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.

To all our customers

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

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

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

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

## Cautions

Keep safety first in your circuit designs!

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

Notes regarding these materials

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

User's Manual

32

# SH7020 and SH7021

## Hardware Manual

## SuperH™ RISC engine

HD6437020, HD6477021,
HD6437021, HD6417021

Renesas Electronics
www.renesas.com

Rev.3.0 1998.09

# Cautions

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

# Introduction

The SH7020 and SH7021 are part of a new generation of reduced instruction-set computer-type (RISC) microcomputers that integrate RISC-type CPUs and the peripheral functions required for system configuration onto a single chip to achieve high-performance operations processing. They can operate in a power-down state, which is an essential feature for portable equipment.

The SH7020 and SH7021 CPUs have RISC-type instruction sets. Basic instructions can be executed in a single clock cycle, which strikingly improves instruction execution speed. The SH7020 and SH7021 include peripheral functions such as large-capacity ROM (PROM or masked ROM), RAM, a direct memory access controller (DMAC), timers, a serial communication interface (SCI), an interrupt controller (INTC), and I/O ports. These on-chip elements enable users to construct systems with the fewest possible components. External memory access support functions enable direct connection to SRAM and DRAM. without the use of glue logics.

This Hardware Manual describes in detail the hardware functions of the SH7020 and SH7021. For information on the instructions, please refer to the Programming Manual.

## Related Manuals

SH7000 Series Instructions

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

For development support tools, contact your Hitachi sales office.

## Organization of This Manual

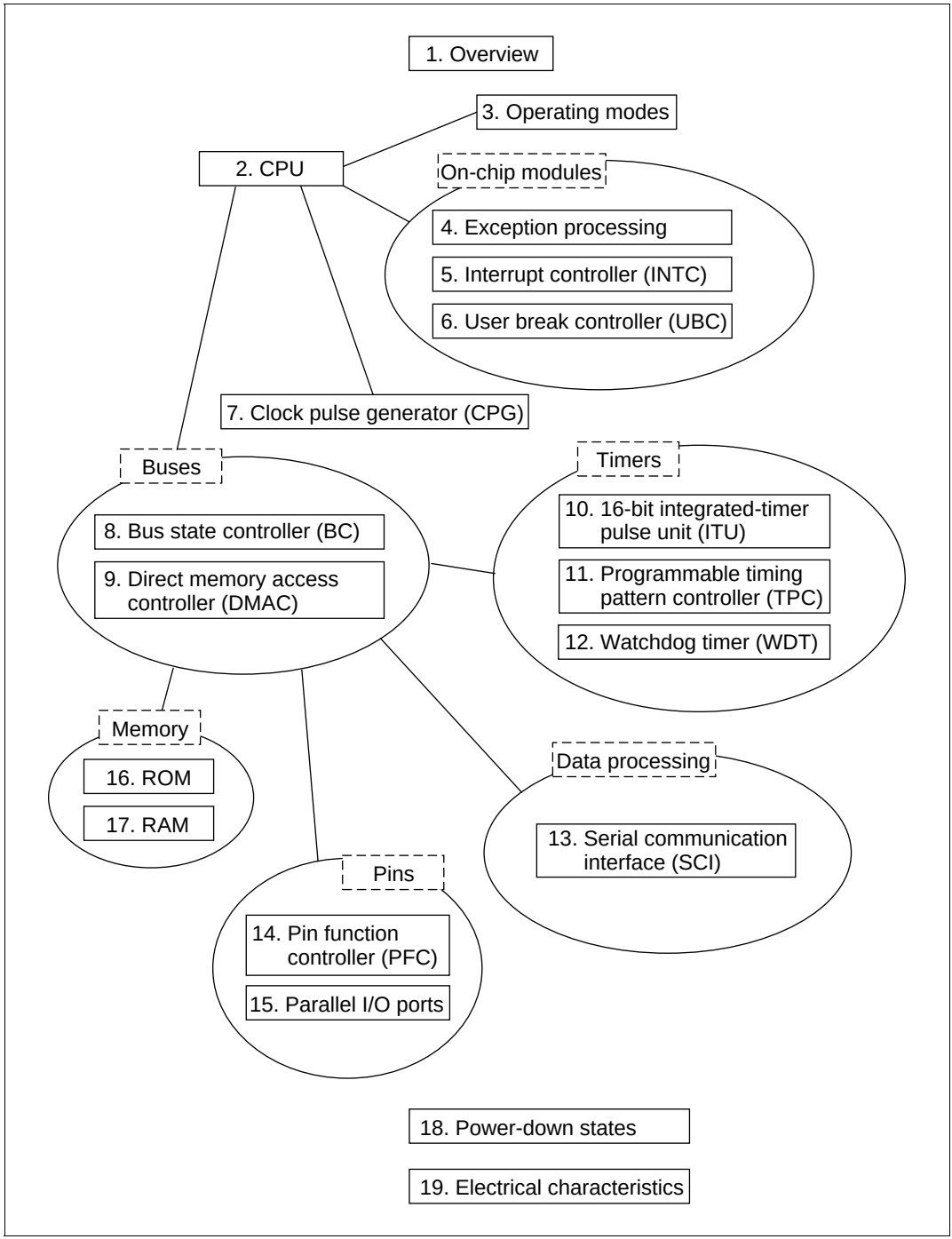
Table 1 describes how this manual is organized. Figure 1 shows the relationships between the Sections within this manual.

**Table 1 Manual Organization**

| Category | Section Title | Abbrevi-ation | Contents |
|---|---|---|---|
| Overview | 1. Overview | — | Features, internal block diagram, pin layout, pin functions |
| CPU | 2. CPU | CPU | Register configuration, data structure. instruction features, instruction types, instruction lists |
| Operating Modes | 3. Operating Modes | — | MCU mode, PROM mode |
| Internal Modules | 4. Exception Processing | — | Resets, address errors, interrupts, trap instructions, illegal instructions |
| | 5. Interrupt Controller | INTC | NMI interrupts, user break interrupts, IRQ interrupts, on-chip module interrupts |
| | 6. User Break Controller | UBC | Break address and break bus cycles selection |
| Clock | 7. Clock Pulse Generator | CPG | Crystal pulse generator, duty correction circuit |
| Buses | 8. Bus State Controller | BSC | Division of memory space, DRAM interface, refresh, wait state control, parity control |
| | 9. Direct Memory Access Controller | DMAC | Auto request, external request, on-chip peripheral module request, cycle steal mode, burst mode |
| Timers | 10. 16-Bit Integrated-Timer Pulse Unit | ITU | Waveform output mode, input capture function, counter clear function, buffer operation, PWM mode, complementary PWM mode, reset synchronized mode, synchronized operation, phase counting mode, compare match output mode |
| | 11. Programmable Timing Pattern Controller | TPC | Compare match output triggers, non-overlap operation |
| | 12. Watchdog Timer | WDT | Watchdog timer mode, interval timer mode |
| Data Processing | 13. Serial Communica-tion Interface | SCI | Asynchronous mode, clock synchronous mode, multiprocessor communication function |

**Table 1 Manual Organization (cont)**

| Category | Section Title | Abbrevi-ation | Contents |
|---|---|---|---|
| Pins | 14. Pin Function Controller | PFC | Pin function selection |
| | 15. Parallel I/O Ports | I/O | I/O port |
| Memory | 16. ROM | ROM | On-chip ROM |
| | 17. RAM | RAM | On-chip RAM |
| Power-Down States | 18. Power-Down States | — | Sleep mode, standby mode |
| Electrical Characteristics | 19. Electrical Characteristics | — | Absolute maximum ratings, AC characteristics, DC characteristics, operation timing |




**Manual Organization Scheme**

## Addresses of On-Chip Peripheral Module Registers

The on-chip peripheral module registers are located in the on-chip peripheral module space (area 5: H'5000000–H'5FFFFFF), but since the actual register space is only 512 bytes, address bits A23–A9 are ignored. 32k shadow areas in 512 byte units that contain exactly the same contents as the actual registers are thus provided in the on-chip peripheral module space.

In this manual, register addresses are specified as though the on-chip peripheral module registers were in the 512 bytes H'5FFFE00–H'5FFFFFF. Only the values of the A27–A24 and A8–A0 bits are valid; the A23–A9 bits are ignored. When H'5000000–H'50001FF is accessed, for example, the result will be the same as when H'5FFFE00–H'5FFFFFF is accessed. For more details, see Section 8.3.5, Area Description: Area 5.

## Free Addresses in the On-chip Peripheral Module Space (Area 5)

Avoid reading/writing from/to the free addresses without registers in the on-chip peripheral module space (area 5: H'5000000-H'5FFFFFF).

# Contents

# Section 1 Overview

## 1.1 SuperH Microcomputer Features

The SuperH microcomputer (SH7000 series) is a new generation reduced instruction set computer (RISC) in which a Hitachi-original CPU and the peripheral functions required for system configuration are integrated onto a single chip.

The CPU has a RISC-type instruction set. Most instructions can be executed in one clock cycle, which strikingly improves instruction execution speed. In addition, the CPU has a 32-bit internal architecture for enhanced data-processing ability. As a result, the CPU enables high-performance systems to be constructed with advanced functionality at low cost, even in applications such as realtime control that require very high speeds, an impossibility with conventional microcomputers.

The SH microcomputer includes peripheral functions such as large-capacity ROM, RAM, a direct memory access controller (DMAC), timers, a serial communication interface (SCI), an interrupt controller (INTC), and I/O ports. External memory access support functions enable direct connection to SRAM and DRAM. These features can drastically reduce system cost.

For on-chip ROM, masked ROM or electrically programmable ROM (PROM) can be selected. The PROM version can be programmed by users with a general-purpose EPROM programmer.

Table 1.1 lists the features of the SH microcomputers (SH7020 and SH7021).

**Table 1.1 Features of the SH7020 and SH7021 Microcomputers**

| Feature | Description |
|---|---|
| CPU | Original Hitachi architecture |
| | 32-bit internal data paths |
| | General-register machine:<br>• Sixteen 32-bit general registers<br>• Three 32-bit control registers<br>• Four 32-bit system registers |
| | RISC-type instruction set:<br>• Instruction length: 16-bit fixed length for improved code efficiency<br>• Load-store architecture (basic arithmetic and logic operations are executed between registers)<br>• Delayed unconditional branch instructions reduce pipeline disruption<br>• Instruction set optimized for C language |
| | Instruction execution time: one instruction/cycle (50 ns/instruction at 20-MHz operation) |
| | Address space: 4 Gbytes available on the architecture |
| | On-chip multiplier: multiplication operations (16 bits × 16 bits → 32 bits) executed in 1–3 cycles, and multiplication/accumulation operations (16 bits × 16 bits + 42 bits → 42 bits) executed in 2–3 cycles |
| | Five-stage pipeline |
| Operating modes | Operating modes:<br>• On-chip ROMless mode<br>• On-chip ROM mode |
| | Processing states:<br>• Power-on reset state<br>• Manual reset state<br>• Exception processing state<br>• Program execution state<br>• Power-down state<br>• Bus-released state |
| | Power-down states:<br>• Sleep mode<br>• Software standby mode |

**Table 1.1 Features of the SH7020 and SH7021 Microcomputers (cont)**

| Feature | Description |
|---|---|
| Interrupt controller (INTC) | Nine external interrupt pins (NMI, IRQ0–IRQ7) |
| | Thirty internal interrupt sources |
| | Sixteen programmable priority levels |
| User break controller (UBC) | Generates an interrupt when the CPU or DMAC generates a bus cycle with specified conditions |
| | Simplifies configuration of a self-debugger |
| Clock pulse generator (CPG) | On-chip clock pulse generator (maximum operating frequency: 20 MHz):<br>• 20-MHz pulses can be generated from a 20-MHz crystal with a duty cycle correcting circuit |
| Bus state controller (BSC) | Supports external memory access:<br>• Sixteen-bit external data bus |
| | Address space divided into eight areas with the following preset features:<br>• Bus size (8 or 16 bits)<br>• Number of wait cycles can be defined by user.<br>• Type of area (external memory area, DRAM area, etc.)<br>— Simplifies connection to ROM, SRAM, DRAM, and peripheral I/O<br>• When the DRAM area is accessed:<br>— RAS and CAS signals for DRAM are output<br>— Tp cycles can be generated to assure RAS precharge time<br>— Address multiplexing is supported internally, so DRAM can be connected directly<br>• Chip select signals (CS0 to CS7) are output for each area |
| | DRAM refresh function:<br>• Programmable refresh interval<br>• Supports CAS-before-RAS refresh and self-refresh modes |
| | DRAM burst access function:<br>• Supports high-speed access modes for DRAM |
| | Wait cycles can be inserted by an external WAIT signal |
| | One-stage write buffer improves the system performance |
| | Data bus parity can be generated and checked |

**Table 1.1 Features of the SH7032 and SH7034 Microcomputers (cont)**

| Feature | Description |
|---|---|
| Direct memory access controller (DMAC) (4 channels) | Permits DMA transfer between the following modules:<br>• External memory<br>• External I/O<br>• On-chip memory<br>• Peripheral on-chip modules (except DMAC) |
| | DMA transfer can be requested from external pins, on-chip SCI, on-chip timers, and on-chip A/D converter |
| | Cycle-steal mode or burst mode |
| | Channel priority level is selectable |
| | Channels 0 and 1: dual or single address transfer mode is selectable; external request sources are supported; Channels 2 and 3: dual address transfer mode, internal request sources only |
| 16-bit integrated-timer pulse unit (ITU) | Ten types of waveforms can be output |
| | Input pulse width and cycle can be measured |
| | PWM mode: pulse output with 0–100% duty cycle (maximum resolution: 50 ns) |
| | Complementary PWM mode: can output a maximum of three pairs of non-overlapping PWM waveforms |
| | Phase counting mode: can count up or down according to the phase of an external two-phase clock |
| Timing pattern controller (TPC) | Maximum 16-bit output (4 bits × 4 channels) can be output |
| | Non-overlap intervals can be established between pairs of waveforms |
| | Timing-source timer is selectable |
| Watchdog timer (WDT) (1 channel) | Can be used as watchdog timer or interval timer |
| | Timer overflow can generate an internal reset, external signal, or interrupt |
| | Power-on reset or manual reset can be selected as the internal reset |
| Serial communication interface (SCI) (2 channels) | Asynchronous or clocked synchronous mode is selectable |
| | Can transmit and receive simultaneously (full duplex) |
| | On-chip baud rate generator in each channel |
| | Multiprocessor communication function |

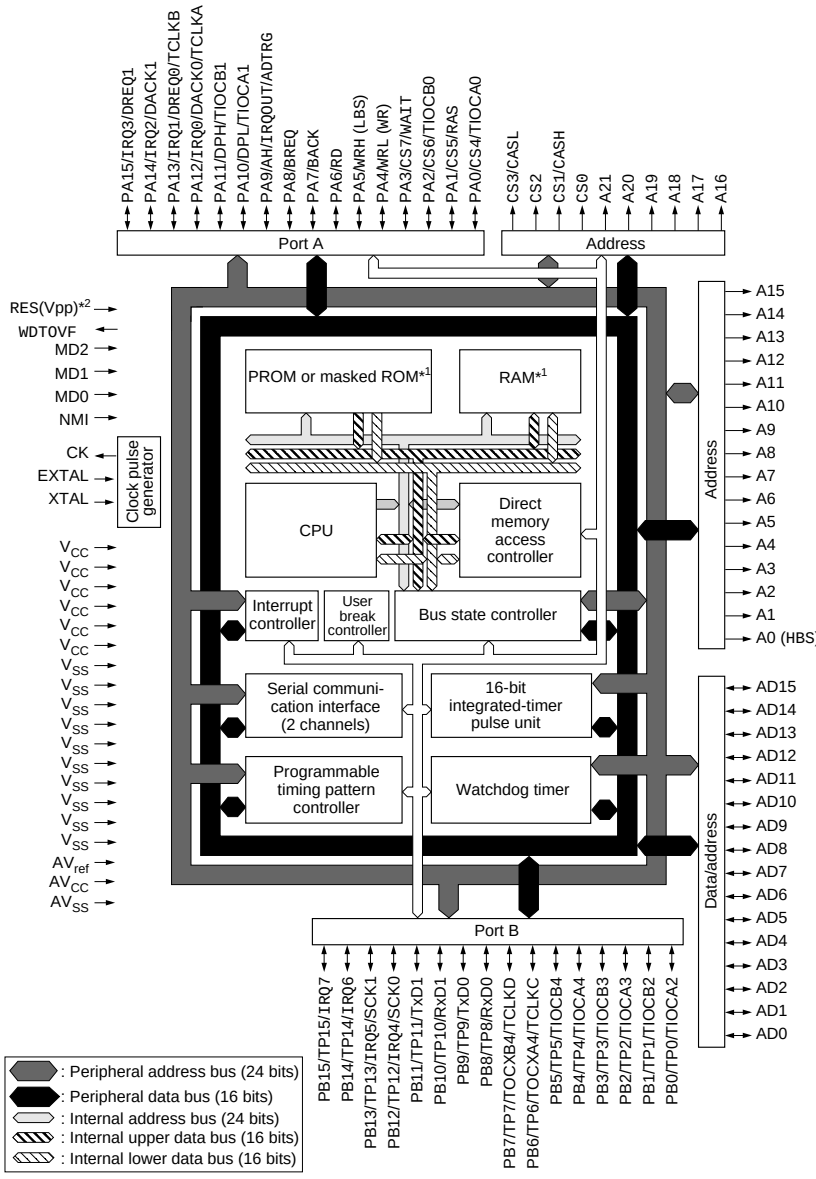
**Table 1.1 Features of the SH7032 and SH7034 Microcomputers (cont)**

| Feature | Description |
|---|---|
| I/O ports | Total of 40 I/O lines (32 input/output lines, 8 input-only lines):<br>• Port A: 16 input/output lines (input or output can be selected for each bit)<br>• Port B: 16 input/output lines (input or output can be selected for each bit) |
| On-chip memory | SH7020: 16-kbyte masked ROM, and 1-kbyte RAM |
| | SH7021: 32-kbyte electrically programmable ROM or masked Rom, and 1-kbyte RAM |
| | 32-bit data can be accessed in one clock cycle |

**Table 1.2 Product Line**

| Product Number | On-Chip ROM | Operating Voltage | Operating Frequency | Operating temperature | Model | Marking Model No. | Package |
|---|---|---|---|---|---|---|---|
| SH7021 | masked ROM | 5.0V | 2 to 20MHz | -20 to +75 °C | HD6437021X | HD6437021TE | 100-pin plastic TQFP (TFP-100B) |
| | | | 2 to 16.6MHz | -40 to +85 °C | HD6437021XI | HD6437021TEI | |
| | | 3.3V | 2 to 12.5MHz | -20 to +75 °C | HD6437021VX | HD6437021VTE | |
| | | | | -40 to +85 °C | HD6437021VXI | HD6437021VTEI | |
| | PROM | 5.0V | 2 to 20MHz | -20 to +75 °C | HD6477021X | HD6477021TE | |
| | | | 2 to 16.6MHz | -40 to +85 °C | HD6477021XI | HD6477021TEI | |
| | | 3.3V | 2 to 12.5MHz | -20 to +75 °C | HD6477021VX | HD6477021VTE | |
| | | | | -40 to +85 °C | HD6477021VXI | HD6477021VTEI | |
| SH7020 | masked ROM | 5.0V | 2 to 20MHz | -20 to +75 °C | HD6437020X | HD6437020TE | |
| | | | 2 to 16.6MHz | -40 to +85 °C | HD6437020XI | HD6437020TEI | |
| | | 3.3V | 2 to 12.5MHz | -20 to +75 °C | HD6437020VX | HD6437020VTE | |
| | | | | -40 to +85 °C | HD6437020VXI | HD6437020VTEI | |
| | ROMless | 5.0V | 2 to 20MHz | -40 to +85 °C | HD6417020SX20I | HD6417020X20I | |
| | | 3.3V | 2 to 12.5MHz | -40 to +85 °C | HD6417020SVX12I | HD6417020VX12I | |

## 1.2 Block Diagram

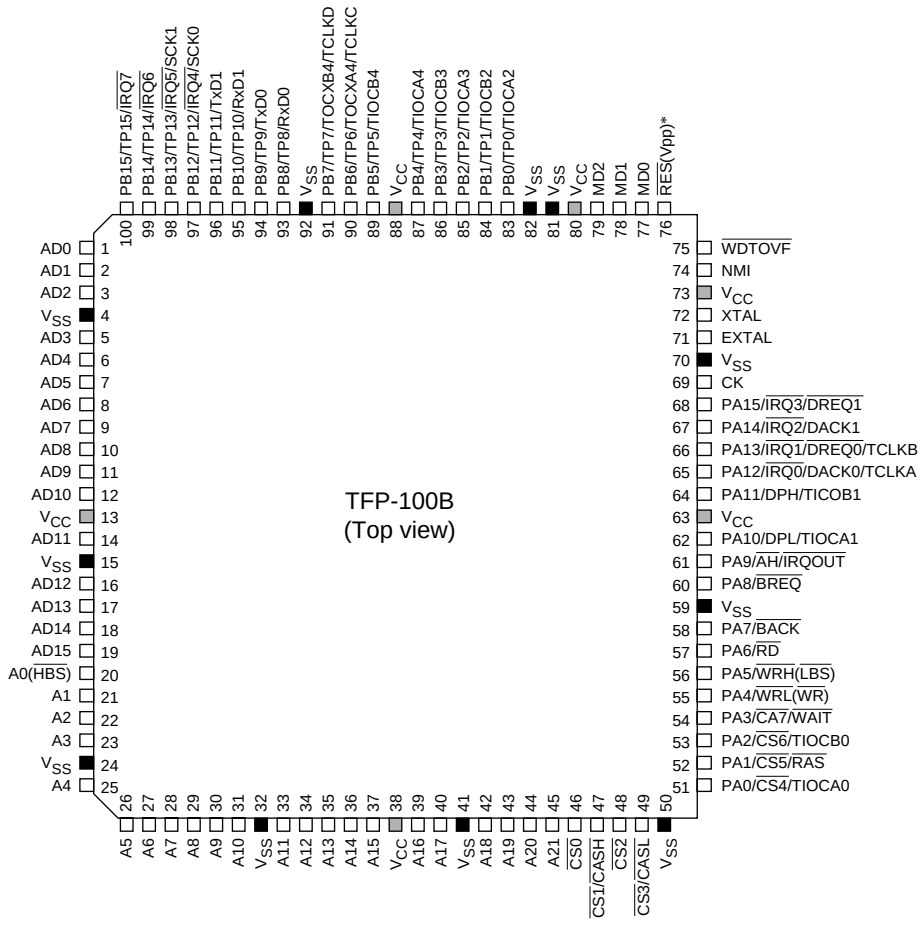



Notes: *1. SH7020: 16-kbyte masked ROM and 1-kbyte RAM.
SH7021: 32-kbyte PROM or Masked ROM and 1-kbyte RAM.
*2. Vpp: SH7021 (PROM version)

**Figure 1.1 Block Diagram**

## 1.3 Pin Descriptions

### 1.3.1 Pin Arrangement




Notes: Vpp: SH7021 (PROM version)

**Figure 1.2 Pin Arrangement**

### 1.3.2 Pin Functions

Table 1.3 describes the pin functions.

**Table 1.3 Pin Functions**

| Type | Symbol | Pin No. | I/O | Name and Function |
|---|---|---|---|---|
| Power | $V_{CC}$ | 13, 38, 63, 73, 80, 88 | I | Power: Connected to the power supply. Connect all $V_{CC}$ pins to the system power supply . The chip will not operate if any $V_{CC}$ pin is left unconnected. |
| | $V_{SS}$ | 4, 15, 24, 32, 41, 50, 59, 70, 81, 82, 92 | I | Ground: Connected to ground. Connect all $V_{SS}$ pins to the system ground. The chip will not operate if any $V_{SS}$ pin is left unconnected. |
| | $V_{PP}$ | 76* | I | RES pin in the MCU mode. Apply +12.5V when programming the PROM in the SH7021 (PROM version). |
| Clock | EXTAL | 71 | I | Crystal/external clock: Connected to a crystal resonator or external clock input having the same frequency as the system clock (CK). |
| | XTAL | 72 | I | Crystal: Connected to a crystal resonator with the same frequency as the system clock (CK). If an external clock is input at the EXTAL pin, leave XTAL open. |
| | CK | 69 | O | System clock: Supplies the system clock (CK) to peripheral devices. |
| System control | RES | 76 | I | Reset: Low input causes a power-on reset if NMI is high, or a manual reset if NMI is low. |
| | WDTOVF | 75 | O | Watchdog timer overflow: Overflow output signal from the watchdog timer. |
| | BREQ | 60 | I | Bus request: Driven low by an external device to request the bus ownership. |
| | BACK | 58 | O | Bus request acknowledge: Indicates that bus ownership has been granted to an external device. By receiving the BACK signal, a device that has sent a BREQ signal can confirm that it has been granted the bus. |

Note: Pin 76 is RES in the SH7020, SH7021 (Masked ROM version) and Vpp in the SH7021 (PROM version).

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. | I/O | Name and Function |
|---|---|---|---|---|
| Operating mode control | MD2, MD1, MD0 | 79–77 | I | Mode select: Selects the operating mode. Do not change these inputs while the chip is operating. The following table lists the possible operating modes and their corresponding MD2–MD0 values. |

| MD2 | MD1 | MD0 | Operating Mode | On-chip ROM | Bus Size in Area 0 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | MCU mode | Disabled | 8 bits |
| 0 | 0 | 1 | | | 16 bits |
| 0 | 1 | 0 | | Enabled*[1] | |
| 0 | 1 | 1 | (Reserved) | | |
| 1 | 0 | 0 | | | |
| 1 | 0 | 1 | | | |
| 1 | 1 | 0 | | | |
| 1 | 1 | 1 | PROM mode*[2] | | |

| Type | Symbol | Pin No. | I/O | Name and Function |
|---|---|---|---|---|
| Interrupts | NMI | 74 | I | Nonmaskable interrupt: Nonmaskable interrupt request signal. The rising or falling edge can be selected for signal detection. |
| | IRQ0–IRQ7 | 65–68, 97–100 | I | Interrupt request 0–7: Maskable interrupt request signals. Level input or edge-triggered input can be selected. |
| | IRQOUT | 61 | O | Slave interrupt request output: Indicates occurrence of an interrupt while the bus is released. |
| Address bus | A21–A0 | 45–42, 40, 39, 37–33, 31–25, 23–20 | O | Address bus: Outputs addresses. |
| Data bus | AD15–AD0 | 19–16, 14, 12-5, 3–1 | I/O | Data bus: 16-bit bidirectional data bus that is multiplexed with the lower 16 bits of the address bus. |
| | DPH | 64 | I/O | Upper data bus parity: Parity data for D15–D8. |
| | DPL | 62 | I/O | Lower data bus parity: Parity data for D7–D0. |
| Bus control | WAIT | 54 | I | Wait: Requests the insertion of wait states ($T_W$) into the bus cycle when the external address space is accessed. |

Notes : 1.Use prohibited in the SH7020 Romless version.

2.Can only be used in the SH7021 ZTAT version.

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. | I/O | Name and Function |
|---|---|---|---|---|
| Bus control (cont) | RAS | 52 | O | Row address strobe: DRAM row-address strobe-timing signal. |
| | CASH | 47 | O | Column address strobe high: DRAM column-address strobe-timing signal outputs low level to access the upper eight data bits. |
| | CASL | 49 | O | Column address strobe low: DRAM column-address strobe-timing signal outputs low level to access the lower eight data bits. |
| | RD | 57 | O | Read: Indicates reading of data from an external device. |
| | WRH | 56 | O | Upper write: Indicates write access to the upper eight bits of an external device. |
| | WRL | 55 | O | Lower write: Indicates write access to the lower eight bits of an external device. |
| | CS0–CS7 | 46–49, 51–54 | O | Chip select 0–7: Chip select signals for accessing external memory and devices. |
| | AH | 61 | O | Address hold: Address hold timing signal for a device using a multiplexed address/data bus. |
| | HBS, LBS | 20, 56 | O | Upper/lower byte strobe: Upper and lower byte strobe signals. (Also used as WRH and A0.) |
| | WR | 55 | O | Write: Brought low during write access. (Also used as WRL.) |
| DMAC | DREQ0, DREQ1 | 66, 68 | I | DMA transfer request (channels 0 and 1): Input pins for external DMA transfer requests. |
| | DACK0, DACK1 | 65, 67 | O | DMA transfer acknowledge (channels 0 and 1): Indicates that DMA transfer is acknowledged. |
| 16-bit integrated-timer pulse unit (ITU) | TIOCA0, TIOCB0 | 51, 53 | I/O | ITU input capture/output compare (channel 0): Input capture or output compare pins. |
| | TIOCA1, TIOCB1 | 62, 64 | I/O | ITU input capture/output compare (channel 1): Input capture or output compare pins. |
| | TIOCA2, TIOCB2 | 83, 84 | I/O | ITU input capture/output compare (channel 2): Input capture or output compare pins. |
| | TIOCA3, TIOCB3 | 85, 86 | I/O | ITU input capture/output compare (channel 3): Input capture or output compare pins. |
| | TIOCA4, TIOCB4 | 87, 89 | I/O | ITU input capture/output compare (channel 4): Input capture or output compare pins. |

**Table 1.3 Pin Functions (cont)**

| Type | Symbol | Pin No. | I/O | Name and Function |
|---|---|---|---|---|
| 16-bit integrated-timer pulse unit (ITU) | TOCXA4, TOCXB4 | 90, 91 | O | ITU output compare (channel 4): Output compare pins. |
| | TCLKA–TCLKD | 65, 66, 90, 91 | I | ITU timer clock input: External clock input pins for ITU counters. |
| Timing pattern controller (TPC) | TP15–TP0 | 100–93, 91–89, 87–83 | O | Timing pattern output 15–0: Timing pattern output pins. |
| Serial com-munication interface (SCI) | TxD0, TxD1 | 94, 96 | O | Transmit data (channels 0 and 1): Transmit data output pins for SCI0 and SCI1. |
| | RxD0, RxD1 | 93, 95 | I | Receive data (channels 0 and 1): Receive data input pins for SCI0 and SCI1. |
| | SCK0, SCK1 | 97, 98 | I/O | Serial clock (channels 0 and 1): Clock input/output pins for SCI0 and SCI1. |
| I/O ports | PA15–PA0 | 68–64, 62–60, 58-51 | I/O | Port A: 16-bit input/output pins. Input or output can be selected individually for each bit. |
| | PB15–PB0 | 100–93, 91–89, 87–83 | I/O | Port B: 16-bit input/output pins. Input or output can be selected individually for each bit. |

### 1.3.3 Pin Layout by Mode

Table 1.4 shows pin layout by mode

**Table 1.4 Pin Layout by Mode**

| Pin No. | MCU Mode | PROM Mode (SH7021PR-OM version) | Pin No. | MCU Mode | PROM Mode (SH7021PR-OM version) |
|---|---|---|---|---|---|
| 1 | AD0 | AD0 | 29 | A8 | A8 |
| 2 | AD1 | AD1 | 30 | A9 | OE |
| 3 | AD2 | AD2 | 31 | A10 | A10 |
| 4 | Vss | Vss | 32 | $V_{SS}$ | $V_{SS}$ |
| 5 | AD3 | AD3 | 33 | A11 | A11 |
| 6 | AD4 | AD4 | 34 | A12 | A12 |
| 7 | AD5 | AD5 | 35 | A13 | A13 |
| 8 | AD6 | AD6 | 36 | A14 | A14 |
| 9 | AD7 | AD7 | 37 | A15 | A15 |
| 10 | AD8 | NC | 38 | $V_{CC}$ | $V_{CC}$ |
| 11 | AD9 | NC | 39 | A16 | A16 |
| 12 | AD10 | NC | 40 | A17 | $V_{CC}$ |
| 13 | $V_{CC}$ | $V_{CC}$ | 41 | $V_{SS}$ | $V_{SS}$ |
| 14 | AD11 | NC | 42 | A18 | $V_{CC}$ |
| 15 | $V_{SS}$ | $V_{SS}$ | 43 | A19 | NC |
| 16 | AD12 | NC | 44 | A20 | NC |
| 17 | AD13 | NC | 45 | A21 | NC |
| 18 | AD14 | NC | 46 | CS0 | NC |
| 19 | AD15 | NC | 47 | CS1/CASH | NC |
| 20 | A0(HBS) | A0 | 48 | CS2 | NC |
| 21 | A1 | A1 | 49 | CS3/CASL | NC |
| 22 | A2 | A2 | 50 | $V_{SS}$ | $V_{SS}$ |
| 23 | A3 | A3 | 51 | PA0/CS4/TIOCA0 | NC |
| 24 | $V_{SS}$ | $V_{SS}$ | 52 | PA1/CS5/RAS | NC |
| 25 | A4 | A4 | 53 | PA2/CS6/ TIOCB0 | PGM |
| 26 | A5 | A5 | 54 | PA3/CS7/WAIT | CE |
| 27 | A6 | A6 | 55 | PA4/WRL(WR) | NC |
| 28 | A7 | A7 | 56 | PA5/WRH(LBS) | NC |

**Table 1.3.3 Pin Layout by Mode (cont)**

| Pin No. | MCU Mode | PROM Mode (SH7021PR-OM version) |
|---|---|---|
| 57 | PA6/RD | NC |
| 58 | PA7/BACK | NC |
| 59 | $V_{SS}$ | $V_{SS}$ |
| 60 | PA8/BREQ | NC |
| 61 | PA9/AH/IRQOUT | NC |
| 62 | PA10/DPL/TIOCA1 | NC |
| 63 | $V_{CC}$ | $V_{CC}$ |
| 64 | PA11/DPH/TIOCB1 | NC |
| 65 | PA12/IRQ0/DACK0/ TCLKA | NC |
| 66 | PA13/IRQ1/DREQ0/ TCLKLB | NC |
| 67 | PA14/IRQ2/DACK1 | NC |
| 68 | PA15/IRQ3/DREQ1 | NC |
| 69 | CK | NC |
| 70 | $V_{SS}$ | $V_{SS}$ |
| 71 | EXTAL | NC |
| 72 | XTAL | NC |
| 73 | $V_{CC}$ | $V_{CC}$ |
| 74 | NMI | A9 |
| 75 | WDTOVF | NC |
| 76 | RES | $V_{pp}$ |
| 77 | MD0 | $V_{CC}$ |
| 78 | MD1 | $V_{CC}$ |
| 79 | MD2 | $V_{CC}$ |
| 80 | $V_{CC}$ | $V_{CC}$ |
| 81 | $V_{SS}$ | $V_{SS}$ |
| 82 | $V_{SS}$ | $V_{SS}$ |
| 83 | PB0/TP0/TIOCA2 | NC |
| 84 | PB1/TP1/TIOCB2 | NC |
| 85 | PB2/TP2/TIOCA3 | NC |
| 86 | PB3/TP3/TIOCB3 | NC |
| 87 | PB4/TP4/TIOCA4 | NC |
| 88 | $V_{CC}$ | $V_{CC}$ |
| 89 | PB5/TP5/TIOCB4 | NC |
| 90 | PB6/TP6/TOCXA4/ TCLKC | NC |
| 91 | PB7/TP7/TOCXB4/ TCLKD | NC |
| 92 | $V_{SS}$ | $V_{SS}$ |
| 93 | PB8/TP8/RxD0 | NC |
| 94 | PB9/TP9/TxD0 | NC |
| 95 | PB10/TP10/RxD1 | NC |
| 96 | PB11/TP11/TXD1 | NC |
| 97 | PB12/TP12/IRQ4/ SCK0 | NC |
| 98 | PB13/TP13/IRQ5/ SCK1 | NC |
| 99 | PB14/TP14/IRQ6 | NC |
| 100 | PB15/TP15/IRQ7 | NC |

## 2.1 Register Configuration

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

### 2.1.1 General Registers (Rn)

General registers Rn consist of sixteen 32-bit registers (R0–R15). General registers are used for data processing and address calculation. Register R0 also functions as an index register. For some instructions, the R0 register must be used. Register R15 functions as a stack pointer to save or recover status registers (SR) and program counter (PC) during exception processing.

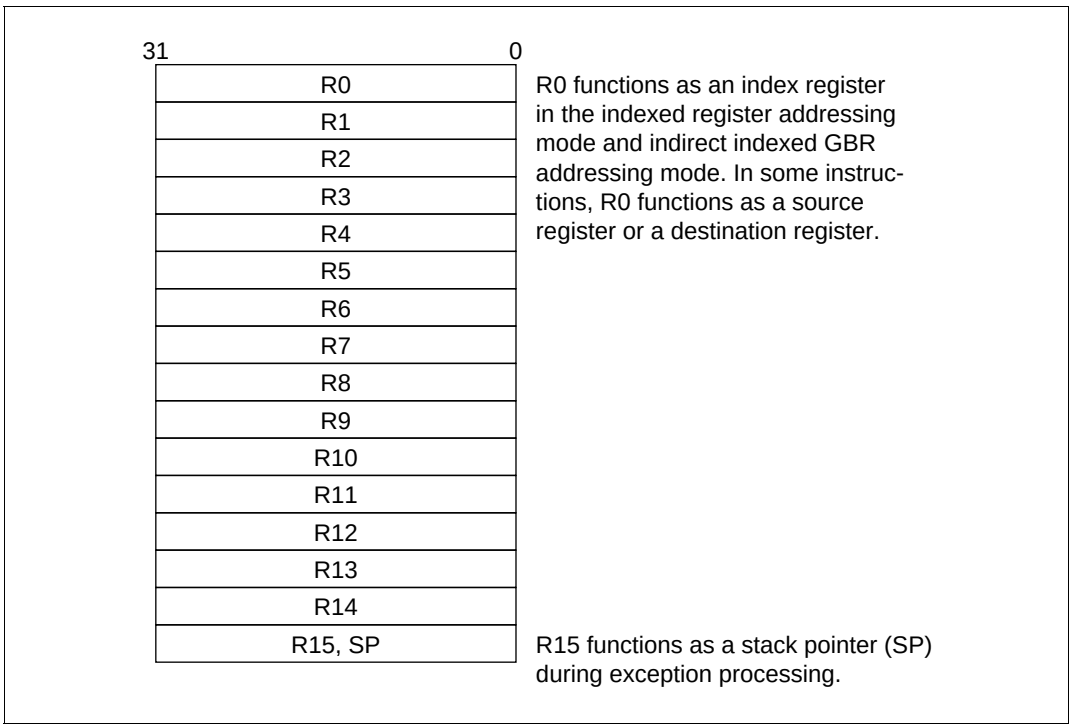



**Figure 2.1 General Registers (Rn)**

### 2.1.2 Control Registers

Control registers consist of the 32-bit status register (SR), global base register (GBR), and vector base register (VBR). The status register indicates processing states. The global base register

functions as a base address for the indirect GBR addressing mode to transfer data to the registers of peripheral on-chip modules. The vector base register functions as the base address of the exception processing vector area including interrupts.

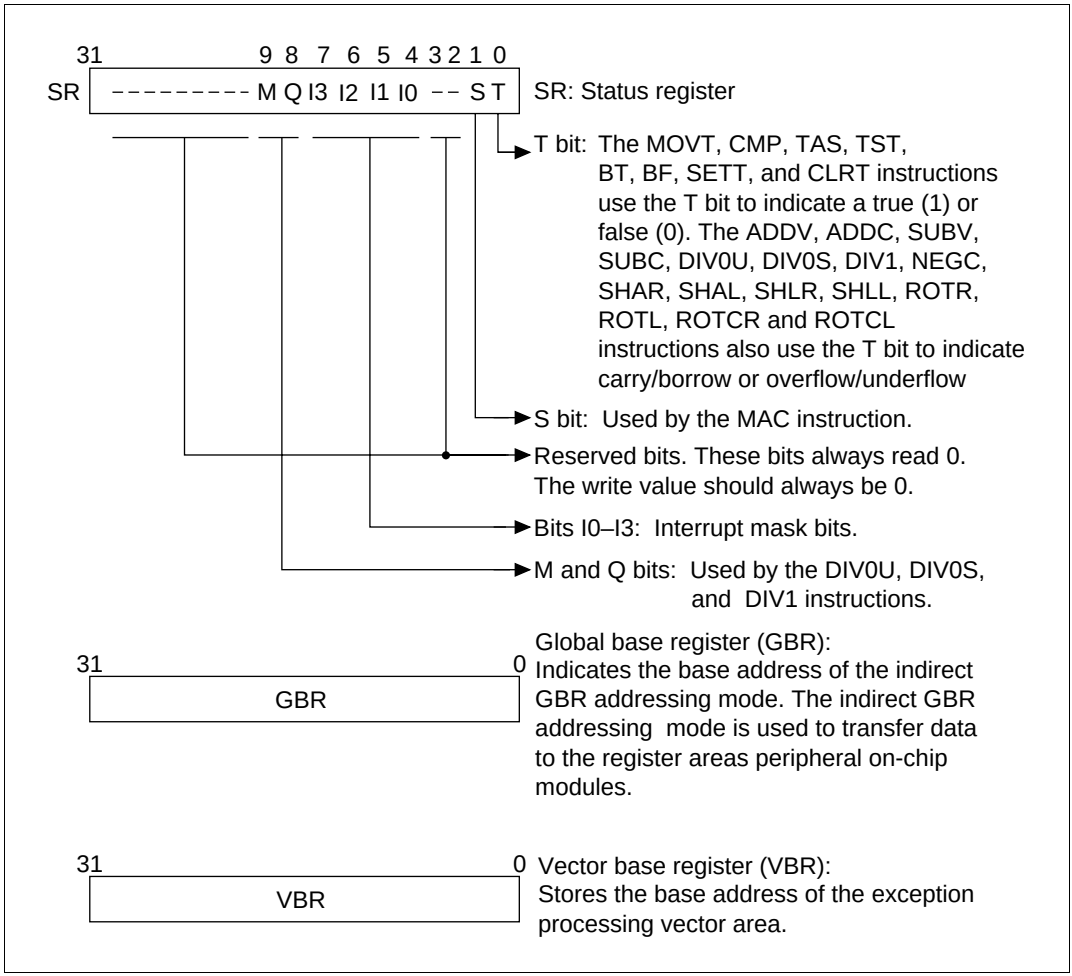



**Figure 2.2 Control Registers**

### 2.1.3 System Registers

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

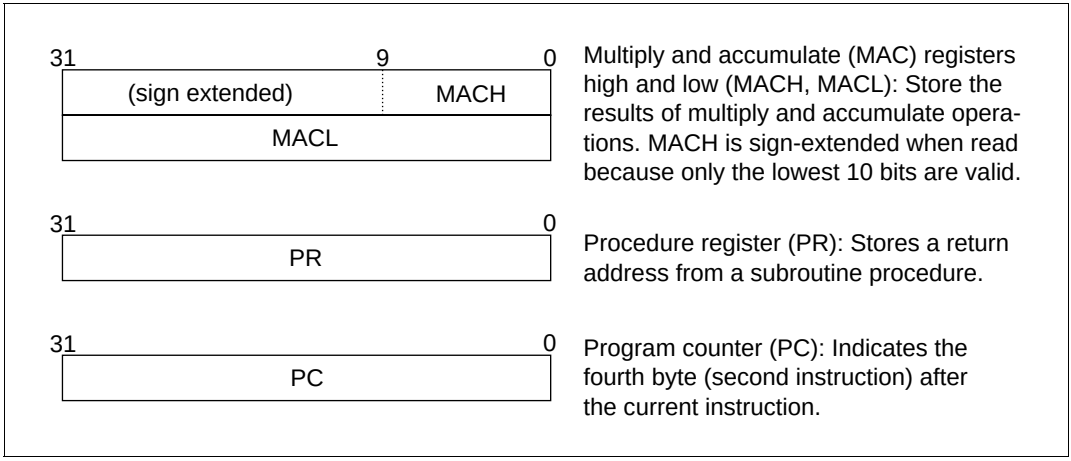



**Figure 2.3 System Registers**

### 2.1.4 Initial Values of Registers

Table 2.1 lists the values of the registers after reset.

**Table 2.1 Initial Values of Registers**

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

## 2.2 Data Formats

### 2.2.1 Data Format in Registers

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

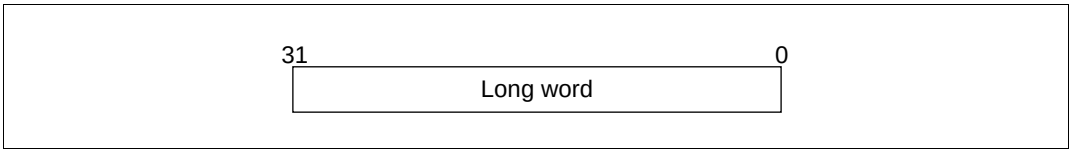



**Figure 2.4 Data Format in Registers**

### 2.2.2 Data Format in Memory

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

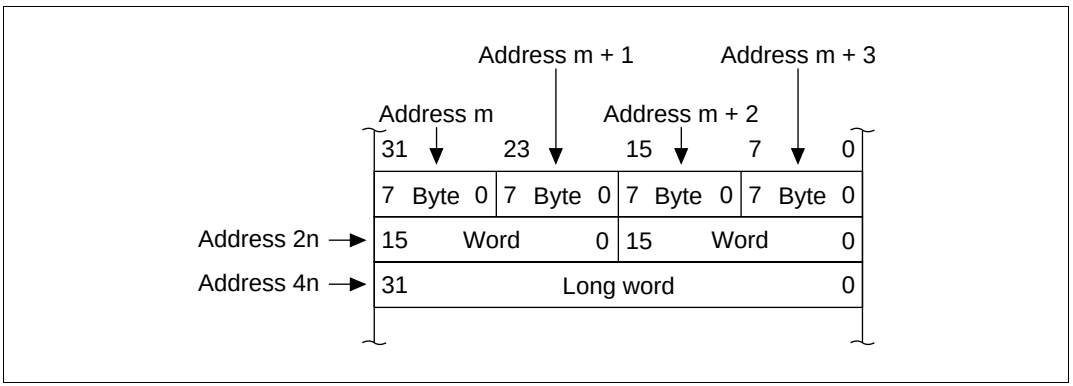



**Figure 2.5 Data Format in Memory**

### 2.2.3 Immediate Data Format

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

Word or long word immediate data is not located in the instruction code but rather is stored in a memory table. The memory table is accessed by a immediate data transfer instruction (MOV) using the PC relative addressing mode with displacement.

## 2.3 Instruction Features

### 2.3.1 RISC-Type Instruction Set

All instructions are RISC type. Their features are as follows:

**16-Bit Fixed Length:** Every instruction is 16 bits long, making program coding much more efficient.

**One Instruction/Cycle:** Basic instructions can be executed in one cycle using the pipeline system. One-cycle instructions are executed in 50 ns at 20 MHz.

**Data Length:** Long word is the standard data length for all operations. Memory can be accessed in bytes, words, or long words. Byte or word data accessed from memory is sign-extended and handled as long word data. Immediate data is sign-extended for arithmetic operations or zero-extended for logic operations (handled as long word data).

**Table 2.2 Sign Extension of Word Data**

| CPU of SH7000 Series | Description | Conventional CPUs |
|---|---|---|
| `MOV.W @(disp,PC),R1`<br>`ADD R1,R0`<br>`.....`<br>`.DATA.W H'1234` | Data is sign-extended to 32 bits, and R1 becomes H'00001234. It is next operated upon by an ADD instruction. | `ADD.W #H'1234, R0` |

Note: The address of the immediate data is accessed by @(disp, PC).

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

**Delayed Branch Instructions:** Unconditional branch instructions are delayed. Pipeline disruption during branching is reduced by first executing the instruction that follows the branch instruction, and then branching. See the SH-1/SH-2 Programming Manual for details.

**Table 2.3 Delayed Branch Instructions**

| CPU of SH7000 Series | Description | Conventional CPU |
|---|---|---|
| `BRA TRGET`<br>`ADD R1, R0` | Executes an ADD before branching to TRGET. | `ADD.W R1, R0`<br>`BRA TRGET` |

**Multiplication/Accumulation Operation:** The five-stage pipeline system and the on-chip multiplier enable 16-bit × 16-bit → 32-bit multiplication operations to be executed in 1–3 cycles. 16-bit × 16-bit + 42-bit → 42-bit multiplication/accumulation operations can be executed in 2–3

cycles.

**T bit:** T bit (in the status register) is set according to the result of a comparison, and in turn is the condition (True/False) that determines if the program will branch. The T bit in the status register is only changed by selected instructions, thus improving the processing speed.

**Table 2.4 T bit**

| CPU of SH7000 Series | Description | Conventional CPU |
|---|---|---|
| CMP/GE R1, R0<br>BT TRGET0<br>BF TRGET1 | T bit is set when R0 ≥ R1. The program branches to TRGET0 when R0 ≥ R1 and to TRGET1 when R0<R1. | CMP.W R1, R0<br>BGE TRGET0<br>BLT TRGET1 |
| ADD #–1, R0<br>TST R0, R0<br>BT TRGET | T bit is not changed by ADD. T bit is set when R0=0. The program branches if R0=0. | SUB.W #1, R0<br>BEQ TRGET |

**Immediate Data:** Byte (8-bit) immediate data is located in the instruction code. Word or long word immediate data is not located in instruction codes but is stored in a memory table. The memory table is accessed by a immediate data transfer instruction (MOV) using the PC relative addressing mode with displacement.

**Table 2.5 Immediate Data Accessing**

| Classification | CPU of SH7000 Series | Conventional CPU |
|---|---|---|
| 8-bit immediate | MOV #H'12, R0 | MOV.B #H'12, R0 |
| 16-bit immediate | MOV.W @(disp,PC), R0<br>.....<br>.DATA.W H'1234 | MOV.W #H'1234, R0 |
| 32-bit immediate | MOV.L @(disp,PC), R0<br>.....<br>.DATA.L H'12345678 | MOV.L #H'12345678, R0 |

Note: The address of the immediate data is accessed by @(disp, PC).

**Absolute Address:** When data is accessed by absolute address, the value already in the absolute address is placed in the memory table. By loading the immediate data when the instruction is executed, that value is transferred to the register and the data is accessed in the indirect register addressing mode.

**Table 2.6 Absolute Address Accessing**

| Classification | CPU of SH7000 Series | Conventional CPU |
|---|---|---|
| Absolute address | `MOV.L @(disp,PC), R1`<br>`MOV. B @R1, R0`<br>`.........`<br>`.DATA.L H'12345678` | `MOV.B @H'12345678, R0` |

Note: The address of the immediate data is accessed by @(disp, PC).

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

**Table 2.7 Accessing by Displacement**

| Classification | CPU of SH7000 Series | Conventional CPU |
|---|---|---|
| 16-bit displacement | `MOV.W @(disp, PC), R0`<br>`MOV.W @(R0, R1), R2`<br>`.........`<br>`.DATA.W H'1234` | `MOV.W @(H'1234, R1), R2` |

Note: The address of the immediate data is accessed by @(disp, PC).

### 2.3.2 Addressing Modes

Addressing modes and effective address calculation are described in table 2.8.

**Table 2.8 Addressing Modes and Effective Addresses**

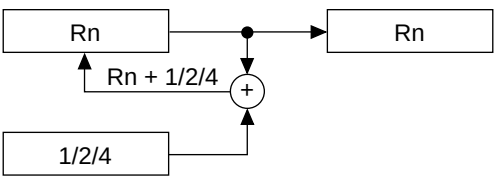
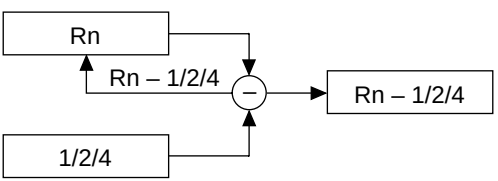
| Addressing Mode | Mnemonic Expression | Effective Addresses Calculation | Equation |
|---|---|---|---|
| Direct register addressing | Rn | The effective address is register Rn. (The operand is the contents of register Rn.) | — |
| Indirect register addressing | @Rn | The effective address is the content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

**Table 2.8 Addressing Modes and Effective Addresses (cont)**

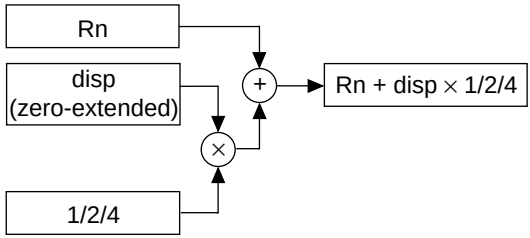
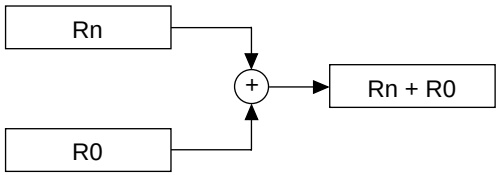
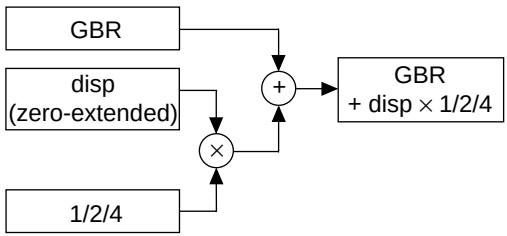
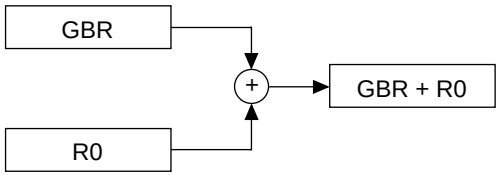
| Addressing Mode | Mnemonic Expression | Effective Addresses Calculation | Equation |
|---|---|---|---|
| Indirect register addressing with displace-ment | @(disp:4, Rn) | The effective address is Rn plus a 4-bit displacement (disp). disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a long word operation.<br>  | Byte: Rn + disp<br>Word: Rn + disp × 2<br>Long word: Rn + disp × 4 |
| Indirect indexed register addressing | @(R0, Rn) | The effective address is the Rn value plus R0.<br>  | Rn + R0 |
| Indirect GBR addressing with displace-ment | @(disp:8, GBR) | The effective address is the GBR value plus an 8-bit displacement (disp). The value of disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a long word operation.<br>  | Byte: GBR + disp<br>Word: GBR + disp × 2<br>Long word: GBR + disp × 4 |
| Indirect indexed GBR addressing | @(R0, GBR) | The effective address is the GBR value plus the R0.<br>  | GBR + R0 |

**Table 2.8 Addressing Modes and Effective Addresses (cont)**

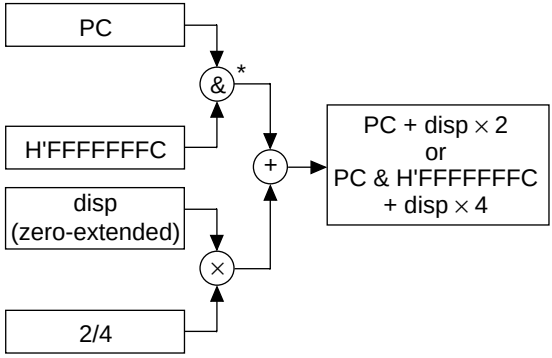
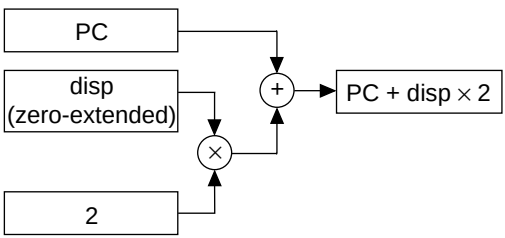
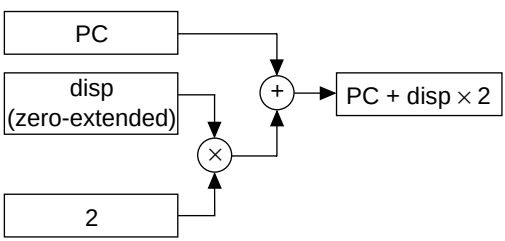
| Addressing Mode | Mnemonic Expression | Effective Addresses Calculation | Equation |
|---|---|---|---|
| PC relative addressing with displacement | @(disp:8, PC) | The effective address is the PC value plus an 8-bit displacement (disp). disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a long word operation. For a long word operation, the lowest two bits of the PC are masked.<br>  <br><br>*: For long word | Word: PC + disp × 2<br>Long word: PC & H'FFFFFFFC + disp × 4 |
| PC relative addressing | disp:8 | The effective address is the PC value sign-extended with an 8-bit displacement (disp), doubled, and added to the PC.<br>  <br> | PC + disp × 2 |
| | disp:12 | The effective address is the PC value sign-extended with a 12-bit displacement (disp), doubled, and added to the PC.<br>  <br> | PC + disp × 2 |

**Table 2.8 Addressing Modes and Effective Addresses (cont)**

| Addressing Mode | Mnemonic Expression | Effective Addresses Calculation | Equation |
|---|---|---|---|
| Immediate addressing | #imm:8 | The 8-bit immediate data (imm) for the TST, AND, OR, and XOR instructions are zero-extended. | — |
| | #imm:8 | The 8-bit immediate data (imm) for the MOV, ADD, and CMP/EQ instructions are sign-extended. | — |
| | #imm:8 | Immediate data (imm) for the TRAPA instruction is zero-extended and is quadrupled. | — |

### 2.3.3 Instruction Formats

The instruction format refers to the source operand and the destination operand. The meaning of the operand depends on the instruction code. Symbols are as follows.

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

**Table 2.9 Instruction Formats**

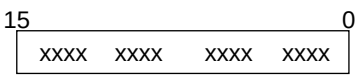
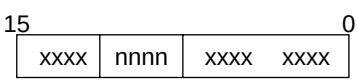
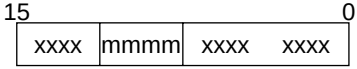
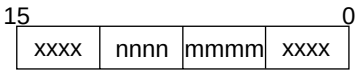
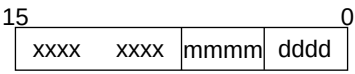
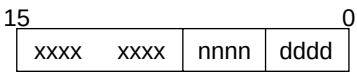
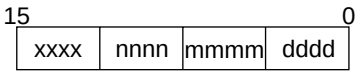
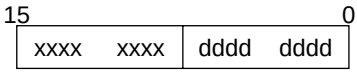
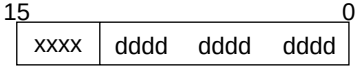
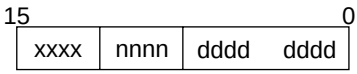
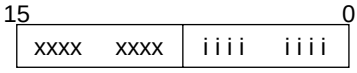
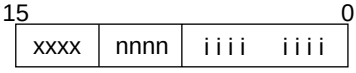
| Instruction Formats | Source Operand | Destination Operand | Instruction Example |
|---|---|---|---|
| 0 format<br>  | — | — | `NOP` |
| n format<br>  | — | nnnn: Direct register | `MOVT Rn` |
| | Control register or system register | nnnn: Direct register | `STS MACH,Rn` |
| | Control register or system register | nnnn: Indirect pre-decrement register | `STC.L SR,@-Rn` |

**Table 2.9 Instruction Formats (cont)**

| Instruction Formats | Source Operand | Destination Operand | Instruction Example |
|---|---|---|---|
| m format<br>  | mmmm: Direct register | Control register or system register | `LDC Rm,SR` |
| | mmmm: Indirect post-increment register | Control register or system register | `LDC.L @Rm+,SR` |
| | mmmm: Direct register | — | `JMP @Rm` |
| nm format<br>  | mmmm: Direct register | nnnn: Direct register | `ADD Rm,Rn` |
| | mmmm: Direct register | nnnn: Direct register | `MOV.L Rm,@Rn` |
| | mmmm: Indirect post-increment register (multiply/ accumulate)<br>nnnn: Indirect post-increment register (multiply/ accumulate)* | MACH, MACL | `MAC.W @Rm+,@Rn+` |
| | mmmm: Indirect post-increment register | nnnn: Direct register | `MOV.L @Rm+,Rn` |
| | mmmm: Direct register | nnnn: Indirect pre-decrement register | `MOV.L Rm,@-Rn` |
| | mmmm: Direct register | nnnn: Indirect indexed register | `MOV.L Rm,@(R0,Rn)` |
| md format<br>  | mmmmdddd: indirect register with displacement | R0 (Direct register) | `MOV.B @(disp,Rm),R0` |
| nd4 format<br>  | R0 (Direct register) | nnnndddd: Indirect register with displacement | `MOV.B R0,@(disp,Rn)` |

Note: In MAC instructions, nnnn is the source register.

**Table 2.9 Instruction Formats (cont)**

| Instruction Formats | Source Operand | Destination Operand | Example |
|---|---|---|---|
| nmd format<br>  | mmmm: Direct register | nnnndddd: Indirect register with displacement | `MOV.L Rm,@(disp,Rn)` |
| | mmmmdddd: Indirect register with displacement | nnnn: Direct register | `MOV.L @(disp,Rm),Rn` |
| d format<br>  | dddddddd: Indirect GBR with displacement | R0 (Direct register) | `MOV.L @(disp,GBR),R0` |
| | R0(Direct register) | dddddddd: Indirect GBR with displacement | `MOV.L R0,@(disp,GBR)` |
| | dddddddd: PC relative with displacement | R0 (Direct register) | `MOVA @(disp,PC),R0` |
| | dddddddd: PC relative | — | `BF disp` |
| d12 format<br>  | ddddddddddddd: PC relative | — | `BRA disp` |
| nd8 format<br>  | dddddddd: PC relative with displacement | nnnn: Direct register | `MOV.L @(disp,PC),Rn` |
| i format<br>  | iiiiiiii: Immediate | Indirect indexed GBR | `AND.B #imm,@(R0,GBR)` |
| | iiiiiiii: Immediate | R0 (Direct register) | `AND #imm,R0` |
| | iiiiiiii: Immediate | — | `TRAPA #imm` |
| ni format<br>  | iiiiiiii: Immediate | nnnn: Direct register | `ADD #imm,Rn` |

## 2.4 Instruction Set

### 2.4.1 Instruction Set by Classification

Table 2.10 lists instructions by classification.

**Table 2.10 Classification of Instructions**

| Classification | Types | Operation Code | Function | Number of Instructions |
|---|---|---|---|---|
| Data transfer | 5 | MOV | Data transfer, immediate data transfer, peripheral module data transfer, structure data transfer | 39 |
| | | MOVA | Effective address transfer | |
| | | MOVT | T bit transfer | |
| | | SWAP | Swap of upper and lower bytes | |
| | | XTRCT | Extraction of the middle of registers connected | |
| Arithmetic operations | 17 | ADD | Binary addition | 28 |
| | | ADDC | Binary addition with carry | |
| | | ADDV | Binary addition with overflow check | |
| | | CMP/cond | Comparison | |
| | | DIV1 | Division | |
| | | DIV0S | Initialization of signed division | |
| | | DIV0U | Initialization of unsigned division | |
| | | EXTS | Sign extension | |
| | | EXTU | Zero extension | |
| | | MAC | Multiplication and accumulation | |
| | | MULS | Signed multiplication | |
| | | MULU | Unsigned multiplication | |
| | | NEG | Negation | |
| | | NEGC | Negation with borrow | |
| | | SUB | Binary subtraction | |
| | | SUBC | Binary subtraction with carry | |
| | | SUBV | Binary subtraction with underflow check | |
| Logic operations | 6 | AND | Logical AND | 14 |
| | | NOT | Bit inversion | |
| | | OR | Logical OR | |
| | | TAS | Memory test and bit set | |

**Table 2.10 Classification of Instructions (cont)**

| Classification | Types | Operation Code | Function | Number of Instructions |
|---|---|---|---|---|
| Logic operations(cont) | 6 | TST | Logical AND and T bit set | 14 |
| | | XOR | Exclusive OR | |
| Shift | 10 | ROTL | One-bit left rotation | 14 |
| | | ROTR | One-bit right rotation | |
| | | ROTCL | One-bit left rotation with T bit | |
| | | ROTCR | One-bit right rotation with T bit | |
| | | SHAL | One-bit arithmetic left shift | |
| | | SHAR | One-bit arithmetic right shift | |
| | | SHLL | One-bit logical left shift | |
| | | SHLLn | n-bit logical left shift | |
| | | SHLR | One-bit logical right shift | |
| | | SHLRn | n-bit logical right shift | |
| Branch | 7 | BF | Conditional branch (T = 0) | 7 |
| | | BT | Conditional branch (T = 1) | |
| | | BRA | Unconditional branch | |
| | | BSR | Branch to subroutine procedure | |
| | | JMP | Unconditional branch | |
| | | JSR | Branch to subroutine procedure | |
| | | RTS | Return from subroutine procedure | |
| System control | 11 | CLRT | T bit clear | 31 |
| | | CLRMAC | MAC register clear | |
| | | LDC | Load to control register | |
| | | LDS | Load to system register | |
| | | NOP | No operation | |
| | | RTE | Return from exception processing | |
| | | SETT | T bit set | |
| | | SLEEP | Shift into power-down mode | |
| | | STC | Storing control register data | |
| | | STS | Storing system register data | |
| | | TRAPA | Trap exception processing | |
| Total | 56 | | | 133 |

Instruction codes, operation, and execution states are listed in the following format in order by classification.

**Table 2.11 Instruction Code Format**

| Item | Format | Explanation |
|---|---|---|
| Instruction mnemonic | `OP.Sz SRC,DEST` | OP: Operation code<br>Sz: Size<br>SRC: Source<br>DEST: Destination<br>Rm: Source register<br>Rn: Destination register<br>imm: Immediate data<br>disp: Displacement* |
| Instruction code | MSB ↔ LSB | mmmm: Source register<br>nnnn: Destination register<br>0000: R0<br>0001: R1<br>.....<br>1111: R15<br>iiii: Immediate data<br>dddd: Displacement |
| Operation summary | →, ←<br>(xx)<br>M/Q/T<br>&<br>\|<br>^<br>~<br><<n, >>n | Direction of transfer<br>Memory operand<br>Flag bits in the SR<br>Logical AND of each bit<br>Logical OR of each bit<br>Exclusive OR of each bit<br>Logical NOT of each bit<br>n-bit shift |
| Execution cycle | | Value when no wait states are inserted |
| | | Instruction execution cycles: The execution cycles shown in the table are minimums. The actual number of cycles may be increased:<br>1. When contention occurs between instruction fetches and data access, or<br>2. When the destination register of the load instruction (memory → register) and the register used by the next instruction are the same. |
| T bit | | Value of T bit after instruction is executed |
| | — | No change |

Note: Scaling (×1, ×2, ×4) is performed according to the instruction operand size. See "SH-1/SH-2 Programming Manual" for details.

**Table 2.12 Data Transfer Instructions**

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

**Table 2.12 Data Transfer Instructions (cont)**

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

**Table 2.13 Arithmetic Instructions**

| Instruction | | Instruction Code | Operation | Execution Cycles | T bit |
|---|---|---|---|---|---|
| ADD | Rm,Rn | 0011nnnnmmmm1100 | Rn + Rm → Rn | 1 | — |
| ADD | #imm,Rn | 0111nnnniiiiiiii | Rn + imm → Rn | 1 | — |
| ADDC | Rm,Rn | 0011nnnnmmmm1110 | Rn + Rm + T → Rn, Carry → T | 1 | Carry |
| ADDV | Rm,Rn | 0011nnnnmmmm1111 | Rn + Rm → Rn, Overflow → T | 1 | Overflow |
| CMP/EQ | #imm,R0 | 10001000iiiiiiii | If R0 = imm, 1 → T | 1 | Comparison result |
| CMP/EQ | Rm,Rn | 0011nnnnmmmm0000 | If Rn = Rm, 1 → T | 1 | Comparison result |
| CMP/HS | Rm,Rn | 0011nnnnmmmm0010 | If Rn≥Rm with unsigned data, 1 → T | 1 | Comparison result |
| CMP/GE | Rm,Rn | 0011nnnnmmmm0011 | If Rn ≥ Rm with signed data, 1 → T | 1 | Comparison result |
| CMP/HI | Rm,Rn | 0011nnnnmmmm0110 | If Rn > Rm with unsigned data, 1 → T | 1 | Comparison result |
| CMP/GT | Rm,Rn | 0011nnnnmmmm0111 | If Rn > Rm with signed data, 1 → T | 1 | Comparison result |
| CMP/PZ | Rn | 0100nnnn00010001 | If Rn ≥ 0, 1 → T | 1 | Comparison result |
| CMP/PL | Rn | 0100nnnn00010101 | If Rn > 0, 1 → T | 1 | Comparison result |
| CMP/STR | Rm,Rn | 0010nnnnmmmm1100 | If Rn and Rm have an equivalent byte, 1 → T | 1 | Comparison result |
| DIV1 | Rm,Rn | 0011nnnnmmmm0100 | Single-step division (Rn/Rm) | 1 | Calculation result |
| DIV0S | Rm,Rn | 0010nnnnmmmm0111 | MSB of Rn → Q, MSB of Rm → M, M ^ Q → T | 1 | Calculation result |
| DIV0U | | 0000000000011001 | 0 → M/Q/T | 1 | 0 |
| EXTS.B | Rm,Rn | 0110nnnnmmmm1110 | A byte in Rm is sign-extended → Rn | 1 | — |

**Table 2.13 Arithmetic Instructions (cont)**

| Instruction | | Instruction Code | Operation | Execution Cycles | T bit |
|---|---|---|---|---|---|
| EXTS.W | Rm,Rn | 0110nnnnmmmm1111 | A word in Rm is sign-extended → Rn | 1 | — |
| EXTU.B | Rm,Rn | 0110nnnnmmmm1100 | A byte in Rm is zero-extended → Rn | 1 | — |
| EXTU.W | Rm,Rn | 0110nnnnmmmm1101 | A word in Rm is zero-extended → Rn | 1 | — |
| MAC.W | @Rm+,@Rn+ | 0100nnnnmmmm1111 | Signed operation of (Rn) × (Rm) + MAC → MAC | 3/(2)* | — |
| MULS | Rm,Rn | 0010nnnnmmmm1111 | Signed operation of Rn × Rm → MAC | 1–3* | — |
| MULU | Rm,Rn | 0010nnnnmmmm1110 | Unsigned operation of Rn × Rm → MAC | 1–3* | — |
| NEG | Rm,Rn | 0110nnnnmmmm1011 | 0–Rm → Rn | 1 | — |
| NEGC | Rm,Rn | 0110nnnnmmmm1010 | 0–Rm–T → Rn, Borrow → T | 1 | Borrow |
| SUB | Rm,Rn | 0011nnnnmmmm1000 | Rn–Rm → Rn | 1 | — |
| SUBC | Rm,Rn | 0011nnnnmmmm1010 | Rn–Rm–T → Rn, Borrow → T | 1 | Borrow |
| SUBV | Rm,Rn | 0011nnnnmmmm1011 | Rn–Rm → Rn, Underflow → T | 1 | Underflow |

Note: The normal minimum number of execution cycles (The number in parenthesis in the number of cycles when there is contension with preceding/following instructions).

**Table 2.14 Logic Operation Instructions**

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

**Table 2.15 Shift Instructions**

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

**Table 2.16 Branch Instructions**

| Instruction | Instruction Code | Operation | Execution Cycles | T bit |
|---|---|---|---|---|
| BF label | 10001011dddddddd | If T = 0, disp×2 + PC → PC; if T = 1, nop | 3/1* | — |
| BT label | 10001001dddddddd | If T = 1, disp×2 + PC → PC; if T = 0, nop | 3/1* | — |
| BRA label | 1010dddddddddddd | Delayed branch, disp×2 + PC → PC | 2 | — |
| BSR label | 1011dddddddddddd | Delayed branch, PC → PR, disp×2 + PC → PC | 2 | — |
| JMP @Rm | 0100mmmm00101011 | Delayed branch, Rm → PC | 2 | — |
| JSR @Rm | 0100mmmm00001011 | Delayed branch, PC → PR, Rm → PC | 2 | — |
| RTS | 0000000000001011 | Delayed branch, PR → PC | 2 | — |

Note: The execution state is three cycles when program branches, and one cycle when program does not branch.

**Table 2.17 System Control Instructions**

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

Note: The number of execution states before the chip enters the sleep state.

**Table 2.17 System Control Instructions (cont)**

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

Note: Instruction execution cycles: The execution cycles shown in the table are minimums. The actual number of cycles may be increased:

1. When contention occurs between instruction fetches and data access, or
2. When the destination register of the load instruction (memory → register) and the register used by the next instruction are the same.

### 2.4.2 Operation Code Map

Table 2.18 is an operation code map.

**Table 2.18 Operation Code Map**

| Instruction Code | | | | Fx: 0000 | Fx: 0001 | Fx: 0010 | Fx: 0011–1111 |
|---|---|---|---|---|---|---|---|
| MSB | | | LSB | MD: 00 | MD: 01 | MD: 10 | MD: 11 |
| 0000 | Rn | Fx | 0000 | | | | |
| 0000 | Rn | Fx | 0001 | | | | |
| 0000 | Rn | Fx | 0010 | STC SR,Rn | STC GBR,Rn | STC VBR,Rn | |
| 0000 | Rn | Fx | 0011 | | | | |
| 0000 | Rn | Rm | 01MD | MOV.B RM, @(R0,Rn) | MOV.W RM, @(R0,Rn) | MOV.L RM, @(R0,Rn) | |
| 0000 | 0000 | Fx | 1000 | CLRT | SETT | CLRMAC | |
| 0000 | 0000 | Fx | 1001 | NOP | DIVOU | | |
| 0000 | 0000 | Fx | 1010 | | | | |
| 0000 | 0000 | Fx | 1011 | RTS | SLEEP | RTE | |
| 0000 | Rn | Fx | 1000 | | | | |
| 0000 | Rn | Fx | 1001 | | | | |
| 0000 | Rn | Fx | 1010 | STS MACH,Rn | STS MACL,Rn | STS PR,Rn | |
| 0000 | Rn | Rm | 1011 | | | | |
| 0000 | Rn | Rm | 11MD | MOV.B @(R0,Rm),Rn | MOV.W @(R0,Rm),Rn | MOV.L @(R0,Rm),Rn | |
| 0001 | Rn | Rm | disp | MOV.L Rm,@(disp:4,Rn) | | | |
| 0010 | Rn | Rm | 00MD | MOV.B Rm,@Rn | MOV.W Rm,@Rn | MOV.L Rm,@Rn | |
| 0010 | Rn | Rm | 01MD | MOV.B Rm,@-Rn | MOV.W Rm,@-Rn | MOV.L Rm,@-Rn | DIV0S Rm,Rn |
| 0010 | Rn | Rm | 10MD | TST Rm,Rn | AND Rm,Rn | XOR Rm,Rn | OR Rm,Rn |
| 0010 | Rn | Rm | 11MD | CMP/STR Rm,Rn | XTRCT Rm,Rn | MULU Rm,Rn | MULS Rm,Rn |
| 0011 | Rn | Rm | 00MD | CMP/EQ Rm,Rn | | CMP/HS Rm,Rn | CMP/GE Rm,Rn |
| 0011 | Rn | Rm | 01MD | DIV1 Rm,Rn | | CMP/HI Rm,Rn | CMP/GT Rm,Rn |
| 0011 | Rn | Rm | 10MD | SUB Rm,Rn | | SUBC Rm,Rn | SUBV Rm,Rn |
| 0011 | Rn | Rm | 11MD | ADD Rm,Rn | | ADDC Rm,Rn | ADDV Rm,Rn |
| 0100 | Rn | Fx | 0000 | SHLL Rn | | SHAL Rn | |
| 0100 | Rn | Fx | 0001 | SHLR Rn | CMP/PZ Rn | SHAR Rn | |
| 0100 | Rn | Fx | 0010 | STS.L MACH, @–Rn | STS.L MACL, @–Rn | STS.L PR, @–Rn | |

**Table 2.18 Operation Code Map (cont)**

| Instruction Code | | | | Fx: 0000 | Fx: 0001 | Fx: 0010 | Fx: 0011–1111 |
|---|---|---|---|---|---|---|---|
| MSB | | | LSB | MD: 00 | MD: 01 | MD: 10 | MD: 11 |
| 0100 | Rn | Fx | 0011 | STC.L SR,@–Rn | STC.L GBR,@–Rn | STC.L VBR,@–Rn | |
| 0100 | Rn | Fx | 0100 | ROTL Rn | | ROTCL Rn | |
| 0100 | Rn | Fx | 0101 | ROTR Rn | CMP/PL Rn | ROTCR Rn | |
| 0100 | Rm | Fx | 0110 | LDS.L @Rm+,MACH | LDS.L @Rm+,MACL | LDS.L @Rm+,PR | |
| 0100 | Rm | Fx | 0111 | LDC.L @Rm+,SR | LDC.L @Rm+,GBR | LDC.L @Rm+,VBR | |
| 0100 | Rn | Fx | 1000 | SHLL2 Rn | SHLL8 Rn | SHLL16 Rn | |
| 0100 | Rn | Fx | 1001 | SHLR2 Rn | SHLR8 Rn | SHLL16 Rn | |
| 0100 | Rm | Fx | 1010 | LDS Rm,MACH | LDS Rm,MACL | LDS Rm,PR | |
| 0100 | Rm/Rn | Fx | 1011 | JSR @Rm | TAS.B @Rn | JMP @Rm | |
| 0100 | Rm | Fx | 1100 | | | | |
| 0100 | Rm | Fx | 1101 | | | | |
| 0100 | Rn | Fx | 1110 | LDC Rm,Sr | LDC Rm,GBR | LDC Rm,VBR | |
| 0100 | Rn | Rm | 1111 | MAC.W @Rm+,@Rn+ | | | |
| 0101 | Rn | Rm | disp | MOV.L @(disp:4,Rm),Rn | | | |
| 0110 | Rn | Rm | 00MD | MOV.B @Rm,Rn | MOV.W @Rm,Rn | MOV.L @Rm,Rn | MOV Rm,Rn |
| 0110 | Rn | Rm | 01MD | MOV.B @Rm+,Rn | MOV.W @Rm+,Rn | MOV.L @Rm+,Rn | NOT Rm,Rn |
| 0110 | Rn | Rm | 10MD | SWAP.B @Rm+,Rn | SWAP.W @Rm+,Rn | NEGC Rm,Rn | NEG Rm,Rn |
| 0110 | Rn | Rm | 11MD | EXTU.B Rm,Rn | EXTU.W Rm,Rn | EXTS.B Rm,Rn | EXTS.W Rm,Rn |
| 0111 | Rn | imm | | ADD #imm:8,Rn | | | |
| 1000 | 00MD | Rn | disp | MOV.B R0, @(disp:4,Rn) | MOV.W R0, @(disp:4,Rn) | | |
| 1000 | 01MD | Rm | disp | MOV.B @(disp:4, Rm),R0 | MOV.W R0, @(disp:4, Rn),R0 | | |
| 1000 | 10MD | imm/disp | | CMP/EQ #imm:8,R0 | BT disp:8 | | BF disp:8 |
| 1000 | 11MD | imm/disp | | | | | |
| 1001 | Rn | disp | | MOV.W @(disp:8,PC),Rn | | | |
| 1010 | disp | | | BRA disp:12 | | | |
| 1011 | disp | | | BSR disp:12 | | | |

**Table 2.18 Operation Code Map (cont)**

| Instruction Code | | | Fx: 0000 | Fx: 0001 | Fx: 0010 | Fx: 0011–1111 |
|---|---|---|---|---|---|---|
| MSB | | LSB | MD: 00 | MD: 01 | MD: 10 | MD: 11 |
| 1100 | 00MD | imm/disp | MOV.B R0,@(disp:8,GBR) | MOV.W R0,@(disp:8,GBR) | MOV.L R0,@(disp:8,GBR) | TRAPA #imm:8 |
| 1100 | 01MD | disp | MOV.B @(disp:8,GBR),R0 | MOV.W @(disp:8,GBR),R0 | MOV.L @(disp:8,GBR),R0 | MOVA @(disp:8,PC),R0 |
| 1100 | 10MD | imm | TST #imm:8,R0 | AND #imm:8,R0 | XOR #imm:8,R0 | OR #imm:8,R0 |
| 1100 | 11MD | imm | TST.B #imm:8,@(R0,GBR) | AND.B #imm:8,@(R0,GBR) | XOR.B #imm:8,@(R0,GBR) | OR.B #imm:8,@(R0,GBR) |
| 1101 | Rn | disp | MOV.L @(disp:8,PC),Rn | | | |
| 1110 | Rn | imm | MOV #imm:8,Rn | | | |
| 1111 | ... | | | | | |

## 2.5 CPU State

### 2.5.1 State Transitions

The CPU has five processing states: reset, exception processing, bus release, program execution and power-down. The transitions between the states are shown in figure 2.6. For more information on the reset and exception processing states, see section 4, Exception Processing. For details on the power-down states, see section 19, Power Down States.

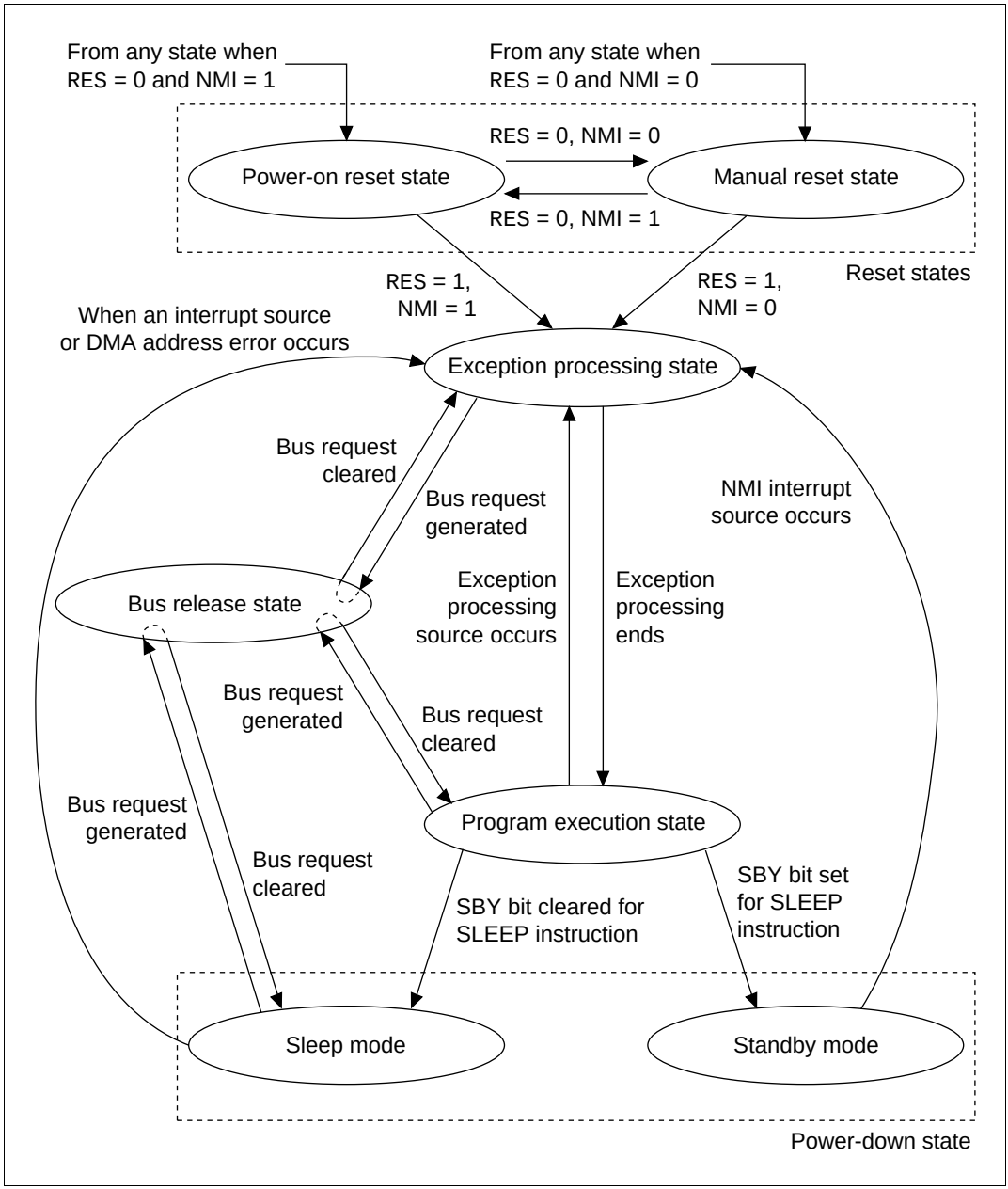



**Figure 2.6 Transitions Between Processing States**

**Reset State:** In the reset state the CPU is reset. This occurs when the $\overline{\text{RES}}$ pin level goes low. When the NMI pin is high, the result is a power-on reset; when it is low, a manual reset will occur.When turning on the power, make sure to carry out a power-on reset.

On a power-on reset, all CPU internal states and on-chip peripheral module registers are initialized. In a manual reset, all CPU internal states and on-chip peripheral module registers, with the exception of the bus state controller (BSC) and pin function controller (PFC), are initialized. On a manual reset, the BSC is not initialized, so the refresh operation will continue.

**Exception Processing State:** Exception processing is a transient state that occurs when the CPU's processing state flow is altered by exception processing sources such as resets or interrupts.

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

For an interrupt, the stack pointer (SP) is accessed and the program counter (PC) and status register (SR) are saved to the stack area. The exception service routine start address is fetched from the exception processing vector table; the CPU then branches to that address and the program starts executing, thereby entering the program execution state.

**Program Execution State:** In the program execution state, the CPU sequentially executes the program.

**Power-Down State:** In the power-down state, the CPU operation halts and power consumption declines. The SLEEP instruction places the CPU in the power-down state. This state has two modes: sleep mode and standby mode. This is described in more detail in section 2.5.1, Power-Down State.

**Bus Release State:** In the bus release state, the CPU releases rights to the bus to the device that has requested them.

### 2.5.2 Power-Down State

In addition to the ordinary program execution states, the CPU also has a power-down state in which CPU operation halts and power consumption is lowered. There are two power-down state modes: sleep mode and standby mode.

**Sleep Mode:** When the standby bit SBY (in the standby control register SBYCR) is cleared to 0 and a SLEEP instruction executed, the CPU moves from program execution state to sleep mode. In the sleep mode, the CPU halts and the contents of its internal registers and the data in on-chip RAM are stored. The on-chip peripheral modules other than the CPU do not halt in the sleep mode.

To return from sleep mode, use a reset, any interrupt, or a DMA address error; the CPU returns to ordinary program execution state through the exception processing state.

**Software Standby Mode:** To enter the standby mode, set the standby bit SBY (in the standby control register SBYCR) to 1 and execute a SLEEP instruction. In standby mode, all CPU, on-chip peripheral module and oscillator functions are halted. CPU internal register contents and on-chip RAM data are held.

To return from standby mode, use a reset or an external NMI interrupt. For resets, the CPU returns to ordinary program execution state through the exception processing state when placed in a reset state during oscillator stabilization time. For NMI interrupts, the CPU returns to ordinary program execution state through the exception processing state after the oscillator stabilization time has elapsed. In this mode, power consumption drops markedly, since the oscillator stops.

**Table 2.19 Power-Down State**

| | | State | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Mode** | **Conditions** | **Clock** | **CPU** | **On-chip Peripheral Modules** | **CPU Registers** | **RAM** | **I/O Ports** | **Canceling** |
| Sleep mode | Execute SLEEP instruction with SBY bit cleared to 0 in SBYCR | Run | Halt | Run | Held | Held | Held | 1. Interrupt<br>2. DMA address error<br>3. Power-on reset<br>4. Manual reset |
| Standby mode | Execute SLEEP instruction with SBY bit set to 1 in SBYCR | Halt | Halt | Halt and initialize* | Held | Held | Held or high-Z* (selectable) | 1. NMI<br>2. Power-on reset<br>3. Manual reset |

Note: Differs depending on the peripheral module and pin.

# Section 3 Operating Modes

## 3.1 Types of Operating Modes and Their Selection

The SH7020 and SH7021 operate in one of four operating modes (modes 0, 1, 2, and 7). Modes 0 and 1 differ in the bus width of memory area 0. The mode is selected by the mode pins (MD2–MD0) as indicated in table 3.1. Do not change the mode selection while the chip is operating.

**Table 3.1 Operating Mode Selection**

| | Pin Settings | | | | |
|---|---|---|---|---|---|
| **Operating Mode** | **MD2** | **MD1** | **MD0** | **Mode Name** | **Bus Width of Area 0** |
| Mode 0*[2] | 0 | 0 | 0 | MCU mode 0 | 8 bits |
| Mode 1*[2] | 0 | 0 | 1 | MCU mode 1 | 16 bits |
| Mode 2 | 0 | 1 | 0 | MCU mode 2 | On-chip ROM |
| Mode 7* | 1 | 1 | 1 | PROM mode | — |

Notes : 1.SH7021 PROM version only
2.Only modes 0 and 1are available in the SH7020 ROMless version.

## 3.2 Operating Mode Descriptions

### 3.2.1 Mode 0 (MCU Mode 0)

In mode 0, memory area 0 has an eight-bit bus width. For the memory map, see section 8, Bus State Controller.

### 3.2.2 Mode 1 (MCU Mode 1)

In mode 1, memory area 0 has a 16-bit bus width.

### 3.2.3 Mode 2 (MCU Mode 2)

In mode 2, memory area 0 is assigned to the on-chip ROM.

### 3.2.4 Mode 7 (PROM Mode)

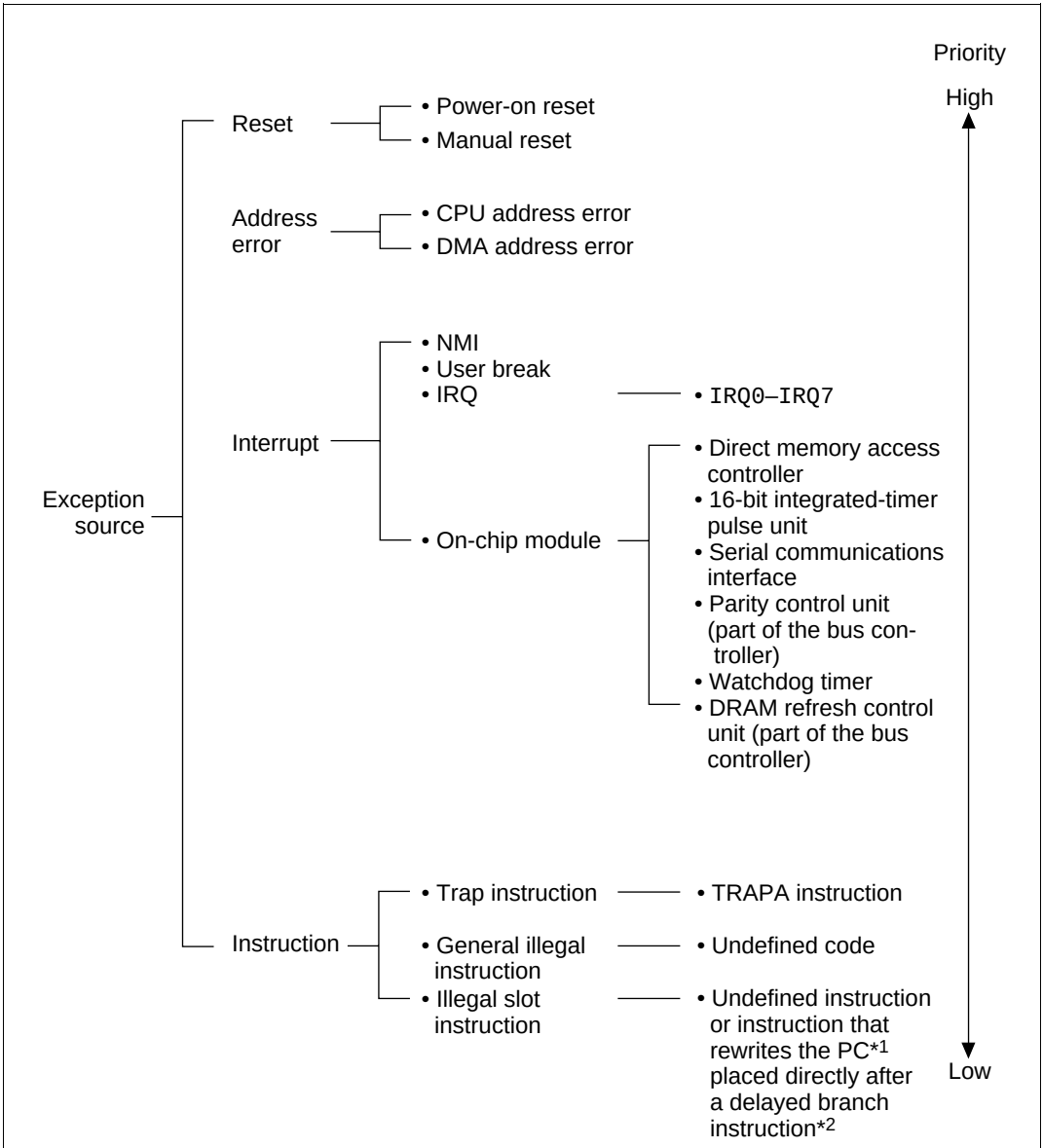
Mode 7 is a PROM mode. In this mode, the EPROM can be programmed.For details,see section 16, ROM. Do not set to mode 7 unless the product is the SH7021(PROM version).

# Section 4 Exception Processing

## 4.1 Overview

### 4.1.1 Exception Processing Types and Priorities

As figure 4.1 indicates, exception processing may be caused by a reset, address error, interrupt, or instruction. Exception sources are prioritized as indicated in figure 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in the priority order shown.




Notes: 1. The instructions that rewrite the PC are JMP, JSR, BRA, BSR, RTS, RTE, BT, BF, and TRAPA.

2. The delayed branch instructions are JMP, JSR. BRA. BSR, RTS, and RTE.

**Figure 4.1 Exception Source Types and Priority**

### 4.1.2 Exception Processing Operation

Exception sources are detected at the times indicated in table 4.1, whereupon processing starts.

**Table 4.1 Exception Source Detection and Time of the Start of Processing**

| Exception Type | | Source Detection and Time of the Start of Processing |
|---|---|---|
| Reset | Power-on | Low-to-high transition at pin RES when NMI is high |
| | Manual | Low-to-high transition at pin RES when NMI is low |
| Address error | | Detected when instruction is decoded and starts after the instruction that was executing prior to this point is completed. |
| Interrupt | | Detected when instruction is decoded and starts after the instruction that was executing prior to this point is completed. |
| Instruction | Trap instruction | Starts when a trap instruction (TRAPA) is executed. |
| | General illegal instruction | Starts when undefined code is decoded at a position other than directly after a delayed branch instruction (a delay slot). |
| | Illegal slot instruction | Starts when undefined code or an instruction that rewrites the PC is decoded directly after a delayed branch instruction (in a delay slot). |

When exception processing begins, the CPU operates as follows:

**Resets:** The initial values of the program counter (PC) and stack pointer (SP) are read from the exception vector table (the respective PC and SP values are H'00000000 and H'00000004 for a power-on reset and H'00000008 and H'0000000C for a manual reset). For more information on the exception vector table, see section 4.1.3, Exception Vector Table. Next, the vector base register (VBR) is cleared to zero and interrupt mask bits (I3–I0) in the status register (SR) are set to 1111. Program execution starts from the PC address read from the exception vector table.

**Address Errors, Interrupts and Instructions:** SR and PC are pushed onto the stack indicated in R15. For interrupts, the interrupt priority level is written in the interrupt mask bits (I3–I0). For address errors and instructions, bits I3–I0 are not affected. Next, the start address is fetched from the exception vector table, and program execution starts from this address.

### 4.1.3 Exception Process Vector Table

Before exception processing can execute, the exception vector table must be set in memory. The exception processing vector table holds the start addresses of exception service routines (the table for reset exception processing stores initial PC and SP values). Different vector numbers and vector table address offsets are assigned to different exception sources. The vector table addresses are calculated from the corresponding vector numbers and vector address offsets. In exception processing, the exception service routine start address is fetched from the exception vector table indicated by this vector table address.

Table 4.2 lists vector numbers and vector table address offsets. Table 4.3 shows how to calculate vector table addresses.

**Table 4.2 Exception Process Vector Table**

| Exception Source | | Vector Number | Vector table Address Offset |
|---|---|---|---|
| Power-on reset | PC | 0 | H'00000000–H'00000003 |
| | SP | 1 | H'00000004–H'00000007 |
| Manual reset | PC | 2 | H'00000008–H'0000000B |
| | SP | 3 | H'0000000C–H'0000000F |
| General illegal instruction | | 4 | H'00000010–H'00000013 |
| (Reserved for system use) | | 5 | H'00000014–H'00000017 |
| Illegal slot instruction | | 6 | H'00000018–H'0000001B |
| (Reserved for system use) | | 7 | H'0000001C–H'0000001F |
| | | 8 | H'00000020–H'00000023 |
| CPU address error | | 9 | H'00000024–H'00000027 |
| DMA address error | | 10 | H'00000028–H'0000002B |
| Interrupts | NMI | 11 | H'0000002C–H'0000002F |
| | User break | 12 | H'00000030–H'00000033 |
| (Reserved for system use) | | 13–31 | H'00000034–H'00000037 to H'0000007C–H'0000007F |
| Trap instruction (user vectors) | | 32–63 | H'00000080–H'00000083 to H'000000FC–H'000000FF |
| Interrupts | IRQ0 | 64 | H'00000100–H'00000103 |
| | IRQ1 | 65 | H'00000104–H'00000107 |
| | IRQ2 | 66 | H'00000108–H'0000010B |
| | IRQ3 | 67 | H'0000010C–H'0000010F |
| | IRQ4 | 68 | H'00000110–H'00000113 |
| | IRQ5 | 69 | H'00000114–H'00000117 |
| | IRQ6 | 70 | H'00000118–H'0000011B |
| | IRQ7 | 71 | H'0000011C–H'0000011F |
| | On-chip modules* | 72–255 | H'00000120–H'00000123 to H'000003FC–H'000003FF |

Note: See table 5.3, Interrupt Exception Processing Vectors and Rankings, in section 5, Interrupt Controller, for details on vector numbers and vector table address offsets of individual on-chip peripheral module interrupts.

**Table 4.3 Calculation of Exception Vector table Addresses**

| Exception Source | Calculation of Vector table Addresses |
|---|---|
| Reset | (Vector table address) = (vector table address offset) = (vector number) × 4 |
| Address error, interrupt, instructions | (Vector table address) = VBR + (vector table address offset) = VBR + (vector number) × 4 |

Note: VBR: Vector base register. For vector table address offsets and vector numbers, see table 4.2.

## 4.2 Reset

### 4.2.1 Reset Types

A reset is the highest-priority exception. There are two types of reset: power-on reset and manual reset. As table 4.4 shows, a power-on reset initializes the internal state of the CPU and all registers of the on-chip peripheral modules. A manual reset initializes the internal state of the CPU and all registers of the on-chip peripheral modules except the bus state controller (BSC), pin function controller (PFC) and I/O ports (I/O).

**Table 4.4 Reset Types**

| | Transition Conditions | | Internal State | |
|---|---|---|---|---|
| **Reset** | **NMI** | **RES** | **CPU** | **On-Chip Peripheral Module** |
| Power-on Reset | High | Low | Initialize | Initialize |
| Manual Reset | Low | Low | Initialize | Initialize all except BSC, PFC and I/O |

### 4.2.2 Power-On Reset

When the NMI pin is high, a low input at the $\overline{\text{RES}}$ pin drives the chip into the power-on reset state. The $\overline{\text{RES}}$ pin should be driven low while the clock pulse generator (CPG) is stopped (or while the CPG is operating during the oscillation settling time) for at least 20 $t_{cyc}$ to assure that the LSI is reset. A power-on reset initializes the internal state of the CPU and all registers of the on-chip peripheral modules. For pin states in the power-on reset state, see appendix B, Pin States.

While the NMI pin remains high, if the $\overline{\text{RES}}$ pin is held low for a certain time then driven high in the power-on state, power-on reset exception processing begins. The CPU then:

1. Reads the start address (initial PC value) from the exception vector table.
2. Reads the initial stack pointer value (SP) from the exception vector table.

3. Clears the vector base register (VBR) to H'00000000, and sets interrupt mask bits I3–I0 in the status register (SR) to H'F (1111).
4. Loads the values read from the exception vector table into PC and SP and starts program execution.

Further, make sure to carry out a power-on reset when turning on the power of the system.

### 4.2.3 Manual Reset

When the NMI pin is high, a low input at the $\overline{\text{RES}}$ pin drives the chip into the manual reset state. To be assured of resetting the LSI, drive the $\overline{\text{RES}}$ pin low for at least 20 $t_{cyc}$. A manual reset initializes the internal state of the CPU and all registers of the on-chip peripheral modules except the bus state controller, pin function controller and I/O ports. Since a manual reset does not affect the bus state controller, the DRAM refresh control function operates even if the manual reset state continues for a long time. When a manual reset is performed during the bus cycle, the manual reset exception processing waits for the end of the bus cycle before beginning. The manual reset thus cannot be used to abort the bus cycle. For the pin states during the manual reset state, see appendix B, Pin States.

While the NMI pin remains low, if the $\overline{\text{RES}}$ pin is held low for a certain time then driven high in the manual reset state, manual reset exception processing begins. The CPU carries out the same operations as for a power-on reset.

## 4.3 Address Errors

### 4.3.1 Address Error Sources

Address errors occur during instruction fetches and data reading/writing as shown in table 4.5.

**Table 4.5 Address Error Sources**

| Bus Cycle | | | |
|---|---|---|---|
| **Type** | **Bus Master** | **Operation** | **Address Error** |
| Instruction fetch | CPU | Instruction fetch from even address | None (normal) |
| | | Instruction fetch from odd address | Address error |
| | | Instruction fetch from outside on-chip peripheral module space | None (normal) |
| | | Instruction fetch from on-chip peripheral module space | Address error |
| Data read/write | CPU or DMAC | Access to word data from even address | None (normal) |
| | | Access to word data from odd address | Address error |
| | | Access to long word data aligned on long word boundary | None (normal) |
| | | Access to long word data not aligned on long word boundary | Address error |
| | | Access to word or byte data in on-chip peripheral module space* | None (normal) |
| | | Access to long word data in 16-bit on-chip peripheral module space* | None (normal) |
| | | Access to long word data in 8-bit on-chip peripheral module space* | Address error |

Note: See section 8, Bus State Controller, for details on the on-chip peripheral module space.

### 4.3.2 Address Error Exception Processing

When an address error occurs, address error exception processing starts after both the bus cycle that caused the address error and the instructions that were being executed at that time have been completed. The CPU then:

1. Pushes the SR onto the stack.
2. Pushes the program counter onto the stack. The PC value saved is the top address of the instruction following the last instruction to be executed.
3. Fetches the exception service routine start address from the exception vector table for the address error that occurred and starts program execution from that address. The branch that occurs here is not a delayed branch.

## 4.4 Interrupts

### 4.4.1 Interrupt Sources

Table 4.6 lists the types of interrupt exception processing sources (NMI, user break, IRQ, on-chip peripheral module).

**Table 4.6 Interrupt Sources**

| Interrupt | Requesting Pin or Module | Number of Sources |
|---|---|---|
| NMI | NMI pin (external input) | 1 |
| User break | User break controller | 1 |
| IRQ | IRQ0–IRQ7 pin (external input) | 8 |
| On-chip | Direct Memory Access Controller | 4 |
| | 16-bit integrated-timer pulse unit | 15 |
| | Serial communications interface | 8 |
| | Watchdog timer | 1 |
| | Bus state controller | 2 |

Each interrupt source has a different vector number and vector address offset value. See table 5.3, Interrupt Exception Vectors and Rankings, in section 5, Interrupt Controller, for details on vector numbers and vector table address offsets.

### 4.4.2 Interrupt Priority Rankings

Interrupt sources are assigned priorities. When multiple interrupts occur at the same time, the interrupt controller (INTC) ascertains their priorities and starts exception processing based on its findings. Priorities from 16–0 can be assigned, with 0 the lowest level and 16 the highest. The NMI has priority level 16 and cannot be masked. NMI is always accepted. The user break priority level is 15. The IRQ and on-chip peripheral module interrupt priority levels can be set in interrupt priority level registers A–E (IPRA–IPRE) as shown in table 4.7. Priority levels 0–15 can be set. See section 5.3.1, Interrupt Priority Level Registers A–E (IPRA–IPRE), for details.

**Table 4.7 Interrupt Priority Rankings**

| Type | Priority | Comments |
|---|---|---|
| NMI | 16 | Fixed and unmaskable |
| User break | 15 | Fixed |
| IRQ and on-chip peripheral modules | 0–15 | Set in interrupt priority level registers A–E (IPRA–IPRE) |

### 4.4.3 Interrupt Exception Processing

When an interrupt is generated, the INTC ascertains the interrupt rankings. NMI is always accepted, but other interrupts are only accepted if their ranking is higher than the ranking set in the interrupt mask bits (I3–I0) of the SR.

When an interrupt is accepted, interrupt exception processing begins. In the interrupt exception processing sequence, the SR and PC are pushed onto the stack, and the priority level of the accepted interrupt is copied to the interrupt mask level bits (I3–I0) in the SR. In NMI exception processing, the priority ranking is 16 but the value 15 (H'F) is stored in I3–I0. The exception service routine start address for the accepted interrupt is fetched from the exception vector table and the program branches to that address and starts executing. For further information on interrupts, see section 5.4, Interrupt Operation.

## 4.5 Instruction Exceptions

### 4.5.1 Types of Instruction Exceptions

Table 4.8 shows the three types of instruction that start exception processing (trap instructions, illegal slot instructions, and general illegal instructions).

**Table 4.8 Types of Instruction Exceptions**

| Type | Source Instruction | Comments |
|---|---|---|
| Trap instruction | TRAPA | — |
| Illegal slot instruction | Undefined code or instruction that rewrites the PC located immediately after a delayed branch instruction (delay slot) | Delayed branch instructions are: JMP, JSR, BRA, BSR, RTS, RTE. Instructions that rewrite the PC are: JMP, JSR, BRA, BSR, RTS, RTE, BT, BF and TRAPA |
| General illegal instructions | Undefined code in other than delayed slot | — |

### 4.5.2 Trap Instruction

Trap instruction exception processing is carried out when a trap instruction (TRAPA) is executed. The CPU then:

1. Saves the status register by pushing register contents onto the stack.
2. Pushes the program counter value onto the stack. The PC value saved is the top address of the next instruction after the TRAPA instruction.
3. Reads an exception processing service routine start address from the vector table corresponding to a vector number specified in the TRAPA instruction, branches to that address, and starts program execution. The branch is not a delayed branch.

### 4.5.3 Illegal Slot Instruction

An instruction located immediately after a delayed branch instruction is called an “instruction placed in a delay slot.” If an undefined instruction is located in a delay slot, illegal slot instruction exception processing begins executing when the undefined code is decoded. Illegal slot instruction exception processing also begins when the instruction located in the delay slot is an instruction that rewrites the program counter. In this case, exception processing begins when the instruction that rewrites the PC is decoded. The CPU performs illegal slot exception processing as follows:

1. Saves the status register onto the stack.
2. Pushes the program counter value onto the stack. The PC value saved is the branch destination address of the delayed branch instruction immediately before the instruction that contains the undefined code or rewrites the PC.
3. Fetches an exception processing service routine start address from the vector table corresponding to the exception that occurred, branches to that address and the program starts executing. The branch is not a delayed branch.

### 4.5.4 General Illegal Instructions

If an undefined instruction located other than a delay slot (immediately after a delayed branch instruction) is decoded, general illegal instruction exception processing is executed. The CPU follows the same procedure as for illegal slot exception processing, except that the program counter (PC) value pushed on the stack in general illegal instruction exception processing is the top address of the illegal instruction with the undefined code.

## 4.6 Cases in Which Exceptions Are Not Accepted

In some cases, address errors and interrupts that directly follow a delayed branch instruction or interrupt-disabled instruction are not accepted immediately. Table 4.9 lists these cases. When this occurs, the exception is accepted when an instruction that can accept the exception is decoded.

**Table 4.9 Cases in Which Exceptions Are Not Accepted**

| | Exception Source | |
|---|---|---|
| **Case** | **Address Error** | **Interrupt** |
| Immediately after delayed branch instruction*1 | X | X |
| Immediately after interrupt-disabled instruction*2 | O | X |

X: Not accepted

O: Accepted

Notes: 1. Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE

2. Interrupt-disabled instructions: LDC, LDC.L, STC, STC.L, LDS, LDS.L, STS, STS.L

### 4.6.1 Immediately after Delayed Branch Instructions

Address errors and interrupts are not accepted when an instruction in a delay slot immediately following a delayed branch instruction is decoded. The delayed branch instruction and the instruction in the delay slot are therefore always executed one after the other. Exception processing is never inserted between them.

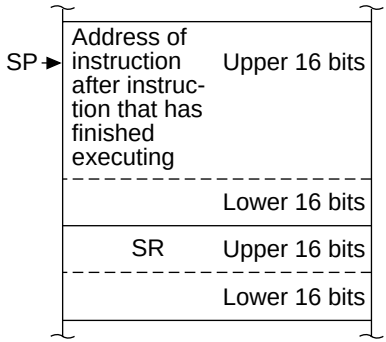
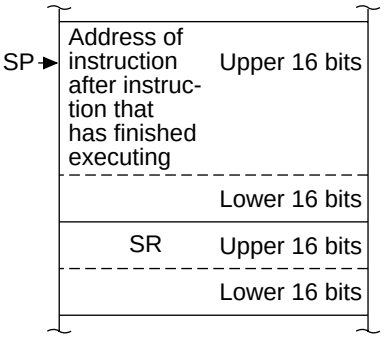
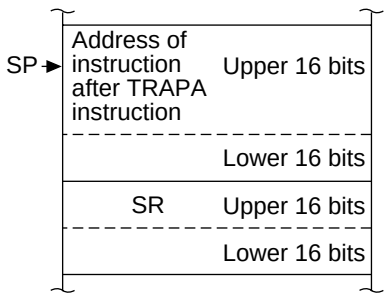
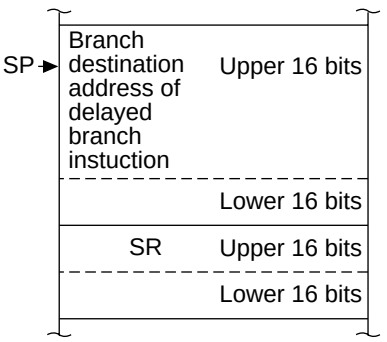
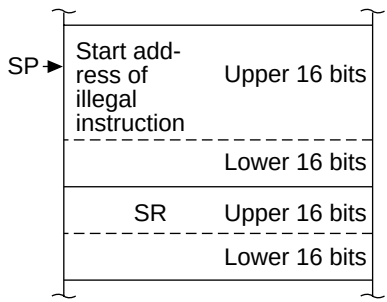
### 4.6.2 Immediately after Interrupt-Disabling Instructions

Interrupts are not accepted when the instruction immediately following an interrupt-disabled instruction is decoded. Address errors are accepted, however.

## 4.7 Stack Status after Exception Processing

Table 4.10 shows the stack after exception processing.

**Table 4.10 Stack after Exception Processing**

| Type | Stack Status | Type | Stack Status |
|---|---|---|---|
| Address error |  | Interrupt |  |
| Trap instruction |  | Illegal slot instruction |  |
| General illegal instruction |  | | |

Note: Stack status is based on a bus width of 16 bits.

## 4.8 Notes

### 4.8.1 Value of the Stack Pointer (SP)

An address error occurs if the stack is accessed for exception processing when the value of the stack pointer (SP) is not a multiple of four. Therefore, a multiple of four should always be stored in SP.

### 4.8.2 Value of the Vector Base Register (VBR)

An address error occurs if the vector table is accessed for exception processing when the value of the vector base register (VBR) is not a multiple of four. Therefore, VBR should always be set to a multiple of four.

### 4.8.3 Address Errors that Are Caused by Stacking During Address Error Exception Processing

When the stack pointer is not a multiple of four, address errors will occur in the exception processing (interrupt, etc.) stacking. After the exception processing ends, the CPU will then shift to address error exception processing. An address error will also occur during the address error exception processing stacking, but the CPU is set up to ignore the address error so that it can avoid an infinite series of address errors. This allows it to shift program control to the address error exception service routine and process the error.

When an address error does occur in exception processing stacking, the stacking bus cycle (write) is executed. In SR and PC stacking, four is subtracted from each of the SPs so the SP values are not multiples of four after stacking either. Since the address value output during stacking is the SP value, the address that produced the error is exactly what is output. In such cases, the stacked write data will be undefined.

# Section 5 Interrupt Controller (INTC)

## 5.1 Overview

The interrupt controller (INTC) determines the priority of interrupt sources and controls interrupt requests to the CPU. INTC has registers for assigning priority levels to interrupt sources. These registers handle interrupt requests according to user-established priorities.

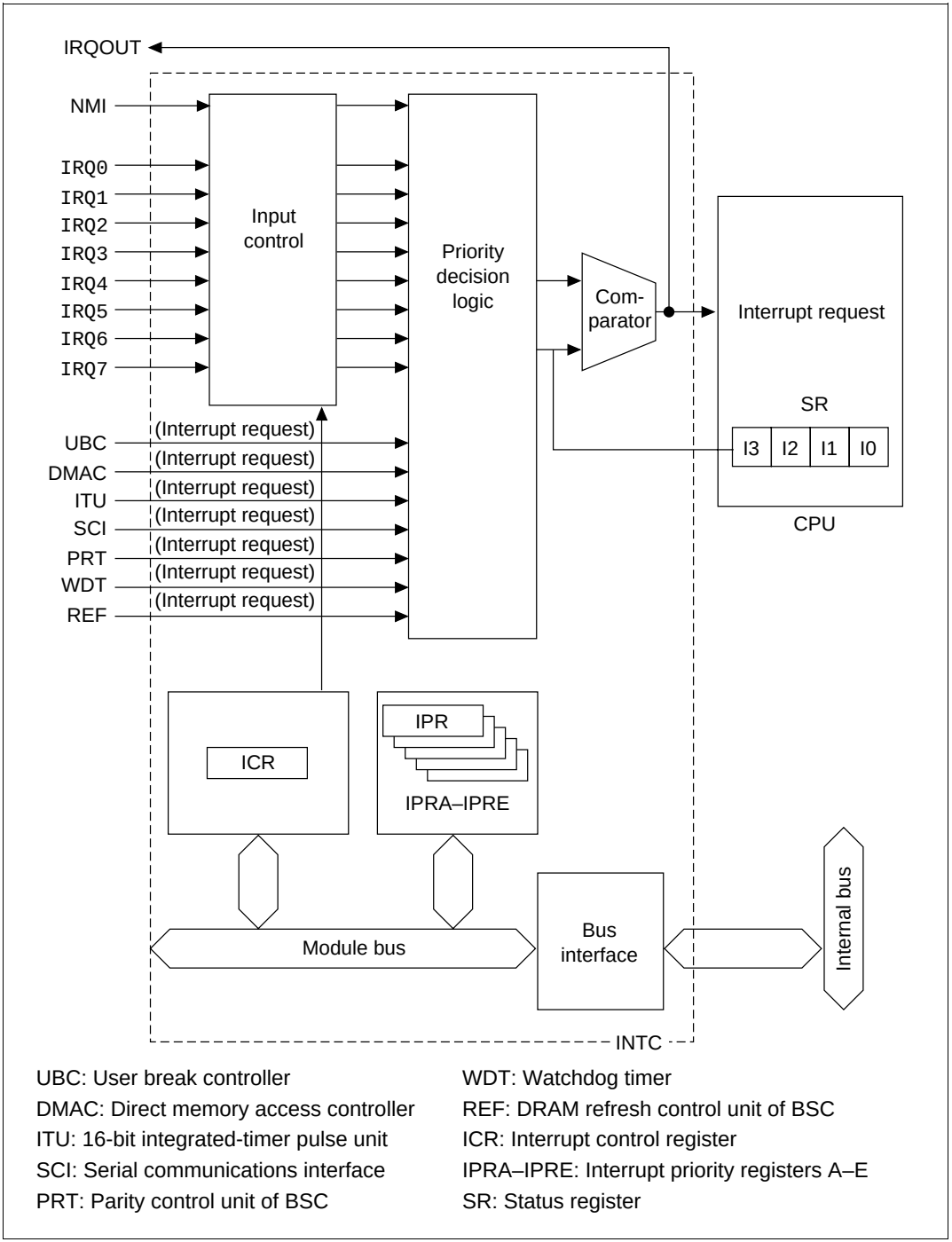
### 5.1.1 Features

The interrupt controller has the following features:

- 16 settable priority levels: Five interrupt priority registers can set 16 levels of interrupt priorities for IRQ and on-chip peripheral interrupt sources.
- The INTC has an NMI input level T bit that indicates NMI pin status. By reading this bit with the interrupt exception service routine, the pin status can be checked for use in a noise canceller function.
- The interrupt controller can notify external devices (via the $\overline{\text{IRQOUT}}$ pin) that an onchip interrupt has been occured. In this way an external device can, for example, be informed if an on-chip interrupt occurs while the chip is operating in a bus-released mode and the bus has been requested.

### 5.1.2 Block Diagram

Figure 5.1 is a block diagram of the interrupt controller.




UBC: User break controller
DMAC: Direct memory access controller
ITU: 16-bit integrated-timer pulse unit
SCI: Serial communications interface
PRT: Parity control unit of BSC
WDT: Watchdog timer
REF: DRAM refresh control unit of BSC
ICR: Interrupt control register
IPRA–IPRE: Interrupt priority registers A–E
SR: Status register

**Figure 5.1 Block Diagram of the Interrupt Controller**

### 5.1.3 Pin Configuration

INTC pins are summarized in table 5.1.

**Table 5.1 INTC Pin Configuration**

| Name | Abbr. | I/O | Function |
|---|---|---|---|
| Nonmaskable interrupt input pin | NMI | I | Inputs a non-maskable interrupt request signal |
| Interrupt request input pins | IRQ0–IRQ7 | I | Inputs maskable interrupt request signals |
| Interrupt request output pin | IRQOUT | O | Outputs a signal indicating an interrupt source has occurred. |

### 5.1.4 Registers

The interrupt controller has six registers as listed in table 5.2. These registers are used for setting interrupt priority levels and controlling the detection of external interrupt input signals.

**Table 5.2 Interrupt Controller Register Configuration**

| Name | Abbr. | R/W | Address*2 | Initial Value | Bus width |
|---|---|---|---|---|---|
| Interrupt priority register A | IPRA | R/W | H'5FFFF84 | H'0000 | 8, 16, 32 |
| Interrupt priority register B | IPRB | R/W | H'5FFFF86 | H'0000 | 8, 16, 32 |
| Interrupt priority register C | IPRC | R/W | H'5FFFF88 | H'0000 | 8, 16, 32 |
| Interrupt priority register D | IPRD | R/W | H'5FFFF8A | H'0000 | 8, 16, 32 |
| Interrupt priority register E | IPRE | R/W | H'5FFFF8C | H'0000 | 8, 16, 32 |
| Interrupt control register | ICR | R/W | H'5FFFF8E | *1 | 8, 16, 32 |

Note: 1. H'8000 when pin NMI is high, H'0000 when pin NMI is low.
2. Only the values of bits A27–A24 and A8–A0 are valid; bits A23–A9 are ignored. For details on the register addresses, see section 8.3.5, Description of Areas.

## 5.2 Interrupt Sources

There are four types of interrupt sources: NMI, user break, IRQ, and on-chip peripheral module interrupts.

Interrupt rankings are expressed as priority levels (0–16), with 0 the lowest and 16 the highest. An interrupt set to level 0 is masked.

### 5.2.1 NMI Interrupts

NMI is the highest-priority interrupt (level 16) and is always accepted. Input at the NMI pin is edge-sensed. Either the rising or falling edge can be selected by setting the NMI edge select bit (NMIE) in the interrupt control register (ICR). NMI interrupt exception processing sets the interrupt mask level bits (I3–I0) in the status register (SR) to level 15.

### 5.2.2 User Break Interrupt

A user break interrupt occurs when a break condition is satisfied in the user break controller (UBC). A user break interrupt has priority level 15. User break interrupt exception processing sets the interrupt mask level bits (I3–I0) in the status register (SR) to level 15. For further details about the user break interrupt, see section 6, User Break Controller.

### 5.2.3 IRQ Interrupts

IRQ interrupts are requested by input from pins $\overline{\text{IRQ0}}$-$\overline{\text{IRQ7}}$. IRQ sense select bits 0–7 (IRQ0S–IRQ7S) in the interrupt control register (ICR) can select low-level sensing or falling-edge sensing for each pin independently. Interrupt priority registers A and B (IPRA and IPRB) can select priority levels from 0–15 for each pin. IRQ interrupt exception processing sets the interrupt mask level bits (I3–I0) in the status register (SR) to the priority level value of the IRQ interrupt that was accepted.

### 5.2.4 On-Chip Interrupts

On-chip interrupts are interrupts generated by the following 5 on-chip peripheral modules:

- Direct memory access controller (DMAC)
- 16-bit integrated-timer pulse unit (ITU)
- Serial communications interface (SCI)
- Bus state controller (BSC)
- Watchdog timer (WDT)

A different interrupt vector is assigned to each interrupt source, so the exception service routine does not have to decide which interrupt has occurred. Priority levels 0–15 can be assigned to individual on-chip peripheral module in interrupt priority registers C–E (IPRC–IPRE). On-chip interrupt exception processing sets the interrupt mask level bits (I3–I0) in the status register (SR) to the priority level value of the on-chip interrupt that was accepted.

### 5.2.5 Interrupt Exception Vectors and Priority Rankings

Table 5.3 lists the vector numbers, vector table address offsets, and interrupt priority order of the interrupt sources.

Each interrupt source is allocated a different vector number and vector table address offset. The vector table address is calculated from this vector number and address offset. In interrupt exception processing, the exception service routine start address is fetched from the vector table indicated by this vector table address. See table 4.3, Calculation of Exception Vector table Addresses, in section 4, Exception Processing, for details on this calculation.

Arbitrary interrupt priority levels between 0 and 15 can be assigned to IRQ and on-chip peripheral module interrupt sources by setting interrupt priority registers A–E (IPRA–IPRE) for each pin or module. The interrupt sources for IPRC–IPRE, however, must be ranked in the order listed under Priority Within Module in table 5.3 and cannot be changed. A reset assigns priority level 0 to IRQ and on-chip peripheral module interrupts. If the same priority level is assigned to two or more interrupt sources, and interrupts from those sources occur simultaneously, their priority order is the default priority order indicated at the right in table 5.3.

**Table 5.3 Interrupt Exception Vectors and Rankings**

| Interrupt Source | | Interrupt Priority Order (initial value) | IPR (bit numbers) | Priority Within Module | Vector No. | Address Offset in Vector table | Default Priority Order |
|---|---|---|---|---|---|---|---|
| NMI | | 16 | — | — | 11 | H'0000002C–H'0000002F | High |
| User break | | 15 | — | — | 12 | H'00000030–H'00000033 | |
| IRQ0 | | 0–15 (0) | IPRA (15–12) | — | 64 | H'00000100–H'00000103 | |
| IRQ1 | | 0–15 (0) | IPRA (11–8) | — | 65 | H'00000104–H'00000107 | |
| IRQ2 | | 0–15 (0) | IPRA (7–4) | — | 66 | H'00000108–H'0000010B | |
| IRQ3 | | 0–15 (0) | IPRA (3–0) | — | 67 | H'0000010C–H'0000010F | |
| IRQ4 | | 0–15 (0) | IPRB (15–12) | — | 68 | H'00000110–H'00000113 | |
| IRQ5 | | 0–15 (0) | IPRB (11–8) | — | 69 | H'00000114–H'00000117 | |
| IRQ6 | | 0–15 (0) | IPRB (7–4) | — | 70 | H'00000118–H'0000011B | |
| IRQ7 | | 0–15 (0) | IPRB (3–0) | — | 71 | H'0000011C–H'0000011F | |
| DMAC0 | DEI0 | 0–15 (0) | IPRC (15–12) | 3 | 72 | H'00000120–H'00000123 | |
| | Reserved | | | 2 | 73 | H'00000124–H'00000127 | |
| DMAC1 | DEI1 | | | 1 | 74 | H'00000128–H'0000012B | |
| | Reserved | | | 0 | 75 | H'0000012C–H'0000012F | |
| DMAC2 | DEI2 | 0–15 (0) | IPRC (11–8) | 3 | 76 | H'00000130–H'00000133 | |
| | Reserved | | | 2 | 77 | H'00000134–H'00000137 | |
| DMAC3 | DEI3 | | | 1 | 78 | H'00000138–H'0000013B | |
| | Reserved | | | 0 | 79 | H'0000013C–H'0000013F | |
| ITU0 | IMIA0 | 0–15 (0) | IPRC (7–4) | 3 | 80 | H'00000140–H'00000143 | |
| | IMIB0 | | | 2 | 81 | H'00000144–H'00000147 | |
| | OVI0 | | | 1 | 82 | H'00000148–H'0000014B | |
| | Reserved | | | 0 | 83 | H'0000014C–H'0000014F | |
| ITU1 | IMIA1 | 0–15 (0) | IPRC (3–0) | 3 | 84 | H'00000150–H'00000153 | |
| | IMIB1 | | | 2 | 85 | H'00000154–H'00000157 | |
| | OVI1 | | | 1 | 86 | H'00000158–H'0000015B | |
| | Reserved | | | 0 | 87 | H'0000015C–H'0000015F | |
| ITU2 | IMIA2 | 0–15 (0) | IPRD (15–12) | 3 | 88 | H'00000160–H'00000163 | |
| | IMIB2 | | | 2 | 89 | H'00000164–H'00000167 | |
| | OVI2 | | | 1 | 90 | H'00000168–H'0000016B | |
| | Reserved | | | 0 | 91 | H'0000016C–H'0000016F | |

**Table 5.3 Interrupt Exception Vectors and Rankings (cont)**

| Interrupt Source | | Interrupt Priority Order (initial value) | IPR (bit numbers) | Priority Within Module | Vector No. | Address Offset in Vector table | Default Priority Order |
|---|---|---|---|---|---|---|---|
| ITU3 | IMIA3 | 0–15 (0) | IPRD (11–8) | 3 | 92 | H'00000170–H'00000173 | |
| | IMIB3 | | | 2 | 93 | H'00000174–H'00000177 | |
| | OVI3 | | | 1 | 94 | H'00000178–H'0000017B | |
| | Reserved | | | 0 | 95 | H'0000017C–H'0000017F | |
| ITU4 | IMIA4 | 0–15 (0) | IPRD (7–4) | 3 | 96 | H'00000180–H'00000183 | |
| | IMIB4 | | | 2 | 97 | H'00000184–H'00000187 | |
| | OVI4 | | | 1 | 98 | H'00000188–H'0000018B | |
| | Reserved | | | 0 | 99 | H'0000018C–H'0000018F | |
| SCI0 | ERI0 | 0–15 (0) | IPRD (3–0) | 3 | 100 | H'00000190–H'00000193 | |
| | RxI0 | | | 2 | 101 | H'00000194–H'00000197 | |
| | TxI0 | | | 1 | 102 | H'00000198–H'0000019B | |
| | TEI0 | | | 0 | 103 | H'0000019C–H'0000019F | |
| SCI1 | ERI1 | 0–15 (0) | IPRE (15–12) | 3 | 104 | H'000001A0–H'000001A3 | |
| | RxI1 | | | 2 | 105 | H'000001A4–H'000001A7 | |
| | TxI1 | | | 1 | 106 | H'000001A8–H'000001AB | |
| | TEI1 | | | 0 | 107 | H'000001AC–H'000001AF | |
| PRT*[1] | PEI | 0–15 (0) | IPRE (11–8) | 3 | 108 | H'000001B0–H'000001B3 | |
| | Reserved | | | 2 | 109 | H'000001B4–H'000001B7 | |
| | Reserved | | | 1 | 110 | H'000001B8–H'000001BB | |
| | Reserved | | | 0 | 111 | H'000001BC–H'000001BF | |
| WDT | ITI | 0–15 (0) | IPRE (7–4) | 3 | 112 | H'000001C0–H'000001C3 | |
| REF*[2] | CMI | | | 2 | 113 | H'000001C4–H'000001C7 | |
| | Reserved | | | 1 | 114 | H'000001C8–H'000001CB | |
| | Reserved | | | 0 | 115 | H'000001CC–H'000001CF | |
| Reserved | | — | — | — | 116 to 255 | H'000001D0–H'000001D3 to H'000003FC–H'000003FF | Low |

Notes: 1. PRT: Parity control unit of bus state controller.
2. REF: DRAM refresh control unit of bus state controller.

## 5.3 Register Descriptions

### 5.3.1 Interrupt Priority Registers A–E (IPRA–IPRE)

The five registers from IPRA–IPRE are 16-bit read/write registers that assign priority levels from 0–15 to the IRQ and on-chip peripheral module interrupt sources. Interrupt request sources are mapped onto IPRA–IPRE as shown in table 5.4.

| 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 |

**Table 5.4 Interrupt Request Sources and IPRA–IPRE**

| Register | Bits 15–12 | Bits 11–8 | Bits 7–4 | Bits 3–0 |
|---|---|---|---|---|
| IPRA | IRQ0 | IRQ1 | IRQ2 | IRQ3 |
| IPRB | IRQ4 | IRQ5 | IRQ6 | IRQ7 |
| IPRC | DMAC0, DMAC1 | DMAC2, DMAC3 | ITU0 | ITU1 |
| IPRD | ITU2 | ITU3 | ITU4 | SCI0 |
| IPRE | SCI1 | PRT*1 | WDT, REF*2 | (Reserved)*3 |

Notes: 1. PRT: Parity control unit of bus state controller. See section 8, Bus State Controller, for details.
2. REF: DRAM refresh control unit of bus controller. See section 8, Bus State Controller, for details.
3. When read, always 0. Always write 0 in reserved bits.

As indicated in table 5.4, four $\overline{\text{IRQ}}$ pins or four groups of on-chip peripheral modules are assigned to each interrupt priority register. The priority levels for the four pins or groups can be set by setting the corresponding 4-bit groups of bits 15–12, bits 11–8, bits 7–4, and bits 3–0 (of IPRA–IPRE) with values in the range of H'0 (0000) to H'F (1111). Setting H'0 gives interrupt priority level 0 (the lowest). Setting H'F gives level 15 (the highest). When two on-chip peripheral modules are assigned to the same bits (DMAC0 and DMAC1, or DMAC2 and DMAC3, or the watchdog timer and DRAM refresh control unit), those two modules have the same priority. A reset initializes IPRA–IPRE to H'0000. They are not initialized by the standby mode.

### 5.3.2 Interrupt Control Register (ICR)

ICR is a 16-bit register that sets the input detection mode of the external interrupt input pins NMI and $\overline{\text{IRQ0}}$–$\overline{\text{IRQ7}}$ and indicates the input signal level to the NMI pin. A reset initializes ICR but the standby mode does not.

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

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | IRQ0S | IRQ1S | IRQ2S | IRQ3S | IRQ4S | IRQ5S | IRQ6S | IRQ7S |
| 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 NMI input is high: 1; when NMI input is low: 0

- Bit 15 (NMI input level (NMIL)): NMIL sets the level of the signal input at the NMI pin. NMIL cannot be modified. The NMI input level can be read to determine the NMI pin level.

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

- Bits 14–9 (reserved): These bits always read as 0. The write value should always be 0.
- Bit 8 (NMI edge select (NMIE)): NMIE selects whether the falling or rising edge of the interrupt request signal to the NMI pin is sensed.

| Bit 8: NMIE | Description |
|---|---|
| 0 | Interrupt is requested on falling edge of NMI input (initial value) |
| 1 | Interrupt is requested on rising edge of NMI input |

Bits 7–0 (IRQ0–IRQ7 sense select (IRQ0S–IRQ7S)): IRQ0–IRQ7 select whether the falling edge or low level of the $\overline{\text{IRQ}}$ inputs is sensed at the pins $\overline{\text{IRQ0}}$–$\overline{\text{IRQ7}}$.

| Bits 7–0: IRQ0S–IRQ7S | Description |
|---|---|
| 0 | Interrupt is requested when $\overline{\text{IRQ}}$ input is low (initial value) |
| 1 | Interrupt is requested on falling edge of $\overline{\text{IRQ}}$ input |

## 5.4 Interrupt Operation

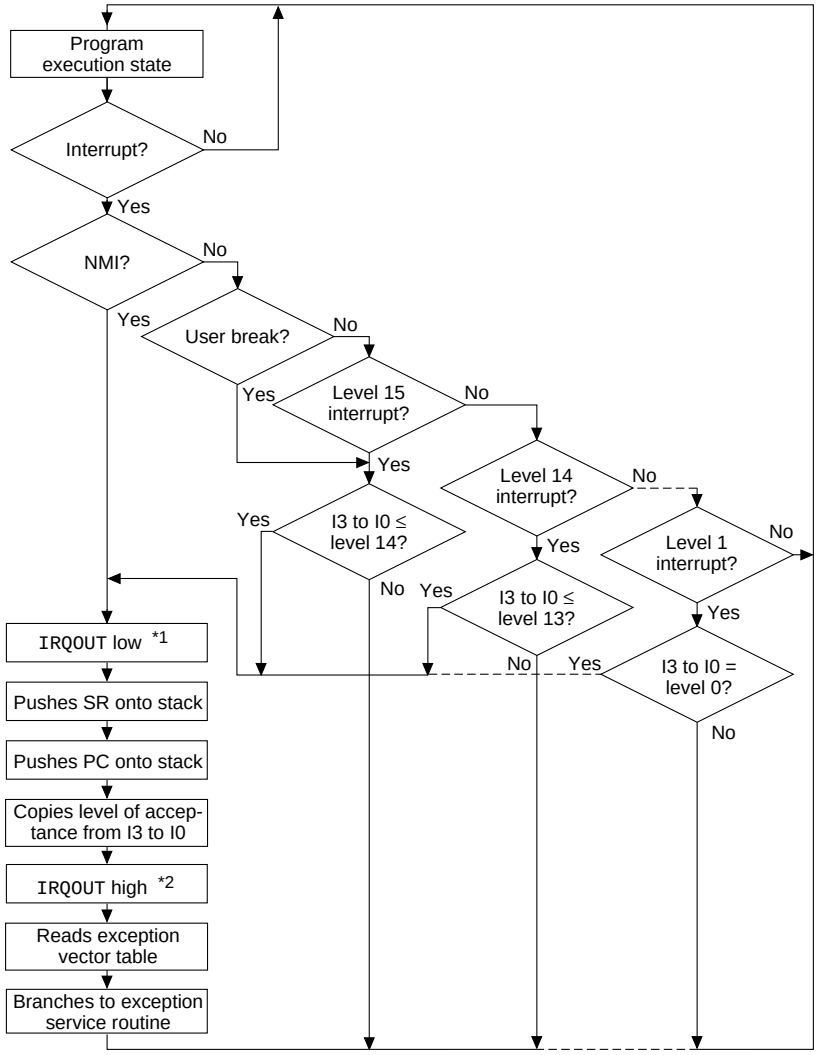
### 5.4.1 Interrupt Sequence

The sequence of interrupt operations will be explained below. Figure 5.2 is a flowchart of the operations up to acceptance of the interrupt.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest-priority interrupt in the interrupt requests sent, following the priority order indicated in table 5.3 and the levels set in interrupt priority registers A–E (IPRA–IPRE). Lower priority interrupts are ignored*. If two interrupts with the same priority level are requested simultaneously or if there are multiple interrupts occurring within a single module, the interrupt with the highest default priority or priority within module as indicated in table 5.3 is selected.
3. The interrupt controller compares the priority level of the selected interrupt request with the interrupt mask level bits (I3–I0) in the CPU's status register (SR). If the request priority level is equal to or less than the interrupt mask level, the request is ignored. If the request priority level is higher than the interrupt mask level, the interrupt controller accepts the request and sends an interrupt request signal to the CPU.
4. When the interrupt controller accepts an interrupt request, it drives the pin $\overline{\text{IRQOUT}}$ low.
5. The CPU detects the interrupt request sent from the interrupt controller when it decodes the next instruction to be executed. Instead of executing that instruction, the CPU starts interrupt exception processing.
6. In interrupt exception processing, first SR and PC are pushed onto the stack.
7. The priority level of the accepted interrupt is copied to the interrupt mask level bits (I3–I0) in the status register (SR).
8. When the accepted interrupt is level-sensed or from an on-chip peripheral module, The pin $\overline{\text{IRQOUT}}$ returns to the high level. If the accepted interrupt is edge-sensed, the pin $\overline{\text{IRQOUT}}$ returns to the high level when the instruction to be executed by the CPU in (5) is replaced by the interrupt exception processing. If the interrupt controller has accepted another interrupt (of a level higher than the current interrupt), however, the pin $\overline{\text{IRQOUT}}$ remains low.
9. The CPU accesses the exception vector table at the entry for the vector number of the accepted interrupt, reads the start address of the exception service routine, branches to that address, and starts executing the program there. This branch is not delayed.

Note: A request for an external interrupt (IRQ) designated as edge-detected is held pending once only. An external interrupt designated as level-detected is held pending as long as the interrupt request continues, but if the request is cleared before the CPU next accepts an interrupt, the interrupt request is regarded as not having been made.
Interrupt requests from on-chip supporting modules are level requests. When the status flag in a particular module is set, an interrupt is requested. For details, see the descriptions of the individual modules. Note that the interrupt request will be continued unless an

operation described in "Clearing Conditions" is performed.




I3 to I0 : Interrupt mask bits of status register

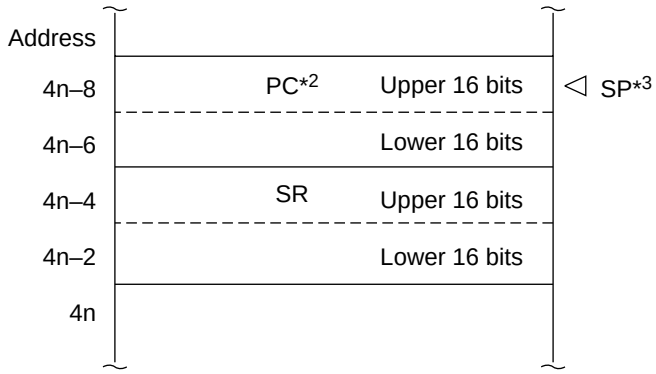
Notes : *1. IRQOUT is the same signal as the interrupt request signal to the CPU (Figure 5.1). The pin IRQOUT return to the high level when the interrupt controller has accepted the interrupt of a level higher than the I3 to I0 bits of the status register in the CPU.

*2. If the accepted interrupt is edge-sensed, the pin IRQOUT returns to the high level when the instruction to be executed by the CPU is replaced by the interrupt exception processing (before the status register is saved to the stack ). If the interrupt controller has accepted another interrupt of a level higher than the current interrupt. and has requested the interrupt to the CPU, however, the pin IRQOUT remains low.

**Figure 5.2 Flowchart of Interrupt Operation**

### 5.4.2 Stack after Interrupt Exception Processing

Figure 5.3 shows the stack after interrupt exception processing.




Notes: 1. Bus width is 16 bits.
2. PC stores the top address of the next instruction (return instruction) after the executed instruction.
3. The value of SP must always be a multiple of four.

**Figure 5.3 Stack after Interrupt Exception Processing**

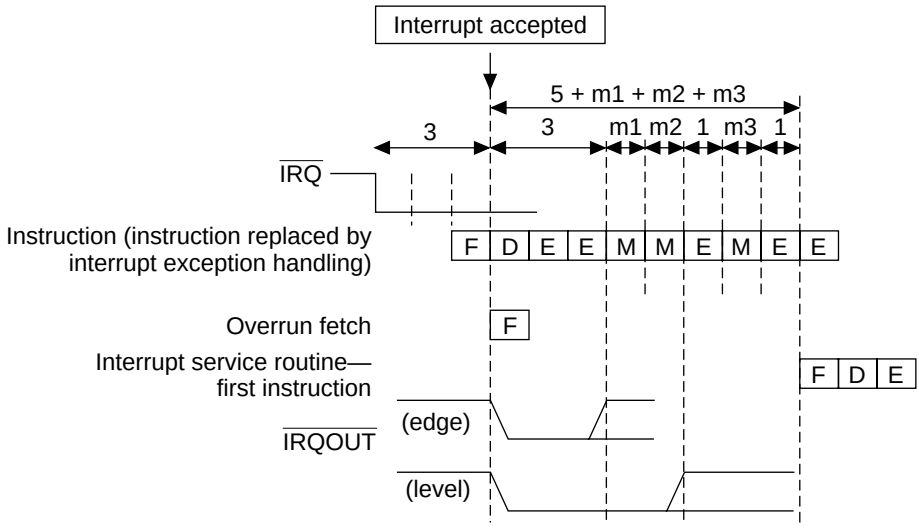
## 5.5 Interrupt Response Time

Table 5.5 indicates the interrupt response time, which is the time from the occurrence of an interrupt request until the interrupt exception processing starts and fetching of the first instruction of the interrupt service routine begins. Figure 5.4 shows the pipeline when an IRQ interrupt is accepted.

**Table 5.5 Interrupt Response Time**

| Item | | Number of States: NMI or On-Chip Interrupt | Number of States: IRQ | Notes |
|---|---|---|---|---|
| Interrupt priority decision and comparison with SR mask bit | | 2 | 3 | |
| Wait for completion of sequence currently being executed by CPU | | X (≥ 0) | | The longest sequence is the interrupt or address error exception processing sequence: X = 4 + m1 + m2 + m3 + m4. If an interrupt-masking instruction follows, however, the time may be longer. |
| Time from interrupt exception processing (saving PC and SR and fetching vector address) until fetching of first instruction of interrupt service routine starts | | 5 + m1 + m2 + m3 | | |
| Interrupt response | Total | 7 + m1 + m2 + m3 | 8 + m1 + m2 + m3 | |
| | Minimum | 10 | 11 | 0.50–0.55 μs at 20 MHz |
| | Maximum | 11 + 2(m1 + m2 + m3) + m4 | 12 + 2(m1 + m2 + m3) + m4 | (m1 = m2 = m3 = m4) 0.90–0.95 μs at 20 MHz |

Notes: m1–m4 are the number of states needed for the following memory accesses:
m1: SR save cycle (long word write)
m2: PC save cycle (long word write)
m3: Vector address read cycle (long word read)
m4: Fetch top instruction of interrupt service routine




When m1 = m2 = m3, the interrupt response time is 11 cycles.

| | | |
|---|---|---|
| F | (Instruction fetch) | Instruction fetched from memory where program is stored. |
| D | (Instruction decoding) | The fetched instruction is decoded. |
| E | (Instruction execution) | Data operations and address calculations are performed according to the decoded results. |
| M | (Memory access) | Data in memory is accessed. |

Note: For the interrupt acceptance timing, see table 4.1, Exception Source Detection and Exception Handling Start Timing, in section 4.1.2, Exception Handling Operation.

**Figure 5.4 Example of Pipelining in IRQ Interrupt Acceptance**

## 5.6 Usage Notes

When the following operations are performed in the order shown when a pin to which IRQ input is assigned is designated as a general input pin by the pin function controller (PFC) and inputs a low-level signal, the IRQ falling edge is detected, and an interrupt request is detected, immediately after the setting in (b) is performed:

- An interrupt control register (ICR) setting is made so that an interrupt is detected at the falling edge of IRQ. …… (a)
- The function of pins to which IRQ input is assigned is switched from general input to IRQ input by a pin function controller (PFC) setting. …… (b)

Therefore, when switching the pin function from general input pin to IRQ input, the pin function controller (PFC) setting should be changed to IRQ input while the pin to which IRQ input is assigned is high.

# Section 6 User Break Controller (UBC)

## 6.1 Overview

The user break controller (UBC) simplifies the debugging of user programs. Break conditions are set in the UBC and a user break interrupt request is sent to the CPU in response to the contents of a CPU or DMAC bus cycle. This function can implement an effective self-monitoring debugger, enabling a program to be debugged by itself without using a large in-circuit emulator.

### 6.1.1 Features

- The following break conditions can be set:
  - Address
  - CPU cycle or DMA cycle
  - Instruction fetch or data access
  - Read or write
  - Operand size (long word access, word access, or byte access)
- When break conditions are met, a user break interrupt is generated. A user-created user break interrupt exception routine can then be executed.
- When a break is set to a CPU instruction fetch, the break occurs just before the fetched instruction.

### 6.1.2 Block Diagram

Figure 6.1 is the block diagram of the user break controller.

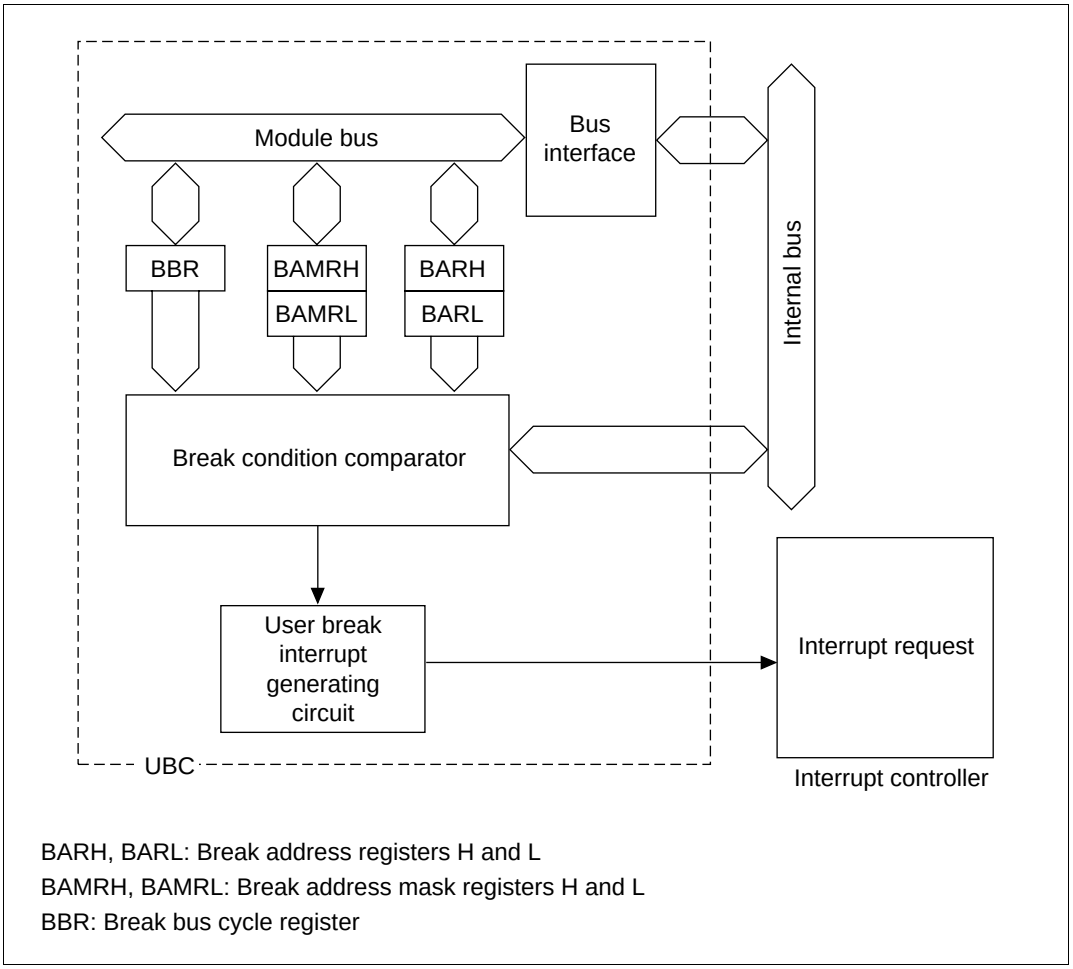



**Figure 6.1 Block Diagram of the User Break Controller**

### 6.1.3 Register Configuration

The user break controller has five registers as listed in table 6.1. These registers are used for setting break conditions.

**Table 6.1 User Break Controller Registers**

| Name | Abbr. | R/W | Address* | Initial Value | Bus width |
|---|---|---|---|---|---|
| Break address register high | BARH | R/W | H'5FFFF90 | H'0000 | 8, 16, 32 |
| Break address register low | BARL | R/W | H'5FFFF92 | H'0000 | 8, 16, 32 |
| Break address mask register high | BAMRH | R/W | H'5FFFF94 | H'0000 | 8, 16, 32 |
| Break address mask register low | BAMRL | R/W | H'5FFFF96 | H'0000 | 8, 16, 32 |
| Break bus cycle register | BBR | R/W | H'5FFFF98 | H'0000 | 8, 16, 32 |

Note: Only the values of bits A27–A24 and A8–A0 are valid; bits A23–A9 are ignored. For details on the register addresses, see section 8.3.5, Description of Areas.

## 6.2 Register Descriptions

### 6.2.1 Break Address Registers (BAR)

There are two break address registers—break address register H (BARH) and break address register L (BARL)—that together form a single group. Both are 16-bit read/write registers. BARH stores the upper bits (bits 31–16) of the address of the break condition. BARL stores the lower bits (bits 15–0) of the address of the break condition. A reset initializes both BARH and BARL to H'0000. Neither is initialized in standby mode.

**BARH:** Break address register H.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | BA31 | BA30 | BA29 | BA28 | BA27 | BA26 | BA25 | BA24 |
| 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: | BA23 | BA22 | BA21 | BA20 | BA19 | BA18 | BA17 | BA16 |
| 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 |

- BARH Bits 15–0 (break address 31–16 (BA31–BA16)): BA31–BA16 store the upper bit values (bits 31–16) of the address of the break condition.

**BARL:** Break address register L.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | BA15 | BA14 | BA13 | BA12 | BA11 | BA10 | BA9 | BA8 |
| 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: | BA7 | BA6 | BA5 | BA4 | BA3 | BA2 | BA1 | BA0 |
| 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 |

- BARL Bits 15–0 (break address 15–0 (BA15–BA0)): BA15–BA0 store the lower bit values (bits 15–0) of the address of the break condition.

### 6.2.2 Break Address Mask Register (BAMR)

The two break address mask registers—break address mask register H (BAMRH) and break address mask register L (BARML)—together form a single group. Both are 16-bit read/write registers. BAMRH determines which of the bits in the break address set in BARH are masked. BAMRL determines which of the bits in the break address set in BARL are masked. A reset initializes BAMRH and BARML to H'0000. They are not initialized in the standby mode.

**BAMRH:** Break address mask register H.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | BAM31 | BAM30 | BAM29 | BAM28 | BAM27 | BAM26 | BAM25 | BAM24 |
| 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: | BAM23 | BAM22 | BAM21 | BAM20 | BAM19 | BAM18 | BAM17 | BAM16 |
| 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 |

- BAMRH bits 15–0 (break address mask 31–16 (BAM31–BAM16)): BAM31–BAM16 specify whether bits BA31–BA16 of the break address set in BARH are masked or not.

**BAMRL:** Break address mask register L.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | BAM15 | BAM14 | BAM13 | BAM12 | BAM11 | BAM10 | BAM9 | BAM8 |
| 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: | BAM7 | BAM6 | BAM5 | BAM4 | BAM3 | BAM2 | BAM1 | BAM0 |
| 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 |

- BAMRL bits 15–0 (break address mask 15–0 (BAM15–BAM0)): BAM15–BAM0 specify whether bits BA15–BA0 of the break address set in BARH are masked or not.

| Bits 15–0: BAMn | Description |
|---|---|
| 0 | Break address bit BAn is included in the break condition (initial value) |
| 1 | Break address bit BAn is not included in the break condition |

n = 31–0

### 6.2.3 Break Bus Cycle Register (BBR)

The break bus cycle register (BBR) is a 16-bit read/write register that selects the following four break conditions:

- CPU cycle or DMA cycle
- Instruction fetch or data access
- Read or write
- Operand size (byte, word, long word).

A reset initializes BBR to H'0000. It is not initialized in the standby mode.

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

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | CD1 | CD0 | ID1 | ID0 | RW1 | RW0 | SZ1 | SZ0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bits 15–8 (reserved): These bits always read as 0. The write value should always be 0.
- Bits 7 and 6 (CPU cycle/DMA cycle select (CD1 and CD0)): CD1 and CD0 select whether to break on CPU and/or DMA bus cycles.

| Bit 7: CD1 | Bit 6: CD0 | Description |
|---|---|---|
| 0 | 0 | No break interrupt occurs (initial value) |
| | 1 | Break only on CPU cycles |
| 1 | 0 | Break only on DMA cycles |
| | 1 | Break on both CPU and DMA cycles |

- Bits 5 and 4 (instruction fetch/data access select (ID1, ID0)): ID1, ID0 select whether to break on instruction fetch and/or data access bus cycles.

| Bit 5: ID1 | Bit 4: ID0 | Description |
|---|---|---|
| 0 | 0 | No break interrupt occurs (initial value) |
| | 1 | Break only on instruction fetch cycles |
| 1 | 0 | Break only on data access cycles |
| | 1 | Break on both instruction fetch and data access cycles |

- Bits 3 and 2 (read/write select (RW1, RW0)): RW1, RW0 select whether to break on read and/or write access cycles.

| Bit 3: RW1 | Bit 2: RW0 | Description |
|---|---|---|
| 0 | 0 | No break interrupt occurs (initial value) |
| | 1 | Break only on read cycles |
| 1 | 0 | Break only on write cycles |
| | 1 | Break on both read and write cycles |

- Bits 1 and 0 (operand size select (SZ1, SZ0)): SZ1, SZ0 select bus cycle operand size as a break condition.

| Bit 1: SZ1 | Bit 0: SZ0 | Description |
|---|---|---|
| 0 | 0 | Operand size is not a break condition (initial value) |
| | 1 | Break on byte access |
| 1 | 0 | Break on word access |
| | 1 | Break on long word access |

Note: When setting to break on an instruction fetch, set the SZ0 bit to 0. All instructions will be considered to be accessed as words (even those instructions in on-chip memory for which two instructions can be fetched simultaneously in a single bus cycle). Instruction fetch is by word access and CPU/DMAC data access is by the specified operand size. They are not determined by the bus width of the space being accessed.

## 6.3 Operation

### 6.3.1 Flow of the User Break Operation

The flow from setting of break conditions to user break interrupt exception processing is described below.

1. Break conditions are set in break address register (BAR), break address mask register (BAMR), and the break bus cycle register (BBR). Set the break address in the BAR, the address bits to be masked in the BAMR and the type of breaking bus cycle in the BBR. When even one of the BBR groups (CPU cycle/DMA cycle select bits (CD1, CD0), instruction fetch/data access select bits (ID1, ID0), read/write select bits (RW1, RW0)) is set to 00 (no user break interrupt), there will be no user break even when all other conditions are consistent. To use a user break interrupt, set conditions for all three pairs.
2. The UBC checks to see if the set conditions are satisfied, using the system shown in figure 6.2. When the break conditions are satisfied, the UBC sends a user break interrupt request to the interrupt controller.

3. When receiving the user break interrupt request, the interrupt controller checks its priority level. The user break interrupt has priority level 15, so it is accepted only if the interrupt mask level in bits I3–I0 in the status register (SR) is 14 or lower. When the I3–I0 bit level is 15, the user break interrupt cannot be accepted but it is held pending until user break interrupt exception processing is carried out. NMI exception processing sets I3–I0 to level 15, so a user break cannot occur during the NMI service routine unless the NMI service routine itself begins by reducing I3–I0 to level 14 or lower. Section 5, Interrupt Controller, described the handling of priority levels in greater detail.
4. The INTC sends a request signal for a user break interrupt to the CPU. When the CPU receives it, it starts user break interrupt exception processing. Section 5.4, Interrupt Operation, describes interrupt exception processing in more detail.

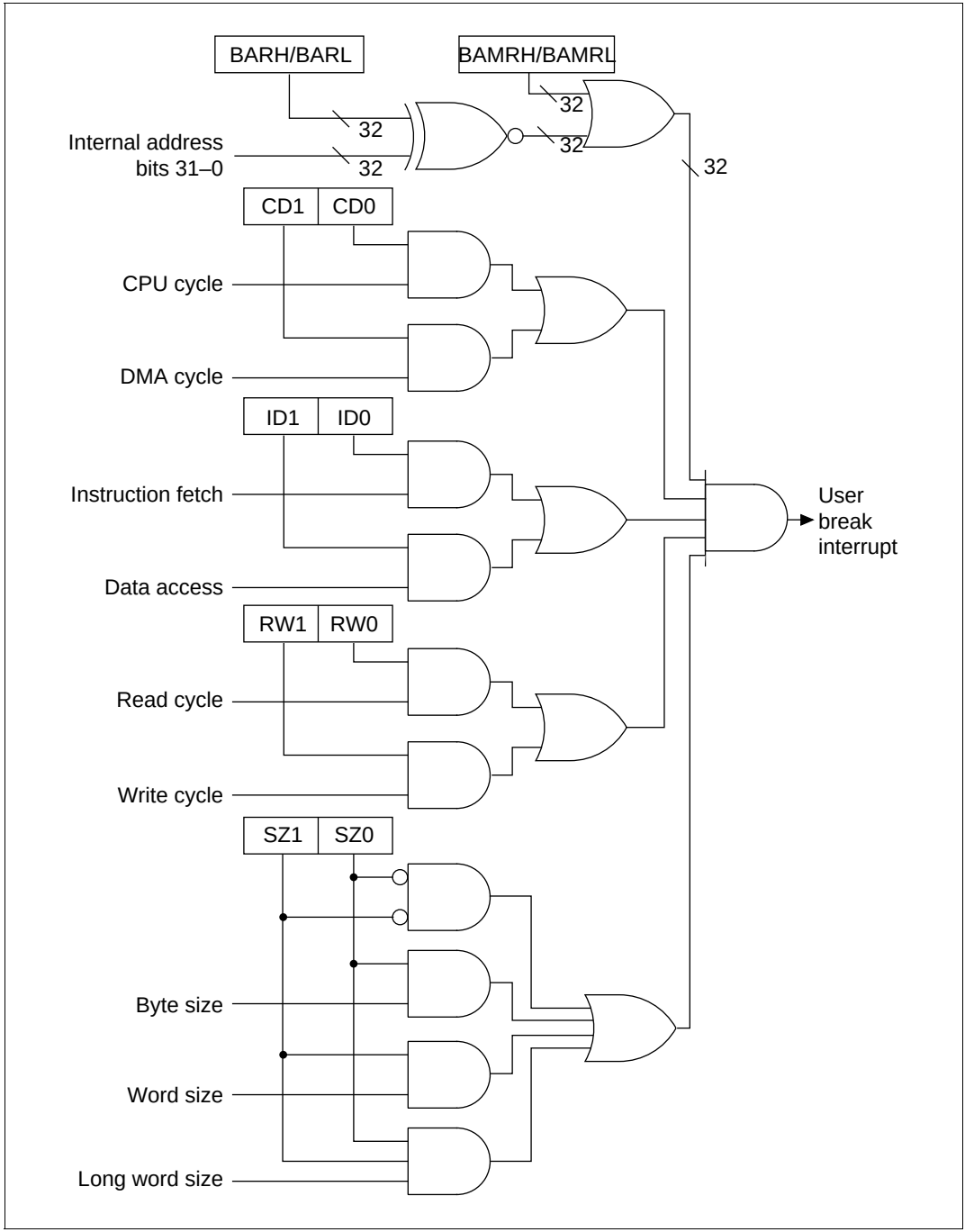



**Figure 6.2 Break Condition Logic**

### 6.3.2 Break on Instruction Fetch Cycles to On-Chip Memory

On-chip memory (on-chip ROM and RAM) is always accessed 32 bits each bus cycle. Two instructionsare therefore fetched in a bus cycle from on-chip memory . Although only a single bus cycle occurs for the two-instruction fetch, a break can be set on either instruction by placing the corresponding address in the break address registers (BAR). In other words, to break the second of the two instructions fetched, set its start address in the BAR. The break will then occur after the first instruction executes.

### 6.3.3 Program Counter (PC) Value Saved in User Break Interrupt Exception Processing

**Break on Instruction Fetch:** The program counter (PC) value saved in user break interrupt exception processing for an instruction fetch is the address set as the break condition. The user break interrupt is generated before the fetched instruction is executed. If a break condition is set on the fetch cycle of a delayed slot instruction immediately following a delayed branch instruction or on the fetch cycle of an instruction that follows an interrupt-disabling instruction, however, the user break interrupt is not accepted immediately, so the instruction is executed. The user break interrupt is not accepted until immediately after that instruction. The PC value that will be saved is the start address of the next instruction that is able to accept the interrupt.

**Break on Data Access (CPU/DMAC):** The program counter (PC) value is the top address of the next instruction after the last executed instruction at the time when the user break exception processing is activated. When data access (CPU/DMAC) is set as a break condition, the place where the break will occur cannot be specified exactly. The break will occur at the instruction fetched close to where the data access that is to receive the break occurs.

## 6.4 Setting User Break Conditions

**CPU Instruction Fetch Bus Cycle:**

- Register settings: BARH = H'0000, BARL = H'0404, BBR = H'0054

  Conditions set: Address = H'00000404, Bus cycle = CPU, instruction fetch, read (operand size not included in conditions)

  A user break interrupt will occur immediately before the instruction at address H'00000404. If the instruction at address H'00000402 can accept an interrupt, the user break exception processing will be executed after that instruction is executed. The instruction at H'00000404 will not be executed. The value saved to PC is H'00000404.

- Register settings: BARH = H'0015, BARL = H'389C, BBR = H'0058

  Conditions set: Address = H'0015389C, Bus cycle = CPU, instruction fetch, write (operand size not included in conditions)

  No user break interrupt occurs, because no instruction fetch cycle is ever a write cycle.

- Register settings: BARH = H'0003, BARL = H'0147, BBR = H'0054

  Conditions set: Address = H'00030147, Bus cycle = CPU, instruction fetch, read (operand size not included in conditions)

  No user break interrupt occurs, because instructions are always fetched from even addresses. If the first fetched address after a branch is odd and a user break is set on this address, however, user break exception processing will be carried out after address error exception processing.

**CPU Data Access Bus Cycle:**

- Register settings: BARH = H'0012, BARL = H'3456, BBR = H'006A

  Conditions set: Address = H'00123456, Bus cycle = CPU, data access, write, word

  A user break interrupt occurs when word data is written to address H'00123456.

- Register settings: BARH = H'00A8, BARL = H'0391, BBR = H'0066

  Conditions set: Address = H'00A80391, Bus cycle = CPU, data access, read, word

  No user break interrupt occurs, because word data access is always to an even address.

**DMA Cycle:**

- Register setting: BARH = H'0076, BARL = H'BCDC, BBR = H'00A7

  Conditions set: Address = H'0076BCDC, Bus cycle = DMA, data access, read, long word

  A user break interrupt occurs when long word data is read from address H'0076BCDC.

- Register setting: BARH = H'0023, BARL = H'45C8, BBR = H'0094

  Conditions set: Address = H'002345C8, Bus cycle = DMA, instruction fetch, read (operand size not included)

  No user break interrupt occurs, because a DMA cycle includes no instruction fetch.

## 6.5 Notes

### 6.5.1 On-Chip Memory Instruction Fetch

Two instructions are simultaneously fetched from on-chip memory. If a break condition is set on the second of these two instructions but the contents of the UBC break condition registers are changed so as to alter the break condition immediately after the first of the two instructions is fetched, a user break interrupt will still occur when the second instruction is fetched.

### 6.5.2 Instruction Fetch at Branches

When a conditional branch instruction or TRAPA instruction causes a branch, instructions are fetched and executed as follows:

1. Conditional branch instruction, branch taken: BT, BF

   Instruction fetch cycles: Conditional branch fetch → Next-instruction overrun fetch → Next-instruction overrun fetch → Branch destination fetch

   Instruction execution: Conditional branch instruction execution → Branch destination instruction execution

2. TRAPA instruction, branch taken: TRAPA

   Instruction fetch cycles: TRAPA instruction fetch → Next-instruction overrun fetch → Next-instruction overrun fetch → Branch destination fetch

   Instruction execution: TRAPA instruction execution → Branch destination instruction execution

When a conditional branch instruction or TRAPA instruction causes a branch, the branch destination will be fetched after the next instruction or the one after that does an overrun fetch. When the next instruction or the one after that is set as a break condition, a branch will result in the generation of a user break interrupt at the next instruction or the instruction after that, neither of which instructions will be executed.

### 6.5.3 Instruction Fetch Break

If a break is attempted at the task A return destination instruction fetch, task B is activated before the UBC interrupt by interrupt B generated during task A processing, and the UBC interrupt is handled after the interrupt B exception handling.

(1) Cause

The SH7032/SH7034 chip operates as follows.

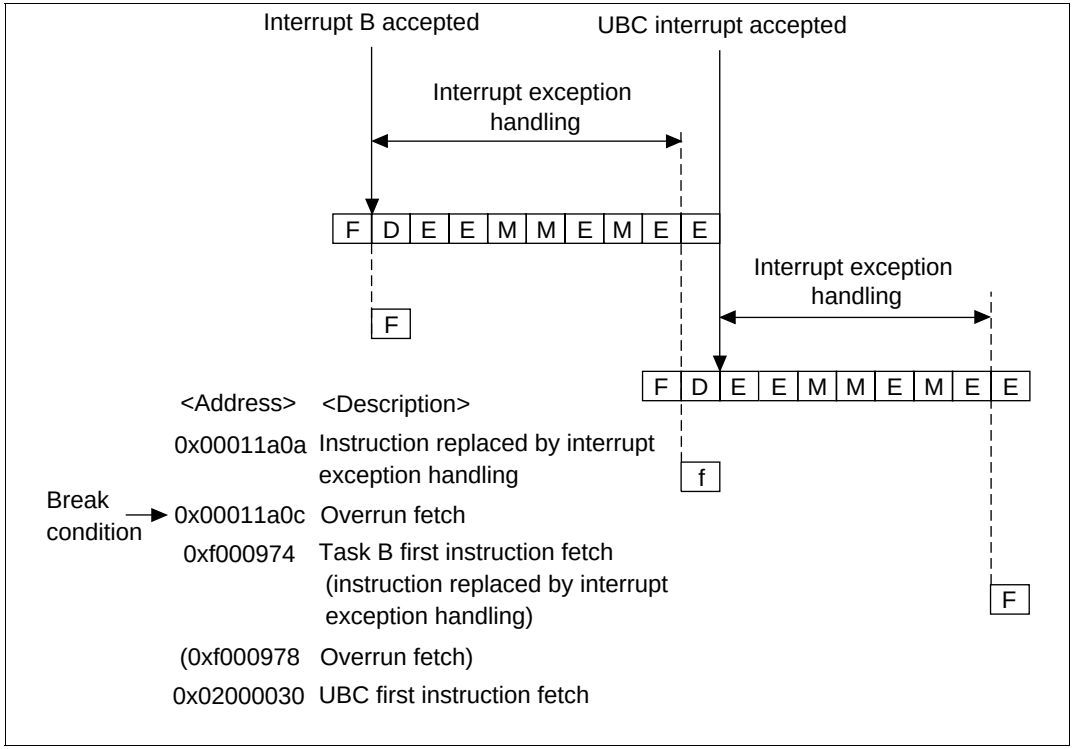



**Figure 6.3 UBC Operation**

It actually takes at least two cycles for the UBC interrupt generated by the address 0x00011a0c instruction fetch cycle to be sent to the interrupt controller and interrupt exception handling to begin. However, as shown in figure 6.3, when the UBC interrupt is generated, previously generated interrupt B initiated by task B is accepted first, and the UBC interrupt is accepted after completion of the interrupt B exception handling.

(2) Remedy

There is no way of preventing this operation by hardware. A software solution, such as the use of a flag, must be employed.

# Section 7 Clock Pulse Generator (CPG)

## 7.1 Overview

The SuperH microcomputer has a built-in clock pulse generator (CPG) that supplies the LSI and external devices with a clock pulse. The CPG makes the LSI run at the oscillation frequency of the crystal resonator. The CPG consists of an oscillator and a duty cycle correcting circuit (figure 7.1). The CPG can be made to generate a clock signal by connecting it to a crystal resonator or by inputting an external clock. (The CPG is halted in standby mode.)

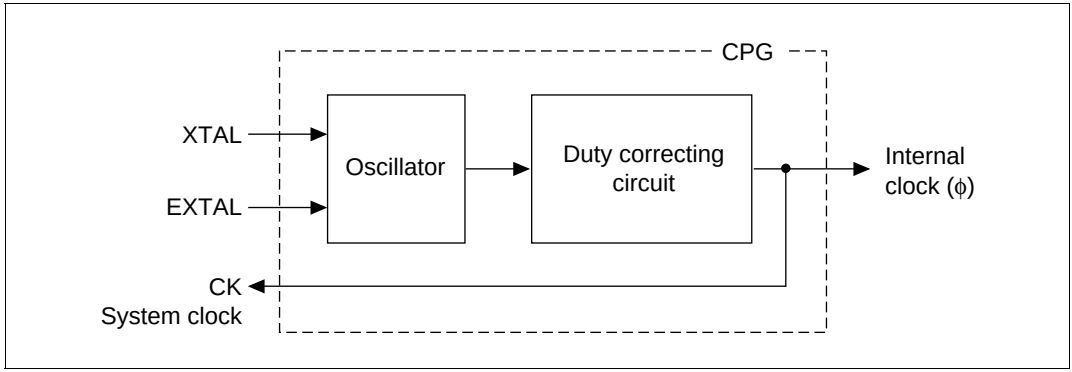



**Figure 7.1 Block Diagram of the Clock Pulse Generator**

## 7.2 Clock Source

Clock pulses can be supplied from a connected crystal resonator or an external clock.

### 7.2.1 Connecting a Crystal Resonator

**Circuit Configuration:** A crystal resonator can be connected as shown in figure 7.2. Use the damping resistance Rd shown in table 7.1. Use an AT-cut parallel resonating crystal with a frequency equal to the system clock (CK) frequency. Connect load capacitors ($C_{L1}$ and $C_{L2}$) as shown in the figure. The clock pulse produced by the crystal resonator and internal pulse generator is sent to the duty cycle correction circuit where its duty cycle is corrected. It is then supplied to the LSI and to external devices.

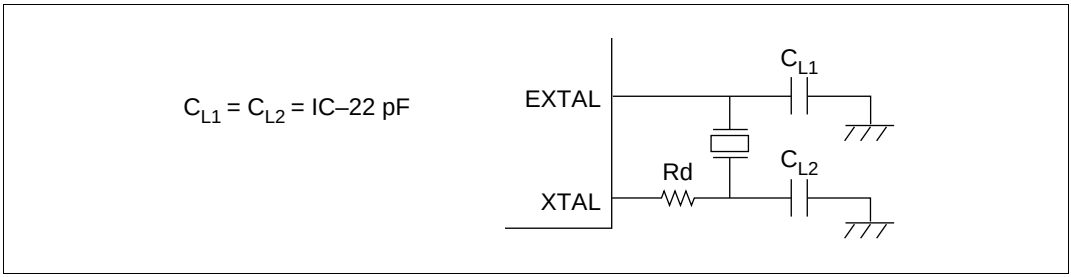



**Figure 7.2 Connection of the Crystal Resonator (Example)**

**Table 7.1 Damping Resistance**

| Frequency [MHz] | 2 | 4 | 8 | 12 | 16 | 20 |
|---|---|---|---|---|---|---|
| Rd [Ω] | 1k | 500 | 200 | 0 | 0 | 0 |

**Crystal Resonator:** Figure 7.3 shows an equivalent circuit of the crystal resonator. Use a crystal resonator with the characteristics listed in table 7.2.

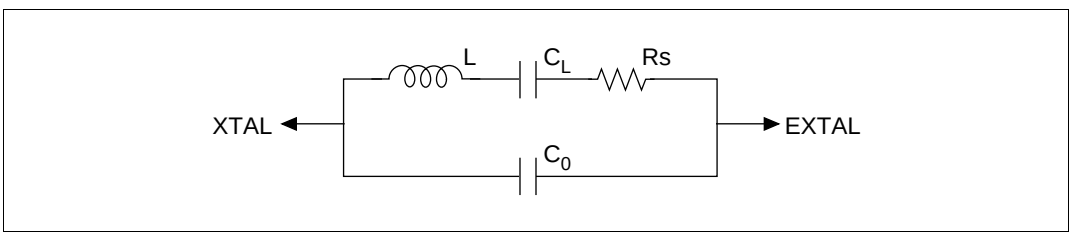



**Figure 7.3 Crystal Resonator Equivalent Circuit**

**Table 7.2 Crystal Resonator Parameters**

| | Frequency (MHz) | | | | | |
|---|---|---|---|---|---|---|
| **Parameter** | **2** | **4** | **8** | **12** | **16** | **20** |
| Rs max [Ω] | 500 | 120 | 80 | 60 | 50 | 40 |
| Co max [pF] | 7 | 7 | 7 | 7 | 7 | 7 |

Value to be determined (TBD)

### 7.2.2 External Clock Input

An external clock signal can be input at the EXTAL pin as shown in figure 7.6. The XTAL pin should be left open. The frequency must be equal to the system clock (CK) frequency. The specifications for the waveform of the external clock input are given below. Make the external clock frequency the same as the system clock (CK).

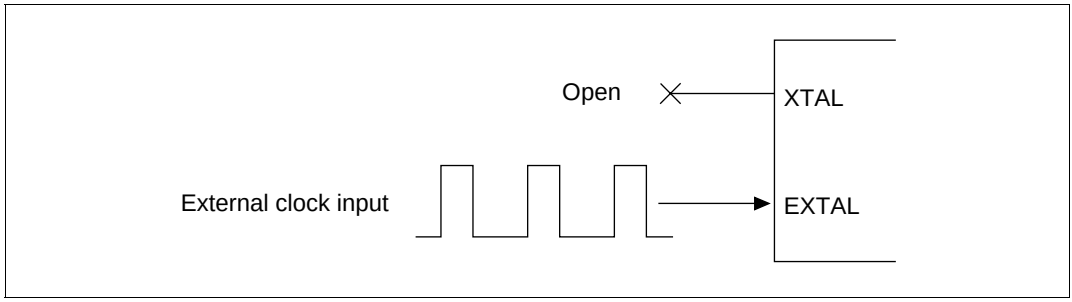



**Figure 7.4 External Clock Input Method**

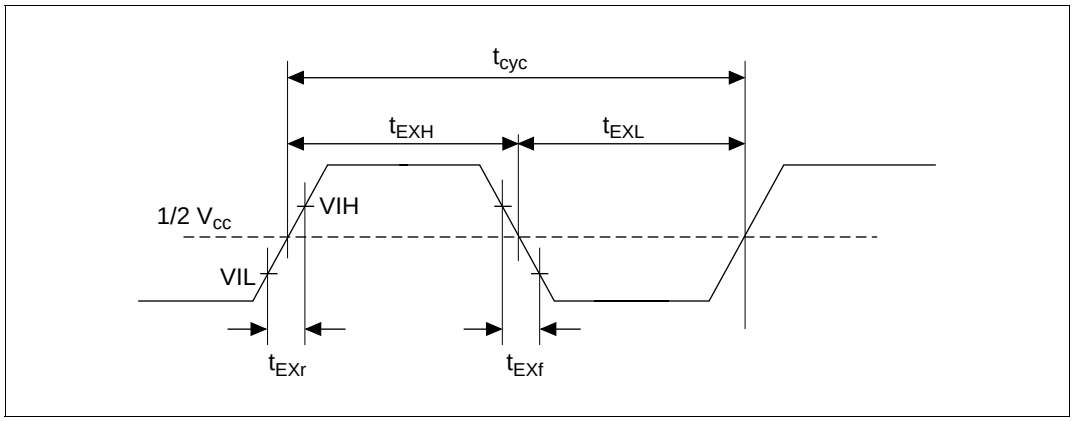



**Figure 7.5 Input Clock Waveform**

**Table 7.3 Input Clock Specifications**

| | **5 V Specifications (fmax = 20 MHz)** | **3.3 V Specifications (fmax = 12.5 MHz)** | **Units** |
|---|---|---|---|
| $t_{EXr/f}$ ($V_{IL}$–$V_{IH}$) | Max = 5 | Max = 10 | ns |
| $t_{EXH/L}$ (1/2 $V_{CC}$ standard) | Min = 10 | Min = 20 | ns |

## 7.3 Usage Notes

**Board design:** When designing the board, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins. Route no other signal lines near the XTAL and EXTAL pin signal lines to prevent induction from interfering with correct oscillation. See figure 7.6.

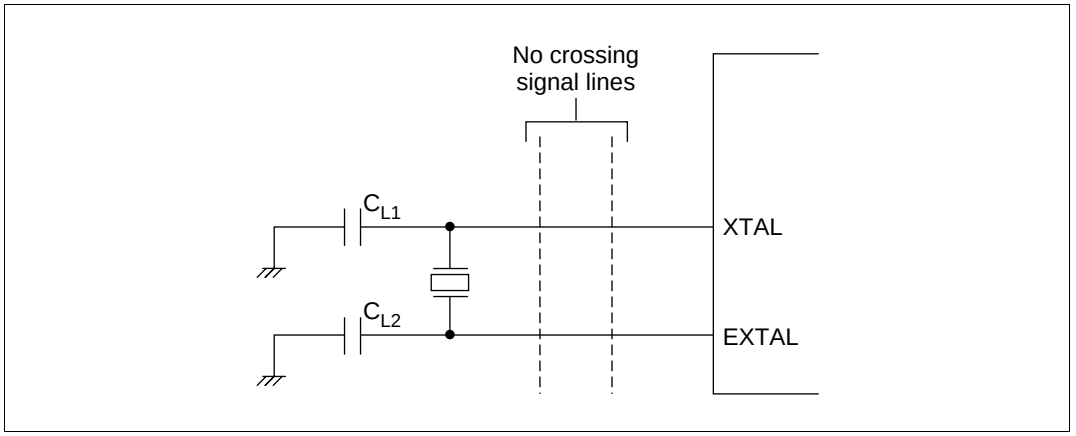



**Figure 7.6 Precaution on Oscillator Circuit Board Design**

**Duty cycle correction circuit:** Duty cycle corrections are conducted for an input clock over 5 MHz. Duty cycles may not be corrected if under 5 MHz, but AC characteristics for the high-level pulse width ($t_{CH}$) and low-level pulse width ($t_{CL}$) of the clock are satisfied, and the LSI will operate normally. Figure 7.7 shows the standard characteristics of a duty cycle correction.

This duty cycle correction circuit is not for correcting the input clock's transient fluctuations and jutters.

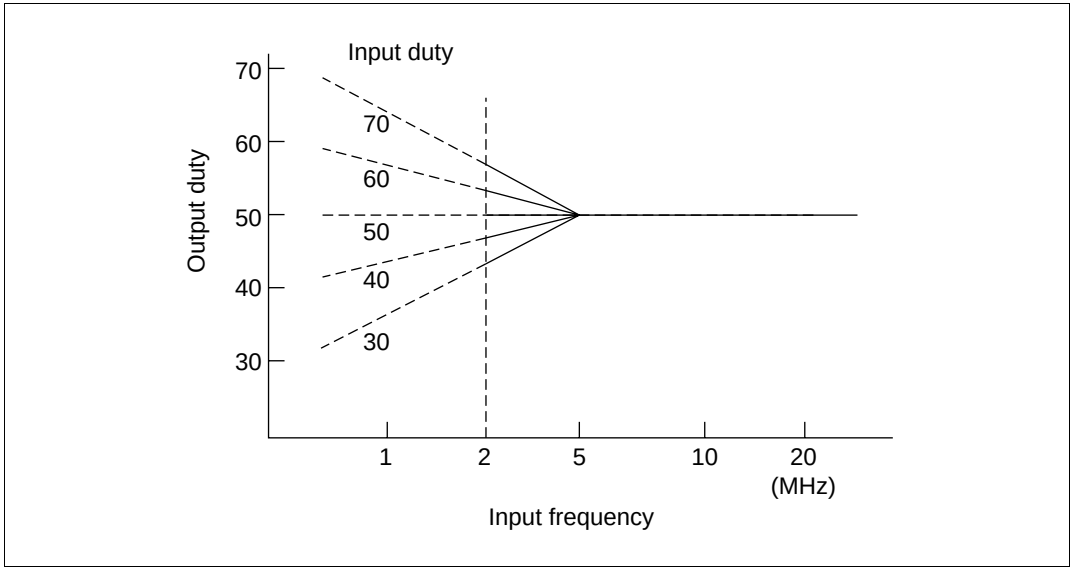



**Figure 7.7 Duty Cycle Correction Circuit Standard Characteristics**

# Section 8 Bus State Controller (BSC)

## 8.1 Overview

The bus state controller (BSC) divides address space and outputs control signals for all kinds of memory and peripheral LSIs. BSC functions enable the LSI to link directly with DRAM, SRAM, ROM, and peripheral LSIs without the use of external circuits, simplifying system design and allowing high-speed data transfers in a compact system.

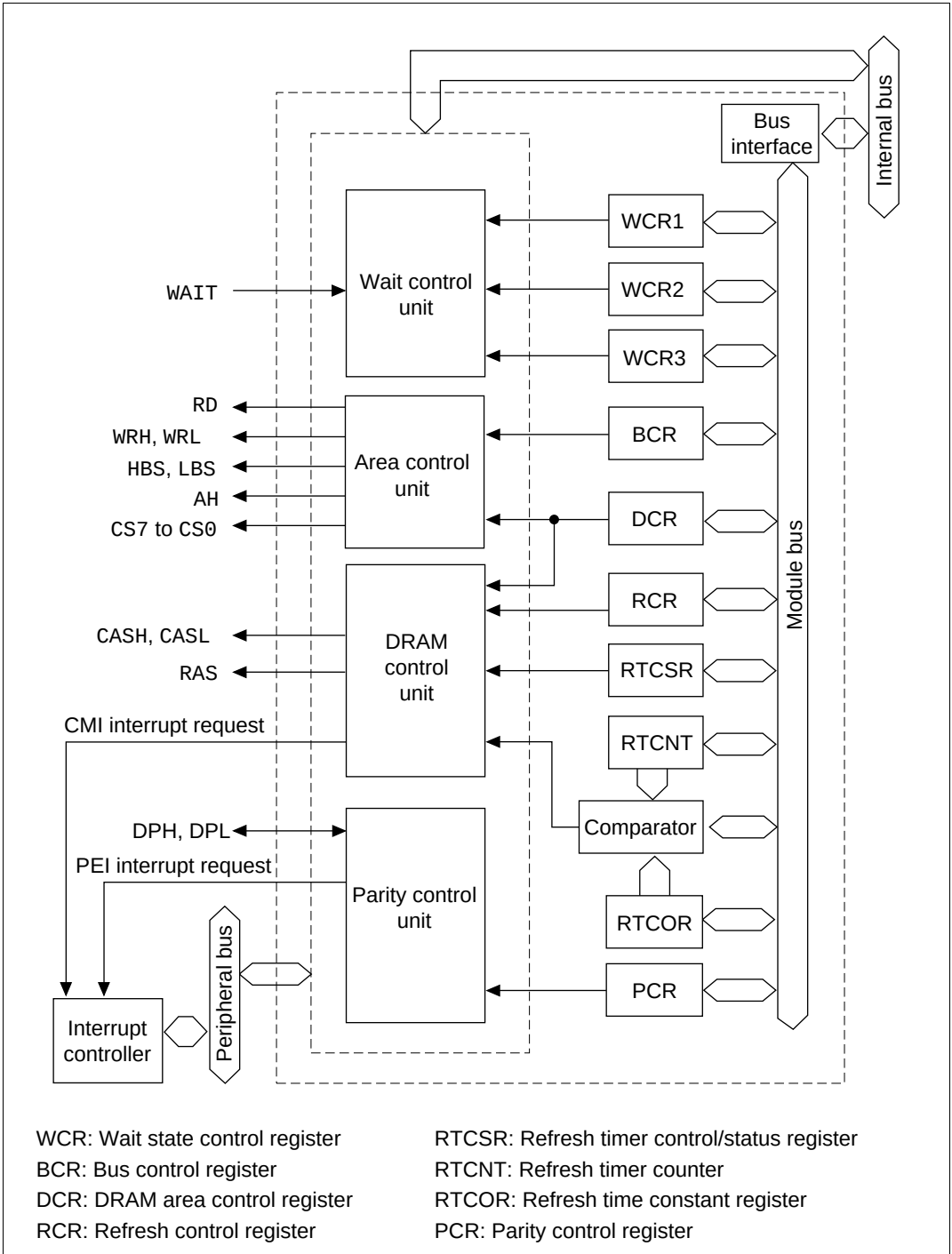
### 8.1.1 Features

The BSC has the following features.

- Address space is divided into eight areas
  - A maximum 4-Mbyte of linear address space for each of eight areas, 0–7 (area 1 can be up to 16-Mbyte linear space when set for DRAM) (The space that can actually be used varies with the type of memory connected)
  - Bus width (8 bits or 16 bits) can be selected by access address
  - On-chip ROM and RAM is accessed in one cycle (32 bits wide)
  - Wait states can be inserted using the $\overline{\text{WAIT}}$ pin
  - Wait state insertion can be controlled through software. Register settings can be used to specify the insertion of 1–4 cycles for areas 0, 2, and 6 (long wait function)
  - The type of memory connected can be specified for each area.
  - Outputs control signals for accessing the memory and peripheral LSIs connected to the area
- Direct interface to DRAM
  - Multiplexes row/column addresses according to DRAM capacity
  - Two types of byte access signals (dual-CAS system and dual-WE system)
  - Supports burst operation (high-speed page mode)
  - Supports CAS-before-RAS refresh and self-refresh
- Access control for all memory and peripheral LSIs
  - Address/data multiplex function
- Parallel execution of external writes and the like with internal access (warp mode)
- Supports parity check and generation for data bus
  - Odd parity/even parity selectable
  - Interrupt request generated for parity error (PEI interrupt request signal)
- Refresh counter can be used as an 8-bit interval timer
  - Interrupt request generated at compare match (CMI interrupt request signal)

### 8.1.2 Block Diagram

Figure 8.1 shows the block diagram of the bus state controller.




WCR: Wait state control register
BCR: Bus control register
DCR: DRAM area control register
RCR: Refresh control register
RTCSR: Refresh timer control/status register
RTCNT: Refresh timer counter
RTCOR: Refresh time constant register
PCR: Parity control register

**Figure 8.1 BSC Block Diagram**

### 8.1.3 Pin Configuration

Table 8.1 shows the BSC pin configuration.

**Table 8.1 Pin Configuration**

| Name | Abbreviation | I/O | Function |
|---|---|---|---|
| Chip select 7–0 | CS7–CS0 | O | Chip select signal that indicates the area being accessed |
| Read | RD | O | Strobe signal that indicates the read cycle |
| High write | WRH | O | Strobe signal that indicates write cycle to upper 8 bits |
| Low write | WRL | O | Strobe signal that indicates write cycle to lower 8 bits |
| Write | WR*[1] | O | Strobe signal that indicates write cycle |
| High byte strobe | HBS*[2] | O | Strobe signal that indicates access to upper 8 bits |
| Low byte strobe | LBS*[3] | O | Strobe signal that indicates access to lower 8 bits |
| Row address strobe | RAS | O | DRAM row address strobe signal |
| High column address strobe | CASH | O | Column address strobe signal for accessing the upper 8 bits of the DRAM |
| Low column address strobe | CASL | O | Column address strobe signal for accessing the lower 8 bits of the DRAM |
| Address hold | AH | O | Signal for holding the address for address/data multiplexing |
| Wait | WAIT | I | Wait state request signal |
| Address bus | A21–A0 | O | Address output |
| Data bus | AD15–AD0 | I/O | Data I/O. During address/data multiplexing, address output and data input/output. |
| Data bus parity high | DPH | I/O | Parity data I/O for upper byte |
| Data bus parity low | DPL | I/O | Parity data I/O for lower byte |

Notes:
1. Doubles with the WRL pin. (Selected by the BAS bit of the BCR. See section 8.2.1, Bus Control Register, for details.
2. Doubles with the A0 pin. (Selected by the BAS bit of the BCR. See section 8.2.1, Bus Control Register, for details.
3. Doubles with the WRH pin. (Selected by the BAS bit of the BCR. See section 8.2.1, Bus Control Register, for details.

### 8.1.4 Register Configuration

The BSC has ten registers (listed in table 8.2) which control space division, wait states, DRAM interface, and parity check.

**Table 8.2 Register Configuration**

| Name | Abbr. | R/W | Initial Value | Address*1 | Bus width |
|---|---|---|---|---|---|
| Bus control register | BCR | R/W | H'0000 | H'5FFFFA0 | 8,16,32 |
| Wait state control register 1 | WCR1 | R/W | H'FFFF | H'5FFFFA2 | 8,16,32 |
| Wait state control register 2 | WCR2 | R/W | H'FFFF | H'5FFFFA4 | 8,16,32 |
| Wait state control register 3 | WCR3 | R/W | H'F800 | H'5FFFFA6 | 8,16,32 |
| DRAM area control register | DCR | R/W | H'0000 | H'5FFFFA8 | 8,16,32 |
| Parity control register | PCR | R/W | H'0000 | H'5FFFFAA | 8,16,32 |
| Refresh control register | RCR | R/W | H'0000 | H'5FFFFAC | 8,16,32*2 |
| Refresh timer control/status register | RTCSR | R/W | H'0000 | H'5FFFFAE | 8,16,32*2 |
| Refresh timer counter | RTCNT | R/W | H'0000 | H'5FFFFB0 | 8,16,32*2 |
| Refresh time constant register | RTCOR | R/W | H'00FF | H'5FFFFB2 | 8,16,32*2 |

Notes: 1. Only the values of bits A27–A24 and A8–A0 are valid; bits A23–A9 are ignored. For details on the register addresses, see section 8.3.5, Description of Areas.
2. Write only with word transfer instructions. See section 8.2.11, Register Access, for details on writing.

### 8.1.5 Overview of Areas

The SH microprocessors have 32-bit address spaces on the architecture, but the top 4 bits are ignored. Table 8.3 outlines the division of space. As shown, the space is divided into areas 0–7 by the value of the top addresses.

Each area is allocated a specific type of space. When the area is accessed, a strobe signal that matches the type of area space is generated. This allocates peripheral LSIs and memory devices according to the type of the area spaces and allows them to be directly linked to this LSI. Some areas are of a fixed type based on their address while others can be selected in registers.

Area 0 can be used as an on-chip ROM space or external memory space. Area 1 can be used as DRAM space or external memory space. DRAM space enables direct connection to DRAM and outputs $\overline{RAS}$, $\overline{CAS}$ and multiplexed addresses. Areas 2–4 can only be used as external memory space. Area 5 can be used as on-chip peripheral module space or external memory space. Area 6 can be used as address/data multiplexed I/O space or external memory space. For address/data multiplexed I/O space, an address and data are multiplexed and input/output from AD15–AD0 pins. Area 7 can be used as on-chip RAM space or external memory space.

The bus width of the data bus is basically switched between 8 bit and 16 bit by the value of address bit A27. For the following areas, however, the bus width is determined by conditions other than the A27 bit value.

- On-chip ROM space in area 0: Always 32 bits
- External memory space in area 0: 8 bits when MD0 pin is 0, 16 bits when the pin is 1
- On-chip peripheral module space in area 5: 8 bits when the A8 address bit is 0, 16 bits when it is 1
- Area 6: If A27 = 0, area 6 is 8 bits when the A14 address bit is 0, 16 bits when A14 is 1
- On-chip RAM space in area 7: Always 32 bits

See table 8.6 in section 8.3, Address Space Subdivision, for more information on how the space is divided.

## 8.2 Register Descriptions

### 8.2.1 Bus Control Register (BCR)

The bus control register (BCR) is a 16-bit read/write register that selects the functions of areas and status of bus cycles. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | DRAME | IOE | WARP | RDDTY | BAS | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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: | — | — | — | — | — | — | — | — |

- Bit 15 (DRAM enable bit (DRAME)): DRAME selects whether area 1 is used as an external memory space or DRAM space. 0 sets it for external memory space and 1 sets it for DRAM space. The setting of the DRAM area control register is valid only when this bit is set to 1.

| Bit 15: DRAME | Description |
|---|---|
| 0 | Area 1 is external memory space (initial value) |
| 1 | Area 1 is a DRAM space |

- Bit 14 (multiplexed I/O enable bit (IOE)): IOE selects whether area 6 is used as external memory space or an address/data multiplexed I/O area. 0 sets it for external memory space and 1 sets it for address/data multiplexed I/O space. With address/data multiplexed I/O space, address and data are multiplexed and input/output is from AD15–AD0.

| Bit 14: IOE | Description |
|---|---|
| 0 | Area 6 is external memory space (initial value) |
| 1 | Area 6 is an address/data multiplexed I/O area |

- Bit 13 (warp mode bit (WARP)): WARP selects warp or normal mode. 0 sets it for normal mode and 1 sets it for warp mode. In warp mode, some external accesses are carried out in parallel with internal access.

| Bit 13: WARP | Description |
|---|---|
| 0 | Normal mode: External and internal accesses are not simultaneously performed (initial value) |
| 1 | Warp mode: External and internal accesses are simultaneously performed |

- Bit 12 (RD duty (RDDTY)): RDDTY selects 35% or 50% of the T1 state as the high-level duty cycle ratio of signal $\overline{RD}$. 0 sets it for 50%, 1 sets it for 35%. Only set to 1 when the operating frequency is a minimum of 10 MHz.

| Bit 12: RDDTY | Description |
|---|---|
| 0 | RD signal high-level duty cycle is 50% of T1 state (initial value) |
| 1 | RD signal high-level duty cycle is 35% of T1 state |

- Bit 11 (byte access select (BAS)): BAS selects whether byte access control signals are $\overline{WRH}$, $\overline{WRL}$, and A0, or $\overline{LBS}$, $\overline{WR}$ and $\overline{HBS}$ during word space accesses. When this bit is cleared to 0, $\overline{WRH}$, $\overline{WRL}$, and A0 signals are valid; when set to 1, $\overline{LBS}$, $\overline{WR}$, and $\overline{HBS}$, signals are valid.

| Bit 11: BAS | Description |
|---|---|
| 0 | WRH, WRL, and A0 enabled (initial value) |
| 1 | LBS, WR, and HBS enabled |

- Bits 10–0 (reserved): These bits always read as 0. The write value should always be 0.

### 8.2.2 Wait State Control Register 1 (WCR1)

Wait state control register 1 is a 16-bit read/write register that controls the number of states for accessing each area and the whether wait states are used. WCR1 is initialized to H'FFFF by a power-on reset. It is not initialized by a manual reset or by the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | RW7 | RW6 | RW5 | RW4 | RW3 | RW2 | RW1 | RW0 |
| 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: | — | — | — | — | — | — | WW1 | — |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | — | — | — | — | — | — | R/W | — |

- Bits 15–8 (wait state control during read (RW7–RW0)): RW7–RW0 determine the number of states in read cycles for each area and whether or not to sample the signal input from the $\overline{\text{WAIT}}$ pin. Bits RW7–RW0 correspond to areas 7–0, respectively. If a bit is cleared to 0, the $\overline{\text{WAIT}}$ signal is not sampled during the read cycle for the corresponding area. If it is set to 1, sampling takes place.

  For the external memory spaces of areas 1, 3–5, and 7, read cycles are completed in one state when the corresponding bits are cleared to 0. When they are set to 1, the number of wait states is 2 plus the $\overline{\text{WAIT}}$ signal value. For the external memory space of areas 0, 2, and 6, read cycles are completed in one state plus the number of long wait states (set in wait state controller 3 (WCR3)) when the corresponding bits are cleared to 0. When they are set to 1, the number of wait states is 1 plus the long wait state; when the $\overline{\text{WAIT}}$ signal is at low level as well, a wait state is inserted.

  The DRAM space (area 1) finishes the column address output cycle in one state (short pitch) when the RW1 bit is 0, and in 2 states plus the $\overline{\text{WAIT}}$ signal value (long pitch) when RW1 is 1. When RW1 is set to 1, the number of wait states selected in wait state insertion bits 1 and 0 (RLW0 and RLW1) for CAS-before-RAS (CBR) refresh of the refresh control register (RCR) are inserted during the CBR refresh cycle, regardless of the status of the $\overline{\text{WAIT}}$ signal.

  The read cycle of the address/data multiplexed I/O space (area 6) is 4 states plus the wait states from the $\overline{\text{WAIT}}$ signal, regardless of the setting of the RW6 bit. The read cycle of the on-chip peripheral module space (area 5) finishes in 3 states, regardless of the setting of the RW5 bit, and the $\overline{\text{WAIT}}$ signal is not sampled. The read cycles of on-chip ROM (area 0) and on-chip RAM (area 7) finish in 1 state, regardless of the settings of bits RW0 and RW7. The $\overline{\text{WAIT}}$ signal is not sampled for either.

  Table 8.3 summarizes read cycle state information.

**Table 8.3 Read Cycle State Description**

| | | Read Cycle States | | | | |
|---|---|---|---|---|---|---|
| | | External Memory Space | | | Internal space | |
| Bits 15–8: RW7–RW0 | WAIT Pin Input Signal | External Memory Space | DRAM Space | Multi-Plexed I/O | On-chip Peripheral Module | On-chip ROM and RAM |
| 0 | Not sampled during read cycle*1 | • Areas 1, 3–5,7: 1 state, fixed<br>Areas 0, 2, 6: 1 state + long wait state | Column address cycle: 1 state, fixed (short pitch) | 4 states + wait states from WAIT | 3 states, fixed | 1 state, fixed |
| 1 | Sampled during read cycle (initial value) | Areas 1, 3–5, 7: 2 states + wait states from WAIT<br>Areas 0, 2, 6: 1 state + long wait state + wait state from WAIT | Column address cycle: 2 states + wait state from WAIT (long pitch)*2 | | | |

Notes: 1. Sampled in the address/data multiplexed I/O space
2. During a CBR refresh, the WAIT signal is ignored and the wait state from the RLW1 and RLW0 bits of RCR is inserted.

- Bits 7–2 (reserved): These bits always read as 1. The write value should always be 1.

- Bit 1 (wait state control during write (WW1)): WW1 determines the number of states in write cycles for the DRAM space (area 1) and whether or not to sample the $\overline{\text{WAIT}}$ signal. When the DRAM enable bit (DRAME) of the BCR is set to 1 and area 1 is being used as DRAM space, clearing WW1 to 0 makes the column address output cycle finish in 1 states (short pitch). When WW1 is set to 1, it finishes in 2 states plus the wait states from the $\overline{\text{WAIT}}$ signal (long pitch).

Note: Write 0 to WW1 only when area 1 is used as DRAM space (DRAME bit of BCR is 1). Never write 0 to WW1 when area 1 is used as external memory space (DRAME is 0).

| Bit 1: WW1 | DRAM Space (DRAME = 1) | Area 1's External Memory Space (DRAME = 0) |
|---|---|---|
| 0 | Column address cycle: 1 state (short pitch) | Setting inhibited |
| 1 | Column address cycle: 2 states + wait state from WAIT (long pitch) (initial value) | 2 states + wait state from WAIT |

- Bit 0 (reserved): This bit always reads 1. The write value should always be 1.

### 8.2.3 Wait State Control Register 2 (WCR2)

Wait state control register 2 is a 16-bit read/write register that controls the number of states for accessing each area with a DMA single address mode transfer and whether wait states are used. WCR2 is initialized to H'FFFF by a power-on reset. It is not initialized by a manual reset or by the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | DRW7 | DRW6 | DRW5 | DRW4 | DRW3 | DRW2 | DRW1 | DRW0 |
| 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: | DWW7 | DWW6 | DWW5 | DWW4 | DWW3 | DWW2 | DWW1 | DWW0 |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bits 15–8 (wait state control during single-mode DMA transfer (DRW7–DRW0)): DRW7–DRW0 determine the number of states in single-mode DMA memory read cycles for each area and whether or not to sample the $\overline{\text{WAIT}}$ signal. Bits DRW7–DRW0 correspond to areas 7–0, respectively. If a bit is cleared to 0, the $\overline{\text{WAIT}}$ signal is not sampled during the single-mode DMA memory read cycle for the corresponding area. If it is set to 1, sampling takes place.

  For the external memory spaces of areas 1, 3–5, and 7, single-mode DMA memory read cycles are completed in one state when the corresponding bits are cleared to 0. When they are set to 1, the number of wait states is 2 plus the wait states from the $\overline{\text{WAIT}}$ signal. For the external memory space of areas 0, 2, and 6, single-mode DMA memory read cycles are completed in one state plus the long wait state number (set in wait state controller 3 (WCR3)) when the corresponding bits are cleared to 0. When they are set to 1, the number of wait states is 1 plus the long wait state; when the $\overline{\text{WAIT}}$ signal is at low level as well, a wait state is inserted.

  The DRAM space (area 1) finishes the column address output cycle in one state (short pitch) when the DRW1 bit is 0, and in 2 states plus the wait states from the $\overline{\text{WAIT}}$ signal (long pitch) when DRW1 is 1. The single-mode DMA memory read cycle of the address/data multiplexed I/O space (area 6) is 4 states plus the wait states from the $\overline{\text{WAIT}}$ signal, regardless of the setting of the DRW6 bit.

  Table 8.4 summarizes single-mode DMA memory read cycle state information.

**Table 8.4 Single-Mode DMA Memory Read Cycle States (External Memory Space) Description**

| | | Single-Mode DMA Memory Read Cycle States (External Memory Space) | | |
|---|---|---|---|---|
| **Bits 15–8: DRW7–DRW0** | **WAIT Pin Input Signal** | **External Memory Space** | **DRAM Space** | **Multiplexed I/O** |
| 0 | Not sampled during single-mode DMA memory read cycle* | Areas 1, 3–5,7: 1 state, fixed<br>Areas 0, 2, 6: 1 state + long wait state | Column address cycle: 1 state, fixed (short pitch) | 4 states + wait states from WAIT |
| 1 | Sampled during single-mode DMA memory read cycle (initial value) | Areas 1, 3–5, 7: 2 states + wait states from WAIT<br>Areas 0, 2, 6: 1 state + long wait state + Wait state from WAIT | Column address cycle: 2 states + wait state from WAIT (long pitch) | |

Note: Sampled in the address/data multiplexed I/O space.

- Bits 7–0 (single-mode DMA memory write wait state control (DWW7–DWW0)): DWW7–DWW0 determine the number of states in single-mode DMA memory write cycles for each area and whether or not to sample the $\overline{\text{WAIT}}$ signal. Bits DWW7–DWW0 correspond to areas 7–0, respectively. If a bit is cleared to 0, the $\overline{\text{WAIT}}$ signal is not sampled during the single-mode DMA memory write cycle for the corresponding area. If it is set to 1, sampling takes place.

  The number of states for areas accesses based on bit settings are the same as indicated for single-mode DMA memory read cycles. See bits 15–8, wait state control during single-mode DMA memory transfer (DRW7–DRW0), for details.

  Table 8.5 summarizes single-mode DMA memory write cycle state information.

**Table 8.5 Single-Mode DMA Memory Write Cycle States (External Memory Space) Description**

| | | Single-mode DMA Memory Write Cycle States (External Memory Space) | | |
|---|---|---|---|---|
| **Bits 15–8: DWW7–DWW0** | **WAIT Pin Input Signal** | **External Memory Space** | **DRAM Space** | **Multiplexed I/O** |
| 0 | Not sampled during single-mode DMA memory write cycle* | Areas 1, 3–5,7: 1 state, fixed<br>Areas 0, 2, 6: 1 state + long wait state | Column address cycle: 1 state, fixed (short pitch) | 4 states + wait state from WAIT |
| 1 | Sampled during single-mode DMA memory write cycle (initial value) | Areas 1, 3–5, 7: 2 states + wait state from WAIT<br>Areas 0, 2, 6: 1 state + long wait state + wait state from WAIT | Column address cycle: 2 states + wait state from WAIT (long pitch) | |

Note: Sampled in the address/data multiplexed I/O space.

### 8.2.4 Wait State Control Register 3 (WCR3)

Wait state control register 3 is a 16-bit read/write register that controls $\overline{\text{WAIT}}$ pin pull-up and the insertion of long wait states. WCR3 is initialized to H'F800 by a power-on reset. It is not initialized by a manual reset or by the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | WPU | A02LW1 | A02LW0 | A6LW1 | A6LW0 | — | — | — |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 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: | — | — | — | — | — | — | — | — |

- Bit 15 (wait pin pull-up control (WPU)): WPU controls whether the $\overline{\text{WAIT}}$ pin is pulled up or not. When cleared to 0, the pin is not pulled up; when set to 1, it is pulled up.

| Bit 15: WPU | Description |
|---|---|
| 0 | WAIT pin is not pulled up |
| 1 | WAIT pin is pulled up (initial value) |

- Bits 14 and 13 (long wait insertion in areas 0 and 2, bits 1, 0 (A02LW1 and A02LW0)): A02LW1 and A02LW0 select the long wait states to be inserted (1–4 states) when accessing external memory space of areas 0 and 2.

| Bit 14: A02LW1 | Bit 13: A02LW0 | Description |
|---|---|---|
| 0 | 0 | Inserts 1 state |
| | 1 | Inserts 2 states |
| 1 | 0 | Inserts 3 states |
| | 1 | Inserts 4 states (initial value) |

- Bits 12 and 11 (long wait insertion in area 6, bits 1, 0 (A6LW1 and A6LW0)): A6LW1 and A6LW0 select the long wait states to be inserted (1–4 states) when accessing external memory space of area 6.

| Bit 12: A6LW1 | Bit 11: A6LW0 | Description |
|---|---|---|
| 0 | 0 | Inserts 1 state |
| | 1 | Inserts 2 states |
| 1 | 0 | Inserts 3 states |
| | 1 | Inserts 4 states (initial value) |

- Bits 10–0 (reserved): These bits always read as 0. The write value should always be 0.

### 8.2.5 DRAM Area Control Register (DCR)

The DRAM area control register (DCR) is a 16-bit read/write register that selects the type of DRAM control signal, the number of precharge cycles, the burst operation mode and the use of address multiplexing. DCR settings are valid only when the DRAME bit of BCR is set to 1. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | CW2 | RASD | TPC | BE | CDTY | MXE | MXC1 | MXC0 |
| 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: | — | — | — | — | — | — | — | — |

- Bit 15 (dual-CAS or dual-WE select bit (CW2)): When accessing a 16-bit bus width space, CW2 selects the dual-CAS or the dual-WE method. When cleared to 0, the $\overline{\text{CASH}}$, $\overline{\text{CASL}}$, and $\overline{\text{WRL}}$ signals are valid ; when set to 1, the $\overline{\text{CASL}}$, $\overline{\text{WRH}}$, and $\overline{\text{WRL}}$ signals are valid. When accessing an 8-bit space, only $\overline{\text{CASL}}$ and $\overline{\text{WRL}}$ signals are valid, regardless of CW2 settings.

| Bit 15L: CW2 | Description |
|---|---|
| 0 | Dual-CAS: $\overline{\text{CASH}}$, $\overline{\text{CASL}}$, and $\overline{\text{WRL}}$ signals are valid (initial value) |
| 1 | Dual-WE: $\overline{\text{CASL}}$, $\overline{\text{WRH}}$, and $\overline{\text{WRL}}$ signals are valid |

- Bit 14 (RAS down (RASD)): When DRAM access pauses, RASD determines whether to keep $\overline{\text{RAS}}$ low while waiting for the next DRAM access (RAS down mode) or return it to high (RAS up mode). When cleared to 0, the $\overline{\text{RAS}}$ signal returns to high; when set to 1, it stays at low.

| Bit 14: RASD | Description |
|---|---|
| 0 | RAS up mode: Return $\overline{\text{RAS}}$ signal to high and wait for the next DRAM access (initial value) |
| 1 | RAS down mode: Keep $\overline{\text{RAS}}$ signal low and wait for the next DRAM access |

- Bit 13 (RAS precharge cycle count (TPC)): TPC selects whether the $\overline{\text{RAS}}$ signal precharge cycle ($T_P$) will be 1 state or 2. When TPC is cleared to 0, a 1-state precharge cycle is inserted; when 1 is set, a 2-state precharge cycle is inserted.

| Bit 13: TPC | Description |
|---|---|
| 0 | Inserts 1-state precharge cycle (initial value) |
| 1 | Inserts 2-state precharge cycle |

- Bit 12 (burst operation enable (BE)): BE selects whether or not to perform burst operation, a high speed page mode. When burst operation is not selected (0), the row address is not compared but instead is transferred to the DRAM every time and full access is performed. When burst operation is selected (1), row addresses are compared and burst operation with the same row address as the previous is performed (in this access, no row address is output and the column address and $\overline{CAS}$ signal alone are output).

| Bit 12: BE | Description |
|---|---|
| 0 | Normal mode: full access (initial value) |
| 1 | Burst operation: high-speed page mode |

- Bit 11 (CAS duty (CDTY)): CDTY selects 35% or 50% of the $T_C$ state as the high-level duty ratio of the signal $\overline{CAS}$ in the short-pitch access. When cleared to 0, the $\overline{CAS}$ signal high level duty is 50%; when set to 1, it is 35%. Only set to 1 when the operating frequency is a minimum of 10 MHz.

| Bit 11: CDTY | Description |
|---|---|
| 0 | $\overline{CAS}$ signal high level duty cycle is 50% of the $T_C$ state (initial value) |
| 1 | $\overline{CAS}$ signal high level duty cycle is 35% of the $T_C$ state |

- Bit 10 (multiplex enable bit (MXE)): MXE determines whether or not DRAM row and column addresses are multiplexed. When cleared to 0, addresses are not multiplexed; when set to 1, they are multiplexed.

| Bit 10: MXE | Description |
|---|---|
| 0 | Multiplex of row and column addresses disabled (initial value) |
| 1 | Multiplex of row and column addresses enabled |

- Bits 9 and 8 (multiplex shift count 1 and 0 (MXC1 and MXC0)): Shift row addresses downward by a certain number of bits (8–10) when row and column addresses are multiplexed (MXE = 1). Regardless of the MXE bit setting, these bits also select the range of row addresses compared in burst operation.

| Bit 9: MXC1 | Bit 8: MXC0 | Row Address Shift (MXE = 1) | Row Address Bits Compared (in burst operation) (MXE = 0 or 1) |
|---|---|---|---|
| 0 | 0 | 8 bits (initial value) | A8–A27 (initial value) |
| | 1 | 9 bits | A9–A27 |
| 1 | 0 | 10 bits | A10–A27 |
| | 1 | Reserved | Reserved |

Bits 7–0 (reserved): These bits always read as 0. The write value should always be 0.

### 8.2.6 Refresh Control Register (RCR)

The refresh control register (RCR) is a 16-bit read/write register that controls the start of refreshing and selects the refresh mode and the number of wait states during refresh. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by the standby mode.

To prevent RCR from being written incorrectly, it must be written by a different method from most other registers. A word transfer operation is used, H'5A is written in the top byte, and the actual data is written in the lower byte. For details, see section 8.2.11, Notes on Register Access.

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

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | RFSHE | RMODE | RLW1 | RLW0 | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | — | — | — | — |

- Bit 15–8 (reserved): These bits always read as 0.
- Bit 7 (refresh control (RFSHE)): RFSHE determines whether or not to perform DRAM refresh operations. When this bit is cleared to 0, no DRAM refresh control is performed and the refresh timer counter (RTCNT) can be used as an 8-bit interval timer. When set to 1, DRAM refresh control is performed.

| Bit 7: RFSHE | Description |
|---|---|
| 0 | Refresh control disabled. RTCNT can be used as an 8-bit interval timer. (initial value) |
| 1 | Refresh control enabled |

- Bit 6 (refresh mode (RMODE)): When DRAM refresh control is selected (RFSHE = 1), RMODE selects whether to perform CAS-before-RAS (CBR) refresh or self-refresh. When this bit is cleared to 0, a CBR refresh is performed at the cycle set in the refresh timer control/status register (RTCSR) and refresh time constant register (RTCOR). When set to 1, it the DRAM does a self-refresh. When refresh control is not selected (RFSHE = 0), the RMODE bit setting is not valid. When canceling self-refresh, set RMODE to 0 with RFSHE set to 1.

| Bit 6: RMODE | Description |
|---|---|
| 0 | CAS-before-RAS refresh (initial value) |
| 1 | Self-refresh |

- Bits 5 and 4—Insert wait states during CBR refresh bits 1 and 0 (RLW1, RLW0): These bits select the number of wait states to be inserted (1–4) during CAS-before-RAS refresh. When CBR refresh is performed and the RW1 bit of WCR1 is set to 1, the number of wait states selected in the RLW1 and RLW0 is inserted regardless of the $\overline{\text{WAIT}}$ signal. When the RW1 bit is cleared to 0, the RLW1 and RLW0 bit settings are ignored and no wait states are inserted.

| Bit 5: RLW1 | Bit 4: RLW0 | Description |
|---|---|---|
| 0 | 0 | Inserts 1 state (initial value) |
| | 1 | Inserts 2 states |
| 1 | 0 | Inserts 3 states |
| | 1 | Inserts 4 states |

- Bits 3–0 (reserved): These bits always read as 0. The write value should always be 0.

### 8.2.7 Refresh Timer Control/Status Register (RTCSR)

The refresh timer control/status register (RTCSR) is a 16-bit read/write register that selects the clock input to refresh timer counter (RTCNT) and controls compare match interrupts (CMI). It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by the standby mode.

To prevent RTCSR from being written incorrectly, it must be written by a different method from most other registers. A word transfer operation is used, H'A5 is written in the top byte and the actual data is written in the lower byte. For details, see section 8.2.11, Notes on Register Access.

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

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | CMF | CMIE | 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 | — | — | — |

- Bits 15–8 (reserved): These bits always read as 0.
- Bit 7 (compare match flag (CMF)): CMF is a flag that indicates whether the values of RTCNT and the refresh time constant register (RTCOR) match. When 0, the value of RTCNT and RTCOR do not match; when 1, the value of RTCNT and RTCOR match.

| **Bit 7: CMF** | **Description** |
|---|---|
| 0 | RTCNT does not equal the value of RTCOR (initial value)<br>To clear CMF, the CPU must read CMF after it has been set to 1, then write a 0 in this bit |
| 1 | Value RTCNT is equal to the value of RTCOR |

- Bit 6 (compare match interrupt enable (CMIE)): CMIE enables or disables the compare match interrupt (CMI) generated when CMF is set to 1 in RTCSR (RTCNT value = RTCOR value). When cleared to 0, CMI interrupt is disabled; when set to 1, it is enabled.

| **Bit 6: CMIE** | **Description** |
|---|---|
| 0 | Compare match interrupt request (CMI) is disabled (initial value) |
| 1 | Compare match interrupt request (CMI) is enabled |

- Bits 5–3 (clock select bits 2–0 (CKS2–CKS0)): CKS2–CKS0 select the clock input to RTCNT from among the seven types of clocks created by dividing the system clock (ϕ). When the input clock is selected with the CKS2–CKS0 bits, RTCNT starts to increment.

| Bit 5: CKS2 | Bit 4: CKS1 | Bit 3: CKS0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Clock input disabled (initial value) |
| | | 1 | ϕ/2 |
| | 1 | 0 | ϕ/8 |
| | | 1 | ϕ/32 |
| 1 | 0 | 0 | ϕ/128 |
| | | 1 | ϕ/512 |
| | 1 | 0 | ϕ/2048 |
| | | 1 | ϕ/4096 |

- Bits 2–0 (reserved): These bits always read as 0. The write value should always be 0.

### 8.2.8 Refresh Timer Counter (RTCNT)

The refresh timer counter (RTCNT) is a 16-bit read/write register that is used as an 8-bit upcounter that generates the refresh or interrupt request. When the input clock is selected by clock select bits 2–0 (CKS2–CKS0) in RTCSR, that clock makes the RTCNT start incrementing. When the values of RTCNT and the refresh time constant register (RTCOR) match, RTCNT is cleared to H'0000 and the CMF flag of the RTCSR is set to 1. When the RFSHE bit of the RCR is also set to 1, a CAS-before-RAS refresh is performed. When the CMIE bit of the RTCSR is also set to 1, a compare match interrupt (CMI) is generated.

Bits 15–8 are reserved bits and do not count. These bits always read as 0.

RTCNT is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by the standby mode.

To prevent RTCSR from being written incorrectly, it must be written by a different method from most other registers. A word transfer operation is used, H'69 is written in the top byte and the actual data is written in the lower byte. For details, see section 8.2.11, Register Access.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

### 8.2.9 Refresh Time Constant Register (RTCOR)

The refresh time constant register (RTCOR) is a 16-bit read/write register that sets the compare match cycle used with RTCNT. The values in RTCOR and RTCNT are constantly compared. When they match, the compare-match flag (CMF) is set in RTCNT and RTCSR is cleared to H'0000. If the bit RFSHE in RCR is set to 1 when this happens, a CAS before RAS (CBR) refresh is performed. When the CMIE bit of the RTCSR is also set to 1, a compare match interrupt (CMI) is generated.

Bits 15–8 are reserved bits and cannot be used to set the cycle. These bits always read as 0. RTCOR is initialized to H'00FF by a power-on reset, but is not initialized by a manual reset or by the standby mode.

To prevent RTCOR from being written incorrectly, it must be written by a different method from most other registers. A word transfer operation is used, H'96 is written in the top byte and the actual data is written in the lower byte. For details, see section 8.2.11, Note on Register Access.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

### 8.2.10 Parity Control Register (PCR)

The parity control register (PCR) is a 16-bit read/write register that selects the parity polarity and space to be parity checked. PCR is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PEF | PFRC | PEO | PCHK1 | PCHK0 | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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: | — | — | — | — | — | — | — | — |

- Bit 15 (parity error flag (PEF)): When a parity check is done, PEF indicates whether a parity error has occurred. 0 indicates that no parity error has occurred; 1 indicates that a parity error has occurred.

| Bit 15: PEF | Description |
|---|---|
| 0 | No parity error (initial value).<br>Cleared by reading PEF after it has been set to 1, then writing 0 in PEF. |
| 1 | Parity error has occurred. |

- Bit 14 (parity output force (PFRC)): PFRC selects whether to produce a forced parity output for testing the parity error check function. When cleared to 0, there is no forced output; when set to 1, it produces a forced output of high level from the DPH and DPL pins when data is output, regardless of the parity.

| Bit 14: PFRC | Description |
|---|---|
| 0 | Parity output not forced (initial value) |
| 1 | High output forced |

- Bit 13 (parity polarity (PEO)): PEO selects even or odd parity. When cleared to 0, parity is even; when set to 1, parity is odd.

| Bit 13: PEO | Description |
|---|---|
| 0 | Even parity (initial value) |
| 1 | Odd parity |

- Bits 12 and 11 (parity check enable bits 1, 0 (PCHK1 and PCHK0)): These bits determine whether or not a parity is checked and generated, and select the check and generation spaces.

| Bit 12: PCHK1 | Bit 11: PCHK0 | Description |
|---|---|---|
| 0 | 0 | Parity not checked and not generated (initial value) |
| | 1 | Parity checked and generated only in DRAM area |
| 1 | 0 | Parity checked and generated in DRAM area and area 2 |
| | 1 | Reserved |

- Bits 10–0 (reserved): These bits always read as 0. The write value should always be 0.

### 8.2.11 Notes on Register Access

RCR, RTCSR, RTCNT, and RTCOR differ from other registers in being more difficult to write. Data requires a password when it is written. This prevents data from being mistakenly overwritten by program overruns and the like.

**Writing to RCR, RTCSR, RTCNT, and RTCOR:** Use only word transfer instructions. You cannot write with byte transfer instructions. As figure 8.2 shows, when writing to RCR, place H'5A in the upper byte and the write data in the lower byte. When writing to RTCSR, place H'A5 in the upper byte and the write data in the lower byte. When writing to RTCNT, place H'69 in the upper byte and the write data in the lower byte. When writing to RTCOR, place H'96 in the upper byte and the write data in the lower byte. These transfers write data in the lower byte to the respective registers. If the upper byte differs from the above passwords, no writing occurs.

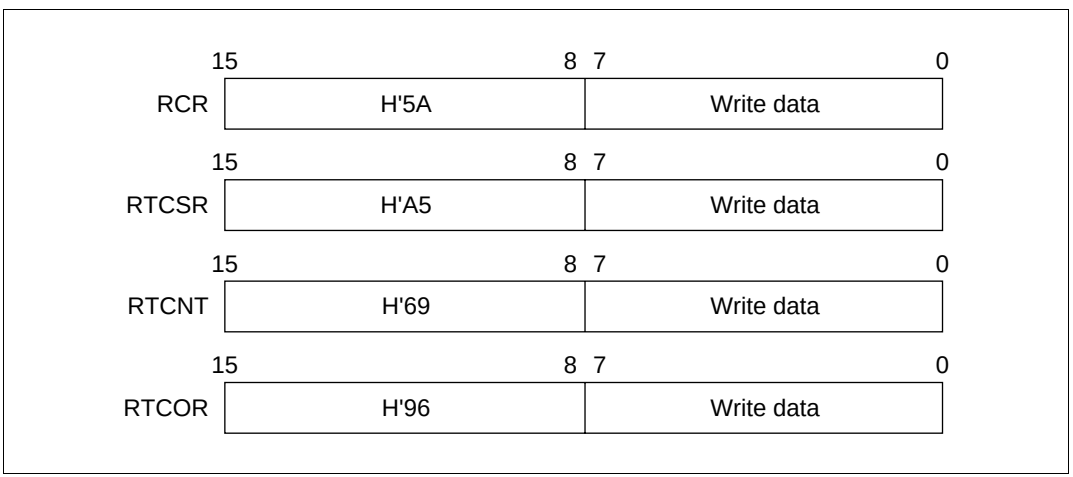



**Figure 8.2 Writing to RCR, RTCSR, RTCNT, and RTCOR**

**Reading from RCR, RTCSR, RTCNT, and RTCORP:** These registers are read like other registers. They can be read by byte and word transfer instructions. If read by word transfer, the value of the upper eight bits is H'00.

## 8.3 Address Space Subdivision

### 8.3.1 Address Spaces and Areas

Figure 8.3 shows the address format used in this LSI.

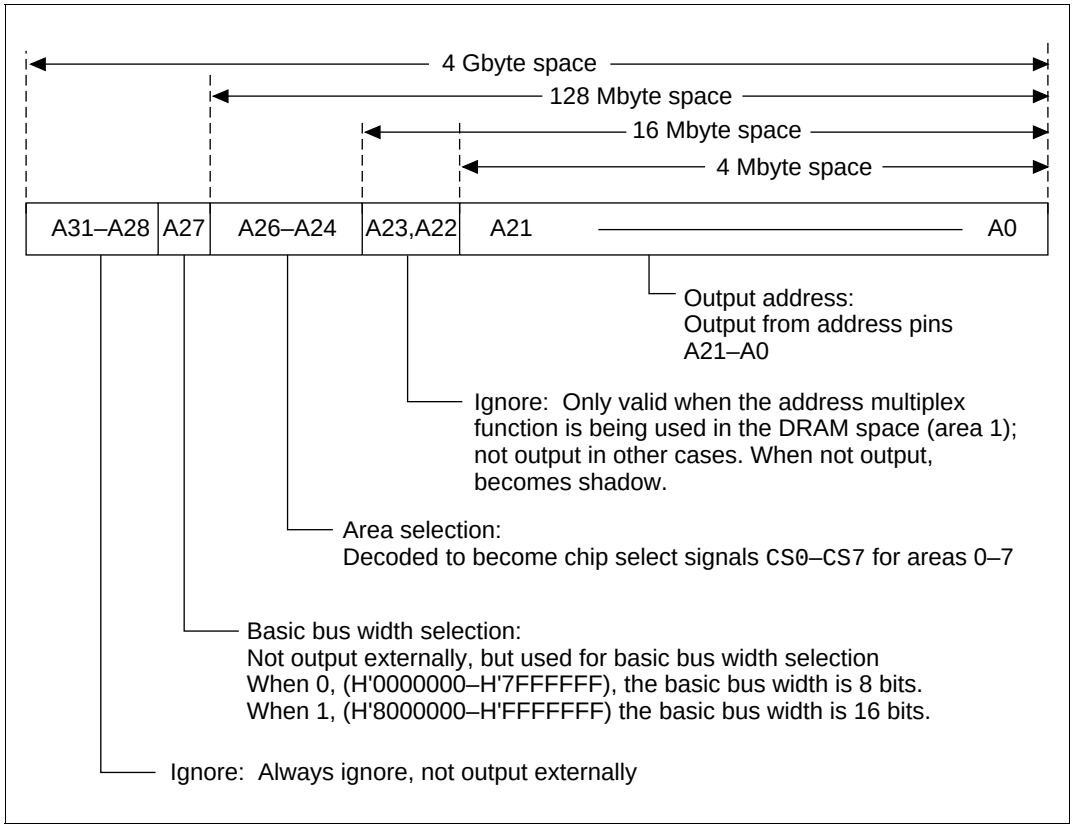



**Figure 8.3 Address Format**

Since this LSI uses a 32-bit address, 4 Gbytes of space can be accessed on the architecture; however, the upper 4 bits (A31–A28) are always ignored and not output. Bit A27 is basically only used for switching the bus width. When the A27 bit is 0 (H'0000000–H'7FFFFFF), the bus width is 8 bits; when the A27 bit is 1 (H'8000000–H'FFFFFFF), the bus width is 16 bits. Of the remaining 27 bits (A26–A0), a total 128 Mbyte can thus be accessed.

The 128 Mbyte space is subdivided into 8 areas (areas 0–7) of 16 Mbytes each according to the values of bits A26–A24. The space with bits A26–A24 as 000 is area 0 and the space 111 is area 7. The A26–A24 bits are decoded and become the chip select signals ($\overline{CS0}$–$\overline{CS7}$) of the corresponding areas 0–7 and output. Table 8.6 shows how the space is divided.

**Table 8.6 How Space is Divided**

| Area | Address | Assign-able Memory | Capacity (linear space) | Bus Width | CS Output |
|---|---|---|---|---|---|
| 0 | H'0000000 – H'0FFFFFF | On-chip ROM*1 | 16 kB*3<br>32 kB*4 | 32 | — |
| | | External memory*2 | 4 MB | 8/16*5 | CS0 |
| 1 | H'1000000 – H'1FFFFFF | External memory | 4 MB | 8 | CS1 |
| | | DRAM*6 | 16 MB | 8 | RAS CAS |
| 2 | H'2000000 – H'2FFFFFF | External memory | 4 MB | 8 | CS2 |
| 3 | H'3000000 – H'3FFFFFF | External memory | 4 MB | 8 | CS3 |
| 4 | H'4000000 – H'4FFFFFF | External memory | 4 MB | 8 | CS4 |
| 5 | H'5000000 – H'5FFFFFF | On-chip peripheral module | 512 B | 8/16*7 | — |
| 6 | H'6000000 – H'6FFFFFF | External memory*9 | 4 MB | 8/16*8 | CS6 |
| | | Multiplexed I/O | 4 MB | | |
| 7 | H'7000000 – H'7FFFFFF | External memory | 4 MB | 8 | CS7 |
| 0 | H'8000000 – H'8FFFFFF | On-chip ROM*1 | 16 kB*3<br>32 kB*4 | 32 | — |
| | | External memory*2 | 4 MB | 8/16*5 | CS0 |
| 1 | H'9000000 – H'9FFFFFF | External memory | 4 MB | 16 | CS1 |
| | | DRAM*6 | 16 MB | 16 | RAS CAS |
| 2 | H'A000000 – H'AFFFFFF | External memory | 4 MB | 16 | CS2 |
| 3 | H'B000000 – H'BFFFFFF | External memory | 4 MB | 16 | CS3 |
| 4 | H'C000000 – H'CFFFFFF | External memory | 4 MB | 16 | CS4 |
| 5 | H'D000000 – H'DFFFFFF | External memory | 4 MB | 16 | CS5 |
| 6 | H'E000000 – H'EFFFFFF | External memory | 4 MB | 16 | CS6 |
| 7 | H'F000000 – H'FFFFFFF | On-chip RAM | 1 kB | 32 | — |

Notes:
1. When MD2–MD0 pins are 010
2. When MD2–MD0 pins are 000 or 001
3. For SH7020
4. For SH7021
5. Select with MD0 pin
6. Select with DRAME bit in BCR
7. Divided into 8-bit and 16-bit space according to value of address bit A8 (Long word accesses are inhibited, however, in on-chip peripheral modules with bus widths of 8 bits. Some on-chip peripheral modules with bus widths of 16 bits also have registers that are only byte-accessible and registers for which byte access is inhibited. For details, see the sections on the individual modules.)
8. Divided into 8-bit space and 16-bit space by value of address bit A14
9. Select with IOE bit of BCR

As figure 8.4 shows, specific spaces such as DRAM space and address/data multiplexed I/O space are allocated to the 8 areas. Each of the spaces is equipped with the necessary interfaces. The control signals needed by DRAM and peripheral LSIs will be output by the chip to devices connected to an area allocated to the appropriate type of space.

### 8.3.2 Bus Width

The primary bus width selection on for this chip is made by switching between 8-bit and 16 bit using the A27 bit. When A27 is 0, the bus width is 8 bits and data is input/output through the AD7–AD0 pins; when A27 is 1, the size is 16 bits and data is input/output through the AD15–AD0 pins for word accesses. For byte access, the top byte is input/output through AD15–AD8 and the lower byte through AD7–AD0. When the bus width is 8 bits or byte access is being performed with a 16-bit bus width, the status of the eight AD pins that are not inputting/outputting data is as shown in appendix B, Pin States.

Bus widths are also determined by conditions other than the A27 bit for specific areas:

- Area 0 is an 8-bit external memory space when the MD2–MD0 pins are 000, a 16-bit external memory space when the same bits are 001, and a 32-bit on-chip ROM space when they are 010.
- Area 5 is an 8-bit on-chip peripheral module space when the A27 bit and A8 bit are both 0 and a 16-bit on-chip peripheral module space when the A27 bit is 0 and the A8 bit is 1. When the A27 bit is 1, it is a 16-bit external memory space.
- Area 6 is an 8-bit bus width when the A27 bit and A14 bit are both 0 and a 16-bit bus width when the A27 bit is 0 and the A14 bit is 1. When the A27 bit is 1, it is a 16-bit space.
- Area 7 is a 32-bit on-chip RAM space when the A27 bit is 1 and an 8-bit external memory space when the A27 bit is 0.

Word (16-bit) data accessed from 8-bit bus areas and longword (32-bit) data accessed from 16-bit bus areas require two consecutive accesses. Longword (32-bit) data accessed from 8-bit bus areas requires four consecutive accesses.

### 8.3.3 Chip Select Signals ($\overline{\text{CS0}}$–$\overline{\text{CS7}}$)

When the A26–A24 bits of the address are decoded, they become chip select signals ($\overline{\text{CS0}}$–$\overline{\text{CS7}}$) for areas 0–7. When an area is accessed, the corresponding chip select pins are driven low. Table 8.7 shows the relationship between the A26–A24 bits and the chip select signals.

**Table 8.7 A26–A24 Bits and Chip Select Signals**

| Address | | | | |
|---|---|---|---|---|
| **A26** | **A25** | **A24** | **Area Selected** | **Chip Select Pin Driven Low** |
| 0 | 0 | 0 | Area 0 | CS0 |
| | | 1 | Area 1 | CS1 |
| | 1 | 0 | Area 2 | CS2 |
| | | 1 | Area 3 | CS3 |
| 1 | 0 | 0 | Area 4 | CS4 |
| | | 1 | Area 5 | CS5 |
| | 1 | 0 | Area 6 | CS6 |
| | | 1 | Area 7 | CS7 |

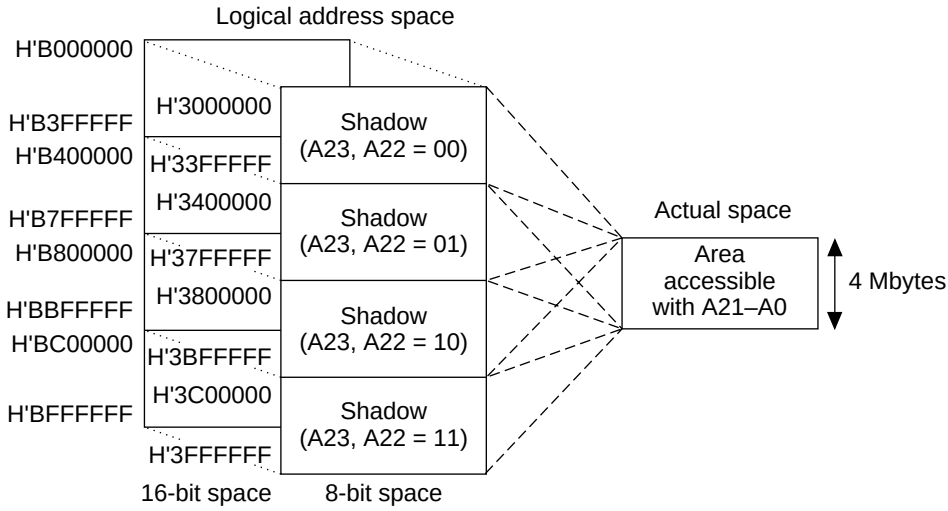
The chip select signal is output only for external accesses. When accessing the on-chip ROM (area 0), on-chip peripheral modules (area 5) and on-chip RAM (area 7), the $\overline{CS0}$, $\overline{CS5}$, and $\overline{CS7}$ pins are not driven low. When accessing DRAM space (area 1), select the $\overline{RAS}$ and $\overline{CAS}$ signals with the pin function controller.

### 8.3.4 Shadows

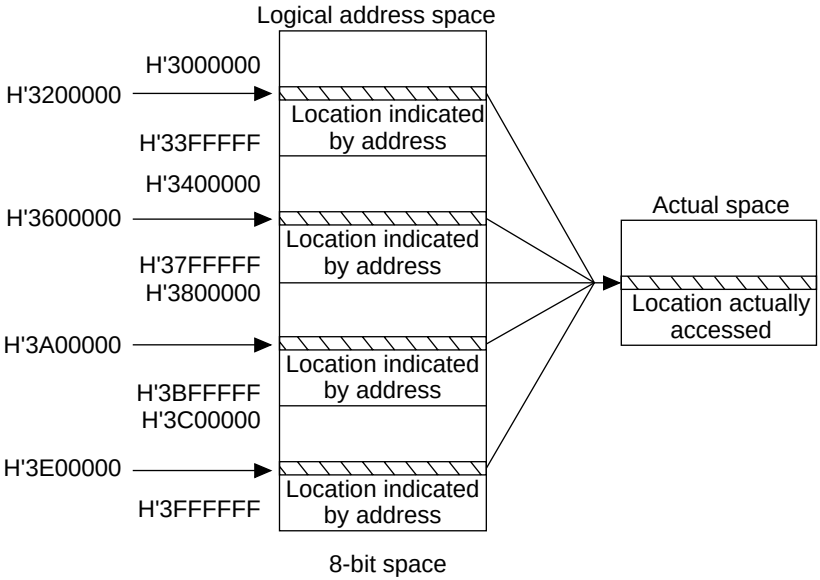
The size of each area is 16 Mbytes, which can be specified with 24 address bits A23–A0 for 8-bit spaces and 16-bit spaces alike. Bits A23 and A22, however, output externally only when the address multiplex function is used in DRAM space (area 1); in all other cases, there is no output, so the actually accessible area for all areas is the 4 Mbyte that can be specified with 22 bits A21–A0. No matter what the values of A23 and A22, the same 4 Mbytes of actual space is accessed. As illustrated in figure 8.4 (a), the A23 and A22 bit regions 00, 01, 10 and 11 are called shadows of actual areas. Shadows are allocated in 4-Mbyte units for both 8-bit and 16-bit bus widths. When the same addresses H'3200000, H'3600000, H'3A00000 and H'3E00000 are specified for values A21–A0, as shown in figure 8.4 (b), the same actual space is accessed regardless of the A23 and A22 bits.

In areas whose bus widths are switchable using the A27 address bit, the shadow of the same actual space is allocated to both A27 = 0 spaces and A27 = 1 spaces (figure 8.4(a)). When the value of A27 is changed, the valid AD pins switch from AD15–AD0 to AD7–AD0, but the actual space accessed remains the same.

The spaces of on-chip ROM (area 0), DRAM (area 1), on-chip peripheral modules (area 5) and on-chip RAM (area 7) have shadows of different sizes from those discussed above. See section 8.3.5, Description of Areas, for details.




a. Shadow allocation




b. Actual space accessed when addresses are specified

**Figure 8.4 Shadows**

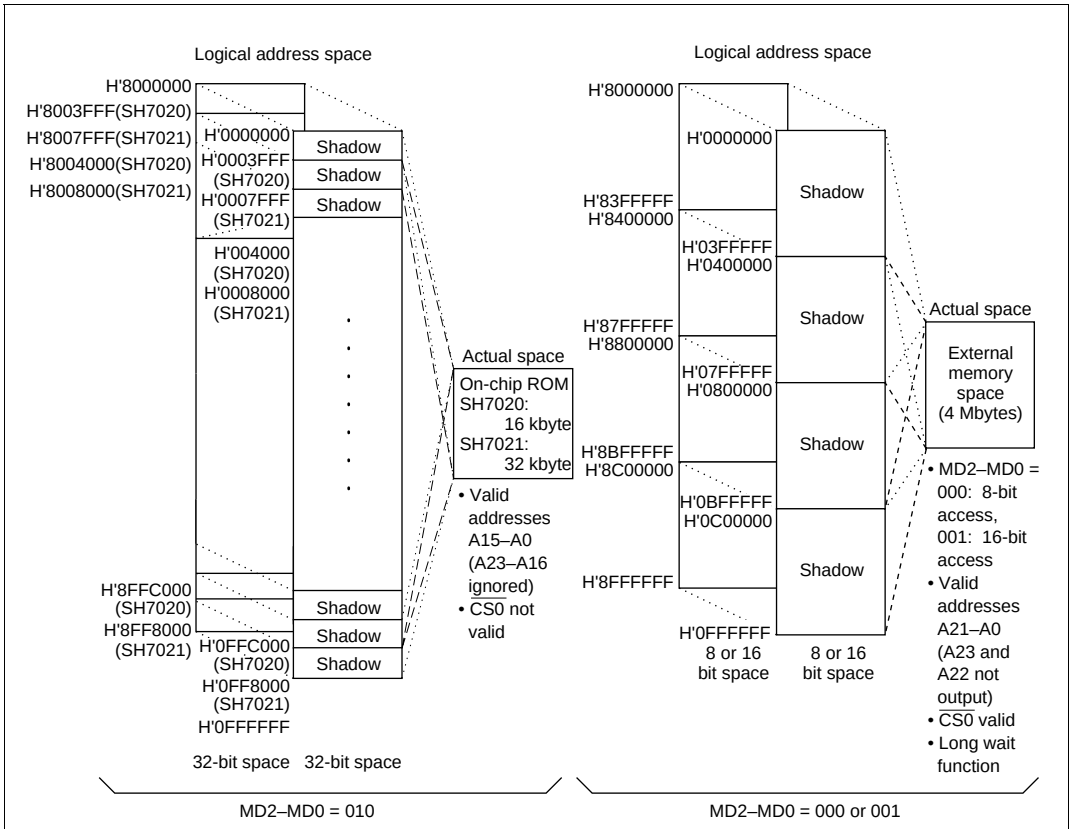
### 8.3.5 Area Description

**Area 0:** Area 0 is the area where addresses A26–A24 are 000 and its address range is H'0000000–H'0FFFFFF and H'8000000–H'8FFFFFF. Figure 8.5 is a memory map of area 0.

Area 0 can be set for use as on-chip ROM space or external memory space with the mode pins (MD2–MD0). The MD2–MD0 pins also determine the bus width, regardless of the A27 address bit. When MD2–MD0 are 000, area 0 is an 8-bit external memory space; when they are 001, area 0 is a 16-bit external memory space; and when they are 010, it is a 32-bit on-chip ROM space.

In the SH7020, the capacity of the on-chip ROM is 16 kbyte, so bits A23–A14 are ignored in on-chip ROM space and the shadow is in 16 kbyte units. In the SH7021, the capacity of the on-chip ROM is 32 kbyte, so bits A23–A15 are ignored in on-chip ROM space and the shadow is in 32 kbyte units. The $\overline{\text{CSO}}$ signal is disabled in on-chip ROM space.

In external memory space, the A23 and A22 bits are not output and the shadow is in 4-Mbyte units. When external memory space is accessed, the $\overline{\text{CS0}}$ signal is valid. The external memory space has a long wait function, so between 1 and 4 states can be selected for the number of long waits inserted into the bus cycle using the areas 0 and 2 long wait insertion bits (A02LW1, A02LW0) of wait state controller 3 (WCR3).




Note: The bus width of area 0 is determined by the MD2–MD0 pins regardless of the A27 bit setting.

**Figure 85 Memory Map of Area 0**

**Area 1:** Area 1 is the area where addresses A26–A24 are 001 and its address range is H'1000000–H'1FFFFFF and H'9000000–H'9FFFFFF. Figure 8.6 is a memory map of area 1.

Area 1 can be set for use as DRAM space or external memory space with the DRAM enable bit (DRAME) of the bus control register (BCR). When the DRAME bit is 0, it is external memory space; when DRAME is 1, it is DRAM space.

In external memory space, the bus width is 8 bits when the A27 bit is 0 and 16 bits when it is 1. Bits A23 and A22 are not output and the shadow is in 4-Mbyte units. When external memory is accessed, the $\overline{CS1}$ signal is valid.

DRAM space is a type of external memory space, but it is configured especially to be connected to DRAM so it outputs strobe signals required for this purpose. Its bus width is 8 bits when it is 0 and 16 bits when it is 1. When the multiplex enable bit (MXE) of the DRAM control register (DCR) is

set to 1 to use the address multiplex function, bits A23–A0 are multiplexed and output from pins A15–A0, so a maximum 16-Mbyte space can be used. When DRAM space is accessed, the $\overline{CS1}$ signal is not valid and the pin function controller should be set for access with $\overline{CAS}$ ($\overline{CASH}$ and $\overline{CASL}$) and $\overline{RAS}$ signals.

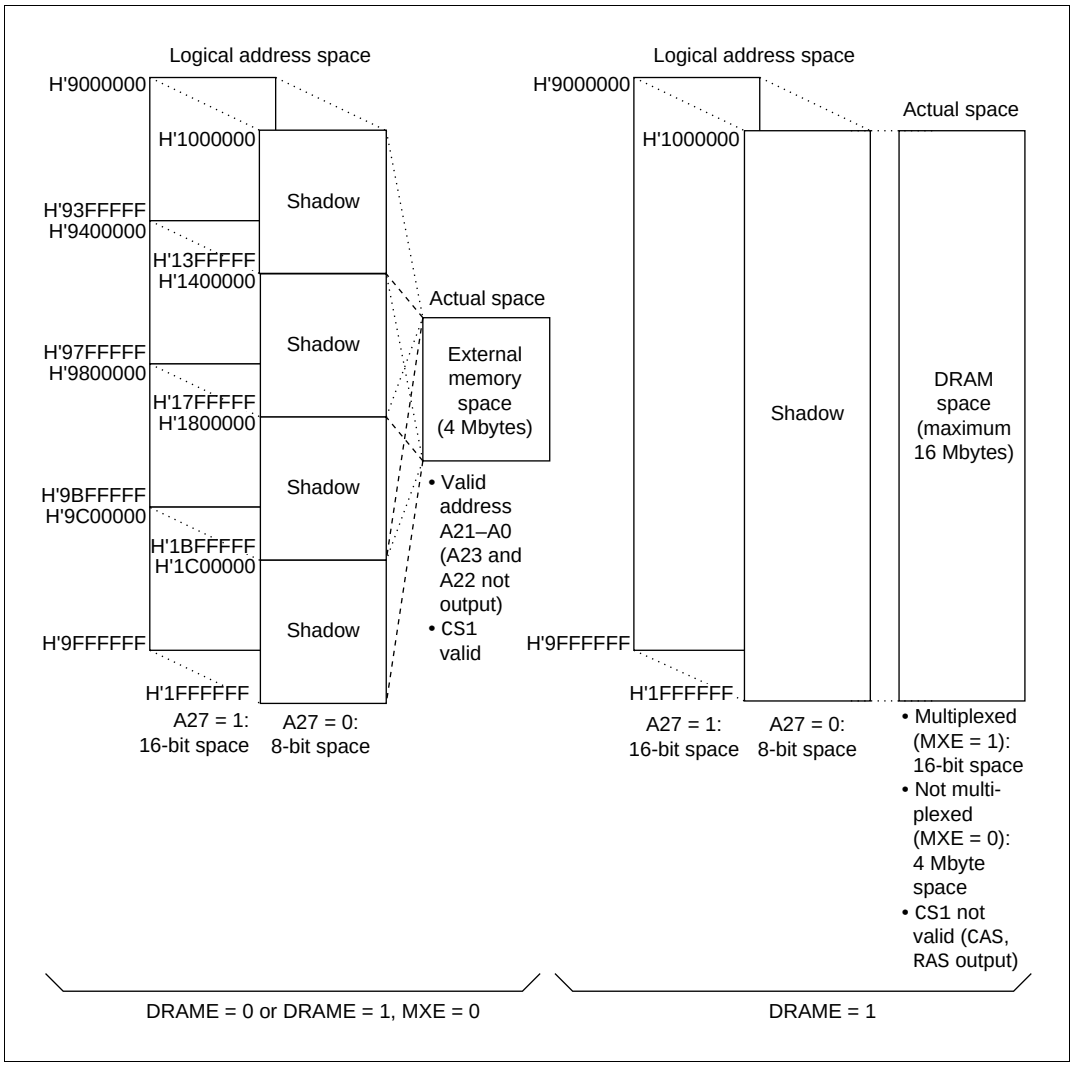



**Figure 8.6 Memory Map of Area 1**

**Areas 2–4:** Areas 2–4 are the areas where addresses A26–A24 are 010, 011 and 100, respectively, and their address ranges are H'2000000–H'2FFFFFF and H'A000000–H'AFFFFFF (area 2), H'3000000–H'3FFFFFF and H'B000000–H'BFFFFFF (area 3), and H'4000000–H'4FFFFFF and H'C000000–H'CFFFFFF (area 4). Figure 8.7 is a memory map of area 2, which is representative of areas 2–4.

Areas 2–4 are always used as external memory space. The bus width is 8 bits when the A27 bit is 0 and 16 bits when it is 1. A23 and A22 bits are not output and the shadow is in 4-Mbyte units. When areas 2–4 are accessed, the $\overline{CS2}$, $\overline{CS3}$, and $\overline{CS4}$ signals are valid. Area 2 has a long wait function, so between 1 and 4 states can be selected for the number of long waits inserted into the bus cycle using the bits A02LW1 and A02LW0 of WCR3.

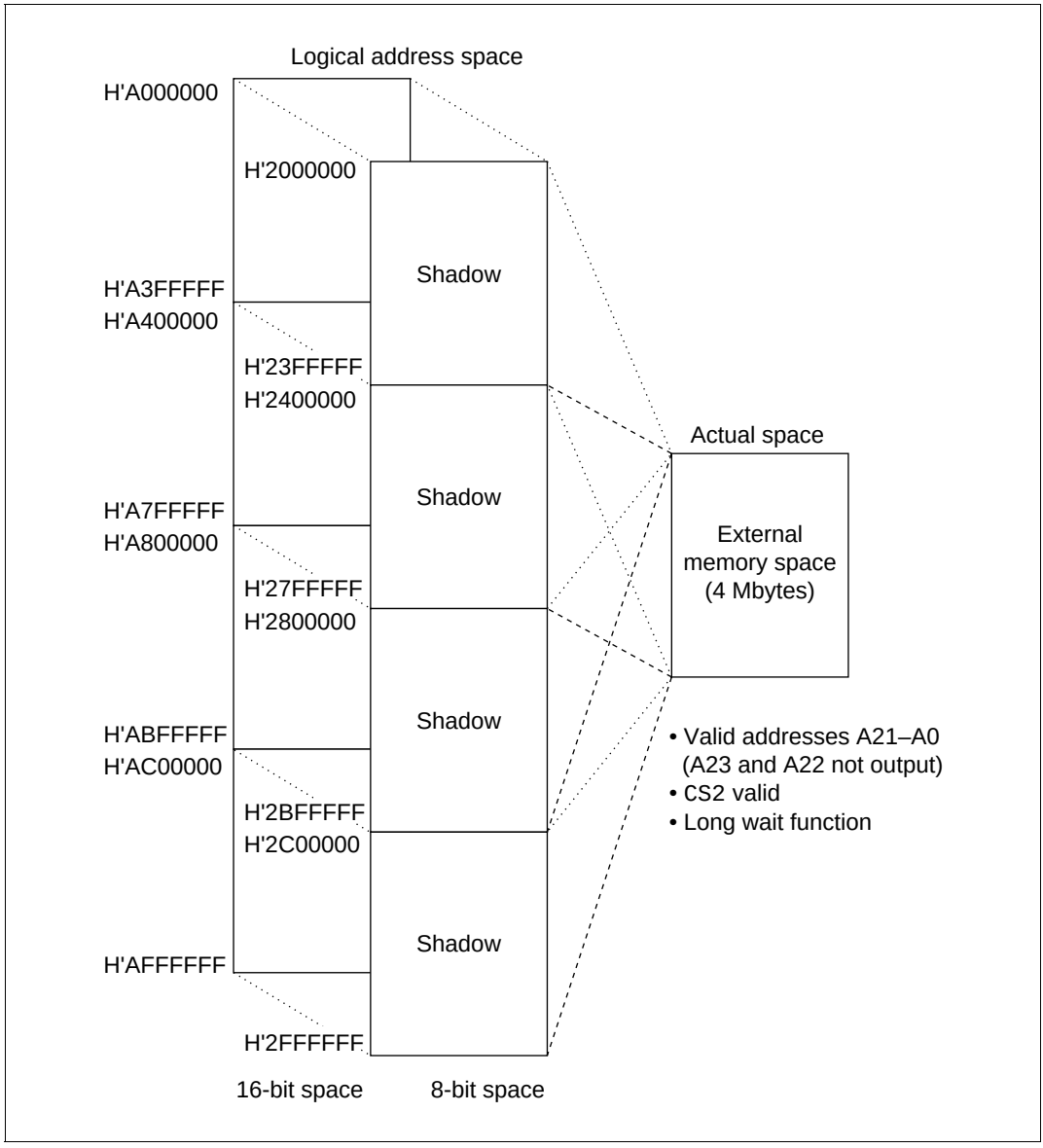



**Figure 8.7 Memory Map of Area 2**

**Area 5:** Area 5 is the area where addresses A26–A24 are 101 and its address range is H'5000000–H'5FFFFFF and H'D000000–H'DFFFFFF. Figure 8.8 is a memory map of area 5.

Area 5 is allocated to on-chip peripheral module space when the A27 address bit is 0 and external memory space when A27 is 1. In on-chip peripheral module space, bits A23–A9 are ignored and the shadows are in 512-byte units. The bus width is 8 bits when the A8 bit is 0 and 16 bits when A8 is 1. When on-chip peripheral module space is accessed, the $\overline{CS5}$ signal is not valid. In external memory space, the A23 and A22 bits are not output and the shadow is in 4-Mbyte units. The bus width is always 16 bits. When external memory space is accessed, the $\overline{CS5}$ signal is valid.

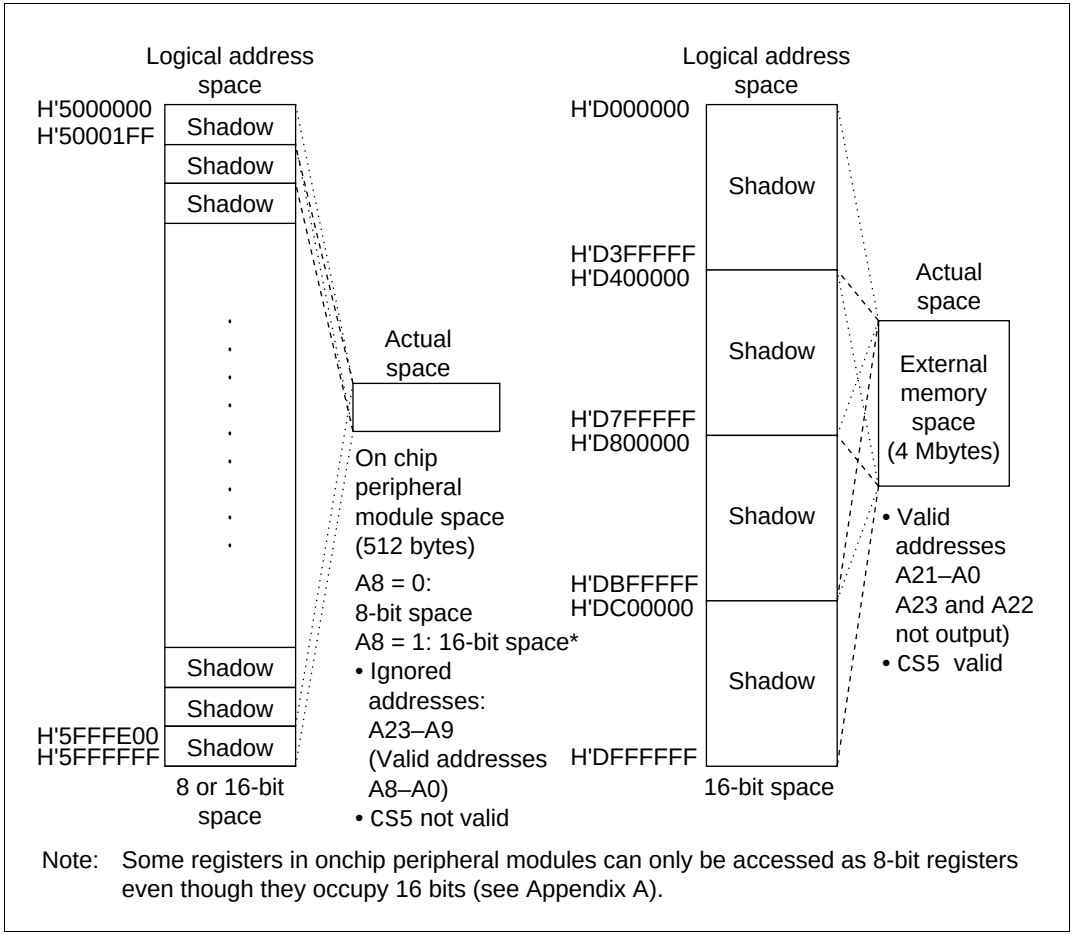



**Figure 8.8 Memory Map of Area 5**

**Area 6:** Area 6 is the area where addresses A26–A24 are 110 and its address range is H'6000000–H'6FFFFFF and H'E000000–H'EFFFFFF. Figure 8.9 is a memory map of area 6.

In area 6, the space when A27 is 0 is allocated to address/data multiplexed I/O space when the multiplexed I/O enable bit (IOE) of the bus control register (BCR) is 1 and external memory space when the IOE bit is 0. When A27 is 1, it is always external memory space.

The multiplexed I/O space is a type of external memory space but the address and data are multiplexed and output from AD15–AD0 or AD7–AD0. The bus width is 8 bits when the A14 bit is 0 and 16 bits when the A14 bit is 1. The A23 and A22 bits are not output and the shadow is in 4-Mbyte units. When multiplexed I/O space is accessed, the $\overline{CS6}$ signal is valid.

In external memory space, the bus width is 8 bits when both the A27 and A14 bits are 0 and 16 bits when the A27 bit is 0 and the A14 bit is 1. When the A27 bit is 1, it is always a 16-bit space. The A23 and A22 bits are not output and the shadow is in 4-Mbyte units. When external memory is accessed, the $\overline{CS6}$ signal is valid. The external memory space has a long wait function so between 1 and 4 states can be selected for the number of long waits inserted into the bus cycle using the area 6 long wait insertion bits (A6LW1 and A6LW0) of WCR3.

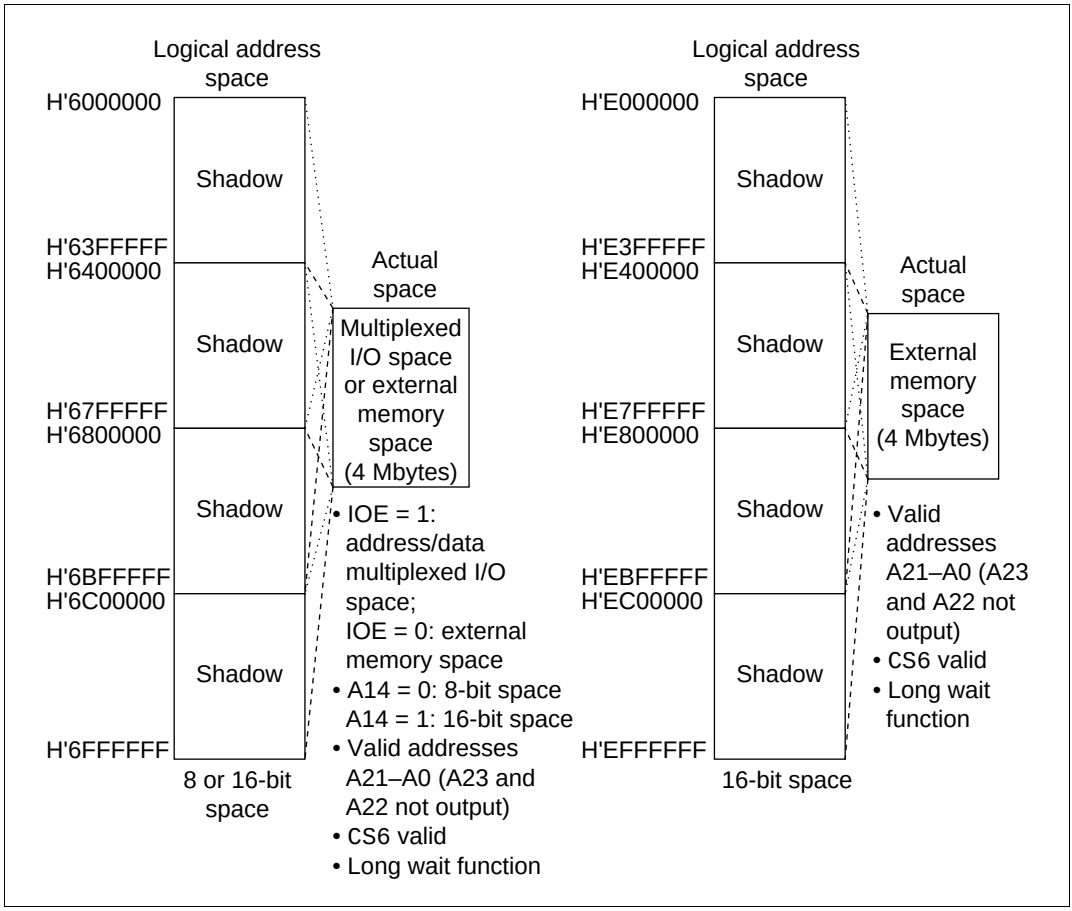



**Figure 8.9 Memory Map of Area 6**

**Area 7:** Area 7 is the area where addresses A26–A24 are 111 and its address range is H'7000000–H'7FFFFFF and H'F000000–H'FFFFFFF. Figure 8.10 is a memory map of area 7.

Area 7 is allocated to external memory space when A27 is 0 and on-chip RAM space when A27 is 1. In external memory space, the bus width is 8 bits. The A23 and A22 bits are not output and the shadow is in 4-Mbyte units. When external memory is accessed, the $\overline{CS7}$ signal is valid.

The on-chip RAM space has an bus width of 32 bits. The on-chip RAM capacity is 1 kbytes, so A23–A10 are ignored and the shadows are in 8-kbyte units. During on-chip RAM access, the $\overline{CS7}$ signal is not valid.

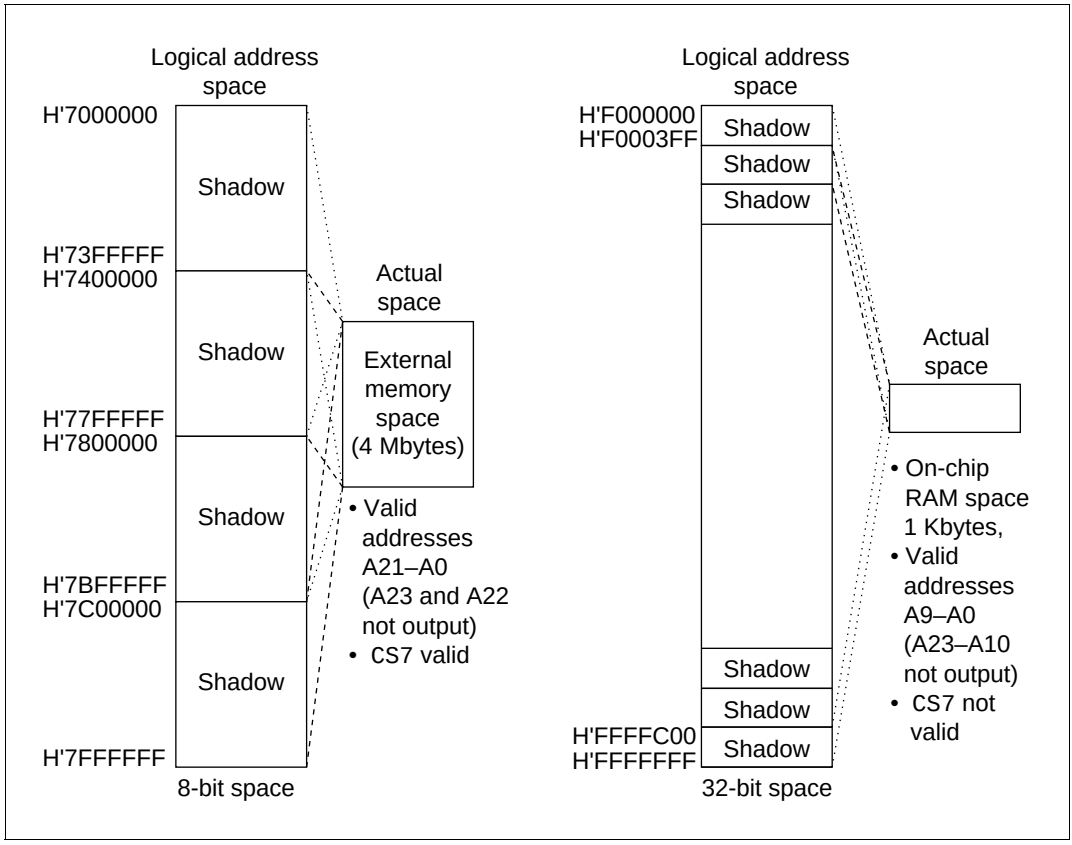



**Figure 8 10 Memory Map of Area 7**

## 8.4 Accessing External Memory Space

In external memory space, strobe signal is output based on the assumption of a directly connected SRAM. The external memory space is allocated to the following areas:

- Area 0 (when MD2–MD0 are 000 or 001)
- Area 1 (when the DRAM enable bit (DRAME) of the BCR is 0)
- Areas 2–4
- Area 5 (space where address A27 is 1)
- Area 6 (when the multiplexed I/O enable bit (IOE) bit of the BCR is 0, or space where address A27 is 1)
- Area 7 (space where address A27 is 0)

### 8.4.1 Basic Timing

The bus cycle for external memory space access is 1 or 2 states. The number of states is controlled with the wait states by the settings of wait state control registers 1–3 (WCR1–WCR3). For details, see section 8.4.2., Wait State Control. Figures 8.11 and 8.12 illustrate the basic timing of external memory space access.

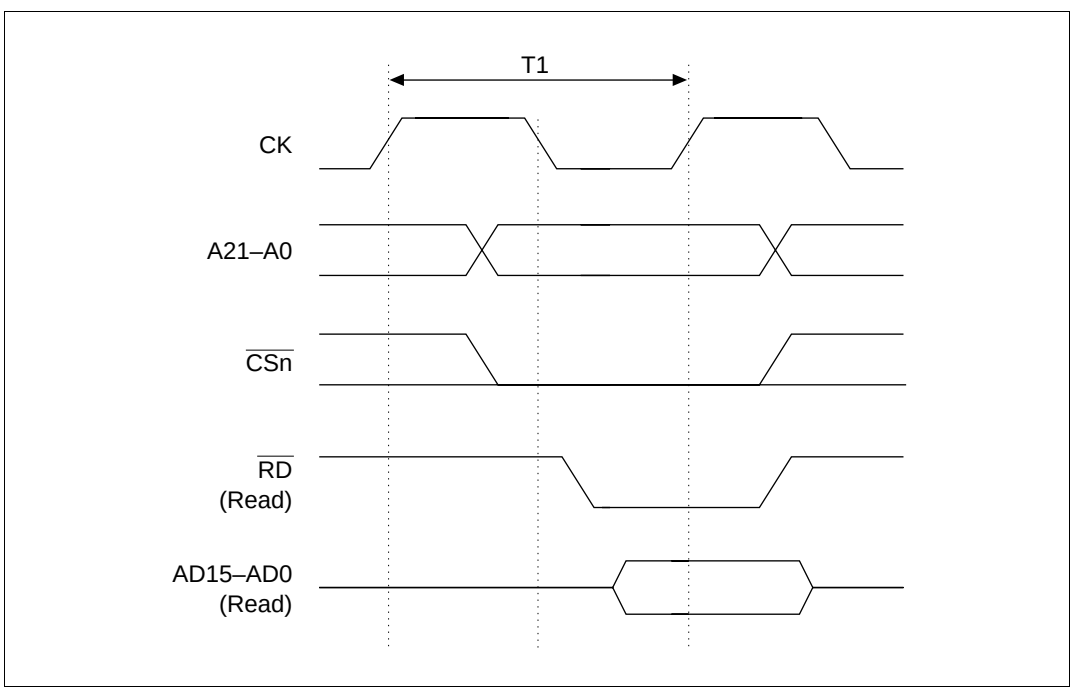



**Figure 8.11 Basic Timing of External Memory Space Access (1-state read timing)**

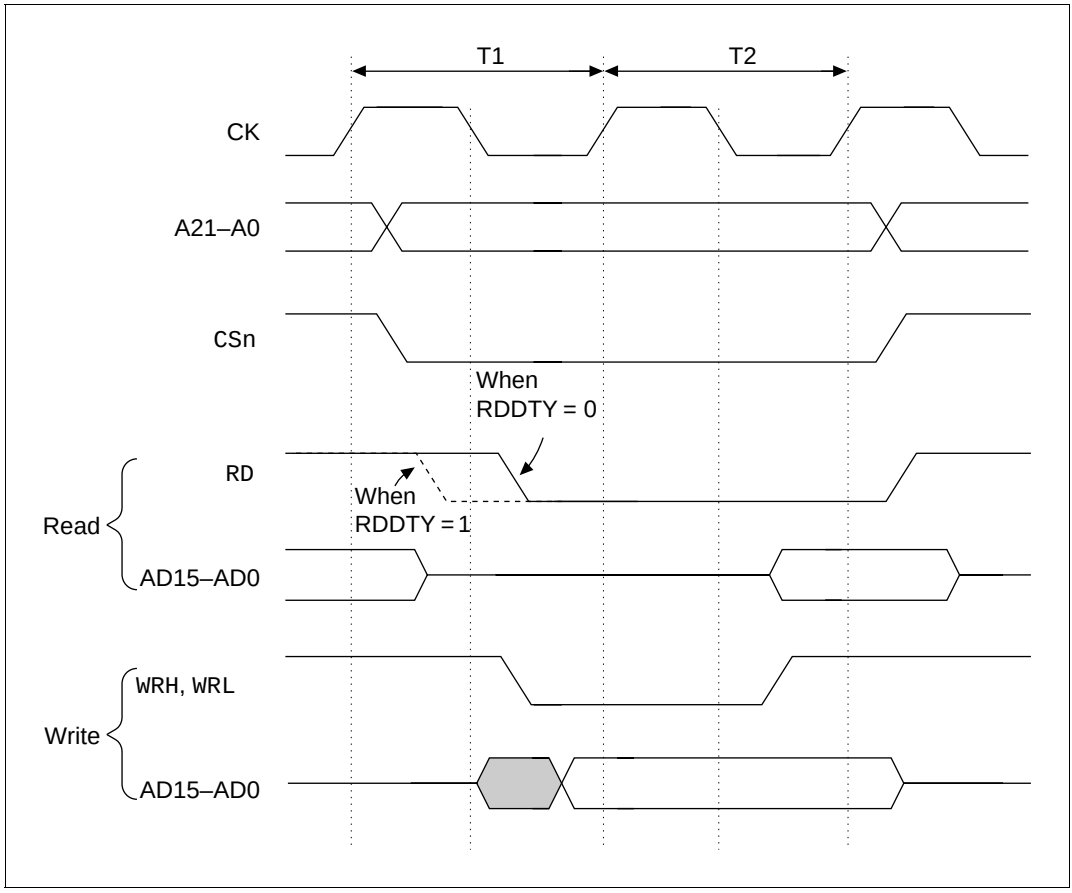



**Figure 8.12 Basic Timing of External Memory Space Access (2-state read timing)**

High-level duties of 35% and 50% can be selected for the $\overline{RD}$ signal using the RD duty bit (RDDTY) of the BCR. When RDDTY is set to 1, the high-level duty is 35% of the T1 state, enabling longer access times for external devices. Only set to 1 when the operating frequency is a minimum of 10 MHz.

### 8.4.2 Wait State Control

The number of external memory space access states and the insertion of wait states can be controlled using the WCR1–WCR3 bits. The bus cycles that can be controlled are the CPU read cycle and the DMAC dual mode read cycle. The bus cycle that can be controlled using the WCR2 is the DMAC single-mode read/write cycle.

Table 8.8 shows the number of states and number of wait states in the access cycles to external memory spaces.

**Table 8.8 Number of States and Number of Wait States in the Access Cycles to External Memory Spaces**

| Area | CPU read cycle, DMAC dual mode read cycle, DMAC single mode read/write cycle | | CPU Write Cycle and DMAC Dual Mode Write Cycle (Cannot be controlled by WCR1)*2 |
|---|---|---|---|
| | Corresponding Bits in WCR1 and WCR2 = 0 | Corresponding Bits in WCR1 and WCR2 = 1 | |
| 1, 3–5, 7 | 1 cycle fixed; WAIT signal ignored | 2 cycles fixed + wait state from WAIT signal | |
| 0, 2, 6 (long wait available) | 1 cycle + long wait state, WAIT signal ignored | 1 cycle + long wait state*1 + wait state from WAIT signal | |

Notes: 1. The number of long wait states is set by WCR3.
2. When DRAME = 1, short pitch/long pitch is selected with the WW1 bit of the WCR1.
3. Pin wait cannot be used for the CS7 and WAIT pins of area 3 because they are multiplexed.

For the CPU read cycle, DMAC dual mode read cycle and DMAC single mode read/write cycle, the access cycle is completed in 1 state when the corresponding bits of WCR1 and WCR2 for areas 1, 3–5, and 7 are cleared to 0 and the $\overline{\text{WAIT}}$ pin input signal is not sampled. When the bits are set to 1, the $\overline{\text{WAIT}}$ signal is sampled and the number of states is 2 plus the number of wait states in the $\overline{\text{WAIT}}$ signal. The $\overline{\text{WAIT}}$ signal is sampled at the rise of the system clock (CK) directly preceding the second state of the bus cycle and the wait states are inserted as long as the level is low. When a high level is detected, it shifts to the second state (final state). Figure 8.13 shows the wait state timing when accessing the external memory spaces of areas 1, 3, 4, 5, and 7.

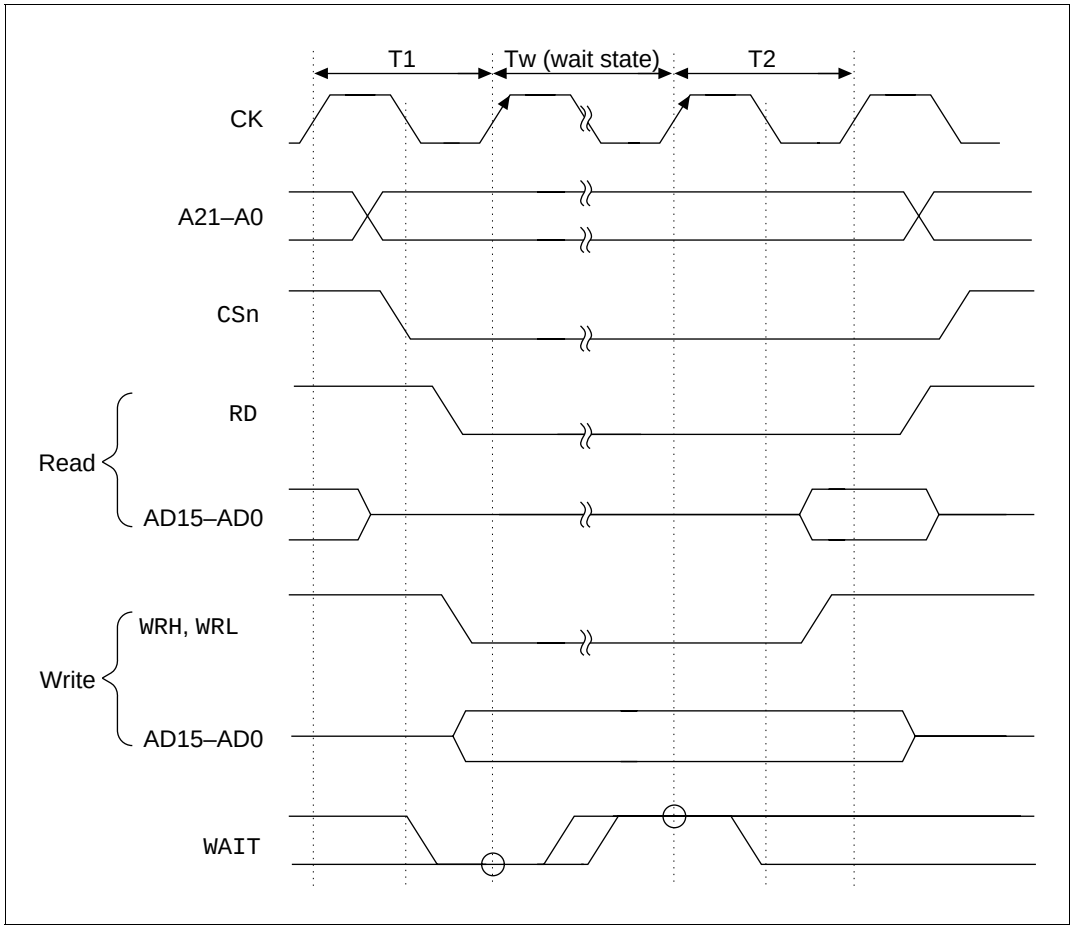



**Figure 8.13 Wait State Timing for External Memory Space Access (2 states plus wait states from $\overline{\text{WAIT}}$ signal)**

Areas 0, 2 and 6 have long wait functions. When the corresponding bits in WCR1 and WCR2 are cleared to 0, the access cycle is 1 state plus the number of long wait states (set in WCR3, selectable between 1 and 4) and the $\overline{\text{WAIT}}$ pin input signal is not sampled. When the bits are set to 1, the $\overline{\text{WAIT}}$ signal is sampled and the number of states is 1 plus the number of long wait states plus the number of wait states in the $\overline{\text{WAIT}}$ signal. The $\overline{\text{WAIT}}$ signal is sampled at the rise of the system clock (CK) directly preceding the last long wait state and the wait states are inserted as long as the level is low. When a high level is detected, it shifts to the final long wait state. Figure 8.14 shows the wait state timing when accessing the external memory spaces of areas 0, 2, and 6.

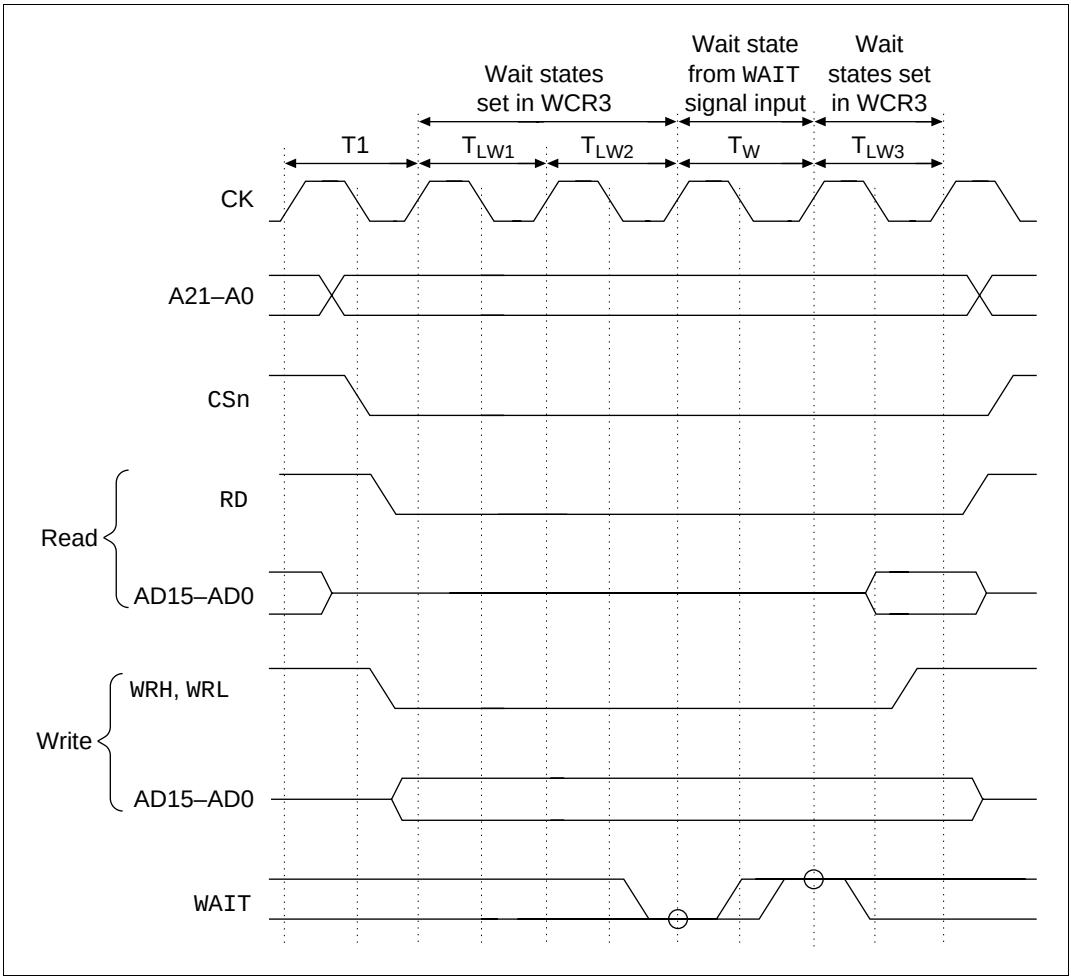



**Figure 8.14 Wait State Timing for External Memory Space Access (1 state plus long wait state (when set to insert 3 states) plus wait states from $\overline{\text{WAIT}}$ signal)**

For CPU write cycles and DMAC dual mode write cycles to external memory space, the number of states and wait state insertion cannot be controlled by WCR1. In areas 1, 3, 4, 5, and 7, the $\overline{\text{WAIT}}$ signal is sampled and the number of states is 2 plus the number of wait states in the $\overline{\text{WAIT}}$ signal (figure 8.13). In areas 0, 2 and 6, the number of states is 1 state plus the number of long wait states plus the number of wait states in the $\overline{\text{WAIT}}$ signal (figure 8.14). Never write 0 in bits 7–2 and 0 of WCR1; only write 1. When area 1 is being used as external memory space, never write 0 to bit 1 (WW1); always write 1.

### 8.4.3 Byte Access Control

The upper byte and lower byte control signals when 16-bit bus width space is being accessed can be selected from ($\overline{WRH}$, $\overline{WRL}$, A0) or ($\overline{WR}$, $\overline{HBS}$, $\overline{LBS}$). When the byte access select bit (BAS) of the BCR is set to 1, the $\overline{WRH}$, $\overline{WRL}$, and A0 pins output $\overline{WR}$, $\overline{LBS}$ and $\overline{HBS}$ signals. Figure 8.15 illustrates the control signal output timing in the byte write cycle.

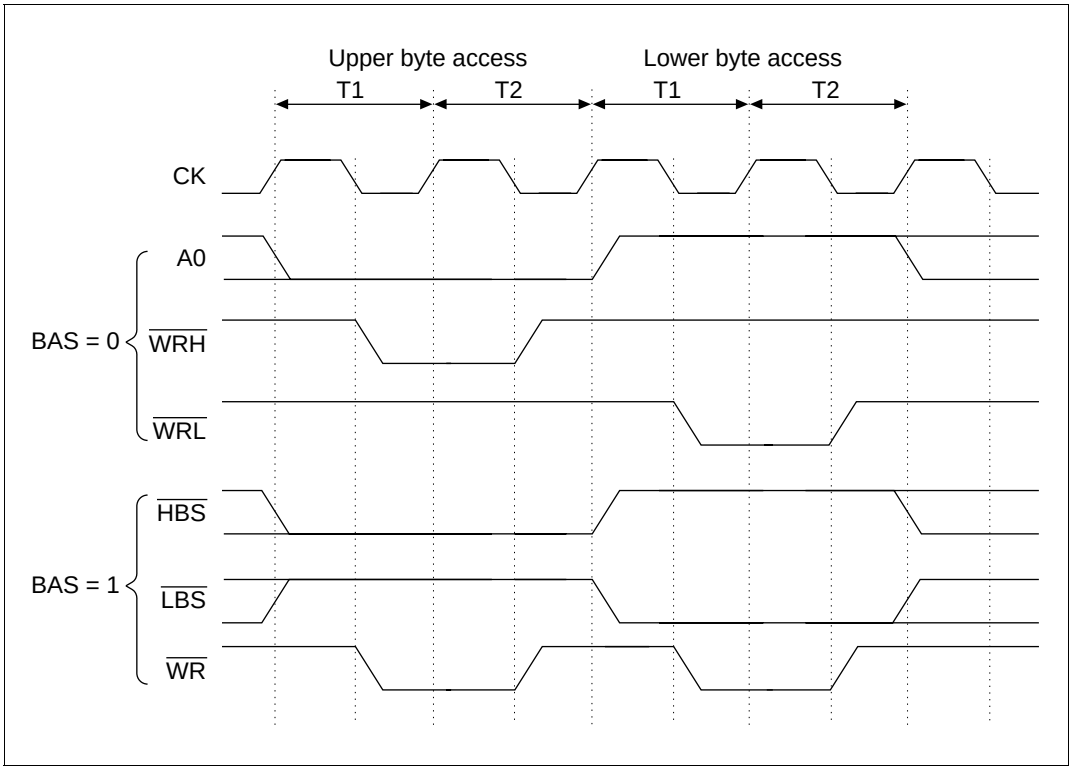



**Figure 8.15 Byte Access Control Timing For External Memory Space Access (Write Cycle)**

The $\overline{WRH}$, $\overline{WRL}$ system and the $\overline{HBS}$, $\overline{LBS}$ system are available as byte access signals for the 16-bit space in the address/data multiplexing space and the external memory space.

These strobe signals are assigned to pins in the manner: A0/$\overline{HBS}$, $\overline{WRH}$/$\overline{LBS}$, $\overline{WRL}$/$\overline{WR}$, and the BAS bit of the bus control register (BCR) is used to switch specify signal sending.

Note that the byte access signals are strobe signals dedicated to byte access to a 16-bit space and not to be used for byte access to an 8-bit space. When making an access to an 8-bit space, use the A0/$\overline{HBS}$ pin as A0 irrespective of the BAS bit value (0 or 1) to use the $\overline{WRL}$/$\overline{WR}$ pin as the $\overline{WR}$ pin, and avoid using the $\overline{WRH}$/$\overline{LBS}$ pin.

## 8.5 DRAM Interface Operation

When the DRAM enable bit (DRAME) of the BCR is set to 1, area 1 becomes DRAM space and the DRAM interface function is available, which permits direct connection of this LSI to DRAMs.

### 8.5.1 DRAM Address Multiplexing

When the multiplex enable bit (MXE) of the DRAM area control register (DCR) is set to 1, row addresses and column addresses are multiplexed. This allows DRAMs that require multiplexing of row and column addresses to be connected directly to the SH microprocessors without additional multiplexing circuits. When addresses are multiplexed (MXE = 1), setting of the DCR's multiplex shift bits (MXC1, MXC0) allows selection of eight, nine and ten-bit row address shifting. Table 8.9 illustrates the relationship between MXC1/MXC0 bits and address multiplexing.

**Table 8.9 Relationship between Multiplex Shift Count Bits (MXC1, MXC0) and Address Multiplexing**

| | Shift Amount 8 bits | | Shift Amount 9 bits | | Shift Amount 10 bits | |
|---|---|---|---|---|---|---|
| Output Pin | Output Row Address | Output Column Address | Output Row Address | Output Column Address | Output Row Address | Output Column Address |
| A21 | Undefined Value | A21 | Undefined Value | A21 | Undefined Value | A21 |
| A20 | | A20 | | A20 | | A20 |
| A19 | | A19 | | A19 | | A19 |
| A18 | | A18 | | A18 | | A18 |
| A17 | | A17 | | A17 | | A17 |
| A16 | | A16 | | A16 | | A16 |
| A15 | A23 | A15 | | A15 | | A15 |
| A14 | A22 | A14 | A23 | A14 | | A14 |
| A13 | A21 | A13 | A22 | A13 | A23 | A13 |
| A12 | A20 | A12 | A21 | A12 | A22 | A12 |
| A11 | A19 | A11 | A20 | A11 | A21 | A11 |
| A10 | A18 | A10 | A19 | A10 | A20 | A10 |
| A9 | A17 | A9 | A18 | A9 | A19 | A9 |
| A8 | A16 | A8 | A17 | A8 | A18 | A8 |
| A7 | A15 | A7 | A16 | A7 | A17 | A7 |
| A6 | A14 | A6 | A15 | A6 | A16 | A6 |
| A5 | A13 | A5 | A14 | A5 | A15 | A5 |
| A4 | A12 | A4 | A13 | A4 | A14 | A4 |
| A3 | A11 | A3 | A12 | A3 | A13 | A3 |
| A2 | A10 | A2 | A11 | A2 | A12 | A2 |
| A1 | A9 | A1 | A10 | A1 | A11 | A1 |
| A0 | A8 | A0 | A9 | A0 | A10 | A0 |

Notes: The MXC1=1, MX0=1 setting is reserved. Do not use it.

For example, when MXC1 and MXC0 are set to 00 and an 8-bit shift is selected, the A23–A8 address bit values are output to pins A15–A0 as row addresses. The values for A21–A16 are undefined. The values of bits address A21–A0 are output to pins A21–A0 as column addresses. Figure 8.16 depicts address multiplexing with an 8-bit shift.

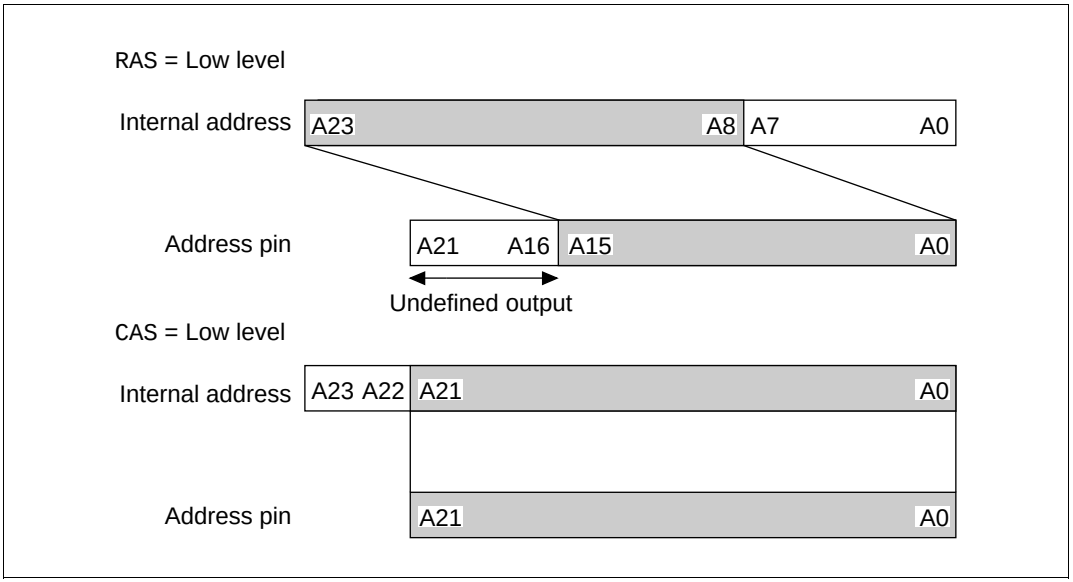



**Figure 8.16 Address Multiplexing States (8-bit shift)**

### 8.5.2 Basic Timing

There are two types of DRAM accesses: short pitch and long pitch. Short pitch or long pitch can be selected for the respective bus cycles using the RW1 and WW1 bits of WCR1 and the DRW1 and DWW1 bits of WCR2. When the corresponding bits are cleared to 0, DRAM access is short pitch and column address output occurs in 1 state. When these bits are 1, DRAM access is long pitch and column address output occurs in 2 states. Figure 8.17 shows short pitch timing; figure 8.18 shows long pitch timing.

The high-level duty of the $\overline{CAS}$ signal can also be selected between 50% and 35% of the $T_C$ state when access is short pitch. By setting the CDTY bit to 1, high level duty becomes 35% and DRAM access time can be lengthened. Only set to 1 when the operating frequency is a minimum of 10 MHz.

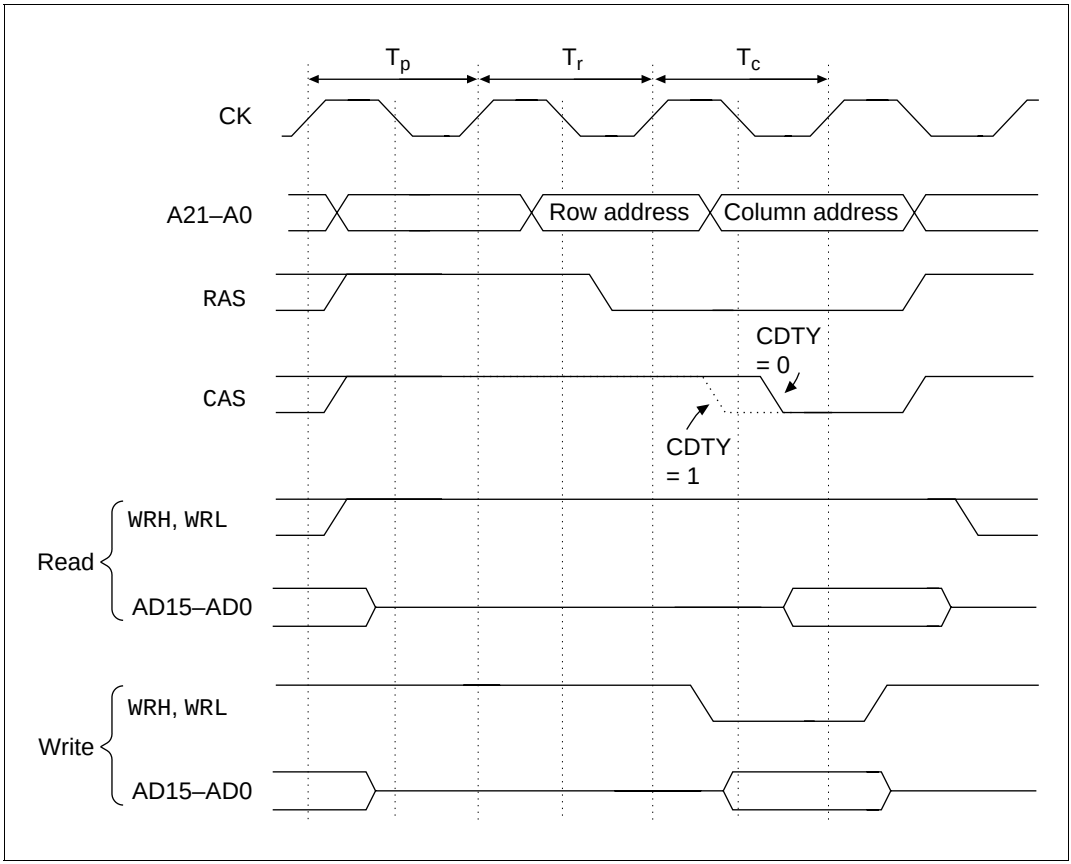



**Figure 8.17 Short Pitch Access Timing**

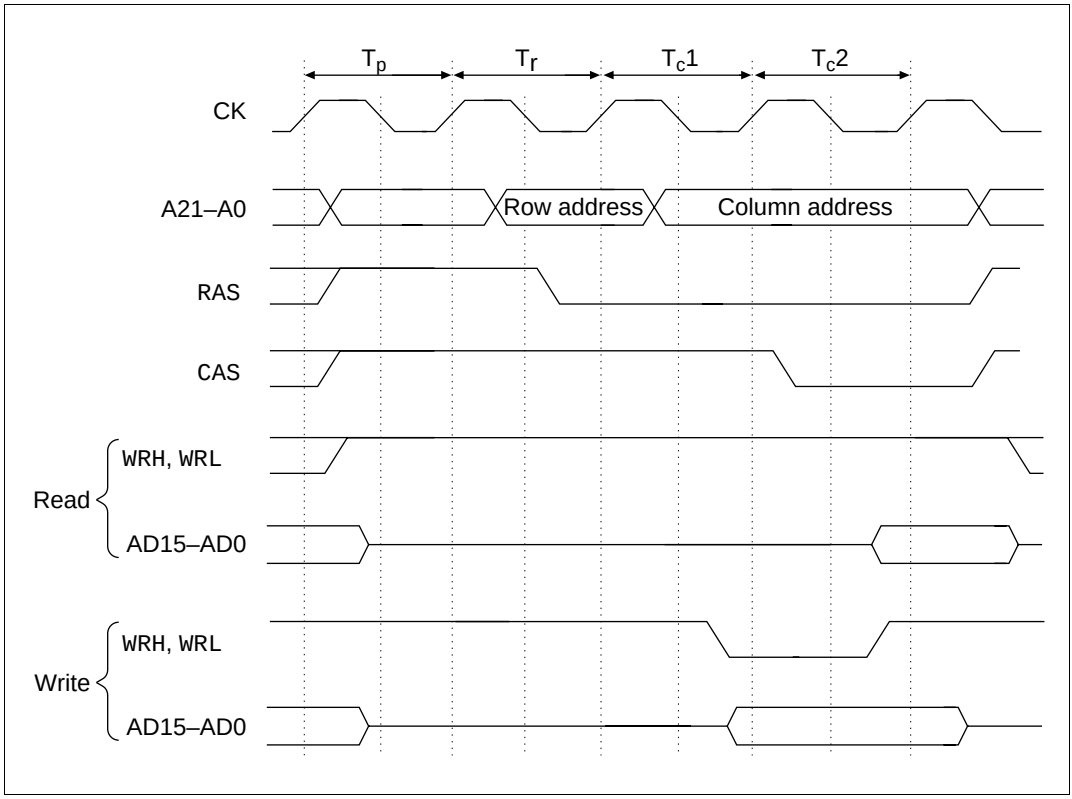



**Figure 8.18 Long Pitch Access Timing**

### 8.5.3 Wait State Control

**Precharge State Control:** When the microprocessor clock frequency is raised and the cycle period shortened, 1 cycle may not always be sufficient for the precharge time for the $\overline{\text{RAS}}$ signal when the DRAM is accessed. The BSC allows the precharge cycle to be set to 1 state or 2 states using the $\overline{\text{RAS}}$ signal precharge cycles bit (TPC) of the DCR. When the TPC bit is 0, the precharge cycle is 1 state; when TPC is 1, the precharge cycle is 2 states. Figure 8.19 shows the timing when the precharge cycle is 2 states.

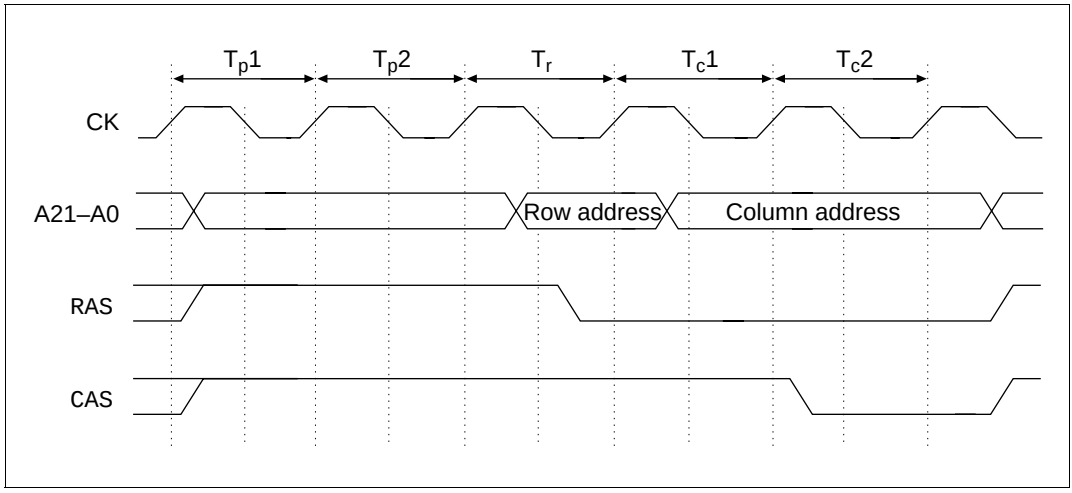



**Figure 8.19 Precharge Timing (Long Pitch)**

**Control of Insertion of Wait States Using the $\overline{\text{WAIT}}$ Pin Input Signal:** The number of wait states inserted into the DRAM access cycle can be controlled by setting WCR1 and WCR2. When the corresponding bits in WCR1 and WCR2 are cleared to 0, the column address output cycle ends in 1 state and no wait states are inserted. When the bit is 1, the $\overline{\text{WAIT}}$ pin input signal is sampled on the rise of the system clock (CK) directly preceding the second state of the column address output cycle and the wait state is inserted as long as the level is low. When a high level is detected, it shifts to the second state. Figure 8.20 shows the wait state timing in a long pitch bus cycle.

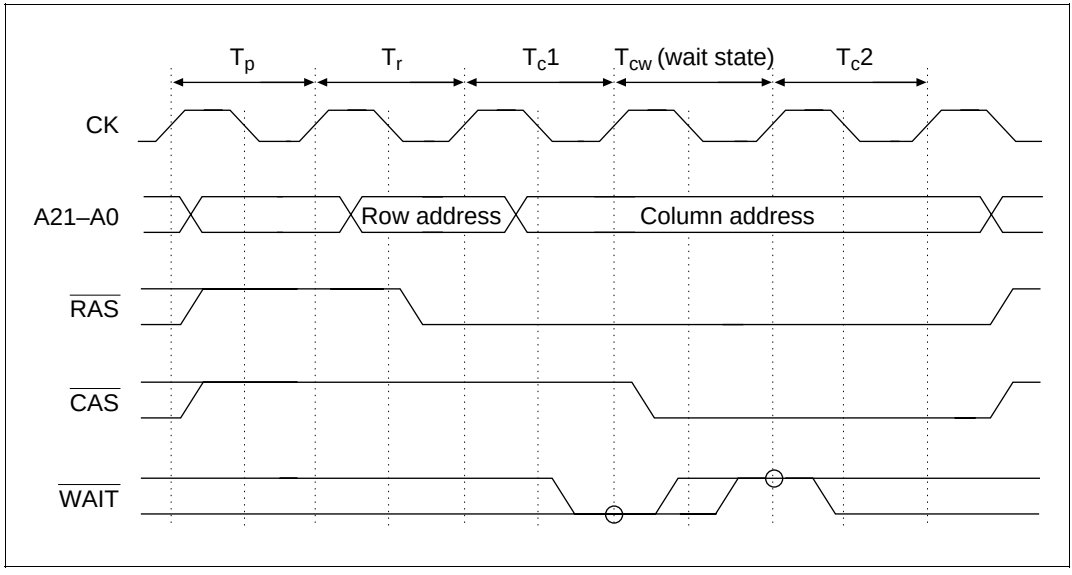


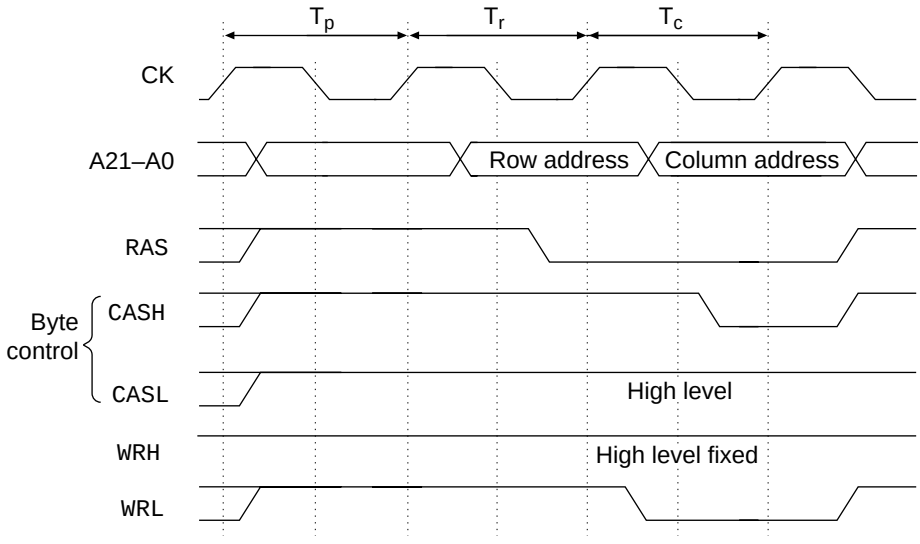

**Figure 8.20 Wait State Timing during DRAM Access (Long Pitch)**

When the RW1 bit is set to 1, the number of wait states selected by CBR refresh wait state insertion bits 1 and 0 (RLW1, RLW0) of the refresh control register (RCR) are inserted into the CAS-before-RAS refresh cycle.

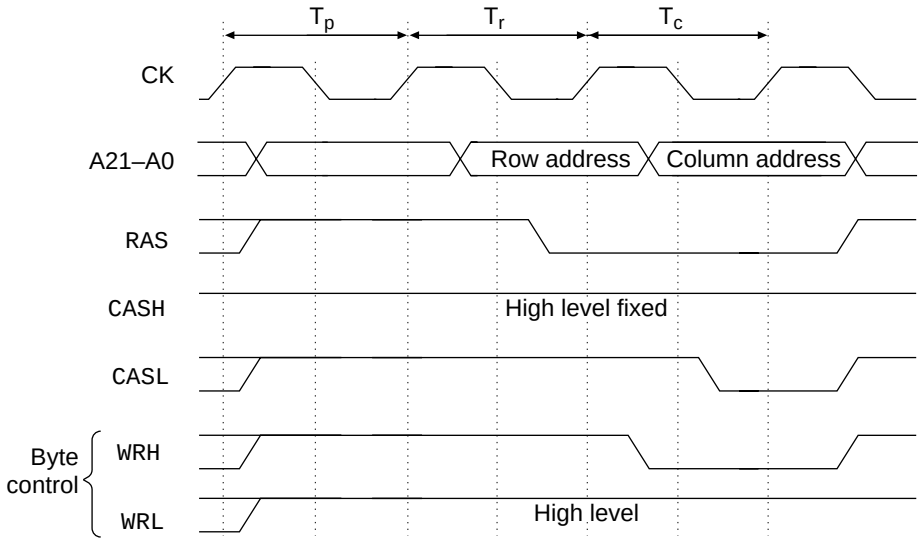
### 8.5.4 Byte Access Control

16-bit width and 18-bit width DRAMs require different types of byte control signals for access. By setting the dual CAS signals/dual WE signals select bit (CW2) in the DCR, the BSC allows selection of either the dual CAS signals or the dual WE signals system of control signals. When 16-bit space is being accessed and the CW2 bit is cleared to 0 for dual CAS signals, $\overline{\text{CASH}}$, $\overline{\text{CASL}}$, and $\overline{\text{WRL}}$ signals are output; when CW2 is set to 1 for dual WE signals, the $\overline{\text{CASL}}$, $\overline{\text{WRH}}$, and $\overline{\text{WRL}}$ signals are output. When accessing 8-bit space, $\overline{\text{WRL}}$ and $\overline{\text{CASL}}$ are output regardless of the CW2 setting.

Figure 8.21 shows the control timing of the upper byte write cycle (short pitch) in 16-bit space.




(a) Dual CAS signals (CW2 = 0)




(a) Dual WE signals (CW2 = 1)

**Figure 8.21 Byte Access Control Timing for DRAM Access (Upper Byte Write Cycle, Short Pitch)**

### 8.5.5 DRAM Burst Mode

In addition to the normal mode of DRAM access, in which row addresses are output at every access and data then accessed (full access), the DRAM also has a high-speed page mode for use when continuously accessing the same row. The high speed page mode enables fast access of data simply by changing the column address after the row address is output (burst mode). Select between full access and burst operation by setting the burst enable bit (BE)) in the DCR. When the BE bit is set to 1, burst operation is performed when the row address matches the previous DRAM access row address. Figure 8.22 shows the comparison of full access and burst operation.

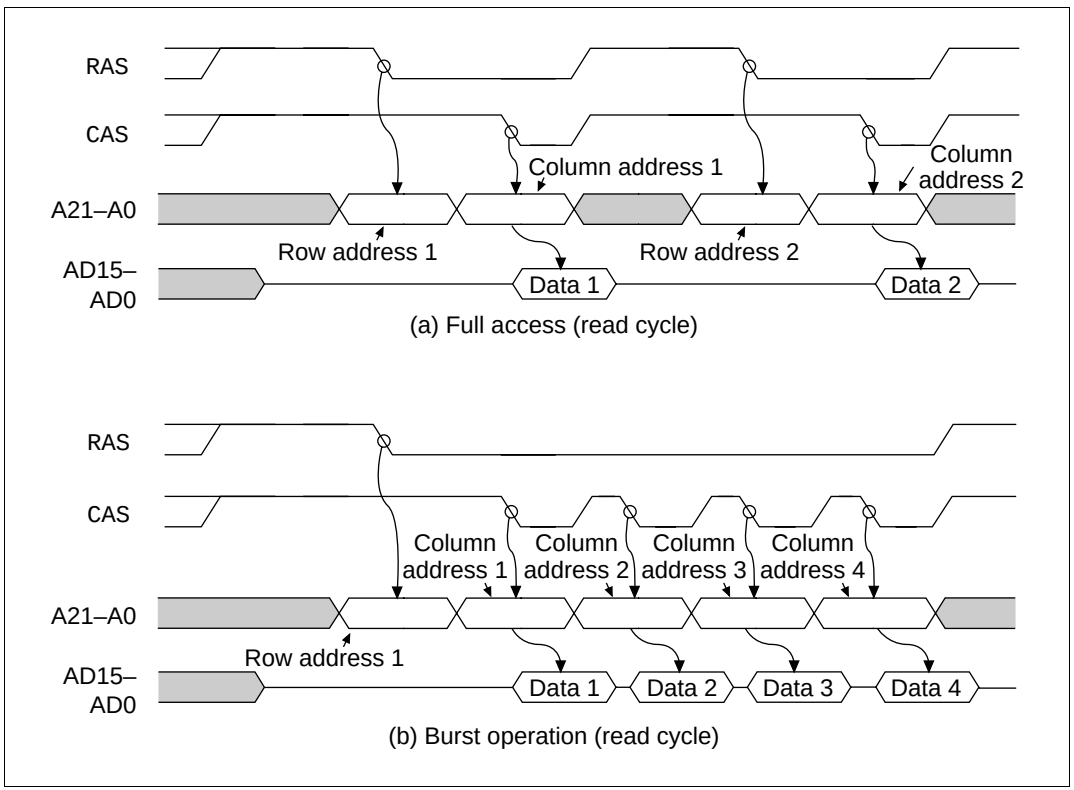



**Figure 8.22 Full Access and Burst Operation**

Short pitch high-speed page mode or long pitch high-speed page mode burst transfers can be selected independently for DRAM read/write cycles even when the burst operation is selected by using the bits corresponding to area 1 in WCR1 and WCR2 (RW1, WW1, DRW1, DWW1). The RAS down mode or RAS up mode can be selected by setting the RAS down bit (RASD) of the DCR when there is an access outside the DRAM space during burst operation.

**Short Pitch High-Speed Page Mode and Long Pitch High-Speed Page Mode:** When burst operation is selected by setting the DCR's BE bit to 1, the short pitch high-speed page mode or long pitch high-speed page mode can be selected by setting the RW1, WW1, DRW1, and DWW1 bits of the WCR1 and WCR2.

- Short-pitch, high-speed page mode: When the RW1, WW1, DRW1 and DWW1 bits in the WCR1 and WCR2 are cleared to 0, and the corresponding DRAM access cycle is continuing, the $\overline{CAS}$ signal and column address output cycles continue as long as the row addresses continue to match. The column address output cycle is performed in 1 state and the $\overline{WAIT}$ signal is not sampled. Figure 8.23 shows the read cycle timing for the short pitch high-speed page mode.

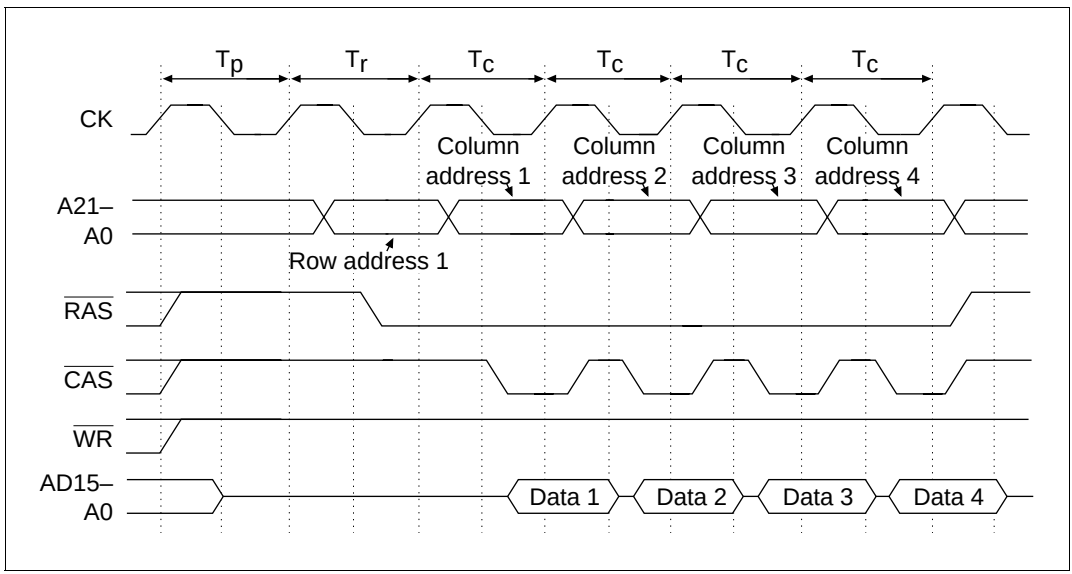
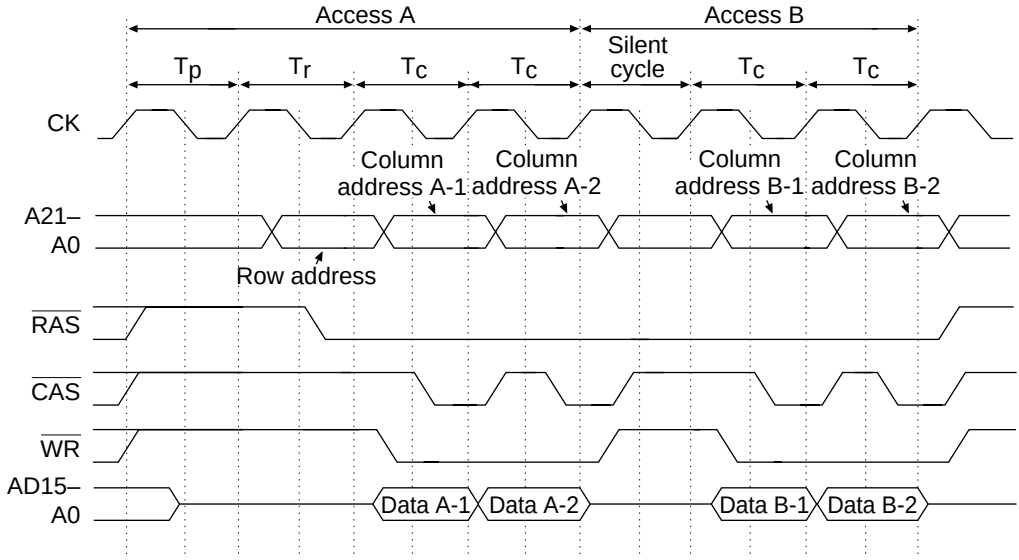


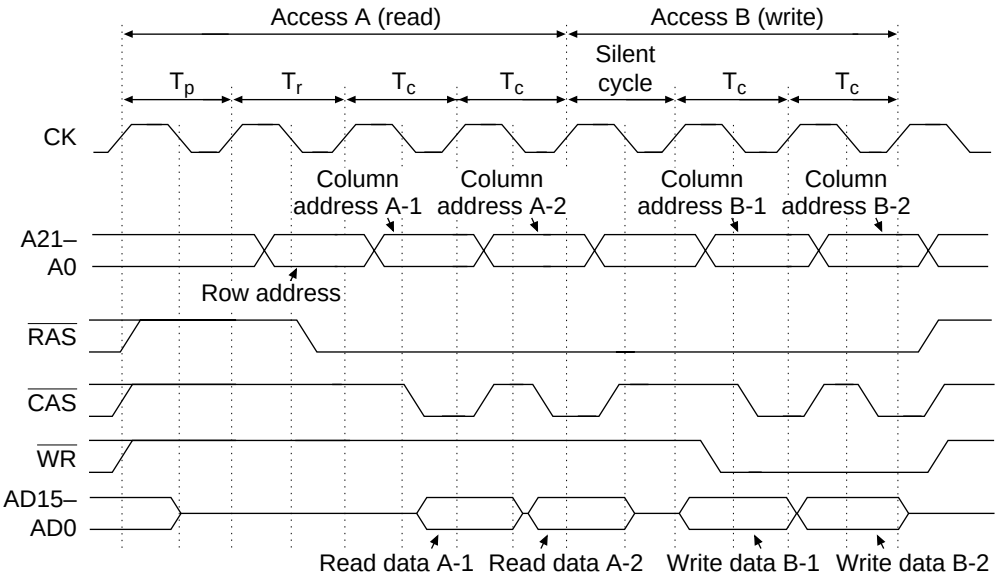

**Figure 8.23 Short Pitch High-Speed Page Mode (Read Cycle)**

When the write cycle continues for the same row address in the short pitch high-speed page mode, an open cycle (silent cycle) is produced for 1 cycle only. This timing is shown in figure 8.24. Likewise, when a write cycle continues after the read cycle for the same row address, a silent cycle is produced for 1 cycle. This timing is shown in figure 8.25. Note also that when DRAM is written to in short-pitch, high-speed page mode when using DMAC single address mode, a silent cycle is inserted in each transfer. The details of timing are discussed in section 20.3.3, Bus Timing.




Note: Access A and B are examples of 32-bit data accesses in their respective 16-bit bus width spaces.

**Figure 8.24 Short Pitch High-Speed Page Mode (Write Cycle)**




Note: Access A and B are examples of 32-bit data accesses in their respective 16-bit bus width spaces.

**Figure 8.25 Short Pitch High-Speed Page Mode (When read and write cycle continues with the same row address)**

The high-level duty of the $\overline{CAS}$ signal can be selected in the short pitch high-speed page mode using the CAS duty bit (CDTY) in the DCR. When the CDTY bit is cleared to 0, high-level duty is 50% of the TC state; when CDTY is set to 1, it is 35% of the $T_C$ state.

- Long-pitch, high-speed page mode: When the RW1, WW1, DRW1, and DWW1 bits in WCR1 and WCR2 are set to 1, and the corresponding DRAM access cycle is continuing, the $\overline{CAS}$ signal and column address output cycles (2 states) continue as long as the row addresses continue to match. When the $\overline{WAIT}$ signal is detected at the low level, the second cycle of the column address output cycle is repeated as the wait state. Figure 8.26 shows the timing for the long pitch high-speed page mode. See section 20.3.3, Bus Timing, for more information about the timing.

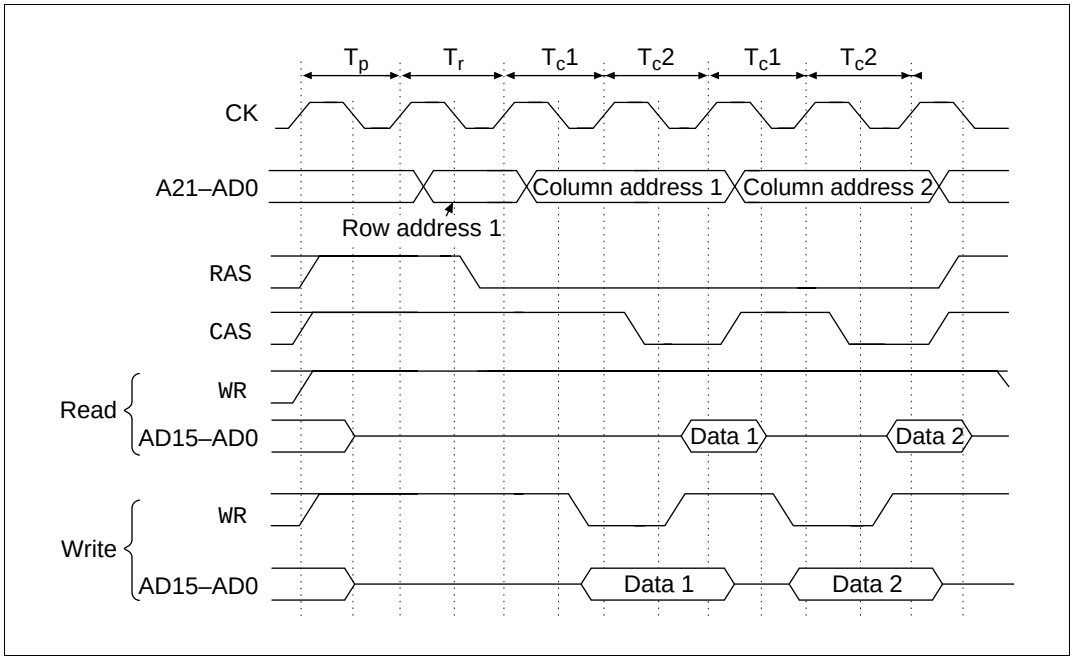



**Figure 8.26 Long Pitch High-Speed Page Mode (Read/Write Cycle)**

**RAS Down Mode and RAS Up Mode:** Sometimes access to another area can occur between accesses to the DRAM even though burst operation has been selected. Keeping the $\overline{\text{RAS}}$ signal at low while this other access is occurring allows burst operation to continue the next time the same row of the DRAM is accessed. The RASD bit in the DCR selects the RAS down mode when set to 1 and the RAS up mode when cleared to 0. In both RAS down mode and RAS up mode, burst operation is continued while the same row address continues to be accessed, even if the bus master is changed.

- RAS Down Mode: When the RASD bit of DCR is set to 1, the DRAM access pauses and the $\overline{\text{RAS}}$ signal is held low throughout the access of the other space while waiting for the next access to the DRAM area. When the row address for the next DRAM access is the same as the previous DRAM access, burst operation continues. Figure 8.27 shows the timing of the RAS down mode when external memory space is accessed during burst operation.

  The $\overline{\text{RAS}}$ signal can be held down in the DRAM for a limited time; the $\overline{\text{RAS}}$ signal must be returned to high within the specified limits even when the RAS down mode is selected since the critical low level period is set. In this LSI, even when the RAS down mode is selected, the $\overline{\text{RAS}}$ signal automatically reverts to high when the DRAM is refreshed, so the BSC's refresh control function can be employed to set a CAS-before-RAS refresh that will keep operation within specifications. See section 8.5.6, Refresh Control, for details.

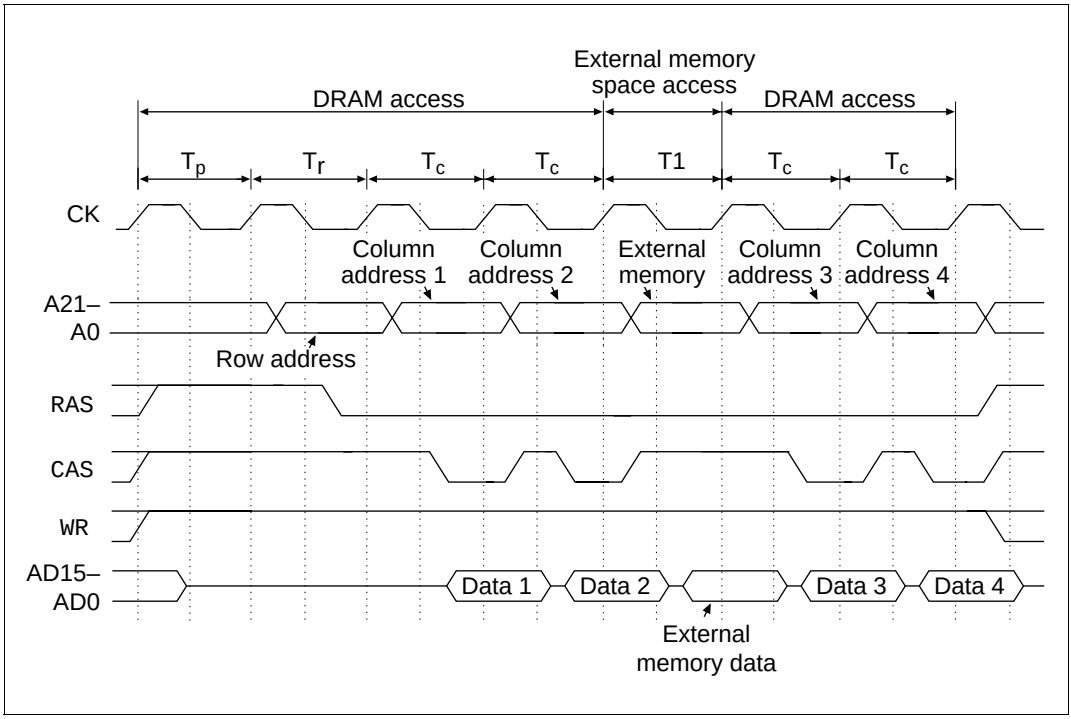



**Figure 8.27 RAS Down Mode**

- RAS Up Mode: When the RASD bit is cleared to 0, the $\overline{\text{RAS}}$ signal reverts to high whenever a DRAM access pauses for access to another space. Burst operation continues only while DRAM access is continuous. Figure 8.28 shows the timing when an external memory space access occurs during burst operation in the RAS up mode.

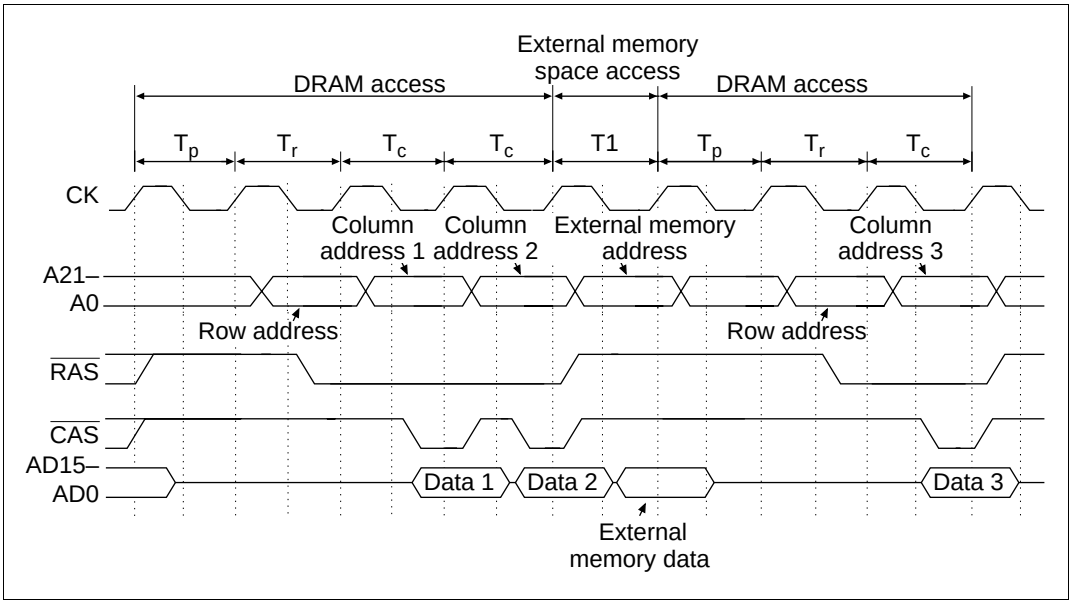



**Figure 8.28 RAS Up Mode**

### 8.5.6 Refresh Control

The BSC has a function for controlling DRAM refreshing. By setting the refresh mode bit (RMODE) in the refresh control register (RCR), either CAS-before-RAS refresh (CBR) or self-refresh can be selected. When no refresh is performed, the refresh timer counter (RTCNT) can be used as an 8-bit interval timer.

**CAS-Before-RAS Refresh (CBR):** A refresh is performed at an interval determined by the input clock selected in the clock select bits 2–0 (CKS2–CKS0) of the refresh timer control/status register (RTCSR) and the value set in the refresh time constant register (RTCOR). Set the values of RTCOR and CKS2–CKS0 so they satisfy the refresh interval specifications of the DRAM being used.

To perform a CBR refresh, clear the RMODE bit of the RCR to 0 and then set the refresh control bit (RFSHE) bit to 1. Also write in the required values to RTCNT and RTCOR. When the clock is thereafter selected in the CKS2–CKS0 bits of the RTCSR, the RTCNT will begin to increment from its current value. The RTCNT value is constantly compared to the RTCOR value and the CBR refresh is performed when they match. The RTCNT is simultaneously cleared to H'00 and incrementing begins again.

When the clock is selected in the CKS2–CKS0 bits, the RTCNT immediately begins to increment from its current value. This means that when the RTCOR cycle is set after the CKS2–CKS0 bits are set, the RTCNT count may already be higher than the RTCOR cycle. When this occurs, the RTCNT will overflow once (H'FF goes to H'00) and incrementing will start again. Since the CBR

refresh will not be performed until the RTCNT again matches the RTCOR value, the initial refresh interval will be rather long. It is thus advisable to set the RTCOR cycle prior to setting the CKS2–CKS0 bits and start it incrementing. When CBR refresh control is being performed after its use as an 8-bit interval timer, the RTCNT count value may be in excess of the refresh cycle. For this reason, clear the RTCNT by writing H'00 before starting refresh control to assure a correct refresh interval.

When the RW1 bit of WCR1 is set to 1 and the read cycle is set to long pitch, the number of wait states selected by the RLW1 and RLW0 bits of the RCR will be inserted into the CBR refresh cycle, regardless of the status of the $\overline{\text{WAIT}}$ signal. Figure 8.29 shows the RTCNT operation and figure 8.30 shows the timing of the CBR refresh. For details on timing, see section 20.3.3, Bus Timing.

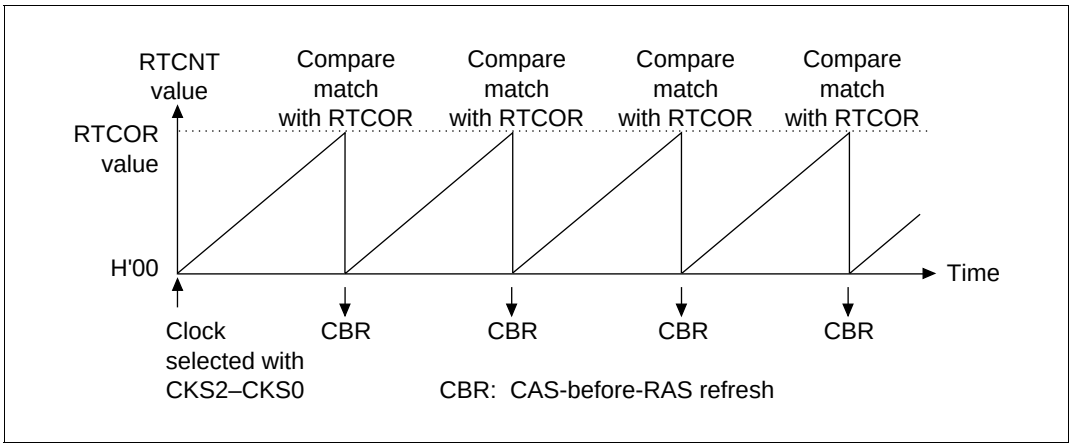



**Figure 8.29 Refresh Timer Counter (RTCNT) Operation**

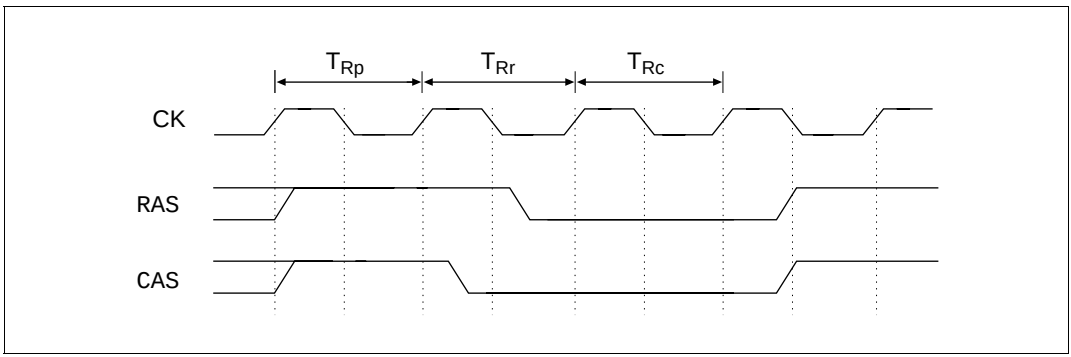



**Figure 8.30 Output Timing for CAS-Before-RAS Refresh Signal**

**Self-Refresh Mode:** Some DRAMs have a self-refresh mode (parity back-up mode). This is a type of a standby mode in which the refresh timing and refresh addresses are generated inside the DRAM chip. When the RFSHE and RMODE bits of the RCR are both set to 1, the DRAM will

enter the self-refresh mode when the $\overline{CAS}$ and $\overline{RAS}$ signals are output as shown in figure 8.31. See section 20.3.3, Bus Timing, for details. The DRAM self-refresh mode is cleared when the RMODE bit in the RCR is cleared to 0 (figure 8.31). The RFSHE bit should be left at 1 when this is done. Some DRAM vendors recommend that after exiting the self-refresh mode, all row addresses should be refreshed again. This can be done using the BCR's CBR refresh function to set all row addresses for refresh in software.

To access a DRAM area in the self-refresh mode, clear the RMODE bit to 0 and exit the self-refresh mode.

The LSI can be kept in the self-refresh state and shifted to standby mode by setting it to self-refresh mode, setting the standby bit (SBY) of the standby control register (SBYCR) to 1, and then executing a SLEEP instruction.

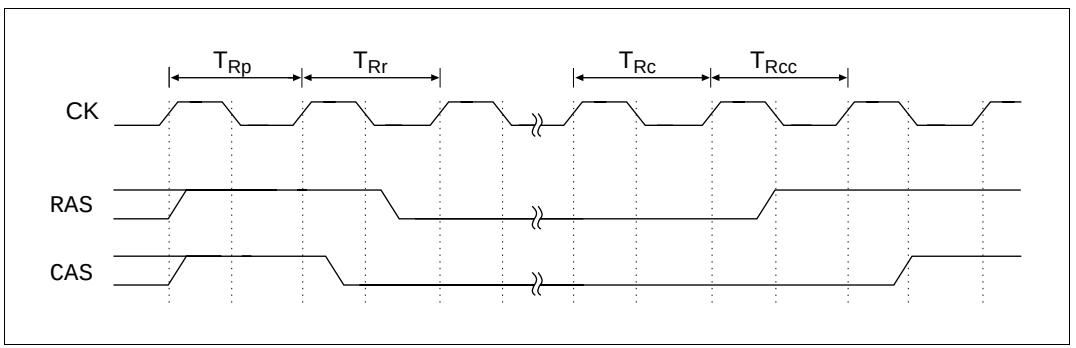



**Figure 8.31 Output Timing of Self-Refresh Signal**

**Refresh Requests and Bus Cycle Requests:** When a CAS-before-RAS refresh or self-refresh is requested during bus cycle execution, parallel execution is sometimes possible. Table 8.10 describes operation when the refresh and bus cycle are in contention.

**Table 8.10 Refresh and Bus Cycle Contention**

| Type of Refresh | Type of Bus Cycle: External Space Access: External Memory Space, Multiplexed I/O Space: Read Cycle | External Memory Space, Multiplexed I/O Space: Write Cycle | DRAM Space: Read Cycle | DRAM Space: Write Cycle | On-Chip ROM, On-Chip RAM , On-Chip Peripheral Access |
|---|---|---|---|---|---|
| CAS-before-RAS refresh | Yes | No | No | No | Yes |
| Self-refresh | Yes | Yes | No | No | Yes |

Yes: Can be executed in parallel

No: Cannot be executed in parallel

When parallel execution is available, the $\overline{RAS}$ and $\overline{CAS}$ signals are output simultaneously during bus cycle execution and the refresh is executed. When parallel execution is not available, refresh occurs after the bus cycle has ended.

**Using RTCNT as an 8-Bit Interval Timer:** When not performing refresh control, RTCNT can be used as an 8-bit interval timer. Simply set the RFSHE bit of the RCR to 0. To produce a compare match interrupt (CMI), set the compare match interrupt enable bit (CMIE) to 1 and set the interrupt generation timing in RTCOR. When the input clock is selected with the CKS2–CKS0 bits of the RTCSR, RTCNT starts incrementing as an 8-bit interval timer. Its value is constantly being compared to RTCOR and when a match occurs, the CMF bit of RTCSR is set to 1 and a CMI interrupt is produced. RTCNT is cleared to H'00.

When the clock is selected with CKS2–CKS0 bits, RTCNT starts incrementing immediately. This means that when the RTCOR cycle is set after the CKS2–CKS0 bits are set, the RTCNT count may already be higher than the RTCOR cycle. When this occurs, the RTCNT will overflow once (H'FF goes to H'00) and the count up will start again. No interrupt will be generated until the RTCNT again matches the RTCOR value. It is thus advisable to set the RTCOR cycle prior to setting the CKS2–CKS0 bits. After its use as an 8-bit interval timer, the RTCNT count value may be in excess of the set cycle. For this reason, write H'00 to the RTCNT to clear it before starting to use it again with new settings. RTCNT can then be restarted and an interrupt obtained after the correct interval.

## 8.6 Address/Data Multiplexed I/O Space Access

The BSC is equipped with a function that multiplexes input/output of address and data to pins AD15–AD0 in area 6. This allows the SH microprocessor to be directly connected to peripheral LSIs that required address/data multiplexing.

### 8.6.1 Basic Timing

When the multiplexed I/O enable bit (IOE) of the BCR is set to 1, the area 6 space with address bit A27 as 0 (H'6000000–H'6FFFFFF) becomes an address/data multiplexed I/O space that, when accessed, multiplexes addresses and data. When the A14 address bit is 0, the bus width is 8 bits and address output and data input/output are performed from the AD7–AD0 pins. When the A14 address bit is 1, the bus width is 16 bits and address output and data input/output are performed from the AD15–AD0 pins. In the address/data multiplexed I/O space, access is controlled with the $\overline{AH}$, $\overline{RD}$ and $\overline{WR}$ signals. Accesses in the address/data multiplexed I/O space is performed in 4 states, regardless of the WCR settings. Figure 8.32 shows the timing when the address/data multiplexed I/O space is accessed.

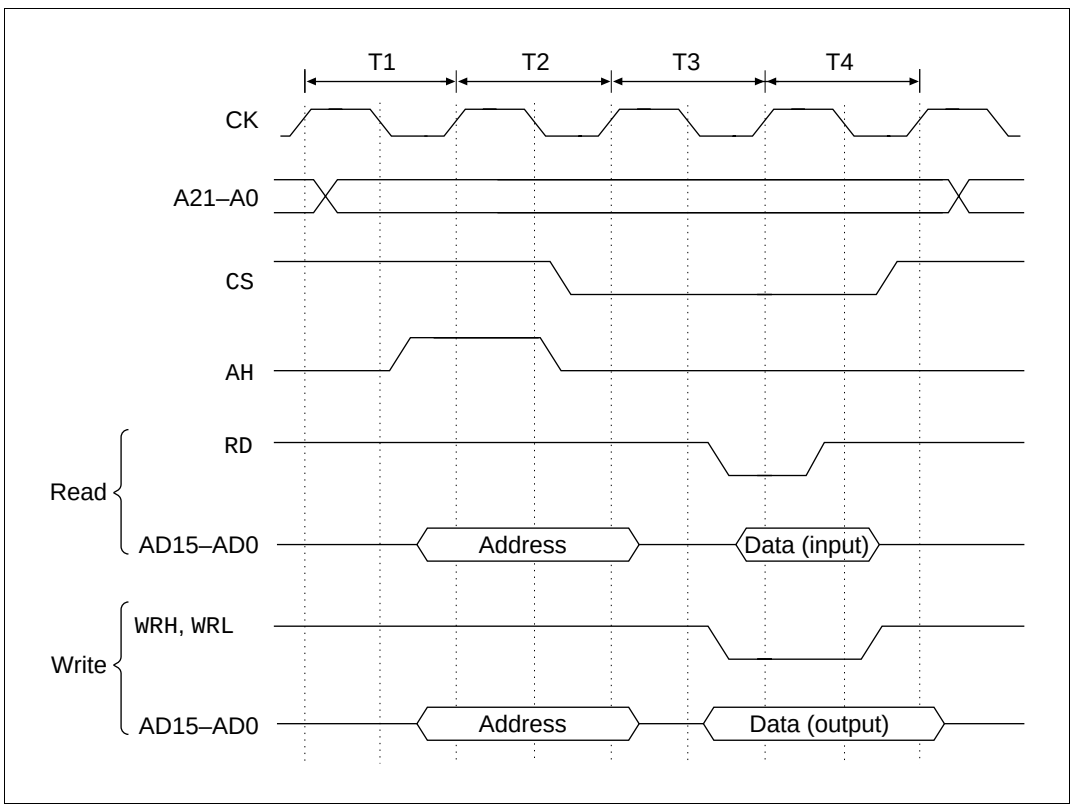



**Figure 8.32 Access Timing For Address/Data Multiplexed I/O Space**

The high-level duty of the $\overline{RD}$ signal can be selected between 35% and 50% using the RD duty bit (RDDTY) of the BCR. When RDDTY is 1, the high-level duty is 35% of the T3 or Tw state, lengthening the access time for external devices.

### 8.6.2 Wait State Control

When the address/data multiplexed I/O space is accessed, the $\overline{\text{WAIT}}$ pin input signal is sampled and a wait state inserted whenever a low level is detected, regardless of the setting of the WCR. Figure 8.33 shows an example in which a $\overline{\text{WAIT}}$ signal causes a wait state of 1 state to be inserted.

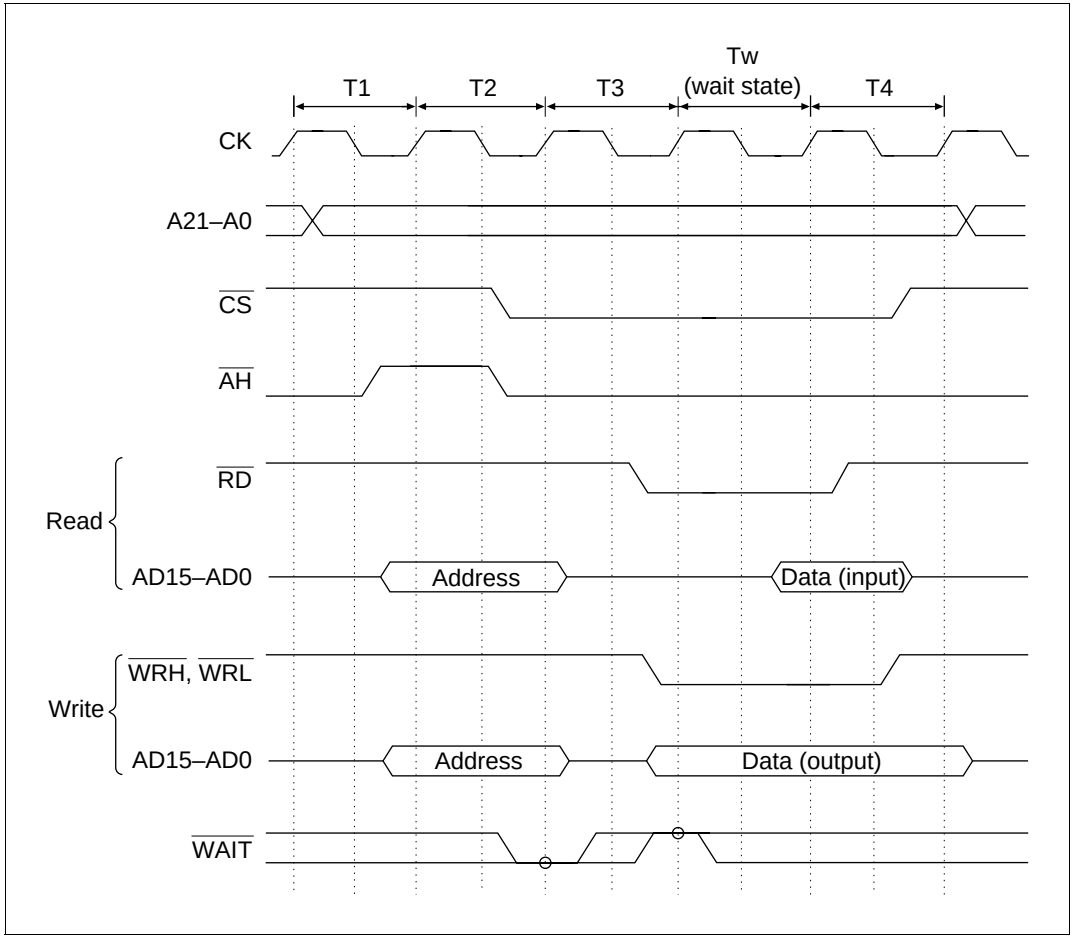



**Figure 8.33 Wait State Timing For Address/Data Multiplexed I/O Space Access**

### 8.6.3 Byte Access Control

The byte access control signals when the address/data multiplexed I/O space is being accessed are of two types ($\overline{\text{WRH}}$, $\overline{\text{WRL}}$, A0, or $\overline{\text{WR}}$, $\overline{\text{HBS}}$, $\overline{\text{LBS}}$), just as for byte access control of external memory space access. These types can be selected using the BAS bit of the BCR. See section 8.4.3, Byte Access Control, for details.

## 8.7 Parity Check and Generation

The BSC can check and generate parity for data input and output to or from in the DRAM space of area 1 and the external memory space of area 2.

To check and generate parity, select the space (DRAM space only, or DRAM space and area 2) for which parity is to be checked and generated using the parity check enable bits (PCHK1 and PCHK0) of the parity control register and select odd or even parity in the parity polarity bit (PEO).

When data is input from the space selected in the PCHK1 and PCHK0 bits, the BSC checks the PEO bit to see if the polarity of the DPH pin input (upper byte parity data) is accurate for the AD15–AD8 pin input (upper byte data) or if the DPL pin input (lower byte parity data) is accurate for the AD7–AD0 pin input (lower byte data). If the check indicates that either the upper or lower byte parity is incorrect, a parity error interrupt is produced (PEI).

When outputting data to the space selected in the PCHK1 and PCHK0 bits, the BSC outputs parity data output of the polarity set in the PEO bit from the DPH pin for the AD15–AD8 pin output (upper byte data) or from the DPL pin for the AD7–AD0 pin input (lower byte data) using the same timing as the data output.

The BSC is also able to force a parity output for use in testing the system's parity error check function. When the parity force output bit (PFRC) of the PCR is set to 1, a high level is forcibly output from the DPH and DPL pins when data is output to the space selected in the PCHK1 and PCHK0 bits.

## 8.8 Warp Mode

In warp mode, an external write cycle or DMA single address mode transfer cycle and an internal access cycle (read/write to on-chip memory or on-chip peripheral modules) operate independently in parallel. The warp mode is entered by setting the warp mode bit (WARP) in the BCR to 1. This allows the LSI to be operated at high speed.

When in the warp mode an external write cycle or DMA single address mode transfer cycle continues for at least 2 states and their is an internal access, only the external write cycle will be performed in the initial state. The external write cycle and internal access cycle will be performed in parallel from the next state on, without waiting for the end of the external write cycle. Figure 8.34 shows the timing when an access to an on-chip peripheral module and an external write cycle are performed in parallel.

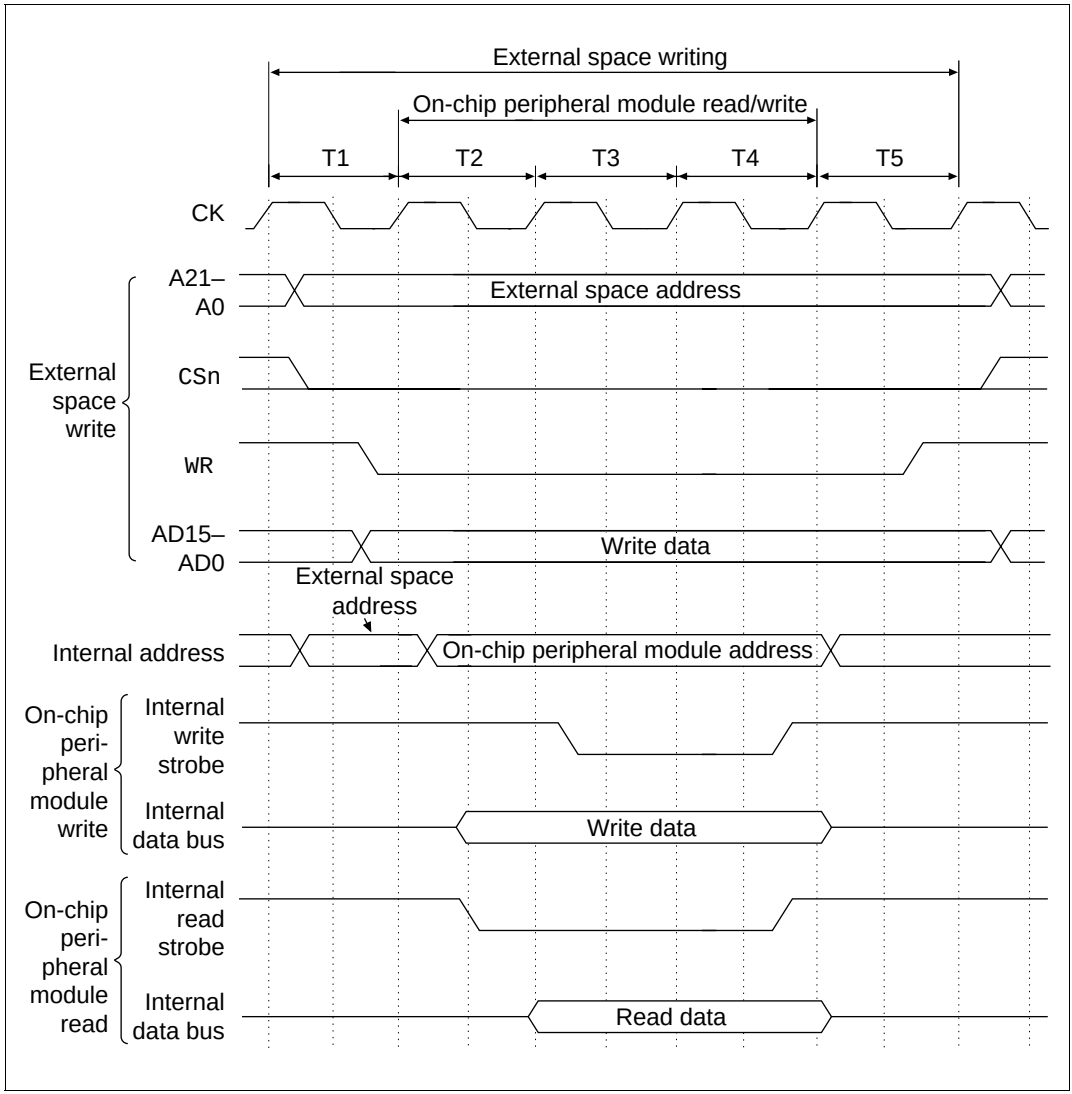



**Figure 8.34 Warp Mode Timing (Access to On-Chip Peripheral Module and External Write Cycle)**

## 8.9 Wait State Control

The WCR1–WCR3 registers of the BSC can be set to control sampling of the $\overline{\text{WAIT}}$ signal when accessing various areas and the number of bus cycle states. Table 8.11 shows the number of bus cycle states when accessing various areas.

**Table 8.11 Bus Cycle States when Accessing Address Spaces**

| | CPU Read Cycle, DMAC Dual Mode Read Cycle, DMAC Single Mode Memory Read/Write Cycle | |
|---|---|---|
| **Address Space** | **Corresponding Bits in WCR1 and WCR2 = 0** | **Corresponding Bits in WCR1 and WCR2 = 1** |
| External memory (areas 1, 3–5, 7) | 1 state fixed; WAIT signal ignored | 2 states + wait states from WAIT signal |
| External memory (Areas 0, 2, 6; long wait available) | 1 state + long wait state*, WAIT signal ignored | 1 state + long wait state* + wait states from WAIT signal |
| DRAM space (area 1) | Column address cycle: 1 state, WAIT signal ignored (short pitch) | Column address cycle: 2 states + wait states from WAIT signal (long pitch) |
| Multiplexed I/O space (area 6) | 4 states + wait states from WAIT signal | |
| On-chip peripheral module space (area 5) | 3 states fixed, WAIT signal ignored | |
| On-chip ROM (area 0) | 1 state fixed, WAIT signal ignored | |
| On-chip RAM (area 7) | 1 state fixed, WAIT signal ignored | |

| | CPU Write Cycle, DMAC Dual Mode Memory Write Cycle | |
|---|---|---|
| **Address Space** | **WW1 of WCR1 = 0** | **WW1 of WCR1 =1** |
| External memory (areas 1, 3–5, 7) | 2 states + wait states from WAIT signal | |
| External memory (Areas 0, 2, 6; long wait available) | 1 state + long wait state* + wait states from WAIT signal | |
| DRAM space (area 1) | Column address cycle: 1 state, WAIT signal ignored (short pitch) | Column address cycle: 2 states + wait states from WAIT signal (long pitch) |
| Multiplexed I/O space (area 6) | 4 states + wait states from WAIT signal | |
| On-chip peripheral module space (area 5) | 3 states fixed, WAIT signal ignored | |
| On-chip ROM (area 0) | 1 state fixed, WAIT signal ignored | |
| On-chip RAM (area 7) | 1 state fixed, WAIT signal ignored | |

Note: The number of long wait states (1 to 4) is set in WCR3.

For details on bus cycles when external spaces are accessed, see section 8.4, External Memory Space Access, section 8.5, DRAM Space Access, and section 8.6, Address/Data Multiplexed I/O Space Access.

Accesses of on-chip spaces are as follows: On-chip peripheral module spaces (area 5 when address bit A27 is 1) are always 3 states, regardless of the WCR, with no $\overline{\text{WAIT}}$ signal sampling. Accesses of on-chip ROM (area 0 when MD2–MD0 are 010) and on-chip RAM (area 7 when address bit A27 is 0) are always performed in 1 state, regardless of the WCR, with no $\overline{\text{WAIT}}$ signal sampling.

If the bus timing specifications ($t_{WTS}$ and $t_{WTH}$) are not observed when the $\overline{\text{WAIT}}$ signal is input in external space access, this will simply mean that $\overline{\text{WAIT}}$ signal assertion and negation will not be detected, but will not result in misoperation. Note, however, that the inability to detect $\overline{\text{WAIT}}$ signal assertion may result in a problem with memory access due to insertion of an insufficient number of waits.

## 8.10 Bus Arbitration

The SuperH microcomputer can release the bus to external devices when they request the bus. It has two internal bus masters, the CPU and the DMAC. Priorities for releasing the bus for these two are as follows.

Bus request from external device > refresh > DMAC > CPU

Thus, an external device has priority when it generates a bus request, even when the DMAC is doing a burst transfer.

Note that when a refresh request is generated while the bus is released to an external device, $\overline{\text{BACK}}$ becomes high level and the bus right can be acquired to perform the refresh upon receipt of a $\overline{\text{BREQ}}$ = high level response from the external device. Input all bus requests from external devices to the $\overline{\text{BREQ}}$ pin. The signal indicating that the bus has been released is output from the $\overline{\text{BACK}}$ pin. Figure 8.35 illustrates the bus release procedure.

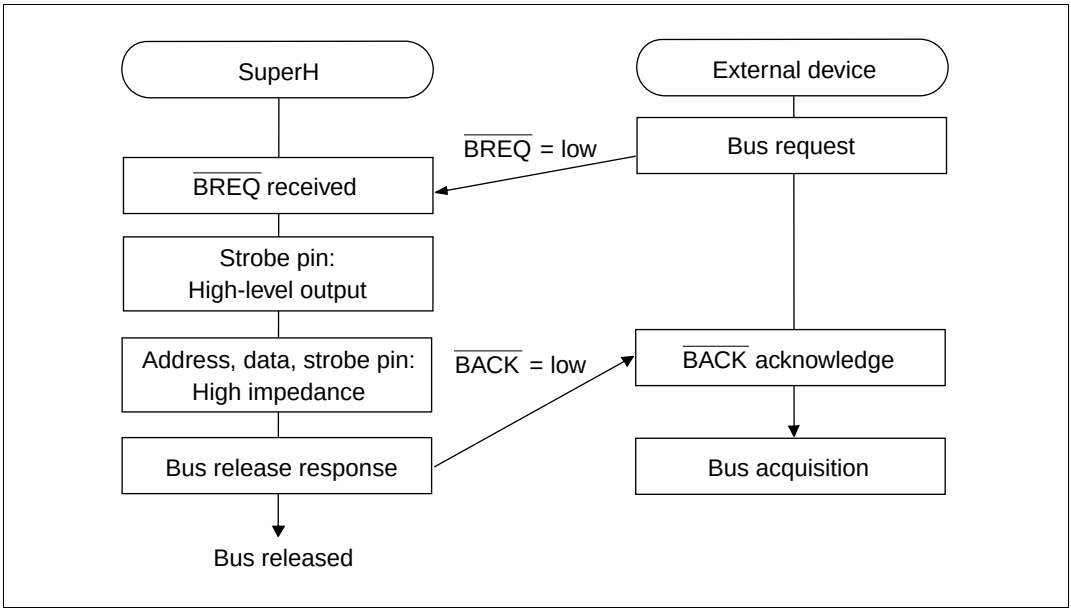



**Figure 8.35 Bus Release Procedure**

### 8.10.1 The Operation of Bus Arbitration

This LSI has the bus arbitration function which can give bus ownership to an external device when the device requests the bus ownership. When $\overline{\text{BREQ}}$ is input and the bus cycle being executed by the CPU or DMAC is completed, $\overline{\text{BACK}}$ becomes low and a bus is released for an external device. At this time, the following operates when bus arbitration conflicts with refresh.

1. If DRAM refresh is requested in this LSI when a bus is released and $\overline{\text{BACK}}$ is low, $\overline{\text{BACK}}$ becomes high and the occurrence of the refresh request can be informed externally. At this time, the external device may generate a bus cycle when $\overline{\text{BREQ}}$ is low even if $\overline{\text{BACK}}$ is high. Therefore, a bus remains released to the external device. Then, when $\overline{\text{BREQ}}$ becomes high, this LSI gets bus ownership, and executes refresh and the bus cycle of the CPU or DMAC. After the external device gets bus ownership and $\overline{\text{BACK}}$ is low, refresh is requested when $\overline{\text{BACK}}$ becomes high even if the low level of $\overline{\text{BREQ}}$ is input. Therefore, turn $\overline{\text{BREQ}}$ high immediately to release a bus for this LSI to hold DRAM data (See figure 8.36).

2. When $\overline{\text{BREQ}}$ changes from high to low and internal refresh is requested at the timing of the bus release of this LSI, $\overline{\text{BACK}}$ may remain high (do not become low). A bus is released to the external device since the low level of $\overline{\text{BREQ}}$ is input. This operation is based on the above specification (1). To hold DRAM data, turn $\overline{\text{BREQ}}$ high and release a bus to this LSI immediately when the external device detects that $\overline{\text{BACK}}$ does not change to low during a fixed time this LSI (See figure 8.37). When a refresh request is generated and $\overline{\text{BACK}}$ returns to high, as shown in figure 8.37, a momentary narrow pulse-shaped spike may be output where $\overline{\text{BACK}}$ was originally supposed to become low.

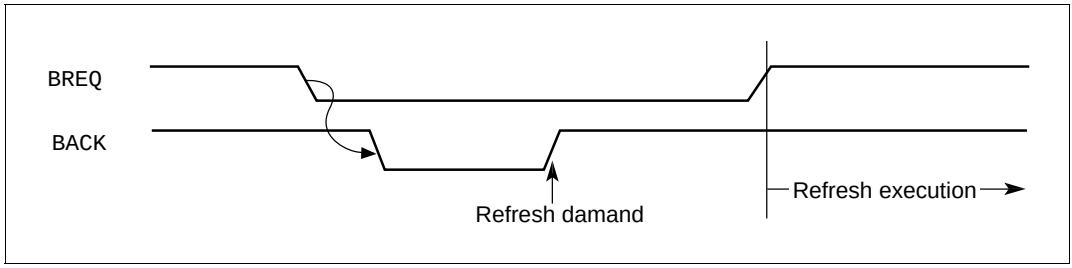
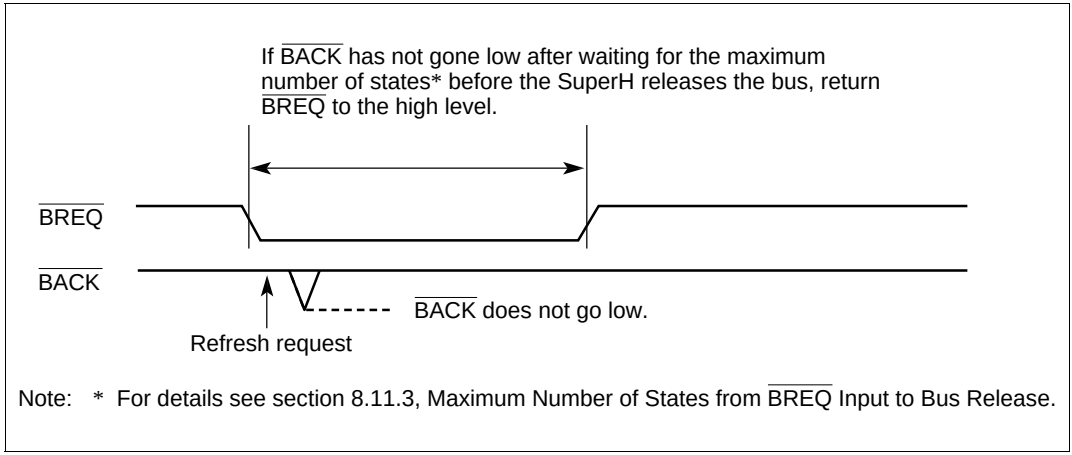



**Figure 8.36 $\overline{\text{BACK}}$ Operation by Refresh Demand (1)**



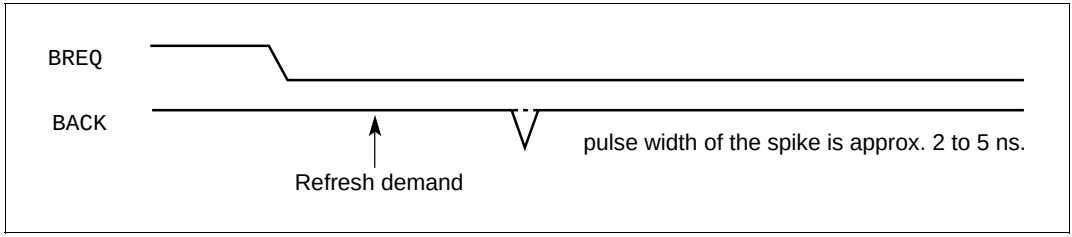

Note: * For details see section 8.11.3, Maximum Number of States from $\overline{\text{BREQ}}$ Input to Bus Release.

**Figure 8.37 $\overline{\text{BACK}}$ Operation in Response to Refresh Request (2)**

### 8.10.2 $\overline{\text{BACK}}$ Operation

1. $\overline{\text{BACK}}$ Operation

   When an internal refresh is requested during an attempt to assert the $\overline{\text{BACK}}$ signal and $\overline{\text{BACK}}$ is not asserted but remains high, a momentary narrow pulse-shaped spike may be output, as shown below.



2. Countermeasure against a spike on the $\overline{\text{BACK}}$ signal

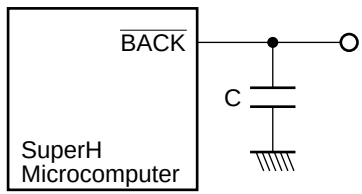
   The following describes the countermeasure against a spike on the $\overline{\text{BACK}}$ signal:

   a. When $\overline{\text{BREQ}}$ is input to release the bus of the LSI, make sure that conflicts with a refresh operation do not occur. Stop the refresh operation or operate the refresh timer counter (RTCNT) or the refresh time constant register (RTCOR) of the bus controller (BSC) to shift the refresh timing.

   b. The spike on the $\overline{\text{BACK}}$ signal has a narrow pulse width of approximately 2 to 5 ns, which can be eliminated by using a capacitor as shown in the figure below.

      For example, adding a capacitance of 220 pF can raise the minimum voltage of the spike above 2.0 V.

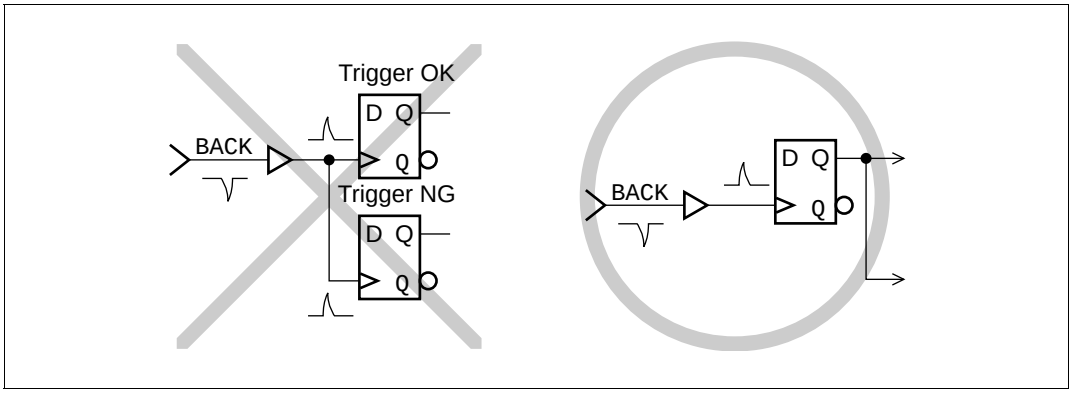
      Note that delay of the $\overline{\text{BACK}}$ signal increases approximately in units of 0.1 ns/pF. (When a capacitance of 220 pF is added, the delay increases approximately by 22 ns.




Capacitor-incorporating circuit for eliminating a spike

   c. Latching the $\overline{\text{BACK}}$ signal by using a flip-flop or triggering the flip-flop may be successful or unsuccessful due to the narrow pulse width of the spike. Implement a circuit configuration which will cause no problems when latching BACK or using $\overline{\text{BACK}}$ as a trigger signal.

      When splitting the $\overline{\text{BACK}}$ signal into two signals and latching each of them using the flip-flop or triggering the flip-flop, the flip-flop may operate for one signal but may not for another. To capture the BACK signal using the flip-flop, receive the $\overline{\text{BACK}}$ signal using a single flip-flop then distribute the signal (see figure below).




## 8.11 Usage Notes

### 8.11.1 Usage Notes on Manual Reset

Condition: When DRAM (long-pitch mode) is used and manual reset is performed.

The low width of $\overline{\text{RAS}}$ output may be shorter than usual in rese + (2.5tcyc→1.5tcyc), causing the specified value (tRAS) of DRAM not to be satisfied.

| Corresponding DRAM conditions: | long pitch/normal mode<br>long pitch/high-speed page mode |
|---|---|

There are no problems regarding operations except for the above conditions.

There are the following four cases (Figure 8.38 to Figure 8.41) for the output states of DRAM control signals ($\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and $\overline{\text{WR}}$) corresponding to $\overline{\text{RES}}$ latch timing. Actual output levels are shown by solid lines (not by dashed lines).

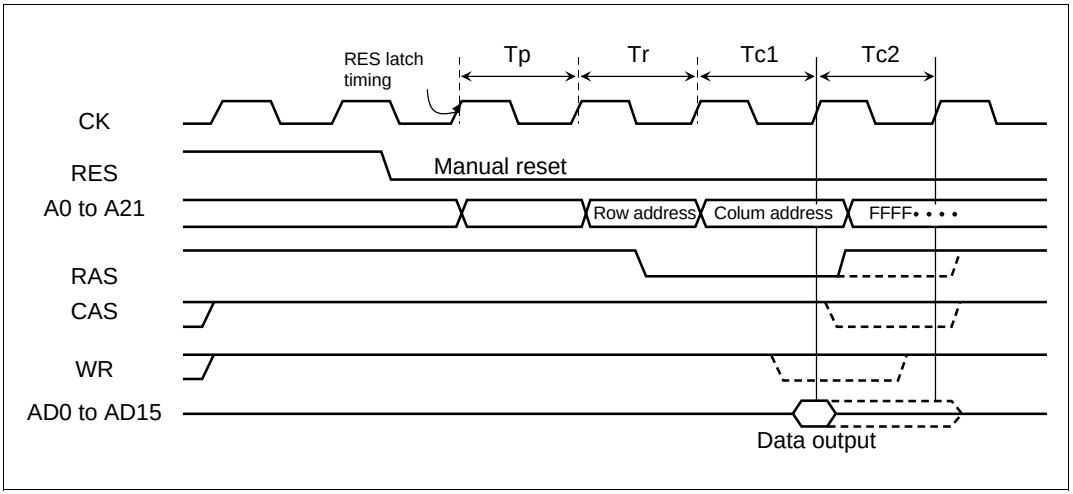



**Figure 8.38 Long - pitch Mode Write (1)**

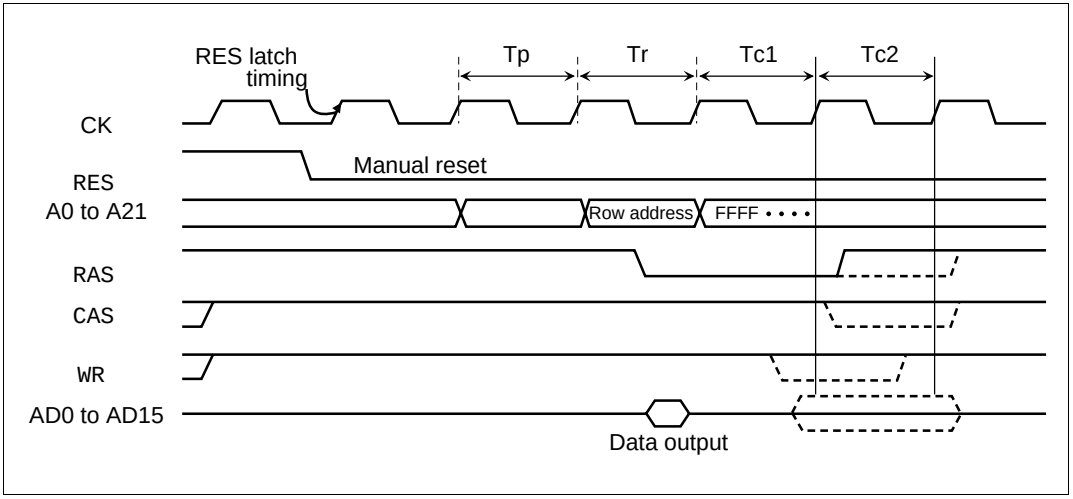



**Figure 8.39 Long - pitch Mode Write (2)**

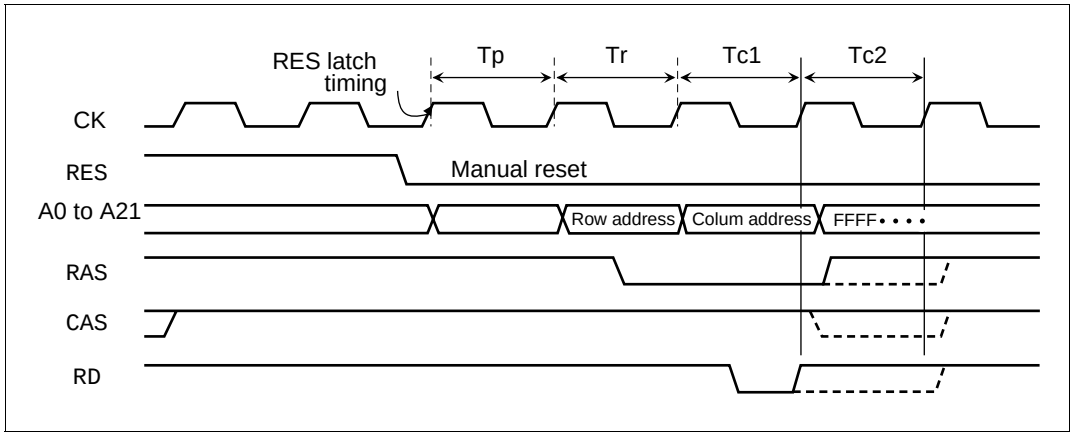



**Figure 8.40 Long - pitch Mode Read (1)**

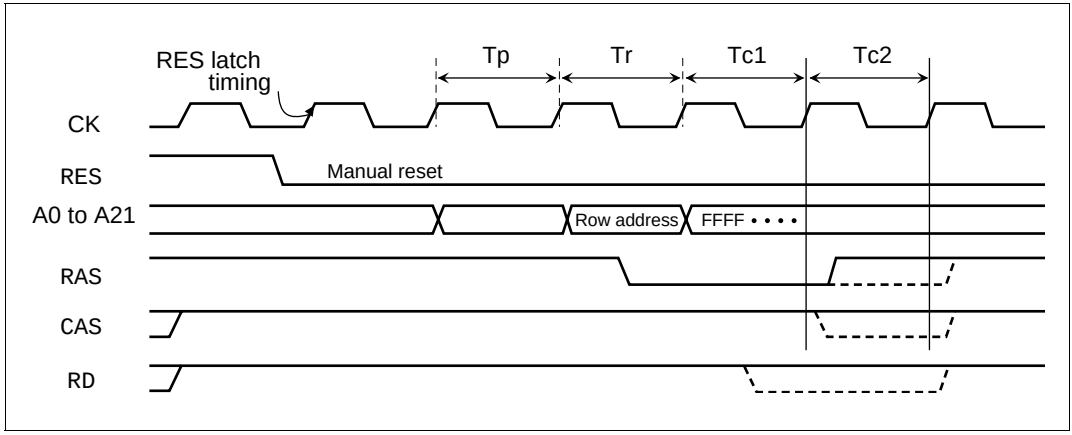



**Figure 8.41 Long - pitch Mode Read (2)**

For the signal output shown by solid lines, DRAM data may not be held. Therefore, when DRAM data must be held after reset, take one of the coutermeasures described as follows.

1. When resetting manually, do this in watchdog timer (WDT) condition.

2. Even if the Low width of $\overline{\text{RAS}}$ becomes as short as 1.5 tcyc as shown above, use with a frequency that satisfies the DRAM standard (tRAS).

3. Even in case the Low width of $\overline{\text{RAS}}$ has become 1.5 tcyc, proceed by using the external circuit so that a $\overline{\text{RAS}}$ signal with a Low width of 2.5 tcyc is input in the DRAM (in case the Low width of $\overline{\text{RAS}}$ is higher than 2.5 tcyc, operate so that the current waveform is input in the DRAM).

The countermeasures are not required when DRAM data is initialized or loaded again after manual reset.

### 8.11.2 Usage Notes on Parity Data Pins DPH and DPL

The following specifies the setup time tDS of the parity dada DPH and DPL to $\overline{CAS}$ signal rising when the parity dada DPH and DPL are written to DRAM in long-pitch mode (early write).

**Table 8.12 Setup Time of Parity Data DPH and DPL**

| Item | Symbol | min |
|---|---|---|
| Data setup time to CAS (for only DPH and DPL in long-pitch mode) | tDS | –5 ns |

Therefore, when writing parity data DPH and DPL to the DRAM in long-pitch mode, delay the $\overline{WRH}$ and $\overline{WRL}$ signals of this LSI and write with delayed writing.

Nomal dada is also delayed-written, causing no problems.

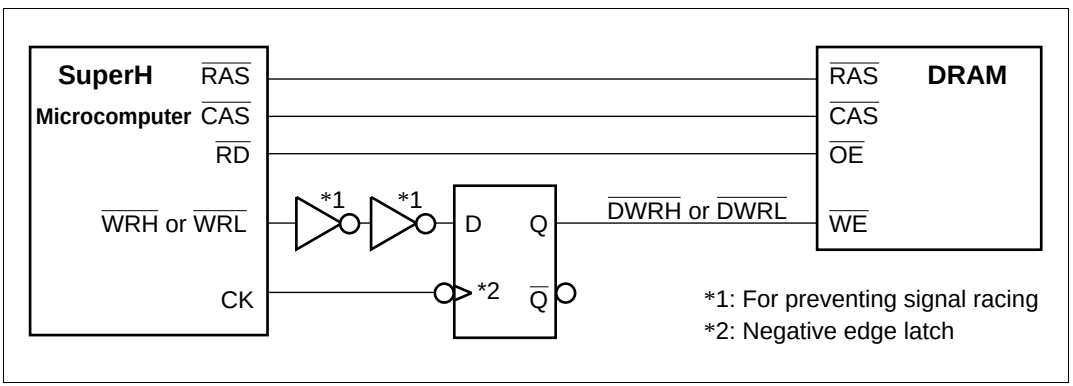



**Figure 8.42 Delayed-Write Control Circuit**

### 8.11.3 Maximum Number of States from $\overline{BREQ}$ Input to Bus Release

The maximum number of states from $\overline{BREQ}$ input to bus release is:

Maximum number of states for which bus is not released + approx. 4.5 states

Note: Breakdown of approx. 4.5 states:

| | |
|---|---|
| 1.5 states: | Until BACK output after end of bus cycle |
| 1 state (min.): | tBACD1 |
| 1 state (max.): | tBRQS |
| 1 state: | Sampling in 1 state before end of bus cycle |

$\overline{\text{BREQ}}$ is sampled one state before the bus cycle. If $\overline{\text{BREQ}}$ is input without satisfying tBRQS, the bus is released after executing cycle B following the end of bus cycle A, as shown in figure 8.43.

The maximum number of states from $\overline{\text{BREQ}}$ input to bus release are used when B is a cycle comprising the maximum number of states for which the bus is not released; the number of states is the maximum number of states for which bus is not released + approx. 4.5 states.

The maximum number of states for which the bus is not released requires careful investigation.

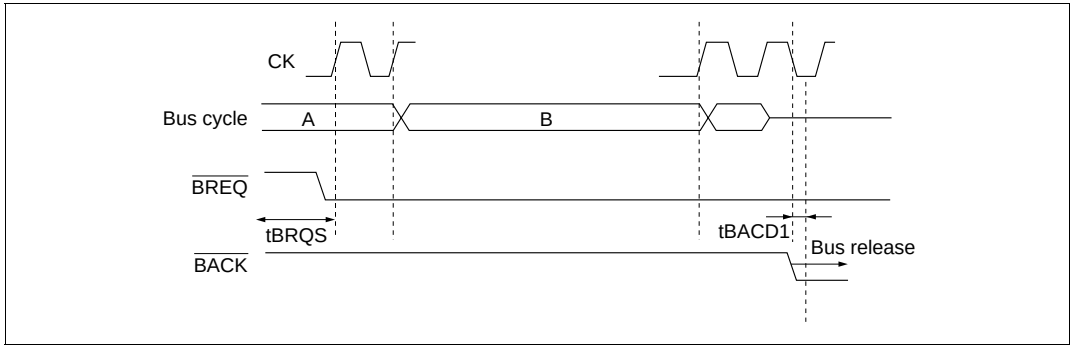



**Figure 8.43 When $\overline{\text{BREQ}}$ is Input without Satisfying tBRQS**

1. Cycles in which bus is not released

   (a) One bus cycle

   The bus is never released during one bus cycle. For example, in the case of a longword read (or write) in 8-bit ordinary space, one bus cycle consists of 4 memory accesses to 8-bit ordinary space, as shown in figure 8.44. The bus is not released between these accesses. Assuming one memory access to require 2 states, the bus is not released for a period of 8 states.

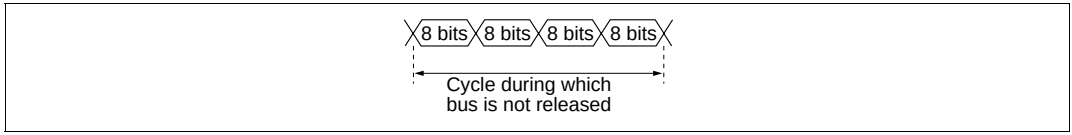



**Figure 8.44 One Bus Cycle**

   (b) TAS instruction read cycle and write cycle

   The bus is never released during a TAS instruction read cycle and write cycle (figure 8.45). The TAS instruction read cycle and write cycle should be regarded as one bus cycle during which the bus is not released.

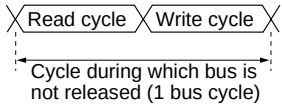



**Figure 8.45 TAS Instruction Read Cycle and Write Cycle**

(c) Refresh cycle + bus cycle

The bus is never released during a refresh cycle and the following bus cycle ((a) or (b) above)) (figure 8.46).

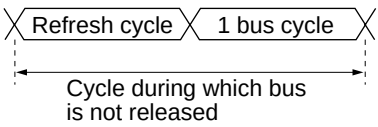



**Figure 8.46 Refresh Cycle and Following Bus Cycle**

2. Bus release procedure

The bus release procedure is shown in figure 8.47. Figure 8.47 shows the case where $\overline{\text{BREQ}}$ is input one state before the break between bus cycles so that tBRQS is satisfied. In the SH7020 and SH7021, the bus is released after the bus cycle in which $\overline{\text{BREQ}}$ is input (if $\overline{\text{BREQ}}$ is input between bus cycles, after the bus cycle starting next).

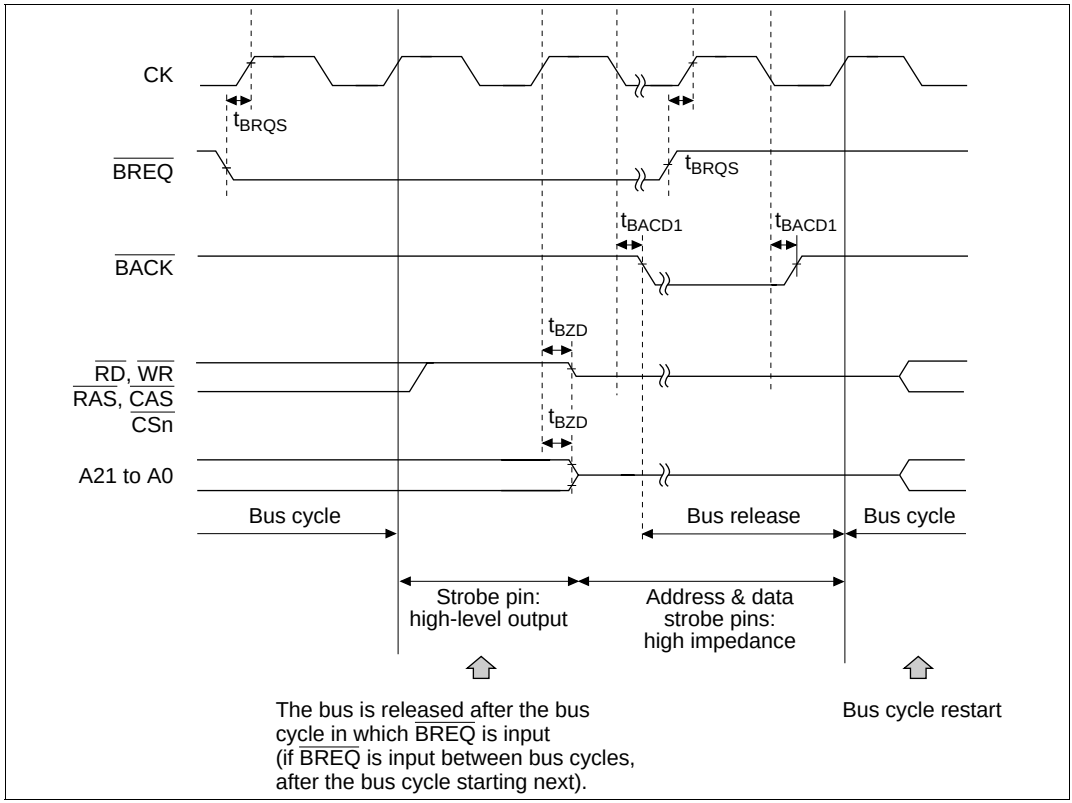



**Figure 8.47 Bus Release Procedure**

# Section 9 Direct Memory Access Controller (DMAC)

## 9.1 Overview

The SuperH microprocomputer chip includes a four-channel direct memory access controller (DMAC). The DMAC can be used in place of the CPU to perform high speed transfers between external devices that have DACK (transfer request acknowledge signal), external memory, memory-mapped external devices, on-chip memory and on-chip peripheral modules (excluding the DMAC itself). Using the DMAC reduces the burden on the CPU and increases overall operating efficiency.

### 9.1.1 Features

The DMAC has the following features.

- Four channels
- Four Gbytes of address space on the architecture
- Byte or word selectable data transfer unit
- 65536 transfers (maximum)
- Single address mode transfers (channels 0 and 1): Either the transfer source or transfer destination (peripheral device) is accessed by a DACK signal (selectable) while the other is accessed by address. 1 transfer unit of data is transferred in each bus cycle.

  Device combinations able to transfer:

  — External devices with DACK and memory-mapped external devices (including external memories)

  — External devices with DACK and memory-mapped external memories
- Dual address mode transfer: (channels 0–3): Both the transfer source and transfer destination are accessed by address. 1 transfer unit of data is transferred in 2 bus cycles.

  Device combinations able to transfer:

  — Two external memories

  — External memory and memory-mapped external devices

  — Two memory-mapped devices

  — External memory and on-chip memory

  — Memory-mapped external devices and on-chip peripheral module (excluding the DMAC itself)

  — External memory and on-chip memory

  — Memory-mapped external device and on-chip peripheral module (excluding the DMAC)

  — Two on-chip memories

  — On-chip memory and on-chip peripheral modules (excluding DMAC)

— Two on-chip peripheral modules (excluding DMAC)

- Transfer requests
  - — External request (From $\overline{\text{DREQ}}$ pins (channels 0 and 1 only). $\overline{\text{DREQ}}$ can be detected either by edge or by level)
  - — Requests from on-chip peripheral modules (serial communications interface (SCI), and 16-bit integrated-timer pulse unit (ITU))
  - — Auto-request (the transfer request is generated automatically within the DMAC)
- Selectable bus modes: Cycle-steal mode or burst mode
- Selectable channel priority levels: Fixed, round-robin, or external-pin round-robin modes
- CPU can be asked for interrupt when data transfer ends
- Maximum transfer rate
  - — 20 M words/s (320 MB/s)

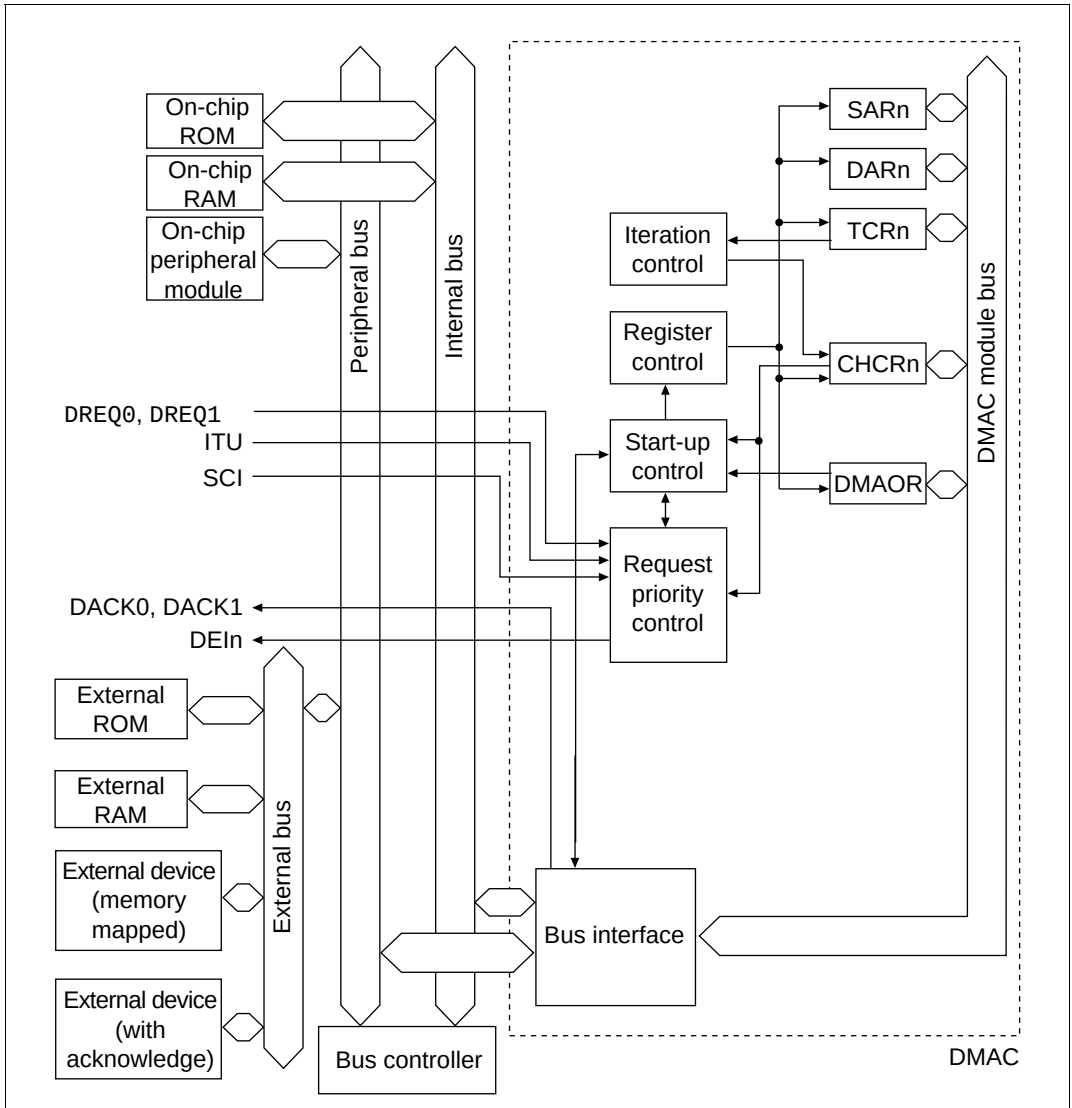
    For 5V and 20 MHz

    Bus mode: Burst mode

    Transmit size: Word

### 9.1.2 Block Diagram

Figure 9.1 is a block diagram of the DMAC.




DMAOR: DMA operation register
SARn: DMA source address register
DARn: DMA destination address register
TCRn: DMA transfer count register
CHCRn: DMA channel control register
DEIn: DMA transfer-end interrupt request to CPU.
n: 0–3

**Figure 9.1 DMAC Block Diagram**

### 9.1.3 Pin Configuration

Table 9.1 shows the DMAC pins.

**Table 9.1 Pin Configuration**

| Channel | Name | Symbol | I/O | Function |
|---|---|---|---|---|
| 0 | DMA transfer request | DREQ0 | I | DMA transfer request input from external device to channel 0 |
| | DMA transfer request acknowledge | DACK0 | O | DMA transfer request acknowledge output from channel 0 to external device |
| 1 | DMA transfer request | DREQ1 | I | DMA transfer request input from external device to channel 1 |
| | DMA transfer request acknowledge | DACK1 | O | DMA transfer request acknowledge output from channel 1 to external device |

### 9.1.4 Register Configuration

Table 9.2 summarizes the DMAC registers. DMAC has a total of 17 registers. Each channel has four control registers. One other control register is shared by all channels

**Table 9.2 DMAC Registers**

| Channel | Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|---|---|---|---|---|---|---|
| 0 | DMA source address register 0 | SAR0*3 | R/W | Undefined | H'5FFFF40 | 16, 32 |
| | DMA destination address register 0 | DAR0*3 | R/W | Undefined | H'5FFFF44 | 16, 32 |
| | DMA transfer count register 0 | TCR0*3 | R/W | Undefined | H'5FFFF4A | 16, 32 |
| | DMA channel control register 0 | CHCR0 | R/(W)*1 | H'0000 | H'5FFFF4E | 8, 16, 32 |
| 1 | DMA source address register 1 | SAR1*3 | R/W | Undefined | H'5FFFF50 | 16, 32 |
| | DMA destination address register 1 | DAR1*3 | R/W | Undefined | H'5FFFF54 | 16, 32 |
| | DMA transfer count register 1 | TCR1*3 | R/W | Undefined | H'5FFFF5A | 16, 32 |
| | DMA channel control register 1 | CHCR1 | R/(W)*1 | H'0000 | H'5FFFF5E | 8, 16, 32 |
| 2 | DMA source address register 2 | SAR2*3 | R/W | Undefined | H'5FFFF60 | 16, 32 |
| | DMA destination address register 2 | DAR2*3 | R/W | Undefined | H'5FFFF64 | 16, 32 |
| | DMA transfer count register 2 | TCR2*3 | R/W | Undefined | H'5FFFF6A | 16, 32 |
| | DMA channel control register 2 | CHCR2 | R/(W)*1 | H'0000 | H'5FFFF6E | 8, 16, 32 |
| 3 | DMA source address register 3 | SAR3*3 | R/W | Undefined | H'5FFFF70 | 16, 32 |
| | DMA destination address register 3 | DAR3*3 | R/W | Undefined | H'5FFFF74 | 16, 32 |
| | DMA transfer count register 3 | TCR3*3 | R/W | Undefined | H'5FFFF7A | 16, 32 |
| | DMA channel control register 3 | CHCR3 | R/(W)*1 | H'0000 | H'5FFFF7E | 8, 16, 32 |
| Shared | DMA operation register | DMAOR | R/(W)*2 | H'0000 | H'5FFFF48 | 8, 16, 32 |

Notes: 1. Write 0 alone in bit 1 of CHCR0–CHCR3 to clear flags.
2. Write 0 alone in bits 1 and 2 of the DMAOR to clear flags.
3. Access SAR0–SAR3, DAR0–DAR3, and TCR0–TCR3 by long word or word. If byte access is used when writing, the value of the register contents becomes undefined; if used when reading, the value read is undefined.

## 9.2 Register Descriptions

### 9.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3)

DMA source address registers 0–3 (SAR0–SAR3) are 32-bit read/write registers that specify the source address of a DMA transfer. During a DMA transfer, these registers indicate the next source address (in single-address mode, SAR is ignored in transfers from external devices with DACK to memory-mapped external devices or external memory).

The initial value after resets or in standby mode is undefined.

| 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 | … | 0 |
|---|---|---|---|---|---|---|
| Bit name: | | | | | … | |
| Initial value: | — | — | — | — | … | — |
| R/W: | R/W | R/W | R/W | R/W | … | R/W |

### 9.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)

DMA destination address registers 0–3 (DAR0–DAR3) are 32-bit read/write registers that specify the destination address of a DMA transfer. During a DMA transfer, these registers indicate the next destination address (in single-address mode, DAR is ignored in transfers from memory-mapped external devices or external memory to external devices with DACK). The initial value after resets or in standby mode is undefined.

| 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 | … | 0 |
|---|---|---|---|---|---|---|
| Bit name: | | | | | … | |
| Initial value: | — | — | — | — | … | — |
| R/W: | R/W | R/W | R/W | R/W | … | R/W |

### 9.2.3 DMA Transfer Count Registers 0–3 (TCR0–TCR3)

DMA transfer count registers 0-3 (TCR0–TCR3) are 16-bit read/write registers that specify the DMA transfer count (bytes or words). The number of transfers is 1 when the setting is H'0001, 65535 when the setting is H'FFFF and 65536 (the maximum) when H'0000 is set. During a DMA transfer, these registers indicate the remaining transfer count. The initial value after resets or in standby mode is undefined.

| 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 |

### 9.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)

DMA channel control registers 0–3 (CHCR0–CHCR3) are 16-bit read/write registers that control the DMA transfer mode. They also indicate DMA transfer status. They are initialized to H'0000 by a reset or standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | DM1 | DM0 | SM1 | SM0 | RS3 | RS2 | RS1 | RS0 |
| 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: | AM | AL | DS | TM | TS | IE | TE | DE |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/(W)*2 | R/(W)*2 | R/(W)*2 | R/W | R/W | R/W | R/(W)* | R/W |

Notes: 1. Write only 0 to clear the flag.
2. Writing is effective only for CHCR0 and CHCR1.

- Bits 15 and 14 (destination address mode bits 1, 0 (DM1 and DM0)): DM1 and DM0 select whether the DMA destination address is incremented, decremented, or left fixed (in the single address mode, DM1 and DM0 are ignored when transfers are made from memory-mapped external devices or external memory to external devices with DACK). DM1 and DM0 are initialized to 00 by resets or in standby mode.

| Bit 15: DM1 | Bit 14: DM0 | Description |
|---|---|---|
| 0 | 0 | Fixed destination address (initial value) |
| 0 | 1 | Destination address is incremented (+1 or +2 depending on if the transfer size is word or byte) |
| 1 | 0 | Destination address is decremented (–1 or –2 depending on if the transfer size is word or byte) |
| 1 | 1 | Reserved (illegal setting) |

- Bits 13 and 12 (source address mode bits 1, 0 (SM1 and SM0)): SM1 and SM0 select whether the DMA source address is incremented, decremented, or left fixed (in the single address mode, SM1 and SM0 are ignored when transfers are made from external devices with DACK to memory-mapped external devices or external memory). SM1 and SM0 are initialized to 00 by resets or in standby mode.

| Bit 13: SM1 | Bit 12: SM0 | Description |
|---|---|---|
| 0 | 0 | Fixed source address (initial value) |
| 0 | 1 | Source address is incremented (+1 or +2 depending on if the transfer size is word or byte) |
| 1 | 0 | Source address is decremented (–1 or –2 depending on if the transfer size is word or byte) |
| 1 | 1 | Reserved (illegal setting) |

- Bits 11–8 (resource select bits 3–0 (RS3-RS0)): RS3–RS0 specify which transfer requests will be sent to the DMAC. Do not change the transfer request source unless the DMA enable bit (DE) is 0. The RS3–RS0 bits are initialized to 0000 by resets or in standby mode.

| Bit 11: RS3 | Bit 10: RS2 | Bit 9: RS1 | Bit 8: RS0 | Description |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | DREQ (External request*1, dual address mode) (initial value) |
| 0 | 0 | 0 | 1 | Reserved (illegal setting) |
| 0 | 0 | 1 | 0 | DREQ (External request*1, single address mode*2) |
| 0 | 0 | 1 | 1 | DREQ (External request*1, single address mode*3) |
| 0 | 1 | 0 | 0 | RXI0 (On-chip serial communication interface 0 receive data full interrupt transfer request)*4 |
| 0 | 1 | 0 | 1 | TXI0 (On-chip serial communication interface 0 transmit data empty interrupt transfer request)*4 |
| 0 | 1 | 1 | 0 | RXI1 (On-chip serial communication interface 1 receive data full interrupt transfer request)*4 |
| 0 | 1 | 1 | 1 | TXI1 (On-chip serial communication interface 1 transmit data empty interrupt transfer request)*4 |
| 1 | 0 | 0 | 0 | IMIA0 (On-chip ITU0 input capture/compare-match A interrupt transfer request)*4 |
| 1 | 0 | 0 | 1 | IMIA1 (On-chip ITU1 input capture/compare-match A interrupt transfer request)*4 |
| 1 | 0 | 1 | 0 | IMIA2 (On-chip ITU2 input capture/compare-match A interrupt transfer request)*4 |
| 1 | 0 | 1 | 1 | IMIA3 (On-chip ITU3 input capture/compare-match A interrupt transfer request)*4 |
| 1 | 1 | 0 | 0 | Auto-request (Transfer requests automatically generated within DMAC)*4 |
| 1 | 1 | 0 | 1 | Reserved (illegal setting) |
| 1 | 1 | 1 | 0 | Reserved (illegal setting) |
| 1 | 1 | 1 | 1 | Reserved (illegal setting) |

SCI0, SCI1: Serial communications interface channels 0 and 1.

ITU0–ITU3: Channels 0–3 of the 16-bit integrated-timer pulse unit.

Notes: 1. These bits are valid only in channels 0 and 1. None of these request sources can be selected in channels 2 and 3.

2. Transfer from memory-mapped external device or external memory to external device with DACK.

3. Transfer from external device with DACK to memory-mapped external device or external memory.

4. Dual address mode.

- Bit 7 (acknowledge mode bit (AM)): In the dual address mode, AM selects whether the DACK signal is output during the data read cycle or write cycle. This bit is valid only in channels 0 and 1. The AM bit is initialized to 0 by resets or in standby mode. The AM bit is not valid in single address mode.

| Bit 7: AM | Description |
|---|---|
| 0 | DACK is output in read cycle (initial value) |
| 1 | DACK is output in write cycle |

- Bit 6 (acknowledge Level Bit (AL)): AL selects active high signal or active low signal for the DACK signal. This bit is valid only in channels 0 and 1. The AL bit is initialized to 0 by resets or in standby mode.

| Bit 6: AL | Description |
|---|---|
| 0 | DACK is active high (initial value) |
| 1 | DACK is active low |

- Bit 5 ($\overline{\text{DREQ}}$ select bit (DS)): DS selects the $\overline{\text{DREQ}}$ input detection method used. This bit is valid only in channels 0 and 1. The DS bit is initialized to 0 by resets or in standby mode.

| Bit 5: DS | Description |
|---|---|
| 0 | $\overline{\text{DREQ}}$ detected by low level (initial value) |
| 1 | $\overline{\text{DREQ}}$ detected by falling edge |

- Bit 4 (transfer bus mode bit (TM)): TM selects the bus mode for DMA transfers. The TM bit is initialized to 0 by resets or in standby mode. When the source of the transfer request is an on-chip peripheral module, see table 9.4, Selecting On-Chip Peripheral Module Request Modes with the RS Bit.

| Bit 4: TM | Description |
|---|---|
| 0 | Cycle-steal mode (initial value) |
| 1 | Burst mode |

- Bit 3 (transfer size bit (TS)): TS selects the transfer unit size. If the on-chip peripheral module that is the source or destination of the transfer can only be accessed in bytes, byte must be selected in this bit. The TS bit is initialized to 0 by resets or in standby mode.

| Bit 3: TS | Description |
|---|---|
| 0 | Byte (8 bits) (initial value) |
| 1 | Word (16 bits) |

- Bit 2 (interrupt enable bit (IE)): IE determines whether or not to request a CPU interrupt at the end of a DMA transfer. When the IE bit is set to 1, an interrupt (DEI) is requested from the CPU when the TE bit is set. The IE bit is initialized to 0 by resets or in standby mode.

| Bit 2: IE | Description |
|---|---|
| 0 | Interrupt request disabled (initial value) |
| 1 | Interrupt requeste enabled |

- Bit 1 (transfer end flag bit (TE)): TE indicates that the transfer has ended. When a DMA transfer ends normally and the value in the DMA transfer count register (TCR) becomes 0, the TE bit is set to 1. This flag is not set if the transfer ends because of an NMI interrupt or address error, or because the DE bit or the DME bit of the DMA operation register (DMAOR) was cleared. To clear the TE bit, read 1 from it and then write 0.

  When this flag is set, setting the DE bit to 1 does not enable a DMA transfer. The TE bit is initialized to 0 by resets or in standby mode.

| Bit 1: TE | Description |
|---|---|
| 0 | DMA has not ended or was aborted (initial value)<br>To clear TE, the CPU must read TE after it has been set to 1, then write a 0 in this bit |
| 1 | DMA has ended normally |

- Bit 0 (DMA enable bit (DE)): DE enables or disables DMA transfers. In the auto-request mode, the transfer starts when this bit or the DME bit of the DMAOR is set to 1. The TE bit and the NMIF and AE bits of the DMAOR must be all cleared to 0. In external request mode or on-chip peripheral module request mode, the transfer begins when the DMA transfer request is received from said device or on-chip peripheral module, provided this bit and the DME bit are set to 1. As with the auto request mode, the TE bit and the NMIF and AE bits of the DMAOR must be all cleared to 0. The transfer can be stopped by clearing this bit to 0.

  The DE bit is initialized to 0 by resets or in standby mode.

| Bit 0: DE | Description |
|---|---|
| 0 | DMA transfer disabled (initial value) |
| 1 | DMA transfer enabled |

### 9.2.5 DMA Operation Register (DMAOR)

The DMA operation register (DMAOR) is a 16-bit read/write register that controls the DMA transfer mode. It also indicates the DMA transfer status. It is initialized to H'0000 by a reset or the standby mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | —— | — | — | — | — | — | PR1 | PR0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R/W | R/W |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | AE | NMIF | DME |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/(W)* | R/(W)* | R |

Note: Write only 0 to clear the flag.

- Bits 15–10 (reserved): These bits always read 0. The write value should always be 0.

- Bits 9 and 8 (priority mode bits 1 and 0 (PR1 and PR0)): PR1 and PR0 select the priority level between channels when there are transfer requests for multiple channels simultaneously.

| Bit 9: PR1 | Bit 8: PR0 | Description |
|---|---|---|
| 0 | 0 | Fixed priority order (Ch. 0 > Ch. 3 > Ch. 2 > Ch. 1) (initial value) |
| 0 | 1 | Fixed priority order (Ch. 1 > Ch. 3 > Ch. 2 > Ch. 0) |
| 1 | 0 | Round-robin mode priority order (the priority order immediately after a reset is Ch. 0 > Ch. 3 > Ch. 2 > Ch. 1) |
| 1 | 1 | External-pin round-robin mode priority order (the priority order immediately after a reset is Ch. 3 > Ch. 2 > Ch. 1 > Ch. 0) |

- Bits 7–3 (reserved): These bits always read 0. The write value should always be 0.
- Bit 2 (address error flag bit (AE)): AE indicates that an address error occurred in the DMAC. When this flag is set to 1, the channel cannot be enabled even if the DE bit in the DMA channel control register (CHCR) and the DME bit are set to 1. To clear the AE bit, read 1 from it and then write 0. It is initialized to 0 by a reset or the standby mode.

| Bit 2: AE | Description |
|---|---|
| 0 | No DMAC address error (initial value)<br>To clear the AE bit, read 1 from it and then write 0. |
| 1 | Address error by DMAC |

- Bit 1 (NMI Flag Bit (NMIF)): NMIF indicates that an NMI interrupt occurred. When this flag is set to 1, the channel cannot be enabled even if the DE bit in the CHCR and the DME bit are set to 1. To clear the NMIF bit, read 1 from it and then write 0. It is initialized to 0 by a reset or the standby mode.

| Bit 1: NMIF | Description |
|---|---|
| 0 | No NMI interrupt (initial value)<br>To clear the NMIF bit, read 1 from it and then write 0. |
| 1 | NMI has occurred |

- Bit 0 (DMA master enable bit (DME)): DME enables or disables DMA transfers on all channels. A channel becomes enabled for a DMA transfer when the DE bit in each DMA's CHCR and the DME bit are set to 1. For this to be effective, however, the TE bit of each CHCR and the NMIF and AE bits must all be 0. When the DME bit is cleared, all channel DMA transfers are aborted.

| Bit 0: DME | Description |
|---|---|
| 0 | Disable DMA transfers on all channels (initial value) |
| 1 | Enable DMA transfers on all channels |

## 9.3 Operation

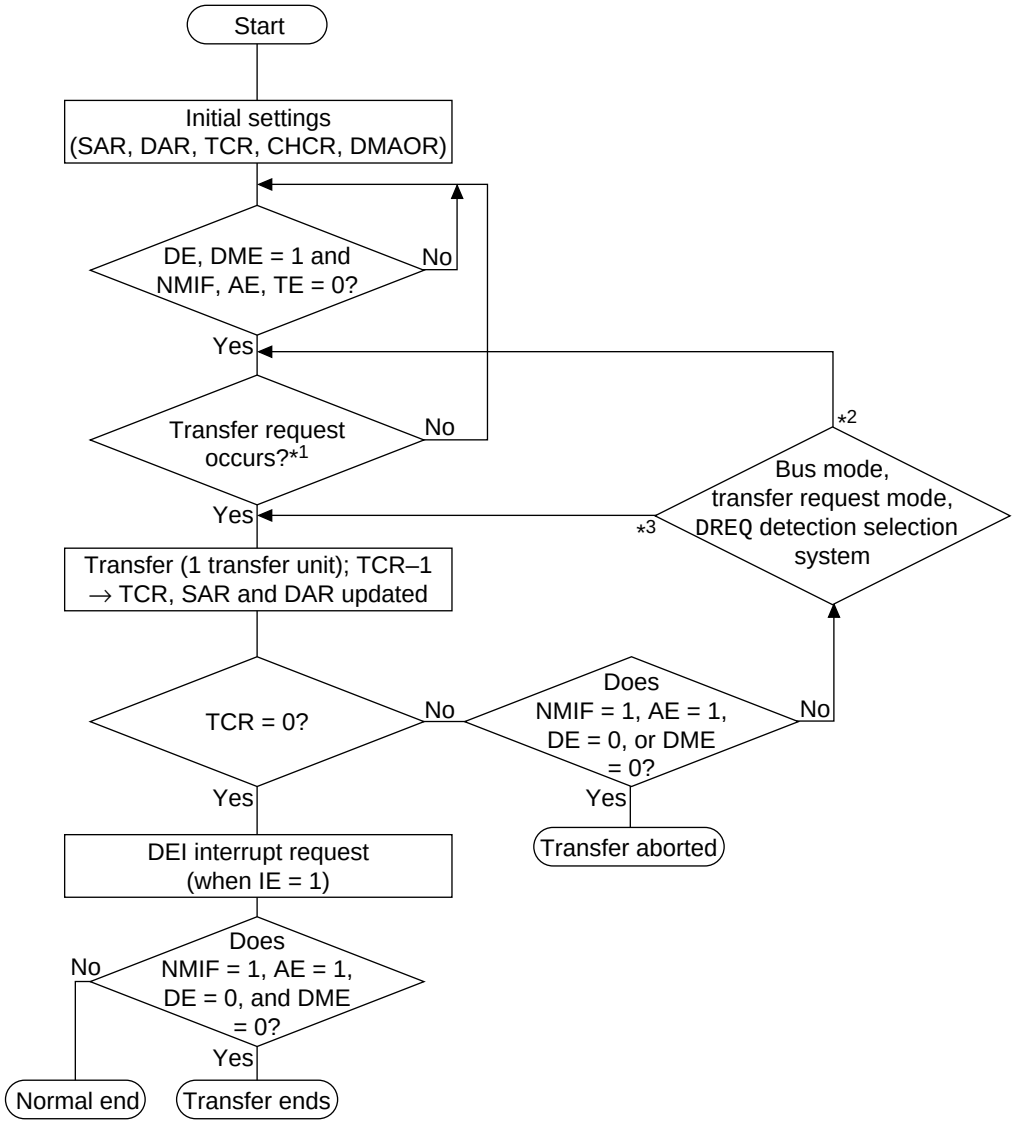
When there is a DMA transfer request, the DMAC starts the transfer according to the predetermined channel priority order; when the transfer end conditions are satisfied, it ends the transfer. Transfers can be requested in three modes: auto-request, external request, and on-chip module request. Transfer can be in either the single address mode or the dual address mode. The bus mode can be either burst or cycle steal

### 9.3.1 DMA Transfer Flow

After the DMA source address registers (SAR), DMA destination address registers (DAR), DMA transfer count registers (TCR), DMA channel control registers (CHCR), and DMA operation register (DMAOR) are set, the DMAC transfers data according to the following procedure:

1. Checks to see if transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0)
2. When a transfer request comes and transfer is enabled, the DMAC transfers 1 transfer unit of data (for an auto-request, the transfer begins automatically when the DE bit and DME bit are set to 1. The TCR value will be decremented by 1). The actual transfer flows vary by address mode and bus mode.
3. When the specified number of transfer have been completed (when TCR reaches 0), the transfer ends normally. If the IE bit of the CHCR is set to 1 at this time, a DEI interrupt is sent to the CPU.
4. When an address error occurs in the DMAC or an NMI interrupt is generated, the transfer is aborted. Transfers are also aborted when the DE bit of the CHCR or the DME bit of the DMAOR are changed to 0.

Figure 9.2 is a flowchart of this procedure.




Notes: 1. In auto-request mode, transfer begins when NMIF, AE and TE are all and the DE and DME bits are set to 1.

2. DREQ = level detection in the burst mode (external request), or cycle steal mode.

3. DREQ = edge detection in the burst mode (external request), or auto request mode in burst mode.

**Figure 9.2 DMA Transfer Flowchart**

### 9.3.2 DMA Transfer Requests

DMA transfer requests are basically generated in either the data transfer source or destination, but they can also be generated by devices and on-chip peripheral modules that are neither the source nor the destination. Transfers can be requested in three modes: auto-request, external request, and on-chip module request. The request mode is selected in the RS3–RS0 bits of the DMA channel control registers 0–3 (CHCR0–CHCR3).

Auto-Request Mode: When there is no transfer request signal from an external source, as in a memory-to-memory transfer or a transfer between memory and an on-chip peripheral module unable to request a transfer, the auto-request mode allows the DMAC to automatically generate a transfer request signal internally. When the DE bits of CHCR0–CHCR3 and the DME bit of the DMAOR are set to 1, the transfer begins (so long as the TE bits of CHCR0–CHCR3 and the NMIF and AE bits of DMAOR are all 0).

External Request Mode: In this mode a transfer is performed at the request signal ($\overline{\text{DREQ}}$) of an external device. Choose one of the modes shown in table 9.3 according to the application system. When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), a transfer is performed upon a request at the $\overline{\text{DREQ}}$ input. Choose to detect $\overline{\text{DREQ}}$ by either the falling edge or low level of the signal input with the DS bit of CHCR0–CHCR3 (DS = 0 is level detection, DS = 1 is edge detection). The source of the transfer request does not have to be the data transfer source or destination.

**Table 9.3 Selecting External Request Modes with the RS Bits**

| RS3 | RS2 | RS1 | RS0 | Address Mode | Source | Destination |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Dual address mode | Any* | Any* |
| 0 | 0 | 1 | 0 | Single address mode | External memory or memory-mapped external device | External device with DACK |
| 0 | 0 | 1 | 1 | Single address mode | External device with DACK | External memory or memory-mapped external device |

Note: External memory, memory-mapped external device, on-chip memory, on-chip peripheral module (excluding DMAC)

**On-Chip Module Request:** In this mode a transfer is performed at the transfer request signal (interrupt request signal) of an on-chip module. The transfer request signals include the receive data full interrupt (RXI) of the serial communication interface (SCI), the transmit data empty interrupt (TXI) of the SCI, the input capture A/compare match A interrupt request (IMIA) of the 16-bit integrated-pulse timer (ITU), (table 9.4). When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), a transfer is performed upon the input of a transfer request signal. The source of the transfer request does not have to be the data transfer

source or destination. When RXI is set as the transfer request, however, the transfer source must be the SCI's receive data register (RDR). Likewise, when TXI is set as the transfer request, the transfer source must be the SCI's transmit data register (TDR).

**Table 9.4 Selecting On-Chip Peripheral Module Request Modes with the RS Bit**

| RS3 | RS2 | RS1 | RS0 | DMA Transfer Request Source | DMA Transfer Request Signal | Source | Destination | Bus Mode |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | SCI0 receiver | RXI0 (SCI0 receive data full interrupt transfer request) | RDR0 | Any* | Cycle steal |
| 0 | 1 | 0 | 1 | SCI0 transmitter | TXI0 (SCI0 transmit data empty interrupt transfer request) | Any | TDR0 | Cycle steal |
| 0 | 1 | 1 | 0 | SCI1 receiver | RXI1 (SCI1 receive data full interrupt transfer request) | RDR1 | Any* | Cycle steal |
| 0 | 1 | 1 | 1 | SCI1 transmitter | TXI1 (SCI1 transmit data empty interrupt transfer request) | Any* | TDR1 | Cycle steal |
| 1 | 0 | 0 | 0 | ITU0 | IMIA0 (ITU0 input capture A/ compare-match A) | Any* | Any* | Burst/Cycle steal |
| 1 | 0 | 0 | 1 | ITU1 | IMIA1 (ITU1 input capture A/ compare-match A) | Any* | Any* | Burst/Cycle steal |
| 1 | 0 | 1 | 0 | ITU2 | IMIA2 (ITU2 input capture A/ compare-match A) | Any* | Any* | Burst/Cycle steal |
| 1 | 0 | 1 | 1 | ITU3 | IMIA3 (ITU3 input capture A/ compare-match A) | Any* | Any* | Burst/Cycle steal |

SCI0, SCI1: Serial communications interface channels 0 and 1

ITU0–ITU3: Channels 0–3 of the 16-bit integrated-timer pulse unit.

RDR0, RDR1: Receive data registers 0, 1 of SCI

TDR0, TDR1: Transmit data registers 0, 1 of SCI

Note: External memory, memory-mapped external device, on-chip memory, on-chip peripheral module (excluding DMAC)

When outputting transfer requests from on-chip peripheral modules, the appropriate interrupt enable bits must be set to output the interrupt signals. Note that transfer request signals from on-chip peripheral modules (interrupt request signals) are sent not just to the DMAC but to the CPU as well. When an on-chip peripheral module is specified as the transfer request source, set the priority level values in the interrupt priority level registers (IPRC–IPRE) of the interrupt controller (INTC) at or below the levels set in the I3–I0 bits of the CPU's status register (SR) so that the CPU does not acknowledge the interrupt request signal.

The DMA transfer request signals of table 9.4 are automatically withdrawn when the corresponding DMA transfer is performed. If the cycle steal mode is being employed, the DMA transfer request (interrupt request) will be cleared at the first transfer; if the burst mode is being used, it will be cleared at the last transfer.

### 9.3.3 Channel Priority

When the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority order. The three modes (fixed mode, round-robin mode, and external-pin round-robin mode) are selected by the priority bits PR1 and PR0 in the DMA operation register.

**Fixed Mode:** In these modes, the priority levels among the channels remain fixed. When PR1 and PR0 bits are set 00, the priority order, high to low, is Ch. 0 > Ch. 3 > Ch. 2 > Ch. 1. When PR1 and PR0 bits are set 01, the priority order, high to low, is Ch. 1 > Ch. 3 > Ch. 2 > Ch. 0.

**Round-Robin Mode:** Each time one word or byte is transferred on one channel, the priority order is rotated. The channel on which the transfer was just finished rotates to the bottom of the priority order. When necessary, the priority order of channels other than the one that just finished the transfer can also be shifted to keep the relationship between the channels from changing (figure 9.3). The priority order immediately after a reset is channel 0 > channel 3 > channel 2 > channel 1.

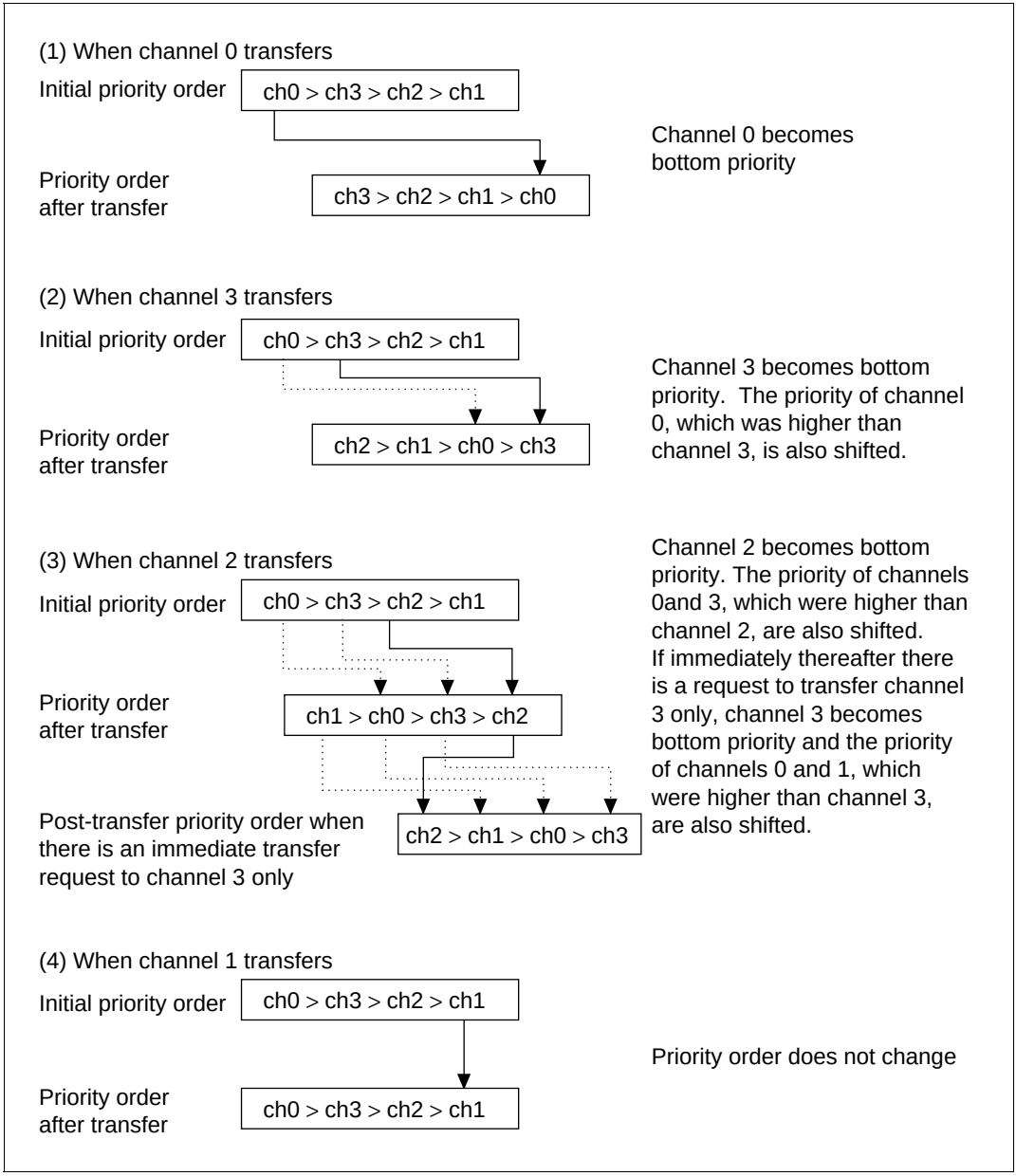



**Figure 9.3 Round-Robin Mode**

Figure 9.4 shows how the priority order changes when channel 0 and channel 1 transfers are requested simultaneously and a channel 3 transfer is requested during the channel 0 transfer. The DMAC operates as follows:

1. Transfer requests are generated simultaneously to channels 1 and 0.
2. Channel 0 has a higher priority, so the channel 0 transfer begins first (channel 1 waits for transfer).
3. A channel 3 transfer request occurs during the channel 0 transfer (channels 1 and 3 are both waiting)
4. When the channel 0 transfer ends, channel 0 becomes lowest priority.
5. At this point, channel 3 has a higher priority than channel 1, so the channel 3 transfer begins (channel 1 waits for transfer).
6. When the channel 3 transfer ends, channel 3 becomes lowest priority.
7. The channel 1 transfer begins.
8. When the channel 1 transfer ends, channels 1 and 2 shift downward in priority so that channel 1 becomes the lowest priority.

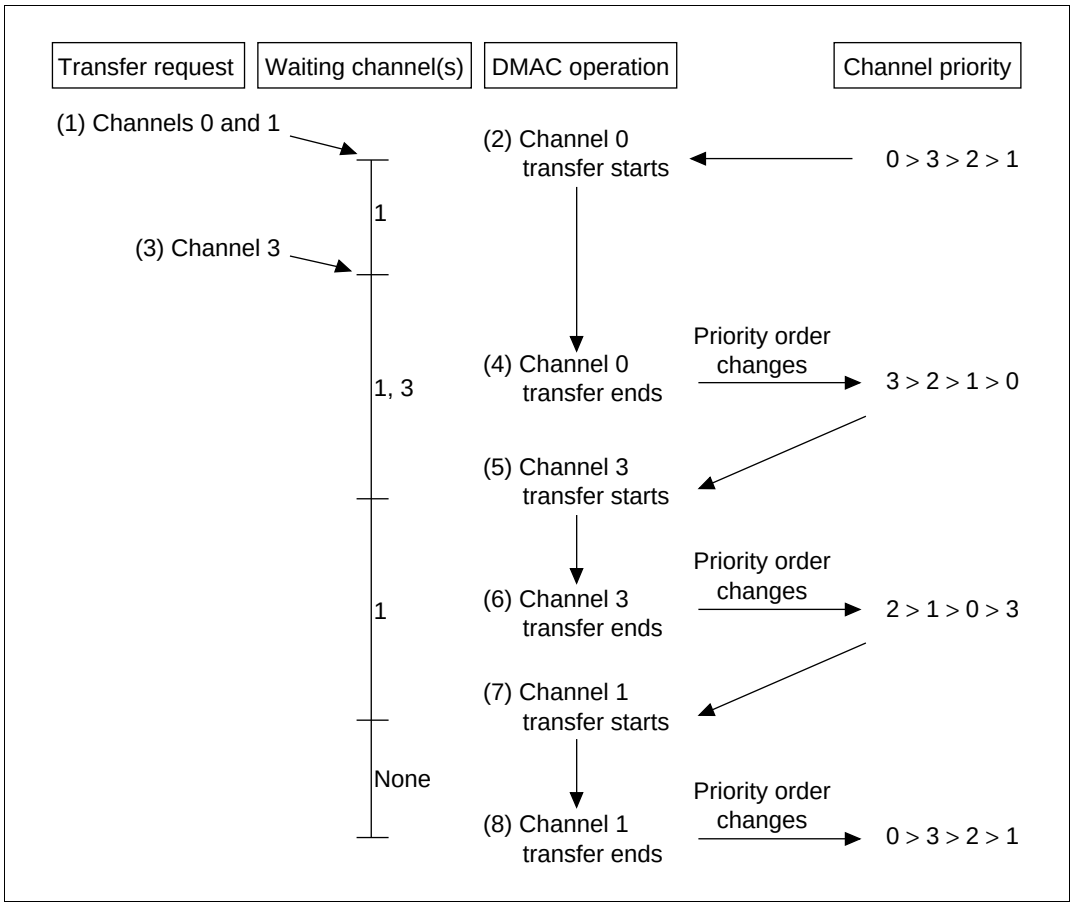



**Figure 9.4 Changes in Channel Priority in Round-Robin Mode**

**External-Pin Round-Robin Mode:** External-pin round-robin mode switches the priority levels of channel 0 and channel 1, which are the channels that can receive transfer requests from external pins $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$. The priority levels are changed after each (byte or word) transfer on channel 0 or channel 1 is completed. The channel which just finished the transfer rotates to the bottom of the priority order. The priority levels of channels 2 and 3 do not change. The initial priority order after a reset is channel 3 > channel 2 > channel 1 > channel 0.

Figure 9.5 shows how the priority order changes when channel 0 and channel 1 transfers are requested simultaneously and a channel 0 transfer is requested again after both channels finish their transfers. The DMAC operates as follows:

1. Transfer requests are generated simultaneously to channels 1 and 0.
2. Channel 1 has a higher priority, so the channel 1 transfer begins first (channel 0 waits for transfer).
3. When the channel 1 transfer ends, channel 1 becomes lowest priority.
4. The channel 0 transfer begins.
5. When the channel 0 transfer ends, channel 0 becomes lowest priority.
6. A channel 0 transfer request occurs again.
7. The channel 0 transfer begins.
8. When the channel 0 transfer ends, the priority order does not change, because channel 0 is already the lowest priority.

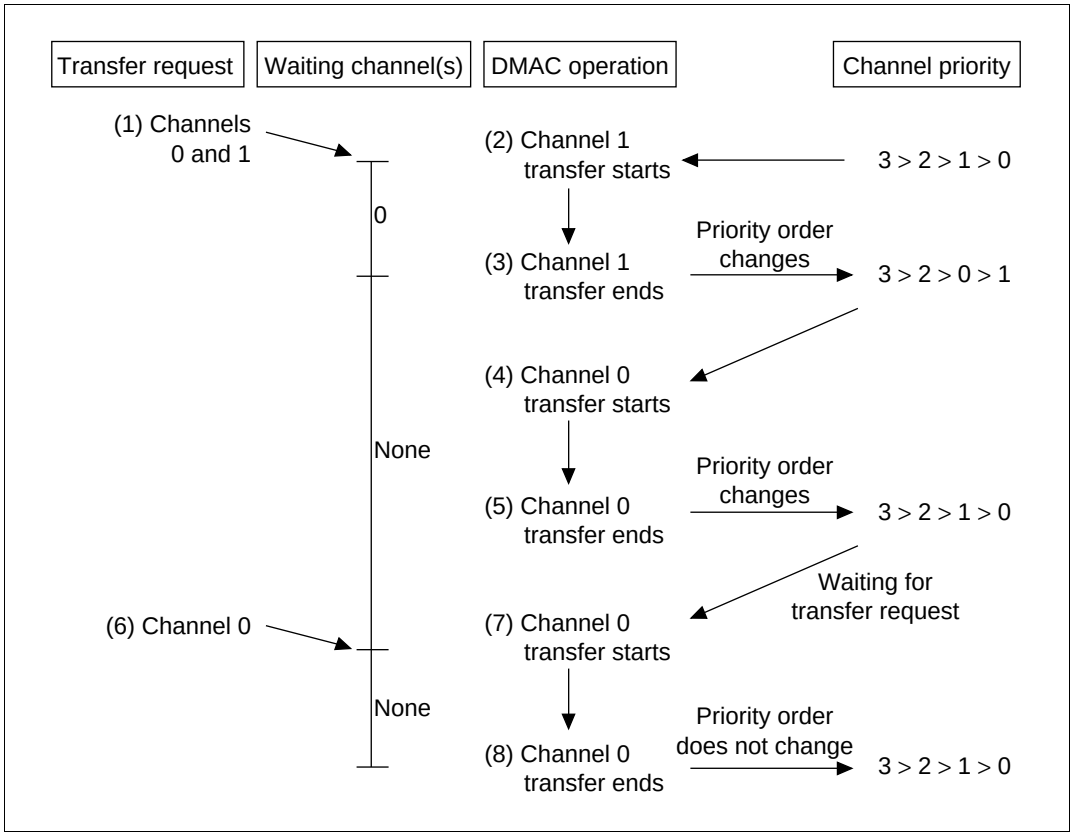



**Figure 9.5 Example of Changes in Priority in External-Pin Round-Robin Mode**

### 9.3.4 DMA Transfer Types

The DMAC supports the transfers shown in table 9.5. It can operate in the single address mode or dual address mode, which are defined by how many bus cycles the DMAC takes to access the transfer source and transfer destination. The actual transfer operation timing varies with the bus mode. The DMAC has two bus modes: cycle-steal mode and burst mode.

**Table 9.5 Supported DMA Transfers**

| | Destination | | | | |
|---|---|---|---|---|---|
| **Source** | **External Device with DACK** | **External Memory** | **Memory-Mapped External Device** | **On-Chip Memory** | **On-Chip Peripheral Module** |
| External device with DACK | Not available | Single | Single | Not available | Not available |
| External memory | Single | Dual | Dual | Dual | Dual |
| Memory-mapped external device | Single | Dual | Dual | Dual | Dual |
| On-chip memory | Not available | Dual | Dual | Dual | Dual |
| On-chip peripheral module | Not available | Dual | Dual | Dual | Dual |

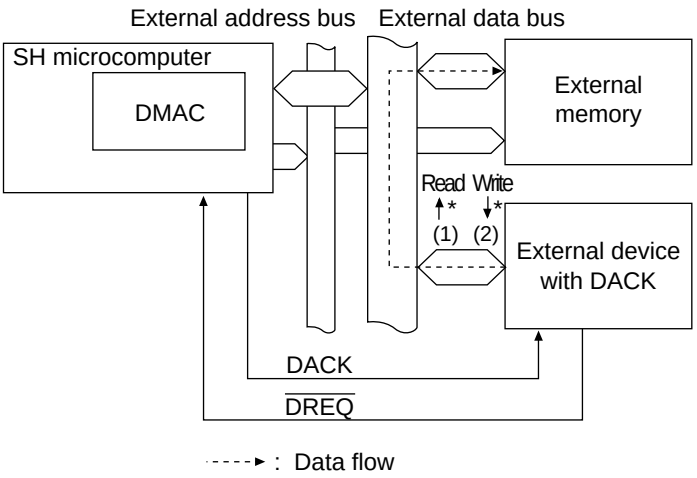
Single: Single address mode

Dual: Dual address mode

**Address Modes:**

- Single Address Mode

In the single address mode, both the transfer source and destination are external; one (selectable) is accessed by a DACK signal while the other is accessed by an address. In this mode, the DMAC performs the DMA transfer in 1 bus cycle by simultaneously outputting a transfer request acknowledge DACK signal to one external device to access it while outputting an address to the other end of the transfer. Figure 9.6 shows an example of a transfer between an external memory and an external device with DACK in which the external device outputs data to the data bus while that data is written in external memory in the same bus cycle.




Note: The read/write direction is decided by the RS3-RS0 bits of the CHCRn registers. If RS3-RS0=0010, the direction is shown as case 1 (circled number above); if RS3-RS0=0010, the direction is shown as case 2. Also, DACK output (when writing) indicates case 2.

**Figure 9.6 Data Flow in Single Address Mode**

Two types of transfers are possible in the single address mode: 1) transfers between external devices with DACK and memory-mapped external devices, and 2) transfers between external devices with DACK and external memory. The only transfer requests for either of these is the external request ($\overline{\text{DREQ}}$). Figure 9.7 shows the DMA transfer timing for the single address mode.

The DACK output when a transfer occurs from an external device with DACK to a memory-mapped external device is the write waveform. The DACK output when a transfer occurs from a memory-mapped external device to an external device with DACK is the read waveform. The setting of the acknowledge mode (AM) bits in the channel control registers (CHCR0, CHCR1) have no effect.

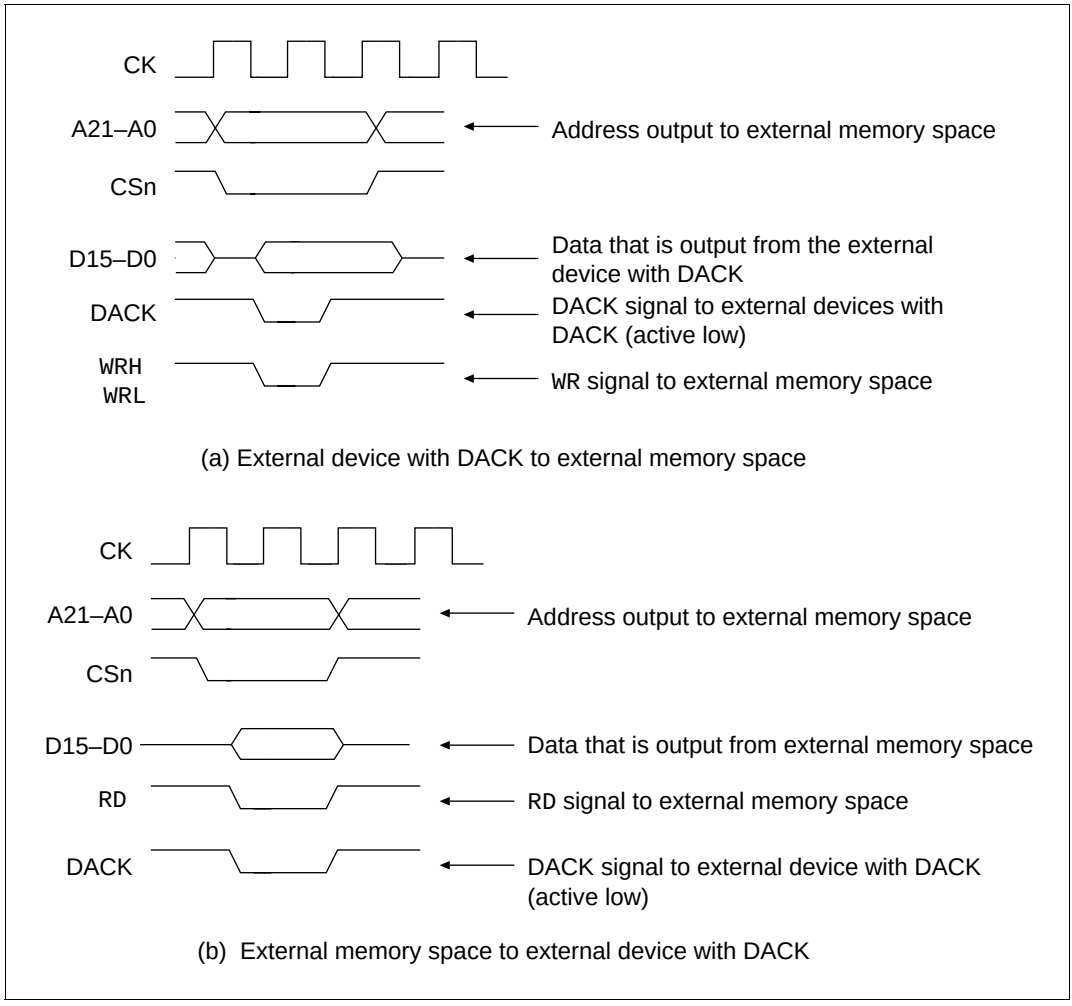



**Figure 9.7 Example of DMA Transfer Timing in the Single Address Mode**

- Dual Address Mode

In the dual address mode, both the transfer source and destination are accessed (selectable) by an address. The source and destination can be located externally or internally. The source is accessed in the read cycle and the destination in the write cycle, so the transfer is performed in two separate bus cycles. The transfer data is temporarily stored in the DMAC. Figure 9.8 shows an example of a transfer between two external memories in which data is read from one memory in the read cycle and written to the other memory in the following write cycle.

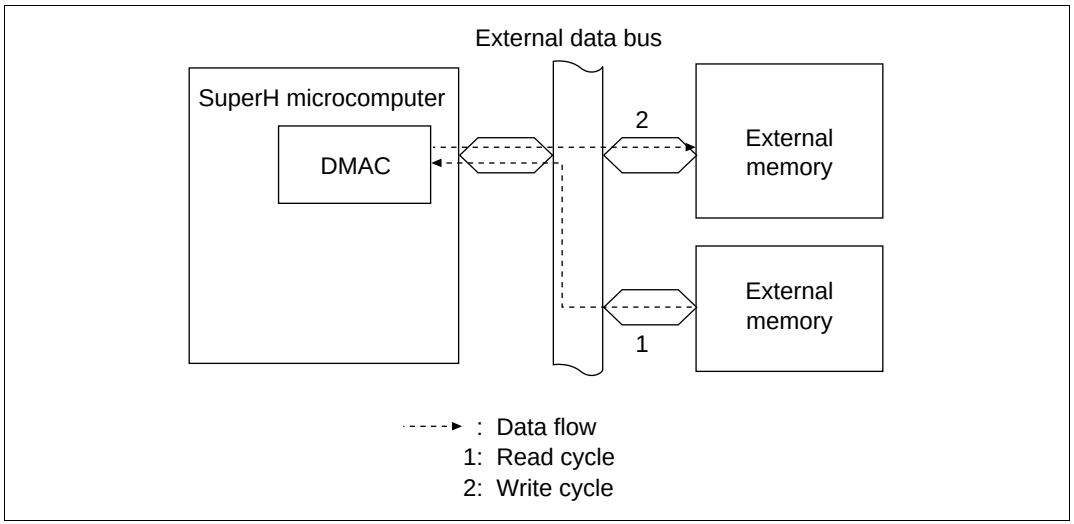



**Figure 9.8 Data Flow in Dual Address Mode**

In the dual address mode transfers, external memory, memory-mapped external devices, on-chip memory and on-chip peripheral modules can be mixed without restriction. Specifically, this enables the following transfer types:

1. External memory and external memory transfer
2. External memory and memory-mapped external devices transfer
3. Memory-mapped external devices and memory-mapped external devices transfer
4. External memory and on-chip memory transfer
5. External memory and on-chip peripheral modules (excluding the DMAC) transfer
6. Memory-mapped external devices and on-chip memory transfer
7. Memory-mapped external devices and on-chip peripheral modules (excluding the DMAC) transfer
8. On-chip memory and on-chip memory transfer
9. On-chip memory and on-chip peripheral modules (excluding the DMAC) transfer
10. On-chip peripheral modules (excluding the DMAC) and on-chip peripheral modules (excluding the DMAC) transfer

Transfer requests can be auto requests, external requests, or on-chip peripheral module requests. When the transfer request source is either the SCI or A/D converter, however, either the data destination or source must be the SCI or A/D converter (figure 9.4), In dual address mode, DACK is output in read or write cycles to onchip memory or onchip peripheral modules. The CHCR controls the cycle of DACK output.

Figure 9.9 shows the DMA transfer timing in the dual address mode.

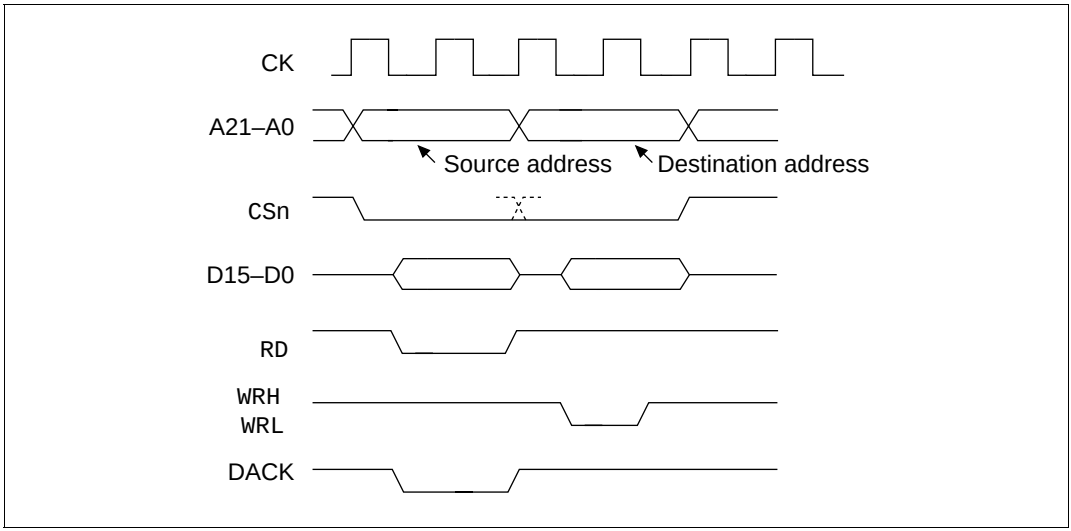



**Figure 9.9 DMA Transfer Timing in the Dual Address Mode (External memory space to external memory space transfer with DACK output in the read cycle)**

**Bus Modes:** There are two bus modes: cycle steal and burst. Select the mode in the TM bits of CHCR0–CHCR3.

- Cycle-Steal Mode

  In the cycle steal mode, the bus right is given to another bus master after a one-transfer-unit (word or byte) DMA transfer. When another transfer request occurs, the bus rights are obtained from the other bus master and a transfer is performed for one transfer unit. When that transfer ends, the bus right is passed to the other bus master. This is repeated until the transfer end conditions are satisfied.

  The cycle steal mode can be used with all categories of transfer destination, transfer source and transfer request. Figure 9.10 shows an example of DMA transfer timing in the cycle steal mode. Transfer conditions shown in the figure are:

  — Dual address mode
  — $\overline{\text{DREQ}}$ level detection

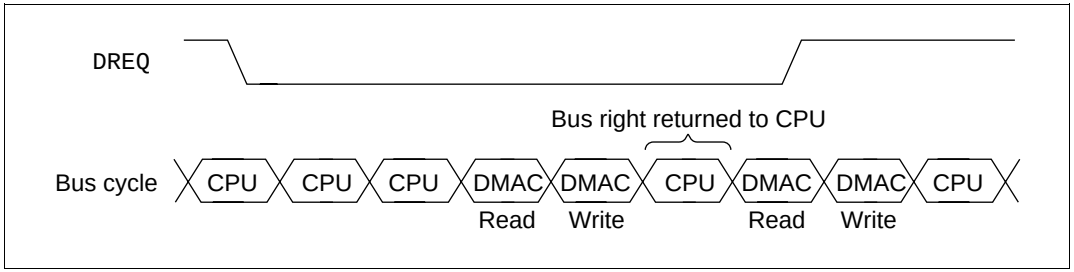



**Figure 9.10 Transfer Example in the Cycle-Steal Mode (Dual address mode, $\overline{\text{DREQ}}$ level detection)**

- Burst Mode

  Once the bus right is obtained, the transfer is performed continuously until the transfer end condition is satisfied. In the external request mode with low level detection of the $\overline{\text{DREQ}}$ pin, however, when the DREQ pin is driven high, the bus passes to the other bus master after the bus cycle of the DMAC that currently has an acknowledged request ends, even if the transfer end conditions have not been satisfied.

  The burst mode cannot be used when the serial communications interface (SCI) is the transfer request source. Figure 9.11 shows an example of DMA transfer timing in the burst mode. The transfer conditions shown in the figure are:

  — Single address mode
  — $\overline{\text{DREQ}}$ level detection

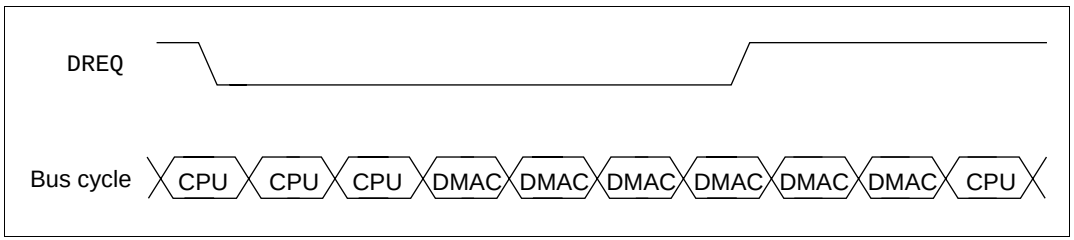



**Figure 9.11 Transfer Example in the Burst Mode (Single address mode, DREQ level detection)**

**Relationship between Request Modes and Bus Modes by DMA Transfer Category:** Table 9.6 shows the relationship between request modes and bus modes by DMA transfer category.

**Table 9.6 Relationship of Request Modes and Bus Modes by DMA Transfer Category**

| Address Mode | Transfer Category | Request Mode | Bus Mode | Transfer Size (bits) | Usable Channels |
|---|---|---|---|---|---|
| Single | External device with DACK and external memory | External | B/C | 8/16 | 0,1 |
| | External device with DACK and memory-mapped external device | External | B/C | 8/16 | 0, 1 |
| Dual | External memory and external memory | Everything*1 | B/C | 8/16 | 0–3*5 |
| | External memory and memory-mapped external device | Everything*1 | B/C | 8/16 | 0–3*5 |
| | Memory-mapped external device and memory-mapped external device | Everything*1 | B/C | 8/16 | 0–3*5 |
| | External memory and on-chip memory | Everything*1 | B/C | 8/16 | 0–3*5 |
| | External memory and on-chip peripheral module | Everything*2 | B/C*3 | 8/16*4 | 0–3*5 |
| | Memory-mapped external device and on-chip memory | Everything*1 | B/C | 8/16 | 0–3*5 |
| | Memory-mapped external device and on-chip peripheral module | Everything*2 | B/C*3 | 8/16*4 | 0–3*5 |
| | On-chip memory and on-chip memory | Everything*1 | B/C | 8/16 | 0–3*5 |
| | On-chip memory and on-chip peripheral module | Everything*2 | B/C*3 | 8/16*4 | 0–3*5 |
| | On-chip peripheral module and on-chip peripheral module | Everything*2 | B/C*3 | 8/16*4 | 0–3*5 |

B: Burst, C: Cycle steal

Notes:
1. External requests, auto requests and on-chip peripheral module requests are all available. For on-chip peripheral module requests, however, SCI cannot be specified as the transfer request source.
2. External requests, auto requests and on-chip peripheral module requests are all available. When the SCI is also the transfer request source, however, the transfer destination or transfer source must be the SCI respectively.
3. If the transfer request source is the SCI, cycle steal only.
4. The access size permitted when the transfer destination or source is an on-chip peripheral module register.
5. If the transfer request is an external request, channels 0 and 1 only.

**Bus Mode and Channel Priority Order:** When a given channel (1) is transferring in burst mode and there is a transfer request to a channel (2) with a higher priority, the transfer of the channel

with higher priority (2) will begin immediately. When channel 2 is also operating in the burst mode, the channel 1 transfer will continue when the channel 2 transfer has completely finished. When channel 2 is in the cycle steal mode, channel 1 will begin operating again after channel 2 completes the transfer of one transfer unit, but the bus will then switch between the two in the order channel 1, channel 2, channel 1, channel 2. Since channel 1 is in burst mode, it will not give the bus to the CPU. This example is illustrated in figure 9.12.

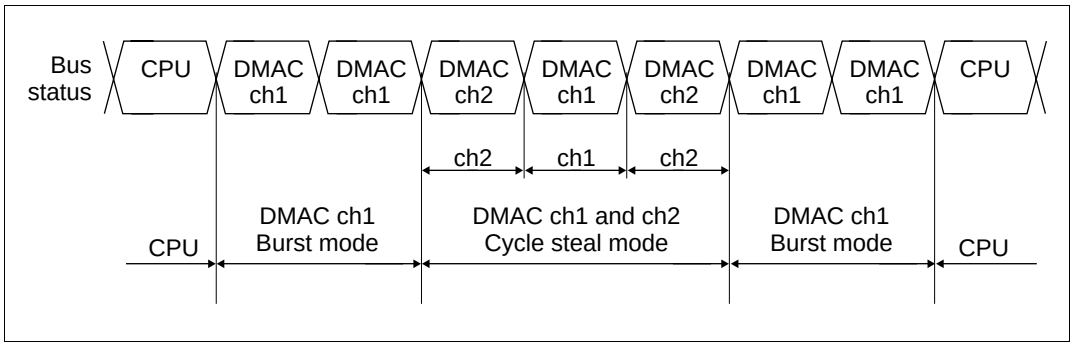



**Figure 9.12 Bus Handling when Multiple Channels Are Operating**

### 9.3.5 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sample Timing

**Number of States in Bus Cycle:** The number of states in the bus cycle when the DMAC is the bus master is controlled by the bus state controller just as it is when the CPU is the bus master. The bus cycle in the dual address mode is controlled by wait state control register 1 (WCR1) while the single address mode bus cycle is controlled by wait state control register 2 (WCR2). For details, see section 8.9, Wait State Control.

**$\overline{\text{DREQ}}$ Pin Sampling Timing:** Normally, when DREQ input is detected immediately prior to the rise edge of the clock pulse (CK) in external request mode, a DMAC bus cycle will be generated and the DMA transfer performed two states later at the earliest. The sampling timing after $\overline{\text{DREQ}}$ input detection differs by bus mode, address mode and method of $\overline{\text{DREQ}}$ input detection.

- $\overline{\text{DREQ}}$ pin sampling timing in the cycle steal mode

  In the cycle steal mode, the sampling timing is the same regardless of whether the $\overline{\text{DREQ}}$ is detected by edge or level. When edge is being detected, however, once sampled it will not be sampled again until the next edge detection. Once $\overline{\text{DREQ}}$ input is detected, the next sampling is not performed until the first state, among those DMAC bus cycles thereby produced, in which a DACK signal is output (including the detection state itself). The next sampling occurs immediately prior to the rise edge of the clock pulse(CK) of the third state after the bus cycle previous to the bus cycle in which the DACK signal is output.

Figure 9.13 to 9.22 show the sampling timing of the pin $\overline{\text{DREQ}}$ in the cycle steal mode for each bus cycle. When no $\overline{\text{DREQ}}$ input is detected at the sampling after the aforementioned $\overline{\text{DREQ}}$ detection, the next sampling occurs in the next stage in which a DACK signal is output. If no DREQ input is detected at this time, sampling occurs at every state thereafter.

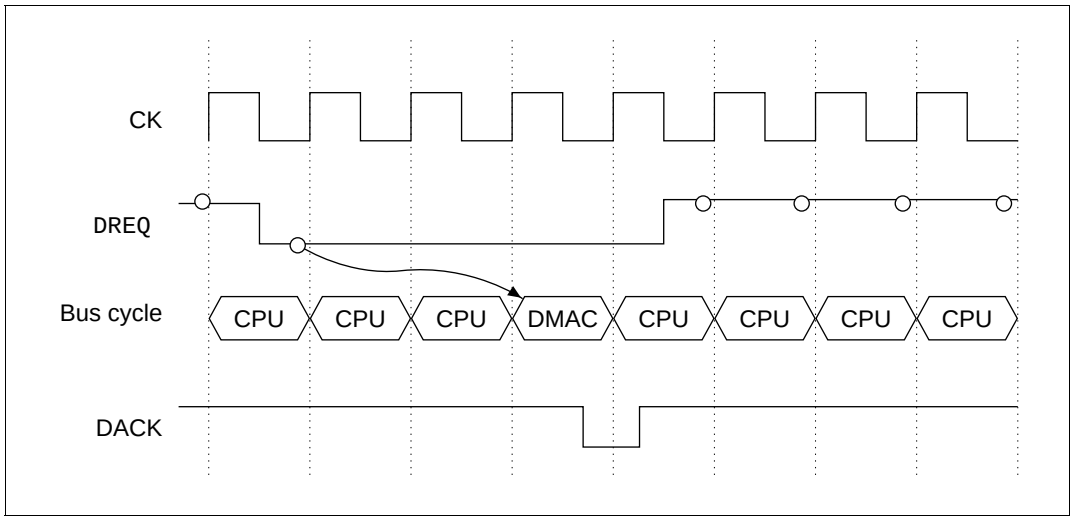



**Figure 9.13 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Single address mode, bus cycle = 1 state)**

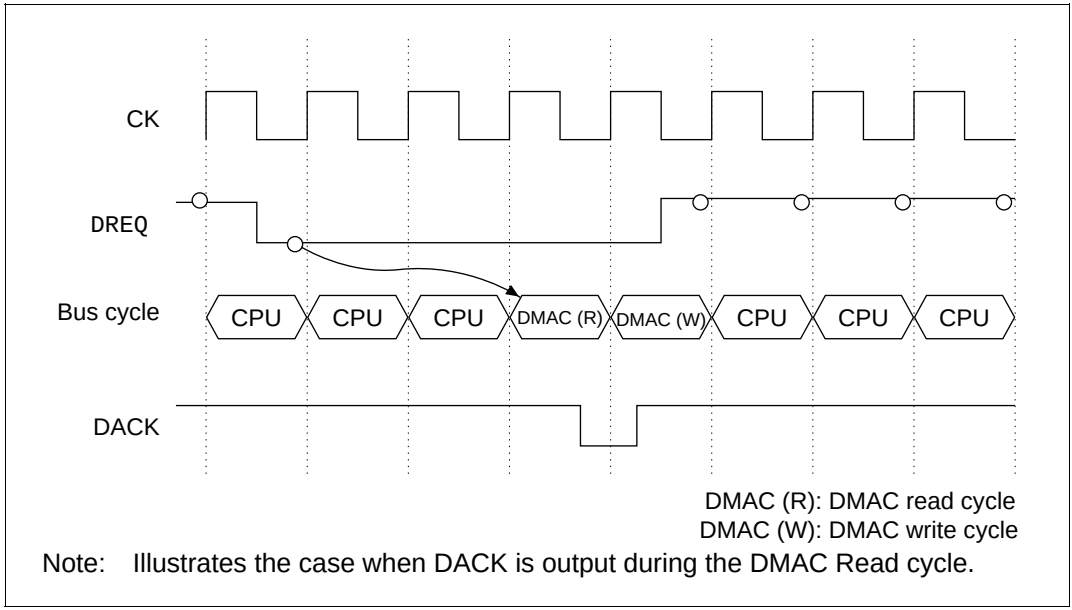



**Figure 9.14 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Dual address mode, bus cycle = 1 state)**

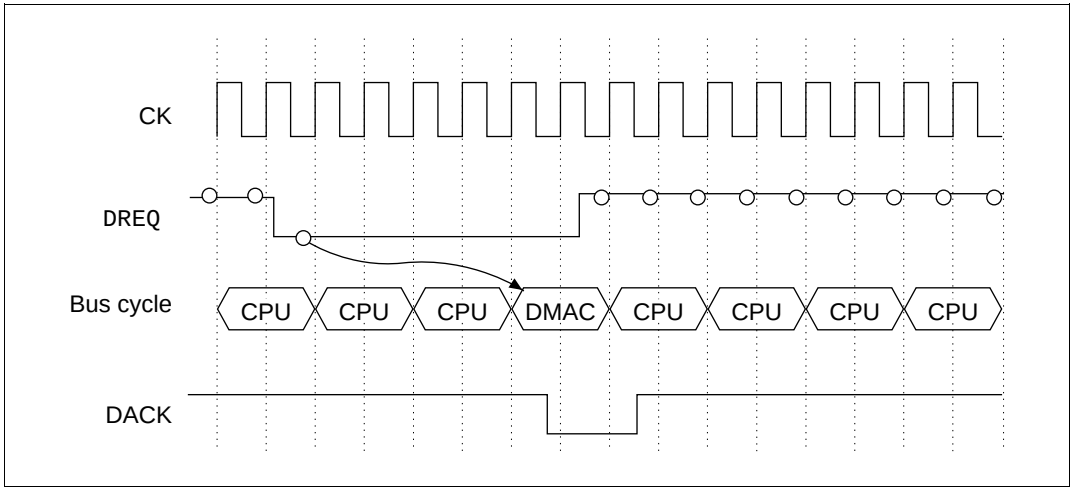
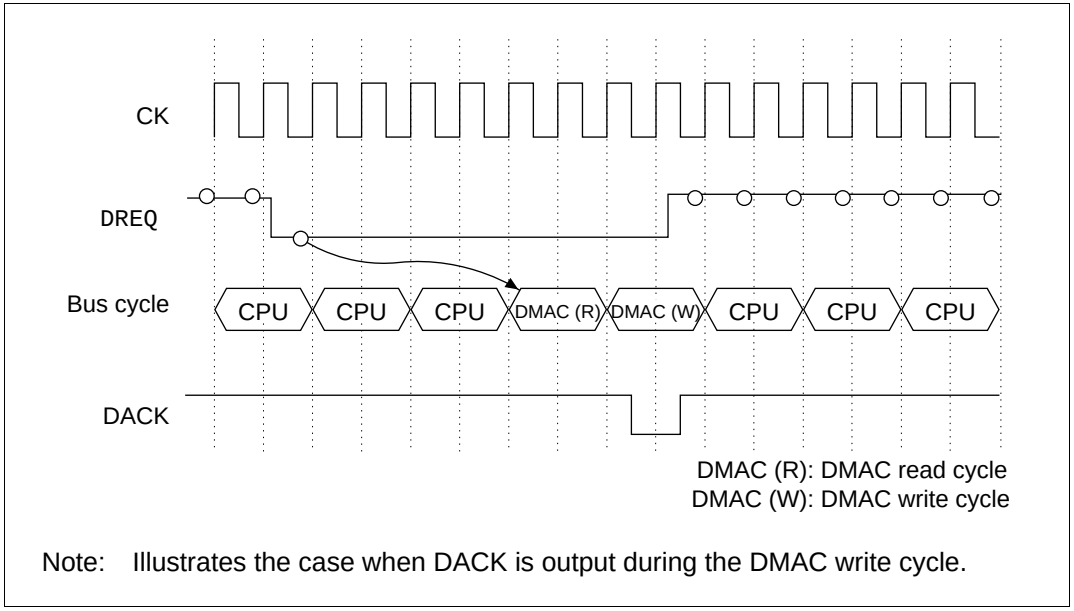
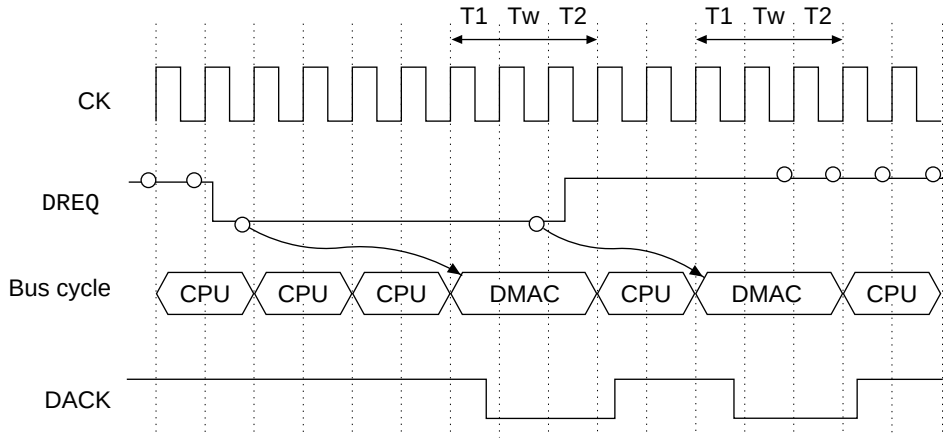



**Figure 9.15 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Single address mode, bus cycle = 2 states)**




Note: Illustrates the case when DACK is output during the DMAC write cycle.

**Figure 9.16 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Dual address mode, bus cycle = 2 states)**




Note: When DREQ is negated at the third state of the DMAC cycle, the next DMA transfer will be executed because the sampling is done at the second state of the DMAC cycle.

**Figure 9.17 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Single address mode, bus cycle = 2 states + 1 wait state)**

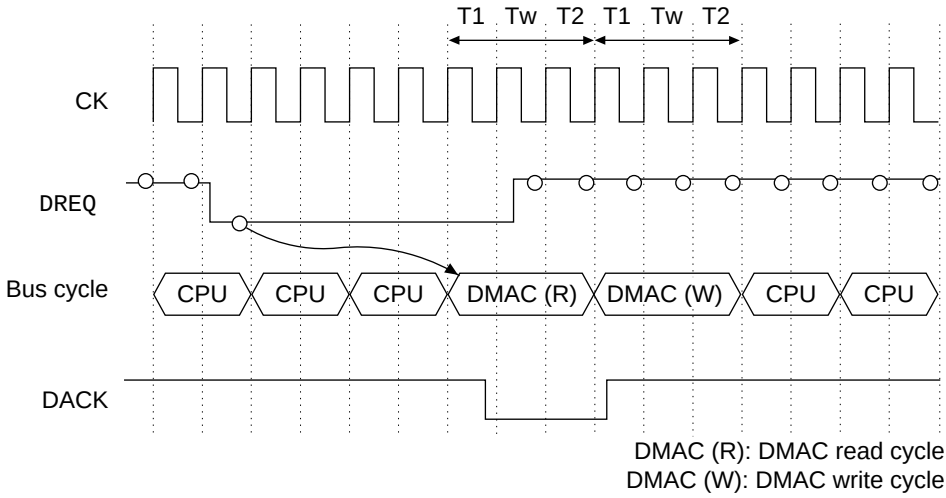
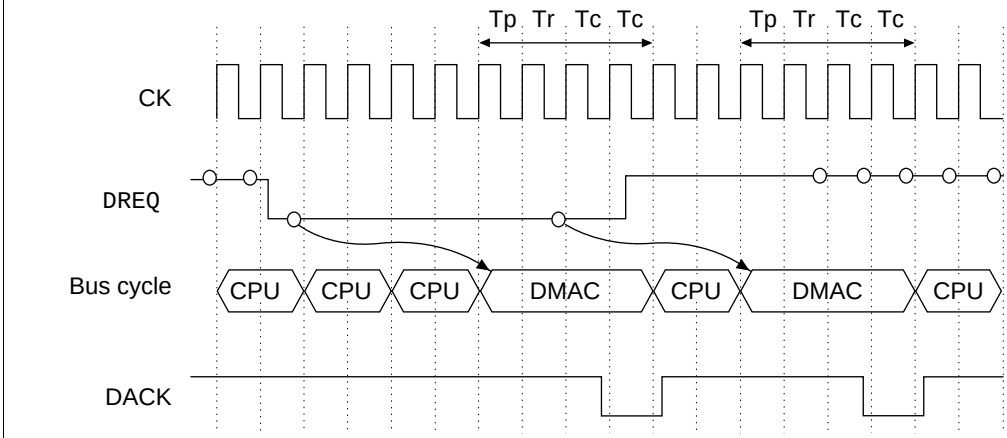
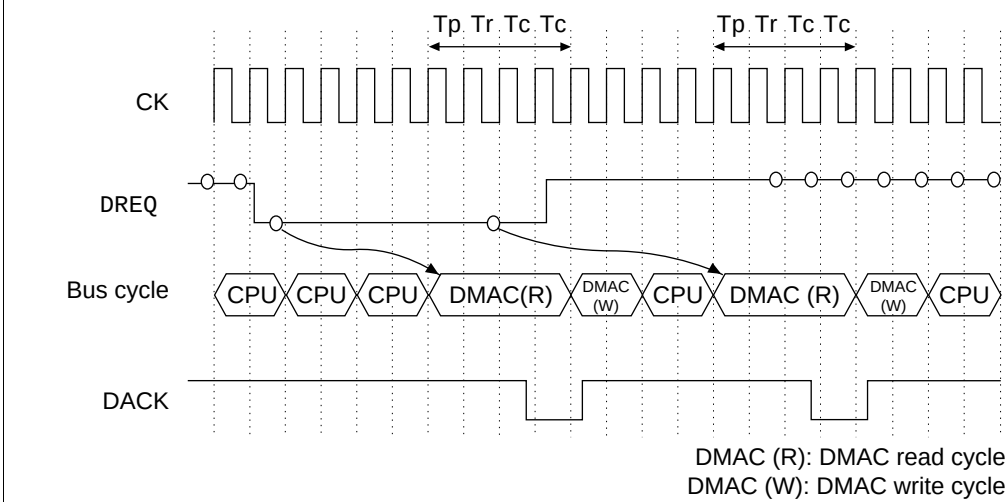



**Figure 9.18 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Dual address mode, bus cycle = 2 states + 1 wait state)**



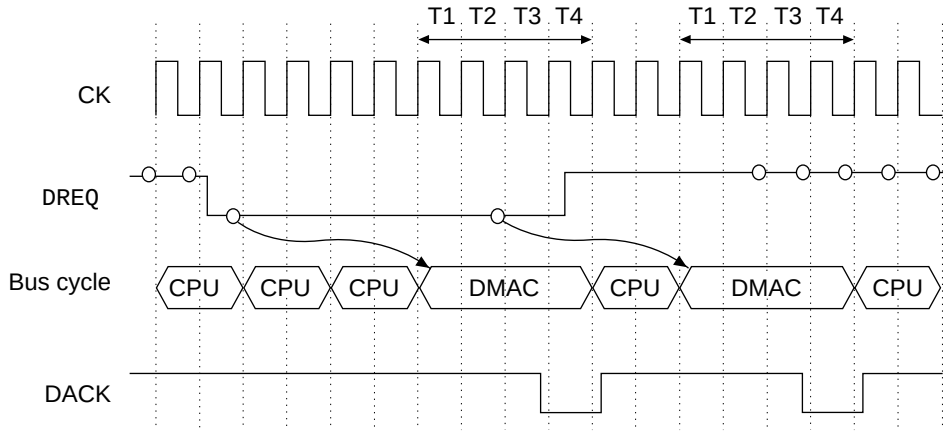

Note: When DREQ is negated at the fourth state of the DMAC cycle, the next DMA transfer will be executed because the sampling is done at the second state of the DMAC cycle.

**Figure 9.19 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Single address mode, bus cycle = DRAM bus cycle (long pitch normal mode))**



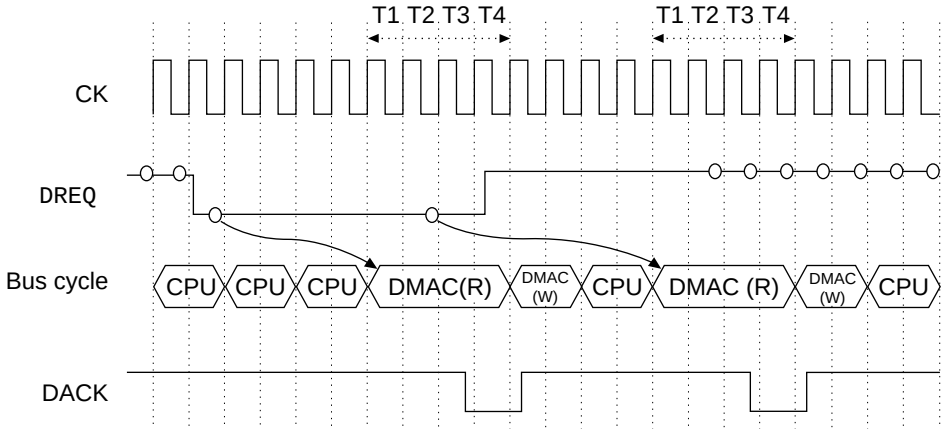

Note: When DREQ is negated at the fourth state of the DMAC cycle, the next DMA transfer will be executed because the sampling is done at the second state of the DMAC cycle.

**Figure 9.20 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Dual address mode, bus cycle = DRAM bus cycle (long pitch normal mode))**




Note: When DREQ is negated at the fourth state of the DMAC cycle, the next DMA transfer will be executed because the sampling is done at the second state of the DMAC cycle.

**Figure 9.21 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Single address mode, bus cycle = Address/data multiplex I/O bus cycle)**




DMAC (R): DMAC read cycle
DMAC (W): DMAC write cycle

Note: When DREQ is negated at the fourth state of the DMAC cycle, the next DMA transfer will be executed because the sampling is done at the second state of the DMAC cycle.

**Figure 9.22 $\overline{\text{DREQ}}$ Sampling Timing in Cycle Steal Mode (Output with $\overline{\text{DREQ}}$ level detection and DACK active low) (Dual address mode, bus cycle = Address/data multiplex I/O bus cycle)**

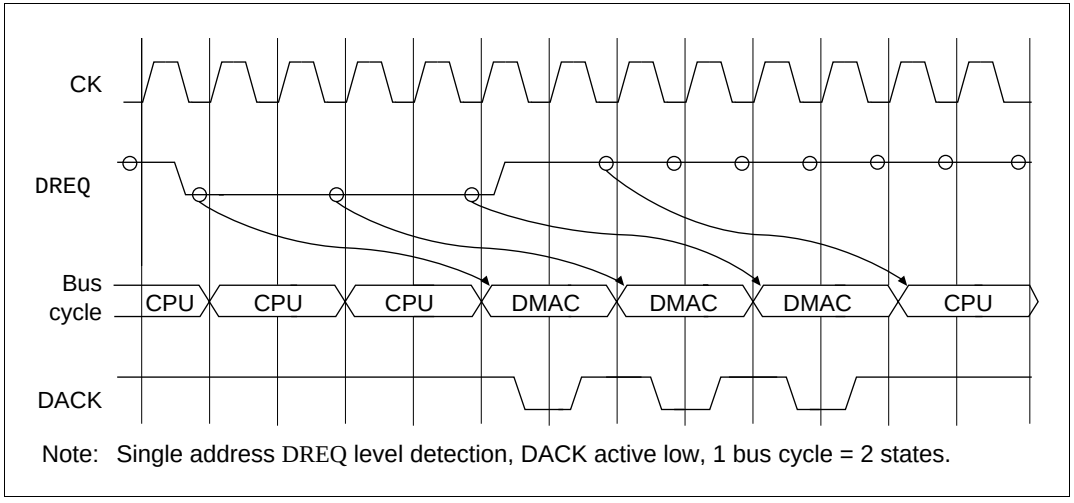
- $\overline{\text{DREQ}}$ pin sampling timing in the burst mode

  In the burst mode, the sampling timing differs depending on whether $\overline{\text{DREQ}}$ is detected by edge or level.

  When $\overline{\text{DREQ}}$ input is being detected by edge, once the falling edge of the $\overline{\text{DREQ}}$ signal is detected, the DMA transfer continues until the transfer end conditions are satisfied, regardless of the status of the $\overline{\text{DREQ}}$ pin. No sampling happens during this time. After the transfer ends, sampling occurs every state until the TE bit of the CHCR is cleared.

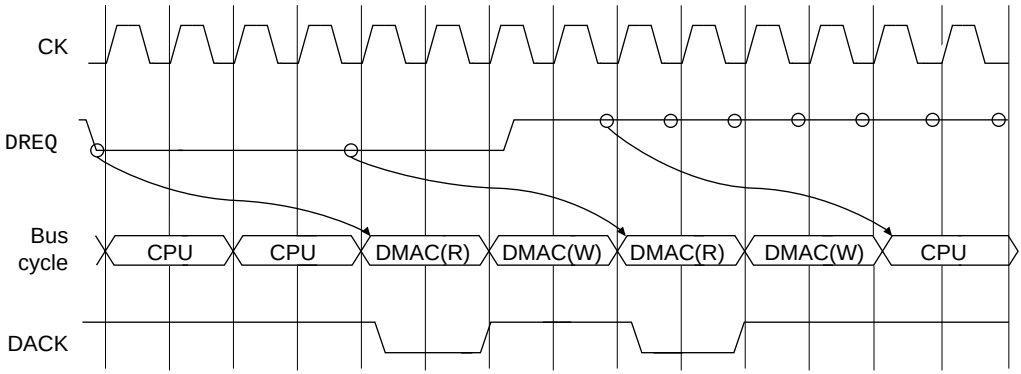
  When $\overline{\text{DREQ}}$ input is being detected by level, once the $\overline{\text{DREQ}}$ input is detected, next sampling is performed at the end of every CPU or DMAC bus cycle in the single address mode. In the dual address mode, the next sampling is performed at the start of every DMAC read cycle. In both the single address mode and dual address mode, if no $\overline{\text{DREQ}}$ input is detected at this time, sampling thereafter occurs at every state.

  Figures 9.23 and 9.24 show the $\overline{\text{DREQ}}$ pin sampling timing in burst mode when $\overline{\text{DREQ}}$ input is detected by low level.




Note: Single address DREQ level detection, DACK active low, 1 bus cycle = 2 states.

**Figure 9.23 $\overline{\text{DREQ}}$ Pin Sampling Timing in Burst Mode**




Note: Dual address DREQ level detection, DACK active low, DACK output in read cycle, 1 bus cycle = 2 states.

**Figure 9.24 $\overline{\text{DREQ}}$ Pin Sampling Timing in Burst Mode**

### 9.3.6 DMA Transfer Ending Conditions

The DMA transfer ending conditions vary for individual channels ending and all channels ending together.

**Individual Channel Ending Conditions:** There are two ending conditions. A transfer ends when the value of the channel's DMA transfer count register (TCR) is 0, or when the DE bit of the channel's CHCR is cleared to 0.

- When TCR is 0: When the TCR value becomes 0 and the corresponding channel’s DMA transfer ends, the transfer end flag bit (TE) is set in the CHCR. If the IE (interrupt enable) bit has been set, a DMAC interrupt (DEI) is requested to the CPU.
- When DE of CHCR is 0: Software can halt a DMA transfer by clearing the DE bit in the channel’s CHCR. The TE bit is not set when this happens.

**Conditions for Ending All Channels Simultaneously:** Transfers on all channels end when 1) the NMIF (NMI flag) bit or AE (address error flag) bit is set to 1 in the DMAOR, or 2) when the DME bit in the DMAOR is cleared to 0.

- Transfers ending when the NMIF or AE bit is set to 1 in DMAOR: When an NMI interrupt or DMAC address error occurs, the NMIF or AE bit is set to 1 in the DMAOR and all channels stop their transfers. The SAR, DAR, TCR are all updated by the transfer immediately preceding the halt. The TE bit is not set. To resume the transfers after NMI interrupt exception processing or address error exception processing, clear the appropriate flag bit to 0. When a channel's DE bit is then set to 1, the transfer on that channel will restart. To avoid restarting a transfer on a particular channel, keep its DE bit cleared to 0. In the dual address mode, the DMA transfer will be halted after the completion of the write cycle that follows the initial read cycle in which the address error occurs. SAR, DAR and TCR are updated by the final transfer.
- Transfers ending when DME is cleared to 0 in DMAOR: Clearing the DME bit to 0 in the DMAOR forcibly aborts the transfers on all channels at the end of the current cycle. The TE bit is not set.

## 9.4 Examples of Use

### 9.4.1 DMA Transfer between On-Chip RAM and a Memory-Mapped External Device

In the following example, data is transferred from an on-chip RAM to a memory-mapped external device with an input capture A/compare match A interrupt (IMIA0) from channel 0 of the 16-bit integrated-timer pulse unit (ITU) as the transfer request signal. The transfer is performed by DMAC channel 3. Table 9.7 shows the transfer conditions and register values.

**Table 9.7 Transfer Conditions and Register Settings for Transfer Between On-Chip RAM and Memory-Mapped External Device**

| Transfer Conditions | Register | Setting |
|---|---|---|
| Transfer source: on-chip RAM | SAR3 | H'FFFFE00 |
| Transfer destination: memory-mapped external device | DAR3 | Destination address |
| Number of transfers: 8 | TCR3 | H'0008 |
| Transfer destination address: fixed | CHCR3 | H'1805 |
| Transfer source address: incremented | | |
| Transfer request source (transfer request signal): ITU channel 0 (IMIA0) | | |
| Bus mode: cycle steal | | |
| Transfer unit: byte | | |
| DEI interrupt request generated at end of transfer (channel 3 enabled for transfer) | | |
| Channel priority order: fixed (0 > 3 > 2 > 1) (all channels transfer enabled) | DMAOR | H'0001 |

### 9.4.2 Example of DMA Transfer between On-Chip SCI and External Memory

In this example, receive data of on-chip serial communications interface (SCI) channel 0 is transferred to external memory using DMAC channel 3. Table 9.8 shows the transfer conditions and register settings.

**Table 9.8 Transfer Conditions and Register Settings for Transfer between On-Chip SCI and External Memory**

| Transfer Conditions | Register | Setting |
|---|---|---|
| Transfer source: RDR0 of on-chip SCI0 | SAR3 | H'FFFFEC5 |
| Transfer destination: external memory | DAR3 | Destination address |
| Number of transfers: 64 | TCR3 | H'0040 |
| Transfer destination address: incremented | CHCR3 | H'4405 |
| Transfer source address: fixed | | |
| Transfer request source (transfer request signal): SCI0 (RXI0) | | |
| Bus mode: cycle steal | | |
| Transfer unit: byte | | |
| DEI interrupt request generated at end of transfer (channel 3 enabled for transfer | | |
| Channel priority order: fixed (0 > 3 > 2 > 1) (all channels transfer enabled) | DMAOR | H'0001 |

## 9.5 Cautions

1. All registers other than the DMA operations register (DMAOR) and DMA channel control registers 0–3 (CHCR0–CHCR3) should be accessed in word or long word units.
2. Before rewriting the RS0–RS3 bits of CHCR0–CHCR3, first clear the DE bit to 0 (when rewriting CHCR with a byte access, be sure to set the DE bit to 0 in advance).
3. Even when the NMI interrupt is input when the DMAC is not operating, the NMIF bit of the DMAOR will be set.
4. Interrupt during DMAC Transfer
   a. When an NMI interrupt is input, the DMAC stops operation and returns the bus right to the CPU. The CPU then executes the interrupt processing.
   b. When an interrupt other than an NMI occurs.
      - When the DMAC is in burst mode.

        The DMAC does not return the bus right to the CPU in burst mode. Therefore, even when an interrupt is requested in DMAC operation, the CPU cannot get the bus right, causing the interrupt processing not to be executed. When the DMAC completes transfer and the CPU gets the bus right, the CPU executes the interrupt processing if the interrupt requested during DMAC transfer is not cleared.*

        * Clear conditions for an interrupt request.

      — When an interrupt is requested from an on-chip peripheral module, the interrupt factor flag is cleared.

      — When an interrupt is requested by $\overline{\text{IRQ}}$ (edge detection), the CPU begins the $\overline{\text{IRQ}}$ interrupt processing of the request source.

      — When an interrupt is requested by $\overline{\text{IRQ}}$ (level detection), the $\overline{\text{IRQ}}$ interrupt request signal returned to high level.

      - When the DMAC is in cycle-steal mode.

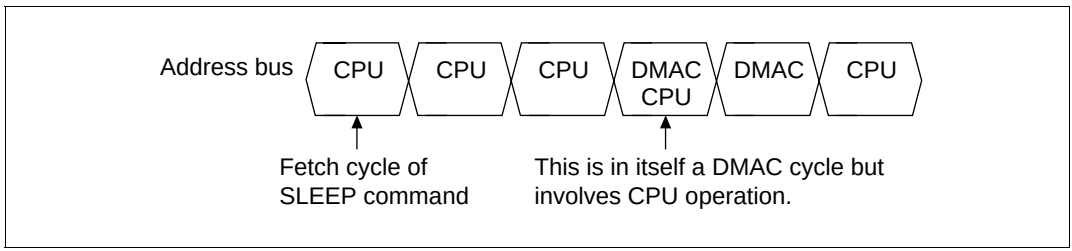
        The DMAC returns the bus right to the CPU every when the DMAC completes a transfer unit in cycle-steal mode. Therefore, the CPU executes the requested interrupt processing when getting the bus right.
5. The CPU and DMAC leaves the bus right released and the operation of the LSI is stopped when the following conditions are satisfied.
   - The warp bit (WARP) of the bus control register (BCR) of the bus controller (BSC)is set.
   - The DMAC is in cycle-steal transfer mode.
   - The CPU accesses (reads/writes) the on-chip I/O space.

- Countermeasure
  Set the warp bit of BCR to 0 and set it to normal mode.

6. Notes on use of the SLEEP command
   a. Operation contents
      When the bus cycle of DMAC is entered immediately after executing the SLEEP command, there are cases when the DMA transfer is carried out correctly.
   b. Countermeasure
      - Stop the operation (for exemple, clearing of the DMA enable bit (DE) of the DMA channel control register(CHCRn)) before entering SLEEP.
      - When using DMAC during SLEEP, operate DMAC after releasing SLEEP through interruption.

In cases when the CPU does not carry out any other processing but is waiting for DMAC to end its transfer during DMAC operation, do not use the SLEEP command, but use the transfer end flag bit (TE) of the channel DMA control register and the polling software loop.

**Phenomenon:** If the bus cycle of DMAC is entered immediately after executing the SLEEP command, the bus cycle of DMAC may conflict with that of CPU.




Accordingly, the bus cycle of DMAC which has conflicted with that of CPU may malfunction.

7. Sampling of $\overline{\text{DREQ}}$
   If $\overline{\text{DREQ}}$ is set to level detection in the DMA cycle steal mode, sampling of $\overline{\text{DREQ}}$ may take place before DACK is output. Note that some system configurations involve unnecessary DMA transfers.

- Operation
  As shown in Figure 9.16, sampling of $\overline{\text{DREQ}}$ is carried out immediately before the leading edge of the third-state clock (CK) after completion of the bus cycle preceding the DMA bus cycle where DACK is output.
  If DACK is output after the third state of the DMA bus cycle, sampling of $\overline{\text{DREQ}}$ must be carried out before DACK is output.

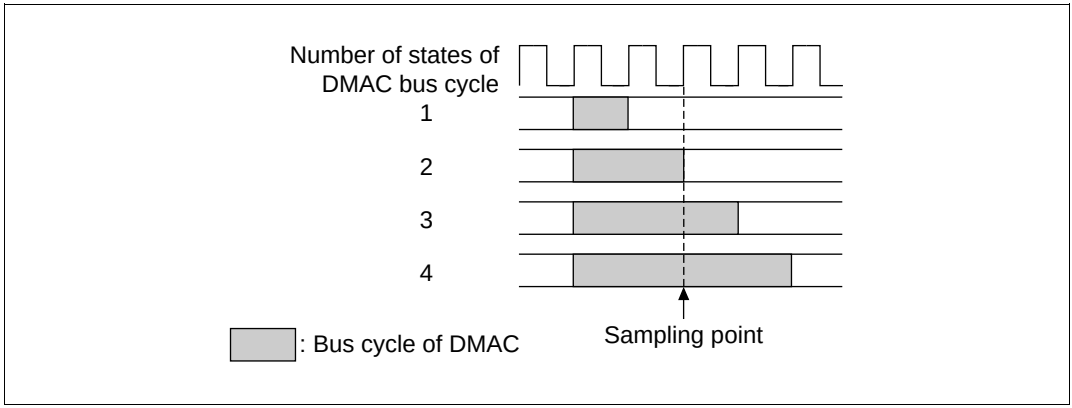



**Figure 9.16 Sampling Points of DREQ**

Especially as shown in Figure 9.17, if the bus cycle of DMA is a full access to DRAM or if refresh demand is generated, sampling of $\overline{\text{DREQ}}$ takes place before DACK is output as mentioned above. This phenomenon is found when one of the following transfers is made with $\overline{\text{DREQ}}$ set to the level detection in the DMA cycle steal mode, in a system which employs DRAM (refresh enabled).

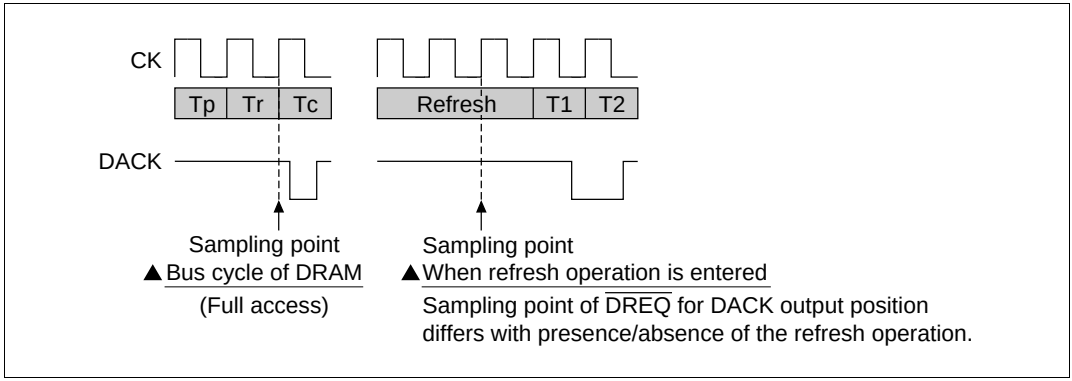



**Figure 9.17 Example of DREQ Sampling before Output of DACK**

- Transfer from a device having DACK to memory in the single address mode (not restricted to DRAM)
- Transfer from DRAM to a device having DACK in the single address mode
- Output at DACK write in the dual address mode
  Output at DACK read in the dual address mode and DMA transfer using DRAM as a source

- Countermeasure
  To prevent unnecessary DMA transfers, configure the system where $\overline{\text{DREQ}}$ is used for edge detection and the edge corresponding to the next transfer request occurs after the DACK output.

8. When the following operations are performed in the order shown when the pin to which $\overline{\text{DREQ}}$ input is assigned is designated as a general input pin by the pin function controller (PFC) and inputs a low-level signal, the $\overline{\text{DREQ}}$ falling edge is detected, and a DMA transfer request accepted, immediately after the setting in (b) is performed:
   (a) A channel control register (CHCRn) setting is made so that an interrupt is detected at the falling edge of $\overline{\text{DREQ}}$.
   (b) The function of the pin to which $\overline{\text{DREQ}}$ input is assigned is switched from general input to $\overline{\text{DREQ}}$ input by a pin function controller (PFC) setting.

   Therefore, when switching the pin function from general input pin to $\overline{\text{DREQ}}$ input, the pin function controller (PFC) setting should be changed to $\overline{\text{DREQ}}$ input while the pin to which $\overline{\text{DREQ}}$ input is assigned is high.

# Section 10 16-Bit Integrated-Timer Pulse Unit (ITU)

## 10.1 Overview

The SuperH microcomputer has an on-chip 16-bit integrated-timer pulse unit (ITU) with five channels of 16-bit timers.

### 10.1.1 Features

ITU features are listed below:

- Can process a maximum of twelve different pulse outputs and ten different pulse inputs.
- Has ten general registers (GR), two per channel, that can be set to function independently as output compare or input capture.
- Selection of eight counter input clock sources for all channels
  - — Internal clock: $\phi$, $\phi/2$, $\phi/4$, $\phi/8$,
  - — External clock: TCLKA, TCLKB, TCLKC, TCLKD
- All channels can be set for the following operating modes:
  - — Compare match waveform output: 0 output/1 output/selectable toggle output (0 output/1 output for channel 2).
  - — Input capture function: Selectable rising edge, falling edge, or both rising and falling edges.
  - — Counter clearing function: Counters can be cleared by a compare match or input capture.
  - — Synchronizing mode: Two or more timer counters (TCNT) can be written to simultaneously. Two or more timer counters can be simultaneously cleared by a compare match or input capture. Counter synchronization functions enable synchronized input/output.
  - — PWM mode: PWM output can be provided with any duty cycle. When combined with the counter synchronizing function, enables up to five-phase PWM output.
- Channel 2 can be set to the phase counting mode: Two-phase encoder output can be counted automatically.
- Channels 3 and 4 can be set in the following modes:
  - — Reset-synchronized PWM mode: By combining channels 3 and 4, 3-phase PWM output is possible with positive and negative waveforms .
  - — Complementary PWM mode: By combining channels 3 and 4, 3-phase PWM output is possible with non-overlapping positive and negative waveforms.
- Buffer operation: Input capture registers can be double-buffered. Output compare registers can be updated automatically.
- High-speed access via internal 16-bit bus: The TCNT, GR, and buffer register (BR) 16-bit registers can be accessed at high speed via a 16-bit bus.

- Fifteen interrupt sources: Ten compare match/input capture interrupts (2 sources per channel) and five overflow interrupts are vectored independently for a total of 15 sources.
- Can activate DMAC: The compare match/input capture interrupts of channels 0–3 can start the DMAC (one for each of four channels).
- Output trigger can be generated for the programmable timing pattern controller (TPC): The compare match/input capture signals of channel 0–3 can be used as output triggers for the TPC.

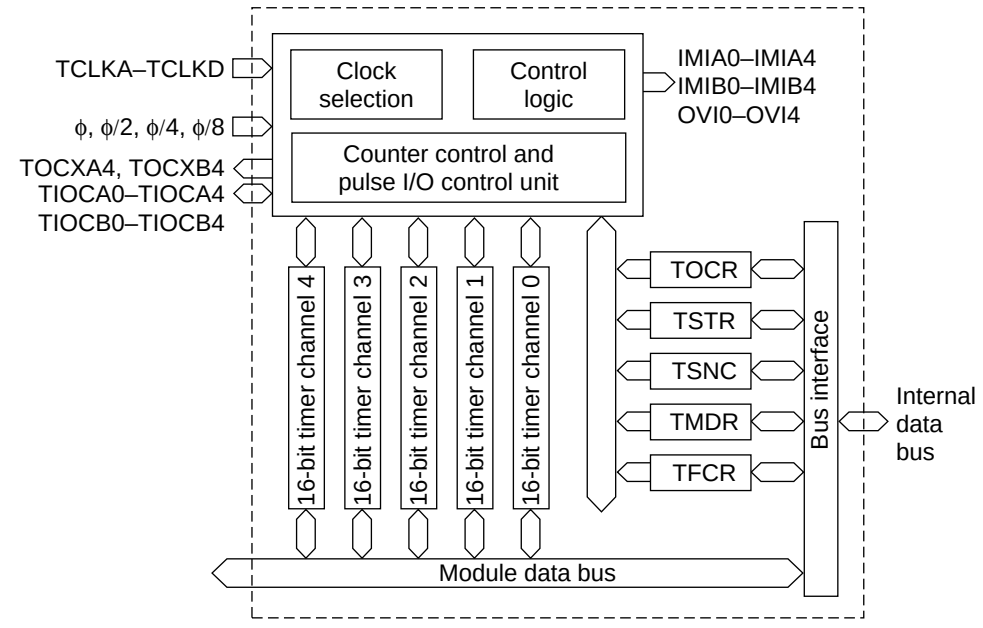
Table 10.1 summarizes the ITU functions.

**Table 10.1 ITU Functions**

| Item | | Channel 0 | Channel 1 | Channel 2 | Channel 3 | Channel 4 |
|---|---|---|---|---|---|---|
| Counter clocks | | Internal: φ, φ/2, φ/4, φ/8<br>External: Independently selectable from TCLKA, TCLKB, TCLKC, and TCLKD | | | | |
| General registers (output compare/ input capture dual registers) | | GRA0, GRB0 | GRA1, GRB1 | GRA2, GRB2 | GRA3, GRB3 | GRA4, GRB4 |
| Buffer registers | | No | No | No | BRA3, BRB3 | BRA4, BRB4 |
| Input/output pins | | TIOCA0, TIOCB0 | TIOCA1, TIOCB1 | TIOCA2, TIOCB2 | TIOCA3, TIOCB3 | TIOCA4, TIOCB4 |
| Output pins | | No | No | No | No | TOCXA4, TOCXB4 |
| Counter clear function (compare match or input capture) | | GRA0/GRB0 | GRA1/GRB1 | GRA2/GRB2 | GRA3/GRB3 | GRA4/GRB4 |
| Compare match output | 0 | Yes | Yes | Yes | Yes | Yes |
| | 1 | Yes | Yes | Yes | Yes | Yes |
| | Toggle output | Yes | Yes | No | Yes | Yes |
| Input capture function | | Yes | Yes | Yes | Yes | Yes |
| Synchronization | | Yes | Yes | Yes | Yes | Yes |
| PWM mode | | Yes | Yes | Yes | Yes | Yes |
| Reset-synchronized PWM mode | | No | No | No | Yes | Yes |
| Complementary PWM mode | | No | No | No | Yes | Yes |
| Phase counting mode | | No | No | Yes | No | No |
| Buffer operation | | No | No | No | Yes | Yes |
| DMAC activation | | GRA0 compare match or input capture | GRA1 compare match or input capture | GRA2 compare match or input capture | GRA3 compare match or input capture | No |
| Interrupt sources (three) | | • Compare match/input capture A0<br>• Compare match/input capture B0<br>• Overflow | • Compare match/input capture A1<br>• Compare match/input capture B1<br>• Overflow | • Compare match/input capture A2<br>• Compare match/input capture B2<br>• Overflow | • Compare match/input capture A3<br>• Compare match/input capture B3<br>• Overflow | • Compare match/input capture A4<br>• Compare match/input capture B4<br>• Overflow |

### 10.1.2 Block Diagram

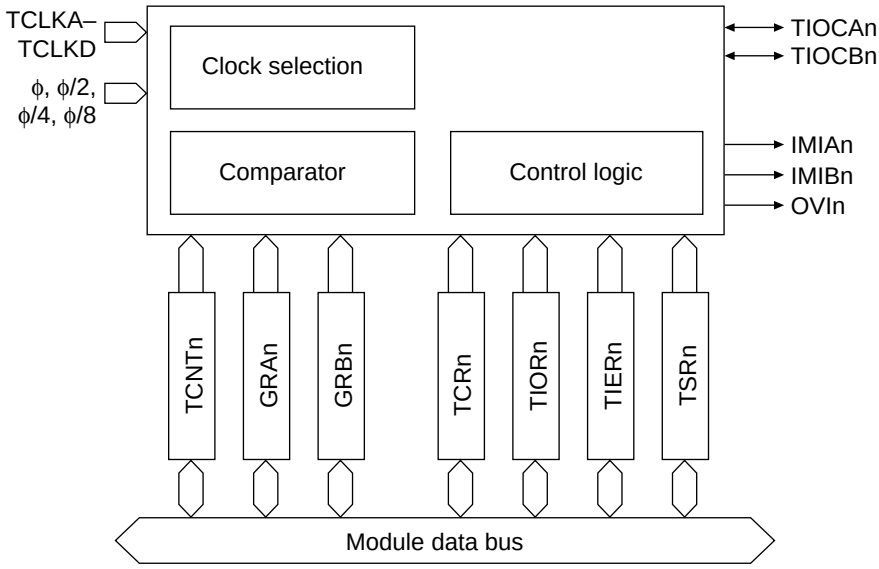
**ITU Block Diagram (Complete):** Figure 10.1 is the block diagram of the ITU.




TOCR: Timer output control register (8 bits)
TSTR: Timer start regsiter (8 bits)
TSNC: Timer synchronization register (8 bits)
TMDR: Timer mode register (8 bits)
TFCR: Timer function control register (8 bits)

**Figure 10.1 ITU Block Diagram**

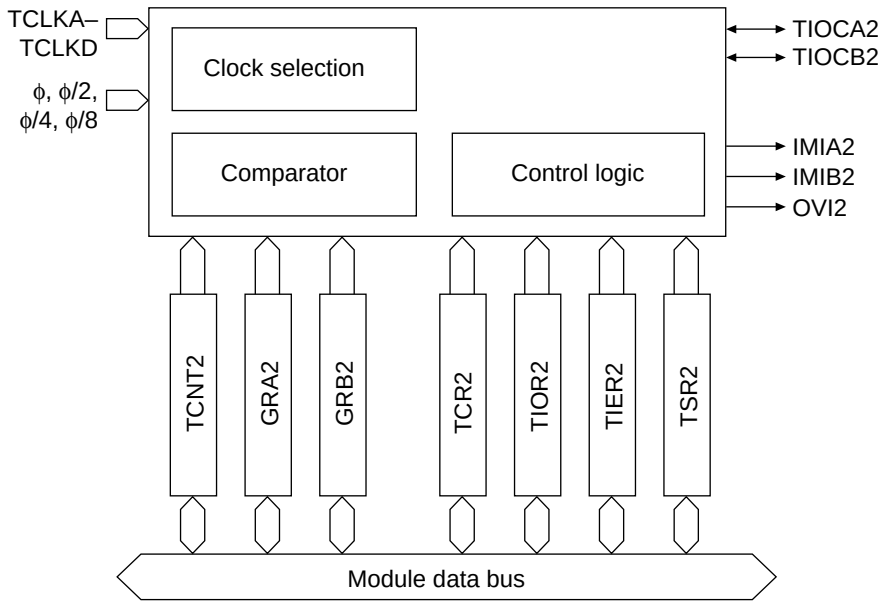
**Block Diagram of Channels 0 and 1:** ITU channels 0 and 1 have the same function. Figure 10.2 is a block diagram of channels 0 and 1.




TCNTn: Timer counter n (16 bits)
GRAn, GRBn: General registers An, Bn (input capture/output compare dual use) (16 bits × 2)
TCRn: Timer control register n (8 bits)
TIORn: Timer I/O control register n (8 bits)
TIERn: Timer interrupt enable register n (8 bits)
TSRn: Timer status register n (8 bits) (n = 0 or 1)

**Figure 10.2 Channels 0 and 1 Block Diagram (One Channel Shown)**

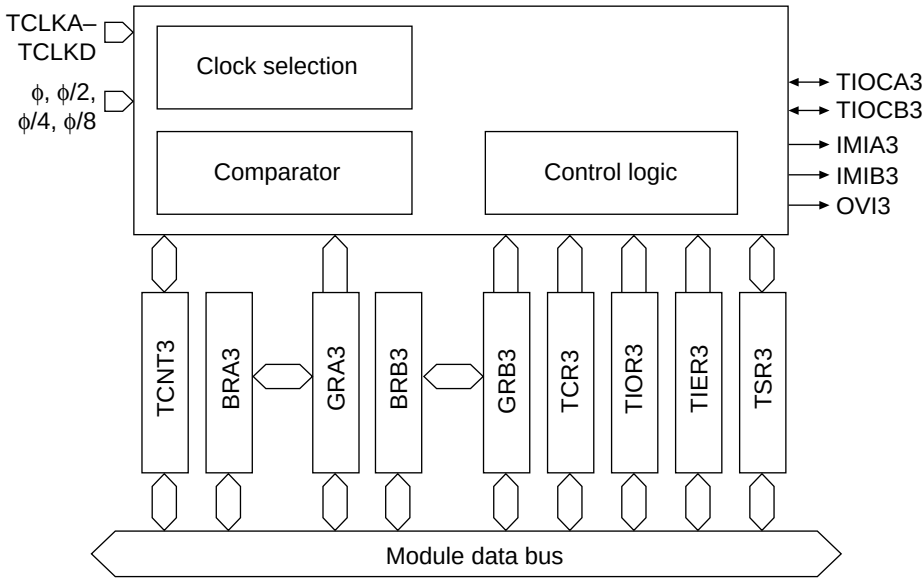
**Block Diagram of Channel 2:** Figure 10.3 is a block diagram of channel 2. Channel 2 is 0 output/1 output only.




TCNT2: Timer counter 2 (16 bits)
GRA2, GRB2: General registers A2, B2 (input capture/output compare dual use) (16 bits × 2)
TCR2: Timer control register 2 (8 bits)
TIOR2: Timer I/O control register 2 (8 bits)
TIER2: Timer interrupt enable register 2 (8 bits)
TSR2: Timer status register 2 (8 bits)

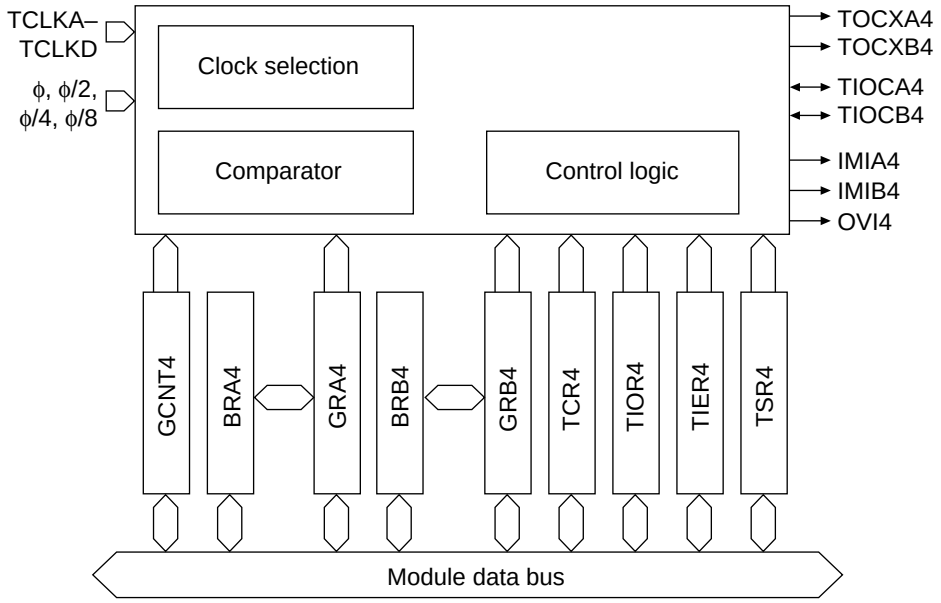
**Figure 10.3 Channel 2 Block Diagram**

**Block Diagrams of Channels 3 and 4:** Figure 10.4 is a block diagram of channel 3; figure 10.5 is a block diagram of channel 4.




TCNT3: Timer counter 3 (16 bits)
GRA3, GRB3: General registers A3, B3 (input capture/output compare dual use) (16 bits × 2)
BRA3, BRB3: Buffer registers A3, B3 (input capture/output compare dual use) (16 bits × 2)
TCR3: Timer control register 3 (8 bits)
TIOR3: Timer I/O control register 3 (8 bits)
TIER3: Timer interrupt enable register 3 (8 bits)
TSR3: Timer status register 3 (8 bits)

**Figure 10.4 Channels 3 Block Diagram**




TCNT4: Timer counter 4 (16 bits)
GRA4, GRB4: General registers A4, B4 (input capture/output compare dual use) (16 bits × 2)
BRA4, BRB4: Buffer registers A4, B4 (input capture/output compare dual use) (16 bits × 2)
TCR4: Timer control register 4 (8 bits)
TIOR4: Timer I/O control register 4 (8 bits)
TIER4: Timer interrupt enable register 4 (8 bits)
TSR4: Timer status register 4 (8 bits)

**Figure 10.5 Channel 4 Block Diagram**

### 10.1.3 Input/Output Pins

Table 10.2 summarizes the ITU pins. External pin functions should be set with the pin function controller to match to the ITU setting. See section 15, Pin Function Controller, for details. ITU pins need to be set using the pin function controller (PFC) after the LSI is set to the ITU mode.

**Table 10.2 Pin Configuration**

| Channel | Name | Pin Name | I/O | Function |
|---|---|---|---|---|
| Shared | Clock input A | TCLKA | I | External clock A input pin (A-phase input pin in phase counting mode) |
| | Clock input B | TCLKB | I | External clock B input pin (B-phase input pin in phase counting mode) |
| | Clock input C | TCLKC | I | External clock C input pin |
| | Clock input D | TCLKD | I | External clock D input pin |
| 0 | Input capture/output compare A0 | TIOCA0 | I/O | GRA0 output compare/GRA0 input capture/PWM output pin (in PWM mode) |
| | Input capture/output compare B0 | TIOCB0 | I/O | GRB0 output compare/GRB0 input capture |
| 1 | Input capture/output compare A1 | TIOCA1 | I/O | GRA1 output compare/GRA1 input capture/PWM output pin (in PWM mode) |
| | Input capture/output compare B1 | TIOCB1 | I/O | GRB1 output compare/GRB1 input capture |
| 2 | Input capture/output compare A2 | TIOCA2 | I/O | GRA2 output compare/GRA2 input capture/PWM output pin (in PWM mode) |
| | Input capture/output compare B2 | TIOCB2 | I/O | GRB2 output compare/GRB2 input capture |
| 3 | Input capture/output compare A3 | TIOCA3 | I/O | GRA3 output compare/GRA3 input capture/PWM output pin (in PWM mode, complementary PWM mode, or reset-synchronized PWM mode) |
| | Input capture/output compare B3 | TIOCB3 | I/O | GRB3 output compare/GRB3 input capture/PWM output pin (in complementary PWM mode or reset-synchronized PWM mode) |
| 4 | Input capture/output compare A4 | TIOCA4 | I/O | GRA4 output compare/GRA4 input capture/PWM output pin (in PWM mode, complementary PWM mode or reset-synchronized PWM mode) |
| | Input capture/output compare B4 | TIOCB4 | I/O | GRB4 output compare/GRB4 input capture/PWM output pin (in complementary PWM mode or reset-synchronized PWM mode) |
| | Output compare XA4 | TOCXA4 | I/O | PWM output pin (in complementary PWM mode or reset-synchronized PWM mode) |
| | Output compare XB4 | TOCXB4 | I/O | PWM output pin (in complementary PWM mode or reset-synchronized PWM mode) |

### 10.1.4 Register Configuration

Table 10.3 summarizes the ITU register configuration.

**Table 10.3 Register Configuration**

| Channel | Name | Abbrevi-ation | R/W | Initial Value | Address*1 | Access Size |
|---|---|---|---|---|---|---|
| Shared | Timer start register | TSTR | R/W | H'E0/H'60 | H'5FFFF00 | 8 |
| | Timer synchro register | TSNC | R/W | H'E0/H'60 | H'5FFFF01 | 8 |
| | Timer mode register | TMDR | R/W | H'80/H'00 | H'5FFFF02 | 8 |
| | Timer function control register | TFCR | R/W | H'C0/H'40 | H'5FFFF03 | 8 |
| | Timer output control register | TOCR | R/W | H'FF/H'7F | H'5FFFF31 | 8 |
| 0 | Timer control register 0 | TCR0 | R/W | H'80/H'00 | H'5FFFF04 | 8 |
| | Timer I/O control register 0 | TIOR0 | R/W | H'88/H'08 | H'5FFFF05 | 8 |
| | Timer interrupt enable register 0 | TIER0 | R/W | H'F8/H'78 | H'5FFFF06 | 8 |
| | Timer status register 0 | TSR0 | R/(W)*2 | H'F8/H'78 | H'5FFFF07 | 8 |
| | Timer counter 0 | TCNT0 | R/W | H'00 | H'5FFFF08 | 8, 16, 32 |
| | | | | | H'5FFFF09 | 8, 16, 32 |
| | General register A0 | GRA0 | R/W | H'FF | H'5FFFF0A | 8, 16, 32 |
| | | | | | H'5FFFF0B | 8, 16, 32 |
| | General register B0 | GRB0 | R/W | H'FF | H'5FFFF0C | 8, 16 |
| | | | | | H'5FFFF0D | 8, 16 |
| 1 | Timer control register 1 | TCR1 | R/W | H'80/H'00 | H'5FFFF0E | 8 |
| | Timer I/O control register 1 | TIOR1 | R/W | H'88/H'08 | H'5FFFF0F | 8 |
| | Timer interrupt enable register 1 | TIER1 | R/W | H'F8/H'78 | H'5FFFF10 | 8 |
| | Timer status register 1 | TSR1 | R/(W)*2 | H'F8/H'78 | H'5FFFF11 | 8 |
| | Timer counter 1 | TCNT1 | R/W | H'00 | H'5FFFF12 | 8, 16 |
| | | | | | H'5FFFF13 | 8, 16 |
| | General register A1 | GRA1 | R/W | H'FF | H'5FFFF14 | 8, 16, 32 |
| | | | | | H'5FFFF15 | 8, 16, 32 |
| | General register B1 | GRB1 | R/W | H'FF | H'5FFFF16 | 8, 16, 32 |
| | | | | | H'5FFFF17 | 8, 16, 32 |

**Table 10.3 Register Configuration (cont)**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] | Access Size |
|---|---|---|---|---|---|---|
| 2 | Timer control register 2 | TCR2 | R/W | H'80/H'00 | H'5FFFF18 | 8 |
| | Timer I/O control register 2 | TIOR2 | R/W | H'88/H'08 | H'5FFFF19 | 8 |
| | Timer interrupt enable register 2 | TIER2 | R/W | H'F8/H'78 | H'5FFFF1A | 8 |
| | Timer status register 2 | TSR2 | R/(W)*[2] | H'F8/H'78 | H'5FFFF1B | 8 |
| | Timer counter 2 | TCNT2 | R/W | H'00 | H'5FFFF1C | 8, 16, 32 |
| | | | | | H'5FFFF1D | 8, 16, 32 |
| | General register A2 | GRA2 | R/W | H'FF | H'5FFFF1E | 8, 16, 32 |
| | | | | | H'5FFFF1F | 8, 16, 32 |
| | General register B2 | GRB2 | R/W | H'FF | H'5FFFF20 | 8, 16 |
| | | | | | H'5FFFF21 | 8, 16 |
| 3 | Timer control register 3 | TCR3 | R/W | H'80/H'00 | H'5FFFF22 | 8 |
| | Timer I/O control register 3 | TIOR3 | R/W | H'88/H'08 | H'5FFFF23 | 8 |
| | Timer interrupt enable register 3 | TIER3 | R/W | H'F8/H'78 | H'5FFFF24 | 8 |
| | Timer status register 3 | TSR3 | R/(W)*[2] | H'F8/H'78 | H'5FFFF25 | 8 |
| | Timer counter 3 | TCNT3 | R/W | H'00 | H'5FFFF26 | 8, 16 |
| | | | | | H'5FFFF27 | 8, 16 |
| | General register A3 | GRA3 | R/W | H'FF | H'5FFFF28 | 8, 16, 32 |
| | | | | | H'5FFFF29 | 8, 16, 32 |
| | General register B3 | GRB3 | R/W | H'FF | H'5FFFF2A | 8, 16, 32 |
| | | | | | H'5FFFF2B | 8, 16, 32 |
| | Buffer register A3 | BRA3 | R/W | H'FF | H'5FFFF2C | 8, 16, 32 |
| | | | | | H'5FFFF2D | 8, 16, 32 |
| | Buffer register B3 | BRB3 | R/W | H'FF | H'5FFFF2E | 8, 16, 32 |
| | | | | | H'5FFFF2F | 8, 16, 32 |
| 4 | Timer control register 4 | TCR4 | R/W | H'80/H'00 | H'5FFFF32 | 8 |
| | Timer I/O control register 4 | TIOR4 | R/W | H'88/H'08 | H'5FFFF33 | 8 |
| | Timer interrupt enable register 4 | TIER4 | R/W | H'F8/H'78 | H'5FFFF34 | 8 |
| | Timer status register 4 | TSR4 | R/(W)*[2] | H'F8/H'78 | H'5FFFF35 | 8 |

**Table 10.3 Register Configuration (cont)**

| Channel | Name | Abbreviation | R/W | Initial Value | Address*[1] | Access Size |
|---|---|---|---|---|---|---|
| 4 (cont) | Timer counter 4 | TCNT4H | R/W | H'00 | H'5FFFF36 | 8, 16 |
| | | | | | H'5FFFF37 | 8, 16 |
| | General register A4 | GRA4H | R/W | H'FF | H'5FFFF38 | 8, 16, 32 |
| | | | | | H'5FFFF39 | 8, 16, 32 |
| | General register B4 | GRB4H | R/W | H'FF | H'5FFFF3A | 8, 16, 32 |
| | | | | | H'5FFFF3B | 8, 16, 32 |
| | Buffer register A4 | BRA4H | R/W | H'FF | H'5FFFF3C | 8, 16, 32 |
| | | | | | H'5FFFF3D | 8, 16, 32 |
| | Buffer register B4 | BRB4H | R/W | H'FF | H'5FFFF3E | 8, 16, 32 |
| | | | | | H'5FFFF3F | 8, 16, 32 |

Notes: 1. Only the values of bits A27–A24 and A8–A0 are valid; bits A23–A9 are ignored. For details on the register addresses, see section 8.3.5, Description of Areas.
2. Write 0 to clear flags.

## 10.2 ITU Register Descriptions

### 10.2.1 Timer Start Register (TSTR)

The timer start register (TSTR) is an eight-bit read/write register that starts and stops the timer counters (TCNT) of channels 0–4. TSTR is initialized to H'E0 or H'60 upon reset or standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | STR4 | STR3 | STR2 | STR1 | STR0 |
| Initial value: | * | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | R/W | R/W | R/W | R/W | R/W |

Note: Undefined

- Bits 7–5 (reserved): Cannot be modified. Bit 7 is read as undefined. Bits 6 and 5 are always read as 1. The write value to bit 7 should be 0 or 1, and the write value to bits 6 and 5 should always be 1.

- Bit 4 (count start 4 (STR4)): STR4 starts and stops TCNT4.

| Bit 4: STR4 | Description |
|---|---|
| 0 | TCNT4 is halted (initial value) |
| 1 | TCNT4 is counting |

- Bit 3 (count start 3 (STR3)): STR3 starts and stops TCNT3.

| Bit 3: STR3 | Description |
|---|---|
| 0 | TCNT3 is halted (initial value) |
| 1 | TCNT3 is counting |

- Bit 2 (count start 2 (STR2)): STR2 starts and stops TCNT2.

| Bit 2: STR2 | Description |
|---|---|
| 0 | TCNT2 is halted (initial value) |
| 1 | TCNT2 is counting |

- Bit 1 (count start 1 (STR1)): STR1 starts and stops TCNT1.

| Bit 1: STR1 | Description |
|---|---|
| 0 | TCNT1 is halted (initial value) |
| 1 | TCNT1 is counting |

- Bit 0 (count start 0 (STR0)): STR0 starts and stops TCNT0.

| Bit 0: STR0 | Description |
|---|---|
| 0 | TCNT0 is halted (initial value) |
| 1 | TCNT0 is counting |

### 10.2.2 Timer Synchro Register (TSNC)

The timer synchro register (TSNC) is an eight-bit read/write register that selects timer synchronizing modes for channels 0–4. Channels for which 1 is set to the corresponding bit will be synchronized. TSNC is initialized to H'E0 or H'60 upon reset or standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | SYNC4 | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Initial value: | * | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | R/W | R/W | R/W | R/W | R/W |

Note: Undefined

- Bits 7–5 (reserved): Bit 7 is read as undefined. Bits 6 and 5 are always read as 1. The write value to bit 7 should be 0 or 1, and the write value to bits 6 and 5 should always be 1.
- Bit 4 (timer synchro 4 (SYNC4)): SYNC4 selects the synchronizing mode for channel 4.

| Bit 4: SYNC4 | Description |
|---|---|
| 0 | The timer counter for channel 4 (TCNT4) operates independently (Preset/clear of TCNT4 is independent of other channels) (initial value) |
| 1 | Channel 4 operates synchronously. Synchronized preset/clear of TNCT4 enabled. |

- Bit 3 (timer Synchro 3 (SYNC3)): SYNC3 selects the synchronizing mode for channel 3.

| Bit 3: SYNC3 | Description |
|---|---|
| 0 | The timer counter for channel 3 (TCNT3) operates independently (Preset/clear of TCNT3 is independent of other channels) (initial value) |
| 1 | Channel 3 operates synchronously. Synchronized preset/clear of TNCT3 enabled. |

- Bit 2 (timer synchro 2 (SYNC2)): SYNC2 selects the synchronizing mode for channel 2.

| Bit 2: SYNC2 | Description |
|---|---|
| 0 | The timer counter for channel 2 (TCNT2) operates independently (Preset/clear of TCNT2 is independent of other channels) (initial value) |
| 1 | Channel 2 operates synchronously. Synchronized preset/clear of TNCT2 enabled. |

- Bit 1 (timer synchro 1 (SYNC1)): SYNC1 selects the synchronizing mode for channel 1.

| Bit 1: SYNC1 | Description |
|---|---|
| 0 | The timer counter for channel 1 (TCNT1) operates independently (Preset/clear of TCNT1 is independent of other channels) (initial value) |
| 1 | Channel 1 operates synchronously. Synchronized preset/clear of TNCT1 enabled. |

- Bit 0 (timer synchro 0 (SYNC0)): SYNC0 selects the synchronizing mode for channel 0.

| Bit 0: SYNC0 | Description |
|---|---|
| 0 | The timer counter for channel 0 (TCNT0) operates independently (Preset/clear of TCNT0 is independent of other channels) (initial value) |
| 1 | Channel 0 operates synchronously. Synchronized preset/clear of TNCT0 enabled. |

### 10.2.3 Timer Mode Register (TMDR)

The timer mode register (TMDR) is an eight-bit read/write register that selects the PWM mode for channels 0–4, sets the phase counting mode for channel 2, and sets the conditions for the overflow flag (OVF). TMDR is initialized to H'80 or H'00 by a reset or the standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | MDF | FDIR | PWM4 | PWM3 | PWM2 | PWM1 | PWM0 |
| Initial value: | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: Undefined

- Bit 7 (reserved): Bit 7 is read as undefined. The write value should be 0 or 1.

- Bit 6 (phase counting mode (MDF)): MDF selects the phase counting mode for channel 2.

| Bit 6: MDF | Description |
|---|---|
| 0 | Channel 2 operates normally (initial value) |
| 1 | Channel 2 operates in phase counting mode |

When the MDF is set to 1 to select the phase counting mode, the timer counter (TCNT2) becomes an up/down counter and the TCLKA and TCLKB pins become count clock input pins. TCNT2 counts on both the rising and falling edges of TCLKA and TCLKB, with the increment/decrement chosen as follows:

| Count Direction | Decrement | | | | Increment | | | |
|---|---|---|---|---|---|---|---|---|
| TCLKA pin | Rising | High | Falling | Low | Rising | High | Falling | Low |
| TCLKB pin | L | Rising | High | Falling | High | Falling | Low | Rising |

In the phase counting mode, selections for external clock edge made in the CKEG1 and CKEG0 bits of the timer control register 2 (TCR2) and the selection for counter clock made in the TPSC2 –TPSC0 bits are ignored. The phase counting mode described above takes priority. Settings for counter clear conditions in the CCLR1 and CCLR0 bits of TCR2 and settings for timer I/O control register 2 (TIOR2), timer interrupt enable register (TIER2) and timer status register 2 (TSR2) compare match/input capture functions and interrupts, however, are valid even in the phase counting mode.

- Bit 5 (flag direction (FDIR)): FDIR selects the setting condition for the overflow flag (OVF) in timer status register 2 (TSR2). This bit is valid no matter which mode channel 2 is operating in.

| Bit 5: FDIR | Description |
|---|---|
| 0 | OVF of TSR2 is set to 1 when TCNT2 overflows or underflows (initial value) |
| 1 | OVF of TSR2 is set to 1 when TCNT2 overflows |

- Bit 4 (PWM Mode 4 (PWM4)): PWM4 selects the PWM mode for channel 4. When the PWM4 bit is set to 1 and the PWM mode entered, the TIOCA4 pin becomes a PWM output pin. 1 is output on a compare match of general register A4 (GRA4); 0 is output on a compare match of general register B4 (GRB4). When the complementary PWM mode or reset-synchronized PWM mode are set by the CMD1 and CMD0 bits of the timer function control register (TFCR), the setting of this bit is ignored in favor of the settings of CMD1 and CMD0.

| Bit 4: PWM4 | Description |
|---|---|
| 0 | Channel 4 operates normally (initial value) |
| 1 | Channel 4 operates in PWM mode |

- Bit 3 (PWM Mode 3 (PWM3)): PWM3 selects the PWM mode for channel 3. When the PWM3 bit is set to 1 and the PWM mode entered, the TIOCA3 pin becomes a PWM output pin. 1 is output on a compare match of general register A3 (GRA3); 0 is output on a compare match of general register B3 (GRB3). When the complementary PWM mode or reset-synchronized PWM mode are set by the CMD1 and CMD0 bits of the timer function control register (TFCR), the setting of this bit is ignored in favor of the settings of CMD1 and CMD0.

| Bit 3: PWM3 | Description |
|---|---|
| 0 | Channel 3 operates normally (initial value) |
| 1 | Channel 3 operates in PWM mode |

- Bit 2 (PWM Mode 2 (PWM2)): PWM2 selects the PWM mode for channel 2. When the PWM2 bit is set to 1 and the PWM mode entered, the TIOCA2 pin becomes a PWM output pin. 1 is output on a compare match of general register A2 (GRA2); 0 is output on a compare match of general register B2 (GRB2).

| Bit 2: PWM2 | Description |
|---|---|
| 0 | Channel 2 operates normally (initial value) |
| 1 | Channel 2 operates in PWM mode |

- Bit 1 (PWM Mode 1 (PWM1)): PWM1 selects the PWM mode for channel 1. When the PWM1 bit is set to 1 and the PWM mode entered, the TIOCA1 pin becomes a PWM output pin. 1 is output on a compare match of general register A1 (GRA1); 0 is output on a compare match of general register B1 (GRB1).

| Bit 1: PWM1 | Description |
|---|---|
| 0 | Channel 1 operates normally (initial value) |
| 1 | Channel 1 operates in PWM mode |

- Bit 0 (PWM Mode 0 (PWM0)): PWM0 selects the PWM mode for channel 0. When the PWM0 bit is set to 1 and the PWM mode entered, the TIOCA0 pin becomes a PWM output pin. 1 is output on a compare match of general register A0 (GRA0); 0 is output on a compare match of general register B0 (GRB0).

| Bit 0: PWM0 | Description |
|---|---|
| 0 | Channel 0 operates normally (initial value) |
| 1 | Channel 0 operates in PWM mode |

### 10.2.4 Timer Function Control Register (TFCR)

The timer function control register (TFCR) is an 8-bit read/write register that selects complementary PWM/reset-synchronized PWM for channels 3 and 4 and sets the buffer operation. TFCR is initialized on a reset or standby mode to H'C0 or H'40.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | CMD1 | CMD0 | BFB4 | BFA4 | BFB3 | BFA3 |
| Initial value: | * | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Note: Undefined

- Bits 7 and 6 (reserved): Bit 7 is read as undefined. Bit 6 is always read as 1. The write value to bit 7 should be 0 or 1. The write value to bit 6 should always be 1.

- Bits 5 and 4 (combination mode 1 and 0 (CMD1 and CMD0)): CMD1 and CMD0 select the complementary PWM mode or reset-synchronized mode for channels 3 and 4. Set the complementary PWM/reset-synchronized PWM mode while the timer counter (TCNT) being used is off. When these bits are used to set the complementary PWM/reset-synchronized PWM mode, they take priority over the PWM4 and PWM3 bits of the TMDR. While the complementary PWM/reset-synchronized PWM mode settings and the SYNC4 and SYNC3 bit settings of the timer synchro register (TSNC) are valid simultaneously, when the complementary PWM mode is set, channels 3 and 4 should not be set to operate simultaneously (SYNC 4 and SYNC 3 bits of TSNC should not both be set to 1).

| Bit 5: CMD1 | Bit 4: CMD0 | Description |
|---|---|---|
| 0 | 0 | Channels 3 and 4 operate normally (initial value) |
| | 1 | Channels 3 and 4 operate normally |
| 1 | 0 | Channels 3 and 4 operate together in complementary PWM mode |
| | 1 | Channels 3 and 4 operate together in reset-synchronized PWM mode |

- Bit 3 (buffer mode B4 (BFB4)): BFB4 selects the buffer mode for GRB4 and BRB4 in channel 4.

| Bit 3: BFB4 | Description |
|---|---|
| 0 | GRB4 operates normally in channel 4 (initial value) |
| 1 | GRB4 and BRB4 operate in buffer mode in channel 4 |

- Bit 2 (buffer mode A4 (BFA4)): BFA4 selects the buffer mode for GRA4 and BRA4 in channel 4.

| Bit 2: BFA4 | Description |
|---|---|
| 0 | GRA4 operates normally in channel 4 (initial value) |
| 1 | GRA4 and BRA4 operate in buffer mode in channel 4 |

- Bit 1 (buffer mode B3 (BFB3)): BFB3 selects the buffer mode for GRB3 and BRB3 in channel 3.

| Bit 1: BFB3 | Description |
|---|---|
| 0 | GRB3 operates normally in channel 3 (initial value) |
| 1 | GRB3 and BRB3 operate in buffer mode in channel 3 |

- Bit 0 (buffer Mode A3 (BFA3)): BFA3 selects the buffer mode for GRA3 and BRA3 in channel 3.

| Bit 0: BFA3 | Description |
|---|---|
| 0 | GRA3 operates normally in channel 3 (initial value) |
| 1 | GRA3 and BRA3 operate in buffer mode in channel 3 |

### 10.2.5 Timer Output Control Register (TOCR)

The timer output control register (TOCR) is an eight-bit read/write register that inverts the output level of the complementary PWM mode/reset-synchronized PWM mode. Setting bits OLS3 and OLS4 is valid in only the complementary PWM mode and reset-synchronized PWM mode. In other output situations, these bits are ignored. The TOCR is initialized to H'FF or H'7F by a reset or in the standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | — | OLS4 | OLS3 |
| Initial value: | * | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | — | — | — | — | — | — | R/W | R/W |

Note: Undefined

- Bits 7–2 (reserved): Bit 7 is read as undefined. Bits 6–2 are always read as 1. The write value to bit 7 should be 0 or 1. The write value to bits 6–2 should always be 1.

- Bit 1 (output level select 4 (OLS4)): OLS4 selects the output level of the complementary PWM mode or reset-synchronized PWM mode.

| Bit 1: OLS4 | Description |
|---|---|
| 0 | TIOCA3, TIOCA4, and TIOCB4 are inverted and output |
| 1 | TIOCA3, TIOCA4, and TIOCB4 are output directly (initial value) |

- Bit 0 (output level select 3 (OLS3)): OLS3 selects the output level of the complementary PWM mode or reset-synchronized PWM mode.

| Bit 0: OLS3 | Description |
|---|---|
| 0 | TIOCB3, TOCXA4, and TOCXB4 are inverted and output |
| 1 | TIOCB3, TOCXA4, and TOCXB4 are output directly (initial value) |

### 10.2.6 Timer Counters (TCNT)

The ITU has five 16-bit timer counters (TCNT), one for each channel (table 10.4).

Each TCNT is a 16-bit read/write counter that counts by input from a clock source. The clock source is selected by timer prescalar bits 2–0 (TPSC2–TPSC0) in the timer control register (TCR).

TCNT0 and TCNT 1 are strictly upcounters. Up/down counting occurs for TCNT2 when the phase counting mode is selected, or for TCNT3 and TCNT 4 when complementary PWM mode is selected. In other modes, they are upcounters.

The TCNT can be cleared to H'0000 by compare match with the corresponding general register A or B (GRA, GRB) or input capture to GRA or GRB (counter clear function).

When the TCNT overflows (changes from H'FFFF–H'0000), the overflow flag (OVF) in the timer status register (TSR) is set to 1. The OVF of the corresponding channel TSR is also set to 1 when the TCNT underflows (changes from H'0000–H'FFFF).

The TCNT is connected to the CPU by a 16-bit bus, so it can be written or read by either word access or byte access. The TCNT is initialized to H'0000 by a reset or in standby mode.

**Table 10.4 Timer Counters (TCNT)**

| Channel | Abbreviation | Function |
|---|---|---|
| 0 | TCNT0 | Increment counter |
| 1 | TCNT1 | |
| 2 | TCNT2 | Phase counting mode: Increment/decrement<br>All others: Increment |
| 3 | TCNT3 | Complementary PWM mode: Increment/decrement<br>All others: Increment |
| 4 | TCNT4 | |

| 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 |

### 10.2.7 General Registers A and B (GRA and GRB)

Each of the five ITU channels has two 16-bit general registers (GR) for a total of ten registers (table 10.5).

Each GR is a 16-bit read/write register that can function as either an output compare register or an input capture register. The function is selected by settings in the timer I/O control register (TIOR).

When a general register (GRA/GRB) is used as an output compare register, its value is constantly compared with the timer counter (TCNT) value. When the two values match (compare match), the IMFA/IMFB bit is set to 1 in the timer status register (TSR). If compare match output is selected in the TIOR, a specified value is output at the output compare pin.

When a general register is used as an input capture register, an external input capture signal is detected and the TCNT value is stored. The IMFA/IMFB bit of the corresponding TSR is set to 1 at the same time. The valid edge or edges of the input capture signal are selected in the TIOR. The TIOR setting is ignored when set for the PWM mode, complementary PWM mode or reset-synchronized PWM mode.

General registers are connected to the CPU by a 16-bit bus, so general registers can be written or read by either word access or byte access. General registers are initialized to the output compare register (no pin output) by a reset or in standby mode. The initial value is H'FFFF.

**Table 10.5 General Registers A and B (GRA and GRB)**

| Channel | Abbreviation | Function |
|---|---|---|
| 0 | GRA0, GRB0 | Output compare/input capture dual register |
| 1 | GRA1, GRB1 | |
| 2 | GRA2, GRB2 | |
| 3 | GRA3, GRB3 | Output compare/input capture dual register. Can also be set for buffer operation in combination with the buffer registers (BRA, BRB) |
| 4 | GRA4, GRB4 | |

| 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 |

### 10.2.8 Buffer Registers A and B (BRA, BRB)

Each buffer register is a 16-bit read/write register that is used in the buffer mode. The ITU has four buffer registers, two each for channels 3 and 4 (table 10.6). Buffer operation can be set independently by the timer function control register (TFCR) bits BFB4, BFA4, BFB3, and BFB3 bits. The buffer registers are paired with the general registers and their function changes automatically to match the function of its corresponding general register.

The buffer registers are connected to the CPU by a 16-bit bus, so they can be written or read by either word or byte access. Buffer registers are initialized to H'FFFF by a reset or in standby mode.

**Table 10.6 Buffer Registers A and B (BRA, BRB)**

| Channel | Abbreviation | Function |
|---|---|---|
| 3 | BRA3, BRB3 | When used for buffer operation:<br>When the corresponding GRA and GRB are output compare registers, the buffer registers function as output compare buffer registers that can automatically transfer the BRA and BRB values to GRA and GRB upon a compare match.<br>When the corresponding GRA and GRB are input capture registers, the buffer registers function as input capture buffer registers that can automatically transfer the values stored until an input capture in the GRA and GRB to the BRA and BRB. |
| 4 | BRA4, BRB4 | |

| 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 |

### 10.2.9 Timer Control Register (TCR)

The TCR is an 8-bit read/write register that selects the timer counter clock, the edges of the external clock source, and the counter clear source. Each ITU channel has one TCR. TCR is initialized H'80 or H'00 by a reset or the standby mode (table 10.7).

**Table 10.7 Timer Control Register (TCR)**

| Channel | Abbre-viation | Function |
|---|---|---|
| 0 | TCR0 | The TCR controls the TCNTs. The TCRs have the same functions on all channels. When channel 2 is set for phase counting mode, setting the CKEG1, CKEG2 and TPSC2–TPSC0 bits will have no effect. |
| 1 | TCR1 | |
| 2 | TCR2 | |
| 3 | TCR3 | |
| 4 | TCR4 | |

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

Note: Undefined

- Bit 7 (reserved): Bit 7 is read as undefined. The write value should be 0 or 1.
- Bits 6 and 5 (counter clear 1 and 0 (CCLR1 and CCLR0)): CCLR1 and CCLR0 select the counter clear source.

| Bit 6: CCLR1 | Bit 5: CCLR0 | Description |
|---|---|---|
| 0 | 0 | TCNT is not cleared (initial value) |
| | 1 | TCNT is cleared by general register A (GRA) compare match or input capture*1 |
| 1 | 0 | TCNT is cleared by general register B (GRB) compare match or input capture*1 |
| | 1 | Synchronizing clear: TCNT is cleared in synchronization with clear of other timer counters operating in sync.*2 |

Notes: 1. When GR is functioning as an output compare register, TCNT is cleared upon a compare match. When functioning as an input capture register, TCNT is cleared upon input capture.
2. The timer synchro register (TSNC) set the synchronization.

- Bits 4 and 3 (external clock edge 1/0 (CKEG1 and CKEG0)): CKEG1 and CKEG0 select external clock input edges. When channel 2 is set for phase counting mode, settings of the CKEG1 and CKEG0 of the TCR are ignored and the phase counting mode operation takes priority.

| Bit 4: CKEG 1 | Bit 3: CKEG 0 | Description |
|---|---|---|
| 0 | 0 | Count rising edges (initial value) |
| | 1 | Count falling edges |
| 1 | — | Count both rising and falling edges |

- Bits 2–0 (timer prescalar 2–0 (TPS2–TPS0)): TPS2–TPS0 select the counter clock source. When TPSC2 = 0 and an internal clock source is selected, the timer counts only falling edges. When TPSC2 = 1 and an external clock is selected, the count edge is as set by CKEG1 and CKEG0. When the phase counting mode is selected for channel 2 (MDF bit in the timer mode register is 1), the settings of TPSC2–TPSC0 of TCR2 are ignored and the phase counting operation takes priority.

| Bit 2: TPSC2 | Bit 1: TPSC1 | Bit 0: TPSC0 | Counter Clock (and cycle when $\phi$ = 10 MHz) |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock $\phi$ (initial value) |
| | | 1 | Internal clock $\phi$/2 |
| | 1 | 0 | Internal clock $\phi$/4 |
| | | 1 | Internal clock $\phi$/8 |
| 1 | 0 | 0 | External clock A (TCLKA) |
| | | 1 | External clock B (TCLKB) |
| | 1 | 0 | External clock C (TCLKC) |
| | | 1 | External clock D (TCLKD) |

### 10.2.10 Timer I/O Control Register (TIOR)

The timer I/O control register (TIOR) is an eight-bit read/write register that selects the output compare or input capture function for the general registers GRA and GRB. It also selects the function of the TIOCA and TIOCB pins. If output compare is selected, the TIOR also selects the output settings. If input capture is selected, the TIOR also select the input capture edges. TIOR is initialized to H'88 or H'08 on a reset or standby mode. Each ITU channel has one TIOR (table 10.8).

**Table 10.8 Timer I/O Control Register (TIOR)**

| Channel | Abbre-viation | Function |
|---|---|---|
| 0 | TIOR0 | The TIOR controls the GRs. Some functions vary during PWM. When channels 3 and 4 are set for complementary PWM mode/reset-synchronized PWM mode, TIOR3 and TIOR4 settings are not valid. |
| 1 | TIOR1 | |
| 2 | TIOR2 | |
| 3 | TIOR3 | |
| 4 | TIOR4 | |

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

Note: Undefined

- Bit 7 (reserved): Bit 7 is read as undefined. The write value should be 0 or 1.

- Bits 6–4 (I/O control B2–B0 (IOB2–IOB0)): IOB2–IOB0 selects the GRB function.

| Bit 6: IOB2 | Bit 5: IOB1 | Bit 4: IOB0 | GRB Function | |
|---|---|---|---|---|
| 0 | 0 | 0 | GRB is an output compare register | Compare match with pin output disabled (initial value) |
| | | 1 | | 0 output at GRB compare match*1 |
| | 1 | 0 | | 1 output at GRB compare match*1 |
| | | 1 | | Output toggles at GRB compare match (1 output for channel 2 only)*1,*2 |
| 1 | 0 | 0 | GRB is an input capture register | GRB captures rising edge of input |
| | | 1 | | GRB captures falling edge of input |
| | 1 | 0 | | GRB captures both edges of input |
| | | 1 | | |

Notes: 1. After reset, the value output is 0 until the first compare match occurs.
2. Channel 2 has no compare-match driven toggle output function. If it is set for toggle, 1 is automatically selected as the output.

- Bit 3 (reserved): Bit 3 always reads as 1. The write value should always be 1.

- Bits 2–0 (I/O control A2–A0 (IOA2–IOA0)): IOA2–IOA0 select the GRB function.

| Bit 2: IOA2 | Bit 1: IOA1 | Bit 0: IOA0 | GRA Function | |
|---|---|---|---|---|
| 0 | 0 | 0 | GRA is an output compare register | Compare match with pin output disabled (initial value) |
| | | 1 | | 0 output at GRA compare match*1 |
| | 1 | 0 | | 1 output at GRA compare match*1 |
| | | 1 | | Output toggles at GRA compare match (1 output for channel 2 only)*1,*2 |
| 1 | 0 | 0 | GRA is an input capture register | GRA captures rising edge of input |
| | | 1 | | GRA captures falling edge of input |
| | 1 | 0 | | GRA captures both edges of input |
| | | 1 | | |

Notes: 1. After reset, the value output is 0 until the first compare match occurs.
2. Channel 2 has no compare-match driven toggle output function. If it is set for toggle, 1 is automatically selected as the output.

### 10.2.11 Timer Status Register (TSR)

The timer status register (TSR) is an eight-bit read/write register containing flags that indicate timer counter (TCNT) overflow/underflow and general register (GRA/GRB) compare match or input capture. These flags are interrupt sources. If the interrupt is enabled by the corresponding bit in the timer interrupt enable register (TIER), an interrupt is requested of the CPU. TSR is initialized by a reset or standby mode to H'F8 or H'78. Each ITU channel has one TSR (table 10.9).

**Table 10.9 Timer Status Register (TSR)**

| Channel | Abbreviation | Function |
|---|---|---|
| 0 | TSR0 | The TSR indicates input capture, compare match and overflow status. |
| 1 | TSR1 | |
| 2 | TSR2 | |
| 3 | TSR3 | |
| 4 | TSR4 | |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | OVF | IMFB | IMFA |
| Initial value: | *1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| R/W: | — | — | — | — | — | R/(W)*2 | R/(W)*2 | R/(W)*2 |

Notes: 1. Undefined
2. Write 0 to clear the flag.

- Bits 7–3 (reserved): Bit 7 is read as undefined. Bits 6–3 are always read as 1. The write value to bit 7 should be 0 or 1. The write value to bits 6–3 should always be 1.

- Bit 2 (overflow flag (OVF)): OVF indicates a TCNT overflow/underflow has occurred.

| Bit 2: OVF | Description |
|---|---|
| 0 | Clearing condition: Read OVF when OVF = 1, then write 0 in OVF (initial value) |
| 1 | Setting condition: TCNT overflowed from H'FFFF–H'0000 or underflowed from H'0000–H'FFFF. |

Note: A TCNT underflow occurs when the TCNT up/down counter is functioning. It may occur in the following cases: (1) When channel 2 is set in the phase counting mode (MDF bit of TMDR is 1), or (2) When channel 3 and 4 are set to the complementary PWM mode (CMD1 bit of TFCR is 1 and CMD0 bit is 0).

- Bit 1 (input capture/compare match B (IMFB)): IMFB indicates a GRB compare match or input capture.

| Bit 1: IMFB | Description |
|---|---|
| 0 | Clearing condition: Read IMFB when IMFB = 1, then write 0 in IMFB (initial value) |
| 1 | Setting condition:<br>• GRB is functioning as an output compare register and TCNT = GRB<br>• GRB is functioning as an input capture register and the value of TCNT is transferred to GRB by an input capture signal |

- Bit 0 (input capture/compare match A (IMFA)): IMFA indicates a GRA compare match or input capture.

| Bit 0: IMFA | Description |
|---|---|
| 0 | Read IMFA when IMFA = 1, then write 0 in IMFA (initial value).<br>Clearing condition: DMAC is activated by an IMIA interrupt (only channels 0–3) |
| 1 | Setting condition:<br>• GRA is functions as an output compare register and TCNT = GRA<br>• GRA is functioning as an input capture register and the value of TCNT is transferred to GRA by an input capture signal |

### 10.2.12 Timer Interrupt Enable Register (TIER)

The timer status interrupt enable register (TIER) is an eight-bit read/write register that controls enabling/disabling of overflow interrupt requests and general register compare match/input capture interrupt requests. TIER is initialized by a reset or standby mode to H'F8 or H'78. Each ITU channel has one TIER (table 10.10).

**Table 10.10 Timer Interrupt Enable Register (TIER)**

| Channel | Abbreviation | Function |
|---|---|---|
| 0 | TIER0 | The TIER controls interrupt enable/disable. |
| 1 | TIER1 | |
| 2 | TIER2 | |
| 3 | TIER3 | |
| 4 | TIER4 | |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | OVIE | IMIEB | IMIEA |
| Initial value: | * | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| R/W: | — | — | — | — | — | R/W | R/W | R/W |

Note: Undefined

- Bits 7–3 (reserved): Bit 7 is read as undefined. Bits 6–3 are always read as 1. The write value to bit 7 should be 0 or 1. The write value to bits 6–3 should always be 1.

- Bit 2 (overflow interrupt enable (OVIE)): When the TSR overflow flag (OVF) is set to 1, OVIE enables or disables interrupt requests from the OVF.

| Bit 2: OVIE | Description |
|---|---|
| 0 | Disables interrupt requests by the OVF (initial value) |
| 1 | Enables interrupt requests from the OVF |

- Bit 1 (input capture/compare match interrupt enable B (IMIEB)): When the IMFB bit of the TSR is set to 1, IMIEB enables or disables the interrupt requests from the IMFB.

| Bit 1: IMIEB | Description |
|---|---|
| 0 | Disables interrupt requests by the IMFB (IMIB) (initial value) |
| 1 | Enables interrupt requests from the IMFB (IMIB) |

- Bit 0 (input capture/compare match interrupt enable A (IMIEA)): When the IMFA bit of the TSR is set to 1, IMIEA enables or disables the interrupt requests from the IMFA.

| Bit 0: IMIEA | Description |
|---|---|
| 0 | Disables interrupt requests by the IMFA (IMIA) (initial value) |
| 1 | Enables interrupt requests from the IMFA (IMIA) |

## 10.3 CPU Interface

### 10.3.1 16-Bit Accessible Registers

The timer counters (TCNT), general registers A and B (GRA, GRB), and buffer registers A and B (BRA, BRB) are 16-bit registers. The SH CPU can access these registers a word at a time using a 16-bit data bus. Byte access is also possible. Read and write operations performed on the TCNT in word units are shown in figures 10.6 and 10.7. Byte-unit read and write operations on TCNTH and TCNTL are shown in figures 10.8–10.11.

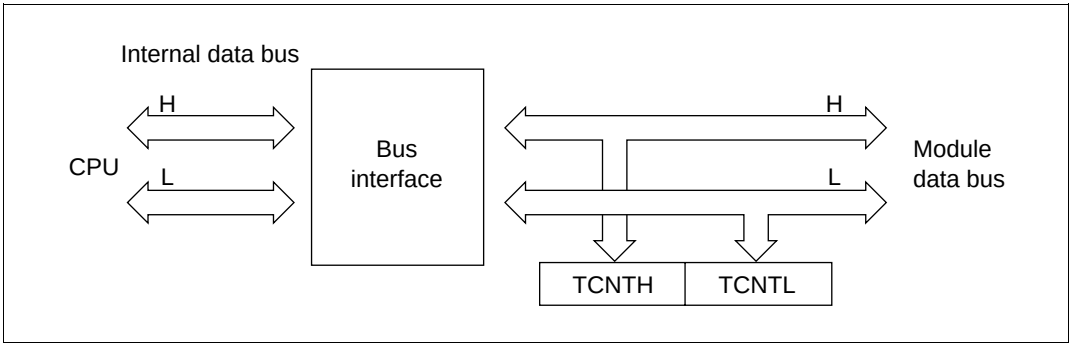



**Figure 10.6 Accessing TCNT (CPU–TCNT (word))**

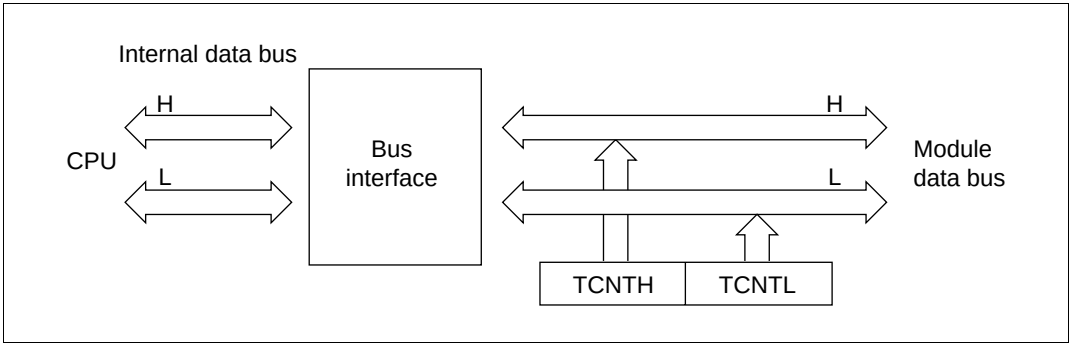



**Figure 10.7 Accessing TCNT (TCNT–CPU (word))**

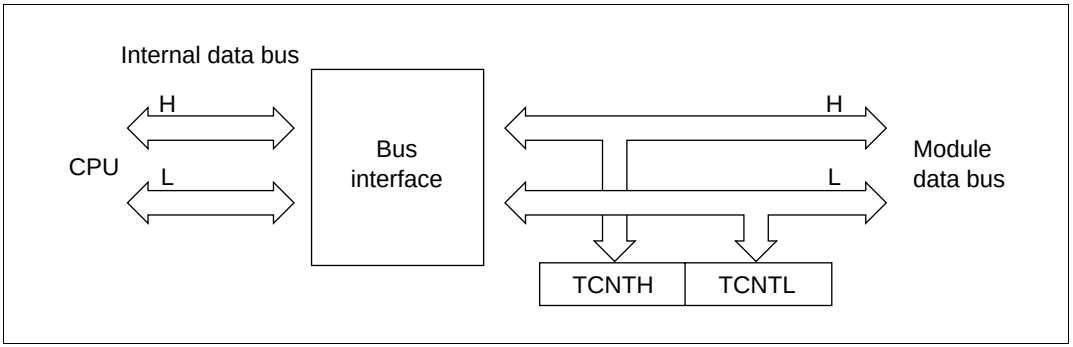



**Figure 10.8 Accessing TCNT (CPU–TCNT (upper byte))**

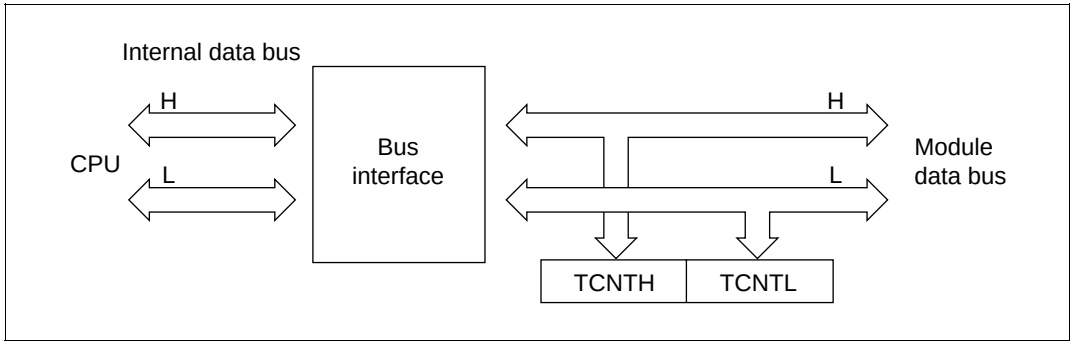



**Figure 10.9 Accessing TCNT (CPU–TCNT (lower byte))**

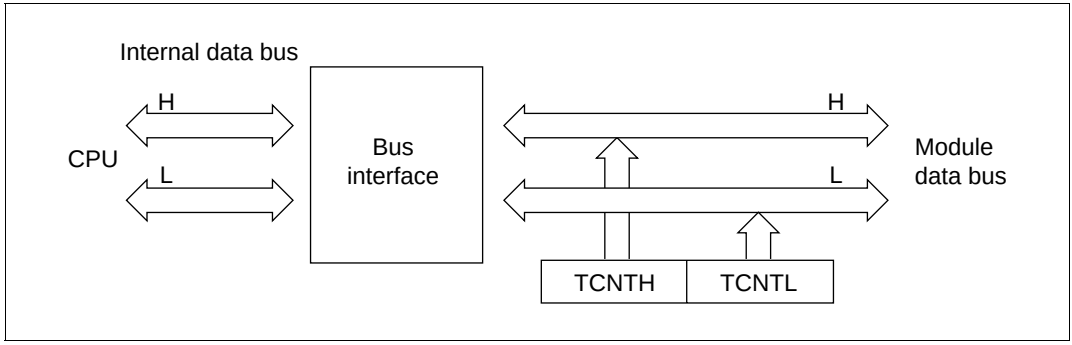



**Figure 10.10 Accessing TCNT (TCNT–CPU (upper byte))**

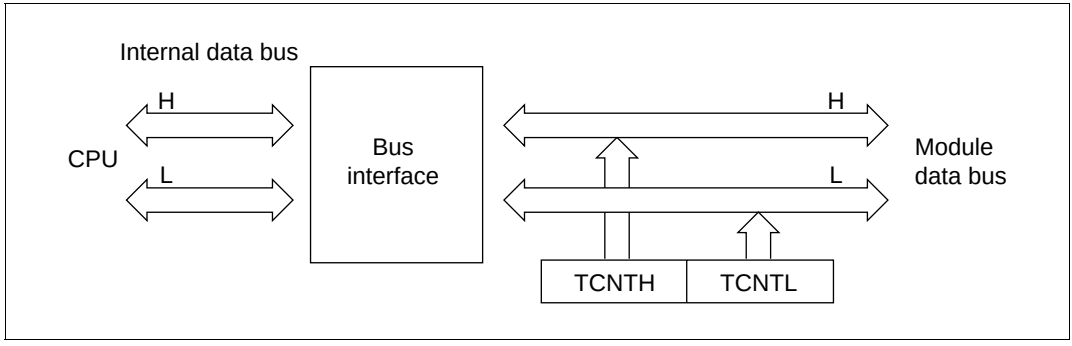



**Figure 10.11 Accessing TCNT (TCNT–CPU (lower byte))**

### 10.3.2 8-Bit Accessible Registers

All registers other than the TCNT, general registers, and buffer registers are 8-bit registers. These are connected to the CPU by an 8-bit data bus. Figures 10.12 and 10.13 illustrate reading and writing in byte units with the timer control register (TCR). These registers must be accessed by byte access.

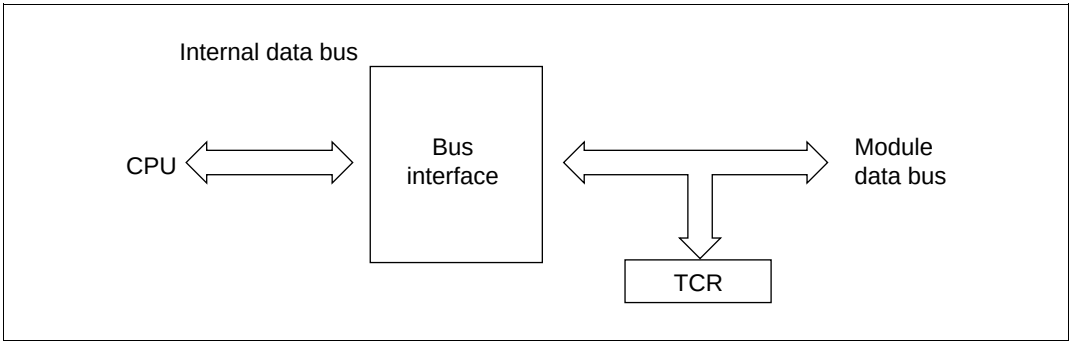



**Figure 10.12 TCR Access (CPU–TCR)**

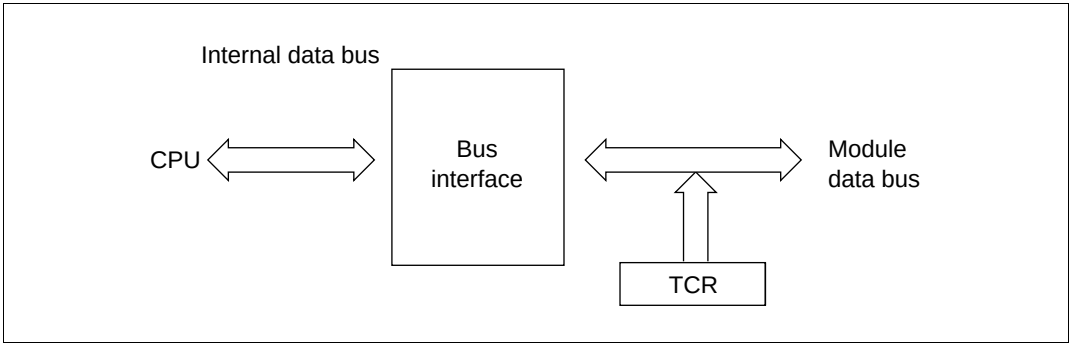



**Figure 10.13 TCR Access (TCR–CPU )**

## 10.4 Description of Operation

### 10.4.1 Overview

The operation modes are described below.

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

**Synchronized Operation:** The TCNT of a channel set for synchronized operation does a synchronized preset. When any TCNT of a channel operating in the synchronized mode is rewritten, the TCNTs of other channels are simultaneously rewritten as well. The CCLR1 and CCLR0 bits of the timer control register of multiple channels set for synchronous operation can be set to clear the TCNTs simultaneously.

**PWM Mode:** In PWM mode, a PWM waveform is output from the TIOCA pin. Output becomes 1 upon compare match A and 0 upon compare match B. GRA and GRB can be set so that the PWM waveform output has a duty cycle between 0% and 100%. When set for PWM mode, the GRA and GRB automatically become output compare registers.

**Reset-synchronized PWM Mode:** Three pairs of positive and negative PWM waveforms can be obtained using channels 3 and 4 (the three phases of the PWM waveform share a transition point on one side). When set for reset-synchronized PWM mode, GRA3, GRB3, GRA4, and GRB4 automatically become output compare registers. The TIOCA3, TIOCB3, TIOCA4, TOCXA4, TIOCB4, and TOCXB4 pins also automatically become PWM output pins and TCNT3 becomes an upcounter. TCNT4 functions independently (although GRA and GRB are isolated from TCNT4).

**Complementary PWM Mode:** Three pairs of complementary positive and negative PWM waveforms whose positive and negative phases do not overlap can be obtained using channels 3 and 4. When set for complementary PWM mode, GRA3, GRB3, GRA4, and GRB4 automatically become output compare registers. The TIOCA3, TIOCB3, TIOCA4, TOCXA4, TIOCB4, and TOCXB4 pins also automatically become PWM output pins while TCNT3 and TCNT4 become upcounters.

**Phase Counting Mode:** In phase counting mode, the phase differential between two clocks input from the TCLKA and TCLKB pins is detected and the TCNT2 operates as an up/downcounter. In phase counting mode, the TCLKA and TCLKB pins become clock inputs and TCNT2 functions as an up/downcounter.

**Buffer Mode:**

- When GR is an output compare register: The BR value of each channel is transferred to the GR when a compare match occurs.
- When GR is an input capture register: The TCNT value is transferred to the GR when an input capture occurs and simultaneously the value previously stored in the GR is transferred to the BR.
- Complementary PWM mode: When the TCNT3 and TCNT4 change count directions, the BR value is transferred to the GR.
- Reset-synchronized PWM mode: The BR value is transferred to GR upon a GRA3 compare match.

### 10.4.2 Basic Functions

**Counter Operation:** When a start bit (STR0–STR4) in the timer start register (TSTR) is set to 1, the corresponding timer counter (TCNT) starts counting. There are two counting modes: a free-running mode and a periodic mode.

- Procedure for selecting counting mode (figure 10.14):

1. Set bits TPSC2–TPSC0 in the TCR to select the counter clock source. If an external clock source is selected, set bits CKEG1 and CKEG0 in the TCR to select the desired edges of the external clock signal.
2. To operate as a periodic counter, set CCLR1 and CCLR0 in the TCR to select whether to clear the TCNT at GRA compare match or GRB compare match.
3. Set the GRA or GRB selected in step 2 as an output compare register using the timer I/O control register (TIOR).
4. Write the desired cycle value in the GRA or GRB selected in step 1.
5. Set the STR bit in the TSTR to 1 to start counting.

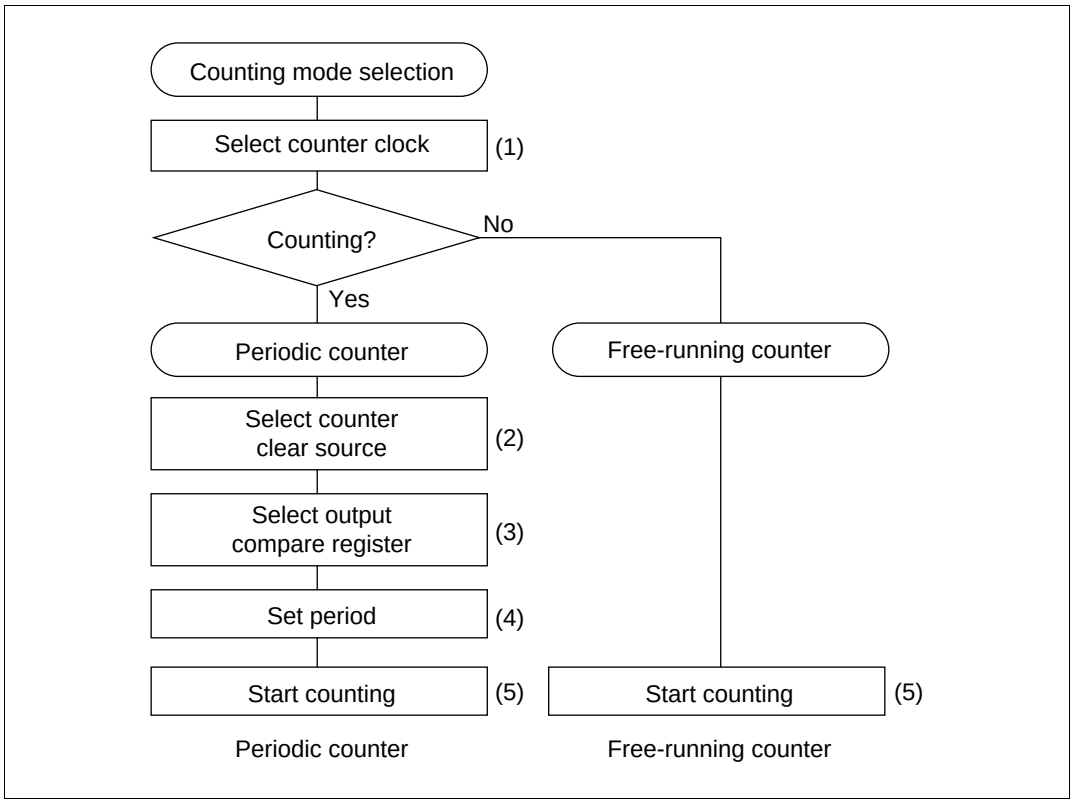



**Figure 10.14 Procedure for Selecting the Counting Mode**

- Free-running count and periodic count

A reset of the counters for channels 0–4 leaves them all in the free-running mode. When a corresponding bit in the TSTR is set to 1, the corresponding timer counter operates as a free-running counter and begins to increment. When the count wraps around from H'FFFF–H'0000, the overflow flag (OVF) in the timer status register (TSR) is set to 1. If the OVIE bit in the timer's corresponding interrupt enable register (TIER) is set to 1, the CPU will be asked for an interrupt. After the TCNT overflows, counting continues from H'0000. Figure 10.15 shows an example of free-running counting.

Periodic counter operation is obtained for a given channel's TCNT by selecting compare match as a TCNT clear source. (Set the GRA or GRB for period setting to output compare register and select counter clear upon compare match using the CCLR1 and CCLR0 bits of the timer control register (TCR).) After setting, the TCNT begins incrementing as a periodic counter when the corresponding bit of TSTR is set to 1. When the count matches GRA or GRB, the IMFA/IMFB bit in the TSR is set to 1 and the counter is automatically cleared to H'0000. If the IMIEA/IMIEB bit of the corresponding TIER is set to 1 at this point, the CPU will be asked for an interrupt. After the compare match, TCNT continues counting from H'0000. Figure 10.16 shows an example of periodic counting.

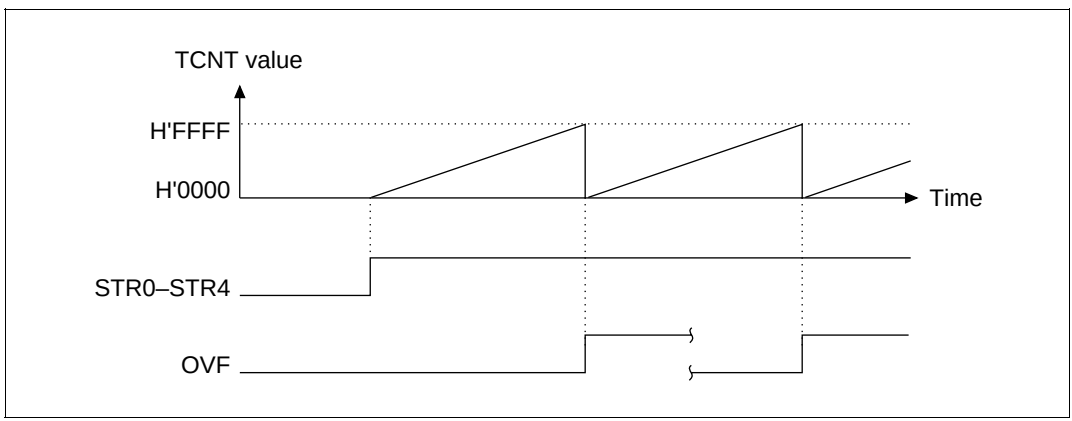



**Figure 10.15 Free-Running Counter Operation**

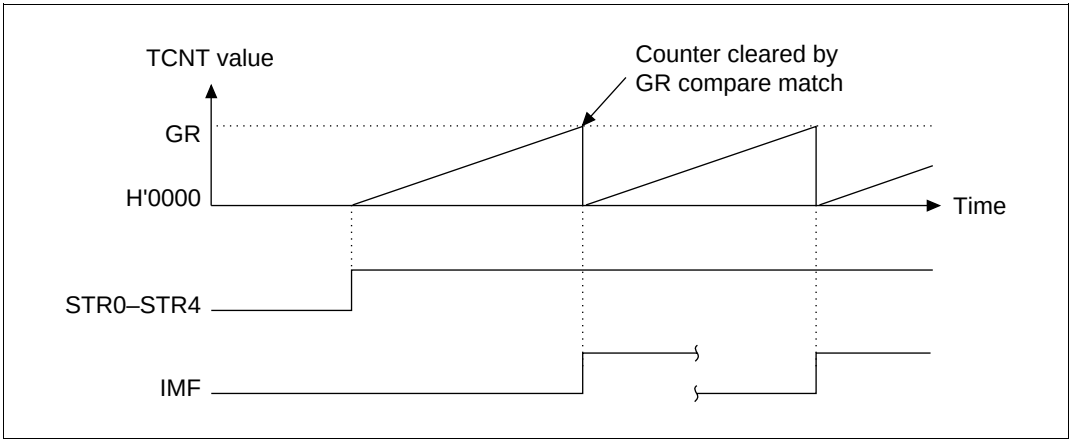



**Figure 10.16 Periodic Counter Operation**

- TCNT counter timing

  Internal clock source: Bits TPSC2–TPSC0 in the TCR select the system clock (CK) or one of three internal clock sources (ϕ/2, ϕ/4, ϕ/8) obtained by prescaling the system clock. Figure 10.17 shows the timing.

  External clock source: The external clock input pin (TCLKA–TCLKD) source is selected by bits TPSC2–TPSC0 in the TCR and its valid edges are selected with the CKEG1 and CKEG0 bits of the TCR. The rising edge, falling edge, or both edges can be selected. The pulse width of the external clock signal must be at least 1.5 system clocks when a single edge is selected and at least 2.5 system clocks when both edges are selected. Shorter pulses will not be counted correctly. Figure 10.18 shows the timing when both edges are detected.

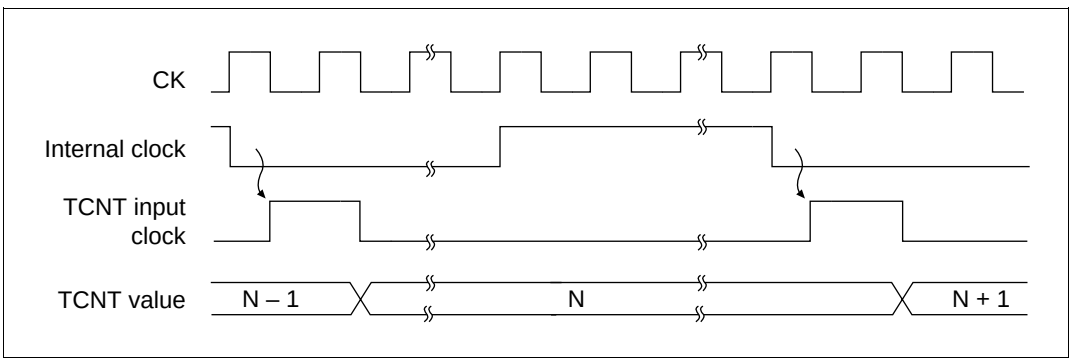



**Figure 10.17 Count Timing for Internal Clock Sources**

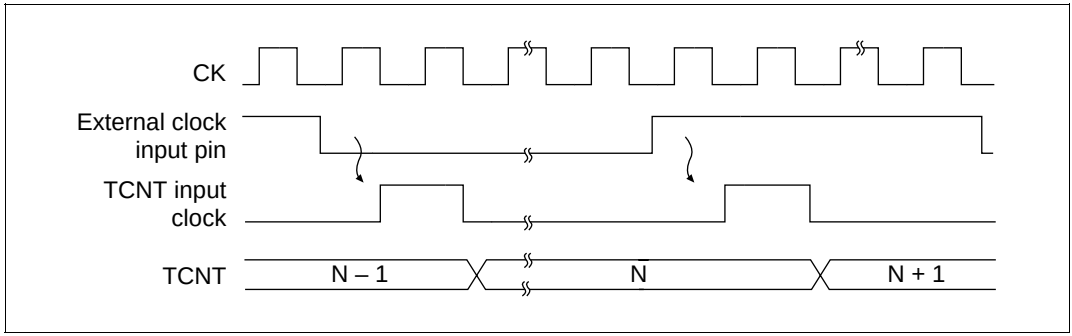



**Figure 10.18 Count Timing for External Clock Sources**

**Compare-Match Waveform Output Function:** For ITU channels 0, 1, 3, and 4, the output from the corresponding TIOCA and TIOCB pins upon compare matches A and B can be in three modes: 0-level output, 1-level output, or toggle. Toggle output cannot be selected in channel 2.

- Procedure for selecting the waveform output mode (figure 10.19):

1. Set the TIOR to select 0 output, 1 output, or toggle output for compare match output. The compare match output pin will output 0 until the first compare match occurs.
2. Set a value in the GRA or GRB to select the compare match timing.
3. Set the STR bit in the TSTR to 1 to start counting.

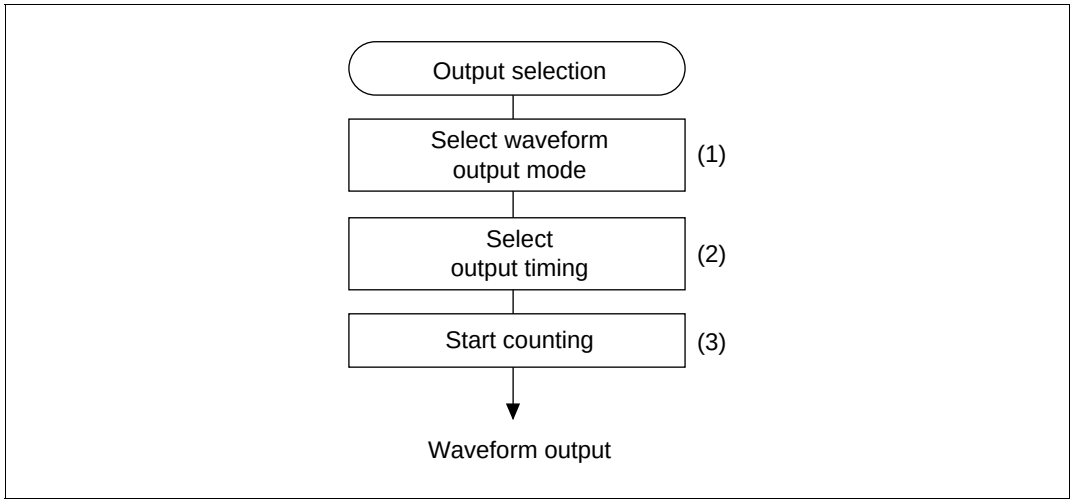



**Figure 10.19 Procedure for Selecting the Compare Match Waveform Output Mode**

- Waveform output operation

Figure 10.20 illustrates 0 output/1 output. In the example, TCNT is a free-running counter, 0 is output upon compare match A and 1 is output upon compare match B. When the pin level matches the set level, the pin level does not change.

Figure 10.21 shows an example of toggle output. In the figure, the TCNT operates as a periodic counter cleared by GRB compare match with toggle output at both compare match A and compare match B.

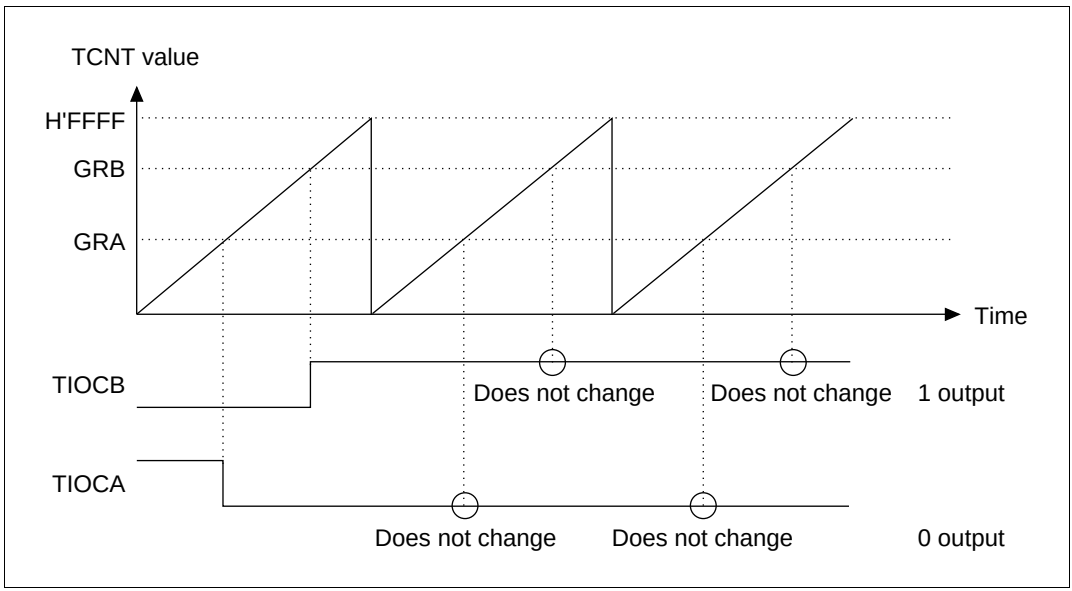



**Figure 10.20 Example of 0 Output/1 Output**

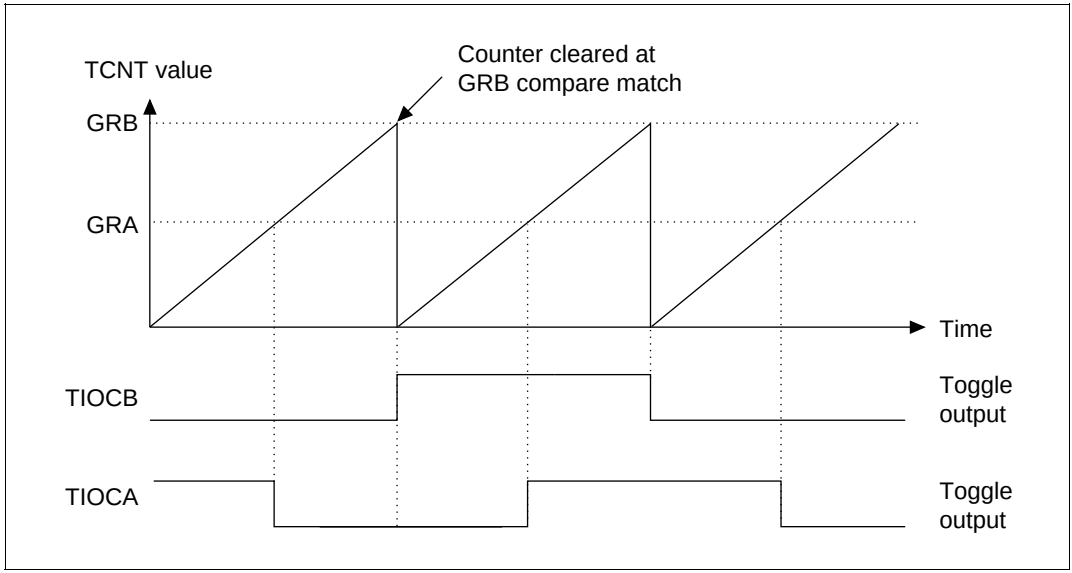



**Figure 10.21 Example of Toggle Output**

- Compare match output timing

  The compare match signal is generated in the last state in which the TCNT and the general register match (when the TCNT changes from the matching value to the next value). When a compare match signal is generated, the output value set in TIOR is output to the output compare pin (TIOCA, TIOCB). Accordingly, when the TCNT matches a general register, the compare match signal is not generated until the next counter clock pulse. Figure 10.22 shows the output timing of the compare match signal.

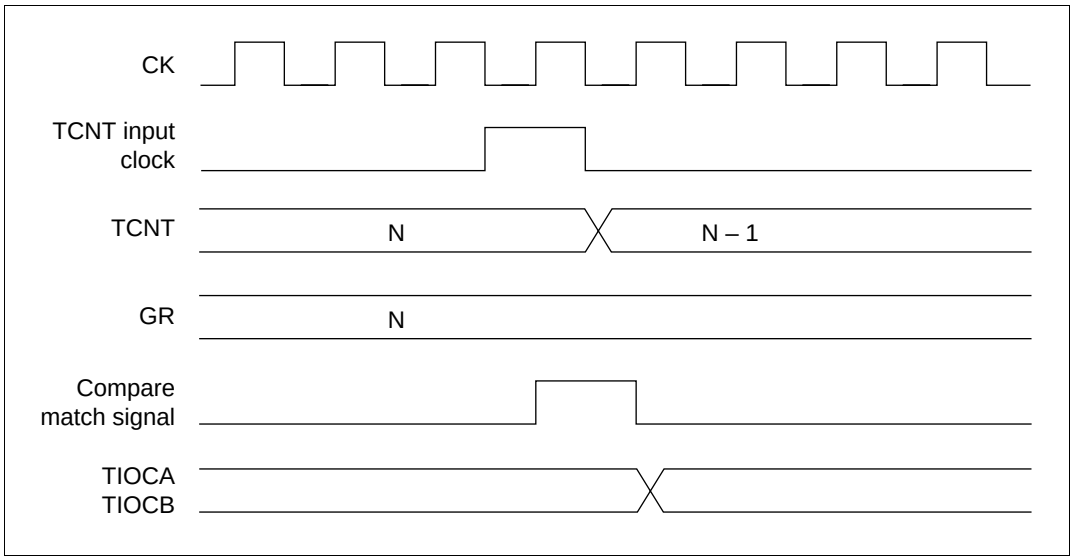



**Figure 10.22 Compare Match Signal Output Timing**

**Input Capture Mode:** In the input capture mode, the counter value is captured into a general register when the input edge is detected at an input capture/output compare pin (TIOCA, TIOCB). Detection can take place on the rising edge, falling edge, or both edges. Pulse width and cycle can be measured by using the input capture function.

- Procedure for selecting the input capture mode (figure 10.23)

1. Set the TIOR to select the input capture function of the GR and select the rising edge, falling edge, or both edges as the input edge of the input capture signal. Put the corresponding port into input-capture using the pin function controller before setting the TIOR.

2. Set the STR bit in the TSTR to 1 to start the TCNT counting.

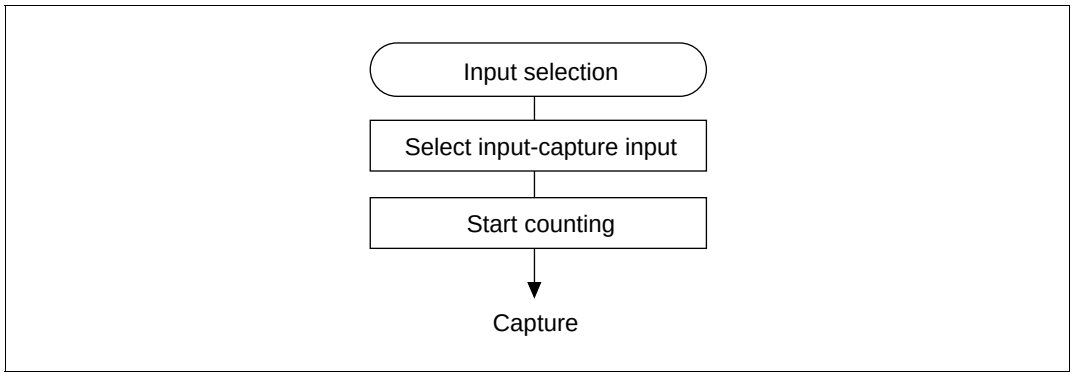



**Figure 10.23 Procedure for Selecting Input Capture Mode**

- Input capture operation

  Figure 10.24 illustrates input capture. The falling edge of TIOCB and both edges of TIOCA are selected as input capture edges. In the example, TCNT is set to clear at the input capture of GRB.

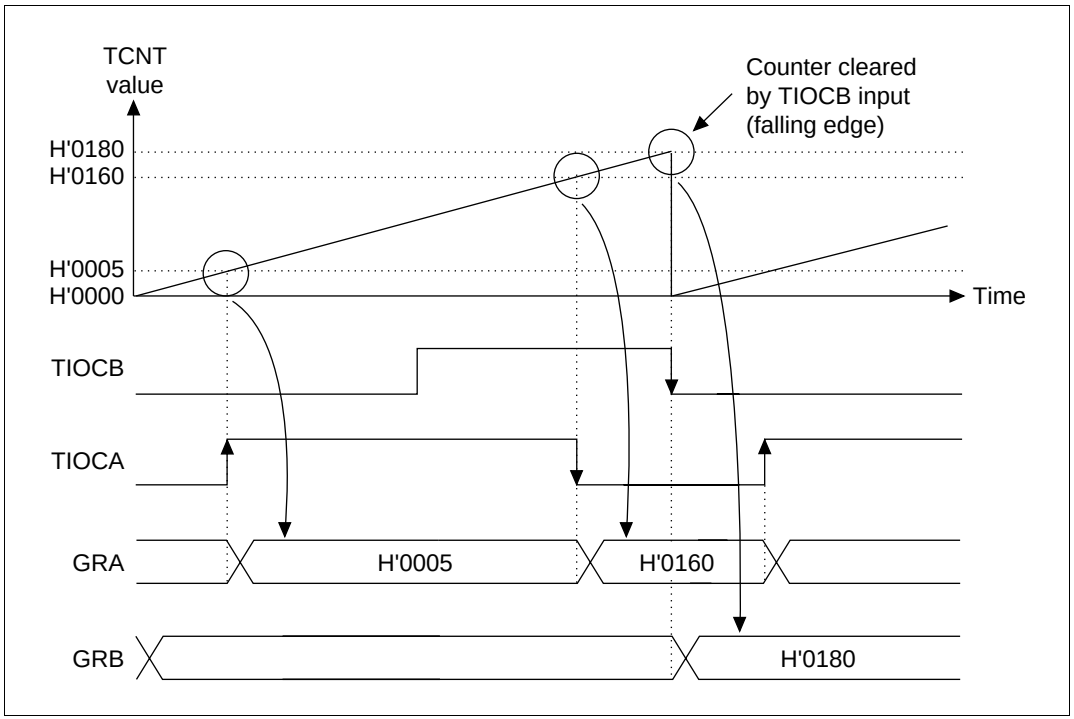



**Figure 10.24 Input Capture Operation**

- Input capture timing

  Input capture on the rising edge, falling edge, or both edges can be selected by settings in the TIOR. Figure 10.25 shows the timing when the rising edge is selected. The pulse width of the input capture signal must be at least 1.5 system clocks for single-edge capture, and 2.5 system clocks for capture of both edges.

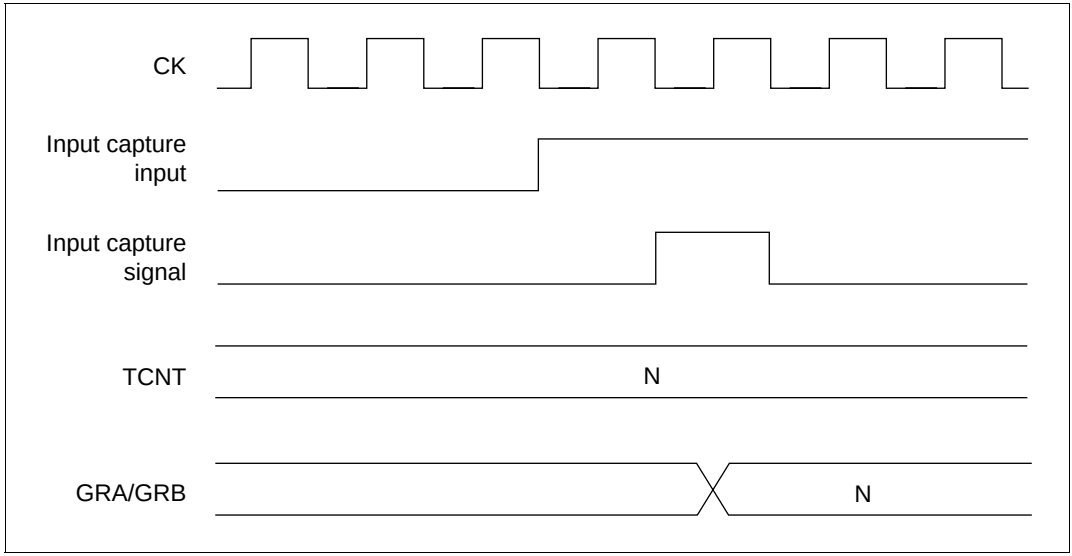



**Figure 10.25 Input Capture Signal Timing**

### 10.4.3 Synchronizing Mode

In the synchronizing mode, two or more timer counters can be rewritten simultaneously (synchronized preset). Multiple timer counters can also be cleared simultaneously using TCR settings (synchronized clear). The synchronizing mode can increase the general registers for a single time base. All five channels can be set for synchronous operation.

**Procedure for Selecting the Synchronizing Mode (figure 10.26):**

1. Set 1 in the SYNC bit of the timer synchro register (TSNC) to use the channels in the synchronizing mode.
2. When a value is written in the TCNT in any of the synchronized channels, the same value is simultaneously written in the TCNT in the other channels.
3. Set the counter to clear with compare match/input capture using bits CCLR1 and CCLR0 in the TCR.
4. Set the counter clear source to synchronized clear using the CCLR1 and CCLR0 bits.
5. Set the STR bits in the TSTR to 1 to start counting in the TCNT.

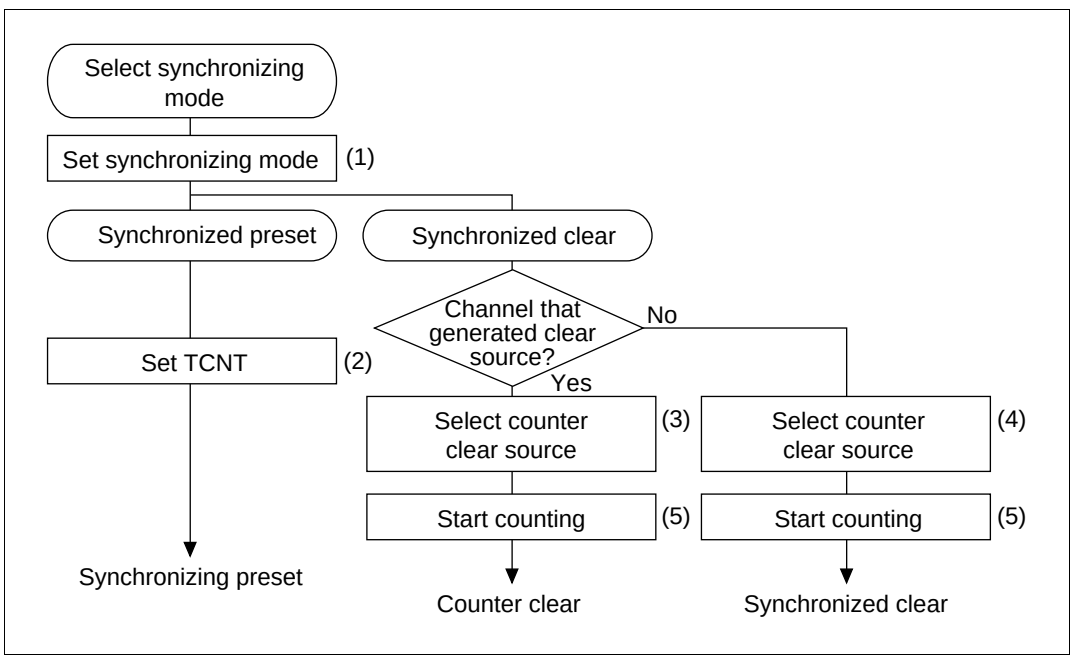



**Figure 10.26 Procedure for Selecting the Synchronizing Mode**

**Synchronized Operation:** Figure 10.27 shows an example of synchronized operation. Channels 0, 1, and 2 are set to synchronized operation and PWM output. Channel 0 is set for a counter clear upon compare match with GRB0. Channels 1 and 2 are set for counter clears by synchronizing clears. Accordingly, their timers are sync preset, then sync cleared by a GRB0 compare match, and then a three-phase PWM waveform is output from the TIOCA0, TIOCA1 and TIOCA2 pins. See section 10.4.4, PWM Mode, for details on the PWM mode.

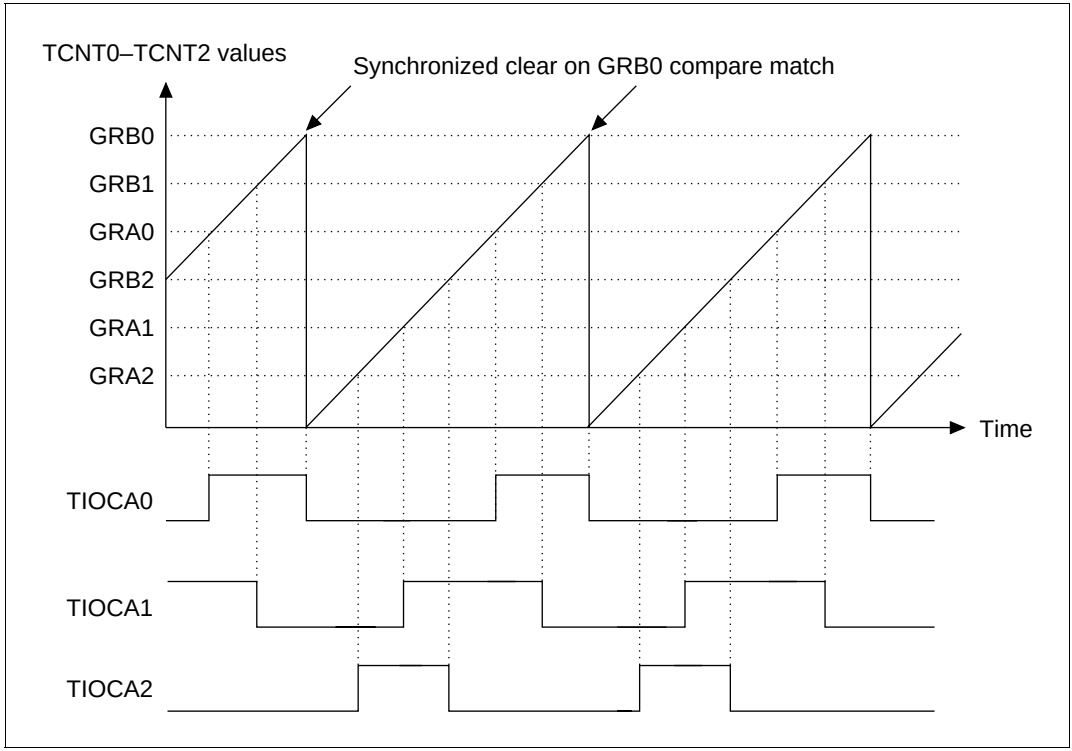



**Figure 10.27 Synchronized Operation Example**

### 10.4.4 PWM Mode

The PWM mode is controlled using both the GRA and GRB in pairs. The PWM waveform is output from the TIOCA output pin. The PWM waveform's 1 output timing is set in GRA and the 0 output timing is set in GRB. A PWM waveform with duty cycle between 0% and 100% can be output from the TIOCA pin by having either compare match GRA or GRB be the counter clear source for the timer counter. All five channels can be set to PWM mode.

Table 10.11 lists the combinations of PWM output pins and registers. Note that when the GRA and GRB are set to the same value, the output will not change even if a compare match occurs.

**Table 10.11 Combinations of PWM Output Pins and Registers**

| Channel | Output Pin | 1 Output | 0 Output |
|---|---|---|---|
| 0 | TIOCA0 | GRA0 | GRB0 |
| 1 | TIOCA1 | GRA1 | GRB1 |
| 2 | TIOCA2 | GRA2 | GRB2 |
| 3 | TIOCA3 | GRA3 | GRB3 |
| 4 | TIOCA4 | GRA4 | GRB4 |

**Procedure for Selecting the PWM Mode (figure 10.28):**

1. Set bits TPSC2–TPSC0 in the TCR to select the counter clock source. If an external clock source is selected, set bits CKEG1 and CKEG0 in the TCR to select the desired edges of the external clock signal.
2. Set CCLR1 and CCLR0 in the TCR to select the counter clear source.
3. Set the time at which the PWM waveform should go to 1 in the GRA.
4. Set the time at which the PWM waveform should go to 0 in the GRB.
5. Set the PWM bit in TMDR to select the PWM mode. When the PWM mode is selected, regardless of the contents of TIOR, the GRA and GRB become output compare registers specifying the times at which the PWM waveform goes high and low. TIOCA automatically becomes a PWM output pin. TIOCB becomes whatever is set in the TIOR's IOB1 and IOB0 bits.
6. Set the STR bit in the TSTR to let the TCNT start counting.

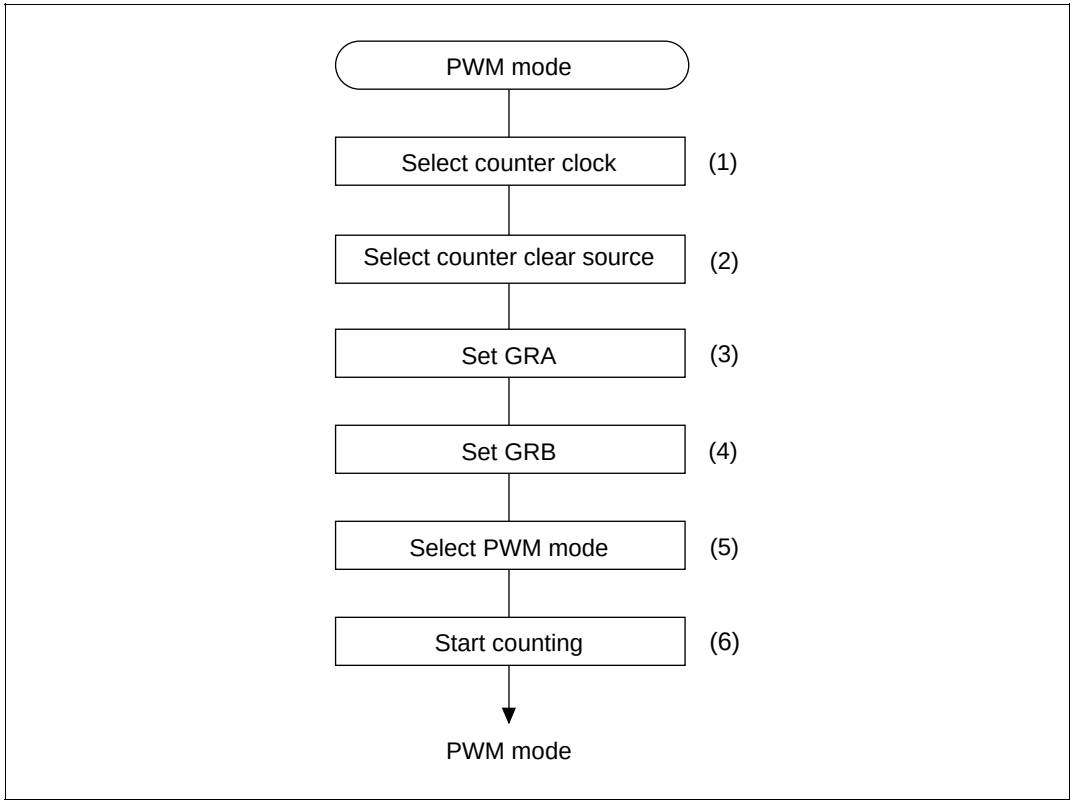



**Figure 10.28 Procedure for Selecting the PWM Mode**

**PWM Mode Operation:** Figure 10.29 illustrates PWM mode operations. When the PWM mode is set, the TIOCA pin becomes the output pin. Output is 1 when the TCNT matches the GRA, and 0 when the TCNT matches the GRB. The TCNT can be cleared by compare match with either GRA or GRB. This can be used in both free-running and synchronized operation.

Figure 10.30 shows examples of PWM waveforms output with 0% and 100% duty cycles. A 0% duty waveform can be obtained by setting the counter clear source to GRB and then setting GRA to a larger value than GRB. A 100% duty waveform can be obtained by setting the counter clear source to GRA and then setting GRB to a larger value than GRA.

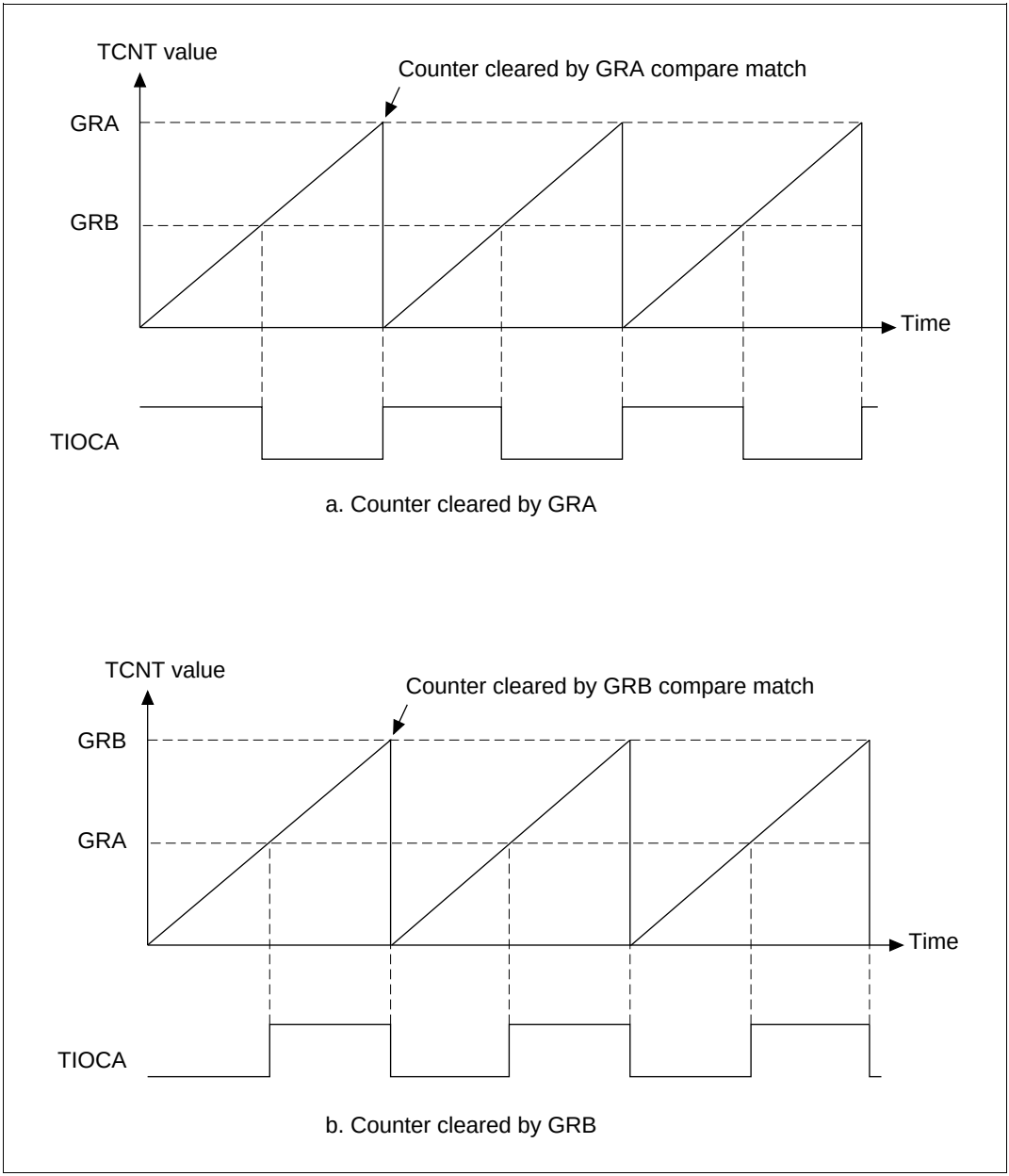
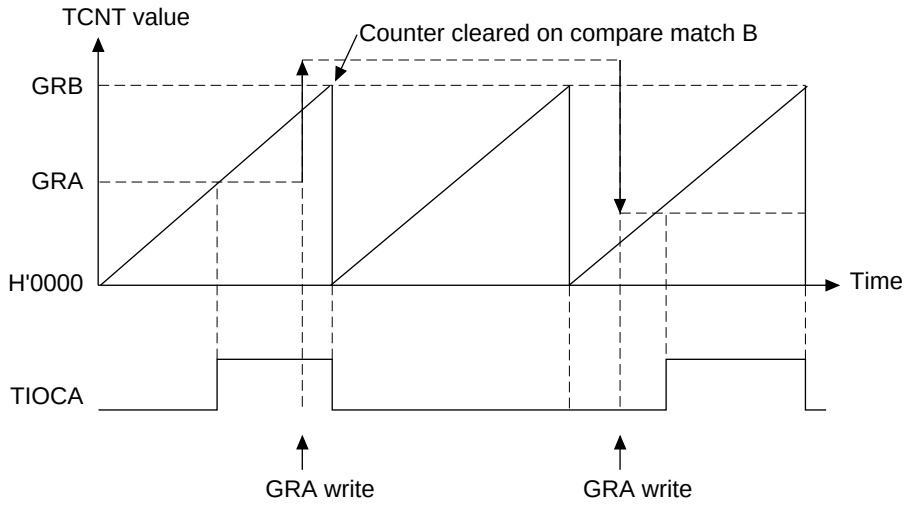
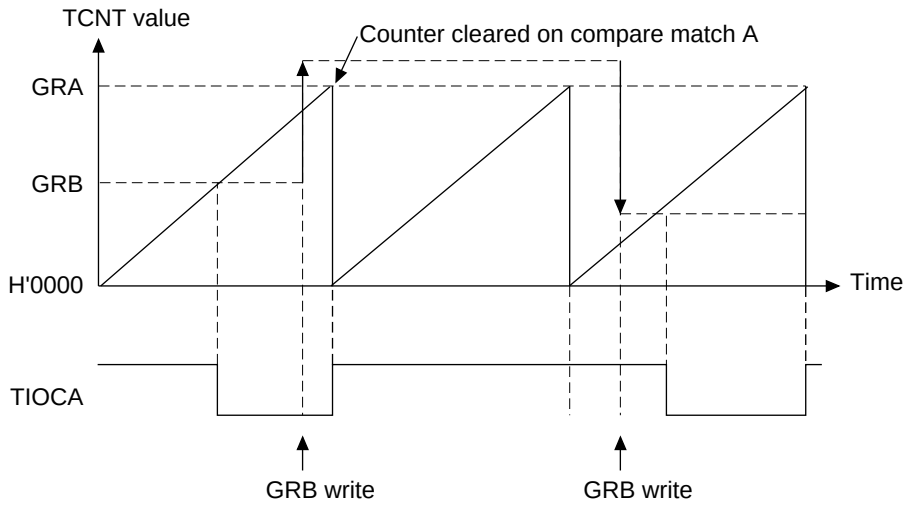



**Figure 10.29 PWM Mode Operation Example 1**




a. 0% duty




b. 100% duty

**Figure 10.30 PWM Mode Operation Example 2**

### 10.4.5 Reset-Synchronized PWM Mode

In the reset-synchronized PWM mode, three pairs of complementary positive and negative PWM waveforms that share a common wave turning point can be obtained using channels 3 and 4. When set for reset-synchronized PWM mode, the TIOCA3, TIOCB3, TIOCA4, TOCXA4, TIOCB4, and TOCXB4 pins automatically become PWM output pins and TCNT3 becomes an upcounter. Table 10.12 shows the PWM output pins used and table 10.13 shows the settings of the registers used.

**Table 10.12 Output Pins for Reset-Synchronized PWM Mode**

| Channel | Output Pin | Description |
|---|---|---|
| 3 | TIOCA3 | PWM output 1 |
| | TIOCB3 | PWM output 1' (negative-phase waveform of PWM output 1) |
| 4 | TIOCA4 | PWM output 2 |
| | TOCXA4 | PWM output 2' (negative-phase waveform of PWM output 2) |
| | TIOCB4 | PWM output 3 |
| | TOCXB4 | PWM output 3' (negative-phase waveform of PWM output 3) |

**Table 10.13 Register Settings for Reset-Synchronized PWM Mode**

| Register | Description of Contents |
|---|---|
| TCNT3 | Initial setting of H'0000 |
| TCNT4 | Not used (functions independently) |
| GRA3 | Set count cycle for TCNT3 |
| GRB3 | Sets the turning point for PWM waveform output by the TIOCA3 and TIOCB3 pins |
| GRA4 | Sets the turning point for PWM waveform output by the TIOCA4 and TOCXA4 pins |
| GRB4 | Sets the turning point for PWM waveform output by the TIOCB4 and TOCXB4 pins |

**Procedure for Selecting the Reset-Synchronized PWM Mode (figure 10.31):**

1. Clear the STR3 bits in the TSTR to halt TCNT3. The reset-synchronized PWM mode must be set up while TCNT3 is halted.
2. Set bits TPSC2–TPSC0 in the TCR to select the counter clock source for channel 3. If an external clock source is selected, select the external clock edges with bits CKEG1 and CKEG0 in the TCR.
3. Set bits CCLR1 and CCLR0 in the TCR3 to select GRA3 as a counter clear source.
4. Set bits CMD1 and CMD0 in TMDB to select the reset-synchronized PWM mode. TIOCA3–TIOCB4, TOCXA4, and TOCXB4 automatically become PWM output pins.

5. Reset the TCNT3 (to H'0000). The TCNT4 need not be set.
6. The GRA3 is the waveform period register. Set the waveform period value in the GRA3. Set the transition times of the PWM output waveforms in the GRB3, GRA4, and GRB4. Set times within the compare match range of the TCNT3.

   $X \leq GRA3$ (X: setting value)

7. Set the STR3 bits in the TSTR to 1 to let the TCNT3 start counting.

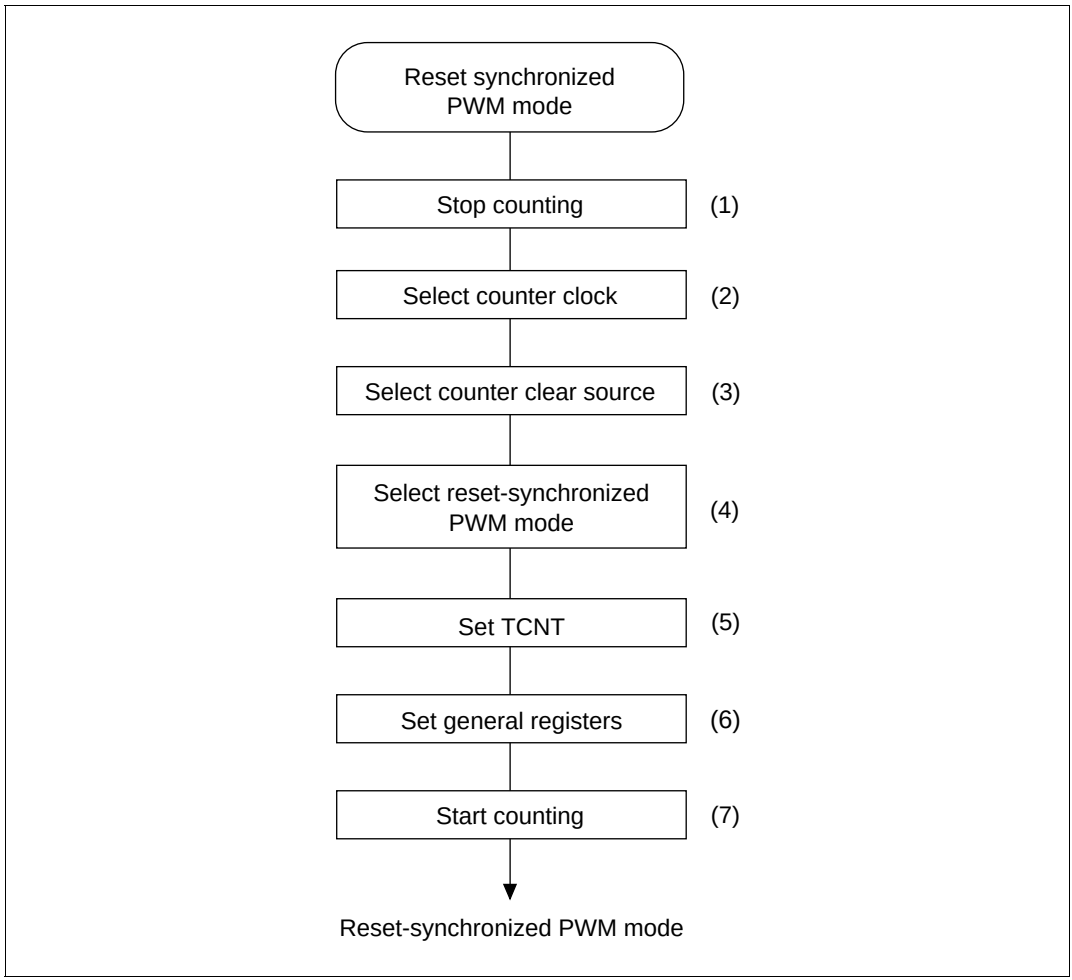



**Figure 10.31 Procedure for Selecting the Reset-Synchronized PWM Mode**

**Reset-Synchronized PWM Mode Operation:** Figure 10.32 shows an example of operation in the reset-synchronized PWM mode. TCNT3 operates as an upcounter that is cleared to H'0000 at compare match with GRA3. TCNT4 runs independently and is isolated from GRA4 and GRB4. The PWM waveform outputs toggle at each compare match (GRB3, GRA3, and GRB4 with TCNT3) and when the counter is cleared.

See section 10.4.8, Buffer Mode, for details on simultaneously setting reset-synchronized PWM mode and buffer operation.

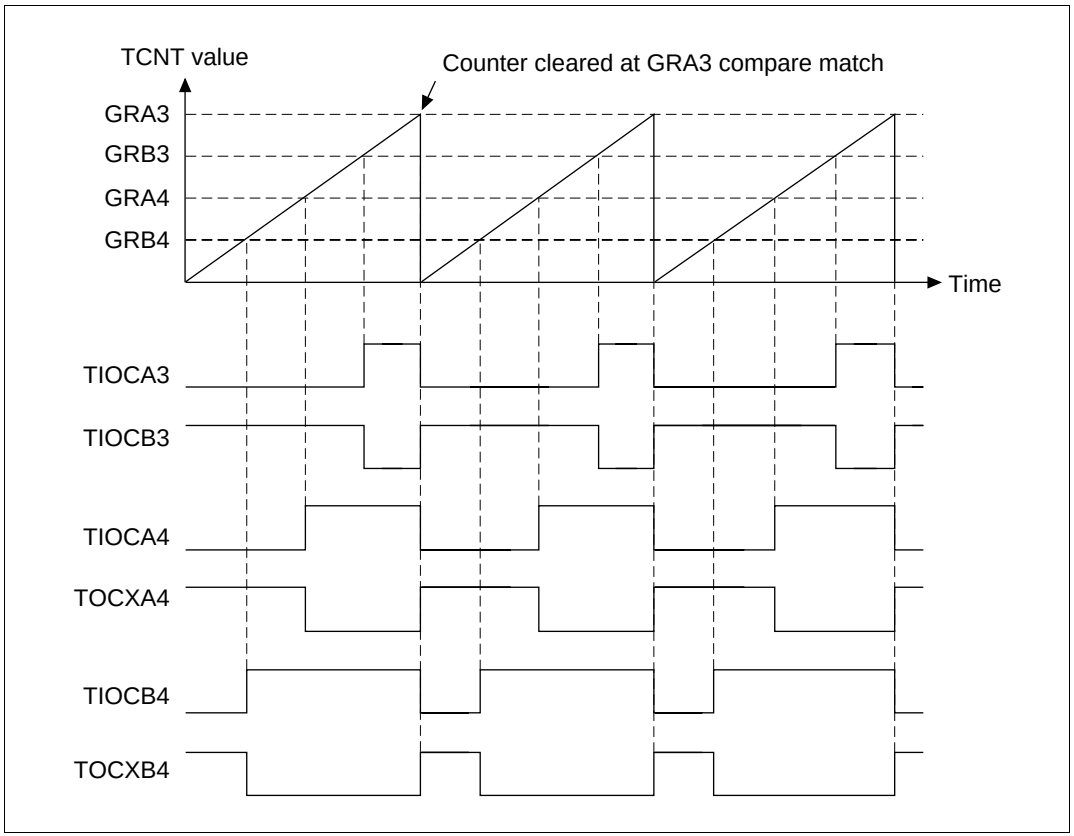



**Figure 10.32 Reset-Synchronized PWM Mode Operation Example 1**

### 10.4.6 Complementary PWM Mode

In the complementary PWM mode, three pairs of complementary, non-overlapping, positive and negative PWM waveforms can be obtained using channels 3 and 4. In complementary PWM mode, the TIOCA3, TIOCB3, TIOCA4, TOCXA4, TIOCB4, and TOCXB4 pins automatically become PWM output pins and TCNT3 and TCNT4 become upcounters. Table 10.14 shows the PWM output pins used and table 10.15 shows the settings of the registers used.

**Table 10.14 Output Pins for Complementary PWM Mode**

| Channel | Output Pin | Description |
|---|---|---|
| 3 | TIOCA3 | PWM output 1 |
| | TIOCB3 | PWM output 1' (non-overlapping negative-phase waveform of PWM output 1) |
| 4 | TIOCA4 | PWM output 2 |
| | TOCXA4 | PWM output 2' (non-overlapping negative-phase waveform of PWM output 2) |
| | TIOCB4 | PWM output 3 |
| | TOCXB4 | PWM output 3' (non-overlapping negative-phase waveform of PWM output 3) |

**Table 10.15 Register Settings for Complementary PWM Mode**

| Register | Description of Contents |
|---|---|
| TCNT3 | Initial setting of non-overlap cycle (the difference with TCNT4) |
| TCNT4 | Initial setting of H'0000 |
| GRA3 | Set upper limit of TCNT3–1 |
| GRB3 | Sets the turning point for PWM waveform output by the TIOCA3 and TIOCB3 pins. |
| GRA4 | Sets the turning point for PWM waveform output by the TIOCA4 and TOCXA4 pins. |
| GRB4 | Sets the turning point for PWM waveform output by the TIOCB4 and TOCXB4 pins. |

**Procedure for Selecting the Complementary PWM Mode (figure 10.33):**

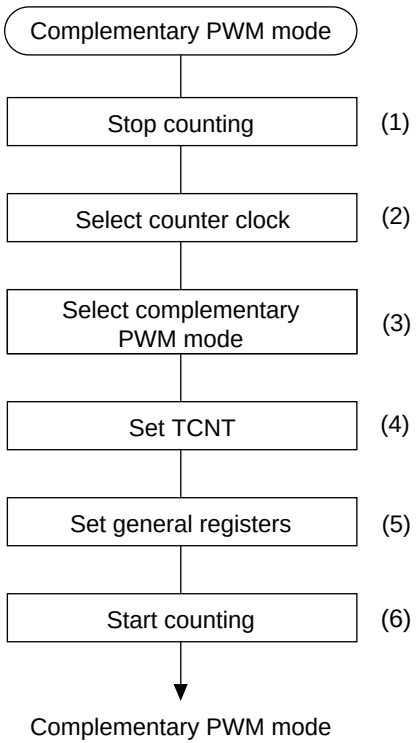
1. Clear STR3 and STR4 bits in the TSTR to halt the timer counters. The complementary PWM mode must be set up while TCNT3 and TCNT4 are halted.
2. Set bits TPSC2–TPSC0 in the TCR to select the same counter clock source for channels 3 and 4. If an external clock source is selected, select the external clock edges with bits CKEG1 and CKEG0 in the TCR. Do not select any counter clear source with bits CCLR1 and CCLR0 in the TCR.
3. Set bits CMD1 and CMD0 in TMDB to select the complementary PWM mode. TIOCA3–TIOCB4, TOCXA4, and TOCXB4 automatically become PWM pins.
4. Reset TCNT4 (to H'0000). Set the non-overlap offset in TCNT3. Do not set TCNT3 and TCNT4 to the same value.

5. GRA3 is the waveform period register. Set the upper limit of TCNT3–1*. Set the transition times of the PWM output waveforms in GRB3, GRA4, and GRB4. Set times within the compare match range of TCNT3 and TCNT4.

   $T \leq X$
   (X: initial setting of GRB3, GRA4, and GRB4; T: initial setting of TCNT3)

   Note: GRA3 = [cycle count/2] + [count of non-overlaps] – 2cyc=[upper limit of TCNT3] – 1

6. Set the STR3 and STR4 bits in the TSTR to 1 to let TCNT3 and TCNT4 start counting.




Note To re-engage the complementary PWM mode after it has been aborted, start settings from step 1.

**Figure 10.33 Procedure for Selecting the Complementary PWM Mode**

**Complementary PWM Mode Operation:** Figure 10.34 shows an example of operation in the complementary PWM mode. TCNT3 and TCNT4 operate as up/downcounters, counting down from compare match of TCNT3 and GRA3 and counting up when TCNT4 underflows. PWM waveforms are output by repeated compare matches with each of the general registers in the

sequence TCNT3, TCNT4, TCNT4, TCNT3 (in this mode, TCNT3 starts out larger than TCNT4).

Figure 10.35 shows examples of PWM waveforms with 0% and 100% duty cycles (in one phase) in the complementary PWM mode. In this example, the pin output changes upon GRB3 compare match, so duty cycles of 0% and 100% can be obtained by setting GRB3 to a value larger than GRA3. Combining buffer operation with the above operation makes it easy to change the duty while operating. See section 10.4.8, Buffer Operation, for details.

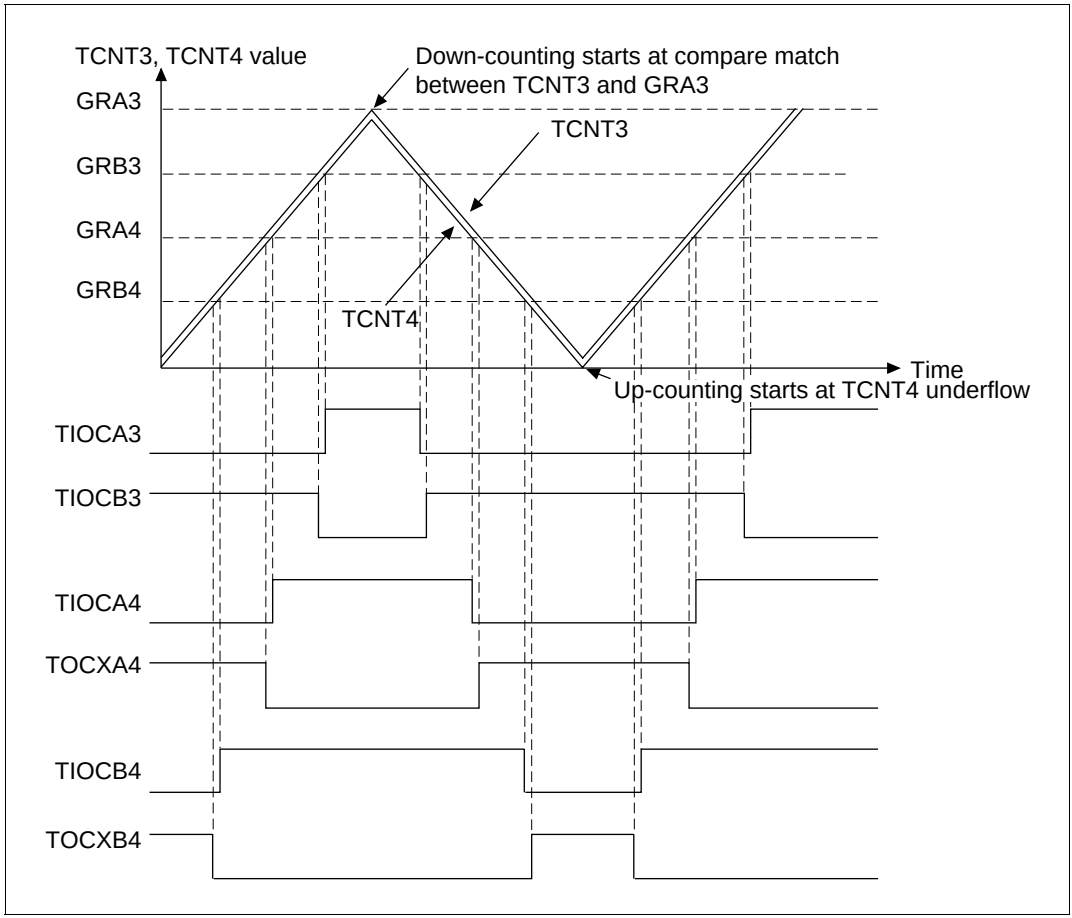



**Figure 10.34 Complementary PWM Mode Operation Example 1**

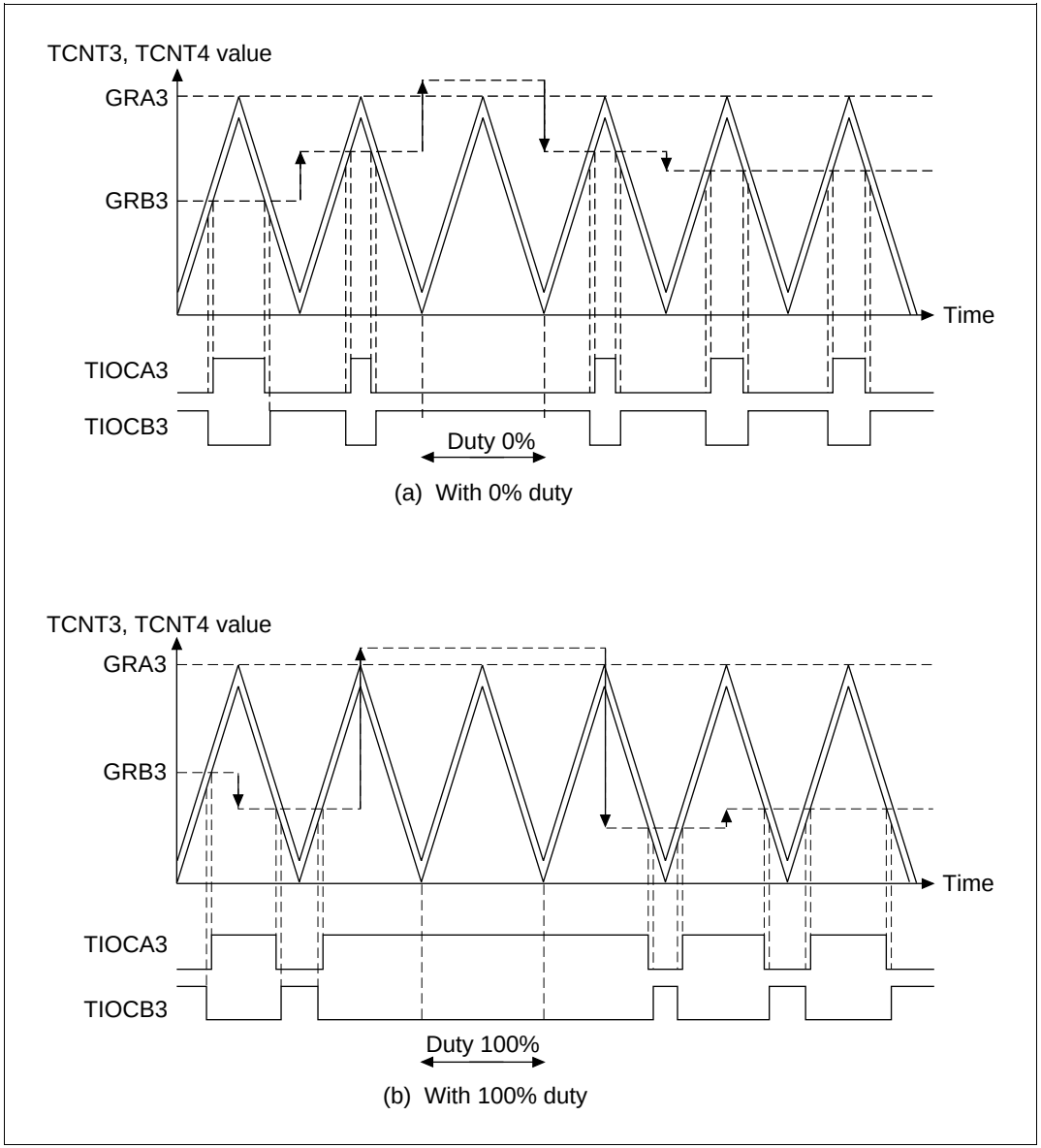



**Figure 10.35 Complementary PWM Mode Operation Example 2**

At the point where the up-count/down-count changes in the complementary PWM mode, TCNT3 and TCNT4 will overshoot and undershoot, respectively. When this occurs, the setting conditions for the IMFA bit of channel 3 and the overflow flag (OVF) of channel 4 are different from usual. Transfer conditions for the buffer also differ. The timing is as shown in figures 10.36 and 10.37.

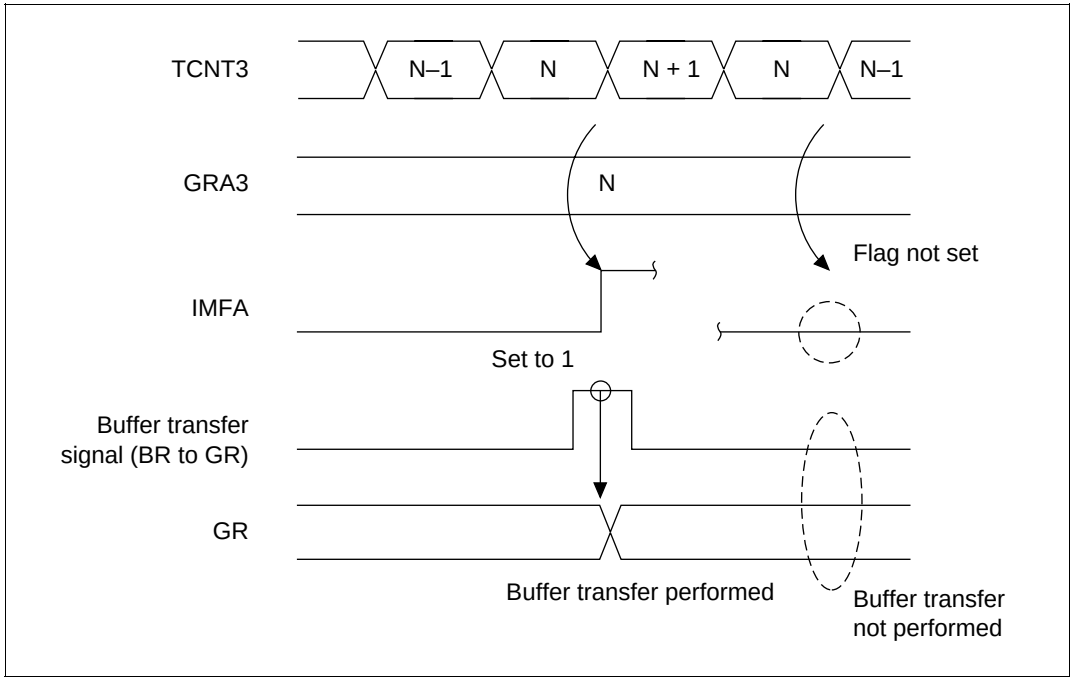



**Figure 10.36 Overshoot Timing**

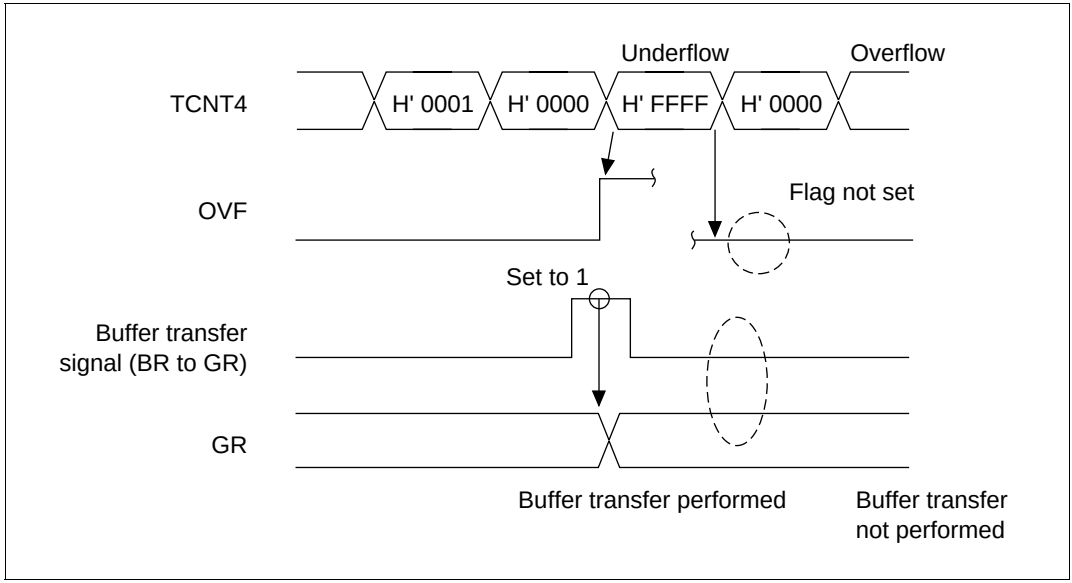



**Figure 10.37 Undershoot Timing**

The IMFA bit of channel 3 is set to 1 for increment pulses and the OVF bit of channel 4 is set to 1 for underflows only. The buffer register (BR) set for the buffer operation is transferred to the GR upon compare match A3 (when incrementing) or TCNT4 underflow.

**GR Setting in Complementary Mode:** Be aware of the following when setting the general registers in complementary PWM mode and when making changes during operation.

- Initial values: Setting H'0000 to T–1 (the initial setting T: TCNT3) is prohibited. After counting starts, this setting is allowed from the point when the first A3 compare match occurs.
- Methods of changing settings: Use the buffer operation. Writing directly to general registers may result in incorrect waveform output.
- When changing settings: See figure 10.38.

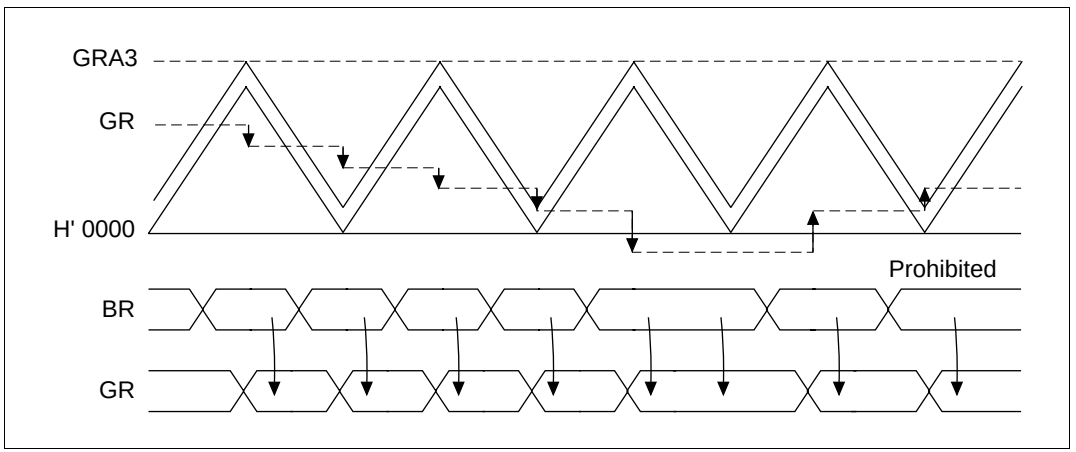



**Figure 10.38 Example of Changing GR Settings with Buffer Operation (1)**

**Buffer Transfers when Changing from Increment to Decrement:** When the contents of the GR are within GRA3 – T + 1 and GRA3, do not transfer a value outside this range. When the contents of GR are outside this range, do not a transfer a value within it. Figure 10.39 illustrates a point of caution regarding changing of GR settings with a buffer operation.

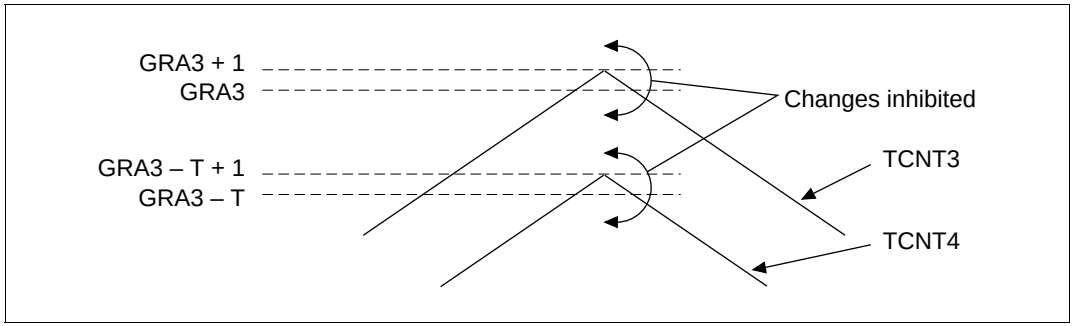



**Figure 10.39 Caution for Changing GR Settings with Buffer Operation (1)**

**Buffer Transfers when Changing from Decrement to Increment:** When the contents of the GR are within H'0000 to T–1, do not transfer a value outside this range. When the contents of GR are outside this range, do not transfer a value within it. Figure 10.40 illustrates this point of caution regarding changing of GR settings with a buffer operation

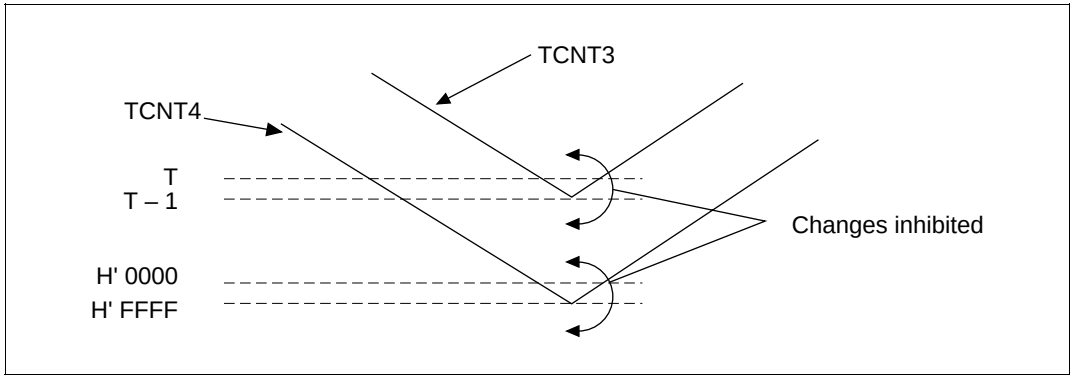



**Figure 10.40 Caution for Changing GR Settings with Buffer Operation (2)**

**When GR Settings are Outside the Count Range (H'0000–GRA3):** Waveforms of duty cycle 0% and 100% can be output by setting GR outside the count area. Be sure to make the direction of the count (increment/decrement) when writing a setting from outside the count area into the buffer register (BR) the same as the count direction when writing the setting that returns to within the count area in the BR.

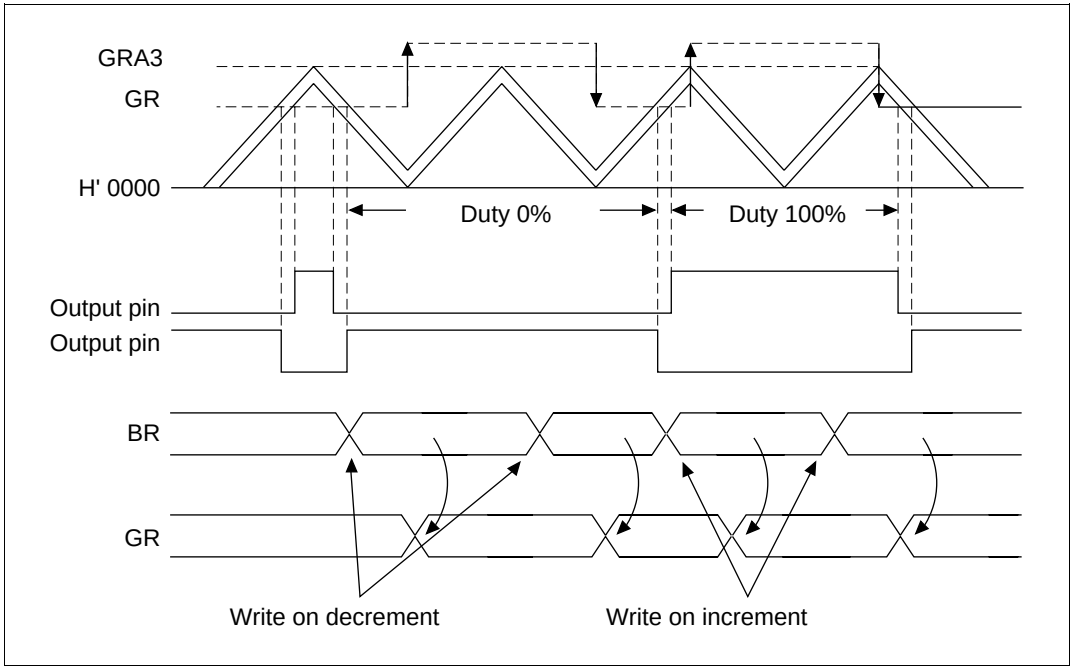



**Figure 10.41 Example of Changing GR Settings with Buffer Operation (2)**

The above settings are made by detecting the occurrence of a GRA3 compare match or underflow of TCNT4 and then writing to BR. They can also be accomplished by starting up the DMAC with a GRA3 compare match.

### 10.4.7 Phase Counting Mode

The phase counting mode detects the phase differential of two external clock inputs (TCLKA and TCLKB) and counts TCNT2 up or down. When set in the phase counting mode, the TCLKA and TCLKB pins automatically become external clock input pins, regardless of the settings of the TPSC2–TPSC0 bits of TCR2 or the CKEG1 and CKEG0 bits. TCNT2 also becomes an up/down counter. Since the TCR2 CCLR1/CCLR0 bits, TIOR2, TIER2, TSR2, GRA2 and GRB2 are all enabled, input capture and compare match functions and interrupt sources can be used. Phase counting is available only in channel 2.

**Procedure for Selecting the Phase Counting Mode:** Figure 10.42 shows the procedure for selecting the phase counting mode.

1. Set the MDF bit of the timer mode register (TMDR) to 1 to select the phase counting mode.
2. Select the flag set conditions using the FDIR bit of the TMDR.
3. Set the STR2 bit of the timer start register (TSTR) to 1 to start the count.

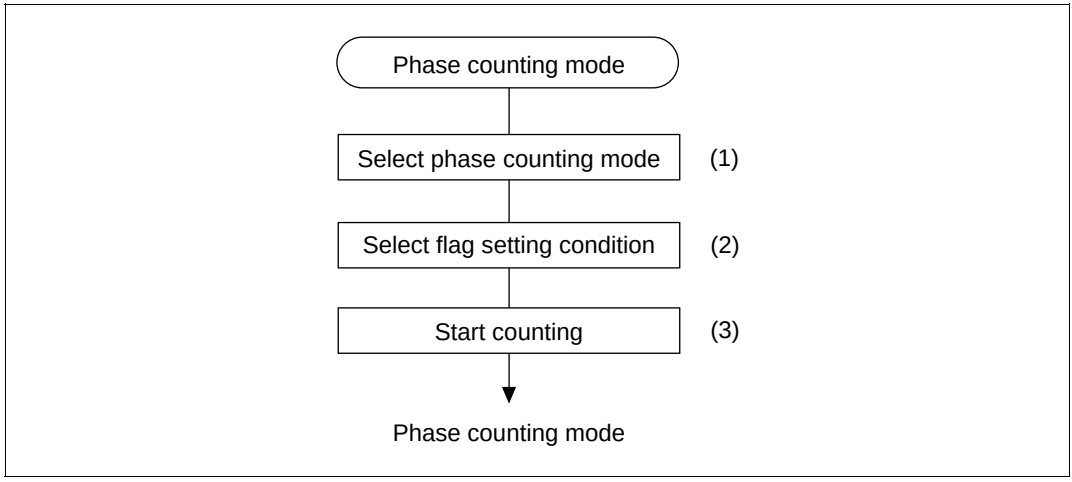



**Figure 10.42 Procedure for Selecting the Phase Counting Mode**

**Phase Counting Operation:** Figure 10.43 shows an example of phase counting mode operation. Table 10.16 lists the upcounting and downcounting conditions for TCNT2. The ITU counts on both rise and fall edges of TCLKA and TCLKB. The phase differential and overlap of TCLKA and TCLKB must be 1.5 cycles or more and the pulse width must be 2.5 cycles or more.

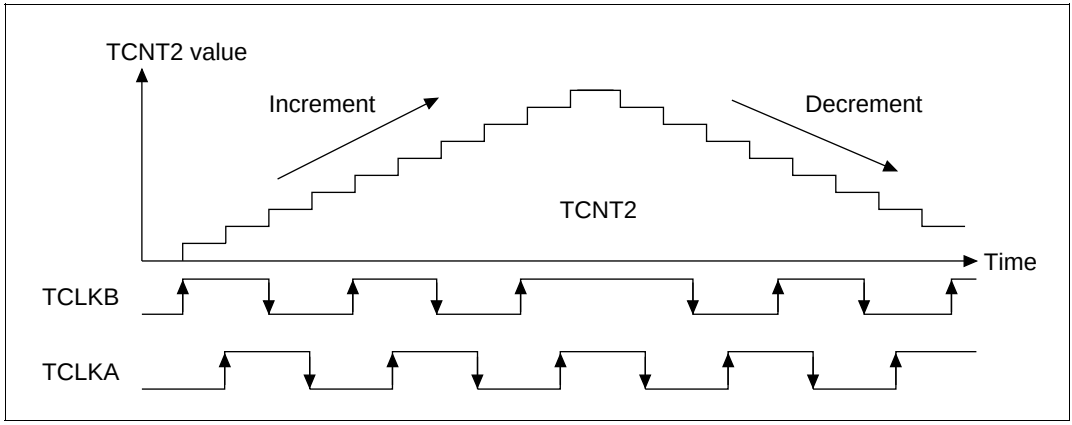



**Figure 10.43 Phase Counting Mode Operation**

**Table 10.16 Up/Down Counting Conditions**

| Counting Direction | Increment | | | | Decrement | | | |
|---|---|---|---|---|---|---|---|---|
| TCLKB | Rising | High | Falling | Low | Rising | High | Falling | Low |
| TCLKA | Low | Rising | High | Falling | High | Falling | Low | Rising |

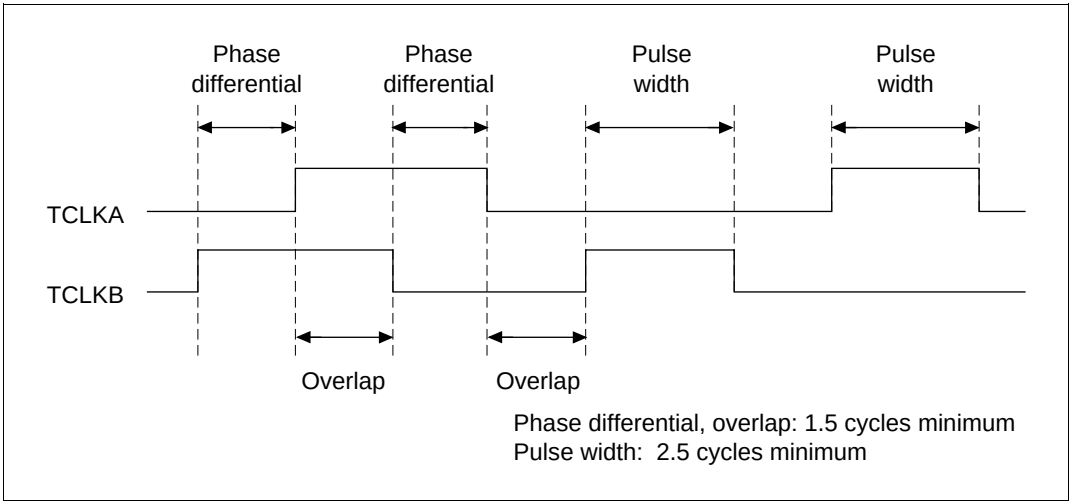



**Figure 10.44 Phase Differentials, Overlap and Pulse Width in the Phase Counting Mode**

### 10.4.8 Buffer Mode

In the buffer mode, the buffer operation functions differ depending on whether the general registers are set to output compare or input capture, the reset-synchronized PWM mode, or complementary PWM mode. The buffer mode is a function of channels 3 and 4 only. Buffer operations set this way function as follows.

**GR is an Output Compare Register:** The value of the buffer registers of a channel is transferred to the GR when the channel experiences a compare match. This is illustrated in figure 10.45.

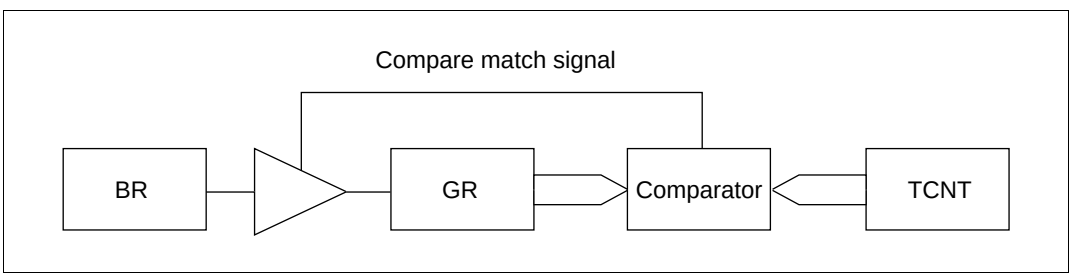



**Figure 10.45 Compare Match Buffer Operation**

**GR is an Input Capture Register:** TCNT values are transferred to GR when input capture occurs and the value previously stored in GR is transferred to BR. This operation is illustrated in figure 10.46.

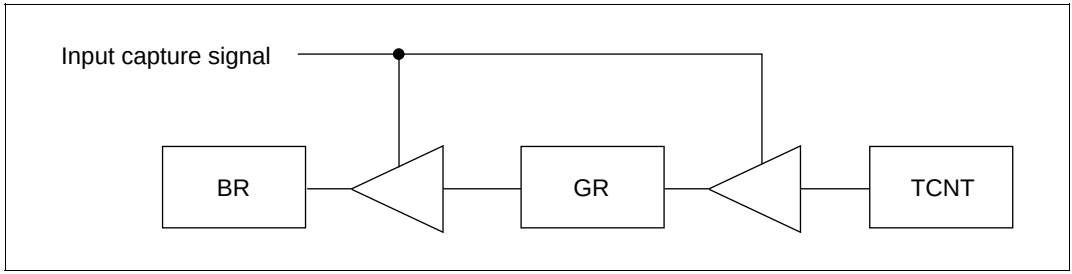



**Figure 10.46 Input Capture Buffer Operation**

**Complementary PWM Mode:** When the count direction of TCNT3 and TCNT4 change, the BR value is transferred to the GR. The following timing is employed for this transfer:

- Whenever TCNT3 and GRA3 compare-match
- Whenever TCNT4 underflows

**Reset-Synchronized PWM Mode:** The BR value is transferred to GR upon a GRA3 compare match.

**Procedure for Selecting the Buffer Mode (figure 10.47):**

1. Set the TIOR to select the output compare or input capture function of the GR.
2. Set bits BFA3, BFB3 and BFB4 in the TFCR to select the buffer mode for the GR.
3. Set the STR bit in the TSTR to 1 to start the TCNT counting.

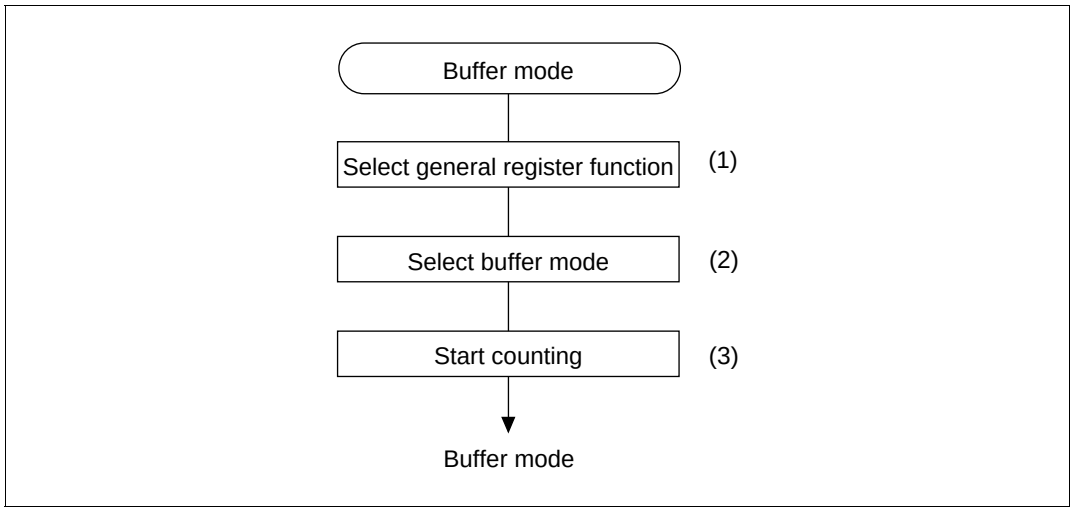



**Figure 10.47 Procedure for Selecting the Buffer Mode**

**Buffer Mode Operation:** Figure 10.48 shows an example of an operation in the buffer mode with GRA set as an output compare register and GRA and buffer register A (BRA) set for buffer operation. The TCNT operates as a periodic counter that is cleared by GRB compare match. TIOCA and TIOCB are set to toggle at compare matches A and B. Since buffer mode is selected, when TIOCA toggles at a compare match A, the BRA value is simultaneously transferred to the GRA. This operation is repeated at every compare match A. The transfer timing is shown in figure 10.49.

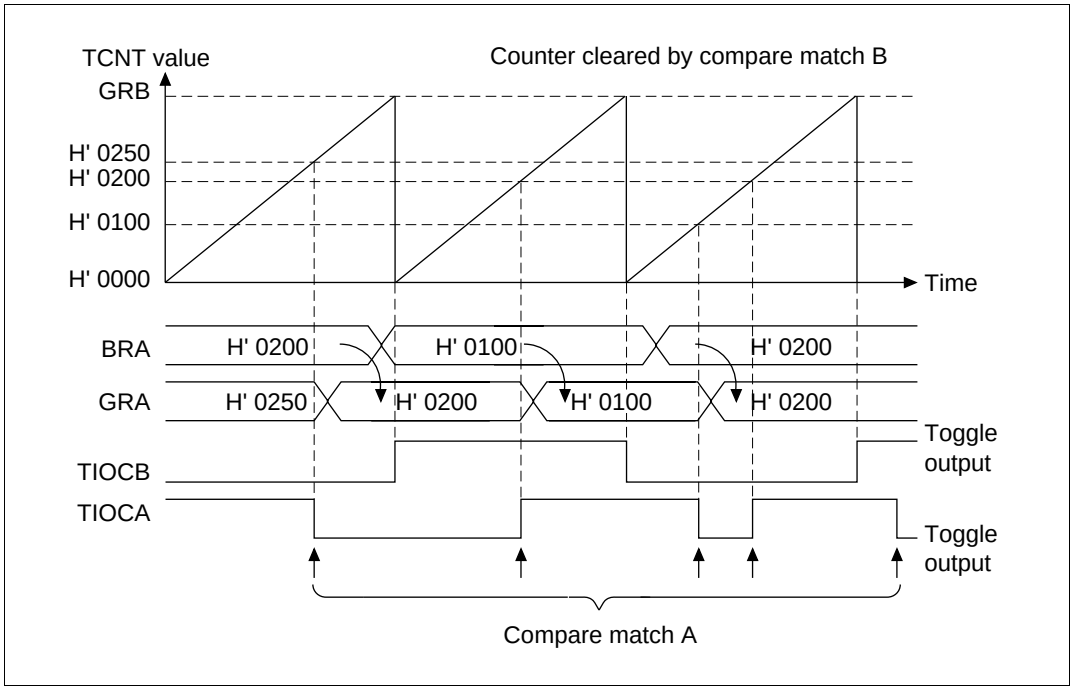



**Figure 10.48 Buffer Mode Operation Example 1 (Output Compare Register)**

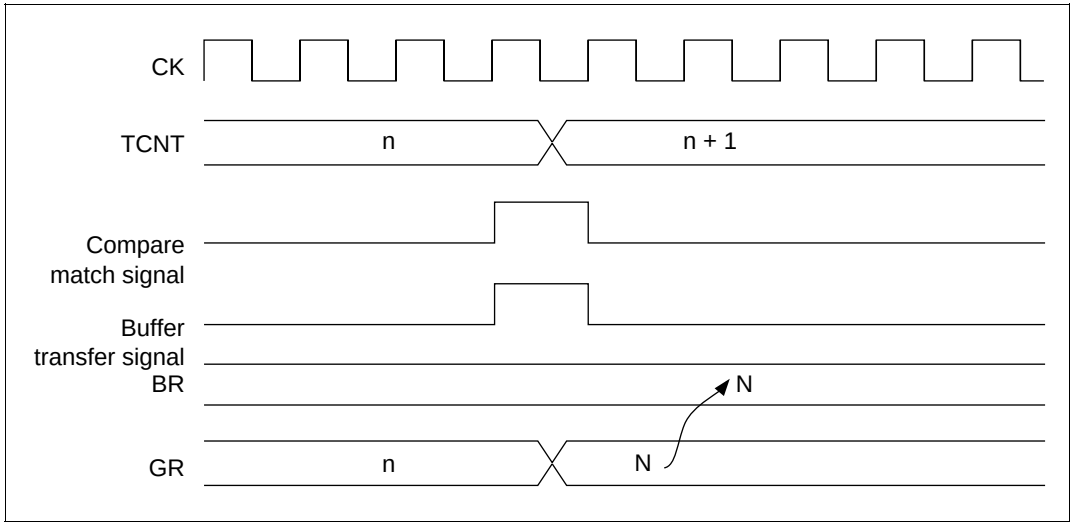



**Figure 10.49 Compare Match Timing Example for Buffer Operation**

Figure 10.50 shows an example of input capture operation in the buffer mode between GRA and BRA with GRA as an input capture register. The TCNT is cleared by input capture B. The falling edge is selected as the input capture edge at TIOCB. Both edges are selected as input capture edges at TIOCA. When the TCNT value is stored in GRA by input capture A, the previous GRA value is transferred to the BRA. The timing is shown in figure 10.51.

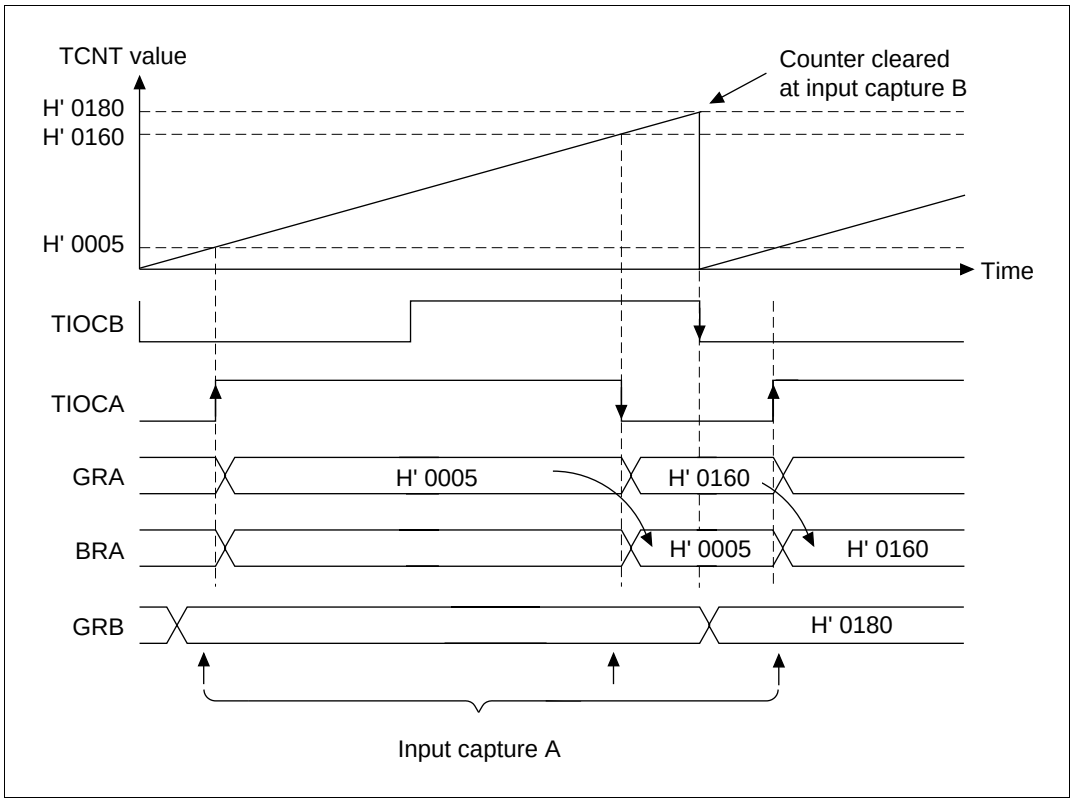



**Figure 10.50 Buffer Mode Operation Example 2 (Input Capture Register)**

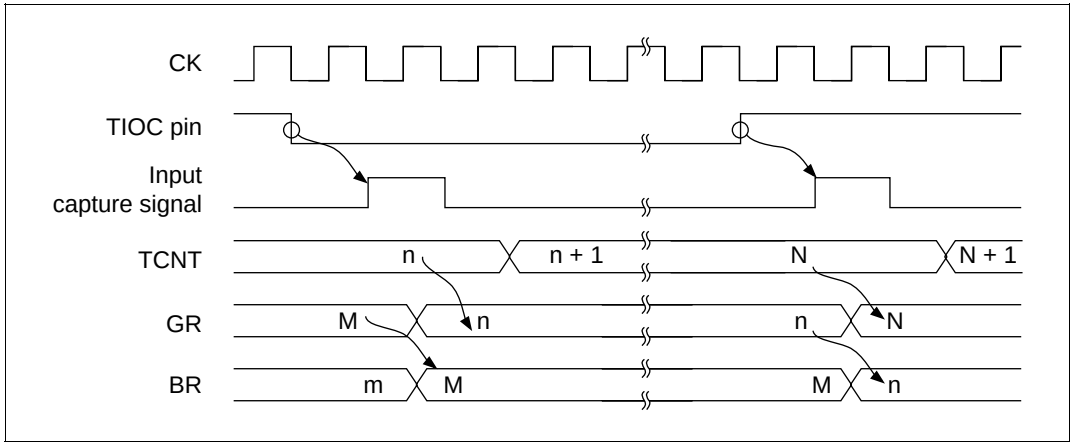



**Figure 10.51 Input Capture Timing Example for Buffer Operation**

An example of buffer operation in the complementary PWM mode between GRB3 and BRB3 is as shown in figure 10.52. By making GRB3 larger than GRA3 using the buffer operation, a PWM waveform with a duty cycle of 0% is generated. The transfer from BRB–GRB occurs upon TCNT3 and GRA compare match and TCNT4 underflow.

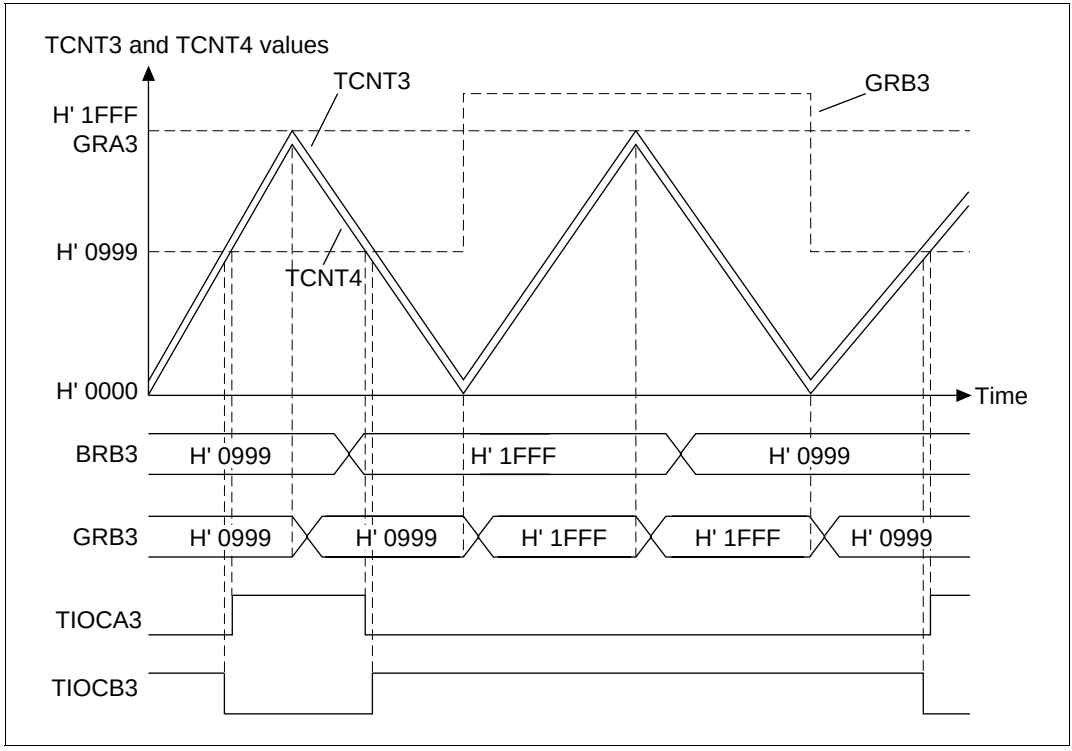



**Figure 10.52 Buffer Operation 4 (Complementary PWM Mode)**

### 10.4.9 ITU Output Timing

ITU outputs in channels 3 and 4 can be inverted with the TOCR.

**Output Inversion Timing with the TOCR:** Output levels can be inverted by inverting the output level select bits (OLS4 and OLS3) of the TOCR in the complementary PWM mode and reset-synchronized PWM mode. Figure 10.53 illustrates the timing.

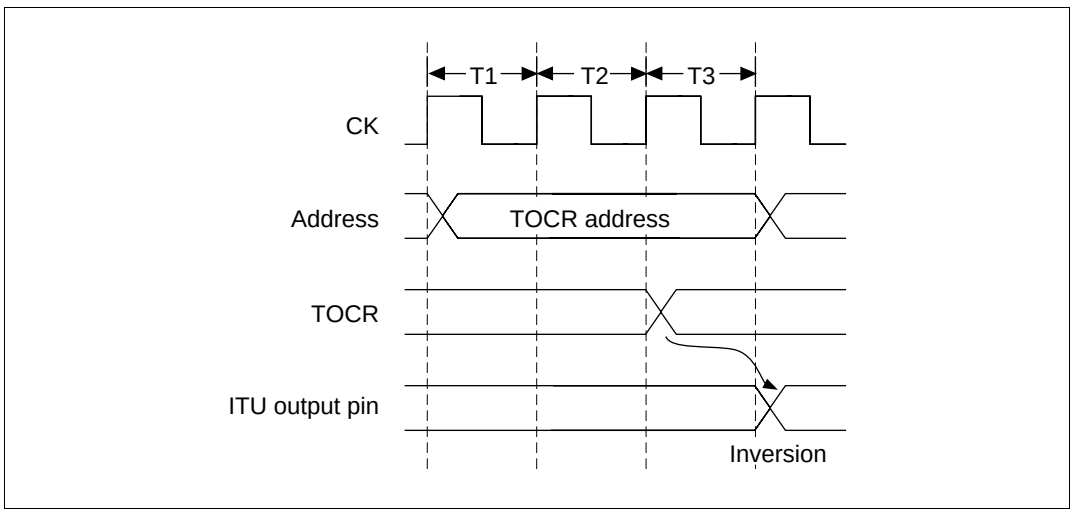



**Figure 10.53 Example of Inverting ITU Output Levels by Writing to TOCR**

## 10.5 Interrupts

The ITU has two interrupt sources: input capture/compare match and overflow.

### 10.5.1 Timing of Setting Status Flags

**Timing for Setting IMFA and IMFB in a Compare Match:** The IMF bits of the TSR are set to 1 by a compare match signal generated when the TCNT matches a general register. The compare match signal is generated in the last state in which the values match (when the TCNT is updated from the matching count to the next count). Therefore, when the TCNT matches the GRA or GRB, the compare match signal is not generated until the next timer clock input. Figure 10.54 shows the timing of setting the IMF bits.

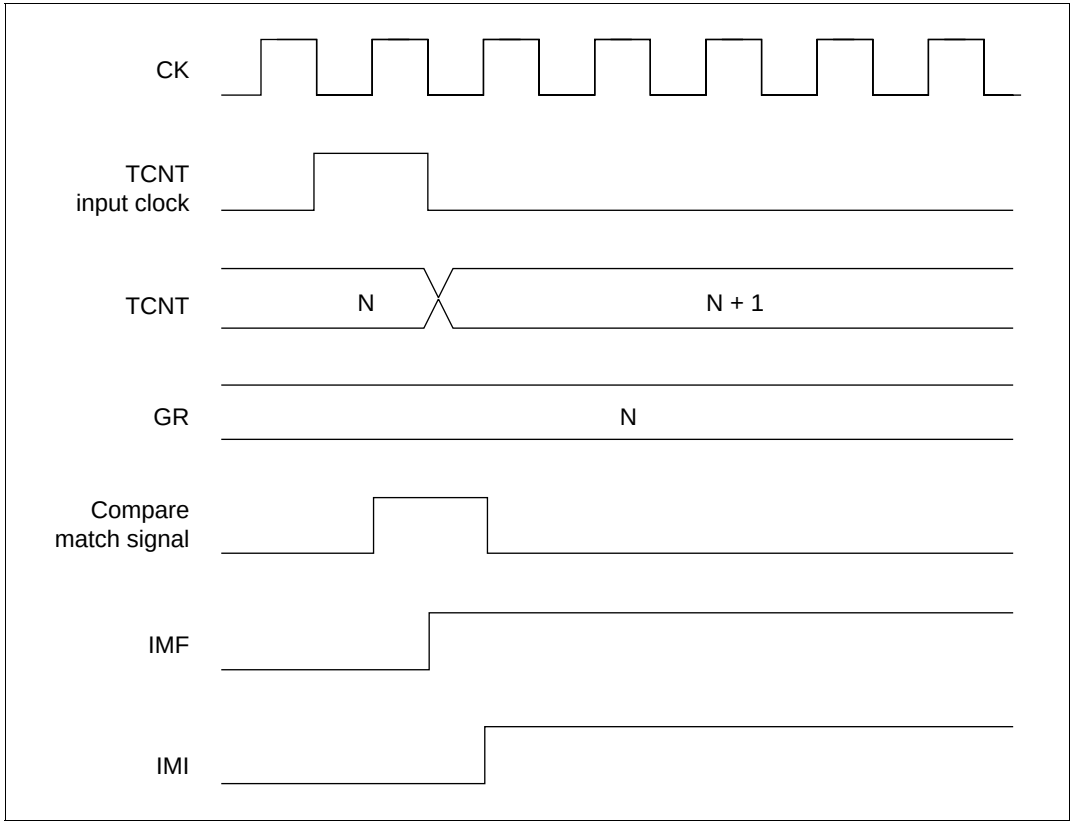



**Figure 10.54 Timing of Setting Compare Match Flags (IMFA, IMFB)**

**Timing of Setting IMFA, IMFB for Input Capture:** The IMFA and IMFB are set to 1 by an input capture signal. At this time, the TCNT contents are transferred to the GR. Figure 10.55 shows the timing.

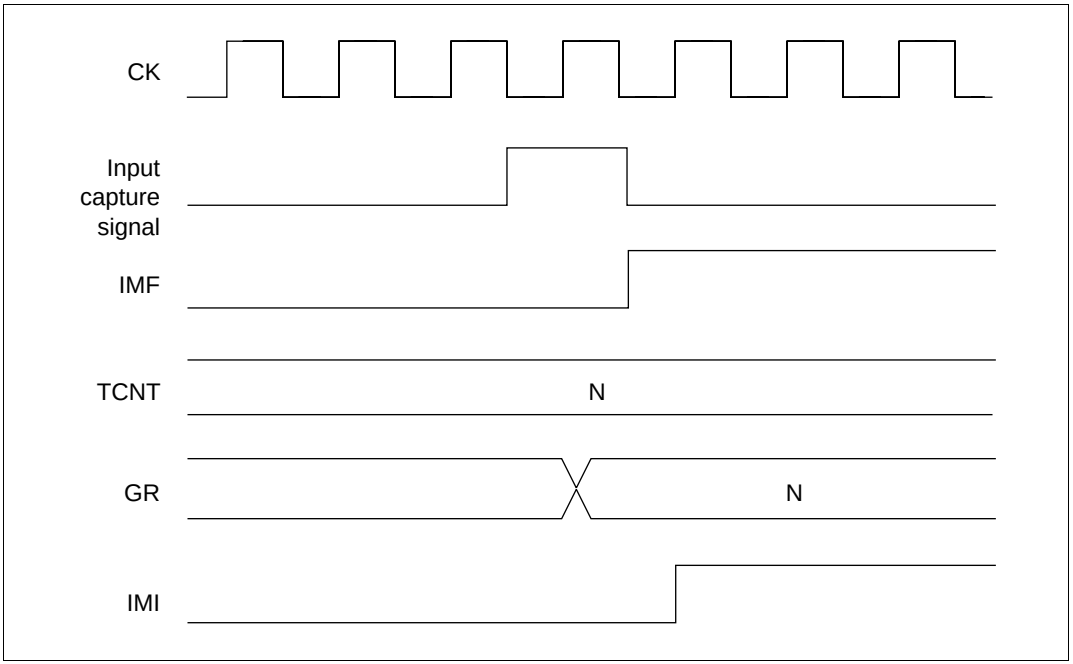



**Figure 10.55 Timing of Setting IMFA and IMFB for Input Capture**

**Timing of Setting Overflow Flag (OVF):** The OVF is set to 1 when the TCNT overflows from H'FFFF–H'0000 or underflows from H'0000–H'FFFF. Figure 10.56 shows the timing.

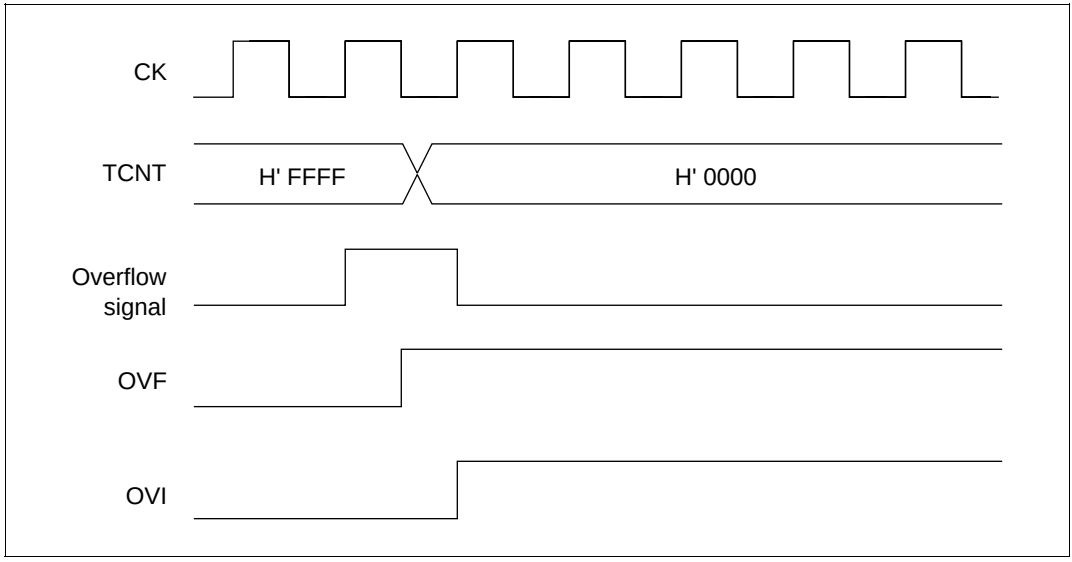



**Figure 10.56 Timing of Setting OVF**

### 10.5.2 Clear Timing of Status Flags

The status flags are cleared by a write cycle in which 1 is read on the CPU and then 0 is written to it. This timing is shown in figure 10.57.

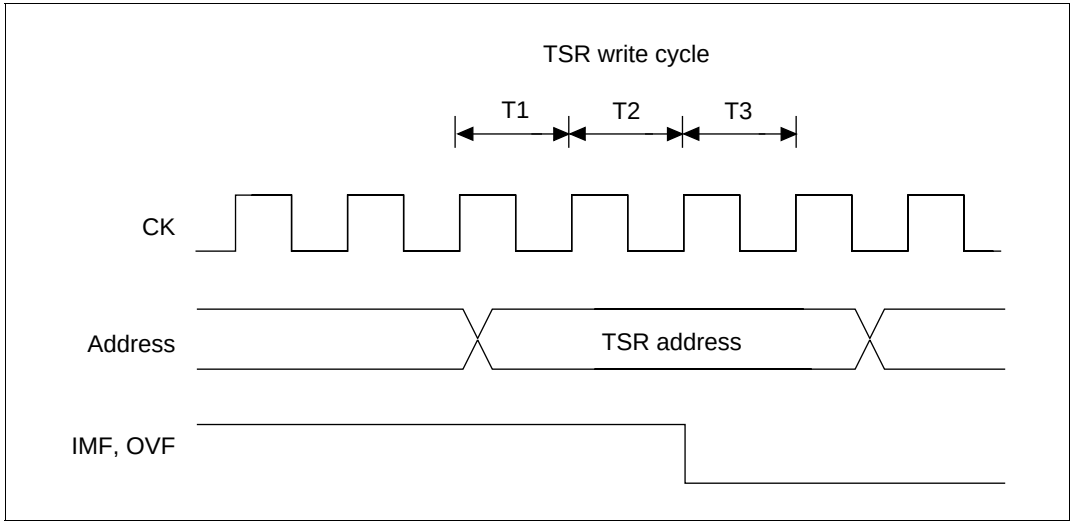



**Figure 10.57 Timing of Status Flag Clearing**

### 10.5.3 Interrupt Sources and Activating the DMAC

The ITU has compare match/input capture A interrupts, compare match/input capture B interrupts and overflow interrupts for each channel. Each of the fifteen of these three types of interrupts are allocated their own independently vectored addresses. When the interrupt's interrupt request flag is set to 1 and the interrupt enable bit is set to 1, the interrupt is requested.

The channel priority order can be changed with the interrupt controller. For more information, see section 5, Interrupt Controller. The compare match/input capture A interrupts of channels 0–3 can start the DMAC to transfer data. Table 10.17 lists the interrupt sources.

**Table 10.17 ITU Interrupt Sources**

| Channel | Interrupt Source | Description | DMAC Activation | Priority Order* |
|---|---|---|---|---|
| 0 | IMIA0 | Compare match or input capture A0 | Yes | High |
| | IMIB0 | Compare match or input capture B0 | No | ↑ |
| | OVI0 | Overflow 0 | No | |
| 1 | IMIA1 | Compare match or input capture A1 | Yes | |
| | IMIB1 | Compare match or input capture B1 | No | |
| | OVI1 | Overflow 1 | No | |
| 2 | IMIA2 | Compare match or input capture A2 | Yes | |
| | IMIB2 | Compare match or input capture B2 | No | |
| | OVI2 | Overflow 2 | No | |
| 3 | IMIA3 | Compare match or input capture A3 | Yes | |
| | IMIB3 | Compare match or input capture B3 | No | |
| | OVI3 | Overflow 3 | No | |
| 4 | IMIA4 | Compare match or input capture A4 | No | |
| | IMIB4 | Compare match or input capture B4 | No | ↓ |
| | OVI4 | Overflow 4 | No | Low |

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

## 10.6 Notes and Precautions

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

### 10.6.1 Contention between TCNT Write and Clear

If a counter clear signal occurs in the T3 state of a TCNT write cycle, clearing the counter takes priority and the write is not performed. The timing is shown in figure 10.58.

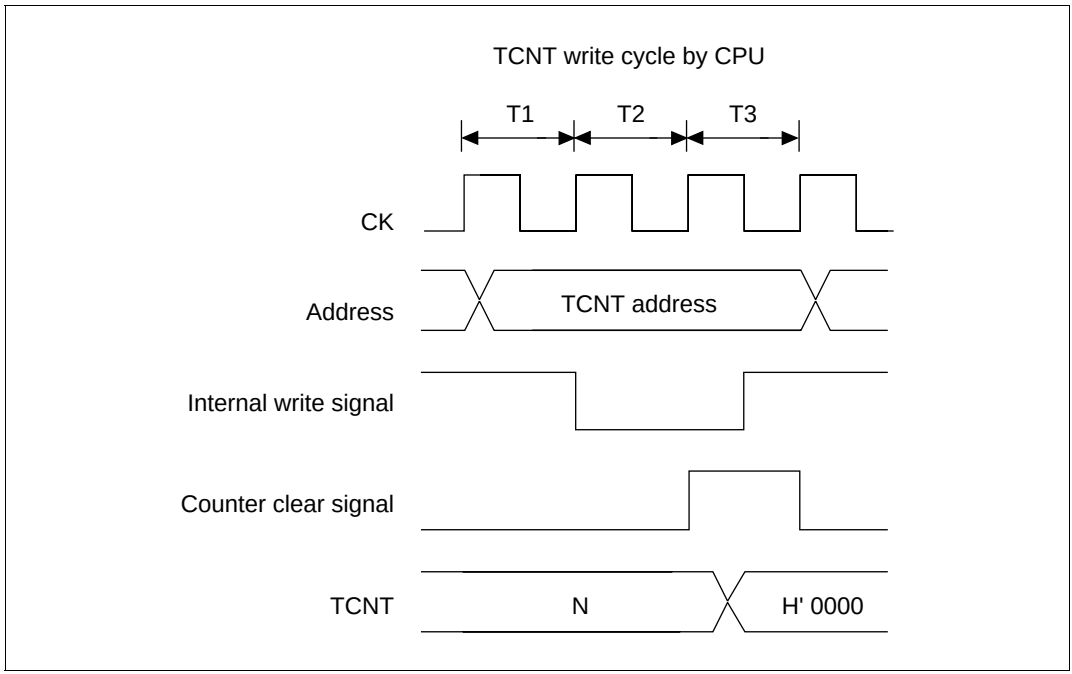



**Figure 10.58 Contention between TCNT Write and Clear**

### 10.6.2 Contention between TCNT Word Write and Increment

If an increment pulse occurs in the T3 state of a TCNT word write cycle, writing takes priority and the TCNT is not incremented. The timing is shown in figure 10.59.

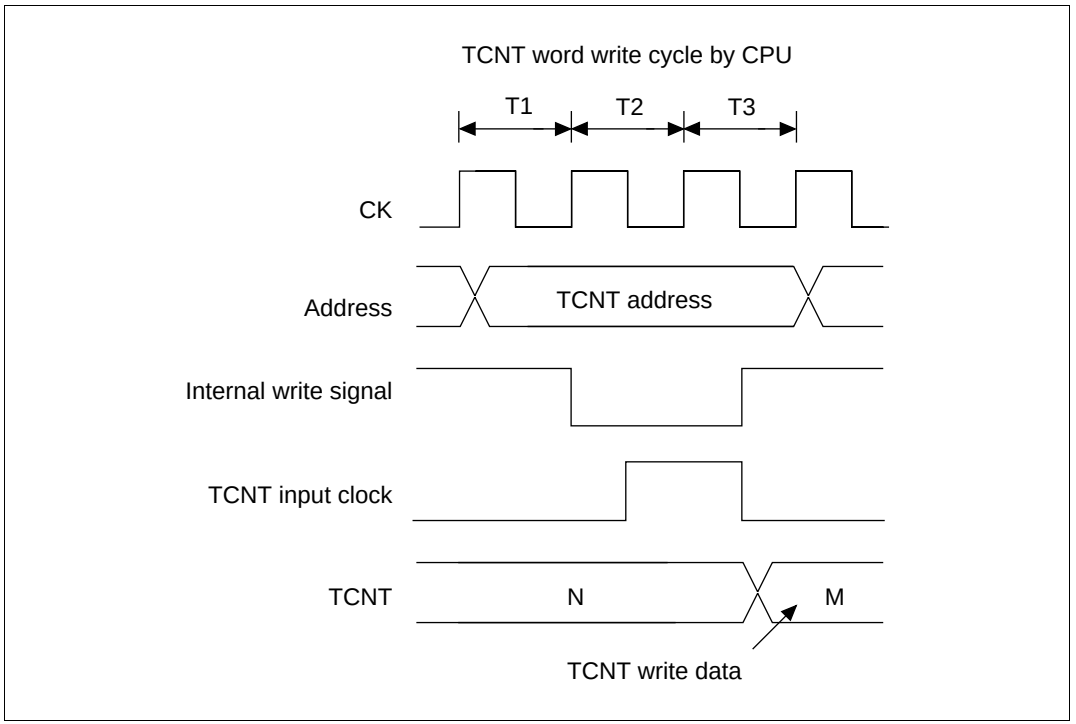



**Figure 10.59 Contention between TCNT Word Write and Increment**

### 10.6.3 Contention between TCNT Byte Write and Increment

If an increment pulse occurs in the T2 state or T3 state of a TCNT byte write cycle, counter writing takes priority and the byte data on the side that was previously written is not incremented. The TCNT byte data that was not written is also not incremented and retains its previous value. The timing is shown in figure 10.60 (which shows an increment during state T2 of a byte write cycle to TCNTH).

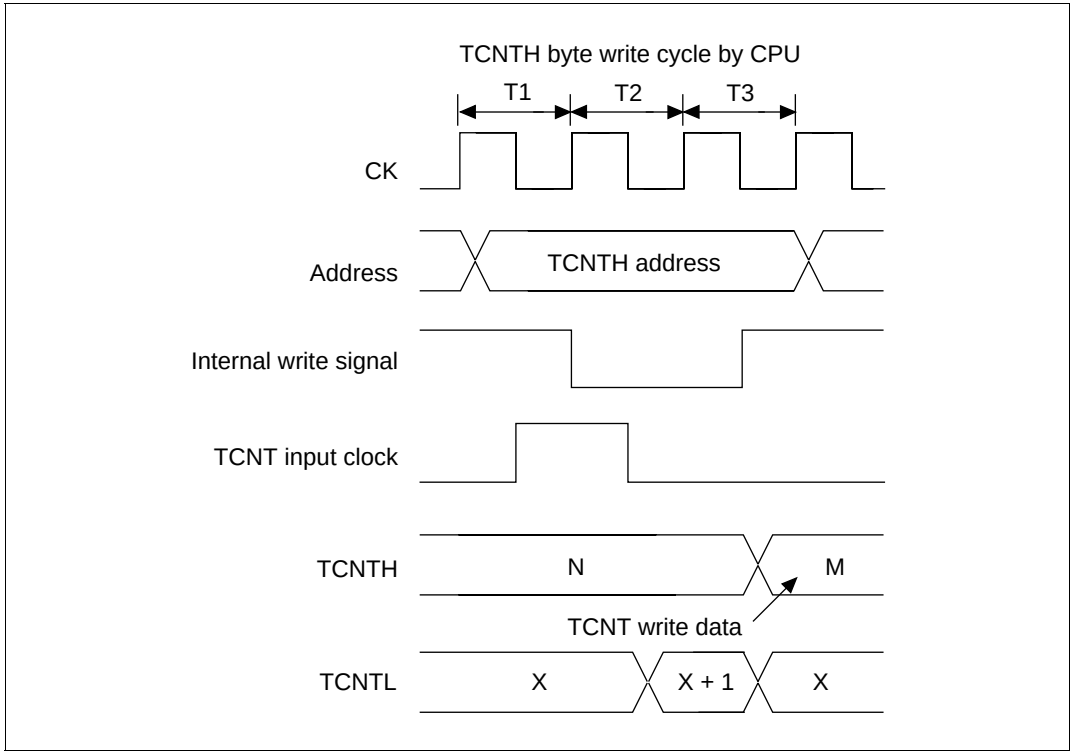



**Figure 10.60 Contention between TCNT Byte Write and Increment**

### 10.6.4 Contention between GR Write and Compare Match

If a compare match occurs in the T3 state of a general register (GR) write cycle, writing takes priority and the compare match signal is inhibited. The timing is shown in figure 10.61.

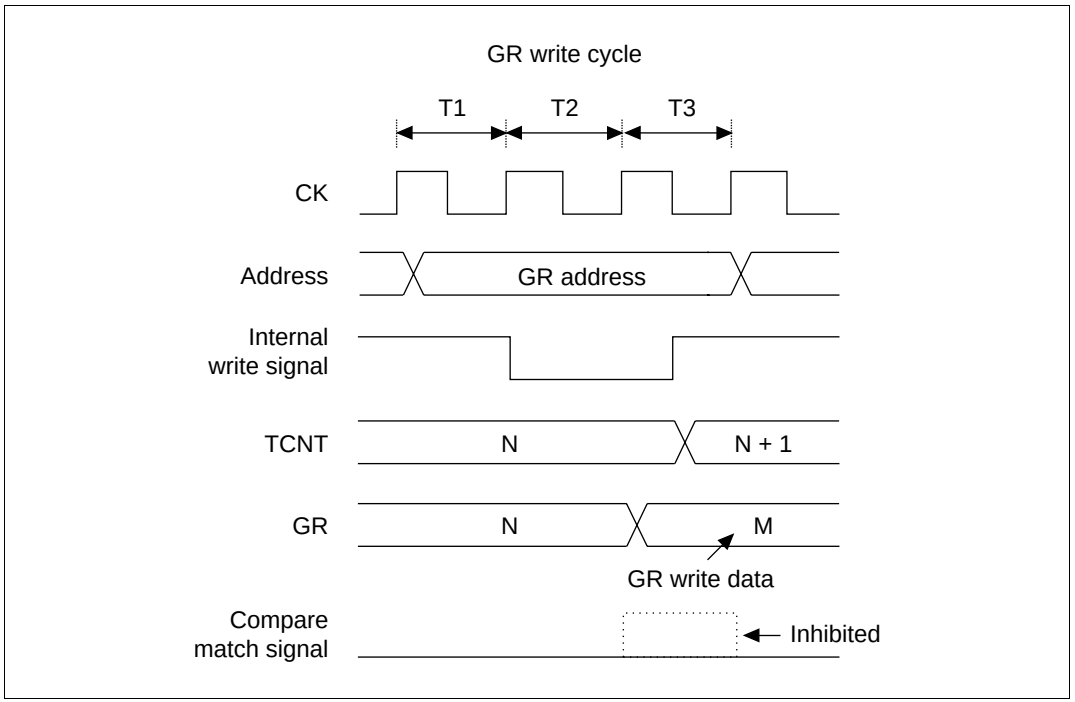



**Figure 10.61 Contention between General Register Write and Compare Match**

### 10.6.5 Contention between TCNT Write and Overflow/Underflow

If an overflow occurs in the T3 state of a TCNT write cycle, writing takes priority over counter incrementing. The OVF is set to 1. The same applies to underflows. This timing is shown in figure 10.62.

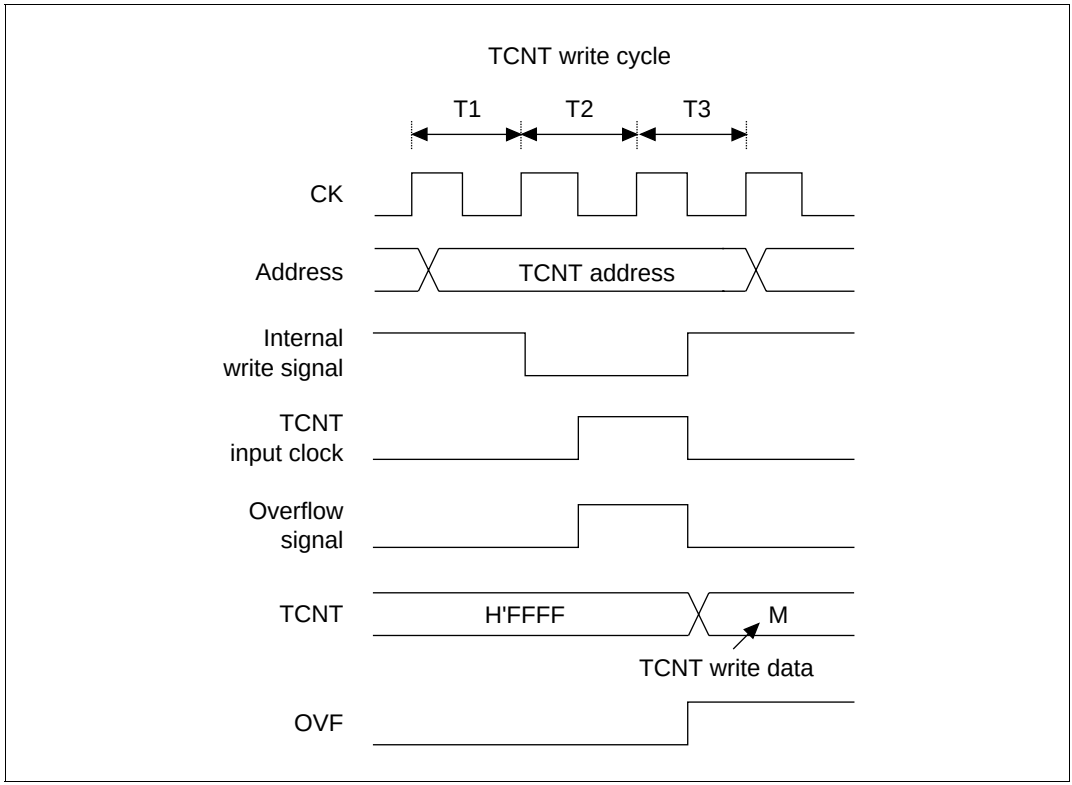



**Figure 10.62 Contention between TCNT Write and Overflow**

### 10.6.6 Contention between General Register Read and Input Capture

If an input capture signal is generated during the T3 state of a general register read cycle, the value before input capture is read. The timing is shown in figure 10.63.

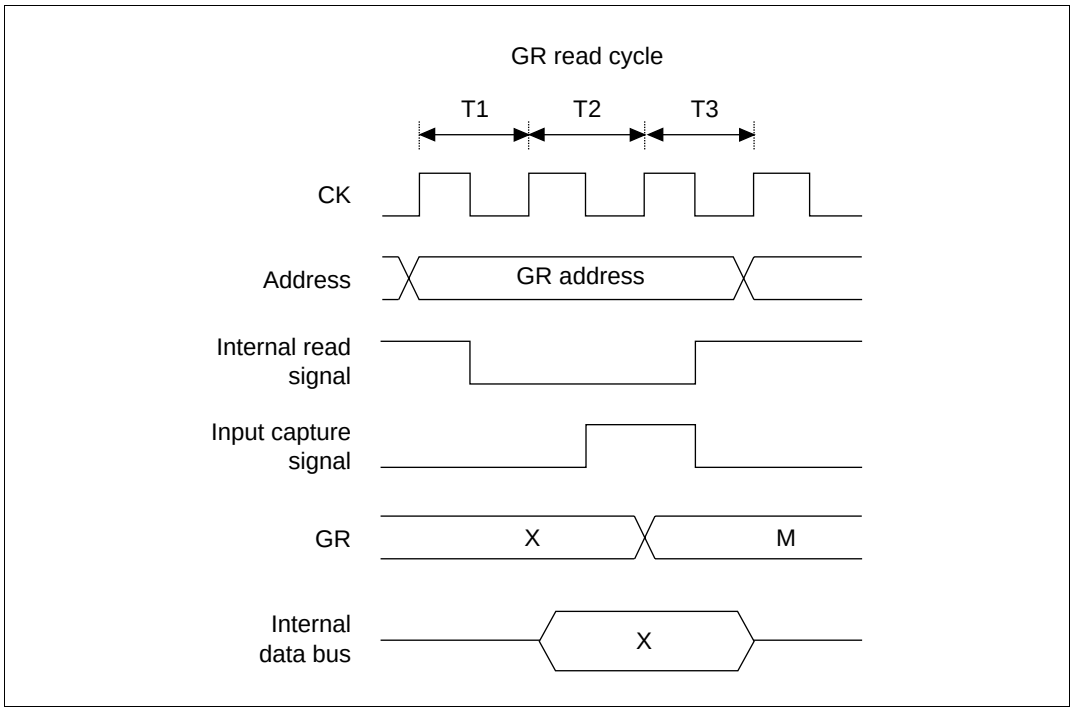



**Figure 10.63 Contention between General Register Read and Input Capture**

### 10.6.7 Contention Between Counter Clearing by Input Capture and Counter Increment

If an input capture signal and counter increment signal occur simultaneously, the counter is cleared according to the input capture signal. The counter is not incremented by the increment signal. The TCNT value before the counter is cleared is transferred to the general register. The timing is shown in figure 10.64.

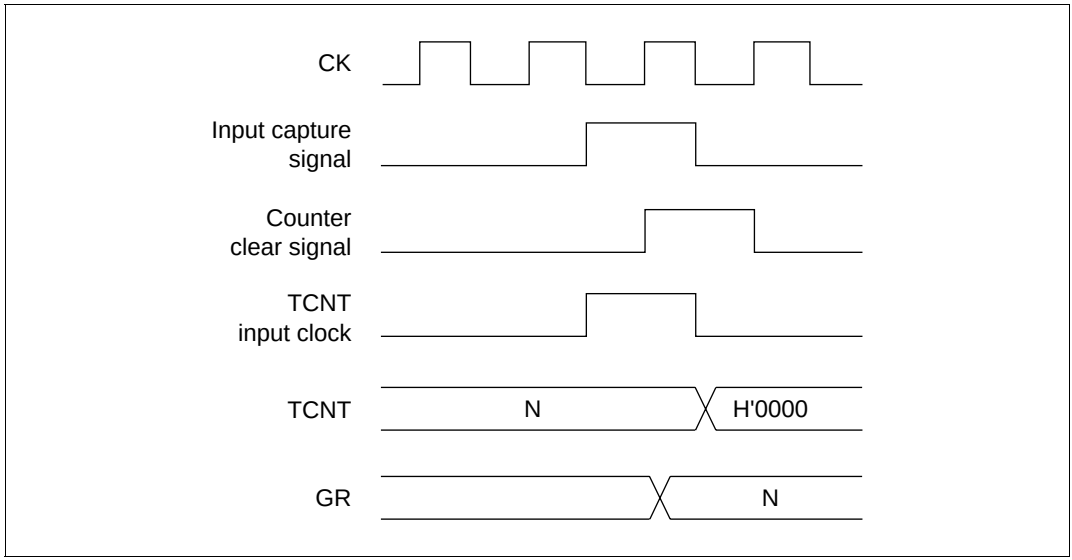



**Figure 10.64 Contention between Counter Clearing by Input Capture and Counter Increment**

### 10.6.8 Contention between General Register Write and Input Capture

If an input capture signal is generated during the T3 state of a general register write cycle, the input capture transfer takes priority and the write to the GR is not performed. The timing is shown in figure 10.65.

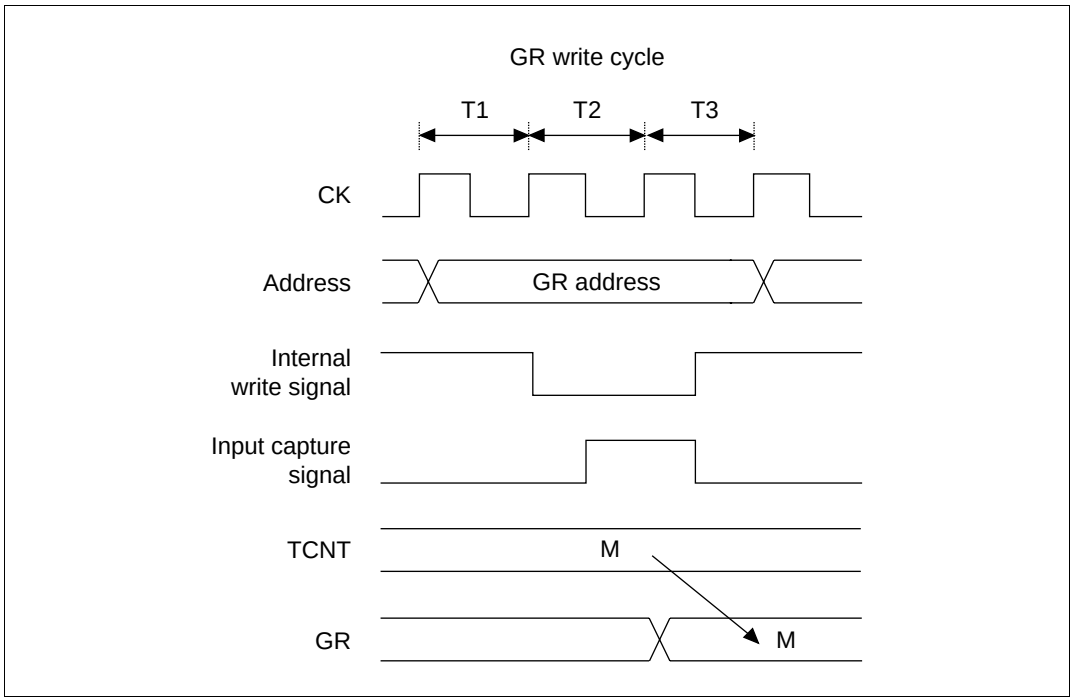



**Figure 10.65 Contention between General Register Write and Input Capture**

### 10.6.9 Note on Waveform Cycle Setting

When a counter is cleared by compare match, the counter is cleared in the last state in which the TCNT value matches the GR value (when the TCNT is updated from the matching count to the next count). The actual counter frequency is therefore given by the following formula:

$f = \phi/(N + 1)$

(f: counter frequency. $\phi$: operating frequency. N: value set in the GR.)

### 10.6.10 Contention Between BR Write and Input Capture

When a buffer register (BR) is being used as an input capture register and an input capture signal is generated in the T3 state of the write cycle, the buffer operation takes priority over the BR write. The timing is shown in figure 10.66.

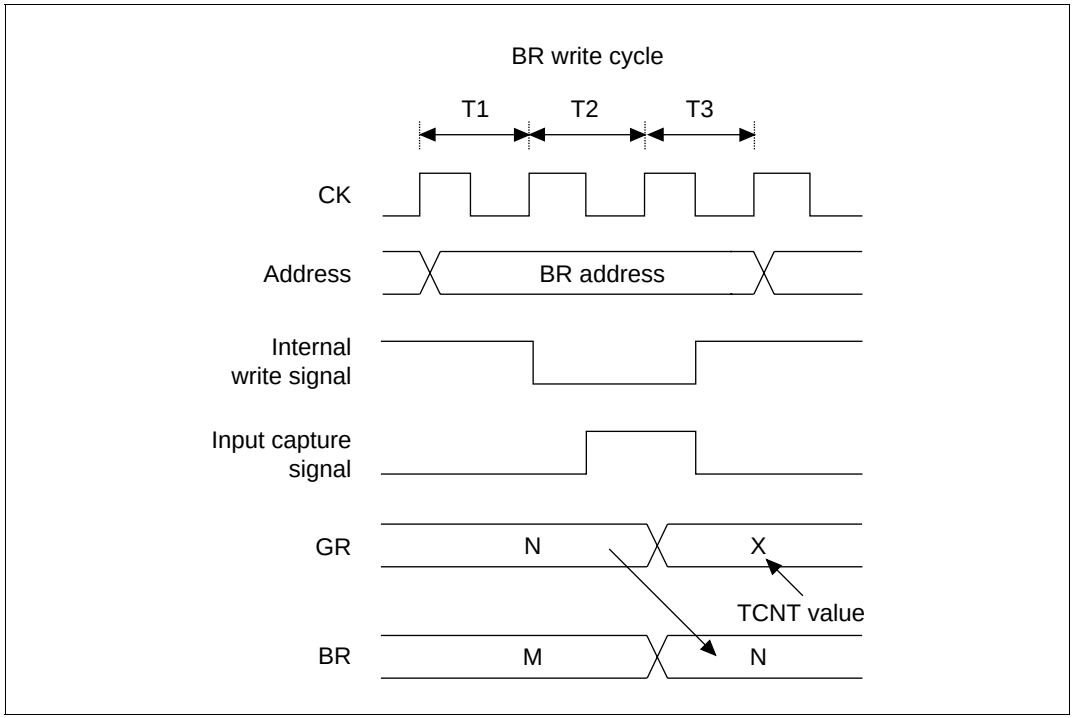



**Figure 10.66 Contention between BR Write and Input Capture**

### 10.6.11 Note on Writing in the Synchronizing Mode

After the synchronizing mode is selected, if the TCNT is written by byte access, all 16 bits of all synchronized counters assume the same value as the counter that was addressed.

Example: Figures 10.67 and 10.68 show byte write and word write when channels 2 and 3 are synchronized

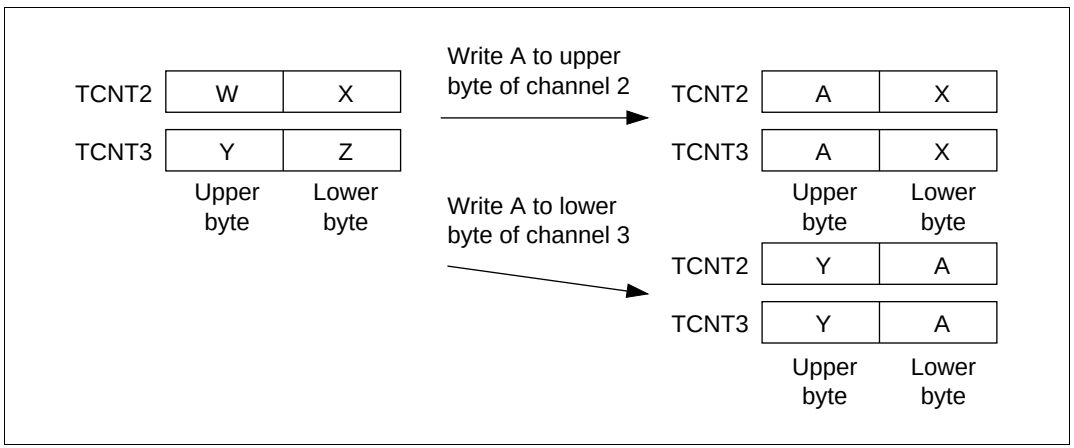



**Figure 10.67 Byte Write to Channel 2 or Byte Write to Channel 3**

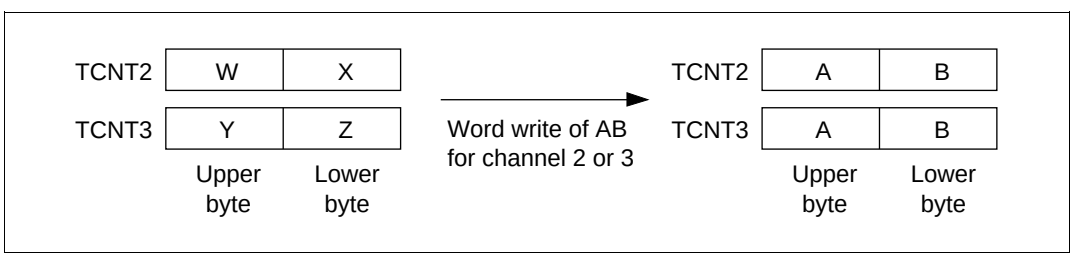



**Figure 10.68 Word Write to Channel 2 or Word Write to Channel 3**

### 10.6.12 Note on Setting Reset-synchronized PWM Mode/Complementary PWM Mode

When the CMD1 and CMD0 bits of TFCR are set, note the following.

1. Writes to CMD1 and CMD0 should be done while TCNT3 and TCNT4 are halted.
2. Changes of setting from the reset-synchronized PWM mode to the complementary PWM mode and vice versa are inhibited. Set the reset-synchronized PWM mode or complementary PWM mode after first setting normal operation (clear CMD1 bit to 0).

### 10.6.13 Clearing the Complementary PWM Mode

Figure 10.69 shows the procedure for clearing the complementary PWM mode. First, reset the combination mode bits CMD1 and CMD0 in the timer function control register (TFCR) from 10 to either 00 or 01. The mode will switch from complementary PWM mode to normal operating mode. Next, wait for at least 1 clock of the counter input clock being used for channels 3 and 4 and then clear the counter start bits STR3 and STR4 of the timer start register (TSTR). The channels 3 and 4 counters TCNT3 and TCNT4 will stop counting. Clearing the complementary PWM mode by any other procedure may result in changes other than those set for the output waveform when complementary PWM mode is set again.

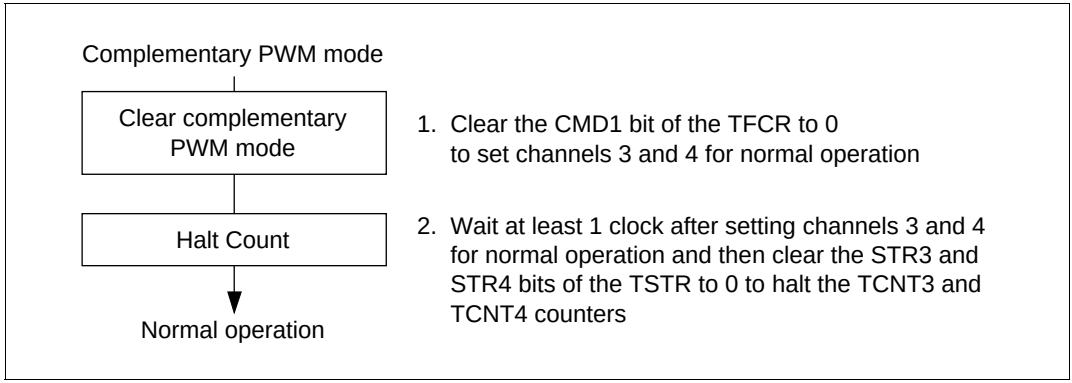



**Figure 10.69 Clearing the Complementary PWM Mode**

### 10.6.14 ITU Operating Modes

Tables 10.18–10.22 show the ITU operating modes for channels 0–4.

### 10.6.15 Note on Counter Clearing by Input Capture

If TCNT is cleared (to H'0000) by input capture when its value is H'FFFF, overflow will not occur.

**Table 10.18 ITU Operating Modes (Channel 0)**

| | Register Setting | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR0 | | TCR0 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buf-fer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Synch-ronized preset | SYNC0 = 1 | — | — | √ | — | — | — | — | √ | √ | √ | √ |
| PWM | √ | — | — | PWM0 = 1 | — | — | — | — | — | √ | √ | √ |
| Output compare A function | √ | — | — | PWM0 = 0 | — | — | — | — | IOA2 = 0, others free | √ | √ | √ |
| Output compare B function | √ | — | — | √ | — | — | — | — | √ | IOB2 = 0, others free | √ | √ |
| Input capture A function | √ | — | — | PWM0 = 0 | — | — | — | — | IOA2 = 1, others free | √ | √ | √ |
| Input capture B function | √ | — | — | PWM0 = 0 | — | — | — | — | √ | IOB2 = 1, others free | √ | √ |
| **Counter Clear Function** | | | | | | | | | | | | |
| Clear at compare match/ input capture A | √ | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 0 CCLR0 = 1 | √ |
| Clear at compare match/ input capture B | √ | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 1 CCLR0 = 0 | √ |
| Synch-ronized clear | SYNC0 = 1 | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 1 CCLR0 = 1 | √ |

√: Settable, —: Setting does not affect current mode

Note: In PWM mode, the input capture function cannot be used. When compare match A and compare match B occur simultaneously, the compare match signal is inhibited.

**Table 10.19 ITU Operating Modes (Channel 1)**

| | Register Setting | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR1 | | TCR1 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buf-fer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Synch-ronized preset | SYNC1 = 1 | — | — | √ | — | — | — | — | √ | √ | √ | √ |
| PWM | √ | — | — | PWM1 = 1 | — | — | — | — | — | √*1 | √ | √ |
| Output compare A function | √ | — | — | PWM1 = 0 | — | — | — | — | IOA2 = 0, others free | √ | √ | √ |
| Output compare B function | √ | — | — | √ | — | — | — | — | √ | IOB2 = 0, others free | √ | √ |
| Input capture A function | √ | — | — | PWM1 = 0 | — | — | — | — | IOA2 = 1, others free | √ | √ | √ |
| Input capture B function | √ | — | — | PWM1 = 0 | — | — | — | — | √ | IOB2 = 1, others free | √ | √ |
| **Counter Clear Function** | | | | | | | | | | | | |
| Clear at compare match/ input capture A | √ | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 0 CCLR0 = 1 | √ |
| Clear at compare match/ input capture B | √ | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 1 CCLR0 = 0 | √ |
| Synch-ronized clear | SYNC1 = 1 | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 1 CCLR0 = 1 | √ |

√: Settable, —: Setting does not affect current mode

Note: In PWM mode, the input capture function cannot be used. When compare match A and compare match B occur simultaneously, the compare match signal is inhibited.

**Table 10.20 ITU Operating Modes (Channel 2)**

| | Register Setting | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR2 | | TCR2 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buf-fer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Synch-ronized preset | SYNC2 = 1 | — | — | √ | — | — | — | — | √ | √ | √ | √ |
| PWM | √ | — | — | PWM2 = 1 | — | — | — | — | — | √ | √ | √ |
| Output compare A function | √ | — | — | PWM2 = 0 | — | — | — | — | IOA2 = 0, others free | √ | √ | √ |
| Output compare B function | √ | — | — | √ | — | — | — | — | √ | IOB2 = 0, others free | √ | √ |
| Input capture A function | √ | — | — | PWM2 = 0 | — | — | — | — | IOA2 = 1, others free | √ | √ | √ |
| Input capture B function | √ | — | — | PWM2 = 0 | — | — | — | — | √ | IOB2 = 1, others free | √ | √ |
| **Counter Clear Function** | | | | | | | | | | | | |
| Clear at compare match/ input capture A | √ | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 0 CCLR0 = 1 | √ |
| Clear at compare match/ input capture B | √ | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 1 CCLR0 = 0 | √ |
| Synch-ronized clear | SYNC2 = 1 | — | — | √ | — | — | — | — | √ | √ | CCLR1 = 1 CCLR0 = 1 | √ |
| Phase counting | √ | MDF = 1 | √ | √ | — | — | — | — | √ | √ | √ | — |

√: Settable, —: Setting does not affect current mode

Note: In PWM mode, the input capture function cannot be used. When compare match A and compare match B occur simultaneously, the compare match signal is inhibited.

**Table 10.21 ITU Operating Modes (Channel 3)**

| | Register Setting | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR3 | | TCR3 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buffer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Synchronized preset | SYNC3 = 1 | — | — | √ | √*2 | √ | √ | — | √ | √ | √ | √ |
| PWM mode | √ | — | — | PWM3 = 1 | CMD1 = 0 | CMD1 = 0 | √ | — | — | √*1 | √ | √ |
| Output compare A function | √ | — | — | PWM3 = 0 | CMD1 = 0 | CMD1 = 0 | √ | — | IOA2 = 0, others free | √ | √ | √ |
| Output compare B function | √ | — | — | √ | CMD1 = 0 | CMD1 = 0 | √ | — | √ | IOB2 = 0, others free | √ | √ |
| Input capture A function | √ | — | — | PWM3 = 0 | CMD1 = 0 | CMD1 = 0 | √ | — | IOA2 = 1, others free | √ | √ | √ |
| Input capture B function | √ | — | — | PWM3 = 0 | CMD1 = 0 | CMD1 = 0 | √ | — | √ | IOB2 = 1, others free | √ | √ |
| **Counter Clear Function** | | | | | | | | | | | | |
| Clear at compare match/ input capture A | √ | — | — | √ | CMD1 = 1, CMD0 = 0 inhibited | √*3 | √ | — | √ | √ | CCLR1 = 0 CCLR0 = 1 | √ |
| Clear at compare match/ input capture B | √ | — | — | √ | CMD1 = 0 | CMD1 = 0 | √ | — | √ | √ | CCLR1 = 1 CCLR0 = 0 | √ |
| Synchronized clear | SYNC3 = 1 | — | — | √ | CMD1 = 1, CMD0 = 0 inhibited | √ | √ | — | √ | √ | CCLR1 = 1 CCLR0 = 1 | √ |

**Table 10.21 ITU Operating Modes (Channel 3) (cont)**

| | Counter Clear Function | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Register Setting | | | | | | | | | | | |
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR3 | | TCR3 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buf-fer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Comple-mentary PWM mode | √*2 | — | — | — | CMD1 = 1 CMD0 = 0 | CMD1 = 1 CMD0 = 0 | √ | √ | — | — | CCLR1 = 0 CCLR0 = 0 | √*4 |
| Reset synchron-ized PWM mode | √ | — | — | — | CMD1 = 1 CMD0 = 1 | CMD1 = 1 CMD0 = 1 | √ | √ | — | — | CCLR1 = 0 CCLR0 = 1 | √ |
| Buffer (BRA) | √ | — | — | √ | √ | √ | BFA3 = 1, others free | — | √ | √ | √ | √ |
| Buffer (BRB) | √ | — | — | √ | √ | √ | BFB3 = 1, others free | — | √ | √ | √ | √ |

√: Settable, —: Setting does not affect current mode

Notes: 1. In PWM mode, the input capture function cannot be used. When compare match A and compare match B occur simultaneously, the compare match signal is inhibited.

2. When set for complementary PWM mode, do not simultaneously set channel 3 and channel 4 to function synchronously.

3. Counter clearing by input capture A cannot be used when the reset-synchronized PWM mode is set.

4. Clock selection when the complementary PWM mode is set should be the same for channels 3 and 4.

**Table 10.22 ITU Operating Modes (Channel 4)**

| | Register Setting | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR4 | | TCR4 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buf-fer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Synch-ronized preset | SYNC4 = 1 | — | — | √ | √*2 | √ | √ | — | √ | √ | √ | √ |
| PWM | √ | — | — | PWM4 = 1 | CMD1 = 0 | CMD1 = 0 | √ | — | — | √*1 | √ | √ |
| Output compare A function | √ | — | — | PWM4 = 0 | CMD1 = 0 | CMD1 = 0 | √ | — | IOA2 = 0, others free | √ | √ | √ |
| Output compare B function | √ | — | — | √ | CMD1 = 0 | CMD1 = 0 | √ | — | √ | IOB2 = 0, others free | √ | √ |
| Input capture A function | √ | — | — | PWM4 = 0 | CMD1 = 0 | CMD1 = 0 | √ | — | IOA2 = 1, others free | √ | √ | √ |
| Input capture B function | √ | — | — | PWM4 = 0 | CMD1 = 0 | CMD1 = 0 | √ | — | √ | IOB2 = 1, others free | √ | √ |
| **Counter Clear Function** | | | | | | | | | | | | |
| Clear at compare match/ input capture A | √ | — | — | √ | CMD1 = 1, CMD0 = 0 inhibited | √*3 | √ | — | √ | √ | CCLR1 = 0 CCLR0 = 1 | √ |
| Clear at compare match/ input capture B | √ | — | — | √ | CMD1 = 1, CMD0 = 0 inhibited | √*3 | √ | — | √ | √ | CCLR1 = 1 CCLR0 = 0 | √ |

**Table 10.22 ITU Operating Modes (Channel 4) (cont)**

**Counter Clear Function**

| | Register Setting | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | TSNC | TMDR | | | TFCR | | | TOCR | TIOR4 | | TCR4 | |
| Operating Mode | Sync | MDF | FDIR | PWM | Comp PWM | Reset Sync PWM | Buf-fer | Output Level Select | IOA | IOB | Clear Select | Clock Select |
| Synch-ronized clear | SYNC4 = 1 | — | — | √ | CMD1 = 1, CMD1 = 0 inhibited | √*3 | √ | — | √ | √ | CCLR1 = 1 CCLR0 = 1 | √ |
| Comple-mentary PWM | √*2 | — | — | — | CMD1 = 1 CMD0 = 0 | CMD1 = 1 CMD0 = 0 | √ | √ | — | — | CCLR1 = 0 CCLR0 = 0 | √*4 |
| Reset synchron-ized PWM | √ | — | — | — | CMD1 = 1 CMD0 = 1 | CMD1 = 1 CMD0 = 1 | √ | √ | — | — | √*5 | √*5 |
| Buffer (BRA) | √ | — | — | √ | √ | √ | BFA4 = 1, others free | — | √ | √ | √ | √ |
| Buffer (BRB) | √ | — | — | √ | √ | √ | BFB4 = 1, others free | — | √ | √ | √ | √ |

√: Settable, —: Setting does not affect current mode

Notes:
1. In PWM mode, the input capture function cannot be used. When compare match A and compare match B occur simultaneously, the compare match signal is inhibited.
2. When set for complementary PWM mode, do not simultaneously set channel 3 and channel 4 to function synchronously.
3. Counter clearing works with the reset-synchronized PWM mode, but TCNT4 runs independently. The output waveform is not affected.
4. Clock selection when the complementary PWM mode is set should be the same for channels 3 and 4.
5. In the reset-synchronized PWM mode, TCNT4 runs independently. The output waveform is not affected.

# Section 11 Programmable Timing Pattern Controller (TPC)

## 11.1 Overview

The SuperH microcomputer has a built-in programmable timing pattern controller (TPC). The TPC can provide pulse outputs by using the 16-bit integrated-timer pulse unit (ITU) as a time base. The TPC pulse outputs are divided into 4-bit groups 3–0. These can operate simultaneously, or independently.

### 11.1.1 Features

Features of the programmable timing pattern controller are listed below.

- 16-bit output data: Maximum 16-bit data can be output. TPC output can be enabled on a bit-by-bit basis.
- Four output groups: Output trigger signals can be selected in 4-bit groups to provide up to four different 4-bit outputs.
- Selectable output trigger signals: Output trigger signals can be selected by group from the
- 4-channel compare-match signals of the 16-bit integrated-timer pulse unit (ITU).
- Non-overlap mode: A non-overlap interval can be set to come between multiple pulse outputs.
- Can connect to DMA controller: The compare-match signals selected as output trigger signals can activate the DMA controller for sequential output of data without CPU intervention.

### 11.1.2 Block Diagram

Figure 11.1 is the block diagram of the TPC.

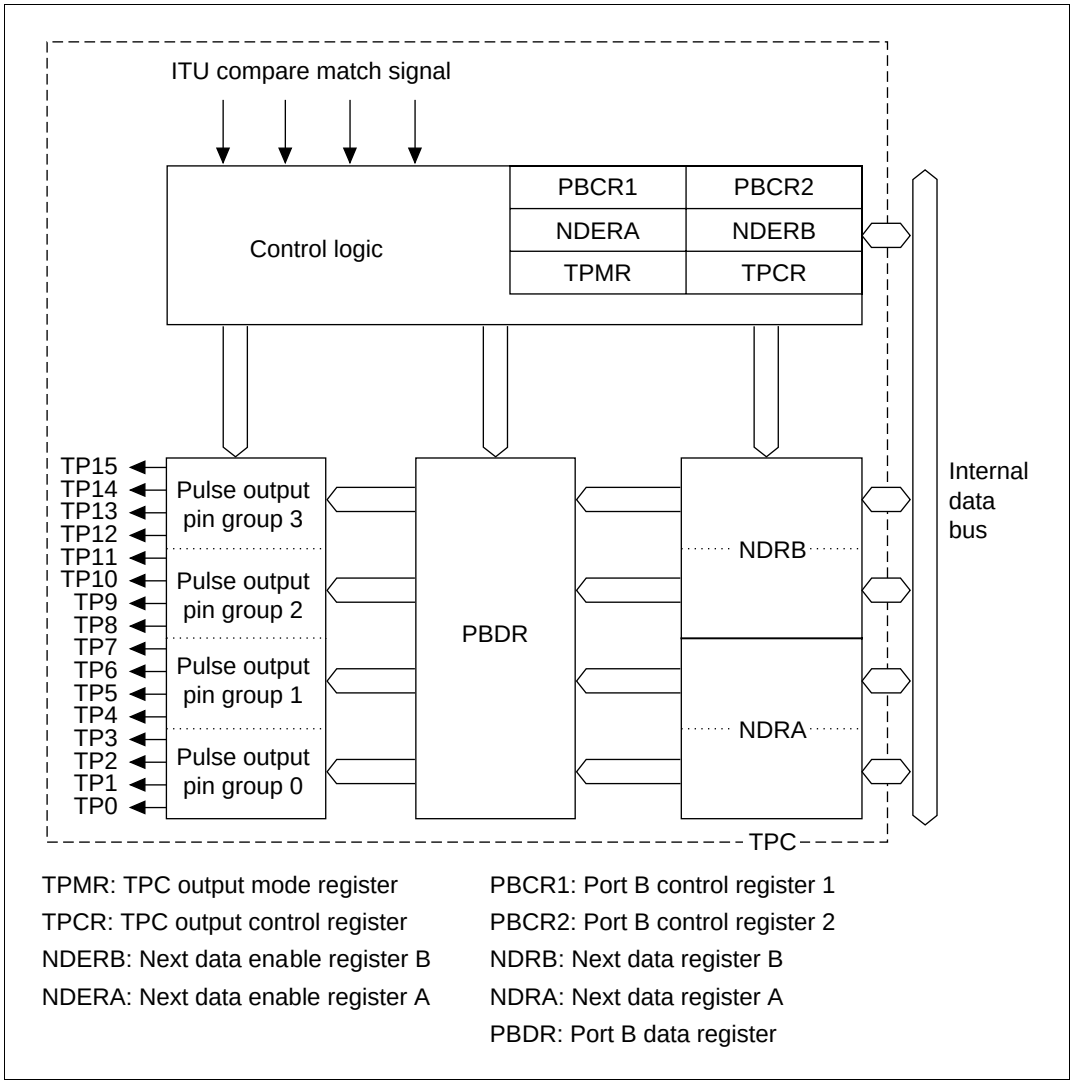



**Figure 11.1 TPC Block Diagram**

### 11.1.3 Input/Output Pins

Table 11.1 summarizes the TPC input/output pins.

**Table 11.1 TPC Pins**

| Name | Symbol | Input/Output | Function |
|---|---|---|---|
| TPC output 0 | TP0 | Output | Group 0 pulse output |
| TPC output 1 | TP1 | Output | |
| TPC output 2 | TP2 | Output | |
| TPC output 3 | TP3 | Output | |
| TPC output 4 | TP4 | Output | Group 1 pulse output |
| TPC output 5 | TP5 | Output | |
| TPC output 6 | TP6 | Output | |
| TPC output 7 | TP7 | Output | |
| TPC output 8 | TP8 | Output | Group 2 pulse output |
| TPC output 9 | TP9 | Output | |
| TPC output 10 | TP10 | Output | |
| TPC output 11 | TP11 | Output | |
| TPC output 12 | TP12 | Output | Group 3 pulse output |
| TPC output 13 | TP13 | Output | |
| TPC output 14 | TP14 | Output | |
| TPC output 15 | TP15 | Output | |

### 11.1.4 Registers

Table 11.2 summarizes the TPC registers.

**Table 11.2 TPC Registers**

| Name | Abbreviation | R/W | Initial Value | Address*[1] | Access Size` |
|---|---|---|---|---|---|
| Port B control register 1 | PBCR1 | R/W | H'0000 | H'5FFFFCC | 8, 16 |
| Port B control register 2 | PBCR2 | R/W | H'0000 | H'5FFFFCE | 8, 16 |
| Port B data register | PBDR | R/(W)*[2] | H'0000 | H'5FFFFC2 | 8, 16 |
| TPC output mode register | TPMR | R/W | H'F0 | H'5FFFFF0 | 8, 16 |
| TPC output control register | TPCR | R/W | H'FF | H'5FFFFF1 | 8, 16 |
| Next data enable register B | NDERB | R/W | H'00 | H'5FFFFF2 | 8, 16 |
| Next data enable register A | NDERA | R/W | H'00 | H'5FFFFF3 | 8, 16 |
| Next data register A | NDRA | R/W | H'00 | H'5FFFFF5/ H'5FFFFF7*[3] | 8, 16 |
| Next data register B | NDRB | R/W | H'00 | H'5FFFFF4/ H'5FFFFF6*[3] | 8, 16 |

Notes: 1. Only the values of bits A27–A24 and A8–A0 are valid; bits A23–A9 are ignored. For details on the register addresses, see section 8.3.5, Description of Areas.
2. Bits used for TPC output cannot be written to.
3. These addresses change depending on TPCR settings. When TPC output groups 0 and 1 have the same output trigger, the NDRA address is H'5FFFFF5; when their output triggers are different, the NDRA address for group 0 is H'5FFFFF7 and the address for group 1 is H'5FFFFF5. Likewise, when TPC output groups 2 and 3 have the same output trigger, the NDRB address is H'5FFFFF4; when their output triggers are different, the NDRB address for group 0 is H'5FFFFF6 and the address for group 1 is H'5FFFFF4.

## 11.2 Register Descriptions

### 11.2.1 Port B Control Registers 1 and 2 (PBCR1, PCBR2)

The port B control register 1 and 2 (PBCR1 and PBCR2) are 16-bit read/write registers that set the functions of port B pins. Port B consists of the dual use pins TP15–TP0. Bits corresponding to the pins to be used for TPC output must be set to 1. For details, see the port B description in the section 15, Pin Function Controller.

**PCBR1:**

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB15 MD1 | PB15 MD0 | PB14 MD1 | PB14 MD0 | PB13 MD1 | PB13 MD0 | PB12 MD1 | PB12 MD0 |
| 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: | PB11 MD1 | PB11 MD0 | PB10 MD1 | PB10 MD0 | PB9MD1 | PB9MD0 | PB8MD1 | PB8MD0 |
| 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 |

**PCBR2:**

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB7MD1 | PB7MD0 | PB6MD1 | PB6MD0 | PB5MD1 | PB5MD0 | PB4MD1 | PB4MD0 |
| 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: | PB3MD1 | PB3MD0 | PB2MD1 | PB2MD0 | PB1MD1 | PB1MD0 | PB0MD1 | PB0MD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

### 11.2.2 Port B Data Register (PBDR)

The port B data register is a 16-bit read/write register that, when used for TPC output stores, output data for groups 0–3. For details, see the port B description in section 16, I/O Ports.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB15DR | PB14DR | PB13DR | PB12DR | PB11DR | PB10DR | PB9DR | PB8DR |
| 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: Bits set to TPC output by NDERA or NDERB are read-only.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: Bits set to TPC output by NDERA or NDERB are read-only.

### 11.2.3 Next Data Register A (NDRA)

NDRA is an eight-bit read/write register that stores the next output data for TPC output groups 1 and 0 (TP7–TP0). When used for TPC output, the contents of the NDRA are transferred to the corresponding PBDR bits when the ITU compare match specified in the TPC output control register TPCR occurs.

The address of the NDRA differs depending on whether TPCR settings select the same trigger or different triggers for TPC output groups 1 and 0. When reset, NDRA is initialized to H'00. It is not initialized by standby mode.

**Same Trigger for TPC Output Groups 1 and 0:** If TPC output groups 1 and 0 are triggered by the same compare match, the address of the NDRA is H'FFFFF5. The 4 upper bits becomes group 1 and the 4 lower bits become group 0. Address H'5FFFFF7 in such cases consists entirely of reserved bits. These bits cannot be modified and always read as 1.

Address H'5FFFFF5:

- Bits 7–4 (next data 7–4 (NDR7–NDR4)): NDR7-NDR4 store the next output data for TPC output group 1.
- Bits 3–0 (next data 3–0 (NDR3–NDR0)): NDR3-NDR0 store the next output data for TPC output group 0.

| 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 |

Address H'5FFFFF7:

- Bits 7–0 (reserved): These bits always read as 1. The write value should always be 1.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | — | — | — |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | — | — | — | — | — | — | — | — |

Different Triggers for TPC Output Groups 1 and 0: If TPC output groups 1 and 0 are triggered by different compare matches, the address of the upper 4 bits of NDRA (group 1) is H'5FFFFF5 and the address of the lower 4 bits of NDRA (group 0) is H'5FFFFF7. Bits 3–0 of address H'5FFFFF5 and bits 7–4 of address H'5FFFFF7 are reserved bits. The write value should always be 1. These bits always read as 1.

Address H'5FFFFF5:

- Bits 7–4 (next data 7–4 (NDR7–NDR4)): NDR7–NDR4 store the next output data for TPC output group 1.
- Bits 3–0 (reserved): These bits always read as 1. The write value should always be 1.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | NDR7 | NDR6 | NDR5 | NDR4 | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| R/W: | R/W | R/W | R/W | R/W | — | — | — | — |

Address H'5FFFFF7:

- Bits 7–4 (reserved): These bits always read as 1. The write value should always be 1.
- Bits 3–0 (next data 3–0 (NDR3–NDR0)): NDR3-NDR0 store the next output data for TPC output group 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value: | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | — | R/W | R/W | R/W | R/W |

### 11.2.4 Next Data Register B (NDRB)

NDRB is an eight-bit read/write register that stores the next output data for TPC output groups 3 and 2 (TP15–TP8). When used for TPC output, the contents of the NDRB are transferred to the corresponding PBDR bits when the ITU compare match specified in the TPC output control register TPCR occurs.

The address of the NDRB differs depending on whether TPCR settings select the same trigger or different triggers for TPC output groups 3 and 2. When reset, NDRB is initialized to H'00. It is not initialized by standby mode.

Same Trigger for TPC Output Groups 3 and 2: If TPC output groups 3 and 2 are triggered by the same compare match, the address of the NDRB is H'FFFFF4. The 4 upper bits becomes group 3 and the 4 lower bits become group 2. Address H'5FFFFF6 becomes completely reserved bits. These bits always read as 1, and the write value should always be 1.

Address H'5FFFFF4:

- Bits 7–4 (next data 15–12 (NDR15–NDR12)): NDR15–NDR12 store next output data for TPC output group 3.
- Bits 3–0 (next data 11–8 (NDR11–NDR8)): NDR11–NDR8 store next output data for TPC output group 2.

| 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 |

Address H'5FFFFF6:

- Bits 7–0 (reserved): These bits always read as 1. The write value should always be 1.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | — | — | — | — |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | — | — | — | — | — | — | — | — |

**Different Triggers for TPC Output Groups 3 and 2:** If TPC output groups 3 and 2 are triggered by different compare matches, the address of the upper 4 bits of NDRB (group 3) is H'5FFFFF4 and the address of the lower 4 bits of NDRB (group 2) is H'5FFFFF6. Bits 3-0 of address H'5FFFFF4 and bits 7–4 of address H'5FFFFF6 are reserved bits. These bits always read as 1. The write value should always be 1.

Address H'5FFFFF4:

- Bits 7–4 (next data 15–12 (NDR15–NDR12)): NDR15–NDR12 store next output data for TPC output group 3.
- Bits 3–0 (reserved): These bits always read as 1. The write value should always be 1.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | NDR15 | NDR14 | NDR13 | NDR12 | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| R/W: | R/W | R/W | R/W | R/W | — | — | — | — |

Address H'5FFFFF6:

- Bits 7–4 (reserved): These bits always read as 1. The write value should always be 1.
- Bits 3–0 (next data 11–8 (NDR11–NDR8)): NDR11–NDR8 store next output data for TPC output group 2.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | NDR11 | NDR10 | NDR9 | NDR8 |
| Initial value: | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | — | R/W | R/W | R/W | R/W |

### 11.2.5 Next Data Enable Register A (NDERA)

NDERA is an eight-bit read/write register that enables TPC output groups 1 and 0 (TP7–TP0) on a bit-by-bit basis.

When the bits enabled for TPC output by NDERA generate the ITU compare match selected in the TPC output control register, the value of the next data register A (NDRA) is automatically transferred to the corresponding PBDR bits and the output value is updated. For disabled bits, there is no transfer and the output value does not change. When reset, NDERA is initialized to H'00. It is not initialized by standby mode.

| 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 |

- Bits 7–0 (next data enable 7–0 (NDER7–NDER0)): NDER7–NDER0 select enable/disable for TPC output groups 1 and 0 (TP7–TP0) in bit units.

| Bit 7–0: NDER7–NDER0 | Description |
|---|---|
| 0 | Disables TPC outputs TP7–TP0 (transfer from NDR7–NDR0 to PB7–PB0 is disabled) (initial value) |
| 1 | Enables TPC outputs TP7–TP0 (transfer from NDR7–NDR0 to PB7–PB0 is enabled) |

### 11.2.6 Next Data Enable Register B (NDERB)

NDERB is an eight-bit read/write register that enables TPC output groups 3 and 2 (TP15–TP8) on a bit-by-bit basis.

When the bits enabled for TPC output by NDERB generate the ITU compare match selected in the TPC output control register, the value of the next data register B (NDRB) is automatically transferred to the corresponding PBDR bits and the output value is updated. For disabled bits, there is no transfer and the output value does not change. When reset, NDERB is initialized to H'00. It is not initialized by standby mode.

| 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 |

- Bits 7–0 (next data enable 15–8 (NDER15–NDER8)): NDER15–NDER8 select enable/disable for TPC output groups 3 and 2 (TP15–TP8) in bit units.

| Bit 7–0: NDER15–NDER8 | Description |
|---|---|
| 0 | Disables TPC outputs TP15–TP8 (transfer from NDR15–NDR8 to PB15–PB8 is disabled) (initial value) |
| 1 | Enables TPC outputs TP15–TP8 (transfer from NDR15–NDR8 to PB15–PB8 is enabled) |

### 11.2.7 TPC Output Control Register (TPCR)

TPCR is an eight-bit read/write register that selects output trigger signals for TPC outputs. When reset, TPCR is initialized to H'FF. It is not initialized by standby mode.

| 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 |

- Bits 7 and 6 (group 3 compare-match select 1 and 0 (G3CMS1 and G3CMS0)): G3CMS1 and G3CMS0 select the compare match that triggers TPC output group 3 (TP15–TP12).

| Bit 7: G3CMS1 | Bit 6: G3CMS0 | Description |
|---|---|---|
| 0 | 0 | TPC output group 3 (TP15–TP12) output is triggered by compare-match in ITU channel 0 |
| | 1 | TPC output group 3 (TP15–TP12) output is triggered by compare-match in ITU channel 1 |
| 1 | 0 | TPC output group 3 (TP15–TP12) output is triggered by compare-match in ITU channel 2 |
| | 1 | TPC output group 3 (TP15–TP12) output is triggered by compare-match in ITU channel 3 (initial value) |

- Bits 5 and 4 (group 2 compare-match select 1 and 0 (G2CMS1 and G2CMS0)): G2CMS1 and G2CMS0 select the ITU channel that triggers TPC output group 2 (TP11–TP8).

| Bit 5: G2CMS1 | Bit 4: G2CMS0 | Description |
|---|---|---|
| 0 | 0 | TPC output group 2 (TP11–TP18) output is triggered by compare-match in ITU channel 0 |
| | 1 | TPC output group 2 (TP11–TP18) output is triggered by compare-match in ITU channel 1 |
| 1 | 0 | TPC output group 2 (TP11–TP18) output is triggered by compare-match in ITU channel 2 |
| | 1 | TPC output group 2 (TP11–TP18) output is triggered by compare-match in ITU channel 3 (initial value) |

Bits 3 and 2 (group 1 compare-match select 1 and 0 (G1CMS1 and G1CMS0)): G1CMS1 and G1CMS0 select the ITU channel that triggers TPC output group 1 (TP7–TP4).

| Bit 3: G1CMS1 | Bit 2: G1CMS0 | Description |
|---|---|---|
| 0 | 0 | TPC output group 1 (TP7–TP4) output is triggered by compare-match in ITU channel 0 |
| | 1 | TPC output group 1 (TP7–TP4) output is triggered by compare-match in ITU channel 1 |
| 1 | 0 | TPC output group 1 (TP7–TP4) output is triggered by compare-match in ITU channel 2 |
| | 1 | TPC output group 1 (TP7–TP4) output is triggered by compare-match in ITU channel 3 (initial value) |

- Bits 1 and 0 (group 0 compare-match select 1 and 0 (G0CMS1 and G0CMS0)): G0CMS1 and G0CMS0 select the ITU channel that triggers TPC output group 0 (TP3–TP0).

| Bit 1: G0CMS1 | Bit 0: G0CMS0 | Description |
|---|---|---|
| 0 | 0 | TPC output group 0 (TP3–TP0) output is triggered by compare-match in ITU channel 0 |
| | 1 | TPC output group 0 (TP3–TP0) output is triggered by compare-match in ITU channel 1 |
| 1 | 0 | TPC output group 0 (TP3–TP0) output is triggered by compare-match in ITU channel 2 |
| | 1 | TPC output group 0 (TP3–TP0) output is triggered by compare-match in ITU channel 3 (initial value) |

### 11.2.8 TPC Output Mode Register (TPMR)

TPMR is an eight-bit read/write register that selects between the TPC's ordinary output and non-overlap output modes in group units. During non-overlap operation, the output waveform cycle is set in ITU general register B (GRB) for use as the output trigger and a non-overlap period is set in general register A (GRA). The output value then changes on compare matches A and B. For details, see section 11.3.4, TPC Output Non-Overlap Operation. TPMR is initialized to H'F0 on a reset. It is not initialized in standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | — | — | — | G3NOV | G2NOV | G1NOV | G0NOV |
| Initial value: | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | — | R/W | R/W | R/W | R/W |

Bits 7–4 (reserved): These bits always read as 1. The write value should always be 1.

- Bit 3 (group 3 non-overlap mode (G3NOV)): G3NOV selects the ordinary or non-overlap mode for TPC output group 3 (TP15–TP12).

| Bit 3: G3NOV | Description |
|---|---|
| 0 | TPC output group 3 operates normally (output value updated according to compare-match A of the ITU channel selected by TPCR) (initial value) |
| 1 | TPC output group 3 operates in non-overlap mode (1 output and 0 output can be performed independently according to compare-match A and B of the ITU channel selected by TPCR) |

- Bit 2 (group 2 non-overlap mode (G2NOV)): G2NOV selects the ordinary or non-overlap mode for TPC output group 2 (TP11–TP8).

| Bit 2: G2NOV | Description |
|---|---|
| 0 | TPC output group 2 operates normally (output value updated according to compare-match A of the ITU channel selected by TPCR) (initial value) |
| 1 | TPC output group 2 operates in non-overlap mode (1 output and 0 output can be performed independently according to compare-match A and B of the ITU channel selected by TPCR) |

- Bit 1 (group 1 non-overlap mode (G1NOV)): G1NOV selects the ordinary or non-overlap mode for TPC output group 1 (TP7–TP4).

| Bit 1: G1NOV | Description |
|---|---|
| 0 | TPC output group 1 operates normally (output value updated according to compare-match A of the ITU channel selected by TPCR) (initial value) |
| 1 | TPC output group 1 operates in non-overlap mode (1 output and 0 output can be performed independently according to compare-match A and B of the ITU channel selected by TPCR) |

- Bit 0 (group 0 non-overlap mode (G0NOV)): G0NOV selects the ordinary or non-overlap mode for TPC output group 0 (TP3–TP0).

| Bit 0: G0NOV | Description |
|---|---|
| 0 | TPC output group 0 operates normally (output value updated according to compare-match A of the ITU channel selected by TPCR) (initial value) |
| 1 | TPC output group 0 operates in non-overlap mode (1 output and 0 output can be performed independently according to compare-match A and B of the ITU channel selected by TPCR) |

## 11.3 Operation

### 11.3.1 Overview

When corresponding bits in the PBCR1, PBCR2, NDERA and NDERB registers are set to 1, TPC output is enabled and the PBDR data register values are output. After that, when the compare-match event selected by TPCR occurs, the next data register contents (NDRA and NDRB) are transferred to the PBDR and output values are updated. Figure 11.2 illustrates the TPC output operation.

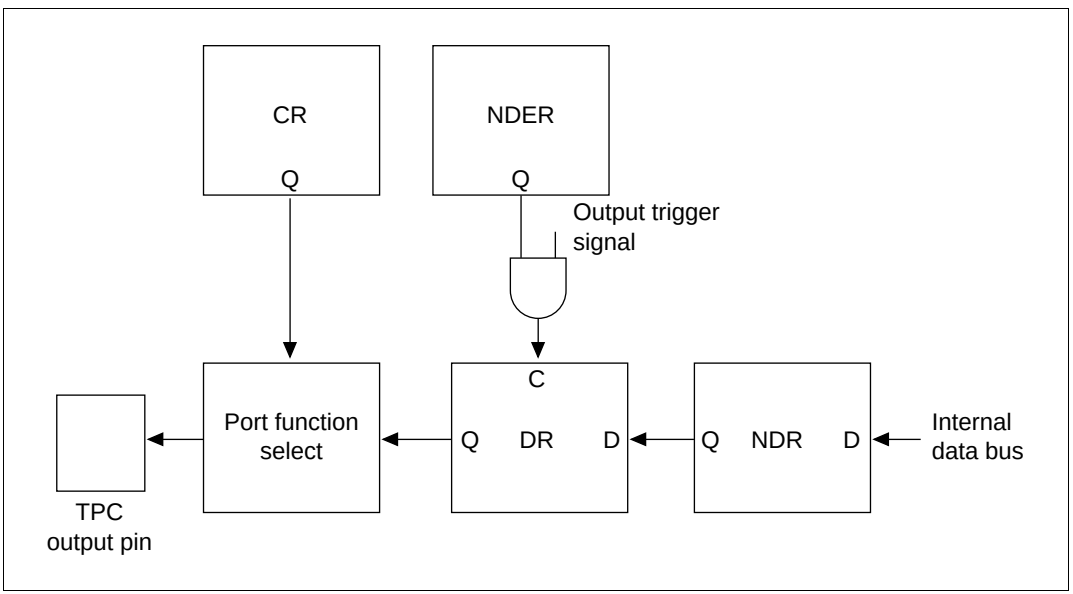



**Figure 11.2 TPC Output Operation**

If new data is written in next data registers A and B before the next compare-match occurs, a maximum 16 bits of data can be output at each successive compare-match. See section 11.3.4, TPC Output Non-Overlap Operation, for details on non-overlap operation.

### 11.3.2 Output Timing

If TPC output is enabled, next data register (NDRA/NDRB) contents are transferred to the data register (PBDR) and output when the selected compare-match occurs. Figure 11.3 shows the timing of these operations. The example is of ordinary output upon compare match A with groups 2 and 3.

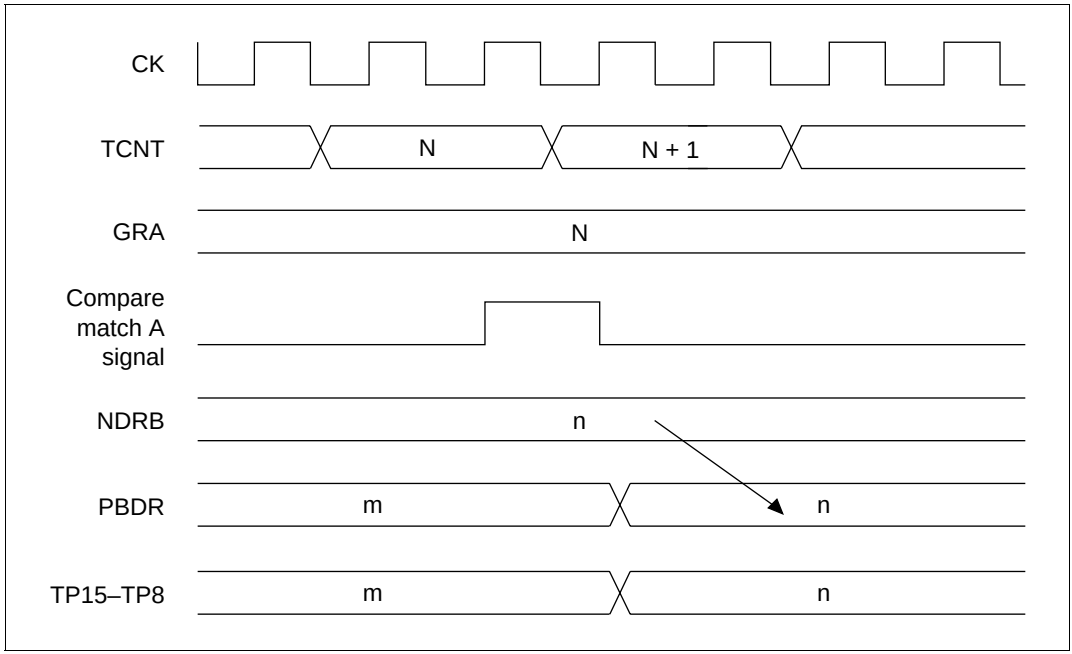



**Figure 11.3 Transfer and Output Timing for NDR Data**

### 11.3.3 Examples of Use of Ordinary TPC Output

**Settings for Ordinary TPC Output (figure 11.4):**

1. Select GRA as the output compare register (output disable) with the timer I/O control register (TIOR).
2. Set the TPC output trigger cycle.
3. Select the counter clock with the TPSC2–TPSC0 bits of the timer control register (TCR). Select the counter clear sources with the CCLR1 and CCLR0 bits.
4. Set the timer interrupt enable register (TIER) to enable IMIA interrupts. Transfers to the NDR can also be set using the DMAC.
5. Set the initial output value in the I/O port data register to be used by TPC.
6. Set the I/O port control register to be used by TPC as the TP pin function (11).

7. Set to 1 the bit that performs TPC output to the next data enable register (NDER).
8. Select the ITU compare match that will be the TPC output trigger using the TPC output control register (TPCR).
9. Set the next TPC output value in the NDR.
10. Set 1 in the STR bit of the timer start register (TSTR) and start the timer counter counting.
11. Set the next output value in the NDR whenever an IMIA interrupt is generated.

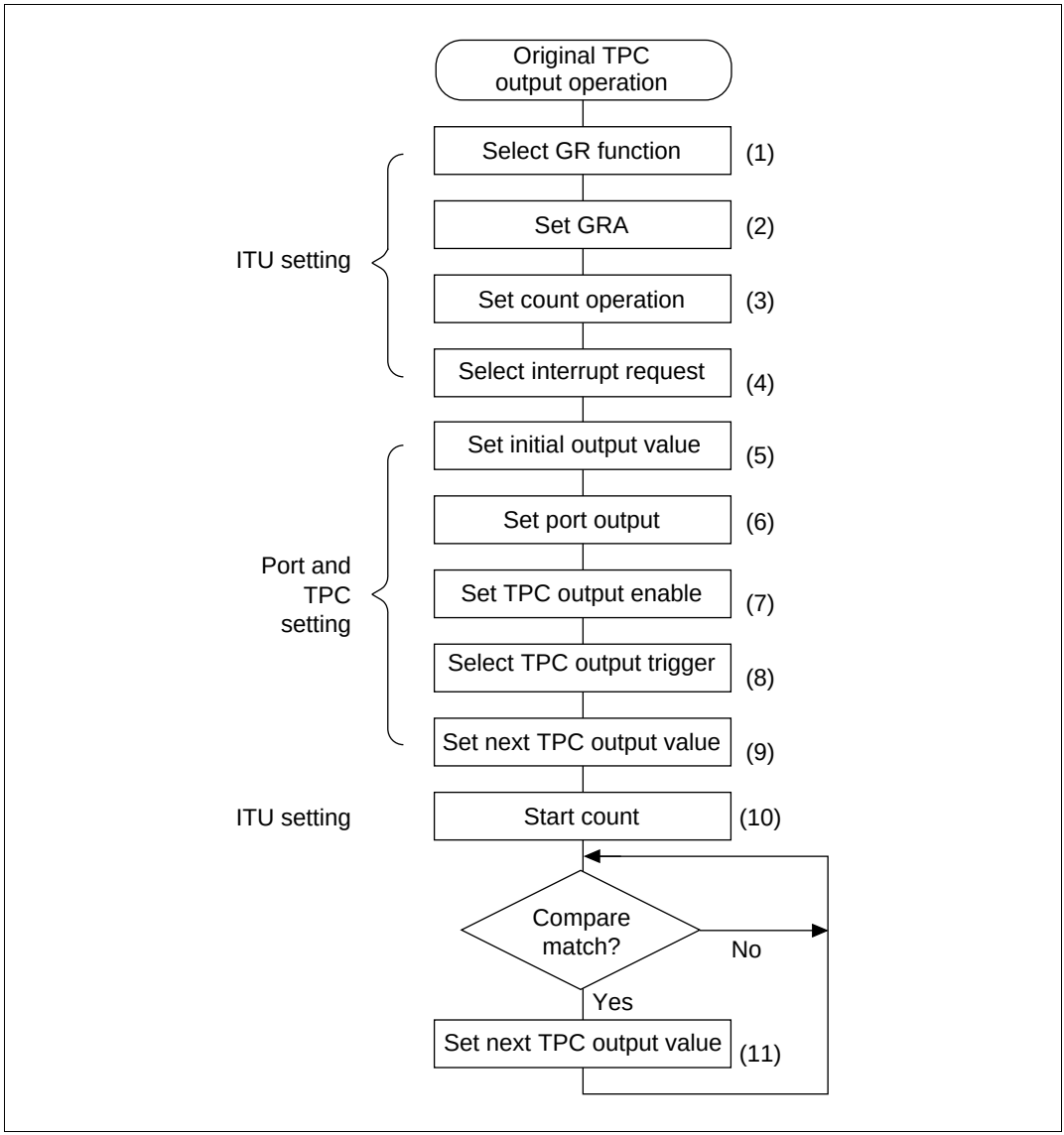



**Figure 11.4 Example of Setting Procedure for TPC Ordinary Output**

**Five-Phase Pulse Output (figure 11.5):**

1. Set the GRA of the ITU that serves as output trigger as the output compare register. Set the cycle time in the GRA of the ITU and select to clear the counter upon compare match A. Set the IMIEA bit of TIER to 1 to enable the compare match A interrupt.
2. Write H'FFC0 in the PBCR1, write H'F8 in the NDERB, and set G3CMS0, G3CMS1, G2CMS1 and G2CMS0 in the TPCR to set the ITU compare match selected in step 1 as the output trigger. Write output data H'80 in the NDRB.
3. When the selected ITU channel starts operating and a compare-match occurs, the values in the NDRB are transferred to the PBDR and output. The compare-match/input capture A (IMIA) interrupt service routine writes the next output data (H'C0) in the NDRB.
4. Five-phase pulse output can be obtained by writing H'40, H'60, H'20, H'30, H'10, H'18, H'08, H'88… at successive compare-match interrupts. If the DMA controller is set for activation by compare-match, pulse output can be obtained without loading the CPU.

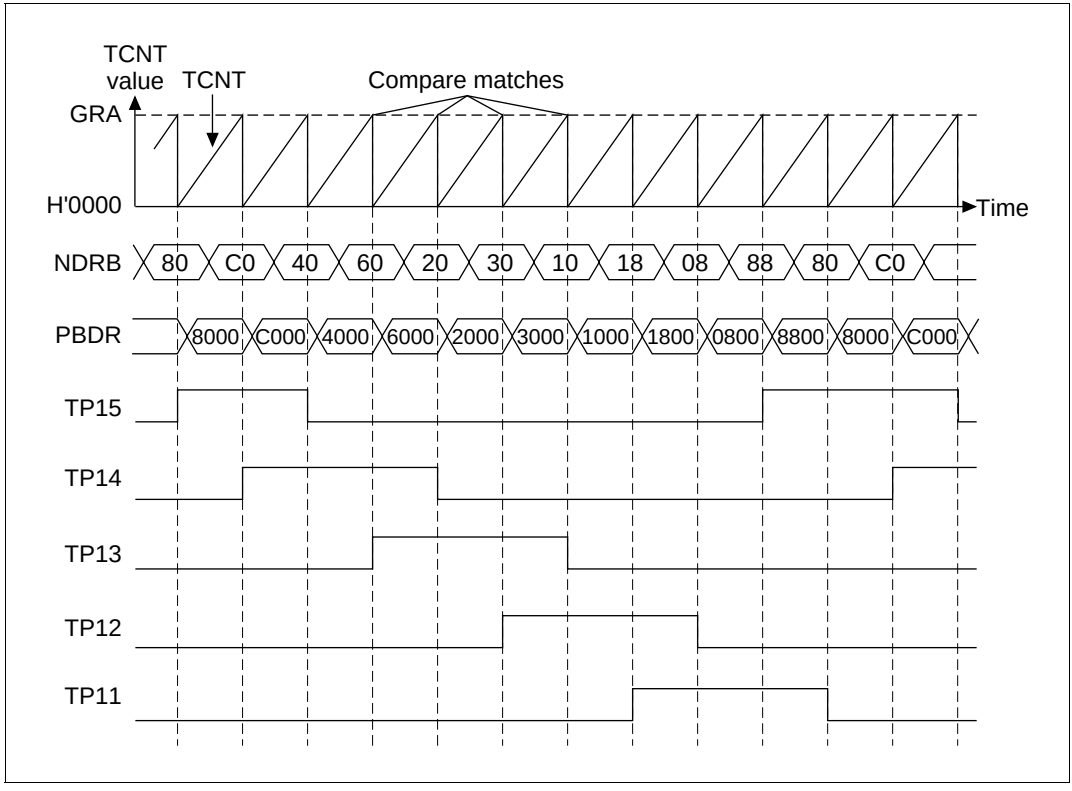



**Figure 11.5 TPC Output Example (5-Phase Pulse Output)**

### 11.3.4 TPC Output Non-Overlap Operation

**Setting Procedures for TPC Output Non-Overlap Operation (figure 11.6):**

1. Select GRA and GRB as output compare registers (output disable) with the timer I/O control register (TIOR).
2. Set the TPC output trigger cycle to GRB and the non-overlap cycle to GRA.
3. Select the counter clock with the TPSC2–TPSC0 bits of the timer control register (TCR). Select the counter clear sources with the CCLR1 and CCLR0 bits.
4. Set the timer interrupt enable register (TIER) to enable IMIA interrupts. Transfers to the NDR can also be set using the DMAC.
5. Set the initial output value in the I/O port data register to be used by TPC.
6. Set the I/O port control register to be used by TPC as the TP pin function (11).
7. Set to 1 the bit that performs TPC output to the next data enable register (NDER).
8. Select the ITU compare match that will be the TPC output trigger using the TPC output control register (TPCR).
9. Select the group that performs the non-overlap operation in the TPC output mode register (TPMR).
10. Set the next TPC output value in the NDR.
11. Set 1 in the STR bit of the timer start register (TSTR) and start the timer counter counting.
12. Set the next output value in the NDR whenever an IMIA interrupt is generated.

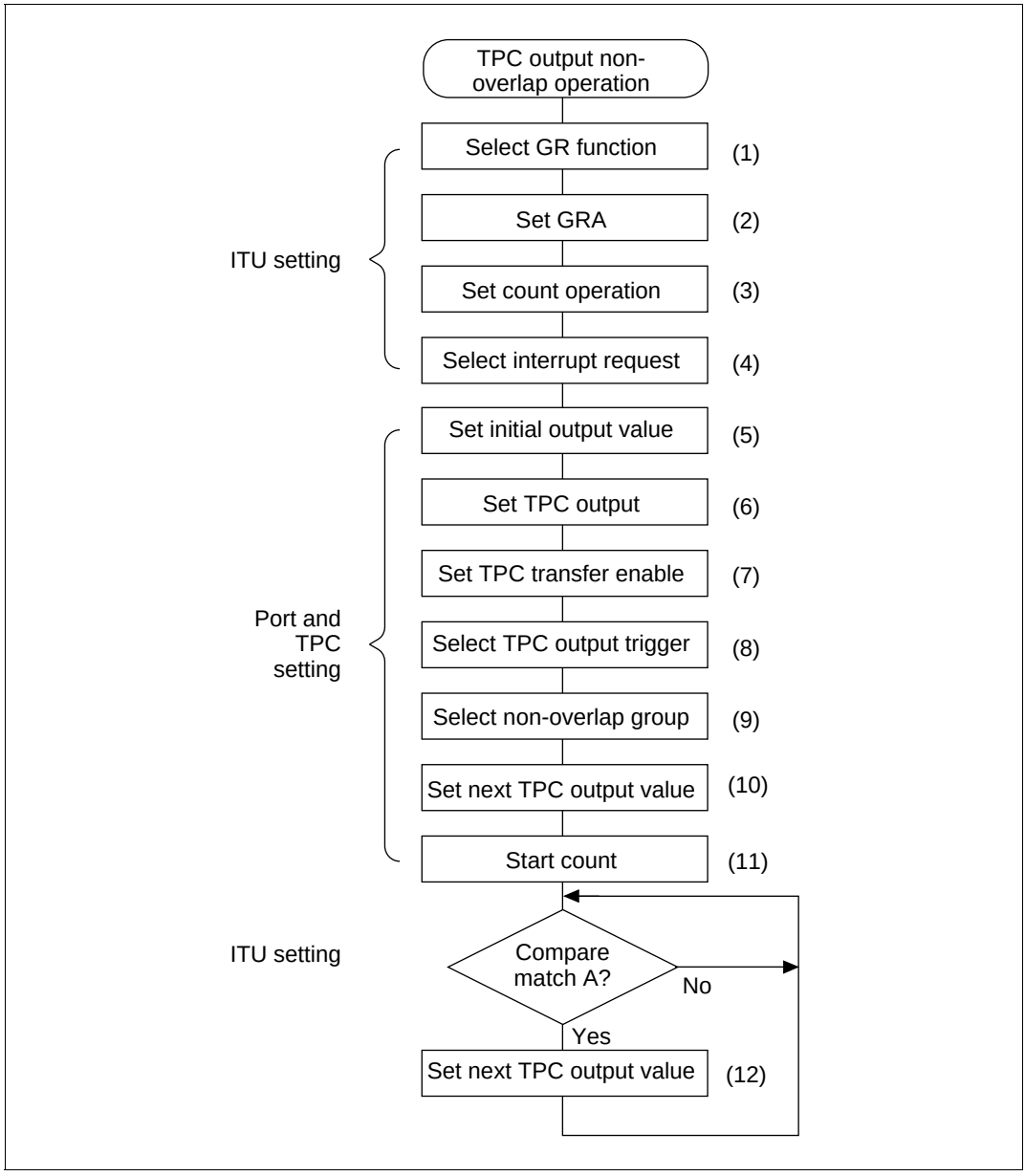



**Figure 11.6 Example of Setting Procedures for TPC Output Non-Overlap Operation**

**TPC Output Non-Overlap Operation (Four-Phase Complementary Non-Overlap Output) (figure 11.7):**

1. Set GRA and GRB of the ITU that serves as output trigger in the output compare registers. Set the cycle in the GRB and the non-overlap cycle time in the GRA and select to clear the counter upon compare match B. Set the IMIEA bit of TIER to 1 to enable the IMIA interrupt.
2. Write H'FFFF in the PBCR1, write H'FF in the NDERB, and set G3CMS1, G3CMS0, G2CMS1 and G2CMS0 in the TPCR to set the ITU compare match selected in step 1 as the output trigger. Set the G3NOV and G2NOV bits in the TPMR to 1 to set the non-overlap operation. Write output data H'95 in the NDRB.
3. When the selected ITU channel starts operating and a GRB compare-match occurs, 1 output changes to 0 output; when a GRA compare match occurs, 0 output changes to 1 output. (The change from 0 output to 1 output is delayed by the value set in GRA.) The IMIA interrupt service routine writes the next output data (H'65) in the NDRB.
4. Four-phase complementary non-overlap output can be obtained by writing H'59, H'56, H'95… at successive IMIA interrupts. If the DMA controller is set for activation by compare-match, pulse output can be obtained without loading the CPU.

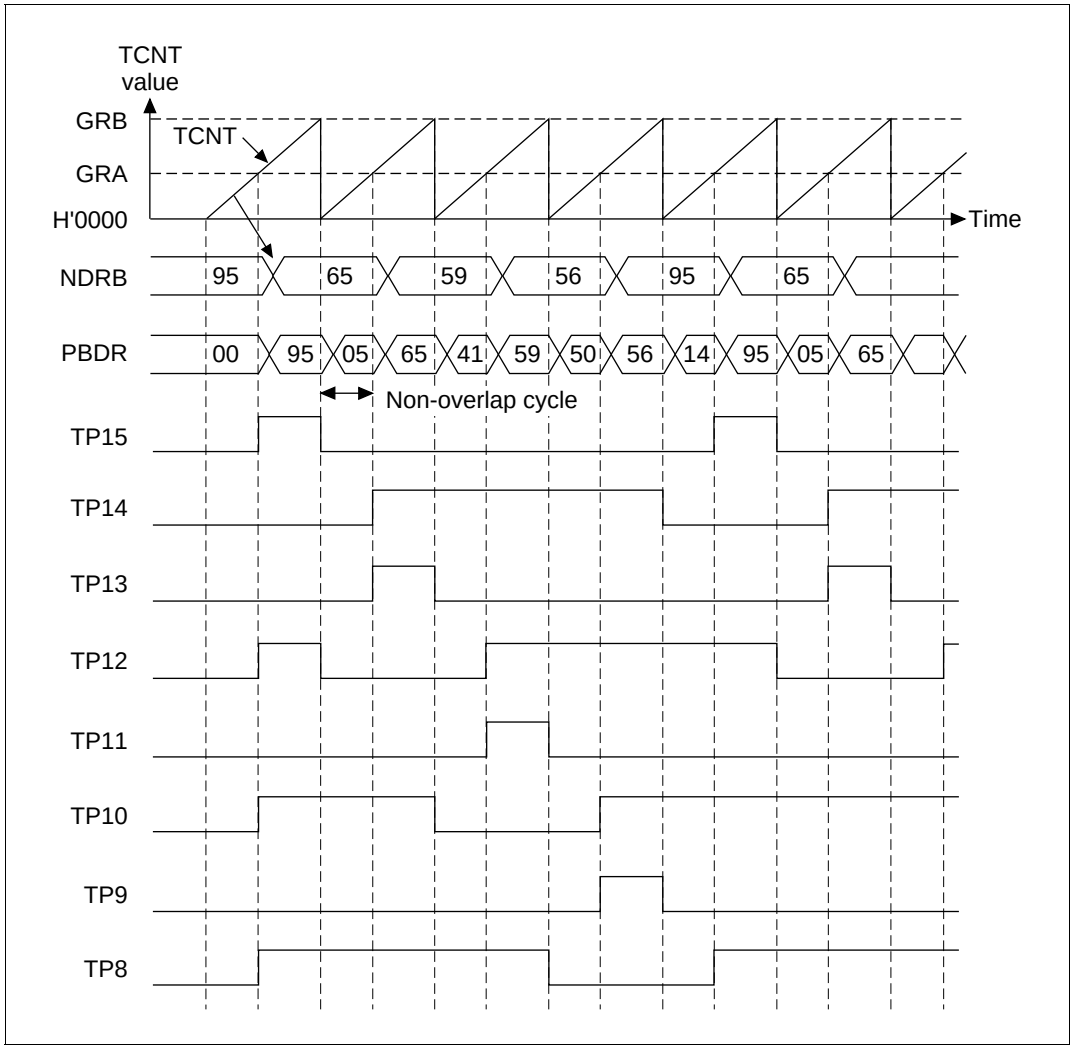



**Figure 11.7 Non-Overlap Output Example (Four-Phase Complementary Output)**

### 11.3.5 TPC Output by Input Capture

TPC can also be output by using input capture rather than ITU compare matches. The general register A (GRA) of the ITU selected by the TPCR functions as an input capture register and TPC output occurs upon an input capture signal. Figure 11.8 shows the timing.

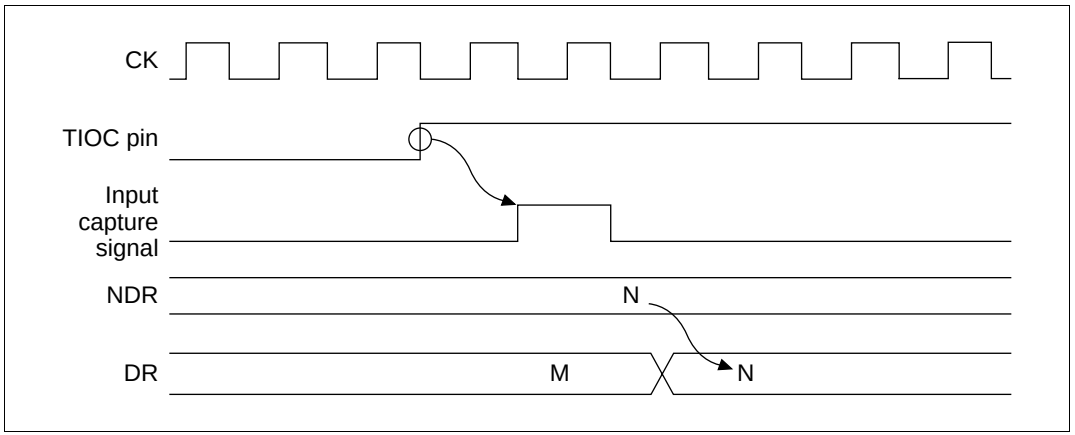



**Figure 11.8 TPC Output by Input Capture**

## 11.4 Usage Notes

### 11.4.1 Non-Overlap Operation

During non-overlap operation, transfers from the NDR to data registers (DR) occurs as follows.

1. NDR contents are always transferred to the DR on compare match A.
2. The contents of the bit transferred by the NDR are only transferred on compare match B when they are 0. No transfer occurs for a 1.

Figure 11.9 illustrates the TPC output operation during non-overlap.

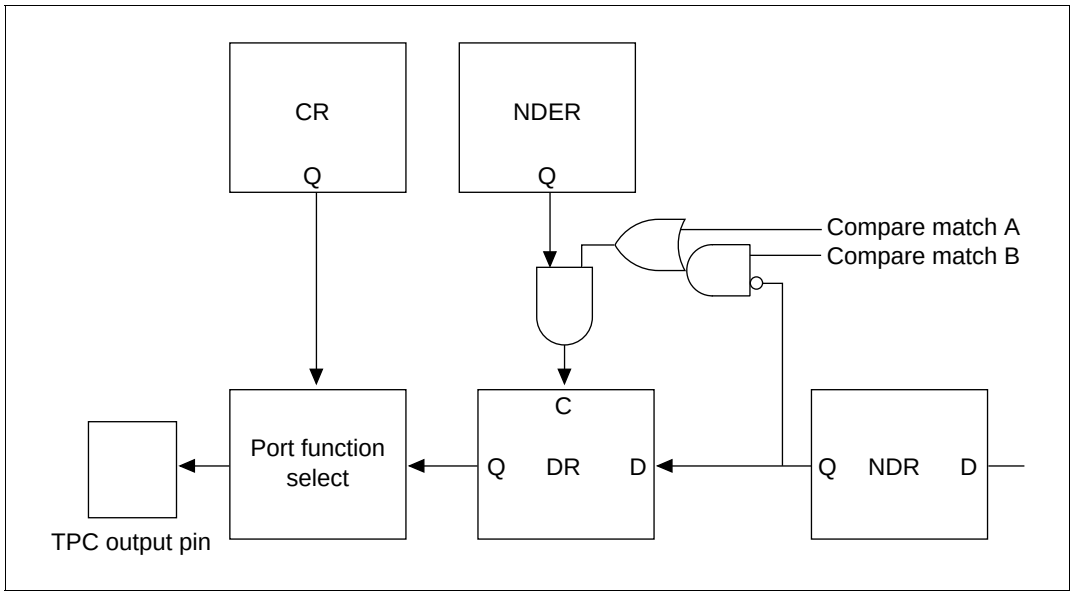



**Figure 11.9 TPC Output Non-Overlap Operation**

When a compare match B occurs before the compare match A, the 0 data transfer can be performed before the 1 data transfer, so a non-overlapping waveform can be output. In such cases, be sure not to change the NDR contents until the compare match A after the compare match B occurs (non-overlap period). This can be ensured by writing the next data to the NDR using the IMIA interrupt service routine. The DMAC can also be started up using an IMIA interrupt. However, these write operations should be performed prior to the next compare match B. The timing is shown in figure 11.10.

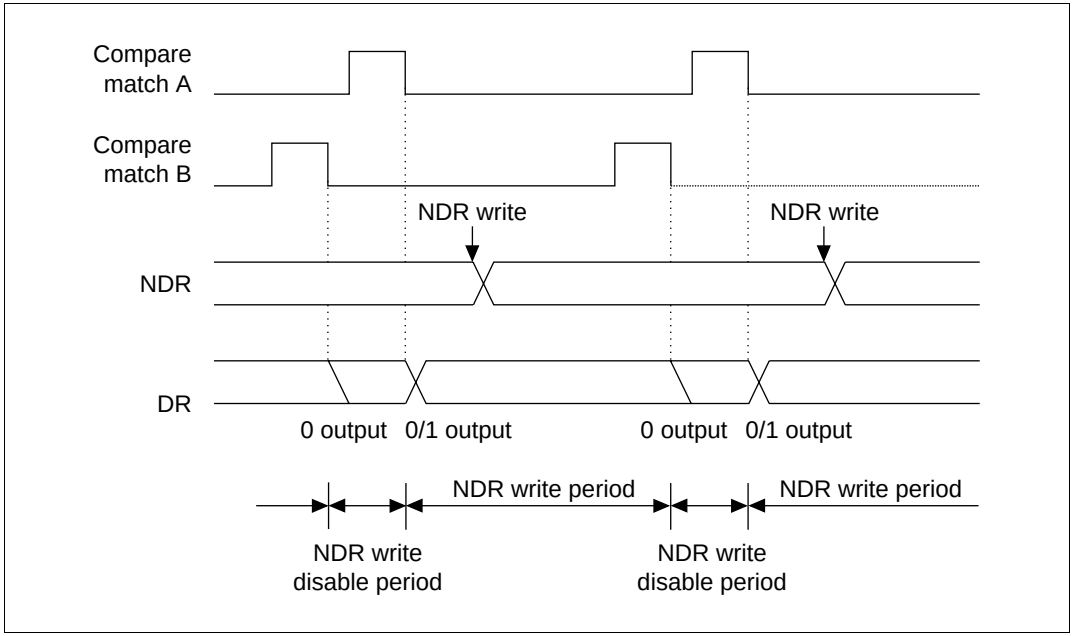



**Figure 11.10 Non-Overlap Operation and NDR Write Timing**

# Section 12 Watchdog Timer (WDT)

## 12.1 Overview

The SuperH microcomputer has a one-channel watchdog timer (WDT) for monitoring system operations. If a system becomes uncontrolled and the timer counter overflows without being rewritten correctly by the CPU, an overflow signal ($\overline{\text{WDTOVF}}$) is output externally . The WDT can simultaneously generate an internal reset signal for the entire chip.

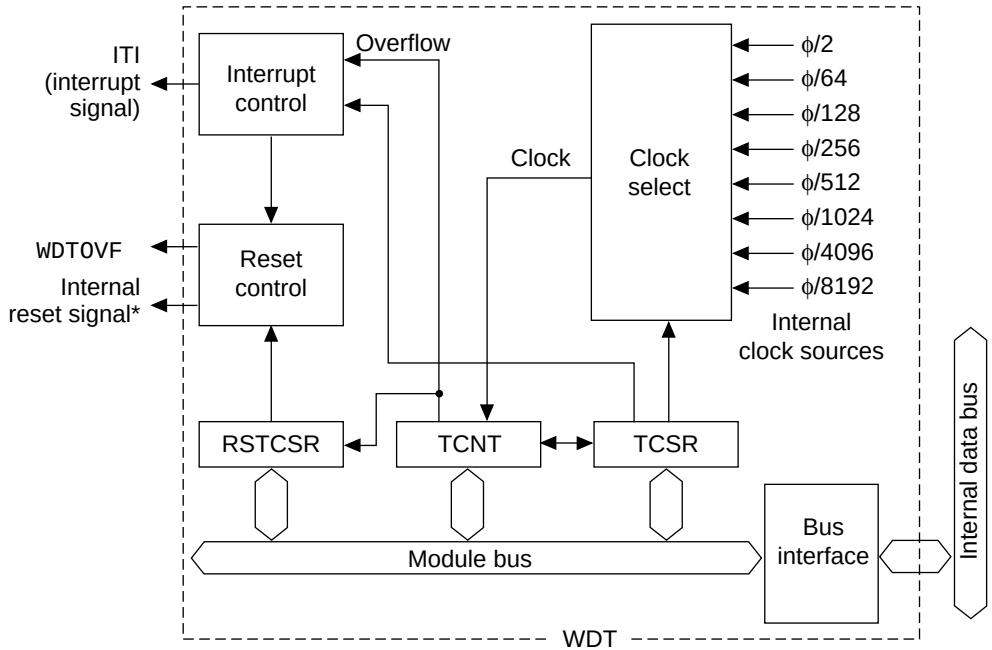
When this watchdog function is not needed, the WDT can be used as an interval timer. In the interval timer operation, an interval timer interrupt is generated at each counter overflow. The WDT is also used in recovering from the standby mode.

### 12.1.1 Features

- Watchdog timer mode and interval timer mode can be selected.
- Outputs $\overline{\text{WDTOVF}}$ in the watchdog timer mode. When the counter overflows in the watchdog timer mode, overflow signal $\overline{\text{WDTOVF}}$ is output externally. You can select whether or not to reset the chip internally when this happens. Either the power-on reset or manual reset signal can be selected as the internal reset signal.
- Generates interrupts in the interval timer mode. When the counter overflows, it generates an interval timer interrupt.
- Used to clear the standby mode.
- Selection of eight counter clock sources

### 12.1.2 Block Diagram

Figure 12.1 is the block diagram of the WDT.




TCSR: Timer control/status register

TCNT: Timer counter

RSTCSR: Reset control/status register

Note: The internal reset signal can be generated by setting the register. The type of reset can be selected (power-on or manual resets).

**Figure 12.1 WDT Block Diagram**

### 12.1.3 Pin Configuration

Table 12.1 shows the pin configuration.

**Table 12.1 Pin Configuration**

| Pin | Abbreviation | I/O | Function |
|---|---|---|---|
| Watchdog timer overflow | WDTOVF | O | Outputs the counter overflow signal in the watchdog mode |

### 12.1.4 Register Configuration

Table 12.2 summarizes the three WDT registers. They are used to select the clock, switch the WDT mode, and control the reset signal.

**Table 12.2 WDT Registers**

| | | | | Address | |
|---|---|---|---|---|---|
| **Name** | **Abbreviation** | **R/W** | **Initial Value** | **Write*1** | **Read*2** |
| Timer control/status register | TCSR | R/(W)*3 | H'18 | H'5FFFFB8 | H'5FFFFB8 |
| Timer counter | TCNT | R/W | H'00 | | H'5FFFFB9 |
| Reset control/status register | RSTCSR | R/(W)*3 | H'3F | H'5FFFFBA | H'5FFFFBB |

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

## 12.2 Register Descriptions

### 12.2.1 Timer Counter (TCNT)

The TCNT is an eight-bit readable and writable upcounter. The TCNT differs from other registers in that it is more difficult to write. See section 12.2.4, Register Access, for details. When the timer enable bit (TME) in the timer control/status register (TCSR) is set to 1, the timer counter starts counting pulses of an internal clock source selected by clock select bits 2–0 (CKS2–CKS0) in the TCSR. When the value of the TCNT overflows (changes from H'FF–H'00), a watchdog timer overflow signal ($\overline{\text{WDTOVF}}$) or interval timer interrupt (ITI) is generated, depending on the mode selected in the WT/$\overline{\text{IT}}$ bit of the TCSR. The TCNT is initialized to H'00 by a reset and when the TME bit is cleared to 0. It is not initialized in the standby mode.

| 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 |

### 12.2.2 Timer Control/Status Register (TCSR)

The timer control/status register (TCSR) is an eight-bit readable and writable register. The TCSR differs from other registers in being more difficult to write. See section 12.2.4, Register Access, for details. Its functions include selecting the timer mode and clock source. Bits 7–5 are initialized to 000 by a reset or in standby mode. Bits 2–0 are initialized to 000 by a reset, but retain their values in the standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | OVF | WT/IT | TME | — | — | CKS2 | CKS1 | CKS0 |
| Initial value: | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| R/W: | R/(W)* | R/W | R/W | — | — | R/W | R/W | R/W |

- Bit 7 (overflow flag (OVF)): OVF indicates that the TCNT has overflowed from H'FF–H'00. It is not set in the watchdog timer mode.

| Bit 7: OVF | Description |
|---|---|
| 0 | No overflow of TCNT in interval timer mode (initial value)<br>Cleared by reading OVF, then writing 0 in OVF |
| 1 | TCNT overflow in the interval timer mode |

- Bit 6 (timer mode select (WT/$\overline{IT}$)): WT/$\overline{IT}$ selects whether to use the WDT as a watchdog timer or interval timer. When the TCNT overflows, the WDT either generates an interval timer interrupt (ITI) or generates a $\overline{WDTOVF}$ signal, depending on the mode selected.

| Bit 6: WT/IT | Description |
|---|---|
| 0 | Interval timer mode: interval timer interrupt to the CPU when TCNT overflows (initial value) |
| 1 | Watchdog timer mode: $\overline{WDTOVF}$ signal output externally when TCNT overflows. Section 12.2.3, Reset Control/Status Register (RSTCSR), describes in detail what happens when TCNT overflows in the watchdog timer mode. |

- Bit 5 (timer enable (TME)): TME enables or disables the timer.

| Bit 5: TME | Description |
|---|---|
| 0 | Timer disabled: TCNT is initialized to H'00 and count-up stops (initial value) |
| 1 | Timer enabled: TCNT starts counting. A WDTOVF signal or interrupt is generated when TCNT overflows. |

- Bits 4 and 3 (reserved): These bits always read as 1. The write value should always be 1.

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

| | | | Description | |
|---|---|---|---|---|
| **Bit 2: CKS2** | **Bit 1: CKS1** | **Bit 0: CKS0** | **Clock Source** | **Overflow Interval* (ϕ = 20 MHz)** |
| 0 | 0 | 0 | ϕ/2 (initial value) | 25.6 µs |
| 0 | 0 | 1 | ϕ/64 | 819.2 µs |
| 0 | 1 | 0 | ϕ/128 | 1.6 ms |
| 0 | 1 | 1 | ϕ/256 | 3.3 ms |
| 1 | 0 | 0 | ϕ/512 | 6.6 ms |
| 1 | 0 | 1 | ϕ/1024 | 13.1 ms |
| 1 | 1 | 0 | ϕ/4096 | 52.4 ms |
| 1 | 1 | 1 | ϕ/8192 | 104.9 ms |

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

### 12.2.3 Reset Control/Status Register (RSTCSR)

The RSTCSR is an eight-bit readable and writable register that controls output of the reset signal generated by timer counter (TCNT) overflow and selects the internal reset signal type. The RSTCSR differs from other registers in that it is more difficult to write. See section 12.2.4 Register Access, for details. RSTCR is initialized to H'1F by input of a reset signal from the $\overline{\text{RES}}$ pin, but is not initialized by the internal reset signal generated by the overflow of the WDT. It is initialized to H'1F in standby mode.

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

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

- Bit 7 (watchdog timer overflow (WOVF)): WOVF indicates that the TCNT has overflowed (H'FF → H'00) in the watchdog timer mode. It is not set in the interval timer mode.

| Bit 7: WOVF | Description |
|---|---|
| 0 | No TCNT overflow in watchdog timer mode (initial value)<br>Cleared when software reads WOVF, then writes 0 in WOVF |
| 1 | Set by TCNT overflow in watchdog timer mode |

- Bit 6 (reset enable (RSTE)): RSTE selects whether to reset the chip internally if the TCNT overflows in the watchdog timer mode.

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

- Bit 5 (reset select (RSTS)): RSTS selects the type of internal reset generated if the TCNT overflows in the watchdog timer mode.

| Bit 5: RSTS | Description |
|---|---|
| 0 | Power-on reset initial value) |
| 1 | Manual reset |

- Bits 4–0 (reserved): These bits always read as 1. The write value should always be 1.

### 12.2.4 Register Access

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

**Writing to the TCNT and TCSR:** These registers must be written by a word transfer instruction. They cannot be written by byte transfer instructions. The TCNT and TCSR both have the same write address. The write data must be contained in the lower byte of the written word. The upper byte must be H'5A (for the TCNT) or H'A5 (for the TCSR) (figure 12.2). This transfers the write data from the lower byte to the TCNT or TCSR.

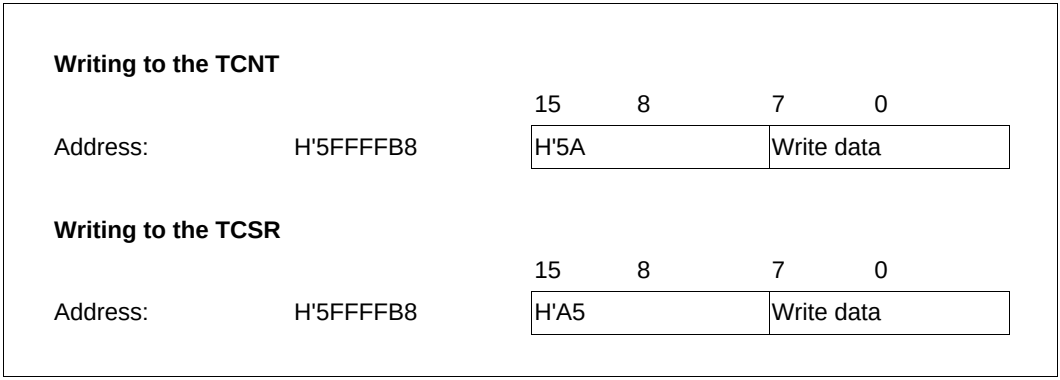



**Figure 12.2 Writing to the TCNT and TCSR**

**Writing to the RSTCSR:** The RSTCSR must be written by a word access to address H'5FFFFFBA. It cannot be written by byte transfer instructions. Procedures for writing 0 in WOVF (bit 7) and for writing to RSTE (bit 6) and RSTS (bit 5) are different, as shown in figure 12.3. To write 0 in the WOVF bit, the write data must be H'A5 in the upper byte and H'00 in the lower byte. This clears the WOVF bit to 0. The RSTE and RSTS bits are not affected. To write to the RSTE and RSTS bits, the upper byte must be H'5A and the lower byte must be the write data. The values of bits 6 and 5 of the lower byte are transferred to the RSTE and RSTS bits, respectively. The WOVF bit is not affected.

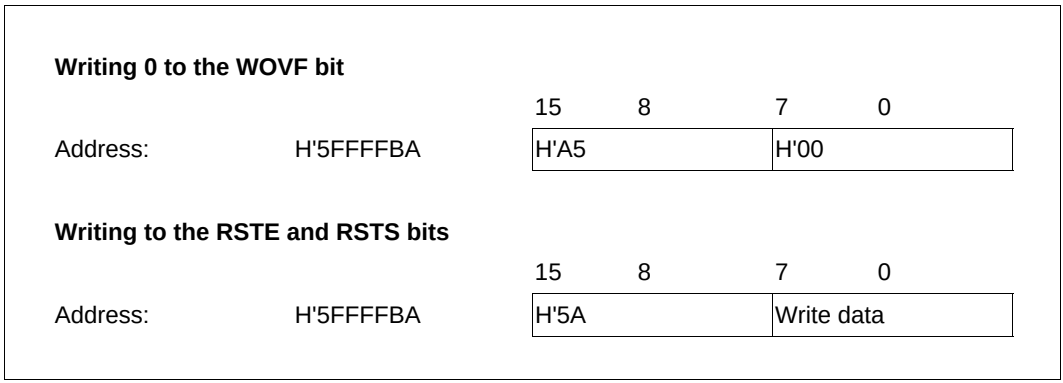



**Figure 12.3 Writing to the RSTCSR**

**Reading from the TCNT, TCSR, and RSTCSR:** TCNT, TCSR, and RSTCSR are read like other registers. Use byte transfer instructions. The read addresses are H'5FFFFB8 for the TCSR, H'5FFFFB9 for the TCNT, and H'5FFFFBB for the RSTCSR.

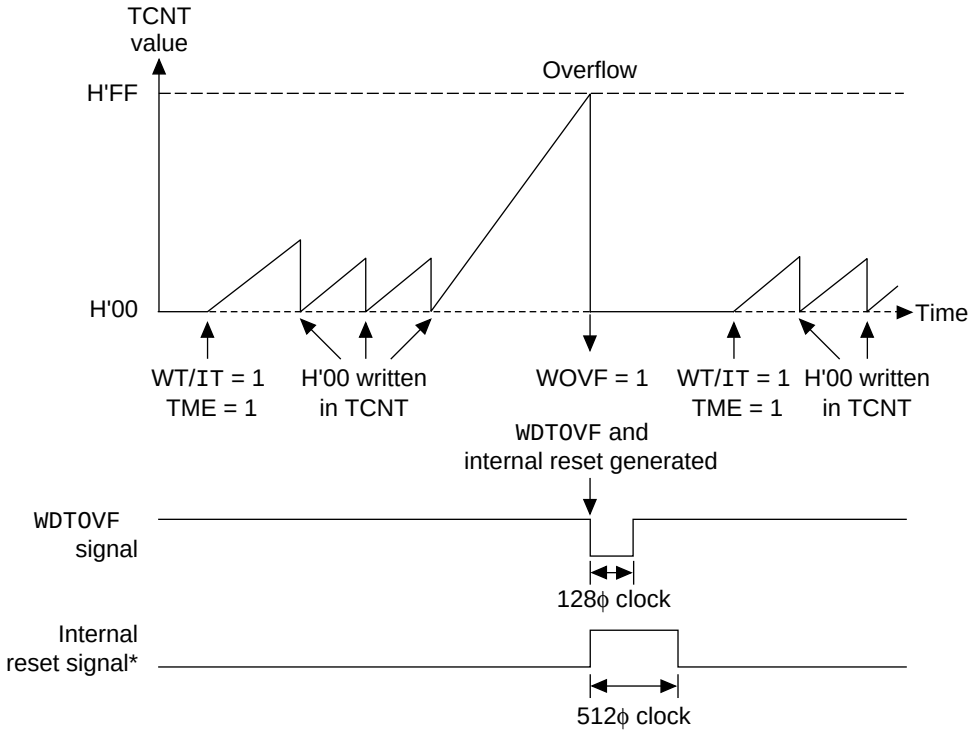
## 12.3 Operation

### 12.3.1 Operation in the Watchdog Timer Mode

To use the WDT as a watchdog timer, set the WT/$\overline{IT}$ and TME bits of the TCSR to 1. Software must prevent TCNT overflow by rewriting the TCNT value (normally by writing H'00) before overflow occurs. If the TCNT fails to be rewritten and overflows due to a system crash or the like, a $\overline{WDTOVF}$ signal is output (figure 12.4). The $\overline{WDTOVF}$ signal can be used to reset external system devices. The $\overline{WDTOVF}$ signal is output for 128$\phi$ clock cycles.

If the RSTE bit in the RSTCSR is set to 1, a signal to reset the chip will be generated internally simultaneous to the $\overline{WDTOVF}$ signal when TCNT overflows. Either a power-on reset or a manual reset can be selected by the RSTS bit. The internal reset signal is output for 512$\phi$ clock cycles.

When a watchdog reset is generated simultaneously with input at the $\overline{RES}$ pin, the software distinguishes the $\overline{RES}$ reset from the watchdog reset by checking the WOVF bit in the RSTCSR. The $\overline{RES}$ reset takes priority. The WOVF bit is cleared to 0.




WT/IT: Timer mode select bit

TME: Timer enable bit

Note: The internal reset signal is only generated when the RSTE bit is 1.

**Figure 12.4 Operation in the Watchdog Timer Mode**

### 12.3.2 Operation in the Interval Timer Mode

To use the WDT as an interval timer, clear WT/$\overline{IT}$ to 0 and set TME to 1. An interval timer interrupt (ITI) is generated each time the timer counter overflows. This function can be used to generate interval timer interrupts at regular intervals (figure 12.5).

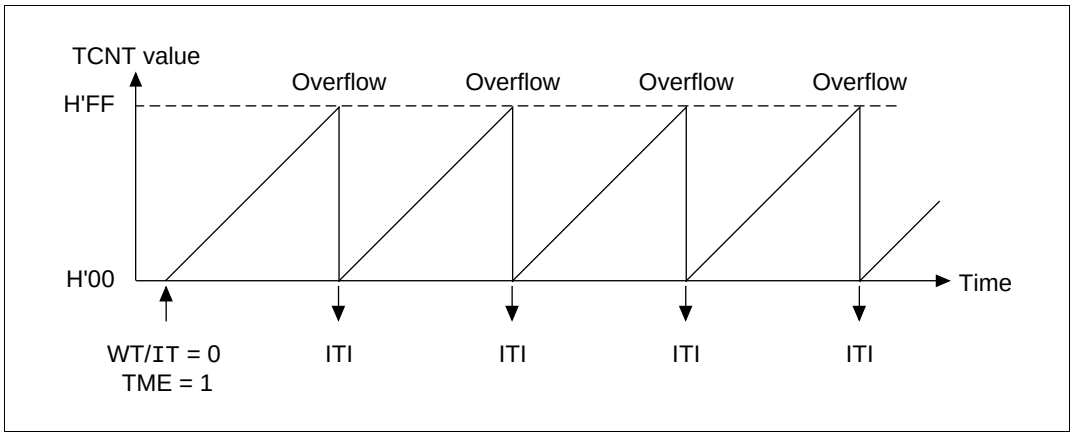



**Figure 12.5 Operation in the Interval Timer Mode**

### 12.3.3 Operation in the Standby Mode

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

**Transition to the Standby Mode:** The TME bit in the TCSR must be cleared to 0 to stop the watchdog timer counter before it enters the standby mode. The chip cannot enter the standby mode while the TME bit is set to 1. Set bits CKS2–CKS0 so that the counter overflow interval is equal to or longer than the oscillation settling time. See section 20.3, AC Characteristics, for the oscillation settling time.

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

For details on the standby mode, see section 19, Power Down States.

### 12.3.4 Timing of Setting the Overflow Flag (OVF)

In the interval timer mode, when the TCNT overflows the OVF flag is set to 1 and an interval timer interrupt is requested (figure 12.6).

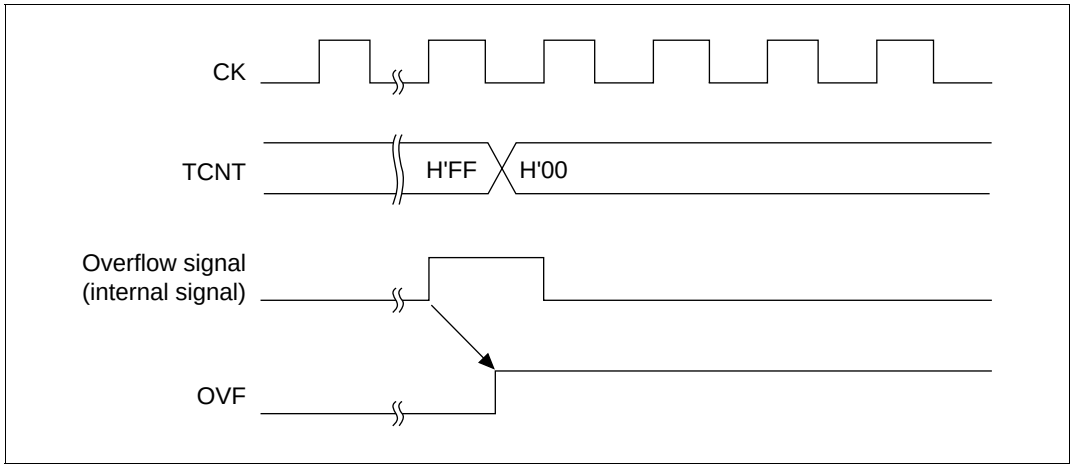



**Figure 12.6 Timing of Setting the OVF**

### 12.3.5 Timing of Setting the Watchdog Timer Overflow Flag (WOVF)

When the TCNT overflows the WOVF bit of the RSTCSR is set to 1 and a $\overline{\text{WDTOVF}}$ signal is output. When the RSTE bit is set to 1, TCNT overflow enables an internal reset signal to be generated for the entire chip (figure 12.7).

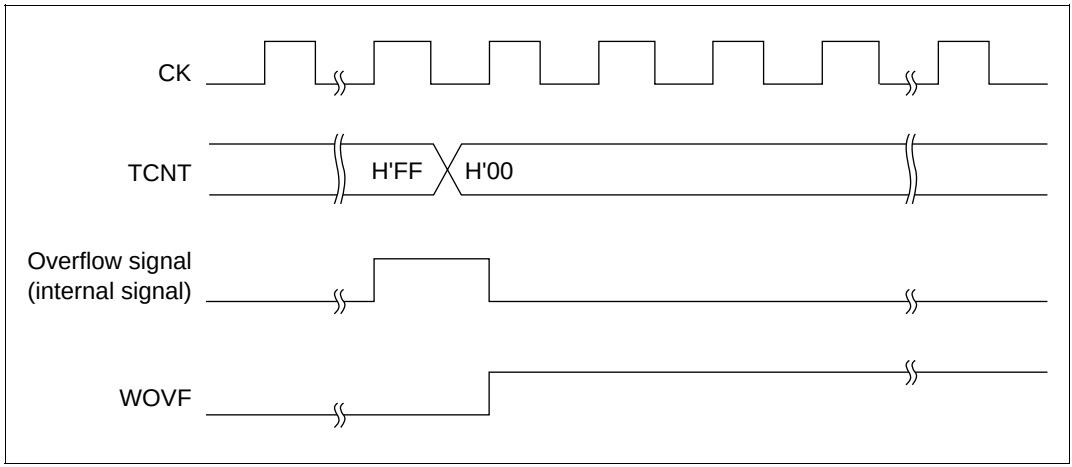



**Figure 12.7 Timing of Setting the WOVF Bit and Internal Reset**

## 12.4 Usage Notes

### 12.4.1 TCNT Write and Count Up Contention

If a timer counter clock pulse is generated during the T3 state of a write cycle to the TCNT, the write takes priority and the timer counter is not incremented (figure 12.8).

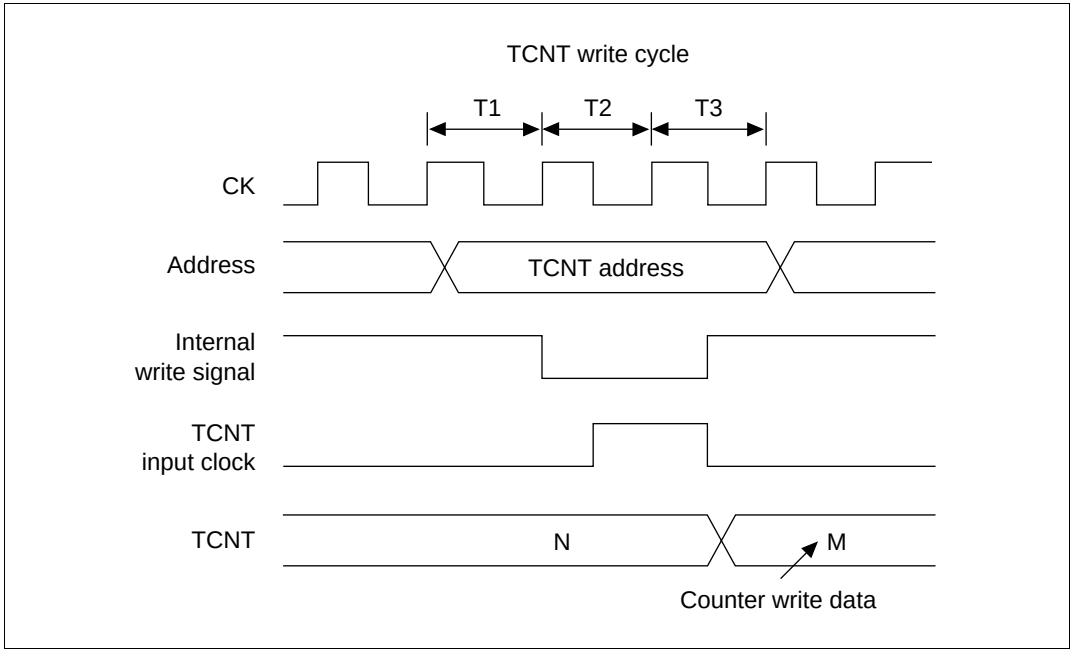



**Figure 12.8 Contention between TCNT Write and Increment**

### 12.4.2 Changing CKS2-CKS0 Bit Values

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

### 12.4.3 Changing Watchdog Timer/Interval Timer Modes

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

### 12.4.4 System Reset With $\overline{\text{WDTOVF}}$

If a $\overline{\text{WDTOVF}}$ signal is input to the $\overline{\text{RES}}$ pin, the LSI cannot initialize correctly. Avoid logical input of the $\overline{\text{WDTOVF}}$ output signal to the $\overline{\text{RES}}$ input pin. To reset the entire system with the $\overline{\text{WDTOVF}}$ signal, use the circuit shown in figure 12.9.

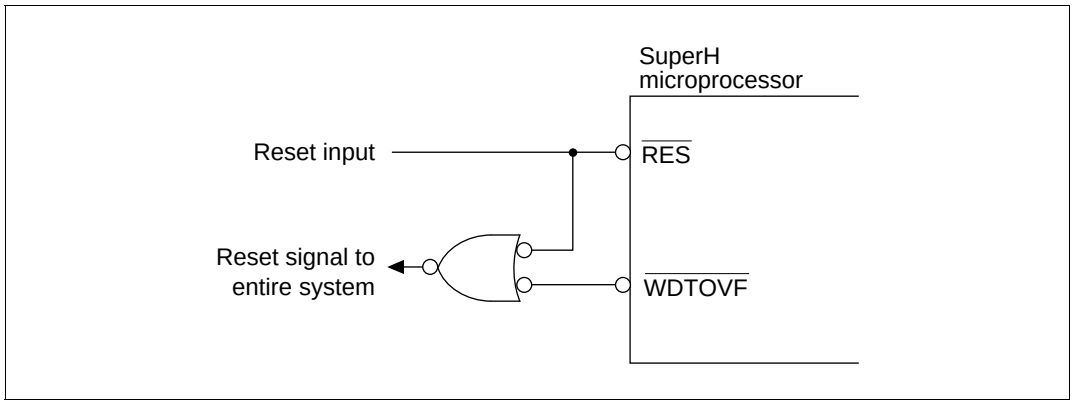



**Figure 12.9 Example of a System Reset Circuit with a $\overline{\text{WDTOVF}}$ Signal**

### 12.4.5 Internal Reset With the Watchdog Timer

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

# Section 13 Serial Communication Interface (SCI)

## 13.1 Overview

The SuperH microcomputer has a serial communication interface (SCI) with two independent channels. Both channels are functionally identical. The SCI supports both asynchronous and clocked synchronous serial communication. It also has a multiprocessor communication function for serial communication among two or more processors.

### 13.1.1 Features

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

### 13.1.2 Block Diagram

Figure 13.1 shows a block diagram of the SCI.

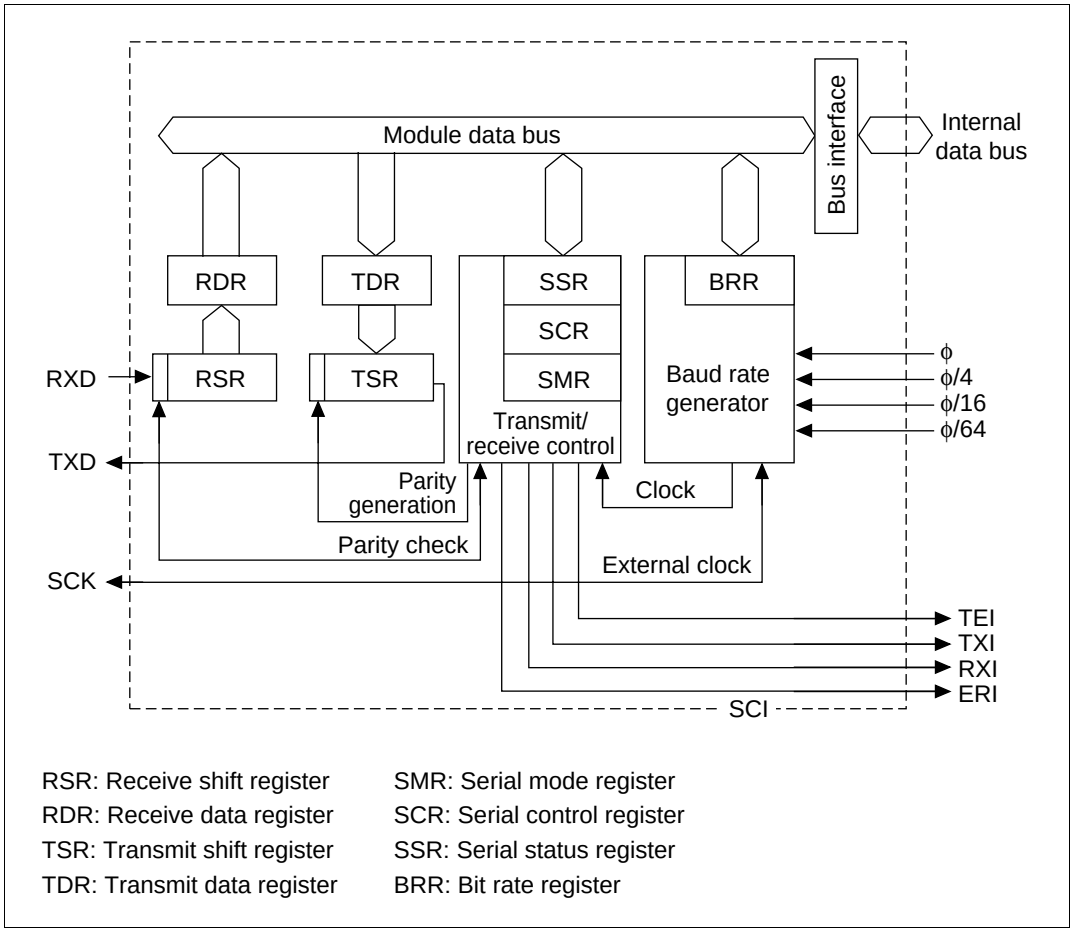



**Figure 13.1 SCI Block Diagram**

### 13.1.3 Input/Output Pins

Table 13.1 summarizes the SCI pins by channel.

**Table 13.1 SCI Pins**

| Channel | Pin Name | Abbreviation | Input/Output | Function |
|---|---|---|---|---|
| 0 | Serial clock pin | SCK0 | Input/output | SCI0 clock input/output |
| | Receive data pin | RxD0 | Input | SCI0 receive data input |
| | Transmit data pin | TxD0 | Output | SCI0 transmit data output |
| 1 | Serial clock pin | SCK1 | Input/output | SCI1 clock input/output |
| | Receive data pin | RxD1 | Input | SCI1 receive data input |
| | Transmit data pin | TxD1 | Output | SCI1 transmit data output |

### 13.1.4 Register Configuration

Table 13.2 summarizes the SCI internal registers. These registers select the communication mode (asynchronous or clocked synchronous), specify the data format and bit rate, and control the transmitter and receiver sections.

**Table 13.2 Registers**

| Channel | Address*1 | Name | Abbreviation | R/W | Initial Value | Access size |
|---|---|---|---|---|---|---|
| 0 | H'05FFFEC0 | Serial mode register | SMR0 | R/W | H'00 | 8, 16 |
| | H'05FFFEC1 | Bit rate register | BRR0 | R/W | H'FF | 8, 16 |
| | H'05FFFEC2 | Serial control register | SCR0 | R/W | H'00 | 8, 16 |
| | H'05FFFEC3 | Transmit data register | TDR0 | R/W | H'FF | 8, 16 |
| | H'05FFFEC4 | Serial status register | SSR0 | R/(W)*2 | H'84 | 8, 16 |
| | H'05FFFEC5 | Receive data register | RDR0 | R | H'00 | 8, 16 |
| 1 | H'05FFFEC8 | Serial mode register | SMR1 | R/W | H'00 | 8, 16 |
| | H'05FFFEC9 | Bit rate register | BRR1 | R/W | H'FF | 8, 16 |
| | H'05FFFECA | Serial control register | SCR1 | R/W | H'00 | 8, 16 |
| | H'05FFFECB | Transmit data register | TDR1 | R/W | H'FF | 8, 16 |
| | H'05FFFECC | Serial status register | SSR1 | R/(W)*2 | H'84 | 8, 16 |
| | H'05FFFECD | Receive data register | RDR1 | R | H'00 | 8, 16 |

Notes: 1. Only the values of bits A27–A24 and A8-A0 are valid; bits A23–A9 are ignored. For details on the register addresses, see section 8.3.5, Description of Areas.
2. Write 0 to clear flags.

## 13.2 Register Descriptions

### 13.2.1 Receive Shift Register

The receive shift register (RSR) receives serial data. Data input at the RxD pin are loaded into the RSR in the order received, LSB (bit 0) first. In this way the SCI converts received data to parallel form. When one byte has been received, it is automatically transferred to the receive data register (RDR). The CPU cannot read or write the RSR directly.

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

### 13.2.2 Receive Data Register

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

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

| 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 |

### 13.2.3 Transmit Shift Register

The transmit shift register (TSR) transmits serial data. The SCI loads transmit data from the transmit data register (TDR) into the TSR, then transmits the data serially from the TxD pin, LSB (bit 0) first. After transmitting one data byte, the SCI automatically loads the next transmit data from the TDR into the TSR and starts transmitting again. If the TDRE bit of the SSR is 1, however, the SCI does not load the TDR contents into the TSR. The CPU cannot read or write the TSR directly.

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

### 13.2.4 Transmit Data Register

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

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

| 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 |

### 13.2.5 Serial Mode Register

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

The CPU can always read and write the SMR. The SMR is initialized to H'00 by a reset or in standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | C/A | CHR | PE | O/E | STOP | MP | CKS1 | CKS0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bit 7 (communication mode (C/$\overline{A}$)): C/$\overline{A}$ selects whether the SCI operates in the asynchronous or clocked synchronous mode.

| Bit 7: C/A | Description |
|---|---|
| 0 | Synchronous mode (initial value) |
| 1 | Clocked synchronous mode |

- Bit 6 (character length (CHR)): CHR selects seven-bit or eight-bit data in the asynchronous mode. In the clocked synchronous mode, the data length is always eight bits, regardless of the CHR setting.

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

- Bit 5 (parity enable (PE)): PE selects whether to add a parity bit to transmit data and check the parity of receive data, in the asynchronous mode. In the clocked synchronous mode, a parity bit is neither added nor checked, regardless of the PE setting.

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

- Bit 4 (parity mode (O/$\overline{E}$): O/$\overline{E}$ selects even or odd parity when parity bits are added and checked. The O/$\overline{E}$ setting is used only in asynchronous mode and only when the parity enable bit (PE) is set to 1 to enable parity addition and check. The O/$\overline{E}$ setting is ignored in the clocked synchronous mode, or in the asynchronous mode when parity addition and check is disabled.

| Bit 4: O/E | Description |
|---|---|
| 0 | Even parity. If even parity is selected, the parity bit added to transmit data makes an even number of 1s in the transmitted character and parity bit combined. Receive data must have an even number of 1s in the received character and parity bit combined (initial value). |
| 1 | Odd parity. If odd parity is selected, the parity bit added to transmit data makes an odd number of 1s in the transmitted character and parity bit combined. Receive data must have an odd number of 1s in the received character and parity bit combined. |

- Bit 3 (stop bit length (STOP)): STOP selects one or two bits as the stop bit length in the asynchronous mode. This setting is used only in the asynchronous mode. It is ignored in the clocked synchronous mode because no stop bits are added.

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

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

- Bit 2 (multiprocessor mode (MP)): MP selects multiprocessor format. When multiprocessor format is selected, settings of the parity enable (PE) and parity mode (O/$\overline{E}$) bits are ignored. The MP bit setting is used only in the asynchronous mode; it is ignored in the clocked synchronous mode. For the multiprocessor communication function, see section 13.3.3, Multiprocessor Communication.

| Bit 2: MP | Description |
|---|---|
| 0 | Multiprocessor function disabled (initial value) |
| 1 | Multiprocessor format selected |

- Bits 1 and 0 (clock select 1 and 0 (CKS1 and CKS0)): CKS1 and CKS0 select the internal clock source of the on-chip baud rate generator. Four clock sources are available: $\phi$, $\phi$/4, $\phi$/16, and $\phi$/64. For further information on the clock source, bit rate register settings, and baud rate, see section 13.2.8, Bit Rate Register.

| Bit 1: CKS1 | Bit 0: CKS0 | Description |
|---|---|---|
| 0 | 0 | System clock ($\phi$) (initial value) |
| | 1 | $\phi$/4 |
| 1 | 0 | $\phi$/16 |
| | 1 | $\phi$/64 |

### 13.2.6 Serial Control Register

The serial control register (SCR) enables the SCI transmitter/receiver, selects serial clock output in the asynchronous mode, enables and disables interrupts, and selects the transmit/receive clock source. The CPU can always read and write the SCR. The SCR is initialized to H'00 by a reset or in standby mode.

| 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 7 (transmit interrupt enable (TIE)): TIE enables or disables the transmit-data-empty interrupt (TXI) requested when the transmit data register empty bit (TDRE) in the serial status register (SSR) is set to 1 due to transfer of serial transmit data from the TDR to the TSR.

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

- Bit 6 (receive interrupt enable (RIE)): RIE enables or disables the receive-data-full interrupt (RXI) requested when the receive data register full bit (RDRF) in the serial status register (SSR) is set to 1 due to transfer of serial receive data from the RSR to the RDR. Also enables or disables receive-error interrupt (ERI) requests.

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

Bit 5 (transmit enable (TE)): TE enables or disables the SCI transmitter.

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

- Bit 4 (receive enable (RE)): RE enables or disables the SCI receiver.

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

- Bit 3 (multiprocessor interrupt enable (MPIE)): MPIE enables or disables multiprocessor interrupts. The MPIE setting is used only in the asynchronous mode, and only if the multiprocessor mode bit (MP) in the serial mode register (SMR) is set to 1 during reception. The MPIE setting is ignored in the clocked synchronous mode or when the MP bit is cleared to 0.

| Bit 3: MPIE | Description |
|---|---|
| 0 | Multiprocessor interrupts are disabled (normal receive operation) (initial value)<br>MPE is cleared to 0 when:<br>1. MPIE is cleared to 0, or<br>2. Multiprocessor bit (MPB) is set to 1 in receive data. |
| 1 | Multiprocessor interrupts are enabled: Receive-data-full interrupt requests (RXI), receive-error interrupt requests (ERI), and setting of the RDRF, FER, and ORER status flags in the serial status register (SSR) are disabled until the multiprocessor bit is set to 1.<br>The SCI does not transfer receive data from the RSR to the RDR, does not detect receive errors, and does not set the RDRF, FER, and ORER flags in the serial status register (SSR). When it receives data that includes MPB = 1, MPB is set to 1, and the SCI automatically clears MPIE to 0, generates RXI and ERI interrupts (if the TIE and RIE bits in the SCR are set to 1), and allows the FER and ORER to be set. |

- Bit 2 (transmit-end interrupt enable (TEIE)): TEIE enables or disables the transmit-end interrupt (TEI) requested if TDR does not contain new transmit data when the MSB is transmitted.

| Bit 2: TEIE | Description |
|---|---|
| 0 | Transmit-end interrupt (TEI) requests are disabled* (initial value)<br>The TEI request can be cleared by reading the TDRE bit in the serial status register (SSR) after it has been set to 1, then clearing TDRE to 0; by clearing the transmit end (TEND) bit to 0; or by clearing the TEIE bit to 0. |
| 1 | Transmit-end interrupt (TEI) requests are enabled. |

- Bits 1 and 0 (clock enable 1 and 0 (CKE1 and CKE0)): CKE1 and CKE0 select the SCI clock source and enable or disable clock output from the SCK pin. Depending on the combination of CKE1 and CKE0, the SCK pin can be used for general-purpose input/output, serial clock output, or serial clock input.

  The CKE0 setting is valid only in the asynchronous mode, and only when the SCI is internally clocked (CKE1 = 0). The CKE0 setting is ignored in the clocked synchronous mode, or when an external clock source is selected (CKE1 = 1). Select the SCI operating mode in the serial mode register (SMR) before setting CKE1 and CKE0. For further details on selection of the SCI clock source, see table 13.9 in section 13.3, Operation.

| Bit 1: CKE1 | Bit 0: CKE0 | Description*1 | |
|---|---|---|---|
| 0 | 0 | Synchronous mode | Internal clock, SCK pin used for input pin (input signal is ignored or output pin output level is undefined) |
| | | Clocked synchronous mode | Internal clock, SCK pin used for serial clock output*2 |
| 0 | 1 | Synchronous mode | Internal clock, SCK pin used for clock output*3 |
| | | Clocked synchronous mode | Internal clock, SCK pin used for serial clock output |
| 1 | 0 | Synchronous mode | External clock, SCK pin used for clock input*4 |
| | | Clocked synchronous mode | External clock, SCK pin used for serial clock input |
| 1 | 1 | Synchronous mode | External clock, SCK pin used for clock input*4 |
| | | Clocked synchronous mode | External clock, SCK pin used for serial clock input |

Notes:
1. The SCK pin is multiplexed with other functions. Set the pin function controller (PFC) to select the SCK function and the SCK input/output of the SCK pin.
2. Initial value
3. The output clock frequency is the same as the bit rate.
4. The input clock frequency is 16 times the bit rate.

### 13.2.7 Serial Status Register

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

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

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

Note: Write 0 to clear flag.

- Bit 7 (transmit data register empty (TDRE)): TDRE indicates that the SCI has loaded transmit data from the TDR into the TSR and serial transmit new data can be written in the TDR.

| Bit 7: TDRE | Description |
|---|---|
| 0 | TDR contains valid transmit data<br>TDRE is cleared to 0 when:<br>• Software reads TDRE after it has been set to 1, then writes 0 in TDRE<br>• The DMAC writes data in TDR |
| 1 | TDR does not contain valid transmit data (initial value)<br>TDRE is set to 1 when:<br>• The chip is reset or enters standby mode<br>• The TE bit in the serial control register (SCR) is cleared to 0<br>• TDR contents are loaded into TSR, so new data can be written in TDR |

- Bit 6 (receive data register full (RDRF)): RDRF indicates that RDR contains received data.

| Bit 6: RDRF | Description |
|---|---|
| 0 | RDR does not contain valid received data (initial value)<br>RDRF is cleared to 0 when:<br>• The chip is reset or enters standby mode<br>• Software reads RDRF after it has been set to 1, then writes 0 in RDRF<br>• The DMAC reads data from RDR |
| 1 | RDR contains valid received data.<br>RDRF is set to 1 when serial data is received normally and transferred from RSR to RDR. |

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

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

| Bit 5: ORER | Description |
|---|---|
| 0 | Receiving is in progress or has ended normally (initial value)*1<br>ORER is cleared to 0 when:<br>• The chip is reset or enters standby mode<br>• Software reads ORER after it has been set to 1, then writes 0 in ORER |
| 1 | A receive overrun error occurred*2<br>ORER is set to 1 if reception of the next serial data ends when RDRF is set to 1 |

Notes: 1. Clearing the RE bit to 0 in the serial control register does not affect the ORER bit, which retains its previous value.
2. RDR continues to hold the data received before the overrun error, so subsequent receive data is lost. Serial receiving cannot continue while ORER is set to 1. In the clocked synchronous mode, serial transmitting is disabled.

- Bit 4 (framing error (FER)): FER indicates that data reception ended abnormally due to a framing error in the asynchronous mode.

| Bit 4: FER | Description |
|---|---|
| 0 | Receiving is in progress or has ended normally. Clearing the RE bit to 0 in the serial control register does not affect the FER bit, which retains its previous value (initial value).<br>FER is cleared to 0 when:<br>• The chip is reset or enters standby mode<br>• Software reads FER after it has been set to 1, then writes 0 in FER |
| 1 | A receive framing error occurred. When the stop bit length is two bits, only the first bit is checked. The second stop bit is not checked. When a framing error occurs, the SCI transfers the receive data into the RDR but does not set RDRF. Serial receiving cannot continue while FER is set to 1. In the clocked synchronous mode, serial transmitting is also disabled.<br>FER is set to 1 if the stop bit at the end of receive data is checked and found to be 0. |

- Bit 3 (parity error (PER)): PER indicates that data reception (with parity) ended abnormally due to a parity error in the asynchronous mode.

| Bit 3: PER | Description |
|---|---|
| 0 | Receiving is in progress or has ended normally. Clearing the RE bit to 0 in the serial control register does not affect the PER bit, which retains its previous value (initial value).<br>PER is cleared to 0 when:<br>• The chip is reset or enters standby mode<br>• Software reads PER after it has been set to 1, then writes 0 in PER |
| 1 | A receive parity error occurred. When a parity error occurs, the SCI transfers the receive data into the RDR but does not set RDRF. Serial receiving cannot continue while PER is set to 1. In the clocked synchronous mode, serial transmitting is also disabled.<br>PER is set to 1 if the number of 1s in receive data, including the parity bit, does not match the even or odd parity setting of the parity mode bit (O/E) in the serial mode register (SMR). |

- Bit 2 (transmit end (TEND)): TEND indicates that when the last bit of a serial character was transmitted, the TDR did not contain new transmit data, so transmission has ended. TEND is a read-only bit and cannot be written.

| Bit 2: TEND | Description |
|---|---|
| 0 | Transmission is in progress<br>TEND is cleared to 0 when:<br>• Software reads TDRE after it has been set to 1, then writing 0 in TDRE<br>• The DMAC writes data in TDR |
| 1 | End of transmission (initial value)<br>TEND is set to 1 when:<br>• The chip is reset or enters standby mode<br>• TE is cleared to 0 in the serial control register (SCR)<br>• TDRE is 1 when the last bit of a one-byte serial character is transmitted |

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

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

- Bit 0 (multiprocessor bit transfer (MPBT)): MPBT stores the value of the multiprocessor bit added to transmit data when a multiprocessor format is selected for transmitting in the asynchronous mode. The MPBT setting is ignored in the clocked synchronous mode, when a multiprocessor format is not selected, or when the SCI is not transmitting.

| Bit 0: MPBT | Description |
|---|---|
| 0 | Multiprocessor bit value in transmit data is 0 (initial value) |
| 1 | Multiprocessor bit value in transmit data is 1 |

### 13.2.8 Bit Rate Register (BRR)

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

The CPU can always read and write the BRR. The BRR is initialized to H'FF by a reset or in standby mode. SCI1 and SCI2 have independent baud rate generator control, so different values can be set in the two channels.

| 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 |

Table 13.3 shows examples of BRR settings in the asynchronous mode; table 13.4 shows examples of BBR settings in the clocked synchronous mode.

**Table 13.3 Bit Rates and BRR Settings in Asynchronous Mode**

| | **ϕ (MHz)** | | | | | |
|---|---|---|---|---|---|---|
| | **2** | | | **2.097152** | | |
| **Bit Rate (bits/s)** | **n** | **N** | **Error (%)** | **n** | **N** | **Error (%)** |
| 110 | 1 | 141 | 0.03 | 1 | 148 | –0.04 |
| 150 | 1 | 103 | 0.16 | 1 | 108 | 0.21 |
| 300 | 0 | 207 | 0.16 | 0 | 217 | 0.21 |
| 600 | 0 | 103 | 0.16 | 0 | 108 | 0.21 |
| 1200 | 0 | 51 | 0.16 | 0 | 54 | –0.70 |
| 2400 | 0 | 25 | 0.16 | 0 | 26 | 1.14 |
| 4800 | 0 | 12 | 0.16 | 0 | 13 | –2.48 |
| 9600 | — | — | — | 0 | 6 | –2.48 |
| 19200 | — | — | — | — | — | — |
| 31250 | 0 | 1 | 0.00 | — | — | — |
| 38400 | — | — | — | — | — | — |

**Table 13.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)**

| | φ (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 2.4576 | | | 3 | | | 3.6864 | | |
| Bit Rate(bits/s) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 1 | 174 | –0.26 | 1 | 212 | 0.03 | 2 | 64 | 0.70 |
| 150 | 1 | 127 | 0.00 | 1 | 155 | 0.16 | 1 | 191 | 0.00 |
| 300 | 0 | 255 | 0.00 | 1 | 77 | 0.16 | 1 | 95 | 0.00 |
| 600 | 0 | 127 | 0.00 | 0 | 155 | 0.16 | 0 | 191 | 0.00 |
| 1200 | 0 | 63 | 0.00 | 0 | 77 | 0.16 | 0 | 95 | 0.00 |
| 2400 | 0 | 31 | 0.00 | 0 | 38 | 0.16 | 0 | 47 | 0.00 |
| 4800 | 0 | 15 | 0.00 | 0 | 19 | –2.34 | 0 | 23 | 0.00 |
| 9600 | 0 | 7 | 0.00 | 0 | 9 | –2.34 | 0 | 11 | 0.00 |
| 19200 | 0 | 3 | 0.00 | 0 | 4 | –2.34 | 0 | 5 | 0.00 |
| 31250 | — | — | — | 0 | 2 | 0.00 | — | — | — |
| 38400 | 0 | 1 | 0.00 | — | — | — | 0 | 2 | 0.00 |

**Table 13.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)**

| | φ (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 4 | | | 4.9152 | | | 5 | | |
| Bit Rate(bits/s) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 70 | 0.03 | 2 | 86 | 0.31 | 2 | 88 | –0.25 |
| 150 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 300 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 600 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 1200 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 2400 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 4800 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | –1.36 |
| 9600 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |
| 19200 | — | — | — | 0 | 7 | 0.00 | 0 | 7 | 1.73 |
| 31250 | 0 | 3 | 0.00 | 0 | 4 | –1.70 | 0 | 4 | 0.00 |
| 38400 | — | — | — | 0 | 3 | 0.00 | 0 | 3 | 1.73 |

**Table 13.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)**

| | φ (MHz) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 6 | | | 6.144 | | | 7.3728 | | |
| Bit Rate(bits/s) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 106 | –0.44 | 2 | 108 | 0.08 | 2 | 130 | –0.07 |
| 150 | 2 | 77 | 0.16 | 2 | 79 | 0.00 | 2 | 95 | 0.00 |
| 300 | 1 | 155 | 0.16 | 1 | 159 | 0.00 | 1 | 191 | 0.00 |
| 600 | 1 | 77 | 0.16 | 1 | 79 | 0.00 | 1 | 95 | 0.00 |
| 1200 | 0 | 155 | 0.16 | 0 | 159 | 0.00 | 0 | 191 | 0.00 |
| 2400 | 0 | 77 | 0.16 | 0 | 79 | 0.00 | 0 | 95 | 0.00 |
| 4800 | 0 | 38 | 0.16 | 0 | 39 | 0.00 | 0 | 47 | 0.00 |
| 9600 | 0 | 19 | –2.34 | 0 | 19 | 0.00 | 0 | 23 | 0.00 |
| 19200 | 0 | 9 | –2.34 | 0 | 9 | 0.00 | 0 | 11 | 0.00 |
| 31250 | 0 | 5 | 0.00 | 0 | 5 | 2.40 | — | — | — |
| 38400 | 0 | 4 | –2.34 | 0 | 4 | 0.00 | 0 | 5 | 0.00 |

**Table 13.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)**

| | φ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | | | 9.8304 | | | 10 | | | 12 | | |
| Bit Rate (bits/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 |

**Table 13.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)**

| | φ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 12.288 | | | 14 | | | 14.7456 | | | 16 | | |
| Bit Rate (bits/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 13.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)**

| | φ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 17.2032 | | | 18 | | | 19.6608 | | | 20 | | |
| Bit Rate (bits/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 | 1 | 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 | 0 | 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 |

**Table 13.4 Bit Rates and BRR Settings in Clocked Synchronous Mode**

| Bit Rate (bits/s) | φ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | | 4 | | 8 | | 10 | | 16 | | 20 | |
| | n | N | n | N | n | N | n | N | n | N | n | N |
| 110 | 3 | 70 | — | — | — | — | — | — | — | — | — | — |
| 250 | 2 | 124 | 2 | 249 | 3 | 124 | — | — | 3 | 249 | — | — |
| 500 | 1 | 249 | 2 | 124 | 2 | 249 | — | — | 3 | 124 | — | — |
| 1k | 1 | 124 | 1 | 249 | 2 | 124 | — | — | 2 | 249 | — | — |
| 2.5k | 0 | 199 | 1 | 99 | 1 | 199 | 1 | 249 | 2 | 99 | 2 | 124 |
| 5k | 0 | 99 | 0 | 199 | 1 | 99 | 1 | 124 | 1 | 199 | 1 | 249 |
| 10k | 0 | 49 | 0 | 99 | 0 | 199 | 0 | 249 | 1 | 99 | 1 | 124 |
| 25k | 0 | 19 | 0 | 39 | 0 | 79 | 0 | 99 | 0 | 159 | 0 | 199 |
| 50k | 0 | 9 | 0 | 19 | 0 | 39 | 0 | 49 | 0 | 79 | 0 | 99 |
| 100k | 0 | 4 | 0 | 9 | 0 | 19 | 0 | 24 | 0 | 39 | 0 | 49 |
| 250k | 0 | 1 | 0 | 3 | 0 | 7 | 0 | 9 | 0 | 15 | 0 | 19 |
| 500k | 0 | 0√ | 0 | 1 | 0 | 3 | 0 | 4 | 0 | 7 | 0 | 9 |
| 1M | | | 0 | 0* | 0 | 1 | — | — | 0 | 3 | 0 | 4 |
| 2.5M | | | | | — | — | 0 | 0* | — | — | 0 | 1 |
| 5M | | | | | | | | | — | — | 0 | 0* |

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

Blank: No setting available

—: Setting possible, but error occurs

√ : Continuous transmit/receive not possible

The BRR setting is calculated as follows:

Asynchronous mode

$$N = [\phi/(64 \times 2^{2n-1} \times B)] \times 10^6 - 1$$

Clocked synchronous mode

$$N = [\phi/(8 \times 2^{2n-1} \times B)] \times 10^6 - 1$$

B: bit rate (bit/s)

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

φ: φ frequency (MHz)

n: baud rate generator clock source (n = 0, 1, 2, 3)

For the clock sources and values of n, see table 13.5.

| | | SMR Settings | |
|---|---|---|---|
| **n** | **Clock Source** | **CKS1** | **CKS0** |
| 0 | $\phi$ | 0 | 0 |
| 1 | $\phi/4$ | 0 | 1 |
| 2 | $\phi/16$ | 1 | 0 |
| 3 | $\phi/24$ | 1 | 1 |

Find the bit rate error for the asynchronous mode by the following formula.

$$\text{Error (\%)} = \{(\phi \times 10^6)/[(N + 1) \times B \times 64 \times 22n^{-1}] - 1\} \times 100$$

Table 13.5 indicates the maximum bit rates in the asynchronous mode when the baud rate generator is being used. Tables 13.6 and 13.7 show the maximum rates for external clock input.

**Table 13.5 Maximum Bit Rates for Various Frequencies with Baud Rate Generator (Asynchronous Mode)**

| | | Settings | |
|---|---|---|---|
| **ϕ (MHz)** | **Maximum Bit Rate (bits/s)** | **n** | **N** |
| 2 | 62500 | 0 | 0 |
| 2.097152 | 65536 | 0 | 0 |
| 2.4576 | 76800 | 0 | 0 |
| 3 | 93750 | 0 | 0 |
| 3.6864 | 115200 | 0 | 0 |
| 4 | 125000 | 0 | 0 |
| 4.9152 | 153600 | 0 | 0 |
| 5 | 156250 | 0 | 0 |
| 6 | 187500 | 0 | 0 |
| 6.144 | 192000 | 0 | 0 |
| 7.3728 | 230400 | 0 | 0 |
| 8 | 250000 | 0 | 0 |
| 9.8304 | 307200 | 0 | 0 |
| 10 | 312500 | 0 | 0 |
| 12 | 375000 | 0 | 0 |
| 12.288 | 384000 | 0 | 0 |
| 14 | 437500 | 0 | 0 |
| 14.7456 | 460800 | 0 | 0 |
| 16 | 500000 | 0 | 0 |
| 17.2032 | 537600 | 0 | 0 |
| 18 | 562500 | 0 | 0 |
| 19.6608 | 614400 | 0 | 0 |
| 20 | 625000 | 0 | 0 |

**Table 13.6 Maximum Bit Rates during External Clock Input (Asynchronous Mode)**

| $\phi$ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bits/s) |
|---|---|---|
| 2 | 0.5000 | 31250 |
| 2.097152 | 0.5243 | 32768 |
| 2.4576 | 0.6144 | 38400 |
| 3 | 0.7500 | 46875 |
| 3.6864 | 0.9216 | 57600 |
| 4 | 1.0000 | 62500 |
| 4.9152 | 1.2288 | 76800 |
| 5 | 1.2500 | 78125 |
| 6 | 1.5000 | 93750 |
| 6.144 | 1.5360 | 96000 |
| 7.3728 | 1.8432 | 115200 |
| 8 | 2.0000 | 125000 |
| 9.8304 | 2.4576 | 153600 |
| 10 | 2.5000 | 156250 |
| 12 | 3.0000 | 187500 |
| 12.288 | 3.0720 | 192000 |
| 14 | 3.5000 | 218750 |
| 14.7456 | 3.6834 | 230400 |
| 16 | 4.0000 | 250000 |
| 17.2032 | 4.3008 | 268800 |
| 18 | 4.5000 | 281250 |
| 19.6608 | 4.9152 | 307200 |
| 20 | 5.0000 | 312500 |

**Table 13.7 Maximum Bit Rates during External Clock Input (Clocked Synchronous Mode)**

| ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bits/s) |
|---|---|---|
| 2 | 0.3333 | 333333.3 |
| 4 | 0.6667 | 666666.7 |
| 6 | 1.0000 | 1000000.0 |
| 8 | 1.3333 | 1333333.3 |
| 10 | 1.6667 | 1666666.7 |
| 12 | 2.0000 | 2000000.0 |
| 14 | 2.3333 | 2333333.3 |
| 16 | 2.6667 | 2666666.7 |
| 18 | 3.0000 | 3000000.0 |
| 20 | 3.3333 | 3333333.3 |

## 13.3 Operation

### 13.3.1 Overview

The SCI has an asynchronous mode in which characters are synchronized individually, and a clocked synchronous mode in which communication is synchronized with clock pulses. Serial communication is possible in either mode. Asynchronous/clocked synchronous mode and the communication format are selected in the serial mode register (SMR), as shown in table 13.8. The SCI clock source is selected by the C/A bit in the serial mode register (SMR) and the CKE1 and CKE0 bits in the serial control register (SCR), as shown in table 13.9.

**Asynchronous Mode:**

- Data length is selectable: seven or eight bits.
- Parity and multiprocessor bits are selectable. So is the stop bit length (one or two bits). The preceding selections constitute the communication format and character length.
- In receiving, it is possible to detect framing errors (FER), parity errors (PER), overrun errors (ORER), and the break state.
- An internal or external clock can be selected as the SCI clock source.
  - — When an internal clock is selected, the SCI operates using the on-chip baud rate generator, and can output a serial clock signal with a frequency matching the bit rate.
  - — When an external clock is selected, the external clock input must have a frequency 16 times the bit rate. (The on-chip baud rate generator is not used.)

**Clocked Synchronous Mode:**

- The communication format has a fixed eight-bit data length.
- In receiving, it is possible to detect overrun errors (ORER).
- An internal or external clock can be selected as the SCI clock source.
  - — When an internal clock is selected, the SCI operates using the on-chip baud rate generator, and outputs a serial clock signal to external devices.
  - — When an external clock is selected, the SCI operates on the input serial clock. The on-chip baud rate generator is not used.

**Table 13.8 Serial Mode Register Settings and SCI Communication Formats**

| | SMR Settings | | | | | SCI Communication Format | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mode** | **Bit 7: C/A** | **Bit 6: CHR** | **Bit 5: PE** | **Bit 2: MP** | **Bit 3: STOP** | **Data Length** | **Parity Bit** | **Multipro-cessor Bit** | **Stop Bit Length** |
| Asynchronous | 0 | 0 | 0 | 0 | 0 | 8-bit | Absent | Absent | 1 bit |
| | | | | | 1 | | | | 2 bits |
| | | | 1 | | 0 | | Present | | 1 bit |
| | | | | | 1 | | | | 2 bits |
| | | 1 | 0 | | 0 | 7-bit | Absent | | 1 bit |
| | | | | | 1 | | | | 2 bits |
| | | | 1 | | 0 | | Present | | 1 bit |
| | | | | | 1 | | | | 2 bits |
| Asynchronous (multiprocessor format) | | 0 | * | 1 | 0 | 8-bit | Absent | Present | 1 bit |
| | | | * | | 1 | | | | 2 bits |
| | | 1 | * | | 0 | 7-bit | | | 1 bit |
| | | | * | | 1 | | | | 2 bits |
| Clocked synchronous | 1 | * | * | * | * | 8-bit | | Absent | None |

Note: Asterisks (*) in the table indicate don't-care bits.

**Table 13.9 SMR and SCR Settings and SCI Clock Source Selection**

| | SMR | SCR Settings | | SCI Transmit/Receive Clock | |
|---|---|---|---|---|---|
| **Mode** | **Bit 7: C/A** | **Bit 1: CKE1** | **Bit 0: CKE0** | **Clock Source** | **SCK Pin Function*** |
| Asynchronous mode | 0 | 0 | 0 | Internal | SCI does not use the SCK pin |
| | | | 1 | | Outputs a clock with frequency matching the bit rate |
| | | 1 | 0 | External | Inputs a clock with frequency 16 times the bit rate |
| | | | 1 | | |
| Clocked synch-ronous mode | 1 | 0 | 0 | Internal | Outputs the serial clock |
| | | | 1 | | |
| | | 1 | 0 | External | Inputs the serial clock |
| | | | 1 | | |

Note: Select the function in combination with the pin function controller (PFC).

### 13.3.2 Operation in Asynchronous Mode

In the asynchronous mode, each transmitted or received character begins with a start bit and ends with a stop bit. Serial communication is synchronized one character at a time.

The transmitting and receiving sections of the SCI are independent, so full duplex communication is possible. The transmitter and receiver are both double buffered, so data can be written and read while transmitting and receiving are in progress, enabling continuous transmitting and receiving.

Figure 13.2 shows the general format of asynchronous serial communication. In asynchronous serial communication, the communication line is normally held in the mark (high) state. The SCI monitors the line and starts serial communication when the line goes to the space (low) state, indicating a start bit. One serial character consists of a start bit (low), data (LSB first), parity bit (high or low), and stop bit (high), in that order.

When receiving in the asynchronous mode, the SCI synchronizes on the falling edge of the start bit. The SCI samples each data bit on the eighth pulse of a clock with a frequency 16 times the bit rate. Receive data is latched at the center of each bit.

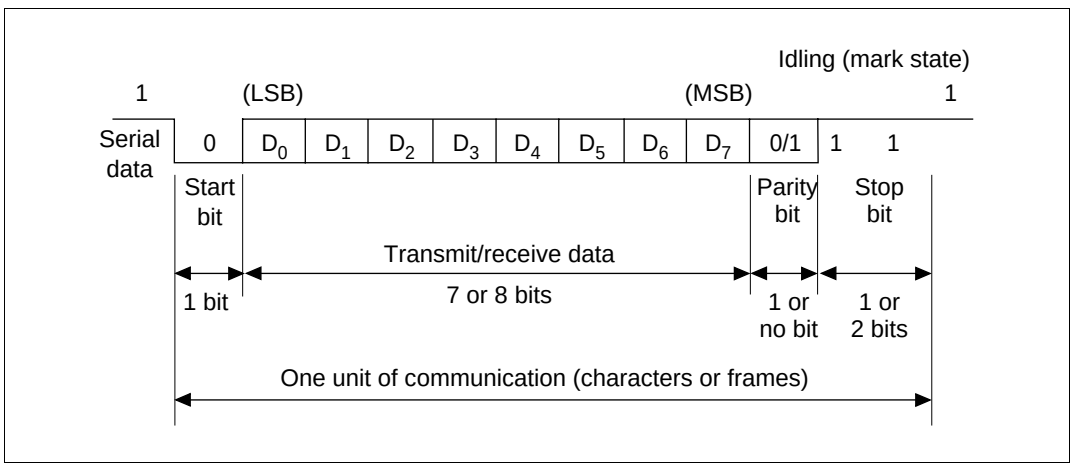



**Figure 13.2 Data Format in Asynchronous Communication (Example: 8-bit data with parity and two stop bits)**

**Transmit/Receive Formats:** Table 13.10 shows the 12 communication formats that can be selected in the asynchronous mode. The format is selected by settings in the serial mode register (SMR).

**Table 13.10 Serial Communication Formats (asynchronous mode)**

| SMR Bits | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CHR | PE | MP | STOP | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 0 | 0 | 0 | 0 | START | 8-Bit data | | | | | | | | STOP | | |
| 0 | 0 | 0 | 1 | START | 8-Bit data | | | | | | | | STOP | STOP | |
| 0 | 1 | 0 | 0 | START | 8-Bit data | | | | | | | | P | STOP | |
| 0 | 1 | 0 | 1 | START | 8-Bit data | | | | | | | | P | STOP | STOP |
| 1 | 0 | 0 | 0 | START | 7-Bit data | | | | | | | STOP | | | |
| 1 | 0 | 0 | 1 | START | 7-Bit data | | | | | | | STOP | STOP | | |
| 1 | 1 | 0 | 0 | START | 7-Bit data | | | | | | | P | STOP | | |
| 1 | 1 | 0 | 1 | START | 7-Bit data | | | | | | | P | STOP | STOP | |
| 0 | — | 1 | 0 | START | 8-Bit data | | | | | | | | MPB | STOP | |
| 0 | — | 1 | 1 | START | 8-Bit data | | | | | | | | MPB | STOP | STOP |
| 1 | — | 1 | 0 | START | 7-Bit data | | | | | | | MPB | STOP | | |
| 1 | — | 1 | 1 | START | 7-Bit data | | | | | | | MPB | STOP | STOP | |

—: Don't care bits.

Note: START: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

**Clock:** An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI transmit/receive clock. The clock source is selected by the C/$\overline{A}$ bit in the serial mode register (SMR) and bits CKE1 and CKE0 in the serial control register (SCR) (table 13.9).

When an external clock is input at the SCK pin, it must have a frequency equal to 16 times the

desired bit rate.

When the SCI operates on an internal clock, it can output a clock signal at the SCK pin. The frequency of this output clock is equal to the bit rate. The phase is aligned as in figure 13.3 so that the rising edge of the clock occurs at the center of each transmit data bit.

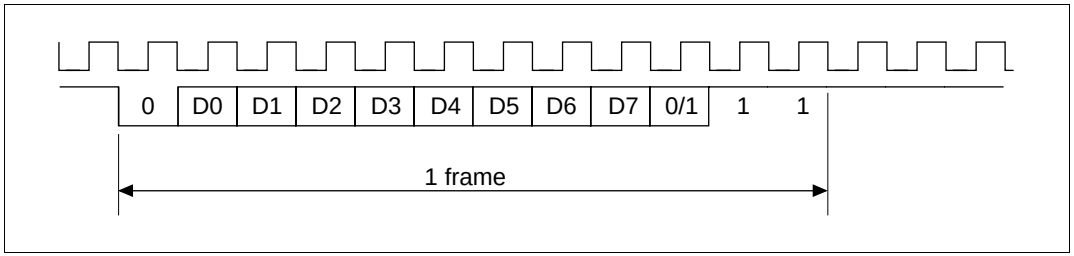



**Figure 13.3 Phase Relationship Between Output Clock and Serial Data (asynchronous mode)**

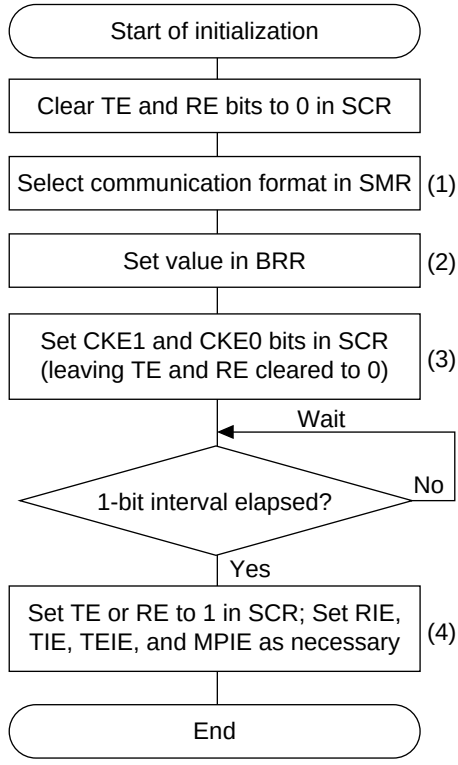
**Transmitting and Receiving Data (SCI initialization (asynchronous mode)):** Before transmitting or receiving, software must clear the TE and RE bits to 0 in the serial control register (SCR), then initialize the SCI as follows.

When changing the communication mode or format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR), which retain their previous contents.

When an external clock is used, the clock should not be stopped during initialization or subsequent operation. SCI operation becomes unreliable if the clock is stopped.

Figure 13.4 is a sample flowchart for initializing the SCI. The procedure for initializing the SCI is as follows:

1. Select the communication format in the serial mode register (SMR).
2. Write the value corresponding to the bit rate in the bit rate register (BRR) unless an external clock is used.
3. Select the clock source in the serial control register (SCR). Leave RIE, TIE, TEIE, MPIE, TE and RE cleared to 0. If clock output is selected in asynchronous mode, clock output starts immediately after the setting is made to SCR.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR) to 1. Also set RIE, TIE, TEIE and MPIE as necessary. Setting TE or RE enables the SCI to use the TxD or RxD pin. The initial states are the mark transmit state, and the idle receive state (waiting for a start bit).

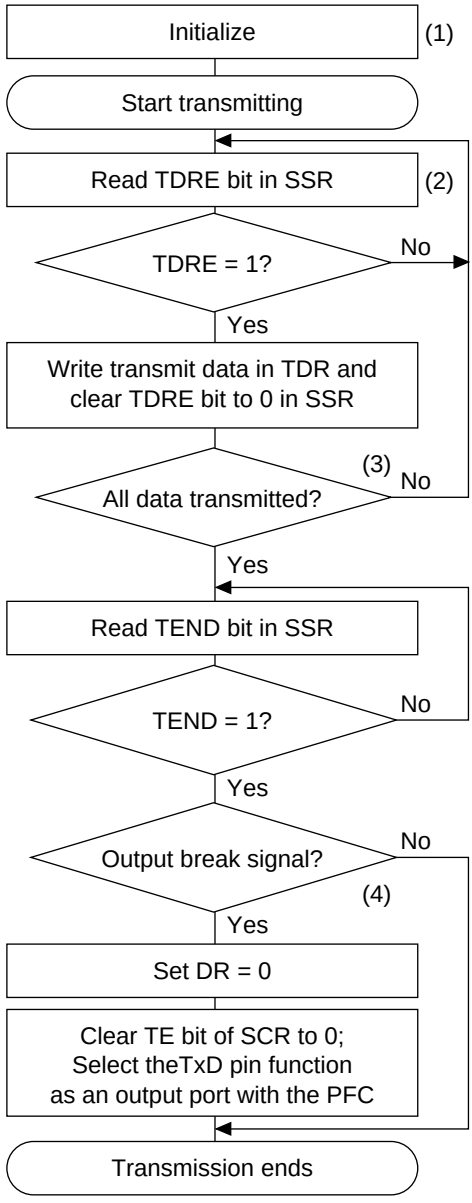



Note: Circled numbers refer to the preceding procedure.

**Figure 13.4 Sample Flowchart for SCI Initialization**

**Transmitting Serial Data (asynchronous mode):** Figure 13.5 shows a sample flowchart for transmitting serial data. The procedure for transmitting serial data is as follows:

1. SCI initialization: select the TxD pin function with the PFC.
2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0.
3. To continue transmitting serial data: read the TDRE bit to check whether it is safe to write (1); if so, write data in TDR, then clear TDRE to 0. When the DMAC is started by a transmit-data-empty interrupt request (TXI) to write data in TDR, the TDRE bit is checked and cleared automatically.
4. To output a break signal at the end of serial transmission: set the DR bit to 0 (I/$\overline{O}$ data port register), then clear TE to 0 in SCR and set the TxD pin function as output port with the PFC.




Note: Circled numbers refer to the preceding procedure.

**Figure 13.5 Sample Flowchart for Transmitting Serial Data**

In transmitting serial data, the SCI operates as follows:

1. The SCI monitors the TDRE bit in the SSR. When TDRE is cleared to 0, the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from the TDR into the transmit shift register (TSR).
2. After loading the data from the TDR into the TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) is set to 1 in the SCR, the SCI requests a transmit-data-empty interrupt (TXI) at this time.

Serial transmit data is transmitted in the following order from the TxD pin:

1. Start bit: one 0 bit is output.
2. Transmit data: seven or eight bits of data are output, LSB first.
3. Parity bit or multiprocessor bit: one parity bit (even or odd parity) or one multiprocessor bit is output. Formats in which neither a parity bit nor a multiprocessor bit is output can also be selected.
4. Stop bit: one or two 1 bits (stop bits) are output.
5. Mark state: output of 1 bits continues until the start bit of the next transmit data.
6. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI loads new data from the TDR into the TSR, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit to 1 in the SSR, outputs the stop bit, then continues output of 1 bits in the mark state. If the transmit-end interrupt enable bit (TEIE) in the SCR is set to 1, a transmit-end interrupt (TEI) is requested.

Figure 13.6 shows an example of SCI transmit operation in the asynchronous mode.

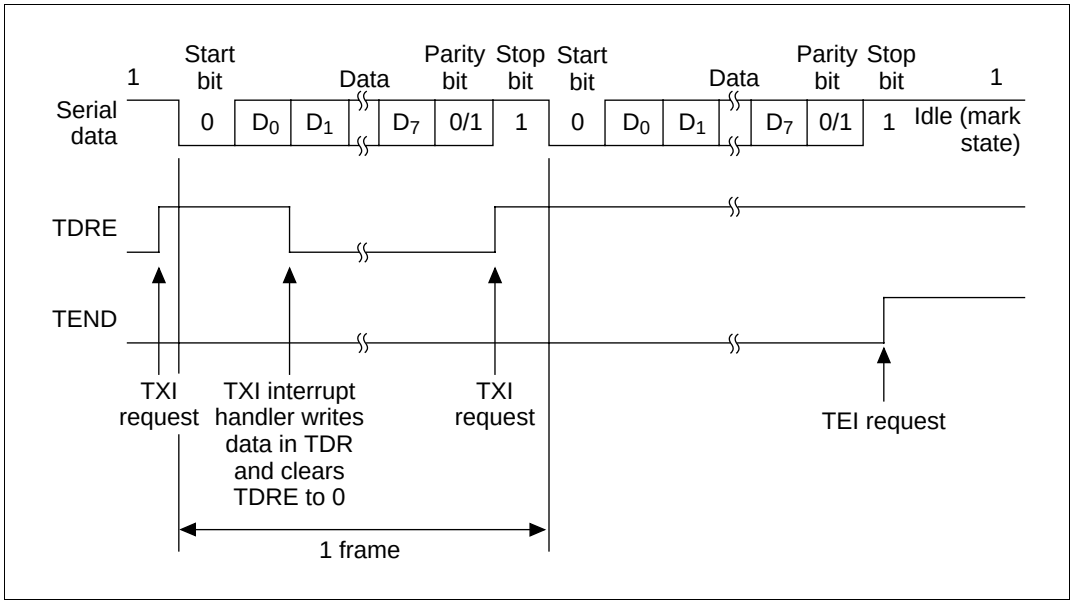
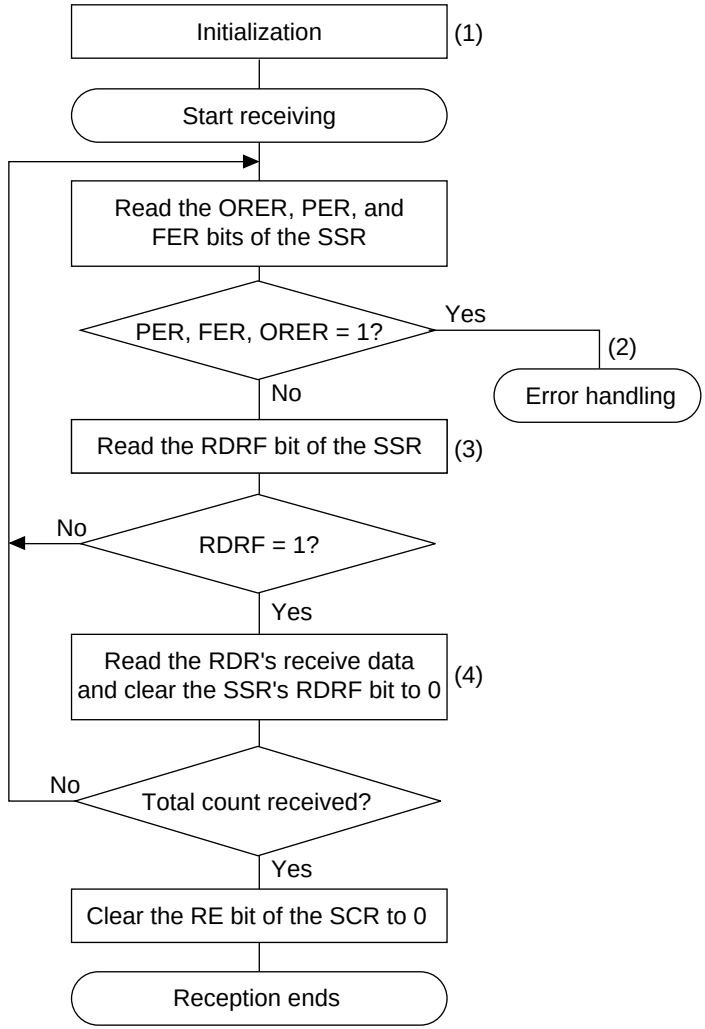



**Figure 13.6 Example of SCI Transmit Operation in Asynchronous Mode (8-bit data with parity and one stop bit)**

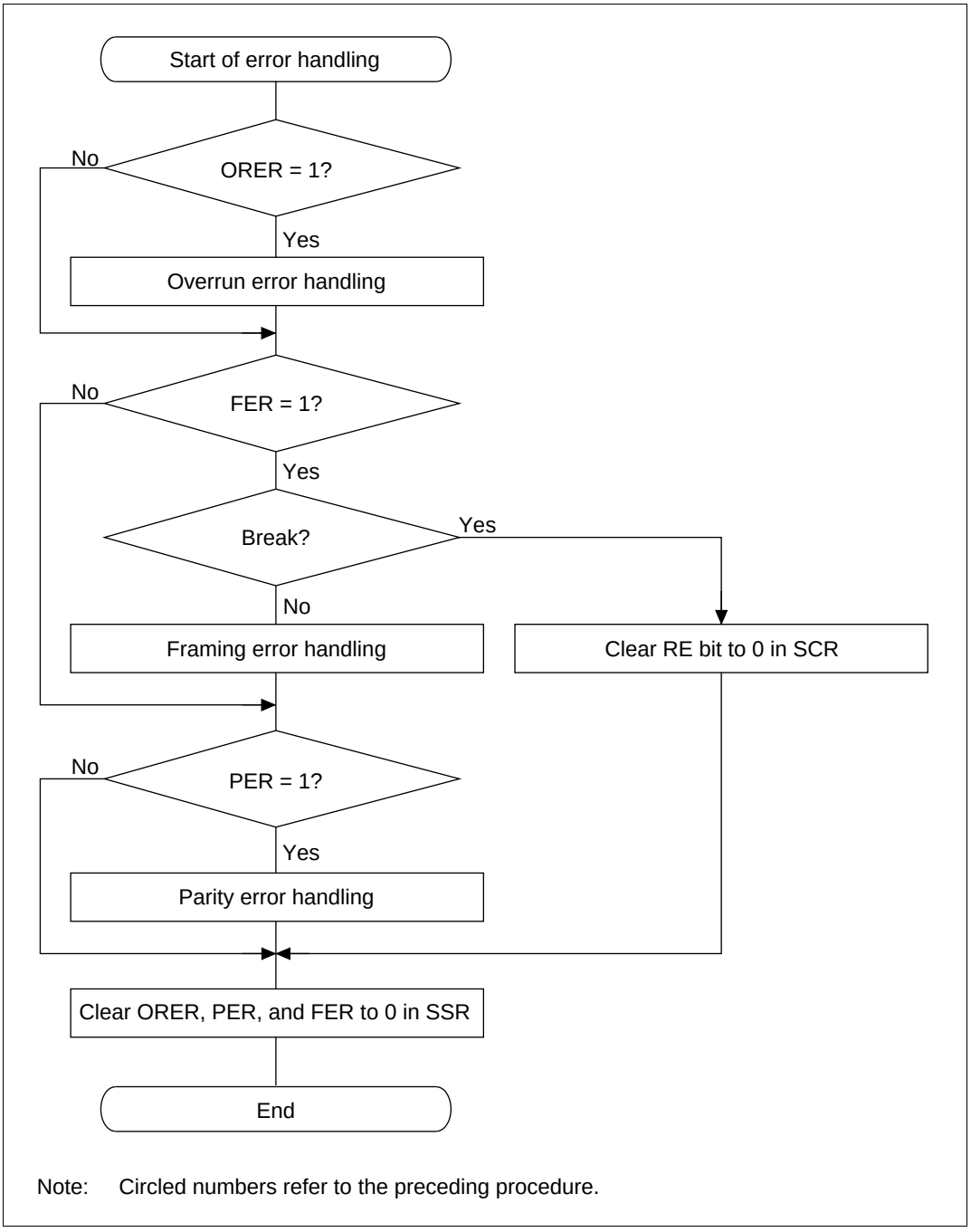
Receiving Serial Data (asynchronous mode): Figure 13.7 shows a sample flowchart for receiving serial data. The procedure for receiving serial data is listed below.

1. SCI initialization: select the RxD pin function with the PFC.
2. Receive error handling and break detection: if a receive error occurs, read the ORER, PER and FER bits of the SSR to identify the error. After executing the necessary error handling, clear ORER, PER and FER all to 0. Receiving cannot resume if ORER, PER or FER remains set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
3. SCI status check and receive data read: read the serial status register (SR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. To continue receiving serial data: read RDRF and RDR, and clear RDRF to 0 before the stop bit of the current frame is received. If the DMAC is started by a receive-data-full interrupt (RXI) to read RDR, the RDRF bit is cleared automatically so this step is unnecessary.




Note: Circled numbers refer to the preceding procedure.

**Figure 13.7 Sample Flowchart for Receiving Serial Data**




Note: Circled numbers refer to the preceding procedure.

**Figure 13.7 Sample Flowchart for Receiving Serial Data (cont)**

In receiving, the SCI operates as follows:

1. The SCI monitors the receive data line. When it detects a start bit (0), the SCI synchronizes internally and starts receiving.
2. Receive data is shifted into the RSR in order from the LSB to the MSB.
3. The parity bit and stop bit are received. After receiving these bits, the SCI makes the following checks:

   a. Parity check: The number of 1s in the receive data must match the even or odd parity setting of the O/$\overline{E}$ bit in the SMR.
   b. Stop bit check: The stop bit value must be 1. If there are two stop bits, only the first stop bit is checked.
   c. Status check: RDRF must be 0 so that receive data can be loaded from the RSR into the RDR.

   If these checks all pass, the SCI sets RDRF to 1 and stores the received data in the RDR. If one of the checks fails (receive error), the SCI operates as indicated in table 13.11.

   Note: When a receive error flag is set, further receiving is disabled. The RDRF bit is not set to 1. Be sure to clear the error flags.

4. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCR, the SCI requests a receive-data-full interrupt (RXI). If one of the error flags (ORER, PER, or FER) is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCR is also set to 1, the SCI requests a receive-error interrupt (ERI).

Figure 13.8 shows an example of SCI receive operation in the asynchronous mode.

**Table 13.11 Receive Error Conditions and SCI Operation**

| Receive Error | Abbreviation | Condition | Data Transfer |
|---|---|---|---|
| Overrun error | ORER | Receiving of next data ends while RDRF is still set to 1 in SSR | Receive data not loaded from RSR into RDR |
| Framing error | FER | Stop bit is 0 | Receive data loaded from RSR into RDR |
| Parity error | PER | Parity of receive data differs from even/odd parity setting in SMR | Receive data loaded from RSR into RDR |

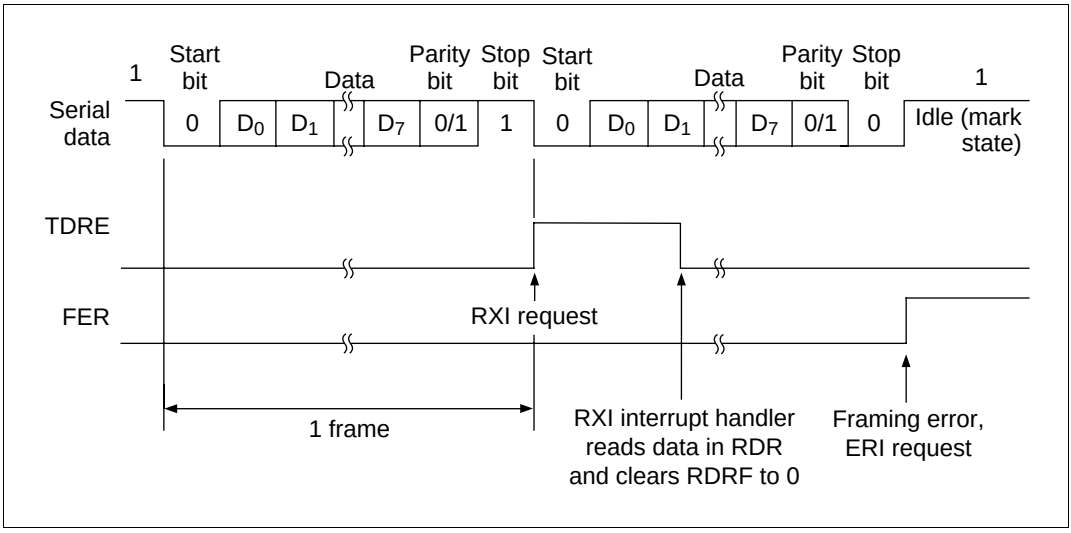



**Figure 13.8 Example of SCI Receive Operation (8-bit data with parity and one stop bit)**

### 13.3.3 Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line. The processors communicate in the asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by a unique ID. A serial communication cycle consists of an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit distinguishes ID-sending cycles from data-sending cycles. The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0.

Receiving processors skip incoming data until they receive data with the multiprocessor bit set to 1. When they receive data with the multiprocessor bit set to 1, receiving processors compare the data with their IDs. The receiving processor with a matching ID continues to receive further incoming data. Processors with IDs not matching the received data skip further incoming data until they again receive data with the multiprocessor bit set to 1. Multiple processors can send and receive data in this way.

Figure 13.9 shows the example of communication among processors using the multiprocessor format.

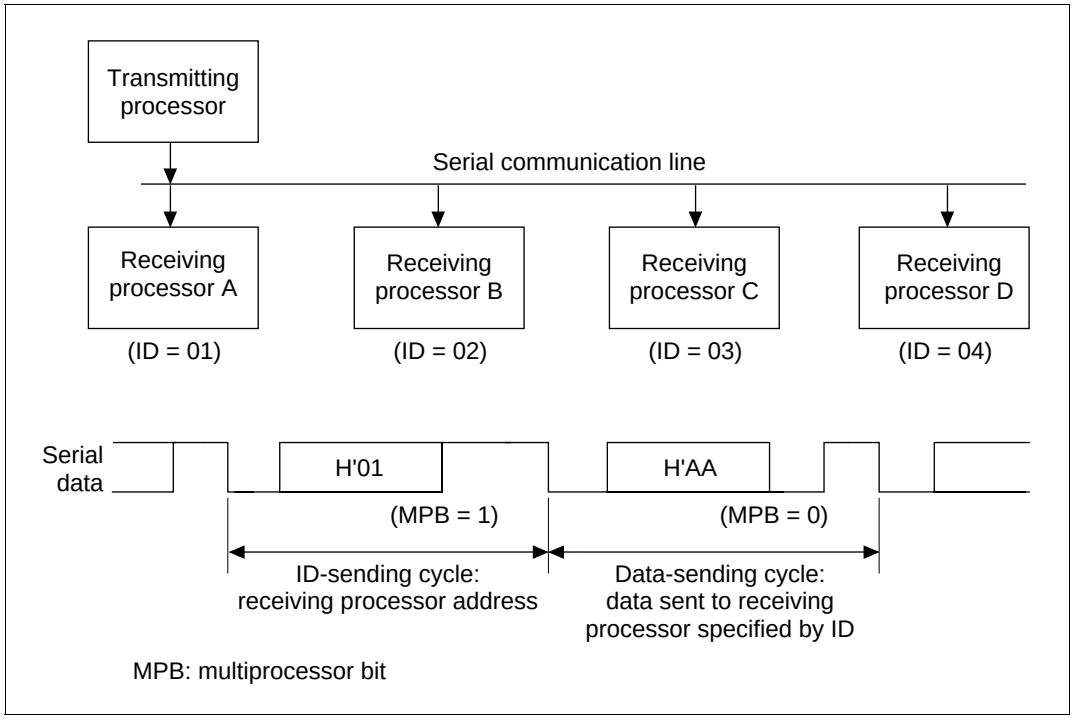


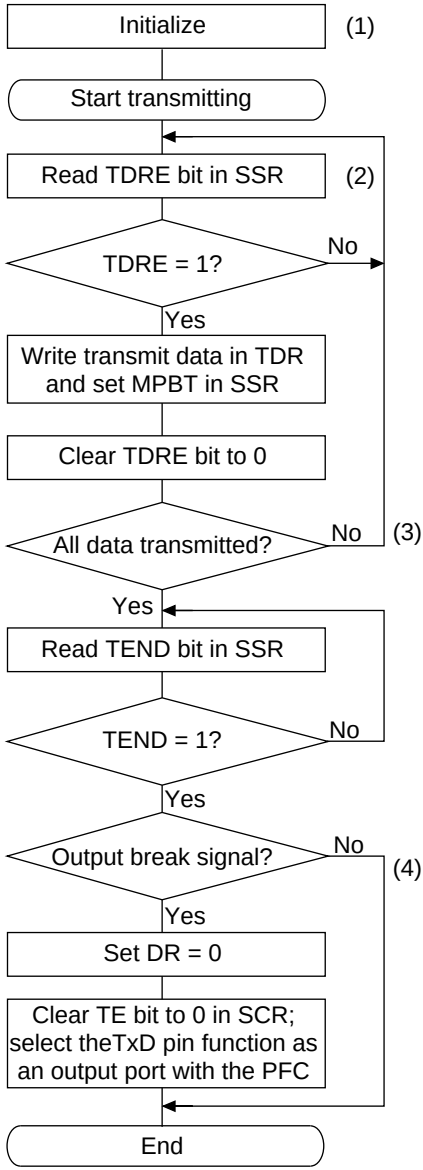

**Figure 13.9 Example of Communication among Processors Using Multiprocessor Format (sending data H'AA to receiving processor A)**

**Communication Formats:** Four formats are available. Parity-bit settings are ignored when the multiprocessor format is selected. For details see table 13.8.

**Clock:** See the description in the asynchronous mode section.

**Transmitting Multiprocessor Serial Data:** Figure 13.10 shows a sample flowchart for transmitting multiprocessor serial data. The procedure for transmitting multiprocessor serial data is listed below.

1. SCI initialization: select the TxD pin function with the PFC.
2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR). Also set MPBT (multiprocessor bit transfer) to 0 or 1 in SSR. Finally, clear TDRE to 0.
3. To continue transmitting serial data: read the TDRE bit to check whether it is safe to write (1); if so, write data in TDR, then clear TDRE to 0. When the DMAC is started by a transmit-data-empty interrupt request (TXI) to write data in TDR, the TDRE bit is checked and cleared automatically.
4. To output a break signal at the end of serial transmission: set the DR bit to 0 (I/O data port register), then clear TE to 0 in SCR and set the TxD pin function as output port with the PFC.




Note: Circled numbers refer to the preceding procedure.

**Figure 13.10 Sample Flowchart for Transmitting Multiprocessor Serial Data**

**In transmitting serial data, the SCI operates as follows:**

1. The SCI monitors the TDRE bit in the SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from the TDR into the transmit shift register (TSR).
2. After loading the data from the TDR into the TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCR is set to 1, the SCI requests a transmit-data-empty interrupt (TXI) at this time.

Serial transmit data is transmitted in the following order from the TxD pin (figure 13.11):

1. Start bit: one 0 bit is output.
2. Transmit data: seven or eight bits are output, LSB first.
3. Multiprocessor bit: one multiprocessor bit (MPBT value) is output.
4. Stop bit: one or two 1 bits (stop bits) are output.
5. Mark state: output of 1 bits continues until the start bit of the next transmit data.
6. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI loads data from the TDR into the TSR, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in the SSR to 1, outputs the stop bit, then continues output of 1 bits in the mark state. If the transmit-end interrupt enable bit (TEIE) in the SCR is set to 1, a transmit-end interrupt (TEI) is requested at this time.

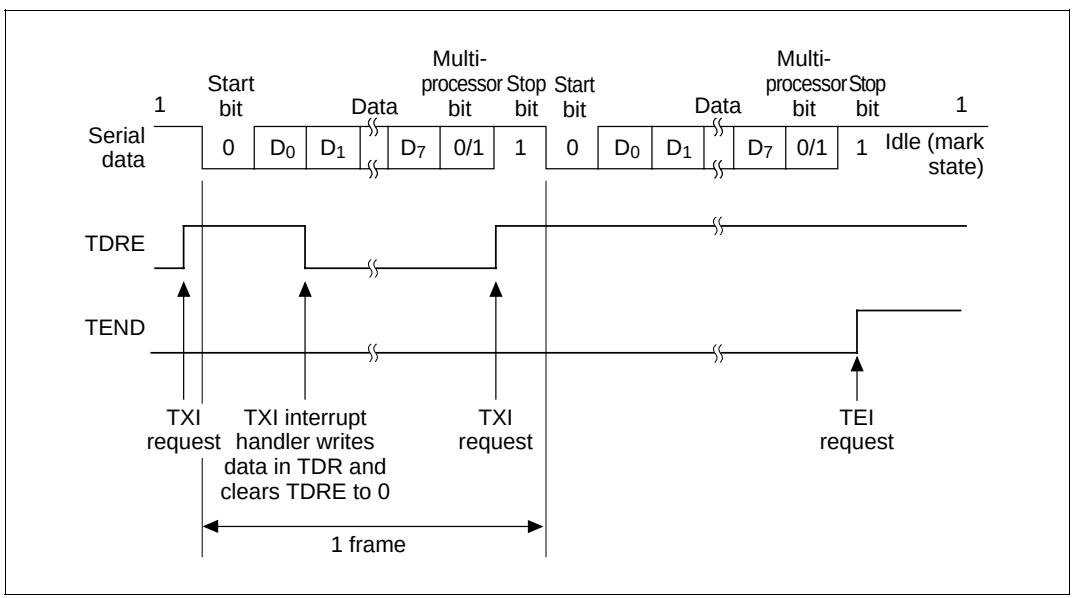



**Figure 13.11 Example of SCI Multiprocessor Transmit Operation (8-bit data with multiprocessor bit and one stop bit)**

**Receiving Multiprocessor Serial Data:** Figure 13.12 shows a sample flowchart for receiving multiprocessor serial data. The procedure for receiving multiprocessor serial data is listed below.

1. SCI initialization: select the RxD pin function with the PFC.
2. ID receive cycle: set the MPIE bit in the serial control register (SCR) to 1.
3. SCI status check and compare to ID reception: read the serial status register (SSR), check that RDRF is set to 1, then read data from the receive data register (RDR) and compare with the processor's own ID. If the ID does not match the receive data, set MPIE to 1 again and clear RDRF to 0. If the ID matches the receive data, clear RDRF to 0.
4. Receive error handling and break detection: if a receive error occurs, read the ORER and FER bits in SSR to identify the error. After executing the necessary error handling, clear both ORER and FER to 0. Receiving cannot resume if ORER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
5. SCI status check and data receiving: read SSR, check that RDRF is set to 1, then read data from the receive data register (RDR).

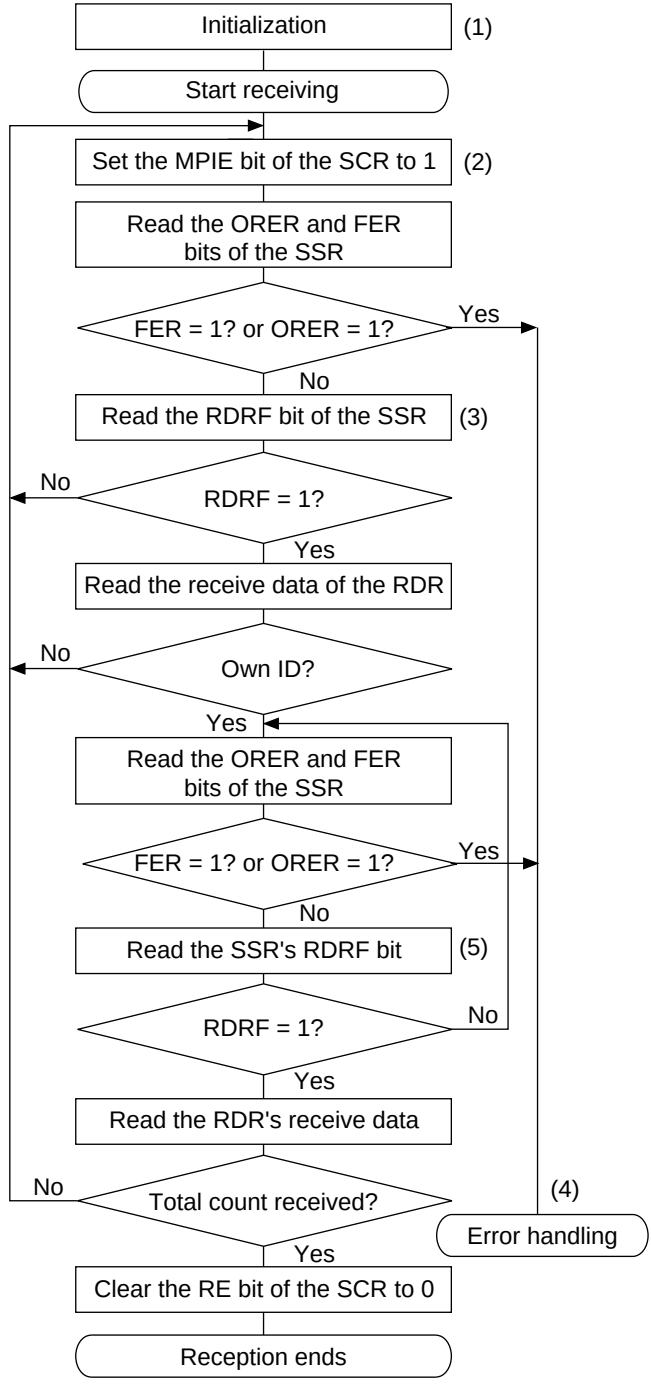



**Figure 13.12 Sample Flowchart for Receiving Multiprocessor Serial Data**

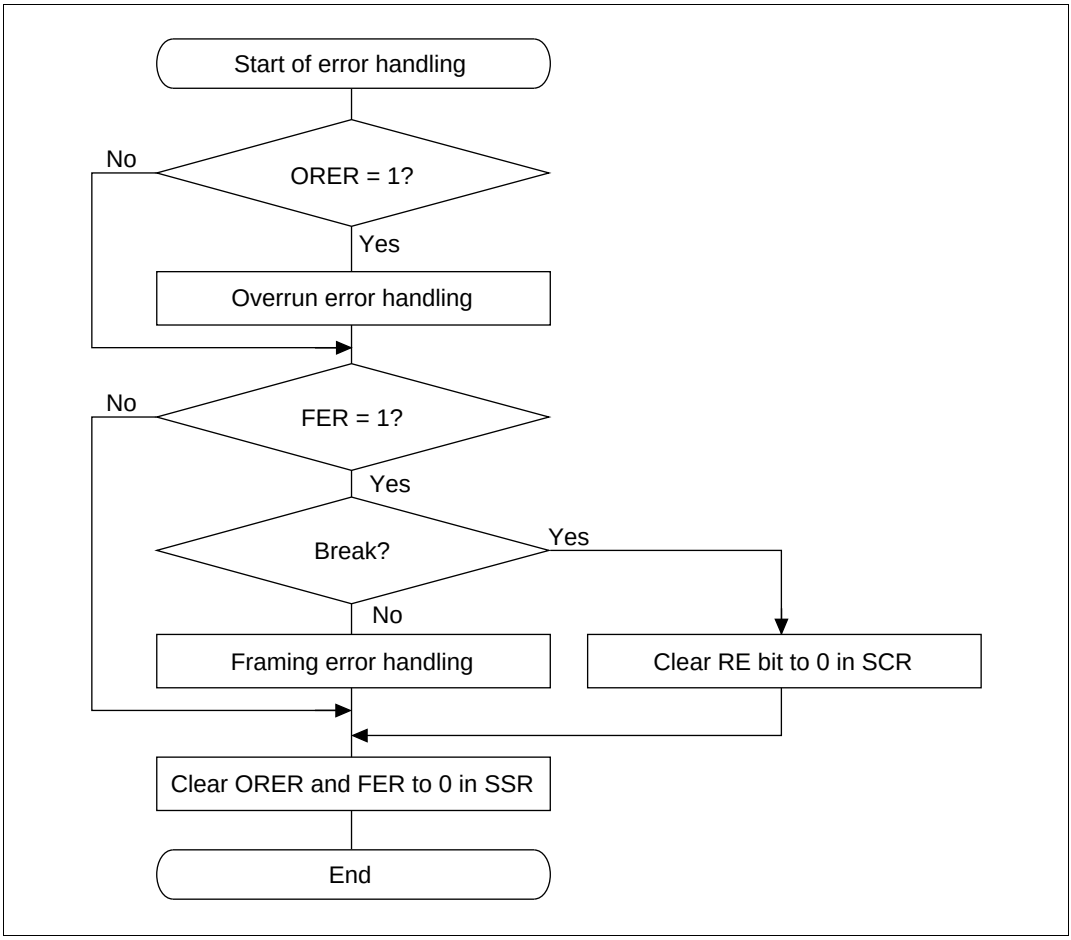



**Figure 13.12 Sample Flowchart for Receiving Multiprocessor Serial Data (cont)**

Figure 13.13 shows an example of SCI receive operation using a multiprocessor format.

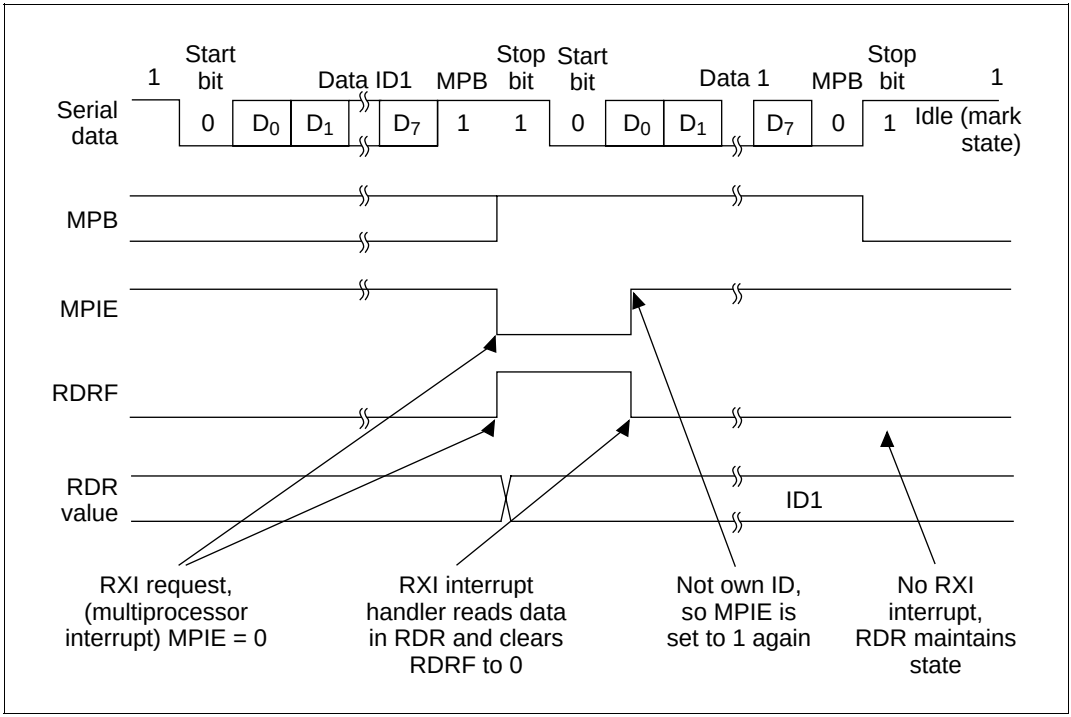



**Figure 13.13 Example of SCI Receive Operation (own ID does not match data) (8-bit data with multiprocessor bit and one stop bit)**

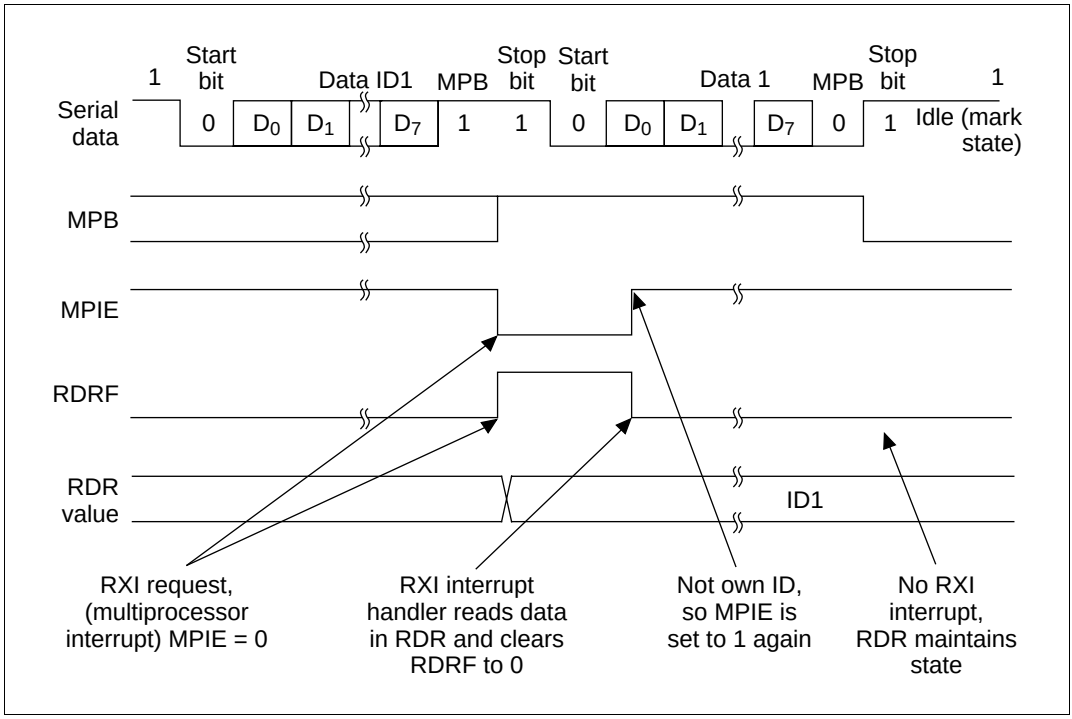



**Figure 13.13 Example of SCI Receive Operation (own ID matches data) (8-bit data with multiprocessor bit and one stop bit) (cont)**

### 13.3.4 Clocked Synchronous Operation

In the clocked synchronous mode, the SCI transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

The SCI transmitter and receiver share the same clock but are otherwise independent, so full duplex communication is possible. The transmitter and receiver are also double buffered, so continuous transmitting or receiving is possible by reading or writing data while transmitting or receiving is in progress.

Figure 13.14 shows the general format in clocked synchronous serial communication.

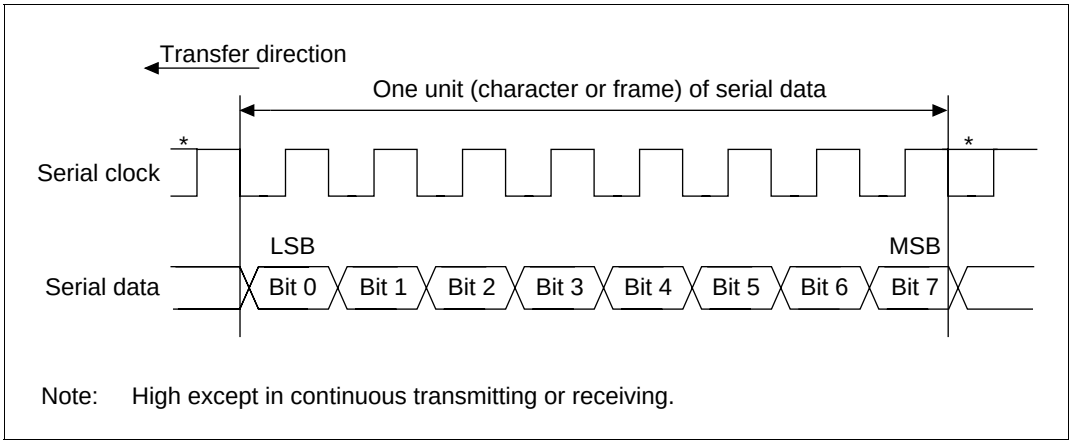



**Figure 13.14 Data Format in Clocked Synchronous Communication**

In clocked synchronous serial communication, each data bit is output on the communication line from one falling edge of the serial clock to the next. Data are guaranteed valid at the rising edge of the serial clock. In each character, the serial data bits are transmitted in order from the LSB (first) to the MSB (last). After output of the MSB, the communication line remains in the state of the MSB. In the clocked synchronous mode, the SCI transmits or receives data by synchronizing with the falling edge of the serial clock.

**Communication Format:** The data length is fixed at eight bits. No parity bit or multiprocessor bit can be added.

**Clock:** An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI transmit/receive clock. The clock source is selected by the C/$\overline{A}$ bit in the serial mode register (SMR) and bits CKE1 and CKE0 in the serial control register (SCR). See table 13.6.

When the SCI operates on an internal clock, it outputs the clock signal at the SCK pin. Eight clock pulses are output per transmitted or received character. When the SCI is not transmitting or receiving, the clock signal remains in the high state.

Figure 13.15 shows an example of SCI transmit operation. In transmitting serial data, the SCI operates as follows.

1. The SCI monitors the TDRE bit in the SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from the TDR into the transmit shift register (TSR).
2. After loading the data from the TDR into the TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCR is set to 1, the SCI requests a transmit-data-empty interrupt (TXI) at this time.

   If clock output is selected, the SCI outputs eight serial clock pulses. If an external clock source

is selected, the SCI outputs data in synchronization with the input clock. Data are output from the TxD pin in order from the LSB (bit 0) to the MSB (bit 7).

3. The SCI checks the TDRE bit when it outputs the MSB (bit 7). If TDRE is 0, the SCI loads data from the TDR into the TSR, transmits the MSB, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in the SSR to 1, transmits the MSB, then holds the transmit data pin (TxD) in the MSB state. If the transmit-end interrupt enable bit (TEIE) in the SCR is set to 1, a transmit-end interrupt (TEI) is requested at this time.
4. After the end of serial transmission, the SCK pin is held in the high state.

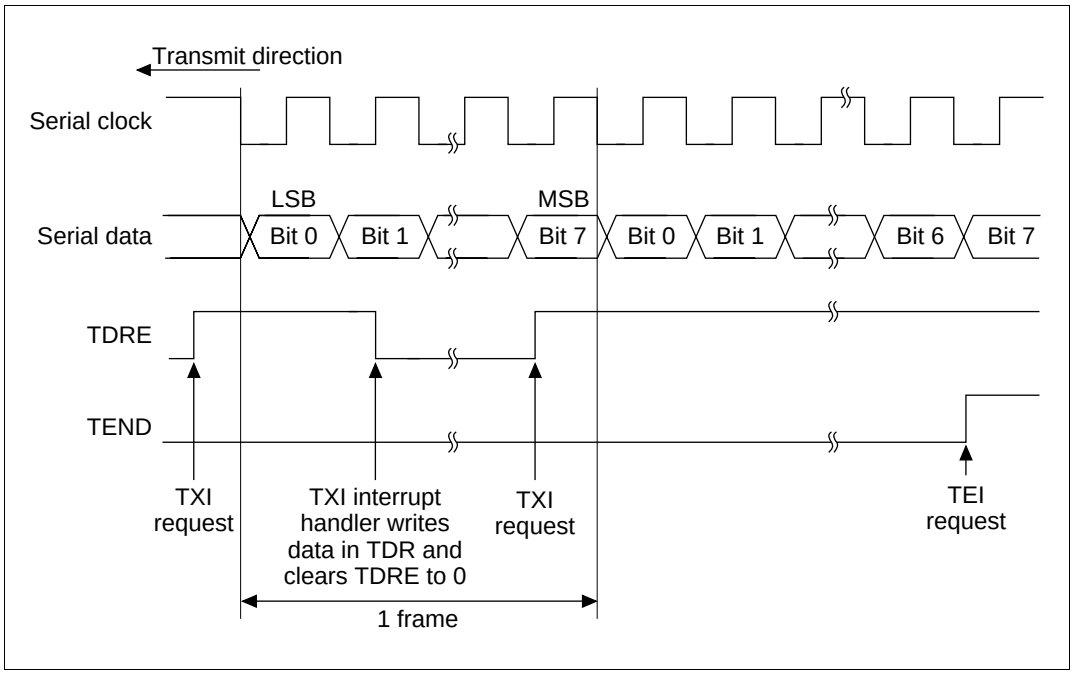


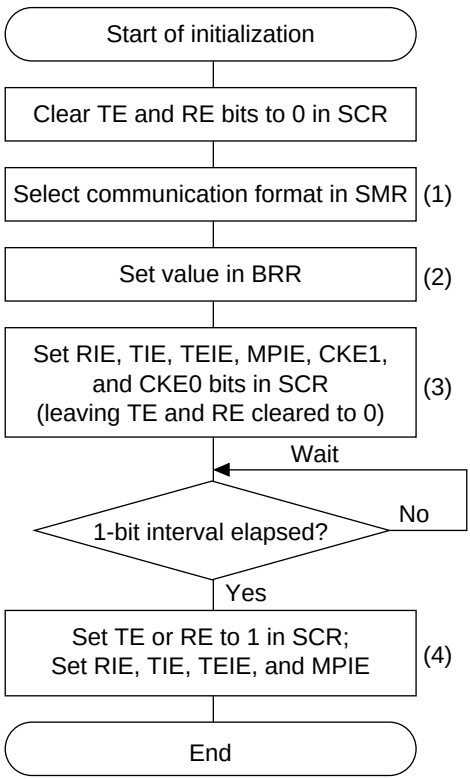

**Figure 13.15 Example of SCI Transmit Operation**

**Transmitting and Receiving Data:** SCI Initialization (clocked synchronous mode): Before transmitting or receiving, software must clear the TE and RE bits to 0 in the serial control register (SCR), then initialize the SCI as follows.

When changing the communication mode or format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR), which retain their previous contents.

Figure 13.16 is a sample flowchart for initializing the SCI. The procedure for initializing the SCI is listed below.

1. Select the communication format in the serial mode register (SMR).
2. Write the value corresponding to the bit rate in the bit rate register (BRR) unless an external clock is used.
3. Select the clock source in the serial control register (SCR). Leave RIE, TIE, TEIE, MPIE, TE and RE cleared to 0.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR) to 1. Also set RIE, TIE, TEIE and MPIE. Setting the corresponding bit of the pin function controller, TE and RE enables the SCI to use the TxD or RxD pin.




Note: High except in continuous transmitting or receiving.

**Figure 13.16 Sample Flowchart for SCI Initialization**

**Transmitting Serial Data (clocked synchronous mode):** Figure 13.17 shows a sample flowchart for transmitting serial data. The procedure for transmitting serial data is listed below.

1. SCI initialization: select the TxD pin function with the PFC.
2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
3. To continue transmitting serial data: read the TDRE bit to check whether it is safe to write (1); if so, write data in TDR, then clear TDRE to 0. When the DMAC is started by a transmit-data-empty interrupt request (TXI) to write data in TDR, the TDRE bit is checked and cleared automatically.

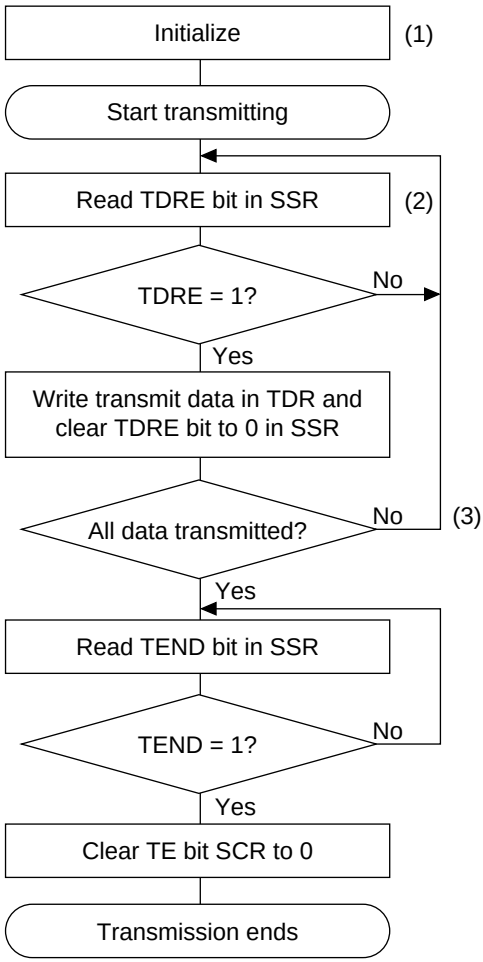



**Figure 13.17 Sample Flowchart for Serial Transmitting**

**Receiving Serial Data (clocked synchronous mode):** Figure 13.18 shows a sample flowchart for receiving serial data. When switching from the asynchronous mode to the clocked synchronous mode, make sure that ORER, PER, and FER are cleared to 0. If PER or FER is set to 1, the RDRF bit will not be set and both transmitting and receiving will be disabled. Figure 13.19 shows an example of SCI recieve operation.

The procedure for recieving serial data is listed below.

1. SCI initialization: select the RxD pin function with the PFC.
2. Receive error handling: if a receive error occurs, read the ORER bit in SSR to identify the error. After executing the necessary error handling, clear ORER to 0. Transmitting/receiving

cannot resume if ORER remains set to 1.

3. SCI status check and receive data read: read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. To continue receiving serial data: read RDR, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. If the DMAC is started by a receive-data-full interrupt (RXI) to read RDR, the RDRF bit is cleared automatically so this step is unnecessary.

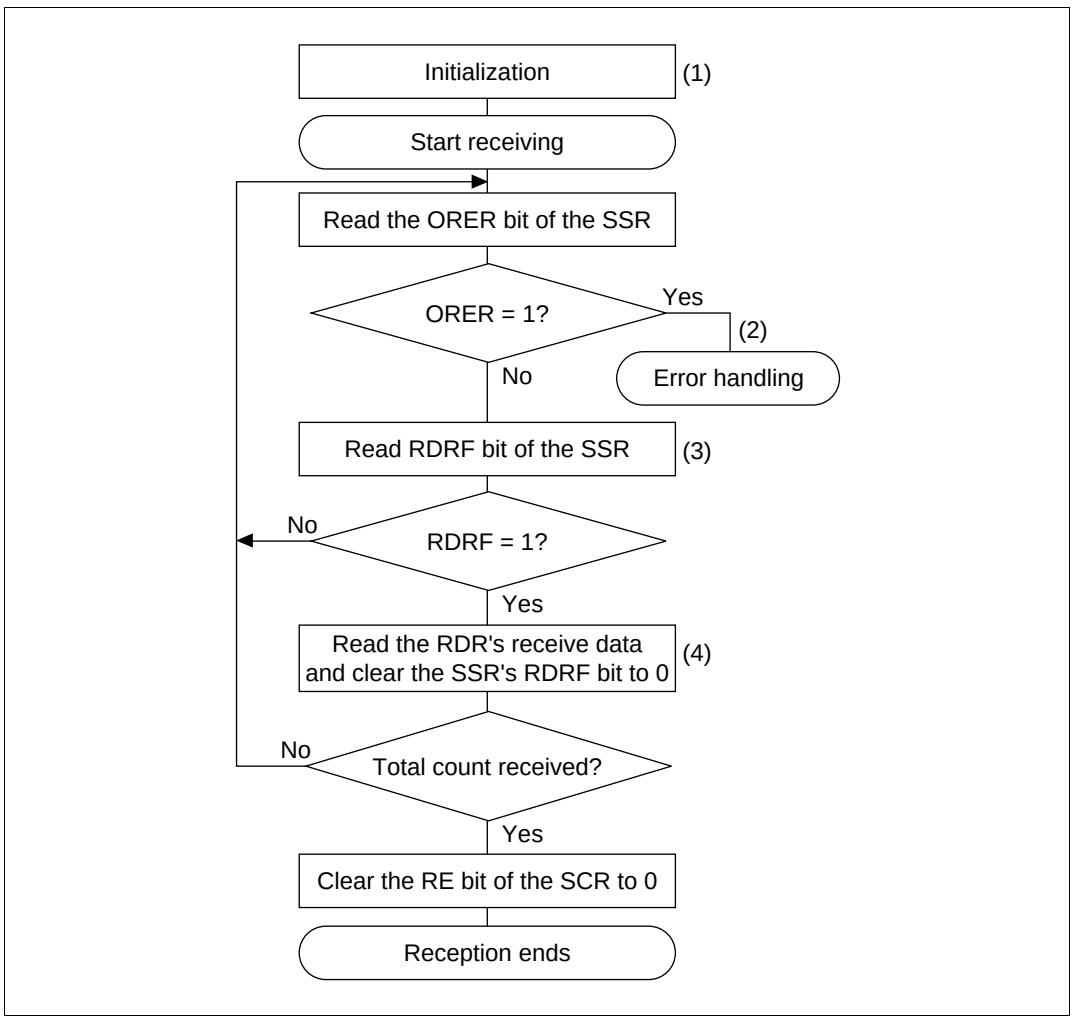



**Figure 13.18 Sample Flowchart for Serial Receiving**

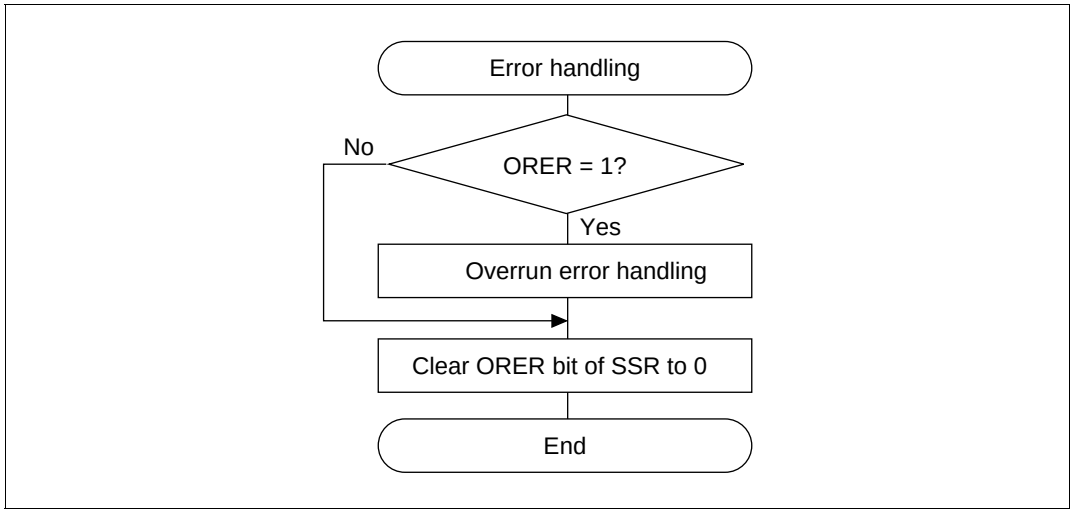



**Figure 13.18 Sample Flowchart for Serial Receiving (cont)**

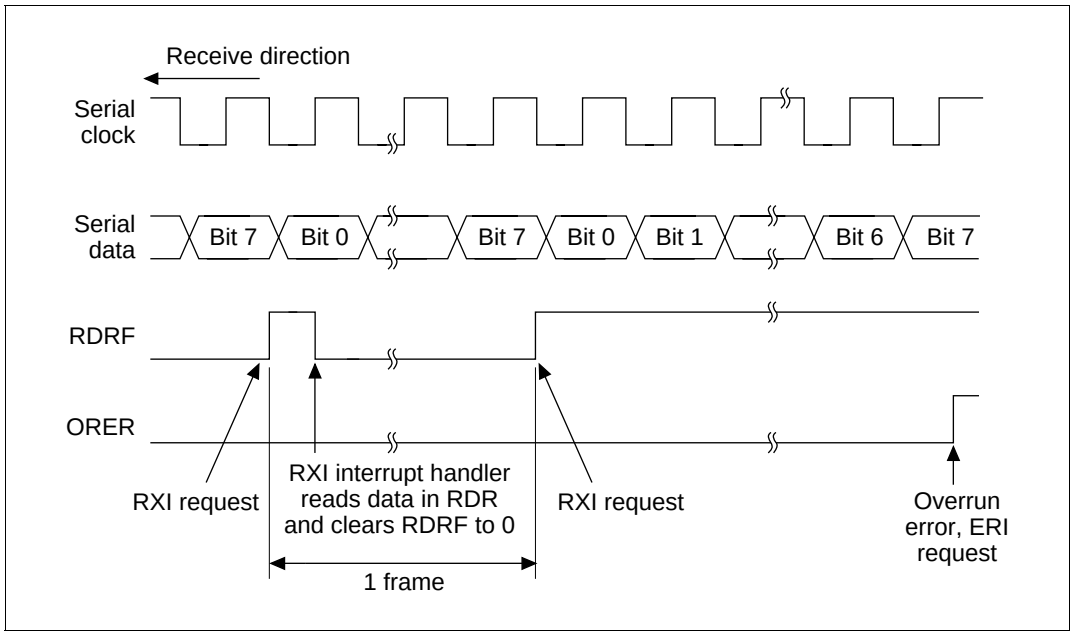



**Figure 13.19 Example of SCI Receive Operation**

In receiving, the SCI operates as follows:

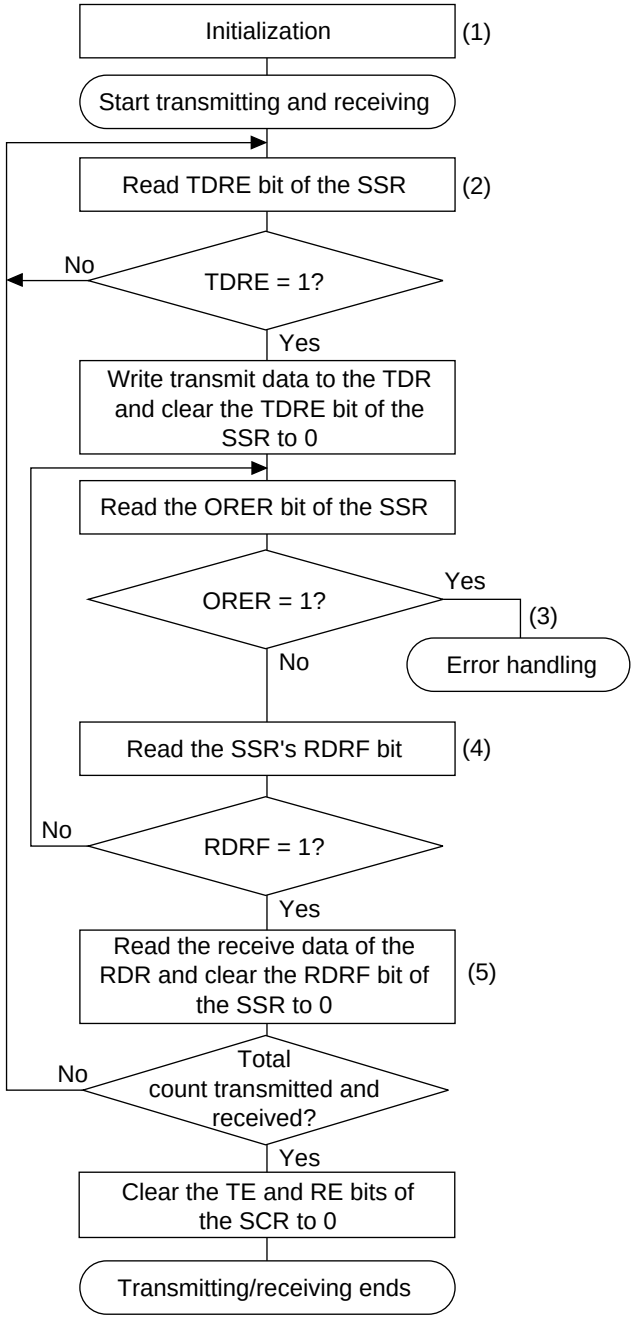
1. The SCI synchronizes with serial clock input or output and initializes internally.
2. Receive data is shifted into the RSR in order from the LSB to the MSB. After receiving the data, the SCI checks that RDRF is 0 so that receive data can be loaded from the RSR into the

RDR. If this check passes, the SCI sets RDRF to 1 and stores the received data in the RDR. If the check does not pass (receive error), the SCI operates as indicated in table 13.8. When the error flag is set to 1 and the RDRF bit is cleared to 0, the RDRF bit will not be set to 1 during reception. When restarting the reception, make sure to clear the error flag to 0.

3. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCR, the SCI requests a receive-data-full interrupt (RXI). If the ORER bit is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCR is also set to 1, the SCI requests a receive-error interrupt (ERI).

**Transmitting and Receiving Serial Data Simultaneously (clocked synchronous mode):** Figure 13.20 shows a sample flowchart for transmitting and receiving serial data simultaneously. The procedure for transmitting and receiving serial data simultaneously is listed below.

1. SCI initialization: select the TxD and RxD pin function with the PFC.
2. SCI status check and transmit data write: read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. The TXI interrupt can also be used to determine if the TDRE bit has changed from 0 to 1.
3. Receive error handling: if a receive error occurs, read the ORER bit in SSR to identify the error. After executing the necessary error handling, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
4. SCI status check and receive data read: read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
5. To continue transmitting and receiving serial data: read the RDRF bit and RDR, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. Also read the TDRE bit to check whether it is safe to write (1); if so, write data in TDR, then clear TDRE to 0 before the MSB (bit 7) of the current frame is transmitted. When the DMAC is started by a transmit-data-empty interrupt request (TXI) to write data in TDR, the TDRE bit is checked and cleared automatically. When the DMAC is started by a receive-data-full interrupt (RXI) to read RDR, the RDRF bit is cleared automatically.




Note: In switching from transmitting or receiving to simultaneous transmitting and receiving, clear both TE and RE to 0, then set both TE and RE to 1.

**Figure 13.20 Sample Flowchart for Serial Transmitting**

## 13.4 SCI Interrupt Sources and the DMAC

The SCI has four interrupt sources in each channel: transmit-end (TEI), receive-error (ERI), receive-data-full (RXI), and transmit-data-empty (TXI). Table 13.12 lists the interrupt sources and indicates their priority. These interrupts can be enabled and disabled by the TIE, RIE, and TEIE bits in the serial control register (SCR). Each interrupt request is sent separately to the interrupt controller.

TXI is requested when the TDRE bit in the SSR is set to 1. TXI can start the direct memory access controller (DMAC) to transfer data. TDRE is automatically cleared to 0 when the DMAC executes the data transfer to the transmit data register (TDR).

RXI is requested when the RDRF bit in the SSR is set to 1. RXI can start the DMAC to transfer data. RDRF is automatically cleared to 0 when the DMAC executes the data transfer to the receive data register (RDR). ERI is requested when the ORER, PER, or FER bit in the SSR is set to 1. ERI cannot start the DMAC.

TEI is requested when the TEND bit in the SSR is set to 1. TEI cannot start the DMAC. A TXI interrupt indicates that transmit data writing is enabled. A TEI interrupt indicates that the transmit operation is complete.

**Table 13.12 SCI Interrupt Sources**

| Interrupt Source | Description | DMAC Availability | Priority |
|---|---|---|---|
| ERI | Receive error (ORER, PER, or FER) | No | High |
| RXI | Receive data full (RDRF) | Yes | ↑ |
| TXI | Transmit data empty (TDRE) | Yes | ↓ |
| TEND | Transmit end (TEND) | No | Low |

## 13.5 Usage Notes

Note the following points when using the SCI.

TDR Write and TDRE Flags: The TDRE bit in the serial status register (SSR) is a status flag indicating loading of transmit data from the TDR into the TSR. The SCI sets TDRE to 1 when it transfers data from the TDR to the TSR. If new data is written in the TDR when TDRE is 0, the old data stored in the TDR will be lost because these data have not yet been transferred to the TSR. Before writing transmit data to the TDR, be sure to check that TDRE is set to 1.

Simultaneous Multiple Receive Errors: Table 13.13 indicates the state of the SSR status flags when multiple receive errors occur simultaneously. When an overrun error occurs, the RSR contents cannot be transferred to the RDR, so receive data is lost.

**Table 13.13 SSR Status Flags and Transfer of Receive Data**

| Receive Error Status | SSR Status Flags | | | | Receive Data Transfer |
|---|---|---|---|---|---|
| | RDRF | ORER | FER | PER | RSR → RDR |
| Overrun error | 1 | 1 | 0 | 0 | X |
| Framing error | 0 | 0 | 1 | 0 | O |
| Parity error | 0 | 0 | 0 | 1 | O |
| Overrun error + framing error | 1 | 1 | 1 | 0 | X |
| Overrun error + parity error | 1 | 1 | 0 | 1 | X |
| Framing error + parity error | 0 | 0 | 1 | 1 | O |
| Overrun error + framing error + parity error | 1 | 1 | 1 | 1 | X |

O: Receive data is transferred from RSR–RDR.

X: Receive data is not transferred from RSR–RDR.

**Break Detection and Processing:** Break signals can be detected by reading the RxD pin directly when a framing error (FER) is detected. In the break state, the input from the RxD pin consists of all 0s, so FER is set and the parity error flag (PER) may also be set. In the break state, the SCI receiver continues to operate, so if the FER bit is cleared to 0, it will be set to 1 again.

**Sending a Break Signal:** When TE is cleared to 0 the TxD pin becomes an I/O port, the level and direction (input or output) of which are determined by the data register (DR) of the I/O port and the control register (CR) of the PFC. This feature can be used to send a break signal. The DR value substitutes for the mark state until the PFC setting is performed. The DR bits should therefore be set as an output port that outputs 1 beforehand. To send a break signal during serial transmission, clear the DR bit to 0, and select output port as the TxD pin function by the PFC. When TE is cleared to 0, the transmitter is initialized, regardless of its current state.

**Receive Error Flags and Transmitter Operation (clocked synchronous mode only):** When a receive error flag (ORER, PER, or FER) is set to 1, the SCI will not start transmitting even if TDRE is set to 1. Be sure to clear the receive error flags to 0 before starting to transmit. Note that clearing RE to 0 does not clear the receive error flags.

**Receive Data Sampling Timing and Receive Margin in the Asynchronous Mode:** In the asynchronous mode, the SCI operates on a base clock of 16 times the bit rate frequency. In receiving, the SCI synchronizes internally with the falling edge of the start bit, which it samples on the base clock. Receive data is latched on the rising edge of the eighth base clock pulse. See figure 13.21.

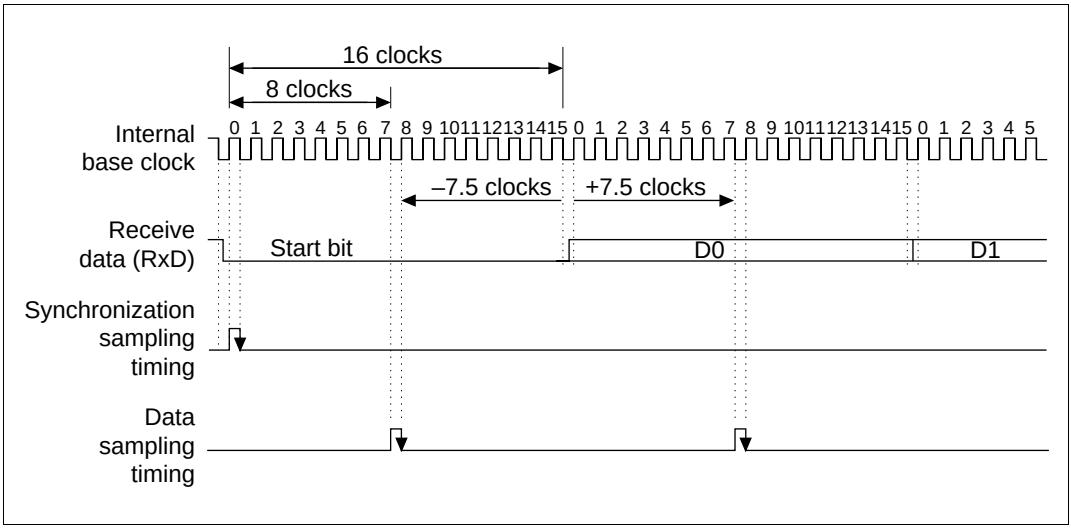



**Figure 13.21 Receive Data Sampling Timing in Asynchronous Mode**

The receive margin in the asynchronous mode can therefore be expressed as in equation 1.

Equation 1:

$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

M: Receive margin (%)
N: Ratio of clock frequency to bit rate (N = 16)
D: Clock duty cycle (D = 0–1.0)
L: Frame length (L = 9–12)
F: Absolute deviation of clock frequency

From equation (1), if F = 0 and D = 0.5 the receive margin is 46.875%, as given by equation 2.

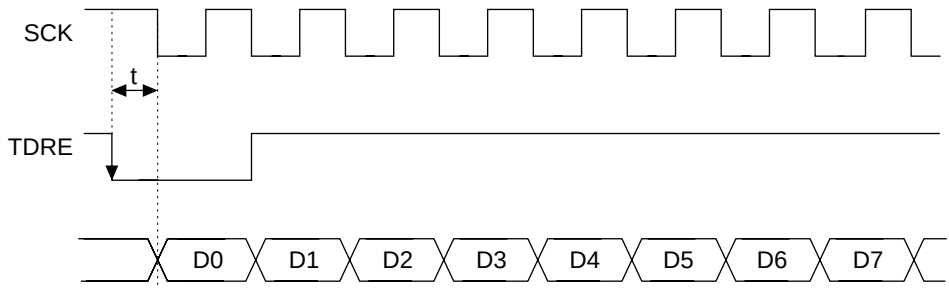
Equation 2:

$$D = 0.5, F = 0$$
$$M = (0.5 - 1/(2 \times 16)) \times 100\%$$
$$= 46.875\% \quad (2)$$

This is a theoretical value. A reasonable margin to allow in system designs is 20–30%.

**Constraints on DMAC Use:**

- When using an external clock source for the serial clock, update the TDR with the DMAC, and then after five system clocks or more elapse, input a transmit clock. If a transmit clock is input in the first four system clocks after the TDR is written, an error may occur (figure 13.22).
- Before reading the receive data register (RDR) with the DMAC, select the receive-data-full interrupt of the SCI as a start-up source using the resource select bit (RS) in the channel control register (CHCR).




Note: During external clock operation, an error may occur if t is 4$\phi$ or less.

**Figure 13.22 Clocked Synchronous Transmitting Example with DMAC**

**Cautions for Clocked Synchronous External Clock Mode:**

- Set TE = RE = 1 only when the external clock SCI is 1.
- Do not set TE = RE = 1 until at least 4 clocks after the external clock SCK has changed from 0 to 1.
- When receiving, RDRF = 1 when RE is set to zero 2.5–3.5 clocks after the rise edge of the RxD D7 bit SCK input, but it cannot be copied to RDR.

**Caution for Clocked Synchronous Internal Clock Mode:** When receiving, RDRF = 1 when RE is set to zero 1.5 clocks after the rise edge of the RxD D7 bit SCK input, but it cannot be copied to RDR.

# Section 14 Pin Function Controller (PFC)

## 14.1 Overview

The pin function controller (PFC) is composed of registers for selecting the function of multiplexed pins and the direction of input/output. The pin function and input/output direction can be selected for each pin individually without regard to the operating mode of the LSI. Table 14.1 lists the multiplexed pins.

**Table 14.1 List of Multiplexed Pins**

| Port | Function 1 (related module) | Function 2 (related module) | Function 3 (related module) | Function 4 (related module) | Pin # |
|---|---|---|---|---|---|
| A | PA15 I/O (port) | IRQ3 input (INTC) | DREQ1 input (DMAC) | — | 68 |
| A | PA14 I/O (port) | IRQ2 input (INTC) | DACK1 output (DMAC) | — | 67 |
| A | PA13 I/O (port) | IRQ1 input (INTC) | TCLKB input (ITU) | DREQ0 input (DMAC) | 66 |
| A | PA12 I/O (port) | IRQ0 input (INTC) | TCLKA input (ITU) | DACK0 output (DMAC) | 65 |
| A | PA11 I/O (port) | DPH I/O (D bus) | TIOCB1 I/O (ITU) | — | 64 |
| A | PA10 I/O (port) | DPL I/O (D bus) | TIOCA1 I/O (ITU) | — | 62 |
| A | PA9 I/O (port) | AH output (BSC) | — | IRQOUT output (INTC) | 61 |
| A | PA8 I/O (port) | BREQ input (system) | — | — | 60 |
| A | PA7 I/O (port) | BACK output (system) | — | — | 58 |
| A | PA6 I/O (port) | RD output (BSC) | — | — | 57 |
| A | PA5 I/O (port) | WRH output (BSC) (LBS output (BSC))*1 | — | — | 56 |
| A | PA4 I/O (port) | WRL output (BSC) (WR output (BSC))*1 | — | — | 55 |
| A | PA3 I/O (port) | CS7 output (BSC) | WAIT input (BSC) | — | 54 |
| A | PA2 I/O (port) | CS6 output (BSC) | TIOCB0 I/O (ITU) | — | 53 |
| A | PA1 I/O (port) | CS5 output (BSC) | RAS output (BSC) | — | 52 |
| A | PA0 I/O (port) | CS4 output (BSC) | TIOCA0 I/O (ITU) | — | 51 |

**Table 14.1 List of Multiplexed Pins (cont)**

| Port | Function 1 (related module) | Function 2 (related module) | Function 3 (related module) | Function 4 (related module) | Pin # |
|---|---|---|---|---|---|
| B | PB15 I/O (port) | IRQ7 input (INTC) | — | TP15 output (TPC) | 100 |
| B | PB14 I/O (port) | IRQ6 input (INTC) | — | TP14 output (TPC) | 99 |
| B | PB13 I/O (port) | IRQ5 input (INTC) | SCK1 I/O (SCI) | TP13 output (TPC) | 98 |
| B | PB12 I/O (port) | IRQ4 input (INTC) | SCK0 I/O (SCI) | TP12 output (TPC) | 97 |
| B | PB11 I/O (port) | TxD1 output (SCI) | TP11 output (TPC) | — | 96 |
| B | PB10 I/O (port) | RxD1 input (SCI) | TP10 output (TPC) | — | 95 |
| B | PB9 I/O (port) | TxD0 output (SCI) | TP9 output (TPC) | — | 94 |
| B | PB8 I/O (port) | RxD0 input (SCI) | TP8 output (TPC) | — | 93 |
| B | PB7 I/O (port) | TCLKD input (ITU) | TOCXB4 output (ITU) | TP7 output (TPC) | 91 |
| B | PB6 I/O (port) | TCLKC input (ITU) | TOCXA4 output (ITU) | TP6 output (TPC) | 90 |
| B | PB5 I/O (port) | TIOCB4 I/O (ITU) | TP5 output (TPC) | — | 89 |
| B | PB4 I/O (port) | TIOCA4 I/O (ITU) | TP4 output (TPC) | — | 87 |
| B | PB3 I/O (port) | TIOCB3 I/O (ITU) | TP3 output (TPC) | — | 86 |
| B | PB2 I/O (port) | TIOCA3 I/O (ITU) | TP2 output (TPC) | — | 85 |
| B | PB1 I/O (port) | TIOCB2 I/O (ITU) | TP1 output (TPC) | — | 84 |
| B | PB0 I/O (port) | TIOCA2 I/O (ITU) | TP0 output (TPC) | — | 83 |
| — | CS1 output (BSC) | CASH output (BSC) | — | — | 47 |
| — | CS3 output (BSC) | CASL output (BSC) | — | — | 49 |

INTC: Interrupt controller

DMAC: Direct memory access controller

ITU: 16-bit integrated timer pulse unit

D bus: Data bus control

BSC: Bus state controller

System: System control

SCI: Serial communications interface

TPC: Programmable timing pattern controller

Port: I/O port

Notes: 1. The bus control register of the bus state controller handles switching between the two functions.

## 14.2 Register Configuration

Table 14.2 summarizes the registers of the pin function controller.

**Table 14.2 Pin Function Controller Registers**

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|---|---|---|---|---|---|
| Port A I/O register | PAIOR | R/W | H'0000 | H'5FFFFC4 | 8, 16, 32 |
| Port A control register 1 | PACR1 | R/W | H'3302 | H'5FFFFC8 | 8, 16, 32 |
| Port A control register 2 | PACR2 | R/W | H'FF95 | H'5FFFFCA | 8, 16, 32 |
| Port B I/O register | PBIOR | R/W | H'0000 | H'5FFFFC6 | 8, 16, 32 |
| Port B control register 1 | PBCR1 | R/W | H'0000 | H'5FFFFCC | 8, 16, 32 |
| Port B control register 2 | PBCR2 | R/W | H'0000 | H'5FFFFCE | 8, 16, 32 |
| Column address strobe pin control register | CASCR | R/W | H'5FFF | H'5FFFFEE | 8, 16, 32 |

## 14.3 Register Descriptions

### 14.3.1 Port A I/O Register (PAIOR)

The port A I/O register (PAIOR) is a 16-bit read/write register that selects input or output for individual pins on a bit-by-bit basis. Bits PA15IOR–PA0IOR correspond to pins PA15/$\overline{\text{IRQ3}}$/$\overline{\text{DREQ1}}$–PA0/$\overline{\text{CS4}}$/TIOCA0. PAIOR is enabled when the port A pins function as input/outputs (PA15–PA0) and for ITU input capture and output compare (TIOCA1, TIOCA0, TIOCB1, and TIOCB0). For other functions, they are disabled. For port A pin functions PA15–PA0 and TIOCA1, TIOCA0, TIOCB1, and TIOCB0, a given pin in port A is an output pin if its corresponding PAIOR bit is set to 1, and an input pin if the bit is cleared to 0.

PAIOR is initialized to H'0000 by power-on resets; however, it is not initialized for manual resets, standby mode, or sleep mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PA15 IOR | PA14 IOR | PA13 IOR | PA12 IOR | PA11 IOR | PA10 IOR | PA9 IOR | PA8 IOR |
| 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: | PA7 IOR | PA6 IOR | PA5 IOR | PA4 IOR | PA3 IOR | PA2 IOR | PA1 IOR | PA0 IOR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

### 14.3.2 Port A Control Registers (PACR1 and PACR2)

PACR1 and PACR2 are 16-bit read/write registers that select the functions of the sixteen multiplexed pins of port A. PACR1 selects the function of the top eight bits of port A; PACR2 selects the function of the bottom eight bits of port A. PACR1 and PACR2 are initialized to H'3302 and H'FF95 respectively by power-on resets but are not initialized for manual resets, standby mode, or sleep mode.

**PACR1:**

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PA15 MD1 | PA15 MD0 | PA14 MD1 | PA14 MD0 | PA13 MD1 | PA13 MD0 | PA12 MD1 | PA12 MD0 |
| Initial value: | 0 | 0 | 1 | 1 | 0 | 0 | 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: | PA11 MD1 | PA11 MD0 | PA10 MD1 | PA10 MD0 | PA9MD1 | PA9MD0 | — | PA8MD |
| 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 |

- Bits 15 and 14 (PA15 mode (PA15MD1 and PA15MD0)): PA15MD1 and PA15MD0 select the function of the PA15/$\overline{\text{IRQ3}}$/$\overline{\text{DREQ1}}$ pin.

| Bit 15: PA15MD1 | Bit 14: PA15MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA15) (initial value) |
| | 1 | Interrupt request input (IRQ3) |
| 1 | 0 | Reserved |
| | 1 | DMA transfer request input (DREQ1) |

- Bits 13 and 12 (PA14 mode (PA14MD1 and PA14MD0)): PA14MD1 and PA14MD0 select the function of the PA14/$\overline{\text{IRQ2}}$/DACK1 pin.

| Bit 13: PA14MD1 | Bit 12: PA14MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA14) |
| | 1 | Interrupt request input (IRQ2) |
| 1 | 0 | Reserved |
| | 1 | DMA transfer acknowledge output (DACK1) (initial value) |

- Bits 11 and 10 (PA13 Mode (PA13MD1 and PA13MD0)): PA13MD1 and PA13MD0 select the function of the PA13/$\overline{\text{IRQ1}}$/$\overline{\text{DREQ0}}$/TCLKB pin.

| Bit 11: PA13MD1 | Bit 10: PA13MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA13) (initial value) |
| | 1 | Interrupt request input (IRQ1) |
| 1 | 0 | ITU timer clock input (TCLKB) |
| | 1 | DMA transfer request input (DREQ0) |

- Bits 9 and 8 (PA12 mode (PA12MD1 and PA12MD0)): These bits select the function of the PA12/$\overline{\text{IRQ0}}$/DACK0/TCLKA pin.

| Bit 9: PA12MD1 | Bit 8: PA12MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA12) |
| | 1 | Interrupt request input (IRQ0) |
| 1 | 0 | ITU timer clock input (TCLKA) |
| | 1 | DMA transfer acknowledge output (DACK0) (initial value) |

- Bits 7 and 6 (PA11 mode (PA11MD1 and PA11MD0)): These bits select the function of the PA11/DPH/TIOCB1 pin.

| Bit 7: PA11MD1 | Bit 6: PA11MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA11) (initial value) |
| | 1 | Upper data bus parity input/output (DPH) |
| 1 | 0 | ITU input capture/output compare (TIOCB1) |
| | 1 | Reserved |

- Bits 5 and 4 (PA10 mode (PA10MD1 and PA10MD0)): These bits select the function of the PA10/DPL/TIOCA1 pin.

| Bit 5: PA10MD1 | Bit 4: PA10MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA10) (initial value) |
| | 1 | Lower data bus parity input/output (DPL) |
| 1 | 0 | ITU input capture/output compare (TIOCA1) |
| | 1 | Reserved |

- Bits 3 and 2 (PA9 mode (PA9MD1 and PA9MD0)): These bits select the function of the PA9/$\overline{\text{AH}}$/$\overline{\text{IRQOUT}}$ pin.

| Bit 3: PA9MD1 | Bit 2: PA9MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA9) (initial value) |
| | 1 | Address hold output ($\overline{\text{AH}}$) |
| 1 | 0 | Reserved |
| | 1 | Interrupt request output ($\overline{\text{IRQOUT}}$) |

- Bit 1 (reserved): This bit always reads as 1. The write value should always be 1.
- Bit 0 (PA8 mode (PA8MD)): PA8MD selects the function of the PA8/$\overline{\text{BREQ}}$.

| Bit 0: PA8MD | Function |
|---|---|
| 0 | Input/output (PA8) (initial value) |
| 1 | Bus request input ($\overline{\text{BREQ}}$) |

**PACR2:**

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | — | PA7MD | — | PA6MD | — | PA5MD | — | PA4MD |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | — | R/W | — | R/W | — | R/W | — | R/W |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PA3MD1 | PA3MD0 | PA2MD1 | PA2MD0 | PA1MD1 | PA1MD0 | PA0MD1 | PA0MD0 |
| Initial value: | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bit 15 (reserved): This bit always reads as 1. The write value should always be 1.
- Bit 14 (PA7 mode (PA7MD)): PA7MD selects the function of the PA7/$\overline{\text{BACK}}$ pin.

| Bit 14: PA7MD | Function |
|---|---|
| 0 | Input/output (PA7) |
| 1 | Bus request acknowledge output ($\overline{\text{BACK}}$) (initial value) |

- Bit 13 (reserved): This bit always reads as 1. The write value should always be 1.
- Bit 12 (PA6 mode (PA6MD)): PA6MD selects the function of the PA6/$\overline{\text{RD}}$ pin.

| Bit 12: PA6MD | Function |
|---|---|
| 0 | Input/output (PA6) |
| 1 | Read output ($\overline{\text{RD}}$) (initial value) |

- Bit 11 (reserved): This bit always reads as 1. The write value should always be 1.
- Bit 10 (PA5 mode (PA5MD)): PA5MD selects the function of the PA5/$\overline{\text{WRH}}$ ($\overline{\text{LBS}}$) pin.

| Bit 10: PA5MD | Function |
|---|---|
| 0 | Input/output (PA5) |
| 1 | Upper write output ($\overline{\text{WRH}}$) or lower byte strobe output ($\overline{\text{LBS}}$) (initial value) |

- Bit 9 (reserved): This bit always reads as 1. The write value should always be 1.

- Bit 8 (PA4 mode (PA4MD)): PA4MD selects the function of the PA4/$\overline{\text{WRL}}$ ($\overline{\text{WR}}$) pin.

| Bit 8: PA4MD | Function |
|---|---|
| 0 | Input/output (PA4) |
| 1 | Lower write output ($\overline{\text{WRL}}$) or write output ($\overline{\text{WR}}$) (initial value) |

- Bits 7 and 6 (PA3 mode (PA3MD1 and PA3MD0)): PA3MD1 and PA3MD0 select the function of the PA3/$\overline{\text{CS7}}$/$\overline{\text{WAIT}}$ pin. This pin has a pull-up MOS that is used when it functions as a $\overline{\text{WAIT}}$ pin to allow selection of pull up or no pull up (for the $\overline{\text{WAIT}}$ pin) using the wait state control register of the bus state controller (BSC). There is no pull up when if functions as PA3 or CS7.

| Bit 7: PA3MD1 | Bit 6: PA3MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA3) |
| | 1 | Chip select output ($\overline{\text{CS7}}$) |
| 1 | 0 | Wait state input ($\overline{\text{WAIT}}$) (initial value) |
| | 1 | Reserved |

- Bits 5 and 4 (PA2 mode (PA2MD1 and PA2MD0)): PA2MD1 and PA2MD0 select the function of the PA2/$\overline{\text{CS6}}$/TIOCB0 pin.

| Bit 5: PA2MD1 | Bit 4: PA2MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA2) |
| | 1 | Chip select output ($\overline{\text{CS6}}$) (initial value) |
| 1 | 0 | ITU input capture/output compare (TIOCB0) |
| | 1 | Reserved |

- Bits 3 and 2 (PA1 mode (PA1MD1 and PA1MD0)): PA1MD1 and PA1MD0 select the function of the PA1/$\overline{\text{CS5}}$/$\overline{\text{RAS}}$ pin.

| Bit 3: PA1MD1 | Bit 2: PA1MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA1) |
| | 1 | Chip select output ($\overline{\text{CS5}}$) (initial value) |
| 1 | 0 | Row address strobe output ($\overline{\text{RAS}}$) |
| | 1 | Reserved |

- Bits 1 and 0 (PA0 mode (PA0MD1 and PA0MD0)): PA0MD1 and PA0MD0 select the function of the PA0/$\overline{CS4}$/TIOCA0 pin.

| Bit 1: PA0MD1 | Bit 0: PA0MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PA0) |
| | 1 | Chip select output ($\overline{CS4}$) (initial value) |
| 1 | 0 | ITU input capture/output compare (TIOCA0) |
| | 1 | Reserved |

### 14.3.3 Port B I/O Register (PBIOR)

The port B I/O register (PBIOR) is a 16-bit read/write register that selects input or output for individual pins on a bit-by-bit basis. Bits PB15IOR–PB0IOR correspond to pins of port B. PBIOR is enabled when the port B pins function as input/outputs (PB15–PB0), for ITU input capture and output compare (TIOCA4, TIOCA3, TIOCA2, TIOCB4, TIOCB3, and TIOCB2), and as serial clocks (SCK1, SCK0). For other functions, they are disabled. For port B pin functions PB15–PB0, and TIOCA4, TIOCA3, TIOCA2, TIOCB4, TIOCB3, and TIOCB2, and SCK1/SCK0, a given pin in port B is an output pin if its corresponding PBIOR bit is set to 1, and an input pin if the bit is cleared to 0.

PBIOR is initialized to H'0000 by power-on resets; however, it is not initialized for manual resets, standby mode, or sleep mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB15 IOR | PB14 IOR | PB13 IOR | PB12 IOR | PB11 IOR | PB10 IOR | PB9 IOR | PB8 IOR |
| 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: | PB7 IOR | PB6 IOR | PB5 IOR | PB4 IOR | PB3 IOR | PB2 IOR | PB1 IOR | PB0 IOR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

### 14.3.4 Port B Control Registers (PBCR1 and PBCR2)

PBCR1 and PBCR2 are 16-bit read/write registers that select the functions of the sixteen multiplexed pins of port B. PBCR1 selects the function of the top eight bits of port B; PBCR2 selects the function of the bottom eight bits of port B. PBCR1 and PBCR2 are initialized to H'0000 by power-on resets but are not initialized for manual resets, standby mode, or sleep mode.

**PBCR1:**

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB15 MD1 | PB15 MD0 | PB14 MD1 | PB14 MD0 | PB13 MD1 | PB13 MD0 | PB12 MD1 | PB12 MD0 |
| 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: | PB11 MD1 | PB11 MD0 | PB10 MD1 | PB10 MD0 | PB9 MD1 | PB9 MD0 | PB8 MD1 | PB8 MD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bits 15 and 14 (PB15 mode (PB15MD1 and PB15MD0)): PB15MD1 and PB15MD0 select the function of the PB15/TP15/$\overline{\text{IRQ7}}$ pin.

| Bit 15: PB15MD1 | Bit 14: PB15MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB15) (initial value) |
| | 1 | Interrupt request input (IRQ7) |
| 1 | 0 | Reserved |
| | 1 | Timing pattern output (TP15) |

- Bits 13 and 12 (PB14 mode (PB14MD1 and PB14MD0)): PB14MD1 and PB14MD0 select the function of the PB14/TP14/$\overline{\text{IRQ6}}$ pin.

| Bit 13: PB14MD1 | Bit 12: PB14MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB14) (initial value) |
| | 1 | Interrupt request input (IRQ6) |
| 1 | 0 | Reserved |
| | 1 | Timing pattern output (TP14) |

- Bits 11 and 10 (PB13 mode (PB13MD1 and PB13MD0)): PB13MD1 and PB13MD0 select the function of the PB13/TP13/$\overline{\text{IRQ5}}$/SCK1 pin.

| Bit 11: PB13MD1 | Bit 10: PB13MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB13) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ5}}$) |
| 1 | 0 | Serial clock input/output (SCK1) |
| | 1 | Timing pattern output (TP13) |

- Bits 9 and 8 (PB12 mode (PB12MD1 and PB12MD0)): PB12MD1 and PB12MD0 select the function of the PB12/TP12/$\overline{\text{IRQ4}}$/SCK0 pin.

| Bit 9: PB12MD1 | Bit 8: PB12MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB12) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ4}}$) |
| 1 | 0 | Serial clock input/output (SCK0) |
| | 1 | Timing pattern output (TP12) |

- Bits 7 and 6: PB11 mode (PB11MD1 and PB11MD0): PB11MD1 and PB11MD0 select the function of the PB11/TP11/TxD1 pin.

| Bit 7: PB11MD1 | Bit 6: PB11MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB11) (initial value) |
| | 1 | Reserved |
| 1 | 0 | Transmit data output (TxD1) |
| | 1 | Timing pattern output (TP11) |

- Bits 5 and 4 (PB10 mode (PB10MD1 and PB10MD0): PB10MD1 and PB10MD0 select the function of the PB10/TP10/RxD1 pin.

| Bit 5: PB10MD1 | Bit 4: PB10MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB10) (initial value) |
| | 1 | Reserved |
| 1 | 0 | Receive data input (RxD1) |
| | 1 | Timing pattern output (TP10) |

- Bits 3 and 2 (PB9 mode (PB9MD1 and PB9MD0)): PB9MD1 and PB9MD0 select the function of the PB9/TP9/TxD0 pin.

| Bit 3: PB9MD1 | Bit 2: PB9MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB9) (initial value) |
| | 1 | Reserved |
| 1 | 0 | Transmit data output (TxD0) |
| | 1 | Timing pattern output (TP9) |

- Bits 1 and 0 (PB8 mode (PB8MD1 and PB8MD0)): PB8MD1 and PB8MD0 select the function of the PB8/TP8/RxD0 pin.

| Bit 1: PB8MD1 | Bit 0: PB8MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB8) (initial value) |
| | 1 | Reserved |
| 1 | 0 | Receive data input (RxD0) |
| | 1 | Timing pattern output (TP8) |

**PBCR2:**

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB7MD1 | PB7MD0 | PB6MD1 | PB6MD0 | PB5MD1 | PB5MD0 | PB4MD1 | PB4MD0 |
| 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: | PB3MD1 | PB3MD0 | PB2MD1 | PB2MD0 | PB1MD1 | PB1MD0 | PB0MD1 | PB0MD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bits 15 and 14 (PB7 mode (PB7MD1 and PB7MD0)): PB7MD1 and PB7MD0 select the function of the PB7/TP7/TOCXB4/TCLKD pin.

| Bit 15: PB7MD1 | Bit 14: PB7MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB7) (initial value) |
| | 1 | ITU timer clock input (TCLKD) |
| 1 | 0 | ITU output compare (TOCXB4) |
| | 1 | Timing pattern output (TP7) |

- Bits 13 and 12 (PB6 mode (PB6MD1 and PB6MD0)): PB6MD1 and PB6MD0 select the function of the PB6/TP6/TOCXA4/TCLKC pin.

| Bit 13: PB6MD1 | Bit 12: PB6MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB6) (initial value) |
| | 1 | ITU timer clock input (TCLKC) |
| 1 | 0 | ITU output compare (TOCXA4) |
| | 1 | Timing pattern output (TP6) |

- Bits 11 and 10 (PB5 mode (PB5MD1 and PB5MD0)): PB5MD1 and PB5MD0 select the function of the PB5/TP5/TIOCB4 pin.

| Bit 11: PB5MD1 | Bit 10: PB5MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB5) (initial value) |
| | 1 | Reserved |
| 1 | 0 | ITU input capture/output compare (TIOCB4) |
| | 1 | Timing pattern output (TP5) |

- Bits 9 and 8 (PB4 mode (PB4MD1 and PB4MD0)): PB4MD1 and PB4MD0 select the function of the PB4/TP4/TIOCA4 pin.

| Bit 9: PB4MD1 | Bit 8: PB4MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB4) (initial value) |
| | 1 | Reserved |
| 1 | 0 | ITU input capture/output compare (TIOCA4) |
| | 1 | Timing pattern output (TP4) |

- Bits 7 and 6 (PB3 mode (PB3MD1 and PB3MD0)): PB3MD1 and PB3MD0 select the function of the PB3/TP3/TIOCB3 pin.

| Bit 7: PB3MD1 | Bit 6: PB3MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB3) (initial value) |
| | 1 | Reserved |
| 1 | 0 | ITU input capture/output compare (TIOCB3) |
| | 1 | Timing pattern output (TP3) |

- Bits 5 and 4 (PB2 mode (PB2MD1 and PB2MD0)): PB2MD1 and PB2MD0 select the function of the PB2/TP2/TIOCA3 pin.

| Bit 5: PB2MD1 | Bit 4: PB2MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB2) (initial value) |
| | 1 | Reserved |
| 1 | 0 | ITU input capture/output compare (TIOCA3) |
| | 1 | Timing pattern output (TP2) |

- Bits 3 and 2 (PB1 mode (PB1MD1 and PB1MD0)): PB1MD1 and PB1MD0 select the function of the PB1/TP1/TIOCB2 pin.

| Bit 3: PB1MD1 | Bit 2: PB1MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB1) (initial value) |
| | 1 | Reserved |
| 1 | 0 | ITU input capture/output compare (TIOCB2) |
| | 1 | Timing pattern output (TP1) |

- Bits 1 and 0 (PB0 mode (PB0MD1 and PB0MD0)): PB0MD1 and PB0MD0 select the function of the PB0/TP0/TIOCA2 pin.

| Bit 1: PB0MD1 | Bit 0: PB0MD0 | Function |
|---|---|---|
| 0 | 0 | Input/output (PB0) (initial value) |
| | 1 | Reserved |
| 1 | 0 | ITU input capture/output compare (TIOCA2) |
| | 1 | Timing pattern output (TP0) |

### 14.3.5 Column Address Strobe Pin Control Register (CASCR)

CASCR is a 16-bit read/write register that allows selection between column address strobe and chip select pin functions. The CASCR is initialized to H'5FFF by power-on resets but is not initialized for manual resets, standby mode, or sleep mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | CASH MD1 | CASH MD0 | CASL MD1 | CASL MD0 | — | — | — | — |
| Initial value: | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 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: | — | — | — | — | — | — | — | — |

- Bits 15 and 14 (CASH mode (CASHMD1 and CASHMD0)): CASHMD1 and CASHMD0 select the function of the $\overline{\text{CS1}}/\overline{\text{CASH}}$ pin.

| Bit 15: CASHMD1 | Bit 14: CASHMD0 | Function |
|---|---|---|
| 0 | 0 | Reserved |
| | 1 | Chip select output ($\overline{\text{CS1}}$) (initial value) |
| 1 | 0 | Column address strobe output ($\overline{\text{CASH}}$) |
| | 1 | Reserved |

- Bits 13 and 12 (CASL mode (CASLMD1 and CASLMD0)): CASLMD1 and CASLMD0 select the function of the $\overline{\text{CS3}}/\overline{\text{CASL}}$ pin.

| Bit 13: CASLMD1 | Bit 12: CASLMD0 | Function |
|---|---|---|
| 0 | 0 | Reserved |
| | 1 | Chip select output ($\overline{\text{CS3}}$) (initial value) |
| 1 | 0 | Column address strobe output ($\overline{\text{CASL}}$) |
| | 1 | Reserved |

- Bits 11–0 (reserved): This bit always reads as 1. The write value should always be 1.

## 15.1 Overview

There are two ports, A and B. Ports A and B are 16-bit I/O ports. The pins of the ports are all multiplexed for use as general-purpose I/Os or for other functions. (Use the pin function controller (PFC) to select the function of multiplexed pins.) Ports A and B each have one data register for storing pin data.

## 15.2 Port A

Port A is a 16-pin input/output port, as shown in figure 15.1. The PA3/$\overline{CS7}$/$\overline{WAIT}$ pin of port A has a pull-up MOS so that when it is functioning as a $\overline{WAIT}$ pin, the wait state control register of the bus state controller can be used to select whether to pull up the $\overline{WAIT}$ pin or not. It is not pulled up when the pin is functioning as either PA3 or $\overline{CS7}$.

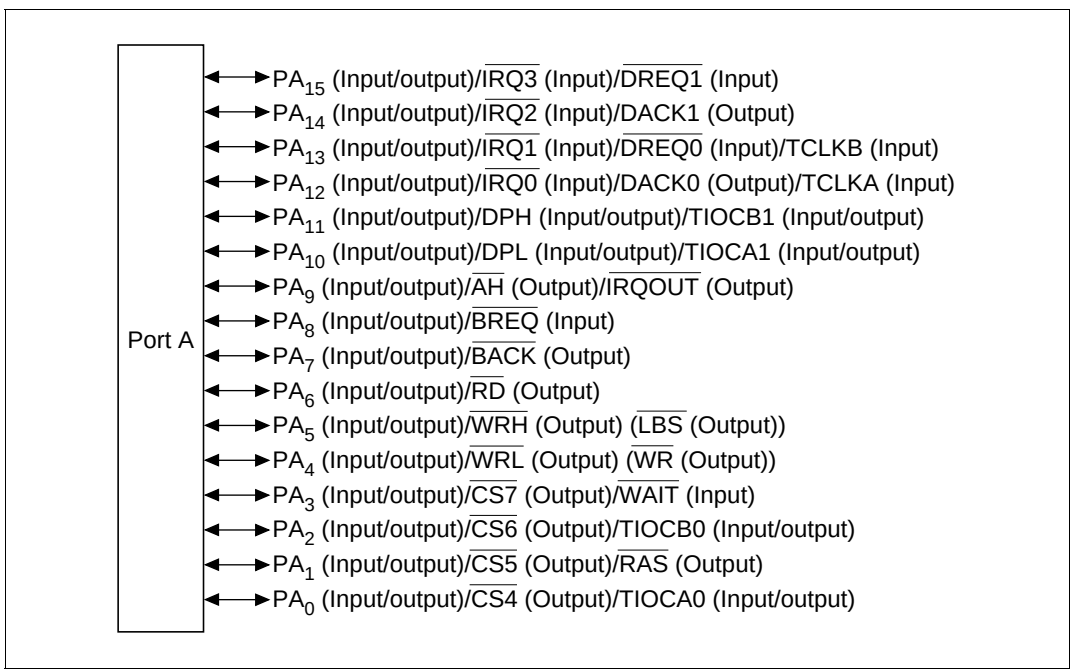



**Figure 15.1 Port A Configuration**

### 15.2.1 Register Configuration

Table 15.1 summarizes the port A register.

**Table 15.1 Port A Register**

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|---|---|---|---|---|---|
| Port A data register | PADR | R/W | H'0000 | H'5FFFFC0 | 8, 16, 32 |

### 15.2.2 Port A Data Register (PADR)

PADR is a 16-bit read/write register that stores data for port A. The bits PA15DR–PA0DR correspond to the PA15/$\overline{\text{IRQ3}}$/$\overline{\text{DREQ1}}$–PA0/$\overline{\text{CS4}}$/TIOCA0 pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PADR; when PADR is read, the register value will be output regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PADR is read. When a value is written to PADR, that value can be written into PADR, but it will not affect the pin status. Table 15.2 shows the read/write operations of the port A data register.

PADR is initialized by a power-on reset. However, PADR is not initialized for manual reset, standby mode, or sleep mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PA15DR | PA14DR | PA13DR | PA12DR | PA11DR | PA10DR | PA9DR | PA8DR |
| 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: | PA7DR | PA6DR | PA5DR | PA4DR | PA3DR | PA2DR | PA1DR | PA0DR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Table 15.2 Read/Write Operation of the Port A Data Register (PADR)**

| PAIOR | Pin Status | Read | Write |
|---|---|---|---|
| 0 | Input | Pin status | Can write to PADR, but it has no effect on pin status. |
| | Other function | Pin status | Can write to PADR, but it has no effect on pin status. |
| 1 | Output | PADR value | Value written is output by pin |
| | Other function | PADR value | Can write to PADR, but it has no effect on pin status. |

## 15.3 Port B

Port B is a 16-bit input/output port as shown in figure 16.2.

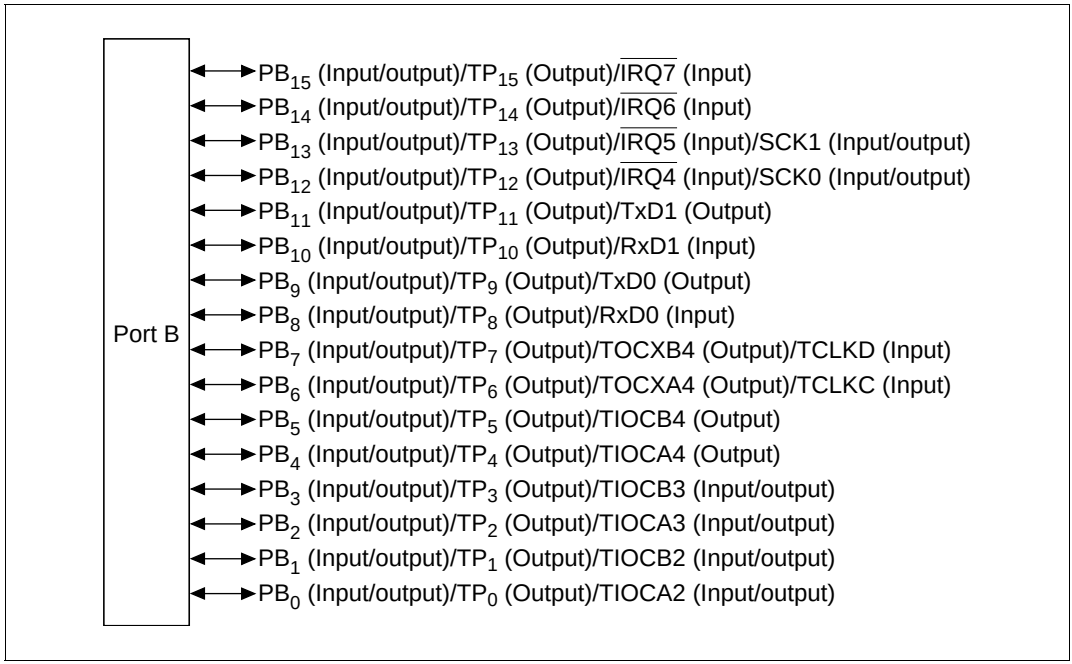



**Figure 15.2 Port B Configuration**

### 15.3.1 Register Configuration

Table 15.3 summarizes the port B register.

**Table 15.3 Port B Register**

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|---|---|---|---|---|---|
| Port B data register | PBDR | R/W | H'0000 | H'5FFFFC2 | 8, 16, 32 |

### 15.3.2 Port B Data Register (PBDR)

PBDR is a 16-bit read/write register that stores data for port B. The bits PB15DR–PB0DR correspond to the PB15/TP15/$\overline{\text{IRQ7}}$–PB0/TP0/TIOCA2 pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PBDR; when PBDR is read, the register value will be output regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PBDR is read. When a value is written to PBDR, that value can be written into PBDR, but it will not affect the pin status. When the pin function is set to timing pattern output and the TPC output is enabled by the TPC next data enable register (NDER), no value can be written to PBDR. Table 15.4 shows the read/write operations of the port B data register.

PBDR is initialized by a power-on reset. However, PBDR is not initialized for a manual reset, standby mode, or sleep mode.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | PB15DR | PB14DR | PB13DR | PB12DR | PB11DR | PB10DR | PB9DR | PB8DR |
| 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: | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Table 15.4 Read/Write Operation of the Port B Data Register (PBDR)**

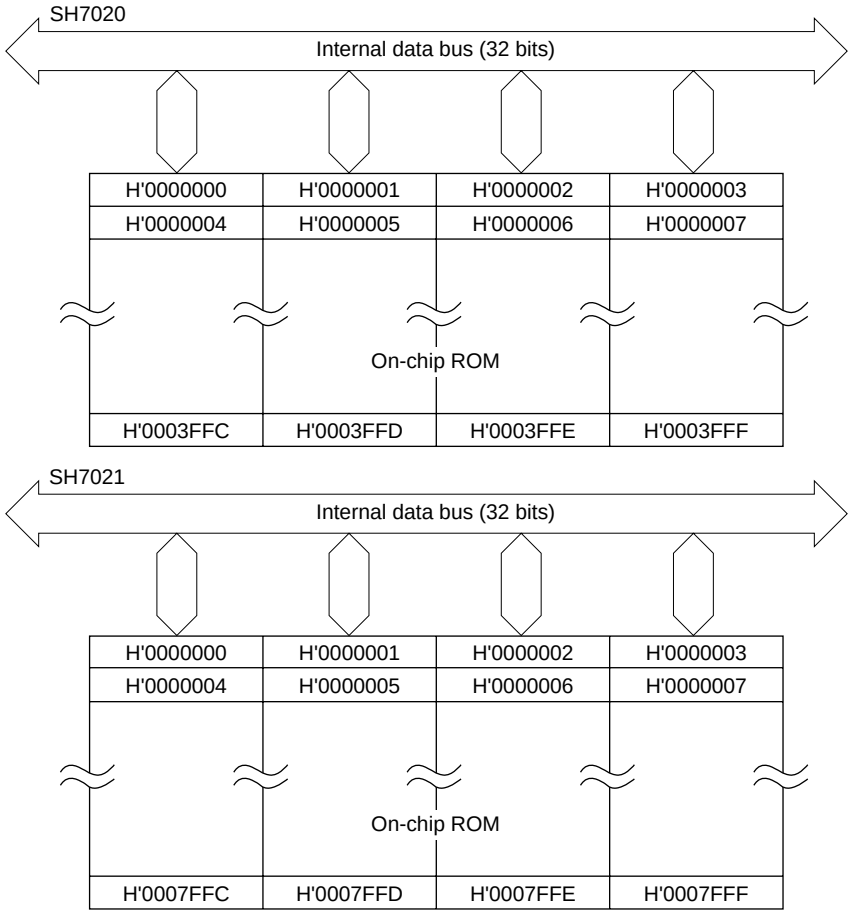
| PBIOR | Pin Status | Read | Write |
|---|---|---|---|
| 0 | Input | Pin status | Can write to PBDR, but it has no effect on pin status |
| | TPn | Pin status | Disabled |
| | Other function | Pin status | Can write to PBDR, but it has no effect on pin status |
| 1 | Output | PBDR value | Value written is output by pin |
| | TPn | PBDR value | Disabled |
| | Other function | PBDR value | Can write to PBDR, but it has no effect on pin status |

TPn: Timing pattern output

# Section 16 ROM

## 16.1 Overview

The SH7020 microcomputer has 16 kbytes of on-chip ROM (mask ROM). The SH7021 microcomputer has 32 kbytes of on-chip ROM (mask ROM or PROM). The on-chip ROM is connected to the CPU and the direct memory access controller (DMAC) through a 32-bit data bus (figure 16.1). The CPU can access the on-chip ROM in 8-, 16- and 32-bit widths and the DMAC can access the ROM in 8- and 16-bit widths. Data in the on-chip ROM can always be accessed in one cycle.




Note: The addresses shown in the figure are the first shadow addresses in the on-chip ROM space.

**Figure 16.1 Block Diagram of ROM**

The operating mode determines whether the on-chip ROM is valid or not. The operating mode is selected using mode-setting pins MD0-MD2 as shown in table17.1. If you are using the on-chip ROM, select mode 2; if you are not, select mode 0 or 1. The on-chip ROM is allocated to address H'0000000–H'0003FFF (SH7020), H'0000000–H'0007FFF (SH7021) of memory area 0. Memory area 0 (H'0000000-H'0FFFFFF and H'8000000–H'8FFFFFF) is divided into 16-kbyte (SH7020) or 32-kbyte (SH7021) shadows. No matter which shadow is accessed, the on-chip ROM is accessed. See section 8, Bus State Controller, for more information on shadows.

**Table 16.1 Operating Modes and ROM**

| | Mode Setting Pin | | | |
|---|---|---|---|---|
| **Operating Mode** | **MD2** | **MD1** | **MD0** | **Area 0** |
| Mode 0 (MCU mode 0) | 0 | 0 | 0 | On-chip ROM invalid, external 8-bit space |
| Mode 1 (MCU mode 1) | 0 | 0 | 1 | On-chip ROM invalid, external 16-bit space |
| Mode 2 (MCU mode 2) | 0 | 1 | 0 | On-chip ROM valid |
| Mode 7 (PROM mode) | 1 | 1 | 1 | — |

0: Low
1: High

When the SH7021 is set to PROM mode, the PROM version can write programs exactly like ordinary EPROM using a general purpose EPROM writer.

## 16.2 PROM Mode

### 16.2.1 Setting the PROM Mode

To program the on-chip PROM, set the pins as shown in figure 16.2 and use the chip in PROM mode.

### 16.2.2 Socket Adapter Pin Correspondence and Memory Map

Mount the socket adapter to the SH7021 as shown in figure 16.2. This allows the on-chip PROM to be programmed in exactly the same way as ordinary 32-pin EPROMs (HN27C101). Figure 16.2 shows the correspondence of SH7021 pins and HN27C101 pins. Figure 16.3 shows the memory map of the on-chip ROM.

The address range of the HN27C101 (128 kbytes) is H'00000–H'1FFFF. The on-chip PROM (34 kbytes) is not found in H'08000–H'1FFFF.

When programming with a PROM writer, the program address range must be set to H'00000–H'07FFF. The data for the H'08000–H'1FFFF address area should all be H'FF. Set byte mode, not page mode.

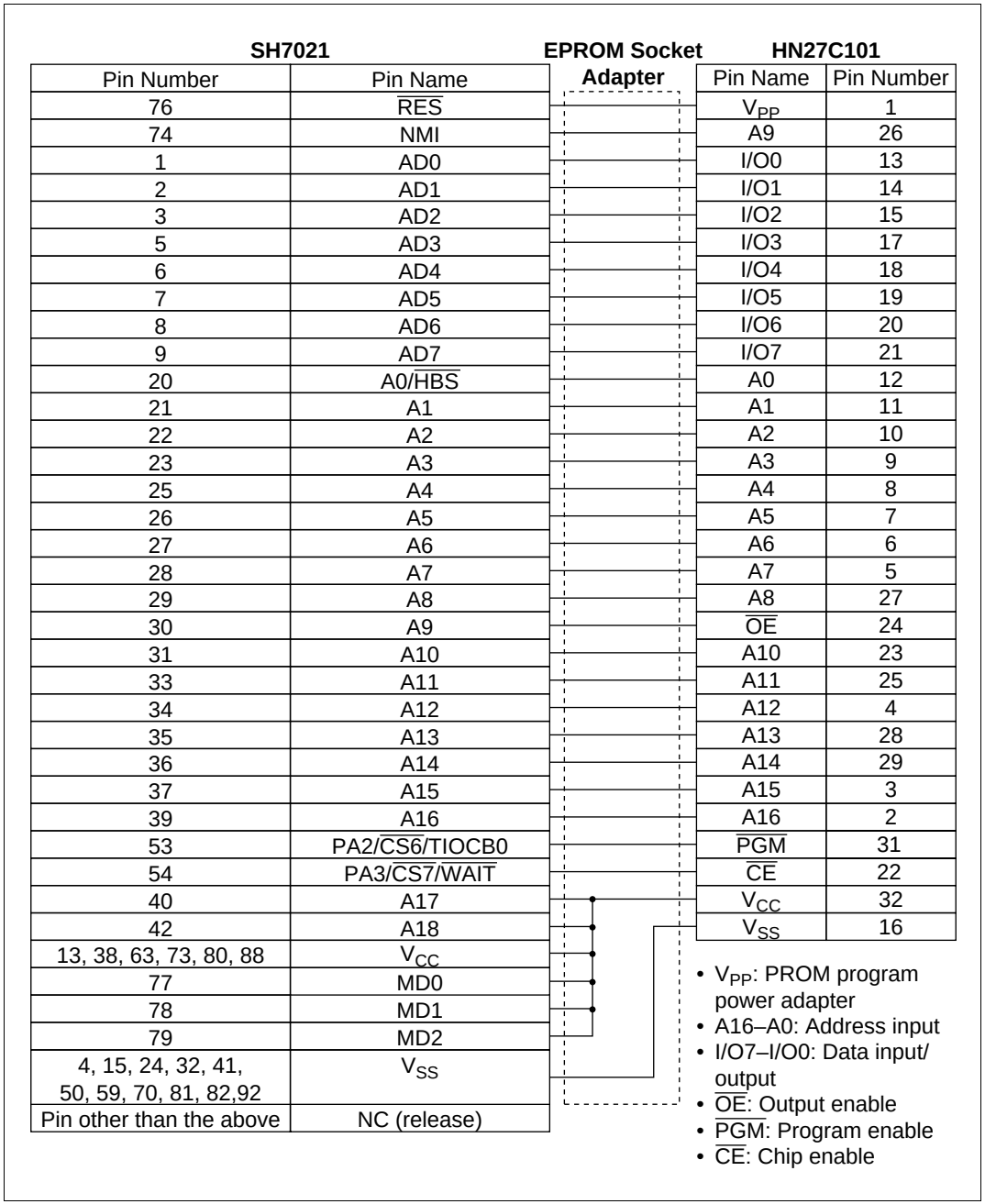
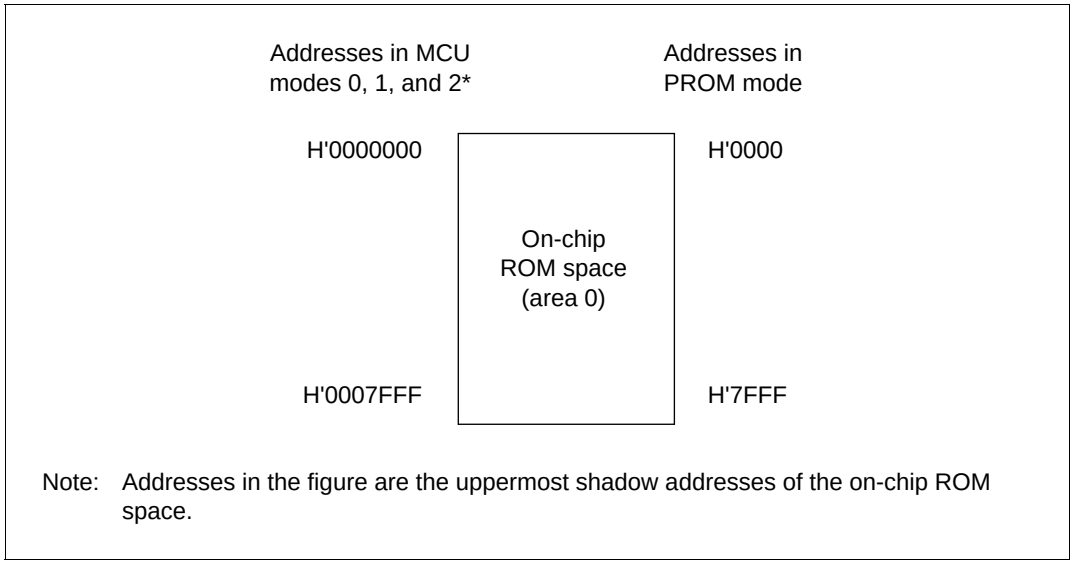



**Figure 16.2 Correspondence Between SH7021 Pins and HN27C101 Pins**




Note: Addresses in the figure are the uppermost shadow addresses of the on-chip ROM space.

**Figure 16.3 Memory Map of On-chip ROM**

## 16.3 PROM Programming

The write/verify specifications in PROM mode are the same as for the standard EPROM HN27C101. The page program system is not supported, so *do not set the PROM writer to the page programming mode*. Naturally, PROM writers that only support the page programming mode cannot be used. When selecting a PROM writer, check that the high-speed, high-reliability programming system for each byte is supported.

### 16.3.1 Selecting the Programming Mode

There are two on-chip PROM programming modes: write and verify (which reads and confirms the data written). Use the pins to select the modes (table 16.2).

**Table 16.2 Select PROM Programming Mode**

| | Pin | | | | | | |
|---|---|---|---|---|---|---|---|
| **Mode** | **CE** | **OE** | **PGM** | **$V_{PP}$** | **$V_{CC}$** | **I/O7–I/O0** | **A16–A0** |
| Write | 0 | 1 | 0 | $V_{PP}$ | $V_{CC}$ | Data input | Address input |
| Verify | 0 | 0 | 1 | | | Data output | |
| Program inhibit | 0 | 0 | 0 | | | High impedance | |
| | 0 | 1 | 1 | | | | |
| | 1 | 0 | 0 | | | | |
| | 1 | 1 | 1 | | | | |

Symbols:
0: Low
1: High
$V_{PP}$: $V_{PP}$ level
$V_{CC}$: $V_{CC}$ level

### 16.3.2 Write/Verify and Electrical Characteristics

**Write/Verify:** Write/verify can be accomplished by an efficient high-speed high-reliability programming system. This system can write data quickly and accurately without placing voltage stress on the device. The basic flowchart for this high-speed, high-reliability programming system is shown in figure 16.4.

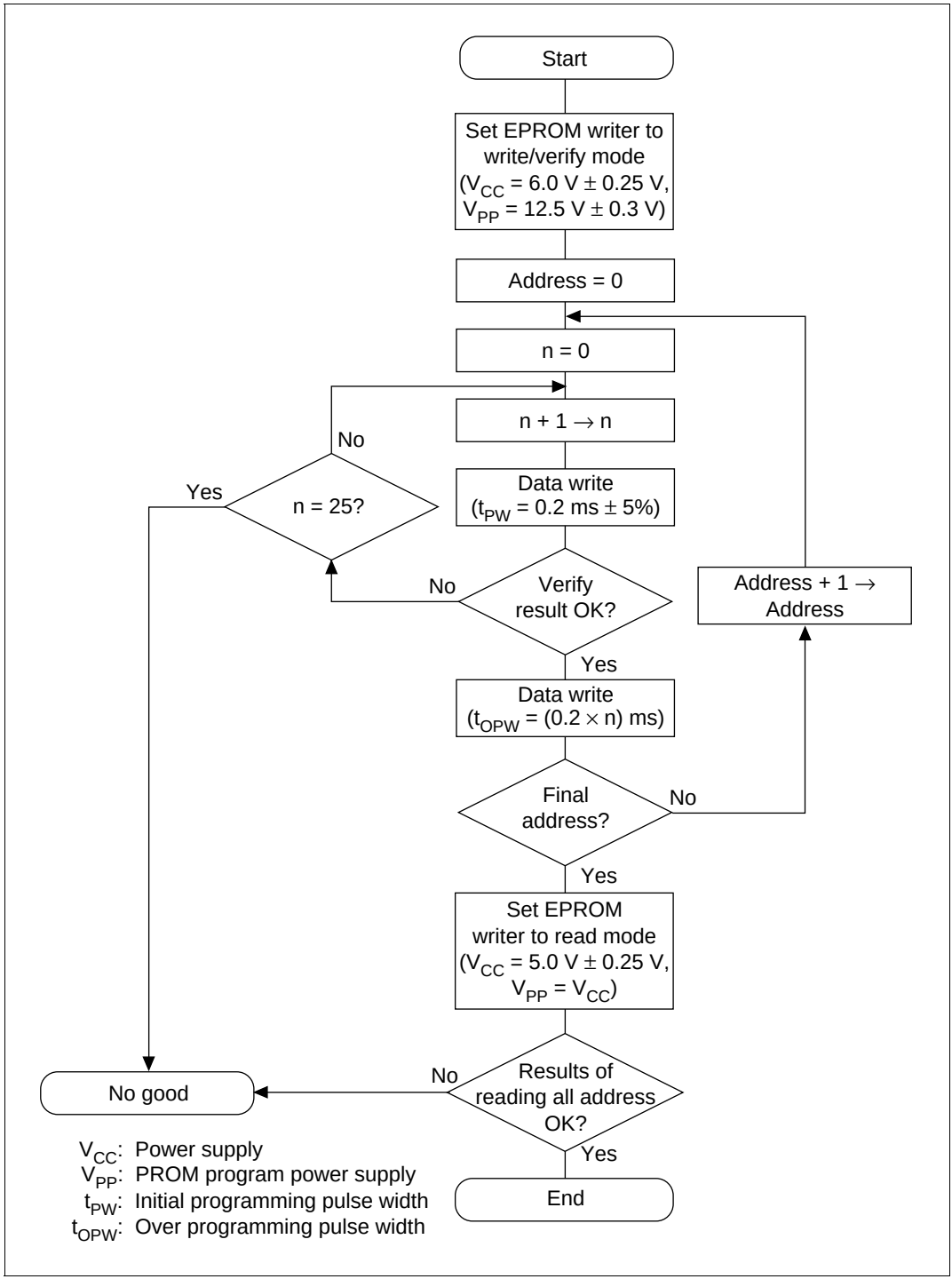



**Figure 16.4 Basic Flowchart of High-Speed High-Reliability Programming**

**Electrical Characteristics:** Tables 16.3 and 16.4 show the electrical characteristics of programming. Figure 16.5 shows the timing.

**Table 16.3 DC Characteristics ($V_{CC}$ = 6.0 V ± 0.25 V, $V_{PP}$ = 12.5 ± 0.3 V, $V_{SS}$ = 0 V, Ta = 25 ± 5˚C)**

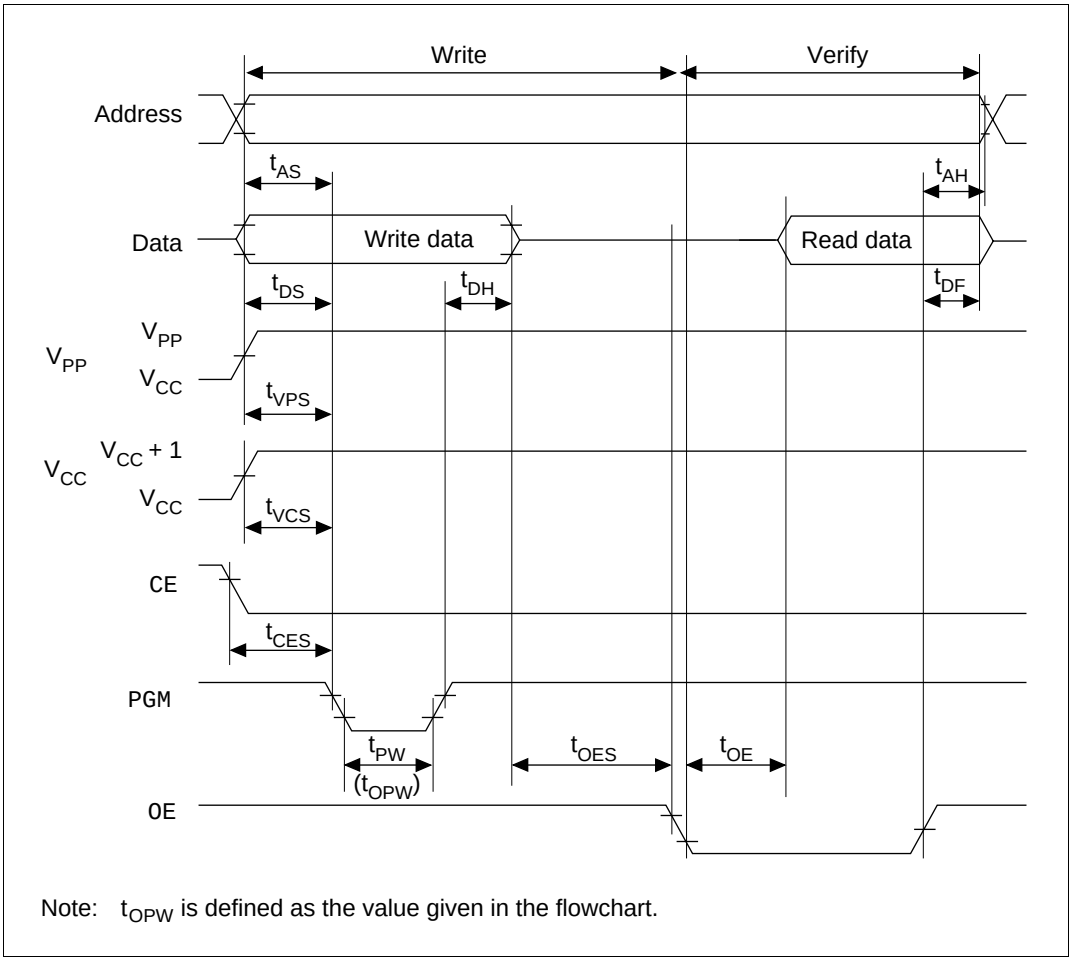
| Item | Pins | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Input high voltage | I/O7–I/O0, A16–A0, OE, CE, PGM | $V_{IH}$ | 2.4 | — | $V_{CC}$ + 0.3 | V | |
| Input low voltage | I/O7–I/O0, A16–A0, OE, CE, PGM | $V_{IL}$ | –0.3 | — | 0.8 | V | |
| Output high voltage | I/O7–I/O0 | $V_{OH}$ | 2.4 | — | — | V | $I_{OH}$ = -200 µA |
| Output low voltage | I/O7–I/O0 | $V_{OL}$ | — | — | 0.45 | V | $I_{OL}$ = 1.6 mA |
| Input leak current | I/O7–I/O0, A16–A0, OE, CE, PGM | $\|I_{LI}\|$ | — | — | 2 | µA | $V_{IN}$ = 5.25 V/0.5 V |
| $V_{CC}$ current | | $I_{CC}$ | — | — | 40 | mA | |
| $V_{PP}$ current | | $I_{PP}$ | — | — | 40 | mA | |

**Table 16.4 AC Characteristics ($V_{CC}$ = 6.0 V ± 0.25 V, $V_{PP}$ = 12.5 ± 0.3 V, $V_{SS}$ = 0 V, Ta = 25 ± 5°C)**

| Item | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|
| Address setup time | $t_{AS}$ | 2 | — | — | µs | Figure 16.5*1 |
| OE setup time | $t_{OES}$ | 2 | — | — | µs | |
| Data setup time | $t_{DS}$ | 2 | — | — | µs | |
| Address hold time | $t_{AH}$ | 0 | — | — | µs | |
| Data hold time | $t_{DH}$ | 2 | — | — | µs | |
| Data output disable time | $t_{DF}$*2 | — | — | 130 | ns | |
| $V_{PP}$ setup time | $t_{VPS}$ | 2 | — | — | µs | |
| PGM pulse width in initial programming | $t_{PW}$ | 0.19 | 0.20 | 0.21 | ms | |
| PGM pulse width in over programming | $t_{OPW}$*3 | 0.19 | — | 5.25 | ms | |
| $V_{CC}$ setup time | $t_{VCS}$ | 2 | — | — | µs | |
| CE setup time | $t_{CES}$ | 2 | — | — | µs | |
| Data output delay time | $t_{OE}$ | 0 | — | 150 | ns | |

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

2. $t_{DF}$ is defined as when output reaches the release state and the output level could not be referenced.
3. $t_{OPW}$ is defined as the value given in the flowchart.




Note: $t_{OPW}$ is defined as the value given in the flowchart.

**Figure 16.5 Write/Verify Timing**

### 16.3.3 Points to Note About Writing

1. Always write using the prescribed voltage and timing. The write voltage (programming voltage) $V_{PP}$ is 12.5 V (when the EPROM writer is set to the Hitachi specifications for HN27C101, $V_{PP}$ becomes 12.5 V.) Applying a voltage in excess of the rated voltage may damage the device. Pay particular attention to overshooting in the EPROM writer.

2. Before programming, always check that the indexes of the EPROM writer socket, socket adapter, and devices are consistent with each other. If they are not mounted in the proper location, an overcurrent may be generated, damaging the device.

3. Do not touch the socket adapter or device during writing. Contact can cause malfunctions that prevent data from being written accurately.

4. You cannot write in the page programming mode. Always set the equipment to the byte programming mode.

5. The capacity of the on-chip ROM is 32 kbytes, so the data of PROM writer addresses H'08000–H'1FFFF should be H'FF. Always set the range for PROM addresses to H'0000–H'7FFF.

6. When write errors occur on consecutive addresses, stop writing. Check to see if there are any abnormalities in the EPROM writer and socket adapter.

### 16.3.4 Reliability After Writing

After programming, we recommend letting the device stand at high temperature (125–150°C) for 24–48 hours to increase the reliability of data retention. Letting it stand at high temperature is a type of screening method that can get rid of initial data retention defects of the on-chip PROM's memory cell within a short period of time. Figure 16.6 shows the flow from programming of the on-chip PROM, including screening, to mounting on the device board.

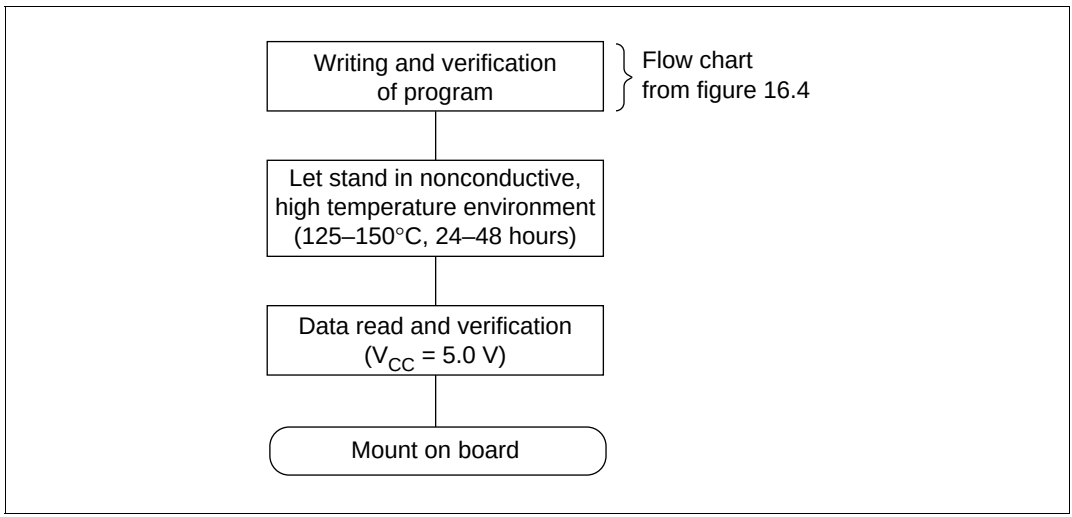



**Figure 16.6 Screening Flow**

If abnormalities are found when the program is written and verified or the program is read and checked after the writing/verification or letting the chip stand at high temperature, contact Hitachi's engineering departments.

# Section 17 RAM

## 17.1 Overview

The SH7020 and SH7021 has 1-kbytes of on-chip RAM. The on-chip RAM is linked to the CPU and direct memory access controller (DMAC) with a 32-bit data bus (figure 17.1). The CPU can access data in the on-chip RAM in byte, word, or long word units. The DMAC can access byte or word data. On-chip RAM data can always be accessed in one state, making the RAM ideal for use as a program area, stack area, or data area, which require high-speed access. The contents of the on-chip RAM are held in both the sleep and standby modes. Memory area 7 addresses H'FFFFC00 to H'FFFFFFF are allocated to the on-chip RAM.

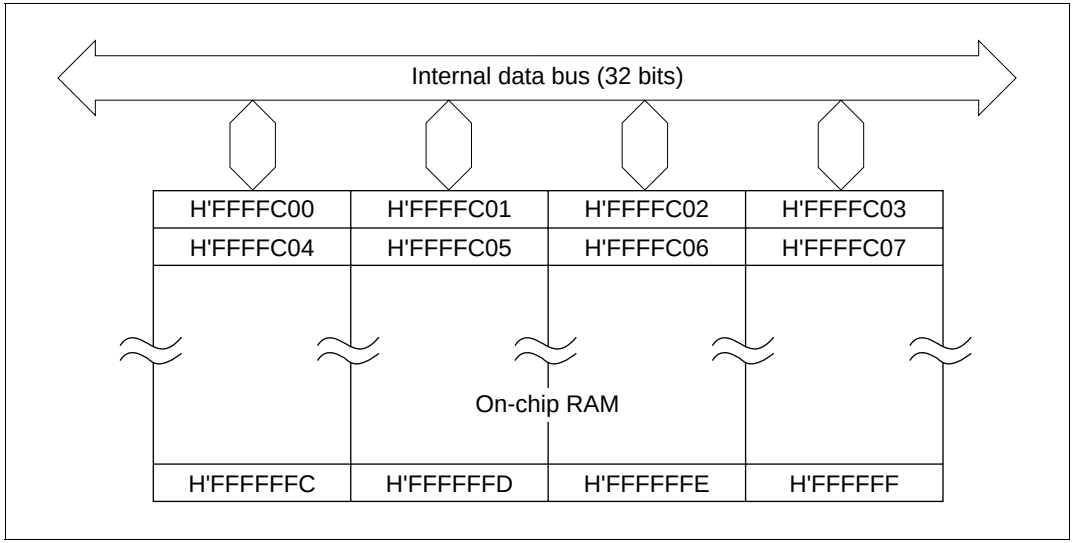



**Figure 17.1 Block Diagram of RAM**

## 17.2 Operation

Accesses to addresses H'FFFFC00–H'FFFFFFF are directed to the on-chip RAM. Memory area 7 (H'F000000–H'FFFFFFF) is divided into shadows in 1 kbyte units. All shadow accesses are on-chip RAM accesses. For more information on shadows, see section 8, Bus State Controller.

# Section 18 Power-Down States

## 18.1 Overview

In the power-down mode, all CPU functions are halted. This lowers power consumption dramatically.

### 18.1.1 Power-Down Modes

The SH microprocessor has two power-down modes.

1. Sleep mode
2. Standby mode

The sleep mode and standby mode are entered from the program execution state according to the transition conditions given in table 18.1. Table 18.1 also describes procedures for canceling each mode and the states of the CPU and peripheral functions.

**Table 18.1 Power-Down States**

| | | State | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Mode** | **Entering Procedure** | **Clock** | **CPU** | **Peripheral Functions** | **CPU Registers** | **RAM** | **I/O Ports** | **Canceling Procedure** |
| Sleep mode | Execute SLEEP instruction with SBY bit set to 0 in SBYCR | Run | Halt | Run | Held | Held | Held | • Interrupt<br>• DMA address error<br>• Power-on reset<br>• Manual reset |
| Standby mode | Execute SLEEP instruction with SBY bit set to 1 in SBYCR | Halt | Halt | Halt*1 | Held | Held | Held or high-Z*2 | • NMI<br>• Power-on reset<br>• Manual reset |

SBYCR: Standby control register

SBY: Standby bit

Notes: 1. Some of the registers of the on-chip peripheral modules are not initialized in the standby mode. For details, see table 18.3, Status of Registers in the Standby Mode in section 18.4.1, Transition to the Standby Mode, or the descriptions of registers given where the on-chip peripheral modules are covered.

2. The status of I/O ports in the standby mode are set by the port high-impedance bit (HIZ) of the SBYCR. See section 18.2, Standby Control Register (SBYCR) for details. The status of pins other than the I/O ports are described in appendix B, Pin States.

### 18.1.2 Register

Table 18.2 summarizes the register related to the power-down state.

**Table 18.2 Standby Control Register (SBYCR)**

| Name | Abbreviation | R/W | Initial Value | Address | Access size |
|---|---|---|---|---|---|
| Standby control register | SBYCR | R/W | H'1F | H'5FFFFBC | 8, 16, 32 |

## 18.2 Standby Control Register (SBYCR)

The standby control register (SBYCR) is an 8-bit register that can be read or written to. It is set in order to enter the standby mode and also sets the port states in standby mode. The SBYCR is initialized to H'1F when reset.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Bit name: | SBY | HIZ | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| R/W: | R/W | R/W | — | — | — | — | — | — |

- Bit 7 (standby (SBY)): SBY enables transition to the standby mode. The SBY bit cannot be set to 1 while the timer enable bit (bit TME) in timer control/status register TCSR of watchdog timer WDT is set to 1. To enter the standby mode, clear the TME bit to 0 to halt the WDT and set the SBY bit.

| SBY | Description |
|---|---|
| 0 | Executing SLEEP instruction puts the LSI into sleep mode (initial value) |
| 1 | Executing SLEEP instruction puts the LSI into standby mode |

- Bit 6 (port high-impedance (HIZ)): HIZ selects whether I/O ports remain in their previous states during standby, or are placed in the high-impedance state when the standby mode is entered. The HIZ bit cannot be set to 1 while the TME bit is set to 1. To place the pins of the I/O ports in high impedance, clear the TME bit to 0 before setting the HIZ bit.

| HIZ | Description |
|---|---|
| 0 | Port states are maintained during standby (initial value) |
| 1 | Ports are placed in the high-impedance state in standby |

- Bits 5–0 (reserved): Bit 5 is a read-only bit that always reads as 0. Only write 0 to bit 5. Writing to bits 4–0 is disabled. These bits always read 1.

## 18.3 Sleep Mode

### 18.3.1 Transition to the Sleep Mode

Execution of the SLEEP instruction when the standby bit (SBY) in the standby control register (SBYCR) is cleared to 0 causes a transition from the program execution state to the sleep mode. Although the CPU halts immediately after executing the SLEEP instruction, the contents of its internal registers remain unchanged. The on-chip peripheral modules do not halt in the sleep mode.

### 18.3.2 Canceling the Sleep Mode

The sleep mode is canceled by an interrupt, DMA address error, power-on reset, or manual reset.

**Cancellation by an Interrupt:** When an interrupt occurs, the sleep mode is canceled and interrupt exception processing is executed. The sleep mode is not canceled if the interrupt cannot be accepted because its priority level is equal to or less than the mask level set in the CPU's status register (SR). Likewise, the sleep mode is not canceled if the interrupt is disabled by the on-chip peripheral module.

**Cancellation by a DMA Address Error:** If the DMAC operates during the sleep mode and a DMA address error occurs, the sleep mode is canceled and DMA address error exception-processing is executed.

**Cancellation by a Power-On Reset:** If the $\overline{\text{RES}}$ signal goes low while the NMI signal is high, the sleep mode is canceled and the power-on reset state is entered. If the NMI signal is brought from low to high in order to set the LSI for power-on resets, an NMI interrupt will occur whenever the rising edge of the NMI is selected as the valid edge (in NMI edge select bit NMIE of the interrupt control register ICR of the interrupt controller). When this occurs, the NMI interrupt cancels the sleep mode.

**Cancellation by a Manual Reset:** If the $\overline{\text{RES}}$ signal goes low while the NMI signal is low, the sleep mode is canceled and the manual reset state is entered. If the NMI signal is brought from high to low in order to set the LSI for manual resets, the sleep mode will be canceled by an NMI interrupt whenever the falling edge of the NMI is selected as the valid edge (in the NMIE bit).

## 18.4 Standby Mode

### 18.4.1 Transition to the Standby Mode

To enter the standby mode, set the standby bit (SBY) to 1 in the standby control register (SBYCR), then execute the SLEEP instruction. The LSI moves from the program execution state to the standby mode. The standby mode greatly reduces power consumption by halting not only the CPU, but the clock and on-chip peripheral modules as well. Some registers of the on-chip peripheral modules are initialized, others are not (See table 18.3). As long as the specified voltage is supplied, however, CPU register contents and on-chip RAM data are held. The I/O port state (hold or high impedance) depends on the port high-impedance bit (HIZ) in the SBYCR. For details on the states of these pins, see appendix B. Pin States.

**Table 18.3 Register States in the Standby Mode**

| Module | Register Initialized | Registers That Hold Data |
|---|---|---|
| Interrupt controller (INTC) | — | All registers |
| User break controller (UBC) | — | All registers |
| Bus state controller (BSC) | — | All registers |
| Pin function controller (PFC) | — | All registers |
| I/O ports | — | All registers |
| Direct memory access controller (DMAC) | All registers | — |
| Watchdog timer (WDT) | • Bits 7–5 (OVF, WT/IT, TME) of the timer control status register (TCSR)<br>• Reset control/status register (RSTCSR) | • Bits 2–0 (CKS2–CKS0) of the timer control status register (TCSR)<br>• Timer counter (TCNT) |
| 16-bit integrated timer pulse unit (ITU) | All registers | — |
| Programmable timing pattern controller (TPC) | — | All registers |
| Serial communications interface (SCI) | • Receive data register (RDR)<br>• Transmit data register (TDR)<br>• Serial mode register (SMR)<br>• Serial control register (SCR)<br>• Serial status register (SSR)<br>• Bit rate register (BBR) | — |
| Power-down state register | — | Standby control register (SBYCR) |

### 18.4.2 Canceling the Standby Mode

The standby mode is canceled by an NMI interrupt, a power-on reset, or a manual reset.

**Cancellation by an NMI:** When a rising edge or falling edge (as selected by the NMIE bit in interrupt control register ICR of interrupt controller INTC) is detected at the NMI pin, the clock oscillator begins operating. At first, clock pulses are supplied only to the watchdog timer. After the time that was selected before entering the standby mode using clock select bits 2–0 (CKS2–CKS0) in the timer control/status register TCSR of the watchdog timer WDT, the watchdog timer overflows. After the overflow, the clock is considered stable and supplied to the entire chip. The standby mode is canceled and the NMI exception-processing sequence begins.

When the standby mode is cleared by an NMI interrupt, bits CKS2–CKS0 must be set so that the WDT overflow interval is equal to or greater than the clock settling time. When the standby mode is cleared when the fall edge has been selected in the NMI bit, be sure that the NMI pin is high when standby is entered (when the clock is halted) and low when the chip returns from standby (clock starts up after oscillator is stabilized). Likewise, when the standby mode is cleared when the rise edge has been selected in the NMI bit, be sure that the NMI pin is low when standby is entered (clock halted) and high when the chip returns from standby (clock starts up after oscillator is stabilized).

**Cancellation by a Power-On Reset:** If the $\overline{\text{RES}}$ signal goes low while the NMI signal is high, the standby mode is canceled and the power-on reset state is entered. If the NMI signal is brought from low to high in order to set the LSI for power-on resets, the standby mode will not be canceled by an NMI interrupt, because the NMI signal is initialized for the falling edge in the standby mode (by the NMIE bit).

**Cancellation by a Manual Reset:** If the $\overline{\text{RES}}$ signal goes low while the NMI signal is low, the standby mode is canceled and the manual reset state is entered. If the NMI signal is brought from high to low in order to set the LSI for manual resets, the standby mode will first be canceled by an NMI interrupt, because the NMI signal is initialized for the falling edge in the standby mode (by the NMIE bit).

### 18.4.3 Standby Mode Application

In this example, the standby mode is entered on the falling edge of the NMI signal and canceled on the rising edge of the NMI signal. Figure 18.1 shows the timing.

After an NMI interrupt is accepted (high goes to low) while the NMI edge select bit NMIE in the interrupt control register ICR is cleared to 0 to select detection of the falling edge, the NMI exception service routine sets the NMIE to 1 (selecting detection of the rising edge) and sets the SBY bit to 1. Finally, it executes a SLEEP instruction to enter the standby mode.

The standby mode is canceled on the rising edge of the NMI signal.

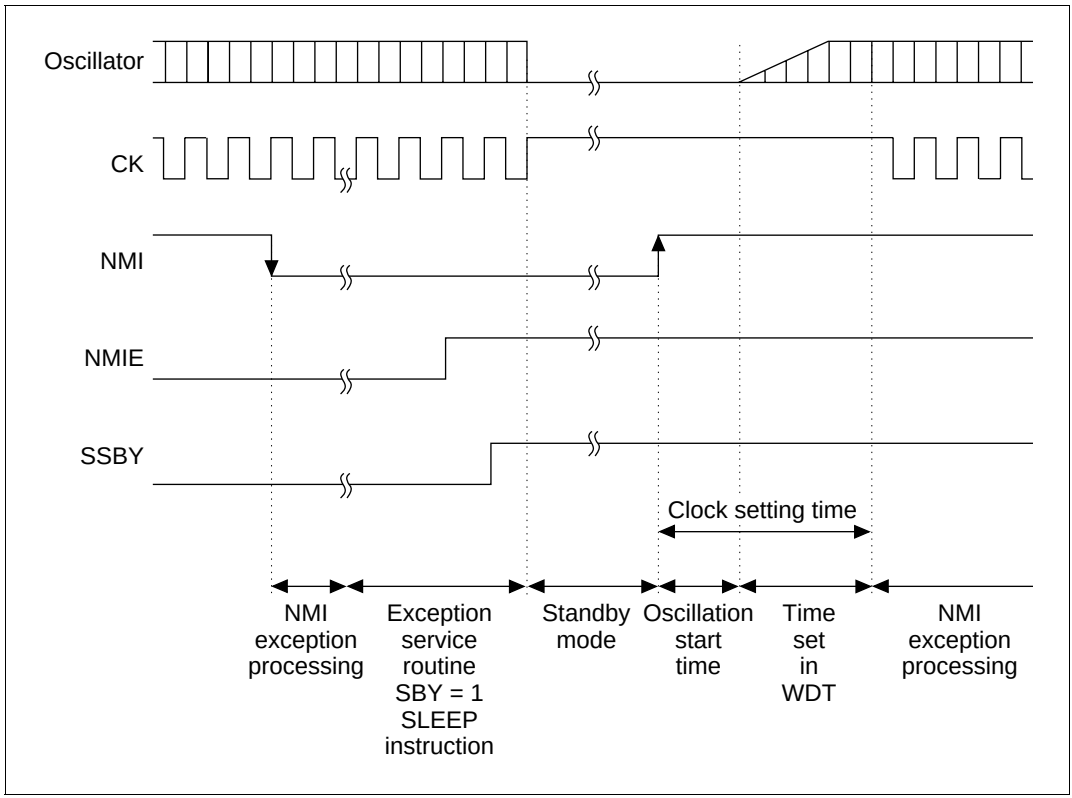



**Figure 18.1 NMI Timing for the Standby Mode (Example)**

# Section 19 Electrical Characteristics

## 19.1 Absolute Maximum Ratings

**Table 19.1 Absolute Maximum Ratings**

| Item | Symbol | Rating | Unit |
|---|---|---|---|
| Power supply voltage | $V_{CC}$ | –0.3 to +7.0 | V |
| Program voltage | $V_{PP}$ | –0.3 to +13.5 | V |
| Input voltage | Vin | –0.3 to $V_{CC}$ + 0.3 | V |
| Operating temperature | Topr | –20 to +75* | ˚C |
| Storage temperature | Tstg | –55 to +125 | ˚C |

Caution: Operating the LSI in excess of the absolute maximum rating may result in permanent damage.

Note: Normal Products: Topr = –40 to +85°C for wide-temperature range products

## 19.2 DC Characteristics

Table 19.2 lists DC characteristics. Table 19.3 lists the permissible output current values.

**Usage Conditions:**

- The current consumption value is measured under conditions of $V_{IH}$ min = $V_{CC}$ – 0.5 V and $V_{IL}$ max = 0.5 V with no load on any output pin and the on-chip pull-up MOS off.

**Table 19.2 DC Characteristics (1)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Input high-level voltage | RES, NMI, MD2–MD0 | $V_{IH}$ | $V_{CC}$ – 0.7 | — | $V_{CC}$ + 0.3 | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC}$ + 0.3 | V | |
| | Other input pins | | 2.2 | — | $V_{CC}$ + 0.3 | V | |
| Input low-level voltage | RES, NMI, MD2–MD0 | $V_{IL}$ | –0.3 | — | 0.5 | V | |
| | Other input pins | | –0.3 | — | 0.8 | V | |
| Schmidt trigger input voltage | PA13–PA10, PA2, PA0, PB7–PB0 | $V_T^+$ | 4.0 | — | — | V | |
| | | $V_T^-$ | — | — | 1.0 | V | |
| | | $V_T^+ – V_T^-$ | 0.4 | — | — | V | |
| Input leak current | RES | \|Iin\| | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| | NMI, MD2–MD0 | | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| 3-state leak current (while off) | Ports A and B, CS3–CS0, A21–A0, AD15–AD0 | $\|I_{TSI}\|$ | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| Input pull-up MOS current | PA3 | –Ip | 20 | — | 300 | µA | Vin = 0V |
| Output high-level voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | 3.5 | — | — | V | $I_{OH}$ = –1 mA |

**Table 19.2 DC Characteristics (1) (cont)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products .

| Item | | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Output low level voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | | | — | — | 1.2 | V | $I_{OL}$ = 8 mA |
| Input capacitance | RES | Cin | — | — | 30 | pF | Vin = 0 V<br>Input signal<br>f = 1 MHz<br>Ta = 25°C |
| | NMI | | — | — | 30 | pF | |
| | All other input pins | | — | — | 20 | pF | |
| Current consumption | Ordinary operation | $I_{CC}$ | — | 65 | 80 | mA | f = 12.5 MHz |
| | | | — | 75 | 90 | mA | f = 16.6 MHz |
| | | | — | 90 | 100 | mA | f = 20 MHz |
| | Sleep | | — | 30 | 50 | mA | f = 12.5 MHz |
| | | | — | 35 | 55 | mA | f = 16.6 MHz |
| | | | — | 40 | 60 | mA | f = 20 MHz |
| | Standby | | — | 0.01 | 5 | µA | Ta ≤ 50°C |
| | | | — | — | 20.0 | µA | 50°C < Ta |
| RAM stand-by | voltage | $V_{RAM}$ | 2.0 | — | — | V | |

**Usage Notes:**

1. Current dissipation values are for $V_{IH}$ min = $V_{CC}$ - 0.5 V and $V_{IL}$ max = 0.5 V with all output pins unloaded and the on-chip pull-up transistors in the off state.
5. The ZTAT and mask versions have the same functions, and the electrical characteristics of both are within specification, but characteristic-related performance values, operating margins, noise margins, noise emission, etc., are different. Caution is therefore required in carrying out system design, and when switching between ZTAT and mask versions.

**Table 19.2 DC Characteristics (2)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Input high-level voltage | RES, NMI, MD2–MD0 | $V_{IH}$ | $V_{CC}$ – 0.7 | — | $V_{CC}$ + 0.3 | V | |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC}$ + 0.3 | V | |
| | Other input pins | | 2.2 | — | $V_{CC}$ + 0.3 | V | |
| Input low-level voltage | RES, NMI, MD2–MD0 | $V_{IL}$ | –0.3 | — | 0.5 | V | |
| | Other input pins | | –0.3 | — | 0.8 | V | |
| Schmidt trigger input voltage | PA13–10, PA2, PA0, PB7–PB0 | $V_T^+$ | 4.0 | — | — | V | |
| | | $V_T^-$ | — | — | 1 | V | |
| | | $V_T^+ – V_T^-$ | 0.4 | — | — | V | |
| Input leak current | RES | \|Iin\| | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| | NMI, MD2–MD0 | | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| 3-state leak current (while off) | Ports A and B, CS3–CS0, A21–A0, AD15–AD0 | $\|I_{TSI}\|$ | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| Input pull-up MOS current | PA3 | –Ip | 20 | — | 300 | µA | Vin = 0 V |
| Output high-level voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | 3.5 | — | — | V | $I_{OH}$ = –1 mA |
| Output low level voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | | | — | — | 1.2 | V | $I_{OL}$ = 8 mA |

**Table 19.2 DC Characteristics (2) (cont)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products .

| Item | | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Input capacitance | RES | Cin | — | — | 30 | pF | Vin = 0 V<br>Input signal<br>f = 1 MHz<br>Ta = 25°C |
| | NMI | | — | — | 30 | pF | |
| | All other input pins | | — | — | 20 | pF | |
| Current consumption | Ordinary operation | $I_{CC}$ | — | 65 | 80 | mA | f = 12.5 MHz |
| | | | — | 75 | 90 | mA | f = 16.6 MHz |
| | Sleep | | — | 30 | 50 | mA | f = 12.5 MHz |
| | | | — | 35 | 55 | mA | f = 16.6 MHz |
| | Standby | | — | 0.01 | 5 | µA | Ta ≤ 50°C |
| | | | — | — | 20.0 | µA | 50°C < Ta |
| RAM stand-by | voltage | $V_{RAM}$ | 2.0 | — | — | V | |

**Table 19.2 DC Characteristics (3)**

Conditions: $V_{CC}$ = 3.0 V to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Input high-level voltage | RES, NMI, MD2–MD0 | $V_{IH}$ | $V_{CC} \times 0.9$ | — | $V_{CC} + 0.3$ | 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 | |
| Input low-level voltage | RES, NMI, MD2–MD0 | $V_{IL}$ | –0.3 | — | $V_{CC} \times 0.1$ | V | |
| | Other input pins | | –0.3 | — | $V_{CC} \times 0.2$ | V | |
| Schmidt trigger input voltage | PA13–10, PA2, PA0, PB7–PB0 | $V_T^+$ | $V_{CC} \times 0.9$ | — | — | V | |
| | | $V_T^-$ | — | — | $V_{CC} \times 0.2$ | V | |
| | | $V_T^+ – V_T^-$ | $V_{CC} \times 0.07$ | — | — | V | |
| Input leak current | RES | \|Iin\| | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| | NMI, MD2–MD0 | | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| 3-state leak current (while off) | Ports A and B, CS3–CS0, A21–A0, AD15–AD0 | $\|I_{TSI}\|$ | — | — | 1.0 | µA | Vin = 0.5 to $V_{CC}$ – 0.5 V |
| Input pull-up MOS current | PA3 | –Ip | 20 | — | 300 | µA | Vin = 0V |
| Output high-level voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | $V_{CC}$ – 1.0 | — | — | V | $I_{OH}$ = –1 mA |
| Output low level voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 1.6 mA |
| | | | — | — | 1.2 | V | $I_{OL}$ = 8 mA |

**Table 19.2 DC Characteristics (3) (cont)**

Conditions: $V_{CC}$ = 3.0 V to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products .

| Item | | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|---|---|---|---|---|---|---|---|
| Input capacitance | RES | Cin | — | — | 30 | pF | Vin = 0 V<br>Input signal<br>f = 1 MHz<br>Ta = 25°C |
| | NMI | | — | — | 30 | pF | |
| | All other input pins | | — | — | 20 | pF | |
| Current consumption | Ordinary operation | $I_{CC}$ | — | 65 | 80 | mA | f = 12.5 MHz |
| | Sleep | | — | 30 | 50 | mA | f = 12.5 MHz |
| | Standby | | — | 0.01 | 5.0 | µA | Ta ≤ 50°C |
| | | | — | — | 20 | µA | 50°C < Ta |
| RAM stand-by voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

**Table 19.3 Permitted Output Current Values**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*
Case C: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | 12.5 MHz | | | |
|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Typ** | **Max** | **Unit** |
| Output low-level permissible current (per pin) | $I_{OL}$ | — | — | 10 | mA |
| Output low-level permissible current (total) | $\Sigma I_{OL}$ | — | — | 80 | mA |
| Output high-level permissible current (per pin) | $-I_{OH}$ | — | — | 2.0 | mA |
| Output high-level permissible current (total) | $-\Sigma I_{OH}$ | — | — | 25 | mA |

Caution: To ensure LSI reliability, do not exceed the value for output current given in table 19.3.

## 19.3 AC Characteristics

The following AC timing chart represents the AC characteristics, not signal functions. For signal functions, see the explanation in the text.

### 19.3.1 Clock Timing

**Table 19.4 Clock Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | Case A | | Case B | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 12.5 MHz | | 16.6 MHz | | 20 MHz | | | |
| Item | Symbol | Min | Max | Min | Max | Min | Max | Unit | Figures |
| EXTAL input high level pulse width | $t_{EXH}$ | 20 | — | 10 | — | 10 | — | ns | 19.1 |
| EXTAL input low level pulse width | $t_{EXL}$ | 20 | — | 10 | — | 10 | — | ns | |
| EXTAL input rise time | $t_{EXr}$ | — | 10 | — | 5 | — | 5 | ns | |
| EXTAL input fall time | $t_{EXf}$ | — | 10 | — | 5 | — | 5 | ns | |
| Clock cycle time | $t_{cyc}$ | 80 | — | 60 | 500 | 50 | 500 | ns | 19.1, 19.2 |
| Clock high pulse width | $t_{CH}$ | 30 | — | 20 | — | 20 | — | ns | 19.2 |
| Clock low pulse width | $t_{CL}$ | 30 | — | 20 | — | 20 | — | ns | |
| Clock rise time | $t_{Cr}$ | — | 10 | — | 5 | — | 5 | ns | |
| Clock fall time | $t_{Cf}$ | — | 10 | — | 5 | — | 5 | ns | |
| Reset oscillation settling time | $t_{OSC1}$ | 10 | — | 10 | — | 10 | — | ms | 19.3 |
| Software standby oscillation settling time | $t_{OSC2}$ | 10 | — | 10 | — | 10 | — | ms | |

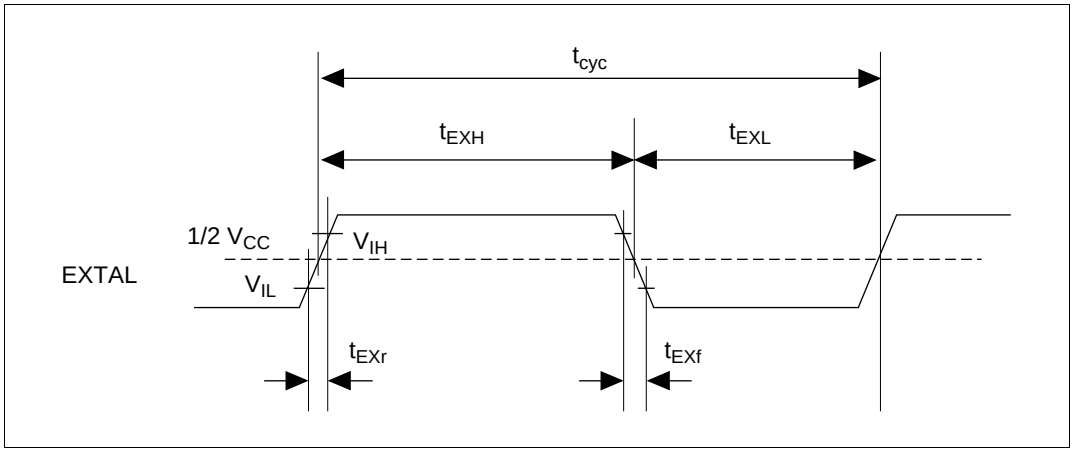



**Figure 19.1 EXTAL Input Timing**

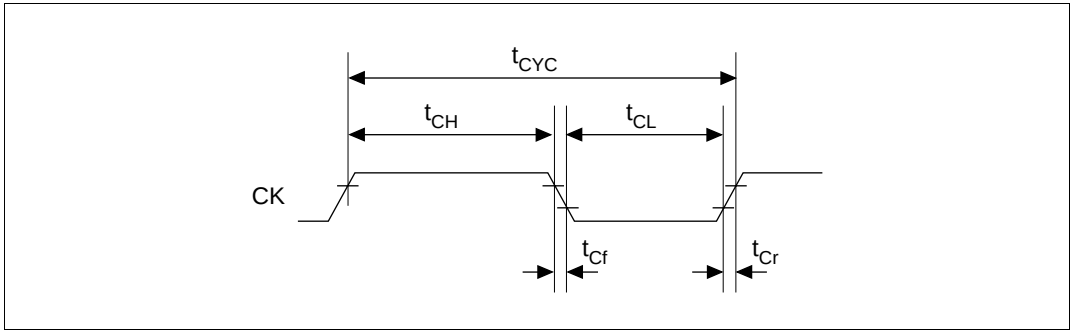



**Figure 19.2 System Clock Timing**

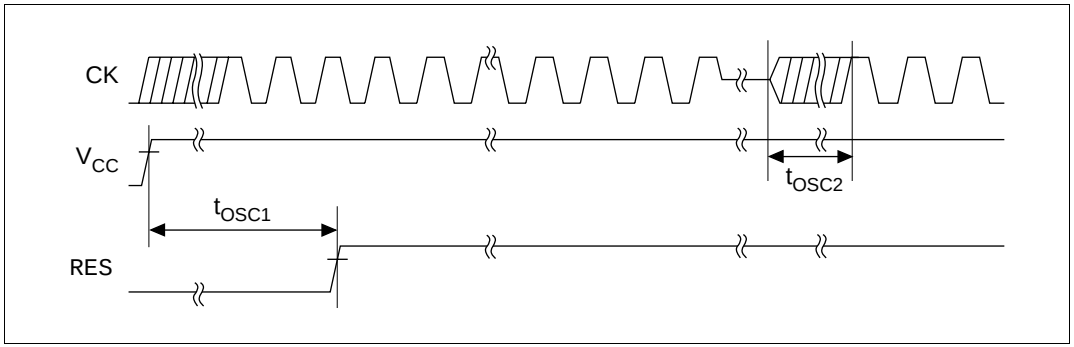



**Figure 19.3 Oscillation Settling Time**

### 19.3.2 Control Signal Timing

**Table 19.5 Control Signal Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | Case A | | Case B | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 12.5 MHz | | 16.6 MHz | | 20 MHz | | | |
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Figure** |
| RES setup time | $t_{RESS}$ | 320 | — | 240 | — | 200 | — | ns | 19.4 |
| RES pulse width | $t_{RESW}$ | 20 | — | 20 | — | 20 | — | $t_{cyc}$ | |
| NMI reset setup time | $t_{NMIRS}$ | 320 | — | 240 | — | 200 | — | ns | |
| NMI reset hold time | $t_{NMIRH}$ | 320 | — | 240 | — | 200 | — | ns | |
| NMI setup time | $t_{NMIS}$ | 160 | — | 120 | — | 100 | — | ns | 19.5 |
| NMI hold time | $t_{NMIH}$ | 80 | — | 60 | — | 50 | — | ns | |
| IRQ0–IRQ7 setup time (edge detection time) | $t_{IRQES}$ | 160 | — | 120 | — | 100 | — | ns | |
| IRQ0–IRQ7 setup time (level detection time) | $t_{IRQLS}$ | 160 | — | 120 | — | 100 | — | ns | |
| IRQ0–IRQ7 hold time | $t_{IRQEH}$ | 80 | — | 60 | — | 50 | — | ns | |
| IRQOUT output delay time) | $t_{IRQOD}$ | — | 80 | — | 60 | — | 50 | ns | 19.6 |
| Bus request setup time | $t_{BRQS}$ | 80 | — | 60 | — | 50 | — | ns | 19.7 |
| Bus acknowledge delay time 1 | $t_{BACD1}$ | — | 80 | — | 60 | — | 50 | ns | |
| Bus acknowledge delay time 2 | $t_{BACD2}$ | — | 80 | — | 60 | — | 50 | ns | |
| Bus 3-state delay time | $t_{BZD}$ | — | 80 | — | 60 | — | 50 | ns | |

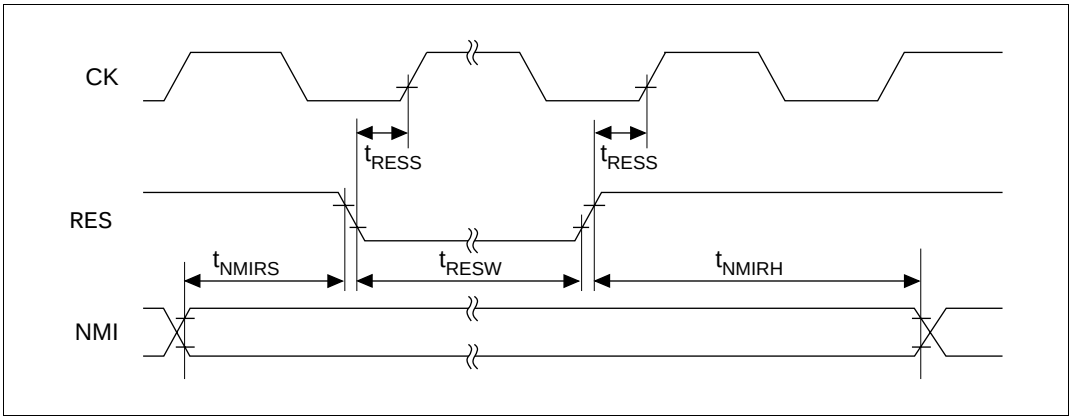



**Figure 19.4 Reset Input Timing**

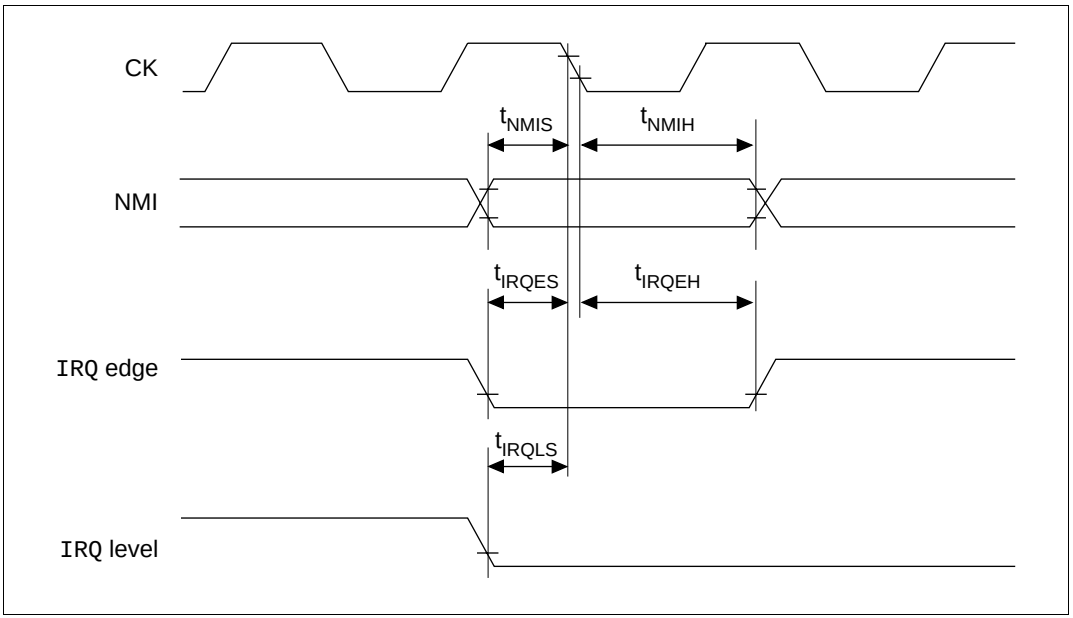



**Figure 19.5 Interrupt Signal Input Timing**

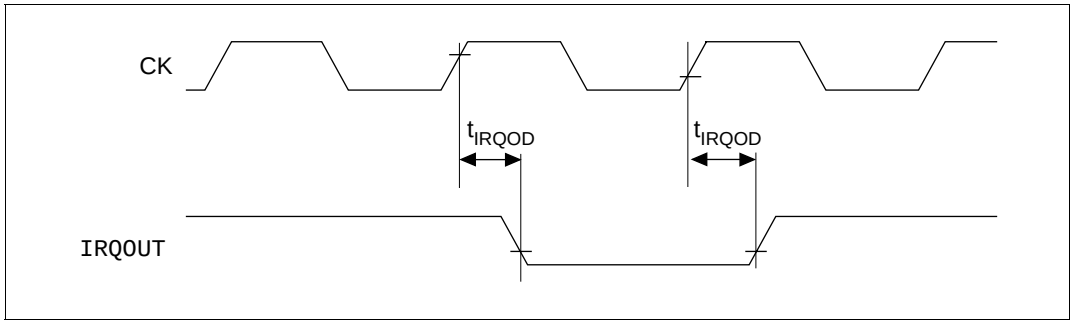



**Figure 19.6 Interrupt Signal Output Timing**

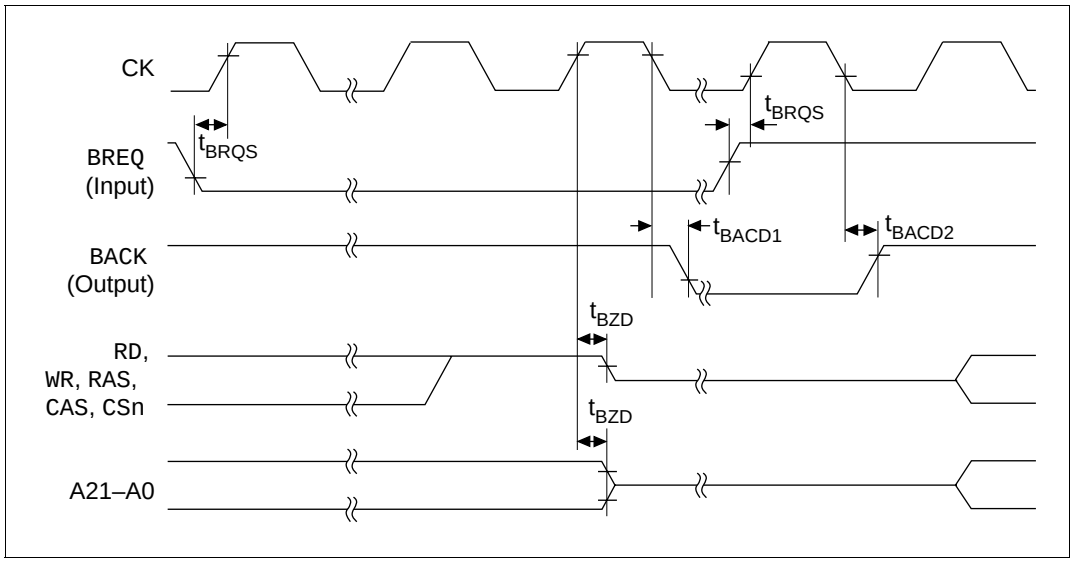



**Figure 19.7 Bus Release Timing**

### 19.3.3 Bus Timing

**Table 19.6 Bus Timing (1)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| Address delay time | | $t_{AD}$ | — | 20[*1] | ns | 19.8, 19.9, 19.11–19.14, 19.19, 19.20 |
| CS delay time 1 | | $t_{CSD1}$ | — | 25 | ns | 19.8, 19.9, 19.20 |
| CS delay time 2 | | $t_{CSD2}$ | — | 25 | ns | |
| CS delay time 3 | | $t_{CSD3}$ | — | 20 | ns | 19.19 |
| CS delay time 4 | | $t_{CSD4}$ | — | 20 | ns | |
| Access time 1[*6] from read strobe | 35% duty[*2] | $t_{RDAC1}$ | $t_{cyc} \times 0.65 - 20$ | — | ns | 19.8 |
| | 50% duty | | $t_{cyc} \times 0.5 - 20$ | — | ns | |
| Access time 2[*6] from read strobe | 35% duty[*2] | $t_{RDAC2}$ | $t_{cyc} \times (n+1.65) - 20$[*3] | — | ns | 19.9, 19.10 |
| | 50% duty | | $t_{cyc} \times (n+1.5) - - 20$[*3] | — | ns | |
| Access time 3[*6] from read strobe | 35% duty[*2] | $t_{RDAC3}$ | $t_{cyc} \times (n+0.65) - 20$[*3] | — | ns | 19.19 |
| | 50% duty | | $t_{cyc} \times (n+0.5) - - 20$[*3] | — | ns | |
| Read strobe delay time | | $t_{RSD}$ | — | 20 | ns | 19.8, 19.9, 19.11–19.15, 19.19, 19.24–19.28 |
| Read data setup time | | $t_{RDS}$ | 15 | — | ns | 19.8, 19.9, 19.11–19.14, 19.19 |
| Read data hold time | | $t_{RDH}$ | 0 | — | ns | |
| Write strobe delay time 1 | | $t_{WSD1}$ | — | 20 | ns | 19.9, 19.13, 19.14, 19.19, 19.20 |
| Write strobe delay time 2 | | $t_{WSD2}$ | — | 20 | ns | 19.9, 19.13, 19.14, 19.19 |
| Write strobe delay time 3 | | $t_{WSD3}$ | — | 20 | ns | 19.11, 19.12 |
| Write strobe delay time 4 | | $t_{WSD4}$ | — | 20 | ns | 19.11, 19.12, 19.20 |
| Write data delay time 1 | | $t_{WDD1}$ | — | 35 | ns | 19.9, 19.13, 19.14, 19 |
| Write data delay time 2 | | $t_{WDD2}$ | — | 20 | ns | 19.11, 19.12 |
| Write data hold time | | $t_{WDH}$ | 0 | — | ns | 19.9, 19.11–19.14 |

**Table 19.6 Bus Timing (1) (cont)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| Parity output delay time 1 | | $t_{WPDD1}$ | — | 40 | ns | 19.9, 19.13, 19.14 |
| Parity output delay time 2 | | $t_{WPDD2}$ | — | 20 | ns | 19.11, 19.12 |
| Parity output hold time | | $t_{WPDH}$ | 0 | — | ns | 19.9, 19.11–19.14 |
| Wait setup time | | $t_{WTS}$ | 14 | — | ns | 19.10, 19.15, 19.19 |
| Wait hold time | | $t_{WTH}$ | 10 | — | ns | |
| Read data access time 1*6 | | $t_{ACC1}$ | $t_{cyc}$ – 30*4 | — | ns | 19.8, 19.11, 19.12 |
| Read data access time 2*6 | | $t_{ACC2}$ | $t_{cyc} \times (n+2) – 30$*3 | — | ns | 19.9, 19.10, 19.13, 19.14 |
| RAS delay time 1 | | $t_{RASD1}$ | — | 20 | ns | 19.11–19.14, 19.16–19.18 |
| RAS delay time 2 | | $t_{RASD2}$ | — | 30 | ns | |
| CAS delay time 1 | | $t_{CASD1}$ | — | 20 | ns | 19.11 |
| CAS delay time 2 | | $t_{CASD2}$ | — | 20 | ns | 19.13, 19.14, 19.16–19.18 |
| CAS delay time 3 | | $t_{CASD3}$ | — | 20 | ns | |
| Column address setup time | | $t_{ASC}$ | 0 | — | ns | 19.11, 19.12 |
| Read data access time from CAS 1*6 | 35% duty*2 | $t_{CAC1}$ | $t_{cyc} \times 0.65 – 19$ | — | ns | |
| | 50% duty | | $t_{cyc} \times 0.5 – 19$ | — | ns | |
| Read data access time from CAS 2*6 | | $t_{CAC2}$ | $t_{cyc} \times (n+1) – 25$*3 | — | ns | 19.13, 19.14, 19.15 |
| Read data access time from RAS 1*6 | | $t_{RAC1}$ | $t_{cyc} \times 1.5 – 20$ | — | ns | 19.11, 19.12 |
| Read data access time from RAS 2*6 | | $t_{RAC2}$ | $t_{cyc} \times (n+2.5) – 20$*3 | — | ns | 19.13, 19.14, 19.15 |
| High-speed page mode CAS precharge time | | $t_{CP}$ | $t_{cyc} \times 0.25$ | — | ns | 19.12 |

**Table 19.6 Bus Timing (1) (cont)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| AH delay time 1 | | $t_{AHD1}$ | — | 20 | ns | 19.19 |
| AH delay time 2 | | $t_{AHD2}$ | — | 20 | ns | |
| Multiplexed address delay time | | $t_{MAD}$ | — | 30 | ns | |
| Multiplexed address hold time | | $t_{MAH}$ | 0 | — | ns | |
| DACK0, DACK1 delay time 1 | | $t_{DACD1}$ | — | 23 | ns | 19.8, 19.9, 19.11–19.14, 19.19, 19.20 |
| DACK0, DACK1 delay time 2 | | $t_{DACD2}$ | — | 23 | ns | |
| DACK0, DACK1 delay time 3 | | $t_{DACD3}$ | — | 20 | ns | 19.9, 19.13, 19.14, 19.19 |
| DACK0, DACK1 delay time 4 | | $t_{DACD4}$ | — | 20 | ns | 19.11, 19.12 |
| DACK0, DACK1 delay time 5 | | $t_{DACD5}$ | — | 20 | ns | |
| Read delay time | 35% duty*2 | $t_{RDD}$ | — | $t_{cyc} \times 0.35 + 12$ | ns | 19.8, 19.9, 19.11-19.15, 19.19, 19.24-19.28 |
| | 50% duty | | — | $t_{cyc} \times 0.5 + 15$ | ns | |
| Data setup time for CAS | | $t_{DS}$ | 0*5 | — | ns | 19.11, 19.13 |
| CAS setup time for RAS | | $t_{CSR}$ | 10 | — | ns | 19.16, 19.17, 19.18 |
| Row address hold time | | $t_{RAH}$ | 10 | — | ns | 19.11, 19.13 |
| Write command hold time | | $t_{WCH}$ | 15 | — | ns | |
| Write command setup time | 35% duty*2 | $t_{WCS}$ | 0 | — | ns | 19.11 |
| | 50% duty | | 0 | — | ns | |
| Access time from CAS precharge*6 | | $t_{ACP}$ | $t_{cyc}$ – 20 | — | ns | 19.12 |

Notes:
1. HBS and LBS signals are 25 ns.
2. When frequency is 10 MHz or more.
3. n is the number of wait cycles.
4. Access time from addresses A0 to A21 is tcyc-25.
5. –5 ns for parity output of DRAM long-pitch access.
6. It is not necessary to meet the $t_{RDS}$ specification as long as the access time specification is met.

**Table 19.7 Bus Timing (2)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| Address delay time | | $t_{AD}$ | — | 25*1 | ns | 19.8, 19.9, 19.11–19.14, 19.19, 19.20 |
| CS delay time 1 | | $t_{CSD1}$ | — | 30 | ns | 19.8, 19.9, 19.20 |
| CS delay time 2 | | $t_{CSD2}$ | — | 25 | ns | |
| CS delay time 3 | | $t_{CSD3}$ | — | 25 | ns | 19.19 |
| CS delay time 4 | | $t_{CSD4}$ | — | 25 | ns | |
| Access time 1*6 from read strobe | 35% duty*2 | $t_{RDAC1}$ | $t_{cyc} \times 0.65 – 20$ | — | ns | 19.8 |
| | 50% duty | | $t_{cyc} \times 0.5 – 20$ | — | ns | |
| Access time 2*6 from read strobe | 35% duty*2 | $t_{RDAC2}$ | $t_{cyc} \times (n + 1.65) – 20$*3 | — | ns | 19.9, 19.10 |
| | 50% duty | | $t_{cyc} \times (n + 1.5) – 20$*3 | — | ns | |
| Access time 3*6 from read strobe | 35% duty*1 | $t_{RDAC3}$ | $t_{cyc} \times (n + 0.65) – 20$*3 | — | ns | 19.19 |
| | 50% duty | | $t_{cyc} \times (n + 0.5) – 20$*3 | — | ns | |
| Read strobe delay time | | $t_{RSD}$ | — | 25 | ns | 19.8, 19.9, 19.19 |
| Read data setup time | | $t_{RDS}$ | 15 | — | ns | 19.8, 19.9, 19.11–19.14, |
| Read data hold time | | $t_{RDH}$ | 0 | — | ns | 19.19 |
| Write strobe delay time 1 | | $t_{WSD1}$ | — | 25 | ns | 19.9, 19.13, 19.14, 19.19, 19.20 |
| Write strobe delay time 2 | | $t_{WSD2}$ | — | 25 | ns | 19.9, 19.13, 19.14, 19.19 |
| Write strobe delay time 3 | | $t_{WSD3}$ | — | 25 | ns | 19.11, 19.12 |
| Write strobe delay time 4 | | $t_{WSD4}$ | — | 25 | ns | 19.11, 19.12, 19.20 |
| Write data delay time 1 | | $t_{WDD1}$ | — | 45 | ns | 19.9, 19.13, 19.14, 19.19 |
| Write data delay time 2 | | $t_{WDD2}$ | — | 25 | ns | 19.11, 19.12 |
| Write data hold time | | $t_{WDH}$ | 0 | — | ns | 19.9, 19.11–19.14 |
| Parity output delay time 1 | | $t_{WPDD1}$ | — | 45 | ns | 19.9, 19.13, 19.14 |
| Parity output delay time 2 | | $t_{WPDD2}$ | — | 25 | ns | 19.11, 19.12 |
| Parity output hold time | | $t_{WPDH}$ | 0 | — | ns | 19.9, 19.11–19.14 |

**Table 19.7 Bus Timing (2) (cont)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| Wait setup time | | $t_{WTS}$ | 19 | — | ns | 19.10, 19.15, 19.19 |
| Wait hold time | | $t_{WTH}$ | 10 | — | ns | |
| Read data access time 1*6 | | $t_{ACC1}$ | $t_{cyc}$ – 30*4 | — | ns | 19.8, 19.11, 19.12 |
| Read data access time 2*6 | | $t_{ACC2}$ | $t_{cyc} \times (n+2) – 30$*3 | — | ns | 19.9, 19.10, 19.13, 19.14 |
| RAS delay time 1 | | $t_{RASD1}$ | — | 25 | ns | 19.11–19.14, 19.16–19.18 |
| RAS delay time 2 | | $t_{RASD2}$ | — | 35 | ns | |
| CAS delay time 1 | | $t_{CASD1}$ | — | 25 | ns | 19.11 |
| CAS delay time 2 | | $t_{CASD2}$ | — | 25 | ns | 19.13, 19.14, 19.16–19.18 |
| CAS delay time 3 | | $t_{CASD3}$ | — | 25 | ns | |
| Column address setup time | | $t_{ASC}$ | 0 | — | ns | 19.11, 19.12 |
| Read data access time from CAS 1*6 | 35% duty*2 | $t_{CAC1}$ | $t_{cyc} \times 0.65 – 19$ | — | ns | |
| | 50% duty | | $t_{cyc} \times 0.5 – 19$ | — | ns | |
| Read data access time from CAS 2*6 | | $t_{CAC2}$ | $t_{cyc} \times (n + 1) – 25$*3 | — | ns | 19.13, 19.14, 19.15 |
| Read data access time from RAS 1*6 | | $t_{RAC1}$ | $t_{cyc} \times 1.5 – 20$ | — | ns | 19.11, 19.12 |
| Read data access time from RAS 2*6 | | $t_{RAC2}$ | $t_{cyc} \times (n + 2.5) – 20$*3 | — | ns | 19.13, 19.14, 19.15 |
| High-speed page mode CAS precharge time | | $t_{CP}$ | $t_{cyc} \times 0.25$ | — | ns | 19.12 |
| AH delay time 1 | | $t_{AHD1}$ | — | 25 | ns | 19.19 |
| AH delay time 2 | | $t_{AHD2}$ | — | 25 | ns | |
| Multiplexed address delay time | | $t_{MAD}$ | — | 30 | ns | |
| Multiplexed address hold time | | $t_{MAH}$ | 0 | — | ns | |

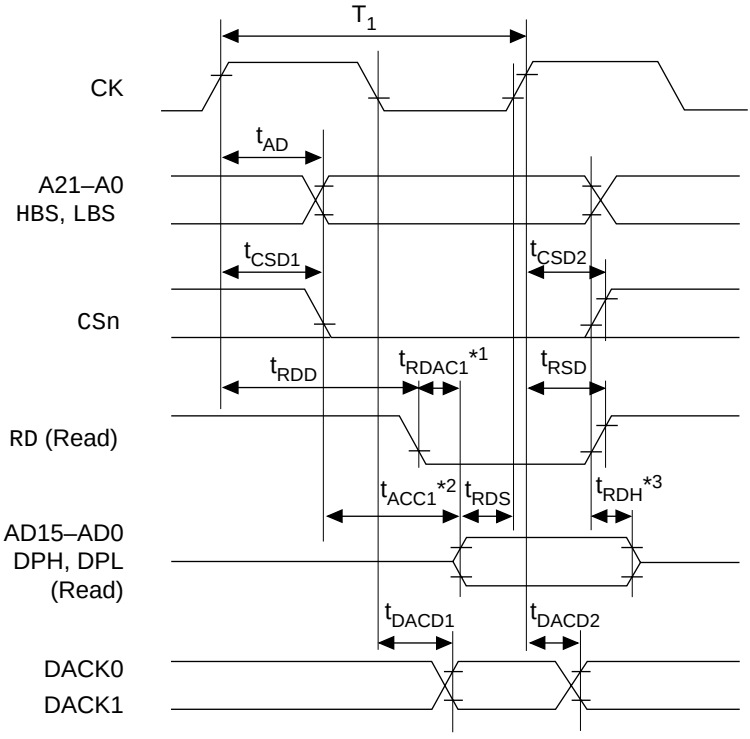
**Table 19.7 Bus Timing (2) (cont)**

Conditions: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| DACK0, DACK1 delay time 1 | | $t_{DACD1}$ | — | 25 | ns | 19.8, 19.9, 19.11–19.14, 19.19, 19.20 |
| DACK0, DACK1 delay time 2 | | $t_{DACD2}$ | — | 25 | ns | |
| DACK0, DACK1 delay time 3 | | $t_{DACD3}$ | — | 25 | ns | 19.9, 19.13, 19.14, 19.19 |
| DACK0, DACK1 delay time 4 | | $t_{DACD4}$ | — | 25 | ns | 19.11, 19.12 |
| DACK0, DACK1 delay time 5 | | $t_{DACD5}$ | — | 25 | ns | |
| Read delay time | 35% duty*2 | $t_{RDD}$ | — | $t_{cyc} \times 0.35 + 12$ | ns | 19.8, 19.9, 19.11-19.15, 19.19 |
| | 50% duty | | — | $t_{cyc} \times 0.5 + 15$ | ns | |
| Data setup time for CAS | | $t_{DS}$ | 0*5 | — | ns | 19.11, 19.13 |
| CAS setup time for RAS | | $t_{CSR}$ | 10 | — | ns | 19.16, 19.17, 19.18 |
| Row address hold time | | $t_{RAH}$ | 10 | — | ns | 19.11, 19.13 |
| Write command hold time | | $t_{WCH}$ | 15 | — | ns | |
| Write command setup time | 35% duty*2 | $t_{WCS}$ | 0 | — | ns | 19.11 |
| | 50% duty | | 0 | — | ns | |
| Access time from CAS precharge*6 | | $t_{ACP}$ | $t_{cyc}$ –20 | — | ns | 19.12 |

Notes
1. HBS and LBS signals are 30 ns.
2. When frequency is 10 MHz or more
3. n is the number of wait cycles.
4. Access time from addresses A0 to A21 is tcyc-25.
5. –5 ns for parity output of DRAM long-pitch access
6. It is not necessary to meet the $t_{RDS}$ specification as long as the access time specification is met.

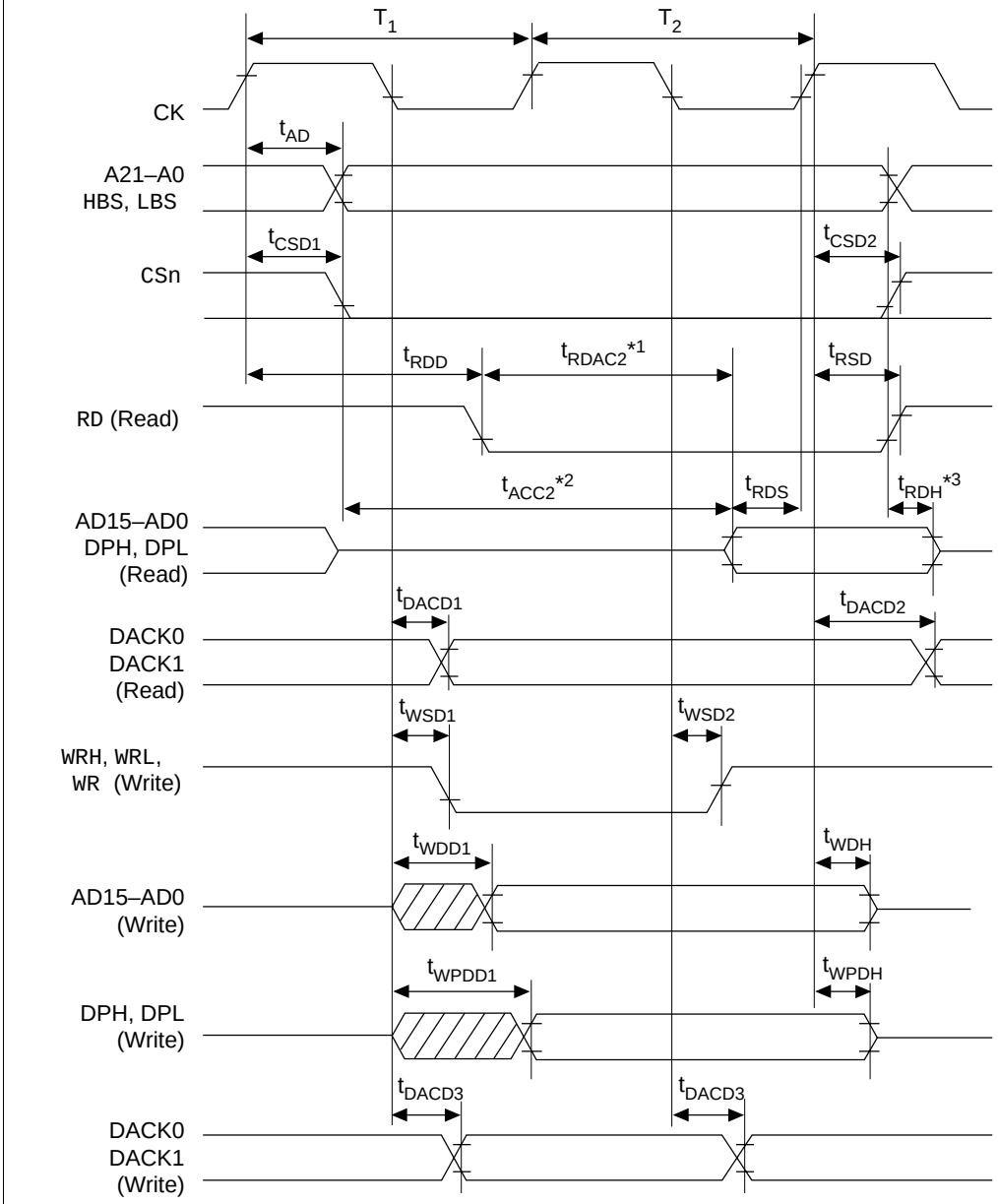



Notes: 1. For $t_{RDAC1}$, use $t_{cyc} \times 0.65 - 20$ (for 35% duty) or $t_{cyc} \times 0.5 - 20$ (for 50% duty) instead of $t_{cyc} - t_{RDD} - t_{RDS}$.

2. For $t_{ACC1}$, use $t_{cyc} - 30$ instead of $t_{cyc} - t_{AD}$ (or $t_{CSD1}$) – $t_{RDS}$.

3. $t_{RDH}$ is measured from A21–A0, CSn, or RD, whichever is negated first.

**Figure 19.8 Basic Bus Cycle: One-State Access**

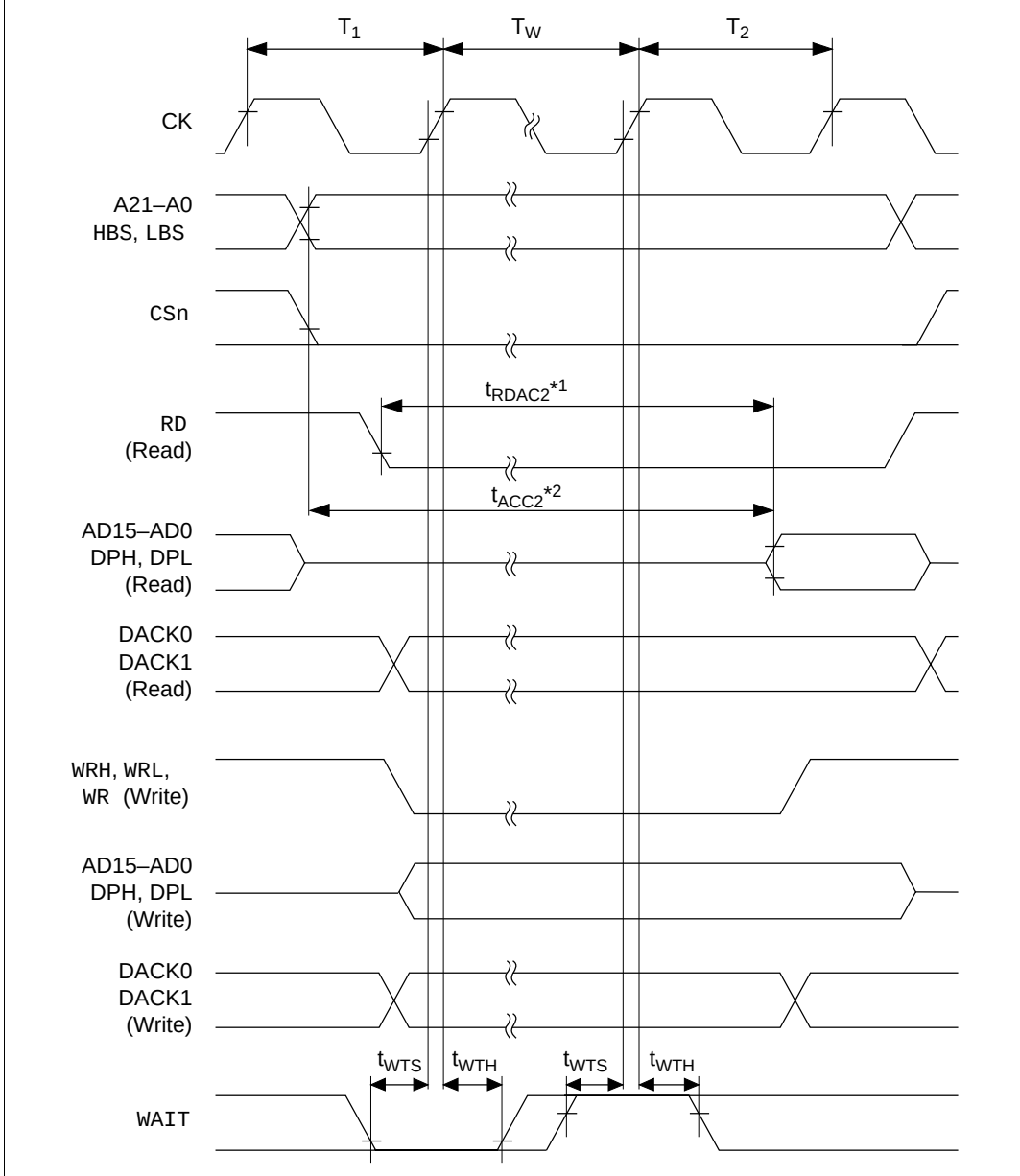



Notes: 1 For $t_{RDAC2}$, use $t_{cyc} \times (n + 1.65) - 20$ (for 35% duty) or $t_{cyc} \times (n + 1.5) - 20$ (for 50% duty) instead of $t_{cyc} \times (n + 2) - t_{RDD} - t_{RDS}$.

2 For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 30$ instead of $t_{cyc} \times (n + 2) - t_{AD}$ (or $t_{CSD1}$) – $t_{RDS}$.

3 $t_{RDH}$ is measured from A21–A0, CSn, or RD, whichever is negated first.

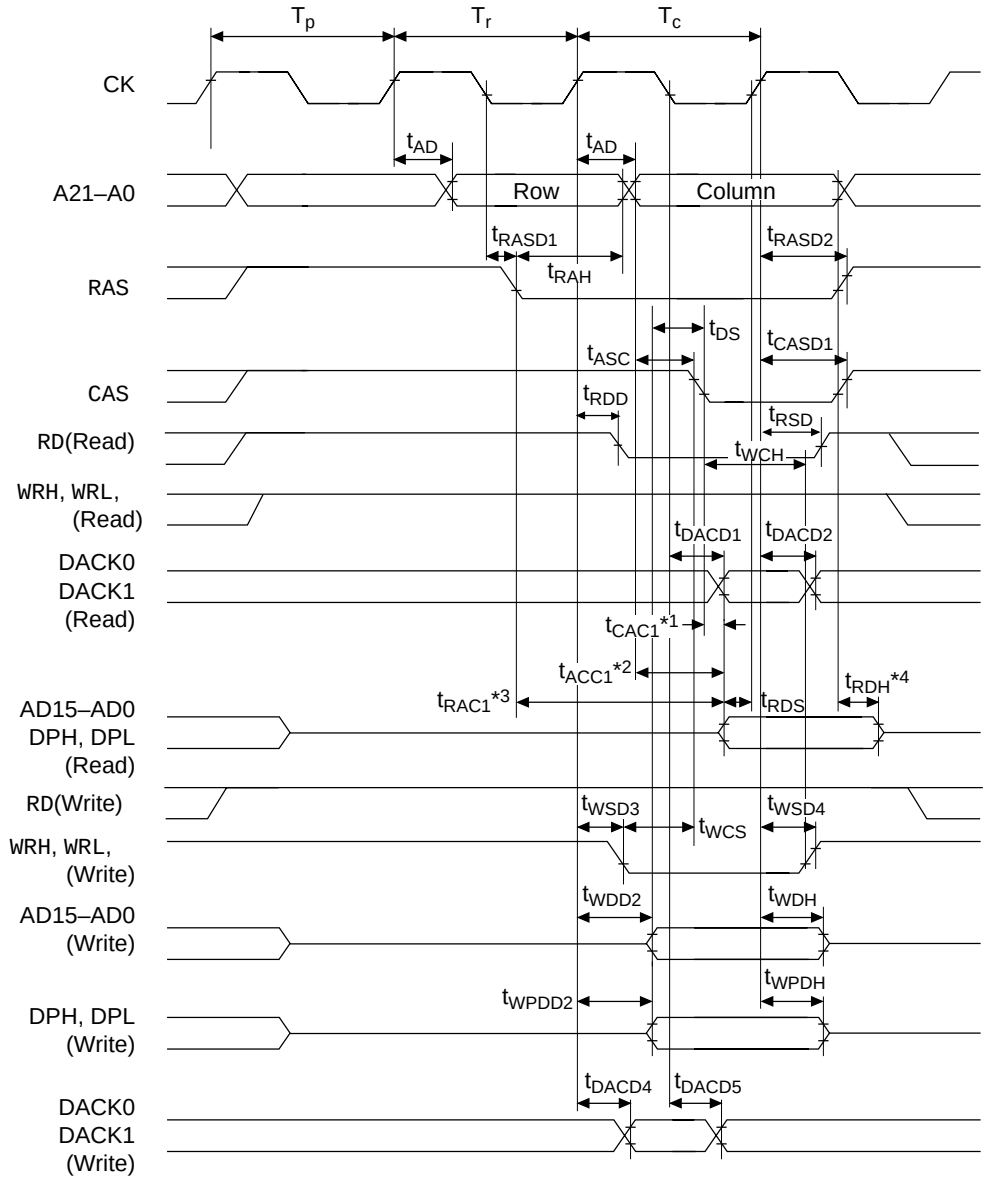
**Figure 19.9 Basic Bus Cycle: Two-State Access**




Notes: 1. For $t_{RDAC2}$, use $t_{cyc} \times (n + 1.65) - 20$ (for 35% duty) or $t_{cyc} \times (n + 1.5) - 20$ (for 50% duty) instead of $t_{cyc} \times (n + 2) - t_{RDD} - t_{RDS}$.

2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 30$ instead of $t_{cyc} \times (n + 2) - t_{AD}$ (or $t_{CSD1}$) $- t_{RDS}$.

**Figure 19.10 Basic Bus Cycle: Two States + Wait State**



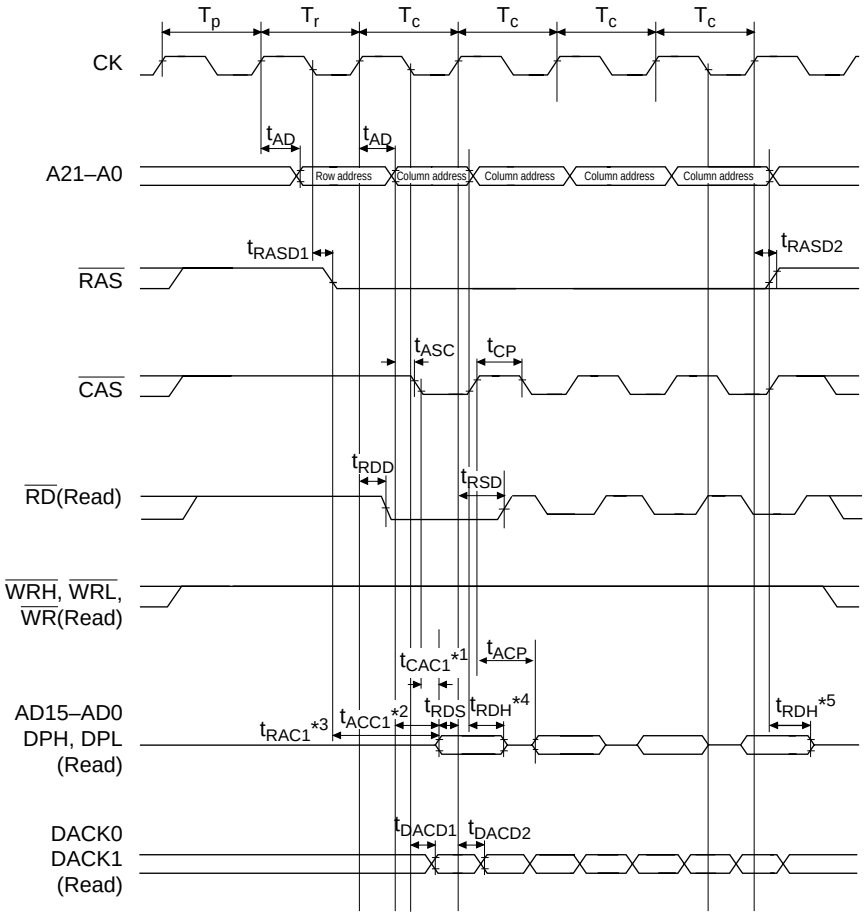

Notes: 1. For $t_{CAC1}$, use $t_{cyc} \times 0.65 - 19$ (for 35% duty) or $t_{cyc} \times 0.5 - 19$ (for 50% duty) instead of $t_{cyc} - t_{AD} - t_{ASC} - t_{RDS}$.

2. For $t_{ACC1}$, use $t_{cyc} - 30$ instead of $t_{cyc} - t_{AD} - t_{RDS}$.

3. For $t_{RAC1}$, use $t_{cyc} \times 1.5 - 20$ instead of $t_{cyc} \times 1.5 - t_{RASD1} - t_{RDS}$.

4. $t_{RDH}$ is measured from A21–A0, RAS, or CAS, whichever is negated first.

**Figure 19.11 DRAM Bus Cycle (Short Pitch, Normal Mode)**




Notes:
1. For $t_{CAC1}$, use $t_{cyc} \times 0.65 - 19$ (for 35% duty) or $t_{cyc} \times 0.5 - 19$ (for 50% duty) instead of $t_{cyc} - t_{AD} - t_{ASC} - t_{RDS}$.
   It is not necessary to meet the $t_{RDS}$ specification as long as the $t_{CAC1}$ specification is met.
2. For $t_{ACC1}$, use $t_{cyc} - 30$ instead of $t_{cyc} - t_{AD} - t_{RDS}$.
   It is not necessary to meet the $t_{RDS}$ specification as long as the $t_{ACC1}$ specification is met.
3. For $t_{RAC1}$, use $t_{cyc} \times 1.5 - 20$ instead of $t_{cyc} \times 1.5 - t_{RASD1} - t_{RDS}$.
   It is not necessary to meet the $t_{RDS}$ specification as long as the $t_{RAC1}$ specification is met.
4. $t_{RDH}$ is measured from A21—A0 or $\overline{CAS}$, whichever is negated first.
5. $t_{RDH}$ is measured from A21—A0, $\overline{RAS}$, or $\overline{CAS}$, whichever is negated first.

**Figure 19.12 (a) DRAM Bus Cycle (Short-Pitch, High-Speed Page Mode: Read)**

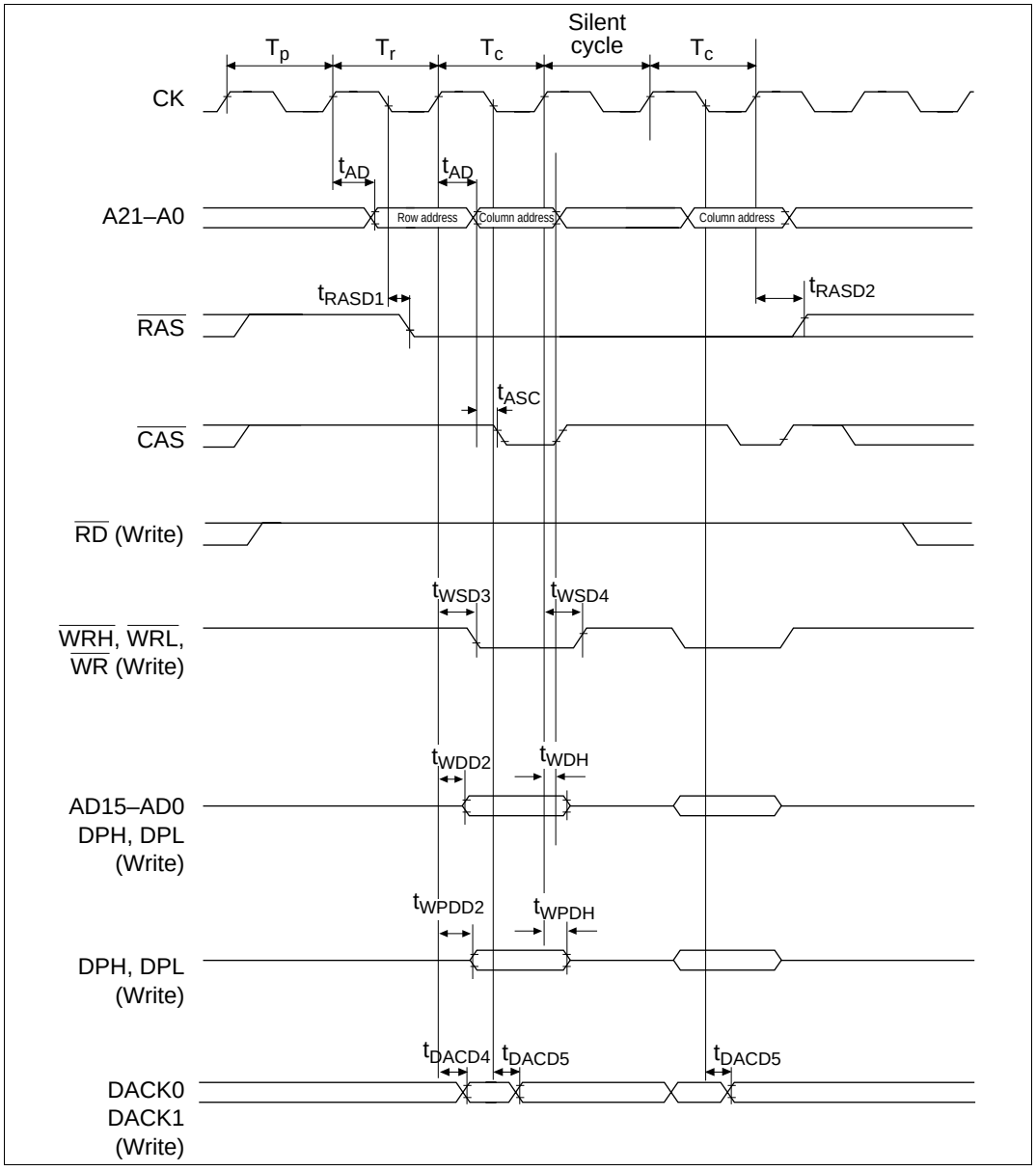
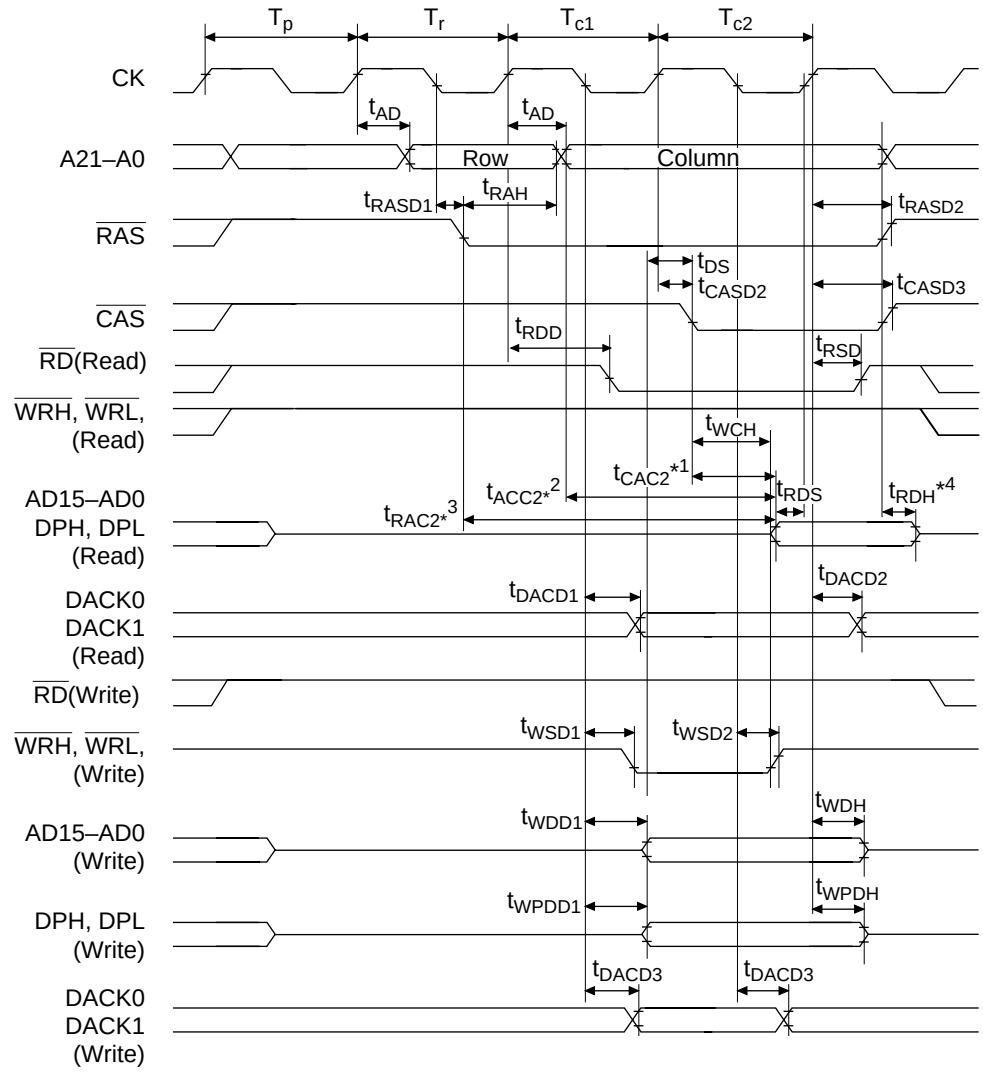



**Figure 19.12 (b) DRAM Bus Cycle (Short-Pitch, High-Speed Page Mode: Write)**

Note: For details of the silent cycle, see section 8.5.5, Burst Operation.



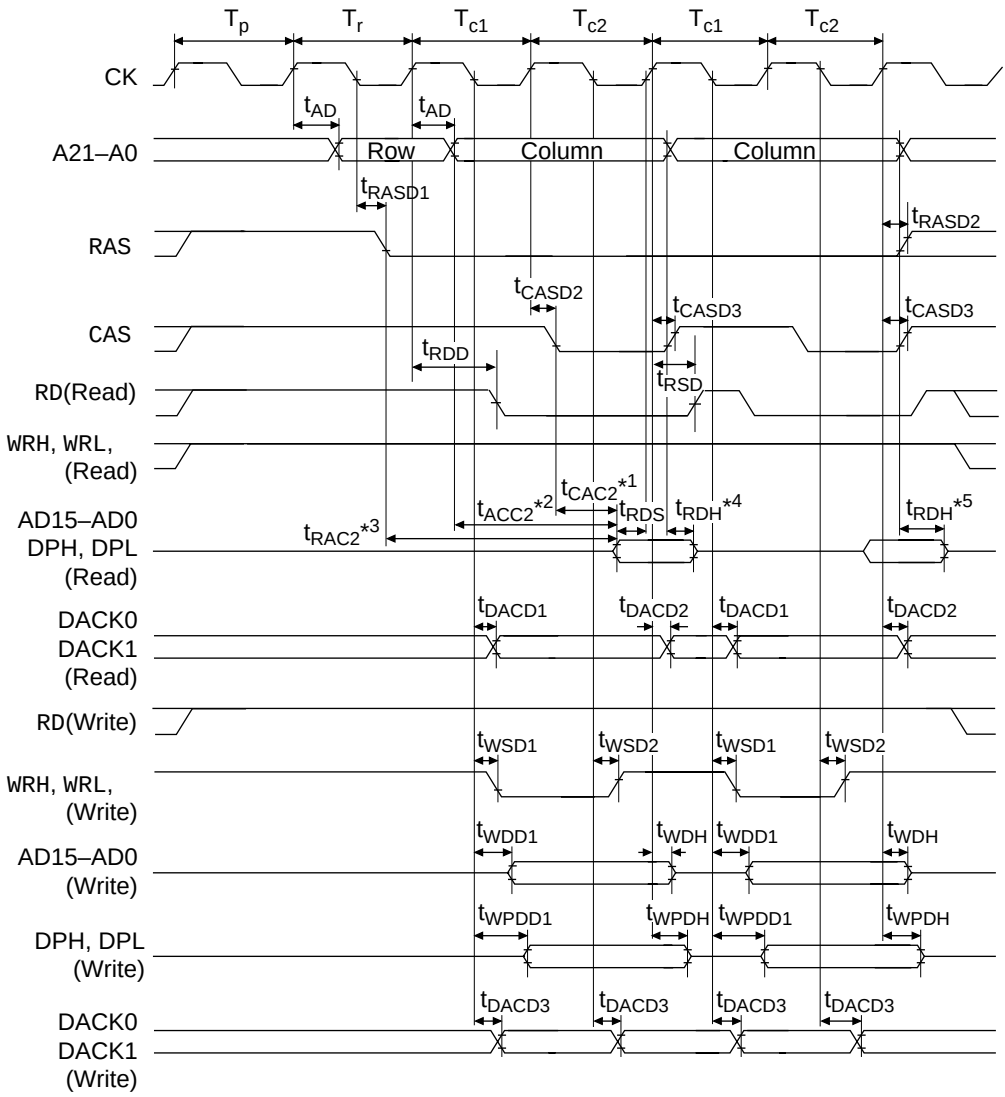

Notes: 1. For $t_{CAC2}$, use $t_{cyc} \times (n + 1) - 25$ instead of $t_{cyc} \times (n + 1) - t_{CASD2} - t_{RDS}$.

2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 30$ instead of $t_{cyc} \times (n + 2) - t_{AD} - t_{RDS}$.

3. For $t_{RAC2}$, use $t_{cyc} \times (n + 2.5) - 20$ instead of $t_{cyc} \times (n + 2.5) - t_{RASD1} - t_{RDS}$.

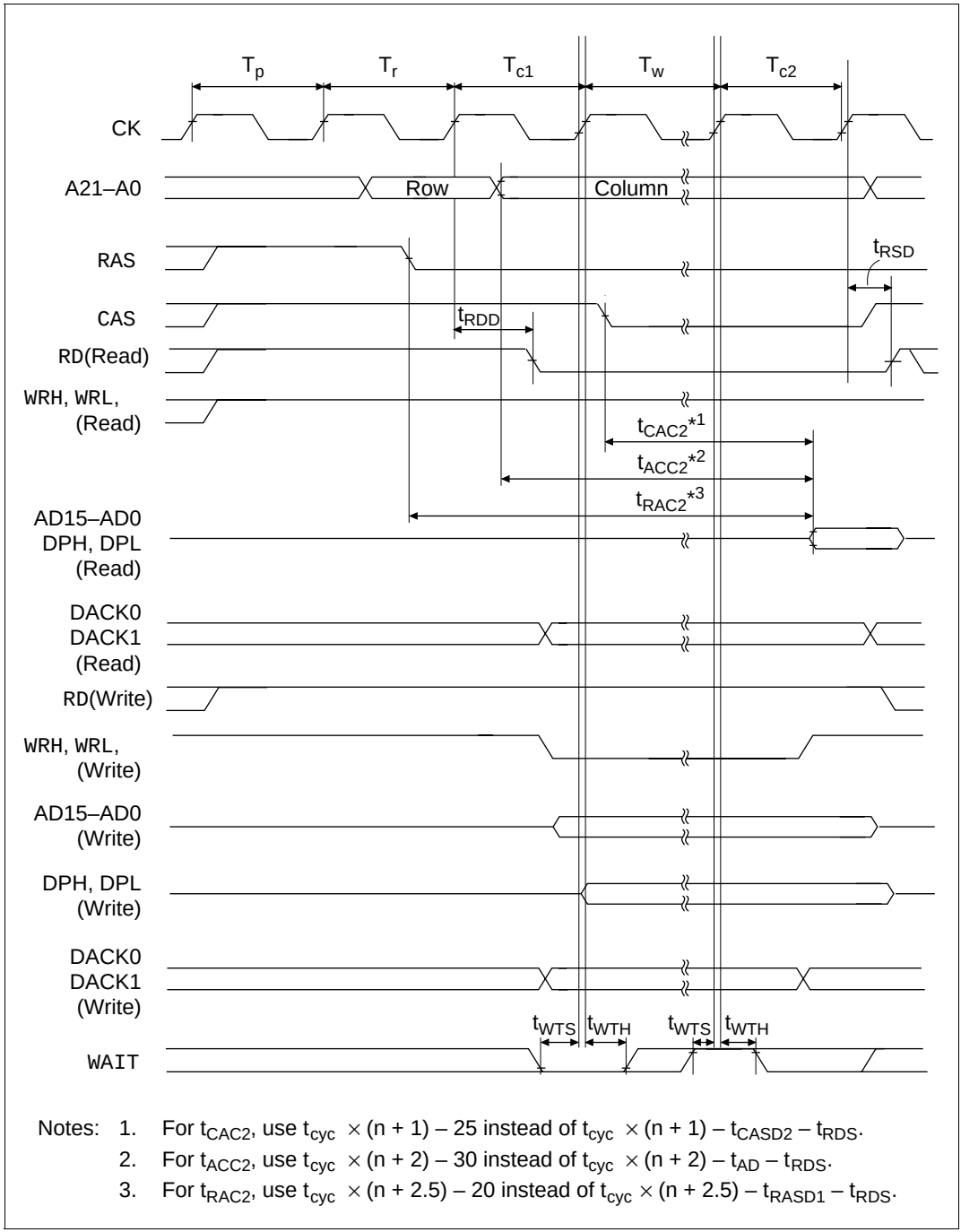
4. $t_{RDH}$ is measured from A21–A0, CAS, or RAS, whichever is negated first.

**Figure 19.13 DRAM Bus Cycle: (Long Pitch, Normal Mode)**




Notes:
1. For $t_{CAC2}$, use $t_{cyc} \times (n + 1) - 25$ instead of $t_{cyc} \times (n + 1) - t_{CASD2} - t_{RDS}$.
2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 30$ instead of $t_{cyc} \times (n + 2) - t_{AD} - t_{RDS}$.
3. For $t_{RAC2}$, use $t_{cyc} \times (n + 2.5) - 20$ instead of $t_{cyc} \times (n + 2.5) - t_{RASD1} - t_{RDS}$.
4. $t_{RDH}$ is measured from A21–A0 or CAS, whichever is negated first.
5. $t_{RDH}$ is measured from A21–A0, RAS, or CAS whichever is negated first.

**Figure 19.14 DRAM Bus Cycle: (Long Pitch, High-Speed Page Mode)**




Notes: 1. For $t_{CAC2}$, use $t_{cyc} \times (n + 1) - 25$ instead of $t_{cyc} \times (n + 1) - t_{CASD2} - t_{RDS}$.
2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 30$ instead of $t_{cyc} \times (n + 2) - t_{AD} - t_{RDS}$.
3. For $t_{RAC2}$, use $t_{cyc} \times (n + 2.5) - 20$ instead of $t_{cyc} \times (n + 2.5) - t_{RASD1} - t_{RDS}$.

**Figure 19.15 DRAM Bus Cycle: (Long Pitch, High-Speed Page Mode + Wait State)**

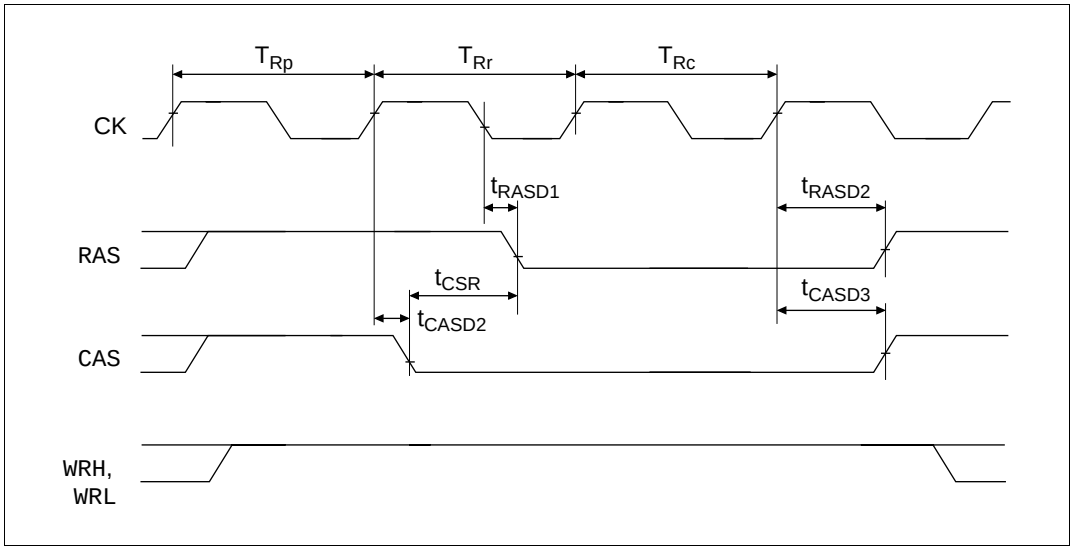



**Figure 19.16 CAS-before-RAS Refresh (Short Pitch)**

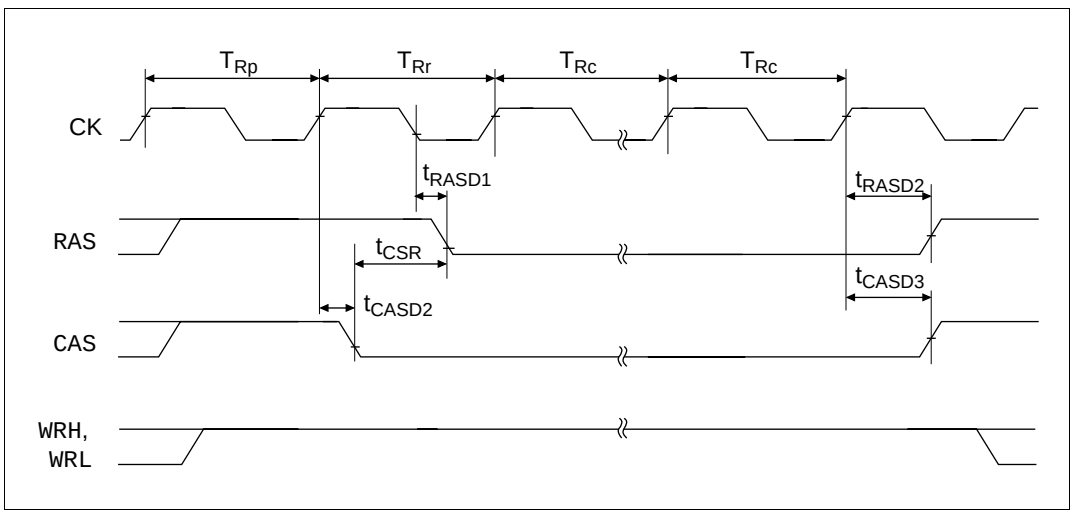



**Figure 19.17 CAS-before-RAS Refresh (Long Pitch)**

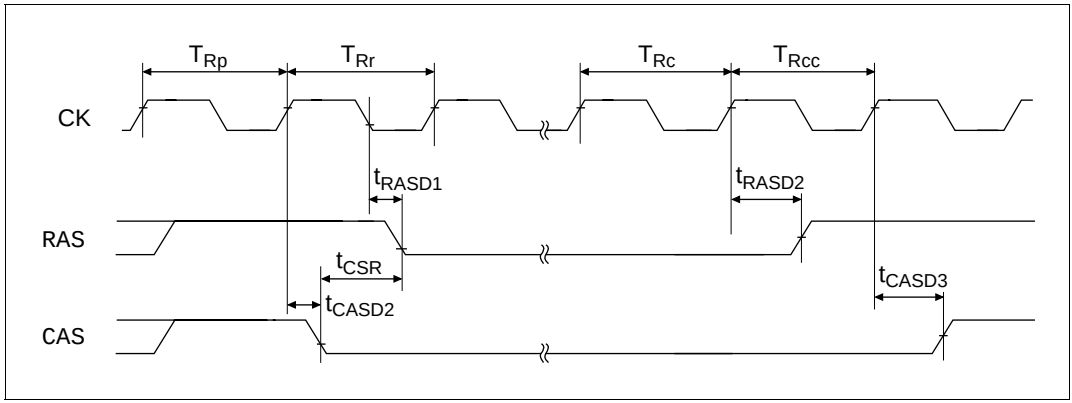



**Figure 19.18 Self Refresh**

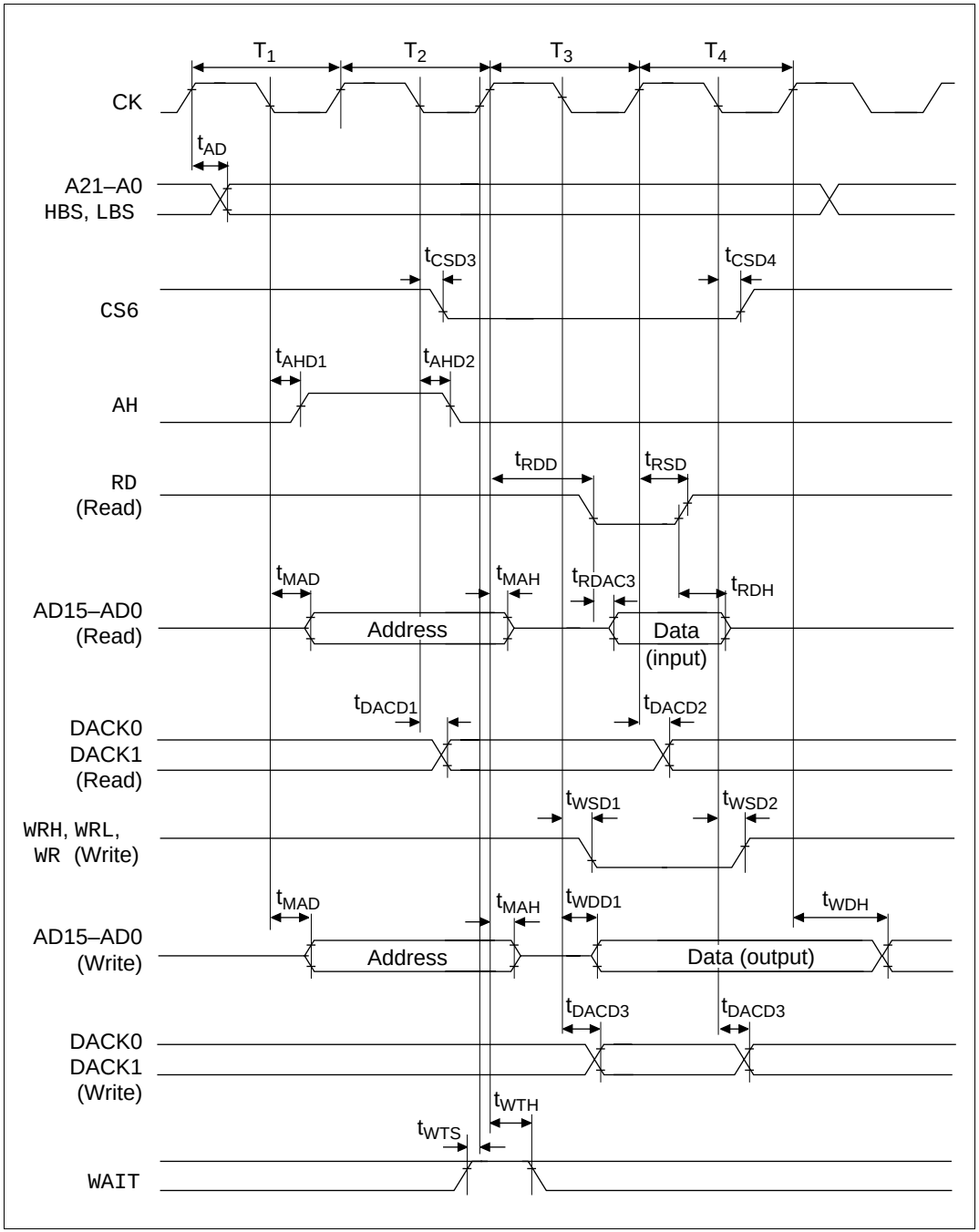



**Figure 19.19 Address/Data Multiplex I/O Bus Cycle**

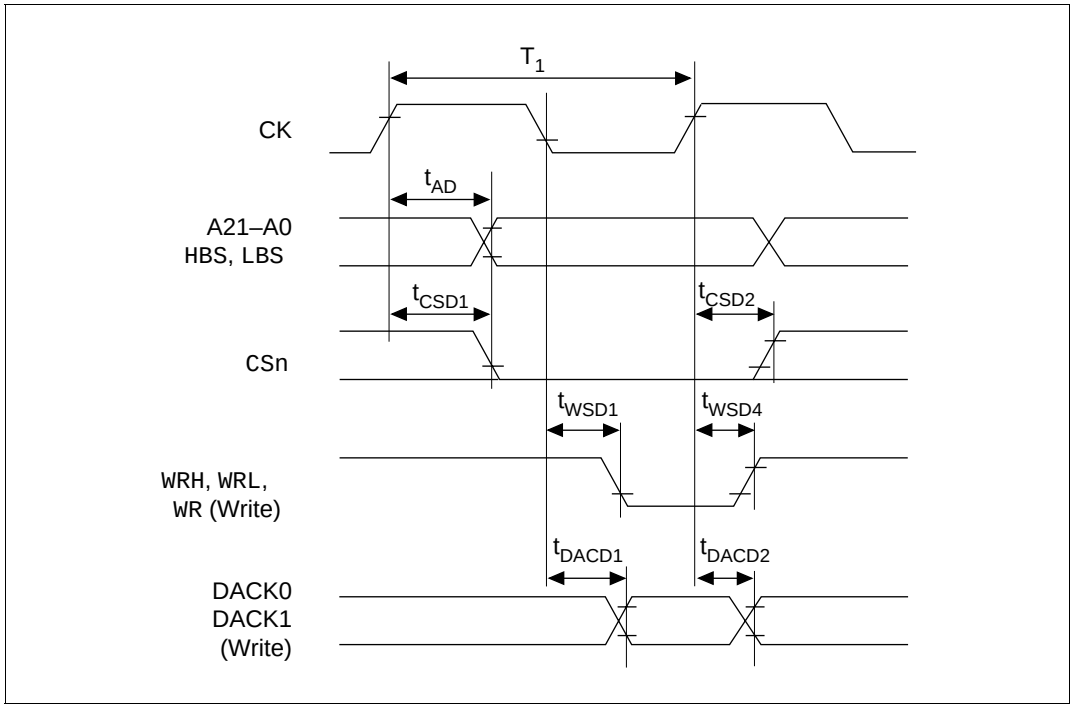



**Figure 19.20 DMA Single Transfer/1 State Access Write**

**Table 19.8 Bus Timing (3)**

Conditions: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| Address delay time | | $t_{AD}$ | — | 40 | ns | 19.21, 19.22, 19.24–19.27, 19.32, 19.33 |
| CS delay time 1 | | $t_{CSD1}$ | — | 40 | ns | 19.21, 19.22, 19.33 |
| CS delay time 2 | | $t_{CSD2}$ | — | 40 | ns | |
| CS delay time 3 | | $t_{CSD3}$ | — | 40 | ns | 19.32 |
| CS delay time 4 | | $t_{CSD4}$ | — | 40 | ns | |
| Access time 1[*4] from read strobe | 35% duty[*1] | $t_{RDAC1}$ | $t_{cyc} \times 0.65 - 35$ | — | ns | 19.21, |
| | 50% duty | | $t_{cyc} \times 0.5 - 35$ | — | ns | |
| Access time 2[*4] from read strobe | 35% duty[*1] | $t_{RDAC2}$ | $t_{cyc} \times (n+1.65) - 35$[*2] | — | ns | 19.22, 19.23 |
| | 50% duty | | $t_{cyc} \times (n+1.5) - 35$[*2] | — | ns | |
| Access time 3[*4] from read strobe | 35% duty[*1] | $t_{RDAC3}$ | $t_{cyc} \times (n+0.65) - 35$[*2] | — | ns | 19.32 |
| | 50% duty | | $t_{cyc} \times (n+0.5) - 35$[*2] | — | ns | |
| Read strobe delay time | | $t_{RSD}$ | — | 40 | ns | 19.21, 19.22, 19.32 |
| Read data set-up time | | $t_{RDS}$ | 30 | — | ns | 19.21, 19.22, |
| Read data hold time | | $t_{RDH}$ | 0 | — | ns | 19.24-19.27, 19.32 |
| Write strobe delay time 1 | | $t_{WSD1}$ | — | 40 | ns | 19.22, 19.26, 19.27, 19.32, 19.33 |
| Write strobe delay time 2 | | $t_{WSD2}$ | — | 30 | ns | 19.22, 19.26, 19.27, 19.32 |
| Write strobe delay time 3 | | $t_{WSD3}$ | — | 40 | ns | 19.24, 19.25 |
| Write strobe delay time 4 | | $t_{WSD4}$ | — | 40 | ns | 19.24, 19.25, 19.33 |
| Write data delay time 1 | | $t_{WDD1}$ | — | 70 | ns | 19.22, 19.26, 19.27, 19.32 |
| Write data delay time 2 | | $t_{WDD2}$ | — | 40 | ns | 19.24, 19.25 |
| Write data hold time | | $t_{WDH}$ | –10 | — | ns | 19.22, 19.24–19.27, 19.32 |
| Parity output delay time 1 | | $t_{WPDD1}$ | — | 80 | ns | 19.22, 19.24, 19.27 |
| Parity output delay time 2 | | $t_{WPDD2}$ | — | 40 | ns | 19.24, 19.25 |
| Parity output hold time | | $t_{WPDH}$ | –10 | — | ns | 19.22, 19.24–19.27 |

**Table 19.8 Bus Timing (3) (cont)**

Conditions: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| Wait setup time | | $t_{WTS}$ | 40 | — | ns | 19.23, 19.28, 19.32 |
| Wait hold time | | $t_{WTH}$ | 10 | — | ns | |
| Read data access time 1[*4] | | $t_{ACC1}$ | $t_{cyc}$ – 44 | — | ns | 19.21, 19.24, 19.25 |
| Read data access time 2[*4] | | $t_{ACC2}$ | $t_{cyc} \times (n+2) – 44$[*2] | — | ns | 19.22, 19.23, 19.26, 19.28 |
| RAS delay time 1 | | $t_{RASD1}$ | — | 40 | ns | 19.24–19.27, 19.29–19.31 |
| RAS delay time 2 | | $t_{RASD2}$ | — | 40 | ns | |
| CAS delay time 1 | | $t_{CASD1}$ | — | 40 | ns | 19.24 |
| CAS delay time 2 | | $t_{CASD2}$ | — | 40 | ns | 19.26, 19.27, 19.29–19.31 |
| CAS delay time 3 | | $t_{CASD3}$ | — | 40 | ns | |
| Column address setup time | | $t_{ASC}$ | 0 | — | ns | 19.24, 19.25 |
| Read data access time from CAS 1[*4] | 35% duty[*1] | $t_{CAC1}$ | $t_{cyc} \times 0.65 – 35$ | — | ns | |
| | 50% duty | | $t_{cyc} \times 0.5 – 35$ | — | ns | |
| Read data access time from CAS 2[*4] | | $t_{CAC2}$ | $t_{cyc} \times (n+1) – 35$[*2] | — | ns | 19.26, 19.27, 19.28 |
| Read data access time from RAS 1[*4] | | $t_{RAC1}$ | $t_{cyc} \times 1.5 – 35$ | — | ns | 19.24, 19.25 |
| Read data access time from RAS 2[*4] | | $t_{RAC2}$ | $t_{cyc} \times (n+2.5) – 35$[*2] | — | ns | 19.26, 19.27, 19.28 |
| High-speed page mode CAS precharge time | | $t_{CP}$ | $t_{cyc} \times 0.25$ | — | ns | 19.25 |
| AH delay time 1 | | $t_{AHD1}$ | — | 40 | ns | 19.32 |
| AH delay time 2 | | $t_{AHD2}$ | — | 40 | ns | |
| Multiplexed address delay time | | $t_{MAD}$ | — | 40 | ns | |
| Multiplexed address hold time | | $t_{MAH}$ | –10 | — | ns | |

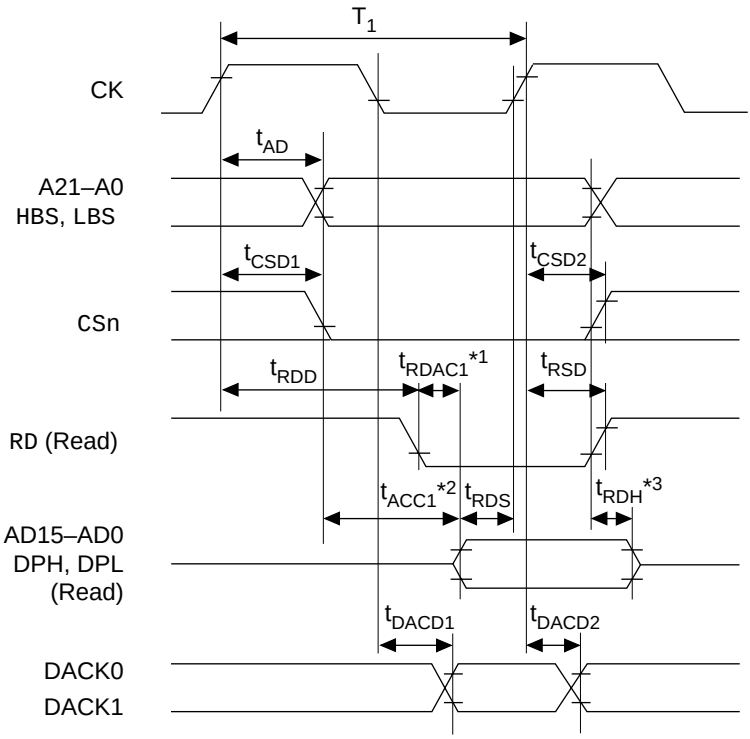
**Table 19.8 Bus Timing (3) (cont)**

Conditions: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products

| Item | | Symbol | Min | Max | Unit | Figures |
|---|---|---|---|---|---|---|
| DACK0, DACK1 delay time 1 | | $t_{DACD1}$ | — | 40 | ns | 19.21, 19.22, 19.24–19.27, 19.32, 19.33 |
| DACK0, DACK1 delay time 2 | | $t_{DACD2}$ | — | 40 | ns | |
| DACK0, DACK1 delay time 3 | | $t_{DACD3}$ | — | 40 | ns | 19.22, 19.26, 19.27, 19.32 |
| DACK0, DACK1 delay time 4 | | $t_{DACD4}$ | — | 40 | ns | 19.24, 19.25 |
| DACK0, DACK1 delay time 5 | | $t_{DACD5}$ | — | 40 | ns | |
| Read delay time | 35% duty*1 | $t_{RDD}$ | — | $t_{cyc} \times 0.35 + 35$ | ns | 19.21, 19.22, 19.24-19.28, 19.32 |
| | 50% duty | | — | $t_{cyc} \times 0.5 + 35$ | ns | |
| Data setup time for CAS | | $t_{DS}$ | 0*3 | — | ns | 19.24, 19.26 |
| CAS setup time for RAS | | $t_{CSR}$ | 10 | — | ns | 19.29–19.31 |
| Row address hold time | | $t_{RAH}$ | 10 | — | ns | 19.24, 19.26 |
| Write command hold time | | $t_{WCH}$ | 15 | — | ns | |
| Write command setup time | 35% duty*1 | $t_{WCS}$ | 0 | — | ns | 19.24 |
| | 50% duty | $t_{WCS}$ | 0 | — | ns | |
| Access time from CAS precharge*4 | | $t_{ACP}$ | tcyc -20 | — | ns | 19.25 |

Notes: 1. When frequency is 10 MHz or more.
2. n is the number of wait cycles.
3. –5 ns for parity output of DRAM long-pitch access
4. It is not necessary to meet the $t_{RDS}$ specification as long as the access time specification is met.

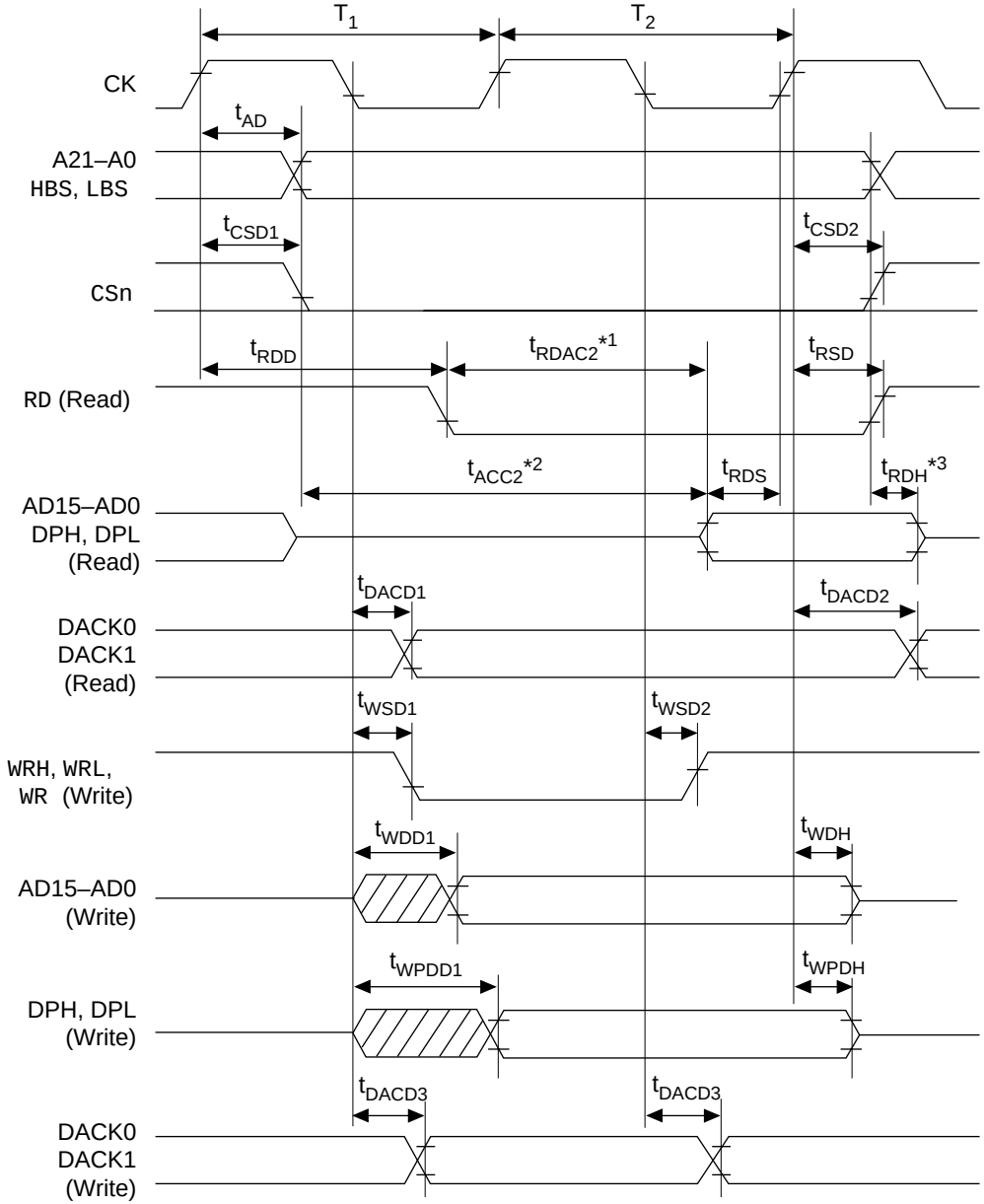



Notes: 1. For $t_{RDAC1}$, use $t_{cyc} \times 0.65 - 35$ (for 35% duty) or $t_{cyc} \times 0.5 - 35$ (for 50% duty) instead of $t_{cyc} - t_{RDD} - t_{RDS}$.

2. For $t_{ACC1}$, use $t_{cyc} - 44$ instead of $t_{cyc} - t_{AD}$ (or $t_{CSD1}$) – $t_{RDS}$.

3. $t_{RDH}$ is measured from A21–A0, CSn, or RD, whichever is negated first.

**Figure 19.21 Basic Bus Cycle: One-State Access**

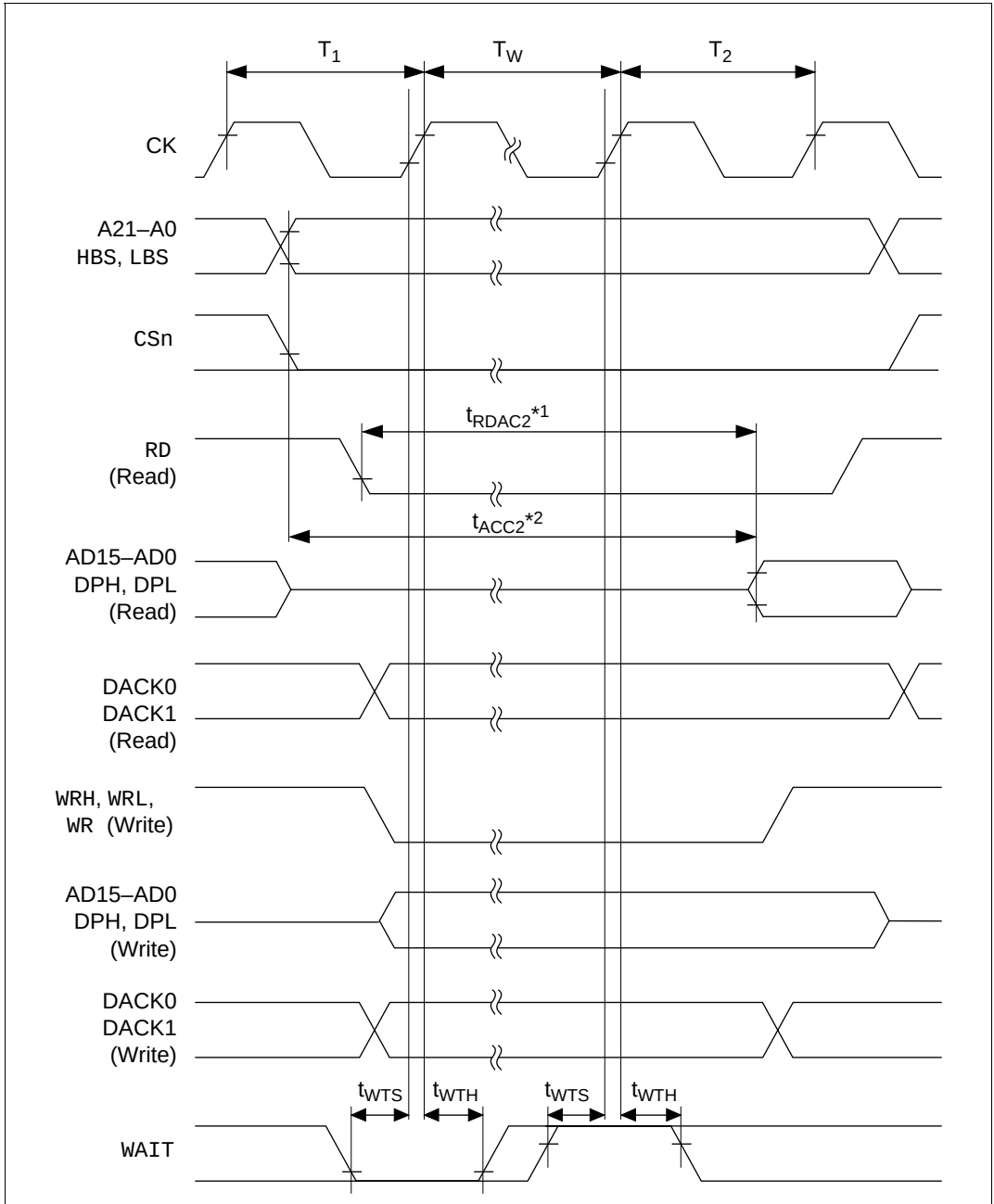



Notes: 1. For $t_{RDAC2}$, use $t_{cyc} \times (n + 1.65) - 35$ (for 35% duty) or $t_{cyc} \times (n + 1.5) - 35$ (for 50% duty) instead of $t_{cyc} \times (n + 2) - t_{RDD} - t_{RDS}$.

2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 44$ instead of $t_{cyc} \times (n + 2) - t_{AD}$ (or $t_{CSD1}$) $- t_{RDS}$.

3. $t_{RDH}$ is measured from A21–A0, CSn, or RD, whichever is negated first.

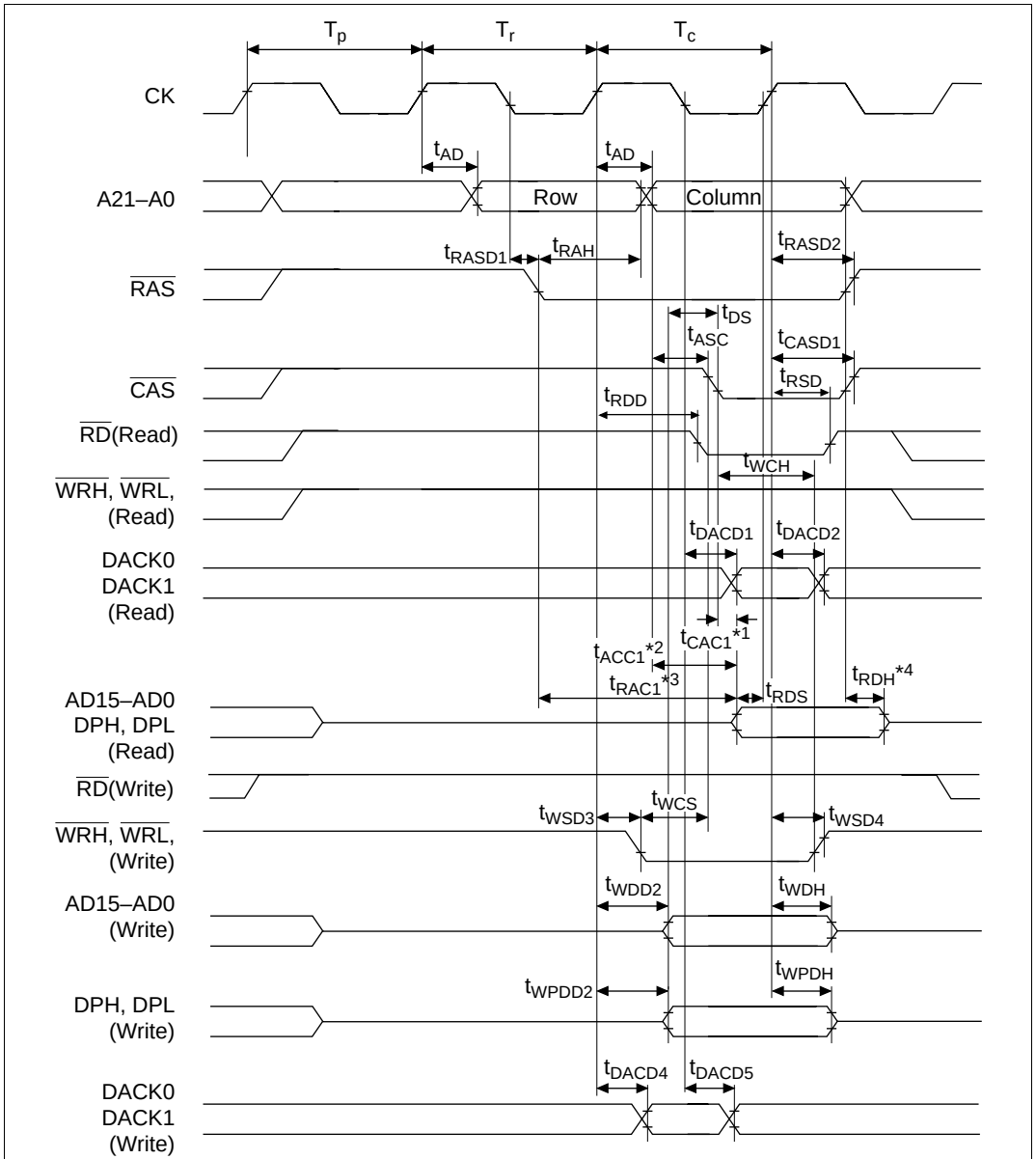
**Figure 19.22 Basic Bus Cycle: Two-State Access**




Notes: 1. For $t_{RDAC2}$, use $t_{cyc} \times (n + 1.65) - 35$ (for 35% duty) or $t_{cyc} \times (n + 1.5) - 35$ (for 50% duty) instead of $t_{cyc} \times (n + 2) - t_{RDD} - t_{RDS}$.

2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 44$ instead of $t_{cyc} \times (n + 2) - t_{AD}$ (or $t_{CSD1}$) $- t_{RDS}$.

**Figure 19.23 Basic Bus Cycle: Two States + Wait State**



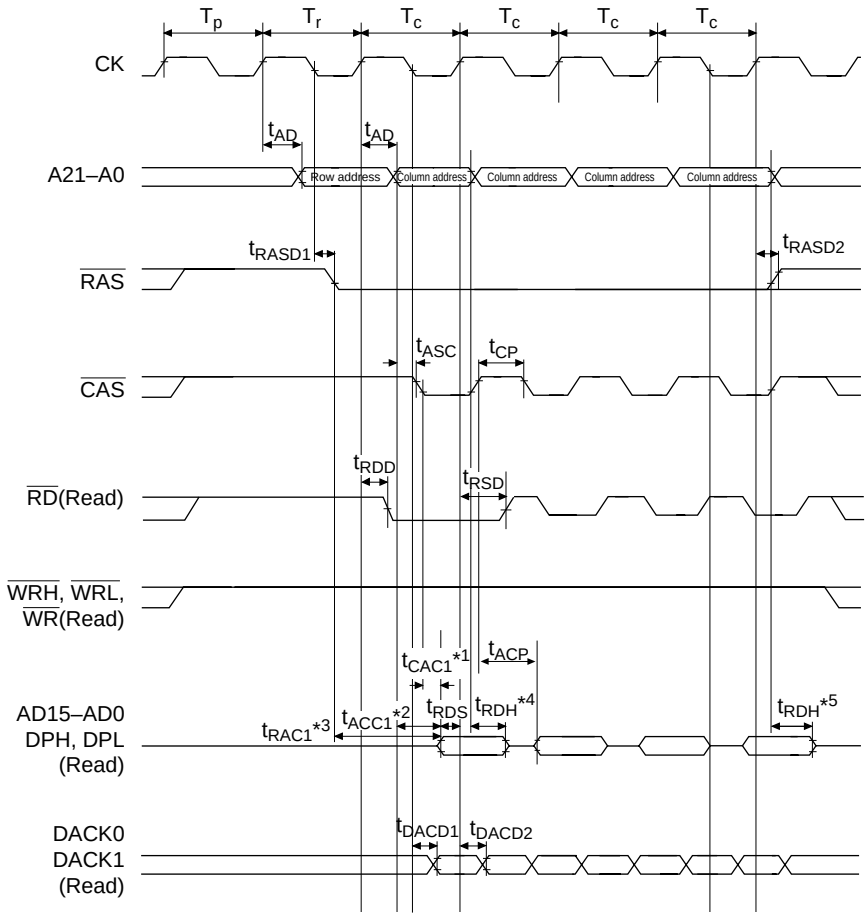

Notes: 1. For $t_{CAC1}$, use $t_{cyc} \times 0.65 - 35$ (for 35% duty) or $t_{cyc} \times 0.5 - 35$ (for 50% duty) instead of $t_{cyc} - t_{AD} - t_{ASC} - t_{RDS}$.

2. For $t_{ACC1}$, use $t_{cyc} - 44$ instead of $t_{cyc} - t_{AD} - t_{RDS}$.

3. For $t_{RAC1}$, use $t_{cyc} \times 1.5 - 35$ instead of $t_{cyc} \times 1.5 - t_{RASD1} - t_{RDS}$.

4. $t_{RDH}$ is measured from A21–A0, RAS, or CAS, whichever is negated first.

**Figure 19.24 DRAM Bus Cycle (Short Pitch, Normal Mode)**




Notes:
1. For $t_{CAC1}$, use $t_{cyc} \times 0.65 - 35$ (for 35% duty) or $t_{cyc} \times 0.5 - 35$ (for 50% duty) instead of $t_{cyc} - t_{AD} - t_{ASC} - t_{RDS}$.
   It is not necessary to meet the $t_{RDS}$ specification as long as the $t_{CAC1}$ specification is met.
2. For $t_{ACC1}$, use $t_{cyc} - 44$ instead of $t_{cyc} - t_{AD} - t_{RDS}$.
   It is not necessary to meet the $t_{RDS}$ specification as long as the $t_{ACC1}$ specification is met.
3. For $t_{RAC1}$, use $t_{cyc} \times 1.5 - 35$ instead of $t_{cyc} \times 1.5 - t_{RASD1} - t_{RDS}$.
   It is not necessary to meet the $t_{RDS}$ specification as long as the $t_{RAC1}$ specification is met.
4. $t_{RDH}$ is measured from A21—A0 or $\overline{CAS}$, whichever is negated first.
5. $t_{RDH}$ is measured from A21—A0, $\overline{RAS}$, or $\overline{CAS}$, whichever is negated first.

**Figure 19.25 (a) DRAM Bus Cycle (Short-Pitch, High-Speed Page Mode: Read)**

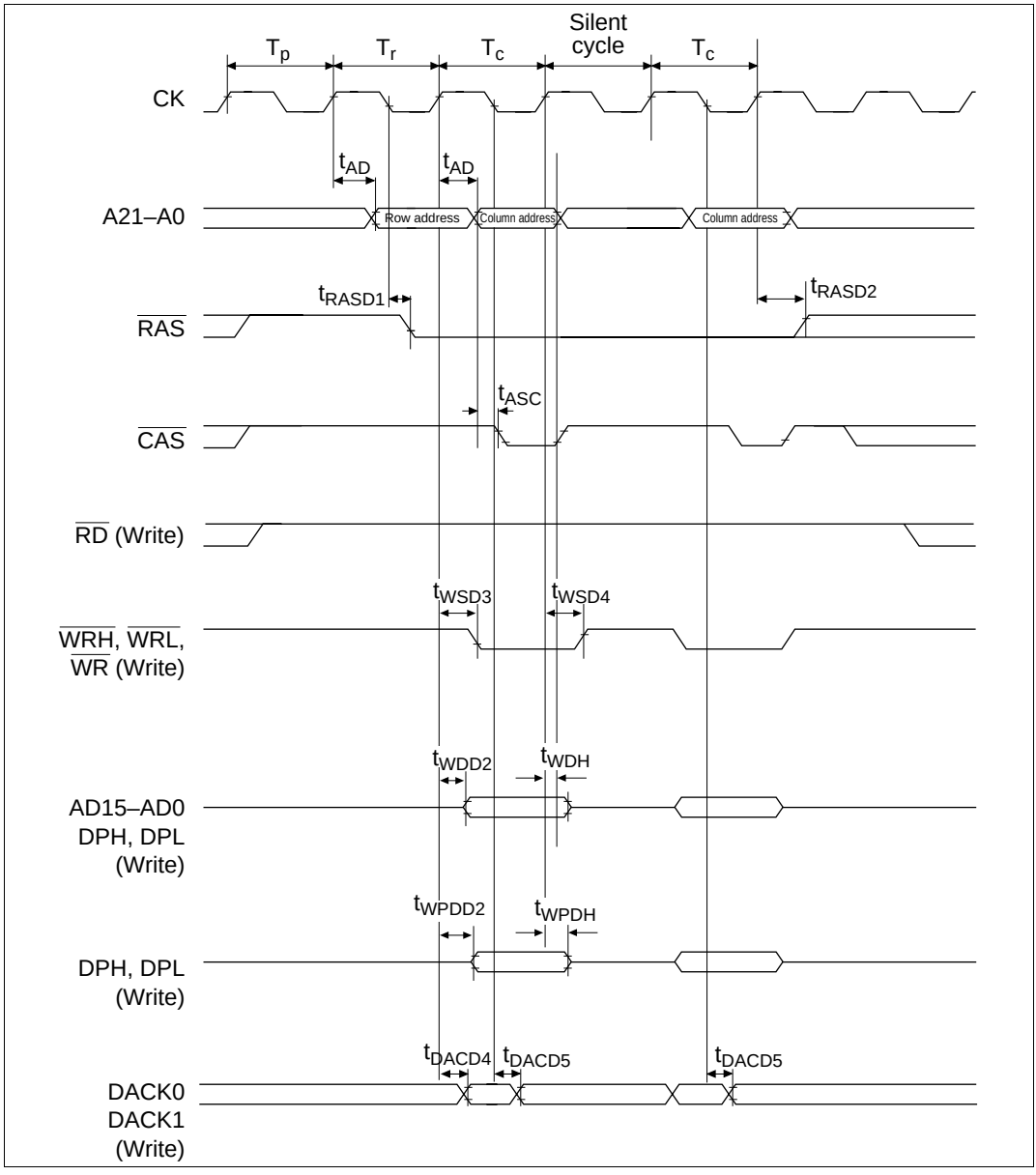
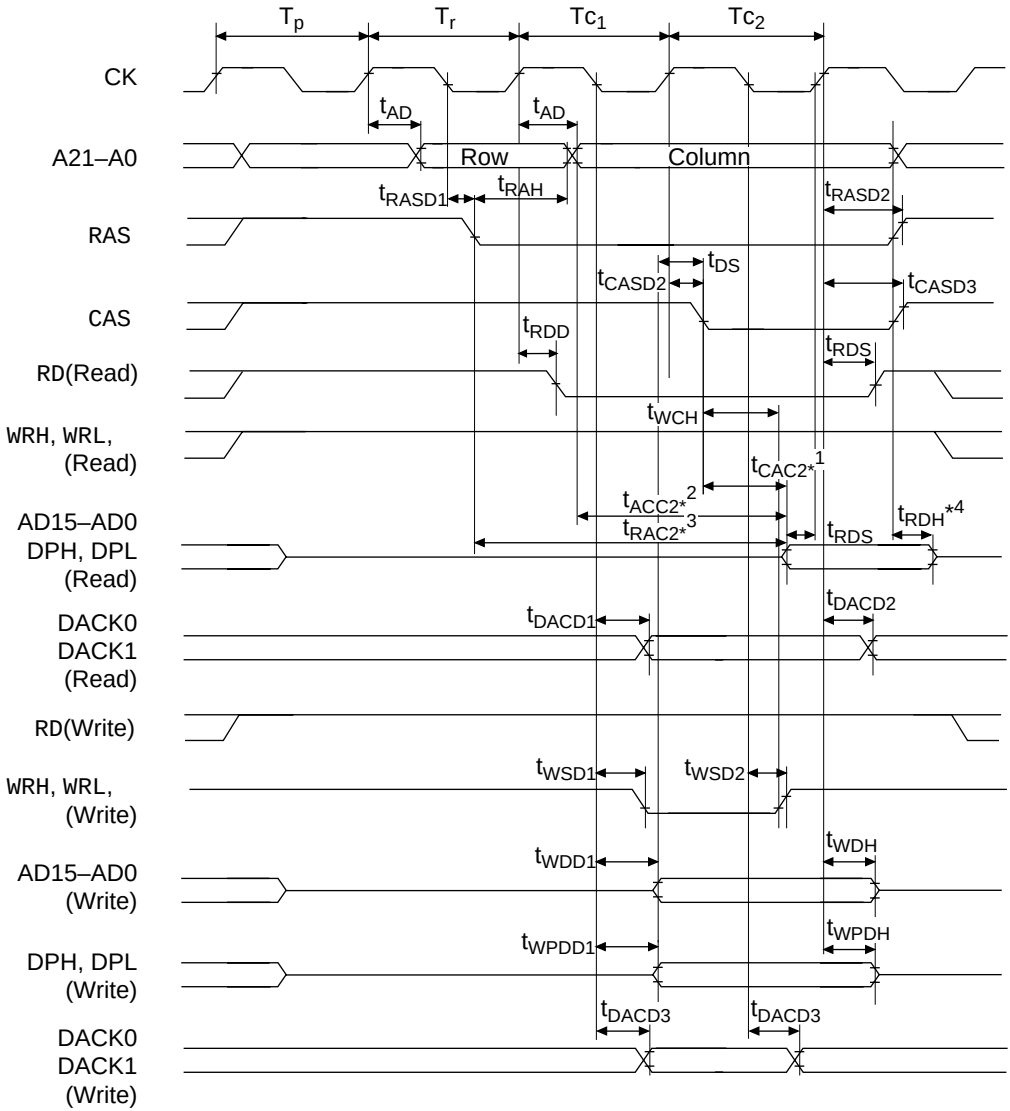



**Figure 19.25 (b) DRAM Bus Cycle (Short-Pitch, High-Speed Page Mode: Write)**

Note: For details of the silent cycle, see section 8.5.5, Burst Operation.



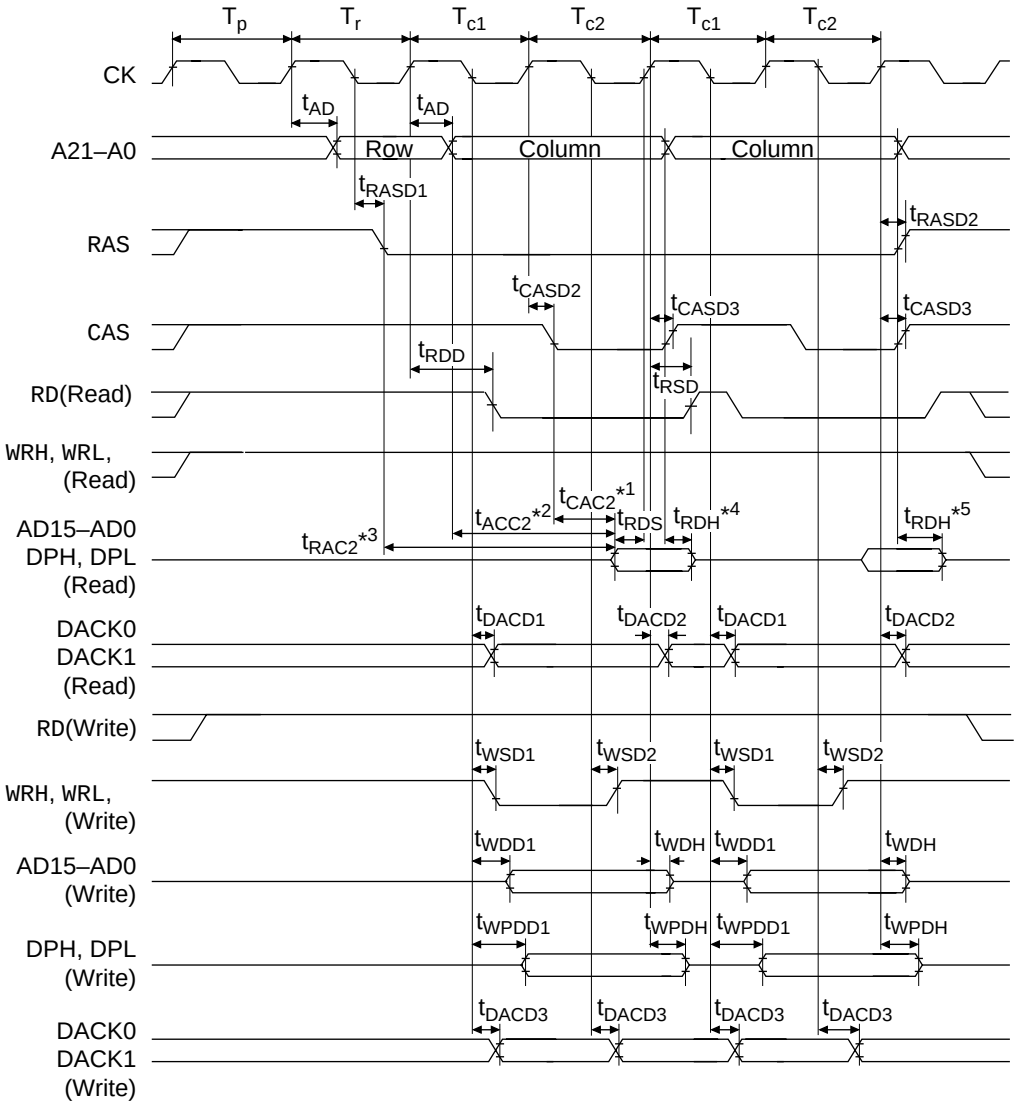

Notes: 1. For $t_{CAC2}$, use $t_{cyc} \times (n + 1) - 35$ instead of $t_{cyc} \times (n + 1) - t_{CASD2} - t_{RDS}$.

2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 44$ instead of $t_{cyc} \times (n + 2) - t_{AD} - t_{RDS}$.

3. For $t_{RAC2}$, use $t_{cyc} \times (n + 2.5) - 35$ instead of $t_{cyc} \times (n + 2.5) - t_{RASD1} - t_{RDS}$.

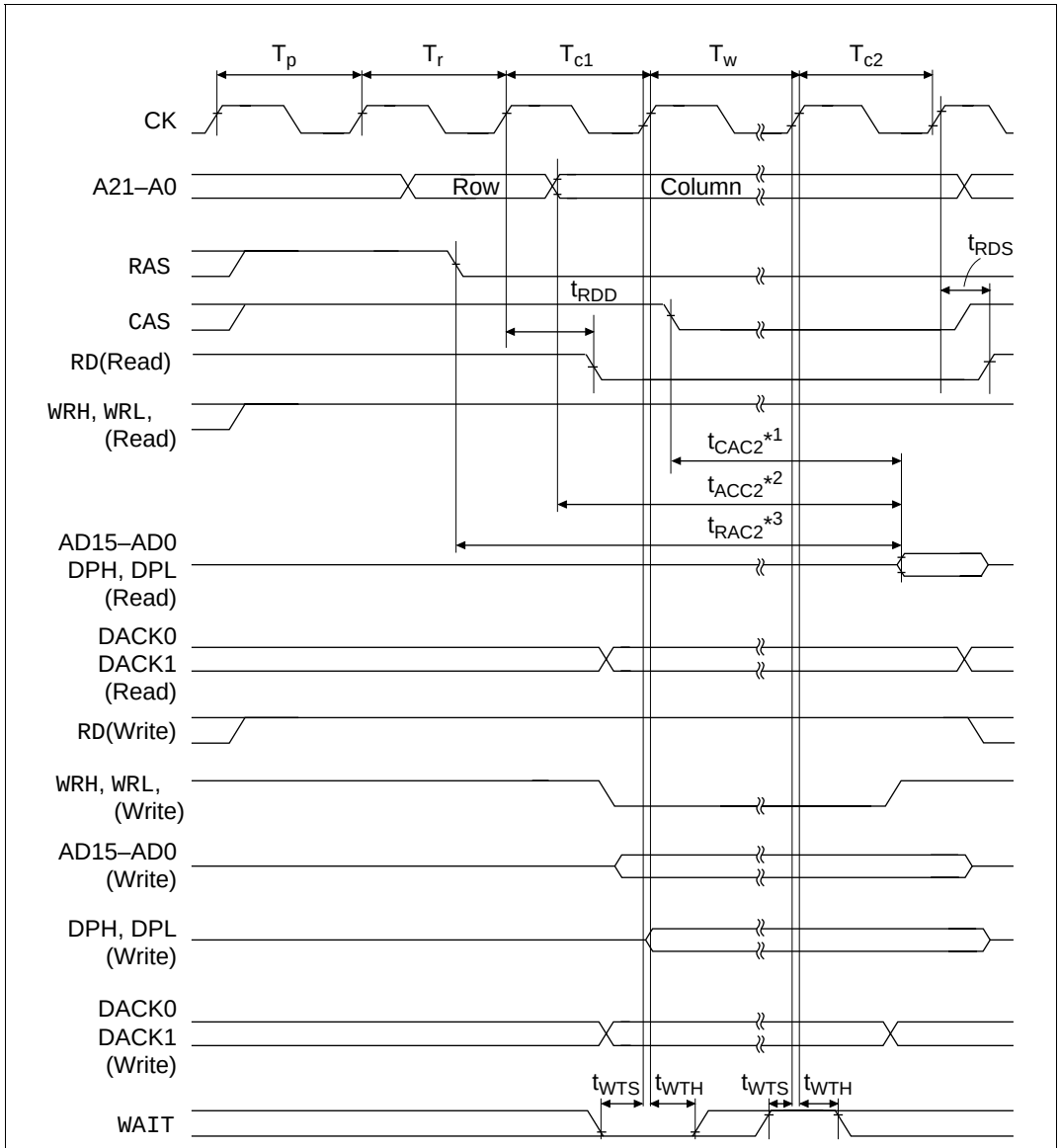
4. $t_{RDH}$ is measured from A21–A0, CAS, or RAS, whichever is negated first.

**Figure 19.26 DRAM Bus Cycle: (Long Pitch, Normal Mode)**




Notes:
1. For $t_{CAC2}$, use $t_{cyc} \times (n + 1) - 35$ instead of $t_{cyc} \times (n + 1) - t_{CASD2} - t_{RDS}$.
2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 44$ instead of $t_{cyc} \times (n + 2) - t_{AD} - t_{RDS}$.
3. For $t_{RAC2}$, use $t_{cyc} \times (n + 2.5) - 35$ instead of $t_{cyc} \times (n + 2.5) - t_{RASD1} - t_{RDS}$.
4. $t_{RDH}$ is measured from A21–A0 or CAS, whichever is negated first.
5. $t_{RDH}$ is measured from A21–A0, RAS, or CAS whichever is negated first.

**Figure 19.27 DRAM Bus Cycle: (Long Pitch, High-Speed Page Mode)**




Notes: 1. For $t_{CAC2}$, use $t_{cyc} \times (n + 1) - 35$ instead of $t_{cyc} \times (n + 1) - t_{CASD2} - t_{RDS}$.
2. For $t_{ACC2}$, use $t_{cyc} \times (n + 2) - 44$ instead of $t_{cyc} \times (n + 2) - t_{AD} - t_{RDS}$.
3. For $t_{RAC2}$, use $t_{cyc} \times (n + 2.5) - 35$ instead of $t_{cyc} \times (n + 2.5) - t_{RASD1} - t_{RDS}$.

**Figure 19.28 DRAM Bus Cycle: (Long Pitch, High-Speed Page Mode + Wait State)**

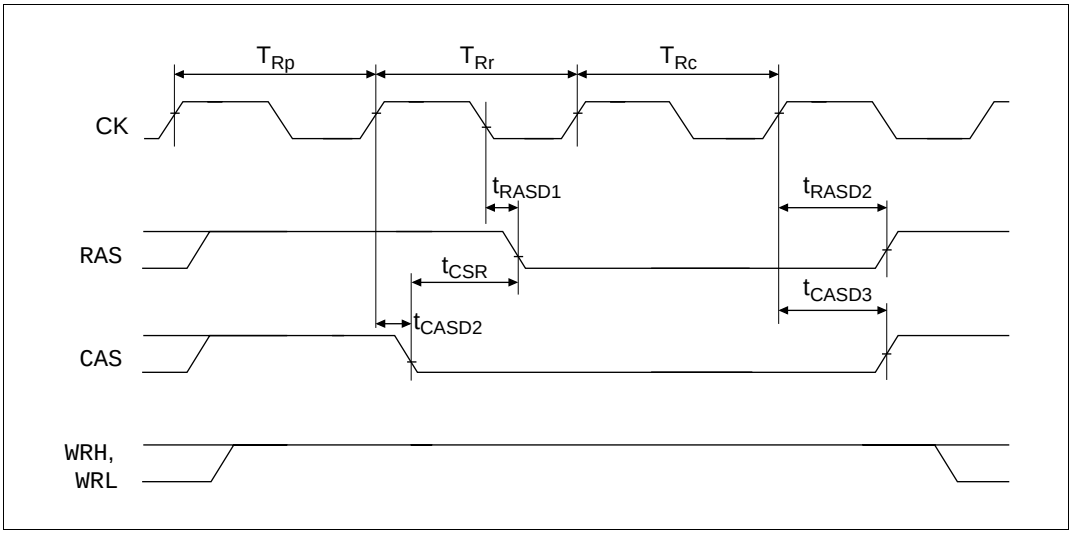



**Figure 19.29 CAS-before-RAS Refresh (Short Pitch)**

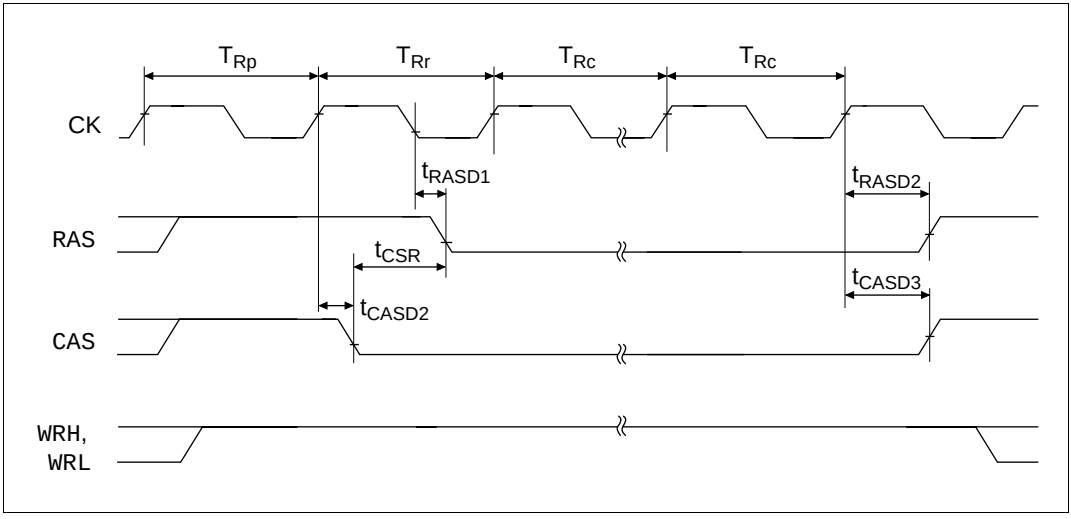



**Figure 19.30 CAS-before-RAS Refresh (Long Pitch)**

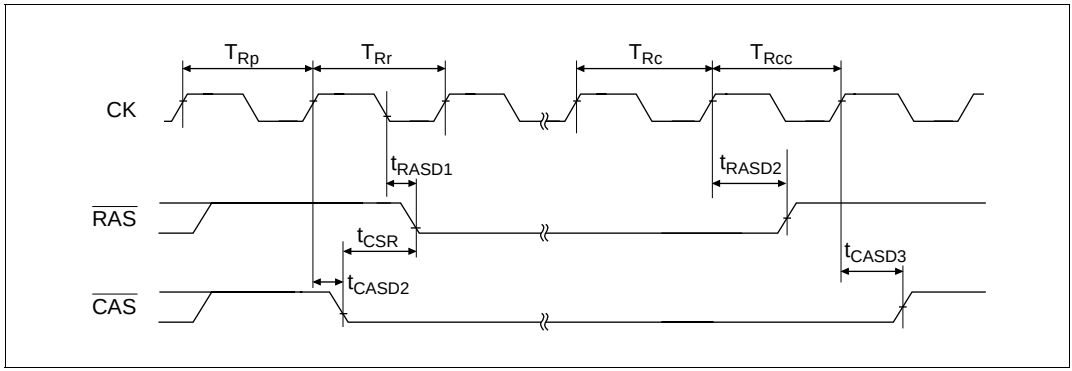



**Figure 19.31 Self Refresh**

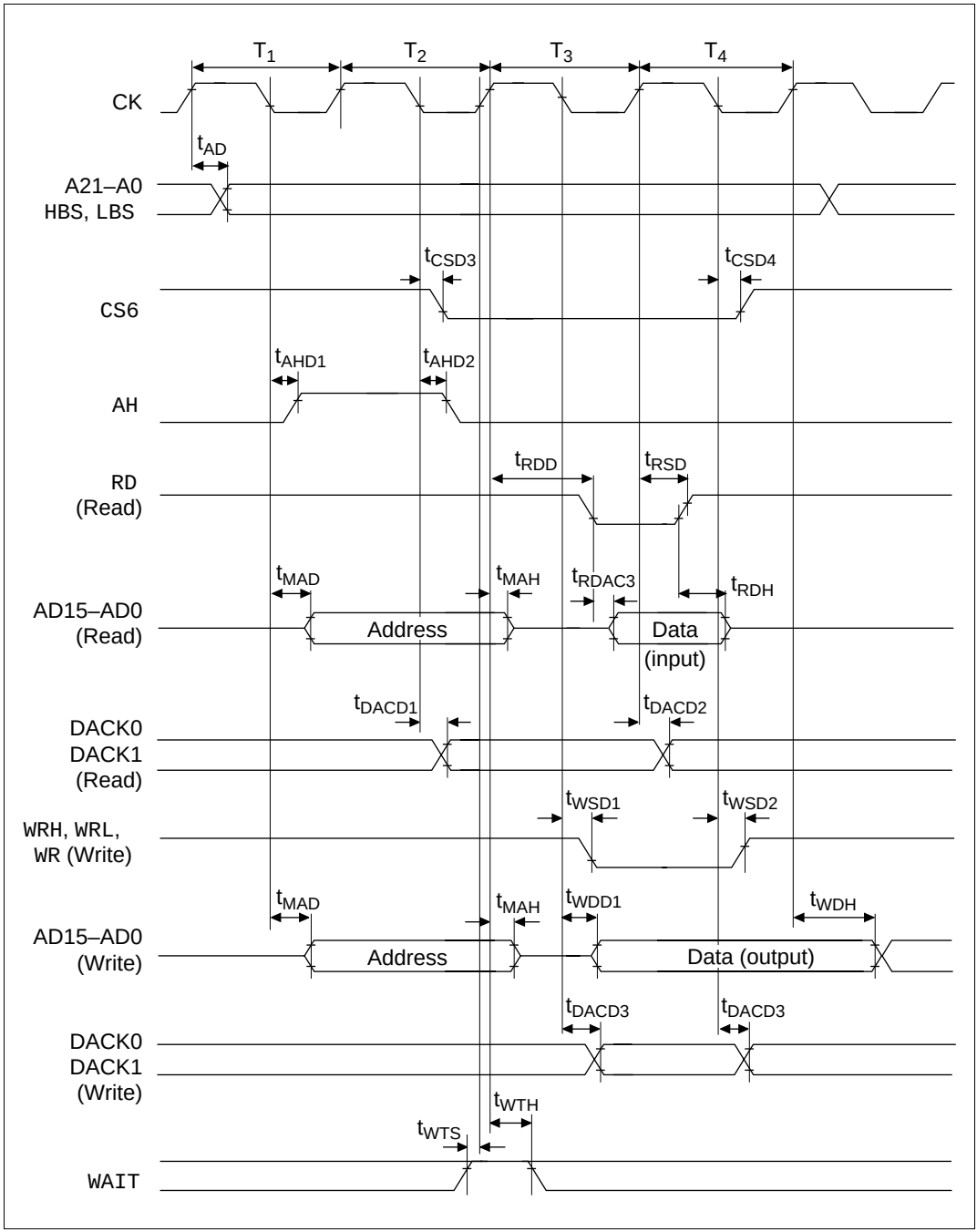



**Figure 19.32 Address/Data Multiplex I/O Bus Cycle**

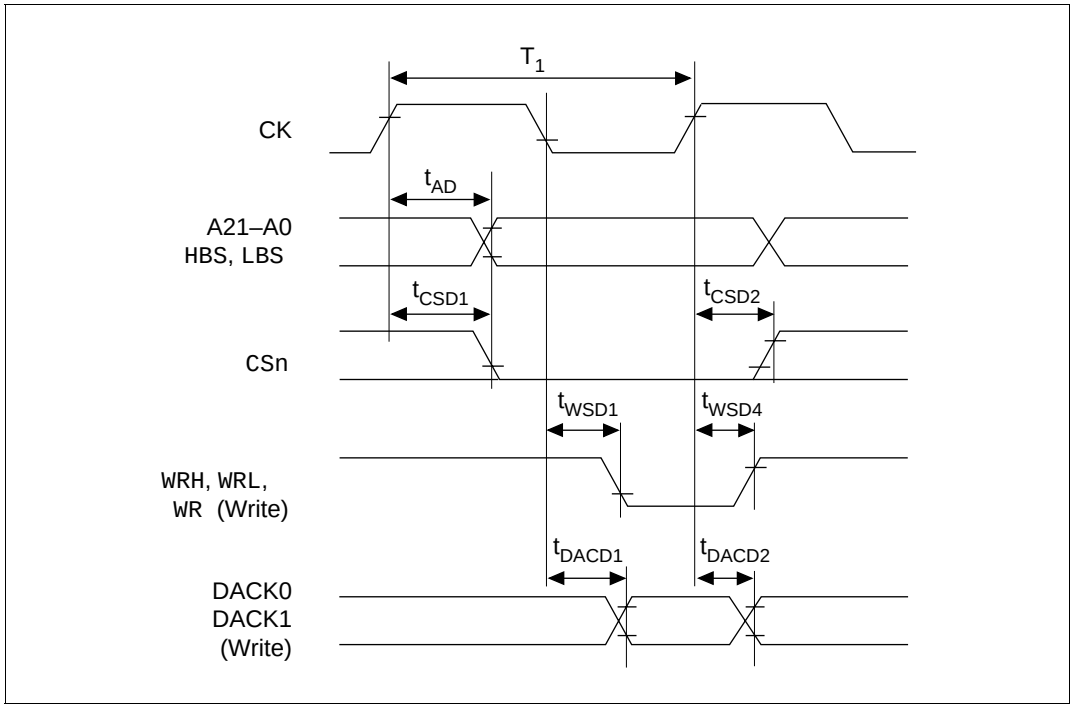



**Figure 19.33 DMA Single Transfer/Single State Access Write**

### 19.3.4 DMAC Timing

**Table 19.9 DMAC Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | Case A | | Case B | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 12.5 MHz | | 16.6 MHz | | 20 MHz | | | |
| Item | Symbol | Min | Max | Min | Max | Min | Max | Unit | Figure |
| DREQ0, DREQ1 setup time | $t_{DRQS}$ | 80 | — | 40 | — | 27 | — | ns | 19.34 |
| DREQ0, DREQ1 hold time | $t_{DRQH}$ | 30 | — | 30 | — | 30 | — | ns | |
| DREQ0, DREQ1 low level width | $t_{DRQW}$ | 1.5 | — | 1.5 | — | 1.5 | — | $t_{cyc}$ | 19.35 |

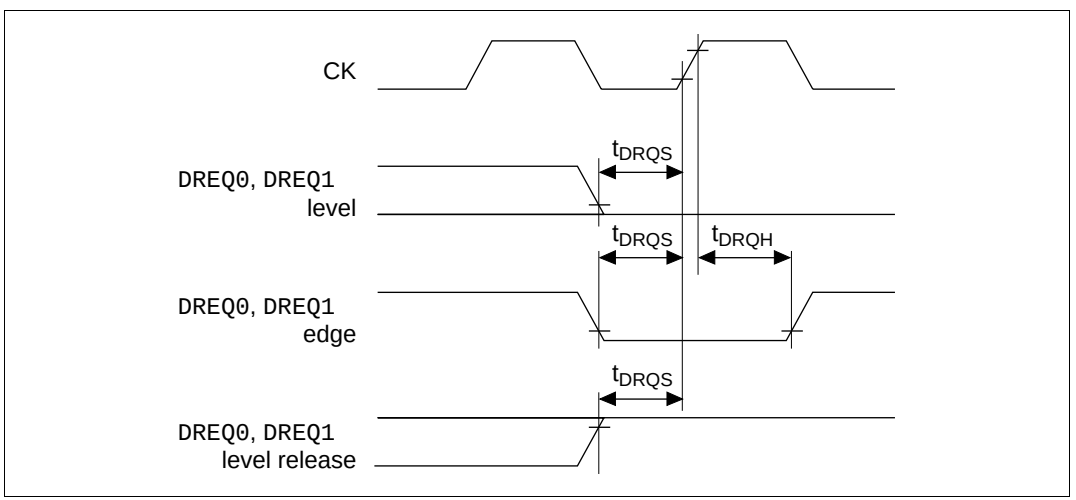



**Figure 19.34 $\overline{\text{DREQ0}}$, $\overline{\text{DREQ1}}$ Input Timing (1)**

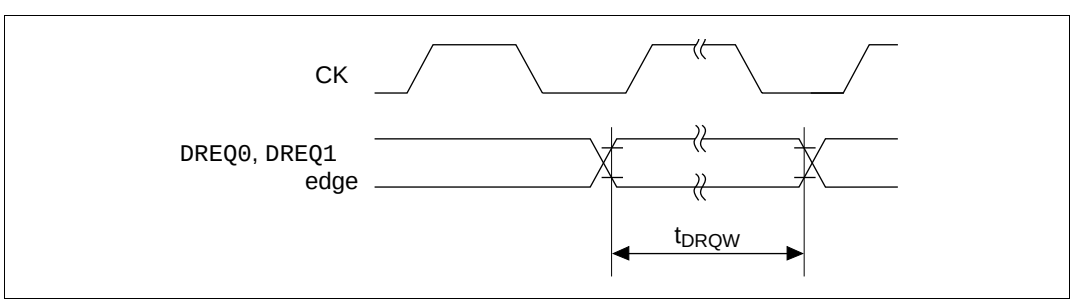



**Figure 19.35 $\overline{\text{DREQ0}}$, $\overline{\text{DREQ1}}$ Input Timing (2)**

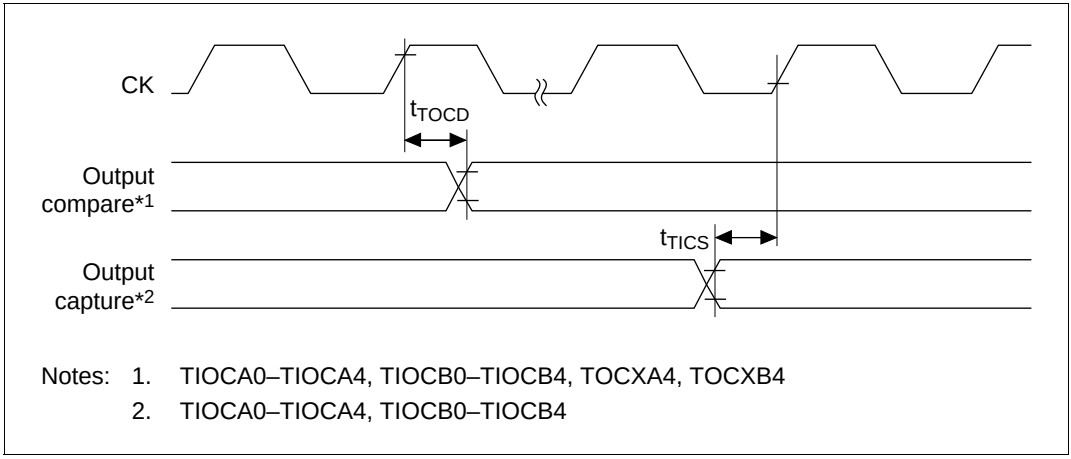
### 19.3.5 16-bit Integrated Timer Pulse Unit Timing

**Table 19.10 16-bit Integrated Timer Pulse Unit Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | Case A | | Case B | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 12.5 MHz | | 16.6 MHz | | 20 MHz | | | |
| **Item** | **Symbol** | **Min** | **Max** | **Min** | **Max** | **Min** | **Max** | **Unit** | **Figure** |
| Output compare delay time | $t_{TOCD}$ | — | 100 | — | 100 | — | 100 | ns | 19.36 |
| Input capture setup time | $t_{TICS}$ | 50 | — | 45 | — | 35 | — | ns | |
| Timer clock input setup time | $t_{TCKS}$ | 50 | — | 50 | — | 50 | — | ns | 19.37 |
| Timer clock pulse width (single edge) | $t_{TCKWH/L}$ | 1.5 | — | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| Timer clock pulse width (both edges) | $t_{TCKWH/L}$ | 2.5 | — | 2.5 | — | 2.5 | — | $t_{cyc}$ | |




Notes: 1. TIOCA0–TIOCA4, TIOCB0–TIOCB4, TOCXA4, TOCXB4
2. TIOCA0–TIOCA4, TIOCB0–TIOCB4

**Figure 19.36 ITU Input/Output Timing**

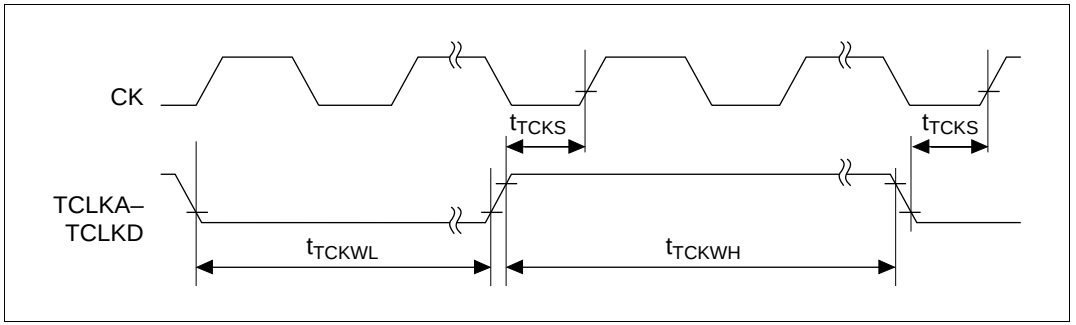



**Figure 19.37 ITU Clock Input Timing**

### 19.3.6 Programmable Timing Pattern Controller and I/O Port Timing

**Table 19.11 Programmable Timing Pattern Controller and I/O Port Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*
Case C: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | Cases A, B and C | | | |
|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Unit** | **Figure** |
| Port output delay time | $t_{PWD}$ | — | 100 | ns | 19.38 |
| Port input hold time | $t_{PRH}$ | 50 | — | ns | |
| Port input setup time | $t_{PRS}$ | 50 | — | ns | |

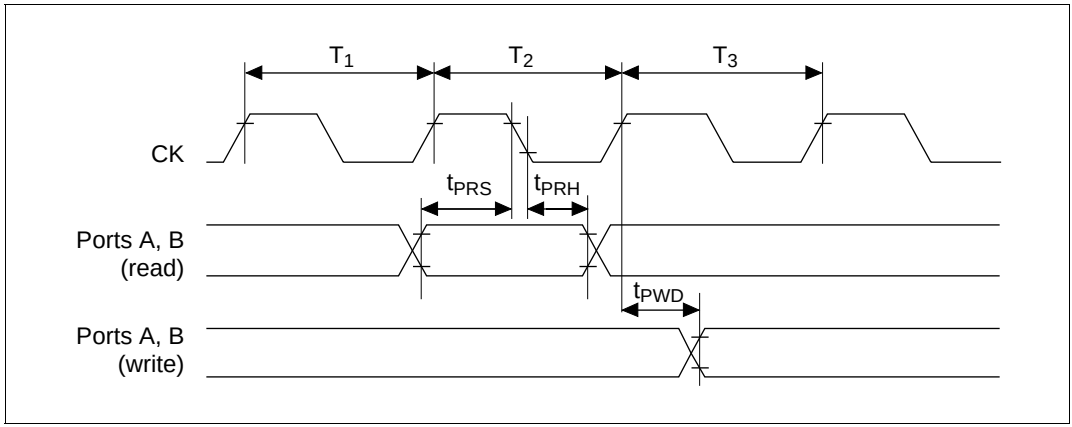



**Figure 19.38 Programmable Timing Pattern Controller Output Timing**

### 19.3.7 Watchdog Timer Timing

**Table 19.12 Watchdog Timer Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 V, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*
Case C: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products. Ta = –40 to +85°C for wide-temperature range products.

| | | Cases A, B and C | | | |
|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Unit** | **Figure** |
| WDTOVF delay time | $t_{WOVD}$ | — | 100 | ns | 20.39 |

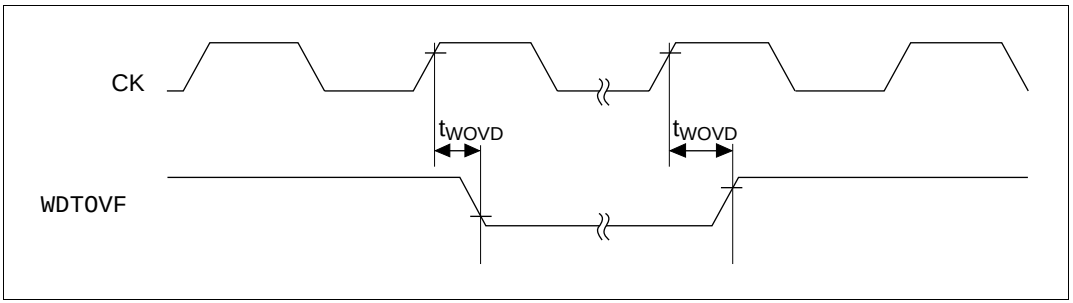



**Figure 19.39 Watchdog Timer Output Timing**

### 19.3.8 Serial Communications Interface Timing

**Table 19.13 Serial Communications Interface Timing**

Case A: $V_{CC}$ = 3.0 to 5.5 V, $V_{SS}$ = 0 V, $\phi$ = 12.5 MHz, Ta = –20 to +75°C*
Case B: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 16.6 MHz, Ta = –20 to +75°C*
Case C: $V_{CC}$ = 5.0 V ±10%, $V_{SS}$ = 0 V, $\phi$ = 20 MHz, Ta = –20 to +75°C*

*: Normal products: Ta = –40 to +85°C for wide-temperature range products.

| | | Cases A, B and C | | | |
|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min** | **Max** | **Unit** | **Figure** |
| Input clock cycle | $t_{scyc}$ | 4 | — | $t_{cyc}$ | 19.40 |
| Input clock cycle (clocked synchronization) | $t_{scyc}$ | 6 | — | $t_{cyc}$ | |
| Input clock pulse width | $t_{sckw}$ | 0.4 | 0.6 | $t_{scyc}$ | |
| Input clock rise time | $t_{sckr}$ | — | 1.5 | $t_{cyc}$ | |
| Input clock fall time | $t_{sckf}$ | — | 1.5 | $t_{cyc}$ | |
| Transmission data delay time (clocked synchronization) | $t_{TXD}$ | — | 100 | ns | 19.41 |
| Receive data setup time (clocked synchronization) | $t_{RXS}$ | 100 | — | ns | |
| Receive data hold time (clocked synchronization) | $t_{RXH}$ | 100 | — | ns | |

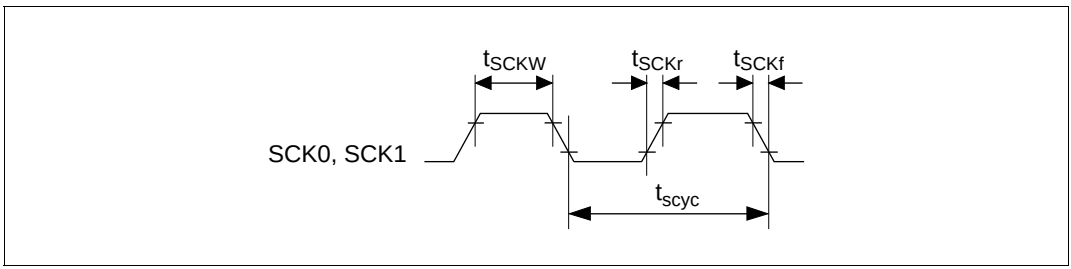



**Figure 19.40 Input Clock Timing**

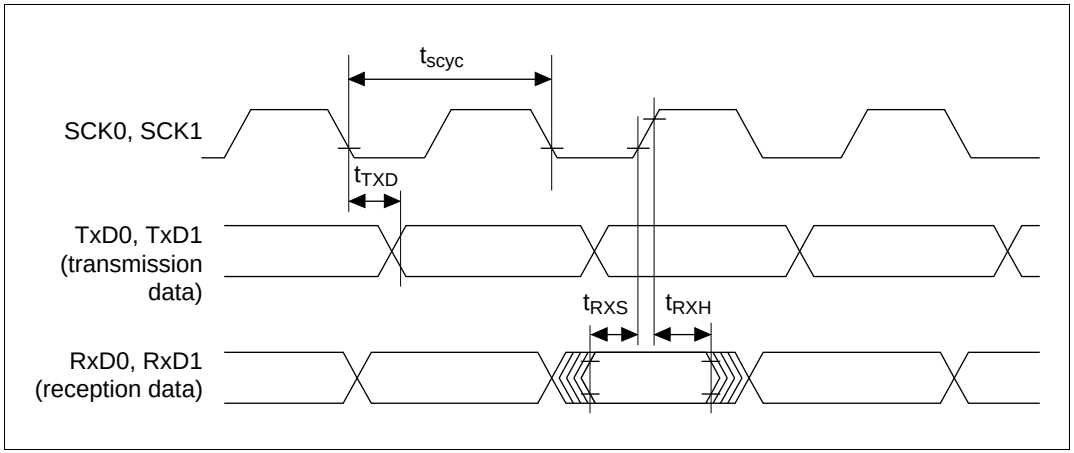
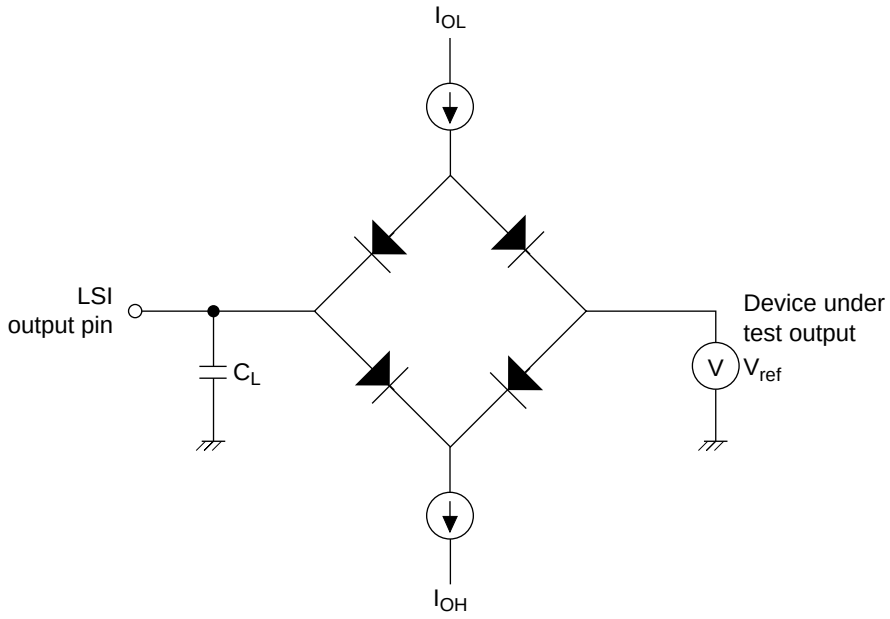



**Figure 19.41 SCI I/O Timing (Clocked Synchronization Mode)**

### 19.3.9 AC Characteristics Measurement Conditions




$C_L$ is set as follows for each pin.

30pF: CK, CASH, CASL, CS0–CS7, BREQ, BACK, AH, IRQOUT, RAS, DACK0, DACK1

50pF: A21–A0, AD15–AD0, DPH, DPL, RD, WRH, WRL, HBS, LBS, WR

70pF: All port outputs and peripheral module output pins other than the above.

$I_{OL}$ and $I_{OH}$ values are as shown in section 19.2, DC Characteristics, and table 19.3, Permitted Output Current Values.

**Figure 19.42 Output Load Circuit**

## 19.4 Usage Note

The ZTAT version and the mask ROM version satisfy the electrical properties given in this document. However, effective values of the electrical properties, the operating margin, and the noise margin may differ with the manufacturing processes, on-chip ROM, and layout patterns. When conducting a system evaluation test using the ZTAT version, conduct a similar evaluation test of the mask ROM version before it replaces the ZTAT version.

**SH7020, SH7021 Hardware Manual**

Publication Date: 1st Edition, September 1994
3rd Edition, September 1998
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.