EBR2). The FLMCR1, EBR1, and EBR2 settings are retained in the error-protected state. (See table 16.8.)

Table 16.8 Hardware Protection

Item	Description	Functions	
		Program	Erase
FWP pin protection	• When a high level is input to the FWP pin, FLMCR1, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered.	Yes	Yes
Reset/standby protection	• In a reset (including a WDT overflow reset) and in standby mode, FLMCR1, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered. • In a power-on 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.	Yes	Yes

16.8.2 Software Protection

Software protection can be implemented by setting the SWE bit in flash memory control register 1 (FLMCR1), the RAMS bit in erase block register 1 (EBR1), erase block register 2 (EBR2), and RAM emulation register (RAMER). When software protection is in effect, setting the P or E bit in flash memory control register 1 (FLMCR1) does not cause a transition to program mode or erase mode. (See table 16.9.)

Table 16.9 Software Protection

Item	Description	Functions	
		Program	Erase
SWE bit protection	• Clearing the SWE bit to 0 in FLMCR1 sets the program/erase-protected state for all blocks. (Execute in on-chip RAM or external memory.)	Yes	Yes
Block specification protection	• Erase protection can be set for individual blocks by settings in erase block register 1 (EBR1) and erase block register 2 (EBR2). • Setting EBR1 and EBR2 to H'00 places all blocks in the erase-protected state.	—	Yes
Emulation protection	• Setting the RAMS bit to 1 in the RAM emulation register (RAMER) places all blocks in the program/erase-protected state.	Yes	Yes

16.8.3 Error Protection

In error protection, an error is detected when SH7018 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.

If the SH7018 malfunctions during flash memory programming/erasing, the FLER bit is set to 1 in FLMCR2 and the error protection state is entered. The FLMCR1, EBR1, and EBR2 settings are retained, but 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.

FLER bit setting conditions are as follows:

1. When flash memory is read during programming/erasing (including a vector read or instruction fetch)
2. Immediately after exception processing (excluding a reset) during programming/erasing
3. When a SLEEP instruction (including software standby) is executed during programming/erasing
4. When the bus is released during programming/erasing

Error protection is released only by a power-on reset and in hardware standby mode.

Figure 16.14 shows the flash memory state transition diagram.

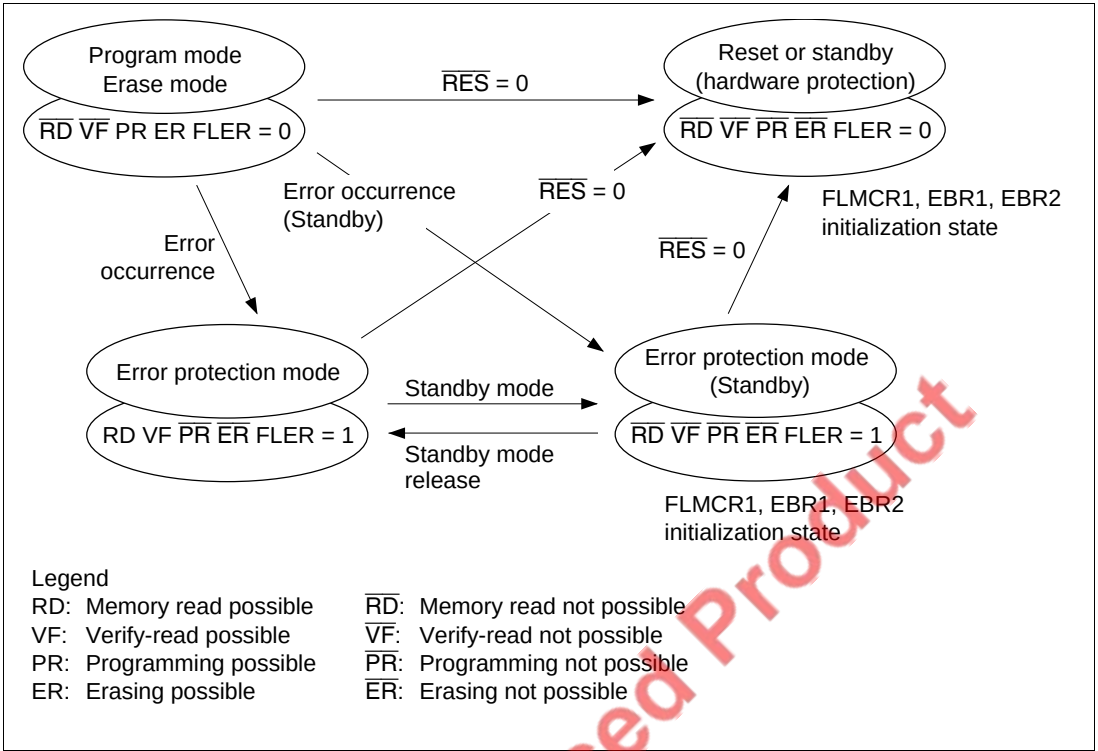


Figure 16.14 Flash Memory State Transitions

16.9 Flash Memory Emulation in RAM

Making a setting in the RAM emulation register (RAMER) enables part of RAM to be overlapped onto the flash memory area so that data to be written to flash memory can be emulated in RAM in real time. After the RAMER setting has been made, accesses cannot be made from the flash memory area or the RAM area overlapping flash memory. Emulation can be performed in user mode and user program mode. Figure 16.15 shows an example of emulation of real-time flash memory programming.

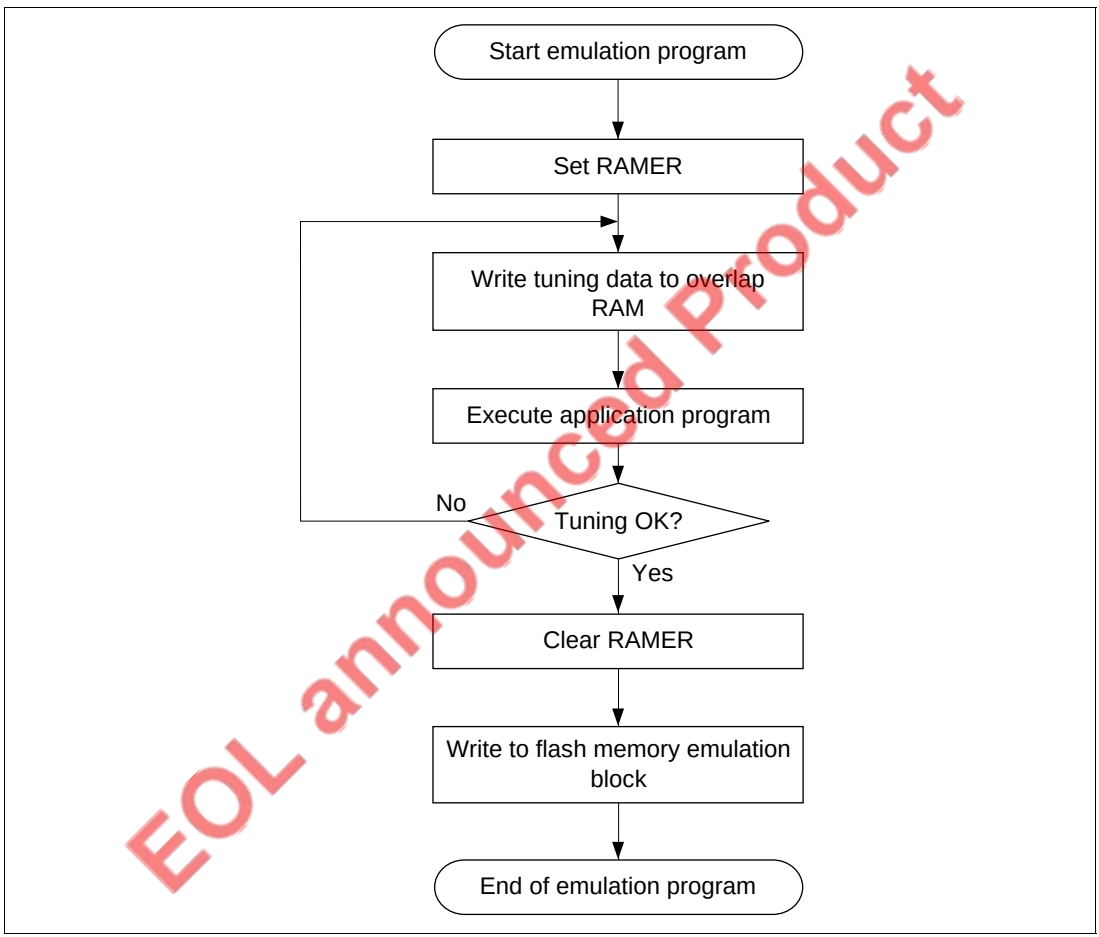


Figure 16.15 Flowchart for Flash Memory Emulation in RAM

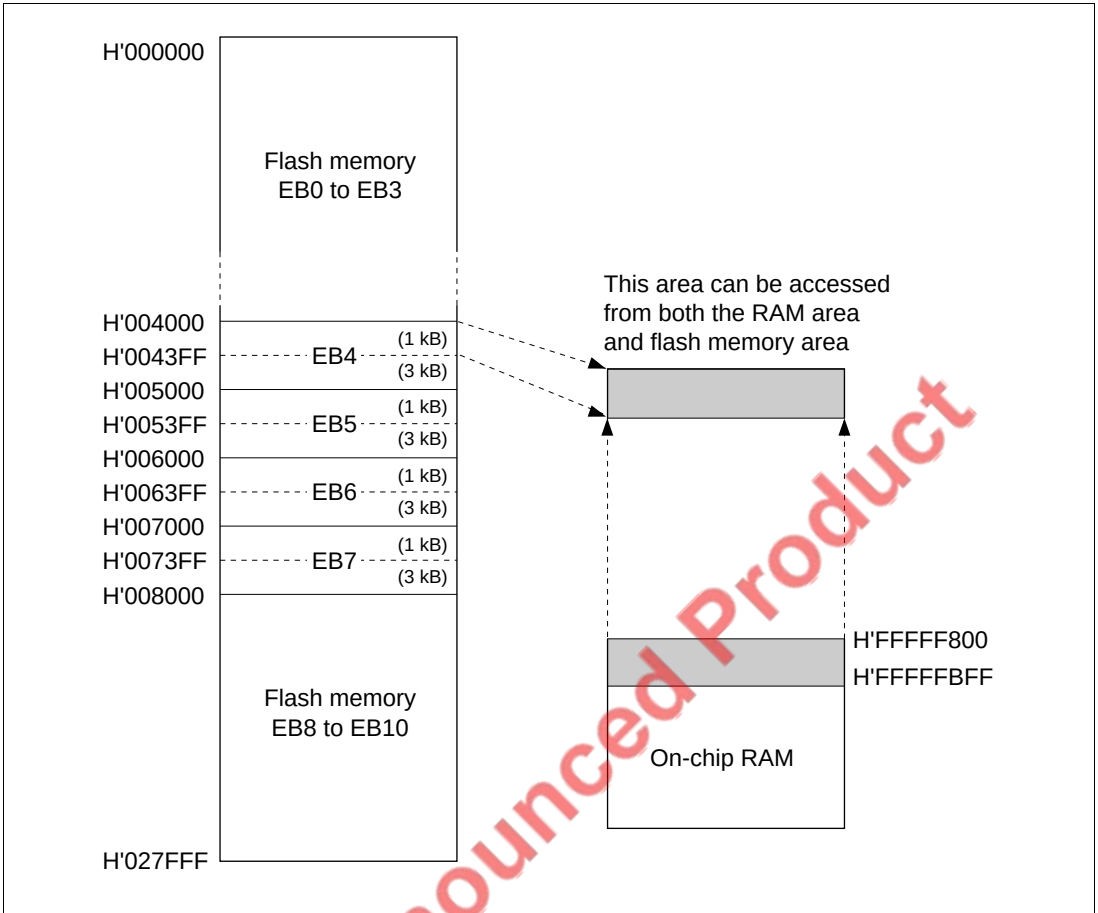


Figure 16.16 Example of RAM Overlap Operation

Example in Which Flash Memory Block Area (EB4) is Overlapped

1. Set bits RAMS, RAM1, and RAM0 in RAMER to 1, 0, 0, to overlap part of RAM (H'FFFFF800 to H'FFFFFBFF) onto the area (part of EB4: H'004000 to H'0043FF) for which real-time programming is required.
2. Real-time programming is performed using the overlapping RAM.
3. After the program data has been confirmed, the RAMS bit is cleared, releasing RAM overlap.
4. The data written in the overlapping RAM is written into the flash memory space (EB4).

Notes: 1. When the RAMS bit is set to 1, program/erase protection is enabled for all blocks regardless of the value of RAM1 and RAM0 (emulation protection). In this state, setting the P or E bit in flash memory control register 1 (FLMCR1) will not cause a transition to program mode or erase mode. When actually programming a flash memory area, the RAMS bit should be cleared to 0.

2. A RAM area cannot be erased by execution of software in accordance with the erase algorithm while flash memory emulation in RAM is being used.

16.10 Note on Flash Memory Programming/Erasing

In the on-board programming modes (user mode and user program mode), NMI input should be disabled to give top priority to the program/erase operations (including RAM emulation).

16.11 Flash Memory Programmer Mode

Programs and data can be written and erased in programmer mode as well as in the on-board programming modes. In programmer mode, flash memory read mode, auto-program mode, auto-erase mode, and status read mode are supported. In auto-program mode, auto-erase mode, and status read mode, a status polling procedure is used, and in status read mode, detailed internal signals are output after execution of an auto-program or auto-erase operation.

Table 16.10 shows the pin settings for programmer mode. For the pin names in programmer mode, see section 1.3.2, Pin Functions.)

Table 16.10 Programmer Mode Pin Settings

Pin Names	Settings
Mode pins: MD3, MD2, MD1, MD0	MD3 = 1, MD2 = 1, MD1 = 0, MD0 = 1
FWE pin	High level input (in auto-program and auto-erase modes)
$\overline{\text{RES}}$ pin	Power-on reset circuit
XTAL, EXTAL pins	Oscillator circuit

Note: In programmer mode, the FWP pin has its polarity reversed and functions as the FWE (flash write enable) pin.

16.11.1 Socket Adapter Pin Correspondence Diagram

Connect the socket adapter to the chip as shown in figure 16.18. This will enable conversion to a 32-pin arrangement. The on-chip ROM memory map is shown in figure 16.17, and the socket adapter pin correspondence diagram in figure 16.18.

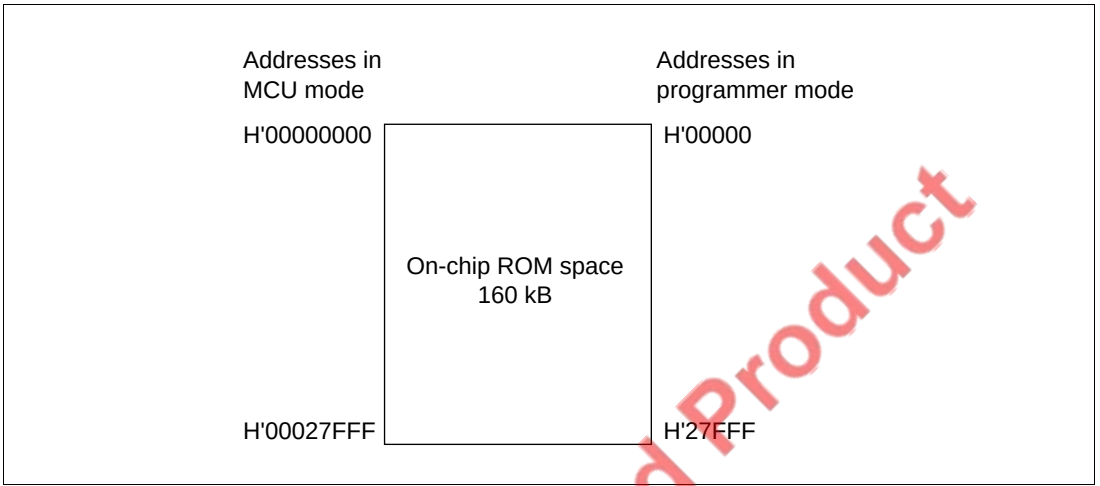
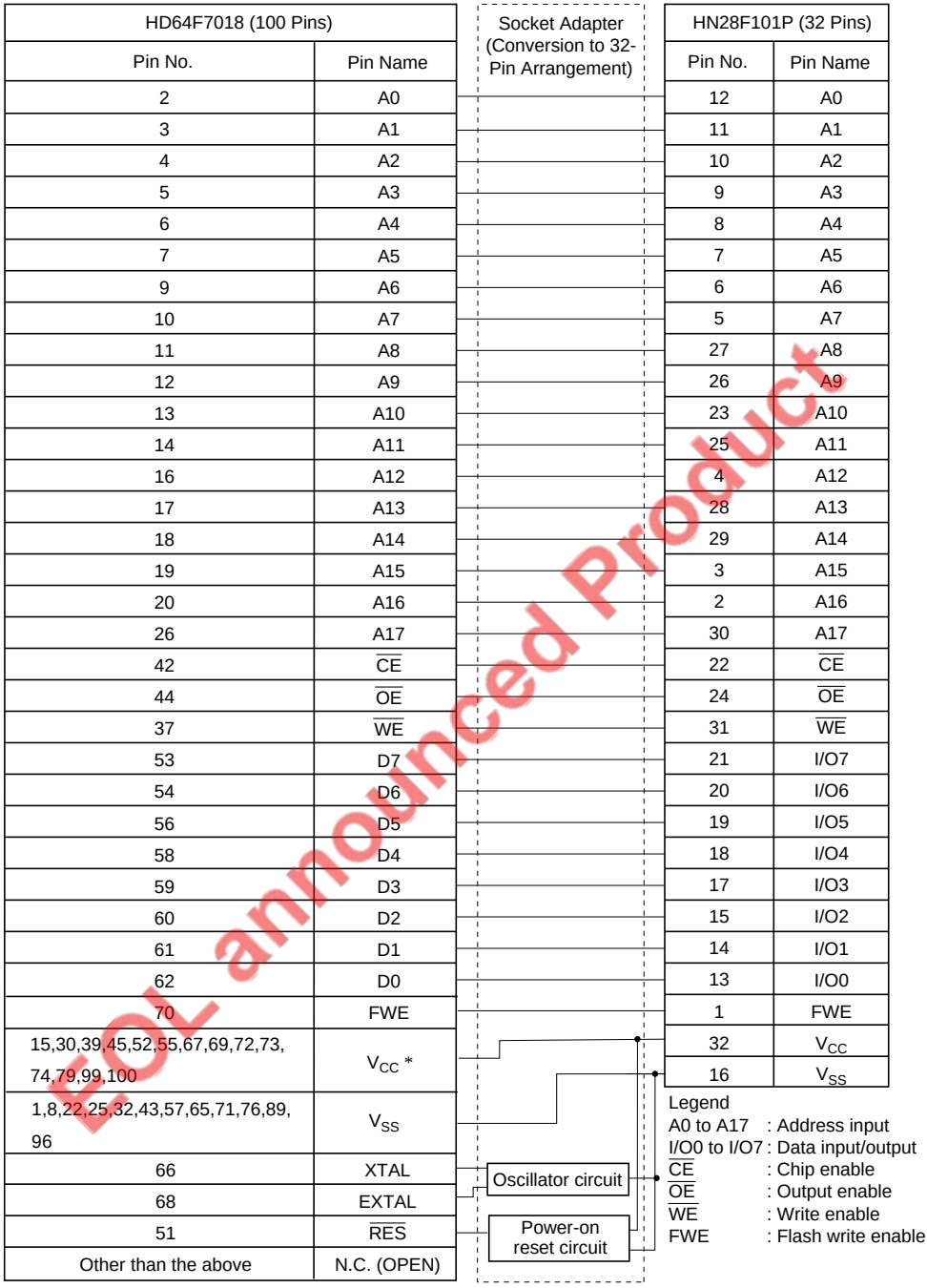


Figure 16.17 On-Chip ROM Memory Map



Note: * Includes NMI as well as MD0, MD1, MD2, and MD3.

Figure 16.18 Socket Adapter Pin Correspondence Diagram

16.11.2 Programmer Mode Operation

Table 16.11 shows how the different operating modes are set when using programmer mode, and table 16.12 lists the commands used in programmer mode. Details of each mode are given below.

- Memory Read Mode

  Memory read mode supports byte reads.
- Auto-Program Mode

  Auto-program mode supports programming of 128 bytes at a time. Status polling is used to confirm the end of auto-programming.
- Auto-Erase Mode

  Auto-erase mode supports automatic erasing of the entire flash memory. Status polling is used to confirm the end of auto-programming.
- Status Read Mode

  Status polling is used for auto-programming and auto-erasing, and normal termination can be confirmed by reading the I/O6 signal. In status read mode, error information is output if an error occurs.

Table 16.11 Settings for Various Operating Modes In Programmer Mode

	Pin Names					
Mode	**FWE**	**$\overline{CE}$**	**$\overline{OE}$**	**$\overline{WE}$**	**I/O0 to I/O7**	**A0 to A17**
Read	H or L	L	L	H	Data output	Ain
Output disable	H or L	L	H	H	Hi-Z	Ain
Command write	H or L	L	H	L	Data input	*Ain
Chip disable	H or L	H	X	X	Hi-Z	Ain

Notes: 1. Chip disable is not a standby state; internally, it is an operation state.
2. *Ain indicates that there is also address input in auto-program mode.
3. For command writes in auto-program and auto-erase modes, input a high level to the FWE pin.

Table 16.12 Programmer Mode Commands

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

Notes: 1. In auto-program mode, 129 cycles are required for command writing by a simultaneous 128-byte write.
2. In memory read mode, the number of cycles depends on the number of address write cycles (n).

16.11.3 Memory Read Mode

1. 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.
2. In memory read mode, command writes can be performed in the same way as in the command wait state.
3. Once memory read mode has been entered, consecutive reads can be performed.
4. After powering on, memory read mode is entered.

Table 16.13 AC Characteristics in Transition to Memory Read Mode (Conditions: V_{CC} = 3.3 V ± 0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	t_{nxtc}	20	—	µs	
$\overline{CE}$ 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	

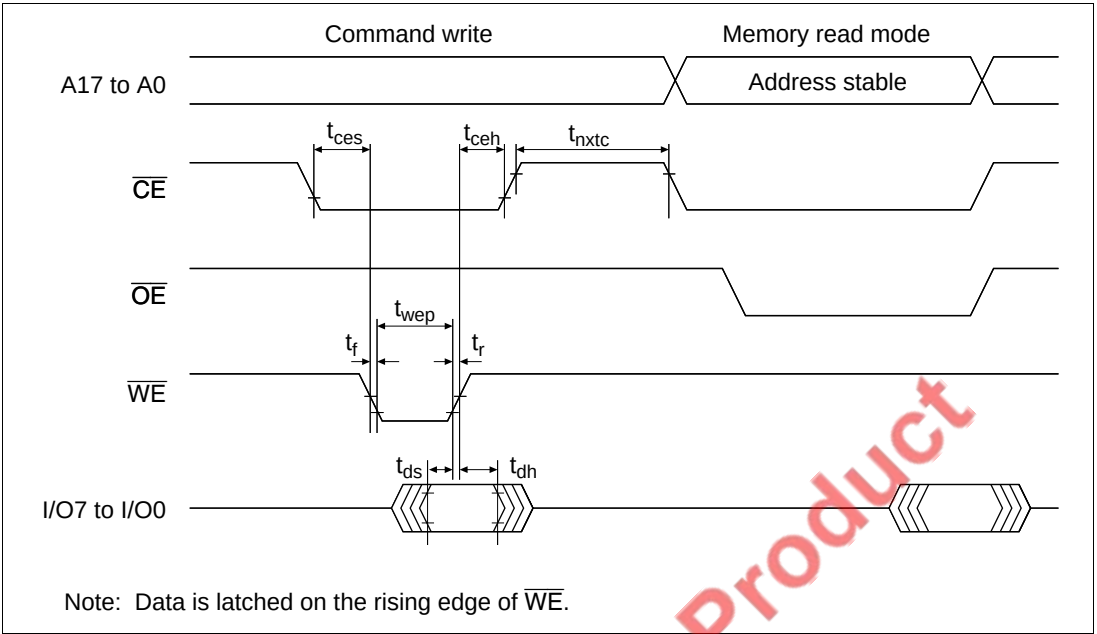
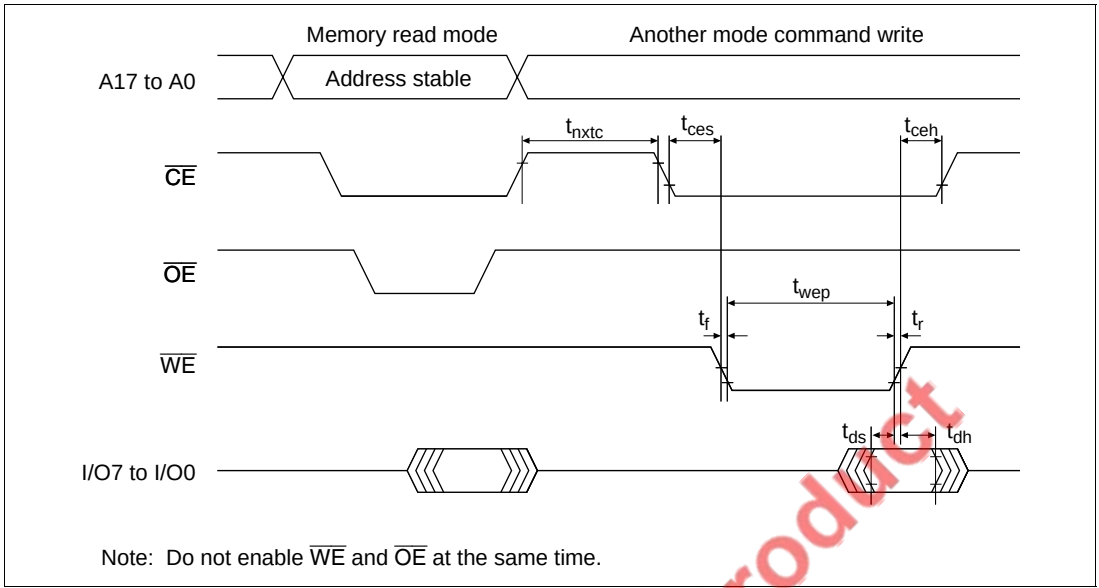


Figure 16.19 Timing Waveforms for Memory Read after Command Write

Table 16.14 AC Characteristics in Transition from Memory Read Mode to Another Mode (Conditions: V_{CC} = 3.3 V ± 0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	t_{nxtc}	20	—	µs	
$\overline{CE}$ 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	



Note: Do not enable $\overline{WE}$ and $\overline{OE}$ at the same time.

Figure 16.20 Timing Waveforms in Transition from Memory Read Mode to Another Mode

Table 16.15 AC Characteristics in Memory Read Mode (Conditions: V_{CC} = 3.3 V ± 0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Notes
Access time	t_{acc}	—	20	µs	
$\overline{CE}$ 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	

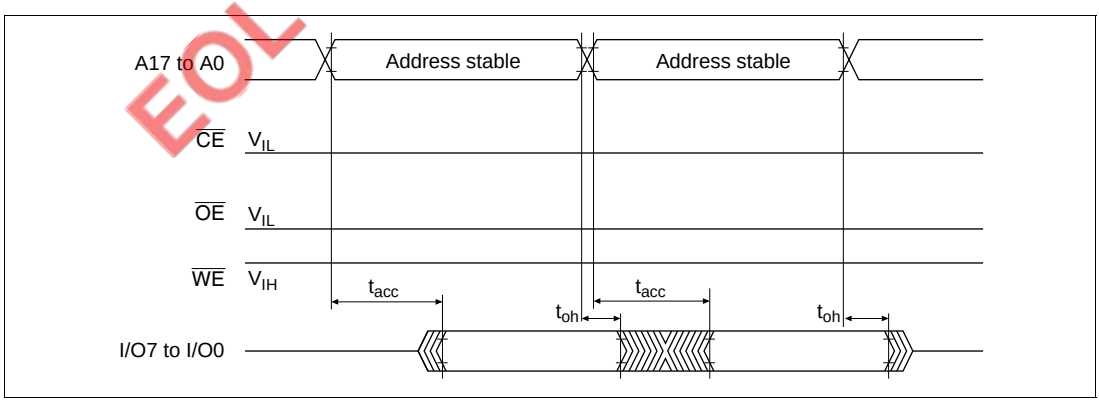


Figure 16.21 $\overline{CE}$ and $\overline{OE}$ Enable State Read Timing Waveforms

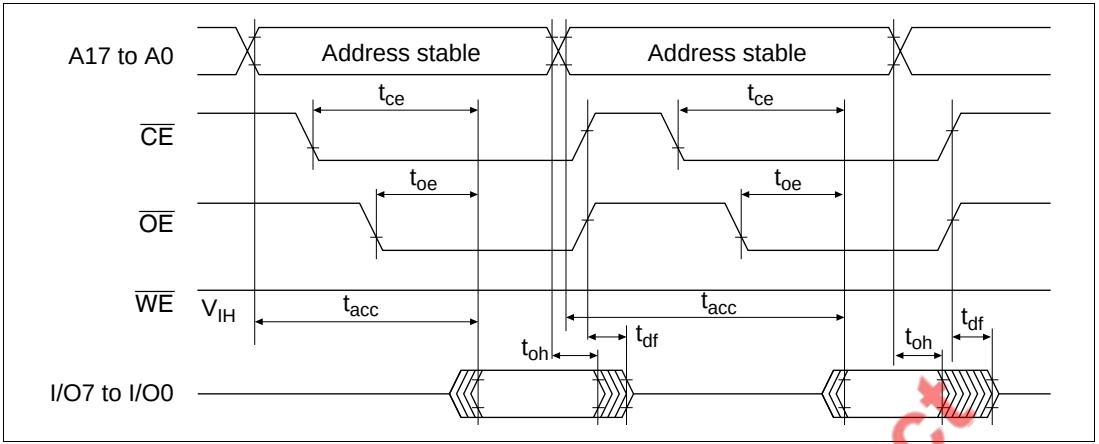


Figure 16.22 $\overline{CE}$ and $\overline{OE}$ Clock System Read Timing Waveforms

16.11.4 Auto-Program Mode

1. In auto-program mode, 128 bytes are programmed simultaneously. This should be carried out by executing 128 consecutive byte transfers.
2. 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.
3. 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.
4. Memory address transfer is performed in the second cycle (figure 16.23). Do not perform transfer after the third cycle.
5. Do not perform a command write during a programming operation.
6. 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.
7. 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 identification pin).
8. 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 $\overline{OE}$.

Table 16.16 AC Characteristics in Auto-Program Mode (Conditions: V_{CC} = 3.3 V ± 0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	t_{nxtc}	20	—	µs	
$\overline{CE}$ 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	
Write setup time	t_{pns}	100	—	ns	
Write end setup time	t_{pnh}	100	—	ns	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

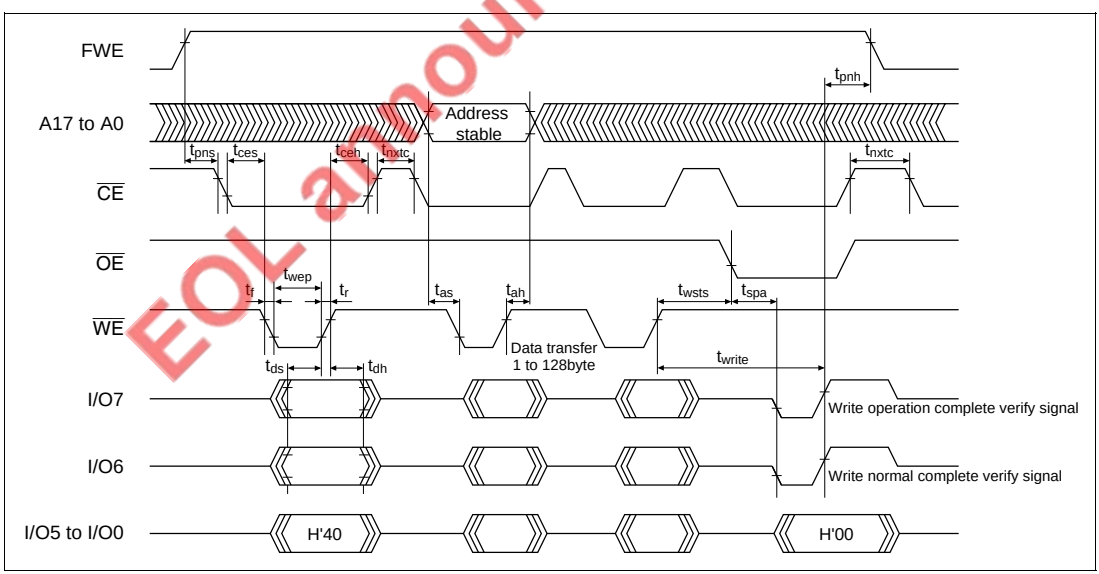


Figure 16.23 Auto-Program Mode Timing Waveforms

16.11.5 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 identification 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 $\overline{OE}$.

Table 16.17 AC Characteristics in Auto-Erase Mode (Conditions: V_{CC} = 3.3 V ± 0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	t_{nxtc}	20	—	µs	
$\overline{CE}$ 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	
Erase setup time	t_{ens}	100	—	ns	
Erase end setup time	t_{enh}	100	—	ns	
$\overline{WE}$ rise time	t_r	—	30	ns	
$\overline{WE}$ fall time	t_f	—	30	ns	

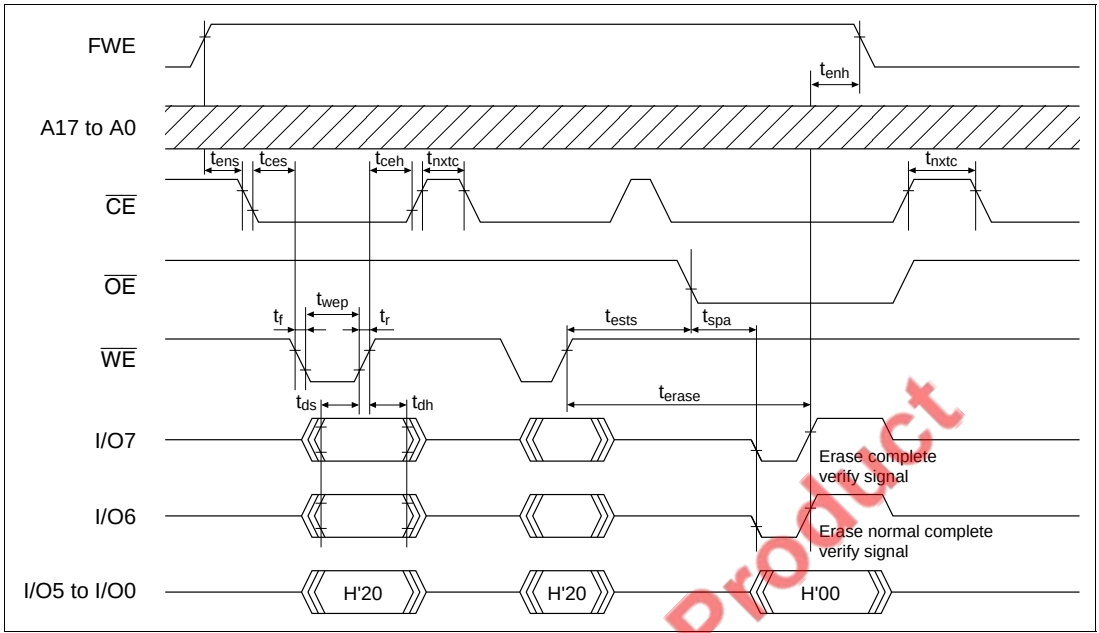


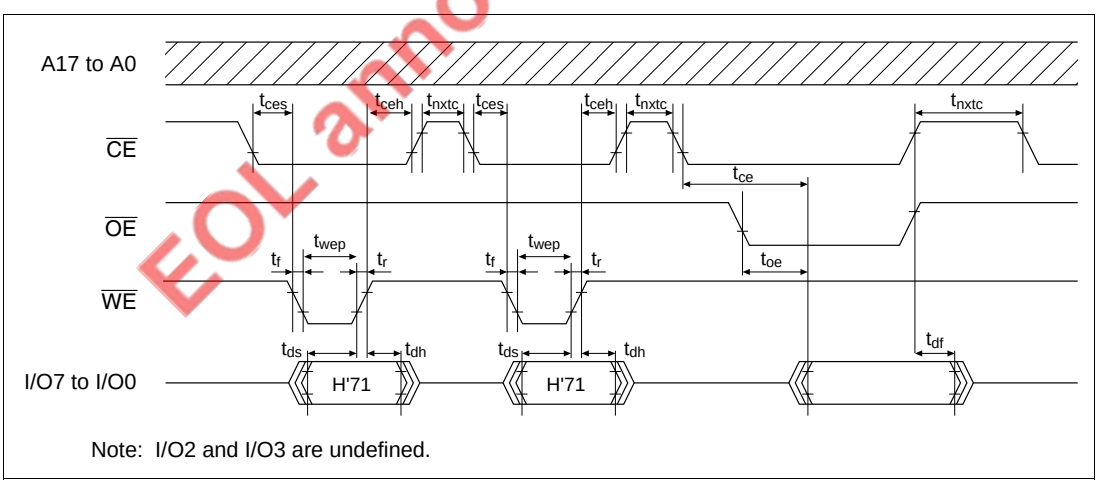
Figure 16.24 Auto-Erase Mode Timing Waveforms

16.11.6 Status Read Mode

1. Status read mode is provided to specify 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 a status read mode command write is executed.

Table 16.18 AC Characteristics in Status Read Mode (Conditions: V_{CC} = 3.3 V ± 0.3 V, V_{SS} = 0 V, T_a = 25°C ±5°C)

Item	Symbol	Min	Max	Unit	Notes
Read time after command write	t_{nxtc}	20	—	µs	
$\overline{CE}$ 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	



Note: I/O2 and I/O3 are undefined.

Figure 16.25 Status Read Mode Timing Waveforms

Table 16.19 Status Read Mode Return Commands

Pin Name	I/O7	I/O6	I/O5	I/O4	I/O3	I/O2	I/O1	I/O0
Attribute	Normal end identification	Command error	Program-ming error	Erase error	—	—	Program-ming or erase count exceeded	Effective address error
Initial value	0	0	0	0	0	0	0	0
Indications	Normal end: 0 Abnormal end: 1	Command Error: 1 Otherwise: 0	Program-ming Error: 1 Otherwise: 0	Erasing Error: 1 Otherwise: 0	—	—	Count exceeded: 1 Otherwise: 0	Effective address Error: 1 Otherwise: 0

Note: I/O2 and I/O3 are undefined at present.

16.11.7 Status Polling

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

Table 16.20 Status Polling Output Truth Table

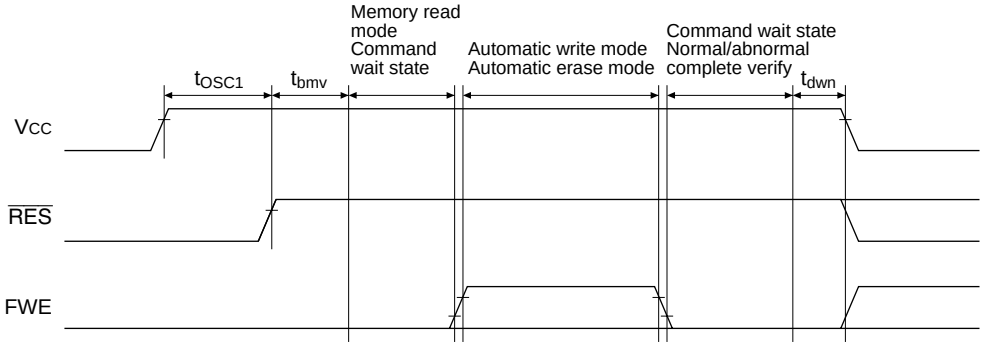
Pin Name	During Internal Operation	Abnormal End		Normal End
I/O7	0	1	0	1
I/O6	0	0	1	1
I/O0 to I/O5	0	0	0	0

16.11.8 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 16.21 Stipulated Transition Times to Command Wait State

Item	Symbol	Min	Max	Unit
Standby release (oscillation stabilization time)	t_{osc1}	30	—	ms
Programmer mode setup time	t_{bmv}	10	—	ms
V_{CC} hold time	t_{dwn}	0	—	ms



Note : For the level of FWE input pin, set V_{IL} when using other than the automatic write mode and automatic erase mode.

Figure 16.26 Oscillation Stabilization Time, Boot Program Transfer Time, and Power-Down Sequence

16.11.9 Notes On Memory Programming

1. When programming addresses which have previously been programmed, carry out auto-erasing before auto-programming.
2. When performing programming using PROM mode on a chip that has been programmed/erased in an on-board programming mode, auto-erasing is recommended before carrying out auto-programming.

Notes: 1. The flash memory is initially in the erased state when the device is shipped by Hitachi. 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.

2. Auto-programming should be performed once only on the same address block. Additional programming cannot be performed on previously programmed address blocks.

17.1 Overview

The SH7018 has 4 kbytes of on-chip RAM. The on-chip RAM is connected to the CPU by a 32-bit data bus (figure 17.1). The CPU can access data in the on-chip RAM in 8, 16, or 32 bit widths. On-chip RAM data can always be accessed in one state, making the RAM ideal for use as a program area, stack area, or data area, which require high-speed access. The contents of the on-chip RAM are held in sleep mode. Memory area addresses H'FFFFE000 to H'FFFFFFFF are allocated to the on-chip RAM.

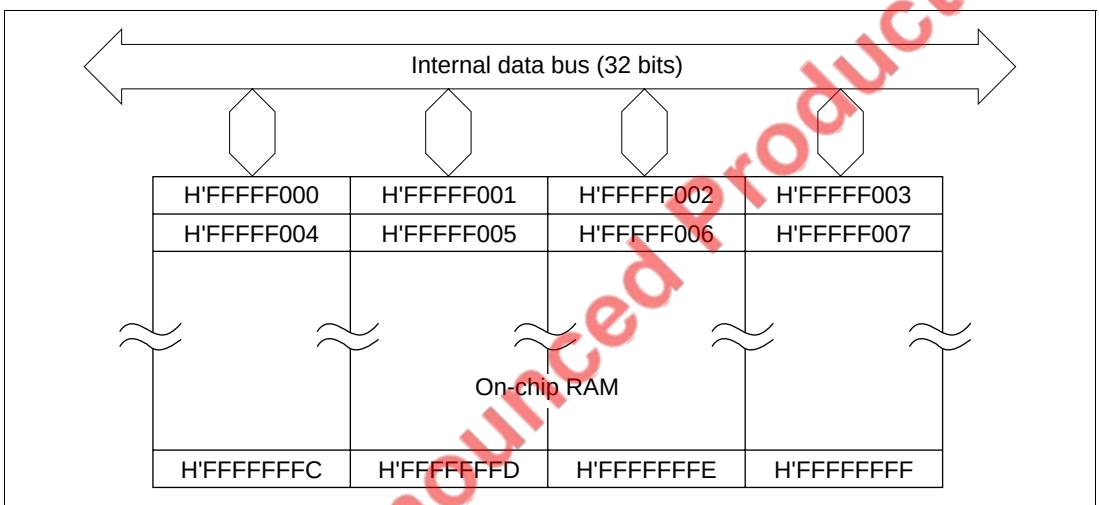


Figure 17.1 Block Diagram of RAM

Section 18 Power-Down State

18.1 Overview

In the power-down state, the CPU functions are halted. This enables a great reduction in power consumption.

18.1.1 Power-Down States

The power-down state is effected by the following two modes:

- Sleep mode
- Standby mode

Table 18.1 describes the transition conditions for entering the modes from the program execution state as well as the CPU and peripheral function status in each mode and the procedures for canceling each mode.

Table 18.1 Power-Down State Conditions

Mode	Entering Procedure	State						Canceling Procedure
		Clock	CPU	On-Chip Peripheral Modules	CPU Registers	RAM	I/O Ports	
Sleep	Execute SLEEP instruction with SBY bit set to 0 in SBYCR	Run	Halt	Run	Held	Held	Held	• Interrupt • DMAC address error • Power-on reset
Stand-by	Execute SLEEP instruction with SBY bit set to 1 in SBYCR	Halt	Halt	Halt*[1]	Held	Held	Held or high impe-dance*[2]	• NMI interrupt • Power-on reset

SBYCR: standby control register.

SBY: standby bit

Notes: 1. Some bits within on-chip peripheral module registers are initialized by the standby mode; some are not. Refer to table 18.3, Register States in the Standby Mode, in section 18.4.1, Transition to Standby Mode. Also refer to the register descriptions for each peripheral module.

2. The status of the I/O port in standby mode is set by the port high impedance bit (HIZ) of the SBYCR. Refer to section 18.2, Standby Control Register (SBYCR). For pin status other than for the I/O port, refer to Appendix B, Pin Status.

18.1.2 Related Register

Table 18.2 shows the register used for power-down state control.

Table 18.2 Related Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Standby control register	SBYCR	R/W	H'1F	H'FFFF8614	8, 16, 32

18.2 Standby Control Register (SBYCR)

The standby control register (SBYCR) is a read/write 8-bit register that sets the transition to standby mode, and the port status in standby mode. The SBYCR is initialized to H'1F when reset.

Bit:	7	6	5	4	3	2	1	0
	SBY	HIZ	—	—	—	—	—	—
Initial value:	0	0	0	1	1	1	1	1
R/W:	R/W	R/W	R	R	R	R	R	R

- Bit 7—Standby (SBY): Specifies transition to the standby mode. The SBY bit cannot be set to 1 while the watchdog timer is running (when the timer enable bit (TME) of the WDT timer control/status register (TCSR) is set to 1). To enter the standby mode, always halt the WDT by 0 clearing the TME bit, then set the SBY bit.

Bit 7: SBY	Description	
0	Executing SLEEP instruction puts the LSI into sleep mode	(Initial value)
1	Executing SLEEP instruction puts the LSI into standby mode	

- Bit 6—Port High Impedance (HIZ): In the standby mode, this bit selects whether to set the I/O port pin to high impedance or hold the pin status. The HIZ bit cannot be set to 1 when the TME bit of the WDT timer control/status register (TCSR) is set to 1. When making the I/O port pin status high impedance, always clear the TME bit to 0 before setting the HIZ bit.

Bit 6: HIZ	Description	
0	Holds pin status while in standby mode	(Initial value)
1	Keeps pin at high impedance while in standby mode	

- Bits 5 to 0—Reserved: Bit 5 is always reads as 0. The write value should always be 0. Bits 4 to 0 are always read as 1. The write value should always be 0.

18.3 Sleep Mode

18.3.1 Transition to Sleep Mode

Executing the SLEEP instruction when the SBY bit of SBYCR is 0 causes a transition from the program execution state to the sleep mode. Although the CPU halts immediately after executing the SLEEP instruction, the contents of its internal registers remain unchanged. The on-chip peripheral modules continue to run during the sleep mode.

18.3.2 Canceling Sleep Mode

Sleep mode is canceled by an interrupt, DMAC address error, or power-on reset.

Cancellation by an Interrupt: When an interrupt occurs, the sleep mode is canceled and interrupt exception processing is executed. The sleep mode is not canceled if the interrupt cannot be accepted because its priority level is equal to or less than the mask level set in the CPU's status register (SR) or if an interrupt by an on-chip peripheral module is disabled at the peripheral module.

Cancellation by a DMAC Address Error: If a DMAC address error occurs, the sleep mode is canceled and DMAC address error exception processing is executed.

Cancellation by a Power-On Reset: A power-on reset resulting from setting the RES pin to low level cancels the sleep mode.

18.4 Standby Mode

18.4.1 Transition to Standby Mode

To enter the standby mode, set the SBY bit to 1 in SBYCR, then execute the SLEEP instruction. The LSI moves from the program execution state to the standby mode. In the standby mode, power consumption is greatly reduced by halting not only the CPU, but the clock and on-chip peripheral modules as well. CPU register contents and on-chip RAM data are held as long as the prescribed voltages are applied. The register contents of some on-chip peripheral modules are initialized, but some are not (table 18.3). The I/O port status can be selected as held or high impedance by the port high impedance bit (HIZ) of the SBYCR. For pin status other than for the I/O port, refer to Appendix B, Pin Status.

Table 18.3 Register States in the Standby Mode

Module	Registers Initialized	Registers that Retain Data	Registers with Undefined Contents
Interrupt controller (INTC)	—	All registers	—
Cache memory (CAC)	—	All registers	—
Bus state controller (BSC)	—	All registers	—
Direct memory access controller (DMAC)	• DMA channel control registers 0 and 1 (CHCR0, CHCR1) • DMA operation register (DMAOR)	—	• DMA source address registers 0 and 1 (SAR0, SAR1) • DMA destination address registers 0 and 1 (DAR0, DAR1) • DMA transfer count registers 0 and 1 (DMATCR0, DMATCR1)
Multifunction timer pulse unit (MTU)	MTU associated registers	—	—
Watchdog timer (WDT)	• Bits 7 to 5 (OVF, WT/$\overline{IT}$, TME) of the timer control status register (TCSR) • Reset control/status register (RSTCSR)	• Bits 2 to 0 (CKS2 to CKS0) of the TCSR • Timer counter (TCNT)	—
Serial communication interface (SCI)	• Receive data register (RDR) • Transmit data register (TDR) • Serial mode register (SMR) • Serial control register (SCR) • Serial status register (SSR) • Bit rate register (BBR)	—	—
A/D converter (A/D)	All registers	—	—
Compare match timer (CMT)	All registers	—	—

Table 18.3 Register States in the Standby Mode (cont)

Module	Registers Initialized	Registers that Retain Data	Registers with Undefined Contents
Pin function controller (PFC)	—	All registers	—
I/O port (I/O)	—	All registers	—
Power-down state related	—	Standby control register (SBYCR)	—

18.4.2 Canceling the Standby Mode

The standby mode is canceled by an NMI interrupt or a power-on reset.

Cancellation by an NMI: Clock oscillation starts when a rising edge or falling edge (selected by the NMI edge select bit (NMIE) of the interrupt control register (ICR) of the INTC) is detected in the NMI signal. This clock is supplied only to the watchdog timer (WDT). A WDT overflow occurs if the time established by the clock select bits (CKS2 to CKS0) in the TCSR of the WDT elapses before transition to the standby mode. The occurrence of this overflow is used to indicate that the clock has stabilized, so the clock is supplied to the entire chip, the standby mode is canceled, and NMI exception processing begins.

When canceling standby mode with NMI interrupts, set the CKS2 to CKS0 bits so that the WDT overflow period is longer than the oscillation stabilization time.

When canceling standby mode with an NMI pin set for falling edge, be sure that the NMI pin level upon entering standby (when the clock is halted) is high level, and that the NMI pin level upon returning from standby (when the clock starts after oscillation stabilization) is low level. When canceling standby mode with an NMI pin set for rising edge, be sure that the NMI pin level upon entering standby (when the clock is halted) is low level, and that the NMI pin level upon returning from standby (when the clock starts after oscillation stabilization) is high level.

Cancellation by a Power-On Reset: A power-on reset caused by setting the $\overline{\text{RES}}$ pin to low level cancels the standby mode.

18.4.3 Standby Mode Application Example

This example describes a transition to standby mode on the falling edge of an NMI signal, and a cancellation on the rising edge of the NMI signal. The timing is shown in figure 18.1.

When the NMI pin is changed from high to low level while the NMI edge select bit (NMIE) of the ICR is set to 0 (falling edge detection), the NMI interrupt is accepted. When the NMIE bit is set to 1 (rising edge detection) by an NMI exception service routine, the standby bit (SBY) of the SBYCR is set to 1, and a SLEEP instruction is executed, standby mode is entered. Thereafter, standby mode is canceled when the NMI pin is changed from low to high level. After the NMI pin is changed to high level, this level should be held until NMI exception handling begins.

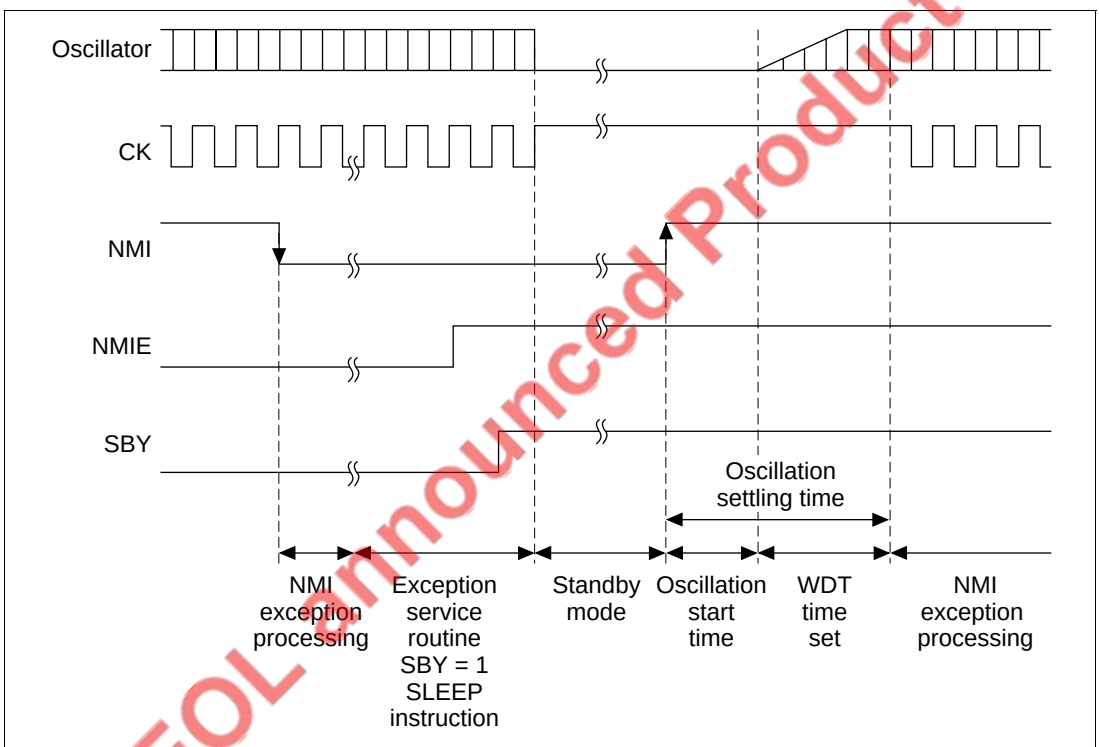


Figure 18.1 Standby Mode NMI Timing (Application Example)

Section 19 Electrical Characteristics

19.1 Absolute Maximum Ratings

The absolute maximum ratings are listed in table 19.1.

Table 19.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage	V_{CC}	–0.3 to +4.3	V
Input voltage (except A/D ports)	V_{in}	–0.3 to V_{CC}+0.3	V
Input voltage (A/D ports)	V_{in}	–0.3 to AV_{CC}+0.3	V
Analog power supply voltage	AV_{CC}	–0.3 to +4.3	V
Analog input voltage	V_{AN}	–0.3 to AV_{CC}+0.3	V
Operating temperature	T_{opr}	–20 to +75	°C
Write/Erase temperature	T_{we}	0 to +70	°C
Storage temperature	T_{stg}	–55 to +125	°C

Note: Permanent damage to the chip may result if the absolute maximum ratings are exceeded.

19.2 DC Characteristics

The DC characteristics are listed in table 19.2.

Table 19.2 DC Characteristics

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input high-level voltage	$\overline{RES}$, NMI	V_{IH}	PV_{CC} – 0.7	—	PV_{CC} + 0.3	V	
	FWP		V_{CC} – 0.3	—	V_{CC} + 0.3	V	
	EXTAL		$V_{CC} \times 0.9$	—	V_{CC} + 0.3	V	
	A/D ports		$V_{CC} \times 0.75$	—	AV_{CC} + 0.3	V	
	PD0 to PD7, PA3, PA4, PB8		2.2		PV_{CC} + 0.3	V	
	Other input pins (except for Schmitt trigger)		$V_{CC} \times 0.75$	—	V_{CC} + 0.3	V	
Input low-level voltage	$\overline{RES}$, NMI	V_{IL}	–0.3		0.5	V	
	FWP		–0.3	—	$V_{CC} \times 0.1$	V	
	PD0 to PD7, PA3, PA4, PB8		–0.3	—	0.8	V	
	Other input pins		–0.3	—	$V_{CC} \times 0.2$	V	
Schmitt-trigger input voltage	PA2, PA5, PE0, PE2	V_T^+	4.0	—	—	V	
		V_T^-		—	1.0	V	
		$V_T^+ – V_T^-$	0.4	—	—	V	
	PA6 to PA9, PE4 to PE14	$V_T^+ – V_T^-$	0.2	—	—	V	

Table 19.2 DC Characteristics (cont)

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input leakage current	$\overline{RES}$, NMI, FWP, PA2, PA5 to PA9, PE0, PE2, PE4 to PE14	$\|I_{in}\|$	—	—	1.0	µA	V_{in} = 0.5 to V_{CC}–0.5V
	A/D ports		—	—	1.0	µA	V_{in} = 0.5 to AV_{CC}–0.5V
	Other input pins		—	—	1.0	µA	V_{in} = 0.5 to V_{CC}–0.5V
Three-state leakage current (off state)	A21 to A0, D7 to D0, $\overline{CS3}$ to $\overline{CS0}$, $\overline{WRL}$, $\overline{RD}$, ports A and E	$\|I_{TSI}\|$	—	—	1.0	µA	V_{in} = 0.5 to V_{CC}–0.5V
Output high-level voltage	PE0, PE2, PD0 to PD7, PA2 to PA5, PB8	V_{OH}	PV_{CC} – 0.7	—	—	V	I_{OH} = –200 µA
			PV_{CC} – 1.0	—	—	V	I_{OH} = –1mA
	Other output pins		V_{CC} – 0.7	—	—	V	I_{OH} = –200 µA
			V_{CC} – 1.0	—	—	V	I_{OH} = –1mA
Output low-level voltage	PE0, PE2, PD0 to PD7, PA2 to PA5, PB8	V_{OL}	—	—	0.4	V	I_{OL} = 1.6mA
	Other output pins		—	—	0.6	V	I_{OL} = 1.6mA

Table 19.2 DC Characteristics (cont)

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input capacitance	$\overline{RES}$	C_{in}	—	—	80	pF	V_{in} = 0 V
	NMI		—	—	50	pF	f = 1 MHz
	All other input pins		—	—	20	pF	T_a = 25°C
Current consumption	Normal operation	I_{CC}	—	80	110	mA	f = 20 MHz
	Sleep mode		—	70	95	mA	f = 20 MHz
	Standby mode		—	5	50	µA	$T_a \leq$ 50°C
			—	—	300	µA	50°C < T_a
Analog power supply current		AI_{CC}	—	5	10	mA	f = 20 MHz

Notes: 1. If the A/D converter is not used, do not leave the AV_{CC} and AV_{SS} pins open. Connect the AV_{CC} pin to V_{CC} and the AV_{SS} pin to V_{SS}.

2. Current consumption values are for conditions of V_{IH}min = V_{CC} –0.5 V and V_{IL}min = 0.5 V with all output pins unloaded.

Table 19.3 Permissible Output Current Values

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item	Symbol	Min	Typ	Max	Unit
Output low-level permissible current (per pin)	I_{OL}	—	—	2.0	mA
Output low-level permissible current (total)	$\sum I_{OL}$	—	—	80	mA
Output high-level permissible current (per pin)	$-I_{OH}$	—	—	2.0	mA
Output high-level permissible current (total)	$\sum(-I_{OH})$	—	—	25	mA

Note: To ensure chip reliability, do not exceed the output current values in table 19.3.

19.3 AC Characteristics

19.3.1 Clock Timing

The clock timing is shown in table 19.4.

Table 19.4 Clock Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item	Symbol	Min	Max	Unit	Figure
Operating frequency	f_{OP}	4	20	MHz	Figure 19.1
Clock cycle time	t_{cyc}	50	250	ns	
Clock low-level pulse width	t_{CL}	15	—	ns	
Clock high-level pulse width	t_{CH}	15	—	ns	
Clock rise time	t_{Cr}	—	5	ns	
Clock fall time	t_{Cf}	—	5	ns	
EXTAL clock input frequency	f_{EX}	4	20	MHz	Figure 19.2
EXTAL clock input cycle time	t_{EXcyc}	50	250	ns	
EXTAL clock input low-level pulse width	t_{EXL}	17.5	—	ns	
EXTAL clock input high-level pulse width	t_{EXH}	17.5	—	ns	
EXTAL clock input rise time	t_{EXr}	—	5	ns	
EXTAL clock input fall time	t_{EXf}	—	5	ns	
Reset oscillation settling time	t_{OSC1}	10	—	ms	Figure 19.3
Standby recovery oscillation settling time	t_{OSC2}	10	—	ms	

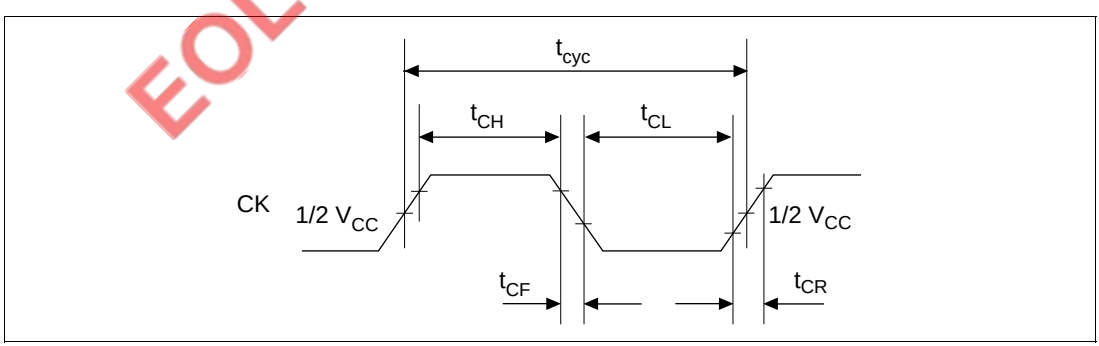


Figure 19.1 System Clock Timing

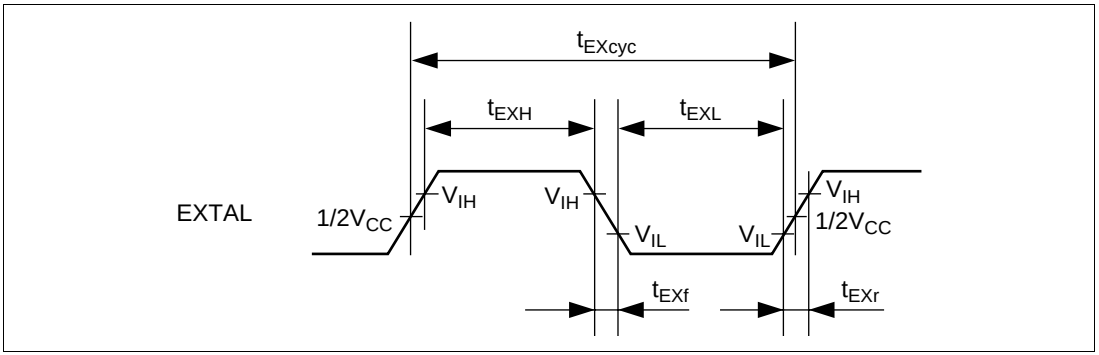


Figure 19.2 EXTAL Clock Input Timing

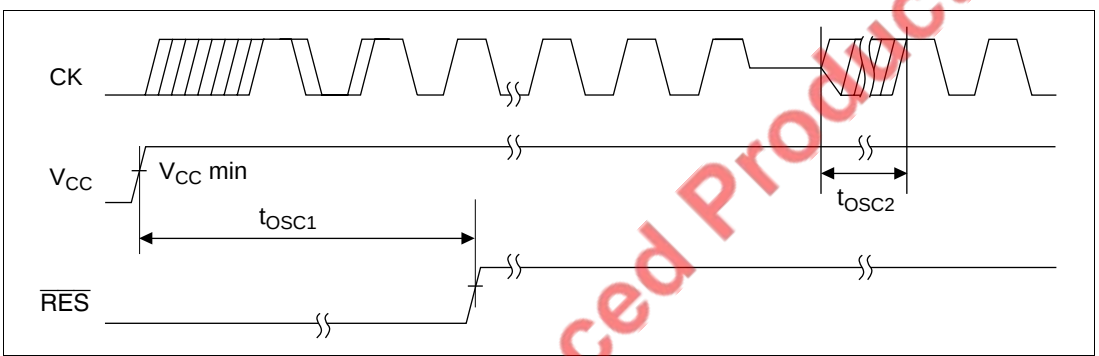


Figure 19.3 Oscillation Settling Time

19.3.2 Control Signal Timing

Table 19.5 Control Signal Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item	Symbol	Min	Max	Unit	Figure
$\overline{\text{RES}}$ rise and fall	t_{RESr}, t_{RESf}	—	200	ns	Figure 19.4
$\overline{\text{RES}}$ pulse width	t_{RESW}	40	—	t_{cyc}	
NMI rise and fall	t_{NMIr}, t_{NMIf}	—	200	ns	
$\overline{\text{RES}}$ setup time*	t_{RESS}	35	—	ns	Figure 19.4, Figure 19.5
NMI setup time*	t_{NMIS}	35	—	ns	
$\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}$ to $\overline{\text{IRQ0}}$ setup time* (edge detection)	t_{IRQES}	35	—	ns	
$\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}$ to $\overline{\text{IRQ0}}$ setup time* (level detection)	t_{IRQLS}	35	—	ns	
NMI hold time	t_{NMIH}	35	—	ns	Figure 19.5
$\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}$ to $\overline{\text{IRQ0}}$ hold time	t_{IRQEH}	35	—	ns	

Note: * The $\overline{\text{RES}}$, NMI, $\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}$ to $\overline{\text{IRQ0}}$ signals are input asynchronously. If the setup time indicated is maintained, the clock rise (in the case of $\overline{\text{RES}}$) or fall (in the case of NMI, $\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}$ to $\overline{\text{IRQ0}}$) will be recognized as a change in level. If the setup time indicated is not maintained, the change may not be recognized until the next clock rise or fall.

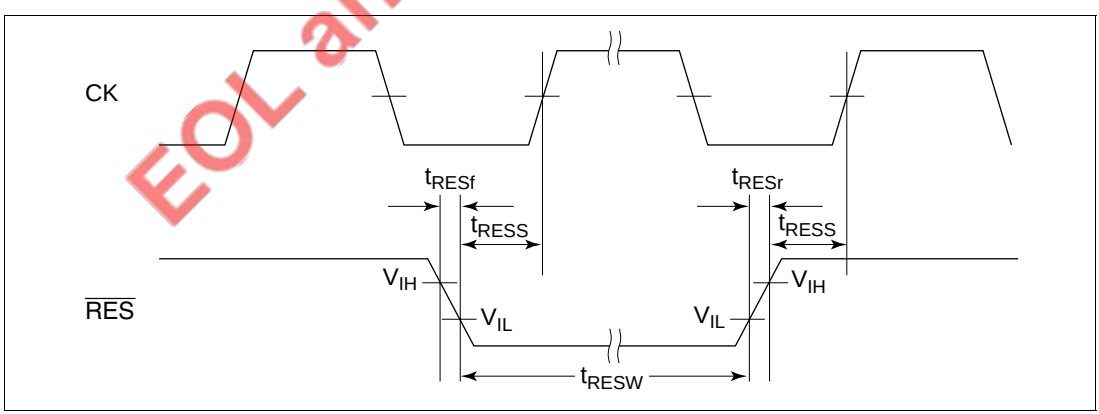


Figure 19.4 Reset Input Timing

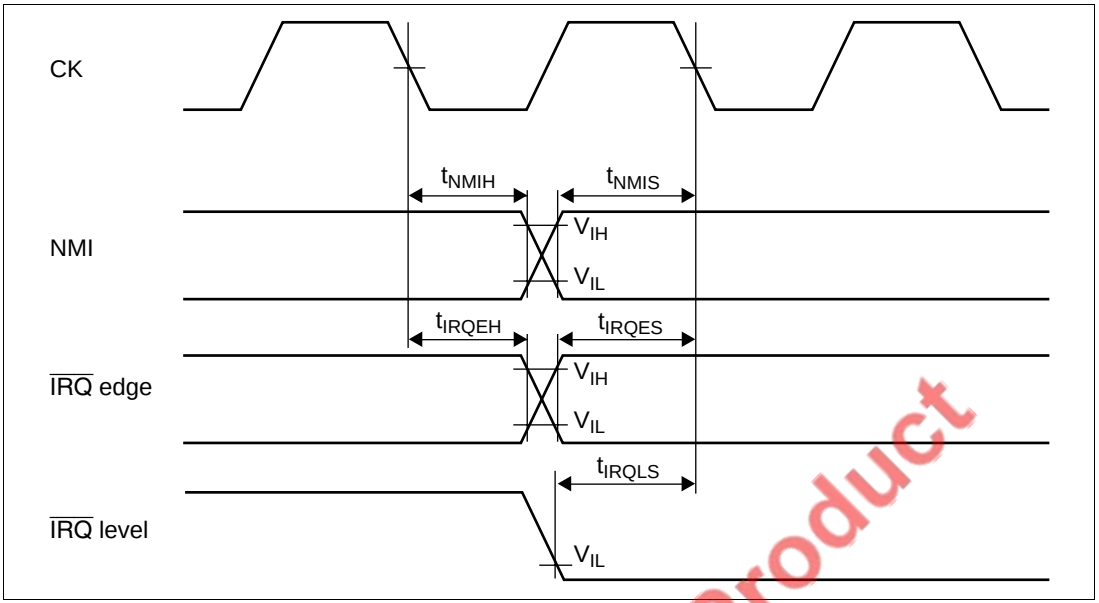


Figure 19.5 Interrupt Signal Input Timing

19.3.3 Bus Timing

Table 19.6 Bus Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

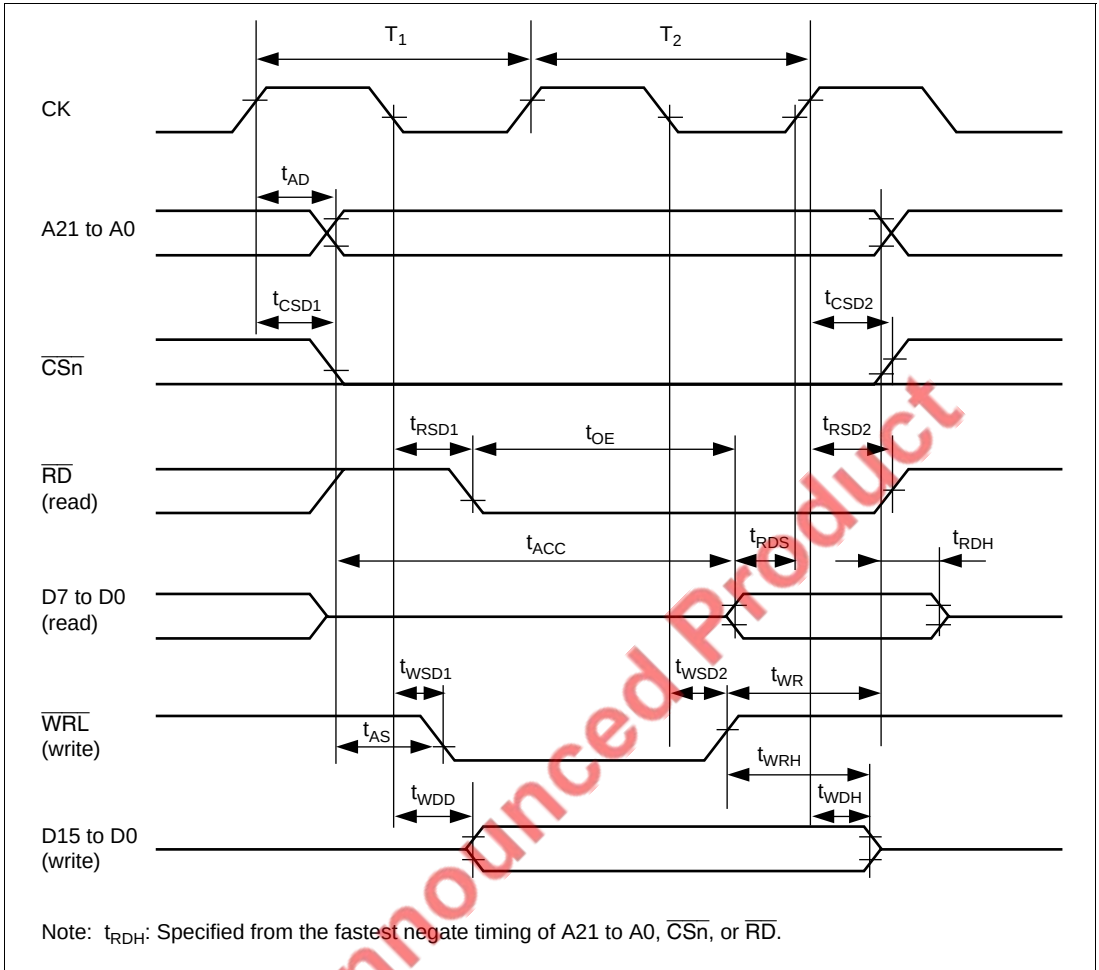
Item	Symbol	Min	Max	Unit	Figure
Address delay time	t_{AD}	3*3	25	ns	Figure 19.6 to figure 19.8
$\overline{CS}$ delay time 1	t_{CSD1}	3*3	21	ns	Figure 19.6 to figure 19.8
$\overline{CS}$ delay time 2	t_{CSD2}	3*3	21	ns	Figure 19.6 to figure 19.8
Read strobe delay time 1	t_{RSD1}	3*3	18	ns	Figure 19.6 to figure 19.8
Read strobe delay time 2	t_{RSD2}	3*3	18	ns	Figure 19.6 to figure 19.8
Read data setup time	t_{RDS}*4	20	—	ns	Figure 19.6 to figure 19.8
Read data hold time	t_{RDH}	0	—	ns	Figure 19.6 to figure 19.8
Write strobe delay time 1	t_{WSD1}	3*3	18	ns	Figure 19.6 to figure 19.8
Write strobe delay time 2	t_{WSD2}	3*3	18	ns	Figure 19.6 to figure 19.8
Write data delay time	t_{WDD}	—	35	ns	Figure 19.6 to figure 19.8
Write data hold time	t_{WDH}	0	20*2	ns	Figure 19.6 to figure 19.8
$\overline{WAIT}$ setup time	t_{WTS}	20	—	ns	Figure 19.6 to figure 19.8
$\overline{WAIT}$ hold time	t_{WTH}	0	—	ns	Figure 19.6 to figure 19.8

Table 19.6 Bus Timing (cont)

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

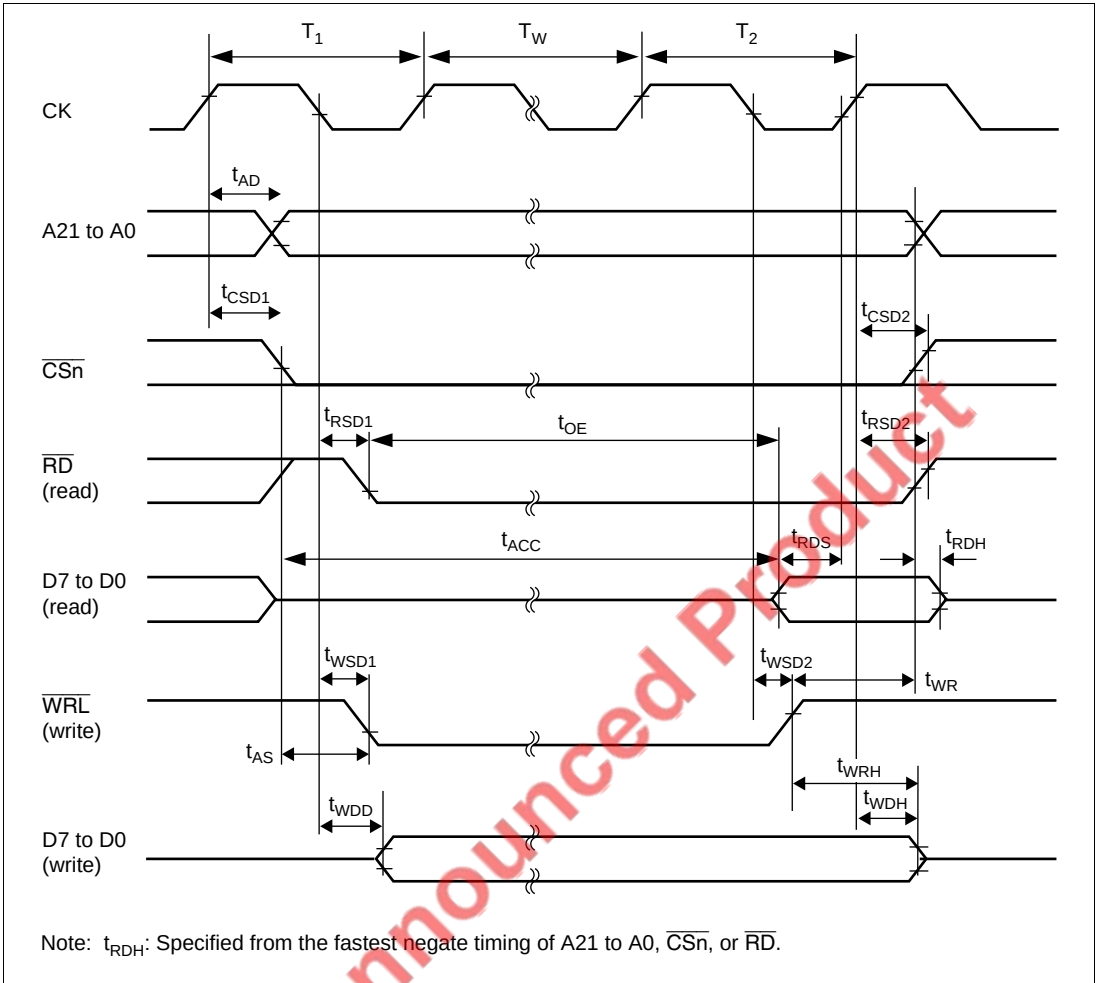
Item	Symbol	Min	Max	Unit	Figure
Read data access time	t_{ACC} *1	$t_{cyc} \times (n$ *5 $+ 2) - 50$	—	ns	Figure 19.6 to figure 19.8
Access time from read strobe	t_{OE} *1	$t_{cyc} \times (n$ *5 $+ 1.5) - 50$	—	ns	Figure 19.6 to figure 19.8
Write address setup time	t_{AS}	0	—	ns	Figure 19.6 to figure 19.8
Write address hold time	t_{WR}	0	—	ns	Figure 19.6 to figure 19.8
Write data hold time	t_{WRH}	0	—	ns	Figure 19.6 to figure 19.8

Notes:
1. The t_{RDS} specification needs not be met as long as the access time specification is met.
2. t_{WDH} (max) is a reference value.
3. The minimum (min) values for delay times are reference (typical) values.
4. t_{RDS} is a reference value.
5. The n is the wait number.



Note: t_{RDH}: Specified from the fastest negate timing of A21 to A0, $\overline{CSn}$, or $\overline{RD}$.

Figure 19.6 Basic Cycle (No Wait)



Note: t_{RDH}: Specified from the fastest negate timing of A21 to A0, $\overline{CSn}$, or $\overline{RD}$.

Figure 19.7 Basic Cycle (Software Wait)

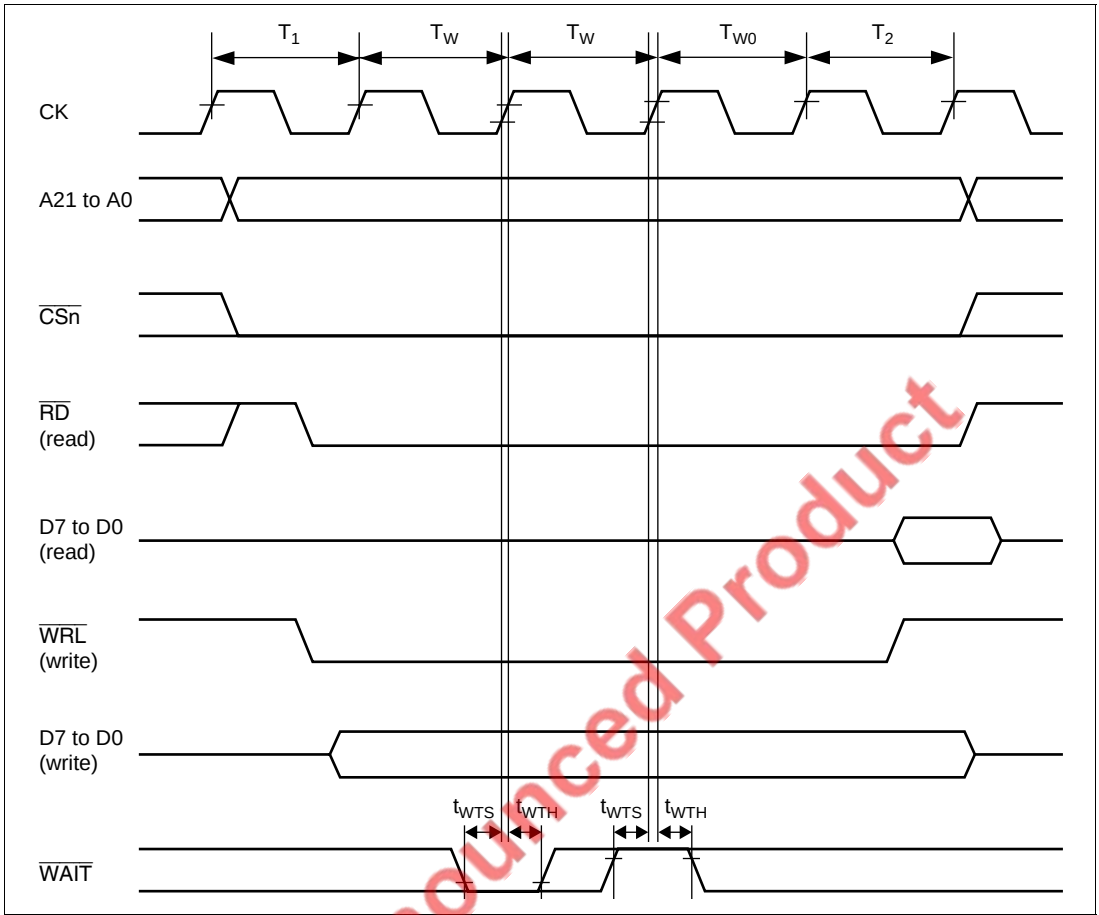


Figure 19.8 Basic Cycle (Wait Using Two Software Waits + $\overline{\text{WAIT}}$ Signal)

19.3.4 Multifunction Timer Pulse Unit Timing

Table 19.7 shows the multifunction timer pulse unit timing.

Table 19.7 Multifunction Timer Pulse Unit Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item	Symbol	Min	Max	Unit	Figure
Output compare output delay time	t_{TOCD}	—	100	ns	Figure 19.9
Input capture input setup time	t_{TICS}	30	—	ns	

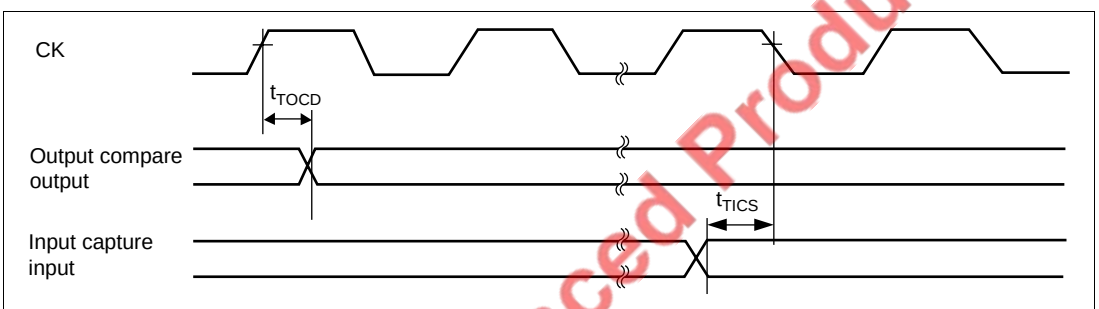


Figure 19.9 MTU Input/Output Timing

19.3.5 I/O Port Timing

Table 19.8 shows the I/O port timing.

Table 19.8 I/O Port Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item	Symbol	Min	Max	Unit	Figure
Port output data delay time	t_{PWD}	—	100	ns	Figure 19.10
Port input hold time	t_{PRH}	100	—	ns	
Port input setup time	t_{PRS}	100	—	ns	

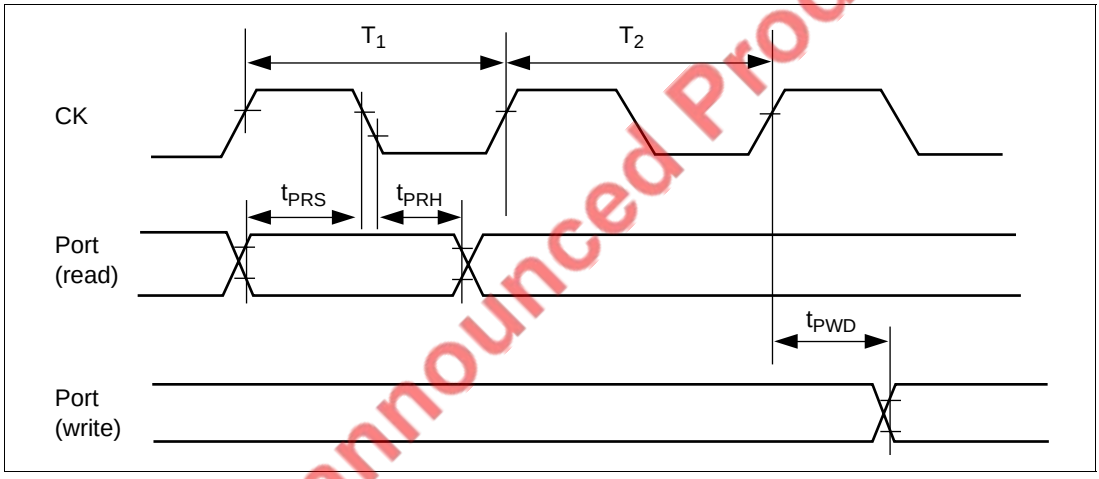


Figure 19.10 I/O Port Input/Output Timing

19.3.6 Serial Communication Interface Timing

Table 19.9 shows the serial communication interface timing.

Table 19.9 Serial Communication Interface Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ± 0.5 V, $PV_{CC} \geq V_{CC}$, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V, f = 20 MHz, T_a = –20 to +75°C

Item	Symbol	Min	Max	Unit	Figure
Input clock cycle	t_{Scyc}	4	—	t_{cyc}	Figure 19.11
Input clock cycle (synchronous)	t_{Scyc}	6	—	t_{cyc}	
Input clock pulse width	t_{Sckw}	0.4	0.6	t_{Scyc}	
Input clock rise time	t_{Sckr}	—	1.5	t_{cyc}	
Input clock fall time	t_{Sckf}	—	1.5	t_{cyc}	
Transmit data delay time (synchronous)	t_{TXD}	—	100	ns	Figure 19.12
Receive data setup time (synchronous)	t_{RXS}	100	—	ns	
Receive data hold time (synchronous)	t_{RXH}	100	—	ns	

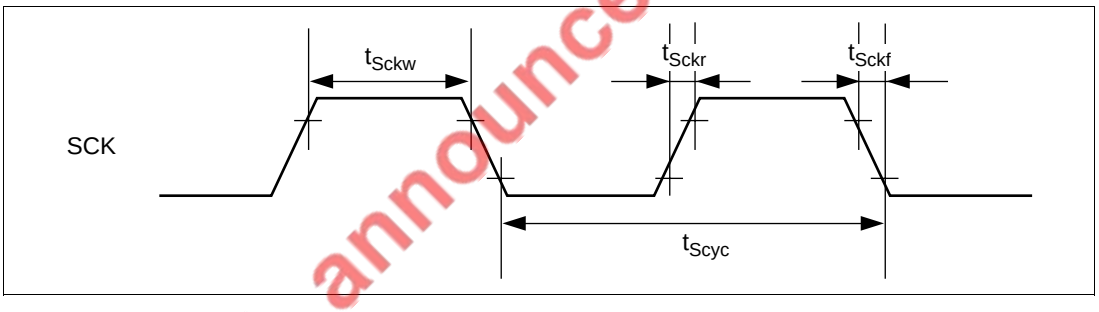


Figure 19.11 Input Clock Timing

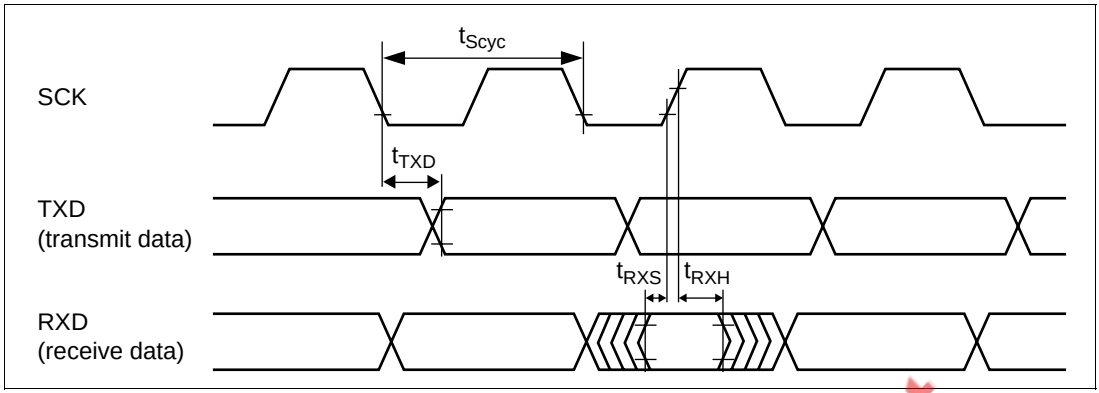


Figure 19.12 SCI Input/Output Timing (Synchronous Mode)

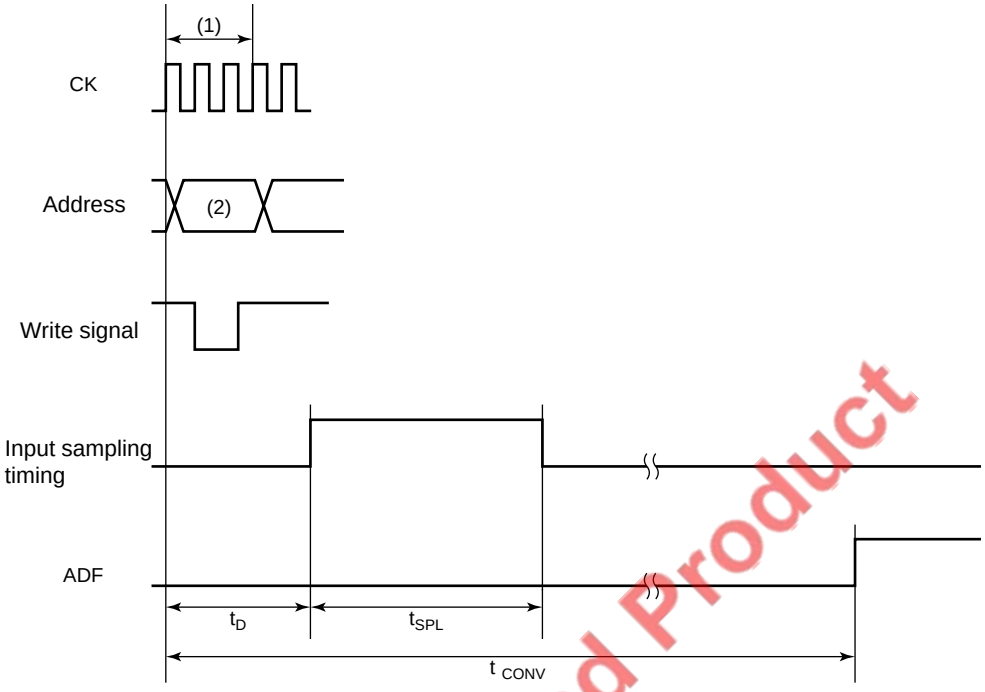
19.3.7 A/D Converter Timing

Table 19.10 shows the A/D converter timing.

Table 19.10 A/D Converter Timing

Conditions: V_{CC} = 3.0 to 3.6 V, PV_{CC} = 5.0 ±0.5 V, $PV_{CC} \geq V_{CC}$, $AV_{CC} \geq V_{CC}$, f = 20 MHz,
AV_{CC} = 3.0 to 3.6 V, $V_{SS} = AV_{SS}$ = 0V, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit	Figure
A/D conversion start delay time	CKS = 0	t_D	10	–	17	t_{cyc}	Figure 19.13
	CKS = 1		6	–	9		
Input sampling time	CKS = 0	t_{SPL}	–	64	–		
	CKS = 1		–	32	–		
A/D conversion time	CKS = 0	t_{CONV}	259	–	266		
	CKS = 1		131	–	134		



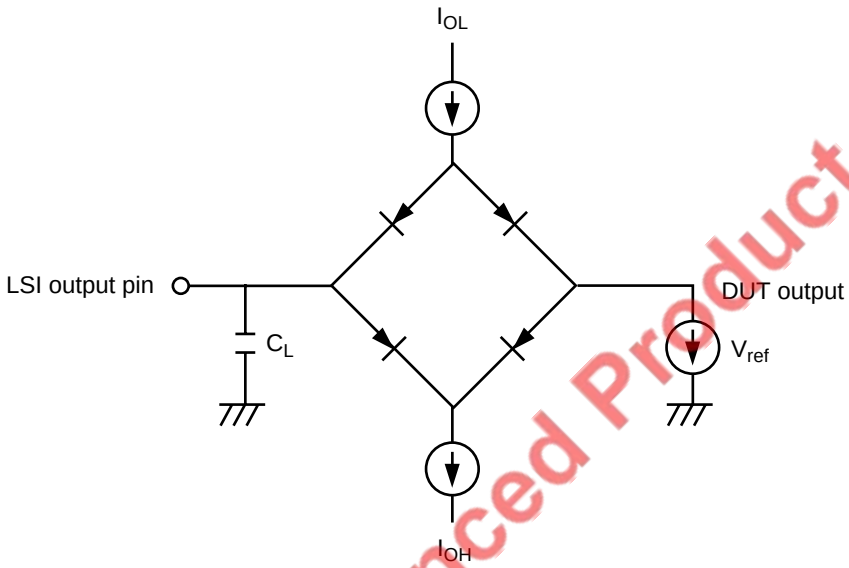
(1): Write cycle of ADCSR
(2): Address of ADCSR
t_D: A/D conversion start delay time
t_{SPL}: Input sampling time
t_{CONV}: A/D conversion time

Figure 19.13 Analog Conversion Timing

19.3.8 Test Conditions for AC Characteristics

Input Reference Levels: High: 2.2 V, low: 0.8 V

Output Reference Levels: High: 2.0 V, low: 0.8 V



CL is the total value including the capacitance of the testing jig. The settings for the different pins are as follows.

30 pF: CK, $\overline{CS0}$ to $\overline{CS3}$
50 pF: A21 to A0, D7 to D0, $\overline{RD}$, $\overline{WRL}$
70 pF: Port output pins other than those listed above and peripheral module output pins.

I_{OL}, I_{OH}: Capacitance values are listed in section 19.2, DC Characteristics and table 19.3, Permissible Output Current Values.

Figure 19.14 Output Load Circuit

19.4 A/D Converter Characteristics

Table 19.11 A/D Converter Characteristics

Conditions: V_{CC} = 3.0 to 3.6 V, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{SS} = AV_{SS}$ = 0 V,
Ta = –20 to +75°C, CKS = 0

	20.0MHz			
Item	**Min**	**Typ**	**Max**	**Unit**
Resolution	10	10	10	Bits
Conversion time	—	—	13.4	µs
Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	1	kΩ
Nonlinearity error	—	—	±5*	LSB
Offset error	—	—	±5*	LSB
Full-scale error	—	—	±5*	LSB
Quantization error	—	—	±0.5*	LSB
Absolute accuracy	—	—	±6	LSB

Note: * Reference values

Table 19.12 A/D Converter Characteristics

Conditions: V_{CC} = 3.0 to 3.6 V, AV_{CC} = 3.0 to 3.6 V, $AV_{CC} \geq V_{CC}$, $V_{ss} = AV_{ss}$ = 0 V,
Ta = –20 to +75°C, CKS = 1

	20MHz			
Item	**Min**	**Typ**	**Max**	**Unit**
Resolution	10	10	10	Bits
Conversion time	—	—	6.7	µs
Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	1	kΩ
Nonlinearity error	—	—	±5*	LSB
Offset error	—	—	±5*	LSB
Full-scale error	—	—	±5*	LSB
Quantization error	—	—	±0.5*	LSB
Absolute accuracy	—	—	±6	LSB

Note: * Reference values

Appendix A On-Chip Peripheral Module Registers

Table A.1 On-Chip Peripheral Module Registers

Address	Register	Bit Names									
H'FFFFxxxx	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	
81B0	SMR1	C/$\overline{A}$	CHR	PE	O/E	STOP	MP	CKS1	CKS0	SCI1	
81B1	BRR1										
81B2	SCR1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0		
81B3	TDR1										
81B4	SSR1	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT		
81B5	RDR1										
8240	TSTR	—	—	—	—	—	CST2	CST1	CST0	All	MTU
8241	TSYR	—	—	—	—	—	SYNC2	SYNC1	SYNC0		
8260	TCR0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	ch0	
8261	TMDR0	—	—	BFB	BFA	MD3	MD2	MD1	MD0		
8262	TIOR0H	—	IOB2	IOB2	IOB1	IOA3	IOA2	IOA1	IOA0		
8263	TIOR0L	—	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0		
8264	TIER0	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA		
8265	TSR0	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA		
8266	TCNT0										
8267											
8268	TGR0A										
8269											
826A	TGR0B										
826B											
826C	TGR0C										
826D											
826E	TGR0D										
826F											
8280	TCR1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	ch1	
8281	TMDR1	—	—	—	—	MD3	MD2	MD1	MD0		
8282	TIOR1	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0		
8284	TIER1	TTGE	—	—	TCIEV	—	—	TGIEB	TGIEA		
8285	TSR1	—	—	—	TCFV	—	—	TGFB	TGFA		
8286	TCNT1										
8287											
8288	TGR1A										
8289											

Table A.1 On-Chip Peripheral Module Registers (cont)

Address H'FFFFxxxx	Register Name	Bit Names Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	
828A	TGR1B									ch1	MTU
828B											
82A0	TCR2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	ch2	
82A1	TMDR2	—	—	—	—	MD3	MD2	MD1	MD0		
82A2	TIOR2	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0		
82A4	TIER2	TTGE	—	—	TCIEV	—	—	TGIEB	TGIEA		
82A5	TSR2	—	—	—	TCFV	—	—	TGFB	TGFA		
82A6	TCNT2										
82A7											
82A8	TGR2A										
82A9											
82AA	TGR2B										
82AB											
8348	IPRA	(IRQ0)	(IRQ0)	(IRQ0)	(IRQ0)	(IRQ1)	(IRQ1)	(IRQ1)	(IRQ1)	INTC	
8349		(IRQ2)	(IRQ2)	(IRQ2)	(IRQ2)	(IRQ3)	(IRQ3)	(IRQ3)	(IRQ3)		
834A	IPRB	—	—	—	—	—	—	—	—		
834B		(IRQ6)	(IRQ6)	(IRQ6)	(IRQ6)	(IRQ7)	(IRQ7)	(IRQ7)	(IRQ7)		
834C	IPRC	—	—	—	—	—	—	—	—		
834D		—	—	—	—	—	—	—	—		
834E	IPRD	(MTU0)	(MTU0)	(MTU0)	(MTU0)	(MTU0)	(MTU0)	(MTU0)	(MTU0)		
834F		(MTU1)	(MTU1)	(MTU1)	(MTU1)	(MTU1)	(MTU1)	(MTU1)	(MTU1)		
8350	IPRE	(MTU2)	(MTU2)	(MTU2)	(MTU2)	(MTU2)	(MTU2)	(MTU2)	(MTU2)		
8351		—	—	—	—	—	—	—	—		
8352	IPRF	—	—	—	—	—	—	—	—		
8353		—	—	—	—	(SCI)	(SCI)	(SCI)	(SCI)		
8354	IPRG	(A/D)	(A/D)	(A/D)	(A/D)	—	—	—	—		
8355		(CMT0)	(CMT0)	(CMT0)	(CMT0)	(CMT1)	(CMT1)	(CMT1)	(CMT1)		
8356	IPRH	(WDT)	(WDT)	(TIM2)	(TIM2)	—	—	—	—		
8357		—	—	—	—	—	—	—	—		
8358	ICR	NMIL	—	—	—	—	—	—	NMIE		
8359		IRQ0S	IRQ1S	IRQ2S	IRQ3S	—	—	IRQ6S	IRQ7S		
835A	ISR	—	—	—	—	—	—	—	—		
835B		IRQ0F	IRQ1F	IRQ2F	IRQ3F	—	—	IRQ6F	IRQ7F		
8382	PADRL	PA15DR	PA14DR	—	PA12DR	PA11DR	PA10DR	PA9DR	PA8DR	I/O	Port A
8383		PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR		

Table A.1 On-Chip Peripheral Module Registers (cont)

Address H'FFFFxxxx	Register Name	Bit Names Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	
8386	PAIORL	PA15IOR	PA14IOR	—	PA12IOR	PA11IOR	PA10IOR	PA9IOR	PA8IOR	PFC	Port A
8387		PA7IOR	PA6IOR	PA5IOR	PA4IOR	PA3IOR	PA2IOR	PA1IOR	PA0IOR		
838C	PACRL1	—	PA15MD	—	PA14MD	—	—	—	PA12MD		
838D		—	PA11MD	—	PA10MD	PA9MD1	PA9MD0	PA8MD1	PA8MD0		
838E	PACRL2	PA7MD1	PA7MD0	PA6MD1	PA6MD0	—	PA5MD	—	PA4MD		
838F		—	PA3MD	PA2MD1	PA2MD0	—	—	—	—		
8390	PBDR	—	—	—	—	—	—	PB9DR	PB8DR	I/O	Port B
8391		PB7DR	PB6DR	PB5DR	PB4DR	PB3DR	PB2DR	PB1DR	PB0DR		
8394	PBIOR	—	—	—	—	—	—	PB9IOR	PB8IOR	PFC	
8395		PB7IOR	PB6IOR	PB5IOR	PB4IOR	PB3IOR	PB2IOR	PB1IOR	PB0IOR		
8398	PBCR1	—	—	—	—	—	—	—	—		
8399		—	—	—	—	PB9MD1	PB9MD0	PB8MD1	PB8MD0		
839A	PBCR2	PB7MD1	PB7MD0	PB6MD1	PB6MD0	—	—	—	—		
839B		—	PB3MD	—	—	—	PB1MD	—	PB0MD		
8392	PCDR	PC15DR	PC14DR	PC13DR	PC12DR	PC11DR	PC10DR	PC9DR	PC8DR	I/O	Port C
8393		PC7DR	PC6DR	PC5DR	PC4DR	PC3DR	PC2DR	PC1DR	PC0DR		
8396	PCIOR	PC15IOR	PC14IOR	PC13IOR	PC12IOR	PC11IOR	PC10IOR	PC9IOR	PC8IOR	PFC	
8397		PC7IOR	PC6IOR	PC5IOR	PC4IOR	PC3IOR	PC2IOR	PC1IOR	PC0IOR		
839C	PCCR	PC15MD	PC14MD	PC13MD	PC12MD	PC11MD	PC10MD	PC9MD	PC8MD		
839D		PC7MD	PC6MD	PC5MD	PC4MD	PC3MD	PC2MD	PC1MD	PC0MD		
83A2	PDDRL	—	—	—	—	—	—	—	—	I/O	Port D
83A3		PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR		
83A6	PDIORL	PD15IOR	PD14IOR	PD13IOR	PD12IOR	PD11IOR	PD10IOR	PD9IOR	PD8IOR	TFC	
83A7		PD7IOR	PD6IOR	PD5IOR	PD4IOR	PD3IOR	PD2IOR	PD1IOR	PD0IOR		
83AC	PDCRL	—	—	—	—	—	—	—	—		
83AD		PD7MD	PD6MD	PD5MD	PD4MD	PD3MD	PD2MD	PD1MD	PD0MD		
83B0	PEDR	—	PE14DR	PE13DR	PE12DR	—	—	—	—	I/O	Port E
83B1		PE7DR	PE6DR	PE5DR	PE4DR	—	PE2DR	—	PE0DR		
83B3	PFDR	PF7DR	PF6DR	PF5DR	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR	I/O	Port F
83B4	PEIOR	PE15IOR	PE14IOR	PE13IOR	PE12IOR	PE11IOR	PE10OR	PE9IOR	PE8IOR	PFC	Port E
83B5		PE7IOR	PE6IOR	PE5IOR	PE4IOR	PE3IOR	PE2IOR	PE1IOR	PE0IOR		
83BA	PECR2	—	PE7MD	—	PE6MD	—	PE5MD	—	PE4MD		
83BB		—	—	—	PE2MD0	—	—	—	PE0MD0		
83D0	CMSTR	—	—	—	—	—	—	—	—	All	CMT
83D1		—	—	—	—	—	—	STR1	STR0		

Table A.1 On-Chip Peripheral Module Registers (cont)

Address H'FFFFxxxx	Register Name	Bit Names Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
83D2	CMCSR0	—	—	—	—	—	—	—	—	ch0 CMT
83D3		CMF	CMIE	—	—	—	—	CKS1	CKS0	
83D4	CMCNT0									
83D5										
83D6	CMCOR0									
83D7										
83D8	CMCSR1	—	—	—	—	—	—	—	—	ch1
83D9		CMF	CMIE	—	—	—	—	CKS1	CKS0	
83DA	CMCNT1									
83DB										
83DC	CMCOR1									
83DD										
8420	ADDRAH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D
8421	ADDRAL	AD1	AD0	—	—	—	—	—	—	
8422	ADDRBH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
8423	ADDRBL	AD1	AD0	—	—	—	—	—	—	
8424	ADDRCH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
8425	ADDRCL	AD1	AD0	—	—	—	—	—	—	
8426	ADDRDH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
8427	ADDRDL	AD1	AD0	—	—	—	—	—	—	
8428	ADCSR	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0	
8429	ADCR	TRGE	—	—	—	—	—	—	—	
8580	FLMCR1	FWE	SWE	ESU	PSU	EV	PV	E	P	FLASH
8581	FLMCR2	FLER	—	—	—	—	—	—	—	
8582	EBR1	EB7	EB6	EB5	EB4	EB3	EB2	EB1	EB0	
8583	EBR2	—	—	—	—	—	EB10	EB9	EB8	
8610	TCSR	OVF	WI/$\overline{IT}$	TME	—	—	CKS2	CKS1	CKS0	WDT
8611	TCNT									
8612	RSTCSR	WOVF	RSTE	—	—	—	—	—	—	
8613	RSTCSR	WOVF	RSTE	—	—	—	—	—	—	
8614	SBYCR	SBY	HIZ	—	—	—	—	—	—	Power-Down State
8620	BCR1	—	—	—	—	—	—	—	—	BSC
8621		—	—	—	—	A3SZ	A2SZ	A1SZ	A0SZ	
8622	BCR2	IW31	IW30	IW21	IW20	IW11	IW10	IW01	IW00	
8623		CW3	CW2	CW1	CW0	SW3	SW2	SW1	SW0	

Table A.1 On-Chip Peripheral Module Registers (cont)

Address H'FFFFxxxx	Register Name	Bit Names Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
8624	WCR1	—	—	W31	W30	—	—	W21	W20	BSC
8625		—	—	W11	W10	—	—	W01	W00	
8628	RAMER	—	—	—	—	—	—	—	—	
8629		—	—	—	—	—	RAMS	RAM1	RAM0	
862C	T2CSR	—	—	—	—	—	—	—	—	TIM2
862D	—	CMF	CMIE	CKS2	CKS1	CKS0	—	—		
862E	T2CNT	—	—	—	—	—	—	—	—	
862F										
8630	T2COR	—	—	—	—	—	—	—	—	
8631										

Appendix B Pin States

B.1 Pin States

Table B.1 Pin States after Reset and in Power-Down State

Pin Function		Pin State	
		After Reset	Power-Down State
Type	Pin Name	Power-On	Sleep
Clock	CK	O	O
System control	$\overline{\text{RES}}$	I	I
Interrupt	NMI	I	I
	$\overline{\text{IRQ0}}$ to $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$	Z	I
Address bus	A0 to A21	O	O
Data bus	D0 to D7	Z	I/O
Bus control	$\overline{\text{WAIT}}$	Z	I
	$\overline{\text{RD}}$	H	H
	$\overline{\text{CS0}}$ to $\overline{\text{CS3}}$	H	H
	$\overline{\text{WRH}}$, $\overline{\text{WRL}}$	H	H
MTU	TIOC0A, TIOC0C	Z	I/O
	TIOC1A, TIOC1B		
	TIOC2A, TIOC2B		
SCI	TxD	Z	O
	RxD	Z	I
A/D converter	AN0 to AN7	Z	I
I/O ports	PA0 to PA15	Z	K
	PB0 to PB9		
	PC0 to PC15		
	PD0 to PD7		
	PE0, PE2, PE4 to PE14		

Legend: I: input, O: output, H: high-level output, L: low-level output,
Z: high impedance, K: high impedance for input pins and maintain status for output pins

B.2 Bus-Related Signals and Pin States

Table B.2 Bus-Related Signals and Pin States

Pin Name		On-Chip RAM Space	On-Chip Peripheral Modules: 8-Bit Space	On-Chip Peripheral Modules: 16-Bit Space: Upper Bytes	On-Chip Peripheral Modules: 16-Bit Space: Lower Bytes	On-Chip Peripheral Modules: 16-Bit Space: Word/ Longword	Normal External Space: 16-Bit Space: Upper Bytes	Normal External Space: 16-Bit Space: Lower Bytes	Normal External Space: 16-Bit Space: Word/ Longword
$\overline{CS0}$ to $\overline{CS3}$		H	H	H	H	H	Enabled	Enabled	Enabled
$\overline{RD}$	R	H	H	H	H	H	L	L	L
	W	H	H	H	H	H	H	H	H
$\overline{WRH}$	R	H	H	H	H	H	H	H	H
	W	H	H	H	H	H	L	H	L
$\overline{WRL}$	R	H	H	H	H	H	H	H	H
	W	H	H	H	H	H	H	L	L
A21 to A0		Address	Address	Address	Address	Address	Address	Address	Address
D7 to D0		Z	Z	Z	Z	Z	Z	Data	Data

Legend: R: read, W: write, H: high-level output, L: low-level output,
Z: high impedance, Enabled: chip select signal corresponding to area being accessed is low-level, otherwise chip select signal is high-level

Appendix C Product Lineup

Table C.1 SH7018 Product Lineup

Product Name	Voltage	Operating Frequency	Marking Code	Package
SH7018	3.3 V	20 MHz	HD64F7018X20	TFP-100B

Appendix D Package Dimensions

Figure D.1 shows the TFP-100B package dimensions of the SH7018.

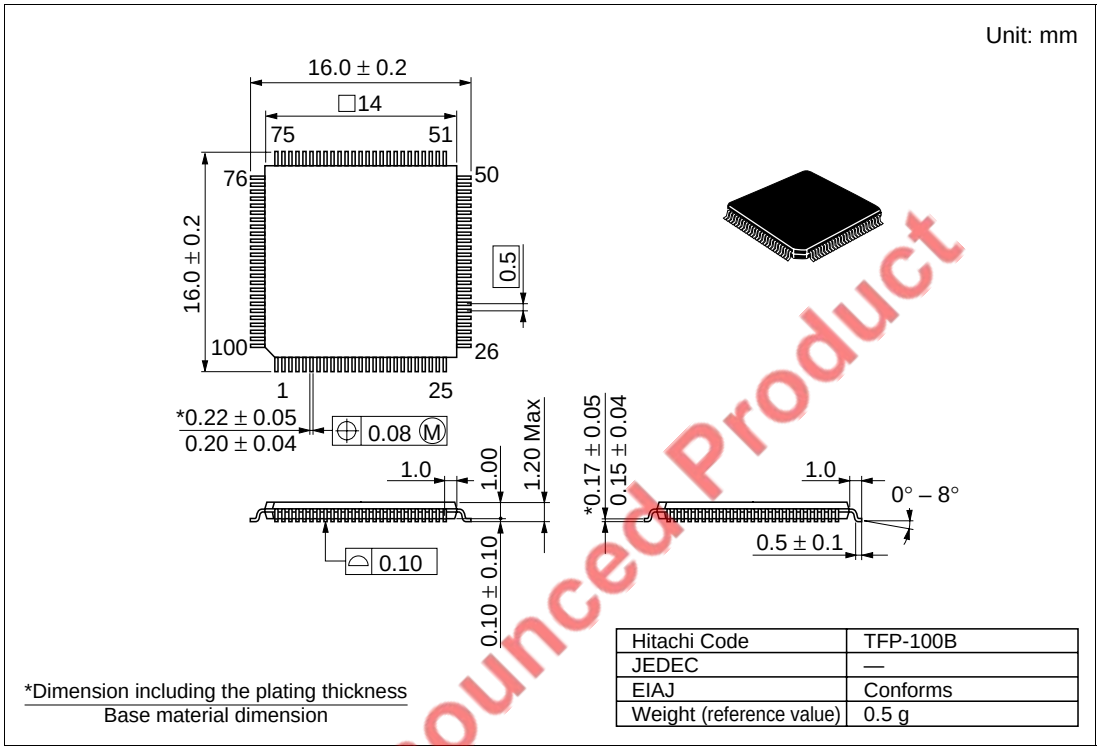


Figure D.1 Package Dimensions (TFP-100B)

SH7018 Hardware Manual

Publication Date: 1st Edition, July 2000
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.