To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

User's Manual

32

H8SX/1657 Group

Hardware Manual

Renesas 32-Bit CISC Microcomputer H8SX Family / H8SX/1600 Series

H8SX/1657C R5F61657C
H8SX/1656C R5F61656C

Renesas Electronics
www.renesas.com

Rev.2.00 2007.06

Notes regarding these materials

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

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

RENESAS

General Precautions in the Handling of MPU/MCU Products

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

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

How to Use This Manual

1. Objective and Target Users

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

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

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

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

   The following documents have been prepared for the H8SX/1657 Group. Before using any of the documents, please visit our web site to verify that you have the most up-to-date available version of the document.

Document Type	Contents	Document Title	Document No.
Data Sheet	Overview of hardware and electrical characteristics	—	—
Hardware Manual	Hardware specifications (pin assignments, memory maps, peripheral specifications, electrical characteristics, and timing charts) and descriptions of operation	H8SX/1657 Group Hardware Manual	This manual
Software Manual	Detailed descriptions of the CPU and instruction set	H8SX Family Software Manual	REJ09B0102
Application Note	Examples of applications and sample programs	The latest versions are available from our web site.	
Renesas Technical Update	Preliminary report on the specifications of a product, document, etc.		

2. Description of Numbers and Symbols

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

(1) Overall notation

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

(2) Register notation

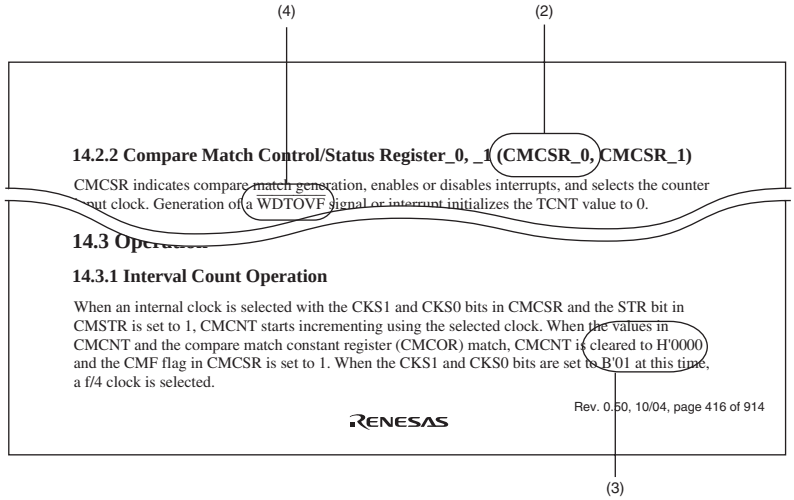
The style "register name"_"instance number" is used in cases where there is more than one instance of the same function or similar functions.
[Example] CMCSR_0: Indicates the CMCSR register for the compare-match timer of channel 0.

(3) Number notation

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

(4) Notation for active-low

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

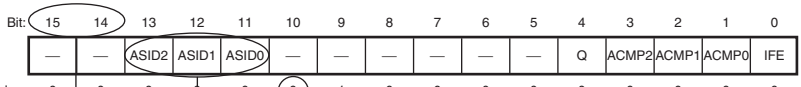


Note: The bit names and sentences in the above figure are examples and have nothing to do with the contents of this manual.

3. Description of Registers

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

[Bit Chart]



[Table of Bits]

(1) (2) (3) (4) (5)

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

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

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

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

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

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

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

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

4. Description of Abbreviations

The abbreviations used in this manual are listed below.

- Abbreviations specific to this product

Abbreviation	Description
BSC	Bus controller
CPG	Clock pulse generator
DTC	Data transfer controller
INTC	Interrupt controller
PPG	Programmable pulse generator
SCI	Serial communication interface
TMR	8-bit timer
TPU	16-bit timer pulse unit
WDT	Watchdog timer

- Abbreviations other than those listed above

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

Contents

Section 1 Overview

1.1 Features

The core of each product in the H8SX/1657 Group of CISC (complex instruction set computer) microcomputers is an H8SX CPU, which has an internal 32-bit architecture. The H8SX CPU provides upward-compatibility with the CPUs of other Renesas Technology-original microcomputers; H8/300, H8/300H, and H8S.

As peripheral functions, each LSI of the Group includes a DMA controller, which enables high-speed data transfer, and a bus-state controller, which enables direct connection to different kinds of memory. The LSI of the Group also includes serial communication interfaces, A/D and D/A converters, and a multi-function timer that makes motor control easy. Together, the modules realize low-cost configurations for end systems. The power consumption of these modules is kept down dynamically by an on-chip power-management function. The on-chip ROM is a flash memory (F-ZTAT™) with a capacity of 768 Kbytes (H8SX/1657C) or 512 Kbytes (H8SX/1656C).

Note: * F-ZTAT™ is a trademark of Renesas Technology Corp.

1.1.1 Applications

Examples of the applications of this LSI include PC peripheral equipment, optical storage devices, office automation equipment, and industrial equipment.

Notes: The following additions and changes have been made in the switch from the H8SX/1657 to the H8SX/1657C.

A sleep exception handling function has been added to the H8SX/1657C.

1.1.2 Overview of Functions

Table 1.1 lists the functions of H8SX/1657 Group products in outline.

Table 1.1 Overview of Functions

Classification	Module/ Function	Description
Memory	ROM	• ROM lineup: Flash memory version H8SX/1657C: 768 Kbytes H8SX/1656C: 512 Kbytes
	RAM	• RAM capacity: 24 Kbytes
CPU	CPU	• 32-bit high-speed H8SX CPU (CISC type) Upward-compatibility with H8/300, H8/300H, and H8S CPUs at object level • Sixteen 16-bit general registers • Eleven addressing modes • 4-Gbyte address space Program: 4 Gbytes available Data: 4 Gbytes available • 87 basic instructions, classifiable as bit arithmetic and logic instructions, multiply and divide instructions, bit manipulation instructions, multiply-and-accumulate instructions, and others • Minimum instruction execution time: 20.0 ns (for an ADD instruction while system clock $I\phi$ = 50 MHz and V_{CC} = 3.0 to 3.6 V) • On-chip multiplier (16 × 16 → 32 bits) • Supports multiply-and-accumulate instructions (16 × 16 + 32 → 32 bits)
	Operating mode	• Advanced mode

Classification	Module/ Function	Description
CPU	MCU operating mode	Mode 1: User boot mode (selected by driving the MD2 and MD1 pins low and driving the MD0 pin high) Mode 2: Boot mode (selected by driving the MD2 and MD0 pins low and driving the MD1 pin high) Mode 4: On-chip ROM disabled external extended mode, 16-bit bus (selected by driving the MD1 and MD0 pins low and driving the MD2 pin high) Mode 5: On-chip ROM disabled external extended mode, 8-bit bus (selected by driving the MD1 pin low and driving the MD2 and MD0 pins high) Mode 6: On-chip ROM enabled external extended mode (selected by driving the MD0 pin low and driving the MD2 and MD1 pins high) Mode 7: Single chip mode (external extension possible) (selected by driving the MD2, MD1, and MD0 pins high) • Power-down state (transition to the power-down state made by the SLEEP instruction)
Interrupt (source)	Interrupt controller (INTC)	• Thirteen external interrupt pins (NMI, and $\overline{\text{IRQ11}}$ to $\overline{\text{IRQ0}}$) • 64 internal interrupt sources • Two interrupt control modes (specified by the interrupt control register) • Eight priority orders specifiable (by setting the interrupt priority register) • Independent vector addresses

Classification	Module/ Function	Description
DMA	DMA controller (DMAC)	• Four-channel DMA transfer available • Three activation methods (auto-request, on-chip module interrupt, and external request) • Three transfer modes (normal transfer, repeat transfer, block transfer) • Dual or single address mode selectable • Extended repeat-area function
DTC	Data transfer controller (DTC)	• Allows DTC transfer of 53 channels (number of DTC activation sources) • Activated by interrupt sources (chain transfer enabled) • Three transfer modes (normal transfer, repeat transfer, and block transfer) • Short-address mode or full-address mode selectable
External bus extension	Bus controller (BSC)	• 16-Mbyte external address space • The external address space can be divided into eight areas, each of which is independently controllable — Chip-select signals ($\overline{CS0}$ to $\overline{CA7}$) can be output — Access in two or three states can be selected for each area — Program wait cycles can be inserted — The period of $\overline{CS}$ assertion can be extended — Idle cycles can be inserted • Bus arbitration function (arbitrates bus mastership among the internal CPU, DMAC, and DTC, and external bus masters) Bus formats • External memory interfaces (for the connection of ROM, burst ROM, SRAM, and byte control SRAM) • Address/data bus format: Support for both separate and multiplexed buses (8-bit access or 16-bit access) • Endian conversion function for connecting devices in little-endian format

Classification	Module/ Function	Description
Clock	Clock pulse generator (CPG)	• One clock generation circuit available • Separate clock signals are provided for each of functional modules (detailed below) and each is independently specifiable (multi-clock function) — System-intended data transfer modules, i.e. the CPU, runs in synchronization with the system clock (Iϕ): 8 to 35 MHz — Internal peripheral functions run in synchronization with the peripheral module clock (Pϕ): 8 to 35 MHz — Modules in the external space run in synchronization with the external bus clock (Bϕ): 8 to 35 MHz • Includes a PLL frequency multiplication circuit and frequency divider, so the operating frequency is selectable • Four power-down modes: Sleep mode, all-module-clock-stop mode, software standby mode, and hardware standby mode
A/D converter	A/D converter (ADC)	• 10-bit resolution × eight input channels • Sample and hold function included • Conversion time: 7.4 μs per channel (with peripheral module clock (Pϕ) at 35-MHz operation) • Two operating modes: single mode and scan mode • Three ways to start A/D conversion: software, timer (TPU/TMR) trigger, and external trigger
D/A converter	D/A converter (DAC)	• 8-bit resolution × two output channels • Output voltage: 0 V to Vref, maximum conversion time: 10 μs (with 20-pF load)

Classification	Module/ Function	Description
Timer	8-bit timer (TMR)	• 8 bits × four channels (can be used as 16 bits × two channels) • Selectable from seven clock sources (six internal clocks and one external clock) • Enables the timer to output pulses with a desired duty cycle or PWM output
	16-bit timer pulse unit (TPU)	• 16 bits × six channels (general pulse timer unit) • Selectable from eight counter-input clocks for each channel • Up to 16 pulse inputs and outputs • Counter clear operation, simultaneous writing to multiple timer counters (TCNT), simultaneous clearing by compare match and input capture possible, simultaneous input/output for registers possible by counter synchronous operation, and up to 15-phase PWM output possible by combination with synchronous operation • Buffered operation, cascaded operation (32 bits × two channels), and phase counting mode (two-phase encoder input) settable for each channel • Input capture function supported • Output compare function (by the output of compare match waveform) supported
	Programmable pulse generator (PPG)	• 16-bit pulse output • Four output groups, non-overlapping mode, and inverted output can be set • Selectable output trigger signals; the PPG can operate in conjunction with the data transfer controller (DTC) and the DMA controller (DMAC)
Watchdog timer	Watchdog timer (WDT)	• 8 bits × one channel (selectable from eight counter input clocks) • Switchable between watchdog timer mode and interval timer mode
Serial interface	Serial communication interface (SCI)	• Four channels (choice of asynchronous or clocked synchronous serial communication mode) • Full-duplex communication capability • Select the desired bit rate and LSB-first or MSB-first transfer
Smart card/ SIM		• The SCI module supports a smart card (SIM) interface.

Classification	Module/ Function	Description
I/O ports		• Nine CMOS input-only pins • 81 CMOS input/output pins • Eight large-current drive pins (port 3) • 40 pull-up resistors • 16 open drains
Package		• 120-pin thin QFP package (package code: TFP-120V, package dimensions: 14 × 14 mm, pin pitch: 0.40 mm) • Lead- (Pb-) free versions available
Operating frequency/ Power supply voltage		• Operating frequency: 8 to 35 MHz • Power supply voltage: Vcc = 3.0 to 3.6 V, AVcc = 3.0 to 3.6 V • Supply current: — 35 mA (typ.) (Vcc = 3.3 V, AVcc = 3.3 V, Iϕ = Pϕ = Bϕ = 35 MHz)
Operating peripheral temperature (°C)		• –20 to +75°C (regular specifications) • –40 to +85°C (wide-range specifications)

1.2 List of Products

Table 1.2 is the list of products, and figure 1.1 shows how to read the product name code.

Table 1.2 List of Products

Product Type No.	ROM Capacity	RAM Capacity	Package	Remarks
R5F61657CFTV	768 Kbytes	24 Kbytes	PTQP0120LA-A (TFP-120V)	Flash memory version
R5F61656CFTV	512 Kbytes			

(as of June, 2007)

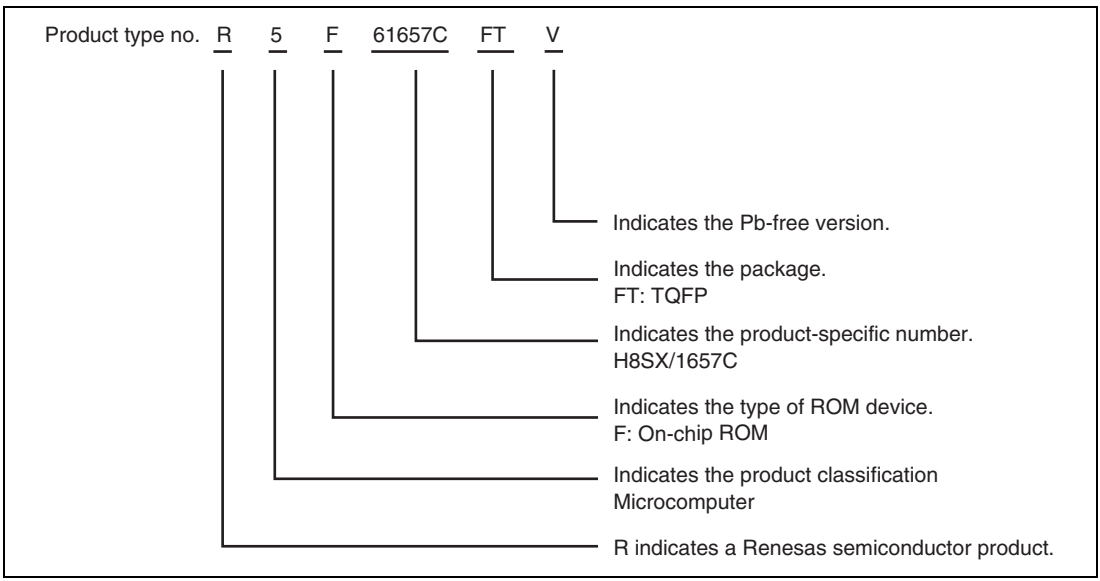


Figure 1.1 How to Read the Product Name Code

1.3 Block Diagram

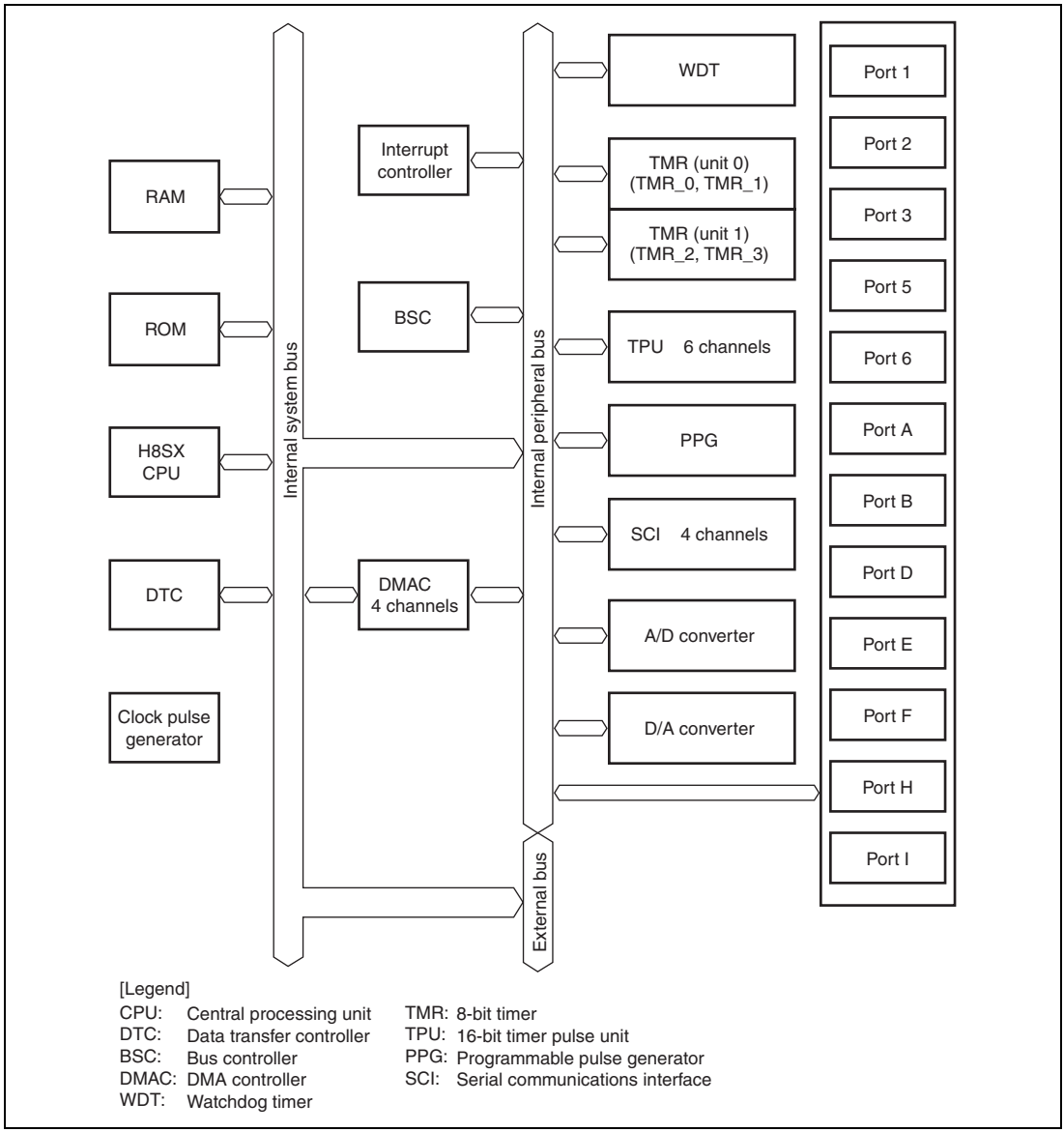


Figure 1.2 Internal Block Diagram

1.4 Pin Descriptions

1.4.1 Pin Assignments

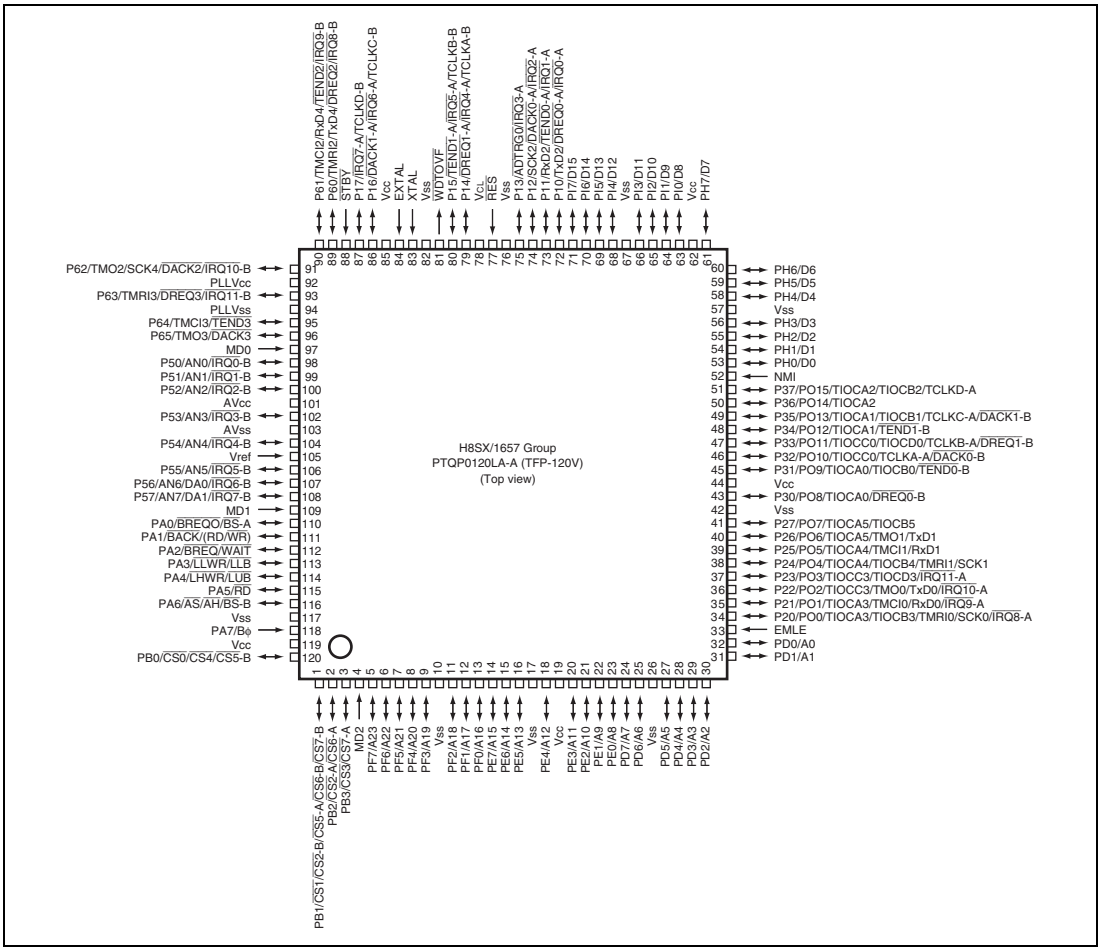


Figure 1.3 Pin Assignments

1.4.2 Pin Functions

Table 1.3 Pin Functions

Classification	Pin Name	I/O	Description
Power supply	V_{CC}	Input	Power supply pin. Connect this to the system power supply.
	V_{CL}	Input	Connect this pin to V_{SS} via a 0.1-uF capacitor (The capacitor should be placed close to the pin).
	V_{SS}	Input	Ground pin. Connect this to the system power supply (0 V).
	$PLLV_{CC}$	Input	Power supply pin for the PLL circuit.
	$PLLV_{SS}$	Input	Ground pin for the PLL circuit.
Clock	XTAL	Input	Pins for a crystal resonator. An external clock signal can be input through the EXTAL pin. For an example of this connection, see section 19, Clock Pulse Generator.
	EXTAL	Input	
	Bϕ	Output	Outputs the system clock for external devices.
Operating mode control	MD2 to MD0	Input	Pins for setting the operating mode. The signal levels on these pins must not be changed during operation.
System control	$\overline{RES}$	Input	Reset pin. This LSI enters the reset state when this signal goes low.
	$\overline{STBY}$	Input	This LSI enters hardware standby mode when this signal goes low.
	EMLE	Input	Input pin for the on-chip emulator enable signal. The signal level should normally be fixed low.
Address bus	A23 to A0	Output	Output pins for the address bits.
Data bus	D15 to D0	Input/ output	Input and output for the bidirectional data bus. These pins also output addresses when accessing an address–data multiplexed I/O interface space.
Bus control	$\overline{BREQ}$	Input	External bus-master modules assert this signal to request the bus.
	$\overline{BREQO}$	Output	Internal bus-master modules assert this signal to request access to the external space via the bus in the external bus released state.

Classification	Pin Name	I/O	Description
Bus control	$\overline{\text{BACK}}$	Output	Bus acknowledge signal, which indicates that the bus has been released.
	$\overline{\text{BS}}$-A/$\overline{\text{BS}}$-B	Output	Indicates the start of a bus cycle.
	$\overline{\text{AS}}$	Output	Strobe signal which indicates that the output address on the address bus is valid in access to the basic bus interface or byte control SRAM interface space.
	$\overline{\text{AH}}$	Output	This signal is used to hold the address when accessing the address-data multiplexed I/O interface space.
	$\overline{\text{RD}}$	Output	Strobe signal which indicates that reading from the basic bus interface space is in progress.
	RD/$\overline{\text{WR}}$	Output	Indicates the direction (input or output) of the data bus.
	$\overline{\text{LHWR}}$	Output	Strobe signal which indicates that the higher-order byte (D15 to D8) is valid in access to the basic bus interface space.
	$\overline{\text{LLWR}}$	Output	Strobe signal which indicates that the lower-order byte (D7 to D0) is valid in access to the basic bus interface space.
	$\overline{\text{LUB}}$	Output	Strobe signal which indicates that the higher-order byte (D15 to D8) is valid in access to the byte control SRAM interface space.
	$\overline{\text{LLB}}$	Output	Strobe signal which indicates that the lower-order byte (D7 to D0) is valid in access to the byte control SRAM interface space.
	$\overline{\text{CS0}}$ $\overline{\text{CS1}}$ $\overline{\text{CS2}}$-A/$\overline{\text{CS2}}$-B $\overline{\text{CS3}}$ $\overline{\text{CS4}}$ $\overline{\text{CS5}}$-A/$\overline{\text{CS5}}$-B $\overline{\text{CS6}}$-A/$\overline{\text{CS6}}$-B $\overline{\text{CS7}}$-A/$\overline{\text{CS7}}$-B	Output	Select signals for areas 7 to 0.
	$\overline{\text{WAIT}}$	Input	Requests wait cycles in access to the external space.

Classification	Pin Name	I/O	Description
Interrupt	NMI	Input	Non-maskable interrupt request signal. When this pin is not in use, this signal must be fixed high.
	$\overline{\text{IRQ11}}$-A/$\overline{\text{IRQ11}}$-B $\overline{\text{IRQ10}}$-A/$\overline{\text{IRQ10}}$-B $\overline{\text{IRQ9}}$-A/$\overline{\text{IRQ9}}$-B $\overline{\text{IRQ8}}$-A/$\overline{\text{IRQ8}}$-B $\overline{\text{IRQ7}}$-A/$\overline{\text{IRQ7}}$-B $\overline{\text{IRQ6}}$-A/$\overline{\text{IRQ6}}$-B $\overline{\text{IRQ5}}$-A/$\overline{\text{IRQ5}}$-B $\overline{\text{IRQ4}}$-A/$\overline{\text{IRQ4}}$-B $\overline{\text{IRQ3}}$-A/$\overline{\text{IRQ3}}$-B $\overline{\text{IRQ2}}$-A/$\overline{\text{IRQ2}}$-B $\overline{\text{IRQ1}}$-A/$\overline{\text{IRQ1}}$-B $\overline{\text{IRQ0}}$-A/$\overline{\text{IRQ0}}$-B	Input	Maskable interrupt request signal.
DMA controller (DMAC)	$\overline{\text{DREQ0}}$-A/$\overline{\text{DREQ0}}$-B $\overline{\text{DREQ1}}$-A/$\overline{\text{DREQ1}}$-B $\overline{\text{DREQ2}}$ $\overline{\text{DREQ3}}$	Input	Requests DMAC activation.
	$\overline{\text{DACK0}}$-A/$\overline{\text{DACK0}}$-B $\overline{\text{DACK1}}$-A/$\overline{\text{DACK1}}$-B $\overline{\text{DACK2}}$ $\overline{\text{DACK3}}$	Output	DMAC single address-transfer acknowledge signal.
	$\overline{\text{TEND0}}$-A/$\overline{\text{TEND0}}$-B $\overline{\text{TEND1}}$-A/$\overline{\text{TEND1}}$-B $\overline{\text{TEND2}}$ $\overline{\text{TEND3}}$	Output	Indicates end of data transfer by the DMAC.
16-bit timer pulse unit (TPU)	TCLKA-A/TCLKA-B TCLKB-A/TCLKB-B TCLKC-A/TCLKC-B TCLKD-A/TCLKD-B	Input	Input pins for the external clock signals.
	TIOCA0 TIOCB0 TIOCC0 TIOCD0	Input/ output	Signals for TGRA_0 to TGRD_0. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA1 TIOCB1	Input/ output	Signals for TGRA_1 and TGRB_1. These pins are used as input capture inputs, output compare outputs, or PWM outputs.

Classification	Pin Name	I/O	Description
16-bit timer pulse unit (TPU)	TIOCA2 TIOCB2	Input/ output	Signals for TGRA_2 and TGRB_2. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA3 TIOCB3 TIOCC3 TIOCD3	Input/ output	Signals for TGRA_3 to TGRD_3. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA4 TIOCB4	Input/ output	Signals for TGRA_4 and TGRB_4. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA5 TIOCB5	Input/ output	Signals for TGRA_5 and TGRB_5. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
Programmable pulse generator (PPG)	PO15 to PO0	Output	Output pins for the pulse signals.
8-bit timer (TMR)	TMO0 to TMO3	Output	Output pins for the compare match signals.
	TMCI0 to TMCI3	Input	Input pins for the external clock signals that drive for the counters.
	TMRI0 to TMRI3	Input	Input pins for the counter-reset signals.
Watchdog timer (WDT)	$\overline{\text{WDTOVF}}$	Output	Output pin for the counter-overflow signal in watchdog-timer mode.
Serial communication interface (SCI)	TxD0 to TxD4	Output	Output pins for data transmission.
	RxD0 to RxD4	Input	Input pins for data reception.
	SCK0 to SCK4	Input/ output	Input/output pins for clock signals.

Classification	Pin Name	I/O	Description
A/D converter	AN7 to AN0	Input	Input pins for the analog signals to be processed by the A/D converter.
	$\overline{\text{ADTRG0}}$	Input	Input pin for the external trigger signal that starts A/D conversion.
D/A converter	DA1, DA0	Output	Output pins for the analog signals from the D/A converter.
A/D converter, D/A converter	AV_{CC}	Input	Analog power supply pin for the A/D and D/A converters. When the A/D and D/A converters are not in use, connect this pin to the system power supply.
	AV_{SS}	Input	Ground pin for the A/D and D/A converters. Connect this pin to the system power supply (0 V).
	Vref	Input	Reference power supply pin for the A/D and D/A converters. When the A/D and D/A converters are not in use, connect this pin to the system power supply.
I/O ports	P17 to P10	Input/ output	8-bit input/output pins.
	P27 to P20	Input/ output	8-bit input/output pins.
	P37 to P30	Input/ output	8-bit input/output pins.
	P57 to P50	Input	8-bit input/output pins.
	P65 to P60	Input/ output	6-bit input/output pins.
	PA7	Input	Input-only pin.
	PA6 to PA0	Input/ output	7-bit input/output pins.
	PB3 to PB0	Input/ output	4-bit input/output pins.
	PD7 to PD0	Input/ output	8-bit input/output pins.
	PE7 to PE0	Input/ output	8-bit input/output pins.
	PF7 to PF0	Input/ output	8-bit input/output pins.
	PH7 to PH0	Input/ output	8-bit input/output pins.
	PI7 to PI0	Input/ output	8-bit input/output pins.

Section 2 CPU

The H8SX CPU is a high-speed CPU with an internal 32-bit architecture that is upward-compatible with the H8/300, H8/300H, and H8S CPUs.

The H8SX CPU has sixteen 16-bit general registers, can handle a 4-Gbyte linear address space, and is ideal for a realtime control system.

2.1 Features

- Upward-compatible with H8/300, H8/300H, and H8S CPUs
  - Can execute H8/300, H8/300H, and H8S/2000 object programs
- Sixteen 16-bit general registers
  - Also usable as sixteen 8-bit registers or eight 32-bit registers
- 87 basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Bit field transfer instructions
  - Powerful bit-manipulation instructions
  - Bit condition branch instructions
  - Multiply-and-accumulate instruction
- Eleven addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:2,ERn), @(d:16,ERn), or @(d:32,ERn)]
  - Index register indirect with displacement [@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)]
  - Register indirect with pre-/post-increment or pre-/post-decrement [@+ERn, @–ERn, @ERn+, or @ERn–]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:3, #xx:4, #xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Program-counter relative with index register [@(RnL.B,PC), @(Rn.W,PC), or @(ERn.L,PC)]
  - Memory indirect [@@aa:8]
  - Extended memory indirect [@@vec:7]

- Two base registers
  - Vector base register
  - Short address base register
- 4-Gbyte address space
  - Program: 4 Gbytes
  - Data: 4 Gbytes
- High-speed operation
  - All frequently-used instructions executed in one or two states
  - 8/16/32-bit register-register add/subtract: 1 state
  - 8 × 8-bit register-register multiply: 1 state
  - 16 ÷ 8-bit register-register divide: 10 states
  - 16 × 16-bit register-register multiply: 1 state
  - 32 ÷ 16-bit register-register divide: 18 states
  - 32 × 32-bit register-register multiply: 5 states
  - 32 ÷ 32-bit register-register divide: 18 states
- Four CPU operating modes
  - Normal mode
  - Middle mode
  - Advanced mode
  - Maximum mode
- Power-down modes
  - Transition is made by execution of SLEEP instruction
  - Choice of CPU operating clocks

Notes: 1. Advanced mode is only supported as the CPU operating mode of the H8SX/1657 Group. Normal, middle, and maximum modes are not supported.
2. The multiplier and divider are supported by the H8SX/1657 Group.

2.2 CPU Operating Modes

The H8SX CPU has four operating modes: normal, middle, advanced and maximum modes. For details on mode settings, see section 3.1, Operating Mode Selection.

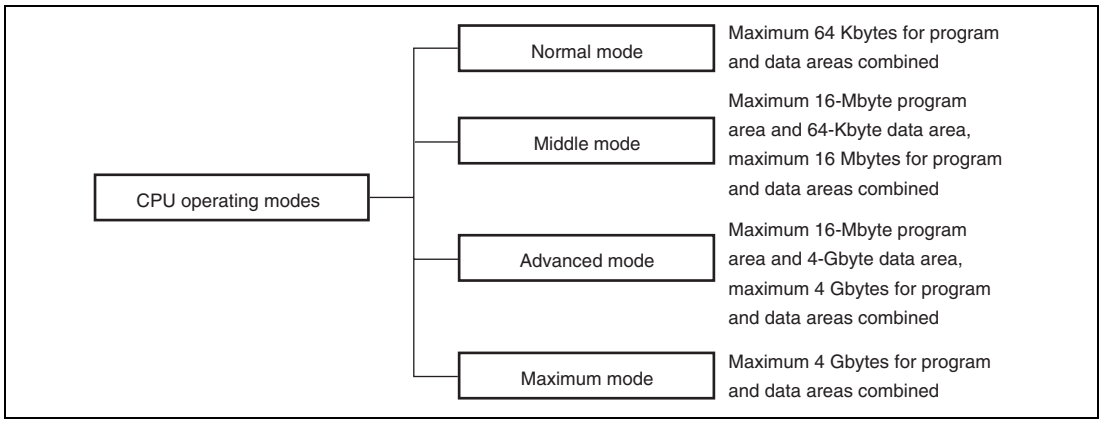


Figure 2.1 CPU Operating Modes

2.2.1 Normal Mode

The exception vector table and stack have the same structure as in the H8/300 CPU.

Note: Normal mode is not supported in this LSI.

- Address Space

  The maximum address space of 64 Kbytes can be accessed.
- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When the extended register En is used as a 16-bit register it can contain any value, even when the corresponding general register Rn is used as an address register. (If the general register Rn is referenced in the register indirect addressing mode with pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, the value in the corresponding extended register En will be affected.)
- Instruction Set

  All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.

- Exception Vector Table and Memory Indirect Branch Addresses

  In normal mode, the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The structure of the exception vector table is shown in figure 2.2.

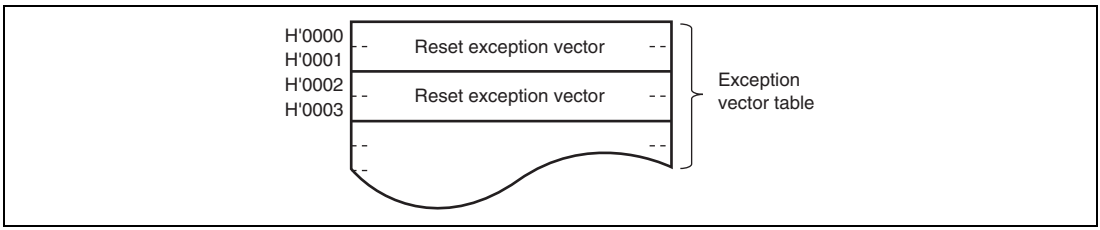


Figure 2.2 Exception Vector Table (Normal Mode)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.3. The PC contents are saved or restored in 16-bit units.

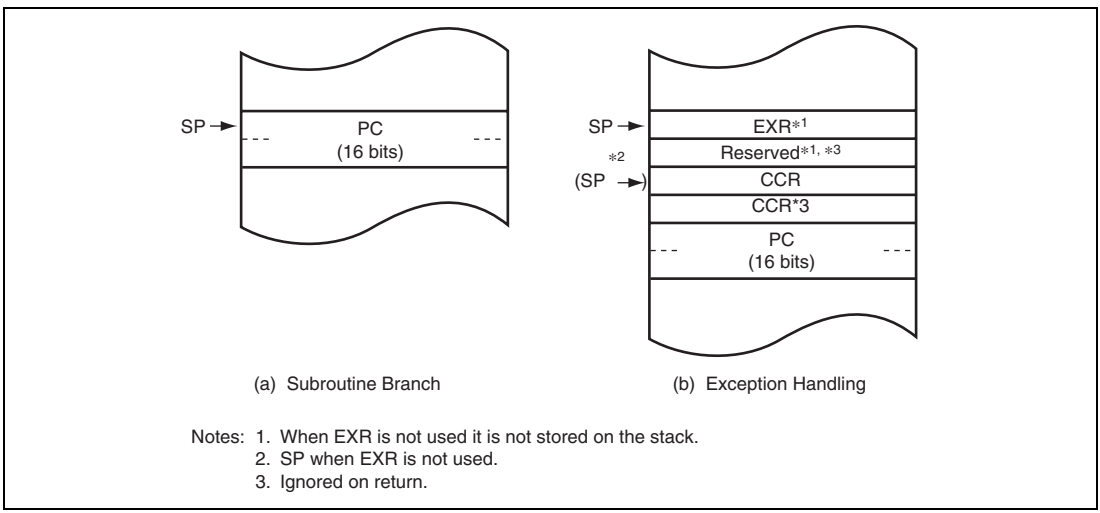


Figure 2.3 Stack Structure (Normal Mode)

2.2.2 Middle Mode

The program area in middle mode is extended to 16 Mbytes as compared with that in normal mode.

- Address Space

  The maximum address space of 16 Mbytes can be accessed as a total of the program and data areas. For individual areas, up to 16 Mbytes of the program area or up to 64 Kbytes of the data area can be allocated.
- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When the extended register En is used as a 16-bit register (in other than the JMP and JSR instructions), it can contain any value even when the corresponding general register Rn is used as an address register. (If the general register Rn is referenced in the register indirect addressing mode with pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, the value in the corresponding extended register En will be affected.)
- Instruction Set

  All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid and the upper eight bits are sign-extended.
- Exception Vector Table and Memory Indirect Branch Addresses

  In middle mode, the top area starting at H'000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.

  The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

  In middle mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address. The upper eight bits are reserved and assumed to be H'00.
- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.

2.2.3 Advanced Mode

The data area is extended to 4 Gbytes as compared with that in middle mode.

- Address Space

  The maximum address space of 4 Gbytes can be linearly accessed. For individual areas, up to 16 Mbytes of the program area and up to 4 Gbytes of the data area can be allocated.
- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers or address registers.
- Instruction Set

  All instructions and addressing modes can be used.
- Exception Vector Table and Memory Indirect Branch Addresses

  In advanced mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.

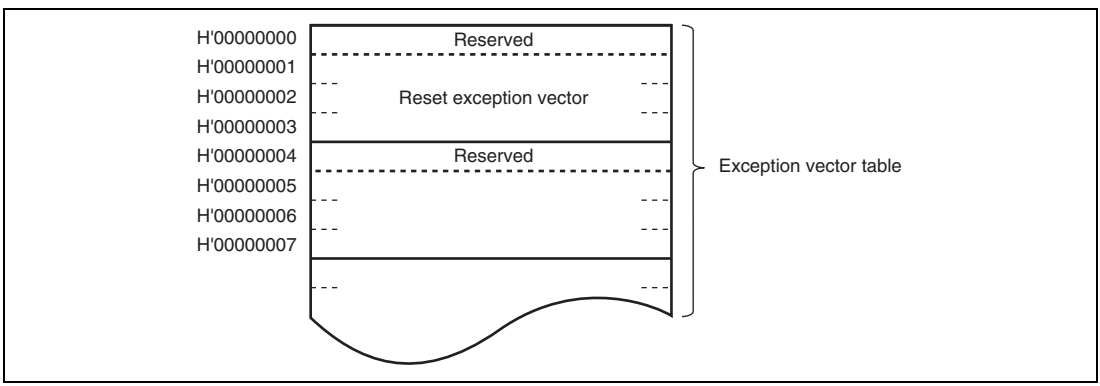


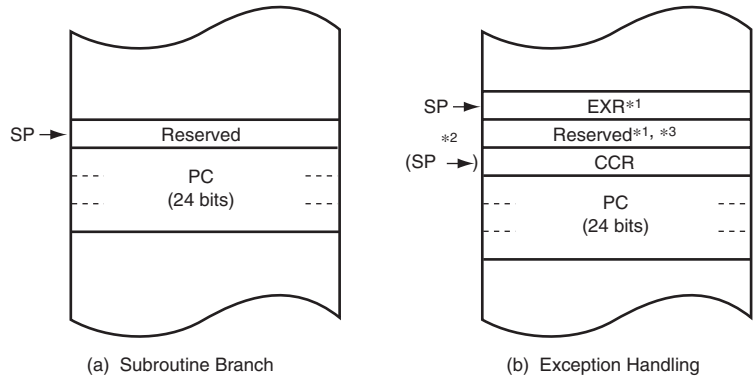
Figure 2.4 Exception Vector Table (Middle and Advanced Modes)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

In advanced mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address. The upper eight bits are reserved and assumed to be H'00.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.



Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored on return.

Figure 2.5 Stack Structure (Middle and Advanced Modes)

2.2.4 Maximum Mode

The program area is extended to 4 Gbytes as compared with that in advanced mode.

- Address Space

  The maximum address space of 4 Gbytes can be linearly accessed.
- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers or as the upper 16-bit segments of 32-bit registers or address registers.
- Instruction Set

  All instructions and addressing modes can be used.
- Exception Vector Table and Memory Indirect Branch Addresses

  In maximum mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The structure of the exception vector table is shown in figure 2.6.

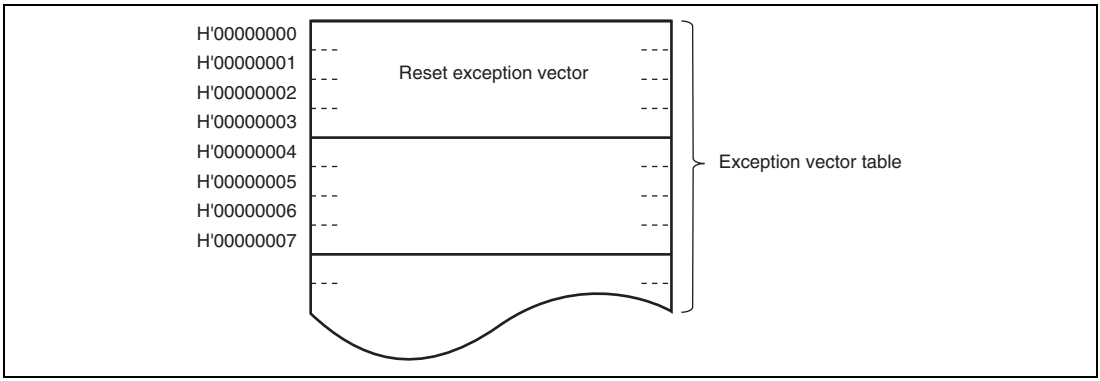


Figure 2.6 Exception Vector Table (Maximum Modes)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

In maximum mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.7. The PC contents are saved or restored in 32-bit units. The EXR contents are saved or restored regardless of whether or not EXR is in use.

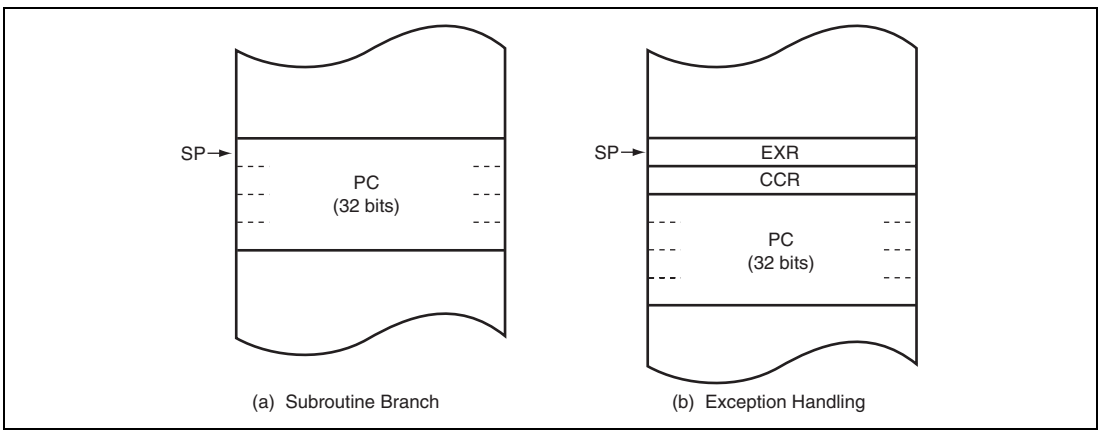


Figure 2.7 Stack Structure (Maximum Mode)

2.3 Instruction Fetch

The H8SX CPU has two modes for instruction fetch: 16-bit and 32-bit modes. It is recommended that the mode be set according to the bus width of the memory in which a program is stored. The instruction-fetch mode setting does not affect operation other than instruction fetch such as data accesses. Whether an instruction is fetched in 16- or 32-bit mode is selected by the FETCHMD bit in SYSCR. For details, see section 3.2.2, System Control Register (SYSCR).

2.4 Address Space

Figure 2.8 shows a memory map of the H8SX CPU. The address space differs depending on the CPU operating mode.

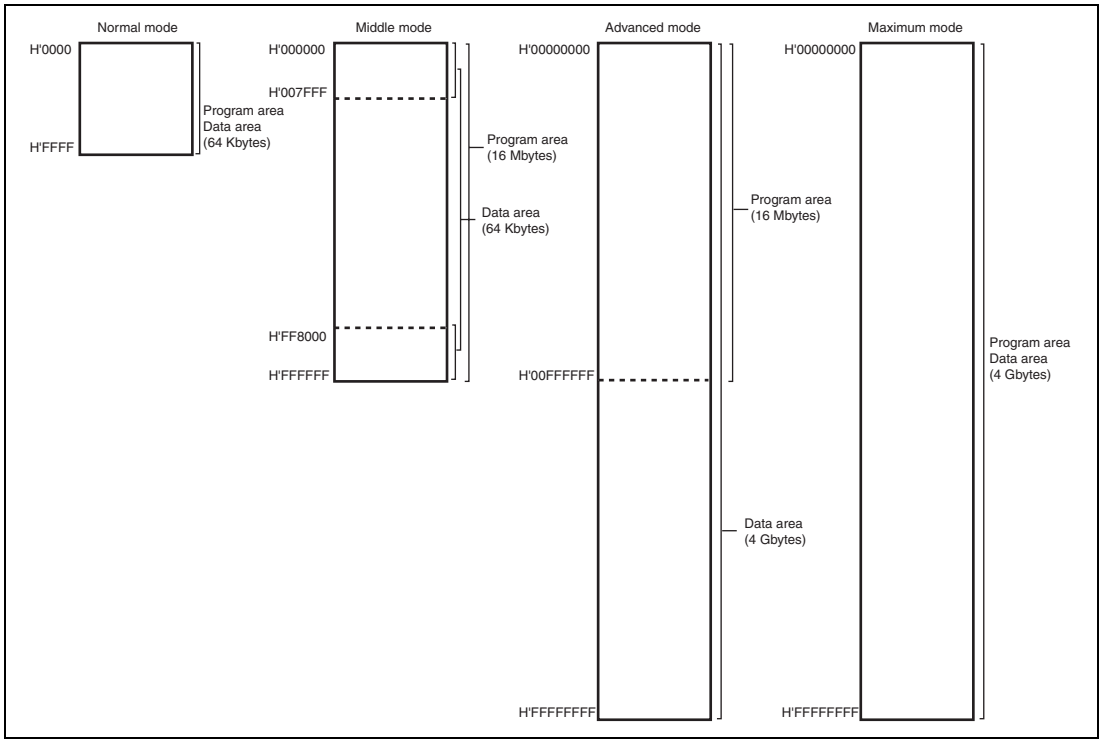


Figure 2.8 Memory Map

2.5 Registers

The H8SX CPU has the internal registers shown in figure 2.9. There are two types of registers: general registers and control registers. The control registers are the 32-bit program counter (PC), 8-bit extended control register (EXR), 8-bit condition-code register (CCR), 32-bit vector base register (VBR), 32-bit short address base register (SBR), and 64-bit multiply-accumulate register (MAC).

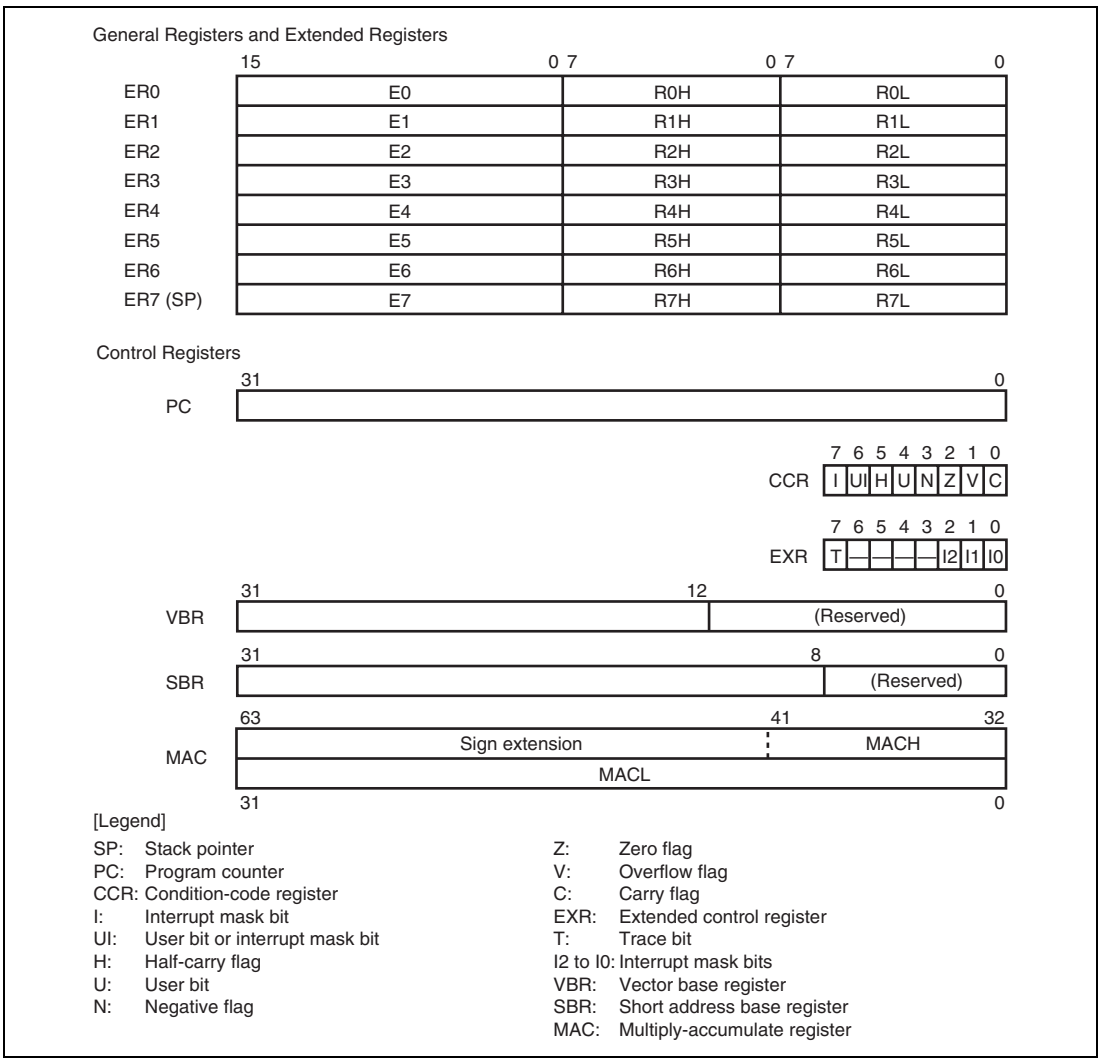


Figure 2.9 CPU Registers

2.5.1 General Registers

The H8SX CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.10 illustrates the usage of the general registers.

When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

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

When the general registers are used as 8-bit registers, the R registers are divided into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

The general registers ER (ER0 to ER7), R (R0 to R7), and RL (R0L to R7L) are also used as index registers. The size in the operand field determines which register is selected.

The usage of each register can be selected independently.

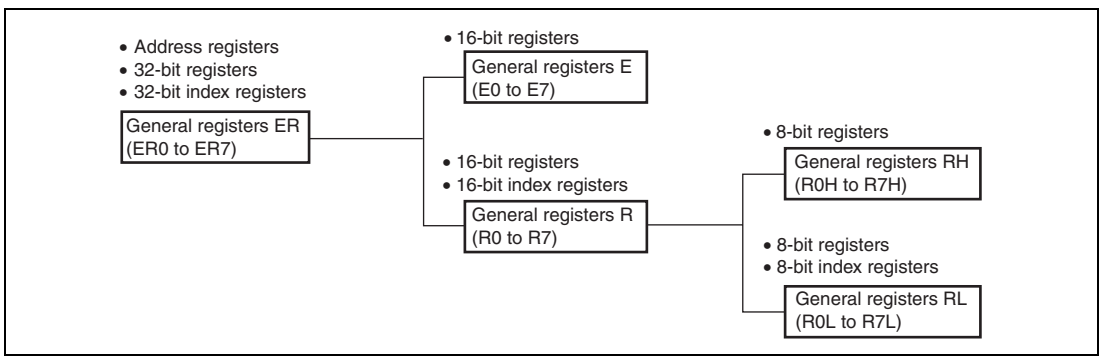


Figure 2.10 Usage of General Registers

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine branches. Figure 2.11 shows the stack.

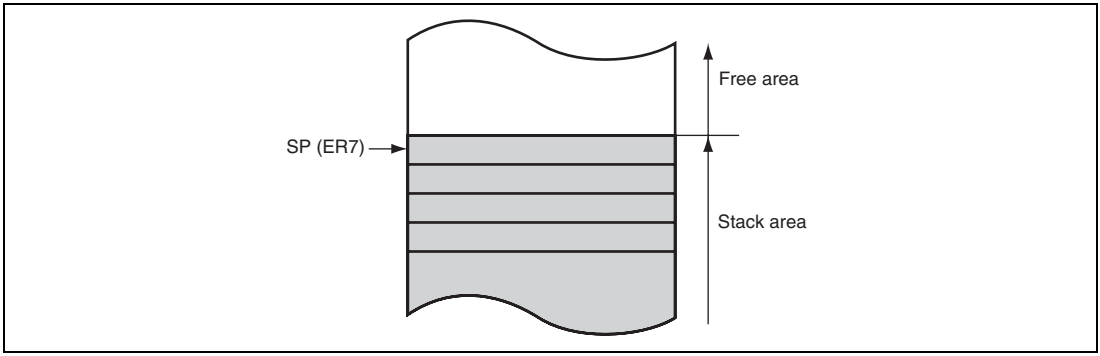


Figure 2.11 Stack

2.5.2 Program Counter (PC)

PC is a 32-bit counter that indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 16 bits (one word) or a multiple of 16 bits, so the least significant bit is ignored. (When the instruction code is fetched, the least significant bit is regarded as 0.

2.5.3 Condition-Code Register (CCR)

CCR is an 8-bit register that contains internal CPU status information, including an interrupt mask (I) and user (UI, U) bits and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

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

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	Interrupt Mask Bit Masks interrupts when set to 1. This bit is set to 1 at the start of an exception handling.
6	UI	Undefined	R/W	User Bit or Interrupt Mask Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit can also be used as an interrupt mask bit.
5	H	Undefined	R/W	Half-Carry Flag When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.
4	U	Undefined	R/W	User Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag Stores the value of the most significant bit (regarded as sign bit) of data.

Bit	Bit Name	Initial Value	R/W	Description
2	Z	Undefined	R/W	Zero Flag Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.
0	C	Undefined	R/W	Carry Flag Set to 1 when a carry occurs, and cleared to 0 otherwise. A carry has the following types: • Carry from the result of addition • Borrow from the result of subtraction • Carry from the result of shift or rotation The carry flag is also used as a bit accumulator by bit manipulation instructions.

2.5.4 Extended Control Register (EXR)

EXR is an 8-bit register that contains the trace bit (T) and three interrupt mask bits (I2 to I0).

Operations can be performed on the EXR bits by the LDC, STC, ANDC, ORC, and XORC instructions.

For details, see section 4, Exception Handling.

Bit	Bit Name	Initial Value	R/W	Description
7	T	0	R/W	Trace Bit When this bit is set to 1, a trace exception is generated each time an instruction is executed. When this bit is cleared to 0, instructions are executed in sequence.
6 to 3	—	All 1	R/W	Reserved These bits are always read as 1.

Bit	Bit Name	Initial Value	R/W	Description
2	I2	1	R/W	Interrupt Mask Bits
1	I1	1	R/W	These bits designate the interrupt mask level (0 to 7).
0	I0	1	R/W	

2.5.5 Vector Base Register (VBR)

VBR is a 32-bit register in which the upper 20 bits are valid. The lower 12 bits of this register are read as 0s. This register is a base address of the vector area for exception handlings other than a reset and a CPU address error (extended memory indirect is also out of the target). The initial value is H'00000000. The VBR contents are changed with the LDC and STC instructions.

2.5.6 Short Address Base Register (SBR)

SBR is a 32-bit register in which the upper 24 bits are valid. The lower eight bits are read as 0s. In 8-bit absolute address addressing mode (@aa:8), this register is used as the upper address. The initial value is H'FFFFFF00. The SBR contents are changed with the LDC and STC instructions.

2.5.7 Multiply-Accumulate Register (MAC)

MAC is a 64-bit register that stores the results of multiply-and-accumulate operations. It consists of two 32-bit registers denoted MACH and MACL. The lower 10 bits of MACH are valid; the upper bits are sign extended. The MAC contents are changed with the MAC, CLRMAC, LDMAC, and STMAC instructions.

2.5.8 Initial Values of CPU Registers

Reset exception handling loads the start address from the vector table into the PC, clears the T bit in EXR to 0, and sets the I bits in CCR and EXR to 1. The general registers, MAC, and the other bits in CCR are not initialized. In particular, the initial value of the stack pointer (ER7) is undefined. The SP should therefore be initialized using an MOV.L instruction executed immediately after a reset.

2.6 Data Formats

The H8SX CPU can process 1-bit, 4-bit BCD, 8-bit (byte), 16-bit (word), and 32-bit (longword) data.

Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

2.6.1 General Register Data Formats

Figure 2.12 shows the data formats in general registers.

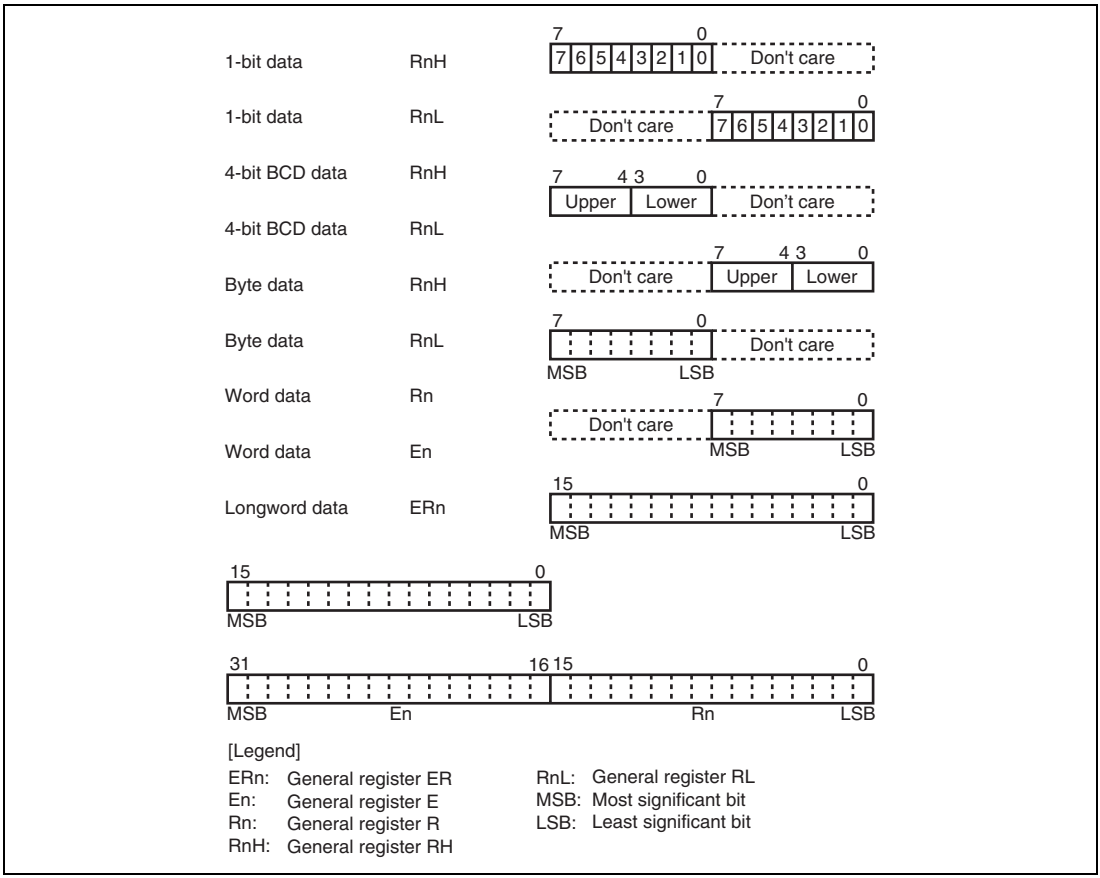


Figure 2.12 General Register Data Formats

2.6.2 Memory Data Formats

Figure 2.13 shows the data formats in memory.

The H8SX CPU can access word data and longword data which are stored at any addresses in memory. When word data begins at an odd address or longword data begins at an address other than a multiple of 4, a bus cycle is divided into two or more accesses. For example, when longword data begins at an odd address, the bus cycle is divided into byte, word, and byte accesses. In this case, these accesses are assumed to be individual bus cycles.

However, instructions to be fetched, word and longword data to be accessed during execution of the stack manipulation, branch table manipulation, block transfer instructions, and MAC instruction should be located to even addresses.

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

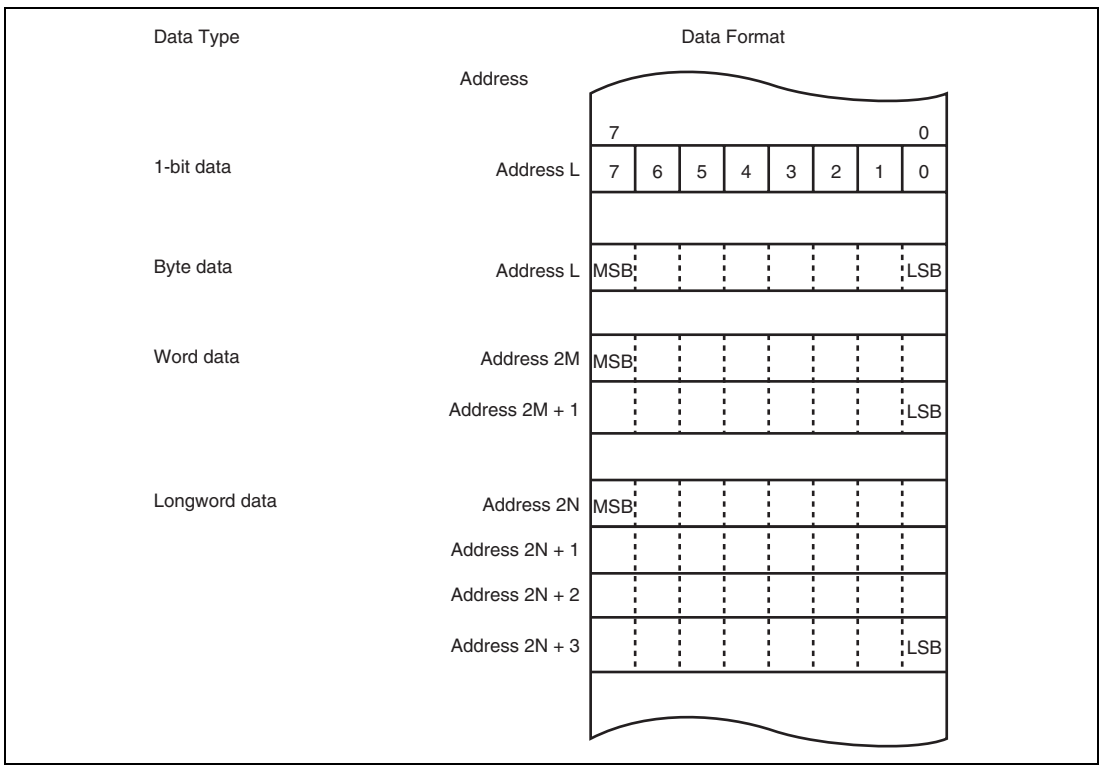


Figure 2.13 Memory Data Formats

2.7 Instruction Set

The H8SX CPU has 87 types of instructions. The instructions are classified by function as shown in table 2.1. The arithmetic operation, logic operation, shift, and bit manipulation instructions are called operation instruction in this manual.

Table 2.1 Instruction Classification

Function	Instructions	Size	Types
Data transfer	MOV	B/W/L	6
	MOVFPE*[6], MOVTPE*[6]	B	
	POP, PUSH*[1]	W/L	
	LDM, STM	L	
	MOVA	B/W*[2]	
Block transfer	EEPMOV	B	3
	MOVMD	B/W/L	
	MOVSD	B	
Arithmetic operations	ADD, ADDX, SUB, SUBX, CMP, NEG, INC, DEC	B/W/L	27
	DAA, DAS	B	
	ADDS, SUBS	L	
	MULXU, DIVXU, MULXS, DIVXS	B/W	
	MULU, DIVU, MULS, DIVS	W/L	
	MULU/U, MULS/U	L	
	EXTU, EXTS	W/L	
	TAS	B	
	MAC	—	
	LDMAC, STMAC	—	
	CLRMAC	—	
Logic operations	AND, OR, XOR, NOT	B/W/L	4
Shift	SHLL, SHLR, SHAL, SHAR, ROTL, ROTR, ROTXL, ROTXR	B/W/L	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	B	20
	BSET/EQ, BSET/NE, BCLR/EQ, BCLR/NE, BSTZ, BISTZ	B	
	BFLD, BFST	B	

Function	Instructions	Size	Types
Branch	BRA/BS, BRA/BC, BSR/BS, BSR/BC	B*3	9
	Bcc*5, JMP, BSR, JSR, RTS	—	
	RTS/L	L*5	
	BRA/S	—	
System control	TRAPA, RTE, SLEEP, NOP	—	10
	RTE/L	L*5	
	LDC, STC, ANDC, ORC, XORC	B/W/L	
		Total	87

[Legend]

B: Byte size

W: Word size

L: Longword size

Notes: 1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @−SP.
POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @−SP.

2. Size of data to be added with a displacement
3. Size of data to specify a branch condition
4. Bcc is the generic designation of a conditional branch instruction.
5. Size of general register to be restored
6. Not available in this LSI.

2.7.1 Instructions and Addressing Modes

Table 2.2 indicates the combinations of instructions and addressing modes that the H8SX CPU can use.

Table 2.2 Combinations of Instructions and Addressing Modes (1)

			Addressing Mode								
Classification	Instruction	Size	#xx	Rn	@ERn	@(d,ERn)	@(d, RnL.B/ Rn.W/ ERn.L)	@–ERn/ @ERn+/ @ERn–/ @+ERn	@aa:8	@aa:16/ @aa:32	—
Data transfer	MOV	B/W/L	S	SD	SD	SD	SD	SD		SD	
		B		S/D					S/D		
	MOVFPE, MOVTPE*12	B		S/D						S/D*1	
	POP, PUSH	W/L		S/D				S/D*2			
	LDM, STM	L		S/D				S/D*2			
	MOVA*4	B/W		S	S	S	S	S		S	
Block transfer	EEPMOV	B									SD*3
	MOVMD	B/W/L									SD*3
	MOVSD	B									SD*3
Arithmetic operations	ADD, CMP	B	S	D	D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	SUB	B	S		D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	ADDX, SUBX	B/W/L	S	SD							
		B/W/L	S		SD						
		B/W/L	S					SD*5			
	INC, DEC	B/W/L		D							

			Addressing Mode									
Classification	**Instruction**	**Size**	**#xx**	**Rn**	**@ERn**	**@(d,ERn)**	**@(d, RnL.B/ Rn.W/ ERn.L)**	**@–ERn/ @ERn+/ @ERn–/ @+ERn**	**@aa:8**	**@aa:16/ @aa:32**	**—**	
Arithmetic operations	ADDS, SUBS	L		D								
	DAA, DAS	B		D								
	MULXU, DIVXU	B/W	S:4	SD								
	MULU, DIVU	W/L	S:4	SD								
	MULXS, DIVXS	B/W	S:4	SD								
	MULS, DIVS	W/L	S:4	SD								
	NEG	B		D	D	D	D	D	D	D		
		W/L		D	D	D	D	D		D		
	EXTU, EXTS	W/L		D	D	D	D	D		D		
	TAS	B			D							
	MAC	—										
	CLRMAC	—										O
	LDMAC	—		S								
	STMAC	—		D								
Logic operations	AND, OR, XOR	B		S	D	D	D	D	D	D		
		B		D	S	S	S	S	S	S		
		B			SD	SD	SD	SD		SD		
		W/L	S	SD	SD	SD	SD	SD		SD		
	NOT	B		D	D	D	D	D	D	D		
		W/L		D	D	D	D	D		D		
Shift	SHLL, SHLR	B		D	D	D	D	D	D	D		
		B/W/L*[6]		D	D	D	D	D		D		
		B/W/L*[7]		D								
	SHAL, SHAR ROTL, ROTR ROTXL, ROTXR	B		D	D	D	D	D	D	D		
		W/L		D	D	D	D	D		D		

			Addressing Mode								
Classification	Instruction	Size	#xx	Rn	@ERn	@(d,ERn)	@(d, RnL.B/ Rn.W/ ERn.L)	@–ERn/ @ERn+/ @ERn–/ @+ERn	@aa:8	@aa:16/ @aa:32	—
Bit manipulation	BSET, BCLR, BNOT, BTST, BSET/cc, BCLR/cc	B		D	D				D	D	
	BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST, BSTZ, BISTZ	B		D	D				D	D	
	BFLD	B		D	S				S	S	
	BFST	B		S	D				D	D	
Branch	BRA/BS, BRA/BC*[8]	B			S				S	S	
	BSR/BS, BSR/BC*[8]	B			S				S	S	
System control	LDC (CCR, EXR)	B/W*[9]	S	S	S	S		S*[10]		S	
	LDC (VBR, SBR)	L		S							
	STC (CCR, EXR)	B/W*[9]		D	D	D		D*[11]		D	
	STC (VBR, SBR)	L		D							
	ANDC, ORC, XORC	B	S								
	SLEEP	—									O
	NOP	—									O

[Legend]

d: d:16 or d:32

S: Can be specified as a source operand.

D: Can be specified as a destination operand.

SD: Can be specified as either a source or destination operand or both.

S/D: Can be specified as either a source or destination operand.

S:4: 4-bit immediate data can be specified as a source operand.

Notes: 1. Only @aa:16 is available.

2. @ERn+ as a source operand and @−ERn as a destination operand

3. Specified by ER5 as a source address and ER6 as a destination address for data transfer.

4. Size of data to be added with a displacement
5. Only @ERn– is available
6. When the number of bits to be shifted is 1, 2, 4, 8, or 16
7. When the number of bits to be shifted is specified by 5-bit immediate data or a general register
8. Size of data to specify a branch condition
9. Byte when immediate or register direct, otherwise, word
10. Only @ERn+ is available
11. Only @–ERn is available
12. Not available in this LSI.

Table 2.2 Combinations of Instructions and Addressing Modes (2)

			Addressing Mode							
Classification	**Instruction**	**Size**	**@ERn**	**@(d,PC)**	**@(RnL.B/Rn.W/ERn.L, PC)**	**@aa:24**	**@ aa:32**	**@@ aa:8**	**@@vec:7**	**—**
Branch	BRA/BS, BRA/BC	—		O						
	BSR/BS, BSR/BC	—		O						
	Bcc	—		O						
	BRA	—		O	O					
	BRA/S	—		O*						
	JMP	—	O			O	O	O	O	
	BSR	—		O						
	JSR	—	O			O	O	O	O	
	RTS, RTS/L	—								O
System control	TRAPA	—								O
	RTE, RTE/L	—								O

[Legend]

d: d:8 or d:16

Note: * Only @(d:8, PC) is available.

2.7.2 Table of Instructions Classified by Function

Tables 2.4 to 2.11 summarize the instructions in each functional category. The notation used in these tables is defined in table 2.3.

Table 2.3 Operation Notation

Operation Notation	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extended control register
CCR	Condition-code register
VBR	Vector base register
SBR	Short address base register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#IMM	Immediate data
disp	Displacement
+	Addition
−	Subtraction
×	Multiplication
÷	Division
∧	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Move
~	Logical not (logical complement)
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

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

Table 2.4 Data Transfer Instructions

Instruction	Size	Function
MOV	B/W/L	#IMM → (EAd), (EAs) → (EAd) Transfers data between immediate data, general registers, and memory.
MOVFPE*	B	(EAs) → Rd
MOVTPE*	B	Rs → (EAs)
POP	W/L	@SP+ → Rn Restores the data from the stack to a general register.
PUSH	W/L	Rn → @−SP Saves general register contents on the stack.
LDM	L	@SP+ → Rn (register list) Restores the data from the stack to multiple general registers. Two, three, or four general registers which have serial register numbers can be specified.
STM	L	Rn (register list) → @−SP Saves the contents of multiple general registers on the stack. Two, three, or four general registers which have serial register numbers can be specified.
MOVA	B/W	EA → Rd Zero-extends and shifts the contents of a specified general register or memory data and adds them with a displacement. The result is stored in a general register.

Note: Not available in this LSI.

Table 2.5 Block Transfer Instructions

Instruction	Size	Function
EEPMOV.B EEPMOV.W	B	Transfers a data block. Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4 or R4L.
MOVMD.B	B	Transfers a data block. Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4.
MOVMD.W	W	Transfers a data block. Transfers word data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of word data to be transferred is specified by R4.
MOVMD.L	L	Transfers a data block. Transfers longword data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of longword data to be transferred is specified by R4.
MOVSD.B	B	Transfers a data block with zero data detection. Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4. When zero data is detected during transfer, the transfer stops and execution branches to a specified address.

Table 2.6 Arithmetic Operation Instructions

Instruction	Size	Function
ADD	B/W/L	(EAd) ± #IMM → (EAd), (EAd) ± (EAs) → (EAd)
SUB		Performs addition or subtraction on data between immediate data, general registers, and memory. Immediate byte data cannot be subtracted from byte data in a general register.
ADDX	B/W/L	(EAd) ± #IMM ± C → (EAd), (EAd) ± (EAs) ± C → (EAd)
SUBX		Performs addition or subtraction with carry on data between immediate data, general registers, and memory. The addressing mode which specifies a memory location can be specified as register indirect with post-decrement or register indirect.
INC	B/W/L	Rd ± 1 → Rd, Rd ± 2 → Rd
DEC		Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.)
ADDS	L	Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd
SUBS		Adds or subtracts the value 1, 2, or 4 to or from data in a general register.
DAA	B	Rd (decimal adjust) → Rd
DAS		Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 2-digit 4-bit BCD data.
MULXU	B/W	Rd × Rs → Rd
		Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits, or 16 bits × 16 bits → 32 bits.
MULU	W/L	Rd × Rs → Rd
		Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits, or 16 bits × 16 bits → 32 bits.
MULU/U	L	Rd × Rs → Rd
		Performs unsigned multiplication on data in two general registers (32 bits × 32 bits → upper 32 bits).
MULXS	B/W	Rd × Rs → Rd
		Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits, or 16 bits × 16 bits → 32 bits.
MULS	W/L	Rd × Rs → Rd
		Performs signed multiplication on data in two general registers: either 16 bits × 16 bits → 16 bits, or 32 bits × 32 bits → 32 bits.
MULS/U	L	Rd × Rs → Rd
		Performs signed multiplication on data in two general registers (32 bits × 32 bits → upper 32 bits).
DIVXU	B/W	Rd ÷ Rs → Rd
		Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder, or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.

Instruction	Size	Function
DIVU	W/L	Rd ÷ Rs → Rd Performs unsigned division on data in two general registers: either 16 bits ÷ 16 bits → 16-bit quotient, or 32 bits ÷ 32 bits → 32-bit quotient.
DIVXS	B/W	Rd ÷ Rs → Rd Performs signed division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder, or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.
DIVS	W/L	Rd ÷ Rs → Rd Performs signed division on data in two general registers: either 16 bits ÷ 16 bits → 16-bit quotient, or 32 bits ÷ 32 bits → 32-bit quotient.
CMP	B/W/L	(EAd) – #IMM, (EAd) – (EAs) Compares data between immediate data, general registers, and memory and stores the result in CCR.
NEG	B/W/L	0 – (EAd) → (EAd) Takes the two's complement (arithmetic complement) of data in a general register or the contents of a memory location.
EXTU	W/L	(EAd) (zero extension) → (EAd) Performs zero-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size. The lower 8 bits to word or longword, or the lower 16 bits to longword can be zero-extended.
EXTS	W/L	(EAd) (sign extension) → (EAd) Performs sign-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size. The lower 8 bits to word or longword, or the lower 16 bits to longword can be sign-extended.
TAS	B	@ERd – 0, 1 → (<bit 7> of @EAd) Tests memory contents, and sets the most significant bit (bit 7) to 1.
MAC	—	(EAs) × (EAd) + MAC → MAC Performs signed multiplication on memory contents and adds the result to MAC.
CLRMAC	—	0 → MAC Clears MAC to zero.
LDMAC	—	Rs → MAC Loads data from a general register to MAC.
STMAC	—	MAC → Rd Stores data from MAC to a general register.

Table 2.7 Logic Operation Instructions

Instruction	Size	Function
AND	B/W/L	(EAd) ∧ #IMM → (EAd), (EAd) ∧ (EAs) → (EAd) Performs a logical AND operation on data between immediate data, general registers, and memory.
OR	B/W/L	(EAd) ∨ #IMM → (EAd), (EAd) ∨ (EAs) → (EAd) Performs a logical OR operation on data between immediate data, general registers, and memory.
XOR	B/W/L	(EAd) ⊕ #IMM → (EAd), (EAd) ⊕ (EAs) → (EAd) Performs a logical exclusive OR operation on data between immediate data, general registers, and memory.
NOT	B/W/L	~ (EAd) → (EAd) Takes the one's complement of the contents of a general register or a memory location.

Table 2.8 Shift Operation Instructions

Instruction	Size	Function
SHLL	B/W/L	(EAd) (shift) → (EAd)
SHLR		Performs a logical shift on the contents of a general register or a memory location. The contents of a general register or a memory location can be shifted by 1, 2, 4, 8, or 16 bits. The contents of a general register can be shifted by any bits. In this case, the number of bits is specified by 5-bit immediate data or the lower 5 bits of the contents of a general register.
SHAL	B/W/L	(EAd) (shift) → (EAd)
SHAR		Performs an arithmetic shift on the contents of a general register or a memory location. 1-bit or 2-bit shift is possible.
ROTL	B/W/L	(EAd) (rotate) → (EAd)
ROTR		Rotates the contents of a general register or a memory location. 1-bit or 2-bit rotation is possible.
ROTXL	B/W/L	(EAd) (rotate) → (EAd)
ROTXR		Rotates the contents of a general register or a memory location with the carry bit. 1-bit or 2-bit rotation is possible.

Table 2.9 Bit Manipulation Instructions

Instruction	Size	Function
BSET	B	1 → (<bit-No.> of <EAd>) Sets a specified bit in the contents of a general register or a memory location to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BSET/cc	B	if cc, 1 → (<bit-No.> of <EAd>) If the specified condition is satisfied, this instruction sets a specified bit in a memory location to 1. The bit number can be specified by 3-bit immediate data, or by the lower three bits of a general register. The Z flag status can be specified as a condition.
BCLR	B	0 → (<bit-No.> of <EAd>) Clears a specified bit in the contents of a general register or a memory location to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BCLR/cc	B	if cc, 0 → (<bit-No.> of <EAd>) If the specified condition is satisfied, this instruction clears a specified bit in a memory location to 0. The bit number can be specified by 3-bit immediate data, or by the lower three bits of a general register. The Z flag status can be specified as a condition.
BNOT	B	~ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>) Inverts a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BTST	B	~ (<bit-No.> of <EAd>) → Z Tests a specified bit in the contents of a general register or a memory location and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BAND	B	C ∧ (<bit-No.> of <EAd>) → C ANDs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIAND	B	C ∧ [~ (<bit-No.> of <EAd>)] → C ANDs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.

Instruction	Size	Function
BOR	B	C ∨ (<bit-No.> of <EAd>) → C ORs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIOR	B	C ∨ [~ (<bit-No.> of <EAd>)] → C ORs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → C Exclusive-ORs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIXOR	B	C ⊕ [~ (<bit-No.> of <EAd>)] → C Exclusive-ORs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BLD	B	(<bit-No.> of <EAd>) → C Transfers a specified bit in the contents of a general register or a memory location to the carry flag. The bit number is specified by 3-bit immediate data.
BILD	B	~ (<bit-No.> of <EAd>) → C Transfers the inverse of a specified bit in the contents of a general register or a memory location to the carry flag. The bit number is specified by 3-bit immediate data.
BST	B	C → (<bit-No.> of <EAd>) Transfers the carry flag value to a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data.
BSTZ	B	Z → (<bit-No.> of <EAd>) Transfers the zero flag value to a specified bit in the contents of a memory location. The bit number is specified by 3-bit immediate data.
BIST	B	~ C → (<bit-No.> of <EAd>) Transfers the inverse of the carry flag value to a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data.

Instruction	Size	Function
BISTZ	B	~ Z → (<bit-No.> of <EAd>) Transfers the inverse of the zero flag value to a specified bit in the contents of a memory location. The bit number is specified by 3-bit immediate data.
BFLD	B	(EAs) (bit field) → Rd Transfers a specified bit field in memory location contents to the lower bits of a specified general register.
BFST	B	Rs → (EAd) (bit field) Transfers the lower bits of a specified general register to a specified bit field in memory location contents.

Table 2.10 Branch Instructions

Instruction	Size	Function
BRA/BS BRA/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a specified address.
BSR/BS BSR/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a subroutine at a specified address.
Bcc	—	Branches to a specified address if the specified condition is satisfied.
BRA/S	—	Branches unconditionally to a specified address after executing the next instruction. The next instruction should be a 1-word instruction except for the block transfer and branch instructions.
JMP	—	Branches unconditionally to a specified address.
BSR	—	Branches to a subroutine at a specified address.
JSR	—	Branches to a subroutine at a specified address.
RTS	—	Returns from a subroutine.
RTS/L	—	Returns from a subroutine, restoring data from the stack to multiple general registers.

Table 2.11 System Control Instructions

Instruction	Size	Function
TRAPA	—	Starts trap-instruction exception handling.
RTE	—	Returns from an exception-handling routine.
RTE/L	—	Returns from an exception-handling routine, restoring data from the stack to multiple general registers.
SLEEP	—	Causes a transition to a power-down state.
LDC	B/W	#IMM → CCR, (EAs) → CCR, #IMM → EXR, (EAs) → EXR Loads immediate data or the contents of a general register or a memory location to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid.
	L	Rs → VBR, Rs → SBR Transfers the general register contents to VBR or SBR.
STC	B/W	CCR → (EAd), EXR → (EAd) Transfers the contents of CCR or EXR to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid.
	L	VBR → Rd, SBR → Rd Transfers the contents of VBR or SBR to a general register.
ANDC	B	CCR ∧ #IMM → CCR, EXR ∧ #IMM → EXR Logically ANDs the CCR or EXR contents with immediate data.
ORC	B	CCR ∨ #IMM → CCR, EXR ∨ #IMM → EXR Logically ORs the CCR or EXR contents with immediate data.
XORC	B	CCR ⊕ #IMM → CCR, EXR ⊕ #IMM → EXR Logically exclusive-ORs the CCR or EXR contents with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

2.7.3 Basic Instruction Formats

The H8SX CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

Figure 2.14 shows examples of instruction formats.

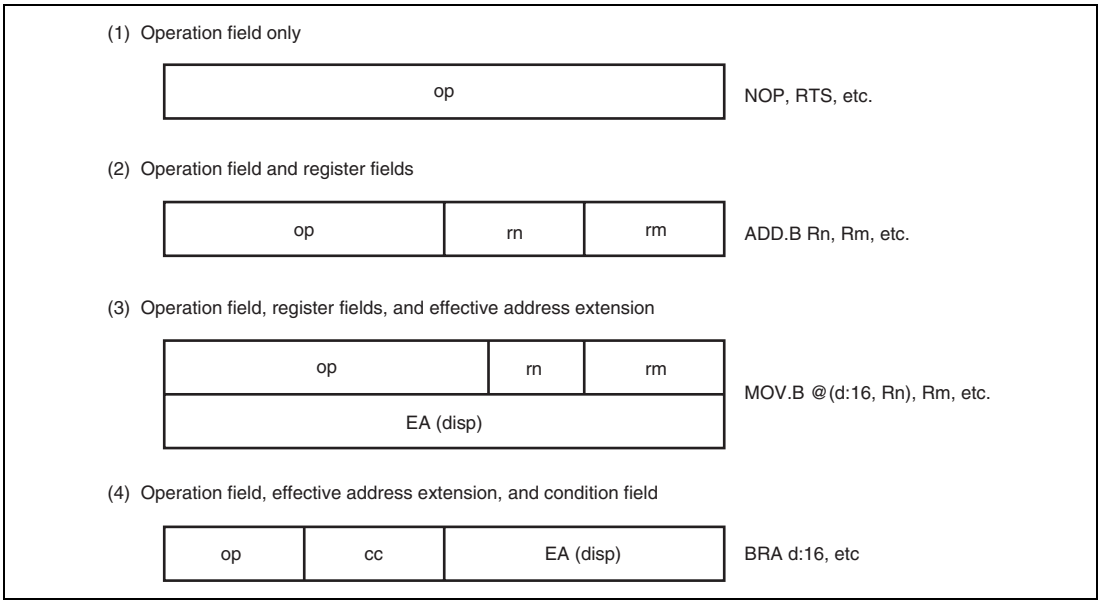


Figure 2.14 Instruction Formats

- Operation Field

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

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

  8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.
- Condition Field

  Specifies the branch condition of Bcc instructions.

2.8 Addressing Modes and Effective Address Calculation

The H8SX CPU supports the 11 addressing modes listed in table 2.12. Each instruction uses a subset of these addressing modes.

Bit manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

Table 2.12 Addressing Modes

No.	Addressing Mode	Symbol
1	Register direct	Rn
2	Register indirect	@ERn
3	Register indirect with displacement	@(d:2,ERn)/@(d:16,ERn)/@(d:32,ERn)
4	Index register indirect with displacement	@(d:16, RnL.B)/@(d:16,Rn.W)/@(d:16,ERn.L) @(d:32, RnL.B)/@(d:32,Rn.W)/@(d:32,ERn.L)
5	Register indirect with post-increment	@ERn+
	Register indirect with pre-decrement	@–ERn
	Register indirect with pre-increment	@+ERn
	Register indirect with post-decrement	@ERn–
6	Absolute address	@aa:8/@aa:16/@aa:24/@aa:32
7	Immediate	#xx:3/#xx:4/#xx:8/#xx:16/#xx:32
8	Program-counter relative	@(d:8,PC)/@(d:16,PC)
9	Program-counter relative with index register	@(RnL.B,PC)/@(Rn.W,PC)/@(ERn.L,PC)
10	Memory indirect	@@aa:8
11	Extended memory indirect	@@vec:7

2.8.1 Register Direct—Rn

The operand value is the contents of an 8-, 16-, or 32-bit general register which is specified by the register field in the instruction code. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

2.8.2 Register Indirect—@ERn

The operand value is the contents of the memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code.

In advanced mode, if this addressing mode is used in a branch instruction, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

2.8.3 Register Indirect with Displacement—@(d:2, ERn), @(d:16, ERn), or @(d:32, ERn)

The operand value is the contents of a memory location which is pointed to by the sum of the contents of an address register (ERn) and a 16- or 32-bit displacement. ERn is specified by the register field of the instruction code. The displacement is included in the instruction code and the 16-bit displacement is sign-extended when added to ERn.

This addressing mode has a short format (@(d:2, ERn)). The short format can be used when the displacement is 1, 2, or 3 and the operand is byte data, when the displacement is 2, 4, or 6 and the operand is word data, or when the displacement is 4, 8, or 12 and the operand is longword data.

2.8.4 Index Register Indirect with Displacement—@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)

The operand value is the contents of a memory location which is pointed to by the sum of the following operation result and a 16- or 32-bit displacement: a specified bits of the contents of an address register (RnL, Rn, ERn) specified by the register field in the instruction code are zero-extended to 32-bit data and multiplied by 1, 2, or 4. The displacement is included in the instruction code and the 16-bit displacement is sign-extended when added to ERn. If the operand is byte data, ERn is multiplied by 1. If the operand is word or longword data, ERn is multiplied by 2 or 4, respectively.

2.8.5 Register Indirect with Post-Increment, Pre-Decrement, Pre-Increment, or Post-Decrement—@ERn+, @−ERn, @+ERn, or @ERn−

(1) Register indirect with post-increment—@ERn+

The operand value is the contents of a memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After the memory location is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access.

(2) Register indirect with pre-decrement—@−ERn

The operand value is the contents of a memory location which is pointed to by the following operation result: the value 1, 2, or 4 is subtracted from the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After that, the operand value is stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access.

(3) Register indirect with pre-increment—@+ERn

The operand value is the contents of a memory location which is pointed to by the following operation result: the value 1, 2, or 4 is added to the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After that, the operand value is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access.

(4) Register indirect with post-decrement—@ERn−

The operand value is the contents of a memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field in the instruction code. After the memory location is accessed, 1, 2, or 4 is subtracted from the address register contents and the subtraction result is stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access.

In the case of (1) to (4) above, if the contents of a general register which is also used as an address register is written to memory using this addressing mode, data to be written is the contents of the general register after calculating an effective address. If the same general register is specified in an instruction and two effective addresses are calculated, the contents of the general register after the first calculation of an effective address is used in the second calculation of an effective address.

Example 1:

```
MOV.W    R0, @ER0+
```

When ER0 before execution is H'12345678, H'567A is written at H'12345678.

Example 2:

```
MOV.B    @ER0+, @ER0+
```

When ER0 before execution is H'00001000, H'00001000 is read and the contents is written at H'00001001.

After execution, ER0 is H'00001002.

2.8.6 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32

The operand value is the contents of a memory location which is pointed to by an absolute address included in the instruction code.

There are 8-bit (@aa:8), 16-bit (@aa:16), 24-bit (@aa:24), and 32-bit (@aa:32) absolute addresses.

To access the data area, the absolute address of 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) is used. For an 8-bit absolute address, the upper 24 bits are specified by SBR. For a 16-bit absolute address, the upper 16 bits are sign-extended. A 32-bit absolute address can access the entire address space.

To access the program area, the absolute address of 24 bits (@aa:24) or 32 bits (@aa:32) is used. For a 24-bit absolute address, the upper 8 bits are all assumed to be 0 (H'00).

Table 2.13 shows the accessible absolute address ranges.

Table 2.13 Absolute Address Access Ranges

Absolute Address		Normal Mode	Middle Mode	Advanced Mode	Maximum Mode
Data area	8 bits (@aa:8)	A consecutive 256-byte area (the upper address is set in SBR)			
	16 bits (@aa:16)	H'0000 to H'FFFF	H'000000 to H'007FFF, H'FF8000 to H'FFFFFF	H'00000000 to H'00007FFF, H'FFFF8000 to H'FFFFFFFF	
	32 bits (@aa:32)			H'00000000 to H'FFFFFFFF	
Program area	24 bits (@aa:24)		H'000000 to H'FFFFFF	H'00000000 to H'00FFFFFF	
	32 bits (@aa:32)			H'00000000 to H'00FFFFFF	H'00000000 to H'FFFFFFFF

2.8.7 Immediate—#xx

The operand value is 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) data included in the instruction code.

This addressing mode has short formats in which 3- or 4-bit immediate data can be used.

When the size of immediate data is less than that of the destination operand value (byte, word, or longword) the immediate data is zero-extended.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, for specifying a bit number. The BFLD and BFST instructions contain 8-bit immediate data in the instruction code, for specifying a bit field. The TRAPA instruction contains 2-bit immediate data in the instruction code, for specifying a vector address.

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

This mode is used in the Bcc and BSR instructions. The operand value is a 32-bit branch address, which is the sum of an 8- or 16-bit displacement in the instruction code and the 32-bit address of the PC contents. The 8-bit or 16-bit displacement is sign-extended to 32 bits when added to the PC contents. The PC contents to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is −126 to +128 bytes (−63 to +64 words) or −32766 to +32768 bytes (−16383 to +16384 words) from the branch instruction. The resulting value should be an even number. In advanced mode, only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00).

2.8.9 Program-Counter Relative with Index Register—@(RnL.B, PC), @(Rn.W, PC), or @(ERn.L, PC)

This mode is used in the Bcc and BSR instructions. The operand value is a 32-bit branch address, which is the sum of the following operation result and the 32-bit address of the PC contents: the contents of an address register specified by the register field in the instruction code (RnL, Rn, or ERn) is zero-extended and multiplied by 2. The PC contents to which the displacement is added is the address of the first byte of the next instruction. In advanced mode, only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00).

2.8.10 Memory Indirect—@@aa:8

This mode can be used by the JMP and JSR instructions. The operand value is a branch address, which is the contents of a memory location pointed to by an 8-bit absolute address in the instruction code.

The upper bits of an 8-bit absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in other modes).

In normal mode, the memory location is pointed to by word-size data and the branch address is 16 bits long. In other modes, the memory location is pointed to by longword-size data. In middle or advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

Note that the top part of the address range is also used as the exception handling vector area. A vector address of an exception handling other than a reset or a CPU address error can be changed by VBR.

Figure 2.15 shows an example of specification of a branch address using this addressing mode.

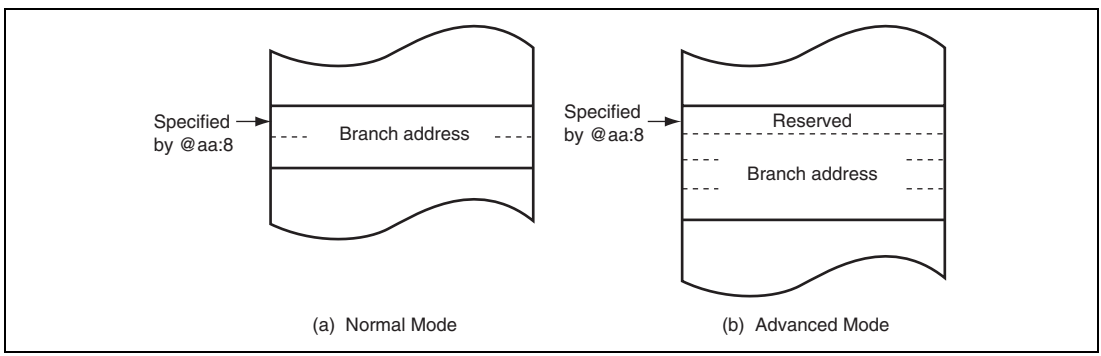


Figure 2.15 Branch Address Specification in Memory Indirect Mode

2.8.11 Extended Memory Indirect—@@vec:7

This mode can be used by the JMP and JSR instructions. The operand value is a branch address, which is the contents of a memory location pointed to by the following operation result: the sum of 7-bit data in the instruction code and the value of H'80 is multiplied by 2 or 4.

The address range to store a branch address is H'0100 to H'01FF in normal mode and H'000200 to H'0003FF in other modes. In assembler notation, an address to store a branch address is specified.

In normal mode, the memory location is pointed to by word-size data and the branch address is 16 bits long. In other modes, the memory location is pointed to by longword-size data. In middle or advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

2.8.12 Effective Address Calculation

Tables 2.14 and 2.15 show how effective addresses are calculated in each addressing mode. The lower bits of the effective address are valid and the upper bits are ignored (zero extended or sign extended) according to the CPU operating mode.

The valid bits in middle mode are as follows:

- The lower 16 bits of the effective address are valid and the upper 16 bits are sign-extended for the transfer and operation instructions.
- The lower 24 bits of the effective address are valid and the upper eight bits are zero-extended for the branch instructions.

Table 2.14 Effective Address Calculation for Transfer and Operation Instructions

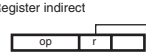
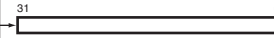
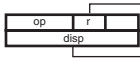
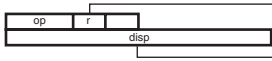
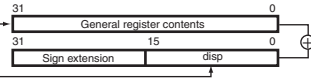
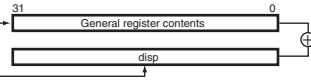
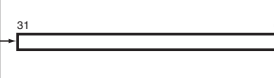
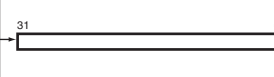
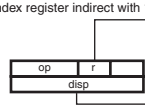
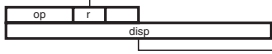
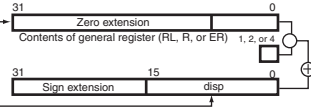
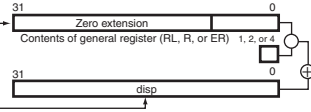
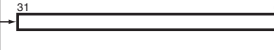
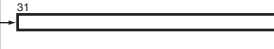
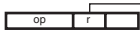
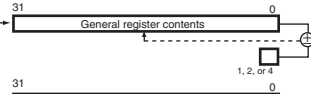
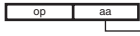
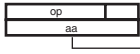
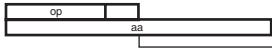
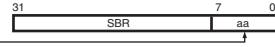
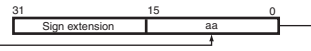
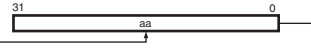
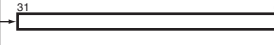
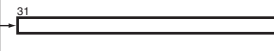
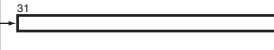
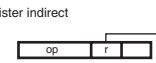
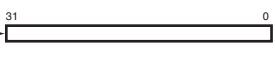
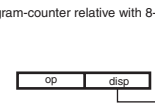
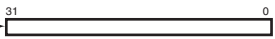
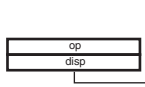
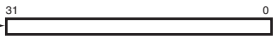
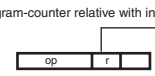
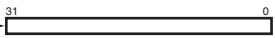
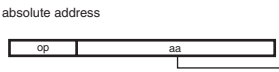
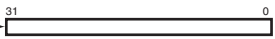
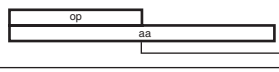
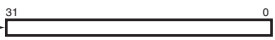
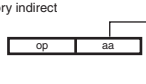
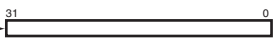
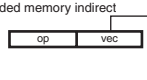
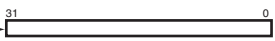
No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Immediate 		
2	Register direct 		
3	Register indirect 		
4	Register indirect with 16-bit displacement  Register indirect with 32-bit displacement 	 	 
5	Index register indirect with 16-bit displacement  Index register indirect with 32-bit displacement 	 	 
6	Register indirect with post-increment or post-decrement  Register indirect with pre-increment or pre-decrement 	 	 
7	8-bit absolute address  16-bit absolute address  32-bit absolute address 	  	  

Table 2.15 Effective Address Calculation for Branch Instructions

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Register indirect		
2	Program-counter relative with 8-bit displacement		
	Program-counter relative with 16-bit displacement		
3	Program-counter relative with index register		
4	24-bit absolute address		
	32-bit absolute address		
5	Memory indirect		
6	Extended memory indirect		

2.8.13 MOVA Instruction

The MOVA instruction stores the effective address in a general register.

1. Firstly, data is obtained by the addressing mode shown in item 2of table 2.14.
2. Next, the effective address is calculated using the obtained data as the index by the addressing mode shown in item 5 of table 2.14. The obtained data is used instead of the general register. The result is stored in a general register. For details, see H8SX Family Software Manual.

2.9 Processing States

The H8SX CPU has five main processing states: the reset state, exception-handling state, program execution state, bus-released state, and program stop state. Figure 2.16 indicates the state transitions.

- Reset state

  In this state the CPU and internal peripheral modules are all initialized and stopped. When the $\overline{\text{RES}}$ input goes low, all current processing stops and the CPU enters the reset state. All interrupts are masked in the reset state. Reset exception handling starts when the $\overline{\text{RES}}$ signal changes from low to high. For details, refer to section 4, Exception Handling.

  The reset state can also be entered by a watchdog timer overflow when available.
- Exception-handling state

  The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to activation of an exception source, such as, a reset, trace, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception handling vector table and branches to that address. For further details, refer to section 4, Exception Handling.
- Program execution state

  In this state the CPU executes program instructions in sequence.
- Bus-released state

  The bus-released state occurs when the bus has been released in response to a bus request from a bus master other than the CPU. While the bus is released, the CPU halts operations.
- Program stop state

  This is a power-down state in which the CPU stops operating. The program stop state occurs when a SLEEP instruction is executed or the CPU enters hardware standby mode. For details, refer to section 20, Power-Down Modes.

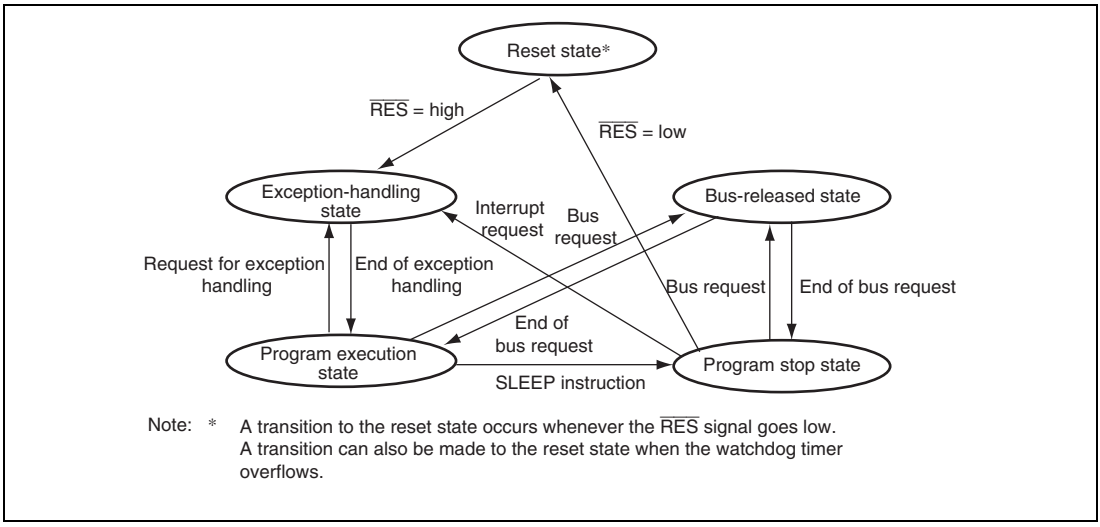


Figure 2.16 State Transitions

Section 3 MCU Operating Modes

3.1 Operating Mode Selection

This LSI has seven operating modes (modes 1 to 7). The operating mode is selected by the setting of mode pins (MD2 to MD0). Table 3.1 lists MCU operating mode settings.

Table 3.1 MCU Operating Mode Settings

MCU Operating Mode	MD2	MD1	MD0	CPU Operating Mode	Address Space	LSI Initiation Mode	On-Chip ROM	External Data Bus Width	
								Default	Max.
1	0	0	1	Advanced	16 Mbytes	User boot mode	Enabled	—	16 bits
2	0	1	0			Boot mode	Enabled	—	16 bits
3	0	1	1	Reserved (setting prohibited)					
4	1	0	0	Advanced	16 Mbytes	On-chip ROM disabled extended mode	Disabled	16 bits	16 bits
5	1	0	1				Disabled	8 bits	16 bits
6	1	1	0			On-chip ROM enabled extended mode	Enabled	8 bits	16 bits
7	1	1	1			Single-chip mode	Enabled	—	16 bits

In this LSI, an advanced mode as the CPU operating mode and a 16-Mbyte address space are available. The initial bus widths are eight or 16 bits. As the LSI initiation mode, the external extended mode, on-chip ROM initiation mode single-chip initiation mode can be selected.

Modes 1 and 2 are the user boot mode and the boot mode in which the flash memory can be programmed and erased. For details on the user boot mode and the boot mode, refer to section 18, Flash Memory (0.18-µm F-ZTAT Version).

Mode 7 is a single-chip mode. All I/O ports can be used as general input/output ports. The external address space cannot be accessed in the initial state, but setting the EXPE bit in the system control register (SYSCR) to 1 enables the external address space. After the external address space is enabled, ports D, E, and F can be used as an address output bus and ports H and I as a data bus by specifying the data direction register (DDR) for each port.

Modes 4 to 6 are external extended modes, in which the external memory and devices can be accessed. In the external extended modes, the external address space can be designated as 8-bit or 16-bit address space for each area by the bus controller after starting program execution.

If 16-bit address space is designated for any one area, it is called the 16-bit bus widths mode. If 8-bit address space is designated for all areas, it is called the 8-bit bus width mode.

3.2 Register Descriptions

The following registers are related to the operating mode setting.

- Mode control register (MDCR)
- System control register (SYSCR)

3.2.1 Mode Control Register (MDCR)

MDCR indicates the current operating mode. When MDCR is read from, the states of signals MD2 to MD0 are latched. This latch is canceled by a reset.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	MDS3	MDS2	MDS1	MDS0
Initial Value	0	1	0	1	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	1	0	1	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R	R

Note: * Determined by pins MD2 to MD0.

Bit	Bit Name	Initial Value	R/W	Descriptions
15	—	0	R	Reserved
14	—	1	R	These are read-only bits and cannot be modified.
13	—	0	R	
12	—	1	R	
11	MDS3	Undefined*	R	Mode Select 3 to 0
10	MDS2	Undefined*	R	These bits indicate the operating mode selected by the mode pins (MD2 to MD0) (see table 3.2).
9	MDS1	Undefined*	R	
8	MDS0	Undefined*	R	When MDCR is read, the signal levels input on pins MD2 to MD0 are latched into these bits. These latches are released by a reset.
7	—	0	R	Reserved
6	—	1	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	—	1	R	
3	—	Undefined*	R	
2	—	Undefined*	R	
1	—	Undefined*	R	
0	—	Undefined*	R	

Note: * Determined by pins MD2 to MD0.

Table 3.2 Settings of Bits MDS3 to MDS0

MCU Operating Mode	Mode Pins			MDCR			
	MD2	MD1	MD0	MDS3	MDS2	MDS1	MDS0
1	0	0	1	1	1	0	1
2	0	1	0	1	1	0	0
4	1	0	0	0	0	1	0
5	1	0	1	0	0	0	1
6	1	1	0	0	1	0	1
7	1	1	1	0	1	0	0

3.2.2 System Control Register (SYSCR)

SYSCR controls MAC saturation operation, selects bus width mode for instruction fetch, sets external bus mode, enables/disables the on-chip RAM, and selects the DTC address mode.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	MACS	—	FETCHMD	—	EXPE	RAME
Initial Value	1	1	0	1	0	0	Undefined*	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
Bit Name	—	—	—	—	—	—	DTCMD	—
Initial Value	0	0	0	0	0	0	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * The initial value depends on the startup mode.

Bit	Bit Name	Initial Value	R/W	Descriptions
15, 14	—	All 1	R/W	Reserved These bits are always read as 1. The write value should always be 1.
13	MACS	0	R/W	MAC Saturation Operation Control Selects either saturation operation or non-saturation operation for the MAC instruction. 0: MAC instruction is non-saturation operation 1: MAC instruction is saturation operation
12	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
11	FETCHMD	0	R/W	Instruction Fetch Mode Select This LSI can prefetch an instruction in units of 16 bits or 32 bits. Select the bus width for instruction fetch depending on the used memory for the storage of programs*[1]. 0: 32-bit mode 1: 16-bit mode

Bit	Bit Name	Initial Value	R/W	Descriptions
10	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
9	EXPE	Undefined*[2]	R/W	External Bus Mode Enable Selects external bus mode. In external extended mode, this bit is fixed 1 and cannot be changed. In single-chip mode, the initial value of this bit is 0, and can be read from or written to. When writing 0 to this bit after reading EXPE = 1, an external bus cycle should not be executed. The external bus cycle may be carried out in parallel with the internal bus cycle depending on the setting of the write data buffer function. 0: External bus disabled 1: External bus enabled
8	RAME	1	R/W	RAM Enable Enables or disables the on-chip RAM. This bit is initialized when the reset state is released. Do not write 0 during access to the on-chip RAM. 0: On-chip RAM disabled 1: On-chip RAM enabled
7 to 2	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
1	DTCMD	1	R/W	DTC Mode Select Selects DTC operating mode. 0: DTC is in full-address mode 1: DTC is in short address mode
0	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.

Notes: 1. For details on instruction fetch mode, see section 2.3, Instruction Fetch.
2. The initial value depends on the LSI initiation mode.
Since operating modes 4, 5 and 6 are external extended modes, EXPE is 1.

3.3 Operating Mode Descriptions

3.3.1 Mode 1

This is the user boot mode for the flash memory. The LSI operates in the same way as in mode 7 except for programming and erasing of the flash memory. For details, refer to section 18, Flash Memory (0.18-µm F-ZTAT Version).

3.3.2 Mode 2

This is the boot mode for the flash memory. The LSI operates in the same way as in mode 7 except for programming and erasing of the flash memory. For details, refer to section 18, Flash Memory (0.18-µm F-ZTAT Version).

3.3.3 Mode 4

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is disabled.

The initial bus width mode immediately after a reset is 16 bits, with 16-bit access to all areas. Ports D, E, and F function as an address bus, ports H and I function as a data bus, and parts of ports A and B function as bus control signals. However, if all areas are designated as an 8-bit access space by the bus controller, the bus mode switches to eight bits, and only port H functions as a data bus.

3.3.4 Mode 5

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is disabled.

The initial bus width mode immediately after a reset is eight bits, with 8-bit access to all areas. Ports D, E, and F function as an address bus, port H functions as a data bus, and parts of ports A and B function as bus control signals. However, if any area is designated as a 16-bit access space by the bus controller, the bus width mode switches to 16 bits, and ports H and I function as a data bus.

3.3.5 Mode 6

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is enabled.

The initial bus width mode immediately after a reset is eight bits, with 8-bit access to all areas. Ports D, E, and F function as input ports, but they can be used as an address bus by specifying the data direction register (DDR) for each port. For details, refer to section 9, I/O Ports. Port H functions as a data bus, and parts of ports A and B function as bus control signals. However, if any area is designated as a 16-bit access space by the bus controller, the bus width mode switches to 16 bits, and ports H and I function as a data bus.

3.3.6 Mode 7

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is enabled. All I/O ports can be used as general input/output ports. The external address space cannot be accessed in the initial state, but setting the EXPE bit in the system control register (SYSCR) to 1 enables the external address space. After the external address space is enabled, ports D, E, and F can be used as an address output bus and ports H and I as a data bus by specifying the data direction register (DDR) for each port. For details, refer to section 9, I/O Ports.

3.3.7 Pin Functions

Table 3.3 lists the pin functions in each operating mode.

Table 3.3 Pin Functions in Each Operating Mode (Advanced Mode)

MCU Operation Mode	Port A			Port B		Port D	Port E	Port F		Port H	Port I
	PA7	PA6 to PA 3	PA2 to PA 0	PB3 to PB 1	PB0			PF4 to PF0	PF7 to PF 5		
1	P*/C	P*/C	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	P*/D	P*/D
2	P*/C	P*/C	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	P*/D	P*/D
4	P/C*	P/C*	P*/C	P*/C	P/C*	A	A	A	P*/A	D	P/D*
5	P/C*	P/C*	P*/C	P*/C	P/C*	A	A	A	P*/A	D	P*/D
6	P/C*	P/C*	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	D	P*/D
7	P*/C	P*/C	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	P*/D	P*/D

[Legend]

P: I/O port

A: Address bus output

D: Data bus input/output

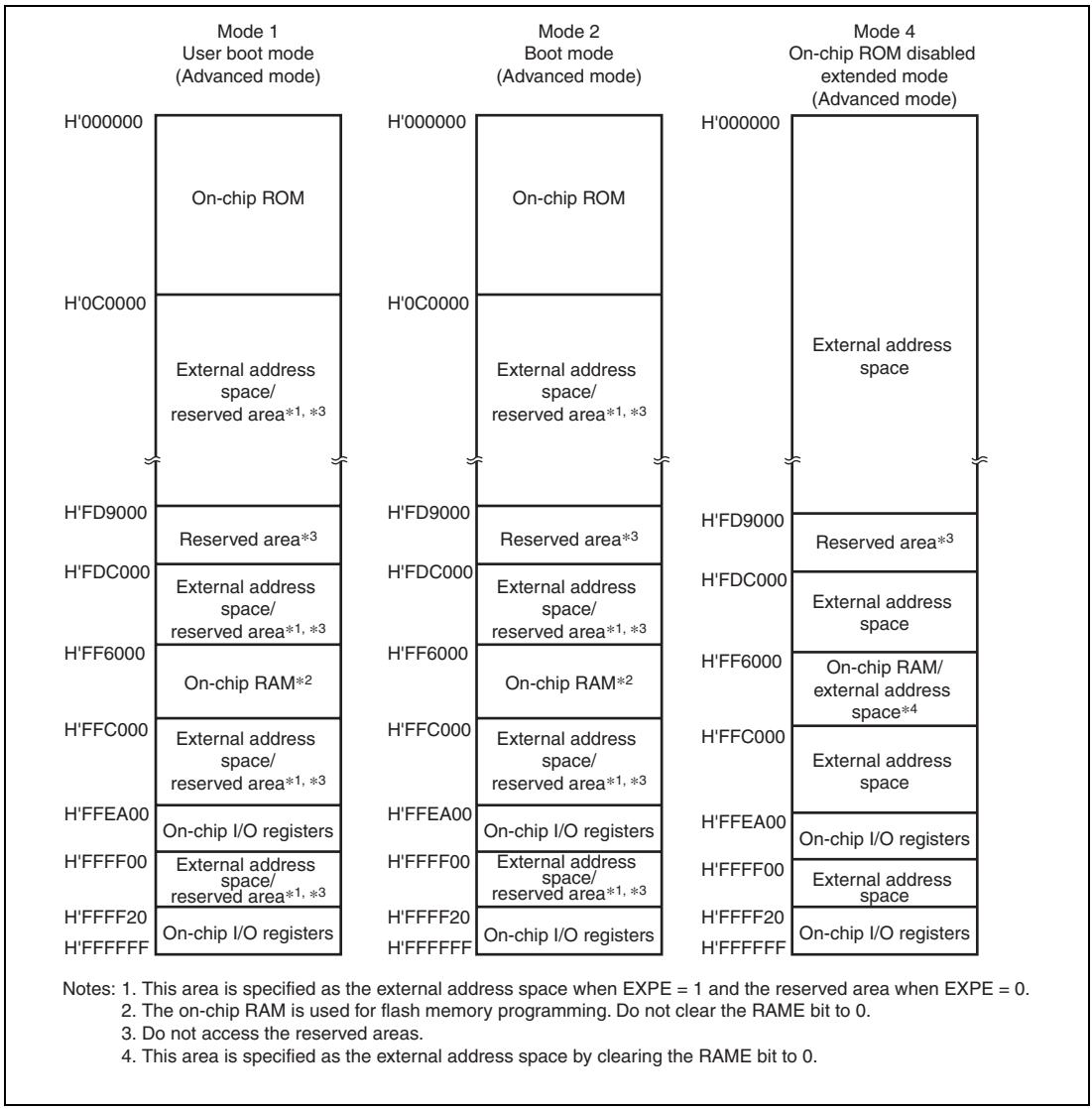
C: Control signals, clock input/output

*: Immediately after a reset

3.4 Address Map

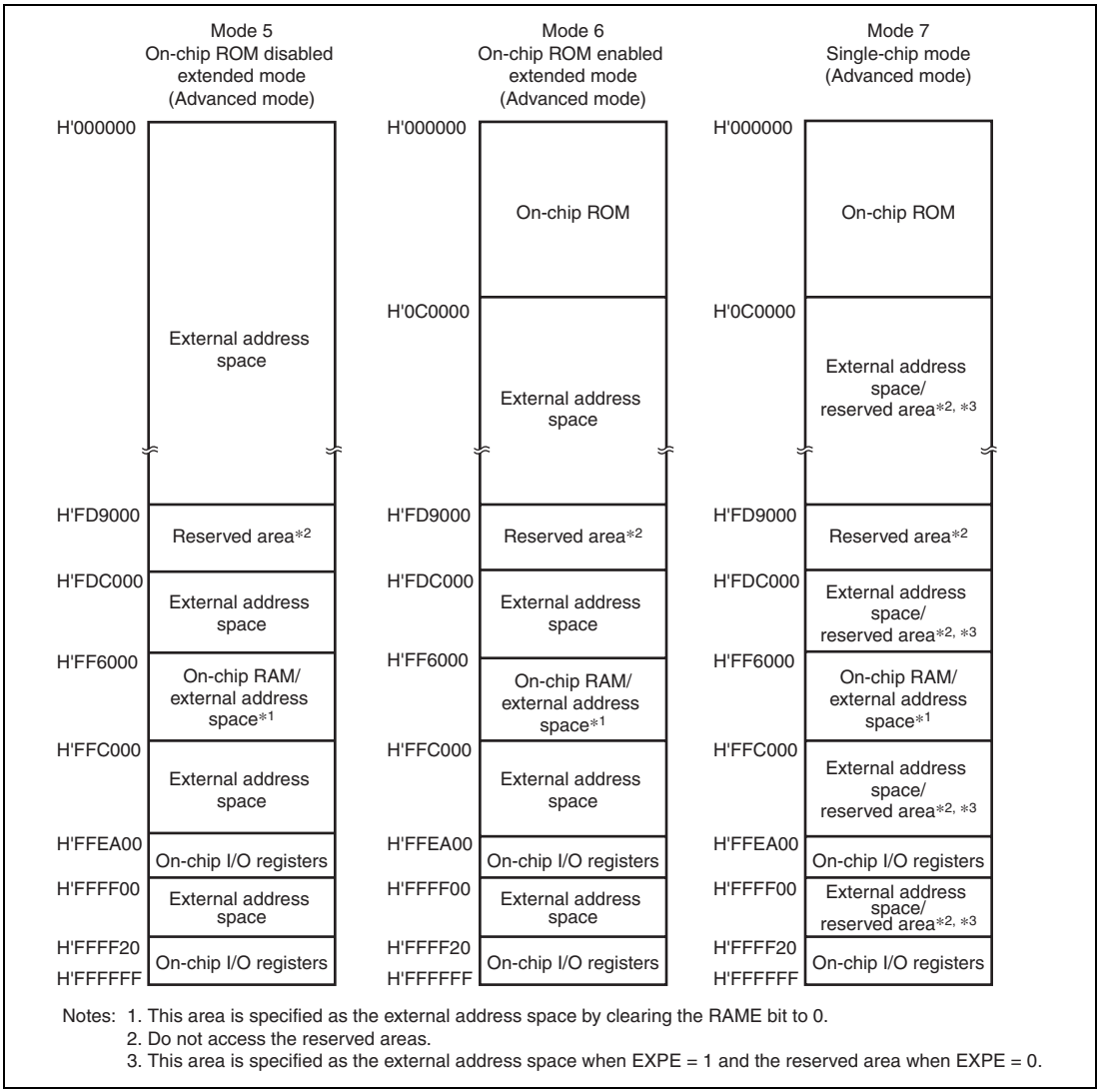
3.4.1 Address Map

Figure 3.1 show the address map in each operating mode.



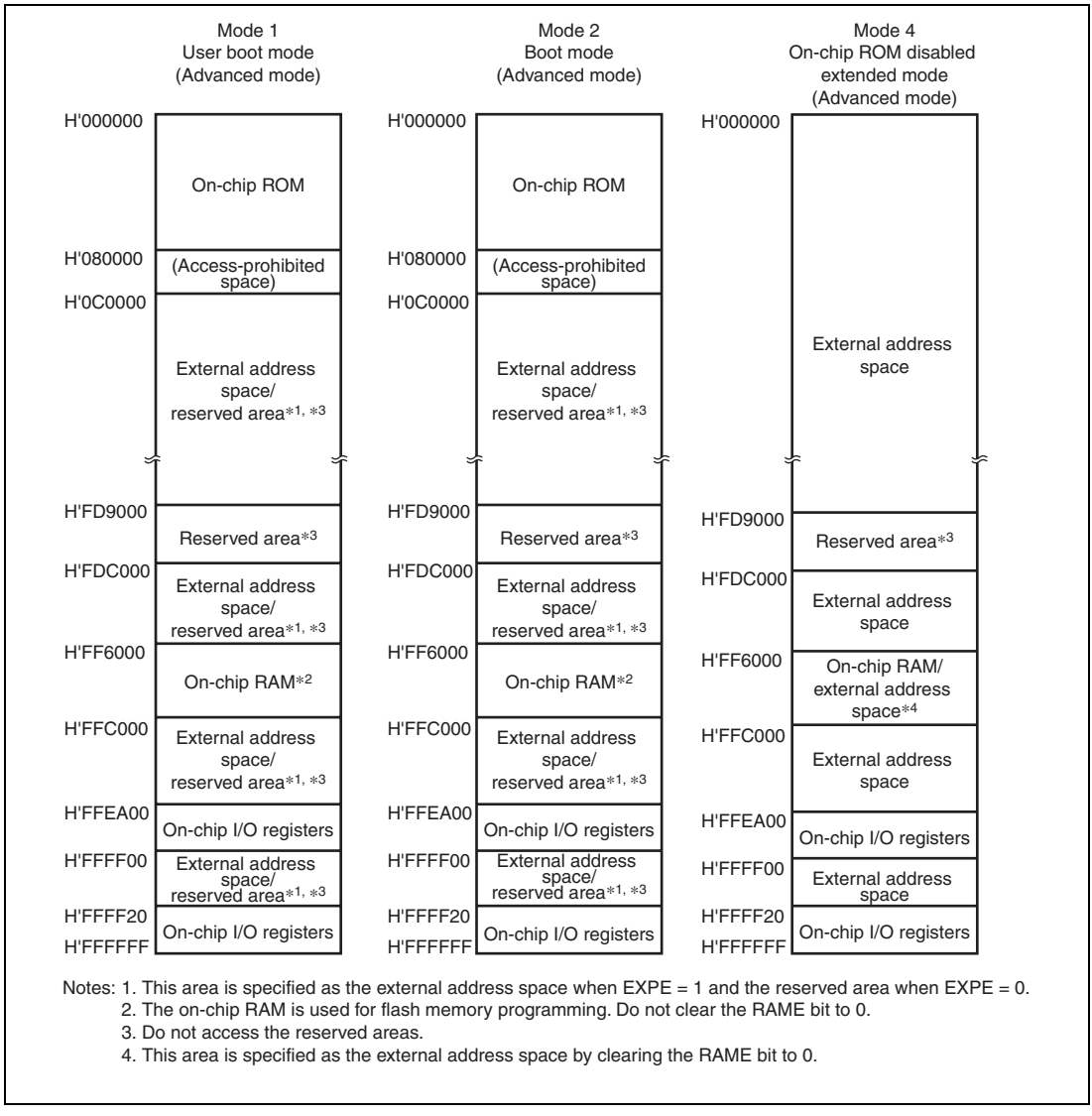
Notes: 1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.
2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0.
3. Do not access the reserved areas.
4. This area is specified as the external address space by clearing the RAME bit to 0.

Figure 3.1 Address Map in each Operating Mode of H8SX/1657C (1)



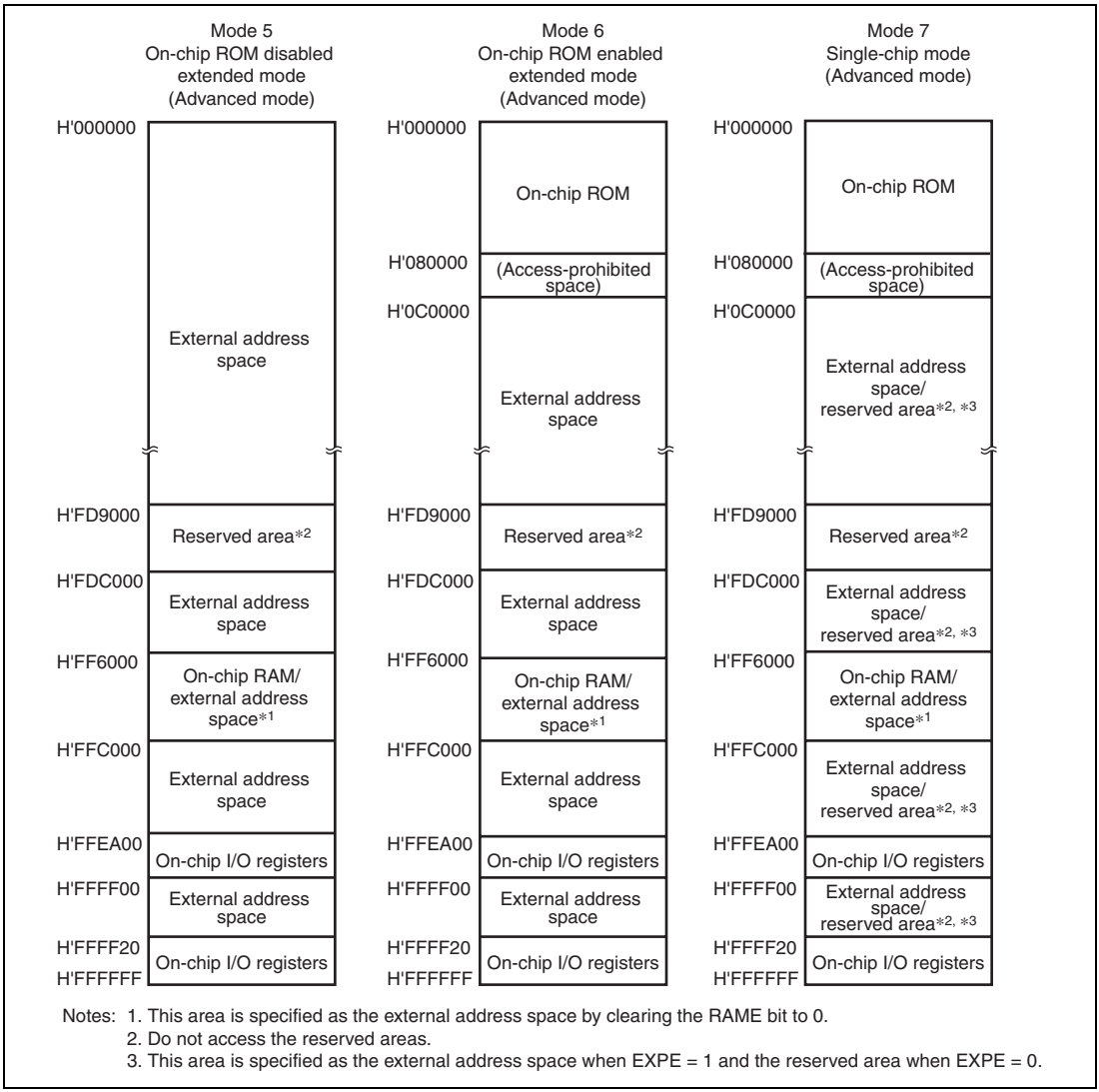
Notes: 1. This area is specified as the external address space by clearing the RAME bit to 0.
2. Do not access the reserved areas.
3. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.

Figure 3.1 Address Map in each Operating Mode of H8SX/1657C (2)



Notes: 1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.
2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0.
3. Do not access the reserved areas.
4. This area is specified as the external address space by clearing the RAME bit to 0.

Figure 3.1 Address Map in each Operating Mode of H8SX/1656C (3)



Notes: 1. This area is specified as the external address space by clearing the RAME bit to 0.
2. Do not access the reserved areas.
3. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.

Figure 3.1 Address Map in each Operating Mode of H8SX/1656C (4)

Section 4 Exception Handling

4.1 Exception Handling Types and Priority

As table 4.1 indicates, exception handling is caused by a reset, a trace, an address error, an interrupt, a trap instruction, a sleep instruction, and an illegal instruction (general illegal instruction or slot illegal instruction). Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Exception sources, the stack structure, and operation of the CPU vary depending on the interrupt control mode. For details on the interrupt control mode, see section 5, Interrupt Controller.

Table 4.1 Exception Types and Priority

Priority	Exception Type	Exception Handling Start Timing
High ▲	Reset	Exception handling starts at the timing of level change from low to high on the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. The CPU enters the reset state when the $\overline{\text{RES}}$ pin is low.
	Illegal instruction	Exception handling starts when an undefined code is executed.
	Trace*[1]	Exception handling starts after execution of the current instruction or exception handling, if the trace (T) bit in EXR is set to 1.
	Address error	After an address error has occurred, exception handling starts on completion of instruction execution.
	Interrupt	Exception handling starts after execution of the current instruction or exception handling, if an interrupt request has occurred.*[2]
	Sleep instruction	Exception handling starts by execution of a sleep instruction (SLEEP), if the SSBY bit in SBYCR is set to 0 and the SLPIE bit in SBYCR is set to 1.
▼ Low	Trap instruction*[3]	Exception handling starts by execution of a trap instruction (TRAPA).

Notes: 1. Traces are enabled only in interrupt control mode 2. Trace exception handling is not executed after execution of an RTE instruction.
2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
3. Trap instruction exception handling requests and sleep instruction exception handling requests are accepted at all times in the program execution state.

4.2 Exception Sources and Exception Handling Vector Table

Different vector table address offsets are assigned to different exception sources. The vector table addresses are calculated from the contents of the vector base register (VBR) and vector table address offset of the vector number. The start address of the exception service routine is fetched from the exception handling vector table indicated by this vector table address.

Table 4.2 shows the correspondence between the exception sources and vector table address offsets. Table 4.3 shows the calculation method of exception handling vector table addresses.

Table 4.2 Exception Handling Vector Table

		Vector Table Address Offset*[1]	
Exception Source	**Vector Number**	**Normal Mode*[2]**	**Advanced, Middle*[2], Maximum*[2] Modes**
Reset	0	H'0000 to H'0001	H'0000 to H'0003
Reserved for system use	1	H'0002 to H'0003	H'0004 to H'0007
	2	H'0004 to H'0005	H'0008 to H'000B
	3	H'0006 to H'0007	H'000C to H'000F
Illegal instruction	4	H'0008 to H'0009	H'0010 to H'0013
Trace	5	H'000A to H'000B	H'0014 to H'0017
Reserved for system use	6	H'000C to H'000D	H'0018 to H'001B
Interrupt (NMI)	7	H'000E to H'000F	H'001C to H'001F
Trap instruction (#0)	8	H'0010 to H'0011	H'0020 to H'0023
(#1)	9	H'0012 to H'0013	H'0024 to H'0027
(#2)	10	H'0014 to H'0015	H'0028 to H'002B
(#3)	11	H'0016 to H'0017	H'002C to H'002F
CPU address error	12	H'0018 to H'0019	H'0030 to H'0033
DMA address error*[3]	13	H'001A to H'001B	H'0034 to H'0037
Reserved for system use	14	H'001C to H'001D	H'0038 to H'003B
	27	H'0022 to H'0023	H'0044 to H'0047
Sleep instruction	18	H'0024 to H'0025	H'0048 to H'004B
Reserved for system use	19	H'0026 to H'0027	H'004C to H'004F
	23	H'002E to H'002F	H'005C to H'005F

Exception Source		Vector Number	Vector Table Address Offset*[1] Normal Mode*[2]	Vector Table Address Offset*[1] Advanced, Middle*[2], Maximum*[2] Modes
User area (not used)		24	H'0030 to H'0031	H'0060 to H'0063
		⋮	⋮	⋮
		63	H'007E to H'007F	H'00FC to H'00FF
External interrupt	IRQ0	64	H'0080 to H'0081	H'0100 to H'0103
	IRQ1	65	H'0082 to H'0083	H'0104 to H'0107
	IRQ2	66	H'0084 to H'0085	H'0108 to H'010B
	IRQ3	67	H'0086 to H'0087	H'010C to H'010F
	IRQ4	68	H'0088 to H'0089	H'0110 to H'0113
	IRQ5	69	H'008A to H'008B	H'0114 to H'0117
	IRQ6	70	H'008C to H'008D	H'0118 to H'011B
	IRQ7	71	H'008E to H'008F	H'011C to H'011F
	IRQ8	72	H'0090 to H'0091	H'0120 to H'0123
	IRQ9	73	H'0092 to H'0093	H'0124 to H'0127
	IRQ10	74	H'0094 to H'0095	H'0128 to H'012B
	IRQ11	75	H'0096 to H'0097	H'012C to H'012F
Reserved for system use		76	H'0098 to H'0099	H'0130 to H'0133
		⋮	⋮	⋮
		79	H'009E to H'009F	H'013C to H'013F
Internal interrupt*[4]		80	H'00A0 to H'00A1	H'0140 to H'0143
		⋮	⋮	⋮
		255	H'01FE to H'01FF	H'03FC to H'03FF

Notes: 1. Lower 16 bits of the address.
2. Not available in this LSI.
3. A DMA address error is generated by the DTC and DMAC.
4. For details of internal interrupt vectors, see section 5.5, Interrupt Exception Handling Vector Table.

Table 4.3 Calculation Method of Exception Handling Vector Table Address

Exception Source	Calculation Method of Vector Table Address
Reset, CPU address error	Vector table address = (vector table address offset)
Other than above	Vector table address = VBR + (vector table address offset)

[Legend]
VBR: Vector base register
Vector table address offset: See table 4.2.

4.3 Reset

A reset has priority over any other exception. When the $\overline{\text{RES}}$ pin goes low, all processing halts and this LSI enters the reset state. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms with the $\overline{\text{STBY}}$ pin driven high when the power is turned on. When operation is in progress, hold the $\overline{\text{RES}}$ pin low for at least 20 cycles.

The chip can also be reset by overflow of the watchdog timer. For details, see section 13, Watchdog Timer (WDT).

A reset initializes the internal state of the CPU and the registers of the on-chip peripheral modules. The interrupt control mode is 0 immediately after a reset.

4.3.1 Reset Exception Handling

When the $\overline{\text{RES}}$ pin goes high after being held low for the necessary time, this LSI starts reset exception handling as follows:

1. The internal state of the CPU and the registers of the on-chip peripheral modules are initialized, VBR is cleared to H'00000000, the T bit is cleared to 0 in EXR, and the I bits are set to 1 in EXR and CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

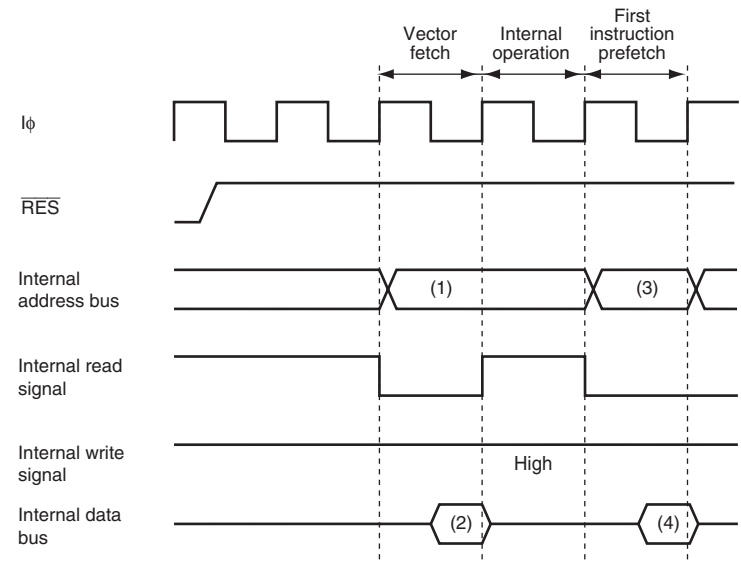
Figures 4.1 and 4.2 show examples of the reset sequence.

4.3.2 Interrupts after Reset

If an interrupt is accepted after a reset but before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests, including NMI, are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: MOV.L #xx: 32, SP).

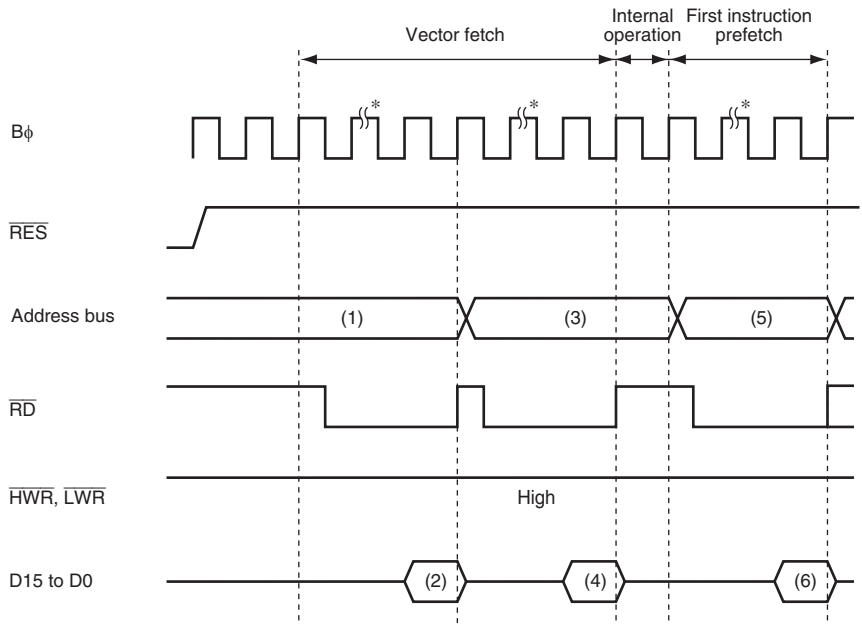
4.3.3 On-Chip Peripheral Functions after Reset Release

After the reset state is released, MSTPCRA and MSTPCRB are initialized to H'0FFF and H'FFFF, respectively, and all modules except the DTC and DMAC enter the module stop state. Consequently, on-chip peripheral module registers cannot be read from or written to. Register reading and writing is enabled when the module stop state is canceled.



(1): Reset exception handling vector address (when reset, (1) = H'000000)
(2): Start address (contents of reset exception handling vector address)
(3) Start address ((3) = (2))
(4) First instruction in the exception handling routine

Figure 4.1 Reset Sequence (On-chip ROM Enabled Advanced Mode)



(1)(3) Reset exception handling vector address (when reset, (1) = H'000000, (3) = H'000002)
(2)(4) Start address (contents of reset exception handling vector address)
(5) Start address ((5) = (2)(4))
(6) First instruction in the exception handling routine

Note: * Seven program wait cycles are inserted.

Figure 4.2 Reset Sequence
(16-Bit External Access in On-chip ROM Disabled Advanced Mode)

4.4 Traces

Traces are enabled in interrupt control mode 2. Trace mode is not activated in interrupt control mode 0, irrespective of the state of the T bit. Before changing interrupt control modes, the T bit must be cleared. For details on interrupt control modes, see section 5, Interrupt Controller.

If the T bit in EXR is set to 1, trace mode is activated. In trace mode, a trace exception occurs on completion of each instruction. Trace mode is not affected by interrupt masking by CCR. Table 4.4 shows the state of CCR and EXR after execution of trace exception handling. Trace mode is canceled by clearing the T bit in EXR to 0 during the trace exception handling. However, the T bit saved on the stack retains its value of 1, and when control is returned from the trace exception handling routine by the RTE instruction, trace mode resumes. Trace exception handling is not carried out after execution of the RTE instruction.

Interrupts are accepted even within the trace exception handling routine.

Table 4.4 Status of CCR and EXR after Trace Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**I2 to I0**	**T**
0	Trace exception handling cannot be used.			
2	1	—	—	0

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

4.5 Address Error

4.5.1 Address Error Source

Instruction fetch, stack operation, or data read/write shown in table 4.5 may cause an address error.

Table 4.5 Bus Cycle and Address Error

Bus Cycle			
Type	**Bus Master**	**Description**	**Address Error**
Instruction fetch	CPU	Fetches instructions from even addresses	No (normal)
		Fetches instructions from odd addresses	Occurs
		Fetches instructions from areas other than on-chip peripheral module space*[1]	No (normal)
		Fetches instructions from on-chip peripheral module space*[1]	Occurs
		Fetches instructions from external memory space in single-chip mode	Occurs
		Fetches instructions from access prohibited area.*[2]	Occurs
Stack operation	CPU	Accesses stack when the stack pointer value is even address	No (normal)
		Accesses stack when the stack pointer value is odd	Occurs
Data read/write	CPU	Accesses word data from even addresses	No (normal)
		Accesses word data from odd addresses	No (normal)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to access prohibited area*[2]	Occurs
Data read/write	DTC or DMAC	Accesses word data from even addresses	No (normal)
		Accesses word data from odd addresses	No (normal)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to access prohibited area*[2]	Occurs
Single address transfer	DMAC	Address access space is the external memory space for single address transfer	No (normal)
		Address access space is not the external memory space for single address transfer	Occurs

Notes: 1. For on-chip peripheral module space, see section 6, Bus Controller (BSC).
2. For the access prohibited area, refer to figure 3.1 in section 3.4, Address Map.

4.5.2 Address Error Exception Handling

When an address error occurs, address error exception handling starts after the bus cycle causing the address error ends and current instruction execution completes. The address error exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the address error is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Even though an address error occurs during a transition to an address error exception handling, the address error is not accepted. This prevents an address error from occurring due to stacking for exception handling, thereby preventing infinitive stacking.

If the SP contents are not a multiple of 2 when an address error exception handling occurs, the stacked values (PC, CCR, and EXR) are undefined.

When an address error occurs, the following is performed to halt the DTC and DMAC.

- The ERR bit of DTCCR in the DTC is set to 1.
- The ERRF bit of DMDR_0 in the DMAC is set to 1.
- The DTE bits of DMDRs for all channels in the DMAC are cleared to 0 to forcibly terminate transfer.

Table 4.6 shows the state of CCR and EXR after execution of the address error exception handling.

Table 4.6 Status of CCR and EXR after Address Error Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	7

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

4.6 Interrupts

4.6.1 Interrupt Sources

Interrupt sources are NMI, IRQ0 to IRQ11, and on-chip peripheral modules, as shown in table 4.7.

Table 4.7 Interrupt Sources

Type	Source	Number of Sources
NMI	NMI pin (external input)	1
IRQ0 to IRQ11	Pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ11}}$ (external input)	12
On-chip peripheral module	DMA controller (DMAC)	8
	Watchdog timer (WDT)	1
	A/D converter	1
	16-bit timer pulse unit (TPU)	26
	8-bit timer (TMR)	12
	Serial communications interface (SCI)	16

Different vector numbers and vector table offsets are assigned to different interrupt sources. For vector number and vector table offset, refer to table 5.2, Interrupt Sources, Vector Address Offsets, and Interrupt Priority in section 5, Interrupt Controller.

4.6.2 Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to eight priority/mask levels to enable multiple-interrupt control. The source to start interrupt exception handling and the vector address differ depending on the product. For details, refer to section 5, Interrupt Controller.

The interrupt exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the interrupt source is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

4.7 Instruction Exception Handling

There are three instructions that cause exception handling: trap instruction, sleep instruction, and illegal instruction.

4.7.1 Trap Instruction

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state. The trap instruction exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the vector number specified in the TRAPA instruction is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

A start address is read from the vector table corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 4.8 shows the state of CCR and EXR after execution of trap instruction exception handling.

Table 4.8 Status of CCR and EXR after Trap Instruction Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	—

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

4.7.2 Sleep Instruction Exception Handling

The sleep instruction exception handling starts when a sleep instruction is executed with the SSBY bit in SBYCR set to 0 and the SLPIE bit in SBYCR set to 1. The sleep instruction exception handling can always be executed in the program execution state. In the exception handling, the CPU operates as follows.

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the vector number specified in the SLEEP instruction is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Bus masters other than the CPU may gain the bus mastership after a sleep instruction has been executed. In such cases, the sleep instruction will be started when the transactions of a bus master other than the CPU has been completed and the CPU has gained the bus mastership.

Table 4.9 shows the state of CCR and EXR after execution of sleep instruction exception handling. For details, see section 20.9, Sleep Instruction Exception Handling.

Table 4.9 Status of CCR and EXR after Sleep Instruction Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	7

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

4.7.3 Exception Handling by Illegal Instruction

The illegal instructions are general illegal instructions and slot illegal instructions. The exception handling by the general illegal instruction starts when an undefined code is executed. The exception handling by the slot illegal instruction starts when a particular instruction (e.g. its code length is two words or more, or it changes the PC contents) at a delay slot (immediately after a delayed branch instruction) is executed. The exception handling by the general illegal instruction and slot illegal instruction is always executable in the program execution state.

The exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the occurred exception is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Table 4.10 shows the state of CCR and EXR after execution of illegal instruction exception handling.

Table 4.10 Status of CCR and EXR after Illegal Instruction Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	—

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

4.8 Stack Status after Exception Handling

Figure 4.3 shows the stack after completion of exception handling.

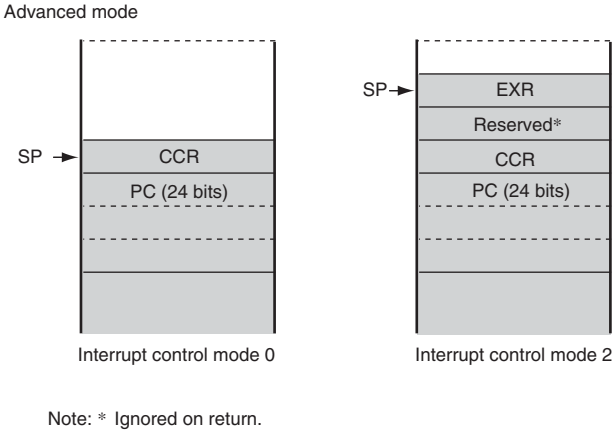


Figure 4.3 Stack Status after Exception Handling

4.9 Usage Note

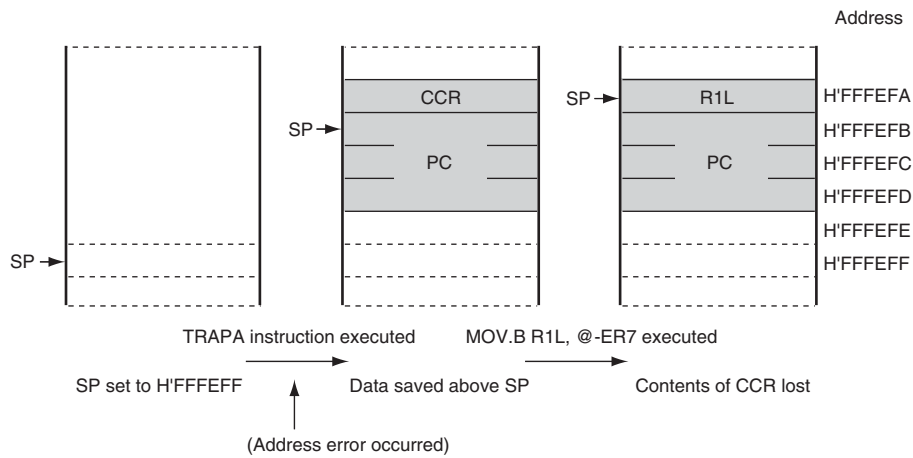
When performing stack-manipulating access, this LSI assumes that the lowest address bit is 0. The stack should always be accessed by a word transfer instruction or a longword transfer instruction, and the value of the stack pointer (SP: ER7) should always be kept even. Use the following instructions to save registers:

- PUSH.W Rn (or MOV.W Rn, @-SP)
- PUSH.L ERn (or MOV.L ERn, @-SP)

Use the following instructions to restore registers:

- POP.W Rn (or MOV.W @SP+, Rn)
- POP.L ERn (or MOV.L @SP+, ERn)

Performing stack manipulation while SP is set to an odd value leads to an address error. Figure 4.4 shows an example of operation when the SP value is odd.



[Legend]
CCR : Condition code register
PC : Program counter
R1L : General register R1L
SP : Stack pointer

Note: This diagram illustrates an example in which the interrupt control mode is 0, in advanced mode.

Figure 4.4 Operation when SP Value Is Odd

Section 5 Interrupt Controller

5.1 Features

- Two interrupt control modes

  Any of two interrupt control modes can be set by means of bits INTM1 and INTM0 in the interrupt control register (INTCR).
- Priority can be assigned by the interrupt priority register (IPR)

  IPR provides for setting interrupt priory. Eight levels can be set for each module for all interrupts except for the interrupt requests listed below. The following seven interrupt requests are given priority of 8, therefore they are accepted at all times.
  - NMI
  - Illegal instructions
  - Trace
  - Trap instructions
  - CPU address error
  - DMA address error (occurred in the DTC and DMAC)
  - Sleep instruction
- Independent vector addresses

  All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.
- Thirteen external interrupts

  NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge detection can be selected for NMI. Falling edge, rising edge, or both edge detection, or level sensing, can be selected for $\overline{\text{IRQ11}}$ to $\overline{\text{IRQ0}}$.
- DTC and DMAC control

  DTC and DMAC can be activated by means of interrupts.
- CPU priority control function

  The priority levels can be assigned to the CPU, DTC, and DMAC. The priority level of the CPU can be automatically assigned on an exception generation. Priority can be given to the CPU interrupt exception handling over that of the DTC and DMAC transfer.

A block diagram of the interrupt controller is shown in figure 5.1.

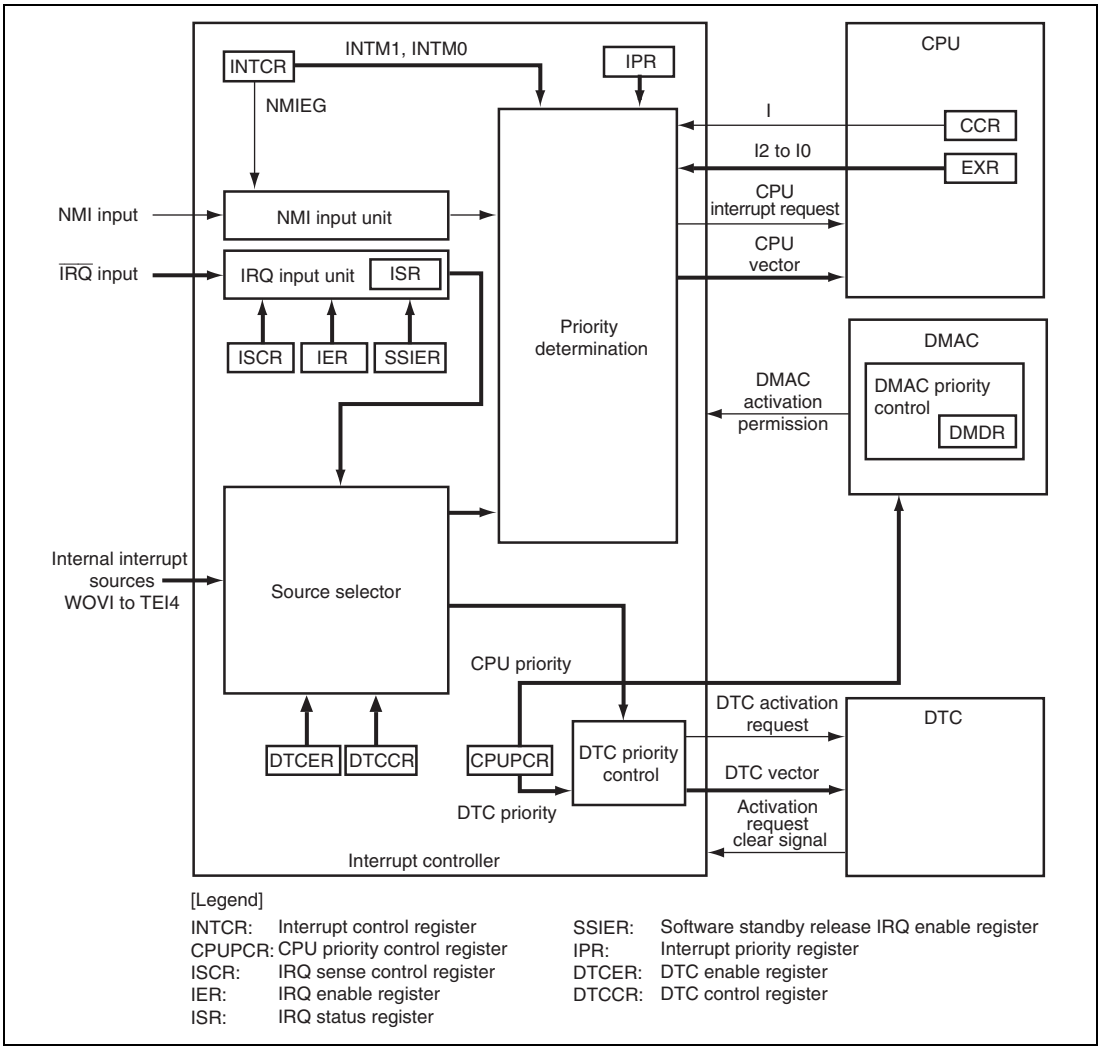


Figure 5.1 Block Diagram of Interrupt Controller

5.2 Input/Output Pins

Table 5.1 shows the pin configuration of the interrupt controller.

Table 5.1 Pin Configuration

Name	I/O	Function
NMI	Input	Nonmaskable External Interrupt Rising or falling edge can be selected.
$\overline{\text{IRQ11}}$ to $\overline{\text{IRQ0}}$	Input	Maskable External Interrupts Rising, falling, or both edges, or level sensing, can be selected.

5.3 Register Descriptions

The interrupt controller has the following registers.

- Interrupt control register (INTCR)
- CPU priority control register (CPUPCR)
- Interrupt priority registers A to C, E to I, K, and L (IPRA to IPRC, IPRE to IPRI, IPRK, and IPRL)
- IRQ enable register (IER)
- IRQ sense control registers H and L (ISCRH, ISCRL)
- IRQ status register (ISR)
- Software standby release IRQ enable register (SSIER)

5.3.1 Interrupt Control Register (INTCR)

INTCR selects the interrupt control mode, and the detected edge for NMI.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	INTM1	INTM0	NMIEG	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These are read-only bits and cannot be modified.
5	INTM1	0	R/W	Interrupt Control Select Mode 1 and 0
4	INTM0	0	R/W	These bits select either of two interrupt control modes for the interrupt controller. 00: Interrupt control mode 0 Interrupts are controlled by I bit in CCR. 01: Setting prohibited. 10: Interrupt control mode 2 Interrupts are controlled by bits I2 to I0 in EXR, and IPR. 11: Setting prohibited.
3	NMIEG	0	R/W	NMI Edge Select Selects the input edge for the NMI pin. 0: Interrupt request generated at falling edge of NMI input 1: Interrupt request generated at rising edge of NMI input
2 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

5.3.2 CPU Priority Control Register (CPUPCR)

CPUPCR sets whether or not the CPU has priority over the DTC and DMAC. The interrupt exception handling by the CPU can be given priority over that of the DTC and DMAC transfer. The priority level of the DTC is set by bits DTCP2 to DTCP0 in CPUPCR. The priority level of the DMAC is set by the DMAC control register for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	CPUPCE	DTCP2	DTCP1	DTCP0	IPSETE	CPUP2	CPUP1	CPUP0
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: * When the IPSETE bit is set to 1, the CPU priority is automatically updated, so these bits cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	CPUPCE	0	R/W	CPU Priority Control Enable Controls the CPU priority control function. Setting this bit to 1 enables the CPU priority control over the DTC and DMAC. 0: CPU always has the lowest priority 1: CPU priority control enabled
6	DTCP2	0	R/W	DTC Priority Level 2 to 0 These bits set the DTC priority level. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
5	DTCP1	0	R/W	
4	DTCP0	0	R/W	

Bit	Bit Name	Initial Value	R/W	Description
3	IPSETE	0	R/W	Interrupt Priority Set Enable Controls the function which automatically assigns the interrupt priority level of the CPU. Setting this bit to 1 automatically sets bits CPUP2 to CPUP0 by the CPU interrupt mask bit (I bit in CCR or bits I2 to I0 in EXR). 0: Bits CPUP2 to CPUP0 are not updated automatically 1: The interrupt mask bit value is reflected in bits CPUP2 to CPUP0
2	CPUP2	0	R/(W)*	CPU Priority Level 2 to 0 These bits set the CPU priority level. When the CPUPCE is set to 1, the CPU priority control function becomes valid and the priority of CPU processing is assigned in accordance with the settings of bits CPUP2 to CPUP0. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
1	CPUP1	0	R/(W)*	
0	CPUP0	0	R/(W)*	

Note: * When the IPSETE bit is set to 1, the CPU priority is automatically updated, so these bits cannot be modified.

5.3.3 Interrupt Priority Registers A to C, E to I, K, and L (IPRA to IPRC, IPRE to IPRI, IPRK, and IPRL)

IPR sets priory (levels 7 to 0) for interrupts other than NMI.

Setting a value in the range from B'000 to B'111 in the 3-bit groups of bits 14 to 12, 10 to 8, 6 to 4, and 2 to 0 assigns a priority level to the corresponding interrupt. For the correspondence between the interrupt sources and the IPR settings, see table 5.2.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	IPR14	IPR13	IPR12	—	IPR10	IPR9	IPR8
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This is a read-only bit and cannot be modified.
14	IPR14	1	R/W	Sets the priority level of the corresponding interrupt source. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
13	IPR13	1	R/W	
12	IPR12	1	R/W	
11	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
10	IPR10	1	R/W	Sets the priority level of the corresponding interrupt source.
9	IPR9	1	R/W	
8	IPR8	1	R/W	000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
7	—	0	R	Reserved This is a read-only bit and cannot be modified.
6	IPR6	1	R/W	Sets the priority level of the corresponding interrupt source.
5	IPR5	1	R/W	
4	IPR4	1	R/W	000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
3	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
2	IPR2	1	R/W	Sets the priority level of the corresponding interrupt source.
1	IPR1	1	R/W	
0	IPR0	1	R/W	000: Priority level 0 (lowest)
				001: Priority level 1
				010: Priority level 2
				011: Priority level 3
				100: Priority level 4
				101: Priority level 5
				110: Priority level 6
				111: Priority level 7 (highest)

5.3.4 IRQ Enable Register (IER)

IER enables or disables interrupt requests IRQ11 to IRQ0.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	IRQ11E	IRQ10E	IRQ9E	IRQ8E
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
Bit Name	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E
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	Bit Name	Initial Value	R/W	Description
15 to 12	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
11	IRQ11E	0	R/W	IRQ11 Enable The IRQ11 interrupt request is enabled when this bit is 1.

Bit	Bit Name	Initial Value	R/W	Description
10	IRQ10E	0	R/W	IRQ10 Enable The IRQ10 interrupt request is enabled when this bit is 1.
9	IRQ9E	0	R/W	IRQ9 Enable The IRQ9 interrupt request is enabled when this bit is 1.
8	IRQ8E	0	R/W	IRQ8 Enable The IRQ8 interrupt request is enabled when this bit is 1.
7	IRQ7E	0	R/W	IRQ7 Enable The IRQ7 interrupt request is enabled when this bit is 1.
6	IRQ6E	0	R/W	IRQ6 Enable The IRQ6 interrupt request is enabled when this bit is 1.
5	IRQ5E	0	R/W	IRQ5 Enable The IRQ5 interrupt request is enabled when this bit is 1.
4	IRQ4E	0	R/W	IRQ4 Enable The IRQ4 interrupt request is enabled when this bit is 1.
3	IRQ3E	0	R/W	IRQ3 Enable The IRQ3 interrupt request is enabled when this bit is 1.
2	IRQ2E	0	R/W	IRQ2 Enable The IRQ2 interrupt request is enabled when this bit is 1.
1	IRQ1E	0	R/W	IRQ1 Enable The IRQ1 interrupt request is enabled when this bit is 1.
0	IRQ0E	0	R/W	IRQ0 Enable The IRQ0 interrupt request is enabled when this bit is 1.

5.3.5 IRQ Sense Control Registers H and L (ISCRH, ISCRL)

ISCRH and ISCRL select the source that generates an interrupt request on pins $\overline{\text{IRQ11}}$ to $\overline{\text{IRQ0}}$.

Upon changing the setting of ISCR, IRQnF (n = 0 to 11) in ISR is often set to 1 accidentally through an internal operation. In this case, an interrupt exception handling is executed if an IRQn interrupt request is enabled. In order to prevent such an accidental interrupt from occurring, the setting of ISCR should be changed while the IRQn interrupt is disabled, and then the IRQnF in ISR should be cleared to 0.

- ISCRH

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	—	—	—	—
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
Bit Name	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF
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

- ISCRL

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF
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
Bit Name	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF
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

- ISCRH

Bit	Bit Name	Initial Value	R/W	Description
15 to 8	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
7	IRQ11SR	0	R/W	IRQ11 Sense Control Rise
6	IRQ11SF	0	R/W	IRQ11 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ11}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ11}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ11}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ11}}$
5	IRQ10SR	0	R/W	IRQ10 Sense Control Rise
4	IRQ10SF	0	R/W	IRQ10 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ10}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ10}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ10}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ10}}$
3	IRQ9SR	0	R/W	IRQ9 Sense Control Rise
2	IRQ9SF	0	R/W	IRQ9 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ9}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ9}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ9}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ9}}$
1	IRQ8SR	0	R/W	IRQ8 Sense Control Rise
0	IRQ8SF	0	R/W	IRQ8 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ8}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ8}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ8}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ8}}$

- ISCRL

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ7SR	0	R/W	IRQ7 Sense Control Rise
14	IRQ7SF	0	R/W	IRQ7 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ7}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ7}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ7}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ7}}$
13	IRQ6SR	0	R/W	IRQ6 Sense Control Rise
12	IRQ6SF	0	R/W	IRQ6 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ6}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ6}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ6}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ6}}$
11	IRQ5SR	0	R/W	IRQ5 Sense Control Rise
10	IRQ5SF	0	R/W	IRQ5 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ5}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ5}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ5}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ5}}$
9	IRQ4SR	0	R/W	IRQ4 Sense Control Rise
8	IRQ4SF	0	R/W	IRQ4 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ4}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ4}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ4}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ4}}$

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ3SR	0	R/W	IRQ3 Sense Control Rise
6	IRQ3SF	0	R/W	IRQ3 Sense Control Fall 00: Interrupt request generated by low level of $\overline{IRQ3}$ 01: Interrupt request generated at falling edge of $\overline{IRQ3}$ 10: Interrupt request generated at rising edge of $\overline{IRQ3}$ 11: Interrupt request generated at both falling and rising edges of $\overline{IRQ3}$
5	IRQ2SR	0	R/W	IRQ2 Sense Control Rise
4	IRQ2SF	0	R/W	IRQ2 Sense Control Fall 00: Interrupt request generated by low level of $\overline{IRQ2}$ 01: Interrupt request generated at falling edge of $\overline{IRQ2}$ 10: Interrupt request generated at rising edge of $\overline{IRQ2}$ 11: Interrupt request generated at both falling and rising edges of $\overline{IRQ2}$
3	IRQ1SR	0	R/W	IRQ1 Sense Control Rise
2	IRQ1SF	0	R/W	IRQ1 Sense Control Fall 00: Interrupt request generated by low level of $\overline{IRQ1}$ 01: Interrupt request generated at falling edge of $\overline{IRQ1}$ 10: Interrupt request generated at rising edge of $\overline{IRQ1}$ 11: Interrupt request generated at both falling and rising edges of $\overline{IRQ1}$
1	IRQ0SR	0	R/W	IRQ0 Sense Control Rise
0	IRQ0SF	0	R/W	IRQ0 Sense Control Fall 00: Interrupt request generated by low level of $\overline{IRQ0}$ 01: Interrupt request generated at falling edge of $\overline{IRQ0}$ 10: Interrupt request generated at rising edge of $\overline{IRQ0}$ 11: Interrupt request generated at both falling and rising edges of $\overline{IRQ0}$

5.3.6 IRQ Status Register (ISR)

ISR is an IRQ11 to IRQ0 interrupt request register.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	IRQ11F	IRQ10F	IRQ9F	IRQ8F
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
Bit Name	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F
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: * Only 0 can be written, to clear the flag. The bit manipulation instructions or memory operation instructions should be used to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
15 to 12	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
11	IRQ11F	0	R/(W)*	[Setting condition] • When the interrupt selected by ISCR occurs [Clearing conditions] • Writing 0 after reading IRQnF = 1 • When interrupt exception handling is executed when low-level sensing is selected and $\overline{\text{IRQn}}$ input is high • When IRQn interrupt exception handling is executed when falling-, rising-, or both-edge sensing is selected • When the DTC is activated by an IRQn interrupt, and the DISEL bit in MRB of the DTC is cleared to 0
10	IRQ10F	0	R/(W)*	
9	IRQ9F	0	R/(W)*	
8	IRQ8F	0	R/(W)*	
7	IRQ7F	0	R/(W)*	
6	IRQ6F	0	R/(W)*	
5	IRQ5F	0	R/(W)*	
4	IRQ4F	0	R/(W)*	
3	IRQ3F	0	R/(W)*	
2	IRQ2F	0	R/(W)*	
1	IRQ1F	0	R/(W)*	
0	IRQ0F	0	R/(W)*	

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

5.3.7 Software Standby Release IRQ Enable Register (SSIER)

SSIER selects pins used to leave software standby mode from pins $\overline{\text{IRQ11}}$ to $\overline{\text{IRQ0}}$.

The IRQ interrupt used to leave software standby mode should not be set as the DTC activation source.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	SSI11	SSI10	SSI9	SSI8
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
Bit Name	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0
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	Bit Name	Initial Value	R/W	Description
15 to 12	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
11	SSI11	0	R/W	Software Standby Release IRQ Setting These bits select the $\overline{\text{IRQn}}$ pins used to leave software standby mode (n = 11 to 0). 0: IRQn requests are not sampled in software standby mode 1: When an IRQn request occurs in software standby mode, this LSI leaves software standby mode after the oscillation settling time has elapsed
10	SSI10	0	R/W	
9	SSI9	0	R/W	
8	SSI8	0	R/W	
7	SSI7	0	R/W	
6	SSI6	0	R/W	
5	SSI5	0	R/W	
4	SSI4	0	R/W	
3	SSI3	0	R/W	
2	SSI2	0	R/W	
1	SSI1	0	R/W	
0	SSI0	0	R/W	

5.4 Interrupt Sources

5.4.1 External Interrupts

There are thirteen external interrupts: NMI and IRQ11 to IRQ0. These interrupts can be used to leave software standby mode.

(1) NMI Interrupts

Nonmaskable interrupt request (NMI) is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode or the settings of the CPU interrupt mask bits. The NMIEG bit in INTCR selects whether an interrupt is requested at the rising or falling edge on the NMI pin.

When an NMI interrupt is generated, the interrupt controller determines that an error has occurred, and performs the following procedure.

- Sets the ERR bit of DTCCR in the DTC to 1.
- Sets the ERRF bit of DMDR_0 in the DMAC to 1
- Clears the DTE bits of DMDRs for all channels in the DMAC to 0 to forcibly terminate transfer

(2) IRQn Interrupts

An IRQn interrupt is requested by a signal input on pins $\overline{\text{IRQ11}}$ to $\overline{\text{IRQ0}}$. $\overline{\text{IRQn}}$ (n = 11 to 0) have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, on pins $\overline{\text{IRQn}}$.
- Enabling or disabling of interrupt requests IRQn can be selected by IER.
- The interrupt priority can be set by IPR.
- The status of interrupt requests IRQn is indicated in ISR. ISR flags can be cleared to 0 by software. The bit manipulation instructions and memory operation instructions should be used to clear the flag.

Detection of IRQn interrupts is enabled through the P1ICR, P2ICR, and P5ICR register settings, and does not change regardless of the output setting. However, when a pin is used as an external interrupt input pin, the pin must not be used as an I/O pin for another function by clearing the corresponding DDR bit to 0.

A block diagram of interrupts IRQn is shown in figure 5.2.

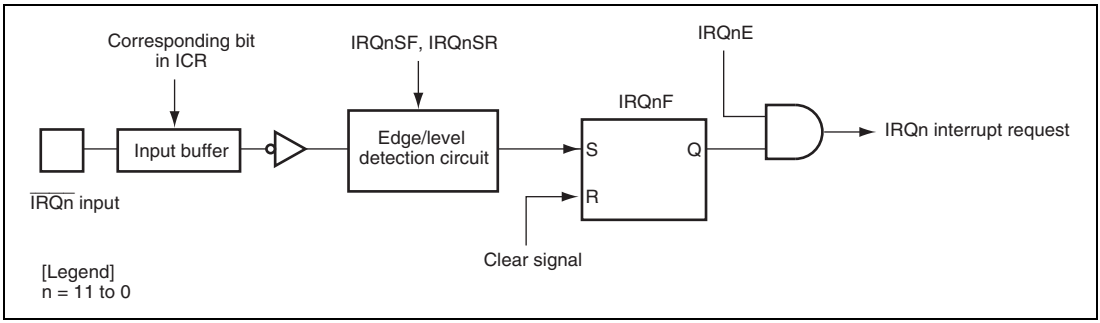


Figure 5.2 Block Diagram of Interrupts IRQn

When the IRQ sensing control in ISCR is set to a low level of signal $\overline{\text{IRQn}}$, the level of $\overline{\text{IRQn}}$ should be held low until an interrupt handling starts. Then set the corresponding input signal $\overline{\text{IRQn}}$ to high in the interrupt handling routine and clear the IRQnF to 0. Interrupts may not be executed when the corresponding input signal $\overline{\text{IRQn}}$ is set to high before the interrupt handling begins.

5.4.2 Internal Interrupts

The sources for internal interrupts from on-chip peripheral modules have the following features:

- For each on-chip peripheral module there are flags that indicate the interrupt request status, and enable bits that enable or disable these interrupts. They can be controlled independently. When the enable bit is set to 1, an interrupt request is issued to the interrupt controller.
- The interrupt priority can be set by means of IPR.
- The DTC and DMAC can be activated by a TPU, SCI, or other interrupt request.
- The priority levels of DTC and DMAC activation can be controlled by the DTC and DMAC priority control functions.

5.5 Interrupt Exception Handling Vector Table

Table 5.2 lists interrupt exception handling sources, vector address offsets, and interrupt priority.

In the default priority order, a lower vector number corresponds to a higher priority. When interrupt control mode 2 is set, priority levels can be changed by setting the IPR contents. The priority for interrupt sources allocated to the same level in IPR follows the default priority, that is, they are fixed.

Table 5.2 Interrupt Sources, Vector Address Offsets, and Interrupt Priority

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
External pin	NMI	7	H'001C	—	High	—	—
	IRQ0	64	H'0100	IPRA14 to IPRA12		O	—
	IRQ1	65	H'0104	IPRA10 to IPRA8		O	—
	IRQ2	66	H'0108	IPRA6 to IPRA4		O	—
	IRQ3	67	H'010C	IPRA2 to IPRA0		O	—
	IRQ4	68	H'0110	IPRB14 to IPRB12		O	—
	IRQ5	69	H'0114	IPRB10 to IPRB8		O	—
	IRQ6	70	H'0118	IPRB6 to IPRB4		O	—
	IRQ7	71	H'011C	IPRB2 to IPRB0		O	—
	IRQ8	72	H'0120	IPRC14 to IPRC12		O	—
	IRQ9	73	H'0124	IPRC10 to IPRC8		O	—
	IRQ10	74	H'0128	IPRC6 to IPRC4		O	—
	IRQ11	75	H'012C	IPRC2 to IPRC0		O	—
—	Reserved for system use	76	H'0130	—		—	—
		77	H'0134			—	—
		78	H'0138			—	—
		79	H'013C			—	—
		80	H'0140			—	—
WDT	WOVI	81	H'0144	IPRE10 to IPRE8	Low	—	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
—	Reserved for system use	82	H'0148	—	High	—	—
		83	H'014C			—	—
		84	H'015C			—	—
		85	H'0154			—	—
A/D	ADI	86	H'0158	IPRF10 to IPRF8		O	O
—	Reserved for system use	87	H'015C	—		—	—
TPU_0	TGI0A	88	H'0160	IPRF6 to IPRF4		O	O
	TGI0B	89	H'0164			O	—
	TGI0C	90	H'0168			O	—
	TGI0D	91	H'016C			O	—
	TCI0V	92	H'0170			—	—
TPU_1	TGI1A	93	H'0174	IPRF2 to IPRF0		O	O
	TGI1B	94	H'0178			O	—
	TCI1V	95	H'017C			—	—
	TCI1U	96	H'0180			—	—
TPU_2	TGI2A	97	H'0184	IPRG14 to IPRG12		O	O
	TGI2B	98	H'0188			O	—
	TCI2V	99	H'018C			—	—
	TCI2U	100	H'0190			—	—
TPU_3	TGI3A	101	H'0194	IPRG10 to IPRG8		O	O
	TGI3B	102	H'0198			O	—
	TGI3C	103	H'019C			O	—
	TGI3D	104	H'01A0			O	—
	TCI3V	105	H'01A4			—	—
TPU_4	TGI4A	106	H'01A8	IPRG6 to IPRG4		O	O
	TGI4B	107	H'01AC			O	—
	TCI4V	108	H'01B0			—	—
	TCI4U	109	H'01B4		Low	—	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
TPU_5	TGI5A	110	H'01B8	IPRG2 to IPRG0	High	O	O
	TGI5B	111	H'01BC			O	—
	TCI5V	112	H'01C0			—	—
	TCI5U	113	H'01C4			—	—
—	Reserved for system use	114	H'01C8	—		—	—
		115	H'01CC			—	—
TMR_0	CMIA0	116	H'01D0	IPRH14 to IPRH12		O	—
	CMIB0	117	H'01D4			O	—
	OVI0	118	H'01D8			—	—
TMR_1	CMIA1	119	H'01DC	IPRH10 to IPRH8		O	—
	CMIB1	120	H'01E0			O	—
	OVI1	121	H'01E4			—	—
TMR_2	CMIA2	122	H'01E8	IPRH6 to IPRH4		O	—
	CMIB2	123	H'01EC			O	—
	OVI2	124	H'01F0			—	—
TMR_3	CMIA3	125	H'01F4	IPRH2 to IPRH0		O	—
	CMIB3	126	H'01F8			O	—
	OVI3	127	H'01FC			—	—
DMAC	DMTEND0	128	H'0200	IPRI14 to IPRI12		O	—
	DMTEND1	129	H'0204	IPRI10 to IPRI8		O	—
	DMTEND2	130	H'0208	IPRI6 to IPRI4		O	—
	DMTEND3	131	H'020C	IPRI2 to IPRI0		O	—
—	Reserved for system use	132	H'0210	—		—	—
		133	H'0214			—	—
		134	H'0218			—	—
		135	H'021C			—	—
DMAC	DMEEND0	136	H'0220	IPRK14 to IPRK12		O	—
	DMEEND1	137	H'0224			O	—
	DMEEND2	138	H'0228			O	—
	DMEEND3	139	H'022C		Low	O	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
—	Reserved for system use	140	H'0230	—	High	—	—
		141	H'0234			—	—
		142	H'0238			—	—
		143	H'023C			—	—
SCI_0	ERI0	144	H'0240	IPRK6 to IPRK4		—	—
	RXI0	145	H'0244			O	O
	TXI0	146	H'0248			O	O
	TEI0	147	H'024C			—	—
SCI_1	ERI1	148	H'0250	IPRK2 to IPRK0		—	—
	RXI1	149	H'0254			O	O
	TXI1	150	H'0258			O	O
	TEI1	151	H'025C			—	—
SCI_2	ERI2	152	H'0260	IPRL14 to IPRL12		—	—
	RXI2	153	H'0264			O	O
	TXI2	154	H'0268			O	O
	TEI2	155	H'026C			—	—
—	Reserved for system use	156	H'0270	—		—	—
		157	H'0274			—	—
		158	H'0278			—	—
		159	H'027C			—	—
SCI_4	ERI4	160	H'0280	IPRL6 to IPRL4		—	—
	RXI4	161	H'0284			O	O
	TXI4	162	H'0288			O	O
	TEI4	163	H'028C			—	—
—	Reserved for system use	164	H'0290	—		—	—
		⎮	⎮			⎮	⎮
		255	H'03FC		Low	—	—

Note: * Lower 16 bits of the start address in advanced, middle, and maximum modes.

5.6 Interrupt Control Modes and Interrupt Operation

The interrupt controller has two interrupt control modes: interrupt control mode 0 and interrupt control mode 2. Interrupt operations differ depending on the interrupt control mode. The interrupt control mode is selected by INTCR. Table 5.3 shows the differences between interrupt control mode 0 and interrupt control mode 2.

Table 5.3 Interrupt Control Modes

Interrupt Control Mode	Priority Setting Register	Interrupt Mask Bit	Description
0	Default	I	The priority levels of the interrupt sources are fixed default settings. The interrupts except for NMI is masked by the I bit.
2	IPR	I2 to I0	Eight priority levels can be set for interrupt sources except for NMI with IPR. 8-level interrupt mask control is performed by bits I2 to I0.

5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests except for NMI are masked by the I bit in CCR of the CPU. Figure 5.3 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, the interrupt request is sent to the interrupt controller.
2. If the I bit in CCR is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending. If the I bit is cleared to 0, an interrupt request is accepted.
3. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority, sends the request to the CPU, and holds other interrupt requests pending.
4. When the CPU accepts the interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR contents are saved to the stack area during the interrupt exception handling. The PC contents saved on the stack is the address of the first instruction to be executed after returning from the interrupt handling routine.
6. Next, the I bit in CCR is set to 1. This masks all interrupts except NMI.

7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

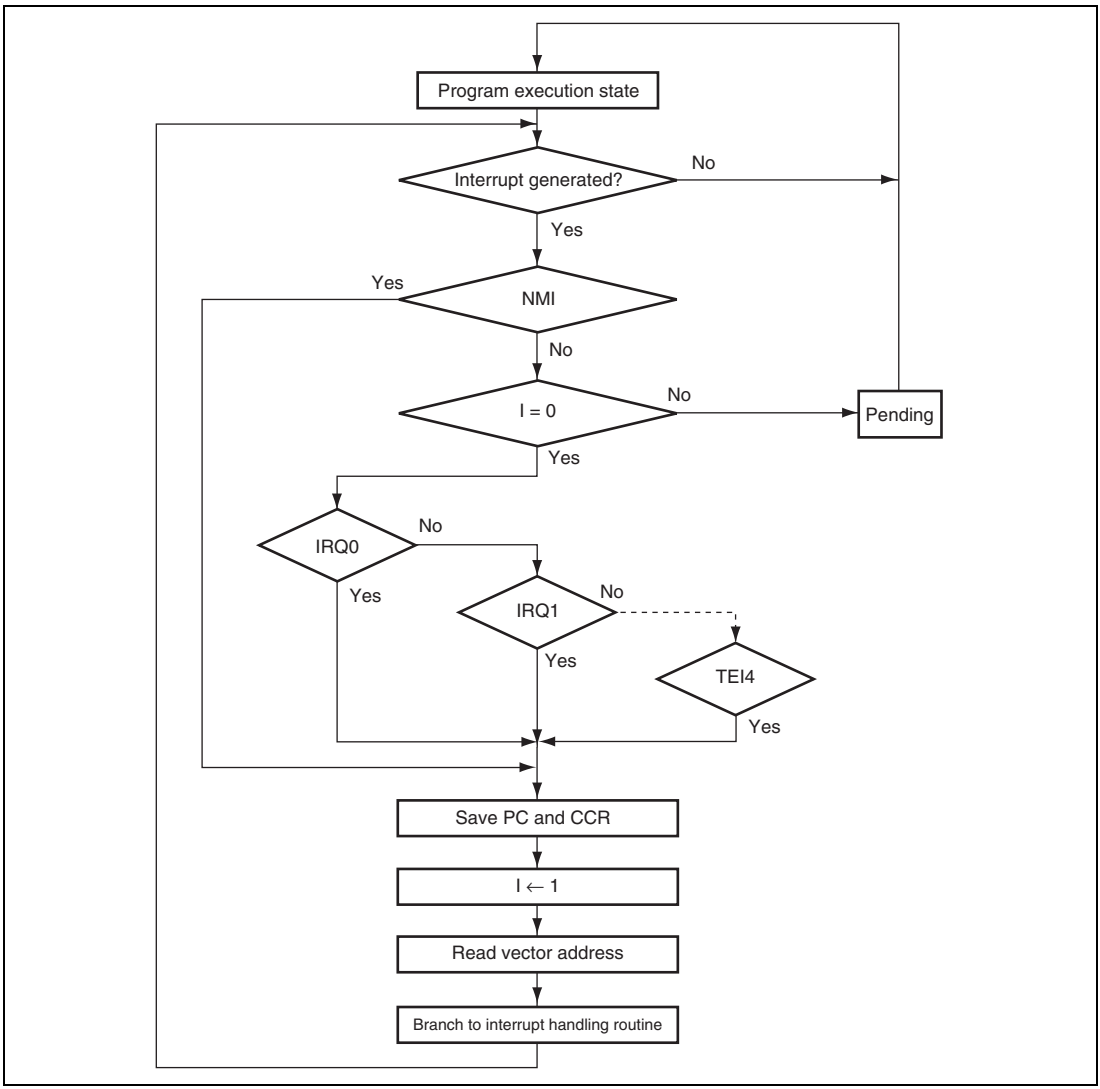


Figure 5.3 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0

5.6.2 Interrupt Control Mode 2

In interrupt control mode 2, interrupt requests except for NMI are masked by comparing the interrupt mask level (I2 to I0 bits) in EXR of the CPU and the IPR setting. There are eight levels in mask control. Figure 5.4 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority according to the IPR setting, and holds other interrupt requests pending. If multiple interrupt requests has the same priority, an interrupt request is selected according to the default setting shown in table 5.2.
3. Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. When the interrupt request does not have priority over the mask level set, it is held pending, and only an interrupt request with a priority over the interrupt mask level is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC, CCR, and EXR contents are saved to the stack area during interrupt exception handling. The PC saved on the stack is the address of the first instruction to be executed after returning from the interrupt handling routine.
6. The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority of the accepted interrupt. If the accepted interrupt is NMI, the interrupt mask level is set to H'7.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

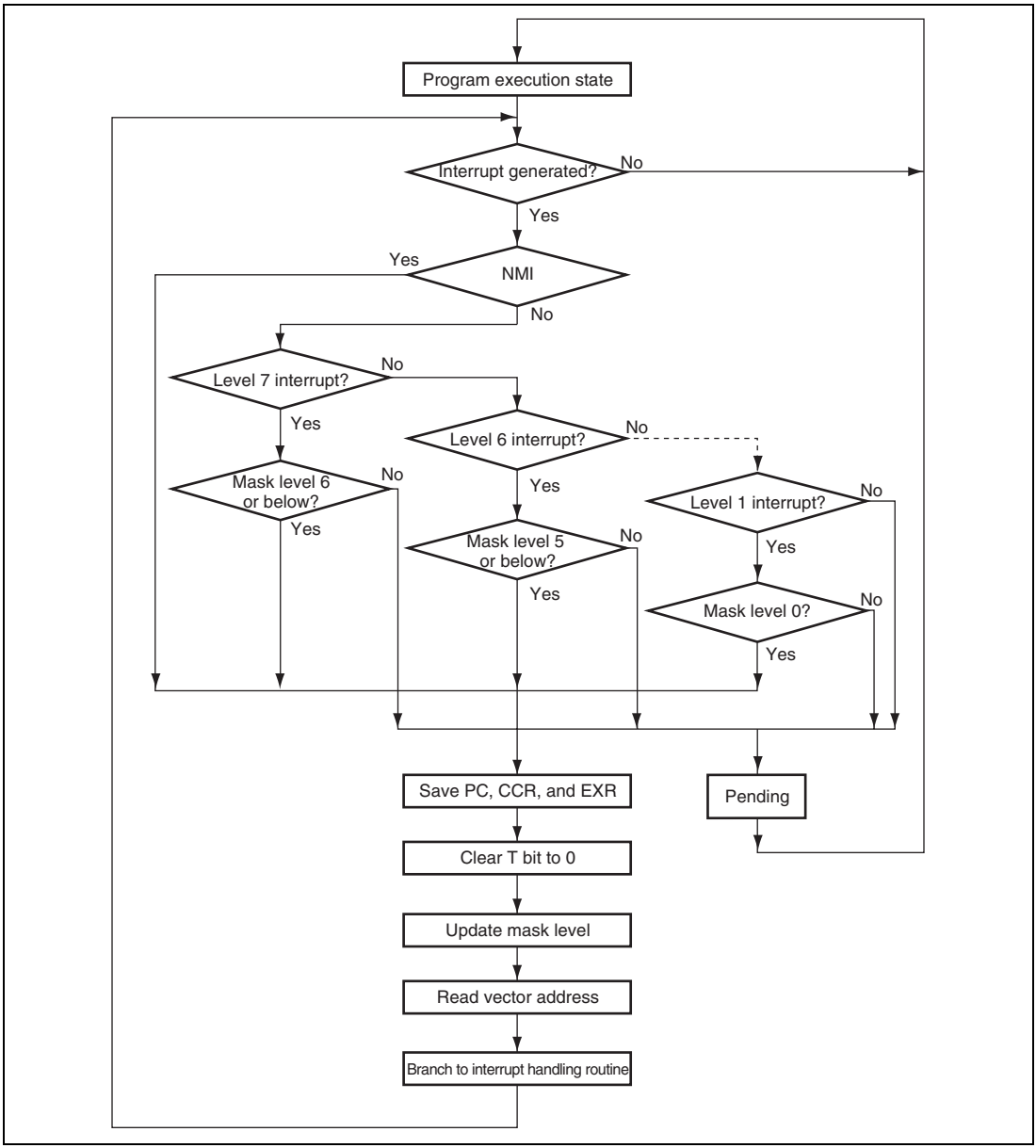


Figure 5.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2

5.6.3 Interrupt Exception Handling Sequence

Figure 5.5 shows the interrupt exception handling sequence. The example is for the case where interrupt control mode 0 is set in maximum mode, and the program area and stack area are in on-chip memory.

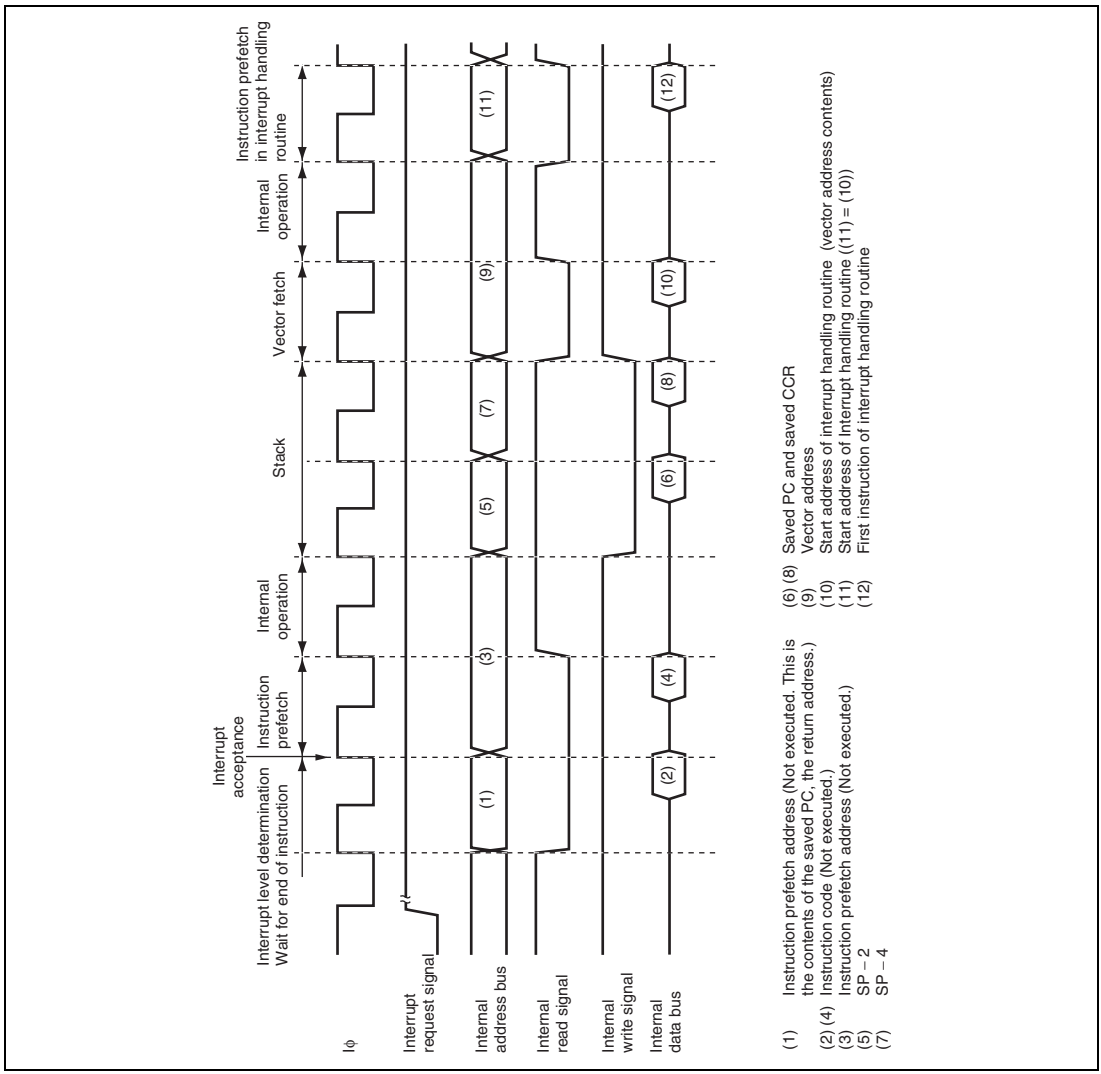


Figure 5.5 Interrupt Exception Handling

5.6.4 Interrupt Response Times

Table 5.4 shows interrupt response times – the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The symbols for execution states used in table 5.4 are explained in table 5.5.

This LSI is capable of fast word transfer to on-chip memory, so allocating the program area in on-chip ROM and the stack area in on-chip RAM enables high-speed processing.

Table 5.4 Interrupt Response Times

	Normal Mode*[5]		Advanced Mode		Maximum Mode*[5]	
Execution State	**Interrupt Control Mode 0**	**Interrupt Control Mode 2**	**Interrupt Control Mode 0**	**Interrupt Control Mode 2**	**Interrupt Control Mode 0**	**Interrupt Control Mode 2**
Interrupt priority determination*[1]	3					
Number of states until executing instruction ends*[2]	1 to 19 + $2 \cdot S_I$					
PC, CCR, EXR stacking	S_K to $2 \cdot S_K$*[6]	$2 \cdot S_K$	S_K to $2 \cdot S_K$*[6]	$2 \cdot S_K$	$2 \cdot S_K$	$2 \cdot S_K$
Vector fetch	S_h					
Instruction fetch*[3]	$2 \cdot S_I$					
Internal processing*[4]	2					
Total (using on-chip memory)	10 to 31	11 to 31	10 to 31	11 to 31	11 to 31	11 to 31

Notes: 1. Two states for an internal interrupt.
2. In the case of the MULXS or DIVXS instruction
3. Prefetch after interrupt acceptance or for an instruction in the interrupt handling routine.
4. Internal operation after interrupt acceptance or after vector fetch
5. Not available in this LSI.
6. When setting the SP value to 4n, the interrupt response time is S_K; when setting to 4n + 2, the interrupt response time is $2 \cdot S_K$.

Table 5.5 Number of Execution States in Interrupt Handling Routine

	Object of Access				
		External Device			
		8-Bit Bus		16-Bit Bus	
Symbol	On-Chip Memory	2-State Access	3-State Access	2-State Access	3-State Access
Vector fetch S_h	1	8	12 + 4m	4	6 + 2m
Instruction fetch S_I	1	4	6 + 2m	2	3 + m
Stack manipulation S_K	1	8	12 + 4m	4	6 + 2m

[Legend]
m: Number of wait cycles in an external device access.

5.6.5 DTC and DMAC Activation by Interrupt

The DTC and DMAC can be activated by an interrupt. In this case, the following options are available:

- Interrupt request to the CPU
- Activation request to the DTC
- Activation request to the DMAC
- Combination of the above

For details on interrupt requests that can be used to activate the DTC and DMAC, see table 5.2, section 7, DMA Controller (DMAC), and section 8, Data Transfer Controller (DTC).

Figure 5.6 shows a block diagram of the DTC, DMAC, and interrupt controller.

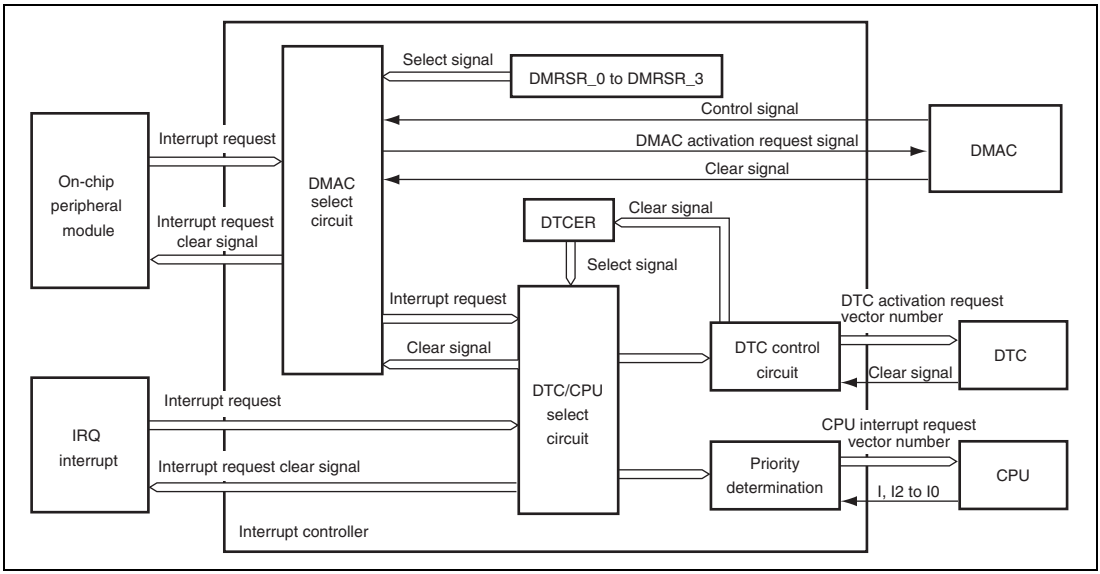


Figure 5.6 Block Diagram of DTC, DMAC, and Interrupt Controller

(1) Selection of Interrupt Sources

The activation source for each DMAC channel is selected by DMRSR. The selected activation source is input to the DMAC through the select circuit. When transfer by an on-chip module interrupt is enabled (DTF1 = 1, DTF0 = 0, and DTE = 1 in DMDR) and the DTA bit in DMDR is set to 1, the interrupt source selected for the DMAC activation source is controlled by the DMAC and cannot be used as a DTC activation source or CPU interrupt source.

Interrupt sources that are not controlled by the DMAC are set for DTC activation sources or CPU interrupt sources by the DTCE bit in DTCERA to DTCERH of the DTC.

Specifying the DISEL bit in MRB of the DTC generates an interrupt request to the CPU by clearing the DTCE bit to 0 after the individual DTC data transfer.

Note that when the DTC performs a predetermined number of data transfers and the transfer counter indicates 0, an interrupt request is made to the CPU by clearing the DTCE bit to 0 after the DTC data transfer.

When the same interrupt source is set as both the DTC and DMAC activation source and CPU interrupt source, the DTC and DMAC must be given priority over the CPU. If the IPSETE bit in CPUPCR is set to 1, the priority is determined according to the IPR setting corresponding to the interrupt source. Therefore, the CPUP setting or the IPR setting corresponding to the interrupt source must be set to lower than or equal to the DTCP and DMAP setting. If the CPU is given priority over the DTC and DMAC, the DTC and DMAC may not be activated, and the data transfer may not be performed.

(2) Priority Determination

The DTC activation source is selected according to the default priority, and the selection is not affected by its mask level or priority level. For respective priority levels, see table 8.1, in section 8.4, Location of Transfer Information and DTC Vector Table.

(3) Operation Order

If the same interrupt is selected as both the DTC activation source and CPU interrupt source, the CPU interrupt exception handling is performed after the DTC data transfer. If the same interrupt is selected as the DTC or DMAC activation source or CPU interrupt source, respective operations are performed independently.

Table 5.6 lists the selection of interrupt sources and interrupt source clear control by setting the DTA bit in DMDR of the DMAC, the DTCE bit in DTCERA to DTCERH of the DTC, and the DISEL bit in MRB of the DTC.

Table 5.6 Interrupt Source Selection and Clear Control

DMAC Setting	DTC Setting		Interrupt Source Selection/Clear Control		
DTA	DTCE	CISEL	DMAC	DTC	CPU
0	0	*	O	X	√
	1	0	O	√	X
		1	O	O	√
1	*	*	√	X	X

[Legend]

√: The corresponding interrupt is used. The interrupt source is cleared.
(The interrupt source flag must be cleared in the CPU interrupt handling routine.)

O: The corresponding interrupt is used. The interrupt source is not cleared.

X: The corresponding interrupt is not available.

*: Don't care.

(4) Usage Note

The interrupt sources of the SCI and A/D converter are cleared according to the setting shown in table 5.6, when the DTC or DMAC reads/writes the prescribed register.

To initiate multiple channels for the DTC and DMAC with the same interrupt, the same priority (DTCP = DMAP) should be assigned.

5.7 CPU Priority Control Function Over DTC and DMAC

The interrupt controller has a function to control the priority among the DTC, DMAC, and the CPU by assigning different priority levels to the DTC, DMAC, and CPU. Since the priority level can automatically be assigned to the CPU on an interrupt occurrence, it is possible to execute the CPU interrupt exception handling prior to the DTC or DMAC transfer.

The priority level of the CPU is assigned by bits CPUP2 to CPUP0 in CPUPCR. The priority level of the DTC is assigned by bits DTCP2 to DTCP0 in CPUPCR. The priority level of the DMAC is assigned by bits DMAP2 to DMAP0 in DMDR for each channel.

The priority control function over the DTC and DMAC is enabled by setting the CPUPCE bit in CPUPCR to 1. When the CPUPCE bit is 1, the DTC and DMAC activation sources are controlled according to the respective priority levels.

The DTC activation source is controlled according to the priority level of the CPU indicated by bits CPUP2 to CPUP0 and the priority level of the DTC indicated by bits DTCP2 to DTCP0. If the CPU has priority, the DTC activation source is held. The DTC is activated when the condition by which the activation source is held is cancelled (CPUCPCE = 1 and value of bits CPUP2 to CPUP0 is greater than that of bits DTCP2 to DTCP0). The priority level of the DTC is assigned by the DTCP2 to DTCP0 bits regardless of the activation source.

For the DMAC, the priority level can be specified for each channel. The DMAC activation source is controlled according to the priority level of each DMAC channel indicated by bits DMAP2 to DMAP0 and the priority level of the CPU. If the CPU has priority, the DMAC activation source is held. The DMAC is activated when the condition by which the activation source is held is cancelled (CPUCPCE = 1 and value of bits CPUP2 to CPUP0 is greater than that of bits DMAP2 to DMAP0). If different priority levels are specified for channels, the channels of the higher priority levels continue transfer and the activation sources for the channels of lower priority levels than that of the CPU are held.

There are two methods for assigning the priority level to the CPU by the IPSETE bit in CPUPCR. Setting the IPSETE bit to 1 enables a function to automatically assign the value of the interrupt mask bit of the CPU to the CPU priority level. Clearing the IPSETE bit to 0 disables the function to automatically assign the priority level. Therefore, the priority level is assigned directly by software rewriting bits CPUP2 to CPUP0. Even if the IPSETE bit is 1, the priority level of the CPU is software assignable by rewriting the interrupt mask bit of the CPU (I bit in CCR or I2 to I0 bits in EXR).

The priority level which is automatically assigned when the IPSETE bit is 1 differs according to the interrupt control mode.

In interrupt control mode 0, the I bit in CCR of the CPU is reflected in bit CPUP2. Bits CPUP1 and CPUP0 are fixed 0. In interrupt control mode 2, the values of bits I2 to I0 in EXR of the CPU are reflected in bits CPUP2 to CPUP0.

Table 5.7 shows the CPU priority control.

Table 5.7 CPU Priority Control

Interrupt Control Mode	Interrupt Priority	Interrupt Mask Bit	IPSETE in CPUPCR	Control Status	
				CPUP2 to CPUP0	Updating of CPUP2 to CPUP0
0	Default	I = any	0	B'111 to B'000	Enabled
		I = 0	1	B'000	Disabled
		I = 1		B'100	
2	IPR setting	I2 to I0	0	B'111 to B'000	Enabled
			1	I2 to I0	Disabled

Table 5.8 shows an setting example of the priority control function over the DTC and DMAC and the transfer request control state. A priority level can be independently set to each DMAC channel, but the table only shows one channel for example. Transfers through the DMAC channels can be separately controlled by assigning different priority levels for channels.

Table 5.8 Example of Priority Control Function Setting and Control State

Interrupt Control Mode	CPUPCE in CPUPCR	CPUP2 to CPUP0	DTCP2 to DTCP0	DMAP2 to DMAP0	Transfer Request Control State	
					DTC	DMAC
0	0	Any	Any	Any	Enabled	Enabled
	1	B'000	B'000	B'000	Enabled	Enabled
		B'100	B'000	B'000	Masked	Masked
		B'100	B'000	B'011	Masked	Masked
		B'100	B'111	B'101	Enabled	Enabled
		B'000	B'111	B'101	Enabled	Enabled
2	0	Any	Any	Any	Enabled	Enabled
	1	B'000	B'000	B'000	Enabled	Enabled
		B'000	B'011	B'101	Enabled	Enabled
		B'011	B'011	B'101	Enabled	Enabled
		B'100	B'011	B'101	Masked	Enabled
		B'101	B'011	B'101	Masked	Enabled
		B'110	B'011	B'101	Masked	Masked
		B'111	B'011	B'101	Masked	Masked
		B'101	B'011	B'101	Masked	Enabled
		B'101	B'110	B'101	Enabled	Enabled

5.8 Usage Notes

5.8.1 Conflict between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to mask the interrupt, the masking becomes effective after execution of the instruction.

When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request with priority over that interrupt, interrupt exception handling will be executed for the interrupt with priority, and another interrupt will be ignored. The same also applies when an interrupt source flag is cleared to 0. Figure 5.7 shows an example in which the TCIEV bit in TIER of the TPU is cleared to 0. The above conflict will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

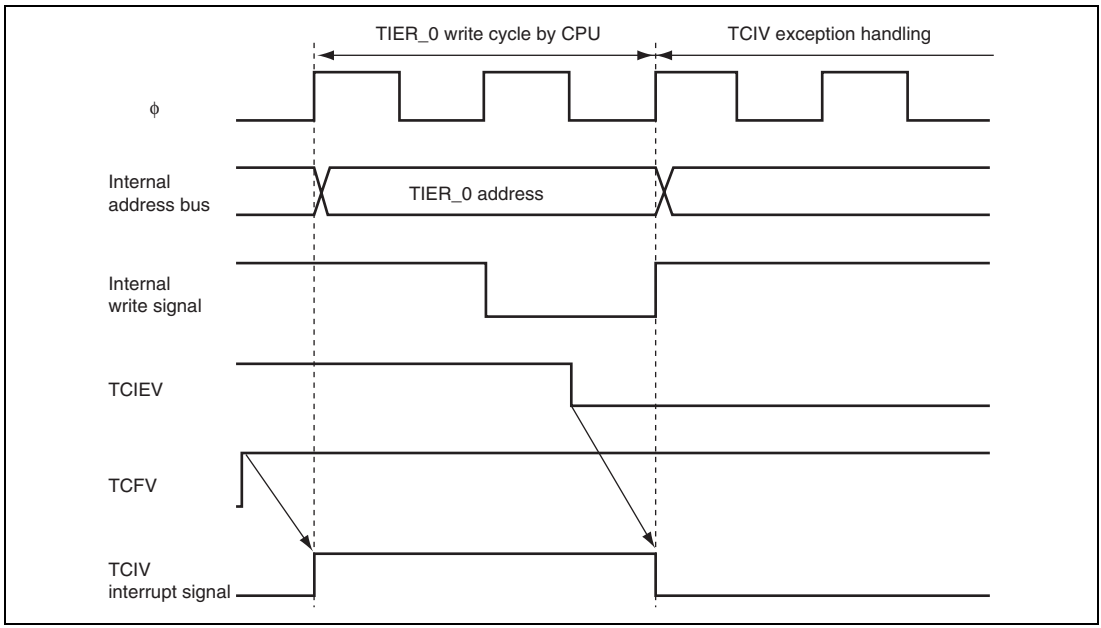


Figure 5.7 Conflict between Interrupt Generation and Disabling

Similarly, when an interrupt is requested immediately before the DTC enable bit is changed to activate the DTC, DTC activation and the interrupt exception handling by the CPU are both executed. When changing the DTC enable bit, make sure that an interrupt is not requested.

5.8.2 Instructions that Disable Interrupts

Instructions that disable interrupts immediately after execution are LDC, ANDC, ORC, and XORC. After any of these instructions is executed, all interrupts including NMI are disabled and the next instruction is always executed. When the I bit is set by one of these instructions, the new value becomes valid two states after execution of the instruction ends.

5.8.3 Times when Interrupts are Disabled

There are times when interrupt acceptance is disabled by the interrupt controller.

The interrupt controller disables interrupt acceptance for a 3-state period after the CPU has updated the mask level with an LDC, ANDC, ORC, or XORC instruction, and for a period of writing to the registers of the interrupt controller.

5.8.4 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B and the EEPMOV.W instructions.

With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the transfer is completed.

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual transfer cycle. The PC value saved on the stack in this case is the address of the next instruction. Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

```
L1:    EEPMOV.W
       MOV.W  R4,R4
       BNE    L1
```

5.8.5 Interrupts during Execution of MOVMD and MOVSD Instructions

With the MOVMD or MOVSD instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual transfer cycle. The PC value saved on the stack in this case is the address of the MOVMD or MOVSD instruction. The transfer of the remaining data is resumed after returning from the interrupt handling routine.

5.8.6 Interrupt Source Flag of Peripheral Module

To clear an interrupt source flag of a peripheral module by the CPU, the flag must be read from after clearing the flag within the interrupt handling routine. This makes the CPU operation synchronized with the peripheral module clock.

Section 6 Bus Controller (BSC)

This LSI has an on-chip bus controller (BSC) that manages the external address space divided into eight areas.

The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters; CPU, DMAC, and DTC.

6.1 Features

- Manages external address space in area units
  Manages the external address space divided into eight areas
  Chip select signals ($\overline{CS0}$ to $\overline{CS7}$) can be output for each area
  Bus specifications can be set independently for each area
  8-bit access or 16-bit access can be selected for each area
  Burst ROM, byte control SRAM, or address/data multiplexed I/O interface can be set
  An endian conversion function is provided to connect a device of little endian
- Basic bus interface
  This interface can be connected to the SRAM and ROM
  2-state access or 3-state access can be selected for each area
  Program wait cycles can be inserted for each area
  Wait cycles can be inserted by the $\overline{WAIT}$ pin.
  Extension cycles can be inserted while $\overline{CSn}$ is asserted for each area (n = 0 to 7)
  The negation timing of the read strobe signal ($\overline{RD}$) can be modified
- Byte control SRAM interface
  Byte control SRAM interface can be set for areas 0 to 7
  The SRAM that has a byte control pin can be directly connected
- Burst ROM interface
  Burst ROM interface can be set for areas 0 and 1
  Burst ROM interface parameters can be set independently for areas 0 and 1
- Address/data multiplexed I/O interface
  Address/data multiplexed I/O interface can be set for areas 3 to 7

- Idle cycle insertion

  Idle cycles can be inserted between external read accesses to different areas

  Idle cycles can be inserted before the external write access after an external read access

  Idle cycles can be inserted before the external read access after an external write access

  Idle cycles can be inserted before the external access after a DMAC single address transfer (write access)
- Write buffer function

  External write cycles and internal accesses can be executed in parallel

  Write accesses to the on-chip peripheral module and on-chip memory accesses can be executed in parallel

  DMAC single address transfers and internal accesses can be executed in parallel
- External bus release function
- Bus arbitration function

  Includes a bus arbiter that arbitrates bus mastership among the CPU, DMAC, and DTC
- Multi-clock function

  The internal peripheral functions can be operated in synchronization with the peripheral module clock (Pϕ). Accesses to the external address space can be operated in synchronization with the external bus clock (Bϕ).
- The bus start ($\overline{BS}$) and read/write (RD/$\overline{WR}$) signals can be output.

A block diagram of the bus controller is shown in figure 6.1.

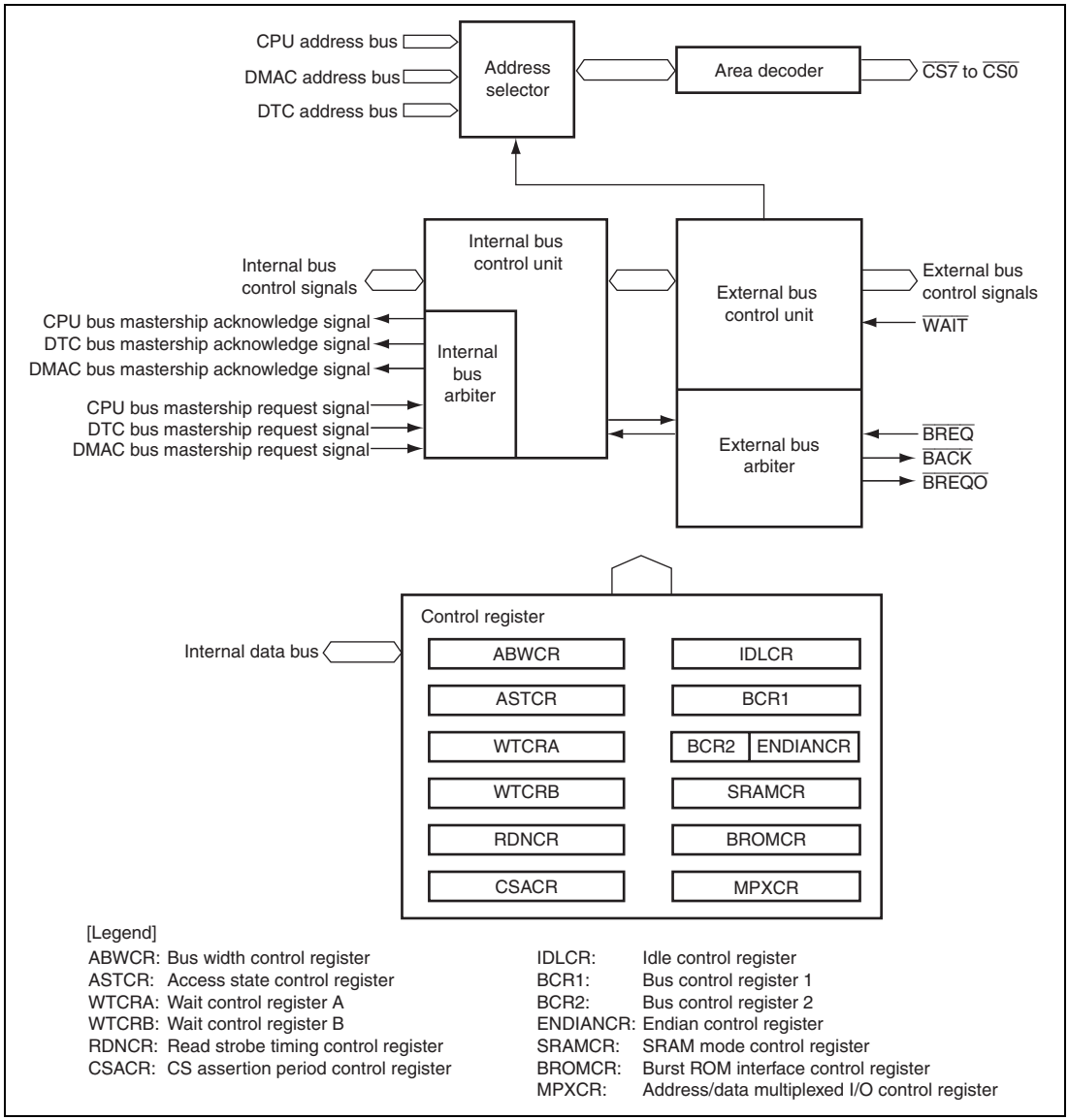


Figure 6.1 Block Diagram of Bus Controller

6.2 Register Descriptions

The bus controller has the following registers.

- Bus width control register (ABWCR)
- Access state control register (ASTCR)
- Wait control register A (WTCRA)
- Wait control register B (WTCRB)
- Read strobe timing control register (RDNCR)
- $\overline{\text{CS}}$ assertion period control register (CSACR)
- Idle control register (IDLCR)
- Bus control register 1 (BCR1)
- Bus control register 2 (BCR2)
- Endian control register (ENDIANCR)
- SRAM mode control register (SRAMCR)
- Burst ROM interface control register (BROMCR)
- Address/data multiplexed I/O control register (MPXCR)

6.2.1 Bus Width Control Register (ABWCR)

ABWCR specifies the data bus width for each area in the external address space.

Bit	15	14	13	12	11	10	9	8
Bit Name	ABWH7	ABWH6	ABWH5	ABWH4	ABWH3	ABWH2	ABWH1	ABWH0
Initial Value	1	1	1	1	1	1	1	1/0
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
Bit Name	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1	ABWL0
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

Note: * Initial value at 16-bit bus initiation is H'FEFF, and that at 8-bit bus initiation is H'FFFF.

Bit	Bit Name	Initial Value*[1]	R/W	Description
15	ABWH7	1	R/W	Area 7 to 0 Bus Width Control
14	ABWH6	1	R/W	These bits select whether the corresponding area is to be designated as 8-bit access space or 16-bit access space.
13	ABWH5	1	R/W	
12	ABWH4	1	R/W	ABWHn ABWLn (n = 7 to 0)
11	ABWH3	1	R/W	× 0: Setting prohibited
10	ABWH2	1	R/W	0 1: Area n is designated as 16-bit access space
9	ABWH1	1	R/W	
8	ABWL0	1/0	R/W	1 1: Area n is designated as 8-bit access space*[2]
7	ABWL7	1	R/W	
6	ABWL6	1	R/W	
5	ABWL5	1	R/W	
4	ABWL4	1	R/W	
3	ABWL3	1	R/W	
2	ABWL2	1	R/W	
1	ABWL1	1	R/W	
0	ABWL0	1	R/W	

[Legend]

×: Don't care

Notes: 1. Initial value at 16-bit bus initiation is H'FEFF, and that at 8-bit bus initiation is H'FFFF.

2. An address space specified as byte control SRAM interface must not be specified as 8-bit access space.

6.2.2 Access State Control Register (ASTCR)

ASTCR designates each area in the external address space as either 2-state access space or 3-state access space and enables/disables wait cycle insertion.

Bit	15	14	13	12	11	10	9	8
Bit Name	AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0
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
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	AST7	1	R/W	Area 7 to 0 Access State Control
14	AST6	1	R/W	These bits select whether the corresponding area is to be designated as 2-state access space or 3-state access space. Wait cycle insertion is enabled or disabled at the same time.
13	AST5	1	R/W	
12	AST4	1	R/W	
11	AST3	1	R/W	0: Area n is designated as 2-state access space
10	AST2	1	R/W	Wait cycle insertion in area n access is disabled
9	AST1	1	R/W	1: Area n is designated as 3-state access space
8	AST0	1	R/W	Wait cycle insertion in area n access is enabled
				(n = 7 to 0)
7 to 0	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

6.2.3 Wait Control Registers A and B (WTCRA, WTCRB)

WTCRA and WTCRB select the number of program wait cycles for each area in the external address space.

• WTCRA

Bit	15	14	13	12	11	10	9	8
Bit Name	—	W72	W71	W70	—	W62	W61	W60
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	W52	W51	W50	—	W42	W41	W40
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

• WTCRB

Bit	15	14	13	12	11	10	9	8
Bit Name	—	W32	W31	W30	—	W22	W21	W20
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	W12	W11	W10	—	W02	W01	W00
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

- WTCRA

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This is a read-only bit and cannot be modified.
14	W72	1	R/W	Area 7 Wait Control 2 to 0 These bits select the number of program wait cycles when accessing area 7 while bit AST7 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
13	W71	1	R/W	
12	W70	1	R/W	
11	—	0	R	Reserved This is a read-only bit and cannot be modified.
10	W62	1	R/W	Area 6 Wait Control 2 to 0 These bits select the number of program wait cycles when accessing area 6 while bit AST6 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
9	W61	1	R/W	
8	W60	1	R/W	
7	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
6	W52	1	R/W	Area 5 Wait Control 2 to 0
5	W51	1	R/W	These bits select the number of program wait cycles when accessing area 5 while bit AST5 in ASTCR is 1.
4	W50	1	R/W	000: Program cycle wait not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
3	—	0	R	Reserved This is a read-only bit and cannot be modified.
2	W42	1	R/W	Area 4 Wait Control 2 to 0
1	W41	1	R/W	These bits select the number of program wait cycles when accessing area 4 while bit AST4 in ASTCR is 1.
0	W40	1	R/W	000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted

- WTCRB

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This is a read-only bit and cannot be modified.
14	W32	1	R/W	Area 3 Wait Control 2 to 0
13	W31	1	R/W	These bits select the number of program wait cycles when accessing area 3 while bit AST3 in ASTCR is 1.
12	W30	1	R/W	000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
11	—	0	R	Reserved This is a read-only bit and cannot be modified.
10	W22	1	R/W	Area 2 Wait Control 2 to 0
9	W21	1	R/W	These bits select the number of program wait cycles when accessing area 2 while bit AST2 in ASTCR is 1.
8	W20	1	R/W	000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
7	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
6	W12	1	R/W	Area 1 Wait Control 2 to 0
5	W11	1	R/W	These bits select the number of program wait cycles when accessing area 1 while bit AST1 in ASTCR is 1.
4	W10	1	R/W	000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
3	—	0	R	Reserved This is a read-only bit and cannot be modified.
2	W02	1	R/W	Area 0 Wait Control 2 to 0
1	W01	1	R/W	These bits select the number of program wait cycles when accessing area 0 while bit AST0 in ASTCR is 1.
0	W00	1	R/W	000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted

6.2.4 Read Strobe Timing Control Register (RDNCR)

RDNCR selects the negation timing of the read strobe signal ($\overline{RD}$) when reading the external address spaces specified as a basic bus interface or the address/data multiplexed I/O interface.

Bit	15	14	13	12	11	10	9	8
Bit Name	RDN7	RDN6	RDN5	RDN4	RDN3	RDN2	RDN1	RDN0
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
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	RDN7	0	R/W	Read Strobe Timing Control
14	RDN6	0	R/W	These bits set the negation timing of the read strobe in a corresponding area read access.
13	RDN5	0	R/W	
12	RDN4	0	R/W	As shown in figure 6.2, the read strobe for an area for which the RDNn bit is set to 1 is negated one half-cycle earlier than that for an area for which the RDNn bit is cleared to 0. The read data setup and hold time are also given one half-cycle earlier.
11	RDN3	0	R/W	
10	RDN2	0	R/W	
9	RDN1	0	R/W	
8	RDN0	0	R/W	0: In an area n read access, the $\overline{RD}$ signal is negated at the end of the read cycle 1: In an area n read access, the $\overline{RD}$ signal is negated one half-cycle before the end of the read cycle (n = 7 to 0)
7 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Notes: 1. In an external address space which is specified as byte control SRAM interface, the RDNCR setting is ignored and the same operation when RDNn = 1 is performed.
2. In an external address space which is specified as burst ROM interface, the RDNCR setting is ignored during CPU read accesses and the same operation when RDNn = 0 is performed.

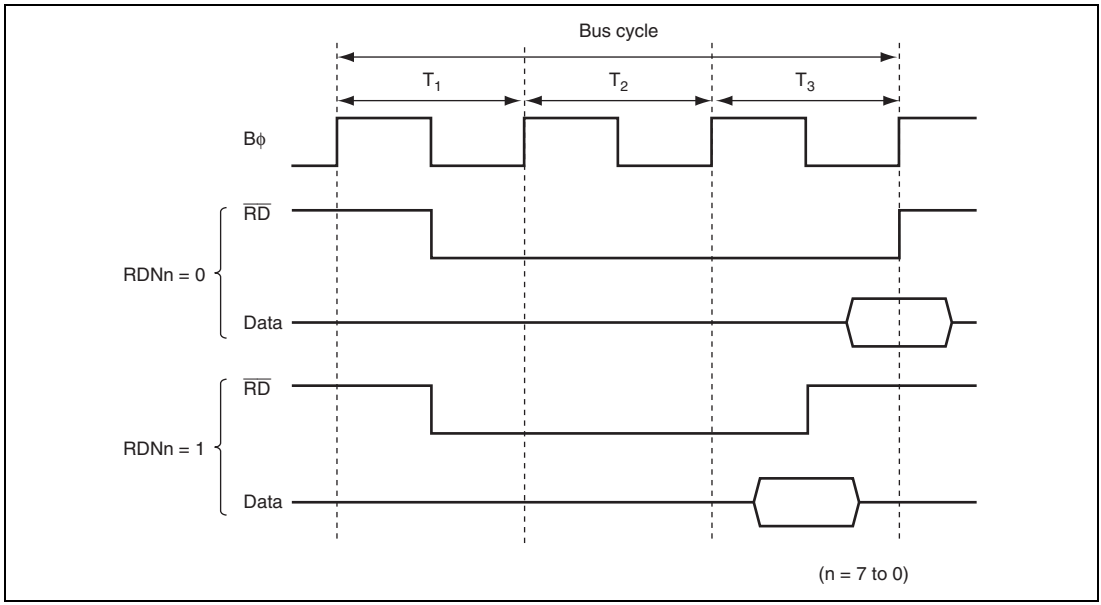


Figure 6.2 Read Strobe Negation Timing (Example of 3-State Access Space)

6.2.5 $\overline{CS}$ Assertion Period Control Registers (CSACR)

CSACR selects whether or not the assertion periods of the chip select signals ($\overline{CSn}$) and address signals for the basic bus, byte-control SRAM, burst ROM, and address/data multiplexed I/O interface are to be extended. Extending the assertion period of the $\overline{CSn}$ and address signals allows the setup time and hold time of read strobe ($\overline{RD}$) and write strobe ($\overline{LHWR}$/$\overline{LLWR}$) to be assured and to make the write data setup time and hold time for the write strobe become flexible.

Bit	15	14	13	12	11	10	9	8
Bit Name	CSXH7	CSXH6	CSXH5	CSXH4	CSXH3	CSXH2	CSXH1	CSXH0
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
Bit Name	CSXT7	CSXT6	CSXT5	CSXT4	CSXT3	CSXT2	CSXT1	CSXT0
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	Bit Name	Initial Value	R/W	Description
15	CSXH7	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Control 1
14	CSXH6	0	R/W	These bits specify whether or not the Th cycle is to be inserted (see figure 6.3). When an area for which bit CSXHn is set to 1 is accessed, one Th cycle, in which the $\overline{CSn}$ and address signals are asserted, is inserted before the normal access cycle.
13	CSXH5	0	R/W	
12	CSXH4	0	R/W	
11	CSXH3	0	R/W	
10	CSXH2	0	R/W	0: In access to area n, the $\overline{CSn}$ and address assertion period (Th) is not extended
9	CSXH1	0	R/W	
8	CSXH0	0	R/W	1: In access to area n, the $\overline{CSn}$ and address assertion period (Th) is extended (n = 7 to 0)
7	CSXT7	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Control 2
6	CSXT6	0	R/W	These bits specify whether or not the Tt cycle is to be inserted (see figure 6.3). When an area for which bit CSXTn is set to 1 is accessed, one Tt cycle, in which the $\overline{CSn}$ and address signals are retained, is inserted after the normal access cycle.
5	CSXT5	0	R/W	
4	CSXT4	0	R/W	
3	CSXT3	0	R/W	
2	CSXT2	0	R/W	0: In access to area n, the $\overline{CSn}$ and address assertion period (Tt) is not extended
1	CSXT1	0	R/W	
0	CSXT0	0	R/W	1: In access to area n, the $\overline{CSn}$ and address assertion period (Tt) is extended (n = 7 to 0)

Note: * In burst ROM interface, the CSXTn settings are ignored during CPU read accesses.

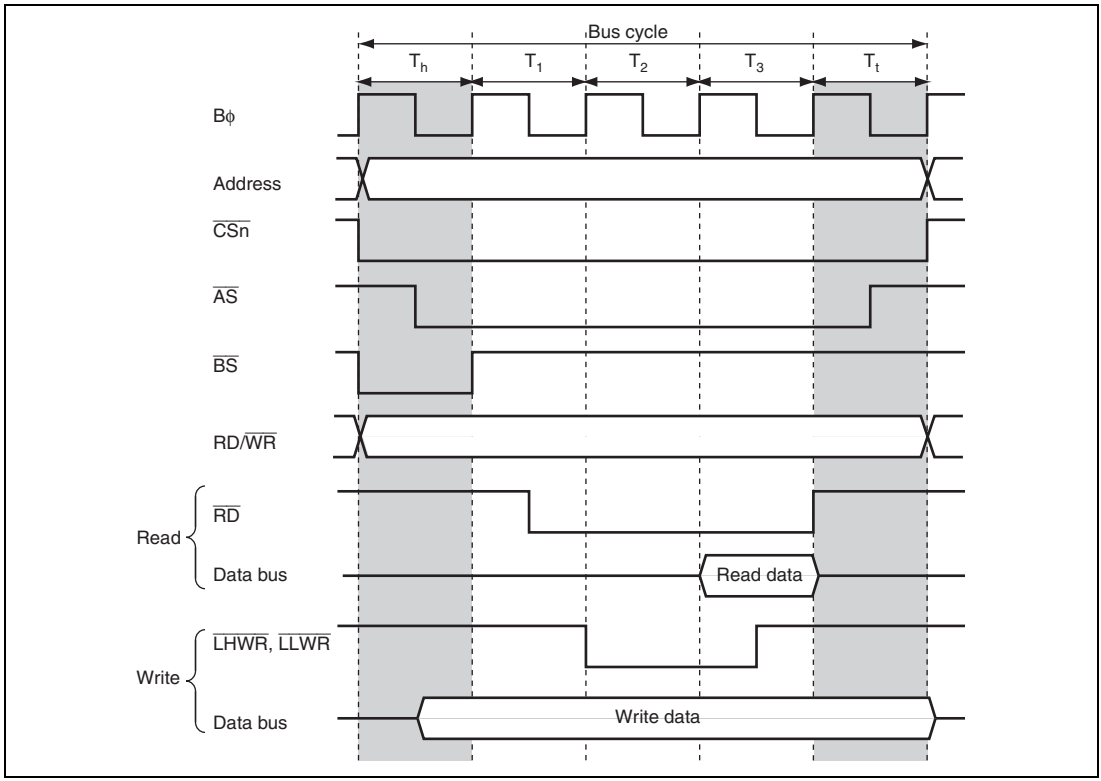


Figure 6.3 $\overline{CS}$ and Address Assertion Period Extension (Example of Basic Bus Interface, 3-State Access Space, and RDNn = 0)

6.2.6 Idle Control Register (IDLCR)

IDLCR specifies the idle cycle insertion conditions and the number of idle cycles.

Bit	15	14	13	12	11	10	9	8
Bit Name	IDLS3	IDLS2	IDLS1	IDLS0	IDLCB1	IDLCB0	IDLCA1	IDLCA0
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
Bit Name	IDLSEL7	IDLSEL6	IDLSEL5	IDLSEL4	IDLSEL3	IDLSEL2	IDLSEL1	IDLSEL0
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	Bit Name	Initial Value	R/W	Description
15	IDLS3	1	R/W	Idle Cycle Insertion 3 Inserts an idle cycle between the bus cycles when the DMAC single address transfer (write cycle) is followed by external access. 0: No idle cycle is inserted 1: An idle cycle is inserted
14	IDLS2	1	R/W	Idle Cycle Insertion 2 Inserts an idle cycle between the bus cycles when the external write cycle is followed by external read cycle. 0: No idle cycle is inserted 1: An idle cycle is inserted
13	IDLS1	1	R/W	Idle Cycle Insertion 1 Inserts an idle cycle between the bus cycles when the external read cycles of different areas continue. 0: No idle cycle is inserted 1: An idle cycle is inserted
12	IDLS0	1	R/W	Idle Cycle Insertion 0 Inserts an idle cycle between the bus cycles when the external read cycle is followed by external write cycle. 0: No idle cycle is inserted 1: An idle cycle is inserted
11	IDLCB1	1	R/W	Idle Cycle State Number Select B
10	IDLCB0	1	R/W	Specifies the number of idle cycles to be inserted for the idle condition specified by IDLS1 and IDLS0. 00: No idle cycle is inserted 01: 2 idle cycles are inserted 00: 3 idle cycles are inserted 01: 4 idle cycles are inserted
9	IDLCA1	1	R/W	Idle Cycle State Number Select A
8	IDLCA0	1	R/W	Specifies the number of idle cycles to be inserted for the idle condition specified by IDLS3 to IDLS0. 00: 1 idle cycle is inserted 01: 2 idle cycles are inserted 10: 3 idle cycles are inserted 11: 4 idle cycles are inserted

Bit	Bit Name	Initial Value	R/W	Description
7	IDLSEL7	0	R/W	Idle Cycle Number Select
6	IDLSEL6	0	R/W	Specifies the number of idle cycles to be inserted for each area for the idle insertion condition specified by IDLS1 and IDLS0. 0: Number of idle cycles to be inserted for area n is specified by IDLCA1 and IDLCA0. 1: Number of idle cycles to be inserted for area n is specified by IDLCB1 and IDLCB0. (n = 7 to 0)
5	IDLSEL5	0	R/W	
4	IDLSEL4	0	R/W	
3	IDLSEL3	0	R/W	
2	IDLSEL2	0	R/W	
1	IDLSEL1	0	R/W	
0	IDLSEL0	0	R/W	

6.2.7 Bus Control Register 1 (BCR1)

BCR1 is used for selection of the external bus released state protocol, enabling/disabling of the write data buffer function, and enabling/disabling of the $\overline{\text{WAIT}}$ pin input.

Bit	15	14	13	12	11	10	9	8
Bit Name	BRLE	BREQOE	—	—	—	—	WDBE	WAITE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DKC	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	BRLE	0	R/W	External Bus Release Enable Enables/disables external bus release. 0: External bus release disabled $\overline{\text{BREQ}}$, $\overline{\text{BACK}}$, and $\overline{\text{BREQO}}$ pins can be used as I/O ports 1: External bus release enabled* For details, see section 9, I/O Ports.

Bit	Bit Name	Initial Value	R/W	Description
14	BREQOE	0	R/W	$\overline{\text{BREQO}}$ Pin Enable Controls outputting the bus request signal ($\overline{\text{BREQO}}$) to the external bus master in the external bus released state when an internal bus master performs an external address space access. 0: $\overline{\text{BREQO}}$ output disabled $\overline{\text{BREQO}}$ pin can be used as I/O port 1: $\overline{\text{BREQO}}$ output enabled
13, 12	—	All 0	R	Reserved These are read-only bits and cannot be modified.
11, 10	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
9	WDBE	0	R/W	Write Data Buffer Enable The write data buffer function can be used for an external write cycle and a DMAC single address transfer cycle. The changed setting may not affect an external access immediately after the change. 0: Write data buffer function not used 1: Write data buffer function used
8	WAITE	0	R/W	$\overline{\text{WAIT}}$ Pin Enable Selects enabling/disabling of wait input by the $\overline{\text{WAIT}}$ pin. 0: Wait input by $\overline{\text{WAIT}}$ pin disabled $\overline{\text{WAIT}}$ pin can be used as I/O port 1: Wait input by $\overline{\text{WAIT}}$ pin enabled For details, see section 9, I/O Ports.
7	DKC	0	R/W	$\overline{\text{DACK}}$ Control Selects the timing of DMAC transfer acknowledge signal assertion. 0: $\overline{\text{DACK}}$ signal is asserted at the Bϕ falling edge 1: $\overline{\text{DACK}}$ signal is asserted at the Bϕ rising edge
6	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
5 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

6.2.8 Bus Control Register 2 (BCR2)

BCR2 is used for bus arbitration control of the CPU, DMAC, and DTC, and enabling/disabling of the write data buffer function to the peripheral modules.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	IBCCS	—	—	—	PWDBE
Initial Value	0	0	0	0	0	0	1	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These are read-only bits and cannot be modified.
5	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
4	IBCCS	0	R/W	Internal Bus Cycle Control Select Selects the internal bus arbiter function. 0: Releases the bus mastership according to the priority 1: Executes the bus cycles alternatively when a CPU bus mastership request conflicts with a DMAC or DTC bus mastership request
3, 2	—	All 0	R	Reserved These are read-only bits and cannot be modified.
1	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
0	PWDBE	0	R/W	Peripheral Module Write Data Buffer Enable Specifies whether or not to use the write data buffer function for the peripheral module write cycles. 0: Write data buffer function not used 1: Write data buffer function used

6.2.9 Endian Control Register (ENDIANCR)

ENDIANCR selects the endian format for each area of the external address space. Though the data format of this LSI is big endian, data can be transferred in the little endian format during external address space access.

Note that the data format for the areas used as a program area or a stack area should be big endian.

Bit	7	6	5	4	3	2	1	0
Bit Name	LE7	LE6	LE5	LE4	LE3	LE2	—	—
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	R

Bit	Bit Name	Initial Value	R/W	Description
7	LE7	0	R/W	Little Endian Select
6	LE6	0	R/W	Selects the endian for the corresponding area.
5	LE5	0	R/W	0: Data format of area n is specified as big endian
4	LE4	0	R/W	1: Data format of area n is specified as little endian
3	LE3	0	R/W	(n = 7 to 2)
2	LE2	0	R/W	
1, 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

6.2.10 SRAM Mode Control Register (SRAMCR)

SRAMCR specifies the bus interface of each area in the external address space as a basic bus interface or a byte control SRAM interface.

In areas specified as 8-bit access space by ABWCR, the SRAMCR setting is ignored and the byte control SRAM interface cannot be specified.

Bit	15	14	13	12	11	10	9	8
Bit Name	BCSEL7	BCSEL6	BCSEL5	BCSEL4	BCSEL3	BCSEL2	BCSEL1	BCSEL0
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
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	BCSEL7	0	R/W	Byte Control SRAM Interface Select
14	BCSEL6	0	R/W	Selects the bus interface for the corresponding area.
13	BCSEL5	0	R/W	When setting a bit to 1, the bus interface select bits in BROMCR and MPXCR must be cleared to 0.
12	BCSEL4	0	R/W	
11	BCSEL3	0	R/W	0: Area n is basic bus interface
10	BCSEL2	0	R/W	1: Area n is byte control SRAM interface
9	BCSEL1	0	R/W	(n = 7 to 0)
8	BCSEL0	0	R/W	
7 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

6.2.11 Burst ROM Interface Control Register (BROMCR)

BROMCR specifies the burst ROM interface.

Bit	15	14	13	12	11	10	9	8
Bit Name	BSRM0	BSTS02	BSTS01	BSTS00	—	—	BSWD01	BSWD00
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

Bit	7	6	5	4	3	2	1	0
Bit Name	BSRM1	BSTS12	BSTS11	BSTS10	—	—	BSWD11	BSWD10
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

Bit	Bit Name	Initial Value	R/W	Description
15	BSRM0	0	R/W	Area 0 Burst ROM Interface Select Specifies the area 0 bus interface. To set this bit to 1, clear bit BCSEL0 in SRAMCR to 0. 0: Basic bus interface or byte-control SRAM interface 1: Burst ROM interface
14	BSTS02	0	R/W	Area 0 Burst Cycle Select
13	BSTS01	0	R/W	Specifies the number of burst cycles of area 0
12	BSTS00	0	R/W	000: 1 cycle 001: 2 cycles 010: 3 cycles 011: 4 cycles 100: 5 cycles 101: 6 cycles 110: 7 cycles 111: 8 cycles
11, 10	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
9	BSWD01	0	R/W	Area 0 Burst Word Number Select
8	BSWD00	0	R/W	Selects the number of words in burst access to the area 0 burst ROM interface 00: Up to 4 words (8 bytes) 01: Up to 8 words (16 bytes) 10: Up to 16 words (32 bytes) 11: Up to 32 words (64 bytes)
7	BSRM1	0	R/W	Area 1 Burst ROM Interface Select Specifies the area 1 bus interface as a basic interface or a burst ROM interface. To set this bit to 1, clear bit BCSEL1 in SRAMCR to 0. 0: Basic bus interface or byte-control SRAM interface 1: Burst ROM interface
6	BSTS12	0	R/W	Area 1 Burst Cycle Select
5	BSTS11	0	R/W	Specifies the number of cycles of area 1 burst cycle
4	BSTS10	0	R/W	000: 1 cycle 001: 2 cycles 010: 3 cycles 011: 4 cycles 100: 5 cycles 101: 6 cycles 110: 7 cycles 111: 8 cycles
3, 2	—	All 0	R	Reserved These are read-only bits and cannot be modified.
1	BSWD11	0	R/W	Area 1 Burst Word Number Select
0	BSWD10	0	R/W	Selects the number of words in burst access to the area 1 burst ROM interface 00: Up to 4 words (8 bytes) 01: Up to 8 words (16 bytes) 10: Up to 16 words (32 bytes) 11: Up to 32 words (64 bytes)

6.2.12 Address/Data Multiplexed I/O Control Register (MPXCR)

MPXCR specifies the address/data multiplexed I/O interface.

When the bus interface of each area in the external address space is specified as a basic interface or a byte control SRAM interface, the MPXCR setting has priority over the SRAMCR setting and the SRAMCR setting is invalid.

Bit	15	14	13	12	11	10	9	8
Bit Name	MPXE7	MPXE6	MPXE5	MPXE4	MPXE3	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R	R

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	ADDEX
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	MPXE7	0	R/W	Address/Data Multiplexed I/O Interface Select
14	MPXE6	0	R/W	Specifies the bus interface for the corresponding area.
13	MPXE5	0	R/W	To set this bit to 1, clear the BCSELn bit in SRAMCR to 0.
12	MPXE4	0	R/W	
11	MPXE3	0	R/W	0: Area n is specified as a basic interface or a byte control SRAM interface. 1: Area n is specified as an address/data multiplexed I/O interface (n = 7 to 3)
10 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	ADDEX	0	R/W	Address Output Cycle Extension Specifies whether a wait cycle is inserted for the address output cycle of address/data multiplexed I/O interface. 0: No wait cycle is inserted for the address output cycle 1: One wait cycle is inserted for the address output cycle

6.3 Bus Configuration

Figure 6.4 shows the internal bus configuration of this LSI. The internal bus of this LSI consists of the following three types.

- Internal system bus

  A bus that connects the CPU, DTC, DMAC, on-chip RAM, on-chip ROM, internal peripheral bus, and external access bus.
- Internal peripheral bus

  A bus that accesses registers in the bus controller, interrupt controller, and DMAC, and registers of peripheral modules such as SCI and timer.
- External access cycle

  A bus that accesses external devices via the external bus interface.

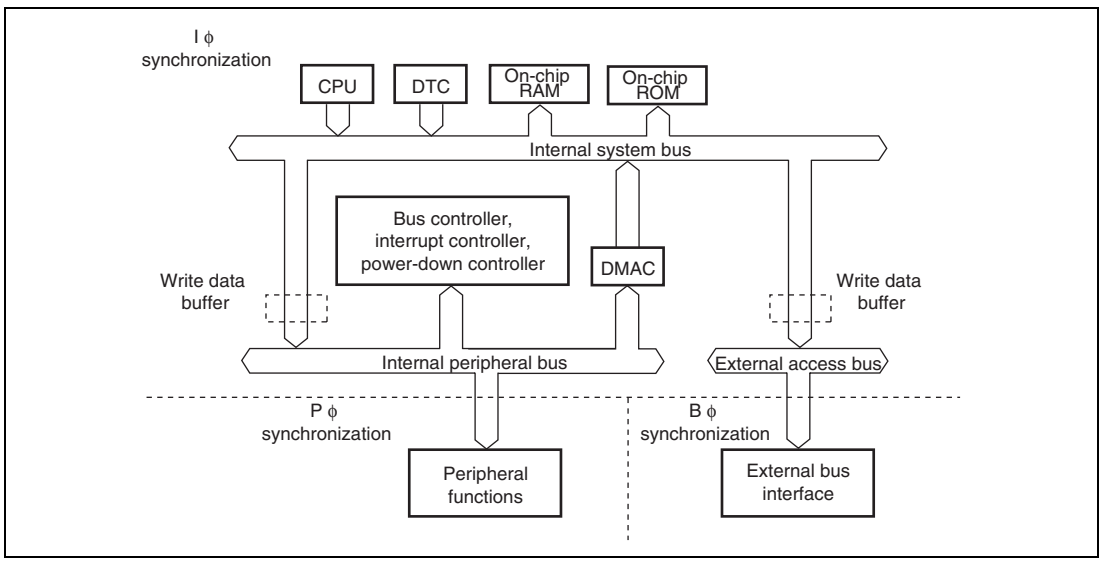


Figure 6.4 Internal Bus Configuration

6.4 Multi-Clock Function and Number of Access Cycles

The internal functions of this LSI operate synchronously with the system clock (Iϕ), the peripheral module clock (Pϕ), or the external bus clock (Bϕ). Table 6.1 shows the synchronization clock and their corresponding functions.

Table 6.1 Synchronization Clocks and Their Corresponding Functions

Synchronization Clock	Function Name
Iϕ	MCU operating mode Interrupt controller Bus controller CPU DTC DMAC Internal memory Clock pulse generator Power down control
Pϕ	I/O ports TPU PPG TMR WDT SCI A/D D/A
Bϕ	External bus interface

The frequency of each synchronization clock (Iϕ, Pϕ, and Bϕ) is specified by the system clock control register (SCKCR) independently. For further details, see section 19, Clock Pulse Generator.

There will be cases when Pϕ and Bϕ are equal to Iϕ and when Pϕ and Bϕ are different from Iϕ according to the SCKCR specifications. In any case, access cycles for internal peripheral functions and external space is performed synchronously with Pϕ and Bϕ, respectively.

For example, in an external address space access where the frequency rate of Iϕ and Bϕ is n : 1, the operation is performed in synchronization with Bϕ. In this case, external 2-state access space is 2n cycles and external 3-state access space is 3n cycles (no wait cycles is inserted) if the number of access cycles is counted based on Iϕ.

If the frequencies of Iϕ, Pϕ and Bϕ are different, the start of bus cycle may not synchronize with Pϕ or Bϕ according to the bus cycle initiation timing. In this case, clock synchronization cycle (Tsy) is inserted at the beginning of each bus cycle.

For example, if an external address space access occurs when the frequency rate of Iϕ and Bϕ is n : 1, 0 to n-1 cycles of Tsy may be inserted. If an internal peripheral module access occurs when the frequency rate of Iϕ and Pϕ is m : 1, 0 to m-1 cycles of Tsy may be inserted.

Figure 6.5 shows the external 2-state access timing when the frequency rate of Iϕ and Bϕ is 4 : 1. Figure 6.6 shows the external 3-state access timing when the frequency rate of Iϕ and Bϕ is 2 : 1.

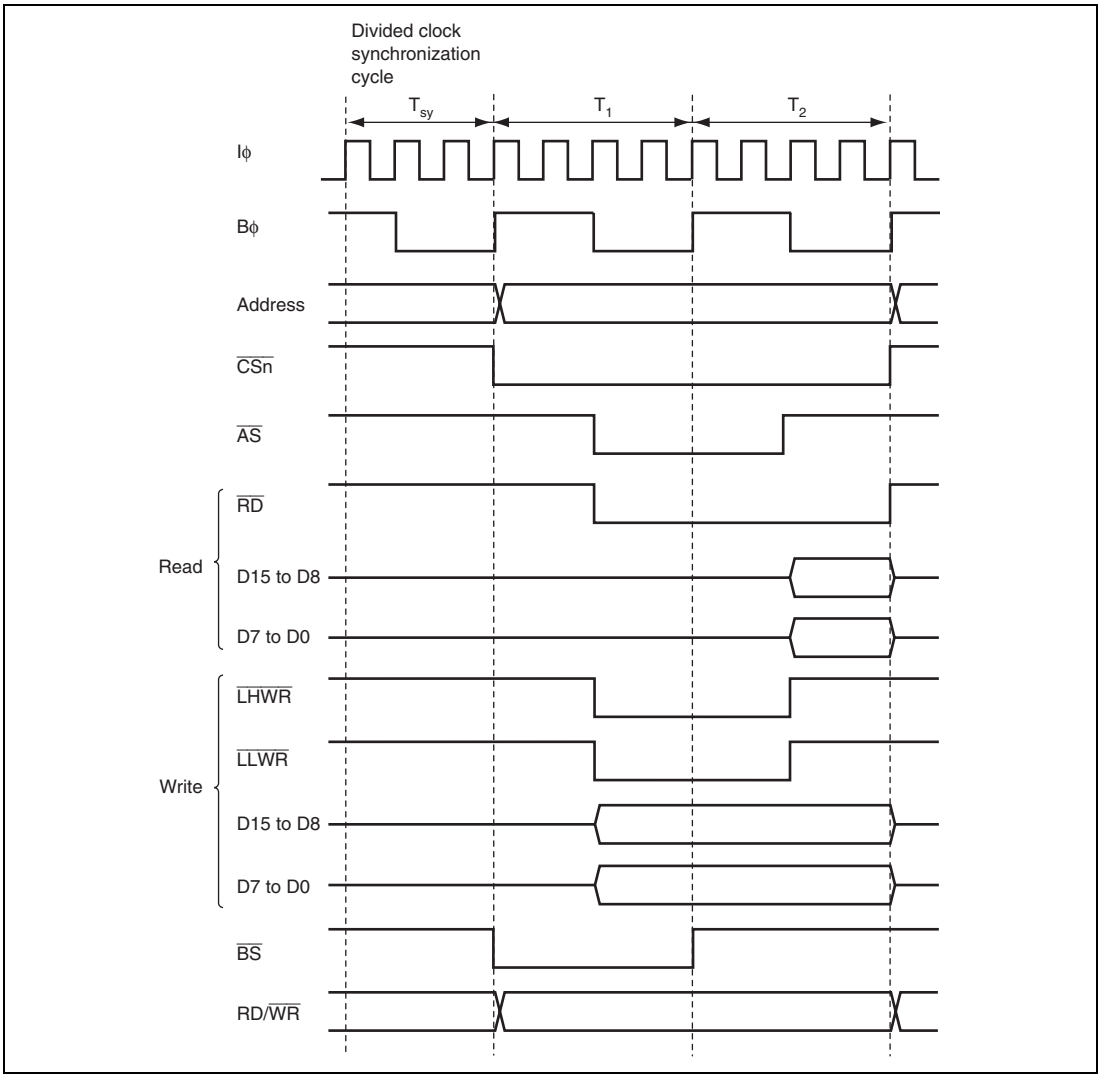


Figure 6.5 System Clock: External Bus Clock = 4:1, External 2-State Access

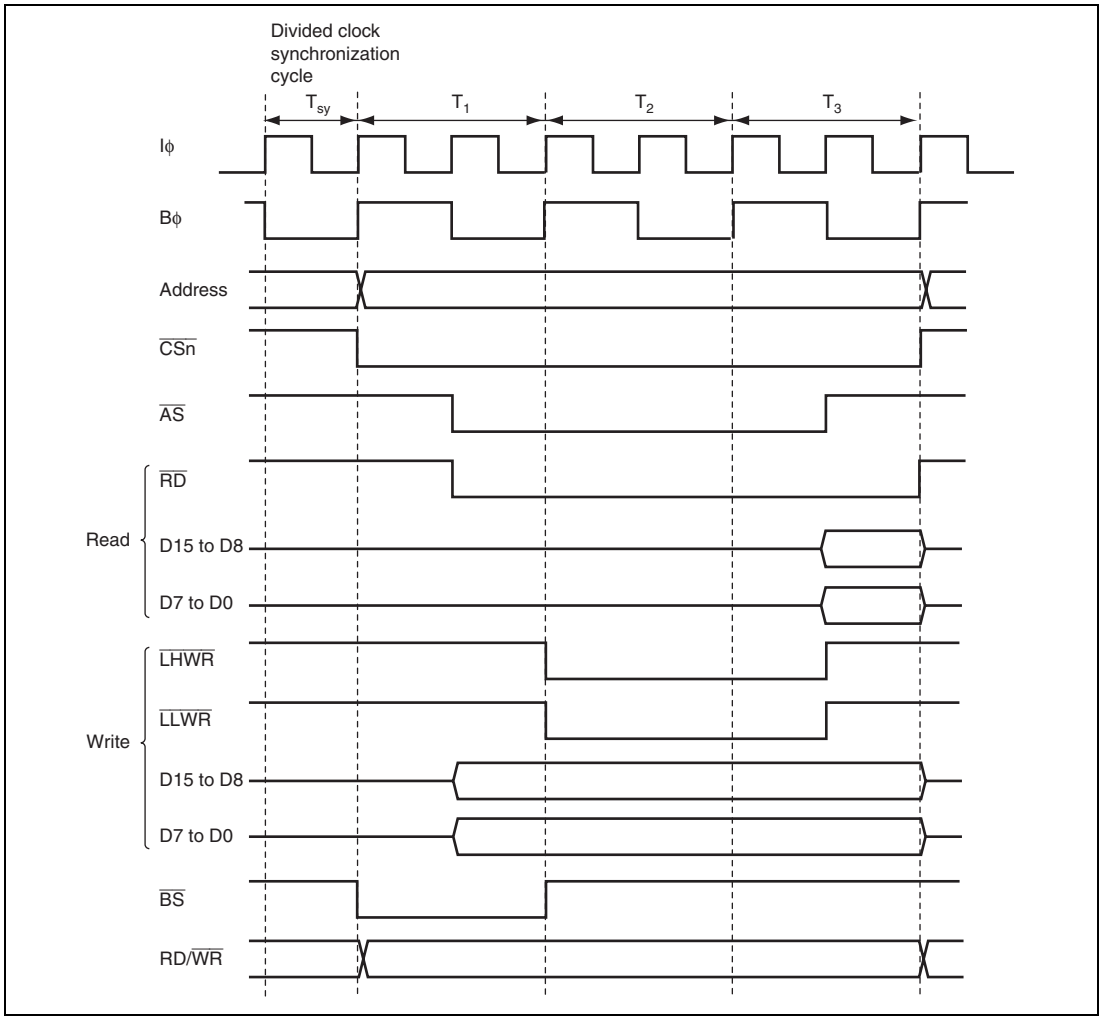


Figure 6.6 System Clock: External Bus Clock = 2:1, External 3-State Access

6.5 External Bus

6.5.1 Input/Output Pins

Table 6.2 shows the pin configuration of the bus controller and table 6.3 shows the pin functions on each interface.

Table 6.2 Pin Configuration

Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started
Address strobe/address hold	$\overline{AS}$/$\overline{AH}$	Output	• Strobe signal indicating that the basic bus, byte control SRAM, or burst ROM space is accessed and address output on address bus is enabled • Signal to hold the address during access to the address/data multiplexed I/O interface
Read strobe	$\overline{RD}$	Output	Strobe signal indicating that the basic bus, byte control SRAM, burst ROM, or address/data multiplexed I/O space is being read
Read/write	RD/$\overline{WR}$	Output	• Signal indicating the input or output direction • Write enable signal of the SRAM during access to the byte control SRAM space
Low-high write/lower-upper byte select	$\overline{LHWR}$/$\overline{LUB}$	Output	• Strobe signal indicating that the basic bus, burst ROM, or address/data multiplexed I/O space is written to, and the upper byte (D15 to D8) of data bus is enabled • Strobe signal indicating that the byte control SRAM space is accessed, and the upper byte (D15 to D8) of data bus is enabled

Name	Symbol	I/O	Function
Low-low write/lower-lower byte select	$\overline{\text{LLWR}}/\overline{\text{LLB}}$	Output	• Strobe signal indicating that the basic bus, burst ROM, or address/data multiplexed I/O space is written to, and the lower byte (D7 to D0) of data bus is enabled • Strobe signal indicating that the byte control SRAM space is accessed, and the lower byte (D7 to D0) of data bus is enabled
Chip select 0	$\overline{\text{CS0}}$	Output	Strobe signal indicating that area 0 is selected
Chip select 1	$\overline{\text{CS1}}$	Output	Strobe signal indicating that area 1 is selected
Chip select 2	$\overline{\text{CS2}}$	Output	Strobe signal indicating that area 2 is selected
Chip select 3	$\overline{\text{CS3}}$	Output	Strobe signal indicating that area 3 is selected
Chip select 4	$\overline{\text{CS4}}$	Output	Strobe signal indicating that area 4 is selected
Chip select 5	$\overline{\text{CS5}}$	Output	Strobe signal indicating that area 5 is selected
Chip select 6	$\overline{\text{CS6}}$	Output	Strobe signal indicating that area 6 is selected
Chip select 7	$\overline{\text{CS7}}$	Output	Strobe signal indicating that area 7 is selected
Wait	$\overline{\text{WAIT}}$	Input	Wait request signal when accessing external address space.
Bus request	$\overline{\text{BREQ}}$	Input	Request signal for release of bus to external bus master
Bus request acknowledge	$\overline{\text{BACK}}$	Output	Acknowledge signal indicating that bus has been released to external bus master
Bus request output	$\overline{\text{BREQO}}$	Output	External bus request signal used when internal bus master accesses external address space in the external-bus released state
Data transfer acknowledge 3 (DMAC_3)	$\overline{\text{DACK3}}$	Output	Data transfer acknowledge signal for DMAC_3 single address transfer
Data transfer acknowledge 2 (DMAC_2)	$\overline{\text{DACK2}}$	Output	Data transfer acknowledge signal for DMAC_2 single address transfer

Name	Symbol	I/O	Function
Data transfer acknowledge 1 (DMAC_1)	$\overline{\text{DACK1}}$	Output	Data transfer acknowledge signal for DMAC_1 single address transfer
Data transfer acknowledge 0 (DMAC_0)	$\overline{\text{DACK0}}$	Output	Data transfer acknowledge signal for DMAC_0 single address transfer
External bus clock	Bϕ	Output	External bus clock

Table 6.3 Pin Functions in Each Interface

	Initial State			Basic Bus		Byte Control SRAM	Burst ROM		Address/Data Multiplexed I/O		
Pin Name	16	8	Single-Chip	16	8	16	16	8	16	8	Remarks
Bϕ	Output	Output	—	O	O	O	O	O	O	O	
$\overline{\text{CS0}}$	Output	Output	—	O	O	O	O	O	—	—	
$\overline{\text{CS1}}$	—	—	—	O	O	O	O	O	—	—	
$\overline{\text{CS2}}$	—	—	—	O	O	O	—	—	—	—	
$\overline{\text{CS3}}$	—	—	—	O	O	O	—	—	O	O	
$\overline{\text{CS4}}$	—	—	—	O	O	O	—	—	O	O	
$\overline{\text{CS5}}$	—	—	—	O	O	O	—	—	O	O	
$\overline{\text{CS6}}$	—	—	—	O	O	O	—	—	O	O	
$\overline{\text{CS7}}$	—	—	—	O	O	O	—	—	O	O	
BS	—	—	—	O	O	O	O	O	O	O	
RD/$\overline{\text{WR}}$	—	—	—	O	O	O	O	O	O	O	
$\overline{\text{AS}}$	Output	Output	—	O	O	O	O	O	—	—	
$\overline{\text{AH}}$	—	—	—	—	—	—	—	—	O	O	
$\overline{\text{RD}}$	Output	Output	—	O	O	O	O	O	O	O	
$\overline{\text{LHWR}}$/$\overline{\text{LUB}}$	Output	Output	—	O	—	O	O	—	O	—	
$\overline{\text{LLWR}}$/$\overline{\text{LLB}}$	Output	Output	—	O	O	O	O	O	O	O	
$\overline{\text{WAIT}}$	—	—	—	O	O	O	O	O	O	O	Controlled by WAITE

[Legend]

O: Used as a bus control signal

—: Not used as a bus control signal (used as a port input when initialized)

6.5.2 Area Division

The bus controller divides the 16-Mbyte address space into eight areas, and performs bus control for the external address space in area units. Chip select signals ($\overline{CS0}$ to $\overline{CS7}$) can be output for each area.

Figure 6.7 shows an area division of the 16-Mbyte address space. For details on address map, see section 3, MCU Operating Modes.

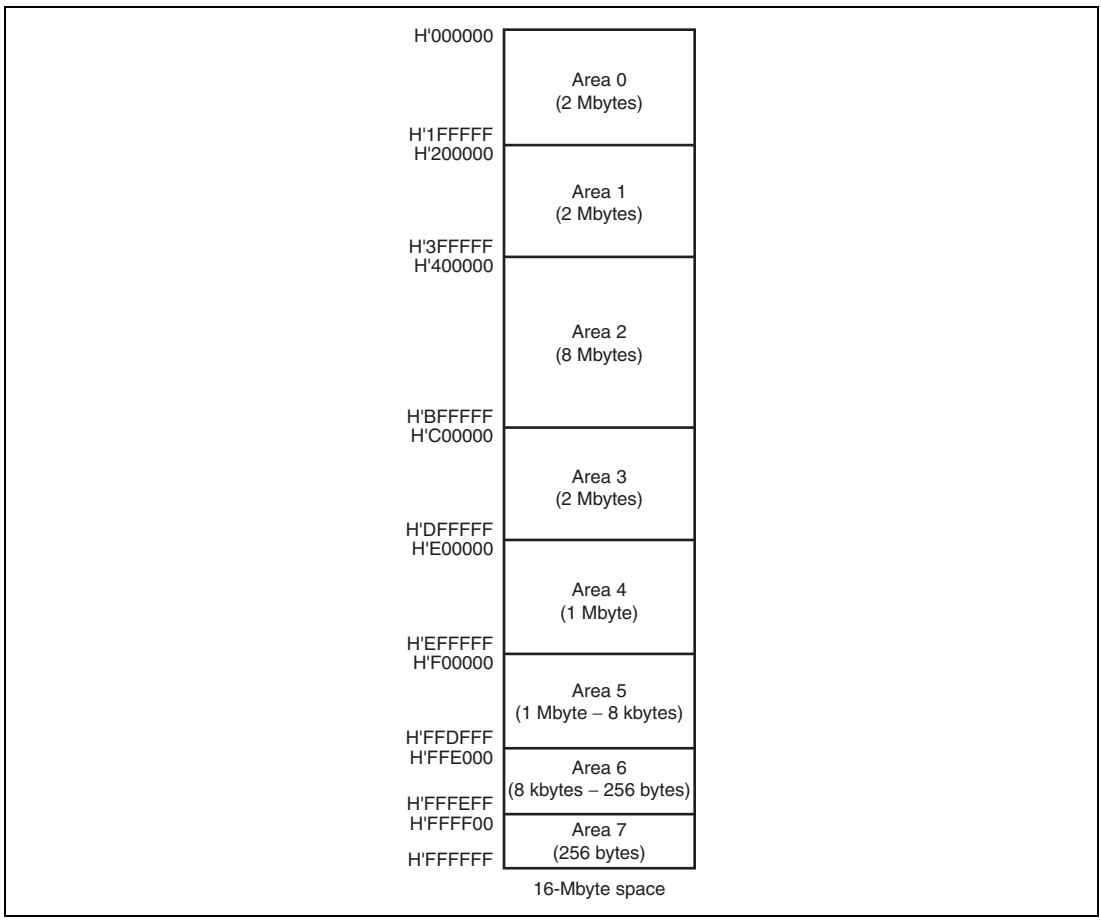


Figure 6.7 Address Space Area Division

6.5.3 Chip Select Signals

This LSI can output chip select signals ($\overline{CS0}$ to $\overline{CS7}$) for areas 0 to 7. The signal outputs low when the corresponding external address space area is accessed. Figure 6.8 shows an example of $\overline{CSn}$ (n = 0 to 7) signal output timing.

Enabling or disabling of $\overline{CSn}$ signal output is set by the port function control register (PFCR). For details, see section 9.3, Port Function Controller.

In on-chip ROM disabled extended mode, pin $\overline{CS0}$ is placed in the output state after a reset. Pins $\overline{CS1}$ to $\overline{CS7}$ are placed in the input state after a reset and so the corresponding PFCR bits should be set to 1 when outputting signals $\overline{CS1}$ to $\overline{CS7}$.

In on-chip ROM enabled extended mode, pins $\overline{CS0}$ to $\overline{CS7}$ are all placed in the input state after a reset and so the corresponding PFCR bits should be set to 1 when outputting signals $\overline{CS0}$ to $\overline{CS7}$.

The PFCR can specify multiple $\overline{CS}$ outputs for a pin. If multiple $\overline{CSn}$ outputs are specified for a single pin by the PFCR, $\overline{CS}$ to be output are generated by mixing all the $\overline{CS}$ signals. In this case, the settings for the external bus interface areas in which the $\overline{CSn}$ signals are output to a single pin should be the same.

Figure 6.9 shows the signal output timing when the $\overline{CS}$ signals to be output to areas 5 and 6 are output to the same pin.

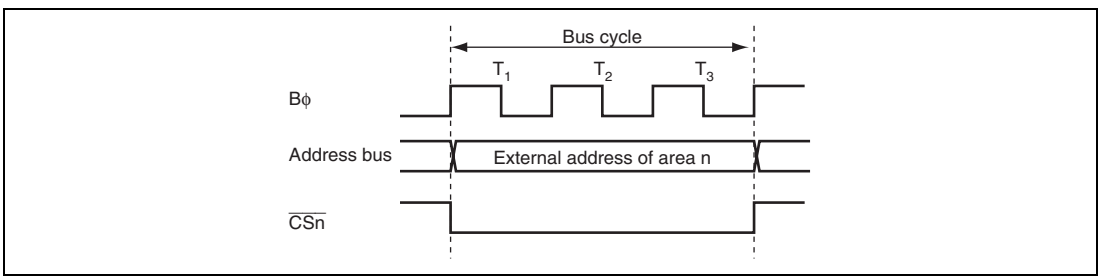


Figure 6.8 $\overline{CSn}$ Signal Output Timing (n = 0 to 7)

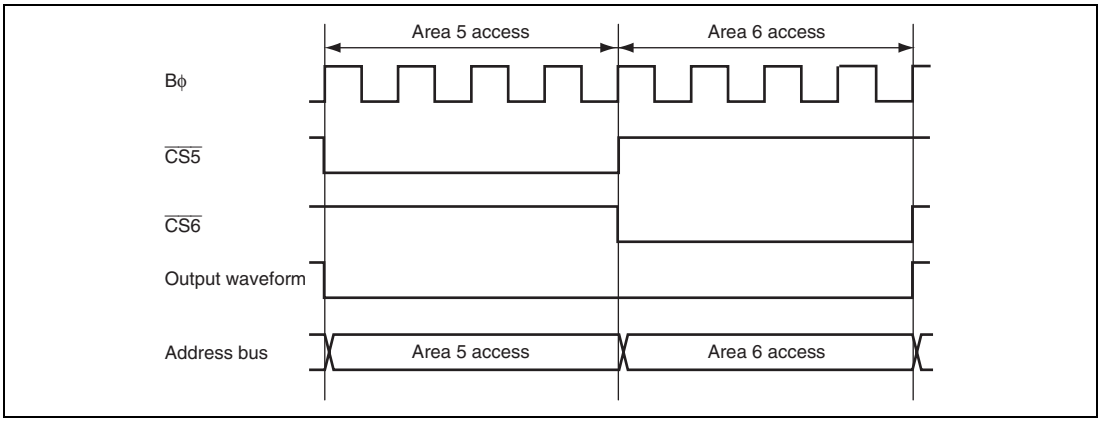


Figure 6.9 Timing When $\overline{CS}$ Signal is Output to the Same Pin

6.5.4 External Bus Interface

The type of the external bus interfaces, bus width, endian format, number of access cycles, and strobe assert/negate timings can be set for each area in the external address space. The bus width and the number of access cycles for both on-chip memory and internal I/O registers are fixed, and are not affected by the external bus settings.

Type of External Bus Interface: Four types of external bus interfaces are provided and can be selected in area units. Table 6.4 shows each interface name, description, area name to be set for each interface. Table 6.5 shows the areas that can be specified for each interface. The initial state of each area is a basic bus interface.

Table 6.4 Interface Names and Area Names

Interface	Description	Area Name
Basic interface	Directly connected to ROM and RAM	Basic bus space
Byte control SRAM interface	Directly connected to byte SRAM with byte control pin	Byte control SRAM space
Burst ROM interface	Directly connected to the ROM that allows page access	Burst ROM space
Address/data multiplexed I/O interface	Directly connected to the peripheral LSI that requires address and data multiplexing	Address/data multiplexed I/O space

Table 6.5 Areas Specifiable for Each Interface

Interface	Related Registers	Areas							
		0	1	2	3	4	5	6	7
Basic interface	SRAMCR	O	O	O	O	O	O	O	O
Byte control SRAM interface		O	O	O	O	O	O	O	O
Burst ROM interface	BROMCR	O	O	—	—	—	—	—	—
Address/data multiplexed I/O interface	MPXCR	—	—	—	O	O	O	O	O

(1) Bus Width

A bus width of 8 or 16 bits can be selected with ABWCR. An area for which an 8-bit bus is selected functions as an 8-bit access space and an area for which a 16-bit bus is selected functions as a 16-bit access space. In addition, the bus width of address/data multiplexed I/O space is 8 bits or 16 bits, and the bus width for the byte control SRAM space is 16 bits.

The initial state of the bus width is specified by the operating mode.

If all areas are designated as 8-bit access space, 8-bit bus mode is set; if any area is designated as 16-bit access space, 16-bit bus mode is set.

(2) Endian Format

Though the endian format of this LSI is big endian, data can be converted into little endian format when reading or writing to the external address space.

Areas 7 to 2 can be specified as either big endian or little endian format by the LE7 to LE2 bits in ENDIANCR.

The initial state of each area is the big endian format.

Note that the data format for the areas used as a program area or a stack area should be big endian.

(3) Number of Access Cycles:

(a) Basic Bus Interface

The number of access cycles in the basic bus interface can be specified as two or three cycles by the ASTCR. An area specified as 2-state access is specified as 2-state access space; an area specified as 3-state access is specified as 3-state access space.

For the 2-state access space, a wait cycle insertion is disabled. For the 3-state access space, a program wait (0 to 7 cycles) specified by WTCRA and WTCRB or an external wait by $\overline{\text{WAIT}}$ can be inserted.

Number of access cycles in the basic bus interface
= number of basic cycles (2, 3) + number of program wait cycles (0 to 7)
+ number of $\overline{\text{CS}}$ extension cycles (0, 1, 2)
[+ number of external wait cycles by the $\overline{\text{WAIT}}$ pin]

Assertion period of the chip select signal can be extended by CSACR.

(b) Byte Control SRAM Interface

The number of access cycles in the byte control SRAM interface is the same as that in the basic bus interface.

Number of access cycles in byte control SRAM interface
= number of basic cycles (2, 3) + number of program wait cycles (0 to 7)
+ number of $\overline{\text{CS}}$ extension cycles (0, 1, 2)
[+ number of external wait cycles by the $\overline{\text{WAIT}}$ pin]

(c) Burst ROM Interface

The number of access cycles at full access in the burst ROM interface is the same as that in the basic bus interface. The number of access cycles in the burst access can be specified as one to eight cycles by the BSTS bit in BROMCR.

Number of access cycles in the burst ROM interface
= number of basic cycles (2, 3) + number of program wait cycles (0 to 7)
+ number of $\overline{\text{CS}}$ extension cycles (0, 1)
[+number of external wait cycles by the $\overline{\text{WAIT}}$ pin]
+ number of burst access cycles (1 to 8) $\times$ number of burst accesses (0 to 63)

(d) Address/data multiplexed I/O interface

The number of access cycles in data cycle of the address/data multiplexed I/O interface is the same as that in the basic bus interface. The number of access cycles in address cycle can be specified as two or three cycles by the ADDEX bit in MPXCR.

Number of access cycles in the address/data multiplexed I/O interface
= number of address output cycles (2, 3) + number of data output cycles (2, 3)
+ number of program wait cycles (0 to 7)
+ number of $\overline{CS}$ extension cycles (0, 1, 2)
[+number of external wait cycles by the $\overline{WAIT}$ pin]

Table 6.6 lists the number of access cycles for each interface.

Table 6.6 Number of Access Cycles

Basic bus interface	=	Th	+T1	+T2				+Tt		
		[0,1]	[1]	[1]				[0,1]		[2 to 4]
	=	Th	+T1	+T2	+Tpw	+TtW	+T3	+Tt		
		[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]		[3 to 12 + n]
Byte control SRAM interface	=	Th	+T1	+T2				+Tt		
		[0,1]	[1]	[1]				[0,1]		[2 to 4]
	=	Th	+T1	+T2	+Tpw	+TtW	+T3	+Tt		
		[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]		[3 to 12 + n]
Burst ROM interface	=	Th	+T1	+T2					+Tb	
		[0,1]	[1]	[1]					[(1 to 8) × m]	[(2 to 3) + (1 to 8) × m]
	=	Th	+T1	+T2	+Tpw	+TtW	+T3		+Tb	
		[0,1]	[1]	[1]	[0 to 7]	[n]	[1]		[(1 to 8) × m]	[(2 to 11 + n) + (1 to 8) × m]
Address/data multiplexed I/O interface	= Tma	+Th	+T1	+T2				+Tt		
	[2,3]	[0,1]	[1]	[1]				[0,1]		[4 to 7]
	= Tma	+Th	+T1	+T2	+Tpw	+TtW	+T3	+Tt		
	[2,3]	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]		[5 to 15 + n]

[Legend]
Numbers: Number of access cycles
n: Pin wait (0 to ∞)
m: Number of burst accesses (0 to 63)

(4) Strobe Assert/Negate Timings

The assert and negate timings of the strobe signals can be modified as well as number of access cycles.

- Read strobe ($\overline{RD}$) in the basic bus interface
- Chip select assertion period extension cycles in the basic bus interface
- Data transfer acknowledge ($\overline{DACK3}$ to $\overline{DACK0}$) output for DMAC single address transfers

6.5.5 Area and External Bus Interface

(1) Area 0

Area 0 includes on-chip ROM*. All of area 0 is used as external address space in on-chip ROM disabled extended mode, and the space excluding on-chip ROM is external address space in on-chip ROM enabled extended mode.

When area 0 external address space is accessed, the $\overline{CS0}$ signal can be output.

Either of the basic bus interface, byte control SRAM interface, or burst ROM interface can be selected for area 0 by bit BSRM0 in BROMCR and bit BCSEL0 in SRAMCR. Table 6.7 shows the external interface of area 0.

Note: Applied to the LSI version that incorporates the ROM.

Table 6.7 Area 0 External Interface

	Register Setting	
Interface	**BSRM0 of BROMCR**	**BCSEL0 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Burst ROM interface	1	0
Setting prohibited	1	1

(2) Area 1

In externally extended mode, all of area 1 is external address space. In on-chip ROM enabled extended mode, the space excluding on-chip ROM* is external address space.

When area 1 external address space is accessed, the $\overline{CS1}$ signal can be output.

Either of the basic bus interface, byte control SRAM, or burst ROM interface can be selected for area 1 by bit BSRM1 in BROMCR and bit BCSEL1 in SRAMCR. Table 6.8 shows the external interface of area 1.

Note: Applied to the LSI version that incorporates the ROM.

Table 6.8 Area 1 External Interface

	Register Setting	
Interface	**BSRM1 of BROMCR**	**BCSEL1 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Burst ROM interface	1	0
Setting prohibited	1	1

(3) Area 2

In externally extended mode, all of area 2 is external address space.

When area 2 external address space is accessed, the $\overline{CS2}$ signal can be output.

Either the basic bus interface or byte control SRAM interface can be selected for area 2 by bit BCSEL2 in SRAMCR. Table 6.9 shows the external interface of area 2.

Table 6.9 Area 2 External Interface

	Register Setting
Interface	**BCSEL2 of SRAMCR**
Basic bus interface	0
Byte control SRAM interface	1

(4) Area 3

In externally extended mode, all of area 3 is external address space.

When area 3 external address space is accessed, the $\overline{CS3}$ signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 3 by bit MPXE3 in MPXCR and bit BCSEL3 in SRAMCR. Table 6.10 shows the external interface of area 3.

Table 6.10 Area 3 External Interface

	Register Setting	
Interface	**MPXE3 of MPXCR**	**BCSEL3 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

(5) Area 4

In externally extended mode, all of area 4 is external address space.

When area 4 external address space is accessed, the $\overline{CS4}$ signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 4 by bit MPXE4 in MPXCR and bit BCSEL4 in SRAMCR. Table 6.11 shows the external interface of area 4.

Table 6.11 Area 4 External Interface

	Register Setting	
Interface	**MPXE4 of MPXCR**	**BCSEL4 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

(6) Area 5

Area 5 includes the on-chip RAM and access prohibited spaces. In external extended mode, area 5, other than the on-chip RAM and access prohibited spaces, is external address space. Note that the on-chip RAM is enabled when the RAME bit in SYSCR are set to 1. If the RAME bit in SYSCR is cleared to 0, the on-chip RAM is disabled and the corresponding addresses are an external address space. For details, see section 3, MCU Operating Modes.

When area 5 external address space is accessed, the $\overline{CS5}$ signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 5 by the MPXE5 bit in MPXCR and the BCSEL5 bit in SRAMCR. Table 6.12 shows the external interface of area 5.

Table 6.12 Area 5 External Interface

	Register Setting	
Interface	**MPXE5 of MPXCR**	**BCSEL5 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

(7) Area 6

Area 6 includes internal I/O registers. In external extended mode, area 6 other than on-chip I/O register area is external address space.

When area 6 external address space is accessed, the $\overline{CS6}$ signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 6 by the MPXE6 bit in MPXCR and the BCSEL6 bit in SRAMCR. Table 6.13 shows the external interface of area 6.

Table 6.13 Area 6 External Interface

	Register Setting	
Interface	**MPXE6 of MPXCR**	**BCSEL6 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

(8) Area 7

Area 7 includes internal I/O registers. In external extended mode, area 7 other than internal I/O register area is external address space.

When area 7 external address space is accessed, the $\overline{CS7}$ signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 7 by the MPXE7 bit in MPXCR and the BCSEL7 bit in SRAMCR. Table 6.14 shows the external interface of area 7.

Table 6.14 Area 7 External Interface

	Register Setting	
Interface	**MPXE7 of MPXCR**	**BCSEL7 of SRAMCR**
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

6.5.6 Endian and Data Alignment

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and controls whether the upper byte data bus (D15 to D8) or lower data bus (D7 to D0) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space), the data size, and endian format when accessing external address space.

(1) 8-Bit Access Space

With the 8-bit access space, the lower byte data bus (D7 to D0) is always used for access. The amount of data that can be accessed at one time is one byte: a word access is performed as two byte accesses, and a longword access, as four byte accesses.

Figures 6.10 and 6.11 illustrate data alignment control for the 8-bit access space. Figure 6.10 shows the data alignment when the data endian format is specified as big endian. Figure 6.11 shows the data alignment when the data endian format is specified as little endian.

Data Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe signal LHWR/LUB, LLWR/LLB, RD; Data bus D15 D8 D7 D0
Byte	n	1	1st	Byte	7 0
Word	n	2	1st	Byte	15 8
			2nd	Byte	7 0
Longword	n	4	1st	Byte	31 24
			2nd	Byte	23 16
			3rd	Byte	15 8
			4th	Byte	7 0

Figure 6.10 Access Sizes and Data Alignment Control for 8-Bit Access Space (Big Endian)

Data Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe signal LHWR/LUB, LLWR/LLB, RD; Data bus D15 D8 D7 D0
Byte	n	1	1st	Byte	7 0
Word	n	2	1st	Byte	7 0
			2nd	Byte	15 8
Longword	n	4	1st	Byte	7 0
			2nd	Byte	15 8
			3rd	Byte	23 16
			4th	Byte	31 24

Figure 6.11 Access Sizes and Data Alignment Control for 8-Bit Access Space (Little Endian)

(2) 16-Bit Access Space

With the 16-bit access space, the upper byte data bus (D15 to D8) and lower byte data bus (D7 to D0) are used for accesses. The amount of data that can be accessed at one time is one byte or one word.

Figures 6.12 and 6.13 illustrate data alignment control for the 16-bit access space. Figure 6.12 shows the data alignment when the data endian format is specified as big endian. Figure 6.13 shows the data alignment when the data endian format is specified as little endian.

In big endian, byte access for an even address is performed by using the upper byte data bus and byte access for an odd address is performed by using the lower byte data bus.

In little endian, byte access for an even address is performed by using the lower byte data bus, and byte access for an odd address is performed by using the third byte data bus.

Access Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe signal LHWR/LUB LLWR/LLB; RD; Data bus D15 D8 D7 D0
Byte	Even (2n)	1	1st	Byte	7 0
	Odd (2n+1)	1	1st	Byte	7 0
Word	Even (2n)	1	1st	Word	15 8 7 0
	Odd (2n+1)	2	1st	Byte	15 8
			2nd	Byte	7 0
Longword	Even (2n)	2	1st	Word	31 24 23 16
			2nd	Word	15 8 7 0
	Odd (2n+1)	3	1st	Byte	31 24
			2nd	Word	23 16 15 8
			3rd	Byte	7 0

Figure 6.12 Access Sizes and Data Alignment Control for 16-Bit Access Space (Big Endian)

Access Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe signal LHWR/LUB LLWR/LLB; RD; Data bus D15 D8 D7 D0
Byte	Even (2n)	1	1st	Byte	7 0
	Odd (2n+1)	1	1st	Byte	7 0
Word	Even (2n)	1	1st	Word	15 8 7 0
	Odd (2n+1)	2	1st	Byte	7 0
			2nd	Byte	15 8
Longword	Even (2n)	2	1st	Word	15 8 7 0
			2nd	Word	31 24 23 16
	Odd (2n+1)	3	1st	Byte	7 0
			2nd	Word	23 16 15 8
			3rd	Byte	31 24

Figure 6.13 Access Sizes and Data Alignment Control for 16-Bit Access Space (Little Endian)

6.6 Basic Bus Interface

The basic bus interface can be connected directly to the ROM and SRAM. The bus specifications can be specified by the ABWCR, ASTCR, WTCRA, WTCRB, RDNCR, CSACR, and ENDINCR.

6.6.1 Data Bus

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and controls whether the upper byte data bus (D15 to D8) or lower byte data bus (D7 to D0) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space), the data size, and endian format when accessing external address space,. For details, see section 6.5.6, Endian and Data Alignment.

6.6.2 I/O Pins Used for Basic Bus Interface

Table 6.15 shows the pins used for basic bus interface.

Table 6.15 I/O Pins for Basic Bus Interface

Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started
Address strobe	$\overline{AS}$*	Output	Strobe signal indicating that an address output on the address bus is valid during access
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Read/write	RD/$\overline{WR}$	Output	Signal indicating the data bus input or output direction
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid during write access
Low-low write	$\overline{LLWR}$	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid during write access
Chip select 0 to 7	$\overline{CS0}$ to $\overline{CS7}$	Output	Strobe signal indicating that the area is selected
Wait	$\overline{WAIT}$	Input	Wait request signal used when an external address space is accessed

Note: * When the address/data multiplexed I/O is selected, this pin only functions as the AH output and does not function as the AS output.

6.6.3 Basic Timing

This section describes the basic timing when the data is specified as big endian.

(1) 16-Bit 2-State Access Space

Figures 6.14 to 6.16 show the bus timing of 16-bit 2-state access space.

When accessing 16-bit access space, the upper byte data bus (D15 to D8) is used for even addresses access, and the lower byte data bus (D7 to D0) is used for odd addresses. No wait cycles can be inserted.

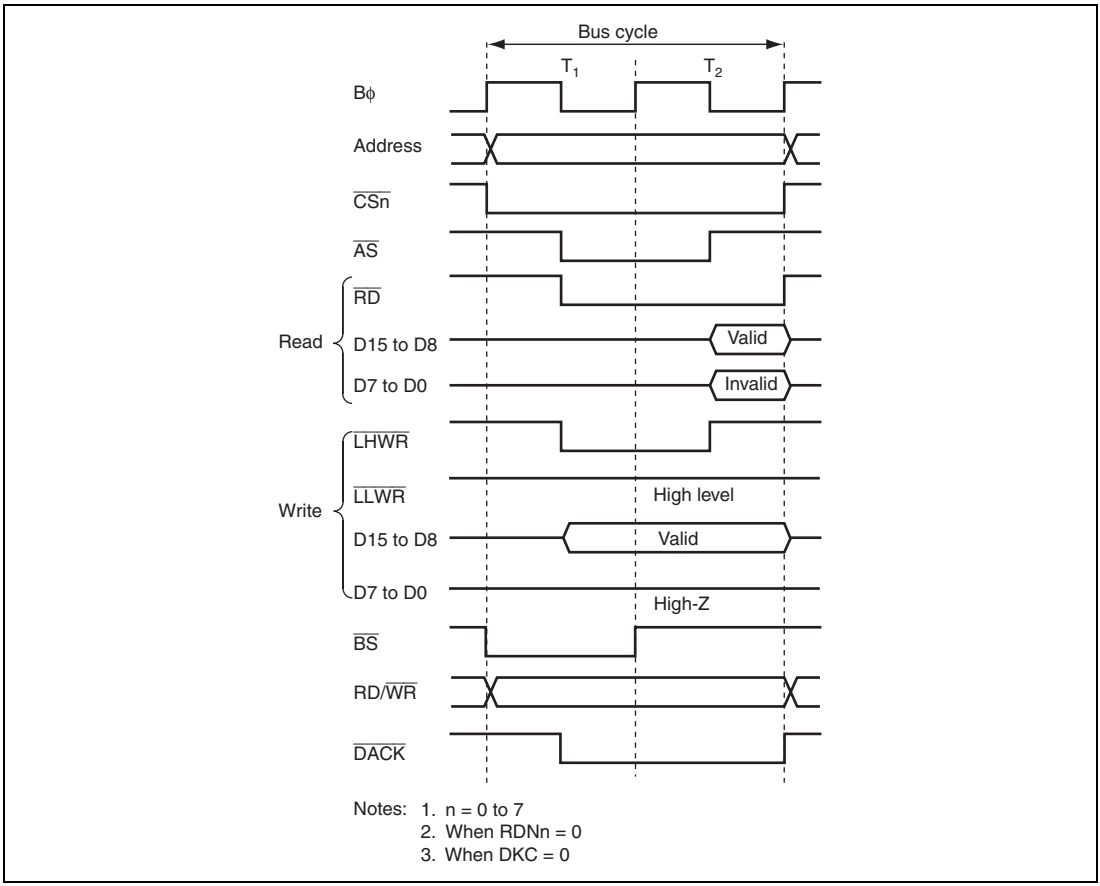
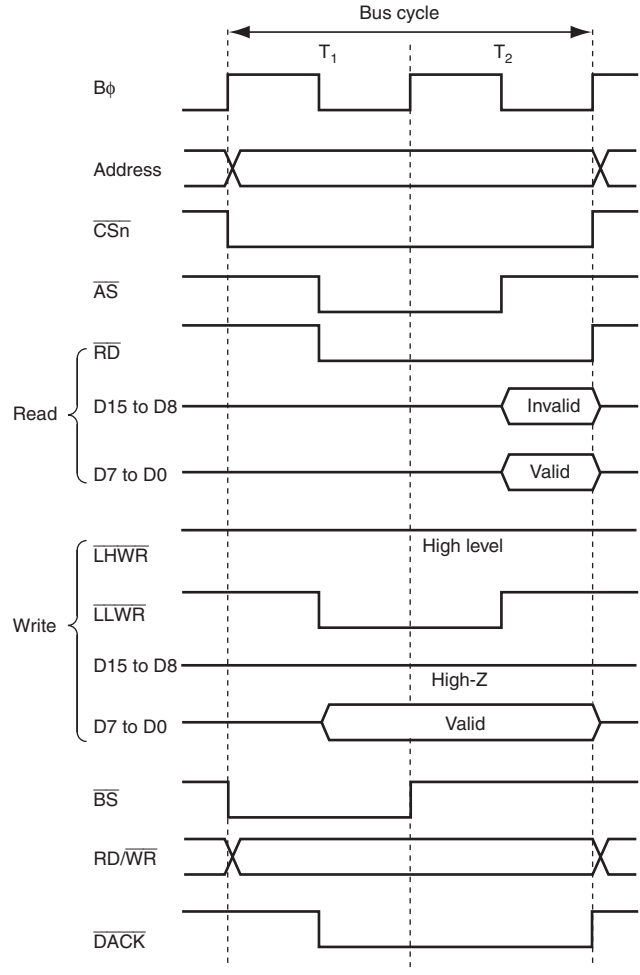


Figure 6.14 16-Bit 2-State Access Space Bus Timing (Byte Access for Even Address)



Notes: 1. n = 0 to 7
2. When RDNn = 0
3. When DKC = 0

Figure 6.15 16-Bit 2-State Access Space Bus Timing (Byte Access for Odd Address)

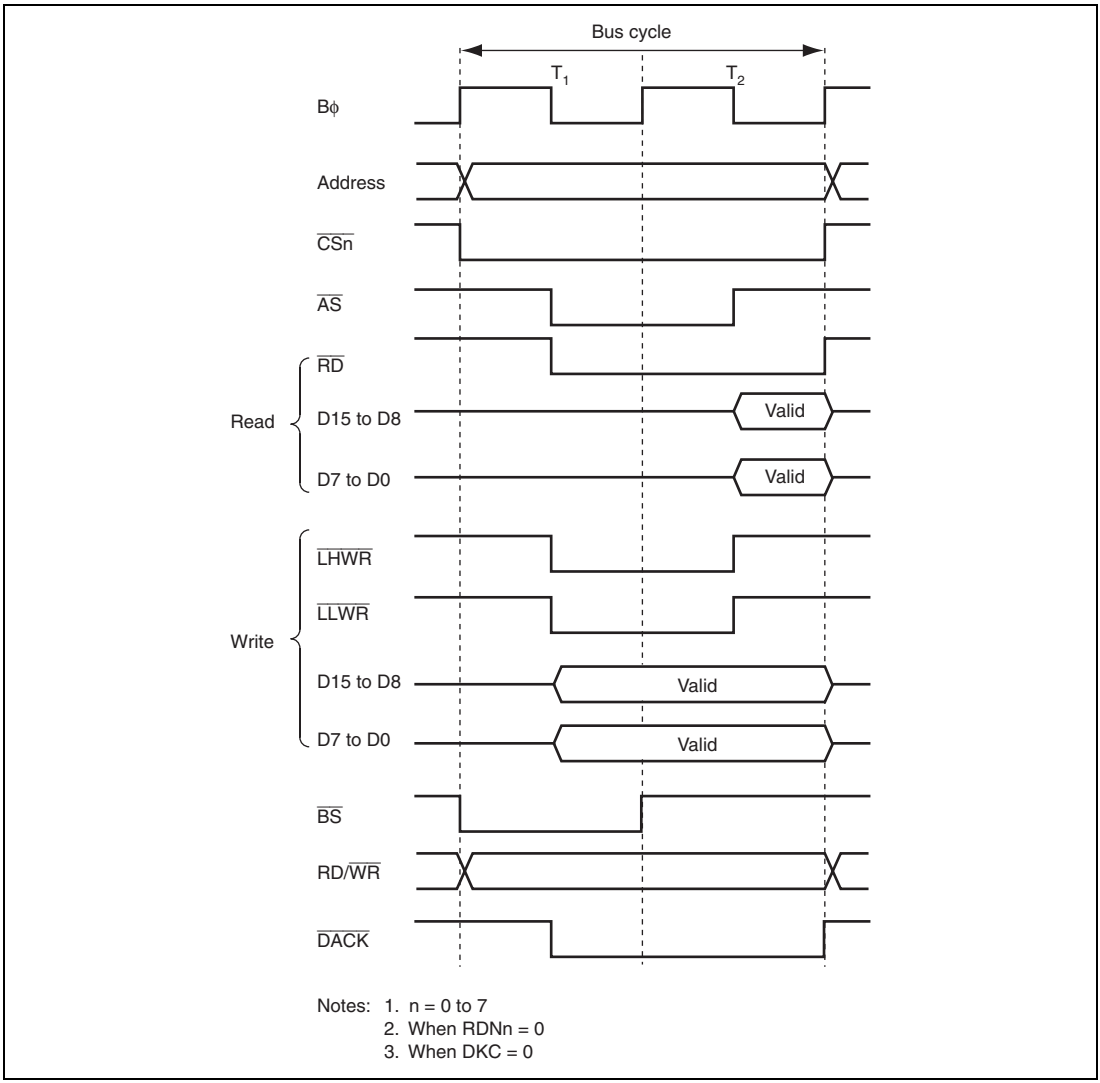


Figure 6.16 16-Bit 2-State Access Space Bus Timing (Word Access for Even Address)

(2) 16-Bit 3-State Access Space

Figures 6.17 to 6.19 show the bus timing of 16-bit 3-state access space.

When accessing 16-bit access space, the upper byte data bus (D15 to D8) is used for even addresses, and the lower byte data bus (D7 to D0) is used for odd addresses. Wait cycles can be inserted.

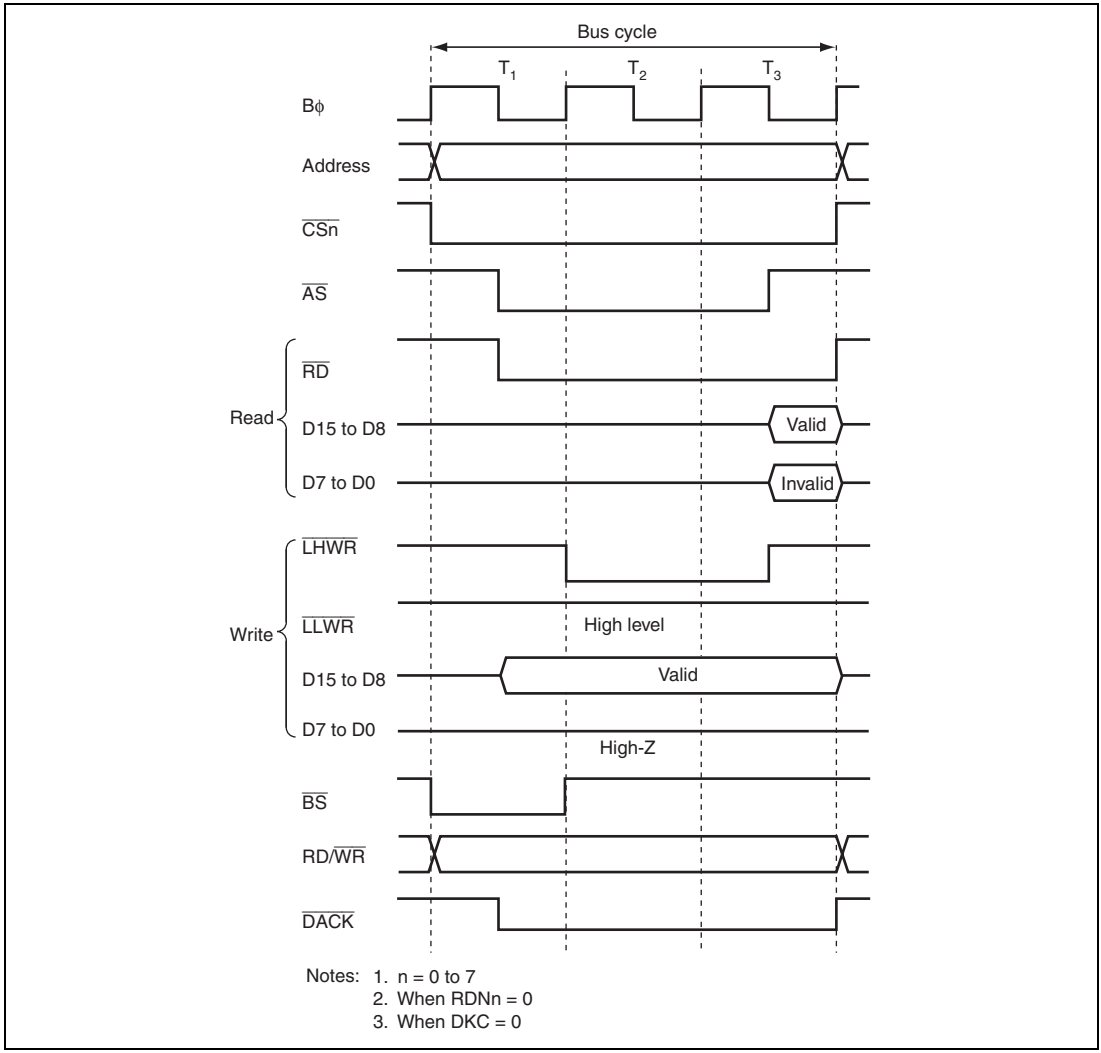


Figure 6.17 16-Bit 3-State Access Space Bus Timing (Byte Access for Even Address)

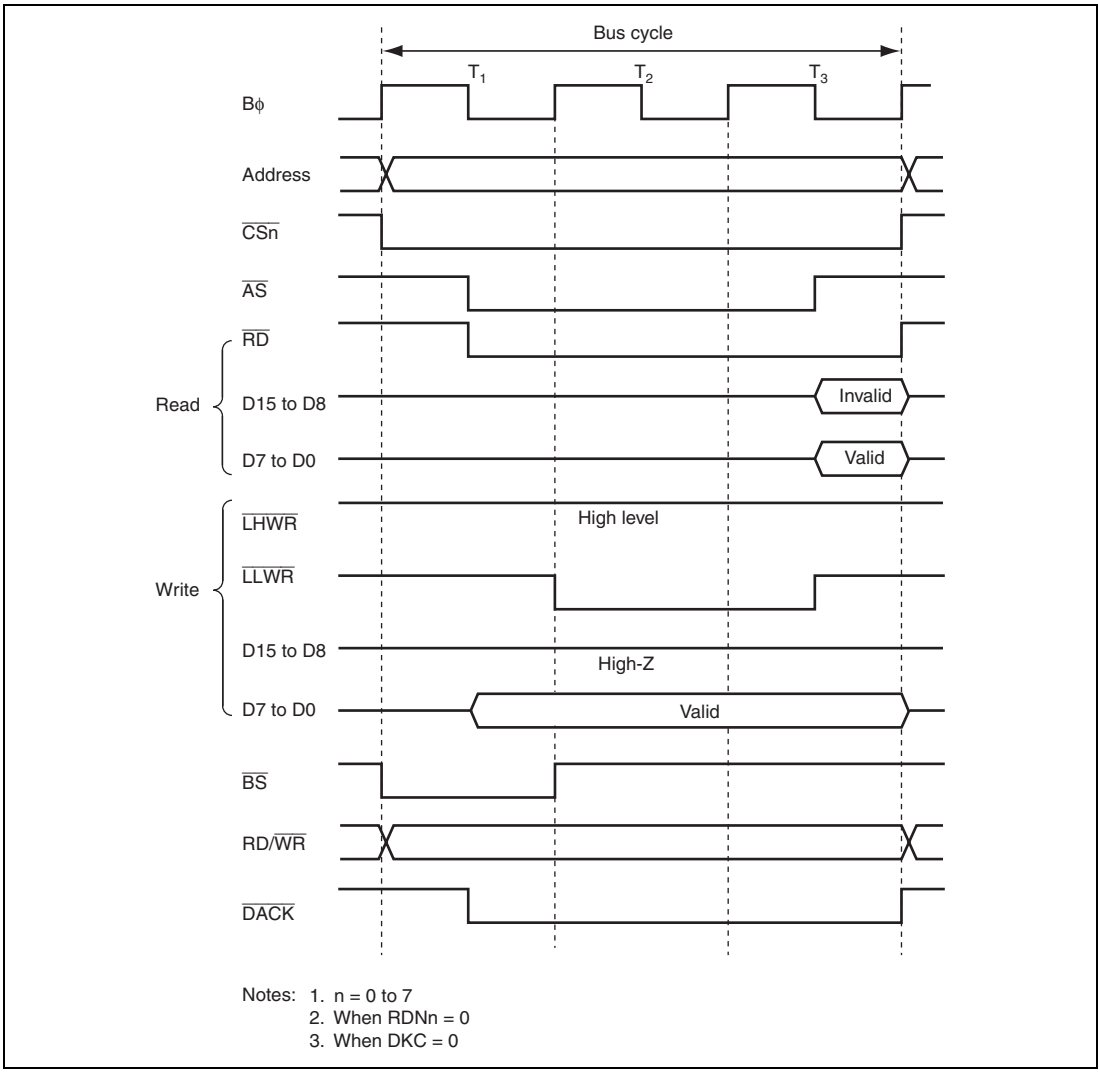


Figure 6.18 16-Bit 3-State Access Space Bus Timing (Word Access for Odd Address)

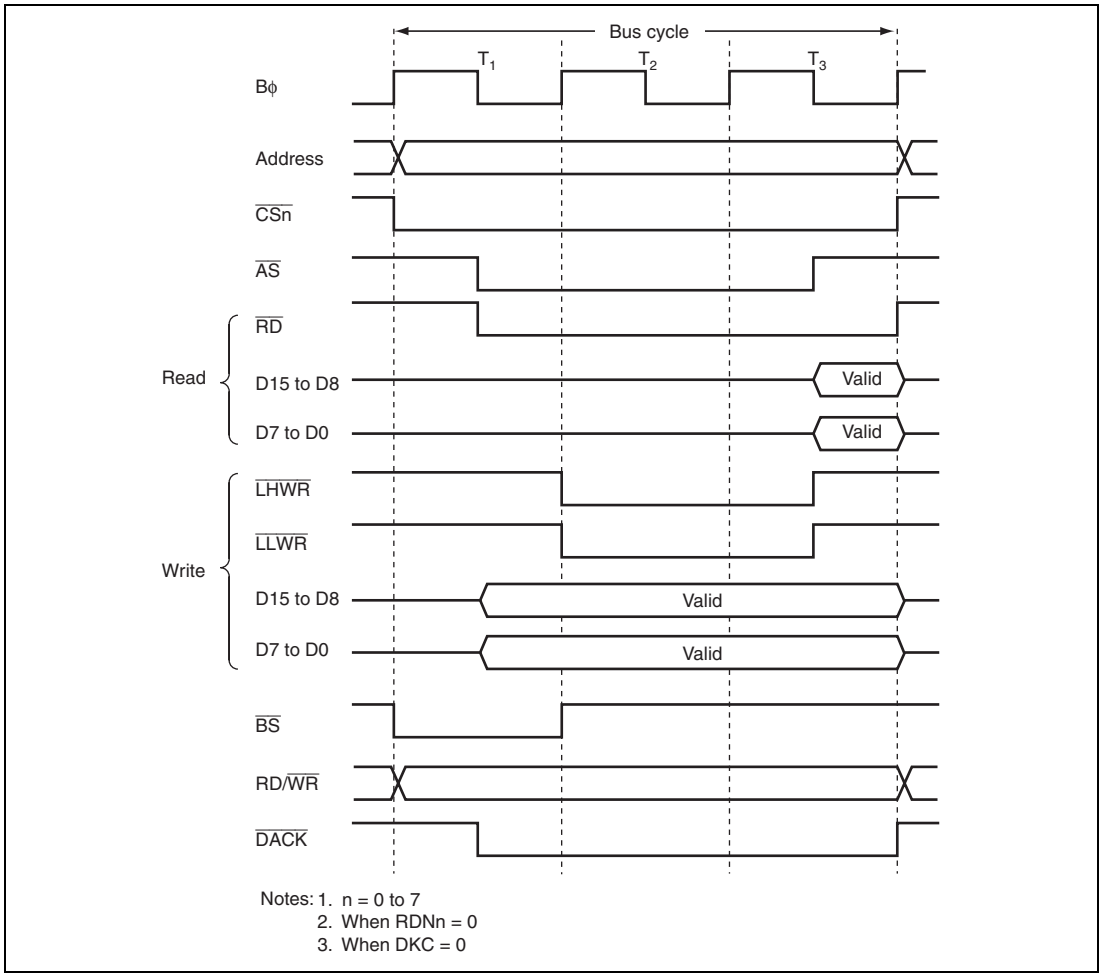


Figure 6.19 16-Bit 3-State Access Space Bus Timing (Word Access for Even Address)

6.6.4 Wait Control

This LSI can extend the bus cycle by inserting wait cycles (Tw) when the external address space is accessed. There are two ways of inserting wait cycles: program wait (Tpw) insertion and pin wait (Ttw) insertion using the $\overline{\text{WAIT}}$ pin.

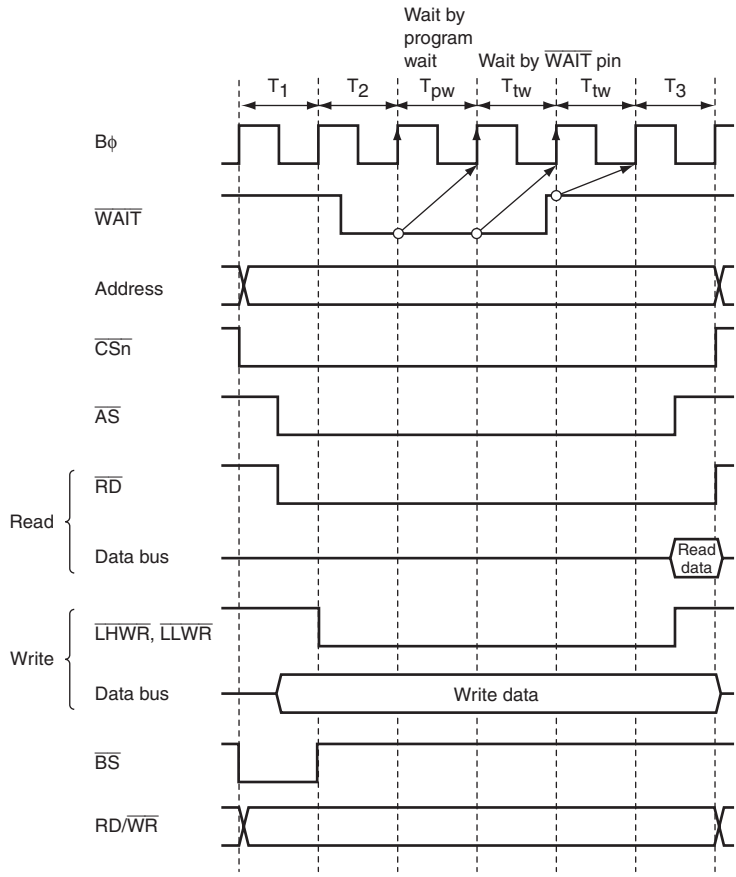
(1) Program Wait Insertion

From 0 to 7 wait cycles can be inserted automatically between the T_2 state and T_3 state for 3-state access space, according to the settings in WTCRA and WTCRB.

(2) Pin Wait Insertion

For 3-state access space, when the WAITE bit in BCR1 is set to 1 and the corresponding ICR bit is set to 1, wait input by means of the $\overline{\text{WAIT}}$ pin is enabled. When the external address space is accessed in this state, a program wait (Tw) is first inserted according to the WTCRA and WTCRB settings. If the $\overline{\text{WAIT}}$ pin is low at the falling edge of Bϕ in the last T2 or Tpw cycle, another Ttw cycle is inserted until the $\overline{\text{WAIT}}$ pin is brought high. The pin wait insertion is effective when the Tw cycles are inserted to seven cycles or more, or when the number of Tw cycles to be inserted is changed according to the external devices. The WAITE bit is common to all areas. For details on ICR, see section 9, I/O Ports.

Figure 6.20 shows an example of wait cycle insertion timing. After a reset, the 3-state access is specified, the program wait is inserted for seven cycles, and the $\overline{\text{WAIT}}$ input is disabled.



Notes: 1. Upward arrows indicate the timing of WAIT pin sampling.
2. n = 0 to 7
3. When RDNn = 0

Figure 6.20 Example of Wait Cycle Insertion Timing

6.6.5 Read Strobe ($\overline{RD}$) Timing

The read strobe timing can be modified in area units by setting bits RDN7 to RDN0 in RDNCR to 1.

Note that the $\overline{RD}$ timing with respect to the $\overline{DACK}$ rising edge will change if the read strobe timing is modified by setting RDNn to 1 when the DMAC is used in the single address mode.

Figure 6.21 shows an example of timing when the read strobe timing is changed in the basic bus 3-state access space.

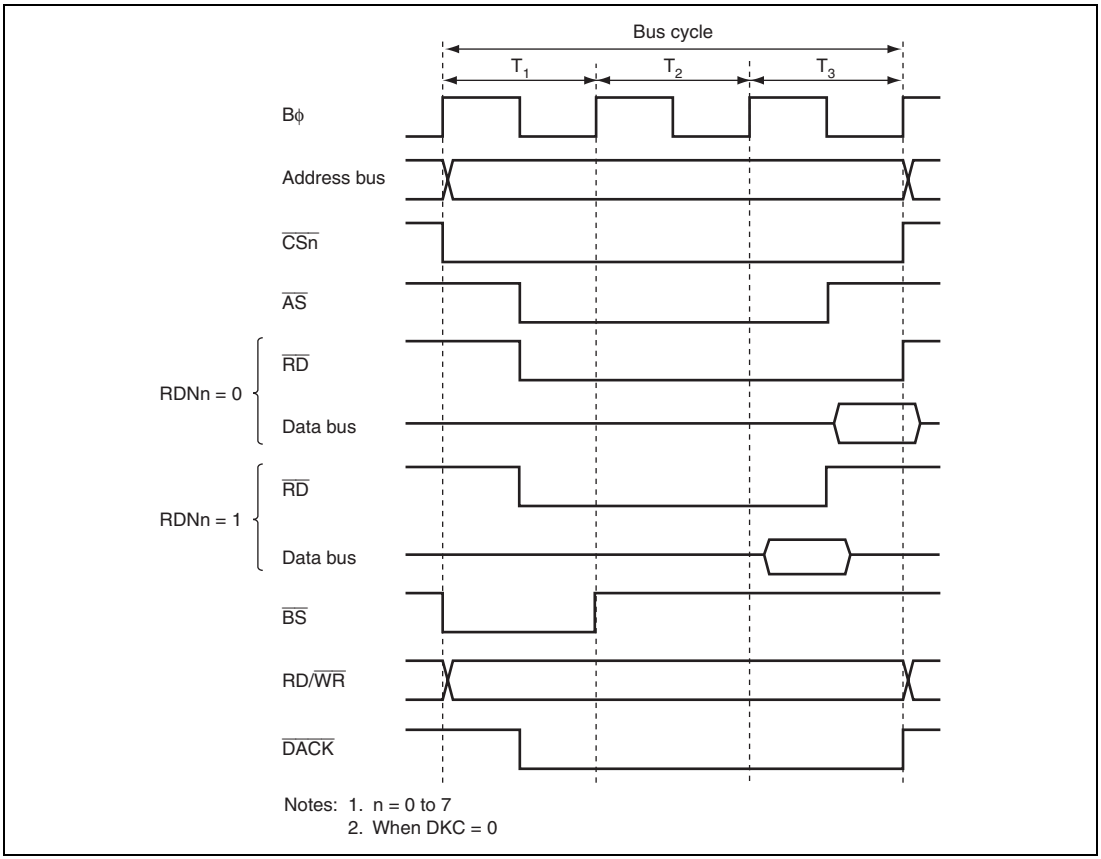


Figure 6.21 Example of Read Strobe Timing

6.6.6 Extension of Chip Select ($\overline{CS}$) Assertion Period

Some external I/O devices require a setup time and hold time between address and $\overline{CS}$ signals and strobe signals such as $\overline{RD}$, $\overline{LHWR}$, and $\overline{LLWR}$.

Settings can be made in CSACR to insert cycles in which only the $\overline{CS}$, $\overline{AS}$, and address signals are asserted before and after a basic bus space access cycle. Extension of the $\overline{CS}$ assertion period can be set in area units. With the $\overline{CS}$ assertion extension period in write access, the data setup and hold times are less stringent since the write data is output to the data bus.

Figure 6.22 shows an example of the timing when the $\overline{CS}$ assertion period is extended in basic bus 3-state access space.

Both extension cycle Th inserted before the basic bus cycle and extension cycle Tt inserted after the basic bus cycle, or only one of these, can be specified for individual areas. Insertion or non-insertion can be specified for the Th cycle with the upper eight bits (CSXH7 to CSXH0) in CSACR, and for the Tt cycle with the lower eight bits (CSXT7 to CSXT0).

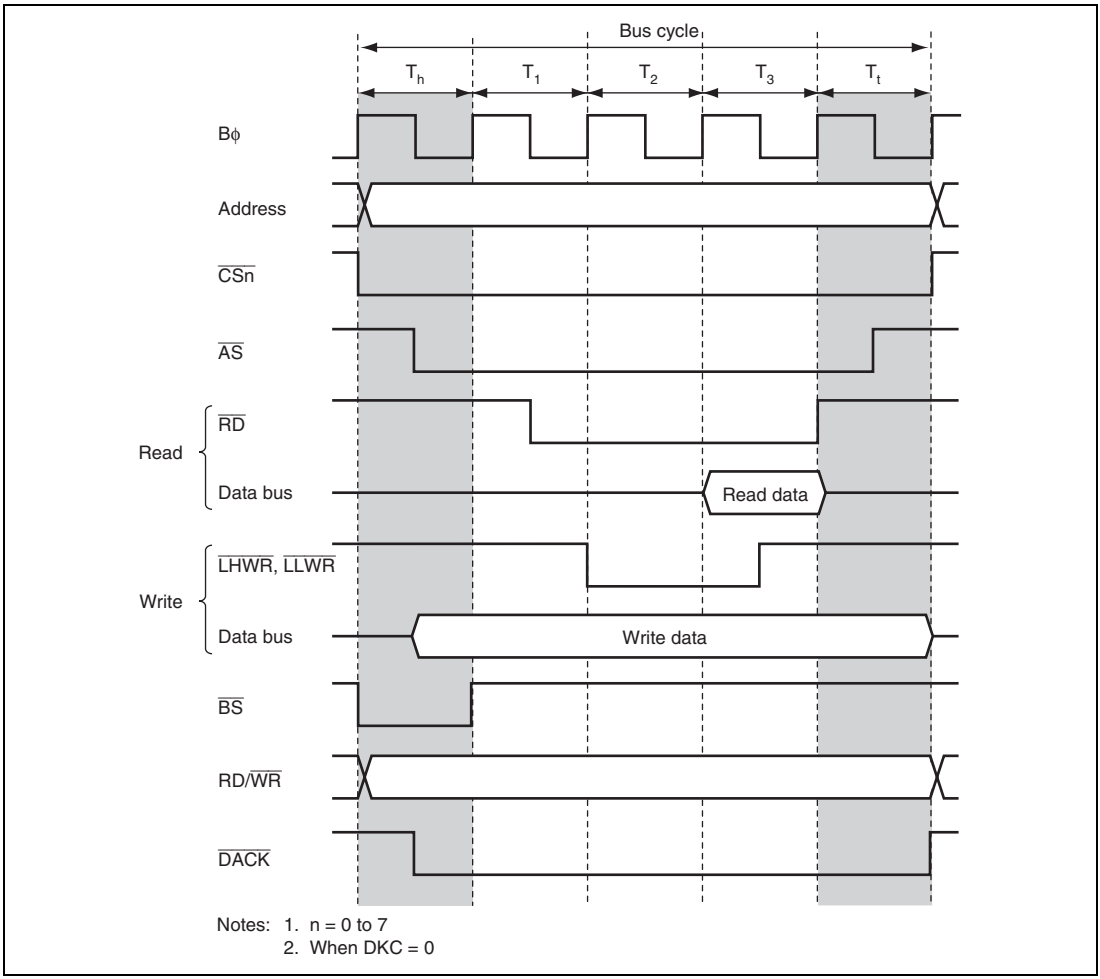


Figure 6.22 Example of Timing when Chip Select Assertion Period is Extended

6.6.7 $\overline{\text{DACK}}$ Signal Output Timing

For DMAC single address transfers, the $\overline{\text{DACK}}$ signal assert timing can be modified by using the DKC bit in BCR1.

Figure 6.23 shows the $\overline{\text{DACK}}$ signal output timing. Setting the DKC bit to 1 asserts the $\overline{\text{DACK}}$ signal a half cycle earlier.

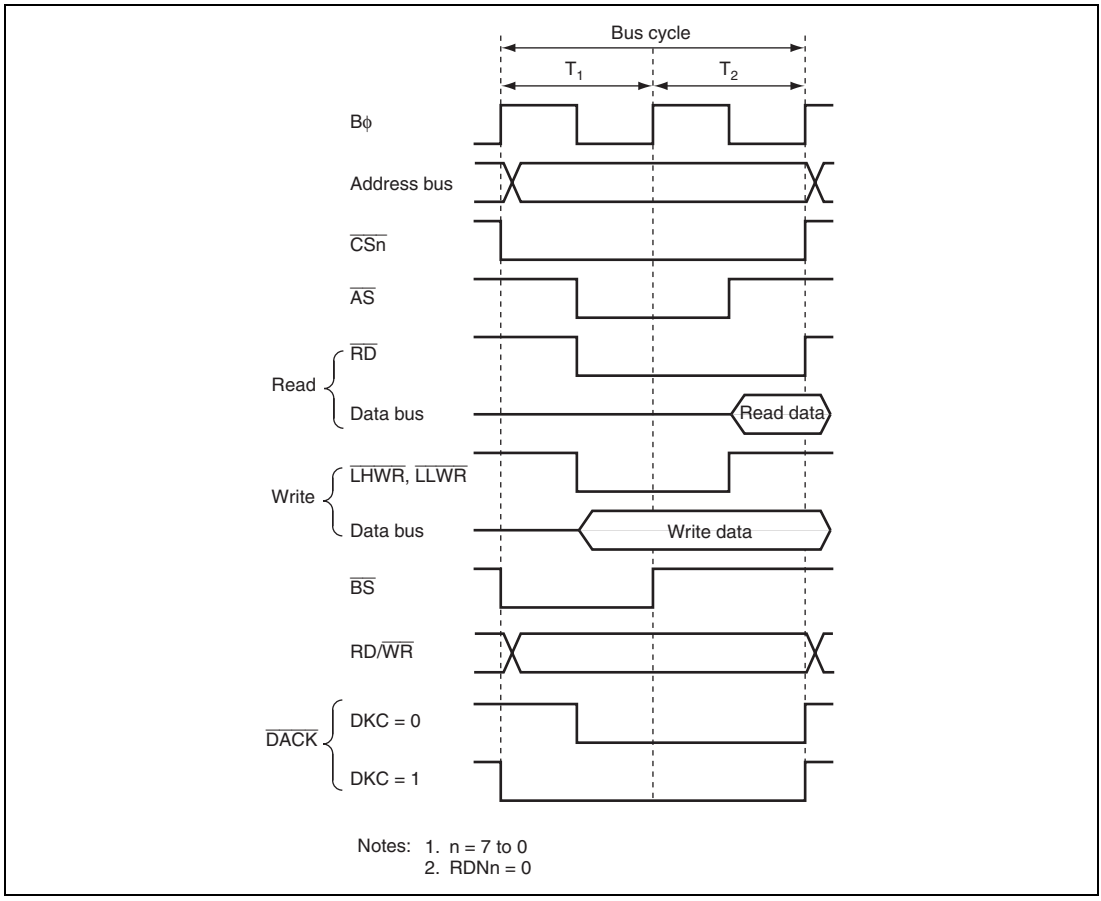


Figure 6.23 $\overline{\text{DACK}}$ Signal Output Timing

6.7 Byte Control SRAM Interface

The byte control SRAM interface is a memory interface for outputting a byte select strobe during a read or a write bus cycle. This interface has 16-bit data input/output pins and can be connected to the SRAM that has the upper byte select and the lower byte select strobes such as $\overline{UB}$ and $\overline{LB}$.

The operation of the byte control SRAM interface is the same as the basic bus interface except that: the byte select strobes ($\overline{LUB}$ and $\overline{LLB}$) are output from the write strobe output pins ($\overline{LHWR}$ and $\overline{LLWR}$), respectively; the read strobe ($\overline{RD}$) negation timing is a half cycle earlier than that in the case where RDNn = 0 in the basic bus interface regardless of the RDNCR settings; and the RD/$\overline{WR}$ signal is used as write enable.

6.7.1 Byte Control SRAM Space Setting

Byte control SRAM interface can be specified for areas 0 to 7. Each area can be specified as byte control SRAM interface by setting bits BCSELn (n = 0 to 7) in SRAMCR. For the area specified as burst ROM interface or address/data multiplexed I/O interface, the SRAMCR setting is invalid and byte control SRAM interface cannot be used.

6.7.2 Data Bus

The bus width of the byte control SRAM space can be specified as 16-bit byte control SRAM space according to bits ABWHn and ABWLn (n = 0 to 7) in ABWCR. The area specified as 8-bit access space cannot be specified as the byte control SRAM space.

For the 16-bit byte control SRAM space, data bus (D15 to D0) is valid.

Access size and data alignment are the same as the basic bus interface. For details, see section 6.5.6, Endian and Data Alignment.

6.7.3 I/O Pins Used for Byte Control SRAM Interface

Table 6.16 shows the pins used for the byte control SRAM interface.

In the byte control SRAM interface, write strobe signals ($\overline{\text{LHWR}}$ and $\overline{\text{LLWR}}$) are output from the byte select strobes. The RD/$\overline{\text{WR}}$ signal is used as a write enable signal.

Table 6.16 I/O Pins for Byte Control SRAM Interface

Pin	When Byte Control SRAM is Specified	Name	I/O	Function
$\overline{\text{AS}}$/$\overline{\text{AH}}$	$\overline{\text{AS}}$	Address strobe	Output	Strobe signal indicating that the address output on the address bus is valid when a basic bus interface space or byte control SRAM space is accessed
$\overline{\text{CSn}}$	$\overline{\text{CSn}}$	Chip select	Output	Strobe signal indicating that area n is selected
$\overline{\text{RD}}$	$\overline{\text{RD}}$	Read strobe	Output	Output enable for the SRAM when the byte control SRAM space is accessed
RD/$\overline{\text{WR}}$	RD/$\overline{\text{WR}}$	Read/write	Output	Write enable signal for the SRAM when the byte control SRAM space is accessed
$\overline{\text{LHWR}}$/$\overline{\text{LUB}}$	$\overline{\text{LUB}}$	Lower-upper byte select	Output	Upper byte select when the 16-bit byte control SRAM space is accessed
$\overline{\text{LLWR}}$/$\overline{\text{LLB}}$	$\overline{\text{LLB}}$	Lower-lower byte select	Output	Lower byte select when the 16-bit byte control SRAM space is accessed
$\overline{\text{WAIT}}$	$\overline{\text{WAIT}}$	Wait	Input	Wait request signal used when an external address space is accessed
A23 to A0	A23 to A0	Address pin	Output	Address output pin
D15 to D0	D15 to D0	Data pin	Input/ output	Data input/output pin

6.7.4 Basic Timing

(1) 2-State Access Space

Figure 6.24 shows the bus timing when the byte control SRAM space is specified as a 2-state access space.

Data buses used for 16-bit access space is the same as those in basic bus interface. No wait cycles can be inserted.

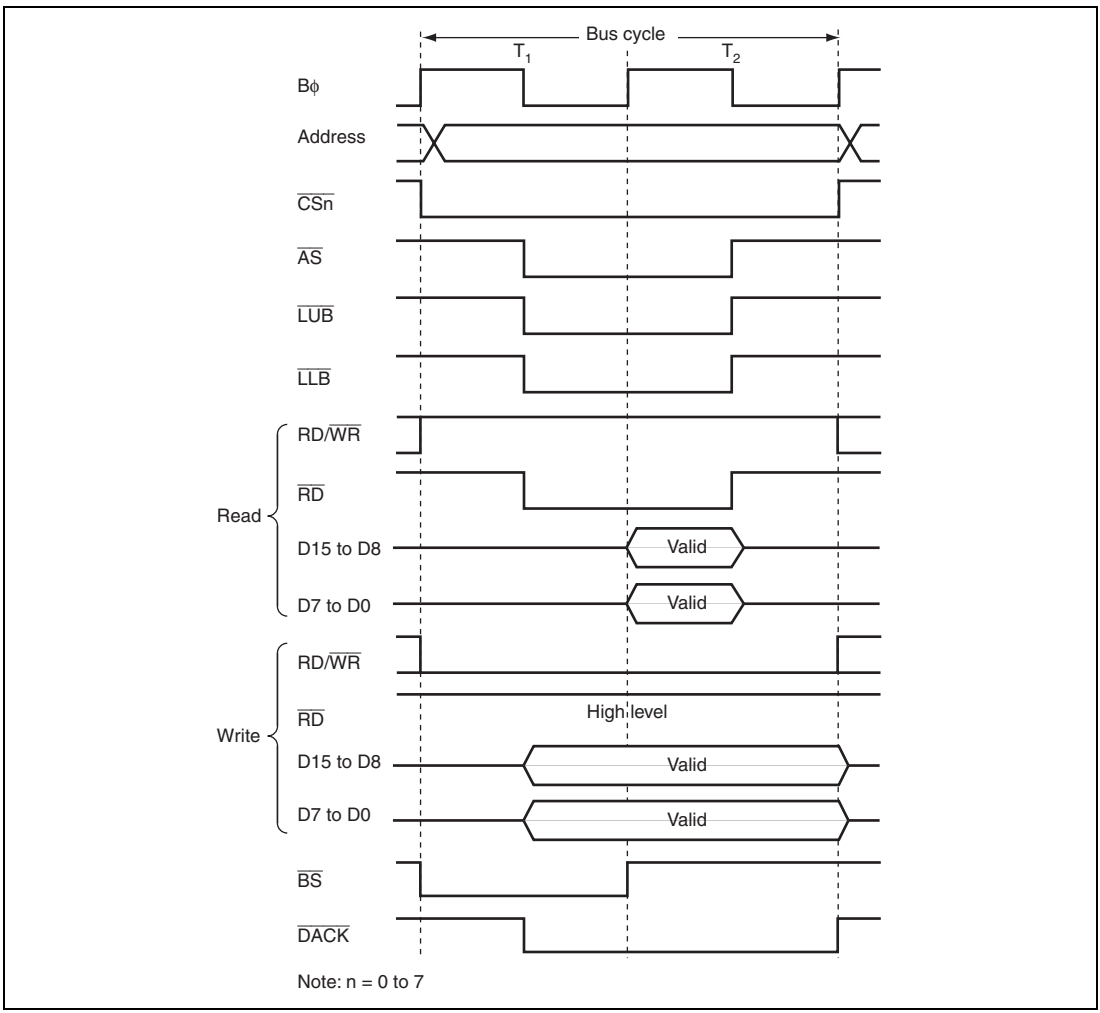


Figure 6.24 16-Bit 2-State Access Space Bus Timing

(2) 3-State Access Space:

Figure 6.25 shows the bus timing when the byte control SRAM space is specified as a 3-state access space.

Data buses used for 16-bit access space is the same as those in the basic bus interface. Wait cycles can be inserted.

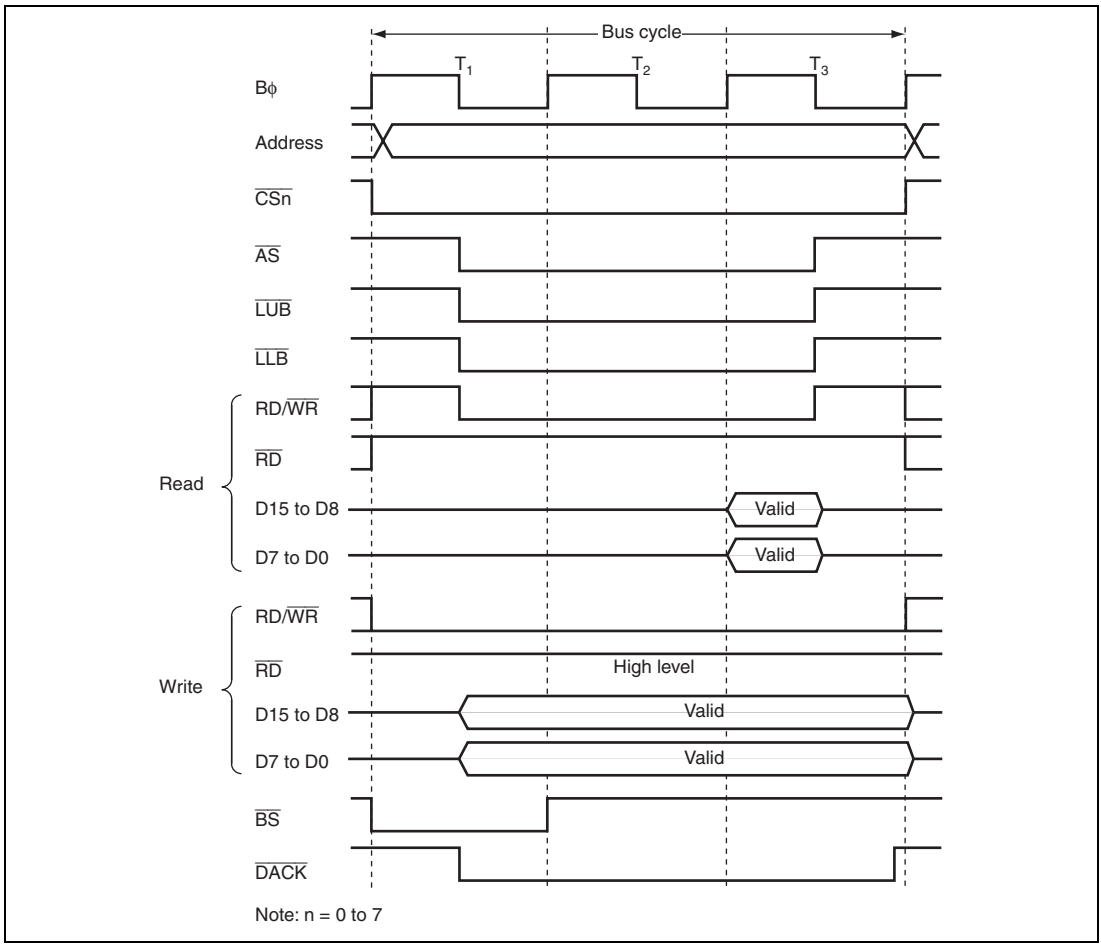


Figure 6.25 16-Bit 3-State Access Space Bus Timing

6.7.5 Wait Control

The bus cycle can be extended for the byte control SRAM interface by inserting wait cycles (Tw) in the same way as the basic bus interface.

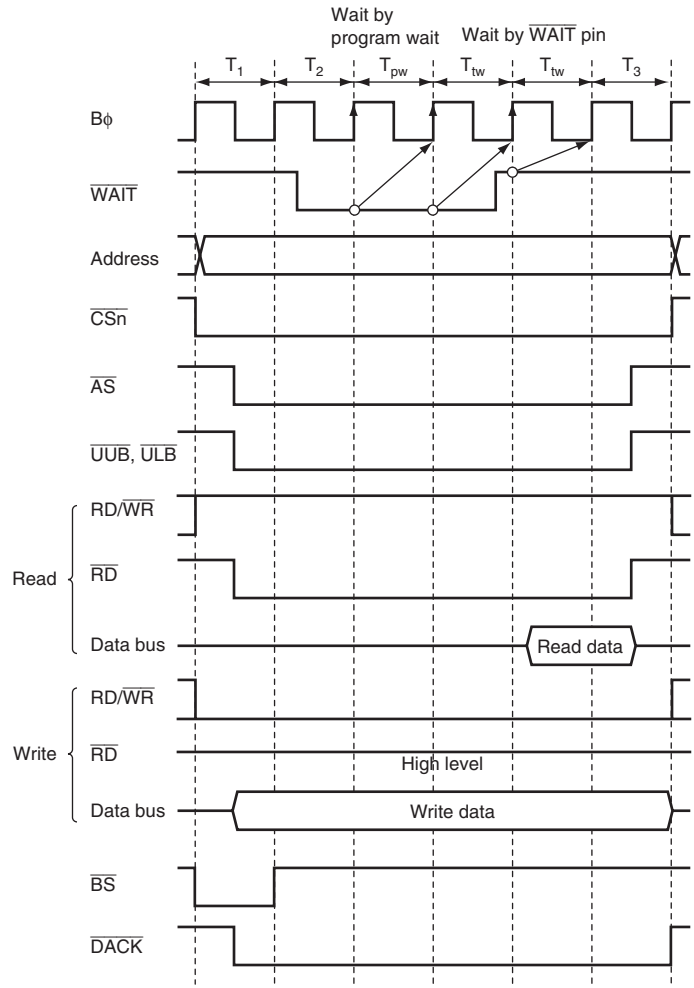
(1) Program Wait Insertion

From 0 to 7 wait cycles can be inserted automatically between T2 cycle and T3 cycle for the 3-state access space in area units, according to the settings in WTCRA and WTCRB.

(2) Pin Wait Insertion

For 3-state access space, when the WAITE bit in BCR1 is set to 1, the corresponding DDR bit is cleared to 0, and the ICR bit is set to 1, wait input by means of the $\overline{\text{WAIT}}$ pin is enabled. For details on DDR and ICR, refer to section 9, I/O Ports.

Figure 6.26 shows an example of wait cycle insertion timing.



Notes:1. Upward arrows indicate the timing of WAIT pin sampling.
2. n = 0 to 7

Figure 6.26 Example of Wait Cycle Insertion Timing

6.7.6 Read Strobe ($\overline{RD}$)

When the byte control SRAM space is specified, the RDNCR setting for the corresponding space is invalid.

The read strobe negation timing is the same timing as when RDNn = 1 in the basic bus interface. Note that the $\overline{RD}$ timing with respect to the $\overline{DACK}$ rising edge becomes different.

6.7.7 Extension of Chip Select ($\overline{CS}$) Assertion Period

In the byte control SRAM interface, the extension cycles can be inserted before and after the bus cycle in the same way as the basic bus interface. For details, refer to section 6.6.6, Extension of Chip Select ($\overline{CS}$) Assertion Period.

6.7.8 $\overline{\text{DACK}}$ Signal Output Timing

For DMAC single address transfers, the $\overline{\text{DACK}}$ signal assert timing can be modified by using the DKC bit in BCR1.

Figure 6.27 shows the $\overline{\text{DACK}}$ signal output timing. Setting the DKC bit to 1 asserts the $\overline{\text{DACK}}$ signal a half cycle earlier.

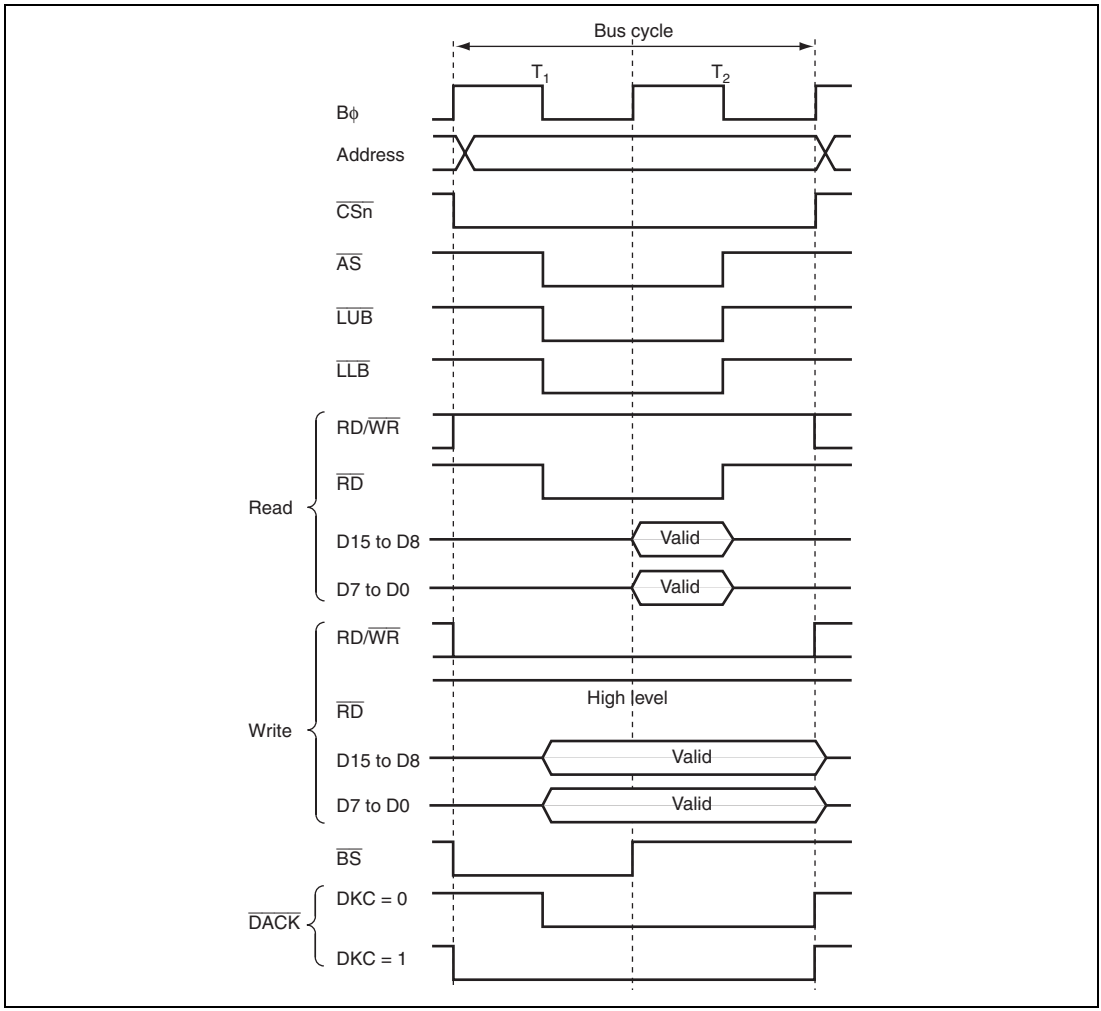


Figure 6.27 $\overline{\text{DACK}}$ Signal Output Timing

6.8 Burst ROM Interface

In this LSI, external address space areas 0 and 1 can be designated as burst ROM space, and burst ROM interfacing performed. The burst ROM interface enables ROM with page access capability to be accessed at high speed.

Areas 1 and 0 can be designated as burst ROM space by means of bits BSRM1 and BSRM0 in BROMCR. Consecutive burst accesses of up to 32 words can be performed, according to the setting of bits BSWDn1 and BSWDn0 (n = 0, 1) in BROMCR. From one to eight cycles can be selected for burst access.

Settings can be made independently for area 0 and area 1.

In the burst ROM interface, the burst access covers only CPU read accesses. Other accesses are performed with the similar method to the basic bus interface.

6.8.1 Burst ROM Space Setting

Burst ROM interface can be specified for areas 0 and 1. Areas 0 and 1 can be specified as burst ROM space by setting bits BSRMn (n = 0, 1) in BROMCR.

6.8.2 Data Bus

The bus width of the burst ROM space can be specified as 8-bit or 16-bit burst ROM interface space according to the ABWHn and ABWLn bits (n = 0, 1) in ABWCR.

For the 8-bit bus width, data bus (D7 to D0) is valid. For the 16-bit bus width, data bus (D15 to D0) is valid.

Access size and data alignment are the same as the basic bus interface. For details, see section 6.5.6, Endian and Data Alignment.

6.8.3 I/O Pins Used for Burst ROM Interface

Table 6.17 shows the pins used for the burst ROM interface.

Table 6.17 I/O Pins Used for Burst ROM Interface

Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started.
Address strobe	$\overline{AS}$	Output	Strobe signal indicating that an address output on the address bus is valid during access
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Read/write	RD/$\overline{WR}$	Output	Signal indicating the data bus input or output direction
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid during write access
Low-low write	$\overline{LLWR}$	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid during write access
Chip select 0, 1	$\overline{CS0}$, $\overline{CS1}$	Output	Strobe signal indicating that the area is selected
Wait	$\overline{WAIT}$	Input	Wait request signal used when an external address space is accessed

6.8.4 Basic Timing

The number of access cycles in the initial cycle (full access) on the burst ROM interface is determined by the basic bus interface settings in ABWCR, ASTCR, WTCRA, WTCRB, and bits CSXHn in CSACR (n = 0 to 7). When area 0 or area 1 designated as burst ROM space is read by the CPU, the settings in RDNCR and bits CSXTn in CSACR (n = 0 to 7) are ignored.

From one to eight cycles can be selected for the burst cycle, according to the settings of bits BSTS02 to BSTS00 and BSTS12 to BSTS10 in BROMCR. Wait cycles cannot be inserted. In addition, 4-word, 8-word, 16-word, or 32-word consecutive burst access can be performed according to the settings of BSTS01, BSTS00, BSTS11, and BSTS10 bits in BROMCR.

The basic access timing for burst ROM space is shown in figures 6.28 and 6.29.

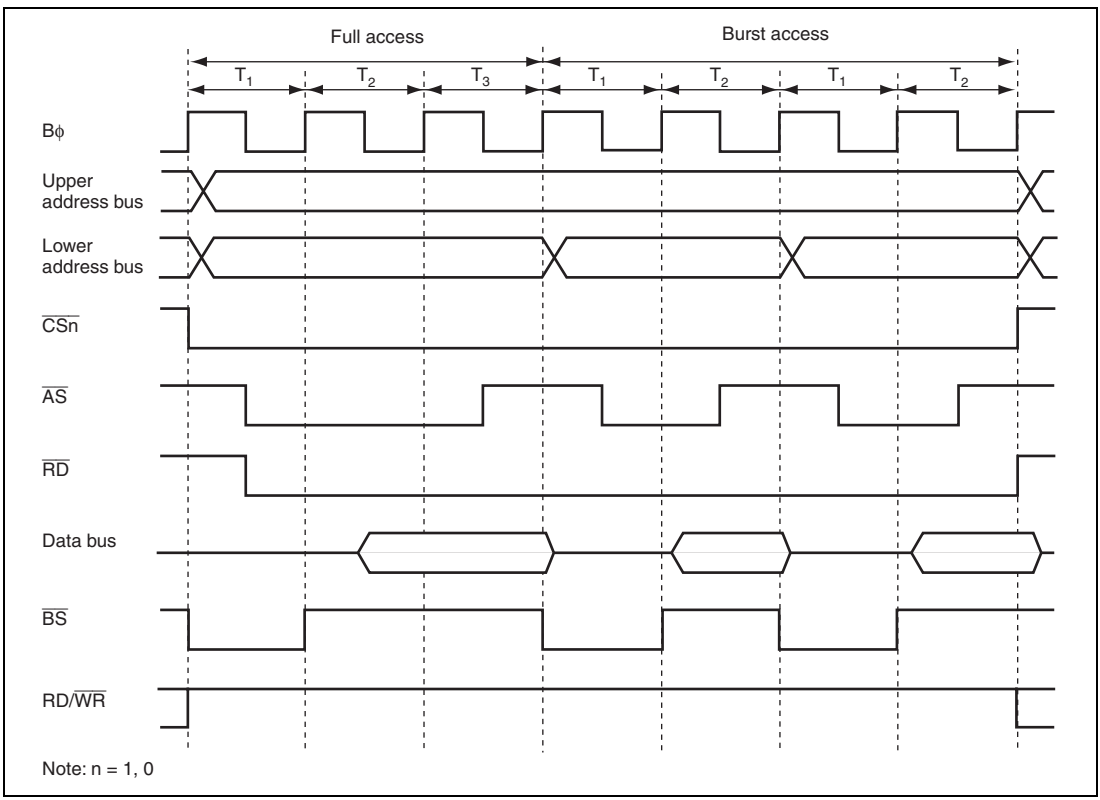


Figure 6.28 Example of Burst ROM Access Timing (ASTn = 1, Two Burst Cycles)

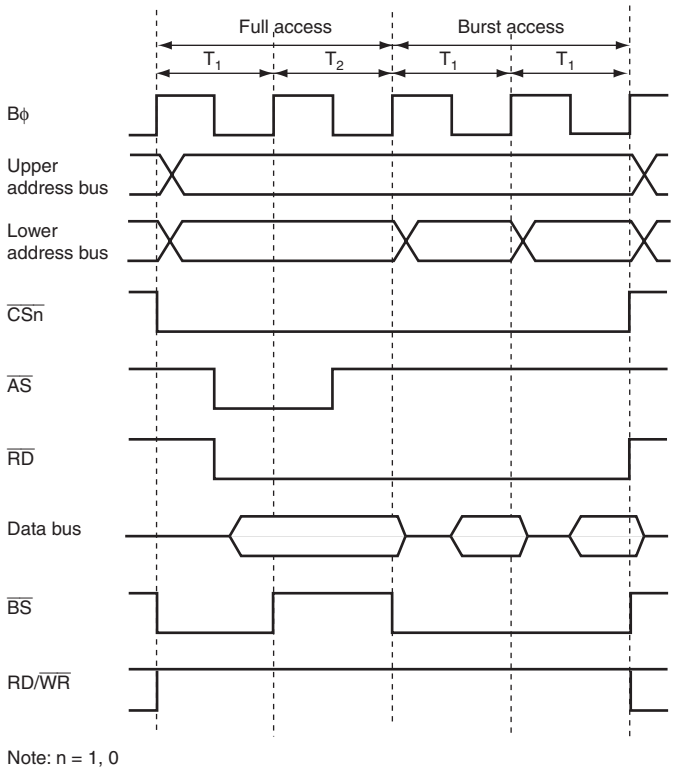


Figure 6.29 Example of Burst ROM Access Timing (ASTn = 0, One Burst Cycle)

6.8.5 Wait Control

As with the basic bus interface, either program wait insertion or pin wait insertion by the $\overline{\text{WAIT}}$ pin can be used in the initial cycle (full access) on the burst ROM interface. See section 6.6.4, Wait Control. Wait cycles cannot be inserted in a burst cycle.

6.8.6 Read Strobe ($\overline{\text{RD}}$) Timing

When the burst ROM space is read by the CPU, the RDNCR setting for the corresponding space is invalid.

The read strobe negation timing is the same timing as when RDNn = 0 in the basic bus interface.

6.8.7 Extension of Chip Select ($\overline{\text{CS}}$) Assertion Period

In the burst ROM interface, the extension cycles can be inserted in the same way as the basic bus interface.

For the burst ROM space, the burst access can be enabled only in read access by the CPU. In this case, the setting of the corresponding CSXTn bit in CSACR is ignored and an extension cycle can be inserted only before the full access cycle. Note that no extension cycle can be inserted before or after the burst access cycles.

In accesses other than read accesses by the CPU, the burst ROM space is equivalent to the basic bus interface space. Accordingly, extension cycles can be inserted before and after the burst access cycles.

6.9 Address/Data Multiplexed I/O Interface

If areas 3 to 7 of external address space are specified as address/data multiplexed I/O space in this LSI, the address/data multiplexed I/O interface can be performed. In the address/data multiplexed I/O interface, peripheral LSIs that require the multiplexed address/data can be connected directly to this LSI.

6.9.1 Address/Data Multiplexed I/O Space Setting

Address/data multiplexed I/O interface can be specified for areas 3 to 7. Each area can be specified as the address/data multiplexed I/O space by setting bits MPXEn (n = 3 to 7) in MPXCR.

6.9.2 Address/Data Multiplex

In the address/data multiplexed I/O space, data bus is multiplexed with address bus. Table 6.18 shows the relationship between the bus width and address output.

Table 6.18 Address/Data Multiplex

Bus Width	Cycle	Data Pins															
		PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
8 bits	Address	-	-	-	-	-	-	-	-	A7	A6	A5	A4	A3	A2	A1	A0
	Data	-	-	-	-	-	-	-	-	D7	D6	D5	D4	D3	D2	D1	D0
16 bits	Address	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
	Data	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

6.9.3 Data Bus

The bus width of the address/data multiplexed I/O space can be specified for either 8-bit access space or 16-bit access space by the ABWHn and ABWLn bits (n = 3 to 7) in ABWCR.

For the 8-bit access space, D7 to D0 are valid for both address and data. For 16-bit access space, D15 to D0 are valid for both address and data. If the address/data multiplexed I/O space is accessed, the corresponding address will be output to the address bus.

For details on access size and data alignment, see section 6.5.6, Endian and Data Alignment.

6.9.4 I/O Pins Used for Address/Data Multiplexed I/O Interface

Table 6.19 shows the pins used for the address/data multiplexed I/O Interface.

Table 6.19 I/O Pins for Address/Data Multiplexed I/O Interface

Pin	When Byte Control SRAM is Specified	Name	I/O	Function
$\overline{CSn}$	$\overline{CSn}$	Chip select	Output	Chip select (n = 3 to 7) when area n is specified as the address/data multiplexed I/O space
$\overline{AS}$/$\overline{AH}$	$\overline{AH}$*	Address hold	Output	Signal to hold an address when the address/data multiplexed I/O space is specified
$\overline{RD}$	$\overline{RD}$	Read strobe	Output	Signal indicating that the address/data multiplexed I/O space is being read
$\overline{LHWR}$/$\overline{LUB}$	$\overline{LHWR}$	Low-high write	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid when the address/data multiplexed I/O space is written
$\overline{LLWR}$/$\overline{LLB}$	$\overline{LLWR}$	Low-low write	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid when the address/data multiplexed I/O space is written
D15 to D0	D15 to D0	Address/data	Input/ output	Address and data multiplexed pins for the address/data multiplexed I/O space. Only D7 to D0 are valid when the 8-bit space is specified. D15 to D0 are valid when the 16-bit space is specified.
A23 to A0	A23 to A0	Address	Output	Address output pin
$\overline{WAIT}$	$\overline{WAIT}$	Wait	Input	Wait request signal used when the external address space is accessed
$\overline{BS}$	$\overline{BS}$	Bus cycle start	Output	Signal to indicate the bus cycle start
RD/$\overline{WR}$	RD/$\overline{WR}$	Read/write	Output	Signal indicating the data bus input or output direction

Note: * The $\overline{AH}$ output is multiplexed with the $\overline{AS}$ output. At the timing that an area is specified as address/data multiplexed I/O, this pin starts to function as the $\overline{AH}$ output meaning that this pin cannot be used as the $\overline{AS}$ output. At this time, when other areas set to the basic bus interface is accessed, this pin does not function as the $\overline{AS}$ output. Until an area is specified as address/data multiplexed I/O, be aware that this pin functions as the $\overline{AS}$ output.

6.9.5 Basic Timing

The bus cycle in the address/data multiplexed I/O interface consists of an address cycle and a data cycle. The data cycle is based on the basic bus interface timing specified by the ABWCR, ASTCR, WTCRA, WTCRB, RDNCR, and CSACR.

Figures 6.30 and 6.31 show the basic access timings.

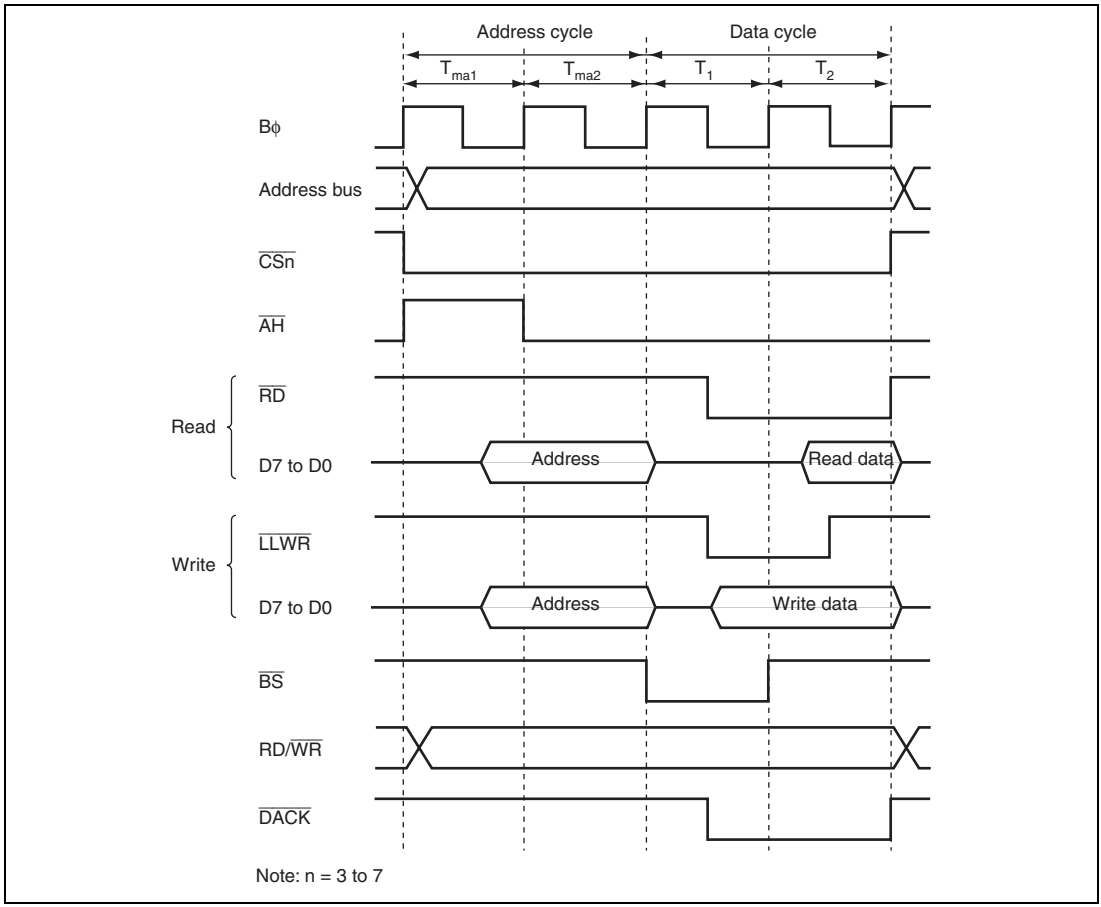


Figure 6.30 8-Bit Access Space Access Timing (ABWHn = 1, ABWLn = 1)

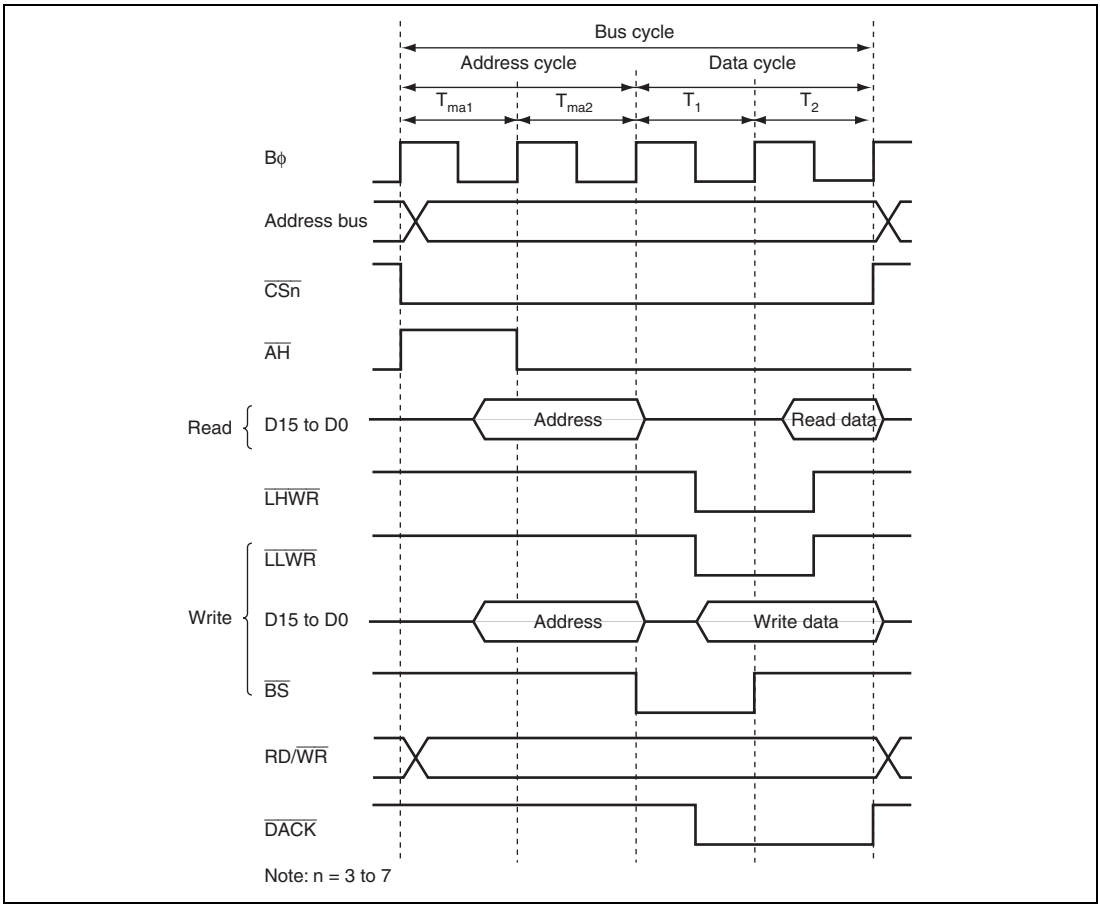


Figure 6.31 16-Bit Access Space Access Timing (ABWHn = 0, ABWLn = 1)

6.9.6 Address Cycle Control

An extension cycle (Tmaw) can be inserted between Tma1 and Tma2 cycles to extend the $\overline{AH}$ signal output period by setting the ADDEX bit in MPXCR. By inserting the Tmaw cycle, the address setup for $\overline{AH}$ and the $\overline{AH}$ minimum pulse width can be assured.

Figure 6.32 shows the access timing when the address cycle is three cycles.

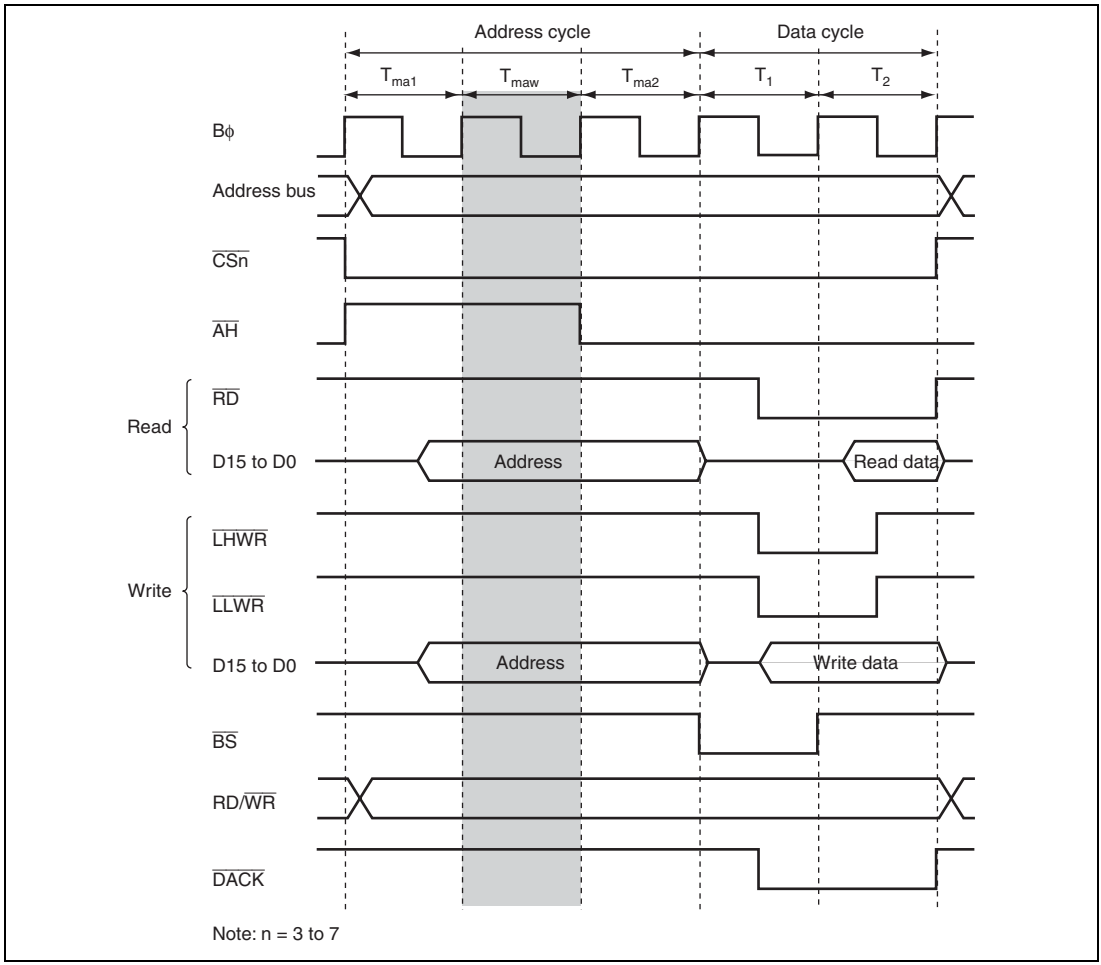


Figure 6.32 Access Timing of 3 Address Cycles (ADDEX = 1)

6.9.7 Wait Control

In the data cycle of the address/data multiplexed I/O interface, program wait insertion and pin wait insertion by the $\overline{\text{WAIT}}$ pin are enabled in the same way as in the basic bus interface. For details, refer to section 6.6.4, Wait Control.

Wait control settings do not affect the address cycles.

6.9.8 Read Strobe ($\overline{\text{RD}}$) Timing

In the address/data multiplexed I/O interface, the read strobe timing of data cycles can be modified in the same way as in basic bus interface. For details, refer to section 6.6.5, Read Strobe ($\overline{\text{RD}}$) Timing.

Figure 6.33 shows an example when the read strobe timing is modified.

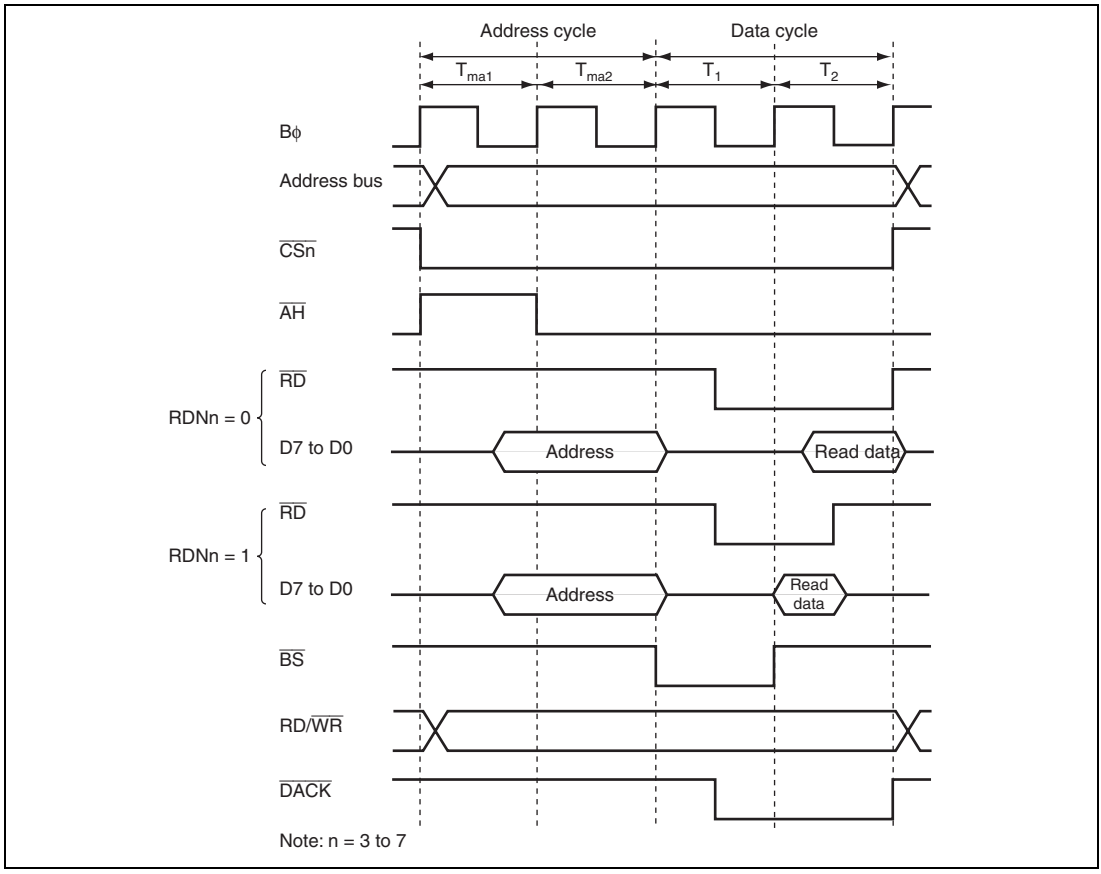


Figure 6.33 Read Strobe Timing

6.9.9 Extension of Chip Select ($\overline{CS}$) Assertion Period

In the address/data multiplexed interface, the extension cycles can be inserted before and after the bus cycle. For details, see section 6.6.6, Extension of Chip Select ($\overline{CS}$) Assertion Period.

Figure 6.34 shows an example of the chip select ($\overline{CS}$) assertion period extension timing.

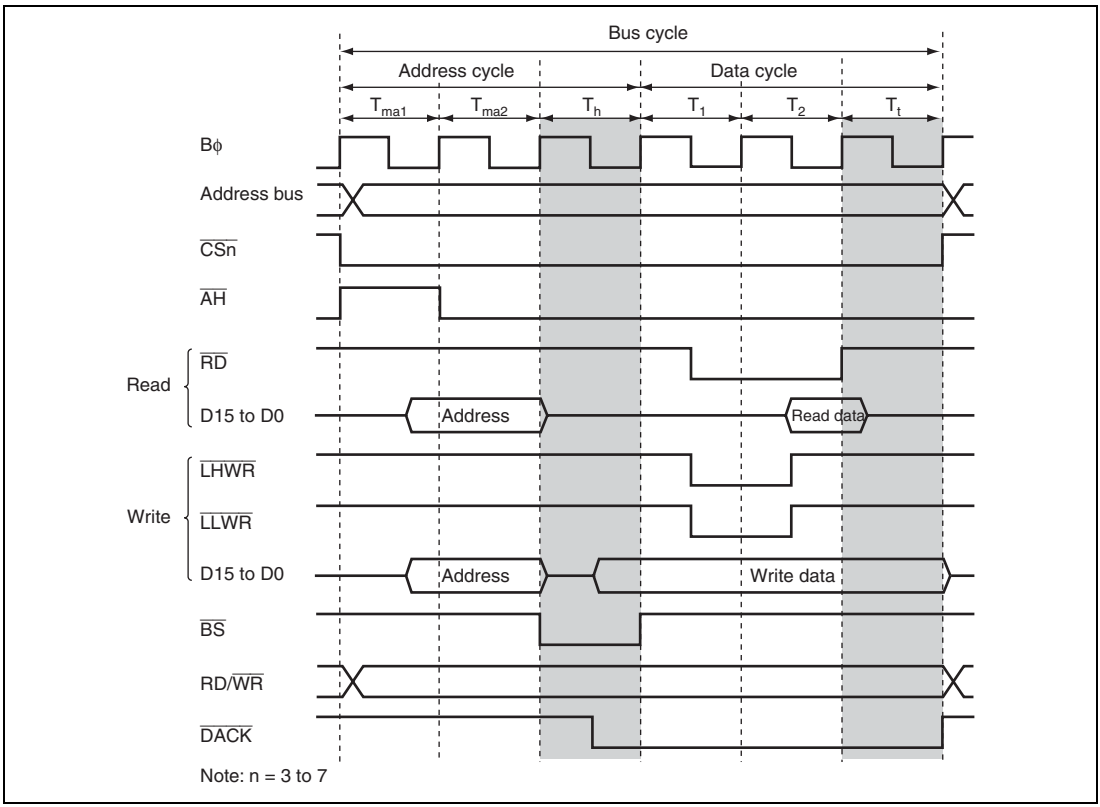
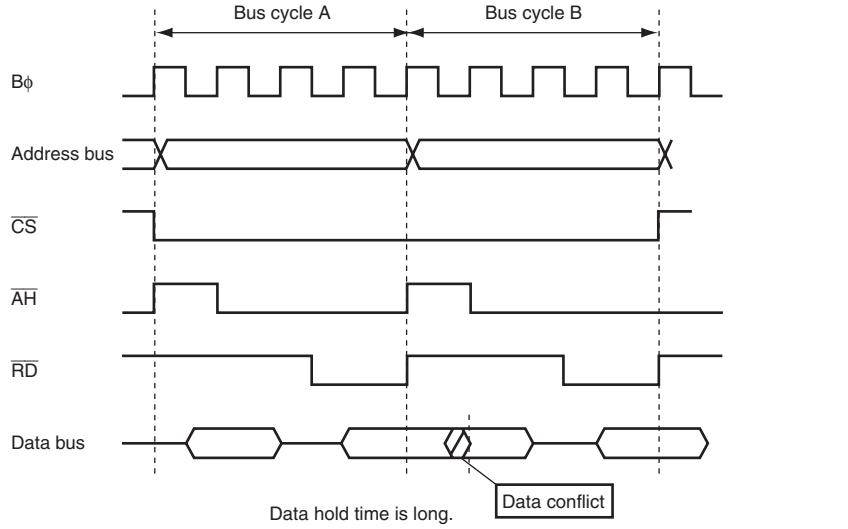


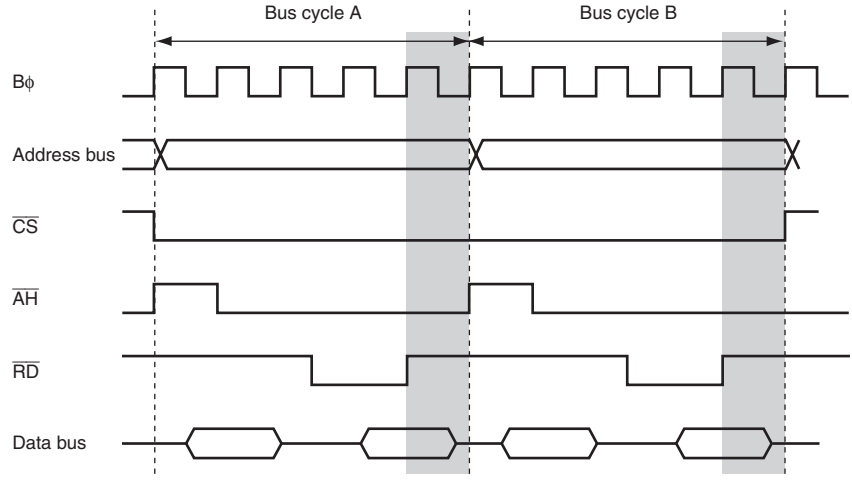
Figure 6.34 Chip Select ($\overline{CS}$) Assertion Period Extension Timing in Data Cycle

When consecutively reading from the same area connected to a peripheral LSI whose data hold time is long, data outputs from the peripheral LSI and this LSI may conflict. Inserting the chip select assertion period extension cycle after the access cycle can avoid the data conflict.

Figure 6.35 shows an example of the operation. In the figure, both bus cycles A and B are read access cycles to the address/data multiplexed I/O space. An example of the data conflict is shown in (a), and an example of avoiding the data conflict by the $\overline{CS}$ assertion period extension cycle in (b).



(a) Without $\overline{CS}$ assertion period extension cycle (CSXTn = 0)



(b) With $\overline{CS}$ assertion period extension cycle (CSXTn = 1)

Figure 6.35 Consecutive Read Accesses to Same Area (Address/Data Multiplexed I/O Space)

6.9.10 $\overline{\text{DACK}}$ Signal Output Timing

For DMAC single address transfers, the $\overline{\text{DACK}}$ signal assert timing can be modified by using the DKC bit in BCR1.

Figure 6.36 shows the $\overline{\text{DACK}}$ signal output timing. Setting the DKC bit to 1 asserts the $\overline{\text{DACK}}$ signal a half cycle earlier.

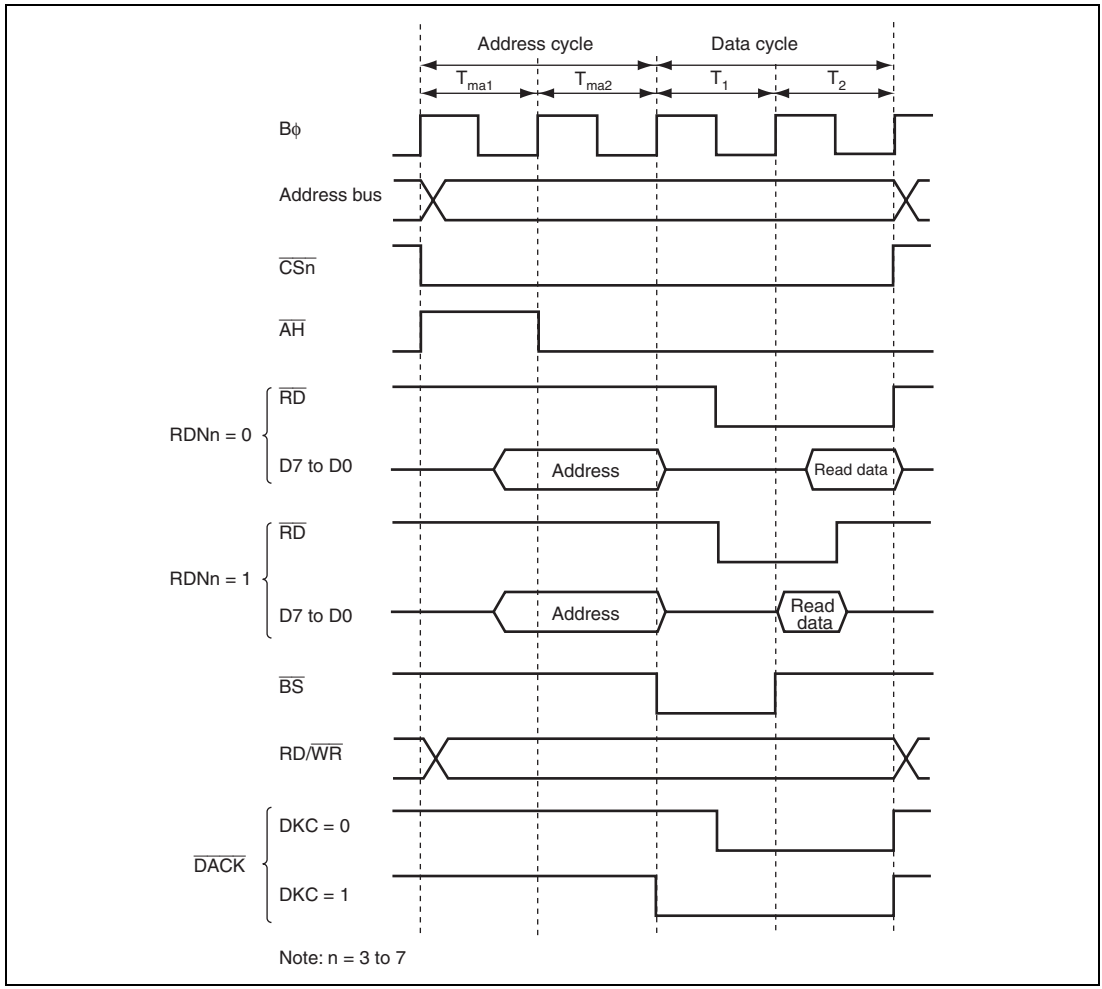


Figure 6.36 $\overline{\text{DACK}}$ Signal Output Timing

6.10 Idle Cycle

In this LSI, idle cycles can be inserted between the consecutive external accesses. By inserting the idle cycle, data conflicts between ROM read cycle whose output floating time is long and an access cycle from/to high-speed memory or I/O interface can be prevented.

6.10.1 Operation

When this LSI consecutively accesses external address space, it can insert an idle cycle between bus cycles in the following four cases. These conditions are determined by the sequence of read and write and previously accessed area.

1. When read cycles of different areas in the external address space occur consecutively
2. When an external write cycle occurs immediately after an external read cycle
3. When an external read cycle occurs immediately after an external write cycle
4. When an external access occurs immediately after a DMAC single address transfer (write cycle)

Up to four idle cycles can be inserted under the conditions shown above. The number of idle cycles to be inserted should be specified to prevent data conflicts between the output data from a previously accessed device and data from a subsequently accessed device.

Under conditions 1 and 2, which are the conditions to insert idle cycles after read, the number of idle cycles can be selected from setting A specified by bits IDLCA1 and IDLCA0 in IDLCR or setting B specified by bits IDLCB1 and IDLCB0 in IDLCR: Setting A can be selected from one to four cycles, and setting B can be selected from one or two to four cycles. Setting A or B can be specified for each area by setting bits IDLSEL7 to IDLSEL0 in IDLCR. Note that bits IDLSEL7 to IDLSEL0 correspond to the previously accessed area of the consecutive accesses.

The number of idle cycles to be inserted under conditions 3 and 4, which are conditions to insert idle cycles after write, can be determined by setting A as described above.

After the reset release, IDLCR is initialized to four idle cycle insertion under all conditions 1 to 4 shown above.

Table 6.20 shows the correspondence between conditions 1 to 4 and number of idle cycles to be inserted for each area. Table 6.21 shows the correspondence between the number of idle cycles to be inserted specified by settings A and B, and number of cycles to be inserted.

Table 6.20 Number of Idle Cycle Insertion Selection in Each Area

	Bit Settings										
	IDLSn		IDLSELn	Area for Previous Access							
Insertion Condition	n	Setting	n = 0 to 7	0	1	2	3	4	5	6	7
Consecutive reads in different areas	1	0	—	Invalid							
		1	0	A	A	A	A	A	A	A	A
			1	B	B	B	B	B	B	B	B
Write after read	0	0	—	Invalid							
		1	0	A	A	A	A	A	A	A	A
			1	B	B	B	B	B	B	B	B
Read after write	2	0	—	Invalid							
		1		A							
External access after single address transfer	3	0	—	Invalid							
		1		A							

[Legend]

A: Number of idle cycle insertion A is selected.

B: Number of idle cycle insertion B is selected.

Invalid: No idle cycle is inserted for the corresponding condition.

Table 6.21 Number of Idle Cycle Insertions

Bit Settings				
A		B		
IDLCA1	IDLCA0	IDLCB1	IDLCB0	Number of Cycles
—	—	0	0	0
0	0	—	—	1
0	1	0	1	2
1	0	1	0	3
1	1	1	1	4

(1) Consecutive Reads in Different Areas

If consecutive reads in different areas occur while bit IDLS1 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 when bit IDLSELn in IDLCR is cleared to 0, or bits IDLCB1 and IDLCB0 when bit IDLSELn is set to 1 are inserted at the start of the second read cycle (n = 0 to 7).

Figure 6.37 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a read cycle for SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data conflict is prevented.

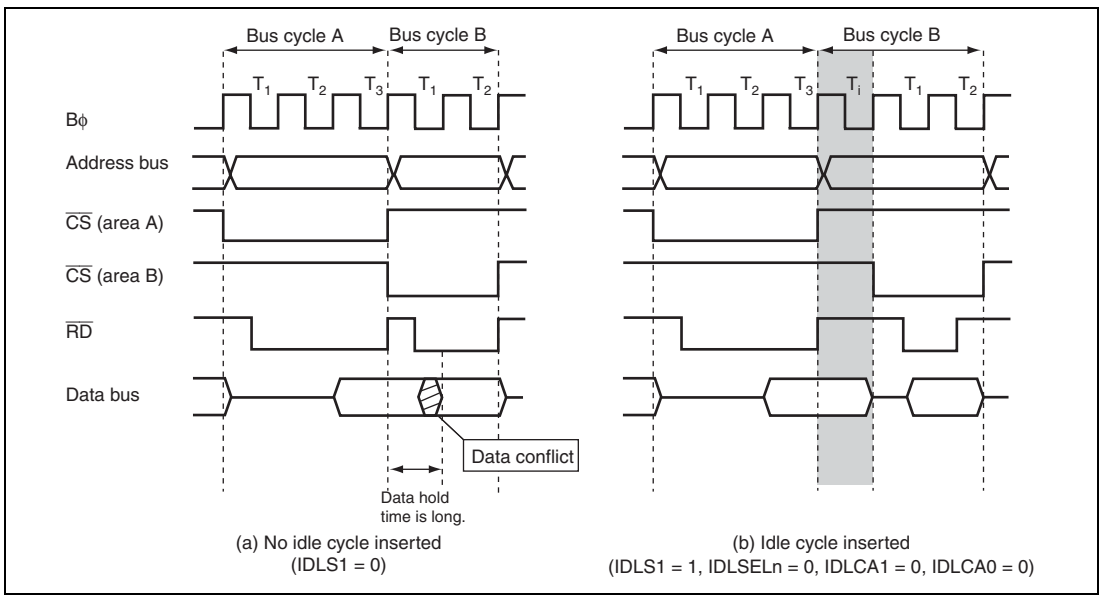


Figure 6.37 Example of Idle Cycle Operation (Consecutive Reads in Different Areas)

(2) Write after Read

If an external write occurs after an external read while bit IDLS0 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 when bit IDLSELn in IDLCR is cleared to 0 when IDLSELn = 0, or bits IDLCB1 and IDLCB0 when IDLSELn is set to 1 are inserted at the start of the write cycle (n = 0 to 7).

Figure 6.38 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data conflict is prevented.

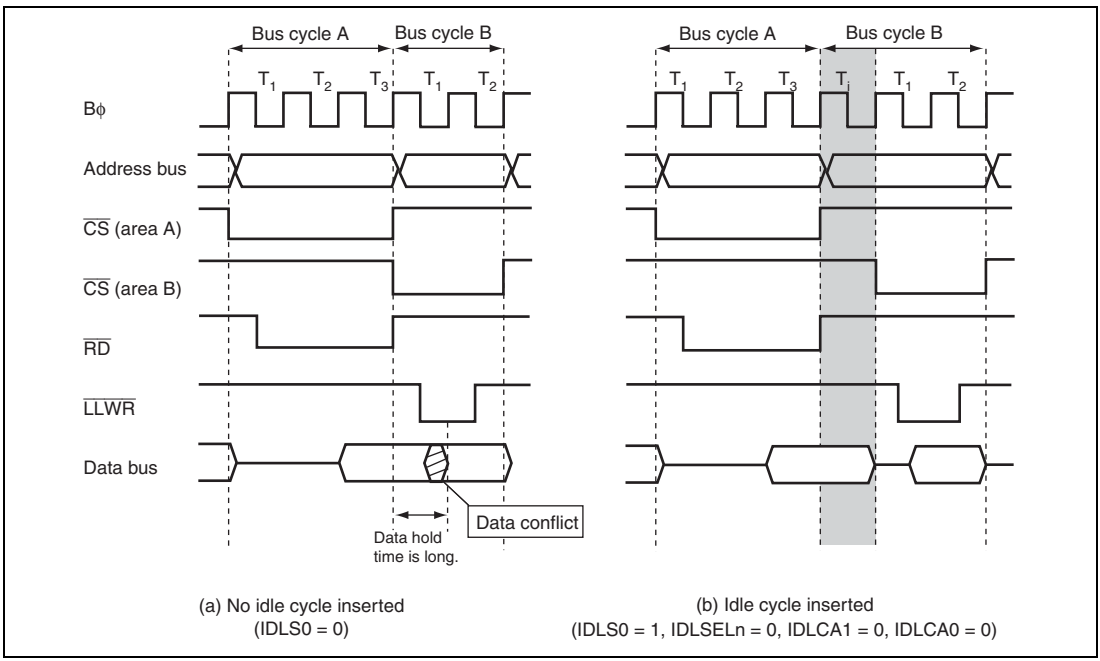


Figure 6.38 Example of Idle Cycle Operation (Write after Read)

(3) Read after Write

If an external read occurs after an external write while bit IDLS2 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 are inserted at the start of the read cycle (n = 0 to 7).

Figure 6.39 shows an example of the operation in this case. In this example, bus cycle A is a CPU write cycle and bus cycle B is a read cycle from the SRAM. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the CPU write data and read data from an SRAM device. In (b), an idle cycle is inserted, and a data conflict is prevented.

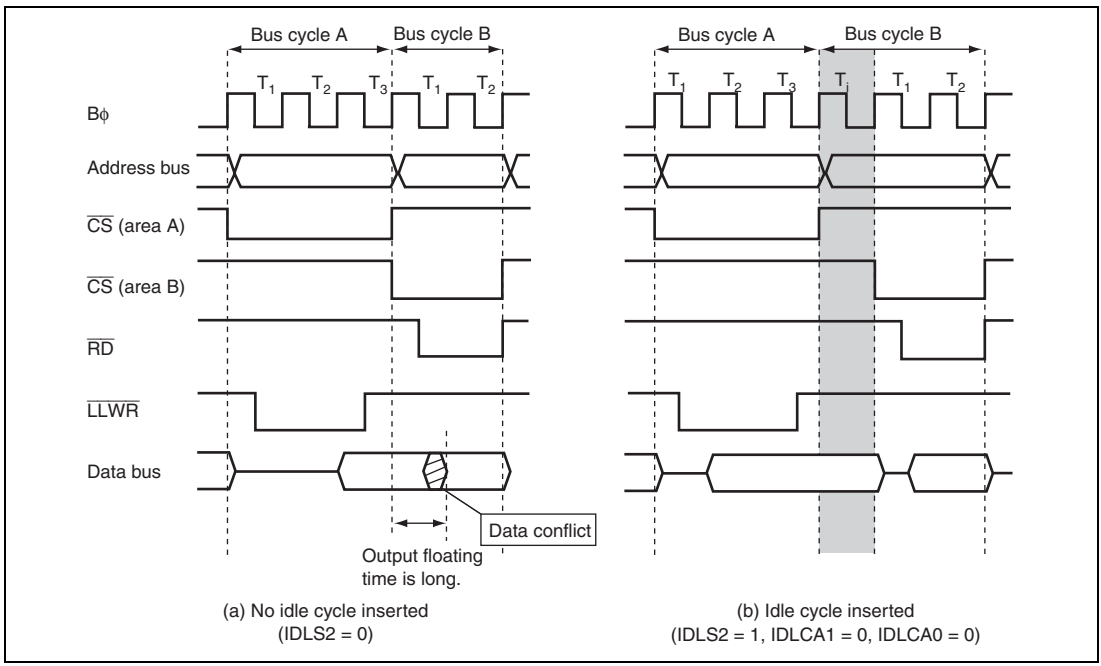


Figure 6.39 Example of Idle Cycle Operation (Read after Write)

(4) External Access after Single Address Transfer Write

If an external access occurs after a single address transfer write while bit IDLS3 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 are inserted at the start of the external access (n = 0 to 7).

Figure 6.40 shows an example of the operation in this case. In this example, bus cycle A is a single address transfer (write cycle) and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the external device write data and this LSI write data. In (b), an idle cycle is inserted, and a data conflict is prevented.

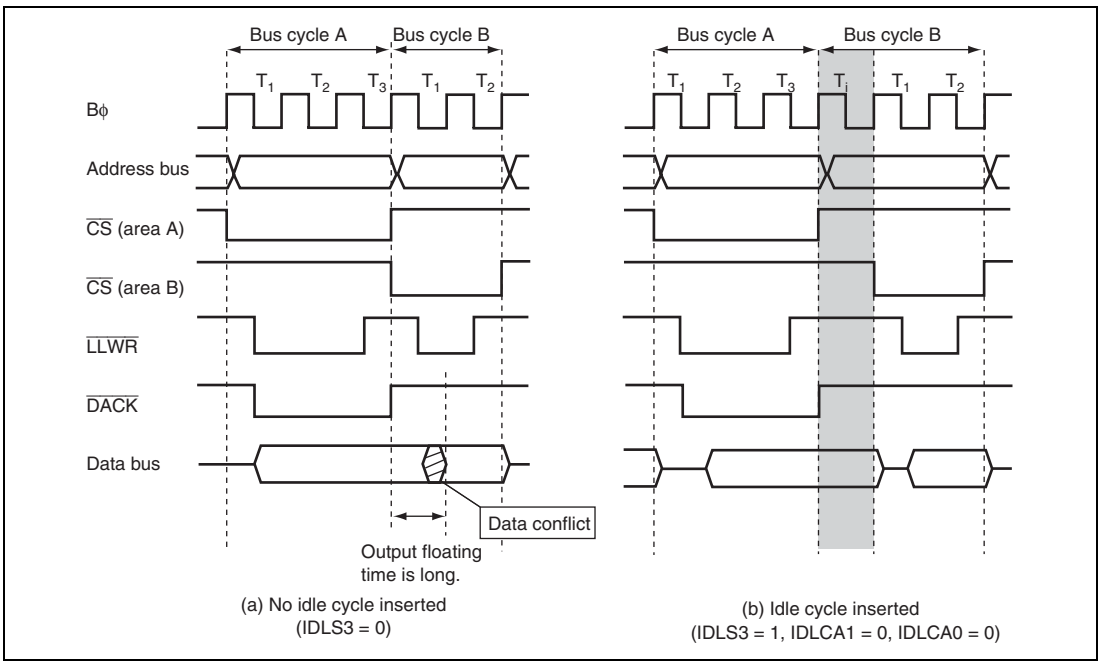


Figure 6.40 Example of Idle Cycle Operation (Write after Single Address Transfer Write)

(5) External NOP Cycles and Idle Cycles

A cycle in which an external space is not accessed due to internal operations is called an external NOP cycle. Even when an external NOP cycle occurs between consecutive external bus cycles, an idle cycle can be inserted. In this case, the number of external NOP cycles is included in the number of idle cycles to be inserted.

Figure 6.41 shows an example of external NOP and idle cycle insertion.

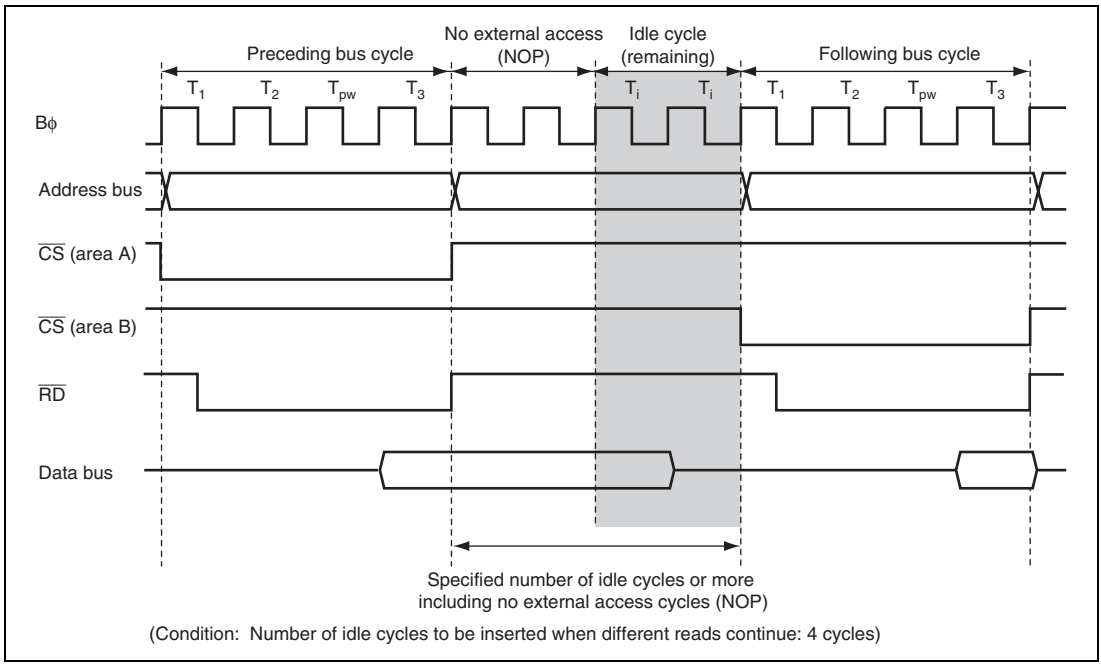


Figure 6.41 Idle Cycle Insertion Example

(6) Relationship between Chip Select ($\overline{CS}$) Signal and Read ($\overline{RD}$) Signal

Depending on the system's load conditions, the $\overline{RD}$ signal may lag behind the $\overline{CS}$ signal. An example is shown in figure 6.44. In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the $\overline{RD}$ signal in bus cycle A and the $\overline{CS}$ signal in bus cycle B. Setting idle cycle insertion, as in (b), however, will prevent any overlap between the $\overline{RD}$ and $\overline{CS}$ signals. In the initial state after reset release, idle cycle indicated in (b) is set.

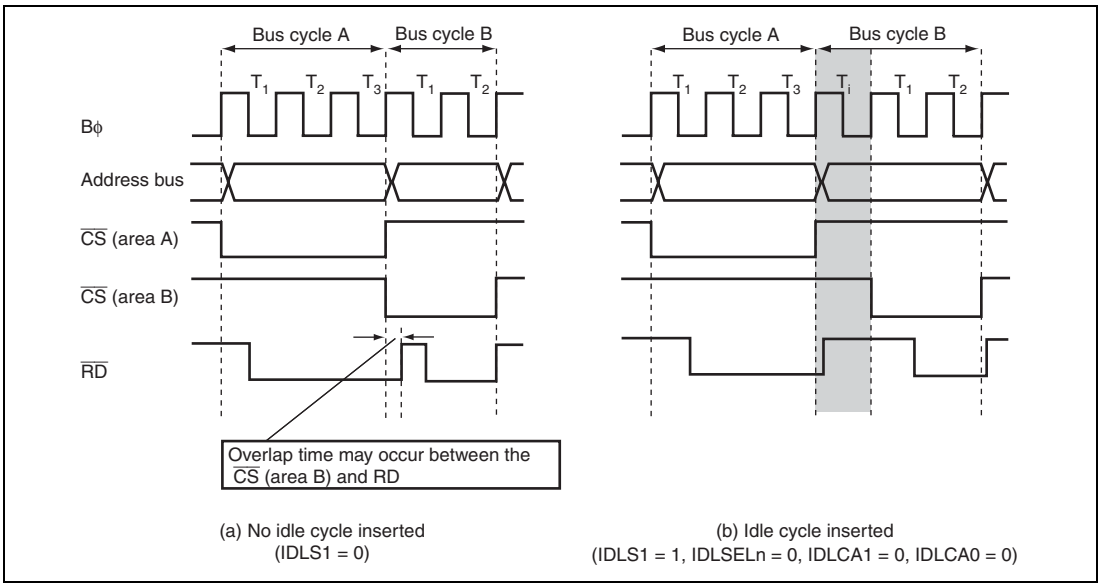


Figure 6.42 Relationship between Chip Select ($\overline{CS}$) and Read ($\overline{RD}$)

Table 6.22 Idle Cycles in Mixed Accesses to Normal Space

Previous Access	Next Access	IDLS 3	IDLS 2	IDLS 1	IDLS 0	IDLSEL 7 to 0	IDLCA 1	IDLCA 0	IDLCB 1	IDLCB 0	Idle Cycle
Normal space read	Normal space read	—	—	0	—	—	—	—	—	—	Disabled
		—	—	1	—	0	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
						1	—	—	0	0	0 cycle inserted
									0	1	2 cycle inserted
									1	0	3 cycles inserted
									1	1	4 cycles inserted
Normal space read	Normal space write	—	—	—	0	—	—	—	—	—	Disabled
		—	—	—	1	0	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
						1	—	—	0	0	0 cycle inserted
									0	1	2 cycle inserted
									1	0	3 cycles inserted
									1	1	4 cycles inserted
Normal space write	Normal space read	—	0	—	—	—	—	—	—	—	Disabled
		—	1	—	—	—	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
Single address transfer write	Normal space read	0	—	—	—	—	—	—	—	—	Disabled
		1	—	—	—	—	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted

6.10.2 Pin States in Idle Cycle

Table 6.23 shows the pin states in an idle cycle.

Table 6.23 Pin States in Idle Cycle

Pins	Pin State
A23 to A0	Contents of following bus cycle
D15 to D0	High impedance
$\overline{\text{CSn}}$ (n = 7 to 0)	High
$\overline{\text{AS}}$	High
$\overline{\text{RD}}$	High
$\overline{\text{BS}}$	High
RD/$\overline{\text{WR}}$	High
$\overline{\text{AH}}$	low
$\overline{\text{LHWR}}$, $\overline{\text{LLWR}}$	High
$\overline{\text{DACKn}}$ (n = 3 to 0)	High

6.11 Bus Release

This LSI can release the external bus in response to a bus request from an external device. In the external bus released state, internal bus masters continue to operate as long as there is no external access.

In addition, in the external bus released state, the $\overline{\text{BREQO}}$ signal can be driven low to output a bus request externally.

6.11.1 Operation

In external extended mode, when the BRLE bit in BCR1 is set to 1 and the ICR bits for the corresponding pin are set to 1, the bus can be released to the external. Driving the $\overline{\text{BREQ}}$ pin low issues an external bus request to this LSI. When the $\overline{\text{BREQ}}$ pin is sampled, at the prescribed timing, the $\overline{\text{BACK}}$ pin is driven low, and the address bus, data bus, and bus control signals are placed in the high-impedance state, establishing the external bus released state. For details on DDR and ICR, see section 9, I/O Ports.

In the external bus released state, the CPU, DTC, and DMAC can access the internal space using the internal bus. When the CPU, DTC, or DMAC attempts to access the external address space, it temporarily defers initiation of the bus cycle, and waits for the bus request from the external bus master to be canceled.

If the BREQOE bit in BCR1 is set to 1, the $\overline{\text{BREQO}}$ pin can be driven low when any of the following requests are issued, to request cancellation of the bus request externally.

- When the CPU, DTC, or DMAC attempts to access the external address space
- When a SLEEP instruction is executed to place the chip in software standby mode or all-module-clock-stop mode
- When SCKCR is written to for setting the clock frequency

If an external bus release request and external access occur simultaneously, the priority is as follows:

(High) External bus release > External access by CPU, DTC, or DMAC (Low)

6.11.2 Pin States in External Bus Released State

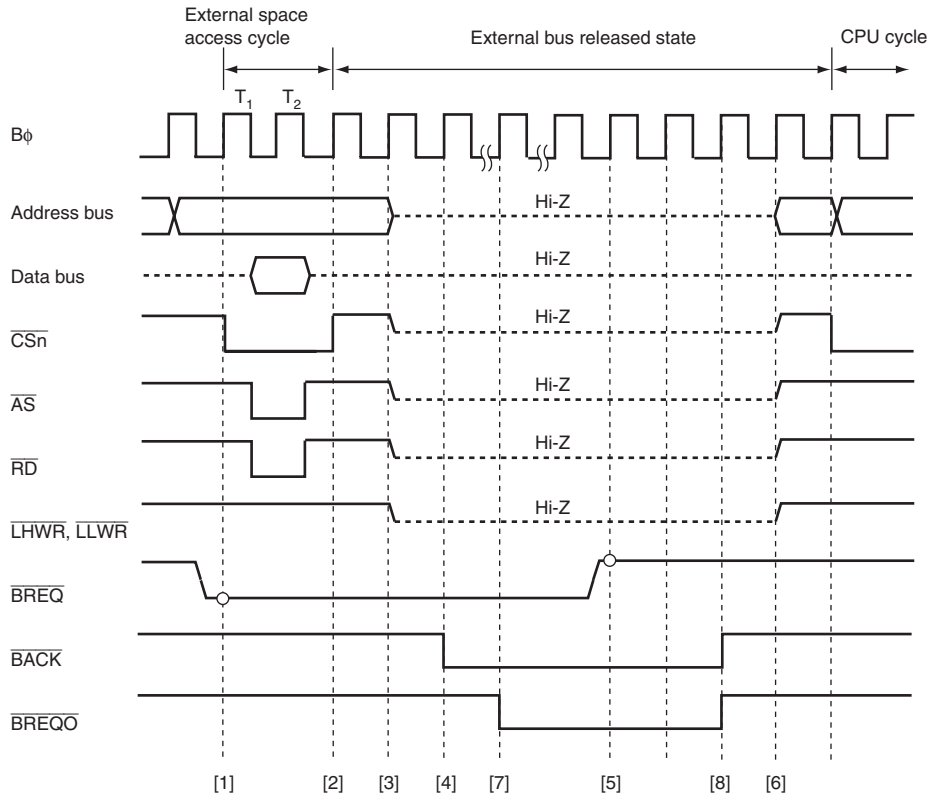
Table 6.24 shows pin states in the external bus released state.

Table 6.24 Pin States in Bus Released State

Pins	Pin State
A23 to A0	High impedance
D15 to D0	High impedance
$\overline{BS}$	High impedance
$\overline{CSn}$ (n = 7 to 0)	High impedance
$\overline{AS}$	High impedance
$\overline{AH}$	High impedance
RD/$\overline{WR}$	High impedance
$\overline{RD}$	High impedance
$\overline{LUB}$, $\overline{LLB}$	High impedance
$\overline{LHWR}$, $\overline{LLWR}$	High impedance
$\overline{DACKn}$ (n = 3 to 0)	High level

6.11.3 Transition Timing

Figure 6.43 shows the timing for transition to the bus released state.



[1] A low level of the $\overline{\text{BREQ}}$ signal is sampled at the rising edge of the Bϕ signal.
[2] The bus control signals are driven high at the end of the external space access cycle. It takes two cycles or more after the low level of the $\overline{\text{BREQ}}$ signal is sampled.
[3] The $\overline{\text{BACK}}$ signal is driven low, releasing bus to the external bus master.
[4] The $\overline{\text{BREQ}}$ signal state sampling is continued in the external bus released state.
[5] A high level of the $\overline{\text{BREQ}}$ signal is sampled.
[6] The external bus released cycles are ended one cycle after the $\overline{\text{BREQ}}$ signal is driven high.
[7] When the external space is accessed by an internal bus master during external bus released while the BREQOE bit is set to 1, the $\overline{\text{BREQO}}$ signal goes low.
[8] Normally the $\overline{\text{BREQO}}$ signal goes high at the rising edge of the $\overline{\text{BACK}}$ signal.

Figure 6.43 Bus Released State Transition Timing

6.12 Internal Bus

6.12.1 Access to Internal Address Space

The internal address spaces of this LSI are the on-chip ROM space, on-chip RAM space, and register space for the on-chip peripheral modules. The number of cycles necessary for access differs according the space.

Table 6.25 shows the number of access cycles for each on-chip memory space.

Table 6.25 Number of Access Cycles for On-Chip Memory Spaces

Access Space	Access	Number of Access Cycles
On-chip ROM space	Read	One Iϕ cycle
	Write	Six Iϕ cycles
On-chip RAM space	Read	One Iϕ cycle
	Write	One Iϕ cycle

In access to the registers for on-chip peripheral modules, the number of access cycles differs according to the register to be accessed. When the dividing ratio of the operating clock of a bus master and that of a peripheral module is 1 : n, synchronization cycles using a clock divided by 0 to n-1 are inserted for register access in the same way as for external bus clock division.

Table 6.26 lists the number of access cycles for registers of on-chip peripheral modules.

Table 6.26 Number of Access Cycles for Registers of On-Chip Peripheral Modules

	Number of Cycles		
Module to be Accessed	Read	Write	Write Data Buffer Function
DMAC registers	2Iϕ		Disabled
MCU operating mode, clock pulse generator, power-down control registers, interrupt controller, bus controller, and DTC registers	2Iϕ	3Iϕ	Disabled
I/O port PFCR registers and WDT registers	2Pϕ	3Pϕ	Disabled
I/O port registers other than PFCR, TPU, PPG, TMR, SCI, A/D, and D/A registers	2Pϕ		Enabled

6.13 Write Data Buffer Function

6.13.1 Write Data Buffer Function for External Data Bus

This LSI has a write data buffer function for the external data bus. Using the write data buffer function enables internal accesses in parallel with external writes or DMAC single address transfers. The write data buffer function is made available by setting the WDBE bit to 1 in BCR1.

Figure 6.44 shows an example of the timing when the write data buffer function is used. When this function is used, if an external address space write or a DMAC single address transfer continues for two cycles or longer, and there is an internal access next, an external write only is executed in the first two cycles. However, from the next cycle onward, internal accesses (on-chip memory or internal I/O register read/write) and the external address space write rather than waiting until it ends are executed in parallel.

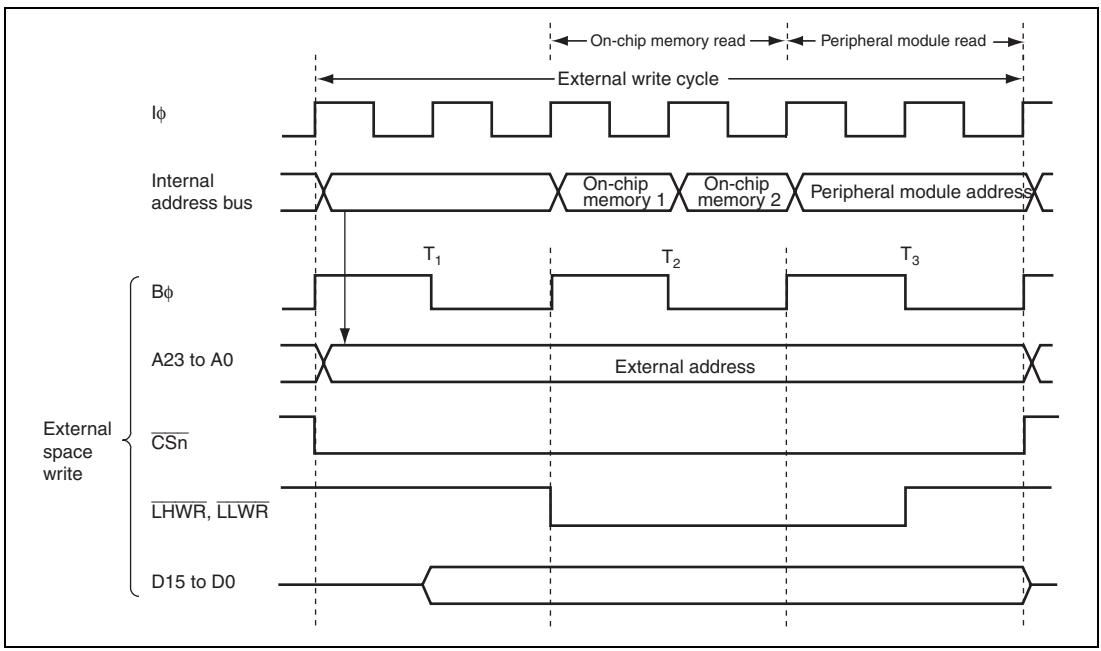


Figure 6.44 Example of Timing when Write Data Buffer Function is Used

6.13.2 Write Data Buffer Function for Peripheral Modules

This LSI has a write data buffer function for the peripheral module access. Using the write data buffer function enables peripheral module writes and on-chip memory or external access to be executed in parallel. The write data buffer function is made available by setting the PWDBE bit in BCR2 to 1. For details on the on-chip peripheral module registers, see table 6.26 in section 6.12, Internal Bus.

Figure 6.45 shows an example of the timing when the write data buffer function is used. When this function is used, if an internal I/O register write continues for two cycles or longer and then there is an on-chip RAM, an on-chip ROM, or an external access, internal I/O register write only is performed in the first two cycles. However, from the next cycle onward an internal memory or an external access and internal I/O register write are executed in parallel rather than waiting until it ends.

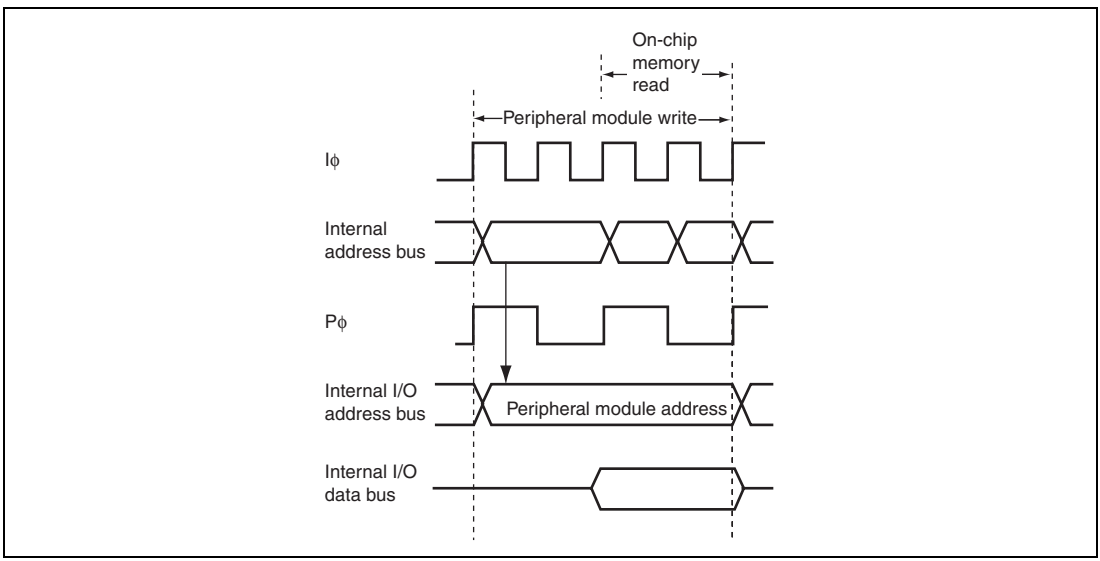


Figure 6.45 Example of Timing when Peripheral Module Write Data Buffer Function is Used

6.14 Bus Arbitration

This LSI has bus arbiters that arbitrate bus mastership operations (bus arbitration). This LSI incorporates internal access and external access bus arbiters that can be used and controlled independently. The internal bus arbiter handles the CPU, DTC, and DMAC accesses. The external bus arbiter handles the external access by the CPU, DTC, and DMAC and external bus release request (external bus master).

The bus arbiters determine priorities at the prescribed timing, and permit use of the bus by means of the bus request acknowledge signal.

6.14.1 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The priority of the internal bus arbitration:

(High) DMAC > DTC > CPU (Low)

The priority of the external bus arbitration:

(High) External bus release request > External access by the CPU, DTC, and DMAC (Low)

If the DMAC or DTC accesses continue, the CPU can be given priority over the DMAC or DTC to execute the bus cycles alternatively between them by setting the IBCCS bit in BCR2. In this case, the priority between the DMAC and DTC does not change.

An internal bus access by the CPU, DTC, or DMAC and an external bus access by an external bus release request can be executed in parallel.

6.14.2 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority over that of the bus master that has taken control of the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific timings at which each bus master can release the bus.

(1) CPU

The CPU is the lowest-priority bus master, and if a bus request is received from the DTC or DMAC, the bus arbiter transfers the bus to the bus master that issued the request.

The timing for transfer of the bus is at the end of the bus cycle. In sleep mode, the bus is transferred synchronously with the clock.

Note, however, that the bus cannot be transferred in the following cases.

- The word or longword access is performed in some divisions.
- Stack handling is performed in multiple bus cycles.
- Transfer data read or write by memory transfer instructions, block transfer instructions, or TAS instruction.

  (In the block transfer instructions, the bus can be transferred in the write cycle and the following transfer data read cycle.)
- From the target read to write in the bit manipulation instructions or memory operation instructions.

  (In an instruction that performs no write operation according to the instruction condition, up to a cycle corresponding the write cycle)

(2) DTC

The DTC sends the internal bus arbiter a request for the bus when an activation request is generated. When the DTC accesses an external bus space, the DTC first takes control of the bus from the internal bus arbiter and then requests a bus to the external bus arbiter.

Once the DTC takes control of the bus, the DTC continues the transfer processing cycles. If a bus master whose priority is higher than the DTC requests the bus, the DTC transfers the bus to the higher priority bus master. If the IBCSS bit in BCR2 is set to 1, the DTC transfers the bus to the CPU.

Note, however, that the bus cannot be transferred in the following cases.

- During transfer information read
- During the first data transfer
- During transfer information write back

The DTC releases the bus when the consecutive transfer cycles completed.

(3) DMAC

The DMAC sends the internal bus arbiter a request for the bus when an activation request is generated. When the DMAC accesses an external bus space, the DMAC first takes control of the bus from the internal bus arbiter and then requests a bus to the external bus arbiter.

After the DMAC takes control of the bus, it may continue the transfer processing cycles or release the bus at the end of every bus cycle depending on the conditions.

The DMAC continues transfers without releasing the bus in the following case:

- Between the read cycle in the dual-address mode and the write cycle corresponding to the read cycle

If no bus master of a higher priority than the DMAC requests the bus and the IBCSS bit in BCR2 is cleared to 0, the DMAC continues transfers without releasing the bus in the following cases:

- During 1-block transfers in the block transfer mode
- During transfers in the burst mode

In other cases, the DMAC transfers the bus at the end of the bus cycle.

(4) External Bus Release

When the $\overline{\text{BREQ}}$ pin goes low and an external bus release request is issued while the BRLE bit in BCR1 is set to 1 with the corresponding ICR bit set to 1, a bus request is sent to the bus arbiter.

External bus release can be performed on completion of an external bus cycle.

6.15 Bus Controller Operation in Reset

In a reset, this LSI, including the bus controller, enters the reset state immediately, and any executing bus cycle is aborted.

6.16 Usage Notes

(1) Setting Registers

The BSC registers must be specified before accessing the external address space. In on-chip ROM disabled mode, the BSC registers must be specified before accessing the external address space for other than an instruction fetch access.

(2) External Bus Release Function and All-Module-Clock-Stop Mode

In this LSI, if the ACSE bit in MSTPCRA is set to 1, and then a SLEEP instruction is executed with the setting for all peripheral module clocks to be stopped (MSTPCRA and MSTPCRB = H'FFFFFFFF) or for operation of the 8-bit timer module alone (MSTPCRA and MSTPCRB = H'F[E to 0]FFFFFF), and a transition is made to the sleep state, the all-module-clock-stop mode is entered in which the clock is also stopped for the bus controller and I/O ports. For details, see section 20, Power-Down Modes.

In this state, the external bus release function is halted. To use the external bus release function in sleep mode, the ACSE bit in MSTPCRA must be cleared to 0. Conversely, if a SLEEP instruction to place the chip in all-module-clock-stop mode is executed in the external bus released state, the transition to all-module-clock-stop mode is deferred and performed until after the bus is recovered.

(3) External Bus Release Function and Software Standby

In this LSI, internal bus master operation does not stop even while the bus is released, as long as the program is running in on-chip ROM, etc., and no external access occurs. If a SLEEP instruction to place the chip in software standby mode is executed while the external bus is released, the transition to software standby mode is deferred and performed after the bus is recovered.

Also, since clock oscillation halts in software standby mode, if the $\overline{\text{BREQ}}$ signal goes low in this mode, indicating an external bus release request, the request cannot be answered until the chip has recovered from the software standby mode.

Note that the $\overline{\text{BACK}}$ and $\overline{\text{BREQO}}$ pins are both in the high-impedance state in software standby mode.

(4) $\overline{\text{BREQO}}$ Output Timing

When the BREQOE bit is set to 1 and the $\overline{\text{BREQO}}$ signal is output, both the $\overline{\text{BREQO}}$ and $\overline{\text{BACK}}$ signals may go low simultaneously.

This will occur if the next external access request occurs while internal bus arbitration is in progress after the chip samples a low level of the $\overline{\text{BREQ}}$ signal.

Section 7 DMA Controller (DMAC)

This LSI includes a 4-channel DMA controller (DMAC).

7.1 Features

- Maximum of 4-G byte address space can be accessed
- Byte, word, or longword can be set as data transfer unit
- Maximum of 4-G bytes (4,294,967,295 bytes) can be set as total transfer size
  Supports free-running mode in which total transfer size setting is not needed
- DMAC activation methods are auto-request, on-chip module interrupt, and external request.
  Auto request: CPU activates (cycle stealing or burst access can be selected)
  On-chip module interrupt: Interrupt requests from on-chip peripheral modules can be selected as an activation source
  External request: Low level or falling edge detection of the $\overline{\text{DREQ}}$ signal can be selected. External request is available for all four channels.
  In block transfer mode, low level detection is only available.
- Dual or single address mode can be selected as address mode
  Dual address mode: Both source and destination are specified by addresses
  Single address mode: Either source or destination is specified by the $\overline{\text{DREQ}}$ signal and the other is specified by address
- Normal, repeat, or block transfer can be selected as transfer mode
  Normal transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  Repeat transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  Repeat size of data is transferred and then a transfer address returns to the transfer start address
  Up to 65536 transfers (65,536 bytes/words/longwords) can be set as repeat size
  Block transfer mode: One block data is transferred at a single transfer request
  Up to 65,536 bytes/words/longwords can be set as block size

- Extended repeat area function which repeats the addressees within a specified area using the transfer address with the fixed upper bits (ring buffer transfer can be performed, as an example) is available

  One bit (two bytes) to 27 bits (128 Mbytes) for transfer source and destination can be set as extended repeat areas
- Address update can be selected from fixed address, offset addition, and increment or decrement by 1, 2, or 4

  Address update by offset addition enables to transfer data at addresses which are not placed continuously
- Word or longword data can be transferred to an address which is not aligned with the respective boundary

  Data is divided according to its address (byte or word) when it is transferred
- Two types of interrupts can be requested to the CPU

  A transfer end interrupt is generated after the number of data specified by the transfer counter is transferred. A transfer escape end interrupt is generated when the remaining total transfer size is less than the transfer data size at a single transfer request, when the repeat size of data transfer is completed, or when the extended repeat area overflows.

A block diagram of the DMAC is shown in figure 7.1.

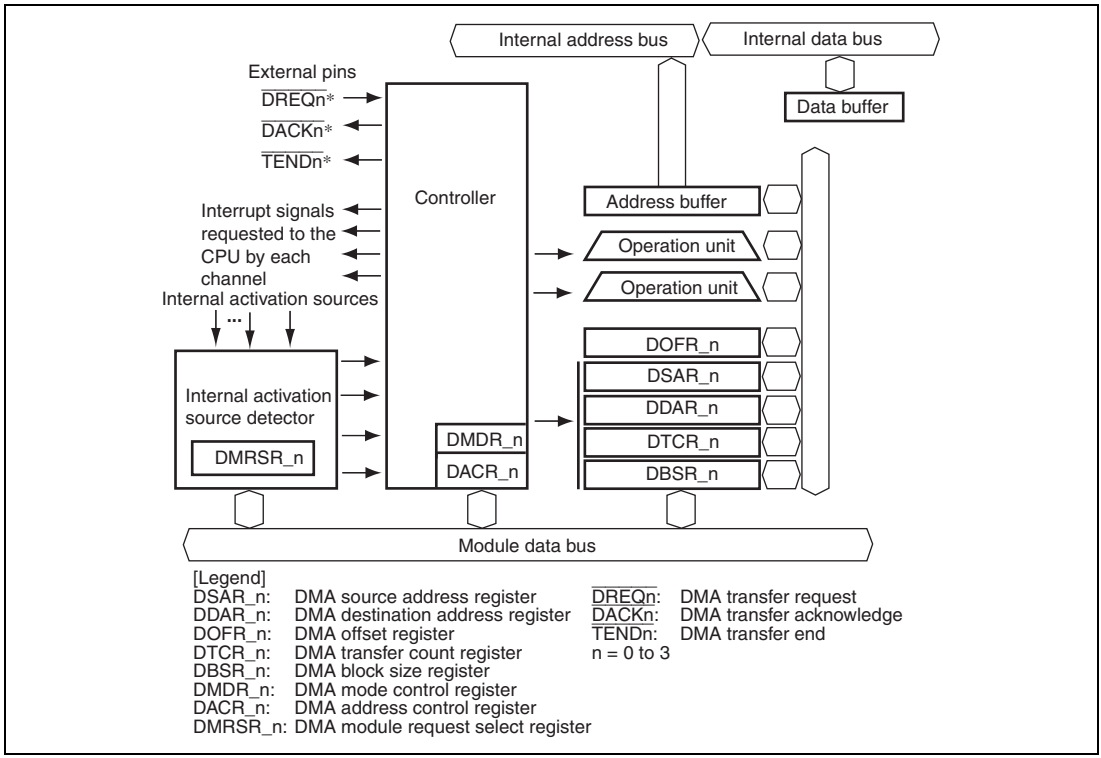


Figure 7.1 Block Diagram of DMAC

7.2 Input/Output Pins

Table 7.1 shows the pin configuration of the DMAC.

Table 7.1 Pin Configuration

Channel	Pin Name	Abbr.	I/O	Function
0	DMA transfer request 0	$\overline{\text{DREQ0}}$	Input	Channel 0 external request
	DMA transfer acknowledge 0	$\overline{\text{DACK0}}$	Output	Channel 0 single address transfer acknowledge
	DMA transfer end 0	$\overline{\text{TEND0}}$	Output	Channel 0 transfer end
1	DMA transfer request 1	$\overline{\text{DREQ1}}$	Input	Channel 1 external request
	DMA transfer acknowledge 1	$\overline{\text{DACK1}}$	Output	Channel 1 single address transfer acknowledge
	DMA transfer end 1	$\overline{\text{TEND1}}$	Output	Channel 1 transfer end
2	DMA transfer request 2	$\overline{\text{DREQ2}}$	Input	Channel 2 external request
	DMA transfer acknowledge 2	$\overline{\text{DACK2}}$	Output	Channel 2 single address transfer acknowledge
	DMA transfer end 2	$\overline{\text{TEND2}}$	Output	Channel 2 transfer end
3	DMA transfer request 3	$\overline{\text{DREQ3}}$	Input	Channel 3 external request
	DMA transfer acknowledge 3	$\overline{\text{DACK3}}$	Output	Channel 3 single address transfer acknowledge
	DMA transfer end 3	$\overline{\text{TEND3}}$	Output	Channel 3 transfer end

7.3 Register Descriptions

The DMAC has the following registers.

Channel 0:

- DMA source address register_0 (DSAR_0)
- DMA destination address register_0 (DDAR_0)
- DMA offset register_0 (DOFR_0)
- DMA transfer count register_0 (DTCR_0)
- DMA block size register_0 (DBSR_0)
- DMA mode control register_0 (DMDR_0)
- DMA address control register_0 (DACR_0)
- DMA module request select register_0 (DMRSR_0)

Channel 1:

- DMA source address register_1 (DSAR_1)
- DMA destination address register_1 (DDAR_1)
- DMA offset register_1 (DOFR_1)
- DMA transfer count register_1 (DTCR_1)
- DMA block size register_1 (DBSR_1)
- DMA mode control register_1 (DMDR_1)
- DMA address control register_1 (DACR_1)
- DMA module request select register_1 (DMRSR_1)

Channel 2:

- DMA source address register_2 (DSAR_2)
- DMA destination address register_2 (DDAR_2)
- DMA offset register_2 (DOFR_2)
- DMA transfer count register_2 (DTCR_2)
- DMA block size register_2 (DBSR_2)
- DMA mode control register_2 (DMDR_2)
- DMA address control register_2 (DACR_2)
- DMA module request select register_2 (DMRSR_2)

Channel 3:

- DMA source address register_3 (DSAR_3)
- DMA destination address register_3 (DDAR_3)
- DMA offset register_3 (DOFR_3)
- DMA transfer count register_3 (DTCR_3)
- DMA block size register_3 (DBSR_3)
- DMA mode control register_3 (DMDR_3)
- DMA address control register_3 (DACR_3)
- DMA module request select register_3 (DMRSR_3)

7.3.1 DMA Source Address Register (DSAR)

DSAR is a 32-bit readable/writable register that specifies the transfer source address. DSAR updates the transfer source address every time data is transferred. When DDAR is specified as the destination address (the DIRS bit in DACR is 1) in single address mode, DSAR is ignored.

Although DSAR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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	14	13	12	11	10	9	8
Bit Name								
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
Bit Name								
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

7.3.2 DMA Destination Address Register (DDAR)

DDAR is a 32-bit readable/writable register that specifies the transfer destination address. DDAR updates the transfer destination address every time data is transferred. When DSAR is specified as the source address (the DIRS bit in DACR is 0) in single address mode, DDAR is ignored.

Although DDAR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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	14	13	12	11	10	9	8
Bit Name								
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
Bit Name								
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

7.3.3 DMA Offset Register (DOFR)

DOFR is a 32-bit readable/writable register that specifies the offset to update the source and destination addresses. Although different values are specified for individual channels, the same values must be specified for the source and destination sides of a single channel.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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	14	13	12	11	10	9	8
Bit Name								
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
Bit Name								
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

7.3.4 DMA Transfer Count Register (DTCR)

DTCR is a 32-bit readable/writable register that specifies the size of data to be transferred (total transfer size).

To transfer 1-byte data in total, set H'00000001 in DTCR. When H'00000000 is set in this register, it means that the total transfer size is not specified and data is transferred with the transfer counter stopped (free running mode). When H'FFFFFFFF is set, the total transfer size is 4 Gbytes (4,294,967,295), which is the maximum size. While data is being transferred, this register indicates the remaining transfer size. The value corresponding to its data access size is subtracted every time data is transferred (byte: –1, word: –2, and longword: –4).

Although DTCR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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	14	13	12	11	10	9	8
Bit Name								
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
Bit Name								
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

7.3.5 DMA Block Size Register (DBSR)

DBSR specifies the repeat size or block size. DBSR is enabled in repeat transfer mode and block transfer mode and is disabled in normal transfer mode.

Bit	31	30	29	28	27	26	25	24
Bit Name	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24
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	23	22	21	20	19	18	17	16
Bit Name	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16
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	14	13	12	11	10	9	8
Bit Name	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8
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
Bit Name	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
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	Bit Name	Initial Value	R/W	Description
31 to 16	BKSZH31 to BKSZH16	Undefined	R/W	Specify the repeat size or block size. When H'0001 is set, the repeat or block size is one byte, one word, or one longword. When H'0000 is set, it means the maximum value (refer to table 7.1). While the DMA is in operation, the setting is fixed.
15 to 0	BKSZ15 to BKSZ0	Undefined	R/W	Indicate the remaining repeat or block size while the DMA is in operation. The value is decremented by 1 every time data is transferred. When the remaining size becomes 0, the value of the BKSZH bits is loaded. Set the same value as the BKSZH bits.

Table 7.2 Data Access Size, Valid Bits, and Settable Size

Mode	Data Access Size	BKSZH Valid Bits	BKSZ Valid Bits	Settable Size (Byte)
Repeat transfer and block transfer	Byte	31 to 16	15 to 0	1 to 65,536
	Word			2 to 131,072
	Longword			4 to 262,144

7.3.6 DMA Mode Control Register (DMDR)

DMDR controls the DMAC operation.

- DMDR_0

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
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	R
Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	ERRF	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/(W)*	R	R/(W)*	R/(W)*
Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Note: * Only 0 can be written to this bit after having been read as 1, to clear the flag.

- DMDR_1 to DMDR_3

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
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	R
Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	—	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R/(W)*	R/(W)*
Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Note: * Only 0 can be written to this bit after having been read as 1, to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
31	DTE	0	R/W	Data Transfer Enable Enables/disables a data transfer for the corresponding channel. When this bit is set to 1, it indicates that the DMAC is in operation. Setting this bit to 1 starts a transfer when the auto-request is selected. When the on-chip module interrupt or external request is selected, a transfer request after setting this bit to 1 starts the transfer. While data is being transferred, clearing this bit to 0 stops the transfer. In block transfer mode, if writing 0 to this bit while data is being transferred, this bit is cleared to 0 after the current 1-block size data transfer. If an event which stops (sustains) a transfer occurs externally, this bit is automatically cleared to 0 to stop the transfer. Operating modes and transfer methods must not be changed while this bit is set to 1. 0: Disables a data transfer 1: Enables a data transfer (DMA is in operation) [Clearing conditions] • When the specified total transfer size of transfers is completed • When a transfer is stopped by an overflow interrupt by a repeat size end • When a transfer is stopped by an overflow interrupt by an extended repeat size end • When a transfer is stopped by a transfer size error interrupt • When clearing this bit to 0 to stop a transfer In block transfer mode, this bit changes after the current block transfer. • When an address error or an NMI interrupt is requested • In the reset state or hardware standby mode

Bit	Bit Name	Initial Value	R/W	Description
30	DACKE	0	R/W	$\overline{\text{DACK}}$ Signal Output Enable Enables/disables the $\overline{\text{DACK}}$ signal output in single address mode. This bit is ignored in dual address mode. 0: Enables $\overline{\text{DACK}}$ signal output 1: Disables $\overline{\text{DACK}}$ signal output
29	TENDE	0	R/W	$\overline{\text{TEND}}$ Signal Output Enable Enables/disables the $\overline{\text{TEND}}$ signal output. 0: Enables $\overline{\text{TEND}}$ signal output 1: Disables $\overline{\text{TEND}}$ signal output
28	—	0	R/W	Reserved Initial value should not be changed.
27	DREQS	0	R/W	$\overline{\text{DREQ}}$ Select Selects whether a low level or the falling edge of the $\overline{\text{DREQ}}$ signal used in external request mode is detected. When a block transfer is performed in external request mode, clear this bit to 0. 0: Low level detection 1: Falling edge detection (the first transfer after a transfer enabled is detected on a low level)
26	NRD	0	R/W	Next Request Delay Selects the accepting timing of the next transfer request. 0: Starts accepting the next transfer request after completion of the current transfer 1: Starts accepting the next transfer request one cycle after completion of the current transfer
25, 24	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
23	ACT	0	R	Active State Indicates the operating state for the channel. 0: Waiting for a transfer request or a transfer disabled state by clearing the DTE bit to 0 1: Active state
22 to 20	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
19	ERRF	0	R/(W)*	System Error Flag Indicates that an address error or an NMI interrupt has been generated. This bit is available only in DMDR_0. Setting this bit to 1 prohibits writing to the DTE bit for all the channels. This bit is reserved in DMDR_1 to DMDR_3. It is always read as 0 and cannot be modified. 0: An address error or an NMI interrupt has not been generated 1: An address error or an NMI interrupt has been generated [Clearing condition] • When clearing to 0 after reading ERRF = 1 [Setting condition] • When an address error or an NMI interrupt has been generated However, when an address error or an NMI interrupt has been generated in DMAC module stop mode, this bit is not set to 1.
18	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
17	ESIF	0	R/(W)*	Transfer Escape Interrupt Flag Indicates that a transfer escape end interrupt has been requested. A transfer escape end means that a transfer is terminated before the transfer counter reaches 0. 0: A transfer escape end interrupt has not been requested 1: A transfer escape end interrupt has been requested [Clearing conditions] • When setting the DTE bit to 1 • When clearing to 0 before reading ESIF = 1 [Setting conditions] • When a transfer size error interrupt is requested • When a repeat size end interrupt is requested • When a transfer end interrupt by an extended repeat area overflow is requested

Bit	Bit Name	Initial Value	R/W	Description
16	DTIF	0	R/(W)*	Data Transfer Interrupt Flag Indicates that a transfer end interrupt by the transfer counter has been requested. 0: A transfer end interrupt by the transfer counter has not been requested 1: A transfer end interrupt by the transfer counter has been requested [Clearing conditions] • When setting the DTE bit to 1 • When clearing to 0 after reading DTIF = 1 [Setting condition] • When DTCR reaches 0 and the transfer is completed
15	DTSZ1	0	R/W	Data Access Size 1 and 0
14	DTSZ0	0	R/W	Select the data access size for a transfer. 00: Byte size (eight bits) 01: Word size (16 bits) 10: Longword size (32 bits) 11: Setting prohibited
13	MDS1	0	R/W	Transfer Mode Select 1 and 0
12	MDS0	0	R/W	Select the transfer mode. 00: Normal transfer mode 01: Block transfer mode 10: Repeat transfer mode 11: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description
11	TSEIE	0	R/W	Transfer Size Error Interrupt Enable Enables/disables a transfer size error interrupt. When the next transfer is requested while this bit is set to 1 and the contents of the transfer counter is less than the size of data to be transferred at a single transfer request, the DTE bit is cleared to 0. At this time, the ESIF bit is set to 1 to indicate that a transfer size error interrupt has been requested. The sources of a transfer size error are as follows: • In normal or repeat transfer mode, the total transfer size set in DTCR is less than the data access size • In block transfer mode, the total transfer size set in DTCR is less than the block size 0: Disables a transfer size error interrupt request 1: Enables a transfer size error interrupt request
10	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
9	ESIE	0	R/W	Transfer Escape Interrupt Enable Enables/disables a transfer escape end interrupt request. When the ESIF bit is set to 1 with this bit set to 1, a transfer escape end interrupt is requested to the CPU or DTC. The transfer end interrupt request is cleared by clearing this bit or the ESIF bit to 0. 0: Disables a transfer escape end interrupt 1: Enables a transfer escape end interrupt
8	DTIE	0	R/W	Data Transfer End Interrupt Enable Enables/disables a transfer end interrupt request by the transfer counter. When the DTIF bit is set to 1 with this bit set to 1, a transfer end interrupt is requested to the CPU or DTC. The transfer end interrupt request is cleared by clearing this bit or the DTIF bit to 0. 0: Disables a transfer end interrupt 1: Enables a transfer end interrupt

Bit	Bit Name	Initial Value	R/W	Description
7	DTF1	0	R/W	Data Transfer Factor 1 and 0
6	DTF0	0	R/W	Select a DMAC activation source. When the on-chip peripheral module setting is selected, the interrupt source should be selected by DMRSR. When the external request setting is selected, the sampling method should be selected by the DREQS bit. 00: Auto request (cycle stealing) 01: Auto request (burst access) 10: On-chip module interrupt 11: External request
5	DTA	0	R/W	Data Transfer Acknowledge This bit is valid in DMA transfer by the on-chip module interrupt source. This bit enables or disables to clear the source flag selected by DMRSR. 0: To clear the source in DMA transfer is disabled. Since the on-chip module interrupt source is not cleared in DMA transfer, it should be cleared by the CPU or DTC transfer. 1: To clear the source in DMA transfer is enabled. Since the on-chip module interrupt source is cleared in DMA transfer, it does not require an interrupt by the CPU or DTC transfer.
4, 3	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
2	DMAP2	0	R/W	DMA Priority Level 2 to 0
1	DMAP1	0	R/W	Select the priority level of the DMAC when using the CPU priority control function over DTC and DMAC. When the CPU has priority over the DMAC, the DMAC masks a transfer request and waits for the timing when the CPU priority becomes lower than the DMAC priority. The priority levels can be set to the individual channels. This bit is valid when the CPUPCE bit in CPUPCR is set to 1. 000: Priority level 0 (low) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (high)
0	DMAP0	0	R/W	

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

7.3.7 DMA Address Control Register (DACR)

DACR specifies the operating mode and transfer method.

Bit	31	30	29	28	27	26	25	24
Bit Name	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	SAT1	SAT0	—	—	DAT1	DAT0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31	AMS	0	R/W	Address Mode Select Selects address mode from single or dual address mode. In single address mode, the $\overline{\text{DACK}}$ pin is enabled according to the DACKE bit. 0: Dual address mode 1: Single address mode
30	DIRS	0	R/W	Single Address Direction Select Specifies the data transfer direction in single address mode. This bit s ignored in dual address mode. 0: Specifies DSAR as source address 1: Specifies DDAR as destination address
29 to 27	—	0	R/W	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
26	RPTIE	0	R/W	Repeat Size End Interrupt Enable Enables/disables a repeat size end interrupt request. In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat-size data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested. Even when the repeat area is not specified (ARS1 = 1 and ARS0 = 0), a repeat size end interrupt after a 1-block data transfer can be requested. In addition, in block transfer mode, when the next transfer is requested after 1-block data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested. 0: Disables a repeat size end interrupt 1: Enables a repeat size end interrupt
25	ARS1	0	R/W	Area Select 1 and 0
24	ARS0	0	R/W	Specify the block area or repeat area in block or repeat transfer mode. 00: Specify the block area or repeat area on the source address 01: Specify the block area or repeat area on the destination address 10: Do not specify the block area or repeat area 11: Setting prohibited
23, 22	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
21	SAT1	0	R/W	Source Address Update Mode 1 and 0
20	SAT0	0	R/W	Select the update method of the source address (DSAR). When DSAR is not specified as the transfer source in single address mode, this bit is ignored. 00: Source address is fixed 01: Source address is updated by adding the offset 10: Source address is updated by adding 1, 2, or 4 according to the data access size 11: Source address is updated by subtracting 1, 2, or 4 according to the data access size

Bit	Bit Name	Initial Value	R/W	Description
19, 18	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
17	DAT1	0	R/W	Destination Address Update Mode 1 and 0
16	DAT0	0	R/W	Select the update method of the destination address (DDAR). When DDAR is not specified as the transfer destination in single address mode, this bit is ignored. 00: Destination address is fixed 01: Destination address is updated by adding the offset 10: Destination address is updated by adding 1, 2, or 4 according to the data access size 11: Destination address is updated by subtracting 1, 2, or 4 according to the data access size
15	SARIE	0	R/W	Interrupt Enable for Source Address Extended Area Overflow Enables/disables an interrupt request for an extended area overflow on the source address. When an extended repeat area overflow on the source address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the source address is requested. When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which a transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped. When the extended repeat area is not specified, this bit is ignored. 0: Disables an interrupt request for an extended area overflow on the source address 1: Enables an interrupt request for an extended area overflow on the source address
14, 13	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
12	SARA4	0	R/W	Source Address Extended Repeat Area
11	SARA3	0	R/W	Specify the extended repeat area on the source address (DSAR). With the extended repeat area, the specified lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2. When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively. When an overflow in the extended repeat area occurs with the SARIE bit set to 1, an interrupt can be requested. Table 7.3 shows the settings and areas of the extended repeat area.
10	SARA2	0	R/W	
9	SARA1	0	R/W	
8	SARA0	0	R/W	
7	DARIE	0	R/W	Destination Address Extended Repeat Area Overflow Interrupt Enable Enables/disables an interrupt request for an extended area overflow on the destination address. When an extended repeat area overflow on the destination address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the destination address is requested. When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which the transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped. When the extended repeat area is not specified, this bit is ignored. 0: Disables an interrupt request for an extended area overflow on the destination address 1: Enables an interrupt request for an extended area overflow on the destination address
6, 5	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
4	DARA4	0	R/W	Destination Address Extended Repeat Area
3	DARA3	0	R/W	Specify the extended repeat area on the destination address (DDAR). With the extended repeat area, the specified lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2. When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively. When an overflow in the extended repeat area occurs with the DARIE bit set to 1, an interrupt can be requested. Table 7.3 shows the settings and areas of the extended repeat area.
2	DARA2	0	R/W	
1	DARA1	0	R/W	
0	DARA0	0	R/W	

Table 7.3 Settings and Areas of Extended Repeat Area

SARA4 to SARA0 or DARA4 to DARA0	Extended Repeat Area
00000	Not specified
00001	2 bytes specified as extended repeat area by the lower 1 bit of the address
00010	4 bytes specified as extended repeat area by the lower 2 bits of the address
00011	8 bytes specified as extended repeat area by the lower 3 bits of the address
00100	16 bytes specified as extended repeat area by the lower 4 bits of the address
00101	32 bytes specified as extended repeat area by the lower 5 bits of the address
00110	64 bytes specified as extended repeat area by the lower 6 bits of the address
00111	128 bytes specified as extended repeat area by the lower 7 bits of the address
01000	256 bytes specified as extended repeat area by the lower 8 bits of the address
01001	512 bytes specified as extended repeat area by the lower 9 bits of the address
01010	1 kbyte specified as extended repeat area by the lower 10 bits of the address
01011	2 kbytes specified as extended repeat area by the lower 11 bits of the address
01100	4 kbytes specified as extended repeat area by the lower 12 bits of the address
01101	8 kbytes specified as extended repeat area by the lower 13 bits of the address
01110	16 kbytes specified as extended repeat area by the lower 14 bits of the address
01111	32 kbytes specified as extended repeat area by the lower 15 bits of the address
10000	64 kbytes specified as extended repeat area by the lower 16 bits of the address
10001	128 kbytes specified as extended repeat area by the lower 17 bits of the address
10010	256 kbytes specified as extended repeat area by the lower 18 bits of the address
10011	512 kbytes specified as extended repeat area by the lower 19 bits of the address
10100	1 Mbyte specified as extended repeat area by the lower 20 bits of the address
10101	2 Mbytes specified as extended repeat area by the lower 21 bits of the address
10110	4 Mbytes specified as extended repeat area by the lower 22 bits of the address
10111	8 Mbytes specified as extended repeat area by the lower 23 bits of the address
11000	16 Mbytes specified as extended repeat area by the lower 24 bits of the address
11001	32 Mbytes specified as extended repeat area by the lower 25 bits of the address
11010	64 Mbytes specified as extended repeat area by the lower 26 bits of the address
11011	128 Mbytes specified as extended repeat area by the lower 27 bits of the address
111××	Setting prohibited

[Legend]

×: Don't care

7.3.8 DMA Module Request Select Register (DMRSR)

DMRSR is an 8-bit readable/writable register that specifies the on-chip module interrupt source. The vector number of the interrupt source is specified in eight bits. However, 0 is regarded as no interrupt source. For the vector numbers of the interrupt sources, refer to table 7.5.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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

7.4 Transfer Modes

Table 7.4 shows the DMAC transfer modes. The transfer modes can be specified to the individual channels.

Table 7.4 Transfer Modes

				Address Register	
Address Mode	**Transfer mode**	**Activation Source**	**Common Function**	**Source**	**Destination**
Dual address	• Normal transfer • Repeat transfer • Block transfer Repeat or block size = 1 to 65,536 bytes, 1 to 65,536 words, or 1 to 65,536 longwords	• Auto request (activated by CPU) • On-chip module interrupt • External request	• Total transfer size: 1 to 4 Gbytes or not specified • Offset addition • Extended repeat area function	DSAR	DDAR
Single address	• Instead of specifying the source or destination address registers, data is directly transferred from/to the external device using the $\overline{\text{DACK}}$ pin • The same settings as above are available other than address register setting (e.g., above transfer modes can be specified) • One transfer can be performed in one bus cycle (the types of transfer modes are the same as those of dual address modes)			DSAR/ $\overline{\text{DACK}}$	$\overline{\text{DACK}}$/ DDAR

When the auto request setting is selected as the activation source, the cycle stealing or burst access can be selected. When the total transfer size is not specified (DTCR = H'00000000), the transfer counter is stopped and the transfer is continued without the limitation of the transfer count.

7.5 Operations

7.5.1 Address Modes

(1) Dual Address Mode

In dual address mode, the transfer source address is specified in DSAR and the transfer destination address is specified in DDAR. A transfer at a time is performed in two bus cycles (when the data bus width is less than the data access size or the access address is not aligned with the boundary of the data access size, the number of bus cycles are needed more than two because one bus cycle is divided into multiple bus cycles).

In the first bus cycle, data at the transfer source address is read and in the next cycle, the read data is written to the transfer destination address.

The read and write cycles are not separated. Other bus cycles (bus cycle by other bus masters, refresh cycle, and external bus release cycle) are not generated between read and write cycles.

The $\overline{\text{TEND}}$ signal output is enabled or disabled by the TENDE bit in DMDR. The $\overline{\text{TEND}}$ signal is output in two bus cycles. When an idle cycle is inserted before the bus cycle, the $\overline{\text{TEND}}$ signal is also output in the idle cycle. The $\overline{\text{DACK}}$ signal is not output.

Figure 7.2 shows an example of the signal timing in dual address mode and figure 7.3 shows the operation in dual address mode.

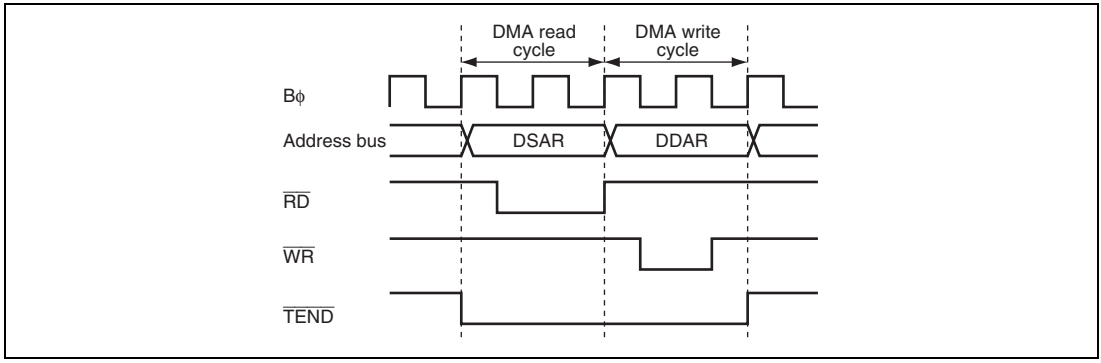


Figure 7.2 Example of Signal Timing in Dual Address Mode

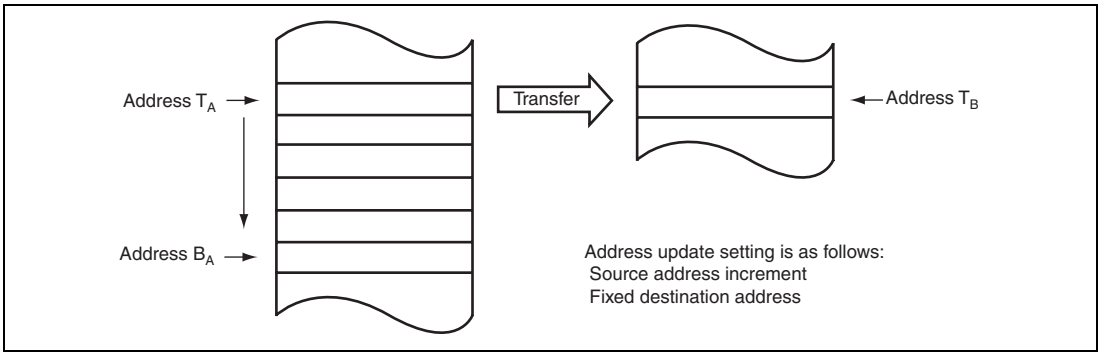


Figure 7.3 Operations in Dual Address Mode

(2) Single Address Mode

In single address mode, data between an external device and an external memory is directly transferred using the $\overline{\text{DACK}}$ pin instead of DSAR or DDAR. A transfer at a time is performed in one bus cycle. In this mode, the data bus width must be the same as the data access size. For details on the data bus width, see section 6, Bus Controller (BSC).

The DMAC accesses an external device as the transfer source or destination by outputting the strobe signal ($\overline{\text{DACK}}$) to the external device with $\overline{\text{DACK}}$ and accesses the other transfer target by outputting the address. Accordingly, the DMA transfer is performed in one bus cycle. Figure 7.4 shows an example of a transfer between an external memory and an external device with the $\overline{\text{DACK}}$ pin. In this example, the external device outputs data on the data bus and the data is written to the external memory in the same bus cycle.

The transfer direction is decided by the DIRS bit in DACR which specifies an external device with the $\overline{\text{DACK}}$ pin as the transfer source or destination. When DIRS = 0, data is transferred from an external memory (DSAR) to an external device with the $\overline{\text{DACK}}$ pin. When DIRS = 1, data is transferred from an external device with the $\overline{\text{DACK}}$ pin to an external memory (DDAR). The settings of registers which are not used as the transfer source or destination are ignored.

The $\overline{\text{DACK}}$ signal output is enabled in single address mode by the DACKE bit in DMDR. The $\overline{\text{DACK}}$ signal is low active.

The $\overline{\text{TEND}}$ signal output is enabled or disabled by the TENDE bit in DMDR. The $\overline{\text{TEND}}$ signal is output in one bus cycle. When an idle cycle is inserted before the bus cycle, the $\overline{\text{TEND}}$ signal is also output in the idle cycle.

Figure 7.5 shows an example of timing charts in single address mode and figure 7.6 shows an example of operation in single address mode.

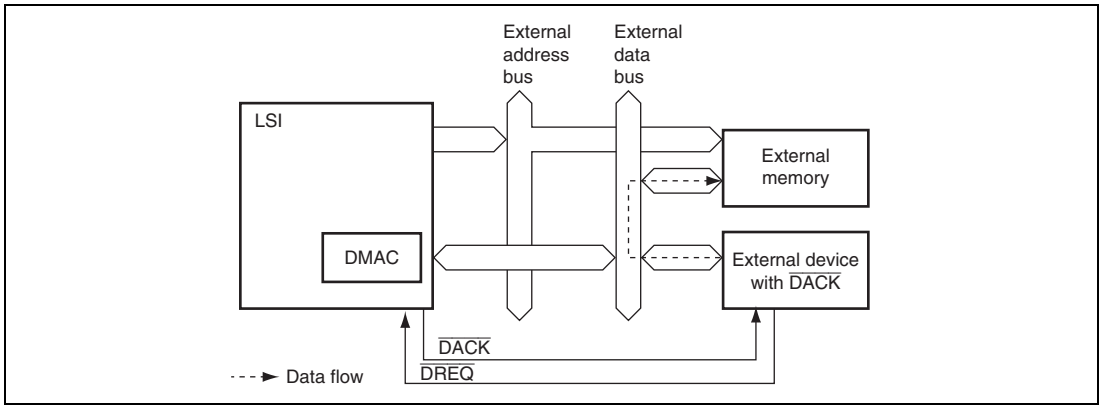


Figure 7.4 Data Flow in Single Address Mode

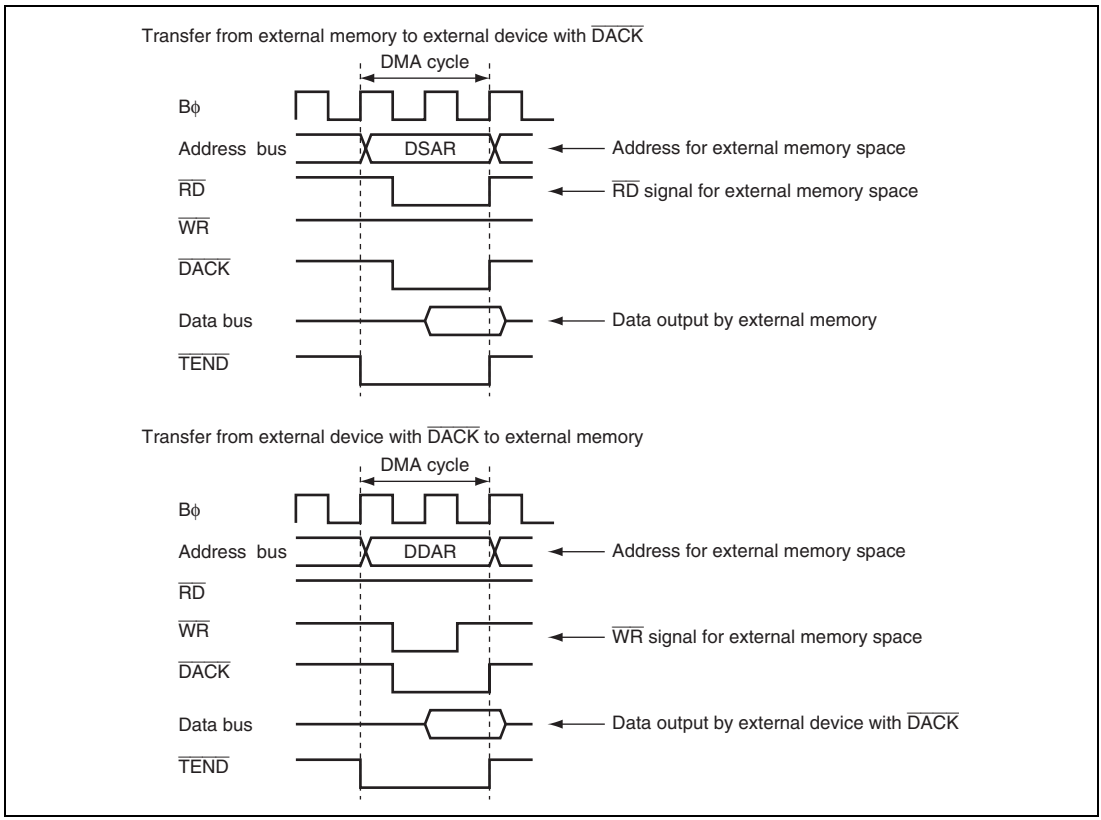


Figure 7.5 Example of Signal Timing in Single Address Mode

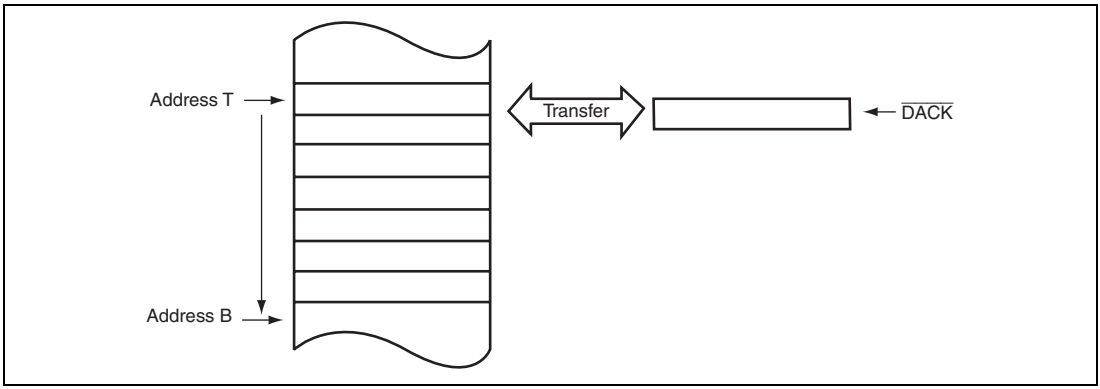


Figure 7.6 Operations in Single Address Mode

7.5.2 Transfer Modes

(1) Normal Transfer Mode

In normal transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. DBSR is ignored in normal transfer mode.

The $\overline{\text{TEND}}$ signal is output only in the last DMA transfer. The $\overline{\text{DACK}}$ signal is output every time a transfer request is received and a transfer starts.

Figure 7.7 shows an example of the signal timing in normal transfer mode and figure 7.8 shows the operation in normal transfer mode.

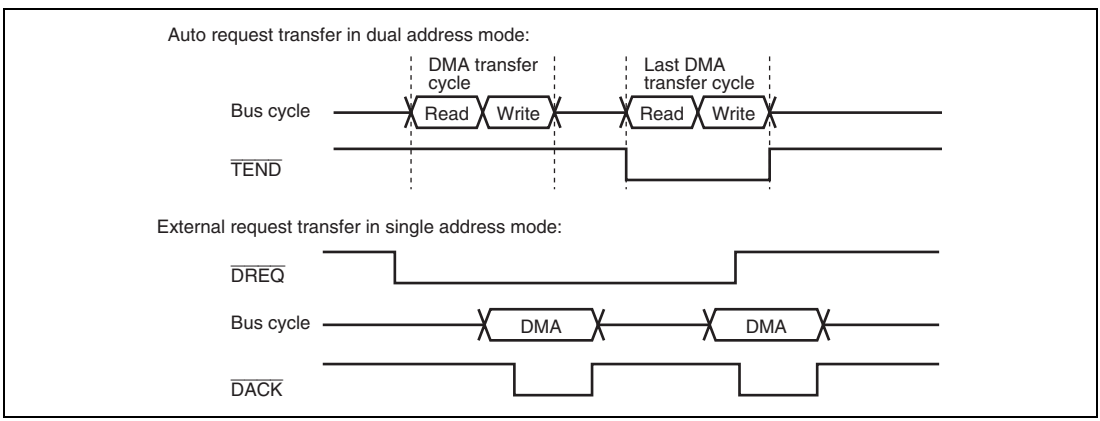


Figure 7.7 Example of Signal Timing in Normal Transfer Mode

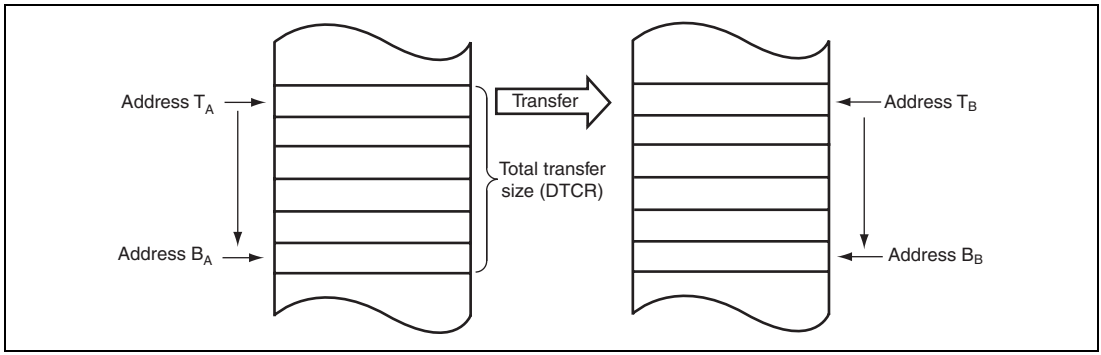


Figure 7.8 Operations in Normal Transfer Mode

(2) Repeat Transfer Mode

In repeat transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. The repeat size can be specified in DBSR up to 65536 × data access size.

The repeat area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the repeat area returns to the transfer start address when the repeat size of transfers is completed. This operation is repeated until the total transfer size specified in DTCR is completed. When H'00000000 is specified in DTCR, it is regarded as the free running mode and repeat transfer is continued until the DTE bit in DMDR is cleared to 0.

In addition, a DMA transfer can be stopped and a repeat size end interrupt can be requested to the CPU or DTC when the repeat size of transfers is completed. When the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit is set to 1, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1 to complete the transfer. At this time, an interrupt is requested to the CPU or DTC when the ESIE bit in DMDR is set to 1.

The timings of the $\overline{\text{TEND}}$ and $\overline{\text{DACK}}$ signals are the same as in normal transfer mode.

Figure 7.9 shows the operation in repeat transfer mode while dual address mode is set.

When the repeat area is specified as neither source nor destination address side, the operation is the same as the normal transfer mode operation shown in figure 7.8. In this case, a repeat size end interrupt can also be requested to the CPU when the repeat size of transfers is completed.

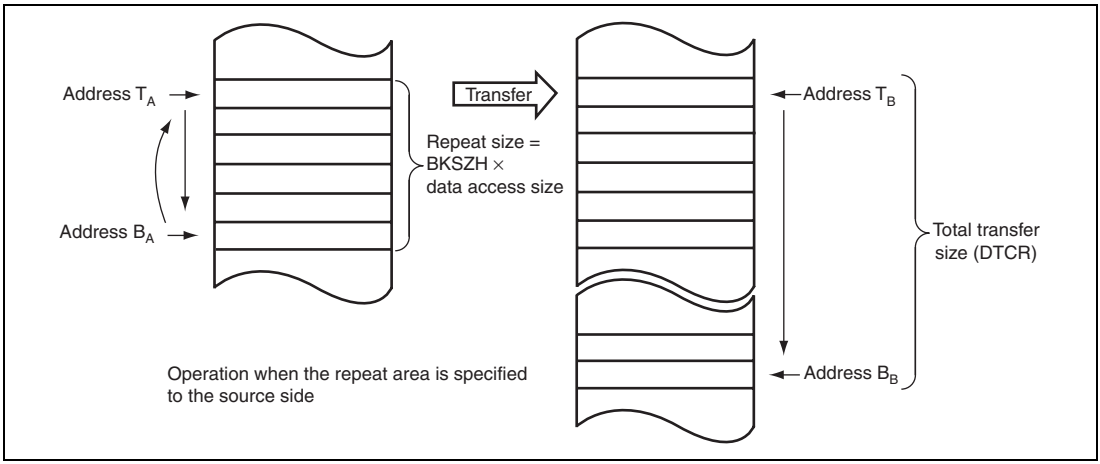


Figure 7.9 Operations in Repeat Transfer Mode

(3) Block Transfer Mode

In block transfer mode, one block size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as total transfer size by DTCR. The block size can be specified in DBSR up to 65536 × data access size.

While one block of data is being transferred, transfer requests from other channels are suspended. When the transfer is completed, the bus is released to the other bus master.

The block area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the block area returns to the transfer start address when the block size of data is completed. When the block area is specified as neither source nor destination address side, the operation continues without returning the address to the transfer start address. A repeat size end interrupt can be requested.

The $\overline{\text{TEND}}$ signal is output every time 1-block data is transferred in the last DMA transfer cycle. When the external request is selected as an activation source, the low level detection of the $\overline{\text{DREQ}}$ signal (DREQS = 0) should be selected.

When an interrupt request by an extended repeat area overflow is used in block transfer mode, settings should be selected carefully. For details, see section 7.5.5, Extended Repeat Area Function.

Figure 7.10 shows an example of the DMA transfer timing in block transfer mode. The transfer conditions are as follows:

- Address mode: single address mode
- Data access size: byte
- 1-block size: three bytes

The block transfer mode operations in single address mode and in dual address mode are shown in figures 7.11 and 7.12, respectively.

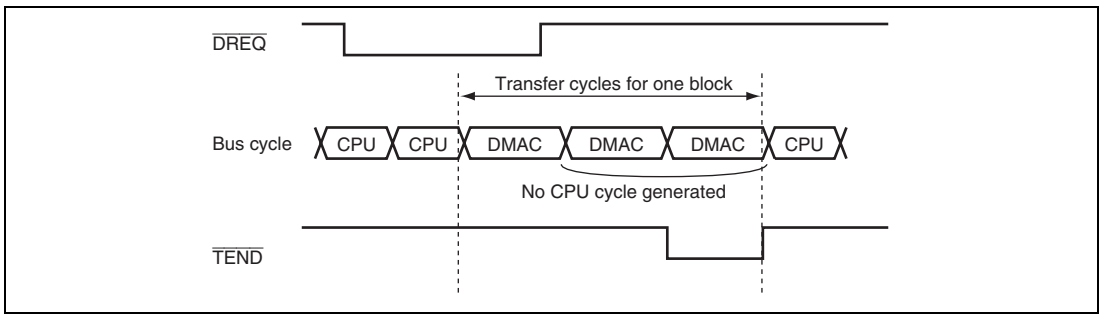


Figure 7.10 Operations in Block Transfer Mode

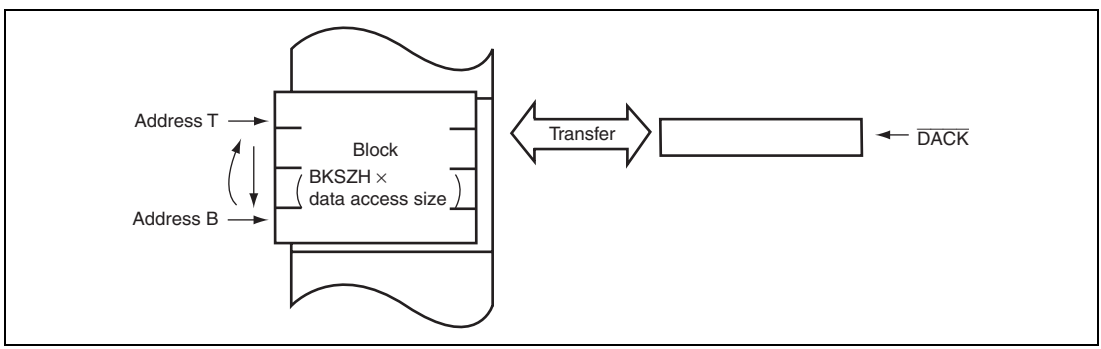


Figure 7.11 Operation in Single Address Mode in Block Transfer Mode (Block Area Specified)

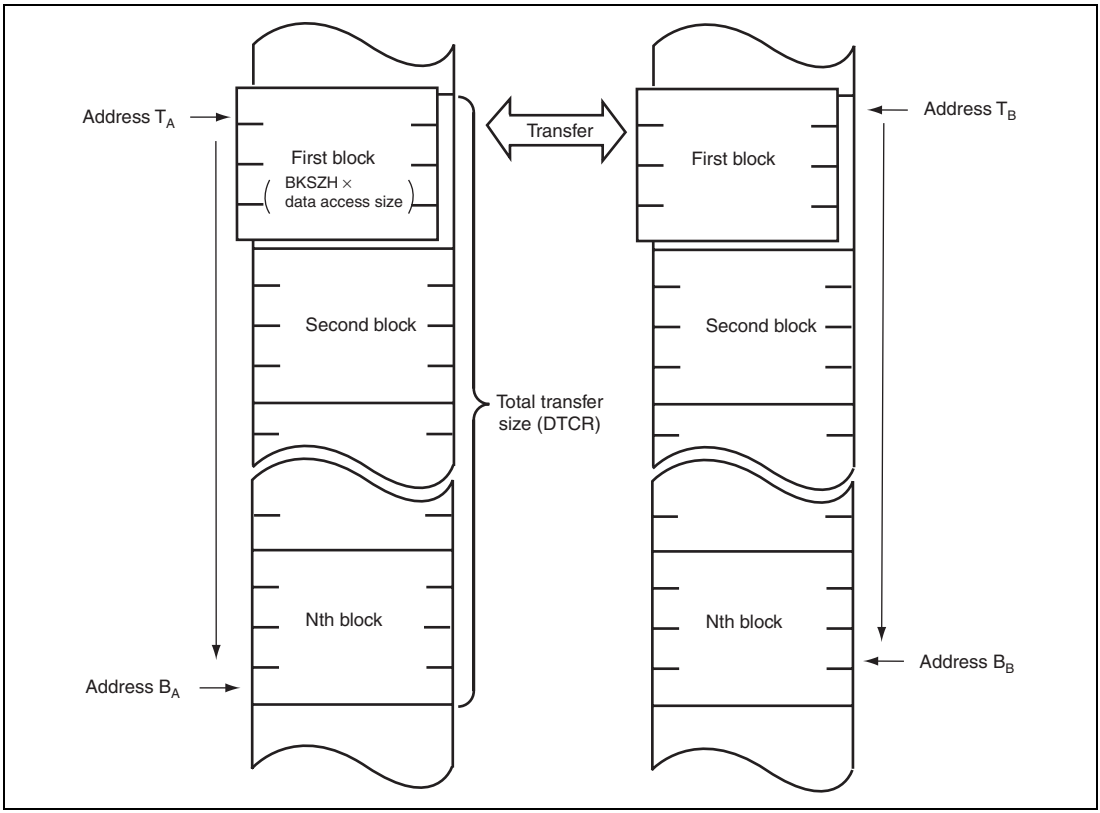


Figure 7.12 Operation in Dual Address Mode in Block Transfer Mode (Block Area Not Specified)

7.5.3 Activation Sources

The DMAC is activated by an auto request, an on-chip module interrupt, and an external request. The activation source is specified by bits DTF1 and DTF0 in DMDR.

(1) Activation by Auto Request

The auto request activation is used when a transfer request from an external device or an on-chip peripheral module is not generated such as a transfer between memory and memory or between memory and an on-chip peripheral module which does not request a transfer. A transfer request is automatically generated inside the DMAC. In auto request activation, setting the DTE bit in DMDR starts a transfer. The bus mode can be selected from cycle stealing and burst modes.

(2) Activation by On-Chip Module Interrupt

An interrupt request from an on-chip peripheral module (on-chip peripheral module interrupt) is used as a transfer request. When a DMA transfer is enabled (DTE = 1), the DMA transfer is started by an on-chip module interrupt.

The activation source of the on-chip module interrupt is selected by the DMA module request select register (DMRSR). The activation sources are specified to the individual channels. Table 7.5 is a list of on-chip module interrupts for the DMAC. The interrupt request selected as the activation source can generate an interrupt request simultaneously to the CPU or DTC. For details, refer to section 5, Interrupt Controller.

The DMAC receives interrupt requests by on-chip peripheral modules independent of the interrupt controller. Therefore, the DMAC is not affected by priority given in the interrupt controller.

When the DMAC is activated while DTA = 1, the interrupt request flag is automatically cleared by a DMA transfer. If multiple channels use a single transfer request as an activation source, when the channel having priority is activated, the interrupt request flag is cleared. In this case, other channels may not be activated because the transfer request is not held in the DMAC.

When the DMAC is activated while DTA = 0, the interrupt request flag is not cleared by the DMAC and should be cleared by the CPU or DTC transfer.

When an activation source is selected while DTE = 0, the activation source does not request a transfer to the DMAC. It requests an interrupt to the CPU or DTC.

In addition, make sure that an interrupt request flag as an on-chip module interrupt source is cleared to 0 before writing 1 to the DTE bit.

Table 7.5 List of On-chip module interrupts to DMAC

On-Chip Module Interrupt Source	On-Chip Module	DMRSR (Vector Number)
ADI (conversion end interrupt for A/D converter)	A/D	86
TGI0A (TGI0A input capture/compare match)	TPU_0	88
TGI1A (TGI1A input capture/compare match)	TPU_1	93
TGI2A (TGI2A input capture/compare match)	TPU_2	97
TGI3A (TGI3A input capture/compare match)	TPU_3	101
TGI4A (TGI4A input capture/compare match)	TPU_4	106
TGI5A (TGI5A input capture/compare match)	TPU_5	110
RXI0 (receive data full interrupt for SCI channel 0)	SCI_0	145
TXI0 (transmit data empty interrupt for SCI channel 0)	SCI_0	146
RXI1 (receive data full interrupt for SCI channel 1)	SCI_1	149
TXI1 (transmit data empty interrupt for SCI channel 1)	SCI_1	150
RXI2 (receive data full interrupt for SCI channel 2)	SCI_2	153
TXI2 (transmit data empty interrupt for SCI channel 2)	SCI_2	154
RXI4 (receive data full interrupt for SCI channel 4)	SCI_4	161
TXI4 (transmit data empty interrupt for SCI channel 4)	SCI_4	162

(3) Activation by External Request

A transfer is started by a transfer request signal ($\overline{\text{DREQ}}$) from an external device. When a DMA transfer is enabled (DTE = 1), the DMA transfer is started by the $\overline{\text{DREQ}}$ assertion. When a DMA transfer between on-chip peripheral modules is performed, select an activation source from the auto request and on-chip module interrupt (the external request cannot be used).

A transfer request signal is input to the $\overline{\text{DREQ}}$ pin. The $\overline{\text{DREQ}}$ signal is detected on the falling edge or low level. Whether the falling edge or low level detection is used is selected by the DREQS bit in DMDR. To perform a block transfer, select the low level detection (DREQS = 0).

When an external request is selected as an activation source, clear the DDR bit to 0 and set the ICR bit to 1 for the corresponding pin. For details, see section 9, I/O Ports.

7.5.4 Bus Modes

There are two types of bus modes: cycle stealing and burst.

When an activation source is the auto request, the cycle stealing or burst mode is selected by bit DTF0 in DMDR. When an activation source is the on-chip module interrupt or external request, the cycle stealing mode is selected.

(1) Cycle Stealing Mode

In cycle stealing mode, the DMAC releases the bus every time one unit of transfers (byte, word, longword, or 1-block size) is completed. After that, when a transfer is requested, the DMAC obtains the bus to transfer 1-unit data and then releases the bus on completion of the transfer. This operation is continued until the transfer end condition is satisfied.

When a transfer is requested to another channel during a DMA transfer, the DMAC releases the bus and then transfers data for the requested channel. For details on operations when a transfer is requested to multiple channels, see section 7.5.8, Priority of Channels.

Figure 7.13 shows an example of timing in cycle stealing mode. The transfer conditions are as follows:

- Address mode: Single address mode
- Sampling method of the $\overline{\text{DREQ}}$ signal: Low level detection

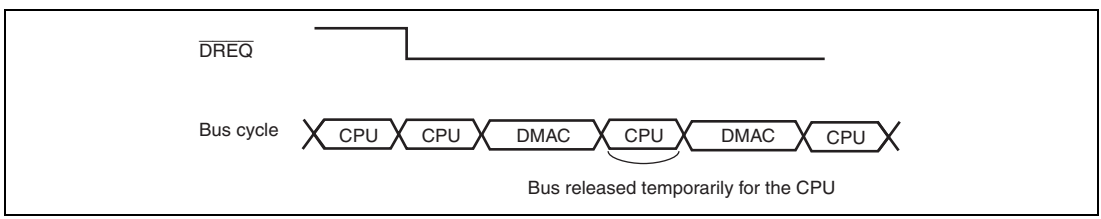


Figure 7.13 Example of Timing in Cycle Stealing Mode

(2) Burst Mode

In burst mode, once it takes the bus, the DMAC continues a transfer without releasing the bus until the transfer end condition is satisfied. Even if a transfer is requested from another channel having priority, the transfer is not stopped once it is started. The DMAC releases the bus in the next cycle after the transfer for the channel in burst mode is completed. This is similarly to operation in cycle stealing mode. However, setting the IBCCS bit in IBCR of the bus controller makes the DMAC release the bus to pass the bus to another bus master.

In block transfer mode, the burst mode setting is ignored (operation is the same as that in burst mode during one block of transfers). The DMAC is always operated in cycle stealing mode.

Clearing the DTE bit in DMDR stops a DMA transfer. A transfer requested before the DTE bit is cleared to 0 by the DMAC is executed. When an interrupt by a transfer size error, a repeat size end, or an extended repeat area overflow occurs, the DTE bit is cleared to 0 and the transfer ends.

Figure 7.14 shows an example of timing in burst mode.

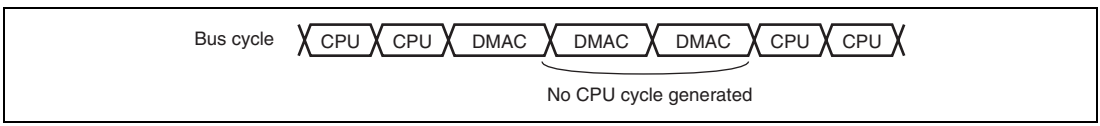


Figure 7.14 Example of Timing in Burst Mode

7.5.5 Extended Repeat Area Function

The source and destination address sides can be specified as the extended repeat area. The contents of the address register repeat addresses within the area specified as the extended repeat area. For example, to use a ring buffer as the transfer target, the contents of the address register should return to the start address of the buffer every time the contents reach the end address of the buffer (overflow on the ring buffer address). This operation can automatically be performed using the extended repeat area function of the DMAC.

The extended repeat areas can be specified independently to the source address register (DSAR) and destination address register (DDAR).

The extended repeat area on the source address is specified by bits SARA4 to SARA0 in DACR. The extended repeat area on the destination address is specified by bits DARA4 to DARA0 in DACR. The extended repeat area sizes for each side can be specified independently.

A DMA transfer is stopped and an interrupt by an extended repeat area overflow can be requested to the CPU when the contents of the address register reach the end address of the extended repeat area. When an overflow on the extended repeat area set in DSAR occurs while the SARIE bit in DACR is set to 1, the ESIF bit in DMDR is set to 1 and the DTE bit in DMDR is cleared to 0 to stop the transfer. At this time, if the ESIE bit in DMDR is set to 1, an interrupt by an extended repeat area overflow is requested to the CPU. When the DARIE bit in DACR is set to 1, an overflow on the extended repeat area set in DDAR occurs, meaning that the destination side is a target. During the interrupt handling, setting the DTE bit in DMDR resumes the transfer.

Figure 7.15 shows an example of the extended repeat area operation.

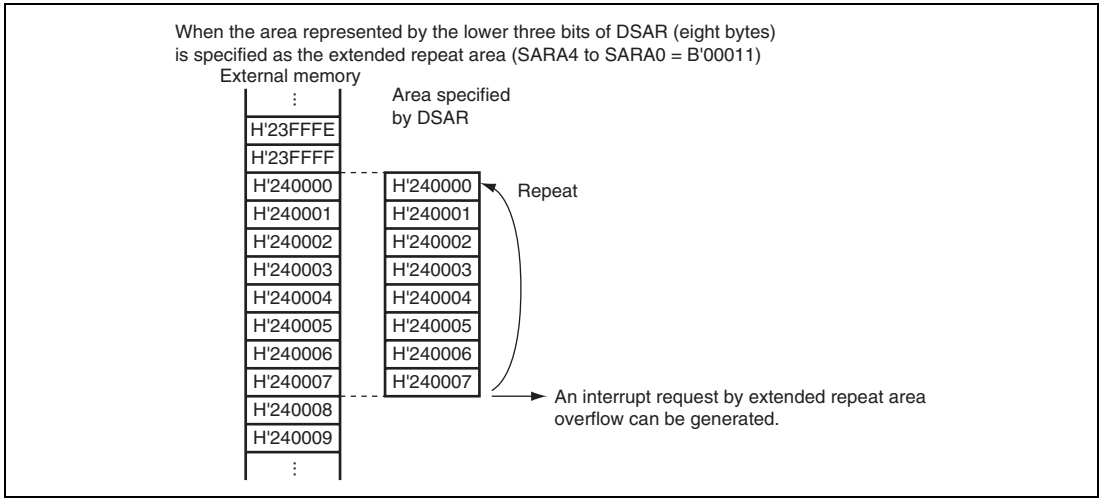


Figure 7.15 Example of Extended Repeat Area Operation

When an interrupt by an extended repeat area overflow is used in block transfer mode, the following should be taken into consideration.

When a transfer is stopped by an interrupt by an extended repeat area overflow, the address register must be set so that the block size is a power of 2 or the block size boundary is aligned with the extended repeat area boundary. When an overflow on the extended repeat area occurs during a transfer of one block, the interrupt by the overflow is suspended and the transfer overruns.

Figure 7.16 shows examples when the extended repeat area function is used in block transfer mode.

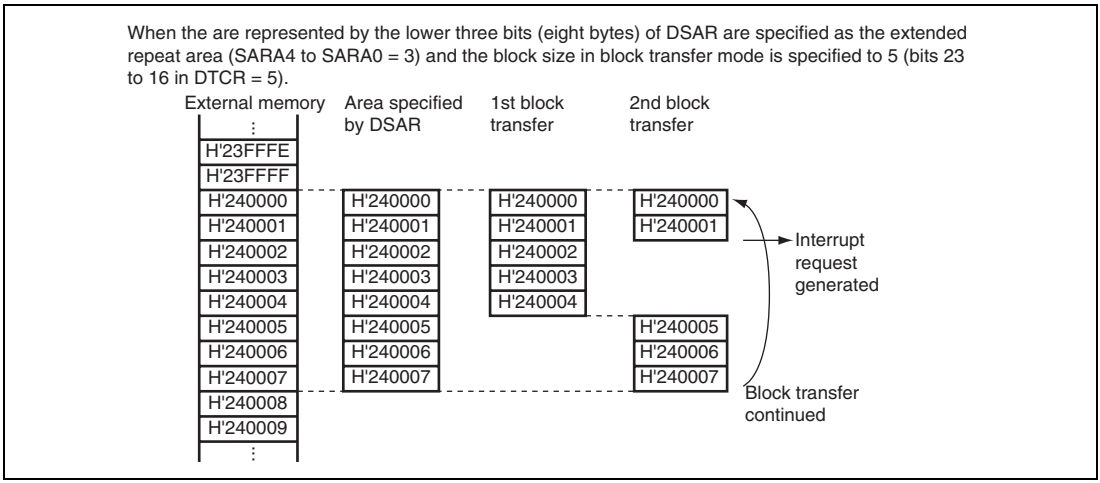


Figure 7.16 Example of Extended Repeat Area Function in Block Transfer Mode

7.5.6 Address Update Function using Offset

The source and destination addresses are updated by fixing, increment/decrement by 1, 2, or 4, or offset addition. When the offset addition is selected, the offset specified by the offset register (DOFR) is added to the address every time the DMAC transfers the data access size of data. This function realizes a data transfer where addresses are allocated to separated areas.

Figure 7.17 shows the address update method.

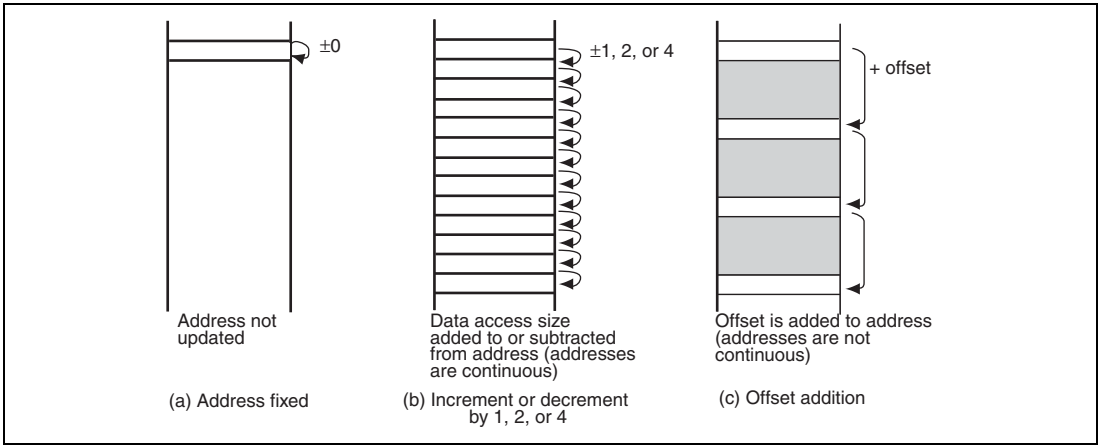


Figure 7.17 Address Update Method

In item (a), Address fixed, the transfer source or destination address is not updated indicating the same address.

In item (b), Increment or decrement by 1, 2, or 4, the transfer source or destination address is incremented or decremented by the value according to the data access size at each transfer. Byte, word, or longword can be specified as the data access size. The value of 1 for byte, 2 for word, and 4 for longword is used for updating the address. This operation realizes the data transfer placed in consecutive areas.

In item (c), Offset addition, the address update does not depend on the data access size. The offset specified by DOFR is added to the address every time the DMAC transfers data of the data access size.

The address is calculated by the offset set in DOFR and the contents of DSAR and DDAR. Although the DMAC calculates only addition, an offset subtraction can be realized by setting the negative value in DOFR. In this case, the negative value must be 2's complement.

(1) Basic Transfer Using Offset

Figure 7.18 shows a basic operation of a transfer using the offset addition.

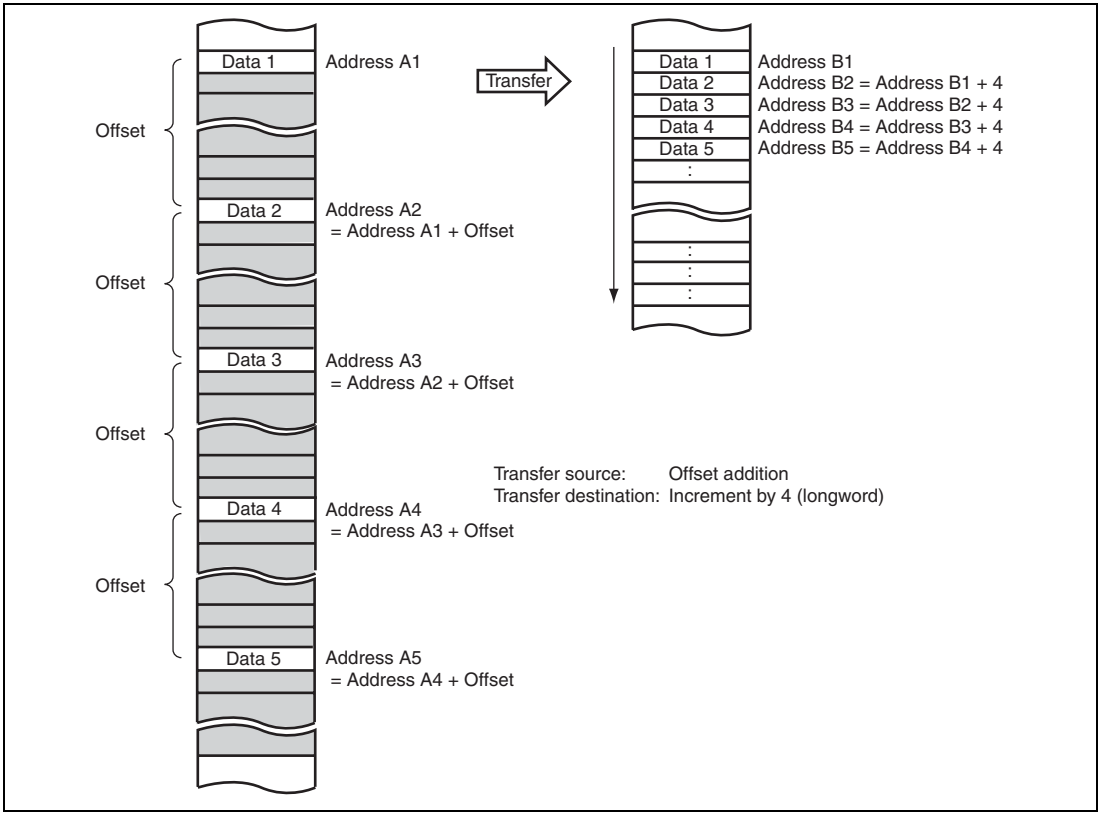


Figure 7.18 Operation of Offset Addition

In figure 7.18, the offset addition is selected as the transfer source address update and increment or decrement by 1, 2, or 4 is selected as the transfer destination address. The address update means that data at the address which is away from the previous transfer source address by the offset is read from. The data read from the address away from the previous address is written to the consecutive area in the destination side.

(2) XY Conversion Using Offset

Figure 7.19 shows the XY conversion using the offset addition in repeat transfer mode.

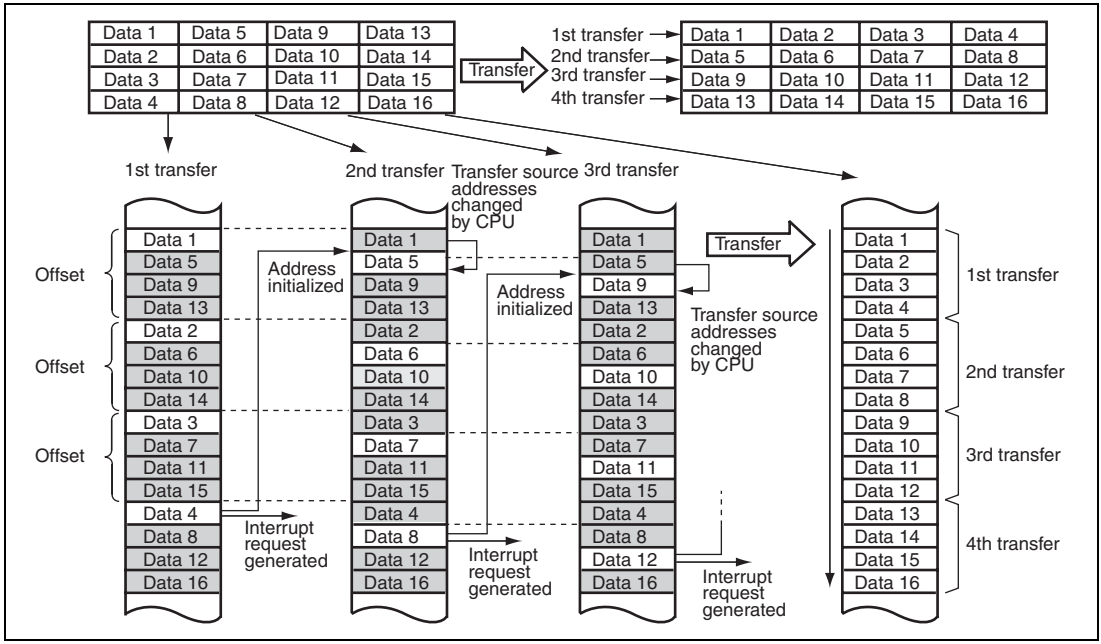


Figure 7.19 XY Conversion Operation Using Offset Addition in Repeat Transfer Mode

In figure 7.19, the source address side is specified to the repeat area by DACR and the offset addition is selected. The offset value is set to 4 × data access size (when the data access size is longword, H'00000010 is set in DOFR, as an example). The repeat size is set to 4 × data access size (when the data access size is longword, the repeat size is set to 4 × 4 = 16 bytes, as an example). The increment or decrement by 1, 2, or 4 is specified as the transfer destination address. A repeat size end interrupt is requested when the repeat size of transfers is completed.

When a transfer starts, the transfer source address is added to the offset every time data is transferred. The transfer data is written to the destination continuous addresses. When data 4 is transferred meaning that the repeat size of transfers is completed, the transfer source address returns to the transfer start address (address of data 1 on the transfer source) and a repeat size end interrupt is requested. While this interrupt stops the transfer temporarily, the contents of DSAR are written to the address of data 5 by the CPU (when the data access size is longword, write the data 1 address + 4). When the DTE bit in DMDR is set to 1, the transfer is resumed from the state when the transfer is stopped. Accordingly, operations are repeated and the transfer source data is transposed to the destination area (XY conversion).

Figure 7.20 shows a flowchart of the XY conversion.

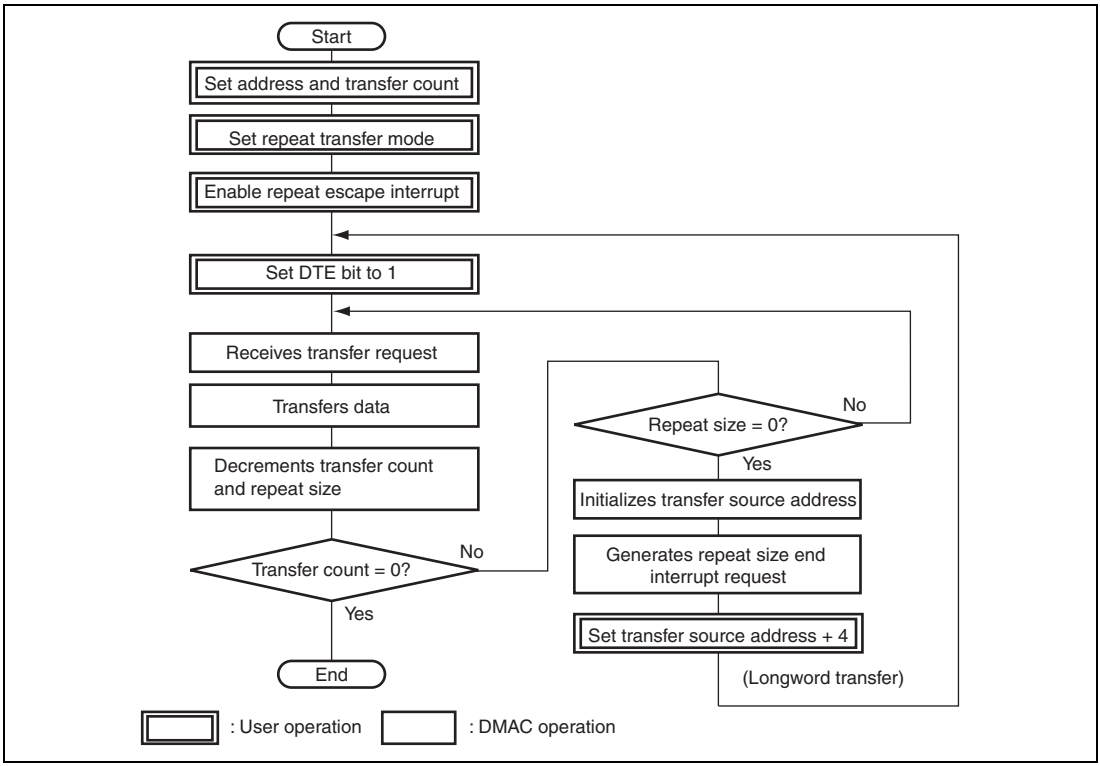


Figure 7.20 XY Conversion Flowchart Using Offset Addition in Repeat Transfer Mode

(3) Offset Subtraction

When setting the negative value in DOFR, the offset value must be 2's complement. The 2's complement is obtained by the following formula.

2's complement of offset = 1 + ~offset (~: bit inversion)

Example: 2's complement of H'0001FFFF
= H'FFFE0000 + H'00000001
= H'FFFE0001

The value of 2's complement can be obtained by the NEG.L instruction.

7.5.7 Register during DMA Transfer

The DMAC registers are updated by a DMA transfer. The value to be updated differs according to the other settings and transfer state. The registers to be updated are DSAR, DDAR, DTCR, bits BKSZH and BKSZ in DBSR, and the DTE, ACT, ERRF, ESIF, and DTIF bits in DMDR.

(1) DMA Source Address Register

When the transfer source address set in DSAR is accessed, the contents of DSAR are output and then are updated to the next address.

The increment or decrement can be specified by bits SAT1 and SAT0 in DACR. When SAT1 and SAT0 = B'00, the address is fixed. When SAT1 and SAT0 = B'01, the address is added with the offset. When SAT1 and SAT0 = B'10, the address is incremented. When SAT1 and SAT0 = B'11, the address is decremented. The size of increment or decrement depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the source address is word or longword, when the source address is not aligned with the word or longword boundary, the read bus cycle is divided into byte or word cycles. While data of one word or one longword is being read, the size of increment or decrement is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After one word or one longword of data is read, the address when the read cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the source address side, the source address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the source address side, operation follows the setting. The upper address bits are fixed and is not affected by the address update.

While data is being transferred, DSAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DSAR during the transfer may be updated regardless of the access by the CPU. Moreover, DSAR for the channel being transferred must not be written to.

(2) DMA Destination Address Register

When the transfer destination address set in DDAR is accessed, the contents of DDAR are output and then are updated to the next address.

The increment or decrement can be specified by bits DAT1 and DAT0 in DACR. When DAT1 and DAT0 = B'00, the address is fixed. When DAT1 and DAT0 = B'01, the address is added with the offset. When DAT1 and DAT0 = B'10, the address is incremented. When DAT1 and DAT0 = B'11, the address is decremented. The incrementing or decrementing size depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the destination address is word or longword, when the destination address is not aligned with the word or longword boundary, the write bus cycle is divided into byte and word cycles. While one word or one longword of data is being written, the incrementing or decrementing size is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After the one word or one longword of data is written, the address when the write cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the destination address side, the destination address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the destination address side, operation follows the setting. The upper address bits are fixed and is not affected by the address update.

While data is being transferred, DDAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DDAR during the transfer may be updated regardless of the access by the CPU. Moreover, DDAR for the channel being transferred must not be written to.

(3) DMA Transfer Count Register (DTCR)

A DMA transfer decrements the contents of DTCR by the transferred bytes. When byte data is transferred, DTCR is decremented by 1. When word data is transferred, DTCR is decremented by 2. When longword data is transferred, DTCR is decremented by 4. However, when DTCR = 0, the contents of DTCR are not changed since the number of transfers is not counted.

While data is being transferred, all the bits of DTCR may be changed. DTCR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DTCR during the transfer may be updated regardless of the access by the CPU. Moreover, DTCR for the channel being transferred must not be written to.

When a conflict occurs between the address update by DMA transfer and write access by the CPU, the CPU has priority. When a conflict occurs between change from 1, 2, or 4 to 0 in DTCR and write access by the CPU (other than 0), the CPU has priority in writing to DTCR. However, the transfer is stopped.

(4) DMA Block Size Register (DBSR)

DBSR is enabled in block or repeat transfer mode. Bits 31 to 16 in DBSR function as BKSZH and bits 15 to 0 in DBSR function as BKSZ. The BKSZH bits (16 bits) store the block size and repeat size and its value is not changed. The BKSZ bits (16 bits) function as a counter for the block size and repeat size and its value is decremented every transfer by 1. When the BKSZ value is to change from 1 to 0 by a DMA transfer, 0 is not stored but the BKSZH value is loaded into the BKSZ bits.

Since the upper 16 bits of DBSR are not updated, DBSR can be accessed in words.

DBSR for the channel being transferred must not be written to.

(5) DTE Bit in DMDR

Although the DTE bit in DMDR enables or disables data transfer by the CPU write access, it is automatically cleared to 0 according to the DMA transfer state by the DMAC.

The conditions for clearing the DTE bit by the DMAC are as follows:

- When the total size of transfers is completed
- When a transfer is completed by a transfer size error interrupt
- When a transfer is completed by a repeat size end interrupt
- When a transfer is completed by an extended repeat area overflow interrupt
- When a transfer is stopped by an NMI interrupt
- When a transfer is stopped by and address error
- Reset state
- Hardware standby mode
- When a transfer is stopped by writing 0 to the DTE bit

Writing to the registers for the channels when the corresponding DTE bit is set to 1 is prohibited (except for the DTE bit). When changing the register settings after writing 0 to the DTE bit, confirm that the DTE bit has been cleared to 0.

Figure 7.21 show the procedure for changing the register settings for the channel being transferred.

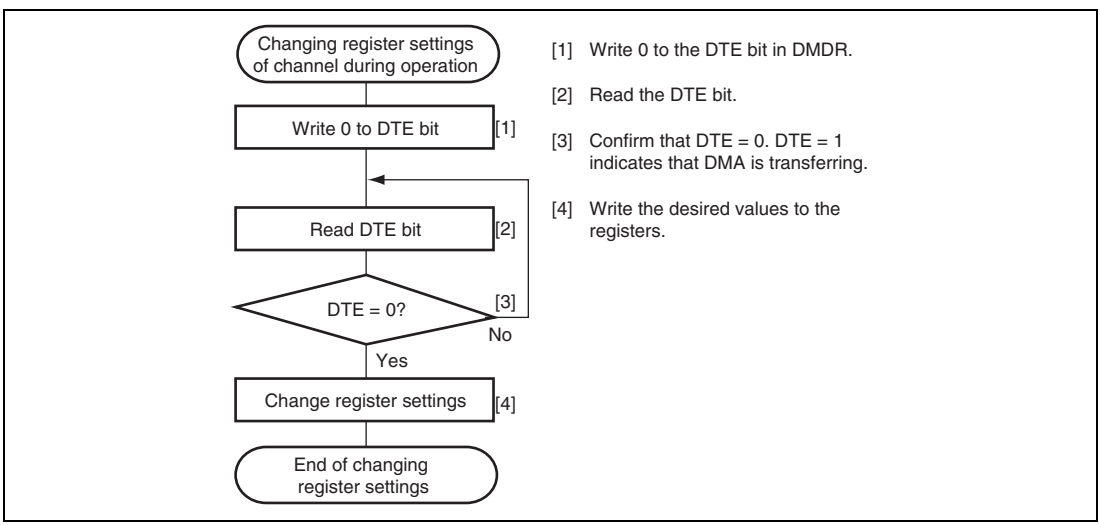


Figure 7.21 Procedure for Changing Register Setting For Channel being Transferred

(6) ACT Bit in DMDR

The ACT bit in DMDR indicates whether the DMAC is in the idle or active state. When DTE = 0 or DTE = 1 and the DMAC is waiting for a transfer request, the ACT bit is 0. Otherwise (the DMAC is in the active state), the ACT bit is 1. When individual transfers are stopped by writing 0 and the transfer is not completed, the ACT bit retains 1.

In block transfer mode, even if individual transfers are stopped by writing 0 to the DTE bit, the 1-block size of transfers is not stopped. The ACT bit retains 1 from writing 0 to the DTE bit to completion of a 1-block size transfer.

In burst mode, up to three times of DMA transfer are performed from the cycle in which the DTE bit is written to 0. The ACT bit retains 1 from writing 0 to the DTE bit to completion of DMA transfer.

(7) ERRF Bit in DMDR

When an address error or an NMI interrupt occur, the DMAC clears the DTE bits for all the channels to stop a transfer. In addition, it sets the ERRF bit in DMDR_0 to 1 to indicate that an address error or an NMI interrupt has occurred regardless of whether or not the DMAC is in operation.

(8) ESIF Bit in DMDR

When an interrupt by an transfer size error, a repeat size end, or an extended repeat area overflow is requested, the ESIF bit in DMDR is set to 1. When both the ESIF and ESIE bits are set to 1, a transfer escape interrupt is requested to the CPU or DTC.

The ESIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle of the interrupt source is completed.

The ESIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 7.8, Interrupt Sources.

(9) DTIF Bit in DMDR

The DTIF bit in DMDR is set to 1 after the total transfer size of transfers is completed. When both the DTIF and DTIE bits in DMDR are set to 1, a transfer end interrupt by the transfer counter is requested to the CPU or DTC.

The DTIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle is completed.

The DTIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 7.8, Interrupt Sources.

7.5.8 Priority of Channels

The channels of the DMAC are given following priority levels: channel 0 > channel 1 > channel 2 > channel3. Table 7.6 shows the priority levels among the DMAC channels.

Table 7.6 Priority among DMAC Channels

Channel	Priority
Channel 0	High
Channel 1	
Channel 2	
Channel 3	Low

The channel having highest priority other than the channel being transferred is selected when a transfer is requested from other channels. The selected channel starts the transfer after the channel being transferred releases the bus. At this time, when a bus master other than the DMAC requests the bus, the cycle for the bus master is inserted.

In a burst transfer or a block transfer, channels are not switched.

Figure 7.22 shows a transfer example when multiple transfer requests from channels 0 to 2.

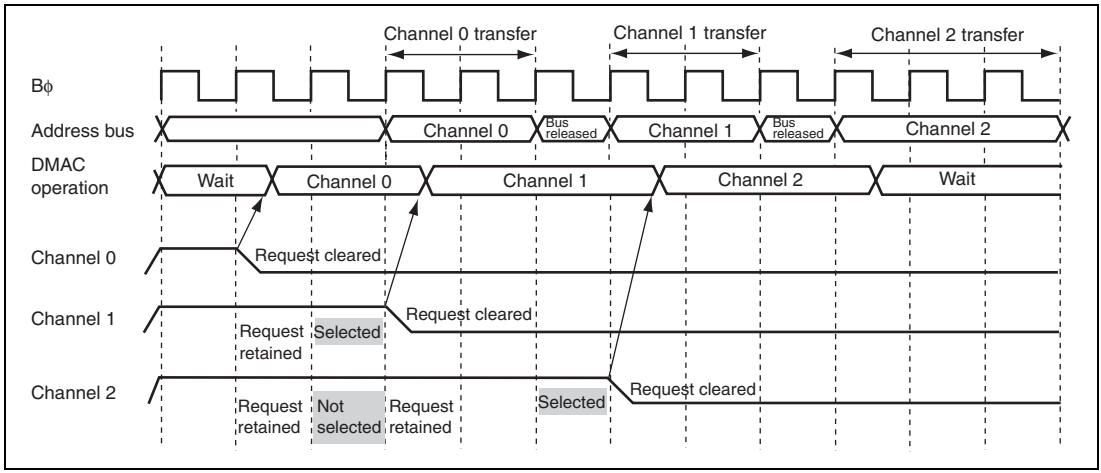


Figure 7.22 Example of Timing for Channel Priority

7.5.9 DMA Basic Bus Cycle

Figure 7.23 shows an examples of signal timing of a basic bus cycle. In figure 7.23, data is transferred in words from the 16-bit 2-state access space to the 8-bit 3-state access space. When the bus mastership is passed from the DMAC to the CPU, data is read from the source address and it is written to the destination address. The bus is not released between the read and write cycles by other bus requests. DMAC bus cycles follows the bus controller settings.

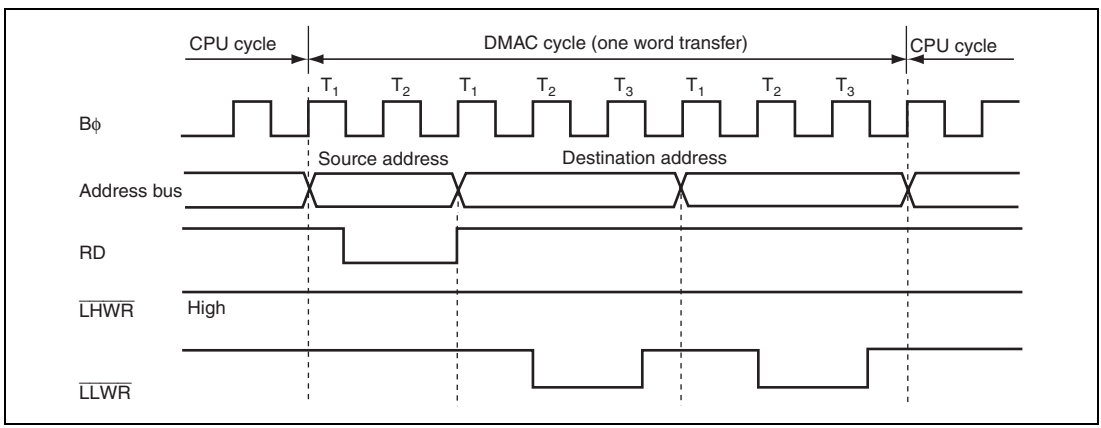


Figure 7.23 Example of Bus Timing of DMA Transfer

7.5.10 Bus Cycles in Dual Address Mode

(1) Normal Transfer Mode (Cycle Stealing Mode)

In cycle stealing mode, the bus is released every time one transfer size of data (one byte, one word, or one longword) is completed. One bus cycle or more by the CPU or DTC are executed in the bus released cycles.

In figure 7.24, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by cycle stealing.

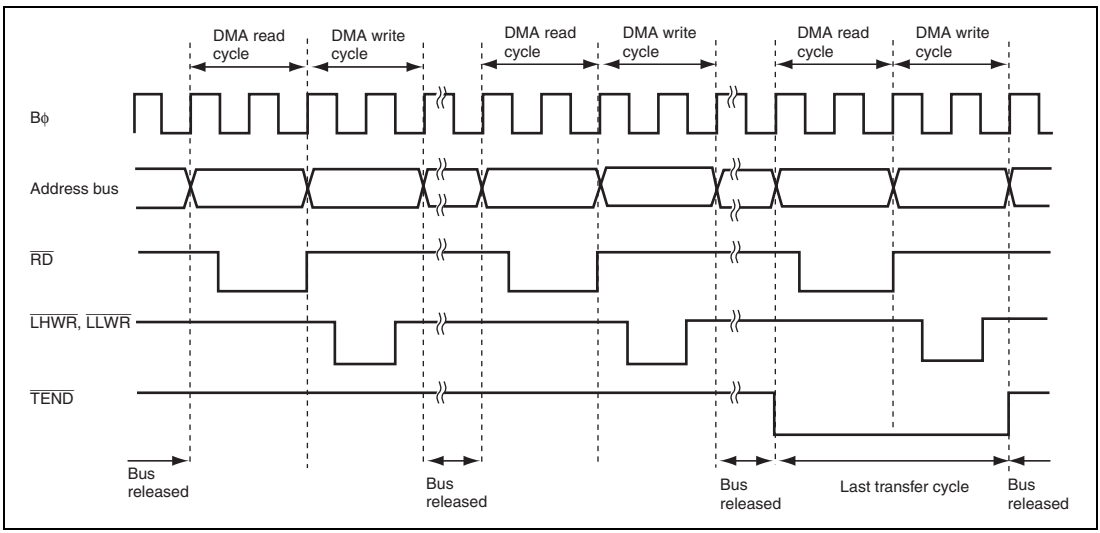


Figure 7.24 Example of Transfer in Normal Transfer Mode by Cycle Stealing

In figures 7.25 and 7.26, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in longwords from the external 16-bit 2-state access space to the 16-bit 2-state access space in normal transfer mode by cycle stealing.

In figure 7.25, the transfer source (DSAR) is not aligned with a longword boundary and the transfer destination (DDAR) is aligned with a longword boundary.

In figure 7.26, the transfer source (DSAR) is aligned with a longword boundary and the transfer destination (DDAR) is not aligned with a longword boundary.

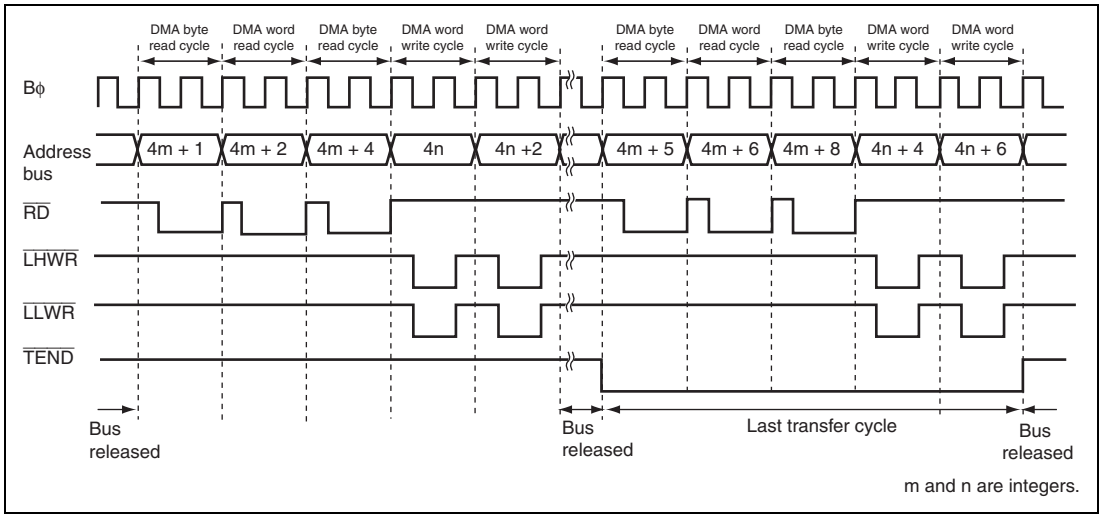


Figure 7.25 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Source DSAR = Odd Address and Source Address Increment)

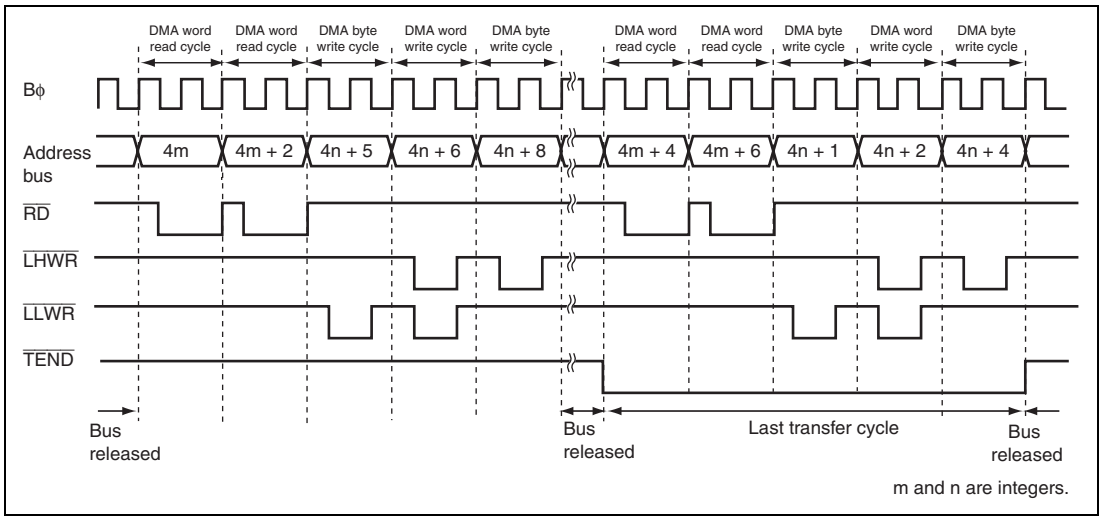


Figure 7.26 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Destination DDAR = Odd Address and Destination Address Decrement)

(2) Normal Transfer Mode (Burst Mode)

In burst mode, one byte, one word, or one longword of data continues to be transferred until the transfer end condition is satisfied.

When a burst transfer starts, a transfer request from a channel having priority is suspended until the burst transfer is completed.

In figure 7.27, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by burst access.

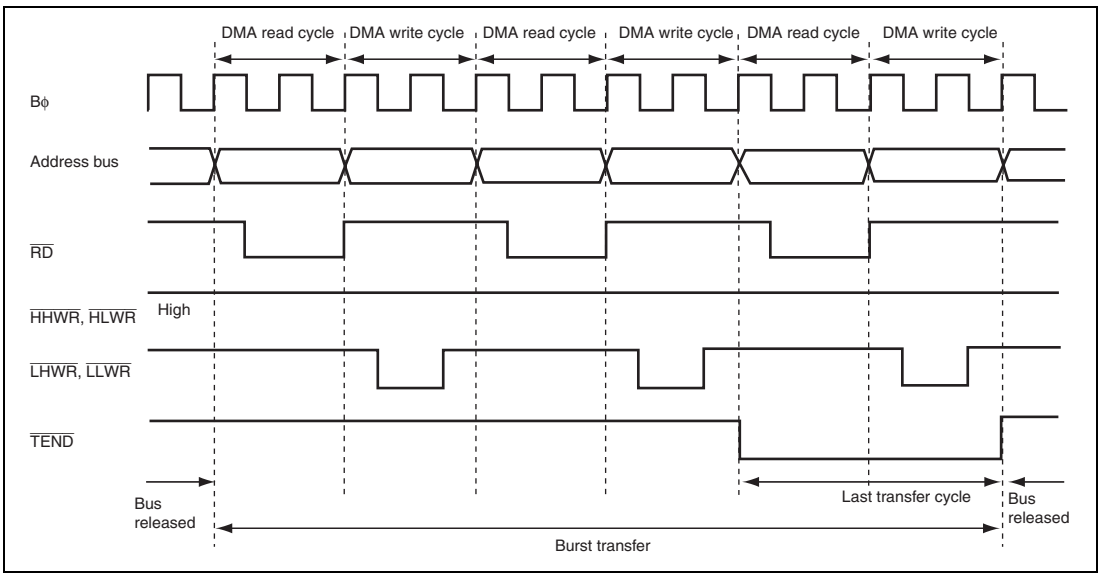


Figure 7.27 Example of Transfer in Normal Transfer Mode by Burst Access

(3) Block Transfer Mode

In block transfer mode, the bus is released every time a 1-block size of transfers at a single transfer request is completed.

In figure 7.28, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in block transfer mode.

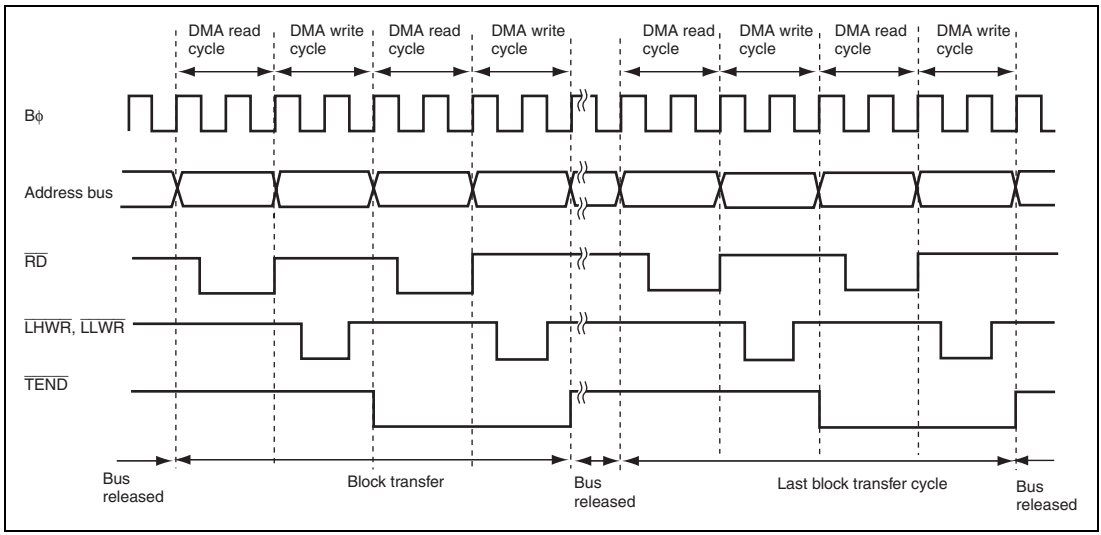


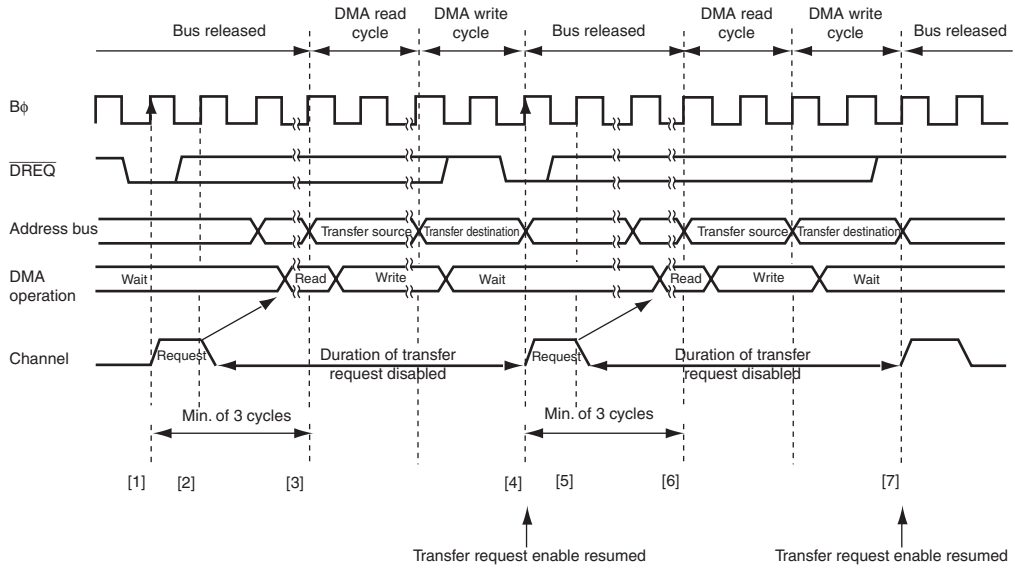
Figure 7.28 Example of Transfer in Block Transfer Mode

(4) Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.29 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the DMA write cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



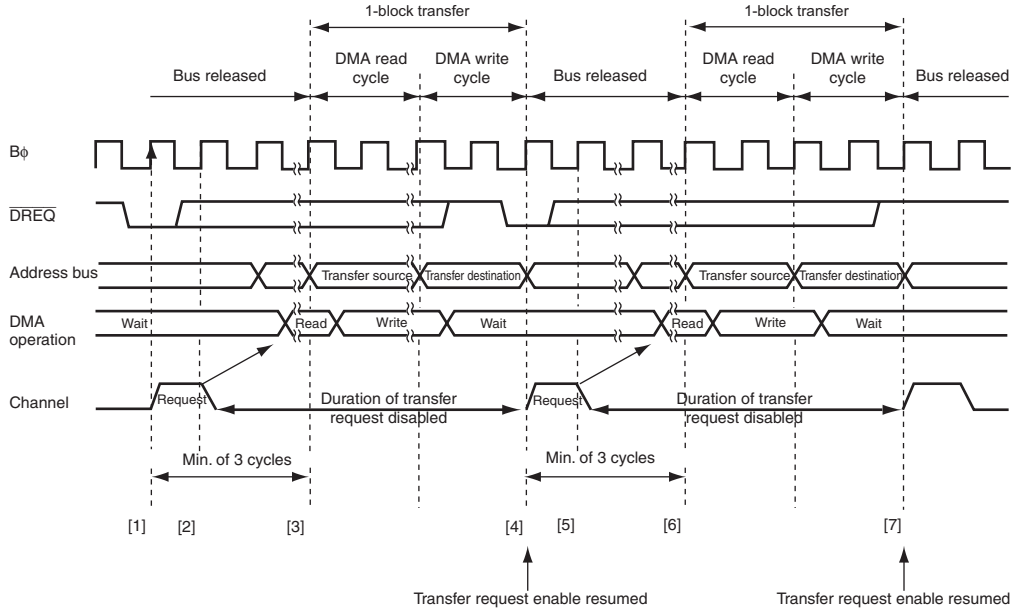
[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bϕ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.
[4][7] When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer request enable is resumed after completion of the write cycle. (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.29 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.30 shows an example of block transfer mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the DMA write cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bϕ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.
[4][7] When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer request enable is resumed after completion of the write cycle. (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

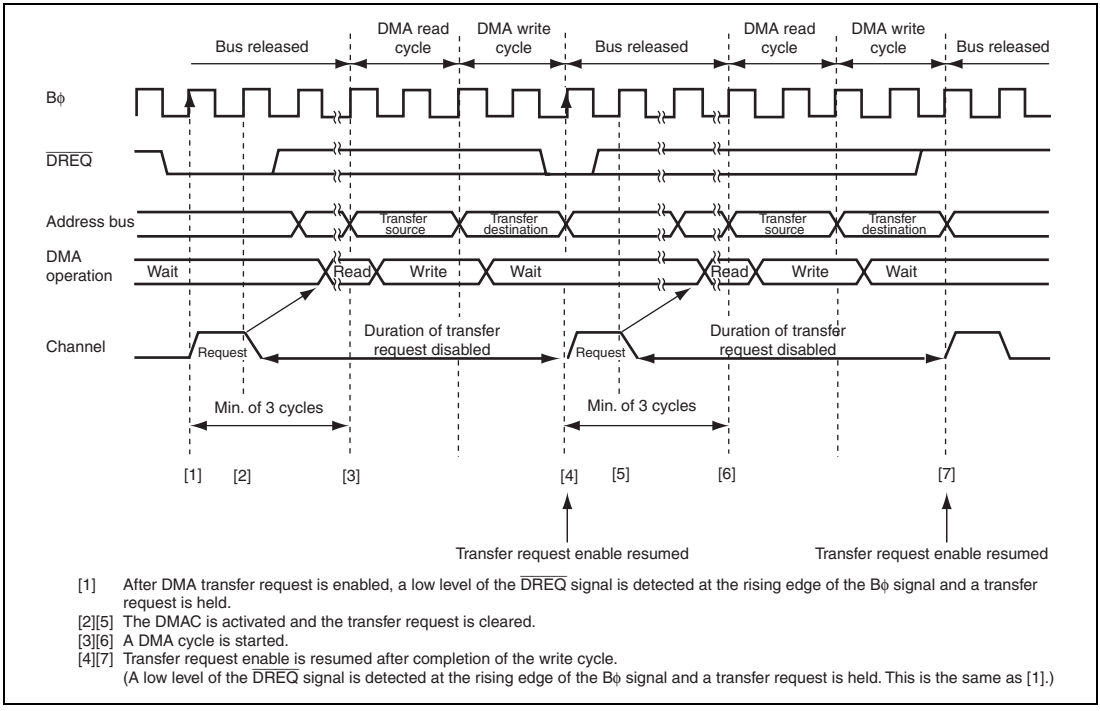
Figure 7.30 Example of Transfer in Block Transfer Mode Activated by $\overline{\text{DREQ}}$ Falling Edge

(5) Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 7.31 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



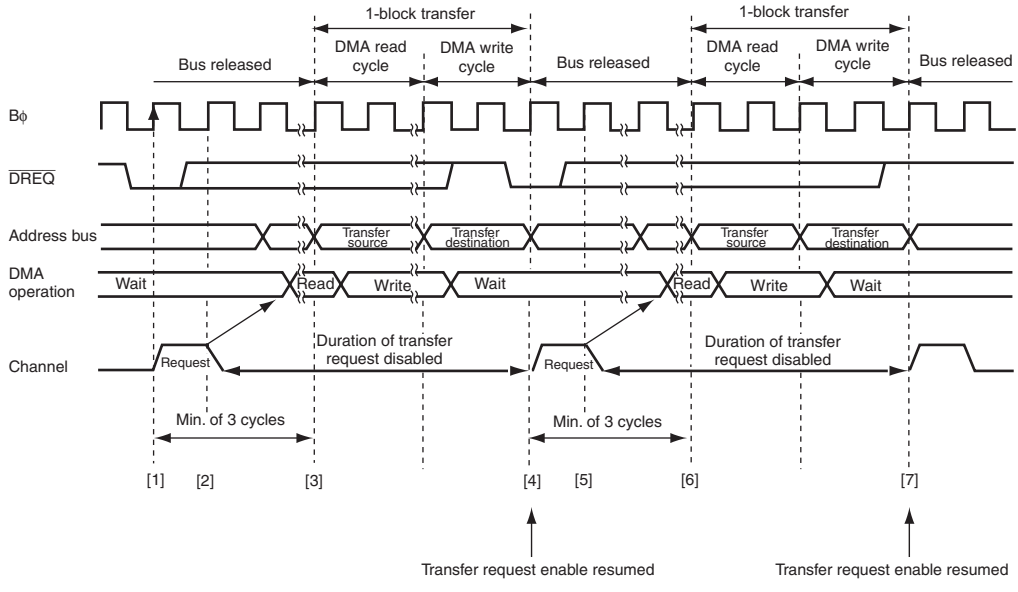
[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.31 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level

Figure 7.32 shows an example of block transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.32 Example of Transfer in Block Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level

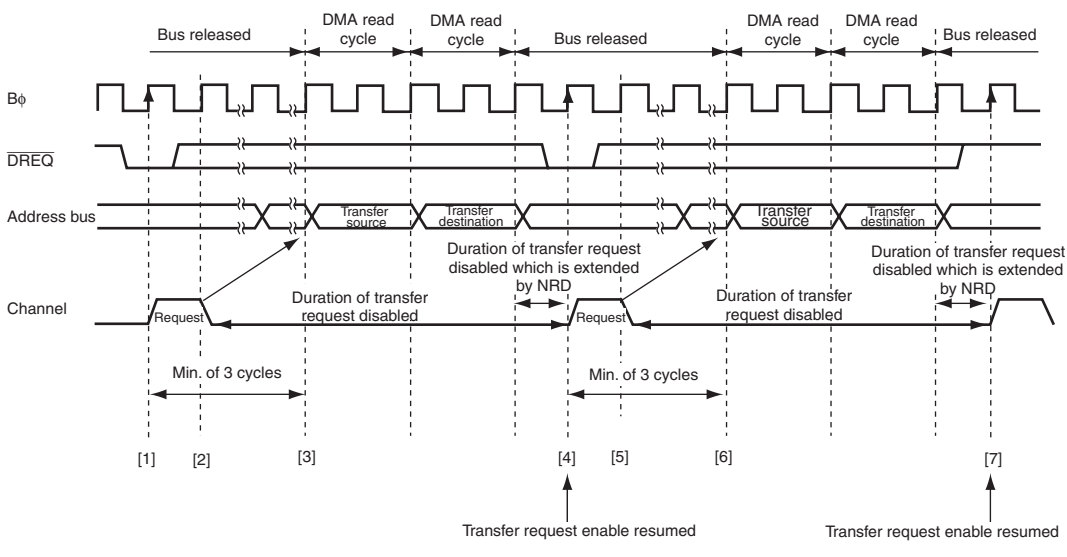
(6) Activation Timing by $\overline{\text{DREQ}}$ Low Level with NRD = 1

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 7.33 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level with NRD = 1.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed one cycle after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.33 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1

7.5.11 Bus Cycles in Single Address Mode

(1) Single Address Mode (Read and Cycle Stealing)

In single address mode, one byte, one word, or one longword of data is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU or DTC are executed in the bus released cycles.

In figure 7.34, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (read).

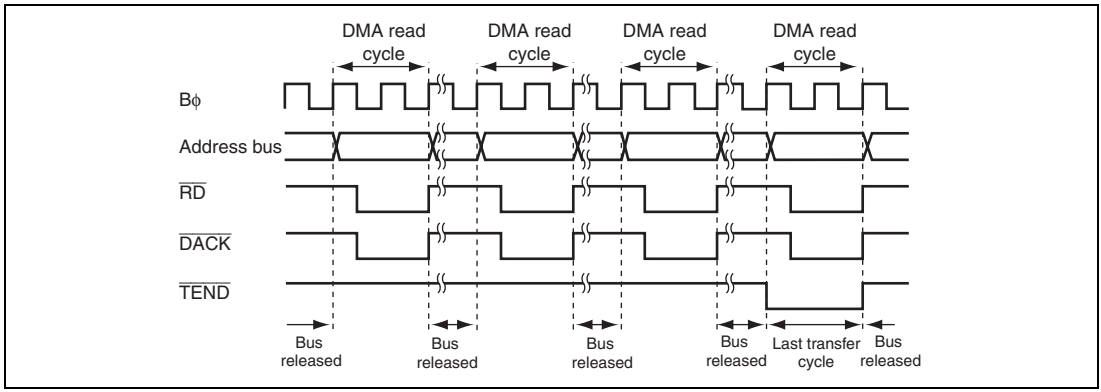


Figure 7.34 Example of Transfer in Single Address Mode (Byte Read)

(2) Single Address Mode (Write and Cycle Stealing)

In single address mode, data of one byte, one word, or one longword is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU or DTC are executed in the bus released cycles.

In figure 7.35, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (write).

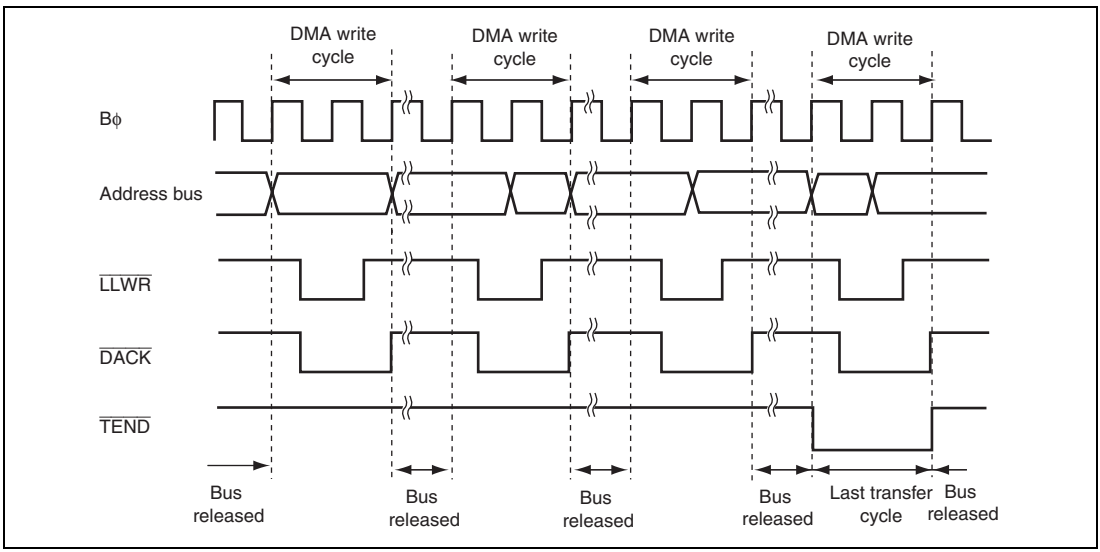


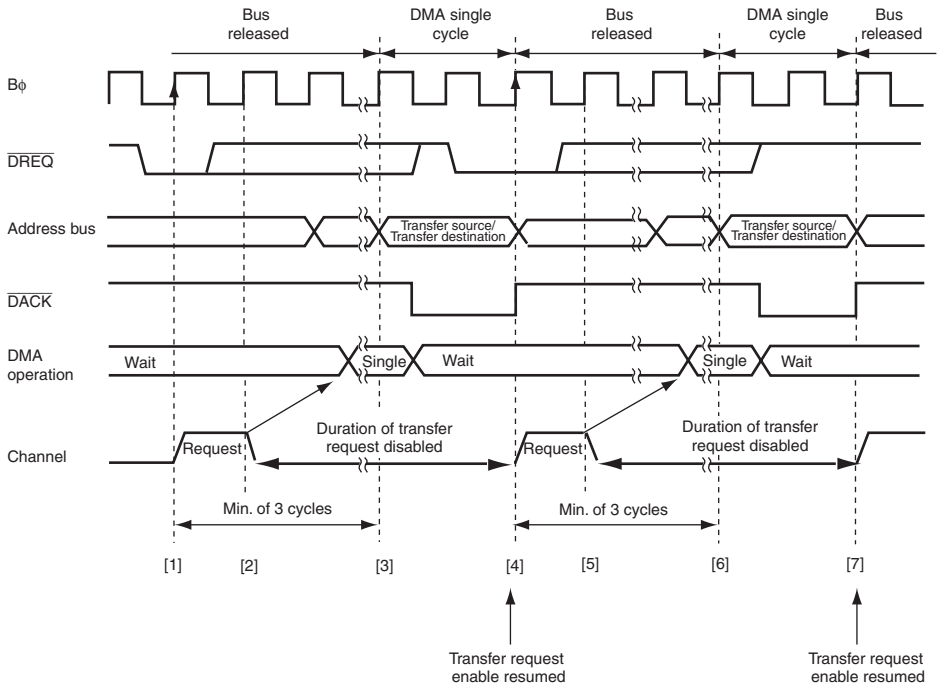
Figure 7.35 Example of Transfer in Single Address Mode (Byte Write)

(3) Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.36 shows an example of single address mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the single cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bφ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bφ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.
[4][7] When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer enable is resumed after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bφ signal and a transfer request is held. This is the same as [1].)

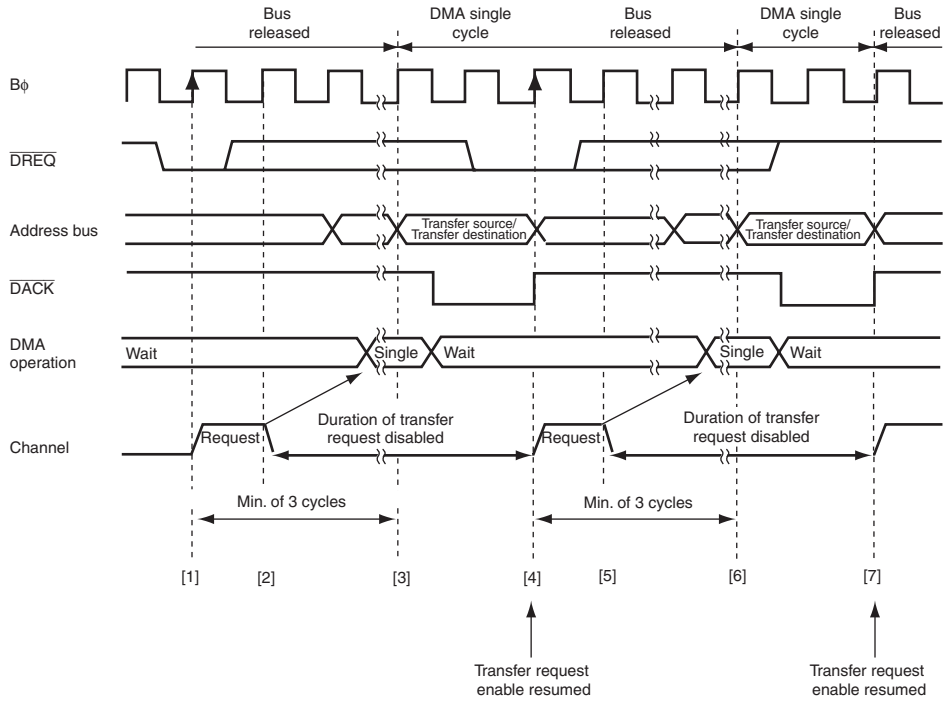
Figure 7.36 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Falling Edge

(4) Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 7.37 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the single cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the single cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.37 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level

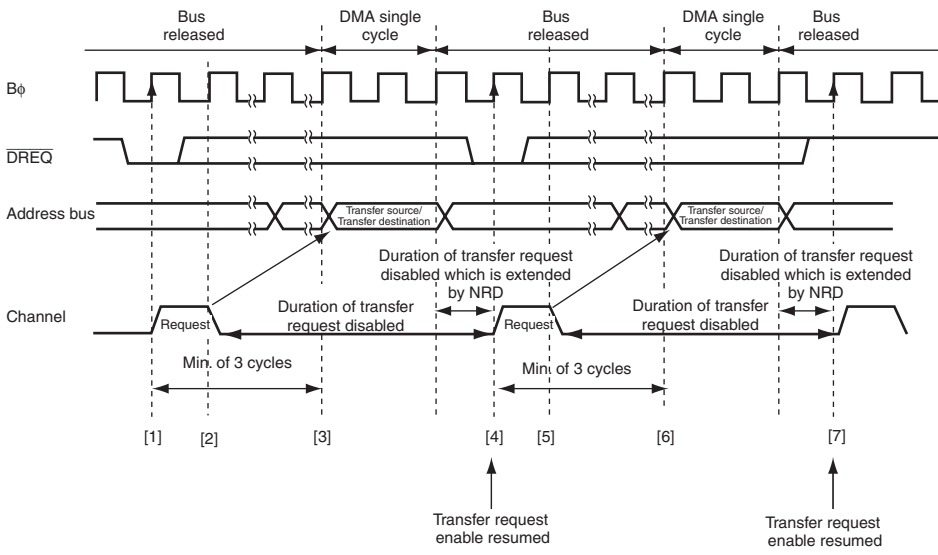
(5) Activation Timing by $\overline{\text{DREQ}}$ Low Level with NRD = 1

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 7.38 shows an example of single address mode activated by the $\overline{\text{DREQ}}$ signal low level with NRD = 1.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after one cycle of the transfer request duration inserted by NRD = 1 on completion of the single cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed one cycle after completion of the single cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.38 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1

7.6 DMA Transfer End

Operations on completion of a transfer differ according to the transfer end condition. DMA transfer completion is indicated that the DTE and ACT bits in DMDR are changed from 1 to 0.

(1) Transfer End by DTCR Change from 1, 2, or 4, to 0

When DTCR is changed from 1, 2, or 4 to 0, a DMA transfer for the channel is completed. The DTE bit in DMDR is cleared to 0 and the DTIF bit in DMDR is set to 1. At this time, when the DTIE bit in DMDR is set to 1, a transfer end interrupt by the transfer counter is requested. When the DTCR value is 0 before the transfer, the transfer is not stopped.

(2) Transfer End by Transfer Size Error Interrupt

When the following conditions are satisfied while the TSEIE bit in DMDR is set to 1, a transfer size error occurs and a DMA transfer is terminated. At this time, the DTE bit in DMR is cleared to 0 and the ESIF bit in DMDR is set to 1.

- In normal transfer mode and repeat transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the data access size
- In block transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the block size

When the TSEIE bit in DMDR is cleared to 0, data is transferred until the DTCR value reaches 0. A transfer size error is not generated. Operation in each transfer mode is shown below.

- In normal transfer mode and repeat transfer mode, when the DTCR value is less than the data access size, data is transferred in bytes
- In block transfer mode, when the DTCR value is less than the block size, the specified size of data in DTCR is transferred instead of transferring the block size of data. The transfer is performed in bytes.

(3) Transfer End by Repeat Size End Interrupt

In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit in DACR is set to 1, a repeat size end interrupt is requested. When the interrupt is requested to complete DMA transfer, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1. Under this condition, setting the DTE bit to 1 resumes the transfer.

In block transfer mode, when the next transfer is requested after completion of a 1-block size data transfer, a repeat size end interrupt can be requested.

(4) Transfer End by Interrupt on Extended Repeat Area Overflow

When an overflow on the extended repeat area occurs while the extended repeat area is specified and the SARIE or DARIE bit in DACR is set to 1, an interrupt by an extended repeat area overflow is requested. When the interrupt is requested, the DMA transfer is terminated, the DTE bit in DMDR is cleared to 0, and the ESIF bit in DMDR is set to 1.

In dual address mode, even if an interrupt by an extended repeat area overflow occurs during a read cycle, the following write cycle is performed.

In block transfer mode, even if an interrupt by an extended repeat area overflow occurs during a 1-block transfer, the remaining data is transferred. The transfer is not terminated by an extended repeat area overflow interrupt unless the current transfer is complete.

(5) Transfer End by Clearing DTE Bit in DMDR

When the DTE bit in DMDR is cleared to 0 by the CPU, a transfer is completed after the current DMA cycle and a DMA cycle in which the transfer request is accepted are completed.

In block transfer mode, a DMA transfer is completed after 1-block data is transferred.

(6) Transfer End by NMI Interrupt

When an NMI interrupt is requested, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR_0 is set to 1. When an NMI interrupt is requested during a DMA transfer, the transfer is forced to stop. To perform DMA transfer after an NMI interrupt is requested, clear the ERRF bit to 0 and then set the DTE bits for the channels to 1.

The transfer end timings after an NMI interrupt is requested are shown below.

(a) Normal Transfer Mode and Repeat Transfer Mode

In dual address mode, a DMA transfer is completed after completion of the write cycle for one transfer unit.

In single address mode, a DMA transfer is completed after completion of the bus cycle for one transfer unit.

(b) Block Transfer Mode

A DMA transfer is forced to stop. Since a 1-block size of transfers is not completed, operation is not guaranteed.

In dual address mode, the write cycle corresponding to the read cycle is performed. This is similar to (a) in normal transfer mode.

(7) Transfer End by Address Error

When an address error occurs, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR_0 is set to 1. When an address error occurs during a DMA transfer, the transfer is forced to stop. To perform a DMA transfer after an address error occurs, clear the ERRF bit to 0 and then set the DTE bits for the channels.

The transfer end timing after an address error is the same as that after an NMI interrupt.

(8) Transfer End by Hardware Standby Mode or Reset

The DMAC is initialized by a reset and a transition to the hardware standby mode. A DMA transfer is not guaranteed.

7.7 Relationship among DMAC and Other Bus Masters

7.7.1 CPU Priority Control Function Over DMAC

The CPU priority control function over DMAC can be used according to the CPU priority control register (CPUPCR) setting. For details, see section 5.7, CPU Priority Control Function Over DTC and DMAC.

The priority level of the DMAC is specified by bits DMAP2 to DMAP0 and can be specified for each channel.

The priority level of the CPU is specified by bits CPUP2 to CPUP0. The value of bits CPUP2 to CPUP0 is updated according to the exception handling priority.

If the CPU priority control is enabled by the CPUPCE bit in CPUPCR, when the CPU has priority over the DMAC, a transfer request for the corresponding channel is masked and the transfer is not activated. When another channel has priority over or the same as the CPU, a transfer request is received regardless of the priority between channels and the transfer is activated.

The transfer request masked by the CPU priority control function is suspended. When the transfer channel is given priority over the CPU by changing priority levels of the CPU or channel, the transfer request is received and the transfer is resumed. Writing 0 to the DTE bit clears the suspended transfer request.

When the CPUPCE bit is cleared to 0, it is regarded as the lowest priority.

7.7.2 Bus Arbitration among DMAC and Other Bus Masters

When DMA transfer cycles are consecutively performed, bus cycles of other bus masters may be inserted between the transfer cycles. The DMAC can release the bus temporarily to pass the bus to other bus masters.

The consecutive DMA transfer cycles may not be divided according to the transfer mode settings to achieve high-speed access.

The read and write cycles of a DMA transfer are not separated. Refreshing, external bus release, and on-chip bus master (CPU or DTC) cycles are not inserted between the read and write cycles of a DMA transfer.

In block transfer mode and an auto request transfer by burst access, bus cycles of the DMA transfer are consecutively performed. For this duration, since the DMAC has priority over the CPU and DTC, accesses to the external space is suspended (the IBCCS bit in the bus control register 2 (BCR2) is cleared to 0).

When the bus is passed to another channel or an auto request transfer by cycle stealing, bus cycles of the DMAC and on-chip bus master are performed alternatively.

When the arbitration function among the DMAC and on-chip bus masters is enabled by setting the IBCCS bit in BCR2, the bus is used alternatively except the bus cycles which are not separated. For details, see section 6, Bus Controller (BSC).

A conflict may occur between external space access of the DMAC and an external bus release cycle. Even if a burst or block transfer is performed by the DMAC, the transfer is stopped temporarily and a cycle of external bus release is inserted by the BSC according to the external bus priority (when the CPU external access and the DTC external access do not have priority over a DMAC transfer, the transfers are not operated until the DMAC releases the bus).

In dual address mode, the DMAC releases the external bus after the external space write cycle. Since the read and write cycles are not separated, the bus is not released.

An internal space (on-chip memory and internal I/O registers) access of the DMAC and an external bus release cycle may be performed at the same time.

7.8 Interrupt Sources

The DMAC interrupt sources are a transfer end interrupt by the transfer counter and a transfer escape end interrupt which is generated when a transfer is terminated before the transfer counter reaches 0. Table 7.7 shows interrupt sources and priority.

Table 7.7 Interrupt Sources and Priority

Abbr.	Interrupt Sources	Priority
DMTEND0	Transfer end interrupt by channel 0 transfer counter	High
DMTEND1	Transfer end interrupt by channel 1 transfer counter	
DMTEND2	Transfer end interrupt by channel 2 transfer counter	
DMTEND3	Transfer end interrupt by channel 3 transfer counter	
DMEEND0	Interrupt by channel 0 transfer size error	
	Interrupt by channel 0 repeat size end	
	Interrupt by channel 0 extended repeat area overflow on source address	
	Interrupt by channel 0 extended repeat area overflow on destination address	
DMEEND1	Interrupt by channel 1 transfer size error	
	Interrupt by channel 1 repeat size end	
	Interrupt by channel 1 extended repeat area overflow on source address	
	Interrupt by channel 1 extended repeat area overflow on destination address	
DMEEND2	Interrupt by channel 2 transfer size error	
	Interrupt by channel 2 repeat size end	
	Interrupt by channel 2 extended repeat area overflow on source address	
	Interrupt by channel 2 extended repeat area overflow on destination address	
DMEEND3	Interrupt by channel 3 transfer size error	
	Interrupt by channel 3 repeat size end	
	Interrupt by channel 3 extended repeat area overflow on source address	
	Interrupt by channel 3 extended repeat area overflow on destination address	Low

Each interrupt is enabled or disabled by the DTIE and ESIE bits in DMDR for the corresponding channel. A DMTEND interrupt is generated by the combination of the DTIF and DTIE bits in DMDR. A DMEEND interrupt is generated by the combination of the ESIF and ESIE bits in DMDR. The DMEEND interrupt sources are not distinguished. The priority among channels are decided by the interrupt controller and it is shown in table 7.7. For details, see section 5, Interrupt Controller.

Each interrupt source is specified by the interrupt enable bit in the register for the corresponding channel. A transfer end interrupt by the transfer counter, a transfer size error interrupt, a repeat size end interrupt, an interrupt by an extended repeat area overflow on the source address, and an interrupt by an extended repeat area overflow on the destination address are enabled or disabled by the DTIE bit in DMDR, the TSEIE bit in DMDR, the RPTIE bit in DACR, SARIE bit in DACR, and the DARIE bit in DACR, respectively.

A transfer end interrupt by the transfer counter is generated when the DTIF bit in DMDR is set to 1. The DTIF bit is set to 1 when DTCR becomes 0 by a transfer while the DTIE bit in DMDR is set to 1.

An interrupt other than the transfer end interrupt by the transfer counter is generated when the ESIF bit in DMDR is set to 1. The ESIF bit is set to 1 when the conditions are satisfied by a transfer while the enable bit is set to 1.

A transfer size error interrupt is generated when the next transfer cannot be performed because the DTCR value is less than the data access size, meaning that the data access size of transfers cannot be performed. In block transfer mode, the block size is compared with the DTCR value for transfer error decision.

A repeat size end interrupt is generated when the next transfer is requested after completion of the repeat size of transfers in repeat transfer mode. Even when the repeat area is not specified in the address register, the transfer can be stopped periodically according to the repeat size. At this time, when a transfer end interrupt by the transfer counter is generated, the ESIF bit is set to 1.

An interrupt by an extended repeat area overflow on the source and destination addresses is generated when the address exceeds the extended repeat area (overflow). At this time, when a transfer end interrupt by the transfer counter, the ESIF bit is set to 1.

Figure 7.39 is a block diagram of interrupts and interrupt flags. To clear an interrupt, clear the DTIF or ESIF bit in DMDR to 0 in the interrupt handling routine or continue the transfer by setting the DTE bit in DMDR after setting the register. Figure 7.40 shows procedure to resume the transfer by clearing a interrupt.

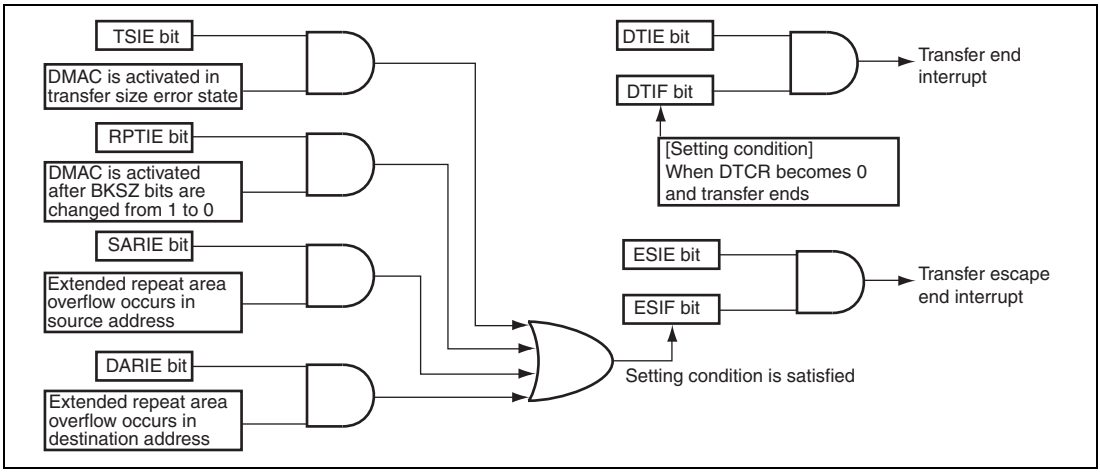
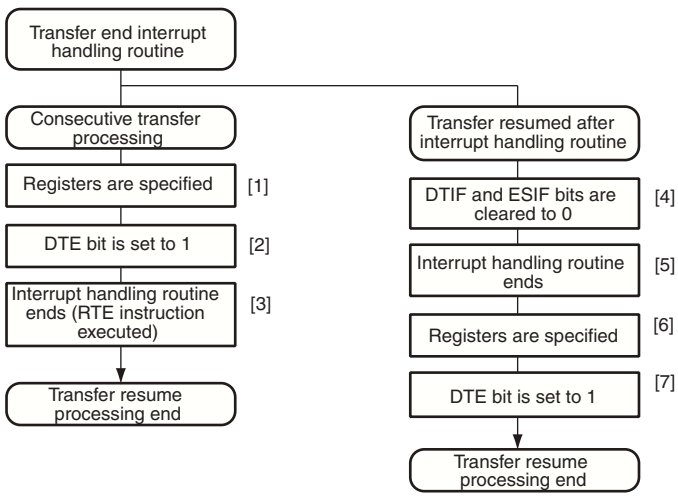


Figure 7.39 Interrupt and Interrupt Sources



[1] Specify the values in the registers such as transfer counter and address register.
[2] Set the DTE bit in DMDR to 1 to resume DMA operation. Setting the DTE bit to 1 automatically clears the DTIF or ESIF bit in DMDR to 0 and an interrupt source is cleared.
[3] End the interrupt handling routine by the RTE instruction.
[4] Read that the DTIF or the ESIF bit in DMDR = 1 and then write 0 to the bit.
[5] Complete the interrupt handling routine and clear the interrupt mask.
[6] Specify the values in the registers such as transfer counter and address register.
[7] Set the DTE bit to 1 to resume DMA operation.

Figure 7.40 Procedure Example of Resuming Transfer by Clearing Interrupt Source

7.9 Notes on Usage

1. DMAC Register Access During Operation

   Except for clearing the DTE bit in DMDR, the settings for channels being transferred (including waiting state) must not be changed. The register settings must be changed during the transfer prohibited state.

2. Settings of Module Stop Function

   The DMAC operation can be enabled or disabled by the module stop control register. The DMAC is enabled by the initial value.

   Setting bit MSTPA13 in MSTPCRA stops the clock supplied to the DMAC and the DMAC enters the module stop state. However, when a transfer for a channel is enabled or when an interrupt is being requested, bit MSTPA13 cannot be set to 1. Clear the DTE bit to 0, clear the DTIF or DTIE bit in DMDR to 0, and then set bit MSTPA13.

   When the clock is stopped, the DMAC registers cannot be accessed. However, the following register settings are valid in the module stop state. Disable them before entering the module stop state, if necessary.

   — TENDE bit in DMDR is 1 (the TEND signal output enabled)

   — DACKE bit in DMDR is 1 (the DACK signal output enabled)

3. Activation by $\overline{\text{DREQ}}$ Falling Edge

   The $\overline{\text{DREQ}}$ falling edge detection is synchronized with the DMAC internal operation.

   A. Activation request waiting state: Waiting for detecting the $\overline{\text{DREQ}}$ low level. A transition to 2. is made.

   B. Transfer waiting state: Waiting for a DMAC transfer. A transition to 3. is made.

   C. Transfer prohibited state: Waiting for detecting the $\overline{\text{DREQ}}$ high level. A transition to 1. is made.

   After a DMAC transfer enabled, a transition to 1. is made. Therefore, the $\overline{\text{DREQ}}$ signal is sampled by low level detection at the first activation after a DMAC transfer enabled.

4. Acceptation of Activation Source

   At the beginning of an activation source reception, a low level is detected regardless of the setting of $\overline{\text{DREQ}}$ falling edge or low level detection. Therefore, if the $\overline{\text{DREQ}}$ signal is driven low before setting DMDR, the low level is received as a transfer request.

   When the DMAC is activated, clear the $\overline{\text{DREQ}}$ signal of the previous transfer.

Section 8 Data Transfer Controller (DTC)

This LSI includes a data transfer controller (DTC). The DTC can be activated to transfer data by an interrupt request.

8.1 Features

- Transfer possible over any number of channels:
  Multiple data transfer enabled for one activation source (chain transfer)
  Chain transfer specifiable after data transfer (when the counter is 0)
- Three transfer modes
  Normal/repeat/block transfer modes selectable
  Transfer source and destination addresses can be selected from increment/decrement/fixed
- Short address mode or full address mode selectable
  - Short address mode
    Transfer information is located on a 3-longword boundary
    The transfer source and destination addresses can be specified by 24 bits to select a 16-Mbyte address space directly
  - Full address mode
    Transfer information is located on a 4-longword boundary
    The transfer source and destination addresses can be specified by 32 bits to select a 4-Gbyte address space directly
- Size of data for data transfer can be specified as byte, word, or longword
  The bus cycle is divided if an odd address is specified for a word or longword transfer.
  The bus cycle is divided if address 4n + 2 is specified for a longword transfer.
- A CPU interrupt can be requested for the interrupt that activated the DTC
  A CPU interrupt can be requested after one data transfer completion
  A CPU interrupt can be requested after the specified data transfer completion
- Read skip of the transfer information specifiable
- Writeback skip executed for the fixed transfer source and destination addresses
- Module stop function specifiable

Figure 8.1 shows a block diagram of the DTC. The DTC transfer information can be allocated to the data area*. When the transfer information is allocated to the on-chip RAM, a 32-bit bus connects the DTC to the on-chip RAM, enabling 32-bit/1-state reading and writing of the DTC transfer information.

Note: * When the transfer information is stored in the on-chip RAM, the RAME bit in SYSCR must be set to 1.

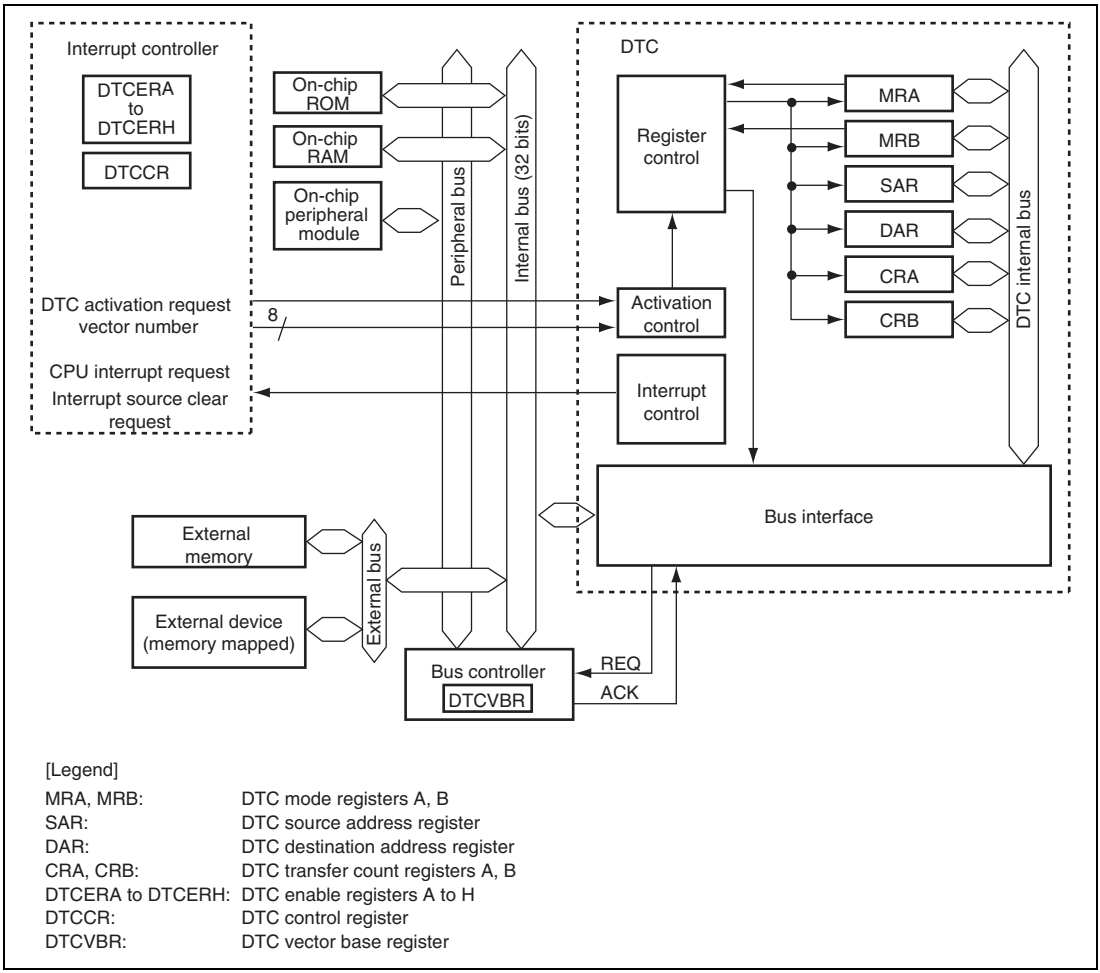


Figure 8.1 Block Diagram of DTC

8.2 Register Descriptions

DTC has the following registers.

- DTC mode register A (MRA)
- DTC mode register B (MRB)
- DTC source address register (SAR)
- DTC destination address register (DAR)
- DTC transfer count register A (CRA)
- DTC transfer count register B (CRB)

These six registers MRA, MRB, SAR, DAR, CRA, and CRB cannot be directly accessed by the CPU. The contents of these registers are stored in the data area as transfer information. When a DTC activation request occurs, the DTC reads a start address of transfer information that is stored in the data area according to the vector address, reads the transfer information, and transfers data. After the data transfer, it writes a set of updated transfer information back to the data area.

- DTC enable registers A to H (DTCERA to DTCERH)
- DTC control register (DTCCR)
- DTC vector base register (DTCVBR)

8.2.1 DTC Mode Register A (MRA)

MRA selects DTC operating mode. MRA cannot be accessed directly by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name	MD1	MD0	Sz1	Sz0	SM1	SM0	—	—
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	—	—	—	—	—	—	—	—

Bit	Bit Name	Initial Value	R/W	Description
7	MD1	Undefined	—	DTC Mode 1 and 0
6	MD0	Undefined	—	Specify DTC transfer mode. 00: Normal mode 01: Repeat mode 10: Block transfer mode 11: Setting prohibited
5	Sz1	Undefined	—	DTC Data Transfer Size 1 and 0
4	Sz0	Undefined	—	Specify the size of data to be transferred. 00: Byte-size transfer 01: Word-size transfer 10: Longword-size transfer 11: Setting prohibited
3	SM1	Undefined	—	Source Address Mode 1 and 0
2	SM0	Undefined	—	Specify an SAR operation after a data transfer. 0x: SAR is fixed (SAR writeback is skipped) 10: SAR is incremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10) 11: SAR is decremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10)

Bit	Bit Name	Initial Value	R/W	Description
1, 0	—	Undefined	—	Reserved The write value should always be 0.

[Legend]
X: Don't care

8.2.2 DTC Mode Register B (MRB)

MRB selects DTC operating mode. MRB cannot be accessed directly by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name	CHNE	CHNS	DISEL	DTS	DM1	DM0	—	—
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	—	—	—	—	—	—	—	—

Bit	Bit Name	Initial Value	R/W	Description
7	CHNE	Undefined	—	DTC Chain Transfer Enable Specifies the chain transfer. For details, see section 8.5.7, Chain Transfer. The chain transfer condition is selected by the CHNS bit. 0: Disables the chain transfer 1: Enables the chain transfer
6	CHNS	Undefined	—	DTC Chain Transfer Select Specifies the chain transfer condition. If the following transfer is a chain transfer, the completion check of the specified transfer count is not performed and activation source flag or DTCER is not cleared. 0: Chain transfer every time 1: Chain transfer only when transfer counter = 0

Bit	Bit Name	Initial Value	R/W	Description
5	DISEL	Undefined	—	DTC Interrupt Select When this bit is set to 1, a CPU interrupt request is generated every time after a data transfer ends. When this bit is set to 0, a CPU interrupt request is only generated when the specified number of data transfer ends.
4	DTS	Undefined	—	DTC Transfer Mode Select Specifies either the source or destination as repeat or block area during repeat or block transfer mode. 0: Specifies the destination as repeat or block area 1: Specifies the source as repeat or block area
3	DM1	Undefined	—	Destination Address Mode 1 and 0 Specify a DAR operation after a data transfer. 0X: DAR is fixed (DAR writeback is skipped) 10: DAR is incremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10) 11: SAR is decremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10)
2	DM0	Undefined	—	
1, 0	—	Undefined	—	Reserved The write value should always be 0.

[Legend]
X: Don't care

8.2.3 DTC Source Address Register (SAR)

SAR is a 32-bit register that designates the source address of data to be transferred by the DTC.

In full address mode, 32 bits of SAR are valid. In short address mode, the lower 24 bits of SAR is valid and bits 31 to 24 are ignored. At this time, the upper eight bits are filled with the value of bit 23.

If a word or longword access is performed while an odd address is specified in SAR or if a longword access is performed while address 4n + 2 is specified in SAR, the bus cycle is divided into multiple cycles to transfer data. For details, see section 8.5.1, Bus Cycle Division.

SAR cannot be accessed directly from the CPU.

8.2.4 DTC Destination Address Register (DAR)

DAR is a 32-bit register that designates the destination address of data to be transferred by the DTC.

In full address mode, 32 bits of DAR are valid. In short address mode, the lower 24 bits of DAR is valid and bits 31 to 24 are ignored. At this time, the upper eight bits are filled with the value of bit 23.

If a word or longword access is performed while an odd address is specified in DAR or if a longword access is performed while address 4n + 2 is specified in DAR, the bus cycle is divided into multiple cycles to transfer data. For details, see section 8.5.1, Bus Cycle Division.

DAR cannot be accessed directly from the CPU.

8.2.5 DTC Transfer Count Register A (CRA)

CRA is a 16-bit register that designates the number of times data is to be transferred by the DTC.

In normal transfer mode, CRA functions as a 16-bit transfer counter (1 to 65,536). It is decremented by 1 every time data is transferred, and bit DTCEn (n = 15 to 0) corresponding to the activation source is cleared and then an interrupt is requested to the CPU when the count reaches H'0000. The transfer count is 1 when CRA = H'0001, 65,535 when CRA = H'FFFF, and 65,536 when CRA = H'0000.

In repeat transfer mode, CRA is divided into two parts: the upper eight bits (CRAH) and the lower eight bits (CRAL). CRAH holds the number of transfers while CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are sent to CRAL when the count reaches H'00. The transfer count is 1 when CRAH = CRAL = H'01, 255 when CRAH = CRAL = H'FF, and 256 when CRAH = CRAL = H'00.

In block transfer mode, CRA is divided into two parts: the upper eight bits (CRAH) and the lower eight bits (CRAL). CRAH holds the block size while CRAL functions as an 8-bit block-size counter (1 to 256 for byte, word, or longword). CRAL is decremented by 1 every time a byte (word or longword) data is transferred, and the contents of CRAH are sent to CRAL when the count reaches H'00. The block size is 1 byte (word or longword) when CRAH = CRAL =H'01, 255 bytes (words or longwords) when CRAH = CRAL = H'FF, and 256 bytes (words or longwords) when CRAH = CRAL =H'00.

CRA cannot be accessed directly from the CPU.

8.2.6 DTC Transfer Count Register B (CRB)

CRB is a 16-bit register that designates the number of times data is to be transferred by the DTC in block transfer mode. It functions as a 16-bit transfer counter (1 to 65,536) that is decremented by 1 every time data is transferred, and bit DTCEn (n = 15 to 0) corresponding to the activation source is cleared and then an interrupt is requested to the CPU when the count reaches H'0000. The transfer count is 1 when CRB = H'0001, 65,535 when CRB = H'FFFF, and 65,536 when CRB = H'0000.

CRB is not available in normal and repeat modes and cannot be accessed directly by the CPU.

8.2.7 DTC Enable Registers A to H (DTCERA to DTCERE)

DTCER which is comprised of eight registers, DTCERA to DTCERE, is a register that specifies DTC activation interrupt sources. The correspondence between interrupt sources and DTCE bits is shown in table 8.1. Use bit manipulation instructions such as BSET and BCLR to read or write a DTCE bit. If all interrupts are masked, multiple activation sources can be set at one time (only at the initial setting) by writing data after executing a dummy read on the relevant register.

Bit	15	14	13	12	11	10	9	8
Bit Name	DTCE15	DTCE14	DTCE13	DTCE12	DTCE11	DTCE10	DTCE9	DTCE8
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
Bit Name	DTCE7	DTCE6	DTCE5	DTCE4	DTCE3	DTCE2	DTCE1	DTCE0
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	Bit Name	Initial Value	R/W	Description
15	DTCE15	0	R/W	DTC Activation Enable 15 to 0 Setting this bit to 1 specifies a relevant interrupt source to a DTC activation source. [Clearing conditions] • When writing 0 to the bit to be cleared after reading 1 • When the DISEL bit is 1 and the data transfer has ended • When the specified number of transfers have ended These bits are not cleared when the DISEL bit is 0 and the specified number of transfers have not ended
14	DTCE14	0	R/W	
13	DTCE13	0	R/W	
12	DTCE12	0	R/W	
11	DTCE11	0	R/W	
10	DTCE10	0	R/W	
9	DTCE9	0	R/W	
8	DTCE8	0	R/W	
7	DTCE7	0	R/W	
6	DTCE6	0	R/W	
5	DTCE5	0	R/W	
4	DTCE4	0	R/W	
3	DTCE3	0	R/W	
2	DTCE2	0	R/W	
1	DTCE1	0	R/W	
0	DTCE0	0	R/W	

8.2.8 DTC Control Register (DTCCR)

DTCCR specifies transfer information read skip.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	RRS	RCHNE	—	—	ERR
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/(W)*

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
4	RRS	0	R/W	DTC Transfer Information Read Skip Enable Controls the vector address read and transfer information read. A DTC vector number is always compared with the vector number for the previous activation. If the vector numbers match and this bit is set to 1, the DTC data transfer is started without reading a vector address and transfer information. If the previous DTC activation is a chain transfer, the vector address read and transfer information read are always performed. 0: Transfer read skip is not performed. 1: Transfer read skip is performed when the vector numbers match.
3	RCHNE	0	R/W	Chain Transfer Enable After DTC Repeat Transfer Enables/disables the chain transfer while transfer counter (CRAL) is 0 in repeat transfer mode. In repeat transfer mode, the CRAH value is written to CRAL when CRAL is 0. Accordingly, chain transfer may not occur when CRAL is 0. If this bit is set to 1, the chain transfer is enabled when CRAH is written to CRAL. 0: Disables the chain transfer after repeat transfer 1: Enables the chain transfer after repeat transfer
2, 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
0	ERR	0	R/(W)*	Transfer Stop Flag Indicates that an address error or an NMI interrupt occurs. If an address error or an NMI interrupt occurs, the DTC stops. 0: No interrupt occurs 1: An interrupt occurs [Clearing condition] • When writing 0 after reading 1

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

8.2.9 DTC Vector Base Register (DTCVBR)

DTCVBR is a 32-bit register that specifies the base address for vector table address calculation. Bits 31 to 28 and bits 11 to 0 are fixed 0 and cannot be written to. The initial value of DTCVBR is H'00000000.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Bit Name																
Initial Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Name																
Initial Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R	R	R	R	R	R

8.3 Activation Sources

The DTC is activated by an interrupt request. The interrupt source is selected by DTCER. A DTC activation source can be selected by setting the corresponding bit in DTCER; the CPU interrupt source can be selected by clearing the corresponding bit in DTCER. At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the activation source interrupt flag or corresponding DTCER bit is cleared.

8.4 Location of Transfer Information and DTC Vector Table

Locate the transfer information in the data area. The start address of transfer information should be located at the address that is a multiple of four (4n). Otherwise, the lower two bits are ignored during access ([1:0] = B'00.) Transfer information can be located in either short address mode (three longwords) or full address mode (four longwords). The DTCMD bit in SYSCR specifies either short address mode (DTCMD = 1) or full address mode (DTCMD = 0). For details, see section 3.2.2, System Control Register (SYSCR). Transfer information located in the data area is shown in figure 8.2

The DTC reads the start address of transfer information from the vector table according to the activation source, and then reads the transfer information from the start address. Figure 8.3 shows correspondences between the DTC vector address and transfer information.

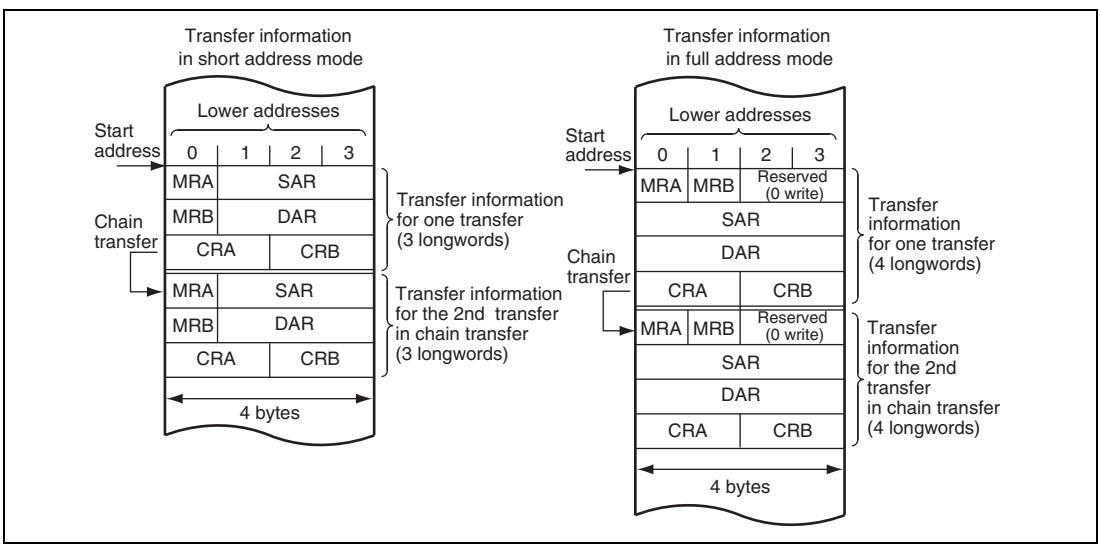


Figure 8.2 Transfer Information on Data Area

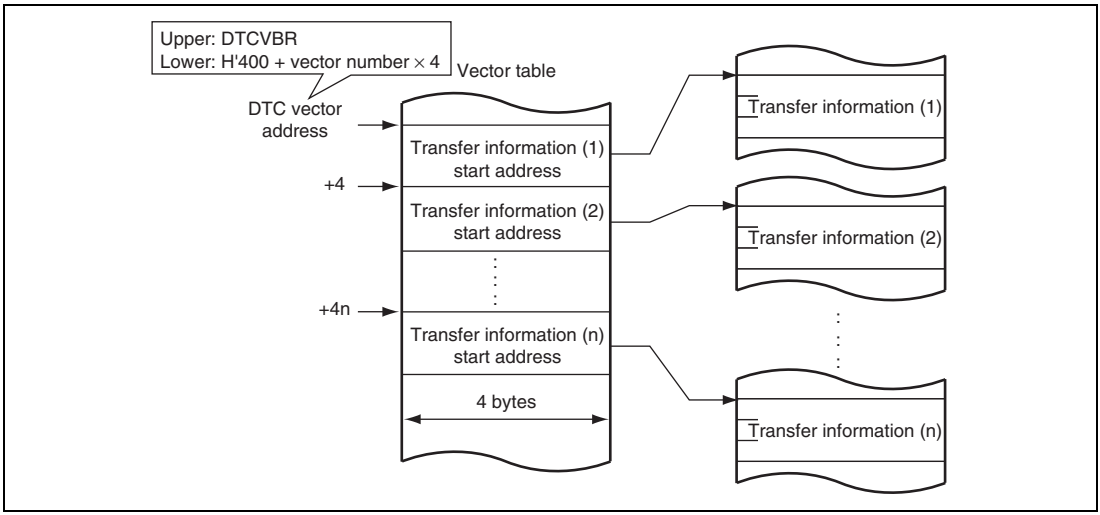


Figure 8.3 Correspondence between DTC Vector Address and Transfer Information

Table 8.1 shows correspondence between the DTC activation source and vector address.

Table 8.1 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs

Origin of Activation Source	Activation Source	Vector Number	DTC Vector Address Offset	DTCE*	Priority
External pin	IRQ0	64	H'500	DTCEA15	High
	IRQ1	65	H'504	DTCEA14	↑
	IRQ2	66	H'508	DTCEA13	
	IRQ3	67	H'50C	DTCEA12	
	IRQ4	68	H'510	DTCEA11	
	IRQ5	69	H'514	DTCEA10	
	IRQ6	70	H'518	DTCEA9	
	IRQ7	71	H'51C	DTCEA8	
	IRQ8	72	H'520	DTCEA7	
	IRQ9	73	H'524	DTCEA6	
	IRQ10	74	H'528	DTCEA5	
	IRQ11	75	H'52C	DTCEA4	
A/D	ADI	86	H'558	DTCEB15	
TPU_0	TGI0A	88	H'560	DTCEB13	
	TGI0B	89	H'564	DTCEB12	
	TGI0C	90	H'568	DTCEB11	
	TGI0D	91	H'56C	DTCEB10	
TPU_1	TGI1A	93	H'574	DTCEB9	
	TGI1B	94	H'578	DTCEB8	
TPU_2	TGI2A	97	H'584	DTCEB7	
	TGI2B	98	H'588	DTCEB6	
TPU_3	TGI3A	101	H'594	DTCEB5	
	TGI3B	102	H'598	DTCEB4	
	TGI3C	103	H'59C	DTCEB3	
	TGI3D	104	H'5A0	DTCEB2	
TPU_4	TGI4A	106	H'5A8	DTCEB1	
	TGI4B	107	H'5AC	DTCEB0	
TPU_5	TGI5A	110	H'5B8	DTCEC15	
	TGI5B	111	H'5BC	DTCEC14	Low

Origin of Activation Source	Activation Source	Vector Number	DTC Vector Address Offset	DTCE*	Priority
TMR_0	CMIA0	116	H'5D0	DTCEC13	High
	CMIB0	117	H'5D4	DTCEC12	
TMR_1	CMIA1	119	H'5DC	DTCEC11	
	CMIB1	120	H'5E0	DTCEC10	
TMR_2	CMIA2	122	H'5E8	DTCEC9	
	CMIB2	123	H'5EC	DTCEC8	
TMR_3	CMIA3	125	H'5F4	DTCEC7	
	CMIB3	126	H'5F8	DTCEC6	
DMAC	DMTEND0	128	H'600	DTCEC5	
	DMTEND1	129	H'604	DTCEC4	
	DMTEND2	130	H'608	DTCEC3	
	DMTEND3	131	H'60C	DTCEC2	
DMAC	DMEEND0	136	H'620	DTCED13	
	DMEEND1	137	H'624	DTCED12	
	DMEEND2	138	H'628	DTCED11	
	DMEEND3	139	H'62C	DTCED10	
SCI_0	RXI0	145	H'644	DTCED5	
	TXI0	146	H'648	DTCED4	
SCI_1	RXI1	149	H'654	DTCED3	
	TXI1	150	H'658	DTCED2	
SCI_2	RXI2	153	H'664	DTCED1	
	TXI2	154	H'668	DTCED0	
SCI_4	RXI4	161	H'684	DTCEE13	
	TXI4	162	H'688	DTCEE12	Low

Note: * The DTCE bits with no corresponding interrupt are reserved, and the write value should always be 0. To leave software standby mode or all-module-clock-stop mode with an interrupt, write 0 to the corresponding DTCE bit.

8.5 Operation

The DTC stores transfer information in the data area. When activated, the DTC reads transfer information that is stored in the data area and transfers data on the basis of that transfer information. After the data transfer, it writes updated transfer information back to the data area. Since transfer information is in the data area, it is possible to transfer data over any required number of channels. There are three transfer modes: normal, repeat, and block.

The DTC specifies the source address and destination address in SAR and DAR, respectively. After a transfer, SAR and DAR are incremented, decremented, or fixed independently.

Table 8.2 shows the DTC transfer modes.

Table 8.2 DTC Transfer Modes

Transfer Mode	Size of Data Transferred at One Transfer Request	Memory Address Increment or Decrement	Transfer Count
Normal	1 byte/word/longword	Incremented/decremented by 1, 2, or 4, or fixed	1 to 65536
Repeat*[1]	1 byte/word/longword	Incremented/decremented by 1, 2, or 4, or fixed	1 to 256*[3]
Block*[2]	Block size specified by CRAH (1 to 256 bytes/words/longwords)	Incremented/decremented by 1, 2, or 4, or fixed	1 to 65536

Notes: 1. Either source or destination is specified to repeat area.
2. Either source or destination is specified to block area.
3. After transfer of the specified transfer count, initial state is recovered to continue the operation.

Setting the CHNE bit in MRB to 1 makes it possible to perform a number of transfers with a single activation (chain transfer). Setting the CHNS bit in MRB to 1 can also be made to have chain transfer performed only when the transfer counter value is 0.

Figure 8.4 shows a flowchart of DTC operation, and table 8.3 summarizes the chain transfer conditions (combinations for performing the second and third transfers are omitted).

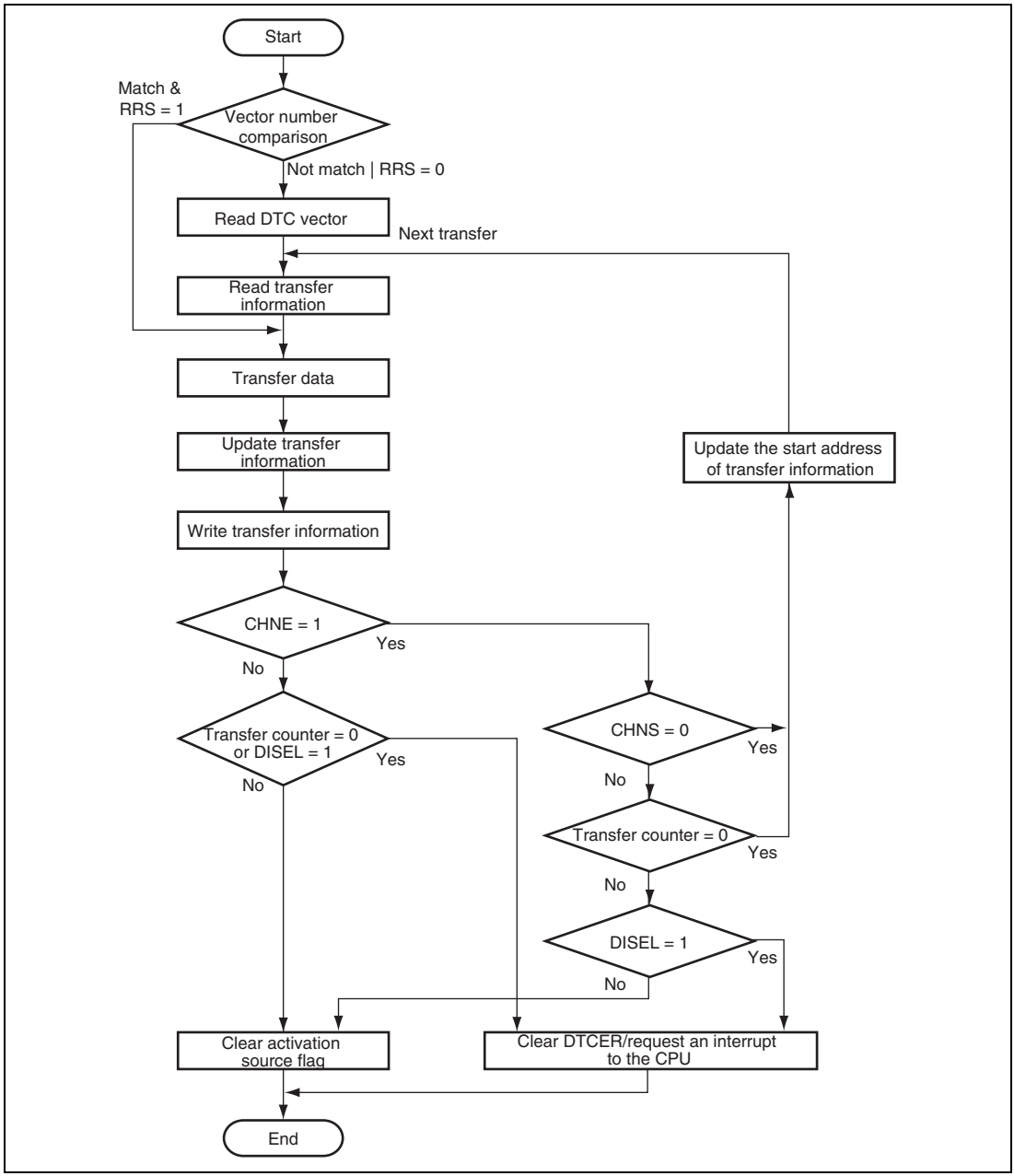


Figure 8.4 Flowchart of DTC Operation

Table 8.3 Chain Transfer Conditions

1st Transfer				2nd Transfer				
CHNE	CHNS	DISEL	Transfer Counter*[1]	CHNE	CHNS	DISEL	Transfer Counter*[1]	DTC Transfer
0	—	0	Not 0	—	—	—	—	Ends at 1st transfer
0	—	0	0*[2]	—	—	—	—	Ends at 1st transfer
0	—	1		—	—	—	—	Interrupt request to CPU
1	0	—	—	0	—	0	Not 0	Ends at 2nd transfer
				0	—	0	0*[2]	Ends at 2nd transfer
				0	—	1	—	Interrupt request to CPU
1	1	0	Not 0	—	—		—	Ends at 1st transfer
1	1	—	0*[2]	0	—	0	Not 0	Ends at 2nd transfer
				0	—	0	0*[2]	Ends at 2nd transfer
				0	—	1		Interrupt request to CPU
1	1	1	Not 0	—	—	—	—	Ends at 1st transfer
								Interrupt request to CPU

Notes: 1. CRA in normal mode transfer, CRAL in repeat transfer mode, or CRB in block transfer mode
2. When the contents of the CRAH is written to the CRAL in repeat transfer mode

8.5.1 Bus Cycle Division

When the transfer data size is word and the SAR and DAR values are not a multiple of 2, the bus cycle is divided and the transfer data is read from or written to in bytes. Similarly, when the transfer data size is longword and the SAR and DAR values are not a multiple of 4, the bus cycle is divided and the transfer data is read from or written to in words.

Table 8.4 shows the relationship among, SAR, DAR, transfer data size, bus cycle divisions, and access data size. Figure 8.5 shows the bus cycle division example.

Table 8.4 Number of Bus Cycle Divisions and Access Size

	Specified Data Size		
SAR and DAR Values	Byte (B)	Word (W)	Longword (LW)
Address 4n	1 (B)	1 (W)	1 (LW)
Address 2n + 1	1 (B)	2 (B-B)	3 (B-W-B)
Address 4n + 2	1 (B)	1 (W)	2 (W-W)

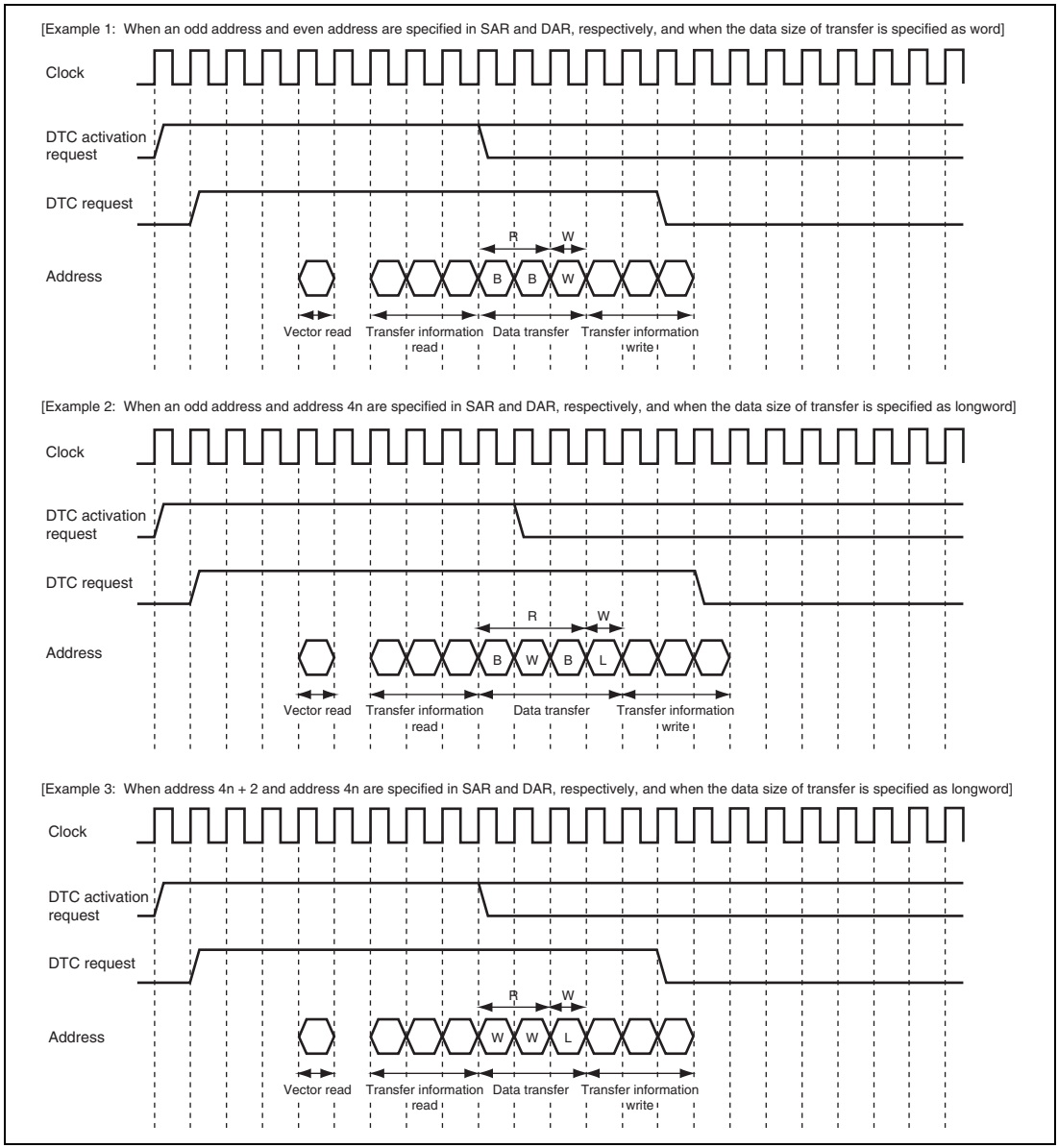


Figure 8.5 Bus Cycle Division Example

8.5.2 Transfer Information Read Skip Function

By setting the RRS bit of DTCCR, the vector address read and transfer information read can be skipped. The current DTC vector number is always compared with the vector number of previous activation. If the vector numbers match when RRS = 1, a DTC data transfer is performed without reading the vector address and transfer information. If the previous activation is a chain transfer, the vector address read and transfer information read are always performed. Figure 8.6 shows the transfer information read skip timing.

To modify the vector table and transfer information, temporarily clear the RRS bit to 0, modify the vector table and transfer information, and then set the RRS bit to 1 again. When the RRS bit is cleared to 0, the stored vector number is deleted, and the updated vector table and transfer information are read at the next activation.

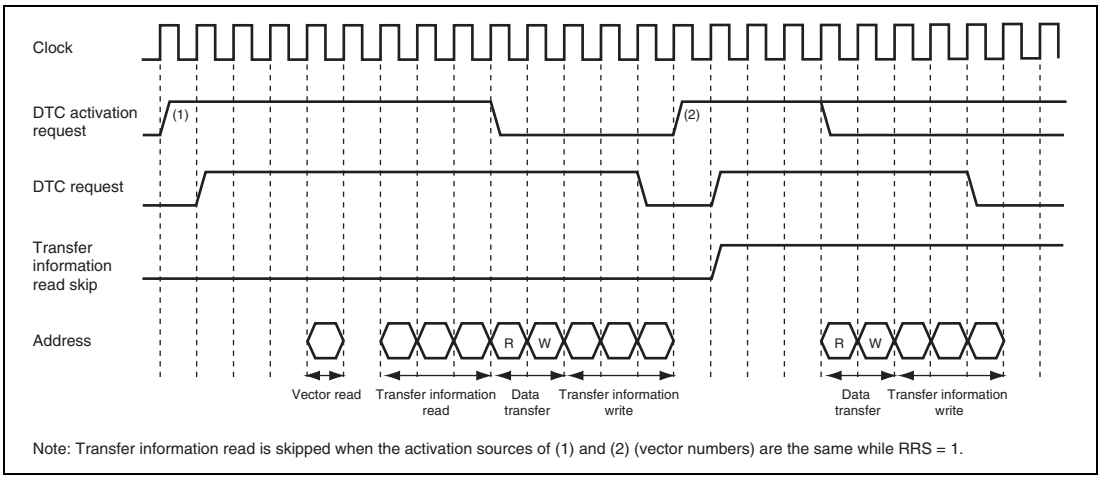


Figure 8.6 Transfer Information Read Skip Timing

8.5.3 Transfer Information Writeback Skip Function

By specifying bit SM1 in MRA and bit DM1 in MRB to the fixed address mode, a part of transfer information will not be written back. This function is performed regardless of short or full address mode. Table 8.5 shows the transfer information writeback skip condition and writeback skipped registers. Note that the CRA and CRB are always written back regardless of the short or full address mode. In addition in full address mode, the writeback of the MRA and MRB are always skipped.

Table 8.5 Transfer Information Writeback Skip Condition and Writeback Skipped Registers

SM1	DM1	SAR	DAR
0	0	Skipped	Skipped
0	1	Skipped	Written back
1	0	Written back	Skipped
1	1	Written back	Written back

8.5.4 Normal Transfer Mode

In normal transfer mode, one operation transfers one byte, one word, or one longword of data. From 1 to 65,536 transfers can be specified. The transfer source and destination addresses can be specified as incremented, decremented, or fixed. When the specified number of transfers ends, an interrupt can be requested to the CPU.

Table 8.6 lists the register function in normal transfer mode. Figure 8.7 shows the memory map in normal transfer mode.

Table 8.6 Register Function in Normal Transfer Mode

Register	Function	Written Back Value
SAR	Source address	Incremented/decremented/fixed*
DAR	Destination address	Incremented/decremented/fixed*
CRA	Transfer count A	CRA – 1
CRB	Transfer count B	Not updated

Note: * Transfer information writeback is skipped.

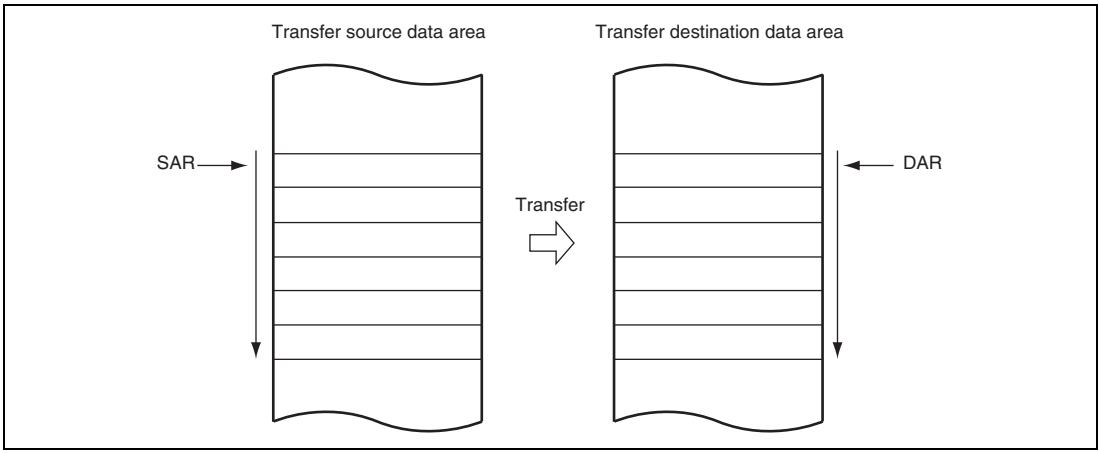


Figure 8.7 Memory Map in Normal Transfer Mode

8.5.5 Repeat Transfer Mode

In repeat transfer mode, one operation transfers one byte, one word, or one longword of data. By the DTS bit in MRB, either the source or destination can be specified as a repeat area. From 1 to 256 transfers can be specified. When the specified number of transfers ends, the transfer counter and address register specified as the repeat area is restored to the initial state, and transfer is repeated. The other address register is then incremented, decremented, or left fixed. In repeat transfer mode, the transfer counter (CRAL) is updated to the value specified in CRAH when CRAL becomes H'00. Thus the transfer counter value does not reach H'00, and therefore a CPU interrupt cannot be requested when DISEL = 0.

Table 8.7 lists the register function in repeat transfer mode. Figure 8.8 shows the memory map in repeat transfer mode.

Table 8.7 Register Function in Repeat Transfer Mode

		Written Back Value	
Register	**Function**	**CRAL is not 1**	**CRAL is 1**
SAR	Source address	Incremented/decremented/fixed*	DTS =0: Incremented/ decremented/fixed* DTS = 1: SAR initial value
DAR	Destination address	Incremented/decremented/fixed*	DTS = 0: DAR initial value DTS =1: Incremented/ decremented/fixed*
CRAH	Transfer count storage	CRAH	CRAH
CRAL	Transfer count A	CRAL – 1	CRAH
CRB	Transfer count B	Not updated	Not updated

Note: * Transfer information writeback is skipped.

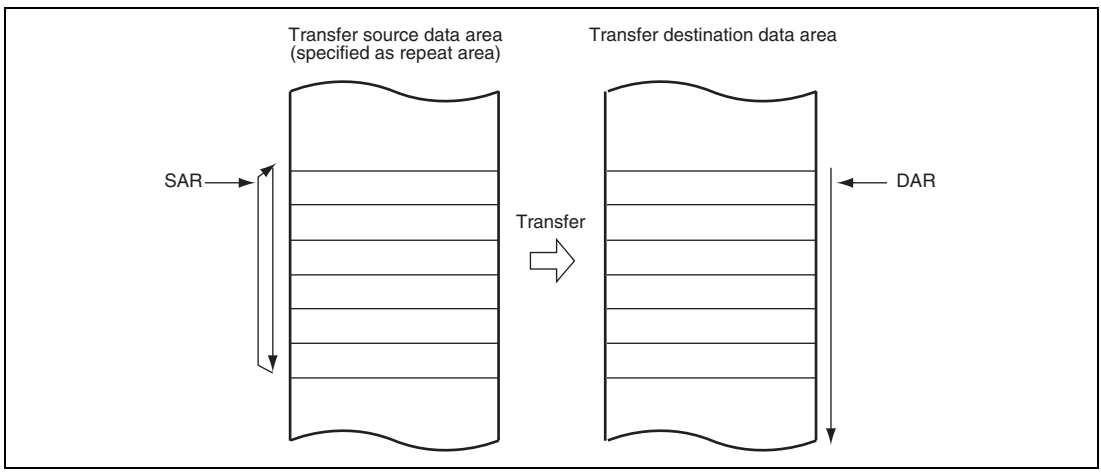


Figure 8.8 Memory Map in Repeat Transfer Mode (When Transfer Source is Specified as Repeat Area)

8.5.6 Block Transfer Mode

In block transfer mode, one operation transfers one block of data. Either the transfer source or the transfer destination is designated as a block area by the DTS bit in MRB.

The block size is 1 to 256 bytes (1 to 256 words, or 1 to 256 longwords). When the transfer of one block ends, the block size counter (CRAL) and address register (SAR when DTS = 1 or DAR when DTS = 0) specified as the block area is restored to the initial state. The other address register is then incremented, decremented, or left fixed. From 1 to 65,536 transfers can be specified. When the specified number of transfers ends, an interrupt is requested to the CPU.

Table 8.8 lists the register function in block transfer mode. Figure 8.9 shows the memory map in block transfer mode.

Table 8.8 Register Function in Block Transfer Mode

Register	Function	Written Back Value
SAR	Source address	DTS =0: Incremented/decremented/fixed*
		DTS = 1: SAR initial value
DAR	Destination address	DTS = 0: DAR initial value
		DTS =1: Incremented/decremented/fixed*
CRAH	Block size storage	CRAH
CRAL	Block size counter	CRAH
CRB	Block transfer counter	CRB – 1

Note: * Transfer information writeback is skipped.

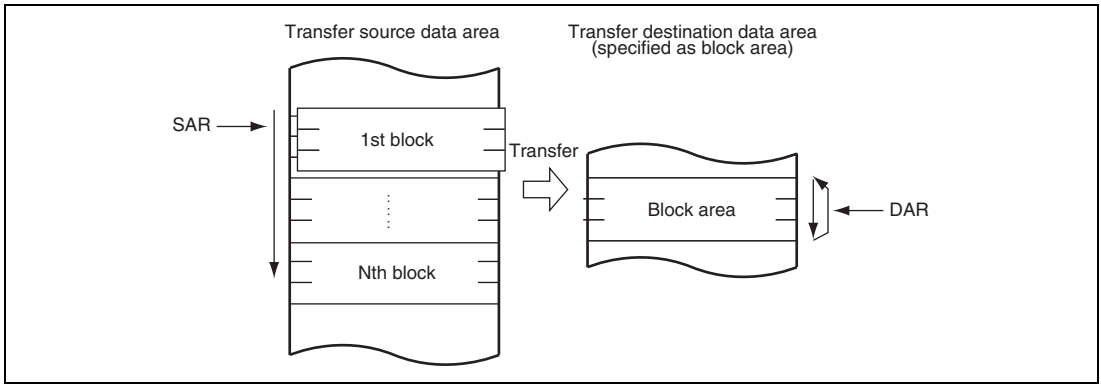


Figure 8.9 Memory Map in Block Transfer Mode (When Transfer Destination is Specified as Block Area)

8.5.7 Chain Transfer

Setting the CHNE bit in MRB to 1 enables a number of data transfers to be performed consecutively in response to a single transfer request. Setting the CHNE and CHNS bits in MRB set to 1 enables a chain transfer only when the transfer counter reaches 0. SAR, DAR, CRA, CRB, MRA, and MRB, which define data transfers, can be set independently. Figure 8.10 shows the chain transfer operation.

In the case of transfer with CHNE set to 1, an interrupt request to the CPU is not generated at the end of the specified number of transfers or by setting the DISEL bit to 1, and the interrupt source flag for the activation source and DTCER are not affected.

In repeat transfer mode, setting the RCHNE bit in DTCCR and the CHNE and CHNS bits in MRB to 1 enables a chain transfer after transfer with transfer counter = 1 has been completed.

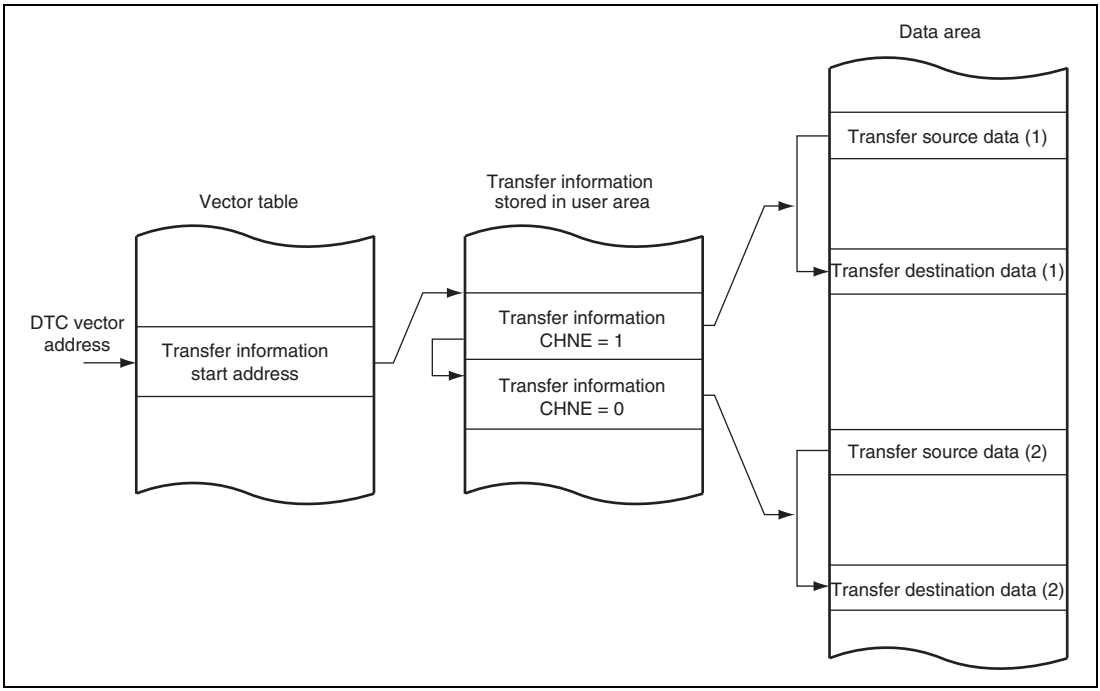


Figure 8.10 Operation of Chain Transfer

8.5.8 Operation Timing

Figures 8.11 to 8.14 show the DTC operation timings.

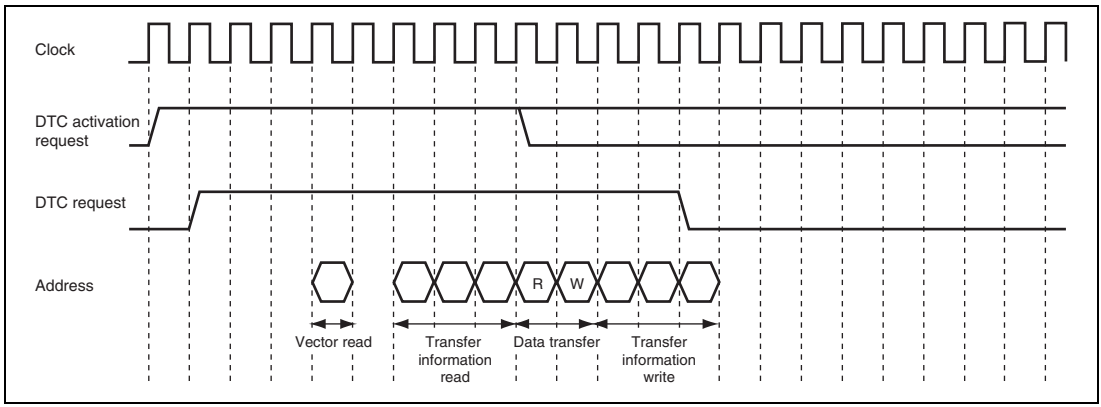


Figure 8.11 DTC Operation Timing
(Example of Short Address Mode in Normal Transfer Mode or Repeat Transfer Mode)

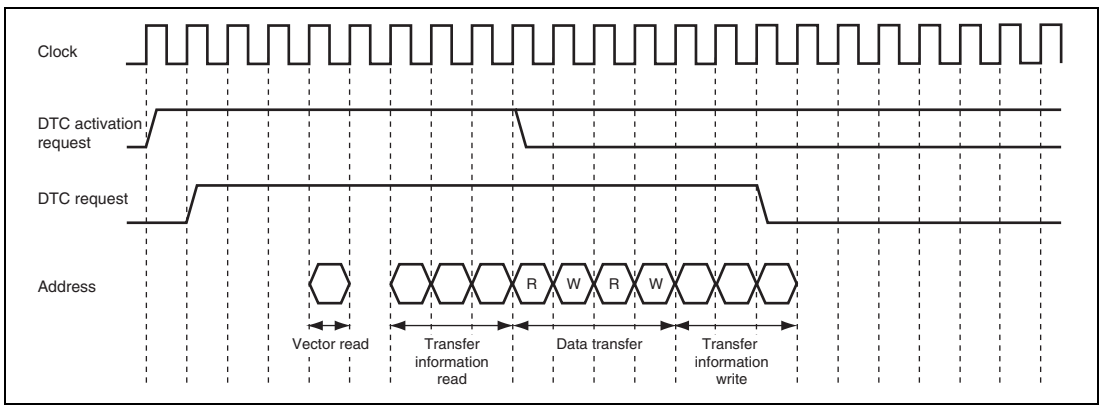


Figure 8.12 DTC Operation Timing
(Example of Short Address Mode in Block Transfer Mode with Block Size of 2)

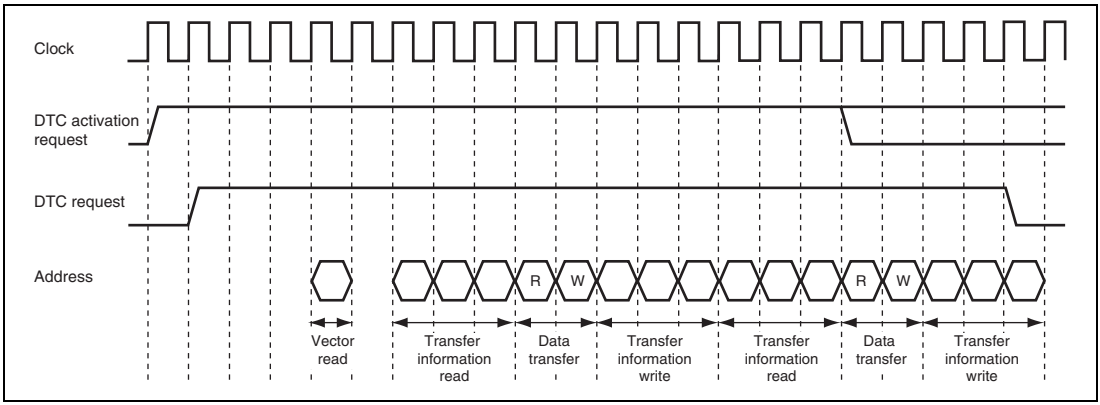


Figure 8.13 DTC Operation Timing (Example of Short Address Mode in Chain Transfer)

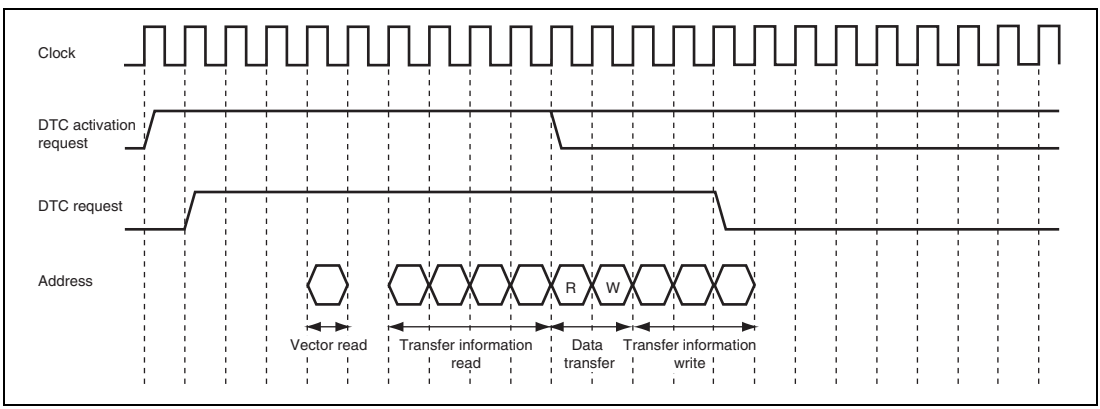


Figure 8.14 DTC Operation Timing
(Example of Full Address Mode in Normal Transfer Mode or Repeat Transfer Mode)

8.5.9 Number of DTC Execution Cycles

Table 8.9 shows the execution status for a single DTC data transfer, and table 8.10 shows the number of cycles required for each execution.

Table 8.9 DTC Execution Status

Mode	Vector Read I		Transfer Information Read J			Transfer Information Write L			Data Read L			Data Write M			Internal Operation N	
Normal	1	0*[1]	4*[2]	3*[3]	0*[1]	3*[2,3]	2*[4]	1*[5]	3*[6]	2*[7]	1	3*[6]	2*[7]	1	1	0*[1]
Repeat	1	0*[1]	4*[2]	3*[3]	0*[1]	3*[2,3]	2*[4]	1*[5]	3*[6]	2*[7]	1	3*[6]	2*[7]	1	1	0*[1]
Block transfer	1	0*[1]	4*[2]	3*[3]	0*[1]	3*[2,3]	2*[4]	1*[5]	3•P*[6]	2•P*[7]	1•P	3•P*[6]	2•P*[7]	1•P	1	0*[1]

[Legend]

P: Block size (CRAH and CRAL value)

Note: 1. When transfer information read is skipped
2. In full address mode operation
3. In short address mode operation
4. When the SAR or DAR is in fixed mode
5. When the SAR and DAR are in fixed mode
6. When a longword is transferred while an odd address is specified in the address register
7. When a word is transferred while an odd address is specified in the address register or when a longword is transferred while address 4n + 2 is specified

Table 8.10 Number of Cycles Required for Each Execution State

Object to be Accessed		On-Chip RAM	On-Chip ROM	On-Chip I/O Registers			External Devices			
Bus width		32	32	8	16	32		8		16
Access cycles		1	1	2	2	2	2	3	2	3
Execution status	Vector read S_I	1	1	—	—	—	8	12 + 4m	4	6 + 2m
	Transfer information read S_J	1	1	—	—	—	8	12 + 4m	4	6 + 2m
	Transfer information write S_k	1	1	—	—	—	8	12 + 4m	4	6 + 2m
	Byte data read S_L	1	1	2	2	2	2	3 + m	2	3 + m
	Word data read S_L	1	1	4	2	2	4	4 + 2m	2	3 + m
	Longword data read S_L	1	1	8	4	2	8	12 + 4m	4	6 + 2m
	Byte data write S_M	1	1	2	2	2	2	3 + m	2	3 + m
	Word data write S_M	1	1	4	2	2	4	4 + 2m	2	3 + m
	Longword data write S_M	1	1	8	4	2	8	12 + 4m	4	6 + 2m
	Internal operation S_N	1								

[Legend]

m: Number of wait cycles 0 to 7 (For details, see section 6, Bus Controller (BSC).)

The number of execution cycles is calculated from the formula below. Note that Σ means the sum of all transfers activated by one activation event (the number in which the CHNE bit is set to 1, plus 1).

$$\text{Number of execution cycles} = I \bullet S_I + \Sigma (J \bullet S_J + K \bullet S_K + L \bullet S_L + M \bullet S_M) + N \bullet S_N$$

8.5.10 DTC Bus Release Timing

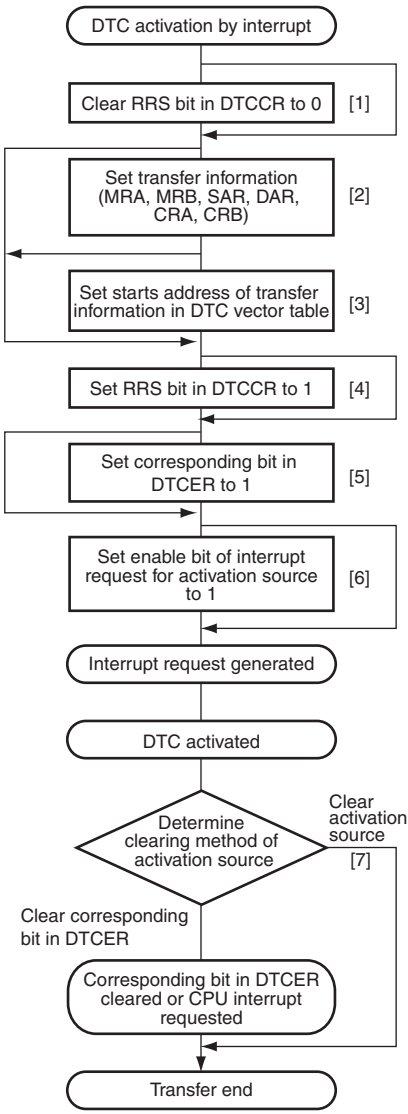
The DTC requests the bus mastership to the bus arbiter when an activation request occurs. The DTC releases the bus after a vector read, transfer information read, a single data transfer, or transfer information writeback. The DTC does not release the bus during transfer information read, single data transfer, or transfer information writeback.

8.5.11 DTC Priority Level Control to the CPU

The priority of the DTC activation sources over the CPU can be controlled by the CPU priority level specified by bits CPUP2 to CPUP0 in CPUPCR and the DTC priority level specified by bits DTCP2 to DTCP0. For details, see section 5, Interrupt Controller.

8.6 DTC Activation by Interrupt

The procedure for using the DTC with interrupt activation is shown in figure 8.15.



[1] Clearing the RRS bit in DTCCR to 0 clears the read skip flag of transfer information. Read skip is not performed when the DTC is activated after clearing the RRS bit. When updating transfer information, the RRS bit must be cleared.

[2] Set the MRA, MRB, SAR, DAR, CRA, and CRB transfer information in the data area. For details on setting transfer information, see section 8.2, Register Descriptions. For details on location of transfer information, see section 8.4, Location of Transfer Information and DTC Vector Table.

[3] Set the start address of the transfer information in the DTC vector table. For details on setting DTC vector table, see section 8.4, Location of Transfer Information and DTC Vector Table.

[4] Setting the RRS bit to 1 performs a read skip of second time or later transfer information when the DTC is activated consecutively by the same interrupt source. Setting the RRS bit to 1 is always allowed. However, the value set during transfer will be valid from the next transfer.

[5] Set the bit in DTCER corresponding to the DTC activation interrupt source to 1. For the correspondence of interrupts and DTCER, refer to table 8.1. The bit in DTCER may be set to 1 on the second or later transfer. In this case, setting the bit is not needed.

[6] Set the enable bits for the interrupt sources to be used as the activation sources to 1. The DTC is activated when an interrupt used as an activation source is generated. For details on the settings of the interrupt enable bits, see the corresponding descriptions of the corresponding module.

[7] After the end of one data transfer, the DTC clears the activation source flag or clears the corresponding bit in DTCER and requests an interrupt to the CPU. The operation after transfer depends on the transfer information. For details, see section 8.2, Register Descriptions and figure 8.4.

Figure 8.15 DTC with Interrupt Activation

8.7 Examples of Use of the DTC

8.7.1 Normal Transfer Mode

An example is shown in which the DTC is used to receive 128 bytes of data via the SCI.

1. Set MRA to fixed source address (SM1 = SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), normal transfer mode (MD1 = MD0 = 0), and byte size (Sz1 = Sz0 = 0). The DTS bit can have any value. Set MRB for one data transfer by one interrupt (CHNE = 0, DISEL = 0). Set the RDR address of the SCI in SAR, the start address of the RAM area where the data will be received in DAR, and 128 (H'0080) in CRA. CRB can be set to any value.
2. Set the start address of the transfer information for an RXI interrupt at the DTC vector address.
3. Set the corresponding bit in DTCER to 1.
4. Set the SCI to the appropriate receive mode. Set the RIE bit in SCR to 1 to enable the receive end (RXI) interrupt. Since the generation of a receive error during the SCI reception operation will disable subsequent reception, the CPU should be enabled to accept receive error interrupts.
5. Each time reception of one byte of data ends on the SCI, the RDRF flag in SSR is set to 1, an RXI interrupt is generated, and the DTC is activated. The receive data is transferred from RDR to RAM by the DTC. DAR is incremented and CRA is decremented. The RDRF flag is automatically cleared to 0.
6. When CRA becomes 0 after the 128 data transfers have ended, the RDRF flag is held at 1, the DTCE bit is cleared to 0, and an RXI interrupt request is sent to the CPU. Termination processing should be performed in the interrupt handling routine.

8.7.2 Chain Transfer

An example of DTC chain transfer is shown in which pulse output is performed using the PPG. Chain transfer can be used to perform pulse output data transfer and PPG output trigger cycle updating. Repeat mode transfer to the PPG's NDR is performed in the first half of the chain transfer, and normal mode transfer to the TPU's TGR in the second half. This is because clearing of the activation source and interrupt generation at the end of the specified number of transfers are restricted to the second half of the chain transfer (transfer when CHNE = 0).

1. Perform settings for transfer to the PPG's NDR. Set MRA to source address incrementing (SM1 = 1, SM0 = 0), fixed destination address (DM1 = DM0 = 0), repeat mode (MD1 = 0, MD0 = 1), and word size (Sz1 = 0, Sz0 = 1). Set the source side as a repeat area (DTS = 1). Set MRB to chain transfer mode (CHNE = 1, CHNS = 0, DISEL = 0). Set the data table start address in SAR, the NDRH address in DAR, and the data table size in CRAH and CRAL. CRB can be set to any value.
2. Perform settings for transfer to the TPU's TGR. Set MRA to source address incrementing (SM1 = 1, SM0 = 0), fixed destination address (DM1 = DM0 = 0), normal mode (MD1 = MD0 = 0), and word size (Sz1 = 0, Sz0 = 1). Set the data table start address in SAR, the TGRA address in DAR, and the data table size in CRA. CRB can be set to any value.
3. Locate the TPU transfer information consecutively after the NDR transfer information.
4. Set the start address of the NDR transfer information to the DTC vector address.
5. Set the bit corresponding to the TGIA interrupt in DTCER to 1.
6. Set TGRA as an output compare register (output disabled) with TIOR, and enable the TGIA interrupt with TIER.
7. Set the initial output value in PODR, and the next output value in NDR. Set bits in DDR and NDER for which output is to be performed to 1. Using PCR, select the TPU compare match to be used as the output trigger.
8. Set the CST bit in TSTR to 1, and start the TCNT count operation.
9. Each time a TGRA compare match occurs, the next output value is transferred to NDR and the set value of the next output trigger period is transferred to TGRA. The activation source TGFA flag is cleared.
10. When the specified number of transfers are completed (the TPU transfer CRA value is 0), the TGFA flag is held at 1, the DTCE bit is cleared to 0, and a TGIA interrupt request is sent to the CPU. Termination processing should be performed in the interrupt handling routine.

8.7.3 Chain Transfer when Counter = 0

By executing a second data transfer and performing re-setting of the first data transfer only when the counter value is 0, it is possible to perform 256 or more repeat transfers.

An example is shown in which a 128-kbyte input buffer is configured. The input buffer is assumed to have been set to start at lower address H'0000. Figure 8.16 shows the chain transfer when the counter value is 0.

1. For the first transfer, set the normal transfer mode for input data. Set the fixed transfer source address, CRA = H'0000 (65,536 times), CHNE = 1, CHNS = 1, and DISEL = 0.
2. Prepare the upper 8-bit addresses of the start addresses for 65,536-transfer units for the first data transfer in a separate area (in ROM, etc.). For example, if the input buffer is configured at addresses H'200000 to H'21FFFF, prepare H'21 and H'20.
3. For the second transfer, set repeat transfer mode (with the source side as the repeat area) for re-setting the transfer destination address for the first data transfer. Use the upper eight bits of DAR in the first transfer information area as the transfer destination. Set CHNE = DISEL = 0. If the above input buffer is specified as H'200000 to H'21FFFF, set the transfer counter to 2.
4. Execute the first data transfer 65536 times by means of interrupts. When the transfer counter for the first data transfer reaches 0, the second data transfer is started. Set the upper eight bits of the transfer source address for the first data transfer to H'21. The lower 16 bits of the transfer destination address of the first data transfer and the transfer counter are H'0000.
5. Next, execute the first data transfer the 65536 times specified for the first data transfer by means of interrupts. When the transfer counter for the first data transfer reaches 0, the second data transfer is started. Set the upper eight bits of the transfer source address for the first data transfer to H'20. The lower 16 bits of the transfer destination address of the first data transfer and the transfer counter are H'0000.
6. Steps 4 and 5 are repeated endlessly. As repeat mode is specified for the second data transfer, no interrupt request is sent to the CPU.

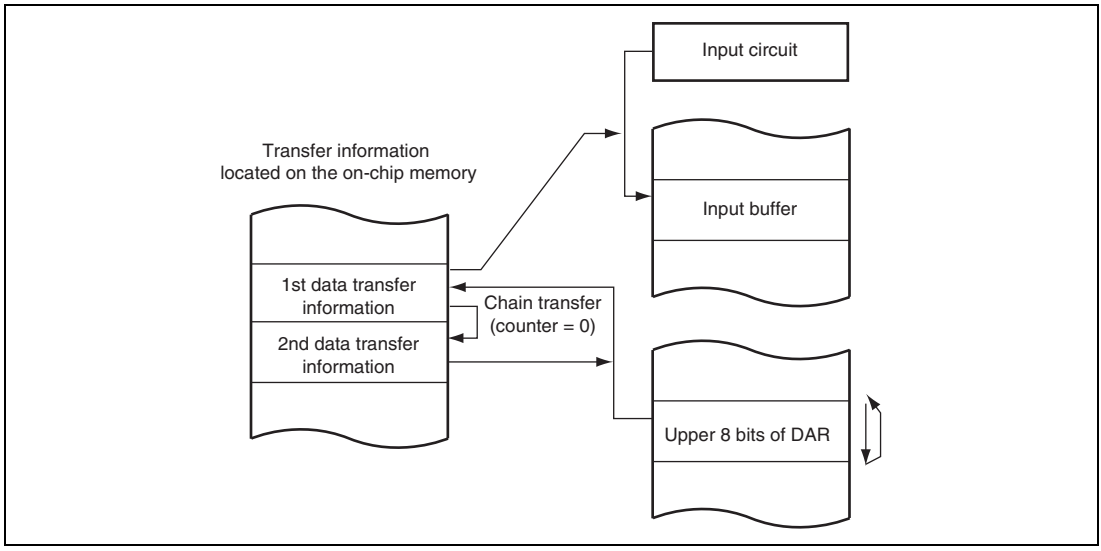


Figure 8.16 Chain Transfer when Counter = 0

8.8 Interrupt Sources

An interrupt request is issued to the CPU when the DTC finishes the specified number of data transfers or a data transfer for which the DISEL bit was set to 1. In the case of interrupt activation, the interrupt set as the activation source is generated. These interrupts to the CPU are subject to CPU mask level and priority level control in the interrupt controller.

8.9 Usage Notes

8.9.1 Module Stop Function Setting

Operation of the DTC can be disabled or enabled using the module stop control register. The initial setting is for operation of the DTC to be enabled. Register access is disabled by setting the module stop state. The module stop state cannot be set while the DTC is activated. For details, refer to section 20, Power-Down Modes.

8.9.2 On-Chip RAM

Transfer information can be located in on-chip RAM. In this case, the RAME bit in SYSCR must not be cleared to 0.

8.9.3 DMAC Transfer End Interrupt

When the DTC is activated by a DMAC transfer end interrupt, the DTE bit of DMDR is not controlled by the DTC but its value is modified with the write data regardless of the transfer counter value and DISEL bit setting. Accordingly, even if the DTC transfer counter value becomes 0, no interrupt request may be sent to the CPU in some cases.

8.9.4 DTCE Bit Setting

For DTCE bit setting, use bit manipulation instructions such as BSET and BCLR. If all interrupts are disabled, multiple activation sources can be set at one time (only at the initial setting) by writing data after executing a dummy read on the relevant register.

8.9.5 Chain Transfer

When chain transfer is used, clearing of the activation source or DTCER is performed when the last of the chain of data transfers is executed. SCI and A/D converter interrupt/activation sources are cleared when the DTC reads from or writes to the relevant register.

Therefore, when the DTC is activated by an interrupt or activation source, if a read/write of the relevant register is not included in the last chained data transfer, the interrupt or activation source will be retained.

8.9.6 Transfer Information Start Address, Source Address, and Destination Address

The transfer information start address to be specified in the vector table should be address 4n. If an address other than address 4n is specified, the lower 2 bits of the address are regarded as 0s.

The source and destination addresses specified in SAR and DAR, respectively, will be transferred in the divided bus cycles depending on the address and data size.

8.9.7 Transfer Information Modification

When IBCCS = 1 and the DMAC is used, clear the IBCCS bit to 0 and then set to 1 again before modifying the DTC transfer information in the CPU exception handling routine initiated by a DTC transfer end interrupt.

8.9.8 Endian Format

The DTC supports big and little endian formats. The endian formats used when transfer information is written to and when transfer information is read from by the DTC must be the same.

Section 9 I/O Ports

Table 9.1 summarizes the port functions. The pins of each port also have other functions such as input/output pins of on-chip peripheral modules or external interrupt input pins. Each I/O port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, a port register (PORT) used to read the pin states, and an input buffer control register (ICR) that controls input buffer on/off. Port 5 does not have a DR or a DDR register.

Ports D to F, H, and I have internal input pull-up MOSs and a pull-up MOS control register (PCR) that controls the on/off state of the input pull-up MOSs.

Ports 2 and F include an open-drain control register (ODR) that controls on/off of the output buffer PMOSs.

All of the I/O ports can drive a single TTL load and capacitive loads up to 30 pF.

All of the I/O ports can drive Darlington transistors when functioning as output ports.

Ports 2 and 3 are Schmitt-trigger inputs. Schmitt-trigger inputs for other ports are enabled when used as the $\overline{\text{IRQ}}$, TPU, or TMR inputs.

Table 9.1 Port Functions

Port	Description	Bit	Function			Schmitt-Trigger Input*[1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 1	General I/O port also functioning as interrupt inputs, SCI I/Os, DMAC I/Os, A/D converter inputs, and TPU inputs	7	P17	$\overline{\text{IRQ7}}$-A/ TCLKD-B	—	$\overline{\text{IRQ7}}$-A, TCLKD-B	—	—
		6	P16	$\overline{\text{IRQ6}}$-A/ TCLKC-B	$\overline{\text{DACK1}}$-A	$\overline{\text{IRQ6}}$-A, TCLKC-B		
		5	P15	$\overline{\text{IRQ5}}$-A/ TCLKB-B	$\overline{\text{TEND1}}$-A	$\overline{\text{IRQ5}}$-A, TCLKB-B		
		4	P14	$\overline{\text{DREQ1}}$-A/ $\overline{\text{IRQ4}}$-A/ TCLKA-B	—	$\overline{\text{IRQ4}}$-A, TCLKA-B		
		3	P13	$\overline{\text{ADTRG0}}$/ $\overline{\text{IRQ3}}$-A	—	$\overline{\text{IRQ3}}$-A		
		2	P12/SCK2	$\overline{\text{IRQ2}}$-A	$\overline{\text{DACK0}}$-A	$\overline{\text{IRQ2}}$-A		
		1	P11	RxD2/ $\overline{\text{IRQ1}}$-A	$\overline{\text{TEND0}}$-A	$\overline{\text{IRQ1}}$-A		
		0	P10	$\overline{\text{DREQ0}}$-A/ $\overline{\text{IRQ0}}$-A	TxD2	$\overline{\text{IRQ0}}$-A		

Port	Description	Bit	Function			Schmitt-Trigger Input *[1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 2	General I/O port also functioning as interrupt inputs, PPG outputs, TPU I/Os, TMR I/Os, and SCI I/Os	7	P27/ TIOCB5	TIOCA5	PO7	All input functions	—	O
		6	P26/ TIOCA5	—	PO6/TMO1/ TxD1	All input functions		
		5	P25/ TIOCA4	TMCI1/ RxD1	PO5	P25, TIOCA4, MIOCB4, TMCI1		
		4	P24/ TIOCB4/ SCK1	TIOCA4/ TMRI1	PO4	P24, TIOCB4, TIOCA4, TMRI1		
		3	P23/ TIOCD3	$\overline{\text{IRQ11}}$-A/ TIOCC3	PO3	All input functions		
		2	P22/ TIOCC3	$\overline{\text{IRQ10}}$-A	PO2/TMO0/ TxD0	All input functions		
		1	P21/ TIOCA3	TMCI0/ RxD0/ $\overline{\text{IRQ9}}$-A	PO1	P21, $\overline{\text{IRQ9}}$-A, TIOCA3, TMCI0		
		0	P20/ TIOCB3/ SCK0	TIOCA3/ TMRI0/ $\overline{\text{IRQ8}}$-A	PO0	P20, $\overline{\text{IRQ8}}$-A, TIOCB3, TIOCA3, TMRI0		

Port	Description	Bit	Function			Schmitt-Trigger Input*[1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 3	General I/O port also functioning as PPG outputs, DMAC I/Os, and TPU I/Os	7	P37/ TIOCB2	TIOCA2/ TCLKD-A	PO15	All input functions	—	—
		6	P36/ TIOCA2	—	PO14	All input functions		
		5	P35/ TIOCB1	TIOCA1/ TCLKC-A	PO13/ $\overline{\text{DACK1}}$-B	All input functions		
		4	P34/ TIOCA1	—	PO12/ $\overline{\text{TEND1}}$-B	All input functions		
		3	P33/ TIOCD0	TIOCC0/ TCLKB-A $\overline{\text{DREQ1}}$-B	PO11	All input functions		
		2	P32/ TIOCC0	TCLKA-A	PO10/ $\overline{\text{DACK0}}$-B	All input functions		
		1	P31/ TIOCB0	TIOCA0	PO9/ $\overline{\text{TEND0}}$-B	All input functions		
		0	P30/ TIOCA0	$\overline{\text{DREQ0}}$-B	PO8	All input functions		
Port 5	General input port also functioning as A/D converter inputs and D/A converter outputs	7	—	P57/AN7 $\overline{\text{IRQ7}}$-B	DA1	$\overline{\text{IRQ7}}$-B	—	—
		6	—	P56/AN6 $\overline{\text{IRQ6}}$-B	DA0	$\overline{\text{IRQ6}}$-B		
		5	—	P55/AN5 $\overline{\text{IRQ5}}$-B	—	$\overline{\text{IRQ5}}$-B		
		4	—	P54/AN4 $\overline{\text{IRQ4}}$-B	—	$\overline{\text{IRQ4}}$-B		
		3	—	P53/AN3 $\overline{\text{IRQ3}}$-B	—	$\overline{\text{IRQ3}}$-B		
		2	—	P52/AN2 $\overline{\text{IRQ2}}$-B	—	$\overline{\text{IRQ2}}$-B		
		1	—	P51/AN1 $\overline{\text{IRQ1}}$-B	—	$\overline{\text{IRQ1}}$-B		
		0	—	P50/AN0 $\overline{\text{IRQ0}}$-B	—	$\overline{\text{IRQ0}}$-B		

Port	Description	Bit	Function			Schmitt-Trigger Input*[1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 6	General I/O port also functioning as TMR I/Os, SCI I/Os, DMAC I/Os, and interrupt inputs	7	—	—	—	—	—	—
		6	—	—	—	—		
		5	P65	—	TMO3/ $\overline{\text{DACK3}}$	—		
		4	P64	TMCI3	$\overline{\text{TEND3}}$	TMCI3		
		3	P63	TMRI3/ $\overline{\text{DREQ3}}$/ $\overline{\text{IRQ11}}$-B	—	TMRI3, $\overline{\text{IRQ11}}$-B		
		2	P62/SCK4	$\overline{\text{IRQ10}}$-B	TMO2/ $\overline{\text{DACK2}}$	$\overline{\text{IRQ10}}$-B		
		1	P61	TMCI2/ RxD4/ $\overline{\text{IRQ9}}$-B	$\overline{\text{TEND2}}$	TMCI2, $\overline{\text{IRQ9}}$-B		
		0	P60	TMRI2/ $\overline{\text{DREQ2}}$/ $\overline{\text{IRQ8}}$-B	TxD4	TMRI2, $\overline{\text{IRQ8}}$-B		
Port A	General I/O port also functioning as system clock output and bus control I/Os	7	—	PA7	Bϕ	—	—	—
		6	PA6	—	$\overline{\text{AS}}$/$\overline{\text{AH}}$/ $\overline{\text{BS}}$-B			
		5	PA5	—	$\overline{\text{RD}}$			
		4	PA4	—	$\overline{\text{LHWR}}$/$\overline{\text{LUB}}$			
		3	PA3	—	$\overline{\text{LLWR}}$/$\overline{\text{LLB}}$			
		2	PA2	$\overline{\text{BREQ}}$/ $\overline{\text{WAIT}}$	—			
		1	PA1	—	$\overline{\text{BACK}}$/ (RD/$\overline{\text{WR}}$)			
		0	PA0	—	$\overline{\text{BREQO}}$/ $\overline{\text{BS}}$-A			

Port	Description	Bit	Function			Schmitt-Trigger Input*[1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port B	General I/O port also functioning as bus control outputs	7	—	—	—	—	—	—
		6	—	—	—			
		5	—	—	—			
		4	—	—	—			
		3	PB3	—	$\overline{CS3}$/ $\overline{CS7}$-A			
		2	PB2	—	$\overline{CS2}$-A/ $\overline{CS6}$-A			
		1	PB1	—	$\overline{CS1}$/ $\overline{CS2}$-B/ $\overline{CS5}$-A/ $\overline{CS6}$-B/ $\overline{CS7}$-B			
		0	PB0	—	$\overline{CS0}$/$\overline{CS4}$/ $\overline{CS5}$-B			
Port D	General I/O port also functioning as address outputs	7	PD7	—	A7	—	O	—
		6	PD6	—	A6			
		5	PD5	—	A5			
		4	PD4	—	A4			
		3	PD3	—	A3			
		2	PD2	—	A2			
		1	PD1	—	A1			
		0	PD0	—	A0			
Port E	General I/O port also functioning as address outputs	7	PE7	—	A15	—	O	—
		6	PE6	—	A14			
		5	PE5	—	A13			
		4	PE4	—	A12			
		3	PE3	—	A11			
		2	PE2	—	A10			
		1	PE1	—	A9			
		0	PE0	—	A8			

Port	Description	Bit	Function			Schmitt-Trigger Input*[1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port F	General I/O port also functioning as address outputs	7	PF7	—	A23	—	O	O
		6	PF6	—	A22			
		5	PF5	—	A21			
		4	PF4	—	A20			
		3	PF3	—	A19			
		2	PF2	—	A18			
		1	PF1	—	A17			
		0	PF0	—	A16			
Port H	General I/O port also functioning as bi-directional data bus	7	PH7/D7*[2]	—	—	—	O	—
		6	PH6/D6*[2]	—	—			
		5	PH5/D5*[2]	—	—			
		4	PH4/D4*[2]	—	—			
		3	PH3/D3*[2]	—	—			
		2	PH2/D2*[2]	—	—			
		1	PH1/D1*[2]	—	—			
		0	PH0/D0*[2]	—	—			
Port I	General I/O port also functioning as bi-directional data bus	7	PI7/D15*[2]	—	—	—	O	—
		6	PI6/D14*[2]	—	—			
		5	PI5/D13*[2]	—	—			
		4	PI4/D12*[2]	—	—			
		3	PI3/D11*[2]	—	—			
		2	PI2/D10*[2]	—	—			
		1	PI1/D9*[2]	—	—			
		0	PI0/D8*[2]	—	—			

Notes: 1. Pins without Schmitt-trigger input buffer have CMOS input buffer.
2. Addresses are also output when accessing to the address/data multiplexed I/O space.

9.1 Register Descriptions

Table 9.2 lists each port registers.

Table 9.2 Register Configuration in Each Port

Port	Number of Pins	Registers					
		DDR	DR	PORT	ICR	PCR	ODR
Port 1	8	O	O	O	O	—	—
Port 2	8	O	O	O	O	—	O
Port 3	8	O	O	O	O	—	—
Port 5	8	—	—	O	O	—	—
Port 6*[1]	6	O	O	O	O	—	—
Port A	8	O	O	O	O	—	—
Port B*[2]	4	O	O	O	O	—	—
Port D	8	O	O	O	O	O	—
Port E	8	O	O	O	O	O	—
Port F	8	O	O	O	O	O	O
Port H	8	O	O	O	O	O	—
Port I	8	O	O	O	O	O	—

[Legend]

O: Register exists

—: No register exists

Notes: 1. The lower six bits are valid and the upper two bits are reserved. The write value should always be the initial value.

2. The lower four bits are valid and the upper four bits are reserved. The write value should always be the initial value.

9.1.1 Data Direction Register (PnDDR) (n = 1 to 3, 6, A, B, D to F, H, and I)

DDR is an 8-bit write-only register that specifies the port input or output for each bit. A read from the DDR is invalid and DDR is always read as an undefined value.

When the general I/O port function is selected, the corresponding pin functions as an output port by setting the corresponding DDR bit to 1; the corresponding pin functions as an input port by clearing the corresponding DDR bit to 0.

The initial DDR values are shown in table 9.3.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DDR	Pn6DDR	Pn5DDR	Pn4DDR	Pn3DDR	Pn2DDR	Pn1DDR	Pn0DDR
Initial Value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Note: The lower six bits are valid and the upper two bits are reserved for port 6 registers.
The lower four bits are valid and the upper four bits are reserved for port B registers.

Table 9.3 Startup Mode and Initial Value

	Startup Mode	
Port	**External Extended Mode**	**Single-Chip Mode**
Port A	H'80	H'00
Other ports	H'00	

9.1.2 Data Register (PnDR) (n = 1 to 3, 6, A, B, D to F, H, and I)

DR is an 8-bit readable/writable register that stores the output data of the pins to be used as the general output port.

The initial value of DR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DR	Pn6DR	Pn5DR	Pn4DR	Pn3DR	Pn2DR	Pn1DR	Pn0DR
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 lower six bits are valid and the upper two bits are reserved for port 6 registers.
The lower four bits are valid and the upper four bits are reserved for port B registers.

9.1.3 Port Register (PORTn) (n = 1 to 3, 5, 6, A, B, D to F, H, and I)

PORT is an 8-bit read-only register that reflects the port pin state. A write to PORT is invalid. When PORT is read, the DR bits that correspond to the respective DDR bits set to 1 are read and the status of each pin whose corresponding DDR bit is cleared to 0 is also read regardless of the ICR value.

The initial value of PORT is undefined and is determined based on the port pin state.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	R	R	R	R	R	R	R	R

Note: The lower six bits are valid and the upper two bits are reserved for port 6 registers.
The lower four bits are valid and the upper four bits are reserved for port B registers.

9.1.4 Input Buffer Control Register (PnICR) (n = 1 to 3, 5, 6, A, B, D to F, H, and I)

ICR is an 8-bit readable/writable register that controls the port input buffers.

For bits in ICR set to 1, the input buffers of the corresponding pins are valid. For bits in ICR cleared to 0, the input buffers of the corresponding pins are invalid and the input signals are fixed high.

When the pin functions as an input for the peripheral modules, the corresponding bits should be set to 1. The initial value should be written to a bit whose corresponding pin is not used as an input or is used as an analog input/output pin.

When PORT is read, the pin state is always read regardless of the ICR value. When the ICR value is cleared to 0 at this time, the read pin state is not reflected in a corresponding on-chip peripheral module.

If ICR is modified, an internal edge may occur depending on the pin state. Accordingly, ICR should be modified when the corresponding input pins are not used. For example, an $\overline{\text{IRQ}}$ input, modify ICR while the corresponding interrupt is disabled, clear the IRQF flag in ISR of the interrupt controller to 0, and then enable the corresponding interrupt. If an edge occurs after the ICR setting, the edge should be cancelled.

The initial value of ICR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ICR	Pn6ICR	Pn5ICR	Pn4ICR	Pn3ICR	Pn2ICR	Pn1ICR	Pn0ICR
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 lower six bits are valid and the upper two bits are reserved for port 6 registers.
The lower four bits are valid and the upper four bits are reserved for port B registers.

9.1.5 Pull-Up MOS Control Register (PnPCR) (n = D to F, H, and I)

PCR is an 8-bit readable/writable register that controls on/off of the port input pull-up MOS.

If a bit in PCR is set to 1 while the pin is in input state, the input pull-up MOS corresponding to the bit in PCR is turned on. Table 9.4 shows the input pull-up MOS status.

The initial value of PCR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7PCR	Pn6PCR	Pn5PCR	Pn4PCR	Pn3PCR	Pn2PCR	Pn1PCR	Pn0PCR
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 9.4 Input Pull-Up MOS State

Port	Pin State	Reset	Hardware Standby Mode	Software Standby Mode	Other Operation
Port D	Address output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port E	Address output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port F	Address output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port H	Data input/output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port I	Data input/output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF

[Legend]

OFF: The input pull-up MOS is always off.

ON/OFF: If PCR is set to 1, the input pull-up MOS is on; if PCR is cleared to 0, the input pull-up MOS is off.

9.1.6 Open-Drain Control Register (PnODR) (n = 2 and F)

ODR is an 8-bit readable/writable register that selects the open-drain output function.

If a bit in ODR is set to 1, the pin corresponding to that bit in ODR functions as an NMOS open-drain output. If a bit in ODR is cleared to 0, the pin corresponding to that bit in ODR functions as a CMOS output.

The initial value of ODR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ODR	Pn6ODR	Pn5ODR	Pn4ODR	Pn3ODR	Pn2ODR	Pn1ODR	Pn0ODR
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 Output Buffer Control

This section describes the output priority of each pin.

The name of each peripheral module pin is followed by “_OE”. This (for example: TIOCA4_OE) indicates whether the output of the corresponding function is valid (1) or if another setting is specified (0). Table 9.5 lists each port output signal's valid setting. For details on the corresponding output signals, see the register description of each peripheral module. If the name of each peripheral module pin is followed by A or B, the pin function can be modified by the port function control register (PFCR). For details, see section 9.3, Port Function Controller.

For a pin whose initial value changes according to the activation mode, “Initial value E” indicates the initial value when the LSI is started up in external extended mode and “Initial value S” indicates the initial value when the LSI is started in single-chip mode.

9.2.1 Port 1

(1) P17/$\overline{\text{IRQ7}}$-A/TCLKD-B

The pin function is switched as shown below according to the P17DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P17DDR
I/O port	P17 output	1
	P17 input (initial setting)	0

(2) P16/$\overline{\text{DACK1}}$-A/$\overline{\text{IRQ6}}$-A/TCLKC-B

The pin function is switched as shown below according to the combination of the DMAC register setting and P16DDR bit setting.

		Setting	
		DMAC	I/O Port
Module Name	Pin Function	$\overline{\text{DACK1A}}$_OE	P16DDR
DMAC	$\overline{\text{DACK1}}$-A output	1	—
I/O port	P16 output	0	1
	P16 input (initial setting)	0	0

(3) P15/$\overline{\text{TEND1}}$-A/$\overline{\text{IRQ5}}$-A/TCLKB-B

The pin function is switched as shown below according to the combination of the DMAC register setting and P15DDR bit setting.

		Setting	
		DMAC	I/O Port
Module Name	Pin Function	$\overline{\text{TEND1A}}$_OE	P15DDR
DMAC	$\overline{\text{TEND1}}$-A output	1	—
I/O port	P15 output	0	1
	P15 input (initial setting)	0	0

(4) P14/$\overline{\text{DREQ1}}$-A/$\overline{\text{IRQ4}}$-A/TCLKA-B

The pin function is switched as shown below according to the P14DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P14DDR
I/O port	P14 output	1
	P14 input (initial setting)	0

(5) P13/ADTRG0/IRQ3-A

The pin function is switched as shown below according to the P13DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P13DDR
I/O port	P13 output	1
	P13 input (initial setting)	0

(6) P12/SCK2/$\overline{\text{DACK0}}$-A/$\overline{\text{IRQ2}}$-A

The pin function is switched as shown below according to the combination of the DMAC and SCI register settings and P12DDR bit setting.

		Setting		
		DMAC	SCI	I/O Port
Module Name	Pin Function	$\overline{\text{DACK0A}}$_OE	SCK2_OE	P12DDR
DMAC	$\overline{\text{DACK0}}$-A output	1	—	—
SCI	SCK2 output	0	1	—
I/O port	P12 output	0	0	1
	P12 input (initial setting)	0	0	0

(7) P11/RxD2/$\overline{\text{TEND0}}$-A/$\overline{\text{IRQ1}}$-A

The pin function is switched as shown below according to the combination of the DMAC register setting and P11DDR bit setting.

		Setting	
		DMAC	I/O Port
Module Name	Pin Function	$\overline{\text{TEND0A}}$_OE	P11DDR
DMAC	$\overline{\text{TEND0}}$-A output	1	—
I/O port	P11 output	0	1
	P11 input (initial setting)	0	0

(8) P10/TxD2/$\overline{\text{DREQ0}}$-A/$\overline{\text{IRQ0}}$-A

The pin function is switched as shown below according to the combination of the SCI register setting and P10DDR bit setting.

		Setting	
		SCI	I/O Port
Module Name	Pin Function	TxD2_OE	P10DDR
SCI	TxD2 output	1	—
I/O port	P10 output	0	1
	P10 input (initial setting)	0	0

9.2.2 Port 2

(1) P27/PO7/TIOCA5/TIOCB5

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P27DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCB5_OE	PO7_OE	P27DDR
TPU	TIOCB5 output	1	—	—
PPG	PO7 output	0	1	—
I/O port	P27 output	0	0	1
	P27 input (initial setting)	0	0	0

(2) P26/PO6/TIOCA5/TMO1/TxD1

The pin function is switched as shown below according to the combination of the TPU, TMR, SCI, and PPG register settings and P26DDR bit setting.

		Setting				
		TPU	TMR	SCI	PPG	I/O Port
Module Name	Pin Function	TIOCA5_OE	TMO1_OE	TxD1_OE	PO6_OE	P26DDR
TPU	TIOCA5 output	1	—	—	—	—
TMR	TMO1 output	0	1	—	—	—
SCI	TxD1 output	0	0	1	—	—
PPG	PO6 output	0	0	0	1	—
I/O port	P26 output	0	0	0	0	1
	P26 input (initial setting)	0	0	0	0	0

(3) P25/PO5/TIOCA4/TMCI1/RxD1

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P25DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCA4_OE	PO5_OE	P25DDR
TPU	TIOCA4 output	1	—	—
PPG	PO5 output	0	1	—
I/O port	P25 output	0	0	1
	P25 input (initial setting)	0	0	0

(4) P24/PO4/TIOCA4/TIOCB4/TMRI1/SCK1

The pin function is switched as shown below according to the combination of the TPU, SCI, and PPG register settings and P24DDR bit setting.

		Setting			
		TPU	SCI	PPG	I/O Port
Module Name	Pin Function	TIOCB4_OE	SCK1_OE	PO4_OE	P24DDR
TPU	TIOCB4 output	1	—	—	—
SCI	SCK1 output	0	1	—	—
PPG	PO4 output	0	0	1	—
I/O port	P24 output	0	0	0	1
	P24 input (initial setting)	0	0	0	0

(5) P23/PO3/TIOCC3/TIOCD3/$\overline{\text{IRQ11}}$-A

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P23DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCD3_OE	PO3_OE	P23DDR
TPU	TIOCD3 output	1	—	—
PPG	PO3 output	0	1	—
I/O port	P23 output	0	0	1
	P23 input (initial setting)	0	0	0

(6) P22 /PO2/TIOCC3/TMO0/TxD0/$\overline{\text{IRQ10}}$-A

The pin function is switched as shown below according to the combination of the TPU, TMR, SCI, and PPG register settings and P22DDR bit setting.

		Setting				
		TPU	TMR	SCI	PPG	I/O Port
Module Name	Pin Function	TIOCC3_OE	TMO0_OE	TxD0_OE	PO2_OE	P22DDR
TPU	TIOCC3 output	1	—	—	—	—
TMR	TMO0 output	0	1	—	—	—
SCI	TxD0 output	0	0	1	—	—
PPG	PO2 output	0	0	0	1	—
I/O port	P22 output	0	0	0	0	1
	P22 input (initial setting)	0	0	0	0	0

(7) P21/PO1/TIOCA3/TMCI0/RxD0/$\overline{\text{IRQ9}}$-A

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P21DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCA3_OE	PO1_OE	P21DDR
TPU	TIOCA3 output	1	—	—
PPG	PO1 output	0	1	—
I/O port	P21 output	0	0	1
	P21 input (initial setting)	0	0	0

(8) P20/PO0/TIOCA3/TIOCB3/TMRI0/SCK0/$\overline{\text{IRQ8}}$-A

The pin function is switched as shown below according to the combination of the TPU, SCI, and PPG register settings and P20DDR bit setting.

		Setting			
		TPU	SCI	PPG	I/O Port
Module Name	Pin Function	TIOCB3_OE	SCK0_OE	PO0_OE	P20DDR
TPU	TIOCB3 output	1	—	—	—
SCI	SCK0 output	0	1	—	—
PPG	PO0 output	0	0	1	—
I/O port	P20 output	0	0	0	1
	P20 input (initial setting)	0	0	0	0

9.2.3 Port 3

(1) P37/PO15/TIOCA2/TIOCB2/TCLKD-A

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P37DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCB2_OE	PO15_OE	P37DDR
TPU	TIOCB2 output	1	—	—
PPG	PO15 output	0	1	—
I/O port	P37 output	0	0	1
	P37 input (initial setting)	0	0	0

(2) P36/PO14/TIOCA2

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P36DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCA2_OE	PO14_OE	P36DDR
TPU	TIOCA2 output	1	—	—
PPG	PO14 output	0	1	—
I/O port	P36 output	0	0	1
	P36 input (initial setting)	0	0	0

(3) P35/PO13/TIOCA1/TIOCB1/TCLKC-A/$\overline{\text{DACK1}}$-B

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P35DDR bit setting.

		Setting			
		DMAC	TPU	PPG	I/O Port
Module Name	Pin Function	$\overline{\text{DACK1B}}$_OE	TIOCB1_OE	PO13_OE	P35DDR
DMAC	$\overline{\text{DACK1}}$-B output	1	—	—	—
TPU	TIOCB1 output	0	1	—	—
PPG	PO13 output	0	0	1	—
I/O port	P35 output	0	0	0	1
	P35 input (initial setting)	0	0	0	0

(4) P34/PO12/TIOCA1/$\overline{\text{TEND1}}$-B

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P34DDR bit setting.

		Setting			
		DMAC	TPU	PPG	I/O Port
Module Name	Pin Function	$\overline{\text{TEND1B}}$_OE	TIOCA1_OE	PO12_OE	P34DDR
DMAC	$\overline{\text{TEND1}}$-B output	1	—	—	—
TPU	TIOCA1 output	0	1	—	—
PPG	PO12 output	0	0	1	—
I/O port	P34 output	0	0	0	1
	P34 input (initial setting)	0	0	0	0

(5) P33/PO11/TIOCC0/TIOCD0/TCLKB-A/$\overline{\text{DREQ1}}$-B

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P33DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCD0_OE	PO11_OE	P33DDR
TPU	TIOCD0 output	1	—	—
PPG	PO11 output	0	1	—
I/O port	P33 output	0	0	1
	P33 input (initial setting)	0	0	0

(6) P32/PO10/TIOCC0/TCLKA-A/$\overline{\text{DACK0}}$-B

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P32DDR bit setting.

		Setting			
		DMAC	TPU	PPG	I/O Port
Module Name	Pin Function	$\overline{\text{DACK0B}}$_OE	TIOCC0_OE	PO10_OE	P32DDR
DMAC	$\overline{\text{DACK0}}$-B output	1	—	—	—
TPU	TIOCC0 output	0	1	—	—
PPG	PO10 output	0	0	1	—
I/O port	P32 output	0	0	0	1
	P32 input (initial setting)	0	0	0	0

(7) P31/PO9/TIOCA0/TIOCB0/$\overline{\text{TEND0}}$-B

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P31DDR bit setting.

		Setting			
		DMAC	TPU	PPG	I/O Port
Module Name	Pin Function	$\overline{\text{TEND0B}}$_OE	TIOCB0_OE	PO9_OE	P31DDR
DMAC	$\overline{\text{TEND0}}$-B output	1	—	—	—
TPU	TIOCB0 output	0	1	—	—
PPG	PO9 output	0	0	1	—
I/O port	P31 output	0	0	0	1
	P31 input (initial setting)	0	0	0	0

(8) P30/PO8/TIOCA0/$\overline{\text{DREQ0}}$-B

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P30DDR bit setting.

		Setting		
		TPU	PPG	I/O Port
Module Name	Pin Function	TIOCA0_OE	PO8_OE	P30DDR
TPU	TIOCA0 output	1	—	—
PPG	PO8 output	0	1	—
I/O port	P30 output	0	0	1
	P30 input (initial setting)	0	0	0

9.2.4 Port 5

(1) P57/AN7/DA1/$\overline{\text{IRQ7}}$-B

Module Name	Pin Function
D/A converter	DA1 output

(2) P56/AN6/DA0/$\overline{\text{IRQ6}}$-B

Module Name	Pin Function
D/A converter	DA0 output

9.2.5 Port 6

(1) P65/TMO3/$\overline{\text{DACK3}}$

The pin function is switched as shown below according to the combination of the DMAC and TMR register settings and P65DDR bit setting.

		Setting		
		DMAC	TMR	I/O Port
Module Name	Pin Function	$\overline{\text{DACK3}}$_OE	TMO3_OE	P65DDR
DMAC	$\overline{\text{DACK3}}$ output	1	—	—
TMR	TMO3 output	0	1	—
I/O port	P65 output	0	0	1
	P65 input (initial setting)	0	0	0

(2) P64/TMCI3/$\overline{\text{TEND3}}$

The pin function is switched as shown below according to the combination of the DMAC register setting and P64DDR bit setting.

		Setting	
		DMAC	I/O Port
Module Name	Pin Function	$\overline{\text{TEND3}}$_OE	P64DDR
DMAC	$\overline{\text{TEND3}}$ output	1	—
I/O port	P64 output	0	1
	P64 input (initial setting)	0	0

(3) P63/TMRI3/$\overline{\text{DREQ3}}$/$\overline{\text{IRQ11}}$-B

The pin function is switched as shown below according to the P63DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P63DDR
I/O port	P63 output	1
	P63 input	0

(4) P62/TMO2/SCK4/$\overline{\text{DACK2}}$/$\overline{\text{IRQ10}}$-B

The pin function is switched as shown below according to the combination of the DMAC, TMR, and SCI register settings and P62DDR bit setting.

		Setting			
		DMAC	TMR	SCI	I/O Port
Module Name	Pin Function	$\overline{\text{DACK2}}$_OE	TMO2_OE	SCK4_OE	P62DDR
DMAC	$\overline{\text{DACK2}}$ output	1	—	—	—
TMR	TMO2 output	0	1	—	—
SCI	SCK4 output	0	0	1	—
I/O port	P62 output	0	0	0	1
	P62 input (initial setting)	0	0	0	0

(5) P61/TMCI2/RxD4/$\overline{\text{TEND2}}$/$\overline{\text{IRQ9}}$-B

The pin function is switched as shown below according to the combination of the DMAC register setting and P61DDR bit setting.

		Setting	
		DMAC	I/O Port
Module Name	Pin Function	$\overline{\text{TEND2}}$_OE	P61DDR
DMAC	$\overline{\text{TEND2}}$ output	1	—
I/O port	P61 output	0	1
	P61 input (initial setting)	0	0

(6) P60/TMRI2/TxD4/$\overline{\text{DREQ2}}$/$\overline{\text{IRQ8}}$-B

The pin function is switched as shown below according to the combination of the SCI register setting and P60DDR bit setting.

		Setting	
		SCI	I/O Port
Module Name	Pin Function	TxD4_OE	P60DDR
SCI	TxD4 output	1	—
I/O port	P60 output	0	1
	P60 input (initial setting)	0	0

9.2.6 Port A

(1) PA7/Bϕ

The pin function is switched as shown below according to the PA7DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	PA7DDR
I/O port	Bϕ output* (initial setting E)	1
	PA7 input (initial setting S)	0

[Legend]

Initial setting E: Initial setting in external extended mode

Initial setting S: Initial setting in single-chip mode

Note: * The type of ϕ to be output switches according to the POSEL1 bit in SCKCR. For details, see section 19.1.1, System Clock Control Register (SCKCR).

(2) PA6/$\overline{AS}$/$\overline{AH}$/$\overline{BS}$-B

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, port function control register (PFCR), and the PA6DDR bit settings.

		Setting			
		Bus Controller		I/O Port	
Module Name	Pin Function	$\overline{AH}$_OE	$\overline{BSB}$_OE	$\overline{AS}$_OE	PA6DDR
Bus controller	$\overline{AH}$ output*	1	—	—	—
	$\overline{BS}$-B output*	0	1	—	—
	$\overline{AS}$ output* (initial setting E)	0	0	1	—
I/O port	PA6 output	0	0	0	1
	PA6 input (initial setting S)	0	0	0	0

[Legend]
Initial setting E: Initial setting in external extended mode
Initial setting S: Initial setting in single-chip mode
Note: * Valid in external extended mode (EXPE = 1)

(3) PA5/$\overline{RD}$

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, and the PA5DDR bit settings.

		Setting	
		MCU Operating Mode	I/O Port
Module Name	Pin Function	EXPE	PA5DDR
Bus controller	$\overline{RD}$ output* (Initial setting E)	1	—
I/O port	PA5 output	0	1
	PA5 input (initial setting S)	0	0

[Legend]
Initial setting E: Initial setting in external extended mode
Initial setting S: Initial setting in single-chip mode
Note: * Valid in external extended mode (EXPE = 1)

(4) PA4/$\overline{\text{LHWR}}$/$\overline{\text{LUB}}$

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, port function control register (PFCR), and the PA4DDR bit settings.

		Setting		
		Bus Controller	I/O Port	
Module Name	Pin Function	$\overline{\text{LUB}}$_OE*[2]	$\overline{\text{LHWR}}$_OE*[2]	PA4DDR
Bus controller	$\overline{\text{LUB}}$ output*[1]	1	—	—
	$\overline{\text{LHWR}}$ output*[1] (initial setting E)	—	1	—
I/O port	PA4 output	0	0	1
	PA4 input (initial setting S)	0	0	0

[Legend]

Initial setting E: Initial setting in external extended mode

Initial setting S: Initial setting in single-chip mode

Notes: 1. Valid in external extended mode (EXPE = 1)

2. When the byte control SRAM space is accessed while the byte control SRAM space is specified or while LHWROE =1, this pin functions as the $\overline{\text{LUB}}$ output; otherwise, the $\overline{\text{LHWR}}$ output.

(5) PA3/LLWR/LLB

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, and the PA3DDR bit settings.

		Setting		
		Bus Controller	I/O Port	
Module Name	Pin Function	$\overline{\text{LLB}}$_OE*[2]	$\overline{\text{LLWR}}$_OE*[2]	PA3DDR
Bus controller	$\overline{\text{LLB}}$ output*[1]	1	—	—
	$\overline{\text{LLWR}}$ output*[1] (initial setting E)	—	1	—
I/O port	PA3 output	0	0	1
	PA3 input (initial setting S)	0	0	0

[Legend]

Initial setting E: Initial setting in external extended mode

Initial setting S: Initial setting in single-chip mode

Notes: 1. Valid in external extended mode (EXPE = 1)

2. If the byte control SRAM space is accessed, this pin functions as the $\overline{\text{LLB}}$ output; otherwise, the $\overline{\text{LLWR}}$.

(6) PA2/BREQ/WAIT

The pin function is switched as shown below according to the combination of the bus controller register setting and the PA2DDR bit setting.

		Setting		
		Bus Controller		I/O Port
Module Name	Pin Function	BCR_BRLE	BCR_WAITE	PA2DDR
Bus controller	$\overline{\text{BREQ}}$ input	1	—	—
	$\overline{\text{WAIT}}$ input	0	1	—
I/O port	PA2 output	0	0	1
	PA2 input (initial setting)	0	0	0

(7) PA1/$\overline{\text{BACK}}$/(RD/$\overline{\text{WR}}$)

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, port function control register (PFCR), and the PA1DDR bit settings.

		Setting			
		Bus Controller		I/O Port	
Module Name	Pin Function	$\overline{\text{BACK}}$_OE	Byte control SRAM Selection	(RD/$\overline{\text{WR}}$)_OE	PA1DDR
Bus controller	$\overline{\text{BACK}}$ output *	1	—	—	—
	RD/$\overline{\text{WR}}$ output *	0	1	—	—
		0	0	1	—
I/O port	PA1 output	0	0	0	1
	PA1 input (initial setting)	0	0	0	0

Note: * Valid in external extended mode (EXPE = 1)

(8) PA0/$\overline{\text{BREQO}}$/$\overline{\text{BS}}$-A

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, port function control register (PFCR), and the PA0DDR bit settings.

		Setting		
		I/O Port	Bus Controller	I/O Port
Module Name	Pin Function	$\overline{\text{BSA}}$_OE	$\overline{\text{BREQ}}$_OE	PA0DDR
Bus controller	$\overline{\text{BS}}$-A output*	1	—	—
	$\overline{\text{BREQO}}$ output*	0	1	—
I/O port	PA0 output	0	0	1
	PA0 input (initial setting)	0	0	0

Note: * Valid in external extended mode (EXPE = 1)

9.2.7 Port B

PB3/$\overline{\text{CS3}}$/$\overline{\text{CS7}}$-A: The pin function is switched as shown below according to the combination of operating mode, EXPE bit, port function control register (PFCR), and the PB3DDR bit settings.

		Setting		
		I/O Port		
Module Name	**Pin Function**	**$\overline{\text{CS3}}$_OE**	**$\overline{\text{CS7A}}$_OE**	**PB3DDR**
Bus controller	$\overline{\text{CS3}}$ output*	1	—	—
	$\overline{\text{CS7}}$-A output*	—	1	—
I/O port	PB3 output	0	0	1
	PB3 input (initial setting)	0	0	0

Note: * Valid in external extended mode (EXPE = 1)

(1) PB2/$\overline{\text{CS2}}$-A/$\overline{\text{CS6}}$-A

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, port function control register (PFCR), and the PB2DDR bit settings.

		Setting		
		I/O Port		
Module Name	**Pin Function**	**$\overline{\text{CS2A}}$_OE**	**$\overline{\text{CS6A}}$_OE**	**PB2DDR**
Bus controller	$\overline{\text{CS2}}$-A output*	1	—	—
	$\overline{\text{CS6}}$-A output*	—	1	—
I/O port	PB2 output	0	0	1
	PB2 input (initial setting)	0	0	0

Note: * Valid in external extended mode (EXPE = 1)

(2) PB1/$\overline{CS1}$/$\overline{CS2}$-B/$\overline{CS5}$-A/$\overline{CS6}$-B/$\overline{CS7}$-B

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, port function control register (PFCR), and the PB1DDR bit settings.

		Setting					
		I/O Port					
Module Name	**Pin Function**	**$\overline{CS1}$_OE**	**$\overline{CS2B}$_OE**	**$\overline{CS5A}$_OE**	**$\overline{CS6B}$_OE**	**$\overline{CS7B}$_OE**	**PB1DDR**
Bus controller	$\overline{CS1}$ output*	1	—	—	—	—	—
	$\overline{CS2}$-B output*	—	1	—	—	—	—
	$\overline{CS5}$-A output*	—	—	1	—	—	—
	$\overline{CS6}$-B output*	—	—	—	1	—	—
	$\overline{CS7}$-B output*	—	—	—	—	1	—
I/O port	PB1 output	0	0	0	0	0	1
	PB1 input (initial setting)	0	0	0	0	0	0

Note: * Valid in external extended mode (EXPE = 1)

(3) PB0/$\overline{CS0}$/$\overline{CS4}$/$\overline{CS5}$-B

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, port function control register (PFCR), and the PB0DDR bit settings.

		Setting			
		I/O Port			
Module Name	**Pin Function**	**$\overline{CS0}$_OE**	**$\overline{CS4}$_OE**	**$\overline{CS5B}$_OE**	**PB0DDR**
Bus controller	$\overline{CS0}$ output* (initial setting E)	1	—	—	—
	$\overline{CS4}$ output*	—	1	—	—
	$\overline{CS5}$-B output*	—	—	1	—
I/O port	PB0 output	0	0	0	1
	PB0 input (initial setting S)	0	0	0	0

[Legend]

Initial setting E: Initial setting in on-chip ROM disabled external extended mode

Initial setting S: Initial setting in other modes

Note: * Valid in external extended mode (EXPE = 1)

9.2.8 Port D

(1) PD7/A7, PD6/A6, PD5/A5, PD4/A4, PD3/A3, PD2/A2, PD1/A1, PD0/A0

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, and the PDnDDR bit settings.

		Setting	
			I/O Port
Module Name	Pin Function	MCU Operating Mode	PDnDDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PDn output	Single-chip mode*	1
	PDn input (initial setting)	Modes other than on-chip ROM disabled extended mode	0

[Legend]

n = 0 to 7

Note: * Address output is enabled by setting PDnDDR = 1 in external extended mode (EXPE = 1)

9.2.9 Port E

(1) PE7/A15, PE6/A14, PE5/A13, PE4/A12, PE3/A11, PE2/A10, PE1/A9, PE0/A8

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, and the PEnDDR bit settings.

		Setting	
			I/O Port
Module Name	Pin Function	MCU Operating Mode	PEnDDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PEn output	Single-chip mode*	1
	PEn input (initial setting)	Modes other than on-chip ROM disabled extended mode	0

[Legend]

n = 0 to 7

Note: * Address output is enabled by setting PDnDDR = 1 in external extended mode (EXPE = 1)

9.2.10 Port F

(1) PF7/A23

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF7DDR bit settings.

			Setting	
			I/O Port	
MCU Operating Mode	Module Name	Pin Function	A23_OE	PF7DDR
Modes other than on-chip ROM disabled extended mode	Bus controller	A23 output*	1	—
	I/O port	PF7 output	0	1
		PF7 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(2) PF6/A22

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF6DDR bit settings.

			Setting	
			I/O Port	
MCU Operating Mode	Module Name	Pin Function	A22_OE	PF6DDR
Modes other than on-chip ROM disabled extended mode	Bus controller	A22 output*	1	—
	I/O port	PF6 output	0	1
		PF6 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(3) PF5/A21

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF5DDR bit settings.

			Setting	
			I/O Port	
MCU Operating Mode	Module Name	Pin Function	A21_OE	PF5DDR
Modes other than on-chip ROM disabled extended mode	Bus controller	A21 output*	1	—
	I/O port	PF5 output	0	1
		PF5 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(4) PF4/A20

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF4DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting: I/O Port A20_OE	Setting: I/O Port PF4DDR
On-chip ROM disabled extended mode	Bus controller	A20 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A20 output*	1	—
	I/O port	PF4 output	0	1
		PF4 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(5) PF3/A19

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF3DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting: I/O Port A19_OE	Setting: I/O Port PF3DDR
On-chip ROM disabled extended mode	Bus controller	A19 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A19 output*	1	—
	I/O port	PF3 output	0	1
		PF3 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(6) PF2/A18

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF2DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting: I/O Port A18_OE	Setting: I/O Port PF2DDR
On-chip ROM disabled extended mode	Bus controller	A18 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A18 output*	1	—
	I/O port	PF2 output	0	1
		PF2 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(7) PF1/A17

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF1DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting: I/O Port A17_OE	Setting: I/O Port PF1DDR
On-chip ROM disabled extended mode	Bus controller	A17 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A17 output*	1	—
	I/O port	PF1 output	0	1
		PF1 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

(8) PF0/A16

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, port function control register (PFCR), and the PF0DDR bit settings.

			Setting	
			I/O Port	
MCU Operating Mode	Module Name	Pin Function	A16_OE	PF0DDR
On-chip ROM disabled extended mode	Bus controller	A16 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A16 output*	1	—
	I/O port	PF0 output	0	1
		PF0 input (initial setting)	0	0

Note: * Valid in external extended mode (EXPE = 1)

9.2.11 Port H

(1) PH7/D7, PH6/D6, PH5/D5, PH4/D4, PH3/D3, PH2/D2, PH1/D1, PH0/D0

The pin function is switched as shown below according to the combination of operating mode, the EXPE bit, and the PHnDDR bit settings.

		Setting	
		MCU Operating Mode	I/O Port
Module Name	Pin Function	EXPE	PHnDDR
Bus controller	Data I/O* (initial setting E)	1	—
I/O port	PHn output	0	1
	PHn input (initial setting S)	0	0

[Legend]

Initial setting E: Initial setting in external extended mode

Initial setting S: Initial setting in single-chip mode

n = 0 to 7

Note: * Valid in external extended mode (EXPE = 1)

9.2.12 Port I

(1) PI7/D15, PI6/D14, PI5/D13, PI4/D12, PI3/D11, PI2/D10, PI1/D9, PI0/D8

The pin function is switched as shown below according to the combination of operating mode, bus mode, the EXPE bit, and the PInDDR bit settings.

		Setting	
		Bus Controller	I/O Port
Module Name	Pin Function	16-Bit Bus Mode	PInDDR
Bus controller	Data I/O* (initial setting E)	1	—
I/O port	PIn output	0	1
	PIn input (initial setting S)	0	0

[Legend]

Initial setting E: Initial setting in external extended mode

Initial setting S: Initial setting in single-chip mode

n = 0 to 7

Note: * Valid in external extended mode (EXPE = 1)

Table 9.5 Available Output Signals and Settings in Each Port

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P1	6	$\overline{\text{DACK1A}}$_OE	$\overline{\text{DACK1}}$	PFCR7.DMAS1[A,B] = 00	DACR.AMS = 1, DMDR.DACKE = 1
	5	$\overline{\text{TEND1A}}$_OE	$\overline{\text{TEND1}}$	PFCR7.DMAS1[A,B] = 00	DMDR.TENDE = 1
	2	$\overline{\text{DACK0A}}$_OE	$\overline{\text{DACK0}}$	PFCR7.DMAS0[A,B] = 00	DACR.AMS = 1, DMDR.DACKE = 1
		SCK2_OE	SCK2		When SCMR_2.SMIF = 1: SCR_2.TE = 1 or SCR_2.RE = 1 while SMR_2.GM = 0, SCR_2.CKE [1, 0] = 01 or while SMR_2.GM = 1 When SCMR_2.SMIF = 0: SCR_2.TE = 1 or SCR_2.RE = 1 while SMR_2.C/A = 0, SCR_2.CKE [1, 0] = 01 or while SMR_2.C/A = 1, SCR_2.CKE 1 = 0
	1	$\overline{\text{TEND0A}}$_OE	$\overline{\text{TEND0}}$	PFCR7.DMAS0[A,B] = 00	DMDR.TENDE = 1
	0	TxD2_OE	TxD2		SCR.TE = 1
P2	7	TIOCB5_OE	TIOCB5		TPU.TIOR5.IOB3 = 0, TPU.TIOR5.IOB[1,0] = 01/10/11
		PO7_OE	PO7		NDERL.NDER7 = 1
	6	TIOCA5_OE	TIOCA5		TPU.TIOR5.IOA3 = 0, TPU.TIOR5.IOA[1,0] = 01/10/11
		TMO1_OE	TMO1		TCSR.OS3,2 = 01/10/11 or TCSR.OS[1,0] = 01/10/11
		TxD1_OE	TxD1		SCR.TE = 1
		PO6_OE	PO6		NDERL.NDER6 = 1
	5	TIOCA4_OE	TIOCA4		TPU.TIOR4.IOA3 = 0, TPU.TIOR4.IOA[1,0] = 01/10/11
		PO5_OE	PO5		NDERL.NDER5 = 1
	4	TIOCB4_OE	TIOCB4		TPU.TIOR4.IOB3 = 0, TPU.TIOR4.IOB[1,0] = 01/10/11
		SCK1_OE	SCK1		When SCMR_1.SMIF = 1: SCR_1.TE = 1 or SCR_1.RE = 1 while SMR_1.GM = 0, SCR_1.CKE [1, 0] = 01 or while SMR_1.GM = 1 When SCMR_1.SMIF = 0: SCR_1.TE = 1 or SCR_1.RE = 1 while SMR_1.C/A = 0, SCR_1.CKE [1, 0] = 01 or while SMR_1.C/A = 1, SCR_1.CKE 1 = 0
		PO4_OE	PO4		NDERL.NDER4 = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P2	3	TIOCD3_OE	TIOCD3		TPU.TMDR.BFB = 0, TPU.TIORL3.IOD3 = 0, TPU.TIORL3.IOD[1,0] = 01/10/11
		PO3_OE	PO3		NDERL.NDER3 = 1
	2	TIOCC3_OE	TIOCC3		TPU.TMDR.BFA = 0, TPU.TIORL3.IOC3 = 0, TPU.TIORL3.IOD[1,0] = 01/10/11
		TMO0_OE	TMO0		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
		TxD0_OE	TxD0		SCR.TE = 1
		PO2_OE	PO2		NDERL.NDER2 = 1
	1	TIOCA3_OE	TIOCA3		TPU.TIORH3.IOA3 = 0, TPU.TIORH3.IOA[1,0] = 01/10/11
		PO1_OE	PO1		NDERL.NDER1 = 1
	0	TIOCB3_OE	TIOCB3		TPU.TIORH3.IOB3 = 0, TPU.TIORH3.IOB[1,0] = 01/10/11
		SCK0_OE	SCK0		When SCMR_0.SMIF = 1: SCR_0.TE = 1 or SCR_0.RE = 1 while SMR_0.GM = 0, SCR_0.CKE [1, 0] = 01 or while SMR_0.GM = 1 When SCMR_0.SMIF = 0: SCR_0.TE = 1 or SCR_0.RE = 1 while SMR_0.C/A = 0, SCR_0.CKE [1, 0] = 01 or while SMR_0.C/A = 1, SCR_0.CKE 1 = 0
		PO0_OE	PO0		NDERL.NDER0 = 1
P3	7	TIOCB2_OE	TIOCB2		TPU.TIOR2.IOB3 = 0, TPU.TIOR2.IOB[1,0] = 01/10/11
		PO15_OE	PO15		NDERH.NDER15 = 1
	6	TIOCA2_OE	TIOCA2		TPU.TIOR2.IOA3 = 0, TPU.TIOR2.IOA[1,0] = 01/10/11
		PO14_OE	PO14		NDERH.NDER14 = 1
	5	$\overline{\text{DACK1B}}$_OE	$\overline{\text{DACK1}}$	PFCR7.DMAS1[A,B] = 01	DACR.AMS = 1, DMDR.DACKE = 1
		TIOCB1_OE	TIOCB1		TPU.TIOR1.IOB3 = 0, TPU.TIOR1.IOB[1,0] = 01/10/11
		PO13_OE	PO13		NDERH.NDER13 = 1
	4	$\overline{\text{TEND1B}}$_OE	$\overline{\text{TEND1}}$	PFCR7.DMAS1[A,B] = 01	DMDR.TENDE = 1
		TIOCA1_OE	TIOCA1		TPU.TIOR1.IOA3 = 0, TPU.TIOR1.IOA[1,0] = 01/10/11
		PO12_OE	PO12		NDERH.NDER12 = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P3	3	TIOCD0_OE	TIOCD0		TPU.TMDR.BFB = 0, TPU.TIORL0.IOD3 = 0, TPU.TIORL0.IOD[1,0] = 01/10/11
		PO11_OE	PO11		NDERH.NDER11 = 1
	2	$\overline{\text{DACK0B}}$_OE	$\overline{\text{DACK0}}$	PFCR7.DMAS0[A,B] = 01	DACR.AMS = 1, DMDR.DACKE = 1
		TIOCC0_OE	TIOCC0		TPU.TMDR.BFA = 0, TPU.TIORL0.IOC3 = 0, TPU.TIORL0.IOD[1,0] = 01/10/11
		PO10_OE	PO10		NDERH.NDER10 = 1
	1	$\overline{\text{TEND0B}}$_OE	$\overline{\text{TEND0}}$	PFCR7.DMAS0[A,B] = 01	DMDR.TENDE = 1
		TIOCB0_OE	TIOCB0		TPU.TIORH0.IOB3 = 0, TPU.TIORH0.IOB[1,0] = 01/10/11
		PO9_OE	PO9		NDERH.NDER9 = 1
	0	TIOCA0_OE	TIOCA0		TPU.TIORH0.IOA3 = 0, TPU.TIORH0.IOA[1,0] = 01/10/11
		PO8_OE	PO8		NDERH.NDER8 = 1
P6	5	$\overline{\text{DACK3}}$_OE	$\overline{\text{DACK3}}$	PFCR7.DMAS3[A,B] = 01	DACR.AMS = 1, DMDR.DACKE = 1
		TMO3_OE	TMO3		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	4	$\overline{\text{TEND3}}$_OE	$\overline{\text{TEND3}}$	PFCR7.DMAS3[A,B] = 01	DMDR.TENDE = 1
	2	$\overline{\text{DACK2}}$_OE	$\overline{\text{DACK2}}$	PFCR7.DMAS2[A,B] = 01	DACR.AMS = 1, DMDR.DACKE = 1
		TMO2_OE	TMO2		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
		SCK4_OE	SCK4		When SCMR_4.SMIF = 1: SCR_4.TE = 1 or SCR_4.RE = 1 while SMR_4.GM = 0, SCR_4.CKE [1, 0] = 01 or while SMR_4.GM = 1 When SCMR_4.SMIF = 0: SCR_4.TE = 1 or SCR_4.RE = 1 while SMR_4.C/A = 0, SCR_4.CKE [1, 0] = 01 or while SMR_4.C/A = 1, SCR_4.CKE 1 = 0
	1	$\overline{\text{TEND2}}$_OE	$\overline{\text{TEND2}}$	PFCR7.DMAS2[A,B] = 01	DMDR.TENDE = 1
	0	TxD4_OE	TxD4		SCR.TE = 1
PA	7	Bϕ_OE	Bϕ		PADDR.PA7DDR = 1, SCKCR.POSEL1 = 0
	6	$\overline{\text{AH}}$_OE	$\overline{\text{AH}}$		SYSCR.EXPE = 1, MPXCR.MPXEn (n = 7 to 3) = 1
		$\overline{\text{BSB}}$_OE	$\overline{\text{BS}}$	PFCR2.BSS = 1	SYSCR.EXPE = 1, PFCR2.BSE = 1
		$\overline{\text{AS}}$_OE	$\overline{\text{AS}}$		SYSCR.EXPE = 1, PFCR2.ASOE = 1
	5	$\overline{\text{RD}}$_OE	$\overline{\text{RD}}$		SYSCR.EXPE = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PA	4	$\overline{\text{LUB}}$_OE	$\overline{\text{LUB}}$		SYSCR.EXPE = 1, PFCR6.LHWROE = 1 or SRAMCR.BCSELn = 1
		$\overline{\text{LHWR}}$_OE	$\overline{\text{LHWR}}$		SYSCR.EXPE = 1, PFCR6.LHWROE = 1
	3	$\overline{\text{LLB}}$_OE	$\overline{\text{LLB}}$		SYSCR.EXPE = 1, SRAMCR.BCSELn = 1
		$\overline{\text{LLWR}}$_OE	$\overline{\text{LLWR}}$		SYSCR.EXPE = 1
	1	$\overline{\text{BACK}}$_OE	$\overline{\text{BACK}}$		SYSCR.EXPE = 1,BCR1.BRLE = 1
		(RD/$\overline{\text{WR}}$)_OE	RD/$\overline{\text{WR}}$		SYSCR.EXPE = 1, PFCR2.REWRE = 1 or SRAMCR.BCSELn = 1
	0	$\overline{\text{BSA}}$_OE	$\overline{\text{BS}}$	PFCR2.BSS = 0	SYSCR.EXPE = 1, PFCR2.BSE = 1
		$\overline{\text{BREQO}}$_OE	$\overline{\text{BREQO}}$		SYSCR.EXPE = 1, BCR1.BRLE = 1, BCR1.BREQOE = 1
PB	3	$\overline{\text{CS3}}$_OE	$\overline{\text{CS3}}$		SYSCR.EXPE = 1, PFCR0.CS3E = 1
		$\overline{\text{CS7A}}$_OE	$\overline{\text{CS7}}$	PFCR1.CS7S[A,B] = 00	SYSCR.EXPE = 1, PFCR0.CS7E = 1
	2	$\overline{\text{CS2A}}$_OE	$\overline{\text{CS2}}$	PFCR2.CS2S = 0	SYSCR.EXPE = 1, PFCR0.CS2E = 1
		$\overline{\text{CS6A}}$_OE	$\overline{\text{CS6}}$	PFCR1.CS6S[A,B] = 00	SYSCR.EXPE = 1, PFCR0.CS6E = 1
	1	$\overline{\text{CS1}}$_OE	$\overline{\text{CS1}}$		SYSCR.EXPE = 1, PFCR0.CS1E = 1
		$\overline{\text{CS2B}}$_OE	$\overline{\text{CS2}}$	PFCR2.CS2S = 1	SYSCR.EXPE = 1, PFCR0.CS2E = 1
		$\overline{\text{CS5A}}$_OE	$\overline{\text{CS5}}$	PFCR1.CS5S[A,B] = 00	SYSCR.EXPE = 1, PFCR0.CS5E = 1
		$\overline{\text{CS6B}}$_OE	$\overline{\text{CS6}}$	PFCR1.CS6S[A,B] = 01	SYSCR.EXPE = 1, PFCR0.CS6E = 1
		$\overline{\text{CS7B}}$_OE	$\overline{\text{CS7}}$	PFCR1.CS7S[A,B] = 01	SYSCR.EXPE = 1, PFCR0.CS7E = 1
	0	$\overline{\text{CS0}}$_OE	$\overline{\text{CS0}}$		SYSCR.EXPE = 1, PFCR0.CS0E = 1
		$\overline{\text{CS4}}$_OE	$\overline{\text{CS4}}$		SYSCR.EXPE = 1, PFCR0.CS4E = 1
		$\overline{\text{CS5B}}$_OE	$\overline{\text{CS5}}$	PFCR1.CS5S[A,B] = 01	SYSCR.EXPE = 1, PFCR0.CS5E = 1
PD	7	A7_OE	A7		SYSCR.EXPE = 1, PDDDR.PD7DDR = 1
	6	A6_OE	A6		SYSCR.EXPE = 1, PDDDR.PD6DDR = 1
	5	A5_OE	A5		SYSCR.EXPE = 1, PDDDR.PD5DDR = 1
	4	A4_OE	A4		SYSCR.EXPE = 1, PDDDR.PD4DDR = 1
	3	A3_OE	A3		SYSCR.EXPE = 1, PDDDR.PD3DDR = 1
	2	A2_OE	A2		SYSCR.EXPE = 1, PDDDR.PD2DDR = 1
	1	A1_OE	A1		SYSCR.EXPE = 1, PDDDR.PD1DDR = 1
	0	A0_OE	A0		SYSCR.EXPE = 1, PDDDR.PD0DDR = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PE	7	A15_OE	A15		SYSCR.EXPE = 1, PDDDR.PE7DDR = 1
	6	A14_OE	A14		SYSCR.EXPE = 1, PDDDR.PE6DDR = 1
	5	A13_OE	A13		SYSCR.EXPE = 1, PDDDR.PE5DDR = 1
	4	A12_OE	A12		SYSCR.EXPE = 1, PDDDR.PE4DDR = 1
	3	A11_OE	A11		SYSCR.EXPE = 1, PDDDR.PE3DDR = 1
	2	A10_OE	A10		SYSCR.EXPE = 1, PDDDR.PE2DDR = 1
	1	A9_OE	A9		SYSCR.EXPE = 1, PDDDR.PE1DDR = 1
	0	A8_OE	A8		SYSCR.EXPE = 1, PDDDR.PE0DDR = 1
PF	7	A23_OE	A23		SYSCR.EXPE = 1, PFCR4.A23E = 1
	6	A22_OE	A22		SYSCR.EXPE = 1, PFCR4.A22E = 1
	5	A21_OE	A21		SYSCR.EXPE = 1, PFCR4.A21E = 1
	4	A20_OE	A20		SYSCR.EXPE = 1, PFCR4.A20E = 1
	3	A19_OE	A19		SYSCR.EXPE = 1, PFCR4.A19E = 1
	2	A18_OE	A18		SYSCR.EXPE = 1, PFCR4.A18E = 1
	1	A17_OE	A17		SYSCR.EXPE = 1, PFCR4.A17E = 1
	0	A16_OE	A16		SYSCR.EXPE = 1, PFCR4.A16E = 1
PH	7	D7_E	D7		SYSCR.EXPE = 1
	6	D6_E	D6		SYSCR.EXPE = 1
	5	D5_E	D5		SYSCR.EXPE = 1
	4	D4_E	D4		SYSCR.EXPE = 1
	3	D3_E	D3		SYSCR.EXPE = 1
	2	D2_E	D2		SYSCR.EXPE = 1
	1	D1_E	D1		SYSCR.EXPE = 1
	0	D0_E	D0		SYSCR.EXPE = 1
PI	7	D15_E	D15		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	6	D14_E	D14		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	5	D13_E	D13		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	4	D12_E	D12		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	3	D11_E	D11		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	2	D10_E	D10		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	1	D9_E	D9		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	0	D8_E	D8		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01

9.3 Port Function Controller

The port function controller controls the I/O ports.

The port function controller incorporates the following registers.

- Port function control register 0 (PFCR0)
- Port function control register 1 (PFCR1)
- Port function control register 2 (PFCR2)
- Port function control register 4 (PFCR4)
- Port function control register 6 (PFCR6)
- Port function control register 7 (PFCR7)
- Port function control register 9 (PFCR9)
- Port function control register B (PFCRB)
- Port function control register C (PFCRC)

9.3.1 Port Function Control Register 0 (PFCR0)

PFCR0 enables/disables the $\overline{CS}$ output.

Bit	7	6	5	4	3	2	1	0
Bit Name	CS7E	CS6E	CS5E	CS4E	CS3E	CS2E	CS1E	CS0E
Initial Value	0	0	0	0	0	0	0	Undefined*
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * 1 in external extended mode; 0 in other modes.

Bit	Bit Name	Initial Value	R/W	Description
7	CS7E	0	R/W	CS7 to CS0 Enable
6	CS6E	0	R/W	These bits enable/disable the corresponding $\overline{CSn}$ output.
5	CS5E	0	R/W	0: Pin functions as I/O port
4	CS4E	0	R/W	1: Pin functions as $\overline{CSn}$ output pin
3	CS3E	0	R/W	(n = 7 to 0)
2	CS2E	0	R/W	
1	CS1E	0	R/W	
0	CS0E	Undefined*	R/W	

Note: * 1 in external extended mode; 0 in other modes.

9.3.2 Port Function Control Register 1 (PFCR1)

PFCR1 selects the $\overline{CS}$ output pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	CS7SA	CS7SB	CS6SA	CS6SB	CS5SA	CS5SB	—	—
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	Bit Name	Initial Value	R/W	Description
7	CS7SA*	0	R/W	$\overline{CS7}$ Output Pin Select
6	CS7SB*	0	R/W	Selects the output pin for $\overline{CS7}$ when $\overline{CS7}$ output is enabled (CS7E = 1) 00: Specifies pin PB3 as $\overline{CS7}$-A output 01: Specifies pin PB1 as $\overline{CS7}$-B output 10: Setting prohibited 11: Setting prohibited
5	CS6SA*	0	R/W	$\overline{CS6}$ Output Pin Select
4	CS6SB*	0	R/W	Selects the output pin for $\overline{CS6}$ when $\overline{CS6}$ output is enabled (CS6E = 1) 00: Specifies pin PB2 as $\overline{CS6}$-A output 01: Specifies pin PB1 as $\overline{CS6}$-B output 10: Setting prohibited 11: Setting prohibited
3	CS5SA*	0	R/W	$\overline{CS5}$ Output Pin Select
2	CS5SB*	0	R/W	Selects the output pin for $\overline{CS5}$ when $\overline{CS5}$ output is enabled (CS5E = 1) 00: Specifies pin PB1 as $\overline{CS5}$-A output 01: Specifies pin PB0 as $\overline{CS5}$-B output 10: Setting prohibited 11: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description
1, 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

Note: * If multiple $\overline{CS}$ outputs are specified to a single pin according to the $\overline{CSn}$ output pin select bits (n = 5 to 7), multiple $\overline{CS}$ signals are output from the pin. For details, see section 6.5.3, Chip Select Signals.

9.3.3 Port Function Control Register 2 (PFCR2)

PFCR1 selects the $\overline{CS}$ output pin, enables/disables bus control I/O, and selects the bus control I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	CS2S	BSS	BSE	—	RDWRE	ASOE	—
Initial Value	0	0	0	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6	CS2S*[1]	0	R/W	$\overline{CS2}$ Output Pin Select Selects the output pin for $\overline{CS2}$ when $\overline{CS2}$ output is enabled (CS2E = 1) 0: Specifies pin PB2 as $\overline{CS2}$-A output pin 1: Specifies pin PB1 as $\overline{CS2}$-B output pin
5	BSS	0	R/W	$\overline{BS}$ Output Pin Select Selects the $\overline{BS}$ output pin 0: Specifies pin PA0 as $\overline{BS}$-A output pin 1: Specifies pin PA6 as $\overline{BS}$-B output pin

Bit	Bit Name	Initial Value	R/W	Description
4	BSE	0	R/W	$\overline{BS}$ Output Enable Enables/disables the $\overline{BS}$ output 0: Disables the $\overline{BS}$ output 1: Enables the $\overline{BS}$ output
3	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
2	RDWRE*2	0	R/W	RD/$\overline{WR}$ Output Enable Enables/disables the RD/$\overline{WR}$ output 0: Disables the RD/$\overline{WR}$ output 1: Enables the RD/$\overline{WR}$ output
1	ASOE	1	R/W	$\overline{AS}$ Output Enable Enables/disables the $\overline{AS}$ output 0: Specifies pin PA6 as I/O port 1: Specifies pin PA6 as $\overline{AS}$ output pin
0	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

Notes: 1. If multiple $\overline{CS}$ outputs are specified to a single pin according to the $\overline{CS2}$ output pin select bit, multiple $\overline{CS}$ signals are output from the pin. For details, see section 6.5.3, Chip Select Signals.

2. If an area is specified as a byte control SDRAM space, the pin functions as RD/$\overline{WR}$ output regardless of the RDWRE bit value.

9.3.4 Port Function Control Register 4 (PFCR4)

PFCR4 enables/disables the address output.

Bit	7	6	5	4	3	2	1	0
Bit Name	A23E	A22E	A21E	A20E	A19E	A18E	A17E	A16E
Initial Value	0	0	0	1/0*	1/0*	1/0*	1/0*	1/0*
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * The initial value depends on the operating mode: 1 in on-chip ROM disabled mode and 0 in on-chip ROM enabled mode.

Bit	Bit Name	Initial Value	R/W	Description
7	A23E	0	R/W	Address A23 Enable Enables/disables the address output (A23) 0: Disables the A23 output 1: Enables the A23 output
6	A22E	0	R/W	Address A22 Enable Enables/disables the address output (A22) 0: Disables the A22 output 1: Enables the A22 output
5	A21E	0	R/W	Address A21 Enable Enables/disables the address output (A21) 0: Disables the A21 output 1: Enables the A21 output
4	A20E	1/0*	R/W	Address A20 Enable Enables/disables the address output (A20) 0: Disables the A20 output 1: Enables the A20 output
3	A19E	1/0*	R/W	Address A19 Enable Enables/disables the address output (A19) 0: Disables the A19 output 1: Enables the A19 output

Bit	Bit Name	Initial Value	R/W	Description
2	A18E	1/0*	R/W	Address A18 Enable Enables/disables the address output (A18) 0: Disables the A18 output 1: Enables the A18 output
1	A17E	1/0*	R/W	Address A17 Enable Enables/disables the address output (A17) 0: Disables the A17 output 1: Enables the A17 output
0	A16E	1/0*	R/W	Address A16 Enable Enables/disables the address output (A16) 0: Disables the A16 output 1: Enables the A16 output

9.3.5 Port Function Control Register 6 (PFCR6)

PFCR6 selects the TPU clock input pin.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	LHWROE	—	—	TCLKS	—	—	—
Initial Value	1	1	1	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
6	LHWROE	1	R/W	$\overline{\text{LHWR}}$ Output Enable Enables/disables $\overline{\text{LHWR}}$ output (valid in external extended mode). 0: Specifies pin PA4 as I/O port 1: Specifies pin PA4 as $\overline{\text{LHWR}}$ output pin

Bit	Bit Name	Initial Value	R/W	Description
5	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
4	—	0	R	Reserved This is a read-only bit and cannot be modified.
3	TCLKS	0	R/W	TPU External Clock Input Pin Select Selects the TPU external clock input pins. 0: Specifies pins P32, P33, P35, and P37 as external clock inputs 1: Specifies pins P14 to P17 as external clock inputs
2 to 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

9.3.6 Port Function Control Register 7 (PFCR7)

PFCR7 selects the DMAC I/O pins ($\overline{\text{DREQ}}$, $\overline{\text{DACK}}$, and $\overline{\text{TEND}}$).

Bit	7	6	5	4	3	2	1	0
Bit Name	DMAS3A	DMAS3B	DMAS2A	DMAS2B	DMAS1A	DMAS1B	DMAS0A	DMAS0B
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	Bit Name	Initial Value	R/W	Description
7	DMAS3A	0	R/W	DMAC control pin select
6	DMAS3B	0	R/W	Selects the I/O port to control DMAC_3. 00: Setting prohibited 01: Specifies pins P63 to P65 as DMAC control pins 10: Setting prohibited 11: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description
5	DMAS2A	0	R/W	DMAC control pin select
4	DMAS2B	0	R/W	Selects the I/O port to control DMAC_2. 00: Setting prohibited 01: Specifies pins P60 to P62 as DMAC control pins 10: Setting prohibited 11: Setting prohibited
3	DMAS1A	0	R/W	DMAC control pin select
2	DMAS1B	0	R/W	Selects the I/O port to control DMAC_1. 00: Specifies pins P14 to P16 as DMAC control pins 01: Specifies pins P33 to P35 as DMAC control pins 10: Setting prohibited 11: Setting prohibited
1	DMAS0A	0	R/W	DMAC control pin select
0	DMAS0B	0	R/W	Selects the I/O port to control DMAC_0. 00: Specifies pins P10 to P12 as DMAC control pins 01: Specifies pins P30 to P32 as DMAC control pins 10: Setting prohibited 11: Setting prohibited

9.3.7 Port Function Control Register 9 (PFCR9)

PFCR9 selects the multiple functions for the TPU I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B
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	Bit Name	Initial Value	R/W	Description
7	TPUMS5	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA5 function 0: Specifies pin P26 as output compare output and input capture 1: Specifies P27 as input capture input and P26 as output compare
6	TPUMS4	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA4 function 0: Specifies P25 as output compare output and input capture 1: Specifies P24 as input capture input and P25 as output compare
5	TPUMS3A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA3 function 0: Specifies P21 as output compare output and input capture 1: Specifies P20 as input capture input and P21 as output compare
4	TPUMS3B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC3 function 0: Specifies P22 as output compare output and input capture 1: Specifies P23 as input capture input and P22 as output compare

Bit	Bit Name	Initial Value	R/W	Description
3	TPUMS2	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA2 function 0: Specifies P36 as output compare output and input capture 1: Specifies P37 as input capture input and P36 as output compare
2	TPUMS1	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA1 function 0: Specifies P34 as output compare output and input capture 1: Specifies P35 as input capture input and P34 as output compare
1	TPUMS0A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA0 function 0: Specifies P30 as output compare output and input capture 1: Specifies P31 as input capture input and P30 as output compare
0	TPUMS0B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC0 function 0: Specifies P32 as output compare output and input capture 1: Specifies P33 as input capture input and P32 as output compare

9.3.8 Port Function Control Register B (PFCRB)

PFCRB selects the input pins for $\overline{\text{IRQ11}}$ to $\overline{\text{IRQ8}}$.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	ITS11	ITS10	ITS9	ITS8
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	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
3	ITS11	0	R/W	$\overline{\text{IRQ11}}$ Pin Select Selects an input pin for $\overline{\text{IRQ11}}$. 0: Selects pin P23 as $\overline{\text{IRQ11}}$-A input 1: Selects pin P63 as $\overline{\text{IRQ11}}$-B input
2	ITS10	0	R/W	$\overline{\text{IRQ10}}$ Pin Select Selects an input pin for $\overline{\text{IRQ10}}$. 0: Selects pin P22 as $\overline{\text{IRQ10}}$-A input 1: Selects pin P62 as $\overline{\text{IRQ10}}$-B input
1	ITS9	0	R/W	$\overline{\text{IRQ9}}$ Pin Select Selects an input pin for $\overline{\text{IRQ9}}$. 0: Selects pin P21 as $\overline{\text{IRQ9}}$-A input 1: Selects pin P61 as $\overline{\text{IRQ9}}$-B input
0	ITS8	0	R/W	$\overline{\text{IRQ8}}$ Pin Select Selects an input pin for $\overline{\text{IRQ8}}$. 0: Selects pin P20 as $\overline{\text{IRQ8}}$-A input 1: Selects pin P60 as $\overline{\text{IRQ8}}$-B input

9.3.9 Port Function Control Register C (PFCRC)

PFCRC selects input pins for $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.

Bit	7	6	5	4	3	2	1	0
Bit Name	ITS7	ITS6	ITS5	ITS4	ITS3	ITS2	ITS1	ITS0
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	Bit Name	Initial Value	R/W	Description
7	ITS7	0	R/W	$\overline{\text{IRQ7}}$ Pin Select Selects an input pin for $\overline{\text{IRQ7}}$. 0: Selects pin P17 as $\overline{\text{IRQ7}}$-A input 1: Selects pin P57 as $\overline{\text{IRQ7}}$-B output
6	ITS6	0	R/W	$\overline{\text{IRQ6}}$ Pin Select Selects an input pin for $\overline{\text{IRQ6}}$. 0: Selects pin P16 as $\overline{\text{IRQ6}}$-A input 1: Selects pin P56 as $\overline{\text{IRQ6}}$-B output
5	ITS5	0	R/W	$\overline{\text{IRQ5}}$ Pin Select Selects an input pin for $\overline{\text{IRQ5}}$. 0: Selects pin P15 as $\overline{\text{IRQ5}}$-A input 1: Selects pin P55 as $\overline{\text{IRQ5}}$-B output
4	ITS4	0	R/W	$\overline{\text{IRQ4}}$ Pin Select Selects an input pin for $\overline{\text{IRQ4}}$. 0: Selects pin P14 as $\overline{\text{IRQ4}}$-A input 1: Selects pin P54 as $\overline{\text{IRQ4}}$-B output
3	ITS3	0	R/W	$\overline{\text{IRQ3}}$ Pin Select Selects an input pin for $\overline{\text{IRQ3}}$. 0: Selects pin P13 as $\overline{\text{IRQ3}}$-A input 1: Selects pin P53 as $\overline{\text{IRQ3}}$-B output

Bit	Bit Name	Initial Value	R/W	Description
2	ITS2	0	R/W	$\overline{\text{IRQ2}}$ Pin Select Selects an input pin for $\overline{\text{IRQ2}}$. 0: Selects pin P12 as $\overline{\text{IRQ2}}$-A input 1: Selects pin P52 as $\overline{\text{IRQ2}}$-B output
1	ITS1	0	R/W	$\overline{\text{IRQ1}}$ Pin Select Selects an input pin for $\overline{\text{IRQ1}}$. 0: Selects pin P11 as $\overline{\text{IRQ1}}$-A input 1: Selects pin P51 as $\overline{\text{IRQ1}}$-B output
0	ITS0	0	R/W	$\overline{\text{IRQ0}}$ Pin Select Selects an input pin for $\overline{\text{IRQ0}}$. 0: Selects pin P10 as $\overline{\text{IRQ0}}$-A input 1: Selects pin P50 as $\overline{\text{IRQ0}}$-B output

9.4 Usage Notes

9.4.1 Notes on Input Buffer Control Register (ICR) Settings

- When the ICR setting is changed, the LSI may malfunction due to an edge occurred internally according to the pin state. Before changing the ICR setting, fix the pin state high or disable the input function corresponding to the pin by the on-chip peripheral module settings.
- If an input is enabled by setting ICR while multiple input functions are assigned to the pin, the pin state is reflected in all the inputs. Care must be taken for each module settings for unused input functions.
- When a pin is used as an output, data to be output from the pin will be latched as the pin state if the input function corresponding to the pin is enabled. To use the pin as an output, disable the input function for the pin by setting ICR.

9.4.2 Notes on Port Function Control Register (PFCR) Settings

- Port function controller controls the I/O port.
  Before enabling a port function, select the input/output destination.
- When changing input pins, this LSI may malfunction due to the internal edge.
  To change input pins, the following procedure must be performed.
  - Disable the input function by the corresponding on-chip peripheral module settings
  - Select another input pin by PFCR
  - Enable its input function by the corresponding on-chip peripheral module settings
- If a pin function has both a select bit that modifies the input/output destination and an enable bit that enables the pin function, first specify the input/output destination by the selection bit and then enable the pin function by the enable bit.

Section 10 16-Bit Timer Pulse Unit (TPU)

This LSI has an on-chip 16-bit timer pulse unit (TPU) that comprises six 16-bit timer channels.

Tables 10.1 lists the 16-bit timer unit functions and figure 10.1 is a block diagram.

10.1 Features

- Maximum 16-pulse input/output
- Selection of eight counter input clocks for each channel
- The following operations can be set for each channel:
  - — Waveform output at compare match
  - — Input capture function
  - — Counter clear operation
  - — Synchronous operations:
    - Multiple timer counters (TCNT) can be written to simultaneously
    - Simultaneous clearing by compare match and input capture possible
    - Simultaneous input/output for registers possible by counter synchronous operation
    - Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
- Cascaded operation
- Fast access via internal 16-bit bus
- 26 interrupt sources
- Automatic transfer of register data
- Programmable pulse generator (PPG) output trigger can be generated
- Conversion start trigger for the A/D converter can be generated
- Module stop state specifiable

Table 10.1 TPU Functions

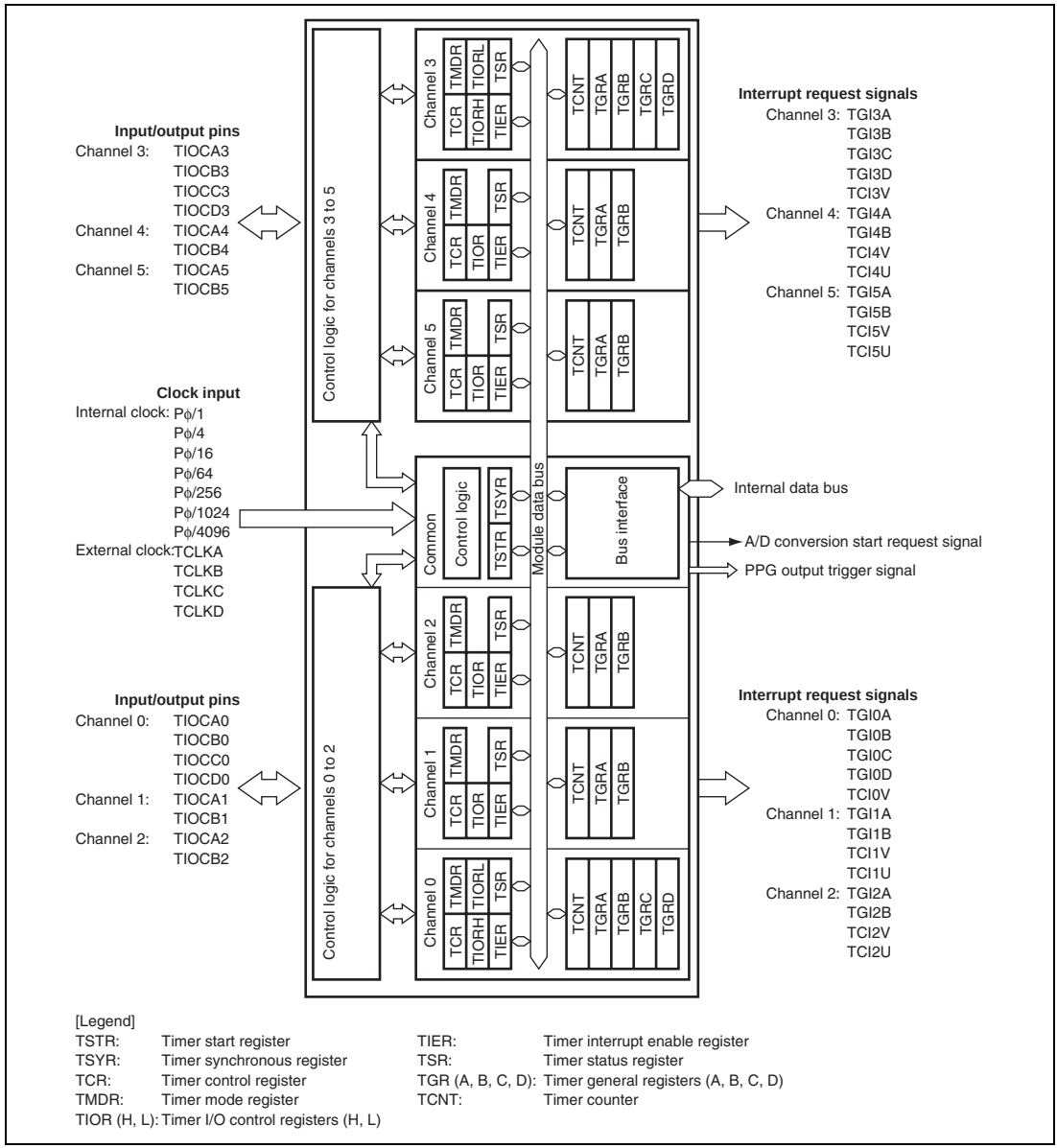
Item		Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
Count clock		Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 TCLKA TCLKB TCLKC TCLKD	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKA TCLKB TCNT2	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKA TCLKB TCLKC	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 Pϕ/1024 Pϕ/4096 TCLKA	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKA TCLKC TCNT5	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKA TCLKC TCLKD
General registers (TGR)		TGRA_0 TGRB_0	TGRA_1 TGRB_1	TGRA_2 TGRB_2	TGRA_3 TGRB_3	TGRA_4 TGRB_4	TGRA_5 TGRB_5
General registers/ buffer registers		TGRC_0 TGRD_0	—	—	TGRC_3 TGRD_3	—	—
I/O pins		TIOCA0 TIOCB0 TIOCC0 TIOCD0	TIOCA1 TIOCB1	TIOCA2 TIOCB2	TIOCA3 TIOCB3 TIOCC3 TIOCD3	TIOCA4 TIOCB4	TIOCA5 TIOCB5
Counter clear function		TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture
Compare match output	0 output	O	O	O	O	O	O
	1 output	O	O	O	O	O	O
	Toggle output	O	O	O	O	O	O
Input capture function		O	O	O	O	O	O
Synchronous operation		O	O	O	O	O	O
PWM mode		O	O	O	O	O	O
Phase counting mode		—	O	O	—	O	O
Buffer operation		O	—	—	O	—	—
DTC activation		TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
DMAC activation	TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture	TGRA_5 compare match or input capture
A/D converter trigger	TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture	TGRA_5 compare match or input capture
PPG trigger	TGRA_0/ TGRB_0 compare match or input capture	TGRA_1/ TGRB_1 compare match or input capture	TGRA_2/ TGRB_2 compare match or input capture	TGRA_3/ TGRB_3 compare match or input capture	—	—
Interrupt sources	5 sources	4 sources	4 sources	5 sources	4 sources	4 sources
	Compare match or input capture 0A	Compare match or input capture 1A	Compare match or input capture 2A	Compare match or input capture 3A	Compare match or input capture 4A	Compare match or input capture 5A
	Compare match or input capture 0B	Compare match or input capture 1B	Compare match or input capture 2B	Compare match or input capture 3B	Compare match or input capture 4B	Compare match or input capture 5B
	Compare match or input capture 0C	Overflow	Overflow	Compare match or input capture 3C	Overflow	Overflow
		Underflow	Underflow		Underflow	Underflow
	Compare match or input capture 0D			Compare match or input capture 3D		
	Overflow			Overflow		

[Legend]

O : Possible

— : Not possible



[Legend]

TSTR:	Timer start register	TIER:	Timer interrupt enable register
TSYR:	Timer synchronous register	TSR:	Timer status register
TCR:	Timer control register	TGR (A, B, C, D):	Timer general registers (A, B, C, D)
TMDR:	Timer mode register	TCNT:	Timer counter
TIOR (H, L):	Timer I/O control registers (H, L)		

Figure 10.1 Block Diagram of TPU

10.2 Input/Output Pins

Table 10.2 shows TPU pin configurations.

Table 10.2 Pin Configuration

Channel	Symbol	I/O	Function
All	TCLKA	Input	External clock A input pin (Channel 1 and 5 phase counting mode A phase input)
	TCLKB	Input	External clock B input pin (Channel 1 and 5 phase counting mode B phase input)
	TCLKC	Input	External clock C input pin (Channel 2 and 4 phase counting mode A phase input)
	TCLKD	Input	External clock D input pin (Channel 2 and 4 phase counting mode B phase input)
0	TIOCA0	I/O	TGRA_0 input capture input/output compare output/PWM output pin
	TIOCB0	I/O	TGRB_0 input capture input/output compare output/PWM output pin
	TIOCC0	I/O	TGRC_0 input capture input/output compare output/PWM output pin
	TIOCD0	I/O	TGRD_0 input capture input/output compare output/PWM output pin
1	TIOCA1	I/O	TGRA_1 input capture input/output compare output/PWM output pin
	TIOCB1	I/O	TGRB_1 input capture input/output compare output/PWM output pin
2	TIOCA2	I/O	TGRA_2 input capture input/output compare output/PWM output pin
	TIOCB2	I/O	TGRB_2 input capture input/output compare output/PWM output pin
3	TIOCA3	I/O	TGRA_3 input capture input/output compare output/PWM output pin
	TIOCB3	I/O	TGRB_3 input capture input/output compare output/PWM output pin
	TIOCC3	I/O	TGRC_3 input capture input/output compare output/PWM output pin
	TIOCD3	I/O	TGRD_3 input capture input/output compare output/PWM output pin
4	TIOCA4	I/O	TGRA_4 input capture input/output compare output/PWM output pin
	TIOCB4	I/O	TGRB_4 input capture input/output compare output/PWM output pin
5	TIOCA5	I/O	TGRA_5 input capture input/output compare output/PWM output pin
	TIOCB5	I/O	TGRB_5 input capture input/output compare output/PWM output pin

10.3 Register Descriptions

The TPU has the following registers in each channel.

- Channel 0
  - Timer control register_0 (TCR_0)
  - Timer mode register_0 (TMDR_0)
  - Timer I/O control register H_0 (TIORH_0)
  - Timer I/O control register L_0 (TIORL_0)
  - Timer interrupt enable register_0 (TIER_0)
  - Timer status register_0 (TSR_0)
  - Timer counter_0 (TCNT_0)
  - Timer general register A_0 (TGRA_0)
  - Timer general register B_0 (TGRB_0)
  - Timer general register C_0 (TGRC_0)
  - Timer general register D_0 (TGRD_0)
- Channel 1
  - Timer control register_1 (TCR_1)
  - Timer mode register_1 (TMDR_1)
  - Timer I/O control register _1 (TIOR_1)
  - Timer interrupt enable register_1 (TIER_1)
  - Timer status register_1 (TSR_1)
  - Timer counter_1 (TCNT_1)
  - Timer general register A_1 (TGRA_1)
  - Timer general register B_1 (TGRB_1)
- Channel 2
  - Timer control register_2 (TCR_2)
  - Timer mode register_2 (TMDR_2)
  - Timer I/O control register_2 (TIOR_2)
  - Timer interrupt enable register_2 (TIER_2)
  - Timer status register_2 (TSR_2)
  - Timer counter_2 (TCNT_2)
  - Timer general register A_2 (TGRA_2)
  - Timer general register B_2 (TGRB_2)

- Channel 3
  - Timer control register_3 (TCR_3)
  - Timer mode register_3 (TMDR_3)
  - Timer I/O control register H_3 (TIORH_3)
  - Timer I/O control register L_3 (TIORL_3)
  - Timer interrupt enable register_3 (TIER_3)
  - Timer status register_3 (TSR_3)
  - Timer counter_3 (TCNT_3)
  - Timer general register A_3 (TGRA_3)
  - Timer general register B_3 (TGRB_3)
  - Timer general register C_3 (TGRC_3)
  - Timer general register D_3 (TGRD_3)
- Channel 4
  - Timer control register_4 (TCR_4)
  - Timer mode register_4 (TMDR_4)
  - Timer I/O control register _4 (TIOR_4)
  - Timer interrupt enable register_4 (TIER_4)
  - Timer status register_4 (TSR_4)
  - Timer counter_4 (TCNT_4)
  - Timer general register A_4 (TGRA_4)
  - Timer general register B_4 (TGRB_4)
- Channel 5
  - Timer control register_5 (TCR_5)
  - Timer mode register_5 (TMDR_5)
  - Timer I/O control register_5 (TIOR_5)
  - Timer interrupt enable register_5 (TIER_5)
  - Timer status register_5 (TSR_5)
  - Timer counter_5 (TCNT_5)
  - Timer general register A_5 (TGRA_5)
  - Timer general register B_5 (TGRB_5)
- Common Registers
  - Timer start register (TSTR)
  - Timer synchronous register (TSYR)

10.3.1 Timer Control Register (TCR)

TCR controls the TCNT operation for each channel. The TPU has a total of six TCR registers, one for each channel. TCR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
7	CCLR2	0	R/W	Counter Clear 2 to 0
6	CCLR1	0	R/W	These bits select the TCNT counter clearing source. For details, see tables 10.3 and 10.4.
5	CCLR0	0	R/W	
4	CKEG1	0	R/W	Clock Edge 1 and 0
3	CKEG0	0	R/W	These bits select the input clock edge. For details, see table 10.5. When the input clock is counted using both edges, the input clock period is halved (e.g. Pϕ/4 both edges = Pϕ/2 rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority. Internal clock edge selection is valid when the input clock is Pϕ/4 or slower. This setting is ignored if the input clock is Pϕ/1, or when overflow/underflow of another channel is selected.
2	TPSC2	0	R/W	Timer Prescaler 2 to 0
1	TPSC1	0	R/W	These bits select the TCNT counter clock. The clock source can be selected independently for each channel. For details, see tables 10.6 to 10.11. To select the external clock as the clock source, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 9, I/O Ports.
0	TPSC0	0	R/W	

Table 10.3 CCLR2 to CCLR0 (Channels 0 and 3)

Channel	Bit 7 CCLR2	Bit 6 CCLR1	Bit 5 CCLR0	Description
0, 3	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]
	1	0	0	TCNT clearing disabled
	1	0	1	TCNT cleared by TGRC compare match/input capture*[2]
	1	1	0	TCNT cleared by TGRD compare match/input capture*[2]
	1	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]

Notes: 1. Synchronous operation is selected by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

Table 10.4 CCLR2 to CCLR0 (Channels 1, 2, 4, and 5)

Channel	Bit 7 Reserved *[2]	Bit 6 CCLR1	Bit 5 CCLR0	Description
1, 2, 4, 5	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]

Notes: 1. Synchronous operation is selected by setting the SYNC bit in TSYR to 1.
2. Bit 7 is reserved in channels 1, 2, 4, and 5. It is always read as 0 and cannot be modified.

Table 10.5 Input Clock Edge Selection

Clock Edge Selection		Input Clock	
CKEG1	**CKEG0**	**Internal Clock**	**External Clock**
0	0	Counted at falling edge	Counted at rising edge
0	1	Counted at rising edge	Counted at falling edge
1	X	Counted at both edges	Counted at both edges

[Legend]
X: Don't care

Table 10.6 TPSC2 to TPSC0 (Channel 0)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
0	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	External clock: counts on TCLKD pin input

Table 10.7 TPSC2 to TPSC0 (Channel 1)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
1	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	Counts on TCNT2 overflow/underflow

Note: This setting is ignored when channel 1 is in phase counting mode.

Table 10.8 TPSC2 to TPSC0 (Channel 2)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
2	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	Internal clock: counts on Pϕ/1024

Note: This setting is ignored when channel 2 is in phase counting mode.

Table 10.9 TPSC2 to TPSC0 (Channel 3)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
3	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	Internal clock: counts on Pϕ/1024
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	Internal clock: counts on Pϕ/4096

Table 10.10 TPSC2 to TPSC0 (Channel 4)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
4	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on Pϕ/1024
	1	1	1	Counts on TCNT5 overflow/underflow

Note: This setting is ignored when channel 4 is in phase counting mode.

Table 10.11 TPSC2 to TPSC0 (Channel 5)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
5	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	External clock: counts on TCLKD pin input

Note: This setting is ignored when channel 5 is in phase counting mode.

10.3.2 Timer Mode Register (TMDR)

TMDR sets the operating mode for each channel. The TPU has six TMDR registers, one for each channel. TMDR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	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

Bit	Bit Name	Initial Value	R/W	Description
7, 6	——	All 1	R	Reserved These are read-only bits and cannot be modified.
5	BFB	0	R/W	Buffer Operation B Specifies whether TGRB is to normally operate, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD input capture/output compare is not generated. In channels 1, 2, 4, and 5, which have no TGRD, bit 5 is reserved. It is always read as 0 and cannot be modified. 0: TGRB operates normally 1: TGRB and TGRD used together for buffer operation
4	BFA	0	R/W	Buffer Operation A Specifies whether TGRA is to normally operate, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated. In channels 1, 2, 4, and 5, which have no TGRC, bit 4 is reserved. It is always read as 0 and cannot be modified. 0: TGRA operates normally 1: TGRA and TGRC used together for buffer operation
3	MD3	0	R/W	Modes 3 to 0
2	MD2	0	R/W	Set the timer operating mode.
1	MD1	0	R/W	MD3 is a reserved bit. The write value should always be 0. For details, see table 10.12 for details.
0	MD0	0	R/W	

Table 10.12 MD3 to MD0

Bit 3 MD3*[1]	Bit 2 MD2*[2]	Bit 1 MD1	Bit 0 MD0	Description
0	0	0	0	Normal operation
0	0	0	1	Reserved
0	0	1	0	PWM mode 1
0	0	1	1	PWM mode 2
0	1	0	0	Phase counting mode 1
0	1	0	1	Phase counting mode 2
0	1	1	0	Phase counting mode 3
0	1	1	1	Phase counting mode 4
1	X	X	X	—

[Legend]

X: Don't care

Notes: 1. MD3 is a reserved bit. The write value should always be 0.

2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

10.3.3 Timer I/O Control Register (TIOR)

TIOR controls TGR. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. Care is required since TIOR is affected by the TMDR setting.

The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified.

When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

To designate the input capture pin in TIOR, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 9, I/O Ports.

- TIORH_0, TIOR_1, TIOR_2, TIORH_3, TIOR_4, TIOR_5

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- TIORL_0, TIORL_3

Bit	7	6	5	4	3	2	1	0
Bit Name	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
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

- TIORH_0, TIOR_1, TIOR_2, TIORH_3, TIOR_4, TIOR_5

Bit	Bit Name	Initial Value	R/W	Description
7	IOB3	0	R/W	I/O Control B3 to B0
6	IOB2	0	R/W	Specify the function of TGRB.
5	IOB1	0	R/W	For details, see tables 10.13, 10.15, 10.16, 10.17, 10.19, and 10.20.
4	IOB0	0	R/W	
3	IOA3	0	R/W	I/O Control A3 to A0
2	IOA2	0	R/W	Specify the function of TGRA.
1	IOA1	0	R/W	For details, see tables 10.21, 10.23, 10.24, 10.25, 10.27, and 10.28.
0	IOA0	0	R/W	

- TIORL_0, TIORL_3:

Bit	Bit Name	Initial Value	R/W	Description
7	IOD3	0	R/W	I/O Control D3 to D0
6	IOD2	0	R/W	Specify the function of TGRD.
5	IOD1	0	R/W	For details, see tables 10.14 and 10.18.
4	IOD0	0	R/W	
3	IOC3	0	R/W	I/O Control C3 to C0
2	IOC2	0	R/W	Specify the function of TGRC.
1	IOC1	0	R/W	For details, see tables 10.22 and 10.26.
0	IOC0	0	R/W	

Table 10.13 TIORH_0

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_0 Function**	**TIOCB0 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB0 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCB0 pin Input capture at both edges
1	1	x	x		Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down*

[Legend]

X: Don't care

Note: When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 count clock, this setting is invalid and input capture is not generated.

Table 10.14 TIORL_0

				Description	
Bit 7 IOD3	**Bit 6 IOD2**	**Bit 5 IOD1**	**Bit 4 IOD0**	**TGRD_0 Function**	**TIOCD0 Pin Function**
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCD0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCD0 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCD0 pin Input capture at both edges
1	1	X	X		Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down*1

[Legend]

X: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR_0 is set to 1 and TGRD_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 10.15 TIOR_1

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_1 Function**	**TIOCB1 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB1 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB1 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCB1 pin Input capture at both edges
1	1	X	X		TGRC_0 compare match/input capture Input capture at generation of TGRC_0 compare match/input capture

[Legend]
X: Don't care

Table 10.16 TIOR_2

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_2 Function**	**TIOCB2 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	X	0	0	Input capture register	Capture input source is TIOCB2 pin Input capture at rising edge
1	X	0	1		Capture input source is TIOCB2 pin Input capture at falling edge
1	X	1	X		Capture input source is TIOCB2 pin Input capture at both edges

[Legend]

X: Don't care

Table 10.17 TIORH_3

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_3 Function**	**TIOCB3 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB3 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCB3 pin Input capture at both edges
1	1	x	x		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down*

[Legend]

X: Don't care

Note: When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pϕ/1 is used as the TCNT_4 count clock, this setting is invalid and input capture is not generated.

Table 10.18 TIORL_3

				Description	
Bit 7 IOD3	**Bit 6 IOD2**	**Bit 5 IOD1**	**Bit 4 IOD0**	**TGRD_3 Function**	**TIOCD3 Pin Function**
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCD3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCD3 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCD3 pin Input capture at both edges
1	1	x	x		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down*1

[Legend]

X: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pϕ/1 is used as the TCNT_4 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR_3 is set to 1 and TGRD_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 10.19 TIOR_4

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_4 Function**	**TIOCB4 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB4 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB4 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCB4 pin Input capture at both edges
1	1	x	x		Capture input source is TGRC_3 compare match/input capture Input capture at generation of TGRC_3 compare match/input capture

[Legend]

X: Don't care

Table 10.20 TIOR_5

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_5 Function**	**TIOCB5 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	x	0	0	Input capture register	Capture input source is TIOCB5 pin Input capture at rising edge
1	x	0	1		Capture input source is TIOCB5 pin Input capture at falling edge
1	x	1	x		Capture input source is TIOCB5 pin Input capture at both edges

[Legend]

X: Don't care

Table 10.21 TIORH_0

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_0 Function**	**TIOCA0 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA0 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA0 pin Input capture at both edges
1	1	X	X		Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down

[Legend]

X: Don't care

Table 10.22 TIORL_0

				Description	
Bit 3 IOC3	**Bit 2 IOC2**	**Bit 1 IOC1**	**Bit 0 IOC0**	**TGRC_0 Function**	**TIOCC0 Pin Function**
0	0	0	0	Output compare register*	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*	Capture input source is TIOCC0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCC0 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCC0 pin Input capture at both edges
1	1	X	X		Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down

[Legend]

X: Don't care

Note: 1. When the BFA bit in TMDR_0 is set to 1 and TGRC_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 10.23 TIOR_1

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_1 Function**	**TIOCA1 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA1 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA1 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA1 pin Input capture at both edges
1	1	X	X		Capture input source is TGRA_0 compare match/input capture Input capture at generation of channel 0/TGRA_0 compare match/input capture

[Legend]

X: Don't care

Table 10.24 TIOR_2

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_2 Function**	**TIOCA2 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	X	0	0	Input capture register	Capture input source is TIOCA2 pin Input capture at rising edge
1	X	0	1		Capture input source is TIOCA2 pin Input capture at falling edge
1	X	1	X		Capture input source is TIOCA2 pin Input capture at both edges

[Legend]

X: Don't care

Table 10.25 TIORH_3

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_3 Function**	**TIOCA3 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA3 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA3 pin Input capture at both edges
1	1	X	X		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down

[Legend]

X: Don't care

Table 10.26 TIORL_3

				Description	
Bit 3 IOC3	**Bit 2 IOC2**	**Bit 1 IOC1**	**Bit 0 IOC0**	**TGRC_3 Function**	**TIOCC3 Pin Function**
0	0	0	0	Output compare register*	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*	Capture input source is TIOCC3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCC3 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCC3 pin Input capture at both edges
1	1	X	X		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down

[Legend]

X: Don't care

Note: * When the BFA bit in TMDR_3 is set to 1 and TGRC_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 10.27 TIOR_4

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_4 Function**	**TIOCA4 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA4 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA4 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA4 pin Input capture at both edges
1	1	X	X		Capture input source is TGRA_3 compare match/input capture Input capture at generation of TGRA_3 compare match/input capture

[Legend]

X: Don't care

Table 10.28 TIOR_5

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_5 Function**	**TIOCA5 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	X	0	0	Input capture register	Input capture source is TIOCA5 pin Input capture at rising edge
1	X	0	1		Input capture source is TIOCA5 pin Input capture at falling edge
1	X	1	X		Input capture source is TIOCA5 pin Input capture at both edges

[Legend]

X: Don't care

10.3.4 Timer Interrupt Enable Register (TIER)

TIER controls enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel.

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

Bit	Bit Name	Initial value	R/W	Description
7	TTGE	0	R/W	A/D Conversion Start Request Enable Enables/disables generation of A/D conversion start requests by TGRA input capture/compare match. 0: A/D conversion start request generation disabled 1: A/D conversion start request generation enabled
6	—	1	R	Reserved This is a read-only bit and cannot be modified.
5	TCIEU	0	R/W	Underflow Interrupt Enable Enables/disables interrupt requests (TCIU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1, 2, 4, and 5. In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified. 0: Interrupt requests (TCIU) by TCFU disabled 1: Interrupt requests (TCIU) by TCFU enabled
4	TCIEV	0	R/W	Overflow Interrupt Enable Enables/disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1. 0: Interrupt requests (TCIV) by TCFV disabled 1: Interrupt requests (TCIV) by TCFV enabled

Bit	Bit Name	Initial value	R/W	Description
3	TGIED	0	R/W	TGR Interrupt Enable D Enables/disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3. In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified. 0: Interrupt requests (TGID) by TGFD bit disabled 1: Interrupt requests (TGID) by TGFD bit enabled
2	TGIEC	0	R/W	TGR Interrupt Enable C Enables/disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3. In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified. 0: Interrupt requests (TGIC) by TGFC bit disabled 1: Interrupt requests (TGIC) by TGFC bit enabled
1	TGIEB	0	R/W	TGR Interrupt Enable B Enables/disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1. 0: Interrupt requests (TGIB) by TGFB bit disabled 1: Interrupt requests (TGIB) by TGFB bit enabled
0	TGIEA	0	R/W	TGR Interrupt Enable A Enables/disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1. 0: Interrupt requests (TGIA) by TGFA bit disabled 1: Interrupt requests (TGIA) by TGFA bit enabled

10.3.5 Timer Status Register (TSR)

TSR indicates the status of each channel. The TPU has six TSR registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TCFD	—	TCFU	TCFV	TGFD	TGFC	TGFB	TGFA
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)*

Note: * Only 0 can be written to bits 5 to 0, to clear flags.

Bit	Bit Name	Initial value	R/W	Description
7	TCFD	1	R	Count Direction Flag Status flag that shows the direction in which TCNT counts in channels 1, 2, 4, and 5. In channels 0 and 3, bit 7 is reserved. It is always read as 1 and cannot be modified. 0: TCNT counts down 1: TCNT counts up
6	—	1	R	Reserved This is a read-only bit and cannot be modified.
5	TCFU	0	R/(W)*	Underflow Flag Status flag that indicates that a TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting mode. In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified. [Setting condition] • When the TCNT value underflows (changes from H'0000 to H'FFFF) [Clearing condition] • When a 0 is written to TCFU after reading TCFU = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
4	TCFV	0	R/(W)*	Overflow Flag Status flag that indicates that a TCNT overflow has occurred. [Setting condition] • When the TCNT value overflows (changes from H'FFFF to H'0000) [Clearing condition] • When a 0 is written to TCFV after reading TCFV = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	TGFD	0	R/(W)*	Input Capture/Output Compare Flag D Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3. In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified. [Setting conditions] • When TCNT = TGRD while TGRD is functioning as output compare register • When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register [Clearing conditions] • When DTC is activated by a TGID interrupt while the DISEL bit in MRB of DTC is 0 • When 0 is written to TGFD after reading TGFD = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
2	TGFC	0	R/(W)*	Input Capture/Output Compare Flag C Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3. In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified. [Setting conditions] • When TCNT = TGRC while TGRC is functioning as output compare register • When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register [Clearing conditions] • When DTC is activated by a TGIC interrupt while the DISEL bit in MRB of DTC is 0 • When 0 is written to TGFC after reading TGFC = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	TGFB	0	R/(W)*	Input Capture/Output Compare Flag B Status flag that indicates the occurrence of TGRB input capture or compare match. [Setting conditions] • When TCNT = TGRB while TGRB is functioning as output compare register • When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register [Clearing conditions] • When DTC is activated by a TGIB interrupt while the DISEL bit in MRB of DTC is 0 • When 0 is written to TGFB after reading TGFB = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
0	TGFA	0	R/(W)*	Input Capture/Output Compare Flag A Status flag that indicates the occurrence of TGRA input capture or compare match. [Setting conditions] • When TCNT = TGRA while TGRA is functioning as output compare register • When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register [Clearing conditions] • When DTC is activated by a TGIA interrupt while the DISEL bit in MRB of DTC is 0 • When DMAC is activated by a TGIA interrupt while the DTA bit in DMDR of DMAC is 1 • When 0 is written to TGFA after reading TGFA = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

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

10.3.6 Timer Counter (TCNT)

TCNT is a 16-bit readable/writable counter. The TPU has six TCNT counters, one for each channel.

TCNT is initialized to H'0000 by a reset or in hardware standby mode.

TCNT cannot be accessed in 8-bit units. TCNT must always be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8
Bit Name								
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
Bit Name								
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

10.3.7 Timer General Register (TGR)

TGR is a 16-bit readable/writable register with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channels 0 and 3 and two each for channels 1, 2, 4, and 5. TGRC and TGRD for channels 0 and 3 can also be designated for operation as buffer registers. The TGR registers cannot be accessed in 8-bit units; they must always be accessed in 16-bit units. TGR and buffer register combinations during buffer operations are TGRA–TGRC and TGRB–TGRD.

Bit	15	14	13	12	11	10	9	8
Bit Name								
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
Bit Name								
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

10.3.8 Timer Start Register (TSTR)

TSTR starts or stops operation for channels 0 to 5. When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	CST5	CST4	CST3	CST2	CST1	CST0
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	Bit Name	Initial value	R/W	Description
7, 6	——	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	CST5	0	R/W	Counter Start 5 to 0 These bits select operation or stoppage for TCNT. If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value. 0: TCNT_5 to TCNT_0 count operation is stopped 1: TCNT_5 to TCNT_0 performs count operation
4	CST4	0	R/W	
3	CST3	0	R/W	
2	CST2	0	R/W	
1	CST1	0	R/W	
0	CST0	0	R/W	

10.3.9 Timer Synchronous Register (TSYR)

TSYR selects independent operation or synchronous operation for the TCNT counters of channels 0 to 5. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
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	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	SYNC5	0	R/W	Timer Synchronization 5 to 0 These bits select whether operation is independent of or synchronized with other channels. When synchronous operation is selected, synchronous presetting of multiple channels, and synchronous clearing through counter clearing on another channel are possible. To set synchronous operation, the SYNC bits for at least two channels must be set to 1. To set synchronous clearing, in addition to the SYNC bit, the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR. 0: TCNT_5 to TCNT_0 operate independently (TCNT presetting/clearing is unrelated to other channels) 1: TCNT_5 to TCNT_0 perform synchronous operation (TCNT synchronous presetting/synchronous clearing is possible)
4	SYNC4	0	R/W	
3	SYNC3	0	R/W	
2	SYNC2	0	R/W	
1	SYNC1	0	R/W	
0	SYNC0	0	R/W	

10.4 Operation

10.4.1 Basic Functions

Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, periodic counting, and external event counting.

Each TGR can be used as an input capture register or output compare register.

(1) Counter Operation

When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

(a) Example of count operation setting procedure

Figure 10.2 shows an example of the count operation setting procedure.

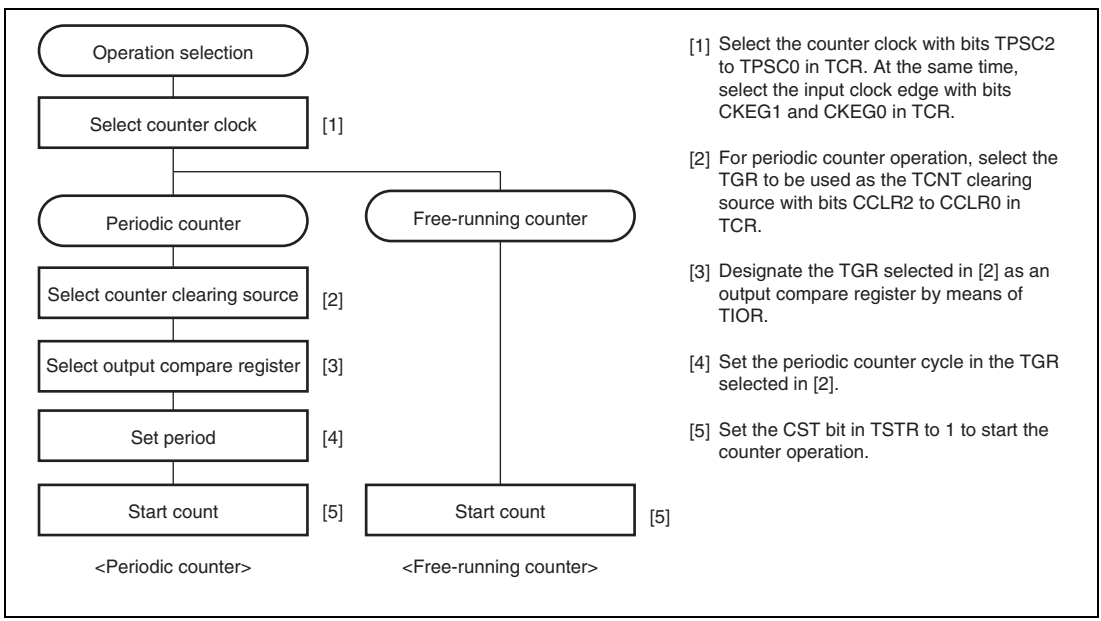


Figure 10.2 Example of Counter Operation Setting Procedure

(b) Free-running count operation and periodic count operation

Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (changes from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000.

Figure 10.3 illustrates free-running counter operation.

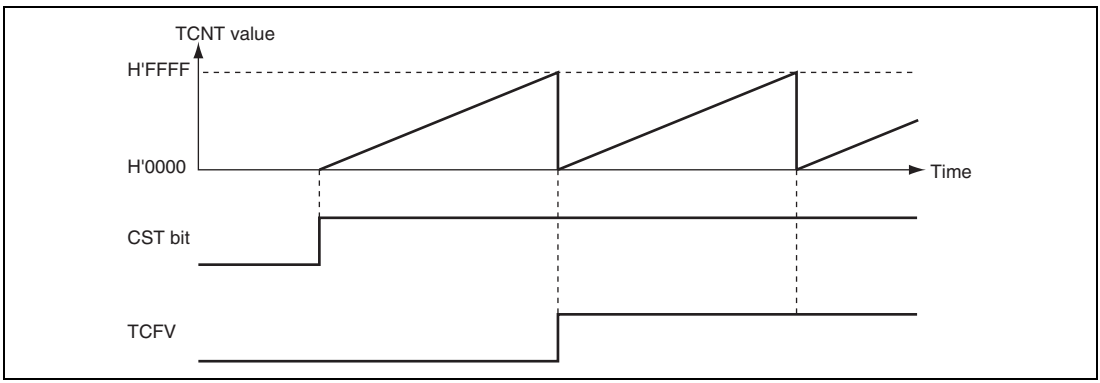


Figure 10.3 Free-Running Counter Operation

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts count-up operation as a periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000.

Figure 10.4 illustrates periodic counter operation.

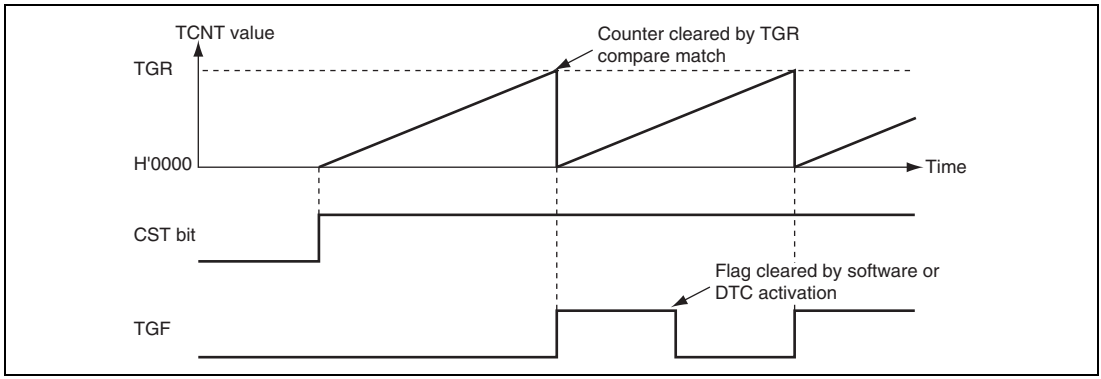


Figure 10.4 Periodic Counter Operation

(2) Waveform Output by Compare Match

The TPU can perform 0, 1, or toggle output from the corresponding output pin using a compare match.

(a) Example of setting procedure for waveform output by compare match

Figure 10.5 shows an example of the setting procedure for waveform output by a compare match.

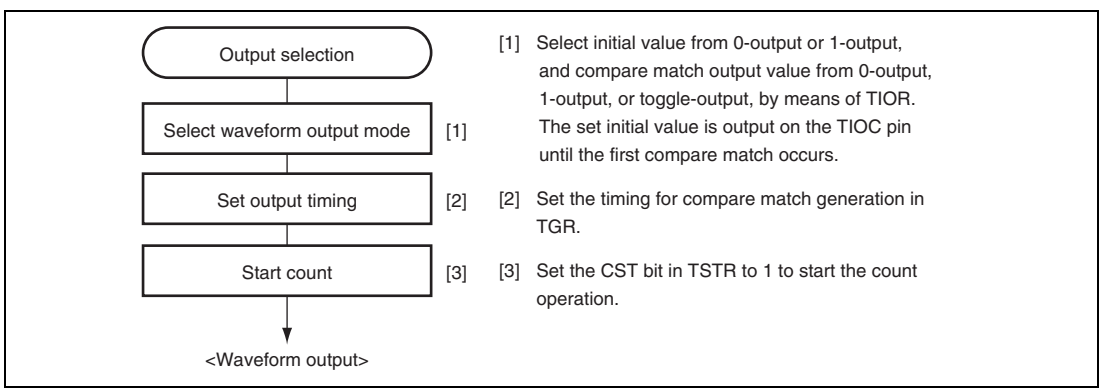


Figure 10.5 Example of Setting Procedure for Waveform Output by Compare Match

(b) Examples of waveform output operation

Figure 10.6 shows an example of 0-output and 1-output.

In this example, TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level match, the pin level does not change.

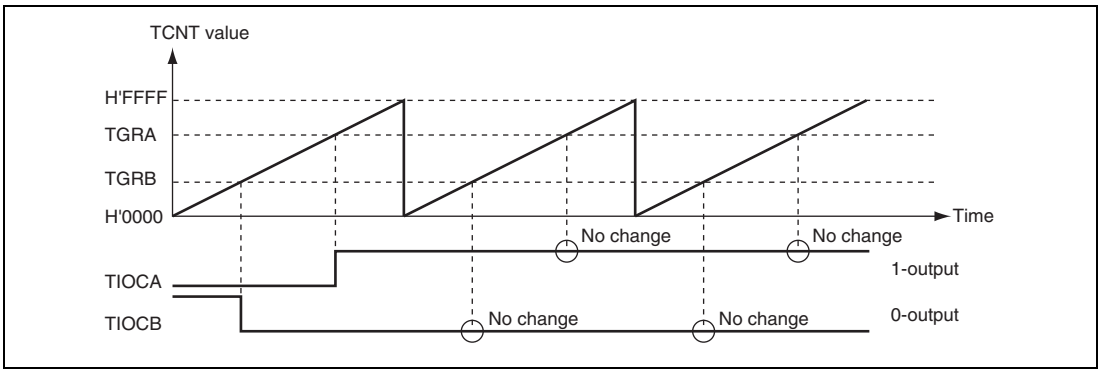


Figure 10.6 Example of 0-Output/1-Output Operation

Figure 10.7 shows an example of toggle output.

In this example, TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.

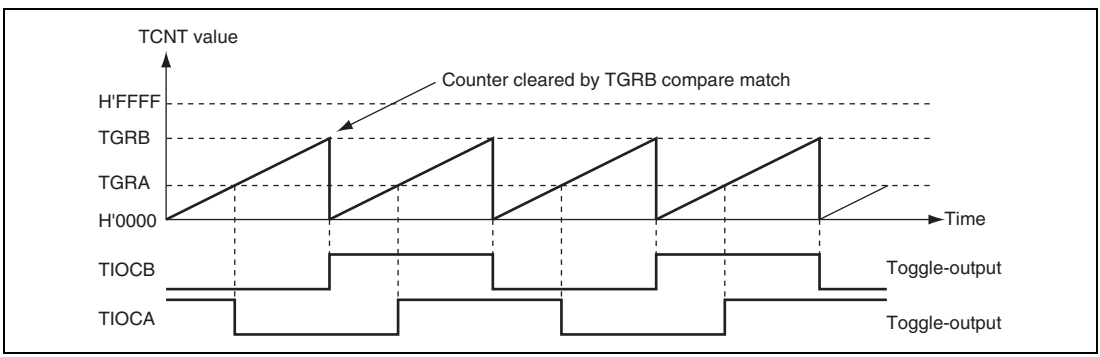


Figure 10.7 Example of Toggle Output Operation

(3) Input Capture Function

The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detection edge. For channels 0, 1, 3, and 4, it is also possible to specify another channel's counter input clock or compare match signal as the input capture source.

Note: When another channel's counter input clock is used as the input capture input for channels 0 and 3, Pϕ/1 should not be selected as the counter input clock used for input capture input. Input capture will not be generated if Pϕ/1 is selected.

(a) Example of setting procedure for input capture operation

Figure 10.8 shows an example of the setting procedure for input capture operation.

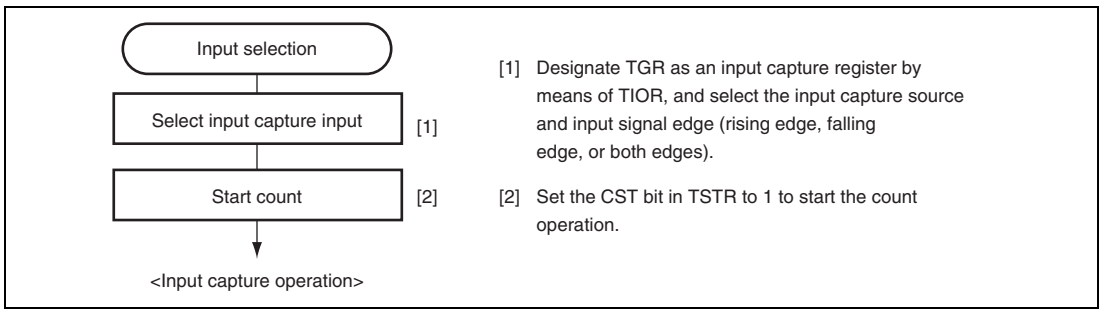


Figure 10.8 Example of Setting Procedure for Input Capture Operation

(b) Example of input capture operation

Figure 10.9 shows an example of input capture operation.

In this example, both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.

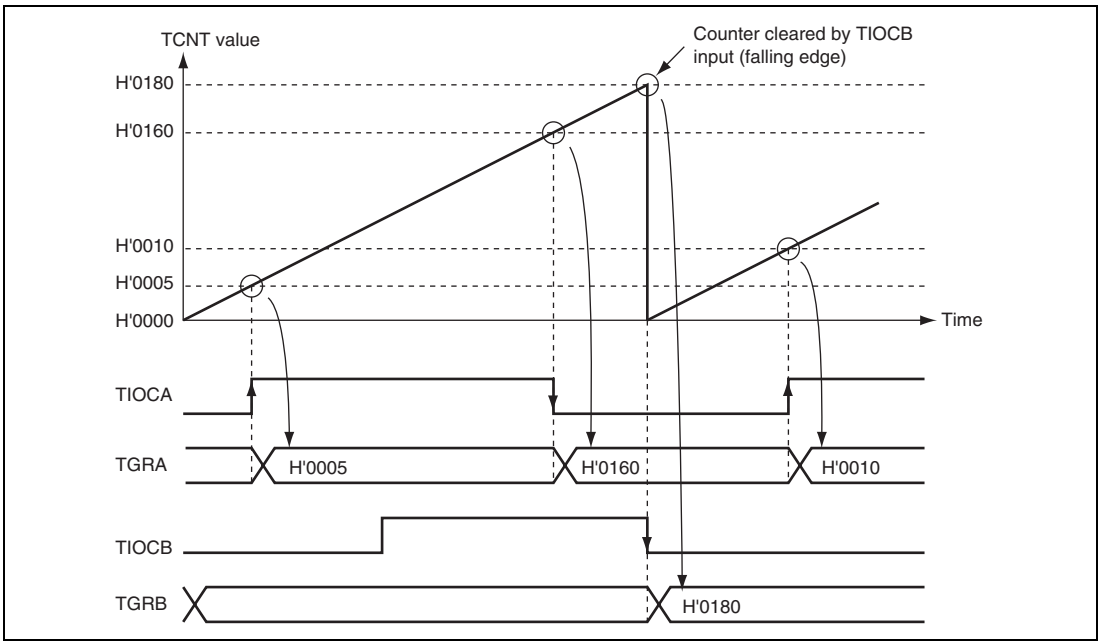


Figure 10.9 Example of Input Capture Operation

10.4.2 Synchronous Operation

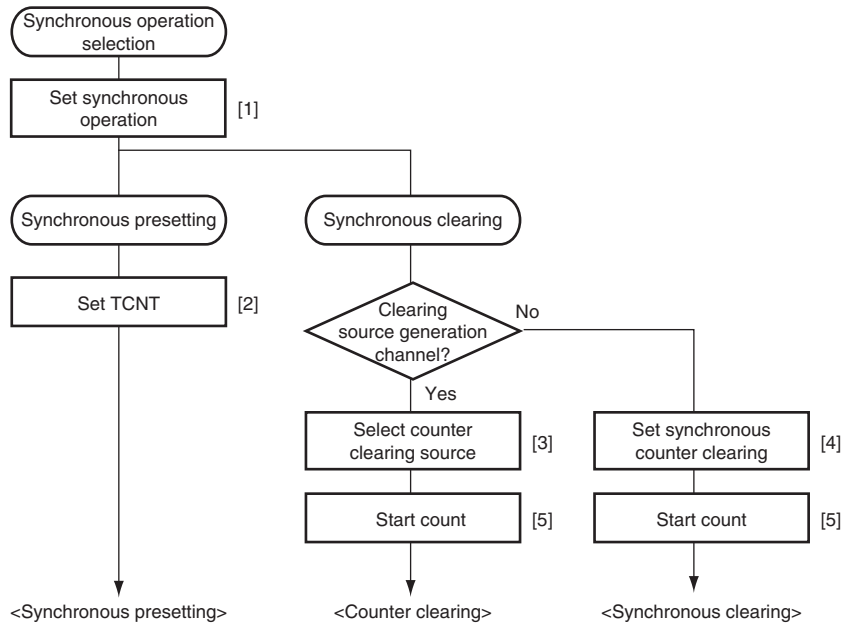
In synchronous operation, the values in multiple TCNT counters can be rewritten simultaneously (synchronous presetting). Also, multiple TCNT counters can be cleared simultaneously (synchronous clearing) by making the appropriate setting in TCR.

Synchronous operation enables TGR to be incremented with respect to a single time base.

Channels 0 to 5 can all be designated for synchronous operation.

(1) Example of Synchronous Operation Setting Procedure

Figure 10.10 shows an example of the synchronous operation setting procedure.



[1] Set the SYNC bits in TSYR corresponding to the channels to be designated for synchronous operation to 1.

[2] When the TCNT counter of any of the channels designated for synchronous operation is written to, the same value is simultaneously written to the other TCNT counters.

[3] Use bits CCLR2 to CCLR0 in TCR to specify TCNT clearing by input capture/output compare, etc.

[4] Use bits CCLR2 to CCLR0 in TCR to designate synchronous clearing for the counter clearing source.

[5] Set the CST bits in TSTR for the relevant channels to 1, to start the count operation.

Figure 10.10 Example of Synchronous Operation Setting Procedure

(2) Example of Synchronous Operation

Figure 10.11 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGRB_0 compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source.

Three-phase PWM waveforms are output from pins TIOCA0, TIOCA1, and TIOCA2. At this time, synchronous presetting and synchronous clearing by TGRB_0 compare match are performed for channel 0 to 2 TCNT counters, and the data set in TGRB_0 is used as the PWM cycle.

For details on PWM modes, see section 10.4.5, PWM Modes.

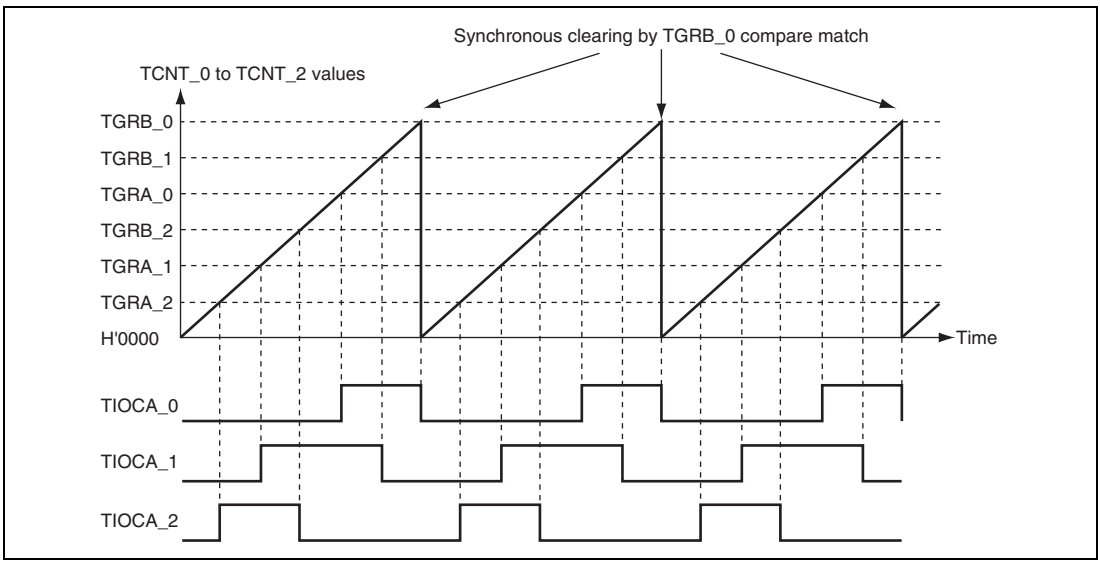


Figure 10.11 Example of Synchronous Operation

10.4.3 Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

Buffer operation differs depending on whether TGR has been designated as an input capture register or a compare match register.

Table 10.29 shows the register combinations used in buffer operation.

Table 10.29 Register Combinations in Buffer Operation

Channel	Timer General Register	Buffer Register
0	TGRA_0	TGRC_0
	TGRB_0	TGRD_0
3	TGRA_3	TGRC_3
	TGRB_3	TGRD_3

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.

  This operation is illustrated in figure 10.12.

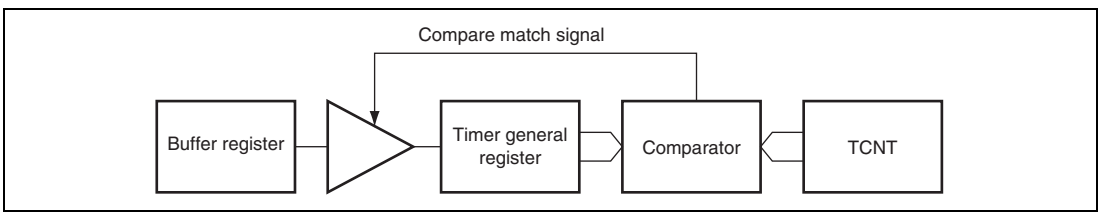


Figure 10.12 Compare Match Buffer Operation

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in TGR is transferred to the buffer register.

  This operation is illustrated in figure 10.13.

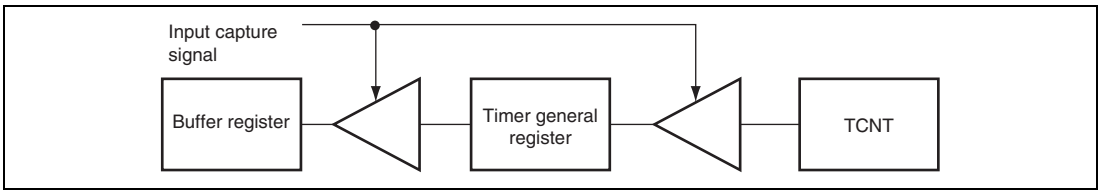


Figure 10.13 Input Capture Buffer Operation

(1) Example of Buffer Operation Setting Procedure

Figure 10.14 shows an example of the buffer operation setting procedure.

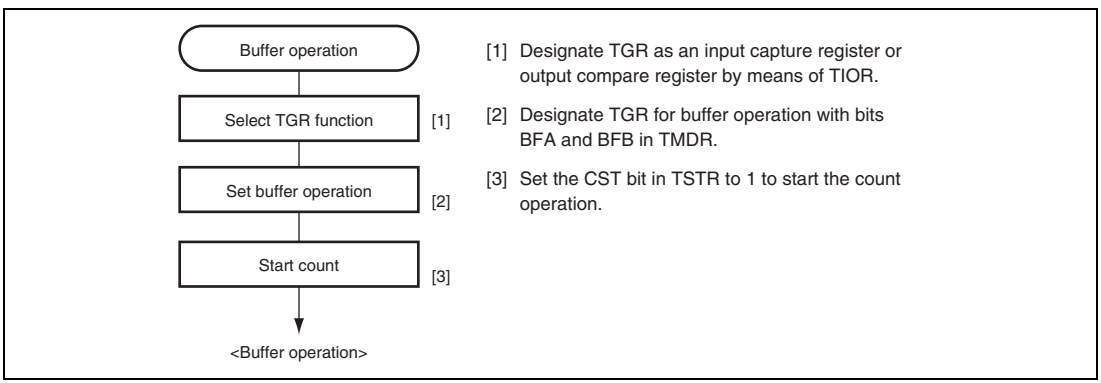


Figure 10.14 Example of Buffer Operation Setting Procedure

(2) Examples of Buffer Operation

(a) When TGR is an output compare register

Figure 10.15 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B.

As buffer operation has been set, when compare match A occurs, the output changes and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

For details on PWM modes, see section 10.4.5, PWM Modes.

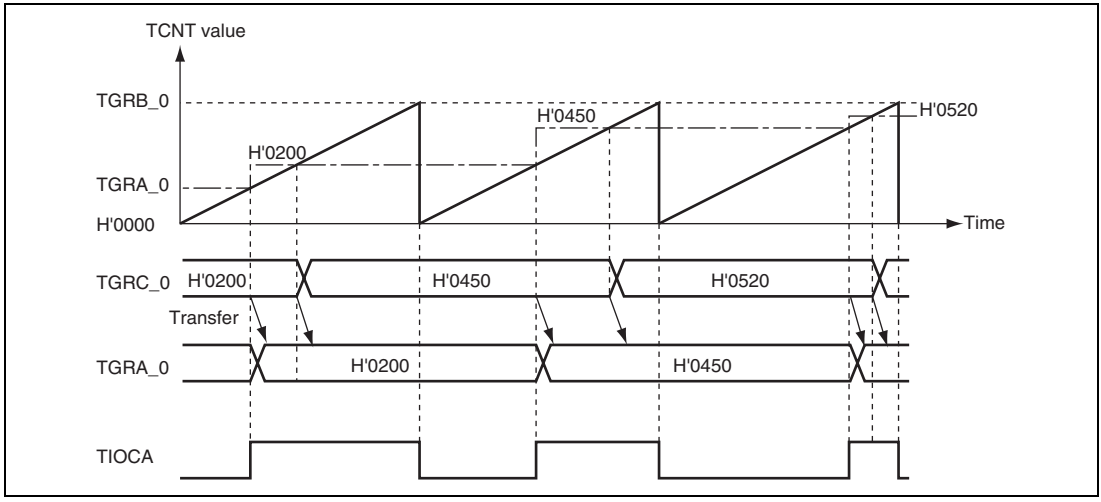


Figure 10.15 Example of Buffer Operation (1)

(b) When TGR is an input capture register

Figure 10.16 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC.

Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge.

As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.

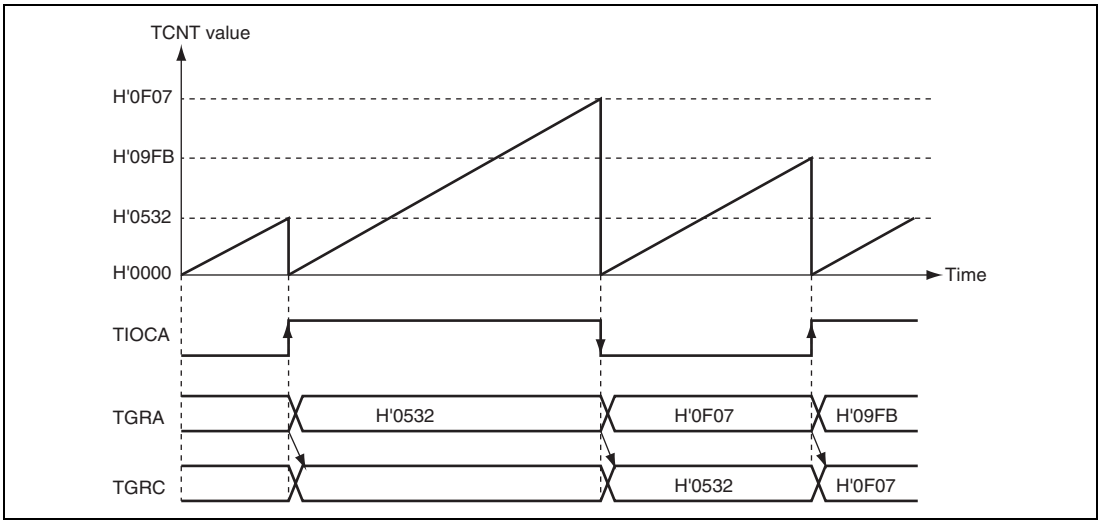


Figure 10.16 Example of Buffer Operation (2)

10.4.4 Cascaded Operation

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter.

This function works by counting the channel 1 (channel 4) counter clock at overflow/underflow of TCNT_2 (TCNT_5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

Table 10.30 shows the register combinations used in cascaded operation.

Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

Table 10.30 Cascaded Combinations

Combination	Upper 16 Bits	Lower 16 Bits
Channels 1 and 2	TCNT_1	TCNT_2
Channels 4 and 5	TCNT_4	TCNT_5

(1) Example of Cascaded Operation Setting Procedure

Figure 10.17 shows an example of the setting procedure for cascaded operation.

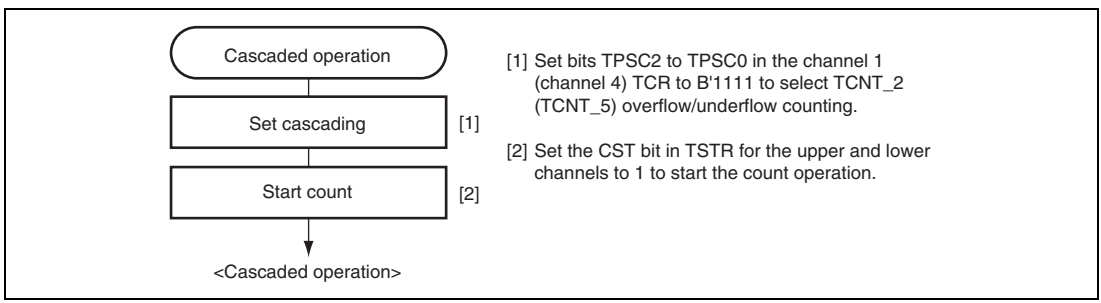


Figure 10.17 Example of Cascaded Operation Setting Procedure

(2) Examples of Cascaded Operation

Figure 10.18 illustrates the operation when counting upon TCNT_2 overflow/underflow has been set for TCNT_1, TGRA_1 and TGRA_2 have been designated as input capture registers, and the TIOC pin rising edge has been selected.

When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGRA_1, and the lower 16 bits to TGRA_2.

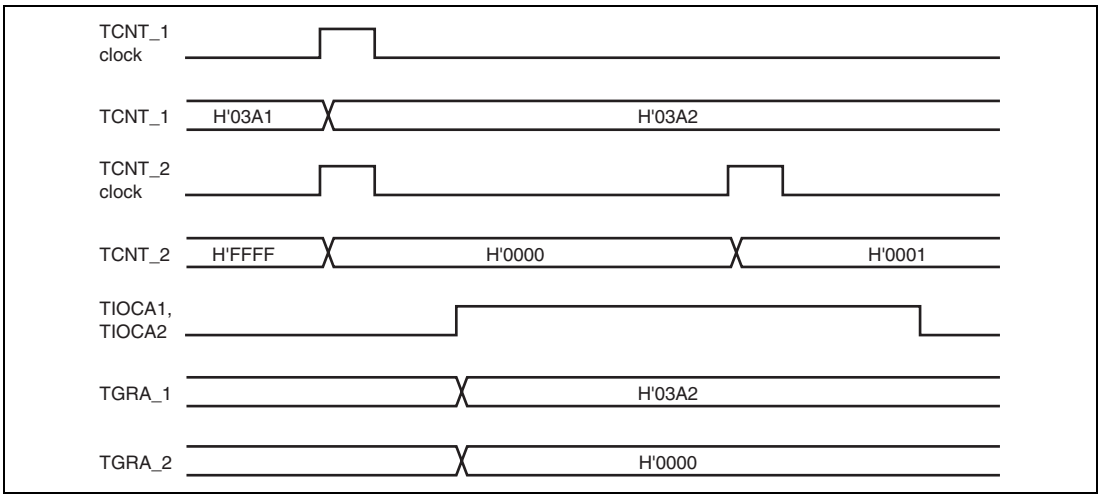


Figure 10.18 Example of Cascaded Operation (1)

Figure 10.19 illustrates the operation when counting upon TCNT_2 overflow/underflow has been set for TCNT_1, and phase counting mode has been designated for channel 2.

TCNT_1 is incremented by TCNT_2 overflow and decremented by TCNT_2 underflow.

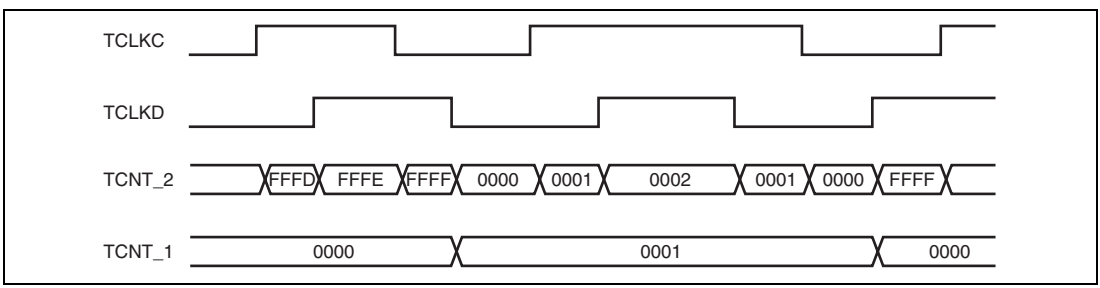


Figure 10.19 Example of Cascaded Operation (2)

10.4.5 PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0-, 1-, or toggle-output can be selected as the output level in response to compare match of each TGR.

Settings of TGR registers can output a PWM waveform in the range of 0% to 100% duty cycle.

Designating TGR compare match as the counter clearing source enables the cycle to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible.

There are two PWM modes, as described below.

1. PWM mode 1

   PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The outputs specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR are output from the TIOCA and TIOCC pins at compare matches A and C, respectively. The outputs specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR are output at compare matches B and D, respectively. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs.

   In PWM mode 1, a maximum 8-phase PWM output is possible.

2. PWM mode 2

   PWM output is generated using one TGR as the cycle register and the others as duty cycle registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronous register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty cycle registers are identical, the output value does not change when a compare match occurs.

   In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

The correspondence between PWM output pins and registers is shown in table 10.31.

Table 10.31 PWM Output Registers and Output Pins

		Output Pins	
Channel	**Registers**	**PWM Mode 1**	**PWM Mode 2**
0	TGRA_0	TIOCA0	TIOCA0
	TGRB_0		TIOCB0
	TGRC_0	TIOCC0	TIOCC0
	TGRD_0		TIOCD0
1	TGRA_1	TIOCA1	TIOCA1
	TGRB_1		TIOCB1
2	TGRA_2	TIOCA2	TIOCA2
	TGRB_2		TIOCB2
3	TGRA_3	TIOCA3	TIOCA3
	TGRB_3		TIOCB3
	TGRC_3	TIOCC3	TIOCC3
	TGRD_3		TIOCD3
4	TGRA_4	TIOCA4	TIOCA4
	TGRB_4		TIOCB4
5	TGRA_5	TIOCA5	TIOCA5
	TGRB_5		TIOCB5

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the cycle is set.

(1) Example of PWM Mode Setting Procedure

Figure 10.20 shows an example of the PWM mode setting procedure.

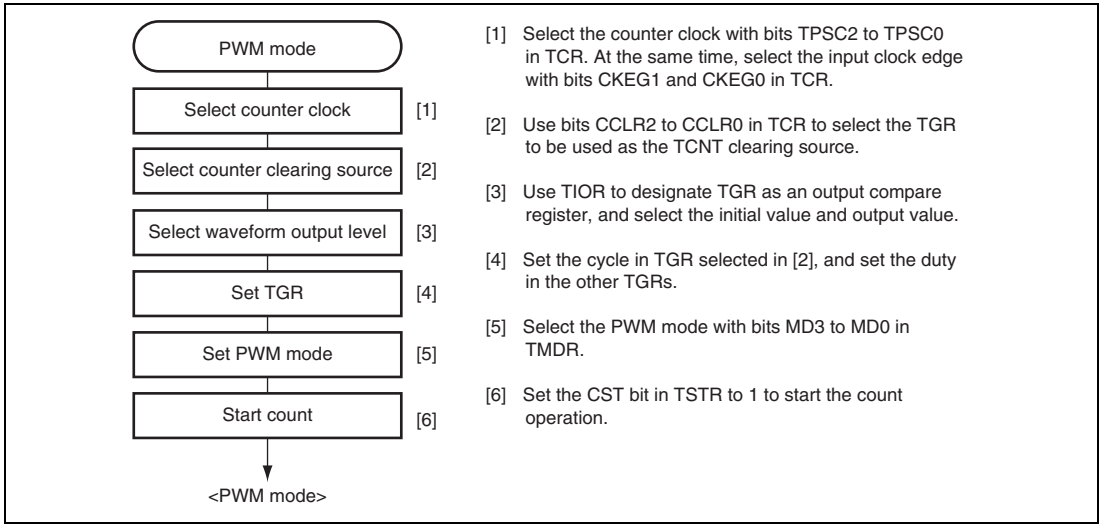


Figure 10.20 Example of PWM Mode Setting Procedure

(2) Examples of PWM Mode Operation

Figure 10.21 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value.

In this case, the value set in TGRA is used as the cycle, and the value set in TGRB register as the duty cycle.

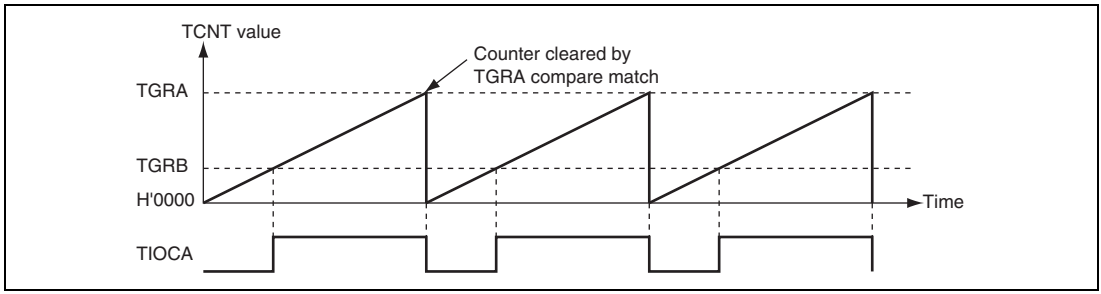


Figure 10.21 Example of PWM Mode Operation (1)

Figure 10.22 shows an example of PWM mode 2 operation.

In this example, synchronous operation is designated for channels 0 and 1, TGRB_1 compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGRA_0 to TGRD_0, TGRA_1), to output a 5-phase PWM waveform.

In this case, the value set in TGRB_1 is used as the cycle, and the values set in the other TGRs as the duty cycle.

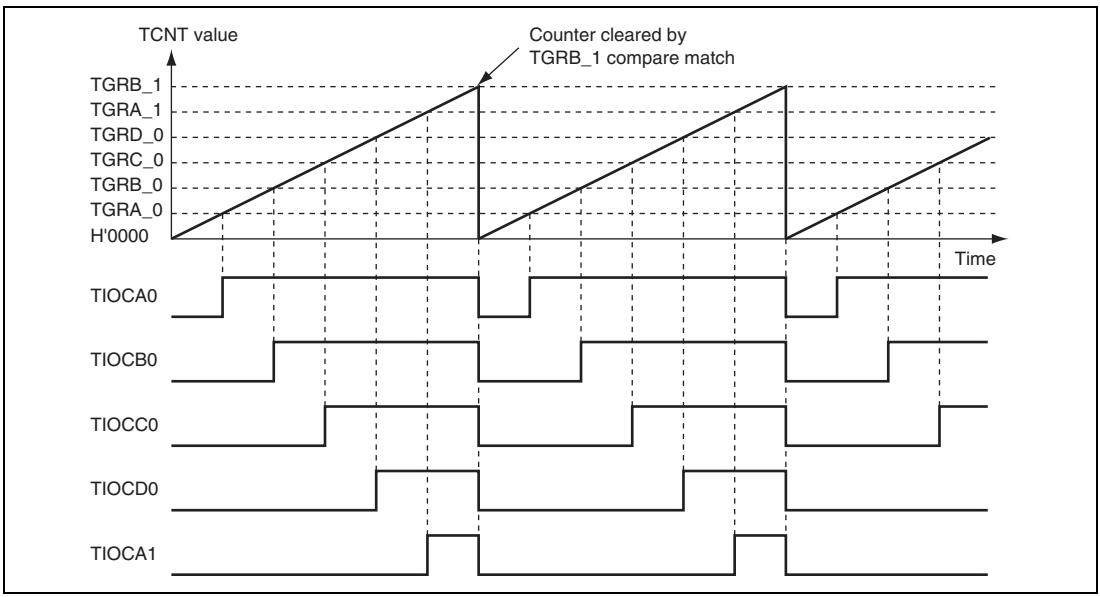


Figure 10.22 Example of PWM Mode Operation (2)

Figure 10.23 shows examples of PWM waveform output with 0% duty cycle and 100% duty cycle in PWM mode.

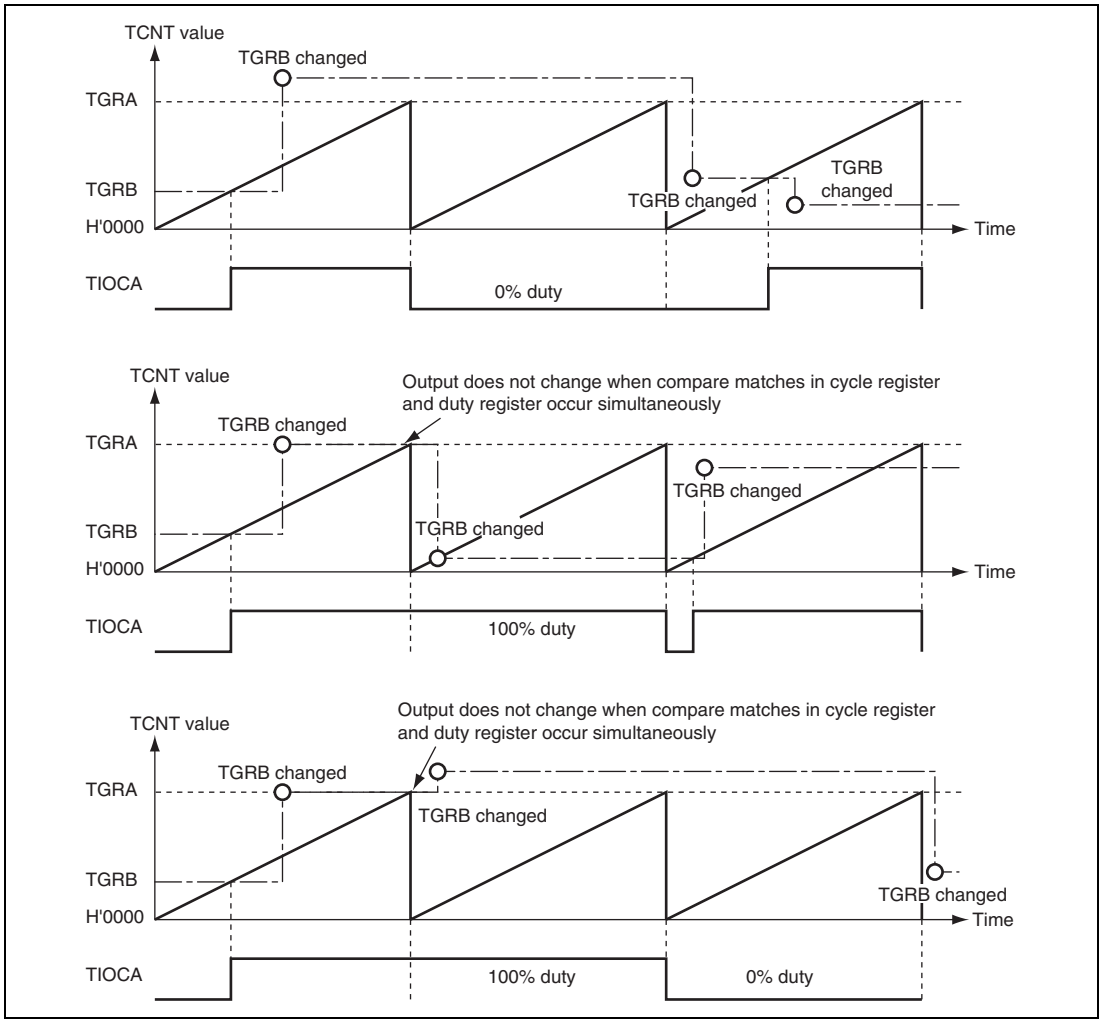


Figure 10.23 Example of PWM Mode Operation (3)

10.4.6 Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1, 2, 4, and 5.

When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used.

This can be used for two-phase encoder pulse input.

When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set.

The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 10.32 shows the correspondence between external clock pins and channels.

Table 10.32 Clock Input Pins in Phase Counting Mode

	External Clock Pins	
Channels	**A-Phase**	**B-Phase**
When channel 1 or 5 is set to phase counting mode	TCLKA	TCLKB
When channel 2 or 4 is set to phase counting mode	TCLKC	TCLKD

(1) Example of Phase Counting Mode Setting Procedure

Figure 10.24 shows an example of the phase counting mode setting procedure.

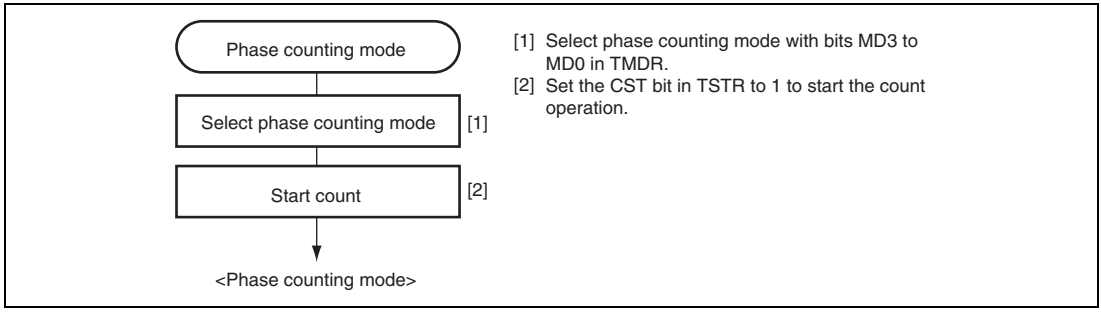


Figure 10.24 Example of Phase Counting Mode Setting Procedure

(2) Examples of Phase Counting Mode Operation

In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

(a) Phase counting mode 1

Figure 10.25 shows an example of phase counting mode 1 operation, and table 10.33 summarizes the TCNT up/down-count conditions.

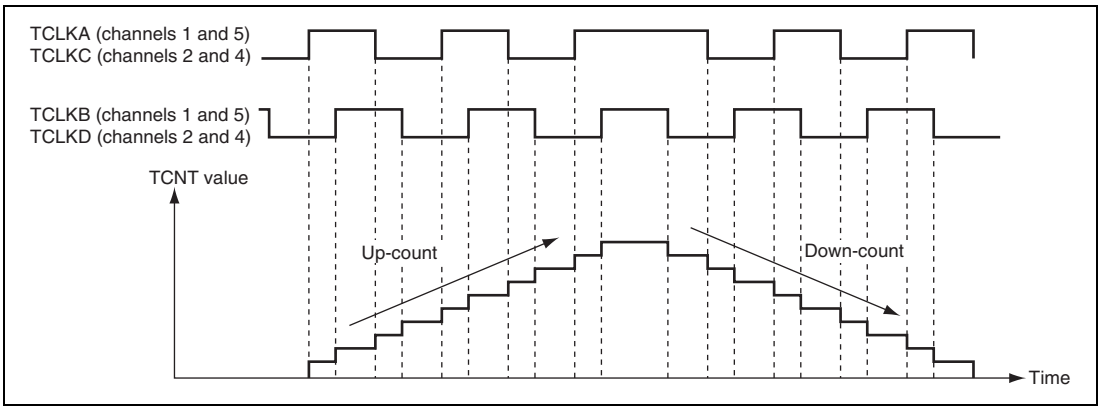


Figure 10.25 Example of Phase Counting Mode 1 Operation

Table 10.33 Up/Down-Count Conditions in Phase Counting Mode 1

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Up-count
Low level	↓	
↑	Low level	
↓	High level	
High level	↓	Down-count
Low level	↑	
↑	High level	
↓	Low level	

[Legend]
↑: Rising edge
↓: Falling edge

(b) Phase counting mode 2

Figure 10.26 shows an example of phase counting mode 2 operation, and table 10.34 summarizes the TCNT up/down-count conditions.

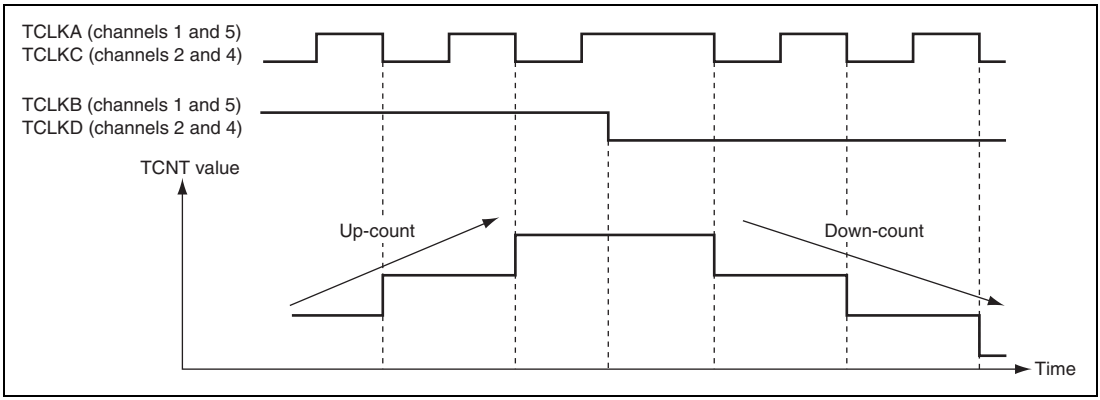


Figure 10.26 Example of Phase Counting Mode 2 Operation

Table 10.34 Up/Down-Count Conditions in Phase Counting Mode 2

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	⌐↑	Don't care
Low level	⌐↓	Don't care
⌐↑	Low level	Don't care
⌐↓	High level	Up-count
High level	⌐↓	Don't care
Low level	⌐↑	Don't care
⌐↑	High level	Don't care
⌐↓	Low level	Down-count

[Legend]
⌐↑ : Rising edge
⌐↓ : Falling edge

(c) Phase counting mode 3

Figure 10.27 shows an example of phase counting mode 3 operation, and table 10.35 summarizes the TCNT up/down-count conditions.

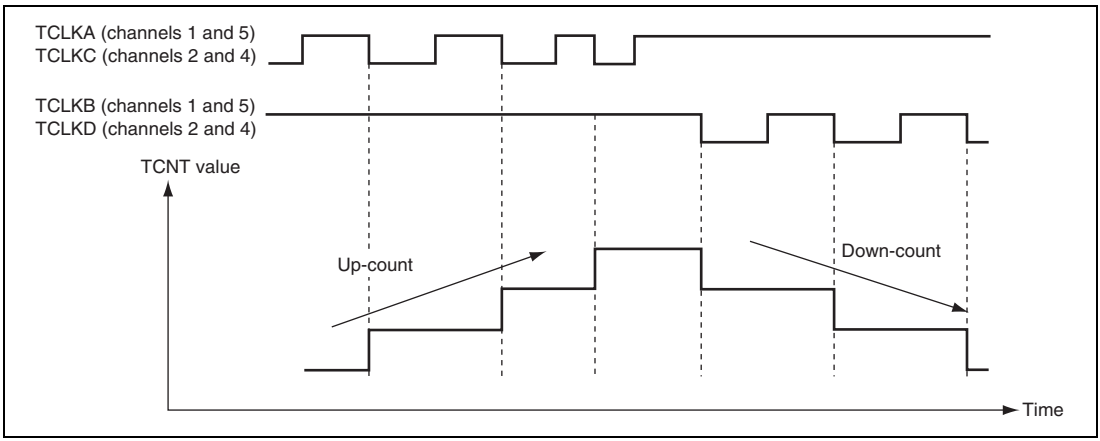


Figure 10.27 Example of Phase Counting Mode 3 Operation

Table 10.35 Up/Down-Count Conditions in Phase Counting Mode 3

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Don't care
Low level	↓	Don't care
↑	Low level	Don't care
↓	High level	Up-count
High level	↓	Down-count
Low level	↑	Don't care
↑	High level	Don't care
↓	Low level	Don't care

[Legend]
↑ : Rising edge
↓ : Falling edge

(d) Phase counting mode 4

Figure 10.28 shows an example of phase counting mode 4 operation, and table 10.36 summarizes the TCNT up/down-count conditions.

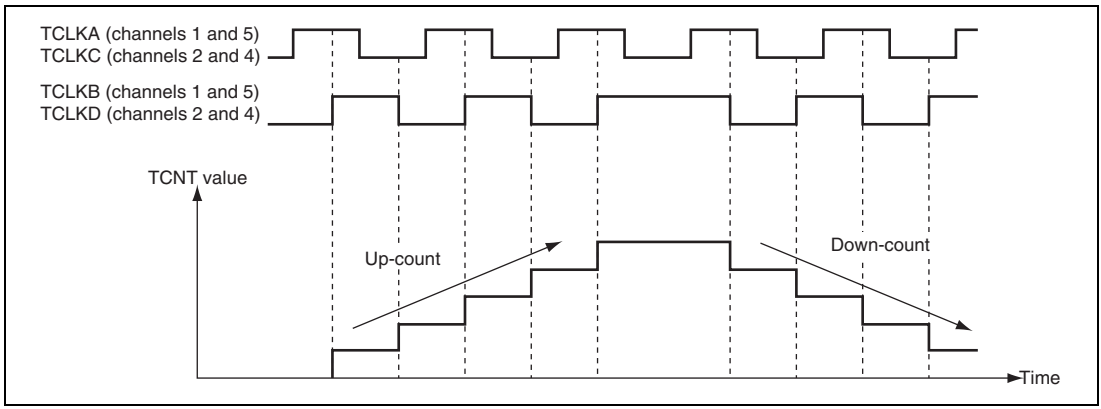


Figure 10.28 Example of Phase Counting Mode 4 Operation

Table 10.36 Up/Down-Count Conditions in Phase Counting Mode 4

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Up-count
Low level	↓	
↑	Low level	Don't care
↓	High level	
High level	↓	Down-count
Low level	↑	
↑	High level	Don't care
↓	Low level	

[Legend]
↑ : Rising edge
↓ : Falling edge

(3) Phase Counting Mode Application Example

Figure 10.29 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

Channel 1 is set to phase counting mode 1, and the encoder pulse A-phase and B-phase are input to TCLKA and TCLKB.

Channel 0 operates with TCNT counter clearing by TGRC_0 compare match; TGRA_0 and TGRC_0 are used for the compare match function and are set with the speed control cycle and position control cycle. TGRB_0 is used for input capture, with TGRB_0 and TGRD_0 operating in buffer mode. The channel 1 counter input clock is designated as the TGRB_0 input capture source, and the pulse width of 2-phase encoder 4-multiplication pulses is detected.

TGRA_1 and TGRB_1 for channel 1 are designated for input capture, channel 0 TGRA_0 and TGRC_0 compare matches are selected as the input capture source, and the up/down-counter values for the control cycles are stored.

This procedure enables accurate position/speed detection to be achieved.

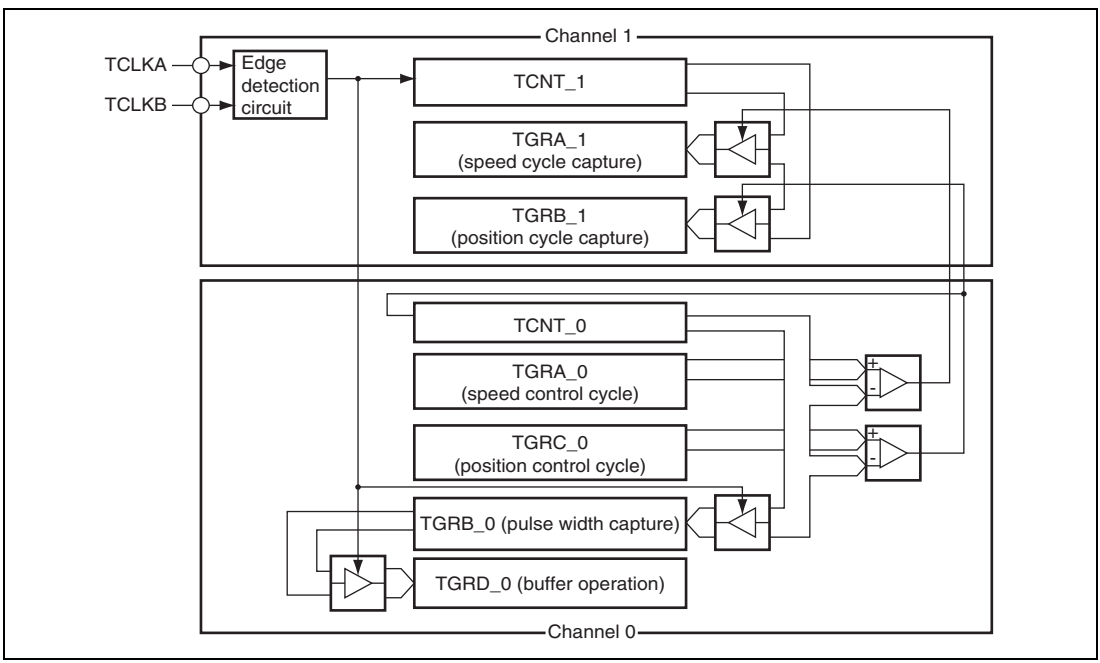


Figure 10.29 Phase Counting Mode Application Example

10.5 Interrupt Sources

There are three kinds of TPU interrupt sources: TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disable bit, allowing generation of interrupt request signals to be enabled or disabled individually.

When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0.

Relative channel priority levels can be changed by the interrupt controller, but the priority within a channel is fixed. For details, see section 5, Interrupt Controller.

Table 10.37 lists the TPU interrupt sources.

Table 10.37 TPU Interrupts

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation
0	TGI0A	TGRA_0 input capture/compare match	TGFA_0	O	O
	TGI0B	TGRB_0 input capture/compare match	TGFB_0	O	—
	TGI0C	TGRC_0 input capture/compare match	TGFC_0	O	—
	TGI0D	TGRD_0 input capture/compare match	TGFD_0	O	—
	TCI0V	TCNT_0 overflow	TCFV_0	—	—
1	TGI1A	TGRA_1 input capture/compare match	TGFA_1	O	O
	TGI1B	TGRB_1 input capture/compare match	TGFB_1	O	—
	TCI1V	TCNT_1 overflow	TCFV_1	—	—
	TCI1U	TCNT_1 underflow	TCFU_1	—	—
2	TGI2A	TGRA_2 input capture/compare match	TGFA_2	O	O
	TGI2B	TGRB_2 input capture/compare match	TGFB_2	O	—
	TCI2V	TCNT_2 overflow	TCFV_2	—	—
	TCI2U	TCNT_2 underflow	TCFU_2	—	—
3	TGI3A	TGRA_3 input capture/compare match	TGFA_3	O	O
	TGI3B	TGRB_3 input capture/compare match	TGFB_3	O	—
	TGI3C	TGRC_3 input capture/compare match	TGFC_3	O	—
	TGI3D	TGRD_3 input capture/compare match	TGFD_3	O	—
	TCI3V	TCNT_3 overflow	TCFV_3	—	—

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation
4	TGI4A	TGRA_4 input capture/compare match	TGFA_4	O	O
	TGI4B	TGRB_4 input capture/compare match	TGFB_4	O	—
	TCI4V	TCNT_4 overflow	TCFV_4	—	—
	TCI4U	TCNT_4 underflow	TCFU_4	—	—
5	TGI5A	TGRA_5 input capture/compare match	TGFA_5	O	O
	TGI5B	TGRB_5 input capture/compare match	TGFB_5	O	—
	TCI5V	TCNT_5 overflow	TCFV_5	—	—
	TCI5U	TCNT_5 underflow	TCFU_5	—	—

[Legend]

O : Possible

— : Not possible

Note: This table shows the initial state immediately after a reset. The relative channel priority levels can be changed by the interrupt controller.

(1) Input Capture/Compare Match Interrupt

An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1 by the occurrence of a TGR input capture/compare match on a channel. The interrupt request is cleared by clearing the TGF flag to 0. The TPU has 16 input capture/compare match interrupts, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

(2) Overflow Interrupt

An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to 1 by the occurrence of a TCNT overflow on a channel. The interrupt request is cleared by clearing the TCFV flag to 0. The TPU has six overflow interrupts, one for each channel.

(3) Underflow Interrupt

An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of a TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has four underflow interrupts, one each for channels 1, 2, 4, and 5.

10.6 DTC Activation

The DTC can be activated by the TGR input capture/compare match interrupt for a channel. For details, see section 8, Data Transfer Controller (DTC).

A total of 16 TPU input capture/compare match interrupts can be used as DTC activation sources, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

10.7 DMAC Activation

The DMAC can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 7, DMA Controller (DMAC).

A total of six TPU input capture/compare match interrupts can be used as DMAC activation sources, one for each channel.

10.8 A/D Converter Activation

The TGRA input capture/compare match for each channel can activate the A/D converter.

If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

10.9 Operation Timing

10.9.1 Input/Output Timing

(1) TCNT Count Timing

Figure 10.30 shows TCNT count timing in internal clock operation, and figure 10.31 shows TCNT count timing in external clock operation.

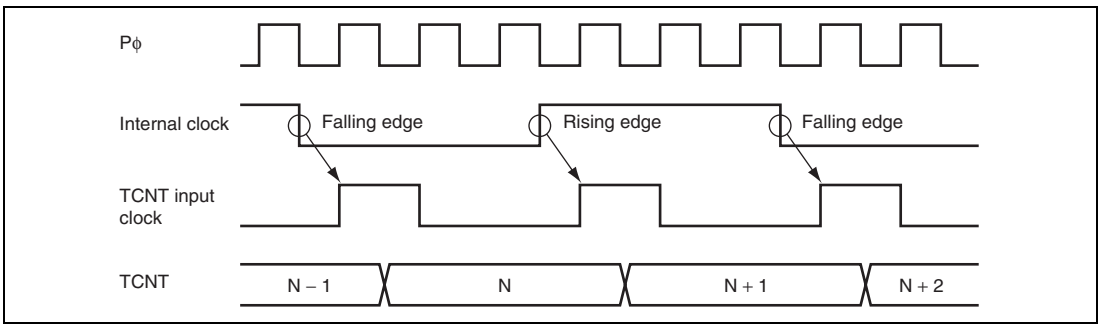


Figure 10.30 Count Timing in Internal Clock Operation

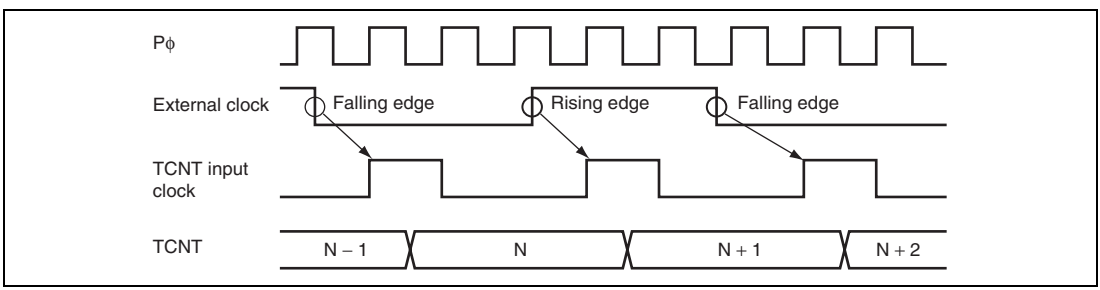


Figure 10.31 Count Timing in External Clock Operation

(2) Output Compare Output Timing

A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin (TIOC pin). After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

Figure 10.32 shows output compare output timing.

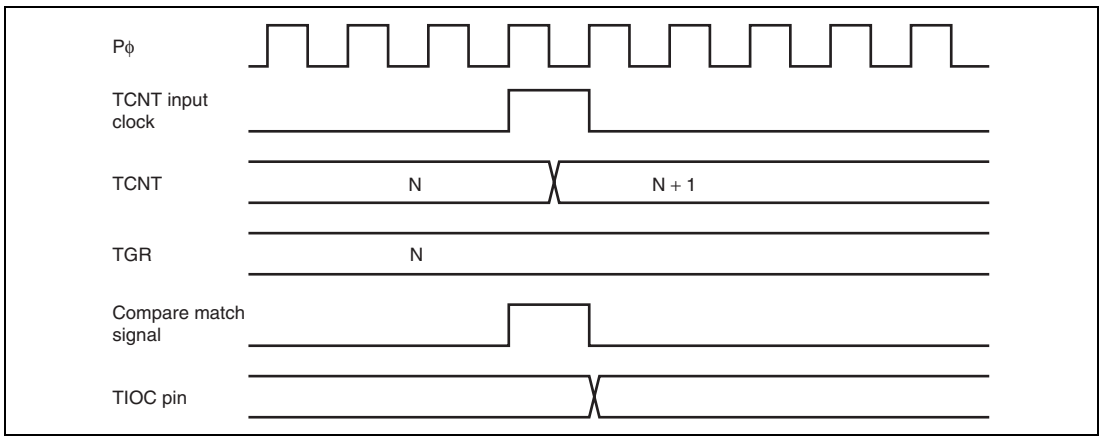


Figure 10.32 Output Compare Output Timing

Input Capture Signal Timing: Figure 10.33 shows input capture signal timing.

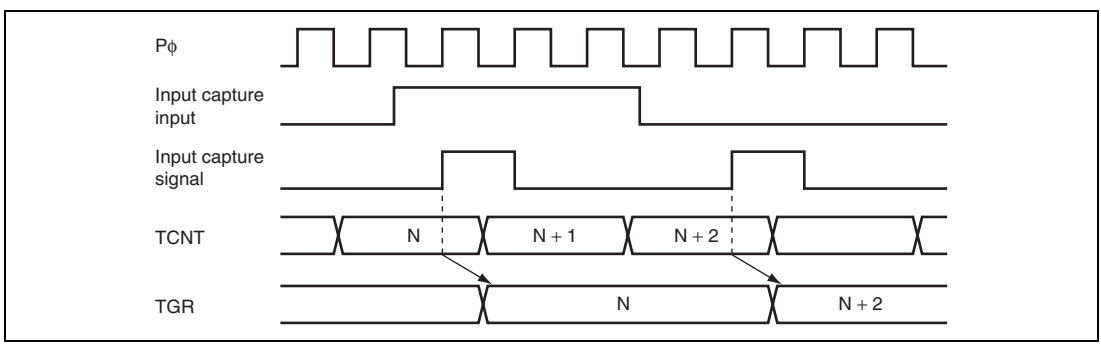


Figure 10.33 Input Capture Input Signal Timing

(3) Timing for Counter Clearing by Compare Match/Input Capture

Figure 10.34 shows the timing when counter clearing by compare match occurrence is specified, and figure 10.35 shows the timing when counter clearing by input capture occurrence is specified.

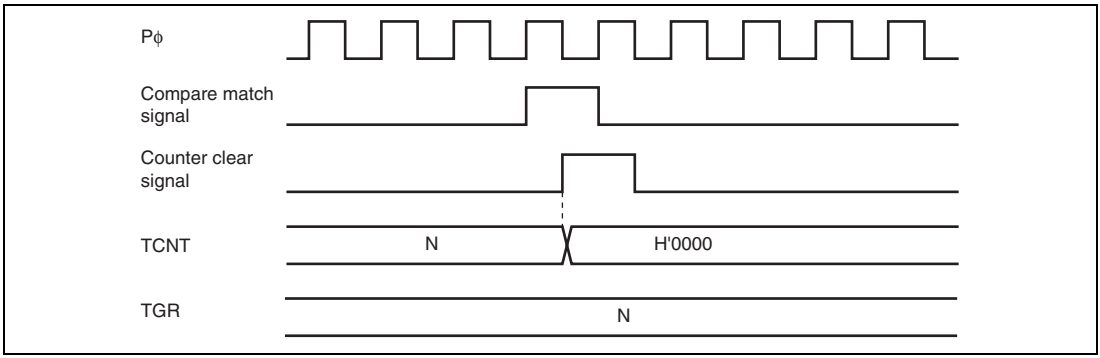


Figure 10.34 Counter Clear Timing (Compare Match)

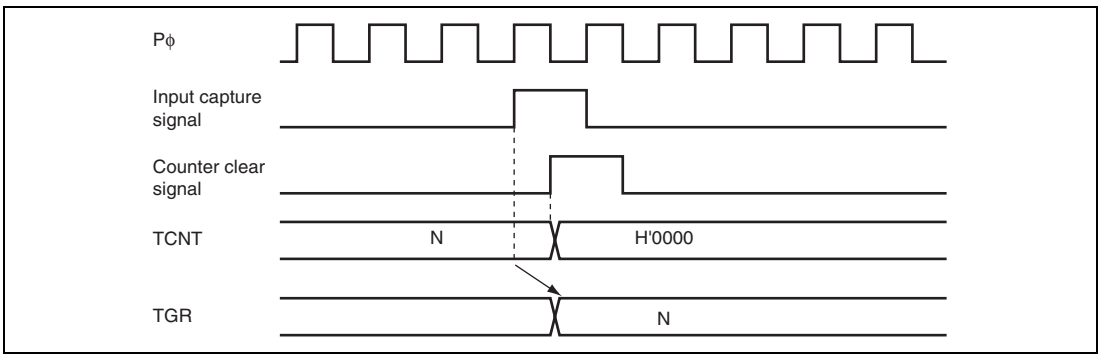


Figure 10.35 Counter Clear Timing (Input Capture)

(4) Buffer Operation Timing

Figures 10.36 and 10.37 show the timings in buffer operation.

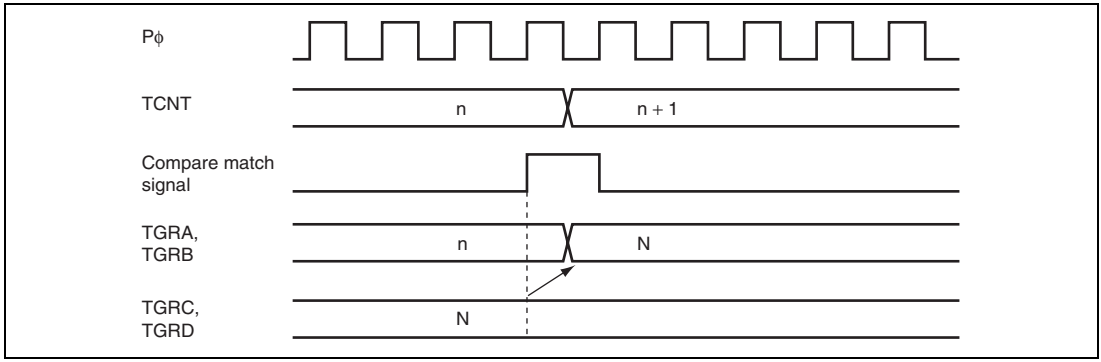


Figure 10.36 Buffer Operation Timing (Compare Match)

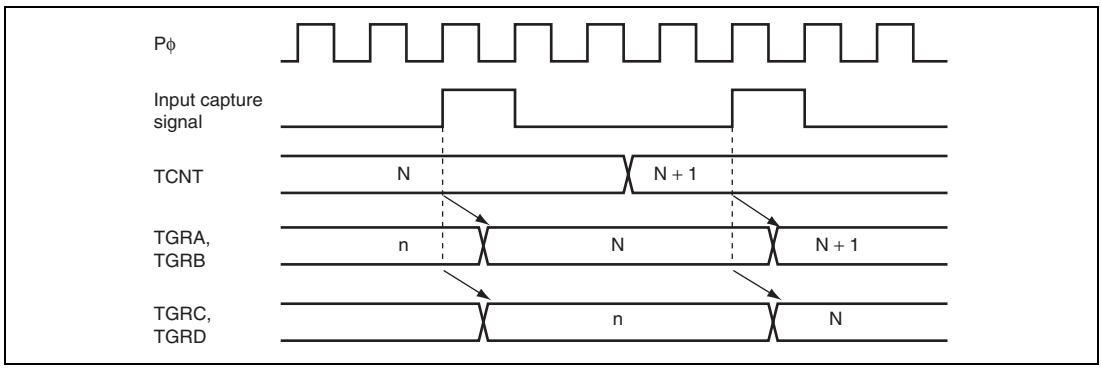


Figure 10.37 Buffer Operation Timing (Input Capture)

10.9.2 Interrupt Signal Timing

(1) TGF Flag Setting Timing in Case of Compare Match

Figure 10.38 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and the TGI interrupt request signal timing.

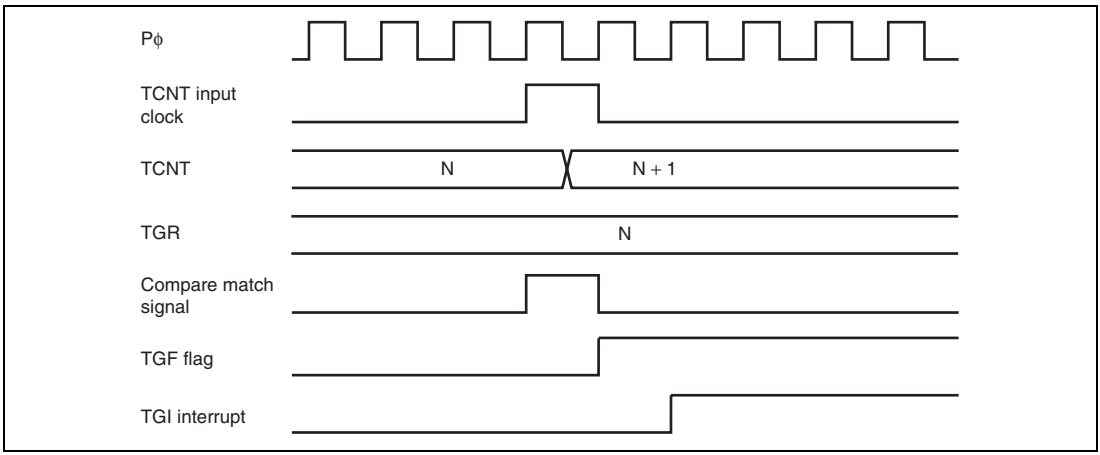


Figure 10.38 TGI Interrupt Timing (Compare Match)

(2) TGF Flag Setting Timing in Case of Input Capture

Figure 10.39 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and the TGI interrupt request signal timing.

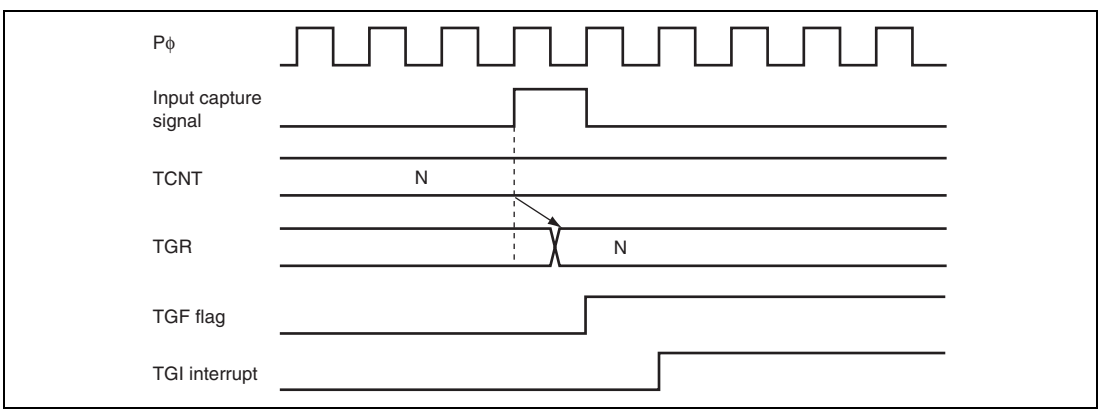


Figure 10.39 TGI Interrupt Timing (Input Capture)

(3) TCFV Flag/TCFU Flag Setting Timing

Figure 10.40 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and the TCIV interrupt request signal timing.

Figure 10.41 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and the TCIU interrupt request signal timing.

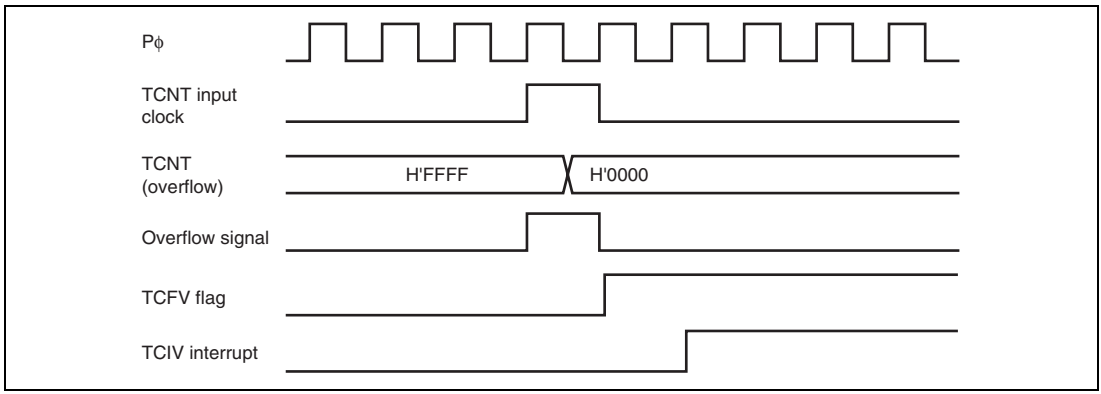


Figure 10.40 TCIV Interrupt Setting Timing

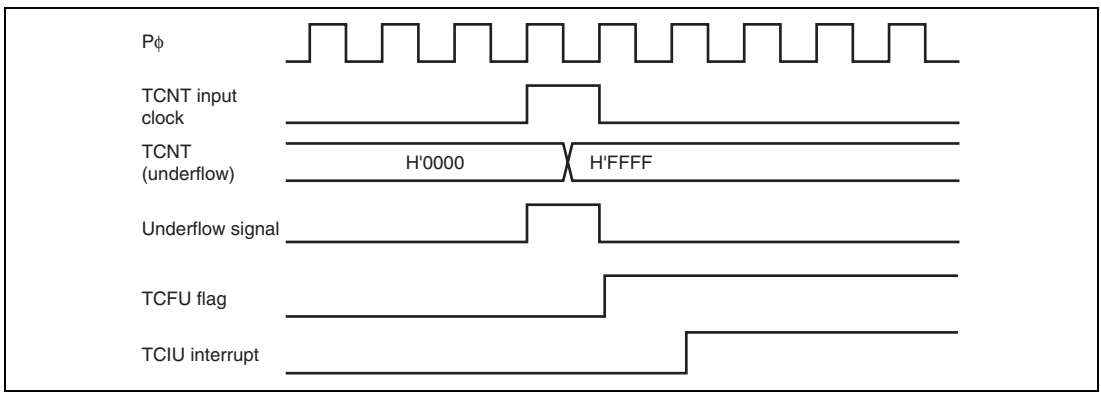


Figure 10.41 TCIU Interrupt Setting Timing

(4) Status Flag Clearing Timing

After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DTC or DMAC is activated, the flag is cleared automatically. Figure 10.42 shows the timing for status flag clearing by the CPU, and figures 10.43 and 10.44 show the timing for status flag clearing by the DTC or DMAC.

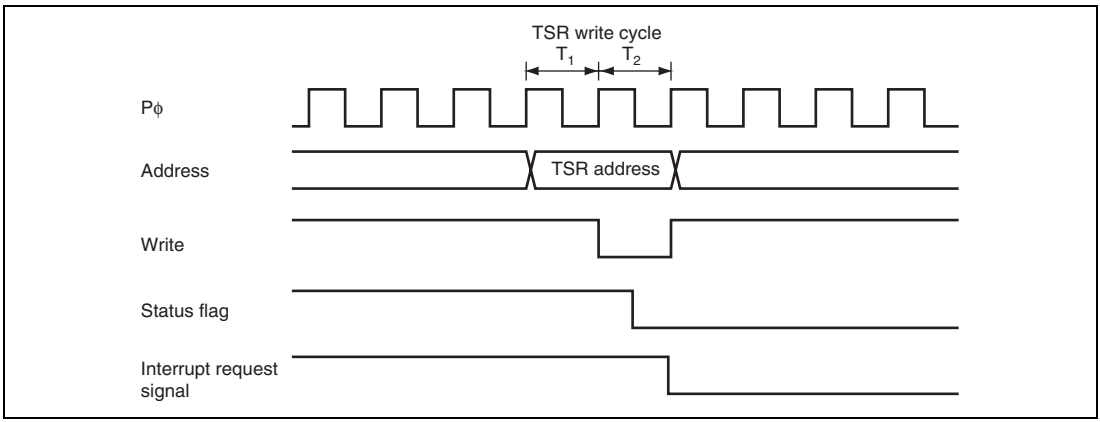


Figure 10.42 Timing for Status Flag Clearing by CPU

The status flag and interrupt request signal are cleared in synchronization with Pϕ after the DTC or DMAC transfer has started, as shown in figure 10.43. If conflict occurs for clearing the status flag and interrupt request signal due to activation of multiple DTC or DMAC transfers, it will take up to five clock cycles (Pϕ) for clearing them, as shown in figure 10.44. The next transfer request is masked for a longer period of either a period until the current transfer ends or a period for five clock cycles (Pϕ) from the beginning of the transfer. Note that in the DTC transfer, the status flag may be cleared during outputting the destination address.

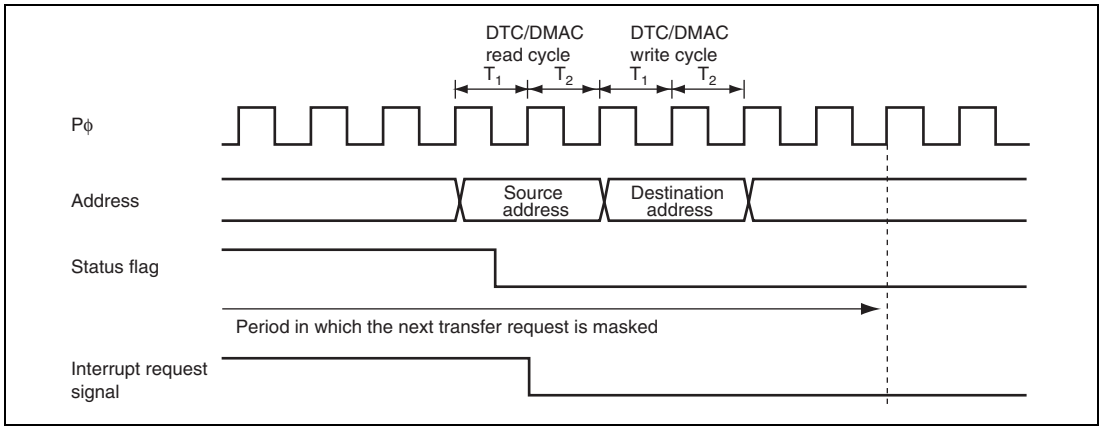


Figure 10.43 Timing for Status Flag Clearing by DTC or DMAC Activation (1)

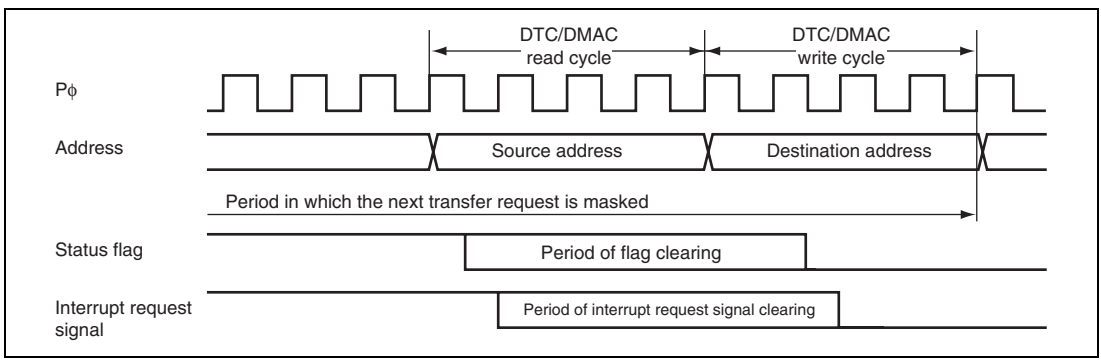


Figure 10.44 Timing for Status Flag Clearing by DTC or DMAC Activation (2)

10.10 Usage Notes

10.10.1 Module Stop State Setting

Operation of the TPU can be disabled or enabled using the module stop control register. The initial setting is for operation of the TPU to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 20, Power-Down Modes.

10.10.2 Input Clock Restrictions

The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width.

In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 10.45 shows the input clock conditions in phase counting mode.

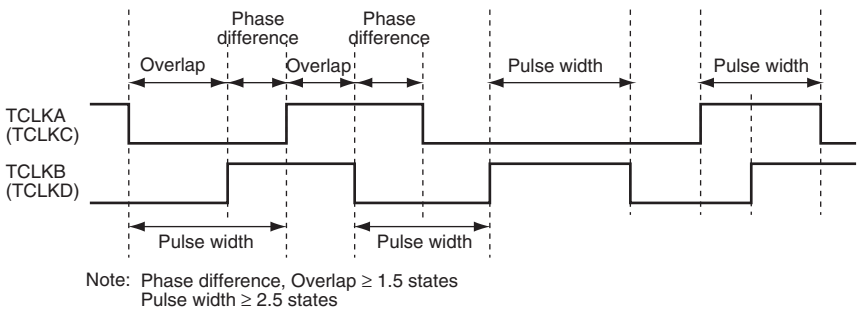


Figure 10.45 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode

10.10.3 Caution on Cycle Setting

When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

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

f: Counter frequency
Pϕ: Operating frequency
N: TGR set value

10.10.4 Conflict between TCNT Write and Clear Operations

If the counter clearing signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed. Figure 10.46 shows the timing in this case.

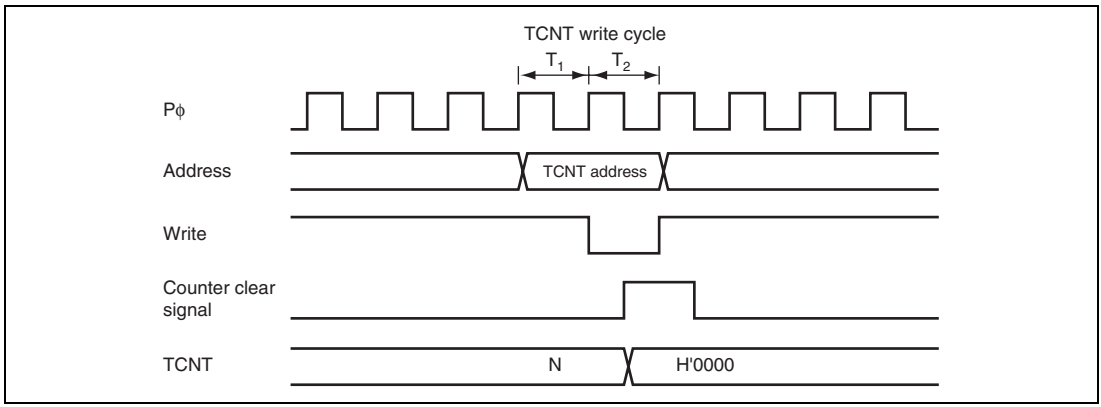


Figure 10.46 Conflict between TCNT Write and Clear Operations

10.10.5 Conflict between TCNT Write and Increment Operations

If incrementing occurs in the T2 state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented. Figure 10.47 shows the timing in this case.

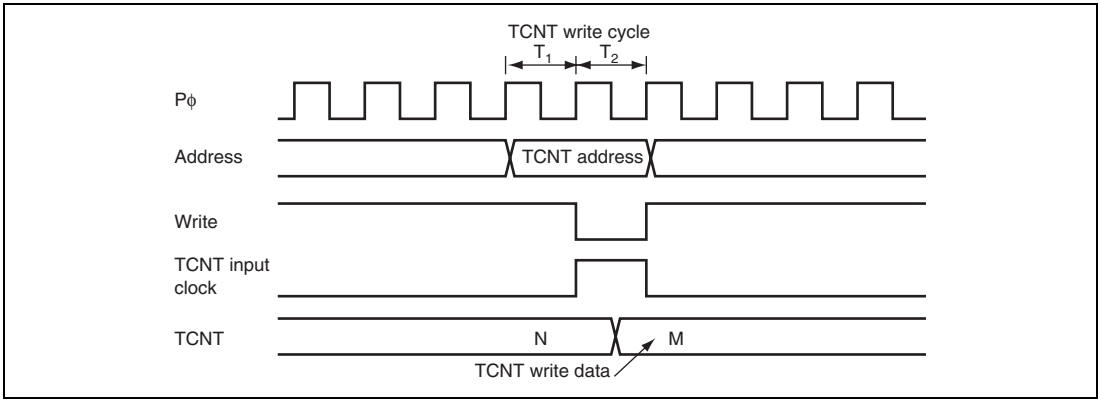


Figure 10.47 Conflict between TCNT Write and Increment Operations

10.10.6 Conflict between TGR Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the TGR write takes precedence and the compare match signal is disabled. A compare match also does not occur when the same value as before is written.

Figure 10.48 shows the timing in this case.

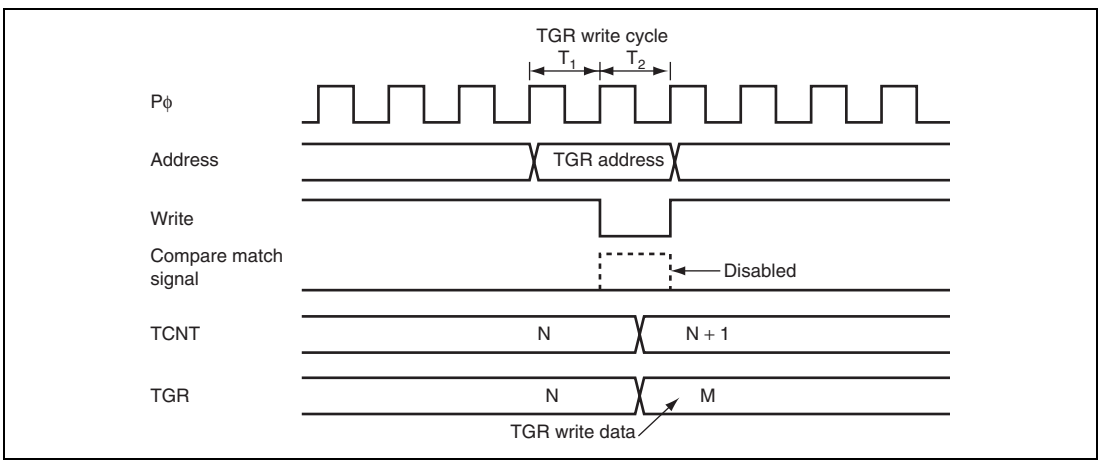


Figure 10.48 Conflict between TGR Write and Compare Match

10.10.7 Conflict between Buffer Register Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the write data.

Figure 10.49 shows the timing in this case.

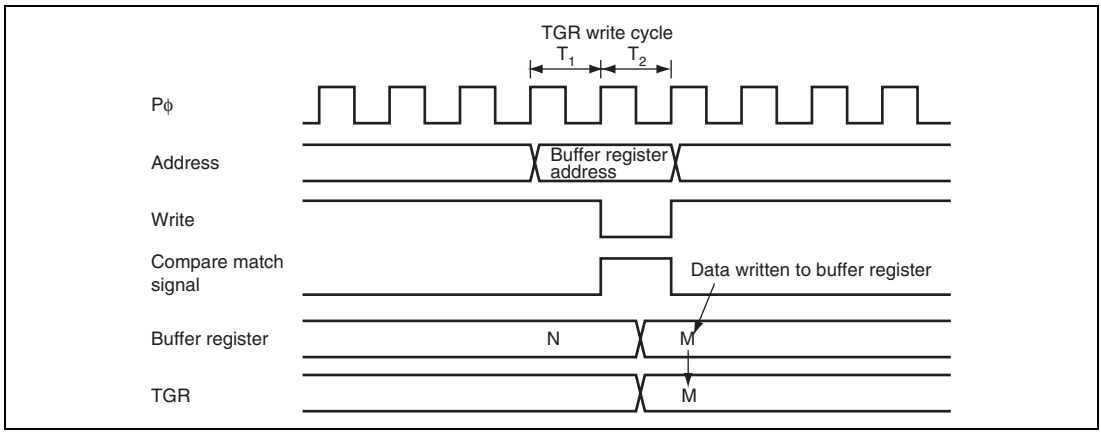


Figure 10.49 Conflict between Buffer Register Write and Compare Match

10.10.8 Conflict between TGR Read and Input Capture

If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is read will be the data after input capture transfer.

Figure 10.50 shows the timing in this case.

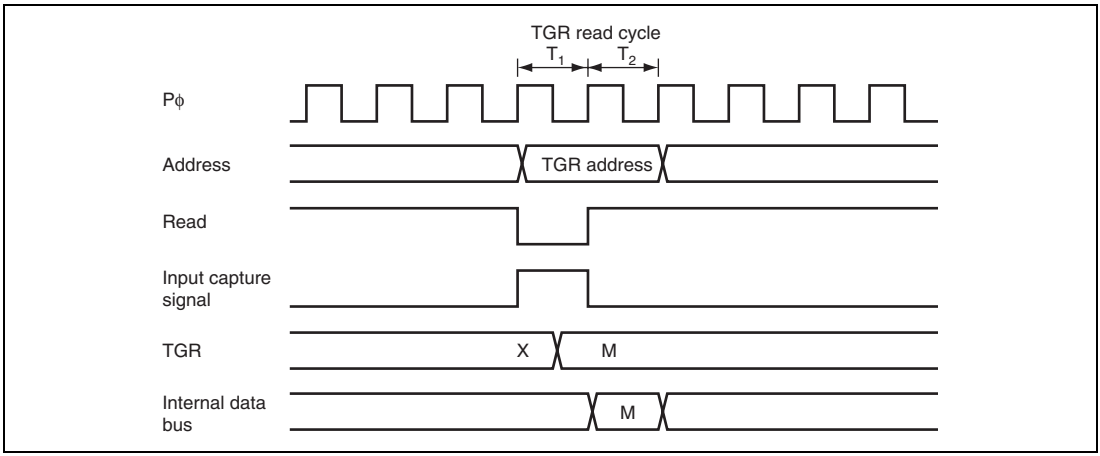


Figure 10.50 Conflict between TGR Read and Input Capture

10.10.9 Conflict between TGR Write and Input Capture

If the input capture signal is generated in the T2 state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

Figure 10.51 shows the timing in this case.

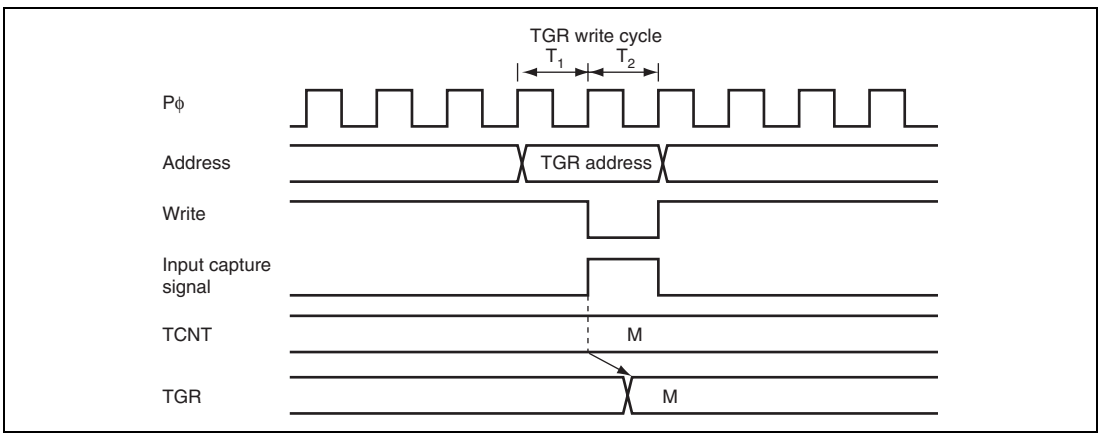


Figure 10.51 Conflict between TGR Write and Input Capture

10.10.10 Conflict between Buffer Register Write and Input Capture

If the input capture signal is generated in the T2 state of a buffer register write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

Figure 10.52 shows the timing in this case.

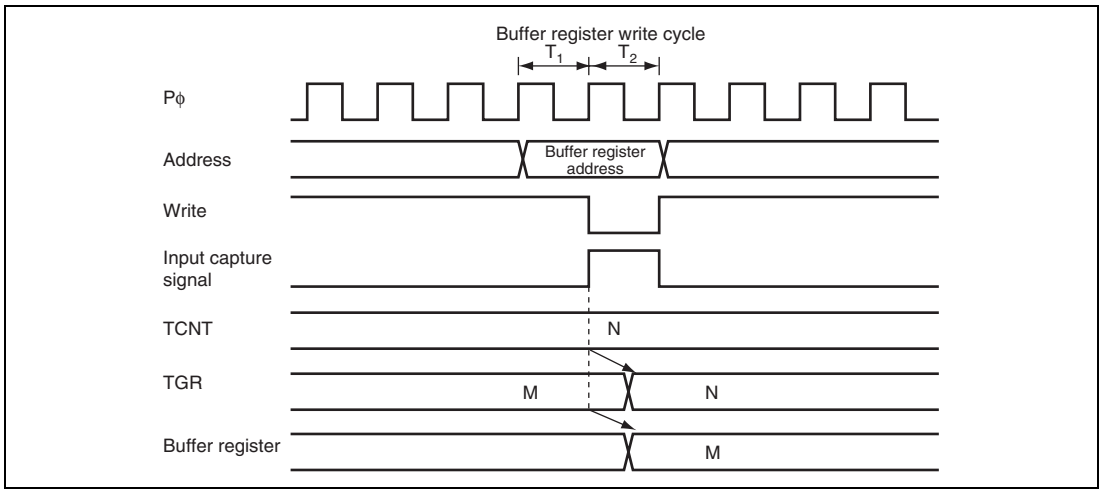


Figure 10.52 Conflict between Buffer Register Write and Input Capture

10.10.11 Conflict between Overflow/Underflow and Counter Clearing

If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

Figure 10.53 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.

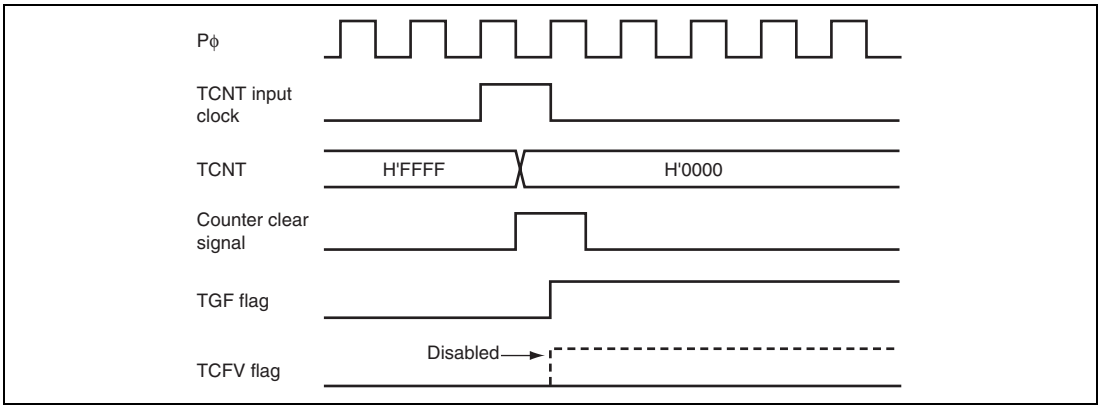


Figure 10.53 Conflict between Overflow and Counter Clearing

10.10.12 Conflict between TCNT Write and Overflow/Underflow

If an overflow/underflow occurs due to increment/decrement in the T2 state of a TCNT write cycle, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 10.54 shows the operation timing when there is conflict between TCNT write and overflow.

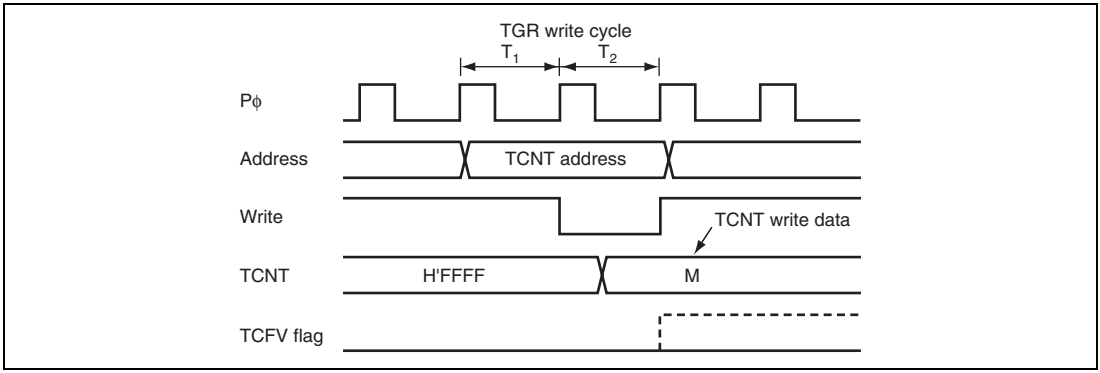


Figure 10.54 Conflict between TCNT Write and Overflow

10.10.13 Multiplexing of I/O Pins

In this LSI, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

10.10.14 Interrupts in Module Stop State

If the module stop state is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC or DTC activation source. Interrupts should therefore be disabled before entering the module stop state.

Section 11 Programmable Pulse Generator (PPG)

The programmable pulse generator (PPG) provides pulse outputs by using the 16-bit timer pulse unit (TPU) as a time base. The PPG pulse outputs are divided into 4-bit groups (groups 3 to 0) that can operate both simultaneously and independently. Figure 11.1 shows a block diagram of the PPG.

11.1 Features

- 16-bit output data
- Four output groups
- Selectable output trigger signals
- Non-overlapping mode
- Can operate together with the data transfer controller (DTC) and DMA controller (DMAC)
- Inverted output can be set
- Module stop state specifiable

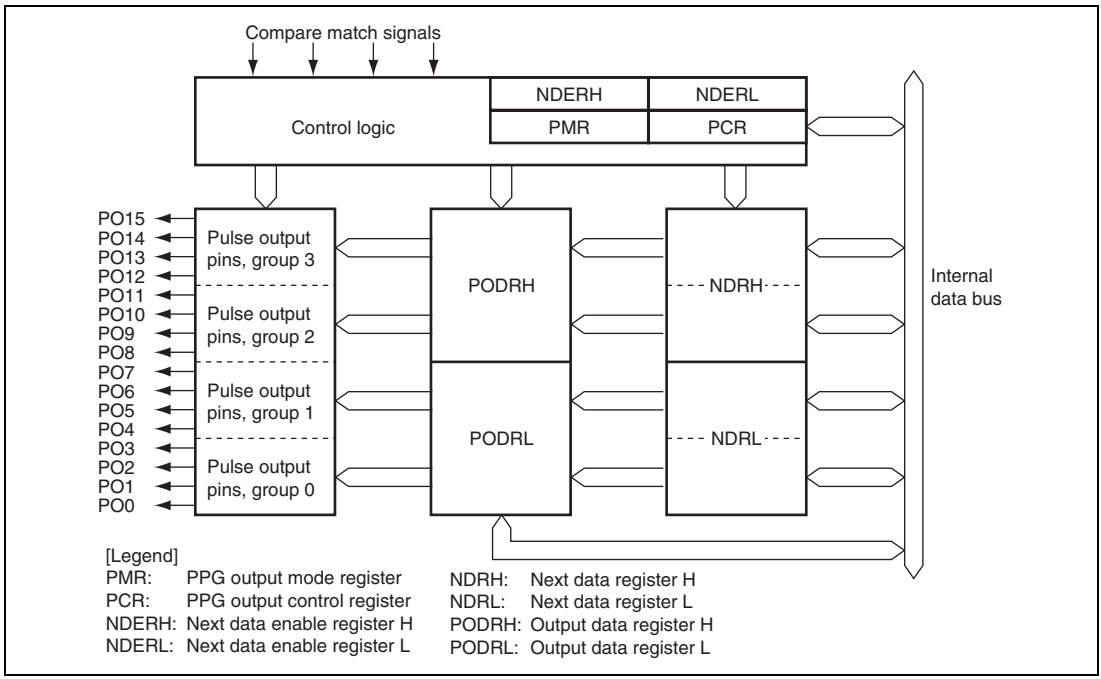


Figure 11.1 Block Diagram of PPG

11.2 Input/Output Pins

Table 11.1 shows the PPG pin configuration.

Table 11.1 Pin Configuration

Pin Name	I/O	Function
PO15	Output	Group 3 pulse output
PO14	Output	
PO13	Output	
PO12	Output	
PO11	Output	Group 2 pulse output
PO10	Output	
PO9	Output	
PO8	Output	
PO7	Output	Group 1 pulse output
PO6	Output	
PO5	Output	
PO4	Output	
PO3	Output	Group 0 pulse output
PO2	Output	
PO1	Output	
PO0	Output	

11.3 Register Descriptions

The PPG has the following registers.

- Next data enable register H (NDERH)
- Next data enable register L (NDERL)
- Output data register H (PODRH)
- Output data register L (PODRL)
- Next data register H (NDRH)
- Next data register L (NDRL)
- PPG output control register (PCR)
- PPG output mode register (PMR)

11.3.1 Next Data Enable Registers H, L (NDERH, NDERL)

NDERH and NDERL enable/disable pulse output on a bit-by-bit basis.

- NDERH

Bit	7	6	5	4	3	2	1	0
Bit Name	NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8
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

- NDERL

Bit	7	6	5	4	3	2	1	0
Bit Name	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0
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

- NDERH

Bit	Bit Name	Initial Value	R/W	Description
7	NDER15	0	R/W	Next Data Enable 15 to 8
6	NDER14	0	R/W	When a bit is set to 1, the value in the corresponding NDRH bit is transferred to the PODRH bit by the selected output trigger. Values are not transferred from NDRH to PODRH for cleared bits.
5	NDER13	0	R/W	
4	NDER12	0	R/W	
3	NDER11	0	R/W	
2	NDER10	0	R/W	
1	NDER9	0	R/W	
0	NDER8	0	R/W	

- NDERL

Bit	Bit Name	Initial Value	R/W	Description
7	NDER7	0	R/W	Next Data Enable 7 to 0
6	NDER6	0	R/W	When a bit is set to 1, the value in the corresponding NDRL bit is transferred to the PODRL bit by the selected output trigger. Values are not transferred from NDRL to PODRL for cleared bits.
5	NDER5	0	R/W	
4	NDER4	0	R/W	
3	NDER3	0	R/W	
2	NDER2	0	R/W	
1	NDER1	0	R/W	
0	NDER0	0	R/W	

11.3.2 Output Data Registers H, L (PODRH, PODRL)

PODRH and PODRL store output data for use in pulse output. A bit that has been set for pulse output by NDER is read-only and cannot be modified.

- PODRH

Bit	7	6	5	4	3	2	1	0
Bit Name	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8
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

- PODRL

Bit	7	6	5	4	3	2	1	0
Bit Name	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
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

- PODRH

Bit	Bit Name	Initial Value	R/W	Description
7	POD15	0	R/W	Output Data Register 15 to 8
6	POD14	0	R/W	For bits which have been set to pulse output by NDERH, the output trigger transfers NDRH values to this register during PPG operation. While NDERH is set to 1, the CPU cannot write to this register. While NDERH is cleared, the initial output value of the pulse can be set.
5	POD13	0	R/W	
4	POD12	0	R/W	
3	POD11	0	R/W	
2	POD10	0	R/W	
1	POD9	0	R/W	
0	POD8	0	R/W	

- PODRL

Bit	Bit Name	Initial Value	R/W	Description
7	POD7	0	R/W	Output Data Register 7 to 0
6	POD6	0	R/W	For bits which have been set to pulse output by NDERL, the output trigger transfers NDRL values to this register during PPG operation. While NDERL is set to 1, the CPU cannot write to this register. While NDERL is cleared, the initial output value of the pulse can be set.
5	POD5	0	R/W	
4	POD4	0	R/W	
3	POD3	0	R/W	
2	POD2	0	R/W	
1	POD1	0	R/W	
0	POD0	0	R/W	

11.3.3 Next Data Registers H, L (NDRH, NDRL)

NDRH and NDRL store the next data for pulse output. The NDR addresses differ depending on whether pulse output groups have the same output trigger or different output triggers.

- NDRH

Bit	7	6	5	4	3	2	1	0
Bit Name	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8
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

- NDRL

Bit	7	6	5	4	3	2	1	0
Bit Name	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0
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

- NDRH

  If pulse output groups 2 and 3 have the same output trigger, all eight bits are mapped to the same address and can be accessed at one time, as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR15	0	R/W	Next Data Register 15 to 8
6	NDR14	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3	NDR11	0	R/W	
2	NDR10	0	R/W	
1	NDR9	0	R/W	
0	NDR8	0	R/W	

If pulse output groups 2 and 3 have different output triggers, the upper four bits and lower four bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR15	0	R/W	Next Data Register 15 to 12
6	NDR14	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
3	NDR11	0	R/W	Next Data Register 11 to 8
2	NDR10	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
1	NDR9	0	R/W	
0	NDR8	0	R/W	

- NDRL

  If pulse output groups 0 and 1 have the same output trigger, all eight bits are mapped to the same address and can be accessed at one time, as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR7	0	R/W	Next Data Register 7 to 0
6	NDR6	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3	NDR3	0	R/W	
2	NDR2	0	R/W	
1	NDR1	0	R/W	
0	NDR0	0	R/W	

If pulse output groups 0 and 1 have different output triggers, the upper four bits and lower four bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR7	0	R/W	Next Data Register 7 to 4
6	NDR6	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
3	NDR3	0	R/W	Next Data Register 3 to 0
2	NDR2	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
1	NDR1	0	R/W	
0	NDR0	0	R/W	

11.3.4 PPG Output Control Register (PCR)

PCR selects output trigger signals on a group-by-group basis. For details on output trigger selection, refer to section 11.3.5, PPG Output Mode Register (PMR).

Bit	7	6	5	4	3	2	1	0
Bit Name	G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0
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	Bit Name	Initial Value	R/W	Description
7	G3CMS1	1	R/W	Group 3 Compare Match Select 1 and 0
6	G3CMS0	1	R/W	These bits select output trigger of pulse output group 3. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
5	G2CMS1	1	R/W	Group 2 Compare Match Select 1 and 0
4	G2CMS0	1	R/W	These bits select output trigger of pulse output group 2. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
3	G1CMS1	1	R/W	Group 1 Compare Match Select 1 and 0
2	G1CMS0	1	R/W	These bits select output trigger of pulse output group 1. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
1	G0CMS1	1	R/W	Group 0 Compare Match Select 1 and 0
0	G0CMS0	1	R/W	These bits select output trigger of pulse output group 0. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3

11.3.5 PPG Output Mode Register (PMR)

PMR selects the pulse output mode of the PPG for each group. If inverted output is selected, a low-level pulse is output when PODRH is 1 and a high-level pulse is output when PODRH is 0. If non-overlapping operation is selected, PPG updates its output values at compare match A or B of the TPU that becomes the output trigger. For details, refer to section 11.4.4, Non-Overlapping Pulse Output.

Bit	7	6	5	4	3	2	1	0
Bit Name	G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV
Initial Value	1	1	1	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	G3INV	1	R/W	Group 3 Inversion Selects direct output or inverted output for pulse output group 3. 0: Inverted output 1: Direct output
6	G2INV	1	R/W	Group 2 Inversion Selects direct output or inverted output for pulse output group 2. 0: Inverted output 1: Direct output
5	G1INV	1	R/W	Group 1 Inversion Selects direct output or inverted output for pulse output group 1. 0: Inverted output 1: Direct output
4	G0INV	1	R/W	Group 0 Inversion Selects direct output or inverted output for pulse output group 0. 0: Inverted output 1: Direct output

Bit	Bit Name	Initial Value	R/W	Description
3	G3NOV	0	R/W	Group 3 Non-Overlap Selects normal or non-overlapping operation for pulse output group 3. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)
2	G2NOV	0	R/W	Group 2 Non-Overlap Selects normal or non-overlapping operation for pulse output group 2. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)
1	G1NOV	0	R/W	Group 1 Non-Overlap Selects normal or non-overlapping operation for pulse output group 1. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)
0	G0NOV	0	R/W	Group 0 Non-Overlap Selects normal or non-overlapping operation for pulse output group 0. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)

11.4 Operation

Figure 11.2 shows a schematic diagram of the PPG. PPG pulse output is enabled when the corresponding bits in NDER are set to 1. An initial output value is determined by its corresponding PODR initial setting. When the compare match event specified by PCR occurs, the corresponding NDR bit contents are transferred to PODR to update the output values. Sequential output of data of up to 16 bits is possible by writing new output data to NDR before the next compare match.

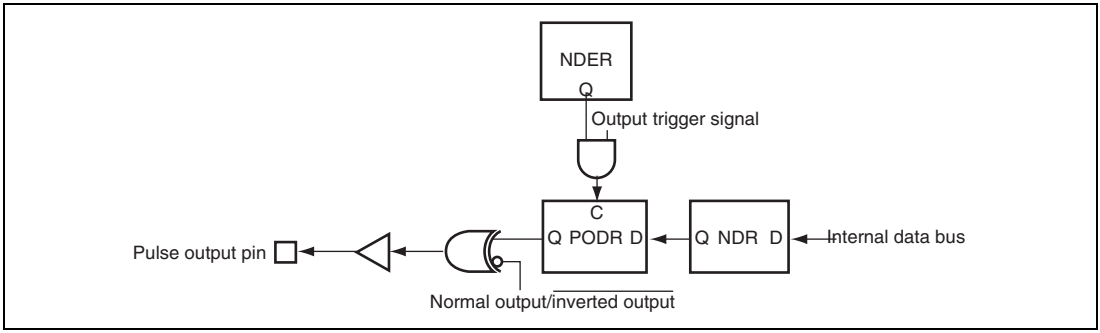


Figure 11.2 Schematic Diagram of PPG

11.4.1 Output Timing

If pulse output is enabled, the NDR contents are transferred to PODR and output when the specified compare match event occurs. Figure 11.3 shows the timing of these operations for the case of normal output in groups 2 and 3, triggered by compare match A.

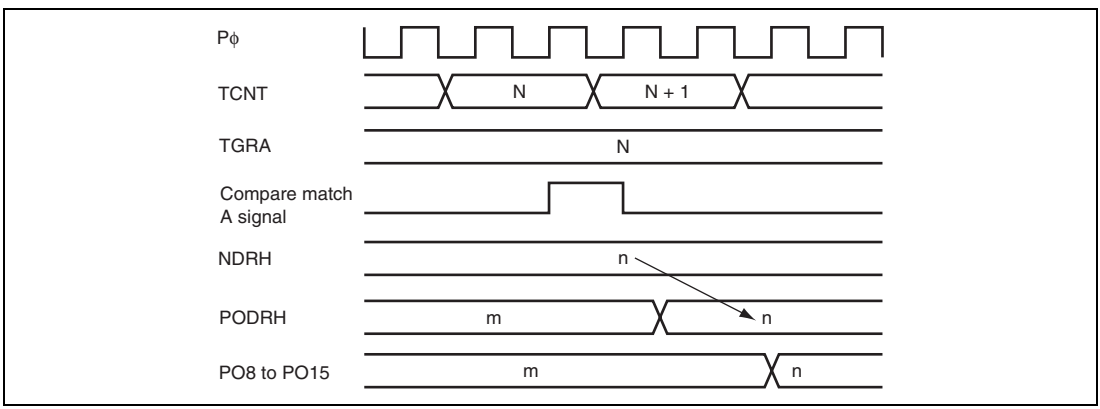


Figure 11.3 Timing of Transfer and Output of NDR Contents (Example)

11.4.2 Sample Setup Procedure for Normal Pulse Output

Figure 11.4 shows a sample procedure for setting up normal pulse output.

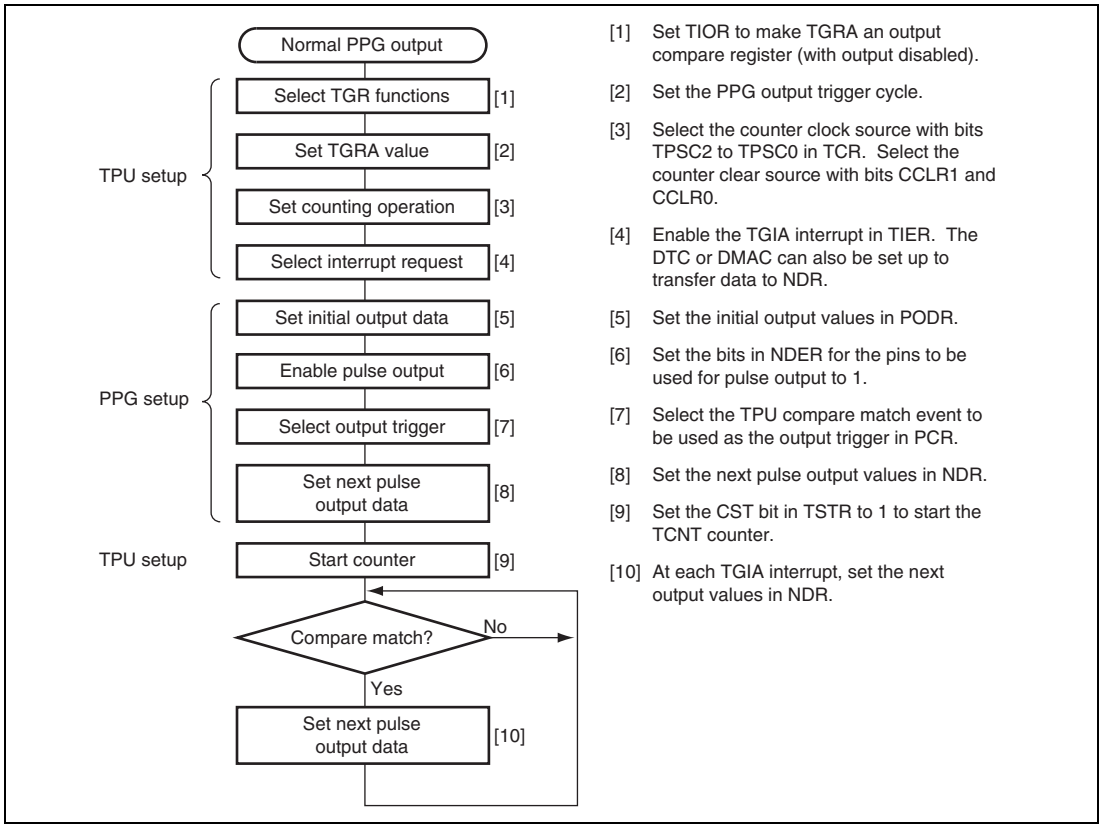


Figure 11.4 Setup Procedure for Normal Pulse Output (Example)

11.4.3 Example of Normal Pulse Output (Example of 5-Phase Pulse Output)

Figure 11.5 shows an example in which pulse output is used for cyclic 5-phase pulse output.

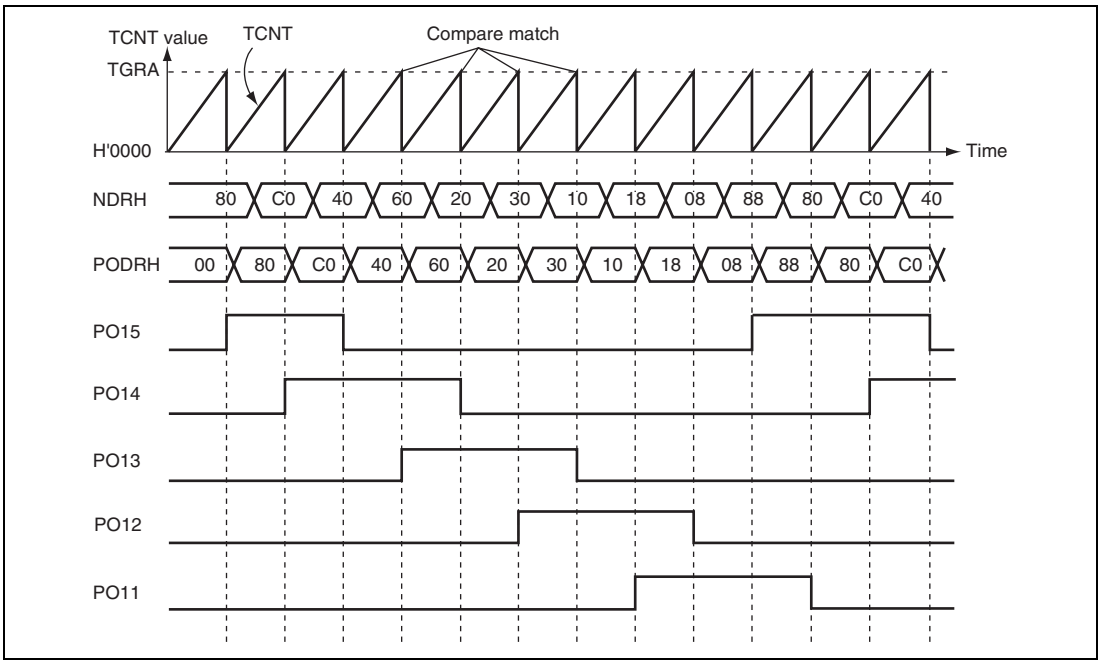


Figure 11.5 Normal Pulse Output Example (5-Phase Pulse Output)

1. Set up TGRA in TPU which is used as the output trigger to be an output compare register. Set a cycle in TGRA so the counter will be cleared by compare match A. Set the TGIEA bit in TIER to 1 to enable the compare match/input capture A (TGIA) interrupt.
2. Write H'F8 to NDERH, and set bits G3CMS1, G3CMS0, G2CMS1, and G2CMS0 in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Write output data H'80 in NDRH.
3. The timer counter in the TPU channel starts. When compare match A occurs, the NDRH contents are transferred to PODRH and output. The TGIA interrupt handling routine writes the next output data (H'C0) in NDRH.
4. 5-phase pulse output (one or two phases active at a time) can be obtained subsequently by writing H'40, H'60, H'20, H'30, H'10, H'18, H'08, H'88... at successive TGIA interrupts.

   If the DTC or DMAC is set for activation by the TGIA interrupt, pulse output can be obtained without imposing a load on the CPU.

11.4.4 Non-Overlapping Pulse Output

During non-overlapping operation, transfer from NDR to PODR is performed as follows:

- At compare match A, the NDR bits are always transferred to PODR.
- At compare match B, the NDR bits are transferred only if their value is 0. The NDR bits are not transferred if their value is 1.

Figure 11.6 illustrates the non-overlapping pulse output operation.

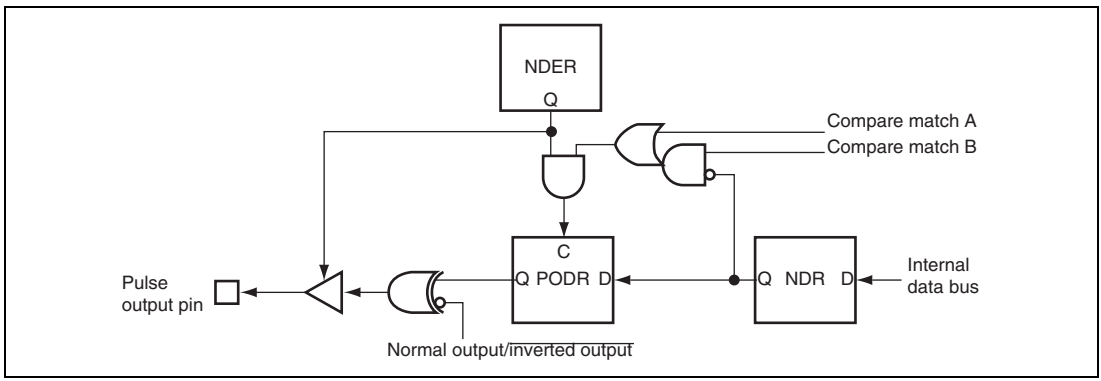


Figure 11.6 Non-Overlapping Pulse Output

Therefore, 0 data can be transferred ahead of 1 data by making compare match B occur before compare match A.

The NDR contents should not be altered during the interval from compare match B to compare match A (the non-overlapping margin).

This can be accomplished by having the TGIA interrupt handling routine write the next data in NDR, or by having the TGIA interrupt activate the DTC or DMAC. Note, however, that the next data must be written before the next compare match B occurs.

Figure 11.7 shows the timing of this operation.

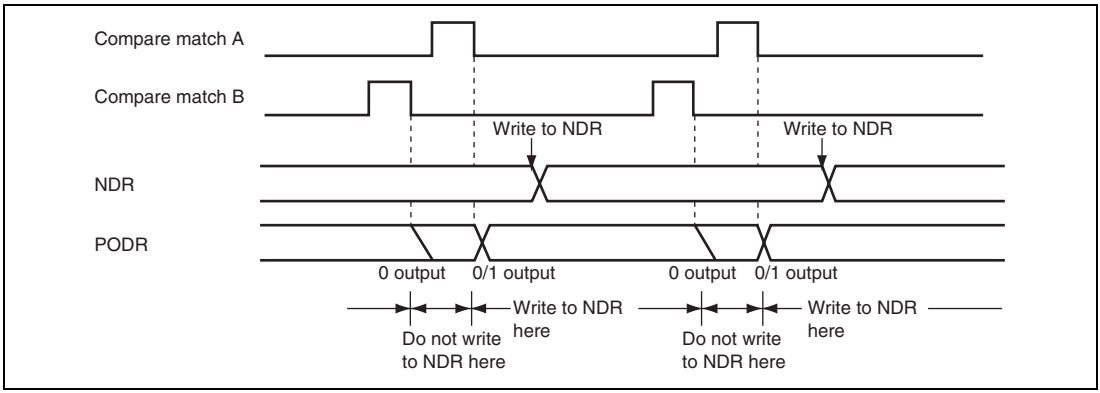


Figure 11.7 Non-Overlapping Operation and NDR Write Timing

11.4.5 Sample Setup Procedure for Non-Overlapping Pulse Output

Figure 11.8 shows a sample procedure for setting up non-overlapping pulse output.

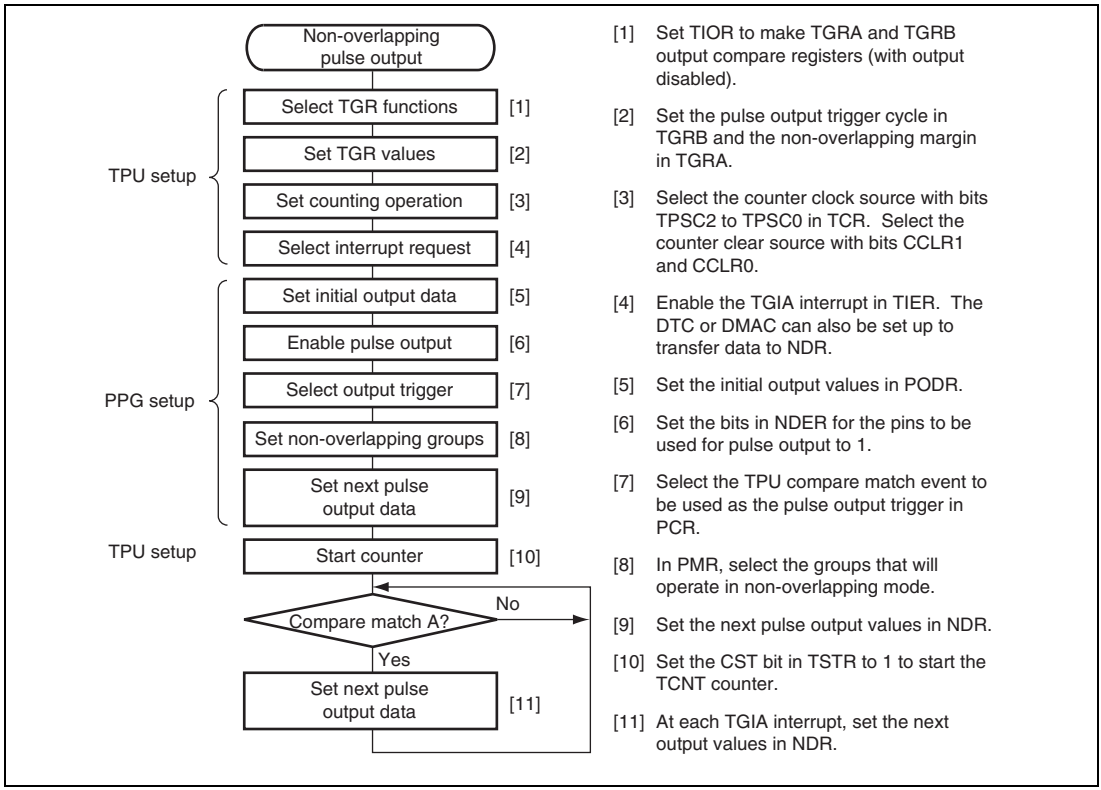


Figure 11.8 Setup Procedure for Non-Overlapping Pulse Output (Example)

11.4.6 Example of Non-Overlapping Pulse Output (Example of 4-Phase Complementary Non-Overlapping Pulse Output)

Figure 11.9 shows an example in which pulse output is used for 4-phase complementary non-overlapping pulse output.

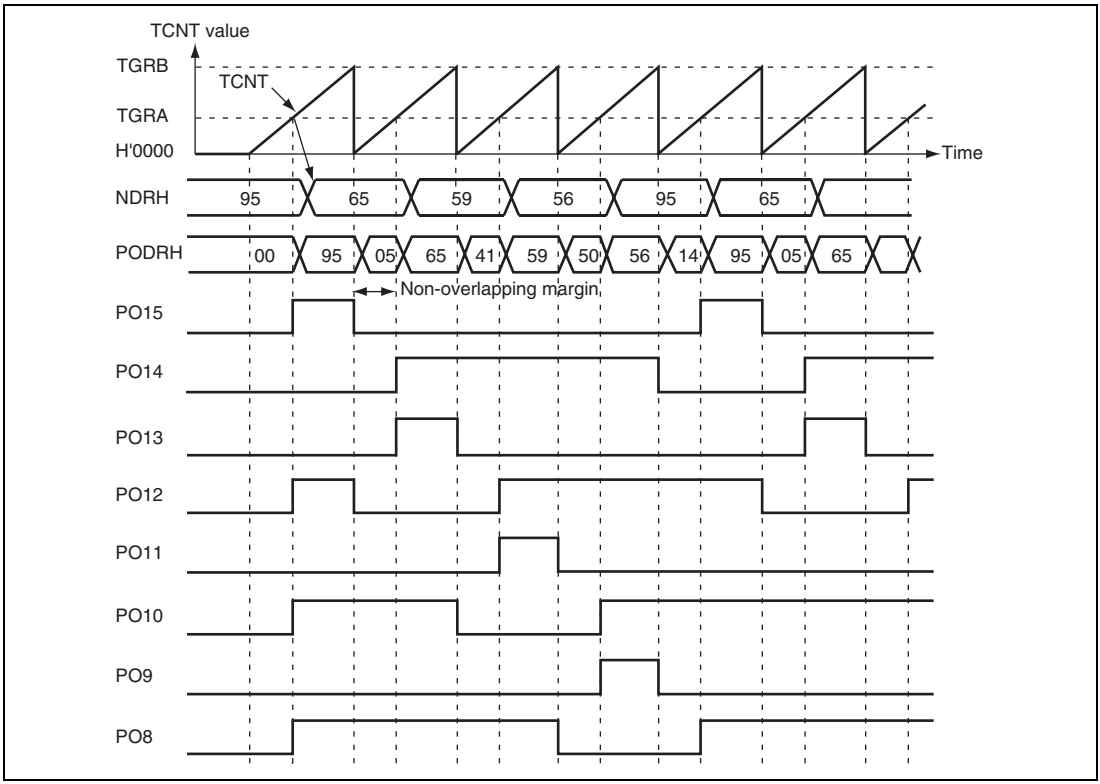


Figure 11.9 Non-Overlapping Pulse Output Example (4-Phase Complementary)

1. Set up the TPU channel to be used as the output trigger channel so that TGRA and TGRB are output compare registers. Set the cycle in TGRB and the non-overlapping margin in TGRA, and set the counter to be cleared by compare match B. Set the TGIEA bit in TIER to 1 to enable the TGIA interrupt.
2. Write H'FF to NDERH, and set bits G3CMS1, G3CMS0, G2CMS1, and G2CMS0 in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger.

   Set bits G3NOV and G2NOV in PMR to 1 to select non-overlapping pulse output.

   Write output data H'95 to NDRH.
3. The timer counter in the TPU channel starts. When a compare match with TGRB occurs, outputs change from 1 to 0. When a compare match with TGRA occurs, outputs change from 0 to 1 (the change from 0 to 1 is delayed by the value set in TGRA).

   The TGIA interrupt handling routine writes the next output data (H'65) to NDRH.

4. 4-phase complementary non-overlapping pulse output can be obtained subsequently by writing H'59, H'56, H'95... at successive TGIA interrupts.

   If the DTC or DMAC is set for activation by a TGIA interrupt, pulse can be output without imposing a load on the CPU.

11.4.7 Inverted Pulse Output

If the G3INV, G2INV, G1INV, and G0INV bits in PMR are cleared to 0, values that are the inverse of the PODR contents can be output.

Figure 11.10 shows the outputs when the G3INV and G2INV bits are cleared to 0, in addition to the settings of figure 11.9.

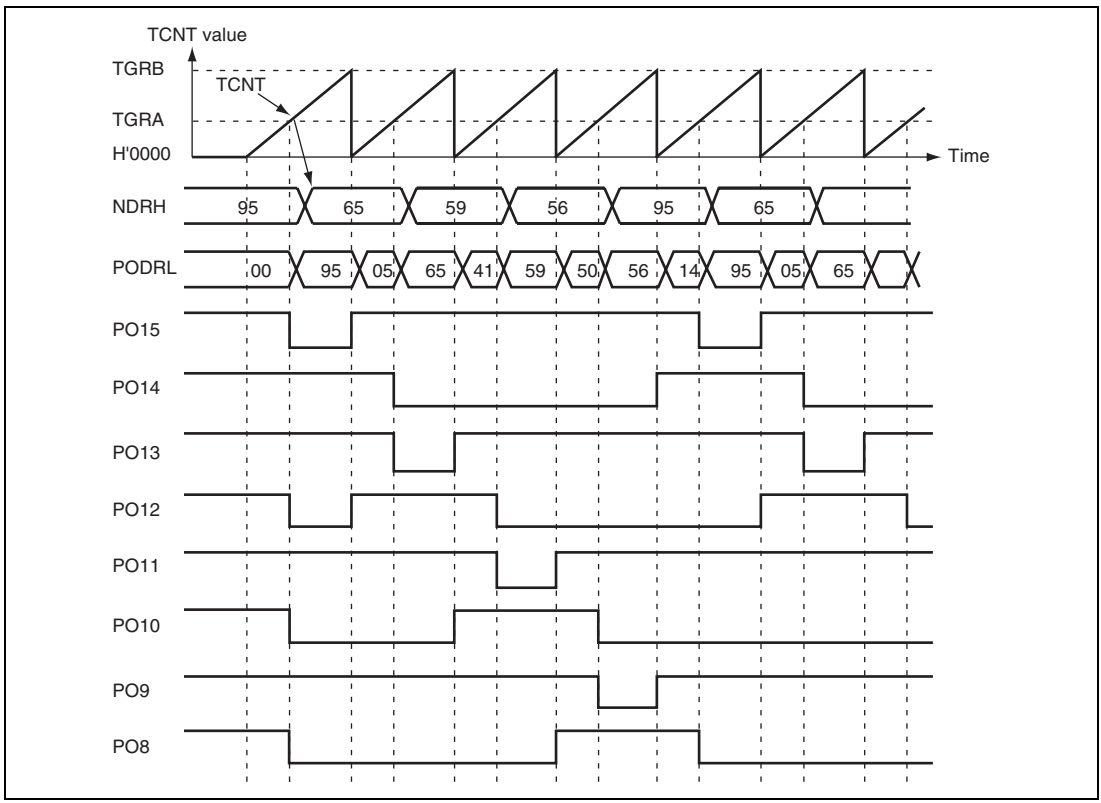


Figure 11.10 Inverted Pulse Output (Example)

11.4.8 Pulse Output Triggered by Input Capture

Pulse output can be triggered by TPU input capture as well as by compare match. If TGRA functions as an input capture register in the TPU channel selected by PCR, pulse output will be triggered by the input capture signal.

Figure 11.11 shows the timing of this output.

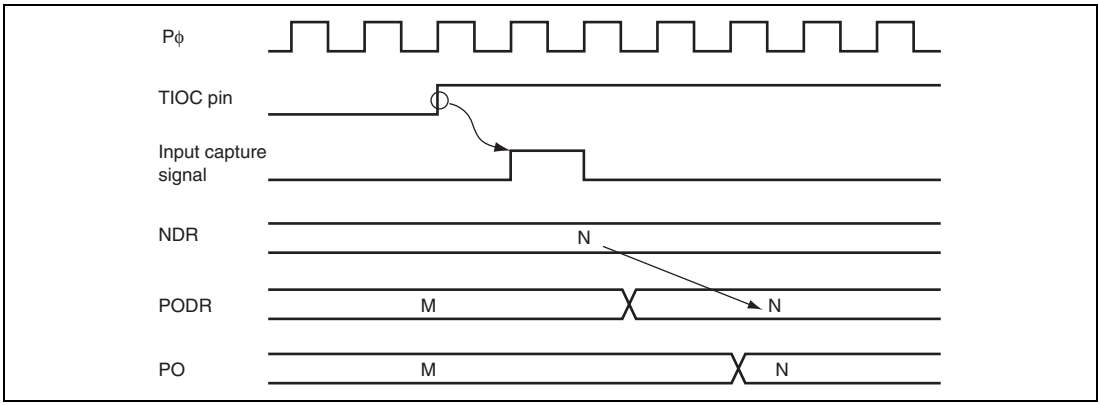


Figure 11.11 Pulse Output Triggered by Input Capture (Example)

11.5 Usage Notes

11.5.1 Module Stop State Setting

PPG operation can be disabled or enabled using the module stop control register. The initial value is for PPG operation to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 20, Power-Down Modes.

11.5.2 Operation of Pulse Output Pins

Pins PO0 to PO15 are also used for other peripheral functions such as the TPU. When output by another peripheral function is enabled, the corresponding pins cannot be used for pulse output. Note, however, that data transfer from NDR bits to PODR bits takes place, regardless of the usage of the pins.

Pin functions should be changed only under conditions in which the output trigger event will not occur.

Section 12 8-Bit Timers (TMR)

This LSI has two units (unit 0 and unit 1) of an on-chip 8-bit timer module that comprise two 8-bit counter channels, totaling four channels. The 8-bit timer module can be used to count external events and also be used as a multifunction timer in a variety of applications, such as generation of counter reset, interrupt requests, and pulse output with a desired duty cycle using a compare-match signal with two registers.

Figures 12.1 and 12.2 show block diagrams of the 8-bit timer module (unit 0 and unit 1).

This section describes unit 0 (channels 0 and 1), which has the same functions as the other unit.

12.1 Features

- Selection of seven clock sources

  The counters can be driven by one of six internal clock signals (Pϕ/2, Pϕ/8, Pϕ/32, Pϕ/64, Pϕ/1024, or Pϕ/8192) or an external clock input.
- Selection of three ways to clear the counters

  The counters can be cleared on compare match A or B, or by an external reset signal.
- Timer output control by a combination of two compare match signals

  The timer output signal in each channel is controlled by a combination of two independent compare match signals, enabling the timer to output pulses with a desired duty cycle or PWM output.
- Cascading of two channels (TMR_0 and TMR_1)

  Operation as a 16-bit timer is possible, using TMR_0 for the upper 8 bits and TMR_1 for the lower 8 bits (16-bit count mode).

  TMR_1 can be used to count TMR_0 compare matches (compare match count mode).
- Three interrupt sources

  Compare match A, compare match B, and overflow interrupts can be requested independently.
- Generation of trigger to start A/D converter conversion
- Module stop state specifiable

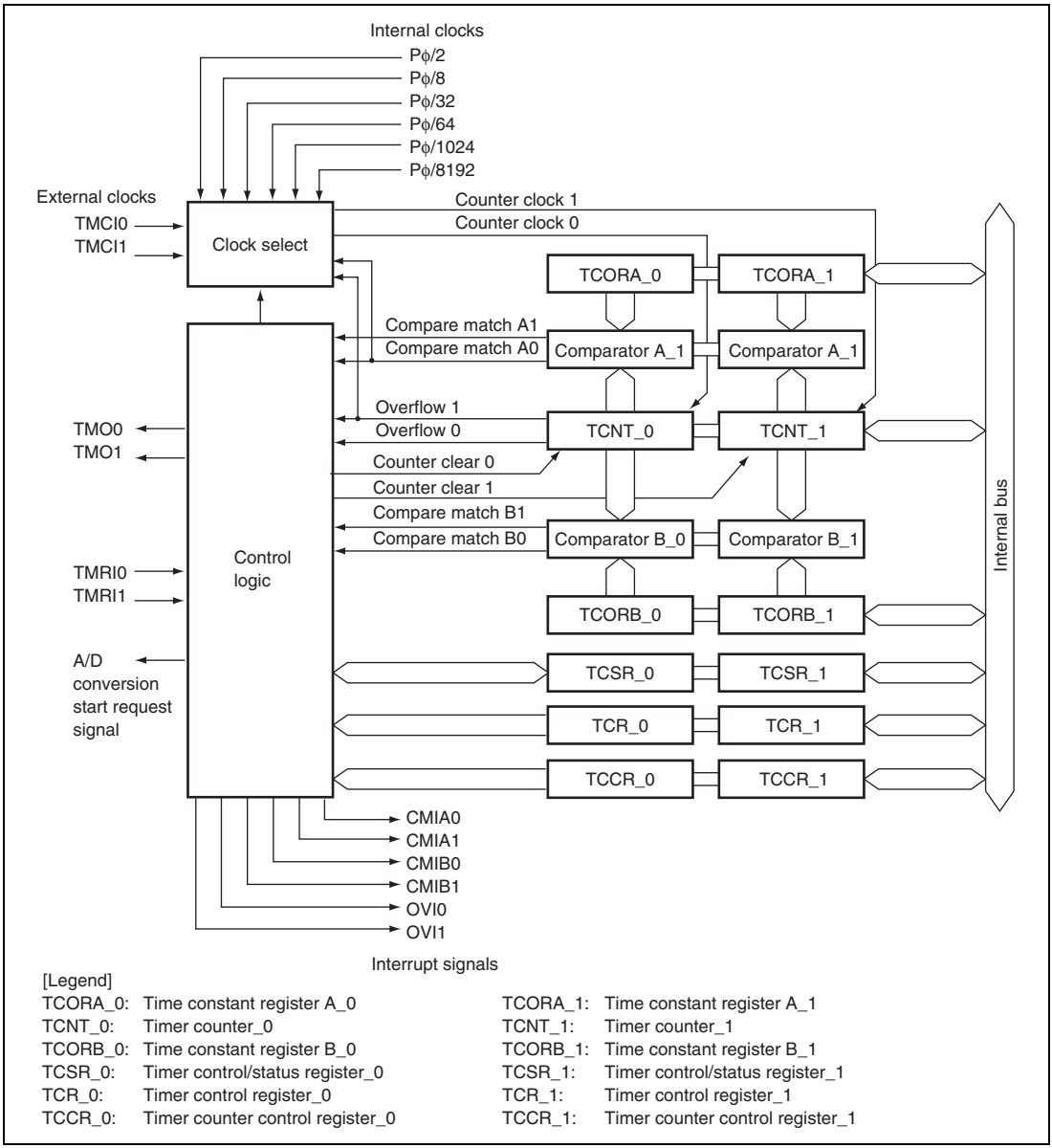
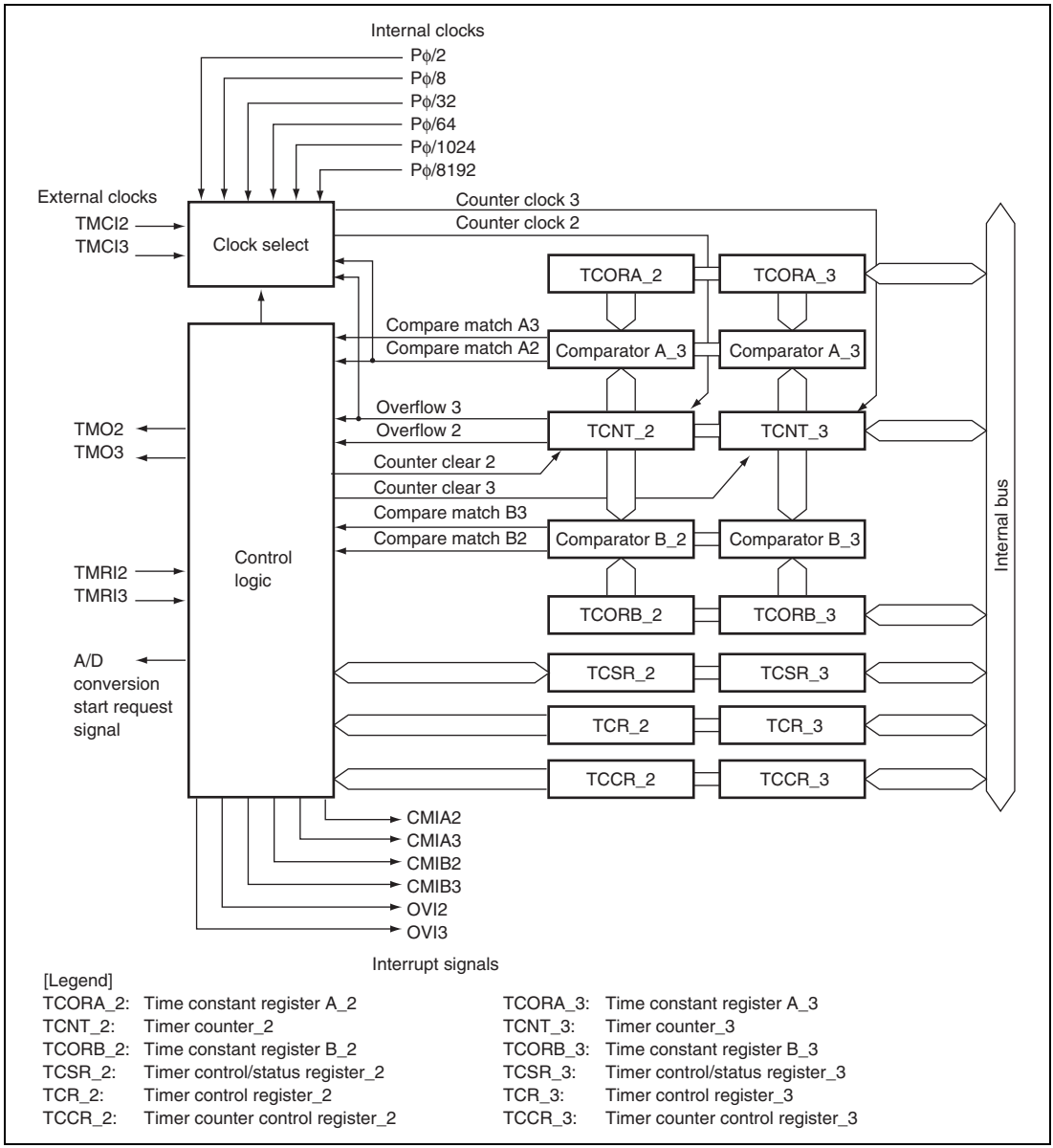


Figure 12.1 Block Diagram of 8-Bit Timer Module (Unit 0)



[Legend]

TCORA_2:	Time constant register A_2	TCORA_3:	Time constant register A_3
TCNT_2:	Timer counter_2	TCNT_3:	Timer counter_3
TCORB_2:	Time constant register B_2	TCORB_3:	Time constant register B_3
TCSR_2:	Timer control/status register_2	TCSR_3:	Timer control/status register_3
TCR_2:	Timer control register_2	TCR_3:	Timer control register_3
TCCR_2:	Timer counter control register_2	TCCR_3:	Timer counter control register_3

Figure 12.2 Block Diagram of 8-Bit Timer Module (Unit 1)

12.2 Input/Output Pins

Table 12.1 shows the pin configuration of the TMR.

Table 12.1 Pin Configuration

Unit	Channel	Name	Symbol	I/O	Function
0	0	Timer output pin	TMO0	Output	Outputs compare match
		Timer clock input pin	TMCI0	Input	Inputs external clock for counter
		Timer reset input pin	TMRI0	Input	Inputs external reset to counter
	1	Timer output pin	TMO1	Output	Outputs compare match
		Timer clock input pin	TMCI1	Input	Inputs external clock for counter
		Timer reset input pin	TMRI1	Input	Inputs external reset to counter
1	2	Timer output pin	TMO2	Output	Outputs compare match
		Timer clock input pin	TMCI2	Input	Inputs external clock for counter
		Timer reset input pin	TMRI2	Input	Inputs external reset to counter
	3	Timer output pin	TMO3	Output	Outputs compare match
		Timer clock input pin	TMCI3	Input	Inputs external clock for counter
		Timer reset input pin	TMRI3	Input	Inputs external reset to counter

12.3 Register Descriptions

The TMR has the following registers.

Unit 0:

- Channel 0
  - Timer counter_0 (TCNT_0)
  - Time constant register A_0 (TCORA_0)
  - Time constant register B_0 (TCORB_0)
  - Timer control register_0 (TCR_0)
  - Timer counter control register_0 (TCCR_0)
  - Timer control/status register_0 (TCSR_0)

- Channel 1
  - Timer counter_1 (TCNT_1)
  - Time constant register A_1 (TCORA_1)
  - Time constant register B_1 (TCORB_1)
  - Timer control register_1 (TCR_1)
  - Timer counter control register_1 (TCCR_1)
  - Timer control/status register_1 (TCSR_1)

Unit 1:

- Channel 2
  - Timer counter_2 (TCNT_2)
  - Time constant register A_2 (TCORA_2)
  - Time constant register B_2 (TCORB_2)
  - Timer control register_2 (TCR_2)
  - Timer counter control register_2 (TCCR_2)
  - Timer control/status register_2 (TCSR_2)
- Channel 3
  - Timer counter_3 (TCNT_3)
  - Time constant register A_3 (TCORA_3)
  - Time constant register B_3 (TCORB_3)
  - Timer control register_3 (TCR_3)
  - Timer counter control register_3 (TCCR_3)
  - Timer control/status register_3 (TCSR_3)

12.3.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT_0 and TCNT_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction. Bits CKS2 to CKS0 in TCR and bits ICKS1 and ICKS0 in TCCR are used to select a clock. TCNT can be cleared by an external reset input signal, compare match A signal, or compare match B signal. Which signal is to be used for clearing is selected by bits CCLR1 and CCLR0 in TCR. When TCNT overflows from H'FF to H'00, bit OVF in TCSR is set to 1. TCNT is initialized to H'00.

	TCNT_0								TCNT_1							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Bit Name																
Initial Value	0	0	0	0	0	0	0	0	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	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12.3.2 Time Constant Register A (TCORA)

TCORA is an 8-bit readable/writable register. TCORA_0 and TCORA_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction. The value in TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding CMFA flag in TCSR is set to 1. Note however that comparison is disabled during the T2 state of a TCORA write cycle. The timer output from the TMO pin can be freely controlled by this compare match signal (compare match A) and the settings of bits OS1 and OS0 in TCSR. TCORA is initialized to H'FF.

	TCORA_0								TCORA_1							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Bit Name																
Initial Value	1	1	1	1	1	1	1	1	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	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12.3.3 Time Constant Register B (TCORB)

TCORB is an 8-bit readable/writable register. TCORB_0 and TCORB_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction. TCORB is continually compared with the value in TCNT. When a match is detected, the corresponding CMFB flag in TCSR is set to 1. Note however that comparison is disabled during the T2 state of a TCORB write cycle. The timer output from the TMO pin can be freely controlled by this compare match signal (compare match B) and the settings of bits OS3 and OS2 in TCSR. TCORB is initialized to H'FF.

	TCORB_0								TCORB_1							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Bit Name																
Initial Value	1	1	1	1	1	1	1	1	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	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12.3.4 Timer Control Register (TCR)

TCR selects the TCNT clock source and the condition for clearing TCNT, and enables/disables interrupt requests.

Bit	7	6	5	4	3	2	1	0
Bit Name	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	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	Bit Name	Initial Value	R/W	Description
7	CMIEB	0	R/W	Compare Match Interrupt Enable B Selects whether CMFB interrupt requests (CMIB) are enabled or disabled when the CMFB flag in TCSR is set to 1. 0: CMFB interrupt requests (CMIB) are disabled 1: CMFB interrupt requests (CMIB) are enabled

Bit	Bit Name	Initial Value	R/W	Description
6	CMIEA	0	R/W	Compare Match Interrupt Enable A Selects whether CMFA interrupt requests (CMIA) are enabled or disabled when the CMFA flag in TCSR is set to 1. 0: CMFA interrupt requests (CMIA) are disabled 1: CMFA interrupt requests (CMIA) are enabled
5	OVIE	0	R/W	Timer Overflow Interrupt Enable Selects whether OVF interrupt requests (OVI) are enabled or disabled when the OVF flag in TCSR is set to 1. 0: OVF interrupt requests (OVI) are disabled 1: OVF interrupt requests (OVI) are enabled
4	CCLR1	0	R/W	Counter Clear 1 and 0*
3	CCLR0	0	R/W	These bits select the method by which TCNT is cleared. 00: Clearing is disabled 01: Cleared by compare match A 10: Cleared by compare match B 11: Cleared at rising edge (TMRIS in TCCR is cleared to 0) of the external reset input or when the external reset input is high (TMRIS in TCCR is set to 1)
2	CKS2	0	R/W	Clock Select 2 to 0*
1	CKS1	0	R/W	These bits select the clock input to TCNT and count condition. See table 12.2.
0	CKS0	0	R/W	

Note: * To use an external reset or external clock, the DDR and ICR bits in the corresponding pin should be set to 0 and 1, respectively. For details, see section 9, I/O Ports.

12.3.5 Timer Counter Control Register (TCCR)

TCCR selects the TCNT internal clock source and controls external reset input.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	TMRIS	—	ICKS1	ICKS0
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	Bit Name	Initial Value	R/W	Description
7 to 4	—	0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
3	TMRIS	0	R/W	Timer Reset Input Select Selects an external reset input when the CCLR1 and CCLR0 bits in TCR are B'11. 0: Cleared at rising edge of the external reset 1: Cleared when the external reset is high
2	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0
1	ICKS1	0	R/W	Internal Clock Select 1 and 0
0	ICKS0	0	R/W	These bits in combination with bits CKS2 to CKS0 in TCR select the internal clock. See table 12.2.

Table 12.2 Clock Input to TCNT and Count Condition

	TCR			TCCR		
Channel	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0	Description
TMR_0	0	0	0	—	—	Clock input prohibited.
	0	0	1	0	0	Uses internal clock. Counts at rising edge of Pϕ/8.
				0	1	Uses internal clock. Counts at rising edge of Pϕ/2.
				1	0	Uses internal clock. Counts at falling edge of Pϕ/8.
				1	1	Uses internal clock. Counts at falling edge of Pϕ/2.
	0	1	0	0	0	Uses internal clock. Counts at rising edge of Pϕ/64.
				0	1	Uses internal clock. Counts at rising edge of Pϕ/32.
				1	0	Uses internal clock. Counts at falling edge of Pϕ/64.
				1	1	Uses internal clock. Counts at falling edge of Pϕ/32.
	0	1	1	0	0	Uses internal clock. Counts at rising edge of Pϕ/8192.
				0	1	Uses internal clock. Counts at rising edge of Pϕ/1024.
				1	0	Uses internal clock. Counts at falling edge of Pϕ/8192.
				1	1	Uses internal clock. Counts at falling edge of Pϕ/1024.
	1	0	0	—	—	Counts at TCNT_1 overflow signal*[1].
TMR_1	0	0	0	—	—	Clock input prohibited.
	0	0	1	0	0	Uses internal clock. Counts at rising edge of Pϕ/8.
				0	1	Uses internal clock. Counts at rising edge of Pϕ/2.
				1	0	Uses internal clock. Counts at falling edge of Pϕ/8.
				1	1	Uses internal clock. Counts at falling edge of Pϕ/2.
	0	1	0	0	0	Uses internal clock. Counts at rising edge of Pϕ/64.
				0	1	Uses internal clock. Counts at rising edge of Pϕ/32.
				1	0	Uses internal clock. Counts at falling edge of Pϕ/64.
				1	1	Uses internal clock. Counts at falling edge of Pϕ/32.
	0	1	1	0	0	Uses internal clock. Counts at rising edge of Pϕ/8192.
				0	1	Uses internal clock. Counts at rising edge of Pϕ/1024.
				1	0	Uses internal clock. Counts at falling edge of Pϕ/8192.
				1	1	Uses internal clock. Counts at falling edge of Pϕ/1024.
	1	0	0	—	—	Counts at TCNT_0 compare match A*[1].

	TCR			TCCR		
Channel	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0	Description
All	1	0	1	—	—	Uses external clock. Counts at rising edge*[2].
	1	1	0	—	—	Uses external clock. Counts at falling edge*[2].
	1	1	1	—	—	Uses external clock. Counts at both rising and falling edges*[2].

Notes: 1. If the clock input of TMR_0 is the TCNT_1 overflow signal and that of TMR_1 is the TCNT_0 compare match signal, no incrementing clock is generated. Do not use this setting.

2. To use the external clock, the DDR and ICR bits in the corresponding pin should be set to 0 and 1, respectively. For details, see section 9, I/O Ports.

12.3.6 Timer Control/Status Register (TCSR)

TCSR displays status flags, and controls compare match output.

• TCSR_0

Bit	7	6	5	4	3	2	1	0
Bit Name	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0
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

• TCSR_1

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

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

- TCSR_0

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*1	Compare Match Flag B [Setting condition] • When TCNT matches TCORB [Clearing conditions] • When writing 0 after reading CMFB = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When the DTC is activated by a CMIB interrupt while the DISEL bit in MRB of the DTC is 0
6	CMFA	0	R/(W)*1	Compare Match Flag A [Setting condition] • When TCNT matches TCORA [Clearing conditions] • When writing 0 after reading CMFA = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When the DTC is activated by a CMIA interrupt while the DISEL bit in MRB in the DTC is 0
5	OVF	0	R/(W)*1	Timer Overflow Flag [Setting condition] • When TCNT overflows from H'FF to H'00 [Clearing condition] • When writing 0 after reading OVF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial Value	R/W	Description
4	ADTE	0	R/W	A/D Trigger Enable Selects enabling or disabling of A/D converter start requests by compare match A. 0: A/D converter start requests by compare match A are disabled 1: A/D converter start requests by compare match A are enabled
3	OS3	0	R/W	Output Select 3 and 2*[2]
2	OS2	0	R/W	These bits select a method of TMO pin output when compare match B of TCORB and TCNT occurs. 00: No change when compare match B occurs 01: 0 is output when compare match B occurs 10: 1 is output when compare match B occurs 11: Output is inverted when compare match B occurs (toggle output)
1	OS1	0	R/W	Output Select 1 and 0*[2]
0	OS0	0	R/W	These bits select a method of TMO pin output when compare match A of TCORA and TCNT occurs. 00: No change when compare match A occurs 01: 0 is output when compare match A occurs 10: 1 is output when compare match A occurs 11: Output is inverted when compare match A occurs (toggle output)

Notes: 1. Only 0 can be written to bits 7 to 5, to clear these flags.
2. Timer output is disabled when bits OS3 to OS0 are all 0. Timer output is 0 until the first compare match occurs after resetting.

- TCSR_1

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*1	Compare Match Flag B [Setting condition] • When TCNT matches TCORB [Clearing conditions] • When writing 0 after reading CMFB = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When the DTC is activated by a CMIB interrupt while the DISEL bit in MRB of the DTC is 0
6	CMFA	0	R/(W)*1	Compare Match Flag A [Setting condition] • When TCNT matches TCORA [Clearing conditions] • When writing 0 after reading CMFA = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When the DTC is activated by a CMIA interrupt while the DISEL bit in MRB of the DTC is 0
5	OVF	0	R/(W)*1	Timer Overflow Flag [Setting condition] • When TCNT overflows from H'FF to H'00 [Clearing condition] • Cleared by reading OVF when OVF = 1, then writing 0 to OVF (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
4	—	1	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
3	OS3	0	R/W	Output Select 3 and 2*[2]
2	OS2	0	R/W	These bits select a method of TMO pin output when compare match B of TCORB and TCNT occurs. 00: No change when compare match B occurs 01: 0 is output when compare match B occurs 10: 1 is output when compare match B occurs 11: Output is inverted when compare match B occurs (toggle output)
1	OS1	0	R/W	Output Select 1 and 0*[2]
0	OS0	0	R/W	These bits select a method of TMO pin output when compare match A of TCORA and TCNT occurs. 00: No change when compare match A occurs 01: 0 is output when compare match A occurs 10: 1 is output when compare match A occurs 11: Output is inverted when compare match A occurs (toggle output)

Notes: 1. Only 0 can be written to bits 7 to 5, to clear these flags.
2. Timer output is disabled when bits OS3 to OS0 are all 0. Timer output is 0 until the first compare match occurs after resetting.

12.4 Operation

12.4.1 Pulse Output

Figure 12.3 shows an example of the 8-bit timer being used to generate a pulse output with a desired duty cycle. The control bits are set as follows:

1. In TCR, clear bit CCLR1 to 0 and set bit CCLR0 to 1 so that TCNT is cleared at a TCORA compare match.
2. In TCSR, set bits OS3 to OS0 to B'0110, causing the output to change to 1 at a TCORA compare match and to 0 at a TCORB compare match.

With these settings, the 8-bit timer provides pulses output at a cycle determined by TCORA with a pulse width determined by TCORB. No software intervention is required. The output level of the 8-bit timer holds 0 until the first compare match occurs after a reset.

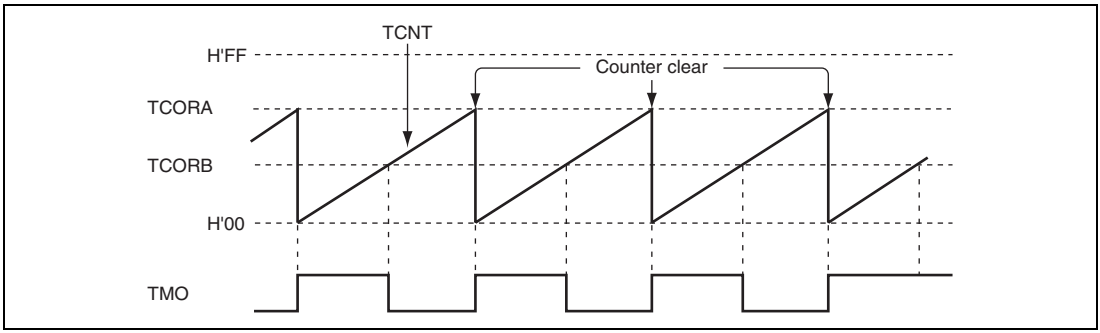


Figure 12.3 Example of Pulse Output

12.4.2 Reset Input

Figure 12.4 shows an example of the 8-bit timer being used to generate a pulse which is output after a desired delay time from a TMRI input. The control bits are set as follows:

1. Set both bits CCLR1 and CCLR0 in TCR to 1 and set the TMRIS bit in TCCR to 1 so that TCNT is cleared at the high level input of the TMRI signal.
2. In TCSR, set bits OS3 to OS0 to B'0110, causing the output to change to 1 at a TCORA compare match and to 0 at a TCORB compare match.

With these settings, the 8-bit timer provides pulses output at a desired delay time from a TMRI input determined by TCORA and with a pulse width determined by TCORB and TCORA.

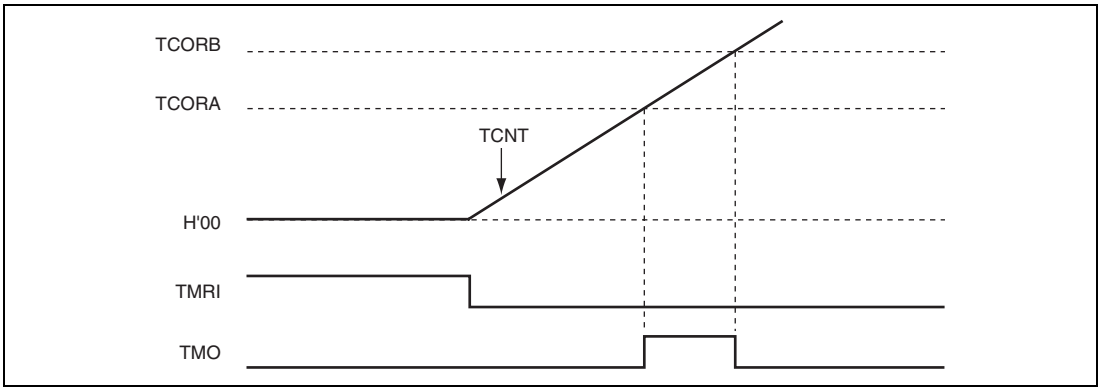


Figure 12.4 Example of Reset Input

12.5 Operation Timing

12.5.1 TCNT Count Timing

Figure 12.5 shows the TCNT count timing for internal clock input. Figure 12.6 shows the TCNT count timing for external clock input. Note that the external clock pulse width must be at least 1.5 states for incrementation at a single edge, and at least 2.5 states for incrementation at both edges. The counter will not increment correctly if the pulse width is less than these values.

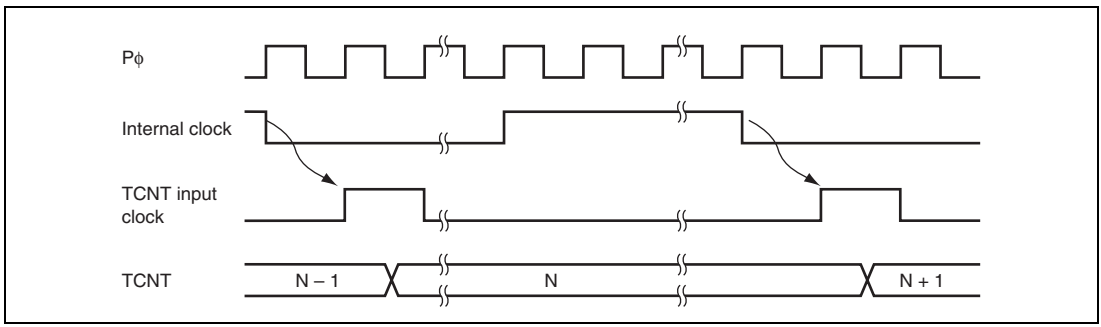


Figure 12.5 Count Timing for Internal Clock Input (Falling Edge)

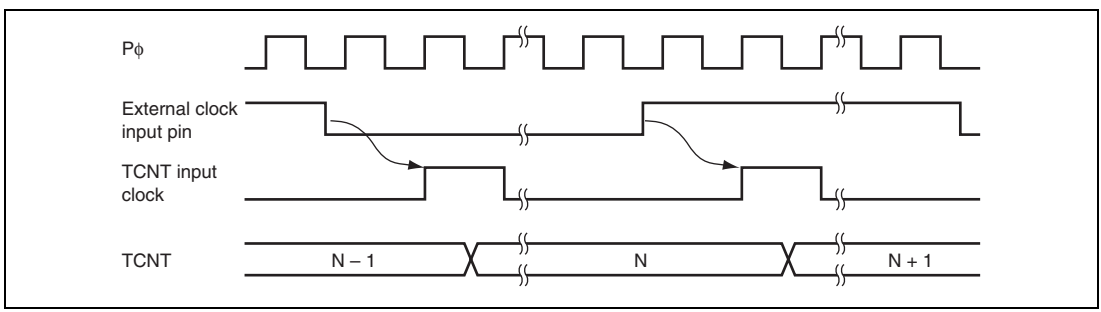


Figure 12.6 Count Timing for External Clock Input (Both Edges)

12.5.2 Timing of CMFA and CMFB Setting at Compare Match

The CMFA and CMFB flags in TCSR are set to 1 by a compare match signal generated when the TCOR and TCNT values match. The compare match signal is generated at the last state in which the match is true, just before the timer counter is updated. Therefore, when the TCOR and TCNT values match, the compare match signal is not generated until the next TCNT clock input. Figure 12.7 shows this timing.

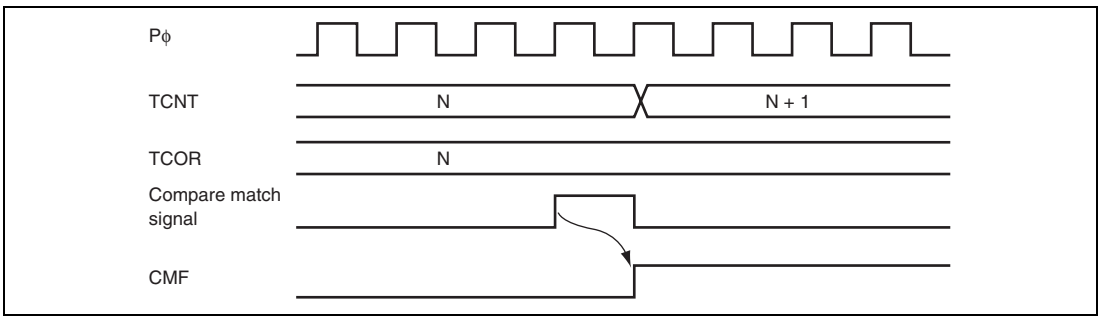


Figure 12.7 Timing of CMF Setting at Compare Match

12.5.3 Timing of Timer Output at Compare Match

When a compare match signal is generated, the timer output changes as specified by bits OS3 to OS0 in TCSR. Figure 12.8 shows the timing when the timer output is toggled by the compare match A signal.

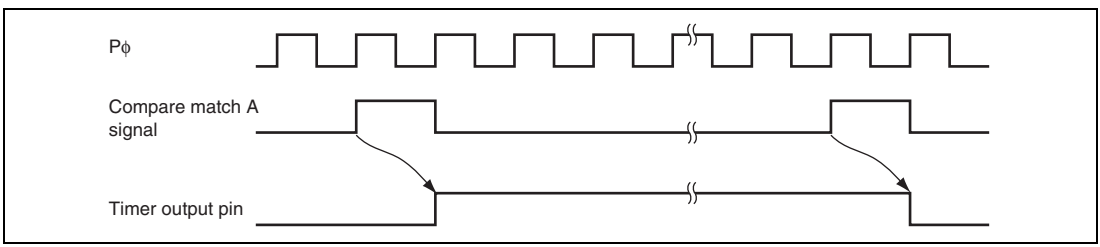


Figure 12.8 Timing of Toggled Timer Output at Compare Match A

12.5.4 Timing of Counter Clear by Compare Match

TCNT is cleared when compare match A or B occurs, depending on the settings of bits CCLR1 and CCLR0 in TCR. Figure 12.9 shows the timing of this operation.

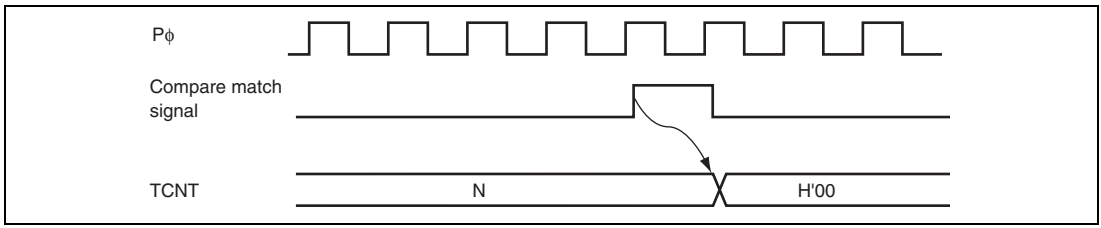


Figure 12.9 Timing of Counter Clear by Compare Match

12.5.5 Timing of TCNT External Reset

TCNT is cleared at the rising edge or high level of an external reset input, depending on the settings of bits CCLR1 and CCLR0 in TCR. The clear pulse width must be at least 2 states. Figures 12.10 and 12.11 show the timing of this operation.

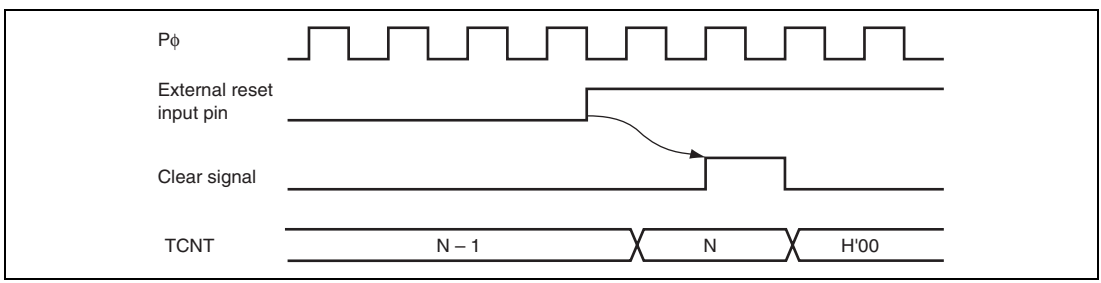


Figure 12.10 Timing of Clearance by External Reset (Rising Edge)

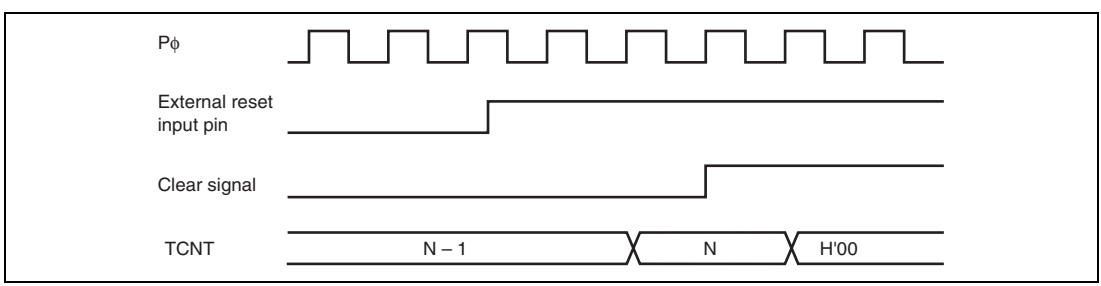


Figure 12.11 Timing of Clearance by External Reset (High Level)

12.5.6 Timing of Overflow Flag (OVF) Setting

The OVF bit in TCSR is set to 1 when TCNT overflows (changes from H'FF to H'00). Figure 12.12 shows the timing of this operation.

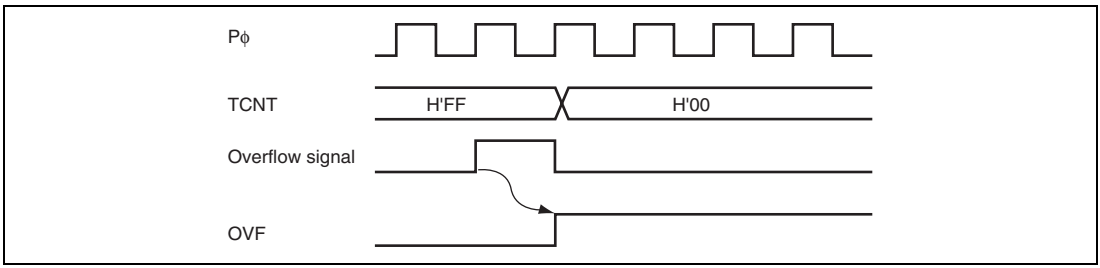


Figure 12.12 Timing of OVF Setting

12.6 Operation with Cascaded Connection

If bits CKS2 to CKS0 in either TCR_0 or TCR_1 are set to B'100, the 8-bit timers of the two channels are cascaded. With this configuration, a single 16-bit timer could be used (16-bit counter mode) or compare matches of the 8-bit channel 0 could be counted by the timer of channel 1 (compare match count mode).

12.6.1 16-Bit Counter Mode

When bits CKS2 to CKS0 in TCR_0 are set to B'100, the timer functions as a single 16-bit timer with channel 0 occupying the upper 8 bits and channel 1 occupying the lower 8 bits.

(1) Setting of Compare Match Flags

- The CMF flag in TCSR_0 is set to 1 when a 16-bit compare match event occurs.
- The CMF flag in TCSR_1 is set to 1 when a lower 8-bit compare match event occurs.

(2) Counter Clear Specification

- If the CCLR1 and CCLR0 bits in TCR_0 have been set for counter clear at compare match, the 16-bit counter (TCNT_0 and TCNT_1 together) is cleared when a 16-bit compare match event occurs. The 16-bit counter (TCNT0 and TCNT1 together) is cleared even if counter clear by the TMRI0 pin has been set.
- The settings of the CCLR1 and CCLR0 bits in TCR_1 are ignored. The lower 8 bits cannot be cleared independently.

(3) Pin Output

- Control of output from the TMO0 pin by bits OS3 to OS0 in TCSR_0 is in accordance with the 16-bit compare match conditions.
- Control of output from the TMO1 pin by bits OS3 to OS0 in TCSR_1 is in accordance with the lower 8-bit compare match conditions.

12.6.2 Compare Match Count Mode

When bits CKS2 to CKS0 in TCR_1 are set to B'100, TCNT_1 counts compare match A for channel 0. Channels 0 and 1 are controlled independently. Conditions such as setting of the CMF flag, generation of interrupts, output from the TMO pin, and counter clear are in accordance with the settings for each channel.

12.7 Interrupt Sources

12.7.1 Interrupt Sources and DTC Activation

There are three interrupt sources for the 8-bit timer (TMR_0 or TMR_1): CMIA, CMIB, and OVI. Their interrupt sources and priorities are shown in table 12.3. Each interrupt source is enabled or disabled by the corresponding interrupt enable bit in TCR or TCSR, and independent interrupt requests are sent for each to the interrupt controller. It is also possible to activate the DTC by means of CMIA and CMIB interrupts.

Table 12.3 8-Bit Timer (TMR_0 or TMR_1) Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority
CMIA0	TCORA_0 compare match	CMFA	Possible (VNUM = 2'b00)	High
CMIB0	TCORB_0 compare match	CMFB	Possible (VNUM = 2'b01)	
OVI0	TCNT_0 overflow	OVF	Not possible	Low
CMIA1	TCORA_1 compare match	CMFA	Possible (VNUM = 2'b10)	High
CMIB1	TCORB_1 compare match	CMFB	Possible (VNUM = 2'b11)	
OVI1	TCNT_1 overflow	OVF	Not possible	Low

Note: VNUM is an internal signal.

12.7.2 A/D Converter Activation

The A/D converter can be activated only by TMR_0 compare match A.

If the ADTE bit in TCSR_0 is set to 1 when the CMFA flag in TCSR_0 is set to 1 by the occurrence of TMR_0 compare match A, a request to start A/D conversion is sent to the A/D converter. If the 8-bit timer conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

12.8 Usage Notes

12.8.1 Notes on Setting Cycle

If the compare match is selected for counter clear, TCNT is cleared at the last state in the cycle in which the values of TCNT and TCOR match. TCNT updates the counter value at this last state. Therefore, the counter frequency is obtained by the following formula.

$$f = \phi / (N + 1)$$

f: Counter frequency
ϕ: Operating frequency
N: TCOR value

12.8.2 Conflict between TCNT Write and Clear

If a counter clear signal is generated during the T_2 state of a TCNT write cycle, the clear takes priority and the write is not performed as shown in figure 12.13.

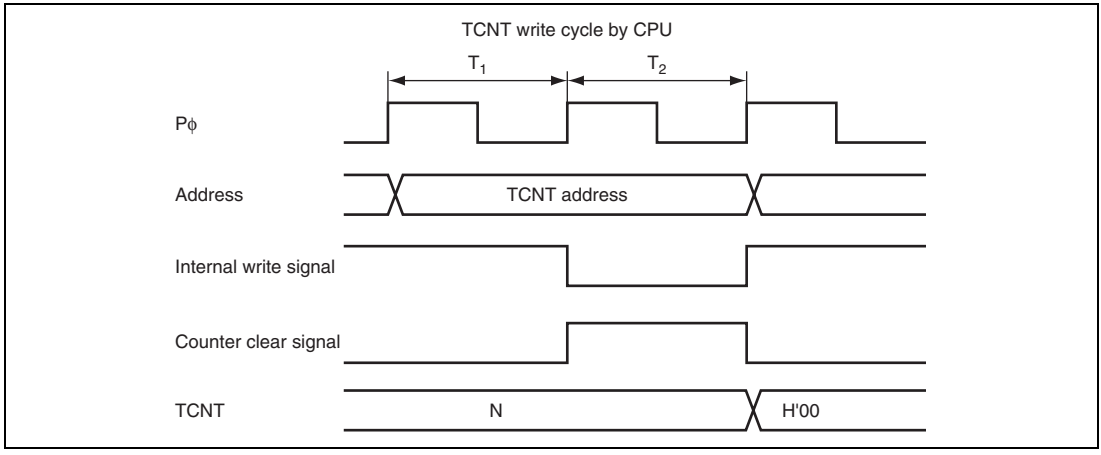


Figure 12.13 Conflict between TCNT Write and Clear

12.8.3 Conflict between TCNT Write and Increment

If a TCNT input clock pulse is generated during the T_2 state of a TCNT write cycle, the write takes priority and the counter is not incremented as shown in figure 12.14.

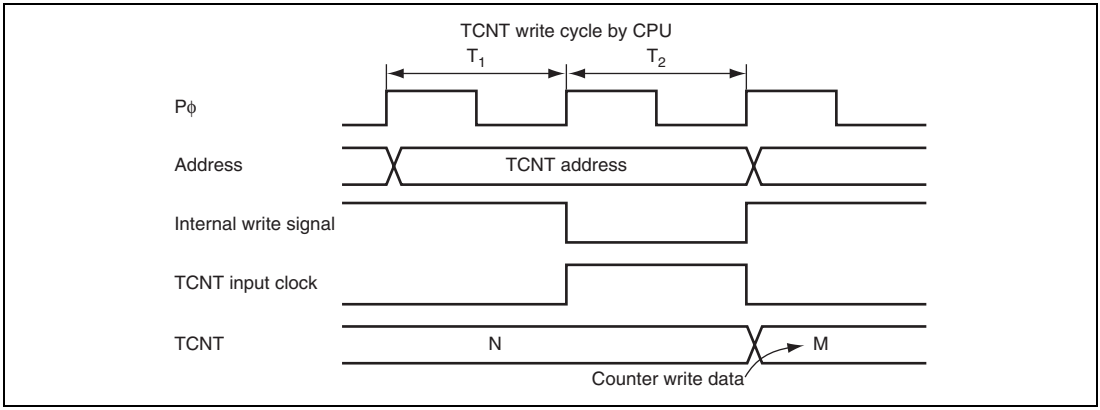


Figure 12.14 Conflict between TCNT Write and Increment

12.8.4 Conflict between TCOR Write and Compare Match

If a compare match event occurs during the T_2 state of a TCOR write cycle, the TCOR write takes priority and the compare match signal is inhibited as shown in figure 12.15.

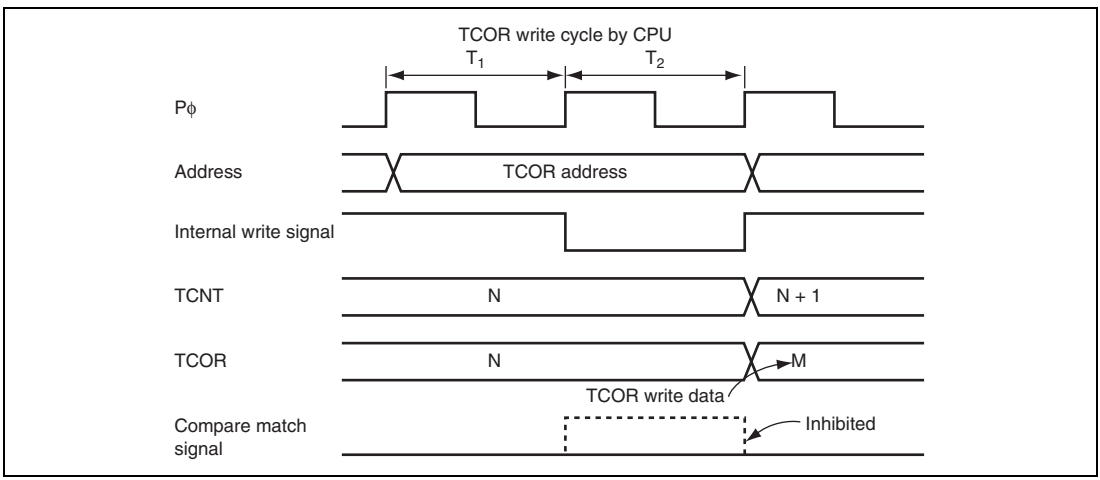


Figure 12.15 Conflict between TCOR Write and Compare Match

12.8.5 Conflict between Compare Matches A and B

If compare match events A and B occur at the same time, the 8-bit timer operates in accordance with the priorities for the output statuses set for compare match A and compare match B, as shown in table 12.4.

Table 12.4 Timer Output Priorities

Output Setting	Priority
Toggle output	High
1-output	
0-output	
No change	Low

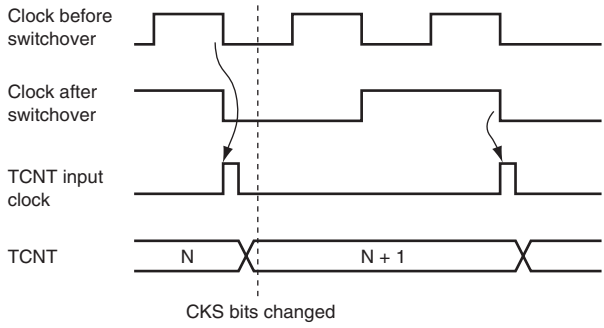
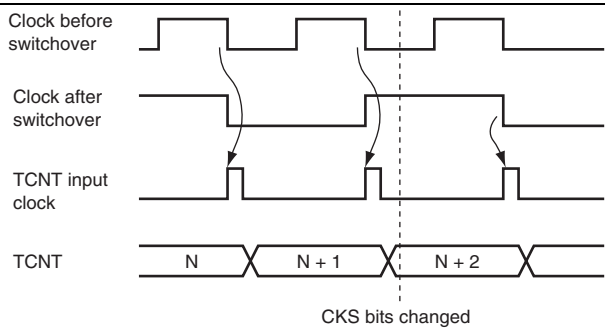
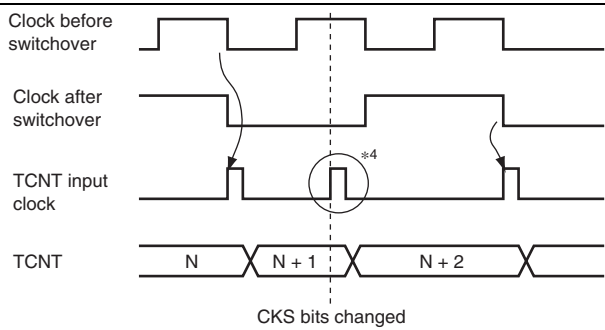
12.8.6 Switching of Internal Clocks and TCNT Operation

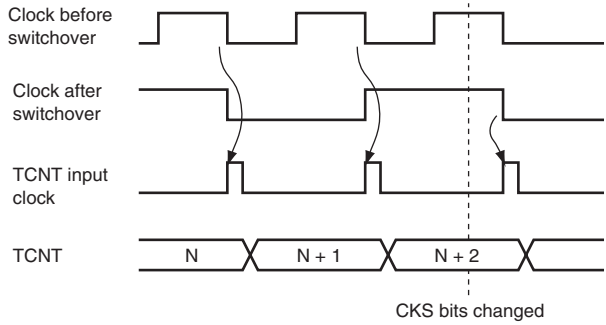
TCNT may be incremented erroneously depending on when the internal clock is switched. Table 12.5 shows the relationship between the timing at which the internal clock is switched (by writing to bits CKS1 and CKS0) and the TCNT operation.

When the TCNT clock is generated from an internal clock, the rising or falling edge of the internal clock pulse are always monitored. Table 12.5 assumes that the falling edge is selected. If the signal levels of the clocks before and after switching change from high to low as shown in item 3, the change is considered as the falling edge. Therefore, a TCNT clock pulse is generated and TCNT is incremented. This is similar to when the rising edge is selected.

The erroneous incrementation of TCNT can also happen when switching between rising and falling edges of the internal clock, and when switching between internal and external clocks.

Table 12.5 Switching of Internal Clock and TCNT Operation

No.	Timing to Change CKS1 and CKS0 Bits	TCNT Clock Operation
1	Switching from low to low*[1]	
2	Switching from low to high*[2]	
3	Switching from high to low*[3]	

No.	Timing to Change CKS1 and CKS0 Bits	TCNT Clock Operation
4	Switching from high to high	

Notes: 1. Includes switching from low to stop, and from stop to low.
2. Includes switching from stop to high.
3. Includes switching from high to stop.
4. Generated because the change of the signal levels is considered as a falling edge; TCNT is incremented.

12.8.7 Mode Setting with Cascaded Connection

If 16-bit counter mode and compare match count mode are specified at the same time, input clocks for TCNT_0 and TCNT_1 are not generated, and the counter stops. Do not specify 16-bit counter mode and compare match count mode simultaneously.

12.8.8 Module Stop Function Setting

Operation of the TMR can be disabled or enabled using the module stop control register. The initial setting is for operation of the TMR to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 20, Power-Down Modes.

12.8.9 Interrupts in Module Stop State

If the module stop state is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DTC activation source. Interrupts should therefore be disabled before entering the module stop state.

Section 13 Watchdog Timer (WDT)

The watchdog timer (WDT) is an 8-bit timer that outputs an overflow signal ($\overline{\text{WDTOVF}}$) if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow. At the same time, the WDT can also generate an internal reset signal.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

Figure 13.1 shows a block diagram of the WDT.

13.1 Features

- Selectable from eight counter input clocks
- Switchable between watchdog timer mode and interval timer mode
  - — In watchdog timer mode

    If the counter overflows, the WDT outputs $\overline{\text{WDTOVF}}$. It is possible to select whether or not the entire LSI is reset at the same time.
  - — In interval timer mode

    If the counter overflows, the WDT generates an interval timer interrupt (WOVI).

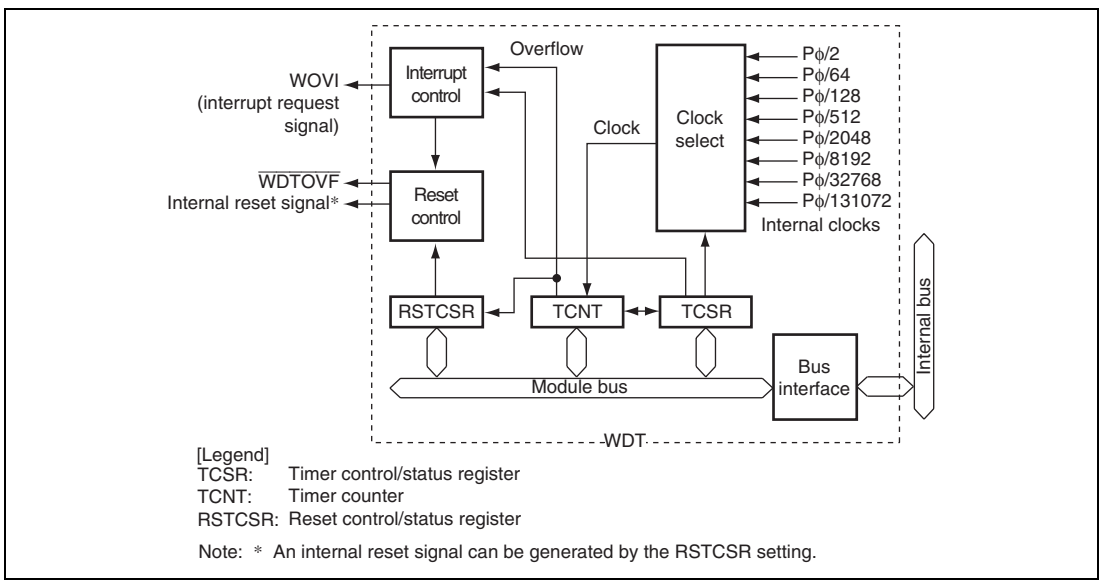


Figure 13.1 Block Diagram of WDT

13.2 Input/Output Pin

Table 13.1 shows the WDT pin configuration.

Table 13.1 Pin Configuration

Name	Symbol	I/O	Function
Watchdog timer overflow	$\overline{\text{WDTOVF}}$	Output	Outputs a counter overflow signal in watchdog timer mode

13.3 Register Descriptions

The WDT has the following three registers. To prevent accidental overwriting, TCSR, TCNT, and RSTCSR have to be written to in a method different from normal registers. For details, see section 13.6.1, Notes on Register Access.

- Timer counter (TCNT)
- Timer control/status register (TCSR)
- Reset control/status register (RSTCSR)

13.3.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 when the TME bit in TCSR is cleared to 0.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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

13.3.2 Timer Control/Status Register (TCSR)

TCSR selects the clock source to be input to TCNT, and the timer mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	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

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

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*	Overflow Flag Indicates that TCNT has overflowed in interval timer mode. Only 0 can be written to this bit, to clear the flag. [Setting condition] When TCNT overflows in interval timer mode (changes from H'FF to H'00) • When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset. [Clearing condition] • Cleared by reading TCSR when OVF = 1, then writing 0 to OVF (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
6	WT/$\overline{IT}$	0	R/W	Timer Mode Select Selects whether the WDT is used as a watchdog timer or interval timer. 0: Interval timer mode When TCNT overflows, an interval timer interrupt (WOVI) is requested. 1: Watchdog timer mode When TCNT overflows, the $\overline{WDTOVF}$ signal is output.

Bit	Bit Name	Initial Value	R/W	Description
5	TME	0	R/W	Timer Enable When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00.
4, 3	—	All 1	R	Reserved These are read-only bits and cannot be modified.
2	CKS2	0	R/W	Clock Select 2 to 0 Select the clock source to be input to TCNT. The overflow cycle for Pϕ = 20 MHz is indicated in parentheses. 000: Clock Pϕ/2 (cycle: 25.6 μs) 001: Clock Pϕ/64 (cycle: 819.2 μs) 010: Clock Pϕ/128 (cycle: 1.6 ms) 011: Clock Pϕ/512 (cycle: 6.6 ms) 100: Clock Pϕ/2048 (cycle: 26.2 ms) 101: Clock Pϕ/8192 (cycle: 104.9 ms) 110: Clock Pϕ/32768 (cycle: 419.4 ms) 111: Clock Pϕ/131072 (cycle: 1.68 s)
1	CKS1	0	R/W	
0	CKS0	0	R/W	

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

13.3.3 Reset Control/Status Register (RSTCSR)

RSTCSR controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal. RSTCSR is initialized to H'1F by a reset signal from the $\overline{\text{RES}}$ pin, but not by the WDT internal reset signal caused by WDT overflows.

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

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

Bit	Bit Name	Initial Value	R/W	Description
7	WOVF	0	R/(W)*	Watchdog Timer Overflow Flag This bit is set when TCNT overflows in watchdog timer mode. This bit cannot be set in interval timer mode, and only 0 can be written. [Setting condition] • When TCNT overflows (changed from H'FF to H'00) in watchdog timer mode [Clearing condition] • Reading RSTCSR when WOVF = 1, and then writing 0 to WOVF
6	RSTE	0	R/W	Reset Enable Specifies whether or not this LSI is internally reset if TCNT overflows during watchdog timer operation. 0: LSI is not reset even if TCNT overflows (Though this LSI is not reset, TCNT and TCSR in WDT are reset) 1: LSI is reset if TCNT overflows
5	—	0	R/W	Reserved Although this bit is readable/writable, reading from or writing to this bit does not affect operation.
4 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

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

13.4 Operation

13.4.1 Watchdog Timer Mode

To use the WDT in watchdog timer mode, set both the WT/$\overline{IT}$ and TME bits in TCSR to 1.

During watchdog timer operation, if TCNT overflows without being rewritten because of a system crash or other error, the $\overline{WDTOVF}$ signal is output. This ensures that TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally H'00 is written) before overflow occurs. This $\overline{WDTOVF}$ signal can be used to reset the LSI internally in watchdog timer mode.

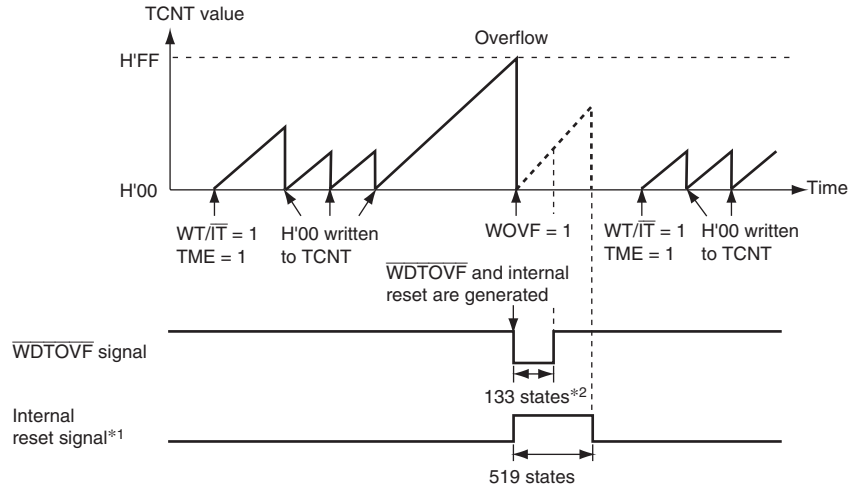
If TCNT overflows when the RSTE bit in RSTCSR is set to 1, a signal that resets this LSI internally is generated at the same time as the $\overline{WDTOVF}$ signal. If a reset caused by a signal input to the $\overline{RES}$ pin occurs at the same time as a reset caused by a WDT overflow, the $\overline{RES}$ pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.

The $\overline{WDTOVF}$ signal is output for 133 cycles of Pϕ when RSTE = 1 in RSTCSR, and for 130 cycles of Pϕ when RSTE = 0 in RSTCSR. The internal reset signal is output for 519 cycles of Pϕ.

When RSTE = 1, an internal reset signal is generated. Since the system clock control register (SCKCR) is initialized, the multiplication ratio of Pϕ becomes the initial value.

When RSTE = 0, an internal reset signal is not generated. Neither SCKCR nor the multiplication ratio of Pϕ is changed.

When TCNT overflows in watchdog timer mode, the WOVF bit in RSTCSR is set to 1. If TCNT overflows when the RSTE bit in RSTCSR is set to 1, an internal reset signal is generated for the entire LSI.



Notes: 1. If TCNT overflows when the RSTE bit is set to 1, an internal reset signal is generated.
2. 130 states when the RSTE bit is cleared to 0.

Figure 13.2 Operation in Watchdog Timer Mode

13.4.2 Interval Timer Mode

To use the WDT as an interval timer, set the WT/$\overline{IT}$ bit to 0 and the TME bit to 1 in TCSR.

When the WDT is used as an interval timer, an interval timer interrupt (WOVI) is generated each time the TCNT overflows. Therefore, an interrupt can be generated at intervals.

When the TCNT overflows in interval timer mode, an interval timer interrupt (WOVI) is requested at the same time the OVF bit in the TCSR is set to 1.

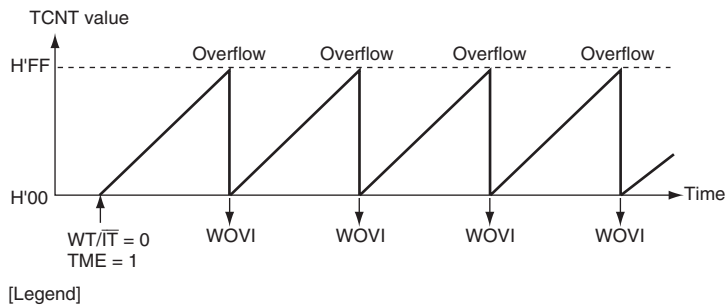


Figure 13.3 Operation in Interval Timer Mode

13.5 Interrupt Source

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR. The OVF flag must be cleared to 0 in the interrupt handling routine.

Table 13.2 WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag	DTC Activation
WOVI	TCNT overflow	OVF	Impossible

13.6 Usage Notes

13.6.1 Notes on Register Access

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

(1) Writing to TCNT, TCSR, and RSTCSR

TCNT and TCSR must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

For writing, TCNT and TCSR are assigned to the same address. Accordingly, perform data transfer as shown in figure 13.4. The transfer instruction writes the lower byte data to TCNT or TCSR.

To write to RSTCSR, execute a word transfer instruction for address H'FFA6. A byte transfer instruction cannot be used to write to RSTCSR.

The method of writing 0 to the WOVF bit in RSTCSR differs from that of writing to the RSTE bit in RSTCSR. Perform data transfer as shown in figure 13.4.

At data transfer, the transfer instruction clears the WOVF bit to 0, but has no effect on the RSTE bit. To write to the RSTE bit, perform data transfer as shown in figure 13.4. In this case, the transfer instruction writes the value in bit 6 of the lower byte to the RSTE bit, but has no effect on the WOVF bit.

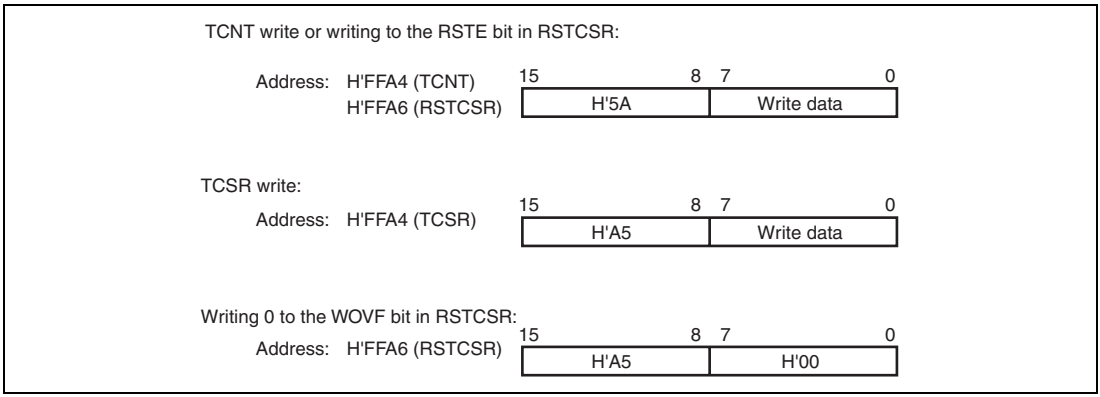


Figure 13.4 Writing to TCNT, TCSR, and RSTCSR

(2) Reading from TCNT, TCSR, and RSTCSR

These registers can be read from in the same way as other registers. For reading, TCSR is assigned to address H'FFA4, TCNT to address H'FFA5, and RSTCSR to address H'FFA7.

13.6.2 Conflict between Timer Counter (TCNT) Write and Increment

If a TCNT clock pulse is generated during the T2 cycle of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 13.5 shows this operation.

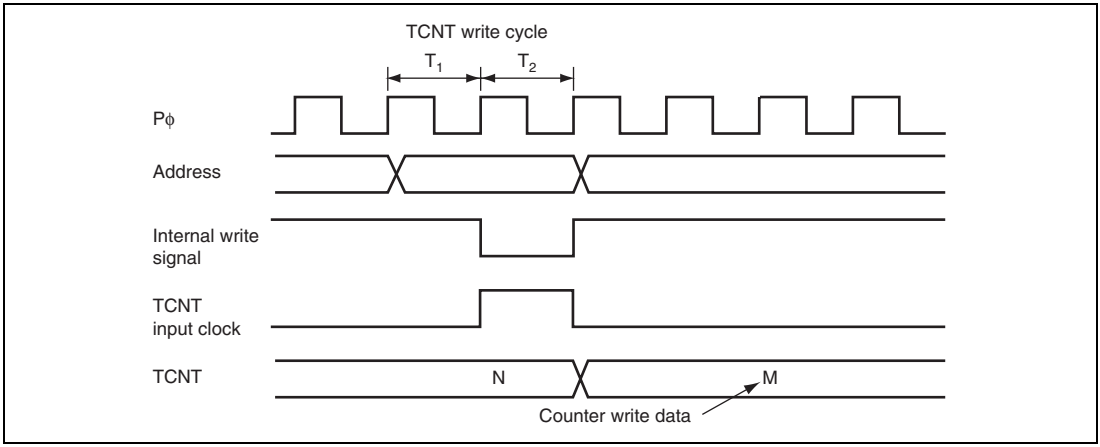


Figure 13.5 Conflict between TCNT Write and Increment

13.6.3 Changing Values of Bits CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before the values of bits CKS2 to CKS0 are changed.

13.6.4 Switching between Watchdog Timer Mode and Interval Timer Mode

If the timer mode is switched from watchdog timer mode to interval timer mode while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before switching the timer mode.

13.6.5 Internal Reset in Watchdog Timer Mode

This LSI is not reset internally if TCNT overflows while the RSTE bit is cleared to 0 during watchdog timer mode operation, but TCNT and TCSR of the WDT are reset.

TCNT, TCSR, and RSTCR cannot be written to while the $\overline{\text{WDTOVF}}$ signal is low. Also note that a read of the WOVF flag is not recognized during this period. To clear the WOVF flag, therefore, read TCSR after the $\overline{\text{WDTOVF}}$ signal goes high, then write 0 to the WOVF flag.

13.6.6 System Reset by $\overline{\text{WDTOVF}}$ Signal

If the $\overline{\text{WDTOVF}}$ signal is input to the $\overline{\text{RES}}$ pin, this LSI will not be initialized correctly. Make sure that the $\overline{\text{WDTOVF}}$ signal is not input logically to the $\overline{\text{RES}}$ pin. To reset the entire system by means of the $\overline{\text{WDTOVF}}$ signal, use a circuit like that shown in figure 13.6.

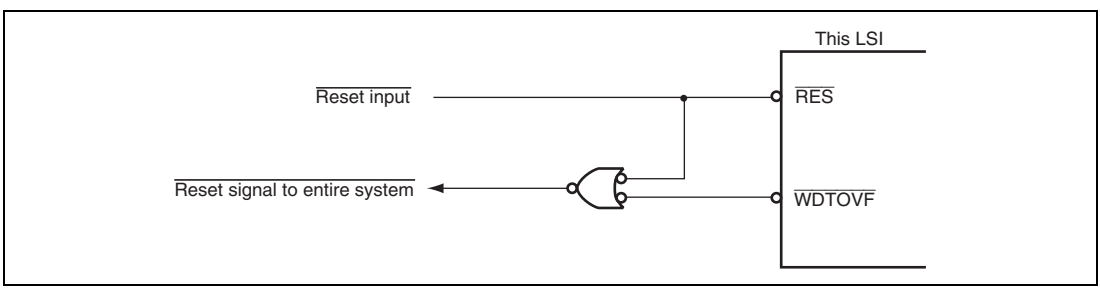


Figure 13.6 Circuit for System Reset by $\overline{\text{WDTOVF}}$ Signal (Example)

13.6.7 Transition to Watchdog Timer Mode or Software Standby Mode

When the WDT operates in watchdog timer mode, a transition to software standby mode is not made even when the SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1. Instead, a transition to sleep mode is made.

To transit to software standby mode, the SLEEP instruction must be executed after halting the WDT (clearing the TME bit to 0).

When the WDT operates in interval timer mode, a transition to software standby mode is made through execution of the SLEEP instruction when the SSBY bit in SBYCR is set to 1.

Section 14 Serial Communication Interface (SCI)

This LSI has four independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clocked synchronous serial communication. Asynchronous serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface supporting ISO/IEC 7816-3 (Identification Card) as an extended asynchronous communication mode. Figure 14.1 shows a block diagram of the SCI.

14.1 Features

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

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected

  The external clock can be selected as a transfer clock source (except for the smart card interface).
- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode 7-bit data)
- Four interrupt sources

  The interrupt sources are transmit-end, transmit-data-empty, receive-data-full, and receive error. The transmit-data-empty and receive-data-full interrupt sources can activate the DMAC or DTC.
- Module stop state specifiable

Asynchronous Mode:

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

- Average transfer rate generator (SCI_2 only)
  10.667-MHz operation: 460.606 kbps or 115.152 kbps can be selected
  16-MHz operation: 720 kbps, 460.784 kbps, or 115.196 kbps can be selected
  32-MHz operation: 720 kbps

Clocked Synchronous Mode:

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

Smart Card Interface:

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on receiving an error signal during transmission
- Both direct convention and inverse convention are supported

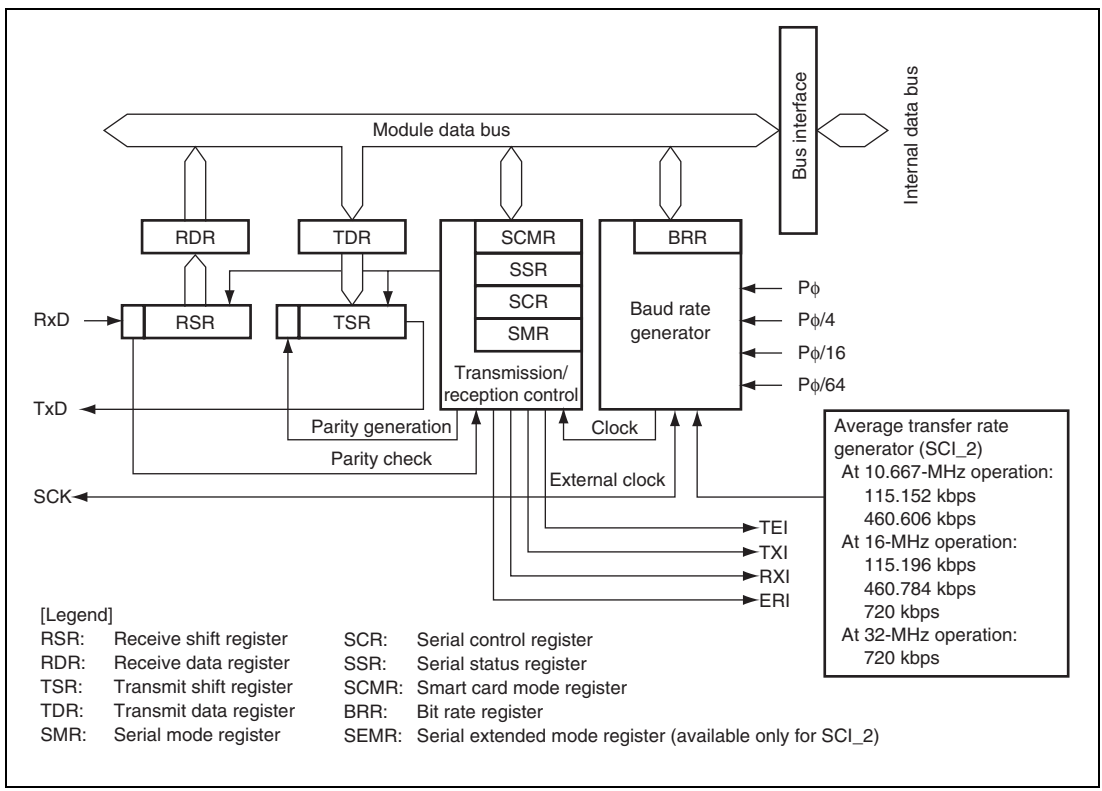


Figure 14.1 Block Diagram of SCI

14.2 Input/Output Pins

Table 14.1 lists the pin configuration of the SCI.

Table 14.1 Pin Configuration

Channel	Pin Name*	I/O	Function
0	SCK0	I/O	Channel 0 clock input/output
	RxD0	Input	Channel 0 receive data input
	TxD0	Output	Channel 0 transmit data output
1	SCK1	I/O	Channel 1 clock input/output
	RxD1	Input	Channel 1 receive data input
	TxD1	Output	Channel 1 transmit data output
2	SCK2	I/O	Channel 2 clock input/output
	RxD2	Input	Channel 2 receive data input
	TxD2	Output	Channel 2 transmit data output
4	SCK4	I/O	Channel 4 clock input/output
	RxD4	Input	Channel 4 receive data input
	TxD4	Output	Channel 4 transmit data output

Note: * Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

14.3 Register Descriptions

The SCI has the following registers. Some bits in the serial mode register (SMR), serial status register (SSR), and serial control register (SCR) have different functions in different modes—normal serial communication interface mode and smart card interface mode; therefore, the bits are described separately for each mode in the corresponding register sections.

Channel 0:

- Receive shift register_0 (RSR_0)
- Transmit shift register_0 (TSR_0)
- Receive data register_0 (RDR_0)
- Transmit data register_0 (TDR_0)
- Serial mode register_0 (SMR_0)
- Serial control register_0 (SCR_0)
- Serial status register_0 (SSR_0)
- Smart card mode register_0 (SCMR_0)
- Bit rate register_0 (BRR_0)

Channel 1:

- Receive shift register_1 (RSR_1)
- Transmit shift register_1 (TSR_1)
- Receive data register_1 (RDR_1)
- Transmit data register_1 (TDR_1)
- Serial mode register_1 (SMR_1)
- Serial control register_1 (SCR_1)
- Serial status register_1 (SSR_1)
- Smart card mode register_1 (SCMR_1)
- Bit rate register_1 (BRR_1)

Channel 2:

- Receive shift register_2 (RSR_2)
- Transmit shift register_2 (TSR_2)
- Receive data register_2 (RDR_2)
- Transmit data register_2 (TDR_2)
- Serial mode register_2 (SMR_2)
- Serial control register_2 (SCR_2)
- Serial status register_2 (SSR_2)
- Smart card mode register_2 (SCMR_2)
- Bit rate register_2 (BRR_2)
- Serial extended mode register_2 (SEMR_2) (SCI_2 only)

Channel 4:

- Receive shift register_4 (RSR_4)
- Transmit shift register_4 (TSR_4)
- Receive data register_4 (RDR_4)
- Transmit data register_4 (TDR_4)
- Serial mode register_4 (SMR_4)
- Serial control register_4 (SCR_4)
- Serial status register_4 (SSR_4)
- Smart card mode register_4 (SCMR_4)
- Bit rate register_4 (BRR_4)

14.3.1 Receive Shift Register (RSR)

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

14.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one frame of serial data, it transfers the received serial data from RSR to RDR where it is stored. This allows RSR to receive the next data. Since RSR and RDR function as a double buffer in this way, continuous receive operations can be performed. After confirming that the RDRF bit in SSR is set to 1, read RDR only once. RDR cannot be written to by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

14.3.3 Transmit Data Register (TDR)

TDR is an 8-bit register that stores transmit data. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structures of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR when one frame of data is transmitted, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read from or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR for only once after confirming that the TDRE bit in SSR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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

14.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first automatically transfers transmit data from TDR to TSR, then sends the data to the TxD pin. TSR cannot be directly accessed by the CPU.

14.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock source. Some bits in SMR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	GM	BLK	PE	O/$\overline{E}$	BCP1	BCP0	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 Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	C/$\overline{A}$	0	R/W	Communication Mode 0: Asynchronous mode 1: Clocked synchronous mode
6	CHR	0	R/W	Character Length (valid only in asynchronous mode) 0: Selects 8 bits as the data length. 1: Selects 7 bits as the data length. LSB-first is fixed and the MSB (bit 7) in TDR is not transmitted in transmission. In clocked synchronous mode, a fixed data length of 8 bits is used.
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. For a multiprocessor format, parity bit addition and checking are not performed regardless of the PE bit setting.

Bit	Bit Name	Initial Value	R/W	Description
4	O/$\overline{E}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity.
3	STOP	0	R/W	Stop Bit Length (valid only in asynchronous mode) Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.
2	MP	0	R/W	Multiprocessor Mode (valid only in asynchronous mode) When this bit is set to 1, the multiprocessor function is enabled. The PE bit and O/$\overline{E}$ bit settings are invalid in multiprocessor mode.
1	CKS1	0	R/W	Clock Select 1, 0
0	CKS0	0	R/W	These bits select the clock source for the baud rate generator. 00: Pϕ clock (n = 0) 01: Pϕ/4 clock (n = 1) 10: Pϕ/16 clock (n = 2) 11: Pϕ/64 clock (n = 3) For the relation between the settings of these bits and the baud rate, see section 14.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 14.3.9, Bit Rate Register (BRR)).

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	GM	0	R/W	GSM Mode Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 etu from the start and the clock output control function is appended. For details, see sections 14.7.6, Data Transmission (Except in Block Transfer Mode) and 14.7.8, Clock Output Control.

Bit	Bit Name	Initial Value	R/W	Description
6	BLK	0	R/W	Setting this bit to 1 allows block transfer mode operation. For details, see section 14.7.3, Block Transfer Mode.
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. Set this bit to 1 in smart card interface mode.
4	O/$\overline{E}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode) 0: Selects even parity 1: Selects odd parity For details on the usage of this bit in smart card interface mode, see section 14.7.2, Data Format (Except in Block Transfer Mode).
3	BCP1	0	R/W	Basic Clock Pulse 1,0
2	BCP0	0	R/W	These bits select the number of basic clock cycles in a 1-bit data transfer time in smart card interface mode. 00: 32 clock cycles (S = 32) 01: 64 clock cycles (S = 64) 10: 372 clock cycles (S = 372) 11: 256 clock cycles (S = 256) For details, see section 14.7.4, Receive Data Sampling Timing and Reception Margin. S is described in section 14.3.9, Bit Rate Register (BRR).
1	CKS1	0	R/W	Clock Select 1,0
0	CKS0	0	R/W	These bits select the clock source for the baud rate generator. 00: Pϕ clock (n = 0) 01: Pϕ/4 clock (n = 1) 10: Pϕ/16 clock (n = 2) 11: Pϕ/64 clock (n = 3) For the relation between the settings of these bits and the baud rate, see section 14.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 14.3.9, Bit Rate Register (BRR)).

Note: etu (Elementary Time Unit): 1-bit transfer time

14.3.6 Serial Control Register (SCR)

SCR is a register that enables/disables the following SCI transfer operations and interrupt requests, and selects the transfer clock source. For details on interrupt requests, see section 14.8, Interrupt Sources. Some bits in SCR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	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 Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, a TXI interrupt request is enabled. A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled. RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.

Bit	Bit Name	Initial Value	R/W	Description
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format. If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format. Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the MP bit in SMR is 1 in asynchronous mode) When this bit is set to 1, receive data in which the multiprocessor bit is 0 is skipped, and setting of the RDRF, FER, and ORER status flags in SSR is disabled. On receiving data in which the multiprocessor bit is 1, this bit is automatically cleared and normal reception is resumed. For details, see section 14.5, Multiprocessor Communication Function. When receive data including MPB = 0 in SSR is being received, transfer of the received data from RSR to RDR, detection of reception errors, and the settings of RDRF, FER, and ORER flags in SSR are not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is automatically cleared to 0, and RXI and ERI interrupt requests (in the case where the TIE and RIE bits in SCR are set to 1) and setting of the FER and ORER flags are enabled.

Bit	Bit Name	Initial Value	R/W	Description
2	TEIE	0	R/W	Transmit End Interrupt Enable When this bit is set to 1, a TEI interrupt request is enabled. A TEI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0 in order to clear the TEND flag to 0, or by clearing the TEIE bit to 0.
1	CKE1	0	R/W	Clock Enable 1, 0
0	CKE0	0	R/W	These bits select the clock source and SCK pin function. • Asynchronous mode 00: On-chip baud rate generator (SCK pin functions as I/O port.) 01: On-chip baud rate generator (Outputs a clock with the same frequency as the bit rate from the SCK pin.) 1X: External clock (Inputs a clock with a frequency 16 times the bit rate from the SCK pin.) • Clocked synchronous mode 0X: Internal clock (SCK pin functions as clock output.) 1X: External clock (SCK pin functions as clock input.)

Note: X: Don't care

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1,a TXI interrupt request is enabled. A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled. RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format. If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format. Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the MP bit in SMR is 1 in asynchronous mode) Write 0 to this bit in smart card interface mode.
2	TEIE	0	R/W	Transmit End Interrupt Enable Write 0 to this bit in smart card interface mode.

Bit	Bit Name	Initial Value	R/W	Description
1	CKE1	0	R/W	Clock Enable 1, 0
0	CKE0	0	R/W	These bits control the clock output from the SCK pin. In GSM mode, clock output can be dynamically switched. For details, see section 14.7.8, Clock Output Control. • When GM in SMR = 0 00: Output disabled (SCK pin functions as I/O port.) 01: Clock output 1X: Reserved • When GM in SMR = 1 00: Output fixed low 01: Clock output 10: Output fixed high 11: Clock output

14.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. TDRE, RDRF, ORER, PER, and FER can only be cleared. Some bits in SSR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	FRE	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: * Only 0 can be written, to clear the flag.

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	ERS	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: * Only 0 can be written, to clear the flag.

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates whether TDR contains transmit data. [Setting conditions] • When the TE bit in SCR is 0 • When data is transferred from TDR to TSR [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When a TXI interrupt request is issued allowing the DMAC or DTC to write data to TDR
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates whether receive data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing conditions] • When 0 is written to RDRF after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When an RXI interrupt request is issued allowing the DMAC or DTC to read data from RDR The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0. Note that when the next serial reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error Indicates that an overrun error has occurred during reception and the reception ends abnormally. [Setting condition] • When the next serial reception is completed while RDRF = 1 In RDR, receive data prior to an overrun error occurrence is retained, but data received after the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to ORER after reading ORER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.

Bit	Bit Name	Initial Value	R/W	Description
4	FER	0	R/(W)*	Framing Error Indicates that a framing error has occurred during reception in asynchronous mode and the reception ends abnormally. [Setting condition] • When the stop bit is 0 In 2-stop-bit mode, only the first stop bit is checked whether it is 1 but the second stop bit is not checked. Note that receive data when the framing error occurs is transferred to RDR, however, the RDRF flag is not set. In addition, when the FER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to FER after reading FER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the FER flag is not affected and retains its previous value.

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally. [Setting condition] • When a parity error is detected during reception Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to PER after reading PER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the PER bit is not affected and retains its previous value.
2	TEND	1	R	Transmit End [Setting conditions] • When the TE bit in SCR is 0 • When TDRE = 1 at transmission of the last bit of a transmit character [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When a TXI interrupt request is issued allowing the DMAC or DTC to write data to TDR
1	MPB	0	R	Multiprocessor Bit Stores the multiprocessor bit value in the receive frame. When the RE bit in SCR is cleared to 0 its previous state is retained.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Sets the multiprocessor bit value to be added to the transmit frame.

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

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates whether TDR contains transmit data. [Setting conditions] • When the TE bit in SCR is 0 • When data is transferred from TDR to TSR [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When a TXI interrupt request is issued allowing the DMAC or DTC to write data to TDR
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates whether receive data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing conditions] • When 0 is written to RDRF after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When an RXI interrupt request is issued allowing the DMAC or DTC to read data from RDR The RDRF flag is not affected and retains its previous value even when the RE bit in SCR is cleared to 0. Note that when the next reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error Indicates that an overrun error has occurred during reception and the reception ends abnormally. [Setting condition] • When the next serial reception is completed while RDRF = 1 In RDR, the receive data prior to an overrun error occurrence is retained, but data received following the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to ORER after reading ORER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.
4	ERS	0	R/(W)*	Error Signal Status [Setting condition] • When a low error signal is sampled [Clearing condition] • When 0 is written to ERS after reading ERS = 1

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally. [Setting condition] • When a parity error is detected during reception Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to PER after reading PER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the PER flag is not affected and retains its previous value.

Bit	Bit Name	Initial Value	R/W	Description
2	TEND	1	R	Transmit End This bit is set to 1 when no error signal is sent from the receiving side and the next transmit data is ready to be transferred to TDR. [Setting conditions] • When both the TE and ERS bits in SCR are 0 • When ERS = 0 and TDRE = 1 after a specified time passed after completion of 1-byte data transfer. The set timing depends on the register setting as follows: When GM = 0 and BLK = 0, 2.5 etu after transmission start When GM = 0 and BLK = 1, 1.5 etu after transmission start When GM = 1 and BLK = 0, 1.0 etu after transmission start When GM = 1 and BLK = 1, 1.0 etu after transmission start [Clearing conditions] • When 0 is written to TEND after reading TEND = 1 • When a TXI interrupt request is issued allowing the DMAC or DTC to write the next data to TDR
1	MPB	0	R	Multiprocessor Bit Not used in smart card interface mode.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Write 0 to this bit in smart card interface mode.

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

14.3.8 Smart Card Mode Register (SCMR)

SCMR selects smart card interface mode and its format.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	SDIR	SINV	—	SMIF
Initial Value	1	1	1	1	0	0	1	0
R/W	R	R	R	R	R/W	R/W	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These are read-only bits and cannot be modified.
3	SDIR	0	R/W	Smart Card Data Transfer Direction Selects the serial/parallel conversion format. 0: Transfer with LSB-first 1: Transfer with MSB-first This bit is valid only when the 8-bit data format is used for transmission/reception; when the 7-bit data format is used, data is always transmitted/received with LSB-first.
2	SINV	0	R/W	Smart Card Data Invert Inverts the transmit/receive data logic level. This bit does not affect the logic level of the parity bit. To invert the parity bit, invert the O/$\overline{E}$ bit in SMR. 0: TDR contents are transmitted as they are. Receive data is stored as it is in RDR. 1: TDR contents are inverted before being transmitted. Receive data is stored in inverted form in RDR.
1	—	1	R	Reserved This is a read-only bit and cannot be modified.
0	SMIF	0	R/W	Smart Card Interface Mode Select When this bit is set to 1, smart card interface mode is selected. 0: Normal asynchronous or clocked synchronous mode 1: Smart card interface mode

14.3.9 Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 14.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clocked synchronous mode, and smart card interface mode. The initial value of BRR is H'FF, and it can be read from or written to by the CPU at all times.

Table 14.2 Relationships between N Setting in BRR and Bit Rate B

Mode	Bit Rate	Error
Asynchronous mode	$N = \frac{P\phi \times 10^6}{64 \times 2^{2n-1} \times B} - 1$	$Error\ (\%) = \{\frac{P\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1\} \times 100$
Clocked synchronous mode	$N = \frac{P\phi \times 10^6}{8 \times 2^{2n-1} \times B} - 1$	
Smart card interface mode	$N = \frac{P\phi \times 10^6}{S \times 2^{2n+1} \times B} - 1$	$Error\ (\%) = \{\frac{P\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N+1)} - 1\} \times 100$

[Legend]

B: Bit rate (bit/s)

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

Pϕ: Operating frequency (MHz)

n and S: Determined by the SMR settings shown in the following table.

SMR Setting			SMR Setting		
CKS1	**CKS0**	**n**	**BCP1**	**BCP0**	**S**
0	0	0	0	0	32
0	1	1	0	1	64
1	0	2	1	0	372
1	1	3	1	1	256

Table 14.3 shows sample N settings in BRR in normal asynchronous mode. Table 14.4 shows the maximum bit rate settable for each operating frequency. Tables 14.6 and 14.8 show sample N settings in BRR in clocked synchronous mode and smart card interface mode, respectively. In smart card interface mode, the number of basic clock cycles S in a 1-bit data transfer time can be selected. For details, see section 14.7.4, Receive Data Sampling Timing and Reception Margin. Tables 14.5 and 14.7 show the maximum bit rates with external clock input.

Table 14.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (1)

	Operating Frequency Pϕ (MHz)											
	8			9.8304			10			12		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	141	0.03	2	174	–0.26	2	177	–0.25	2	212	0.03
150	2	103	0.16	2	127	0.00	2	129	0.16	2	155	0.16
300	1	207	0.16	1	255	0.00	2	64	0.16	2	77	0.16
600	1	103	0.16	1	127	0.00	1	129	0.16	1	155	0.16
1200	0	207	0.16	0	255	0.00	1	64	0.16	1	77	0.16
2400	0	103	0.16	0	127	0.00	0	129	0.16	0	155	0.16
4800	0	51	0.16	0	63	0.00	0	64	0.16	0	77	0.16
9600	0	25	0.16	0	31	0.00	0	32	–1.36	0	38	0.16
19200	0	12	0.16	0	15	0.00	0	15	1.73	0	19	–2.34
31250	0	7	0.00	0	9	–1.70	0	9	0.00	0	11	0.00
38400	—	—	—	0	7	0.00	0	7	1.73	0	9	–2.34

	Operating Frequency Pϕ (MHz)											
	12.288			14			14.7456			16		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	217	0.08	2	248	–0.17	3	64	0.70	3	70	0.03
150	2	159	0.00	2	181	0.16	2	191	0.00	2	207	0.16
300	2	79	0.00	2	90	0.16	2	95	0.00	2	103	0.16
600	1	159	0.00	1	181	0.16	1	191	0.00	1	207	0.16
1200	1	79	0.00	1	90	0.16	1	95	0.00	1	103	0.16
2400	0	159	0.00	0	181	0.16	0	191	0.00	0	207	0.16
4800	0	79	0.00	0	90	0.16	0	95	0.00	0	103	0.16
9600	0	39	0.00	0	45	–0.93	0	47	0.00	0	51	0.16
19200	0	19	0.00	0	22	–0.93	0	23	0.00	0	25	0.16
31250	0	11	2.40	0	13	0.00	0	14	–1.70	0	15	0.00
38400	0	9	0.00	—	—	—	0	11	0.00	0	12	0.16

Table 14.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (2)

	Operating Frequency Pϕ (MHz)											
	17.2032			18			19.6608			20		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	75	0.48	3	79	–0.12	3	86	0.31	3	88	–0.25
150	2	223	0.00	2	233	0.16	2	255	0.00	3	64	0.16
300	2	111	0.00	2	116	0.16	2	127	0.00	2	129	0.16
600	1	223	0.00	1	233	0.16	1	255	0.00	2	64	0.16
1200	1	111	0.00	1	116	0.16	1	127	0.00	1	129	0.16
2400	0	223	0.00	0	233	0.16	0	255	0.00	1	64	0.16
4800	0	111	0.00	0	116	0.16	0	127	0.00	0	129	0.16
9600	0	55	0.00	0	58	–0.69	0	63	0.00	0	64	0.16
19200	0	27	0.00	0	28	1.02	0	31	0.00	0	32	–1.36
31250	0	16	1.20	0	17	0.00	0	19	–1.70	0	19	0.00
38400	0	13	0.00	0	14	–2.34	0	15	0.00	0	15	1.73

	Operating Frequency Pϕ (MHz)											
	25			30			33			35		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	110	–0.02	3	132	0.13	3	145	0.33	3	154	0.23
150	3	80	0.47	3	97	–0.35	3	106	0.39	3	113	–0.06
300	2	162	–0.15	2	194	0.16	2	214	–0.07	2	227	0.00
600	2	80	0.47	2	97	–0.35	2	106	0.39	2	113	0.00
1200	1	162	–0.15	1	194	0.16	1	214	–0.07	1	227	0.00
2400	1	80	0.47	1	97	–0.35	1	106	0.39	1	113	0.00
4800	0	162	–0.15	0	194	0.16	0	214	–0.07	0	227	0.00
9600	0	80	0.47	0	97	–0.35	0	106	0.39	0	113	0.00
19200	0	40	–0.76	0	48	–0.35	0	53	–0.54	0	56	0.00
31250	0	24	0.00	0	29	0	0	32	0	0	34	0.00
38400	0	19	1.73	0	23	1.73	0	26	–0.54	0	28	–1.78

Table 14.4 Maximum Bit Rate for Each Operating Frequency (Asynchronous Mode)

Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N	Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
8	250000	0	0	17.2032	537600	0	0
9.8304	307200	0	0	18	562500	0	0
10	312500	0	0	19.6608	614400	0	0
12	375000	0	0	20	625000	0	0
12.288	384000	0	0	25	781250	0	0
14	437500	0	0	30	937500	0	0
14.7456	460800	0	0	33	1031250	0	0
16	500000	0	0	35	1093750	0	0

Table 14.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	2.0000	125000	17.2032	4.3008	268800
9.8304	2.4576	153600	18	4.5000	281250
10	2.5000	156250	19.6608	4.9152	307200
12	3.0000	187500	20	5.0000	312500
12.288	3.0720	192000	25	6.2500	390625
14	3.5000	218750	30	7.5000	468750
14.7456	3.6864	230400	33	8.2500	515625
16	4.0000	250000	35	8.7500	546875

Table 14.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)

Bit Rate	Operating Frequency Pϕ (MHz)							
	8		10		16		20	
(bit/s)	n	N	n	N	n	N	n	N
110								
250	2	124	—	—	3	249		
500	2	249	—	—	3	124	—	—
1 k	2	124	—	—	2	249	—	—
2.5 k	1	199	1	249	2	99	2	124
5 k	1	99	1	124	1	199	1	249
10 k	0	199	0	249	1	99	1	124
25 k	0	79	0	99	0	159	0	199
50 k	0	39	0	49	0	79	0	99
100 k	0	19	0	24	0	39	0	49
250 k	0	7	0	9	0	15	0	19
500 k	0	3	0	4	0	7	0	9
1 M	0	1			0	3	0	4
2.5 M			0	0*			0	1
5 M							0	0*

Bit Rate (bit/s)	Operating Frequency Pϕ (MHz)							
	25		30		33		35	
	n	N	n	N	n	N	n	N
110								
250								
500			3	233				
1 k	3	97	3	116	3	128	3	136
2.5 k	2	155	2	187	2	205	2	218
5 k	2	77	2	93	2	102	2	108
10 k	1	155	1	187	1	205	1	218
25 k	0	249	1	74	1	82	1	87
50 k	0	124	0	149	0	164	0	174
100 k	0	62	0	74	0	82	0	87
250 k	0	24	0	29	0	32	0	34
500 k	—	—	0	14	—	—	—	—
1 M	—	—	—	—	—	—	—	—
2.5 M	—	—	0	2	—	—	—	—
5 M	—	—	—	—	—	—	—	—

[Legend]

Space : Setting prohibited.

— : Can be set, but there will be error.

* : Continuous transmission or reception is not possible.

Table 14.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	1.3333	1333333.3	20	3.3333	3333333.3
10	1.6667	1666666.7	25	4.1667	4166666.7
12	2.0000	2000000.0	30	5.0000	5000000.0
14	2.3333	2333333.3	33	5.5000	5500000.0
16	2.6667	2666666.7	35	5.8336	5833625.0
18	3.0000	3000000.0			

Table 14.8 BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, S = 372)

	Operating Frequency Pϕ (MHz)											
Bit Rate	7.1424			10.00			10.7136			13.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	0	0.00	0	1	30	0	1	25	0	1	8.99

	Operating Frequency Pϕ (MHz)											
Bit Rate	14.2848			16.00			18.00			20.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	1	0.00	0	1	12.01	0	2	15.99	0	2	6.60

	Operating Frequency Pϕ (MHz)											
Bit Rate	25.00			30.00			33.00			35.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	3	12.49	0	3	5.01	0	4	7.59	0	4	1.99

Table 14.9 Maximum Bit Rate for Each Operating Frequency (Smart Card Interface Mode, S = 372)

Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N	Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
7.1424	9600	0	0	18.00	24194	0	0
10.00	13441	0	0	20.00	26882	0	0
10.7136	14400	0	0	25.00	33602	0	0
13.00	17473	0	0	30.00	40323	0	0
14.2848	19200	0	0	33.00	44355	0	0
16.00	21505	0	0	35.00	47043	0	0

14.3.10 Serial Extended Mode Register (SEMR)

SEMR selects the clock source in asynchronous mode. The basic clock is automatically specified when the average transfer rate operation is selected.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	ABCS	ACS2	ACS1	ACS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6 to 4	—	All 0	R	Reserved These are read-only bits and cannot be modified.
3	ABCS	0	R/W	Asynchronous Mode Basic Clock Select (valid only in asynchronous mode) Selects the basic clock for a 1-bit period. 0: The basic clock has a frequency 16 times the transfer rate 1: The basic clock has a frequency 8 times the transfer rate

Bit	Bit Name	Initial Value	R/W	Description
2	ACS2	0	R/W	Asynchronous Mode Clock Source Select (valid when CKE1 = 1 in asynchronous mode) These bits select the clock source for the average transfer rate function. When the average transfer rate function is enabled, the basic clock is automatically specified regardless of the ABCS bit value. 000: External clock input 001: 115.152 kbps of average transfer rate specific to Pϕ = 10.667 MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate) 010: 460.606 kbps of average transfer rate specific to Pϕ = 10.667 MHz is selected (operated using the basic clock with a frequency 8 times the transfer rate) 011: 720 kbps of average transfer rate specific to Pϕ = 32 MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate) 100: Setting prohibited 101: 115.196 kbps of average transfer rate specific to Pϕ = 16 MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate) 110: 460.784 kbps of average transfer rate specific to Pϕ = 16 MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate) 111: 720 kbps of average transfer rate specific to Pϕ = 16 MHz is selected (operated using the basic clock with a frequency 8 times the transfer rate) The average transfer rate only supports operating frequencies of 10.667 MHz, 16 MHz, and 32 MHz.
1	ACS1	0	R/W	
0	ACS0	0	R/W	

14.4 Operation in Asynchronous Mode

Figure 14.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transmit/receive data, a parity bit, and finally stop bits (high level). In asynchronous serial communication, the communication line is usually held in the mark state (high level). The SCI monitors the communication line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transmission and reception.

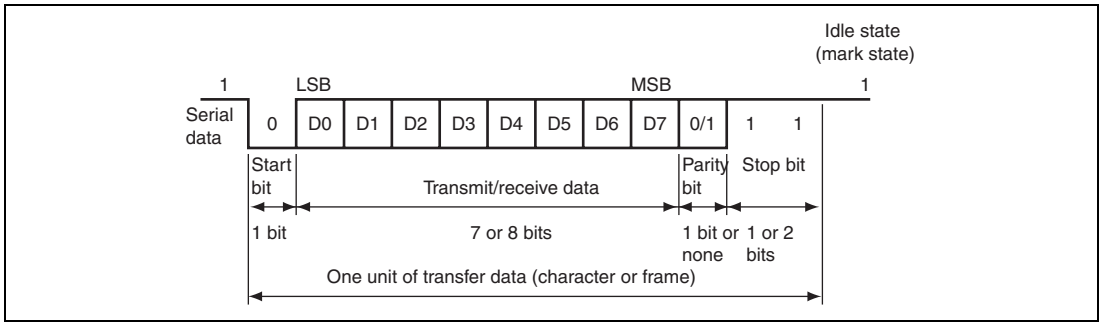


Figure 14.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

14.4.1 Data Transfer Format

Table 14.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting. For details on the multiprocessor bit, see section 14.5, Multiprocessor Communication Function.

Table 14.10 Serial Transfer Formats (Asynchronous Mode)

SMR Settings				Serial Transmit/Receive Format and Frame Length
CHR	PE	MP	STOP	1 2 3 4 5 6 7 8 9 10 11 12
0	0	0	0	S 8-bit data STOP
0	0	0	1	S 8-bit data STOP STOP
0	1	0	0	S 8-bit data P STOP
0	1	0	1	S 8-bit data P STOP STOP
1	0	0	0	S 7-bit data STOP
1	0	0	1	S 7-bit data STOP STOP
1	1	0	0	S 7-bit data P STOP
1	1	0	1	S 7-bit data P STOP STOP
0	—	1	0	S 8-bit data MPB STOP
0	—	1	1	S 8-bit data MPB STOP STOP
1	—	1	0	S 7-bit data MPB STOP
1	—	1	1	S 7-bit data MPB STOP STOP

[Legend]
S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

14.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the bit rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Since receive data is sampled at the rising edge of the 8th pulse of the basic clock, data is latched at the middle of each bit, as shown in figure 14.3. Thus the reception margin in asynchronous mode is determined by formula (1) below.

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)\,F - \frac{\left|D - 0.5\right|}{N}\,(1 + F)\right| \times 100\,[\%] \quad \cdots \quad \text{Formula (1)}$$

M: Reception margin
N: Ratio of bit rate to clock (N = 16)
D: Duty cycle of clock (D = 0.5 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in formula (1), the reception margin is determined by the formula below.

$$M = \left(0.5 - \frac{1}{2 \times 16}\right) \times 100[\%] = 46.875\%$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

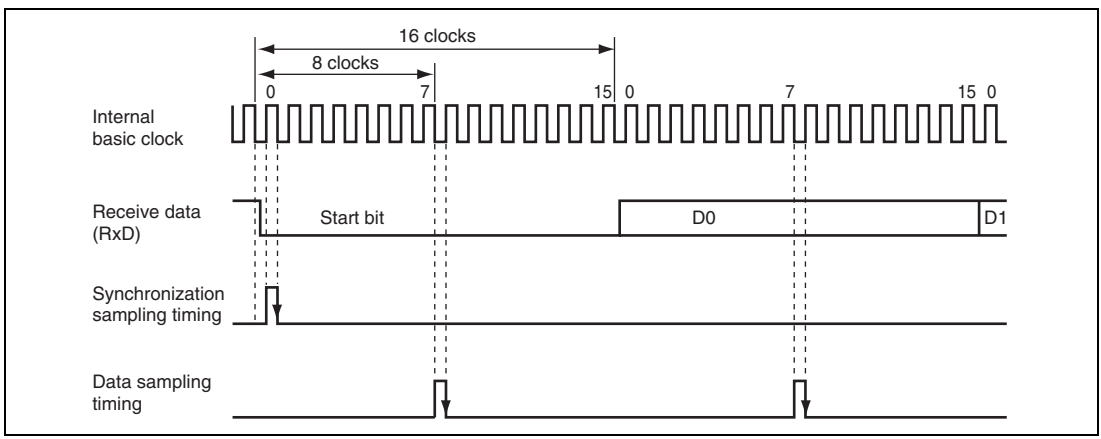


Figure 14.3 Receive Data Sampling Timing in Asynchronous Mode

14.4.3 Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input to the SCK pin can be selected as the SCI's transfer clock, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR. When an external clock is input to the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 14.4.

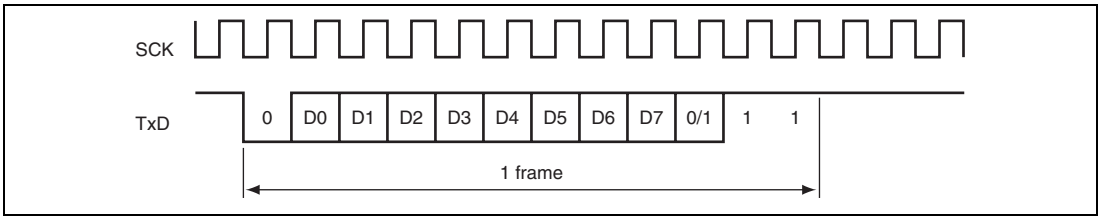


Figure 14.4 Phase Relation between Output Clock and Transmit Data (Asynchronous Mode)

14.4.4 SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 14.5. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization.

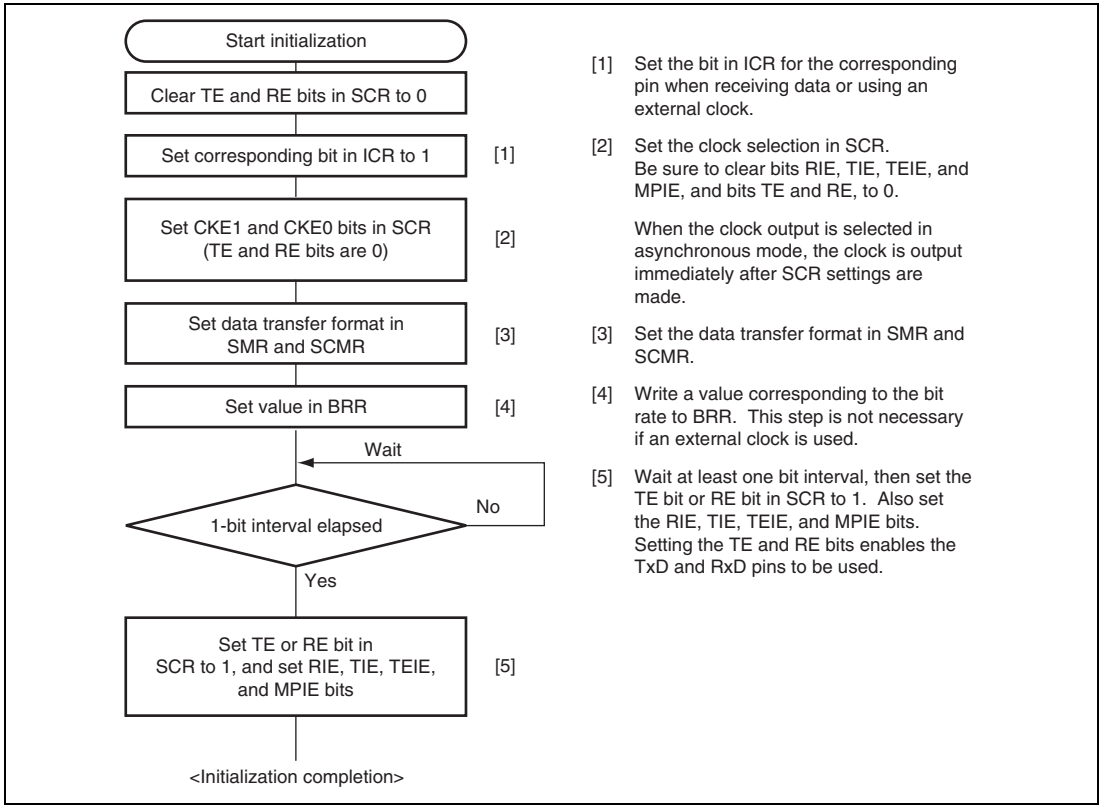


Figure 14.5 Sample SCI Initialization Flowchart

14.4.5 Serial Data Transmission (Asynchronous Mode)

Figure 14.6 shows an example of the operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is cleared to 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit or multiprocessor bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the next transmit data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 14.7 shows a sample flowchart for transmission in asynchronous mode.

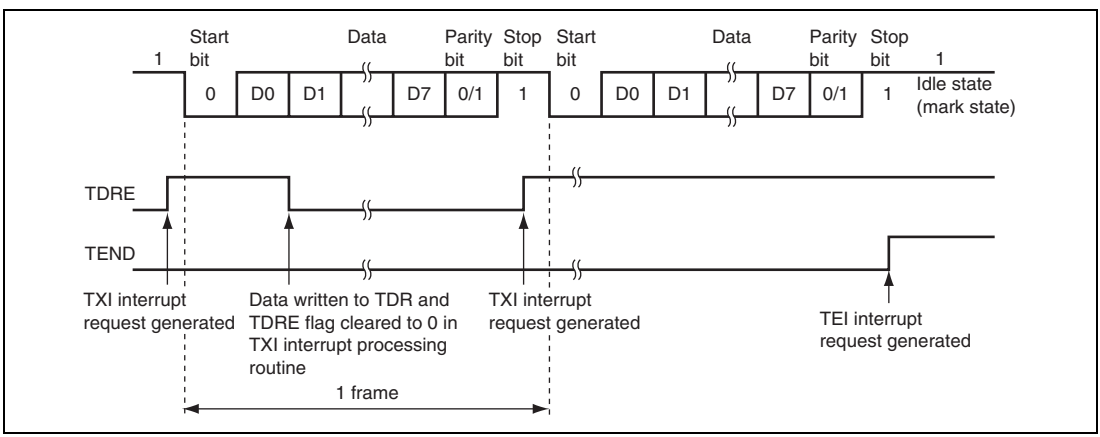
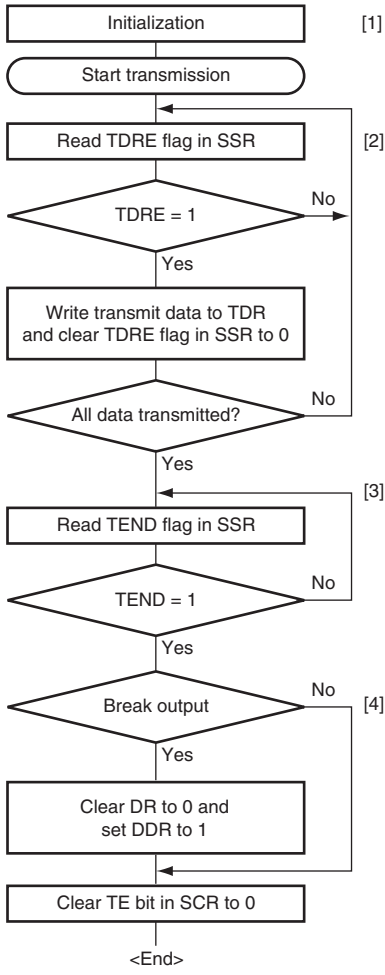


Figure 14.6 Example of Operation for Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin. After the TE bit is set to 1, a 1 is output for a frame, and transmission is enabled.

[2] SCI state check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:

To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DMAC or DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port corresponding to the TxD pin to 1, clear DR to 0, then clear the TE bit in SCR to 0.

Figure 14.7 Sample Serial Transmission Flowchart

14.4.6 Serial Data Reception (Asynchronous Mode)

Figure 14.8 shows an example of the operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, stores receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.

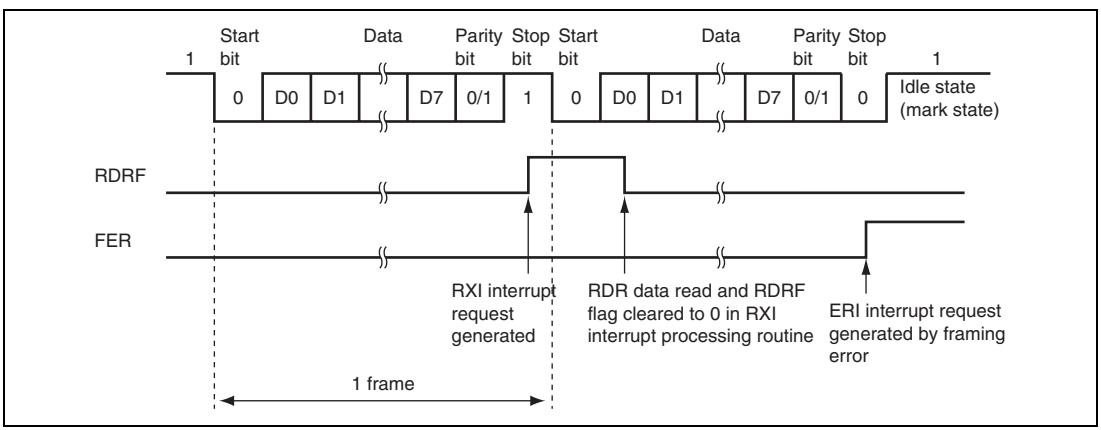


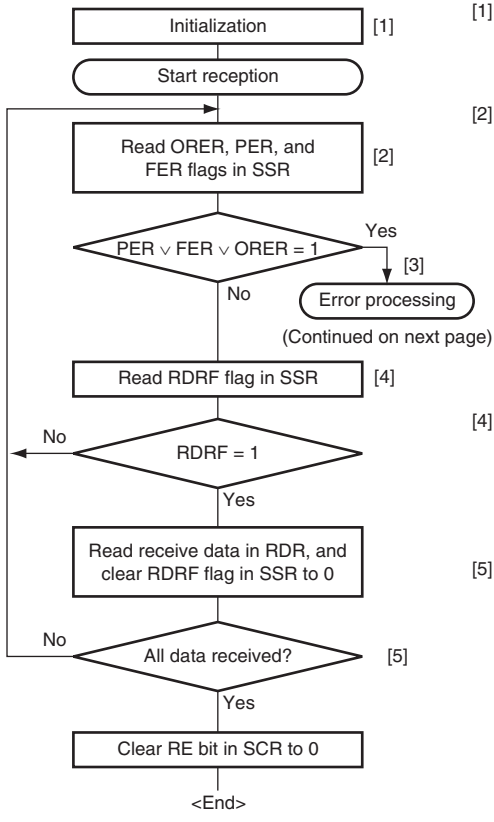
Figure 14.8 Example of SCI Operation for Reception (Example with 8-Bit Data, Parity, One Stop Bit)

Table 14.11 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 14.9 shows a sample flowchart for serial data reception.

Table 14.11 SSR Status Flags and Receive Data Handling

SSR Status Flag					
RDRF*	**ORER**	**FER**	**PER**	**Receive Data**	**Receive Error Type**
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

Note: * The RDRF flag retains the state it had before data reception.



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing and break detection:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.

[4] SCI state check and receive data read:
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag and RDR, and clear the RDRF flag to 0. However, the RDRF flag is cleared automatically when the DMAC or DTC is initiated by an RXI interrupt and reads data from RDR.

Figure 14.9 Sample Serial Reception Flowchart (1)

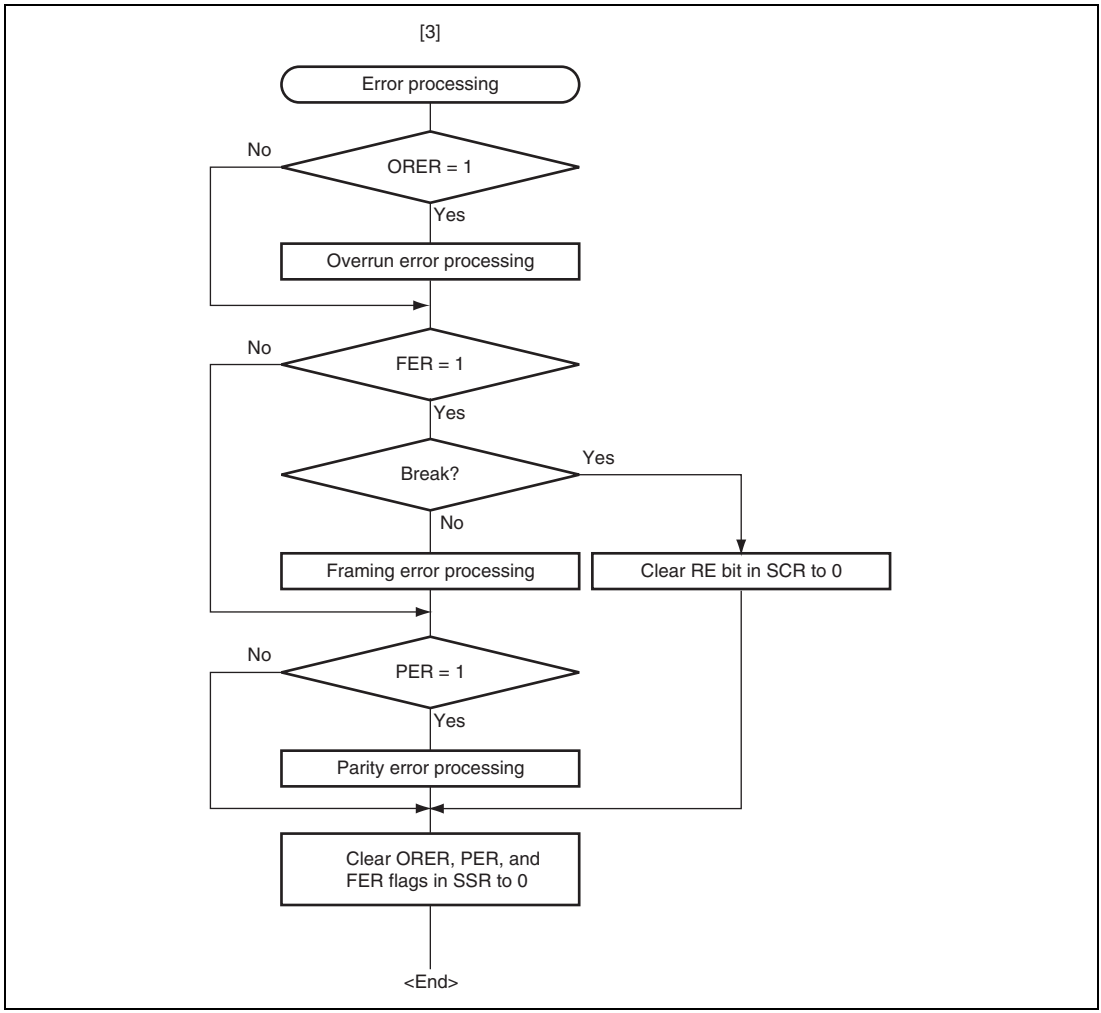


Figure 14.9 Sample Serial Reception Flowchart (2)

14.5 Multiprocessor Communication Function

Use of the multiprocessor communication function enables data transfer to be performed among a number of processors sharing communication lines by means of asynchronous serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data. When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code. The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle for the specified receiving station. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle. If the multiprocessor bit is 1, the cycle is an ID transmission cycle, and if the multiprocessor bit is 0, the cycle is a data transmission cycle. Figure 14.10 shows an example of inter-processor communication using the multiprocessor format. The transmitting station first sends data which includes the ID code of the receiving station and a multiprocessor bit set to 1. It then transmits transmit data added with a multiprocessor bit cleared to 0. The receiving station skips data until data with a 1 multiprocessor bit is sent. When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip data until data with a 1 multiprocessor bit is again received.

The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set to 1, transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status flags, RDRF, FER, and ORER in SSR to 1 are prohibited until data with a 1 multiprocessor bit is received. On reception of a receive character with a 1 multiprocessor bit, the MPB bit in SSR is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt is generated.

When the multiprocessor format is selected, the parity bit setting is invalid. All other bit settings are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.

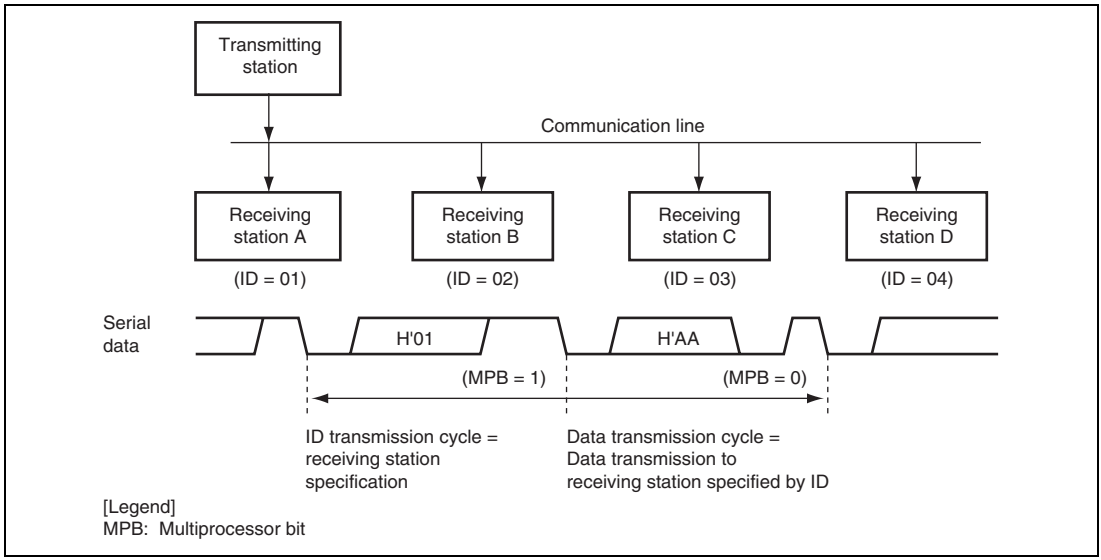


Figure 14.10 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)

14.5.1 Multiprocessor Serial Data Transmission

Figure 14.11 shows a sample flowchart for multiprocessor serial data transmission. For an ID transmission cycle, set the MPBT bit in SSR to 1 before transmission. For a data transmission cycle, clear the MPBT bit in SSR to 0 before transmission. All other SCI operations are the same as those in asynchronous mode.

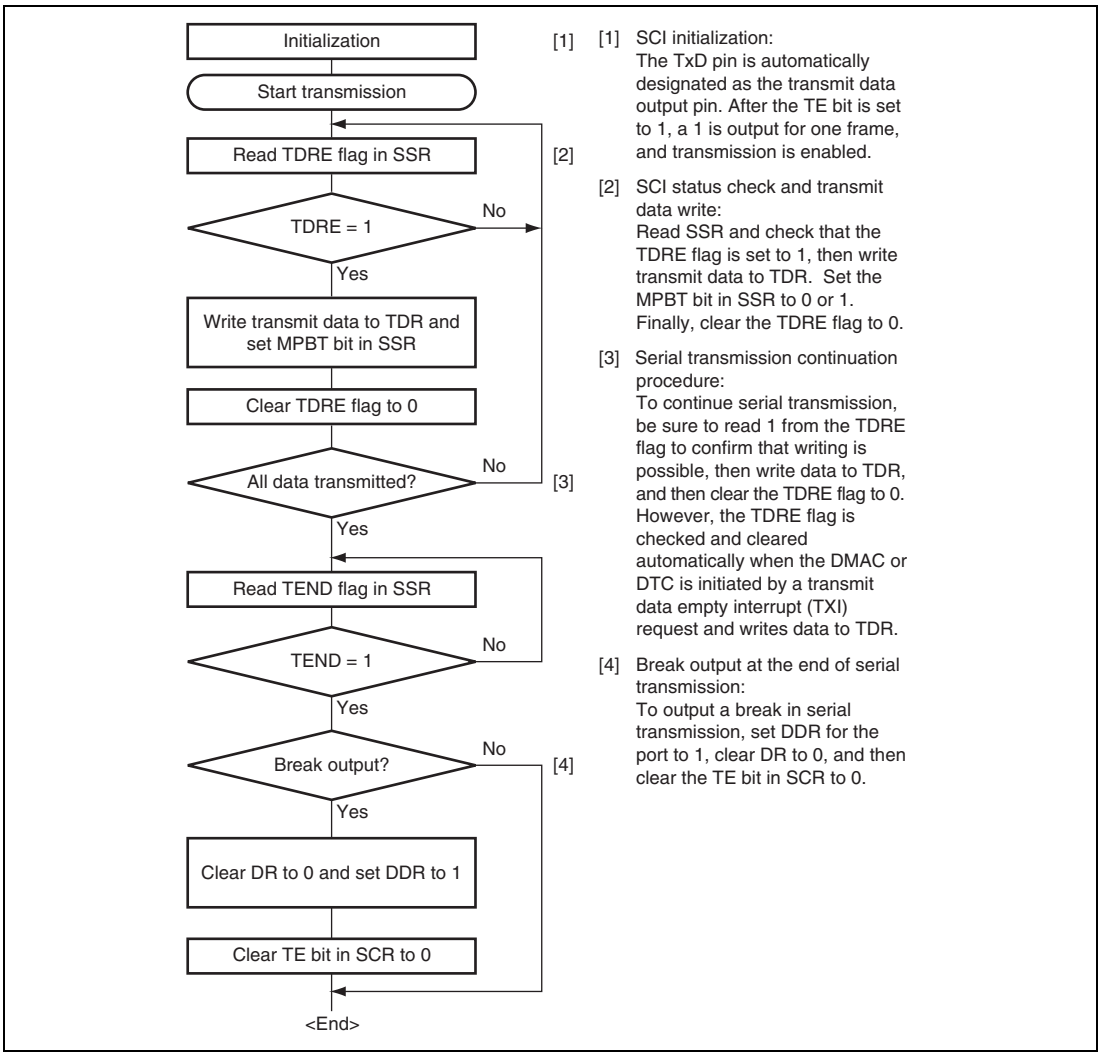


Figure 14.11 Sample Multiprocessor Serial Transmission Flowchart

14.5.2 Multiprocessor Serial Data Reception

Figure 14.13 shows a sample flowchart for multiprocessor serial data reception. If the MPIE bit in SCR is set to 1, data is skipped until data with a 1 multiprocessor bit is sent. On receiving data with a 1 multiprocessor bit, the receive data is transferred to RDR. An RXI interrupt request is generated at this time. All other SCI operations are the same as in asynchronous mode. Figure 14.12 shows an example of SCI operation for multiprocessor format reception.

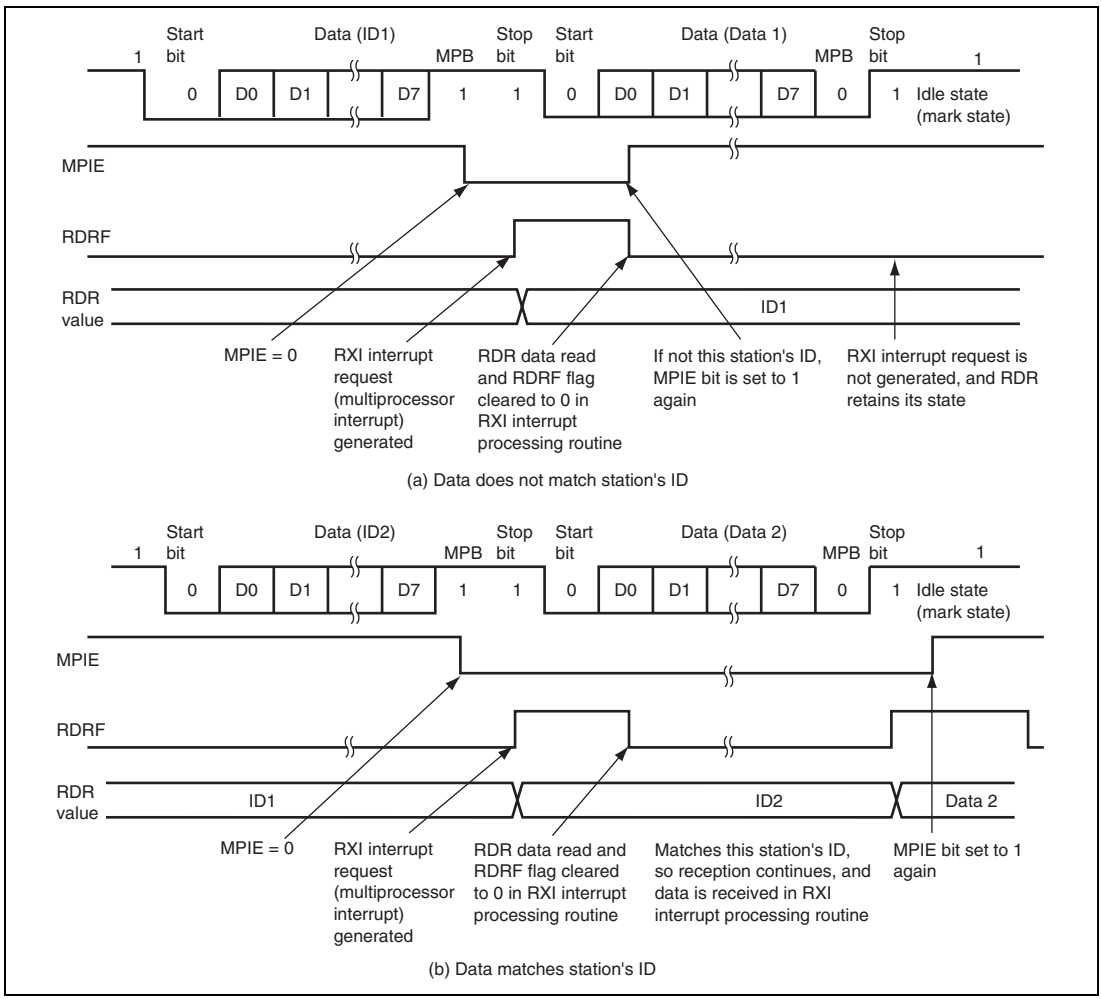
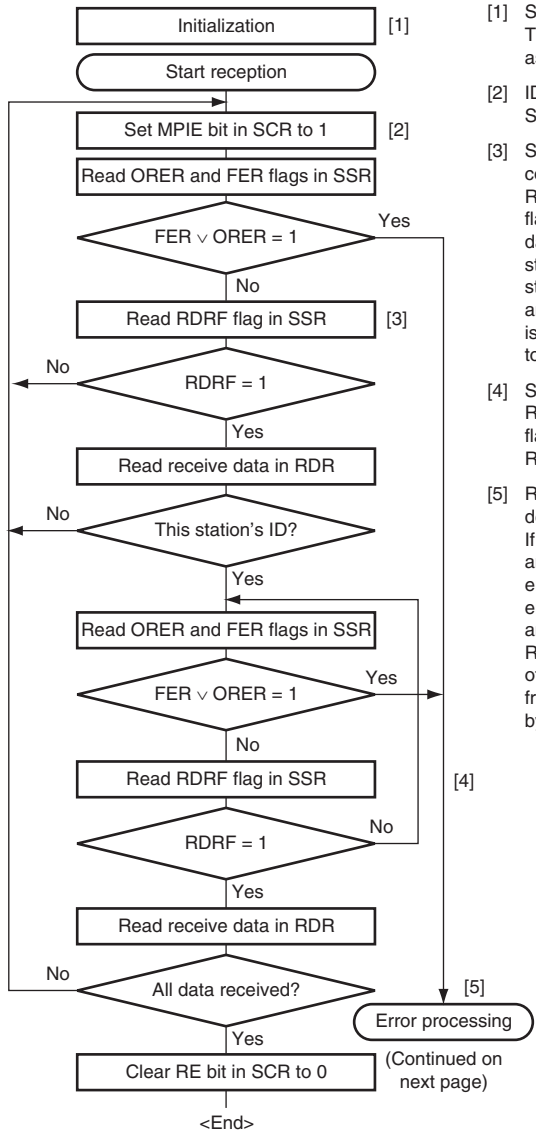


Figure 14.12 Example of SCI Operation for Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] ID reception cycle:
Set the MPIE bit in SCR to 1.

[3] SCI state check, ID reception and comparison:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID. If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0. If the data is this station's ID, clear the RDRF flag to 0.

[4] SCI state check and data reception:
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.

[5] Receive error processing and break detection:
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER and FER flags are both cleared to 0. Reception cannot be resumed if either of these flags is set to 1. In the case of a framing error, a break can be detected by reading the RxD pin value.

Figure 14.13 Sample Multiprocessor Serial Reception Flowchart (1)

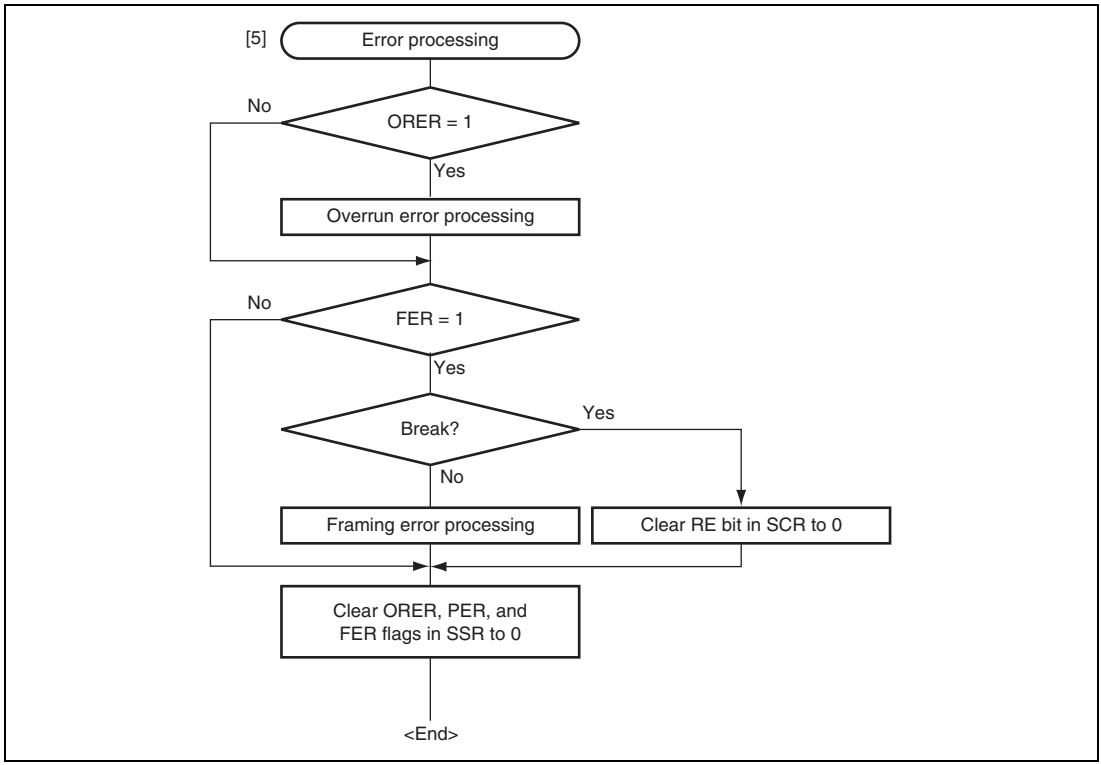
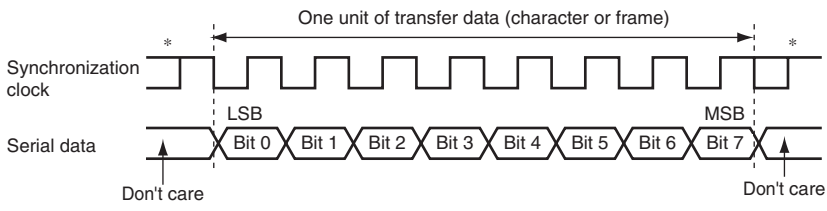


Figure 14.13 Sample Multiprocessor Serial Reception Flowchart (2)

14.6 Operation in Clocked Synchronous Mode

Figure 14.14 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses. One character in transfer data consists of 8-bit data. In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the MSB output state. In clocked synchronous mode, no parity bit or multiprocessor bit is added. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that the next transmit data can be written during transmission or the previous receive data can be read during reception, enabling continuous data transfer.



Note: * Holds a high level except during continuous transfer.

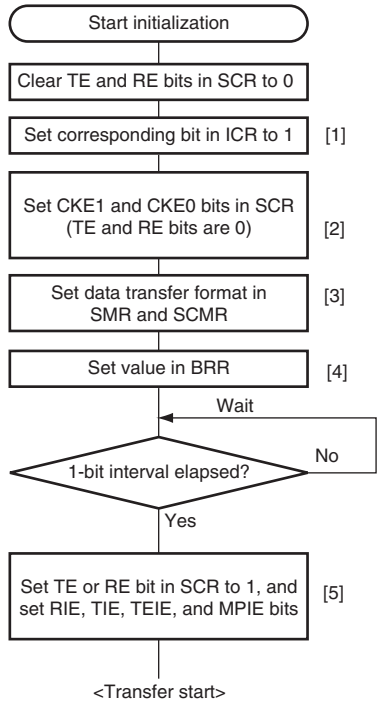
Figure 14.14 Data Format in Clocked Synchronous Communication (LSB-First)

14.6.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCR. When the SCI is operated on an internal clock, the synchronization clock is output from the SCK pin. Eight synchronization clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. Note that in the case of reception only, the synchronization clock is output until an overrun error occurs or until the RE bit is cleared to 0.

14.6.2 SCI Initialization (Clocked Synchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 14.15. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. However, clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR.



[1] Set the bit in ICR for the corresponding pin when receiving data or using an external clock.

[2] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

[3] Set the data transfer format in SMR and SCMR.

[4] Write a value corresponding to the bit rate to BRR. This step is not necessary if an external clock is used.

[5] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE TEIE, and MPIE bits. Setting the TE and RE bits enables the TxD and RxD pins to be used.

Note: In simultaneous transmit and receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

Figure 14.15 Sample SCI Initialization Flowchart

14.6.3 Serial Data Transmission (Clocked Synchronous Mode)

Figure 14.16 shows an example of the operation for transmission in clocked synchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. 8-bit data is sent from the TxD pin synchronized with the output clock when clock output mode has been specified and synchronized with the input clock when use of an external clock has been specified.
4. The SCI checks the TDRE flag at the timing for sending the last bit.
5. If the TDRE flag is cleared to 0, the next transmit data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TxD pin retains the output state of the last bit. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated. The SCK pin is fixed high.

Figure 14.17 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.

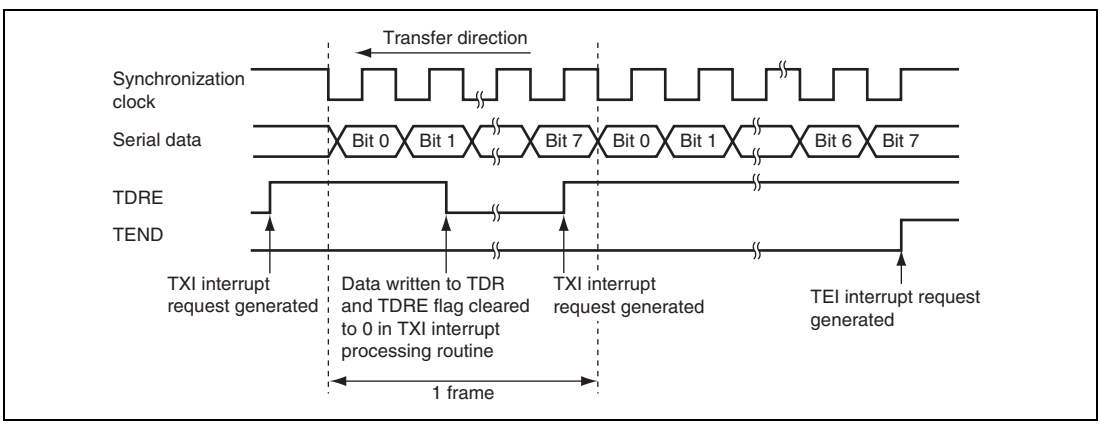
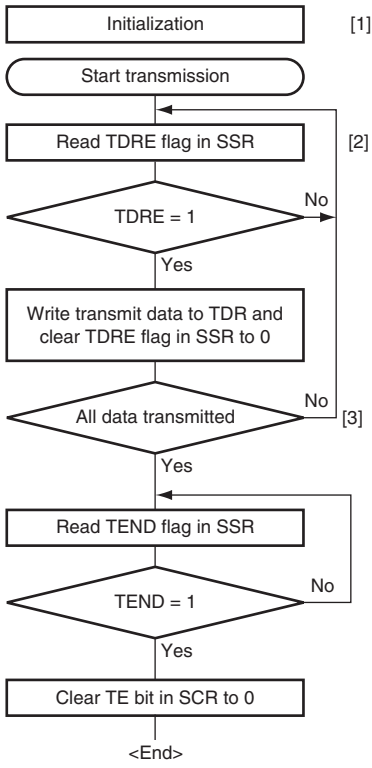


Figure 14.16 Example of Operation for Transmission in Clocked Synchronous Mode



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.

[2] SCI state check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DMAC or DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

Figure 14.17 Sample Serial Transmission Flowchart

14.6.4 Serial Data Reception (Clocked Synchronous Mode)

Figure 14.18 shows an example of SCI operation for reception in clocked synchronous mode. In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with a synchronization clock input or output, starts receiving data, and stores the receive data in RSR.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.

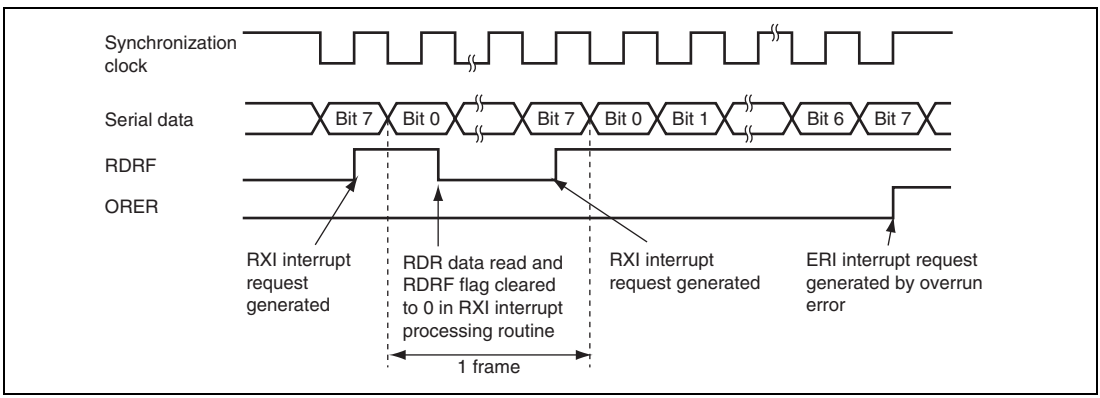


Figure 14.18 Example of Operation for Reception in Clocked Synchronous Mode

Transfer cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 14.19 shows a sample flowchart for serial data reception.

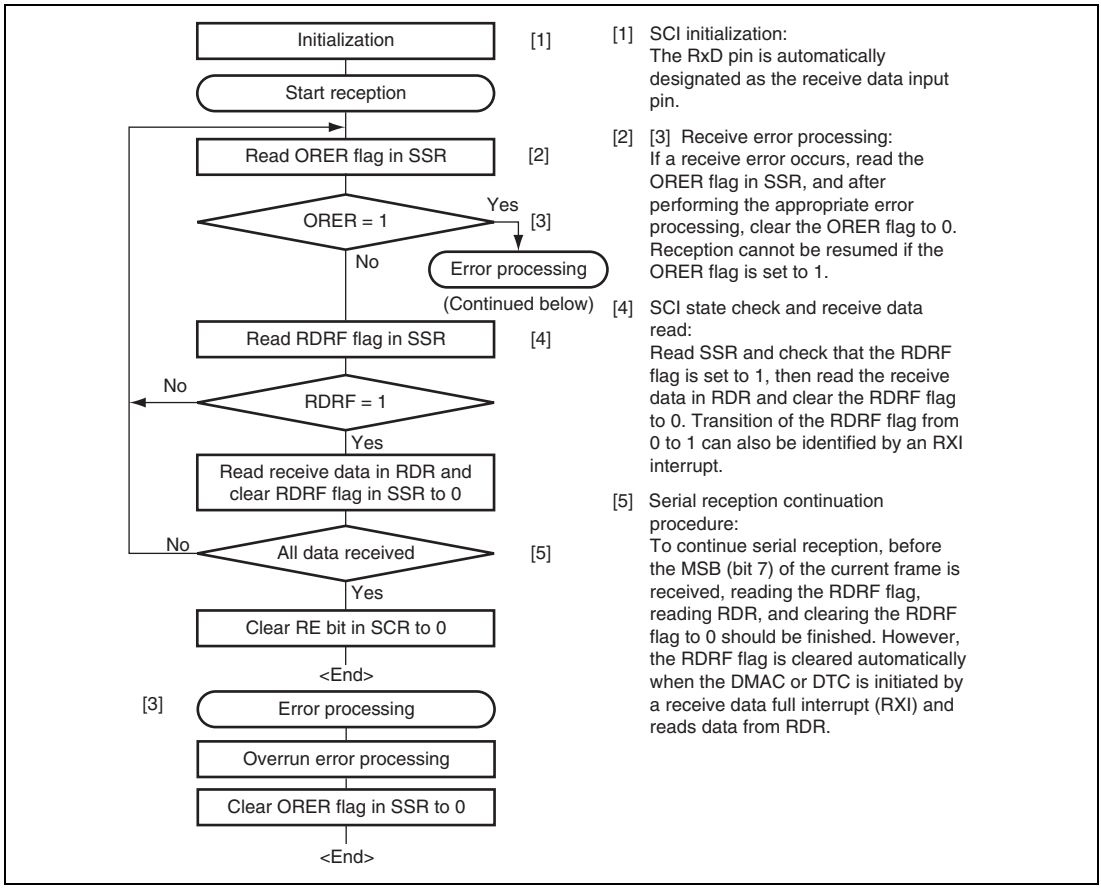
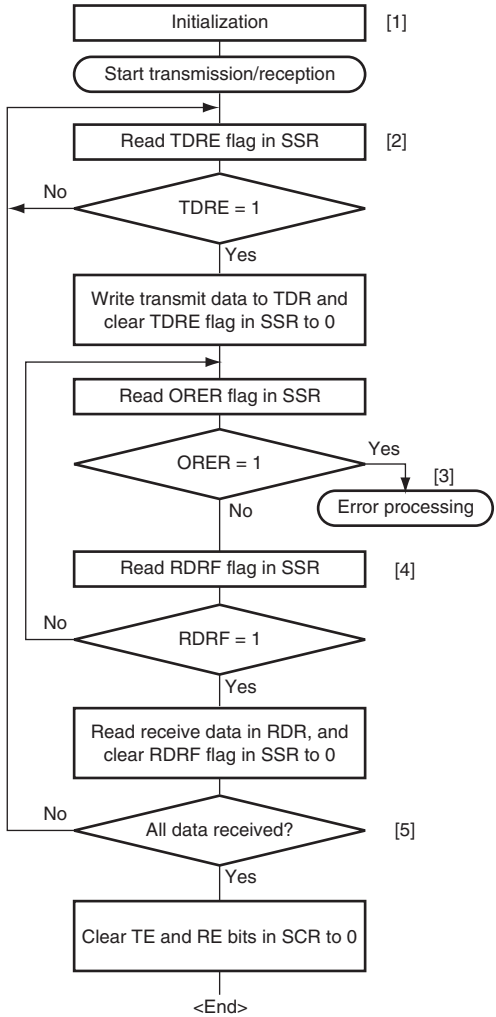


Figure 14.19 Sample Serial Reception Flowchart

14.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode)

Figure 14.20 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags are set to 1, clear the TE bit to 0. Then simultaneously set both the TE and RE bits to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI has finished reception, clear the RE bit to 0. Then after checking that the RDRF bit and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set both the TE and RE bits to 1 with a single instruction.



[1] SCI initialization:
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI state check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Reception cannot be resumed if the ORER flag is set to 1.

[4] SCI state check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/ reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.
However, the TDRE flag is checked and cleared automatically when the DMAC or DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR. Similarly, the RDRF flag is cleared automatically when the DMAC or DTC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

Figure 14.20 Sample Flowchart of Simultaneous Serial Transmission and Reception

14.7 Operation in Smart Card Interface Mode

The SCI supports the IC card (smart card) interface, supporting the ISO/IEC 7816-3 (Identification Card) standard, as an extended serial communication interface function. Smart card interface mode can be selected using the appropriate register.

14.7.1 Sample Connection

Figure 14.21 shows a sample connection between the smart card and this LSI. As in the figure, since this LSI communicates with the IC card using a single transmission line, interconnect the TxD and RxD pins and pull up the data transmission line to V_{CC} using a resistor. Setting the RE and TE bits to 1 with the IC card not connected enables closed transmission/reception allowing self diagnosis. To supply the IC card with the clock pulses generated by the SCI, input the SCK pin output to the CLK pin of the IC card. A reset signal can be supplied via the output port of this LSI.

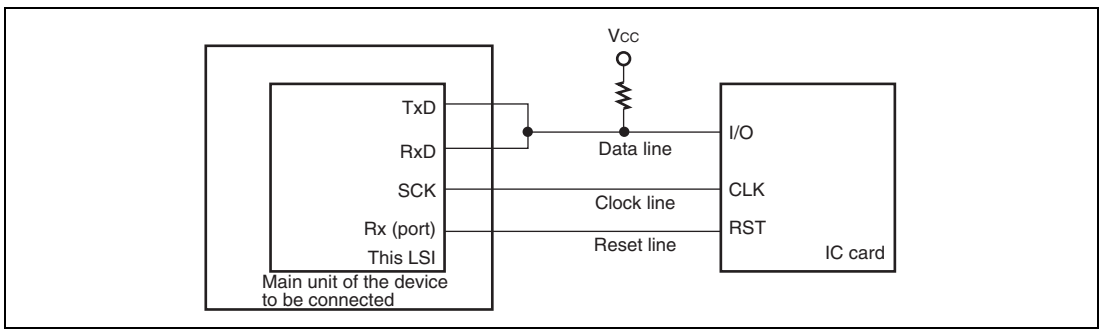


Figure 14.21 Pin Connection for Smart Card Interface

14.7.2 Data Format (Except in Block Transfer Mode)

Figure 14.22 shows the data transfer formats in smart card interface mode.

- One frame contains 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etu (elementary time unit: time required for transferring one bit) is secured as a guard time after the end of the parity bit before the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etu has passed from the start bit.
- If an error signal is sampled during transmission, the same data is automatically re-transmitted after at least 2 etu.

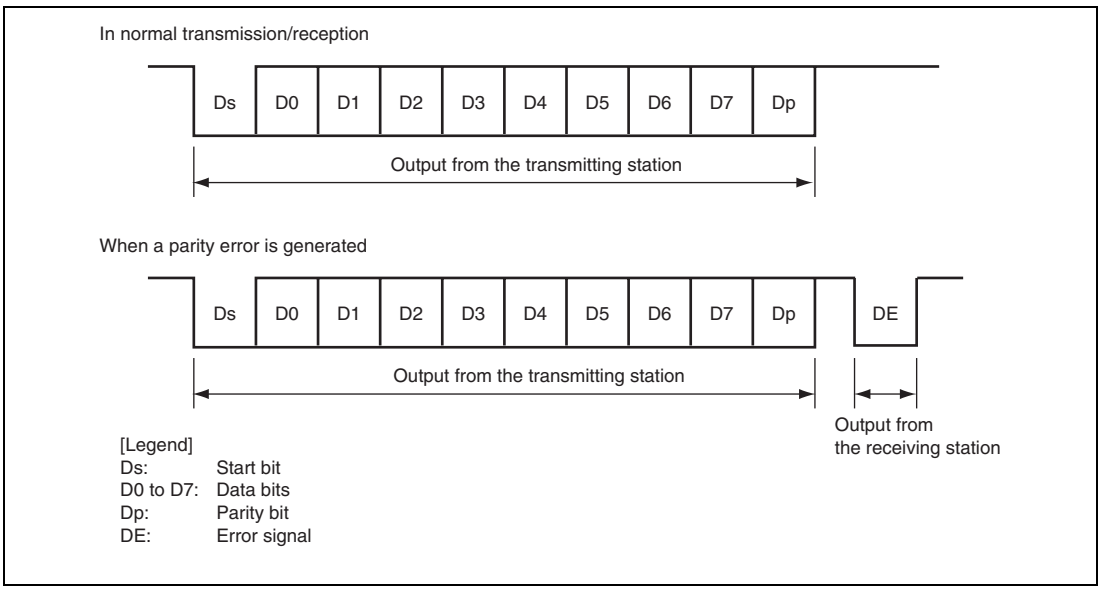


Figure 14.22 Data Formats in Normal Smart Card Interface Mode

For communication with the IC cards of the direct convention and inverse convention types, follow the procedure below.

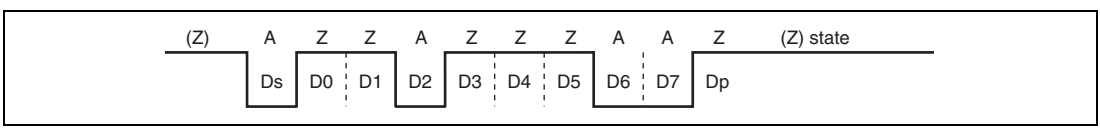


Figure 14.23 Direct Convention (SDIR = SINV = O/$\overline{E}$ = 0)

For the direct convention type, logic levels 1 and 0 correspond to states Z and A, respectively, and data is transferred with LSB-first as the start character, as shown in figure 14.23. Therefore, data in the start character in the figure is H'3B. When using the direct convention type, write 0 to both the SDIR and SINV bits in SCMR. Write 0 to the O/$\overline{E}$ bit in SMR in order to use even parity, which is prescribed by the smart card standard.

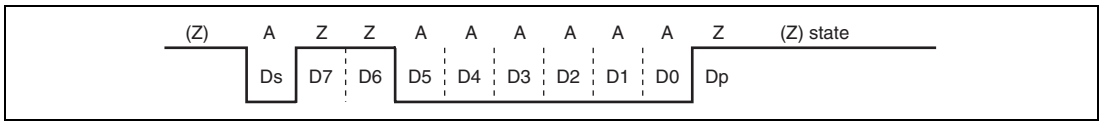


Figure 14.24 Inverse Convention (SDIR = SINV = O/$\overline{E}$ = 1)

For the inverse convention type, logic levels 1 and 0 correspond to states A and Z, respectively and data is transferred with MSB-first as the start character, as shown in figure 14.24. Therefore, data in the start character in the figure is H'3F. When using the inverse convention type, write 1 to both the SDIR and SINV bits in SCMR. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to state Z. Since the SNIV bit of this LSI only inverts data bits D7 to D0, write 1 to the O/$\overline{E}$ bit in SMR to invert the parity bit in both transmission and reception.

14.7.3 Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following respects.

- Even if a parity error is detected during reception, no error signal is output. Since the PER bit in SSR is set by error detection, clear the PER bit before receiving the parity bit of the next frame.
- During transmission, at least 1 etu is secured as a guard time after the end of the parity bit before the start of the next frame.
- Since the same data is not re-transmitted during transmission, the TEND flag is set 11.5 etu after transmission start.
- Although the ERS flag in block transfer mode displays the error signal status as in normal smart card interface mode, the flag is always read as 0 because no error signal is transferred.

14.7.4 Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the on-chip baud rate generator can be used as a transfer clock in smart card interface mode. In this mode, the SCI can operate on a basic clock with a frequency of 32, 64, 372, or 256 times the bit rate according to the BCP1 and BCP0 bit settings (the frequency is always 16 times the bit rate in normal asynchronous mode). At reception, the falling edge of the start bit is sampled using the basic clock in order to perform internal synchronization. Receive data is sampled on the 16th, 32nd, 186th and 128th rising edges of the basic clock so that it can be latched at the middle of each bit as shown in figure 14.25. The reception margin here is determined by the following formula.

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

M: Reception margin (%)
N: Ratio of bit rate to clock (N = 32, 64, 372, 256)
D: Duty cycle of clock (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock frequency deviation

Assuming values of F = 0, D = 0.5, and N = 372 in the above formula, the reception margin is determined by the formula below.

$$M = \left(0.5 - \frac{1}{2 \times 372}\right) \times 100\% = 49.866\%$$

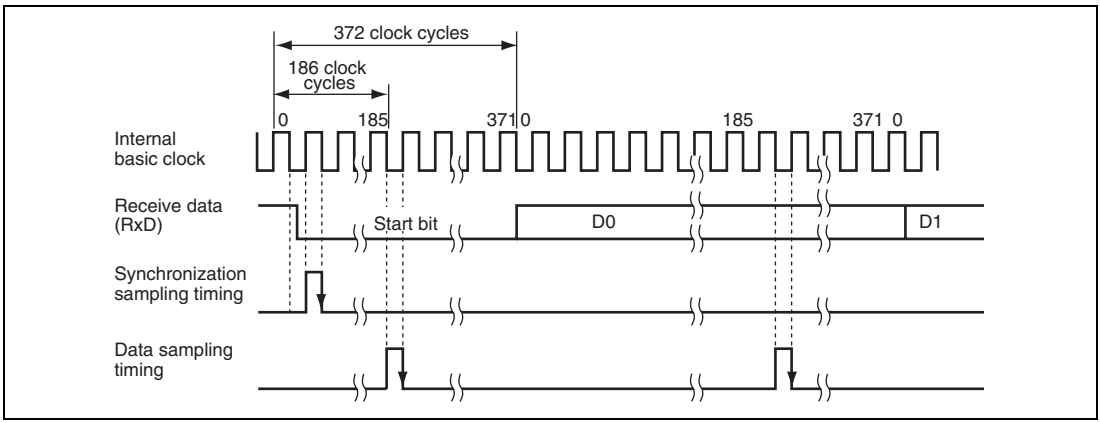


Figure 14.25 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency is 372 Times the Bit Rate)

14.7.5 Initialization

Before transmitting and receiving data, initialize the SCI using the following procedure. Initialization is also necessary before switching from transmission to reception and vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Set the ICR bit of the corresponding pin to 1.
3. Clear the error flags ERS, PER, and ORER in SSR to 0.
4. Set the GM, BLK, O/$\overline{E}$, BCP1, BCP0, CKS1, and CKS0 bits in SMR appropriately. Also set the PE bit to 1.
5. Set the SMIF, SDIR, and SINV bits in SCMR appropriately. When the DDR corresponding to the TxD pin is cleared to 0, the TxD and RxD pins are changed from port pins to SCI pins, placing the pins into high impedance state.
6. Set the value corresponding to the bit rate in BRR.
7. Set the CKE1 and CKE0 bits in SCR appropriately. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0 simultaneously.

   When the CKE0 bit is set to 1, the SCK pin is allowed to output clock pulses.
8. Set the TIE, RIE, TE, and RE bits in SCR appropriately after waiting for at least a 1-bit interval. Setting the TE and RE bits to 1 simultaneously is prohibited except for self diagnosis.

To switch from reception to transmission, first verify that reception has completed, then initialize the SCI. At the end of initialization, RE and TE should be set to 0 and 1, respectively. Reception completion can be verified by reading the RDRF, PER, or ORER flag. To switch from transmission to reception, first verify that transmission has completed, then initialize the SCI. At the end of initialization, TE and RE should be set to 0 and 1, respectively. Transmission completion can be verified by reading the TEND flag.

14.7.6 Data Transmission (Except in Block Transfer Mode)

Data transmission in smart card interface mode (except in block transfer mode) is different from that in normal serial communication interface mode in that an error signal is sampled and data can be re-transmitted. Figure 14.26 shows the data re-transfer operation during transmission.

1. If an error signal from the receiving end is sampled after one frame of data has been transmitted, the ERS bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the ERS bit to 0 before the next parity bit is sampled.
2. For the frame in which an error signal is received, the TEND bit in SSR is not set to 1. Data is re-transferred from TDR to TSR allowing automatic data retransmission.
3. If no error signal is returned from the receiving end, the ERS bit in SSR is not set to 1.
4. In this case, one frame of data is determined to have been transmitted including re-transfer, and the TEND bit in SSR is set to 1. Here, a TXI interrupt request is generated if the TIE bit in SCR is set to 1. Writing transmit data to TDR starts transmission of the next data.

Figure 14.28 shows a sample flowchart for transmission. All the processing steps are automatically performed using a TXI interrupt request to activate the DMAC or DTC. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request if the TIE bit in SCR has been set to 1. This activates the DMAC or DTC by a TXI request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DMAC or DTC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DMAC or DTC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DMAC or DTC. Therefore, the SCI and DMAC or DTC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DMAC or DTC, be sure to set and enable the DMAC or DTC prior to making SCI settings. For DMAC settings, see section 7, DMA Controller (DMAC), and for DTC settings, see section 8, Data Transfer Controller (DTC).

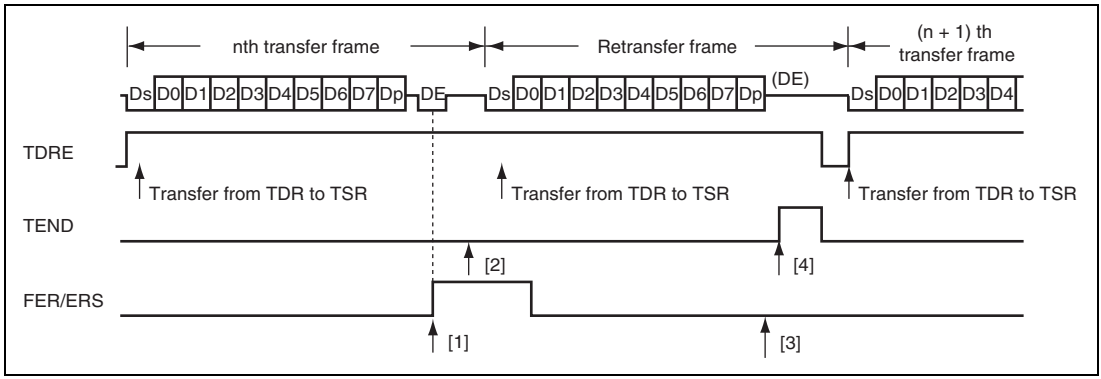


Figure 14.26 Data Re-Transfer Operation in SCI Transmission Mode

Note that the TEND flag is set in different timings depending on the GM bit setting in SMR. Figure 14.27 shows the TEND flag set timing.

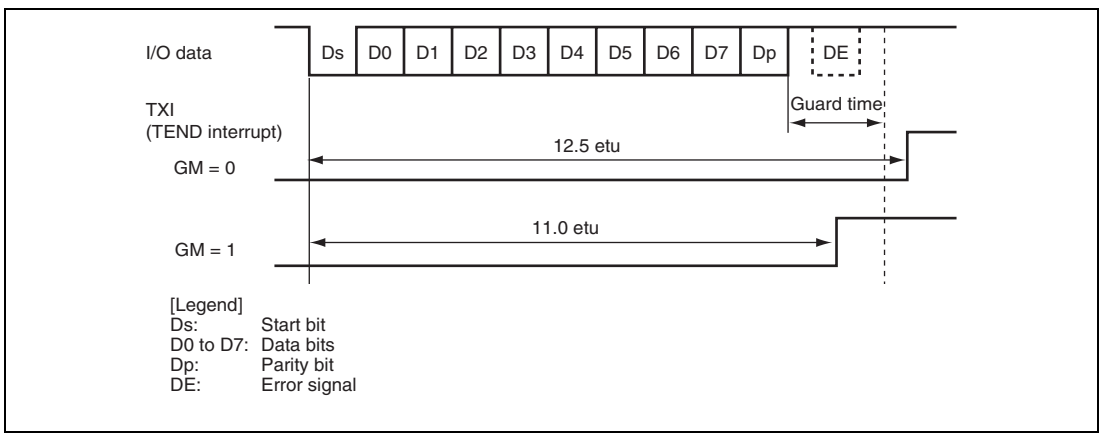


Figure 14.27 TEND Flag Set Timing during Transmission

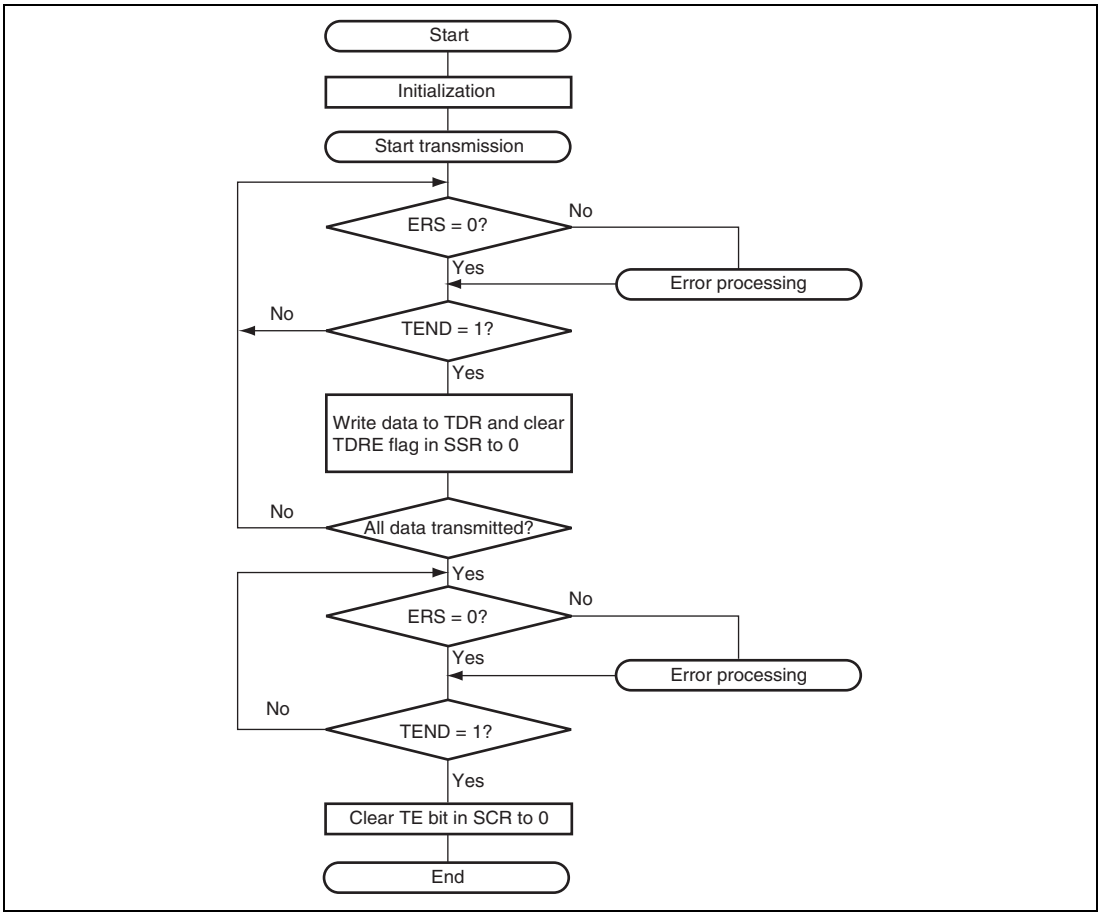


Figure 14.28 Sample Transmission Flowchart

14.7.7 Serial Data Reception (Except in Block Transfer Mode)

Data reception in smart card interface mode is similar to that in normal serial communication interface mode. Figure 14.29 shows the data re-transfer operation during reception.

1. If a parity error is detected in receive data, the PER bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the PER bit to 0 before the next parity bit is sampled.
2. For the frame in which a parity error is detected, the RDRF bit in SSR is not set to 1.
3. If no parity error is detected, the PER bit in SSR is not set to 1.
4. In this case, data is determined to have been received successfully, and the RDRF bit in SSR is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set to 1.

Figure 14.30 shows a sample flowchart for reception. All the processing steps are automatically performed using an RXI interrupt request to activate the DMAC or DTC. In reception, setting the RIE bit to 1 allows an RXI interrupt request to be generated when the RDRF flag is set to 1. This activates the DMAC or DTC by an RXI request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DMAC or DTC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC or DTC. If an error occurs during reception, i.e., either the ORER or PER flag is set to 1, a transmit/receive error interrupt (ERI) request is generated and the error flag must be cleared. If an error occurs, the DMAC or DTC is not activated and receive data is skipped, therefore, the number of bytes of receive data specified in the DMAC or DTC is transferred. Even if a parity error occurs and the PER bit is set to 1 in reception, receive data is transferred to RDR, thus allowing the data to be read.

Note: For operations in block transfer mode, see section 14.4, Operation in Asynchronous Mode.

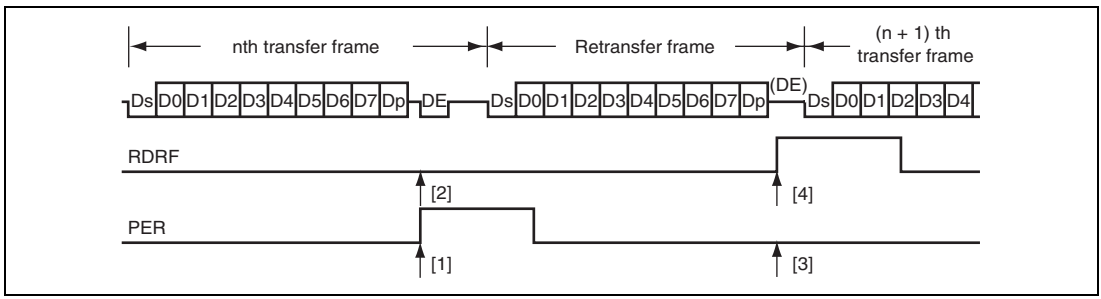


Figure 14.29 Data Re-Transfer Operation in SCI Reception Mode

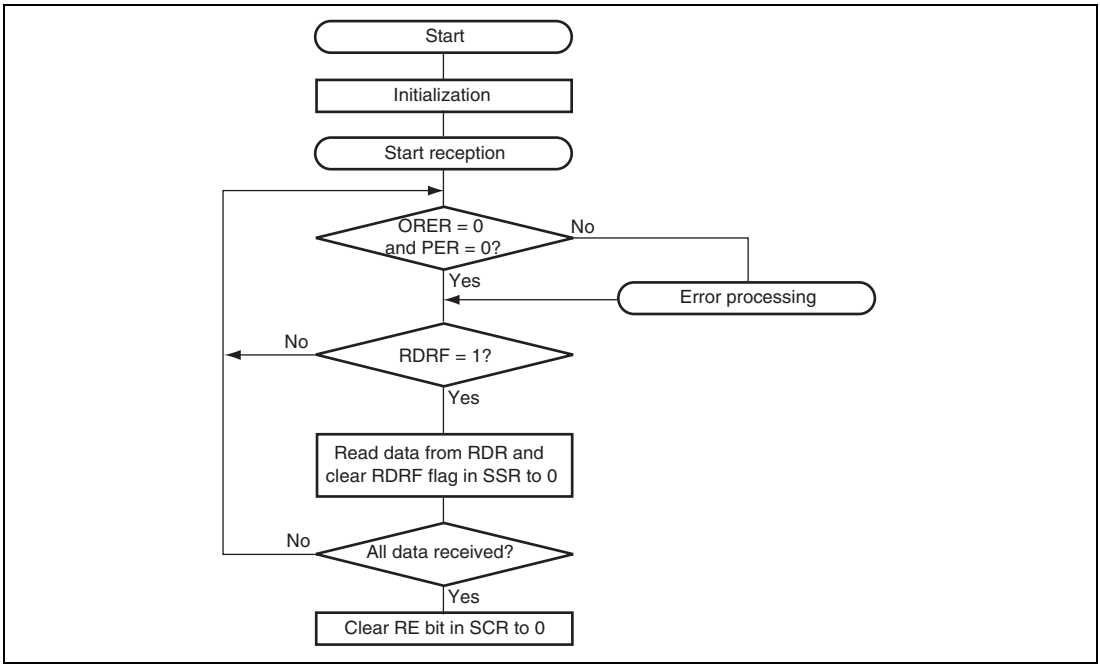


Figure 14.30 Sample Reception Flowchart

14.7.8 Clock Output Control

Clock output can be fixed using the CKE1 and CKE0 bits in SCR when the GM bit in SMR is set to 1. Specifically, the minimum width of a clock pulse can be specified.

Figure 14.31 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 0.

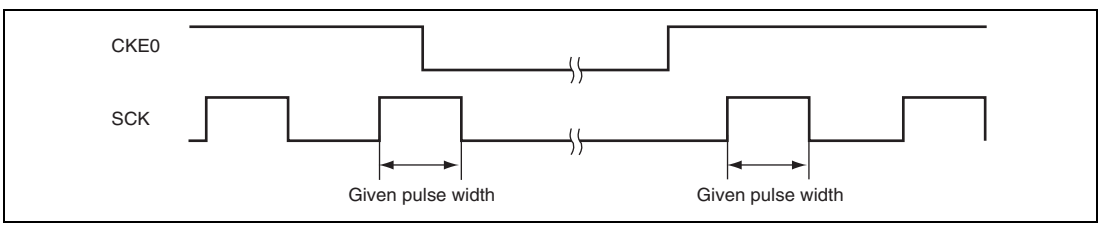


Figure 14.31 Clock Output Fixing Timing

At power-on and transitions to/from software standby mode, use the following procedure to secure the appropriate clock duty cycle.

- At power-on

  To secure the appropriate clock duty cycle simultaneously with power-on, use the following procedure.

  1. Initially, port input is enabled in the high-impedance state. To fix the potential level, use a pull-up or pull-down resistor.
  2. Fix the SCK pin to the specified output using the CKE1 bit in SCR.
  3. Set SMR and SCMR to enable smart card interface mode.

     Set the CKE0 bit in SCR to 1 to start clock output.

- At mode switching

  — At transition from smart card interface mode to software standby mode

  1. Set the data register (DR) and data direction register (DDR) corresponding to the SCK pin to the values for the output fixed state in software standby mode.
  2. Write 0 to the TE and RE bits in SCR to stop transmission/reception. Simultaneously, set the CKE1 bit to the value for the output fixed state in software standby mode.
  3. Write 0 to the CKE0 bit in SCR to stop the clock.
  4. Wait for one cycle of the serial clock. In the mean time, the clock output is fixed to the specified level with the duty cycle retained.
  5. Make the transition to software standby mode.

  — At transition from smart card interface mode to software standby mode

  1. Clear software standby mode.
  2. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty cycle is then generated.

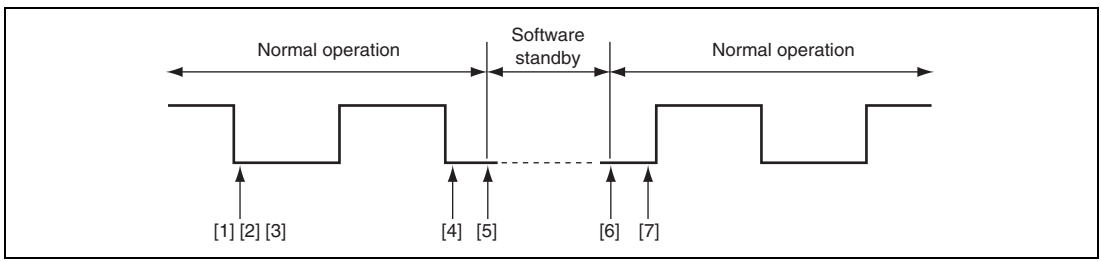


Figure 14.32 Clock Stop and Restart Procedure

14.8 Interrupt Sources

14.8.1 Interrupts in Normal Serial Communication Interface Mode

Table 14.12 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt request can activate the DMAC or DTC to allow data transfer. The TDRE flag is automatically cleared to 0 at data transfer by the DMAC or DTC.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DMAC or DTC to allow data transfer. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC or DTC.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared to 0 simultaneously by the TXI interrupt processing routine, the SCI cannot branch to the TEI interrupt processing routine later.

Table 14.12 SCI Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation	Priority
ERI	Receive error	ORER, FER, or PER	Not possible	Not possible	High
RXI	Receive data full	RDRF	Possible	Possible	
TXI	Transmit data empty	TDRE	Possible	Possible	
TEI	Transmit end	TEND	Not possible	Not possible	Low

14.8.2 Interrupts in Smart Card Interface Mode

Table 14.13 shows the interrupt sources in smart card interface mode. A transmit end (TEI) interrupt request cannot be used in this mode.

Table 14.13 SCI Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation	Priority
ERI	Receive error or error signal detection	ORER, PER, or ERS	Not possible	Not possible	High
RXI	Receive data full	RDRF	Possible	Possible	
TXI	Transmit data empty	TDRE	Possible	Possible	Low

Data transmission/reception using the DMAC or DTC is also possible in smart card interface mode, similar to in the normal SCI mode. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt. This activates the DMAC or DTC by a TXI request thus allowing transfer of transmit data if the TXI request is specified as a source of DMAC or DTC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DMAC or DTC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DMAC or DTC. Therefore, the SCI and DMAC or DTC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag in SSR, which is set at error occurrence, is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit in SCR to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DMAC or DTC, be sure to set and enable the DMAC or DTC prior to making SCI settings. For DMAC settings, see section 7, DMA Controller (DMAC), and for DTC settings, see section 8, Data Transfer Controller (DTC).

In reception, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. This activates the DMAC or DTC by an RXI request thus allowing transfer of receive data if the RXI request is specified as a source of DMAC or DTC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC or DTC. If an error occurs, the RDRF flag is not set but the error flag is set. Therefore, the DMAC or DTC is not activated and an ERI interrupt request is issued to the CPU instead; the error flag must be cleared.

14.9 Usage Notes

14.9.1 Module Stop State Setting

Operation of the SCI can be disabled or enabled using the module stop control register. The initial setting is for operation of the SCI to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 20, Power-Down Modes.

14.9.2 Break Detection and Processing

When framing error detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set, and the PER flag may also be set. Note that, since the SCI continues the receive operation even after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

14.9.3 Mark State and Break Detection

When the TE bit is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line in mark state (the state of 1) until TE is set to 1, set both DDR and DR to 1. Since the TE bit is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set DDR to 1 and DR to 0, and then clear the TE bit to 0. When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

14.9.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that the receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

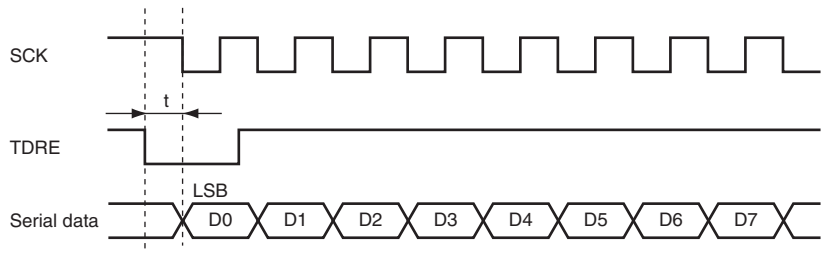
14.9.5 Relation between Writing to TDR and TDRE Flag

The TDRE flag in SSR is a status flag which indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR irrespective of the TDRE flag status. However, if new data is written to TDR when the TDRE flag is 0, that is, when the previous data has not been transferred to TSR yet, the previous data in TDR is lost. Be sure to write transmit data to TDR after verifying that the TDRE flag is set to 1.

14.9.6 Restrictions on Using DMAC or DTC

1. When the external clock source is used as a synchronization clock, update TDR by the DMAC or DTC and wait for at least five Pϕ clock cycles before allowing the transmit clock to be input. If the transmit clock is input within four clock cycles after TDR modification, the SCI may malfunction (figure 14.33).
2. When using the DMAC or DTC to read RDR, be sure to set the receive end interrupt (RXI) as the DMAC or DTC activation source.



Note: When external clock is supplied, t must be more than four clock cycles.

Figure 14.33 Sample Transmission using DTC in Clocked Synchronous Mode

14.9.7 Operations in Power-Down State

(1) Transmission

Before specifying the module stop state or making a transition to software standby mode, stop the transmit operations (TE = TIE = TEIE = 0). TSR, TDR, and SSR are reset. The states of the output pins in the module stop state or in software standby mode depend on the port settings, and the pins output a high-level signal after cancellation. If the transition is made during data transmission, the data being transmitted will be undefined.

To transmit data in the same transmission mode after cancellation of the power-down state, set the TE bit to 1, read SSR, write to TDR, clear TDRE in this order, and then start transmission. To transmit data in a different transmission mode, initialize the SCI first.

Figure 14.34 shows a sample flowchart for transition to software standby mode during transmission. Figures 14.35 and 14.36 show the port pin states in transition to software standby mode.

Before specifying the module stop state or making a transition to software standby mode from the transmission mode using DTC transfer, stop all transmit operations (TE = TIE = TEIE = 0). Setting the TE and TIE bits to 1 after cancellation sets the TXI flag to start transmission using the DTC.

(2) Reception

Before specifying the module stop state or making a transition to software standby mode, stop the receive operations (RE = 0). RSR, RDR, and SSR are reset. If transition is made during data reception, the data being received will be invalid.

To receive data in the same reception mode after cancellation of the power-down state, set the RE bit to 1, and then start reception. To receive data in a different reception mode, initialize the SCI first.

Figure 14.37 shows a sample flowchart for transition to software standby mode during reception.

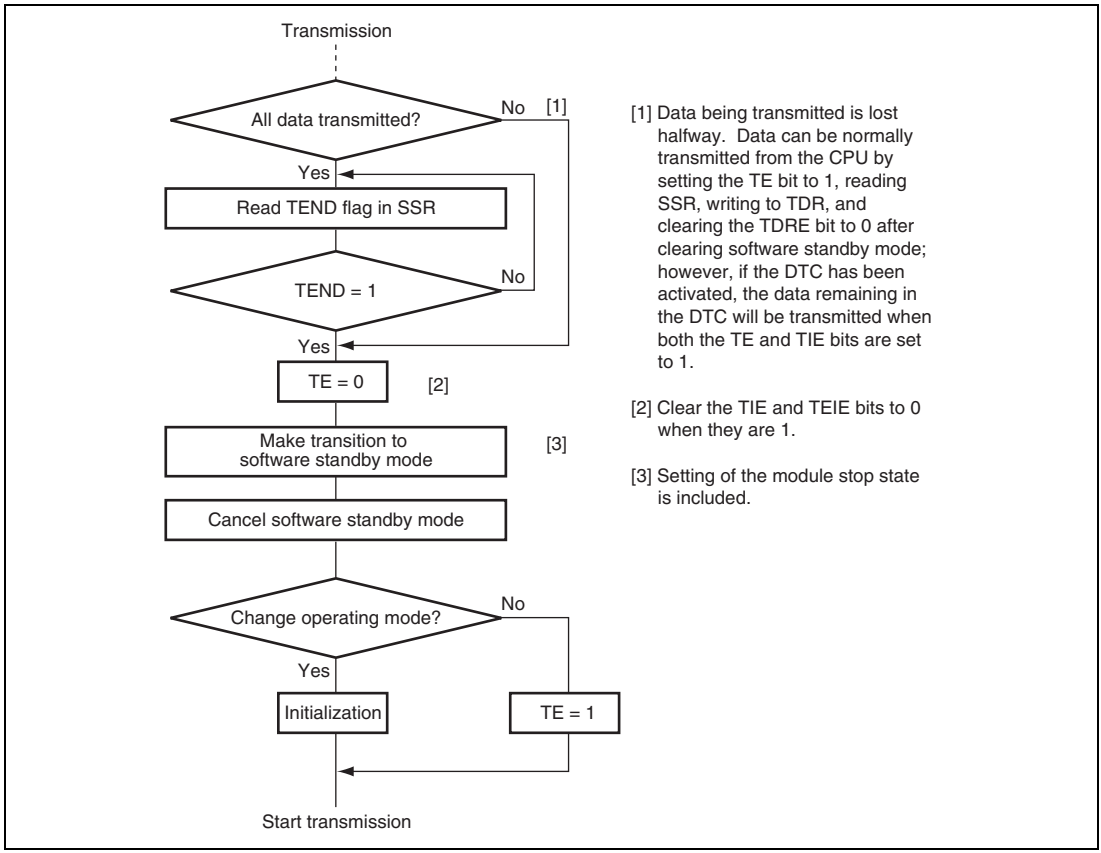


Figure 14.34 Sample Flowchart of Transition to Software Standby Mode in Transmission

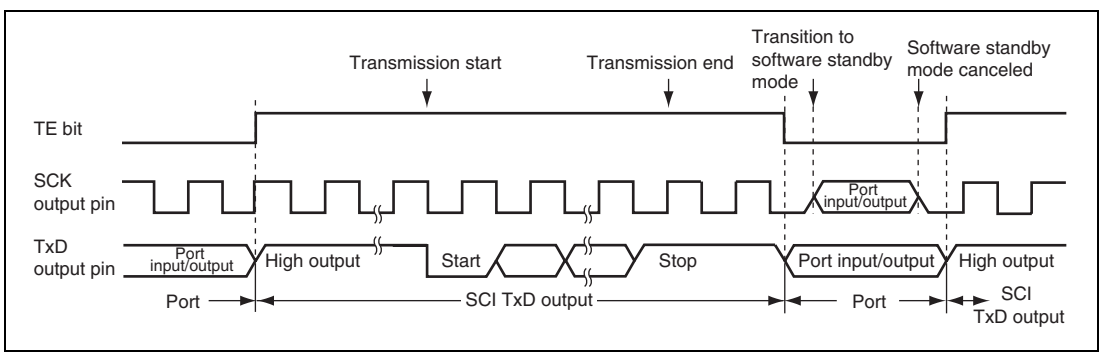


Figure 14.35 Port Pin States during Transition to Software Standby Mode (Internal Clock, Asynchronous Transmission)

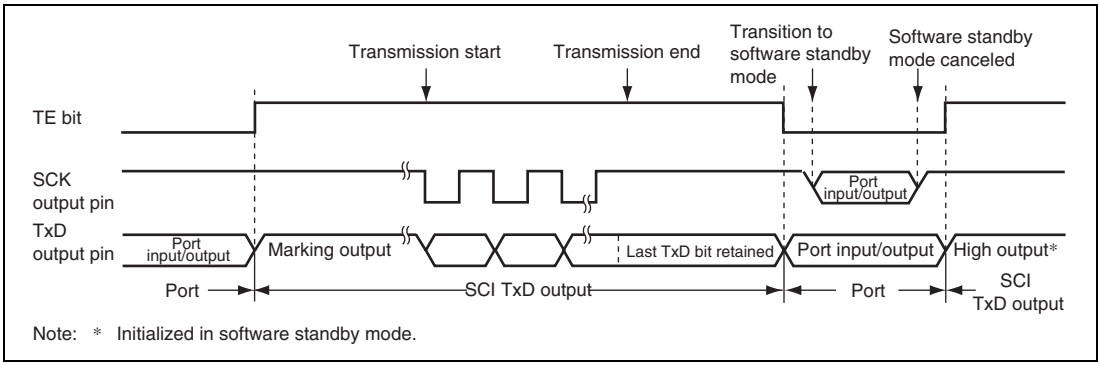


Figure 14.36 Port Pin States during Transition to Software Standby Mode (Internal Clock, Clocked Synchronous Transmission)

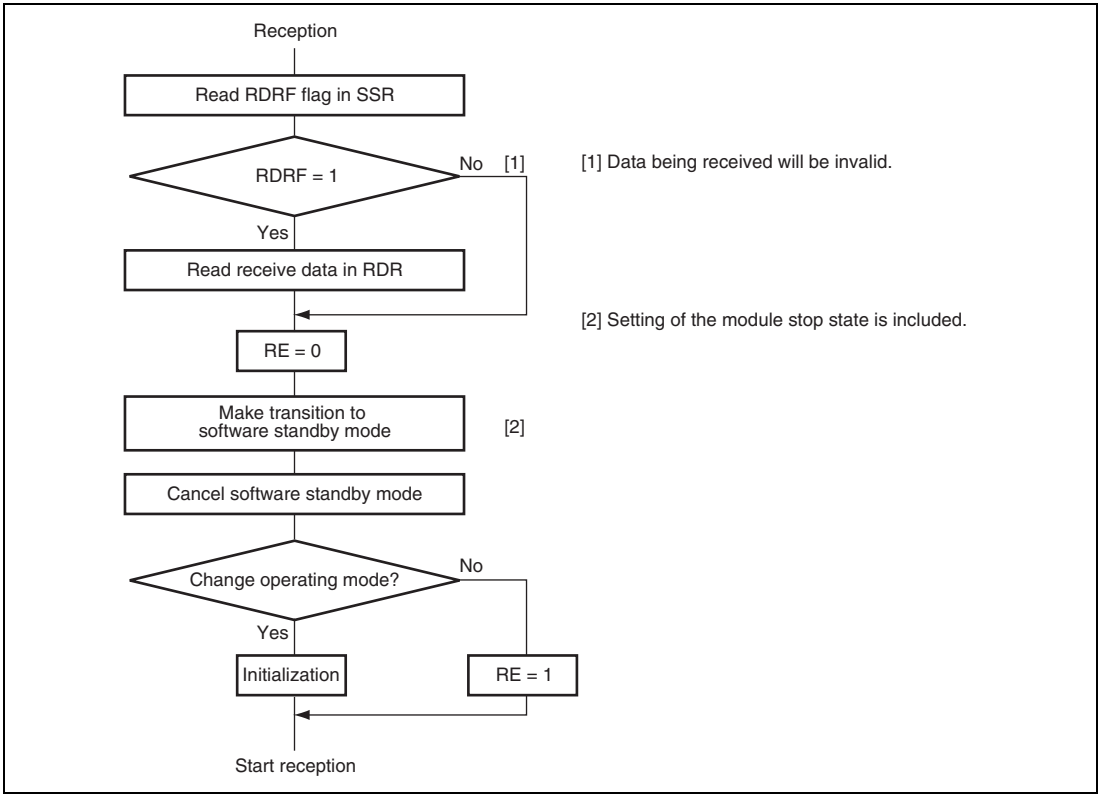


Figure 14.37 Sample Flowchart of Transition to Software Standby Mode in Reception

Section 15 A/D Converter

This LSI includes a successive approximation type 10-bit A/D converter that allows up to eight analog input channels to be selected.

Figure 15.1 shows a block diagram of the A/D converter.

15.1 Features

- 10-bit resolution
- Eight input channels
- Conversion time: 7.4 μs per channel (at 35-MHz operation)
- Two kinds of operating modes
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels, or 1 to 8 channels
- Eight data registers
  A/D conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three types of conversion start
  Conversion can be started by software, a conversion start trigger by the 16-bit timer pulse unit (TPU) or 8-bit timer (TMR), or an external trigger signal.
- Interrupt source
  A/D conversion end interrupt (ADI) request can be generated.
- Module stop state specifiable

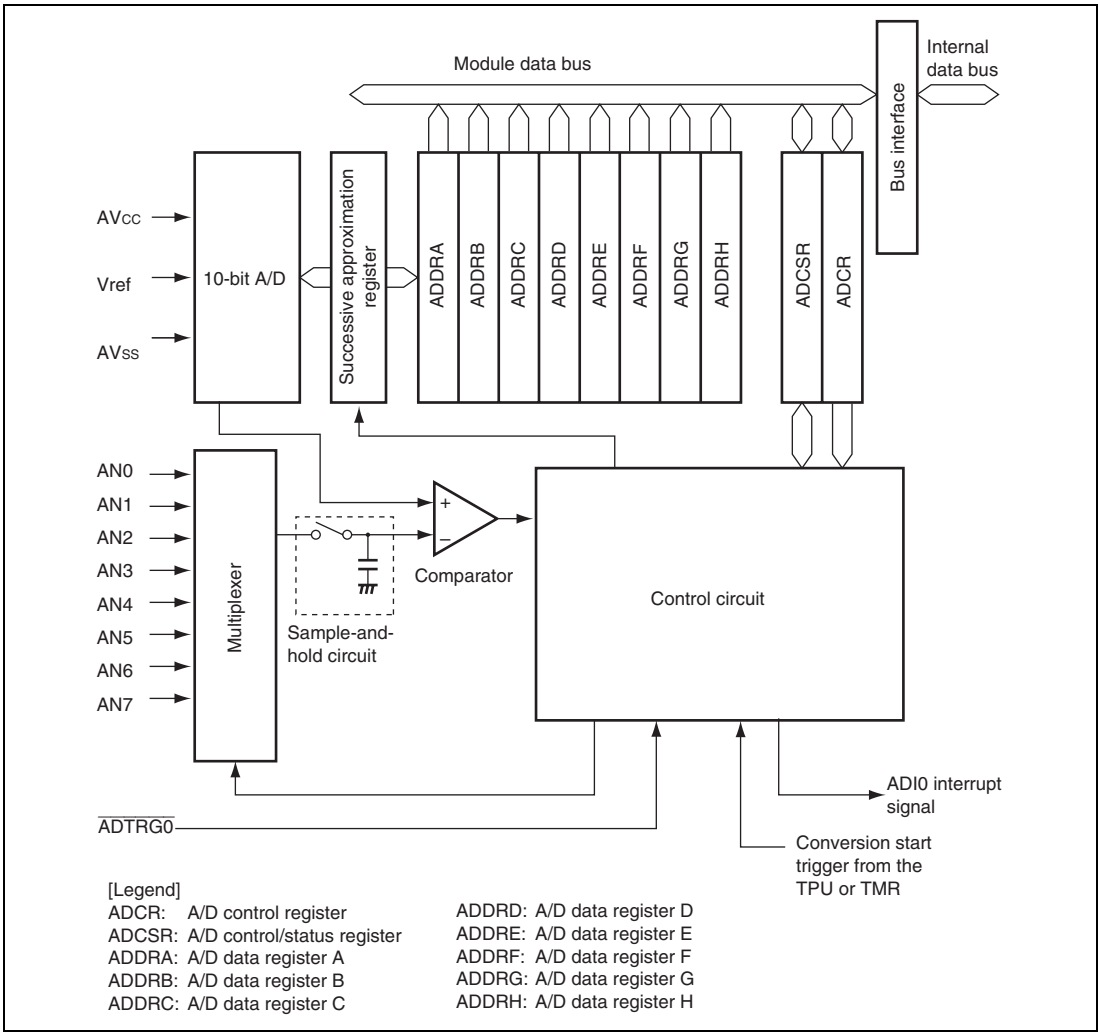


Figure 15.1 Block Diagram of A/D Converter

15.2 Input/Output Pins

Table 15.1 shows the pin configuration of the A/D converter.

Table 15.1 Pin Configuration

Pin Name	Symbol	I/O	Function
Analog input pin 0	AN0	Input	Analog inputs
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin 3	AN3	Input	
Analog input pin 4	AN4	Input	
Analog input pin 5	AN5	Input	
Analog input pin 6	AN6	Input	
Analog input pin 7	AN7	Input	
A/D external trigger input pin	$\overline{\text{ADTRG0}}$	Input	External trigger input for starting A/D conversion
Analog power supply pin	AV_{CC}	Input	Analog block power supply
Analog ground pin	AV_{SS}	Input	Analog block ground
Reference voltage pin	Vref	Input	A/D conversion reference voltage

15.3 Register Descriptions

The A/D converter has the following registers.

- A/D data register A (ADDRA)
- A/D data register B (ADDRB)
- A/D data register C (ADDRC)
- A/D data register D (ADDRD)
- A/D data register E (ADDRE)
- A/D data register F (ADDRF)
- A/D data register G (ADDRG)
- A/D data register H (ADDRH)
- A/D control/status register (ADCSR)
- A/D control register (ADCR)

15.3.1 A/D Data Registers A to H (ADDRA to ADDRH)

There are eight 16-bit read-only ADDR registers, ADDRA to ADDRH, used to store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 15.2.

The converted 10-bit data is stored in bits 15 to 6. The lower 6-bit data is always read as 0.

The data bus between the CPU and the A/D converter has a 16-bit width. The data can be read directly from the CPU. ADDR must not be accessed in 8-bit units and must be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Name											—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15.2 Analog Input Channels and Corresponding ADDR Registers

Analog Input Channel	A/D Data Register Which Stores Conversion Result
AN0	ADDRA
AN1	ADDRB
AN2	ADDRC
AN3	ADDRD
AN4	ADDRE
AN5	ADDRF
AN6	ADDRG
AN7	ADDRH

15.3.2 A/D Control/Status Register (ADCSR)

ADCSR controls A/D conversion operations.

Bit	7	6	5	4	3	2	1	0
Bit Name	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)*	R/W	R/W	R	R/W	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W)*	A/D End Flag A status flag that indicates the end of A/D conversion. [Setting conditions] • When A/D conversion ends in single mode • When A/D conversion ends on all specified channels in scan mode [Clearing conditions] • When 0 is written after reading ADF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When the DMAC or DTC is activated by an ADI interrupt and ADDR is read
6	ADIE	0	R/W	A/D Interrupt Enable When this bit is set to 1, ADI interrupts by ADF are enabled.
5	ADST	0	R/W	A/D Start Clearing this bit to 0 stops A/D conversion, and the A/D converter enters wait state. Setting this bit to 1 starts A/D conversion. In single mode, this bit is cleared to 0 automatically when A/D conversion on the specified channel ends. In scan mode, A/D conversion continues sequentially on the specified channels until this bit is cleared to 0 by a transition to hardware standby mode.
4	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
3	CH3	0	R/W	Channel Select 3 to 0
2	CH2	0	R/W	Selects analog input together with bits SCANE and SCANS in ADCR.
1	CH1	0	R/W	
0	CH0	0	R/W	• When SCANE = 0 and SCANS = X 0000: AN0 0001: AN1 0010: AN2 0011: AN3 0100: AN4 0101: AN5 0110: AN6 0111: AN7 1XXX: Setting prohibited • When SCANE = 1 and SCANS = 0 0000: AN0 0001: AN0 and AN1 0010: AN0 to AN2 0011: AN0 to AN3 0100: AN4 0101: AN4 and AN5 0110: AN4 to AN6 0111: AN4 to AN7 1XXX: Setting prohibited • When SCANE = 1 and SCANS = 1 0000: AN0 0001: AN0 and AN1 0010: AN0 to AN2 0011: AN0 to AN3 0100: AN0 to AN4 0101: AN0 to AN5 0110: AN0 to AN6 0111: AN0 to AN7 1XXX: Setting prohibited

[Legend]

X: Don't care

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

15.3.3 A/D Control Register (ADCR)

ADCR enables A/D conversion to be started by an external trigger input.

Bit	7	6	5	4	3	2	1	0
Bit Name	TRGS1	TRGS0	SCANE	SCANS	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	R

Bit	Bit Name	Initial Value	R/W	Description
7	TRGS1	0	R/W	Timer Trigger Select 1 and 0
6	TRGS0	0	R/W	These bits select enabling or disabling of the start of A/D conversion by a trigger signal. 00: A/D conversion start by external trigger is disabled 01: A/D conversion start by external trigger from TPU is enabled 10: A/D conversion start by external trigger from TMR is enabled 11: A/D conversion start by the $\overline{\text{ADTRG0}}$ pin is enabled*
5	SCANE	0	R/W	Scan Mode
4	SCANS	0	R/W	These bits select the A/D conversion operating mode. 0X: Single mode 10: Scan mode. A/D conversion is performed continuously for channels 1 to 4. 11: Scan mode. A/D conversion is performed continuously for channels 1 to 8.
3	CKS1	0	R/W	Clock Select 1 and 0
2	CKS0	0	R/W	These bits set the A/D conversion time. Set bits CKS1 and CKS0 only while A/D conversion is stopped (ADST = 0). 00: A/D conversion time = 530 states (max) 01: A/D conversion time = 266 states (max) 10: A/D conversion time = 134 states (max) 11: A/D conversion time = 68 states (max)

Bit	Bit Name	Initial Value	R/W	Description
1, 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

[Legend]

X: Don't care

Note: * To set A/D conversion to start by the $\overline{\text{ADTRG}}$ pin, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, refer to section 9, I/O Ports.

15.4 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes: single mode and scan mode. When changing the operating mode or analog input channel, to prevent incorrect operation, first clear the ADST bit in ADCSR to 0 to halt A/D conversion. The ADST bit can be set to 1 at the same time as the operating mode or analog input channel is changed.

15.4.1 Single Mode

In single mode, A/D conversion is to be performed only once on the analog input of the specified single channel.

1. A/D conversion for the selected channel is started when the ADST bit in ADCSR is set to 1 by software or an external trigger input.
2. When A/D conversion is completed, the A/D conversion result is transferred to the corresponding A/D data register of the channel.
3. When A/D conversion is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when A/D conversion ends. The A/D converter enters wait state. If the ADST bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters wait state.

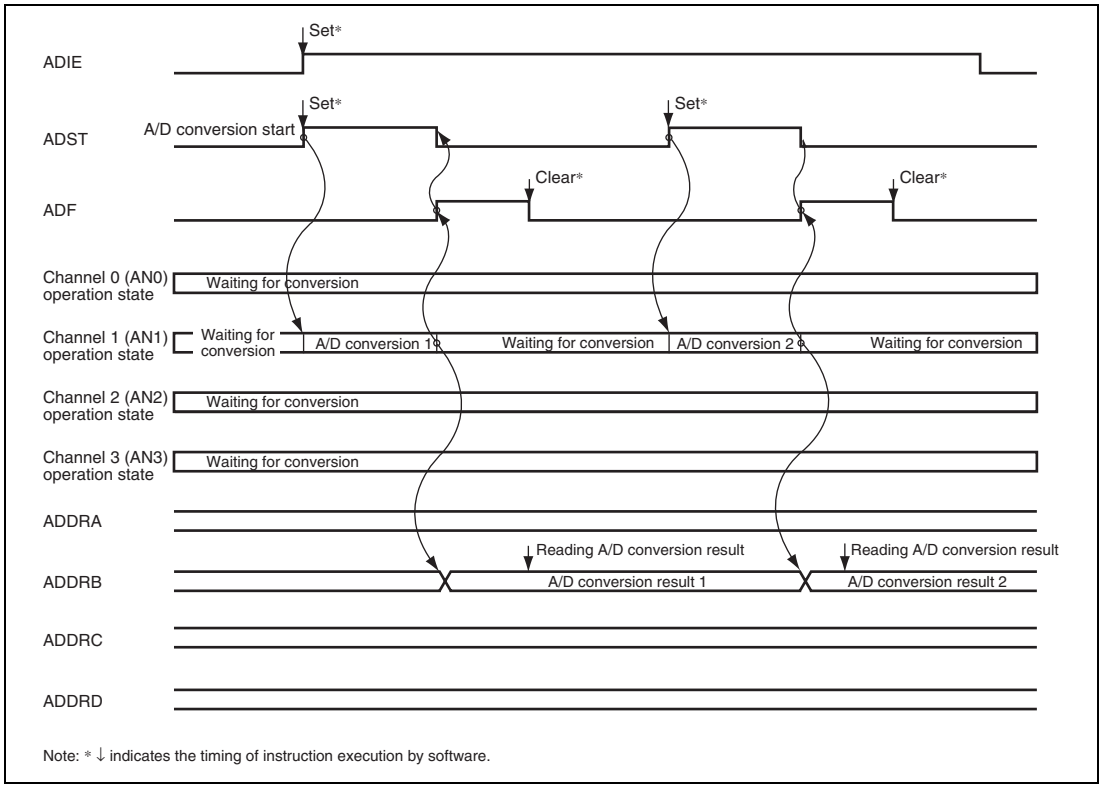


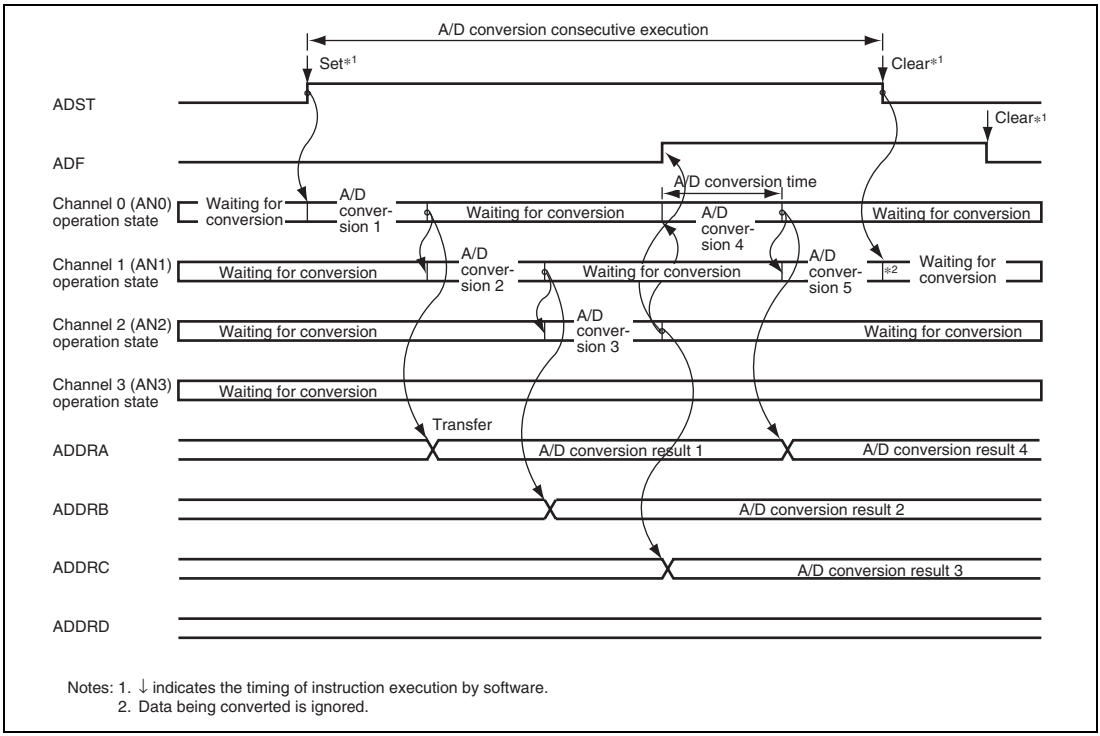
Figure 15.2 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)

15.4.2 Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the analog inputs of the specified channels up to four or eight channels.

1. When the ADST bit in ADCSR is set to 1 by software, TPU, TMR, or an external trigger input, A/D conversion starts on the first channel in the group. Consecutive A/D conversion on a maximum of four channels (SCANE and SCANS = B'10) or on a maximum of eight channels (SCANE and SCANS = B'11) can be selected. When consecutive A/D conversion is performed on four channels, A/D conversion starts on AN4 when CH3 and CH2 = B'01. When consecutive A/D conversion is performed on eight channels, A/D conversion starts on AN0 when CH3 = B'0.
2. When A/D conversion for each channel is completed, the A/D conversion result is sequentially transferred to the corresponding ADDR of each channel.

3. When A/D conversion of all selected channels is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. A/D conversion of the first channel in the group starts again.
4. The ADST bit is not cleared automatically, and steps [2] to [3] are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters wait state. If the ADST bit is later set to 1, A/D conversion starts again from the first channel in the group.



Notes: 1. ↓ indicates the timing of instruction execution by software.
2. Data being converted is ignored.

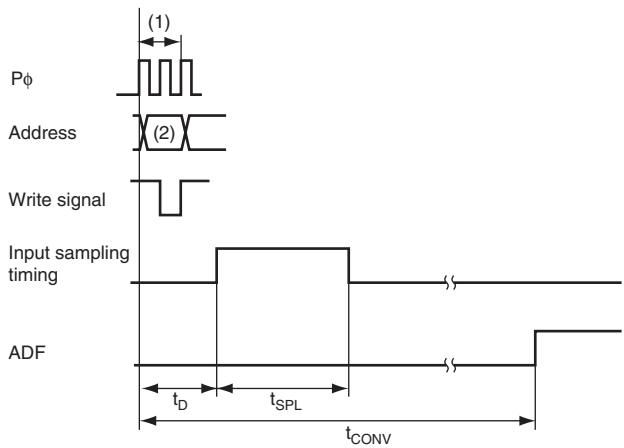
Figure 15.3 Example of A/D Conversion (Scan Mode, Three Channels (AN0 to AN2) Selected)

15.4.3 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 when the A/D conversion start delay time (t_D) passes after the ADST bit in ADCSR is set to 1, then starts A/D conversion. Figure 15.4 shows the A/D conversion timing. Table 15.3 indicates the A/D conversion time.

As indicated in figure 15.4, the A/D conversion time (t_{CONV}) includes t_D and the input sampling time (t_{SPL}). The length of t_D varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 15.3.

In scan mode, the values given in table 15.3 apply to the first conversion time. The values given in table 15.4 apply to the second and subsequent conversions. In either case, bits CKS1 and CKS0 in ADCR should be set so that the conversion time is within the ranges indicated by the A/D conversion characteristics.



[Legend]
(1): ADCSR write cycle
(2): ADCSR address
t_D: A/D conversion start delay time
t_{SPL}: Input sampling time
t_{CONV}: A/D conversion time

Figure 15.4 A/D Conversion Timing

Table 15.3 A/D Conversion Characteristics (Single Mode)

		CKS1 = 0						CKS1 = 1					
		CKS0 = 0			CKS0 = 1			CKS0 = 0			CKS0 = 1		
Item	**Symbol**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**
A/D conversion start delay time	t_D	18	—	33	10	—	17	6	—	9	4	—	5
Input sampling time	t_{SPL}	—	127	—	—	63	—	—	31	—	—	15	—
A/D conversion time	t_{CONV}	515	—	530	259	—	266	131	—	134	67	—	68

Note: Values in the table are the number of states.

Table 15.4 A/D Conversion Characteristics (Scan Mode)

CKS1	CKS0	Conversion Time (Number of States)
0	0	512 (Fixed)
	1	256 (Fixed)
1	0	128 (Fixed)
	1	64 (Fixed)

15.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to B'11 in ADCR, an external trigger is input from the $\overline{\text{ADTRG0}}$ pin. A/D conversion starts when the ADST bit in ADCSR is set to 1 on the falling edge of the $\overline{\text{ADTRG0}}$ pin. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 15.5 shows the timing.

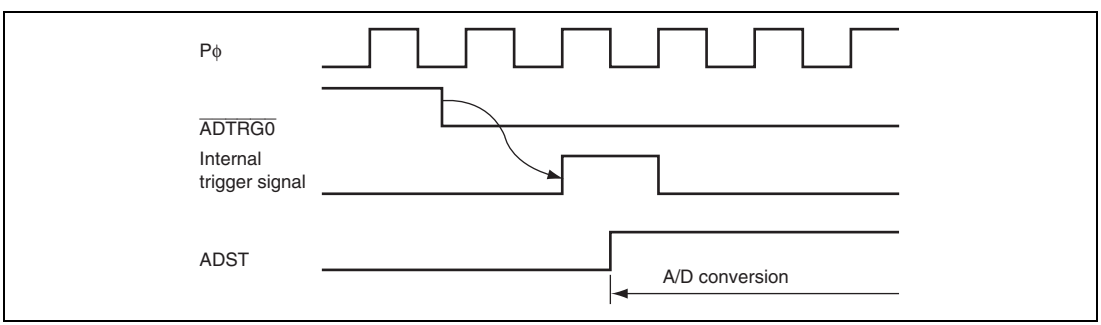


Figure 15.5 External Trigger Input Timing

15.5 Interrupt Source

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 when the ADF bit in ADCSR is set to 1 after A/D conversion is completed enables ADI interrupt requests. The data transfer controller (DTC) can be activated by an ADI interrupt. Having the converted data read by the DTC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

Table 15.5 A/D Converter Interrupt Source

Name	Interrupt Source	Interrupt Flag	DTC Activation
ADI0	A/D conversion end	ADF	Possible

15.6 A/D Conversion Accuracy Definitions

This LSI's A/D conversion accuracy definitions are given below.

- Resolution

  The number of A/D converter digital output codes.
- Quantization error

  The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 15.6).
- Offset error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'0000000000 (H'000) to B'0000000001 (H'001) (see figure 15.7).
- Full-scale error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 (H'3FE) to B'1111111111 (H'3FF) (see figure 15.7).
- Nonlinearity error

  The error with respect to the ideal A/D conversion characteristic between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error (see figure 15.7).
- Absolute accuracy

  The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.

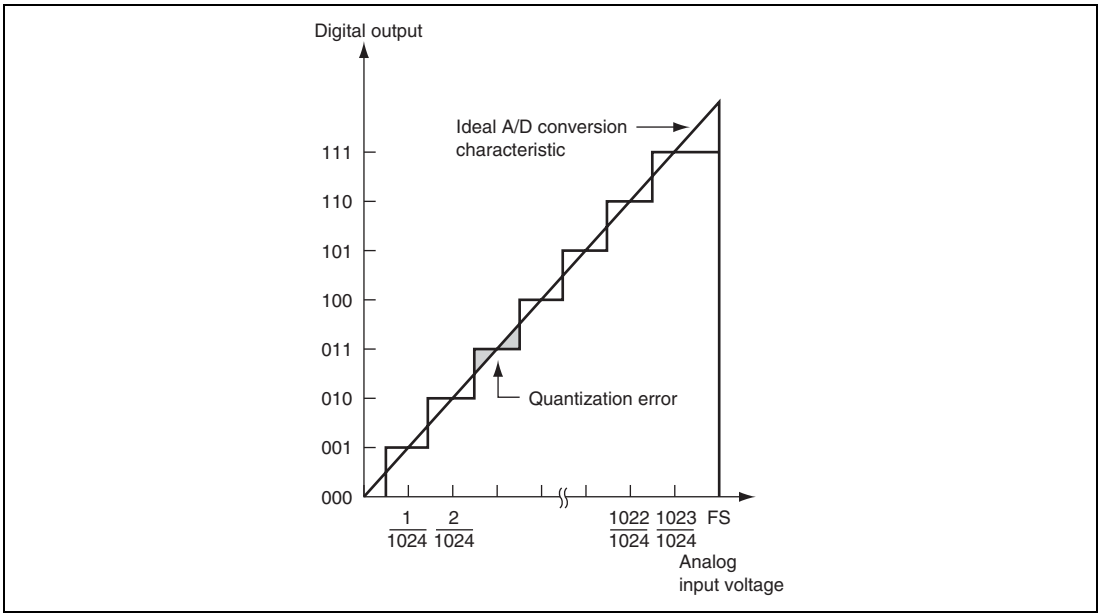


Figure 15.6 A/D Conversion Accuracy Definitions

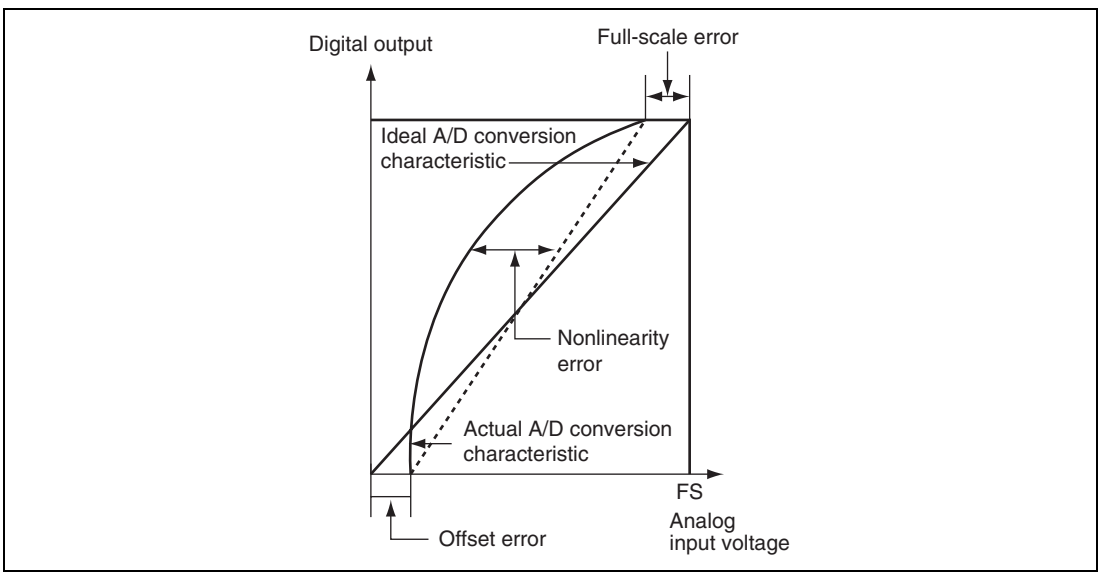


Figure 15.7 A/D Conversion Accuracy Definitions

15.7 Usage Notes

15.7.1 Module Stop State Setting

Operation of the A/D converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the A/D converter to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 20, Power-Down Modes.

15.7.2 Permissible Signal Source Impedance

This LSI's analog input is designed so that the conversion accuracy is guaranteed for an input signal for which the signal source impedance is 10 kΩ or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 10 kΩ, charging may be insufficient and it may not be possible to guarantee the A/D conversion accuracy. However, if a large capacitance is provided externally for conversion in single mode, the input load will essentially comprise only the internal input resistance of 10 kΩ, and the signal source impedance is ignored. However, since a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (e.g., 5 mV/μs or greater) (see figure 15.8). When converting a high-speed analog signal or conversion in scan mode, a low-impedance buffer should be inserted.

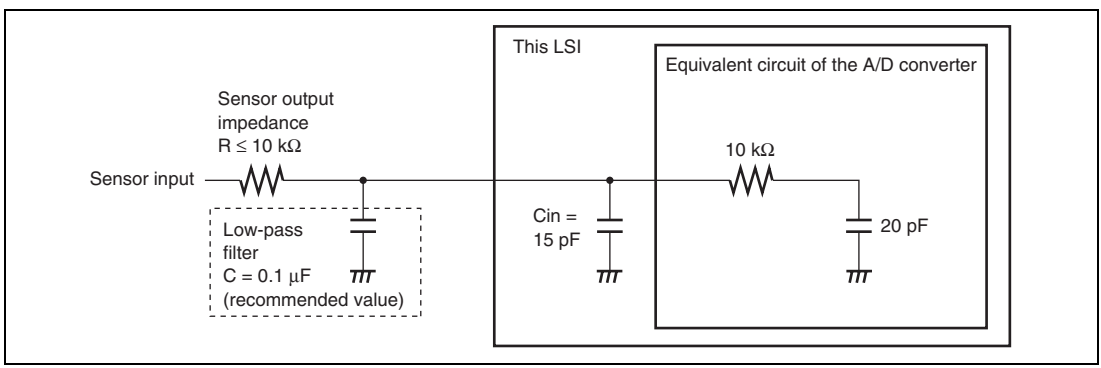


Figure 15.8 Example of Analog Input Circuit

15.7.3 Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute accuracy. Be sure to make the connection to an electrically stable GND such as AVss.

Care is also required to insure that digital signals on the board do not interfere with filter circuits and filter circuits do not act as antennas.

15.7.4 Setting Range of Analog Power Supply and Other Pins

If the conditions shown below are not met, the reliability of the LSI may be adversely affected.

- Analog input voltage range
  The voltage applied to analog input pin ANn during A/D conversion should be in the range AVss ≤ V_{AN} ≤ Vref.
- Relation between AVcc, AVss and Vcc, Vss
  As the relationship between AVcc, AVss and Vcc, Vss, set AVcc = Vcc ± 0.3 V and AVss = Vss. If the A/D converter is not used, set AVcc = Vcc and AVss = Vss.
- Vref setting range
  The reference voltage at the Vref pin should be set in the range Vref ≤ AVcc.

15.7.5 Notes on Board Design

In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values.

Digital circuitry must be isolated from the analog input pins (AN0 to AN7), analog reference power supply (Vref), and analog power supply (AVcc) by the analog ground (AVss). Also, the analog ground (AVss) should be connected at one point to a stable ground (Vss) on the board.

15.7.6 Notes on Noise Countermeasures

A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN7) should be connected between AVcc and AVss as shown in figure 15.9. Also, the bypass capacitors connected to AVcc and the filter capacitor connected to the AN0 to AN7 pins must be connected to AVss.

If a filter capacitor is connected, the input currents at the AN0 to AN7 pins are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance (R_{in}), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.

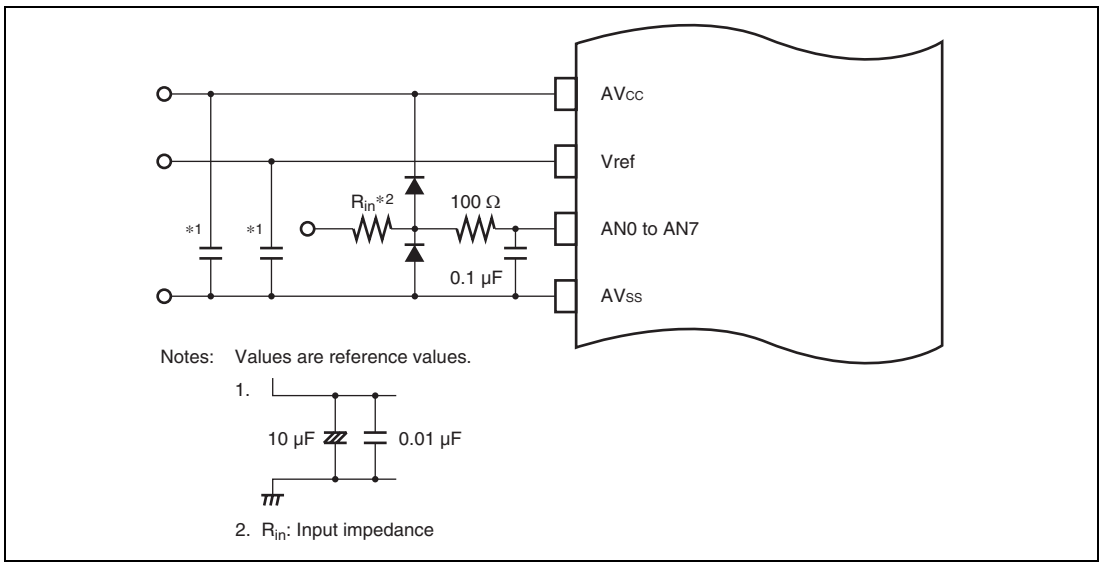
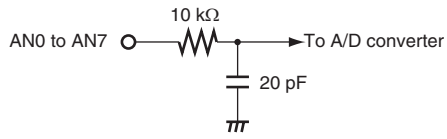


Figure 15.9 Example of Analog Input Protection Circuit

Table 15.6 Analog Pin Specifications

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Permissible signal source impedance	—	10	kΩ



Note: Values are reference values.

Figure 15.10 Analog Input Pin Equivalent Circuit

15.7.7 A/D Input Hold Function in Software Standby Mode

When this LSI enters software standby mode with A/D conversion enabled, the analog inputs are retained, and the analog power supply current is equal to as during A/D conversion. If the analog power supply current needs to be reduced in software standby mode, clear the ADST, TRGS1, and TRGS0 bits all to 0 to disable A/D conversion.

Section 16 D/A Converter

16.1 Features

- 8-bit resolution
- Two output channels
- Maximum conversion time of 10 µs (with 20 pF load)
- Output voltage of 0 V to V_{ref}
- D/A output hold function in software standby mode
- Module stop state specifiable

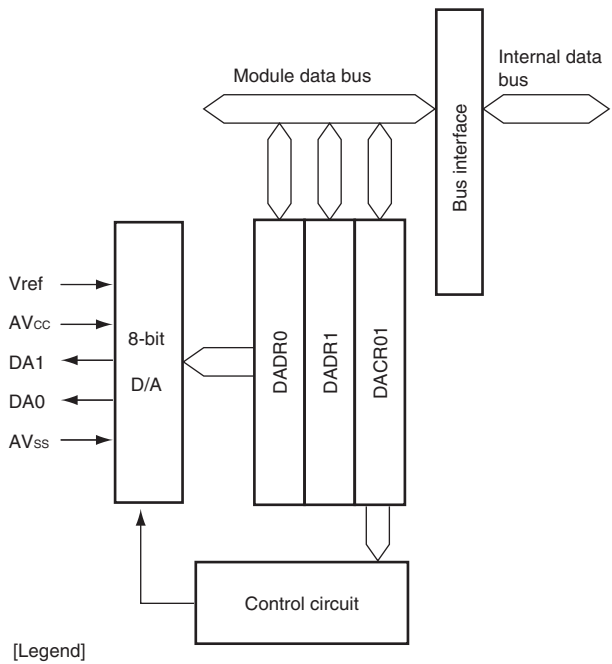


Figure 16.1 Block Diagram of D/A Converter

16.2 Input/Output Pins

Table 16.1 shows the pin configuration of the D/A converter.

Table 16.1 Pin Configuration

Pin Name	Symbol	I/O	Function
Analog power supply pin	AV_{CC}	Input	Analog block power supply
Analog ground pin	AV_{SS}	Input	Analog block ground
Reference voltage pin	Vref	Input	D/A conversion reference voltage
Analog output pin 0	DA0	Output	Channel 0 analog output
Analog output pin 1	DA1	Output	Channel 1 analog output

16.3 Register Descriptions

The D/A converter has the following registers.

- D/A data register 0 (DADR0)
- D/A data register 1 (DADR1)
- D/A control register 01 (DACR01)

16.3.1 D/A Data Registers 0 and 1 (DADR0 and DADR1)

DADR0 and DADR1 are 8-bit readable/writable registers that store data to which D/A conversion is to be performed. Whenever an analog output is enabled, the values in DADR are converted and output to the analog output pins.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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

16.3.2 D/A Control Register 01 (DACR01)

DACR01 controls the operation of the D/A converter.

Bit	7	6	5	4	3	2	1	0
Bit Name	DAOE1	DAOE0	DAE	—	—	—	—	—
Initial Value	0	0	0	1	1	1	1	1
R/W	R/W	R/W	R/W	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	DAOE1	0	R/W	D/A Output Enable 1 Controls D/A conversion and analog output. 0: Analog output of channel 1 (DA1) is disabled 1: D/A conversion of channel 1 is enabled. Analog output of channel 1 (DA1) is enabled.
6	DAOE0	0	R/W	D/A Output Enable 0 Controls D/A conversion and analog output. 0: Analog output of channel 0 (DA0) is disabled 1: D/A conversion of channel 0 is enabled. Analog output of channel 0 (DA0) is enabled.
5	DAE	0	R/W	D/A Enable Used together with the DAOE0 and DAOE1 bits to control D/A conversion. When this bit is cleared to 0, D/A conversion is controlled independently for channels 0 and 1. When this bit is set to 1, D/A conversion for channels 0 and 1 is controlled together. Output of conversion results is always controlled by the DAOE0 and DAOE1 bits. For details, see table 16.2, Control of D/A Conversion.
4 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

Table 16.2 Control of D/A Conversion

Bit 5 DAE	Bit 7 DAOE1	Bit 6 DAOE0	Description
0	0	0	D/A conversion is disabled.
		1	D/A conversion of channel 0 is enabled and D/A conversion of channel 1 is disabled. Analog output of channel 0 (DA0) is enabled and analog output of channel 1 (DA1) is disabled.
	1	0	D/A conversion of channel 0 is disabled and D/A conversion of channel 1 is enabled. Analog output of channel 0 (DA0) is disabled and analog output of channel 1 (DA1) is enabled.
		1	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is enabled.
1	0	0	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is disabled.
		1	D/A conversion of channels 0 and 1 is enabled. Analog output of channel 0 (DA0) is enabled and analog output of channel 1 (DA1) is disabled.
	1	0	D/A conversion of channels 0 and 1 is enabled. Analog output of channel 0 (DA0) is disabled and analog output of channel 1 (DA1) is enabled.
		1	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is enabled.

16.4 Operation

The D/A converter includes D/A conversion circuits for two channels, each of which can operate independently. When the DAOE bit in DACR01 is set to 1, D/A conversion is enabled and the conversion result is output.

An operation example of D/A conversion on channel 0 is shown below. Figure 16.2 shows the timing of this operation.

1. Write the conversion data to DADR0.
2. Set the DAOE0 bit in DACR01 to 1 to start D/A conversion. The conversion result is output from the analog output pin DA0 after the conversion time t_{DCONV} has elapsed. The conversion result continues to be output until DADR0 is written to again or the DAOE0 bit is cleared to 0. The output value is expressed by the following formula:

   Contents of DADR/256 × V_{ref}

3. If DADR0 is written to again, the conversion is immediately started. The conversion result is output after the conversion time t_{DCONV} has elapsed.
4. If the DAOE0 bit is cleared to 0, analog output is disabled.

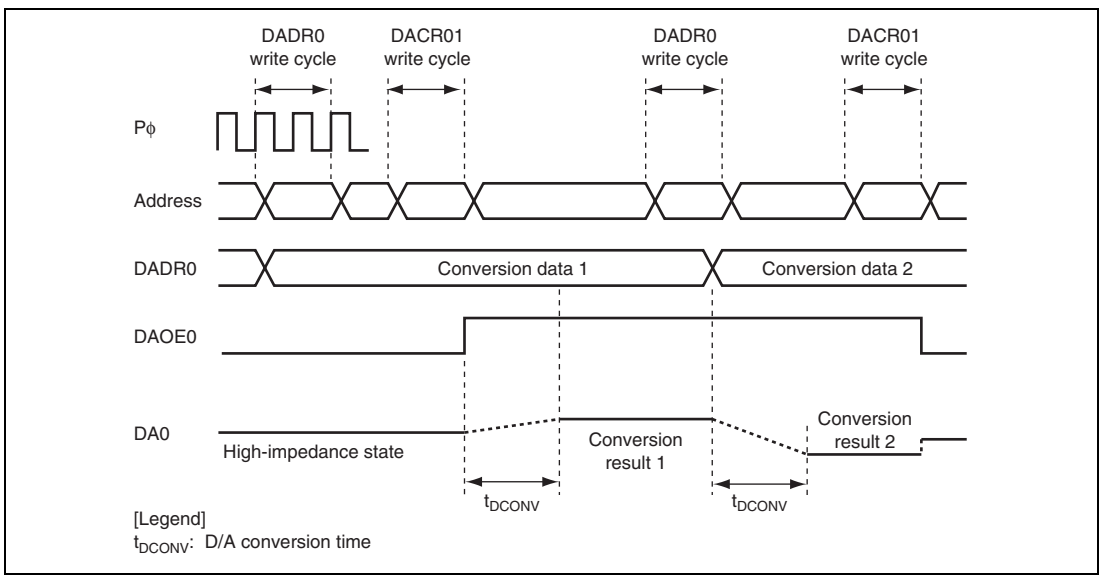


Figure 16.2 Example of D/A Converter Operation

16.5 Usage Notes

16.5.1 Module Stop State Setting

Operation of the D/A converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the D/A converter to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 20, Power-Down Modes.

16.5.2 D/A Output Hold Function in Software Standby Mode

When this LSI enters software standby mode with D/A conversion enabled, the D/A outputs are retained, and the analog power supply current is equal to as during D/A conversion. If the analog power supply current needs to be reduced in software standby mode, clear the ADST, TRGS1, and TRGS0 bits all to 0 to disable D/A conversion.

Section 17 RAM

This LSI has a 24-Kbyte on-chip high-speed static RAM. The RAM is connected to the CPU by a 32-bit data bus, enabling one-state access by the CPU to all byte data, word data, and longword data.

The on-chip RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on SYSCR, refer to section 3.2.2, System Control Register (SYSCR).

Product Classification		RAM Size	RAM Addresses
Flash memory version	H8SX/1657C	24 Kbytes	H'FF6000 to H'FFBFFF
	H8SX/1656C		

Section 18 Flash Memory (0.18-μm F-ZTAT Version)

The flash memory has the following features. Figure 18.1 is a block diagram of the flash memory.

18.1 Features

- Size

Product Classification		ROM Size	ROM Address
H8SX/1657C	R5F61657C	768 Kbytes	H'000000 to H'0BFFFF (modes 1, 2, 6, and 7)
H8SX/1656C	R5F61656C	512 Kbytes	H'000000 to H'07FFFF (modes 1, 2, 6, and 7)

- Two memory MATs

  The start addresses of two memory spaces (memory MATs) are allocated to the same address. The mode setting in the initiation determines which memory MAT is initiated first. The memory MATs can be switched by using the bank-switching method after initiation.

  — User MAT initiated at a power-on reset in user mode: 768 Kbytes/512 Kbytes

  — User boot MAT is initiated at a power-on reset in user boot mode: 8 Kbytes
- Programming/erasing interface by the download of on-chip program

  This LSI has a programming/erasing program. After downloading this program to the on-chip RAM, programming/erasing can be performed by setting the parameters.
- Programming/erasing time

  Programming time: 1 ms (typ) for 128-byte simultaneous programming, 8 μs per byte

  Erasing time: 750 ms (typ) per 1 block (64 Kbytes)
- Number of programming

  The number of programming can be up to 100 times at the minimum. (1 to 100 times are guaranteed.)
- Three on-board programming modes

  Boot mode: Using the on-chip SCI_4, the user MAT and user boot MAT can be programmed/erased. In boot mode, the bit rate between the host and this LSI can be adjusted automatically.

  User program mode: Using a desired interface, the user MAT can be programmed/erased.

  User boot mode: Using a desired interface, the user boot program can be made and the user MAT can be programmed/erased.
- Off-board programming mode

  Programmer mode: Using a PROM programmer, the user MAT and user boot MAT can be programmed/erased.

- Programming/erasing protection

  Protection against programming/erasing of the flash memory can be set by hardware protection, software protection, or error protection.
- Flash memory emulation function using the on-chip RAM

  Realtime emulation of the flash memory programming can be performed by overlaying parts of the flash memory (user MAT) area and the on-chip RAM.

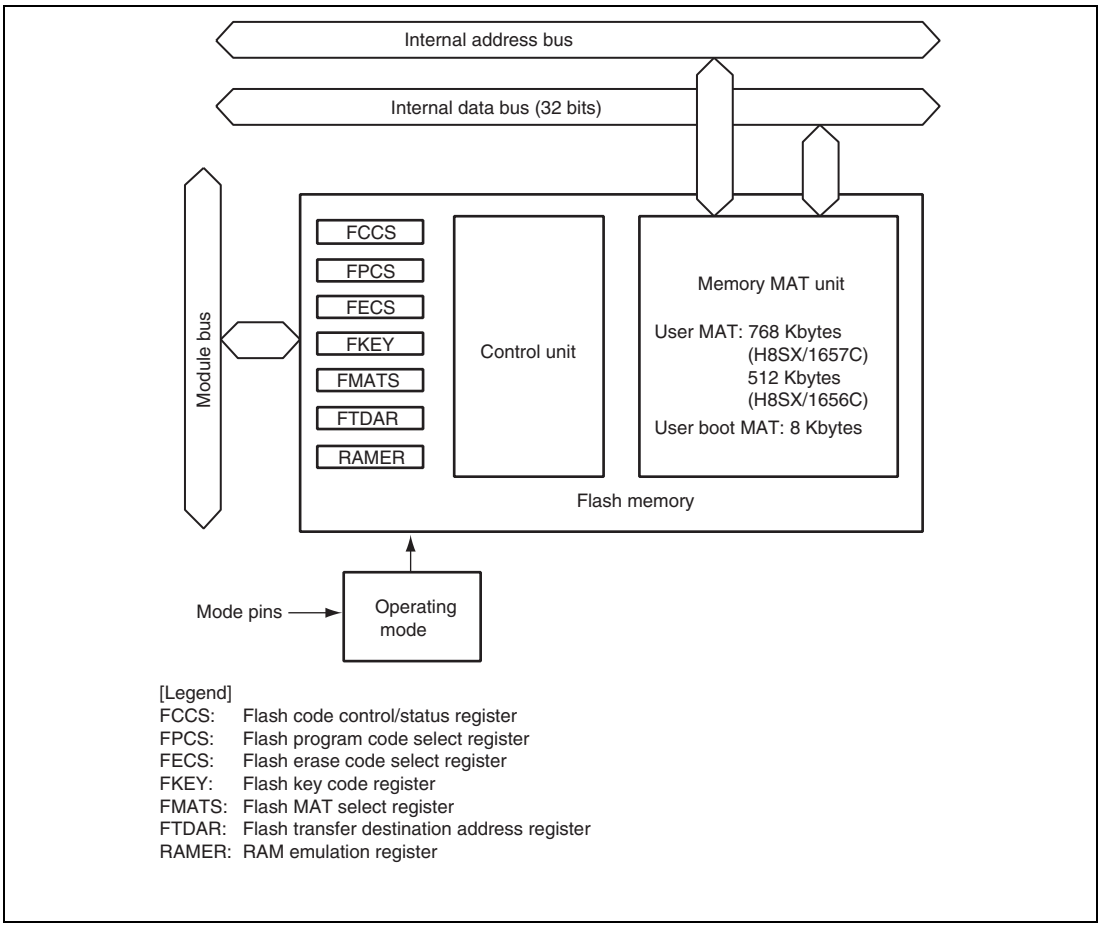
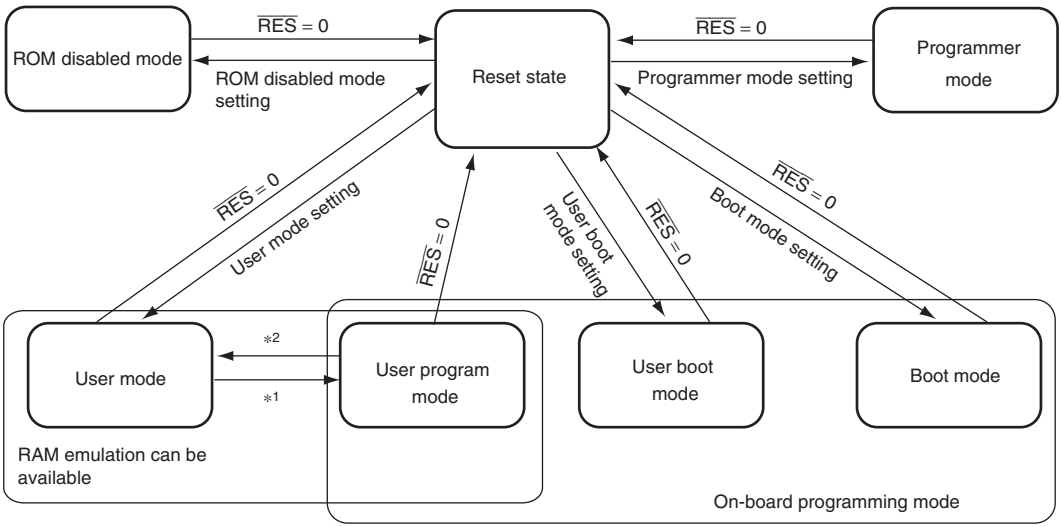


Figure 18.1 Block Diagram of Flash Memory

18.2 Mode Transition Diagram

When the mode pins are set in the reset state and reset start is performed, this LSI enters each operating mode as shown in figure 18.2. Although the flash memory can be read in user mode, it cannot be programmed or erased. The flash memory can be programmed or erased in boot mode, user program mode, user boot mode, and programmer mode. The differences between boot mode, user program mode, user boot mode, and programmer mode are shown in table 18.1.



Notes: * In this LSI, the user program mode is defined as the period from the timing when a program concerning programming and erasure is started in user mode to the timing when the program is completed.
1. Programming and erasure is started.
2. Programing and erasure is completed.

Figure 18.2 Mode Transition of Flash Memory

Table 18.1 Differences between Boot Mode, User Program Mode, User Boot Mode, and Programmer Mode

Item	Boot Mode	User Program Mode	User Boot Mode	Programmer Mode
Programming/ erasing environment	On-board programming	On-board programming	On-board programming	Off-board programming
Programming/ erasing enable MAT	• User MAT • User boot MAT	• User MAT	• User MAT	• User MAT • User boot MAT
Programming/ erasing control	Command	Programming/ erasing interface	Programming/ erasing interface	Command
All erasure	O (Automatic)	O	O	O (Automatic)
Block division erasure	O*[1]	O	O	×
Program data transfer	From host via SCI	From desired device via RAM	From desired device via RAM	Via programmer
RAM emulation	×	O	O	×
Reset initiation MAT	Embedded program storage area	User MAT	User boot MAT*[2]	—
Transition to user mode	Changing mode and reset	Completing Programming/ erasure*[3]	Changing mode and reset	—

Notes: 1. All-erasure is performed. After that, the specified block can be erased.

2. First, the reset vector is fetched from the embedded program storage area. After the flash memory related registers are checked, the reset vector is fetched from the user boot MAT.

3. In this LSI, the user programming mode is defined as the period from the timing when a program concerning programming and erasure is started to the timing when the program is completed. For details on a program concerning programming and erasure, see section 18.8.2, User Program Mode.

18.3 Memory MAT Configuration

The memory MATs of flash memory in this LSI consists of the 768-kbyte user MAT and 8-kbyte user boot MAT. The start addresses of the user MAT and user boot MAT are allocated to the same address. Therefore, when the program execution or data access is performed between the two memory MATs, the memory MATs must be switched by the flash MAT select register (FMATS).

The user MAT or user boot MAT can be read in all modes. However, the user boot MAT can be programmed or erased only in boot mode and programmer mode.

The size of the user MAT is different from that of the user boot MAT. Addresses which exceed the size of the 8-kbyte user boot MAT should not be accessed. If an attempt is made, data is read as an undefined value.

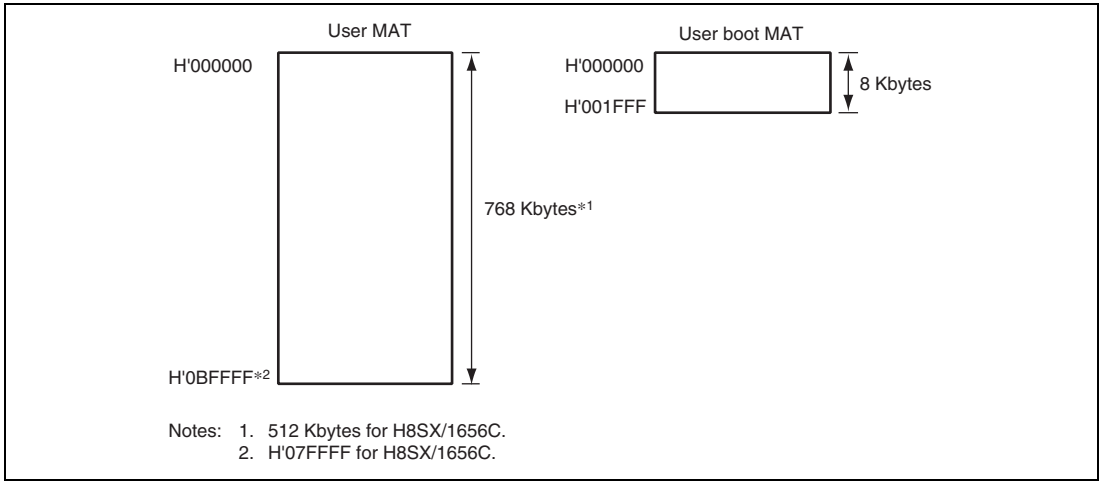


Figure 18.3 Memory MAT Configuration (H8SX/1657C)

18.4 Block Structure

18.4.1 Block Diagram of H8SX/1657C

Figure 18.4 (1) shows the block structure of the 768-Kbyte user MAT. The heavy-line frames indicate the erase blocks. The thin-line frames indicate the programming units and the values inside the frames stand for the addresses. The user MAT is divided into eleven 64-Kbyte blocks, one 32-Kbyte block, and eight 4-Kbyte blocks. The user MAT can be erased in these divided block units. Programming is done in 128-byte units starting from where the lower address is H'00 or H'80. RAM emulation can be performed in the eight 4-Kbyte blocks.

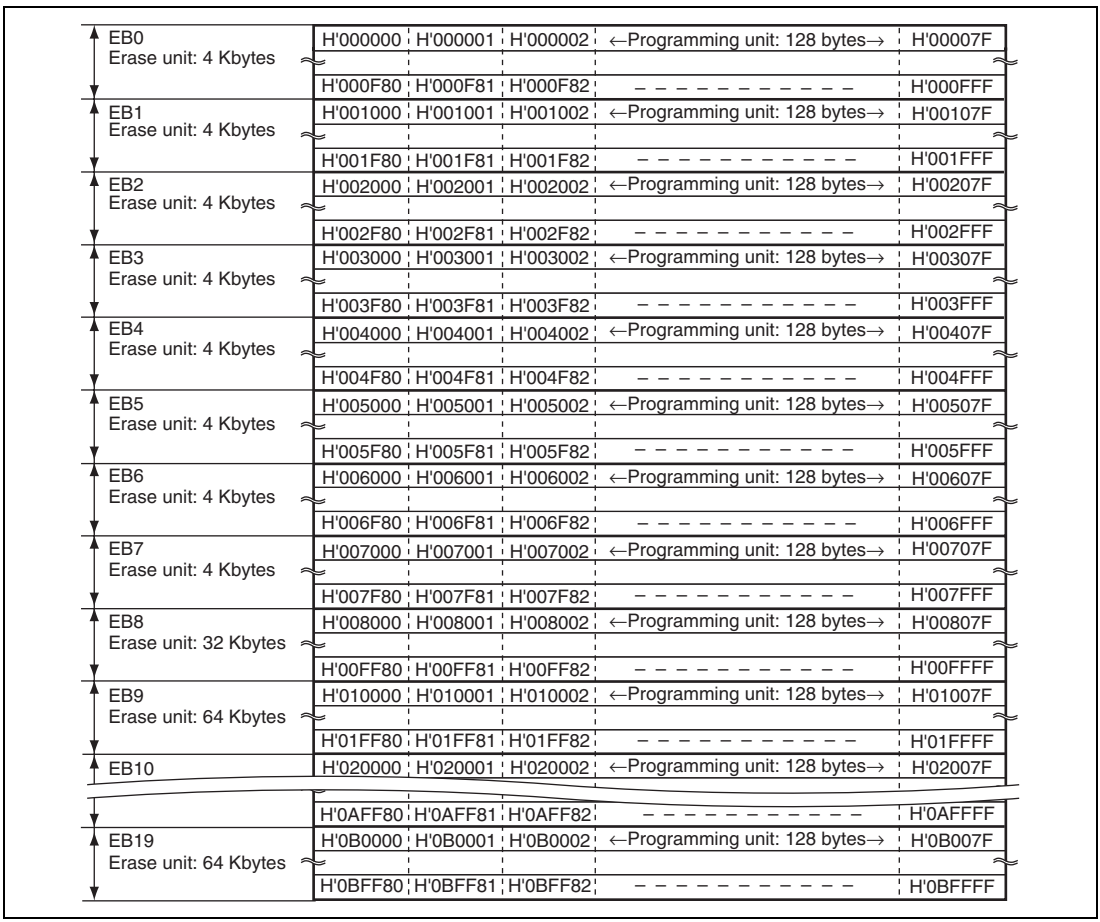


Figure 18.4 User MAT Block Structure of H8SX/1657C (1)

18.4.2 Block Diagram of H8SX/1656C

Figure 18.4 (2) shows the block structure of the 512-Kbyte user MAT. The heavy-line frames indicate the erase blocks. The thin-line frames indicate the programming units and the values inside the frames stand for the addresses. The user MAT is divided into seven 64-Kbyte blocks, one 32-Kbyte block, and eight 4-Kbyte blocks. The user MAT can be erased in these divided block units. Programming is done in 128-byte units starting from where the lower address is H'00 or H'80. RAM emulation can be performed in the eight 4-Kbyte blocks.

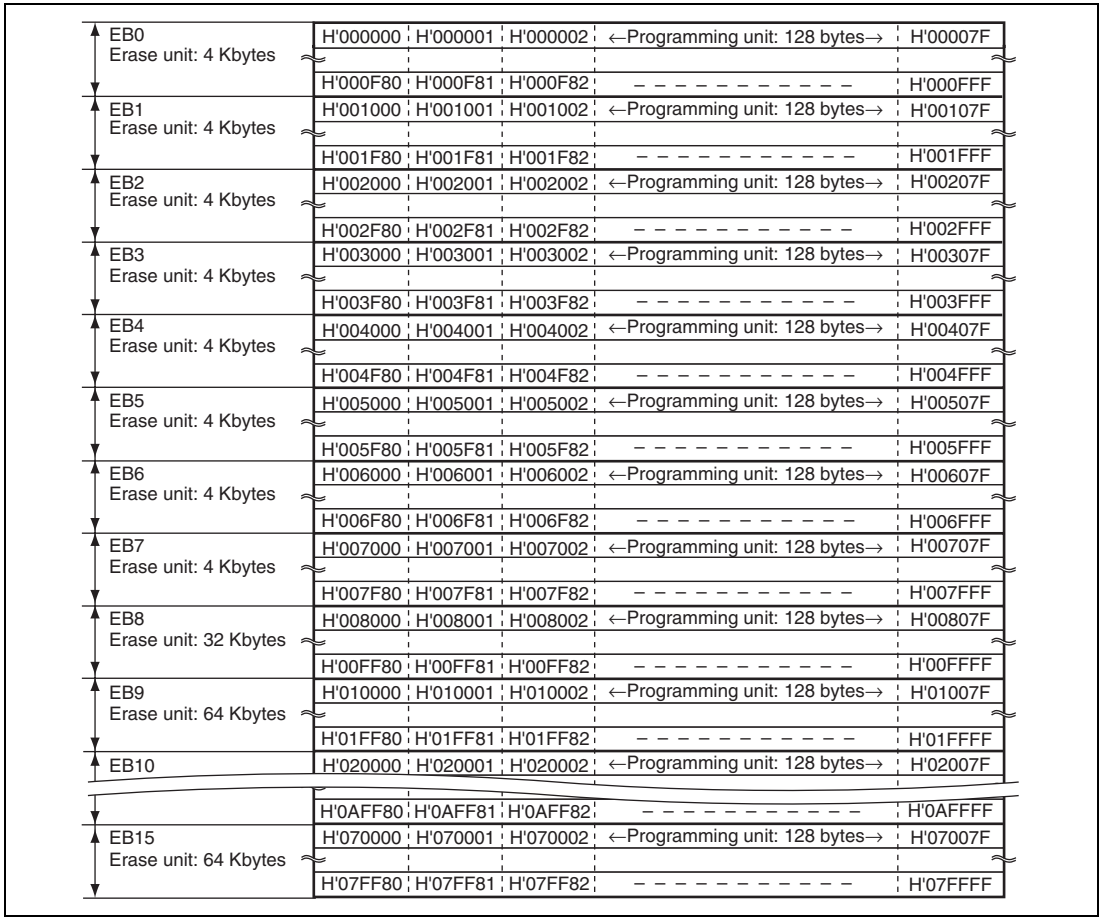


Figure 18.4 User MAT Block Structure of H8SX/1656C (2)

18.5 Programming/Erasing Interface

Programming/erasing of the flash memory is done by downloading an on-chip programming/erasing program to the on-chip RAM and specifying the start address of the programming destination, the program data, and the erase block number using the programming/erasing interface registers and programming/erasing interface parameters.

The procedure program for user program mode and user boot mode is made by the user. Figure 18.5 shows the procedure for creating the procedure program. For details, see section 18.8.2, User Program Mode.

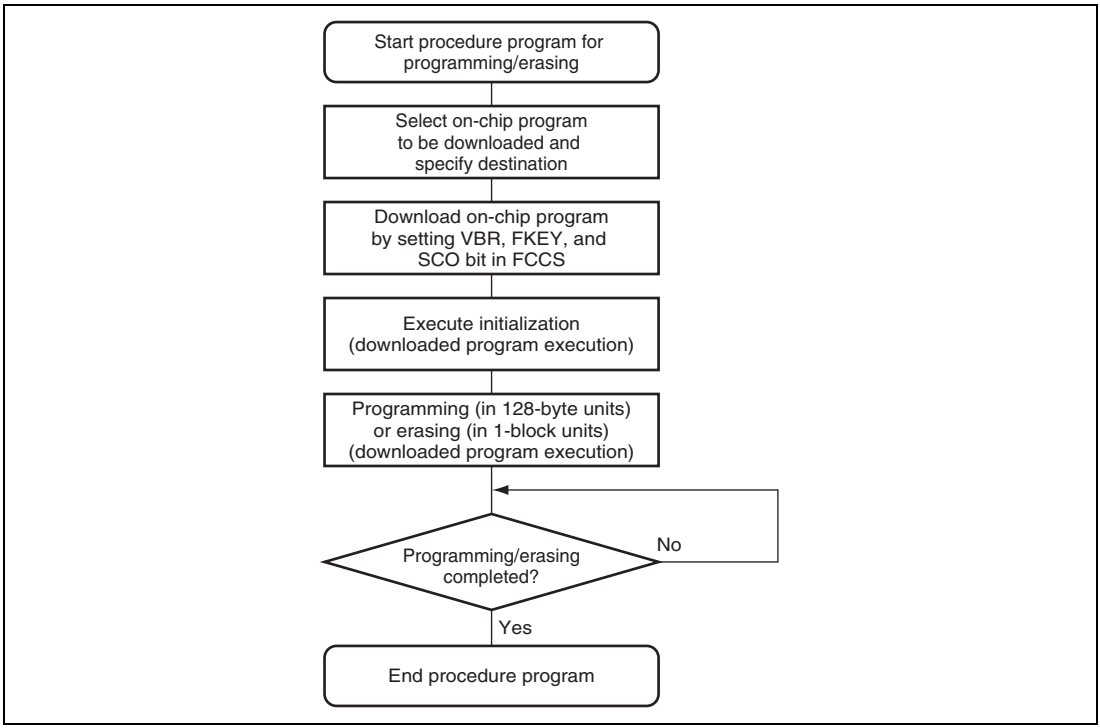


Figure 18.5 Procedure for Creating Procedure Program

(1) Selection of On-Chip Program to be Downloaded

This LSI has programming/erasing programs which can be downloaded to the on-chip RAM. The on-chip program to be downloaded is selected by the programming/erasing interface registers. The start address of the on-chip RAM where an on-chip program is downloaded is specified by the flash transfer destination address register (FTDAR).

(2) Download of On-Chip Program

The on-chip program is automatically downloaded by setting the flash key code register (FKEY) and the SCO bit in the flash code control/status register (FCCS) after initializing the vector base register (VBR). The memory MAT is replaced with the embedded program storage area during download. Since the memory MAT cannot be read during programming/erasing, the procedure program must be executed in a space other than the flash memory (for example, on-chip RAM). Since the download result is returned to the programming/erasing interface parameter, whether download is normally executed or not can be confirmed. The VBR contents can be changed after completion of download.

(3) Initialization of Programming/Erasing

A pulse with the specified period must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. Accordingly, the operating frequency of the CPU needs to be set before programming/erasing. The operating frequency of the CPU is set by the programming/erasing interface parameter.

(4) Execution of Programming/Erasing

The start address of the programming destination and the program data are specified in 128-byte units when programming. The block to be erased is specified with the erase block number in erase-block units when erasing. Specifications of the start address of the programming destination, program data, and erase block number are performed by the programming/erasing interface parameters, and the on-chip program is initiated. The on-chip program is executed by using the JSR or BSR instruction and executing the subroutine call of the specified address in the on-chip RAM. The execution result is returned to the programming/erasing interface parameter.

The area to be programmed must be erased in advance when programming flash memory. All interrupts are disabled during programming/erasing.

(5) When Programming/Erasing is Executed Consecutively

When processing does not end by 128-byte programming or 1-block erasure, consecutive programming/erasing can be realized by updating the start address of the programming destination and program data, or the erase block number. Since the downloaded on-chip program is left in the on-chip RAM even after programming/erasing completes, download and initialization are not required when the same processing is executed consecutively.

18.6 Input/Output Pins

The flash memory is controlled through the input/output pins shown in table 18.2.

Table 18.2 Pin Configuration

Abbreviation	I/O	Function
$\overline{\text{RES}}$	Input	Reset
MD2 to MD0	Input	Set operating mode of this LSI
TxD4	Output	Serial transmit data output (used in boot mode)
RxD4	Input	Serial receive data input (used in boot mode)

18.7 Register Descriptions

The flash memory has the following registers.

Programming/Erasing Interface Registers:

- Flash code control/status register (FCCS)
- Flash program code select register (FPCS)
- Flash erase code select register (FECS)
- Flash key code register (FKEY)
- Flash MAT select register (FMATS)
- Flash transfer destination address register (FTDAR)

Programming/Erasing Interface Parameters:

- Download pass and fail result parameter (DPFR)
- Flash pass and fail result parameter (FPFR)
- Flash program/erase frequency parameter (FPEFEQ)
- Flash multipurpose address area parameter (FMPAR)
- Flash multipurpose data destination area parameter (FMPDR)
- Flash erase block select parameter (FEBS)

- RAM emulation register (RAMER)

There are several operating modes for accessing the flash memory. Respective operating modes, registers, and parameters are assigned to the user MAT and user boot MAT. The correspondence between operating modes and registers/parameters for use is shown in table 18.3.

Table 18.3 Registers/Parameters and Target Modes

Register/Parameter		Down-load	Initiali-zation	Program-ming	Erasure	Read	RAM Emulation
Programming/ erasing interface registers	FCCS	O	—	—	—	—	—
	FPCS	O	—	—	—	—	—
	FECS	O	—	—	—	—	—
	FKEY	O	—	O	O	—	—
	FMATS	—	—	O*[1]	O*[1]	O*[2]	—
	FTDAR	O	—	—	—	—	—
Programming/ erasing interface parameters	DPFR	O	—	—	—	—	—
	FPFR	—	O	O	O	—	—
	FPEFEQ	—	O	—	—	—	—
	FMPAR	—	—	O	—	—	—
	FMPDR	—	—	O	—	—	—
	FEBS	—	—	—	O	—	—
RAM emulation	RAMER	—	—	—	—	—	O

Notes: 1. The setting is required when programming or erasing the user MAT in user boot mode.
2. The setting may be required according to the combination of initiation mode and read target memory MAT.

18.7.1 Programming/Erasing Interface Registers

The programming/erasing interface registers are 8-bit registers that can be accessed only in bytes. These registers are initialized by a power-on reset.

(1) Flash Code Control/Status Register (FCCS)

FCCS monitors errors during programming/erasing the flash memory and requests the on-chip program to be downloaded to the on-chip RAM.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	FLER	—	—	—	SCO
Initial Value	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	(R)/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R	Reserved
6	—	0	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	FLER	0	R	Flash Memory Error Indicates that an error has occurred during programming or erasing the flash memory. When this bit is set to 1, the flash memory enters the error protection state. When this bit is set to 1, high voltage is applied to the internal flash memory. To reduce the damage to the flash memory, the reset must be released after the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs. 0: Flash memory operates normally (Error protection is invalid) [Clearing condition] • At a power-on reset 1: An error occurs during programming/erasing flash memory (Error protection is valid) [Setting conditions] • When an interrupt, such as NMI, occurs during programming/erasing. • When the flash memory is read during programming/erasing (including a vector read and an instruction fetch). • When the SLEEP instruction is executed during programming/erasing (including software standby mode). • When a bus master other than the CPU, such as the DMAC and DTC, obtains bus mastership during programming/erasing.
3 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
0	SCO	0	(R)/W*	Source Program Copy Operation Requests the on-chip programming/erasing program to be downloaded to the on-chip RAM. When this bit is set to 1, the on-chip program which is selected by FPCS or FECS is automatically downloaded in the on-chip RAM area specified by FTDAR. In order to set this bit to 1, the RAM emulation mode must be canceled, H'A5 must be written to FKEY, and this operation must be executed in the on-chip RAM. Dummy read of FCCS must be executed twice immediately after setting this bit to 1. All interrupts must be disabled during download. This bit is cleared to 0 when download is completed. During program download initiated with this bit, particular processing which accompanies bank-switching of the program storage area is executed. Before a download request, initialize the VBR contents to H'00000000. After download is completed, the VBR contents can be changed. 0: Download of the programming/erasing program is not requested. [Clearing condition] • When download is completed 1: Download of the programming/erasing program is requested. [Setting conditions] (When all of the following conditions are satisfied) • Not in RAM emulation mode (the RAMS bit in RAMER is cleared to 0) • H'A5 is written to FKEY • Setting of this bit is executed in the on-chip RAM

Note: * This is a write-only bit. This bit is always read as 0.

(2) Flash Program Code Select Register (FPCS)

FPCS selects the programming program to be downloaded.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	PPVS
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	PPVS	0	R/W	Program Pulse Verify Selects the programming program to be downloaded. 0: Programming program is not selected. [Clearing condition] When transfer is completed 1: Programming program is selected.

(3) Flash Erase Code Select Register (FECS)

FECS selects the erasing program to be downloaded.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	EPVB
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	EPVB	0	R/W	Erase Pulse Verify Block Selects the erasing program to be downloaded. 0: Erasing program is not selected. [Clearing condition] When transfer is completed 1: Erasing program is selected.

(4) Flash Key Code Register (FKEY)

FKEY is a register for software protection that enables to download the on-chip program and perform programming/erasing of the flash memory.

Bit	7	6	5	4	3	2	1	0
Bit Name	K7	K6	K5	K4	K3	K2	K1	K0
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	Bit Name	Initial Value	R/W	Description
7	K7	0	R/W	Key Code
6	K6	0	R/W	When H'A5 is written to FKEY, writing to the SCO bit in FCCS is enabled. When a value other than H'A5 is written, the SCO bit cannot be set to 1. Therefore, the on-chip program cannot be downloaded to the on-chip RAM.
5	K5	0	R/W	
4	K4	0	R/W	
3	K3	0	R/W	
2	K2	0	R/W	Only when H'5A is written can programming/erasing of the flash memory be executed. When a value other than H'5A is written, even if the programming/erasing program is executed, programming/erasing cannot be performed.
1	K1	0	R/W	
0	K0	0	R/W	
				H'A5: Writing to the SCO bit is enabled. (The SCO bit cannot be set to 1 when FKEY is a value other than H'A5.)
				H'5A: Programming/erasing of the flash memory is enabled. (When FKEY is a value other than H'A5, the software protection state is entered.)
				H'00: Initial value

(5) Flash MAT Select Register (FMATS)

FMATS selects the user MAT or user boot MAT. Writing to FMATS should be done when a program in the on-chip RAM is being executed.

Bit	7	6	5	4	3	2	1	0
Bit Name	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0
Initial Value	0/1*	0	0/1*	0	0/1*	0	0/1*	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * This bit is set to 1 in user boot mode, otherwise cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7	MS7	0/1*	R/W	MAT Select
6	MS6	0	R/W	The memory MATs can be switched by writing a value to FMATS.
5	MS5	0/1*	R/W	
4	MS4	0	R/W	When H'AA is written to FMATS, the user boot MAT is selected. When a value other than H'AA is written, the user MAT is selected. Switch the MATs following the memory MAT switching procedure in section 18.11, Switching between User MAT and User Boot MAT. The user boot MAT cannot be selected by FMATS in user programming mode. The user boot MAT can be selected in boot mode or programmer mode.
3	MS3	0/1*	R/W	
2	MS2	0	R/W	
1	MS1	0/1*	R/W	
0	MS0	0	R/W	
				H'AA: The user boot MAT is selected. (The user MAT is selected when FMATS is a value other than H'AA.) (Initial value when initiated in user boot mode.)
				H'00: The user MAT is selected. (Initial value when initiated in a mode except for user boot mode.)

Note: * This bit is set to 1 in user boot mode, otherwise cleared to 0.

(6) Flash Transfer Destination Address Register (FTDAR)

FTDAR specifies the start address of the on-chip RAM at which to download an on-chip program. FTDAR must be set before setting the SCO bit in FCCS to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0
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	Bit Name	Initial Value	R/W	Description
7	TDER	0	R/W	Transfer Destination Address Setting Error This bit is set to 1 when an error has occurred in setting the start address specified by bits TDA6 to TDA0. A start address error is determined by whether the value set in bits TDA6 to TDA0 is within the range of H'00 to H'02 when download is executed by setting the SCO bit in FCCS to 1. Make sure that this bit is cleared to 0 before setting the SCO bit to 1 and the value specified by bits TDA6 to TDA0 should be within the range of H'00 to H'02. 0: The value specified by bits TDA6 to TDA0 is within the range. 1: The value specified by bits TDA6 to TDA0 is between H'03 and H'FF and download has stopped.
6	TDA6	0	R/W	Transfer Destination Address Specifies the on-chip RAM start address of the download destination. A value between H'00 and H'02, and up to 4 Kbytes can be specified as the start address of the on-chip RAM. H'00: H'FF9000 is specified as the start address. H'01: H'FFA000 is specified as the start address. H'02: H'FFB000 is specified as the start address. H'03 to H'7F: Setting prohibited. (Specifying a value from H'03 to H'7F sets the TDER bit to 1 and stops download of the on-chip program.)
5	TDA5	0	R/W	
4	TDA4	0	R/W	
3	TDA3	0	R/W	
2	TDA2	0	R/W	
1	TDA1	0	R/W	
0	TDA0	0	R/W	

18.7.2 Programming/Erasing Interface Parameters

The programming/erasing interface parameters specify the operating frequency, storage place for program data, start address of programming destination, and erase block number, and exchanges the execution result. These parameters use the general registers of the CPU (ER0 and ER1) or the on-chip RAM area. The initial values of programming/erasing interface parameters are undefined at a power-on reset or a transition to software standby mode.

Since registers of the CPU except for ER0 and ER1 are saved in the stack area during download of an on-chip program, initialization, programming, or erasing, allocate the stack area before performing these operations (the maximum stack size is 128 bytes). The return value of the processing result is written in R0. The programming/erasing interface parameters are used in download control, initialization before programming or erasing, programming, and erasing. Table 18.4 shows the usable parameters and target modes. The meaning of the bits in the flash pass and fail result parameter (FPFR) varies in initialization, programming, and erasure.

Table 18.4 Parameters and Target Modes

Parameter	Download	Initialization	Programming	Erasure	R/W	Initial Value	Allocation
DPFR	O	—	—	—	R/W	Undefined	On-chip RAM*
FPFR	O	O	O	O	R/W	Undefined	R0L of CPU
FPEFEQ	—	O	—	—	R/W	Undefined	ER0 of CPU
FMPAR	—	—	O	—	R/W	Undefined	ER1 of CPU
FMPDR	—	—	O	—	R/W	Undefined	ER0 of CPU
FEBS	—	—	—	O	R/W	Undefined	ER0 of CPU

Note: * A single byte of the start address of the on-chip RAM specified by FTDAR

Download Control: The on-chip program is automatically downloaded by setting the SCO bit in FCCS to 1. The on-chip RAM area to download the on-chip program is the 4-kbyte area starting from the start address specified by FTDAR. Download is set by the programming/erasing interface registers, and the download pass and fail result parameter (DPFR) indicates the return value.

Initialization before Programming/Erasing: The on-chip program includes the initialization program. A pulse with the specified period must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. Accordingly, the operating frequency of the CPU must be set. The initial program is set as a parameter of the programming/erasing program which has been downloaded to perform these settings.

Programming: When the flash memory is programmed, the start address of the programming destination on the user MAT and the program data must be passed to the programming program.

The start address of the programming destination on the user MAT must be stored in general register ER1. This parameter is called the flash multipurpose address area parameter (FMPAR).

The program data is always in 128-byte units. When the program data does not satisfy 128 bytes, 128-byte program data is prepared by filling the dummy code (H'FF). The boundary of the start address of the programming destination on the user MAT is aligned at an address where the lower eight bits (A7 to A0) are H'00 or H'80.

The program data for the user MAT must be prepared in consecutive areas. The program data must be in a consecutive space which can be accessed using the MOV.B instruction of the CPU and is not in the flash memory space.

The start address of the area that stores the data to be written in the user MAT must be set in general register ER0. This parameter is called the flash multipurpose data destination area parameter (FMPDR).

For details on the programming procedure, see section 18.8.2, User Program Mode.

Erasure: When the flash memory is erased, the erase block number on the user MAT must be passed to the erasing program which is downloaded.

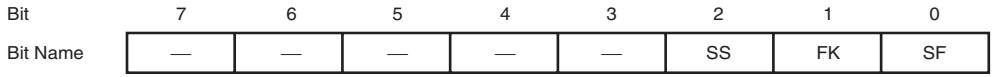
The erase block number on the user MAT must be set in general register ER0. This parameter is called the flash erase block select parameter (FEBS).

One block is selected from the block numbers of 0 to 19 as the erase block number.

For details on the erasing procedure, see section 18.8.2, User Program Mode.

(1) Download Pass and Fail Result Parameter (DPFR: Single Byte of Start Address in On-Chip RAM Specified by FTDAR)

DPFR indicates the return value of the download result. The DPFR value is used to determine the download result.



Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused These bits return 0.
2	SS	—	R/W	Source Select Error Detect Only one type can be specified for the on-chip program which can be downloaded. When the program to be downloaded is not selected, more than two types of programs are selected, or a program which is not mapped is selected, an error occurs. 0: Download program selection is normal 1: Download program selection is abnormal
1	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'A5) and returns the result. 0: FKEY setting is normal (H'A5) 1: FKEY setting is abnormal (value other than H'A5)
0	SF	—	R/W	Success/Fail Returns the download result. Reads back the program downloaded to the on-chip RAM and determines whether it has been transferred to the on-chip RAM. 0: Download of the program has ended normally (no error) 1: Download of the program has ended abnormally (error occurs)

(2) Flash Pass and Fail Parameter (FPFR: General Register R0L of CPU)

FPFR indicates the return values of the initialization, programming, and erasure results. The meaning of the bits in FPFR varies depending on the processing.

(a) Initialization before programming/erasing

FPFR indicates the return value of the initialization result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	FQ	SF

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	—	—	Unused These bits return 0.
1	FQ	—	R/W	Frequency Error Detect Compares the specified CPU operating frequency with the operating frequencies supported by this LSI, and returns the result. 0: Setting of operating frequency is normal 1: Setting of operating frequency is abnormal
0	SF	—	R/W	Success/Fail Returns the initialization result. 0: Initialization has ended normally (no error) 1: Initialization has ended abnormally (error occurs)

(b) Programming

FPFR indicates the return value of the programming result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	—	WD	WA	SF

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Returns 0.
6	MD	—	R/W	Programming Mode Related Setting Error Detect Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 18.9.3, Error Protection. 0: Normal operation (FLER = 0) 1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Programming Execution Error Detect Writes 1 to this bit when the specified data could not be written because the user MAT was not erased. If this bit is set to 1, there is a high possibility that the user MAT has been written to partially. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when programming is performed. In this case, both the user MAT and user boot MAT have not been written to. Programming the user boot MAT should be performed in boot mode or programmer mode. 0: Programming has ended normally 1: Programming has ended abnormally (programming result is not guaranteed)

Bit	Bit Name	Initial Value	R/W	Description
4	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'5A) before programming starts, and returns the result. 0: FKEY setting is normal (H'5A) 1: FKEY setting is abnormal (value other than H'5A)
3	—	—	—	Unused Returns 0.
2	WD	—	R/W	Write Data Address Detect When an address not in the flash memory area is specified as the start address of the storage destination for the program data, an error occurs. 0: Setting of the start address of the storage destination for the program data is normal 1: Setting of the start address of the storage destination for the program data is abnormal
1	WA	—	R/W	Write Address Error Detect When the following items are specified as the start address of the programming destination, an error occurs. • An area other than flash memory • The specified address is not aligned with the 128-byte boundary (lower eight bits of the address are other than H'00 and H'80) 0: Setting of the start address of the programming destination is normal 1: Setting of the start address of the programming destination is abnormal
0	SF	—	R/W	Success/Fail Returns the programming result. 0: Programming has ended normally (no error) 1: Programming has ended abnormally (error occurs)

(c) Erasure

FPFR indicates the return value of the erasure result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	EB	—	—	SF

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Returns 0.
6	MD	—	R/W	Erasure Mode Related Setting Error Detect Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 18.9.3, Error Protection. 0: Normal operation (FLER = 0) 1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Erasure Execution Error Detect Returns 1 when the user MAT could not be erased or when the flash memory related register settings are partially changed. If this bit is set to 1, there is a high possibility that the user MAT has been erased partially. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when erasure is performed. In this case, both the user MAT and user boot MAT have not been erased. Erasing of the user boot MAT should be performed in boot mode or programmer mode. 0: Erasure has ended normally 1: Erasure has ended abnormally

Bit	Bit Name	Initial Value	R/W	Description
4	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'5A) before erasure starts, and returns the result. 0: FKEY setting is normal (H'5A) 1: FKEY setting is abnormal (value other than H'5A)
3	EB	—	R/W	Erase Block Select Error Detect Checks whether the specified erase block number is in the block range of the user MAT, and returns the result. 0: Setting of erase block number is normal 1: Setting of erase block number is abnormal
2, 1	—	—	—	Unused These bits return 0.
0	SF	—	R/W	Success/Fail Indicates the erasure result. 0: Erasure has ended normally (no error) 1: Erasure has ended abnormally (error occurs)

(3) Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU)

FPEFEQ sets the operating frequency of the CPU. The operating frequency available in this LSI ranges from 8 MHz to 35 MHz.

Bit	31	30	29	28	27	26	25	24
Bit Name	—	—	—	—	—	—	—	—

Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	—	—	—	—	—	—

Bit	15	14	13	12	11	10	9	8
Bit Name	F15	F14	F13	F12	F11	F10	F9	F8

Bit	7	6	5	4	3	2	1	0
Bit Name	F7	F6	F5	F4	F3	F2	F1	F0

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	—	—	Unused These bits should be cleared to 0.
15 to 0	F15 to F0	—	R/W	Frequency Set These bits set the operating frequency of the CPU. When the PLL multiplication function is used, set the multiplied frequency. The setting value must be calculated as follows: 1. The operating frequency shown in MHz units must be rounded in a number of three decimal places and be shown in a number of two decimal places. 2. The value multiplied by 100 is converted to the binary digit and is written to FPEFEQ (general register ER0). For example, when the operating frequency of the CPU is 35.000 MHz, the value is as follows: 1. The number of three decimal places of 35.000 is rounded. 2. The formula of 35.00 × 100 = 3500 is converted to the binary digit and B'0000 1101 1010 1100 (H'0DAC) is set to ER0.

(4) Flash Multipurpose Address Area Parameter (FMPAR: General Register ER1 of CPU)

FMPAR stores the start address of the programming destination on the user MAT.

When an address in an area other than the flash memory is set, or the start address of the programming destination is not aligned with the 128-byte boundary, an error occurs. The error occurrence is indicated by the WA bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOA31	MOA30	MOA29	MOA28	MOA27	MOA26	MOA25	MOA24

Bit	23	22	21	20	19	18	17	16
Bit Name	MOA23	MOA22	MOA21	MOA20	MOA19	MOA18	MOA17	MOA16

Bit	15	14	13	12	11	10	9	8
Bit Name	MOA15	MOA14	MOA13	MOA12	MOA11	MOA10	MOA9	MOA8

Bit	7	6	5	4	3	2	1	0
Bit Name	MOA7	MOA6	MOA5	MOA4	MOA3	MOA2	MOA1	MOA0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOA31 to MOA0	—	R/W	These bits store the start address of the programming destination on the user MAT. Consecutive 128-byte programming is executed starting from the specified start address of the user MAT. Therefore, the specified start address of the programming destination becomes a 128-byte boundary, and MOA6 to MOA0 are always cleared to 0.

(5) Flash Multipurpose Data Destination Parameter (FMPDR: General Register ER0 of CPU)

FMPDR stores the start address in the area which stores the data to be programmed in the user MAT.

When the storage destination for the program data is in flash memory, an error occurs. The error occurrence is indicated by the WD bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOD31	MOD30	MOD29	MOD28	MOD27	MOD26	MOD25	MOD24

Bit	23	22	21	20	19	18	17	16
Bit Name	MOD23	MOD22	MOD21	MOD20	MOD19	MOD18	MOD17	MOD16

Bit	15	14	13	12	11	10	9	8
Bit Name	MOD15	MOD14	MOD13	MOD12	MOD11	MOD10	MOD9	MOD8

Bit	7	6	5	4	3	2	1	0
Bit Name	MOD7	MOD6	MOD5	MOD4	MOD3	MOD2	MOD1	MOD0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MOD0	—	R/W	These bits store the start address of the area which stores the program data for the user MAT. Consecutive 128-byte data is programmed to the user MAT starting from the specified start address.

(6) Flash Erase Block Select Parameter (FEBS: General Register ER0 of CPU)

- H8SX/1657C

  FEBS specifies the erase block number. Settable values range from 0 to 19 (H'0000 to H'0013). A value of 0 corresponds to block EB0 and a value of 19 corresponds to block EB19. An error occurs when a value over the range (from 0 to 19) is set.

- H8SX/1656C

  FEBS specifies the erase block number. Settable values range from 0 to 15 (H'0000 to H'000F). A value of 0 corresponds to block EB0 and a value of 15 corresponds to block EB15. An error occurs when a value over the range (from 0 to 19) is set.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
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
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

18.7.3 RAM Emulation Register (RAMER)

RAMER specifies the user MAT area overlaid with part of the on-chip RAM (H'FFA000 to H'FFAFFF) when performing emulation of programming the user MAT. RAMER should be set in user mode or user program mode. To ensure dependable emulation, the memory MAT to be emulated must not be accessed immediately after changing the RAMER contents. When accessed at such a timing, correct operation is not guaranteed.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	RAMS	RAM2	RAM1	RAM0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	0	R	Reserved These are read-only bits and cannot be modified.
3	RAMS	0	R/W	RAM Select Selects the function which emulates the flash memory using the on-chip RAM. 0: Disables RAM emulation function 1: Enables RAM emulation function (all blocks of the user MAT are protected against programming and erasing)
2	RAM2	0	R/W	Flash Memory Area Select These bits select the user MAT area overlaid with the on-chip RAM when RAMS = 1. The following areas correspond to the 4-kbyte erase blocks. 000: H'000000 to H'000FFF (EB0) 001: H'001000 to H'001FFF (EB1) 010: H'002000 to H'002FFF (EB2) 011: H'003000 to H'003FFF (EB3) 100: H'004000 to H'004FFF (EB4) 101: H'005000 to H'005FFF (EB5) 110: H'006000 to H'006FFF (EB6) 111: H'007000 to H'007FFF (EB7)
1	RAM1	0	R/W	
0	RAM0	0	R/W	

18.8 On-Board Programming Mode

When the mode pins (MD0, MD1, and MD2) are set to on-board programming mode and the reset start is executed, a transition is made to on-board programming mode in which the on-chip flash memory can be programmed/erased. On-board programming mode has three operating modes: boot mode, user boot mode, and user program mode.

Table 18.5 shows the pin setting for each operating mode. For details on the state transition of each operating mode for flash memory, see figure 18.2.

Table 18.5 On-Board Programming Mode Setting

Mode Setting	MD2	MD1	MD0
User boot mode	0	0	1
Boot mode	0	1	0
User program mode	1	1	0
	1	1	1

18.8.1 Boot Mode

Boot mode executes programming/erasing of the user MAT or user boot MAT by means of the control command and program data transmitted from the externally connected host via the on-chip SCI_4.

In boot mode, the tool for transmitting the control command and program data, and the program data must be prepared in the host. The serial communication mode is set to asynchronous mode. The system configuration in boot mode is shown in figure 18.6. Interrupts are ignored in boot mode. Configure the user system so that interrupts do not occur.

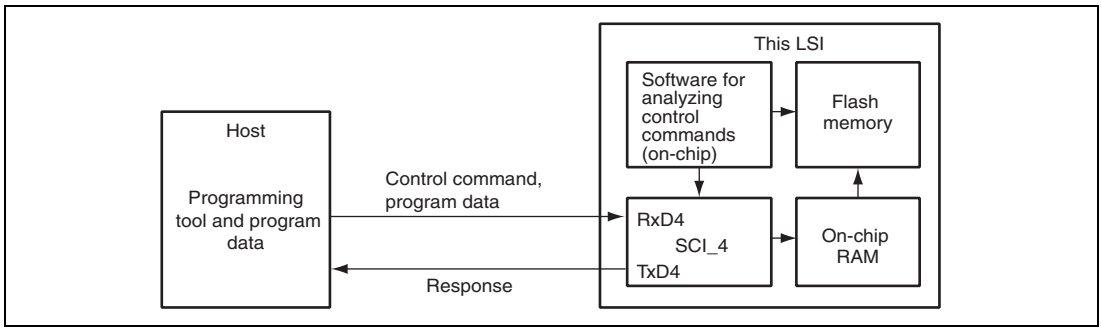


Figure 18.6 System Configuration in Boot Mode

(1) Serial Interface Setting by Host

The SCI_4 is set to asynchronous mode, and the serial transmit/receive format is set to 8-bit data, one stop bit, and no parity.

When a transition to boot mode is made, the boot program embedded in this LSI is initiated.

When the boot program is initiated, this LSI measures the low period of asynchronous serial communication data (H'00) transmitted consecutively by the host, calculates the bit rate, and adjusts the bit rate of the SCI_4 to match that of the host.

When bit rate adjustment is completed, this LSI transmits 1 byte of H'00 to the host as the bit adjustment end sign. When the host receives this bit adjustment end sign normally, it transmits 1 byte of H'55 to this LSI. When reception is not executed normally, initiate boot mode again. The bit rate may not be adjusted within the allowable range depending on the combination of the bit rate of the host and the system clock frequency of this LSI. Therefore, the transfer bit rate of the host and the system clock frequency of this LSI must be as shown in table 18.6.

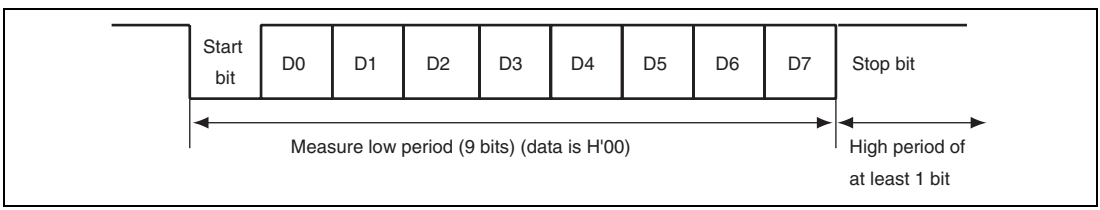


Figure 18.7 Automatic-Bit-Rate Adjustment Operation

Table 18.6 System Clock Frequency for Automatic-Bit-Rate Adjustment

Bit Rate of Host	System Clock Frequency of This LSI
9,600 bps	8 to 18 MHz
19,200 bps	8 to 18 MHz

(2) State Transition Diagram

The state transition after boot mode is initiated is shown in figure 18.8.

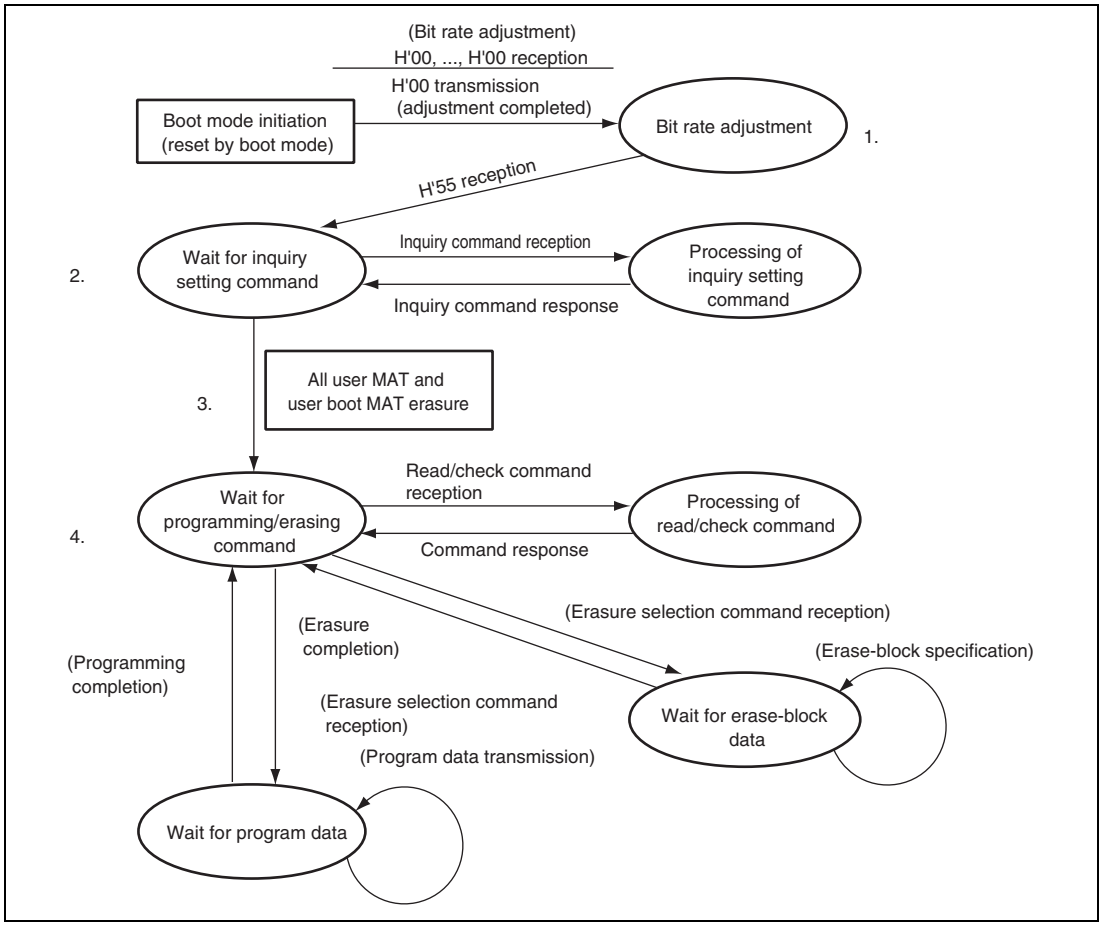


Figure 18.8 Boot Mode State Transition Diagram

1. After boot mode is initiated, the bit rate of the SCI_4 is adjusted with that of the host.
2. Inquiry information about the size, configuration, start address, and support status of the user MAT is transmitted to the host.
3. After inquiries have finished, all user MAT and user boot MAT are automatically erased.
4. When the program preparation notice is received, the state of waiting for program data is entered. The start address of the programming destination and program data must be transmitted after the programming command is transmitted. When programming is finished, the start address of the programming destination must be set to H'FFFFFFFF and transmitted. Then the state of waiting for program data is returned to the state of waiting for programming/erasing command. When reprogramming an erase block including an area on which the programming end command is issued, erase the erase block. An example of the erase block is shown in figure 18.9. When the erasure preparation notice is received, the state of waiting for erase block data is entered. The erase block number must be transmitted after the erasing command is transmitted. When the erasure is finished, the erase block number must be set to H'FF and transmitted. Then the state of waiting for erase block data is returned to the state of waiting for programming/erasing command. Erasure must be executed when the specified block is programmed without a reset start after programming is executed in boot mode. When programming can be executed by only one operation, all blocks are erased before entering the state of waiting for programming/erasing command or another command. Thus, in this case, the erasing operation is not required. The commands other than the programming/erasing command perform sum check, blank check (erasure check), and memory read of the user MAT/user boot MAT and acquisition of current status information.

Memory read of the user MAT/user boot MAT can only read the data programmed after all user MAT/user boot MAT has automatically been erased. No other data can be read.

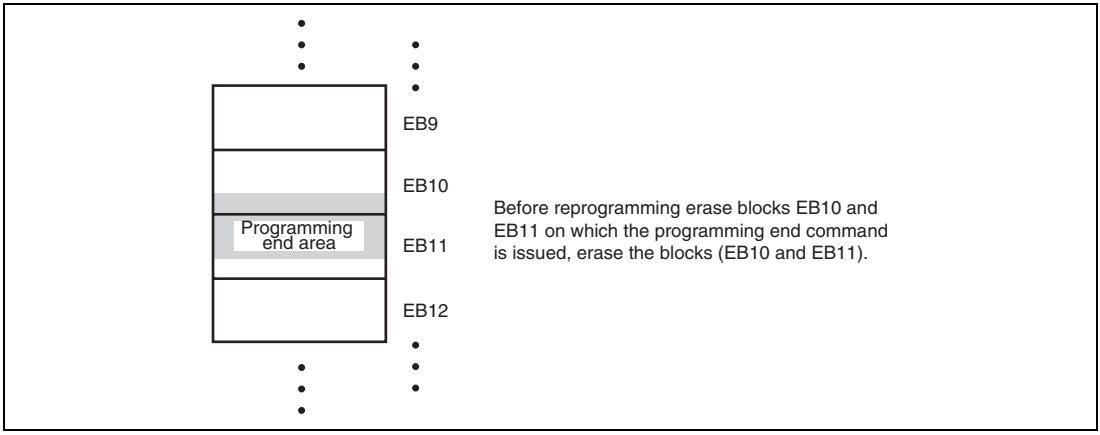


Figure 18.9 Example of Erase Block Including Programmed Area

18.8.2 User Program Mode

Programming/erasing of the user MAT is executed by downloading an on-chip program. The user boot MAT cannot be programmed/erased in user program mode. The programming/erasing flow is shown in figure 18.10.

Since high voltage is applied to the internal flash memory during programming/erasing, a transition to the reset state or hardware standby mode must not be made during programming/erasing. A transition to the reset state or hardware standby mode during programming/erasing may damage the flash memory. If a reset is input, the reset must be released after the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.

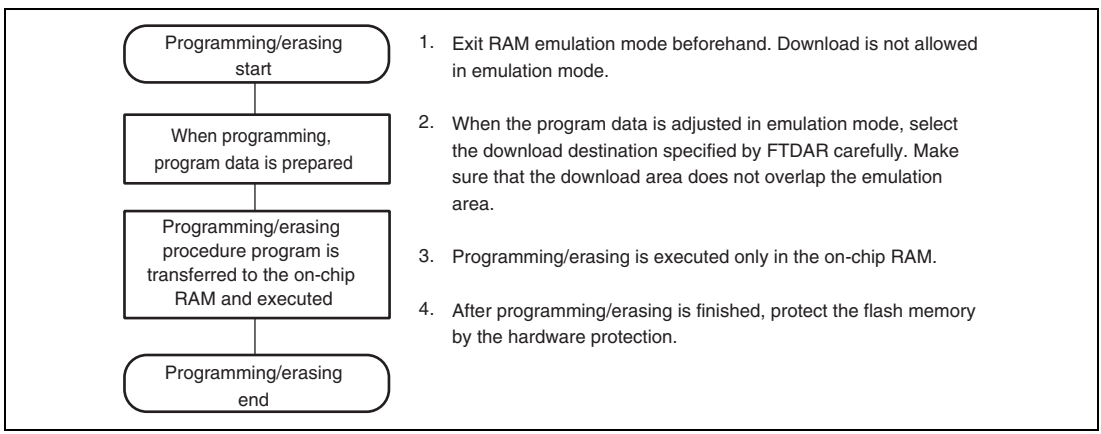


Figure 18.10 Programming/Erasing Flow

(1) On-Chip RAM Address Map when Programming/Erasing is Executed

Parts of the procedure program that is made by the user, like download request, programming/erasing procedure, and decision of the result, must be executed in the on-chip RAM. Since the on-chip program to be downloaded is embedded in the on-chip RAM, make sure the on-chip program and procedure program do not overlap. Figure 18.11 shows the area of the on-chip program to be downloaded.

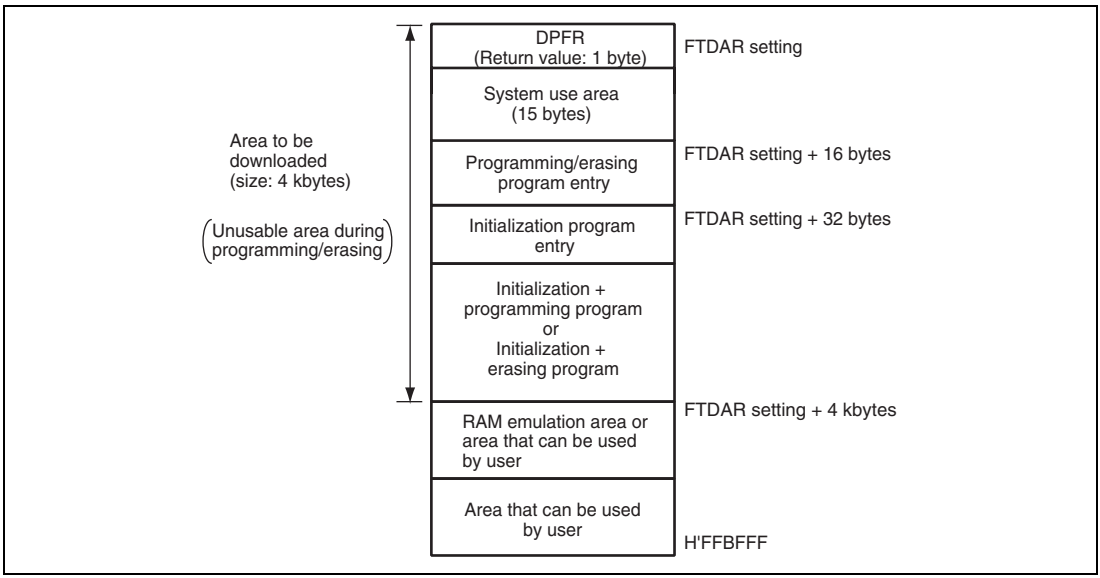


Figure 18.11 RAM Map when Programming/Erasing is Executed

(2) Programming Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and programming are shown in figure 18.12.

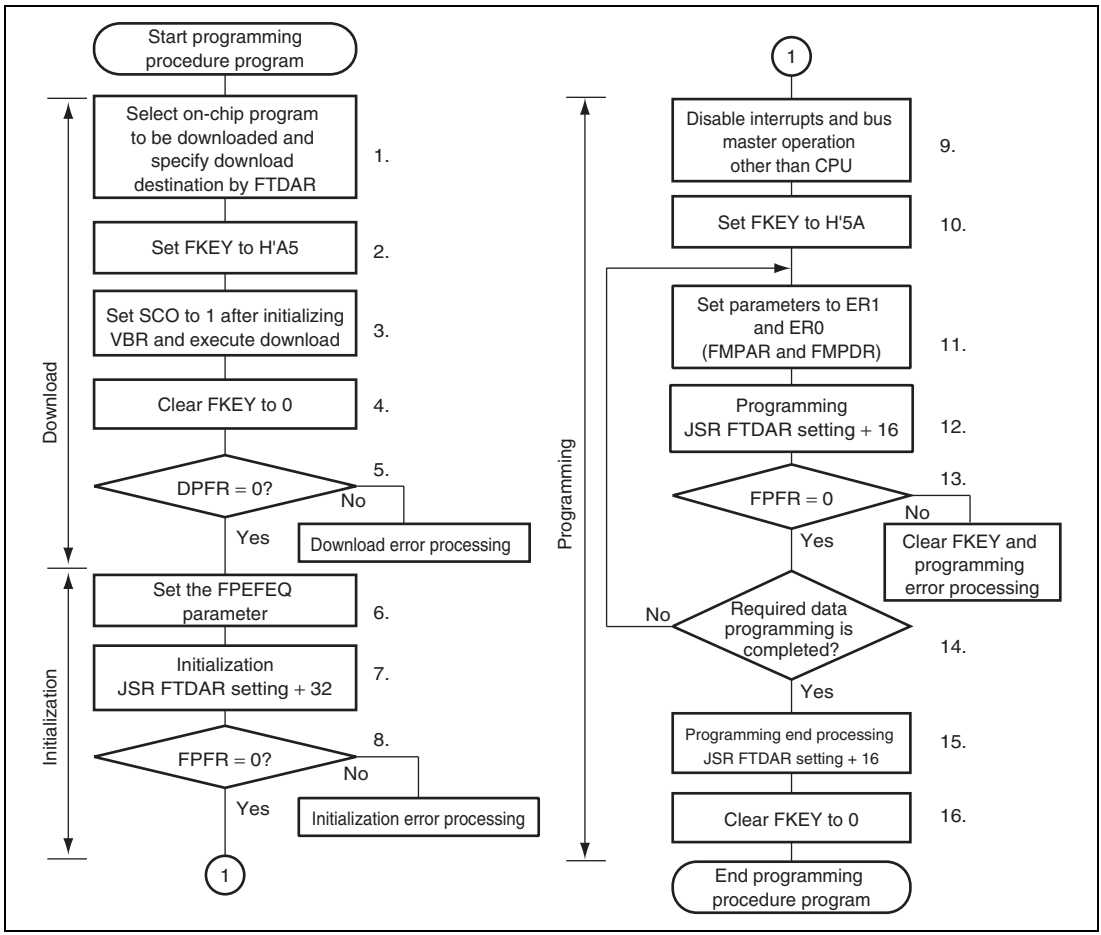


Figure 18.12 Programming Procedure in User Program Mode

The procedure program must be executed in an area other than the flash memory to be programmed. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 18.8.4, On-Chip Program and Storable Area for Program Data. The following description assumes that the area to be programmed on the user MAT is erased and that program data is prepared in the consecutive area.

The program data for one programming operation is always 128 bytes. When the program data exceeds 128 bytes, the start address of the programming destination and program data parameters are updated in 128-byte units and programming is repeated. When the program data is less than 128 bytes, invalid data is filled to prepare 128-byte program data. If the invalid data to be added is H'FF, the program processing time can be shortened.

1. Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.
2. Write H'A5 in FKEY. If H'A5 is not written to FKEY, the SCO bit in FCCS cannot be set to 1 to request download of the on-chip program.
3. After initializing VBR to H'00000000, set the SCO bit to 1 to execute download. To set the SCO bit to 1, all of the following conditions must be satisfied.
   - RAM emulation mode has been canceled.
   - H'A5 is written to FKEY.
   - Setting the SCO bit is executed in the on-chip RAM.

   When the SCO bit is set to 1, download is started automatically. Since the SCO bit is cleared to 0 when the procedure program is resumed, the SCO bit cannot be confirmed to be 1 in the procedure program. The download result can be confirmed by the return value of the DPFR parameter. To prevent incorrect decision, before setting the SCO bit to 1, set one byte of the on-chip RAM start address specified by FTDAR, which becomes the DPFR parameter, to a value other than the return value (e.g. H'FF). Since particular processing that is accompanied by bank switching as described below is performed when download is executed, initialize the VBR contents to H'00000000. Dummy read of FCCS must be performed twice immediately after the SCO bit is set to 1.
   - The user-MAT space is switched to the on-chip program storage area.
   - After the program to be downloaded and the on-chip RAM start address specified by FTDAR are checked, they are transferred to the on-chip RAM.
   - FPCS, FECS, and the SCO bit in FCCS are cleared to 0.

— The return value is set in the DPFR parameter.
— After the on-chip program storage area is returned to the user-MAT space, the procedure program is resumed. After that, VBR can be set again.
— The values of general registers other than ER0 and ER1 are held during download.
— During download, no interrupts can be accepted. However, since the interrupt requests are held, when the procedure program is resumed, the interrupts are requested.
— To hold a level-detection interrupt request, the interrupt must continue to be input until the download is completed.
— Allocate a stack area of 128 bytes at the maximum in the on-chip RAM before setting the SCO bit to 1.
— If access to the flash memory is requested by the DMAC or DTC during download, the operation cannot be guaranteed. Make sure that an access request by the DMAC or DTC is not generated.

4. FKEY is cleared to H'00 for protection.
5. The download result must be confirmed by the value of the DPFR parameter. Check the value of the DPFR parameter (one byte of start address of the download destination specified by FTDAR). If the value of the DPFR parameter is H'00, download has been performed normally. If the value is not H'00, the source that caused download to fail can be investigated by the description below.
   — If the value of the DPFR parameter is the same as that before downloading, the setting of the start address of the download destination in FTDAR may be abnormal. In this case, confirm the setting of the TDER bit in FTDAR.
   — If the value of the DPFR parameter is different from that before downloading, check the SS bit or FK bit in the DPFR parameter to confirm the download program selection and FKEY setting, respectively.
6. The operating frequency of the CPU is set in the FPEFEQ parameter for initialization. The settable operating frequency of the FPEFEQ parameter ranges from 8 to 35 MHz. When the frequency is set otherwise, an error is returned to the FPFR parameter of the initialization program and initialization is not performed. For details on setting the frequency, see section 18.7.2 (3), Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU).

7. Initialization is executed. The initialization program is downloaded together with the programming program to the on-chip RAM. The entry point of the initialization program is at the address which is 32 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute initialization by using the following steps.

```
MOV.L #DLTOP+32,ER2      ; Set entry address to ER2
JSR   @ER2               ; Call initialization routine
NOP
```

   — The general registers other than ER0 and ER1 are held in the initialization program.
   — R0L is a return value of the FPFR parameter.
   — Since the stack area is used in the initialization program, a stack area of 128 bytes at the maximum must be allocated in RAM.
   — Interrupts can be accepted during execution of the initialization program. Make sure the program storage area and stack area in the on-chip RAM and register values are not overwritten.
8. The return value in the initialization program, the FPFR parameter is determined.
9. All interrupts and the use of a bus master other than the CPU are disabled during programming/erasing. The specified voltage is applied for the specified time when programming or erasing. If interrupts occur or the bus mastership is moved to other than the CPU during programming/erasing, causing a voltage exceeding the specifications to be applied, the flash memory may be damaged. Therefore, interrupts are disabled by setting bit 7 (I bit) in the condition code register (CCR) to B'1 in interrupt control mode 0 and by setting bits 2 to 0 (I2 to I0 bits) in the extend register (EXR) to B'111 in interrupt control mode 2. Accordingly, interrupts other than NMI are held and not executed. Configure the user system so that NMI interrupts do not occur. The interrupts that are held must be executed after all programming completes. When the bus mastership is moved to other than the CPU, such as to the DMAC or DTC, the error protection state is entered. Therefore, make sure the DMAC does not acquire the bus.
10. FKEY must be set to H'5A and the user MAT must be prepared for programming.
11. The parameters required for programming are set. The start address of the programming destination on the user MAT (FMPAR parameter) is set in general register ER1. The start address of the program data storage area (FMPDR parameter) is set in general register ER0.
   — Example of FMPAR parameter setting: When an address other than one in the user MAT area is specified for the start address of the programming destination, even if the programming program is executed, programming is not executed and an error is returned to the FPFR parameter. Since the program data for one programming operation is 128 bytes, the lower eight bits of the address must be H'00 or H'80 to be aligned with the 128-byte boundary.

— Example of FMPDR parameter setting: When the storage destination for the program data is flash memory, even if the programming routine is executed, programming is not executed and an error is returned to the FPFR parameter. In this case, the program data must be transferred to the on-chip RAM and then programming must be executed.

12. Programming is executed. The entry point of the programming program is at the address which is 16 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute programming by using the following steps.

```
MOV.L    #DLTOP+16,ER2   ; Set entry address to ER2
JSR      @ER2            ; Call programming routine
NOP
```

— The general registers other than ER0 and ER1 are held in the programming program.

— R0L is a return value of the FPFR parameter.

— Since the stack area is used in the programming program, a stack area of 128 bytes at the maximum must be allocated in RAM.

13. The return value in the programming program, the FPFR parameter is determined.

14. Determine whether programming of the necessary data has finished. If more than 128 bytes of data are to be programmed, update the FMPAR and FMPDR parameters in 128-byte units, and repeat steps 11 to 14. Increment the programming destination address by 128 bytes and update the programming data pointer correctly. If an address which has already been programmed is written to again, not only will a programming error occur, but also flash memory will be damaged.

15. Programming end processing is executed.

The entry point of the programming library is at the address which is 16 bytes after the start address of the download destination specified by FTDAR. Call the subroutine by using the following steps.

```
MOV.L    #H'F0F0F0F0,ER0
MOV.L    #H'0F0F0F0F,ER1
MOV.L    #DLTOP+16,ER2   ; Set entry address to ER2
JSR      @ER2            ; Call programming end routine
```

— The general registers other than ER0 and ER1 are held in the programming end program.

— R0L is a return value of the FPFR parameter.

— Since the stack area is used in the programming end program, a stack area of 128 bytes at the maximum must be allocated in RAM.

— When reprogramming an erase block including an area on which the programming end processing is executed, erase the erase block. An example of the erase block is shown in figure 18.13.

— Make sure that the programming end processing has been performed on completion of the programming. Before executing the programming end processing, the following processing are not allowed: executing initialization, downloading an internal program, writing to a RAM area which is a download destination, and switching MATs. If attempted, programming is not performed correctly.

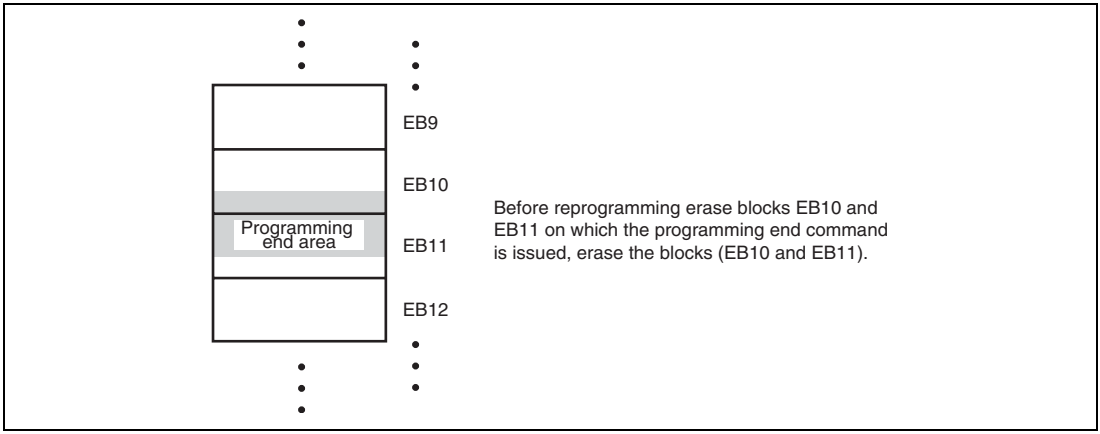


Figure 18.13 Example of Erase Block Including Programmed Area

16. After programming finishes, clear FKEY and specify software protection. If this LSI is restarted by a reset immediately after programming has finished, secure the reset input period (period of $\overline{RES} = 0$) of at least 100 μs.

(3) Erasing Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and erasing are shown in figure 18.14.

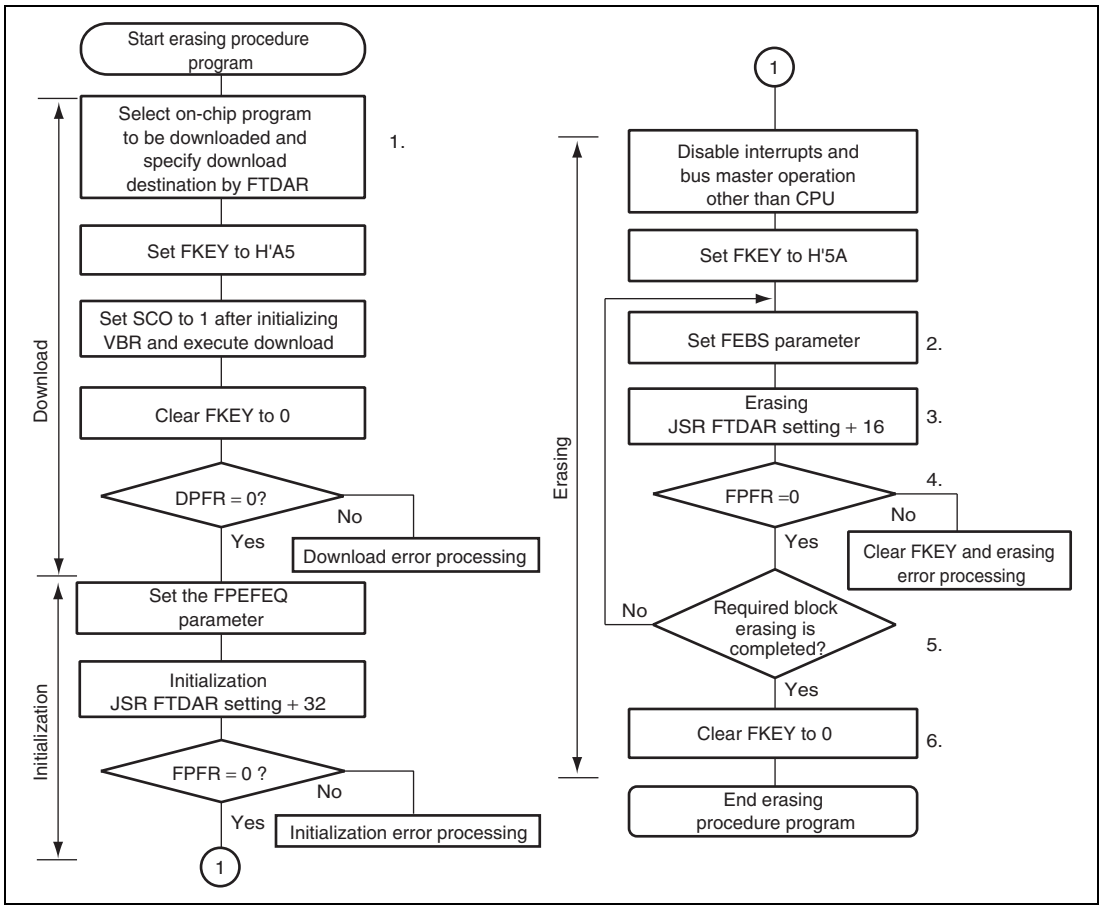


Figure 18.14 Erasing Procedure in User Program Mode

The procedure program must be executed in an area other than the user MAT to be erased. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 18.8.4, On-Chip Program and Storable Area for Program Data. For the downloaded on-chip program area, see figure 18.11.

One erasure processing erases one block. For details on block divisions, refer to figure 18.4. To erase two or more blocks, update the erase block number and repeat the erasing processing for each block.

1. Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.

   For the procedures to be carried out after setting FKEY, see section 18.8.2 (2), Programming Procedure in User Program Mode.
2. Set the FEBS parameter necessary for erasure. Set the erase block number (FEBS parameter) of the user MAT in general register ER0. If a value other than an erase block number of the user MAT is set, no block is erased even though the erasing program is executed, and an error is returned to the FPFR parameter.
3. Erasure is executed. Similar to as in programming, the entry point of the erasing program is at the address which is 16 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute erasure by using the following steps.

```
MOV.L #DLTOP+16, ER2      ; Set entry address to ER2
JSR   @ER2                ; Call erasing routine
NOP
```

   - The general registers other than ER0 and ER1 are held in the erasing program.
   - R0L is a return value of the FPFR parameter.
   - Since the stack area is used in the erasing program, a stack area of 128 bytes at the maximum must be allocated in RAM.
4. The return value in the erasing program, the FPFR parameter is determined.
5. Determine whether erasure of the necessary blocks has finished. If more than one block is to be erased, update the FEBS parameter and repeat steps 2 to 5.
6. After erasure completes, clear FKEY and specify software protection. If this LSI is restarted by a power-on reset immediately after erasure has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.

(4) Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

By changing the on-chip RAM start address of the download destination in FTDAR, the erasing program and programming program can be downloaded to separate on-chip RAM areas.

Figure 18.15 shows a repeating procedure of erasing, programming, and RAM emulation.

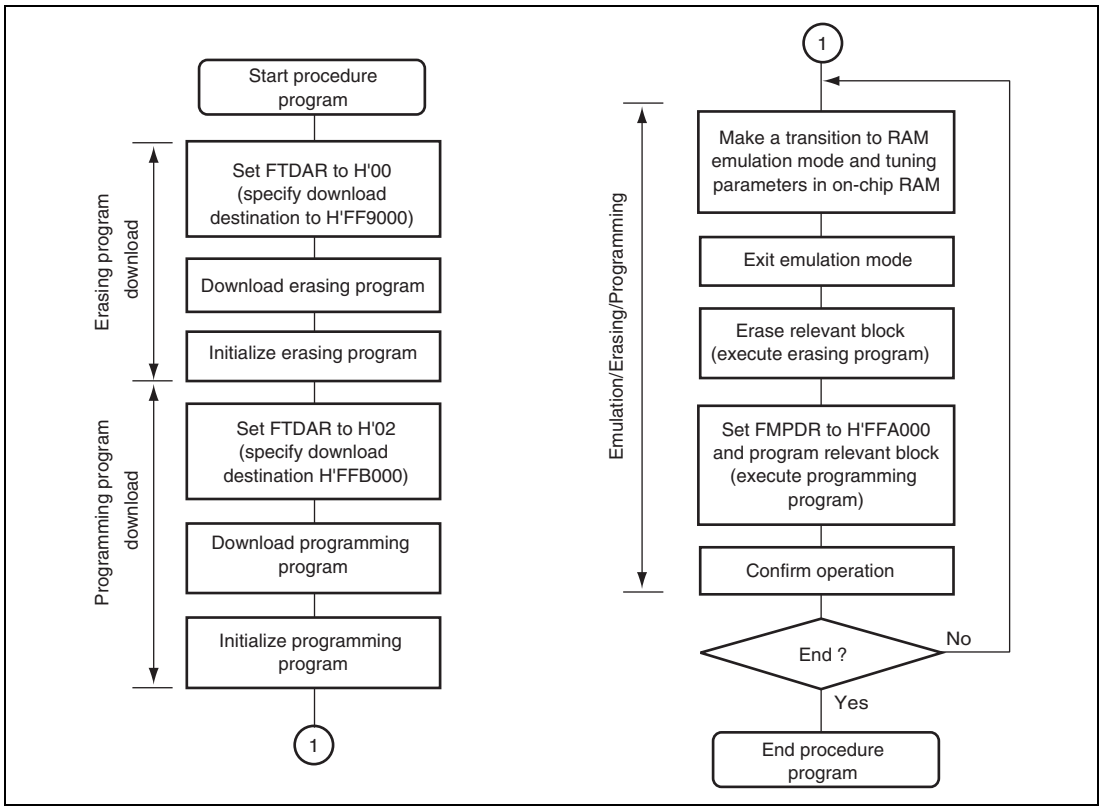


Figure 18.15 Repeating Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

In figure 18.15, since RAM emulation is performed, the erasing/programming program is downloaded to avoid the 4-kbyte on-chip RAM area (H'FFA000 to H'FFAFFF). Download and initialization are performed only once at the beginning. Note the following when executing the procedure program.

- Be careful not to overwrite data in the on-chip RAM with overlay settings. In addition to the programming program area, erasing program area, and RAM emulation area, areas for the procedure programs, work area, and stack area are reserved in the on-chip RAM. Do not make settings that will overwrite data in these areas.
- Be sure to initialize both the programming program and erasing program. When the FPEFEQ parameter is initialized, also initialize both the erasing program and programming program. Initialization must be executed for both entry addresses: #DLTOP (start address of download destination for erasing program) + 32 bytes, and #DLTOP (start address of download destination for programming program) + 32 bytes.

18.8.3 User Boot Mode

Branching to a programming/erasing program prepared by the user enables user boot mode which is a user-arbitrary boot mode to be used.

Only the user MAT can be programmed/erased in user boot mode. Programming/erasing of the user boot MAT is only enabled in boot mode or programmer mode.

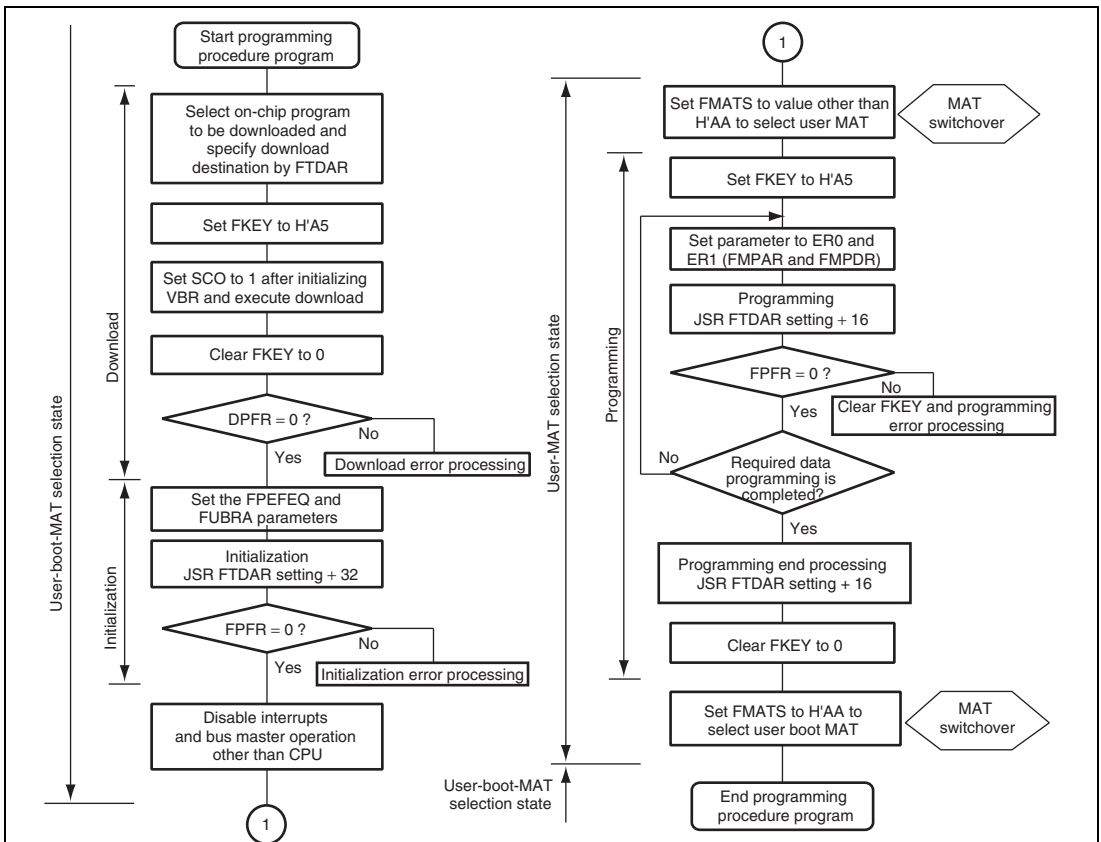
(1) Initiation in User Boot Mode

When the reset start is executed with the mode pins set to user boot mode, the built-in check routine runs and checks the user MAT and user boot MAT states. While the check routine is running, NMI and all other interrupts cannot be accepted. Next, processing starts from the execution start address of the reset vector in the user boot MAT. At this point, the user boot MAT is selected (FMATS = H'AA) as the execution memory MAT.

(2) User MAT Programming in User Boot Mode

Figure 18.16 shows the procedure for programming the user MAT in user boot mode.

The difference between the programming procedures in user program mode and user boot mode is the memory MAT switching as shown in figure 18.16. For programming the user MAT in user boot mode, additional processing made by setting FMATS is required: switching from the user boot MAT to the user MAT, and switching back to the user boot MAT after programming completes.



Note: The MAT must be switched by FMATS to perform the programming error processing in the user boot MAT.

Figure 18.16 Procedure for Programming User MAT in User Boot Mode

In user boot mode, though the user boot MAT can be seen in the flash memory space, the user MAT is hidden in the background. Therefore, the user MAT and user boot MAT are switched while the user MAT is being programmed. Because the user boot MAT is hidden while the user MAT is being programmed, the procedure program must be executed in an area other than flash memory. After programming completes, switch the memory MATs again to return to the first state.

Memory MAT switching is enabled by setting FMATS. However note that access to a memory MAT is not allowed until memory MAT switching is completed. During memory MAT switching, the LSI is in an unstable state, e.g. if an interrupt occurs, from which memory MAT the interrupt vector is read is undetermined. Perform memory MAT switching in accordance with the description in section 18.11, Switching between User MAT and User Boot MAT.

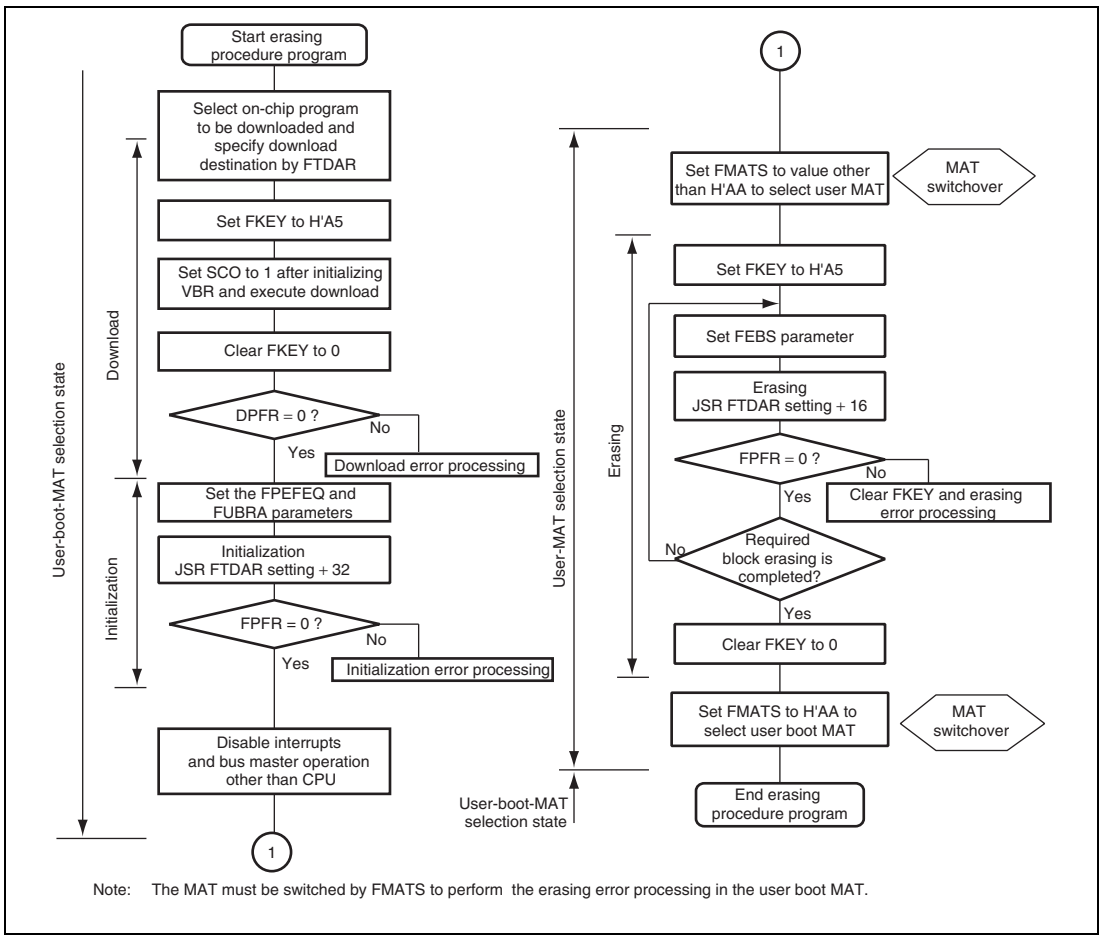
Except for memory MAT switching, the programming procedure is the same as that in user program mode.

The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 18.8.4, On-Chip Program and Storable Area for Program Data.

(3) User MAT Erasing in User Boot Mode

Figure 18.17 shows the procedure for erasing the user MAT in user boot mode.

The difference between the erasing procedures in user program mode and user boot mode is the memory MAT switching as shown in figure 18.17. For erasing the user MAT in user boot mode, additional processing made by setting FMATS is required: switching from the user boot MAT to the user MAT, and switching back to the user boot MAT after erasing completes.



Note: The MAT must be switched by FMATS to perform the erasing error processing in the user boot MAT.

Figure 18.17 Procedure for Erasing User MAT in User Boot Mode

Memory MAT switching is enabled by setting FMATS. However note that access to a memory MAT is not allowed until memory MAT switching is completed. During memory MAT switching, the LSI is in an unstable state, e.g. if an interrupt occurs, from which memory MAT the interrupt vector is read is undetermined. Perform memory MAT switching in accordance with the description in section 18.11, Switching between User MAT and User Boot MAT.

Except for memory MAT switching, the erasing procedure is the same as that in user program mode.

The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 18.8.4, On-Chip Program and Storable Area for Program Data.

18.8.4 On-Chip Program and Storable Area for Program Data

In the descriptions in this manual, the on-chip programs and program data storage areas are assumed to be in the on-chip RAM. However, they can be executed from part of the flash memory which is not to be programmed or erased as long as the following conditions are satisfied.

- The on-chip program is downloaded to and executed in the on-chip RAM specified by FTDAR. Therefore, this on-chip RAM area is not available for use.
- Since the on-chip program uses a stack area, allocate 128 bytes at the maximum as a stack area.
- Download requested by setting the SCO bit in FCCS to 1 should be executed from the on-chip RAM because it will require switching of the memory MATs.
- In an operating mode in which the external address space is not accessible, such as single-chip mode, the required procedure programs, NMI handling vector table, and NMI handling routine should be transferred to the on-chip RAM before programming/erasing starts (download result is determined).
- The flash memory is not accessible during programming/erasing. Programming/erasing is executed by the program downloaded to the on-chip RAM. Therefore, the procedure program that initiates operation, the NMI handling vector table, and the NMI handling routine should be stored in the on-chip RAM other than the flash memory.
- After programming/erasing starts, access to the flash memory should be inhibited until FKEY is cleared. The reset input state (period of $\overline{\text{RES}}$ = 0) must be set to at least 100 μs when the operating mode is changed and the reset start executed on completion of programming/erasing. Transitions to the reset state are inhibited during programming/erasing. When the reset signal is input, a reset input state (period of $\overline{\text{RES}}$ = 0) of at least 100 μs is needed before the reset signal is released.

- Switching of the memory MATs by FMATS should be needed when programming/erasing of the user MAT is operated in user boot mode. The program which switches the memory MATs should be executed from the on-chip RAM. For details, see section 18.11, Switching between User MAT and User Boot MAT. Make sure you know which memory MAT is currently selected when switching them.
- When the program data storage area is within the flash memory area, an error will occur even when the data stored is normal program data. Therefore, the data should be transferred to the on-chip RAM to place the address that the FMPDR parameter indicates in an area other than the flash memory.

In consideration of these conditions, the areas in which the program data can be stored and executed are determined by the combination of the processing contents, operating mode, and bank structure of the memory MATs, as shown in tables 18.7 to 18.11.

Table 18.7 Executable Memory MAT

	Operating Mode	
Processing Contents	**User Program Mode**	**User Boot Mode***
Programming	See table 18.8	See table 18.10
Erasing	See table 18.9	See table 18.11

Note: * Programming/Erasing is possible to the user MAT.

Table 18.8 Usable Area for Programming in User Program Mode

	Storable/Executable Area		Selected MAT	
Item	**On-Chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage MAT**
Storage area for program data	O	×*	—	—
Operation for selecting on-chip program to be downloaded	O	O	O	
Operation for writing H'A5 to FKEY	O	O	O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×		O
Operation for clearing FKEY	O	O	O	
Decision of download result	O	O	O	
Operation for download error	O	O	O	
Operation for setting initialization parameter	O	O	O	
Execution of initialization	O	×	O	
Decision of initialization result	O	O	O	
Operation for initialization error	O	O	O	
NMI handling routine	O	×	O	
Operation for disabling interrupts	O	O	O	
Operation for writing H'5A to FKEY	O	O	O	
Operation for setting programming parameter	O	×	O	
Execution of programming	O	×	O	
Decision of programming result	O	×	O	
Operation for programming error	O	×	O	
Operation for clearing FKEY	O	×	O	

Note: * Transferring the program data to the on-chip RAM beforehand enables this area to be used.

Table 18.9 Usable Area for Erasure in User Program Mode

	Storable/Executable Area		Selected MAT	
Item	**On-Chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage MAT**
Operation for selecting on-chip program to be downloaded	O	O	O	
Operation for writing H'A5 to FKEY	O	O	O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×		O
Operation for clearing FKEY	O	O	O	
Decision of download result	O	O	O	
Operation for download error	O	O	O	
Operation for setting initialization parameter	O	O	O	
Execution of initialization	O	×	O	
Decision of initialization result	O	O	O	
Operation for initialization error	O	O	O	
NMI handling routine	O	×	O	
Operation for disabling interrupts	O	O	O	
Operation for writing H'5A to FKEY	O	O	O	
Operation for setting erasure parameter	O	×	O	
Execution of erasure	O	×	O	
Decision of erasure result	O	×	O	
Operation for erasure error	O	×	O	
Operation for clearing FKEY	O	×	O	

Table 18.10 Usable Area for Programming in User Boot Mode

	Storable/Executable Area			Selected MAT	
Item	**On-Chip RAM**	**User Boot MAT**	**User MAT**	**User Boot MAT**	**Embedded Program Storage MAT**
Storage area for program data	O	×*1	—	—	—
Operation for selecting on-chip program to be downloaded	O	O		O	
Operation for writing H'A5 to FKEY	O	O		O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×			O
Operation for clearing FKEY	O	O		O	
Decision of download result	O	O		O	
Operation for download error	O	O		O	
Operation for setting initialization parameter	O	O		O	
Execution of initialization	O	×		O	
Decision of initialization result	O	O		O	
Operation for initialization error	O	O		O	
NMI handling routine	O	×		O	
Operation for disabling interrupts	O	O		O	
Switching memory MATs by FMATS	O	×	O		
Operation for writing H'5A to FKEY	O	×	O		
Operation for setting programming parameter	O	×	O		
Execution of programming	O	×	O		
Decision of programming result	O	×	O		
Operation for programming error	O	×*2	O		
Operation for clearing FKEY	O	×	O		
Switching memory MATs by FMATS	O	×		O	

Notes: 1. Transferring the program data to the on-chip RAM beforehand enables this area to be used.

2. Switching memory MATs by FMATS by a program in the on-chip RAM enables this area to be used.

Table 18.11 Usable Area for Erasure in User Boot Mode

	Storable/Executable Area			Selected MAT	
Item	On-Chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage MAT
Operation for selecting on-chip program to be downloaded	O	O		O	
Operation for writing H'A5 to FKEY	O	O		O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×			O
Operation for clearing FKEY	O	O		O	
Decision of download result	O	O		O	
Operation for download error	O	O		O	
Operation for setting initialization parameter	O	O		O	
Execution of initialization	O	×		O	
Decision of initialization result	O	O		O	
Operation for initialization error	O	O		O	
NMI handling routine	O	×		O	
Operation for disabling interrupts	O	O		O	
Switching memory MATs by FMATS	O	×	O		
Operation for writing H'5A to FKEY	O	×	O		
Operation for setting erasure parameter	O	×	O		
Execution of erasure	O	×	O		
Decision of erasure result	O	×	O		
Operation for erasure error	O	×*	O		
Operation for clearing FKEY	O	×	O		
Switching memory MATs by FMATS	O	×	O		

Note: Switching memory MATs by FMATS by a program in the on-chip RAM enables this area to be used.

18.9 Protection

There are three types of protection against the flash memory programming/erasing: hardware protection, software protection, and error protection.

18.9.1 Hardware Protection

Programming and erasure of the flash memory is forcibly disabled or suspended by hardware protection. In this state, download of an on-chip program and initialization are possible. However, programming or erasure of the user MAT cannot be performed even if the programming/erasing program is initiated, and the error in programming/erasing is indicated by the FPFR parameter.

Table 18.12 Hardware Protection

		Function to be Protected	
Item	**Description**	**Download**	**Programming/ Erasing**
Reset protection	• The programming/erasing interface registers are initialized in the reset state (including a reset by the WDT) and the programming/erasing protection state is entered. • The reset state will not be entered by a reset using the $\overline{\text{RES}}$ pin unless the $\overline{\text{RES}}$ pin is held low until oscillation has settled after a power is initially supplied. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width given in the AC characteristics. If a reset is input during programming or erasure, data in the flash memory is not guaranteed. In this case, execute erasure and then execute programming again.	O	O

18.9.2 Software Protection

The software protection protects the flash memory against programming/erasing by disabling download of the programming/erasing program, using the key code, and by the RAMER setting.

Table 18.13 Software Protection

		Function to be Protected	
Item	**Description**	**Download**	**Programming/ Erasing**
Protection by SCO bit	The programming/erasing protection state is entered when the SCO bit in FCCS is cleared to 0 to disable download of the programming/erasing programs.	O	O
Protection by FKEY	The programming/erasing protection state is entered because download and programming/erasing are disabled unless the required key code is written in FKEY.	O	O
Emulation protection	The programming/erasing protection state is entered when the RAMS bit in the RAM emulation register (RAMER) is set to 1.	O	O

18.9.3 Error Protection

Error protection is a mechanism for aborting programming or erasure when a CPU runaway occurs or operations not according to the programming/erasing procedures are detected during programming/erasing of the flash memory. Aborting programming or erasure in such cases prevents damage to the flash memory due to excessive programming or erasing.

If an error occurs during programming/erasing of the flash memory, the FLER bit in FCCS is set to 1 and the error protection state is entered.

- When an interrupt request, such as NMI, occurs during programming/erasing.
- When the flash memory is read from during programming/erasing (including a vector read or an instruction fetch).
- When a SLEEP instruction is executed (including software-standby mode) during programming/erasing.
- When a bus master other than the CPU, such as the DMAC and DTC, obtains bus mastership during programming/erasing.

Error protection is canceled by a reset. Note that the reset should be released after the reset input period of at least 100μs has passed. Since high voltages are applied during programming/erasing of the flash memory, some voltage may remain after the error protection state has been entered. For this reason, it is necessary to reduce the risk of damaging the flash memory by extending the reset input period so that the charge is released.

The state-transition diagram in figure 18.18 shows transitions to and from the error protection state.

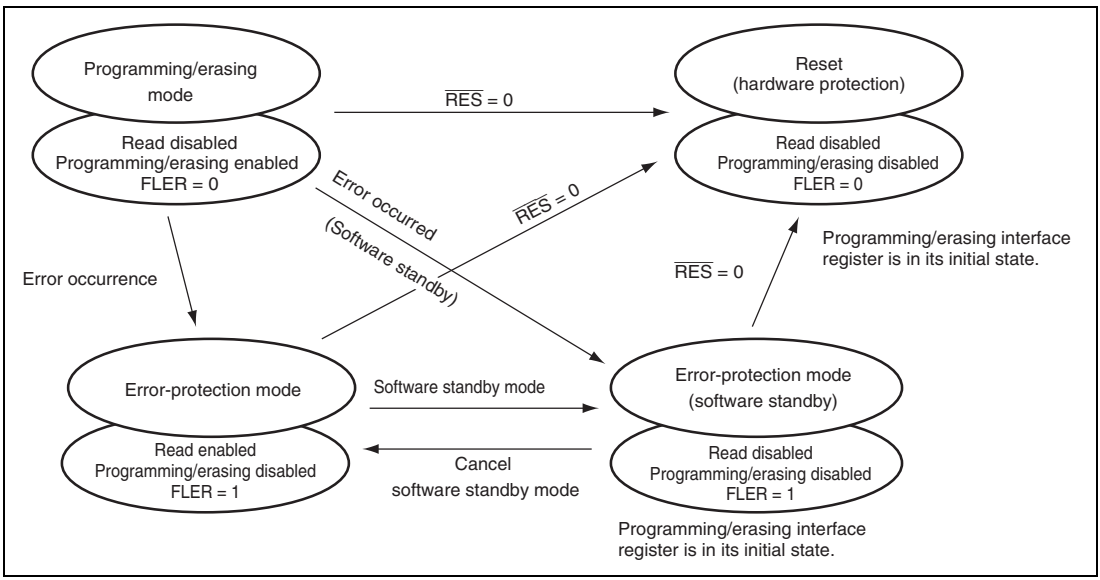


Figure 18.18 Transitions to Error Protection State

18.10 Flash Memory Emulation Using RAM

For realtime emulation of the data written to the flash memory using the on-chip RAM, the on-chip RAM area can be overlaid with several flash memory blocks (user MAT) using the RAM emulation register (RAMER).

The overlaid area can be accessed from both the user MAT area specified by RAMER and the overlaid RAM area. The emulation can be performed in user mode and user program mode.

Figure 18.19 shows an example of emulating realtime programming of the user MAT.

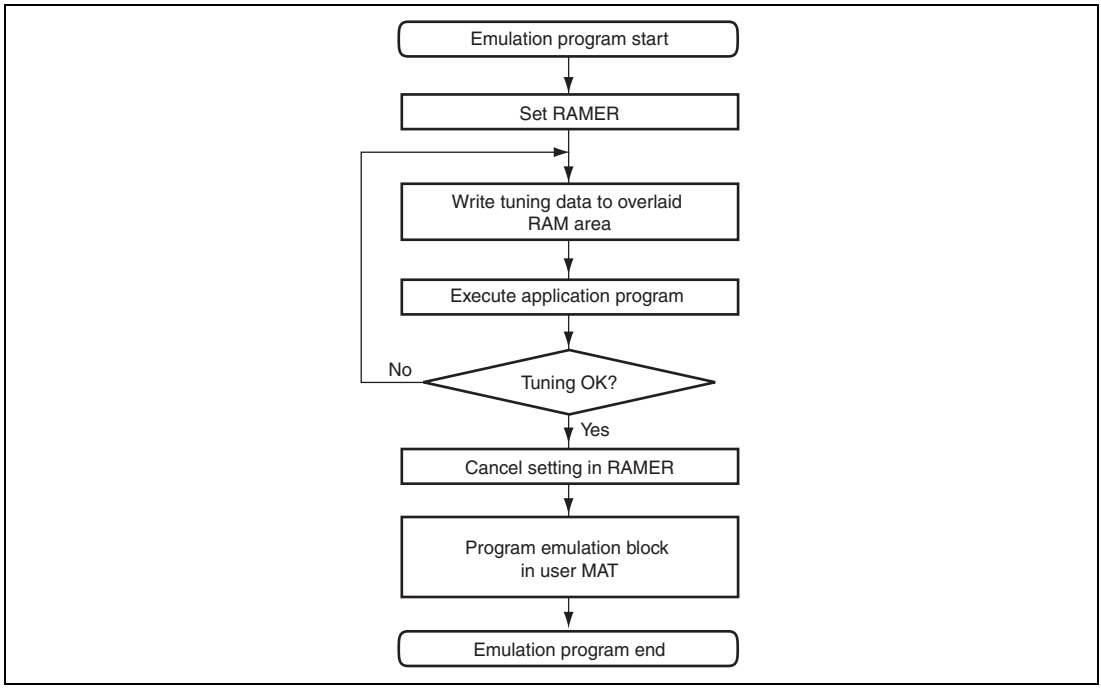


Figure 18.19 RAM Emulation Flow

Figure 18.20 shows an example of overlaying flash memory block area EB0.

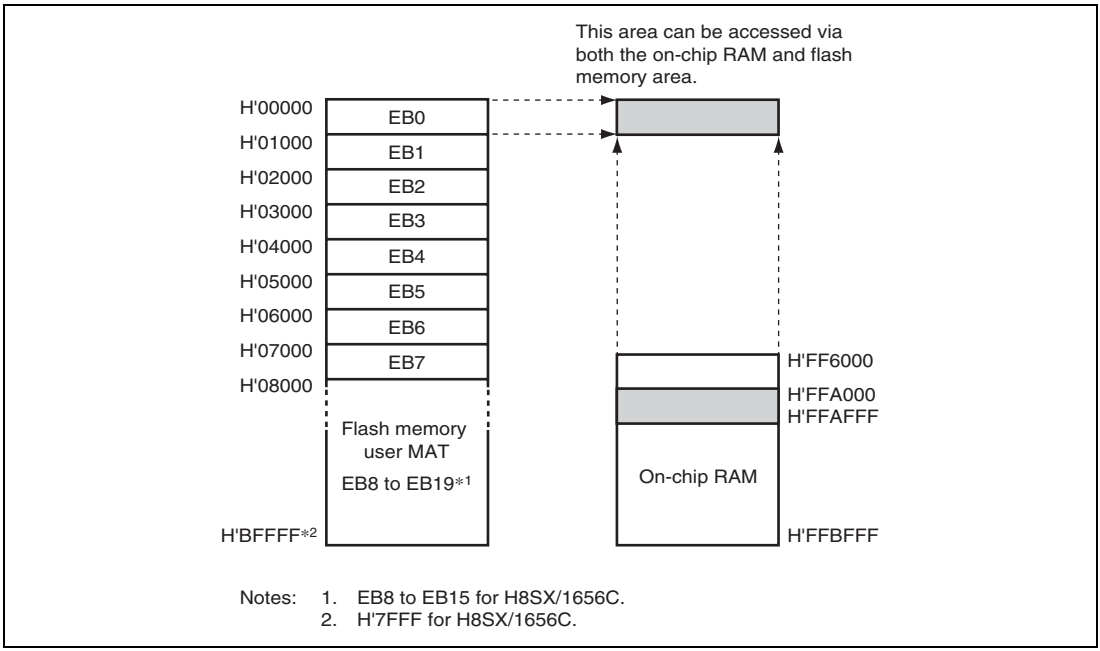


Figure 18.20 Address Map of Overlaid RAM Area (H8SX/1657C)

The flash memory area that can be emulated is the one area selected by bits RAM2 to RAM0 in RAMER from among the eight blocks, EB0 to EB7, of the user MAT.

To overlay a part of the on-chip RAM with block EB0 for realtime emulation, set the RAMS bit in RAMER to 1 and bits RAM2 to RAM0 to B'000.

For programming/erasing the user MAT, the procedure programs including a download program of the on-chip program must be executed. At this time, the download area should be specified so that the overlaid RAM area is not overwritten by downloading the on-chip program. Since the area in which the tuned data is stored is overlaid with the download area when FTDAR = H'01, the tuned data must be saved in an unused area beforehand.

Figure 18.21 shows an example of the procedure to program the tuned data in block EB0 of the user MAT.

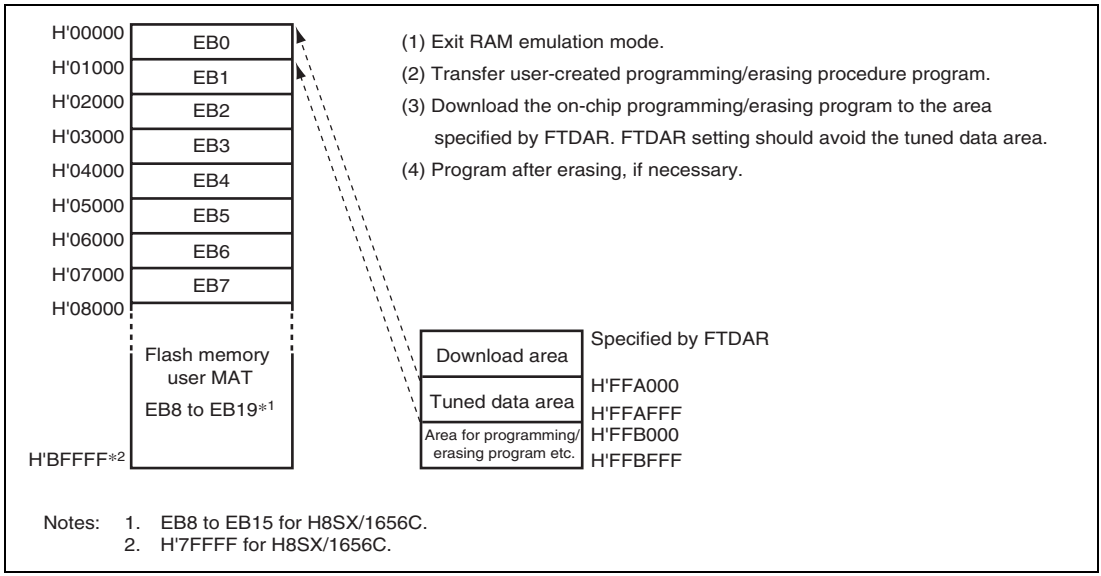


Figure 18.21 Programming Tuned Data (H8SX/1657C)

1. After tuning program data is completed, clear the RAMS bit in RAMER to 0 to cancel the overlaid RAM.
2. Transfer the user-created procedure program to the on-chip RAM.
3. Start the procedure program and download the on-chip program to the on-chip RAM. The start address of the download destination should be specified by FTDAR so that the tuned data area does not overlay the download area.
4. When block EB0 of the user MAT has not been erased, the programming program must be downloaded after block EB0 is erased. Specify the tuned data saved in the FMPAR and FMPDR parameters and then execute programming.

Note: Setting the RAMS bit to 1 makes all the blocks of the user MAT enter the programming/erasing protection state (emulation protection state) regardless of the setting of the RAM2 to RAM0 bits. Under this condition, the on-chip program cannot be downloaded. When data is to be actually programmed and erased, clear the RAMS bit to 0.

18.11 Switching between User MAT and User Boot MAT

It is possible to switch between the user MAT and user boot MAT. However, the following procedure is required because the start addresses of these MATs are allocated to the same address.

Switching to the user boot MAT disables programming and erasing. Programming of the user boot MAT should take place in boot mode or programmer mode.

1. Memory MAT switching by FMATS should always be executed from the on-chip RAM.
2. When accessing the memory MAT immediately after switching the memory MATs by FMATS from the on-chip RAM, similarly execute the NOP instruction in the on-chip RAM for eight times (this prevents access to the flash memory during memory MAT switching).
3. If an interrupt request has occurred during memory MAT switching, there is no guarantee of which memory MAT is accessed. Always mask the maskable interrupts before switching memory MATs. In addition, configure the system so that NMI interrupts do not occur during memory MAT switching.
4. After the memory MATs have been switched, take care because the interrupt vector table will also have been switched. If interrupt processing is to be the same before and after memory MAT switching, transfer the interrupt processing routines to the on-chip RAM and specify VBR to place the interrupt vector table in the on-chip RAM.
5. The size of the user MAT is different from that of the user boot MAT. Addresses which exceed the size of the 8-kbyte user boot MAT should not be accessed. If an attempt is made, data is read as an undefined value.

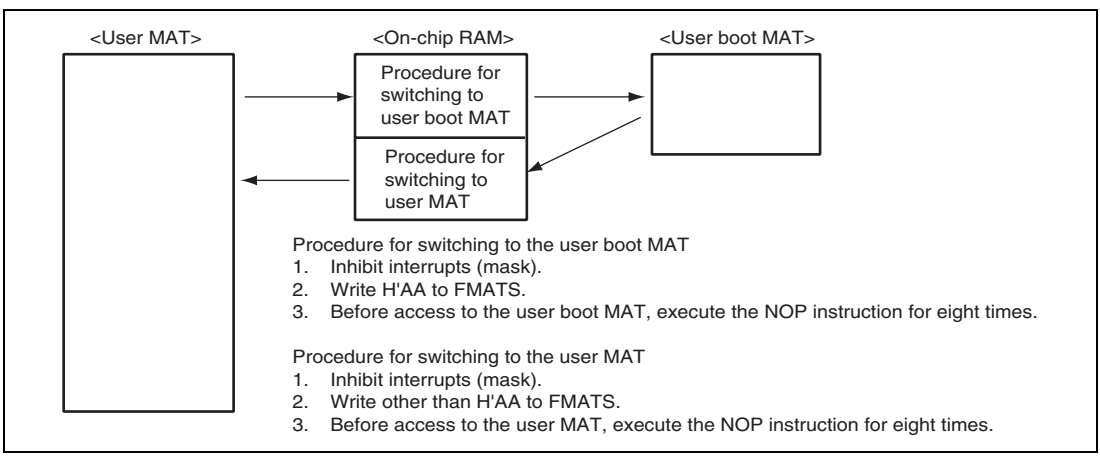


Figure 18.22 Switching between User MAT and User Boot MAT

18.12 Programmer Mode

Along with its on-board programming mode, this LSI also has a programmer mode as a further mode for the writing and erasing of programs and data. In programmer mode, a general-purpose PROM programmer that supports the device types shown in table 18.14 can be used to write programs to the on-chip ROM without any limitation.

Table 18.14 Device Types Supported in Programmer Mode

Target Memory MAT	Product Classification	ROM Size	Device Type
User MAT	H8SX/1657C	768 Kbytes	FZTAT1024V3A
	H8SX/1656C	512 Kbytes	
User boot MAT	H8SX/1657C	8 Kbytes	FZTATUSBTV3A
	H8SX/1656C		

18.13 Standard Serial Communication Interface Specifications for Boot Mode

The boot program initiated in boot mode performs serial communication using the host and on-chip SCI_4. The serial communication interface specifications are shown below.

The boot program has three states.

1. Bit-rate-adjustment state

   In this state, the boot program adjusts the bit rate to achieve serial communication with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, the program enters the inquiry/selection state.
2. Inquiry/selection state

   In this state, the boot program responds to inquiry commands from the host. The device name, clock mode, and bit rate are selected. After selection of these settings, the program is made to enter the programming/erasing state by the command for a transition to the programming/erasing state. The program transfers the libraries required for erasure to the on-chip RAM and erases the user MATs and user boot MATs before the transition.

3. Programming/erasing state

   Programming and erasure by the boot program take place in this state. The boot program is made to transfer the programming/erasing programs to the on-chip RAM by commands from the host. Sum checks and blank checks are executed by sending these commands from the host.

These boot program states are shown in figure 18.23.

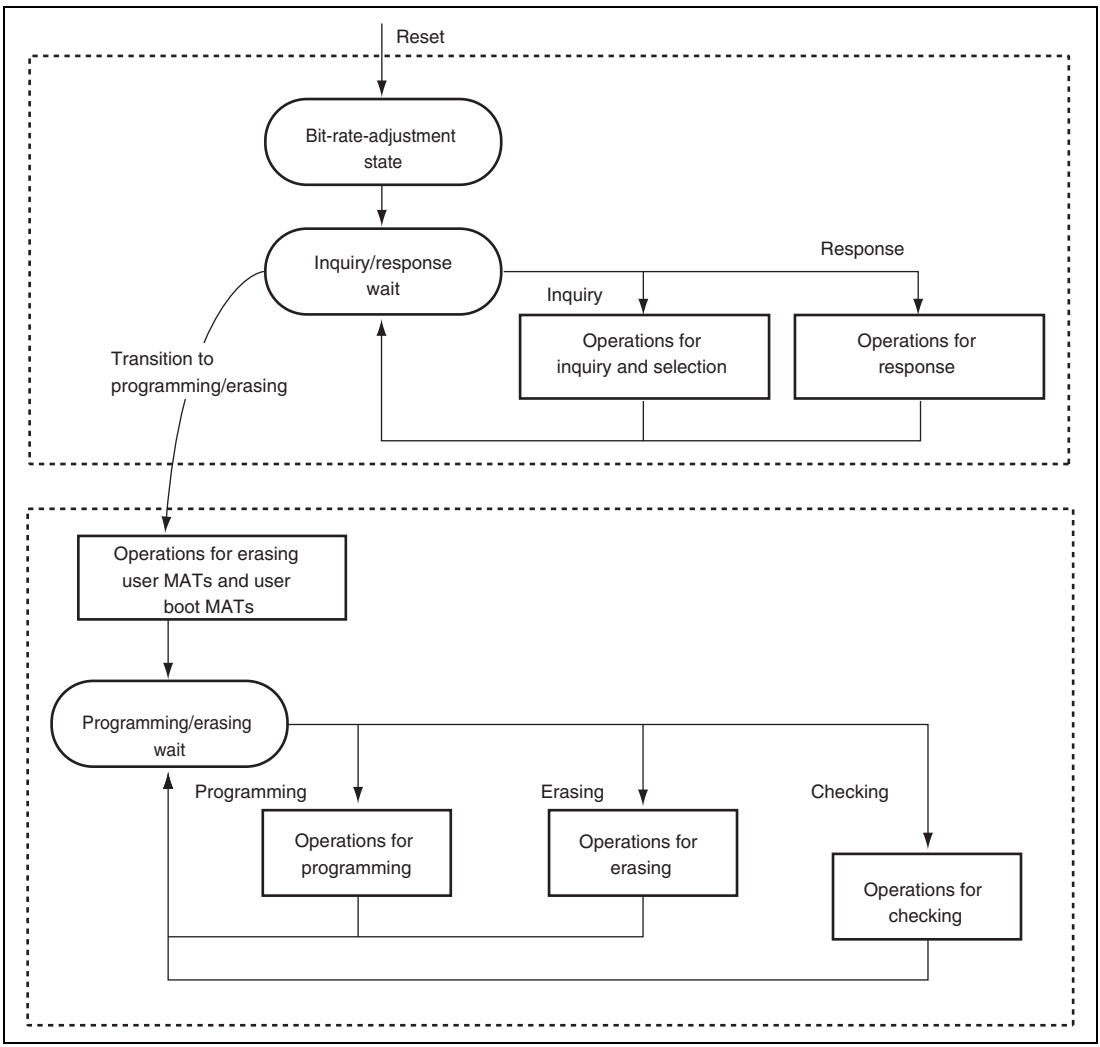


Figure 18.23 Boot Program States

(1) Bit-Rate-Adjustment State

The bit rate is calculated by measuring the period of transfer of a low-level byte (H'00) from the host. The bit rate can be changed by the command for a new bit rate selection. After the bit rate has been adjusted, the boot program enters the inquiry and selection state. The bit-rate-adjustment sequence is shown in figure 18.24.

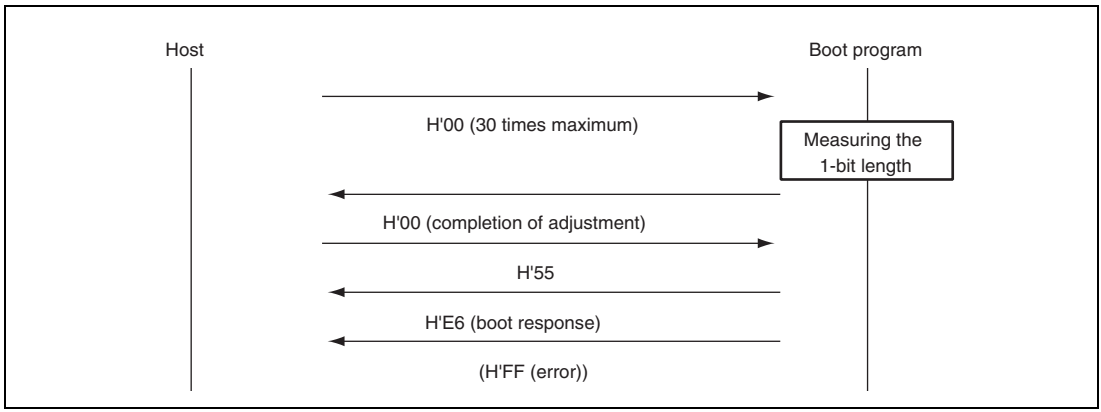


Figure 18.24 Bit-Rate-Adjustment Sequence

(2) Communications Protocol

After adjustment of the bit rate, the protocol for serial communications between the host and the boot program is as shown below.

1. One-byte commands and one-byte responses

   These one-byte commands and one-byte responses consist of the inquiries and the ACK for successful completion.
2. n-byte commands or n-byte responses

   These commands and responses are comprised of n bytes of data. These are selections and responses to inquiries.

   The program data size is not included under this heading because it is determined in another command.
3. Error response

   The error response is a response to inquiries. It consists of an error response and an error code and comes two bytes.
4. Programming of 128 bytes

   The size is not specified in commands. The size of n is indicated in response to the programming unit inquiry.

5. Memory read response

   This response consists of four bytes of data.

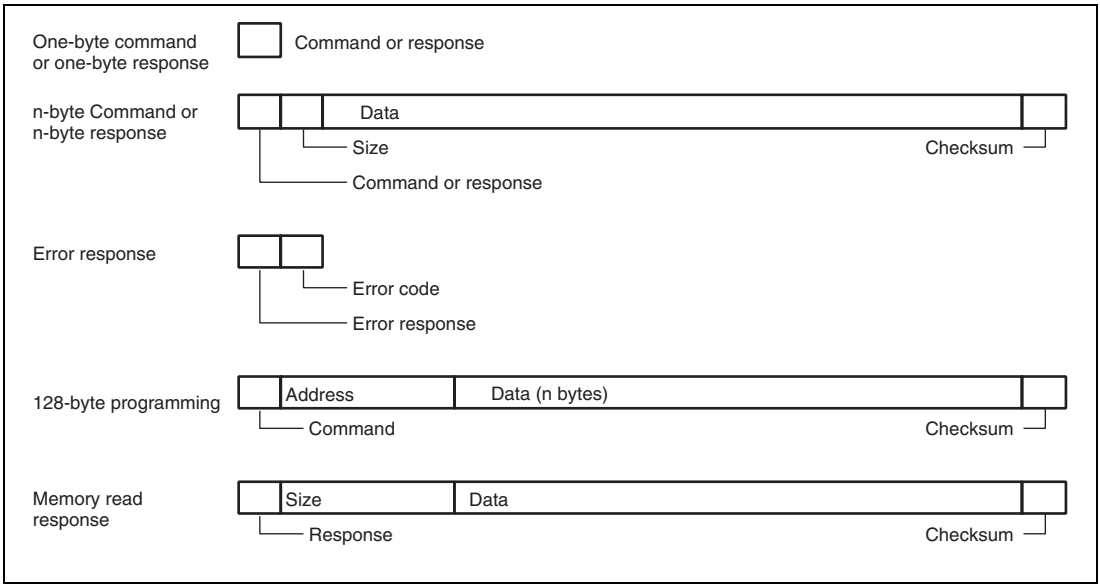


Figure 18.25 Communication Protocol Format

- Command (one byte): Commands including inquiries, selection, programming, erasing, and checking
- Response (one byte): Response to an inquiry
- Size (one byte): The amount of data for transmission excluding the command, amount of data, and checksum
- Checksum (one byte): The checksum is calculated so that the total of all values from the command byte to the SUM byte becomes H'00.
- Data (n bytes): Detailed data of a command or response
- Error response (one byte): Error response to a command
- Error code (one byte): Type of the error
- Address (four bytes): Address for programming
- Data (n bytes): Data to be programmed (the size is indicated in the response to the programming unit inquiry.)
- Size (four bytes): Four-byte response to a memory read

(3) Inquiry and Selection States

The boot program returns information from the flash memory in response to the host's inquiry commands and sets the device code, clock mode, and bit rate in response to the host's selection command.

Table 18.15 lists the inquiry and selection commands.

Table 18.15 Inquiry and Selection Commands

Command	Command Name	Description
H'20	Supported device inquiry	Inquiry regarding device codes
H'10	Device selection	Selection of device code
H'21	Clock mode inquiry	Inquiry regarding numbers of clock modes and values of each mode
H'11	Clock mode selection	Indication of the selected clock mode
H'22	Multiplication ratio inquiry	Inquiry regarding the number of frequency-multiplied clock types, the number of multiplication ratios, and the values of each multiple
H'23	Operating clock frequency inquiry	Inquiry regarding the maximum and minimum values of the main clock and peripheral clocks
H'24	User boot MAT information inquiry	Inquiry regarding the number of user boot MATs and the start and last addresses of each MAT
H'25	User MAT information inquiry	Inquiry regarding the a number of user MATs and the start and last addresses of each MAT
H'26	Block for erasing information Inquiry	Inquiry regarding the number of blocks and the start and last addresses of each block
H'27	Programming unit inquiry	Inquiry regarding the unit of program data
H'3F	New bit rate selection	Selection of new bit rate
H'40	Transition to programming/erasing state	Erasing of user MAT and user boot MAT, and entry to programming/erasing state
H'4F	Boot program status inquiry	Inquiry into the operated status of the boot program

The selection commands, which are device selection (H'10), clock mode selection (H'11), and new bit rate selection (H'3F), should be sent from the host in that order. When two or more selection commands are sent at once, the last command will be valid.

All of these commands, except for the boot program status inquiry command (H'4F), will be valid until the boot program receives the programming/erasing transition (H'40). The host can choose the needed commands and make inquiries while the above commands are being transmitted. H'4F is valid even after the boot program has received H'40.

(a) Supported Device Inquiry

The boot program will return the device codes of supported devices and the product code in response to the supported device inquiry.

Command

H'20

- Command, H'20, (one byte): Inquiry regarding supported devices

Response

H'30	Size	Number of devices	
Number of characters	Device code		Product name
...			
SUM			

- Response, H'30, (one byte): Response to the supported device inquiry
- Size (one byte): Number of bytes to be transmitted, excluding the command, size, and checksum, that is, the amount of data contributes by the number of devices, characters, device codes and product names
- Number of devices (one byte): The number of device types supported by the boot program
- Number of characters (one byte): The number of characters in the device codes and boot program's name
- Device code (four bytes): ASCII code of the supporting product
- Product name (n bytes): Type name of the boot program in ASCII-coded characters
- SUM (one byte): Checksum

  The checksum is calculated so that the total number of all values from the command byte to the SUM byte becomes H'00.

(b) Device Selection

The boot program will set the supported device to the specified device code. The program will return the selected device code in response to the inquiry after this setting has been made.

Command	H'10	Size	Device code	SUM

- Command, H'10, (one byte): Device selection
- Size (one byte): Amount of device-code data
  This is fixed at 4.
- Device code (four bytes): Device code (ASCII code) returned in response to the supported device inquiry
- SUM (one byte): Checksum

Response	H'06

- Response, H'06, (one byte): Response to the device selection command
  ACK will be returned when the device code matches.

Error response	H'90	ERROR

- Error response, H'90, (one byte): Error response to the device selection command
  ERROR : (one byte): Error code
  H'11: Sum check error
  H'21: Device code error, that is, the device code does not match

(c) Clock Mode Inquiry

The boot program will return the supported clock modes in response to the clock mode inquiry.

Command	H'21

- Command, H'21, (one byte): Inquiry regarding clock mode

Response	H'31	Size	Mode	···	SUM

- Response, H'31, (one byte): Response to the clock-mode inquiry
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): Values of the supported clock modes (i.e. H'01 means clock mode 1.)
- SUM (one byte): Checksum

(d) Clock Mode Selection

The boot program will set the specified clock mode. The program will return the selected clock-mode information after this setting has been made.

The clock-mode selection command should be sent after the device-selection commands.

Command	H'11	Size	Mode	SUM

- Command, H'11, (one byte): Selection of clock mode
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): A clock mode returned in reply to the supported clock mode inquiry.
- SUM (one byte): Checksum

Response	H'06

- Response, H'06, (one byte): Response to the clock mode selection command
  ACK will be returned when the clock mode matches.

Error Response	H'91	ERROR

- Error response, H'91, (one byte) : Error response to the clock mode selection command
- ERROR : (one byte): Error code
  H'11: Checksum error
  H'22: Clock mode error, that is, the clock mode does not match.

Even if the clock mode numbers are H'00 and H'01 by a clock mode inquiry, the clock mode must be selected using these respective values.

(e) Multiplication Ratio Inquiry

The boot program will return the supported multiplication and division ratios.

Command

H'22

- Command, H'22, (one byte): Inquiry regarding multiplication ratio

Response

H'32	Size	Number of multiplica-tion types					
Number of multiplication ratios	Multiplica-tion ratio	···					
···							
SUM							

- Response, H'32, (one byte): Response to the multiplication ratio inquiry
- Size (one byte): The amount of data that represents the number of multiplication types and multiplication ratios and the multiplication ratios
- Number of multiplication types (one byte): The number of multiplication types to which the device can be set.
  (e.g. when there are two multiplied clock types, which are the main and peripheral clocks, the number of types will be H'02.)
- Number of multiplication ratios (one byte): The number of multiplication ratios for each type (e.g. the number of multiplication ratios to which the main clock can be set and the peripheral clock can be set.)
- Multiplication ratio (one byte)

  Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H'04.)

  Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

  The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are multiplication types.
- SUM (one byte): Checksum

(f) Operating Clock Frequency Inquiry

The boot program will return the number of operating clock frequencies, and the maximum and minimum values.

Command

H'23

- Command, H'23, (one byte): Inquiry regarding operating clock frequencies

Response

H'33	Size	Number of operating clock frequencies
Minimum value of operating clock frequency		Maximum value of operating clock frequency
...		
SUM		

- Response, H'33, (one byte): Response to operating clock frequency inquiry
- Size (one byte): The number of bytes that represents the minimum values, maximum values, and the number of frequencies.
- Number of operating clock frequencies (one byte): The number of supported operating clock frequency types
  (e.g. when there are two operating clock frequency types, which are the main and peripheral clocks, the number of types will be H'02.)
- Minimum value of operating clock frequency (two bytes): The minimum value of the multiplied or divided clock frequency.

  The minimum and maximum values of the operating clock frequency represent the values in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 17.00 MHz, it will be 2000, which is H'07D0.)
- Maximum value (two bytes): Maximum value among the multiplied or divided clock frequencies.

  There are as many pairs of minimum and maximum values as there are operating clock frequencies.
- SUM (one byte): Checksum

(g) User Boot MAT Information Inquiry

The boot program will return the number of user boot MATs and their addresses.

Command	H'24

- Command, H'24, (one byte): Inquiry regarding user boot MAT information

Response	H'34	Size	Number of areas	
	Area-start address			Area-last address
	...			
	SUM			

- Response, H'34, (one byte): Response to user boot MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start addresses, and area-last address
- Number of Areas (one byte): The number of consecutive user boot MAT areas
  When user boot MAT areas are consecutive, the number of areas returned is H'01.
- Area-start address (four byte): Start address of the area
- Area-last address (four byte): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(h) User MAT Information Inquiry

The boot program will return the number of user MATs and their addresses.

Command	H'25

- Command, H'25, (one byte): Inquiry regarding user MAT information

Response	H'35	Size	Number of areas	
	Start address area			Last address area
	...			
	SUM			

- Response, H'35, (one byte): Response to the user MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start address and area-last address
- Number of areas (one byte): The number of consecutive user MAT areas
  When the user MAT areas are consecutive, the number of areas is H'01.
- Area-start address (four bytes): Start address of the area

- Area-last address (four bytes): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(i) Erased Block Information Inquiry

The boot program will return the number of erased blocks and their addresses.

Command	H'26

- Command, H'26, (two bytes): Inquiry regarding erased block information

Response	H'36	Size	Number of blocks	
	Block start address			Block last address
	...			
	SUM			

- Response, H'36, (one byte): Response to the number of erased blocks and addresses
- Size (three bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
- Number of blocks (one byte): The number of erased blocks
- Block start address (four bytes): Start address of a block
- Block last Address (four bytes): Last address of a block
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(j) Programming Unit Inquiry

The boot program will return the programming unit used to program data.

Command	H'27

- Command, H'27, (one byte): Inquiry regarding programming unit

Response	H'37	Size	Programming unit	SUM

- Response, H'37, (one byte): Response to programming unit inquiry
- Size (one byte): The number of bytes that indicate the programming unit, which is fixed to 2
- Programming unit (two bytes): A unit for programming
  This is the unit for reception of programming.
- SUM (one byte): Checksum

(k) New Bit-Rate Selection

The boot program will set a new bit rate and return the new bit rate.

This selection should be sent after sending the clock mode selection command.

Command

H'3F	Size	Bit rate	Input frequency
Number of multiplication types	Multiplication ratio 1	Multiplication ratio 2	
SUM			

- Command, H'3F, (one byte): Selection of new bit rate
- Size (one byte): The amount of data that represents the bit rate, input frequency, number of multiplication types, and multiplication ratio
- Bit rate (two bytes): New bit rate
  One hundredth of the value (e.g. when the value is 19200 bps, it will be 192, which is H'00C0.)
- Input frequency (two bytes): Frequency of the clock input to the boot program
  This is valid to the hundredths place and represents the value in MHz multiplied by 100. (E.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Number of multiplication types (one byte): The number of multiplication types to which the device can be set.
- Multiplication ratio 1 (one byte): The value of multiplication or division ratios for the main operating frequency
  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)
  Division ratio: The inverse of the division ratio, as a negative number (e.g. when the clock frequency is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- Multiplication ratio 2 (one byte): The value of multiplication or division ratios for the peripheral frequency
  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)
  (Division ratio: The inverse of the division ratio, as a negative number (E.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to selection of a new bit rate
  When it is possible to set the bit rate, the response will be ACK.

Error Response	H'BF	ERROR

- Error response, H'BF, (one byte): Error response to selection of new bit rate
- ERROR: (one byte): Error code

H'11: Sum checking error
H'24: Bit-rate selection error
The rate is not available.
H'25: Error in input frequency
This input frequency is not within the specified range.
H'26: Multiplication-ratio error
The ratio does not match an available ratio.
H'27: Operating frequency error
The frequency is not within the specified range.

(4) Receive Data Check

The methods for checking of receive data are listed below.

1. Input frequency

   The received value of the input frequency is checked to ensure that it is within the range of minimum to maximum frequencies which matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

2. Multiplication ratio

   The received value of the multiplication ratio or division ratio is checked to ensure that it matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

3. Operating frequency error

   Operating frequency is calculated from the received value of the input frequency and the multiplication or division ratio. The input frequency is input to the LSI and the LSI is operated at the operating frequency. The expression is given below.

   Operating frequency = Input frequency × Multiplication ratio, or
   Operating frequency = Input frequency ÷ Division ratio

   The calculated operating frequency should be checked to ensure that it is within the range of minimum to maximum frequencies which are available with the clock modes of the specified device. When it is out of this range, an operating frequency error is generated.

4. Bit rate

   To facilitate error checking, the value (n) of clock select (CKS) in the serial mode register (SMR), and the value (N) in the bit rate register (BRR), which are found from the peripheral operating clock frequency (ϕ) and bit rate (B), are used to calculate the error rate to ensure that it is less than 4%. If the error is more than 4%, a bit rate error is generated. The error is calculated using the following expression:

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

When the new bit rate is selectable, the rate will be set in the register after sending ACK in response. The host will send an ACK with the new bit rate for confirmation and the boot program will response with that rate.

Confirmation	H'06

- Confirmation, H'06, (one byte): Confirmation of a new bit rate

Response	H'06

- Response, H'06, (one byte): Response to confirmation of a new bit rate

The sequence of new bit-rate selection is shown in figure 18.26.

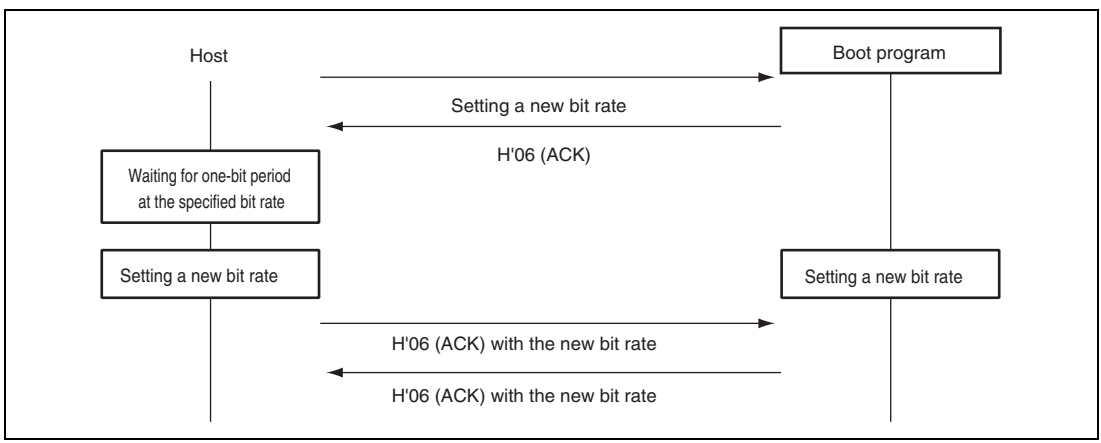


Figure 18.26 New Bit-Rate Selection Sequence

(5) Transition to Programming/Erasing State

The boot program will transfer the erasing program, and erase the user MATs and user boot MATs in that order. On completion of this erasure, ACK will be returned and will enter the programming/erasing state.

The host should select the device code, clock mode, and new bit rate with device selection, clock-mode selection, and new bit-rate selection commands, and then send the command for the transition to programming/erasing state. These procedures should be carried out before sending of the programming selection command or program data.

Command	H'40

- Command, H'40, (one byte): Transition to programming/erasing state

Response	H'06

- Response, H'06, (one byte): Response to transition to programming/erasing state
  The boot program will send ACK when the user MAT and user boot MAT have been erased by the transferred erasing program.

Error Response	H'C0	H'51

- Error response, H'C0, (one byte): Error response for user boot MAT blank check
- Error code, H'51, (one byte): Erasing error
  An error occurred and erasure was not completed.

(6) Command Error

A command error will occur when a command is undefined, the order of commands is incorrect, or a command is unacceptable. Issuing a clock-mode selection command before a device selection or an inquiry command after the transition to programming/erasing state command, are examples.

Error Response	H'80	H'xx

- Error response, H'80, (one byte): Command error
- Command, H'xx, (one byte): Received command

(7) Command Order

The order for commands in the inquiry selection state is shown below.

1. A supported device inquiry (H'20) should be made to inquire about the supported devices.
2. The device should be selected from among those described by the returned information and set with a device-selection (H'10) command.
3. A clock-mode inquiry (H'21) should be made to inquire about the supported clock modes.
4. The clock mode should be selected from among those described by the returned information and set.
5. After selection of the device and clock mode, inquiries for other required information should be made, such as the multiplication-ratio inquiry (H'22) or operating frequency inquiry (H'23), which are needed for a new bit-rate selection.
6. A new bit rate should be selected with the new bit-rate selection (H'3F) command, according to the returned information on multiplication ratios and operating frequencies.
7. After selection of the device and clock mode, the information of the user boot MAT and user MAT should be made to inquire about the user boot MATs information inquiry (H'24), user MATs information inquiry (H'25), erased block information inquiry (H'26), and programming unit inquiry (H'27).
8. After making inquiries and selecting a new bit rate, issue the transition to programming/erasing state command (H'40). The boot program will then enter the programming/erasing state.

(8) Programming/Erasing State

A programming selection command makes the boot program select the programming method, a 128-byte programming command makes it program the memory with data, and an erasing selection command and block erasing command make it erase the block. Table 18.16 lists the programming/erasing commands.

Table 18.16 Programming/Erasing Commands

Command	Command Name	Description
H'42	User boot MAT programming selection	Transfers the user boot MAT programming program
H'43	User MAT programming selection	Transfers the user MAT programming program
H'50	128-byte programming	Programs 128 bytes of data
H'48	Erasing selection	Transfers the erasing program
H'58	Block erasing	Erases a block of data
H'52	Memory read	Reads the contents of memory
H'4A	User boot MAT sum check	Checks the checksum of the user boot MAT
H'4B	User MAT sum check	Checks the checksum of the user MAT
H'4C	User boot MAT blank check	Checks the blank data of the user boot MAT
H'4D	User MAT blank check	Checks the blank data of the user MAT
H'4C	User boot MAT blank check	Checks whether the contents of the user boot MAT are blank
H'4D	User MAT blank check	Checks whether the contents of the user MAT are blank
H'4F	Boot program status inquiry	Inquires into the boot program's status

- Programming

  Programming is executed by the programming selection and 128-byte programming commands.

  Firstly, the host should send the programming selection command and select the programming method and programming MATs. There are two programming selection commands, and selection is according to the area and method for programming.

  1. User boot MAT programming selection
  2. User MAT programming selection

  After issuing the programming selection command, the host should send the 128-byte programming command. The 128-byte programming command that follows the selection command represents the data programmed according to the method specified by the selection command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H'FFFFFFFF as the address will stop the programming. On completion of programming, the boot program will wait for selection of programming or erasing.

  Where the sequence of programming operations that is executed includes programming with another method or of another MAT, the procedure must be repeated from the programming selection command.

  The sequence for the programming selection and 128-byte programming commands is shown in figure 18.27.

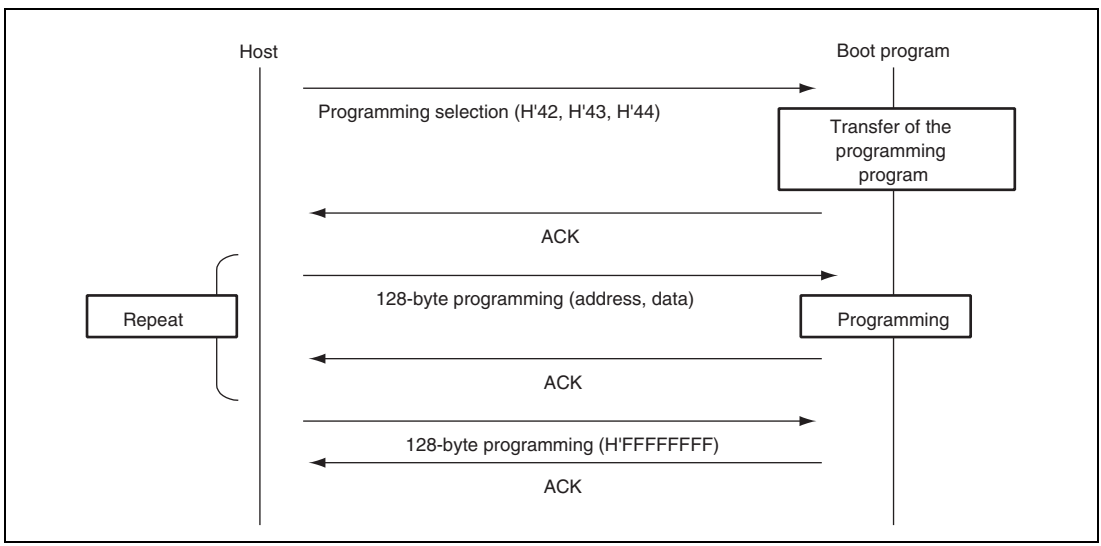


Figure 18.27 Programming Sequence

- Erasure

  Erasure is executed by the erasure selection and block erasure commands.

  Firstly, erasure is selected by the erasure selection command and the boot program then erases the specified block. The command should be repeatedly executed if two or more blocks are to be erased. Sending a block erasure command from the host with the block number H'FF will stop the erasure operating. On completion of erasing, the boot program will wait for selection of programming or erasing.

  The sequence for the erasure selection and block erasure commands is shown in figure 18.28.

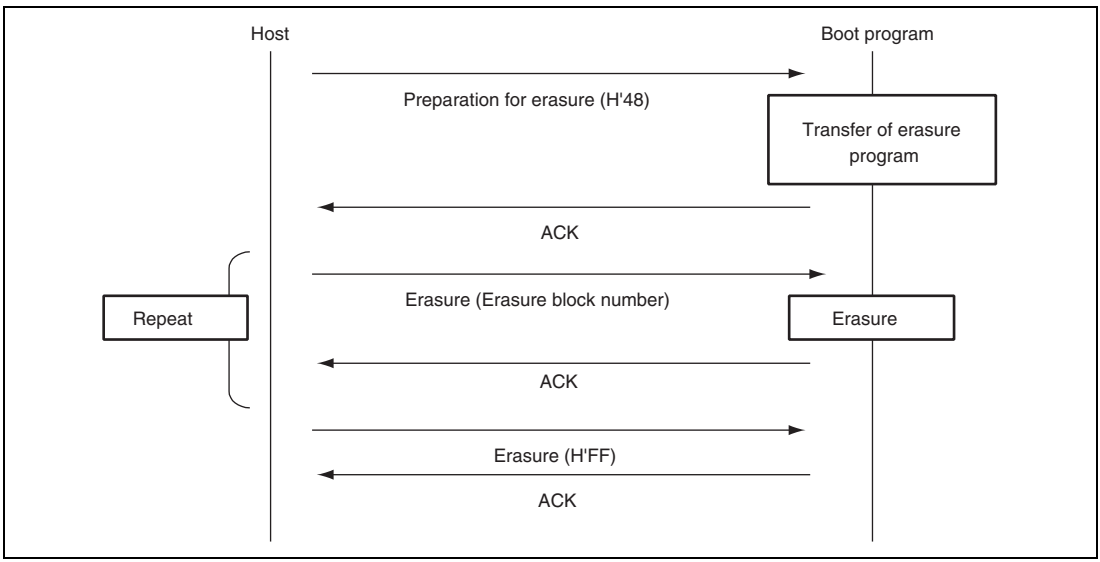


Figure 18.28 Erasure Sequence

(a) User Boot MAT Programming Selection

The boot program will transfer a programming program. The data is programmed to the user boot MATs by the transferred programming program.

Command	H'42

- Command, H'42, (one byte): User boot-program programming selection

Response	H'06

- Response, H'06, (one byte): Response to user boot-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response	H'C2	ERROR

- Error response : H'C2 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code
  H'54: Selection processing error (transfer error occurs and processing is not completed)

(b) User MAT Programming Selection

The boot program will transfer a program for user MAT programming selection. The data is programmed to the user MATs by the transferred program for programming.

Command	H'43

- Command, H'43, (one byte): User-program programming selection

Response	H'06

- Response, H'06, (one byte): Response to user-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response	H'C3	ERROR

- Error response : H'C3 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code
  H'54: Selection processing error (transfer error occurs and processing is not completed)

(c) 128-Byte Programming

The boot program will use the programming program transferred by the programming selection to program the user boot MATs or user MATs in response to 128-byte programming.

Command

H'50	Address						
Data	···						
···							
SUM							

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): Start address for programming
  Multiple of the size specified in response to the programming unit inquiry
  (i.e. H'00, H'01, H'00, H'00 : H'01000000)
- Program data (128 bytes): Data to be programmed
  The size is specified in the response to the programming unit inquiry.
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response

H'D0	ERROR

- Error response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code
  - H'11: Checksum Error
  - H'2A: Address error
    The address is not in the specified MAT.
  - H'53: Programming error
    A programming error has occurred and programming cannot be continued.

The specified address should match the unit for programming of data. For example, when the programming is in 128-byte units, the lower eight bits of the address should be H'00 or H'80. When there are less than 128 bytes of data to be programmed, the host should fill the rest with H'FF.

Sending the 128-byte programming command with the address of H'FFFFFFFF will stop the programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.

Command	H'50	Address	SUM

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): End code is H'FF, H'FF, H'FF, H'FF.
- SUM (one byte): Checksum

Response	H'06

- Response, H'06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response	H'D0	ERROR

- Error Response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

  H'11: Checksum error

  H'53: Programming error

  An error has occurred in programming and programming cannot be continued.

(d) Erasure Selection

The boot program will transfer the erasure program. User MAT data is erased by the transferred erasure program.

Command	H'48

- Command, H'48, (one byte): Erasure selection

Response	H'06

- Response, H'06, (one byte): Response for erasure selection
  After the erasure program has been transferred, the boot program will return ACK.

Error Response	H'C8	ERROR

- Error Response, H'C8, (one byte): Error response to erasure selection
- ERROR: (one byte): Error code

  H'54: Selection processing error (transfer error occurs and processing is not completed)

(e) Block Erasure

The boot program will erase the contents of the specified block.

Command	H'58	Size	Block number	SUM

- Command, H'58, (one byte): Erasure
- Size (one byte): The number of bytes that represents the erase block number
  This is fixed to 1.
- Block number (one byte): Number of the block to be erased
- SUM (one byte): Checksum

Response	H'06

- Response, H'06, (one byte): Response to Erasure
  After erasure has been completed, the boot program will return ACK.

Error Response	H'D8	ERROR

- Error Response, H'D8, (one byte): Response to Erasure
- ERROR (one byte): Error code

  H'11: Sum check error
  H'29: Block number error
  Block number is incorrect.
  H'51: Erasure error
  An error has occurred during erasure.

On receiving block number H'FF, the boot program will stop erasure and wait for a selection command.

Command	H'58	Size	Block number	SUM

- Command, H'58, (one byte): Erasure
- Size, (one byte): The number of bytes that represents the block number
  This is fixed to 1.
- Block number (one byte): H'FF
  Stop code for erasure
- SUM (one byte): Checksum

Response	H'06

- Response, H'06, (one byte): Response to end of erasure (ACK)
  When erasure is to be performed after the block number H'FF has been sent, the procedure should be executed from the erasure selection command.

(f) Memory Read

The boot program will return the data in the specified address.

Command	H'52	Size	Area	Read address	
	Read size				SUM

- Command: H'52 (1 byte): Memory read
- Size (1 byte): Amount of data that represents the area, read address, and read size (fixed at 9)
- Area (1 byte)

  H'00: User boot MAT

  H'01: User MAT

  An address error occurs when the area setting is incorrect.
- Read address (4 bytes): Start address to be read from
- Read size (4 bytes): Size of data to be read
- SUM (1 byte): Checksum

Response	H'52	Read size						
	Data	···						
	SUM							

- Response: H'52 (1 byte): Response to memory read
- Read size (4 bytes): Size of data to be read
- Data (n bytes): Data for the read size from the read address
- SUM (1 byte): Checksum

Error Response	H'D2	ERROR

- Error response: H'D2 (1 byte): Error response to memory read
- ERROR: (1 byte): Error code

  H'11: Sum check error

  H'2A: Address error

  The read address is not in the MAT.

  H'2B: Size error

  The read size exceeds the MAT.

(g) User-Boot Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user-boot program, as a four-byte value.

Command	H'4A

- Command, H'4A, (one byte): Sum check for user-boot program

Response	H'5A	Size	Checksum of user boot program	SUM

- Response, H'5A, (one byte): Response to the sum check of user-boot program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user boot MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(h) User-Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user program.

Command	H'4B

- Command, H'4B, (one byte): Sum check for user program

Response	H'5B	Size	Checksum of user program	SUM

- Response, H'5B, (one byte): Response to the sum check of the user program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(i) User Boot MAT Blank Check

The boot program will check whether or not all user boot MATs are blank and return the result.

Command	H'4C

- Command, H'4C, (one byte): Blank check for user boot MAT

Response	H'06

- Response, H'06, (one byte): Response to the blank check of user boot MAT
  If all user MATs are blank (H'FF), the boot program will return ACK.

Error Response	H'CC	H'52

- Error Response, H'CC, (one byte): Response to blank check for user boot MAT
- Error Code, H'52, (one byte): Erasure has not been completed.

(j) User MAT Blank Check

The boot program will check whether or not all user MATs are blank and return the result.

Command	H'4D

- Command, H'4D, (one byte): Blank check for user MATs

Response	H'06

- Response, H'06, (one byte): Response to the blank check for user MATs
  If the contents of all user MATs are blank (H'FF), the boot program will return ACK.

Error Response	H'CD	H'52

- Error Response, H'CD, (one byte): Error response to the blank check of user MATs.
- Error code, H'52, (one byte): Erasure has not been completed.

(k) Boot Program State Inquiry

The boot program will return indications of its present state and error condition. This inquiry can be made in the inquiry/selection state or the programming/erasing state.

Command	H'4F

- Command, H'4F, (one byte): Inquiry regarding boot program's state

Response	H'5F	Size	Status	ERROR	SUM

- Response, H'5F, (one byte): Response to boot program state inquiry
- Size (one byte): The number of bytes. This is fixed to 2.
- Status (one byte): State of the boot program
- ERROR (one byte): Error status

  ERROR = 0 indicates normal operation.
  ERROR = 1 indicates error has occurred.

- SUM (one byte): Sum check

Table 18.17 Status Code

Code	Description
H'11	Device selection wait
H'12	Clock mode selection wait
H'13	Bit rate selection wait
H'1F	Programming/erasing state transition wait (bit rate selection is completed)
H'31	Programming state for erasure
H'3F	Programming/erasing selection wait (erasure is completed)
H'4F	Program data receive wait
H'5F	Erase block specification wait (erasure is completed)

Table 18.18 Error Code

Code	Description
H'00	No error
H'11	Sum check error
H'12	Program size error
H'21	Device code mismatch error
H'22	Clock mode mismatch error
H'24	Bit rate selection error
H'25	Input frequency error
H'26	Multiplication ratio error
H'27	Operating frequency error
H'29	Block number error
H'2A	Address error
H'2B	Data length error
H'51	Erasure error
H'52	Erasure incomplete error
H'53	Programming error
H'54	Selection processing error
H'80	Command error
H'FF	Bit-rate-adjustment confirmation error

18.14 Usage Notes

1. The initial state of the product at its shipment is in the erased state. For the product whose revision of erasing is undefined, we recommend to execute automatic erasure for checking the initial state (erased state) and compensating.
2. For the PROM programmer suitable for programmer mode in this LSI and its program version, refer to the instruction manual of the socket adapter.
3. If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.
4. Use a PROM programmer that supports the device with 1-Mbyte on-chip flash memory and 3.3-V programming voltage. Use only the specified socket adapter.
5. Do not remove the chip from the PROM programmer nor input a reset signal during programming/erasing in which a high voltage is applied to the flash memory. Doing so may damage the flash memory permanently. If a reset is input accidentally, the reset must be released after the reset input period of at least 100μs.
6. The flash memory is not accessible until FKEY is cleared after programming/erasing starts. If the operating mode is changed and this LSI is restarted by a reset immediately after programming/erasing has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100μs. Transition to the reset state during programming/erasing is inhibited. If a reset is input accidentally, the reset must be released after the reset input period of at least 100μs.
7. At powering on or off the Vcc power supply, fix the $\overline{RES}$ pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.
8. In on-board programming mode or programmer mode, programming of the 128-byte programming-unit block must be performed only once. Perform programming in the state where the programming-unit block is fully erased.
9. When the chip is to be reprogrammed with the programmer after execution of programming or erasure in on-board programming mode, it is recommended that automatic programming is performed after execution of automatic erasure.
10. To program the flash memory, the program data and program must be allocated to addresses which are higher than those of the external interrupt vector table and H'FF must be written to all the system reserved areas in the exception handling vector table.
11. The programming program that includes the initialization routine and the erasing program that includes the initialization routine are each 4 Kbytes or less. Accordingly, when the CPU clock frequency is 35 MHz, the download for each program takes approximately 60 μs at the maximum.

12. A programming/erasing program for the flash memory used in a conventional F-ZTAT H8, H8S microcomputer which does not support download of the on-chip program by setting the SCO bit in FCCS to 1 cannot run in this LSI. Be sure to download the on-chip program to execute programming/erasing of the flash memory in this F-ZTAT H8SX microcomputer.
13. Unlike a conventional F-ZTAT H8 or H8S microcomputers, measures against a program crash are not taken by WDT while programming/erasing and downloading a programming/erasing program. When needed, measures should be taken by user. A periodic interrupt generated by the WDT can be used as the measures, as an example. In this case, the interrupt generation period should take into consideration time to program/erase the flash memory.
14. When downloading the programming/erasing program, do not clear the SCO bit in FCCS to 0 after immediately setting it to 1. Otherwise, download cannot be performed normally. Immediately after executing the instruction to set the SCO bit to 1, dummy read of the FCCS must be executed twice.
15. The contents of general registers ER0 and ER1 are not saved during download of an on-chip program, initialization, programming, or erasure. When needed, save the general registers before a download request or before execution of initialization, programming, or erasure using the procedure program.

Section 19 Clock Pulse Generator

This LSI has an on-chip clock pulse generator (CPG) that generates the system clock (Iϕ), peripheral module clock (Pϕ), and external bus clock (Bϕ).

The clock pulse generator consists of an oscillator, a PLL (Phase Locked Loop) circuit, a divider, and selecters. Figure 19.1 shows a block diagram of the clock pulse generat2or.

Clock frequencies can be changed by the PLL circuit and divider in the CPG. Changing the system clock control register (SCKCR) setting by software can change the clock frequencies.

This LSI supports three types of clocks: a system clock provided to the CPU and bus masters, a peripheral module clock provided to the peripheral modules, and an external bus clock provided to the external bus. These clocks can be specified independently. Note, however, that the frequencies of the peripheral clock and external bus clock are lower than that of the system clock.

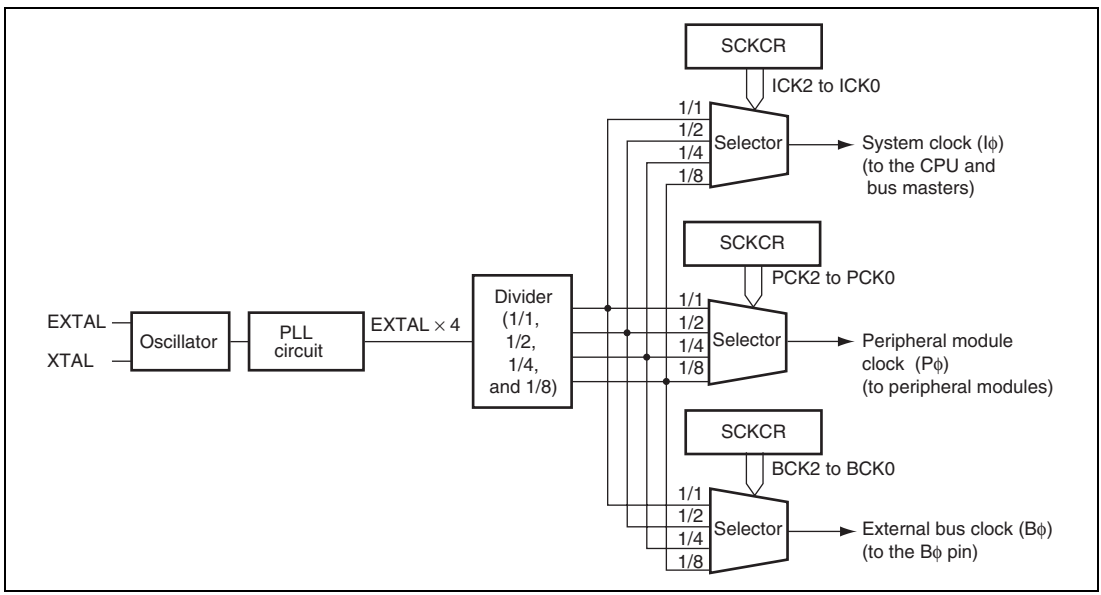


Figure 19.1 Block Diagram of Clock Pulse Generator

19.1 Register Description

The clock pulse generator has the following register.

- System clock control register (SCKCR)

19.1.1 System Clock Control Register (SCKCR)

SCKCR controls Bϕ clock output and frequencies of the system, peripheral module, and external bus clocks.

Bit	15	14	13	12	11	10	9	8
Bit Name	PSTOP1	—	—	—	—	ICK2	ICK1	ICK0
Initial Value	0	0	0	0	0	0	1	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
Bit Name	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0
Initial Value	0	0	1	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	PSTOP1	0	R/W	B ϕ Clock Output Enable Controls Bϕ output on PA7. Normal operation 0: Bϕ output 1: Fixed high
14 to 11	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
10	ICK2	0	R/W	System Clock (Iϕ) Select
9	ICK1	1	R/W	These bits select the frequency of the system clock provided to the CPU, DMAC, and DTC. The ratio to the input clock is as follows:
8	ICK0	0	R/W	000: × 4 001: × 2 010: × 1 011: × 1/2 1XX: Setting prohibited The frequencies of the peripheral module clock and external bus clock change to the same frequency as the system clock if the frequency of the system clock is lower than that of the two clocks.
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6	PCK2	0	R/W	Peripheral Module Clock (Pϕ) Select
5	PCK1	1	R/W	These bits select the frequency of the peripheral module clock. The ratio to the input clock is as follows:
4	PCK0	0	R/W	000: × 4 001: × 2 010: × 1 011: × 1/2 1XX: Setting prohibited The frequency of the peripheral module clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the peripheral module clock higher than that of the system clock, the clocks will have the same frequency in reality.
3	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
2	BCK2	0	R/W	External Bus Clock (Bϕ) Select
1	BCK1	1	R/W	These bits select the frequency of the external bus clock. The ratio to the input clock is as follows:
0	BCK0	0	R/W	000: × 4 001: × 2 010: × 1 011: × 1/2 1XX: Setting prohibited The frequency of the external bus clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the external bus clock higher than that of the system clock, the clocks will have the same frequency in reality.

Note: X: Don't care

19.2 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

19.2.1 Connecting Crystal Resonator

A crystal resonator can be connected as shown in the example in figure 19.2. Select the damping resistance R_d according to table 19.1. An AT-cut parallel-resonance type should be used.

When the clock is provided by connecting a crystal resonator, a crystal resonator having a frequency of 8 to 18 MHz should be connected.

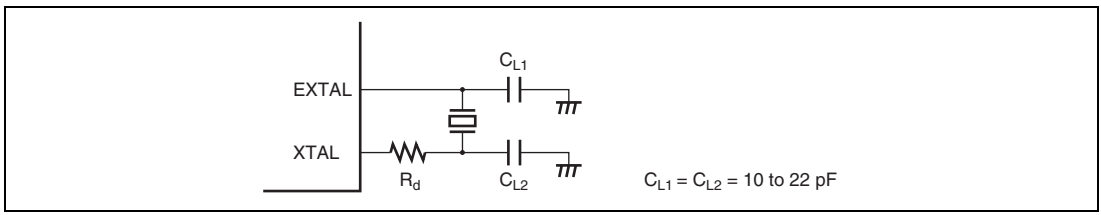


Figure 19.2 Connection of Crystal Resonator (Example)

Table 19.1 Damping Resistance Value

Frequency (MHz)	8	12	18
R_d (Ω)	200	0	0

Figure 19.3 shows an equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 19.2.

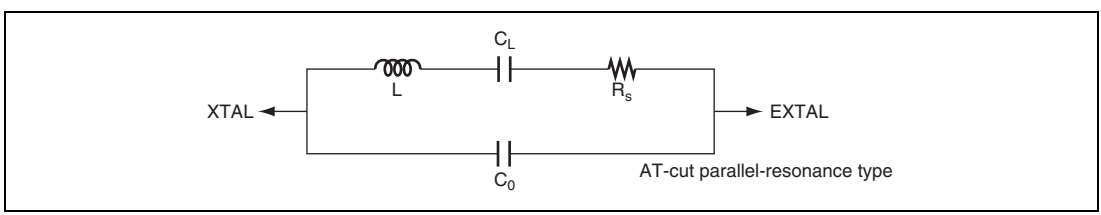


Figure 19.3 Crystal Resonator Equivalent Circuit

Table 19.2 Crystal Resonator Characteristics

Frequency (MHz)	8	12	18
R_S Max. (Ω)	80	60	40
C_0 Max. (pF)		7	

19.2.2 External Clock Input

An external clock signal can be input as shown in the examples in figure 19.4. If the XTAL pin is left open, make sure that parasitic capacitance is no more than 10 pF. When the counter clock is input to the XTAL pin, make sure that the external clock is held high in standby mode.

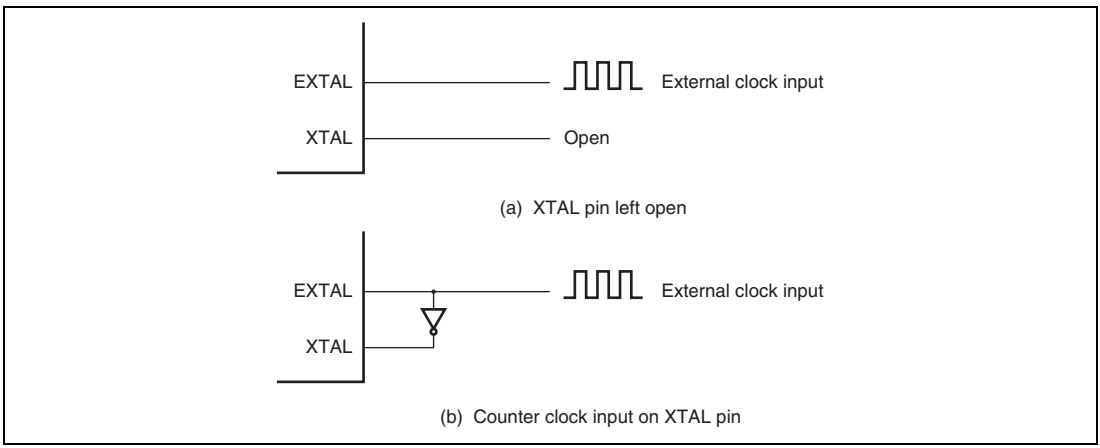


Figure 19.4 External Clock Input (Examples)

For the input conditions of the external clock, refer to table 22.4 in section 22.3.1, Clock Timing. The input external clock should be from 8 to 18 MHz.

19.3 PLL Circuit

The PLL circuit has the function of multiplying the frequency of the clock from the oscillator by a factor of 4. The frequency multiplication factor is fixed. The phase difference is controlled so that the timing of the rising edge of the internal clock is the same as that of the EXTAL pin signal.

19.4 Frequency Divider

The frequency divider divides the PLL clock to generate a 1/2, 1/4, or 1/8 clock. After bits ICK2 to ICK0, PCK 2 to PCK0, and BCK2 to BCK0 are modified, this LSI operates at the modified frequency.

19.5 Usage Notes

19.5.1 Notes on Clock Pulse Generator

1. The following points should be noted since the frequency of ϕ (Iϕ: system clock, Pϕ: peripheral module clock, Bϕ: external bus clock) supplied to each module changes according to the setting of SCKCR.

   Select a clock division ratio that is within the operation guaranteed range of clock cycle time t_{cyc} shown in the AC timing of electrical characteristics.

   For example, the following settings are not permitted under the conditions of 8 MHz ≤ Iϕ ≤ 35 MHz, 8 MHz ≤ Pϕ ≤ 35 MHz, and 8 MHz ≤ Bϕ ≤ 35 MHz: Iϕ < 8 MHz, 35 MHz < Iϕ, Pϕ < 8 MHz, 35 MHz < Pϕ, Bϕ < 8 MHz, and 35 MHz < Bϕ.
2. All the on-chip peripheral modules (except for the DMAC and DTC) operate on the Pϕ. Therefore, note that the time processing of modules such as a timer and SCI differs before and after changing the clock division ratio.

   In addition, wait time for clearing software standby mode differs by changing the clock division ratio. For details, see section 20.7.3, Setting Oscillation Settling Time after Clearing Software Standby Mode.
3. The relationship among the system clock, peripheral module clock, and external bus clock is Iϕ ≥ Pϕ and Iϕ ≥ Bϕ. In addition, the system clock setting has the highest priority. Accordingly, Pϕ or Bϕ may have the frequency set by bits ICK2 to ICK0 regardless of the settings of bits PCK2 to PCK0 or BCK2 to BCK0.
4. Figure 19.5 shows the clock modification timing. After a value is written to SCKCR, this LSI waits for the current bus cycle to complete. After the current bus cycle completes, each clock frequency will be modified within one cycle (worst case) of the external input clock.

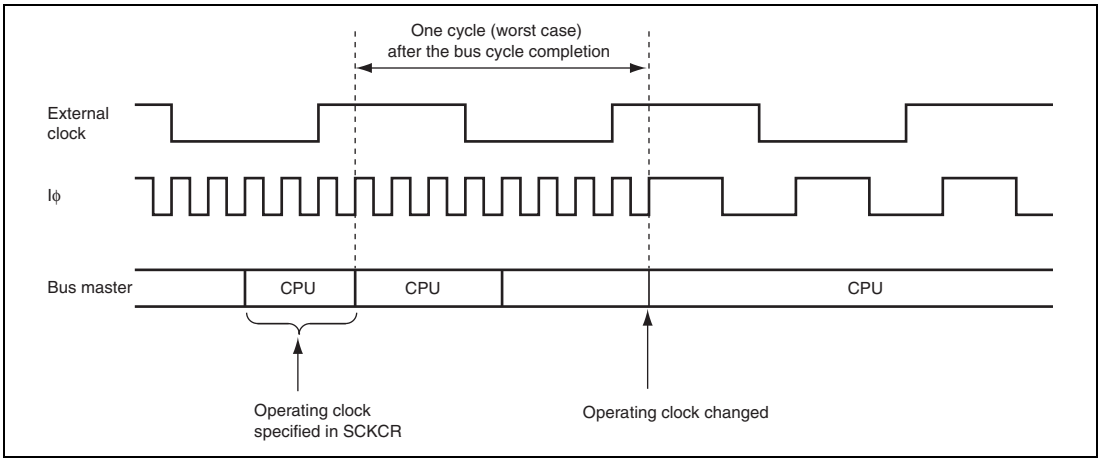


Figure 19.5 Clock Modification Timing

19.5.2 Notes on Resonator

Since various characteristics related to the resonator are closely linked to the user's board design, thorough evaluation is necessary on the user's part, using the resonator connection examples shown in this section as a reference. As the parameters for the resonator will depend on the floating capacitance of the resonator and the mounting circuit, the parameters should be determined in consultation with the resonator manufacturer. The design must ensure that a voltage exceeding the maximum rating is not applied to the resonator pin.

19.5.3 Notes on Board Design

When using the crystal resonator, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins. Other signal lines should be routed away from the oscillation circuit as shown in figure 19.6 to prevent induction from interfering with correct oscillation.

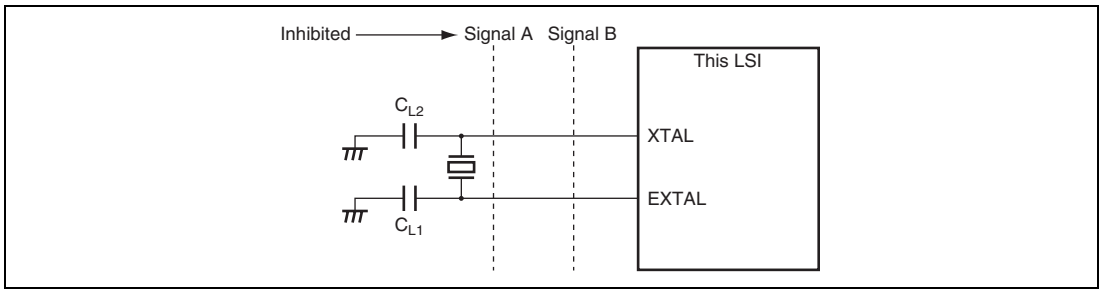


Figure 19.6 Note on Board Design for Oscillation Circuit

Figure 19.7 shows the external circuitry recommended for the PLL circuit. Separate PLLVcc and PLLVss from the other Vcc and Vss lines at the board power supply source, and be sure to insert bypass capacitors CPB and CB close to the pins.

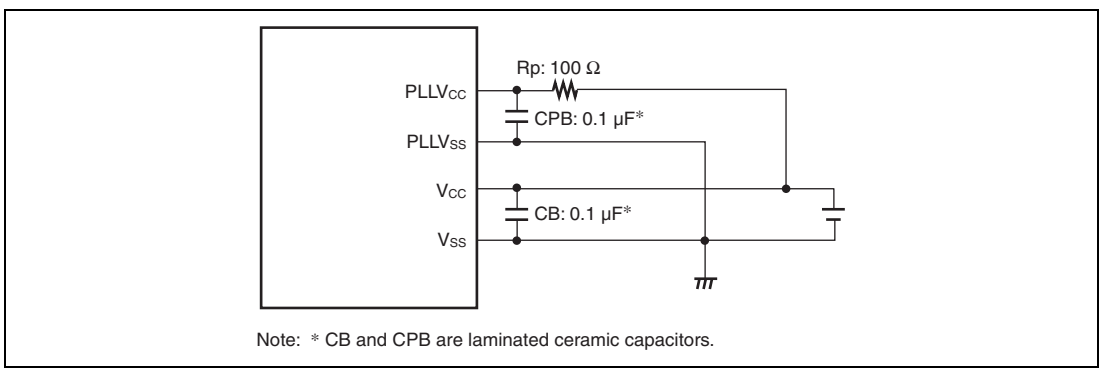


Figure 19.7 Recommended External Circuitry for PLL Circuit

Section 20 Power-Down Modes

This LSI has power consumption reduction functions, such as multi-clock function, module stop function, and transition function to power-down mode.

20.1 Features

- Multi-clock function
  The frequency division ratio is settable independently for the system clock, peripheral module clock, and external bus clock.
- Module stop function
  The functions for each peripheral modules can be stopped to make a transition to a power-down state.
- Transition function to power-down mode
  Transition to a power-down mode is possible to stop the CPU, peripheral modules, and oscillator.
- Four power-down modes
  Sleep mode
  All-module-clock-stop mode
  Software standby mode
  Hardware standby mode

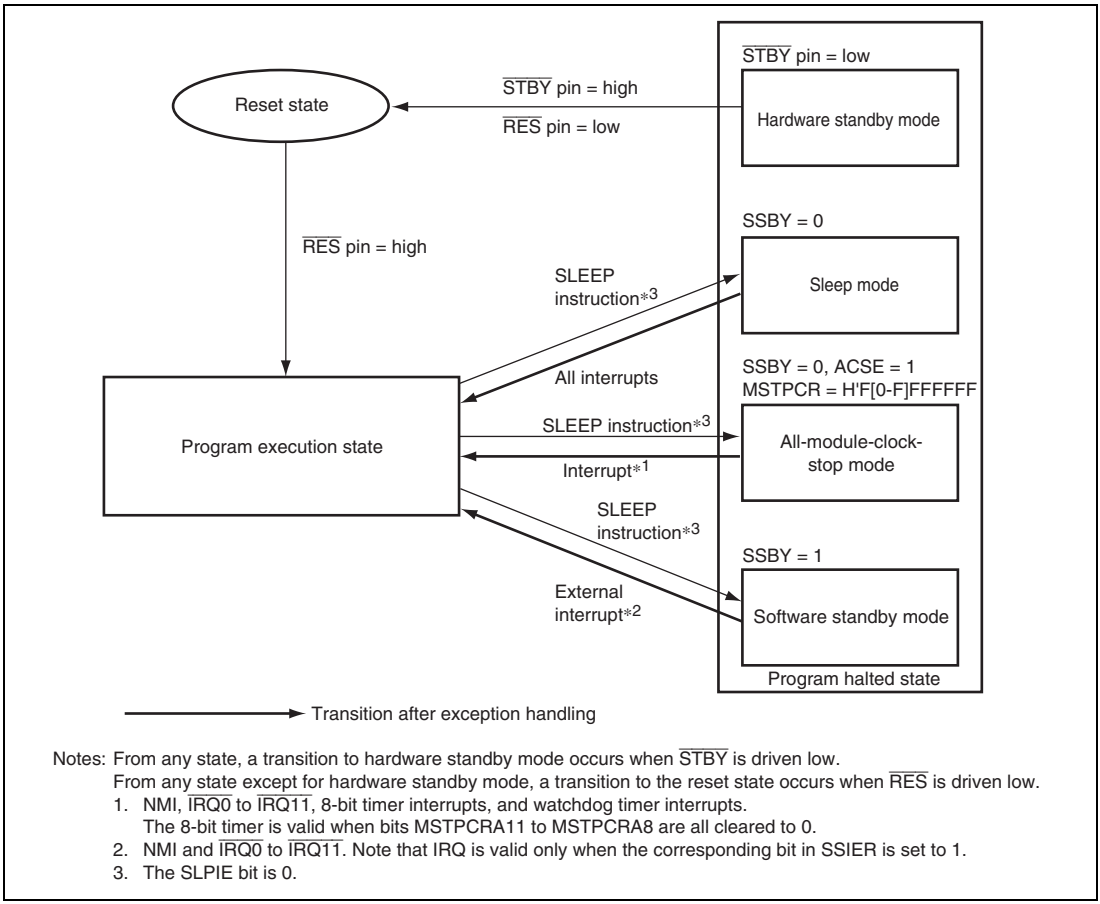
Table 20.1 shows conditions for making a transition to a power-down mode, states of the CPU and peripheral modules, and clearing method for each mode. After the reset state, since this LSI operates in normal program execution state, the modules, other than the DMAC and DTC are stopped.

Table 20.1 Operating States

Operating State	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode	Hardware Standby Mode
Transition condition	Control register + instruction	Control register + instruction	Control register + instruction	Pin input
Cancellation method	Interrupt	Interrupt*2	External interrupt	
Oscillator	Functions	Functions	Halted	Halted
CPU	Halted (retained)	Halted (retained)	Halted (retained)	Halted
Watchdog timer	Functions	Functions	Halted (retained)	Halted
8-bit timer	Functions	Functions*4	Halted (retained)	Halted
Other peripheral modules	Functions	Halted*1	Halted*1	Halted*3
I/O port	Functions	Retained	Retained	Hi-Z

Notes: "Halted (retained)" in the table means that the internal register values are retained and internal operations are suspended.

1. SCI enters the reset state, and other peripheral modules retain their states.
2. External interrupt and some internal interrupts (8-bit timer and watchdog timer)
3. All peripheral modules enter the reset state.
4. "Functions" or "Halted" is selectable through the setting of bits MSTPA11 to MSTPA8 in MSTPCRA. However, pin output is disabled even when "Functions" is selected.



Notes: From any state, a transition to hardware standby mode occurs when STBY is driven low.
From any state except for hardware standby mode, a transition to the reset state occurs when RES is driven low.
1. NMI, IRQ0 to IRQ11, 8-bit timer interrupts, and watchdog timer interrupts.
The 8-bit timer is valid when bits MSTPCRA11 to MSTPCRA8 are all cleared to 0.
2. NMI and IRQ0 to IRQ11. Note that IRQ is valid only when the corresponding bit in SSIER is set to 1.
3. The SLPIE bit is 0.

Figure 20.1 Mode Transitions

20.2 Register Descriptions

The registers related to the power-down modes are shown below. For details on the system clock control register (SCKCR), refer to section 19.1.1, System Clock Control Register (SCKCR).

- Standby control register (SBYCR)
- Module stop control register A (MSTPCRA)
- Module stop control register B (MSTPCRB)
- Module stop control register C (MSTPCRC)

20.2.1 Standby Control Register (SBYCR)

SBYCR controls software standby mode.

Bit	15	14	13	12	11	10	9	8
Bit Name	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0
Initial Value	0	1	0	0	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
Bit Name	SLPIE	—	—	—	—	—	—	—
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	Bit Name	Initial Value	R/W	Description
15	SSBY	0	R/W	Software Standby Specifies the transition mode after executing the SLEEP instruction 0: Shifts to sleep mode after the SLEEP instruction is executed 1: Shifts to software standby mode after the SLEEP instruction is executed This bit remains set to 1 when software standby mode is cleared by using external interrupts and a transition to the normal operation is made. For clearing, write 0 to this bit. When the WDT is used as the watchdog timer, the setting of this bit is disabled. In this case, a transition is always made to sleep mode or all-module-clock-stop mode after the SLEEP instruction is executed. This bit should be cleared to 0 when setting the SLPIE bit to 1.
14	OPE	1	R/W	Output Port Enable Specifies whether the output of the address bus and bus control signals ($\overline{CS0}$ to $\overline{CS7}$, $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$) is retained or set to the high-impedance state in software standby mode. 0: In software standby mode, address bus and bus control signals are high-impedance 1: In software standby mode, address bus and bus control signals retain output state

Bit	Bit Name	Initial Value	R/W	Description
13	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
12	STS4	0	R/W	Standby Timer Select 4 to 0 These bits select the time the MCU waits for the clock to settle when software standby mode is cleared by an external interrupt. With a crystal resonator, refer to table 18.2 and make a selection according to the operating frequency so that the standby time is at least equal to the oscillation settling time. With an external clock, a PLL circuit settling time is necessary. Refer to table 18.2 to set the standby time. While oscillation is being settled, the timer is counted on the Pϕ clock frequency. Careful consideration is required in multi-clock mode. 00000: Reserved 00001: Reserved 00010: Reserved 00011: Reserved 00100: Reserved 00101: Standby time = 64 states 00110: Standby time = 512 states 00111: Standby time = 1024 states 01000: Standby time = 2048 states 01001: Standby time = 4096 states 01010: Standby time = 16384 states 01011: Standby time = 32768 states 01100: Standby time = 65536 states 01101: Standby time = 131072 states 01110: Standby time = 262144 states 01111: Standby time = 524288 states 1XXXX: Reserved
11	STS3	1	R/W	
10	STS2	1	R/W	
9	STS1	1	R/W	
8	STS0	1	R/W	

Bit	Bit Name	Initial Value	R/W	Description
7	SLPIE	0	R/W	Sleep Instruction Exception Handling Enable Selects whether the execution of a SLEEP instruction causes sleep instruction exception handling or causes a transition to the power-down state. 0: The execution of a SLEEP instruction causes a transition to the power-down state. 1: The execution of a SLEEP instruction initiates sleep instruction exception handling. After the execution of the sleep instruction exception handling, this bit remains set to 1. Writing 0 clears this bit.
6 to 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

20.2.2 Module Stop Control Registers A and B (Function and MSTPCRB)

MSTPCRA and MSTPCRB control module stop function. Setting a bit to 1 makes the corresponding module enter module stop state, while clearing the bit to 0 clears module stop state.

- MSTPCRA

Bit	15	14	13	12	11	10	9	8
Bit Name	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8
Initial Value	0	0	0	0	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
Bit Name	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0
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

- MSTPCRB

Bit	15	14	13	12	11	10	9	8
Bit Name	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8
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
Bit Name	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0
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

- MSTPCRA

Bit	Bit Name	Initial Value	R/W	Module
15	ACSE	0	R/W	All-Module-Clock-Stop Mode Enable Enables/disables all-module-clock-stop mode for reducing current consumption by stopping the bus controller and I/O ports operations when the CPU executes the SLEEP instruction after module stop state has been set for all the on-chip peripheral modules controlled by MSTPCR. 0: All-module-clock-stop mode disabled 1: All-module-clock-stop mode enabled
14	MSTPA14	0	R/W	Reserved
13	MSTPA13	0	R/W	These bits are always read as 0. The write value should always be 0.
12	MSTPA12	0	R/W	Data transfer controller (DTC)
11	MSTPA11	1	R/W	Reserved
10	MSTPA10	1	R/W	These bits are always read as 1. The write value should always be 1.
9	MSTPA9	1	R/W	8-bit timer (TMR_3 and TMR_2)
8	MSTPA8	1	R/W	8-bit timer (TMR_1 and TMR_0)
7	MSTPA7	1	R/W	Reserved
6	MSTPA6	1	R/W	These bits are always read as 1. The write value should always be 1.
5	MSTPA5	1	R/W	D/A converter (channels 1 and 0)
4	MSTPA4	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
3	MSTPA3	1	R/W	A/D converter (unit 0)
2	MSTPA2	1	R/W	Reserved
1	MSTPA1	1	R/W	These bits are always read as 1. The write value should always be 1.
0	MSTPA0	1	R/W	16-bit timer pulse unit (TPU channels 5 to 0)

- MSTPCRB

Bit	Bit Name	Initial Value	R/W	Module
15	MSTPB15	1	R/W	Programmable pulse generator (PPG)
14	MSTPB14	1	R/W	Reserved
13	MSTPB13	1	R/W	These bits are always read as 1. The write value should always be 1.
12	MSTPB12	1	R/W	Serial communication interface_4 (SCI_4)
11	MSTPB11	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
10	MSTPB10	1	R/W	Serial communication interface_2 (SCI_2)
9	MSTPB9	1	R/W	Serial communication interface_1 (SCI_1)
8	MSTPB8	1	R/W	Serial communication interface_0 (SCI_0)
7	MSTPB7	1	R/W	Reserved
6	MSTPB6	1	R/W	These bits are always read as 1. The write value should always be 1.
5	MSTPB5	1	R/W	
4	MSTPB4	1	R/W	
3	MSTPB3	1	R/W	
2	MSTPB2	1	R/W	
1	MSTPB1	1	R/W	
0	MSTPB0	1	R/W	

20.2.3 Module Stop Control Register C (MSTPCRC)

When bits MSTPC2 to MSTPC0 are set to 1, the corresponding on-chip RAM stops. Do not set the corresponding MSTPC2 to MSTPC0 bits to 1 while accessing on-chip RAM.

Bit	15	14	13	12	11	10	9	8
Bit Name	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8
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
Bit Name	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0
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	Bit Name	Initial Value	R/W	Module
15	MSTPC15	1	R/W	Reserved
14	MSTPC14	1	R/W	These bits are always read as 1. The write value should always be 1.
13	MSTPC13	1	R/W	
12	MSTPC12	1	R/W	
11	MSTPC11	1	R/W	
10	MSTPC10	1	R/W	
9	MSTPC9	1	R/W	
8	MSTPC8	1	R/W	
7	MSTPC7	0	R/W	Reserved
6	MSTPC6	0	R/W	These bits are always read as 0. The write value should always be 0.
5	MSTPC5	0	R/W	
4	MSTPC4	0	R/W	
3	MSTPC3	0	R/W	
2	MSTPC2	0	R/W	On-chip RAM_2 (H'FF6000 to H'FF7FFF)
1	MSTPC1	0	R/W	On-chip RAM_1 (H'FF8000 to H'FF9FFF)
0	MSTPC0	0	R/W	On-chip RAM_0 (H'FFA000 to H'FFBFFF)

20.3 Multi-Clock Function

When bits ICK2 to ICK0, PCK2 to PCK0, and BCK2 to BCK0 in SCKCR are set, the main clock frequency changes at the end of the bus cycle. The CPU and bus masters operate on the operating clock specified by bits ICK2 to ICK0. The peripheral modules operate on the operating clock specified by bits PCK2 to PCK0. The external bus clock operates on the operating clock specified by bits BCK2 to BCK0. Even if the frequencies specified by bits PCK2 to PCK0 and BCK2 to BCK0 are higher than the frequency specified by bits ICK2 to ICK0, the specified values are not reflected in the peripheral module and external bus clocks. The peripheral module and external bus clocks are restricted to the operating clock specified by bits ICK2 to ICK0.

20.4 Module Stop Function

Module stop function can be set for individual on-chip peripheral modules.

When the corresponding MSTP bit in MSTPCRA, MSTPCRB, or MSTPCRC is set to 1, module operation stops at the end of the bus cycle and a transition is made to the module stop state. The CPU continues operating independently.

When the corresponding MSTP bit is cleared to 0, the module stop state is cleared and the module starts operating at the end of the bus cycle. In the module stop state, the internal states of modules other than the SCI are retained.

After the reset state is cleared, all modules other than the DMAC, DTC, and on-chip RAM are in the module stop state.

The registers of the module for which the module stop state is selected cannot be read from or written to.

20.5 Sleep Mode

20.5.1 Transition to Sleep Mode

When the SLEEP instruction is executed when the SSBY bit in SBYCR is 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other peripheral functions do not stop.

20.5.2 Clearing Sleep Mode

Sleep mode is exited by any interrupt, signals on the $\overline{\text{RES}}$ or $\overline{\text{STBY}}$ pin, and a reset caused by a watchdog timer overflow.

1. Clearing by interrupt

   When an interrupt occurs, sleep mode is exited and interrupt exception processing starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.
2. Clearing by $\overline{\text{RES}}$ pin

   Setting the $\overline{\text{RES}}$ pin level low selects the reset state. After the stipulated reset input duration, driving the $\overline{\text{RES}}$ pin high makes the CPU start the reset exception processing.
3. Clearing by $\overline{\text{STBY}}$ pin

   When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.
4. Clearing by reset caused by watchdog timer overflow

   Sleep mode is exited by an internal reset caused by a watchdog timer overflow.

20.6 All-Module-Clock-Stop Mode

When the ACSE bit is set to 1 and all modules controlled by MSTPCR are stopped (MSTPCRA, MSTPCRB = H'FFFFFFFF), or all modules except for the 8-bit timer are stopped (MSTPCRA, MSTPCRB = H'F[0 to F]FFFFFF), executing a SLEEP instruction with the SSBY bit in SBYCR cleared to 0 will cause all modules (except for the 8-bit timer* and watchdog timer), the bus controller, and the I/O ports to stop operating, and to make a transition to all-module-clock-stop mode at the end of the bus cycle.

All-module-clock-stop mode is cleared by an external interrupt (NMI or $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ11}}$ pins), $\overline{\text{RES}}$ pin input, or an internal interrupt (8-bit timer* or watchdog timer), and the CPU returns to the normal program execution state via the exception handling state. All-module-clock-stop mode is not cleared if interrupts are disabled, if interrupts other than NMI are masked on the CPU side, or if the relevant interrupt is designated as a DTC activation source.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Note: * Operation or halting of the 8-bit timer can be selected by bits MSTPA11 to MSTPA8 in MSTPCRA.

20.7 Software Standby Mode

20.7.1 Transition to Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip peripheral functions, and oscillator all stop. However, the contents of the CPU's internal registers, on-chip RAM data, and the states of on-chip peripheral functions other than the SCI, and the states of the I/O ports, are retained. Whether the address bus and bus control signals are placed in the high-impedance state or retain the output state can be specified by the OPE bit in SBYCR. In this mode the oscillator stops, allowing power consumption to be significantly reduced.

If the WDT is used as a watchdog timer, it is impossible to make a transition to software standby mode. The WDT should be stopped before the SLEEP instruction execution.

20.7.2 Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, or pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ11}}$*), or by means of the $\overline{\text{RES}}$ pin or $\overline{\text{STBY}}$ pin.

1. Clearing by interrupt

   When an NMI or IRQ0 to IRQ11* interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS4 to STS0 in SBYCR, stable clocks are supplied to the entire LSI, software standby mode is cleared, and interrupt exception handling is started.

   When clearing software standby mode with an IRQ0 to IRQ11* interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ11* is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side or has been designated as a DTC activation source.

   Note: * By setting the SSIn bit in SSIER to 1, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ11}}$ can be used as a software standby mode clearing source.

2. Clearing by $\overline{\text{RES}}$ pin

   When the $\overline{\text{RES}}$ pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire LSI. Note that the $\overline{\text{RES}}$ pin must be held low until clock oscillation settles. When the $\overline{\text{RES}}$ pin goes high, the CPU begins reset exception handling.

3. Clearing by $\overline{\text{STBY}}$ pin

   When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

20.7.3 Setting Oscillation Settling Time after Clearing Software Standby Mode

Bits STS4 to STS0 in SBYCR should be set as described below.

1. Using a crystal resonator
   Set bits STS4 to STS0 so that the standby time is at least equal to the oscillation settling time.
   Table 20.2 shows the standby times for operating frequencies and settings of bits STS4 to STS0.
2. Using an external clock
   A PLL circuit settling time is necessary. Refer to table 20.2 to set the standby time.

Table 20.2 Oscillation Settling Time Settings

						Pϕ* [MHz]			
STS4	STS3	STS2	STS1	STS0	Standby Time	35	25	20	Unit
0	0	0	0	0	Reserved	—	—	—	μs
				1	Reserved	—	—	—	
			1	0	Reserved	—	—	—	
				1	Reserved	—	—	—	
		1	0	0	Reserved	—	—	—	
				1	64	1.8	2.6	3.2	
			1	0	512	14.6	20.5	25.6	
				1	1024	29.3	41.0	51.2	
	1	0	0	0	2048	58.5	81.9	102.4	
				1	4096	0.12	0.16	0.20	ms
			1	0	16384	0.47	0.66	0.82	
				1	32768	0.94	1.31	1.64	
		1	0	0	65536	1.87	2.62	3.28	
				1	131072	3.74	5.24	6.55	
			1	0	262144	7.49	10.49	13.11	
				1	524288	14.98	20.97	26.21	
1	0	0	0	0	Reserved	—	—	—	

: Recommended time setting when using a crystal resonator.
: Recommended time setting when using an external clock.

Note: * Pϕ is the output from the peripheral module frequency divider.

STS4	STS3	STS2	STS1	STS0	Standby Time	Pϕ* [MHz] 13	10	8	Unit
0	0	0	0	0	Reserved	—	—	—	μs
				1	Reserved	—	—	—	
			1	0	Reserved	—	—	—	
				1	Reserved	—	—	—	
		1	0	0	Reserved	—	—	—	
				1	64	4.9	6.4	8.0	
			1	0	512	39.4	51.2	64.0	
				1	1024	78.8	102.4	128.0	
	1	0	0	0	2048	157.5	204.8	256.0	
				1	4096	0.32	0.41	0.51	ms
			1	0	16384	1.26	1.64	2.05	
				1	32765	2.52	3.28	4.10	
		1	0	0	65536	5.04	6.55	8.19	
				1	131072	10.08	13.11	16.38	
			1	0	262144	20.16	26.21	32.77	
				1	524288	40.33	52.43	65.54	
1	0	0	0	0	Reserved	—	—	—	

: Recommended time setting when using a crystal resonator.

: Recommended time setting when using an external clock.

Note: * ϕ is the output from the peripheral module frequency divider.

20.7.4 Software Standby Mode Application Example

Figure 20.2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in INTCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.

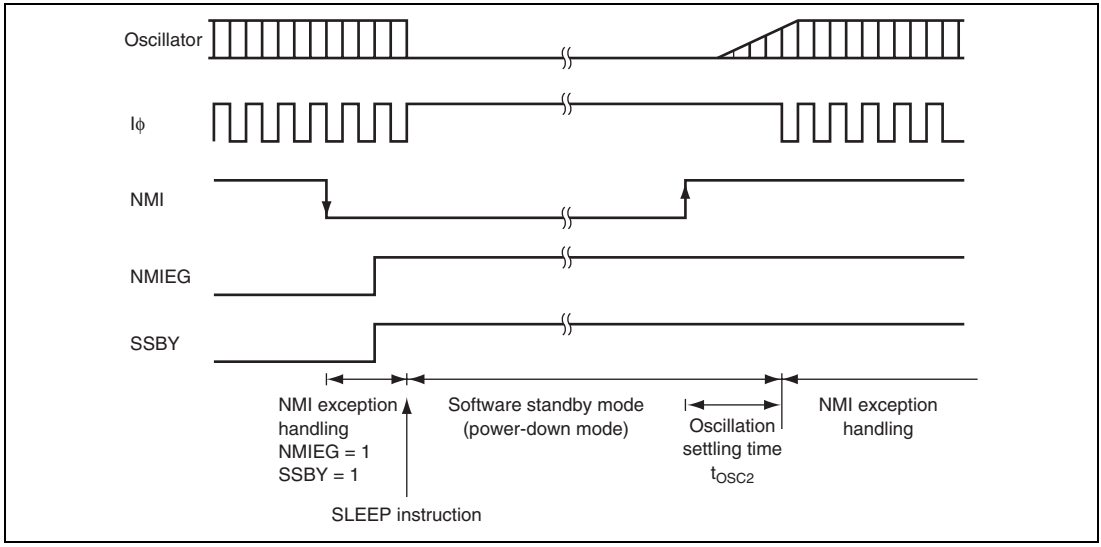


Figure 20.2 Software Standby Mode Application Example

20.8 Hardware Standby Mode

20.8.1 Transition to Hardware Standby Mode

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power consumption. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the $\overline{\text{STBY}}$ pin low. Do not change the state of the mode pins (MD2 to MD0) while this LSI is in hardware standby mode.

20.8.2 Clearing Hardware Standby Mode

Hardware standby mode is cleared by means of the $\overline{\text{STBY}}$ pin and the $\overline{\text{RES}}$ pin. When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, the reset state is entered and clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until clock oscillation settles (for details on the oscillation settling time, refer to table 20.2). When the $\overline{\text{RES}}$ pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

20.8.3 Hardware Standby Mode Timing

Figure 20.3 shows an example of hardware standby mode timing.

When the $\overline{\text{STBY}}$ pin is driven low after the $\overline{\text{RES}}$ pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the $\overline{\text{STBY}}$ pin high, waiting for the oscillation settling time, then changing the $\overline{\text{RES}}$ pin from low to high.

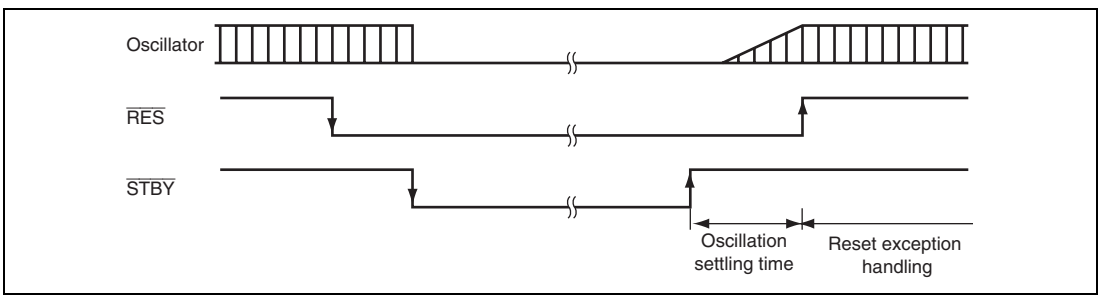


Figure 20.3 Hardware Standby Mode Timing

20.8.4 Timing Sequence at Power-On

Figure 20.4 shows the timing sequence at power-on.

At power-on, the $\overline{\text{RES}}$ pin must be driven low with the $\overline{\text{STBY}}$ pin driven high for a given time in order to clear the reset state.

To enter hardware standby mode immediately after power-on, drive the $\overline{\text{STBY}}$ pin low after exiting the reset state.

For details on clearing hardware standby mode, see section 20.8.3, Hardware Standby Mode Timing.

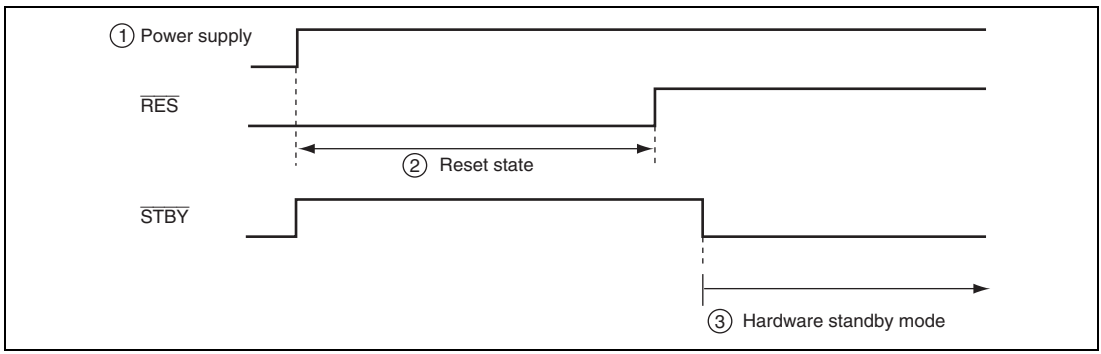


Figure 20.4 Timing Sequence at Power-On

20.9 Sleep Instruction Exception Handling

Sleep instruction exception handling is the exception handling initiated by the execution of a SLEEP instruction. Sleep instruction exception handling is always accepted while the program is in execution.

When the SLPIE bit is set to 0, the execution of a SLEEP instruction does not initiate sleep instruction exception handling. Instead, the CPU enters the power-down state. After this, generation of an exception handling request that cancels the power-down state causes the power-down state to be canceled, after which the CPU starts to handle the exception. When the SLPIE bit is set to 1, sleep instruction exception handling starts after the execution of a SLEEP instruction. Transitions to the power-down state are inhibited when sleep instruction exception handling is initiated, and the CPU immediately starts sleep instruction exception handling.

When a SLEEP instruction is executed while the SLPIE bit is cleared to 0, a transition is made to the power-down state. The power-down state is canceled by a canceling factor interrupt (see figure 20.5).

When a canceling factor interrupt is generated immediately before the execution of a SLEEP instruction, exception handling for the interrupt starts. When execution returns from the exception service routine, the SLEEP instruction is executed to enter the power-down state. In this case, the power-down state is not canceled until the next canceling factor interrupt is generated (see figure 20.6).

When the SLPIE bit is set to 1 in the service routine for a canceling factor interrupt so that the execution of a SLEEP instruction will produce sleep instruction exception handling, the operation of the system is as shown in figure 20.7. Even if a canceling factor interrupt is generated immediately before the SLEEP instruction is executed, sleep instruction exception handling is initiated by execution of the SLEEP instruction. Therefore, the CPU executes the instruction that follows the SLEEP instruction after sleep instruction exception and exception service routine without shifting to the power-down state.

When the SLPIE bit is set to 1 to start sleep exception handling, clear the SSBY bit in SBYCR to 0.

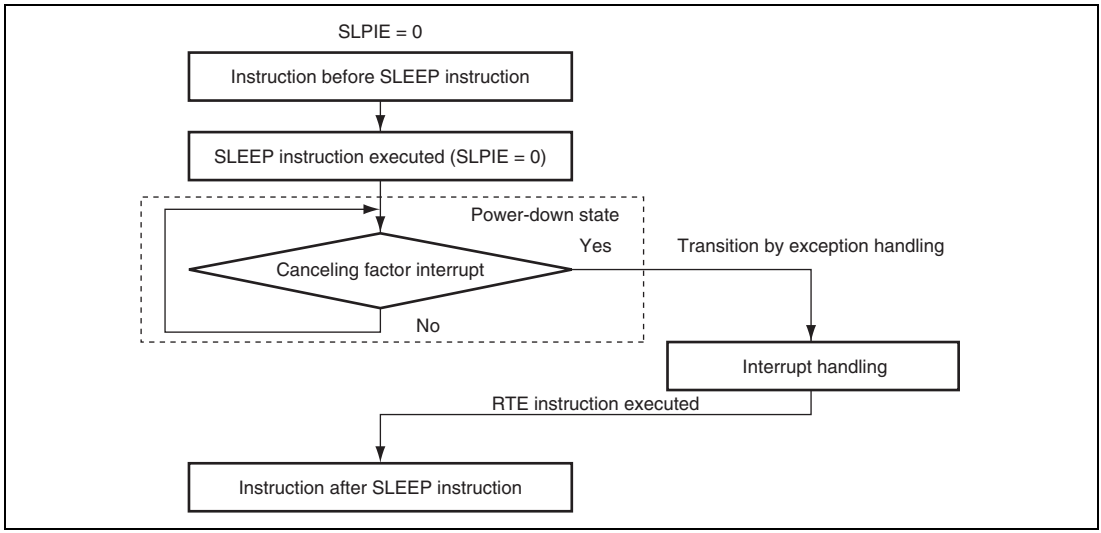


Figure 20.5 When Canceling Factor Interrupt is Generated after SLEEP Instruction Execution

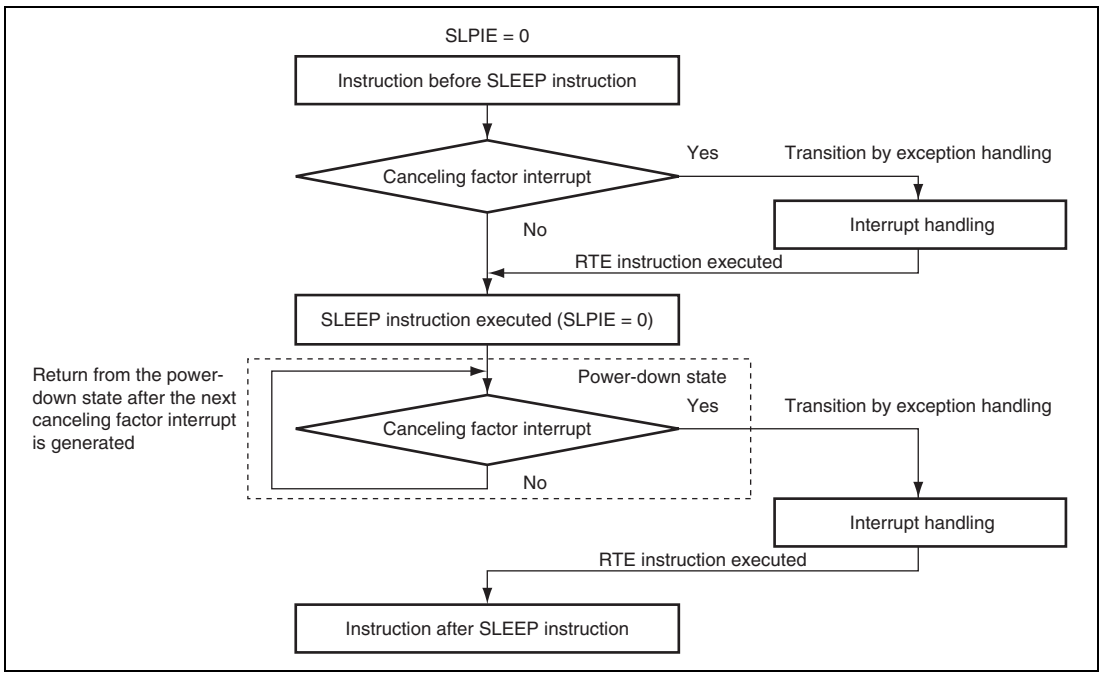


Figure 20.6 When Canceling Factor Interrupt is Generated before SLEEP Instruction Execution (Sleep Instruction Exception Handling Not Initiated)

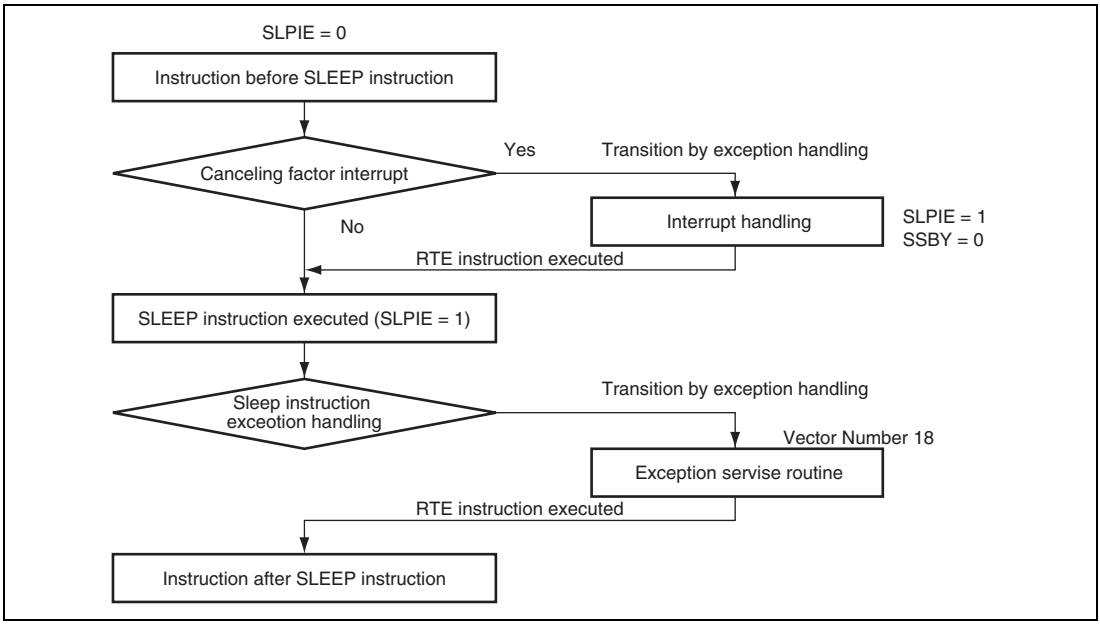


Figure 20.7 When Canceling Factor Interrupt is Generated before SLEEP Instruction Execution (Sleep Instruction Exception Handling Initiated)

20.10 Bϕ Clock Output Control

Output of the Bϕ clock can be controlled by bits PSTOP1 and POSEL1 in SCKCR, and DDR for the corresponding PA7 pin.

Clearing both bits PSTOP1 and POSEL1 to 0 enables the Bϕ clock output on the PA7 pin. When bit PSTOP1 is set to 1, the Bϕ clock output stops at the end of the bus cycle, and the Bϕ clock output goes high. When DDR for the PA7 pin is cleared to 0, the Bϕ clock output is disabled and the pin becomes an input port.

Tables 20.3 shows the states of the Bϕ pin in each processing state.

Table 20.3 Bϕ Pin (PA7) State in Each Processing State

Register Setting Value			Normal Operating State	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode		Hardware Standby Mode
DDR	PSTOP1	POSEL1				OPE = 0	OPE = 1	
0	X	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
1	0	0	Bϕ output	Bϕ output	Bϕ output	High	High	Hi-Z
1	0	1	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
1	1	X	High	High	High	High	High	Hi-Z

20.11 Usage Notes

20.11.1 I/O Port Status

In software standby mode, the I/O port states are retained. Therefore, there is no reduction in current consumption for the output current when a high-level signal is output.

20.11.2 Current Consumption during Oscillation Settling Standby Period

Current consumption increases during the oscillation settling standby period.

20.11.3 Module Stop of DMAC or DTC

Depending on the operating state of the DMAC and DTC, bits MSTPA13 and MSTPA12 may not be set to 1, respectively. The module stop state setting for the DMAC or DTC should be carried out only when the DMAC or DTC is not activated.

For details, refer to section 7, DMA Controller (DMAC), and section 8, Data Transfer Controller (DTC).

20.11.4 On-Chip Peripheral Module Interrupts

Relevant interrupt operations cannot be performed in module stop mode. Consequently, if the module stop state is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC /DTC activation source. Interrupts should therefore be disabled before entering the module stop state.

20.11.5 Writing to MSTPCRA, MSTPCRB, and MSTPCRC

MSTPCRA, MSTPCRB, and MSTPCRC should only be written to by the CPU.

Section 21 List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register addresses (address order)
- Registers are listed from the lower allocation addresses.
- Registers are classified according to functional modules.
- The number of Access Cycles indicates the number of states based on the specified reference clock. For details, refer to section 6.5.4, External Bus Interface.
- Among the internal I/O register area, addresses not listed in the list of registers are undefined or reserved addresses. Undefined and reserved addresses cannot be accessed. Do not access these addresses; otherwise, the operation when accessing these bits and subsequent operations cannot be guaranteed.

2. Register bits
- Bit configurations of the registers are listed in the same order as the register addresses.
- Reserved bits are indicated by — in the bit name column.
- Space in the bit name field indicates that the entire register is allocated to either the counter or data.
- For the registers of 16 or 32 bits, the MSB is listed first.

  Byte configuration description order is subject to big endian.

3. Register states in each operating mode
- Register states are listed in the same order as the register addresses.
- For the initialized state of each bit, refer to the register description in the corresponding section.
- The register states shown here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

21.1 Register Addresses (Address Order)

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Port 1 data direction register	P1DDR	8	H'FFB80	I/O ports	8	2Pϕ/2Pϕ
Port 2 data direction register	P2DDR	8	H'FFB81	I/O ports	8	2Pϕ/2Pϕ
Port 3 data direction register	P3DDR	8	H'FFB82	I/O ports	8	2Pϕ/2Pϕ
Port 6 data direction register	P6DDR	8	H'FFB85	I/O ports	8	2Pϕ/2Pϕ
Port A data direction register	PADDR	8	H'FFB89	I/O ports	8	2Pϕ/2Pϕ
Port B data direction register	PBDDR	8	H'FFB8A	I/O ports	8	2Pϕ/2Pϕ
Port D data direction register	PDDDR	8	H'FFB8C	I/O ports	8	2Pϕ/2Pϕ
Port E data direction register	PEDDR	8	H'FFB8D	I/O ports	8	2Pϕ/2Pϕ
Port F data direction register	PFDDR	8	H'FFB8E	I/O ports	8	2Pϕ/2Pϕ
Port 1 input buffer control register	P1ICR	8	H'FFB90	I/O ports	8	2Pϕ/2Pϕ
Port 2 input buffer control register	P2ICR	8	H'FFB91	I/O ports	8	2Pϕ/2Pϕ
Port 3 input buffer control register	P3ICR	8	H'FFB92	I/O ports	8	2Pϕ/2Pϕ
Port 5 input buffer control register	P5ICR	8	H'FFB94	I/O ports	8	2Pϕ/2Pϕ
Port 6 input buffer control register	P6ICR	8	H'FFB95	I/O ports	8	2Pϕ/2Pϕ
Port A input buffer control register	PAICR	8	H'FFB99	I/O ports	8	2Pϕ/2Pϕ
Port B input buffer control register	PBICR	8	H'FFB9A	I/O ports	8	2Pϕ/2Pϕ
Port D input buffer control register	PDICR	8	H'FFB9C	I/O ports	8	2Pϕ/2Pϕ
Port E input buffer control register	PEICR	8	H'FFB9D	I/O ports	8	2Pϕ/2Pϕ
Port F input buffer control register	PFICR	8	H'FFB9E	I/O ports	8	2Pϕ/2Pϕ
Port H register	PORTH	8	H'FFBA0	I/O ports	8	2Pϕ/2Pϕ
Port I register	PORTI	8	H'FFBA1	I/O ports	8	2Pϕ/2Pϕ
Port H data register	PHDR	8	H'FFBA4	I/O ports	8	2Pϕ/2Pϕ
Port I data register	PIDR	8	H'FFBA5	I/O ports	8	2Pϕ/2Pϕ
Port H data direction register	PHDDR	8	H'FFBA8	I/O ports	8	2Pϕ/2Pϕ
Port I data direction register	PIDDR	8	H'FFBA9	I/O ports	8	2Pϕ/2Pϕ
Port H input buffer control register	PHICR	8	H'FFBAC	I/O ports	8	2Pϕ/2Pϕ
Port I input buffer control register	PIICR	8	H'FFBAD	I/O ports	8	2Pϕ/2Pϕ
Port D pull-up MOS control register	PDPCR	8	H'FFBB4	I/O ports	8	2Pϕ/2Pϕ
Port E pull-up MOS control register	PEPCR	8	H'FFBB5	I/O ports	8	2Pϕ/2Pϕ
Port F pull-up MOS control register	PFPCR	8	H'FFBB6	I/O ports	8	2Pϕ/2Pϕ
Port H pull-up MOS control register	PHPCR	8	H'FFBB8	I/O ports	8	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Port I pull-up MOS control register	PIPCR	8	H'FFBB9	I/O ports	8	2Pϕ/2Pϕ
Port 2 open drain control register	P2ODR	8	H'FFBBC	I/O ports	8	2Pϕ/2Pϕ
Port F open drain control register	PFODR	8	H'FFBBD	I/O ports	8	2Pϕ/2Pϕ
Port function control register 0	PFCR0	8	H'FFBC0	I/O ports	8	2Pϕ/3Pϕ
Port function control register 1	PFCR1	8	H'FFBC1	I/O ports	8	2Pϕ/3Pϕ
Port function control register 2	PFCR2	8	H'FFBC2	I/O ports	8	2Pϕ/3Pϕ
Port function control register 4	PFCR4	8	H'FFBC4	I/O ports	8	2Pϕ/3Pϕ
Port function control register 6	PFCR6	8	H'FFBC6	I/O ports	8	2Pϕ/3Pϕ
Port function control register 7	PFCR7	8	H'FFBC7	I/O ports	8	2Pϕ/3Pϕ
Port function control register 9	PFCR9	8	H'FFBC9	I/O ports	8	2Pϕ/3Pϕ
Port function control register B	PFCRB	8	H'FFBCB	I/O ports	8	2Pϕ/3Pϕ
Port function control register C	PFCRC	8	H'FFBCC	I/O ports	8	2Pϕ/3Pϕ
Software standby release IRQ enable register	SSIER	16	H'FFBCE	INTC	8	2Pϕ/3Pϕ
DMA source address register_0	DSAR_0	32	H'FFC00	DMAC_0	16	2Iϕ/2Iϕ
DMA destination address register_0	DDAR_0	32	H'FFC04	DMAC_0	16	2Iϕ/2Iϕ
DMA offset register_0	DOFR_0	32	H'FFC08	DMAC_0	16	2Iϕ/2Iϕ
DMA transfer count register_0	DTCR_0	32	H'FFC0C	DMAC_0	16	2Iϕ/2Iϕ
DMA block size register_0	DBSR_0	32	H'FFC10	DMAC_0	16	2Iϕ/2Iϕ
DMA mode control register_0	DMDR_0	32	H'FFC14	DMAC_0	16	2Iϕ/2Iϕ
DMA address control register_0	DACR_0	32	H'FFC18	DMAC_0	16	2Iϕ/2Iϕ
DMA source address register_1	DSAR_1	32	H'FFC20	DMAC_1	16	2Iϕ/2Iϕ
DMA destination address register_1	DDAR_1	32	H'FFC24	DMAC_1	16	2Iϕ/2Iϕ
DMA offset register_1	DOFR_1	32	H'FFC28	DMAC_1	16	2Iϕ/2Iϕ
DMA transfer count register_1	DTCR_1	32	H'FFC2C	DMAC_1	16	2Iϕ/2Iϕ
DMA block size register_1	DBSR_1	32	H'FFC30	DMAC_1	16	2Iϕ/2Iϕ
DMA mode control register_1	DMDR_1	32	H'FFC34	DMAC_1	16	2Iϕ/2Iϕ
DMA address control register_1	DACR_1	32	H'FFC38	DMAC_1	16	2Iϕ/2Iϕ
DMA source address register_2	DSAR_2	32	H'FFC40	DMAC_2	16	2Iϕ/2Iϕ
DMA destination address register_2	DDAR_2	32	H'FFC44	DMAC_2	16	2Iϕ/2Iϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
DMA offset register_2	DOFR_2	32	H'FFC48	DMAC_2	16	2Iϕ/2Iϕ
DMA transfer count register_2	DTCR_2	32	H'FFC4C	DMAC_2	16	2Iϕ/2Iϕ
DMA block size register_2	DBSR_2	32	H'FFC50	DMAC_2	16	2Iϕ/2Iϕ
DMA mode control register_2	DMDR_2	32	H'FFC54	DMAC_2	16	2Iϕ/2Iϕ
DMA address control register_2	DACR_2	32	H'FFC58	DMAC_2	16	2Iϕ/2Iϕ
DMA source address register_3	DSAR_3	32	H'FFC60	DMAC_3	16	2Iϕ/2Iϕ
DMA destination address register_3	DDAR_3	32	H'FFC64	DMAC_3	16	2Iϕ/2Iϕ
DMA offset register_3	DOFR_3	32	H'FFC68	DMAC_3	16	2Iϕ/2Iϕ
DMA transfer count register_3	DTCR_3	32	H'FFC6C	DMAC_3	16	2Iϕ/2Iϕ
DMA block size register_3	DBSR_3	32	H'FFC70	DMAC_3	16	2Iϕ/2Iϕ
DMA mode control register_3	DMDR_3	32	H'FFC74	DMAC_3	16	2Iϕ/2Iϕ
DMA address control register_3	DACR_3	32	H'FFC78	DMAC_3	16	2Iϕ/2Iϕ
DMA module request select register_0	DMRSR_0	8	H'FFD20	DMAC_0	16	2Iϕ/2Iϕ
DMA module request select register_1	DMRSR_1	8	H'FFD21	DMAC_1	16	2Iϕ/2Iϕ
DMA module request select register_2	DMRSR_2	8	H'FFD22	DMAC_2	16	2Iϕ/2Iϕ
DMA module request select register_3	DMRSR_3	8	H'FFD23	DMAC_3	16	2Iϕ/2Iϕ
Interrupt priority register A	IPRA	16	H'FFD40	INTC	16	2Iϕ/3Iϕ
Interrupt priority register B	IPRB	16	H'FFD42	INTC	16	2Iϕ/3Iϕ
Interrupt priority register C	IPRC	16	H'FFD44	INTC	16	2Iϕ/3Iϕ
Interrupt priority register E	IPRE	16	H'FFD48	INTC	16	2Iϕ/3Iϕ
Interrupt priority register F	IPRF	16	H'FFD4A	INTC	16	2Iϕ/3Iϕ
Interrupt priority register G	IPRG	16	H'FFD4C	INTC	16	2Iϕ/3Iϕ
Interrupt priority register H	IPRH	16	H'FFD4E	INTC	16	2Iϕ/3Iϕ
Interrupt priority register I	IPRI	16	H'FFD50	INTC	16	2Iϕ/3Iϕ
Interrupt priority register K	IPRK	16	H'FFD54	INTC	16	2Iϕ/3Iϕ
Interrupt priority register L	IPRL	16	H'FFD56	INTC	16	2Iϕ/3Iϕ
IRQ sense control register H	ISCRH	16	H'FFD68	INTC	16	2Iϕ/3Iϕ
IRQ sense control register L	ISCRL	16	H'FFD6A	INTC	16	2Iϕ/3Iϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
DTC vector base register	DTCVBR	32	H'FFD80	BSC	16	2Iϕ/3Iϕ
Bus width control register	ABWCR	16	H'FFD84	BSC	16	2Iϕ/3Iϕ
Access state control register	ASTCR	16	H'FFD86	BSC	16	2Iϕ/3Iϕ
Wait control register A	WTCRA	16	H'FFD88	BSC	16	2Iϕ/3Iϕ
Wait control register B	WTCRB	16	H'FFD8A	BSC	16	2Iϕ/3Iϕ
Read strobe timing control register	RDNCR	16	H'FFD8C	BSC	16	2Iϕ/3Iϕ
$\overline{CS}$ assert period control register	CSACR	16	H'FFD8E	BSC	16	2Iϕ/3Iϕ
Idle control register	IDLCR	16	H'FFD90	BSC	16	2Iϕ/3Iϕ
Bus control register 1	BCR1	16	H'FFD92	BSC	16	2Iϕ/3Iϕ
Bus control register 2	BCR2	8	H'FFD94	BSC	16	2Iϕ/3Iϕ
Endian control register	ENDIANCR	8	H'FFD95	BSC	16	2Iϕ/3Iϕ
SRAM mode control register	SRAMCR	16	H'FFD98	BSC	16	2Iϕ/3Iϕ
Burst ROM interface control register	BROMCR	16	H'FFD9A	BSC	16	2Iϕ/3Iϕ
Address/data multiplexed I/O control register	MPXCR	16	H'FFD9C	BSC	16	2Iϕ/3Iϕ
RAM emulation register	RAMER	8	H'FFD9E	BSC	16	2Iϕ/3Iϕ
Mode control register	MDCR	16	H'FFDC0	SYSTEM	16	2Iϕ/3Iϕ
System control register	SYSCR	16	H'FFDC2	SYSTEM	16	2Iϕ/3Iϕ
System clock control register	SCKCR	16	H'FFDC4	SYSTEM	16	2Iϕ/3Iϕv
Standby control register	SBYCR	16	H'FFDC6	SYSTEM	16	2Iϕ/3Iϕ
Module stop control register A	MSTPCRA	16	H'FFDC8	SYSTEM	16	2Iϕ/3Iϕ
Module stop control register B	MSTPCRB	16	H'FFDCA	SYSTEM	16	2Iϕ/3Iϕ
Module stop control register C	MSTPCRC	16	H'FFDCC	SYSTEM	16	2Iϕ/3Iϕ
Serial extended mode register_2	SEMR_2	8	H'FFE84	SCI_2	8	2Pϕ/2Pϕ
Serial mode register_4	SMR_4	8	H'FFE90	SCI_4	8	2Pϕ/2Pϕ
Bit rate register_4	BRR_4	8	H'FFE91	SCI_4	8	2Pϕ/2Pϕ
Serial control register_4	SCR_4	8	H'FFE92	SCI_4	8	2Pϕ/2Pϕ
Transmit data register_4	TDR_4	8	H'FFE93	SCI_4	8	2Pϕ/2Pϕ
Serial status register_4	SSR_4	8	H'FFE94	SCI_4	8	2Pϕ/2Pϕ
Receive data register_4	RDR_4	8	H'FFE95	SCI_4	8	2Pϕ/2Pϕ
Smart card mode register_4	SCMR_4	8	H'FFE96	SCI_4	8	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Flash code control/status register	FCCS	8	H'FFEA8	FLASH	8	2Pϕ/2Pϕ
Flash program code select register	FPCS	8	H'FFEA9	FLASH	8	2Pϕ/2Pϕ
Flash erase code select register	FECS	8	H'FFEAA	FLASH	8	2Pϕ/2Pϕ
Flash key code register	FKEY	8	H'FFEAC	FLASH	8	2Pϕ/2Pϕ
Flash MAT select register	FMATS	8	H'FFEAD	FLASH	8	2Pϕ/2Pϕ
Flash transfer destination address register	FTDAR	8	H'FFEAE	FLASH	8	2Pϕ/2Pϕ
Timer control register_2	TCR_2	8	H'FFEC0	TMR_2	16	2Pϕ/2Pϕ
Timer control register_3	TCR_3	8	H'FFEC1	TMR_3	16	2Pϕ/2Pϕ
Timer control/status register_2	TCSR_2	8	H'FFEC2	TMR_2	16	2Pϕ/2Pϕ
Timer control/status register_3	TCSR_3	8	H'FFEC3	TMR_3	16	2Pϕ/2Pϕ
Time constant register A_2	TCORA_2	8	H'FFEC4	TMR_2	16	2Pϕ/2Pϕ
Time constant register A_3	TCORA_3	8	H'FFEC5	TMR_3	16	2Pϕ/2Pϕ
Time constant register B_2	TCORB_2	8	H'FFEC6	TMR_2	16	2Pϕ/2Pϕ
Time constant register B_3	TCORB_3	8	H'FFEC7	TMR_3	16	2Pϕ/2Pϕ
Timer counter_2	TCNT_2	8	H'FFEC8	TMR_2	16	2Pϕ/2Pϕ
Timer counter_3	TCNT_3	8	H'FFEC9	TMR_3	16	2Pϕ/2Pϕ
Timer counter control register_2	TCCR_2	8	H'FFECA	TMR_2	16	2Pϕ/2Pϕ
Timer counter control register_3	TCCR_3	8	H'FFECB	TMR_3	16	2Pϕ/2Pϕ
Timer control register_4	TCR_4	8	H'FFEE0	TPU_4	16	2Pϕ/2Pϕ
Timer mode register_4	TMDR_4	8	H'FFEE1	TPU_4	16	2Pϕ/2Pϕ
Timer I/O control register_4	TIOR_4	8	H'FFEE2	TPU_4	16	2Pϕ/2Pϕ
Timer interrupt enable register_4	TIER_4	8	H'FFEE4	TPU_4	16	2Pϕ/2Pϕ
Timer status register_4	TSR_4	8	H'FFEE5	TPU_4	16	2Pϕ/2Pϕ
Timer counter_4	TCNT_4	16	H'FFEE6	TPU_4	16	2Pϕ/2Pϕ
Timer general register A_4	TGRA_4	16	H'FFEE8	TPU_4	16	2Pϕ/2Pϕ
Timer general register B_4	TGRB_4	16	H'FFEEA	TPU_4	16	2Pϕ/2Pϕ
Timer control register_5	TCR_5	8	H'FFEF0	TPU_5	16	2Pϕ/2Pϕ
Timer mode register_5	TMDR_5	8	H'FFEF1	TPU_5	16	2Pϕ/2Pϕ
Timer I/O control register_5	TIOR_5	8	H'FFEF2	TPU_5	16	2Pϕ/2Pϕ
Timer interrupt enable register_5	TIER_5	8	H'FFEF4	TPU_5	16	2Pϕ/2Pϕ
Timer status register_5	TSR_5	8	H'FFEF5	TPU_5	16	2Pϕ/2Pϕ
Timer counter_5	TCNT_5	16	H'FFEF6	TPU_5	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Timer general register A_5	TGRA_5	16	H'FFEF8	TPU_5	16	2Pϕ/2Pϕ
Timer general register B_5	TGRB_5	16	H'FFEFA	TPU_5	16	2Pϕ/2Pϕ
DTC enable register A	DTCERA	16	H'FFF20	INTC	16	2Iϕ/3Iϕ
DTC enable register B	DTCERB	16	H'FFF22	INTC	16	2Iϕ/3Iϕ
DTC enable register C	DTCERC	16	H'FFF24	INTC	16	2Iϕ/3Iϕ
DTC enable register D	DTCERD	16	H'FFF26	INTC	16	2Iϕ/3Iϕ
DTC enable register E	DTCERE	16	H'FFF28	INTC	16	2Iϕ/3Iϕ
DTC control register	DTCCR	8	H'FFF30	INTC	16	2Iϕ/3Iϕ
Interrupt control register	INTCR	8	H'FFF32	INTC	16	2Iϕ/3Iϕ
CPU priority control register	CPUPCR	8	H'FFF33	INTC	16	2Iϕ/3Iϕ
IRQ enable register	IER	16	H'FFF34	INTC	16	2Iϕ/3Iϕ
IRQ status register	ISR	16	H'FFF36	INTC	16	2Iϕ/3Iϕ
Port 1 register	PORT1	8	H'FFF40	I/O ports	8	2Pϕ/—
Port 2 register	PORT2	8	H'FFF41	I/O ports	8	2Pϕ/—
Port 3 register	PORT3	8	H'FFF42	I/O ports	8	2Pϕ/—
Port 5 register	PORT5	8	H'FFF44	I/O ports	8	2Pϕ/—
Port 6 register	PORT6	8	H'FFF45	I/O ports	8	2Pϕ/—
Port A register	PORTA	8	H'FFF49	I/O ports	8	2Pϕ/—
Port B register	PORTB	8	H'FFF4A	I/O ports	8	2Pϕ/—
Port D register	PORTD	8	H'FFF4C	I/O ports	8	2Pϕ/—
Port E register	PORTE	8	H'FFF4D	I/O ports	8	2Pϕ/—
Port F register	PORTF	8	H'FFF4E	I/O ports	8	2Pϕ/—
Port 1 data register	P1DR	8	H'FFF50	I/O ports	8	2Pϕ/2Pϕ
Port 2 data register	P2DR	8	H'FFF51	I/O ports	8	2Pϕ/2Pϕ
Port 3 data register	P3DR	8	H'FFF52	I/O ports	8	2Pϕ/2Pϕ
Port 6 data register	P6DR	8	H'FFF55	I/O ports	8	2Pϕ/2Pϕ
Port A data register	PADR	8	H'FFF59	I/O ports	8	2Pϕ/2Pϕ
Port B data register	PBDR	8	H'FFF5A	I/O ports	8	2Pϕ/2Pϕ
Port D data register	PDDR	8	H'FFF5C	I/O ports	8	2Pϕ/2Pϕ
Port E data register	PEDR	8	H'FFF5D	I/O ports	8	2Pϕ/2Pϕ
Port F data register	PFDR	8	H'FFF5E	I/O ports	8	2Pϕ/2Pϕ
Serial mode register_2	SMR_2	8	H'FFF60	SCI_2	8	2Pϕ/2Pϕ
Bit rate register_2	BRR_2	8	H'FFF61	SCI_2	8	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Serial control register_2	SCR_2	8	H'FFF62	SCI_2	8	2Pϕ/2Pϕ
Transmit data register_2	TDR_2	8	H'FFF63	SCI_2	8	2Pϕ/2Pϕ
Serial status register_2	SSR_2	8	H'FFF64	SCI_2	8	2Pϕ/2Pϕ
Receive data register_2	RDR_2	8	H'FFF65	SCI_2	8	2Pϕ/2Pϕ
Smart card mode register_2	SCMR_2	8	H'FFF66	SCI_2	8	2Pϕ/2Pϕ
D/A data register 0	DADR0	8	H'FFF68	D/A	8	2Pϕ/2Pϕ
D/A data register 1	DADR1	8	H'FFF69	D/A	8	2Pϕ/2Pϕ
D/A control register 01	DACR01	8	H'FFF6A	D/A	8	2Pϕ/2Pϕ
PPG output control register	PCR	8	H'FFF76	PPG	8	2Pϕ/2Pϕ
PPG output mode register	PMR	8	H'FFF77	PPG	8	2Pϕ/2Pϕ
Next data enable register H	NDERH	8	H'FFF78	PPG	8	2Pϕ/2Pϕ
Next data enable register L	NDERL	8	H'FFF79	PPG	8	2Pϕ/2Pϕ
Output data register H	PODRH	8	H'FFF7A	PPG	8	2Pϕ/2Pϕ
Output data register L	PODRL	8	H'FFF7B	PPG	8	2Pϕ/2Pϕ
Next data register H*	NDRH	8	H'FFF7C	PPG	8	2Pϕ/2Pϕ
Next data register L*	NDRL	8	H'FFF7D	PPG	8	2Pϕ/2Pϕ
Next data register H*	NDRH	8	H'FFF7E	PPG	8	2Pϕ/2Pϕ
Next data register L*	NDRL	8	H'FFF7F	PPG	8	2Pϕ/2Pϕ
Serial mode register_0	SMR_0	8	H'FFF80	SCI_0	8	2Pϕ/2Pϕ
Bit rate register_0	BRR_0	8	H'FFF81	SCI_0	8	2Pϕ/2Pϕ
Serial control register_0	SCR_0	8	H'FFF82	SCI_0	8	2Pϕ/2Pϕ
Transmit data register_0	TDR_0	8	H'FFF83	SCI_0	8	2Pϕ/2Pϕ
Serial status register_0	SSR_0	8	H'FFF84	SCI_0	8	2Pϕ/2Pϕ
Receive data register_0	RDR_0	8	H'FFF85	SCI_0	8	2Pϕ/2Pϕ
Smart card mode register_0	SCMR_0	8	H'FFF86	SCI_0	8	2Pϕ/2Pϕ
Serial mode register_1	SMR_1	8	H'FFF88	SCI_1	8	2Pϕ/2Pϕ
Bit rate register_1	BRR_1	8	H'FFF89	SCI_1	8	2Pϕ/2Pϕ
Serial control register_1	SCR_1	8	H'FFF8A	SCI_1	8	2Pϕ/2Pϕ
Transmit data register_1	TDR_1	8	H'FFF8B	SCI_1	8	2Pϕ/2Pϕ
Serial status register_1	SSR_1	8	H'FFF8C	SCI_1	8	2Pϕ/2Pϕ
Receive data register_1	RDR_1	8	H'FFF8D	SCI_1	8	2Pϕ/2Pϕ
Smart card mode register_1	SCMR_1	8	H'FFF8E	SCI_1	8	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
A/D data register A	ADDRA	16	H'FFF90	A/D	16	2Pϕ/2Pϕ
A/D data register B	ADDRB	16	H'FFF92	A/D	16	2Pϕ/2Pϕ
A/D data register C	ADDRC	16	H'FFF94	A/D	16	2Pϕ/2Pϕ
A/D data register D	ADDRD	16	H'FFF96	A/D	16	2Pϕ/2Pϕ
A/D data register E	ADDRE	16	H'FFF98	A/D	16	2Pϕ/2Pϕ
A/D data register F	ADDRF	16	H'FFF9A	A/D	16	2Pϕ/2Pϕ
A/D data register G	ADDRG	16	H'FFF9C	A/D	16	2Pϕ/2Pϕ
A/D data register H	ADDRH	16	H'FFF9E	A/D	16	2Pϕ/2Pϕ
A/D control/status register	ADCSR	8	H'FFFA0	A/D	16	2Pϕ/2Pϕ
A/D control register	ADCR	8	H'FFFA1	A/D	16	2Pϕ/2Pϕ
Timer control/status register	TCSR	8	H'FFFA4	WDT		2Pϕ/3Pϕ
Timer counter	TCNT	8	H'FFFA5	WDT		2Pϕ/3Pϕ
Reset control/status register	RSTCSR	8	H'FFFA7	WDT		2Pϕ/3Pϕ
Timer control register_0	TCR_0	8	H'FFFB0	TMR_0	16	2Pϕ/2Pϕ
Timer control register_1	TCR_1	8	H'FFFB1	TMR_1	16	2Pϕ/2Pϕ
Timer control/status register_0	TCSR_0	8	H'FFFB2	TMR_0	16	2Pϕ/2Pϕ
Timer control/status register_1	TCSR_1	8	H'FFFB3	TMR_1	16	2Pϕ/2Pϕ
Time constant register A_0	TCORA_0	8	H'FFFB4	TMR_0	16	2Pϕ/2Pϕ
Time constant register A_1	TCORA_1	8	H'FFFB5	TMR_1	16	2Pϕ/2Pϕ
Time constant register B_0	TCORB_0	8	H'FFFB6	TMR_0	16	2Pϕ/2Pϕ
Time constant register B_1	TCORB_1	8	H'FFFB7	TMR_1	16	2Pϕ/2Pϕ
Timer counter_0	TCNT_0	8	H'FFFB8	TMR_0	16	2Pϕ/2Pϕ
Timer counter_1	TCNT_1	8	H'FFFB9	TMR_1	16	2Pϕ/2Pϕ
Timer counter control register_0	TCCR_0	8	H'FFFBA	TMR_0	16	2Pϕ/2Pϕ
Timer counter control register_1	TCCR_1	8	H'FFFBB	TMR_1	16	2Pϕ/2Pϕ
Timer start register	TSTR	8	H'FFFBC	TPU	16	2Pϕ/2Pϕ
Timer synchronous register	TSYR	8	H'FFFBD	TPU	16	2Pϕ/2Pϕ
Timer control register_0	TCR_0	8	H'FFFC0	TPU_0	16	2Pϕ/2Pϕ
Timer mode register_0	TMDR_0	8	H'FFFC1	TPU_0	16	2Pϕ/2Pϕ
Timer I/O control register H_0	TIORH_0	8	H'FFFC2	TPU_0	16	2Pϕ/2Pϕ
Timer I/O control register L_0	TIORL_0	8	H'FFFC3	TPU_0	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Timer interrupt enable register_0	TIER_0	8	H'FFFC4	TPU_0	16	2Pϕ/2Pϕ
Timer status register_0	TSR_0	8	H'FFFC5	TPU_0	16	2Pϕ/2Pϕ
Timer counter_0	TCNT_0	16	H'FFFC6	TPU_0	16	2Pϕ/2Pϕ
Timer general register A_0	TGRA_0	16	H'FFFC8	TPU_0	16	2Pϕ/2Pϕ
Timer general register B_0	TGRB_0	16	H'FFFCA	TPU_0	16	2Pϕ/2Pϕ
Timer general register C_0	TGRC_0	16	H'FFFCC	TPU_0	16	2Pϕ/2Pϕ
Timer general register D_0	TGRD_0	16	H'FFFCE	TPU_0	16	2Pϕ/2Pϕ
Timer control register_1	TCR_1	8	H'FFFD0	TPU_1	16	2Pϕ/2Pϕ
Timer mode register_1	TMDR_1	8	H'FFFD1	TPU_1	16	2Pϕ/2Pϕ
Timer I/O control register_1	TIOR_1	8	H'FFFD2	TPU_1	16	2Pϕ/2Pϕ
Timer interrupt enable register_1	TIER_1	8	H'FFFD4	TPU_1	16	2Pϕ/2Pϕ
Timer status register_1	TSR_1	8	H'FFFD5	TPU_1	16	2Pϕ/2Pϕ
Timer counter_1	TCNT_1	16	H'FFFD6	TPU_1	16	2Pϕ/2Pϕ
Timer general register A_1	TGRA_1	16	H'FFFD8	TPU_1	16	2Pϕ/2Pϕ
Timer general register B_1	TGRB_1	16	H'FFFDA	TPU_1	16	2Pϕ/2Pϕ
Timer control register_2	TCR_2	8	H'FFFE0	TPU_2	16	2Pϕ/2Pϕ
Timer mode register_2	TMDR_2	8	H'FFFE1	TPU_2	16	2Pϕ/2Pϕ
Timer I/O control register_2	TIOR_2	8	H'FFFE2	TPU_2	16	2Pϕ/2Pϕ
Timer interrupt enable register_2	TIER_2	8	H'FFFE4	TPU_2	16	2Pϕ/2Pϕ
Timer status register_2	TSR_2	8	H'FFFE5	TPU_2	16	2Pϕ/2Pϕ
Timer counter_2	TCNT_2	16	H'FFFE6	TPU_2	16	2Pϕ/2Pϕ
Timer general register A_2	TGRA_2	16	H'FFFE8	TPU_2	16	2Pϕ/2Pϕ
Timer general register B_2	TGRB_2	16	H'FFFEA	TPU_2	16	2Pϕ/2Pϕ
Timer control register_3	TCR_3	8	H'FFFF0	TPU_3	16	2Pϕ/2Pϕ
Timer mode register_3	TMDR_3	8	H'FFFF1	TPU_3	16	2Pϕ/2Pϕ
Timer I/O control register H_3	TIORH_3	8	H'FFFF2	TPU_3	16	2Pϕ/2Pϕ
Timer I/O control register L_3	TIORL_3	8	H'FFFF3	TPU_3	16	2Pϕ/2Pϕ
Timer interrupt enable register_3	TIER_3	8	H'FFFF4	TPU_3	16	2Pϕ/2Pϕ
Timer status register_3	TSR_3	8	H'FFFF5	TPU_3	16	2Pϕ/2Pϕ
Timer counter_3	TCNT_3	16	H'FFFF6	TPU_3	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Timer general register A_3	TGRA_3	16	H'FFFF8	TPU_3	16	2Pϕ/2Pϕ
Timer general register B_3	TGRB_3	16	H'FFFFA	TPU_3	16	2Pϕ/2Pϕ
Timer general register C_3	TGRC_3	16	H'FFFFC	TPU_3	16	2Pϕ/2Pϕ
Timer general register D_3	TGRD_3	16	H'FFFFE	TPU_3	16	2Pϕ/2Pϕ

Note: * When the same output trigger is specified for pulse output groups 2 and 3 by the PCR setting, the NDRH address is H'FFF7C. When different output triggers are specified, the NDRH addresses for pulse output groups 2 and 3 are H'FFF7E and H'FFF7C, respectively. Similarly, When the same output trigger is specified for pulse output groups 0 and 1 by the PCR setting, the NDRL address is H'FFF7D. When different output triggers are specified, the NDRL addresses for pulse output groups 0 and 1 are H'FFF7F and H'FFF7D, respectively.

21.2 Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, and 16-bit and 32-bit registers are shown as 2 or 4 lines, respectively.

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	I/O ports
P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P6DDR	—	—	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	
PADDR	PA7DDR	PA6DDR	PA5DDR	PA4DDR	PA3DDR	PA2DDR	PA1DDR	PA0DDR	
PBDDR	—	—	—	—	PB3DDR	PB2DDR	PB1DDR	PB0DDR	
PDDDR	PD7DDR	PD6DDR	PD5DDR	PD4DDR	PD3DDR	PD2DDR	PD1DDR	PD0DDR	
PEDDR	PE7DDR	PE6DDR	PE5DDR	PE4DDR	PE3DDR	PE2DDR	PE1DDR	PE0DDR	
PFDDR	PF7DDR	PF6DDR	PF5DDR	PF4DDR	PF3DDR	PF2DDR	PF1DDR	PF0DDR	
P1ICR	P17ICR	P16ICR	P15ICR	P14ICR	P13ICR	P12ICR	P11ICR	P10ICR	
P2ICR	P27ICR	P26ICR	P25ICR	P24ICR	P23ICR	P22ICR	P21ICR	P20ICR	
P3ICR	P37ICR	P36ICR	P35ICR	P34ICR	P33ICR	P32ICR	P31ICR	P30ICR	
P5ICR	P57ICR	P56ICR	P55ICR	P54ICR	P53ICR	P52ICR	P51ICR	P50ICR	
P6ICR	—	—	P65ICR	P64ICR	P63ICR	P62ICR	P61ICR	P60ICR	
PAICR	PA7ICR	PA6ICR	PA5ICR	PA4ICR	PA3ICR	PA2ICR	PA1ICR	PA0ICR	
PBICR	—	—	—	—	PB3ICR	PB2ICR	PB1ICR	PB0ICR	
PDICR	PD7ICR	PD6ICR	PD5ICR	PD4ICR	PD3ICR	PD2ICR	PD1ICR	PD0ICR	
PEICR	PE7ICR	PE6ICR	PE5ICR	PE4ICR	PE3ICR	PE2ICR	PE1ICR	PE0ICR	
PFICR	PF7ICR	PF6ICR	PF5ICR	PF4ICR	PF3ICR	PF2ICR	PF1ICR	PF0ICR	
PORTH	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0	
PORTI	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0	
PHDR	PH7DR	PH6DR	PH5DR	PH4DR	PH3DR	PH2DR	PH1DR	PH0DR	
PIDR	PI7DR	PI6DR	PI5DR	PI4DR	PI3DR	PI2DR	PI1DR	PI0DR	
PHDDR	PH7DDR	PH6DDR	PH5DDR	PH4DDR	PH3DDR	PH2DDR	PH1DDR	PH0DDR	
PIDDR	PI7DDR	PI6DDR	PI5DDR	PI4DDR	PI3DDR	PI2DDR	PI1DDR	PI0DDR	
PHICR	PH7ICR	PH6ICR	PH5ICR	PH4ICR	PH3ICR	PH2ICR	PH1ICR	PH0ICR	
PIICR	PI7ICR	PI6ICR	PI5ICR	PI4ICR	PI3ICR	PI2ICR	PI1ICR	PI0ICR	
PDPCR	PD7PCR	PD6PCR	PD5PCR	PD4PCR	PD3PCR	PD2PCR	PD1PCR	PD0PCR	
PEPCR	PE7PCR	PE6PCR	PE6PCR	PE4PCR	PE3PCR	PE2PCR	PE1PCR	PE0PCR	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PFPCR	PF7PCR	PF6PCR	PF5PCR	PF4PCR	PF3PCR	PF2PCR	PF1PCR	PF0PCR	I/O ports
PHPCR	PH7PCR	PH6PCR	PH5PCR	PH4PCR	PH3PCR	PH2PCR	PH1PCR	PH0PCR	
PIPCR	PI7PCR	PI6PCR	PI5PCR	PI4PCR	PI3PCR	PI2PCR	PI1PCR	PI0PCR	
P2ODR	P27ODR	P26ODR	P25ODR	P24ODR	P23ODR	P22ODR	P21ODR	P20ODR	
PFODR	PF7ODR	PF6ODR	PF5ODR	PF4ODR	PF3ODR	PF2ODR	PF1ODR	PF0ODR	
PFCR0	CS7E	CS6E	CS5E	CS4E	CS3E	CS2E	CS1E	CS0E	
PFCR1	CS7SA	CS7SB	CS6SA	CS6SB	CS5SA	CS5SB	—	—	
PFCR2	—	CS2S	BSS	BSE	—	RDWRE	ASOE	—	
PFCR4	A23E	A22E	A21E	A20E	A19E	A18E	A17E	A16E	
PFCR6	—	LHWROE	—	—	TCLKS	—	—	—	
PFCR7	DMAS3A	DMAS3B	DMAS2A	DMAS2B	DMAS1A	DMAS1B	DMAS0A	DMAS0B	
PFCR9	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B	
PFCRB	—	—	—	—	ITS11	ITS10	ITS9	ITS8	
PFCRC	ITS7	ITS6	ITS5	ITS4	ITS3	ITS2	ITS1	ITS0	
SSIER	—	—	—	—	SSI11	SSI10	SSI9	SSI8	INTC
	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0	
DSAR_0									DMAC_0
DDAR_0									
DOFR_0									
DTCR_0									
DBSR_0	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZH15	BKSZH14	BKSZH13	BKSZH12	BKSZH11	BKSZH10	BKSZH9	BKSZH8	
	BKSZH7	BKSZH6	BKSZH5	BKSZH4	BKSZH3	BKSZH2	BKSZH1	BKSZH0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DMDR_0	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	DMAC_0
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_0	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_1									DMAC_1
DDAR_1									
DOFR_1									
DTCR_1									
DBSR_1	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZH15	BKSZH14	BKSZH13	BKSZH12	BKSZH11	BKSZH10	BKSZH9	BKSZH8	
	BKSZH7	BKSZH6	BKSZH5	BKSZH4	BKSZH3	BKSZH2	BKSZH1	BKSZH0	
DMDR_1	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_1	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DSAR_2									DMAC_2
DDAR_2									
DOFR_2									
DTCR_2									
DBSR_2	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZH15	BKSZH14	BKSZH13	BKSZH12	BKSZH11	BKSZH10	BKSZH9	BKSZH8	
	BKSZH7	BKSZH6	BKSZH5	BKSZH4	BKSZH3	BKSZH2	BKSZH1	BKSZH0	
DMDR_2	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_2	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_3									DMAC_3
DDAR_3									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DOFR_3									DMAC_3
DTCR_3									
DBSR_3	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZH15	BKSZH14	BKSZH13	BKSZH12	BKSZH11	BKSZH10	BKSZH9	BKSZH8	
	BKSZH7	BKSZH6	BKSZH5	BKSZH4	BKSZH3	BKSZH2	BKSZH1	BKSZH0	
DMDR_3	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_3	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DMRSR_0									DMAC_0
DMRSR_1									DMAC_1
DMRSR_2									DMAC_2
DMRSR_3									DMAC_3
IPRA	—	IPRA14	IPRA13	IPRA12	—	IPRA10	IPRA9	IPRA8	INTC
	—	IPRA6	IPRA5	IPRA4	—	IPRA2	IPRA1	IPRA0	
IPRB	—	IPRB14	IPRB13	IPRB12	—	IPRB10	IPRB9	IPRB8	
	—	IPRB6	IPRB5	IPRB4	—	IPRB2	IPRB1	IPRB0	
IPRC	—	IPRC14	IPRC13	IPRC12	—	IPRC10	IPRC9	IPRC8	
	—	IPRC6	IPRC5	IPRC4	—	IPRC2	IPRC1	IPRC0	
IPRE	—	—	—	—	—	IPRE10	IPRE9	IPRE8	
	—	—	—	—	—	—	—	—	
IPRF	—	—	—	—	—	IPRF10	IPRF9	IPRF8	
	—	IPRF6	IPRF5	IPRF4	—	IPRF2	IPRF1	IPRF0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
IPRG	—	IPRG14	IPRG13	IPRG12	—	IPRG10	IPRG9	IPRG8	INTC
	—	IPRG6	IPRG5	IPRG4	—	IPRG2	IPRG1	IPRG0	
IPRH	—	IPRH14	IPRH13	IPRH12	—	IPRH10	IPRH9	IPRH8	
	—	IPRH6	IPRH5	IPRH4	—	IPRH2	IPRH1	IPRH0	
IPRI	—	IPRI14	IPRI13	IPRI12	—	IPRI10	IPRI9	IPRI8	
	—	IPRI6	IPRI5	IPRI4	—	IPRI2	IPRI1	IPRI0	
IPRK	—	IPRK14	IPRK13	IPRK12	—	—	—	—	
	—	IPRK6	IPRK5	IPRK4	—	IPRK2	IPRK1	IPRK0	
IPRL	—	IPRL14	IPRL13	IPRL12	—	—	—	—	
	—	IPRL6	IPRL5	IPRL4	—	—	—	—	
ISCRH	—	—	—	—	—	—	—	—	
	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF	
ISCRL	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF	
	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF	
DTCVBR									BSC
ABWCR	ABWH7	ABWH6	ABWH5	ABWH4	ABWH3	ABWH2	ABWH1	ABWH0	
	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1	ABWL0	
ASTCR	AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0	
	—	—	—	—	—	—	—	—	
WTCRA	—	W72	W71	W70	—	W62	W61	W60	
	—	W52	W51	W50	—	W42	W41	W40	
WTCRB	—	W32	W31	W30	—	W22	W21	W20	
	—	W12	W11	W10	—	W02	W01	W00	
RDNCR	RDN7	RDN6	RDN5	RDN4	RDN3	RDN2	RDN1	RDN0	
	—	—	—	—	—	—	—	—	
CSACR	CSXH7	CSXH6	CSXH5	CSXH4	CSXH3	CSXH2	CSXH1	CSXH0	
	CSXT7	CSXT6	CSXT5	CSXT4	CSXT3	CSXT2	CSXT1	CSXT0	
IDLCR	IDLS3	IDLS2	IDLS1	IDLS0	IDLCB1	IDLCB0	IDLCA1	IDLCA0	
	IDLSEL7	IDLSEL6	IDLSEL5	IDLSEL4	IDLSEL3	IDLSEL2	IDLSEL1	IDLSEL0	
BCR1	BRLE	BREQOE	—	—	—	—	WDBE	WAITE	
	DKC	—	—	—	—	—	—	—	
BCR2	—	—	—	IBCCS	—	—	—	PWDBE	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
ENDIANCR	LE7	LE6	LE5	LE4	LE3	LE2	—	—	BSC
SRAMCR	BCSEL7	BCSEL6	BCSEL5	BCSEL4	BCSEL3	BCSEL2	BCSEL1	BCSEL0	
	—	—	—	—	—	—	—	—	
BROMCR	BSRM0	BSTS02	BSTS01	BSTS00	—	—	BSWD01	BSWD00	
	BSRM1	BSTS12	BSTS11	BSTS10	—	—	BSWD11	BSWD10	
MPXCR	MPXE7	MPXE6	MPXE5	MPXE4	MPXE3	—	—	—	
	—	—	—	—	—	—	—	ADDEX	
RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0	
MDCR	—	—	—	—	MDS3	MDS2	MDS1	MDS0	SYSTEM
	—	—	—	—	—	—	—	—	
SYSCR	—	—	MACS	—	FETCHMD	—	EXPE	RAME	
	—	—	—	—	—	—	DTCMD	—	
SCKCR	PSTOP1	—	—	—	—	ICK2	ICK1	ICK0	
	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0	
SBYCR	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0	
	SLPIE	—	—	—	—	—	—	—	
MSTPCRA	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8	
	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0	
MSTPCRB	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8	
	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0	
MSTPCRC	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8	
	MSTPC7	MSTPC5	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0	
SEMR_2	—	—	—	—	ABCS	ACS2	ACS1	ACS0	SCI_2
SMR_4*	C/$\overline{A}$ (GM)	CHR (BLK)	PE (PE)	O/$\overline{E}$ (O/$\overline{E}$)	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_4
BRR_4									
SCR_4*	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_4									
SSR_4*	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_4									
SCMR_4	—	—	—	—	SDIR	SINV	—	SMIF	
FCCS	—	—	—	FLER	—	—	—	SCO	FLASH
FPCS	—	—	—	—	—	—	—	PPVS	
FECS	—	—	—	—	—	—	—	EPVB	
FKEY	K7	K6	K5	K4	K3	K2	K1	K0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
FMATS	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0	FLASH
FTDAR	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0	
TCR_2	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_2
TCR_3	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_3
TCSR_2	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0	TMR_2
TCSR_3	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_3
TCORA_2									TMR_2
TCORA_3									TMR_3
TCORB_2									TMR_2
TCORB_3									TMR_3
TCNT_2									TMR_2
TCNT_3									TMR_3
TCCR_2	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_2
TCCR_3	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_3
TCR_4	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_4
TMDR_4	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_4	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_4	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_4	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_4									
TGRA_4									
TGRB_4									
TCR_5	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_5
TMDR_5	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_5	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_5	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_5	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_5									
TGRA_5									
TGRB_5									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DTCERA	DTCEA15	DTCEA14	DTCEA13	DTCEA12	DTCEA11	DTCEA10	DTCEA9	DTCEA8	INTC
	DTCEA7	DTCEA6	DTCEA5	DTCEA4	—	—	—	—	
DTCERB	DTCEB15	—	DTCEB13	DTCEB12	DTCEB11	DTCEB10	DTCEB9	DTCEB8	
	DTCEB7	DTCEB6	DTCEB5	DTCEB4	DTCEB3	DTCEB2	DTCEB1	DTCEB0	
DTCERC	DTCEC15	DTCEC14	DTCEC13	DTCEC12	DTCEC11	DTCEC10	DTCEC9	DTCEC8	
	DTCEC7	DTCEC6	DTCEC5	DTCEC4	DTCEC3	DTCEC2	—	—	
DTCERD	—	—	DTCED13	DTCED12	DTCED11	DTCED10	—	—	
	—	—	DTCED5	DTCED4	DTCED3	DTCED2	DTCED1	DTCED0	
DTCERE	—	—	DTCEE13	DTCEE12	—	—	—	—	
	—	—	—	—	—	—	—	—	
DTCCR	—	—	—	RRS	RCHNE	—	—	ERR	
INTCR	—	—	INTM1	INTM0	NMIEG	—	—	—	
CPUPCR	CPUPCE	DTCP2	DTCP1	DTCP0	IPSETE	CPUP2	CPUP1	CPUP0	
IER	—	—	—	—	IRQ11E	IRQ10E	IRQ9E	IRQ8E	
	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	
ISR	—	—	—	—	IRQ11F	IRQ10F	IRQ9F	IRQ8F	
	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
PORT1	P17	P16	P15	P14	P13	P12	P11	P10	I/O ports
PORT2	P27	P26	P25	P24	P23	P22	P21	P20	
PORT3	P37	P36	P35	P34	P33	P32	P31	P30	
PORT5	P57	P56	P55	P54	P53	P52	P51	P50	
PORT6	—	—	P65	P64	P63	P62	P61	P60	
PORTA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	
PORTB	—	—	—	—	PB3	PB2	PB1	PB0	
PORTD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	
PORTE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	
PORTF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	P27DR	P26DR	P25DR	P24DR	P23DR	P22DR	P21DR	P20DR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P6DR	—	—	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
PADR	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR	
PBDR	—	—	—	—	PB3DR	PB2DR	PB1DR	PA0DR	
PDDR	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR	
PEDR	PE7DR	PE6DR	PE5DR	PE4DR	PE3DR	PE2DR	PE1DR	PE0DR	
PFDR	PF7DR	PF6DR	PF5DR	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SMR_2*1	C/$\overline{A}$ (GM)	CHR (BLK)	PE (PE)	O/$\overline{E}$ (O/$\overline{E}$)	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_2
BRR_2									
SCR_2*1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_2									
SSR_2*1	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_2									
SCMR_2	—	—	—	—	SDIR	SINV	—	SMIF	
DADR0									D/A
DADR1									
DACR01	DAOE1	DAOE0	DAE	—	—	—	—	—	
PCR	G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0	PPG
PMR	G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV	
NDERH	NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8	
NDERL	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0	
PODRH	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	
PODRL	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0	
NDRH*2	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8	
NDRL*2	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0	
NDRH*2	—	—	—	—	NDR11	NDR10	NDR9	NDR8	
NDRL*2	—	—	—	—	NDR3	NDR2	NDR1	NDR0	
SMR_0*1	C/$\overline{A}$ (GM)	CHR (BLK)	PE (PE)	O/$\overline{E}$ (O/$\overline{E}$)	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_0
BRR_0									
SCR_0*1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_0									
SSR_0*1	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_0									
SCMR_0	—	—	—	—	SDIR	SINV	—	SMIF	
SMR_1*1	C/$\overline{A}$ (GM)	CHR (BLK)	PE (PE)	O/$\overline{E}$ (O/$\overline{E}$)	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_1
BRR_1									
SCR_1*1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_1									
SSR_1*1	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_1									
SCMR_1	—	—	—	—	SDIR	SINV	—	SMIF	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
ADDRA									A/D
ADDRB									
ADDRC									
ADDRD									
ADDRE									
ADDRF									
ADDRG									
ADDRH									
ADCSR	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0	
ADCR	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—	
TCSR	OVF	WT/$\overline{IT}$	TME	—	—	CKS2	CKS1	CKS0	WDT
TCNT									
RSTCSR	WOVF	RSTE	—	—	—	—	—	—	
TCR_0	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_0
TCR_1	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_1
TCSR_0	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0	TMR_0
TCSR_1	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_1
TCORA_0									TMR_0
TCORA_1									TMR_1
TCORB_0									TMR_0
TCORB_1									TMR_1
TCNT_0									TMR_0
TCNT_1									TMR_1
TCCR_0	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_0
TCCR_1	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_1
TSTR	—	—	CST5	CST4	CST3	CST2	CST1	CST0	TPU
TSYR	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_0
TMDR_0	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_0	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_0	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_0	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_0	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_0									
TGRA_0									
TGRB_0									
TGRC_0									
TGRD_0									
TCR_1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_1
TMDR_1	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_1	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_1	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_1	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_1									
TGRA_1									
TGRB_1									
TCR_2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_2
TMDR_2	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_2	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_2	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_2	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_2									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TGRA_2									TPU_2
TGRB_2									
TCR_3	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_3
TMDR_3	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_3	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_3	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_3	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_3	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_3									
TGRA_3									
TGRB_3									
TGRC_3									
TGRD_3									

Notes: 1. Parts of the bit functions differ in normal mode and the smart card interface.

2. When the same output trigger is specified for pulse output groups 2 and 3 by the PCR setting, the NDRH address is H'FFF7C. When different output triggers are specified, the NDRH addresses for pulse output groups 2 and 3 are H'FFF7E and H'FFF7C, respectively. Similarly, When the same output trigger is specified for pulse output groups 0 and 1 by the PCR setting, the NDRL address is H'FFF7D. When different output triggers are specified, the NDRL addresses for pulse output groups 0 and 1 are H'FFF7F and H'FFF7D, respectively.

21.3 Register States in Each Operating Mode

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
P1DDR	Initialized	—	—	—	—	Initialized	I/O ports
P2DDR	Initialized	—	—	—	—	Initialized	
P3DDR	Initialized	—	—	—	—	Initialized	
P6DDR	Initialized	—	—	—	—	Initialized	
PADDR	Initialized	—	—	—	—	Initialized	
PBDDR	Initialized	—	—	—	—	Initialized	
PDDDR	Initialized	—	—	—	—	Initialized	
PEDDR	Initialized	—	—	—	—	Initialized	
PFDDR	Initialized	—	—	—	—	Initialized	
P1ICR	Initialized	—	—	—	—	Initialized	
P2ICR	Initialized	—	—	—	—	Initialized	
P3ICR	Initialized	—	—	—	—	Initialized	
P5ICR	Initialized	—	—	—	—	Initialized	
P6ICR	Initialized	—	—	—	—	Initialized	
PAICR	Initialized	—	—	—	—	Initialized	
PBICR	Initialized	—	—	—	—	Initialized	
PDICR	Initialized	—	—	—	—	Initialized	
PEICR	Initialized	—	—	—	—	Initialized	
PFICR	Initialized	—	—	—	—	Initialized	
PORTH	—	—	—	—	—	—	
PORTI	—	—	—	—	—	—	
PHDR	Initialized	—	—	—	—	Initialized	
PIDR	Initialized	—	—	—	—	Initialized	
PHDDR	Initialized	—	—	—	—	Initialized	
PIDDR	Initialized	—	—	—	—	Initialized	
PHICR	Initialized	—	—	—	—	Initialized	
PIICR	Initialized	—	—	—	—	Initialized	
PDPCR	Initialized	—	—	—	—	Initialized	
PEPCR	Initialized	—	—	—	—	Initialized	
PFPCR	Initialized	—	—	—	—	Initialized	
PHPCR	Initialized	—	—	—	—	Initialized	
PIPCR	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
P2ODR	Initialized	—	—	—	—	Initialized	I/O ports
PFODR	Initialized	—	—	—	—	Initialized	
PFCR0	Initialized	—	—	—	—	Initialized	
PFCR1	Initialized	—	—	—	—	Initialized	
PFCR2	Initialized	—	—	—	—	Initialized	
PFCR4	Initialized	—	—	—	—	Initialized	
PFCR6	Initialized	—	—	—	—	Initialized	
PFCR7	Initialized	—	—	—	—	Initialized	
PFCR9	Initialized	—	—	—	—	Initialized	
PFCRB	Initialized	—	—	—	—	Initialized	
PFCRC	Initialized	—	—	—	—	Initialized	
SSIER	Initialized	—	—	—	—	Initialized	INTC
DSAR_0	Initialized	—	—	—	—	Initialized	DMAC_0
DDAR_0	Initialized	—	—	—	—	Initialized	
DOFR_0	Initialized	—	—	—	—	Initialized	
DTCR_0	Initialized	—	—	—	—	Initialized	
DBSR_0	Initialized	—	—	—	—	Initialized	
DMDR_0	Initialized	—	—	—	—	Initialized	
DACR_0	Initialized	—	—	—	—	Initialized	
DSAR_1	Initialized	—	—	—	—	Initialized	DMAC_1
DDAR_1	Initialized	—	—	—	—	Initialized	
DOFR_1	Initialized	—	—	—	—	Initialized	
DTCR_1	Initialized	—	—	—	—	Initialized	
DBSR_1	Initialized	—	—	—	—	Initialized	
DMDR_1	Initialized	—	—	—	—	Initialized	
DACR_1	Initialized	—	—	—	—	Initialized	
DSAR_2	Initialized	—	—	—	—	Initialized	DMAC_2
DDAR_2	Initialized	—	—	—	—	Initialized	
DOFR_2	Initialized	—	—	—	—	Initialized	
DTCR_2	Initialized	—	—	—	—	Initialized	
DBSR_2	Initialized	—	—	—	—	Initialized	
DMDR_2	Initialized	—	—	—	—	Initialized	
DACR_2	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
DSAR_3	Initialized	—	—	—	—	Initialized	DMAC_3
DDAR_3	Initialized	—	—	—	—	Initialized	
DOFR_3	Initialized	—	—	—	—	Initialized	
DTCR_3	Initialized	—	—	—	—	Initialized	
DBSR_3	Initialized	—	—	—	—	Initialized	
DMDR_3	Initialized	—	—	—	—	Initialized	
DACR_3	Initialized	—	—	—	—	Initialized	
DMRSR_0	Initialized	—	—	—	—	Initialized	DMAC_0
DMRSR_1	Initialized	—	—	—	—	Initialized	DMAC_1
DMRSR_2	Initialized	—	—	—	—	Initialized	DMAC_2
DMRSR_3	Initialized	—	—	—	—	Initialized	DMAC_3
IPRA	Initialized	—	—	—	—	Initialized	INTC
IPRB	Initialized	—	—	—	—	Initialized	
IPRC	Initialized	—	—	—	—	Initialized	
IPRE	Initialized	—	—	—	—	Initialized	
IPRF	Initialized	—	—	—	—	Initialized	
IPRG	Initialized	—	—	—	—	Initialized	
IPRH	Initialized	—	—	—	—	Initialized	
IPRI	Initialized	—	—	—	—	Initialized	
IPRK	Initialized	—	—	—	—	Initialized	
IPRL	Initialized	—	—	—	—	Initialized	
ISCRH	Initialized	—	—	—	—	Initialized	
ISCRL	Initialized	—	—	—	—	Initialized	
DTCVBR	Initialized	—	—	—	—	Initialized	BSC
ABWCR	Initialized	—	—	—	—	Initialized	
ASTCR	Initialized	—	—	—	—	Initialized	
WTCRA	Initialized	—	—	—	—	Initialized	
WTCRB	Initialized	—	—	—	—	Initialized	
RDNCR	Initialized	—	—	—	—	Initialized	
CSACR	Initialized	—	—	—	—	Initialized	
IDLCR	Initialized	—	—	—	—	Initialized	
BCR1	Initialized	—	—	—	—	Initialized	
BCR2	Initialized	—	—	—	—	Initialized	
ENDIANCR	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
SRAMCR	Initialized	—	—	—	—	Initialized	BSC
BROMCR	Initialized	—	—	—	—	Initialized	
MPXCR	Initialized	—	—	—	—	Initialized	
RAMER	Initialized	—	—	—	—	Initialized	
MDCR	Initialized	—	—	—	—	Initialized	SYSTEM
SYSCR	Initialized	—	—	—	—	Initialized	
SCKCR	Initialized	—	—	—	—	Initialized	
SBYCR	Initialized	—	—	—	—	Initialized	
MSTPCRA	Initialized	—	—	—	—	Initialized	
MSTPCRB	Initialized	—	—	—	—	Initialized	
MSTPCRC	Initialized	—	—	—	—	Initialized	
SEMR_2	Initialized	—	—	—	—	Initialized	SCI_2
SMR_4	Initialized	—	—	—	—	Initialized	SCI_4
BRR_4	Initialized	—	—	—	—	Initialized	
SCR_4	Initialized	—	—	—	—	Initialized	
TDR_4	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SSR_4	Initialized	Initialized	—	Initialized	Initialized	Initialized	
RDR_4	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SCMR_4	Initialized	—	—	—	—	Initialized	
FCCS	Initialized	—	—	—	—	Initialized	FLASH
FPCS	Initialized	—	—	—	—	Initialized	
FECS	Initialized	—	—	—	—	Initialized	
FKEY	Initialized	—	—	—	—	Initialized	
FMATS	Initialized	—	—	—	—	Initialized	
FTDAR	Initialized	—	—	—	—	Initialized	
TCR_2	Initialized	—	—	—	—	Initialized	TMR_2
TCR_3	Initialized	—	—	—	—	Initialized	TMR_3
TCSR_2	Initialized	—	—	—	—	Initialized	TMR_2
TCSR_3	Initialized	—	—	—	—	Initialized	TMR_3
TCORA_2	Initialized	—	—	—	—	Initialized	TMR_2
TCORA_3	Initialized	—	—	—	—	Initialized	TMR_3
TCORB_2	Initialized	—	—	—	—	Initialized	TMR_2
TCORB_3	Initialized	—	—	—	—	Initialized	TMR_3

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
TCNT_2	Initialized	—	—	—	—	Initialized	TMR_2
TCNT_3	Initialized	—	—	—	—	Initialized	TMR_3
TCCR_2	Initialized	—	—	—	—	Initialized	TMR_2
TCCR_3	Initialized	—	—	—	—	Initialized	TMR_3
TCR_4	Initialized	—	—	—	—	Initialized	TPU_4
TMDR_4	Initialized	—	—	—	—	Initialized	
TIOR_4	Initialized	—	—	—	—	Initialized	
TIER_4	Initialized	—	—	—	—	Initialized	
TSR_4	Initialized	—	—	—	—	Initialized	
TCNT_4	Initialized	—	—	—	—	Initialized	
TGRA_4	Initialized	—	—	—	—	Initialized	
TGRB_4	Initialized	—	—	—	—	Initialized	
TCR_5	Initialized	—	—	—	—	Initialized	TPU_5
TMDR_5	Initialized	—	—	—	—	Initialized	
TIOR_5	Initialized	—	—	—	—	Initialized	
TIER_5	Initialized	—	—	—	—	Initialized	
TSR_5	Initialized	—	—	—	—	Initialized	
TCNT_5	Initialized	—	—	—	—	Initialized	
TGRA_5	Initialized	—	—	—	—	Initialized	
TGRB_5	Initialized	—	—	—	—	Initialized	
DTCERA	Initialized	—	—	—	—	Initialized	INTC
DTCERB	Initialized	—	—	—	—	Initialized	
DTCERC	Initialized	—	—	—	—	Initialized	
DTCERD	Initialized	—	—	—	—	Initialized	
DTCERE	Initialized	—	—	—	—	Initialized	
DTCCR	Initialized	—	—	—	—	Initialized	
INTCR	Initialized	—	—	—	—	Initialized	
CPUPCR	Initialized	—	—	—	—	Initialized	
IER	Initialized	—	—	—	—	Initialized	
ISR	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
PORT1	—	—	—	—	—	—	I/O ports
PORT2	—	—	—	—	—	—	
PORT3	—	—	—	—	—	—	
PORT5	—	—	—	—	—	—	
PORT6	—	—	—	—	—	—	
PORTA	—	—	—	—	—	—	
PORTB	—	—	—	—	—	—	
PORTD	—	—	—	—	—	—	
PORTE	—	—	—	—	—	—	
PORTF	—	—	—	—	—	—	
P1DR	Initialized	—	—	—	—	Initialized	
P2DR	Initialized	—	—	—	—	Initialized	
P3DR	Initialized	—	—	—	—	Initialized	
P6DR	Initialized	—	—	—	—	Initialized	
PADR	Initialized	—	—	—	—	Initialized	
PBDR	Initialized	—	—	—	—	Initialized	
PDDR	Initialized	—	—	—	—	Initialized	
PEDR	Initialized	—	—	—	—	Initialized	
PFDR	Initialized	—	—	—	—	Initialized	
SMR_2	Initialized	—	—	—	—	Initialized	SCI_2
BRR_2	Initialized	—	—	—	—	Initialized	
SCR_2	Initialized	—	—	—	—	Initialized	
TDR_2	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SSR_2	Initialized	Initialized	—	Initialized	Initialized	Initialized	
RDR_2	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SCMR_2	Initialized	—	—	—	—	Initialized	
DADR0	Initialized	—	—	—	—	Initialized	D/A
DADR1	Initialized	—	—	—	—	Initialized	
DACR01	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
PCR	Initialized	—	—	—	—	Initialized	PPG
PMR	Initialized	—	—	—	—	Initialized	
NDERH	Initialized	—	—	—	—	Initialized	
NDERL	Initialized	—	—	—	—	Initialized	
PODRH	Initialized	—	—	—	—	Initialized	
PODRL	Initialized	—	—	—	—	Initialized	
NDRH	Initialized	—	—	—	—	Initialized	
NDRL	Initialized	—	—	—	—	Initialized	
SMR_0	Initialized	—	—	—	—	Initialized	SCI_0
BRR_0	Initialized	—	—	—	—	Initialized	
SCR_0	Initialized	—	—	—	—	Initialized	
TDR_0	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SSR_0	Initialized	Initialized	—	Initialized	Initialized	Initialized	
RDR_0	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SCMR_0	Initialized	—	—	—	—	Initialized	
SMR_1	Initialized	—	—	—	—	Initialized	SCI_1
BRR_1	Initialized	—	—	—	—	Initialized	
SCR_1	Initialized	—	—	—	—	Initialized	
TDR_1	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SSR_1	Initialized	Initialized	—	Initialized	Initialized	Initialized	
RDR_1	Initialized	Initialized	—	Initialized	Initialized	Initialized	
SCMR_1	Initialized	—	—	—	—	Initialized	
ADDRA	Initialized	—	—	—	—	Initialized	A/D
ADDRB	Initialized	—	—	—	—	Initialized	
ADDRC	Initialized	—	—	—	—	Initialized	
ADDRD	Initialized	—	—	—	—	Initialized	
ADDRE	Initialized	—	—	—	—	Initialized	
ADDRF	Initialized	—	—	—	—	Initialized	
ADDRG	Initialized	—	—	—	—	Initialized	
ADDRH	Initialized	—	—	—	—	Initialized	
ADCSR	Initialized	—	—	—	—	Initialized	
ADCR	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
TCSR	Initialized	—	—	—	—	Initialized	WDT
TCNT	Initialized	—	—	—	—	Initialized	
RSTCSR	Initialized	—	—	—	—	Initialized	
TCR_0	Initialized	—	—	—	—	Initialized	TMR_0
TCR_1	Initialized	—	—	—	—	Initialized	TMR_1
TCSR_0	Initialized	—	—	—	—	Initialized	TMR_0
TCSR_1	Initialized	—	—	—	—	Initialized	TMR_1
TCORA_0	Initialized	—	—	—	—	Initialized	TMR_0
TCORA_1	Initialized	—	—	—	—	Initialized	TMR_1
TCORB_0	Initialized	—	—	—	—	Initialized	TMR_0
TCORB_1	Initialized	—	—	—	—	Initialized	TMR_1
TCNT_0	Initialized	—	—	—	—	Initialized	TMR_0
TCNT_1	Initialized	—	—	—	—	Initialized	TMR_1
TCCR_0	Initialized	—	—	—	—	Initialized	TMR_0
TCCR_1	Initialized	—	—	—	—	Initialized	TMR_1
TSTR	Initialized	—	—	—	—	Initialized	TPU
TSYR	Initialized	—	—	—	—	Initialized	
TCR_0	Initialized	—	—	—	—	Initialized	TPU_0
TMDR_0	Initialized	—	—	—	—	Initialized	
TIORH_0	Initialized	—	—	—	—	Initialized	
TIORL_0	Initialized	—	—	—	—	Initialized	
TIER_0	Initialized	—	—	—	—	Initialized	
TSR_0	Initialized	—	—	—	—	Initialized	
TCNT_0	Initialized	—	—	—	—	Initialized	
TGRA_0	Initialized	—	—	—	—	Initialized	
TGRB_0	Initialized	—	—	—	—	Initialized	
TGRC_0	Initialized	—	—	—	—	Initialized	
TGRD_0	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	Module Stop state	Sleep	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
TCR_1	Initialized	—	—	—	—	Initialized	TPU_1
TMDR_1	Initialized	—	—	—	—	Initialized	
TIOR_1	Initialized	—	—	—	—	Initialized	
TIER_1	Initialized	—	—	—	—	Initialized	
TSR_1	Initialized	—	—	—	—	Initialized	
TCNT_1	Initialized	—	—	—	—	Initialized	
TGRA_1	Initialized	—	—	—	—	Initialized	
TGRB_1	Initialized	—	—	—	—	Initialized	
TCR_2	Initialized	—	—	—	—	Initialized	TPU_2
TMDR_2	Initialized	—	—	—	—	Initialized	
TIOR_2	Initialized	—	—	—	—	Initialized	
TIER_2	Initialized	—	—	—	—	Initialized	
TSR_2	Initialized	—	—	—	—	Initialized	
TCNT_2	Initialized	—	—	—	—	Initialized	
TGRA_2	Initialized	—	—	—	—	Initialized	
TGRB_2	Initialized	—	—	—	—	Initialized	
TCR_3	Initialized	—	—	—	—	Initialized	TPU_3
TMDR_3	Initialized	—	—	—	—	Initialized	
TIORH_3	Initialized	—	—	—	—	Initialized	
TIORL_3	Initialized	—	—	—	—	Initialized	
TIER_3	Initialized	—	—	—	—	Initialized	
TSR_3	Initialized	—	—	—	—	Initialized	
TCNT_3	Initialized	—	—	—	—	Initialized	
TGRA_3	Initialized	—	—	—	—	Initialized	
TGRB_3	Initialized	—	—	—	—	Initialized	
TGRC_3	Initialized	—	—	—	—	Initialized	
TGRD_3	Initialized	—	—	—	—	Initialized	

Section 22 Electrical Characteristics

22.1 Absolute Maximum Ratings

Table 22.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage	V_{CC} PLV_{CC}	–0.3 to +4.6	V
Input voltage (except for port 5)	V_{in}	–0.3 to V_{CC} +0.3	V
Input voltage (port 5)	V_{in}	–0.3 to AV_{CC} +0.3	V
Reference power supply voltage	V_{ref}	–0.3 to AV_{CC} +0.3	V
Analog power supply voltage	AV_{CC}	–0.3 to +4.6	V
Analog input voltage	V_{AN}	–0.3 to AV_{CC} +0.3	V
Operating temperature	T_{opr}	Regular specifications: –20 to +75*	°C
		Wide-range specifications: –40 to +85*	
Storage temperature	T_{stg}	–55 to +125	°C

Caution: Permanent damage to the LSI may result if absolute maximum ratings are exceeded.

Note: * The operating temperature range during programming/erasing of the flash memory is 0°C to +75°C for regular specifications and 0°C to +85°C for wide-range specifications.

22.2 DC Characteristics

Table 22.2 DC Characteristics (1)

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
$V_{SS} = AV_{SS}$ = 0 V*[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Schmitt trigger input voltage	$\overline{IRQ}$ input pin, TPU input pin, TMR input pin, port 2, port 3	VT^-	$V_{CC} \times 0.2$	—	—	V	
		VT^+	—	—	$V_{CC} \times 0.7$	V	
		$VT^+ - VT^-$	$V_{CC} \times 0.06$	—	—	V	
	Port 5*[2]	VT^-	$AV_{CC} \times 0.2$	—	—	V	
		VT^+	—	—	$AV_{CC} \times 0.7$	V	
		$VT^+ - VT^-$	$AV_{CC} \times 0.06$	—	—	V	
Input high voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$, $\overline{STBY}$, EMLE, NMI	V_{IH}	$V_{CC} \times 0.9$	—	$V_{CC} + 0.3$	V	
	EXTAL		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$	V	
	Other input pins		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$	V	
	Port 5		$AV_{CC} \times 0.7$	—	$AV_{CC} + 0.3$	V	
Input low voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$, $\overline{STBY}$, EMLE	V_{IL}	–0.3	—	$V_{CC} \times 0.1$	V	
	EXTAL, NMI		–0.3	—	$V_{CC} \times 0.2$	V	
	Other pins		–0.3	—	$V_{CC} \times 0.2$	V	
Output high voltage	All output pins	V_{OH}	$V_{CC} - 0.5$	—	—	V	I_{OH} = –200 μA
			$V_{CC} - 1.0$	—	—		I_{OH} = –1 mA
Output low voltage	All output pins	V_{OL}	—	—	0.4	V	I_{OL} = 1.6 mA
	Port 3		—	—	1.0		I_{OL} = 10 mA
Input leakage current	$\overline{RES}$	$\|I_{in}\|$	—	—	10.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V
	MD, $\overline{STBY}$, EMLE, NMI		—	—	1.0		
	Port 5		—	—	1.0		V_{in} = 0.5 to AV_{CC} – 0.5 V

Table 22.2 DC Characteristics (2)

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V*[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Three-state leakage current (off state)	Ports 1 to 3, 6, A, B, D to F, H, I	$\lvert I_{TSI} \rvert$	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input pull-up MOS current	Ports D to F, H, I	$-I_p$	10	—	300	µA	V_{CC} = 3.0 to 3.6 V V_{in} = 0 V
Input capacitance	All input pins	C_{in}	—	—	15	pF	V_{in} = 0 V f = 1 MHz T_a = 25°C
Current consumption *[3]	Normal operation	I_{CC}*[5]	—	35 (3.3 V)	45	mA	f = 35 MHz
	Sleep mode		—	30 (3.3 V)	37		
	Standby mode*[4]		—	0.15	0.5		$T_a \leq$ 50°C
			—	—	3.0		50°C < T_a
	All-module-clock-stop mode*[6]		—	15	25		
Analog power supply current	During A/D and D/A conversion	AI_{CC}	—	1.0 (3.0 V)	2.0	mA	
	Standby for A/D and D/A conversion		—	1.0	20	µA	
Reference power supply current	During A/D and D/A conversion	AI_{CC}	—	1.5 (3.0 V)	3.0	mA	
	Standby for A/D and D/A conversion		—	0.4	5.0	µA	
RAM standby voltage		V_{RAM}	2.5	—	—	V	
Vcc start voltage*[7]		$V_{CCSTART}$	—	—	0.8	V	
Vcc rising gradient*[7]		SV_{CC}	—	—	20	ms/V	

Notes: 1. When the A/D and D/A converters are not used, the AV_{CC}, V_{ref}, and AV_{SS} pins should not be open. Connect the AV_{CC} and V_{ref} pins to V_{CC}, and the AV_{SS} pin to V_{SS}.

2. The case where port 5 is used as $\overline{IRQ0}$ to $\overline{IRQ7}$.
3. Current consumption values are for V_{IH}min = V_{CC} – 0.5 V and V_{IL}max = 0.5 V with all output pins unloaded and all input pull-up MOSs in the off state.
4. The values are for $V_{RAM} \leq V_{CC} < 3.0$ V, V_{IH}min = $V_{CC} \times 0.9$, and V_{IL}max = 0.3 V.
5. I_{CC} depends on V_{CC} and f as follows:
   I_{CC}max = 5.0 (mA) + 0.32 (mA/(MHz × V)) × V_{CC} × f (normal operation)
   I_{CC}max = 5.0 (mA) + 0.24 (mA/(MHz × V)) × V_{CC} × f (sleep mode)
6. The values are for reference.
7. This can be applied when the $\overline{RES}$ pin is held low at power-on.

Table 22.3 Permissible Output Currents

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
$V_{SS} = AV_{SS}$ = 0 V*,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min.	Typ.	Max.	Unit
Permissible output low current (per pin)	Output pins except port 3	I_{OL}	—	—	2.0	mA
Permissible output low current (per pin)	Port 3	I_{OL}	—	—	10	mA
Permissible output low current (total)	Total of all output pins	ΣI_{OL}	—	—	80	mA
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2.0	mA
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	40	mA

Caution: To protect the LSI's reliability, do not exceed the output current values in table 22.3.

Note: * When the A/D and D/A converters are not used, the AV_{CC}, V_{ref}, and AV_{SS} pins should not be open. Connect the AV_{CC} and V_{ref} pins to V_{CC}, and the AV_{SS} pin to V_{SS}.

22.3 AC Characteristics

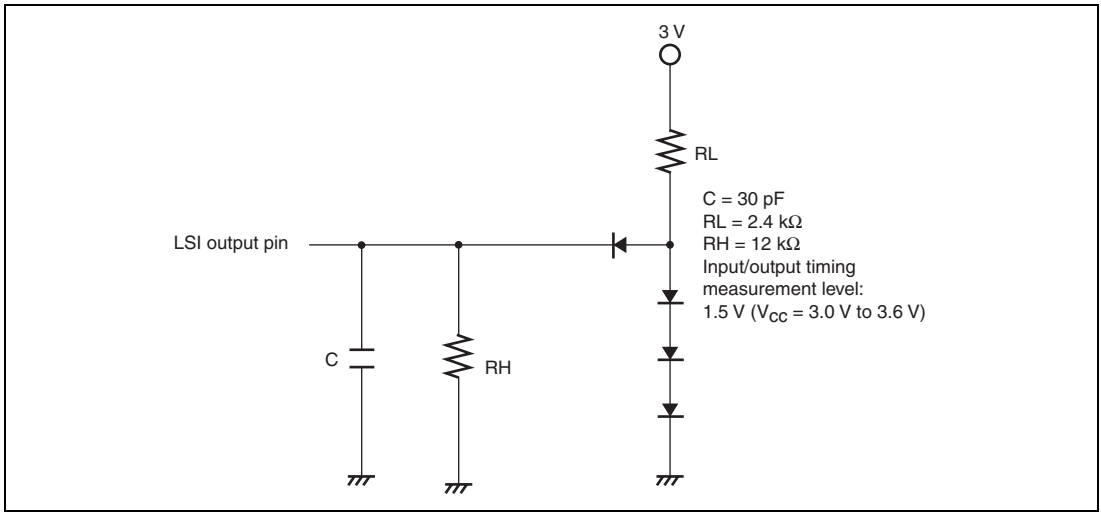


Figure 22.1 Output Load Circuit

22.3.1 Clock Timing

Table 22.4 Clock Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
$V_{SS} = AV_{SS}$ = 0 V, Iϕ = 8 MHz to 35 MHz, Bϕ = 8 MHz to 35 MHz,
Pϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit.	Test Conditions
Clock cycle time	t_{cyc}	28.0	125	ns	Figure 22.2
Clock high pulse width	t_{CH}	5	—	ns	
Clock low pulse width	t_{CL}	5	—	ns	
Clock rising time	t_{Cr}	—	5	ns	
Clock falling time	t_{Cf}	—	5	ns	
Oscillation settling time after reset (crystal)	t_{OSC1}	10	—	ms	Figure 22.4
Oscillation settling time after leaving software standby mode (crystal)	t_{OSC2}	10	—	ms	Figure 22.3
External clock output delay settling time	t_{DEXT}	1	—	ms	Figure 22.4
External clock input low pulse width	T_{EXL}	27.7	—	ns	Figure 22.5
External clock input high pulse width	T_{EXH}	27.7	—	ns	
External clock rising time	T_{EXr}	—	5	ns	
External clock falling time	T_{EXf}	—	5	ns	

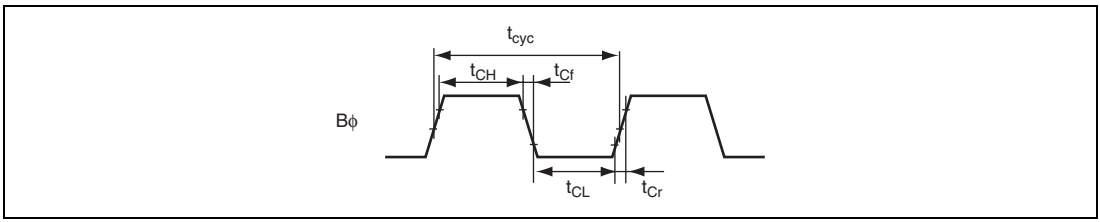


Figure 22.2 External Bus Clock Timing

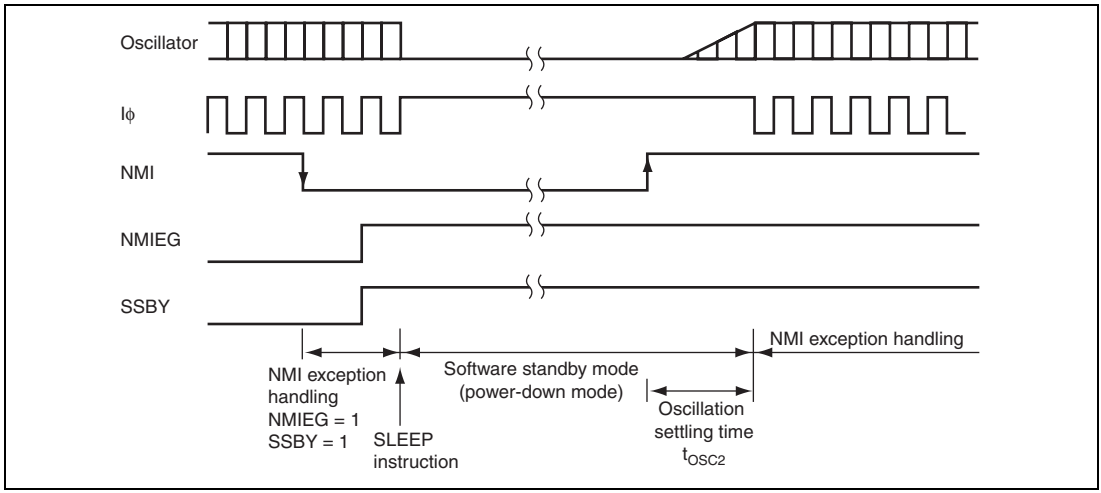


Figure 22.3 Oscillation Settling Timing after Software Standby Mode

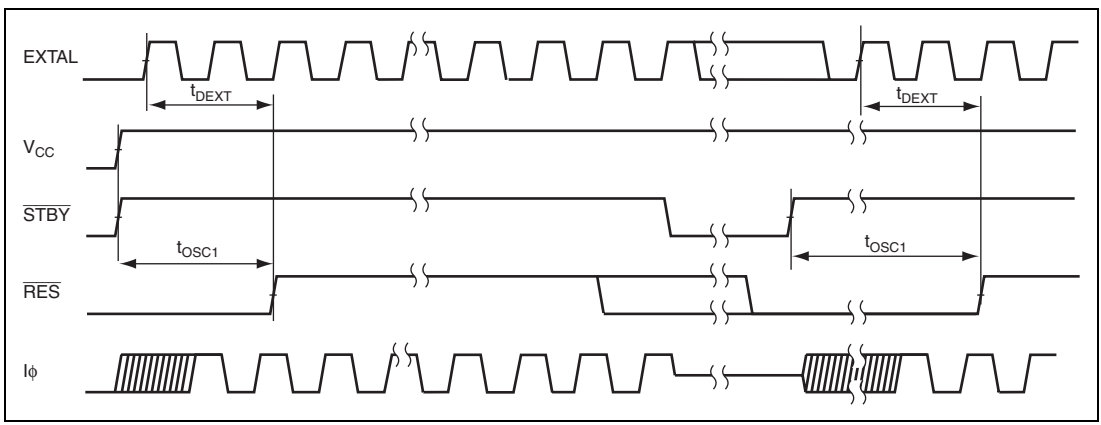


Figure 22.4 Oscillation Settling Timing

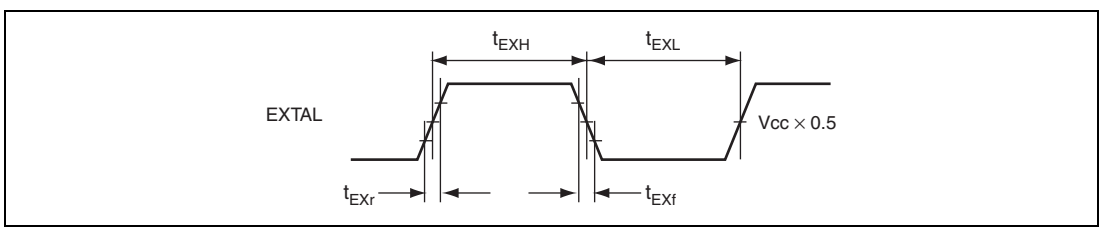


Figure 22.5 External Input Clock Timing

22.3.2 Control Signal Timing

Table 22.5 Control Signal Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
$V_{SS} = AV_{SS}$ = 0 V, Iϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{RES}$ setup time	t_{RESS}	200	—	ns	Figure 22.6
$\overline{RES}$ pulse width	t_{RESW}	20	—	t_{cyc}	
NMI setup time	t_{NMIS}	150	—	ns	Figure 22.7
NMI hold time	t_{NMIH}	10	—	ns	
NMI pulse width (after leaving software standby mode)	t_{NMIW}	200	—	ns	
$\overline{IRQ}$ setup time	t_{IRQS}	150	—	ns	
$\overline{IRQ}$ hold time	t_{IRQH}	10	—	ns	
$\overline{IRQ}$ pulse width (after leaving software standby mode)	t_{IRQW}	200	—	ns	

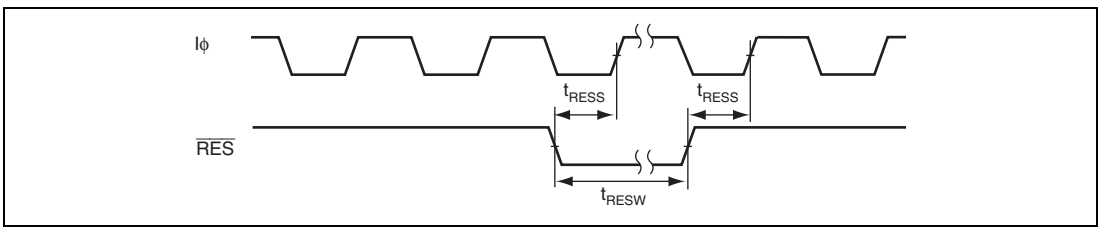
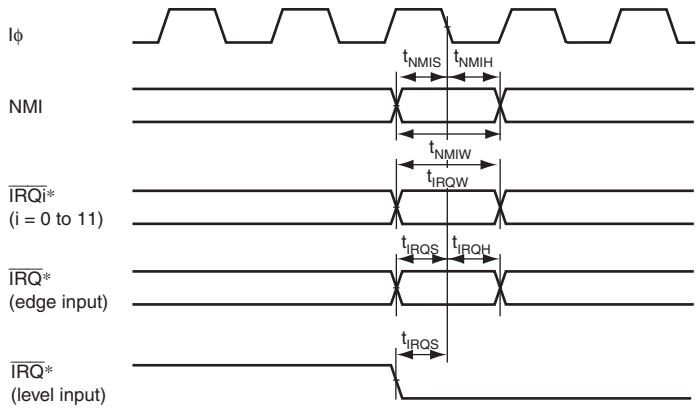


Figure 22.6 Reset Input Timing



Note: * SSIER must be set to cancel software standby mode.

Figure 22.7 Interrupt Input Timing

22.3.3 Bus Timing

Table 22.6 Bus Timing (1)

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Bϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
Address delay time	t_{AD}	—	15	ns	Figures 22.8 to 22.20
Address setup time 1	t_{AS1}	0.5 × t_{cyc} – 8	—	ns	
Address setup time 2	t_{AS2}	1.0 × t_{cyc} – 8	—	ns	
Address setup time 3	t_{AS3}	1.5 × t_{cyc} – 8	—	ns	
Address setup time 4	t_{AS4}	2.0 × t_{cyc} – 8	—	ns	
Address hold time 1	t_{AH1}	0.5 × t_{cyc} – 8	—	ns	
Address hold time 2	t_{AH2}	1.0 × t_{cyc} – 8	—	ns	
Address hold time 3	t_{AH3}	1.5 × t_{cyc} – 8	—	ns	
$\overline{CS}$ delay time 1	t_{CSD1}	—	15	ns	
$\overline{AS}$ delay time	t_{ASD}	—	15	ns	

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{RD}$ delay time 1	t_{RSD1}	—	15	ns	
$\overline{RD}$ delay time 2	t_{RSD2}	—	15	ns	
Read data setup time 1	t_{RDS1}	15	—	ns	Figures 22.8 to 22.20
Read data setup time 2	t_{RDS2}	15	—	ns	
Read data hold time 1	t_{RDH1}	0	—	ns	
Read data hold time 2	t_{RDH2}	0	—	ns	
Read data access time 2	t_{AC2}	—	$1.5 \times t_{cyc} - 20$	ns	
Read data access time 4	t_{AC4}	—	$2.5 \times t_{cyc} - 20$	ns	
Read data access time 5	t_{AC5}	—	$1.0 \times t_{cyc} - 20$	ns	
Read data access time 6	t_{AC6}	—	$2.0 \times t_{cyc} - 20$	ns	
Read data access time (from address) 1	t_{AA1}	—	$1.0 \times t_{cyc} - 20$	ns	
Read data access time (from address) 2	t_{AA2}	—	$1.5 \times t_{cyc} - 20$	ns	
Read data access time (from address) 3	t_{AA3}	—	$2.0 \times t_{cyc} - 20$	ns	
Read data access time (from address) 4	t_{AA4}		$2.5 \times t_{cyc} - 20$	ns	
Read data access time (from address) 5	t_{AA5}	—	$3.0 \times t_{cyc} - 20$	ns	

Table 22.6 Bus Timing (2)

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Bϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{WR}$ delay time 1	t_{WRD1}	—	15	ns	Figures 22.8 to 22.20
$\overline{WR}$ delay time 2	t_{WRD2}	—	15	ns	
$\overline{WR}$ pulse width 1	t_{WSW1}	$1.0 \times t_{cyc} - 13$	—	ns	
$\overline{WR}$ pulse width 2	t_{WSW2}	$1.5 \times t_{cyc} - 13$	—	ns	
Write data delay time	t_{WDD}	—	20	ns	
Write data setup time 1	t_{WDS1}	$0.5 \times t_{cyc} - 13$	—	ns	
Write data setup time 2	t_{WDS2}	$1.0 \times t_{cyc} - 13$	—	ns	
Write data setup time 3	t_{WDS3}	$1.5 \times t_{cyc} - 13$	—	ns	
Write data hold time 1	t_{WDH1}	$0.5 \times t_{cyc} - 8$	—	ns	
Write data hold time 3	t_{WDH3}	$1.5 \times t_{cyc} - 8$	—	ns	
Byte control delay time	t_{UBD}	—	15	ns	Figures 22.13, 22.14
Byte control pulse width 1	t_{UBW1}	—	$1.0 \times t_{cyc} - 15$	ns	Figure 22.13
Byte control pulse width 2	t_{UBW2}	—	$2.0 \times t_{cyc} - 15$	ns	Figure 22.14
Multiplexed address delay time 1	t_{MAD1}	—	15	ns	Figures 22.17, 22.18
Multiplexed address hold time	t_{MAH}	$1.0 \times t_{cyc} - 15$	—	ns	
Multiplexed address setup time 1	t_{MAS1}	$0.5 \times t_{cyc} - 15$	—	ns	
Multiplexed address setup time 2	t_{MAS2}	$1.5 \times t_{cyc} - 15$	—	ns	
Address hold delay time	t_{AHD}	—	15	ns	
Address hold pulse width 1	t_{AHW1}	$1.0 \times t_{cyc} - 15$	—	ns	
Address hold pulse width 2	t_{AHW2}	$2.0 \times t_{cyc} - 15$	—	ns	
$\overline{WAIT}$ setup time	t_{WTS}	15	—	ns	Figures 22.10, 22.18
$\overline{WAIT}$ hold time	t_{WTH}	5.0	—	ns	
$\overline{BREQ}$ setup time	t_{BREQS}	20	—	ns	Figure 22.19
$\overline{BACK}$ delay time	t_{BACD}	—	15	ns	
Bus floating time	t_{BZD}	—	30	ns	
$\overline{BREQO}$ delay time	t_{BRQOD}	—	15	ns	Figure 22.20
$\overline{BS}$ delay time	T_{BSD}	1.0	15	ns	Figures 22.8, 22.9, 22.11 to 22.14
RD/$\overline{WR}$ delay time	T_{RWD}	—	15	ns	

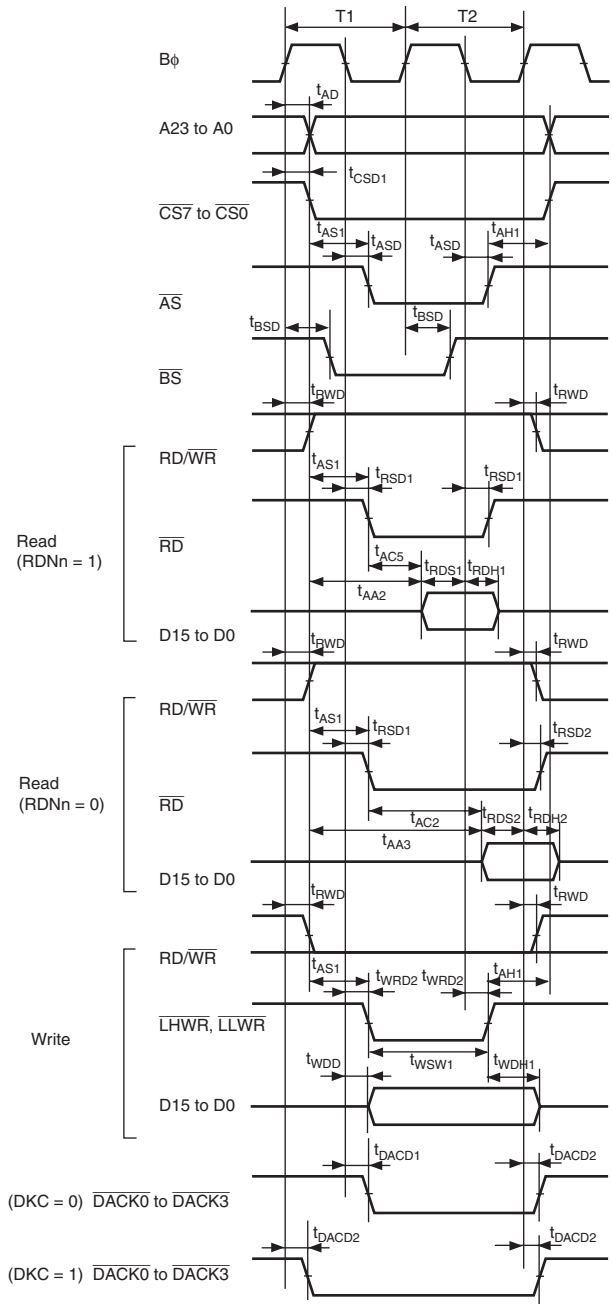


Figure 22.8 Basic Bus Timing: 2-State Access

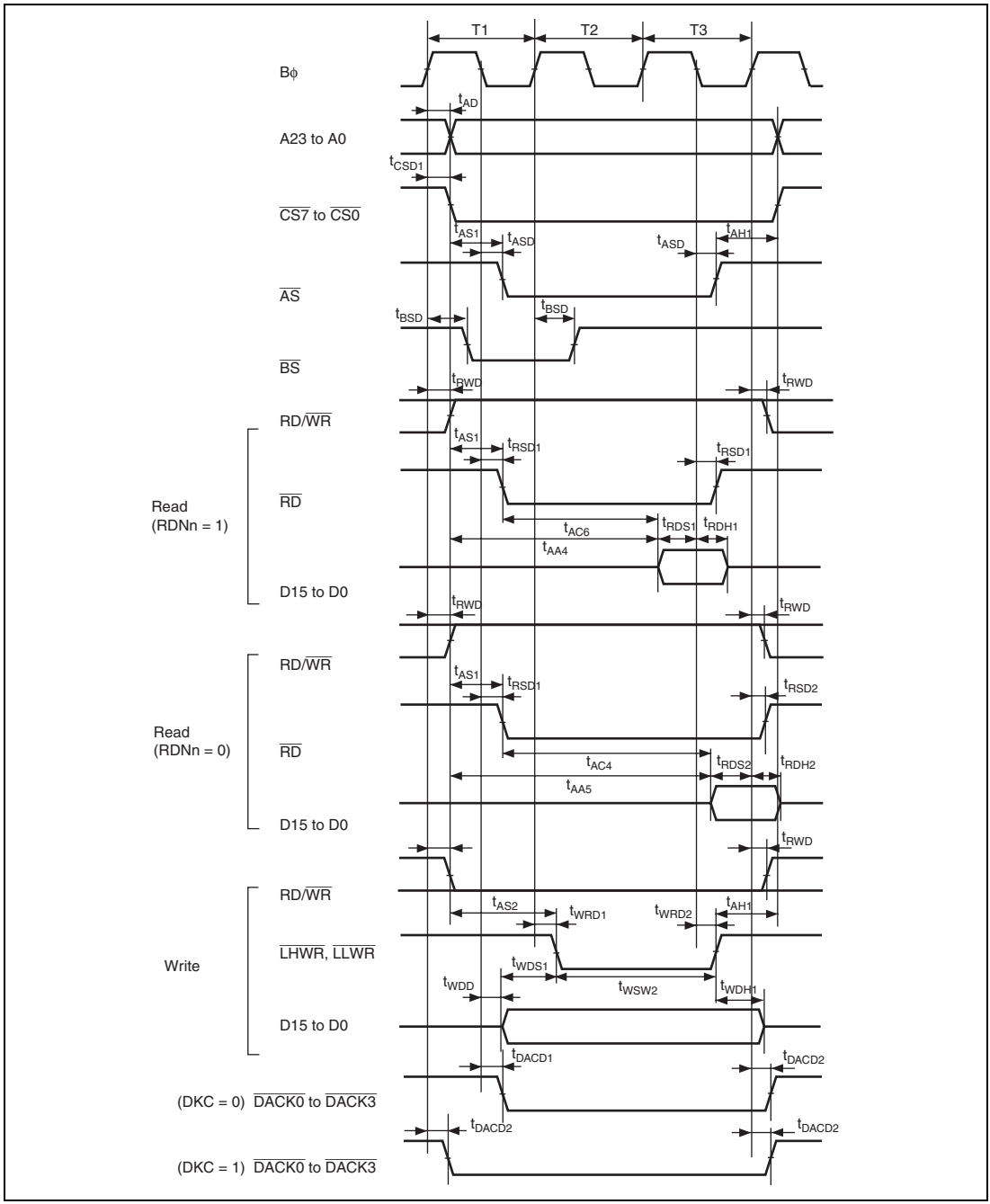


Figure 22.9 Basic Bus Timing: 3-State Access

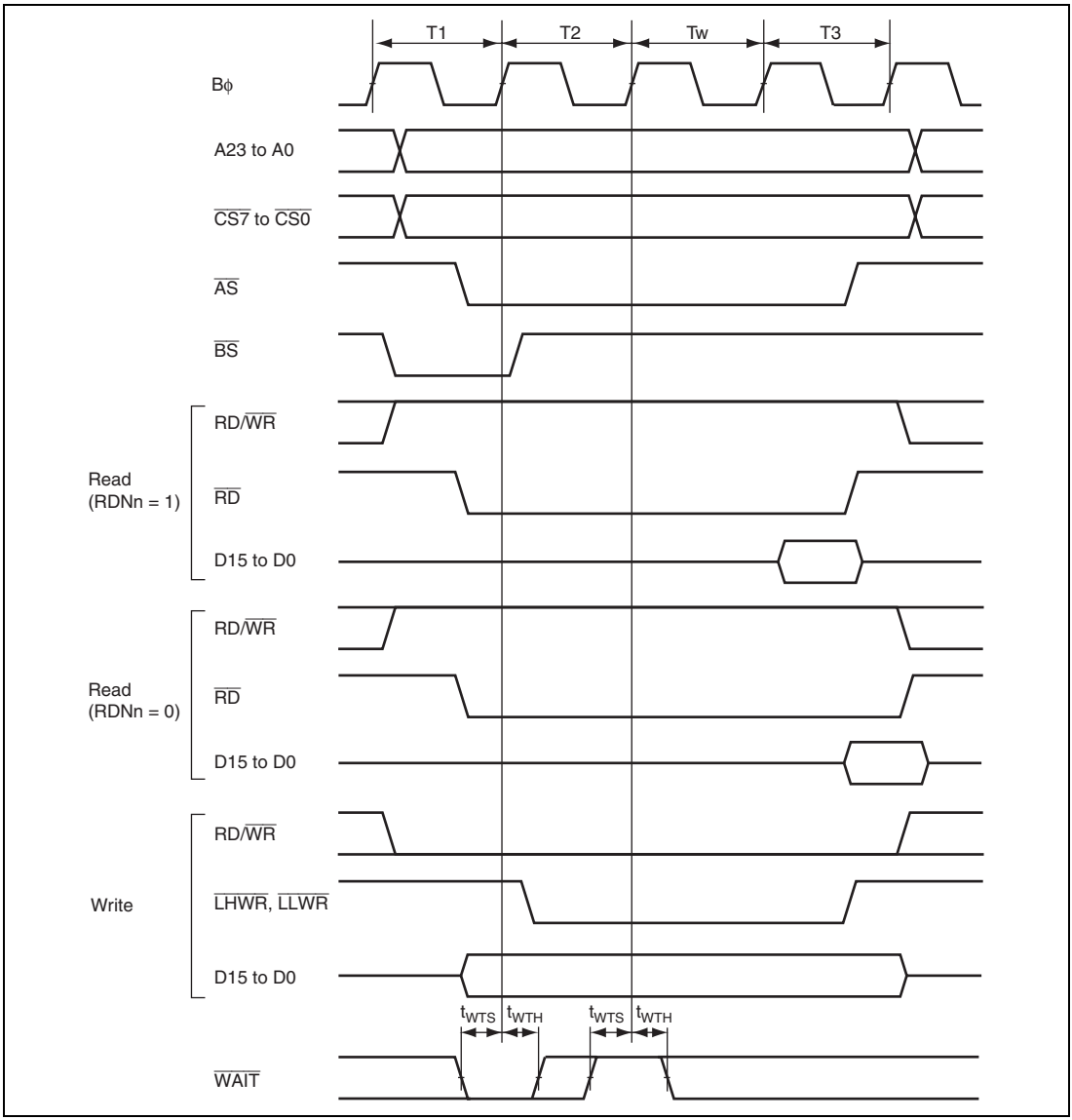


Figure 22.10 Basic Bus Timing: Three-State Access, One Wait

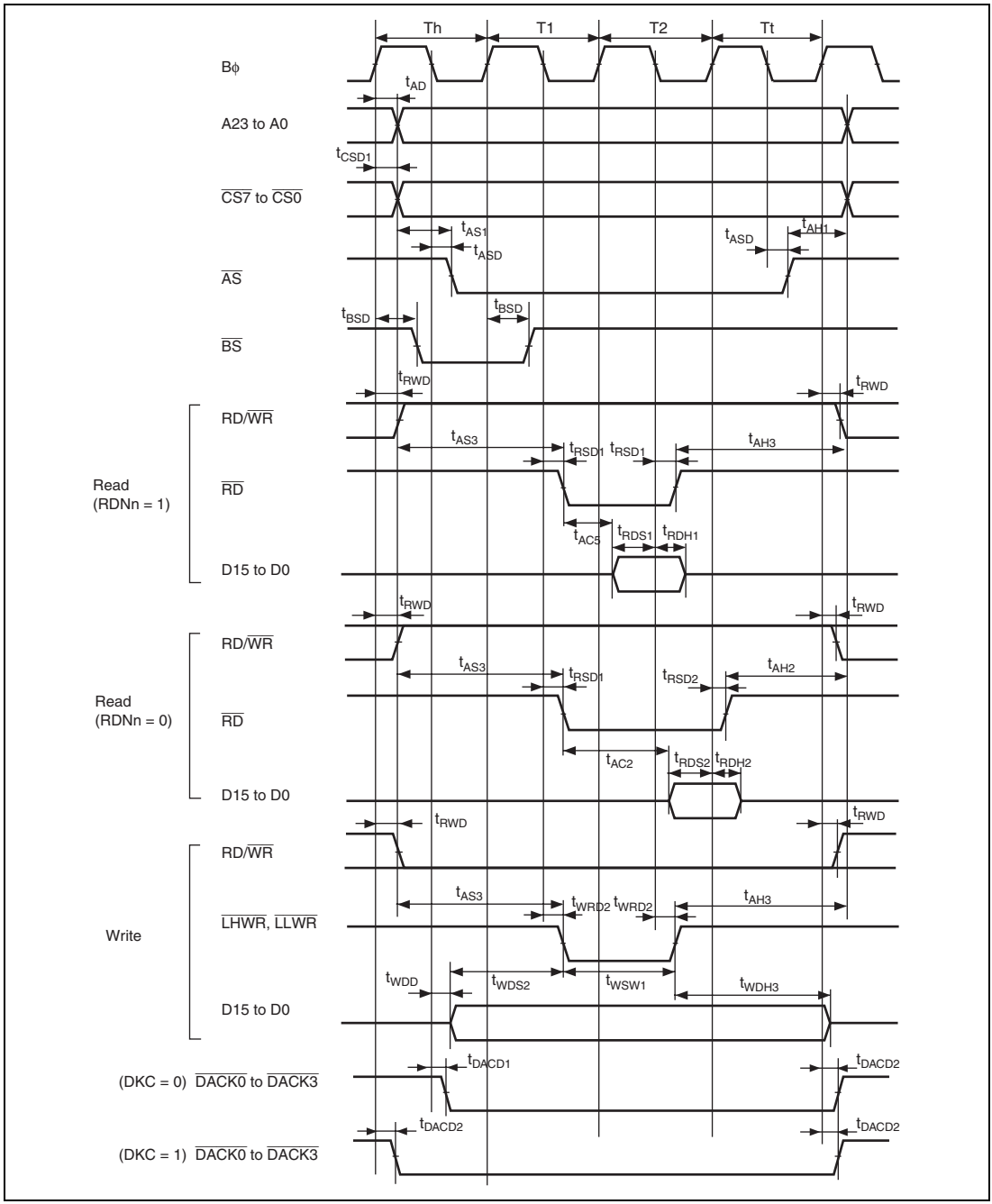


Figure 22.11 Basic Bus Timing: 2-State Access ($\overline{CS}$ Assertion Period Extended)

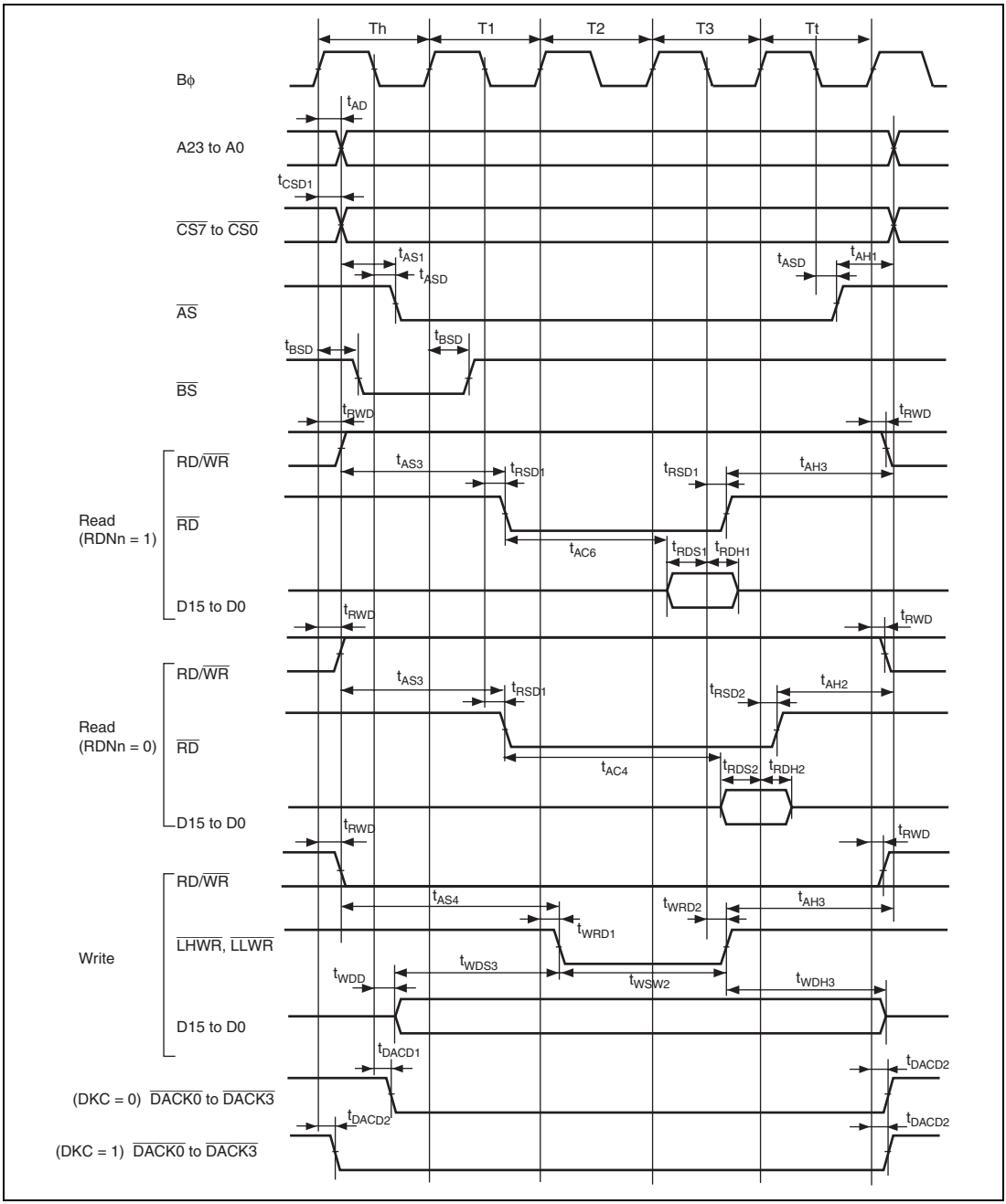


Figure 22.12 Basic Bus Timing: 3-State Access ($\overline{CS}$ Assertion Period Extended)

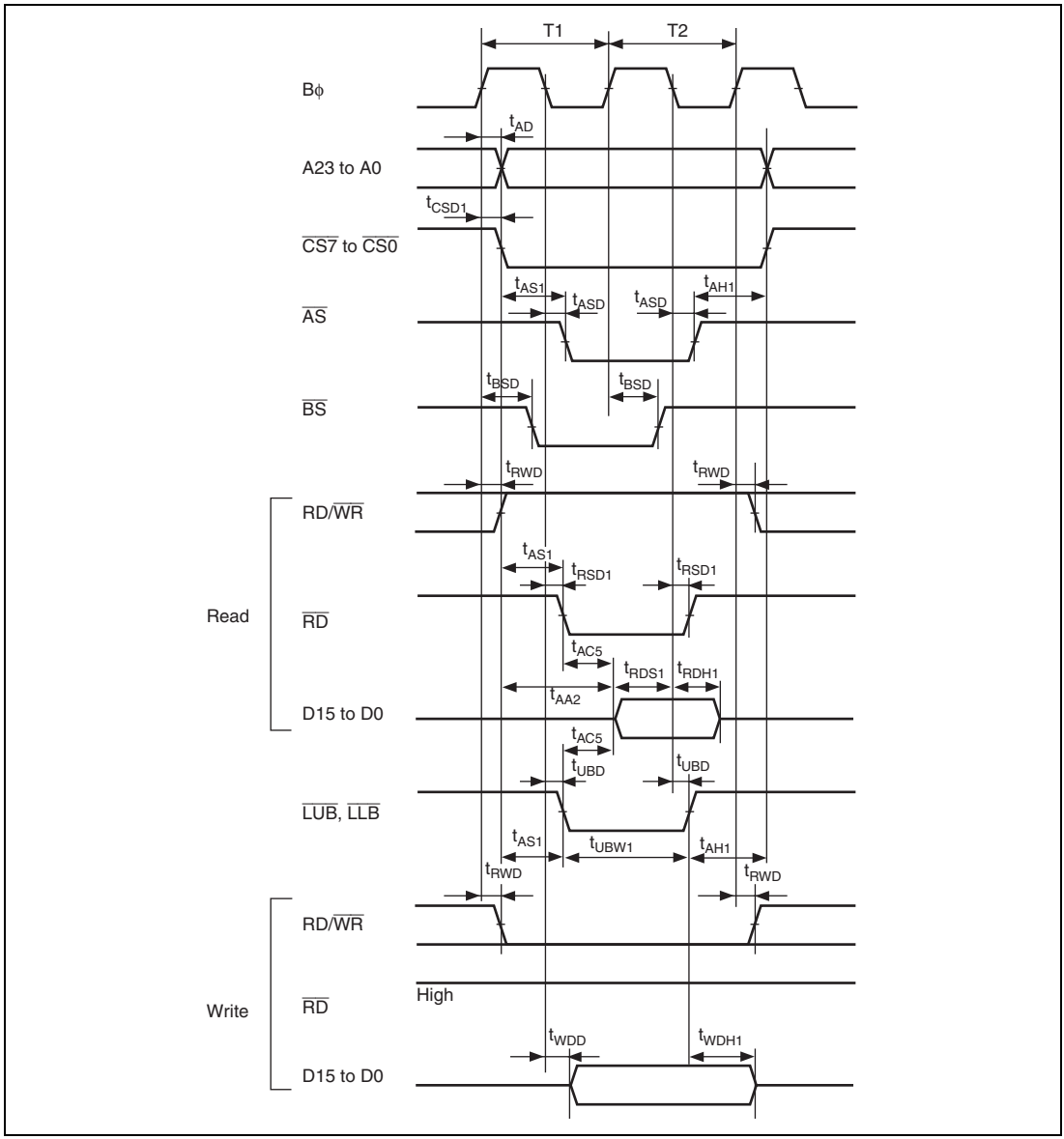


Figure 22.13 Byte Control SRAM: 2-State Read/Write Access

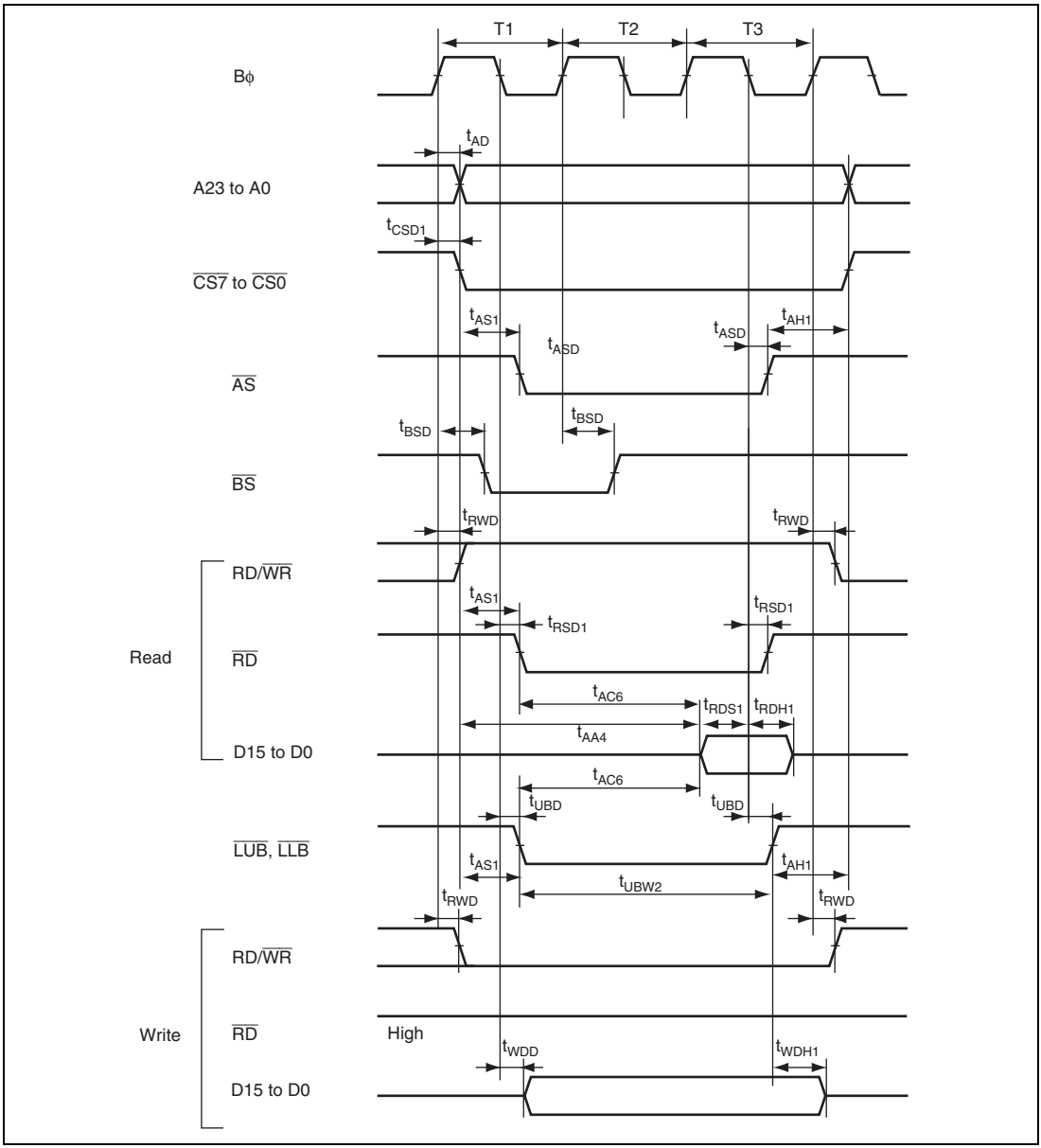


Figure 22.14 Byte Control SRAM: 3-State Read/Write Access

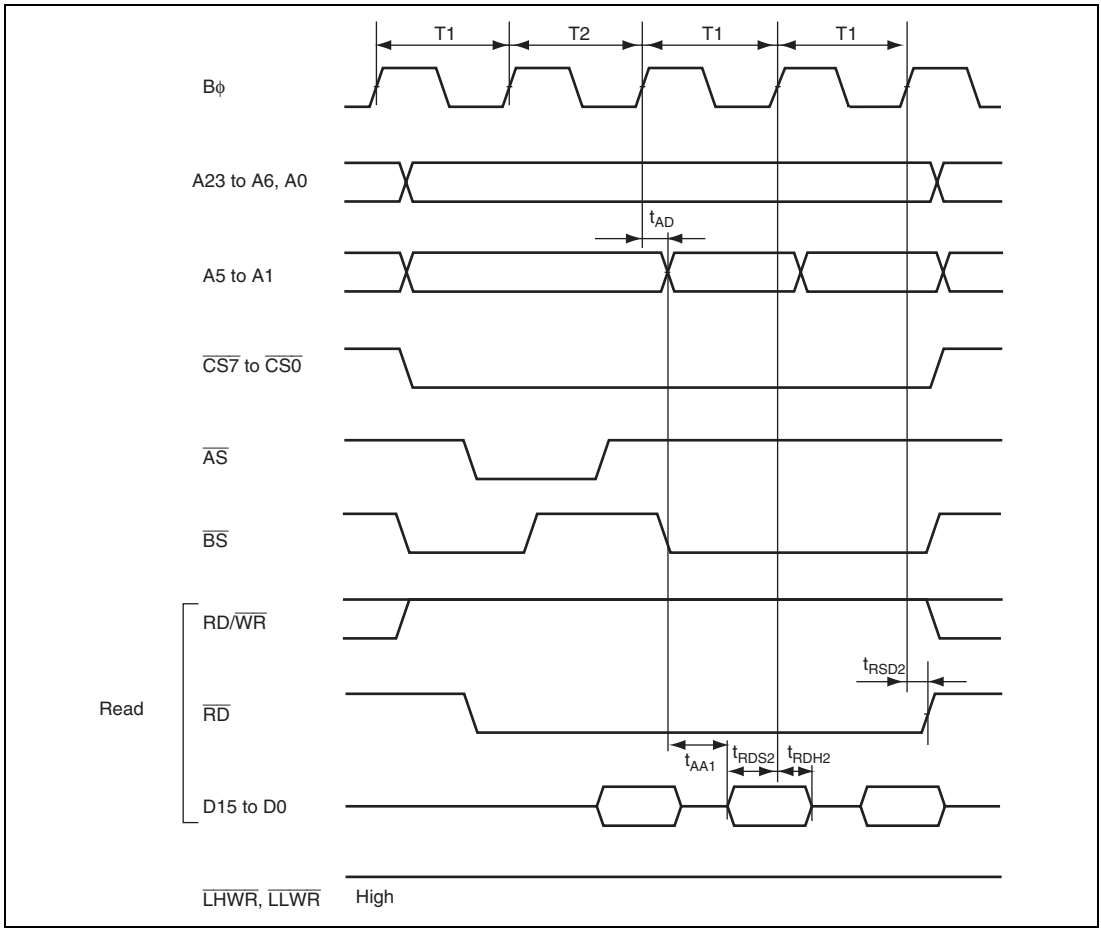


Figure 22.15 Burst ROM Access Timing: 1-State Burst Access

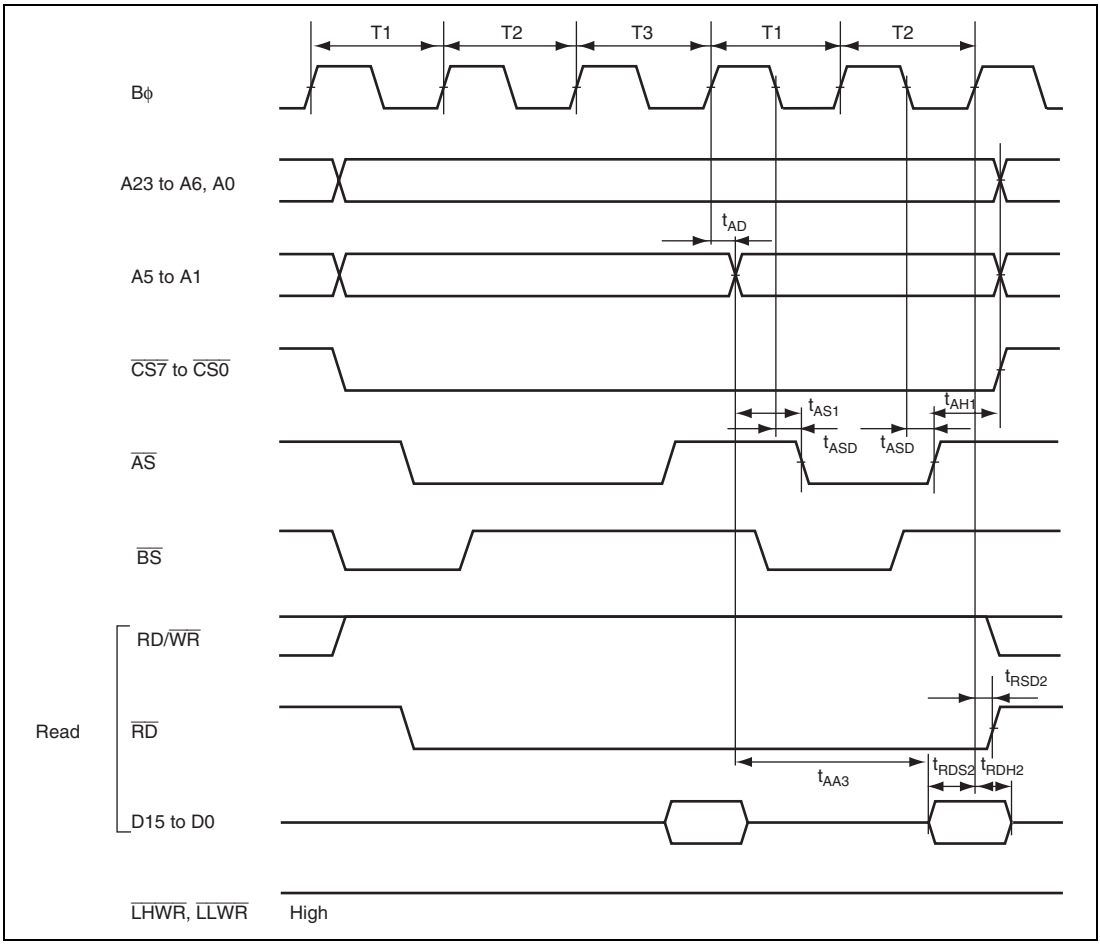


Figure 22.16 Burst ROM Access Timing: 2-State Burst Access

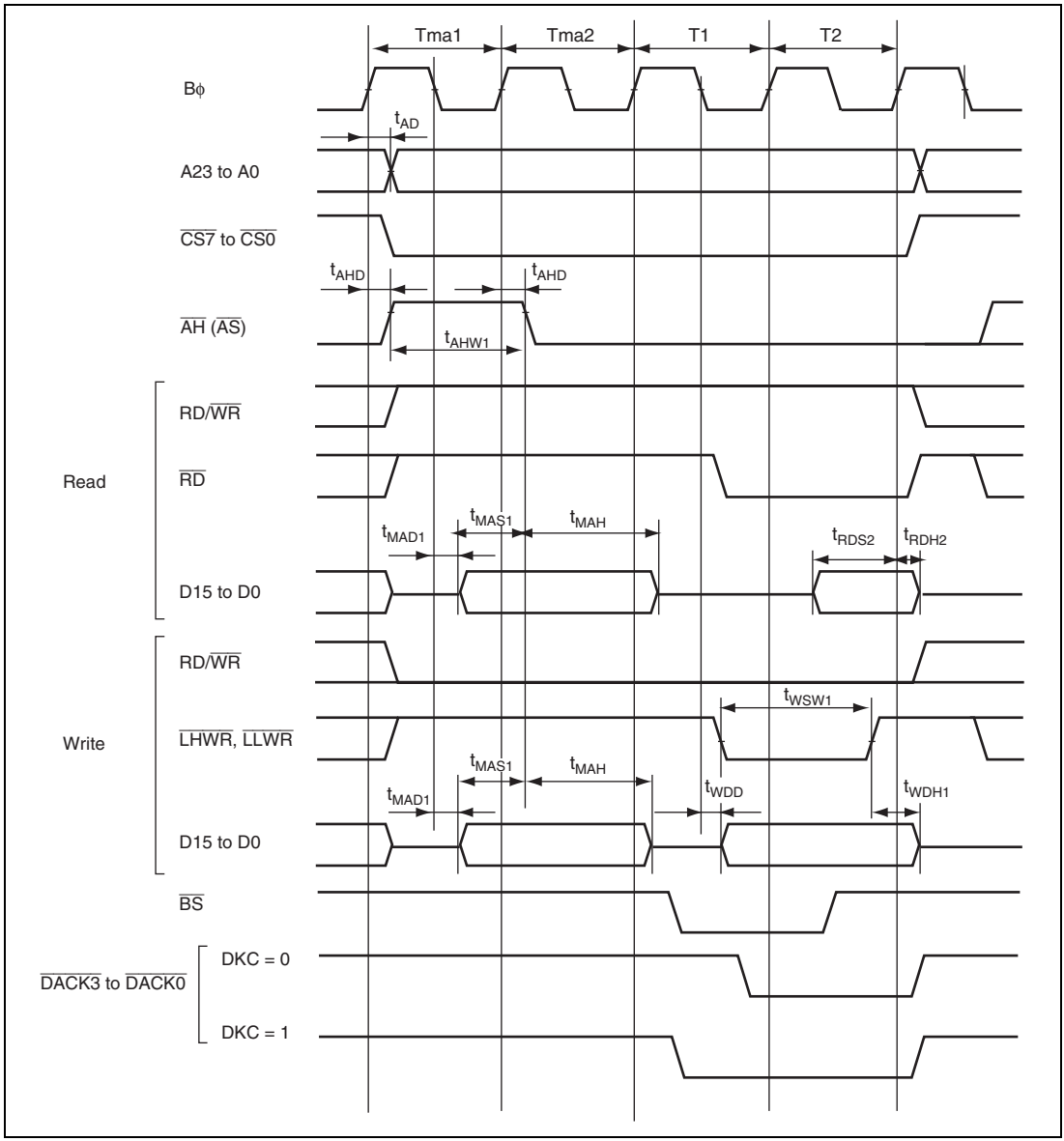


Figure 22.17 Address/Data Multiplexed Access Timing (No Wait) (Basic, 4-State Access)

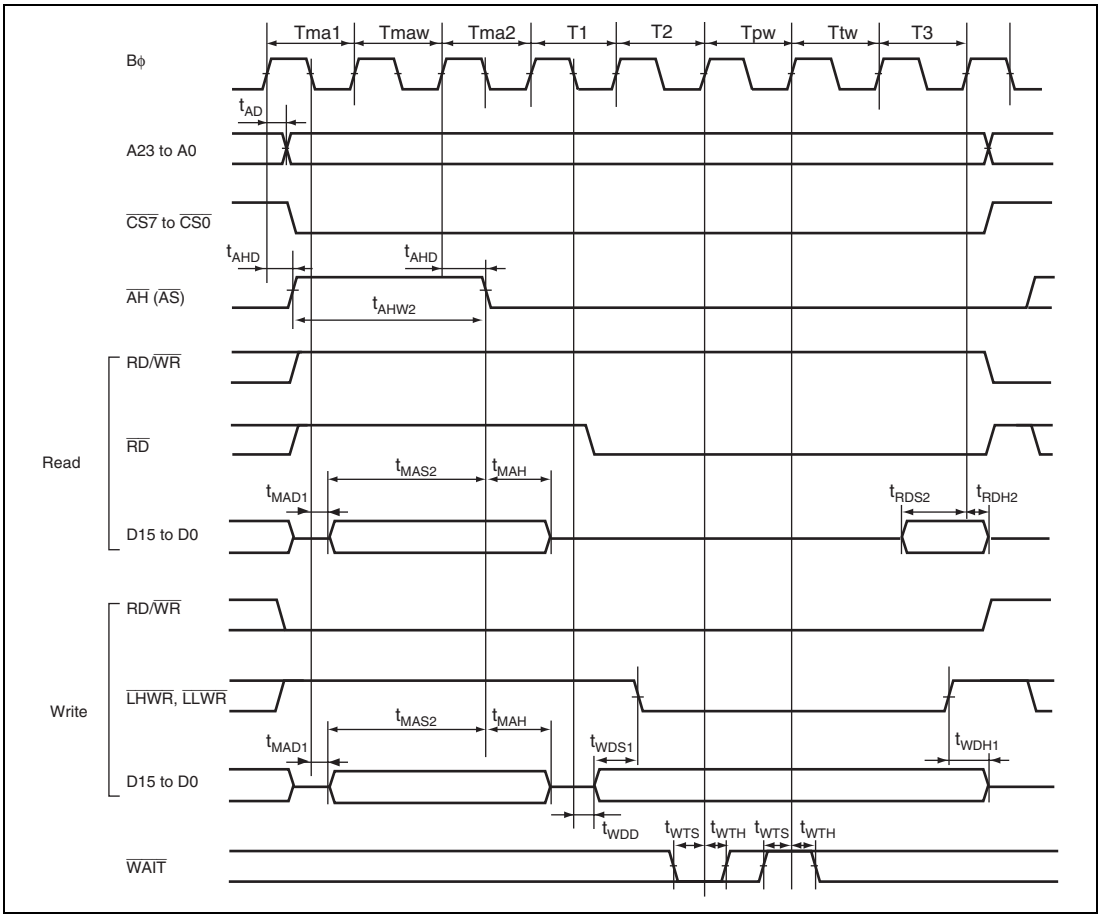


Figure 22.18 Address/Data Multiplexed Access Timing (Wait Control) (Address Cycle Program Wait × 1 + Data Cycle Program Wait × 1 + Data Cycle Pin Wait × 1)

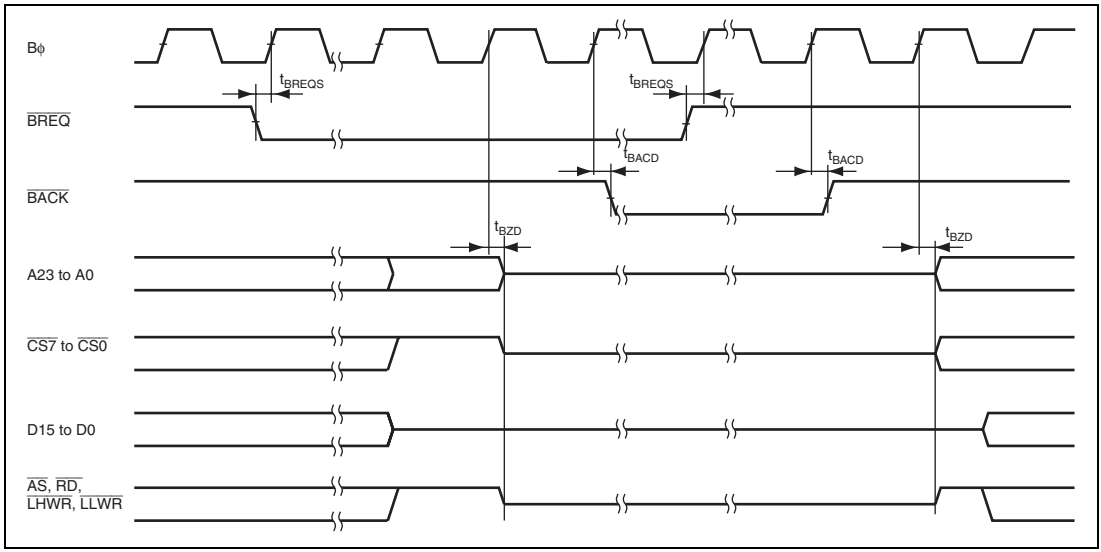


Figure 22.19 External Bus Release Timing

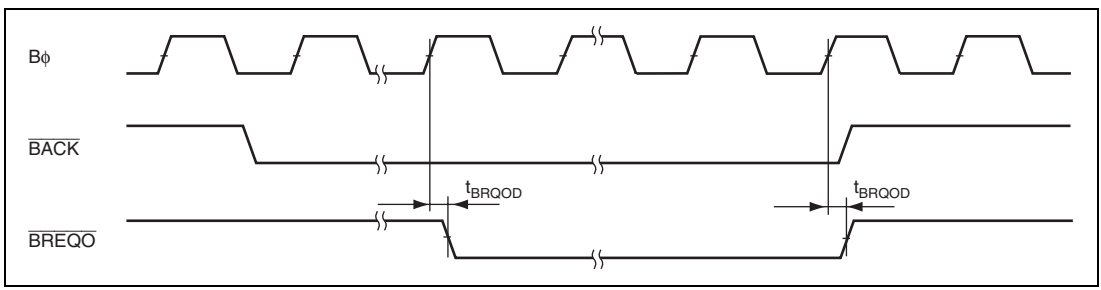


Figure 22.20 External Bus Request Output Timing

22.3.4 DMAC Timing

Table 22.7 DMAC Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Bϕ = 8 MHz to 48 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{\text{DREQ}}$ setup time	t_{DRQS}	20	—	ns	Figure 22.21
$\overline{\text{DREQ}}$ hold time	t_{DRQH}	5	—	ns	
$\overline{\text{TEND}}$ delay time	t_{TED}	—	15	ns	Figure 22.22
$\overline{\text{DACK}}$ delay time 1	t_{DACD1}	—	15	ns	Figures 22.23 and 22.24
$\overline{\text{DACK}}$ delay time 2	t_{DACD2}	—	15	ns	

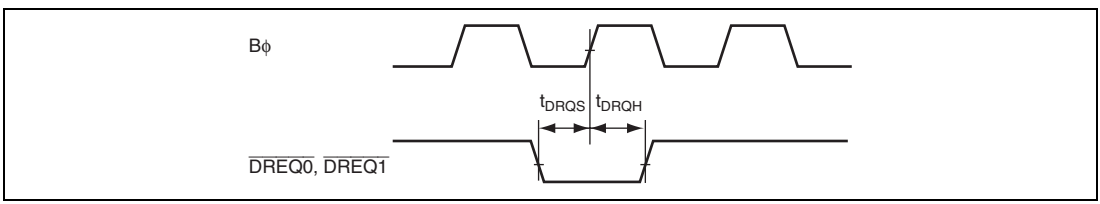


Figure 22.21 DMAC ($\overline{\text{DREQ}}$) Input Timing

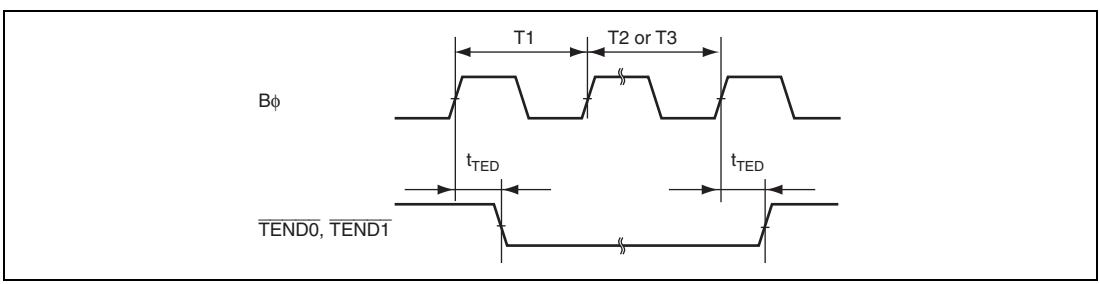


Figure 22.22 DMAC ($\overline{\text{TEND}}$) Output Timing

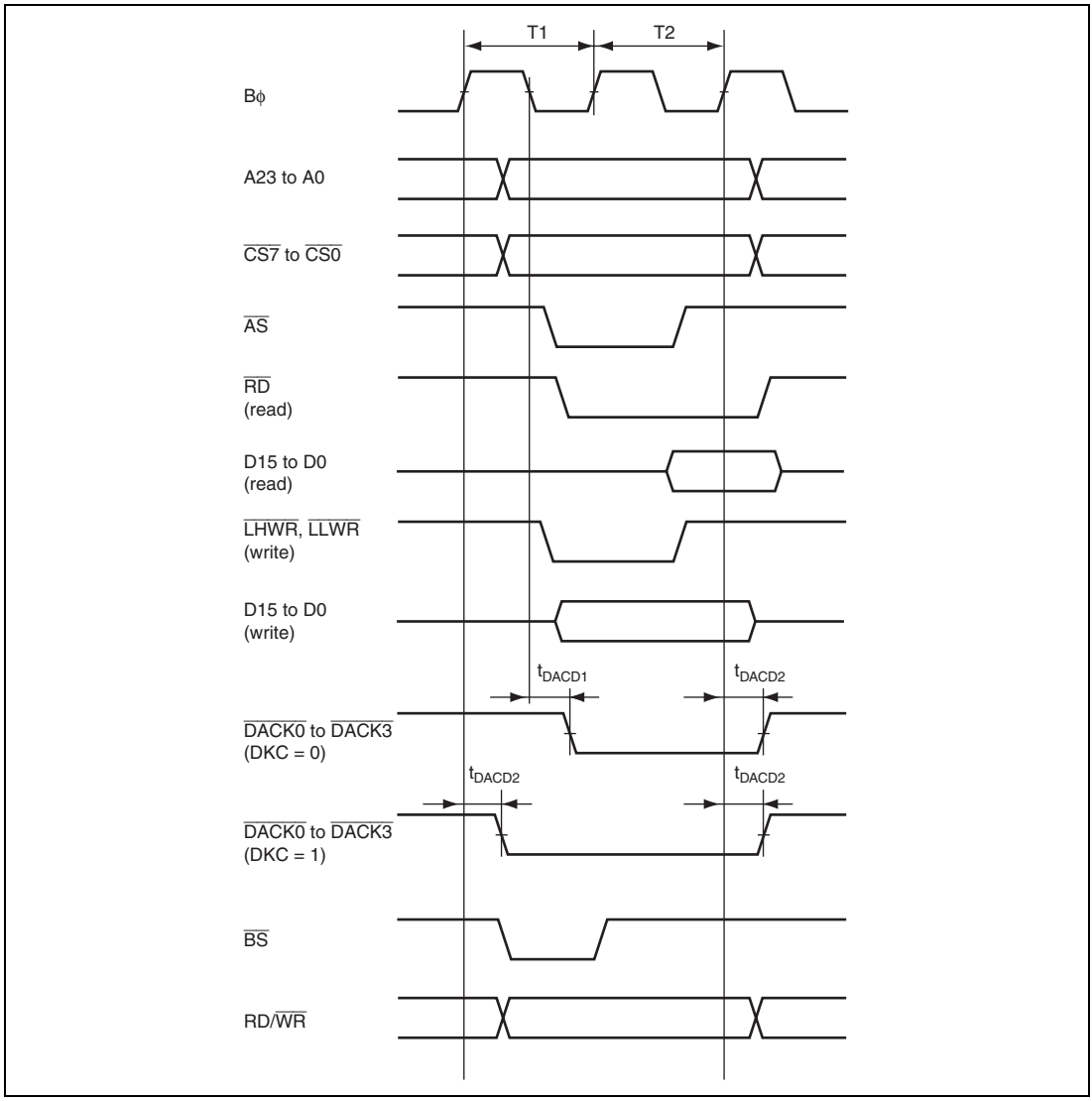


Figure 22.23 DMAC Single-Address Transfer Timing: 2-State Access

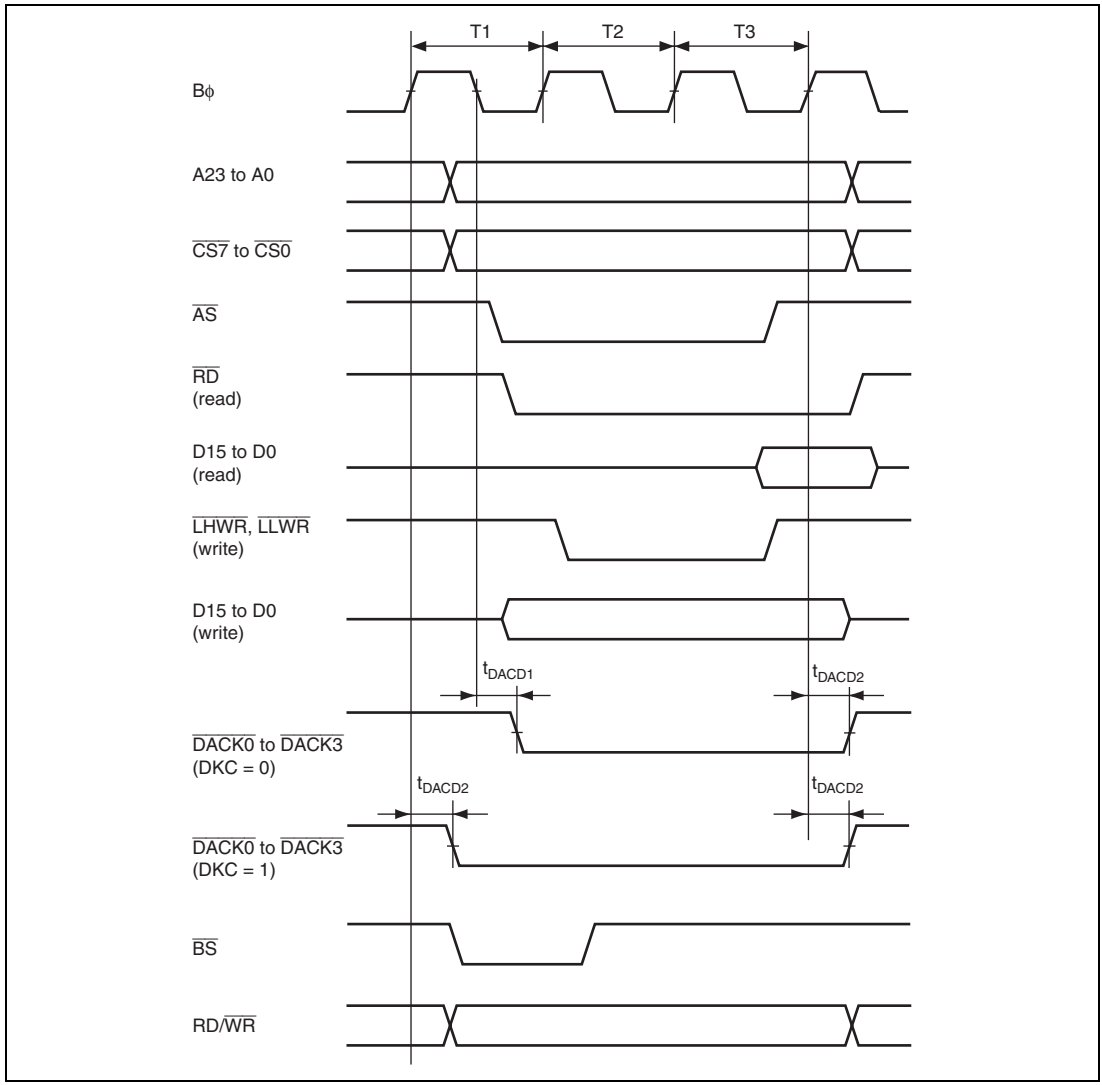


Figure 22.24 DMAC Single-Address Transfer Timing: 3-State Access

22.3.5 Timing of On-Chip Peripheral Modules

Table 22.8 Timing of On-Chip Peripheral Modules

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Pϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item			Symbol	Min.	Max.	Unit	Test Conditions
I/O ports	Output data delay time		t_{PWD}	—	40	ns	Figure 22.25
	Input data setup time		t_{PRS}	25	—	ns	
	Input data hold time		t_{PRH}	25	—	ns	
TPU	Timer output delay time		t_{TOCD}	—	40	ns	Figure 22.26
	Timer input setup time		t_{TICS}	25	—	ns	
	Timer clock input setup time		t_{TCKS}	25	—	ns	Figure 22.27
	Timer clock pulse width	Single-edge setting	t_{TCKWH}	1.5	—	t_{cyc}	
		Both-edge setting	t_{TCKWL}	2.5	—	t_{cyc}	
PPG	Pulse output delay time		t_{POD}	—	40	ns	Figure 22.28
8-bit timer	Timer output delay time		t_{TMOD}	—	40	ns	Figure 22.29
	Timer reset input setup time		t_{TMRS}	25	—	ns	Figure 22.30
	Timer clock input setup time		t_{TMCS}	25	—	ns	Figure 22.31
	Timer clock pulse width	Single-edge setting	t_{TMCWH}	1.5	—	t_{cyc}	
		Both-edge setting	t_{TMCWL}	2.5	—	t_{cyc}	
WDT	Overflow output delay time		t_{WOVD}	—	40	ns	Figure 22.32
SCI	Input clock cycle	Asynchronous	t_{Scyc}	4	—	t_{cyc}	Figure 22.33
		Clocked synchronous		6	—		
	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}	

	Item	Symbol	Min.	Max.	Unit	Test Conditions
SCI	Transmit data delay time	t_{TXD}	—	40	ns	Figure 22.34
	Receive data setup time (clocked synchronous)	t_{RXS}	40	—	ns	
	Receive data hold time (clocked synchronous)	t_{RXH}	40	—	ns	
A/D converter	Trigger input setup time	t_{TRGS}	30	—	ns	Figure 22.35

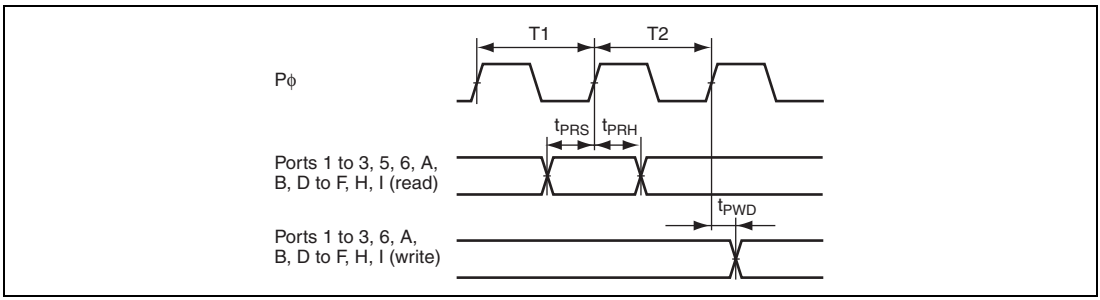


Figure 22.25 I/O Port Input/Output Timing

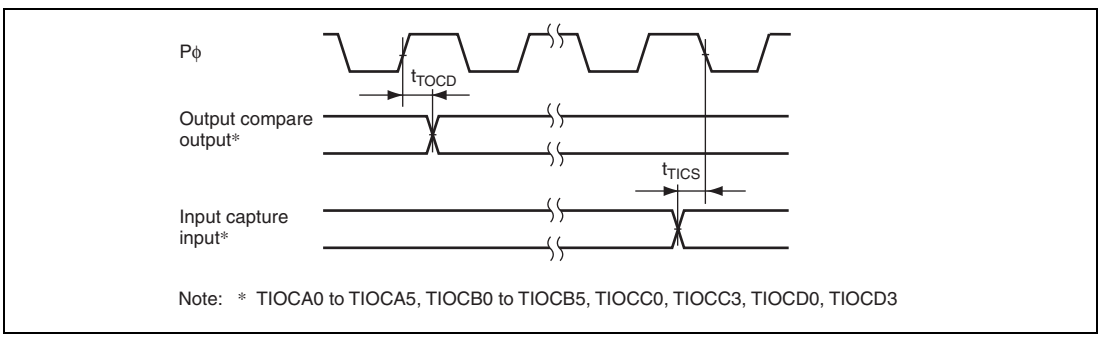


Figure 22.26 TPU Input/Output Timing

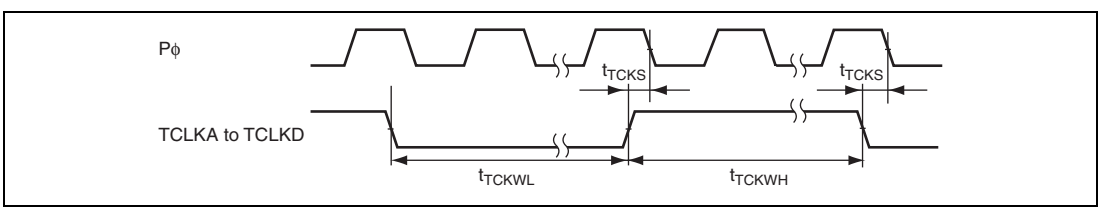


Figure 22.27 TPU Clock Input Timing

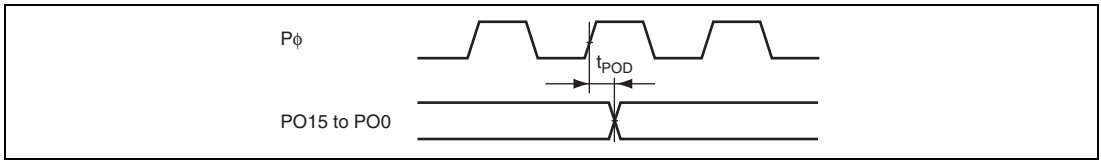


Figure 22.28 PPG Output Timing

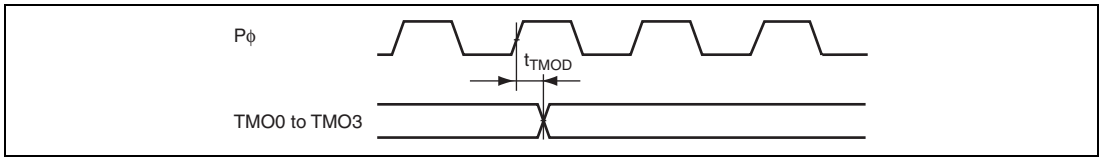


Figure 22.29 8-Bit Timer Output Timing

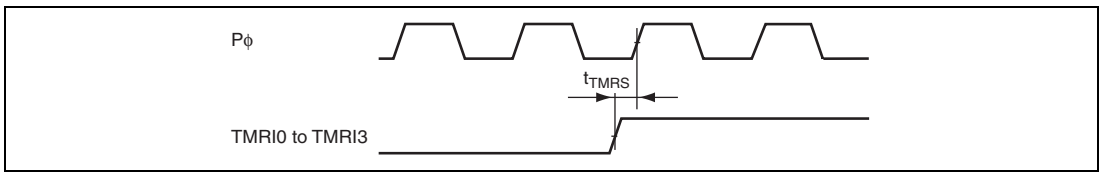


Figure 22.30 8-Bit Timer Reset Input Timing

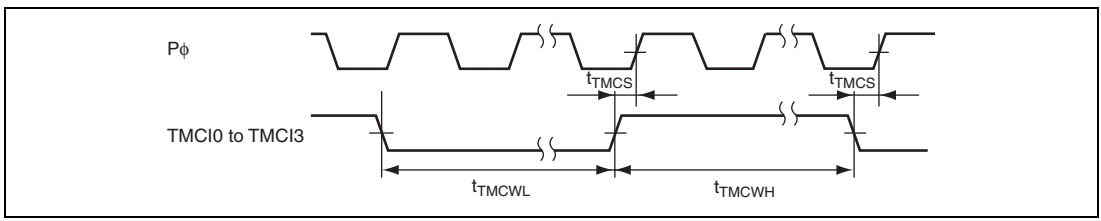


Figure 22.31 8-Bit Timer Clock Input Timing

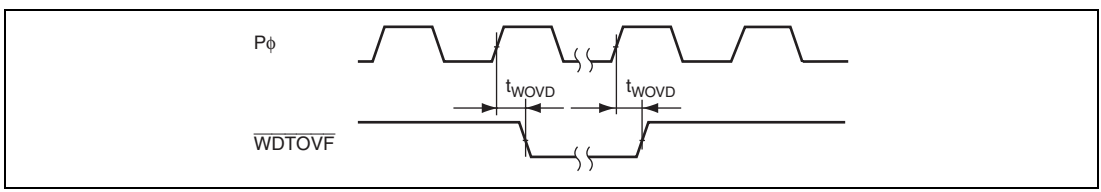


Figure 22.32 WDT Output Timing

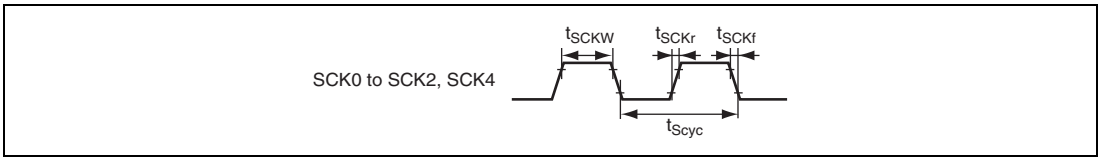


Figure 22.33 SCK Clock Input Timing

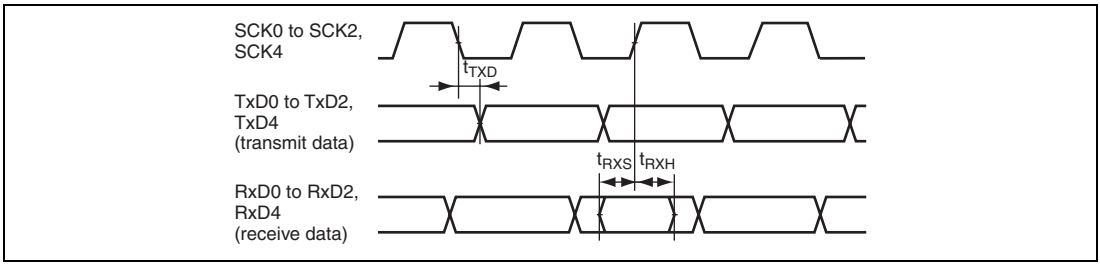


Figure 22.34 SCI Input/Output Timing: Clocked Synchronous Mode

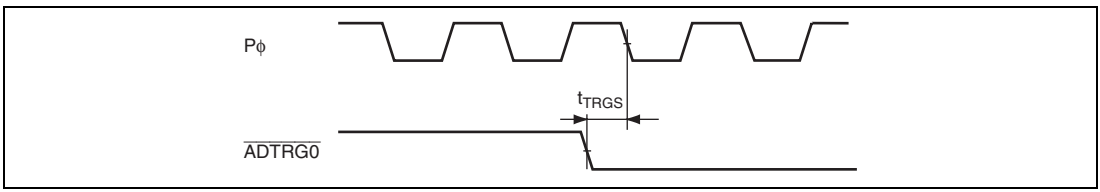


Figure 22.35 A/D Converter External Trigger Input Timing

22.4 A/D Conversion Characteristics

Table 22.9 A/D Conversion Characteristics

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Pϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit
Resolution	10	10	10	Bit
Conversion time	7.4	—	—	µs
Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	10	kΩ
Nonlinearity error	—	—	±7.5	LSB
Offset error	—	—	±7.5	LSB
Full-scale error	—	—	±7.5	LSB
Quantization error	—	±0.5	—	LSB
Absolute accuracy	—	—	±8.0	LSB

22.5 D/A Conversion Characteristics

Table 22.10 D/A Conversion Characteristics

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Pϕ = 8 MHz to 35 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit	Test Conditions
Resolution	8	8	8	Bit	
Conversion time	—	—	10	µs	20-pF capacitive load
Absolute accuracy	—	±2.0	±3.0	LSB	2-MΩ resistive load
	—	—	±2.0	LSB	4-MΩ resistive load

22.6 Flash Memory Characteristics

22.6.1 H8SX/1657C

Table 22.11 Flash Memory Characteristics

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC}, V_{SS} = AV_{SS} = 0 V,
Operating temperature range during programming/erasing:
T_a = 0°C to +75°C (regular specifications),
T_a = 0°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time*1, *2, *4	t_P	—	1	10	ms/128 bytes	
Erasure time*1, *2, *4	t_E	—	250	1500	ms/4-Kbyte block	
		—	500	4000	ms/32-Kbyte block	
		—	750	6500	ms/64-Kbyte block	
Programming time (total)*1, *2, *4	Σ_{tP}	—	6	18	s/768 Kbytes	T_a = 25°C
Erasure time (total)*1, *2, *4	Σ_{tE}	—	10	30	s/768 Kbytes	T_a = 25°C
Programming, Erasure time (total)*1, *2, *4	Σ_{tPE}	—	16	48	s/768 Kbytes	T_a = 25°C
Overwrite count	N_{WEC}	100*3	—	—	times	
Data save time*5	T_{DRP}	10	—	—	years	

Notes: 1. Programming time and erase time depend on data in the flash memory.
2. Programming time and erase time do not include time for data transfer.
3. All the characteristics after programming are guaranteed within this value (guaranteed value is from 1 to Min. value).
4. Characteristics when programming is performed within the Min. value

22.6.2 H8SX/1656C

Table 22.12 Flash Memory Characteristics

Conditions: V_{CC} = 3.0 V to 3.6 V, AV_{CC} = 3.0 V to 3.6 V, V_{ref} = 3.0 V to AV_{CC}, V_{SS} = AV_{SS} = 0 V,
Operating temperature range during programming/erasing:
T_a = 0°C to +75°C (regular specifications)
T_a = 0°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time*1, *2, *4	t_P	—	1	10	ms/128 bytes	
Erasure time*1, *2, *4	t_E	—	250	1500	ms/4-Kbyte block	
		—	500	4000	ms/32-Kbyte block	
		—	750	6500	ms/64-Kbyte block	
Programming time (total)*1, *2, *4	Σ_{tP}	—	4	12	s/512 Kbytes	T_a = 25°C
Erasure time (total)*1, *2, *4	Σ_{tE}	—	10	30	s/512 Kbytes	T_a = 25°C
Programming, Erasure time (total)*1, *2, *4	Σ_{tPE}	—	14	42	s/512 Kbytes	T_a = 25°C
Overwrite count	N_{WEC}	100*3	—	—	times	
Data retention time*4	T_{DRP}	10	—	—	years	

Notes: 1. Programming time and erasure time depend on the data in the flash memory.
2. Programming time and erasure time do not include time for data transfer.
3. All the characteristics after programming are guaranteed within this value (guaranteed value is from 1 to Min. value).
4. Characteristics when programming is performed within the Min. value

Appendix

A. Port States in Each Pin State

Table A.1 Port States in Each Pin State

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	
Port 1	All	Hi-Z	Hi-Z	Keep	Keep	Keep
Port 2	All	Hi-Z	Hi-Z	Keep	Keep	Keep
Part3	All	Hi-Z	Hi-Z	Keep	Keep	Keep
R55 to P50	All	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Keep
P56/ AN6/ DA0/ $\overline{\text{IRQ6}}$-B	All	Hi-Z	Hi-Z	[DAOE0 = 1] Keep [DAOE0 = 0] Hi-Z	[DAOE0 = 1] Keep [DAOE0 = 0] Hi-Z	Keep
P57/ AN7/ DA1/ $\overline{\text{IRQ7}}$-B	All	Hi-Z	Hi-Z	[DAOE1 = 1] Keep [DAOE1 = 0] Hi-Z	[DAOE1 = 1] Keep [DAOE1 = 0] Hi-Z	Keep
P65 to P60	All	Hi-Z	Hi-Z	Keep	Keep	Keep
PA0/ $\overline{\text{BREQO}}$/ $\overline{\text{BS}}$-A	All	Hi-Z	Hi-Z	[$\overline{\text{BREQO}}$ output] Hi-Z [$\overline{\text{BS}}$ output] Keep [Other than above] Keep	[$\overline{\text{BREQO}}$ output] Hi-Z [$\overline{\text{BS}}$ output] Hi-Z [Other than above] Keep	[$\overline{\text{BREQO}}$ output] $\overline{\text{BREQO}}$ [$\overline{\text{BS}}$ output] Hi-Z [Other than above] Keep

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	
PA1/ $\overline{\text{BACK}}$/ (RD/$\overline{\text{WR}}$)	All	Hi-Z	Hi-Z	[$\overline{\text{BACK}}$ output]	[$\overline{\text{BACK}}$ output]	[$\overline{\text{BACK}}$ output]
				Hi-Z	Hi-Z	$\overline{\text{BACK}}$
				[RD/$\overline{\text{WR}}$ output]	[RD/$\overline{\text{WR}}$ output]	
				Keep	Hi-Z	
				[Other than above]	[Other than above]	
				Keep	Keep	
PA2/ $\overline{\text{BREQ}}$/ $\overline{\text{WAIT}}$	All	Hi-Z	Hi-Z	[$\overline{\text{BREQ}}$ input]	[$\overline{\text{BREQ}}$ input]	[$\overline{\text{BREQ}}$ input]
				Hi-Z	Hi-Z	Hi-Z ($\overline{\text{BREQ}}$)
				[$\overline{\text{WAIT}}$ input]	[$\overline{\text{WAIT}}$ input]	[$\overline{\text{WAIT}}$ input]
				Hi-Z	Hi-Z	Hi-Z ($\overline{\text{WAIT}}$)
				[Other than above]	[Other than above]	
				Keep	Keep	
PA3/ $\overline{\text{LLWR}}$/ $\overline{\text{LLB}}$	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
	External extended mode (EXPE = 1)	H	Hi-Z	H	Hi-Z	Hi-Z
PA4/ $\overline{\text{LHWR}}$/ $\overline{\text{LUB}}$	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
	External extended mode (EXPE = 1)	H	Hi-Z	[$\overline{\text{LHWR}}$, $\overline{\text{LUB}}$ output]	[$\overline{\text{LHWR}}$, $\overline{\text{LUB}}$ output]	[$\overline{\text{LHWR}}$, $\overline{\text{LUB}}$ output]
				H	Hi-Z	Hi-Z
				[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep
PA5/$\overline{\text{RD}}$	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
	External extended mode (EXPE = 1)	H	Hi-Z	H	Hi-Z	Hi-Z

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	
PA6/ $\overline{AS}$/ $\overline{AH}$/ $\overline{BS}$-B	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	[$\overline{AS}$, $\overline{BS}$ output]	[$\overline{AS}$, $\overline{AH}$, $\overline{BS}$ output]	[$\overline{AS}$, $\overline{AH}$, $\overline{BS}$ output]
	External extended mode (EXPE = 1)	H		H	Hi-Z	Hi-Z
				[$\overline{AH}$ output]	[Other than above]	[Other than above]
				L	Keep	Keep
				[Other than above]		
				Keep		
PA7/Bϕ	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	[Clock output]	[Clock output]	[Clock output]
	External extended mode (EXPE = 1)	Clock output	Hi-Z	H	H	Clock output
				[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep
PB0/ $\overline{CS0}$/ $\overline{CS4}$/ $\overline{CS5}$-B	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	[$\overline{CS}$ output]	[$\overline{CS}$ output]	[$\overline{CS}$ output]
	External extended mode (EXPE = 1)	H		H	Hi-Z	Hi-Z
				[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep
PB1/ $\overline{CS1}$/ $\overline{CS2}$-B/ $\overline{CS5}$-A/ $\overline{CS6}$-B/ $\overline{CS7}$-B	All	Hi-Z	Hi-Z	[$\overline{CS}$ output]	[$\overline{CS}$ output]	[$\overline{CS}$ output]
				H	Hi-Z	Hi-Z
				[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep
PB2/ $\overline{CS2}$-A/ $\overline{CS6}$-A	All	Hi-Z	Hi-Z	[$\overline{CS}$ output]	[$\overline{CS}$ output]	[$\overline{CS}$ output]
				H	Hi-Z	Hi-Z
				[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep
PB3/ $\overline{CS3}$/ $\overline{CS7}$-A	All	Hi-Z	Hi-Z	[$\overline{CS}$ output]	[$\overline{CS}$ output]	[$\overline{CS}$ output]
				H	Hi-Z	Hi-Z
				[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	
Port D	External extended mode (EXPE = 1)	L	Hi-Z	Keep	Hi-Z	Hi-Z
	ROM enabled extended mode	Hi-Z	Hi-Z	Keep	[Address output]	[Address output]
					Hi-Z	Hi-Z
					[Other than above]	[Other than above]
					Keep	Keep
	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
Port E	External extended mode (EXPE = 1)	L	Hi-Z	Keep	Hi-Z	Hi-Z
	ROM enabled extended mode	Hi-Z	Hi-Z	Keep	[Address output]	[Address output]
					Hi-Z	Hi-Z
					[Other than above]	[Other than above]
					Keep	Keep
	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
PF7 to PF0	External extended mode (EXPE = 1)	L/ Hi-Z*	Hi-Z	Keep	[Address output]	[Address output]
					Hi-Z	Hi-Z
					[Other than above]	[Other than above]
					Keep	Keep
	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
Port H	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep
	External extended mode (EXPE = 1)	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z

Port Name	MCU Operating Mode		Reset	Hardware Standby Mode	Software Standby Mode		Bus Released State
					OPE = 1	OPE = 0	
Port I	Single-chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	Keep	Keep
	External extended mode (EXPE = 1)	8-bit bus mode	Hi-Z	Hi-Z	Keep	Keep	Keep
		16-bit bus mode	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
		32-bit bus mode	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z

[Legend]

H: High level

L: Low level

Keep: Input ports become high-impedance, and output ports retain the state.

Hi-Z: High impedance

B. Product Lineup

Product Classification	Product Model	Marking	Package (Package Code)
H8SX/1657C	R5F61657CFTV	R5F61657FTV	TFP-120 (TFP-120V*)
H8SX/1656C	R5F61656CFTV	R5F61656FTV	

Note: * Pb-free version

C. Package Dimensions

For the package dimensions, data in the Renesas IC Package General Catalog has priority.

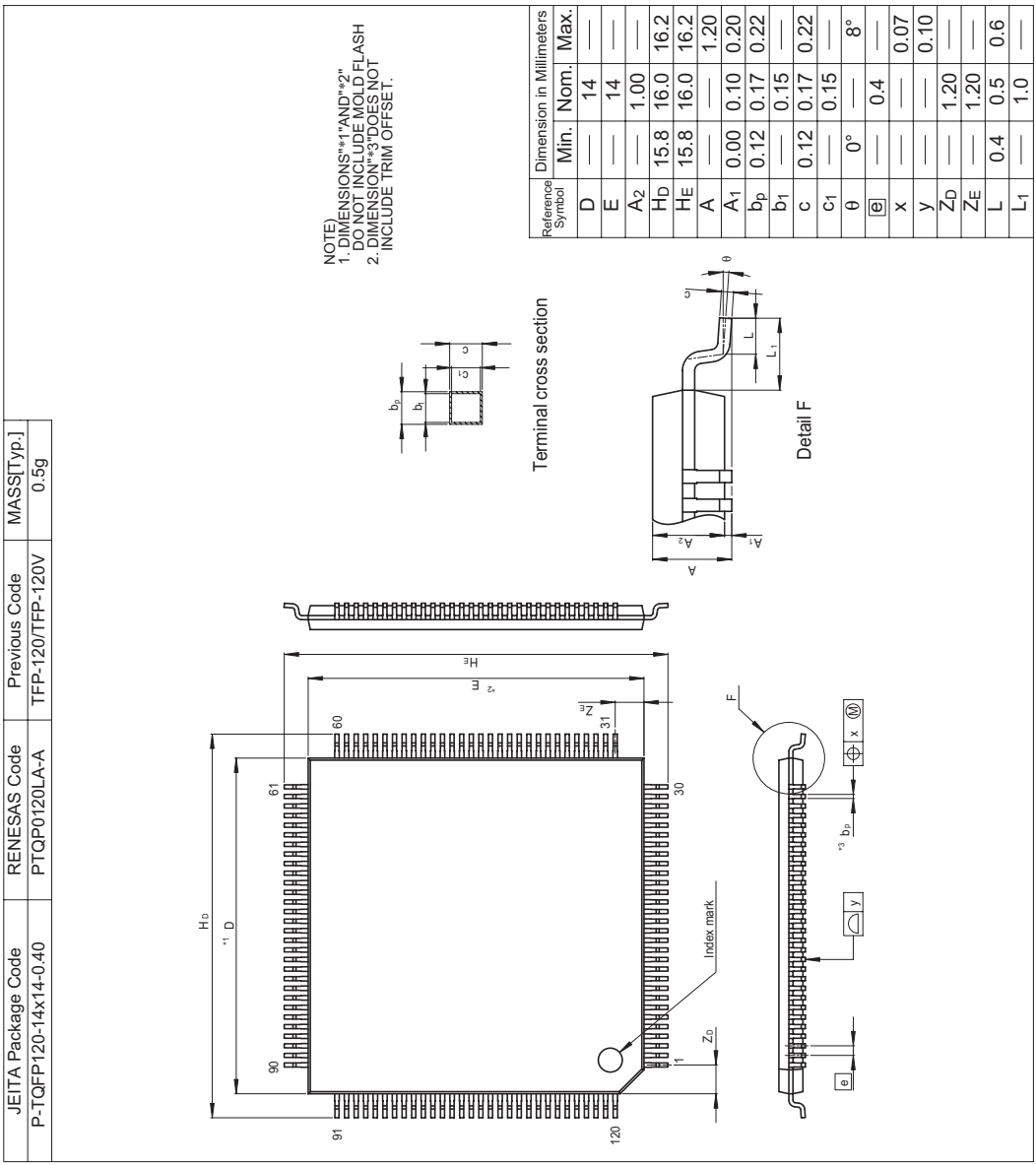


Figure C.1 Package Dimensions (TFP-120V)

D. Treatment of Unused Pins

The treatments of unused pins are listed in table D.1.

Table D.1 Treatment of Unused Pins

Pin Name	Mode 4	Mode 5	Mode 6	Mode 7
$\overline{\text{RES}}$	(Always used as a reset pin)			
$\overline{\text{STBY}}$	• Connect to V_{CC} via a pull-up resistor.			
EMLE	• Connect to V_{SS} via a pull-down resistor.			
MD2, MD1, MD0	(Always used as mode pins)			
NMI	• Connect to V_{CC} via a pull-up resistor.			
EXTAL	(Always used as a clock pin)			
XTAL	• Leave the pin unconnected.			
$\overline{\text{WDTOVF}}$	• Leave the pin unconnected.			
Port 1 Port 2 Port 3 Port 6 PA2 to PA0 PB3 to PB1 PF7 to PF5	• Connect each pin to V_{CC} via a pull-up resistor or to V_{SS} via a pull-down resistor.			
Port 5	• Connect each pin to AV_{CC} via a pull-up resistor or to AV_{SS} via a pull-down resistor.			

<table>
<tr><th>Pin Name</th><th>Mode 4</th><th>Mode 5</th><th>Mode 6</th><th>Mode 7</th></tr>
<tr><td>PA7</td><td colspan="3">• Since this is the Bϕ output in its initial state, leave the pin unconnected.</td><td rowspan="9">• Connect each pin to V_{CC} via a pull-up resistor or to V_{SS} via a pull-down resistor.</td></tr>
<tr><td>PA6</td><td colspan="3">• Since this is the $\overline{AS}$ output in its initial state, leave the pin unconnected.</td></tr>
<tr><td>PA5</td><td colspan="3">• Since this is the $\overline{RD}$ output in its initial state, leave the pin unconnected.</td></tr>
<tr><td>PA4</td><td colspan="3">• Since this is the $\overline{LHWR}$ output in its initial state, leave the pin unconnected.</td></tr>
<tr><td>PA3</td><td colspan="3">• Since this is the $\overline{LLWR}$ output in its initial state, leave the pin unconnected.</td></tr>
<tr><td>PB0</td><td colspan="2">• Since this is the $\overline{CS0}$ output in its initial state, leave the pin unconnected.</td><td></td></tr>
<tr><td>Port D
Port E
PF4 to PF0</td><td colspan="2">• Since this is the address output in its initial state, leave the pin unconnected.</td><td></td></tr>
<tr><td>Port H</td><td colspan="3">(Used as a data bus)</td></tr>
<tr><td>Port I</td><td>(Used as a data bus)</td><td colspan="2">Since this is a general-purpose input port in its initial state, connect each pin to V_{CC} via a pull-up resistor or connect each pin to V_{SS} via a pull-down resistor.</td></tr>
<tr><td>V_{ref}</td><td colspan="4">• Connect to AV_{CC}.</td></tr>
</table>

Notes: 1. Do not change the initial value (input buffer disabled) of PnICR corresponding to an unused pin.
2. When changing the pin function from its initial state, use a pull-up or pull-down resistor as needed.

Main Revisions and Additions in this Edition

Item	Page	Revision (See Manual for Details)
All	—	Modified Type classification changed (from H8SX/1657B to H8SX/1657C). Type classification changed (from H8SX/1656 to H8SX/1656C). Type name changed (from R5F61657BFTV to R5F61657CFTV). Type name changed (from R5F61656FTV to R5F61656CFTV).
Figure 1.3 Pin Assignments	10	Deleted Note
18.8.2 User Program Mode (2) Programming Procedure in User Program Mode	689	Modified 3. After initializing VBR to H'00000000, set the SCO bit to 1 to execute download. To set the SCO bit to 1, all of the following conditions must be satisfied. : —The values of general registers other than ER0 and ER1 are held during download.
18.13 Standard Serial Communication Interface Specifications for Boot Mode (3) Inquiry and Selection States (e) Multiplication Ratio Inquiry	721	Modified • Command, H'22, (one byte): Inquiry regarding multiplication ratio

Response

H'32	Size	Number of multiplication types					
Number of multiplication ratios	Multiplication ratio	· · ·					
...							
SUM							

• Response, H'32, (one byte): Response to the multiplication ratio inquiry

• Size (one byte): The amount of data that represents the number of multiplication types and multiplication ratios and the multiplication ratios

• Number of multiplication types (one byte): The number of multiplication types to which the device can be set.

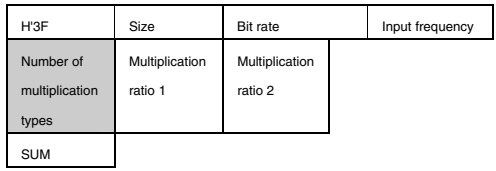
:

• Multiplication ratio (one byte)

Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H'04.)

Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are multiplication types.

Item	Page	Revision (See Manual for Details)
18.13 Standard Serial Communication Interface Specifications for Boot Mode (3) Inquiry and Selection States (k) New Bit-Rate Selection	725	Modified Command  H'3F | Size | Bit rate | Input frequency Number of multiplication types | Multiplication ratio 1 | Multiplication ratio 2 SUM • Command, H'3F, (one byte): Selection of new bit rate • Size (one byte): The amount of data that represents the bit rate, input frequency, number of multiplication types, and multiplication ratio : • Number of multiplication types (one byte): The number of multiplication types to which the device can be set.
18.14 Usage Notes	743	Modified 15. The contents of general registers ER0 and ER1 are not saved during download of an on-chip program, initialization, programming, or erasure. When needed, save the general registers before a download request or before execution of initialization, programming, or erasure using the procedure program.
19.5.1 Notes on Clock Pulse Generator	751	Deleted ~~4. Note that the frequency of φ will be changed in the middle of a bus cycle when setting SCKCR while executing the external bus cycle with the write data buffer function.~~

Index

Q

R

S

T

U

V

W

Renesas 32-Bit CISC Microcomputer
Hardware Manual
H8SX/1657 Group

Publication Date:	Rev.1.00, Sep. 25, 2006
	Rev.2.00, Jun. 28, 2007
Published by:	Sales Strategic Planning Div.
	Renesas Technology Corp.
Edited by:	Customer Support Department
	Global Strategic Communication Div.
	Renesas Solutions Corp.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

H8SX/1657 Group
Hardware Manual

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ09B0341-0200