

The revision list summarizes the locations of revisions and additions. Details should always be checked by referring to the relevant text.

SH7751 Group, SH7751R Group

User's Manual: Hardware

Renesas 32-Bit RISC Microcomputer
SuperH™ RISC engine Family / SH7750 Series

Notice

1. 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
"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 etc.
"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.

6. 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.
 7. 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 systems manufactured by you.
 8. 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.
 9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
 10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
 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.

(2012.4)

General Precautions on Handling of Product

1. Treatment of NC Pins

Note: Do not connect anything to the NC pins.

The NC (not connected) pins are either not connected to any of the internal circuitry or are they are used as test pins or to reduce noise. If something is connected to the NC pins, the operation of the LSI is not guaranteed.

2. Treatment of Unused Input Pins

Note: Fix all unused input pins to high or low level.

Generally, the input pins of CMOS products are high-impedance input pins. If unused pins are in their open states, intermediate levels are induced by noise in the vicinity, a pass-through current flows internally, and a malfunction may occur.

3. Processing before Initialization

Note: When power is first supplied, the product's state is undefined.

The states of internal circuits are undefined until full power is supplied throughout the chip and a low level is input on the reset pin. During the period where the states are undefined, the register settings and the output state of each pin are also undefined. Design your system so that it does not malfunction because of processing while it is in this undefined state. For those products which have a reset function, reset the LSI immediately after the power supply has been turned on.

4. Prohibition of Access to Undefined or Reserved Addresses

Note: Access to undefined or reserved addresses is prohibited.

The undefined or reserved addresses may be used to expand functions, or test registers may have been allocated to these addresses. Do not access these registers; the system's operation is not guaranteed if they are accessed.

5. Reading from/Writing to Reserved Bit of Each Register

Note: Treat the reserved bit of register used in each module as follows except in cases where the specifications for values which are read from or written to the bit are provided in the description.

The bit is always read as 0. The write value should be 0 or one, which has been read immediately before writing.

Writing the value, which has been read immediately before writing has the advantage of preventing the bit from being affected on its extended function when the function is assigned.

Preface

The SH-4 (SH7751 Group (SH7751, SH7751R)) microprocessor incorporates the 32-bit SH-4 CPU and is also equipped with peripheral functions necessary for configuring a user system.

The SH7751 Group is built in with a variety of peripheral functions such as cache memory, memory management unit (MMU), interrupt controller, floating-point unit (FPU), timers, two serial communication interfaces (SCI, SCIF), real-time clock (RTC), user break controller (UBC), bus state controller (BSC) and PCI controller (PCIC). This series can be used in a wide range of multimedia equipment. The bus controller is compatible with ROM, SRAM, DRAM, synchronous DRAM and PCMCIA.

Target Readers: This manual is designed for use by people who design application systems using the SH7751 or SH7751R.

To use this manual, basic knowledge of electric circuits, logic circuits and microcomputers is required.

This hardware manual contains revisions related to the addition of R-mask functionality. Be sure to check the text for the updated content.

Purpose: This manual provides the information of the hardware functions and electrical characteristics of the SH7751 and SH7751R.

The SH-4 Software Manual contains detailed information of executable instructions. Please read the Software Manual together with this manual.

How to Use the Book:

- To understand general functions
 - Read the manual from the beginning.
 - The manual explains the CPU, system control functions, peripheral functions and electrical characteristics in that order.
- To understanding CPU functions
 - Refer to the separate SH-4 Software Manual.

Explanatory Note: Bit sequence: upper bit at left, and lower bit at right

List of Related Documents: The latest documents are available on our Web site. Please make sure that you have the latest version.

(<http://www.renesas.com/>)

- User manuals for SH7751 and SH7751R

| Name of Document | Document No. |
|---|---------------------|
| SH7751 Group, SH7751R Group Hardware Manual | This manual |
| SH-4 Software Manual | REJ09B0318-0600 |

- User manuals for development tools

| Name of Document | Document No. |
|--|---------------------|
| SuperH™ C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual | REJ10B0047-0100H |
| SuperH™ RISC engine Simulator/Debugger User's Manual | REJ10B0210-0300 |
| High-performance Embedded Workshop User's Manual | REJ10J1554-0100 |

Main Revisions for This Edition

| Item | Page | Revision (See Manual for Details) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--|--|--|---------------------|-----------------|-------------|------|------------------------------------|-------------|-------------|--|----------------|----------------------------|--|--------------|------|----------|-------------------------------|------------------------------|-------------|---|----------------|----------------------------|---------|---------------------|-----------|------------------------|--------|-----------------------|----------|-------------------------------------|---------------|---------------------|-----------------|-------------|---------|-------|---------|------------------|--|--|----------------|-------------|--|-----------------|-------------|--|
| All | — | Added ONPAC-BGA products (HD6417750SBA200V) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1.1 SH7751/SH7751R Group Features | 2 | Table amended | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 1.1 SH7751/SH7751R Group Features | 8 | <table border="1"> <thead> <tr> <th>Item</th> <th colspan="5">Features</th> </tr> </thead> <tbody> <tr> <td>CPU</td> <td colspan="5">Renesas Electronics original SuperH architecture</td> </tr> <tr> <th>Item</th> <th colspan="5">Features</th> </tr> <tr> <td>Product lineup</td> <td>Abbreviation</td> <td>Voltage</td> <td>Operating Frequency</td> <td>Model No.</td> <td>Package</td> </tr> <tr> <td rowspan="2">SH7751</td> <td rowspan="2">1.8 V</td> <td rowspan="2">167 MHz</td> <td rowspan="2">HD6417751BP167</td> <td>HD6417751F167</td> <td>256-pin QFP</td> </tr> <tr> <td>HD6417751RBP240</td> <td>256-pin BGA</td> </tr> <tr> <td rowspan="3">SH7751R</td> <td rowspan="3">1.5 V</td> <td rowspan="3">240 MHz</td> <td>HD6417751RBA240H</td> <td></td> <td></td> </tr> <tr> <td>HD6417751RF240</td> <td>256-pin QFP</td> <td></td> </tr> <tr> <td>HD6417751RBG240</td> <td>292-pin BGA</td> <td></td> </tr> </tbody> </table> | Item | Features | | | | | CPU | Renesas Electronics original SuperH architecture | | | | | Item | Features | | | | | Product lineup | Abbreviation | Voltage | Operating Frequency | Model No. | Package | SH7751 | 1.8 V | 167 MHz | HD6417751BP167 | HD6417751F167 | 256-pin QFP | HD6417751RBP240 | 256-pin BGA | SH7751R | 1.5 V | 240 MHz | HD6417751RBA240H | | | HD6417751RF240 | 256-pin QFP | | HD6417751RBG240 | 292-pin BGA | |
| Item | Features | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CPU | Renesas Electronics original SuperH architecture | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Item | Features | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Product lineup | Abbreviation | Voltage | Operating Frequency | Model No. | Package | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| SH7751 | 1.8 V | 167 MHz | HD6417751BP167 | HD6417751F167 | 256-pin QFP | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | HD6417751RBP240 | 256-pin BGA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| SH7751R | 1.5 V | 240 MHz | HD6417751RBA240H | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | HD6417751RF240 | 256-pin QFP | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | HD6417751RBG240 | 292-pin BGA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 22.2.1 PCI Configuration Register 0 (PCICONF0) | 857 | Note amended Note: * The vendor ID H'1054 specifies Hitachi, Ltd., but the SH7751 and SH7751R are now products of Renesas Electronics Corp. For information on these products, contact Renesas Electronics Corp. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 22.12.5 Notes on Parity Error Detection during Master Access | 980, 981 | Newly added | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 23.1 Absolute Maximum Ratings | 983 | Table amended and note added | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.1 Absolute Maximum Ratings | | <table border="1"> <thead> <tr> <th>Item</th> <th>Symbol</th> <th>Value</th> <th>Unit</th> </tr> </thead> <tbody> <tr> <td rowspan="3">I/O, RTC, CPG power supply voltage</td> <td>V_{DD0}^1</td> <td>-0.3 to 4.2</td> <td>V</td> </tr> <tr> <td>V_{DD-RTC}^1</td> <td>-0.3 to 4.6⁽¹⁾</td> <td></td> </tr> <tr> <td>V_{DD-CPG}</td> <td></td> <td></td> </tr> <tr> <td rowspan="2">Internal power supply voltage</td> <td>V_{DD}^1, $V_{DD-PLL1,2}$</td> <td>-0.3 to 2.5</td> <td>V</td> </tr> <tr> <td></td> <td>-0.3 to 2.1⁽¹⁾</td> <td></td> </tr> <tr> <td>Input voltage</td> <td>V_{in}</td> <td>-0.3 to V_{DD0} +0.3</td> <td>V</td> </tr> <tr> <td>Operating temperature</td> <td>T_{OP}</td> <td>-20 to 75, -40 to 85⁽²⁾</td> <td>°C</td> </tr> <tr> <td>Storage temperature</td> <td>T_{STG}</td> <td>-55 to 125</td> <td>°C</td> </tr> </tbody> </table> | Item | Symbol | Value | Unit | I/O, RTC, CPG power supply voltage | V_{DD0}^1 | -0.3 to 4.2 | V | V_{DD-RTC}^1 | -0.3 to 4.6 ⁽¹⁾ | | V_{DD-CPG} | | | Internal power supply voltage | V_{DD}^1 , $V_{DD-PLL1,2}$ | -0.3 to 2.5 | V | | -0.3 to 2.1 ⁽¹⁾ | | Input voltage | V_{in} | -0.3 to V_{DD0} +0.3 | V | Operating temperature | T_{OP} | -20 to 75, -40 to 85 ⁽²⁾ | °C | Storage temperature | T_{STG} | -55 to 125 | °C | | | | | | | | | | | |
| Item | Symbol | Value | Unit | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| I/O, RTC, CPG power supply voltage | V_{DD0}^1 | -0.3 to 4.2 | V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | V_{DD-RTC}^1 | -0.3 to 4.6 ⁽¹⁾ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | V_{DD-CPG} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Internal power supply voltage | V_{DD}^1 , $V_{DD-PLL1,2}$ | -0.3 to 2.5 | V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | -0.3 to 2.1 ⁽¹⁾ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Input voltage | V_{in} | -0.3 to V_{DD0} +0.3 | V | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operating temperature | T_{OP} | -20 to 75, -40 to 85 ⁽²⁾ | °C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Storage temperature | T_{STG} | -55 to 125 | °C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | Notes: 1. HD6417751R only. 2. HD6417751RBA240HV only. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Item | Page | Revision (See Manual for Details) | | | | | | | | | | | | | | | | | | | | | | | | |
|--|------------|---|------|--------|--------|-----|------|--------|--|------------|---|---|----|--|--|------------|---|---|----|--|--|------------|---|---|----|--|
| 23.2 DC Characteristics | 982, | Table title amended and note added | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.2 DC Characteristics (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV) | 983 | Notes: 3. $T_a = -40$ to 85°C for the HD6417751RBA240HV. $T_a = -20$ to $+75^\circ\text{C}$ * ³ | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.4 DC Characteristics (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV)* ³) | 988, 989 | Table title amended and note added Notes: 3. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz. 4. $T_a = -40$ to 85°C for the HD6417751RBA240HV. $T_a = -20$ to $+75^\circ\text{C}$ * ⁴ | | | | | | | | | | | | | | | | | | | | | | | | |
| 23.3 AC Characteristics | 996 | Table title amended | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.9 Clock Timing (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.11 Clock Timing (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV)* | 997 | Table title amended and note added Note: * This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz. | | | | | | | | | | | | | | | | | | | | | | | | |
| 23.3.1 Clock and Control Signal Timing | 998, 999 | Table title and table amended and note added | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.14 Clock and Control Signal Timing (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV) | | | | | | | | | | | | | | | | | | | | | | | | | | |
| $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C * ² , $C_L = 30$ pF | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | <table border="1"> <thead> <tr> <th>Item</th> <th>Symbol</th> <th>Min</th> <th>Max</th> <th>Unit</th> <th>Figure</th> </tr> </thead> <tbody> <tr> <td>Standby return oscillation settling time 1*¹</td> <td>t_{OSCR}</td> <td>2</td> <td>—</td> <td>ms</td> <td></td> </tr> <tr> <td>Standby return oscillation settling time 2*¹</td> <td>t_{OSCS}</td> <td>2</td> <td>—</td> <td>ms</td> <td></td> </tr> <tr> <td>Standby return oscillation settling time 3*¹</td> <td>t_{OSCL}</td> <td>2</td> <td>—</td> <td>ms</td> <td></td> </tr> </tbody> </table> <p>Notes: 1. When the oscillation settling time of the crystal resonator is 1 ms or less. 2. $T_a = -40$ to 85°C for the HD6417751RBA240HV.</p> | Item | Symbol | Min | Max | Unit | Figure | Standby return oscillation settling time 1* ¹ | t_{OSCR} | 2 | — | ms | | Standby return oscillation settling time 2* ¹ | t_{OSCS} | 2 | — | ms | | Standby return oscillation settling time 3* ¹ | t_{OSCL} | 2 | — | ms | |
| Item | Symbol | Min | Max | Unit | Figure | | | | | | | | | | | | | | | | | | | | | |
| Standby return oscillation settling time 1* ¹ | t_{OSCR} | 2 | — | ms | | | | | | | | | | | | | | | | | | | | | | |
| Standby return oscillation settling time 2* ¹ | t_{OSCS} | 2 | — | ms | | | | | | | | | | | | | | | | | | | | | | |
| Standby return oscillation settling time 3* ¹ | t_{OSCL} | 2 | — | ms | | | | | | | | | | | | | | | | | | | | | | |

Item **Page** **Revision (See Manual for Details)**

23.3.1 Clock and Control Signal Timing 1002, 1003

Table 23.16 Clock and Control Signal Timing (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV*²)

$V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C *³, $C_L = 30$ pF

| Item | Symbol | Min | Max | Unit | Figure |
|---|-------------|-----|-----|------|--------|
| Standby return oscillation settling time 1* ¹⁾ | t_{OSCS1} | 2 | — | ms | |
| Standby return oscillation settling time 2* ¹⁾ | t_{OSCS2} | 2 | — | ms | |
| Standby return oscillation settling time 3* ¹⁾ | t_{OSCS3} | 2 | — | ms | |

- Notes: 1. When the oscillation settling time of the crystal resonator is 1 ms or less.
 2. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
 3. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

23.3.2 Control Signal Timing 1012

Table 23.19 Control Signal Timing

| Item | Symbol | Min | Max | Unit | Figure |
|----------------------|----------------------------------|---------------------|---------------------|-----------------|--------|
| HD6417751 RBP240 (V) | HD6417751 RBP200 (V) | | | | |
| HD6417751 RBG240 (V) | HD6417751 RBG200 (V) | | | | |
| HD6417751 RBA240HV | HD6417751 RBA240HV* ² | HD6417751 RF240 (V) | HD6417751 RF200 (V) | | |
| | a ¹⁾ | a ¹⁾ | a ¹⁾ | a ¹⁾ | |

- Notes: 1. $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C *³, $C_L = 30$ pF, PLL2 on
 2. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
 3. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

23.3.3 Bus Timing 1016, 1017

Table 23.21 Bus Timing (1)

| Item | Symbol | Min | Max | Unit | Notes |
|----------------------|----------------------------------|---------------------|---------------------|-----------------|-------|
| HD6417751 RBP240 (V) | HD6417751 RBP200 (V) | | | | |
| HD6417751 RBG240 (V) | HD6417751 RBG200 (V) | | | | |
| HD6417751 RBA240HV | HD6417751 RBA240HV* ² | HD6417751 RF240 (V) | HD6417751 RF200 (V) | | |
| | a ¹⁾ | a ¹⁾ | a ¹⁾ | a ¹⁾ | |

- Notes: 1. $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C *³, $C_L = 30$ pF, PLL2 on
 2. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
 3. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

| Item | Page | Revision (See Manual for Details) | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|----------------|--|----------------|-----------|--|--|------------|------------|--|--|-----------|-----------|--|--|------------|------------|--|--|-----------|-----------|-----------|-----------|----------|----------|-----------|-----------|----------------|----------------|----------------|----------------|
| 23.3.4 Peripheral Module Signal Timing | 1067 | Table amended and note added | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.23 Peripheral Module Signal Timing (1) | 1069 | <table border="1"> <tr> <td>HD6417751</td> <td>HD6417751</td> <td></td> <td></td> </tr> <tr> <td>RBP240 (V)</td> <td>RBP200 (V)</td> <td></td> <td></td> </tr> <tr> <td>HD6417751</td> <td>HD6417751</td> <td></td> <td></td> </tr> <tr> <td>RBG240 (V)</td> <td>RBG200 (V)</td> <td></td> <td></td> </tr> <tr> <td>HD6417751</td> <td>HD6417751</td> <td>HD6417751</td> <td>HD6417751</td> </tr> <tr> <td>RBA240HV</td> <td>RBA240HV</td> <td>RF240 (V)</td> <td>RF200 (V)</td> </tr> <tr> <td>⌘³</td> <td>⌘³</td> <td>⌘³</td> <td>⌘³</td> </tr> </table> <p>Notes: 1. $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C⁴, $C_L = 30$ pF, PLL2 on</p> <p>3. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.</p> <p>4. $T_a = -40$ to 85°C for the HD6417751RBA240HV.</p> | HD6417751 | HD6417751 | | | RBP240 (V) | RBP200 (V) | | | HD6417751 | HD6417751 | | | RBG240 (V) | RBG200 (V) | | | HD6417751 | HD6417751 | HD6417751 | HD6417751 | RBA240HV | RBA240HV | RF240 (V) | RF200 (V) | ⌘ ³ | ⌘ ³ | ⌘ ³ | ⌘ ³ |
| HD6417751 | HD6417751 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RBP240 (V) | RBP200 (V) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| HD6417751 | HD6417751 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RBG240 (V) | RBG200 (V) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| HD6417751 | HD6417751 | HD6417751 | HD6417751 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RBA240HV | RBA240HV | RF240 (V) | RF200 (V) | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ⌘ ³ | ⌘ ³ | ⌘ ³ | ⌘ ³ | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.25 PCIC Signal Timing (in PCIREQ/PCIGNT Non-Port Mode) (1) | 1076 | Table title amended and note added | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| HD6417751RBP240 (V), HD6417751RBP200 (V), HD6417751RBG240 (V), HD6417751RBG200 (V), HD6417751RBA240HV, HD6417751RF240 (V), HD6417751RF200 (V): $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C ² , $C_L = 30$ pF | | Notes: 1. HD6417751RF240 (V), HD6417751RF200 (V) 2. $T_a = -40$ to 85°C for the HD6417751RBA240HV. | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | Asterisk "*" in table changed to "*1" "3.0 (3.5*)" → "3.0 (3.5*1)" | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Table 23.27 PCIC Signal Timing (With PCIREQ/PCIGNT Port Settings in Non-Host Mode) (1) | 1079 | Table title amended and note added | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| HD6417751RBP240 (V), HD6417751RBP200 (V), HD6417751RBG240 (V), HD6417751RBG200 (V), HD6417751RBA240HV, HD6417751RF240 (V), HD6417751RF200 (V): $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C [*] , $C_L = 30$ pF | | Note: * $T_a = -40$ to 85°C for the HD6417751RBA240HV. | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Item | Page | Revision (See Manual for Details) | | | | | | |
|--|------|--|----------|-----|----------|----------|-----|-------------------|
| Appendix B Package Dimensions Figure B.2 Package Dimensions (256-pin BGA: Devices Other than HD6417751RBA240HV) | 1092 | Figure title amended | | | | | | |
| Figure B.4 Package Dimensions (256-Pin BGA: HD6417750RBA240HV) | 1094 | Figure newly added | | | | | | |
| D.2 Handling of Unused Pins Table D.4 Handling of Pins When PCI Is Not Used | 1105 | Table amended <table border="1"> <thead> <tr> <th>Pin Name</th> <th>I/O</th> <th>Handling</th> </tr> </thead> <tbody> <tr> <td>AD31-AD0</td> <td>I/O</td> <td>Pull up to 3.3 V*</td> </tr> </tbody> </table> | Pin Name | I/O | Handling | AD31-AD0 | I/O | Pull up to 3.3 V* |
| Pin Name | I/O | Handling | | | | | | |
| AD31-AD0 | I/O | Pull up to 3.3 V* | | | | | | |
| Appendix H Product Lineup Table H.1 SH7751/SH7751R Product Lineup | 1125 | Table note amended Notes: 1. Contact a Renesas sales office regarding product versions with specifications for a wider temperature range (-40 to +85°C). The wide temperature range (-40 to +85°C) is the standard specification for the HD6417751RBA240HV. | | | | | | |

All trademarks and registered trademarks are the property of their respective owners.

Contents

| | | |
|-----------|---|----|
| Section 1 | Overview | 1 |
| 1.1 | SH7751/SH7751R Group Features | 1 |
| 1.2 | Block Diagram | 9 |
| 1.3 | Pin Arrangement | 10 |
| 1.4 | Pin Functions | 13 |
| 1.4.1 | Pin Functions (256-Pin QFP) | 13 |
| 1.4.2 | Pin Functions (256-Pin BGA) | 24 |
| 1.4.3 | Pin Functions (292-Pin BGA) | 35 |
| Section 2 | Programming Model | 47 |
| 2.1 | Data Formats | 47 |
| 2.2 | Register Configuration | 48 |
| 2.2.1 | Privileged Mode and Banks | 48 |
| 2.2.2 | General Registers | 51 |
| 2.2.3 | Floating-Point Registers | 53 |
| 2.2.4 | Control Registers | 55 |
| 2.2.5 | System Registers | 56 |
| 2.3 | Memory-Mapped Registers | 58 |
| 2.4 | Data Format in Registers | 59 |
| 2.5 | Data Formats in Memory | 59 |
| 2.6 | Processor States | 60 |
| 2.7 | Processor Modes | 62 |
| Section 3 | Memory Management Unit (MMU) | 63 |
| 3.1 | Overview | 63 |
| 3.1.1 | Features | 63 |
| 3.1.2 | Role of the MMU | 63 |
| 3.1.3 | Register Configuration | 66 |
| 3.1.4 | Caution | 66 |
| 3.2 | Register Descriptions | 67 |
| 3.3 | Address Space | 71 |
| 3.3.1 | Physical Address Space | 71 |
| 3.3.2 | External Memory Space | 74 |
| 3.3.3 | Virtual Address Space | 75 |
| 3.3.4 | On-Chip RAM Space | 76 |
| 3.3.5 | Address Translation | 76 |
| 3.3.6 | Single Virtual Memory Mode and Multiple Virtual Memory Mode | 77 |

| | | |
|---------------------------|--|---------|
| 3.3.7 | Address Space Identifier (ASID)..... | 77 |
| 3.4 | TLB Functions | 78 |
| 3.4.1 | Unified TLB (UTLB) Configuration | 78 |
| 3.4.2 | Instruction TLB (ITLB) Configuration..... | 82 |
| 3.4.3 | Address Translation Method..... | 82 |
| 3.5 | MMU Functions..... | 85 |
| 3.5.1 | MMU Hardware Management..... | 85 |
| 3.5.2 | MMU Software Management | 85 |
| 3.5.3 | MMU Instruction (LDTLB)..... | 85 |
| 3.5.4 | Hardware ITLB Miss Handling | 86 |
| 3.5.5 | Avoiding Synonym Problems..... | 87 |
| 3.6 | MMU Exceptions..... | 88 |
| 3.6.1 | Instruction TLB Multiple Hit Exception..... | 88 |
| 3.6.2 | Instruction TLB Miss Exception..... | 88 |
| 3.6.3 | Instruction TLB Protection Violation Exception | 89 |
| 3.6.4 | Data TLB Multiple Hit Exception | 90 |
| 3.6.5 | Data TLB Miss Exception | 91 |
| 3.6.6 | Data TLB Protection Violation Exception..... | 92 |
| 3.6.7 | Initial Page Write Exception..... | 93 |
| 3.7 | Memory-Mapped TLB Configuration..... | 94 |
| 3.7.1 | ITLB Address Array | 94 |
| 3.7.2 | ITLB Data Array 1..... | 95 |
| 3.7.3 | ITLB Data Array 2..... | 96 |
| 3.7.4 | UTLB Address Array..... | 97 |
| 3.7.5 | UTLB Data Array 1 | 98 |
| 3.7.6 | UTLB Data Array 2 | 99 |
| 3.8 | Usage Notes | 100 |
| Section 4 Caches..... | | 101 |
| 4.1 | Overview..... | 101 |
| 4.1.1 | Features..... | 101 |
| 4.1.2 | Register Configuration..... | 102 |
| 4.2 | Register Descriptions..... | 103 |
| 4.3 | Operand Cache (OC)..... | 105 |
| 4.3.1 | Configuration..... | 105 |
| 4.3.2 | Read Operation | 108 |
| 4.3.3 | Write Operation | 109 |
| 4.3.4 | Write-Back Buffer | 111 |
| 4.3.5 | Write-Through Buffer..... | 111 |
| 4.3.6 | RAM Mode..... | 111 |

| | | |
|---------------------------|---|-----|
| 4.3.7 | OC Index Mode | 113 |
| 4.3.8 | Coherency between Cache and External Memory | 113 |
| 4.3.9 | Prefetch Operation | 113 |
| 4.3.10 | Notes on Using OC RAM Mode (SH7751R Only) when in Cache Enhanced Mode | 114 |
| 4.4 | Instruction Cache (IC)..... | 116 |
| 4.4.1 | Configuration | 116 |
| 4.4.2 | Read Operation | 119 |
| 4.4.3 | IC Index Mode..... | 120 |
| 4.5 | Memory-Mapped Cache Configuration (SH7751)..... | 120 |
| 4.5.1 | IC Address Array | 120 |
| 4.5.2 | IC Data Array..... | 122 |
| 4.5.3 | OC Address Array | 123 |
| 4.5.4 | OC Data Array | 124 |
| 4.6 | Memory-Mapped Cache Configuration (SH7751R)..... | 125 |
| 4.6.1 | IC Address Array | 125 |
| 4.6.2 | IC Data Array..... | 127 |
| 4.6.3 | OC Address Array | 128 |
| 4.6.4 | OC Data Array | 129 |
| 4.6.5 | Summary of Memory-Mapped OC Addresses..... | 130 |
| 4.7 | Store Queues | 131 |
| 4.7.1 | SQ Configuration..... | 131 |
| 4.7.2 | SQ Writes..... | 131 |
| 4.7.3 | Transfer to External Memory..... | 132 |
| 4.7.4 | Determination of SQ Access Exception..... | 133 |
| 4.7.5 | SQ Read (SH7751R only)..... | 133 |
| 4.7.6 | SQ Usage Notes (SH7751 Only) | 134 |
| Section 5 Exceptions..... | | 137 |
| 5.1 | Overview..... | 137 |
| 5.1.1 | Features..... | 137 |
| 5.1.2 | Register Configuration..... | 137 |
| 5.2 | Register Descriptions | 138 |
| 5.3 | Exception Handling Functions..... | 139 |
| 5.3.1 | Exception Handling Flow | 139 |
| 5.3.2 | Exception Handling Vector Addresses | 139 |
| 5.4 | Exception Types and Priorities | 140 |
| 5.5 | Exception Flow | 143 |
| 5.5.1 | Exception Flow | 143 |
| 5.5.2 | Exception Source Acceptance..... | 144 |

| | | |
|---|--|------------|
| 5.5.3 | Exception Requests and BL Bit | 146 |
| 5.5.4 | Return from Exception Handling..... | 146 |
| 5.6 | Description of Exceptions..... | 146 |
| 5.6.1 | Resets..... | 147 |
| 5.6.2 | General Exceptions..... | 152 |
| 5.6.3 | Interrupts..... | 166 |
| 5.6.4 | Priority Order with Multiple Exceptions | 169 |
| 5.7 | Usage Notes | 170 |
| 5.8 | Restrictions | 171 |
| Section 6 Floating-Point Unit..... | | 173 |
| 6.1 | Overview..... | 173 |
| 6.2 | Data Formats..... | 173 |
| 6.2.1 | Floating-Point Format..... | 173 |
| 6.2.2 | Non-Numbers (NaN) | 175 |
| 6.2.3 | Denormalized Numbers | 176 |
| 6.3 | Registers | 177 |
| 6.3.1 | Floating-Point Registers | 177 |
| 6.3.2 | Floating-Point Status/Control Register (FPSCR)..... | 179 |
| 6.3.3 | Floating-Point Communication Register (FPUL) | 180 |
| 6.4 | Rounding..... | 181 |
| 6.5 | Floating-Point Exceptions..... | 181 |
| 6.6 | Graphics Support Functions..... | 183 |
| 6.6.1 | Geometric Operation Instructions..... | 183 |
| 6.6.2 | Pair Single-Precision Data Transfer..... | 184 |
| 6.7 | Usage Notes | 185 |
| 6.7.1 | Rounding Mode and Underflow Flag | 185 |
| 6.7.2 | Setting of Overflow Flag by FIPR or FTRV Instruction | 186 |
| 6.7.3 | Sign of Operation Result when Using FIPR or FTRV Instruction | 187 |
| 6.7.4 | Notes on Double-Precision FADD and FSUB Instructions..... | 187 |
| Section 7 Instruction Set..... | | 189 |
| 7.1 | Execution Environment | 189 |
| 7.2 | Addressing Modes | 191 |
| 7.3 | Instruction Set..... | 195 |
| 7.4 | Usage Notes | 207 |
| 7.4.1 | Notes on TRAPA Instruction, SLEEP Instruction, and Undefined Instruction (H'FFFD) | 207 |

| | | |
|-----------|--|-----|
| Section 8 | Pipelining | 211 |
| 8.1 | Pipelines | 211 |
| 8.2 | Parallel-Executability | 218 |
| 8.3 | Execution Cycles and Pipeline Stalling | 222 |
| 8.4 | Usage Notes | 238 |
| Section 9 | Power-Down Modes | 239 |
| 9.1 | Overview | 239 |
| 9.1.1 | Types of Power-Down Modes | 239 |
| 9.1.2 | Register Configuration | 241 |
| 9.1.3 | Pin Configuration | 241 |
| 9.2 | Register Descriptions | 242 |
| 9.2.1 | Standby Control Register (STBCR) | 242 |
| 9.2.2 | Peripheral Module Pin High Impedance Control | 244 |
| 9.2.3 | Peripheral Module Pin Pull-Up Control | 244 |
| 9.2.4 | Standby Control Register 2 (STBCR2) | 245 |
| 9.2.5 | Clock Stop Register 00 (CLKSTP00) | 246 |
| 9.2.6 | Clock Stop Clear Register 00 (CLKSTPCLR00) | 247 |
| 9.3 | Sleep Mode | 248 |
| 9.3.1 | Transition to Sleep Mode | 248 |
| 9.3.2 | Exit from Sleep Mode | 248 |
| 9.4 | Deep Sleep Mode | 248 |
| 9.4.1 | Transition to Deep Sleep Mode | 248 |
| 9.4.2 | Exit from Deep Sleep Mode | 249 |
| 9.5 | Pin Sleep Mode | 249 |
| 9.5.1 | Transition to Pin Sleep Mode | 249 |
| 9.5.2 | Exit from Pin Sleep Mode | 249 |
| 9.6 | Standby Mode | 249 |
| 9.6.1 | Transition to Standby Mode | 249 |
| 9.6.2 | Exit from Standby Mode | 250 |
| 9.6.3 | Clock Pause Function | 251 |
| 9.7 | Module Standby Function | 251 |
| 9.7.1 | Transition to Module Standby Function | 251 |
| 9.7.2 | Exit from Module Standby Function | 252 |
| 9.8 | Hardware Standby Mode | 253 |
| 9.8.1 | Transition to Hardware Standby Mode | 253 |
| 9.8.2 | Exit from Hardware Standby Mode | 253 |
| 9.8.3 | Usage Notes | 254 |
| 9.9 | STATUS Pin Change Timing | 254 |

| | | |
|--|--|------------|
| 9.9.1 | In Reset | 255 |
| 9.9.2 | In Exit from Standby Mode | 256 |
| 9.9.3 | In Exit from Sleep Mode | 257 |
| 9.9.4 | In Exit from Deep Sleep Mode | 260 |
| 9.9.5 | Hardware Standby Mode Timing | 262 |
| 9.10 | Usage Notes | 264 |
| 9.10.1 | Note on Current Consumption | 264 |
| Section 10 Clock Oscillation Circuits | | 267 |
| 10.1 | Overview..... | 267 |
| 10.1.1 | Features..... | 267 |
| 10.2 | Overview of CPG..... | 269 |
| 10.2.1 | Block Diagram of CPG..... | 269 |
| 10.2.2 | CPG Pin Configuration..... | 272 |
| 10.2.3 | CPG Register Configuration..... | 272 |
| 10.3 | Clock Operating Modes | 273 |
| 10.4 | CPG Register Description..... | 275 |
| 10.4.1 | Frequency Control Register (FRQCR) | 275 |
| 10.5 | Changing the Frequency | 278 |
| 10.5.1 | Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 Is Off) | 278 |
| 10.5.2 | Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 Is On)..... | 278 |
| 10.5.3 | Changing Bus Clock Division Ratio (When PLL Circuit 2 Is On)..... | 279 |
| 10.5.4 | Changing Bus Clock Division Ratio (When PLL Circuit 2 Is Off) | 279 |
| 10.5.5 | Changing CPU or Peripheral Module Clock Division Ratio | 279 |
| 10.6 | Output Clock Control..... | 280 |
| 10.7 | Overview of Watchdog Timer | 280 |
| 10.7.1 | Block Diagram..... | 280 |
| 10.7.2 | Register Configuration..... | 281 |
| 10.8 | WDT Register Descriptions | 281 |
| 10.8.1 | Watchdog Timer Counter (WTCNT)..... | 281 |
| 10.8.2 | Watchdog Timer Control/Status Register (WTCSR)..... | 282 |
| 10.8.3 | Notes on Register Access | 284 |
| 10.9 | Using the WDT..... | 285 |
| 10.9.1 | Standby Clearing Procedure | 285 |
| 10.9.2 | Frequency Changing Procedure..... | 285 |
| 10.9.3 | Using Watchdog Timer Mode | 286 |
| 10.9.4 | Using Interval Timer Mode | 286 |
| 10.10 | Notes on Board Design | 287 |
| 10.11 | Usage Notes | 289 |
| 10.11.1 | Invalid Manual Reset Triggered by Watchdog Timer (SH7751 Only)..... | 289 |

| | | |
|------------|---|-----|
| Section 11 | Realtime Clock (RTC) | 291 |
| 11.1 | Overview | 291 |
| 11.1.1 | Features | 291 |
| 11.1.2 | Block Diagram | 292 |
| 11.1.3 | Pin Configuration | 293 |
| 11.1.4 | 11.1.4 Register Configuration | 293 |
| 11.2 | Register Descriptions | 295 |
| 11.2.1 | 64 Hz Counter (R64CNT) | 295 |
| 11.2.2 | Second Counter (RSECCNT) | 296 |
| 11.2.3 | Minute Counter (RMINCNT) | 296 |
| 11.2.4 | Hour Counter (RHRCNT) | 297 |
| 11.2.5 | Day-of-Week Counter (RWKCNT) | 297 |
| 11.2.6 | Day Counter (RDAYCNT) | 298 |
| 11.2.7 | Month Counter (RMONCNT) | 298 |
| 11.2.8 | Year Counter (RYRCNT) | 299 |
| 11.2.9 | Second Alarm Register (RSECAR) | 300 |
| 11.2.10 | Minute Alarm Register (RMINAR) | 300 |
| 11.2.11 | Hour Alarm Register (RHRAR) | 301 |
| 11.2.12 | Day-of-Week Alarm Register (RWKAR) | 301 |
| 11.2.13 | Day Alarm Register (RDAYAR) | 302 |
| 11.2.14 | Month Alarm Register (RMONAR) | 303 |
| 11.2.15 | RTC Control Register 1 (RCR1) | 303 |
| 11.2.16 | RTC Control Register 2 (RCR2) | 305 |
| 11.2.17 | RTC Control Register (RCR3) and Year-Alarm Register (RYRAR) (SH7751R Only) | 308 |
| 11.3 | Operation | 309 |
| 11.3.1 | Time Setting Procedures | 309 |
| 11.3.2 | Time Reading Procedures | 311 |
| 11.3.3 | Alarm Function | 312 |
| 11.4 | Interrupts | 313 |
| 11.5 | Usage Notes | 313 |
| 11.5.1 | Register Initialization | 313 |
| 11.5.2 | Carry Flag and Interrupt Flag in Standby Mode | 313 |
| 11.5.3 | Crystal Oscillation Circuit | 313 |
| Section 12 | Timer Unit (TMU) | 315 |
| 12.1 | Overview | 315 |
| 12.1.1 | Features | 315 |
| 12.1.2 | Block Diagram | 316 |
| 12.1.3 | Pin Configuration | 316 |

| | | |
|-------------------|--|------------|
| 12.1.4 | Register Configuration..... | 317 |
| 12.2 | Register Descriptions..... | 318 |
| 12.2.1 | Timer Output Control Register (TOCR)..... | 318 |
| 12.2.2 | Timer Start Register (TSTR)..... | 319 |
| 12.2.3 | Timer Start Register 2 (TSTR2)..... | 320 |
| 12.2.4 | Timer Constant Registers (TCOR)..... | 321 |
| 12.2.5 | Timer Counters (TCNT)..... | 321 |
| 12.2.6 | Timer Control Registers (TCR)..... | 322 |
| 12.2.7 | Input Capture Register 2 (TCPR2)..... | 326 |
| 12.3 | Operation..... | 327 |
| 12.3.1 | Counter Operation..... | 327 |
| 12.3.2 | Input Capture Function..... | 330 |
| 12.4 | Interrupts..... | 332 |
| 12.5 | Usage Notes..... | 332 |
| 12.5.1 | Register Writes..... | 332 |
| 12.5.2 | TCNT Register Reads..... | 333 |
| 12.5.3 | Resetting the RTC Frequency Divider..... | 333 |
| 12.5.4 | External Clock Frequency..... | 333 |
| Section 13 | Bus State Controller (BSC)..... | 335 |
| 13.1 | Overview..... | 335 |
| 13.1.1 | Features..... | 335 |
| 13.1.2 | Block Diagram..... | 337 |
| 13.1.3 | Pin Configuration..... | 338 |
| 13.1.4 | Register Configuration..... | 340 |
| 13.1.5 | Overview of Areas..... | 341 |
| 13.1.6 | PCMCIA Support..... | 344 |
| 13.2 | Register Descriptions..... | 348 |
| 13.2.1 | Bus Control Register 1 (BCR1)..... | 348 |
| 13.2.2 | Bus Control Register 2 (BCR2)..... | 357 |
| 13.2.3 | Bus Control Register 3 (BCR3) (SH7751R Only)..... | 359 |
| 13.2.4 | Bus Control Register 4 (BCR4) (SH7751R Only)..... | 361 |
| 13.2.5 | Wait Control Register 1 (WCR1)..... | 363 |
| 13.2.6 | Wait Control Register 2 (WCR2)..... | 366 |
| 13.2.7 | Wait Control Register 3 (WCR3)..... | 374 |
| 13.2.8 | Memory Control Register (MCR)..... | 376 |
| 13.2.9 | PCMCIA Control Register (PCR)..... | 383 |
| 13.2.10 | Synchronous DRAM Mode Register (SDMR)..... | 386 |
| 13.2.11 | Refresh Timer Control/Status Register (RTCSR)..... | 388 |
| 13.2.12 | Refresh Timer Counter (RTCNT)..... | 390 |

| | |
|--|---------|
| 13.2.13 Refresh Time Constant Register (RTCOR) | 391 |
| 13.2.14 Refresh Count Register (RFCR) | 392 |
| 13.2.15 Notes on Accessing Refresh Control Registers..... | 392 |
| 13.3 Operation | 393 |
| 13.3.1 Endian/Access Size and Data Alignment..... | 393 |
| 13.3.2 Areas | 400 |
| 13.3.3 SRAM Interface..... | 405 |
| 13.3.4 DRAM Interface | 413 |
| 13.3.5 Synchronous DRAM Interface..... | 427 |
| 13.3.6 Burst ROM Interface..... | 457 |
| 13.3.7 PCMCIA Interface | 460 |
| 13.3.8 MPX Interface..... | 471 |
| 13.3.9 Byte Control SRAM Interface | 485 |
| 13.3.10 Waits between Access Cycles..... | 489 |
| 13.3.11 Bus Arbitration | 490 |
| 13.3.12 Master Mode | 493 |
| 13.3.13 Slave Mode | 494 |
| 13.3.14 Cooperation between Master and Slave..... | 495 |
| 13.3.15 Notes on Usage | 495 |
| Section 14 Direct Memory Access Controller (DMAC) | 497 |
| 14.1 Overview..... | 497 |
| 14.1.1 Features..... | 497 |
| 14.1.2 Block Diagram (SH7751) | 500 |
| 14.1.3 Pin Configuration (SH7751)..... | 501 |
| 14.1.4 Register Configuration (SH7751)..... | 502 |
| 14.2 Register Descriptions | 504 |
| 14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3) | 504 |
| 14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)..... | 505 |
| 14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3)..... | 506 |
| 14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)..... | 507 |
| 14.2.5 DMA Operation Register (DMAOR) | 515 |
| 14.3 Operation | 517 |
| 14.3.1 DMA Transfer Procedure | 517 |
| 14.3.2 DMA Transfer Requests | 520 |
| 14.3.3 Channel Priorities | 523 |
| 14.3.4 Types of DMA Transfer..... | 526 |
| 14.3.5 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sampling Timing | 535 |
| 14.3.6 Ending DMA Transfer | 549 |
| 14.4 Examples of Use | 552 |

| | | |
|---|--|------------|
| 14.4.1 | Examples of Transfer between External Memory and an External Device with DACK..... | 552 |
| 14.5 | On-Demand Data Transfer Mode (DDT Mode) | 553 |
| 14.5.1 | Operation | 553 |
| 14.5.2 | Pins in DDT Mode..... | 555 |
| 14.5.3 | Transfer Request Acceptance on Each Channel | 558 |
| 14.5.4 | Notes on Use of DDT Module | 580 |
| 14.6 | Configuration of the DMAC (SH7751R)..... | 583 |
| 14.6.1 | Block Diagram of the DMAC..... | 583 |
| 14.6.2 | Pin Configuration (SH7751R) | 584 |
| 14.6.3 | Register Configuration (SH7751R) | 585 |
| 14.7 | Register Descriptions (SH7751R)..... | 588 |
| 14.7.1 | DMA Source Address Registers 0–7 (SAR0–SAR7)..... | 588 |
| 14.7.2 | DMA Destination Address Registers 0–7 (DAR0–DAR7) | 588 |
| 14.7.3 | DMA Transfer Count Registers 0–7 (DMATCR0–DMATCR7) | 589 |
| 14.7.4 | DMA Channel Control Registers 0–7 (CHCR0–CHCR7) | 589 |
| 14.7.5 | DMA Operation Register (DMAOR) | 593 |
| 14.8 | Operation (SH7751R)..... | 595 |
| 14.8.1 | Channel Specification for a Normal DMA Transfer..... | 595 |
| 14.8.2 | Channel Specification for DDT-Mode DMA Transfer..... | 595 |
| 14.8.3 | Transfer Channel Notification in DDT Mode..... | 596 |
| 14.8.4 | Clearing Request Queues by DTR Format | 597 |
| 14.8.5 | Interrupt-Request Codes | 597 |
| 14.9 | Usage Notes | 600 |
| Section 15 Serial Communication Interface (SCI)..... | | 603 |
| 15.1 | Overview..... | 603 |
| 15.1.1 | Features..... | 603 |
| 15.1.2 | Block Diagram..... | 605 |
| 15.1.3 | Pin Configuration..... | 606 |
| 15.1.4 | Register Configuration..... | 606 |
| 15.2 | Register Descriptions | 607 |
| 15.2.1 | Receive Shift Register (SCRSR1) | 607 |
| 15.2.2 | Receive Data Register (SCRDR1) | 607 |
| 15.2.3 | Transmit Shift Register (SCTSR1) | 608 |
| 15.2.4 | Transmit Data Register (SCTDR1)..... | 608 |
| 15.2.5 | Serial Mode Register (SCSMR1)..... | 609 |
| 15.2.6 | Serial Control Register (SCSCR1)..... | 611 |
| 15.2.7 | Serial Status Register (SCSSR1) | 615 |
| 15.2.8 | Serial Port Register (SCSPTR1) | 619 |

| | | |
|--|--|---------|
| 15.2.9 | Bit Rate Register (SCBRR1) | 623 |
| 15.3 | Operation | 631 |
| 15.3.1 | Overview..... | 631 |
| 15.3.2 | Operation in Asynchronous Mode | 633 |
| 15.3.3 | Multiprocessor Communication Function..... | 644 |
| 15.3.4 | Operation in Synchronous Mode | 655 |
| 15.4 | SCI Interrupt Sources and DMAC | 665 |
| 15.5 | Usage Notes | 666 |
| Section 16 Serial Communication Interface with FIFO (SCIF) | | 671 |
| 16.1 | Overview..... | 671 |
| 16.1.1 | Features..... | 671 |
| 16.1.2 | Block Diagram..... | 673 |
| 16.1.3 | Pin Configuration..... | 674 |
| 16.1.4 | Register Configuration..... | 674 |
| 16.2 | Register Descriptions | 675 |
| 16.2.1 | Receive Shift Register (SCRSR2)..... | 675 |
| 16.2.2 | Receive FIFO Data Register (SCFRDR2) | 675 |
| 16.2.3 | Transmit Shift Register (SCTSR2) | 676 |
| 16.2.4 | Transmit FIFO Data Register (SCFTDR2)..... | 676 |
| 16.2.5 | Serial Mode Register (SCSMR2)..... | 677 |
| 16.2.6 | Serial Control Register (SCSCR2)..... | 679 |
| 16.2.7 | Serial Status Register (SCFSR2)..... | 682 |
| 16.2.8 | Bit Rate Register (SCBRR2) | 688 |
| 16.2.9 | FIFO Control Register (SCFCR2) | 689 |
| 16.2.10 | FIFO Data Count Register (SCFDR2)..... | 692 |
| 16.2.11 | Serial Port Register (SCSPTR2) | 693 |
| 16.2.12 | Line Status Register (SCLSR2) | 700 |
| 16.3 | Operation | 701 |
| 16.3.1 | Overview..... | 701 |
| 16.3.2 | Serial Operation | 703 |
| 16.4 | SCIF Interrupt Sources and the DMAC | 713 |
| 16.5 | Usage Notes | 714 |
| Section 17 Smart Card Interface | | 719 |
| 17.1 | Overview..... | 719 |
| 17.1.1 | Features..... | 719 |
| 17.1.2 | Block Diagram..... | 720 |
| 17.1.3 | Pin Configuration..... | 721 |
| 17.1.4 | Register Configuration..... | 721 |

| | | |
|---|--|-----|
| 17.2 | Register Descriptions | 722 |
| 17.2.1 | Smart Card Mode Register (SCSCMR1)..... | 722 |
| 17.2.2 | Serial Mode Register (SCSMR1)..... | 723 |
| 17.2.3 | Serial Control Register (SCSCR1)..... | 724 |
| 17.2.4 | Serial Status Register (SCSSR1) | 725 |
| 17.3 | Operation | 726 |
| 17.3.1 | Overview | 726 |
| 17.3.2 | Pin Connections..... | 727 |
| 17.3.3 | Data Format | 728 |
| 17.3.4 | Register Settings | 729 |
| 17.3.5 | Clock..... | 731 |
| 17.3.6 | Data Transfer Operations..... | 734 |
| 17.4 | Usage Notes | 741 |
| Section 18 I/O Ports..... | | 747 |
| 18.1 | Overview..... | 747 |
| 18.1.1 | Features..... | 747 |
| 18.1.2 | Block Diagrams | 748 |
| 18.1.3 | Pin Configuration..... | 755 |
| 18.1.4 | Register Configuration..... | 758 |
| 18.2 | Register Descriptions..... | 759 |
| 18.2.1 | Port Control Register A (PCTRA)..... | 759 |
| 18.2.2 | Port Data Register A (PDTRA) | 760 |
| 18.2.3 | Port Control Register B (PCTRB) | 761 |
| 18.2.4 | Port Data Register B (PDTRB)..... | 762 |
| 18.2.5 | GPIO Interrupt Control Register (GPIOIC)..... | 763 |
| 18.2.6 | Serial Port Register (SCSPTR1) | 764 |
| 18.2.7 | Serial Port Register (SCSPTR2) | 766 |
| Section 19 Interrupt Controller (INTC)..... | | 769 |
| 19.1 | Overview..... | 769 |
| 19.1.1 | Features..... | 769 |
| 19.1.2 | Block Diagram..... | 769 |
| 19.1.3 | Pin Configuration..... | 771 |
| 19.1.4 | Register Configuration..... | 771 |
| 19.2 | Interrupt Sources..... | 772 |
| 19.2.1 | NMI Interrupt..... | 772 |
| 19.2.2 | IRL Interrupts | 773 |
| 19.2.3 | On-Chip Peripheral Module Interrupts | 775 |
| 19.2.4 | Interrupt Exception Handling and Priority..... | 776 |

| | | |
|--|---|-----|
| 19.3 | Register Descriptions | 780 |
| 19.3.1 | Interrupt Priority Registers A to D (IPRA–IPRD) | 780 |
| 19.3.2 | Interrupt Control Register (ICR)..... | 781 |
| 19.3.3 | Interrupt Priority Level Setting Register 00 (INTPRI00) | 783 |
| 19.3.4 | Interrupt Factor Register 00 (INTREQ00)..... | 784 |
| 19.3.5 | Interrupt Mask Register 00 (INTMSK00)..... | 784 |
| 19.3.6 | Interrupt Mask Clear Register 00 (INTMSKCLR00) | 785 |
| 19.3.7 | INTREQ00, INTMSK00, and INTMSKCLR00 Bit Allocation | 786 |
| 19.4 | INTC Operation | 787 |
| 19.4.1 | Interrupt Operation Sequence | 787 |
| 19.4.2 | Multiple Interrupts | 789 |
| 19.4.3 | Interrupt Masking with MAI Bit..... | 789 |
| 19.5 | Interrupt Response Time..... | 790 |
| 19.6 | Usage Notes | 791 |
| 19.6.1 | NMI Interrupts (SH7751 Only)..... | 791 |
| Section 20 User Break Controller (UBC) | | 795 |
| 20.1 | Overview..... | 795 |
| 20.1.1 | Features..... | 795 |
| 20.1.2 | Block Diagram..... | 796 |
| 20.2 | Register Descriptions | 798 |
| 20.2.1 | Access to UBC Registers | 798 |
| 20.2.2 | Break Address Register A (BARA) | 799 |
| 20.2.3 | Break ASID Register A (BASRA)..... | 800 |
| 20.2.4 | Break Address Mask Register A (BAMRA)..... | 800 |
| 20.2.5 | Break Bus Cycle Register A (BBRA)..... | 801 |
| 20.2.6 | Break Address Register B (BARB) | 803 |
| 20.2.7 | Break ASID Register B (BASRB) | 803 |
| 20.2.8 | Break Address Mask Register B (BAMRB) | 803 |
| 20.2.9 | Break Data Register B (BDRB)..... | 803 |
| 20.2.10 | Break Data Mask Register B (BDMRB)..... | 804 |
| 20.2.11 | Break Bus Cycle Register B (BBRB) | 805 |
| 20.2.12 | Break Control Register (BRCR) | 805 |
| 20.3 | Operation | 808 |
| 20.3.1 | Explanation of Terms Relating to Accesses..... | 808 |
| 20.3.2 | Explanation of Terms Relating to Instruction Intervals | 808 |
| 20.3.3 | User Break Operation Sequence | 809 |
| 20.3.4 | Instruction Access Cycle Break | 810 |
| 20.3.5 | Operand Access Cycle Break..... | 811 |
| 20.3.6 | Condition Match Flag Setting..... | 812 |

| | | |
|---|--|------------|
| 20.3.7 | Program Counter (PC) Value Saved | 812 |
| 20.3.8 | Contiguous A and B Settings for Sequential Conditions | 813 |
| 20.3.9 | Usage Notes | 814 |
| 20.4 | User Break Debug Support Function | 816 |
| 20.5 | Examples of Use | 818 |
| 20.6 | User Break Controller Stop Function..... | 820 |
| 20.6.1 | Transition to User Break Controller Stopped State..... | 820 |
| 20.6.2 | Cancelling the User Break Controller Stopped State | 820 |
| 20.6.3 | Examples of Stopping and Restarting the User Break Controller..... | 821 |
| Section 21 High-performance User Debug Interface (H-UDI) | | 823 |
| 21.1 | Overview..... | 823 |
| 21.1.1 | Features..... | 823 |
| 21.1.2 | Block Diagram..... | 823 |
| 21.1.3 | Pin Configuration..... | 825 |
| 21.1.4 | Register Configuration..... | 826 |
| 21.2 | Register Descriptions | 827 |
| 21.2.1 | Instruction Register (SDIR) | 827 |
| 21.2.2 | Data Register (SDDR) | 828 |
| 21.2.3 | Bypass Register (SDBPR) | 828 |
| 21.2.4 | Interrupt Factor Register (SDINT)..... | 829 |
| 21.2.5 | Boundary Scan Register (SDBSR) | 829 |
| 21.3 | Operation | 843 |
| 21.3.1 | TAP Control..... | 843 |
| 21.3.2 | H-UDI Reset | 844 |
| 21.3.3 | H-UDI Interrupt..... | 844 |
| 21.3.4 | Boundary Scan (EXTEST, SAMPLE/PRELOAD, BYPASS) | 845 |
| 21.4 | Usage Notes | 845 |
| Section 22 PCI Controller (PCIC)..... | | 847 |
| 22.1 | Overview..... | 847 |
| 22.1.1 | Features..... | 847 |
| 22.1.2 | Block Diagram..... | 848 |
| 22.1.3 | Pin Configuration..... | 849 |
| 22.1.4 | Register Configuration..... | 850 |
| 22.2 | PCIC Register Descriptions | 856 |
| 22.2.1 | PCI Configuration Register 0 (PCICONF0) | 856 |
| 22.2.2 | PCI Configuration Register 1 (PCICONF1) | 857 |
| 22.2.3 | PCI Configuration Register 2 (PCICONF2) | 863 |
| 22.2.4 | PCI Configuration Register 3 (PCICONF3) | 865 |

| | | |
|---------|---|-----|
| 22.2.5 | PCI Configuration Register 4 (PCICONF4) | 867 |
| 22.2.6 | PCI Configuration Register 5 (PCICONF5) | 869 |
| 22.2.7 | PCI Configuration Register 6 (PCICONF6) | 871 |
| 22.2.8 | PCI Configuration Register 7 (PCICONF7) to PCI Configuration Register 10 (PCICONF10) | 873 |
| 22.2.9 | PCI Configuration Register 11 (PCICONF11) | 874 |
| 22.2.10 | PCI Configuration Register 12 (PCICONF12) | 875 |
| 22.2.11 | PCI Configuration Register 13 (PCICONF13) | 875 |
| 22.2.12 | PCI Configuration Register 14 (PCICONF14) | 876 |
| 22.2.13 | PCI Configuration Register 15 (PCICONF15) | 877 |
| 22.2.14 | PCI Configuration Register 16 (PCICONF16) | 879 |
| 22.2.15 | PCI Configuration Register 17 (PCICONF17) | 881 |
| 22.2.16 | Reserved Area | 883 |
| 22.2.17 | PCI Control Register (PCICR) | 884 |
| 22.2.18 | PCI Local Space Register [1:0] (PCILSR [1:0]) | 888 |
| 22.2.19 | PCI Local Address Register [1:0] (PCILAR [1:0]) | 890 |
| 22.2.20 | PCI Interrupt Register (PCIINT) | 892 |
| 22.2.21 | PCI Interrupt Mask Register (PCIINTM) | 895 |
| 22.2.22 | PCI Address Data Register at Error (PCIALR) | 897 |
| 22.2.23 | PCI Command Data Register at Error (PCICLR) | 898 |
| 22.2.24 | PCI Arbiter Interrupt Register (PCIAINT) | 900 |
| 22.2.25 | PCI Arbiter Interrupt Mask Register (PCIAINTM) | 902 |
| 22.2.26 | PCI Error Bus Master Data Register (PCIBMLR) | 903 |
| 22.2.27 | PCI DMA Transfer Arbitration Register (PCIDMABT) | 904 |
| 22.2.28 | PCI DMA Transfer PCI Address Register [3:0] (PCIDPA [3:0]) | 905 |
| 22.2.29 | PCI DMA Transfer Local Bus Start Address Register [3:0] (PCIDLA [3:0]) | 907 |
| 22.2.30 | PCI DMA Transfer Counter Register [3:0] (PCIDTC [3:0]) | 908 |
| 22.2.31 | PCI DMA Control Register [3:0] (PCIDCR [3:0]) | 910 |
| 22.2.32 | PIO Address Register (PCIPAR) | 913 |
| 22.2.33 | Memory Space Base Register (PCIMBR) | 915 |
| 22.2.34 | I/O Space Base Register (PCIIOBR) | 917 |
| 22.2.35 | PCI Power Management Interrupt Register (PCIPINT) | 918 |
| 22.2.36 | PCI Power Management Interrupt Mask Register (PCIPINTM) | 919 |
| 22.2.37 | PCI Clock Control Register (PCICLKR) | 920 |
| 22.2.38 | PCIC-BSC Registers | 921 |
| 22.2.39 | Port Control Register (PCIPCTR) | 923 |
| 22.2.40 | Port Data Register (PCIPDTR) | 926 |
| 22.2.41 | PIO Data Register (PCIPDR) | 927 |
| 22.3 | Description of Operation | 928 |
| 22.3.1 | Operating Modes | 928 |

| | | |
|---------|---|-----|
| 22.3.2 | PCI Commands | 929 |
| 22.3.3 | PCIC Initialization | 930 |
| 22.3.4 | Local Register Access..... | 931 |
| 22.3.5 | Host Functions | 931 |
| 22.3.6 | PCI Bus Arbitration in Non-host Mode | 934 |
| 22.3.7 | PIO Transfers..... | 934 |
| 22.3.8 | Target Transfers..... | 937 |
| 22.3.9 | DMA Transfers | 940 |
| 22.3.10 | Transfer Contention within PCIC | 946 |
| 22.3.11 | PCI Bus Basic Interface..... | 947 |
| 22.4 | Endians | 959 |
| 22.4.1 | Internal Bus (Peripheral Bus) Interface for Peripheral Modules..... | 959 |
| 22.4.2 | Endian Control for Local Bus | 961 |
| 22.4.3 | Endian Control in DMA Transfers | 961 |
| 22.4.4 | Endian Control in Target Transfers (Memory Read/Memory Write) | 963 |
| 22.4.5 | Endian Control in Target Transfers (I/O Read/I/O Write)..... | 966 |
| 22.4.6 | Endian Control in Target Transfers (Configuration Read/Configuration Write)..... | 966 |
| 22.5 | Resetting | 968 |
| 22.6 | Interrupts..... | 969 |
| 22.6.1 | Interrupts from PCIC to CPU | 969 |
| 22.6.2 | Interrupts from External PCI Devices..... | 970 |
| 22.6.3 | INTA..... | 971 |
| 22.7 | Error Detection | 971 |
| 22.8 | PCIC Clock | 971 |
| 22.9 | Power Management | 972 |
| 22.9.1 | Power Management Overview..... | 972 |
| 22.9.2 | Stopping the Clock | 973 |
| 22.9.3 | Compatibility with Standby and Sleep..... | 976 |
| 22.10 | Port Functions | 976 |
| 22.11 | Version Management..... | 977 |
| 22.12 | Usage Notes | 977 |
| 22.12.1 | Notes on Arbiter Interrupt Usage (SH7751 Only)..... | 977 |
| 22.12.2 | Notes on I/O Read and I/O Write Commands (SH7751 Only)..... | 980 |
| 22.12.3 | Notes on Configuration-Read and Configuration-Write Commands (SH7751 Only)..... | 980 |
| 22.12.4 | Notes on Target Read/Write Cycle Timing (SH7751 Only)..... | 980 |
| 22.12.5 | Notes on Parity Error Detection during Master Access | 980 |

| | | |
|------------|--|------|
| Section 23 | Electrical Characteristics | 983 |
| 23.1 | Absolute Maximum Ratings | 983 |
| 23.2 | DC Characteristics | 984 |
| 23.3 | AC Characteristics | 996 |
| 23.3.1 | Clock and Control Signal Timing | 998 |
| 23.3.2 | Control Signal Timing | 1012 |
| 23.3.3 | Bus Timing | 1016 |
| 23.3.4 | Peripheral Module Signal Timing | 1067 |
| 23.3.5 | AC Characteristic Test Conditions | 1081 |
| 23.3.6 | Change in Delay Time Based on Load Capacitance | 1082 |
| Appendix A | Address List..... | 1083 |
| Appendix B | Package Dimensions..... | 1091 |
| Appendix C | Mode Pin Settings..... | 1095 |
| Appendix D | Pin Functions..... | 1099 |
| D.1 | Pin States..... | 1099 |
| D.2 | Handling of Unused Pins | 1104 |
| D.3 | Note on Pin Processing | 1105 |
| Appendix E | Synchronous DRAM Address Multiplexing Tables..... | 1107 |
| Appendix F | Instruction Prefetching and Its Side Effects..... | 1119 |
| Appendix G | Power-On and Power-Off Procedures..... | 1121 |
| G.1 | Power-On Stipulations | 1121 |
| G.2 | Power-Off Stipulations | 1121 |
| G.3 | Common Stipulations for Power-On and Power-Off..... | 1124 |
| Appendix H | Product Lineup | 1125 |
| Appendix I | Version Registers..... | 1127 |

Figures

Section 1 Overview

| | | |
|------------|---|----|
| Figure 1.1 | Block Diagram of SH7751/SH7751R Group Functions | 9 |
| Figure 1.2 | Pin Arrangement (256-Pin QFP) | 10 |
| Figure 1.3 | Pin Arrangement (256-Pin BGA)..... | 11 |
| Figure 1.4 | Pin Arrangement (292-Pin BGA)..... | 12 |

Section 2 Programming Model

| | | |
|------------|--|----|
| Figure 2.1 | Data Formats | 47 |
| Figure 2.2 | CPU Register Configuration in Each Processor Mode..... | 50 |
| Figure 2.3 | General Registers | 52 |
| Figure 2.4 | Floating-Point Registers..... | 54 |
| Figure 2.5 | Data Formats In Memory | 60 |
| Figure 2.6 | Processor State Transitions | 61 |

Section 3 Memory Management Unit (MMU)

| | | |
|-------------|--|-----|
| Figure 3.1 | Role of the MMU | 65 |
| Figure 3.2 | MMU-Related Registers | 67 |
| Figure 3.3 | Physical Address Space (MMUCR.AT = 0) | 71 |
| Figure 3.4 | P4 Area..... | 72 |
| Figure 3.5 | External Memory Space..... | 74 |
| Figure 3.6 | Virtual Address Space (MMUCR.AT = 1)..... | 75 |
| Figure 3.7 | UTLB Configuration..... | 78 |
| Figure 3.8 | Relationship between Page Size and Address Format..... | 79 |
| Figure 3.9 | ITLB Configuration..... | 82 |
| Figure 3.10 | Flowchart of Memory Access Using UTLB | 83 |
| Figure 3.11 | Flowchart of Memory Access Using ITLB | 84 |
| Figure 3.12 | Operation of LDTLB Instruction..... | 86 |
| Figure 3.13 | Memory-Mapped ITLB Address Array..... | 95 |
| Figure 3.14 | Memory-Mapped ITLB Data Array 1 | 96 |
| Figure 3.15 | Memory-Mapped ITLB Data Array 2 | 97 |
| Figure 3.16 | Memory-Mapped UTLB Address Array | 98 |
| Figure 3.17 | Memory-Mapped UTLB Data Array 1..... | 99 |
| Figure 3.18 | Memory-Mapped UTLB Data Array 2..... | 100 |

Section 4 Caches

| | | |
|------------|--|-----|
| Figure 4.1 | Cache and Store Queue Control Registers (CCR)..... | 103 |
| Figure 4.2 | Configuration of Operand Cache (SH7751)..... | 106 |

| | | |
|-------------|---|-----|
| Figure 4.3 | Configuration of Operand Cache (SH7751R) | 107 |
| Figure 4.4 | Configuration of Write-Back Buffer | 111 |
| Figure 4.5 | Configuration of Write-Through Buffer..... | 111 |
| Figure 4.6 | Configuration of Instruction Cache (SH7751) | 117 |
| Figure 4.7 | Configuration of Instruction Cache (SH7751R)..... | 118 |
| Figure 4.8 | Memory-Mapped IC Address Array | 121 |
| Figure 4.9 | Memory-Mapped IC Data Array | 122 |
| Figure 4.10 | Memory-Mapped OC Address Array | 124 |
| Figure 4.11 | Memory-Mapped OC Data Array | 125 |
| Figure 4.12 | Memory-Mapped IC Address Array | 126 |
| Figure 4.13 | Memory-Mapped IC Data Array | 127 |
| Figure 4.14 | Memory-Mapped OC Address Array | 129 |
| Figure 4.15 | Memory-Mapped OC Data Array | 130 |
| Figure 4.16 | Store Queue Configuration..... | 131 |

Section 5 Exceptions

| | | |
|------------|--|-----|
| Figure 5.1 | Register Bit Configurations..... | 138 |
| Figure 5.2 | Instruction Execution and Exception Handling..... | 143 |
| Figure 5.3 | Example of General Exception Acceptance Order..... | 145 |

Section 6 Floating-Point Unit

| | | |
|------------|--|-----|
| Figure 6.1 | Format of Single-Precision Floating-Point Number..... | 173 |
| Figure 6.2 | Format of Double-Precision Floating-Point Number | 174 |
| Figure 6.3 | Single-Precision NaN Bit Pattern..... | 176 |
| Figure 6.4 | Floating-Point Registers..... | 178 |

Section 8 Pipelining

| | | |
|------------|---------------------------------------|-----|
| Figure 8.1 | Basic Pipelines | 212 |
| Figure 8.2 | Instruction Execution Patterns | 213 |
| Figure 8.3 | Examples of Pipelined Execution | 225 |

Section 9 Power-Down Modes

| | | |
|------------|--|-----|
| Figure 9.1 | STATUS Output in Power-On Reset | 255 |
| Figure 9.2 | STATUS Output in Manual Reset | 255 |
| Figure 9.3 | STATUS Output in Standby → Interrupt Sequence | 256 |
| Figure 9.4 | STATUS Output in Standby → Power-On Reset Sequence | 256 |
| Figure 9.5 | STATUS Output in Standby → Manual Reset Sequence | 257 |
| Figure 9.6 | STATUS Output in Sleep → Interrupt Sequence..... | 257 |
| Figure 9.7 | STATUS Output in Sleep → Power-On Reset Sequence | 258 |
| Figure 9.8 | STATUS Output in Sleep → Manual Reset Sequence..... | 259 |

| | | |
|-------------|---|-----|
| Figure 9.9 | STATUS Output in Deep Sleep → Interrupt Sequence | 260 |
| Figure 9.10 | STATUS Output in Deep Sleep → Power-On Reset Sequence | 260 |
| Figure 9.11 | STATUS Output in Deep Sleep → Manual Reset Sequence | 261 |
| Figure 9.12 | Hardware Standby Mode Timing (When CA = Low in Normal Operation)..... | 262 |
| Figure 9.13 | Hardware Standby Mode Timing (When CA = Low in WDT Operation)..... | 263 |
| Figure 9.14 | Timing When Power Other than VDD-RTC Is Off..... | 263 |
| Figure 9.15 | Timing When VDD-RTC Power Is Off → On..... | 264 |

Section 10 Clock Oscillation Circuits

| | | |
|-----------------|--|-----|
| Figure 10.1 (1) | Block Diagram of CPG (SH7751) | 269 |
| Figure 10.1 (2) | Block Diagram of CPG (SH7751R)..... | 270 |
| Figure 10.2 | Block Diagram of WDT..... | 280 |
| Figure 10.3 | Writing to WTCNT and WTCSR..... | 284 |
| Figure 10.4 | Points for Attention when Using Crystal Resonator | 287 |
| Figure 10.5 | Points for Attention when Using PLL Oscillator Circuit | 288 |

Section 11 Realtime Clock (RTC)

| | | |
|-------------|---|-----|
| Figure 11.1 | Block Diagram of RTC | 292 |
| Figure 11.2 | Examples of Time Setting Procedures | 309 |
| Figure 11.3 | Examples of Time Reading Procedures | 311 |
| Figure 11.4 | Example of Use of Alarm Function..... | 312 |
| Figure 11.5 | Example of Crystal Oscillation Circuit Connection | 314 |

Section 12 Timer Unit (TMU)

| | | |
|-------------|--|-----|
| Figure 12.1 | Block Diagram of TMU | 316 |
| Figure 12.2 | Example of Count Operation Setting Procedure | 328 |
| Figure 12.3 | TCNT Auto-Reload Operation..... | 329 |
| Figure 12.4 | Count Timing when Operating on Internal Clock..... | 329 |
| Figure 12.5 | Count Timing when Operating on External Clock..... | 330 |
| Figure 12.6 | Count Timing when Operating on On-Chip RTC Output Clock..... | 330 |
| Figure 12.7 | Operation Timing when Using Input Capture Function..... | 331 |

Section 13 Bus State Controller (BSC)

| | | |
|-------------|--|-----|
| Figure 13.1 | Block Diagram of BSC | 337 |
| Figure 13.2 | Correspondence between Virtual Address Space and External Memory Space..... | 341 |
| Figure 13.3 | External Memory Space Allocation | 343 |
| Figure 13.4 | Example of RDY Sampling Timing at which BCR4 Is Set (Two Wait Cycles Are Inserted by WCR2)..... | 362 |
| Figure 13.5 | Writing to RTCSR, RTCNT, RTCOR, and RFCR..... | 393 |
| Figure 13.6 | Basic Timing of SRAM Interface | 406 |

| | | |
|------------------|--|-----|
| Figure 13.7 | Example of 32-Bit Data Width SRAM Connection..... | 407 |
| Figure 13.8 | Example of 16-Bit Data Width SRAM Connection..... | 408 |
| Figure 13.9 | Example of 8-Bit Data Width SRAM Connection..... | 409 |
| Figure 13.10 | SRAM Interface Wait Timing (Software Wait Only)..... | 410 |
| Figure 13.11 | SRAM Interface Wait State Timing (Wait State Insertion by $\overline{\text{RDY}}$ Signal)..... | 411 |
| Figure 13.12 | SRAM Interface Read Strobe Negate Timing (AnS = 1, AnW = 4, and AnH = 2)..... | 412 |
| Figure 13.13 | Example of DRAM Connection (32-Bit Data Width, Area 3)..... | 413 |
| Figure 13.14 | Basic DRAM Access Timing..... | 415 |
| Figure 13.15 | DRAM Wait State Timing..... | 416 |
| Figure 13.16 | DRAM Burst Access Timing..... | 417 |
| Figure 13.17 | DRAM Bus Cycle (EDO Mode, RCD = 0, AnW = 0, TPC = 1)..... | 418 |
| Figure 13.18 | Burst Access Timing in DRAM EDO Mode..... | 419 |
| Figure 13.19 (1) | DRAM Burst Bus Cycle, RAS Down Mode Start (Fast Page Mode, RCD = 0, AnW = 0)..... | 420 |
| Figure 13.19 (2) | DRAM Burst Bus Cycle, RAS Down Mode Continuation (Fast Page Mode, RCD = 0, AnW = 0)..... | 421 |
| Figure 13.19 (3) | DRAM Burst Bus Cycle, RAS Down Mode Start (EDO Mode, RCD = 0, AnW = 0)..... | 422 |
| Figure 13.19 (4) | DRAM Burst Bus Cycle, RAS Down Mode Continuation (EDO Mode, RCD = 0, AnW = 0)..... | 423 |
| Figure 13.20 | CAS-Before-RAS Refresh Operation..... | 424 |
| Figure 13.21 | DRAM CAS-Before-RAS Refresh Cycle Timing (TRAS = 0, TRC = 1)..... | 425 |
| Figure 13.22 | DRAM Self-Refresh Cycle Timing..... | 426 |
| Figure 13.23 | Example of 32-Bit Data Width Synchronous DRAM Connection (Area 3)..... | 428 |
| Figure 13.24 | Basic Timing for Synchronous DRAM Burst Read..... | 431 |
| Figure 13.25 | Basic Timing for Synchronous DRAM Single Read..... | 433 |
| Figure 13.26 | Basic Timing for Synchronous DRAM Burst Write..... | 434 |
| Figure 13.27 | Basic Timing for Synchronous DRAM Single Write..... | 436 |
| Figure 13.28 | Burst Read Timing..... | 438 |
| Figure 13.29 | Burst Read Timing (RAS Down, Same Row Address)..... | 439 |
| Figure 13.30 | Burst Read Timing (RAS Down, Different Row Addresses)..... | 440 |
| Figure 13.31 | Burst Write Timing..... | 441 |
| Figure 13.32 | Burst Write Timing (Same Row Address)..... | 442 |
| Figure 13.33 | Burst Write Timing (Different Row Addresses)..... | 443 |
| Figure 13.34 | Burst Read Cycle for Different Bank and Row Address Following Preceding Burst Read Cycle..... | 446 |
| Figure 13.35 | Auto-Refresh Operation..... | 448 |
| Figure 13.36 | Synchronous DRAM Auto-Refresh Timing..... | 448 |
| Figure 13.37 | Synchronous DRAM Self-Refresh Timing..... | 450 |

| | |
|---|-----|
| Figure 13.38 (1) Synchronous DRAM Mode Write Timing (PALL)..... | 452 |
| Figure 13.38 (2) Synchronous DRAM Mode Write Timing (Mode Register Setting) | 453 |
| Figure 13.39 Basic Timing of a Burst Read from Synchronous DRAM (Burst Length = 8)..... | 455 |
| Figure 13.40 Basic Timing of a Burst Write to Synchronous DRAM | 456 |
| Figure 13.41 Burst ROM Basic Access Timing..... | 458 |
| Figure 13.42 Burst ROM Wait Access Timing..... | 459 |
| Figure 13.43 Burst ROM Wait Access Timing..... | 460 |
| Figure 13.44 Example of PCMCIA Interface..... | 464 |
| Figure 13.45 Basic Timing for PCMCIA Memory Card Interface | 465 |
| Figure 13.46 Wait Timing for PCMCIA Memory Card Interface..... | 466 |
| Figure 13.47 PCMCIA Space Allocation..... | 467 |
| Figure 13.48 Basic Timing for PCMCIA I/O Card Interface..... | 468 |
| Figure 13.49 Wait Timing for PCMCIA I/O Card Interface..... | 469 |
| Figure 13.50 Dynamic Bus Sizing Timing for PCMCIA I/O Card Interface..... | 470 |
| Figure 13.51 Example of 32-Bit Data Width MPX Connection | 472 |
| Figure 13.52 MPX Interface Timing 1 (Single Read Cycle, AnW = 0, No External Wait)..... | 473 |
| Figure 13.53 MPX Interface Timing 2 (Single Read, AnW = 0, One External Wait Inserted).. | 474 |
| Figure 13.54 MPX Interface Timing 3 (Single Write Cycle, AnW = 0, No External Wait) | 475 |
| Figure 13.55 MPX Interface Timing 4 (Single Write, AnW = 1, One External Wait Inserted) | 476 |
| Figure 13.56 MPX Interface Timing 5 (Burst Read Cycle, AnW = 0, No External Wait) | 477 |
| Figure 13.57 MPX Interface Timing 6 (Burst Read Cycle, AnW = 0, External Wait Control) . | 478 |
| Figure 13.58 MPX Interface Timing 7 (Burst Write Cycle, AnW = 0, No External Wait)..... | 479 |
| Figure 13.59 MPX Interface Timing 8 (Burst Write Cycle, AnW = 1, External Wait Control) | 480 |
| Figure 13.60 MPX Interface Timing 9 (Burst Read Cycle, AnW = 0, No External Wait, Bus Width: 32 Bits, Transfer Data Size: 64 Bits) | 481 |
| Figure 13.61 MPX Interface Timing 10 (Burst Read Cycle, AnW = 0, One External Wait Inserted, Bus Width: 32 Bits, Transfer Data Size: 64 Bits) | 482 |
| Figure 13.62 MPX Interface Timing 11 (Burst Write Cycle, AnW = 0, No External Wait, Bus Width: 32 Bits, Transfer Data Size: 64 Bits) | 483 |
| Figure 13.63 MPX Interface Timing 12 (Burst Write Cycle, AnW = 1, One External Wait Inserted, Bus Width: 32 Bits, Transfer Data Size: 64 Bits) | 484 |
| Figure 13.64 Example of 32-Bit Data Width Byte Control SRAM | 485 |
| Figure 13.65 Byte Control SRAM Basic Read Cycle (No Wait)..... | 486 |
| Figure 13.66 Byte Control SRAM Basic Read Cycle (One Internal Wait Cycle)..... | 487 |
| Figure 13.67 Byte Control SRAM Basic Read Cycle (One Internal Wait + One External Wait)..... | 488 |
| Figure 13.68 Waits between Access Cycles..... | 490 |
| Figure 13.69 Arbitration Sequence | 492 |

Section 14 Direct Memory Access Controller (DMAC)

| | | |
|--------------|---|-----|
| Figure 14.1 | Block Diagram of DMAC..... | 500 |
| Figure 14.2 | DMAC Transfer Flowchart..... | 519 |
| Figure 14.3 | Round Robin Mode..... | 524 |
| Figure 14.4 | Example of Changes in Priority Order in Round Robin Mode..... | 525 |
| Figure 14.5 | Data Flow in Single Address Mode..... | 527 |
| Figure 14.6 | DMA Transfer Timing in Single Address Mode..... | 528 |
| Figure 14.7 | Operation in Dual Address Mode..... | 529 |
| Figure 14.8 | Example of Transfer Timing in Dual Address Mode..... | 530 |
| Figure 14.9 | Example of DMA Transfer in Cycle Steal Mode..... | 531 |
| Figure 14.10 | Example of DMA Transfer in Burst Mode..... | 531 |
| Figure 14.11 | Bus Handling with Two DMAC Channels Operating..... | 535 |
| Figure 14.12 | Dual Address Mode/Cycle Steal Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Level Detection), DACK (Read Cycle)..... | 538 |
| Figure 14.13 | Dual Address Mode/Cycle Steal Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Edge Detection), DACK (Read Cycle)..... | 539 |
| Figure 14.14 | Dual Address Mode/Burst Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Level Detection), DACK (Read Cycle)..... | 540 |
| Figure 14.15 | Dual Address Mode/Burst Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Edge Detection), DACK (Read Cycle)..... | 541 |
| Figure 14.16 | Dual Address Mode/Cycle Steal Mode On-Chip SCI (Level Detection) → External Bus..... | 542 |
| Figure 14.17 | Dual Address Mode/Cycle Steal Mode External Bus → On-Chip SCI (Level Detection)..... | 543 |
| Figure 14.18 | Single Address Mode/Cycle Steal Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Level Detection)..... | 544 |
| Figure 14.19 | Single Address Mode/Cycle Steal Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Edge Detection)..... | 545 |
| Figure 14.20 | Single Address Mode/Burst Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Level Detection)..... | 546 |
| Figure 14.21 | Single Address Mode/Burst Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Edge Detection)..... | 547 |
| Figure 14.22 | Single Address Mode/Burst Mode External Bus → External Bus/ $\overline{\text{DREQ}}$ (Level Detection)/32-Byte Block Transfer (Bus Width: 32 Bits, SDRAM: Row Hit Write)..... | 548 |
| Figure 14.23 | On-Demand Transfer Mode Block Diagram..... | 553 |
| Figure 14.24 | System Configuration in On-Demand Data Transfer Mode..... | 555 |
| Figure 14.25 | Data Transfer Request Format..... | 556 |

| | | |
|--------------|---|-----|
| Figure 14.26 | Single Address Mode/Synchronous DRAM → External Device Longword Transfer SDRAM Auto-Precharge Read Bus Cycle, Burst (RCD = 1, CAS latency = 3, TPC = 3) | 559 |
| Figure 14.27 | Single Address Mode/External Device → Synchronous DRAM Longword Transfer SDRAM Auto-Precharge Write Bus Cycle, Burst (RCD = 1, TRWL = 2, TPC = 1) | 560 |
| Figure 14.28 | Dual Address Mode/Synchronous DRAM → SRAM Longword Transfer | 561 |
| Figure 14.29 | Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer | 562 |
| Figure 14.30 | Single Address Mode/Burst Mode/External Device → External Bus 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer | 562 |
| Figure 14.31 | Single Address Mode/Burst Mode/External Bus → External Device 32-Bit Transfer/Channel 0 On-Demand Data Transfer | 563 |
| Figure 14.32 | Single Address Mode/Burst Mode/External Device → External Bus 32-Bit Transfer/Channel 0 On-Demand Data Transfer | 564 |
| Figure 14.33 | Handshake Protocol Using Data Bus (Channel 0 On-Demand Data Transfer).... | 565 |
| Figure 14.34 | Handshake Protocol without Use of Data Bus (Channel 0 On-Demand Data Transfer)..... | 566 |
| Figure 14.35 | Read from Synchronous DRAM Precharge Bank..... | 567 |
| Figure 14.36 | Read from Synchronous DRAM Non-Precharge Bank (Row Miss)..... | 567 |
| Figure 14.37 | Read from Synchronous DRAM (Row Hit)..... | 568 |
| Figure 14.38 | Write to Synchronous DRAM Precharge Bank | 568 |
| Figure 14.39 | Write to Synchronous DRAM Non-Precharge Bank (Row Miss) | 569 |
| Figure 14.40 | Write to Synchronous DRAM (Row Hit) | 569 |
| Figure 14.41 | Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer | 570 |
| Figure 14.42 | DDT Mode Setting..... | 571 |
| Figure 14.43 | Single Address Mode/Burst Mode/Edge Detection/ External Device → External Bus Data Transfer..... | 571 |
| Figure 14.44 | Single Address Mode/Burst Mode/Level Detection/ External Bus → External Device Data Transfer..... | 572 |
| Figure 14.45 | Single Address Mode/Burst Mode/Edge Detection/Byte, Word, Longword, Quadword/External Bus → External Device Data Transfer | 572 |
| Figure 14.46 | Single Address Mode/Burst Mode/Edge Detection/Byte, Word, Longword, Quadword/External Device → External Bus Data Transfer | 573 |
| Figure 14.47 | Single Address Mode/Burst Mode/32-Byte Block Transfer/DMA Transfer Request to Channels 1–3 Using Data Bus..... | 574 |
| Figure 14.48 | Single Address Mode/Burst Mode/32-Byte Block Transfer/ External Bus → External Device Data Transfer/ Direct Data Transfer Request to Channel 2 without Using Data Bus | 575 |

| | | |
|--------------|---|-----|
| Figure 14.49 | Single Address Mode/Burst Mode/External Bus → External Device Data Transfer/Direct Data Transfer Request to Channel 2..... | 576 |
| Figure 14.50 | Single Address Mode/Burst Mode/External Device → External Bus Data Transfer/Direct Data Transfer Request to Channel 2..... | 577 |
| Figure 14.51 | Single Address Mode/Burst Mode/External Bus → External Device Data Transfer (Active Bank Address)/Direct Data Transfer Request to Channel 2 | 578 |
| Figure 14.52 | Single Address Mode/Burst Mode/External Device → External Bus Data Transfer (Active Bank Address)/Direct Data Transfer Request to Channel 2 | 579 |
| Figure 14.53 | Block Diagram of the DMAC..... | 583 |
| Figure 14.54 | DTR Format (Transfer Request Format) (SH7751R) | 594 |
| Figure 14.55 | Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer | 598 |
| Figure 14.56 | Single Address Mode/Cycle Steal Mode/External Bus → External Device/ 32-Byte Block Transfer/On-Demand Data Transfer on Channel 4..... | 599 |

Section 15 Serial Communication Interface (SCI)

| | | |
|--------------|---|-----|
| Figure 15.1 | Block Diagram of SCI | 605 |
| Figure 15.2 | SCK Pin | 621 |
| Figure 15.3 | TxD Pin..... | 622 |
| Figure 15.4 | RxD Pin | 622 |
| Figure 15.5 | Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits) | 634 |
| Figure 15.6 | Relation between Output Clock and Transfer Data Phase (Asynchronous Mode) | 636 |
| Figure 15.7 | Sample SCI Initialization Flowchart..... | 637 |
| Figure 15.8 | Sample Serial Transmission Flowchart..... | 638 |
| Figure 15.9 | Example of Transmit Operation in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)..... | 640 |
| Figure 15.10 | Sample Serial Reception Flowchart (1) | 641 |
| Figure 15.11 | Example of SCI Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit)..... | 644 |
| Figure 15.12 | Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)..... | 645 |
| Figure 15.13 | Sample Multiprocessor Serial Transmission Flowchart | 647 |
| Figure 15.14 | Example of SCI Transmit Operation (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit) | 649 |
| Figure 15.15 | Sample Flowchart of Multiprocessor Serial Reception with Interrupt Generation..... | 651 |
| Figure 15.16 | Sample Multiprocessor Serial Reception Flowchart (1) | 652 |
| Figure 15.16 | Sample Multiprocessor Serial Reception Flowchart (2) | 653 |

| | | |
|--------------|--|-----|
| Figure 15.17 | Example of SCI Receive Operation (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit) | 654 |
| Figure 15.18 | Data Format in Synchronous Communication | 655 |
| Figure 15.19 | Sample SCI Initialization Flowchart..... | 657 |
| Figure 15.20 | Sample Serial Transmission Flowchart..... | 658 |
| Figure 15.21 | Example of SCI Transmit Operation..... | 660 |
| Figure 15.22 | Sample Serial Reception Flowchart (1) | 661 |
| Figure 15.23 | Example of SCI Receive Operation | 663 |
| Figure 15.24 | Sample Flowchart for Serial Data Transmission and Reception..... | 664 |
| Figure 15.25 | Receive Data Sampling Timing in Asynchronous Mode..... | 668 |
| Figure 15.26 | Example of Synchronous Transmission by DMAC | 669 |

Section 16 Serial Communication Interface with FIFO (SCIF)

| | | |
|--------------|--|-----|
| Figure 16.1 | Block Diagram of SCIF | 673 |
| Figure 16.2 | MD8/ $\overline{\text{RTS2}}$ Pin | 696 |
| Figure 16.3 | MD7/ $\overline{\text{CTS2}}$ Pin | 697 |
| Figure 16.4 | MD1/TxD2 Pin | 698 |
| Figure 16.5 | MD2/RxD2 Pin | 698 |
| Figure 16.6 | MD0/SCK2 Pin..... | 699 |
| Figure 16.7 | Sample SCIF Initialization Flowchart..... | 705 |
| Figure 16.8 | Sample Serial Transmission Flowchart..... | 706 |
| Figure 16.9 | Example of Transmit Operation (Example with 8-Bit Data, Parity, One Stop Bit)..... | 708 |
| Figure 16.10 | Example of Operation Using Modem Control (CTS2) | 708 |
| Figure 16.11 | Sample Serial Reception Flowchart (1) | 709 |
| Figure 16.11 | Sample Serial Reception Flowchart (2) | 710 |
| Figure 16.12 | Example of SCIF Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit) | 712 |
| Figure 16.13 | Example of Operation Using Modem Control (RTS2) | 712 |
| Figure 16.14 | Receive Data Sampling Timing in Asynchronous Mode..... | 716 |

Section 17 Smart Card Interface

| | | |
|-------------|---|-----|
| Figure 17.1 | Block Diagram of Smart Card Interface | 720 |
| Figure 17.2 | Schematic Diagram of Smart Card Interface Pin Connections | 727 |
| Figure 17.3 | Smart Card Interface Data Format | 728 |
| Figure 17.4 | TEND Generation Timing | 730 |
| Figure 17.5 | Sample Start Character Waveforms..... | 731 |
| Figure 17.6 | Difference in Clock Output According to GM Bit Setting | 734 |
| Figure 17.7 | Sample Initialization Flowchart..... | 735 |
| Figure 17.8 | Sample Transmission Processing Flowchart..... | 737 |

| | | |
|--------------|---|-----|
| Figure 17.9 | Sample Reception Processing Flowchart..... | 739 |
| Figure 17.10 | Receive Data Sampling Timing in Smart Card Mode | 741 |
| Figure 17.11 | Retransfer Operation in SCI Receive Mode | 743 |
| Figure 17.12 | Retransfer Operation in SCI Transmit Mode..... | 743 |
| Figure 17.13 | Procedure for Stopping and Restarting the Clock..... | 744 |

Section 18 I/O Ports

| | | |
|--------------|---|-----|
| Figure 18.1 | 16-Bit Port A..... | 748 |
| Figure 18.2 | 16-Bit Port B..... | 749 |
| Figure 18.3 | SCK Pin | 750 |
| Figure 18.4 | TxD Pin..... | 751 |
| Figure 18.5 | RxD Pin | 751 |
| Figure 18.6 | MD1/TxD2 Pin | 752 |
| Figure 18.7 | MD2/RxD2 Pin | 752 |
| Figure 18.8 | MD0/SCK2 Pin..... | 753 |
| Figure 18.9 | MD7/ $\overline{\text{CTS2}}$ Pin | 754 |
| Figure 18.10 | MD8/ $\overline{\text{RTS2}}$ Pin | 755 |

Section 19 Interrupt Controller (INTC)

| | | |
|-------------|---|-----|
| Figure 19.1 | Block Diagram of INTC | 770 |
| Figure 19.2 | Example of IRL Interrupt Connection | 773 |
| Figure 19.3 | Interrupt Operation Flowchart | 788 |

Section 20 User Break Controller (UBC)

| | | |
|-------------|--|-----|
| Figure 20.1 | Block Diagram of User Break Controller | 796 |
| Figure 20.2 | User Break Debug Support Function Flowchart..... | 817 |

Section 21 High-performance User Debug Interface (H-UDI)

| | | |
|-------------|--|-----|
| Figure 21.1 | Block Diagram of H-UDI Circuit | 824 |
| Figure 21.2 | TAP Control State Transition Diagram | 843 |
| Figure 21.3 | H-UDI Reset | 844 |

Section 22 PCI Controller (PCIC)

| | | |
|-------------|--|-----|
| Figure 22.1 | PCIC Block Diagram..... | 848 |
| Figure 22.2 | PIO Memory Space Access | 936 |
| Figure 22.3 | PIO I/O Space Access..... | 937 |
| Figure 22.4 | Local Address Space Accessing Method..... | 938 |
| Figure 22.5 | Example of DMA Transfer Control Register Settings..... | 942 |
| Figure 22.6 | Example of DMA Transfer Flowchart..... | 944 |
| Figure 22.7 | Master Write Cycle in Host Mode (Single) | 948 |

| | | |
|------------------|--|-----|
| Figure 22.8 | Master Read Cycle in Host Mode (Single) | 949 |
| Figure 22.9 | Master Memory Write Cycle in Non-Host Mode (Burst) | 950 |
| Figure 22.10 | Master Memory Read Cycle in Non-Host Mode (Burst) | 951 |
| Figure 22.11 | Target Read Cycle in Non-Host Mode (Single) | 953 |
| Figure 22.12 | Target Write Cycle in Non-Host Mode (Single) | 954 |
| Figure 22.13 | Target Memory Read Cycle in Host Mode (Burst) | 955 |
| Figure 22.14 | Target Memory Write Cycle in Host Mode (Burst) | 956 |
| Figure 22.15 | Master Memory Write Cycle in Host Mode (Burst, With Stepping) | 957 |
| Figure 22.16 | Target Memory Read Cycle in Host Mode (Burst, With Stepping) | 958 |
| Figure 22.17 | Endian Conversion Modes for Peripheral Bus | 959 |
| Figure 22.18 | Peripheral Bus ↔ PCI Bus Data Alignment | 960 |
| Figure 22.19 | Endian Control for Local Bus | 961 |
| Figure 22.20 | Data Alignment at DMA Transfer | 962 |
| Figure 22.21 (1) | Data Alignment at Target Memory Transfer (Big-Endian Local Bus) | 964 |
| Figure 22.21 (2) | Data Alignment at Target Memory Transfer (Little-Endian Local Bus) | 965 |
| Figure 22.22 | Data Alignment at Target I/O Transfer (Both Big Endian and Little Endian) | 966 |
| Figure 22.23 | Data Alignment at Target Configuration Transfer (Both Big Endian and Little Endian) | 967 |
| Figure 22.24 | Target Bus Timeout Interrupt Generation Example 1 (Example in which the Target Device Asserts \overline{STOP} at the Sixteenth Clock Cycle after \overline{FRAME} Was Asserted) | 978 |
| Figure 22.25 | Target Bus Timeout Interrupt Generation Example 2 (Example in which the Target Device Takes 8 Clock Cycles to Prepare for the Third Data Transfer) | 979 |
| Figure 22.26 | Master Bus Timeout Interrupt Generation Example 1 (Example in which the Master Device Prepares the Data and Asserts \overline{IRDY} at the Eighth Clock Cycle after \overline{FRAME} Was Asserted) | 979 |
| Figure 22.27 | Master Bus Timeout Interrupt Generation Example 2 (Example in which the Master Device Takes 8 Clock Cycles to Prepare for the Third Data Transfer following the Second Data Phase) | 980 |

Section 23 Electrical Characteristics

| | | |
|-----------------|---|------|
| Figure 23.1 | EXTAL Clock Input Timing | 1007 |
| Figure 23.2 (1) | CKIO Clock Output Timing | 1007 |
| Figure 23.2 (2) | CKIO Clock Output Timing | 1007 |
| Figure 23.3 | Power-On Oscillation Settling Time | 1008 |
| Figure 23.4 | Standby Return Oscillation Settling Time (Return by \overline{RESET} or \overline{MRESET}) | 1008 |
| Figure 23.5 | Power-On Oscillation Settling Time | 1009 |
| Figure 23.6 | Standby Return Oscillation Settling Time (Return by \overline{RESET} or \overline{MRESET}) | 1009 |
| Figure 23.7 | Standby Return Oscillation Settling Time (Return by NMI) | 1010 |

| | | |
|------------------|---|------|
| Figure 23.8 | Standby Return Oscillation Settling Time (Return by $\overline{IRL3}$ – $\overline{IRL0}$)..... | 1010 |
| Figure 23.9 | PLL Synchronization Settling Time in Case of \overline{RESET} , \overline{MRESET} or NMI Interrupt..... | 1011 |
| Figure 23.10 | PLL Synchronization Settling Time in Case of IRL Interrupt..... | 1011 |
| Figure 23.11 | Control Signal Timing | 1014 |
| Figure 23.12 (1) | Pin Drive Timing for Reset or Sleep Mode | 1014 |
| Figure 23.12 (2) | Pin Drive Timing for Software Standby Mode..... | 1015 |
| Figure 23.13 | SRAM Bus Cycle: Basic Bus Cycle (No Wait)..... | 1020 |
| Figure 23.14 | SRAM Bus Cycle: Basic Bus Cycle (One Internal Wait)..... | 1021 |
| Figure 23.15 | SRAM Bus Cycle: Basic Bus Cycle (One Internal Wait + One External Wait)..... | 1022 |
| Figure 23.16 | SRAM Bus Cycle: Basic Bus Cycle (No Wait, Address Setup/ Hold Time Insertion, AnS = 1, AnH = 1)..... | 1023 |
| Figure 23.17 | Burst ROM Bus Cycle (No Wait)..... | 1024 |
| Figure 23.18 | Burst ROM Bus Cycle (1st Data: One Internal Wait + One External Wait; 2nd/3rd/4th Data: One Internal Wait)..... | 1025 |
| Figure 23.19 | Burst ROM Bus Cycle (No Wait, Address Setup/Hold Time Insertion, AnS = 1, AnH = 1)..... | 1026 |
| Figure 23.20 | Burst ROM Bus Cycle (One Internal Wait + One External Wait)..... | 1027 |
| Figure 23.21 | Synchronous DRAM Auto-Precharge Read Bus Cycle: Single (RCD [1:0] = 01, CAS Latency = 3, TPC [2:0] = 011)..... | 1028 |
| Figure 23.22 | Synchronous DRAM Auto-Precharge Read Bus Cycle: Burst (RCD [1:0] = 01, CAS Latency = 3, TPC [2:0] = 011)..... | 1029 |
| Figure 23.23 | Synchronous DRAM Normal Read Bus Cycle: ACT + READ Commands, Burst (RASD = 1, RCD [1:0] = 01, CAS Latency = 3)..... | 1030 |
| Figure 23.24 | Synchronous DRAM Normal Read Bus Cycle: PRE + ACT + READ Commands, Burst (RASD = 1, RCD [1:0] = 01, TPC [2:0] = 001, CAS Latency = 3)..... | 1031 |
| Figure 23.25 | Synchronous DRAM Normal Read Bus Cycle: READ Command, Burst (RASD = 1, CAS Latency = 3)..... | 1032 |
| Figure 23.26 | Synchronous DRAM Auto-Precharge Write Bus Cycle: Single (RCD [1:0] = 01, TPC [2:0] = 001, TRWL [2:0] = 010)..... | 1033 |
| Figure 23.27 | Synchronous DRAM Auto-Precharge Write Bus Cycle: Burst (RCD [1:0] = 01, TPC [2:0] = 001, TRWL [2:0] = 010)..... | 1034 |
| Figure 23.28 | Synchronous DRAM Normal Write Bus Cycle: ACT + WRITE Commands, Burst (RASD = 1, RCD [1:0] = 01, TRWL [2:0] = 010)..... | 1035 |
| Figure 23.29 | Synchronous DRAM Normal Write Bus Cycle: PRE + ACT + WRITE Commands, Burst (RASD = 1, RCD [1:0] = 01, TPC [2:0] = 001, TRWL [2:0] = 010)..... | 1036 |
| Figure 23.30 | Synchronous DRAM Normal Write Bus Cycle: WRITE Command, Burst (RASD = 1, TRWL [2:0] = 010)..... | 1037 |

| | | |
|------------------|---|------|
| Figure 23.31 | Synchronous DRAM Bus Cycle: Precharge Command (TPC [2:0] = 001)..... | 1038 |
| Figure 23.32 | Synchronous DRAM Bus Cycle: Auto-Refresh (TRAS = 1, TRC [2:0] = 001). | 1039 |
| Figure 23.33 | Synchronous DRAM Bus Cycle: Self-Refresh (TRC [2:0] = 001)..... | 1040 |
| Figure 23.34 (a) | Synchronous DRAM Bus Cycle: Mode Register Setting (PALL)..... | 1041 |
| Figure 23.34 (b) | Synchronous DRAM Bus Cycle: Mode Register Setting (SET)..... | 1042 |
| Figure 23.35 | DRAM Bus Cycles (1) RCD [1:0] = 00, AnW [2:0] = 000, TPC [2:0] = 001 (2) RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 010..... | 1043 |
| Figure 23.36 | DRAM Bus Cycle (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000, TRC [2:0] = 001) | 1044 |
| Figure 23.37 | DRAM Bus Cycle (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000, TPC [2:0] = 001)..... | 1045 |
| Figure 23.38 | DRAM Burst Bus Cycle (EDO Mode, RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 001)..... | 1046 |
| Figure 23.39 | DRAM Burst Bus Cycle (EDO Mode, RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 001, 2-Cycle CAS Negate Pulse Width)..... | 1047 |
| Figure 23.40 | DRAM Burst Bus Cycle: RAS Down Mode State (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000) | 1048 |
| Figure 23.41 | DRAM Burst Bus Cycle: RAS Down Mode Continuation (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000) | 1049 |
| Figure 23.42 | DRAM Burst Bus Cycle (Fast Page Mode, RCD [1:0] = 00, AnW [2:0] = 000, TPC [2:0] = 001) | 1050 |
| Figure 23.43 | DRAM Burst Bus Cycle (Fast Page Mode, RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 001) | 1051 |
| Figure 23.44 | DRAM Burst Bus Cycle (Fast Page Mode, RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 001, 2-Cycle CAS Negate Pulse Width) | 1052 |
| Figure 23.45 | DRAM Burst Bus Cycle: RAS Down Mode State (Fast Page Mode, RCD [1:0] = 00, AnW [2:0] = 000) | 1053 |
| Figure 23.46 | DRAM Burst Bus Cycle: RAS Down Mode Continuation (Fast Page Mode, RCD [1:0] = 00, AnW [2:0] = 000) | 1054 |
| Figure 23.47 | DRAM Bus Cycle: DRAM CAS-Before-RAS Refresh (TRAS [2:0] = 000, TRC [2:0] = 001)..... | 1055 |
| Figure 23.48 | DRAM Bus Cycle: DRAM CAS-Before-RAS Refresh (TRAS [2:0] = 001, TRC [2:0] = 001) | 1056 |
| Figure 23.49 | DRAM Bus Cycle: DRAM Self-Refresh (TRC [2:0] = 001)..... | 1057 |
| Figure 23.50 | PCMCIA Memory Bus Cycle (1) TED [2:0] = 000, TEH [2:0] = 000, No Wait (2) TED [2:0] = 001, TEH [2:0] = 001, One Internal Wait + One External Wait | 1058 |
| Figure 23.51 | PCMCIA I/O Bus Cycle (1) TED [2:0] = 000, TEH [2:0] = 000, No Wait (2) TED [2:0] = 001, TEH [2:0] = 001, One Internal Wait + One External Wait..... | 1059 |

| | | |
|------------------|---|------|
| Figure 23.52 | PCMCIA I/O Bus Cycle (TED [2:0] = 001, TEH [2:0] = 001, One Internal Wait, Bus Sizing)..... | 1060 |
| Figure 23.53 | MPX Basic Bus Cycle: Read (1) 1st Data (One Internal Wait) (2) 1st Data (One Internal Wait + One External Wait)..... | 1061 |
| Figure 23.54 | MPX Basic Bus Cycle: Write (1) 1st Data (No Wait) (2) 1st Data (One Internal Wait) (3) 1st Data (One Internal Wait + One External Wait)..... | 1062 |
| Figure 23.55 | MPX Bus Cycle: Burst Read (1) 1st Data (One Internal Wait), 2nd to 8th Data (No Internal Wait) (2) 1st Data (No Internal Wait), 2nd to 8th Data (No Internal Wait + External Wait Control)..... | 1063 |
| Figure 23.56 | MPX Bus Cycle: Burst Write (1) No Internal Wait (2) 1st Data (One Internal Wait), 2nd to 8th Data (No Internal Wait + External Wait Control)..... | 1064 |
| Figure 23.57 | Memory Byte Control SRAM Bus Cycles (1) Basic Read Cycle (No Wait) (2) Basic Read Cycle (One Internal Wait) (3) Basic Read Cycle (One Internal Wait + One External Wait)..... | 1065 |
| Figure 23.58 | Memory Byte Control SRAM Bus Cycle: Basic Read Cycle (No Wait, Address Setup/Hold Time Insertion, AnS [0] = 1, AnH [1:0] = 01).. | 1066 |
| Figure 23.59 | TCLK Input Timing..... | 1072 |
| Figure 23.60 | RTC Oscillation Settling Time at Power-On | 1072 |
| Figure 23.61 | SCK Input Clock Timing..... | 1072 |
| Figure 23.62 | SCI I/O Synchronous Mode Clock Timing..... | 1073 |
| Figure 23.63 | I/O Port Input/Output Timing | 1073 |
| Figure 23.64 (a) | $\overline{\text{DREQ}}/\overline{\text{DRAK}}$ Timing | 1073 |
| Figure 23.64 (b) | $\overline{\text{DBREQ}}/\overline{\text{TR}}$ Input Timing and BAVL Output Timing | 1074 |
| Figure 23.65 | TCK Input Timing | 1074 |
| Figure 23.66 | $\overline{\text{RESET}}$ Hold Timing | 1075 |
| Figure 23.67 | H-UDI Data Transfer Timing | 1075 |
| Figure 23.68 | Pin Break Timing..... | 1075 |
| Figure 23.69 | NMI Input Timing | 1075 |
| Figure 23.70 | PCI Clock Input Timing | 1078 |
| Figure 23.71 | Output Signal Timing | 1078 |
| Figure 23.72 | Output Signal Timing | 1079 |
| Figure 23.73 | I/O Port Input/Output Timing | 1080 |
| Figure 23.74 | Output Load Circuit..... | 1081 |
| Figure 23.75 | Load Capacitance–Delay Time..... | 1082 |

Appendix B Package Dimensions

| | | |
|------------|--|------|
| Figure B.1 | Package Dimensions (256-pin QFP)..... | 1091 |
| Figure B.2 | Package Dimensions (256-pin BGA: Devices Other than HD6417751RBA240HV)..... | 1092 |

| | | |
|------------|--|------|
| Figure B.3 | Package Dimensions (292-pin BGA) | 1093 |
| Figure B.4 | Package Dimensions (256-pin BGA: HD6417751RBA240HV)..... | 1094 |

Appendix F Instruction Prefetching and Its Side Effects

| | | |
|------------|----------------------------|------|
| Figure F.1 | Instruction Prefetch | 1119 |
|------------|----------------------------|------|

Appendix G Power-On and Power-Off Procedures

| | | |
|------------|--|------|
| Figure G.1 | Method for Temporarily Selecting Clock Operation Mode 6..... | 1123 |
| Figure G.2 | Power-On Procedure 1 | 1124 |
| Figure G.3 | Power-On Procedure 2 | 1124 |

Tables

Section 1 Overview

| | | |
|-----------|-------------------------------------|----|
| Table 1.1 | SH7751/SH7751R Group Features | 2 |
| Table 1.2 | Pin Functions..... | 13 |
| Table 1.3 | Pin Functions..... | 24 |
| Table 1.4 | Pin Functions..... | 35 |

Section 2 Programming Model

| | | |
|-----------|-------------------------------|----|
| Table 2.1 | Initial Register Values | 49 |
|-----------|-------------------------------|----|

Section 3 Memory Management Unit (MMU)

| | | |
|-----------|---------------------|----|
| Table 3.1 | MMU Registers | 66 |
|-----------|---------------------|----|

Section 4 Caches

| | | |
|-----------|-------------------------------|-----|
| Table 4.1 | Cache Features (SH7751)..... | 101 |
| Table 4.2 | Cache Features (SH7751R)..... | 102 |
| Table 4.3 | Store Queue Features | 102 |
| Table 4.4 | Cache Control Registers..... | 102 |

Section 5 Exceptions

| | | |
|-----------|-----------------------------------|-----|
| Table 5.1 | Exception-Related Registers | 137 |
| Table 5.2 | Exceptions | 140 |
| Table 5.3 | Types of Reset..... | 148 |

Section 6 Floating-Point Unit

| | | |
|-----------|--|-----|
| Table 6.1 | Floating-Point Number Formats and Parameters | 174 |
| Table 6.2 | Floating-Point Ranges | 175 |

Section 7 Instruction Set

| | | |
|-----------|---|-----|
| Table 7.1 | Addressing Modes and Effective Addresses | 191 |
| Table 7.2 | Notation Used in Instruction List | 195 |
| Table 7.3 | Fixed-Point Transfer Instructions..... | 196 |
| Table 7.4 | Arithmetic Operation Instructions..... | 198 |
| Table 7.5 | Logic Operation Instructions..... | 200 |
| Table 7.6 | Shift Instructions | 201 |
| Table 7.7 | Branch Instructions | 202 |
| Table 7.8 | System Control Instructions | 203 |
| Table 7.9 | Floating-Point Single-Precision Instructions..... | 205 |

| | | |
|------------|---|-----|
| Table 7.10 | Floating-Point Double-Precision Instructions | 206 |
| Table 7.11 | Floating-Point Control Instructions | 206 |
| Table 7.12 | Floating-Point Graphics Acceleration Instructions | 207 |

Section 8 Pipelining

| | | |
|-----------|------------------------------|-----|
| Table 8.1 | Instruction Groups..... | 218 |
| Table 8.2 | Parallel-Executability | 222 |
| Table 8.3 | Execution Cycles..... | 229 |

Section 9 Power-Down Modes

| | | |
|-----------|---|-----|
| Table 9.1 | Status of CPU and Peripheral Modules in Power-Down Modes..... | 240 |
| Table 9.2 | Power-Down Mode Registers | 241 |
| Table 9.3 | Power-Down Mode Pins | 241 |
| Table 9.4 | State of Registers in Standby Mode | 250 |

Section 10 Clock Oscillation Circuits

| | | |
|----------------|---|-----|
| Table 10.1 | CPG Pins | 272 |
| Table 10.2 | CPG Register..... | 272 |
| Table 10.3 (1) | Clock Operating Modes (SH7751) | 273 |
| Table 10.3 (2) | Clock Operating Modes (SH7751R)..... | 273 |
| Table 10.4 | FRQCR Settings and Internal Clock Frequencies | 274 |
| Table 10.5 | WDT Registers..... | 281 |

Section 11 Realtime Clock (RTC)

| | | |
|------------|---|-----|
| Table 11.1 | RTC Pins | 293 |
| Table 11.2 | RTC Registers | 293 |
| Table 11.3 | Crystal Oscillation Circuit Constants (Recommended Values)..... | 313 |

Section 12 Timer Unit (TMU)

| | | |
|------------|-----------------------------|-----|
| Table 12.1 | TMU Pins..... | 316 |
| Table 12.2 | TMU Registers..... | 317 |
| Table 12.3 | TMU Interrupt Sources | 332 |

Section 13 Bus State Controller (BSC)

| | | |
|------------|--------------------------------------|-----|
| Table 13.1 | BSC Pins | 338 |
| Table 13.2 | BSC Registers | 340 |
| Table 13.3 | External Memory Space Map..... | 342 |
| Table 13.4 | PCMCIA Interface Features..... | 344 |
| Table 13.5 | PCMCIA Support Interfaces | 345 |
| Table 13.6 | Idle Insertion between Accesses..... | 365 |

| | | |
|-------------|--|-----|
| Table 13.7 | When MPX Interface Is Set (Areas 0 to 6)..... | 373 |
| Table 13.8 | 32-Bit External Device/Big-Endian Access and Data Alignment | 394 |
| Table 13.9 | 16-Bit External Device/Big-Endian Access and Data Alignment | 395 |
| Table 13.10 | 8-Bit External Device/Big-Endian Access and Data Alignment | 396 |
| Table 13.11 | 32-Bit External Device/Little-Endian Access and Data Alignment | 397 |
| Table 13.12 | 16-Bit External Device/Little-Endian Access and Data Alignment | 398 |
| Table 13.13 | 8-Bit External Device/Little-Endian Access and Data Alignment | 399 |
| Table 13.14 | Relationship between AMXEXT and AMX2–0 Bits and Address Multiplexing.... | 414 |
| Table 13.15 | Example of Correspondence between LSI and Synchronous DRAM Address Pins (32-Bit Bus Width, AMX2–AMX0 = 000, AMXEXT = 0) | 429 |
| Table 13.16 | Cycles in Which Pipelined Access Can Be Used | 445 |
| Table 13.17 | Relationship between Address and CE When Using PCMCIA Interface | 462 |

Section 14 Direct Memory Access Controller (DMAC)

| | | |
|-------------|--|-----|
| Table 14.1 | DMAC Pins | 501 |
| Table 14.2 | DMAC Pins in DDT Mode | 502 |
| Table 14.3 | DMAC Registers | 503 |
| Table 14.4 | Selecting External Request Mode with RS Bits | 521 |
| Table 14.5 | Selecting On-Chip Peripheral Module Request Mode with RS Bits | 522 |
| Table 14.6 | Supported DMA Transfers | 526 |
| Table 14.7 | Relationship between DMA Transfer Type, Request Mode, and Bus Mode | 532 |
| Table 14.8 | External Request Transfer Sources and Destinations in Normal DMA Mode | 533 |
| Table 14.9 | External Request Transfer Sources and Destinations in DDT Mode | 534 |
| Table 14.10 | Conditions for Transfer between External Memory and an External Device with DACK, and Corresponding Register Settings | 552 |
| Table 14.11 | Usable SZ, ID, and MD Combination in DDT Mode | 557 |
| Table 14.12 | DMAC Pins | 584 |
| Table 14.13 | DMAC Pins in DDT Mode | 585 |
| Table 14.14 | Register Configuration | 586 |
| Table 14.15 | Channel Selection by DTR Format (DMAOR.DBL = 1)..... | 594 |
| Table 14.16 | Notification of Transfer Channel in Eight-Channel DDT Mode..... | 596 |
| Table 14.17 | Function of $\overline{\text{BAVL}}$ | 596 |
| Table 14.18 | DTR Format for Clearing Request Queues | 597 |
| Table 14.19 | DMAC Interrupt-Request Codes | 598 |

Section 15 Serial Communication Interface (SCI)

| | | |
|------------|--|-----|
| Table 15.1 | SCI Pins..... | 606 |
| Table 15.2 | SCI Registers..... | 606 |
| Table 15.3 | Examples of Bit Rates and SCBRR1 Settings in Asynchronous Mode | 625 |
| Table 15.4 | Examples of Bit Rates and SCBRR1 Settings in Synchronous Mode..... | 628 |

| | | |
|-------------|--|-----|
| Table 15.5 | Maximum Bit Rate for Various Frequencies with Baud Rate Generator (Asynchronous Mode)..... | 629 |
| Table 15.6 | Maximum Bit Rate with External Clock Input (Asynchronous Mode)..... | 630 |
| Table 15.7 | Maximum Bit Rate with External Clock Input (Synchronous Mode)..... | 630 |
| Table 15.8 | SCSMR1 Settings for Serial Transfer Format Selection..... | 632 |
| Table 15.9 | SCSMR1 and SCSCR1 Settings for SCI Clock Source Selection | 633 |
| Table 15.10 | Serial Transfer Formats (Asynchronous Mode)..... | 635 |
| Table 15.11 | Receive Error Conditions..... | 643 |
| Table 15.12 | SCI Interrupt Sources..... | 666 |
| Table 15.13 | SCSSR1 Status Flags and Transfer of Receive Data..... | 667 |

Section 16 Serial Communication Interface with FIFO (SCIF)

| | | |
|------------|---|-----|
| Table 16.1 | SCIF Pins | 674 |
| Table 16.2 | SCIF Registers | 674 |
| Table 16.3 | SCSMR2 Settings for Serial Transfer Format Selection..... | 702 |
| Table 16.4 | SCSCR2 Settings for SCIF Clock Source Selection | 702 |
| Table 16.5 | Serial Transfer Formats..... | 703 |
| Table 16.6 | SCIF Interrupt Sources..... | 714 |

Section 17 Smart Card Interface

| | | |
|------------|---|-----|
| Table 17.1 | Smart Card Interface Pins | 721 |
| Table 17.2 | Smart Card Interface Registers..... | 721 |
| Table 17.3 | Smart Card Interface Register Settings | 729 |
| Table 17.4 | Values of n and Corresponding CKS1 and CKS0 Settings | 732 |
| Table 17.5 | Examples of Bit Rate B (bits/s) for Various SCBRR1 Settings (When n = 0)..... | 732 |
| Table 17.6 | Examples of SCBRR1 Settings for Bit Rate B (bits/s) (When n = 0) | 732 |
| Table 17.7 | Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode) | 733 |
| Table 17.8 | Register Settings and SCK Pin State..... | 733 |
| Table 17.9 | Smart Card Mode Operating States and Interrupt Sources..... | 740 |

Section 18 I/O Ports

| | | |
|------------|--|-----|
| Table 18.1 | 32-Bit General-Purpose I/O Port Pins | 755 |
| Table 18.2 | SCI I/O Port Pins..... | 757 |
| Table 18.3 | SCIF I/O Port Pins | 757 |
| Table 18.4 | I/O Port Registers..... | 758 |

Section 19 Interrupt Controller (INTC)

| | | |
|------------|--|-----|
| Table 19.1 | INTC Pins | 771 |
| Table 19.2 | INTC Registers..... | 771 |
| Table 19.3 | IRL3–IRL0 Pins and Interrupt Levels..... | 774 |

| | | |
|------------|---|-----|
| Table 19.4 | Interrupt Exception Handling Sources and Priority Order | 777 |
| Table 19.5 | Interrupt Request Sources and IPRA–IPRD Registers | 781 |
| Table 19.6 | Interrupt Request Sources and INTPRI00 Register | 783 |
| Table 19.7 | Bit Allocation | 786 |
| Table 19.8 | Interrupt Response Time | 790 |

Section 20 User Break Controller (UBC)

| | | |
|------------|--------------------|-----|
| Table 20.1 | UBC Registers..... | 797 |
|------------|--------------------|-----|

Section 21 High-performance User Debug Interface (H-UDI)

| | | |
|------------|---|-----|
| Table 21.1 | H-UDI Pins..... | 825 |
| Table 21.2 | H-UDI Registers..... | 826 |
| Table 21.3 | Structure of Boundary Scan Register | 830 |

Section 22 PCI Controller (PCIC)

| | | |
|-------------|--|-----|
| Table 22.1 | Pin Configuration | 849 |
| Table 22.2 | List of PCI Configuration Registers | 851 |
| Table 22.3 | PCI Configuration Register Configuration..... | 852 |
| Table 22.4 | List of PCIC Local Registers..... | 853 |
| Table 22.5 | List of CLASS23 to 16 Base Class Codes (CLASS23 to 16)..... | 864 |
| Table 22.6 | Memory Space Base Address Register (BASE0)..... | 870 |
| Table 22.7 | Memory Space Base Address Register (BASE1)..... | 872 |
| Table 22.8 | Operating Modes | 928 |
| Table 22.9 | PCI Command Support | 929 |
| Table 22.10 | Access Size..... | 960 |
| Table 22.11 | DMA Transfer Access Size and Endian Conversion Mode | 962 |
| Table 22.12 | Target Transfer Access Size and Endian Conversion Mode | 963 |
| Table 22.13 | Interrupts | 969 |
| Table 22.14 | Method of Stopping Clock per Operating Mode | 974 |

Section 23 Electrical Characteristics

| | | |
|------------|---|-----|
| Table 23.1 | Absolute Maximum Ratings..... | 983 |
| Table 23.2 | DC Characteristics (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV)..... | 984 |
| Table 23.3 | DC Characteristics (HD6417751RF240 (V))..... | 986 |
| Table 23.4 | DC Characteristics (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV* ³)..... | 988 |
| Table 23.5 | DC Characteristics (HD6417751RF200 (V))..... | 990 |
| Table 23.6 | DC Characteristics (HD6417751BP167 (V))..... | 992 |
| Table 23.7 | DC Characteristics (HD6417751F167 (V))..... | 994 |

| | | |
|-------------|---|------|
| Table 23.8 | Permissible Output Currents | 996 |
| Table 23.9 | Clock Timing (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV)..... | 996 |
| Table 23.10 | Clock Timing (HD6417751RF240 (V))..... | 996 |
| Table 23.11 | Clock Timing (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV*)..... | 997 |
| Table 23.12 | Clock Timing (HD6417751RF200 (V))..... | 997 |
| Table 23.13 | Clock Timing (HD6417751BP167 (V), HD6417751F167 (V))..... | 997 |
| Table 23.14 | Clock and Control Signal Timing (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV)..... | 998 |
| Table 23.15 | Clock and Control Signal Timing (HD6417751RF240 (V))..... | 1000 |
| Table 23.16 | Clock and Control Signal Timing (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV* ³)..... | 1002 |
| Table 23.17 | Clock and Control Signal Timing (HD6417751RF200 (V))..... | 1004 |
| Table 23.18 | Clock and Control Signal Timing (HD6417751BP167 (V), HD6417751F167 (V))..... | 1006 |
| Table 23.19 | Control Signal Timing..... | 1012 |
| Table 23.20 | Control Signal Timing..... | 1013 |
| Table 23.21 | Bus Timing (1)..... | 1016 |
| Table 23.22 | Bus Timing (2)..... | 1018 |
| Table 23.23 | Peripheral Module Signal Timing (1) | 1067 |
| Table 23.24 | Peripheral Module Signal Timing (2) | 1070 |
| Table 23.25 | PCIC Signal Timing (in PCIREQ/PCIGNT Non-Port Mode) (1)..... | 1076 |
| Table 23.26 | PCIC Signal Timing (in PCIREQ/PCIGNT Non-Port Mode) (2)..... | 1077 |
| Table 23.27 | PCIC Signal Timing (With PCIREQ/PCIGNT Port Settings in Non-Host Mode) (1)..... | 1079 |
| Table 23.28 | PCIC Signal Timing (With PCIREQ/PCIGNT Port Settings in Non-Host Mode) (2)..... | 1079 |

Appendix A Address List

| | | |
|-----------|--------------------|------|
| Table A.1 | Address List | 1083 |
|-----------|--------------------|------|

Appendix C Mode Pin Settings

| | | |
|-----------|---------------------------------------|------|
| Table C.1 | Clock Operating Modes (SH7751)..... | 1095 |
| Table C.2 | Clock Operating Modes (SH7751R)..... | 1096 |
| Table C.3 | Area 0 Memory Map and Bus Width | 1096 |
| Table C.4 | Endian | 1096 |
| Table C.5 | Master/Slave..... | 1097 |
| Table C.6 | Clock Input..... | 1097 |
| Table C.7 | PCI Mode | 1097 |

Appendix D Pin Functions

| | | |
|-----------|---|------|
| Table D.1 | Pin States in Reset, Power-Down State, and Bus-Released State (PCI Enable, Disable Common) | 1099 |
| Table D.2 | Pin States in Reset, Power-Down State, and Bus-Released State (PCI Enable).... | 1101 |
| Table D.3 | Pin States in Reset, Power-Down State, and Bus-Released State (PCI Disable) .. | 1103 |
| Table D.4 | Handling of Pins When PCI Is Not Used | 1105 |

Appendix H Product Lineup

| | | |
|-----------|-------------------------------------|------|
| Table H.1 | SH7751/SH7751R Product Lineup | 1125 |
|-----------|-------------------------------------|------|

Appendix I Version Registers

| | | |
|-----------|------------------------------|------|
| Table I.1 | Register Configuration | 1127 |
|-----------|------------------------------|------|

Section 1 Overview

1.1 SH7751/SH7751R Group Features

The SH7751/SH7751R Group microprocessor, featuring a built-in PCI bus controller compatible with PCs and multimedia devices. The SuperH™* RISC engine is a Renesas original 32-bit RISC (Reduced Instruction Set Computer) microcomputer. The SuperH™ RISC engine employs a fixed-length 16-bit instruction set, allowing an approximately 50% reduction in program size over a 32-bit instruction set.

The SH7751/SH7751R Group feature the SH-4 Core, which at the object code level is upwardly compatible with the SH-1, SH-2, and SH-3 microcomputers. The SH7751/SH7751R Group have an instruction cache, an operand cache that can be switched between copy-back and write-through modes, a 4-entry full-associative instruction TLB (table look aside buffer), and MMU (memory management unit) with 64-entry full-associative shared TLB.

The SH7751/SH7751R Group also feature a bus state controller (BSC) that can be coupled to DRAM (page/EDO) and synchronous DRAM. Also, because of its built-in functions, such as PCI bus controller, timers, and serial communications functions, required for multimedia and OA equipment, use of the SH7751/SH7751R Group enable a dramatic reduction in system costs.

The features of the SH7751/SH7751R Group are summarized in table 1.1.

Note: * SuperH is a trademark of Renesas Electronics Corp.

Table 1.1 SH7751/SH7751R Group Features

| Item | Features |
|------|---|
| LSI | <ul style="list-style-type: none"> • Superscalar architecture: Parallel execution of two instructions • External buses (SH buses) <ul style="list-style-type: none"> — Separate 26-bit address and 32-bit data buses — External bus frequency of 1, 1/2, 1/3, 1/4, 1/6, or 1/8 times internal bus frequency • External bus (PCI bus): <ul style="list-style-type: none"> — 32-bit address/data multiplexing — Selection of internal clock or external PCI-dedicated clock |
| CPU | <ul style="list-style-type: none"> • Renesas Electronics original SuperH architecture • 32-bit internal data bus • General register file: <ul style="list-style-type: none"> — Sixteen 32-bit general registers (and eight 32-bit shadow registers) — Seven 32-bit control registers — Four 32-bit system registers • RISC-type instruction set (upward-compatible with SuperH Series) <ul style="list-style-type: none"> — Fixed 16-bit instruction length for improved code efficiency — Load-store architecture — Delayed branch instructions — Conditional execution — C-based instruction set • Superscalar architecture (providing simultaneous execution of two instructions) including FPU • Instruction execution time: Maximum 2 instructions/cycle • Virtual address space: 4 Gbytes (448-Mbyte external memory space) • Space identifier ASIDs: 8 bits, 256 virtual address spaces • On-chip multiplier • Five-stage pipeline |

| Item | Features |
|------|---|
| FPU | <ul style="list-style-type: none">• On-chip floating-point coprocessor• Supports single-precision (32 bits) and double-precision (64 bits)• Supports IEEE754-compliant data types and exceptions• Two rounding modes: Round to Nearest and Round to Zero• Handling of denormalized numbers: Truncation to zero or interrupt generation for compliance with IEEE754• Floating-point registers: 32 bits \times 16 \times 2 banks (single-precision 32 bits \times 16 or double-precision 64 bits \times 8) \times 2 banks• 32-bit CPU-FPU floating-point communication register (FPUL)• Supports FMAC (multiply-and-accumulate) instruction• Supports FDIV (divide) and FSQRT (square root) instructions• Supports FLDI0/FLDI1 (load constant 0/1) instructions• Instruction execution times<ul style="list-style-type: none">— Latency (FMAC/FADD/FSUB/FMUL): 3 cycles (single-precision), 8 cycles (double-precision)— Pitch (FMAC/FADD/FSUB/FMUL): 1 cycle (single-precision), 6 cycles (double-precision)Note: FMAC is supported for single-precision only.• 3-D graphics instructions (single-precision only):<ul style="list-style-type: none">— 4-dimensional vector conversion and matrix operations (FTRV): 4 cycles (pitch), 7 cycles (latency)— 4-dimensional vector inner product (FIPR): 1 cycle (pitch), 4 cycles (latency) |

| Item | Features |
|------------------------------|---|
| Clock pulse generator (CPG) | <ul style="list-style-type: none">• Choice of main clock<ul style="list-style-type: none">— SH7751: 1/2, 1, 3, or 6 times EXTAL— SH7751R: 1, 6, or 12 times EXTAL• Clock modes: (Maximum frequency: Varies with models)<ul style="list-style-type: none">— CPU frequency: 1, 1/2, 1/3, 1/4, 1/6, or 1/8 times main clock— Bus frequency: 1, 1/2, 1/3, 1/4, 1/6, or 1/8 times main clock— Peripheral frequency: 1/2, 1/3, 1/4, 1/6, or 1/8 times main clock• Power-down modes<ul style="list-style-type: none">— Sleep mode— Deep sleep mode— Pin sleep mode— Standby mode— Hardware standby mode— Module standby function• Single-channel watchdog timer |
| Memory management unit (MMU) | <ul style="list-style-type: none">• 4-Gbyte address space, 256 address space identifiers (8-bit ASIDs)• Single virtual mode and multiple virtual memory mode• Supports multiple page sizes: 1 Kbyte, 4 Kbytes, 64 Kbytes, 1 Mbyte• 4-entry fully-associative TLB for instructions• 64-entry fully-associative TLB for instructions and operands• Supports software-controlled replacement and random-counter replacement algorithm• TLB contents can be accessed directly by address mapping |

| Item | Features |
|---------------------------|--|
| Cache memory [SH7751] | <ul style="list-style-type: none"> • Instruction cache (IC) <ul style="list-style-type: none"> — 8 Kbytes, direct mapping — 256 entries, 32-byte block length — Normal mode (8-Kbyte cache) — Index mode • Operand cache (OC) <ul style="list-style-type: none"> — 16 Kbytes, direct mapping — 512 entries, 32-byte block length — Normal mode (16-Kbyte cache) — Index mode — RAM mode (8-Kbyte cache + 8-Kbyte RAM) — Choice of write method (copy-back or write-through) • Single-stage copy-back buffer, single-stage write-through buffer • Cache memory contents can be accessed directly by address mapping (usable as on-chip memory) • Store queue (32 bytes × 2 entries) |
| Cache memory [SH7751R] | <ul style="list-style-type: none"> • Instruction cache (IC) <ul style="list-style-type: none"> — 16 Kbytes, 2-way set associative — 256 entries/way, 32-byte block length — Cache-double-mode (16-Kbyte cache) — Index mode — SH7751-compatible mode (8 Kbytes, direct mapping) • Operand cache (OC) <ul style="list-style-type: none"> — 32 Kbytes, 2-way set associative — 512 entries/way, 32-byte block length — Cache-double-mode (32-Kbyte cache) — Index mode — RAM mode (16-Kbyte cache + 16-Kbyte RAM) — Choice of write method (copy-back or write-through) — SH7751-compatible mode (16 Kbytes, direct mapping) • Single-stage copy-back buffer, single-stage write-through buffer • Cache memory contents can be accessed directly by address mapping (usable as on-chip memory) • Store queue (32 bytes × 2 entries) |

| Item | Features |
|-----------------------------|---|
| Interrupt controller (INTC) | <ul style="list-style-type: none">• Five independent external interrupts (NMI, IRL3 to IRL0)• 15-level signed external interrupts: IRL3 to IRL0• On-chip peripheral module interrupts: Priority level can be set for each module |
| User break controller (UBC) | <ul style="list-style-type: none">• Supports debugging by means of user break interrupts• Two break channels• Address, data value, access type, and data size can all be set as break conditions• Supports sequential break function |
| Bus state controller (BSC) | <ul style="list-style-type: none">• Supports external memory access<ul style="list-style-type: none">— 32/16/8-bit external data bus• External memory space divided into seven areas, each of up to 64 Mbytes, with the following parameters settable for each area:<ul style="list-style-type: none">— Bus size (8, 16, or 32 bits)— Number of wait cycles (hardware wait function also supported)— Direct connection of DRAM, synchronous DRAM, and burst ROM possible by setting space type— Supports fast page mode and DRAM EDO— Supports PCMCIA interface— Chip select signals ($\overline{CS0}$ to $\overline{CS6}$) output for relevant areas• DRAM/synchronous DRAM refresh functions<ul style="list-style-type: none">— Programmable refresh interval— Supports CAS-before-RAS refresh mode and self-refresh mode• DRAM/synchronous DRAM burst access function• Big endian or little endian mode can be set |

| Item | Features |
|--|---|
| Direct memory access controller (DMAC) | <ul style="list-style-type: none"> • Physical address DMA controller <ul style="list-style-type: none"> — SH7751: 4-channel — SH7751R: 8-channel • Transfer data size: 8, 16, 32, or 64 bits, or 32 bytes • Address modes: <ul style="list-style-type: none"> — Single address mode — Dual address mode • Transfer requests: External, on-chip peripheral module, or auto-requests • Bus modes: Cycle-steal or burst mode • Supports on-demand data transfer mode (external bus 32 bit) |
| Timer unit (TMU) | <ul style="list-style-type: none"> • 5-channel auto-reload 32-bit timer Input-capture function on one channel • Selection from 7 counter input clocks in 3 of 5 channels and from 5 counter input clocks on remaining 2 of 5 channels |
| Realtime clock (RTC) | <ul style="list-style-type: none"> • On-chip clock and calendar functions • Built-in 32 kHz crystal oscillation circuit with maximum 1/256 second resolution (cycle interrupts) |
| Serial communication interface (SCI, SCIF) | <ul style="list-style-type: none"> • Two full-duplex communication channels (SCI, SCIF) • Channel 1 (SCI): <ul style="list-style-type: none"> — Choice of asynchronous mode or synchronous mode — Supports smart card interface • Channel 2 (SCIF): <ul style="list-style-type: none"> — Supports asynchronous mode — Separate 16-byte FIFOs provided for transmitter and receiver |

| Item | Features |
|---------------------------|--|
| PCI bus controller (PCIC) | <ul style="list-style-type: none"> • PCI bus controller (supports a subset of PCI revision 2.1)* <ul style="list-style-type: none"> — 32-bit bus — 33 MHz/66 MHz support • PCI master/slave support • PCI host function support <ul style="list-style-type: none"> — Built-in bus arbiter • 4 built-in PCI-dedicated DMAC (direct memory access controller) channels <ul style="list-style-type: none"> — Each channel equipped with 64-byte FIFO • Selection of built-in clock or external PCI-dedicated clock • Interrupt requests can be sent to CPU |

| Product lineup | | | Operating | | |
|----------------|--------------|---------|-----------|------------------|-------------|
| | Abbreviation | Voltage | Frequency | Model No. | Package |
| SH7751 | 1.8 V | 167 MHz | | HD6417751BP167 | 256-pin BGA |
| | | | | HD6417751F167 | 256-pin QFP |
| SH7751R | 1.5 V | 240 MHz | | HD6417751RBP240 | 256-pin BGA |
| | | | | HD6417751RBA240H | |
| | | | | HD6417751RF240 | 256-pin QFP |
| | | 200 MHz | | HD6417751RBG240 | 292-pin BGA |
| | | | | HD6417751RBP200 | 256-pin BGA |
| | | | | HD6417751RF200 | 256-pin QFP |
| | | | | HD6417751RBG200 | 292-pin BGA |

Note: * Some items are not compatible with PCI 2.1.
For more information, see section 22.1.1, Features.

1.2 Block Diagram

Figure 1.1 shows an internal block diagram of the SH7751/SH7751R Group.

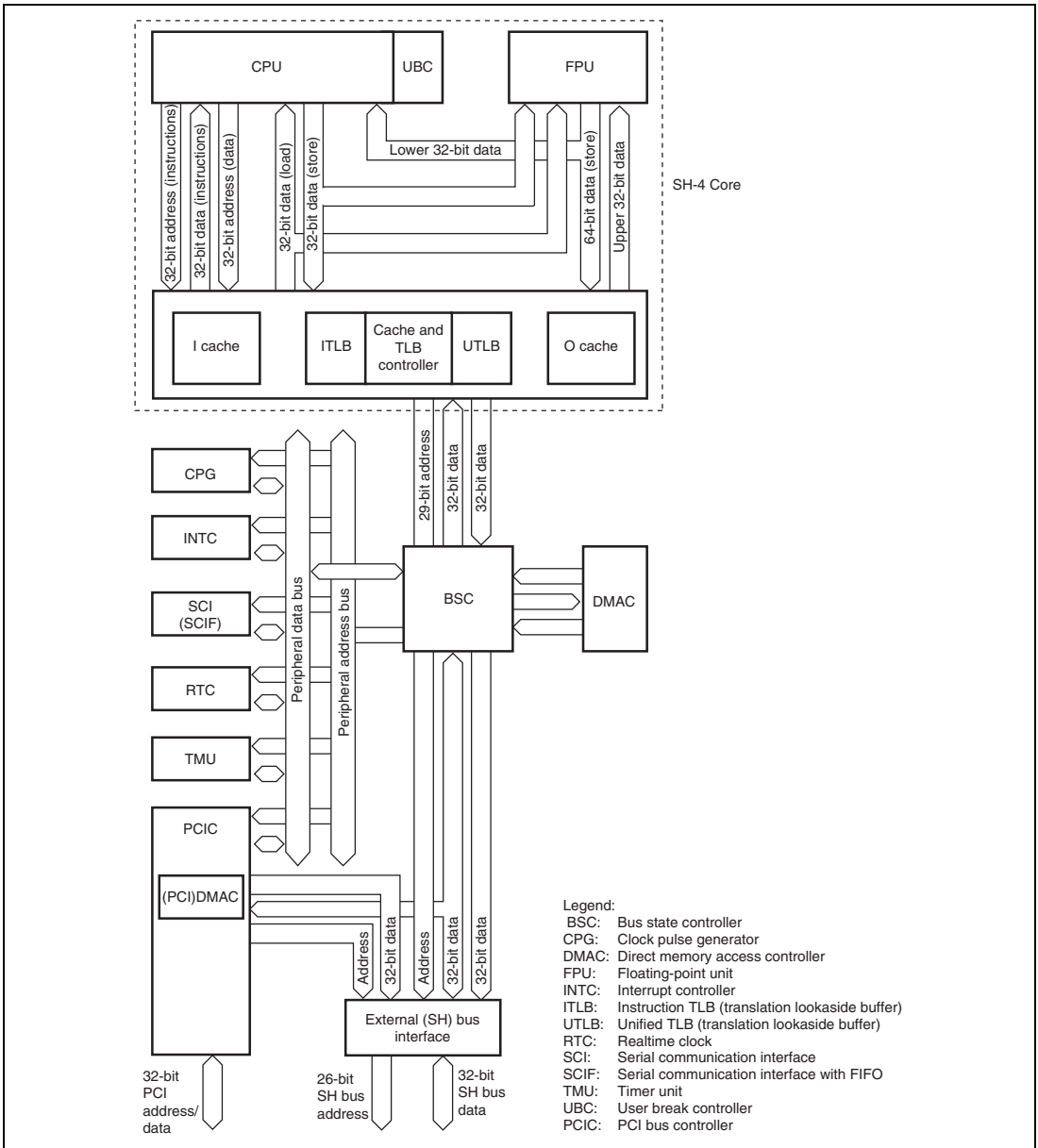


Figure 1.1 Block Diagram of SH7751/SH7751R Group Functions

1.3 Pin Arrangement

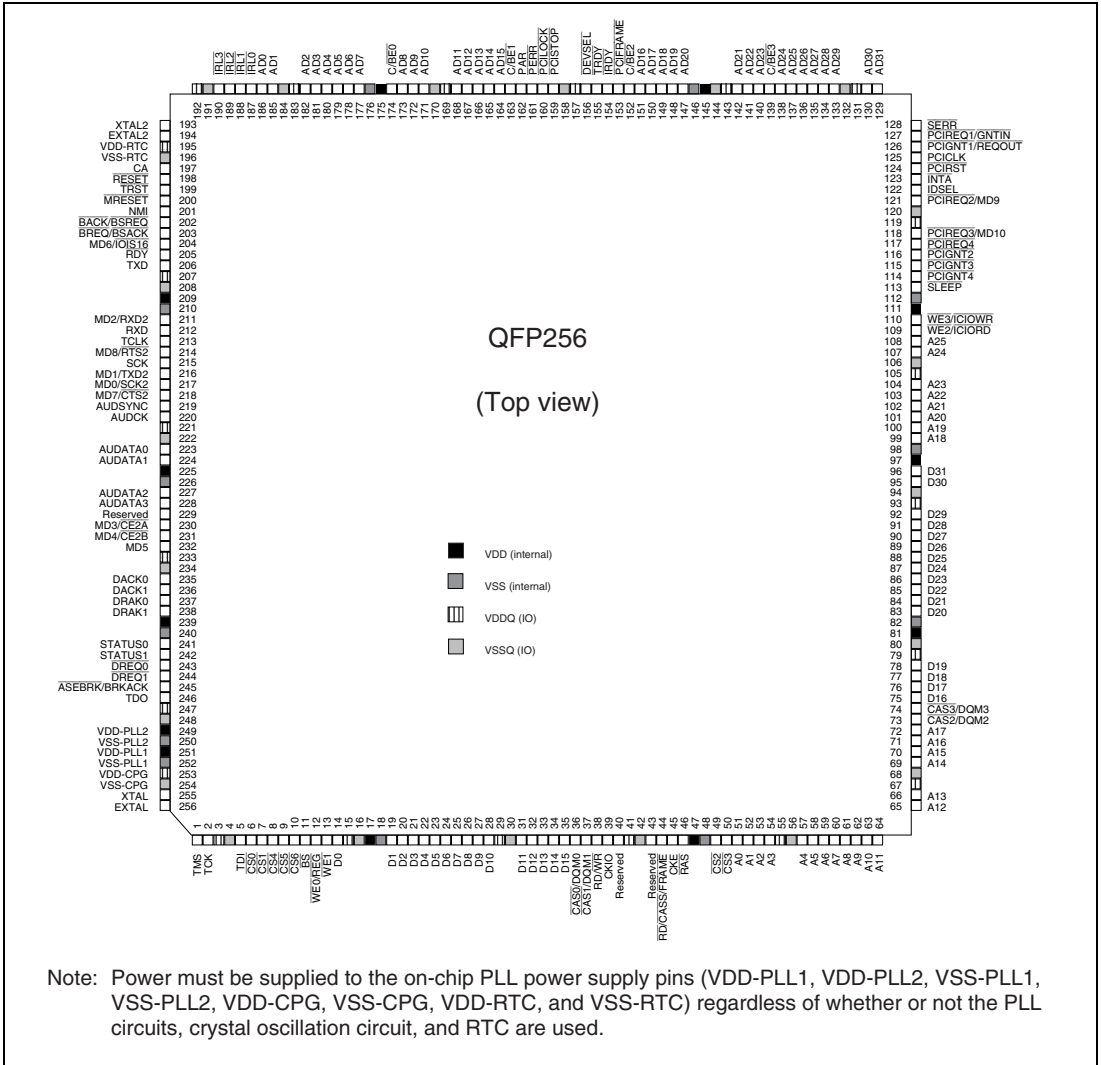
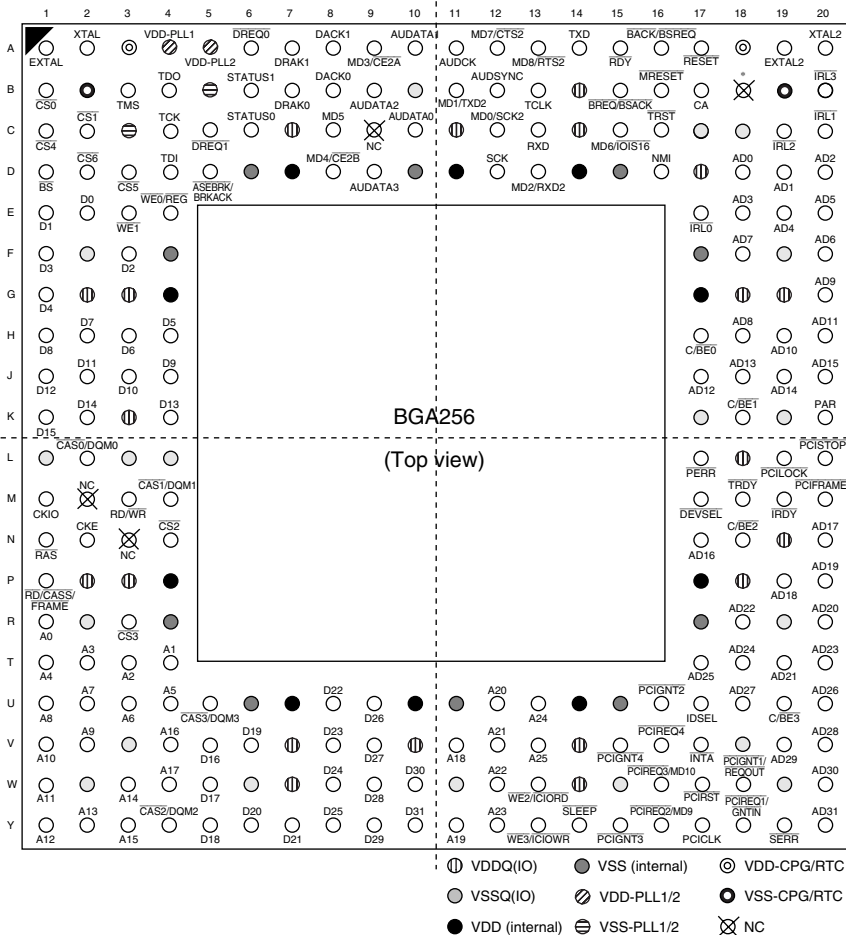


Figure 1.2 Pin Arrangement (256-Pin QFP)

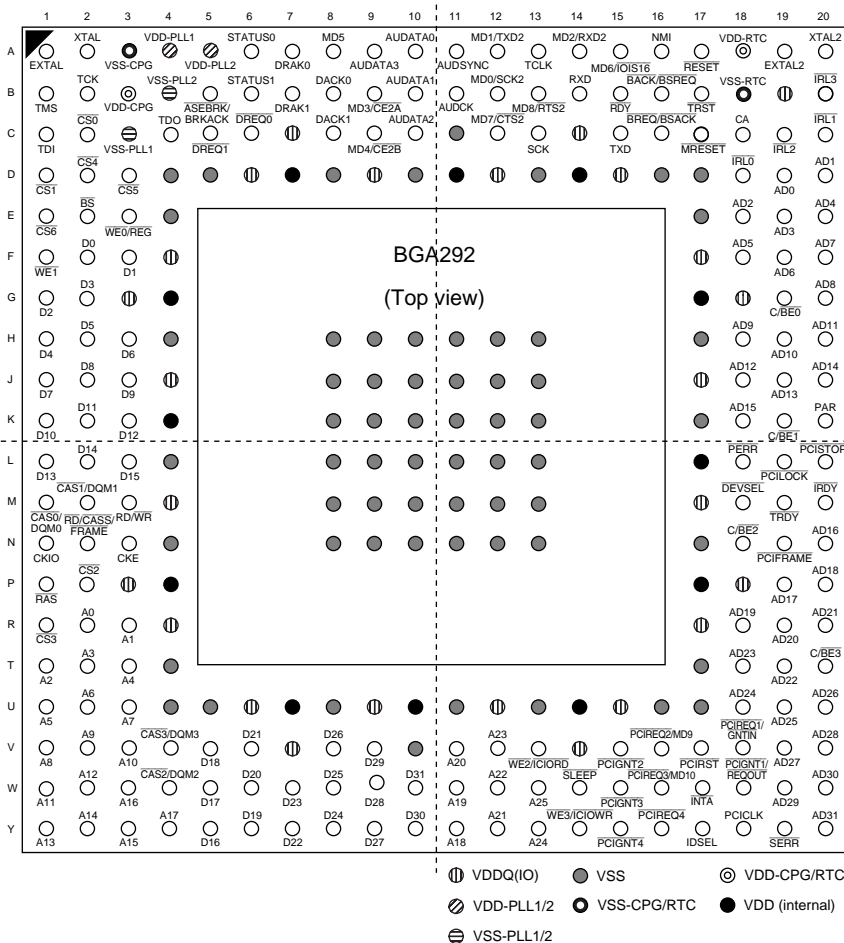
Note: Power must be supplied to the on-chip PLL power supply pins (VDD-PLL1, VDD-PLL2, VSS-PLL1, VSS-PLL2, VDD-CPG, VSS-CPG, VDD-RTC, and VSS-RTC) regardless of whether or not the PLL circuits, crystal oscillation circuit, and RTC are used.



Notes: Power must be supplied to the on-chip PLL power supply pins (VDD-PLL1, VDD-PLL2, VSS-PLL1, VSS-PLL2, VDD-CPG, VSS-CPG, VDD-RTC, and VSS-RTC) regardless of whether or not the PLL circuits, crystal oscillation circuit, and RTC are used.

* May be connected to V_{SSQ} .

Figure 1.3 Pin Arrangement (256-Pin BGA)



Note: Power must be supplied to the on-chip PLL power supply pins (VDD-PLL1, VDD-PLL2, VSS-PLL1, VSS-PLL2, VDD-CPG, VSS-CPG, VDD-RTC, and VSS-RTC) regardless of whether or not the PLL circuits, crystal oscillation circuit, and RTC are used.

Figure 1.4 Pin Arrangement (292-Pin BGA)

1.4 Pin Functions

1.4.1 Pin Functions (256-Pin QFP)

Table 1.2 Pin Functions

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|----------------------|-------|-------------------------|-------|------------------|------|-------|-------------------|------------------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 1 | TMS | I | Mode (H-UDI) | | | | | | |
| 2 | TCK | I | Clock (H-UDI) | | | | | | |
| 3 | VDDQ | Power | IO VDD | | | | | | |
| 4 | VSSQ | Power | IO GND | | | | | | |
| 5 | TDI | I | Data in (H-UDI) | | | | | | |
| 6 | $\overline{CS0}$ | O | Chip select 0 | | $\overline{CS0}$ | | | | $\overline{CS0}$ |
| 7 | $\overline{CS1}$ | O | Chip select 1 | | $\overline{CS1}$ | | | | $\overline{CS1}$ |
| 8 | $\overline{CS4}$ | O | Chip select 4 | | $\overline{CS4}$ | | | | $\overline{CS4}$ |
| 9 | $\overline{CS5}$ | O | Chip select 5 | | $\overline{CS5}$ | | | $\overline{CE1A}$ | $\overline{CS5}$ |
| 10 | $\overline{CS6}$ | O | Chip select 6 | | $\overline{CS6}$ | | | $\overline{CE1B}$ | $\overline{CS6}$ |
| 11 | BS | O | Bus start | | (BS) | (BS) | (BS) | (BS) | (BS) |
| 12 | $\overline{WE0/REG}$ | O | D7–D0 select signal | | $\overline{WE0}$ | | | \overline{REG} | |
| 13 | $\overline{WE1}$ | O | D15–D8 select signal | | $\overline{WE1}$ | | | $\overline{WE1}$ | |
| 14 | D0 | I/O | Data | | | | | | A0 |
| 15 | VDDQ | Power | IO VDD | | | | | | |
| 16 | VSSQ | Power | IO GND | | | | | | |
| 17 | VDD | Power | Internal VDD | | | | | | |
| 18 | VSS | Power | Internal GND | | | | | | |
| 19 | D1 | I/O | Data | | | | | | A1 |
| 20 | D2 | I/O | Data | | | | | | A2 |
| 21 | D3 | I/O | Data | | | | | | A3 |
| 22 | D4 | I/O | Data | | | | | | A4 |
| 23 | D5 | I/O | Data | | | | | | A5 |
| 24 | D6 | I/O | Data | | | | | | A6 |
| 25 | D7 | I/O | Data | | | | | | A7 |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|-------------------------|-------|-------------------------|-------|---------------------|---------------------|---------------------|---------------------|---------------------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 26 | D8 | I/O | Data | | | | | | A8 |
| 27 | D9 | I/O | Data | | | | | | A9 |
| 28 | D10 | I/O | Data | | | | | | A10 |
| 29 | VDDQ | Power | IO VDD | | | | | | |
| 30 | VSSQ | Power | IO GND | | | | | | |
| 31 | D11 | I/O | Data | | | | | | A11 |
| 32 | D12 | I/O | Data | | | | | | A12 |
| 33 | D13 | I/O | Data | | | | | | A13 |
| 34 | D14 | I/O | Data | | | | | | A14 |
| 35 | D15 | I/O | Data | | | | | | A15 |
| 36 | CAS $\bar{0}$ / DQM0 | O | D7–D0 select signal | | | CAS $\bar{0}$ | DQM0 | | |
| 37 | CAS $\bar{1}$ / DQM1 | O | D15–D8 select signal | | | CAS $\bar{1}$ | DQM1 | | |
| 38 | RD/ \overline{WR} | O | Read/write | | RD/ \overline{WR} | RD/ \overline{WR} | RD/ \overline{WR} | RD/ \overline{WR} | RD/ \overline{WR} |
| 39 | CKIO | O | Clock output | | | CKIO | CKIO | CKIO | CKIO |
| 40 | Reserved | | Do not connect | | | | | | |
| 41 | VDDQ | Power | IO VDD | | | | | | |
| 42 | VSSQ | Power | IO GND | | | | | | |
| 43 | Reserved | | Do not connect | | | | | | |
| 44 | RD/CASS/ FRAME | O | Read/CAS/ FRAME | | OE | | CAS | OE | FRAME |
| 45 | CKE | O | Clock output enable | | | | CKE | | |
| 46 | RAS | O | RAS | | | RAS | RAS | | |
| 47 | VDD | Power | Internal VDD | | | | | | |
| 48 | VSS | Power | Internal GND | | | | | | |
| 49 | CS $\bar{2}$ | O | Chip select 2 | | CS $\bar{2}$ | (CS2) | CS $\bar{2}$ | | CS $\bar{2}$ |
| 50 | CS $\bar{3}$ | O | Chip select 3 | | CS $\bar{3}$ | (CS3) | CS $\bar{3}$ | | CS $\bar{3}$ |
| 51 | A0 | O | Address | | | | | | |
| 52 | A1 | O | Address | | | | | | |
| 53 | A2 | O | Address | | | | | | |
| 54 | A3 | O | Address | | | | | | |
| 55 | VDDQ | Power | IO VDD | | | | | | |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------------------------------|-------|--------------------------|-------|------------------|--------------------------|-------|--------|-----|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 56 | VSSQ | Power | IO GND | | | | | | |
| 57 | A4 | O | Address | | | | | | |
| 58 | A5 | O | Address | | | | | | |
| 59 | A6 | O | Address | | | | | | |
| 60 | A7 | O | Address | | | | | | |
| 61 | A8 | O | Address | | | | | | |
| 62 | A9 | O | Address | | | | | | |
| 63 | A10 | O | Address | | | | | | |
| 64 | A11 | O | Address | | | | | | |
| 65 | A12 | O | Address | | | | | | |
| 66 | A13 | O | Address | | | | | | |
| 67 | VDDQ | Power | IO VDD | | | | | | |
| 68 | VSSQ | Power | IO GND | | | | | | |
| 69 | A14 | O | Address | | | | | | |
| 70 | A15 | O | Address | | | | | | |
| 71 | A16 | O | Address | | | | | | |
| 72 | A17 | O | Address | | | | | | |
| 73 | $\overline{\text{CAS2}}$ / DQM2 | O | D23–D16 select signal | | | $\overline{\text{CAS2}}$ | DQM2 | | |
| 74 | $\overline{\text{CAS3}}$ / DQM3 | O | D31–D24 select signal | | | $\overline{\text{CAS3}}$ | DQM3 | | |
| 75 | D16 | I/O | Data | | | | | | A16 |
| 76 | D17 | I/O | Data | | | | | | A17 |
| 77 | D18 | I/O | Data | | | | | | A18 |
| 78 | D19 | I/O | Data | | | | | | A19 |
| 79 | VDDQ | Power | IO VDD | | | | | | |
| 80 | VSSQ | Power | IO GND | | | | | | |
| 81 | VDD | Power | Internal VDD | | | | | | |
| 82 | VSS | Power | Internal GND | | | | | | |
| 83 | D20 | I/O | Data | | | | | | A20 |
| 84 | D21 | I/O | Data | | | | | | A21 |
| 85 | D22 | I/O | Data | | | | | | A22 |
| 86 | D23 | I/O | Data | | | | | | A23 |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|--|-------|------------------------------|-------|------------------|------|-------|--------|---------------------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 87 | D24 | I/O | Data | | | | | | A24 |
| 88 | D25 | I/O | Data | | | | | | A25 |
| 89 | D26 | I/O | Data | | | | | | |
| 90 | D27 | I/O | Data | | | | | | |
| 91 | D28 | I/O | Data | | | | | | |
| 92 | D29 | I/O | Data | | | | | | ACCSIZE0 |
| 93 | VDDQ | Power | IO VDD | | | | | | |
| 94 | VSSQ | Power | IO GND | | | | | | |
| 95 | D30 | I/O | Data | | | | | | ACCSIZE1 |
| 96 | D31 | I/O | Data | | | | | | ACCSIZE2 |
| 97 | VDD | Power | Internal VDD | | | | | | |
| 98 | VSS | Power | Internal GND | | | | | | |
| 99 | A18 | O | Address | | | | | | |
| 100 | A19 | O | Address | | | | | | |
| 101 | A20 | O | Address | | | | | | |
| 102 | A21 | O | Address | | | | | | |
| 103 | A22 | O | Address | | | | | | |
| 104 | A23 | O | Address | | | | | | |
| 105 | VDDQ | Power | IO VDD | | | | | | |
| 106 | VSSQ | Power | IO GND | | | | | | |
| 107 | A24 | O | Address | | | | | | |
| 108 | A25 | O | Address | | | | | | |
| 109 | $\overline{WE2}/$ \overline{ICIORD} | O | D23–D16 select signal | | WE2 | | | | \overline{ICIORD} |
| 110 | $\overline{WE3}/$ \overline{ICIOWR} | O | D31–D24 select signal | | WE3 | | | | \overline{ICIOWR} |
| 111 | VDD | Power | Internal VDD | | | | | | |
| 112 | VSS | Power | Internal GND | | | | | | |
| 113 | \overline{SLEEP} | I | Sleep | | | | | | |
| 114 | $\overline{PCIGNT4}$ | O | Bus grant (host function) | | | | | | |
| 115 | $\overline{PCIGNT3}$ | O | Bus grant (host function) | | | | | | |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|--------------------|-------|---|-------|------------------|--------|--------|--------|--------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 116 | PCIGNT2 | O | Bus grant (host function) | | | | | | |
| 117 | PCIREQ4 | I* | Bus request (host function) | | | | | | |
| 118 | PCIREQ3/ MD10 | I* | Bus request (host function)/ mode | MD10 | | | | | |
| 119 | VDDQ | Power | IO VDD | | | | | | |
| 120 | VSSQ | Power | IO GND | | | | | | |
| 121 | PCIREQ2/ MD9 | I* | Bus request (host function)/ mode | MD9 | | | | | |
| 122 | IDSEL | I | Configuration device select | | | | | | |
| 123 | INTA | O | Interrupt (async) | | | | | | |
| 124 | PCIRST | O | Reset output | | | | | | |
| 125 | PCICLK | I | PCI input clock | | | | | | |
| 126 | PCIGNT1/ REQOUT | O | Bus grant (host function)/ bus request | | | | | | |
| 127 | PCIREQ1/ GNTIN | I | Bus request (host function) /bus grant | | | | | | |
| 128 | SERR | I/O | System error | | | | | | |
| 129 | AD31 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 130 | AD30 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 131 | VDDQ | Power | IO VDD | | | | | | |
| 132 | VSSQ | Power | IO GND | | | | | | |
| 133 | AD29 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|----------------|-------|---------------------------|-------|------------------|--------|--------|--------|--------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 134 | AD28 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 135 | AD27 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 136 | AD26 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 137 | AD25 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 138 | AD24 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 139 | C/BE $\bar{3}$ | I/O | Command/byt e enable | | | | | | |
| 140 | AD23 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 141 | AD22 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 142 | AD21 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 143 | VDDQ | Power | IO VDD | | | | | | |
| 144 | VSSQ | Power | IO GND | | | | | | |
| 145 | VDD | Power | Internal VDD | | | | | | |
| 146 | VSS | Power | Internal GND | | | | | | |
| 147 | AD20 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 148 | AD19 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 149 | AD18 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 150 | AD17 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 151 | AD16 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 152 | C/BE $\bar{2}$ | I/O | Command/ byte enable | | | | | | |
| 153 | PCIFRAME | I/O | Bus cycle | | | | | | |
| 154 | IRDY | I/O | Initiator ready | | | | | | |
| 155 | TRDY | I/O | Target ready | | | | | | |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|----------|-------|-----------------------|-------|------------------|--------|--------|--------|--------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 156 | DEVSEL | I/O | Device select | | | | | | |
| 157 | VDDQ | Power | IO VDD | | | | | | |
| 158 | VSSQ | Power | IO GND | | | | | | |
| 159 | PCISTOP | I/O | Transaction stop | | | | | | |
| 160 | PCILOCK | I/O | Exclusive access | | | | | | |
| 161 | PERR | I/O | Parity error | | | | | | |
| 162 | PAR | I/O | Parity | | | | | | |
| 163 | C/BE1 | I/O | Command/byte enable | | | | | | |
| 164 | AD15 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 165 | AD14 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 166 | AD13 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 167 | AD12 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 168 | AD11 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 169 | VDDQ | Power | IO VDD | | | | | | |
| 170 | VSSQ | Power | IO GND | | | | | | |
| 171 | AD10 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 172 | AD9 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 173 | AD8 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 174 | C/BE0 | I/O | Command/byte enable | | | | | | |
| 175 | VDD | Power | Internal VDD | | | | | | |
| 176 | VSS | Power | Internal GND | | | | | | |
| 177 | AD7 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 178 | AD6 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|----------------------------|-------|------------------------------|-------|------------------|--------|---------------------------|--------|--------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 179 | AD5 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 180 | AD4 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 181 | AD3 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 182 | AD2 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 183 | VDDQ | Power | I/O VDD | | | | | | |
| 184 | VSSQ | Power | I/O GND | | | | | | |
| 185 | AD1 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 186 | AD0 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 187 | $\overline{\text{IRL0}}$ | I | Interrupt 0 | | | | | | |
| 188 | $\overline{\text{IRL1}}$ | I | Interrupt 1 | | | | | | |
| 189 | $\overline{\text{IRL2}}$ | I | Interrupt 2 | | | | | | |
| 190 | $\overline{\text{IRL3}}$ | I | Interrupt 3 | | | | | | |
| 191 | VSSQ | Power | I/O GND | | | | | | |
| 192 | VDDQ | Power | I/O VDD | | | | | | |
| 193 | XTAL2 | O | RTC crystal resonator pin | | | | | | |
| 194 | EXTAL2 | I | RTC crystal resonator pin | | | | | | |
| 195 | VDD-RTC | Power | RTC VDD | | | | | | |
| 196 | VSS-RTC | Power | RTC GND | | | | | | |
| 197 | CA*2 | I | Hardware standby | | | | | | |
| 198 | $\overline{\text{RESET}}$ | I | Reset | | | | $\overline{\text{RESET}}$ | | |
| 199 | $\overline{\text{TRST}}$ | I | Reset (H-UDI) | | | | | | |
| 200 | $\overline{\text{MRESET}}$ | I | Manual reset | | | | | | |
| 201 | NMI | I | Nonmaskable interrupt | | | | | | |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|---|-------|--|-------|--------------------------|--------------------------|--------------------------|----------------------------|--------------------------|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 202 | $\overline{\text{BACK}}$ / $\overline{\text{BSREQ}}$ | O | Bus acknowledge/ bus request | | | | | | |
| 203 | $\overline{\text{BREQ}}$ / $\overline{\text{BSACK}}$ | I | Bus request/bus acknowledge | | | | | | |
| 204 | MD6/ $\overline{\text{IOIS16}}$ | I | Mode/ $\overline{\text{IOIS16}}$ (PCMCIA) | MD6 | | | | $\overline{\text{IOIS16}}$ | |
| 205 | $\overline{\text{RDY}}$ | I | Bus ready | | $\overline{\text{RDY}}$ | | | $\overline{\text{RDY}}$ | $\overline{\text{RDY}}$ |
| 206 | TXD | O | SCI data output | | | | | | |
| 207 | VDDQ | Power | IO VDD | | | | | | |
| 208 | VSSQ | Power | IO GND | | | | | | |
| 209 | VDD | Power | Internal VDD | | | | | | |
| 210 | VSS | Power | Internal GND | | | | | | |
| 211 | MD2/RXD2 | I | Mode/SCIF data input | MD2 | RXD2 | RXD2 | RXD2 | RXD2 | RXD2 |
| 212 | RXD | I | SCI data input | | | | | | |
| 213 | TCLK | I/O | RTC/TMU clock | | | | | | |
| 214 | MD8/RTS2 | I/O | Mode/SCIF data control (RTS) | MD8 | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ |
| 215 | SCK | I/O | SCIF clock | | | | | | |
| 216 | MD1/TXD2 | I/O | Mode/SCIF data output | MD1 | TXD2 | TXD2 | TXD2 | TXD2 | TXD2 |
| 217 | MD0/SCK2 | I/O | Mode/SCIF clock | MD0 | SCK2 | SCK2 | SCK2 | SCK2 | SCK2 |
| 218 | MD7/ $\overline{\text{CTS2}}$ | I/O | Mode/SCIF data control (CTS) | MD7 | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ |
| 219 | AUDSYNC | | AUD sync | | | | | | |
| 220 | AUDCK | | AUD clock | | | | | | |
| 221 | VDDQ | Power | IO VDD | | | | | | |
| 222 | VSSQ | Power | IO GND | | | | | | |
| 223 | AUDATA0 | | AUD data | | | | | | |
| 224 | AUDATA1 | | AUD data | | | | | | |

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|---------------------------------------|-------|--------------------------------------|-------|------------------|------|-------|--------------------------|-----|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 225 | VDD | Power | Internal VDD | | | | | | |
| 226 | VSS | Power | Internal GND | | | | | | |
| 227 | AUDATA2 | | AUD data | | | | | | |
| 228 | AUDATA3 | | AUD data | | | | | | |
| 229 | Reserved | | Do not connect | | | | | | |
| 230 | MD3/ $\overline{\text{CE2A}}$ | I/O | Mode/ PCMCIA-CE | MD3 | | | | $\overline{\text{CE2A}}$ | |
| 231 | MD4/ $\overline{\text{CE2B}}$ | I/O | Mode/ PCMCIA-CE | MD4 | | | | $\overline{\text{CE2B}}$ | |
| 232 | MD5 | I | Mode | MD5 | | | | | |
| 233 | VDDQ | Power | IO VDD | | | | | | |
| 234 | VSSQ | Power | IO GND | | | | | | |
| 235 | DACK0 | O | DMAC0 bus acknowledge | | | | | | |
| 236 | DACK1 | O | DMAC1 bus acknowledge | | | | | | |
| 237 | DRAK0 | O | DMAC0 request acknowledge | | | | | | |
| 238 | DRAK1 | O | DMAC1 request acknowledge | | | | | | |
| 239 | VDD | Power | Internal VDD | | | | | | |
| 240 | VSS | Power | Internal GND | | | | | | |
| 241 | STATUS0 | O | Status | | | | | | |
| 242 | STATUS1 | O | Status | | | | | | |
| 243 | $\overline{\text{DREQ0}}$ | I | Request from DMAC0 | | | | | | |
| 244 | $\overline{\text{DREQ1}}$ | I | Request from DMAC1 | | | | | | |
| 245 | $\overline{\text{ASEBRK}}/$ BRKACK | I/O | Pin break/ acknowledge (H-UDI) | | | | | | |
| 246 | TDO | O | Data out (H-UDI) | | | | | | |

Memory Interface

| No. | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|----------|-------|--|-------|------------------|------|-------|--------|-----|
| | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 247 | VDDQ | Power | IO VDD | | | | | | |
| 248 | VSSQ | Power | IO GND | | | | | | |
| 249 | VDD-PLL2 | Power | PLL2 VDD | | | | | | |
| 250 | VSS-PLL2 | Power | PLL2 GND | | | | | | |
| 251 | VDD-PLL1 | Power | PLL1 VDD | | | | | | |
| 252 | VSS-PLL1 | Power | PLL1 GND | | | | | | |
| 253 | VDD-CPG | Power | CPG VDD | | | | | | |
| 254 | VSS-CPG | Power | CPG GND | | | | | | |
| 255 | XTAL | O | Crystal resonator | | | | | | |
| 256 | EXTAL | I | External input clock/crystal resonator | | | | | | |

Legend:

I: Input

O: Output

I/O: Input/output

Power: Power supply

Notes: Supply power to all power pins. However, on the SH7751 in hardware standby mode, supply power to RTC at the minimum.

Power must be supplied to VDD-PLL1/2 and VSS-PLL1/2 regardless of whether or not the on-chip PLL circuits are used.

Power must be supplied to VDD-CPG and VSS-CPG regardless of whether or not the on-chip crystal oscillation circuit is used.

Power must be supplied to VDD-RTC and VSS-RTC regardless of whether or not the on-chip RTC is used.

For the handling of the PCI bus pins in PCI-disabled mode, see table D.4 in appendix D.

* I/O attribute is I/O when used as a port.

1.4.2 Pin Functions (256-Pin BGA)

Table 1.3 Pin Functions

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---------------------------------------|-------|----------------------|-------|------------------|-----------------|-----------------|-------------------|------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 1 | B3 | TMS | I | Mode (H-UDI) | | | | | | |
| 2 | C4 | TCK | I | Clock (H-UDI) | | | | | | |
| 3 | G3 | VDDQ | Power | IO VDD | | | | | | |
| 4 | F2 | VSSQ | Power | IO GND | | | | | | |
| 5 | D4 | TDI | I | Data in (H-UDI) | | | | | | |
| 6 | B1 | $\overline{CS0}$ | O | Chip select 0 | | $\overline{CS0}$ | | | | $\overline{CS0}$ |
| 7 | C2 | $\overline{CS1}$ | O | Chip select 1 | | $\overline{CS1}$ | | | | $\overline{CS1}$ |
| 8 | C1 | $\overline{CS4}$ | O | Chip select 4 | | $\overline{CS4}$ | | | | $\overline{CS4}$ |
| 9 | D3 | $\overline{CS5}$ | O | Chip select 5 | | $\overline{CS5}$ | | | $\overline{CE1A}$ | $\overline{CS5}$ |
| 10 | D2 | $\overline{CS6}$ | O | Chip select 6 | | $\overline{CS6}$ | | | $\overline{CE1B}$ | $\overline{CS6}$ |
| 11 | D1 | \overline{BS} | O | Bus start | | \overline{BS} | \overline{BS} | \overline{BS} | \overline{BS} | \overline{BS} |
| 12 | E4 | $\overline{WE0}/$ \overline{REG} | O | D7–D0 select signal | | $\overline{WE0}$ | | | \overline{REG} | |
| 13 | E3 | $\overline{WE1}$ | O | D15–D8 select signal | | $\overline{WE1}$ | | | $\overline{WE1}$ | |
| 14 | E2 | D0 | I/O | Data | | | | | | A0 |
| 15 | G2 | VDDQ | Power | IO VDD | | | | | | |
| 16 | L4 | VSSQ | Power | IO GND | | | | | | |
| 17 | G4 | VDD | Power | Internal VDD | | | | | | |
| 18 | F4 | VSS | Power | Internal GND | | | | | | |
| 19 | E1 | D1 | I/O | Data | | | | | | A1 |
| 20 | F3 | D2 | I/O | Data | | | | | | A2 |
| 21 | F1 | D3 | I/O | Data | | | | | | A3 |
| 22 | G1 | D4 | I/O | Data | | | | | | A4 |
| 23 | H4 | D5 | I/O | Data | | | | | | A5 |
| 24 | H3 | D6 | I/O | Data | | | | | | A6 |
| 25 | H2 | D7 | I/O | Data | | | | | | A7 |
| 26 | H1 | D8 | I/O | Data | | | | | | A8 |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---|-------|--|-------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 27 | J4 | D9 | I/O | Data | | | | | | A9 |
| 28 | J3 | D10 | I/O | Data | | | | | | A10 |
| 29 | K3 | VDDQ | Power | IO VDD | | | | | | |
| 30 | L3 | VSSQ | Power | IO GND | | | | | | |
| 31 | J2 | D11 | I/O | Data | | | | | | A11 |
| 32 | J1 | D12 | I/O | Data | | | | | | A12 |
| 33 | K4 | D13 | I/O | Data | | | | | | A13 |
| 34 | K2 | D14 | I/O | Data | | | | | | A14 |
| 35 | K1 | D15 | I/O | Data | | | | | | A15 |
| 36 | L2 | $\overline{\text{CAS0}}$ / DQM0 | O | D7–D0 select signal | | | $\overline{\text{CAS0}}$ | DQM0 | | |
| 37 | M4 | $\overline{\text{CAS1}}$ / DQM1 | O | D15–D8 select signal | | | $\overline{\text{CAS1}}$ | DQM1 | | |
| 38 | M3 | $\text{RD}/\overline{\text{WR}}$ | O | Read/write | | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ |
| 39 | M1 | CKIO | O | Clock output | | | CKIO | CKIO | CKIO | CKIO |
| 40 | M2 | NC | | Do not connect | | | | | | |
| 41 | P3 | VDDQ | Power | IO VDD | | | | | | |
| 42 | L1 | VSSQ | Power | IO GND | | | | | | |
| 43 | N3 | NC | | Do not connect | | | | | | |
| 44 | P1 | $\overline{\text{RD}}$ / $\overline{\text{CASS}}$ / $\overline{\text{FRAME}}$ | O | Read/ $\overline{\text{CAS}}$ / $\overline{\text{FRAME}}$ | | $\overline{\text{OE}}$ | | $\overline{\text{CAS}}$ | $\overline{\text{OE}}$ | $\overline{\text{FRAME}}$ |
| 45 | N2 | CKE | O | Clock output enable | | | | CKE | | |
| 46 | N1 | $\overline{\text{RAS}}$ | O | $\overline{\text{RAS}}$ | | | $\overline{\text{RAS}}$ | $\overline{\text{RAS}}$ | | |
| 47 | P4 | VDD | Power | Internal VDD | | | | | | |
| 48 | R4 | VSS | Power | Internal GND | | | | | | |
| 49 | N4 | $\overline{\text{CS2}}$ | O | Chip select 2 | | $\overline{\text{CS2}}$ | ($\overline{\text{CS2}}$) | $\overline{\text{CS2}}$ | | $\overline{\text{CS2}}$ |
| 50 | R3 | $\overline{\text{CS3}}$ | O | Chip select 3 | | $\overline{\text{CS3}}$ | ($\overline{\text{CS3}}$) | $\overline{\text{CS3}}$ | | $\overline{\text{CS3}}$ |
| 51 | R1 | A0 | O | Address | | | | | | |
| 52 | T4 | A1 | O | Address | | | | | | |
| 53 | T3 | A2 | O | Address | | | | | | |
| 54 | T2 | A3 | O | Address | | | | | | |
| 55 | P2 | VDDQ | Power | IO VDD | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---------------|-------|--------------------------|-------|------------------|------|-------|--------|-----|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 56 | R2 | VSSQ | Power | IO GND | | | | | | |
| 57 | T1 | A4 | O | Address | | | | | | |
| 58 | U4 | A5 | O | Address | | | | | | |
| 59 | U3 | A6 | O | Address | | | | | | |
| 60 | U2 | A7 | O | Address | | | | | | |
| 61 | U1 | A8 | O | Address | | | | | | |
| 62 | V2 | A9 | O | Address | | | | | | |
| 63 | V1 | A10 | O | Address | | | | | | |
| 64 | W1 | A11 | O | Address | | | | | | |
| 65 | Y1 | A12 | O | Address | | | | | | |
| 66 | Y2 | A13 | O | Address | | | | | | |
| 67 | V7 | VDDQ | Power | IO VDD | | | | | | |
| 68 | V3 | VSSQ | Power | IO GND | | | | | | |
| 69 | W3 | A14 | O | Address | | | | | | |
| 70 | Y3 | A15 | O | Address | | | | | | |
| 71 | V4 | A16 | O | Address | | | | | | |
| 72 | W4 | A17 | O | Address | | | | | | |
| 73 | Y4 | CAS2/ DQM2 | O | D23–D16 select signal | | | CAS2 | DQM2 | | |
| 74 | U5 | CAS3/ DQM3 | O | D31–D24 select signal | | | CAS3 | DQM3 | | |
| 75 | V5 | D16 | I/O | Data | | | | | | A16 |
| 76 | W5 | D17 | I/O | Data | | | | | | A17 |
| 77 | Y5 | D18 | I/O | Data | | | | | | A18 |
| 78 | V6 | D19 | I/O | Data | | | | | | A19 |
| 79 | W7 | VDDQ | Power | IO VDD | | | | | | |
| 80 | W2 | VSSQ | Power | IO GND | | | | | | |
| 81 | U7 | VDD | Power | Internal VDD | | | | | | |
| 82 | U6 | VSS | Power | Internal GND | | | | | | |
| 83 | Y6 | D20 | I/O | Data | | | | | | A20 |
| 84 | Y7 | D21 | I/O | Data | | | | | | A21 |
| 85 | U8 | D22 | I/O | Data | | | | | | A22 |
| 86 | V8 | D23 | I/O | Data | | | | | | A23 |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--|-------|------------------------------|-------|------------------|------|-------|--------|---------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 87 | W8 | D24 | I/O | Data | | | | | | A24 |
| 88 | Y8 | D25 | I/O | Data | | | | | | A25 |
| 89 | U9 | D26 | I/O | Data | | | | | | |
| 90 | V9 | D27 | I/O | Data | | | | | | |
| 91 | W9 | D28 | I/O | Data | | | | | | |
| 92 | Y9 | D29 | I/O | Data | | | | | | ACCSIZE0 |
| 93 | V10 | VDDQ | Power | IO VDD | | | | | | |
| 94 | W6 | VSSQ | Power | IO GND | | | | | | |
| 95 | W10 | D30 | I/O | Data | | | | | | ACCSIZE1 |
| 96 | Y10 | D31 | I/O | Data | | | | | | ACCSIZE2 |
| 97 | U10 | VDD | Power | Internal VDD | | | | | | |
| 98 | U11 | VSS | Power | Internal GND | | | | | | |
| 99 | V11 | A18 | O | Address | | | | | | |
| 100 | Y11 | A19 | O | Address | | | | | | |
| 101 | U12 | A20 | O | Address | | | | | | |
| 102 | V12 | A21 | O | Address | | | | | | |
| 103 | W12 | A22 | O | Address | | | | | | |
| 104 | Y12 | A23 | O | Address | | | | | | |
| 105 | V14 | VDDQ | Power | IO VDD | | | | | | |
| 106 | W11 | VSSQ | Power | IO GND | | | | | | |
| 107 | U13 | A24 | O | Address | | | | | | |
| 108 | V13 | A25 | O | Address | | | | | | |
| 109 | W13 | $\overline{WE2}/$ \overline{ICIORD} | O | D23–D16 select signal | | $\overline{WE2}$ | | | | \overline{ICIORD} |
| 110 | Y13 | $\overline{WE3}/$ \overline{ICIOWR} | O | D31–D24 select signal | | $\overline{WE3}$ | | | | \overline{ICIOWR} |
| 111 | U14 | VDD | Power | Internal VDD | | | | | | |
| 112 | U15 | VSS | Power | Internal GND | | | | | | |
| 113 | Y14 | \overline{SLEEP} | I | Sleep | | | | | | |
| 114 | V15 | $\overline{PCIGNT4}$ | O | Bus grant (host function) | | | | | | |
| 115 | Y15 | $\overline{PCIGNT3}$ | O | Bus grant (host function) | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|------------------------------------|-------|--|-------|------------------|--------|--------|--------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 116 | U16 | $\overline{\text{PCIGNT2}}$ | O | Bus grant (host function) | | | | | | |
| 117 | V16 | $\overline{\text{PCIREQ4}}$ | I* | Bus request (host function) | | | | | | |
| 118 | W16 | $\overline{\text{PCIREQ3/MD10}}$ | I* | Bus request (host function)/ mode | MD10 | | | | | |
| 119 | W14 | VDDQ | Power | IO VDD | | | | | | |
| 120 | W15 | VSSQ | Power | IO GND | | | | | | |
| 121 | Y16 | $\overline{\text{PCIREQ2/MD9}}$ | I* | Bus request (host function)/ mode | MD9 | | | | | |
| 122 | U17 | IDSEL | I | Configuration device select | | | | | | |
| 123 | V17 | $\overline{\text{INTA}}$ | O | Interrupt (async) | | | | | | |
| 124 | W17 | $\overline{\text{PCIRST}}$ | O | Reset output | | | | | | |
| 125 | Y17 | PCICLK | I | PCI input clock | | | | | | |
| 126 | W18 | $\overline{\text{PCIGNT1/REQOUT}}$ | O | Bus grant (host function)/ bus request | | | | | | |
| 127 | Y18 | $\overline{\text{PCIREQ1/GNTIN}}$ | I | Bus request (host function)/ bus grant | | | | | | |
| 128 | Y19 | $\overline{\text{SERR}}$ | I/O | System error | | | | | | |
| 129 | Y20 | AD31 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 130 | W20 | AD30 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 131 | P18 | VDDQ | Power | IO VDD | | | | | | |
| 132 | V18 | VSSQ | Power | IO GND | | | | | | |
| 133 | V19 | AD29 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 134 | V20 | AD28 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 135 | U18 | AD27 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|----------|-------|---------------------------|-------|------------------|--------|--------|--------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 136 | U20 | AD26 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 137 | T17 | AD25 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 138 | T18 | AD24 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 139 | U19 | C/BE3 | I/O | PCI address/ data/port | | | | | | |
| 140 | T20 | AD23 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 141 | R18 | AD22 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 142 | T19 | AD21 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 143 | N19 | VDDQ | Power | IO VDD | | | | | | |
| 144 | W19 | VSSQ | Power | IO GND | | | | | | |
| 145 | P17 | VDD | Power | Internal VDD | | | | | | |
| 146 | R17 | VSS | Power | Internal GND | | | | | | |
| 147 | R20 | AD20 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 148 | P20 | AD19 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 149 | P19 | AD18 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 150 | N20 | AD17 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 151 | N17 | AD16 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 152 | N18 | C/BE2 | I/O | Command/ byte enable | | | | | | |
| 153 | M20 | PCIFRAME | I/O | Bus cycle | | | | | | |
| 154 | M19 | IRDY | I/O | Initiator ready | | | | | | |
| 155 | M18 | TRDY | I/O | Target ready | | | | | | |
| 156 | M17 | DEVSEL | I/O | Device select | | | | | | |
| 157 | L18 | VDDQ | Power | IO VDD | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|----------|-------|-----------------------|-------|------------------|--------|--------|--------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 158 | R19 | VSSQ | Power | IO GND | | | | | | |
| 159 | L20 | PCISTOP | I/O | Transaction stop | | | | | | |
| 160 | L19 | PCILOCK | I/O | Exclusive access | | | | | | |
| 161 | L17 | PERR | I/O | Parity error | | | | | | |
| 162 | K20 | PAR | I/O | Parity | | | | | | |
| 163 | K18 | C/BE1 | I/O | Command/byte enable | | | | | | |
| 164 | J20 | AD15 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 165 | J19 | AD14 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 166 | J18 | AD13 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 167 | J17 | AD12 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 168 | H20 | AD11 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 169 | G18 | VDDQ | Power | IO VDD | | | | | | |
| 170 | K17 | VSSQ | Power | IO GND | | | | | | |
| 171 | H19 | AD10 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 172 | G20 | AD9 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 173 | H18 | AD8 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 174 | H17 | C/BE0 | I/O | Command/byte enable | | | | | | |
| 175 | G17 | VDD | Power | Internal VDD | | | | | | |
| 176 | F17 | VSS | Power | Internal GND | | | | | | |
| 177 | F18 | AD7 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 178 | F20 | AD6 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 179 | E20 | AD5 | I/O | PCI address/data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--|-------|------------------------------------|-------|------------------|--------|--------|---------------------------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 180 | E19 | AD4 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 181 | E18 | AD3 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 182 | D20 | AD2 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 183 | G19 | VDDQ | Power | I/O VDD | | | | | | |
| 184 | K19 | VSSQ | Power | I/O GND | | | | | | |
| 185 | D19 | AD1 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 186 | D18 | AD0 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 187 | E17 | $\overline{\text{IRL0}}$ | I | Interrupt 0 | | | | | | |
| 188 | C20 | $\overline{\text{IRL1}}$ | I | Interrupt 1 | | | | | | |
| 189 | C19 | $\overline{\text{IRL2}}$ | I | Interrupt 2 | | | | | | |
| 190 | B20 | $\overline{\text{IRL3}}$ | I | Interrupt 3 | | | | | | |
| 191 | B18 | NC | | Do not connect *2 | | | | | | |
| 192 | D17 | VDDQ | Power | I/O VDD | | | | | | |
| 193 | A20 | XTAL2 | O | RTC crystal resonator pin | | | | | | |
| 194 | A19 | EXTAL2 | I | RTC crystal resonator pin | | | | | | |
| 195 | A18 | VDD-RTC | Power | RTC VDD | | | | | | |
| 196 | B19 | VSS-RTC | Power | RTC GND | | | | | | |
| 197 | B17 | CA | I | Hardware standby | | | | | | |
| 198 | A17 | $\overline{\text{RESET}}$ | I | Reset | | | | | $\overline{\text{RESET}}$ | |
| 199 | C16 | $\overline{\text{TRST}}$ | I | Reset (H-UDI) | | | | | | |
| 200 | B16 | $\overline{\text{MRESET}}$ | I | Manual reset | | | | | | |
| 201 | D16 | NMI | I | Nonmaskable interrupt | | | | | | |
| 202 | A16 | $\overline{\text{BACK}}/\overline{\text{BSREQ}}$ | O | Bus acknowledge/ bus request | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---|-------|---|-------|--------------------------|--------------------------|--------------------------|----------------------------|--------------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 203 | B15 | $\overline{\text{BREQ}}$ / $\overline{\text{BSACK}}$ | I | Bus request/bus acknowledge | | | | | | |
| 204 | C15 | MD6/ $\overline{\text{IOIS16}}$ | I | Mode/ $\overline{\text{IOIS16}}$ (PCMCIA) | MD6 | | | | $\overline{\text{IOIS16}}$ | |
| 205 | A15 | $\overline{\text{RDY}}$ | I | Bus ready | | $\overline{\text{RDY}}$ | | | $\overline{\text{RDY}}$ | $\overline{\text{RDY}}$ |
| 206 | A14 | TXD | O | SCI data output | | | | | | |
| 207 | B14 | VDDQ | Power | IO VDD | | | | | | |
| 208 | F19 | VSSQ | Power | IO GND | | | | | | |
| 209 | D14 | VDD | Power | Internal VDD | | | | | | |
| 210 | D15 | VSS | Power | Internal GND | | | | | | |
| 211 | D13 | MD2/ RXD2 | I | Mode/SCIF data input | MD2 | RXD2 | RXD2 | RXD2 | RXD2 | RXD2 |
| 212 | C13 | RXD | I | SCI data input | | | | | | |
| 213 | B13 | TCLK | I/O | RTC/TMU clock | | | | | | |
| 214 | A13 | MD8/ $\overline{\text{RTS2}}$ | I/O | Mode/SCIF data control (RTS) | MD8 | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ | $\overline{\text{RTS2}}$ |
| 215 | D12 | SCK | I/O | SCIF clock | | | | | | |
| 216 | B11 | MD1/ TXD2 | I/O | Mode/SCIF data output | MD1 | TXD2 | TXD2 | TXD2 | TXD2 | TXD2 |
| 217 | C12 | MD0/ SCK2 | I/O | Mode/SCIF clock | MD0 | SCK2 | SCK2 | SCK2 | SCK2 | SCK2 |
| 218 | A12 | MD7/ $\overline{\text{CTS2}}$ | I/O | Mode/SCIF data control (CTS) | MD7 | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ | $\overline{\text{CTS2}}$ |
| 219 | B12 | AUDSYNC | | AUD sync | | | | | | |
| 220 | A11 | AUDCK | | AUD clock | | | | | | |
| 221 | C14 | VDDQ | Power | IO VDD | | | | | | |
| 222 | C18 | VSSQ | Power | IO GND | | | | | | |
| 223 | C10 | AUDATA0 | | AUD data | | | | | | |
| 224 | A10 | AUDATA1 | | AUD data | | | | | | |
| 225 | D11 | VDD | Power | Internal VDD | | | | | | |
| 226 | D10 | VSS | Power | Internal GND | | | | | | |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--------------------------------|-------|--------------------------------------|-------|------------------|------|-------|-------------------|-----|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 227 | B9 | AUDATA2 | | AUD data | | | | | | |
| 228 | D9 | AUDATA3 | | AUD data | | | | | | |
| 229 | C9 | NC | | Do not connect | | | | | | |
| 230 | A9 | MD3/ $\overline{CE2A}$ | I/O | Mode/ PCMCIA-CE | MD3 | | | | $\overline{CE2A}$ | |
| 231 | D8 | MD4/ $\overline{CE2B}$ | I/O | Mode/ PCMCIA-CE | MD4 | | | | $\overline{CE2B}$ | |
| 232 | C8 | MD5 | I | Mode | MD5 | | | | | |
| 233 | C11 | VDDQ | Power | IO VDD | | | | | | |
| 234 | C17 | VSSQ | Power | IO GND | | | | | | |
| 235 | B8 | DACK0 | O | DMAC0 bus acknowledge | | | | | | |
| 236 | A8 | DACK1 | O | DMAC1 bus acknowledge | | | | | | |
| 237 | B7 | DRAK0 | O | DMAC0 request acknowledge | | | | | | |
| 238 | A7 | DRAK1 | O | DMAC1 request acknowledge | | | | | | |
| 239 | D7 | VDD | Power | Internal VDD | | | | | | |
| 240 | D6 | VSS | Power | Internal GND | | | | | | |
| 241 | C6 | STATUS0 | O | Status | | | | | | |
| 242 | B6 | STATUS1 | O | Status | | | | | | |
| 243 | A6 | $\overline{DREQ0}$ | I | Request from DMAC0 | | | | | | |
| 244 | C5 | $\overline{DREQ1}$ | I | Request from DMAC1 | | | | | | |
| 245 | D5 | $\overline{ASEBRK}/$ BRKACK | I/O | Pin break/ acknowledge (H-UDI) | | | | | | |
| 246 | B4 | TDO | O | Data out (H-UDI) | | | | | | |
| 247 | C7 | VDDQ | Power | IO VDD | | | | | | |
| 248 | B10 | VSSQ | Power | IO GND | | | | | | |
| 249 | A5 | VDD-PLL2 | Power | PLL2 VDD | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|----------|-------|--|-------|------------------|------|-------|--------|-----|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 250 | B5 | VSS-PLL2 | Power | PLL2 GND | | | | | | |
| 251 | A4 | VDD-PLL1 | Power | PLL1 VDD | | | | | | |
| 252 | C3 | VSS-PLL1 | Power | PLL1 GND | | | | | | |
| 253 | A3 | VDD-CPG | Power | CPG VDD | | | | | | |
| 254 | B2 | VSS-CPG | Power | CPG GND | | | | | | |
| 255 | A2 | XTAL | O | Crystal resonator | | | | | | |
| 256 | A1 | EXTAL | I | External input clock/crystal resonator | | | | | | |

Legend:

I: Input

O: Output

I/O: Input/output

Power: Power supply

Notes: Supply power to all power pins. However, on the SH7751 in hardware standby mode, supply power to RTC at the minimum.

Power must be supplied to VDD-PLL1/2 and VSS-PLL1/2 regardless of whether or not the on-chip PLL circuits are used.

Power must be supplied to VDD-CPG and VSS-CPG regardless of whether or not the on-chip crystal oscillation circuit is used.

Power must be supplied to VDD-RTC and VSS-RTC regardless of whether or not the on-chip RTC is used.

For the handling of the PCI bus pins in PCI-disabled mode, see table D.4 in appendix D.

1. I/O attribute is I/O when used as a port.
2. May be connected to V_{SSQ} .

1.4.3 Pin Functions (292-Pin BGA)

Table 1.4 Pin Functions

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---------------------------------------|-------|----------------------|-------|-------------------|-------------------|-------------------|-------------------|-------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 1 | B1 | TMS | I | Mode (H-UDI) | | | | | | |
| 2 | B2 | TCK | I | Clock (H-UDI) | | | | | | |
| 3 | F4 | VDDQ | Power | IO VDD | | | | | | |
| 4 | E4 | VSS | Power | GND | | | | | | |
| 5 | C1 | TDI | I | Data in (H-UDI) | | | | | | |
| 6 | C2 | $\overline{CS0}$ | O | Chip select 0 | | $\overline{CS0}$ | | | | $\overline{CS0}$ |
| 7 | D1 | $\overline{CS1}$ | O | Chip select 1 | | $\overline{CS1}$ | | | | $\overline{CS1}$ |
| 8 | D2 | $\overline{CS4}$ | O | Chip select 4 | | $\overline{CS4}$ | | | | $\overline{CS4}$ |
| 9 | D3 | $\overline{CS5}$ | O | Chip select 5 | | $\overline{CS5}$ | | | $\overline{CE1A}$ | $\overline{CS5}$ |
| 10 | E1 | $\overline{CS6}$ | O | Chip select 6 | | $\overline{CS6}$ | | | $\overline{CE1B}$ | $\overline{CS6}$ |
| 11 | E2 | \overline{BS} | O | Bus start | | (\overline{BS}) | (\overline{BS}) | (\overline{BS}) | (\overline{BS}) | (\overline{BS}) |
| 12 | E3 | $\overline{WE0}/$ \overline{REG} | O | D7–D0 select signal | | $\overline{WE0}$ | | | | \overline{REG} |
| 13 | F1 | $\overline{WE1}$ | O | D15–D8 select signal | | $\overline{WE1}$ | | | | $\overline{WE1}$ |
| 14 | F2 | D0 | I/O | Data | | | | | | A0 |
| 15 | G3 | VDDQ | Power | IO VDD | | | | | | |
| 16 | D4 | VSS | Power | GND | | | | | | |
| 17 | G4 | VDD | Power | Internal VDD | | | | | | |
| 18 | H4 | VSS | Power | GND | | | | | | |
| 19 | F3 | D1 | I/O | Data | | | | | | A1 |
| 20 | G1 | D2 | I/O | Data | | | | | | A2 |
| 21 | G2 | D3 | I/O | Data | | | | | | A3 |
| 22 | H1 | D4 | I/O | Data | | | | | | A4 |
| 23 | H2 | D5 | I/O | Data | | | | | | A5 |
| 24 | H3 | D6 | I/O | Data | | | | | | A6 |
| 25 | J1 | D7 | I/O | Data | | | | | | A7 |
| 26 | J2 | D8 | I/O | Data | | | | | | A8 |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--|-------|-------------------------|-------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 27 | J3 | D9 | I/O | Data | | | | | | A9 |
| 28 | K1 | D10 | I/O | Data | | | | | | A10 |
| 29 | J4 | VDDQ | Power | IO VDD | | | | | | |
| 30 | D5 | VSS | Power | GND | | | | | | |
| 31 | K2 | D11 | I/O | Data | | | | | | A11 |
| 32 | K3 | D12 | I/O | Data | | | | | | A12 |
| 33 | L1 | D13 | I/O | Data | | | | | | A13 |
| 34 | L2 | D14 | I/O | Data | | | | | | A14 |
| 35 | L3 | D15 | I/O | Data | | | | | | A15 |
| 36 | M1 | $\overline{\text{CAS0}}$ / DQM0 | O | D7–D0 select signal | | | $\overline{\text{CAS0}}$ | DQM0 | | |
| 37 | M2 | $\overline{\text{CAS1}}$ / DQM1 | O | D15–D8 select signal | | | $\overline{\text{CAS1}}$ | DQM1 | | |
| 38 | M3 | $\text{RD}/\overline{\text{WR}}$ | O | Read/write | | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ | $\text{RD}/\overline{\text{WR}}$ |
| 39 | N1 | CKIO | O | Clock output | | CKIO | CKIO | CKIO | CKIO | CKIO |
| 40 | K4 | VDD | Power | Internal VDD | | | | | | |
| 41 | R4 | VDDQ | Power | IO VDD | | | | | | |
| 42 | L4 | VSS | Power | IO GND | | | | | | |
| 43 | M4 | VDDQ | Power | I/O VDD | | | | | | |
| 44 | N2 | $\overline{\text{RD}}/\text{CASS}/$ FRAME | O | Read/CAS/ FRAME | | OE | | $\overline{\text{CAS}}$ | OE | FRAME |
| 45 | N3 | CKE | O | Clock output enable | | | | CKE | | |
| 46 | P1 | $\overline{\text{RAS}}$ | O | $\overline{\text{RAS}}$ | | | $\overline{\text{RAS}}$ | $\overline{\text{RAS}}$ | | |
| 47 | P4 | VDD | Power | Internal VDD | | | | | | |
| 48 | N4 | VSS | Power | GND | | | | | | |
| 49 | P2 | $\overline{\text{CS2}}$ | O | Chip select 2 | | $\overline{\text{CS2}}$ | (CS2) | $\overline{\text{CS2}}$ | | $\overline{\text{CS2}}$ |
| 50 | R1 | $\overline{\text{CS3}}$ | O | Chip select 3 | | $\overline{\text{CS3}}$ | (CS3) | $\overline{\text{CS3}}$ | | $\overline{\text{CS3}}$ |
| 51 | R2 | A0 | O | Address | | | | | | |
| 52 | R3 | A1 | O | Address | | | | | | |
| 53 | T1 | A2 | O | Address | | | | | | |
| 54 | T2 | A3 | O | Address | | | | | | |
| 55 | P3 | VDDQ | Power | IO VDD | | | | | | |
| 56 | T4 | VSS | Power | GND | | | | | | |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---------------|-------|--------------------------|-------|------------------|------|-------|--------|-----|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 57 | T3 | A4 | O | Address | | | | | | |
| 58 | U1 | A5 | O | Address | | | | | | |
| 59 | U2 | A6 | O | Address | | | | | | |
| 60 | U3 | A7 | O | Address | | | | | | |
| 61 | V1 | A8 | O | Address | | | | | | |
| 62 | V2 | A9 | O | Address | | | | | | |
| 63 | V3 | A10 | O | Address | | | | | | |
| 64 | W1 | A11 | O | Address | | | | | | |
| 65 | W2 | A12 | O | Address | | | | | | |
| 66 | Y1 | A13 | O | Address | | | | | | |
| 67 | V7 | VDDQ | Power | IO VDD | | | | | | |
| 68 | U4 | VSS | Power | GND | | | | | | |
| 69 | Y2 | A14 | O | Address | | | | | | |
| 70 | Y3 | A15 | O | Address | | | | | | |
| 71 | W3 | A16 | O | Address | | | | | | |
| 72 | Y4 | A17 | O | Address | | | | | | |
| 73 | W4 | CAS2/ DQM2 | O | D23–D16 select signal | | | CAS2 | DQM2 | | |
| 74 | V4 | CAS3/ DQM3 | O | D31–D24 select signal | | | CAS3 | DQM3 | | |
| 75 | Y5 | D16 | I/O | Data | | | | | | A16 |
| 76 | W5 | D17 | I/O | Data | | | | | | A17 |
| 77 | V5 | D18 | I/O | Data | | | | | | A18 |
| 78 | Y6 | D19 | I/O | Data | | | | | | A19 |
| 79 | U6 | VDDQ | Power | IO VDD | | | | | | |
| 80 | U5 | VSS | Power | GND | | | | | | |
| 81 | U7 | VDD | Power | Internal VDD | | | | | | |
| 82 | U8 | VSS | Power | GND | | | | | | |
| 83 | W6 | D20 | I/O | Data | | | | | | A20 |
| 84 | V6 | D21 | I/O | Data | | | | | | A21 |
| 85 | Y7 | D22 | I/O | Data | | | | | | A22 |
| 86 | W7 | D23 | I/O | Data | | | | | | A23 |
| 87 | Y8 | D24 | I/O | Data | | | | | | A24 |
| 88 | W8 | D25 | I/O | Data | | | | | | A25 |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--|-------|------------------------------|-------|------------------|------|-------|--------|---------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 89 | V8 | D26 | I/O | Data | | | | | | |
| 90 | Y9 | D27 | I/O | Data | | | | | | |
| 91 | W9 | D28 | I/O | Data | | | | | | |
| 92 | V9 | D29 | I/O | Data | | | | | | ACCSIZE0 |
| 93 | U9 | VDDQ | Power | IO VDD | | | | | | |
| 94 | V10 | VSS | Power | GND | | | | | | |
| 95 | Y10 | D30 | I/O | Data | | | | | | ACCSIZE1 |
| 96 | W10 | D31 | I/O | Data | | | | | | ACCSIZE2 |
| 97 | U10 | VDD | Power | Internal VDD | | | | | | |
| 98 | U11 | VSS | Power | GND | | | | | | |
| 99 | Y11 | A18 | O | Address | | | | | | |
| 100 | W11 | A19 | O | Address | | | | | | |
| 101 | V11 | A20 | O | Address | | | | | | |
| 102 | Y12 | A21 | O | Address | | | | | | |
| 103 | W12 | A22 | O | Address | | | | | | |
| 104 | V12 | A23 | O | Address | | | | | | |
| 105 | U15 | VDDQ | Power | IO VDD | | | | | | |
| 106 | U17 | VSS | Power | GND | | | | | | |
| 107 | Y13 | A24 | O | Address | | | | | | |
| 108 | W13 | A25 | O | Address | | | | | | |
| 109 | V13 | $\overline{WE2}/$ \overline{ICIORD} | O | D23–D16 select signal | | WE2 | | | | \overline{ICIORD} |
| 110 | Y14 | $\overline{WE3}/$ \overline{CIOWR} | O | D31–D24 select signal | | WE3 | | | | \overline{CIOWR} |
| 111 | U14 | VDD | Power | Internal VDD | | | | | | |
| 112 | U13 | VSS | Power | GND | | | | | | |
| 113 | W14 | \overline{SLEEP} | I | Sleep | | | | | | |
| 114 | Y15 | $\overline{PCIGNT4}$ | O | Bus grant (host function) | | | | | | |
| 115 | W15 | $\overline{PCIGNT3}$ | O | Bus grant (host function) | | | | | | |
| 116 | V15 | $\overline{PCIGNT2}$ | O | Bus grant (host function) | | | | | | |
| 117 | Y16 | $\overline{PCIREQ4}$ | I* | Bus grant (host function) | | | | | | |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|---|-------|--|-------|------------------|--------|--------|--------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 118 | W16 | $\overline{\text{PCIREQ3}}$ / MD10 | I* | Bus request (host function)/ mode | MD10 | | | | | |
| 119 | V14 | VDDQ | Power | IO VDD | | | | | | |
| 120 | U16 | VSS | Power | GND | | | | | | |
| 121 | V16 | $\overline{\text{PCIREQ2}}$ / MD9 | I* | Bus request (host function)/ mode | MD9 | | | | | |
| 122 | Y17 | IDSEL | I | Configuration device select | | | | | | |
| 123 | W17 | $\overline{\text{INTA}}$ | O | Interrupt (async) | | | | | | |
| 124 | V17 | $\overline{\text{PCIRST}}$ | O | Reset output | | | | | | |
| 125 | Y18 | PCICLK | I | PCI input clock | | | | | | |
| 126 | W18 | $\overline{\text{PCIGNT1}}$ / $\overline{\text{REQOUT}}$ | O | Bus grant (host function)/ bus request | | | | | | |
| 127 | V18 | $\overline{\text{PCIREQ1}}$ / $\overline{\text{GNTIN}}$ | I | Bus grant (host function)/ bus request | | | | | | |
| 128 | Y19 | $\overline{\text{SERR}}$ | I/O | System error | | | | | | |
| 129 | Y20 | AD31 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 130 | W20 | AD30 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 131 | R17 | VDDQ | Power | IO VDD | | | | | | |
| 132 | T17 | VSS | Power | GND | | | | | | |
| 133 | W19 | AD29 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 134 | V20 | AD28 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 135 | V19 | AD27 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 136 | U20 | AD26 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 137 | U19 | AD25 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|----------------|-------|---------------------------|-------|------------------|--------|--------|--------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 138 | U18 | AD24 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 139 | T20 | C/BE $\bar{3}$ | I/O | PCI address/ data/port | | | | | | |
| 140 | T18 | AD23 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 141 | T19 | AD22 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 142 | R20 | AD21 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 143 | P18 | VDDQ | Power | IO VDD | | | | | | |
| 144 | U12 | VDDQ | Power | I/O VDD | | | | | | |
| 145 | P17 | VDD | Power | Internal VDD | | | | | | |
| 146 | N17 | VSS | Power | GND | | | | | | |
| 147 | R19 | AD20 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 148 | R18 | AD19 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 149 | P20 | AD18 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 150 | P19 | AD17 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 151 | N20 | AD16 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 152 | N18 | C/BE $\bar{2}$ | I/O | Command/ byte enable | | | | | | |
| 153 | N19 | PCIFRAME | I/O | Bus cycle | | | | | | |
| 154 | M20 | IRDY | I/O | Initiator ready | | | | | | |
| 155 | M19 | TRDY | I/O | Target ready | | | | | | |
| 156 | M18 | DEVSEL | I/O | Device select | | | | | | |
| 157 | M17 | VDDQ | Power | IO VDD | | | | | | |
| 158 | L17 | VDD | Power | Internal VDD | | | | | | |
| 159 | L20 | PCISTOP | I/O | Transaction stop | | | | | | |
| 160 | L19 | PCILOCK | I/O | Exclusive access | | | | | | |
| 161 | L18 | PERR | I/O | Parity error | | | | | | |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|----------------|-------|---------------------------|-------|------------------|--------|--------|--------|--------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 162 | K20 | PAR | I/O | Parity | | | | | | |
| 163 | K19 | C/BE $\bar{1}$ | I/O | Command/ byte enable | | | | | | |
| 164 | K18 | AD15 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 165 | J20 | AD14 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 166 | J19 | AD13 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 167 | J18 | AD12 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 168 | H20 | AD11 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 169 | F17 | VDDQ | Power | IO VDD | | | | | | |
| 170 | K17 | VSS | Power | GND | | | | | | |
| 171 | H19 | AD10 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 172 | H18 | AD9 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 173 | G20 | AD8 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 174 | G19 | C/BE $\bar{0}$ | I/O | Command/ byte enable | | | | | | |
| 175 | G17 | VDD | Power | Internal VDD | | | | | | |
| 176 | H17 | VSS | Power | GND | | | | | | |
| 177 | F20 | AD7 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 178 | F19 | AD6 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 179 | F18 | AD5 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 180 | E20 | AD4 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 181 | E19 | AD3 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 182 | E18 | AD2 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 183 | G18 | VDDQ | Power | I/O VDD | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--|-------|--|-------|-------------------------|--------|--------|----------------------------|-------------------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 184 | J17 | VDDQ | Power | I/O VDD | | | | | | |
| 185 | D20 | AD1 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 186 | D19 | AD0 | I/O | PCI address/ data/port | | (Port) | (Port) | (Port) | (Port) | (Port) |
| 187 | D18 | $\overline{\text{IRL0}}$ | I | Interrupt 0 | | | | | | |
| 188 | C20 | $\overline{\text{IRL1}}$ | I | Interrupt 1 | | | | | | |
| 189 | C19 | $\overline{\text{IRL2}}$ | I | Interrupt 2 | | | | | | |
| 190 | B20 | $\overline{\text{IRL3}}$ | I | Interrupt 3 | | | | | | |
| 191 | B18 | VSS-RTC | Power | RTC GND | | | | | | |
| 192 | E17 | VSS | Power | GND | | | | | | |
| 193 | A20 | XTAL2 | O | RTC crystal resonator pin | | | | | | |
| 194 | A19 | EXTAL2 | I | RTC crystal resonator pin | | | | | | |
| 195 | A18 | VDD-RTC | Power | RTC VDD | | | | | | |
| 196 | B19 | VDDQ | Power | IO VDD | | | | | | |
| 197 | C18 | CA | I | Hardware standby | | | | | | |
| 198 | A17 | $\overline{\text{RESET}}$ | I | Reset | | | | | RESET | |
| 199 | B17 | $\overline{\text{TRST}}$ | I | Reset (H-UDI) | | | | | | |
| 200 | C17 | $\overline{\text{MRESET}}$ | I | Manual reset | | | | | | |
| 201 | A16 | NMI | I | Nonmaskable interrupt | | | | | | |
| 202 | B16 | $\overline{\text{BACK}}/\overline{\text{BSREQ}}$ | O | Bus acknowledge/ bus request | | | | | | |
| 203 | C16 | $\overline{\text{BREQ}}/\overline{\text{BSACK}}$ | I | Bus request/bus acknowledge | | | | | | |
| 204 | A15 | MD6/ $\overline{\text{IOIS16}}$ | I | Mode/ $\overline{\text{IOIS16}}$ (PCMCIA) | MD6 | | | | $\overline{\text{IOIS16}}$ | |
| 205 | B15 | $\overline{\text{RDY}}$ | I | Bus ready | | $\overline{\text{RDY}}$ | | | $\overline{\text{RDY}}$ | $\overline{\text{RDY}}$ |
| 206 | C15 | TXD | O | SCI data output | | | | | | |
| 207 | C14 | VDDQ | Power | IO VDD | | | | | | |
| 208 | C11 | VSS | Power | GND | | | | | | |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|----------|-------|------------------------------|-------|------------------|------|-------|--------|------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 209 | D14 | VDD | Power | Internal VDD | | | | | | |
| 210 | D16 | VSS | Power | GND | | | | | | |
| 211 | A14 | MD2/RXD2 | I | Mode/SCIF data input | MD2 | RXD2 | RXD2 | RXD2 | RXD2 | RXD2 |
| 212 | B14 | RXD | I | SCI data input | | | | | | |
| 213 | A13 | TCLK | I/O | RTC/TMU clock | | | | | | |
| 214 | B13 | MD8/RTS2 | I/O | Mode/SCIF data control (RTS) | MD8 | RTS2 | RTS2 | RTS2 | RTS2 | RTS2 |
| 215 | C13 | SCK | I/O | SCIF clock | | | | | | |
| 216 | A12 | MD1/TXD2 | I/O | Mode/SCIF data output | MD1 | TXD2 | TXD2 | TXD2 | TXD2 | TXD2 |
| 217 | B12 | MD0/SCK2 | I/O | Mode/SCIF clock | MD0 | SCK2 | SCK2 | SCK2 | SCK2 | SCK2 |
| 218 | C12 | MD7/CTS2 | I/O | Mode/SCIF data control (RTS) | MD7 | CTS2 | CTS2 | CTS2 | CTS2 | CTS2 |
| 219 | A11 | AUDSYNC | | AUD Sync | | | | | | |
| 220 | B11 | AUDCK | | AUD clock | | | | | | |
| 221 | D15 | VDDQ | Power | IO VDD | | | | | | |
| 222 | D10 | VSS | Power | GND | | | | | | |
| 223 | A10 | AUDATA0 | | AUD data | | | | | | |
| 224 | B10 | AUDATA1 | | AUD data | | | | | | |
| 225 | D11 | VDD | Power | Internal VDD | | | | | | |
| 226 | D17 | VSS | Power | GND | | | | | | |
| 227 | C10 | AUDATA2 | | AUD data | | | | | | |
| 228 | A9 | AUDATA3 | | AUD data | | | | | | |
| 229 | D8 | VSS | | GND | | | | | | |
| 230 | B9 | MD3/CE2A | I/O | Mode/PCMCIA-CE | MD3 | | | | CE2A | |
| 231 | C9 | MD4/CE2B | I/O | Mode/PCMCIA-CE | MD4 | | | | CE2B | |
| 232 | A8 | MD5 | I | Mode | MD5 | | | | | |
| 233 | D12 | VDDQ | Power | IO VDD | | | | | | |
| 234 | D9 | VDDQ | Power | I/O VDD | | | | | | |

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | | |
|-----|------------|--|-------|--|-------|------------------|------|-------|--------|-----|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA | MPX |
| 235 | B8 | DACK0 | O | DMAC0 bus acknowledge | | | | | | |
| 236 | C8 | DACK1 | O | DMAC1 bus acknowledge | | | | | | |
| 237 | A7 | DRAK0 | O | DMAC0 request acknowledge | | | | | | |
| 238 | B7 | DRAK1 | O | DMAC1 request acknowledge | | | | | | |
| 239 | D7 | VDD | Power | Internal VDD | | | | | | |
| 240 | D6 | VDDQ | Power | I/O VDD | | | | | | |
| 241 | A6 | STATUS0 | O | Status | | | | | | |
| 242 | B6 | STATUS1 | O | Status | | | | | | |
| 243 | C6 | $\overline{\text{DREQ0}}$ | I | Request from DMAC0 | | | | | | |
| 244 | C5 | $\overline{\text{DREQ1}}$ | I | Request from DMAC1 | | | | | | |
| 245 | B5 | $\overline{\text{ASEBRK}}$ / BRKACK | I/O | Pin break/acknowledge (H-UDI) | | | | | | |
| 246 | C4 | TDO | O | Data out (H-UDI) | | | | | | |
| 247 | C7 | VDDQ | Power | IO VDD | | | | | | |
| 248 | D13 | VSS | Power | GND | | | | | | |
| 249 | A5 | VDD-PLL2 | Power | PLL2 VDD | | | | | | |
| 250 | B4 | VSS-PLL2 | Power | PLL2 GND | | | | | | |
| 251 | A4 | VDD-PLL1 | Power | PLL1 VDD | | | | | | |
| 252 | C3 | VSS-PLL1 | Power | PLL1 GND | | | | | | |
| 253 | B3 | VDD-CPG | Power | CPG VDD | | | | | | |
| 254 | A3 | VSS-CPG | Power | CPG GND | | | | | | |
| 255 | A2 | XTAL | O | Crystal resonator | | | | | | |
| 256 | A1 | EXTAL | I | External input clock/crystal resonator | | | | | | |
| 257 | H8 | VSS | Power | GND | | | | | | |
| 258 | J8 | VSS | Power | GND | | | | | | |
| 259 | K8 | VSS | Power | GND | | | | | | |

Memory Interface

| No. | Pin Number | Pin Name | I/O | Function | Reset | Memory Interface | | | |
|-----|------------|----------|-------|----------|-------|------------------|------|-------|------------|
| | | | | | | SRAM | DRAM | SDRAM | PCMCIA MPX |
| 260 | L8 | VSS | Power | GND | | | | | |
| 261 | M8 | VSS | Power | GND | | | | | |
| 262 | N8 | VSS | Power | GND | | | | | |
| 263 | N9 | VSS | Power | GND | | | | | |
| 264 | N10 | VSS | Power | GND | | | | | |
| 265 | N11 | VSS | Power | GND | | | | | |
| 266 | N12 | VSS | Power | GND | | | | | |
| 267 | N13 | VSS | Power | GND | | | | | |
| 268 | M13 | VSS | Power | GND | | | | | |
| 269 | L13 | VSS | Power | GND | | | | | |
| 270 | K13 | VSS | Power | GND | | | | | |
| 271 | J13 | VSS | Power | GND | | | | | |
| 272 | H13 | VSS | Power | GND | | | | | |
| 273 | H12 | VSS | Power | GND | | | | | |
| 274 | H11 | VSS | Power | GND | | | | | |
| 275 | H10 | VSS | Power | GND | | | | | |
| 276 | H9 | VSS | Power | GND | | | | | |
| 277 | J9 | VSS | Power | GND | | | | | |
| 278 | K9 | VSS | Power | GND | | | | | |
| 279 | L9 | VSS | Power | GND | | | | | |
| 280 | M9 | VSS | Power | GND | | | | | |
| 281 | M10 | VSS | Power | GND | | | | | |
| 282 | M11 | VSS | Power | GND | | | | | |
| 283 | M12 | VSS | Power | GND | | | | | |
| 284 | L12 | VSS | Power | GND | | | | | |
| 285 | K12 | VSS | Power | GND | | | | | |
| 286 | J12 | VSS | Power | GND | | | | | |
| 287 | J11 | VSS | Power | GND | | | | | |
| 288 | J10 | VSS | Power | GND | | | | | |
| 289 | K10 | VSS | Power | GND | | | | | |
| 290 | L10 | VSS | Power | GND | | | | | |
| 291 | L11 | VSS | Power | GND | | | | | |
| 292 | K11 | VSS | Power | GND | | | | | |

Legend:

I: Input

O: Output

I/O: Input/output

Power: Power supply

Notes: Supply power to all power pins.

Power must be supplied to VDD-PLL1/2 and VSS-PLL1/2 regardless of whether or not the on-chip PLL circuits are used.

Power must be supplied to VDD-CPG and VSS-CPG regardless of whether or not the on-chip crystal oscillation circuit is used.

Power must be supplied to VDD-RTC and VSS-RTC regardless of whether or not the on-chip RTC is used.

For the handling of the PCI bus pins in PCI-disabled mode, see table D.4 in appendix D.

* I/O attribute is I/O when used as a port.

Section 2 Programming Model

2.1 Data Formats

The data formats handled by the SH-4 are shown in figure 2.1.

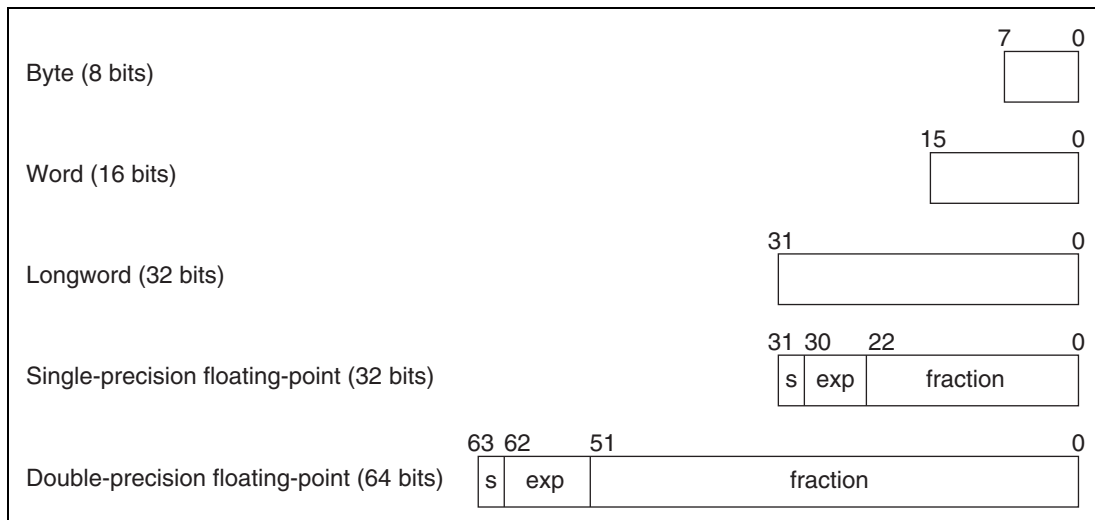


Figure 2.1 Data Formats

2.2 Register Configuration

2.2.1 Privileged Mode and Banks

Processor Modes: The SH-4 has two processor modes, user mode and privileged mode. The SH-4 normally operates in user mode, and switches to privileged mode when an exception occurs or an interrupt is accepted. There are four kinds of registers—general registers, system registers, control registers, and floating-point registers—and the registers that can be accessed differ in the two processor modes.

General Registers: There are 16 general registers, designated R0 to R15. General registers R0 to R7 are banked registers which are switched by a processor mode change.

In privileged mode, the register bank bit (RB) in the status register (SR) defines which banked register set is accessed as general registers, and which set is accessed only through the load control register (LDC) and store control register (STC) instructions.

When the RB bit is 1 (that is, when bank 1 is selected), the 16 registers comprising bank 1 general registers R0_BANK1 to R7_BANK1 and non-banked general registers R8 to R15 can be accessed as general registers R0 to R15. In this case, the eight registers comprising bank 0 general registers R0_BANK0 to R7_BANK0 are accessed by the LDC/STC instructions. When the RB bit is 0 (that is, when bank 0 is selected), the 16 registers comprising bank 0 general registers R0_BANK0 to R7_BANK0 and non-banked general registers R8 to R15 can be accessed as general registers R0 to R15. In this case, the eight registers comprising bank 1 general registers R0_BANK1 to R7_BANK1 are accessed by the LDC/STC instructions.

In user mode, the 16 registers comprising bank 0 general registers R0_BANK0 to R7_BANK0 and non-banked general registers R8 to R15 can be accessed as general registers R0 to R15. The eight registers comprising bank 1 general registers R0_BANK1 to R7_BANK1 cannot be accessed.

Control Registers: Control registers comprise the global base register (GBR) and status register (SR), which can be accessed in both processor modes, and the saved status register (SSR), saved program counter (SPC), vector base register (VBR), saved general register 15 (SGR), and debug base register (DBR), which can only be accessed in privileged mode. Some bits of the status register (such as the RB bit) can only be accessed in privileged mode.

System Registers: System registers comprise the multiply-and-accumulate registers (MACH/MACL), the procedure register (PR), the program counter (PC), the floating-point status/control register (FPSCR), and the floating-point communication register (FPUL). Access to these registers does not depend on the processor mode.

Floating-Point Registers: There are thirty-two floating-point registers, FR0–FR15 and XF0–XF15. FR0–FR15 and XF0–XF15 can be assigned to either of two banks (FPR0_BANK0–FPR15_BANK0 or FPR0_BANK1–FPR15_BANK1).

FR0–FR15 can be used as the eight registers DR0/2/4/6/8/10/12/14 (double-precision floating-point registers, or pair registers) or the four registers FV0/4/8/12 (register vectors), while XF0–XF15 can be used as the eight registers XD0/2/4/6/8/10/12/14 (register pairs) or register matrix XMTRX.

Register values after a reset are shown in table 2.1.

Table 2.1 Initial Register Values

| Type | Registers | Initial Value* |
|--------------------------|--|---|
| General registers | R0_BANK0–R7_BANK0, R0_BANK1–R7_BANK1, R8–R15 | Undefined |
| Control registers | SR | MD bit = 1, RB bit = 1, BL bit = 1, FD bit = 0, IMASK = 1111 (H'F), reserved bits = 0, others undefined |
| | GBR, SSR, SPC, SGR, DBR | Undefined |
| | VBR | H'00000000 |
| System registers | MACH, MACL, PR, FPUL | Undefined |
| | PC | H'A0000000 |
| | FPSCR | H'00040001 |
| Floating-point registers | FR0–FR15, XF0–XF15 | Undefined |

Note: * Initialized by a power-on reset and manual reset.

The register configuration in each processor mode is shown in figure 2.2.

Switching between user mode and privileged mode is controlled by the processor mode bit (MD) in the status register.

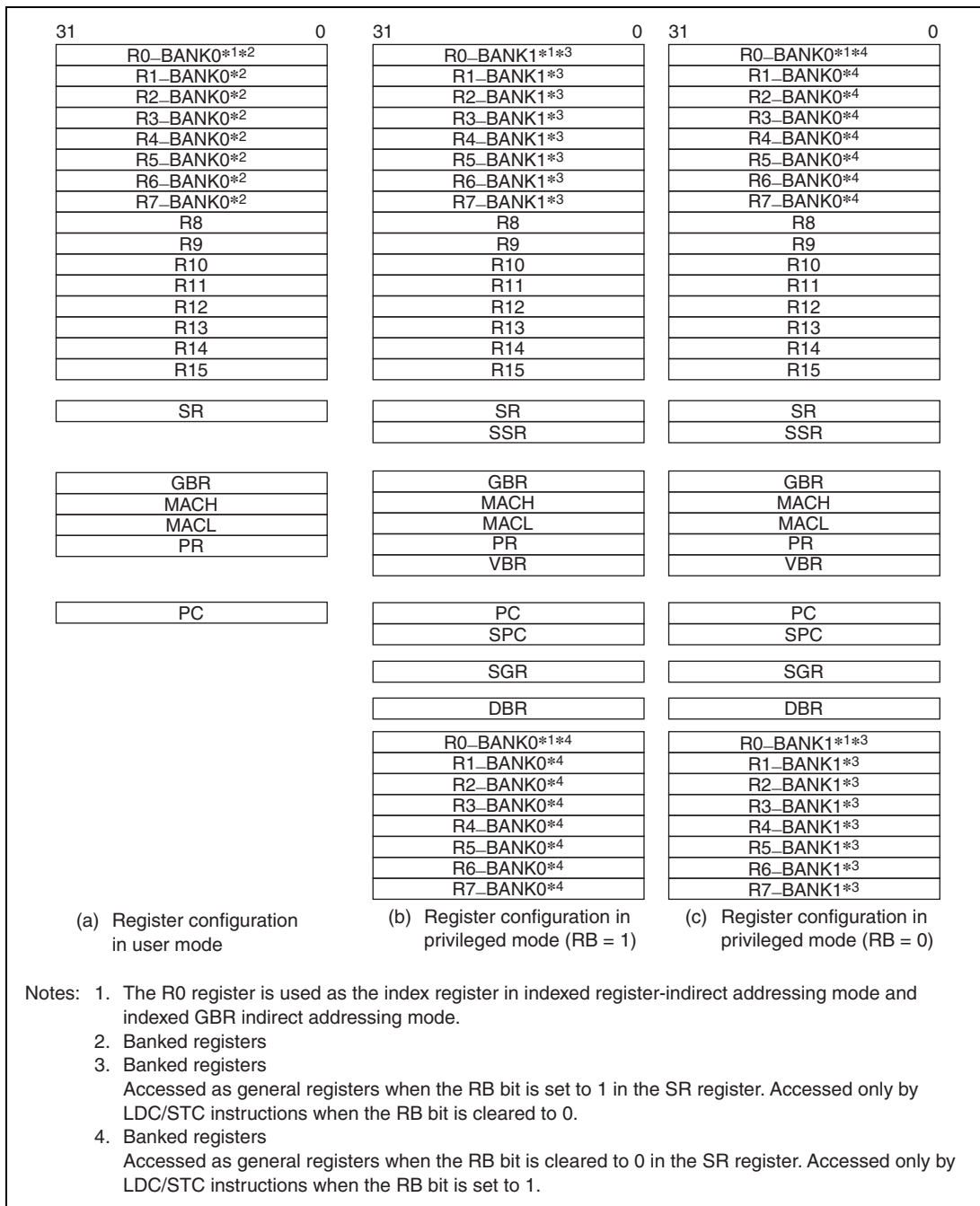


Figure 2.2 CPU Register Configuration in Each Processor Mode

2.2.2 General Registers

Figure 2.3 shows the relationship between the processor modes and general registers. The SH-4 has twenty-four 32-bit general registers (R0_BANK0–R7_BANK0, R0_BANK1–R7_BANK1, and R8–R15). However, only 16 of these can be accessed as general registers R0–R15 in one processor mode. The SH-4 has two processor modes, user mode and privileged mode, in which R0–R7 are assigned as shown below.

- R0_BANK0–R7_BANK0

In user mode (SR.MD = 0), R0–R7 are always assigned to R0_BANK0–R7_BANK0.

In privileged mode (SR.MD = 1), R0–R7 are assigned to R0_BANK0–R7_BANK0 only when SR.RB = 0.

- R0_BANK1–R7_BANK1

In user mode, R0_BANK1–R7_BANK1 cannot be accessed.

In privileged mode, R0–R7 are assigned to R0_BANK1–R7_BANK1 only when SR.RB = 1.

| SR.MD = 0 or (SR.MD = 1, SR.RB = 0) | | (SR.MD = 1, SR.RB = 1) |
|--|----------|------------------------|
| R0 | R0_BANK0 | R0_BANK0 |
| R1 | R1_BANK0 | R1_BANK0 |
| R2 | R2_BANK0 | R2_BANK0 |
| R3 | R3_BANK0 | R3_BANK0 |
| R4 | R4_BANK0 | R4_BANK0 |
| R5 | R5_BANK0 | R5_BANK0 |
| R6 | R6_BANK0 | R6_BANK0 |
| R7 | R7_BANK0 | R7_BANK0 |
| R0_BANK1 | R0_BANK1 | R0 |
| R1_BANK1 | R1_BANK1 | R1 |
| R2_BANK1 | R2_BANK1 | R2 |
| R3_BANK1 | R3_BANK1 | R3 |
| R4_BANK1 | R4_BANK1 | R4 |
| R5_BANK1 | R5_BANK1 | R5 |
| R6_BANK1 | R6_BANK1 | R6 |
| R7_BANK1 | R7_BANK1 | R7 |
| R8 | R8 | R8 |
| R9 | R9 | R9 |
| R10 | R10 | R10 |
| R11 | R11 | R11 |
| R12 | R12 | R12 |
| R13 | R13 | R13 |
| R14 | R14 | R14 |
| R15 | R15 | R15 |

Figure 2.3 General Registers

Programming Note: As the user's R0–R7 are assigned to R0_BANK0–R7_BANK0, and after an exception or interrupt R0–R7 are assigned to R0_BANK1–R7_BANK1, it is not necessary for the interrupt handler to save and restore the user's R0–R7 (R0_BANK0–R7_BANK0).

After a reset, the values of R0_BANK0–R7_BANK0, R0_BANK1–R7_BANK1, and R8–R15 are undefined.

2.2.3 Floating-Point Registers

Figure 2.4 shows the floating-point registers. There are thirty-two 32-bit floating-point registers, divided into two banks (FPR0_BANK0–FPR15_BANK0 and FPR0_BANK1–FPR15_BANK1). These 32 registers are referenced as FR0–FR15, DR0/2/4/6/8/10/12/14, FV0/4/8/12, XF0–XF15, XD0/2/4/6/8/10/12/14, or XMTRX. The correspondence between FPRn_BANKi and the reference name is determined by the FR bit in FPSCR (see figure 2.4).

- Floating-point registers, FPRn_BANKi (32 registers)
 FPR0_BANK0, FPR1_BANK0, FPR2_BANK0, FPR3_BANK0, FPR4_BANK0,
 FPR5_BANK0, FPR6_BANK0, FPR7_BANK0, FPR8_BANK0, FPR9_BANK0,
 FPR10_BANK0, FPR11_BANK0, FPR12_BANK0, FPR13_BANK0, FPR14_BANK0,
 FPR15_BANK0

 FPR0_BANK1, FPR1_BANK1, FPR2_BANK1, FPR3_BANK1, FPR4_BANK1,
 FPR5_BANK1, FPR6_BANK1, FPR7_BANK1, FPR8_BANK1, FPR9_BANK1,
 FPR10_BANK1, FPR11_BANK1, FPR12_BANK1, FPR13_BANK1, FPR14_BANK1,
 FPR15_BANK1
- Single-precision floating-point registers, FRi (16 registers)
 When FPSCR.FR = 0, FR0–FR15 are assigned to FPR0_BANK0–FPR15_BANK0.
 When FPSCR.FR = 1, FR0–FR15 are assigned to FPR0_BANK1–FPR15_BANK1.
- Double-precision floating-point registers or single-precision floating-point register pairs, DRi (8 registers): A DR register comprises two FR registers.
 DR0 = {FR0, FR1}, DR2 = {FR2, FR3}, DR4 = {FR4, FR5}, DR6 = {FR6, FR7},
 DR8 = {FR8, FR9}, DR10 = {FR10, FR11}, DR12 = {FR12, FR13}, DR14 = {FR14, FR15}
- Single-precision floating-point vector registers, FVi (4 registers): An FV register comprises four FR registers
 FV0 = {FR0, FR1, FR2, FR3}, FV4 = {FR4, FR5, FR6, FR7},
 FV8 = {FR8, FR9, FR10, FR11}, FV12 = {FR12, FR13, FR14, FR15}
- Single-precision floating-point extended registers, XFi (16 registers)
 When FPSCR.FR = 0, XF0–XF15 are assigned to FPR0_BANK1–FPR15_BANK1.
 When FPSCR.FR = 1, XF0–XF15 are assigned to FPR0_BANK0–FPR15_BANK0.
- Single-precision floating-point extended register pairs, XD_i (8 registers): An XD register comprises two XF registers

XD0 = {XF0, XF1}, XD2 = {XF2, XF3}, XD4 = {XF4, XF5}, XD6 = {XF6, XF7},
 XD8 = {XF8, XF9}, XD10 = {XF10, XF11}, XD12 = {XF12, XF13}, XD14 = {XF14, XF15}

- Single-precision floating-point extended register matrix, XMTRX: XMTRX comprises all 16 XF registers

$$\text{XMTRX} = \begin{bmatrix} \text{XF0} & \text{XF4} & \text{XF8} & \text{XF12} \\ \text{XF1} & \text{XF5} & \text{XF9} & \text{XF13} \\ \text{XF2} & \text{XF6} & \text{XF10} & \text{XF14} \\ \text{XF3} & \text{XF7} & \text{XF11} & \text{XF15} \end{bmatrix}$$

| <u>FPSCR.FR = 0</u> | | | | <u>FPSCR.FR = 1</u> | | | |
|---------------------|------|------|-------------|---------------------|------|-------|------|
| FV0 | DR0 | FR0 | FPR0_BANK0 | XF0 | XD0 | XMTRX | |
| | | FR1 | FPR1_BANK0 | | | | XF1 |
| FV4 | DR2 | FR2 | FPR2_BANK0 | XF2 | XD2 | | |
| | | FR3 | FPR3_BANK0 | | | XF3 | |
| | | FR4 | FPR4_BANK0 | | | XF4 | XD4 |
| FV8 | DR4 | FR5 | FPR5_BANK0 | XF5 | XD6 | | |
| | | FR6 | FPR6_BANK0 | | | XF6 | |
| | | FR7 | FPR7_BANK0 | | | XF7 | |
| FV12 | DR6 | FR8 | FPR8_BANK0 | XF8 | XD8 | | |
| | | FR9 | FPR9_BANK0 | | | XF9 | |
| | | FR10 | FPR10_BANK0 | | | XF10 | XD10 |
| FV16 | DR8 | FR11 | FPR11_BANK0 | XF11 | XD12 | | |
| | | FR12 | FPR12_BANK0 | | | XF12 | |
| | | FR13 | FPR13_BANK0 | | | XF13 | |
| FV20 | DR10 | FR14 | FPR14_BANK0 | XF14 | XD14 | | |
| | | FR15 | FPR15_BANK0 | | | XF15 | |
| | | | | | | | |
| XMTRX | XD0 | XF0 | FPR0_BANK1 | FR0 | DR0 | FV0 | |
| | | XF1 | FPR1_BANK1 | | | | FR1 |
| XMTRX | XD2 | XF2 | FPR2_BANK1 | FR2 | DR2 | | |
| | | XF3 | FPR3_BANK1 | | | FR3 | |
| | | XF4 | FPR4_BANK1 | | | FR4 | DR4 |
| XMTRX | XD4 | XF5 | FPR5_BANK1 | FR5 | DR6 | | |
| | | XF6 | FPR6_BANK1 | | | FR6 | |
| | | XF7 | FPR7_BANK1 | | | FR7 | |
| XMTRX | XD6 | XF8 | FPR8_BANK1 | FR8 | DR8 | FV8 | |
| | | XF9 | FPR9_BANK1 | | | | FR9 |
| | | XF10 | FPR10_BANK1 | | | | FR10 |
| XMTRX | XD8 | XF11 | FPR11_BANK1 | FR11 | DR12 | FV12 | |
| | | XF12 | FPR12_BANK1 | | | | FR12 |
| | | XF13 | FPR13_BANK1 | | | | FR13 |
| XMTRX | XD10 | XF14 | FPR14_BANK1 | FR14 | DR14 | | |
| | | XF15 | FPR15_BANK1 | | | FR15 | |
| | | | | | | | |

Figure 2.4 Floating-Point Registers

Programming Note: After a reset, the values of FPR0_BANK0–FPR15_BANK0 and FPR0_BANK1–FPR15_BANK1 are undefined.

2.2.4 Control Registers

Status register, SR (32 bits, privilege protection, initial value = 0111 0000 0000 0000 0000 00XX 1111 00XX (X = undefined))

| | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|---|----|----|----|--|----|---|---|-------|--|---|---|---|---|---|
| 31 | 30 | 29 | 28 | 27 | | 16 | 15 | 14 | | 10 | 9 | 8 | 7 | | 4 | 3 | 2 | 1 | 0 |
| — | MD | RB | BL | | — | FD | | — | | M | Q | | IMASK | | — | | S | T | |

Note: —: Reserved. These bits are always read as 0, and should only be written with 0.

- **MD: Processor mode**
 MD = 0: User mode (some instructions cannot be executed, and some resources cannot be accessed)
 MD = 1: Privileged mode
- **RB: General register specification bit in privileged mode (set to 1 by a reset, exception, or interrupt)**
 RB = 0: R0_BANK0–R7_BANK0 are accessed as general registers R0–R7. (R0_BANK1–R7_BANK1 can be accessed using LDC/STC instructions.)
 RB = 1: R0_BANK1–R7_BANK1 are accessed as general registers R0–R7. (R0_BANK0–R7_BANK0 can be accessed using LDC/STC instructions.)
- **BL: Exception/interrupt block bit (set to 1 by a reset, exception, or interrupt)**
 BL = 1: Interrupt requests are masked. If a general exception other than a user break occurs while BL = 1, the processor switches to the reset state.
- **FD: FPU disable bit (cleared to 0 by a reset)**
 FD = 1: An FPU instruction causes a general FPU disable exception, and if the FPU instruction is in a delay slot, a slot FPU disable exception is generated. (FPU instructions: HF*** instructions, LDC(.L)/STS(.L) instructions for FPUL/FPSCR)
- **M, Q: Used by the DIV0S, DIV0U, and DIV1 instructions.**
- **IMASK: Interrupt mask level**
 Interrupts of a lower level than IMASK are masked. IMASK does not change when an interrupt is generated.
- **S: Specifies a saturation operation for a MAC instruction.**

- T: True/false condition or carry/borrow bit

Saved status register, SSR (32 bits, privilege protection, initial value undefined): The current contents of SR are saved to SSR in the event of an exception or interrupt.

Saved program counter, SPC (32 bits, privilege protection, initial value undefined): The address of an instruction at which an interrupt or exception occurs is saved to SPC.

Global base register, GBR (32 bits, initial value undefined): GBR is referenced as the base address in a GBR-referencing MOV instruction.

Vector base register, VBR (32 bits, privilege protection, initial value = H'0000 0000): VBR is referenced as the branch destination base address in the event of an exception or interrupt. For details, see section 5, Exceptions.

Saved general register 15, SGR (32 bits, privilege protection, initial value undefined): The contents of R15 are saved to SGR in the event of an exception or interrupt.

Debug base register, DBR (32 bits, privilege protection, initial value undefined): When the user break debug function is enabled (BRCR.UBDE = 1), DBR is referenced as the user break handler branch destination address instead of VBR.

2.2.5 System Registers

Multiply-and-accumulate register high, MACH (32 bits, initial value undefined)

Multiply-and-accumulate register low, MACL (32 bits, initial value undefined)

MACH/MACL is used for the added value in a MAC instruction, and to store a MAC instruction or MUL instruction operation result.

Procedure register, PR (32 bits, initial value undefined): The return address is stored in PR in a subroutine call using a BSR, BSRF, or JSR instruction, and PR is referenced by the subroutine return instruction (RTS).

Program counter, PC (32 bits, initial value = H'A000 0000): PC indicates the executing instruction address.

Floating-point status/control register, FPSCR (32 bits, initial value = H'0004 0001)

| | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|-------|--------|------|----|---|---|
| 31 | 22 | 21 | 20 | 19 | 18 | 17 | 12 | 11 | 7 | 6 | 2 | 1 | 0 |
| — | | | | FR | SZ | PR | DN | Cause | Enable | Flag | RM | | |

Note: —: Reserved. These bits are always read as 0, and should only be written with 0.

- **FR: Floating-point register bank**
FR = 0: FPR0_BANK0–FPR15_BANK0 are assigned to FR0–FR15; FPR0_BANK1–FPR15_BANK1 are assigned to XF0–XF15.
FR = 1: FPR0_BANK0–FPR15_BANK0 are assigned to XF0–XF15; FPR0_BANK1–FPR15_BANK1 are assigned to FR0–FR15.
- **SZ: Transfer size mode**
SZ = 0: The data size of the FMOV instruction is 32 bits.
SZ = 1: The data size of the FMOV instruction is a 32-bit register pair (64 bits).
- **PR: Precision mode**
PR = 0: Floating-point instructions are executed as single-precision operations.
PR = 1: Floating-point instructions are executed as double-precision operations (the result of instructions for which double-precision is not supported is undefined).
Do not set SZ and PR to 1 simultaneously; this setting is reserved.
[SZ, PR = 11]: Reserved (FPU operation instruction is undefined).
- **DN: Denormalization mode**
DN = 0: A denormalized number is treated as such.
DN = 1: A denormalized number is treated as zero.
- **Cause: FPU exception cause field**
- **Enable: FPU exception enable field**
- **Flag: FPU exception flag field**

| | | FPU Error (E) | Invalid Operation (V) | Division by Zero (Z) | Overflow (O) | Underflow (U) | Inexact (I) |
|--------|----------------------------|----------------------|------------------------------|-----------------------------|---------------------|----------------------|--------------------|
| Cause | FPU exception cause field | Bit 17 | Bit 16 | Bit 15 | Bit 14 | Bit 13 | Bit 12 |
| Enable | FPU exception enable field | None | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 |
| Flag | FPU exception flag field | None | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 |

When an FPU operation instruction is executed, the FPU exception cause field is cleared to zero first. When the next FPU exception is occurred, the corresponding bits in the FPU exception cause field and FPU exception flag field are set to 1. The FPU exception flag field holds the status of the exception generated after the field was last cleared.

- RM: Rounding mode
RM = 00: Round to Nearest
RM = 01: Round to Zero
RM = 10: Reserved
RM = 11: Reserved
- Bits 22 to 31: Reserved

Floating-point communication register, FPUL (32 bits, initial value undefined): Data transfer between FPU registers and CPU registers is carried out via the FPUL register.

Programming Note: When SZ = 1 and big endian mode is selected, FMOV can be used for double-precision floating-point data load or store operations. In little endian mode, two 32-bit data size moves must be executed, with SZ = 0, to load or store a double-precision floating-point data.

2.3 Memory-Mapped Registers

Appendix A shows the control registers mapped to memory. The control registers are double-mapped to the following two memory areas. All registers have two addresses.

H'1C00 0000–H'1FFF FFFF
H'FC00 0000–H'FFFF FFFF

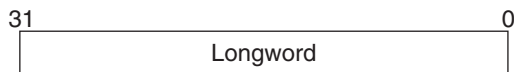
These two areas are used as follows.

- H'1C00 0000–H'1FFF FFFF
This area must be accessed using the address translation function of the MMU. Setting the page number of this area to the corresponding filed of the TLB enables access to a memory-mapped register. Accessing this area without using the address translation function of the MMU is not guaranteed.
- H'FC00 0000–H'FFFF FFFF
Access to area H'FC00 0000–H'FFFF FFFF in user mode will cause an address error. Memory-mapped registers can be referenced in user mode by means of access that involves address translation.

Note: Do not access undefined locations in either area. The operation of an access to an undefined location is undefined. Also, memory-mapped registers must be accessed using a fixed data size. The operation of an access using an invalid data size is undefined.

2.4 Data Format in Registers

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



2.5 Data Formats in Memory

Memory data formats are classified into bytes, words, and longwords. Memory can be accessed in 8-bit byte, 16-bit word, or 32-bit longword form. A memory operand less than 32 bits in length is sign-extended before being loaded into a register.

A word operand must be accessed starting from a word boundary (even address of a 2-byte unit: address $2n$), and a longword operand starting from a longword boundary (even address of a 4-byte unit: address $4n$). An address error will result if this rule is not observed. A byte operand can be accessed from any address.

Big endian or little endian byte order can be selected for the data format. The endian should be set with the MD5 external pin in a power-on reset. Big endian is selected when the MD5 pin is low, and little endian when high. The endian cannot be changed dynamically. Bit positions are numbered left to right from most-significant to least-significant. Thus, in a 32-bit longword, the leftmost bit, bit 31, is the most significant bit and the rightmost bit, bit 0, is the least significant bit.

The data format in memory is shown in figure 2.5.

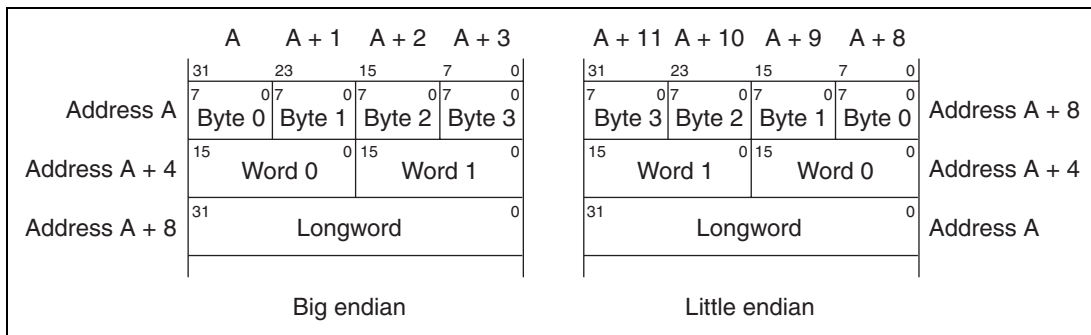


Figure 2.5 Data Formats In Memory

Note: The SH-4 does not support endian conversion for the 64-bit data format. Therefore, if double-precision floating-point format (64-bit) access is performed in little endian mode, the upper and lower 32 bits will be reversed.

2.6 Processor States

The SH-4 has five processor states: the reset state, exception-handling state, bus-released state, program execution state, and power-down state.

Reset State: In this state the CPU is reset. The power-on reset state is entered when the $\overline{\text{RESET}}$ pin goes low. The CPU enters the manual reset state if the $\overline{\text{RESET}}$ pin is high and the $\overline{\text{MRESET}}$ pin is low. For more information on resets, see section 5, Exceptions.

In the power-on reset state, the internal state of the CPU and the on-chip peripheral module registers are initialized. In the manual reset state, the internal state of the CPU and registers of on-chip peripheral modules other than the bus state controller (BSC) are initialized. Since the bus state controller (BSC) is not initialized in the manual reset state, refreshing operations continue. Refer to the register configurations in the relevant sections for further details.

Exception-Handling State: This is a transient state during which the CPU's processor state flow is altered by a reset, general exception, or interrupt exception source.

In the case of a reset, the CPU branches to address H'A000 0000 and starts executing the user-coded exception handling program.

In the case of a general exception or interrupt, the program counter (PC) contents are saved in the saved program counter (SPC), the status register (SR) contents are saved in the saved status register (SSR), and the R15 contents are saved in saved general register 15 (SGR). The CPU

branches to the start address of the user-coded exception service routine found from the sum of the contents of the vector base address and the vector offset. See section 5, Exceptions, for more information on resets, general exceptions, and interrupts.

Program Execution State: In this state the CPU executes program instructions in sequence.

Power-Down State: In the power-down state, CPU operation halts and power consumption is reduced. The power-down state is entered by executing a SLEEP instruction. There are three modes in the power-down state: sleep mode, deep sleep mode, and standby mode. For details, see section 9, Power-Down Modes.

Bus-Released State: In this state the CPU has released the bus to a device that requested it.

Transitions between the states are shown in figure 2.6.

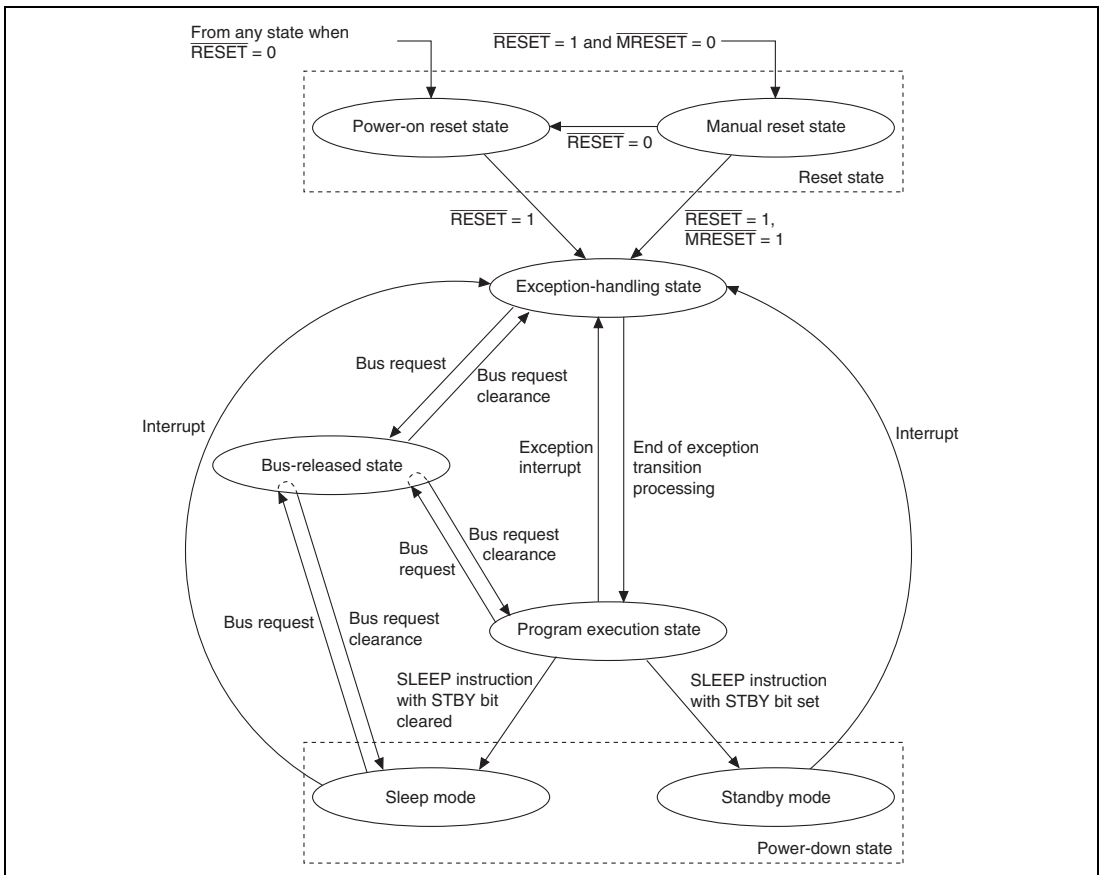


Figure 2.6 Processor State Transitions

2.7 Processor Modes

There are two processor modes: user mode and privileged mode. The processor mode is determined by the processor mode bit (MD) in the status register (SR). User mode is selected when the MD bit is cleared to 0, and privileged mode when the MD bit is set to 1. When the reset state or exception state is entered, the MD bit is set to 1. There are certain registers and bits which can only be accessed in privileged mode.

Section 3 Memory Management Unit (MMU)

3.1 Overview

3.1.1 Features

The SH-4 can handle 29-bit external memory space from an 8-bit address space identifier and 32-bit logical (virtual) address space. Address translation from virtual address to physical address is performed using the memory management unit (MMU) built into the SH-4. The MMU performs high-speed address translation by caching user-created address translation table information in an address translation buffer (translation lookaside buffer: TLB). The SH-4 has four instruction TLB (ITLB) entries and 64 unified TLB (UTLB) entries. UTLB copies are stored in the ITLB by hardware. A paging system is used for address translation, with support for four page sizes (1, 4, and 64 Kbytes, and 1 Mbyte). It is possible to set the virtual address space access right and implement storage protection independently for privileged mode and user mode.

3.1.2 Role of the MMU

The MMU was conceived as a means of making efficient use of physical memory. As shown in figure 3.1, when a process is smaller in size than the physical memory, the entire process can be mapped onto physical memory, but if the process increases in size to the point where it does not fit into physical memory, it becomes necessary to divide the process into smaller parts, and map the parts requiring execution onto physical memory on an ad hoc basis ((1)). Having this mapping onto physical memory executed consciously by the process itself imposes a heavy burden on the process. The virtual memory system was devised as a means of handling all physical memory mapping to reduce this burden ((2)). With a virtual memory system, the size of the available virtual memory is much larger than the actual physical memory, and processes are mapped onto this virtual memory. Thus processes only have to consider their operation in virtual memory, and mapping from virtual memory to physical memory is handled by the MMU. The MMU is normally managed by the OS, and physical memory switching is carried out so as to enable the virtual memory required by a task to be mapped smoothly onto physical memory. Physical memory switching is performed via secondary storage, etc.

The virtual memory system that came into being in this way works to best effect in a time sharing system (TSS) that allows a number of processes to run simultaneously ((3)). Running a number of processes in a TSS did not increase efficiency since each process had to take account of physical memory mapping. Efficiency is improved and the load on each process reduced by the use of a virtual memory system ((4)). In this system, virtual memory is allocated to each process. The task of the MMU is to map a number of virtual memory areas onto physical memory in an efficient

manner. It is also provided with memory protection functions to prevent a process from inadvertently accessing another process's physical memory.

When address translation from virtual memory to physical memory is performed using the MMU, it may happen that the translation information has not been recorded in the MMU, or the virtual memory of a different process is accessed by mistake. In such cases, the MMU will generate an exception, change the physical memory mapping, and record the new address translation information.

Although the functions of the MMU could be implemented by software alone, having address translation performed by software each time a process accessed physical memory would be very inefficient. For this reason, a buffer for address translation (the translation lookaside buffer: TLB) is provided in hardware, and frequently used address translation information is placed here. The TLB can be described as a cache for address translation information. However, unlike a cache, if address translation fails—that is, if an exception occurs—switching of the address translation information is normally performed by software. Thus memory management can be performed in a flexible manner by software.

There are two methods by which the MMU can perform mapping from virtual memory to physical memory: the paging method, using fixed-length address translation, and the segment method, using variable-length address translation. With the paging method, the unit of translation is a fixed-size address space called a page (usually from 1 to 64 Kbytes in size).

In the following descriptions, the address space in virtual memory in the SH-4 is referred to as virtual address space, and the address space in physical memory as physical address space.

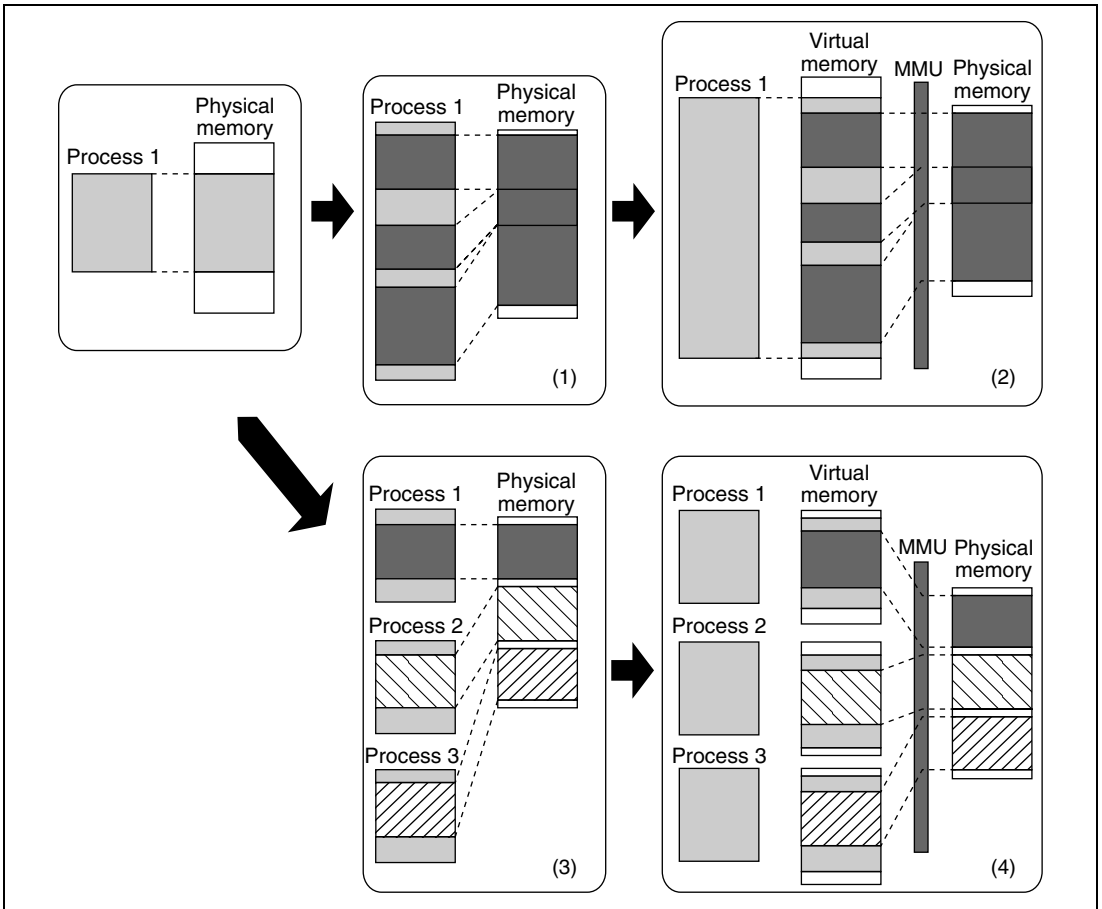


Figure 3.1 Role of the MMU

3.1.3 Register Configuration

The MMU registers are shown in table 3.1.

Table 3.1 MMU Registers

| Name | Abbrevia- tion | R/W | Initial Value* ¹ | P4 Address* ² | Area 7 Address* ² | Acces s Size |
|--------------------------------------|-------------------|-----|--------------------------------|-----------------------------|---------------------------------|-----------------|
| Page table entry high register | PTEH | R/W | Undefined | H'FF00 0000 | H'1F00 0000 | 32 |
| Page table entry low register | PTEL | R/W | Undefined | H'FF00 0004 | H'1F00 0004 | 32 |
| Page table entry assistance register | PTEA | R/W | Undefined | H'FF00 0034 | H'1F00 0034 | 32 |
| Translation table base register | TTB | R/W | Undefined | H'FF00 0008 | H'1F00 0008 | 32 |
| TLB exception address register | TEA | R/W | Undefined | H'FF00 000C | H'1F00 000C | 32 |
| MMU control register | MMUCR | R/W | H'0000 0000 | H'FF00 0010 | H'1F00 0010 | 32 |

Notes: 1. The initial value is the value after a power-on reset or manual reset.

2. P4 address is the address when using the virtual/physical address space P4 area. The area 7 address is the address used when making an access from physical address space area 7 using the TLB.

3.1.4 Caution

Operation is not guaranteed if an area designated as a reserved area in this manual is accessed.

3.2 Register Descriptions

There are six MMU-related registers.

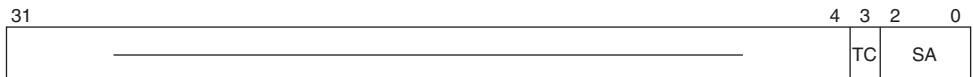
1. PTEH



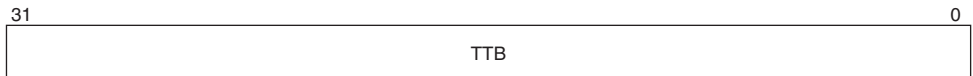
2. PTEL



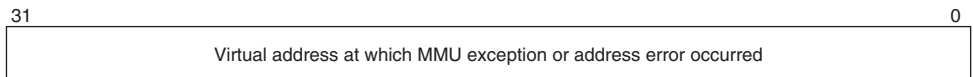
3. PTEA



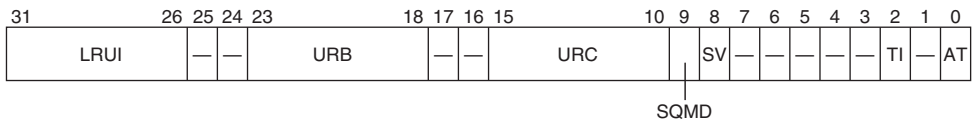
4. TTB



5. TEA



6. MMUCR



Note: — indicates a reserved bit: the write value must be 0, and a read will return 0.

Figure 3.2 MMU-Related Registers

1. Page table entry high register (PTEH): Longword access to PTEH can be performed from H'FF00 0000 in the P4 area and H'1F00 0000 in area 7. PTEH consists of the virtual page number (VPN) and address space identifier (ASID). When an MMU exception or address error exception occurs, the VPN of the virtual address at which the exception occurred is set in the VPN field by hardware. VPN varies according to the page size, but the VPN set by hardware when an exception occurs consists of the upper 22 bits of the virtual address which caused the exception. VPN setting can also be carried out by software. The number of the currently executing process is set in the ASID field by software. ASID is not updated by hardware. VPN and ASID are recorded in the UTLB by means of the LDLTB instruction. A branch to the P0, P3, or V0 area which uses the updated ASID after the ASID field in PTEH is rewritten should be made at least 6 instructions after the PTEH update instruction.

2. Page table entry low register (PTEL): Longword access to PTEL can be performed from H'FF00 0004 in the P4 area and H'1F00 0004 in area 7. PTEL is used to hold the physical page number and page management information to be recorded in the UTLB by means of the LDTLB instruction. The contents of this register are not changed unless a software directive is issued.

3. Page table entry assistance register (PTEA): Longword access to PTEA can be performed from H'FF00 0034 in the P4 area and H'1F00 0034 in area 7. PTEA is used to store assistance bits for PCMCIA access to the UTLB by means of the LDTLB instruction. When performing PCMCIA access with the MMU off, access is always performed using the values of the SA and TC bits in this register. Access to a PCMCIA interface area by the DMAC is always performed using the DMAC's CHCRn.SSAn, CHCRn.DSAn, CHCRn.STC, and CHCRn.DTC values. The contents of this register are not changed unless a software directive is issued.

4. Translation table base register (TTB): Longword access to TTB can be performed from H'FF00 0008 in the P4 area and H'1F00 0008 in area 7. TTB is used, for example, to hold the base address of the currently used page table. The contents of TTB are not changed unless a software directive is issued. This register can be freely used by software.

5. TLB exception address register (TEA): Longword access to TEA can be performed from H'FF00 000C in the P4 area and H'1F00 000C in area 7. After an MMU exception or address error exception occurs, the virtual address at which the exception occurred is set in TEA by hardware. The contents of this register can be changed by software.

6. MMU control register (MMUCR): MMUCR contains the following bits:

LRUI: Least recently used ITLB
URB: UTLB replace boundary
URC: UTLB replace counter
SQMD: Store queue mode bit
SV: Single virtual mode bit

TI: TLB invalidate
 AT: Address translation bit

Longword access to MMUCR can be performed from H'FF00 0010 in the P4 area and H'1F00 0010 in area 7. The individual bits perform MMU settings as shown below. Therefore, MMUCR rewriting should be performed by a program in the P1 or P2 area. After MMUCR is updated, an instruction that performs data access to the P0, P3, U0, or store queue area should be located at least four instructions after the MMUCR update instruction. Also, a branch instruction to the P0, P3, or U0 area should be located at least eight instructions after the MMUCR update instruction. MMUCR contents can be changed by software. The LRUI bits and URC bits may also be updated by hardware.

- LRUI: LRU bits that indicate the ITLB entry for which replacement is to be performed. The LRU (least recently used) method is used to decide the ITLB entry to be replaced in the event of an ITLB miss. The entry to be purged from the ITLB can be confirmed using the LRUI bits. LRUI is updated by means of the algorithm shown below. A dash in this table means that updating is not performed.

| | LRUI | | | | | |
|---------------------------|------|-----|-----|-----|-----|-----|
| | [5] | [4] | [3] | [2] | [1] | [0] |
| When ITLB entry 0 is used | 0 | 0 | 0 | — | — | — |
| When ITLB entry 1 is used | 1 | — | — | 0 | 0 | — |
| When ITLB entry 2 is used | — | 1 | — | 1 | — | 0 |
| When ITLB entry 3 is used | — | — | 1 | — | 1 | 1 |
| Other than the above | — | — | — | — | — | — |

When the LRUI bit settings are as shown below, the corresponding ITLB entry is updated by an ITLB miss. An asterisk in this table means “Don't care”.

| | LRUI | | | | | |
|-------------------------|--------------------|-----|-----|-----|-----|-----|
| | [5] | [4] | [3] | [2] | [1] | [0] |
| ITLB entry 0 is updated | 1 | 1 | 1 | * | * | * |
| ITLB entry 1 is updated | 0 | * | * | 1 | 1 | * |
| ITLB entry 2 is updated | * | 0 | * | 0 | * | 1 |
| ITLB entry 3 is updated | * | * | 0 | * | 0 | 0 |
| Other than the above | Setting prohibited | | | | | |

Ensure that values for which “Setting prohibited” is indicated in the above table are not set at the discretion of software. After a power-on or manual reset the LRUI bits are initialized to 0, and therefore a prohibited setting is never made by a hardware update.

- **URB:** Bits that indicate the UTLB entry boundary at which replacement is to be performed. Valid only when $URB > 0$.
- **URC:** Random counter for indicating the UTLB entry for which replacement is to be performed with an LDTLB instruction. URC is incremented each time the UTLB is accessed. When $URB > 0$, URC is reset to 0 when the condition $URC = URB$ occurs. Also note that, if a value is written to URC by software which results in the condition $URC > URB$, incrementing is first performed in excess of URB until $URC = H'3F$. URC is not incremented by an LDTLB instruction.
- **SQMD:** Store queue mode bit. Specifies the right of access to the store queues.
0: User/privileged access possible
1: Privileged access possible (address error exception in case of user access)
- **SV:** Bit that switches between single virtual memory mode and multiple virtual memory mode.
0: Multiple virtual memory mode
1: Single virtual memory mode
When this bit is changed, ensure that 1 is also written to the TI bit.
- **TI:** TLB invalidation bit. Writing 1 to this bit invalidates (clears to 0) all valid UTLB/ITLB bits. This bit always returns 0 when read.
- **AT:** Address translation enable bit. Specifies MMU enabling or disabling.
0: MMU disabled
1: MMU enabled
MMU exceptions are not generated when the AT bit is 0. In the case of software that does not use the MMU, therefore, the AT bit should be cleared to 0.

3.3 Address Space

3.3.1 Physical Address Space

The SH-4 supports a 32-bit physical address space, and can access a 4-Gbyte address space. When the MMUCR.AT bit is cleared to 0 and the MMU is disabled, the address space is this physical address space. The physical address space is divided into a number of areas, as shown in figure 3.3. The physical address space is permanently mapped onto 29-bit external memory space; this correspondence can be implemented by ignoring the upper 3 bits of the physical address space addresses. In privileged mode, the 4-Gbyte space from the P0 area to the P4 area can be accessed. In user mode, a 2-Gbyte space in the U0 area can be accessed. Accessing the P1 to P4 areas (except the store queue area) in user mode will cause an address error.

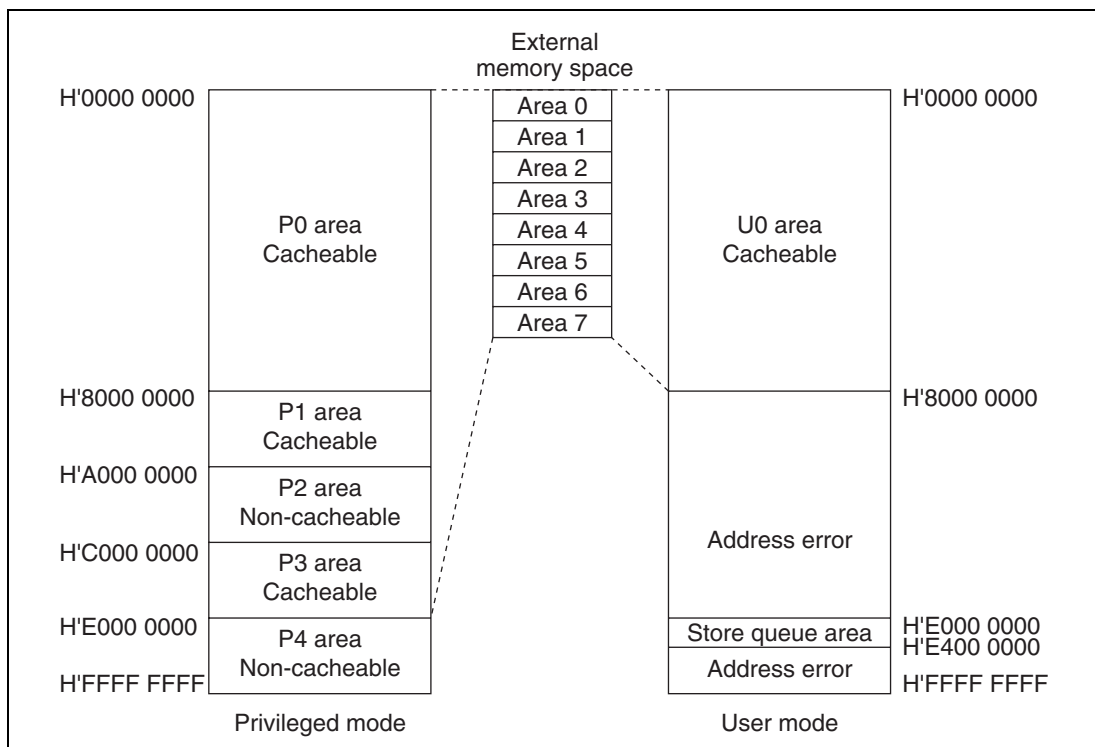


Figure 3.3 Physical Address Space (MMUCR.AT = 0)

When performing access from the CPU to a PCMCIA interface area in the SH-4, access is always performed using the values of the SA and TC bits set in the PTEA register. Access to a PCMCIA interface area by the DMAC is always performed using the DMAC's CHCRn.SSAn,

CHCRn.DSAn, CHCRn.STC, and CHCRn.DTC values. For details, see section 14, Direct Memory Access Controller (DMAC).

P0, P1, P3, U0 Areas: The P0, P1, P3, and U0 areas can be accessed using the cache. Whether or not the cache is used is determined by the cache control register (CCR). When the cache is used, with the exception of the P1 area, switching between the copy-back method and the write-through method for write accesses is specified by the CCR.WT bit. For the P1 area, switching is specified by the CCR.CB bit. Zeroizing the upper 3 bits of an address in these areas gives the corresponding external memory space address. However, since area 7 in the external memory space is a reserved area, a reserved area also appears in these areas.

P2 Area: The P2 area cannot be accessed using the cache. In the P2 area, zeroizing the upper 3 bits of an address gives the corresponding external memory space address. However, since area 7 in the external memory space is a reserved area, a reserved area also appears in this area.

P4 Area: The P4 area is mapped onto SH-4 on-chip I/O channels. This area cannot be accessed using the cache. The P4 area is shown in detail in figure 3.4.

| | |
|-------------|-------------------------------------|
| H'E000 0000 | Store queue |
| H'E400 0000 | |
| | Reserved area |
| H'F000 0000 | Instruction cache address array |
| H'F100 0000 | Instruction cache data array |
| H'F200 0000 | Instruction TLB address array |
| H'F300 0000 | Instruction TLB data arrays 1 and 2 |
| H'F400 0000 | Operand cache address array |
| H'F500 0000 | Operand cache data array |
| H'F600 0000 | Unified TLB address array |
| H'F700 0000 | Unified TLB data arrays 1 and 2 |
| H'F800 0000 | Reserved area |
| H'FC00 0000 | |
| | Control register area |
| H'FFFF FFFF | |

Figure 3.4 P4 Area

The area from H'E000 0000 to H'E3FF FFFF comprises addresses for accessing the store queues (SQs). When the MMU is disabled (MMUCR.AT = 0), the SQ access right is specified by the MMUCR.SQMD bit. For details, see section 4.7, Store Queues.

The area from H'F000 0000 to H'F0FF FFFF is used for direct access to the instruction cache address array. For details, see section 4.5.1, IC Address Array.

The area from H'F100 0000 to H'F1FF FFFF is used for direct access to the instruction cache data array. For details, see section 4.5.2, IC Data Array.

The area from H'F200 0000 to H'F2FF FFFF is used for direct access to the instruction TLB address array. For details, see section 3.7.1, ITLB Address Array.

The area from H'F300 0000 to H'F3FF FFFF is used for direct access to instruction TLB data arrays 1 and 2. For details, see sections 3.7.2, ITLB Data Array 1, and 3.7.3, ITLB Data Array 2.

The area from H'F400 0000 to H'F4FF FFFF is used for direct access to the operand cache address array. For details, see section 4.5.3, OC Address Array.

The area from H'F500 0000 to H'F5FF FFFF is used for direct access to the operand cache data array. For details, see section 4.5.4, OC Data Array.

The area from H'F600 0000 to H'F6FF FFFF is used for direct access to the unified TLB address array. For details, see section 3.7.4, UTLB Address Array.

The area from H'F700 0000 to H'F7FF FFFF is used for direct access to unified TLB data arrays 1 and 2. For details, see sections 3.7.5, UTLB Data Array 1, and 3.7.6, UTLB Data Array 2.

The area from H'FC00 0000 to H'FFFF FFFF is the on-chip peripheral module control register area. For details, see appendix A, Address List.

3.3.2 External Memory Space

The SH-4 supports a 29-bit external memory space. The external memory space is divided into eight areas as shown in figure 3.5. Areas 0 to 6 relate to memory, such as SRAM, synchronous DRAM, DRAM, and PCMCIA. Area 7 is a reserved area. For details, see section 13, Bus State Controller (BSC).

| | |
|-------------|------------------------|
| H'0000 0000 | Area 0 |
| H'0400 0000 | Area 1 |
| H'0800 0000 | Area 2 |
| H'0C00 0000 | Area 3 |
| H'1000 0000 | Area 4 |
| H'1400 0000 | Area 5 |
| H'1800 0000 | Area 6 |
| H'1C00 0000 | Area 7 (reserved area) |
| H'1FFF FFFF | |

Figure 3.5 External Memory Space

3.3.3 Virtual Address Space

Setting the MMUCR.AT bit to 1 enables the P0, P3, and U0 areas of the physical address space in the SH-4 to be mapped onto any external memory space in 1-, 4-, or 64-Kbyte, or 1-Mbyte, page units. By using an 8-bit address space identifier, the P0, U0, P3, and store queue areas can be increased to a maximum of 256. This is called the virtual address space. Mapping from virtual address space to 29-bit external memory space is carried out using the TLB. Only when area 7 in external memory space is accessed using virtual address space, addresses H'1C00 0000 to H'1FFF FFFF of area 7 are not designated as a reserved area, but are equivalent to the P4 area control register area in the physical address space. Virtual address space is illustrated in figure 3.6.

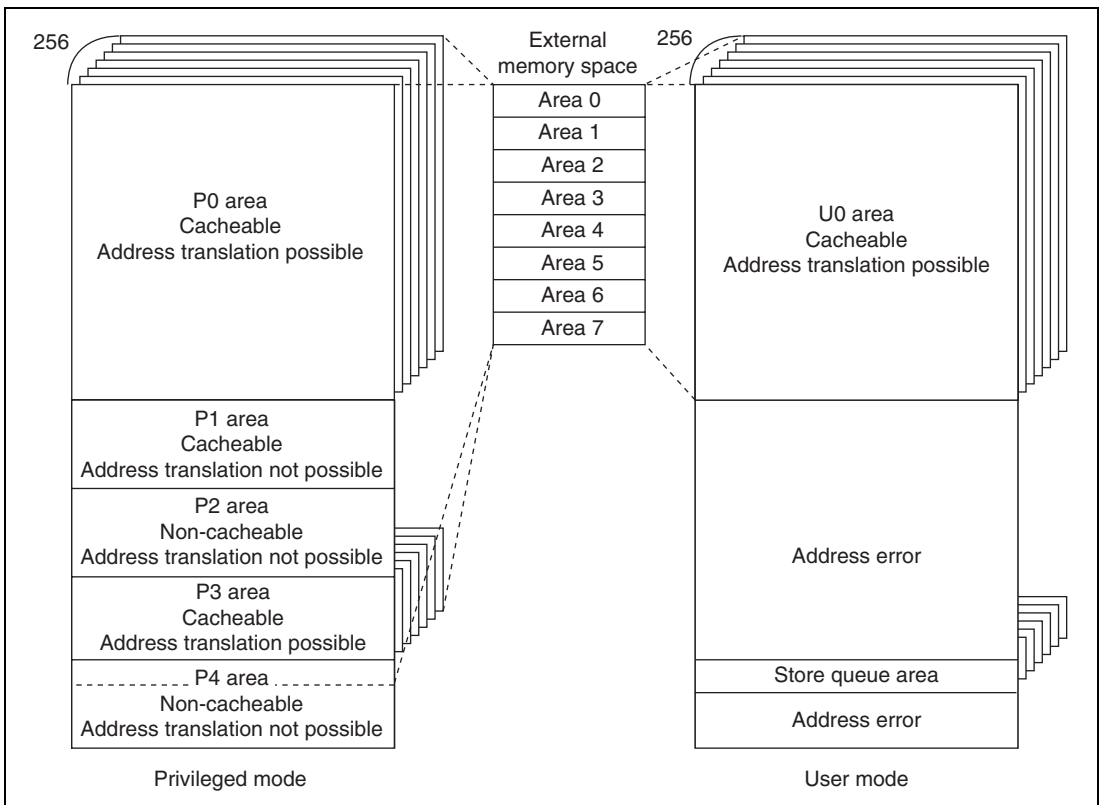


Figure 3.6 Virtual Address Space (MMUCR.AT = 1)

In the state of cache enabling, when the areas of P0, P3, and U0 are mapped onto the area of the PCMCIA interface by means of the TLB, it is necessary either to specify 1 for the WT bit or to specify 0 for the C bit on that page. At that time, the regions are accessed by the values of SA and TC set in page units of the TLB.

Here, access to an area of the PCMCIA interface by accessing an area of P1, P2, or P4 from the CPU is disabled. In addition, the PCMCIA interface is always accessed by the DMAC with the values of CHCRn, SSAn, CHCRn.DsAn, CHCRn.STC and CHCRn.DTC in the DMAC. For details, see Section 14, Direct Memory Access Controller (DMAC).

P0, P3, U0 Areas: The P0 area (excluding addresses H'7C00 0000 to H'7FFF FFFF), P3 area, and U0 area (excluding addresses H'7C00 0000 to H'7FFF FFFF) allow access using the cache and address translation using the TLB. These areas can be mapped onto any external memory space in 1-, 4-, or 64-Kbyte, or 1-Mbyte, page units. When CCR is in the cache-enabled state and the cacheability bit (C bit) in the TLB is 1, accesses can be performed using the cache. In write accesses to the cache, switching between the copy-back method and the write-through method is indicated by the TLB write-through bit (WT bit), and is specified in page units.

Only when the P0, P3, and U0 areas are mapped onto external memory space by means of the TLB, addresses H'1C00 0000 to H'1FFF FFFF of area 7 in external memory space are allocated to the control register area. This enables control registers to be accessed from the U0 area in user mode. In this case, the C bit for the corresponding page must be cleared to 0.

P1, P2, P4 Areas: Address translation using the TLB cannot be performed for the P1, P2, or P4 area (except for the store queue area). Accesses to these areas are the same as for physical address space. The store queue area can be mapped onto any external memory space by the MMU. However, operation in the case of an exception differs from that for normal P0, U0, and P3 spaces. For details, see section 4.7, Store Queues.

3.3.4 On-Chip RAM Space

In the SH-4, half of the operand cache can be used as on-chip RAM. This can be done by changing the CCR settings.

When the operand cache is used as on-chip RAM (CCR.ORA = 1), P0, U0 area addresses H'7C00 0000 to H'7FFF FFFF are an on-chip RAM area. Data accesses (byte/word/longword/quadword) can be used in this area. This area can only be used in RAM mode.

3.3.5 Address Translation

When the MMU is used, the virtual address space is divided into units called pages, and translation to physical addresses is carried out in these page units. The address translation table in external memory contains the physical addresses corresponding to virtual addresses and additional information such as memory protection codes. Fast address translation is achieved by caching the contents of the address translation table located in external memory into the TLB. In the SH-4, basically, the ITLB is used for instruction accesses and the UTLB for data accesses. In the event

of an access to an area other than the P4 area, the accessed virtual address is translated to a physical address. If the virtual address belongs to the P1 or P2 area, the physical address is uniquely determined without accessing the TLB. If the virtual address belongs to the P0, U0, or P3 area, the TLB is searched using the virtual address, and if the virtual address is recorded in the TLB, a TLB hit is made and the corresponding physical address is read from the TLB. If the accessed virtual address is not recorded in the TLB, a TLB miss exception is generated and processing switches to the TLB miss exception handling routine. In the TLB miss exception handling routine, the address translation table in external memory is searched, and the corresponding physical address and page management information are recorded in the TLB. After the return from the exception handling routine, the instruction which caused the TLB miss exception is re-executed.

3.3.6 Single Virtual Memory Mode and Multiple Virtual Memory Mode

There are two virtual memory systems, single virtual memory and multiple virtual memory, either of which can be selected with the MMUCR.SV bit. In the single virtual memory system, a number of processes run simultaneously, using virtual address space on an exclusive basis, and the physical address corresponding to a particular virtual address is uniquely determined. In the multiple virtual memory system, a number of processes run while sharing the virtual address space, and a particular virtual address may be translated into different physical addresses depending on the process. The only difference between the single virtual memory and multiple virtual memory systems in terms of operation is in the TLB address comparison method (see section 3.4.3, Address Translation Method).

3.3.7 Address Space Identifier (ASID)

In multiple virtual memory mode, the 8-bit address space identifier (ASID) is used to distinguish between processes running simultaneously while sharing the virtual address space. Software can set the ASID of the currently executing process in PTEH in the MMU. The TLB does not have to be purged when processes are switched by means of ASID.

In single virtual memory mode, ASID is used to provide memory protection for processes running simultaneously while using the virtual memory space on an exclusive basis.

- Notes:
1. In single virtual memory mode of the SH-4, entries with the same virtual page number (VPN) but different ASIDs cannot be set in the TLB simultaneously.
 2. When the SH7751 is operating in single virtual memory mode and user mode, the LSI may hang during hardware ITLB miss handling (see section 3.5.4, Hardware ITLB Miss Handling), or an ITLB multiple hit exception may occur, if an ITLB miss occurs and the UTLB contains address translation information including an ITLB miss address

with a different ASID and unshared state (SH bit is 0). To avoid this, use workaround (1) or (2) below.

- (1) Purge the UTLB when switching the ASID values (PTEH and ASID) of the current processing.
- (2) Manage the behavior of program instruction addresses in user mode so that no instruction is executed in an address area (including overrun prefetch of an instruction) that is registered in the UTLB with a different ASID and unshared address translation information. Note that accessing a different ASID in single virtual memory mode can only be used to trigger an exception during data access.

3.4 TLB Functions

3.4.1 Unified TLB (UTLB) Configuration

The unified TLB (UTLB) is so called because of its use for the following two purposes:

1. To translate a virtual address to a physical address in a data access
2. As a table of address translation information to be recorded in the instruction TLB in the event of an ITLB miss

Information in the address translation table located in external memory is cached into the UTLB. The address translation table contains virtual page numbers and address space identifiers, and corresponding physical page numbers and page management information. Figure 3.7 shows the overall configuration of the UTLB. The UTLB consists of 64 fully-associative type entries. Figure 3.8 shows the relationship between the address format and page size.

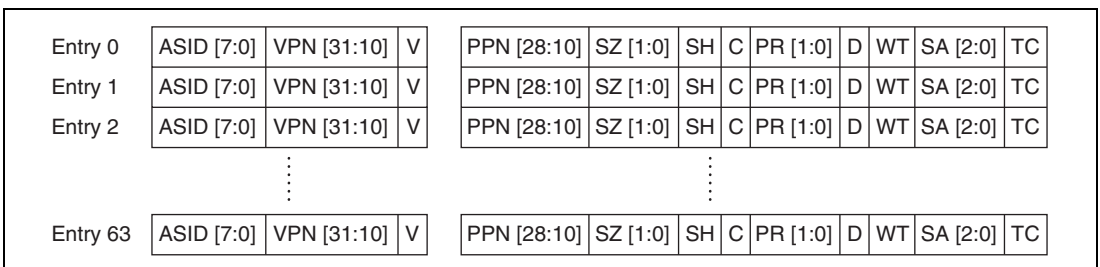


Figure 3.7 UTLB Configuration

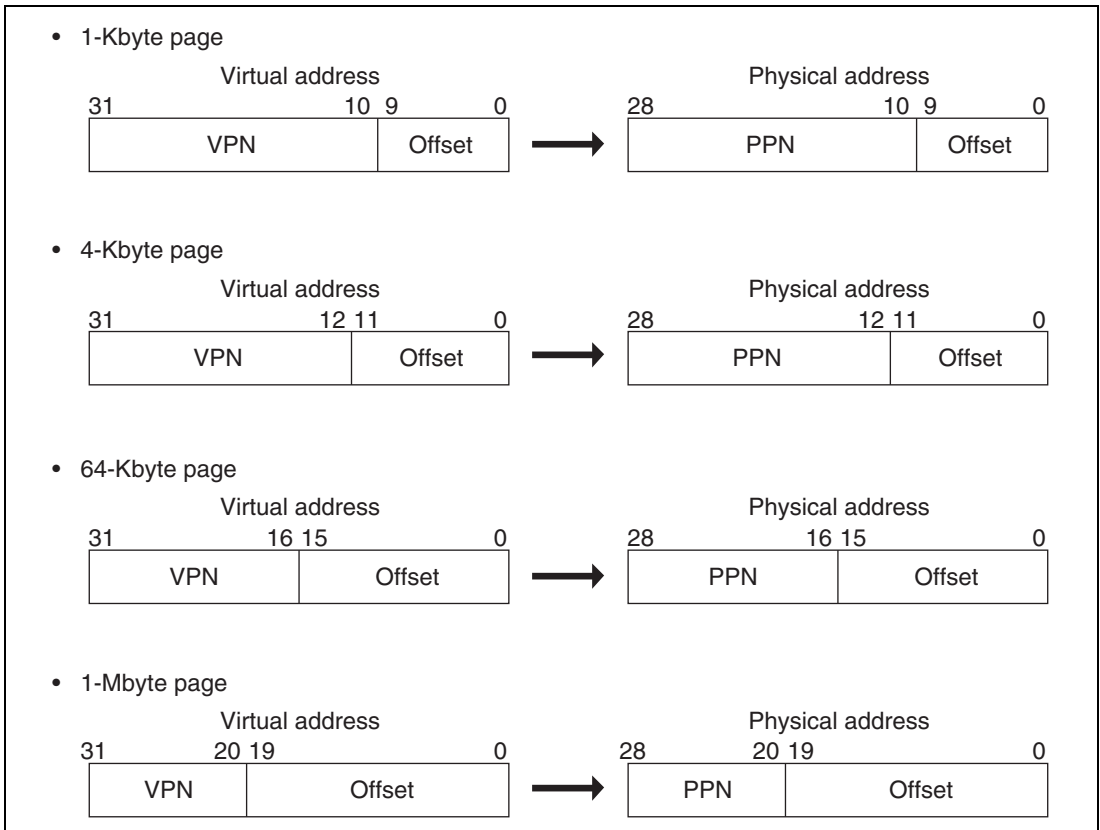


Figure 3.8 Relationship between Page Size and Address Format

- VPN: Virtual page number
 - For 1-Kbyte page: upper 22 bits of virtual address
 - For 4-Kbyte page: upper 20 bits of virtual address
 - For 64-Kbyte page: upper 16 bits of virtual address
 - For 1-Mbyte page: upper 12 bits of virtual address
- ASID: Address space identifier

Indicates the process that can access a virtual page.

In single virtual memory mode and user mode, or in multiple virtual memory mode, if the SH bit is 0, this identifier is compared with the ASID in PTEH when address comparison is performed.

- **SH: Share status bit**
When 0, pages are not shared by processes.
When 1, pages are shared by processes.
- **SZ: Page size bits**
Specify the page size.
00: 1-Kbyte page
01: 4-Kbyte page
10: 64-Kbyte page
11: 1-Mbyte page
- **V: Validity bit**
Indicates whether the entry is valid.
0: Invalid
1: Valid
Cleared to 0 by a power-on reset.
Not affected by a manual reset.
- **PPN: Physical page number**
Upper 22 bits of the physical address.
With a 1-Kbyte page, PPN bits [28:10] are valid.
With a 4-Kbyte page, PPN bits [28:12] are valid.
With a 64-Kbyte page, PPN bits [28:16] are valid.
With a 1-Mbyte page, PPN bits [28:20] are valid.
The synonym problem must be taken into account when setting the PPN (see section 3.5.5, Avoiding Synonym Problems).
- **PR: Protection key data**
2-bit data expressing the page access right as a code.
00: Can be read only, in privileged mode
01: Can be read and written in privileged mode
10: Can be read only, in privileged or user mode
11: Can be read and written in privileged mode or user mode

- **C: Cacheability bit**
Indicates whether a page is cacheable.
0: Not cacheable
1: Cacheable
When control register space is mapped, this bit must be cleared to 0.
When performing PCMCIA space mapping in the cache enabled state, either clear this bit to 0 or set the WT bit to 1.
- **D: Dirty bit**
Indicates whether a write has been performed to a page.
0: Write has not been performed
1: Write has been performed
- **WT: Write-through bit**
Specifies the cache write mode.
0: Copy-back mode
1: Write-through mode
When performing PCMCIA space mapping in the cache enabled state, either set this bit to 1 or clear the C bit to 0.
- **SA: Space attribute bits**
Valid only when the page is mapped onto PCMCIA connected to area 5 or 6.
000: Undefined
001: Variable-size I/O space (base size according to $\overline{\text{IOIS16}}$ signal)
010: 8-bit I/O space
011: 16-bit I/O space
100: 8-bit common memory space
101: 16-bit common memory space
110: 8-bit attribute memory space
111: 16-bit attribute memory space
- **TC: Timing control bit**
Used to select wait control register bits in the bus control unit for areas 5 and 6.
0: WCR2 (A5W2–A5W0) and PCR (A5PCW1–A5PCW0, A5TED2–A5TED0, A5TEH2–A5TEH0) are used
1: WCR2 (A6W2–A6W0) and PCR (A6PCW1–A6PCW0, A6TED2–A6TED0, A6TEH2–A6TEH0) are used

3.4.2 Instruction TLB (ITLB) Configuration

The ITLB is used to translate a virtual address to a physical address in an instruction access. Information in the address translation table located in the UTLB is cached into the ITLB. Figure 3.9 shows the overall configuration of the ITLB. The ITLB consists of 4 fully-associative type entries. The address translation information is almost the same as that in the UTLB, but with the following differences:

1. D and WT bits are not supported.
2. There is only one PR bit, corresponding to the upper of the PR bits in the UTLB.

| | | | | | | | | | | |
|---------|------------|-------------|---|-------------|----------|----|---|----|----------|----|
| Entry 0 | ASID [7:0] | VPN [31:10] | V | PPN [28:10] | SZ [1:0] | SH | C | PR | SA [2:0] | TC |
| Entry 1 | ASID [7:0] | VPN [31:10] | V | PPN [28:10] | SZ [1:0] | SH | C | PR | SA [2:0] | TC |
| Entry 2 | ASID [7:0] | VPN [31:10] | V | PPN [28:10] | SZ [1:0] | SH | C | PR | SA [2:0] | TC |
| Entry 3 | ASID [7:0] | VPN [31:10] | V | PPN [28:10] | SZ [1:0] | SH | C | PR | SA [2:0] | TC |

Figure 3.9 ITLB Configuration

3.4.3 Address Translation Method

Figures 3.10 and 3.11 show flowcharts of memory accesses using the UTLB and ITLB.

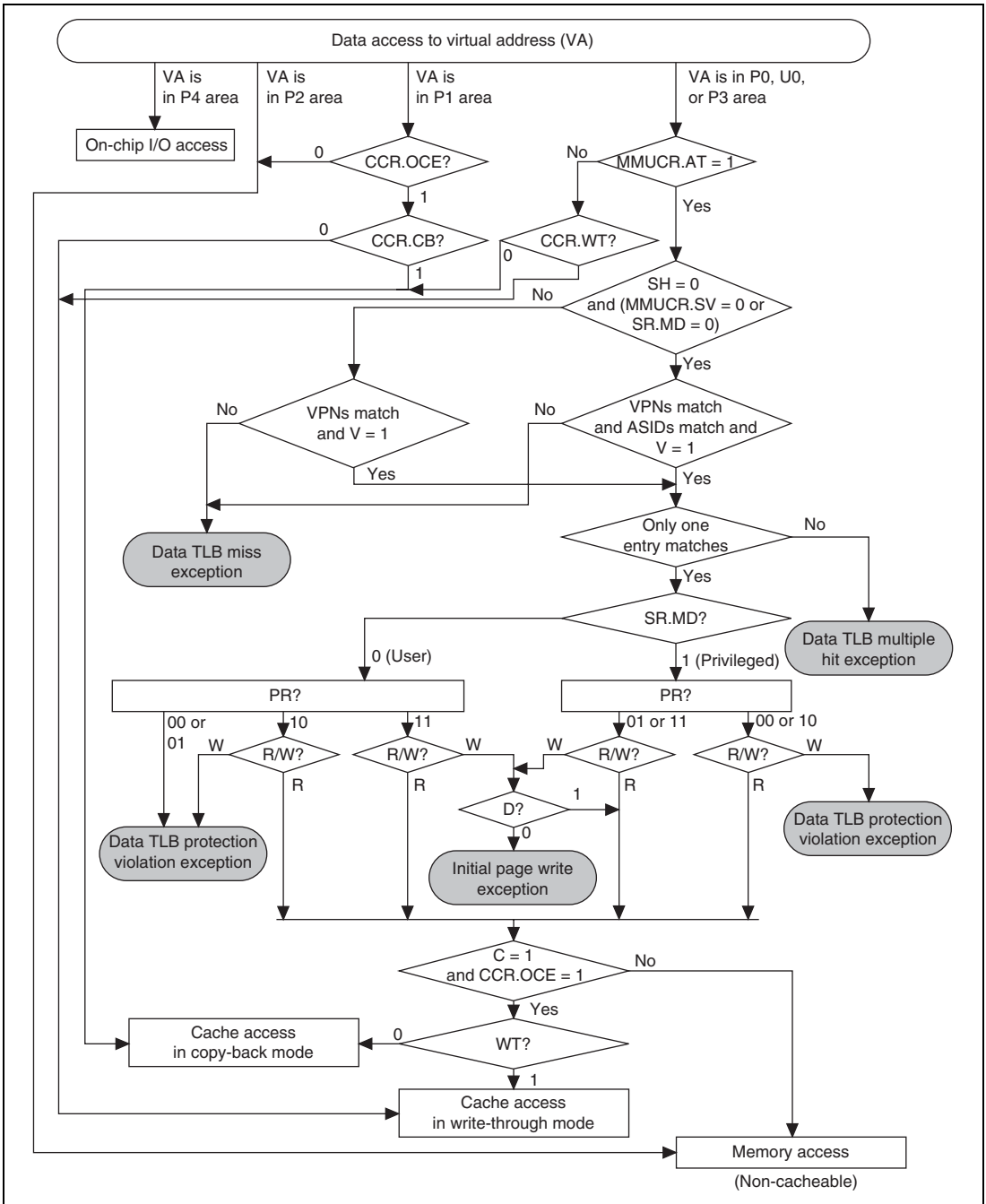


Figure 3.10 Flowchart of Memory Access Using UTLB

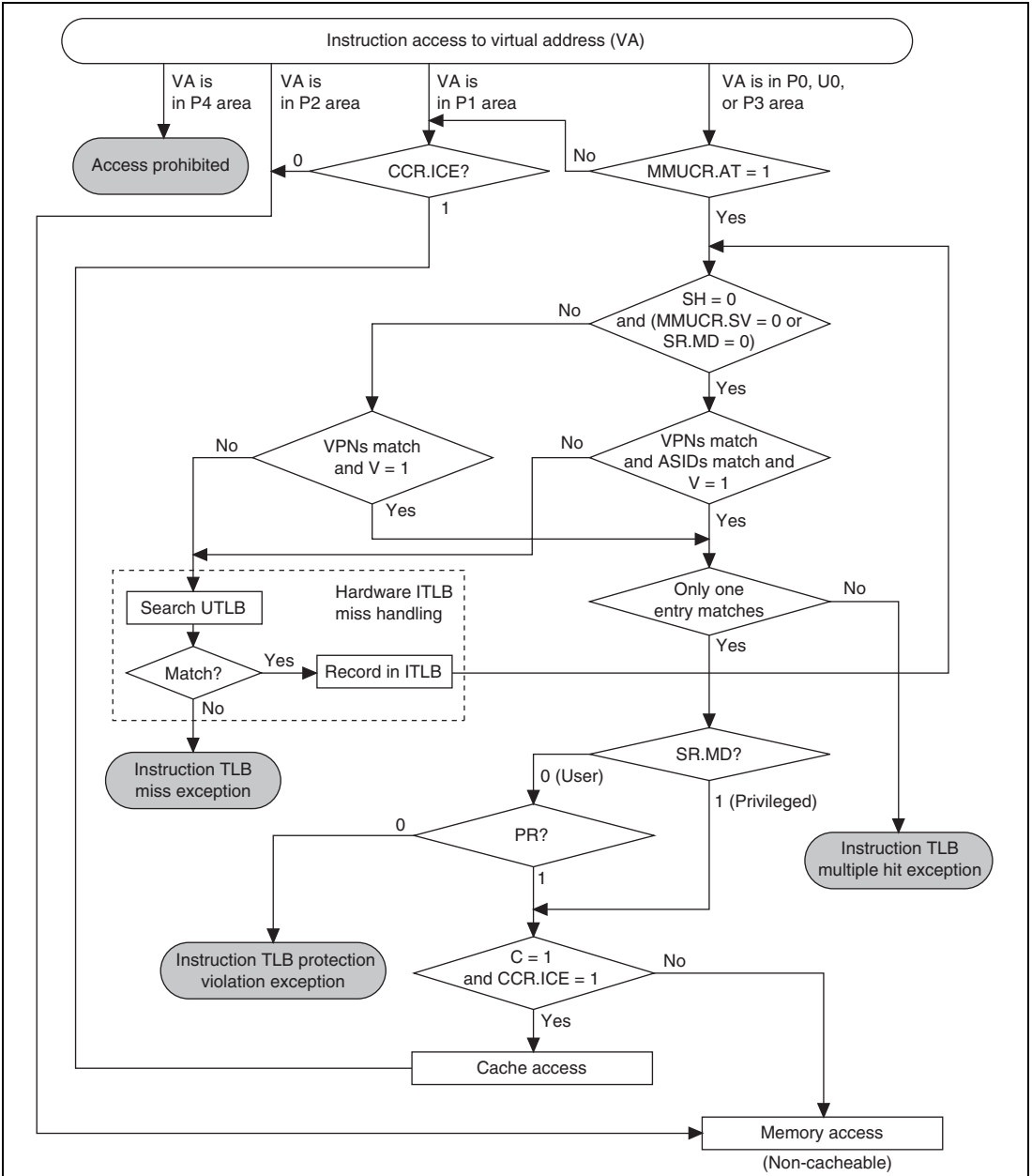


Figure 3.11 Flowchart of Memory Access Using ITLB

3.5 MMU Functions

3.5.1 MMU Hardware Management

The SH-4 supports the following MMU functions.

1. The MMU decodes the virtual address to be accessed by software, and performs address translation by controlling the UTLB/ITLB in accordance with the MMUCR settings.
2. The MMU determines the cache access status on the basis of the page management information read during address translation (C, WT, SA, and TC bits).
3. If address translation cannot be performed normally in a data access or instruction access, the MMU notifies software by means of an MMU exception.
4. If address translation information is not recorded in the ITLB in an instruction access, the MMU searches the UTLB, and if the necessary address translation information is recorded in the UTLB, the MMU copies this information into the ITLB in accordance with MMUCR.LRUI.

3.5.2 MMU Software Management

Software processing for the MMU consists of the following:

1. Setting of MMU-related registers. Some registers are also partially updated by hardware automatically.
2. Recording, deletion, and reading of TLB entries. There are two methods of recording UTLB entries: by using the LDTLB instruction, or by writing directly to the memory-mapped UTLB. ITLB entries can only be recorded by writing directly to the memory-mapped ITLB. For deleting or reading UTLB/ITLB entries, it is possible to access the memory-mapped UTLB/ITLB.
3. MMU exception handling. When an MMU exception occurs, processing is performed based on information set by hardware.

3.5.3 MMU Instruction (LDTLB)

A TLB load instruction (LDTLB) is provided for recording UTLB entries. When an LDTLB instruction is issued, the SH-4 copies the contents of PTEH, PTEL, and PTEA to the UTLB entry indicated by MMUCR.URC. ITLB entries are not updated by the LDTLB instruction, and therefore address translation information purged from the UTLB entry may still remain in the ITLB entry. As the LDTLB instruction changes address translation information, ensure that it is

issued by a program in the P1 or P2 area. The operation of the LDTLB instruction is shown in figure 3.12.

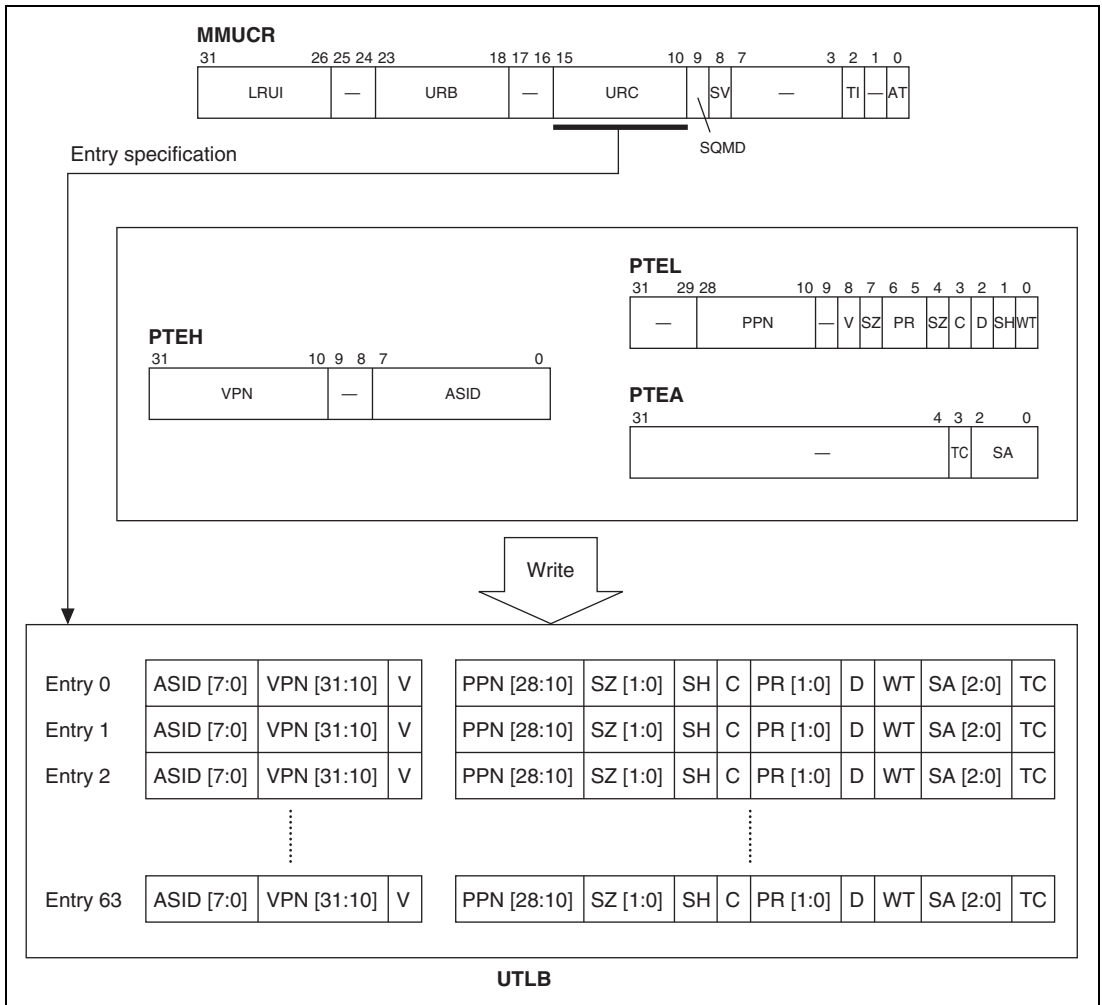


Figure 3.12 Operation of LDTLB Instruction

3.5.4 Hardware ITLB Miss Handling

In an instruction access, the SH-4 searches the ITLB. If it cannot find the necessary address translation information (i.e. in the event of an ITLB miss), the UTLB is searched by hardware, and if the necessary address translation information is present, it is recorded in the ITLB. This procedure is known as hardware ITLB miss handling. If the necessary address translation

information is not found in the UTLB search, an instruction TLB miss exception is generated and processing passes to software.

3.5.5 Avoiding Synonym Problems

When 1- or 4-Kbyte pages are recorded in TLB entries, a synonym problem may arise. The problem is that, when a number of virtual addresses are mapped onto a single physical address, the same physical address data is recorded in a number of cache entries, and it becomes impossible to guarantee data integrity. This problem does not occur with the instruction TLB or instruction cache. In the SH-4, entry specification is performed using bits [13:5] of the virtual address in order to achieve fast operand cache operation. However, bits [13:10] of the virtual address in the case of a 1-Kbyte page, and bits [13:12] of the virtual address in the case of a 4-Kbyte page, are subject to address translation. As a result, bits [13:10] of the physical address after translation may differ from bits [13:10] of the virtual address.

Consequently, the following restrictions apply to the recording of address translation information in UTLB entries.

1. When address translation information whereby a number of 1-Kbyte page UTLB entries are translated into the same physical address is recorded in the UTLB, ensure that the VPN [13:10] values are the same.
2. When address translation information whereby a number of 4-Kbyte page UTLB entries are translated into the same physical address is recorded in the UTLB, ensure that the VPN [13:12] values are the same.
3. Do not use 1-Kbyte page UTLB entry physical addresses with UTLB entries of a different page size.
4. Do not use 4-Kbyte page UTLB entry physical addresses with UTLB entries of a different page size.

The above restrictions apply only when performing accesses using the cache. When cache index mode is used, VPN [25] is used for the entry address instead of VPN [13], and therefore the above restrictions apply to VPN [25].

Note: When multiple items of address translation information use the same physical memory to provide for future SuperH RISC engine family expansion, ensure that the VPN [20:10] values are the same. Also, do not use the same physical address for address translation information of different page sizes.

3.6 MMU Exceptions

There are seven MMU exceptions: the instruction TLB multiple hit exception, instruction TLB miss exception, instruction TLB protection violation exception, data TLB multiple hit exception, data TLB miss exception, data TLB protection violation exception, and initial page write exception. Refer to figures 3.10 and 3.11 for the conditions under which each of these exceptions occurs.

3.6.1 Instruction TLB Multiple Hit Exception

An instruction TLB multiple hit exception occurs when more than one ITLB entry matches the virtual address to which an instruction access has been made. If multiple hits occur when the UTLB is searched by hardware in hardware ITLB miss handling, a data TLB multiple hit exception will result.

When an instruction TLB multiple hit exception occurs a reset is executed, and cache coherency is not guaranteed.

Hardware Processing: In the event of an instruction TLB multiple hit exception, hardware carries out the following processing:

1. Sets the virtual address at which the exception occurred in TEA.
2. Sets exception code H'140 in EXPEVT.
3. Branches to the reset handling routine (H'A000 0000).

Software Processing (Reset Routine): The ITLB entries which caused the multiple hit exception are checked in the reset handling routine. This exception is intended for use in program debugging, and should not normally be generated.

3.6.2 Instruction TLB Miss Exception

An instruction TLB miss exception occurs when address translation information for the virtual address to which an instruction access is made is not found in the UTLB entries by the hardware ITLB miss handling procedure. The instruction TLB miss exception processing carried out by hardware and software is shown below. This is the same as the processing for a data TLB miss exception.

Hardware Processing: In the event of an instruction TLB miss exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'040 in EXPEVT.
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR. The R15 contents at this time are saved in SGR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0400 to the contents of VBR, and starts the instruction TLB miss exception handling routine.

Software Processing (Instruction TLB Miss Exception Handling Routine): Software is responsible for searching the external memory page table and assigning the necessary page table entry. Software should carry out the following processing in order to find and assign the necessary page table entry.

1. Write to PTEL the values of the PPN, PR, SZ, C, D, SH, V, and WT bits in the page table entry recorded in the external memory address translation table. If necessary, the values of the SA and TC bits should be written to PTEA.
2. When the entry to be replaced in entry replacement is specified by software, write that value to URC in the MMUCR register. If URC is greater than URB at this time, the value should be changed to an appropriate value after issuing an LDTLB instruction.
3. Execute the LDTLB instruction and write the contents of PTEH, PTEL, and PTEA to the TLB.
4. Finally, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.3 Instruction TLB Protection Violation Exception

An instruction TLB protection violation exception occurs when, even though an ITLB entry contains address translation information matching the virtual address to which an instruction access is made, the actual access type is not permitted by the access right specified by the PR bit.

The instruction TLB protection violation exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of an instruction TLB protection violation exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'0A0 in EXPEVT.
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR. Set the current R15 value in SGR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0100 to the contents of VBR, and starts the instruction TLB protection violation exception handling routine.

Software Processing (Instruction TLB Protection Violation Exception Handling Routine):

Resolve the instruction TLB protection violation, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.4 Data TLB Multiple Hit Exception

A data TLB multiple hit exception occurs when more than one UTLB entry matches the virtual address to which a data access has been made. A data TLB multiple hit exception is also generated if multiple hits occur when the UTLB is searched in hardware ITLB miss handling.

When a data TLB multiple hit exception occurs a reset is executed, and cache coherency is not guaranteed. The contents of PPN in the UTLB prior to the exception may also be corrupted.

Hardware Processing: In the event of a data TLB multiple hit exception, hardware carries out the following processing:

1. Sets the virtual address at which the exception occurred in TEA.
2. Sets exception code H'140 in EXPEVT.
3. Branches to the reset handling routine (H'A000 0000).

Software Processing (Reset Routine): The UTLB entries which caused the multiple hit exception are checked in the reset handling routine. This exception is intended for use in program debugging, and should not normally be generated.

3.6.5 Data TLB Miss Exception

A data TLB miss exception occurs when address translation information for the virtual address to which a data access is made is not found in the UTLB entries. The data TLB miss exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of a data TLB miss exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'040 in the case of a read, or H'060 in the case of a write, in EXPEVT (OCBP, OCBWB: read; OCBI, MOVCA.L: write).
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR, and sets the R15 contents at the time in SGR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0400 to the contents of VBR, and starts the data TLB miss exception handling routine.

Software Processing (Data TLB Miss Exception Handling Routine): Software is responsible for searching the external memory page table and assigning the necessary page table entry. Software should carry out the following processing in order to find and assign the necessary page table entry.

1. Write to PTEL the values of the PPN, PR, SZ, C, D, SH, V, and WT bits in the page table entry recorded in the external memory address translation table. If necessary, the values of the SA and TC bits should be written to PTEA.
2. When the entry to be replaced in entry replacement is specified by software, write that value to URC in the MMUCR register. If URC is greater than URB at this time, the value should be changed to an appropriate value after issuing an LDTLB instruction.

3. Execute the LDTLB instruction and write the contents of PTEH, PTEL, and PTEA to the UTLB.
4. Finally, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.6 Data TLB Protection Violation Exception

A data TLB protection violation exception occurs when, even though a UTLB entry contains address translation information matching the virtual address to which a data access is made, the actual access type is not permitted by the access right specified by the PR bit. The data TLB protection violation exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of a data TLB protection violation exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'0A0 in the case of a read, or H'0C0 in the case of a write, in EXPEVT (OCBP, OCBWB: read; OCBI, MOVCA.L: write).
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR. The R15 contents at this time are saved in SGR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0100 to the contents of VBR, and starts the data TLB protection violation exception handling routine.

Software Processing (Data TLB Protection Violation Exception Handling Routine): Resolve the data TLB protection violation, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.7 Initial Page Write Exception

An initial page write exception occurs when the D bit is 0 even though a UTLB entry contains address translation information matching the virtual address to which a data access (write) is made, and the access is permitted. The initial page write exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of an initial page write exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'080 in EXPEVT.
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR. The R15 contents at this time are saved in SGR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0100 to the contents of VBR, and starts the initial page write exception handling routine.

Software Processing (Initial Page Write Exception Handling Routine): The following processing should be carried out as the responsibility of software:

1. Retrieve the necessary page table entry from external memory.
2. Write 1 to the D bit in the external memory page table entry.
3. Write to PTEL the values of the PPN, PR, SZ, C, D, WT, SH, and V bits in the page table entry recorded in external memory. If necessary, the values of the SA and TC bits should be written to PTEA.
4. When the entry to be replaced in entry replacement is specified by software, write that value to URC in the MMUCR register. If URC is greater than URB at this time, the value should be changed to an appropriate value after issuing an LDTLB instruction.
5. Execute the LDTLB instruction and write the contents of PTEH, PTEL, and PTEA to the UTLB.

6. Finally, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.7 Memory-Mapped TLB Configuration

To enable the ITLB and UTLB to be managed by software, their contents can be read and written by a P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in the other area. A branch to an area other than the P2 area should be made at least 8 instructions after this MOV instruction. The ITLB and UTLB are allocated to the P4 area in physical address space. VPN, V, and ASID in the ITLB can be accessed as an address array, PPN, V, SZ, PR, C, and SH as data array 1, and SA and TC as data array 2. VPN, D, V, and ASID in the UTLB can be accessed as an address array, PPN, V, SZ, PR, C, D, WT, and SH as data array 1, and SA and TC as data array 2. V and D can be accessed from both the address array side and the data array side. Only longword access is possible. Instruction fetches cannot be performed in these areas. For reserved bits, a write value of 0 should be specified; their read value is undefined.

3.7.1 ITLB Address Array

The ITLB address array is allocated to addresses H'F200 0000 to H'F2FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and VPN, V, and ASID to be written to the address array are specified in the data field.

In the address field, bits [31:24] have the value H'F2 indicating the ITLB address array, and the entry is selected by bits [9:8]. As longword access is used, 0 should be specified for address field bits [1:0].

In the data field, VPN is indicated by bits [31:10], V by bit [8], and ASID by bits [7:0].

The following two kinds of operation can be used on the ITLB address array:

1. ITLB address array read
VPN, V, and ASID are read into the data field from the ITLB entry corresponding to the entry set in the address field.
2. ITLB address array write
VPN, V, and ASID specified in the data field are written to the ITLB entry corresponding to the entry set in the address field.

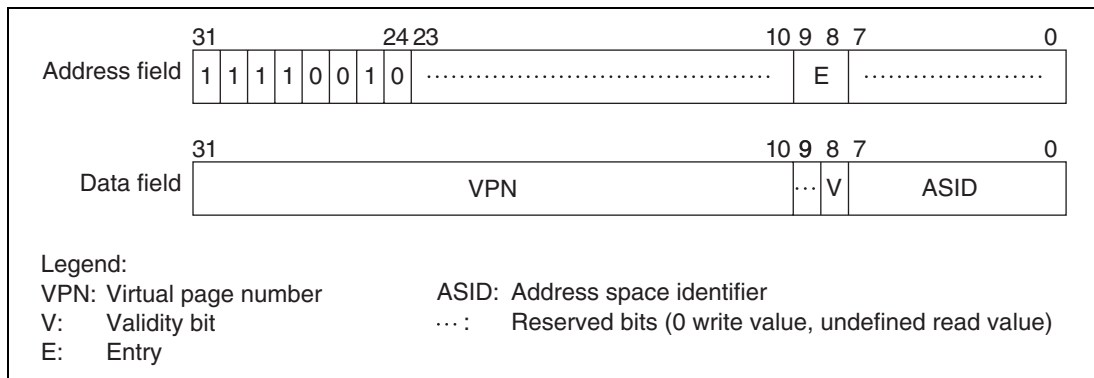


Figure 3.13 Memory-Mapped ITLB Address Array

3.7.2 ITLB Data Array 1

ITLB data array 1 is allocated to addresses H'F300 0000 to H'F37F FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and PPN, V, SZ, PR, C, and SH to be written to the data array are specified in the data field.

In the address field, bits [31:23] have the value H'F30 indicating ITLB data array 1, and the entry is selected by bits [9:8].

In the data field, PPN is indicated by bits [28:10], V by bit [8], SZ by bits [7] and [4], PR by bit [6], C by bit [3], and SH by bit [1].

The following two kinds of operation can be used on ITLB data array 1:

1. ITLB data array 1 read

PPN, V, SZ, PR, C, and SH are read into the data field from the ITLB entry corresponding to the entry set in the address field.

2. ITLB data array 1 write

PPN, V, SZ, PR, C, and SH specified in the data field are written to the ITLB entry corresponding to the entry set in the address field.

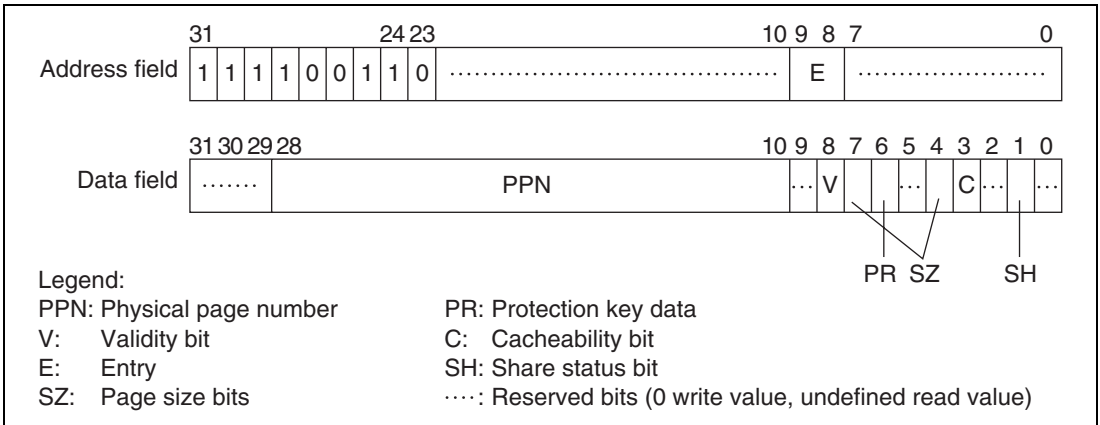


Figure 3.14 Memory-Mapped ITLB Data Array 1

3.7.3 ITLB Data Array 2

ITLB data array 2 is allocated to addresses H'F380 0000 to H'F3FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and SA and TC to be written to data array 2 are specified in the data field.

In the address field, bits [31:23] have the value H'F38 indicating ITLB data array 2, and the entry is selected by bits [9:8].

In the data field, SA is indicated by bits [2:0], and TC by bit [3].

The following two kinds of operation can be used on ITLB data array 2:

1. ITLB data array 2 read
SA and TC are read into the data field from the ITLB entry corresponding to the entry set in the address field.
2. ITLB data array 2 write
SA and TC specified in the data field are written to the ITLB entry corresponding to the entry set in the address field.

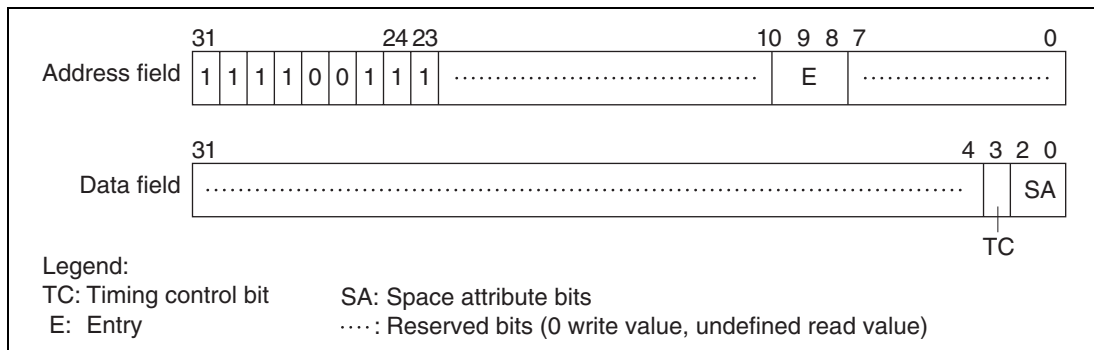


Figure 3.15 Memory-Mapped ITLB Data Array 2

3.7.4 UTLB Address Array

The UTLB address array is allocated to addresses H'F600 0000 to H'F6FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and VPN, D, V, and ASID to be written to the address array are specified in the data field.

In the address field, bits [31:24] have the value H'F6 indicating the UTLB address array, and the entry is selected by bits [13:8]. The address array bit [7] association bit (A bit) specifies whether or not address comparison is performed when writing to the UTLB address array.

In the data field, VPN is indicated by bits [31:10], D by bit [9], V by bit [8], and ASID by bits [7:0].

The following three kinds of operation can be used on the UTLB address array:

1. UTLB address array read
VPN, D, V, and ASID are read into the data field from the UTLB entry corresponding to the entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.
2. UTLB address array write (non-associative)
VPN, D, V, and ASID specified in the data field are written to the UTLB entry corresponding to the entry set in the address field. The A bit in the address field should be cleared to 0.
3. UTLB address array write (associative)

When a write is performed with the A bit in the address field set to 1, comparison of all the UTLB entries is carried out using the VPN specified in the data field and PTEH.ASID. The usual address comparison rules are followed, but if a UTLB miss occurs, the result is no operation, and an exception is not generated. If the comparison identifies a UTLB entry corresponding to the VPN specified in the data field, D and V specified in the data field are written to that entry. If there is more than one matching entry, a data TLB multiple hit exception results. This associative operation is simultaneously carried out on the ITLB, and if a matching entry is found in the ITLB, V is written to that entry. Even if the UTLB comparison results in no operation, a write to the ITLB side only is performed as long as there is an ITLB match. If there is a match in both the UTLB and ITLB, the UTLB information is also written to the ITLB.

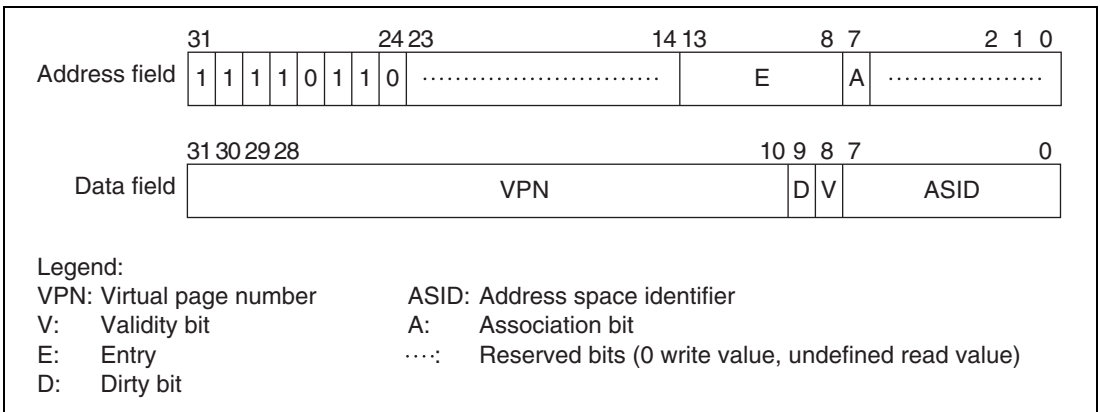


Figure 3.16 Memory-Mapped UTLB Address Array

3.7.5 UTLB Data Array 1

UTLB data array 1 is allocated to addresses H'F700 0000 to H'F77F FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and PPN, V, SZ, PR, C, D, SH, and WT to be written to the data array are specified in the data field.

In the address field, bits [31:23] have the value H'F70 indicating UTLB data array 1, and the entry is selected by bits [13:8].

In the data field, PPN is indicated by bits [28:10], V by bit [8], SZ by bits [7] and [4], PR by bits [6:5], C by bit [3], D by bit [2], SH by bit [1], and WT by bit [0].

The following two kinds of operation can be used on UTLB data array 1:

1. UTLB data array 1 read

PPN, V, SZ, PR, C, D, SH, and WT are read into the data field from the UTLB entry corresponding to the entry set in the address field.

2. UTLB data array 1 write

PPN, V, SZ, PR, C, D, SH, and WT specified in the data field are written to the UTLB entry corresponding to the entry set in the address field.

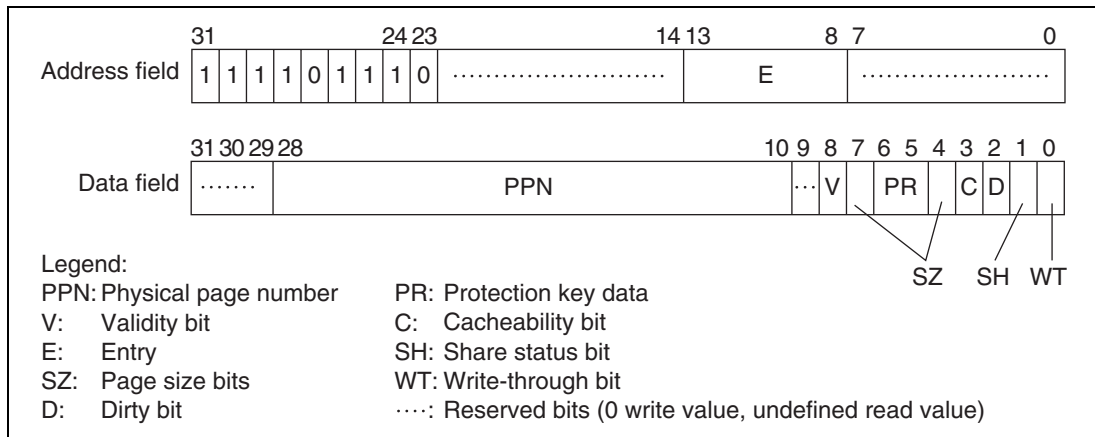


Figure 3.17 Memory-Mapped UTLB Data Array 1

3.7.6 UTLB Data Array 2

UTLB data array 2 is allocated to addresses H'F780 0000 to H'F7FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and SA and TC to be written to data array 2 are specified in the data field.

In the address field, bits [31:23] have the value H'F78 indicating UTLB data array 2, and the entry is selected by bits [13:8].

In the data field, TC is indicated by bit [3], and SA by bits [2:0].

The following two kinds of operation can be used on UTLB data array 2:

1. UTLB data array 2 read

SA and TC are read into the data field from the UTLB entry corresponding to the entry set in the address field.

2. UTLB data array 2 write

SA and TC specified in the data field are written to the UTLB entry corresponding to the entry set in the address field.

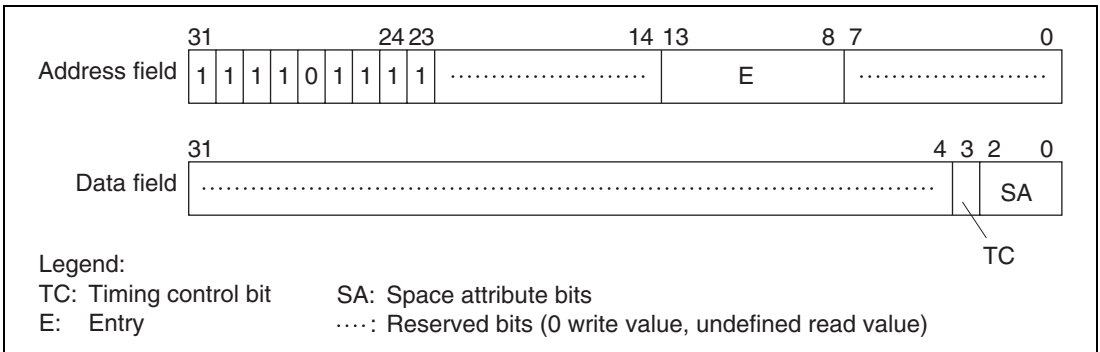


Figure 3.18 Memory-Mapped UTLB Data Array 2

3.8 Usage Notes

1. Address Space Identifier (ASID) in Single Virtual Memory Mode

Refer to the note in 3.3.7, Address Space Identifier (ASID).

Section 4 Caches

4.1 Overview

4.1.1 Features

The SH7751 has an on-chip 8-Kbyte instruction cache (IC) for instructions and 16-Kbyte operand cache (OC) for data. Half of the memory of the operand cache (8 Kbytes) can also be used as on-chip RAM. The features of these caches are summarized in table 4.1.

The SH7751 has an on-chip 16-Kbyte instruction cache (IC) for instructions and 32-Kbyte operand cache (OC) for data. Half of the operand cache memory (16 Kbytes) can also be used as on-chip RAM. When the EMODE bit in the CCR register is cleared to 0 in the SH7751R, both the IC and OC are set to SH7751 compatible mode. When the EMODE bit in the CCR register is set to 1, the cache characteristics are as shown in table 4.2. After a power-on reset or manual reset, the initial value of the EMODE bit is 0.

This LSI supports two 32-byte store queues (SQs) for performing high-speed writes to external memory. SQ features are shown in table 4.3.

Table 4.1 Cache Features (SH7751)

| Item | Instruction Cache | Operand Cache |
|--------------|-------------------|---|
| Capacity | 8-Kbyte cache | 16-Kbyte cache or 8-Kbyte cache + 8-Kbyte RAM |
| Type | Direct mapping | Direct mapping |
| Line size | 32 bytes | 32 bytes |
| Entries | 256 entry | 512 entry |
| Write method | | Copy-back/write-through selectable |

Table 4.2 Cache Features (SH7751R)

| Item | Instruction Cache | Operand Cache |
|----------------|-------------------------------------|---|
| Capacity | 16-Kbyte cache | 32-Kbyte cache or 16-Kbyte cache + 16-Kbyte RAM |
| Type | 2-way set-associative | 2-way set-associative |
| Line size | 32 bytes | 32 bytes |
| Entries | 256 entry/way | 512 entry/way |
| Write method | | Copy-back/write-through selectable |
| Replace method | LRU (Least Recently Used) algorithm | LRU (Least Recently Used) algorithm |

Table 4.3 Store Queue Features

| Item | Store Queues |
|--------------|---|
| Capacity | 2 × 32 bytes |
| Addresses | H'E000 0000 to H'E3FF FFFF |
| Write | Store instruction (1-cycle write) |
| Write-back | Prefetch instruction (PREF instruction) |
| Access right | MMU off: according to MMUCR.SQMD MMU on: according to individual page PR |

4.1.2 Register Configuration

Table 4.4 shows the cache control registers.

Table 4.4 Cache Control Registers

| Name | Abbreviation | R/W | Initial Value* ¹ | P4 Address* ² | Area 7 Address* ² | Access Size |
|----------------------------------|--------------|-----|-----------------------------|--------------------------|------------------------------|-------------|
| Cache control register | CCR | R/W | H'0000 0000 | H'FF00 001C | H'1F00 001C | 32 |
| Queue address control register 0 | QACR0 | R/W | Undefined | H'FF00 0038 | H'1F00 0038 | 32 |
| Queue address control register 1 | QACR1 | R/W | Undefined | H'FF00 003C | H'1F00 003C | 32 |

Notes: 1. The initial value is the value after a power-on or manual reset.

2. P4 address is the address when using the virtual/physical address space P4 area. The area 7 address is the address used when making an access from physical address space area 7 using the TLB.

4.2 Register Descriptions

There are three cache and store queue related control registers, as shown in figure 4.1.

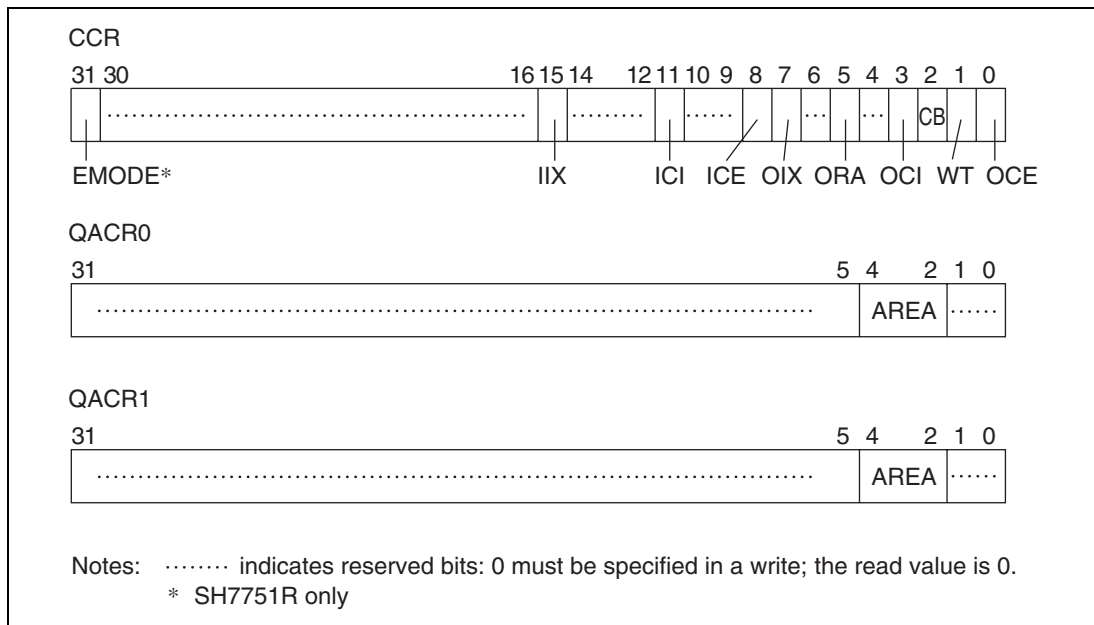


Figure 4.1 Cache and Store Queue Control Registers (CCR)

(1) Cache Control Register (CCR): CCR contains the following bits:

EMODE: Cache-double-mode (SH7751R only. Reserved bit in SH7751.)

IIX: IC index enable

ICI: IC invalidation

ICE: IC enable

OIX: OC index enable

ORA: OC RAM enable

OCI: OC invalidation

CB: Copy-back enable

WT: Write-through enable

OCE: OC enable

CCR can be accessed by longword-size access from H'FF00001C in the P4 area and H'1F00001C in area 7. The CCR bits are used for the cache settings described below. Consequently, CCR modifications must only be made by a program in the non-cached P2 area. After CCR is updated, an instruction that performs data access to the P0, P1, P3, or U0 area should be located at least

four instructions after the CCR update instruction. Also, a branch instruction to the P0, P1, P3, or U0 area should be located at least eight instructions after the CCR update instruction.

- **EMODE: Cache-double-mode bit**

Indicates whether or not cache-double-mode is used in the SH7751R. This bit is reserved in the SH7751. The EMODE bit cannot be modified while the cache is in use.

0: SH7751-compatible-mode*¹ (Initial value)

1: Cache-double-mode

Note: 1. Address allocation in OC index mode and RAM mode is not compatible with that in RAM mode.

- **IIX: IC index enable bit**

0: Effective address bits [12:5] used for IC entry selection

1: Effective address bits [25] and [11:5] used for IC entry selection

- **ICI: IC invalidation bit**

When 1 is written to this bit, the V bits of all IC entries are cleared to 0. This bit always returns 0 when read.

- **ICE: IC enable bit**

Indicates whether or not the IC is to be used. When address translation is performed, the IC cannot be used unless the C bit in the page management information is also 1.

0: IC not used

1: IC used

- **OIX: OC index enable bit*²**

0: Effective address bits [13:5] used for OC entry selection

1: Effective address bits [25] and [12:5] used for OC entry selection

Note: 2. In the SH7751R, clear the OIX bit to 0 when the ORA bit is 1.

- **ORA: OC RAM enable bit*³**

When the OC is enabled (OCE = 1), the ORA bit specifies whether the half of the OC are to be used as RAM. When the OC is not enabled (OCE = 0), the ORA bit should be cleared to 0.

0: Normal mode (the entire OC is used as a cache)

1: RAM mode (half of the OC is used as a cache and the other half is used as RAM)

Note: 3. In the SH7751R, clear the ORA bit to 0 when the OIX bit is 1.

- **OCI: OC invalidation bit**
When 1 is written to this bit, the V and U bits of all OC entries are cleared to 0. This bit always returns 0 when read.
- **CB: Copy-back bit**
Indicates the P1 area cache write mode.
0: Write-through mode
1: Copy-back mode
- **WT: Write-through bit**
Indicates the P0, U0, and P3 area cache write mode. When address translation is performed, the value of the WT bit in the page management information has priority.
0: Copy-back mode
1: Write-through mode
- **OCE: OC enable bit**
Indicates whether or not the OC is to be used. When address translation is performed, the OC cannot be used unless the C bit in the page management information is also 1.
0: OC not used
1: OC used

(2) Queue Address Control Register 0 (QACR0): QACR0 can be accessed by longword-size access from H'FF000038 in the P4 area and H'1F000038 in area 7. QACR0 specifies the area onto which store queue 0 (SQ0) is mapped when the MMU is off.

(3) Queue Address Control Register 1 (QACR1): QACR1 can be accessed by longword-size access from H'FF00003C in the P4 area and H'1F00003C in area 7. QACR1 specifies the area onto which store queue 1 (SQ1) is mapped when the MMU is off.

4.3 Operand Cache (OC)

4.3.1 Configuration

The operand cache in the SH7751 adopts the direct-mapping method, and consists of 512 cache lines. Each cache line is composed of a 19-bit tag, V bit, U bit, and 32-byte data. The operand cache in the SH7751R adopts the 2-way set-associative method, and each way consists of 512 cache lines.

Figure 4.2 shows the configuration of the operand cache in the SH7751.

Figure 4.3 shows the configuration of the operand cache in the SH7751R.

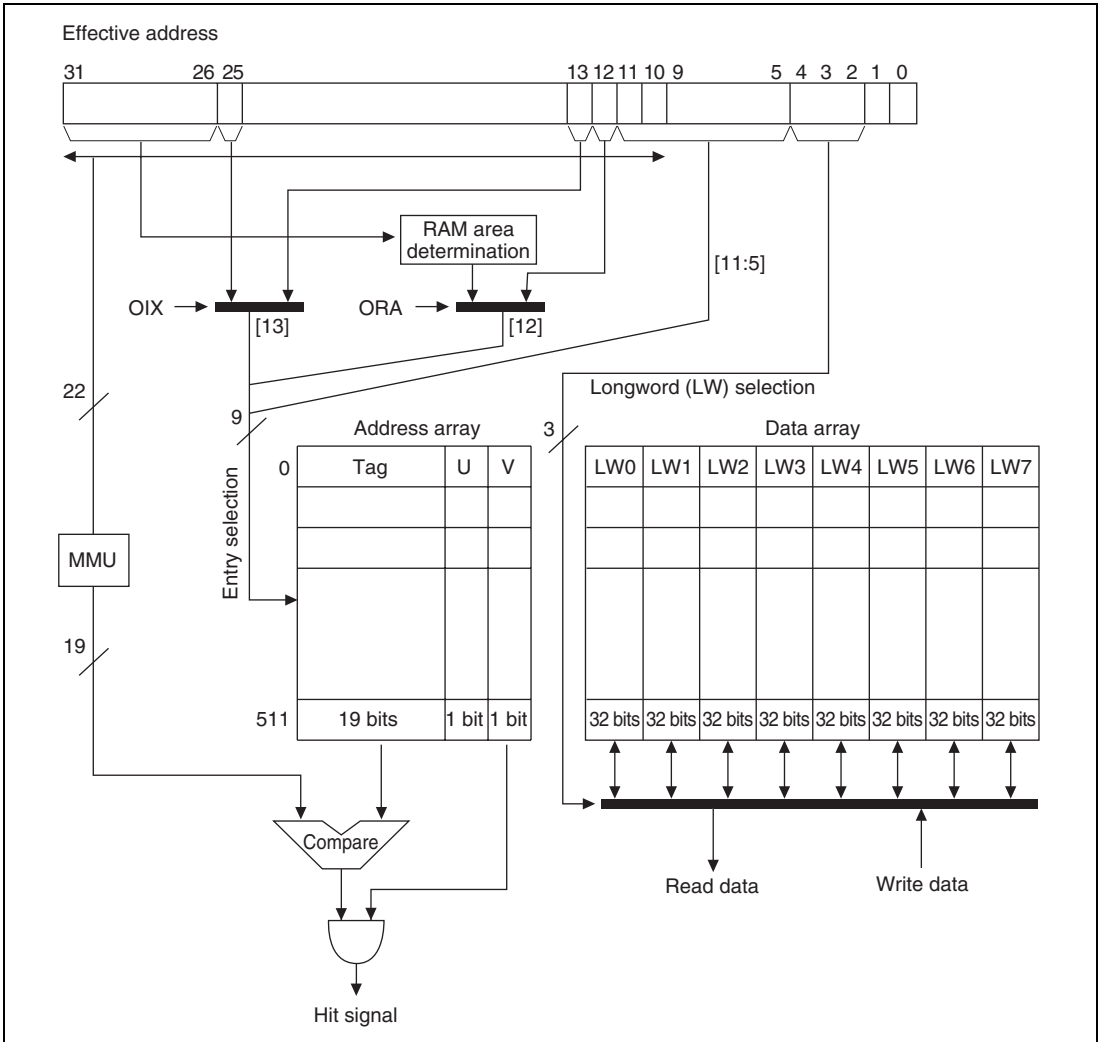


Figure 4.2 Configuration of Operand Cache (SH7751)

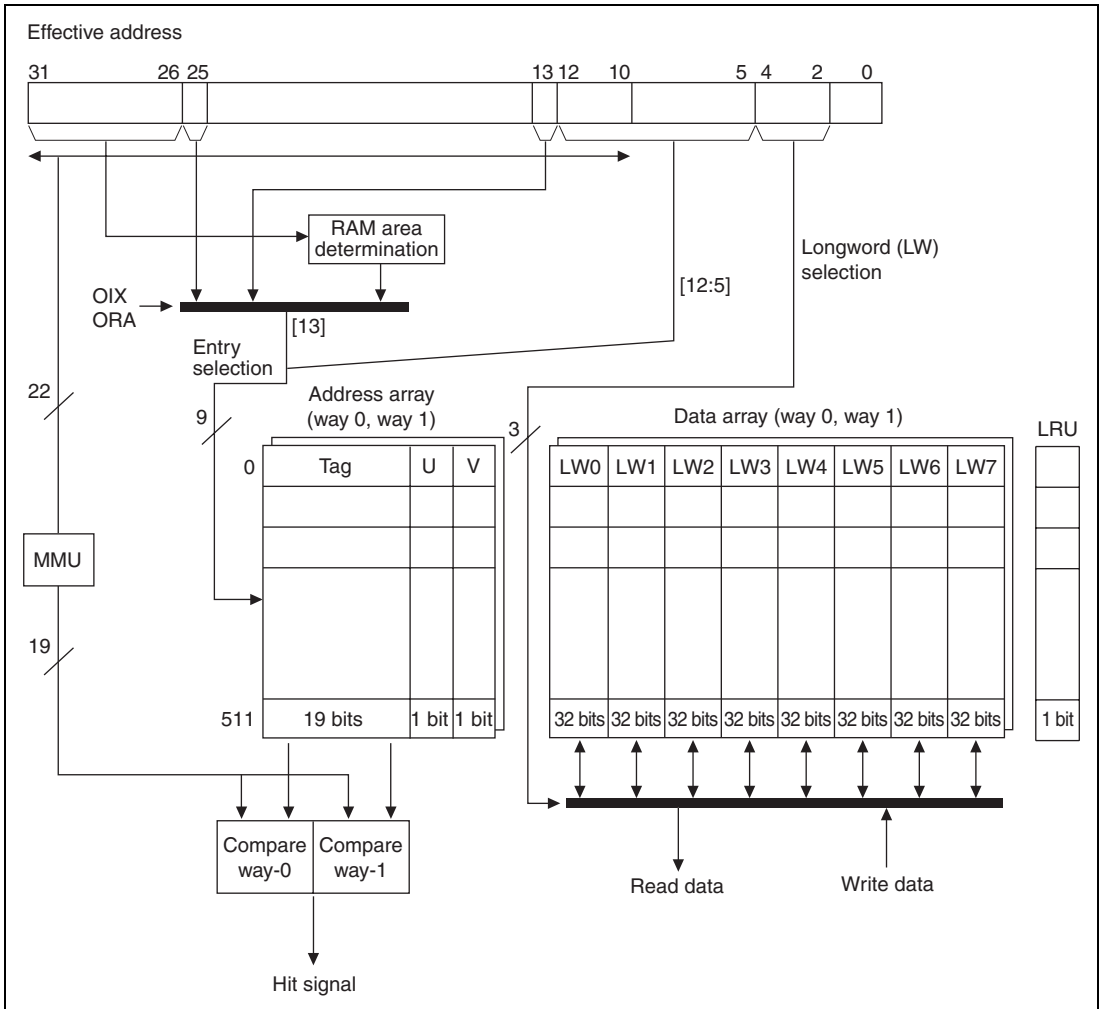


Figure 4.3 Configuration of Operand Cache (SH7751R)

- **Tag**
Stores the upper 19 bits of the 29-bit external address of the data line to be cached. The tag is not initialized by a power-on or manual reset.
- **V bit (validity bit)**
Indicates that valid data is stored in the cache line. When this bit is 1, the cache line data is valid. The V bit is initialized to 0 by a power-on reset, but retains its value in a manual reset.

- U bit (dirty bit)

The U bit is set to 1 if data is written to the cache line while the cache is being used in copy-back mode. That is, the U bit indicates a mismatch between the data in the cache line and the data in external memory. The U bit is never set to 1 while the cache is being used in write-through mode, unless it is modified by accessing the memory-mapped cache (see section 4.5, Memory-Mapped Cache Configuration (SH7751) and 4.6, Memory-Mapped Cache Configuration (SH7751R)). The U bit is initialized to 0 by a power-on reset, but retains its value in a manual reset.

- Data field

The data field holds 32 bytes (256 bits) of data per cache line. The data array is not initialized by a power-on or manual reset.

- LRU (SH7751R only)

In a 2-way set-associative system, up to two entry addresses can register the same data in cache. The LRU bit indicates to which way the entry is to be registered among the two ways. There is one LRU bit in each entry, and it is controlled by hardware. The LRU (Last Recently Used) algorithm that selects the most recently accessed way is used for way selection. The LRU bit is initialized to 0 by a power-on reset, but is not initialized by a manual reset. The LRU bit cannot be read from or written to by software.

4.3.2 Read Operation

When the OC is enabled (CCR.OCE = 1) and data is read by means of an effective address from a cacheable area, the cache operates as follows:

1. The tag, V bit, and U bit are read from the cache line indexed by effective address bits [13:5].
2. The tag is compared with bits [28:10] of the address resulting from effective address translation by the MMU:
 - If the tag matches and the V bit is 1 → (3a)
 - If the tag matches and the V bit is 0 → (3b)
 - If the tag does not match and the V bit is 0 → (3b)
 - If the tag does not match, the V bit is 1, and the U bit is 0 → (3b)
 - If the tag does not match, the V bit is 1, and the U bit is 1 → (3c)

3a. Cache hit

The data indexed by effective address bits [4:0] is read from the data field of the cache line indexed by effective address bits [13:5] in accordance with the access size (quadword/longword/word/byte).

3b. Cache miss (no write-back)

Data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and when the corresponding data arrives in the cache, the read data is returned to the CPU. While the remaining one cache line of data is being read, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit.

3c. Cache miss (with write-back)

The tag and data field of the cache line indexed by effective address bits [13:5] are saved in the write-back buffer. Then data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and when the corresponding data arrives in the cache, the read data is returned to the CPU. While the remaining one cache line of data is being read, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, 1 is written to the V bit, and 0 to the U bit. The data in the write-back buffer is then written back to external memory.

4.3.3 Write Operation

When the OC is enabled ($CCR.OCE = 1$) and data is written by means of an effective address to a cacheable area, the cache operates as follows:

1. The tag, V bit, and U bit are read from the cache line indexed by effective address bits [13:5].
2. The tag is compared with bits [28:10] of the address resulting from effective address translation by the MMU:

| | Copy-back | Write-through |
|---|-----------|---------------|
| • If the tag matches and the V bit is 1 | → (3a) | → (3b) |
| • If the tag matches and the V bit is 0 | → (3c) | → (3d) |
| • If the tag does not match and the V bit is 0 | → (3c) | → (3d) |
| • If the tag does not match, the V bit is 1, and the U bit is 0 | → (3c) | → (3d) |
| • If the tag does not match, the V bit is 1, and the U bit is 1 | → (3e) | → (3d) |

3a. Cache hit (copy-back)

A data write in accordance with the access size (quadword/longword/word/byte) is performed for the data indexed by bits [4:0] of the effective address and the data field of the cache line indexed by effective address bits [13:5]. Then 1 is set in the U bit.

3b. Cache hit (write-through)

A data write in accordance with the access size (quadword/longword/word/byte) is performed for the data field of the cache line indexed by effective address bits [13:5] and for the data indexed by effective address bits [4:0]. A write is also performed to the corresponding external memory using the specified access size.

3c. Cache miss (copy-back/no write-back)

A data write in accordance with the access size (quadword/longword/word/byte) is performed for the data field indexed by effective address bits [13:5] and for the data indexed by effective address bits [4:0]. Then, data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and one cache line of data is read excluding the written data. During this time, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit and U bit.

3d. Cache miss (write-through)

A write of the specified access size is performed to the external memory corresponding to the effective address. In this case, a write to cache is not performed.

3e. Cache miss (copy-back/with write-back)

The tag and data field of the cache line indexed by effective address bits [13:5] are first saved in the write-back buffer, and then a data write in accordance with the access size (quadword/longword/word/byte) is performed for the data indexed by bits [4:0] of the effective address of the data field of the cache line indexed by effective address bits [13:5]. Then, data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and one cache line of data is read excluding the written data. During this time, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit and U bit. The data in the write-back buffer is then written back to external memory.

4.3.4 Write-Back Buffer

In order to give priority to data reads to the cache and improve performance, this LSI has a write-back buffer which holds the relevant cache entry when it becomes necessary to purge a dirty cache entry into external memory as the result of a cache miss. The write-back buffer contains one cache line of data and the physical address of the purge destination.

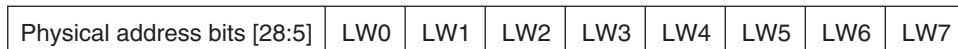


Figure 4.4 Configuration of Write-Back Buffer

4.3.5 Write-Through Buffer

This LSI has a 64-bit buffer for holding write data when writing data in write-through mode or writing to a non-cacheable area. This allows the CPU to proceed to the next operation as soon as the write to the write-through buffer is completed, without waiting for completion of the write to external memory.



Figure 4.5 Configuration of Write-Through Buffer

4.3.6 RAM Mode

Setting CCR.ORA to 1 enables 8 Kbytes of the operand cache to be used as RAM. The operand cache entries used as RAM are the 8 Kbytes of entries 128 to 255 and 384 to 511. In SH7751-compatible-mode in the SH7751R, the 8 Kbytes of operand cache entries 256 to 511 are used as RAM. In cache-double-mode in the SH7751R, the total 16 Kbytes of entries 256 to 511 in each way of the operand cache are used as RAM. Other entries can still be used as cache. RAM can be accessed using addresses H'7C00 0000 to H'7FFF FFFF. Byte-, word-, longword-, and quadword-size data reads and writes can be performed in the operand cache RAM area. Instruction fetches cannot be performed in this area.

Note that in the SH7751R, OC index mode cannot be used when RAM mode is used.

An example of RAM use is shown below. Here, the 4 Kbytes comprising OC entries 128 to 256 are designated as RAM area 1, and the 4 Kbytes comprising OC entries 384 to 511 as RAM area 2.

- When OC index mode is off (CCR.OIX = 0)

H'7C00 0000 to H'7C00 0FFF (4 KB): Corresponds to RAM area 1

H'7C00 1000 to H'7C00 1FFF (4 KB): Corresponds to RAM area 1

H'7C00 2000 to H'7C00 2FFF (4 KB): Corresponds to RAM area 2

H'7C00 3000 to H'7C00 3FFF (4 KB): Corresponds to RAM area 2

H'7C00 4000 to H'7C00 4FFF (4 KB): Corresponds to RAM area 1

: : :

RAM areas 1 and 2 then repeat every 8 Kbytes up to H'7FFF FFFF.

Thus, to secure a continuous 8-Kbyte RAM area, the area from H'7C00 1000 to H'7C00 2FFF can be used, for example.

- When OC index mode is on (CCR.OIX = 1)

H'7C00 0000 to H'7C00 0FFF (4 KB): Corresponds to RAM area 1

H'7C00 1000 to H'7C00 1FFF (4 KB): Corresponds to RAM area 1

H'7C00 2000 to H'7C00 2FFF (4 KB): Corresponds to RAM area 1

: : :

H'7DFF F000 to H'7DFF FFFF (4 KB): Corresponds to RAM area 1

H'7E00 0000 to H'7E00 0FFF (4 KB): Corresponds to RAM area 2

H'7E00 1000 to H'7E00 1FFF (4 KB): Corresponds to RAM area 2

: : :

H'7FFF F000 to H'7FFF FFFF (4 KB): Corresponds to RAM area 2

As the distinction between RAM areas 1 and 2 is indicated by address bit [25], the area from H'7DFF F000 to H'7E00 0FFF should be used to secure a continuous 8-Kbyte RAM area.

An example of RAM use in the SH7751R is shown below.

- SH7751-compatible-mode (CCR.EMODE = 0)

H'7C00 0000 to H'7C00 1FFF (8 KB): Corresponds to RAM area (entries 256 to 511)

H'7C00 2000 to H'7C00 3FFF (8 KB): Corresponds to RAM area (entries 256 to 511)

: : :

A shadow of the RAM area occurs every 8 Kbytes up to H'7FFF FFFF.

- Cache-double-mode (CCR.EMODE = 1)

The 8 Kbytes of entries 256 to 511 in OC way 0 are used as RAM area 1, and the 8 Kbytes of entries 256 to 511 in OC way 1 are used as RAM area 2.

H'7C00 0000 to H'7C00 1FFF (8 KB): Corresponds to RAM area 1

H'7C00 2000 to H'7C00 3FFF (8 KB): Corresponds to RAM area 2

H'7C00 4000 to H'7C00 5FFF (8 KB): Corresponds to RAM area 1

H'7C00 6000 to H'7C00 7FFF (8 KB): Corresponds to RAM area 2

: : :

A shadow of the RAM area occurs every 16 Kbytes up to H'7FFF FFFF.

4.3.7 OC Index Mode

Setting CCR.OIX to 1 enables OC indexing to be performed using bit [25] of the effective address. This is called OC index mode. In normal mode, with CCR.OIX cleared to 0, OC indexing is performed using bits [13:5] of the effective address. Using index mode allows the OC to be handled as two areas by means of effective address bit [25], providing efficient use of the cache.

Note that in the SH7751R, RAM mode cannot be used when OC index mode is used.

4.3.8 Coherency between Cache and External Memory

Coherency between cache and external memory should be assured by software. In this LSI, the following four new instructions are supported for cache operations. Details of these instructions are given in the Programming Manual.

| | | |
|-------------------------|----------------|--------------------------------------|
| Invalidate instruction: | OCBI @Rn | Cache invalidation (no write-back) |
| Purge instruction: | OCBP @Rn | Cache invalidation (with write-back) |
| Write-back instruction: | OCBWB @Rn | Cache write-back |
| Allocate instruction: | MOVCA.L R0,@Rn | Cache allocation |

4.3.9 Prefetch Operation

This LSI supports a prefetch instruction to reduce the cache fill penalty incurred as the result of a cache miss. If it is known that a cache miss will result from a read or write operation, it is possible to fill the cache with data beforehand by means of the prefetch instruction to prevent a cache miss due to the read or write operation, and so improve software performance. If a prefetch instruction is executed for data already held in the cache, or if the prefetch address results in a UTLB miss or a protection violation, the result is no operation, and an exception is not generated. Details of the prefetch instruction are given in the Programming Manual.

Prefetch instruction: PEF @Rn

4.3.10 Notes on Using OC RAM Mode (SH7751R Only) when in Cache Enhanced Mode

When in cache enhanced mode ($CCR.EMODE = 1$) on the SH7751R, and the OC RAM mode, in which half of the operand cache is used as internal RAM, is selected ($CCR.ORA = 1$), data in RAM may be updated incorrectly.

Conditions Under which Problem Occurs: Incorrect data may be written to RAM when the following four conditions are satisfied.

Condition 1: Cache enhanced mode ($CCR.EMODE = 1$) is specified.

Condition 2: The RAM mode ($CCR.ORA = 1$) in which half of the operand cache is used as RAM is specified.

Condition 3: An exception or an interrupt occurs.

Note: This includes a break triggered by a debugging tool swapping an instruction (a break occurring when a TRAPA instruction or undefined instruction code H'FFFD is swapped for an instruction).

Condition 4: A store instruction (MOV, FMOV, AND.B, OR.B, XOR.B, MOVCA.L, STC.L, or STS.L) that accesses internal RAM (H'7C000000 to H'7FFFFFFF) exists within four words after the instruction associated with the exception or interrupt described in condition 3. This includes cases where the store instruction that accesses internal RAM itself generates an exception.

Description: When the problem occurs, 8 bytes of incorrect data is written to the 8-byte boundary that includes an address that differs by H'2000 from the address accessed by the store instruction that accesses internal RAM mentioned in condition 4. For example, when a long word is stored at address H'7C000204, the 8 bytes of data in the internal RAM mapped to addresses H'7C002200 to H'7C002207 becomes corrupted.

Examples

Example 1 A store instruction accessing internal RAM occurs within four instructions after an instruction generating a TLB miss exception.

| | |
|-----------------------|--|
| MOV.L #H'0C400000, R0 | R0 is an address causing a TLB miss. |
| MOV.L #H'7C000204, R1 | R1 is an address mapped to internal RAM. |
| MOV.L @R0, R2 | TLB miss exception occurs. |
| NOP | 1st word |
| NOP | 2nd word |
| NOP | 3rd word |
| MOV.L R3, @R1 | Store instruction accessing internal RAM |

Example 2 A store instruction accessing internal RAM occurs within four instructions after an instruction causing an interrupt to be accepted.

| | |
|-----------------------|--|
| MOV.L #H'7C002000, R1 | R1 is an address mapped to internal RAM. |
| MOV.L #H'12345678, R0 | An interrupt is accepted after this instruction. |
| NOP | 1st word |
| NOP | 2nd word |
| NOP | 3rd word |
| MOV.L R0, @R1 | Store instruction accessing internal RAM |

Example 3 A debugging tool generates a break to swap an instruction.

| Original Instruction String | After Instruction Swap Break | |
|-----------------------------|------------------------------|---|
| MOV.L #H'7C000000, R0 | MOV.L #H'7C000000, R0 | Contains address corresponding to R0. |
| ADD R0, R0 | TRAPA #H'01 | R0 address is not a problem in original instruction string. |
| MOV.L R1, @R0 | MOV.L R1, @R0 | Internal RAM is accessed by a store operation because ADD is not executed. The store is cancelled, but 2LW starting at H'7C002000 is corrupted. |

Workarounds: When RAM mode is specified in cache enhanced mode, either of the following workarounds can be used to avoid the problem.

Workaround 1:

Use only 8 Kbytes of the 16-Kbyte internal RAM area. In this case, RAM areas for which address bits [12:0] are identical and only bit [13] differs must not be used. For example, the 8-Kbyte RAM area from H'7C000000 to H'7C001FFF or from H'7C001000 to H'7C002FFF may be used.

Note: When a break is used to swap instructions by a debugging tool, etc., a memory access address may be changed when an instruction following the instruction generating the break is swapped for another instruction, causing the unused 8-Kbyte RAM area to be accessed. This will result in the problem described above. However, this phenomenon only occurs during debugging when a break is used to swap instructions. Using a break with no instruction swapping will not cause the problem.

Workaround 2:

Ensure that there are no instructions that generate an interrupt or exception within four instructions after an instruction that accesses internal RAM. For example, the internal RAM area can be used as a data table that is accessed only by load instructions, with writes to the internal RAM area only being performed when the table is generated. In this case, set SR.BL to 1 to disable interrupts while writing to the table. Also take measures to ensure that no exceptions due to TLB misses, etc., occur while writing to the table.

Note: The problem still may occur when a break is used to swap instructions by a debugging tool. This phenomenon only occurs during debugging when a break is used to swap instructions. Using a break with no instruction swapping will not cause the problem.

4.4 Instruction Cache (IC)

4.4.1 Configuration

The instruction cache consists of 256 cache lines, each composed of a 19-bit tag, V bit, and 32-byte data (16 instructions). The instruction cache in the SH7751R adopts the 2-way set-associative method, and each way consists of 256 cache lines.

Figure 4.6 shows the configuration of the instruction cache in the SH7751.

Figure 4.7 shows the configuration of the instruction cache in the SH7751R.

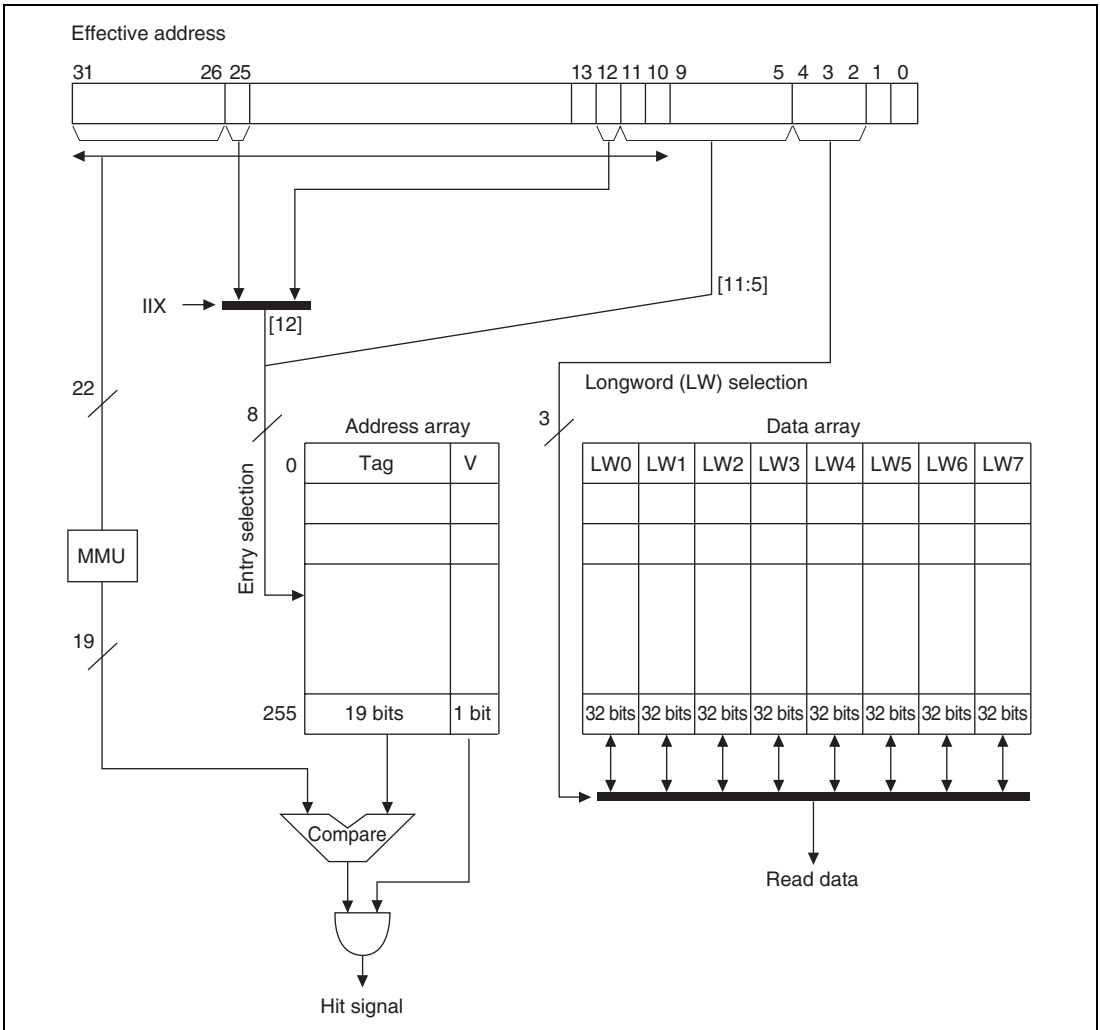


Figure 4.6 Configuration of Instruction Cache (SH7751)

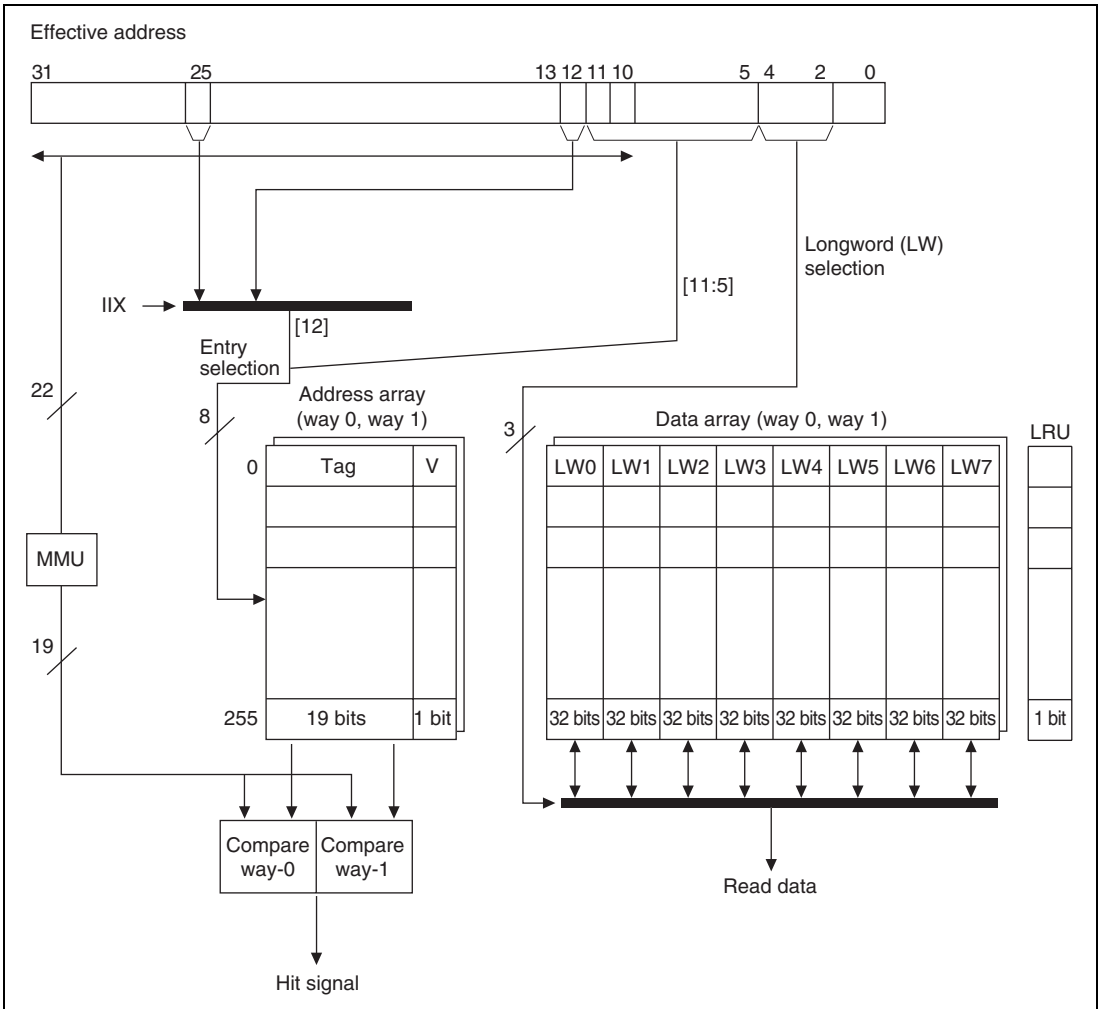


Figure 4.7 Configuration of Instruction Cache (SH7751R)

- **Tag**
Stores the upper 19 bits of the 29-bit external address of the data line to be cached. The tag is not initialized by a power-on or manual reset.
- **V bit (validity bit)**
Indicates that valid data is stored in the cache line. When this bit is 1, the cache line data is valid. The V bit is initialized to 0 by a power-on reset, but retains its value in a manual reset.

- Data array

The data field holds 32 bytes (256 bits) of data per cache line. The data array is not initialized by a power-on or manual reset.

- LRU (SH7751R only)

In a 2-way set-associative system, up to two entry addresses can register the same data in cache. The LRU bit indicates to which way the entry is to be registered among the two ways. There is one LRU bit in each entry, and it is controlled by hardware. The LRU (Last Recently Used) algorithm that selects the most recently accessed way is used for way selection. The LRU bit is initialized to 0 by a power-on reset, but is not initialized by a manual reset. The LRU bit cannot be read from or written to by software.

4.4.2 Read Operation

When the IC is enabled (CCR.ICE = 1) and instruction fetches are performed by means of an effective address from a cacheable area, the instruction cache operates as follows:

1. The tag and V bit are read from the cache line indexed by effective address bits [12:5].
2. The tag is compared with bits [28:10] of the address resulting from effective address translation by the MMU:
 - If the tag matches and the V bit is 1 → (3a)
 - If the tag matches and the V bit is 0 → (3b)
 - If the tag does not match and the V bit is 0 → (3b)
 - If the tag does not match and the V bit is 1 → (3b)

3a. Cache hit

The data indexed by effective address bits [4:2] is read as an instruction from the data field of the cache line indexed by effective address bits [12:5].

3b. Cache miss

Data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and when the corresponding data arrives in the cache, the read data is returned to the CPU as an instruction. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit.

4.4.3 IC Index Mode

Setting CCR.IIX to 1 enables IC indexing to be performed using bit [25] of the effective address. This is called IC index mode. In normal mode, with CCR.IIX cleared to 0, IC indexing is performed using bits [12:5] of the effective address. Using index mode allows the IC to be handled as two areas by means of effective address bit [25], providing efficient use of the cache.

4.5 Memory-Mapped Cache Configuration (SH7751)

To enable the IC and OC to be managed by software, the IC contents can be read and written by a P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in another area. In this case, a branch to the P0, U0, P1, or P3 area should be made at least 8 instructions after this MOV instruction. The OC contents can be read and written by a P1 or P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in another area. In this case, a branch to the P0, U0, or P3 area should be made at least 8 instructions after this MOV instruction. The IC and OC are allocated to the P4 area in physical memory space. Only data accesses can be used on both the IC address array and data array and the OC address array and data array, and the access size is always longword. Instruction fetches cannot be performed in these areas. For reserved bits, a write value of 0 should be specified, and read values are undefined.

4.5.1 IC Address Array

The IC address array is allocated to addresses H'F000 0000 to H'FOFF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the write tag and V bit are specified in the data field.

In the address field, bits [31:24] have the value H'F0 indicating the IC address array, and the entry is specified by bits [12:5]. CCR.IIX has no effect on this entry specification. The address array bit [3] association bit (A bit) specifies whether or not association is performed when writing to the IC address array. As only longword access is used, 0 should be specified for address field bits [1:0].

In the data field, the tag is indicated by bits [31:10], and the V bit by bit [0]. As the IC address array tag is 19 bits in length, data field bits [31:29] are not used in the case of a write in which association is not performed. Data field bits [31:29] are used for the virtual address specification only in the case of a write in which association is performed.

The following three kinds of operation can be used on the IC address array:

1. IC address array read

The tag and V bit are read into the data field from the IC entry corresponding to the entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.

2. IC address array write (non-associative)

The tag and V bit specified in the data field are written to the IC entry corresponding to the entry set in the address field. The A bit in the address field should be cleared to 0.

3. IC address array write (associative)

When a write is performed with the A bit in the address field set to 1, the tag stored in the entry specified in the address field is compared with the tag specified in the data field. If the MMU is enabled at this time, comparison is performed after the virtual address specified by data field bits [31:10] has been translated to a physical address using the ITLB. If the addresses match and the V bit is 1, the V bit specified in the data field is written into the IC entry. In other cases, no operation is performed. This operation is used to invalidate a specific IC entry. If an ITLB miss occurs during address translation, or the comparison shows a mismatch, an interrupt is not generated, no operation is performed, and the write is not executed. If an instruction TLB multiple hit exception occurs during address translation, processing switches to the instruction TLB multiple hit exception handling routine.

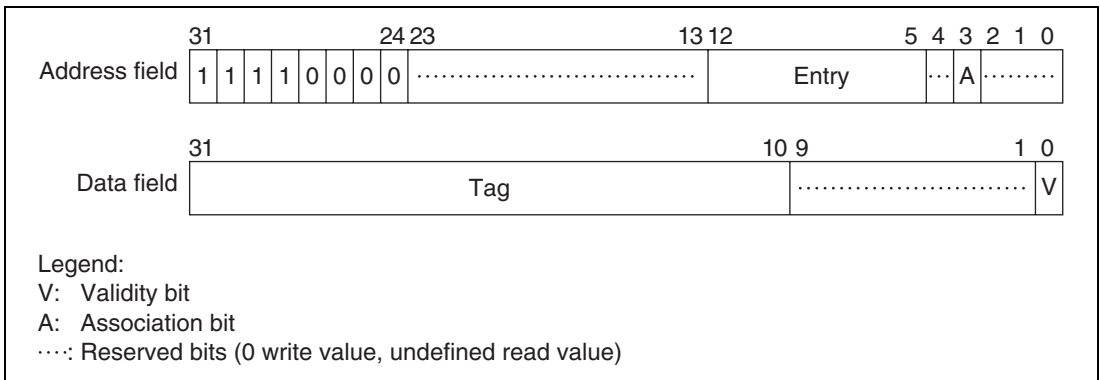


Figure 4.8 Memory-Mapped IC Address Array

4.5.2 IC Data Array

The IC data array is allocated to addresses H'F100 0000 to H'F1FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the longword data to be written is specified in the data field.

In the address field, bits [31:24] have the value H'F1 indicating the IC data array, and the entry is specified by bits [12:5]. CCR.IIX has no effect on this entry specification. Address field bits [4:2] are used for the longword data specification in the entry. As only longword access is used, 0 should be specified for address field bits [1:0].

The data field is used for the longword data specification.

The following two kinds of operation can be used on the IC data array:

1. IC data array read

Longword data is read into the data field from the data specified by the longword specification bits in the address field in the IC entry corresponding to the entry set in the address field.

2. IC data array write

The longword data specified in the data field is written for the data specified by the longword specification bits in the address field in the IC entry corresponding to the entry set in the address field.

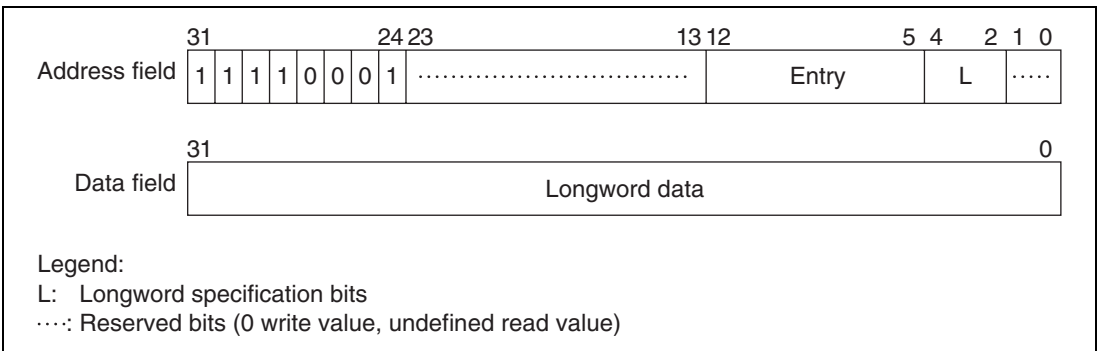


Figure 4.9 Memory-Mapped IC Data Array

4.5.3 OC Address Array

The OC address array is allocated to addresses H'F400 0000 to H'F4FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the write tag, U bit, and V bit are specified in the data field.

In the address field, bits [31:24] have the value H'F4 indicating the OC address array, and the entry is specified by bits [13:5]. CCR.OIX and CCR.ORA have no effect on this entry specification. The address array bit [3] association bit (A bit) specifies whether or not association is performed when writing to the OC address array. As only longword access is used, 0 should be specified for address field bits [1:0].

In the data field, the tag is indicated by bits [31:10], the U bit by bit [1], and the V bit by bit [0]. As the OC address array tag is 19 bits in length, data field bits [31:29] are not used in the case of a write in which association is not performed. Data field bits [31:29] are used for the virtual address specification only in the case of a write in which association is performed.

The following three kinds of operation can be used on the OC address array:

1. OC address array read

The tag, U bit, and V bit are read into the data field from the OC entry corresponding to the entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.

2. OC address array write (non-associative)

The tag, U bit, and V bit specified in the data field are written to the OC entry corresponding to the entry set in the address field. The A bit in the address field should be cleared to 0.

When a write is performed to a cache line for which the U bit and V bit are both 1, after write-back of that cache line, the tag, U bit, and V bit specified in the data field are written.

3. OC address array write (associative)

When a write is performed with the A bit in the address field set to 1, the tag stored in the entry specified in the address field is compared with the tag specified in the data field. If the MMU is enabled at this time, comparison is performed after the virtual address specified by data field bits [31:10] has been translated to a physical address using the UTLB. If the addresses match and the V bit is 1, the U bit and V bit specified in the data field are written into the OC entry. This operation is used to invalidate a specific OC entry. In other cases, no operation is performed. If the OC entry U bit is 1, and 0 is written to the V bit or to the U bit, write-back is performed. If a UTLB miss occurs during address translation, or the comparison shows a mismatch, an exception is not generated, no operation is performed, and the write is

not executed. If a data TLB multiple hit exception occurs during address translation, processing switches to the data TLB multiple hit exception handling routine.

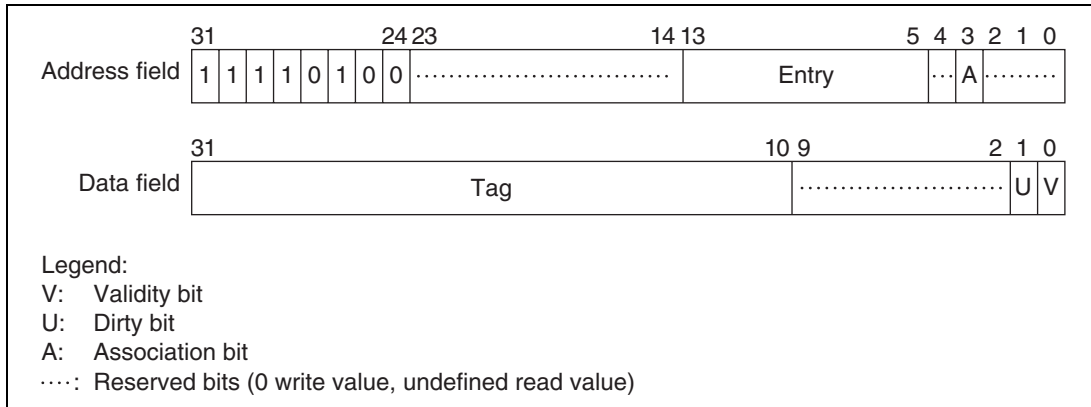


Figure 4.10 Memory-Mapped OC Address Array

4.5.4 OC Data Array

The OC data array is allocated to addresses H'F500 0000 to H'F5FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the longword data to be written is specified in the data field.

In the address field, bits [31:24] have the value H'F5 indicating the OC data array, and the entry is specified by bits [13:5]. CCR.OIX and CCR.ORA have no effect on this entry specification. Address field bits [4:2] are used for the longword data specification in the entry. As only longword access is used, 0 should be specified for address field bits [1:0].

The data field is used for the longword data specification.

The following two kinds of operation can be used on the OC data array:

1. OC data array read

Longword data is read into the data field from the data specified by the longword specification bits in the address field in the OC entry corresponding to the entry set in the address field.

2. OC data array write

The longword data specified in the data field is written for the data specified by the longword specification bits in the address field in the OC entry corresponding to the entry set in the address field. This write does not set the U bit to 1 on the address array side.

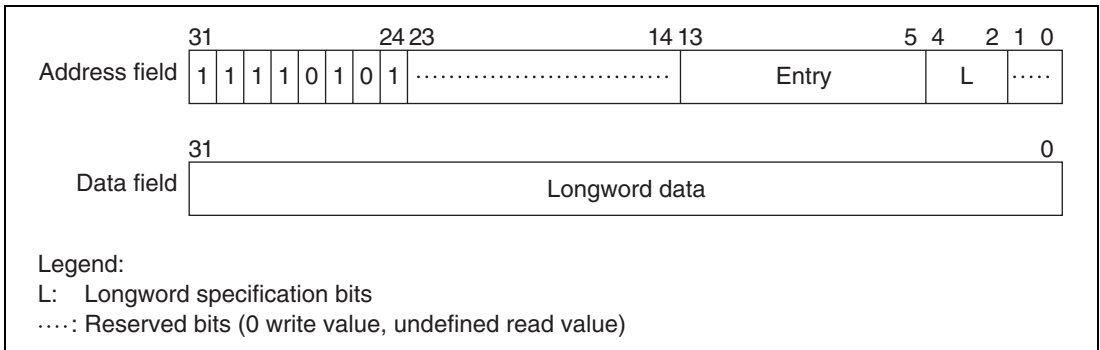


Figure 4.11 Memory-Mapped OC Data Array

4.6 Memory-Mapped Cache Configuration (SH7751R)

To enable the IC and OC to be managed by software, IC contents can be read and written by a P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in another area. In this case, a branch to the P0, U0, P1, or P3 area should be made at least 8 instructions after this MOV instruction. The OC contents can be read and written by a P1 or P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in another area. In this case, a branch to the P0, U0, or P3 area should be made at least 8 instructions after this MOV instruction. The IC and OC are allocated to the P4 area in physical memory space. Only data accesses can be used on both the IC address array and data array and the OC address array and data array, and the access size is always longword. Instruction fetches cannot be performed in these areas. For reserved bits, a write value of 0 should be specified, and read values are undefined. Note that the memory-mapped cache configuration in SH7751-compatible-mode of the SH7751R is the same as that in the SH7751.

4.6.1 IC Address Array

The IC address array is allocated to addresses H'F000 0000 to H'FOFF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The way and entry to be accessed are specified in the address field, and the write tag and V bit are specified in the data field.

In the address field, bits [31:24] have the value H'F0 indicating the IC address array, the way is specified by bit [13], and the entry is specified by bits [12:5]. CCR.IIX has no effect on this entry specification. Address field bit [3], that is the association bit (A bit), specifies whether or not association is performed when writing to the IC address array. As only longword access is used, 0 should be specified for address field bits [1:0].

In the data field, the tag is indicated by bits [31:10], and the V bit by bit [0]. As the IC address array tag is 19 bits in length, data field bits [31:29] are not used in the case of a write in which association is not performed. Data field bits [31:29] are used for the virtual address specification only in the case of a write in which association is performed.

The following three kinds of operation can be used on the IC address array:

1. IC address array read

The tag and V bit are read into the data field from the IC entry corresponding to the way and entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.

2. IC address array write (non-associative)

The tag and V bit specified in the data field are written to the IC entry corresponding to the way and entry set in the address field. The A bit in the address field should be cleared to 0.

3. IC address array write (associative)

When a write is performed with the A bit in the address field set to 1, each way's tag stored in the entry specified in the address field is compared with the tag specified in the data field. The way number set in bit [13] is ignored. If the MMU is enabled at this time, comparison is performed after the virtual address specified by data field bits [31:10] has been translated to a physical address using the ITLB. If the addresses match and the V bit in that way is 1, the V bit specified in the data field is written into the IC entry. In other cases, no operation is performed. This operation is used to invalidate a specific IC entry. If an ITLB miss occurs during address translation, or the comparison shows a mismatch, an interrupt is not generated, no operation is performed, and the write is not executed. If an instruction TLB multiple hit exception occurs during address translation, processing switches to the instruction TLB multiple hit exception handling routine.

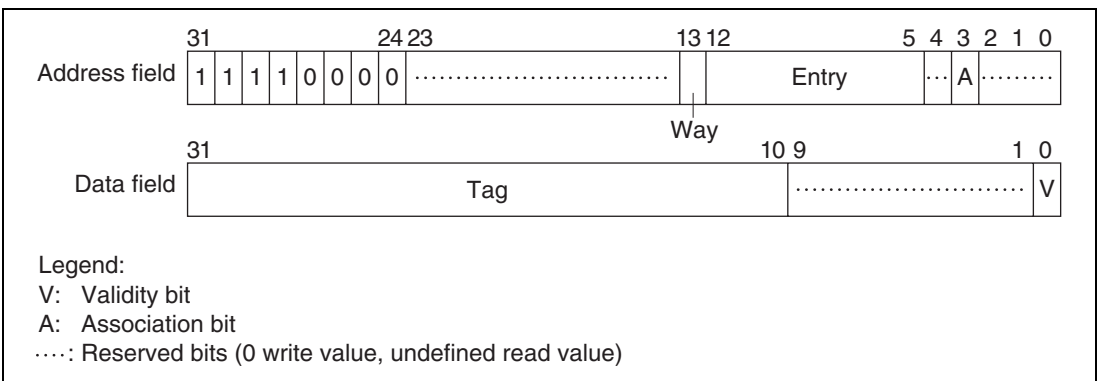


Figure 4.12 Memory-Mapped IC Address Array

4.6.2 IC Data Array

The IC data array is allocated to addresses H'F100 0000 to H'F1FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The way and entry to be accessed are specified in the address field, and the longword data to be written is specified in the data field.

In the address field, bits [31:24] have the value H'F1 indicating the IC data array, the way is specified by bit [13], and the entry is specified by bits [12:5]. CCR.IIX has no effect on this entry specification. Address field bits [4:2] are used for the longword data specification in the entry. As only longword access is used, 0 should be specified for address field bits [1:0].

The data field is used for the longword data specification.

The following two kinds of operation can be used on the IC data array:

1. IC data array read

Longword data is read into the data field from the data specified by the longword specification bits in the address field in the IC entry corresponding to the way and entry set in the address field.

2. IC data array write

The longword data specified in the data field is written for the data specified by the longword specification bits in the address field in the IC entry corresponding to the way and entry set in the address field.

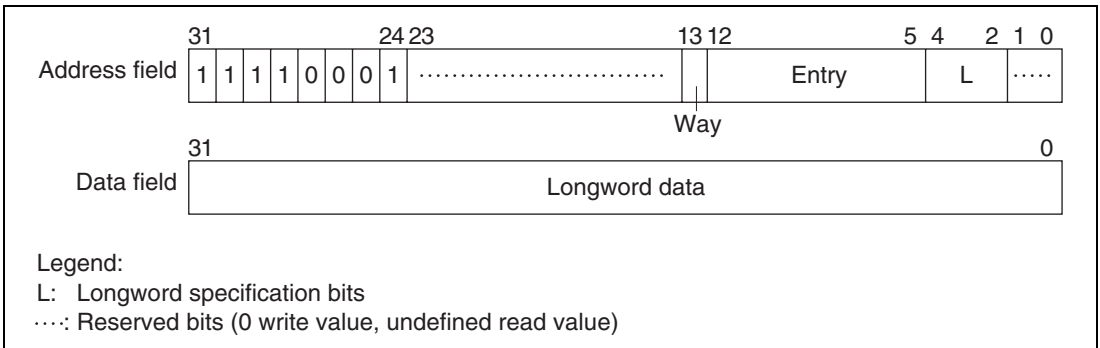


Figure 4.13 Memory-Mapped IC Data Array

4.6.3 OC Address Array

The OC address array is allocated to addresses H'F400 0000 to H'F4FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The way and entry to be accessed are specified in the address field, and the write tag, U bit, and V bit are specified in the data field.

In the address field, bits [31:24] have the value H'F4 indicating the OC address array, the way is specified by bit [14], and the entry is specified by bits [13:5]. CCR.OIX has no effect on this entry specification. The OC address array access in RAM mode (CCR.ORA = 1) is performed only to cache, and bit [13] specifies the way. For details on address allocation, see section 4.6.5, Summary of Memory-Mapped OC Addresses. Address field bit [3], that is the association bit (A bit), specifies whether or not association is performed when writing to the OC address array. As only longword access is used, 0 should be specified for address field bits [1:0].

In the data field, the tag is indicated by bits [31:10], the U bit by bit [1], and the V bit by bit [0]. As the OC address array tag is 19 bits in length, data field bits [31:29] are not used in the case of a write in which association is not performed. Data field bits [31:29] are used for the virtual address specification only in the case of a write in which association is performed.

The following three kinds of operation can be used on the OC address array:

1. OC address array read

The tag, U bit, and V bit are read into the data field from the OC entry corresponding to the way and entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.

2. OC address array write (non-associative)

The tag, U bit, and V bit specified in the data field are written to the OC entry corresponding to the way and entry set in the address field. The A bit in the address field should be cleared to 0. When a write is performed to a cache line for which the U bit and V bit are both 1, after write-back of that cache line, the tag, U bit, and V bit specified in the data field are written.

3. OC address array write (associative)

When a write is performed with the A bit in the address field set to 1, each way's tag stored in the entry specified in the address field is compared with the tag specified in the data field. The way number set in bit [14] is ignored. If the MMU is enabled at this time, comparison is performed after the virtual address specified by data field bits [31:10] has been translated to a physical address using the UTLB. If the addresses match and the V bit in that way is 1, the U bit and V bit specified in the data field are written into the OC entry. This operation is used to invalidate a specific OC entry. In other cases, no operation is performed. If the OC entry U bit is 1, and 0 is written to the V bit or to the U bit, write-back is performed. If a UTLB miss

occurs during address translation, or the comparison shows a mismatch, an exception is not generated, no operation is performed, and the write is not executed. If a data TLB multiple hit exception occurs during address translation, processing switches to the data TLB multiple hit exception handling routine.

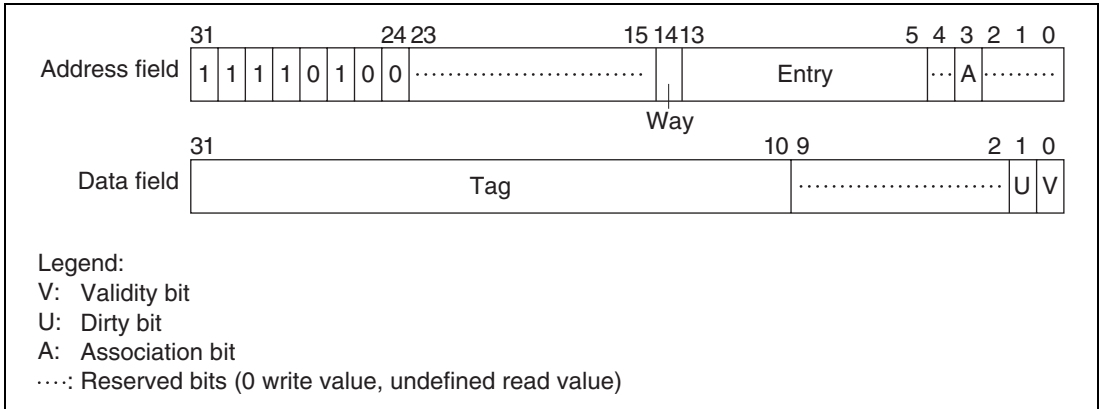


Figure 4.14 Memory-Mapped OC Address Array

4.6.4 OC Data Array

The OC data array is allocated to addresses H'F500 0000 to H'F5FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The way and entry to be accessed are specified in the address field, and the longword data to be written is specified in the data field.

In the address field, bits [31:24] have the value H'F5 indicating the OC data array, the way is specified by bit [14], and the entry is specified by bits [13:5]. CCR.OIX has no effect on this entry specification. The OC address array access in RAM mode (CCR.ORA = 1) is performed only to cache, and bit [13] specifies the way. For details on address allocation, see section 4.6.5, Summary of Memory-Mapped OC Addresses. Address field bits [4:2] are used for the longword data specification in the entry. As only longword access is used, 0 should be specified for address field bits [1:0].

The data field is used for the longword data specification.

The following two kinds of operation can be used on the OC data array:

1. OC data array read

Longword data is read into the data field from the data specified by the longword specification bits in the address field in the OC entry corresponding to the way and entry set in the address field.

2. OC data array write

The longword data specified in the data field is written for the data specified by the longword specification bits in the address field in the OC entry corresponding to the way and entry set in the address field. This write does not set the U bit to 1 on the address array side.

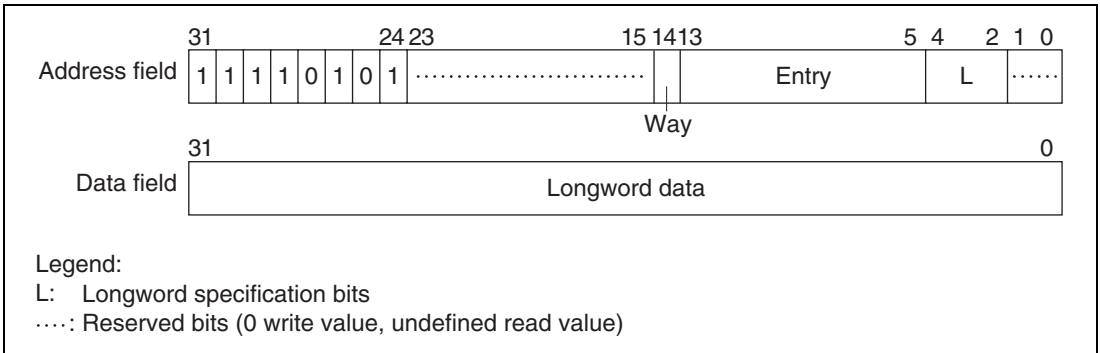


Figure 4.15 Memory-Mapped OC Data Array

4.6.5 Summary of Memory-Mapped OC Addresses

The memory-mapped OC addresses in cache-double-mode in the SH7751R are summarized below using data area access as an example.

- Normal mode (CCR.ORA = 0)
 - H'F500 0000 to H'F500 3FFF (16 KB): Way 0 (entries 0 to 511)
 - H'F500 4000 to H'F500 7FFF (16 KB): Way 1 (entries 0 to 511)
 - : : :
 - A shadow of the cache area occurs every 32 Kbytes up to H'F5FF FFFF.
- RAM mode (CCR.ORA = 1)
 - H'F500 0000 to H'F500 1FFF (8 KB): Way 0 (entries 0 to 255)
 - H'F500 2000 to H'F500 3FFF (8 KB): Way 1 (entries 0 to 255)
 - : : :
 - A shadow of the cache area occurs every 16 Kbytes up to H'F5FF FFFF.

4.7 Store Queues

Two 32-byte store queues (SQs) are supported to perform high-speed writes to external memory. When not using the SQs, the low power dissipation power-down modes, in which SQ functions are stopped, can be used. The queue address control registers (QACR0 and QACR1) cannot be accessed while SQ functions are stopped. See section 9, Power-Down Modes, for the procedure for stopping SQ functions. Note that power-down modes (STBCR2.MSTP6 = 1) that stop SQ functions cannot be used on the SH7751 when using the operand cache for write-back operations.*

Note: * Cases where write-back operations are performed:

- When the operand cache is used in copy-back mode (determined by the CCR.CB and CCR.WT bits and, if address translation is performed, the WT bit in the page management information)
- When the memory allocation cache function is used to write to the OC address array, and an entry is generated when both the V and U bits are set to 1

4.7.1 SQ Configuration

There are two 32-byte store queues, SQ0 and SQ1, as shown in figure 4.16. These two store queues can be set independently.

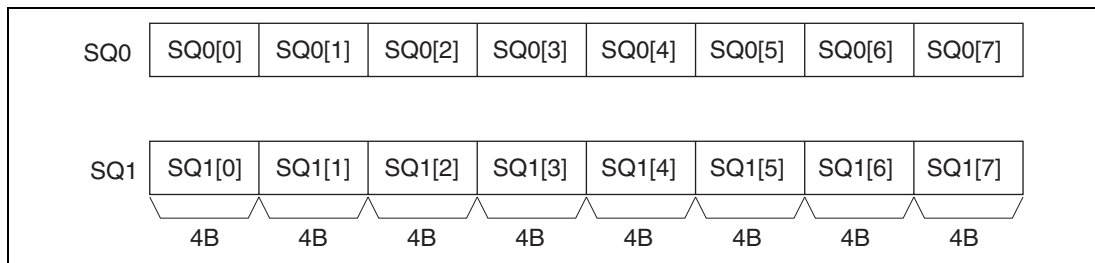


Figure 4.16 Store Queue Configuration

4.7.2 SQ Writes

A write to the SQs can be performed using a store instruction on P4 area H'E000 0000 to H'E3FF FFFC. A longword or quadword access size can be used. The meaning of the address bits is as follows:

| | | |
|----------|------------|--|
| [31:26]: | 111000 | Store queue specification |
| [25:6]: | Don't care | Used for external memory transfer/access right |
| [5]: | 0/1 | 0: SQ0 specification 1: SQ1 specification |

| | | |
|--------|------------------|--|
| [4:2]: | LW specification | Specifies longword position in SQ0/SQ1 |
| [1:0] | 00 | Fixed at 0 |

4.7.3 Transfer to External Memory

Transfer from the SQs to external memory can be performed with a prefetch instruction (PREF). Issuing a PREF instruction for P4 area H'E000 0000 to H'E3FF FFFC starts a burst transfer from the SQs to external memory. The burst transfer length is fixed at 32 bytes, and the start address is always at a 32-byte boundary. While the contents of one SQ are being transferred to external memory, the other SQ can be written to without a penalty cycle, but writing to the SQ involved in the transfer to external memory is deferred until the transfer is completed.

The SQ transfer destination external address bit [28:0] specification is as shown below, according to whether the MMU is on or off.

- When MMU is on

The SQ area (H'E000 0000 to H'E3FF FFFF) is set in VPN of the UTLB, and the transfer destination external address in PPN. The ASID, V, SZ, SH, PR, and D bits have the same meaning as for normal address translation, but the C and WT bits have no meaning with regard to this page. Since burst transfer is prohibited for PCMCIA areas, the SA and TC bits also have no meaning.

When a prefetch instruction is issued for the SQ area, address translation is performed and external address bits [28:10] are generated in accordance with the SZ bit specification. For external address bits [9:5], the address prior to address translation is generated in the same way as when the MMU is off. External address bits [4:0] are fixed at 0. Transfer from the SQs to external is performed to this address.

- When MMU is off

The SQ area (H'E000 0000 to H'E3FF FFFF) is specified as the address at which a PREF instruction is issued. The meaning of address bits [31:0] is as follows:

| | | |
|----------|------------|---|
| [31:26]: | 111000 | Store queue specification |
| [25:6]: | Address | External address bits [25:6] |
| [5]: | 0/1 | 0: SQ0 specification 1: SQ1 specification and external address bit [5] |
| [4:2]: | Don't care | No meaning in a prefetch |
| [1:0] | 00 | Fixed at 0 |

External address bits [28:26], which cannot be generated from the above address, are generated from the QACR0/1 registers.

QACR0 [4:2]: External address bits [28:26] corresponding to SQ0

QACR1 [4:2]: External address bits [28:26] corresponding to SQ1

External address bits [4:0] are always fixed at 0 since burst transfer starts at a 32-byte boundary.

In this LSI, data transfer to a PCMCIA interface area is always performed using the SA and TC bits in the PTEA register.

4.7.4 Determination of SQ Access Exception

Determination of an exception in a write to an SQ or transfer to external memory (PREF instruction) is performed as follows. If an exception occurs in an SQ write, the SQ contents may be corrupted in the SH7751 (see section 4.7.6, SQ Usage Notes), but the previous values of the SQ contents are guaranteed in the SH7751R. If an exception occurs in transfer from an SQ to external memory, the transfer to external memory will be aborted.

- When MMU is on

Operation is in accordance with the address translation information recorded in the UTLB, and MMUCR.SQMD. Write type exception judgment is performed for writes to the SQs, and read type for transfer from the SQs to external memory (PREF instruction), and a TLB miss exception, protection violation exception, or initial page write exception is generated. However, if SQ access is enabled, in privileged mode only, by MMUCR.SQMD, an address error will be flagged in user mode even if address translation is successful.
- When MMU is off

Operation is in accordance with MMUCR.SQMD.

0: Privileged/user access possible
1: Privileged access possible

If the SQ area is accessed in user mode when MMUCR.SQMD is set to 1, an address error will be flagged.

4.7.5 SQ Read (SH7751R only)

In the SH7751R, the SQ contents can be read by a load instruction for addresses H'FF001000 to H'FF00103C in the P4 area in privileged mode. The access size is always longword.

[31:6]: H'FF001000 (store queue specification)

[5]: 0/1 (0: SQ0 specification, 1: SQ1 specification)

[4:2]: LW specification (specification of longword position in SQ0 or SQ1)

[1:0]: 00 (fixed to 0)

4.7.6 SQ Usage Notes (SH7751 Only)

If an exception occurs within the three instructions preceding an instruction that writes to an SQ in the SH7751, a branch may be made to the exception handling routine after execution of the SQ write that should be suppressed when an exception occurs.

This may be due to the bug described in (1) or (2) below.

(1) When SQ data is transferred to external memory within a normal program

If a PREF instruction for transfer from an SQ to external memory is included in the three instructions preceding an SQ store instruction, the SQ is updated because the SQ write that should be suppressed when a branch is made to the exception handling routine is executed, and after returning from the exception handling routine the execution order of the PREF instruction and SQ store instruction is reversed, so that erroneous data may be transferred to external memory.

(2) When SQ data is transferred to external memory in an exception handling routine

If store queue contents are transferred to external memory within an exception handling routine, erroneous data may be transferred to external memory.

Example 1: When an SQ store instruction is executed after a PREF instruction for transfer from that same SQ to external memory

PREF instruction

- ; PREF instruction for transfer from SQ to external memory
- ; Address of this instruction is saved to SPC when exception occurs.
- ; Instruction 1, instruction 2, or instruction 3 may be executed on return from exception handling
- ; routine.

Instruction 1 ; May be executed if an SQ store instruction.

Instruction 2 ; May be executed if an SQ store instruction.

Instruction 3 ; May be executed if an SQ store instruction.

Instruction 4 ; Not executed even if an SQ store instruction.

Example 2: When an instruction at which an exception occurs is a branch instruction and a branch is made

Instruction 1 (branch instruction); Address of this instruction is saved to SPC when exception occurs.

Instruction 2 ; May be executed if an instruction 1 delay slot instruction and an SQ store instruction.

Instruction 3

Instruction 4

Instruction 5

Instruction 6

Instruction 7 (instruction 1 branch destination)

; May be executed if an SQ store instruction.

Instruction 8 ; May be executed if an SQ store instruction.

Example 3: When an instruction at which an exception occurs is a branch instruction but a branch is not made

Instruction 1 (branch instruction); Address of this instruction is saved to SPC when exception occurs.

Instruction 2 ; May be executed if an SQ store instruction.

Instruction 3 ; May be executed if an SQ store instruction.

Instruction 4 ; May be executed if an SQ store instruction.

Instruction 5

Both A and B below must be satisfied in order to prevent this bug.

A: When a store queue store instruction is executed after a PREF instruction for transfer from that same store queue (SQ0, SQ1) to external memory, (1) and (2) below must be satisfied.

(1) Insert three NOP instructions^{*1} between the two instructions.

(2) Do not place a PREF instruction for transfer from a store queue to external memory in the delay slot of a branch instruction.

B: Do not execute a PREF instruction for transfer from a store queue to external memory within an exception handling routine.

If the above is executed and there is a store queue store instruction among the four instructions^{*2} including the instruction at the address indicated by the SPC, the state of the contents transferred to external memory by the PREF instruction may be that when execution of this store instruction is completed.

Notes: 1. If there are other instructions between the two instructions, this bug can be prevented if the total number of other instructions plus NOP instructions is at least three.

2. If the instruction at the address indicated by the SPC is a branch instruction, this also applies to two instructions at the branch destination.

Section 5 Exceptions

5.1 Overview

5.1.1 Features

Exception handling is processing handled by a special routine, separate from normal program processing, that is executed by the CPU in case of abnormal events. For example, if the executing instruction ends abnormally, appropriate action must be taken in order to return to the original program sequence, or report the abnormality before terminating the processing. The process of generating an exception handling request in response to abnormal termination, and passing control flow to an exception handling routine, etc., is given the generic name of exception handling.

SH-4 exception handling is of three kinds: for resets, general exceptions, and interrupts.

5.1.2 Register Configuration

The registers used in exception handling are shown in table 5.1.

Table 5.1 Exception-Related Registers

| Name | Abbrevia- tion | R/W | Initial Value | P4 Address* ² | Area 7 Address* ² | Access Size |
|--------------------------|-------------------|-----|---|-----------------------------|---------------------------------|----------------|
| TRAPA exception register | TRA | R/W | Undefined | H'FF00 0020 | H'1F00 0020 | 32 |
| Exception event register | EXPEVT | R/W | H'0000 0000/ H'0000 0020* ¹ | H'FF00 0024 | H'1F00 0024 | 32 |
| Interrupt event register | INTEVT | R/W | Undefined | H'FF00 0028 | H'1F00 0028 | 32 |

- Notes: 1. H'0000 0000 is set in a power-on reset, and H'0000 0020 in a manual reset.
 2. P4 address is the address when using the virtual/physical address space P4 area. When making an access from area 7 in the physical address space using the TLB, the three high most bits of the address are ignored.

5.2 Register Descriptions

There are three registers related to exception handling. Addresses are allocated for these, and can be accessed by specifying the P4 address or area 7 address.

1. The exception event register (EXPEVT) resides at P4 address H'FF00 0024, and contains a 12-bit exception code. The exception code set in EXPEVT is that for a reset or general exception event. The exception code is set automatically by hardware when an exception is accepted. EXPEVT can also be modified by software.
2. The interrupt event register (INTEVT) resides at P4 address H'FF00 0028, and contains a 14-bit exception code. The exception code set in INTEVT is that for an interrupt request. The exception code is set automatically by hardware when an exception is accepted. INTEVT can also be modified by software.
3. The TRAPA exception register (TRA) resides at P4 address H'FF00 0020, and contains 8-bit immediate data (imm) for the TRAPA instruction. TRA is set automatically by hardware when a TRAPA instruction is executed. TRA can also be modified by software.

The bit configurations of EXPEVT, INTEVT, and TRA are shown in figure 5.1.

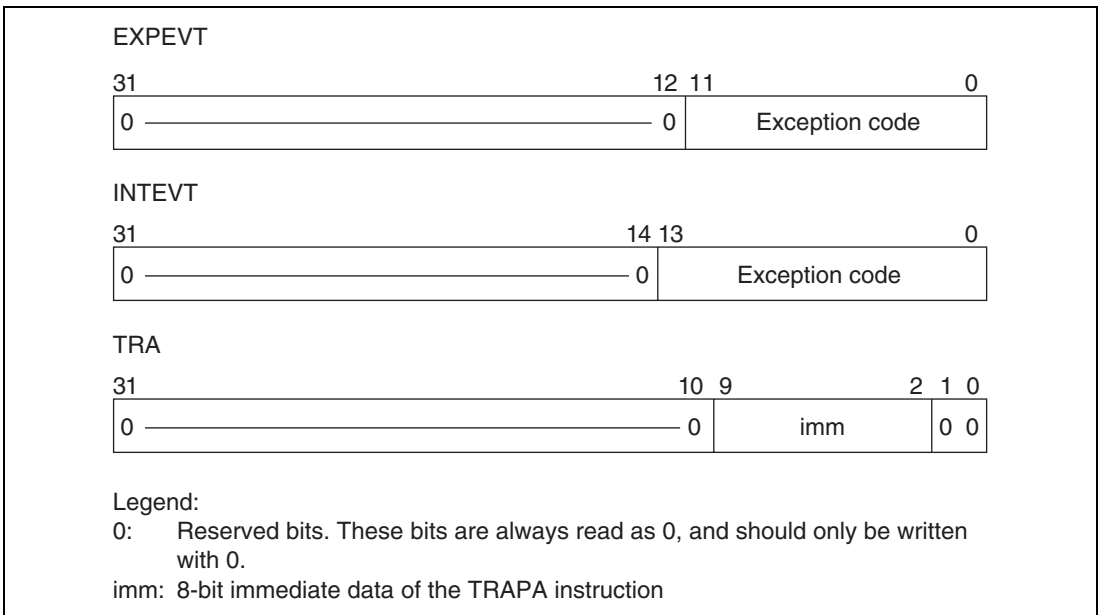


Figure 5.1 Register Bit Configurations

5.3 Exception Handling Functions

5.3.1 Exception Handling Flow

In exception handling, the contents of the program counter (PC), status register (SR) and R15 are saved in the saved program counter (SPC), saved status register (SSR), and saved general register 15 (SGR), and the CPU starts execution of the appropriate exception handling routine according to the vector address. An exception handling routine is a program written by the user to handle a specific exception. The exception handling routine is terminated and control returned to the original program by executing a return-from-exception instruction (RTE). This instruction restores the PC and SR contents and returns control to the normal processing routine at the point at which the exception occurred. The SGR contents are not written back to R15 by an RTE instruction.

The basic processing flow is as follows. See section 2, Programming Model, for the meaning of the individual SR bits.

1. The PC, SR, and R15 contents are saved in SPC, SSR, and SGR.
2. The block bit (BL) in SR is set to 1.
3. The mode bit (MD) in SR is set to 1.
4. The register bank bit (RB) in SR is set to 1.
5. In a reset, the FPU disable bit (FD) in SR is cleared to 0.
6. The exception code is written to bits 11–0 of the exception event register (EXPEVT) or to bits 13–0 of the interrupt event register (INTEVT).
7. The CPU branches to the determined exception handling vector address, and the exception handling routine begins.

5.3.2 Exception Handling Vector Addresses

The reset vector address is fixed at H'A000 0000. General exception and interrupt vector addresses are determined by adding the offset for the specific event to the vector base address, which is set by software in the vector base register (VBR). In the case of the TLB miss exception, for example, the offset is H'0000 0400, so if H'9C08 0000 is set in VBR, the exception handling vector address will be H'9C08 0400. If a further exception occurs at the exception handling vector address, a duplicate exception will result, and recovery will be difficult; therefore, fixed physical addresses (P1, P2) should be specified for vector addresses.

5.4 Exception Types and Priorities

Table 5.2 shows the types of exceptions, with their relative priorities, vector addresses, and exception/interrupt codes.

Table 5.2 Exceptions

| Exception Category | Execution Mode | Exception | Priority Level | Priority Order | Vector Address | Offset | Exception Code |
|--|-------------------|---|----------------------------|--|----------------|---------|----------------|
| Reset | Abort type | Power-on reset | 1 | 1 | H'A000 0000 | — | H'000 |
| | | Manual reset | 1 | 2 | H'A000 0000 | — | H'020 |
| | | H-UDI reset | 1 | 1 | H'A000 0000 | — | H'000 |
| | | Instruction TLB multiple-hit exception | 1 | 3 | H'A000 0000 | — | H'140 |
| | | Data TLB multiple-hit exception | 1 | 4 | H'A000 0000 | — | H'140 |
| General exception | Re-execution type | User break before instruction execution* ¹ | 2 | 0 | (VBR/DBR) | H'100/— | H'1E0 |
| | | Instruction address error | 2 | 1 | (VBR) | H'100 | H'0E0 |
| | | Instruction TLB miss exception | 2 | 2 | (VBR) | H'400 | H'040 |
| | | Instruction TLB protection violation exception | 2 | 3 | (VBR) | H'100 | H'0A0 |
| | | General illegal instruction exception | 2 | 4 | (VBR) | H'100 | H'180 |
| | | Slot illegal instruction exception | 2 | 4 | (VBR) | H'100 | H'1A0 |
| | | General FPU disable exception | 2 | 4 | (VBR) | H'100 | H'800 |
| | | Slot FPU disable exception | 2 | 4 | (VBR) | H'100 | H'820 |
| | | Data address error (read) | 2 | 5 | (VBR) | H'100 | H'0E0 |
| | | Data address error (write) | 2 | 5 | (VBR) | H'100 | H'100 |
| | | Data TLB miss exception (read) | 2 | 6 | (VBR) | H'400 | H'040 |
| | | Data TLB miss exception (write) | 2 | 6 | (VBR) | H'400 | H'060 |
| | | Data TLB protection violation exception (read) | 2 | 7 | (VBR) | H'100 | H'0A0 |
| | | Data TLB protection violation exception (write) | 2 | 7 | (VBR) | H'100 | H'0C0 |
| | | FPU exception | 2 | 8 | (VBR) | H'100 | H'120 |
| | | Initial page write exception | 2 | 9 | (VBR) | H'100 | H'080 |
| | | Completion type | Unconditional trap (TRAPA) | User break after instruction execution* ¹ | 2 | 4 | (VBR) |
| User break after instruction execution* ¹ | 2 | | | 10 | (VBR/DBR) | H'100/— | H'1E0 |

| Exception Category | Execution Mode | Exception | Priority Level | Priority Order | Vector Address | Offset | Exception Code | | |
|--------------------|-----------------|---|----------------|----------------|----------------|--------|----------------|-------|-------|
| Interrupt | Completion type | Nonmaskable interrupt | 3 | — | (VBR) | H'600 | H'1C0 | | |
| | | External interrupts | IRL3–IRLO | 0 | 4 | *2 | (VBR) | H'600 | H'200 |
| | | | 1 | | | | | | H'220 |
| | | | 2 | | | | | | H'240 |
| | | | 3 | | | | | | H'260 |
| | | | 4 | | | | | | H'280 |
| | | | 5 | | | | | | H'2A0 |
| | | | 6 | | | | | | H'2C0 |
| | | | 7 | | | | | | H'2E0 |
| | | | 8 | | | | | | H'300 |
| | | | 9 | | | | | | H'320 |
| | | | A | | | | | | H'340 |
| | | | B | | | | | | H'360 |
| | | | C | | | | | | H'380 |
| | | | D | | | | | | H'3A0 |
| | | | E | | | | | | H'3C0 |
| | | Peripheral module interrupt (module/source) | TMU0 | TUNI0 | 4 | *2 | (VBR) | H'600 | H'400 |
| | | | TMU1 | TUNI1 | | | | | H'420 |
| | | | TMU2 | TUNI2 | | | | | H'440 |
| | | | | TICPI2 | | | | | H'460 |
| | | | TMU3 | TUNI3 | | | | | H'B00 |
| | | | TMU4 | TUNI4 | | | | | H'B80 |
| | | | RTC | ATI | | | | | H'480 |
| | | | | PRI | | | | | H'4A0 |
| | | | | CUI | | | | | H'4C0 |
| | | | SCI | ERI | | | | | H'4E0 |
| | | | | RXI | | | | | H'500 |
| TXI | | | | | | | H'520 | | |
| TEI | | | | | | H'540 | | | |
| WDT | ITI | | | | | H'560 | | | |
| REF | RCMI | | | | | H'580 | | | |
| | ROVI | | | | | H'5A0 | | | |

| Exception Category | Execution Mode | Exception | Priority Level | Priority Order | Vector Address | Offset | Exception Code | |
|--------------------|-----------------|---|----------------|---------------------|----------------|--------|----------------|-------|
| Interrupt | Completion type | Peripheral module interrupt (module/source) | H-UDI | H-UDI | 4 | *2 | (VBR) | H'600 |
| | | | GPIO | GPIOI | | | | H'620 |
| | | | DMAC | DMTE0 | | | | H'640 |
| | | | | DMTE1 | | | | H'660 |
| | | | | DMTE2 | | | | H'680 |
| | | | | DMTE3 | | | | H'6A0 |
| | | | | DMTE4* ³ | | | | H'780 |
| | | | | DMTE5* ³ | | | | H'7A0 |
| | | | | DMTE6* ³ | | | | H'7C0 |
| | | | | DMTE7* ³ | | | | H'7E0 |
| | | | | DMAE | | | | H'6C0 |
| | | | SCIF | ERI | | | | H'700 |
| | | | | RXI | | | | H'720 |
| | | | | BRI | | | | H'740 |
| | | | | TXI | | | | H'760 |
| | | | PCIC | PCISERR | | | | H'A00 |
| | | | | PCIEERR | | | | H'AE0 |
| | | | | PCIPWDWN | | | | H'AC0 |
| | | | | PCIPWON | | | | H'AA0 |
| | | | | PCIDMA0 | | | | H'A80 |
| | | | | PCIDMA1 | | | | H'A60 |
| | PCIDMA2 | H'A40 | | | | | | |
| | PCIDMA3 | H'A20 | | | | | | |

Priority: Priority is first assigned by priority level, then by priority order within each level (the lowest number represents the highest priority).

Exception transition destination: Control passes to H'A000 0000 in a reset, and to [VBR + offset] in other cases.

Exception code: Stored in EXPEVT for a reset or general exception, and in INTEVT for an interrupt.

IRL: Interrupt request level (pins IRL3–IRL0).

Module/source: See the sections on the relevant peripheral modules.

- Notes:
1. When BRCR.UBDE = 1, PC = DBR. In other cases, PC = VBR + H'100.
 2. The priority order of external interrupts and peripheral module interrupts can be set by software.
 3. SH7751R only

5.5 Exception Flow

5.5.1 Exception Flow

Figure 5.2 shows an outline flowchart of the basic operations in instruction execution and exception handling. For the sake of clarity, the following description assumes that instructions are executed sequentially, one by one. Figure 5.2 shows the relative priority order of the different kinds of exceptions (reset/general exception/interrupt). Register settings in the event of an exception are shown only for SSR, SPC, SGR, EXPEVT/INTEVT, SR, and PC, but other registers may be set automatically by hardware, depending on the exception. For details, see section 5.6, Description of Exceptions. Also, see section 5.6.4, Priority Order with Multiple Exceptions, for exception handling during execution of a delayed branch instruction and a delay slot instruction, and in the case of instructions in which two data accesses are performed.

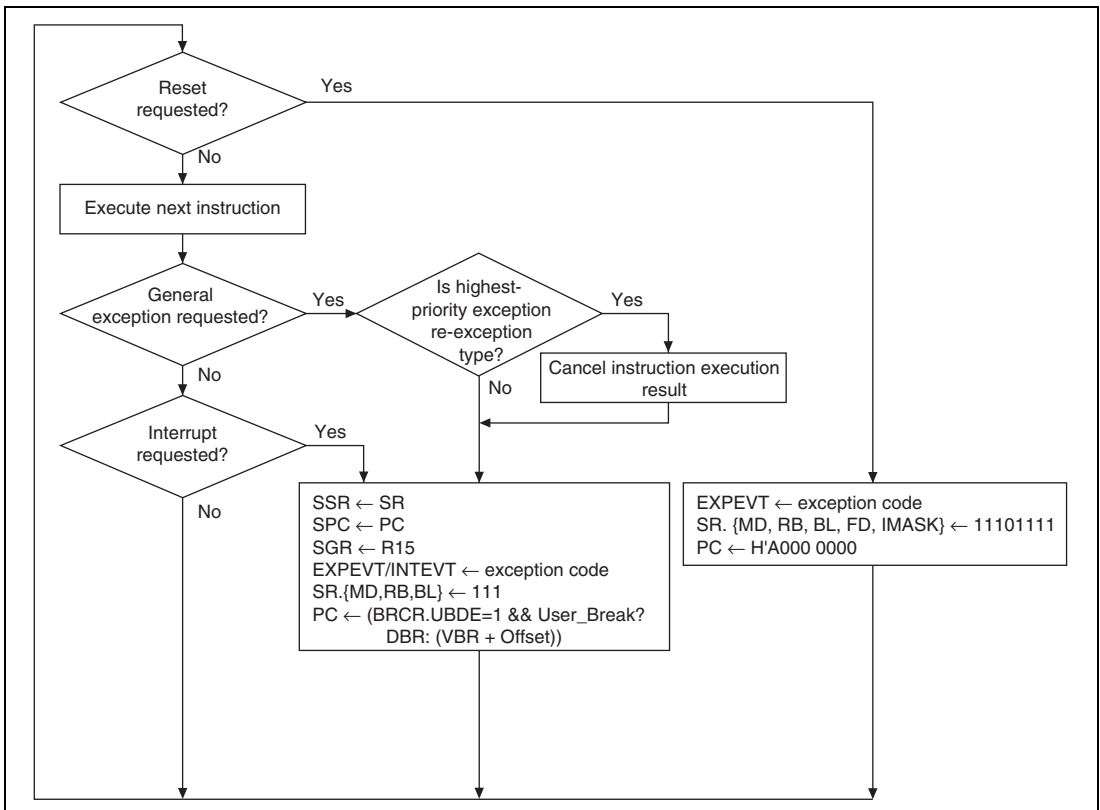


Figure 5.2 Instruction Execution and Exception Handling

5.5.2 Exception Source Acceptance

A priority ranking is provided for all exceptions for use in determining which of two or more simultaneously generated exceptions should be accepted. Five of the general exceptions—the general illegal instruction exception, slot illegal instruction exception, general FPU disable exception, slot FPU disable exception, and unconditional trap exception—are detected in the process of instruction decoding, and do not occur simultaneously in the instruction pipeline. These exceptions therefore all have the same priority. General exceptions are detected in the order of instruction execution. However, exception handling is performed in the order of instruction flow (program order). Thus, an exception for an earlier instruction is accepted before that for a later instruction. An example of the order of acceptance for general exceptions is shown in figure 5.3.

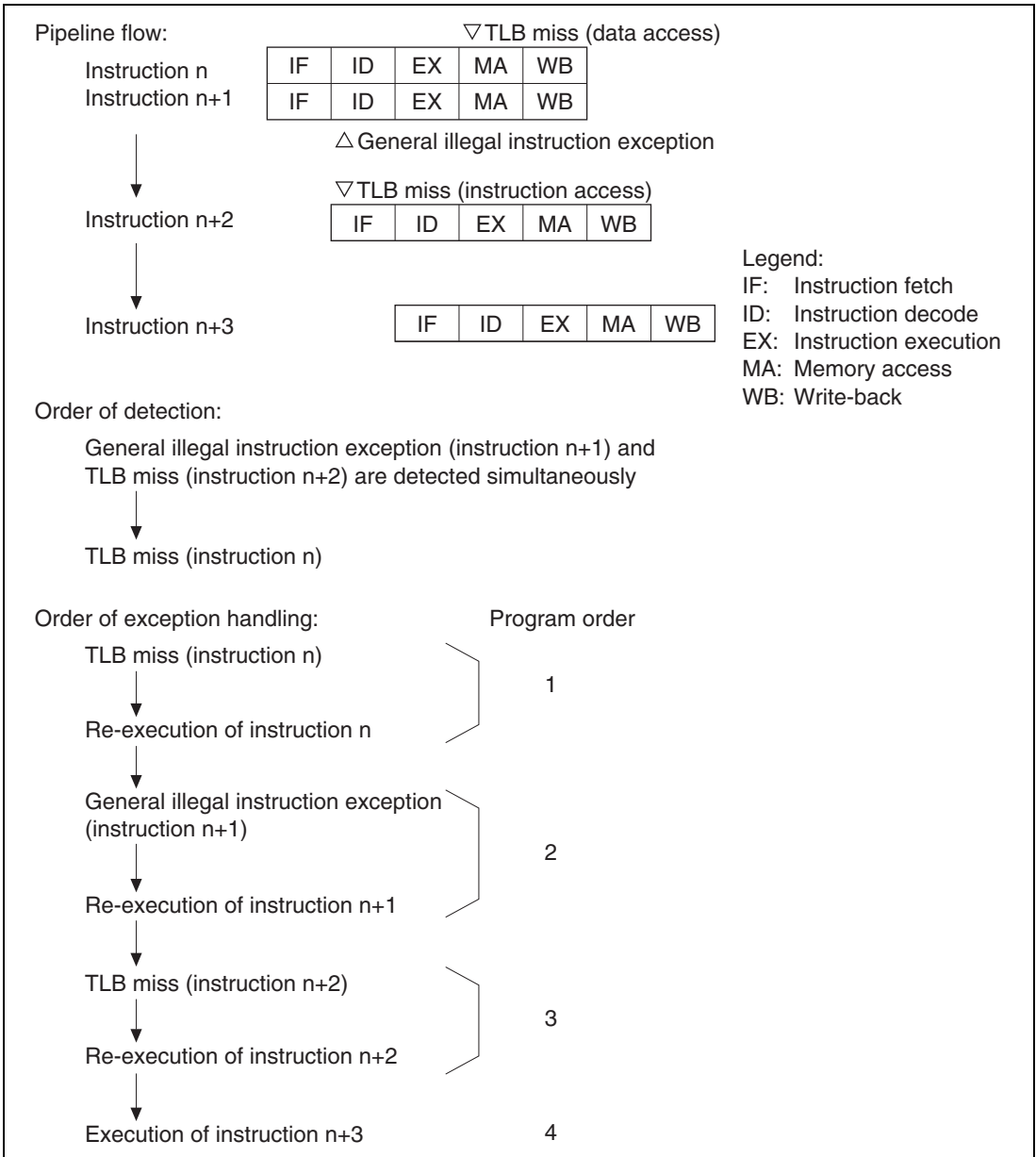


Figure 5.3 Example of General Exception Acceptance Order

5.5.3 Exception Requests and BL Bit

When the BL bit in SR is 0, general exceptions and interrupts are accepted.

When the BL bit in SR is 1 and a general exception other than a user break is generated, the CPU's internal registers and the registers of the other modules are set to their post-reset state, and the CPU branches to the same address as in a reset (H'A000 0000). For the operation in the event of a user break, see section 20, User Break Controller (UBC).

If an ordinary interrupt occurs, the interrupt request is held pending and is accepted after the BL bit has been cleared to 0 by software. If a nonmaskable interrupt (NMI) occurs, it can be held pending or accepted according to the setting made by software.

Thus, normally, SPC and SSR are saved and then the BL bit in SR is cleared to 0, to enable multiple exception state acceptance.

5.5.4 Return from Exception Handling

The RTE instruction is used to return from exception handling. When the RTE instruction is executed, the SPC contents are restored to PC and the SSR contents to SR, and the CPU returns from the exception handling routine by branching to the SPC address. If SPC and SSR were saved to external memory, set the BL bit in SR to 1 before restoring the SPC and SSR contents and issuing the RTE instruction.

5.6 Description of Exceptions

The various exception handling operations are described here, covering exception sources, transition addresses, and processor operation when a transition is made.

5.6.1 Resets

(1) Power-On Reset

- Sources:
 - $\overline{\text{RESET}}$ pin low level
 - When the watchdog timer overflows while the $\overline{\text{WT/IT}}$ bit is set to 1 and the RSTS bit is cleared to 0 in WTCSR. For details, see section 10, Clock Oscillation Circuits.
- Transition address: H'A000 0000
- Transition operations:

Exception code H'000 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (IMASK) are set to B'1111.

CPU and on-chip peripheral module initialization is performed. For details, see the register descriptions in the relevant sections. For some CPU functions, the $\overline{\text{TRST}}$ pin and $\overline{\text{RESET}}$ pin must be driven low. It is therefore essential to execute a power-on reset and drive the $\overline{\text{TRST}}$ pin low when powering on.

If the $\overline{\text{RESET}}$ pin is driven high before the $\overline{\text{MRESET}}$ pin while both these pins are low, a manual reset may occur after the power-on reset operation. The $\overline{\text{RESET}}$ pin must be driven high at the same time as, or after, the $\overline{\text{MRESET}}$ pin.

```
Power_on_reset()
{
    EXPEVT = H'00000000;
    VBR = H'00000000;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    SR.IMASK = B'1111;
    SR.FD=0;
    Initialize_CPU();
    Initialize_Module(PowerOn);
    PC = H'A0000000;
}
```

(2) Manual Reset

- Sources:
 - $\overline{\text{MRESET}}$ pin low level and $\overline{\text{RESET}}$ pin high level
 - When a general exception other than a user break occurs while the BL bit is set to 1 in SR
 - When the watchdog timer overflows while the RSTS bit is set to 1 in WTCSR. For details, see section 10, Clock Oscillation Circuits.

- Transition address: H'A000 0000

- Transition operations:

Exception code H'020 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (IMASK) are set to B'1111.

CPU and on-chip peripheral module initialization is performed. For details, see the register descriptions in the relevant sections.

```
Manual_reset()
{
    EXPEVT = H'00000020;
    VBR = H'00000000;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    SR.IMASK = B'1111;
    SR.FD = 0;
    Initialize_CPU();
    Initialize_Module(Manual);
    PC = H'A0000000;
}
```

Table 5.3 Types of Reset

| Type | Reset State Transition Conditions | | Internal States | |
|----------------|-----------------------------------|---------------------------|-----------------|--|
| | $\overline{\text{MRESET}}$ | $\overline{\text{RESET}}$ | CPU | On-Chip Peripheral Modules |
| Power-on reset | — | Low | Initialized | See Register Configuration in each section |
| Manual reset | Low | High | Initialized | |

(3) H-UDI Reset

- Source: SDIR.TI3–TI0 = B'0110 (negation) or B'0111 (assertion)
- Transition address: H'A000 0000
- Transition operations:

Exception code H'000 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (IMASK) are set to B'1111.

CPU and on-chip peripheral module initialization is performed. For details, see the register descriptions in the relevant sections.

```
H-UDI_reset()  
{  
    EXPEVT = H'00000000;  
    VBR = H'00000000;  
    SR.MD = 1;  
    SR.RB = 1;  
    SR.BL = 1;  
    SR.IMASK = B'1111;  
    SR.FD = 0;  
    Initialize_CPU();  
    Initialize_Module(PowerOn);  
    PC = H'A0000000;  
}
```

(4) Instruction TLB Multiple-Hit Exception

- Source: Multiple ITLB address matches
- Transition address: H'A000 0000
- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

Exception code H'140 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (IMASK) are set to B'1111.

CPU and on-chip peripheral module initialization is performed in the same way as in a manual reset. For details, see the register descriptions in the relevant sections.

```
TLB_multi_hit()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    EXPEVT = H'00000140;
    VBR = H'00000000;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    SR.IMASK = B'1111;
    SR.FD = 0;
    Initialize_CPU();
    Initialize_Module(Manual);
    PC = H'A0000000;
}
```

(5) Data TLB Multiple-Hit Exception

- Source: Multiple UTLB address matches
- Transition address: H'A000 0000
- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

Exception code H'140 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (IMASK) are set to B'1111.

CPU and on-chip peripheral module initialization is performed in the same way as in a manual reset. For details, see the register descriptions in the relevant sections.

```
TLB_multi_hit()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    EXPEVT = H'00000140;
    VBR = H'00000000;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    SR.IMASK = B'1111;
    SR.FD = 0;
    Initialize_CPU();
    Initialize_Module(Manual);
    PC = H'A0000000;
}
```

5.6.2 General Exceptions

(1) Data TLB Miss Exception

- Source: Address mismatch in UTLB address comparison
- Transition address: VBR + H'0000 0400
- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'040 (for a read access) or H'060 (for a write access) is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0400.

To speed up TLB miss processing, the offset is separate from that of other exceptions.

```
Data_TLB_miss_exception()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = read_access ? H'00000040 : H'00000060;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000400;
}
```

(2) Instruction TLB Miss Exception

- Source: Address mismatch in ITLB address comparison
- Transition address: VBR + H'0000 0400
- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'040 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0400.

To speed up TLB miss processing, the offset is separate from that of other exceptions.

```
ITLB_miss_exception()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = H'00000040;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000400;
}
```

(3) Initial Page Write Exception

- Source: TLB is hit in a store access, but dirty bit D = 0
- Transition address: VBR + H'0000 0100
- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'080 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
Initial_write_exception()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = H'00000080;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```


(4) Data TLB Protection Violation Exception

- Source: The access does not accord with the UTLB protection information (PR bits) shown below.

| PR | Privileged Mode | User Mode |
|----|----------------------------|----------------------------|
| 00 | Only read access possible | Access not possible |
| 01 | Read/write access possible | Access not possible |
| 10 | Only read access possible | Only read access possible |
| 11 | Read/write access possible | Read/write access possible |

- Transition address: VBR + H'0000 0100
- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'0A0 (for a read access) or H'0C0 (for a write access) is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
Data_TLB_protection_violation_exception()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = read_access ? H'000000A0 : H'000000C0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(5) Instruction TLB Protection Violation Exception

- Source: The access does not accord with the ITLB protection information (PR bits) shown below.

| PR | Privileged Mode | User Mode |
|-----------|------------------------|---------------------|
| 0 | Access possible | Access not possible |
| 1 | Access possible | Access possible |

- Transition address: VBR + H'0000 0100

- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'0A0 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
ITLB_protection_violation_exception()
```

```
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = H'000000A0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(6) Data Address Error

- Sources:
 - Word data access from other than a word boundary ($2n + 1$)
 - Longword data access from other than a longword data boundary ($4n + 1$, $4n + 2$, or $4n + 3$)
 - Quadword data access from other than a quadword data boundary ($8n + 1$, $8n + 2$, $8n + 3$, $8n + 4$, $8n + 5$, $8n + 6$, or $8n + 7$)
 - Access to area H'8000 0000–H'FFFF FFFF in user mode
- Transition address: VBR + H'0000 0100

- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'0E0 (for a read access) or H'100 (for a write access) is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. For details, see section 3, Memory Management Unit (MMU).

```
Data_address_error()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = read_access? H'000000E0: H'00000100;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(7) Instruction Address Error

- Sources:
 - Instruction fetch from other than a word boundary (2n + 1)
 - Instruction fetch from area H'8000 0000–H'FFFF FFFF in user mode

- Transition address: VBR + H'0000 0100

- Transition operations:

The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

The PC and SR contents for the instruction at which this exception occurred are saved in the SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'0E0 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. For details, see section 3, Memory Management Unit (MMU).

```
Instruction_address_error()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = H'000000E0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(8) Unconditional Trap

- Source: Execution of TRAPA instruction
- Transition address: VBR + H'0000 0100
- Transition operations:

As this is a processing-completion-type exception, the PC contents for the instruction following the TRAPA instruction are saved in SPC. The value of SR and R15 when the TRAPA instruction is executed are saved in SSR and SGR. The 8-bit immediate value in the TRAPA instruction is multiplied by 4, and the result is set in TRA [9:0]. Exception code H'160 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
TRAPA_exception()  
{  
    SPC = PC + 2;  
    SSR = SR;  
    SGR = R15;  
    TRA = imm << 2;  
    EXPEVT = H'00000160;  
    SR.MD = 1;  
    SR.RB = 1;  
    SR.BL = 1;  
    PC = VBR + H'00000100;  
}
```

(9) General Illegal Instruction Exception

- Sources:

- Decoding of an undefined instruction not in a delay slot

Delayed branch instructions: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT/S, BF/S

Undefined instruction: H'FFFD

- Decoding in user mode of a privileged instruction not in a delay slot

Privileged instructions: LDC, STC, RTE, LDTLB, SLEEP, but excluding LDC/STC instructions that access GBR

- Transition address: VBR + H'0000 0100

- Transition operations:

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'180 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. Operation is not guaranteed if an undefined code other than H'FFFD is decoded.

```
General_illegal_instruction_exception()
```

```
{  
    SPC = PC;  
    SSR = SR;  
    SGR = R15;  
    EXPEVT = H'00000180;  
    SR.MD = 1;  
    SR.RB = 1;  
    SR.BL = 1;  
    PC = VBR + H'00000100;  
}
```

(10) Slot Illegal Instruction Exception

- Sources:
 - Decoding of an undefined instruction in a delay slot
 - Delayed branch instructions: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT/S, BF/S
 - Undefined instruction: H'FFFD
 - Decoding of an instruction that modifies PC in a delay slot
 - Instructions that modify PC: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT, BF, BT/S, BF/S, TRAPA, LDC Rm,SR, LDC.L @Rm+,SR
 - Decoding in user mode of a privileged instruction in a delay slot
 - Privileged instructions: LDC, STC, RTE, LDTLB, SLEEP, but excluding LDC/STC instructions that access GBR
 - Decoding of a PC-relative MOV instruction or MOVA instruction in a delay slot
- Transition address: VBR + H'0000 0100
- Transition operations:

The PC contents for the preceding delayed branch instruction are saved in SPC. The SR and R15 contents when this exception occurred are saved in SSR and SGR.

Exception code H'1A0 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. Operation is not guaranteed if an undefined code other than H'FFFD is decoded.

```
Slot_illegal_instruction_exception()
{
    SPC = PC - 2;
    SSR = SR;
    SGR = R15;
    EXPEVT = H'000001A0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(11) General FPU Disable Exception

- Source: Decoding of an FPU instruction* not in a delay slot with SR.FD =1
- Transition address: VBR + H'0000 0100
- Transition operations:

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'800 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

Note: * FPU instructions are instructions in which the first 4 bits of the instruction code are H'F (but excluding undefined instruction H'FFFD), and the LDS, STS, LDS.L, and STS.L instructions corresponding to FPUL and FPSCR.

```
General_fpu_disable_exception()
{
    SPC = PC;
    SSR = SR;
    SGR = R15;
    EXPEVT = H'00000800;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```


(12) Slot FPU Disable Exception

- Source: Decoding of an FPU instruction in a delay slot with SR.FD =1
- Transition address: VBR + H'0000 0100
- Transition operations:

The PC contents for the preceding delayed branch instruction are saved in SPC. The SR and R15 contents when this exception occurred are saved in SSR and SGR.

Exception code H'820 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
Slot_fpu_disable_exception()  
{  
    SPC = PC - 2;  
    SSR = SR;  
    SGR = R15;  
    EXPEVT = H'00000820;  
    SR.MD = 1;  
    SR.RB = 1;  
    SR.BL = 1;  
    PC = VBR + H'00000100;  
}
```

(13) User Breakpoint Trap

- Source: Fulfilling of a break condition set in the user break controller
- Transition address: VBR + H'0000 0100, or DBR
- Transition operations:

In the case of a post-execution break, the PC contents for the instruction following the instruction at which the breakpoint is set are set in SPC. In the case of a pre-execution break, the PC contents for the instruction at which the breakpoint is set are set in SPC.

The SR and R15 contents when the break occurred are saved in SSR and SGR. Exception code H'1E0 is set in EXPEVT.

The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. It is also possible to branch to PC = DBR.

For details of PC, etc., when a data break is set, see section 20, User Break Controller (UBC).

```
User_break_exception()
{
    SPC = (pre_execution break? PC : PC + 2);
    SSR = SR;
    SGR = R15;
    EXPEVT = H'000001E0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = (BRCR.UBDE==1 ? DBR : VBR + H'00000100);
}
```

(14) FPU Exception

- Source: Exception due to execution of a floating-point operation
- Transition address: VBR + H'0000 0100
- Transition operations:

The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. The R15 contents at this time are saved in SGR. Exception code H'120 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
FPU_exception()  
{  
    SPC = PC;  
    SSR = SR;  
    SGR = R15;  
    EXPEVT = H'00000120;  
    SR.MD = 1;  
    SR.RB = 1;  
    SR.BL = 1;  
    PC = VBR + H'00000100;  
}
```

5.6.3 Interrupts

(1) NMI

- Source: NMI pin edge detection
- Transition address: VBR + H'0000 0600
- Transition operations:

The PC and SR contents for the instruction at which this exception is accepted are saved in SPC and SSR. The R15 contents at this time are saved in SGR.

Exception code H'1C0 is set in INTEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0600. When the BL bit in SR is 0, this interrupt is not masked by the interrupt mask bits in SR, and is accepted at the highest priority level. When the BL bit in SR is 1, a software setting can specify whether this interrupt is to be masked or accepted. For details, see section 19, Interrupt Controller (INTC).

NMI ()

```
{  
    SPC = PC;  
    SSR = SR;  
    SGR = R15;  
    INTEVT = H'000001C0;  
    SR.MD = 1;  
    SR.RB = 1;  
    SR.BL = 1;  
    PC = VBR + H'00000600;  
}
```

(2) IRL Interrupts

- Source: The interrupt mask bit setting in SR is smaller than the IRL (3–0) level, and the BL bit in SR is 0 (accepted at instruction boundary).
- Transition address: VBR + H'0000 0600
- Transition operations:

The PC contents immediately after the instruction at which the interrupt is accepted are set in SPC. The SR and R15 contents at the time of acceptance are set in SSR and SGR.

The code corresponding to the IRL (3–0) level is set in INTEVT. See table 19.4, for the corresponding codes. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to VBR + H'0600. The acceptance level is not set in the interrupt mask bits in SR. When the BL bit in SR is 1, the interrupt is masked. For details, see section 19, Interrupt Controller (INTC).

```
IRL ()
{
    SPC = PC;
    SSR = SR;
    SGR = R15;
    INTEVT = H'00000200 ~ H'000003C0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000600;
}
```

(3) Peripheral Module Interrupts

- Source: The interrupt mask bit setting in SR is smaller than the peripheral module (H-UDI, GPIO, DMAC, PCIC, TMU, RTC, SCI, SCIF, WDT, or REF) interrupt level, and the BL bit in SR is 0 (accepted at instruction boundary).
- Transition address: VBR + H'0000 0600
- Transition operations:

The PC contents immediately after the instruction at which the interrupt is accepted are set in SPC. The SR and R15 contents at the time of acceptance are set in SSR and SGR.

The code corresponding to the interrupt source is set in INTEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to VBR + H'0600. The module interrupt levels should be set as values between B'0000 and B'1111 in the interrupt priority registers (IPRA–IPRC) in the interrupt controller. For details, see section 19, Interrupt Controller (INTC).

```
Module_interruption()
{
    SPC = PC;
    SSR = SR;
    SGR = R15;
    INTEVT = H'00000400 ~ H'00000B40;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000600;
}
```

5.6.4 Priority Order with Multiple Exceptions

With some instructions, such as instructions that make two accesses to memory, and the indivisible pair comprising a delayed branch instruction and delay slot instruction, multiple exceptions occur. Care is required in these cases, as the exception priority order differs from the normal order.

1. Instructions that make two accesses to memory

With MAC instructions, memory-to-memory arithmetic/logic instructions, and TAS instructions, two data transfers are performed by a single instruction, and an exception will be detected for each of these data transfers. In these cases, therefore, the following order is used to determine priority.

- a. Data address error in first data transfer
- b. TLB miss in first data transfer
- c. TLB protection violation in first data transfer
- d. Initial page write exception in first data transfer
- e. Data address error in second data transfer
- f. TLB miss in second data transfer
- g. TLB protection violation in second data transfer
- h. Initial page write exception in second data transfer

2. Indivisible delayed branch instruction and delay slot instruction

As a delayed branch instruction and its associated delay slot instruction are indivisible, they are treated as a single instruction. Consequently, the priority order for exceptions that occur in these instructions differs from the usual priority order. The priority order shown below is for the case where the delay slot instruction has only one data transfer.

- a. A check is performed for the abort type and reexecution type exceptions of priority levels 1 and 2 in the delayed branch instruction.
- b. A check is performed for the abort type and reexecution type exceptions of priority levels 1 and 2 in the delay slot instruction.
- c. A check is performed for the completion type exception of priority level 2 in the delayed branch instruction.
- d. A check is performed for the completion type exception of priority level 2 in the delay slot instruction.
- e. A check is performed for priority level 3 in the delayed branch instruction and priority level 3 in the delay slot instruction. (There is no priority ranking between these two.)
- f. A check is performed for priority level 4 in the delayed branch instruction and priority level 4 in the delay slot instruction. (There is no priority ranking between these two.)

If the delay slot instruction has a second data transfer, two checks are performed in step b, as in 1 above.

If the accepted exception (the highest-priority exception) is a delay slot instruction re-execution type exception, the branch instruction PR register write operation (PC → PR operation performed in BSR, BSRF, JSR) is not inhibited.

5.7 Usage Notes

1. Return from exception handling
 - a. Check the BL bit in SR with software. If SPC and SSR have been saved to external memory, set the BL bit in SR to 1 before restoring them.
 - b. Issue an RTE instruction. When RTE is executed, the SPC contents are set in PC, the SSR contents are set in SR, and branch is made to the SPC address to return from the exception handling routine.
2. If a general exception or interrupt occurs when SR.BL = 1
 - a. General exception

When a general exception other than a user break occurs, manual reset occurs. The value in EXPEVT at this time is H'0000 0020; the value of the SPC and SSR registers is undefined.
 - b. Interrupt

If an ordinary interrupt occurs, the interrupt request is held pending and is accepted after the BL bit in SR has been cleared to 0 by software. If a nonmaskable interrupt (NMI) occurs, it can be held pending or accepted according to the setting made by software. In the sleep or standby state, however, an interrupt is accepted even if the BL bit in SR is set to 1.
3. SPC when an exception occurs
 - a. Re-execution type general exception

The PC value for the instruction in which the general exception occurred is set in SPC, and the instruction is re-executed after returning from exception handling. If an exception occurs in a delay slot instruction, however, the PC value for the delay slot instruction is saved in SPC regardless of whether or not the preceding delayed branch instruction condition is satisfied.
 - b. Completion type general exception or interrupt

The PC value for the instruction following that in which the general exception occurred is set in SPC. If an exception occurs in a branch instruction with delay slot, however, the PC value for the branch destination is saved in SPC.
4. An exception must not be generated in an RTE instruction delay slot, as the operation will be undefined in this case.

5.8 Restrictions

1. Restrictions on first instruction of exception handling routine
 - Do not locate a BT, BF, BT/S, BF/S, BRA, or BSR instruction at address VBR + H'100, VBR + H'400, or VBR + H'600.
 - When the UBDE bit in the BRCCR register is set to 1 and the user break debug support function* is used, do not locate a BT, BF, BT/S, BF/S, BRA, or BSR instruction at the address indicated by the DBR register.

Note: * See section 20.4, User Break Debug Support Function.

Section 6 Floating-Point Unit

6.1 Overview

The floating-point unit (FPU) has the following features:

- Conforms to IEEE754 standard
- 32 single-precision floating-point registers (can also be referenced as 16 double-precision registers)
- Two rounding modes: Round to Nearest and Round to Zero
- Two denormalization modes: Flush to Zero and Treat Denormalized Number
- Six exception sources: FPU Error, Invalid Operation, Divide By Zero, Overflow, Underflow, and Inexact
- Comprehensive instructions: Single-precision, double-precision, graphics support, system control

When the FD bit in SR is set to 1, the FPU cannot be used, and an attempt to execute an FPU instruction will cause an FPU disable exception.

6.2 Data Formats

6.2.1 Floating-Point Format

A floating-point number consists of the following three fields:

- Sign (s)
- Exponent (e)
- Fraction (f)

The FPU can handle single-precision and double-precision floating-point numbers, using the formats shown in figures 6.1 and 6.2.

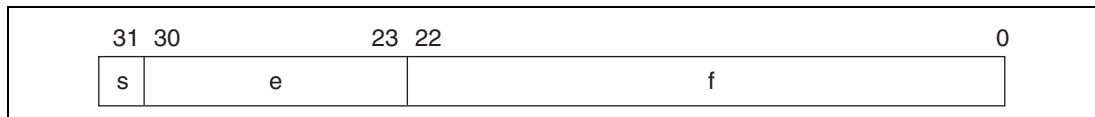


Figure 6.1 Format of Single-Precision Floating-Point Number

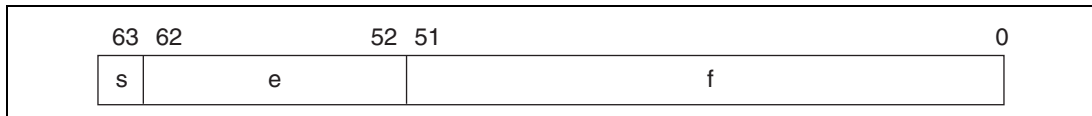


Figure 6.2 Format of Double-Precision Floating-Point Number

The exponent is expressed in biased form, as follows:

$$e = E + \text{bias}$$

The range of unbiased exponent E is $E_{\min} - 1$ to $E_{\max} + 1$. The two values $E_{\min} - 1$ and $E_{\max} + 1$ are distinguished as follows. $E_{\min} - 1$ indicates zero (both positive and negative sign) and a denormalized number, and $E_{\max} + 1$ indicates positive or negative infinity or a non-number (NaN). Table 6.1 shows bias, E_{\min} , and E_{\max} values.

Table 6.1 Floating-Point Number Formats and Parameters

| Parameter | Single-Precision | Double-Precision |
|-----------------|------------------|------------------|
| Total bit width | 32 bits | 64 bits |
| Sign bit | 1 bit | 1 bit |
| Exponent field | 8 bits | 11 bits |
| Fraction field | 23 bits | 52 bits |
| Precision | 24 bits | 53 bits |
| Bias | +127 | +1023 |
| E_{\max} | +127 | +1023 |
| E_{\min} | -126 | -1022 |

Floating-point number value v is determined as follows:

If $E = E_{\max} + 1$ and $f \neq 0$, v is a non-number (NaN) irrespective of sign s

If $E = E_{\max} + 1$ and $f = 0$, $v = (-1)^s$ (infinity) [positive or negative infinity]

If $E_{\min} \leq E \leq E_{\max}$, $v = (-1)^s 2^E (1.f)$ [normalized number]

If $E = E_{\min} - 1$ and $f \neq 0$, $v = (-1)^s 2^{E_{\min}} (0.f)$ [denormalized number]

If $E = E_{\min} - 1$ and $f = 0$, $v = (-1)^s 0$ [positive or negative zero]

Table 6.2 shows the ranges of the various numbers in hexadecimal notation.

Table 6.2 Floating-Point Ranges

| Type | Single-Precision | Double-Precision |
|------------------------------|----------------------------|---|
| Signaling non-number | H'7FFFFFFF to H'7FC00000 | H'7FFFFFFF FFFFFFFF to H'7FF80000 00000000 |
| Quiet non-number | H'7FBFFFFFFF to H'7F800001 | H'7FF7FFFF FFFFFFFF to H'7FF00000 00000001 |
| Positive infinity | H'7F800000 | H'7FF00000 00000000 |
| Positive normalized number | H'7F7FFFFFFF to H'00800000 | H'7FEFFFFFFF FFFFFFFF to H'00100000 00000000 |
| Positive denormalized number | H'007FFFFFFF to H'00000001 | H'000FFFFFFF FFFFFFFF to H'00000000 00000001 |
| Positive zero | H'00000000 | H'00000000 00000000 |
| Negative zero | H'80000000 | H'80000000 00000000 |
| Negative denormalized number | H'80000001 to H'807FFFFFFF | H'80000000 00000001 to H'800FFFFFFF FFFFFFFF |
| Negative normalized number | H'80800000 to H'FF7FFFFFFF | H'80100000 00000000 to H'FFEFFFFFFF FFFFFFFF |
| Negative infinity | H'FF800000 | H'FFF00000 00000000 |
| Quiet non-number | H'FF800001 to H'FFBFFFFFFF | H'FFF00000 00000001 to H'FFF7FFFF FFFFFFFF |
| Signaling non-number | H'FFC00000 to H'FFFFFFFF | H'FFF80000 00000000 to H'FFFFFFFF FFFFFFFF |

6.2.2 Non-Numbers (NaN)

Figure 6.3 shows the bit pattern of a non-number (NaN). A value is NaN in the following case:

- Sign bit: Don't care
- Exponent field: All bits are 1
- Fraction field: At least one bit is 1

The NaN is a signaling NaN (sNaN) if the MSB of the fraction field is 1, and a quiet NaN (qNaN) if the MSB is 0.

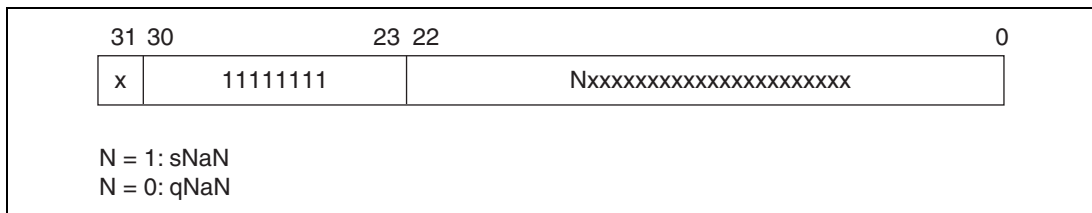


Figure 6.3 Single-Precision NaN Bit Pattern

An sNaN is input in an operation, except copy, FABS, and FNEG, that generates a floating-point value.

- When the EN.V bit in the FPSCR register is 0, the operation result (output) is a qNaN.
- When the EN.V bit in the FPSCR register is 1, an invalid operation exception will be generated. In this case, the contents of the operation destination register are unchanged.

If a qNaN is input in an operation that generates a floating-point value, and an sNaN has not been input in that operation, the output will always be a qNaN irrespective of the setting of the EN.V bit in the FPSCR register. An exception will not be generated in this case.

The qNaN values generated by the FPU as operation results are as follows:

- Single-precision qNaN: H'7FBFFFFF
- Double-precision qNaN: H'7FF7FFFF FFFFFFFF

See the individual instruction descriptions for details of floating-point operations when a non-number (NaN) is input.

6.2.3 Denormalized Numbers

For a denormalized number floating-point value, the exponent field is expressed as 0, and the fraction field as a non-zero value.

When the DN bit in the FPU's status register FPSCR is 1, a denormalized number (source operand or operation result) is always flushed to 0 in a floating-point operation that generates a value (an operation other than copy, FNEG, or FABS).

When the DN bit in FPSCR is 0, a denormalized number (source operand or operation result) is processed as it is. See the individual instruction descriptions for details of floating-point operations when a denormalized number is input.

6.3 Registers

6.3.1 Floating-Point Registers

Figure 6.4 shows the floating-point register configuration. There are thirty-two 32-bit floating-point registers, referenced by specifying FR0–FR15, DR0/2/4/6/8/10/12/14, FV0/4/8/12, XF0–XF15, XD0/2/4/6/8/10/12/14, or XMTRX.

1. Floating-point registers, FPR_i_BANK_j (32 registers)
 FPR0_BANK0–FPR15_BANK0
 FPR0_BANK1–FPR15_BANK1
2. Single-precision floating-point registers, FR_i (16 registers)
 When FPSCR.FR = 0, FR0–FR15 indicate FPR0_BANK0–FPR15_BANK0;
 when FPSCR.FR = 1, FR0–FR15 indicate FPR0_BANK1–FPR15_BANK1.
3. Double-precision floating-point registers, DR_i (8 registers): A DR register comprises two FR registers
 DR0 = {FR0, FR1}, DR2 = {FR2, FR3}, DR4 = {FR4, FR5}, DR6 = {FR6, FR7},
 DR8 = {FR8, FR9}, DR10 = {FR10, FR11}, DR12 = {FR12, FR13}, DR14 = {FR14, FR15}
4. Single-precision floating-point vector registers, FV_i (4 registers): An FV register comprises four FR registers
 FV0 = {FR0, FR1, FR2, FR3}, FV4 = {FR4, FR5, FR6, FR7},
 FV8 = {FR8, FR9, FR10, FR11}, FV12 = {FR12, FR13, FR14, FR15}
5. Single-precision floating-point extended registers, XF_i (16 registers)
 When FPSCR.FR = 0, XF0–XF15 indicate FPR0_BANK1–FPR15_BANK1;
 when FPSCR.FR = 1, XF0–XF15 indicate FPR0_BANK0–FPR15_BANK0.
6. Double-precision floating-point extended registers, XD_i (8 registers): An XD register comprises two XF registers
 XD0 = {XF0, XF1}, XD2 = {XF2, XF3}, XD4 = {XF4, XF5}, XD6 = {XF6, XF7},
 XD8 = {XF8, XF9}, XD10 = {XF10, XF11}, XD12 = {XF12, XF13}, XD14 = {XF14, XF15}

7. Single-precision floating-point extended register matrix: XMTRX

XMTRX comprises all 16 XF registers

$$\text{XMTRX} = \begin{bmatrix} \text{XF0} & \text{XF4} & \text{XF8} & \text{XF12} \\ \text{XF1} & \text{XF5} & \text{XF9} & \text{XF13} \\ \text{XF2} & \text{XF6} & \text{XF10} & \text{XF14} \\ \text{XF3} & \text{XF7} & \text{XF11} & \text{XF15} \end{bmatrix}$$

| <u>FPSCR.FR = 0</u> | | | | <u>FPSCR.FR = 1</u> | | |
|---------------------|------|------|-------------|---------------------|------|-------|
| FV0 | DR0 | FR0 | FPR0_BANK0 | XF0 | XD0 | XMTRX |
| | | FR1 | FPR1_BANK0 | XF1 | | |
| | DR2 | FR2 | FPR2_BANK0 | XF2 | XD2 | |
| | | FR3 | FPR3_BANK0 | XF3 | | |
| FV4 | DR4 | FR4 | FPR4_BANK0 | XF4 | XD4 | |
| | | FR5 | FPR5_BANK0 | XF5 | | |
| | DR6 | FR6 | FPR6_BANK0 | XF6 | XD6 | |
| | | FR7 | FPR7_BANK0 | XF7 | | |
| FV8 | DR8 | FR8 | FPR8_BANK0 | XF8 | XD8 | |
| | | FR9 | FPR9_BANK0 | XF9 | | |
| | DR10 | FR10 | FPR10_BANK0 | XF10 | XD10 | |
| | | FR11 | FPR11_BANK0 | XF11 | | |
| FV12 | DR12 | FR12 | FPR12_BANK0 | XF12 | XD12 | |
| | | FR13 | FPR13_BANK0 | XF13 | | |
| | DR14 | FR14 | FPR14_BANK0 | XF14 | XD14 | |
| | | FR15 | FPR15_BANK0 | XF15 | | |
| XMTRX | XD0 | XF0 | FPR0_BANK1 | FR0 | DR0 | FV0 |
| | | XF1 | FPR1_BANK1 | FR1 | | |
| | XD2 | XF2 | FPR2_BANK1 | FR2 | DR2 | |
| | | XF3 | FPR3_BANK1 | FR3 | | |
| | XD4 | XF4 | FPR4_BANK1 | FR4 | DR4 | FV4 |
| | | XF5 | FPR5_BANK1 | FR5 | | |
| | XD6 | XF6 | FPR6_BANK1 | FR6 | DR6 | |
| | | XF7 | FPR7_BANK1 | FR7 | | |
| | XD8 | XF8 | FPR8_BANK1 | FR8 | DR8 | FV8 |
| | | XF9 | FPR9_BANK1 | FR9 | | |
| | XD10 | XF10 | FPR10_BANK1 | FR10 | DR10 | |
| | | XF11 | FPR11_BANK1 | FR11 | | |
| | XD12 | XF12 | FPR12_BANK1 | FR12 | DR12 | FV12 |
| | | XF13 | FPR13_BANK1 | FR13 | | |
| | XD14 | XF14 | FPR14_BANK1 | FR14 | DR14 | |
| | | XF15 | FPR15_BANK1 | FR15 | | |

Figure 6.4 Floating-Point Registers

6.3.2 Floating-Point Status/Control Register (FPSCR)

Floating-point status/control register, FPSCR (32 bits, initial value = H'0004 0001)

| | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|-------|--------|------|----|---|---|
| 31 | 22 | 21 | 20 | 19 | 18 | 17 | 12 | 11 | 7 | 6 | 2 | 1 | 0 |
| — | | | | FR | SZ | PR | DN | Cause | Enable | Flag | RM | | |

Note: —: Reserved. These bits are always read as 0, and should only be written with 0.

- **FR: Floating-point register bank**
FR = 0: FPR0_BANK0–FPR15_BANK0 are assigned to FR0–FR15; FPR0_BANK1–FPR15_BANK1 are assigned to XF0–XF15.
FR = 1: FPR0_BANK0–FPR15_BANK0 are assigned to XF0–XF15; FPR0_BANK1–FPR15_BANK1 are assigned to FR0–FR15.
- **SZ: Transfer size mode**
SZ = 0: The data size of the FMOV instruction is 32 bits.
SZ = 1: The data size of the FMOV instruction is a 32-bit register pair (64 bits).
- **PR: Precision mode**
PR = 0: Floating-point instructions are executed as single-precision operations.
PR = 1: Floating-point instructions are executed as double-precision operations (graphics support instructions are undefined).
Do not set SZ and PR to 1 simultaneously; this setting is reserved.
[SZ, PR = 11]: Reserved (FPU operation instruction is undefined.)
- **DN: Denormalization mode**
DN = 0: A denormalized number is treated as such.
DN = 1: A denormalized number is treated as zero.
- **Cause: FPU exception cause field**
- **Enable: FPU exception enable field**

- Flag: FPU exception flag field

| | | FPU Error (E) | Invalid Operation (V) | Division by Zero (Z) | Overflow (O) | Underflow (U) | Inexact (I) |
|--------|----------------------------|--------------------------|----------------------------------|---------------------------------|-------------------------|--------------------------|------------------------|
| Cause | FPU exception cause field | Bit 17 | Bit 16 | Bit 15 | Bit 14 | Bit 13 | Bit 12 |
| Enable | FPU exception enable field | None | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 |
| Flag | FPU exception flag field | None | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 |

When an FPU operation instruction is executed, the FPU exception cause field is cleared to zero first. When the next FPU exception is occurred, the corresponding bits in the FPU exception cause field and FPU exception flag field are set to 1. The FPU exception flag field holds the status of the exception generated after the field was last cleared.

- RM: Rounding mode

RM = 00: Round to Nearest

RM = 01: Round to Zero

RM = 10: Reserved

RM = 11: Reserved

- Bits 22 to 31: Reserved

These bits are always read as 0, and should only be written with 0.

6.3.3 Floating-Point Communication Register (FPUL)

Information is transferred between the FPU and CPU via the FPUL register. The 32-bit FPUL register is a system register, and is accessed from the CPU side by means of LDS and STS instructions. For example, to convert the integer stored in general register R1 to a single-precision floating-point number, the processing flow is as follows:

R1 → (LDS instruction) → FPUL → (single-precision FLOAT instruction) → FR1

6.4 Rounding

In a floating-point instruction, rounding is performed when generating the final operation result from the intermediate result. Therefore, the result of combination instructions such as FMAC, FTRV, and FIPR will differ from the result when using a basic instruction such as FADD, FSUB, or FMUL. Rounding is performed once in FMAC, but twice in FADD, FSUB, and FMUL.

There are two rounding methods, the method to be used being determined by the RM field in FPSCR.

- RM = 00: Round to Nearest
- RM = 01: Round to Zero

Round to Nearest: The operation result is rounded to the nearest expressible value. If there are two nearest expressible values, the one with an LSB of 0 is selected.

If the unrounded value is $2^{\text{E}_{\text{max}}}(2 - 2^{-\text{P}})$ or more, the result will be infinity with the same sign as the unrounded value. The values of E_{max} and P, respectively, are 127 and 24 for single-precision, and 1023 and 53 for double-precision.

Round to Zero: The digits below the round bit of the unrounded value are discarded.

If the unrounded value is larger than the maximum expressible absolute value, the value will be the maximum expressible absolute value.

6.5 Floating-Point Exceptions

FPU-related exceptions are as follows:

- General illegal instruction/slot illegal instruction exception
The exception occurs if an FPU instruction is executed when SR.FD = 1.
- FPU exceptions
The exception sources are as follows:
 - FPU error (E): When FPSCR.DN = 0 and a denormalized number is input
 - Invalid operation (V): In case of an invalid operation, such as NaN input
 - Division by zero (Z): Division with a zero divisor
 - Overflow (O): When the operation result overflows
 - Underflow (U): When the operation result underflows
 - Inexact exception (I): When overflow, underflow, or rounding occurs

The FPSCR FPU exception cause field contains bits corresponding to all of above E, V, Z, O, U, and I, and the FPSCR flag and enable fields contain bits corresponding to V, Z, O, U, and I, but not E. Thus, FPU errors cannot be disabled.

When an FPU exception occurs, the corresponding bit in the FPU exception cause field is set to 1, and 1 is added to the corresponding bit in the FPU exception flag field. When an FPU exception does not occur, the corresponding bit in the FPU exception cause field is cleared to 0, but the corresponding bit in the FPU exception flag field remains unchanged.

- Enable/disable exception handling

The FPU supports enable exception handling and disable exception handling.

Enable exception handling is initiated in the following cases:

- FPU error (E): FPSCR.DN = 0 and a denormalized number is input
- Invalid operation (V): FPSCR.EN.V = 1 and (instruction = FTRV or invalid operation)
- Division by zero (Z): FPSCR.EN.Z = 1 and division with a zero divisor
- Overflow (O): FPSCR.EN.O = 1 and instruction with possibility of operation result overflow
- Underflow (U): FPSCR.EN.U = 1 and instruction with possibility of operation result underflow
- Inexact exception (I): FPSCR.EN.I = 1 and instruction with possibility of inexact operation result

For information on possibilities (which differ depending on the individual instruction), see section 9, Instruction Descriptions, in the *SH-4 Software Manual*. All exception events that originate in the FPU are assigned as the same exception event. The meaning of an exception is determined by software by reading system register FPSCR and interpreting the information it contains. If no bits are set in the FPU exception cause field of FPSCR when one or more of bits O, U, I, and V (in case of FTRV only) are set in the FPU exception enable field, this indicates that an actual FPU exception is not generated. Also, the destination register is not changed by any FPU exception handling operation.

Except for the above, the bit corresponding to V, Z, O, U, or I is set to 1 in all processing, and the default value is generated as the result of the operation.

- Invalid operation (V): qNaN is generated as the result.
- Division by zero (Z): Infinity with the same sign as the unrounded value is generated.
- Overflow (O):

In round to zero mode, the maximum normalized number, with the same sign as the unrounded value, is generated.

In round to nearest mode, infinity with the same sign as the unrounded value is generated.

— Underflow (U):

When FPSCR.DN = 0, a denormalized number with the same sign as the unrounded value, or zero with the same sign as the unrounded value, is generated.

When FPSCR.DN = 1, zero with the same sign as the unrounded value, is generated.

— Inexact exception (I): An inexact result is generated.

6.6 Graphics Support Functions

The FPU supports two kinds of graphics functions: new instructions for geometric operations, and pair single-precision transfer instructions that enable high-speed data transfer.

6.6.1 Geometric Operation Instructions

Geometric operation instructions perform approximate-value computations. To enable high-speed computation with a minimum of hardware, the FPU ignores comparatively small values in the partial computation results of four multiplications. Consequently, the error shown below is produced in the result of the computation:

$$\text{Maximum error} = \text{MAX}(\text{individual multiplication result} \times 2^{-\text{MIN}(\text{number of multiplier significant digits}-1, \text{number of multiplicand significant digits}-1)}) + \text{MAX}(\text{result value} \times 2^{-23}, 2^{-149})$$

The number of significant digits is 24 for a normalized number and 23 for a denormalized number (number of leading zeros in the fractional part).

In future version of SuperH RISC engine Family, the above error is guaranteed, but the same result as SH7751 Group is not guaranteed.

FIPR FVm, FVn (m, n: 0, 4, 8, 12): Examples of the use of this instruction are given below.

- Inner product (m ≠ n):
This operation is generally used for surface/rear surface determination for polygon surfaces.
- Sum of square of elements (m = n):
This operation is generally used to find the length of a vector.

Since approximate-value computations are performed to enable high-speed computation, the inexact exception (I) bit in the FPU exception cause field and FPU exception flag field is always set to 1 when an FIPR instruction is executed. Therefore, if the corresponding bit is set in the FPU exception enable field, FPU exception handling will be executed.

FTRV XMTRX, FVn (n: 0, 4, 8, 12): Examples of the use of this instruction are given below.

- Matrix (4×4) · vector (4):

This operation is generally used for viewpoint changes, angle changes, or movements called vector transformations (4-dimensional). Since affine transformation processing for angle + parallel movement basically requires a 4×4 matrix, the FPU supports 4-dimensional operations.

- Matrix $(4 \times 4) \times$ matrix (4×4) :

This operation requires the execution of four FTRV instructions.

Since approximate-value computations are performed to enable high-speed computation, the inexact exception (I) bit in the FPU exception cause field and FPU exception flag field is always set to 1 when an FTRV instruction is executed. Therefore, if the corresponding bit is set in the FPU exception enable field, FPU exception handling will be executed. For the same reason, it is not possible to check all data types in the registers beforehand when executing an FTRV instruction. If the V bit is set in the FPU exception enable field, FPU exception handling will be executed.

FRCHG: This instruction modifies banked registers. For example, when the FTRV instruction is executed, matrix elements must be set in an array in the background bank. However, to create the actual elements of a translation matrix, it is easier to use registers in the foreground bank. When the LDC instruction is used on FPSCR, this instruction expends 4 to 5 cycles in order to maintain the FPU state. With the FRCHG instruction, an FPSCR.FR bit modification can be performed in one cycle.

6.6.2 Pair Single-Precision Data Transfer

The powerful geometric operation instructions, FPU also supports high-speed data transfer instructions.

When $FPSCR.SZ = 1$, FPU can perform data transfer by means of pair single-precision data transfer instructions.

- FMOV DRm/XDm, DRn/XDRn (m, n: 0, 2, 4, 6, 8, 10, 12, 14)
- FMOV DRm/XDm, @Rn (m: 0, 2, 4, 6, 8, 10, 12, 14; n: 0 to 15)

These instructions enable two single-precision (2×32 -bit) data items to be transferred; that is, the transfer performance of these instructions is doubled.

- **FSCHG**

This instruction changes the value of the SZ bit in FPSCR, enabling fast switching between use and non-use of pair single-precision data transfer.

Programming Note:

When FPSCR.SZ = 1 and big-endian mode is used, FMOV can be used for a double-precision floating-point load or store. In little-endian mode, a double-precision floating-point load or store requires execution of two 32-bit data size operations with FPSCR.SZ = 0.

6.7 Usage Notes

6.7.1 Rounding Mode and Underflow Flag

When using the Round to Nearest rounding mode, the underflow flag may not be set in cases defined as underflow by the IEEE754 standard.

Under the IEEE754 standard, when the Round to Nearest rounding mode is used and infinite-precision operation result x is (i) or (ii) (single-precision) or (iii) or (iv) (double-precision), there are cases where “the result after rounding is a normalized number, but an underflow results.”

In such cases where “the result after rounding is a normalized number, but an underflow results,” the FPU does not set the underflow flag to 1. In these cases the operation result, the value written to FRn, is correct. Also, if an FPU exception occurs, the underflow flag is not set to 1 but the inexact flag is set to 1 in such cases. Generation of FPU exceptions can be enabled by setting the enable field to 1.

- (i) $H'007FFFFFFF < x < H'00800000$
- (ii) $H'807FFFFFFF > x > H'80800000$
- (iii) $H'000FFFFFFF FFFFFFFF < x < H'00100000 00000000$
- (iv) $H'800FFFFFFF FFFFFFFF > x > H'80100000 00000000$

Examples

- Single-precision

When FPSCR.RM = 00 (Round to Nearest) and FPSCR.PR = 0 (single-precision), and the FMUL instruction ($H'00FFF000 * H'3F000800$) is executed.

a. According to IEEE754 standard

Operation result: $H'00800000$

FPSCR: $H'0004300C$

b. FPU

Operation result: H'00800000

FPSCR: H'00041004

• Double-precision

When FPSCR.RM = 00 (Round to Nearest) and FPSCR.PR = 1 (double-precision), and the FDIV instruction (H'001FFFFFF FFFFFFFF / H'40000000 00000000) is executed.

a. According to IEEE754 standard

Operation result: H'00100000 00000000

FPSCR: H'000C300C

b. FPU

Operation result: H'00100000 00000000

FPSCR: H'000C1004

Workarounds

1. Use FPSCR.RM = 01, that is to say Round to Zero rather than Round to Nearest mode.
2. Use FPSCR.RM = 00, that is to say Round to Nearest mode, and set the enable field to 1 to enable generation of inexact exceptions so that the exception handling routine can be used to check whether or not an underflow has occurred.

6.7.2 Setting of Overflow Flag by FIPR or FTRV Instruction

When the maximum error produced by the FIPR or FTRV instruction exceeds the maximum value expressible as a normalized number (H'7F7FFFFFF), the overflow flag may be set, even through the operation result is a positive or negative zero (H'00000000 or H'80000000).

Example: The operation result (FR7) after executing the instruction FIPR FV4, FV0 is H'00000000 (positive zero), but the overflow flag may be set nevertheless.

FPSCR = H'00040001

FR0 = H'FF7EF631 , FR1 = H80000000 , FR2 = H'8087F451 , FR3 = H'7F7EF631

FR4 = H'7F7EF631 , FR5 = H'0087F451 , FR6 = H'7F7EF631 , FR7 = H'7F7EF631

Workaround: Avoid using the FIPR and FTRV instructions, and use the FADD, FMUL, and FMAC instructions instead.

6.7.3 Sign of Operation Result when Using FIPR or FTRV Instruction

When two or more data items used in an operation by the FIPR or FTRV instruction are infinity, and all of the infinity items in the multiplication results have the same sign, the sign of the operation result may be incorrect.

Workarounds

1. Do not use infinity. If conditions (a) to (c) below are satisfied, infinity is never used in operations.
 - a. Use Round to Zero (FPSCR.RM = 01) as the rounding mode.
 - b. Do not divide by zero.
 - c. Do not transfer a value of positive or negative infinity to FR0 to FR15 or to XF0 to XF15.
2. Avoid using the FIPR and FTRV instructions, and use the FADD, FMUL, and FMAC instructions instead.

6.7.4 Notes on Double-Precision FADD and FSUB Instructions

Description: If the input data for a double-precision FADD instruction or a double-precision FSUB instruction satisfies all of the conditions listed below, the inexact bits (FPSCR.Flag.I and FPSCR.Cause.I) may not be set even through the operation result is inexact.

Condition 1: The operation instruction is a double-precision FADD instruction or a double-precision FSUB instruction.

Condition 2: The difference between the DRn and DRm exponents is between 43 and 50.

Condition 3: At least one of bits 31 to 24 of the mantissa portion of whichever of DRn and DRm has the smaller absolute value is 1.

Condition 4: Bits 23 to 0 of the mantissa portion of whichever of DRn and DRm has the smaller absolute value are all 0.

Condition 5: Bits 40 to 32 of the mantissa portion of whichever of DRn and DRm has the smaller absolute value are all 0.

In addition, the result of an operation meeting the above conditions may have a rounding error. Specifically, in a case where the closest expressible value less than the unrounded value should be selected, the closest expressible value greater than the unrounded value is selected instead. Conversely, in a case where the closest expressible value greater than the unrounded value should be selected, the closest expressible value less than the unrounded value is selected instead.

Example: If the double-precision FSUB instruction (FSUB DR0, DR2) is executed with input data DR0 = H'C1F00000 80000000, DR2 = H'C4B250D2 0CC1FB74, and FPSCR = H'000C0001, the correct operation result is DR2 = H'C4B250D2 0CC1F973, and FPSCR.Flag.I and FPSCR.Cause.I should be set to 1. However, the result actually produced by the FPU is DR2 = H'C4B250D2 0CC1F974, and FPSCR.Flag.I and FPSCR.Cause.I are not set to 1.

Effects: In addition to the problem described above, the numerical size of the result of the operation may contain a minute operation error equivalent to 1/256 of the LSB of the mantissa of the unrounded value. This is can be described as within the scope of the subsequent rounding mechanism. Strictly speaking, it consists of the following.

- a: The infinite-precision operation result
 - b: The closest expressible value less than a
 - c: The closest expressible value greater than a
 - d: The operation result when a is rounded correctly
 - e: The operation result when a is rounded by the FPU
- The rounding error when rounding is performed correctly in Round to Nearest mode is:

$$0 \leq |d - a| \leq (1/2) \times (c - b)$$

And the rounding error when rounding is performed by the FPU is:

$$0 \leq |e - a| < (129/256) \times (c - b)$$

If $c - b$ is considered the LSB of the mantissa, the range of rounding error is equivalent to 1/256 of the LSB of the mantissa of the correctly rounded value.

- The rounding error when rounding is performed correctly in Round to Zero mode is:

$$(-1) \times (c - b) < |d - a| \leq 0$$

And the rounding error when rounding is performed by the FPU is:

$$(-1) \times (c - b) < |e - a| < (1/256) \times (c - b)$$

If $c - b$ is considered the LSB of the mantissa, the range of rounding error is equivalent to 1/256 of the LSB of the mantissa of the correctly rounded value.

Section 7 Instruction Set

7.1 Execution Environment

PC: PC indicates the address of the instruction itself.

Data sizes and data types: The SH-4 instruction set is implemented with 16-bit fixed-length instructions. The SH-4 can use byte (8-bit), word (16-bit), longword (32-bit), and quadword (64-bit) data sizes for memory access. Single-precision floating-point data (32 bits) can be moved to and from memory using longword or quadword size. Double-precision floating-point data (64 bits) can be moved to and from memory using longword size. When a double-precision floating-point operation is specified (FPSCR.PR = 1), the result of an operation using quadword access will be undefined. When the SH-4 moves byte-size or word-size data from memory to a register, the data is sign-extended.

Load-Store Architecture: The SH-4 features a load-store architecture in which operations are basically executed using registers. Except for bit-manipulation operations such as logical AND that are executed directly in memory, operands in an operation that requires memory access are loaded into registers and the operation is executed between the registers.

Delayed Branches: Except for the two branch instructions BF and BT, the SH-4 branch instructions and RTE are delayed branches. In a delayed branch, the instruction following the branch is executed before the branch destination instruction. This execution slot following a delayed branch is called a delay slot. For example, the BRA execution sequence is as follows:

| Static Sequence | | Dynamic Sequence | | |
|-----------------|------------------|------------------|------------------------|---|
| BRA | TARGET | BRA | TARGET | |
| ADD | R1, R0 next_2 | ADD | R1, R0 target_instr | ADD in delay slot is executed before branching to TARGET |

Delay Slot: A slot illegal instruction exception may occur when a specific instruction is executed in a delay slot. See section 5, Exceptions. The instruction following BF/S or BT/S for which the branch is not taken is also a delay slot instruction.

T Bit: The T bit in the status register (SR) is used to show the result of a compare operation, and is referenced by a conditional branch instruction. An example of the use of a conditional branch instruction is shown below.

```
ADD #1, R0      ; T bit is not changed by ADD operation
CMP/EQ R1, R0   ; If R0 = R1, T bit is set to 1
BT TARGET      ; Branches to TARGET if T bit = 1 (R0 = R1)
```

In an RTE delay slot, status register (SR) bits are referenced as follows. In instruction access, the MD bit is used before modification, and in data access, the MD bit is accessed after modification. The other bits—S, T, M, Q, FD, BL, and RB—after modification are used for delay slot instruction execution. The STC and STC.L SR instructions access all SR bits after modification.

Constant Values: An 8-bit constant value can be specified by the instruction code and an immediate value. 16-bit and 32-bit constant values can be defined as literal constant values in memory, and can be referenced by a PC-relative load instruction.

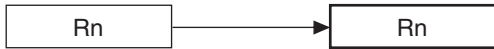
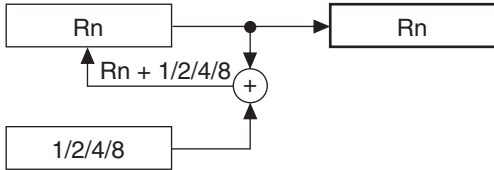
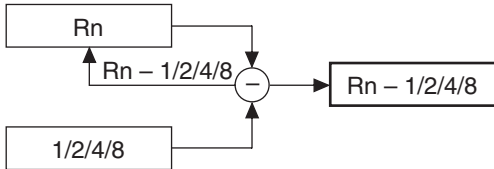
```
MOV.W  @(disp, PC), Rn
MOV.L  @(disp, PC), Rn
```

There are no PC-relative load instructions for floating-point operations. However, it is possible to set 0.0 or 1.0 by using the FLDI0 or FLDI1 instruction on a single-precision floating-point register.

7.2 Addressing Modes

Addressing modes and effective address calculation methods are shown in table 7.1. When a location in virtual memory space is accessed (MMUCR.AT = 1), the effective address is translated into a physical address. If multiple virtual memory space systems are selected (MMUCR.SV = 0), the least significant bit of PTEH is also referenced as the access ASID. See section 3, Memory Management Unit (MMU).

Table 7.1 Addressing Modes and Effective Addresses

| Addressing Mode | Instruction Format | Effective Address Calculation Method | Calculation Formula |
|---------------------------------------|--------------------|---|---|
| Register direct | Rn | Effective address is register Rn. (Operand is register Rn contents.) | — |
| Register indirect | @Rn | Effective address is register Rn contents.  | Rn → EA (EA: effective address) |
| Register indirect with post-increment | @Rn+ | Effective address is register Rn contents. A constant is added to Rn after instruction execution: 1 for a byte operand, 2 for a word operand, 4 for a longword operand, 8 for a quadword operand.  | Rn → EA After instruction execution Byte: Rn + 1 → Rn Word: Rn + 2 → Rn Longword: Rn + 4 → Rn Quadword: Rn + 8 → Rn |
| Register indirect with pre-decrement | @-Rn | Effective address is register Rn contents, decremented by a constant beforehand: 1 for a byte operand, 2 for a word operand, 4 for a longword operand, 8 for a quadword operand.  | Byte: Rn - 1 → Rn Word: Rn - 2 → Rn Longword: Rn - 4 → Rn Quadword: Rn - 8 → Rn Rn → EA (Instruction executed with Rn after calculation) |

| Addressing Mode | Instruction Format | Effective Address Calculation Method | Calculation Formula |
|-------------------------------------|--------------------|---|---|
| Register indirect with displacement | @(disp:4, Rn) | Effective address is register Rn contents with 4-bit displacement disp added. After disp is zero-extended, it is multiplied by 1 (byte), 2 (word), or 4 (longword), according to the operand size. | Byte: $Rn + disp \rightarrow EA$ Word: $Rn + disp \times 2 \rightarrow EA$ Longword: $Rn + disp \times 4 \rightarrow EA$ |
| | | | |
| Indexed register indirect | @(R0, Rn) | Effective address is sum of register Rn and R0 contents. | $Rn + R0 \rightarrow EA$ |
| | | | |
| GBR indirect with displacement | @(disp:8, GBR) | Effective address is register GBR contents with 8-bit displacement disp added. After disp is zero-extended, it is multiplied by 1 (byte), 2 (word), or 4 (longword), according to the operand size. | Byte: $GBR + disp \rightarrow EA$ Word: $GBR + disp \times 2 \rightarrow EA$ Longword: $GBR + disp \times 4 \rightarrow EA$ |
| | | | |
| Indexed GBR indirect | @(R0, GBR) | Effective address is sum of register GBR and R0 contents. | $GBR + R0 \rightarrow EA$ |
| | | | |

| Addressing Mode | Instruction Format | Effective Address Calculation Method | Calculation Formula |
|---|--------------------|--|--|
| PC-relative with displacement | @(disp:8, PC) | Effective address is PC+4 with 8-bit displacement disp added. After disp is zero-extended, it is multiplied by 2 (word), or 4 (longword), according to the operand size. With a longword operand, the lower 2 bits of PC are masked. | Word: $PC + 4 + disp \times 2 \rightarrow EA$ Longword: $PC \& H'FFFFFFFC + 4 + disp \times 4 \rightarrow EA$ |
| <p style="text-align: right;">* With longword operand</p> | | | |
| PC-relative | disp:8 | Effective address is PC+4 with 8-bit displacement disp added after being sign-extended and multiplied by 2. | $PC + 4 + disp \times 2 \rightarrow \text{Branch-Target}$ |
| | | | |

| Addressing Mode | Instruction Format | Effective Address Calculation Method | Calculation Formula |
|-----------------|--------------------|---|--|
| PC-relative | disp:12 | Effective address is PC+4 with 12-bit displacement disp added after being sign-extended and multiplied by 2. | $PC + 4 + \text{disp} \times 2 \rightarrow \text{Branch-Target}$ |
| | | <pre> graph TD PC[PC] --> A((+)) 4[4] --> A A --> B((+)) disp["disp (sign-extended)"] --> C((·)) 2[2] --> C C --> B B --> Result["PC + 4 + disp · 2"] </pre> | |
| | Rn | Effective address is sum of PC+4 and Rn. | $PC + 4 + Rn \rightarrow \text{Branch-Target}$ |
| | | <pre> graph TD PC[PC] --> A((+)) 4[4] --> A A --> B((+)) Rn[Rn] --> B B --> Result["PC + 4 + Rn"] </pre> | |
| Immediate | #imm:8 | 8-bit immediate data imm of TST, AND, OR, or XOR instruction is zero-extended. | — |
| | #imm:8 | 8-bit immediate data imm of MOV, ADD, or CMP/EQ instruction is sign-extended. | — |
| | #imm:8 | 8-bit immediate data imm of TRAPA instruction is zero-extended and multiplied by 4. | — |

Note: For the addressing modes below that use a displacement (disp), the assembler descriptions in this manual show the value before scaling ($\times 1$, $\times 2$, or $\times 4$) is performed according to the operand size. This is done to clarify the operation of the chip. Refer to the relevant assembler notation rules for the actual assembler descriptions.

@ (disp:4, Rn) ; Register indirect with displacement

@ (disp:8, GBR) ; GBR indirect with displacement

@ (disp:8, PC) ; PC-relative with displacement

disp:8, disp:12 ; PC-relative

7.3 Instruction Set

Table 7.2 shows the notation used in the following SH instruction list.

Table 7.2 Notation Used in Instruction List

| Item | Format | Description |
|----------------------|--|--|
| Instruction mnemonic | OP.Sz SRC, DEST | OP: Operation code Sz: Size SRC: Source DEST: Source and/or destination operand |
| Summary of operation | | →, ←: Transfer direction (xx): Memory operand M/Q/T: SR flag bits &: Logical AND of individual bits : Logical OR of individual bits ^: Logical exclusive-OR of individual bits ~: Logical NOT of individual bits <<n, >>n: n-bit shift |
| Instruction code | MSB ↔ LSB | m m m m: Register number (Rm, FRm) n n n n: Register number (Rn, FRn) 0000: R0, FR0 0001: R1, FR1 : 1111: R15, FR15 m m m: Register number (DRm, XDm, Rm_BANK) n n n: Register number (DRm, XDm, Rn_BANK) 000: DR0, XD0, R0_BANK 001: DR2, XD2, R1_BANK : 111: DR14, XD14, R7_BANK m m: Register number (FVm) n n: Register number (FVn) 00: FV0 01: FV4 10: FV8 11: FV12 i i i i: Immediate data d d d d: Displacement |
| Privileged mode | | “Privileged” means the instruction can only be executed in privileged mode. |
| T bit | Value of T bit after instruction execution | —: No change |

Note: Scaling (×1, ×2, ×4, or ×8) is executed according to the size of the instruction operand(s).

Table 7.3 Fixed-Point Transfer Instructions

| Instruction | | Operation | Instruction Code | Privileged | T Bit |
|-------------|---------------|--|------------------|------------|-------|
| MOV | #imm,Rn | imm → sign extension → Rn | 1110nnnniiiiiii | — | — |
| MOV.W | @(disp,PC),Rn | (disp × 2 + PC + 4) → sign extension → Rn | 1001nnnnddddddd | — | — |
| MOV.L | @(disp,PC),Rn | (disp × 4 + PC & H'FFFFFFC + 4) → Rn | 1101nnnnddddddd | — | — |
| MOV | Rm,Rn | Rm → Rn | 0110nnnnmmmm0011 | — | — |
| MOV.B | Rm,@Rn | Rm → (Rn) | 0010nnnnmmmm0000 | — | — |
| MOV.W | Rm,@Rn | Rm → (Rn) | 0010nnnnmmmm0001 | — | — |
| MOV.L | Rm,@Rn | Rm → (Rn) | 0010nnnnmmmm0010 | — | — |
| MOV.B | @Rm,Rn | (Rm) → sign extension → Rn | 0110nnnnmmmm0000 | — | — |
| MOV.W | @Rm,Rn | (Rm) → sign extension → Rn | 0110nnnnmmmm0001 | — | — |
| MOV.L | @Rm,Rn | (Rm) → Rn | 0110nnnnmmmm0010 | — | — |
| MOV.B | Rm,@-Rn | Rn-1 → Rn, Rm → (Rn) | 0010nnnnmmmm0100 | — | — |
| MOV.W | Rm,@-Rn | Rn-2 → Rn, Rm → (Rn) | 0010nnnnmmmm0101 | — | — |
| MOV.L | Rm,@-Rn | Rn-4 → Rn, Rm → (Rn) | 0010nnnnmmmm0110 | — | — |
| MOV.B | @Rm+,Rn | (Rm) → sign extension → Rn, Rm + 1 → Rm | 0110nnnnmmmm0100 | — | — |
| MOV.W | @Rm+,Rn | (Rm) → sign extension → Rn, Rm + 2 → Rm | 0110nnnnmmmm0101 | — | — |
| MOV.L | @Rm+,Rn | (Rm) → Rn, Rm + 4 → Rm | 0110nnnnmmmm0110 | — | — |
| MOV.B | R0,@(disp,Rn) | R0 → (disp + Rn) | 10000000nnnndddd | — | — |
| MOV.W | R0,@(disp,Rn) | R0 → (disp × 2 + Rn) | 10000001nnnndddd | — | — |
| MOV.L | Rm,@(disp,Rn) | Rm → (disp × 4 + Rn) | 0001nnnnmmmmdddd | — | — |
| MOV.B | @(disp,Rm),R0 | (disp + Rm) → sign extension → R0 | 10000100mmmmdddd | — | — |
| MOV.W | @(disp,Rm),R0 | (disp × 2 + Rm) → sign extension → R0 | 10000101mmmmdddd | — | — |
| MOV.L | @(disp,Rm),Rn | (disp × 4 + Rm) → Rn | 0101nnnnmmmmdddd | — | — |
| MOV.B | Rm,@(R0,Rn) | Rm → (R0 + Rn) | 0000nnnnmmmm0100 | — | — |
| MOV.W | Rm,@(R0,Rn) | Rm → (R0 + Rn) | 0000nnnnmmmm0101 | — | — |
| MOV.L | Rm,@(R0,Rn) | Rm → (R0 + Rn) | 0000nnnnmmmm0110 | — | — |
| MOV.B | @(R0,Rm),Rn | (R0 + Rm) → sign extension → Rn | 0000nnnnmmmm1100 | — | — |
| MOV.W | @(R0,Rm),Rn | (R0 + Rm) → sign extension → Rn | 0000nnnnmmmm1101 | — | — |
| MOV.L | @(R0,Rm),Rn | (R0 + Rm) → Rn | 0000nnnnmmmm1110 | — | — |

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|-------------|--|------------------|------------|-------|
| MOV.B | R0,@(disp,GBR) R0 → (disp + GBR) | 11000000dddddddd | — | — |
| MOV.W | R0,@(disp,GBR) R0 → (disp × 2 + GBR) | 11000001dddddddd | — | — |
| MOV.L | R0,@(disp,GBR) R0 → (disp × 4 + GBR) | 11000010dddddddd | — | — |
| MOV.B | @(disp,GBR),R0 (disp + GBR) → sign extension → R0 | 11000100dddddddd | — | — |
| MOV.W | @(disp,GBR),R0 (disp × 2 + GBR) → sign extension → R0 | 11000101dddddddd | — | — |
| MOV.L | @(disp,GBR),R0 (disp × 4 + GBR) → R0 | 11000110dddddddd | — | — |
| MOVA | @(disp,PC),R0 disp × 4 + PC & H'FFFFFFC + 4 → R0 | 11000111dddddddd | — | — |
| MOVT | Rn T → Rn | 0000nnnn00101001 | — | — |
| SWAP.B | Rm,Rn Rm → swap lower 2 bytes → Rn | 0110nnnnmmmm1000 | — | — |
| SWAP.W | Rm,Rn Rm → swap upper/lower words → Rn | 0110nnnnmmmm1001 | — | — |
| XTRCT | Rm,Rn Rm:Rn middle 32 bits → Rn | 0010nnnnmmmm1101 | — | — |

Table 7.4 Arithmetic Operation Instructions

| Instruction | | Operation | Instruction Code | Privileged | T Bit |
|-------------|---------|--|------------------|------------|--------------------|
| ADD | Rm,Rn | $Rn + Rm \rightarrow Rn$ | 0011nnnnmmmm1100 | — | — |
| ADD | #imm,Rn | $Rn + imm \rightarrow Rn$ | 0111nnnniiiiiii | — | — |
| ADDC | Rm,Rn | $Rn + Rm + T \rightarrow Rn$, carry $\rightarrow T$ | 0011nnnnmmmm1110 | — | Carry |
| ADDV | Rm,Rn | $Rn + Rm \rightarrow Rn$, overflow $\rightarrow T$ | 0011nnnnmmmm1111 | — | Overflow |
| CMP/EQ | #imm,R0 | When $R0 = imm$, $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 10001000iiiiiii | — | Comparison result |
| CMP/EQ | Rm,Rn | When $Rn = Rm$, $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0011nnnnmmmm0000 | — | Comparison result |
| CMP/HS | Rm,Rn | When $Rn \geq Rm$ (unsigned), $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0011nnnnmmmm0010 | — | Comparison result |
| CMP/GE | Rm,Rn | When $Rn \geq Rm$ (signed), $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0011nnnnmmmm0011 | — | Comparison result |
| CMP/HI | Rm,Rn | When $Rn > Rm$ (unsigned), $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0011nnnnmmmm0110 | — | Comparison result |
| CMP/GT | Rm,Rn | When $Rn > Rm$ (signed), $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0011nnnnmmmm0111 | — | Comparison result |
| CMP/PZ | Rn | When $Rn \geq 0$, $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0100nnnn00010001 | — | Comparison result |
| CMP/PL | Rn | When $Rn > 0$, $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0100nnnn00010101 | — | Comparison result |
| CMP/STR | Rm,Rn | When any bytes are equal, $1 \rightarrow T$ Otherwise, $0 \rightarrow T$ | 0010nnnnmmmm1100 | — | Comparison result |
| DIV1 | Rm,Rn | 1-step division ($Rn \div Rm$) | 0011nnnnmmmm0100 | — | Calculation result |
| DIV0S | Rm,Rn | MSB of $Rn \rightarrow Q$, MSB of $Rm \rightarrow M$, $M \div Q \rightarrow T$ | 0010nnnnmmmm0111 | — | Calculation result |
| DIV0U | | $0 \rightarrow M/Q/T$ | 0000000000011001 | — | 0 |
| DMULS.L | Rm,Rn | Signed, $Rn \times Rm \rightarrow MAC$, $32 \times 32 \rightarrow 64$ bits | 0011nnnnmmmm1101 | — | — |
| DMULU.L | Rm,Rn | Unsigned, $Rn \times Rm \rightarrow MAC$, $32 \times 32 \rightarrow 64$ bits | 0011nnnnmmmm0101 | — | — |
| DT | Rn | $Rn - 1 \rightarrow Rn$; when $Rn = 0$, $1 \rightarrow T$ When $Rn \neq 0$, $0 \rightarrow T$ | 0100nnnn00010000 | — | Comparison result |
| EXTS.B | Rm,Rn | Rm sign-extended from byte $\rightarrow Rn$ | 0110nnnnmmmm1110 | — | — |

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|-------------|--|------------------|------------|-----------|
| EXTS.W | Rm,Rn Rm sign-extended from word → Rn | 0110nnnnmmmm1111 | — | — |
| EXTU.B | Rm,Rn Rm zero-extended from byte → Rn | 0110nnnnmmmm1100 | — | — |
| EXTU.W | Rm,Rn Rm zero-extended from word → Rn | 0110nnnnmmmm1101 | — | — |
| MAC.L | @Rm+,@Rn+ Signed, (Rn) × (Rm) + MAC → MAC Rn + 4 → Rn, Rm + 4 → Rm 32 × 32 + 64 → 64 bits | 0000nnnnmmmm1111 | — | — |
| MAC.W | @Rm+,@Rn+ Signed, (Rn) × (Rm) + MAC → MAC Rn + 2 → Rn, Rm + 2 → Rm 16 × 16 + 64 → 64 bits | 0100nnnnmmmm1111 | — | — |
| MUL.L | Rm,Rn Rn × Rm → MACL 32 × 32 → 32 bits | 0000nnnnmmmm0111 | — | — |
| MULS.W | Rm,Rn Signed, Rn × Rm → MACL 16 × 16 → 32 bits | 0010nnnnmmmm1111 | — | — |
| MULU.W | Rm,Rn Unsigned, Rn × Rm → MACL 16 × 16 → 32 bits | 0010nnnnmmmm1110 | — | — |
| NEG | Rm,Rn 0 – Rm → Rn | 0110nnnnmmmm1011 | — | — |
| NEGC | Rm,Rn 0 – Rm – T → Rn, borrow → T | 0110nnnnmmmm1010 | — | Borrow |
| SUB | Rm,Rn Rn – Rm → Rn | 0011nnnnmmmm1000 | — | — |
| SUBC | Rm,Rn Rn – Rm – T → Rn, borrow → T | 0011nnnnmmmm1010 | — | Borrow |
| SUBV | Rm,Rn Rn – Rm → Rn, underflow → T | 0011nnnnmmmm1011 | — | Underflow |

Table 7.5 Logic Operation Instructions

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|----------------------|--|------------------|------------|-------------|
| AND Rm,Rn | $Rn \& Rm \rightarrow Rn$ | 0010nnnnmmmm1001 | — | — |
| AND #imm,R0 | $R0 \& imm \rightarrow R0$ | 11001001iiiiiiii | — | — |
| AND.B #imm,@(R0,GBR) | $(R0 + GBR) \& imm \rightarrow (R0 + GBR)$ | 11001101iiiiiiii | — | — |
| NOT Rm,Rn | $\sim Rm \rightarrow Rn$ | 0110nnnnmmmm0111 | — | — |
| OR Rm,Rn | $Rn Rm \rightarrow Rn$ | 0010nnnnmmmm1011 | — | — |
| OR #imm,R0 | $R0 imm \rightarrow R0$ | 11001011iiiiiiii | — | — |
| OR.B #imm,@(R0,GBR) | $(R0 + GBR) imm \rightarrow (R0 + GBR)$ | 11001111iiiiiiii | — | — |
| TAS.B @Rn | When (Rn) = 0, 1 → T Otherwise, 0 → T In both cases, 1 → MSB of (Rn) | 0100nnnn00011011 | — | Test result |
| TST Rm,Rn | $Rn \& Rm$; when result = 0, 1 → T Otherwise, 0 → T | 0010nnnnmmmm1000 | — | Test result |
| TST #imm,R0 | $R0 \& imm$; when result = 0, 1 → T Otherwise, 0 → T | 11001000iiiiiiii | — | Test result |
| TST.B #imm,@(R0,GBR) | $(R0 + GBR) \& imm$; when result = 0, 1 → T Otherwise, 0 → T | 11001100iiiiiiii | — | Test result |
| XOR Rm,Rn | $Rn \wedge Rm \rightarrow Rn$ | 0010nnnnmmmm1010 | — | — |
| XOR #imm,R0 | $R0 \wedge imm \rightarrow R0$ | 11001010iiiiiiii | — | — |
| XOR.B #imm,@(R0,GBR) | $(R0 + GBR) \wedge imm \rightarrow (R0 + GBR)$ | 11001110iiiiiiii | — | — |

Table 7.6 Shift Instructions

| Instruction | | Operation | Instruction Code | Privileged | T Bit |
|-------------|-------|---|------------------|------------|-------|
| ROTL | Rn | $T \leftarrow Rn \leftarrow \text{MSB}$ | 0100nnnn00000100 | — | MSB |
| ROTR | Rn | $\text{LSB} \rightarrow Rn \rightarrow T$ | 0100nnnn00000101 | — | LSB |
| ROTCL | Rn | $T \leftarrow Rn \leftarrow T$ | 0100nnnn00100100 | — | MSB |
| ROTCR | Rn | $T \rightarrow Rn \rightarrow T$ | 0100nnnn00100101 | — | LSB |
| SHAD | Rm,Rn | When $Rn \geq 0$, $Rn \ll Rm \rightarrow Rn$ When $Rn < 0$, $Rn \gg Rm \rightarrow$ [MSB $\rightarrow Rn$] | 0100nnnnnnmm1100 | — | — |
| SHAL | Rn | $T \leftarrow Rn \leftarrow 0$ | 0100nnnn00100000 | — | MSB |
| SHAR | Rn | $\text{MSB} \rightarrow Rn \rightarrow T$ | 0100nnnn00100001 | — | LSB |
| SHLD | Rm,Rn | When $Rn \geq 0$, $Rn \ll Rm \rightarrow Rn$ When $Rn < 0$, $Rn \gg Rm \rightarrow$ [0 $\rightarrow Rn$] | 0100nnnnnnmm1101 | — | — |
| SHLL | Rn | $T \leftarrow Rn \leftarrow 0$ | 0100nnnn00000000 | — | MSB |
| SHLR | Rn | $0 \rightarrow Rn \rightarrow T$ | 0100nnnn00000001 | — | LSB |
| SHLL2 | Rn | $Rn \ll 2 \rightarrow Rn$ | 0100nnnn00001000 | — | — |
| SHLR2 | Rn | $Rn \gg 2 \rightarrow Rn$ | 0100nnnn00001001 | — | — |
| SHLL8 | Rn | $Rn \ll 8 \rightarrow Rn$ | 0100nnnn00011000 | — | — |
| SHLR8 | Rn | $Rn \gg 8 \rightarrow Rn$ | 0100nnnn00011001 | — | — |
| SHLL16 | Rn | $Rn \ll 16 \rightarrow Rn$ | 0100nnnn00101000 | — | — |
| SHLR16 | Rn | $Rn \gg 16 \rightarrow Rn$ | 0100nnnn00101001 | — | — |

Table 7.7 Branch Instructions

| Instruction | | Operation | Instruction Code | Privileged | T Bit |
|-------------|-------|--|-------------------|------------|-------|
| BF | label | When T = 0, $\text{disp} \times 2 + \text{PC} + 4 \rightarrow \text{PC}$ When T = 1, nop | 10001011ddddddddd | — | — |
| BF/S | label | Delayed branch; when T = 0, $\text{disp} \times 2 + \text{PC} + 4 \rightarrow \text{PC}$ When T = 1, nop | 10001111ddddddddd | — | — |
| BT | label | When T = 1, $\text{disp} \times 2 + \text{PC} + 4 \rightarrow \text{PC}$ When T = 0, nop | 10001001ddddddddd | — | — |
| BT/S | label | Delayed branch; when T = 1, $\text{disp} \times 2 + \text{PC} + 4 \rightarrow \text{PC}$ When T = 0, nop | 10001101ddddddddd | — | — |
| BRA | label | Delayed branch, $\text{disp} \times 2 + \text{PC} + 4 \rightarrow \text{PC}$ | 1010ddddddddd | — | — |
| BRAF | Rn | $\text{Rn} + \text{PC} + 4 \rightarrow \text{PC}$ | 0000nnnn00100011 | — | — |
| BSR | label | Delayed branch, $\text{PC} + 4 \rightarrow \text{PR}$, $\text{disp} \times 2 + \text{PC} + 4 \rightarrow \text{PC}$ | 1011ddddddddd | — | — |
| BSRF | Rn | Delayed branch, $\text{PC} + 4 \rightarrow \text{PR}$, $\text{Rn} + \text{PC} + 4 \rightarrow \text{PC}$ | 0000nnnn00000011 | — | — |
| JMP | @Rn | Delayed branch, $\text{Rn} \rightarrow \text{PC}$ | 0100nnnn00101011 | — | — |
| JSR | @Rn | Delayed branch, $\text{PC} + 4 \rightarrow \text{PR}$, $\text{Rn} \rightarrow \text{PC}$ | 0100nnnn00001011 | — | — |
| RTS | | Delayed branch, $\text{PR} \rightarrow \text{PC}$ | 0000000000001011 | — | — |

Table 7.8 System Control Instructions

| Instruction | | Operation | Instruction Code | Privileged | T Bit |
|-------------|--------------|--|------------------|------------|-------|
| CLRMAC | | 0 → MACH, MACL | 000000000101000 | — | — |
| CLRS | | 0 → S | 000000001001000 | — | — |
| CLRT | | 0 → T | 000000000001000 | — | 0 |
| LDC | Rm,SR | Rm → SR | 0100mmmm00001110 | Privileged | LSB |
| LDC | Rm,GBR | Rm → GBR | 0100mmmm00011110 | — | — |
| LDC | Rm,VBR | Rm → VBR | 0100mmmm00101110 | Privileged | — |
| LDC | Rm,SSR | Rm → SSR | 0100mmmm00111110 | Privileged | — |
| LDC | Rm,SPC | Rm → SPC | 0100mmmm01001110 | Privileged | — |
| LDC | Rm,DBR | Rm → DBR | 0100mmmm11111010 | Privileged | — |
| LDC | Rm,Rn_BANK | Rm → Rn_BANK (n = 0 to 7) | 0100mmmm1nnn1110 | Privileged | — |
| LDC.L | @Rm+,SR | (Rm) → SR, Rm + 4 → Rm | 0100mmmm00000111 | Privileged | LSB |
| LDC.L | @Rm+,GBR | (Rm) → GBR, Rm + 4 → Rm | 0100mmmm00010111 | — | — |
| LDC.L | @Rm+,VBR | (Rm) → VBR, Rm + 4 → Rm | 0100mmmm00100111 | Privileged | — |
| LDC.L | @Rm+,SSR | (Rm) → SSR, Rm + 4 → Rm | 0100mmmm00110111 | Privileged | — |
| LDC.L | @Rm+,SPC | (Rm) → SPC, Rm + 4 → Rm | 0100mmmm01000111 | Privileged | — |
| LDC.L | @Rm+,DBR | (Rm) → DBR, Rm + 4 → Rm | 0100mmmm11110110 | Privileged | — |
| LDC.L | @Rm+,Rn_BANK | (Rm) → Rn_BANK, Rm + 4 → Rm | 0100mmmm1nnn0111 | Privileged | — |
| LDS | Rm,MACH | Rm → MACH | 0100mmmm00001010 | — | — |
| LDS | Rm,MACL | Rm → MACL | 0100mmmm00011010 | — | — |
| LDS | Rm,PR | Rm → PR | 0100mmmm00101010 | — | — |
| LDS.L | @Rm+,MACH | (Rm) → MACH, Rm + 4 → Rm | 0100mmmm00000110 | — | — |
| LDS.L | @Rm+,MACL | (Rm) → MACL, Rm + 4 → Rm | 0100mmmm00010110 | — | — |
| LDS.L | @Rm+,PR | (Rm) → PR, Rm + 4 → Rm | 0100mmmm00100110 | — | — |
| LDTLB | | PTEH/PTEL → TLB | 000000000111000 | Privileged | — |
| MOVCA. L | R0,@Rn | R0 → (Rn) (without fetching cache block) | 0000nynn11000011 | — | — |
| NOP | | No operation | 000000000001001 | — | — |
| OCBI | @Rn | Invalidates operand cache block | 0000nynn10010011 | — | — |
| OCBP | @Rn | Writes back and invalidates operand cache block | 0000nynn10100011 | — | — |
| OCBWB | @Rn | Writes back operand cache block | 0000nynn10110011 | — | — |
| PREF | @Rn | (Rn) → operand cache | 0000nynn10000011 | — | — |

| Instruction | Operation | Instruction Code | Privileged | T Bit | |
|-------------|---------------------------------|---|------------------|------------|---|
| RTE | Delayed branch, SSR/SPC → SR/PC | 000000000101011 | Privileged | — | |
| SETS | 1 → S | 000000001011000 | — | — | |
| SETT | 1 → T | 000000000011000 | — | 1 | |
| SLEEP | Sleep or standby | 000000000011011 | Privileged | — | |
| STC | SR,Rn | SR → Rn | 0000nnnn0000010 | Privileged | — |
| STC | GBR,Rn | GBR → Rn | 0000nnnn00010010 | — | — |
| STC | VBR,Rn | VBR → Rn | 0000nnnn00100010 | Privileged | — |
| STC | SSR,Rn | SSR → Rn | 0000nnnn00110010 | Privileged | — |
| STC | SPC,Rn | SPC → Rn | 0000nnnn01000010 | Privileged | — |
| STC | SGR,Rn | SGR → Rn | 0000nnnn00111010 | Privileged | — |
| STC | DBR,Rn | DBR → Rn | 0000nnnn11111010 | Privileged | — |
| STC | Rm_BANK,Rn | Rm_BANK → Rn (m = 0 to 7) | 0000nnnn1mmm0010 | Privileged | — |
| STC.L | SR,@-Rn | Rn - 4 → Rn, SR → (Rn) | 0100nnnn0000011 | Privileged | — |
| STC.L | GBR,@-Rn | Rn - 4 → Rn, GBR → (Rn) | 0100nnnn00010011 | — | — |
| STC.L | VBR,@-Rn | Rn - 4 → Rn, VBR → (Rn) | 0100nnnn00100011 | Privileged | — |
| STC.L | SSR,@-Rn | Rn - 4 → Rn, SSR → (Rn) | 0100nnnn00110011 | Privileged | — |
| STC.L | SPC,@-Rn | Rn - 4 → Rn, SPC → (Rn) | 0100nnnn01000011 | Privileged | — |
| STC.L | SGR,@-Rn | Rn - 4 → Rn, SGR → (Rn) | 0100nnnn00110010 | Privileged | — |
| STC.L | DBR,@-Rn | Rn - 4 → Rn, DBR → (Rn) | 0100nnnn11110010 | Privileged | — |
| STC.L | Rm_BANK,@-Rn | Rn - 4 → Rn, Rm_BANK → (Rn) (m = 0 to 7) | 0100nnnn1mmm0011 | Privileged | — |
| STS | MACH,Rn | MACH → Rn | 0000nnnn00001010 | — | — |
| STS | MACL,Rn | MACL → Rn | 0000nnnn00011010 | — | — |
| STS | PR,Rn | PR → Rn | 0000nnnn00101010 | — | — |
| STS.L | MACH,@-Rn | Rn - 4 → Rn, MACH → (Rn) | 0100nnnn00000010 | — | — |
| STS.L | MACL,@-Rn | Rn - 4 → Rn, MACL → (Rn) | 0100nnnn00010010 | — | — |
| STS.L | PR,@-Rn | Rn - 4 → Rn, PR → (Rn) | 0100nnnn00100010 | — | — |
| TRAPA | #imm | PC + 2 → SPC, SR → SSR, #imm << 2 → TRA, H'160 → EXPEVT, VBR + H'0100 → PC | 11000011iiiiiii | — | — |

Table 7.9 Floating-Point Single-Precision Instructions

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|-------------|--------------|---|------------|-------------------|
| FLDI0 | FRn | H'00000000 → FRn | — | — |
| FLDI1 | FRn | H'3F800000 → FRn | — | — |
| FMOV | FRm,FRn | FRm → FRn | — | — |
| FMOV.S | @Rm,FRn | (Rm) → FRn | — | — |
| FMOV.S | @(R0,Rm),FRn | (R0 + Rm) → FRn | — | — |
| FMOV.S | @Rm+,FRn | (Rm) → FRn, Rm + 4 → Rm | — | — |
| FMOV.S | FRm,@Rn | FRm → (Rn) | — | — |
| FMOV.S | FRm,@-Rn | Rn-4 → Rn, FRm → (Rn) | — | — |
| FMOV.S | FRm,@(R0,Rn) | FRm → (R0 + Rn) | — | — |
| FMOV | DRm,DRn | DRm → DRn | — | — |
| FMOV | @Rm,DRn | (Rm) → DRn | — | — |
| FMOV | @(R0,Rm),DRn | (R0 + Rm) → DRn | — | — |
| FMOV | @Rm+,DRn | (Rm) → DRn, Rm + 8 → Rm | — | — |
| FMOV | DRm,@Rn | DRm → (Rn) | — | — |
| FMOV | DRm,@-Rn | Rn-8 → Rn, DRm → (Rn) | — | — |
| FMOV | DRm,@(R0,Rn) | DRm → (R0 + Rn) | — | — |
| FLDS | FRm,FPUL | FRm → FPUL | — | — |
| FSTS | FPUL,FRn | FPUL → FRn | — | — |
| FABS | FRn | FRn & H'7FFF FFFF → FRn | — | — |
| FADD | FRm,FRn | FRn + FRm → FRn | — | — |
| FCMP/EQ | FRm,FRn | When FRn = FRm, 1 → T Otherwise, 0 → T | — | Comparison result |
| FCMP/GT | FRm,FRn | When FRn > FRm, 1 → T Otherwise, 0 → T | — | Comparison result |
| FDIV | FRm,FRn | FRn/FRm → FRn | — | — |
| FLOAT | FPUL,FRn | (float) FPUL → FRn | — | — |
| FMAC | FR0,FRm,FRn | FR0*FRm + FRn → FRn | — | — |
| FMUL | FRm,FRn | FRn*FRm → FRn | — | — |
| FNEG | FRn | FRn ^ H'80000000 → FRn | — | — |
| FSQRT | FRn | $\sqrt{\text{FRn}}$ → FRn | — | — |
| FSUB | FRm,FRn | FRn - FRm → FRn | — | — |
| FTRC | FRm,FPUL | (long) FRm → FPUL | — | — |

Table 7.10 Floating-Point Double-Precision Instructions

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|-----------------|---|------------------|------------|-------------------|
| FABS DRn | DRn & H'7FFF FFFF FFFF FFFF → DRn | 1111nnn001011101 | — | — |
| FADD DRm,DRn | DRn + DRm → DRn | 1111nnn0mmm00000 | — | — |
| FCMP/EQ DRm,DRn | When DRn = DRm, 1 → T Otherwise, 0 → T | 1111nnn0mmm00100 | — | Comparison result |
| FCMP/GT DRm,DRn | When DRn > DRm, 1 → T Otherwise, 0 → T | 1111nnn0mmm00101 | — | Comparison result |
| FDIV DRm,DRn | DRn /DRm → DRn | 1111nnn0mmm00011 | — | — |
| FCNVDS DRm,FPUL | double_to_float[DRm] → FPUL | 1111mmm010111101 | — | — |
| FCNVSD FPUL,DRn | float_to_double [FPUL] → DRn | 1111nnn010101101 | — | — |
| FLOAT FPUL,DRn | (float)FPUL → DRn | 1111nnn000101101 | — | — |
| FMUL DRm,DRn | DRn *DRm → DRn | 1111nnn0mmm00010 | — | — |
| FNEG DRn | DRn ^ H'8000 0000 0000 0000 → DRn | 1111nnn001001101 | — | — |
| FSQRT DRn | $\sqrt{\text{DRn}}$ → DRn | 1111nnn001101101 | — | — |
| FSUB DRm,DRn | DRn - DRm → DRn | 1111nnn0mmm00001 | — | — |
| FTRC DRm,FPUL | (long) DRm → FPUL | 1111mmm000111101 | — | — |

Table 7.11 Floating-Point Control Instructions

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|------------------|---------------------------|------------------|------------|-------|
| LDS Rm,FPSCR | Rm → FPSCR | 0100mmmm01101010 | — | — |
| LDS Rm,FPUL | Rm → FPUL | 0100mmmm01011010 | — | — |
| LDS.L @Rm+,FPSCR | (Rm) → FPSCR, Rm+4 → Rm | 0100mmmm01100110 | — | — |
| LDS.L @Rm+,FPUL | (Rm) → FPUL, Rm+4 → Rm | 0100mmmm01010110 | — | — |
| STS FPSCR,Rn | FPSCR → Rn | 0000nnnn01101010 | — | — |
| STS FPUL,Rn | FPUL → Rn | 0000nnnn01011010 | — | — |
| STS.L FPSCR,@-Rn | Rn - 4 → Rn, FPSCR → (Rn) | 0100nnnn01100010 | — | — |
| STS.L FPUL,@-Rn | Rn - 4 → Rn, FPUL → (Rn) | 0100nnnn01010010 | — | — |

Table 7.12 Floating-Point Graphics Acceleration Instructions

| Instruction | Operation | Instruction Code | Privileged | T Bit |
|---------------------|-------------------------------------|-------------------|------------|-------|
| FMOV DRm, XDn | DRm → XDn | 1111nnn1mmmm01100 | — | — |
| FMOV XDm, DRn | XDm → DRn | 1111nnn0mmmm11100 | — | — |
| FMOV XDm, XDn | XDm → XDn | 1111nnn1mmmm11100 | — | — |
| FMOV @Rm, XDn | (Rm) → XDn | 1111nnn1mmmm1000 | — | — |
| FMOV @Rm+, XDn | (Rm) → XDn, Rm + 8 → Rm | 1111nnn1mmmm1001 | — | — |
| FMOV @(R0, Rm), XDn | (R0 + Rm) → XDn | 1111nnn1mmmm0110 | — | — |
| FMOV XDm, @Rn | XDm → (Rn) | 1111nnnnmmmm11010 | — | — |
| FMOV XDm, @-Rn | Rn - 8 → Rn, XDm → (Rn) | 1111nnnnmmmm11011 | — | — |
| FMOV XDm, @(R0, Rn) | XDm → (R0+Rn) | 1111nnnnmmmm10111 | — | — |
| FIPR FVm, FVn | inner_product [FVm, FVn] → FR[n+3] | 1111nnmm11101101 | — | — |
| FTRV XMTRX, FVn | transform_vector [XMTRX, FVn] → FVn | 1111nn0111111101 | — | — |
| FRCHG | ~FPSCR.FR → FPSCR.FR | 1111101111111101 | — | — |
| FSCHG | ~FPSCR.SZ → FPSCR.SZ | 1111001111111101 | — | — |

7.4 Usage Notes

7.4.1 Notes on TRAPA Instruction, SLEEP Instruction, and Undefined Instruction (H'FFFD)

- Incorrect data may be written to the cache when a TRAPA instruction or undefined instruction code H'FFFD is executed.
- The ITLB hit judgment may be incorrect when a TRAPA instruction or undefined instruction code H'FFFD is executed, causing a multi-hit exception to occur after re-registration.
- Incorrect data may be written to an FPU-related register or to the MACH or MACL register when a TRAPA instruction, SLEEP instruction, or undefined instruction code H'FFFD is executed.

Conditions Under which Problem Occurs

1. Incorrect data may be written to the instruction cache when the following three conditions occur at the same time.
 - a. The instruction cache is enabled (CCR.ICE = 1).

- b. A TRAPA instruction or undefined instruction code H'FFFD in a cache-enabled area (U0, P0, P1, or P3 area) is executed.
 - c. The four words of data following the TRAPA instruction or undefined instruction code H'FFFD mentioned in b. contain code that can be interpreted as an instruction to access (read or write) an address (H'F0000000 to H'F7FFFFFF) mapped to the internal cache or internal TLB.
2. Incorrect data may be written to the operand cache when the following three conditions occur at the same time.
 - a. The operand cache is enabled (CCR.OCE = 1).
 - b. Undefined instruction code H'FFFD is executed.
 - c. The four words of data following the undefined instruction code H'FFFD mentioned in b. contain code that can be interpreted as an OCBI, OCBP, OCBWB, or TAS.B instruction accessing an address (H'E0000000 to H'E3FFFFFF) mapped to the internal store queue.
 3. The ITLB hit judgment may be incorrect when the following three conditions occur at the same time. If an ITLB hit is erroneously judged to be a miss, ITLB re-registration is performed. This can cause an ITLB multi-hit exception to occur.
 - a. The MMU is enabled (MMUCR.AT = 1).
 - b. A TRAPA instruction or undefined instruction code H'FFFD in a TLB conversion area (U0, P0, or P3 area) is executed.
 - c. The four words of data following the TRAPA instruction or undefined instruction code H'FFFD mentioned in b. contain code that can be interpreted as an instruction to access (read or write) an address (H'F0000000 to H'F7FFFFFF) mapped to the internal cache or internal TLB.
 4. Incorrect data may be written to an FPU-related register (FR0 to FR15, XF0 to XF15, FPSCR, or FPUL) or to the MACH or MACL register when the following two conditions occur at the same time.
 - a. A TRAPA instruction, SLEEP instruction, or undefined instruction code H'FFFD is executed
 - b. The eight words of data following the TRAPA instruction, SLEEP instruction, or undefined instruction code H'FFFD mentioned in a. contain H'Fxxx (an instruction with H'F as the first four bits), excluding H'FFFD, and the code can be interpreted, in combination with FPSCR.PR at that point, as an undefined instruction.
Example: Instruction H'FxxE (x: any hexadecimal digit) is defined here as undefined when FPSCR.PR is set to 1.

Note: The number of instructions following the instructions mentioned above that may be affected by the problem is as follows: in the case of 1. to 3., the number of instructions that can be executed in 2xIck, and in the case of 4., the number of instructions that can be

executed in 4xIck. The maximum number of instructions that can be executed in 2xIck or 4xIck is four or eight, respectively. Therefore, the affected codes are those occurring in “the four words (or eight words) of data following the instruction.”

Workarounds

1. To prevent the problem, use either of workarounds a. or b. below.
 - a. Include a NOP instruction in the eight words of data following each TRAPA instruction, SLEEP instruction, or undefined instruction code H'FFFD.
 - b. Include an OR R0,R0 instruction in the five words of data following each TRAPA instruction, SLEEP instruction, or undefined instruction code H'FFFD. This workaround also applies to cases where “the eight words of data following the ... instruction ... contain H'Fxxx,” as mentioned in condition 4. b., because two OR instructions are never executed simultaneously, so a minimum of 5xIck is required for execution.

Section 8 Pipelining

This LSI is a 2-ILP (instruction-level-parallelism) superscalar pipelining microprocessor. Instruction execution is pipelined, and two instructions can be executed in parallel. The execution cycles depend on the implementation of a processor. Definitions in this section may not be applicable to SH-4 Core models other than this LSI.

8.1 Pipelines

Figure 8.1 shows the basic pipelines. Normally, a pipeline consists of five or six stages: instruction fetch (I), decode and register read (D), execution (EX/SX/F0/F1/F2/F3), data access (NA/MA), and write-back (S/FS). An instruction is executed as a combination of basic pipelines. Figure 8.2 shows the instruction execution patterns.

1. General Pipeline

| I | D | EX | NA | S |
|---|---|----|----|---|
|---|---|----|----|---|

- Instruction fetch
- Instruction decode
- Issue
- Register read
- Destination address calculation for PC-relative branch
- Operation
- Non-memory data access
- Write-back

2. General Load/Store Pipeline

| I | D | EX | MA | S |
|---|---|----|----|---|
|---|---|----|----|---|

- Instruction fetch
- Instruction decode
- Issue
- Register read
- Address calculation
- Memory data access
- Write-back

3. Special Pipeline

| I | D | SX | NA | S |
|---|---|----|----|---|
|---|---|----|----|---|

- Instruction fetch
- Instruction decode
- Issue
- Register read
- Operation
- Non-memory data access
- Write-back

4. Special Load/Store Pipeline

| I | D | SX | MA | S |
|---|---|----|----|---|
|---|---|----|----|---|

- Instruction fetch
- Instruction decode
- Issue
- Register read
- Address calculation
- Memory data access
- Write-back

5. Floating-Point Pipeline

| I | D | F1 | F2 | FS |
|---|---|----|----|----|
|---|---|----|----|----|

- Instruction fetch
- Instruction decode
- Issue
- Register read
- Computation 1
- Computation 2
- Computation 3
- Write-back

6. Floating-Point Extended Pipeline

| I | D | F0 | F1 | F2 | FS |
|---|---|----|----|----|----|
|---|---|----|----|----|----|

- Instruction fetch
- Instruction decode
- Issue
- Register read
- Computation 0
- Computation 1
- Computation 2
- Computation 3
- Write-back

7. FDIV/FSQRT Pipeline

| F3 |
|----|
|----|

Computation: Takes several cycles

Figure 8.1 Basic Pipelines

1. 1-step operation: 1 issue cycle

EXT[SU].[BW], MOV, MOV#, MOVA, MOV, SWAP.[BW], XTRCT, ADD*, CMP*,
 DIV*, DT, NEG*, SUB*, AND, AND#, NOT, OR, OR#, TST, TST#, XOR, XOR#,
 ROT*, SHA*, SHL*, BF*, BT*, BRA, NOP, CLRS, CLRT, SETS, SETT,
 LDS to FPUL, STS from FPUL/FPSCR, FLDI0, FLDI1, FMOV, FLDS, FSTS,
 single-/double-precision FABS/FNEG

| | | | | |
|---|---|----|----|---|
| I | D | EX | NA | S |
|---|---|----|----|---|

2. Load/store: 1 issue cycle

MOV.[BWL], FMOV*@[, LDS.L to FPUL, LDTLB, PREF, STS.L from FPUL/FPSCR

| | | | | |
|---|---|----|----|---|
| I | D | EX | MA | S |
|---|---|----|----|---|

3. GBR-based load/store: 1 issue cycle

MOV.[BWL]@(d,GBR)

| | | | | |
|---|---|----|----|---|
| I | D | SX | MA | S |
|---|---|----|----|---|

4. JMP, RTS, BRAF: 2 issue cycles

| | | | | | |
|---|---|----|----|----|---|
| I | D | EX | NA | S | |
| | | D | EX | NA | S |

5. TST.B: 3 issue cycles

| | | | | | | |
|---|---|----|----|----|----|---|
| I | D | SX | MA | S | | |
| | | D | SX | NA | S | |
| | | | D | SX | NA | S |

6. AND.B, OR.B, XOR.B: 4 issue cycles

| | | | | | | | |
|---|---|----|----|----|----|----|---|
| I | D | SX | MA | S | | | |
| | | D | SX | NA | S | | |
| | | | D | SX | NA | S | |
| | | | | D | SX | MA | S |

7. TAS.B: 5 issue cycles

| | | | | | | | | |
|---|---|----|----|----|----|----|----|---|
| I | D | EX | MA | S | | | | |
| | | D | EX | MA | S | | | |
| | | | D | EX | NA | S | | |
| | | | | D | EX | NA | S | |
| | | | | | D | EX | MA | S |

8. RTE: 5 issue cycles

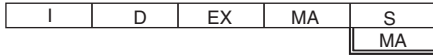
| | | | | | | | | |
|---|---|----|----|----|----|----|----|---|
| I | D | EX | NA | S | | | | |
| | | D | EX | NA | S | | | |
| | | | D | EX | NA | S | | |
| | | | | D | EX | NA | S | |
| | | | | | D | EX | NA | S |

9. SLEEP: 4 issue cycles

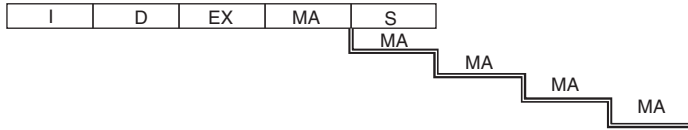
| | | | | | | | |
|---|---|----|----|----|----|----|---|
| I | D | EX | NA | S | | | |
| | | D | EX | NA | S | | |
| | | | D | EX | NA | S | |
| | | | | D | EX | NA | S |

Figure 8.2 Instruction Execution Patterns

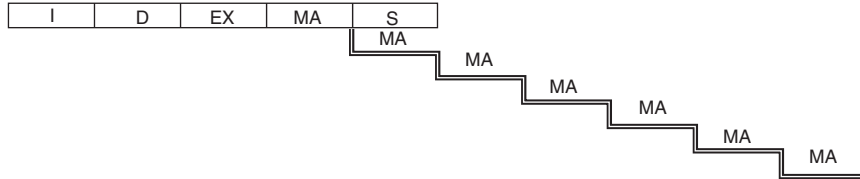
10. OCBI: 1 issue cycle



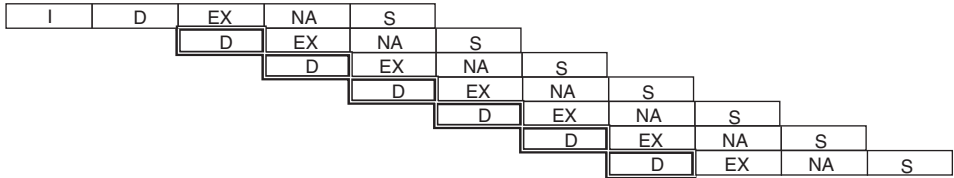
11. OCBP, OCBWB: 1 issue cycle



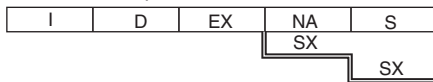
12. MOVCA.L: 1 issue cycle



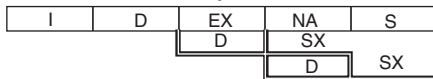
13. TRAPA: 7 issue cycles



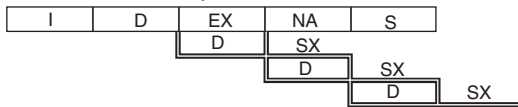
14. LDC to DBR/Rp_BANK/SSR/SPC/VBR, BSR: 1 issue cycle



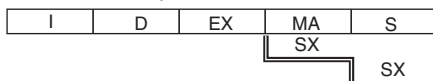
15. LDC to GBR: 3 issue cycles



16. LDC to SR: 4 issue cycles



17. LDC.L to DBR/Rp_BANK/SSR/SPC/VBR: 1 issue cycle



18. LDC.L to GBR: 3 issue cycles

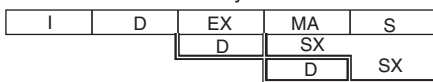
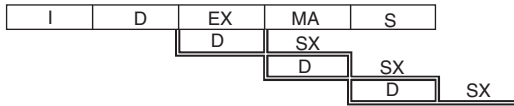
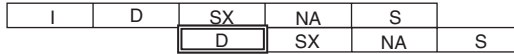


Figure 8.2 Instruction Execution Patterns (cont)

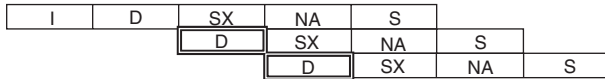
19. LDC.L to SR: 4 issue cycles



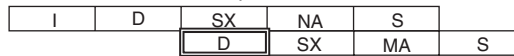
20. STC from DBR/GBR/Rp_BANK/SR/SSR/SPC/VBR: 2 issue cycles



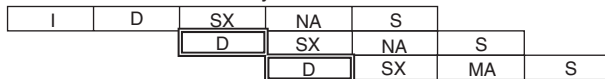
21. STC.L from SGR: 3 issue cycles



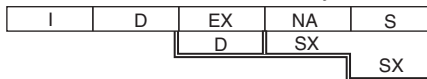
22. STC.L from DBR/GBR/Rp_BANK/SR/SSR/SPC/VBR: 2 issue cycles



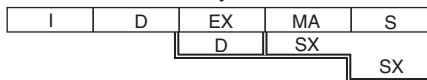
23. STC.L from SGR: 3 issue cycles



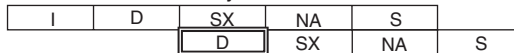
24. LDS to PR, JSR, BSRF: 2 issue cycles



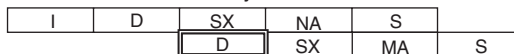
25. LDS.L to PR: 2 issue cycles



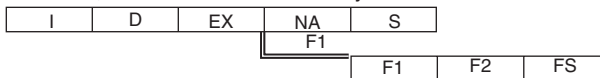
26. STS from PR: 2 issue cycles



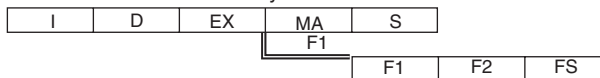
27. STS.L from PR: 2 issue cycles



28. CLRMAC, LDS to MACH/L: 1 issue cycle



29. LDS.L to MACH/L: 1 issue cycle



30. STS from MACH/L: 1 issue cycle



Figure 8.2 Instruction Execution Patterns (cont)

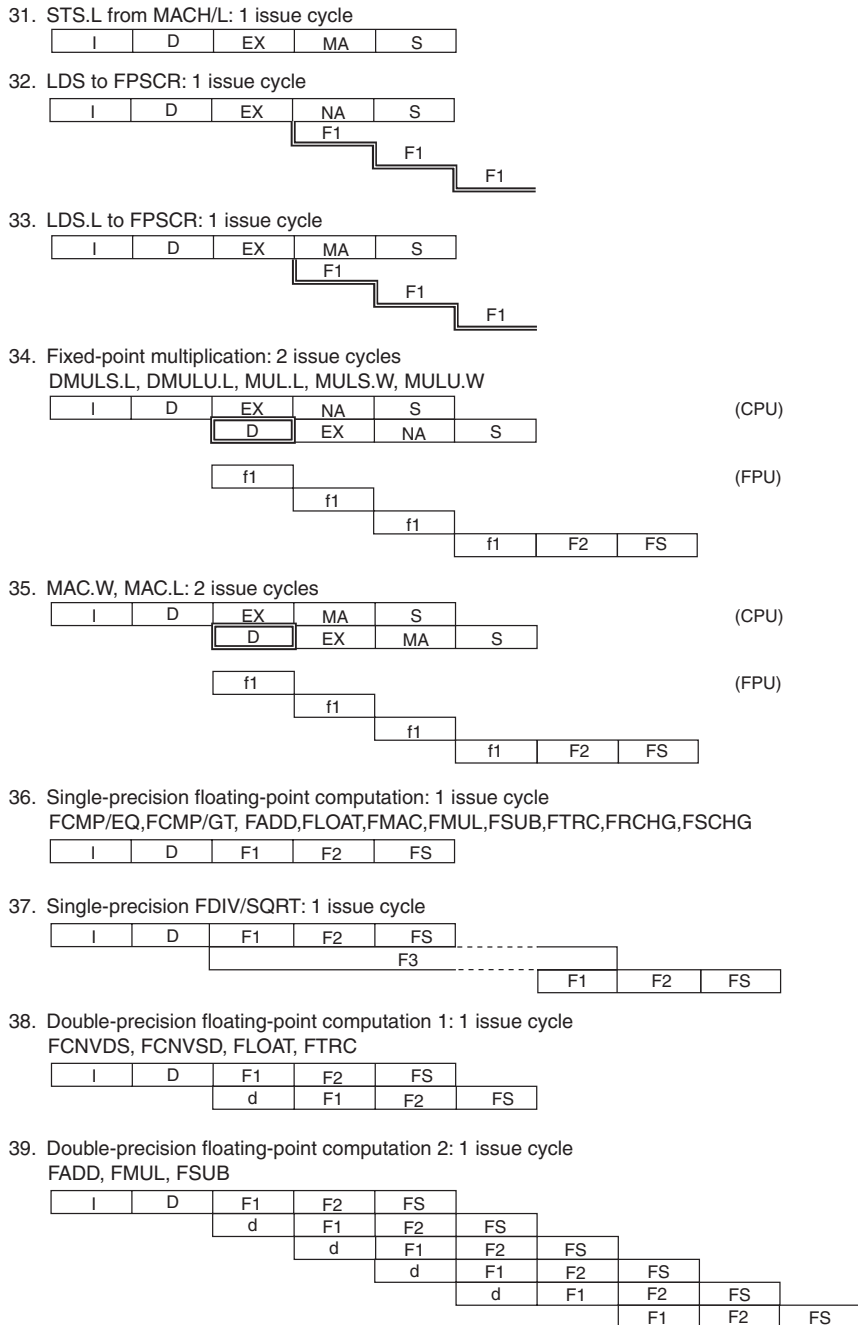
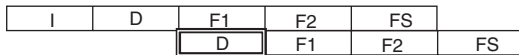


Figure 8.2 Instruction Execution Patterns (cont)

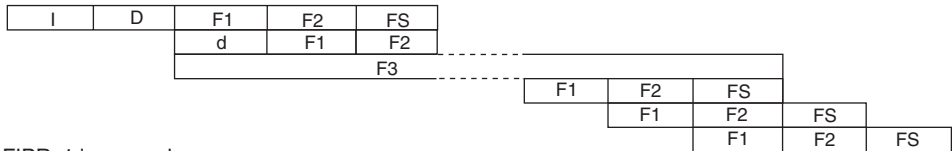
40. Double-precision FCMP: 2 issue cycles

FCMP/EQ,FCMP/GT



41. Double-precision FDIV/SQRT: 1 issue cycle

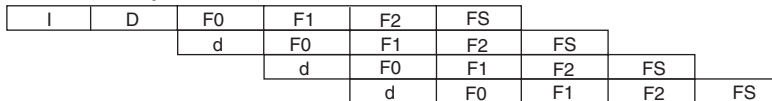
FDIV, FSQRT



42. FIPR: 1 issue cycle



43. FTRV: 1 issue cycle



Notes:

| |
|----|
| ?? |
|----|

 : Cannot overlap a stage of the same kind, except when two instructions are executed in parallel.

| |
|----------|
| D |
|----------|

 : Locks D-stage

| |
|---|
| d |
|---|

 : Register read only

| |
|----|
| ?? |
|----|

 : Locks, but no operation is executed.

| |
|----|
| f1 |
|----|

 : Can overlap another f1, but not another F1.

Figure 8.2 Instruction Execution Patterns (cont)

8.2 Parallel-Executability

Instructions are categorized into six groups according to the internal function blocks used, as shown in table 8.1. Table 8.2 shows the parallel-executability of pairs of instructions in terms of groups. For example, ADD in the EX group and BRA in the BR group can be executed in parallel.

Table 8.1 Instruction Groups

1. MT Group

| | | | | | |
|--------|---------|---------|-------|------|---------|
| CLRT | | CMP/HI | Rm,Rn | MOV | Rm,Rn |
| CMP/EQ | #imm,R0 | CMP/HS | Rm,Rn | NOP | |
| CMP/EQ | Rm,Rn | CMP/PL | Rn | SETT | |
| CMP/GE | Rm,Rn | CMP/PZ | Rn | TST | #imm,R0 |
| CMP/GT | Rm,Rn | CMP/STR | Rm,Rn | TST | Rm,Rn |

2. EX Group

| | | | | | |
|--------|---------------|------------------|---------|--------|---------|
| ADD | #imm,Rn | MOV _T | Rn | SHLL2 | Rn |
| ADD | Rm,Rn | NEG | Rm,Rn | SHLL8 | Rn |
| ADDC | Rm,Rn | NEGC | Rm,Rn | SHLR | Rn |
| ADDV | Rm,Rn | NOT | Rm,Rn | SHLR16 | Rn |
| AND | #imm,R0 | OR | #imm,R0 | SHLR2 | Rn |
| AND | Rm,Rn | OR | Rm,Rn | SHLR8 | Rn |
| DIV0S | Rm,Rn | ROTCL | Rn | SUB | Rm,Rn |
| DIV0U | | ROTCR | Rn | SUBC | Rm,Rn |
| DIV1 | Rm,Rn | ROTL | Rn | SUBV | Rm,Rn |
| DT | Rn | ROTR | Rn | SWAP.B | Rm,Rn |
| EXTS.B | Rm,Rn | SHAD | Rm,Rn | SWAP.W | Rm,Rn |
| EXTS.W | Rm,Rn | SHAL | Rn | XOR | #imm,R0 |
| EXTU.B | Rm,Rn | SHAR | Rn | XOR | Rm,Rn |
| EXTU.W | Rm,Rn | SHLD | Rm,Rn | XTRCT | Rm,Rn |
| MOV | #imm,Rn | SHLL | Rn | | |
| MOVA | @(disp,PC),R0 | SHLL16 | Rn | | |

3. BR Group

| | | | | | |
|------|------|-----|------|------|------|
| BF | disp | BRA | disp | BT | disp |
| BF/S | disp | BSR | disp | BT/S | disp |

4. LS Group

| | | | | | |
|--------|--------------|--------|----------------|---------|----------------|
| FABS | DRn | FMOV.S | @Rm+,FRn | MOV.L | R0,@(disp,GBR) |
| FABS | FRn | FMOV.S | FRm,@(R0,Rn) | MOV.L | Rm,@(disp,Rn) |
| FLDI0 | FRn | FMOV.S | FRm,@-Rn | MOV.L | Rm,@(R0,Rn) |
| FLDI1 | FRn | FMOV.S | FRm,@Rn | MOV.L | Rm,@-Rn |
| FLDS | FRm,FPUL | FNEG | DRn | MOV.L | Rm,@Rn |
| FMOV | @(R0,Rm),DRn | FNEG | FRn | MOV.W | @(disp,GBR),R0 |
| FMOV | @(R0,Rm),XDn | FSTS | FPUL,FRn | MOV.W | @(disp,PC),Rn |
| FMOV | @Rm,DRn | LDS | Rm,FPUL | MOV.W | @(disp,Rm),R0 |
| FMOV | @Rm,XDn | MOV.B | @(disp,GBR),R0 | MOV.W | @(R0,Rm),Rn |
| FMOV | @Rm+,DRn | MOV.B | @(disp,Rm),R0 | MOV.W | @Rm,Rn |
| FMOV | @Rm+,XDn | MOV.B | @(R0,Rm),Rn | MOV.W | @Rm+,Rn |
| FMOV | DRm,@(R0,Rn) | MOV.B | @Rm,Rn | MOV.W | R0,@(disp,GBR) |
| FMOV | DRm,@-Rn | MOV.B | @Rm+,Rn | MOV.W | R0,@(disp,Rn) |
| FMOV | DRm,@Rn | MOV.B | R0,@(disp,GBR) | MOV.W | Rm,@(R0,Rn) |
| FMOV | DRm,DRn | MOV.B | R0,@(disp,Rn) | MOV.W | Rm,@-Rn |
| FMOV | DRm,XDn | MOV.B | Rm,@(R0,Rn) | MOV.W | Rm,@Rn |
| FMOV | FRm,FRn | MOV.B | Rm,@-Rn | MOVCA.L | R0,@Rn |
| FMOV | XDm,@(R0,Rn) | MOV.B | Rm,@Rn | OCBI | @Rn |
| FMOV | XDm,@-Rn | MOV.L | @(disp,GBR),R0 | OCBP | @Rn |
| FMOV | XDm,@Rn | MOV.L | @(disp,PC),Rn | OCBWB | @Rn |
| FMOV | XDm,DRn | MOV.L | @(disp,Rm),Rn | PREF | @Rn |
| FMOV | XDm,XDn | MOV.L | @(R0,Rm),Rn | STS | FPUL,Rn |
| FMOV.S | @(R0,Rm),FRn | MOV.L | @Rm,Rn | | |
| FMOV.S | @Rm,FRn | MOV.L | @Rm+,Rn | | |

5. FE Group

| | | | | | |
|---------|----------|-------|-------------|-------|-----------|
| FADD | DRm,DRn | FIPR | FVm,FVn | FSQRT | DRn |
| FADD | FRm,FRn | FLOAT | FPUL,DRn | FSQRT | FRn |
| FCMP/EQ | FRm,FRn | FLOAT | FPUL,FRn | FSUB | DRm,DRn |
| FCMP/GT | FRm,FRn | FMAC | FR0,FRm,FRn | FSUB | FRm,FRn |
| FCNVDS | DRm,FPUL | FMUL | DRm,DRn | FTRC | DRm,FPUL |
| FCNVSD | FPUL,DRn | FMUL | FRm,FRn | FTRC | FRm,FPUL |
| FDIV | DRm,DRn | FRCHG | | FTRV | XMTRX,FVn |
| FDIV | FRm,FRn | FSCHG | | | |

6. CO Group

| | | | | | |
|---------|----------------|--------|----------------|-------|----------------|
| AND.B | #imm,@(R0,GBR) | LDS | Rm,FPSCR | STC | SR,Rn |
| BRAF | Rm | LDS | Rm,MACH | STC | SSR,Rn |
| BSRF | Rm | LDS | Rm,MACL | STC | VBR,Rn |
| CLRMAC | | LDS | Rm,PR | STC.L | DBR,@-Rn |
| CLRS | | LDS.L | @Rm+,FPSCR | STC.L | GBR,@-Rn |
| DMULS.L | Rm,Rn | LDS.L | @Rm+,FPUL | STC.L | Rp_BANK,@-Rn |
| DMULU.L | Rm,Rn | LDS.L | @Rm+,MACH | STC.L | SGR,@-Rn |
| FCMP/EQ | DRm,DRn | LDS.L | @Rm+,MACL | STC.L | SPC,@-Rn |
| FCMP/GT | DRm,DRn | LDS.L | @Rm+,PR | STC.L | SR,@-Rn |
| JMP | @Rn | LDTLB | | STC.L | SSR,@-Rn |
| JSR | @Rn | MAC.L | @Rm+,@Rn+ | STC.L | VBR,@-Rn |
| LDC | Rm,DBR | MAC.W | @Rm+,@Rn+ | STS | FPSCR,Rn |
| LDC | Rm,GBR | MUL.L | Rm,Rn | STS | MACH,Rn |
| LDC | Rm,Rp_BANK | MULS.W | Rm,Rn | STS | MACL,Rn |
| LDC | Rm,SPC | MULU.W | Rm,Rn | STS | PR,Rn |
| LDC | Rm,SR | OR.B | #imm,@(R0,GBR) | STS.L | FPSCR,@-Rn |
| LDC | Rm,SSR | RTE | | STS.L | FPUL,@-Rn |
| LDC | Rm,VBR | RTS | | STS.L | MACH,@-Rn |
| LDC.L | @Rm+,DBR | SETS | | STS.L | MACL,@-Rn |
| LDC.L | @Rm+,GBR | SLEEP | | STS.L | PR,@-Rn |
| LDC.L | @Rm+,Rp_BANK | STC | DBR,Rn | TAS.B | @Rn |
| LDC.L | @Rm+,SPC | STC | GBR,Rn | TRAPA | #imm |
| LDC.L | @Rm+,SR | STC | Rp_BANK,Rn | TST.B | #imm,@(R0,GBR) |
| LDC.L | @Rm+,SSR | STC | SGR,Rn | XOR.B | #imm,@(R0,GBR) |
| LDC.L | @Rm+,VBR | STC | SPC,Rn | | |

Table 8.2 Parallel-Executability

| | | 2nd Instruction | | | | | |
|-----------------|----|-----------------|----|----|----|----|----|
| | | MT | EX | BR | LS | FE | CO |
| 1st Instruction | MT | O | O | O | O | O | X |
| | EX | O | X | O | O | O | X |
| | BR | O | O | X | O | O | X |
| | LS | O | O | O | X | O | X |
| | FE | O | O | O | O | X | X |
| | CO | X | X | X | X | X | X |

Legend:

O: Can be executed in parallel

X: Cannot be executed in parallel

8.3 Execution Cycles and Pipeline Stalling

There are three basic clocks in this processor: the I-clock, B-clock, and P-clock. Each hardware unit operates on one of these clocks, as follows:

- I-clock: CPU, FPU, MMU, caches
- B-clock: External bus controller
- P-clock: Peripheral units

The frequency ratios of the three clocks are determined with the frequency control register (FRQCR). In this section, machine cycles are based on the I-clock unless otherwise specified. For details of FRQCR, see section 10, Clock Oscillation Circuits.

Instruction execution cycles are summarized in table 8.3. Penalty cycles due to a pipeline stall or freeze are not considered in this table.

- Issue rate: Interval between the issue of an instruction and that of the next instruction
- Latency: Interval between the issue of an instruction and the generation of its result (completion)
- Instruction execution pattern (see figure 8.2)
- Lock stage: Locked pipeline stages(see table 8.3)
- Lock start: Interval between the issue of an instruction and the start of locking (see table 8.3)
- Lock cycle: Lock time (see table 8.3)

The instruction execution sequence is expressed as a combination of the execution patterns shown in figure 8.2. One instruction is separated from the next by the number of machine cycles for its issue rate. Normally, execution, data access, and write-back stages cannot be overlapped onto the same stages of another instruction; the only exception is when two instructions are executed in parallel under parallel-executability conditions. Refer to (a) through (d) in figure 8.3 for some simple examples.

Latency is the interval between issue and completion of an instruction, and is also the interval between the execution of two instructions with an interdependent relationship. When there is interdependency between two instructions fetched simultaneously, the latter of the two is stalled for the following number of cycles:

- (Latency) cycles when there is flow dependency (read-after-write)
- (Latency - 1) or (latency - 2) cycles when there is output dependency (write-after-write)
 - Single/double-precision FDIV, FSQRT is the preceding instruction (latency - 1) cycles
 - The other FE group except above is the preceding instruction (latency - 2) cycles
- 5 or 2 cycles when there is anti-flow dependency (write-after-read), as in the following cases:
 - FTRV is the preceding instruction (5 cycles)
 - A double-precision FADD, FSUB, or FMUL is the preceding instruction (2 cycles)

In the case of flow dependency, latency may be exceptionally increased or decreased, depending on the combination of sequential instructions (figure 8.3 (e)).

- When a floating-point computation is followed by a floating-point register store, the latency of the floating-point computation may be decreased by 1 cycle.
- If there is a load of the shift amount immediately before an SHAD/SHLD instruction, the latency of the load is increased by 1 cycle.
- If an instruction with a latency of less than 2 cycles, including write-back to a floating-point register, is followed by a double-precision floating-point instruction, FIPR, or FTRV, the latency of the first instruction is increased to 2 cycles.

The number of cycles in a pipeline stall due to flow dependency will vary depending on the combination of interdependent instructions or the fetch timing (see figure 8.3. (e)).

Output dependency occurs when the destination operands are the same in a preceding FE group instruction and a following LS group instruction.

For the stall cycles of an instruction with output dependency, the longest latency to the last write-back among all the destination operands must be applied instead of “latency” (see figure 8.3 (f)). A stall due to output dependency with respect to FPSCR, which reflects the result of a floating-

point operation, never occurs. For example, when FADD follows FDIV with no dependency between floating-point registers, FADD is not stalled even if both instructions update the cause field of FPSCR.

Anti-flow dependency can occur only between a preceding double-precision FADD, FMUL, FSUB, or FTRV and a following FMOV, FLDI0, FLDI1, FABS, FNEG, or FSTS. See figure 8.3 (g).

If an executing instruction locks any resource—i.e. a function block that performs a basic operation—a following instruction that attempts to use the locked resource is stalled (figure 8.3 (h)). This kind of stall can be compensated by inserting one or more instructions independent of the locked resource to separate the interfering instructions. For example, when a load instruction and an ADD instruction that references the loaded value are consecutive, the 2-cycle stall of the ADD is eliminated by inserting three instructions without dependency. Software performance can be improved by such instruction scheduling.

Other causes of a stall are as follows.

- Instruction TLB miss
- Instruction access to external memory (instruction cache miss, etc.)
- Data access to external memory (operand cache miss, etc.)
- Data access to a memory-mapped control register

During the penalty cycles of an instruction TLB miss or external instruction access, no instruction is issued, but execution of instructions that have already been issued continues. The penalty for a data access is a pipeline freeze: that is, the execution of uncompleted instructions is interrupted until the arrival of the requested data. The number of penalty cycles for instruction and data accesses is largely dependent on the user's memory subsystems.

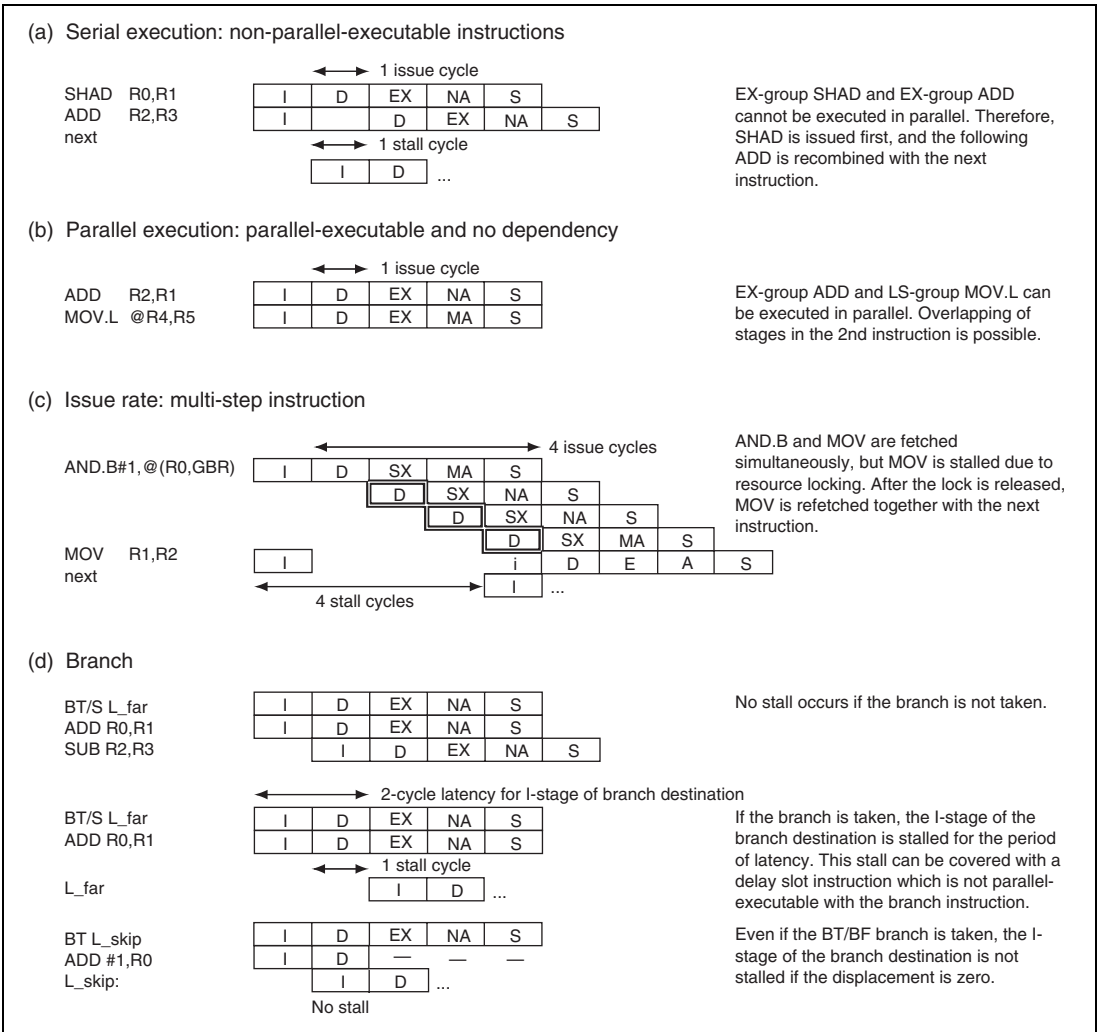
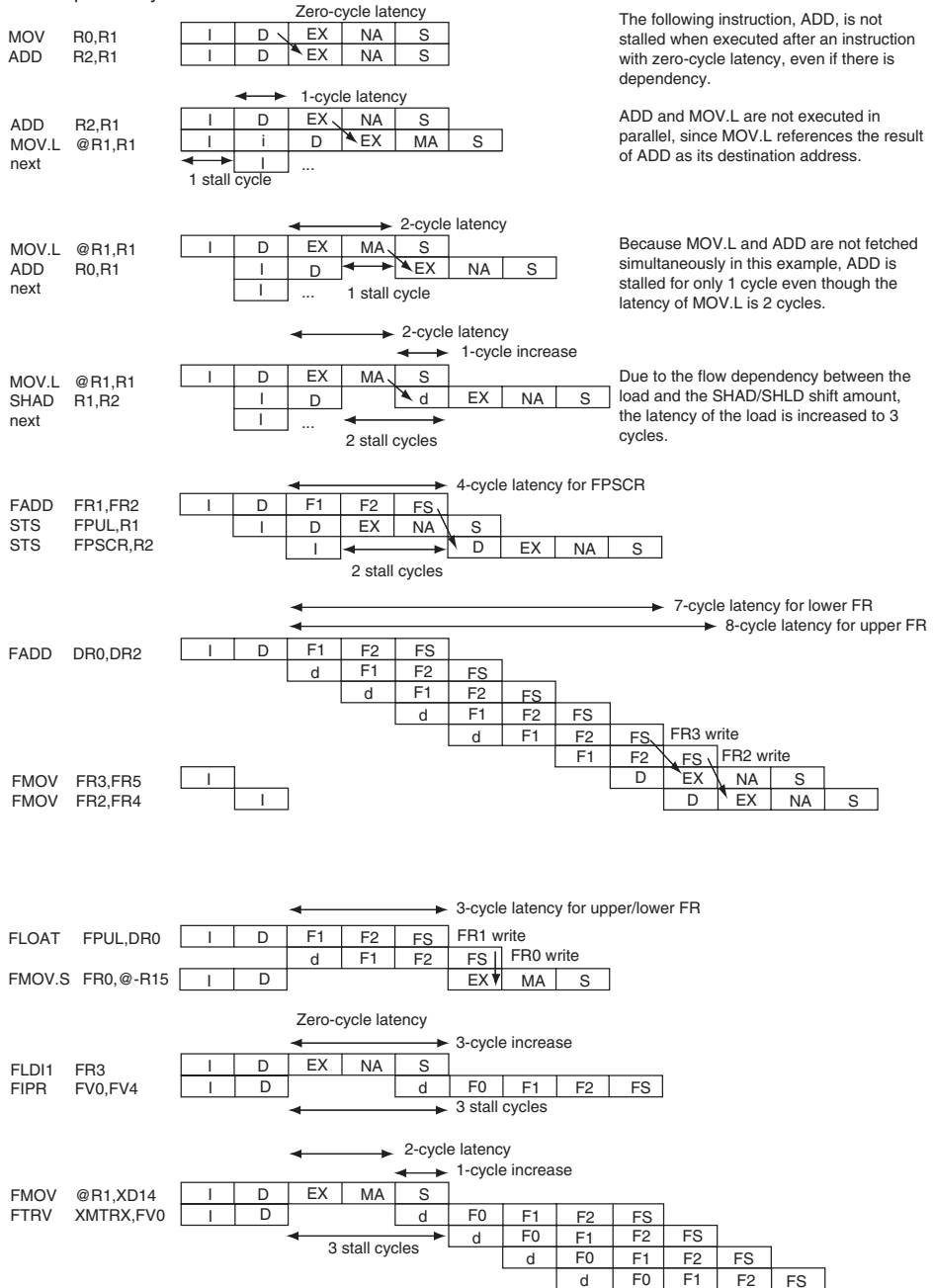


Figure 8.3 Examples of Pipelined Execution

(e) Flow dependency



The following instruction, ADD, is not stalled when executed after an instruction with zero-cycle latency, even if there is dependency.

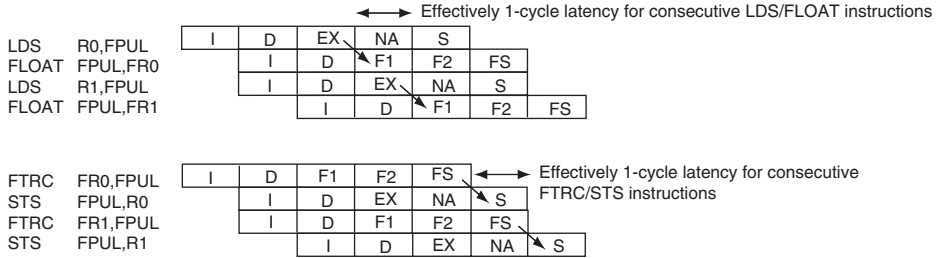
ADD and MOV.L are not executed in parallel, since MOV.L references the result of ADD as its destination address.

Because MOV.L and ADD are not fetched simultaneously in this example, ADD is stalled for only 1 cycle even though the latency of MOV.L is 2 cycles.

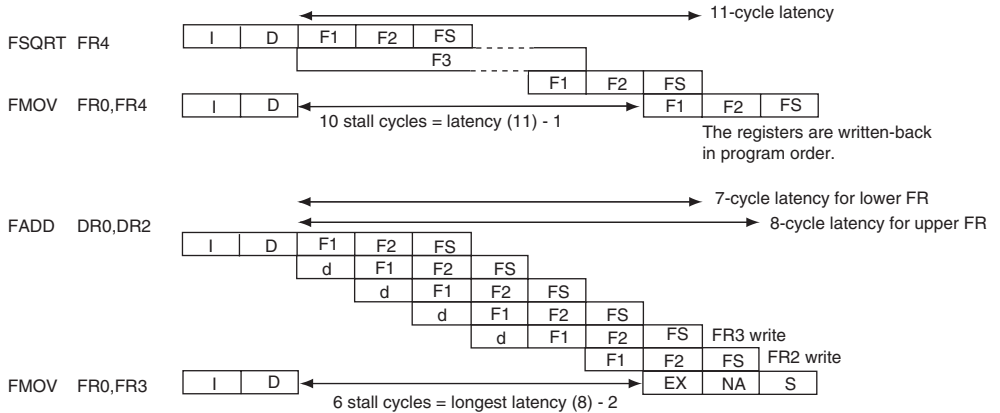
Due to the flow dependency between the load and the SHAD/SHLD shift amount, the latency of the load is increased to 3 cycles.

Figure 8.3 Examples of Pipelined Execution (cont)

(e) Flow dependency (cont)



(f) Output dependency



(g) Anti-flow dependency

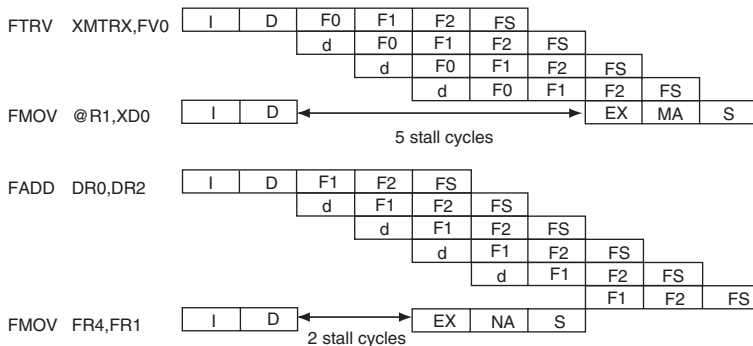


Figure 8.3 Examples of Pipelined Execution (cont)

(h) Resource conflict

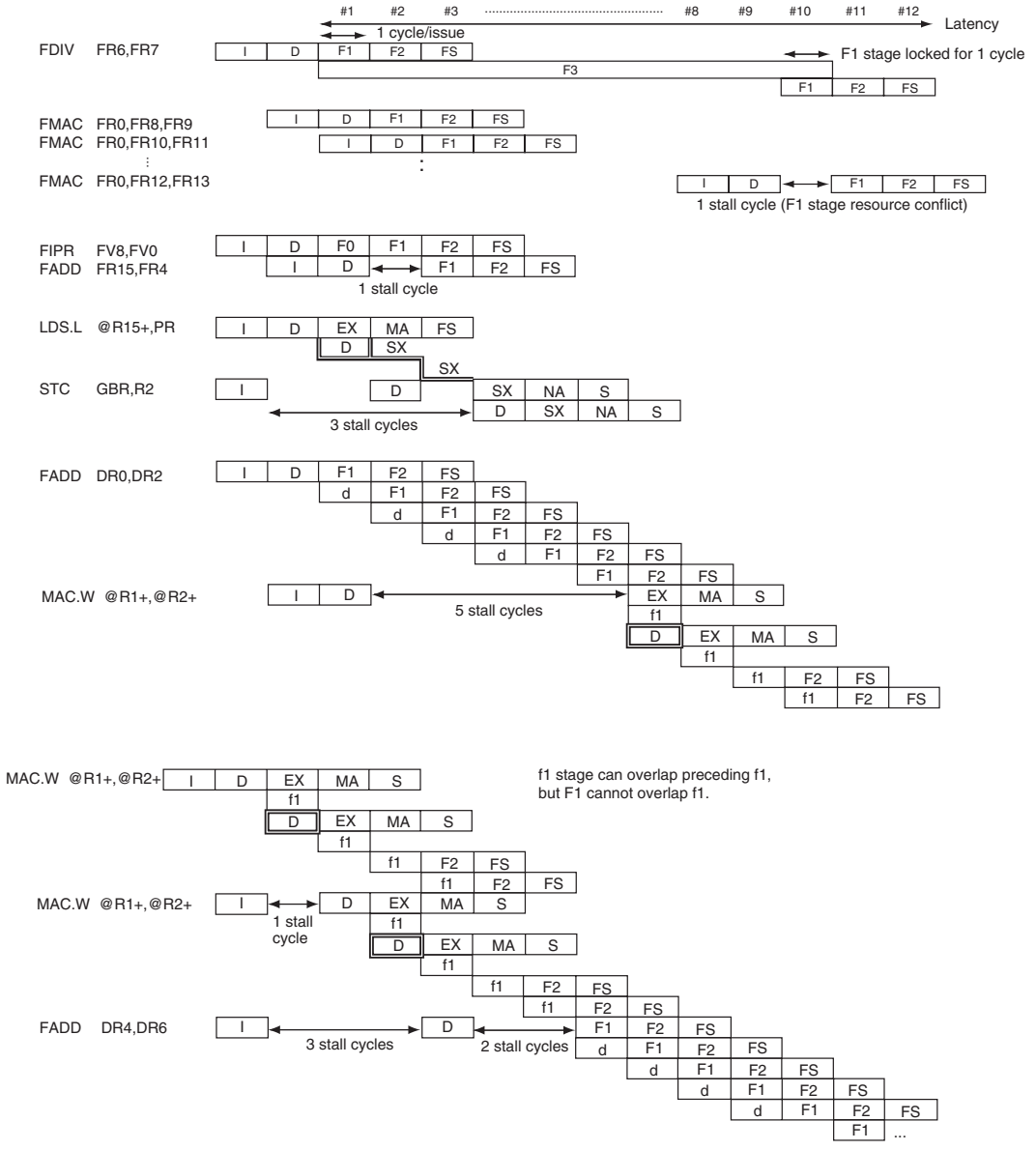


Figure 8.3 Examples of Pipelined Execution (cont)

Table 8.3 Execution Cycles

| Functional Category | No. | Instruction | Instruction Group | Issue Rate | Latency | Execution Pattern | Lock | | |
|----------------------------|-----|----------------------|-------------------|------------|---------|-------------------|-------|-------|--------|
| | | | | | | | Stage | Start | Cycles |
| Data transfer instructions | 1 | EXTS.B Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 2 | EXTS.W Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 3 | EXTU.B Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 4 | EXTU.W Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 5 | MOV Rm,Rn | MT | 1 | 0 | #1 | — | — | — |
| | 6 | MOV #imm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 7 | MOVA @(disp,PC),R0 | EX | 1 | 1 | #1 | — | — | — |
| | 8 | MOV.W @(disp,PC),Rn | LS | 1 | 2 | #2 | — | — | — |
| | 9 | MOV.L @(disp,PC),Rn | LS | 1 | 2 | #2 | — | — | — |
| | 10 | MOV.B @Rm,Rn | LS | 1 | 2 | #2 | — | — | — |
| | 11 | MOV.W @Rm,Rn | LS | 1 | 2 | #2 | — | — | — |
| | 12 | MOV.L @Rm,Rn | LS | 1 | 2 | #2 | — | — | — |
| | 13 | MOV.B @Rm+,Rn | LS | 1 | 1/2 | #2 | — | — | — |
| | 14 | MOV.W @Rm+,Rn | LS | 1 | 1/2 | #2 | — | — | — |
| | 15 | MOV.L @Rm+,Rn | LS | 1 | 1/2 | #2 | — | — | — |
| | 16 | MOV.B @(disp,Rm),R0 | LS | 1 | 2 | #2 | — | — | — |
| | 17 | MOV.W @(disp,Rm),R0 | LS | 1 | 2 | #2 | — | — | — |
| | 18 | MOV.L @(disp,Rm),Rn | LS | 1 | 2 | #2 | — | — | — |
| | 19 | MOV.B @(R0,Rm),Rn | LS | 1 | 2 | #2 | — | — | — |
| | 20 | MOV.W @(R0,Rm),Rn | LS | 1 | 2 | #2 | — | — | — |
| | 21 | MOV.L @(R0,Rm),Rn | LS | 1 | 2 | #2 | — | — | — |
| | 22 | MOV.B @(disp,GBR),R0 | LS | 1 | 2 | #3 | — | — | — |
| | 23 | MOV.W @(disp,GBR),R0 | LS | 1 | 2 | #3 | — | — | — |
| | 24 | MOV.L @(disp,GBR),R0 | LS | 1 | 2 | #3 | — | — | — |
| | 25 | MOV.B Rm,@Rn | LS | 1 | 1 | #2 | — | — | — |
| | 26 | MOV.W Rm,@Rn | LS | 1 | 1 | #2 | — | — | — |
| | 27 | MOV.L Rm,@Rn | LS | 1 | 1 | #2 | — | — | — |
| | 28 | MOV.B Rm,@-Rn | LS | 1 | 1/1 | #2 | — | — | — |
| | 29 | MOV.W Rm,@-Rn | LS | 1 | 1/1 | #2 | — | — | — |
| | 30 | MOV.L Rm,@-Rn | LS | 1 | 1/1 | #2 | — | — | — |
| | 31 | MOV.B R0,@(disp,Rn) | LS | 1 | 1 | #2 | — | — | — |

| Functional Category | No. | Instruction | Instruc- tion Group | Issue Rate | Latency | Execu- tion Pattern | Lock | | |
|----------------------------------|---|----------------------|---------------------------|---------------|---------|---------------------------|-------|-------|--------|
| | | | | | | | Stage | Start | Cycles |
| Data transfer instructions | 32 | MOV.W R0,@(disp,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 33 | MOV.L Rm,@(disp,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 34 | MOV.B Rm,@(R0,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 35 | MOV.W Rm,@(R0,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 36 | MOV.L Rm,@(R0,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 37 | MOV.B R0,@(disp,GBR) | LS | 1 | 1 | #3 | — | — | — |
| | 38 | MOV.W R0,@(disp,GBR) | LS | 1 | 1 | #3 | — | — | — |
| | 39 | MOV.L R0,@(disp,GBR) | LS | 1 | 1 | #3 | — | — | — |
| | 40 | MOVCA.L R0,@Rn | LS | 1 | 3–7 | #12 | MA | 4 | 3–7 |
| | 41 | MOVT Rn | EX | 1 | 1 | #1 | — | — | — |
| | 42 | OCBI @Rn | LS | 1 | 1–2 | #10 | MA | 4 | 1–2 |
| | 43 | OCBP @Rn | LS | 1 | 1–5 | #11 | MA | 4 | 1–5 |
| | 44 | OCBWB @Rn | LS | 1 | 1–5 | #11 | MA | 4 | 1–5 |
| | 45 | PREF @Rn | LS | 1 | 1 | #2 | — | — | — |
| | 46 | SWAP.B Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 47 | SWAP.W Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 48 | XTRCT Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | Fixed-point arithmetic instructions | 49 | ADD Rm,Rn | EX | 1 | 1 | #1 | — | — |
| 50 | | ADD #imm,Rn | EX | 1 | 1 | #1 | — | — | — |
| 51 | | ADDC Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| 52 | | ADDV Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| 53 | | CMP/EQ #imm,R0 | MT | 1 | 1 | #1 | — | — | — |
| 54 | | CMP/EQ Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| 55 | | CMP/GE Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| 56 | | CMP/GT Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| 57 | | CMP/HI Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| 58 | | CMP/HS Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| 59 | | CMP/PL Rn | MT | 1 | 1 | #1 | — | — | — |
| 60 | | CMP/PZ Rn | MT | 1 | 1 | #1 | — | — | — |
| 61 | | CMP/STR Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| 62 | | DIV0S Rm,Rn | EX | 1 | 1 | #1 | — | — | — |

| Functional Category | No. | Instruction | Instruction Group | Issue Rate | Latency | Execution Pattern | Lock | | |
|-------------------------------------|------------|----------------------|-------------------|------------|---------|-------------------|-------|-------|--------|
| | | | | | | | Stage | Start | Cycles |
| Fixed-point arithmetic instructions | 63 | DIV0U | EX | 1 | 1 | #1 | — | — | — |
| | 64 | DIV1 Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 65 | DMULS.L Rm,Rn | CO | 2 | 4/4 | #34 | F1 | 4 | 2 |
| | 66 | DMULU.L Rm,Rn | CO | 2 | 4/4 | #34 | F1 | 4 | 2 |
| | 67 | DT Rn | EX | 1 | 1 | #1 | — | — | — |
| | 68 | MAC.L @Rm+,@Rn+ | CO | 2 | 2/2/4/4 | #35 | F1 | 4 | 2 |
| | 69 | MAC.W @Rm+,@Rn+ | CO | 2 | 2/2/4/4 | #35 | F1 | 4 | 2 |
| | 70 | MUL.L Rm,Rn | CO | 2 | 4/4 | #34 | F1 | 4 | 2 |
| | 71 | MULS.W Rm,Rn | CO | 2 | 4/4 | #34 | F1 | 4 | 2 |
| | 72 | MULU.W Rm,Rn | CO | 2 | 4/4 | #34 | F1 | 4 | 2 |
| | 73 | NEG Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 74 | NEGC Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 75 | SUB Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 76 | SUBC Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| 77 | SUBV Rm,Rn | EX | 1 | 1 | #1 | — | — | — | |
| Logical instructions | 78 | AND Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 79 | AND #imm,R0 | EX | 1 | 1 | #1 | — | — | — |
| | 80 | AND.B #imm,@(R0,GBR) | CO | 4 | 4 | #6 | — | — | — |
| | 81 | NOT Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 82 | OR Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 83 | OR #imm,R0 | EX | 1 | 1 | #1 | — | — | — |
| | 84 | OR.B #imm,@(R0,GBR) | CO | 4 | 4 | #6 | — | — | — |
| | 85 | TAS.B @Rn | CO | 5 | 5 | #7 | — | — | — |
| | 86 | TST Rm,Rn | MT | 1 | 1 | #1 | — | — | — |
| | 87 | TST #imm,R0 | MT | 1 | 1 | #1 | — | — | — |
| | 88 | TST.B #imm,@(R0,GBR) | CO | 3 | 3 | #5 | — | — | — |
| | 89 | XOR Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 90 | XOR #imm,R0 | EX | 1 | 1 | #1 | — | — | — |
| | 91 | XOR.B #imm,@(R0,GBR) | CO | 4 | 4 | #6 | — | — | — |

| Functional Category | No. | Instruction | Instruc- tion Group | Issue Rate | Latency | Execu- tion Pattern | Lock | | |
|------------------------|-----------|-------------|---------------------------|---------------|----------|---------------------------|-------|-------|--------|
| | | | | | | | Stage | Start | Cycles |
| Shift instructions | 92 | ROTL Rn | EX | 1 | 1 | #1 | — | — | — |
| | 93 | ROTR Rn | EX | 1 | 1 | #1 | — | — | — |
| | 94 | ROTCL Rn | EX | 1 | 1 | #1 | — | — | — |
| | 95 | ROTCR Rn | EX | 1 | 1 | #1 | — | — | — |
| | 96 | SHAD Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 97 | SHAL Rn | EX | 1 | 1 | #1 | — | — | — |
| | 98 | SHAR Rn | EX | 1 | 1 | #1 | — | — | — |
| | 99 | SHLD Rm,Rn | EX | 1 | 1 | #1 | — | — | — |
| | 100 | SHLL Rn | EX | 1 | 1 | #1 | — | — | — |
| | 101 | SHLL2 Rn | EX | 1 | 1 | #1 | — | — | — |
| | 102 | SHLL8 Rn | EX | 1 | 1 | #1 | — | — | — |
| | 103 | SHLL16 Rn | EX | 1 | 1 | #1 | — | — | — |
| | 104 | SHLR Rn | EX | 1 | 1 | #1 | — | — | — |
| | 105 | SHLR2 Rn | EX | 1 | 1 | #1 | — | — | — |
| 106 | SHLR8 Rn | EX | 1 | 1 | #1 | — | — | — | |
| 107 | SHLR16 Rn | EX | 1 | 1 | #1 | — | — | — | |
| Branch instructions | 108 | BF disp | BR | 1 | 2 (or 1) | #1 | — | — | — |
| | 109 | BF/S disp | BR | 1 | 2 (or 1) | #1 | — | — | — |
| | 110 | BT disp | BR | 1 | 2 (or 1) | #1 | — | — | — |
| | 111 | BT/S disp | BR | 1 | 2 (or 1) | #1 | — | — | — |
| | 112 | BRA disp | BR | 1 | 2 | #1 | — | — | — |
| | 113 | BRAF Rn | CO | 2 | 3 | #4 | — | — | — |
| | 114 | BSR disp | BR | 1 | 2 | #14 | SX | 3 | 2 |
| | 115 | BSRF Rn | CO | 2 | 3 | #24 | SX | 3 | 2 |
| | 116 | JMP @Rn | CO | 2 | 3 | #4 | — | — | — |
| | 117 | JSR @Rn | CO | 2 | 3 | #24 | SX | 3 | 2 |
| | 118 | RTS | CO | 2 | 3 | #4 | — | — | — |

| Functional Category | No. | Instruction | Instruc- tion Group | Issue Rate | Latency | Execu- tion Pattern | Lock | | |
|-----------------------------------|------------|--------------------|---------------------------|---------------|---------|---------------------------|-------|-------|--------|
| | | | | | | | Stage | Start | Cycles |
| System control instructions | 119 | NOP | MT | 1 | 0 | #1 | — | — | — |
| | 120 | CLRMAC | CO | 1 | 3 | #28 | F1 | 3 | 2 |
| | 121 | CLRS | CO | 1 | 1 | #1 | — | — | — |
| | 122 | CLRT | MT | 1 | 1 | #1 | — | — | — |
| | 123 | SETS | CO | 1 | 1 | #1 | — | — | — |
| | 124 | SETT | MT | 1 | 1 | #1 | — | — | — |
| | 125 | TRAPA #imm | CO | 7 | 7 | #13 | — | — | — |
| | 126 | RTE | CO | 5 | 5 | #8 | — | — | — |
| | 127 | SLEEP | CO | 4 | 4 | #9 | — | — | — |
| | 128 | LDTLB | CO | 1 | 1 | #2 | — | — | — |
| | 129 | LDC Rm,DBR | CO | 1 | 3 | #14 | SX | 3 | 2 |
| | 130 | LDC Rm,GBR | CO | 3 | 3 | #15 | SX | 3 | 2 |
| | 131 | LDC Rm,Rp_BANK | CO | 1 | 3 | #14 | SX | 3 | 2 |
| | 132 | LDC Rm,SR | CO | 4 | 4 | #16 | SX | 3 | 2 |
| | 133 | LDC Rm,SSR | CO | 1 | 3 | #14 | SX | 3 | 2 |
| | 134 | LDC Rm,SPC | CO | 1 | 3 | #14 | SX | 3 | 2 |
| | 135 | LDC Rm,VBR | CO | 1 | 3 | #14 | SX | 3 | 2 |
| | 136 | LDC.L @Rm+,DBR | CO | 1 | 1/3 | #17 | SX | 3 | 2 |
| | 137 | LDC.L @Rm+,GBR | CO | 3 | 3/3 | #18 | SX | 3 | 2 |
| | 138 | LDC.L @Rm+,Rp_BANK | CO | 1 | 1/3 | #17 | SX | 3 | 2 |
| | 139 | LDC.L @Rm+,SR | CO | 4 | 4/4 | #19 | SX | 3 | 2 |
| | 140 | LDC.L @Rm+,SSR | CO | 1 | 1/3 | #17 | SX | 3 | 2 |
| | 141 | LDC.L @Rm+,SPC | CO | 1 | 1/3 | #17 | SX | 3 | 2 |
| | 142 | LDC.L @Rm+,VBR | CO | 1 | 1/3 | #17 | SX | 3 | 2 |
| | 143 | LDS Rm,MACH | CO | 1 | 3 | #28 | F1 | 3 | 2 |
| | 144 | LDS Rm,MACL | CO | 1 | 3 | #28 | F1 | 3 | 2 |
| | 145 | LDS Rm,PR | CO | 2 | 3 | #24 | SX | 3 | 2 |
| | 146 | LDS.L @Rm+,MACH | CO | 1 | 1/3 | #29 | F1 | 3 | 2 |
| | 147 | LDS.L @Rm+,MACL | CO | 1 | 1/3 | #29 | F1 | 3 | 2 |
| | 148 | LDS.L @Rm+,PR | CO | 2 | 2/3 | #25 | SX | 3 | 2 |
| 149 | STC DBR,Rn | CO | 2 | 2 | #20 | — | — | — | |
| 150 | STC SGR,Rn | CO | 3 | 3 | #21 | — | — | — | |

| Functional Category | No. | Instruction | Instruction Group | Issue Rate | Latency | Execution Pattern | Lock | | | |
|--|-----|-------------|-------------------|------------|---------|-------------------|-------|-------|--------|---|
| | | | | | | | Stage | Start | Cycles | |
| System control instructions | 151 | STC | GBR,Rn | CO | 2 | 2 | #20 | — | — | — |
| | 152 | STC | Rp_BANK,Rn | CO | 2 | 2 | #20 | — | — | — |
| | 153 | STC | SR,Rn | CO | 2 | 2 | #20 | — | — | — |
| | 154 | STC | SSR,Rn | CO | 2 | 2 | #20 | — | — | — |
| | 155 | STC | SPC,Rn | CO | 2 | 2 | #20 | — | — | — |
| | 156 | STC | VBR,Rn | CO | 2 | 2 | #20 | — | — | — |
| | 157 | STC.L | DBR,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 158 | STC.L | SGR,@-Rn | CO | 3 | 3/3 | #23 | — | — | — |
| | 159 | STC.L | GBR,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 160 | STC.L | Rp_BANK,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 161 | STC.L | SR,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 162 | STC.L | SSR,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 163 | STC.L | SPC,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 164 | STC.L | VBR,@-Rn | CO | 2 | 2/2 | #22 | — | — | — |
| | 165 | STS | MACH,Rn | CO | 1 | 3 | #30 | — | — | — |
| | 166 | STS | MACL,Rn | CO | 1 | 3 | #30 | — | — | — |
| | 167 | STS | PR,Rn | CO | 2 | 2 | #26 | — | — | — |
| | 168 | STS.L | MACH,@-Rn | CO | 1 | 1/1 | #31 | — | — | — |
| | 169 | STS.L | MACL,@-Rn | CO | 1 | 1/1 | #31 | — | — | — |
| | 170 | STS.L | PR,@-Rn | CO | 2 | 2/2 | #27 | — | — | — |
| Single-precision floating-point instructions | 171 | FLDI0 | FRn | LS | 1 | 0 | #1 | — | — | — |
| | 172 | FLDI1 | FRn | LS | 1 | 0 | #1 | — | — | — |
| | 173 | FMOV | FRm,FRn | LS | 1 | 0 | #1 | — | — | — |
| | 174 | FMOV.S | @Rm,FRn | LS | 1 | 2 | #2 | — | — | — |
| | 175 | FMOV.S | @Rm+,FRn | LS | 1 | 1/2 | #2 | — | — | — |
| | 176 | FMOV.S | @(R0,Rm),FRn | LS | 1 | 2 | #2 | — | — | — |
| | 177 | FMOV.S | FRm,@Rn | LS | 1 | 1 | #2 | — | — | — |
| | 178 | FMOV.S | FRm,@-Rn | LS | 1 | 1/1 | #2 | — | — | — |
| | 179 | FMOV.S | FRm,@(R0,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 180 | FLDS | FRm,FPUL | LS | 1 | 0 | #1 | — | — | — |
| | 181 | FSTS | FPUL,FRn | LS | 1 | 0 | #1 | — | — | — |

| Functional Category | No. | Instruction | Instruc- tion Group | Issue Rate | Latency | Execu- tion Pattern | Lock | | | |
|--|-------|--------------|---------------------------|---------------|----------|---------------------------|-------|-------|--------|----|
| | | | | | | | Stage | Start | Cycles | |
| Single- precision floating- point instructions | 182 | FABS | FRn | LS | 1 | 0 | #1 | — | — | — |
| | 183 | FADD | FRm,FRn | FE | 1 | 3/4 | #36 | — | — | — |
| | 184 | FCMP/EQ | FRm,FRn | FE | 1 | 2/4 | #36 | — | — | — |
| | 185 | FCMP/GT | FRm,FRn | FE | 1 | 2/4 | #36 | — | — | — |
| | 186 | FDIV | FRm,FRn | FE | 1 | 12/13 | #37 | F3 | 2 | 10 |
| | | | | | | | | F1 | 11 | 1 |
| | 187 | FLOAT | FPUL,FRn | FE | 1 | 3/4 | #36 | — | — | — |
| | 188 | FMAC | FR0,FRm,FRn | FE | 1 | 3/4 | #36 | — | — | — |
| | 189 | FMUL | FRm,FRn | FE | 1 | 3/4 | #36 | — | — | — |
| | 190 | FNEG | FRn | LS | 1 | 0 | #1 | — | — | — |
| | 191 | FSQRT | FRn | FE | 1 | 11/12 | #37 | F3 | 2 | 9 |
| | | | | | | | | F1 | 10 | 1 |
| | 192 | FSUB | FRm,FRn | FE | 1 | 3/4 | #36 | — | — | — |
| | 193 | FTRC | FRm,FPUL | FE | 1 | 3/4 | #36 | — | — | — |
| | 194 | FMOV | DRm,DRn | LS | 1 | 0 | #1 | — | — | — |
| | 195 | FMOV | @Rm,DRn | LS | 1 | 2 | #2 | — | — | — |
| | 196 | FMOV | @Rm+,DRn | LS | 1 | 1/2 | #2 | — | — | — |
| | 197 | FMOV | @(R0,Rm),DRn | LS | 1 | 2 | #2 | — | — | — |
| | 198 | FMOV | DRm,@Rn | LS | 1 | 1 | #2 | — | — | — |
| | 199 | FMOV | DRm,@-Rn | LS | 1 | 1/1 | #2 | — | — | — |
| 200 | FMOV | DRm,@(R0,Rn) | LS | 1 | 1 | #2 | — | — | — | |
| Double- precision floating- point instructions | 201 | FABS | DRn | LS | 1 | 0 | #1 | — | — | — |
| | 202 | FADD | DRm,DRn | FE | 1 | (7, 8)/9 | #39 | F1 | 2 | 6 |
| | 203 | FCMP/EQ | DRm,DRn | CO | 2 | 3/5 | #40 | F1 | 2 | 2 |
| | 204 | FCMP/GT | DRm,DRn | CO | 2 | 3/5 | #40 | F1 | 2 | 2 |
| | 205 | FCNVDS | DRm,FPUL | FE | 1 | 4/5 | #38 | F1 | 2 | 2 |
| | 206 | FCNVSD | FPUL,DRn | FE | 1 | (3, 4)/5 | #38 | F1 | 2 | 2 |
| | 207 | FDIV | DRm,DRn | FE | 1 | (24, 25)/ 26 | #41 | F3 | 2 | 23 |
| | | | | | | | | F1 | 22 | 3 |
| | | | | | | | | F1 | 2 | 2 |
| 208 | FLOAT | FPUL,DRn | FE | 1 | (3, 4)/5 | #38 | F1 | 2 | 2 | |
| 209 | FMUL | DRm,DRn | FE | 1 | (7, 8)/9 | #39 | F1 | 2 | 6 | |

| Functional Category | No. | Instruction | Instruc- tion Group | Issue Rate | Latency | Execu- tion Pattern | Lock | | | |
|--|-------|-------------|---------------------------|---------------|-------------------|---------------------------|-------|-------|--------|----|
| | | | | | | | Stage | Start | Cycles | |
| Double-precision floating-point instructions | 210 | FNEG | DRn | LS | 1 | 0 | #1 | — | — | — |
| | 211 | FSQRT | DRn | FE | 1 | (23, 24)/ 25 | #41 | F3 | 2 | 22 |
| | | | | | | | | F1 | 21 | 3 |
| | | | | | | | | F1 | 2 | 2 |
| | 212 | FSUB | DRm,DRn | FE | 1 | (7, 8)/9 | #39 | F1 | 2 | 6 |
| 213 | FTRC | DRm,FPUL | FE | 1 | 4/5 | #38 | F1 | 2 | 2 | |
| FPU system control instructions | 214 | LDS | Rm,FPUL | LS | 1 | 1 | #1 | — | — | — |
| | 215 | LDS | Rm,FPSCR | CO | 1 | 4 | #32 | F1 | 3 | 3 |
| | 216 | LDS.L | @Rm+,FPUL | CO | 1 | 1/2 | #2 | — | — | — |
| | | | | | | | | F1 | 3 | 3 |
| | 217 | LDS.L | @Rm+,FPSCR | CO | 1 | 1/4 | #33 | F1 | 3 | 3 |
| | 218 | STS | FPUL,Rn | LS | 1 | 3 | #1 | — | — | — |
| | 219 | STS | FPSCR,Rn | CO | 1 | 3 | #1 | — | — | — |
| | 220 | STS.L | FPUL,@-Rn | CO | 1 | 1/1 | #2 | — | — | — |
| 221 | STS.L | FPSCR,@-Rn | CO | 1 | 1/1 | #2 | — | — | — | |
| Graphics acceleration instructions | 222 | FMOV | DRm,XDn | LS | 1 | 0 | #1 | — | — | — |
| | 223 | FMOV | XDm,DRn | LS | 1 | 0 | #1 | — | — | — |
| | 224 | FMOV | XDm,XDn | LS | 1 | 0 | #1 | — | — | — |
| | 225 | FMOV | @Rm,XDn | LS | 1 | 2 | #2 | — | — | — |
| | 226 | FMOV | @Rm+,XDn | LS | 1 | 1/2 | #2 | — | — | — |
| | 227 | FMOV | @(R0,Rm),XDn | LS | 1 | 2 | #2 | — | — | — |
| | 228 | FMOV | XDm,@Rn | LS | 1 | 1 | #2 | — | — | — |
| | 229 | FMOV | XDm,@-Rm | LS | 1 | 1/1 | #2 | — | — | — |
| | 230 | FMOV | XDm,@(R0,Rn) | LS | 1 | 1 | #2 | — | — | — |
| | 231 | FIPR | FVm,FVn | FE | 1 | 4/5 | #42 | F1 | 3 | 1 |
| | 232 | FRCHG | | FE | 1 | 1/4 | #36 | — | — | — |
| | 233 | FSCHG | | FE | 1 | 1/4 | #36 | — | — | — |
| 234 | FTRV | XMTRX,FVn | FE | 1 | (5, 5, 6, 7)/8 | #43 | F0 | 2 | 4 | |
| | | | | | | | F1 | 3 | 4 | |

Notes: 1. See table 8.1 for the instruction groups.

2. Latency “L1/L2...”: Latency corresponding to a write to each register, including MACH/MACL/FPSCR

Example: MOV.B @Rm+, Rn “1/2”: The latency for Rm is 1 cycle, and the latency for Rn is 2 cycles.

3. Branch latency: Interval until the branch destination instruction is fetched

4. Conditional branch latency “2 (or 1)”: The latency is 2 for a nonzero displacement, and 1 for a zero displacement.
5. Double-precision floating-point instruction latency “(L1, L2)/L3”: L1 is the latency for FR [n+1], L2 that for FR [n], and L3 that for FPSCR.
6. FTRV latency “(L1, L2, L3, L4)/L5”: L1 is the latency for FR [n], L2 that for FR [n+1], L3 that for FR [n+2], L4 that for FR [n+3], and L5 that for FPSCR.
7. Latency “L1/L2/L3/L4” of MAC.L and MAC.W instructions: L1 is the latency for Rm, L2 that for Rn, L3 that for MACH, and L4 that for MACL.
8. Latency “L1/L2” of MUL.L, MULS.W, MULU.W, DMULS.L, and DMULU.L instructions: L1 is the latency for MACH, and L2 that for MACL.
9. Execution pattern: The instruction execution pattern number (see figure 8.2)
10. Lock/stage: Stage locked by the instruction
11. Lock/start: Locking start cycle; 1 is the first D-stage of the instruction.
12. Lock/cycles: Number of cycles locked

Exceptions:

1. When a floating-point computation instruction is followed by an FMOV store, an STS FPUL, Rn instruction, or an STS.L FPUL, @-Rn instruction, the latency of the floating-point computation is decreased by 1 cycle.
2. When the preceding instruction loads the shift amount of the following SHAD/SHLD, the latency of the load is increased by 1 cycle.
3. When an LS group instruction with a latency of less than 3 cycles is followed by a double-precision floating-point instruction, FIPR, or FTRV, the latency of the first instruction is increased to 3 cycles.
Example: In the case of FMOV FR4,FR0 and FIPR FV0,FV4, FIPR is stalled for 2 cycles.
4. When MAC.W/MAC.L/MUL.L/MULS.W/MULU.W/DMULS.L/DMULU.L is followed by an STS.L MACH/MACL, @-Rn instruction, the latency of MAC.W/MAC.L/MUL.L/MULS.W/MULU.W/DMULS.L/DMULU.L is 5 cycles.
5. In the case of consecutive executions of MAC.W/MAC.L/MUL.L/MULS.W/MULU.W/DMULS.L/DMULU.L, the latency is decreased to 2 cycles.
6. When an LDS to MACH/MACL is followed by an STS.L MACH/MACL, @-Rn instruction, the latency of the LDS to MACH/MACL is 4 cycles.
7. When an LDS to MACH/MACL is followed by MAC.W/MAC.L/MUL.L/MULS.W/MULU.W/DMULS.L/DMULU.L, the latency of the LDS to MACH/MACL is 1 cycle.
8. When an FSCHG or FRCHG instruction is followed by an LS group instruction that reads or writes to a floating-point register, the preceding LS group instruction[s] cannot be executed in parallel.
9. When a single-precision FTRC instruction is followed by an STS FPUL, Rn instruction, the latency of the single-precision FTRC instruction is 1 cycle.

8.4 Usage Notes

The following are additional notes on pipeline operation and the method of calculating the number of clock cycles.

The number of states (I clock cycles) required for stages where an external bus access, etc., occurs may include an increased number of cycles, in addition to the number of memory access cycles set by the bus state controller (BSC), etc.

For example, the occurrence of the following may result in idle cycles as observed from the external bus.

1. Transfer of data from the logical address bus to the physical address bus
2. Transfer of data between buses using different operation clocks

The stages where external memory access occurs include some instruction fetch (I) and some memory access (MA) stages.

Section 9 Power-Down Modes

9.1 Overview

In the power-down modes, some of the on-chip peripheral modules and the CPU functions are halted, enabling power consumption to be reduced.

9.1.1 Types of Power-Down Modes

The following power-down modes and functions are provided:

- Sleep mode
- Deep sleep mode
- Standby mode
- Hardware standby mode
- Module standby function (TMU, RTC, SCI/SCIF, DMAC, SQ, and UBC)

Table 9.1 shows the conditions for entering these modes from the program execution state, the status of the CPU and peripheral modules in each mode, and the method of exiting each mode.

Table 9.1 Status of CPU and Peripheral Modules in Power-Down Modes

| Power-Down Mode | Entering Condition | Status | | | | | | |
|------------------|--|-----------|-------------------------|----------------|----------------------------|----------------------|-----------------|---|
| | | CPG | CPU | On-Chip Memory | On-chip Peripheral Modules | Pins | External Memory | Exiting Method |
| Sleep | SLEEP instruction executed while STBY bit is 0 in STBCR | Operating | Halted (registers held) | Held | Operating | Held | Refreshing | <ul style="list-style-type: none"> Interrupt Reset |
| Deep sleep | SLEEP instruction executed while STBY bit is 0 in STBCR, and DSLP bit is 1 in STBCR2 | Operating | Halted (registers held) | Held | Operating (DMA halted) | Held | Self-refreshing | <ul style="list-style-type: none"> Interrupt Reset |
| Standby | SLEEP instruction executed while STBY bit is 1 in STBCR | Halted | Halted (registers held) | Held | Halted* | Held | Self-refreshing | <ul style="list-style-type: none"> Interrupt Reset |
| Hardware standby | Setting CA pin to low level | Halted | Halted | Undefined | Halted* | High-impedance state | Undefined | <ul style="list-style-type: none"> Power-on reset |
| Module standby | Setting MSTP bit to 1 in STBCR | Operating | Operating | Held | Specified modules halted* | Held | Refreshing | <ul style="list-style-type: none"> Clearing MSTP bit to 0 Reset |

Note: * The RTC operates when the START bit in RCR2 is 1 (see section 11, Realtime Clock (RTC)).

9.1.2 Register Configuration

Table 9.2 shows the registers used for power-down mode control.

Table 9.2 Power-Down Mode Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|----------------------------|--------------|-----|---------------|------------|----------------|-------------|
| Standby control register | STBCR | R/W | H'00 | H'FFC00004 | H'1FC00004 | 8 |
| Standby control register 2 | STBCR2 | R/W | H'00 | H'FFC00010 | H'1FC00010 | 8 |
| Clock stop register | CLKSTP00 | R/W | H'00000000 | H'FE0A0000 | H'1E0A0000 | 32 |
| Clock stop clear register | CLKSTPCLR00 | W | H'00000000 | H'FE0A0008 | H'1E0A0008 | 32 |

9.1.3 Pin Configuration

Table 9.3 shows the pins used for power-down mode control.

Table 9.3 Power-Down Mode Pins

| Pin Name | Abbreviation | I/O | Function |
|--------------------------|---------------------------|--------|--|
| Processor status 1 | STATUS1 | Output | Indicate the processor's operating status (STATUS1, STATUS0). HH: Reset HL: Sleep mode LH: Standby mode LL: Normal operation |
| Processor status 0 | STATUS0 | | |
| Sleep request | $\overline{\text{SLEEP}}$ | Input | A transition to sleep mode is effected by inputting a low-level to the pin. |
| Hardware standby request | CA | Input | A transition to hardware standby mode is effected by inputting a low-level to the pin. |

Legend:

H: High level

L: Low level

9.2 Register Descriptions

9.2.1 Standby Control Register (STBCR)

The standby control register (STBCR) is an 8-bit readable/writable register that specifies the power-down mode status. It is initialized to H'00 by a power-on reset via the $\overline{\text{RESET}}$ pin or due to watchdog timer overflow.

| | | | | | | | | |
|----------------|------|-----|-----|-------|-------|-------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | STBY | PHZ | PPU | MSTP4 | MSTP3 | MSTP2 | MSTP1 | MSTP0 |
| 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—Standby (STBY): Specifies a transition to standby mode.

| Bit 7: STBY | Description |
|-------------|--|
| 0 | Transition to sleep mode on execution of SLEEP instruction (Initial value) |
| 1 | Transition to standby mode on execution of SLEEP instruction |

Bit 6—Peripheral Module Pin High Impedance Control (PHZ): Controls the state of peripheral module related pins in standby mode. When the PHZ bit is set to 1, peripheral module related pins go to the high-impedance state in standby mode.

For the relevant pins, see section 9.2.2, Peripheral Module Pin High Impedance Control.

| Bit 6: PHZ | Description |
|------------|--|
| 0 | Peripheral module related pins are in normal state (Initial value) |
| 1 | Peripheral module related pins go to high-impedance state |

Bit 5—Peripheral Module Pin Pull-Up Control (PPU): Controls the state of peripheral module related pins. When the PPU bit is cleared to 0, the pull-up resistor is turned on for peripheral module related pins in the input or high-impedance state.

For the relevant pins, see section 9.2.3, Peripheral Module Pin Pull-Up Control.

| Bit 5: PPU | Description |
|------------|--|
| 0 | Peripheral module related pin pull-up resistors are on (Initial value) |
| 1 | Peripheral module related pin pull-up resistors are off |

Bit 4—Module Stop 4 (MSTP4): Specifies stopping of the clock supply to the DMAC among the on-chip peripheral modules. The clock supply to the DMAC is stopped when the MSTP4 bit is set to 1. When DMA transfer is used, stop the transfer before setting the MSTP4 bit to 1. When DMA transfer is performed after clearing the MSTP4 bit to 0, DMAC settings must be made again.

| Bit 4: MSTP4 | Description |
|--------------|-------------------------------|
| 0 | DMAC operates (Initial value) |
| 1 | DMAC clock supply is stopped |

Bit 3—Module Stop 3 (MSTP3): Specifies stopping of the clock supply to serial communication interface channel 2 (SCIF) among the on-chip peripheral modules. The clock supply to the SCIF is stopped when the MSTP3 bit is set to 1.

| Bit 3: MSTP3 | Description |
|--------------|-------------------------------|
| 0 | SCIF operates (Initial value) |
| 1 | SCIF clock supply is stopped |

Bit 2—Module Stop 2 (MSTP2): Specifies stopping of the clock supply to the timer unit channel 0 to 2 (TMU) among the on-chip peripheral modules. The clock supply to the TMU is stopped when the MSTP2 bit is set to 1.

| Bit 2: MSTP2 | Description |
|--------------|---|
| y | TMU channel 0 to 2 operates (Initial value) |
| 1 | TMU channel 0 to 2 clock supply is stopped |

Bit 1—Module Stop 1 (MSTP1): Specifies stopping of the clock supply to the realtime clock (RTC) among the on-chip peripheral modules. The clock supply to the RTC is stopped when the MSTP1 bit is set to 1. When the clock supply is stopped, RTC registers cannot be accessed but the counters continue to operate.

| Bit 1: MSTP1 | Description |
|--------------|------------------------------|
| 0 | RTC operates (Initial value) |
| 1 | RTC clock supply is stopped |

Bit 0—Module Stop 0 (MSTP0): Specifies stopping of the clock supply to serial communication interface channel 1 (SCI) among the on-chip peripheral modules. The clock supply to the SCI is stopped when the MSTP0 bit is set to 1.

| Bit 0: MSTP0 | Description | |
|--------------|-----------------------------|-----------------|
| 0 | SCI operates | (Initial value) |
| 1 | SCI clock supply is stopped | |

9.2.2 Peripheral Module Pin High Impedance Control

When bit 6 in the standby control register (STBCR) is set to 1, peripheral module related pins go to the high-impedance state in standby mode.

- Relevant Pins

| | | |
|------------------|----------|----------|
| SCI related pins | SCK | MD0/SCK2 |
| | TXD | MD1/TXD2 |
| | MD7/CTS2 | MD8/RTS2 |
| DMA related pins | DACK0 | DRAK0 |
| | DACK1 | DRAK1 |

- Other Information

The setting in this register is invalid when the above pins are used as port output pins.

For details of pin states, see Appendix D, Pin Functions.

9.2.3 Peripheral Module Pin Pull-Up Control

When bit 5 in the standby control register (STBCR) is cleared to 0, peripheral module related pins are pulled up when in the input or high-impedance state.

- Relevant Pins

| | | | |
|------------------|----------|----------|----------|
| SCI related pins | MD0/SCK2 | MD1/TXD2 | MD2/RXD2 |
| | MD7/CTS2 | MD8/RTS2 | SCK |
| | RXD | TXD | |
| DMA related pins | DREQ0 | DACK0 | DRAK0 |
| | DREQ1 | DACK1 | DRAK1 |
| TMU related pin | TCLK | | |

9.2.4 Standby Control Register 2 (STBCR2)

Standby control register 2 (STBCR2) is an 8-bit readable/writable register that specifies the sleep mode and deep sleep mode transition conditions. It is initialized to H'00 by a power-on reset via the RESET pin or due to watchdog timer overflow.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-------|------|---|---|---|---|-------|-------|
| | DSL P | STHZ | — | — | — | — | MSTP6 | MSTP5 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R | R | R | R | R/W | R/W |

Bit 7—Deep Sleep (DSL P): Specifies a transition to deep sleep mode

| Bit 7: DSL P | Description |
|--------------|--|
| 0 | Transition to sleep mode or standby mode on execution of SLEEP instruction, according to setting of STBY bit in STBCR register (Initial value) |
| 1 | Transition to deep sleep mode on execution of SLEEP instruction* |

Note: * When the STBY bit in the STBCR register is 0

Bit 6—STATUS Pin High-Impedance Control (STHZ): This bit selects whether the STATUS0 and 1 pins are set to high-impedance when in hardware standby mode.

| Bit 6: STHZ | Description |
|-------------|--|
| 0 | Sets STATUS0, 1 pins to high-impedance when in hardware standby mode (Initial value) |
| 1 | Drives STATUS0, 1 pins to LH when in hardware standby mode |

Bits 5 to 2—Reserved: Only 0 should only be written to these bits; operation cannot be guaranteed if 1 is written. These bits are always read as 0.

Bit 1—Module Stop 6 (MSTP6): Specifies that the clock supply to the store queue (SQ) in the cache controller (CCN) is stopped. Setting the MSTP6 bit to 1 stops the clock supply to the SQ, and the SQ functions are therefore unavailable. For details regarding the SH7751, see section 4.7, Store Queues.

| Bit 1: MSTP6 | Description |
|--------------|------------------------------|
| 0 | SQ operating (Initial value) |
| 1 | Clock supply to SQ stopped |

Bit 0—Module Stop 5 (MSTP5): Specifies stopping of the clock supply to the user break controller (UBC) among the on-chip peripheral modules. See section 20.6, User Break Controller Stop Function for how to set the clock supply.

| Bit 0: MSTP5 | Description |
|--------------|-------------------------------|
| 0 | UBC operating (Initial value) |
| 1 | Clock supply to UBC stopped |

9.2.5 Clock Stop Register 00 (CLKSTP00)

Clock stop register 00 (CLKSTP00) is a 32-bit readable/writable register that controls the operating clock for peripheral modules.

The clock supply is restarted by writing 1 to the corresponding bit in the CLKSTPCLR00 register. Writing 0 to CLKSTP00 will not change the bit value.

CLKSTP00 is initialized to H'00000000 by a reset. It is not initialized in standby mode.

| | | | | | | | | |
|----------------|----|----|----|-----|----|-------|-------|-------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | CSTP2 | CSTP1 | CSTP0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

Bits 31 to 3—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 2—Clock Stop 2 (CSTP2): Specifies stopping of the peripheral clock supply to the PCI bus controller (PCIC). For details see section 22, PCI Controller (PCIC).

| Bit 2: CSTP2 | Description |
|--------------|--|
| 0 | Peripheral clock is supplied to PCIC (Initial value) |
| 1 | Peripheral clock supply to PCIC is stopped |

Bit 1—Clock Stop 1 (CSTP1): Specifies stopping of the peripheral clock supply to timer unit (TMU) channels 3 and 4.

| Bit 1: CSTP1 | Description |
|--------------|--|
| 0 | Peripheral clock is supplied to TMU channels 3 and 4 (Initial value) |
| 1 | Peripheral clock supply to TMU channels 3 and 4 is stopped |

Bit 0—Clock Stop 0 (CSTP0): Specifies stopping of the peripheral clock supply to the interrupt controller (INTC). When this bit is set, PCIC and TMU channel 3 and 4 interrupts are not detected.

| Bit 0: CSTP0 | Description |
|--------------|--|
| 0 | INTC detects PCIC and TMU channel 3 and 4 interrupts (Initial value) |
| 1 | INTC does not detect PCIC and TMU channel 3 and 4 interrupts |

9.2.6 Clock Stop Clear Register 00 (CLKSTPCLR00)

Clock stop clear register 00 (CLKSTPCLR00) is a 32-bit write-only register that is used to clear corresponding bits in the CLKSTP00 register.

| | | | | | | | | |
|----------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | ... | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| R/W: | W | W | W | ... | W | W | W | W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | W | W | W | W | W | W | W | W |

Bits 31 to 0—Clock Stop Clear: The value of a Clock Stop Clear bit indicates whether the corresponding Clock Stop bit is to be cleared. See section 9.2.5, Clock Stop Register 00 (CLKSTP00), for the correspondence between bits and the clocks stopped.

| Bits 31 to 0 | Description |
|--------------|---|
| 0 | Corresponding Clock Stop bit is not changed (Initial value) |
| 1 | Corresponding Clock Stop bit is cleared |

9.3 Sleep Mode

9.3.1 Transition to Sleep Mode

If a SLEEP instruction is executed when the STBY bit in STBCR is cleared to 0, the chip switches from the program execution state to sleep mode. After execution of the SLEEP instruction, the CPU halts but its register contents are retained. The on-chip peripheral modules continue to operate, and the clock continues to be output from the CKIO pin.

In sleep mode, a high-level signal is output at the STATUS1 pin, and a low-level signal at the STATUS0 pin.

9.3.2 Exit from Sleep Mode

Sleep mode is exited by means of an interrupt (NMI, IRL, or on-chip peripheral module) or a reset. In sleep mode, interrupts are accepted even if the BL bit in the SR register is 1. If necessary, SPC and SSR should be saved to the stack before executing the SLEEP instruction.

Exit by Interrupt: When an NMI, IRL, or on-chip peripheral module interrupt is generated, sleep mode is exited and interrupt exception handling is executed. The code corresponding to the interrupt source is set in the INTEVT register.

Exit by Reset: Sleep mode is exited by means of a power-on or manual reset via the $\overline{\text{RESET}}$ pin, or a power-on or manual reset executed when the watchdog timer overflows.

9.4 Deep Sleep Mode

9.4.1 Transition to Deep Sleep Mode

If a SLEEP instruction is executed when the STBY bit in STBCR is cleared to 0 and the DSLP bit in STBCR2 is set to 1, the chip switches from the program execution state to deep sleep mode. After execution of the SLEEP instruction, the CPU halts but its register contents are retained. Except for the DMAC*, on-chip peripheral modules continue to operate. The clock continues to be output to the CKIO pin, but all bus access (including auto refresh) stops. When using memory that requires refreshing, set the self-refresh function prior to making the transition to deep sleep mode.

In deep sleep mode, a high-level signal is output at the STATUS1 pin, and a low-level signal at the STATUS0 pin.

Note: * Terminate DMA transfers prior to making the transition to deep sleep mode. If you make a transition to deep sleep mode while DMA transfers are in progress, the results of those transfers cannot be guaranteed.

9.4.2 Exit from Deep Sleep Mode

As with sleep mode, deep sleep mode is exited by means of an interrupt (NMI, IRL, or on-chip peripheral module) or a reset.

9.5 Pin Sleep Mode

9.5.1 Transition to Pin Sleep Mode

Changing the $\overline{\text{SLEEP}}$ pin to the low level causes this LSI to make a transition to sleep mode.

To ensure that memory is correctly refreshed, use this function when the DSLP bit of STBCR2 is set to 0.

9.5.2 Exit from Pin Sleep Mode

Setting the $\overline{\text{SLEEP}}$ pin level high causes this LSI to return to the normal state. The pin sleep mode is also canceled when the conditions specified in section 9.3.2, “Exit From Sleep Mode” are satisfied.

In a power-on reset, the $\overline{\text{SLEEP}}$ pin should be fixed high.

9.6 Standby Mode

9.6.1 Transition to Standby Mode

If a SLEEP instruction is executed when the STBY bit in STBCR is set to 1, the chip switches from the program execution state to standby mode. In standby mode, the on-chip peripheral modules halt as well as the CPU. Clock output from the CKIO pin is also stopped.

The CPU and cache register contents are retained. Some on-chip peripheral module registers are initialized. The state of the peripheral module registers in standby mode is shown in table 9.4.

Table 9.4 State of Registers in Standby Mode

| Module | Initialized Registers | Registers That Retain Their Contents |
|---------------------------------|------------------------------|--------------------------------------|
| Interrupt controller | — | All registers |
| User break controller | — | All registers |
| Bus state controller | — | All registers |
| On-chip oscillation circuits | — | All registers |
| Timer unit | TSTR register* | All registers except TSTR |
| Realtime clock | — | All registers |
| Direct memory access controller | — | All registers |
| Serial communication interface | See Appendix A, Address List | See Appendix A, Address List |

Notes: DMA transfer should be terminated before making a transition to standby mode. Transfer results are not guaranteed if standby mode is entered during transfer.

* Not initialized when the realtime clock (RTC) is in use (see section 12, Timer Unit (TMU)).

The procedure for a transition to standby mode is shown below.

1. Clear the TME bit in the WDT timer control register (WTCSR) to 0, and stop the WDT. Set the initial value for the up-count in the WDT timer counter (WTCNT), and set the clock to be used for the up-count in bits CKS2–CKS0 in the WTCSR register.
2. Set the STBY bit in the STBCR register to 1, then execute a SLEEP instruction.
3. When standby mode is entered and the chip's internal clock stops, a low-level signal is output at the STATUS1 pin, and a high-level signal at the STATUS0 pin.

9.6.2 Exit from Standby Mode

Standby mode is exited by means of an interrupt (NMI, IRL, or on-chip peripheral module) or a reset via the RESET and MRESET pins.

Exit by Interrupt: A hot start can be performed by means of the on-chip WDT. When an NMI, IRL^{*1}, RTC, or GPIO^{*2} interrupt is detected, the WDT starts counting. After the count overflows, clocks are supplied to the entire chip, standby mode is exited, and the STATUS1 and STATUS0 pins both go low. Interrupt exception handling is then executed, and the code corresponding to the interrupt source is set in the INTEVT register. In standby mode, interrupts are accepted even if the BL bit in the SR register is 1, and so, if necessary, SPC and SSR should be saved to the stack before executing the SLEEP instruction.

The phase of the CKIO pin clock output may be unstable immediately after an interrupt is detected, until standby mode is exited.

- Notes:
1. Only when the RTC clock (32.768 kHz) is operating (see section 19.2.2, IRL Interrupts), standby mode can be exited by means of IRL3–IRL0 (when the IRL3–IRL0 level is higher than the SR register IMASK mask level).
 2. GPIC can be used to cancel standby mode when the RTC clock (32.768 kHz) is operating (when the GPIC level is higher than the SR register IMASK mask level).

Exit by Reset: Standby mode is exited by means of a reset (power-on or manual) via the $\overline{\text{RESET}}$ pin. The $\overline{\text{RESET}}$ pin should be held low until clock oscillation stabilizes. The internal clock continues to be output at the CKIO pin.

9.6.3 Clock Pause Function

In standby mode, it is possible to stop or change the frequency of the clock input from the EXTAL pin. This function is used as follows.

1. Enter standby mode following the transition procedure described above.
2. When standby mode is entered and the chip's internal clock stops, a low-level signal is output at the STATUS1 pin, and a high-level signal at the STATUS0 pin.
3. The input clock is stopped, or its frequency changed, after the STATUS1 pin goes low and the STATUS0 pin high.
4. When the frequency is changed, input an NMI or IRL interrupt after the change. When the clock is stopped, input an NMI or IRL interrupt after applying the clock.
5. After the time set in the WDT, clock supply begins inside the chip, the STATUS1 and STATUS0 pins both go low, and operation is resumed from interrupt exception handling.

9.7 Module Standby Function

9.7.1 Transition to Module Standby Function

Setting the MSTP6–MSTP0 and CSTP2–CSTP0 bits in the standby control register, standby control register 2, and clock stop clear register 00 to 1 enables the clock supply to the corresponding on-chip peripheral modules to be halted. Use of this function allows power consumption in sleep mode to be further reduced.

In the module standby state, the on-chip peripheral module external pins retain their states prior to halting of the modules, and most registers retain their states prior to halting of the modules.

| Bit | | Description |
|-------|---|---|
| CSTP2 | 0 | Peripheral clock is supplied to PCIC |
| | 1 | Peripheral clock supply to PCIC is stopped |
| CSTP1 | 0 | Peripheral clock is supplied to TMU channels 3 and 4 |
| | 1 | Peripheral clock supply to TMU channels 3 and 4 is stopped |
| CSTP0 | 0 | INTC detects PCIC and TMU channel 3 and 4 interrupts |
| | 1 | INTC does not detect PCIC and TMU channel 3 and 4 interrupts |
| MSTP6 | 0 | SQ operates |
| | 1 | Clock supplied to SQ is stopped |
| MSTP5 | 0 | UBC operates |
| | 1 | Clock supplied to UBC is stopped* ³ |
| MSTP4 | 0 | DMAC operates |
| | 1 | Clock supplied to DMAC is stopped* ⁴ |
| MSTP3 | 0 | SCIF operates |
| | 1 | Clock supplied to SCIF is stopped |
| MSTP2 | 0 | TMU operates |
| | 1 | Clock supplied to TMU is stopped, and register is initialized* ¹ |
| MSTP1 | 0 | RTC operates |
| | 1 | Clock supplied to RTC is stopped* ² |
| MSTP0 | 0 | SCI operates |
| | 1 | Clock supplied to SCI is stopped |

- Notes:
1. The register initialized is the same as in standby mode, but initialization is not performed if the RTC clock is not in use (see section 12, Timer Unit (TMU)).
 2. The counter operates when the START bit in RCR2 is 1 (see section 11, Realtime Clock (RTC)).
 3. For details, see section 20.6, User Break Controller Stop Function.
 4. Terminate DMA transfers prior to making the transition to module standby mode. If you make a transition to module standby mode while DMA transfers are in progress, the results of those transfers cannot be guaranteed.

9.7.2 Exit from Module Standby Function

In the case of the standby control register and standby control register 2, the module standby function is exited by writing 0 to the MSTP6–MSTP0 bits. In the case of clock stop register 00, the module standby function is exited by writing 1 to the corresponding bit in clock stop clear register 00.

The module standby function is not exited by means of a power-on reset via the $\overline{\text{RESET}}$ pin or a power-on reset caused by watchdog timer overflow.

9.8 Hardware Standby Mode

9.8.1 Transition to Hardware Standby Mode

Setting the CA pin level low effects a transition to hardware standby mode. In this mode, all modules other than the RTC stop, as in the standby mode selected using the SLEEP command.

Hardware standby mode differs from standby mode as follows:

1. Interrupts and manual resets are not available;
2. All output pins other than the STATUS pin are in the high-impedance state and the pull-up resistance is off.
3. On the SH7751, the RTC continues to operate even when no power is supplied to power pins other than the RTC power supply pin.

The status of the STATUS pin is determined by the STHZ bit of STBCR2. See section D, Pin Functions, for details of output pin states.

Operation when a low-level is input to the CA pin when in the standby mode depends on the CPG status, as follows:

1. In standby mode
The clock remains stopped and a transition is made to the hardware standby state. Interrupts and manual resets are disabled, but the output pins remain in the same state as in standby mode.
2. When WDT is operating when standby mode is exited by interrupt
Standby mode is momentarily exited, the CPU restarts, and then a transition is made to hardware standby mode.

Note that the level of the CA pin must be kept low while in hardware standby mode.

9.8.2 Exit from Hardware Standby Mode

Hardware standby mode can only be cancelled by a power-on reset. Driving the CA pin high when the $\overline{\text{RESET}}$ pin is being driven low causes clock oscillation to start. At this point, maintain the $\overline{\text{RESET}}$ pin at low level until clock oscillation stabilizes. The CPU will start power-on reset processing if the $\overline{\text{RESET}}$ pin is driven high. Hardware standby mode cannot be cancelled by an interrupt or a manual reset.

9.8.3 Usage Notes

1. The CA pin level must be kept high when the RTC power supply is started (figure 9.15).
2. On the SH7751R, supply power to the V_{DD} , V_{DDQ} , V_{DD-CPG} , $V_{DD-PLL1}$, and $V_{DD-PLL2}$ power supply pins in addition to the RTC power supply pin in hardware standby mode.

9.9 STATUS Pin Change Timing

The STATUS1 and STATUS0 pin change timing is shown below.

The meaning of the STATUS pin settings is as follows:

Reset: HH (STATUS1 high, STATUS0 high)

Sleep: HL (STATUS1 high, STATUS0 low)

Standby: LH (STATUS1 low, STATUS0 high)

Normal: LL (STATUS1 low, STATUS0 low)

The meaning of the clock units is as follows:

Bcyc: Bus clock cycle

Pcyc: Peripheral clock cycle

9.9.1 In Reset

Power-On Reset

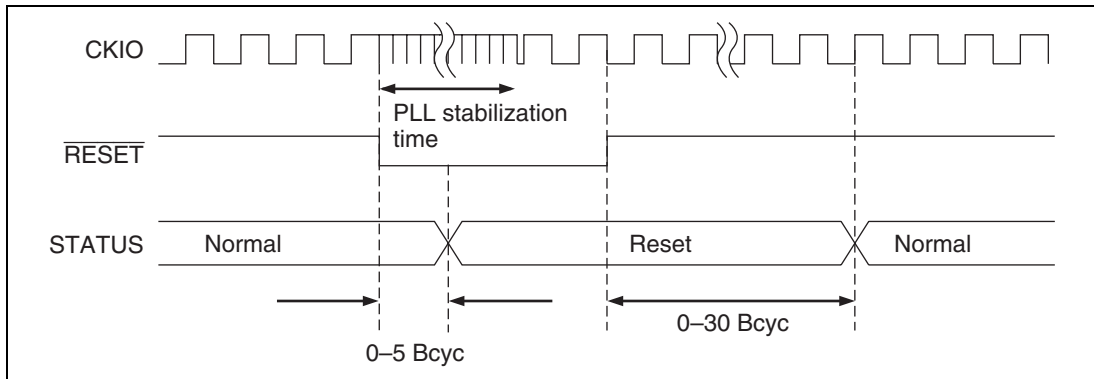
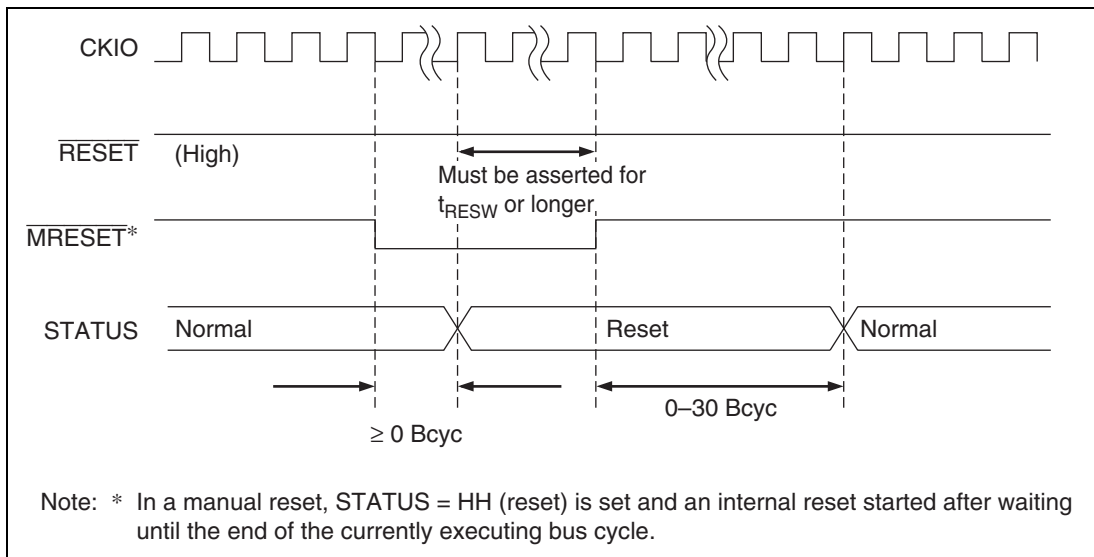


Figure 9.1 STATUS Output in Power-On Reset

Manual Reset



Note: * In a manual reset, STATUS = HH (reset) is set and an internal reset started after waiting until the end of the currently executing bus cycle.

Figure 9.2 STATUS Output in Manual Reset

9.9.2 In Exit from Standby Mode

Standby → Interrupt

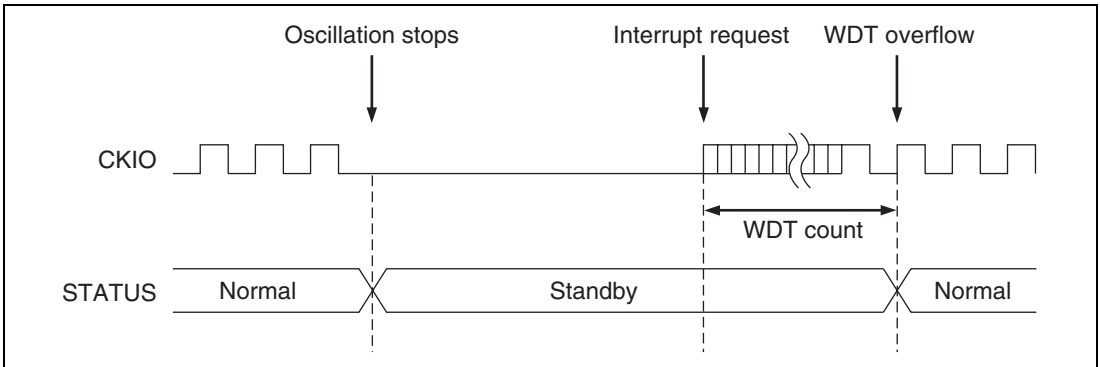


Figure 9.3 STATUS Output in Standby → Interrupt Sequence

Standby → Power-On Reset

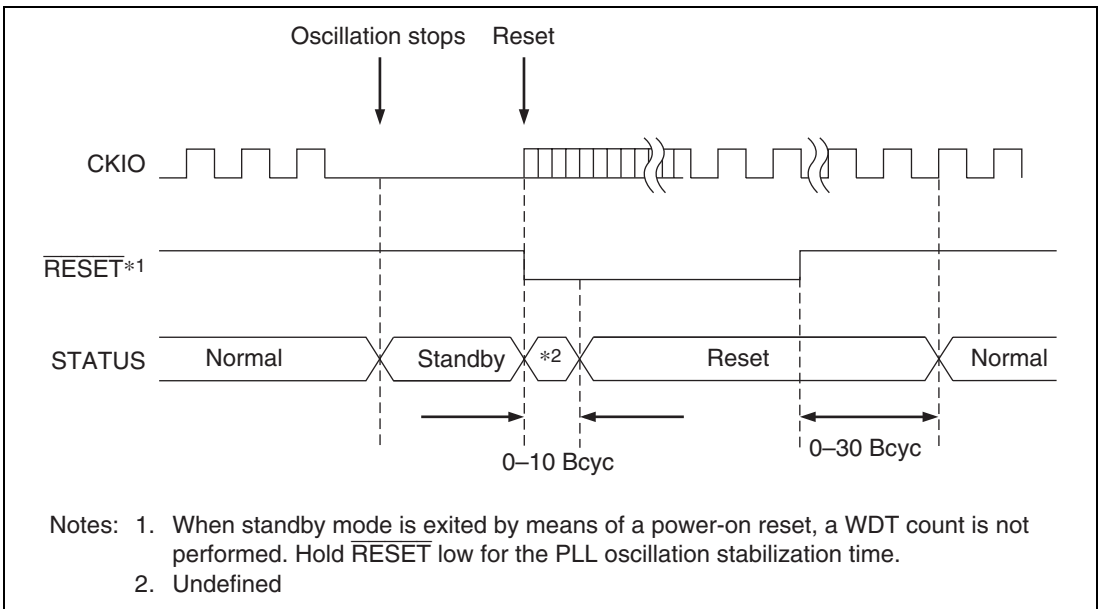
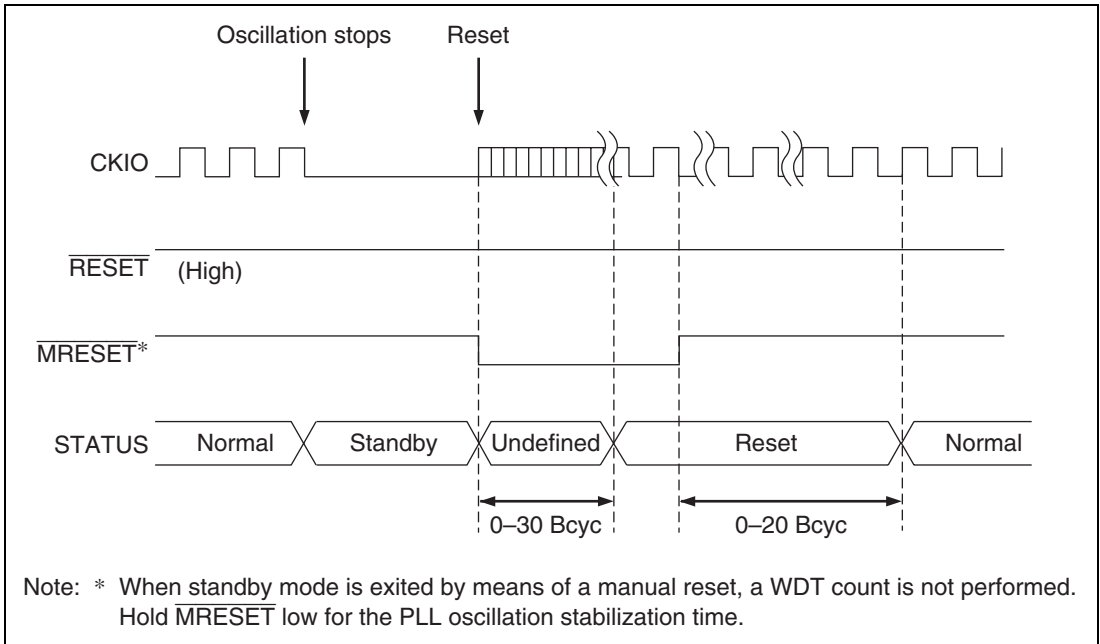
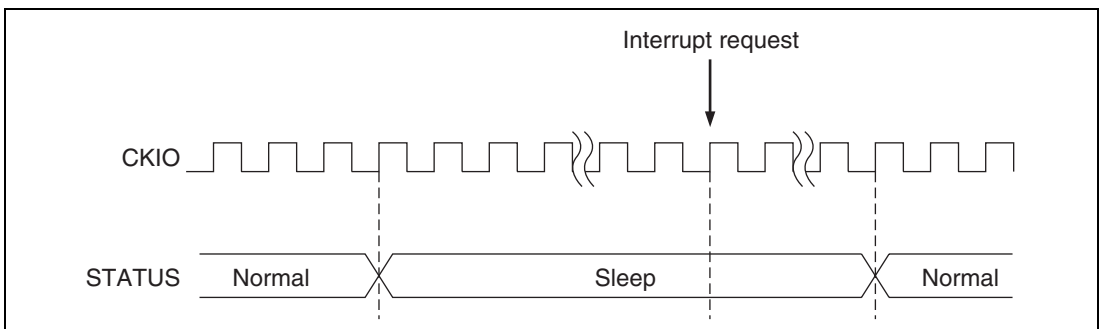


Figure 9.4 STATUS Output in Standby → Power-On Reset Sequence

Standby → Manual Reset**Figure 9.5 STATUS Output in Standby → Manual Reset Sequence****9.9.3 In Exit from Sleep Mode****Sleep → Interrupt****Figure 9.6 STATUS Output in Sleep → Interrupt Sequence**

Sleep → Power-On Reset

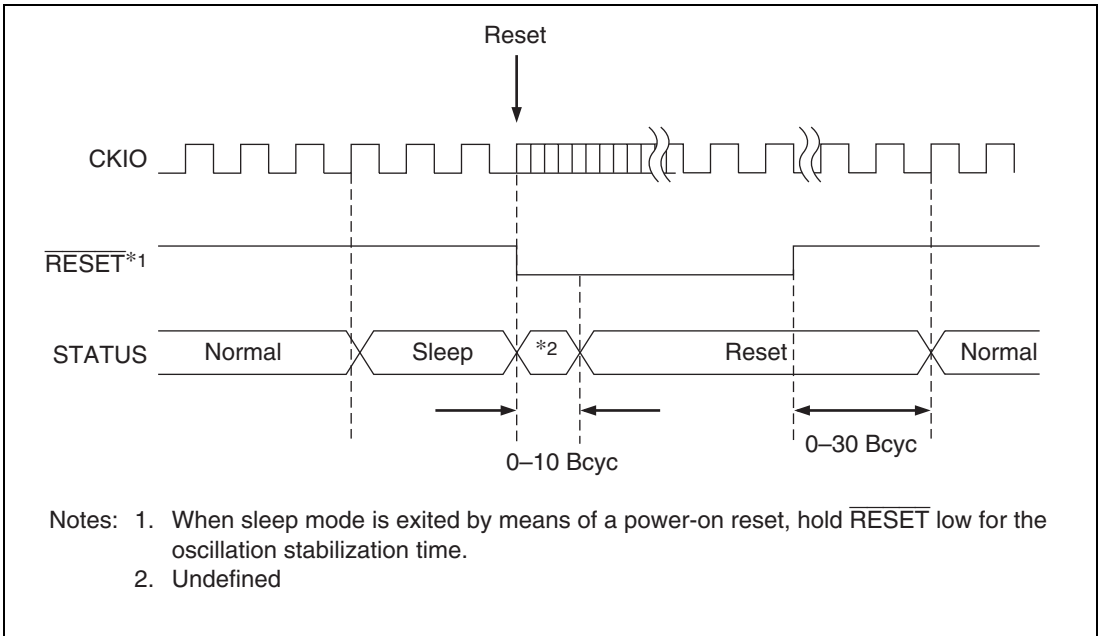
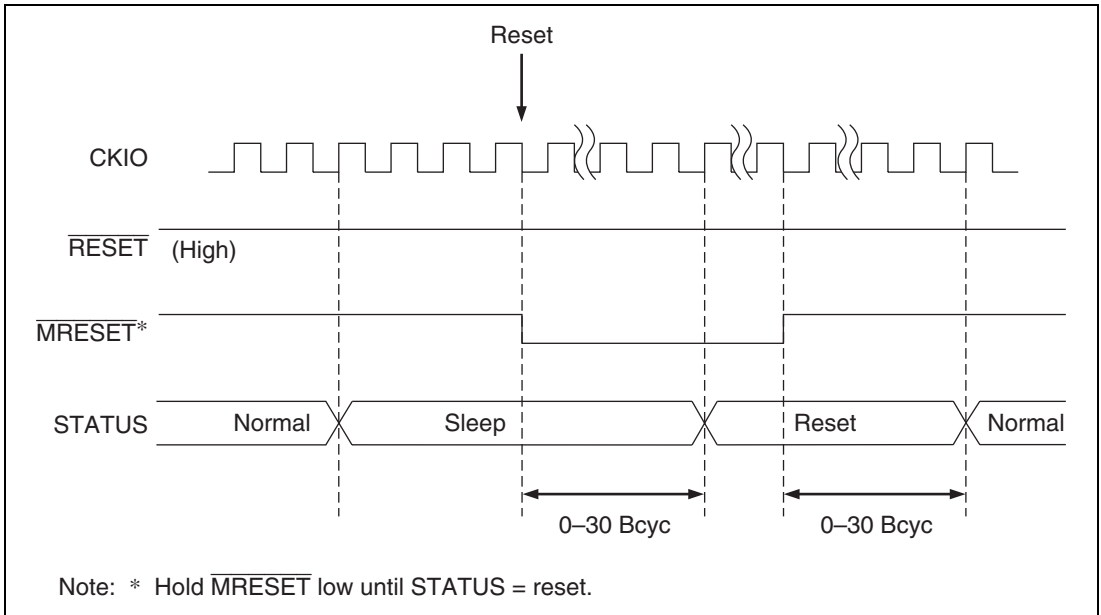


Figure 9.7 STATUS Output in Sleep → Power-On Reset Sequence

Sleep → Manual Reset**Figure 9.8 STATUS Output in Sleep → Manual Reset Sequence**

9.9.4 In Exit from Deep Sleep Mode

Deep Sleep → Interrupt

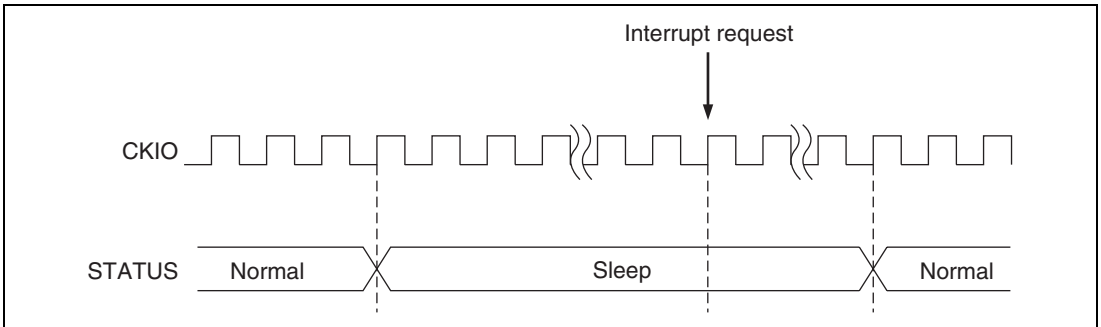
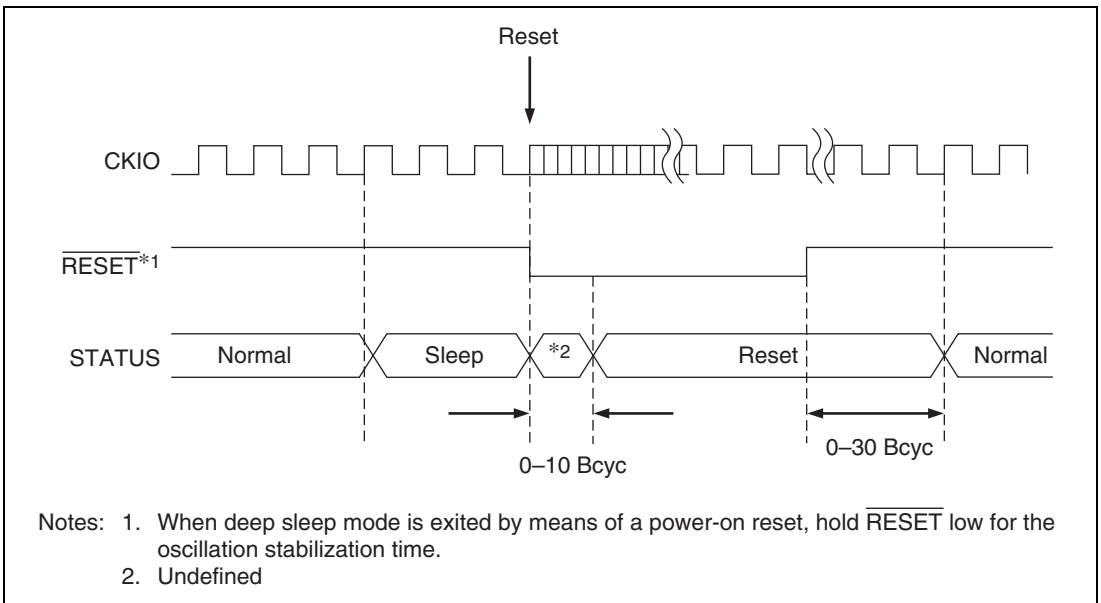


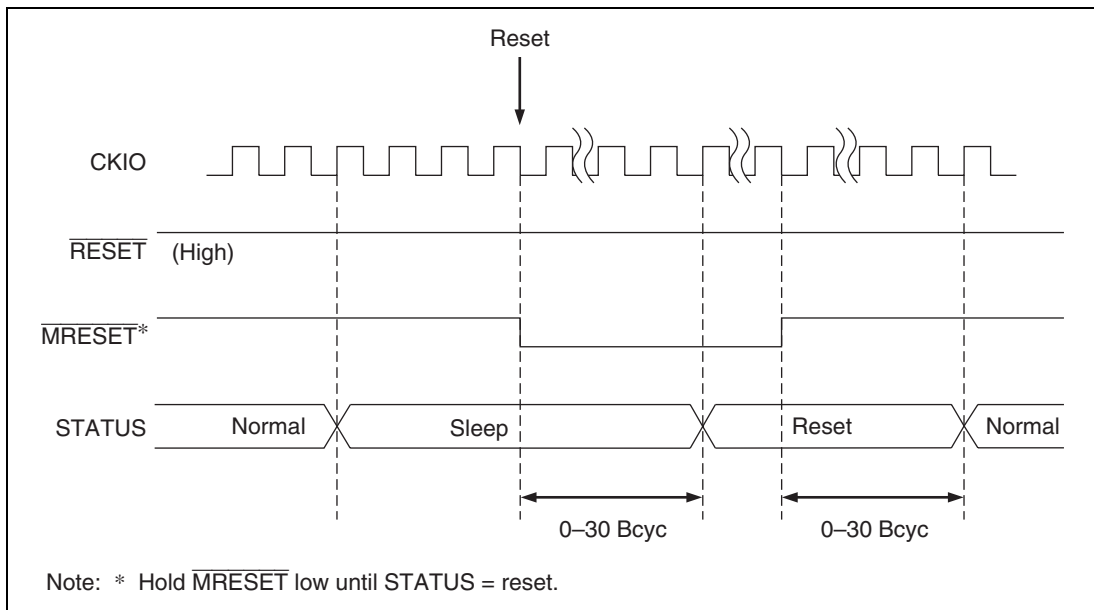
Figure 9.9 STATUS Output in Deep Sleep → Interrupt Sequence

Deep Sleep → Power-On Reset



- Notes: 1. When deep sleep mode is exited by means of a power-on reset, hold $\overline{\text{RESET}}$ low for the oscillation stabilization time.
 2. Undefined

Figure 9.10 STATUS Output in Deep Sleep → Power-On Reset Sequence

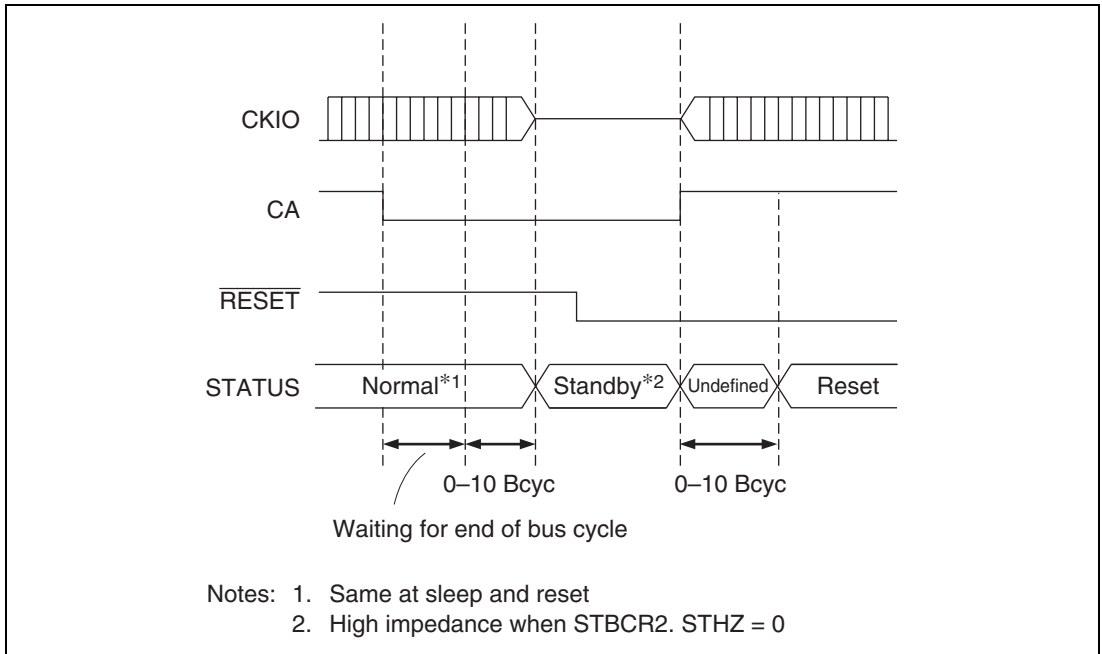
Deep Sleep → Manual Reset**Figure 9.11 STATUS Output in Deep Sleep → Manual Reset Sequence**

9.9.5 Hardware Standby Mode Timing

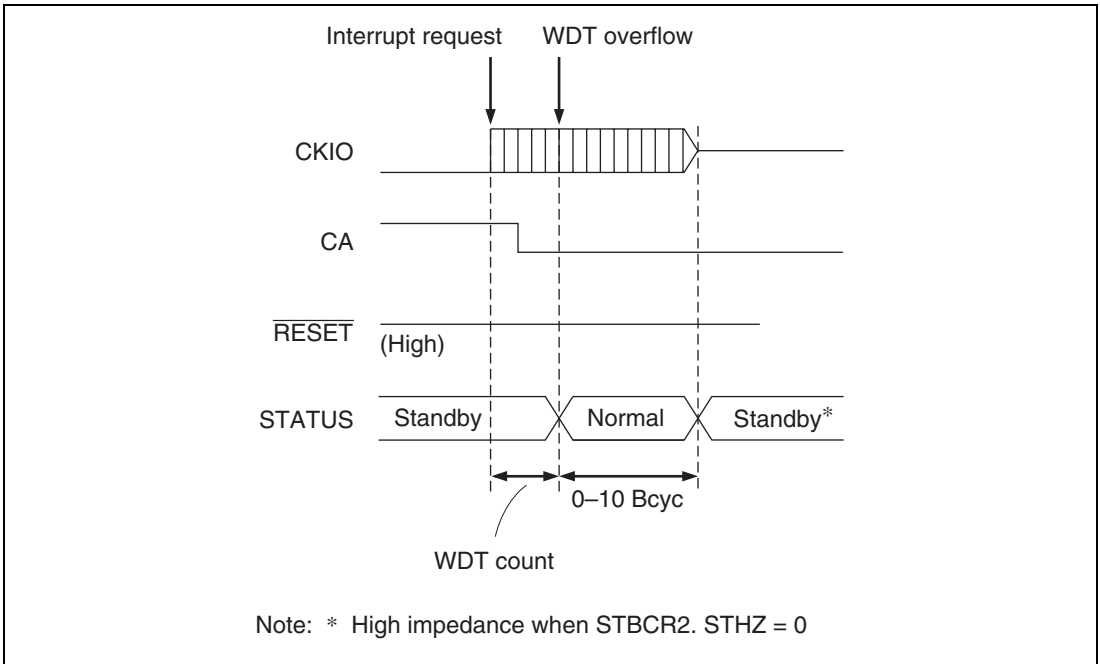
Figure 9.12 shows the timing of the signals of the respective pins in hardware standby mode.

The CA pin level must be kept low while in hardware standby mode.

After setting the $\overline{\text{RESET}}$ pin level low, the clock starts when the CA pin level is switched to high.



**Figure 9.12 Hardware Standby Mode Timing
(When CA = Low in Normal Operation)**



**Figure 9.13 Hardware Standby Mode Timing
(When CA = Low in WDT Operation)**

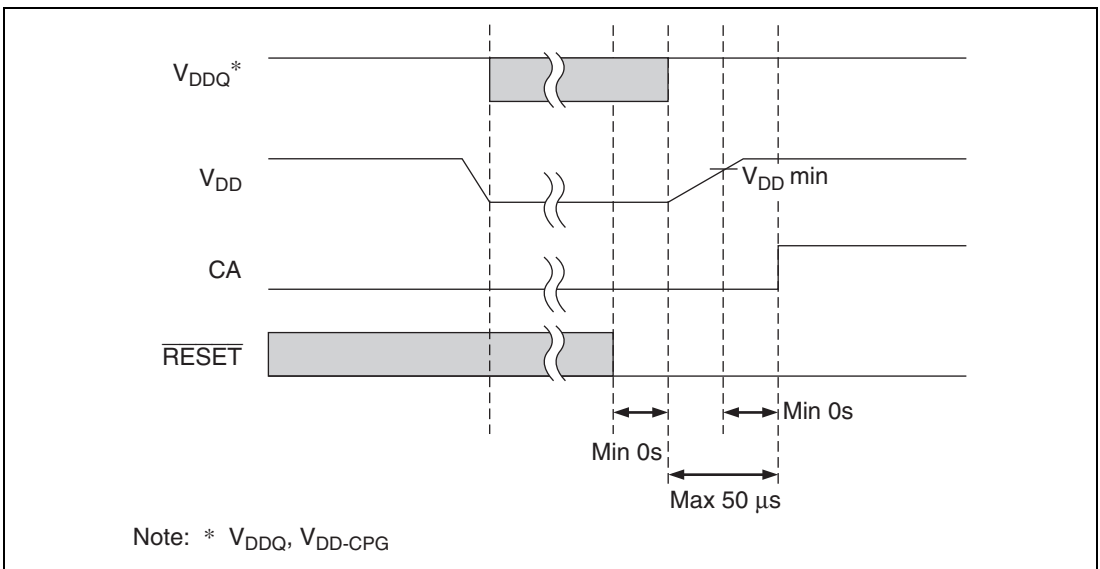


Figure 9.14 Timing When Power Other than VDD-RTC Is Off

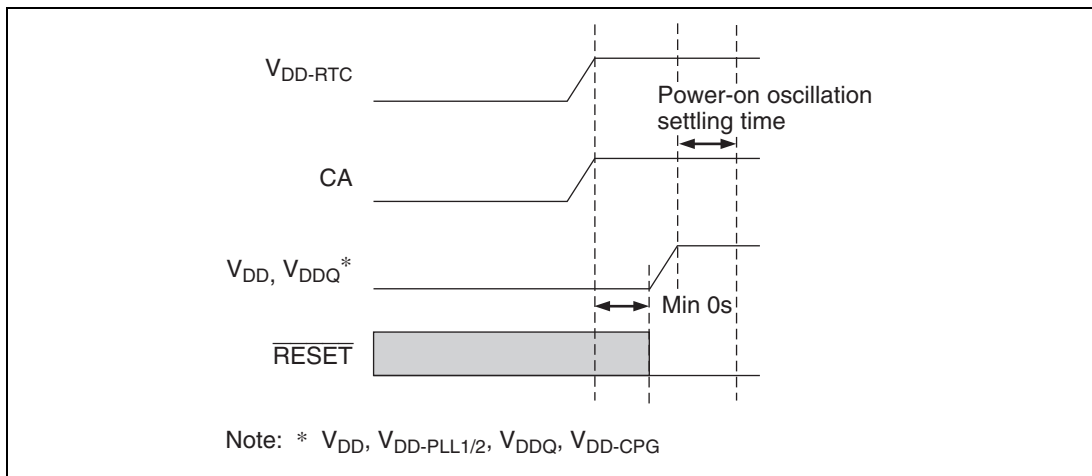


Figure 9.15 Timing When VDD-RTC Power Is Off → On

9.10 Usage Notes

9.10.1 Note on Current Consumption

After a power-on reset, the current consumption may exceed the maximum value for sleep mode or standby mode during the period until one or more of the arithmetic operation or floating-point operation instructions listed below is executed.

1. Arithmetic operation instructions
MAC.W, MAC.L
2. Floating-point operation instructions
 - When FPSCR.PR = 0
FADD, FSUB, FMUL, FMAC, FLOAT, FTRC, FDIV, FSQRT, FIPR, FTRV
 - When FPSCR.PR = 1
FADD, FSUB, FMUL, FLOAT, FTRC, FDIV, FSQRT, FCNVSD, FCNVDS

Workaround: After a power-on reset, execute one or more of the above instructions before transitioning to sleep mode or standby mode.

Example: To reduce the effect on FPSCR, arrange the following two instructions starting at H'A0000000.

| Address | Instruction String |
|------------|---|
| H'A0000000 | FLDI1 FR0 |
| H'A0000002 | FADD FR0, FR0 ; FLDI1 FR0 loads 1 into FR0, |
| : | : ; so the cause and flag bits of FPSCR are not set to 1. |

Section 10 Clock Oscillation Circuits

10.1 Overview

The on-chip oscillation circuits comprise a clock pulse generator (CPG) and a watchdog timer (WDT).

The CPG generates the clocks supplied inside the processor and performs power-down mode control.

The WDT is a single-channel timer used to count the clock stabilization time when exiting standby mode or the frequency is changed. It can be used as a normal watchdog timer or an interval timer.

10.1.1 Features

The CPG has the following features:

- Three clocks

The CPG can generate the CPU clock (Ick) used by the CPU, FPU, caches, and TLB, the peripheral module clock (Pck) used by the peripheral modules, and the bus clock (Bck) used by the external bus interface.

- Six clock modes

Any of six clock operating modes can be selected, with different combinations of CPU clock, bus clock, and peripheral module clock division ratios after a power-on reset.

- Frequency change function

PLL (phase-locked loop) circuits and a frequency divider in the CPG enable the CPU clock, bus clock, and peripheral module clock frequencies to be changed. Frequency changes are performed by software in accordance with the settings in the frequency control register (FRQCR).

- PLL on/off control

Power consumption can be reduced by stopping the PLL circuits during low-frequency operation.

- Power-down mode control

It is possible to stop the clock in sleep mode and standby mode, and to stop specific modules with the module standby function.

The WDT has the following features

- Can be used to secure clock stabilization time
Used when exiting standby mode or a temporary standby state when the clock frequency is changed.
- Can be switched between watchdog timer mode and interval timer mode
- Internal reset generation in watchdog timer mode
An internal reset is executed on counter overflow.
Power-on reset or manual reset can be selected.
- Interrupt generation in interval timer mode
An interval timer interrupt is generated on counter overflow.
- Selection of eight counter input clocks
Any of eight clocks can be selected, scaled from the $\times 1$ clock of frequency divider 2 shown in figure 10.1.

The CPG is described in sections 10.2 to 10.6, and the WDT in sections 10.7 to 10.9.

10.2 Overview of CPG

10.2.1 Block Diagram of CPG

Figures 10.1(1) and 10.1(2) show a block diagram of the CPG in the SH7751 and SH7751R.

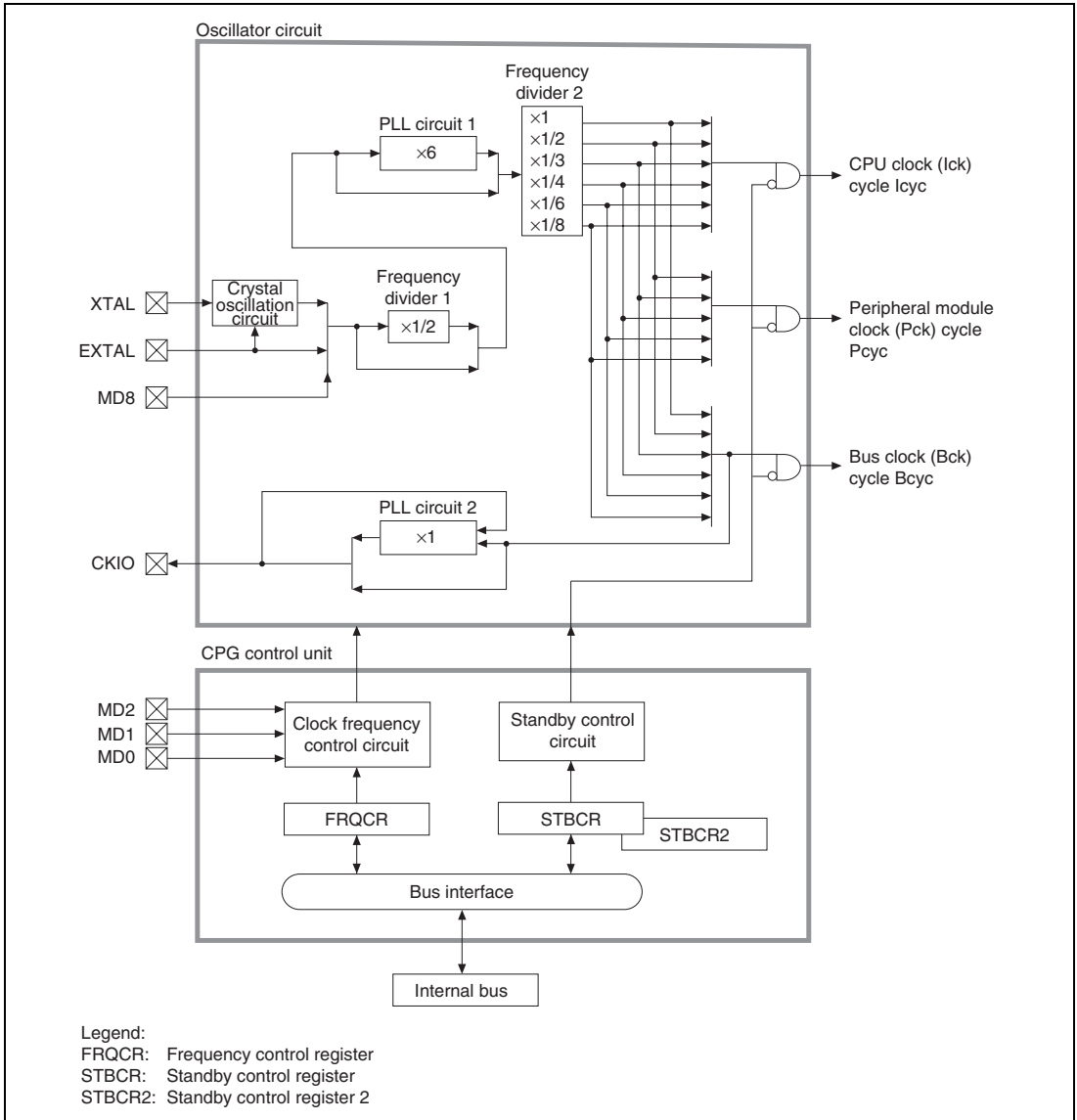


Figure 10.1 (1) Block Diagram of CPG (SH7751)

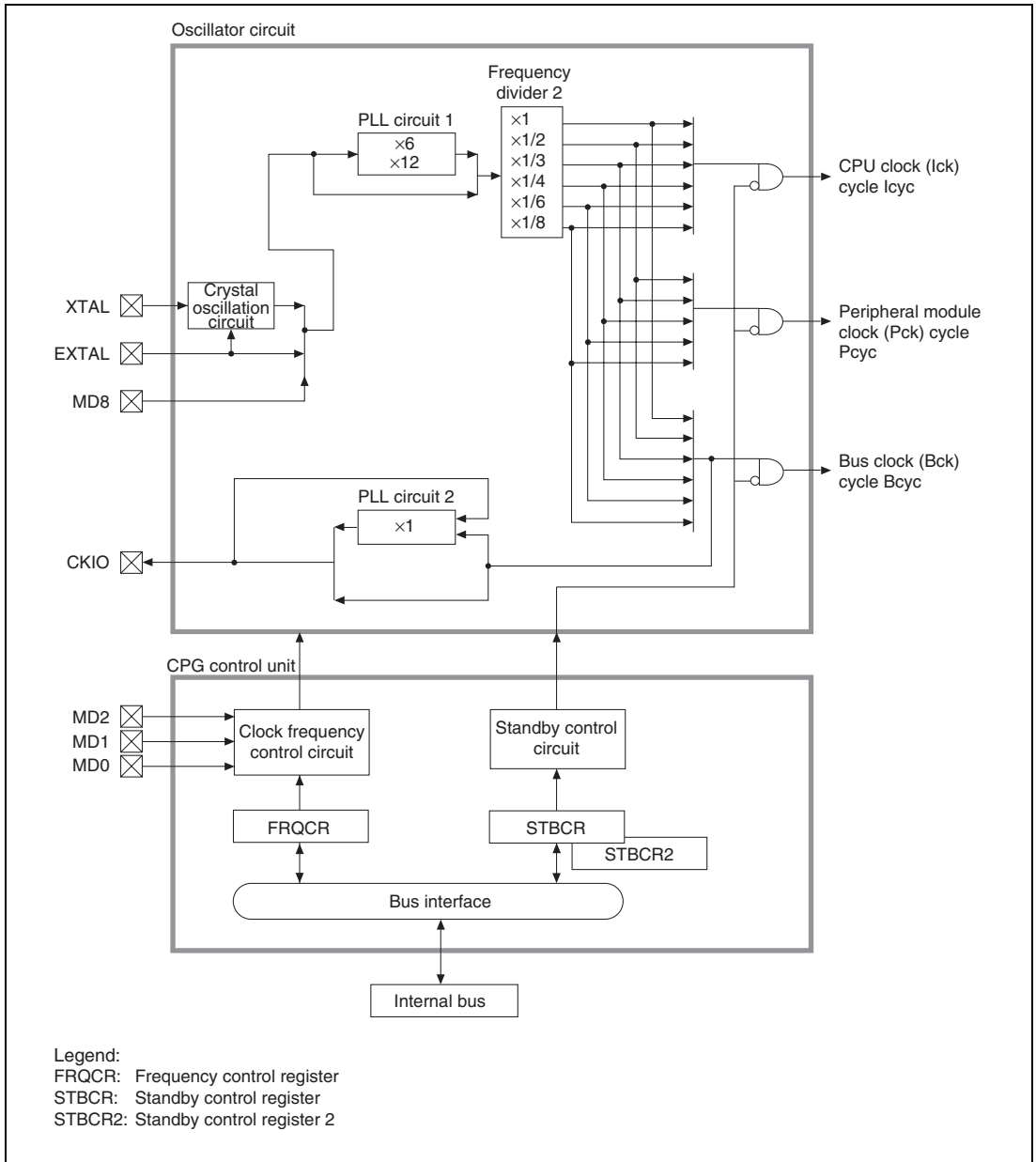


Figure 10.1 (2) Block Diagram of CPG (SH7751R)

The function of each of the CPG blocks is described below.

PLL Circuit 1: PLL circuit 1 has a function for multiplying the clock frequency from the EXTAL pin or crystal oscillation circuit by 6 (SH7751 and SH7751R) or 12 (SH7751R). Starting and stopping is controlled by a frequency control register setting. Control is performed so that the internal clock rising edge phase matches the input clock rising edge phase.

PLL Circuit 2: PLL circuit 2, according to the output clock feedback from the CKIO pin, coordinates the phases of the bus clock and the CKIO pin output clock. Starting and stopping is controlled by a frequency control register setting.

Crystal Oscillation Circuit: This is the oscillator circuit used when a crystal resonator is connected to the XTAL and EXTAL pins. Use of the crystal oscillation circuit can be selected with the MD8 pin.

Frequency Divider 1 (SH7751R only): Frequency divider 1 has a function for adjusting the clock waveform duty to 50% by halving the input clock frequency when clock input from the EXTAL pin is supplied internally without using PLL circuit 1.

Frequency Divider 2: Frequency divider 2 generates the CPU clock (Ick), bus clock (Bck), and peripheral module clock (Pck). The division ratio is set in the frequency control register.

Clock Frequency Control Circuit: The clock frequency control circuit controls the clock frequency by means of the MD pins and frequency control register.

Standby Control Circuit: The standby control circuit controls the state of the on-chip oscillation circuits and other modules when the clock is switched and in sleep and standby modes.

Frequency Control Register (FRQCR): The frequency control register contains control bits for clock output from the CKIO pin, PLL circuit 1 and 2 on/off control, and the CPU clock, bus clock, and peripheral module clock frequency division ratios.

Standby Control Register (STBCR): The standby control register contains power save mode control bits. For further information on the standby control register, see section 9, Power-Down Modes.

Standby Control Register 2 (STBCR2): Standby control register 2 contains a power save mode control bit. For further information on standby control register 2, see section 9, Power-Down Modes.

10.2.2 CPG Pin Configuration

Table 10.1 shows the CPG pins and their functions.

Table 10.1 CPG Pins

| Pin Name | Abbreviation | I/O | Function |
|--|--------------|--------|---|
| Mode control pins | MD0 | Input | Set clock operating mode |
| | MD1 | | |
| | MD2 | | |
| Crystal I/O pins (clock input pins) | XTAL | Output | Connects crystal resonator |
| | EXTAL | Input | Connects crystal resonator, or used as external clock input pin |
| | MD8 | Input | Selects use/non-use of crystal resonator When MD8 = 0, external clock is input from EXTAL When MD8 = 1, crystal resonator is connected directly to EXTAL and XTAL |
| Clock output pin | CKIO | Output | Used as external clock output pin Level can also be fixed |
| CKIO enable pin | CKE | Output | 0 when CKIO output clock is unstable and in case of synchronous DRAM self-refreshing* |

Note: * Set to 1 in a power-on reset.

For details of synchronous DRAM self-refreshing, see section 13.3.5, Synchronous DRAM Interface.

10.2.3 CPG Register Configuration

Table 10.2 shows the CPG register configuration.

Table 10.2 CPG Register

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|----------------------------|--------------|-----|---------------|------------|----------------|-------------|
| Frequency control register | FRQCR | R/W | Undefined | H'FFC00000 | H'1FC00000 | 16 |

10.3 Clock Operating Modes

Tables 10.3 (1) and 10.3 (2) show the clock operating modes corresponding to various combinations of mode control pin (MD2–MD0) settings (initial settings such as the frequency division ratio).

Table 10.4 shows FRQCR settings and internal clock frequencies.

Table 10.3 (1) Clock Operating Modes (SH7751)

| Clock Operating Mode | External Pin Combination | | | Frequency (vs. Input Clock) | | | | | | |
|----------------------|--------------------------|-----|-----|-----------------------------|------|------|-----------|-----------|-------------------------|---------------------|
| | MD2 | MD1 | MD0 | 1/2 Frequency Divider | PLL1 | PLL2 | CPU Clock | Bus Clock | Peripheral Module Clock | FRQCR Initial Value |
| 0 | 0 | 0 | 0 | Off | On | On | 6 | 3/2 | 3/2 | H'0E1A |
| 1 | | | 1 | Off | On | On | 6 | 1 | 1 | H'0E23 |
| 2 | | 1 | 0 | On | On | On | 3 | 1 | 1/2 | H'0E13 |
| 3 | | | 1 | Off | On | On | 6 | 2 | 1 | H'0E13 |
| 4 | 1 | 0 | 0 | On | On | On | 3 | 3/2 | 3/4 | H'0E0A |
| 5 | | | 1 | Off | On | On | 6 | 3 | 3/2 | H'0E0A |
| 6 | | 1 | 0 | Off | Off | Off | 1 | 1/2 | 1/2 | H'0808 |

- Notes: 1. The clock operating mode is the only factor to determine whether to turn the 1/2 frequency divider on or off.
2. For the frequency range of the input clock, see the EXTAL clock input frequency (f_{ex}) and CKIO clock output (f_{op}) in section 23.3.1, Clock and Control Signal Timing.

Table 10.3 (2) Clock Operating Modes (SH7751R)

| Clock Operating Mode | External Pin Combination | | | Frequency (vs. Input Clock) | | | | | | |
|----------------------|--------------------------|-----|-----|-----------------------------|------|-----------|-----------|-------------------------|---------------------|--|
| | MD2 | MD1 | MD0 | PLL1 | PLL2 | CPU Clock | Bus Clock | Peripheral Module Clock | FRQCR Initial Value | |
| 0 | 0 | 0 | 0 | On (×12) | On | 12 | 3 | 3 | H'0E1A | |
| 1 | | | 1 | On (×12) | On | 12 | 3/2 | 3/2 | H'0E2C | |
| 2 | | 1 | 0 | On (×6) | On | 6 | 2 | 1 | H'0E13 | |
| 3 | | | 1 | On (×12) | On | 12 | 4 | 2 | H'0E13 | |
| 4 | 1 | 0 | 0 | On (×6) | On | 6 | 3 | 3/2 | H'0E0A | |
| 5 | | | 1 | On (×12) | On | 12 | 6 | 3 | H'0E0A | |
| 6 | | 1 | 0 | OFF (×6) | OFF | 1 | 1/2 | 1/2 | H'0808 | |

- Notes: 1. The multiplication factor of PLL1 is solely determined by the clock operating mode.

2. For the ranges input clock frequency, see the description of the EXTAL clock input frequency (f_{EX}) and the CKIO clock output (f_{OP}) in section 23.3.1, Clock and Control Signal Timing.

Table 10.4 FRQCR Settings and Internal Clock Frequencies

| FRQCR (Lower 9 Bits) | Frequency Division Ratio of Frequency Divider 2 | | |
|-------------------------|---|-----------|-------------------------|
| | CPU Clock | Bus Clock | Peripheral Module Clock |
| H'000 | 1 | 1 | 1/2 |
| H'002 | | | 1/4 |
| H'004 | | | 1/8 |
| H'008 | | 1/2 | 1/2 |
| H'00A | | | 1/4 |
| H'00C | | | 1/8 |
| H'011 | | 1/3 | 1/3 |
| H'013 | | | 1/6 |
| H'01A | | 1/4 | 1/4 |
| H'01C | | | 1/8 |
| H'023 | | 1/6 | 1/6 |
| H'02C | | 1/8 | 1/8 |
| H'048 | 1/2 | 1/2 | 1/2 |
| H'04A | | | 1/4 |
| H'04C | | | 1/8 |
| H'05A | | 1/4 | 1/4 |
| H'05C | | | 1/8 |
| H'063 | | 1/6 | 1/6 |
| H'06C | | 1/8 | 1/8 |
| H'091 | 1/3 | 1/3 | 1/3 |
| H'093 | | | 1/6 |
| H'0A3 | | 1/6 | 1/6 |
| H'0DA | 1/4 | 1/4 | 1/4 |
| H'0DC | | | 1/8 |
| H'0EC | | 1/8 | |
| H'123 | 1/6 | 1/6 | 1/6 |
| H'16C | 1/8 | 1/8 | 1/8 |

Note: Do not set values other than those shown in the table for the lower 9 bits of FRQCR.

10.4 CPG Register Description

10.4.1 Frequency Control Register (FRQCR)

The frequency control register (FRQCR) is a 16-bit readable/writable register that specifies use/non-use of clock output from the CKIO pin, PLL circuit 1 and 2 on/off control, and the CPU clock, bus clock, and peripheral module clock frequency division ratios. Only word access can be used on FRQCR.

FRQCR is initialized only by a power-on reset via the $\overline{\text{RESET}}$ pin. The initial value of each bit is determined by the clock operating mode.

| | | | | | | | | |
|----------------|------|------|------|------|-------|--------|--------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | CKOEN | PLL1EN | PLL2EN | IFC2 |
| Initial value: | 0 | 0 | 0 | 0 | 1 | 1 | 1 | — |
| R/W: | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IFC1 | IFC0 | BFC2 | BFC1 | BFC0 | PFC2 | PFC1 | PFC0 |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Bits 15 to 12—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 11—Clock Output Enable (CKOEN): Specifies whether a clock is output from the CKIO pin or the CKIO pin is placed in the high-impedance state. When the CKIO pin goes to the high-impedance state, operation continues at the operating frequency before this state was entered. When the CKIO pin becomes high-impedance, it is pulled up.

| Bit 11: CKOEN | Description |
|---------------|--|
| 0 | CKIO pin goes to high-impedance state (pulled up*) |
| 1 | Clock is output from CKIO pin (Initial value) |

Note: * It is not pulled up in hardware standby mode.

Bit 10—PLL Circuit 1 Enable (PLL1EN): Specifies whether PLL circuit 1 is on or off.

| Bit 10: PLL1EN | Description |
|----------------|---------------------------------------|
| 0 | PLL circuit 1 is not used |
| 1 | PLL circuit 1 is used (Initial value) |

Bit 9—PLL Circuit 2 Enable (PLL2EN): Specifies whether PLL circuit 2 is on or off.

| Bit 9: PLL2EN | Description |
|---------------|---------------------------------------|
| 0 | PLL circuit 2 is not used |
| 1 | PLL circuit 2 is used (Initial value) |

Bits 8 to 6—CPU Clock Frequency Division Ratio (IFC): These bits specify the CPU clock frequency division ratio with respect to the input clock, 1/2 frequency divider, or PLL circuit 1 output frequency.

| Bit 8: IFC2 | Bit 7: IFC1 | Bit 6: IFC0 | Description |
|----------------------|-------------|-------------|---------------------------------|
| 0 | 0 | 0 | ×1 |
| | | 1 | ×1/2 |
| | 1 | 0 | ×1/3 |
| | | 1 | ×1/4 |
| 1 | 0 | 0 | ×1/6 |
| | | 1 | ×1/8 |
| Other than the above | | | Setting prohibited (Do not set) |

Bits 5 to 3—Bus Clock Frequency Division Ratio (BFC): These bits specify the bus clock frequency division ratio with respect to the input clock, 1/2 frequency divider, or PLL circuit 1 output frequency.

| Bit 5: BFC2 | Bit 4: BFC1 | Bit 3: BFC0 | Description |
|----------------------|-------------|-------------|---------------------------------|
| 0 | 0 | 0 | ×1 |
| | | 1 | ×1/2 |
| | 1 | 0 | ×1/3 |
| | | 1 | ×1/4 |
| 1 | 0 | 0 | ×1/6 |
| | | 1 | ×1/8 |
| Other than the above | | | Setting prohibited (Do not set) |

Bits 2 to 0—Peripheral Module Clock Frequency Division Ratio (PFC): These bits specify the peripheral module clock frequency division ratio with respect to the input clock, 1/2 frequency divider, or PLL circuit 1 output frequency.

| Bit 2: PFC2 | Bit 1: PFC1 | Bit 0: PFC0 | Description |
|----------------------|-------------|-------------|---------------------------------|
| 0 | 0 | 0 | ×1/2 |
| | | 1 | ×1/3 |
| | 1 | 0 | ×1/4 |
| | | 1 | ×1/6 |
| 1 | 0 | 0 | ×1/8 |
| Other than the above | | | Setting prohibited (Do not set) |

10.5 Changing the Frequency

There are two methods of changing the internal clock frequency: by changing stopping and starting of PLL circuit 1, and by changing the frequency division ratio of each clock. In both cases, control is performed by software by means of the frequency control register. These methods are described below.

10.5.1 Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 Is Off)

When PLL circuit 1 is changed from the stopped to started state, a PLL circuit 1 oscillation stabilization time is required. The oscillation stabilization time count is performed by the on-chip WDT.

1. Set a value in WDT to provide the specified oscillation stabilization time, and stop the WDT. The following settings are necessary:
 - WTCSR register TME bit = 0: WDT stopped
 - WTCSR register CKS2–CKS0 bits: WDT count clock division ratio
 - WTCNT counter: Initial counter value
2. Set the PLL1EN bit to 1.
3. Internal processor operation stops temporarily, and the WDT starts counting up. The internal clock stops and an unstable clock is output to the CKIO pin.
4. After the WDT count overflows, clock supply begins within the chip and the processor resumes operation. The WDT stops after overflowing.

10.5.2 Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 Is On)

When PLL circuit 2 is on, a PLL circuit 1 and PLL circuit 2 oscillation stabilization time is required.

1. Make WDT settings as in section 10.5.1.
2. Set the PLL1EN bit to 1.
3. Internal processor operation stops temporarily, PLL circuit 1 oscillates, and the WDT starts counting up. The internal clock stops and an unstable clock is output to the CKIO pin.
4. After the WDT count overflows, PLL circuit 2 starts oscillating. The WDT resumes its up-count from the value set in step 1 above. During this time, also, the internal clock is stopped and an unstable clock is output to the CKIO pin.
5. After the WDT count overflows, clock supply begins within the chip and the processor resumes operation. The WDT stops after overflowing.

10.5.3 Changing Bus Clock Division Ratio (When PLL Circuit 2 Is On)

If PLL circuit 2 is on when the bus clock frequency division ratio is changed, a PLL circuit 2 oscillation stabilization time is required.

1. Make WDT settings as in section 10.5.1.
2. Set the BFC2–BFC0 bits to the desired value.
3. Internal processor operation stops temporarily, and the WDT starts counting up. The internal clock stops and an unstable clock is output to the CKIO pin.
4. After the WDT count overflows, clock supply begins within the chip and the processor resumes operation. The WDT stops after overflowing.

10.5.4 Changing Bus Clock Division Ratio (When PLL Circuit 2 Is Off)

If PLL circuit 2 is off when the bus clock frequency division ratio is changed, a WDT count is not performed.

1. Set the BFC2–BFC0 bits to the desired value.
2. The set clock is switched to immediately.

10.5.5 Changing CPU or Peripheral Module Clock Division Ratio

When the CPU or peripheral module clock frequency division ratio is changed, a WDT count is not performed.

1. Set the IFC2–IFC0 or PFC2–PFC0 bits to the desired value.
2. The set clock is switched to immediately.

10.7.2 Register Configuration

The WDT has the two registers summarized in table 10.5. These registers control clock selection and timer mode switching.

Table 10.5 WDT Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|--|--------------|------|---------------|------------|----------------|--------------|
| Watchdog timer counter | WTCNT | R/W* | H'00 | H'FFC00008 | H'1FC00008 | R: 8, W: 16* |
| Watchdog timer control/status register | WTCSR | R/W* | H'00 | H'FFC0000C | H'1FC0000C | R: 8, W: 16* |

Note: * Use word-size access when writing. Perform the write with the upper byte set to H'5A or H'A5, respectively. Byte- and longword-size writes cannot be used.
Use byte access when reading.

10.8 WDT Register Descriptions

10.8.1 Watchdog Timer Counter (WTCNT)

The watchdog timer counter (WTCNT) is an 8-bit readable/writable counter that counts up on the selected clock. When WTCNT overflows, a reset is generated in watchdog timer mode, or an interrupt in interval timer mode. WTCNT is initialized to H'00 only by a power-on reset via the RESET pin.

To write to the WTCNT counter, use a word-size access with the upper byte set to H'5A. To read WTCNT, use a byte-size access.

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

10.8.2 Watchdog Timer Control/Status Register (WTCSR)

The watchdog timer control/status register (WTCSR) is an 8-bit readable/writable register containing bits for selecting the count clock and timer mode, and overflow flags.

WTCSR is initialized to H'00 only by a power-on reset via the $\overline{\text{RESET}}$ pin. It retains its value in an internal reset due to WDT overflow. When used to count the clock stabilization time when exiting standby mode, WTCSR retains its value after the counter overflows.

To write to the WTCSR register, use a word-size access with the upper byte set to H'A5. To read WTCSR, use a byte-size access.

| | | | | | | | | |
|----------------|-----|----------------------------|------|------|------|------|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TME | WT/ $\overline{\text{IT}}$ | RSTS | WOVF | IOVF | CKS2 | CKS1 | CKS0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Bit 7—Timer Enable (TME): Specifies starting and stopping of timer operation. Clear this bit to 0 when using the WDT in standby mode or to change a clock frequency.

| Bit 7: TME | Description |
|------------|--|
| 0 | Up-count stopped, WTCNT value retained (Initial value) |
| 1 | Up-count started |

Bit 6—Timer Mode Select (WT/ $\overline{\text{IT}}$): Specifies whether the WDT is used as a watchdog timer or interval timer.

| Bit 6: WT/ $\overline{\text{IT}}$ | Description |
|-----------------------------------|-------------------------------------|
| 0 | Interval timer mode (Initial value) |
| 1 | Watchdog timer mode |

Note: The up-count may not be performed correctly if WT/ $\overline{\text{IT}}$ is modified while the WDT is running.

Bit 5—Reset Select (RSTS): Specifies the kind of reset to be performed when WTCNT overflows in watchdog timer mode. This setting is ignored in interval timer mode.

| Bit 5: RSTS | Description |
|-------------|--------------------------------|
| 0 | Power-on reset (Initial value) |
| 1 | Manual reset |

Bit 4—Watchdog Timer Overflow Flag (WOVF): Indicates that WTCNT has overflowed in watchdog timer mode. This flag is not set in interval timer mode.

| Bit 4: WOVF | Description |
|-------------|---|
| 0 | No overflow (Initial value) |
| 1 | WTCNT has overflowed in watchdog timer mode |

Bit 3—Interval Timer Overflow Flag (IOVF): Indicates that WTCNT has overflowed in interval timer mode. This flag is not set in watchdog timer mode.

| Bit 3: IOVF | Description |
|-------------|---|
| 0 | No overflow (Initial value) |
| 1 | WTCNT has overflowed in interval timer mode |

Bits 2 to 0—Clock Select 2 to 0 (CKS2–CKS0): These bits select the clock used for the WTCNT count from eight clocks obtained by dividing the frequency divider 2 input clock*. The overflow periods shown in the following table are for use of a 33 MHz input clock, with frequency divider 1 off, and PLL circuit 1 on ($\times 6$).

Note: * When PLL1 is switched on or off, the clock following the switch is used.

| Bit 2: CKS2 | Bit 1: CKS1 | Bit 0: CKS0 | Description | |
|-------------|-------------|-------------|----------------------|-----------------|
| | | | Clock Division Ratio | Overflow Period |
| 0 | 0 | 0 | 1/32 (Initial value) | 41 μ s |
| | | 1 | 1/64 | 82 μ s |
| | 1 | 0 | 1/128 | 164 μ s |
| | | 1 | 1/256 | 328 μ s |
| 1 | 0 | 0 | 1/512 | 656 μ s |
| | | 1 | 1/1024 | 1.31 ms |
| | 1 | 0 | 1/2048 | 2.62 ms |
| | | 1 | 1/4096 | 5.25 ms |

Note: The up-count may not be performed correctly if bits CKS2–CKS0 are modified while the WDT is running. Always stop the WDT before modifying these bits.

10.8.3 Notes on Register Access

The watchdog timer counter (WTCNT) and watchdog timer control/status register (WTCSR) differ from other registers in being more difficult to write to. The procedure for writing to these registers is given below.

Writing to WTCNT and WTCSR: These registers must be written to with a word transfer instruction. They cannot be written to with a byte or longword transfer instruction. When writing to WTCNT, perform the transfer with the upper byte set to H'5A and the lower byte containing the write data. When writing to WTCSR, perform the transfer with the upper byte set to H'A5 and the lower byte containing the write data. This transfer procedure writes the lower byte data to WTCNT or WTCSR. The write formats are shown in figure 10.3.

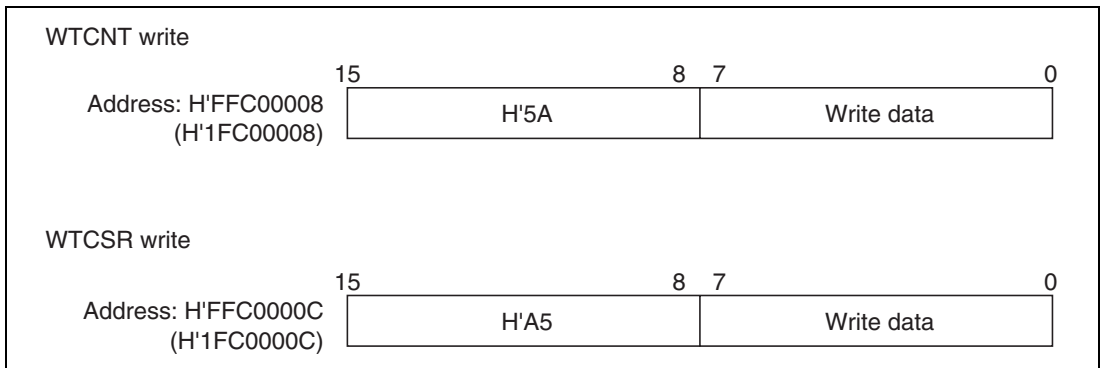


Figure 10.3 Writing to WTCNT and WTCSR

10.9 Using the WDT

10.9.1 Standby Clearing Procedure

The WDT is used when clearing standby mode by means of an NMI or other interrupt. The procedure is shown below. (As the WDT does not operate when standby mode is cleared with a reset, the **RESET** pin should be held low until the clock stabilizes.)

1. Be sure to clear the TME bit in the WTCSR register to 0 before making a transition to standby mode. If the TME bit is set to 1, an inadvertent reset or interval timer interrupt may be caused when the count overflows.
2. Select the count clock to be used with bits CKS2–CKS0 in the WTCSR register, and set the initial value in the WTCNT counter. Make these settings so that the time until the count overflows is at least as long as the clock oscillation stabilization time.
3. Make a transition to standby mode, and stop the clock, by executing a SLEEP instruction.
4. The WDT starts counting on detection of an NMI signal transition edge or an interrupt.
5. When the WDT count overflows, the CPG starts clock supply and the processor resumes operation. The WOVF flag in the WTCSR register is not set at this time.
6. The counter stops at a value of H'00–H'01. The value at which the counter stops depends on the clock ratio.

10.9.2 Frequency Changing Procedure

The WDT is used in a frequency change using the PLL. It is not used when the frequency is changed simply by making a frequency divider switch.

1. Be sure to clear the TME bit in the WTCSR register to 0 before making a frequency change. If the TME bit is set to 1, an inadvertent reset or interval timer interrupt may be caused when the count overflows.
2. Select the count clock to be used with bits CKS2–CKS0 in the WTCSR register, and set the initial value in the WTCNT counter. Make these settings so that the time until the count overflows is at least as long as the clock oscillation stabilization time.
3. When the frequency control register (FRQCR) is modified, the clock stops. The WDT starts counting.
4. When the WDT count overflows, the CPG starts clock supply and the processor resumes operation. The WOVF flag in the WTCSR register is not set at this time.
5. The counter stops at a value of H'00–H'01. The value at which the counter stops depends on the clock ratio.

6. When re-setting WTCNT immediately after modifying the frequency control register (FRQCR), first read the counter and confirm that its value is as described in step 5 above.

10.9.3 Using Watchdog Timer Mode

1. Set the $\overline{WT/IT}$ bit in the WTCSR register to 1, select the type of reset with the RSTS bit, and the count clock with bits CKS2–CKS0, and set the initial value in the WTCNT counter.
2. When the TME bit in the WTCSR register is set to 1, the count starts in watchdog timer mode.
3. During operation in watchdog timer mode, write H'00 to the counter periodically so that it does not overflow.
4. When the counter overflows, the WDT sets the WOVF flag in the WTCSR register to 1, and generates a reset of the type specified by the RSTS bit. The counter then continues counting.

10.9.4 Using Interval Timer Mode

When the WDT is operating in interval timer mode, an interval timer interrupt is generated each time the counter overflows. This enables interrupts to be generated at fixed intervals.

1. Clear the $\overline{WT/IT}$ bit in the WTCSR register to 0, select the count clock with bits CKS2–CKS0, and set the initial value in the WTCNT counter.
2. When the TME bit in the WTCSR register is set to 1, the count starts in interval timer mode.
3. When the counter overflows, the WDT sets the IOVF flag in the WTCSR register to 1, and sends an interval timer interrupt request to INTC. The counter continues counting.

10.10 Notes on Board Design

When Using a Crystal Resonator: Place the crystal resonator and capacitors close to the EXTAL and XTAL pins. To prevent induction from interfering with correct oscillation, ensure that no other signal lines cross the signal lines for these pins.

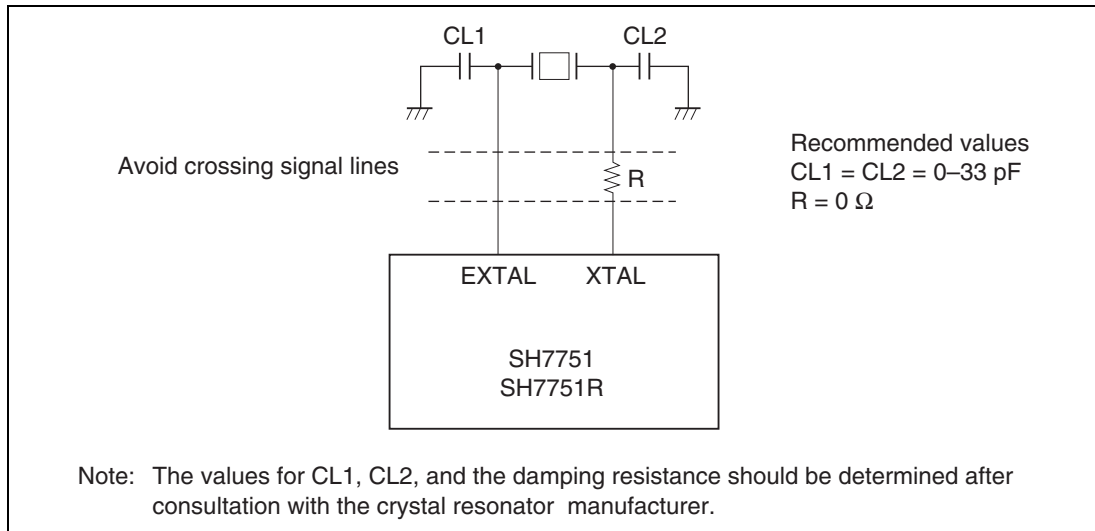


Figure 10.4 Points for Attention when Using Crystal Resonator

When Inputting External Clock from EXTAL Pin: Make no connection to the XTAL pin.

When Using a PLL Oscillator Circuit: Separate VDD-CPG and VSS-CPG from the other VDD and VSS lines at the board power supply source, and insert resistors RCB and RB, and decoupling capacitors CPB and CB, close to the pins.

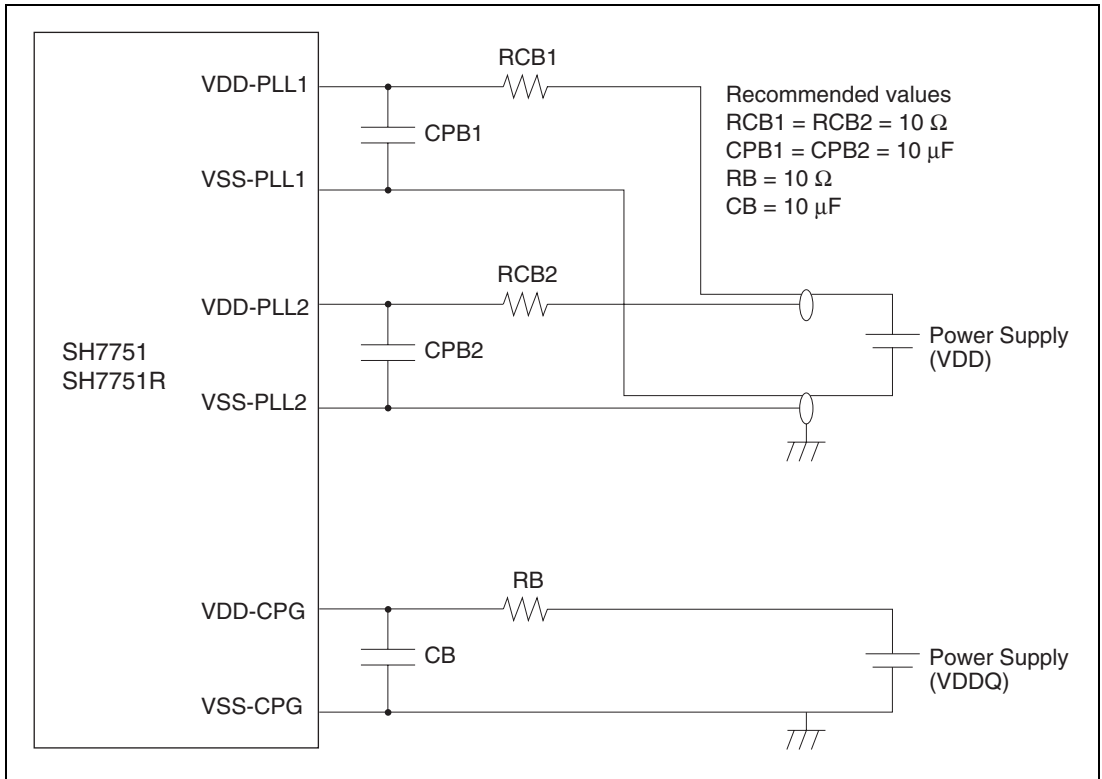


Figure 10.5 Points for Attention when Using PLL Oscillator Circuit

10.11 Usage Notes

10.11.1 Invalid Manual Reset Triggered by Watchdog Timer (SH7751 Only)

Under certain conditions the on-chip watchdog timer (WDT) may trigger an invalid manual reset.

Conditions Under which Problem Occurs: The on-chip WDT triggers an invalid manual reset when all of the following four conditions are satisfied.

1. After the WDT overflows, regardless of the values of the WT/\overline{IT} and RSTS bits in WTCSR.
2. Before the counter (WTCNT) is incremented by the clock specified by the WTCSR.CKS bit.
3. The value of at least one of the TME, WT/\overline{IT} , and RSTS bits in WTCSR is 0.
4. A value of 1 is written to the TME, WT/\overline{IT} , and RSTS bits in WTCSR.

Workaround: A workaround for this problem is to use software to increment WTCNT before writing 1 to the TME, WT/\overline{IT} , and RSTS bits in WTCSR. Specific lines of code for this purpose are listed below.

Example: Add the following lines of code before the instructions for writing 1 to the TME, WT/\overline{IT} , and RSTS bits in WTCSR.

```

MOV.L #WTCNT, R7
MOV.W #H'5A00, R8
MOV.W R8, @R7

MOV.L #WTCSR, R9
MOV.W #H'A580, R10
MOV.W R10, @R9
L.OOP_WDT:
MOV.B @R7, R0
CMP/EQ #H'00, R0
BT L.OOP_WDT

```


Section 11 Realtime Clock (RTC)

11.1 Overview

This LSI includes an on-chip realtime clock (RTC) and a 32.768 kHz crystal oscillation circuit for use by the RTC.

11.1.1 Features

The RTC has the following features.

- Clock and calendar functions (BCD display)
Counts seconds, minutes, hours, day-of-week, days, months, and years.
- 1 to 64 Hz timer (binary display)
The 64 Hz counter register indicates a state of 64 Hz to 1 Hz within the RTC frequency divider
- Start/stop function
- 30-second adjustment function
- Alarm interrupts
Comparison with second, minute, hour, day-of-week, day, month, or year (SH7751R only) can be selected as the alarm interrupt condition
- Periodic interrupts
An interrupt period of 1/256 second, 1/64 second, 1/16 second, 1/4 second, 1/2 second, 1 second, or 2 seconds can be selected
- Carry interrupt
Carry interrupt function indicating a second counter carry, or a 64 Hz counter carry when the 64 Hz counter is read
- Automatic leap year adjustment

11.1.2 Block Diagram

Figure 11.1 shows a block diagram of the RTC.

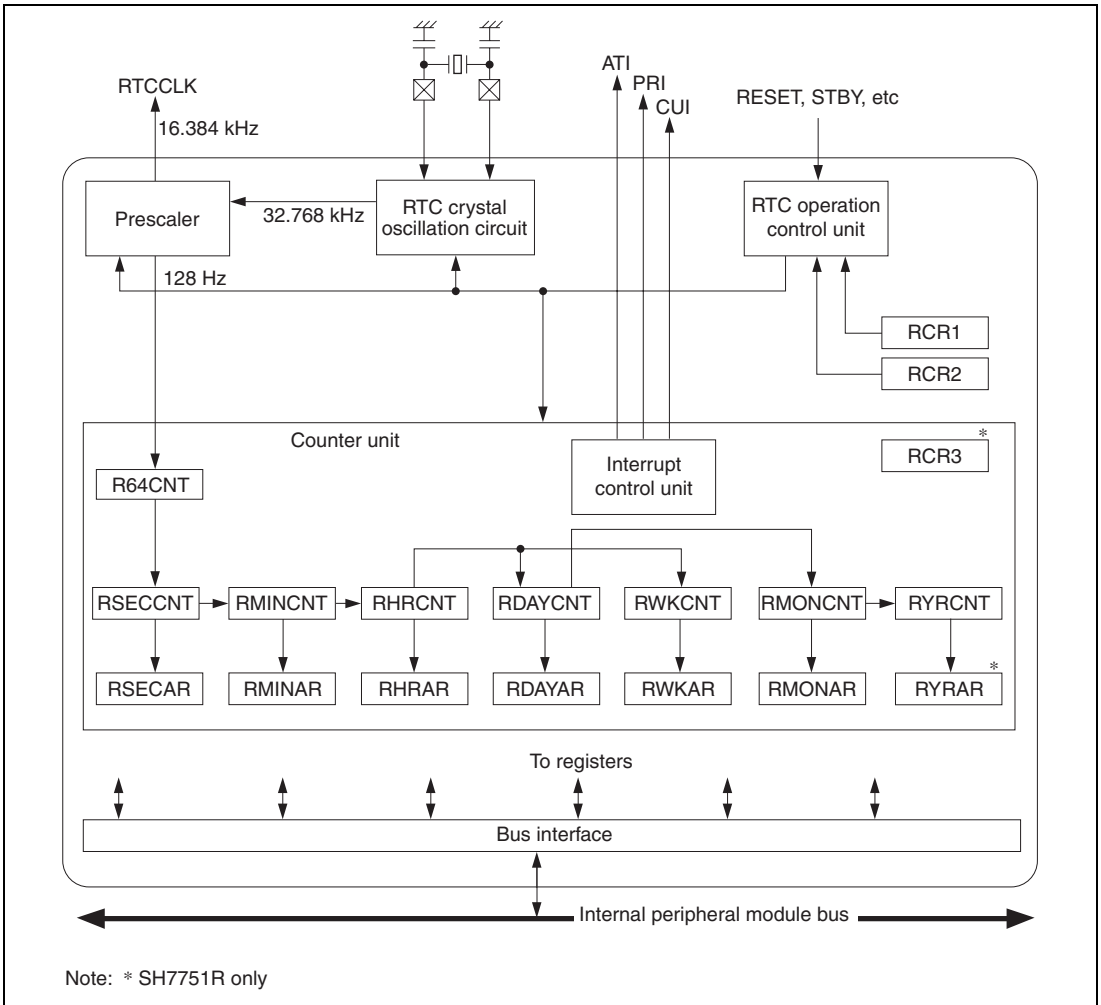


Figure 11.1 Block Diagram of RTC

11.1.3 Pin Configuration

Table 11.1 shows the RTC pins.

Table 11.1 RTC Pins

| Pin Name | Abbreviation | I/O | Function |
|-------------------------------------|--------------|--------|---|
| RTC oscillation circuit crystal pin | EXTAL2 | Input | Connects crystal to RTC oscillation circuit |
| RTC oscillation circuit crystal pin | XTAL2 | Output | Connects crystal to RTC oscillation circuit |
| Clock input/clock output | TCLK | I/O | External clock input pin/input capture control input pin/RTC output pin (shared with TMU) |
| Dedicated RTC power supply | V_{DD-RTC} | — | RTC oscillation circuit power supply pin* |
| Dedicated RTC GND pin | V_{SS-RTC} | — | RTC oscillation circuit GND pin* |

Note: * Power must be supplied to the RTC power supply pins even when the RTC is not used.

11.1.4 Register Configuration

Table 11.2 summarizes the RTC registers.

Table 11.2 RTC Registers

| Name | Abbreviation | R/W | Initialization | | | | Initial Value | P4 Address | Area 7 Address | Access Size |
|---------------------|--------------|-----|----------------|--------------|--------------|-----------|---------------|------------|----------------|-------------|
| | | | Power-On Reset | Manual Reset | Standby Mode | | | | | |
| 64 Hz counter | R64CNT | R | Counts | Counts | Counts | Undefined | H'FFC80000 | H'1FC80000 | 8 | |
| Second counter | RSECCNT | R/W | Counts | Counts | Counts | Undefined | H'FFC80004 | H'1FC80004 | 8 | |
| Minute counter | RMINCNT | R/W | Counts | Counts | Counts | Undefined | H'FFC80008 | H'1FC80008 | 8 | |
| Hour counter | RHRCNT | R/W | Counts | Counts | Counts | Undefined | H'FFC8000C | H'1FC8000C | 8 | |
| Day-of-week counter | RWKCNT | R/W | Counts | Counts | Counts | Undefined | H'FFC80010 | H'1FC80010 | 8 | |
| Day counter | RDAYCNT | R/W | Counts | Counts | Counts | Undefined | H'FFC80014 | H'1FC80014 | 8 | |

| Name | Abbrevia- tion | R/W | Initialization | | | Initial Value | P4 Address | Area 7 Address | Access Size |
|---|-------------------|-----|-------------------------------|-------------------------------|-----------------|-----------------------------|------------|-------------------|----------------|
| | | | Power- On Reset | Manual Reset | Standby Mode | | | | |
| Month counter | RMONCN T | R/W | Counts | Counts | Counts | Undefined | H'FFC80018 | H'1FC80018 | 8 |
| Year counter | RYRCNT | R/W | Counts | Counts | Counts | Undefined | H'FFC8001C | H'1FC8001C | 16 |
| Second alarm register | RSECAR | R/W | Initialized * ¹ | Held | Held | Undefined * ¹ | H'FFC80020 | H'1FC80020 | 8 |
| Minute alarm register | RMINAR | R/W | Initialized * ¹ | Held | Held | Undefined * ¹ | H'FFC80024 | H'1FC80024 | 8 |
| Hour alarm register | RHRAR | R/W | Initialized * ¹ | Held | Held | Undefined * ¹ | H'FFC80028 | H'1FC80028 | 8 |
| Day-of- week alarm register | RWKAR | R/W | Initialized * ¹ | Held | Held | Undefined * ¹ | H'FFC8002C | H'1FC8002C | 8 |
| Day alarm register | RDAYAR | R/W | Initialized * ¹ | Held | Held | Undefined * ¹ | H'FFC80030 | H'1FC80030 | 8 |
| Month alarm register | RMONAR | R/W | Initialized * ¹ | Held | Held | Undefined * ¹ | H'FFC80034 | H'1FC80034 | 8 |
| RTC control register 1 | RRCR1 | R/W | Initialized | Initialized | Held | H'00* ³ | H'FFC80038 | H'1FC80038 | 8 |
| RTC control register 2 | RRCR2 | R/W | Initialized | Initialized * ² | Held | H'09* ⁴ | H'FFC8003C | H'1FC8003C | 8 |
| RTC control register 3* ⁵ | RRCR3 | R/W | Initialized | Held | Held | H'00 | H'FFC80050 | H'1FC80050 | 8 |
| Year alarm register* ⁵ | RYRAR | R/W | Held | Held | Held | Undefined | H'FFC80054 | H'1FC80054 | 16 |

- Notes:
1. The ENB bit in each register is initialized.
 2. Bits other than the RTCEN bit and START bit are initialized.
 3. The value of the CF bit and AF bit is undefined.
 4. The value of the PEF bit is undefined.
 5. SH7751R only

11.2 Register Descriptions

11.2.1 64 Hz Counter (R64CNT)

R64CNT is an 8-bit read-only register that indicates a state of 64 Hz to 1 Hz within the RTC frequency divider.

If this register is read when a carry is generated from the 128 kHz frequency division stage, bit 7 (CF) in RTC control register 1 (RCR1) is set to 1, indicating the simultaneous occurrence of the carry and the 64 Hz counter read. In this case, the read value is not valid, and so R64CNT must be read again after first writing 0 to the CF bit in RCR1 to clear it.

When the RESET bit or ADJ bit in RTC control register 2 (RCR2) is set to 1, the RTC frequency divider is initialized and R64CNT is initialized to H'00.

R64CNT is not initialized by a power-on or manual reset, or in standby mode.

Bit 7 is always read as 0 and cannot be modified.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | — | 1 Hz | 2 Hz | 4 Hz | 8 Hz | 16 Hz | 32 Hz | 64 Hz |
| Initial value: | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R | R | R | R | R | R | R | R |

11.2.2 Second Counter (RSECCNT)

RSECCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded second value in the RTC. It counts on the carry (transition of the R69CNT.1Hz bit from 0 to 1) generated once per second by the 64 Hz counter.

The setting range is decimal 00 to 59. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RSECCNT is not initialized by a power-on or manual reset, or in standby mode.

Bit 7 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|-----------------|-----------|-----------|----------------|-----------|-----------|-----------|
| | — | 10-second units | | | 1-second units | | | |
| Initial value: | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.3 Minute Counter (RMINCNT)

RMINCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded minute value in the RTC. It counts on the carry generated once per minute by the second counter.

The setting range is decimal 00 to 59. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RMINCNT is not initialized by a power-on or manual reset, or in standby mode.

Bit 7 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|-----------------|-----------|-----------|----------------|-----------|-----------|-----------|
| | — | 10-minute units | | | 1-minute units | | | |
| Initial value: | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.4 Hour Counter (RHRCNT)

RHRCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded hour value in the RTC. It counts on the carry generated once per hour by the minute counter.

The setting range is decimal 00 to 23. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RHRCNT is not initialized by a power-on or manual reset, or in standby mode.

Bits 7 and 6 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---------------|-----------|--------------|-----------|-----------|-----------|
| | — | — | 10-hour units | | 1-hour units | | | |
| Initial value: | 0 | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.5 Day-of-Week Counter (RWKCNT)

RWKCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded day-of-week value in the RTC. It counts on the carry generated once per day by the hour counter.

The setting range is decimal 0 to 6. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RWKCNT is not initialized by a power-on or manual reset, or in standby mode.

Bits 7 to 3 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|------------------|-----------|-----------|
| | — | — | — | — | — | Day-of-week code | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | Undefined | Undefined | Undefined |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

| | | | | | | | |
|------------------|-----|-----|-----|-----|-----|-----|-----|
| Day-of-week code | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Day of week | Sun | Mon | Tue | Wed | Thu | Fri | Sat |

11.2.6 Day Counter (RDAYCNT)

RDAYCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded day value in the RTC. It counts on the carry generated once per day by the hour counter.

The setting range is decimal 01 to 31. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RDAYCNT is not initialized by a power-on or manual reset, or in standby mode.

The setting range for RDAYCNT depends on the month and whether the year is a leap year, so care is required when making the setting. Taking the year counter (RYRCNT) value as the year, leap year calculation is performed according to whether or not the value is divisible by 400, 100, and 4.

Bits 7 and 6 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

| | | | | | | | | |
|----------------|---|---|--------------|-----------|-------------|-----------|-----------|-----------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | 10-day units | | 1-day units | | | |
| Initial value: | 0 | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.7 Month Counter (RMONCNT)

RMONCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded month value in the RTC. It counts on the carry generated once per month by the day counter.

The setting range is decimal 01 to 12. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RMONCNT is not initialized by a power-on or manual reset, or in standby mode.

Bits 7 to 5 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

| | | | | | | | | |
|----------------|---|---|---|---------------|---------------|-----------|-----------|-----------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | 10-month unit | 1-month units | | | |
| Initial value: | 0 | 0 | 0 | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |

11.2.8 Year Counter (RYRCNT)

RYRCNT is a 16-bit readable/writable register used as a counter for setting and counting the BCD-coded year value in the RTC. It counts on the carry generated once per year by the month counter.

The setting range is decimal 0000 to 9999. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RYRCNT is not initialized by a power-on or manual reset, or in standby mode.

| | | | | | | | | |
|----------------|-----------------|-----------|-----------|-----------|----------------|-----------|-----------|-----------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | 1000-year units | | | | 100-year units | | | |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| 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 |
| | 10-year units | | | | 1-year units | | | |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.9 Second Alarm Register (RSECAR)

RSECAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded second value counter, RSECCNT. When the ENB bit is set to 1, the RSECAR value is compared with the RSECCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 00 to 59 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RSECAR is initialized to 0 by a power-on reset. The other fields in RSECAR are not initialized by a power-on or manual reset, or in standby mode.

| | | | | | | | | |
|----------------|-----|-----------------|-----------|-----------|----------------|-----------|-----------|-----------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ENB | 10-second units | | | 1-second units | | | |
| Initial value: | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.10 Minute Alarm Register (RMINAR)

RMINAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded minute value counter, RMINCNT. When the ENB bit is set to 1, the RMINAR value is compared with the RMINCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 00 to 59 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RMINAR is initialized by a power-on reset. The other fields in RMINAR are not initialized by a power-on or manual reset, or in standby mode.

| | | | | | | | | |
|----------------|-----|-----------------|-----------|-----------|----------------|-----------|-----------|-----------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ENB | 10-minute units | | | 1-minute units | | | |
| Initial value: | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.11 Hour Alarm Register (RHRAR)

RHRAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded hour value counter, RHCNT. When the ENB bit is set to 1, the RHRAR value is compared with the RHCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 00 to 23 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RHRAR is initialized by a power-on reset. The other fields in RHRAR are not initialized by a power-on or manual reset, or in standby mode.

Bit 6 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-----|---|---------------|-----------|--------------|-----------|-----------|-----------|
| | ENB | — | 10-hour units | | 1-hour units | | | |
| Initial value: | 0 | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.12 Day-of-Week Alarm Register (RWKAR)

RWKAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded day-of-week value counter, RWKCNT. When the ENB bit is set to 1, the RWKAR value is compared with the RWKCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 0 to 6 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RWKAR is initialized by a power-on reset. The other fields in RWKAR are not initialized by a power-on or manual reset, or in standby mode.

Bits 6 to 3 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

| | | | | | | | | |
|----------------|-----|---|---|---|---|------------------|-----------|-----------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ENB | — | — | — | — | Day-of-week code | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | Undefined | Undefined | Undefined |
| R/W: | R/W | R | R | R | R | R/W | R/W | R/W |

| | | | | | | | |
|------------------|-----|-----|-----|-----|-----|-----|-----|
| Day-of-week code | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| Day of week | Sun | Mon | Tue | Wed | Thu | Fri | Sat |

11.2.13 Day Alarm Register (RDAYAR)

RDAYAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded day value counter, RDAYCNT. When the ENB bit is set to 1, the RDAYAR value is compared with the RDAYCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 01 to 31 + ENB bit. The RTC will not operate normally if any other value is set. The setting range for RDAYAR depends on the month and whether the year is a leap year, so care is required when making the setting.

The ENB bit in RDAYAR is initialized by a power-on reset. The other fields in RDAYAR are not initialized by a power-on or manual reset, or in standby mode.

Bit 6 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

| | | | | | | | | |
|----------------|-----|---|--------------|-----------|-------------|-----------|-----------|-----------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ENB | — | 10-day units | | 1-day units | | | |
| Initial value: | 0 | 0 | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |

11.2.14 Month Alarm Register (RMONAR)

RMONAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded month value counter, RMONCNT. When the ENB bit is set to 1, the RMONAR value is compared with the RMONCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 01 to 12 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RMONAR is initialized by a power-on reset. The other fields in RMONAR are not initialized by a power-on or manual reset, or in standby mode.

Bits 6 and 5 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-----|---|---|---------------|---------------|-----------|-----------|-----------|
| | ENB | — | — | 10-month unit | 1-month units | | | |
| Initial value: | 0 | 0 | 0 | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R | R | R/W | R/W | R/W | R/W | R/W |

11.2.15 RTC Control Register 1 (RCR1)

RCR1 is an 8-bit readable/writable register containing a carry flag and alarm flag, plus flags to enable or disable interrupts for these flags.

The CIE and AIE bits are initialized to 0 by a power-on or manual reset; the value of bits other than CIE and AIE is undefined. In standby mode RCR1 is not initialized, and retains its current value.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-----------|-----------|-----------|-----|-----|-----------|-----------|-----------|
| | CF | — | — | CIE | AIE | — | — | AF |
| Initial value: | Undefined | Undefined | Undefined | 0 | 0 | Undefined | Undefined | Undefined |
| R/W: | R/W | R | R | R/W | R/W | R | R | R/W |

Bit 7—Carry Flag (CF): This flag is set to 1 on generation of a second counter carry, or a 64 Hz counter carry when the 64 Hz counter is read. The count register value read at this time is not guaranteed, and so the count register must be read again.

| Bit 7: CF | Description |
|-----------|--|
| 0 | No second counter carry, or 64 Hz counter carry when 64 Hz counter is read [Clearing condition] When 0 is written to CF |
| 1 | Second counter carry, or 64 Hz counter carry when 64 Hz counter is read [Setting conditions] <ul style="list-style-type: none"> • Generation of a second counter carry, or a 64 Hz counter carry when the 64 Hz counter is read • When 1 is written to CF |

Bit 4—Carry Interrupt Enable Flag (CIE): Enables or disables interrupt generation when the carry flag (CF) is set to 1.

| Bit 4: CIE | Description |
|------------|---|
| 0 | Carry interrupt is not generated when CF flag is set to 1 (Initial value) |
| 1 | Carry interrupt is generated when CF flag is set to 1 |

Bit 3—Alarm Interrupt Enable Flag (AIE): Enables or disables interrupt generation when the alarm flag (AF) is set to 1.

| Bit 3: AIE | Description |
|------------|---|
| 0 | Alarm interrupt is not generated when AF flag is set to 1 (Initial value) |
| 1 | Alarm interrupt is generated when AF flag is set to 1 |

Bit 0—Alarm Flag (AF): Set to 1 when the alarm time set in those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1 matches the respective counter values.

| Bit 0: AF | Description |
|-----------|---|
| 0 | Alarm registers and counter values do not match (Initial value) [Clearing condition] When 0 is written to AF |
| 1 | Alarm registers and counter values match* [Setting condition] When alarm registers in which the ENB bit is set to 1 and counter values match* |

Note: * Writing 1 does not change the value.

Bits 6, 5, 2, and 1—Reserved. The initial value of these bits is undefined. A write to these bits is invalid, but the write value should always be 0.

11.2.16 RTC Control Register 2 (RCR2)

RCR2 is an 8-bit readable/writable register used for periodic interrupt control, 30-second adjustment, and frequency divider RESET and RTC count control.

RCR2 is basically initialized to H'09 by a power-on reset, except that the value of the PEF bit is undefined. In a manual reset, bits other than RTCEN and START are initialized, while the value of the PEF bit is undefined. In standby mode RCR2 is not initialized, and retains its current value.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-----------|------|------|------|-------|-----|-------|-------|
| | PEF | PES2 | PES1 | PES0 | RTCEN | ADJ | RESET | START |
| Initial value: | Undefined | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Bit 7—Periodic Interrupt Flag (PEF): Indicates interrupt generation at the interval specified by bits PES2–PES0. When this flag is set to 1, a periodic interrupt is generated.

| Bit 7: PEF | Description |
|------------|--|
| 0 | Interrupt is not generated at interval specified by bits PES2–PES0 [Clearing condition] When 0 is written to PEF |
| 1 | Interrupt is generated at interval specified by bits PES2–PES0 [Setting conditions] <ul style="list-style-type: none"> • Generation of interrupt at interval specified by bits PES2–PES0 • When 1 is written to PEF |

Bits 6 to 4—Periodic Interrupt Enable (PES2–PES0): These bits specify the period for periodic interrupts.

| Bit 6: PES2 | Bit 5: PES1 | Bit 4: PES0 | Description |
|-------------|-------------|-------------|--|
| 0 | 0 | 0 | No periodic interrupt generation (Initial value) |
| | | 1 | Periodic interrupt generated at 1/256-second intervals |
| | 1 | 0 | Periodic interrupt generated at 1/64-second intervals |
| | | 1 | Periodic interrupt generated at 1/16-second intervals |
| 1 | 0 | 0 | Periodic interrupt generated at 1/4-second intervals |
| | | 1 | Periodic interrupt generated at 1/2-second intervals |
| | 1 | 0 | Periodic interrupt generated at 1-second intervals |
| | | 1 | Periodic interrupt generated at 2-second intervals |

Bit 3—Oscillation Circuit Enable (RTCEN): Controls the operation of the RTC crystal oscillation circuit.

| Bit 3: RTCEN | Description |
|--------------|---|
| 0 | RTC crystal oscillation circuit halted |
| 1 | RTC crystal oscillation circuit operating (Initial value) |

Bit 2—30-Second Adjustment (ADJ): Used for 30-second adjustment. When 1 is written to this bit, a value up to 29 seconds is rounded down to 00 seconds, and a value of 30 seconds or more is rounded up to 1 minute. The frequency divider circuits (RTC prescaler and R64CNT) are also reset at this time. This bit always returns 0 if read.

| Bit 2: ADJ | Description | |
|------------|--------------------------------|-----------------|
| 0 | Normal clock operation | (Initial value) |
| 1 | 30-second adjustment performed | |

Bit 1—Reset (RESET): The frequency divider circuits are initialized by writing 1 to this bit. When 1 is written to the RESET bit, the frequency divider circuits (RTC prescaler and R64CNT) are reset and the RESET bit is automatically cleared to 0 (i.e. does not need to be written with 0).

| Bit 1: RESET | Description | |
|--------------|--------------------------------------|-----------------|
| 0 | Normal clock operation | (Initial value) |
| 1 | Frequency divider circuits are reset | |

Bit 0—Start Bit (START): Stops and restarts counter (clock) operation.

| Bit 0: START | Description | |
|--------------|--|-----------------|
| 0 | Second, minute, hour, day, day-of-week, month, and year counters are stopped* | |
| 1 | Second, minute, hour, day, day-of-week, month, and year counters operate normally* | (Initial value) |

Note: * The 64 Hz counter continues to operate unless stopped by means of the RTCEN bit.

11.2.17 RTC Control Register (RCR3) and Year-Alarm Register (RYRAR) (SH7751R Only)

RCR3 and RYRAR are readable/writable registers. RYRAR is the alarm register for the RTC's BCD-coded year-value counter RYRCNT. When the YENB bit of RCR3 is set to 1, the RYRCNT value is compared with the RYRAR value. Comparison between the counter and the alarm register only takes place with the alarm registers in which the ENB and YENB bits are set to 1. The alarm flag of RCR1 is only set to 1 when the respective values all match.

The setting range of RYRAR is decimal 0000 to 9999, and normal operation is not obtained if a value beyond this range is set here.

RCR3 is initialized by a power-on reset, but RYRAR will not be initialized by a power-on or manual reset, or by the device entering standby mode.

Bits 6 to 0 of RCR3 are always read as 0. A write to these bits is invalid. If a value is written to these bits, it should always be 0.

RCR3

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|------|---|---|---|---|---|---|---|
| | YENB | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R | R | R | R | R | R | R |

RYRAR

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | 1000 years | | | | 100 years | | | |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| 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 |
|----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| | 10 years | | | | 1 year | | | |
| Initial value: | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined | Undefined |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

11.3 Operation

Examples of the use of the RTC are shown below.

11.3.1 Time Setting Procedures

Figure 11.2 shows examples of the time setting procedures.

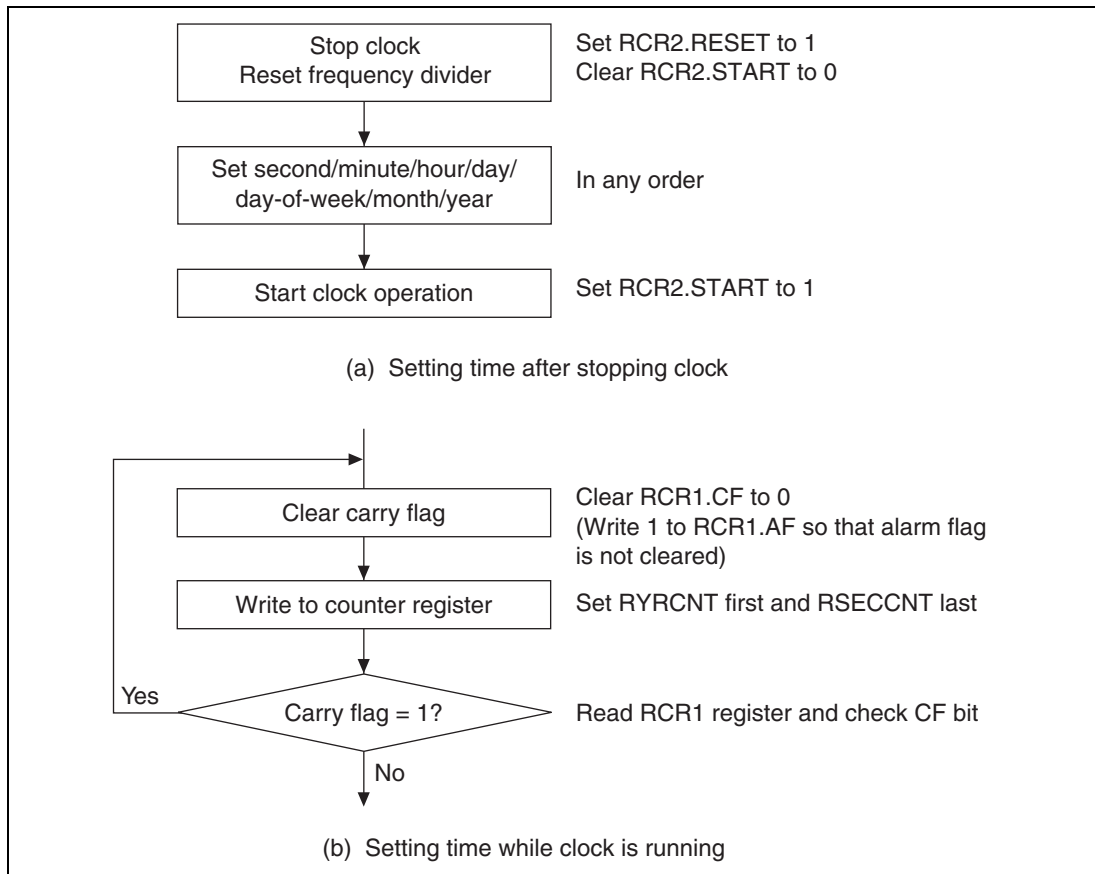


Figure 11.2 Examples of Time Setting Procedures

The procedure for setting the time after stopping the clock is shown in figure 11.2 (a). The programming for this method is simple, and it is useful for setting all the counters, from second to year.

The procedure for setting the time while the clock is running is shown in figure 11.2 (b). This method is useful for modifying only certain counter values (for example, only the second data or hour data). If a carry occurs during the write operation, the write data is automatically updated and there will be an error in the set data. The carry flag should therefore be used to check the write status. If the carry flag (RCR1.CF) is set to 1, the write must be repeated.

The interrupt function can also be used to determine the carry flag status.

11.3.2 Time Reading Procedures

Figure 11.3 shows examples of the time reading procedures.

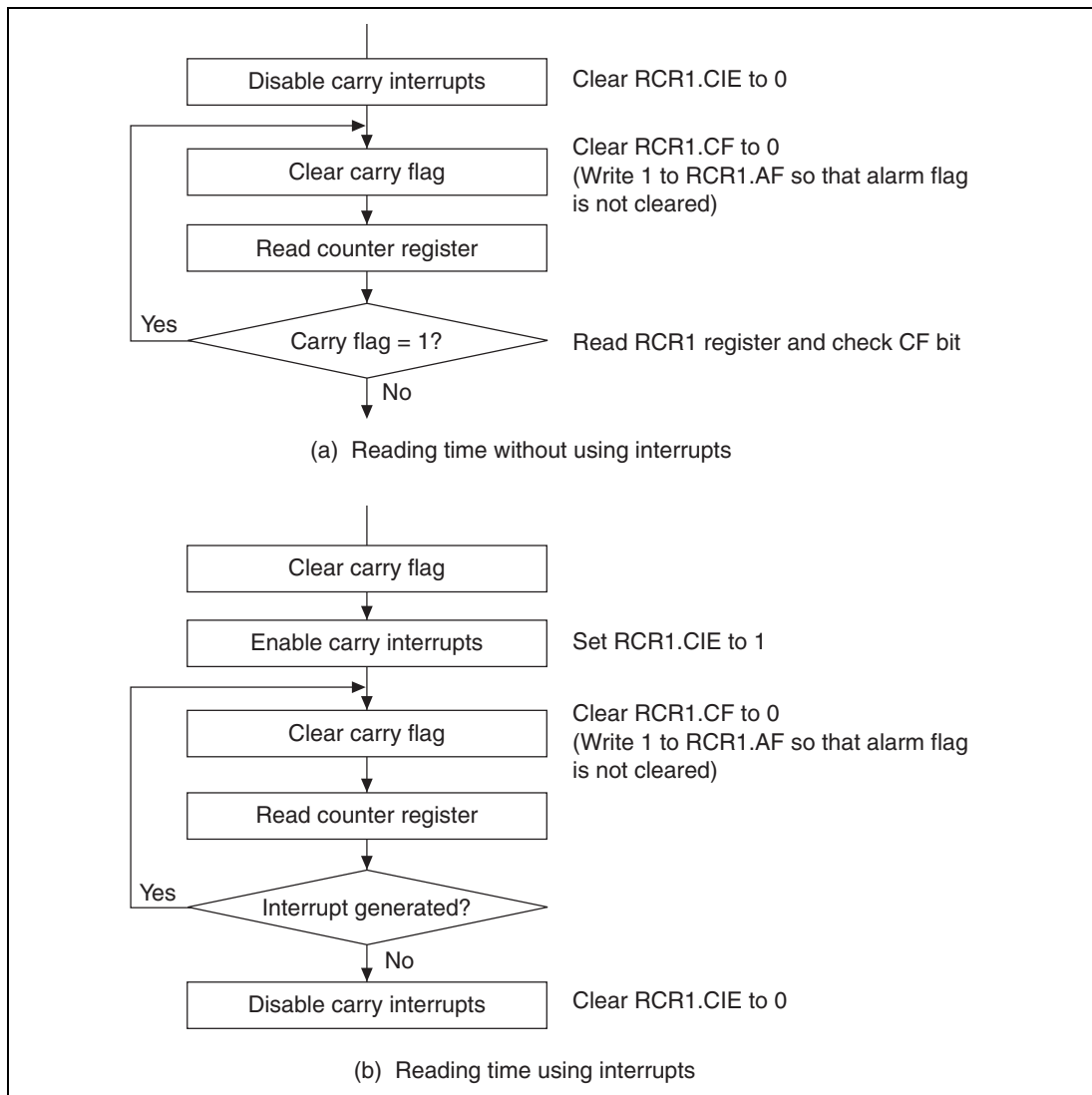


Figure 11.3 Examples of Time Reading Procedures

If a carry occurs while the time is being read, the correct time will not be obtained and the read must be repeated. The procedure for reading the time without using interrupts is shown in figure

11.3 (a), and the procedure using carry interrupts in figure 11.3 (b). The method without using interrupts is normally used to keep the program simple.

11.3.3 Alarm Function

The use of the alarm function is illustrated in figure 11.4.

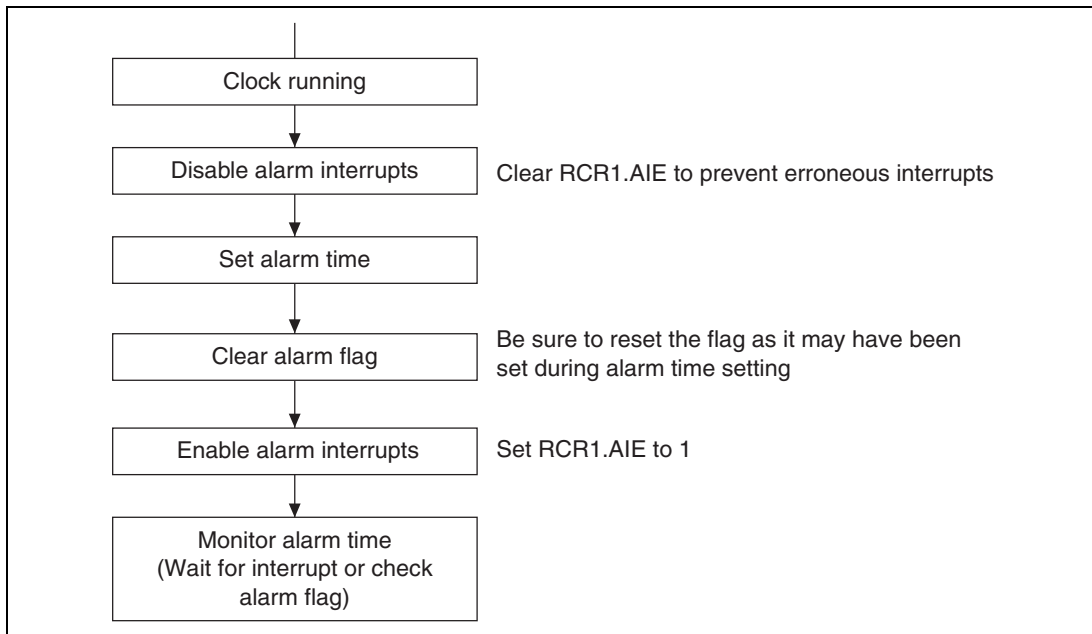


Figure 11.4 Example of Use of Alarm Function

An alarm can be generated by the second, minute, hour, day-of-week, day, month, or year (SH7751R only) value, or a combination of these. Write 1 to the ENB bit in the alarm registers involved in the alarm setting, and set the alarm time in the lower bits. Write 0 to the ENB bit in registers not involved in the alarm setting.

When the counter and the alarm time match, RCR1.AF is set to 1. Alarm detection can be confirmed by reading this bit, but normally an interrupt is used. If 1 has been written to RCR1.AIE, an alarm interrupt is generated in the event of alarm, enabling the alarm to be detected.

The alarm flag remains set while the counter and alarm time match. If the alarm flag is cleared by writing 0 during this period, it will therefore be set again immediately afterward. This needs to be taken into consideration when writing the program.

11.4 Interrupts

There are three kinds of RTC interrupt: alarm interrupts, periodic interrupts, and carry interrupts.

An alarm interrupt request (ATI) is generated when the alarm flag (AF) in RCR1 is set to 1 while the alarm interrupt enable bit (AIE) is also set to 1.

A periodic interrupt request (PRI) is generated when the periodic interrupt enable bits (PES2–PES0) in RCR2 are set to a value other than 000 and the periodic interrupt flag (PEF) is set to 1.

A carry interrupt request (CUI) is generated when the carry flag (CF) in RCR1 is set to 1 while the carry interrupt enable bit (CIE) is also set to 1.

11.5 Usage Notes

11.5.1 Register Initialization

After powering on and making the RCR1 register settings, reset the frequency divider (by setting RCR2.RESET to 1) and make initial settings for all the other registers.

11.5.2 Carry Flag and Interrupt Flag in Standby Mode

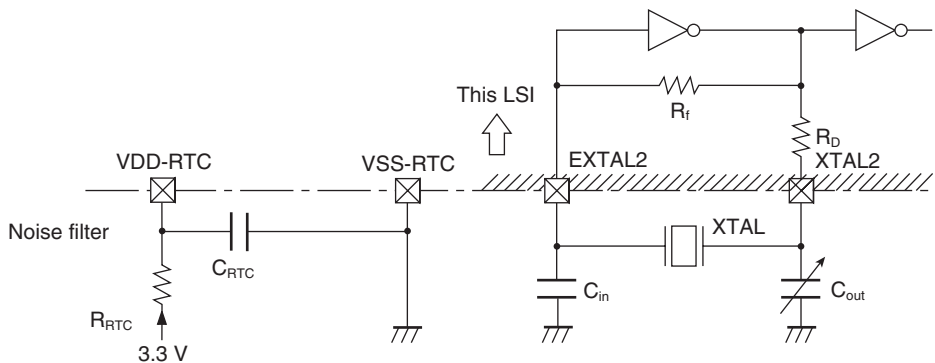
When the carry flag or interrupt flag is set to 1 at the same time this LSI transits to normal mode from standby mode by a reset or interrupt, the flag may not be set to 1. After exiting standby mode, check the counters to judge the flag states if necessary.

11.5.3 Crystal Oscillation Circuit

Crystal oscillation circuit constants (recommended values) are shown in table 11.3, and the RTC crystal oscillation circuit in figure 11.5.

Table 11.3 Crystal Oscillation Circuit Constants (Recommended Values)

| f_{osc} | C_{in} | C_{out} |
|------------|----------|-----------|
| 32.768 kHz | 10–22 pF | 10–22 pF |



- Notes:
1. Select either the C_{in} or C_{out} side for the frequency adjustment variable capacitor according to requirements such as the adjustment range, degree of stability, etc.
 2. Built-in resistance value R_f (typ. value) = 10 M Ω , R_D (typ. value) = 400 k Ω
 3. C_{in} and C_{out} values include floating capacitance due to the wiring. Take care when using a solid-earth board.
 4. The crystal oscillation stabilization time depends on the mounted circuit constants, floating capacitance, etc., and should be decided after consultation with the crystal resonator manufacturer.
 5. Place the crystal resonator and load capacitors C_{in} and C_{out} as close as possible to the chip. (Correct oscillation may not be possible if there is externally induced noise in the EXTAL2 and XTAL2 pins.)
 6. Ensure that the crystal resonator connection pin (EXTAL2 and XTAL2) wiring is routed as far away as possible from other power lines (except GND) and signal lines.
 7. Insert a noise filter in the RTC power supply.

Figure 11.5 Example of Crystal Oscillation Circuit Connection

Section 12 Timer Unit (TMU)

12.1 Overview

This LSI includes an on-chip 32-bit timer unit (TMU) comprising five 32-bit timer channels (channels 0 to 4).

12.1.1 Features

The TMU has the following features.

- Auto-reload type 32-bit down-counter provided for each channel
- Input capture function provided in channel 2
- Selection of rising edge or falling edge as external clock input edge when external clock is selected or input capture function is used
- 32-bit timer constant register for auto-reload use, readable/writable at any time, and 32-bit down-counter provided for each channel
- Selection of seven counter input clocks for channels 0 to 2
External clock (TCLK), on-chip RTC output clock, five internal clocks (Pck/4, Pck/16, Pck/64, Pck/256, Pck/1024) (Pck is the peripheral module clock)
- Selection of five internal clocks for channels 3 and 4
- Channels 0 to 2 can also operate in module standby mode when the on-chip RTC output clock is selected as the counter input clock; that is, timer operation continues even when the clock has been stopped for the TMU.
Timer count operations using an external or internal clock are only possible when a clock is supplied to the timer unit.
- Two interrupt sources
One underflow source (channels 0 to 4) and one input capture source (channel 2)
- DMAC data transfer request capability
On channel 2, a data transfer request is sent to the DMAC when an input capture interrupt is generated.

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the TMU.

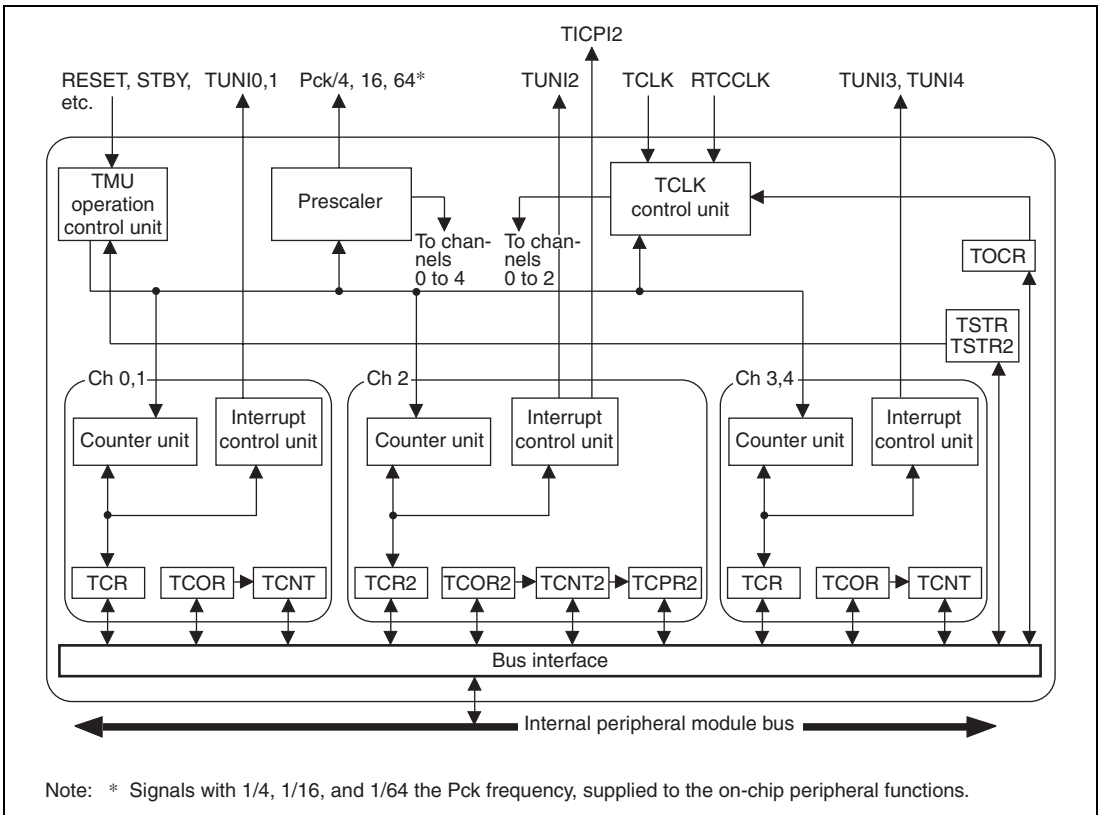


Figure 12.1 Block Diagram of TMU

12.1.3 Pin Configuration

Table 12.1 shows the TMU pins.

Table 12.1 TMU Pins

| Pin Name | Abbreviation | I/O | Function |
|--------------------------|--------------|-----|---|
| Clock input/clock output | TCLK | I/O | External clock input pin/input capture control input pin/RTC output pin (shared with RTC) |

12.1.4 Register Configuration

Table 12.2 summarizes the TMU registers.

Table 12.2 TMU Registers

| Channel | Name | Abbreviation | R/W | Initialization | | | Initial Value | P4 Address | Area 7 Address | Access Size |
|---------|-------------------------------|--------------|-----|----------------|--------------|---------------------------|---------------|------------|----------------|-------------|
| | | | | Power-On Reset | Manual Reset | Stand-by Mode | | | | |
| Common | Timer output control register | TOCR | R/W | Initialized | Initialized | Held | H'00 | H'FFD80000 | H'1FD80000 | 8 |
| | Timer start register | TSTR | R/W | Initialized | Initialized | Initialized* ¹ | H'00 | H'FFD80004 | H'1FD80004 | 8 |
| | Timer start register 2 | TSTR2 | R/W | Initialized | Held | Held | H'00 | H'FE100004 | H'1E100004 | 8 |
| 0 | Timer constant register 0 | TCOR0 | R/W | Initialized | Initialized | Held | H'FFFFFFFF | H'FFD80008 | H'1FD80008 | 32 |
| | Timer counter 0 | TCNT0 | R/W | Initialized | Initialized | Held* ² | H'FFFFFFFF | H'FFD8000C | H'1FD8000C | 32 |
| | Timer control register 0 | TCR0 | R/W | Initialized | Initialized | Held | H'0000 | H'FFD80010 | H'1FD80010 | 16 |
| 1 | Timer constant register 1 | TCOR1 | R/W | Initialized | Initialized | Held | H'FFFFFFFF | H'FFD80014 | H'1FD80014 | 32 |
| | Timer counter 1 | TCNT1 | R/W | Initialized | Initialized | Held* ² | H'FFFFFFFF | H'FFD80018 | H'1FD80018 | 32 |
| | Timer control register 1 | TCR1 | R/W | Initialized | Initialized | Held | H'0000 | H'FFD8001C | H'1FD8001C | 16 |
| 2 | Timer constant register 2 | TCOR2 | R/W | Initialized | Initialized | Held | H'FFFFFFFF | H'FFD80020 | H'1FD80020 | 32 |
| | Timer counter 2 | TCNT2 | R/W | Initialized | Initialized | Held* ² | H'FFFFFFFF | H'FFD80024 | H'1FD80024 | 32 |
| | Timer control register 2 | TCR2 | R/W | Initialized | Initialized | Held | H'0000 | H'FFD80028 | H'1FD80028 | 16 |
| | Input capture register | TCPR2 | R | Held | Held | Held | Undefined | H'FFD8002C | H'1FD8002C | 32 |

| Channel | Name | Abbreviation | R/W | Initialization | | | Initial Value | P4 Address | Area 7 Address | Access Size |
|---------|---------------------------|--------------|-----|----------------|--------------|--------------|---------------|------------|----------------|-------------|
| | | | | Power-On Reset | Manual Reset | Standby Mode | | | | |
| 3 | Timer constant register 3 | TCOR3 | R/W | Initialized | Held | Held | H'FFFFFFF | H'FE10008 | H'1E10008 | 32 |
| | Timer counter 3 | TCNT3 | R/W | Initialized | Held | Held | H'FFFFFFF | H'FE1000C | H'1E1000C | 32 |
| | Timer control register 3 | TCR3 | R/W | Initialized | Held | Held | H'0000 | H'FE10010 | H'1E10010 | 16 |
| 4 | Timer constant register 4 | TCOR4 | R/W | Initialized | Held | Held | H'FFFFFFF | H'FE10014 | H'1E10014 | 32 |
| | Timer counter 4 | TCNT4 | R/W | Initialized | Held | Held | H'FFFFFFF | H'FE10018 | H'1E10018 | 32 |
| | Timer control register 4 | TCR4 | R/W | Initialized | Held | Held | H'0000 | H'FE1001C | H'1E1001C | 16 |

- Notes: 1. Not initialized in module standby mode when the input clock is the on-chip RTC output clock.
 2. Counts in module standby mode when the input clock is the on-chip RTC output clock.

12.2 Register Descriptions

12.2.1 Timer Output Control Register (TOCR)

TOCR is an 8-bit readable/writable register that specifies whether external pin TCLK is used as the external clock or input capture control input pin, or as the on-chip RTC output clock output pin.

TOCR is initialized to H'00 by a power-on or manual reset, but is not initialized in standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|---|---|------|
| | — | — | — | — | — | — | — | TCOE |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R/W |

Bits 7 to 1—Reserved: These bits are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 0—Timer Clock Pin Control (TCOE): Specifies whether timer clock pin TCLK is used as the external clock or input capture control input pin, or as the on-chip RTC output clock output pin.

| Bit 0: TCOE | Description |
|-------------|---|
| 0 | Timer clock pin (TCLK) is used as external clock input or input capture control input pin (Initial value) |
| 1 | Timer clock pin (TCLK) is used as on-chip RTC output clock output pin* |

Note: * Low-level output in standby mode; high-impedance output in hardware standby mode.

12.2.2 Timer Start Register (TSTR)

TSTR is an 8-bit readable/writable register that specifies whether the channel 0–2 timer counters (TCNT) are operated or stopped.

TSTR is initialized to H'00 by a power-on or manual reset. In module standby mode, TSTR is not initialized when the input clock selected by each channel is the on-chip RTC output clock (RTCCLK), and is initialized only when the input clock is the external clock (TCLK) or internal clock (Pck).

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|------|------|------|
| | — | — | — | — | — | STR2 | STR1 | STR0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

Bits 7 to 3—Reserved: These bits are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 2—Counter Start 2 (STR2): Specifies whether timer counter 2 (TCNT2) is operated or stopped.

| Bit 2: STR2 | Description |
|-------------|--|
| 0 | TCNT2 count operation is stopped (Initial value) |
| 1 | TCNT2 performs count operation |

Bit 1—Counter Start 1 (STR1): Specifies whether timer counter 1 (TCNT1) is operated or stopped.

| Bit 1: STR1 | Description |
|-------------|--|
| 0 | TCNT1 count operation is stopped (Initial value) |
| 1 | TCNT1 performs count operation |

Bit 0—Counter Start 0 (STR0): Specifies whether timer counter 0 (TCNT0) is operated or stopped.

| Bit 0: STR0 | Description |
|-------------|--|
| 0 | TCNT0 count operation is stopped (Initial value) |
| 1 | TCNT0 performs count operation |

12.2.3 Timer Start Register 2 (TSTR2)

TSTR2 is an 8-bit readable/writable register that specifies whether the channel 3 and 4 timer counters (TCNT) are operated or stopped.

TSTR2 is initialized to H'00 by a power-on reset. TSTR retain their contents in standby mode. When standby mode is entered when the value of either STR3 or STR4 is 1, the count halts when the peripheral module clock stops and restarts when the clock supply is resumed.

| | | | | | | | | |
|----------------|---|---|---|---|---|---|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | STR4 | STR3 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R/W | R/W |

Bits 7 to 2—Reserved: These bits are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 1—Counter Start 4 (STR4): Specifies whether timer counter 4 (TCNT4) is operated or stopped.

| Bit 1: STR4 | Description |
|-------------|--|
| 0 | TCNT4 count operation is stopped (Initial value) |
| 1 | TCNT4 performs count operation |

Bit 0—Counter Start 3 (STR3): Specifies whether timer counter 3 (TCNT3) is operated or stopped.

| Bit 0: STR3 | Description |
|-------------|--|
| 0 | TCNT3 count operation is stopped (Initial value) |
| 1 | TCNT3 performs count operation |

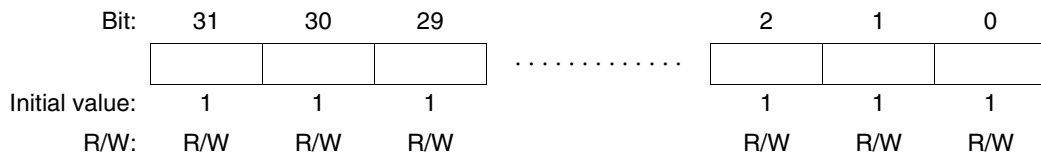
12.2.4 Timer Constant Registers (TCOR)

The TCOR registers are 32-bit readable/writable registers. There are five TCOR registers, one for each channel.

When a TCNT counter underflows while counting down, the TCOR value is set in that TCNT, which continues counting down from the set value.

The TCOR registers in channels 0 to 2 are initialized to H'FFFFFFFF by a power-on or manual reset, but are not initialized and retain their contents in standby mode.

The TCOR registers in channels 3 and 4 are initialized to H'FFFFFFFF by a power-on reset, but are not initialized and retain their contents by a manual reset or in standby mode.



12.2.5 Timer Counters (TCNT)

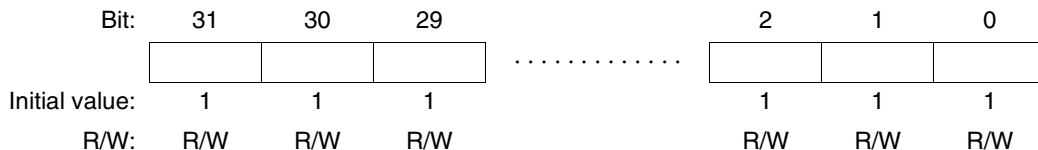
The TCNT registers are 32-bit readable/writable registers. There are five TCNT registers, one for each channel.

Each TCNT counts down on the input clock selected by TPSC2–TPSC0 in the timer control register (TCR).

When a TCNT counter underflows while counting down, the underflow flag (UNF) is set in the corresponding timer control register (TCR). At the same time, the timer constant register (TCOR) value is set in TCNT, and the count-down operation continues from the set value.

The TCNT registers in channels 0 to 2 are initialized to H'FFFFFFFF by a power-on or manual reset, but are not initialized and retain their contents in standby mode.

The TCNT registers in channels 3 and 4 are initialized to H'FFFFFFF by a power-on reset, but are not initialized and retain their contents by a manual reset or in standby mode.



In channels 0 to 2, when the input clock is the on-chip RTC output clock (RTCCLK), TCNT counts even in module standby mode (that is, when the clock for the TMU is stopped). When the input clock is the external clock (TCLK) or internal clock (Pck), TCNT contents are retained in standby mode.

12.2.6 Timer Control Registers (TCR)

The TCR registers are 16-bit readable/writable registers. There are five TCR registers, one for each channel.

Each TCR selects the count clock, specifies the edge when an external clock is selected in channels 0 to 2, and controls interrupt generation when the flag indicating timer counter (TCNT) underflow is set to 1. TCR2 is also used for channel 2 input capture control, and control of interrupt generation in the event of input capture.

The TCR registers in channels 0 to 2 are initialized to H'0000 by a power-on or manual reset, but are not initialized in standby mode.

The TCR registers in channels 3 and 4 are initialized to H'0000 by a power-on reset, but are not initialized by a manual reset or in standby mode.

1. Channel 0 and 1 TCR bit configuration

| | | | | | | | | |
|----------------|----|----|------|-------|-------|-------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | UNF |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | UNIE | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

2. Channel 2 TCR bit configuration

| | | | | | | | | |
|----------------|-------|-------|------|-------|-------|-------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | ICPF | UNF |
| 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 |
| | ICPE1 | ICPE0 | UNIE | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

3. Channel 3 and 4 TCR bit configuration

| | | | | | | | | |
|----------------|----|----|------|----|----|-------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | UNF |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | UNIE | — | — | TPSC2 | TPSC1 | TPSC0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R/W | R | R | R/W | R/W | R/W |

Bits 15 to 9, 7, and 6 (Channels 0 and 1); Bits 15 to 10 (Channel 2)—Reserved: These bits are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 9—Input Capture Interrupt Flag (ICPF) (Channel 2 Only): Status flag, provided in channel 2 only, that indicates the occurrence of input capture.

| Bit 9: ICPF | Description |
|-------------|---|
| 0 | Input capture has not occurred (Initial value) [Clearing condition] When 0 is written to ICPF |
| 1 | Input capture has occurred [Setting condition] When input capture occurs* |

Note: * Writing 1 does not change the value.

Bit 8—Underflow Flag (UNF): Status flag that indicates the occurrence of underflow.

| Bit 8: UNF | Description |
|------------|--|
| 0 | TCNT has not underflowed (Initial value) [Clearing condition] When 0 is written to UNF |
| 1 | TCNT has underflowed [Setting condition] When TCNT underflows* |

Note: * Writing 1 does not change the value.

Bits 7 and 6—Input Capture Control (ICPE1, ICPE0) (Channel 2 Only): These bits, provided in channel 2 only, specify whether the input capture function is used, and control enabling or disabling of interrupt generation when the function is used.

When the input capture function is used, a data transfer request is sent to the DMAC in the event of input capture.

When using the input capture function, the TCLK pin must be designated as an input pin with the TCOE bit in the TOCR register. The CKEG bits specify whether the rising edge or falling edge of the TCLK signal is used to set the TCNT2 value in the input capture register (TCPR2).

The TCNT2 value is set in TCPR2 only when the TCR2.ICPF bit is 0. When the TCR2.ICPF bit is 1, TCPR2 is not set in the event of input capture. When input capture occurs, a DMAC transfer request is generated regardless of the value of the TCR2.ICPF bit. However, a new DMAC transfer request is not generated until processing of the previous request is finished.

| Bit 7: ICPE1 | Bit 6: ICPE0 | Description |
|--------------|--------------|---|
| 0 | 0 | Input capture function is not used (Initial value) |
| | 1 | Reserved (Do not set) |
| 1 | 0 | Input capture function is used, but interrupt due to input capture (TICPI2) is not enabled Data transfer request is sent to DMAC in the event of input capture |
| | 1 | Input capture function is used, and interrupt due to input capture (TICPI2) is enabled Data transfer request is sent to DMAC in the event of input capture |

Bit 5—Underflow Interrupt Control (UNIE): Controls enabling or disabling of interrupt generation when the UNF status flag is set to 1, indicating TCNT underflow.

| Bit 5: UNIE | Description |
|-------------|--|
| 0 | Interrupt due to underflow (TUNI) is not enabled (Initial value) |
| 1 | Interrupt due to underflow (TUNI) is enabled |

Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0): In channels 0 to 2, these bits select the external clock input edge when an external clock is selected or the input capture function is used.

| Bit 4: CKEG1 | Bit 3: CKEG0 | Description |
|--------------|--------------|---|
| 0 | 0 | Count/input capture register set on rising edge (Initial value) |
| | 1 | Count/input capture register set on falling edge |
| 1 | X | Count/input capture register set on both rising and falling edges |

Note: X: 0 or 1 (don't care)

Bits 2 to 0—Timer Prescaler 2 to 0 (TPSC2–TPSC0): In channels 0 to 2, these bits select the TCNT count clock.

When the on-chip RTC output clock is selected as the count clock for a channel, that channel can operate even in module standby mode. When another clock is selected, the channel does not operate in standby mode.

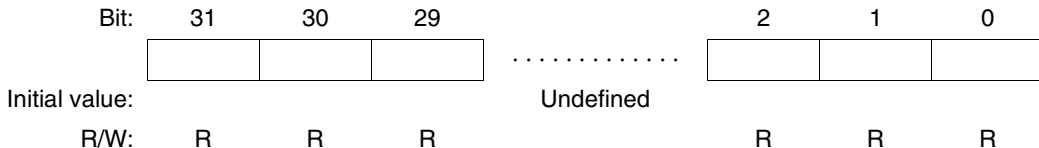
| Bit 2: TPSC2 | Bit 1: TPSC1 | Bit 0: TPSC0 | Description |
|--------------|--------------|--------------|---|
| 0 | 0 | 0 | Counts on Pck/4 (Initial value) |
| | | 1 | Counts on Pck/16 |
| | 1 | 0 | Counts on Pck/64 |
| | | 1 | Counts on Pck/256 |
| 1 | 0 | 0 | Counts on Pck/1024 |
| | | 1 | Reserved (Do not set) |
| | 1 | 0 | Counts on on-chip RTC output clock (Do not set in channels 3 and 4) |
| | | 1 | Counts on external clock (Do not set in channels 3 and 4) |

12.2.7 Input Capture Register 2 (TCPR2)

TCPR2 is a 32-bit read-only register for use with the input capture function, provided only in channel 2.

The input capture function is controlled by means of the input capture control bits (ICPE1, ICPE0) and clock edge bits (CKEG1, CKEG0) in TCR2. When input capture occurs, the TCNT2 value is copied into TCPR2. The value is set in TCPR2 only when the ICPF bit in TCR2 is 0.

TCPR2 is not initialized by a power-on or manual reset, or in standby mode.



12.3 Operation

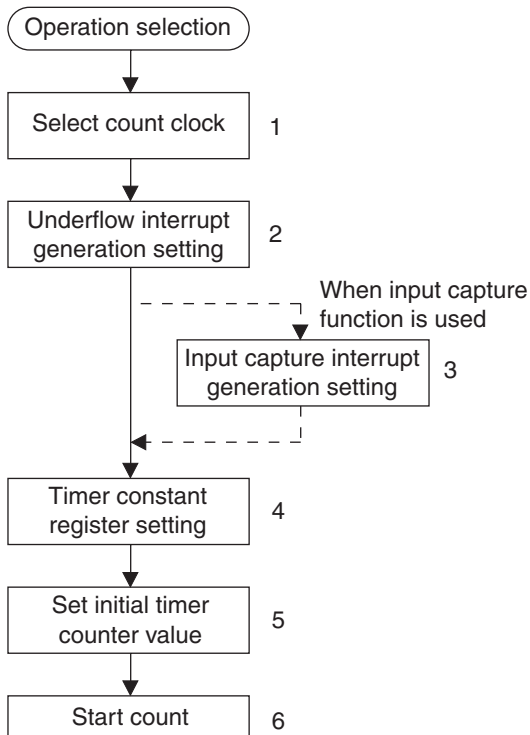
Each channel has a 32-bit timer counter (TCNT) that performs count-down operations, and a 32-bit timer constant register (TCOR). The channels have an auto-reload function that allows cyclic count operations, and can also perform external event counting. Channel 2 also has an input capture function.

12.3.1 Counter Operation

When one of bits STR0–STR4 is set to 1 in the timer start register (TSTR, TSTR2), the timer counter (TCNT) for the corresponding channel starts counting. When TCNT underflows, the UNF flag is set in the corresponding timer control register (TCR). If the UNIE bit in TCR is set to 1 at this time, an interrupt request is sent to the CPU. At the same time, the value is copied from TCOR into TCNT, and the count-down continues (auto-reload function).

Example of Count Operation Setting Procedure: Figure 12.2 shows an example of the count operation setting procedure.

1. Select the count clock with bits TPSC2–TPSC0 in the timer control register (TCR). When an external clock in channels 0 to 2 is selected, set the TCLK pin to input mode with the TCOE bit in TOCR, and select the external clock edge with bits CKEG1 and CKEG0 in TCR.
2. Specify whether an interrupt is to be generated on TCNT underflow with the UNIE bit in TCR.
3. When the input capture function is used, set the ICPE bits in TCR, including specification of whether the interrupt function is to be used.
4. Set a value in the timer constant register (TCOR).
5. Set the initial value in the timer counter (TCNT).
6. Set the STR bit to 1 in the timer start register (TSTR, TSTR2) to start the count.



Note: When an interrupt is generated, clear the source flag in the interrupt handler. If the interrupt enabled state is set without clearing the flag, another interrupt will be generated.

Figure 12.2 Example of Count Operation Setting Procedure

Auto-Reload Count Operation: Figure 12.3 shows the TCNT auto-reload operation.

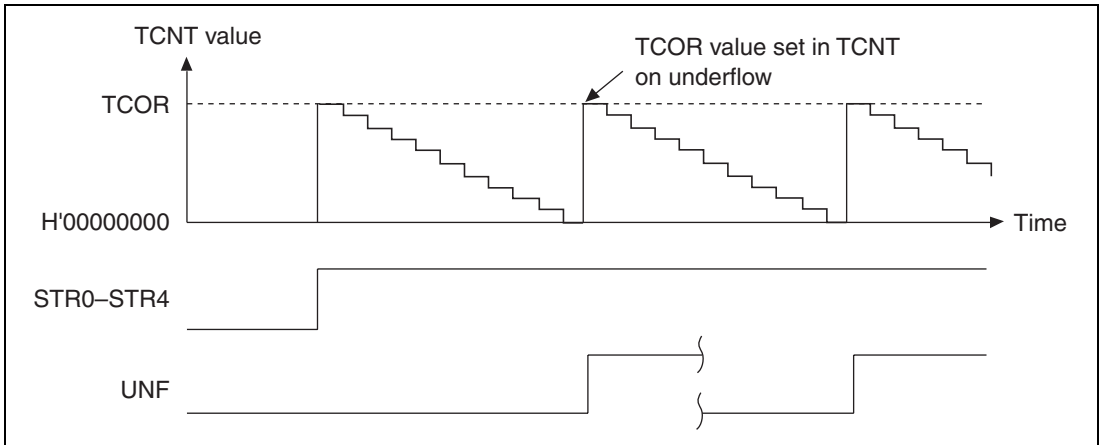


Figure 12.3 TCNT Auto-Reload Operation

TCNT Count Timing:

- Operating on internal clock

Any of five count clocks (Pck/4, Pck/16, Pck/64, Pck/256, or Pck/1024) scaled from the peripheral module clock can be selected as the count clock by means of the TPSC2-TPSC0 bits in TCR.

Figure 12.4 shows the timing in this case.

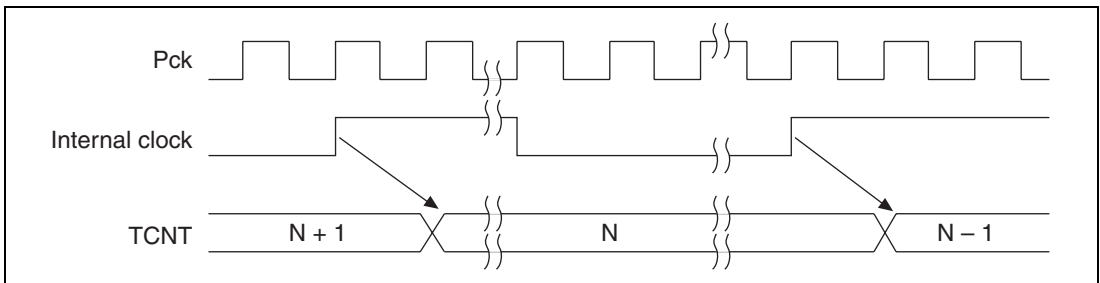


Figure 12.4 Count Timing when Operating on Internal Clock

- Operating on external clock

In channels 0 to 2, external clock pin (TCLK) input can be selected as the timer clock by means of the TPSC2–TPSC0 bits in TCR. The detected edge (rising, falling, or both edges) can be selected with the CKEG1 and CKEG0 bits in TCR.

Figure 12.5 shows the timing for both-edge detection.

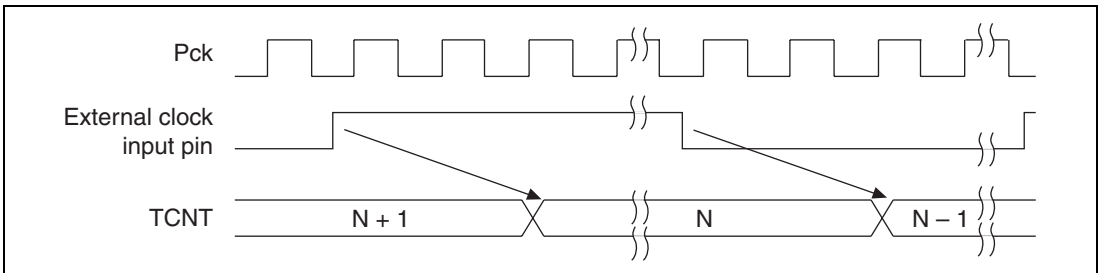


Figure 12.5 Count Timing when Operating on External Clock

- Operating on on-chip RTC output clock

In channels 0 to 2, the on-chip RTC output clock can be selected as the timer clock by means of the TPSC2–TPSC0 bits in TCR. Figure 12.6 shows the timing in this case.

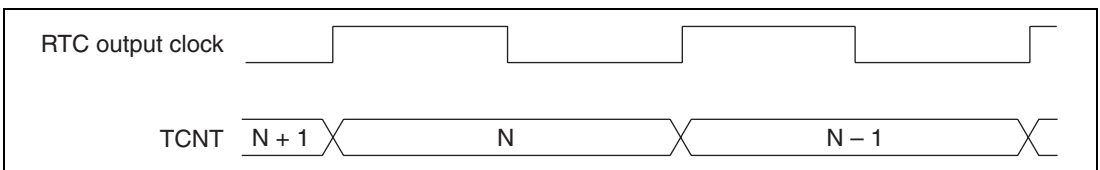


Figure 12.6 Count Timing when Operating on On-Chip RTC Output Clock

12.3.2 Input Capture Function

Channel 2 has an input capture function.

The procedure for using the input capture function is as follows:

1. Use the TCOE bit in the timer output control register (TOCR) to set the TCLK pin to input mode.
2. Use bits TPSC2–TPSC0 in the timer control register (TCR) to set an internal clock or the on-chip RTC output clock as the timer operating clock.
3. Use bits IPCE1 and IPCE0 in TCR to specify use of the input capture function, and whether interrupts are generated when this function is used.

4. Use bits CKEG1 and CKEG0 in TCR to specify whether the rising or falling edge of the TCLK signal is to be used to set the timer counter (TCNT) value in the input capture register (TCPR2).

This function cannot be used in standby mode.

When input capture occurs, the TCNT2 value is set in TCPR2 only when the ICPF bit in TCR2 is 0. Also, a new DMAC transfer request is not generated until processing of the previous request is finished.

Figure 12.7 shows the operation timing when the input capture function is used (with TCLK rising edge detection).

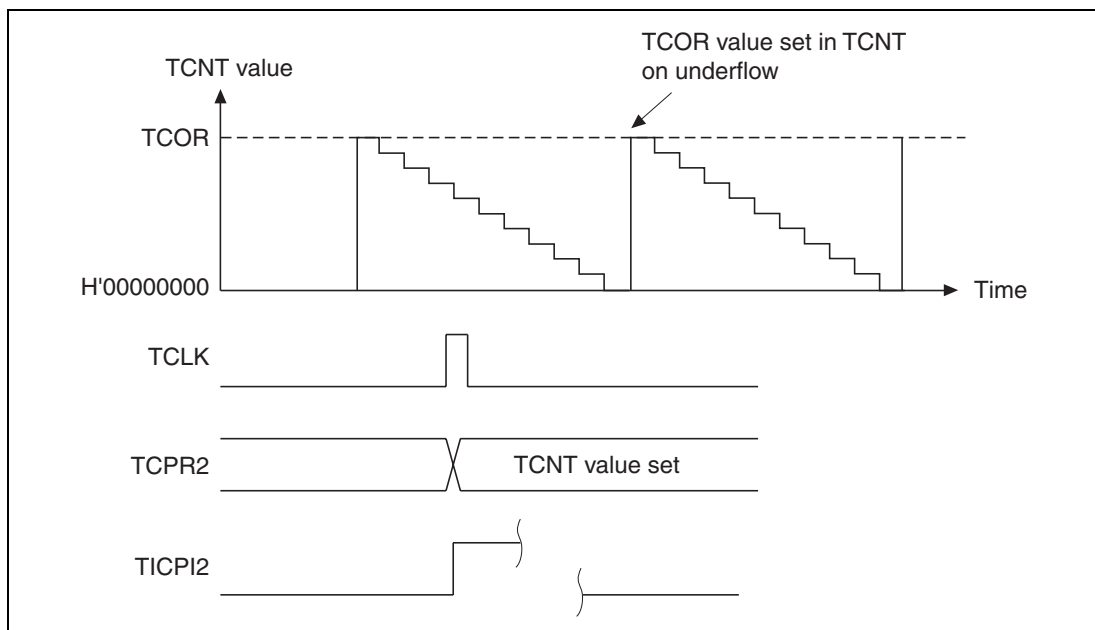


Figure 12.7 Operation Timing when Using Input Capture Function

12.4 Interrupts

There are six TMU interrupt sources, comprising underflow interrupts and the input capture interrupt (when the input capture function is used). Underflow interrupts are generated on channels 0 to 4, and input capture interrupts on channel 2 only.

An underflow interrupt request is generated (on an individual channel basis) when $TCR.UNF = 1$ and the channel's interrupt enable bit is 1.

When the input capture function is used and an input capture request is generated, an interrupt is requested if the input capture input flag (ICPF) in TCR2 is 1 and the input capture control bits (ICPE1, ICPE0) in TCR2 are 11.

The TMU interrupt sources are summarized in table 12.3.

Table 12.3 TMU Interrupt Sources

| Channel | Interrupt Source | Description |
|---------|------------------|---------------------------|
| 0 | TUNIO | Underflow interrupt 0 |
| 1 | TUNI1 | Underflow interrupt 1 |
| 2 | TUNI2 | Underflow interrupt 2 |
| | TICPI2 | Input capture interrupt 2 |
| 3 | TUNI3 | Underflow interrupt 3 |
| 4 | TUNI4 | Underflow interrupt 4 |

12.5 Usage Notes

12.5.1 Register Writes

When performing a TMU register write, timer count operation must be stopped by clearing the start bit (STR0–STR4) for the relevant channel in the timer start register (TSTR, TSTR2).

Note that the timer start register (TSTR, TSTR2) can be written to, and the underflow flag (UNF) and input capture flag (ICPF) of the timer control registers (TRCR0 to TCR4) can be cleared while the count is in progress. When the flags (UNF and ICPF) are cleared while the count is in progress, make sure not to change the values of bits other than those being cleared.

12.5.2 TCNT Register Reads

When performing a TCNT register read, processing for synchronization with the timer count operation is performed. If a timer count operation and register read processing are performed simultaneously, the TCNT counter value prior to the count-down operation is read by means of the synchronization processing.

12.5.3 Resetting the RTC Frequency Divider

When the on-chip RTC output clock is selected as the count clock, the RTC frequency divider should be reset.

12.5.4 External Clock Frequency

Ensure that the external clock frequency for any channel does not exceed $P_{ck}/8$.

Section 13 Bus State Controller (BSC)

13.1 Overview

The functions of the bus state controller (BSC) include division of the external memory space, and output of control signals in accordance with various types of memory and bus interface specifications. The BSC functions allow DRAM, synchronous DRAM, SRAM, ROM, etc., to be connected to this LSI and also support the PCMCIA interface protocol, enabling system design to be simplified and data transfers to be carried out at high speed by a compact system.

13.1.1 Features

The BSC has the following features:

- External memory space is managed as 7 independent areas
 - Maximum 64 Mbytes for each of areas 0 to 6
 - Bus width of each area can be set in a register (except area 0, which uses an external pin setting)
 - Wait state insertion by $\overline{\text{RDY}}$ pin
 - Wait state insertion can be controlled by program
 - Specification of types of memory connectable to each area
 - Output the control signals of memory to each area
 - Automatic wait cycle insertion to prevent data bus collisions in case of consecutive memory accesses to different areas, or a read access followed by a write access to the same area
 - Write strobe setup time and hold time periods can be inserted in a write cycle to enable connection to low-speed memory
- SRAM interface
 - Wait state insertion can be controlled by program
 - Wait state insertion by $\overline{\text{RDY}}$ pin
 - Connectable areas: 0 to 6
 - Settable bus widths: 32, 16, 8
- DRAM interface
 - Row address/column address multiplexing according to DRAM capacity
 - Burst operation (fast page mode, EDO mode)
 - CAS-before-RAS refresh and self-refresh
 - 4-CAS byte control for power-down operation

- DRAM control signal timing can be controlled by register settings
- Consecutive accesses to the same row address
 - Connectable area: 3
 - Settable bus widths: 32, 16
- Synchronous DRAM interface
 - Row address/column address multiplexing according to synchronous DRAM capacity
 - Burst operation
 - Auto-refresh and self-refresh
 - Synchronous DRAM control signal timing can be controlled by register settings
 - Consecutive accesses to the same row address
 - Connectable areas: 2, 3
 - Settable bus widths: 32
- Burst ROM interface
 - Wait state insertion can be controlled by program
 - Burst operation, executing the number of transfers set in a register
 - Connectable areas: 0, 5, 6
 - Settable bus widths: 32, 16, 8
- MPX interface
 - Address/data multiplexing
 - Connectable areas: 0 to 6
 - Settable bus widths: 32
- Byte control SRAM interface
 - SRAM interface with byte control
 - Connectable areas: 1, 4
 - Settable bus widths: 32, 16
- PCMCIA interface
 - Wait state insertion can be controlled by program
 - Bus sizing function for I/O bus width
- Fine refreshing control
 - Supports refresh operation immediately after self-refresh operation in low-power DRAM by means of refresh counter overflow interrupt function
- Refresh counter can be used as interval timer
 - Interrupt request generated by compare-match
 - Interrupt request generated by refresh counter overflow

13.1.2 Block Diagram

Figure 13.1 shows a block diagram of the BSC.

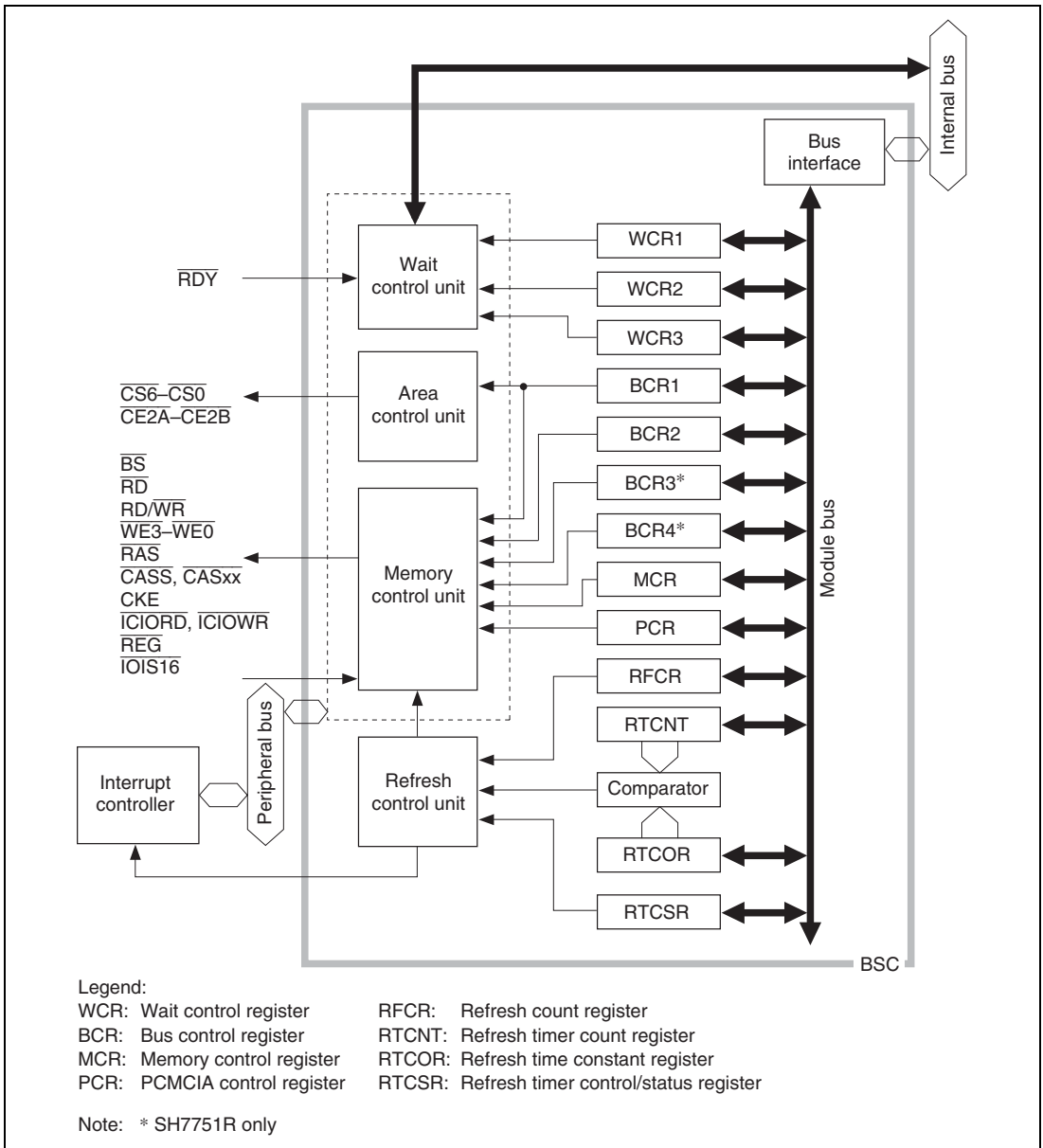


Figure 13.1 Block Diagram of BSC

13.1.3 Pin Configuration

Table 13.1 shows the BSC pin configuration.

Table 13.1 BSC Pins

| Name | Signals | I/O | Description |
|--|---|-----|---|
| Address bus | A25–A0 | O | Address output |
| Data bus | D31–D0 | I/O | Data input/output |
| Bus cycle start | \overline{BS} | O | Signal that indicates the start of a bus cycle When setting synchronous DRAM interface or MPX interface: asserted once for a burst transfer For other burst transfers: asserted each data cycle |
| Chip select 6–0 | $\overline{CS6}$ – $\overline{CS0}$ | O | Chip select signals that indicate the area being accessed $\overline{CS5}$ and $\overline{CS6}$ are also used as PCMCIA $\overline{CE1A}$ and $\overline{CE1B}$ |
| Read/write | RD/ \overline{WR} | O | Data bus input/output direction designation signal Also used as the DRAM/synchronous DRAM/PCMCIA interface write designation signal |
| Row address strobe | \overline{RAS} | O | \overline{RAS} signal when setting DRAM/synchronous DRAM interface |
| Read/column address strobe/ cycle frame | $\overline{RD/CASS/}$ \overline{FRAME} | O | Strobe signal that indicates a read cycle When setting synchronous DRAM interface: \overline{CAS} signal When setting MPX interface: \overline{FRAME} signal |
| Data enable 0 | $\overline{WE0/REG}$ | O | When setting PCMCIA interface: \overline{REG} signal When setting SRAM interface: write strobe signal for D7–D0 |
| Data enable 1 | $\overline{WE1}$ | O | When setting PCMCIA interface: write strobe signal When setting SRAM interface: write strobe signal for D15–D8 |
| Data enable 2 | $\overline{WE2/ICIORD}$ | O | When setting PCMCIA interface: \overline{ICIORD} signal When setting SRAM interface: write strobe signal for D23–D16 |
| Data enable 3 | $\overline{WE3/ICIOWR}$ | O | When setting PCMCIA interface: \overline{ICIOWR} signal When setting SRAM interface: write strobe signal for D31–D24 |

| Name | Signals | I/O | Description |
|---|--|-----|--|
| Column address strobe 0 | $\overline{\text{CAS0/DQM0}}$ | O | When setting DRAM interface: $\overline{\text{CAS}}$ signal for D7–D0 When setting synchronous DRAM interface: selection signal for D7–D0 |
| Column address strobe 1 | $\overline{\text{CAS1/DQM1}}$ | O | When setting DRAM interface: $\overline{\text{CAS}}$ signal for D15–D8 When setting synchronous DRAM interface: selection signal for D15–D8 |
| Column address strobe 2 | $\overline{\text{CAS2/DQM2}}$ | O | When setting DRAM interface: $\overline{\text{CAS}}$ signal for D23–D16 When setting synchronous DRAM interface: selection signal for D23–D16 |
| Column address strobe 3 | $\overline{\text{CAS3/DQM3}}$ | O | When setting DRAM interface: $\overline{\text{CAS}}$ signal for D31–D24 When setting synchronous DRAM interface: selection signal for D31–D24 |
| Ready | $\overline{\text{RDY}}$ | I | Wait state request signal |
| Area 0 MPX interface specification/ 16-bit I/O | $\overline{\text{MD6/IOIS16}}$ | I | In power-on reset: Designates area 0 bus as MPX interface (1: SRAM, 0: MPX) When setting PCMCIA interface: 16-bit I/O designation signal. Valid only in little-endian mode. |
| Clock enable | $\overline{\text{CKE}}$ | O | Synchronous DRAM clock enable control signal |
| Bus release request | $\overline{\text{BREQ/BSACK}}$ | I | Bus release request signal/bus acknowledge signal |
| Bus use permission | $\overline{\text{BACK/BSREQ}}$ | O | Bus use permission signal/bus request |
| Area 0 bus width/PCMCIA card select | $\overline{\text{MD3/CE2A}}^{*1}$ $\overline{\text{MD4/CE2B}}^{*2}$ | I/O | In power-on reset: area 0 bus width specification signal When using PCMCIA: $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$ |
| Endian switchover | $\overline{\text{MD5}}$ | I | Endian specification in a power-on reset |
| Master/slave switchover | $\overline{\text{MD7/CTS2}}$ | I/O | Indicates master/slave status in a power-on reset Serial interface $\overline{\text{CTS2}}$ |
| DMAC0 acknowledge signal | $\overline{\text{DACK0}}$ | O | DMAC channel 0 data acknowledge |
| DMAC1 acknowledge signal | $\overline{\text{DACK1}}$ | O | DMAC channel 1 data acknowledge |

- Notes: 1. MD3/ $\overline{\text{CE2A}}$ input/output switching is performed by BCR1.A56PCM. Output is selected when BCR1.A56PCM = 1.
2. MD4/ $\overline{\text{CE2B}}$ input/output switching is performed by BCR1.A56PCM. Output is selected when BCR1.A56PCM = 1.

13.1.4 Register Configuration

The BSC has the 11 registers shown in table 13.2. In addition, the synchronous DRAM mode register incorporated in synchronous DRAM can also be accessed as this LSI register. The functions of these registers include control of interfaces to various types of memory, wait states, and refreshing.

Table 13.2 BSC Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size | |
|---------------------------------------|--------------|-------|---------------|-------------|---------------------------|-------------|---|
| Bus control register 1 | BCR1 | R/W | H'0000 0000 | H'FF80 0000 | H'1F80 0000 | 32 | |
| Bus control register 2 | BCR2 | R/W | H'3FFC | H'FF80 0004 | H'1F80 0004 | 16 | |
| Bus control register 3* ² | BCR3 | R/W | H'0001 | H'FF80 0050 | H'1F80 0050 | 16 | |
| Bus control register 4* ² | BCR4 | R/W | H'0000 0000 | H'FE0A 00F0 | H'1E0A 00F0 | 32 | |
| Wait state control register 1 | WCR1 | R/W | H'7777 7777 | H'FF80 0008 | H'1F80 0008 | 32 | |
| Wait state control register 2 | WCR2 | R/W | H'FFFE EFFF | H'FF80 000C | H'1F80 000C | 32 | |
| Wait state control register 3 | WCR3 | R/W | H'0777 7777 | H'FF80 0010 | H'1F80 0010 | 32 | |
| Memory control register | MCR | R/W | H'0000 0000 | H'FF80 0014 | H'1F80 0014 | 32 | |
| PCMCIA control register | PCR | R/W | H'0000 | H'FF80 0018 | H'1F80 0018 | 16 | |
| Refresh timer control/status register | RTCSR | R/W | H'0000 | H'FF80 001C | H'1F80 001C | 16 | |
| Refresh timer counter | RTCNT | R/W | H'0000 | H'FF80 0020 | H'1F80 0020 | 16 | |
| Refresh time constant counter | RTCOR | R/W | H'0000 | H'FF80 0024 | H'1F80 0024 | 16 | |
| Refresh count register | RFCR | R/W | H'0000 | H'FF80 0028 | H'1F80 0028 | 16 | |
| Synchronous DRAM mode registers | For area 2 | SDMR2 | W | — | H'FF90 xxxx* ¹ | H'1F90 xxxx | 8 |
| | For area 3 | SDMR3 | | | H'FF94 xxxx* ¹ | H'1F94 xxxx | |

- Notes: 1. For details, see section 13.2.10, Synchronous DRAM Mode Register (SDMR).
2. SH7751R only

13.1.5 Overview of Areas

Space Divisions: The architecture of this LSI provides a 32-bit virtual address space. The virtual address space is divided into five areas according to the upper address value. External memory space comprises a 29-bit address space, divided into eight areas.

The virtual address can be allocated to any external address by means of the memory management unit (MMU). Details are given in section 3, Memory Management Unit (MMU). This section describes the areas into which the external address is divided.

With this LSI, various kinds of memory or PC cards can be connected to the seven areas of external address as shown in table 13.3, and chip select signals ($\overline{CS0}$ – $\overline{CS6}$, $\overline{CE2A}$, $\overline{CE2B}$) are output for each of these areas. $\overline{CS0}$ is asserted when accessing area 0, and $\overline{CS6}$ when accessing area 6. When DRAM or synchronous DRAM is connected to area 2 or 3, signals such as \overline{RAS} , \overline{CAS} , $\overline{RD}/\overline{WR}$, and \overline{DQM} are also asserted. When the PCMCIA interface is selected for area 5 or 6, $\overline{CE2A}$, $\overline{CE2B}$ is asserted in addition to $\overline{CS5}$, $\overline{CS6}$ for the byte to be accessed.

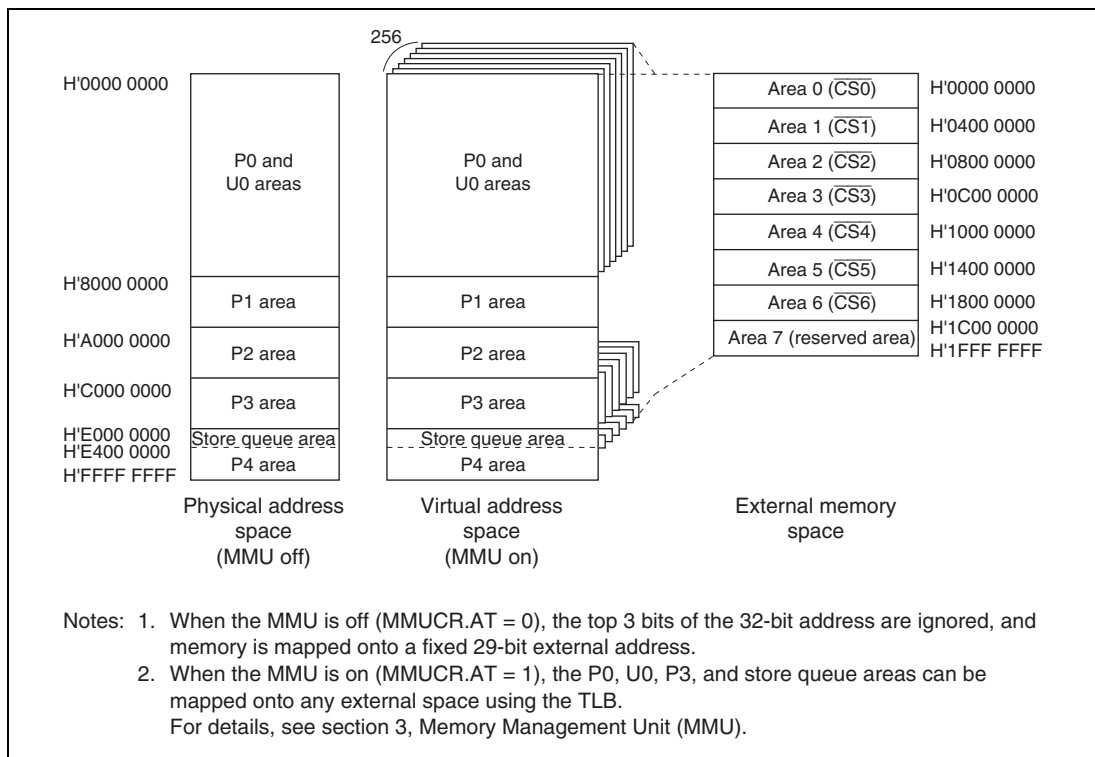


Figure 13.2 Correspondence between Virtual Address Space and External Memory Space

Table 13.3 External Memory Space Map

| Area | External Addresses | Size | Connectable Memory | Settable Bus Widths | Access Size |
|-----------------|---------------------------|-----------|--------------------|--------------------------------------|------------------------|
| 0 | H'00000000– H'03FFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ¹ | 8, 16, 32, |
| | | | Burst ROM | 8, 16, 32* ¹ | 64* ⁶ bits, |
| | | | MPX | 32* ¹ | 32 bytes |
| 1 | H'04000000– H'07FFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ² | 8, 16, 32, |
| | | | MPX | 32* ² | 64* ⁶ bits, |
| | | | Byte control SRAM | 16, 32* ² | 32 bytes |
| 2 | H'08000000– H'0BFFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ² | 8, 16, 32, |
| | | | Synchronous DRAM | 32* ² ,* ³ | 64* ⁶ bits, |
| | | | MPX | 32* ² | 32 bytes |
| 3 | H'0C000000– H'0FFFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ² | 8, 16, 32, |
| | | | Synchronous DRAM | 32* ² ,* ³ | 64* ⁶ bits, |
| | | | DRAM | 16, 32* ² ,* ³ | 32 bytes |
| | | | MPX | 32* ² | |
| 4 | H'10000000– H'13FFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ² | 8, 16, 32, |
| | | | MPX | 32* ² | 64* ⁶ bits, |
| | | | Byte control RAM | 16, 32* ² | 32 bytes |
| 5 | H'14000000– H'17FFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ² | 8, 16, 32, |
| | | | MPX | 32* ² | 64* ⁶ bits, |
| | | | Burst ROM | 8, 16, 32* ² | 32 bytes |
| | | | PCMCIA | 8, 16* ² ,* ⁴ | |
| 6 | H'18000000– H'1BFFFFFF | 64 Mbytes | SRAM | 8, 16, 32* ² | 8, 16, 32, |
| | | | MPX | 32* ² | 64* ⁶ bits, |
| | | | Burst ROM | 8, 16, 32* ² | 32 bytes |
| | | | PCMCIA | 8, 16* ² ,* ⁴ | |
| 7* ⁵ | H'1C000000– H'1FFFFFFF | 64 Mbytes | — | — | |

Notes: 1. Memory bus width specified by external pins

2. Memory bus width specified by register

3. With synchronous DRAM interface, bus width is 32 bits only
With DRAM interface, bus width is 16 or 32 bits only

4. With PCMCIA interface, bus width is 8 or 16 bits only

5. Do not access a reserved area, as operation cannot be guaranteed in this case

6. A 64-bit access size applies only to transfer by the DMAC (CHCRn.TS = 000).
In the case of access to external memory by means of FMOV (FPSCR.SZ = 1), two 32-bit access size transfers are performed.

| | | |
|--------------------|------------------------------------|--|
| Area 0: H'00000000 | SRAM/burst ROM/MPX | |
| Area 1: H'04000000 | SRAM/MPX/byte control SRAM | |
| Area 2: H'08000000 | SRAM/synchronous DRAM/MPX | |
| Area 3: H'0C000000 | SRAM/synchronous DRAM/DRAM/ MPX | |
| Area 4: H'10000000 | SRAM/MPX/byte control SRAM | |
| Area 5: H'14000000 | SRAM/burst ROM/PCMCIA/MPX | } The PCMCIA interface is for memory and I/O card use |
| Area 6: H'18000000 | SRAM/burst ROM/PCMCIA/MPX | |

Figure 13.3 External Memory Space Allocation

Memory Bus Width: In this LSI, the memory bus width can be set independently for each space. For area 0, a bus size of 8, 16, or 32 bits can be selected in a power-on reset by means of the RESET pin, using external pins. The relationship between the external pins (MD4 and MD3) and the bus width in a power-on reset is shown below.

| MD4 | MD3 | Bus Width |
|-----|-----|-----------|
| 0 | 0 | Reserved |
| | 1 | 8 bits |
| 1 | 0 | 16 bits |
| | 1 | 32 bits |

When SRAM interface or ROM is used in areas 1 to 6, a bus width of 8, 16, or 32 bits can be selected with bus control register 2 (BCR2). When burst ROM is used, a bus width of 8, 16, or 32 bits can be selected. When byte control SRAM interface is used, a bus width of 16, or 32 bits can be selected. When the MPX interface is used, a bus width of 32 bit can be set. When the DRAM

interface is used, a bus width of 16, or 32 bits can be selected with the memory control register (MCR). For the synchronous DRAM interface, set a bus width of 32 bit in the MCR register.

When using the PCMCIA interface, set a bus width of 8 or 16 bits. For details, see section 13.3.7, PCMCIA Interface.

For details, see section 13.2.2, Bus Control Register 2 (BCR2), and section 13.2.8, Memory Control Register (MCR).

The area 7 address range, H'1C000000 to H'1FFFFFFF, is a reserved space and must not be used.

13.1.6 PCMCIA Support

This LSI supports PCMCIA interface specifications for external memory space areas 5 and 6.

The interfaces supported are the IC memory card interface and I/O card interface stipulated in JEIDA specifications version 4.2 (PCMCIA2.1).

External memory space areas 5 and 6 support both the IC memory card interface and the I/O card interface.

The PCMCIA interface is supported only in little-endian mode.

Table 13.4 PCMCIA Interface Features

| Item | Features |
|---------------------------|---|
| Access | Random access |
| Data bus | 8/16 bits |
| Memory type | Mask ROM, OTPROM, EPROM, EEPROM, flash memory, SRAM |
| Common memory capacity | Max. 64 Mbytes |
| Attribute memory capacity | Max. 64 Mbytes |
| Others | Dynamic bus sizing for I/O bus width, access to PCMCIA interface from address translation areas |

Table 13.5 PCMCIA Support Interfaces

| Pin | IC Memory Card Interface | | | I/O Card Interface | | | Corresponding LSI Pin |
|-----|--------------------------|-----|--------------------------|---------------------|-----|-------------------------------------|--------------------------------------|
| | Signal Name | I/O | Function | Signal Name | I/O | Function | |
| 1 | GND | | Ground | GND | | Ground | — |
| 2 | D3 | I/O | Data | D3 | I/O | Data | D3 |
| 3 | D4 | I/O | Data | D4 | I/O | Data | D4 |
| 4 | D5 | I/O | Data | D5 | I/O | Data | D5 |
| 5 | D6 | I/O | Data | D6 | I/O | Data | D6 |
| 6 | D7 | I/O | Data | D7 | I/O | Data | D7 |
| 7 | $\overline{CE1}$ | I | Card enable | $\overline{CE1}$ | I | Card enable | $\overline{CS5}$ or $\overline{CS6}$ |
| 8 | A10 | I | Address | A10 | I | Address | A10 |
| 9 | \overline{OE} | I | Output enable | \overline{OE} | I | Output enable | \overline{RD} |
| 10 | A11 | I | Address | A11 | I | Address | A11 |
| 11 | A9 | I | Address | A9 | I | Address | A9 |
| 12 | A8 | I | Address | A8 | I | Address | A8 |
| 13 | A13 | I | Address | A13 | I | Address | A13 |
| 14 | A14 | I | Address | A14 | I | Address | A14 |
| 15 | $\overline{WE/PGM}$ | I | Write enable | $\overline{WE/PGM}$ | I | Write enable | $\overline{WE1}$ |
| 16 | $\overline{RDY/BSY}$ | O | Ready/busy | \overline{IREQ} | O | Interrupt request | Sensed on port |
| 17 | VCC | | Operating power supply | VCC | | Operating power supply | — |
| 18 | VPP1 | | Programming power supply | VPP1 | | Programming/peripheral power supply | — |
| 19 | A16 | I | Address | A16 | I | Address | A16 |
| 20 | A15 | I | Address | A15 | I | Address | A15 |
| 21 | A12 | I | Address | A12 | I | Address | A12 |
| 22 | A7 | I | Address | A7 | I | Address | A7 |
| 23 | A6 | I | Address | A6 | I | Address | A6 |
| 24 | A5 | I | Address | A5 | I | Address | A5 |
| 25 | A4 | I | Address | A4 | I | Address | A4 |
| 26 | A3 | I | Address | A3 | I | Address | A3 |
| 27 | A2 | I | Address | A2 | I | Address | A2 |

| Pin | IC Memory Card Interface | | | I/O Card Interface | | | Corresponding LSI Pin |
|-----|--------------------------|-----|--------------------------|----------------------------|-----|-------------------------------------|--|
| | Signal Name | I/O | Function | Signal Name | I/O | Function | |
| 28 | A1 | I | Address | A1 | I | Address | A1 |
| 29 | A0 | I | Address | A0 | I | Address | A0 |
| 30 | D0 | I/O | Data | D0 | I/O | Data | D0 |
| 31 | D1 | I/O | Data | D1 | I/O | Data | D1 |
| 32 | D2 | I/O | Data | D2 | I/O | Data | D2 |
| 33 | WP* | O | Write protect | $\overline{\text{IOIS16}}$ | O | 16-bit I/O port | $\overline{\text{IOIS16}}$ |
| 34 | GND | | Ground | GND | | Ground | — |
| 35 | GND | | Ground | GND | | Ground | — |
| 36 | $\overline{\text{CD1}}$ | O | Card detection | $\overline{\text{CD1}}$ | O | Card detection | Sensed on port |
| 37 | D11 | I/O | Data | D11 | I/O | Data | D11 |
| 38 | D12 | I/O | Data | D12 | I/O | Data | D12 |
| 39 | D13 | I/O | Data | D13 | I/O | Data | D13 |
| 40 | D14 | I/O | Data | D14 | I/O | Data | D14 |
| 41 | D15 | I/O | Data | D15 | I/O | Data | D15 |
| 42 | $\overline{\text{CE2}}$ | I | Card enable | $\overline{\text{CE2}}$ | I | Card enable | $\overline{\text{CE2A}}$ or $\overline{\text{CE2B}}$ |
| 43 | RFSH | I | Refresh request | RFSH | I | Refresh request | Output from port |
| 44 | RFU | | Reserved | $\overline{\text{IORD}}$ | I | I/O read | $\overline{\text{ICIORD}}$ |
| 45 | RFU | | Reserved | $\overline{\text{IOWR}}$ | I | I/O write | $\overline{\text{ICIOWR}}$ |
| 46 | A17 | I | Address | A17 | I | Address | A17 |
| 47 | A18 | I | Address | A18 | I | Address | A18 |
| 48 | A19 | I | Address | A19 | I | Address | A19 |
| 49 | A20 | I | Address | A20 | I | Address | A20 |
| 50 | A21 | I | Address | A21 | I | Address | A21 |
| 51 | VCC | | Power supply | VCC | | Power supply | — |
| 52 | VPP2 | | Programming power supply | VPP2 | | Programming/peripheral power supply | — |
| 53 | A22 | I | Address | A22 | I | Address | A22 |
| 54 | A23 | I | Address | A23 | I | Address | A23 |
| 55 | A24 | I | Address | A24 | I | Address | A24 |
| 56 | A25 | I | Address | A25 | I | Address | A25 |

| Pin | IC Memory Card Interface | | | I/O Card Interface | | | Corresponding LSI Pin |
|-----|--------------------------|-----|-------------------------------|----------------------------|-----|-------------------------------|------------------------------|
| | Signal Name | I/O | Function | Signal Name | I/O | Function | |
| 57 | RFU | | Reserved | RFU | | Reserved | — |
| 58 | RESET | I | Reset | RESET | I | Reset | Output from port |
| 59 | $\overline{\text{WAIT}}$ | O | Wait request | $\overline{\text{WAIT}}$ | O | Wait request | $\overline{\text{RDY}}^{*2}$ |
| 60 | RFU | | Reserved | $\overline{\text{INPACK}}$ | O | Input acknowledge | — |
| 61 | $\overline{\text{REG}}$ | I | Attribute memory space select | $\overline{\text{REG}}$ | I | Attribute memory space select | $\overline{\text{REG}}$ |
| 62 | BVD2 | O | Battery voltage detection | $\overline{\text{SPKR}}$ | O | Digital speech signal | Sensed on port |
| 63 | BVD1 | O | Battery voltage detection | $\overline{\text{STSCHG}}$ | O | Card status change | Sensed on port |
| 64 | D8 | I/O | Data | D8 | I/O | Data | D8 |
| 65 | D9 | I/O | Data | D9 | I/O | Data | D9 |
| 66 | D10 | I/O | Data | D10 | I/O | Data | D10 |
| 67 | $\overline{\text{CD2}}$ | O | Card detection | $\overline{\text{CD2}}$ | O | Card detection | Sensed on port |
| 68 | GND | | Ground | GND | | Ground | — |

Notes: 1. $\overline{\text{WP}}$ is not supported.

2. Input an external wait request with correct polarity.

13.2 Register Descriptions

13.2.1 Bus Control Register 1 (BCR1)

Bus control register 1 (BCR1) is a 32-bit readable/writable register that specifies the function, bus cycle status, etc., of each area.

BCR1 is initialized to H'00000000 by a power-on reset, but is not initialized by a manual reset or in standby mode. External memory space other than area 0 should not be accessed until register initialization is completed.

| | | | | | | | | |
|----------------|--------|--------|-------|----|----|------|------|------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | ENDIAN | MASTER | A0MPX | — | — | DPUP | IPUP | OPUP |
| Initial value: | 0/1* | 0/1* | 0/1* | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|----|----|-------|-------|--------|----|--------|--------|
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | A1MBC | A4MBC | BREQEN | — | MEMMPX | DMABST |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R/W | R/W | R/W | R | R/W | R/W |

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | HIZMEM | HIZCNT | A0BST2 | A0BST1 | A0BST0 | A5BST2 | A5BST1 | A5BST0 |
| 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 |
| | A6BST2 | A6BST1 | A6BST0 | DRAMTP2 | DRAMTP1 | DRAMTP0 | — | A56PCM |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W |

Note: * These bits sample external pin values in a power-on reset by means of the $\overline{\text{RESET}}$ pin.

Bit 31—Endian Flag (ENDIAN): Samples the value of the endian specification external pin (MD5) in a power-on reset by means of the $\overline{\text{RESET}}$ pin. The endian mode of all spaces is determined by this bit. ENDIAN is a read-only bit.

| Bit 31: ENDIAN | Description |
|----------------|---|
| 0 | In a power-on reset, the endian setting external pin (MD5) is low, designating big-endian mode for this LSI |
| 1 | In a power-on reset, the endian setting external pin (MD5) is high, designating little-endian mode for this LSI |

Bit 30—Master/Slave Flag (MASTER): Samples the value of the master/slave specification external pin (MD7) in a power-on reset by means of the $\overline{\text{RESET}}$ pin. The master/slave status of all spaces is determined by this bit. MASTER is a read-only bit.

| Bit 30: MASTER | Description |
|----------------|--|
| 0 | In a power-on reset, the master/slave setting external pin (MD7) is high, designating master mode for this LSI |
| 1 | In a power-on reset, the master/slave setting external pin (MD7) is low, designating slave mode for this LSI |

Bit 29—Area 0 Memory Type (A0MPX): Samples the value of the area 0 memory type specification external pin (MD6) in a power-on reset by means of the $\overline{\text{RESET}}$ pin. The memory type of area 0 is determined by this bit. A0MPX is a read-only bit.

| Bit 29: A0MPX | Description |
|---------------|---|
| 0 | In a power-on reset, the external pin specifying the area 0 memory type (MD6) is high, designating the area 0 as SRAM interface |
| 1 | In a power-on reset, the external pin specifying the area 0 memory type (MD6) is low, designating the area 0 as MPX interface |

Bits 28, 27, 23, 22, 18, and 1—Reserved: These bits are always read as 0, and the write value should always be 0.

Bit 26—Data pin Pullup Resistor Control (DPUP): Controls the pullup resistance of the data pins (D31 to D0). It is initialized at a power-on reset. The pins are not pulled up when access is performed or when the bus is released, even if the ON setting is selected.

| Bit 26: DPUP | Description |
|--------------|--|
| 0 | Sets pullup resistance of data pins (D31 to D0) ON (Initial value) |
| 1 | Sets pullup resistance of data pins (D31 to D0) OFF |

Bit 25—Control Input Pin Pull-Up Resistor Control (IPUP): Specifies the pull-up resistor status for control input pins (NMI, $\overline{\text{IRL0}}\text{--}\overline{\text{IRL3}}$, $\overline{\text{BREQ}}$, MD6/IOIS16, SLEEP, RDY). IPUP is initialized by a power-on reset.

| Bit 25: IPUP | Description |
|--------------|--|
| 0 | Pull-up resistor is on for control input pins (NMI, $\overline{\text{IRL0}}\text{--}\overline{\text{IRL3}}$, $\overline{\text{BREQ}}$, MD6/IOIS16, SLEEP, RDY) (Initial value) |
| 1 | Pull-up resistor is off for control input pins (NMI, $\overline{\text{IRL0}}\text{--}\overline{\text{IRL3}}$, $\overline{\text{BREQ}}$, MD6/IOIS16, SLEEP, RDY) |

Bit 24—Control Output Pin Pull-Up Resistor Control (OPUP): Specifies the pull-up resistor status for control output pins (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RD}}$, $\overline{\text{WEn}}$, $\overline{\text{CASn}}$, RD/WR, RAS, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, MD5) when high-impedance. OPUP is initialized by a power-on reset.

| Bit 24: OPUP | Description |
|--------------|--|
| 0 | Pull-up resistor is on for control output pins (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RD}}$, $\overline{\text{WEn}}$, $\overline{\text{CASn}}$, RD/WR, RAS, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, MD5) (Initial value) |
| 1 | Pull-up resistor is off for control output pins (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RD}}$, $\overline{\text{WEn}}$, $\overline{\text{CASn}}$, RD/WR, RAS, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, MD5) |

Bit 21—Area 1 SRAM Byte Control Mode (A1MBC): MPX interface has priority when an MPX interface is set. This bit is initialized by a power-on reset.

| Bit 21: A1MBC | Description |
|---------------|---|
| 0 | Area 1 SRAM is set to normal mode (Initial value) |
| 1 | Area 1 SRAM is set to byte control mode |

Bit 20—Area 4 SRAM Byte Control Mode (A4MBC): MPX interface has priority when an MPX interface is set. This bit is initialized by a power-on reset.

| Bit 20: A4MBC | Description |
|---------------|---|
| 0 | Area 4 SRAM is set to normal mode (Initial value) |
| 1 | Area 4 SRAM is set to byte control mode |

Bit 19—BREQ Enable (BREQEN): Indicates whether external requests and bus requests from PCIC can be accepted. BREQEN is initialized to the external request and bus request from PCIC acceptance disabled state by a power-on reset. It is ignored in the case of a slave mode startup.

The bus request from the PCIC is always accepted in a slave mode start up.

| Bit 19: BREQEN | Description |
|----------------|---|
| 0 | External requests and bus requests from PCIC are not accepted (Initial value) |
| 1 | External requests and bus requests from PCIC are accepted |

Bit 17—Area 1 to 6 MPX Bus Specification (MEMMPX): Sets the MPX interface when areas 1 to 6 are set as SRAM interface (or burst ROM interface). MEMMPX is initialized by a power-on reset.

| Bit 17: MEMMPX | Description |
|----------------|--|
| 0 | SRAM interface (or burst ROM interface) is selected when areas 1 to 6 are set as SRAM interface (or burst ROM interface) (Initial value) |
| 1 | MPX interface is selected when areas 1 to 6 are set as SRAM interface (or burst ROM interface) |

Bit 16—DMAC Burst Mode Transfer Priority Setting (DMABST): Specifies the priority of burst mode transfers by the DMAC. When OFF, the priority is as follows: bus privilege released, refresh, DMAC, CPU. When ON, the bus privileges are released and refresh operations are not performed until the end of the DMAC's burst transfer. This bit is initialized at a power-on reset.

| Bit 16: DMABST | Description |
|----------------|---|
| 0 | DMAC burst mode transfer priority specification OFF (Initial value) |
| 1 | DMAC burst mode transfer priority specification ON |

Bit 15—High Impedance Control (HIZMEM): Specifies the state of address and other signals ($A[25:0]$, \overline{BS} , \overline{CSn} , RD/\overline{WR} , $\overline{CE2A}$, $\overline{CE2B}$) in standby mode.

| Bit 15: HIZMEM | Description |
|----------------|---|
| 0 | The $A[25:0]$, \overline{BS} , \overline{CSn} , RD/\overline{WR} , $\overline{CE2A}$, and $\overline{CE2B}$ signals go to high-impedance (Hi-Z) in standby mode and when the bus is released (Initial value) |
| 1 | The $A[25:0]$, \overline{BS} , \overline{CSn} , RD/\overline{WR} , $\overline{CE2A}$, and $\overline{CE2B}$ signals drive in standby mode |

Bit 14—High Impedance Control (HIZCNT): Specifies the state of the \overline{RAS} and \overline{CAS} signals in standby mode and when the bus is released.

| Bit 14: HIZCNT | Description |
|----------------|---|
| 0 | The \overline{RAS} , \overline{WEn} , $\overline{CASn/DQMn}$, and $\overline{RD/CASS/FRAME}$ signals go to high-impedance (Hi-Z) in standby mode and when the bus is released (Initial value) |
| 1 | The \overline{RAS} , \overline{WEn} , $\overline{CASn/DQMn}$, and $\overline{RD/CASS/FRAME}$ signals drive in standby mode and when the bus is released |

Bits 13 to 11—Area 0 Burst ROM Control (A0BST2–A0BST0): These bits specify whether burst ROM interface is used in area 0. When burst ROM interface is used, they also specify the number of accesses in a burst. If area 0 is an MPX interface area, these bits are ignored.

| Bit 13: A0BST2 | Bit 12: A0BST1 | Bit 11: A0BST0 | Description |
|----------------|----------------|----------------|--|
| 0 | 0 | 0 | Area 0 is accessed as SRAM interface (Initial value) |
| | | 1 | Area 0 is accessed as burst ROM interface (4 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width |
| | 1 | 0 | Area 0 is accessed as burst ROM interface (8 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width |
| | | 1 | Area 0 is accessed as burst ROM interface (16 consecutive accesses) Can only be used with 8- or 16-bit bus width. Do not specify for 32-bit bus width |
| 1 | 0 | 0 | Area 0 is accessed as burst ROM interface (32 consecutive accesses) Can only be used with 8-bit bus width |
| | | 1 | Reserved |
| | | 1 | Reserved |
| | 1 | 0 | Reserved |
| | | 1 | Reserved |

Bits 10 to 8—Area 5 Burst Enable (A5BST2–A5BST0): These bits specify whether burst ROM interface is used in area 5. When burst ROM interface is used, they also specify the number of accesses in a burst. If area 5 is an MPX interface area, these bits are ignored.

| Bit 10: A5BST2 | Bit 9: A5BST1 | Bit 8: A5BST0 | Description |
|----------------|---------------|---------------|--|
| 0 | 0 | 0 | Area 5 is accessed as SRAM interface (Initial value) |
| | | 1 | Area 5 is accessed as burst ROM interface (4 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width |
| | 1 | 0 | Area 5 is accessed as burst ROM interface (8 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width |
| | | 1 | Area 5 is accessed as burst ROM interface (16 consecutive accesses) Can only be used with 8- or 16-bit bus width. Do not specify for 32-bit bus width |
| 1 | 0 | 0 | Area 5 is accessed as burst ROM interface (32 consecutive accesses) Can only be used with 8-bit bus width |
| | | 1 | Reserved |
| | | 0 | Reserved |
| | 1 | 1 | Reserved |

Note: Clear to 0 when PCMCIA interface is set.

Bits 7 to 5—Area 6 Burst Enable (A6BST2–A6BST0): These bits specify whether burst ROM interface is used in area 6. When burst ROM is used, they also specify the number of accesses in a burst. If area 6 is an MPX interface area, these bits are ignored.

| Bit 7: A6BST2 | Bit 6: A6BST1 | Bit 5: A6BST0 | Description |
|---------------|---------------|---------------|--|
| 0 | 0 | 0 | Area 6 is accessed as SRAM interface (Initial value) |
| | | 1 | Area 6 is accessed as burst ROM interface (4 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width |
| | 1 | 0 | Area 6 is accessed as burst ROM interface (8 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width |
| | | 1 | Area 6 is accessed as burst ROM interface (16 consecutive accesses) Can only be used with 8- or 16-bit bus width. Do not specify for 32-bit bus width |
| 1 | 0 | 0 | Area 6 is accessed as burst ROM interface (32 consecutive accesses) Can only be used with 8-bit bus width |
| | | 1 | Reserved |
| | | | Reserved |
| | 1 | 0 | Reserved |
| | | 1 | Reserved |

Note: Clear to 0 when PCMCIA is used.

Bits 4 to 2—Area 2 and 3 Memory Type (DRAMTP2–DRAMTP0): These bits specify the type of memory connected to areas 2 and 3. ROM, SRAM, flash ROM, etc., can be connected as SRAM interface. DRAM and synchronous DRAM can also be directly connected.

| Bit 4: DRAMTP2 | Bit 3: DRAMTP1 | Bit 2: DRAMTP0 | Description |
|----------------|----------------|----------------|--|
| 0 | 0 | 0 | Areas 2 and 3 are accessed as SRAM interface or MPX interface* (Initial value) |
| | | 1 | Reserved (Cannot be set) |
| 1 | 1 | 0 | Area 2 is accessed as SRAM interface or MPX interface*, area 3 is synchronous DRAM interface |
| | | 1 | Areas 2 and 3 are accessed as synchronous DRAM interface |
| | 0 | 0 | Area 2 is accessed as SRAM interface or MPX interface*, area 3 is DRAM interface |
| 1 | 1 | 1 | Reserved (Cannot be set) |
| | | 0 | Reserved (Cannot be set) |
| | | 1 | Reserved (Cannot be set) |

Note: * Selection of SRAM interface or MPX interface is determined by the setting of the MEMMPX bit

Bit 0—Area 5 and 6 Bus Type (A56PCM): Specifies whether areas 5 and 6 are accessed as PCMCIA interface. The setting of these bits has priority over the MEMMPX and AnBST bit settings.

| Bit 0: A56PCM | Description |
|---------------|--|
| 0 | Areas 5 and 6 are accessed as SRAM interface (Initial value) |
| 1 | Areas 5 and 6 are accessed as PCMCIA interface* |

Note: * The MD3 pin is designated for output as the CE2A pin.
The MD4 pin is designated for output as the CE2B pin.

13.2.2 Bus Control Register 2 (BCR2)

Bus control register 2 (BCR2) is a 32-bit readable/writable register that specifies the bus width for each area, and whether a 16-bit port is used.

BCR2 is initialized to H'3FFC by a power-on reset, but is not initialized by a manual reset or in standby mode. External memory space other than area 0 should not be accessed until register initialization is completed.

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | A0SZ1 | A0SZ0 | A6SZ1 | A6SZ0 | A5SZ1 | A5SZ0 | A4SZ1 | A4SZ0 |
| Initial value: | 0/1* | 0/1* | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|---|--------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A3SZ1 | A3SZ0 | A2SZ1 | A2SZ0 | A1SZ1 | A0SZ0 | — | PORTEN |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | — | R/W |

Note: * These bits sample the values of the external pins that specify the area 0 bus size.

Bits 15 and 14—Area 0 Bus Width (A0SZ1, A0SZ0): These bits sample the external pins (MD3 and MD4) that specify the bus size in a power-on reset. They are read-only bits.

| Bit 15: MD4 | Bit 14: MD3 | Bus Width |
|-------------|-------------|-------------------------------|
| 0 | 0 | Reserved (Setting prohibited) |
| | 1 | 8 bits |
| 1 | 0 | 16 bits |
| | 1 | 32 bits |

Bits $2n + 1$, $2n$ —Area n (1 to 6) Bus Width Specification (AnSZ1, AnSZ0): These bits specify the bus width of area n ($n = 1$ to 6).

| (Bit 0): PORTEN | Bit $2n + 1$: AnSZ1 | Bit $2n$: AnSZ0 | Description |
|-----------------|----------------------|------------------|--------------------------------------|
| 0 | 0 | 0 | Reserved (Setting prohibited) |
| | | 1 | Bus width is 8 bits |
| | 1 | 0 | Bus width is 16 bits |
| | | 1 | Bus width is 32 bits (Initial value) |
| 1 | 0 | 0 | Reserved (Setting prohibited) |
| | | 1 | Bus width is 8 bits |
| | 1 | 0 | Bus width is 16 bits |
| | | 1 | Bus width is 32 bits |

Bit 1—Reserved: This bit is always read as 0, and should only be written with 0.

Bit 0—Port Function Enable (PORTEN): Specifies whether pins AD31 to AD0 are used as a 32-bit port. However, select PCI-disable mode when using this function.

| Bit 0: PORTEN | Description |
|---------------|--|
| 0 | AD31 to AD0 are not used as a port (Initial value) |
| 1 | AD31 to AD0 are used as a port |

13.2.3 Bus Control Register 3 (BCR3) (SH7751R Only)

Bus control register 3 (BCR3) is a 16-bit readable/writable register that specifies the selection of either the MPX interface or the SRAM interface and specifies the burst length when the synchronous DRAM interface is used.

BCR3 is initialized to H'0001 by a power-on reset, but is not initialized by a manual reset or in standby mode. No external memory space other than area 0 should be accessed before register initialization has been completed.

| | | | | | | | | |
|----------------|---------|-------|-------|----|----|----|---|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Bit name: | MEMMODE | A1MPX | A4MPX | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Bit name: | — | — | — | — | — | — | — | SDBL |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W: | R | R | R | R | R | R | R | R/W |

Bit 15—A1MPX/A4MPX Enable (MEMMODE): Determines whether or not the selection of either the MPX interface or the SRAM interface is by A1MPX and A4MPX rather than by MEMMPX.

| Bit 15: MEMMODE | Description |
|-----------------|---|
| 0 | MPX or SRAM interface is selected by MEMMPX (Initial value) |
| 1 | MPX or SRAM interface is selected by A1MPX and A4MPX |

Bits 14 and 13—MPX-Interface Specification for Area 1 and 4 (A1MPX, A4MPX): These bits specify the types of memory connected to areas 1 and 4. These settings are validated by MEMMODE.

| Bit 14: A1MPX | Description |
|----------------------|---|
| 0 | SRAM/byte control SRAM interface is selected for area 1 (Initial value) |
| 1 | MPX interface is selected for area 1 |

| Bit 13: A4MPX | Description |
|----------------------|---|
| 0 | SRAM/byte control SRAM interface is selected for area 4 (Initial value) |
| 1 | MPX interface is selected for area 4 |

Bit 0—Burst Length (SDBL): Sets the burst length when the synchronous DRAM interface is used. The burst-length setting is only valid when the bus width is 32 bits.

| Bit 0: SDBL | Description |
|--------------------|-----------------------------------|
| 0 | Burst length is 8 |
| 1 | Burst length is 4 (Initial value) |

13.2.4 Bus Control Register 4 (BCR4) (SH7751R Only)

Bus control register 4 (BCR4) is a register that enables asynchronous input for pins corresponding to individual bits.

The BCR4 register is a 32-bit readable/writable register. It is initialized to H'00000000 by a power-on reset, but is not initialized by a manual reset or in standby mode.

When asynchronous input is set ($ASYNCn = 1$), the sampling timing is one cycle earlier than when synchronous input is set ($ASYNCn = 0$)* (see figure 13.4)

The timings shown in this section and section 23, Electrical Characteristics, are all for the case where synchronous input is set ($ASYNCn = 0$).

Note: * With the synchronous input setting, ensure that setup and hold times are observed.

| | | | | | | | | |
|----------------|----|----|----|--------|--------|--------|--------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | ASYNC4 | ASYNC3 | ASYNC2 | ASYNC1 | ASYNC0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |

Bits 31 to 5—Reserved: These bits are always read as 0, and the write value should always be 0.

Bits 4 to 0—Asynchronous Input: These bits enable asynchronous input for the corresponding pins.

| Bit 4–0: ASYNCn | Description |
|-----------------|---|
| 0 | Corresponding pin is synchronous input with respect to CKIO (Initial value) |
| 1 | Asynchronous input with respect to CKIO is enabled for corresponding pin |

| Bit | |
|-----|---------------------|
| 4 | $\overline{IOIS16}$ |
| 3 | $\overline{DREQ1}$ |
| 2 | $\overline{DREQ0}$ |
| 1 | \overline{BREQ} |
| 0 | \overline{RDY} |

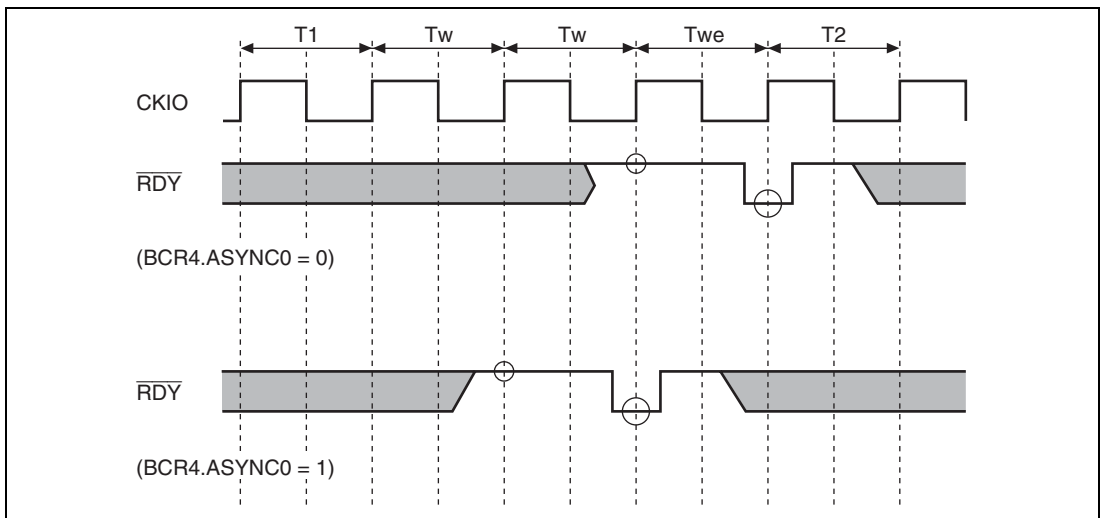


Figure 13.4 Example of \overline{RDY} Sampling Timing at which BCR4 Is Set
(Two Wait Cycles Are Inserted by WCR2)

13.2.5 Wait Control Register 1 (WCR1)

Wait control register 1 (WCR1) is a 32-bit readable/writable register that specifies the number of idle state insertion cycles for each area. With some kinds of memory, data bus drive does not go off immediately after the read signal from off-chip goes off. As a result, there is a possibility of a data bus collision when consecutive memory accesses are performed on memory in different areas, or when a memory write is performed immediately after a read. In this LSI, the number of idle cycles set in the WCR1 register are inserted automatically if there is a possibility of this kind of data bus collision.

WCR1 is initialized to H'77777777 by a power-on reset, but is not initialized by a manual reset or in standby mode.

| | | | | | | | | |
|----------------|----|--------|--------|--------|----|-------|-------|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | DMAIW2 | DMAIW1 | DMAIW0 | — | A6IW2 | A6IW1 | A6IW0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | A5IW2 | A5IW1 | A5IW0 | — | A4IW2 | A4IW1 | A4IW0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | A3IW2 | A3IW1 | A3IW0 | — | A2IW2 | A2IW1 | A2IW0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | A1IW2 | A1IW1 | A1IW0 | — | A0IW2 | A0IW1 | A0IW0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R | R/W | R/W | R/W | R | R/W | R/W | R/W |

Bits 31, 27, 23, 19, 15, 11, 7, and 3—Reserved: These bits are always read as 0, and the write value should always be 0.

Bits 30 to 28—DMAIW-DACK Device Inter-Cycle Idle Specification (DMAIW2–DMAIW0): These bits specify the number of idle cycles between bus cycles to be inserted when switching from a DACK device to another space, or from a read access to a write access on the same device. The DMAIW bits are valid only for DMA single address transfer; with DMA dual address transfer, inter-area idle cycles are inserted.

Bits 4n + 2 to 4n—Area n (6 to 0) Inter-Cycle Idle Specification (AnIW2–AnIW0): These bits specify the number of idle cycles between bus cycles to be inserted when switching from external memory space area n (n = 6 to 0) to another space, or from a read access to a write access in the same space.

| DMAIW2/AnIW2 | DMAIW1/AnIW1 | DMAIW0/AnIW0 | Inserted Idle Cycles |
|--------------|--------------|--------------|----------------------|
| 0 | 0 | 0 | 0 |
| | | 1 | 1 |
| | 1 | 0 | 2 |
| | | 1 | 3 |
| 1 | 0 | 0 | 6 |
| | | 1 | 9 |
| | 1 | 0 | 12 |
| | | 1 | 15 |

(Initial value)

Table 13.6 Idle Insertion between Accesses

| Preceding Cycle | Following Cycle | | | | | | | | | |
|--------------------------------|-----------------|-----|-------|-----------------|----------------|-----|-------|-----|--------------------|--------------------|
| | Same Area | | | | Different Area | | | | Same Area | Different Area |
| | Read | | Write | | Read | | Write | | MPX Address Output | MPX Address Output |
| | CPU | DMA | CPU | DMA | CPU | DMA | CPU | DMA | | |
| Read | | | M | M | M | M | M | M | M (1) | M (1) |
| Write | | | | | M | M | M | M | * ² | M |
| DMA read (memory → device) | | | M | M | M | M | M | M | — | M (1) |
| DMA write (device → memory) | D | D | D | D* ¹ | D | D | D | D | — | D (1) |

Notes: "DMA" in the table indicates DMA single-address transfer. DMA dual-address transfer is in accordance with the CPU.

M, D: Idle wait always inserted by WCR1

(M(1): Once cycle inserted in MPX access even if WCR1 is cleared to 0)

M: Idle cycles according to setting of AnIW2-AnIW0 (areas 0 to 6)

D: Idle cycles according to setting of DMAIW2-DMAIW0

1. Inserted when device is switched

2. On the MPX interface, a WCR1 idle wait may be inserted before an access (either read or write) to the same area after a write access. The specific conditions for idle wait insertion in accesses to the same area are shown below.

(a) Synchronous DRAM set to RAS down mode

(b) Synchronous DRAM accessed by on-chip DMAC

Apart from use under above conditions (a) and (b), an idle wait is also inserted between an MPX interface write access and a following access to the same area. Even under the above conditions, an idle wait may be inserted in a same-area access following an interface write access, depending on the synchronous DRAM pipeline access situation. An idle wait is not inserted when the WCR1 register setting is 0. The setting for the number of idle state cycles inserted after a power-on reset is the default value of 15 (the maximum value), so ensure that the optimum value is set.

When synchronous DRAM is used in RAS down mode, set bits DMAIW2-DMAIW0 to 000 and bits A3IW2-A3IW0 to 000.

13.2.6 Wait Control Register 2 (WCR2)

Wait control register 2 (WCR2) is a 32-bit readable/writable register that specifies the number of wait states to be inserted for each area. It also specifies the data access pitch when performing burst memory access. This enables low-speed memory to be connected without using external circuitry.

WCR2 is initialized to H'FFFFFFF by a power-on reset, but is not initialized by a manual reset or in standby mode.

| | | | | | | | | |
|----------------|------|------|------|------|------|------|------|------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | A6W2 | A6W1 | A6W0 | A6B2 | A6B1 | A6B0 | A5W2 | A5W1 |
| 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: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | A5W0 | A5B2 | A5B1 | A5B0 | A4W2 | A4W1 | A4W0 | — |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | A3W2 | A3W1 | A3W0 | — | A2W2 | A2W1 | A2W0 | A1W2 |
| Initial value: | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| R/W: | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A1W1 | A1W0 | A0W2 | A0W1 | A0W0 | A0B2 | A0B1 | A0B0 |
| 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 31 to 29—Area 6 Wait Control (A6W2–A6W0): These bits specify the number of wait states to be inserted for area 6. For the case where an MPX interface setting is made, see table 13.7.

| Bit 31: A6W2 | Bit 30: A6W1 | Bit 29: A6W0 | Description | |
|--------------|--------------|--------------|----------------------|-----------------------------|
| | | | First Cycle | |
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

Bits 28 to 26—Area 6 Burst Pitch (A6B2–A6B0): These bits specify the number of wait states to be inserted from the second data access onward at the time of setting the burst ROM in a burst transfer.

| Bit 28: A6B2 | Bit 27: A6B1 | Bit 26: A6B0 | Description | |
|--------------|--------------|--------------|---|-----------------------------|
| | | | Burst Cycle (Excluding First Cycle) | |
| | | | Wait States Inserted from Second Data Access Onward | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 4 | Enabled |
| | | 1 | 5 | Enabled |
| | 1 | 0 | 6 | Enabled |
| | | 1 | 7 (Initial value) | Enabled |

Bits 25 to 23—Area 5 Wait Control (A5W2–A5W0): These bits specify the number of wait states to be inserted for area 5. For the case where an MPX interface setting is made, see table 13.7.

| Bit 25: A5W2 | Bit 24: A5W1 | Bit 23: A5W0 | Description | |
|--------------|--------------|--------------|----------------------|-----------------------------|
| | | | First Cycle | |
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

Bits 22 to 20—Area 5 Burst Pitch (A5B2–A5B0): These bits specify the number of wait states to be inserted from the second data access onward at the time of setting the burst ROM in a burst transfer.

| Bit 22: A5B2 | Bit 21: A5B1 | Bit 20: A5B0 | Description | |
|--------------|--------------|--------------|---|-----------------------------|
| | | | Burst Cycle (Excluding First Cycle) | |
| | | | Wait States Inserted from Second Data Access Onward | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 4 | Enabled |
| | | 1 | 5 | Enabled |
| | 1 | 0 | 6 | Enabled |
| | | 1 | 7 (Initial value) | Enabled |

Bits 19 to 17—Area 4 Wait Control (A4W2–A4W0): These bits specify the number of wait states to be inserted for area 4. For the case where an MPX interface setting is made, see table 13.7.

| Bit 19: A4W2 | Bit 18: A4W1 | Bit 17: A4W0 | Description | |
|--------------|--------------|--------------|----------------------|-----------------------------|
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

Bits 16 and 12—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 15 to 13—Area 3 Wait Control (A3W2–A3W0): These bits specify the number of wait states to be inserted for area 3. External wait input is only enabled when the SRAM interface or MPX interface is used, and is ignored when DRAM or synchronous DRAM is used. For the case where an MPX interface setting is made, see table 13.7.

- When SRAM Interface is Set

| Bit 15: A3W2 | Bit 14: A3W1 | Bit 13: A3W0 | Description | |
|--------------|--------------|--------------|----------------------|-----------------------------|
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

- When DRAM or Synchronous DRAM Interface is Set*

Note: * External wait input is always ignored

| Bit 15: A3W2 | Bit 14: A3W1 | Bit 13: A3W0 | Description | |
|--------------|--------------|--------------|--|---|
| | | | DRAM $\overline{\text{CAS}}$ Assertion Width | Synchronous DRAM $\overline{\text{CAS}}$ Latency Cycles |
| 0 | 0 | 0 | 1 | Inhibited |
| | | 1 | 2 | 1* |
| | 1 | 0 | 3 | 2 |
| | | 1 | 4 | 3 |
| 1 | 0 | 0 | 7 | 4* |
| | | 1 | 10 | 5* |
| | 1 | 0 | 13 | Inhibited |
| | | 1 | 16 | Inhibited |

Note: * Inhibited in RAS down mode

Bits 11 to 9—Area 2 Wait Control (A2W2–A2W0): These bits specify the number of wait states to be inserted for area 2. External wait input is only enabled when the SRAM interface or MPX interface is used, and is ignored when synchronous DRAM is used. For the case where an MPX interface setting is made, see table 13.7.

- When SRAM Interface is Set

| Bit 11: A2W2 | Bit 10: A2W1 | Bit 9: A2W0 | Description | |
|--------------|--------------|-------------|----------------------|-----------------------------|
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

- When Synchronous DRAM Interface is Set*¹

| Bit 11: A2W2 | Bit 10: A2W1 | Bit 9: A2W0 | Description | |
|--------------|--------------|-------------|---|--|
| | | | Synchronous DRAM $\overline{\text{CAS}}$ Latency Cycles | |
| 0 | 0 | 0 | Inhibited | |
| | | 1 | 1* ² | |
| | 1 | 0 | 2 | |
| | | 1 | 3 | |
| 1 | 0 | 0 | 4* ² | |
| | | 1 | 5* ² | |
| | 1 | 0 | Inhibited | |
| | | 1 | Inhibited | |

Notes: 1. External wait input is always ignored

2. Inhibited in RAS down mode

Bits 8 to 6—Area 1 Wait Control (A1W2–A1W0): These bits specify the number of wait states to be inserted for area 1. For the case where an MPX interface setting is made, see table 13.7.

| Bit 8: A1W2 | Bit 7: A1W1 | Bit 6: A1W0 | Description | |
|-------------|-------------|-------------|----------------------|-----------------------------|
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

Bits 5 to 3—Area 0 Wait Control (A0W2 to A0W0): These bits specify the number of wait states to be inserted for area 0. For the case where an MPX interface setting is made, see table 13.7.

| Bit 5: A0W2 | Bit 4: A0W1 | Bit 3: A0W0 | Description | |
|-------------|-------------|-------------|----------------------|-----------------------------|
| | | | Inserted Wait States | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 6 | Enabled |
| | | 1 | 9 | Enabled |
| | 1 | 0 | 12 | Enabled |
| | | 1 | 15 (Initial value) | Enabled |

Bits 2 to 0—Area 0 Burst Pitch (A0B2–A0B0): These bits specify the number of wait states to be inserted from the second data access onward at the time of setting the burst ROM in a burst transfer.

| Bit 2: A0B2 | Bit 1: A0B1 | Bit 0: A0B0 | Description | |
|-------------|-------------|-------------|---|-----------------------------|
| | | | Wait States Inserted from Second Data Access Onward | $\overline{\text{RDY}}$ Pin |
| 0 | 0 | 0 | 0 | Ignored |
| | | 1 | 1 | Enabled |
| | 1 | 0 | 2 | Enabled |
| | | 1 | 3 | Enabled |
| 1 | 0 | 0 | 4 | Enabled |
| | | 1 | 5 | Enabled |
| | 1 | 0 | 6 | Enabled |
| | | 1 | 7 (Initial value) | Enabled |

Table 13.7 When MPX Interface Is Set (Areas 0 to 6)

| AnW2 | AnW1 | AnW0 | Description | | | | |
|------|------|------|----------------------|-------|--------------------|---------|---------|
| | | | Inserted Wait States | | | | RDY Pin |
| | | | 1st Data | | 2nd Data Onward | | |
| | | | Read | Write | | | |
| 0 | 0 | 0 | 1 | 0 | 0 | Enabled | |
| | | 1 | | 1 | | Enabled | |
| | 1 | 0 | 2 | 2 | | Enabled | |
| | | 1 | 3 | 3 | | Enabled | |
| 1 | 0 | 0 | 1 | 0 | 1 | Enabled | |
| | | 1 | | 1 | | Enabled | |
| | 1 | 0 | 2 | 2 | | Enabled | |
| | | 1 | 3 | 3 | | Enabled | |

Note: n = 6 to 0

13.2.7 Wait Control Register 3 (WCR3)

Wait control register 3 (WCR3) is a 32-bit readable/writable register that specifies the cycles inserted in the setup time from the address until assertion of the write strobe, and the data hold time from negation of the strobe, for each area. This enables low-speed memory to be connected without using external circuitry.

WCR3 is initialized to H'07777777 by a power-on reset, but is not initialized by a manual reset or in standby mode.

| | | | | | | | | |
|----------------|----|----|----|----|----|------|------|------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | A6S0 | A6H1 | A6H0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|----|------|------|------|--------|------|------|------|
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | A5S0 | A5H1 | A5H0 | A4RDH* | A4S0 | A4H1 | A4H0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R | R/W | R/W | R/W | R/W* | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|----|------|------|------|----|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Bit name: | — | A3S0 | A3H1 | A3H0 | — | A2S0 | A2H1 | A2H0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R | R/W | R/W | R/W | R | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|--------|------|------|------|---|------|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A1RDH* | A1S0 | A1H1 | A0H0 | — | A0S0 | A0H1 | A0H0 |
| Initial value: | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| R/W: | R/W* | R/W | R/W | R/W | R | R/W | R/W | R/W |

Note: * These bits can be set only in the SH7751R.

Bits 31 to 27, 23, 19*, 15, 11, 7*, and 3 (SH7751)

Bits 31 to 27, 23, 15, 11, and 3 (SH7751R)

Reserved: These bits are always read as 0, and should only be written with 0.

Note: * These bits can be set only in the SH7751R.

Bit 4n + 2—Area n (6 to 0) Write Strobe Setup Time (AnS0): Specifies the number of cycles inserted in the setup time from the address until assertion of the read/write strobe. Valid only for SRAM interface, byte control SRAM interface, and burst ROM interface:

| Bit 4n + 2: AnS0 | Waits Inserted in Setup |
|------------------|-------------------------|
| 0 | 0 |
| 1 | 1 (Initial value) |

Note: n = 6 to 0

Bits 4n + 1 and 4n—Area n (6 to 0) Data Hold Time (AnH1, AnH0): When writing, these bits specify the number of cycles to be inserted in the hold time from negation of the write strobe. When reading, they specify the number of cycles to be inserted in the hold time from the data sampling timing. Valid only for SRAM interface, byte control SRAM interface, and burst ROM interface:

| Bit 4n + 1: AnH1 | Bit 4n: AnH0 | Waits Inserted in Hold |
|------------------|--------------|------------------------|
| 0 | 0 | 0 |
| | 1 | 1 |
| 1 | 0 | 2 |
| | 1 | 3 (Initial value) |

Note: n = 6 to 0

Bits 4n + 3—Area n (4 or 1) Read-Strobe Negate Timing (AnRDH) (Setting Only Possible in the SH7751R): When reading, these bits specify the timing for the negation of read strobe. These bits should be cleared to 0 when a byte control SRAM setting is made. Valid only for the SRAM interface.

| Bit 4n + 3: AnRDH | Read-Strobe Negate Timing |
|-------------------|---|
| 0 | Read strobe negated after hold wait cycles specified by WCR3.AnH bits (Initial value) |
| 1 | Read strobe negated according to data sampling timing |

Note: n = 4 or 1

13.2.8 Memory Control Register (MCR)

The memory control register (MCR) is a 32-bit readable/writable register that specifies $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ timing and burst control for DRAM and synchronous DRAM (areas 2 and 3), address multiplexing, and refresh control. This enables DRAM and synchronous DRAM to be connected without using external circuitry.

MCR is initialized to H'00000000 by a power-on reset, but is not initialized by a manual reset or in standby mode. Bits RASD, MRSET, TRC2–0, TPC2–0, RCD1–0, TRWL2–0, TRAS2–0, BE, SZ1–0, AMXEXT, AMX2–0, and EDOMODE are written in the initialization following a power-on reset, and should not be modified subsequently. When writing to bits RFSH and RMODE, the same values should be written to the other bits so that they remain unchanged. When using DRAM or synchronous DRAM, areas 2 and 3 should not be accessed until register initialization is completed.

| | | | | | | | | |
|----------------|-------|--------|-------|-------|-------|-------|-------|-------------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | RASD | MRSET | TRC2 | TRC1 | TRC0 | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | TCAS | — | TPC2 | TPC1 | TPC0 | — | RCD1 | RCD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R | R/W | R/W | R/W | R | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | TRWL2 | TRWL1 | TRWL0 | TRAS2 | TRAS1 | TRAS0 | BE | SZ1 |
| 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 |
| | SZ0 | AMXEXT | AMX2 | AMX1 | AMX0 | RFSH | RMODE | EDO MODE |
| 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 31—RAS Down (RASD): Sets RAS down mode. When RAS down mode is used, set BE to 1. Do not set RAS down mode in slave mode or when areas 2 and 3 are both designated as synchronous DRAM interface.

| Bit 31: RASD | Description |
|--------------|-------------------------------------|
| 0 | Auto-precharge mode (Initial value) |
| 1 | RAS down mode |

Note: When synchronous DRAM is used in RAS down mode, set bits DMAIW2–DMAIW0 to 000 and bits A3IW2–A3IW0 to 000.

Bit 30—Mode Register Set (MRSET): Set when a synchronous DRAM mode register setting is used. See Power-On Sequence in section 13.3.5, Synchronous DRAM Interface.

| Bit 30: MRSET | Description |
|---------------|------------------------------------|
| 0 | All-bank precharge (Initial value) |
| 1 | Mode register setting |

Bits 26 to 24, 22, and 18—Reserved: These bits should only be written with 0.

Bits 29 to 27—RAS Precharge Time at End of Refresh (TRC2–TRC0)

(Synchronous DRAM: auto- and self-refresh both enabled, DRAM: auto- and self-refresh both enabled)

Note: For setting values and the period during which no command is issued, see 23.3.3, Bus Timing.

| Bit 29: TRC2 | Bit 28: TRC1 | Bit 27: TRC0 | RAS Precharge Time Immediately after Refresh |
|--------------|--------------|--------------|--|
| 0 | 0 | 0 | 0 (Initial value) |
| | | 1 | 3 |
| | 1 | 0 | 6 |
| | | 1 | 9 |
| 1 | 0 | 0 | 12 |
| | | 1 | 15 |
| | 1 | 0 | 18 |
| | | 1 | 21 |

Bit 23—CAS Negation Period (TCAS): This bit is valid only when DRAM interface is set.

| Bit 23: TCAS | CAS Negation Period |
|--------------|---------------------|
| 0 | 1 (Initial value) |
| 1 | 2 |

Bits 21 to 19—RAS Precharge Period (TPC2–TPC0): When the DRAM interface is selected, these bits specify the minimum number of cycles until $\overline{\text{RAS}}$ is asserted again after being negated. When the synchronous DRAM interface is selected, these bits specify the minimum number of cycles until the next bank active command after precharging.

Note: For setting values and the period during which no command is issued, see 23.3.3, Bus Timing.

| Bit 21: TPC2 | Bit 20: TPC1 | Bit 19: TPC0 | RAS Precharge Time | |
|--------------|--------------|--------------|--------------------|--------------------|
| | | | DRAM | Synchronous DRAM |
| 0 | 0 | 0 | 0 | 1* (Initial value) |
| | | 1 | 1 | 2 |
| | | 0 | 2 | 3 |
| | 1 | 0 | 3 | 4* |
| | | 1 | 4 | 5* |
| | | 0 | 5 | 6* |
| 1 | 0 | 0 | 6 | 7* |
| | | 1 | 7 | 8* |
| | 1 | 0 | 7 | 8* |
| | | 1 | 8 | 9* |

Note: * Inhibited in RAS down mode

Bits 17 and 16—RAS-CAS Delay (RCD1, RCD0): When the DRAM interface is set, these bits set the $\overline{\text{RAS}}\text{-}\overline{\text{CAS}}$ assertion delay time. When the synchronous DRAM interface is set, these bits set the bank active-read/write command delay time.

| Bit 17: RCD1 | Bit 16: RCD0 | Description | |
|--------------|--------------|-------------|----------------------------------|
| | | DRAM | Synchronous DRAM |
| 0 | 0 | 2 cycles | Reserved (Setting prohibited) |
| | 1 | 3 cycles | 2 cycles |
| 1 | 0 | 4 cycles | 3 cycles |
| | 1 | 5 cycles | 4 cycles* |

Note: * Inhibited in RAS down mode

Bits 15 to 13—Write Precharge Delay (TRWL2–TRWL0): These bits set the synchronous DRAM write precharge delay time. In auto-precharge mode, they specify the time until the next bank active command is issued after a write cycle. After a write cycle, the next active command is not issued for a period set by TPC[2:0] and TRWL[2:0] bits*. In RAS down mode, they specify the time until the next precharge command is issued. After a write cycle, the next precharge command is not issued for a period of TRWL. This setting is valid only when synchronous DRAM interface is set.

Note: * For setting values and the period during which no command is issued, see 23.3.3, Bus Timing.

| Bit 15: TRWL2 | Bit 14: TRWL1 | Bit 13: TRWL0 | Write Precharge ACT Delay Time | |
|---------------|---------------|---------------|--------------------------------|-------------------------------|
| 0 | 0 | 0 | 1 (Initial value) | |
| | | 1 | 2 | |
| | 1 | 0 | 3* | |
| | | 1 | 4* | |
| | 1 | 0 | 0 | 5* |
| | | | 1 | Reserved (Setting prohibited) |
| 1 | | 0 | Reserved (Setting prohibited) | |
| | | 1 | Reserved (Setting prohibited) | |

Note: * Inhibited in RAS down mode

Bits 12 to 10—CAS-Before-RAS Refresh $\overline{\text{RAS}}$ Assertion Period (TRAS2–TRAS0): When the DRAM interface is set, these bits set the $\overline{\text{RAS}}$ assertion period in CAS-before-RAS refreshing. When the synchronous DRAM interface is set, the bank active command is not issued for a period set by TPC[2:0] and TRAS[2:0] bits after an auto-refresh command is issued.

Note: For setting values and the period during which no command is issued, see 23.3.3, Bus Timing.

| Bit 12: TRAS2 | Bit 11: TRAS1 | Bit 10: TRAS0 | $\overline{\text{RAS}}$ /DRAM Assertion Time | Command Interval after Synchronous DRAM Refresh |
|---------------|---------------|---------------|--|---|
| 0 | 0 | 0 | 2 | 4 + TRC (Initial value) |
| | | 1 | 3 | 5 + TRC |
| | 1 | 0 | 4 | 6 + TRC |
| | | 1 | 5 | 7 + TRC |
| 1 | 0 | 0 | 6 | 8 + TRC |
| | | 1 | 7 | 9 + TRC |
| | 1 | 0 | 8 | 10 + TRC |
| | | 1 | 9 | 11 + TRC |

Note: TRC (Bits 29 to 27): RAS precharge interval at end of refresh

Bit 9—Burst Enable (BE): Specifies whether burst access is performed on DRAM interface. In synchronous DRAM access, burst access is always performed regardless of the specification of this bit. The DRAM transfer mode depends on EDOMODE.

| BE | EDOMODE | 8/16/32/64-Bit Transfer | 32-Byte Transfer |
|----|---------|-------------------------|--------------------|
| 0 | 0 | Single | Single |
| | 1 | Setting prohibited | Setting prohibited |
| 1 | 0 | Single/fast page* | Fast page |
| | 1 | EDO | EDO |

Note: * In fast page mode, 32-bit or 64-bit transfer with a 16-bit bus, 64-bit transfer with a 32-bit bus

Bits 8 and 7—Memory Data Size (SZ1, SZ0): These bits specify the bus width of DRAM and synchronous DRAM. This setting has priority over the BCR2 register setting.

| Bit 8: SZ1 | Bit 7: SZ0 | Description | |
|------------|------------|-------------------------------|-------------------------------|
| | | DRAM | SDRAM |
| 0 | 0 | Reserved (Setting prohibited) | Reserved (Setting prohibited) |
| | 1 | Reserved (Setting prohibited) | Reserved (Setting prohibited) |
| 1 | 0 | 16 bits | Reserved (Setting prohibited) |
| | 1 | 32 bits | 32 bits |

Bits 6 to 3—Address Multiplexing (AMXEXT, AMX2–AMX0): These bits specify address multiplexing for DRAM and synchronous DRAM. The address shift value is different for the DRAM interface and the synchronous DRAM interface.

- For DRAM Interface:

| Bit 6: AMXEXT | Bit 5: AMX2 | Bit 4: AMX1 | Bit 3: AMX0 | Description |
|------------------|----------------|----------------|----------------|---|
| | | | | DRAM |
| 0* | 0 | 0 | 0 | 8-bit column address product (Initial value) |
| | | | 1 | 9-bit column address product |
| | | 1 | 0 | 10-bit column address product |
| | | | 1 | 11-bit column address product |
| | 1 | 0 | 0 | 12-bit column address product |
| | | | 1 | Reserved (Setting prohibited) |
| | | 1 | 0 | Reserved (Setting prohibited) |
| | | | 1 | Reserved (Setting prohibited) |

Note: * When the DRAM interface is used, clear the AMXEXT bit to 0.

- For Synchronous DRAM Interface:

| AMX | AMXEXT | SZ | Example Synchronous DRAM Configurations | BANK |
|-----|--------|----|---|-----------|
| 0 | 0 | 32 | (16M: 512K × 16 bits × 2) × 2 | a[21]* |
| | 1 | | (16M: 512K × 16 bits × 2) × 2 | a[20]* |
| 1 | 0 | | (16M: 1M × 8 bits × 2) × 4 | a[22]* |
| | 1 | | (16M: 1M × 8 bits × 2) × 4 | a[21]* |
| 2 | — | | (64M: 1M × 16 bits × 4) × 2 | a[23:22]* |
| 3 | — | | (64M: 2M × 8 bits × 4) × 4 | a[24:23]* |
| 4 | — | | (64M: 512K × 32 bits × 4) × 1 | a[22:21]* |
| 5 | — | | (64M: 1M × 32 bits × 2) × 1 | a[22]* |
| 6 | 0 | | (64M: 4M × 4 bits × 4) × 8 | a[25:24]* |
| | 1 | | (256M: 4M × 16 bits × 4) × 2 | a[25:24]* |
| 7 | — | | (16M: 256K × 32 bits × 2) × 1 | a[20]* |

Note: * a[x]: External address, not address pin

Bit 2—Refresh Control (RFSH): Specifies refresh control. Selects whether refreshing is performed for DRAM and synchronous DRAM. When the refresh function is not used, the refresh request cycle generation timer can be used as an interval timer.

| Bit 2: RFSH | Description |
|-------------|---|
| 0 | Refreshing is not performed (Initial value) |
| 1 | Refreshing is performed |

Bit 1—Refresh Mode (RMODE): Specifies whether normal refreshing or self-refreshing is performed when the RFSH bit is set to 1. When the RFSH bit is 1 and this bit is cleared to 0, CAS-before-RAS refreshing or auto-refreshing is performed for DRAM and synchronous DRAM, using the cycle set by refresh-related registers RTCNT, RTCOR, and RTCSR. If a refresh request is issued during an external bus cycle, the refresh cycle is executed when the bus cycle ends. When the RFSH bit is 1 and this bit is set to 1, the self-refresh state is set for DRAM and synchronous DRAM, after waiting for the end of any currently executing external bus cycle. All refresh requests for memory in the self-refresh state are ignored.

| Bit 1: RMODE | Description |
|--------------|--|
| 0 | CAS-before-RAS refreshing is performed (when RFSH = 1) (Initial value) |
| 1 | Self-refreshing is performed (when RFSH = 1) |

Bit 0—EDO Mode (EDOMODE): Used to specify the data sampling timing for data reads when using EDO mode DRAM interface. The setting of this bit does not affect the operation timing of memory other than DRAM. Set this bit to 1 only when DRAM is used.

13.2.9 PCMCIA Control Register (PCR)

The PCMCIA control register (PCR) is a 16-bit readable/writable register that specifies the \overline{OE} and \overline{WE} signal assertion/negation timing for the PCMCIA interface connected to areas 5 and 6. The \overline{OE} and \overline{WE} signal assertion width is set by the wait control bits in the WCR2 register.

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

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Bit name: | A5PCW1 | A5PCW0 | A6PCW1 | A6PCW0 | A5TED2 | A5TED1 | A5TED0 | A6TED2 |
| 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: | A6TED1 | A6TED0 | A5TEH2 | A5TEH1 | A5TEH0 | A6TEH2 | A6TEH1 | A6TEH0 |
| 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—PCMCIA Wait (A5PCW1, A5PCW0): These bits specify the number of waits to be added to the number of waits specified by WCR2 in a low-speed PCMCIA wait cycle. The setting of these bits is selected when the PCMCIA interface access TC bit is 0.

| Bit 15: A5PCW1 | Bit 14: A5PCW0 | Waits Inserted |
|----------------|----------------|-------------------|
| 0 | 0 | 0 (Initial value) |
| | 1 | 15 |
| 1 | 0 | 30 |
| | 1 | 50 |

Bits 13 and 12—PCMCIA Wait (A6PCW1, A6PCW0): These bits specify the number of waits to be added to the number of waits specified by WCR2 in a low-speed PCMCIA wait cycle. The setting of these bits is selected when the PCMCIA interface access TC bit is 0.

| Bit 13: A6PCW1 | Bit 12: A6PCW0 | Waits Inserted |
|----------------|----------------|-------------------|
| 0 | 0 | 0 (Initial value) |
| | 1 | 15 |
| 1 | 0 | 30 |
| | 1 | 50 |

Bits 11 to 9—Address- $\overline{OE}/\overline{WE}$ Assertion Delay (A5TED2–A5TED0): These bits set the delay time from address output to $\overline{OE}/\overline{WE}$ assertion on the connected PCMCIA interface. The setting of these bits is selected when the PCMCIA interface access TC bit is 0.

| Bit 11: A5TED2 | Bit 10: A5TED1 | Bit 9: A5TED0 | Waits Inserted | |
|----------------|----------------|---------------|-------------------|---|
| 0 | 0 | 0 | 0 (Initial value) | |
| | | 1 | 1 | |
| | 1 | 0 | 2 | |
| | | 1 | 3 | |
| | 1 | 0 | 0 | 6 |
| | | | 1 | 9 |
| 1 | | 0 | 12 | |
| | | 1 | 15 | |

Bits 8 to 6—Address- $\overline{\text{OE}}/\overline{\text{WE}}$ Assertion Delay (A6TED2–A6TED0): These bits set the delay time from address output to $\overline{\text{OE}}/\overline{\text{WE}}$ assertion on the connected PCMCIA interface. The setting of these bits is selected when the PCMCIA interface access TC bit is 0.

| Bit 8: A6TED2 | Bit 7: A6TED1 | Bit 6: A6TED0 | Waits Inserted |
|---------------|---------------|---------------|-------------------|
| 0 | 0 | 0 | 0 (Initial value) |
| | | 1 | 1 |
| | 1 | 0 | 2 |
| | | 1 | 3 |
| 1 | 0 | 0 | 6 |
| | | 1 | 9 |
| | 1 | 0 | 12 |
| | | 1 | 15 |

Bits 5 to 3— $\overline{\text{OE}}/\overline{\text{WE}}$ Negation-Address Delay (A5TEH2–A5TEH0): These bits set the address hold delay time from $\overline{\text{OE}}/\overline{\text{WE}}$ negation in a write on the connected PCMCIA interface or in an I/O card read. In the case of a memory card read, the address hold delay time from the data sampling timing is set. The setting of these bits is selected when the PCMCIA interface access TC bit is 0.

| Bit 5: A5TEH2 | Bit 4: A5TEH1 | Bit 3: A5TEH0 | Waits Inserted |
|---------------|---------------|---------------|-------------------|
| 0 | 0 | 0 | 0 (Initial value) |
| | | 1 | 1 |
| | 1 | 0 | 2 |
| | | 1 | 3 |
| 1 | 0 | 0 | 6 |
| | | 1 | 9 |
| | 1 | 0 | 12 |
| | | 1 | 15 |

Bits 2 to 0— $\overline{OE}/\overline{WE}$ Negation-Address Delay (A6TEH2–A6TEH0): These bits set the address hold delay time from $\overline{OE}/\overline{WE}$ negation in a write on the connected PCMCIA interface or in an I/O card read. In the case of a memory card read, the address hold delay time from the data sampling timing is set. The setting of these bits is selected when the PCMCIA interface access TC bit is 0.

| Bit 2: A6TEH2 | Bit 1: A6TEH1 | Bit 0: A6TEH0 | Waits Inserted |
|---------------|---------------|---------------|-------------------|
| 0 | 0 | 0 | 0 (Initial value) |
| | | 1 | 1 |
| | 1 | 0 | 2 |
| | | 1 | 3 |
| 1 | 0 | 0 | 6 |
| | | 1 | 9 |
| | 1 | 0 | 12 |
| | | 1 | 15 |

13.2.10 Synchronous DRAM Mode Register (SDMR)

The synchronous DRAM mode register (SDMR) is a write-only virtual 16-bit register that is written to via the synchronous DRAM address bus, and sets the mode of the area 2 and area 3 synchronous DRAM.

Settings for the SDMR register must be made before accessing synchronous DRAM.

| | | | | | | | | |
|----------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | W | W | W | W | W | W | W | W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | W | W | W | W | W | W | W | W |

Since the address bus, not the data bus, is used to write to the synchronous DRAM mode register, if the value to be set is “X” and the SDMR register address is “Y”, value “X” is written to the synchronous DRAM mode register by performing a write to address X + Y. When the synchronous DRAM bus width is set to 32 bits, as A0 of the synchronous DRAM is connected to

A2 of this LSI, and A1 of the synchronous DRAM is connected to A3 of this LSI, the value actually written to the synchronous DRAM is the value of “X” shifted 2 bits to the right.

For example, to write H'0230 to the area 2 SDMR register, arbitrary data is written to address H'FF900000 (address “Y”) + H'08C0 (value “X”) (= H'FF9008C0). As a result, H'0230 is written to the SDMR register. The range of value “X” is H'0000 to H'0FFC.

Similarly, to write H'0230 to the area 3 SDMR register, arbitrary data is written to address H'FF940000 (address “Y”) + H'08C0 (value “X”) (= H'FF9408C0). As a result, H'0230 is written to the SDMR register. The range of value “X” is H'0000 to H'0FFC.

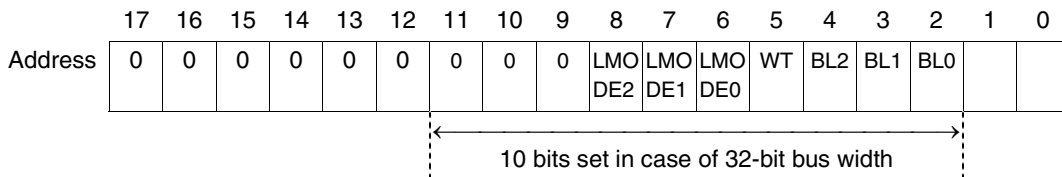
The lower 16 bits of the address are set in the synchronous DRAM mode register.

The burst length is 4 and 8*. Setting to SDMR writes into the following addresses in byte size.

| Bus Width | Burst length | CAS Latency | Area 2 | Area 3 |
|-----------|--------------|-------------|------------|------------|
| 32 | 4 | 1 | H'FF900048 | H'FF940048 |
| | | 2 | H'FF900088 | H'FF940088 |
| | | 3 | H'FF9000C8 | H'FF9400C8 |
| 32 | 8* | 1 | H'FF90004C | H'FF94004C |
| | | 2 | H'FF90008C | H'FF94008C |
| | | 3 | H'FF9000CC | H'FF9400CC |

Note: * SH7751R only

For a 32-bit bus:



LMODE: CAS latency

BL: Burst length

WT: Wrap type (0: Sequential)

| | |
|---------------|---------------|
| BL | LMODE |
| 000: Reserved | 000: Reserved |
| 001: Reserved | 001: 1 |
| 010: 4 | 010: 2 |
| 011: 8* | 011: 3 |
| 100: Reserved | 100: Reserved |
| 101: Reserved | 101: Reserved |
| 110: Reserved | 110: Reserved |
| 111: Reserved | 111: Reserved |

Note: * SH7751R only

13.2.11 Refresh Timer Control/Status Register (RTCSR)

The refresh timer control/status register (RTCSR) is a 16-bit readable/writable register that specifies the refresh cycle and whether interrupts are to be generated.

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

| | | | | | | | | |
|----------------|-----|------|------|------|------|-----|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | — | — | — | — | — |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CMF | CMIE | CKS2 | CKS1 | CKS0 | OVF | OVIE | LMTS |
| 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 to 8—Reserved: These bits are always read as 0. For the write values, see section 13.2.15, Notes on Accessing Refresh Control Registers.

Bit 7—Compare-Match Flag (CMF): Status flag that indicates a match between the refresh timer counter (RTCNT) and refresh time constant register (RTCOR) values.

| Bit 7: CMF | Description |
|------------|---|
| 0 | RTCNT and RTCOR values do not match (Initial value) [Clearing condition] When 0 is written to CMF |
| 1 | RTCNT and RTCOR values match [Setting condition] When RTCNT = RTCOR* |

Note: * If 1 is written, the original value is retained.

Bit 6—Compare-Match Interrupt Enable (CMIE): Controls generation or suppression of an interrupt request when the CMF flag is set to 1 in RTCSR. Do not set this bit to 1 when CAS-before-RAS refreshing or auto-refreshing is used.

| Bit 6: CMIE | Description |
|-------------|--|
| 0 | Interrupt requests initiated by CMF are disabled (Initial value) |
| 1 | Interrupt requests initiated by CMF are enabled |

Bits 5 to 3—Clock Select Bits (CKS2–CKS0): These bits select the input clock for RTCNT. The base clock is the external bus clock (CKIO). The RTCNT count clock is obtained by scaling CKIO by the specified factor.

| Bit 5: CKS2 | Bit 4: CKS1 | Bit 3: CKS0 | Description |
|-------------|-------------|-------------|--------------------------------------|
| 0 | 0 | 0 | Clock input disabled (Initial value) |
| | | 1 | Bus clock (CKIO)/4 |
| | 1 | 0 | CKIO/16 |
| | | 1 | CKIO/64 |
| 1 | 0 | 0 | CKIO/256 |
| | | 1 | CKIO/1024 |
| | 1 | 0 | CKIO/2048 |
| | | 1 | CKIO/4096 |

Bit 2—Refresh Count Overflow Flag (OVF): Status flag that indicates that the number of refresh requests indicated by the refresh count register (RFCR) has exceeded the number specified by the LMTS bit in RTCSR.

| Bit 2: OVF | Description |
|------------|--|
| 0 | RFCR has not overflowed the count limit indicated by LMTS (Initial value) [Clearing condition] When 0 is written to OVF |
| 1 | RFCR has overflowed the count limit indicated by LMTS [Setting condition] When RFCR overflows the count limit set by LMTS* |

Note: * If 1 is written, the original value is retained.

Bit 1—Refresh Count Overflow Interrupt Enable (OVIE): Controls generation or suppression of an interrupt request when the OVF flag is set to 1 in RTCSR.

| Bit 1: OVIE | Description |
|-------------|--|
| 0 | Interrupt requests initiated by OVF are disabled (Initial value) |
| 1 | Interrupt requests initiated by OVF are enabled |

Bit 0—Refresh Count Overflow Limit Select (LMTS): Specifies the count limit to be compared with the refresh count indicated by the refresh count register (RFCR). If the RFCR register value exceeds the value specified by LMTS, the OVF flag is set.

| Bit 0: LMTS | Description |
|-------------|-------------------------------------|
| 0 | Count limit is 1024 (Initial value) |
| 1 | Count limit is 512 |

13.2.12 Refresh Timer Counter (RTCNT)

The refresh timer counter (RTCNT) is an 8-bit readable/writable counter that is incremented by the input clock (selected by bits CKS2–CKS0 in the RTCSR register). When the RTCNT counter value matches the RTCOR register value, the CMF bit is set in the RTCSR register and the RTCNT counter is cleared.

RTCNT is initialized to H'0000 by a power-on reset, but continues to count when a manual reset is performed. In standby mode, RTCNT is not initialized, and retains its contents.

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

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

13.2.13 Refresh Time Constant Register (RTCOR)

The refresh time constant register (RTCOR) is a readable/writable register that specifies the upper limit of the RTCNT counter. The RTCOR register and RTCNT counter values (lower 8 bits) are constantly compared, and when they match the CMF bit is set in the RTCSR register and the RTCNT counter is cleared to 0. If the refresh bit (RFSH) has been set to 1 in the memory control register (MCR) and CAS-before-RAS has been selected as the refresh mode, a memory refresh cycle is generated when the CMF bit is set.

RTCOR is initialized to H'0000 by a power-on reset, but is not initialized, and retains its contents, in a manual reset and in standby mode.

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

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

13.2.14 Refresh Count Register (RFCR)

The refresh count register (RFCR) is a 10-bit readable/writable counter that counts the number of refreshes by being incremented each time the RTCOR register and RTCNT counter values match. If the RFCR register value exceeds the count limit specified by the LMTS bit in the RTCSR register, the OVF flag is set in the RTCSR register and the RFCR register is cleared.

RFCR is initialized to H'0000 by a power-on reset, but is not initialized, and retains its contents, in a manual reset and in standby mode.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | — | — | — | — | — | — | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

13.2.15 Notes on Accessing Refresh Control Registers

When the refresh timer control/status register (RTCSR), refresh timer counter (RTCNT), refresh time constant register (RTCOR), and refresh count register (RFCR) are written to, a special code is added to the data to prevent inadvertent rewriting in the event of program runaway, etc. The following procedures should be used for read/write operations.

Writing to RTCSR, RTCNT, RTCOR, and RFCR: A word transfer instruction must always be used when writing to RTCSR, RTCNT, RTCOR, or RFCR. A write cannot be performed with a byte transfer instruction.

When writing to RTCSR, RTCNT, or RTCOR, set B'10100101 in the upper byte and the write data in the lower byte, as shown in figure 13.5. When writing to RFCR, set B'101001 in the 6 bits starting from the MSB in the upper byte, and the write data in the remaining bits.

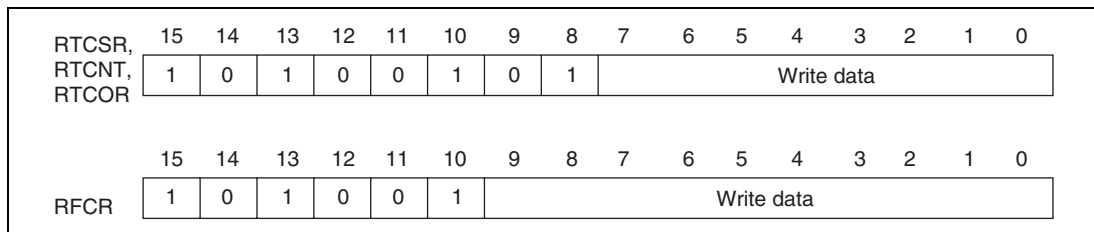


Figure 13.5 Writing to RTCSR, RTCNT, RTCOR, and RFCR

Reading RTCSR, RTCNT, RTCOR, and RFCR: A 16-bit access must always be used when reading RTCSR, RTCNT, RTCOR, or RFCR. Undefined bits are read as 0.

13.3 Operation

13.3.1 Endian/Access Size and Data Alignment

This LSI supports both big-endian mode, in which the most significant byte (MSByte) is at the 0 address end in a string of byte data, and little-endian mode, in which the least significant byte (LSByte) is at the 0 address end. The mode is set by means of the MD5 external pin in a power-on reset by means of the RESET pin, big-endian mode being set if the MD5 pin is low, and little-endian mode if it is high.

A data bus width of 8, 16, or 32 bits can be selected for normal memory, 16 or 32 bits for DRAM, 32 bit for synchronous DRAM, and 8 or 16 bits for the PCMCIA interface. Data alignment is carried out according to the data bus width and endian mode of each device. Accordingly, when the data bus width is narrower than the access size, multiple bus cycles are automatically generated to reach the access size. In this case, access is performed by automatically incrementing addresses to the bus width. For example, when a long word access is performed at the area with an 8-bit bus width in the SRAM interface, each address is incremented one by one, and then access is performed four times. In the 32-byte transfer, a total of 32-byte data is continuously transferred according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed in 32-byte boundary data using wraparound. During these transfers, the bus is not released and refresh operation is not performed. In this LSI, data alignment and data length conversion between the different interfaces is performed automatically. Quadword access is used only in transfer by the DMAC.

The relationship between the endian mode, device data length, and access unit, is shown in tables 13.8 to 13.13.

Data Configuration

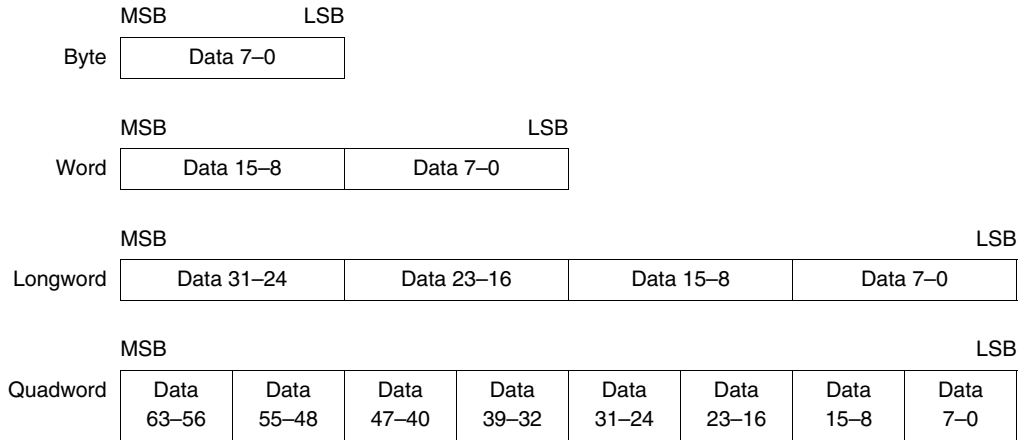


Table 13.8 32-Bit External Device/Big-Endian Access and Data Alignment

| Operation | | Data Bus | | | | | Strobe Signals | | | |
|-------------|---------|----------|------------|------------|------------|------------|---|---|---|---|
| Access Size | Address | No. | D31-D24 | D23-D16 | D15-D8 | D7-D0 | $\overline{WE3}$, $\overline{CAS3}$, DQM3 | $\overline{WE2}$, $\overline{CAS2}$, DQM2 | $\overline{WE1}$, $\overline{CAS1}$, DQM1 | $\overline{WE0}$, $\overline{CAS0}$, DQM0 |
| Byte | 4n | 1 | Data 7-0 | — | — | — | Asserted | | | |
| | 4n + 1 | 1 | — | Data 7-0 | — | — | Asserted | | | |
| | 4n + 2 | 1 | — | — | Data 7-0 | — | Asserted | | | |
| | 4n + 3 | 1 | — | — | — | Data 7-0 | Asserted | | | |
| Word | 4n | 1 | Data 15-8 | Data 7-0 | — | — | Asserted | | Asserted | |
| | 4n + 2 | 1 | — | — | Data 15-8 | Data 7-0 | Asserted | | Asserted | |
| Long-word | 4n | 1 | Data 31-24 | Data 23-16 | Data 15-8 | Data 7-0 | Asserted | Asserted | Asserted | Asserted |
| Quad-word | 8n | 1 | Data 63-56 | Data 55-48 | Data 47-40 | Data 39-32 | Asserted | Asserted | Asserted | Asserted |
| | 8n + 4 | 2 | Data 31-24 | Data 23-16 | Data 15-8 | Data 7-0 | Asserted | Asserted | Asserted | Asserted |

Table 13.9 16-Bit External Device/Big-Endian Access and Data Alignment

| Operation | | Data Bus | | | | | Strobe Signals | | | |
|---------------|---------|----------|---------|---------|---------------|---------------|--|--|--|--|
| Access Size | Address | No. | D31–D24 | D23–D16 | D15–D8 | D7–D0 | <u>WE3,</u> <u>CAS3,</u> <u>DQM3</u> | <u>WE2,</u> <u>CAS2,</u> <u>DQM2</u> | <u>WE1,</u> <u>CAS1,</u> <u>DQM1</u> | <u>WE0,</u> <u>CAS0,</u> <u>DQM0</u> |
| Byte | 2n | 1 | — | — | Data 7–0 | — | | | | Asserted |
| | 2n + 1 | 1 | — | — | — | Data 7–0 | | | | Asserted |
| Word | 2n | 1 | — | — | Data 15–8 | Data 7–0 | | | Asserted | Asserted |
| Long- word | 4n | 1 | — | — | Data 31–24 | Data 23–16 | | | Asserted | Asserted |
| | 4n + 2 | 2 | — | — | Data 15–8 | Data 7–0 | | | Asserted | Asserted |
| Quad- word | 8n | 1 | — | — | Data 63–56 | Data 55–48 | | | Asserted | Asserted |
| | 8n + 2 | 2 | — | — | Data 47–40 | Data 39–32 | | | Asserted | Asserted |
| | 8n + 4 | 3 | — | — | Data 31–24 | Data 23–16 | | | Asserted | Asserted |
| | 8n + 6 | 4 | — | — | Data 15–8 | Data 7–0 | | | Asserted | Asserted |

Table 13.10 8-Bit External Device/Big-Endian Access and Data Alignment

| Operation | | | Data Bus | | | | Strobe Signals | | | |
|---------------|---------|-----|----------|---------|--------|---------------|-------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|
| Access Size | Address | No. | D31–D24 | D23–D16 | D15–D8 | D7–D0 | $\overline{WE3}$, CAS3, DQM3 | $\overline{WE2}$, CAS2, DQM2 | $\overline{WE1}$, CAS1, DQM1 | $\overline{WE0}$, CAS0, DQM0 |
| Byte | n | 1 | — | — | — | Data 7–0 | | | | Asserted |
| Word | 2n | 1 | — | — | — | Data 15–8 | | | | Asserted |
| | 2n + 1 | 2 | — | — | — | Data 7–0 | | | | Asserted |
| Long- word | 4n | 1 | — | — | — | Data 31–24 | | | | Asserted |
| | 4n + 1 | 2 | — | — | — | Data 23–16 | | | | Asserted |
| | 4n + 2 | 3 | — | — | — | Data 15–8 | | | | Asserted |
| | 4n + 3 | 4 | — | — | — | Data 7–0 | | | | Asserted |
| Quad- word | 8n | 1 | — | — | — | Data 63–56 | | | | Asserted |
| | 8n + 1 | 2 | — | — | — | Data 55–48 | | | | Asserted |
| | 8n + 2 | 3 | — | — | — | Data 47–40 | | | | Asserted |
| | 8n + 3 | 4 | — | — | — | Data 39–32 | | | | Asserted |
| | 8n + 4 | 5 | — | — | — | Data 31–24 | | | | Asserted |
| | 8n + 5 | 6 | — | — | — | Data 23–16 | | | | Asserted |
| | 8n + 6 | 7 | — | — | — | Data 15–8 | | | | Asserted |
| | 8n + 7 | 8 | — | — | — | Data 7–0 | | | | Asserted |

Table 13.11 32-Bit External Device/Little-Endian Access and Data Alignment

| Operation | | Data Bus | | | | | Strobe Signals | | | | |
|---------------|--------|----------|---------------|---------------|---------------|---------------|----------------|-----------------------|-----------------------|-----------------------|-----------------------|
| | | Address | No. | D31–D24 | D23–D16 | D15–D8 | D7–D0 | WE3, CAS3, DQM3 | WE2, CAS2, DQM2 | WE1, CAS1, DQM1 | WE0, CAS0, DQM0 |
| Byte | 4n | 1 | — | — | — | Data 7–0 | — | — | — | — | Asserted |
| | 4n + 1 | 1 | — | — | — | Data 7–0 | — | — | — | Asserted | |
| | 4n + 2 | 1 | — | Data 7–0 | — | — | — | — | Asserted | — | |
| | 4n + 3 | 1 | Data 7–0 | — | — | — | — | — | Asserted | — | |
| Word | 4n | 1 | — | — | Data 15–8 | Data 7–0 | — | — | — | — | Asserted Asserted |
| | 4n + 2 | 1 | Data 15–8 | Data 7–0 | — | — | — | Asserted | Asserted | — | — |
| Long- word | 4n | 1 | Data 31–24 | Data 23–16 | Data 15–8 | Data 7–0 | — | — | Asserted | Asserted | Asserted Asserted |
| Quad- word | 8n | 1 | Data 31–24 | Data 23–16 | Data 15–8 | Data 7–0 | — | — | Asserted | Asserted | Asserted Asserted |
| | 8n + 4 | 2 | Data 63–56 | Data 55–48 | Data 47–40 | Data 39–32 | — | — | Asserted | Asserted | Asserted Asserted |

Table 13.12 16-Bit External Device/Little-Endian Access and Data Alignment

| Operation | | | Data Bus | | | | Strobe Signals | | | |
|---------------|---------|-----|----------|---------|---------------|---------------|-------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|
| Access Size | Address | No. | D31–D24 | D23–D16 | D15–D8 | D7–D0 | $\overline{WE3}$, CAS3, DQM3 | $\overline{WE2}$, CAS2, DQM2 | $\overline{WE1}$, CAS1, DQM1 | $\overline{WE0}$, CAS0, DQM0 |
| Byte | 2n | 1 | — | — | — | Data 7–0 | | | | Asserted |
| | 2n + 1 | 1 | — | — | Data 7–0 | — | | | | Asserted |
| Word | 2n | 1 | — | — | Data 15–8 | Data 7–0 | | | | Asserted Asserted |
| Long- word | 4n | 1 | — | — | Data 15–8 | Data 7–0 | | | | Asserted Asserted |
| | 4n + 2 | 2 | — | — | Data 31–24 | Data 23–16 | | | | Asserted Asserted |
| Quad- word | 8n | 1 | — | — | Data 15–8 | Data 7–0 | | | | Asserted Asserted |
| | 8n + 2 | 2 | — | — | Data 31–24 | Data 23–16 | | | | Asserted Asserted |
| | 8n + 4 | 3 | — | — | Data 47–40 | Data 39–32 | | | | Asserted Asserted |
| | 8n + 6 | 4 | — | — | Data 63–56 | Data 55–48 | | | | Asserted Asserted |

Table 13.13 8-Bit External Device/Little-Endian Access and Data Alignment

| Operation | | Data Bus | | | | | Strobe Signals | | | |
|---------------|--------|----------|-----|---------|---------|---------------|----------------|------------------------------|------------------------------|------------------------------|
| | | Address | No. | D31–D24 | D23–D16 | D15–D8 | D7–D0 | <u>WE3,</u> CAS3, DQM3 | <u>WE2,</u> CAS2, DQM2 | <u>WE1,</u> CAS1, DQM1 |
| Byte | n | 1 | — | — | — | Data 7–0 | | | | Asserted |
| Word | 2n | 1 | — | — | — | Data 7–0 | | | | Asserted |
| | 2n + 1 | 2 | — | — | — | Data 15–8 | | | | Asserted |
| Long- word | 4n | 1 | — | — | — | Data 7–0 | | | | Asserted |
| | 4n + 1 | 2 | — | — | — | Data 15–8 | | | | Asserted |
| | 4n + 2 | 3 | — | — | — | Data 23–16 | | | | Asserted |
| | 4n + 3 | 4 | — | — | — | Data 31–24 | | | | Asserted |
| Quad- word | 8n | 1 | — | — | — | Data 7–0 | | | | Asserted |
| | 8n + 1 | 2 | — | — | — | Data 15–8 | | | | Asserted |
| | 8n + 2 | 3 | — | — | — | Data 23–16 | | | | Asserted |
| | 8n + 3 | 4 | — | — | — | Data 31–24 | | | | Asserted |
| | 8n + 4 | 5 | — | — | — | Data 39–32 | | | | Asserted |
| | 8n + 5 | 6 | — | — | — | Data 47–40 | | | | Asserted |
| | 8n + 6 | 7 | — | — | — | Data 55–48 | | | | Asserted |
| | 8n + 7 | 8 | — | — | — | Data 63–56 | | | | Asserted |

13.3.2 Areas

Area 0: For area 0, external address bits 28 to 26 are 000.

SRAM, MPX, and burst ROM can be set for this area.

A bus width of 8, 16, or 32 bits can be selected in a power-on reset by means of external pins MD4 and MD3. For details, see Memory Bus Width in section 13.1.5, Overview of Areas.

When area 0 is accessed, the $\overline{CS0}$ signal is asserted. In addition, the \overline{RD} signal, which can be used as \overline{OE} , and write control signals $\overline{WE0}$ to $\overline{WE3}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A0W2 to A0W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin (\overline{RDY}).

When the burst ROM interface is used, the number of burst cycle transfer states is selected in the range 2 to 9 according to the number of waits.

The read/write strobe signal address and the CS setup/hold time can be set, respectively, to 0 or 1 and to 0 to 3 cycles using the A0S0, A0H1, and A0H0 bits in the WCR3 register.

Area 1: For area 1, external address bits 28 to 26 are 001.

SRAM, MPX, and byte control SRAM can be set for this area.

A bus width of 8, 16, or 32 bits can be selected with bits A1SZ1 and A1SZ0 in the BCR2 register. When MPX interface is set, a bus width of 32 bit should be selected with bits A1SZ1 and A1SZ0 in the BCR2 register. When byte control SRAM interface is set, select a bus width of 16 or 32 bits.

When area 1 is accessed, the $\overline{CS1}$ signal is asserted. In addition, the \overline{RD} signal, which can be used as \overline{OE} , and write control signals $\overline{WE0}$ to $\overline{WE3}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A1W2 to A1W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin (\overline{RDY}).

The read/write strobe signal address and \overline{CS} setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A1S0 and bits A1H1 and A1H0 in the WCR3 register.

Area 2: For area 2, external address bits 28 to 26 are 010.

SRAM, MPX, and synchronous DRAM can be set to this area.

When SRAM interface is set, a bus width of 8, 16, or 32 bits can be selected with bits A2SZ1 and A2SZ0 in the BCR2 register. When MPX interface is set, a bus width of 32 bit should be selected with bits A2SZ1 and A2SZ0 in the BCR2 register. When synchronous DRAM interface is set, select 32 bit with the SZ bits in the MCR register. For details, see Memory Bus Width in section 13.1.5, Overview of Areas.

When area 2 is accessed, the $\overline{CS2}$ signal is asserted.

When SRAM interface is set, the \overline{RD} signal, which can be used as \overline{OE} , and write control signals $\overline{WE0}$ to $\overline{WE3}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A2W2 to A2W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin (\overline{RDY}).

The read/write strobe signal address and \overline{CS} setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A2S0 and bits A2H1 and A2H0 in the WCR3 register.

When synchronous DRAM interface is set, the \overline{RAS} and \overline{CAS} signals, RD/\overline{WR} signal, and byte control signals DQM0 to DQM3 are asserted, and address multiplexing is performed. \overline{RAS} , \overline{CAS} , and data timing control, and address multiplexing control, can be set using the MCR register.

Area 3: For area 3, external address bits 28 to 26 are 011.

SRAM, MPX, DRAM, and synchronous DRAM, can be set to this area.

When SRAM interface is set, a bus width of 8, 16, or 32 bits can be selected with bits A3SZ1 and A3SZ0 in the BCR2 register. When MPX interface is set, a bus width of 32 bit should be selected with bits A3SZ1 and A3SZ0 in the BCR2 register. When DRAM interface is set, 16 or 32 bits can be selected with the SZ bits in the MCR register. When synchronous DRAM interface is set, select 32 bit with the SZ bits in MCR. For details, see Memory Bus Width in section 13.1.5, Overview of Areas.

When area 3 is accessed, the $\overline{CS3}$ signal is asserted.

When SRAM interface is set, the \overline{RD} signal, which can be used as \overline{OE} , and write control signals $\overline{WE0}$ to $\overline{WE3}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A3W2 to A3W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A3S0 and bits A3H1 and A3H0 in the WCR3 register.

When synchronous DRAM interface is set, the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals, $\text{RD}/\overline{\text{WR}}$ signal, and byte control signals DQM0 to DQM3 are asserted, and address multiplexing is performed. When DRAM interface is set, the $\overline{\text{RAS}}$ signal, $\overline{\text{CAS0}}$ to $\overline{\text{CAS3}}$ signals, and $\text{RD}/\overline{\text{WR}}$ signal are asserted, and address multiplexing is performed. $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and data timing control, and address multiplexing control, can be set using the MCR register.

Area 4: For area 4, physical address bits 28 to 26 are 100.

SRAM, MPX, and byte control SRAM can be set to this area.

A bus width of 8, 16, or 32 bits can be selected with bits A4SZ1 and A4SZ0 in the BCR2 register. When MPX interface is set, a bus width of 32 bit should be selected with bits A4SZ1 and A4SZ0 in the BCR2 register. When byte control SRAM interface is set, select a bus width of 16 or 32 bits. For details, see Memory Bus Width in section 13.1.5, Overview of Areas.

When area 4 is accessed, the $\overline{\text{CS4}}$ signal is asserted, and the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE3}}$, are also asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A4W2 to A4W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A4S0 and bits A4H1 and A4H0 in the WCR3 register.

Area 5: For area 5, external address bits 28 to 26 are 101.

SRAM, MPX, burst ROM, and a PCMCIA interface can be set to this area.

When SRAM interface is set, a bus width of 8, 16, or 32 bits can be selected with bits A5SZ1 and A5SZ0 in the BCR2 register. When burst ROM interface is set, a bus width of 8, 16 or 32 bits can be selected with bits A5SZ1 and A5SZ0 in BCR2. When MPX interface is set, a bus width of 32 bit should be selected with bits A5SZ1 and A5SZ0 in BCR2. When a PCMCIA interface is set, either 8 or 16 bits should be selected with bits A5SZ1 and A5SZ0 in BCR2. For details, see Memory Bus Width in section 13.1.5, Overview of Areas.

When area 5 is accessed with SRAM interface set, the $\overline{\text{CS5}}$ signal is asserted. In addition, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE3}}$, are asserted. When a PCMCIA interface is connected, the $\overline{\text{CE1A}}$ and $\overline{\text{CE2A}}$ signals, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and the $\overline{\text{WE1}}$, $\overline{\text{WE2}}$, $\overline{\text{WE3}}$, and $\overline{\text{WE0}}$ signals, which can be used as $\overline{\text{WE}}$, $\overline{\text{ICIORD}}$, $\overline{\text{ICIOWR}}$, and $\overline{\text{REG}}$, respectively, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A5W2 to A5W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

When the burst function is used, the number of burst cycle transfer states is determined in the range 2 to 9 according to the number of waits.

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A5S0 and bits A5H1 and A5H0 in the WCR3 register.

When a PCMCIA interface is used, the address $\overline{\text{CE1A}}$ and $\overline{\text{CE2A}}$ setup and hold times with respect to the read/write strobe signals can be set in the range of 0 to 15 cycles with bits AnTED1 and AnTED0, and bits AnTEH1 and AnTEH0, in the PCR register. In addition, the number of wait cycles can be set in the range 0 to 50 with bits AnPCW1 and AnPCW0. The number of waits set in PCR is added to the number of waits set in WCR2.

Area 6: For area 6, external address bits 28 to 26 are 110.

SRAM, MPX, burst ROM, and a PCMCIA interface can be set to this area.

When SRAM interface is set, a bus width of 8, 16, or 32 bits can be selected with bits A6SZ1 and A6SZ0 in the BCR2 register. When burst ROM interface is set, a bus width of 8, 16 or 32 bits can be selected with bits A6SZ1 and A6SZ0 in BCR2. When MPX interface is set, a bus width of 32 bit should be selected with bits A6SZ1 and A6SZ0 in BCR2. When a PCMCIA interface is set, either 8 or 16 bits should be selected with bits A6SZ1 and A6SZ0 in BCR2. For details, see Memory Bus Width in section 13.1.5, Overview of Areas.

When area 6 space is accessed with SRAM interface set, the $\overline{CS6}$ signal is asserted. In addition, the \overline{RD} signal, which can be used as \overline{OE} , and write control signals $\overline{WE0}$ to $\overline{WE3}$, are asserted. When a PCMCIA interface is connected, the $\overline{CE1B}$ and $\overline{CE2B}$ signals, the \overline{RD} signal, which can be used as \overline{OE} , and the $\overline{WE1}$, $\overline{WE2}$, $\overline{WE3}$, and $\overline{WE0}$ signals, which can be used as \overline{WE} , \overline{ICIORD} , \overline{ICIOWR} , and \overline{REG} , respectively, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A6W2 to A6W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin (\overline{RDY}).

When the burst function is used, the number of burst cycle transfer states is determined in the range 2 to 9 according to the number of waits.

The read/write strobe signal address and \overline{CS} setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A6S0 and bits A6H1 and A6H0 in the WCR3 register.

When a PCMCIA interface is used, the address $\overline{CE1B}/\overline{CE2B}$ setup and hold times with respect to the read/write strobe signals can be set in the range of 0 to 15 cycles with bits AnTED1 and AnTED0, and bits AnTEH1 and AnTEH0, in the PCR register. In addition, the number of wait cycles can be set in the range 0 to 50 with bits AnPCW1 and AnPCW0. The number of waits set in PCR is added to the number of waits set in WCR2.

13.3.3 SRAM Interface

Basic Timing: The SRAM interface of this LSI uses strobe signal output in consideration of the fact that mainly SRAM will be connected. Figure 13.6 shows the SRAM timing of normal space accesses. A no-wait normal access is completed in two cycles. The \overline{BS} signal is asserted for one cycle to indicate the start of a bus cycle. The \overline{CSn} signal is asserted on the T1 rising edge, and negated on the next T2 clock rising edge. Therefore, there is no negation period in case of access at minimum pitch.

There is no access size specification when reading. The correct access address is output to the address pins (A[25:0]), but since there is no access size specification, 32 bits are always read in the case of a 32-bit device, and 16 bits in the case of a 16-bit device. When writing, only the \overline{WE} signal for the byte to be written is asserted. For details, see section 13.3.1, Endian/Access Size and Data Alignment.

In 32-byte transfer, a total of 32 bytes are transferred consecutively according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed in wraparound mode on the data at the 32-byte boundary. The bus is not released during this transfer.

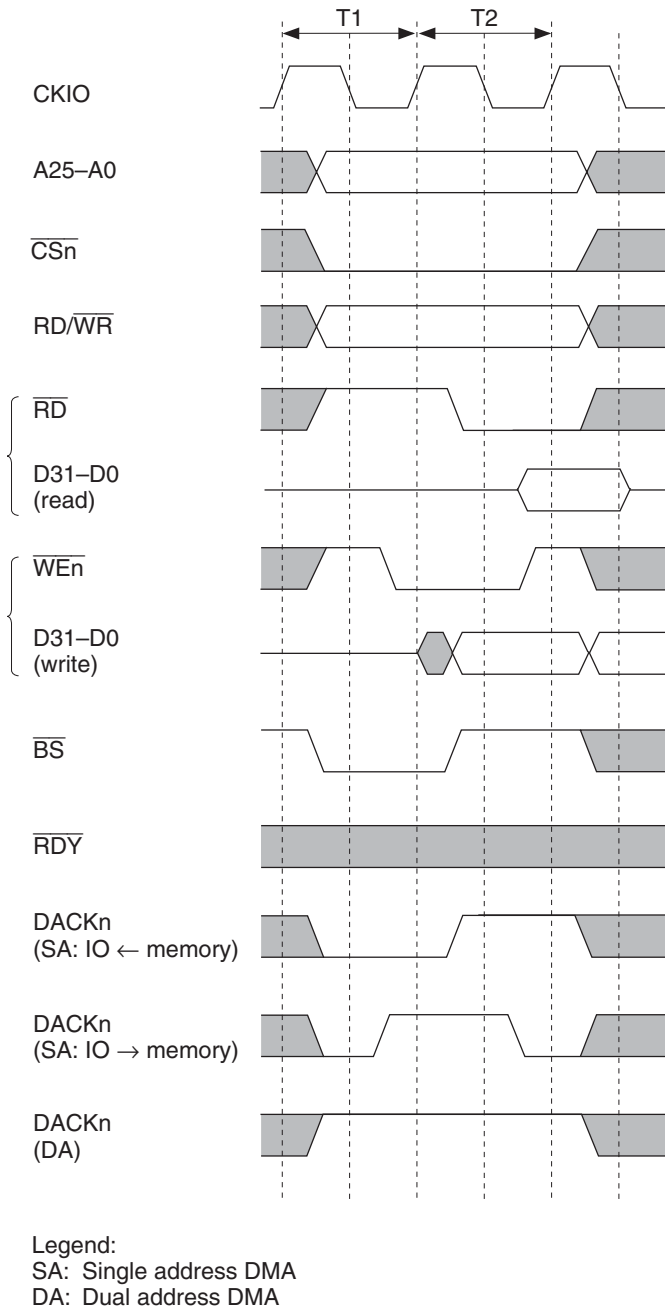


Figure 13.6 Basic Timing of SRAM Interface

Figures 13.7, 13.8, and 13.9 show examples of connection to 32-, 16-, and 8-bit data width SRAM.

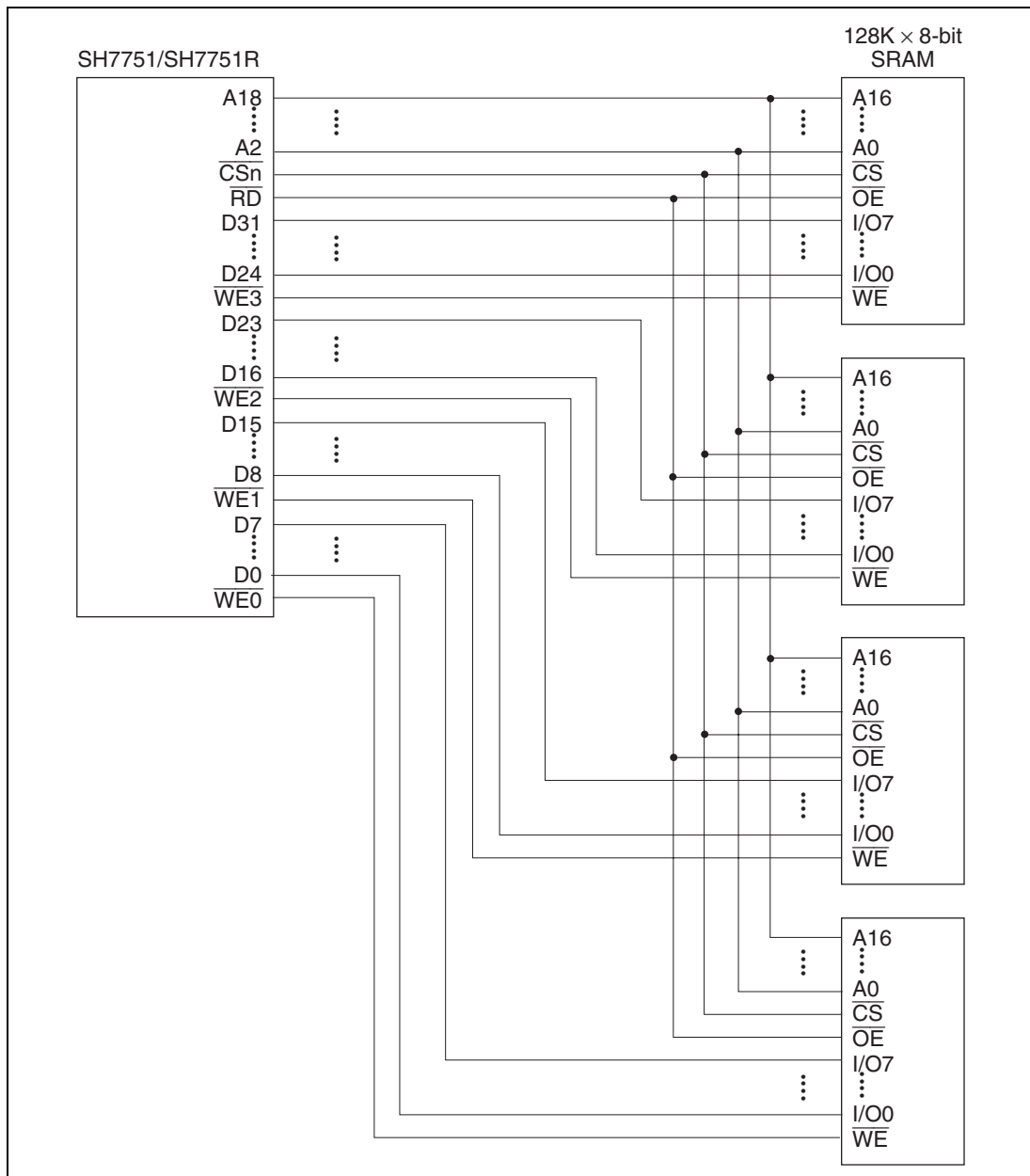


Figure 13.7 Example of 32-Bit Data Width SRAM Connection

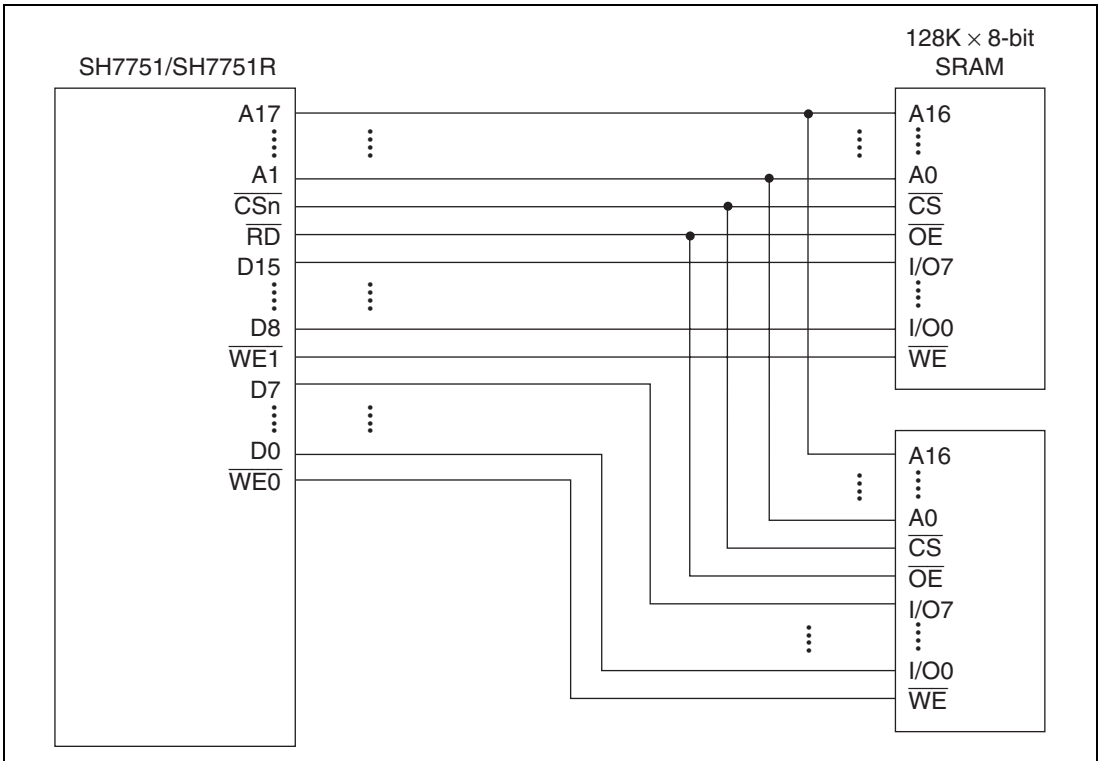


Figure 13.8 Example of 16-Bit Data Width SRAM Connection

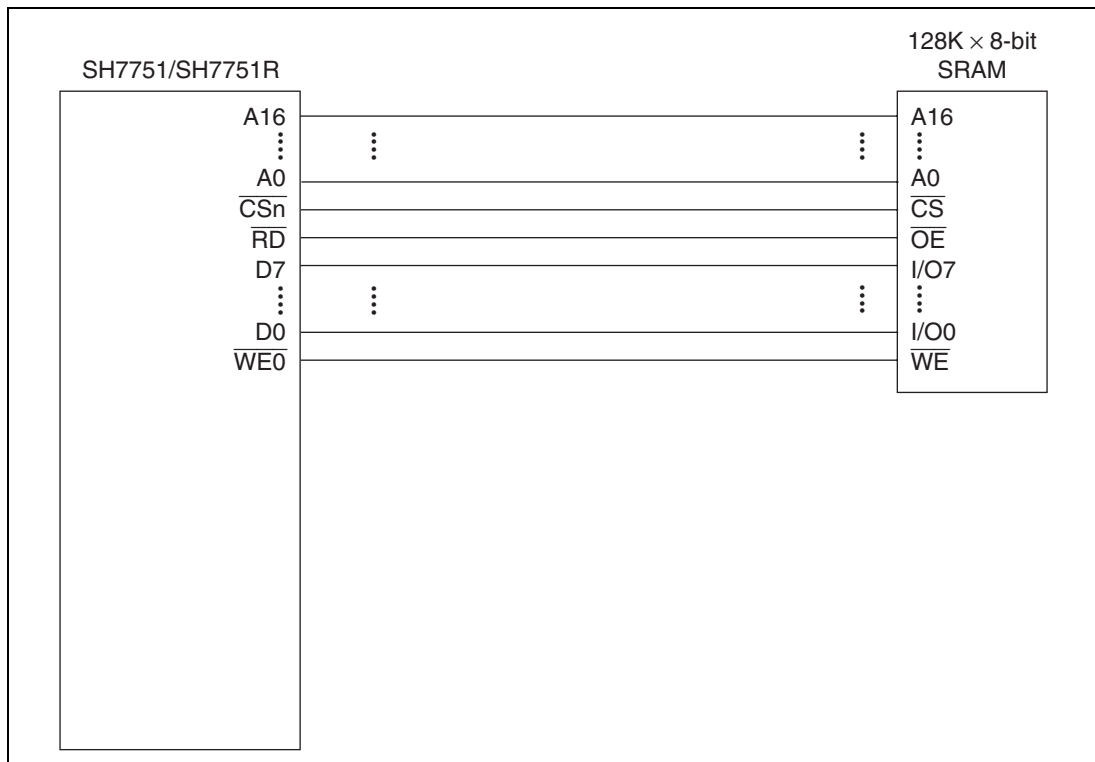
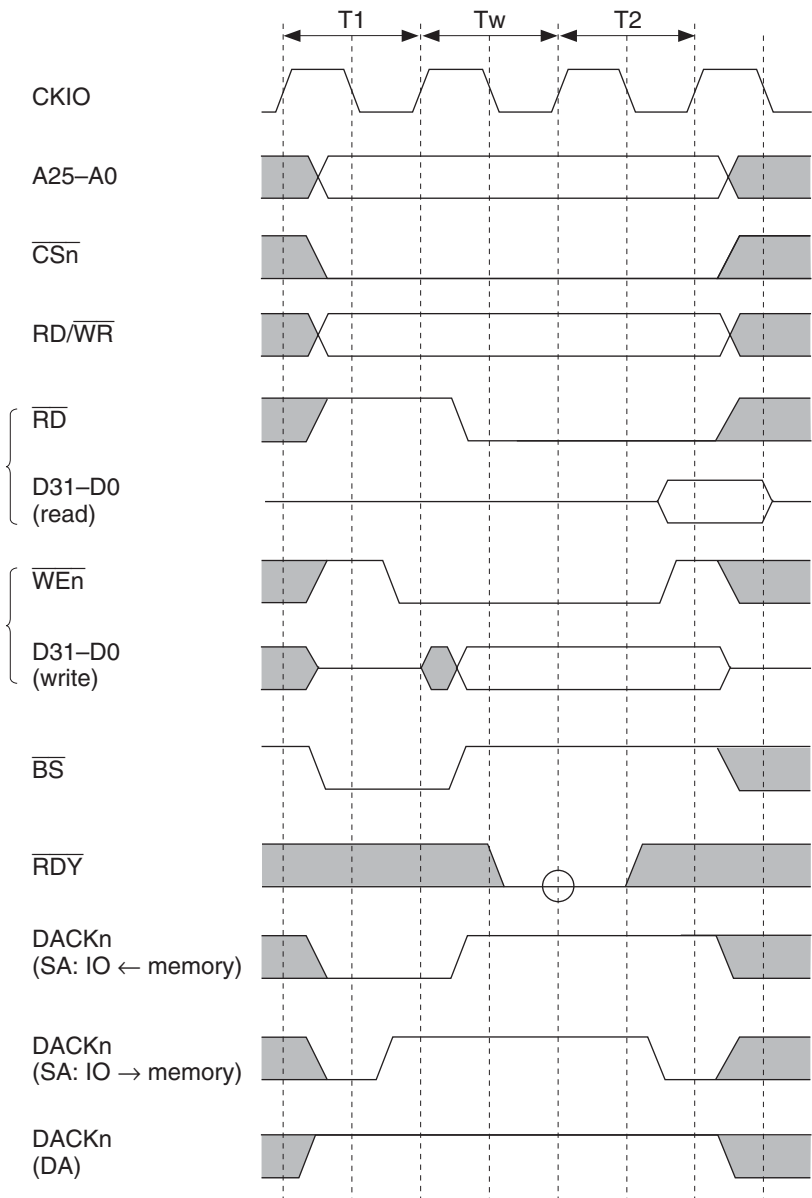


Figure 13.9 Example of 8-Bit Data Width SRAM Connection

Wait State Control: Wait state insertion on the SRAM interface can be controlled by the WCR2 settings. If the WCR2 wait specification bits corresponding to a particular area are not zero, a software wait is inserted in accordance with that specification. For details, see section 13.2.6, Wait Control Register 2 (WCR2).

The specified number of T_w cycles are inserted as wait cycles using the SRAM interface wait timing shown in figure 13.10.



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.10 SRAM Interface Wait Timing (Software Wait Only)

When software wait insertion is specified by WCR2, the external wait input $\overline{\text{RDY}}$ signal is also sampled. $\overline{\text{RDY}}$ signal sampling is shown in figure 13.11. A single-cycle wait is specified as a software wait. Sampling is performed at the transition from the Tw state to the T2 state; therefore, the $\overline{\text{RDY}}$ signal has no effect if asserted in the T1 cycle or the first Tw cycle. The $\overline{\text{RDY}}$ signal is sampled on the rising edge of the clock.

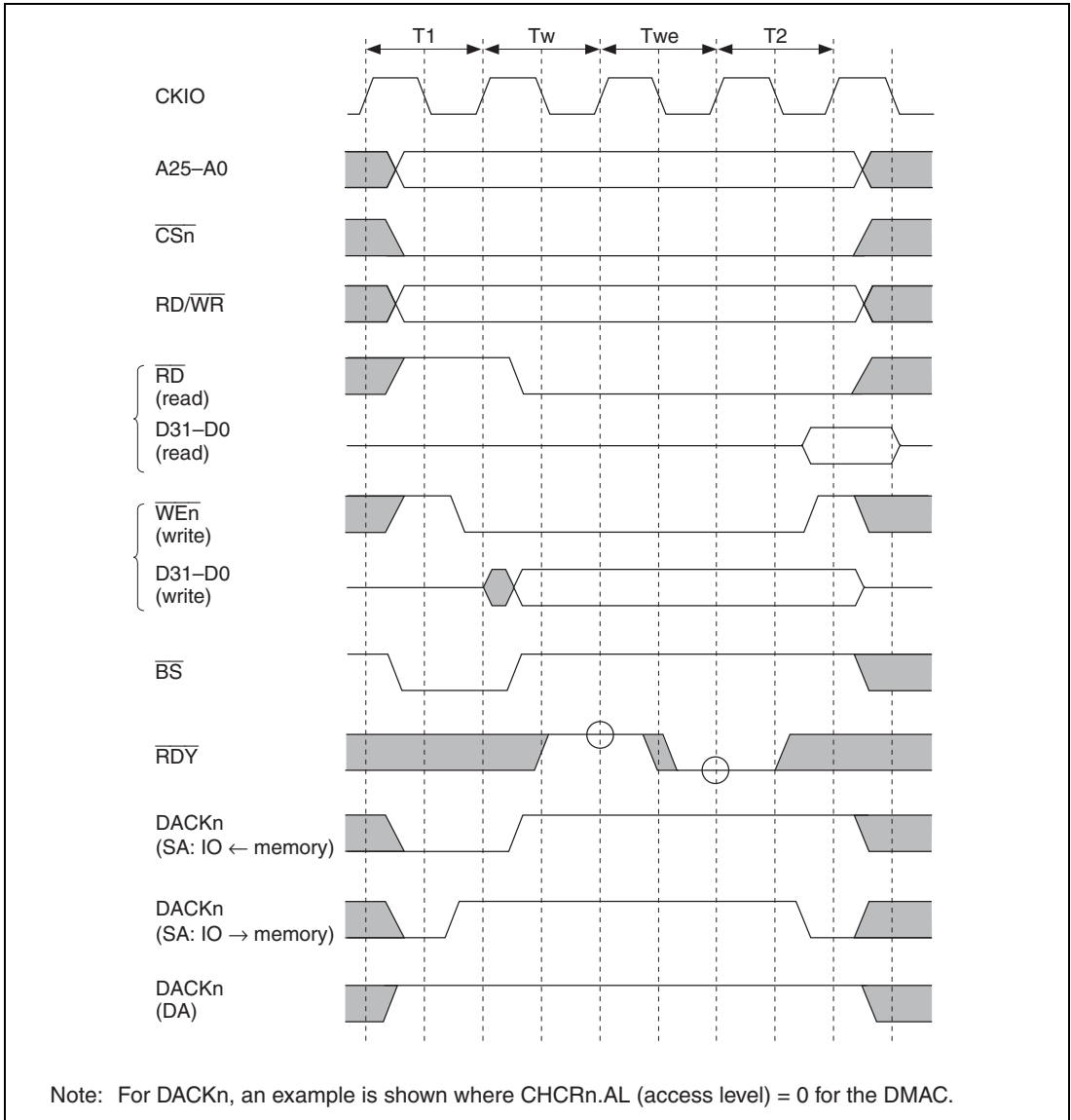


Figure 13.11 SRAM Interface Wait State Timing (Wait State Insertion by $\overline{\text{RDY}}$ Signal)

Read-Strobe Negate Timing (Setting Only Possible in the SH7751R): When the SRAM interface is used, timing for the negation of the strobe during read operations can be specified by the setting of the A1RDH and A4RDH bits of the WCR3 register. For information about this setting, see the description of the WCR3 register. When a byte control SRAM setting is made, AnRDH should be cleared to 0.

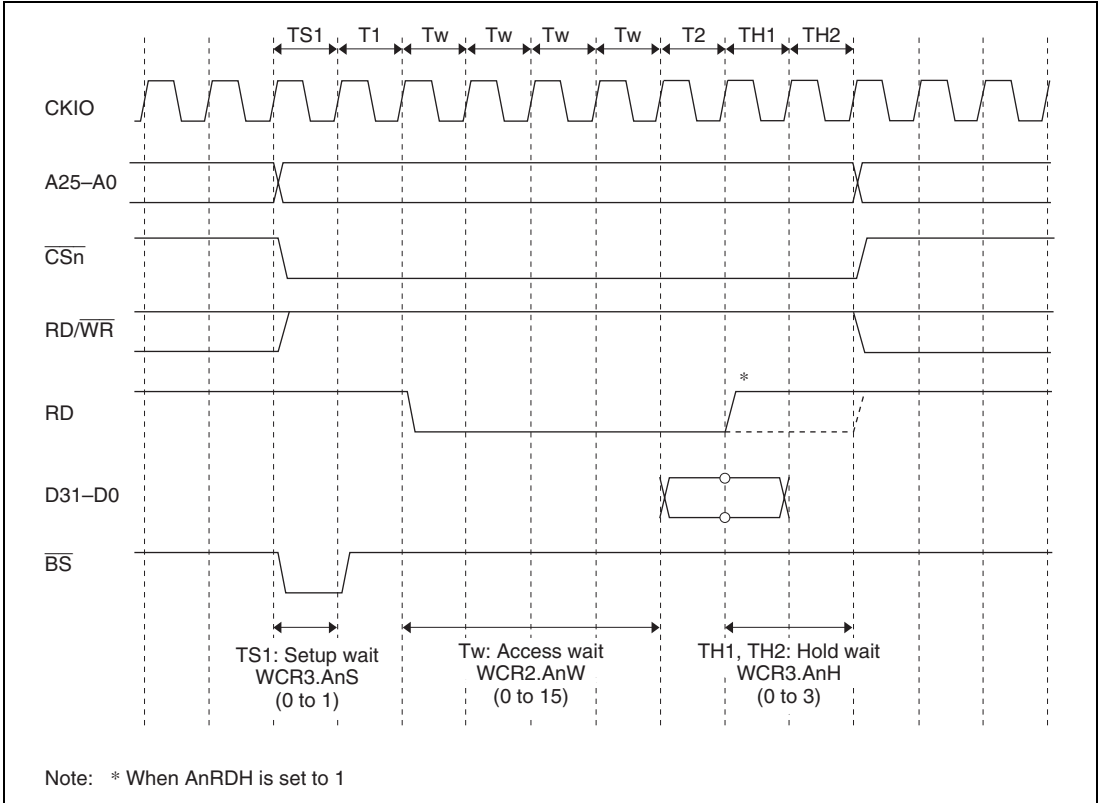


Figure 13.12 SRAM Interface Read Strobe Negate Timing
(AnS = 1, AnW = 4, and AnH = 2)

13.3.4 DRAM Interface

Direct Connection of DRAM: When the memory type bits (DRAMTP2–0) in BCR1 are set to 100, area 3 becomes DRAM interface. The DRAM interface function can then be used to connect DRAM to this LSI.

16 or 32 bits can be selected as the interface data width.

2-CAS 16-bit DRAMs can be connected, since $\overline{\text{CAS}}$ is used to control byte access.

Signals used for connection are $\overline{\text{CS3}}$, $\overline{\text{RAS}}$, $\overline{\text{CAS0}}$ to $\overline{\text{CAS3}}$, and $\text{RD}/\overline{\text{WR}}$. $\overline{\text{CAS2}}$ to $\overline{\text{CAS3}}$ are not used when the data width is 16 bits.

In addition to normal read and write access modes, fast page mode is supported for burst access. EDO mode, which enables the DRAM access time to be increased, is supported.

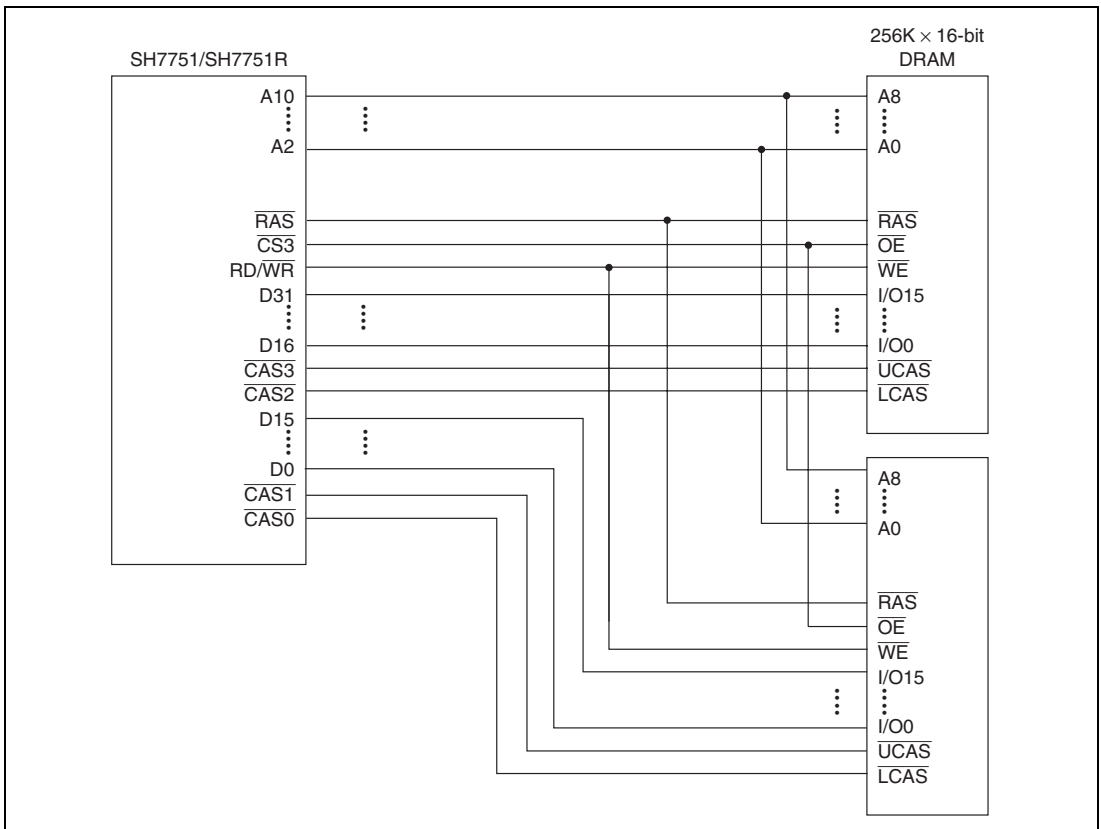


Figure 13.13 Example of DRAM Connection (32-Bit Data Width, Area 3)

Address Multiplexing: When area 3 is designated as DRAM interface, address multiplexing is always performed in accesses to DRAM. This enables DRAM, which requires row and column address multiplexing, to be connected to this LSI without using an external address multiplexer circuit. Any of the five multiplexing methods shown below can be selected, by setting bits AMXEXT and AMX2–0 in MCR. The relationship between the AMXEXT and AMX2–0 bits and address multiplexing is shown in table 13.14. The address output pins subject to address multiplexing are A17 to A1. The address signals output by pins A25 to A18 are undefined.

Table 13.14 Relationship between AMXEXT and AMX2–0 Bits and Address Multiplexing

| Setting | | | | Number of Column Address Bits | Output Timing | External Address Pins | | | | |
|----------------|------|------|---------|--|----------------|-----------------------|-----|-----|-----|-----|
| AMXEXT | AMX2 | AMX1 | AMX0 | | | A1–A13 | A14 | A15 | A16 | A17 |
| 0 | 0 | 0 | 0 | 8 bits | Column address | A1–A13 | A14 | A15 | A16 | A17 |
| | | | | | Row address | A9–A21 | A22 | A23 | A24 | A25 |
| 0 | 0 | 0 | 1 | 9 bits | Column address | A1–A13 | A14 | A15 | A16 | A17 |
| | | | | | Row address | A10–A22 | A23 | A24 | A25 | A17 |
| | | 1 | 0 | 10 bits | Column address | A1–A13 | A14 | A15 | A16 | A17 |
| | | | | | Row address | A11–A23 | A24 | A25 | A16 | A17 |
| 1 | 0 | 1 | 11 bits | Column address | A1–A13 | A14 | A15 | A16 | A17 | |
| | | | | Row address | A12–A24 | A25 | A15 | A16 | A17 | |
| 1 | 0 | 0 | 0 | 12 bits | Column address | A1–A13 | A14 | A15 | A16 | A17 |
| | | | | | Row address | A13–A25 | A14 | A15 | A16 | A17 |
| Other settings | | | | Reserved | — | — | — | — | — | — |

Basic Timing: The basic timing for DRAM access is 4 cycles. This basic timing is shown in figure 13.14. T_{pc} is the precharge cycle, T_r the \overline{RAS} assert cycle, T_{c1} the \overline{CAS} assert cycle, and T_{c2} the read data latch cycle.

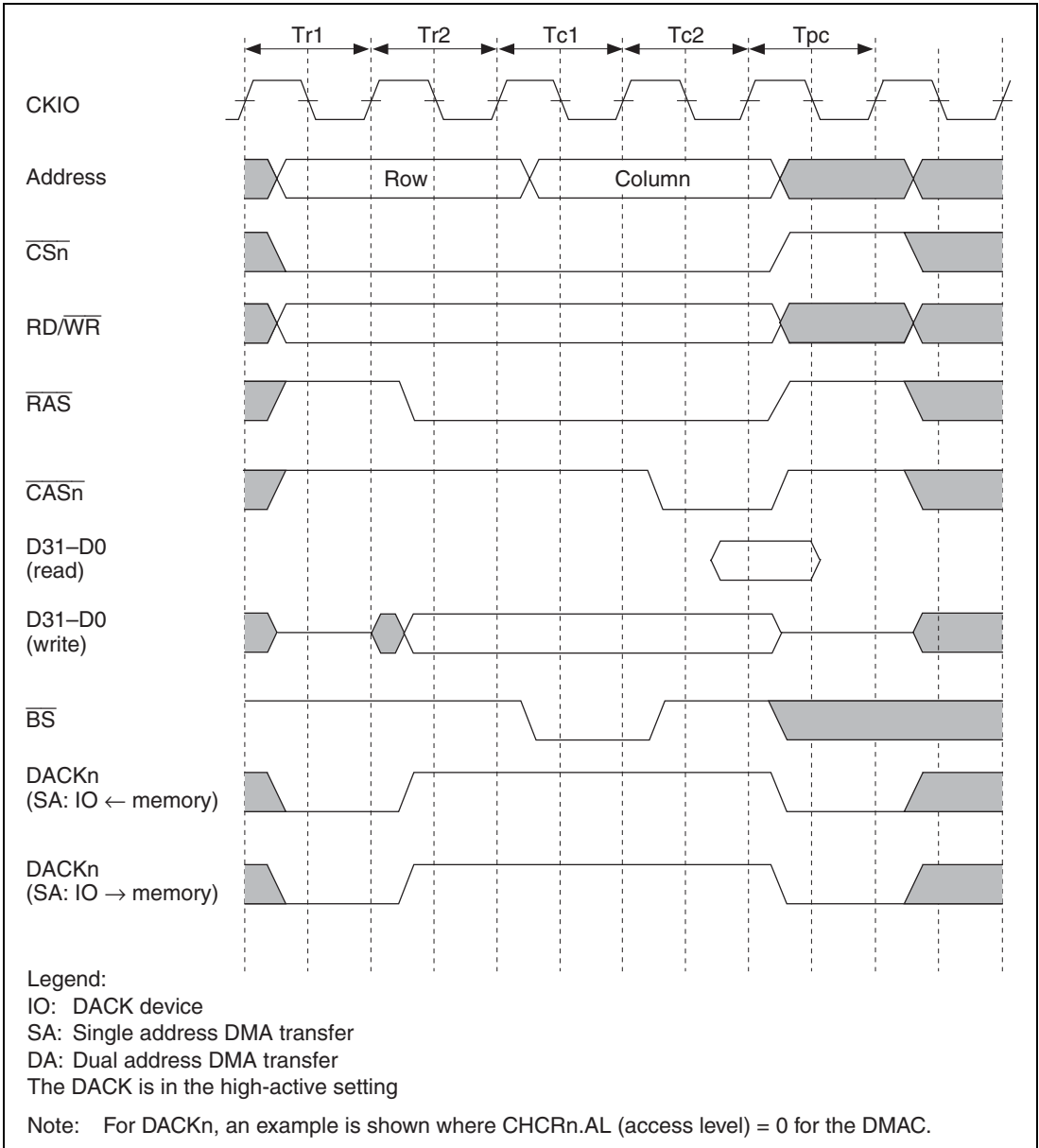


Figure 13.14 Basic DRAM Access Timing

Wait State Control: As the clock frequency increases, it becomes impossible to complete all states in one cycle as in basic access. Therefore, provision is made for state extension by using the setting bits in WCR2 and MCR. The timing with state extension using these settings is shown in figure 13.15. Additional Tpc cycles (cycles used to secure the $\overline{\text{RAS}}$ precharge time) can be inserted by means of the TPC bit in MCR, giving from 1 to 7 cycles. The number of cycles from $\overline{\text{RAS}}$ assertion to $\overline{\text{CAS}}$ assertion can be set to between 2 and 5 by inserting Trw cycles by means of the RCD bit in MCR. Also, the number of cycles from $\overline{\text{CAS}}$ assertion to the end of the access can be varied between 1 and 16 according to the setting of A3W2 to A3W0 in WCR2.

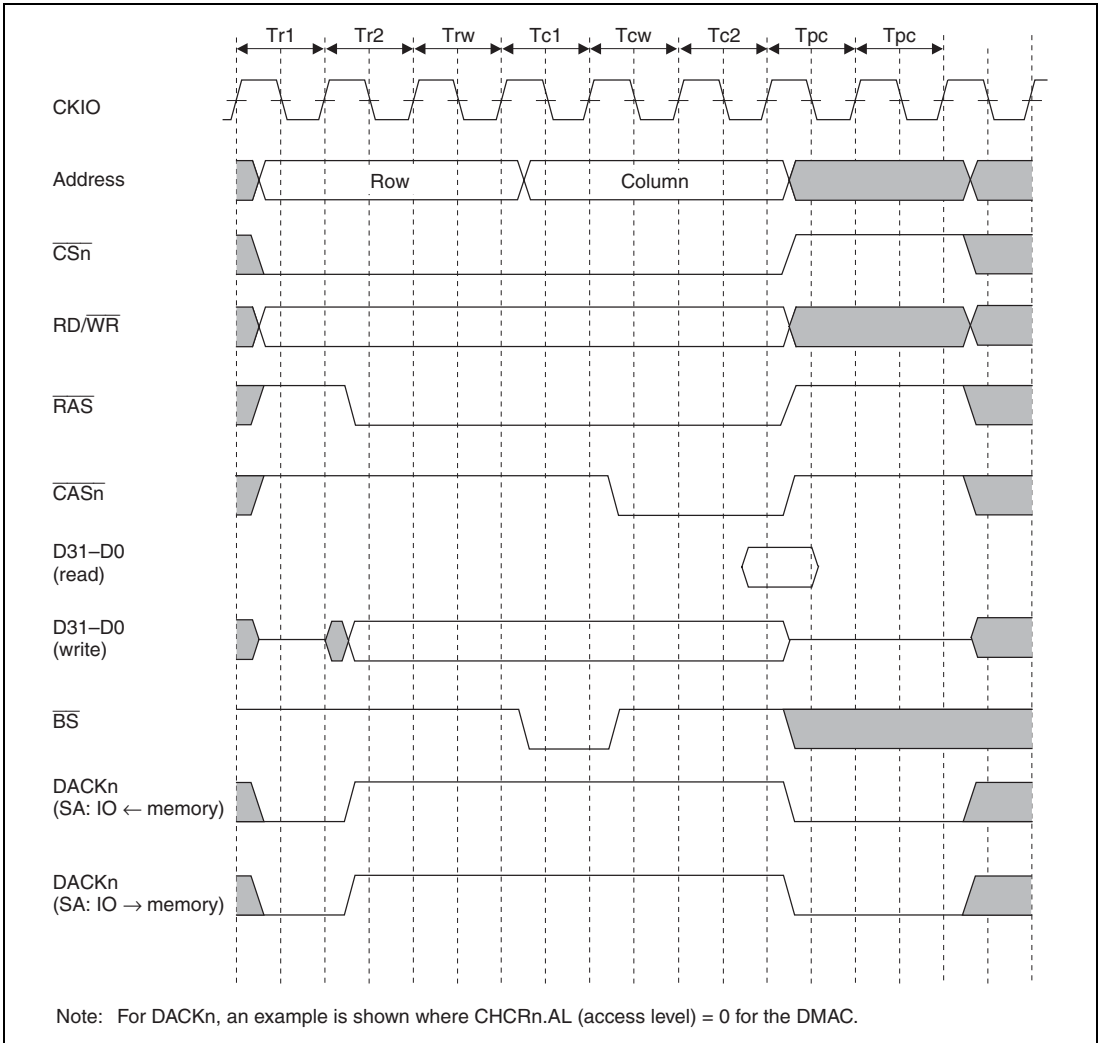


Figure 13.15 DRAM Wait State Timing

Burst Access: In addition to the normal DRAM access mode in which a row address is output in each data access, a fast page mode is also provided for the case where consecutive accesses are made to the same row. This mode allows fast access to data by outputting the row address only once, then changing only the column address for each subsequent access. Normal access or burst access using fast page mode can be selected by means of the burst enable (BE) bit in MCR. The timing for burst access using fast page mode is shown in figure 13.16.

If the access size exceeds the set bus width, burst access is performed. In a 32-byte transfer, the first access comprises a longword that includes the data requiring access. The remaining accesses are performed on 32-byte boundary data that includes the relevant data. In burst transfer, wraparound writing is performed for 32-byte data.

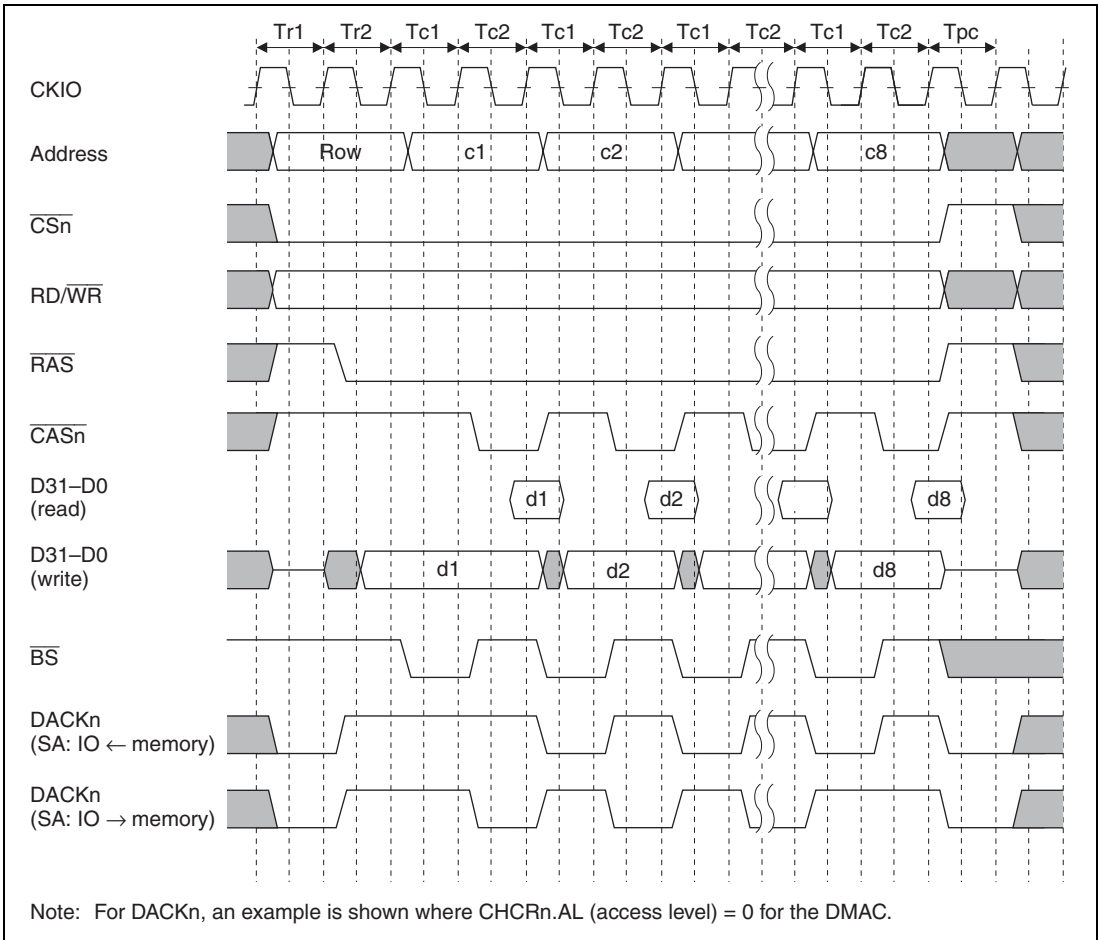


Figure 13.16 DRAM Burst Access Timing

EDO Mode: With DRAM, in addition to the mode in which data is output to the data bus only while the $\overline{\text{CAS}}$ signal is asserted in a data read cycle, an EDO (extended data out) mode is also provided in which, once the $\overline{\text{CAS}}$ signal is asserted while the $\overline{\text{RAS}}$ signal is asserted, even if the $\overline{\text{CAS}}$ signal is negated, data is output to the data bus until the $\overline{\text{CAS}}$ signal is next asserted. In this LSI, the EDO mode bit (EDOMODE) in MCR enables either normal access/burst access using fast page mode, or EDO mode normal access/burst access, to be selected for DRAM. When EDO mode is set, BE must be set to 1 in MCR. EDO mode normal access is shown in figure 13.17, and burst access in figure 13.18.

CAS Negation Period: The CAS negation period can be set to 1 or 2 by means of the TCAS bit in the MCR register.

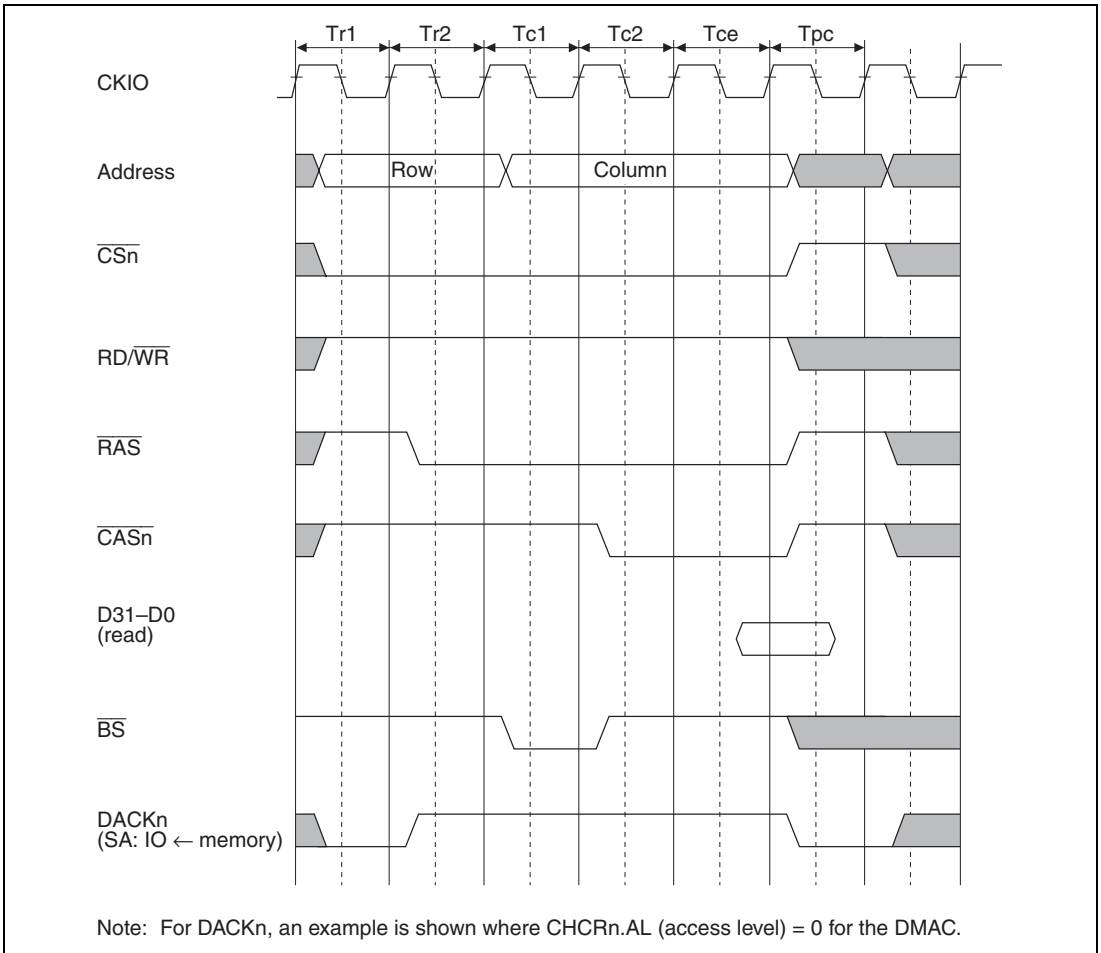


Figure 13.17 DRAM Bus Cycle (EDO Mode, RCD = 0, AnW = 0, TPC = 1)

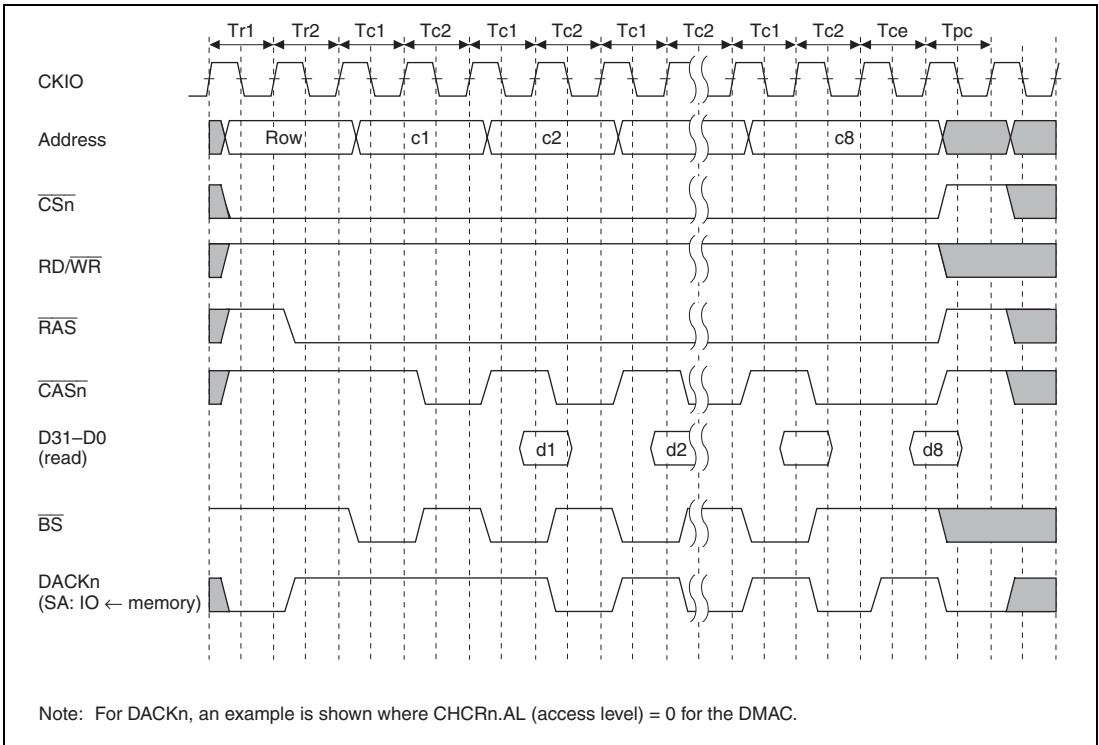
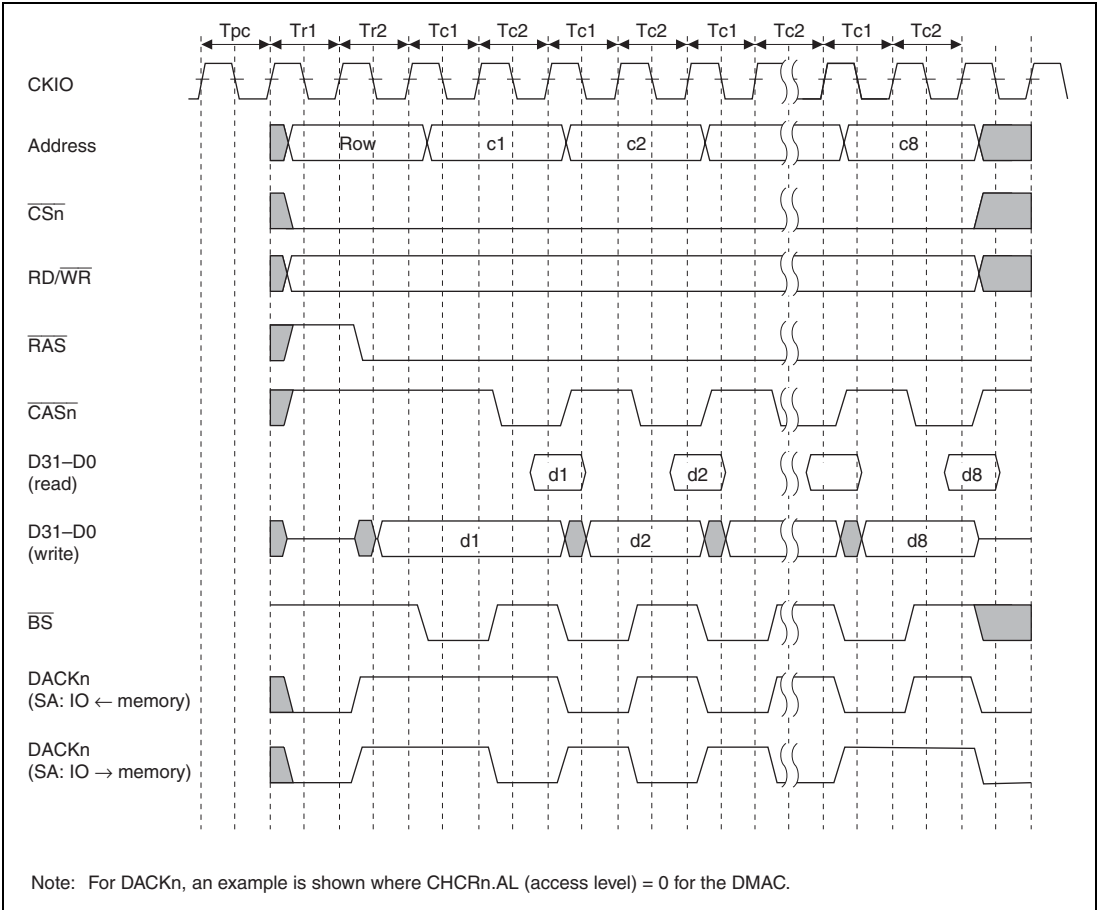


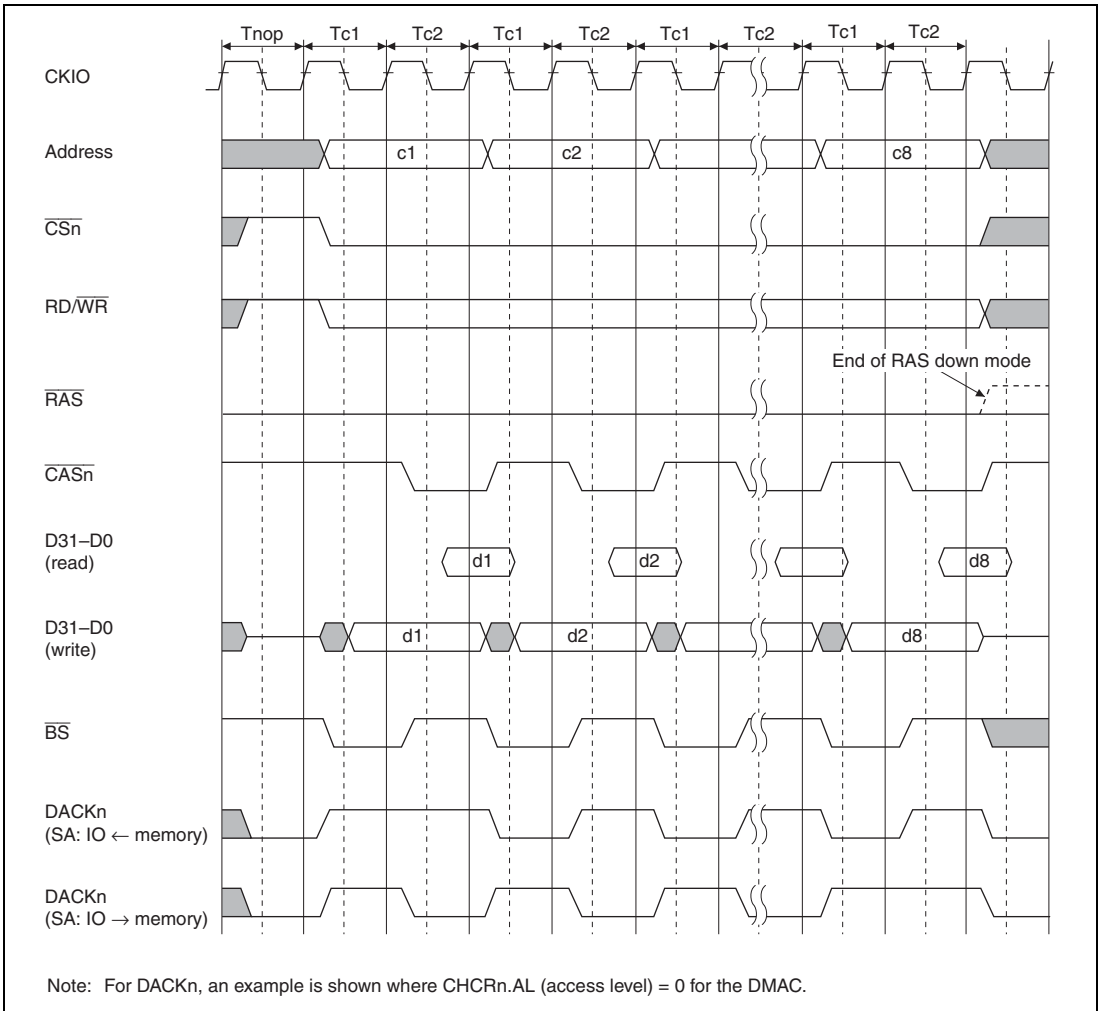
Figure 13.18 Burst Access Timing in DRAM EDO Mode

RAS Down Mode: This LSI has an address comparator for detecting row address matches in burst mode. By using this address comparator, and also setting RAS down mode specification bit RASD to 1, it is possible to select RAS down mode, in which $\overline{\text{RAS}}$ remains asserted after the end of an access. When RAS down mode is used, if the refresh cycle is longer than the maximum DRAM $\overline{\text{RAS}}$ assert time, the refresh cycle must be decreased to or below the maximum value of t_{RAS} .

In RAS down mode, in the event of an access to an address with a different row address, an access to a different area, a refresh request, or a bus release request, $\overline{\text{RAS}}$ is negated and the necessary operation is performed. When DRAM access is resumed after this, since this is the start of RAS down mode, the operation starts with row address output. Timing charts are shown in figures 13.19 (1), (2), (3), and (4).



**Figure 13.19 (1) DRAM Burst Bus Cycle, RAS Down Mode Start
(Fast Page Mode, RCD = 0, AnW = 0)**



**Figure 13.19 (2) DRAM Burst Bus Cycle, RAS Down Mode Continuation
(Fast Page Mode, RCD = 0, AnW = 0)**

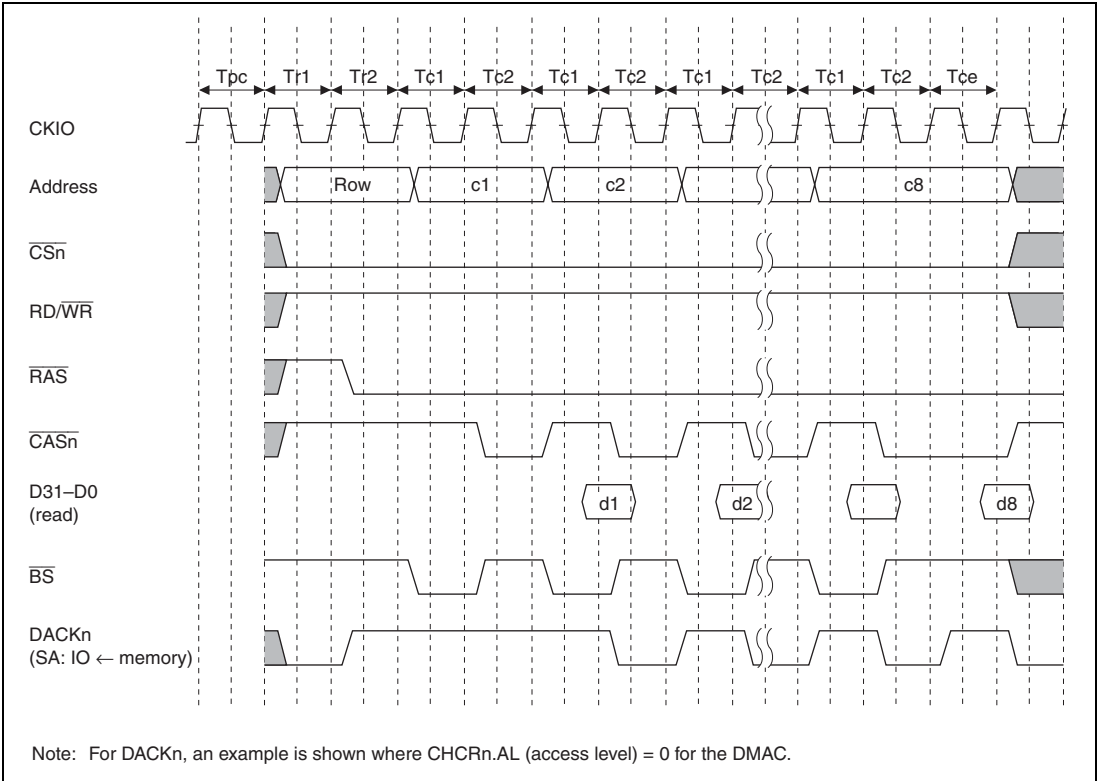
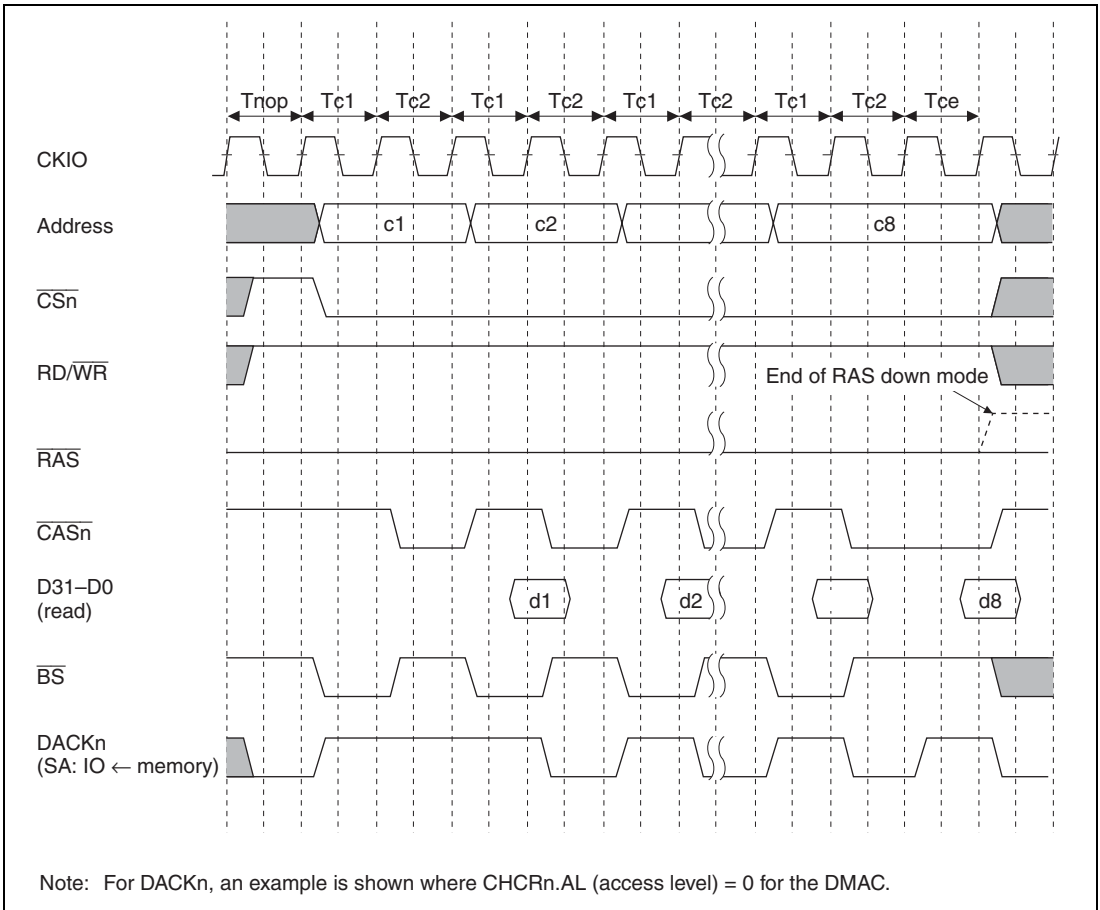


Figure 13.19 (3) DRAM Burst Bus Cycle, RAS Down Mode Start (EDO Mode, RCD = 0, AnW = 0)



**Figure 13.19 (4) DRAM Burst Bus Cycle, RAS Down Mode Continuation
(EDO Mode, RCD = 0, AnW = 0)**

Refresh: The bus state controller includes a function for controlling DRAM refreshing. Distributed refreshing using a CAS-before-RAS cycle can be performed for DRAM by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in MCR. Self-refresh mode is also supported.

- CAS-before-RAS Refresh

When CAS-before-RAS refresh cycles are executed, refreshing is performed at intervals determined by the input clock selected by bits CKS2–CKS0 in RTCSR, and the value set in RTCOR. The value of bits CKS2–CKS0 in RTCOR should be set so as to satisfy the specification for the DRAM refresh interval. First make the settings for RTCOR, RTCNT, and the RMODE and RFSH bits in MCR, then make the CKS2–CKS0 setting. When the clock is selected by CKS2–CKS0, RTCNT starts counting up from the value at that time. The RTCNT value is constantly compared with the RTCOR value, and if the two values are the same, a refresh request is generated and the $\overline{\text{BACK}}$ pin goes high. If this LSI external bus can be used, CAS-before-RAS refreshing is performed. At the same time, RTCNT is cleared to zero and the count-up is restarted. Figure 13.20 shows the operation of CAS-before-RAS refreshing.

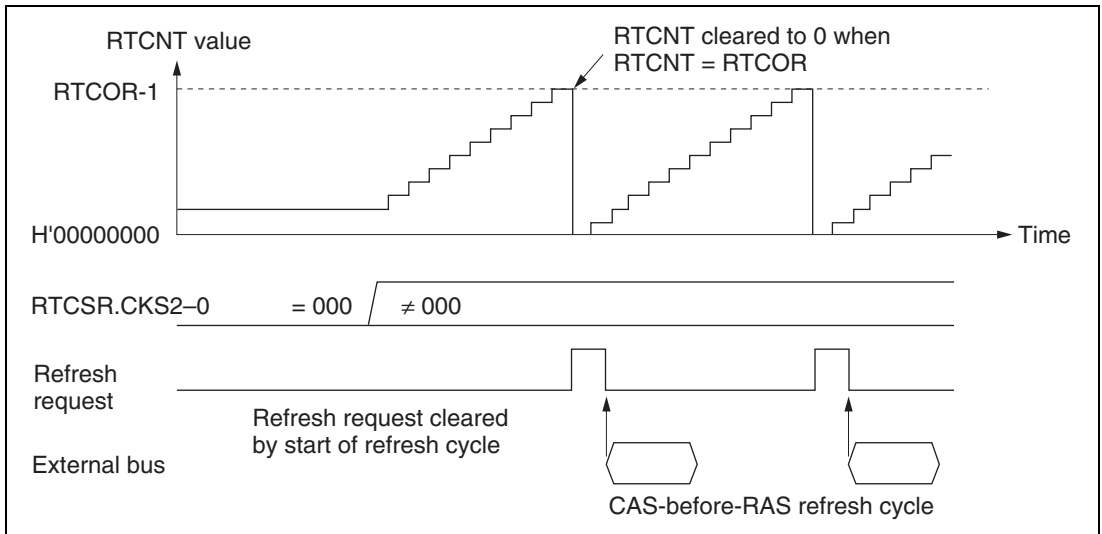


Figure 13.20 CAS-Before-RAS Refresh Operation

Figure 13.21 shows the timing of the CAS-before-RAS refresh cycle.

The number of RAS assert cycles in the refresh cycle is specified by bits TRAS2–TRAS0 in MCR. The specification of the RAS precharge time in the refresh cycle is determined by the setting of bits TRC2–TRC0 in MCR.

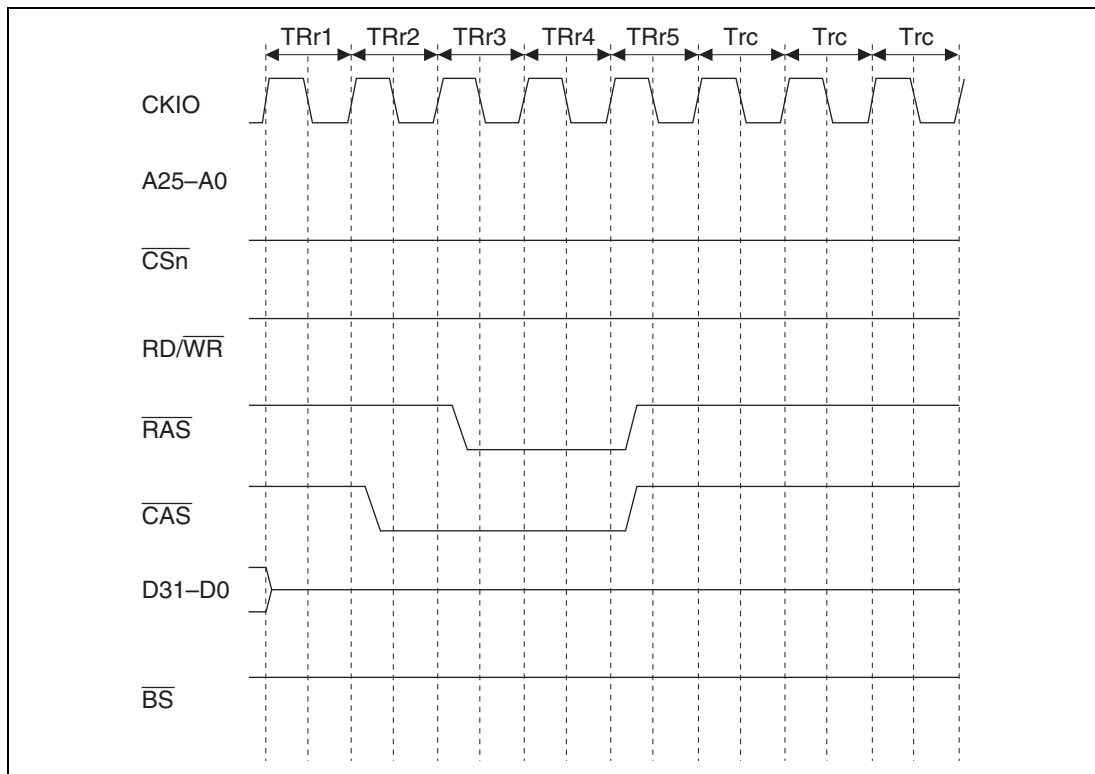


Figure 13.21 DRAM CAS-Before-RAS Refresh Cycle Timing (TRAS = 0, TRC = 1)

- Self-Refresh

The self-refreshing supported by this LSI is shown in figure 13.22.

After the self-refresh is cleared, the refresh controller immediately generates a refresh request. The RAS precharge time immediately after the end of the self-refreshing can be set by bits TRC2–TRC0 in MCR.

CAS-before-RAS refreshing is performed in normal operation, in sleep mode, and in the case of a manual reset.

Self-refreshing is performed in normal operation, in sleep mode, in standby mode, and in the case of a manual reset.

When the bus has been released in response to a bus arbitration request, or when a transition is made to standby mode, signals generally become high-impedance, but whether the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals become high-impedance or continue to be output can be controlled by the HIZCNT bit in BCR1. This enables the DRAM to be kept in the self-refreshing state.

- Relationship between Refresh Requests and Bus Cycle Requests

If a refresh request is generated during execution of a bus cycle, execution of the refresh is deferred until the bus cycle is completed. Refresh operations are deferred during multiple bus cycles generated because the data bus width is smaller than the access size (for example, when performing longword access to 8-bit bus width memory) and during a 32-byte transfer such as a cache fill or write-back, and also between read and write cycles during execution of a TAS instruction, and between read and write cycles when DMAC dual address transfer is executed. If a refresh request occurs when the bus has been released by the bus arbiter, refresh execution is deferred until the bus is acquired. If a match between RTCNT and RTCOR occurs while a refresh is waiting to be executed, so that a new refresh request is generated, the previous refresh request is eliminated. In order for refreshing to be performed normally, care must be taken to ensure that no bus cycle or bus mastership occurs that is longer than the refresh interval. When a refresh request is generated, the $\overline{\text{BACK}}$ pin is negated (driven high). Therefore, normal refreshing can be performed by having the $\overline{\text{BACK}}$ pin monitored by a bus master other than this LSI requesting the bus, or the bus arbiter, and returning the bus to this LSI.

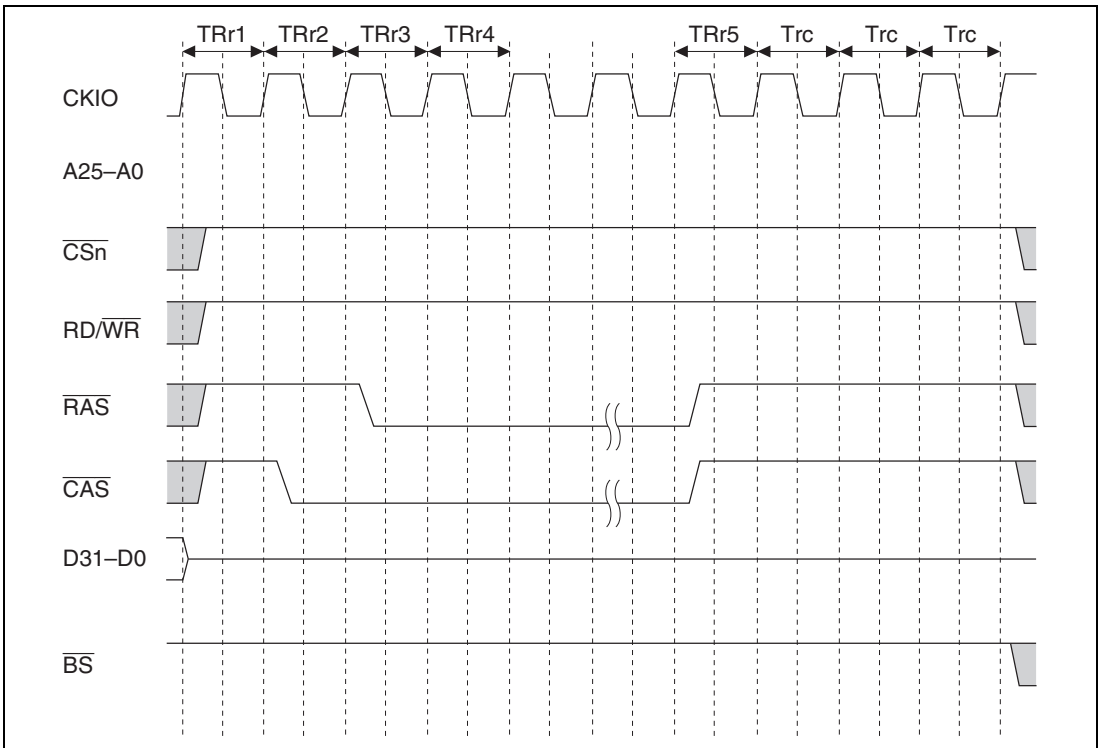


Figure 13.22 DRAM Self-Refresh Cycle Timing

Power-On Sequence: Regarding use of DRAM after powering on, it is requested that a wait time (at least 100 μ s or 200 μ s) during which no access can be performed be provided, followed by at least the prescribed number (usually 8) of dummy CAS-before-RAS refresh cycles. As the bus state controller does not perform any special operations for a power-on reset, the necessary power-on sequence must be carried out by the initialization program executed after a power-on reset.

13.3.5 Synchronous DRAM Interface

Direct Connection of Synchronous DRAM: Since synchronous DRAM can be selected by the $\overline{\text{CS}}$ signal, it can be connected to external memory space areas 2 and 3 using $\overline{\text{RAS}}$ and other control signals in common. If the memory type bits (DRAMTP2–0) in BCR1 are set to 010, area 3 is synchronous DRAM interface; if set to 011, areas 2 and 3 are both synchronous DRAM interface.

This LSI supports burst read and burst write operations with a burst length of 4 as a synchronous DRAM operating mode. The data bus width is 32 bit, and the SZ size bits in MCR must be set to 11. The burst enable bit (BE) in MCR is ignored, a 32-byte burst transfer is performed in a cache fill/copy-back cycle. In write-through area write operations and non-cacheable area read/write operations, 16-byte data is read even in a single read because accessing synchronous DRAM is by burst-length 4 burst read/write operations. 16-byte data transfer is also performed in a single write, but DQMn is not asserted when unnecessary data is transferred.

In the SH7751R, an 8-burst-length burst read/burst write mode is also supported as a synchronous DRAM operating mode. The data bus width is 32 bits, and the SZ size bits in MCR must be set to 11. Burst enable bit BE in MCR is ignored, and a 32-byte burst transfer is performed in a cache fill/copy-back cycle. For write-through area writes and non-cacheable area reads/writes, synchronous DRAM is accessed with an 8-burst-length burst read/write, and therefore 32 bytes of data are read even in the case of a single read. In the case of a single write, 32-byte data transfer is performed but DQMn is not asserted in the case of an unnecessary data transfer. For a description of the case where an 8-burst-length setting is made, see section 13.3.6, Burst ROM Interface. For information on the burst length, see section 13.2.10, Synchronous DRAM Mode Register (SDMR), and section 13.3.5, Power-On Sequence.

The control signals for connection of synchronous DRAM are $\overline{\text{RAS}}$, $\overline{\text{CASS}}$, $\text{RD}/\overline{\text{WR}}$, $\overline{\text{CS2}}$ or $\overline{\text{CS3}}$, DQM0 to DQM3, and CKE. All the signals other than $\overline{\text{CS2}}$ and $\overline{\text{CS3}}$ are common to all areas, and signals other than CKE are valid and latched only when $\overline{\text{CS2}}$ or $\overline{\text{CS3}}$ is asserted. Synchronous DRAM can therefore be connected in parallel to a number of areas. CKE is negated (driven low) when the frequency is changed, when the clock is unstable after the clock supply is stopped and restarted, or when self-refreshing is performed, and is always asserted (high) at other times.

Commands for synchronous DRAM are specified by $\overline{\text{RAS}}$, $\overline{\text{CASS}}$, $\overline{\text{RD/WR}}$, and specific address signals. The commands are NOP, auto-refresh (REF), self-refresh (SELF), precharge all banks (PALL), precharge specified bank (PRE), row address strobe bank active (ACTV), read (READ), read with precharge (READA), write (WRIT), write with precharge (WRITA), and mode register setting (MRS).

Byte specification is performed by DQM0 to DQM3. A read/write is performed for the byte for which the corresponding DQM signal is low. When the bus width is 32 bits, in big-endian mode DQM3 specifies an access to address $4n$, and DQM0 specifies an access to address $4n + 3$. In little-endian mode, DQM3 specifies an access to address $4n + 3$, and DQM0 specifies an access to address $4n$.

Figure 13.23 shows examples of the connection of $16\text{M} \times 16\text{-bit}$ synchronous DRAMs.

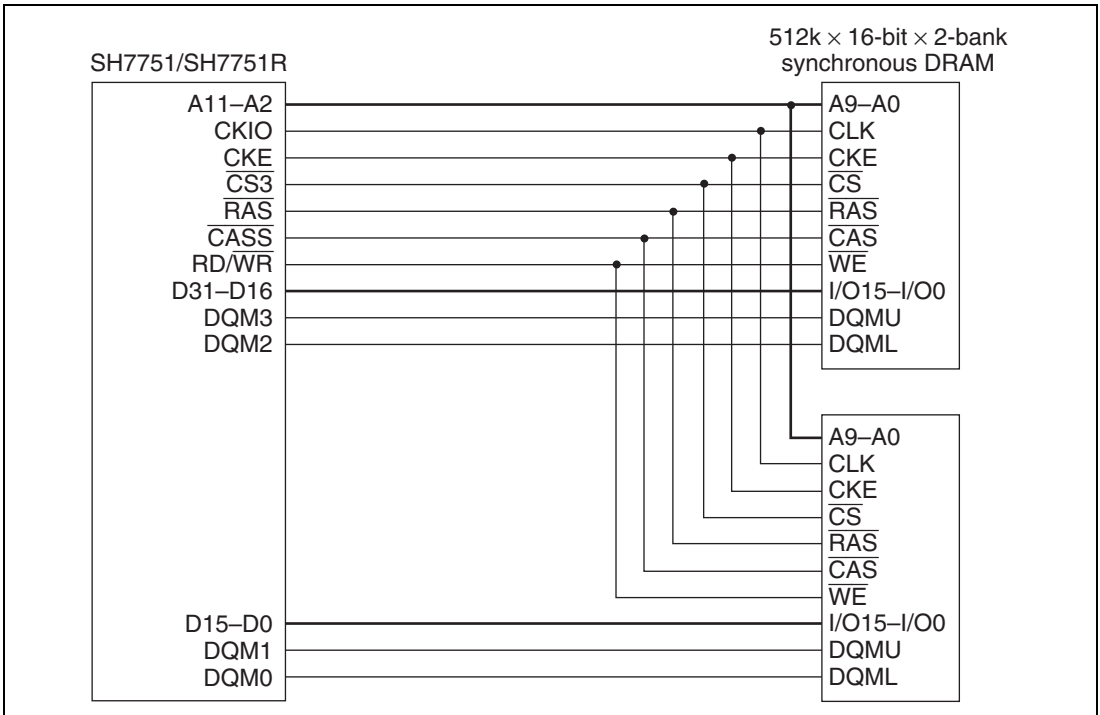


Figure 13.23 Example of 32-Bit Data Width Synchronous DRAM Connection (Area 3)

Address Multiplexing: Synchronous DRAM can be connected without external multiplexing circuitry in accordance with the address multiplex specification bits AMXEXT and AMX2–AMX0 in MCR. Table 13.15 shows the relationship between the address multiplex specification bits and the bits output at the address pins. See Appendix E, Synchronous DRAM Address Multiplexing Tables.

The address signals output at address pins A25–A18, A1, and A0 are not guaranteed.

When A0, the LSB of the synchronous DRAM address, is connected to this LSI, it makes a longword address specification. Connection should therefore be made in this order: connect pin A0 of the synchronous DRAM to pin A2 of this LSI, then connect pin A1 to pin A3.

Table 13.15 Example of Correspondence between LSI and Synchronous DRAM Address Pins (32-Bit Bus Width, AMX2–AMX0 = 000, AMXEXT = 0)

| | LSI Address Pin | | Synchronous DRAM Address Pin | |
|-----|-----------------|-----------|------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | Function |
| A13 | A21 | A21 | A11 | BANK select bank address |
| A12 | A20 | H/L | A10 | Address precharge setting |
| A11 | A19 | 0 | A9 | Address |
| A10 | A18 | 0 | A8 | |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | Not used | Not used | |
| A0 | Not used | Not used | Not used | |

Burst Read: The timing chart for a burst read is shown in figure 13.24. In the following example it is assumed that two 512k × 16-bit × 2-bank synchronous DRAMs are connected, and a 32-bit data width is used. The burst length is 4. After the Tr cycle in which the ACTV command is output, a READ command is issued in the Tc1 cycle and, 4 cycles after that, a READA command is issued and read data is fetched on the rising edge of the external command clock (CKIO) from cycle Td1 to cycle Td8. The Tpc cycle is used to wait for completion of auto-precharge based on

the READA command inside the synchronous DRAM; no new access command can be issued to the same bank during this cycle. In this LSI, the number of Tpc cycles is determined by the specification of bits TPC2–TPC0 in MCR, and commands are not issued for the synchronous DRAM during this interval.

The example in figure 13.24 shows the basic cycle. To connect slower synchronous DRAM, the cycle can be extended by setting WCR2 and MCR bits. The number of cycles from the ACTV command output cycle, Tr, to the READ command output cycle, Tc1, can be specified by bits RCD1 and RCD0 in MCR, with a value of 0 to 3 specifying 2 to 4 cycles, respectively. In the case of 2 or more cycles, a Trw cycle, in which an NOP command is issued for the synchronous DRAM, is inserted between the Tr cycle and the Tc cycle. The number of cycles from READ command output cycle Tc1 to the first read data latch cycle, Td1, can be specified as 1 to 5 cycles independently for areas 2 and 3 by means of bits A2W2–A2W0 and A3W2–A3W0 in WCR2. This number of cycles corresponds to the number of synchronous DRAM CAS latency cycles.

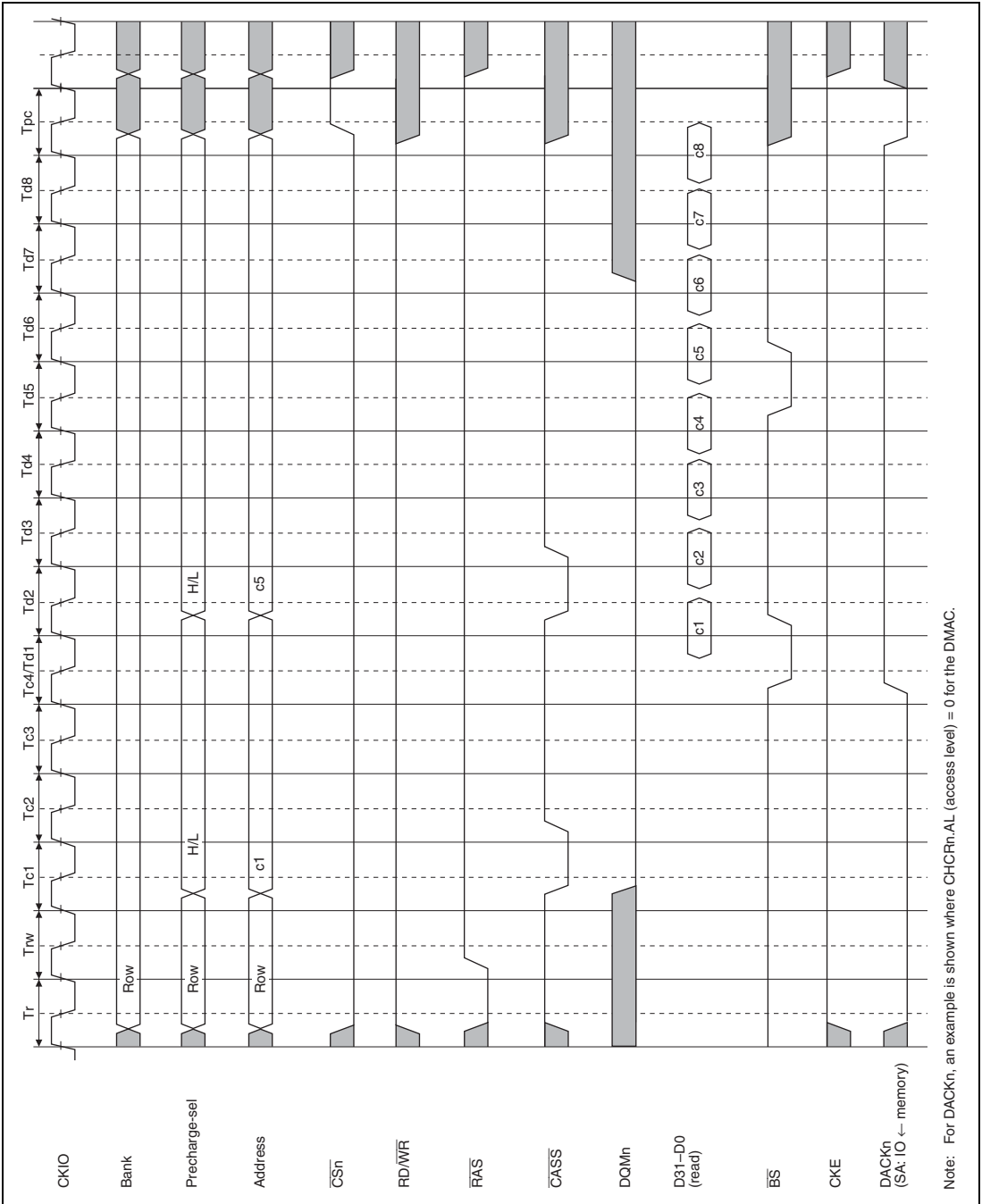


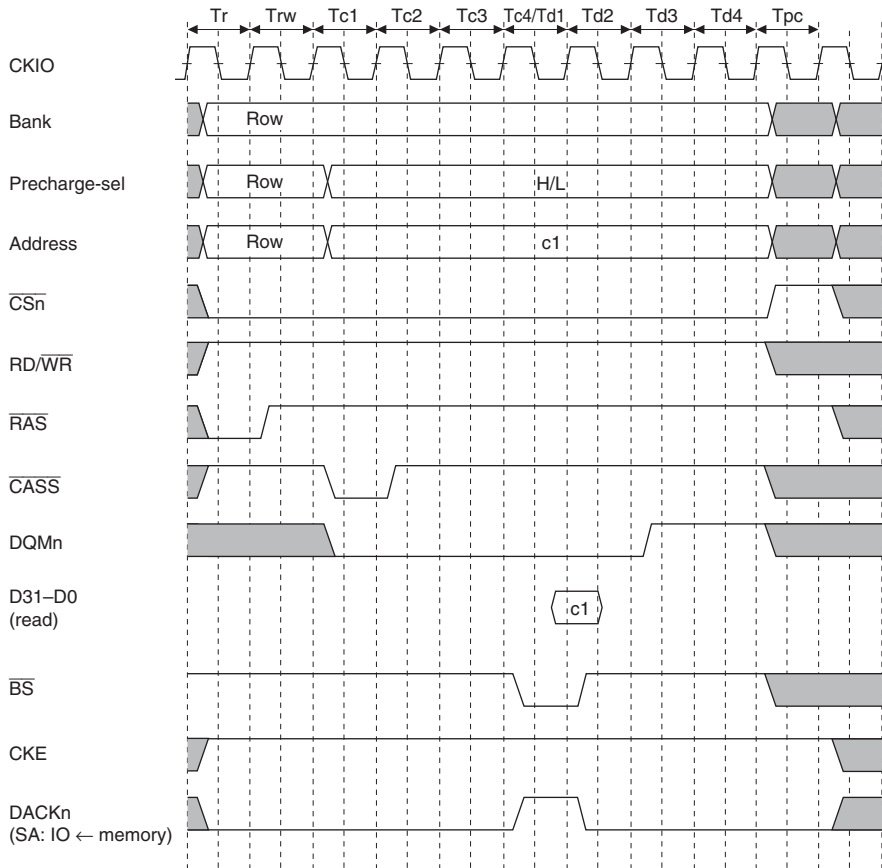
Figure 13.24 Basic Timing for Synchronous DRAM Burst Read

In a synchronous DRAM cycle, the \overline{BS} signal is asserted for one cycle at the beginning of each data transfer cycle that is in response to a READ or READA command. Data are accessed in the following sequence: in the fill operation for a cache miss, the data between 64-bit boundaries that include the missing data are first read by the initial READ command; after that, the data between 16-bit boundaries data that include the missing data are read in a wraparound way. The subsequently issued READA command reads the 16 bytes of data, which is the remainder of the data between 32-byte boundaries.

Single Read: With this LSI, as synchronous DRAM is set to burst read/burst write mode, read data output continues after the required data has been read. To prevent data collisions, after the required data is read in Td1, empty read cycles Td2 to Td4 are performed, and this LSI waits for the end of the synchronous DRAM operation. The \overline{BS} signal is asserted only in Td1.

There are 4 burst transfers in a read. In cache-through and other DMA read cycles, of cycles Td1 to Td4.

Since such empty cycles increase the memory access time, and tend to reduce program execution speed and DMA transfer speed, it is important both to avoid unnecessary cache-through area accesses, and to use a data structure that will allow data to be placed at a 32-byte boundary, and to be transferred in 32-byte units, when carrying out DMA transfer with synchronous DRAM specified as the source.

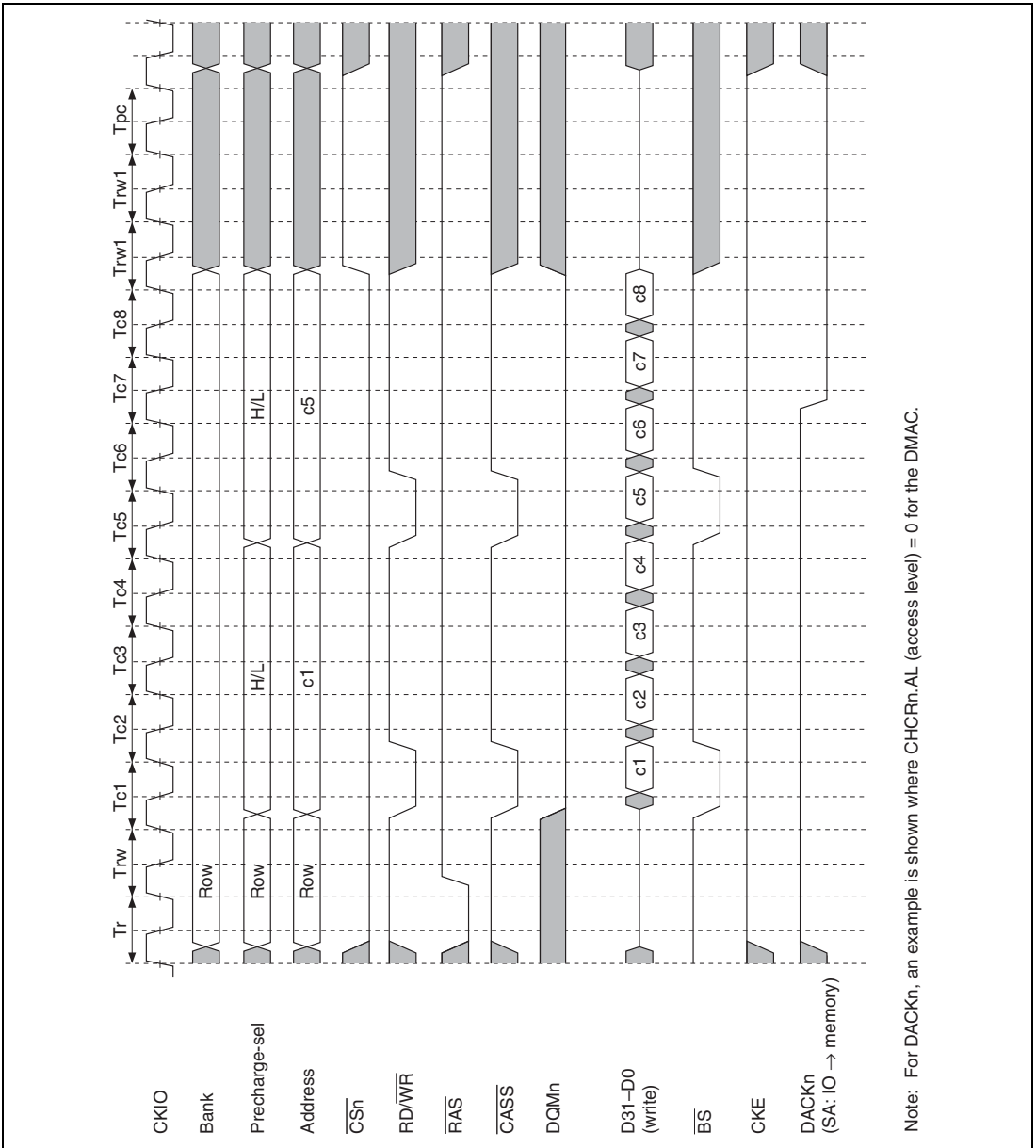


Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.25 Basic Timing for Synchronous DRAM Single Read

Burst Write: The timing chart for a burst write is shown in figure 13.26. In this LSI, a burst write occurs only in the event of 32-byte transfer. In a burst write operation, the WRIT command is issued in the Tc1 cycle following the Tr cycle in which the ACTV command is output and, 4 cycles later, the WRITA command is issued. In the write cycle, the write data is output at the same time as the write command. In the case of the write with auto-precharge command, precharging of the relevant bank is performed in the synchronous DRAM after completion of the write command, and therefore no command can be issued for the same bank until precharging is completed. Consequently, in addition to the precharge wait cycle, Tpc, used in a read access, cycle Trw1 is also added as a wait interval until precharging is started following the write command. Issuance of a new command for the synchronous DRAM is postponed during this interval. The number of

Trwl cycles can be specified by bits TRWL2–TRWL0 in MCR. Access starts from 16-byte boundary data, and 32-byte boundary data is written in wraparound mode. DACK is asserted two cycles before the data write cycle.

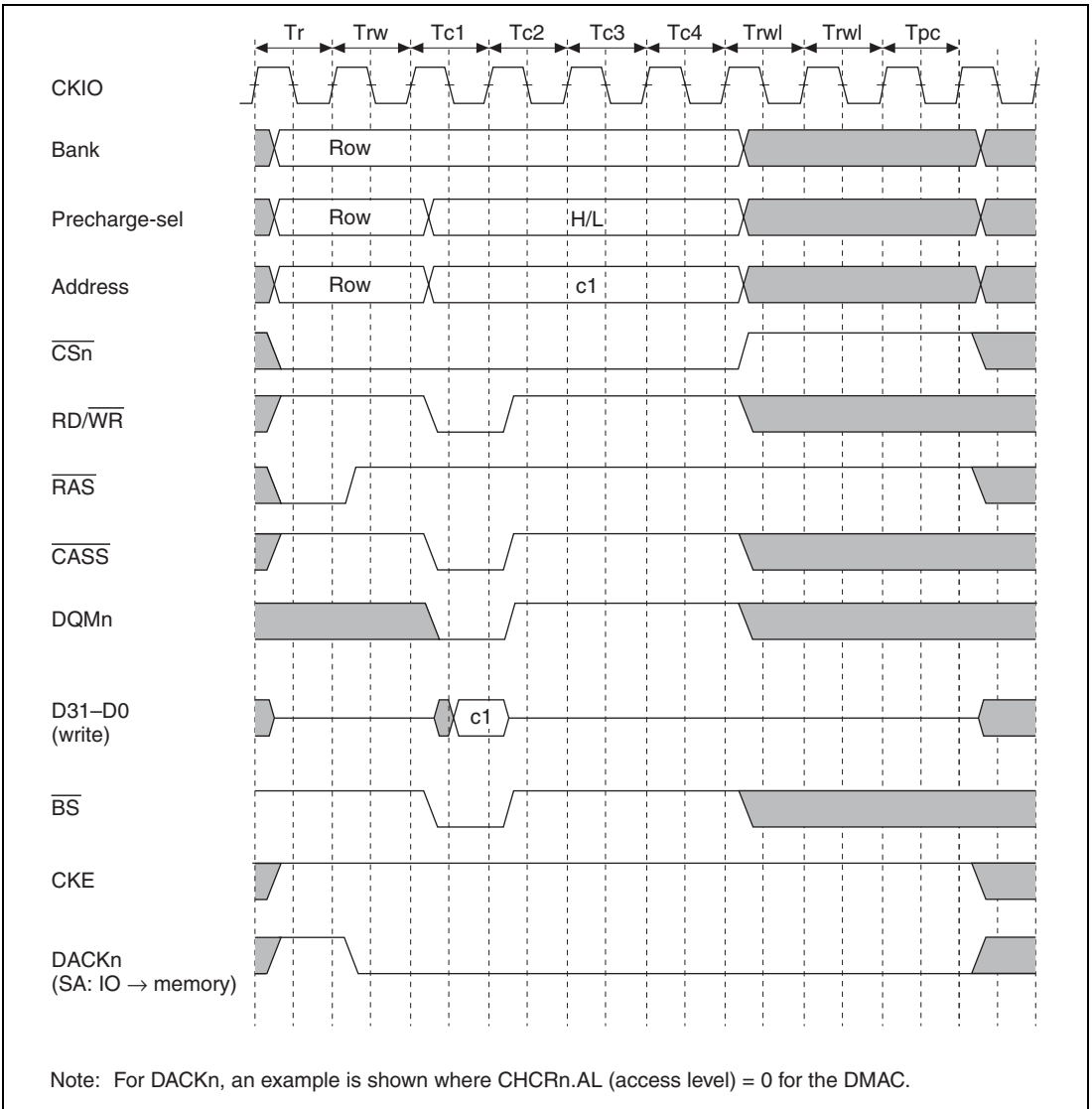


Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.26 Basic Timing for Synchronous DRAM Burst Write

Single Write: The basic timing chart for write access is shown in figure 13.27. In a single write operation, following the T_r cycle in which ACTV command output is performed, a WRITA command that performs auto-precharge is issued in the T_{c1} cycle. In the write cycle, the write data is output at the same time as the write command. In the case of a write with auto-precharge, precharging of the relevant bank is performed in the synchronous DRAM after completion of the write command, and therefore no command can be issued for the synchronous DRAM until precharging is completed. Consequently, in addition to the precharge wait cycle, T_{pc} , used in a read access, cycle $Trwl$ is also added as a wait interval until precharging is started following the write command. Issuance of a new command for the same bank is postponed during this interval. The number of $Trwl$ cycles can be specified by bits $TRWL2$ – $TRWL0$ in MCR. DACK is asserted two cycles before the data write cycle.

This LSI supports burst-length 4 burst read and burst write operations of synchronous DRAM. A wait cycle is therefore generated even with single write operations.



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.27 Basic Timing for Synchronous DRAM Single Write

RAS Down Mode: The synchronous DRAM bank function is used to support high-speed accesses to the same row address. When the RASD bit in MCR is 1, read/write command accesses are performed using commands without auto-precharge (READ, WRIT). In this case, precharging is not performed when the access ends. When accessing the same row address in the same bank, it is possible to issue the READ or WRIT command immediately, without issuing an ACTV command, in the same way as in the DRAM RAS down state. As synchronous DRAM is internally divided into two or four banks, it is possible to activate one row address in each bank. If the next access is to a different row address, a PRE command is first issued to precharge the relevant bank, then when precharging is completed, the access is performed by issuing an ACTV command followed by a READ or WRIT command. If this is followed by an access to a different row address, the access time will be longer because of the precharging performed after the access request is issued.

In a write, when auto-precharge is performed, a command cannot be issued for a period of $Trwl + Tpc$ cycles after issuance of the WRITA command. When RAS down mode is used, READ or WRIT commands can be issued successively if the row address is the same. The number of cycles can thus be reduced by $Trwl + Tpc$ cycles for each write. The number of cycles between issuance of the PRE command and the ACTV command is determined by bits TPC2–TPC0 in MCR.

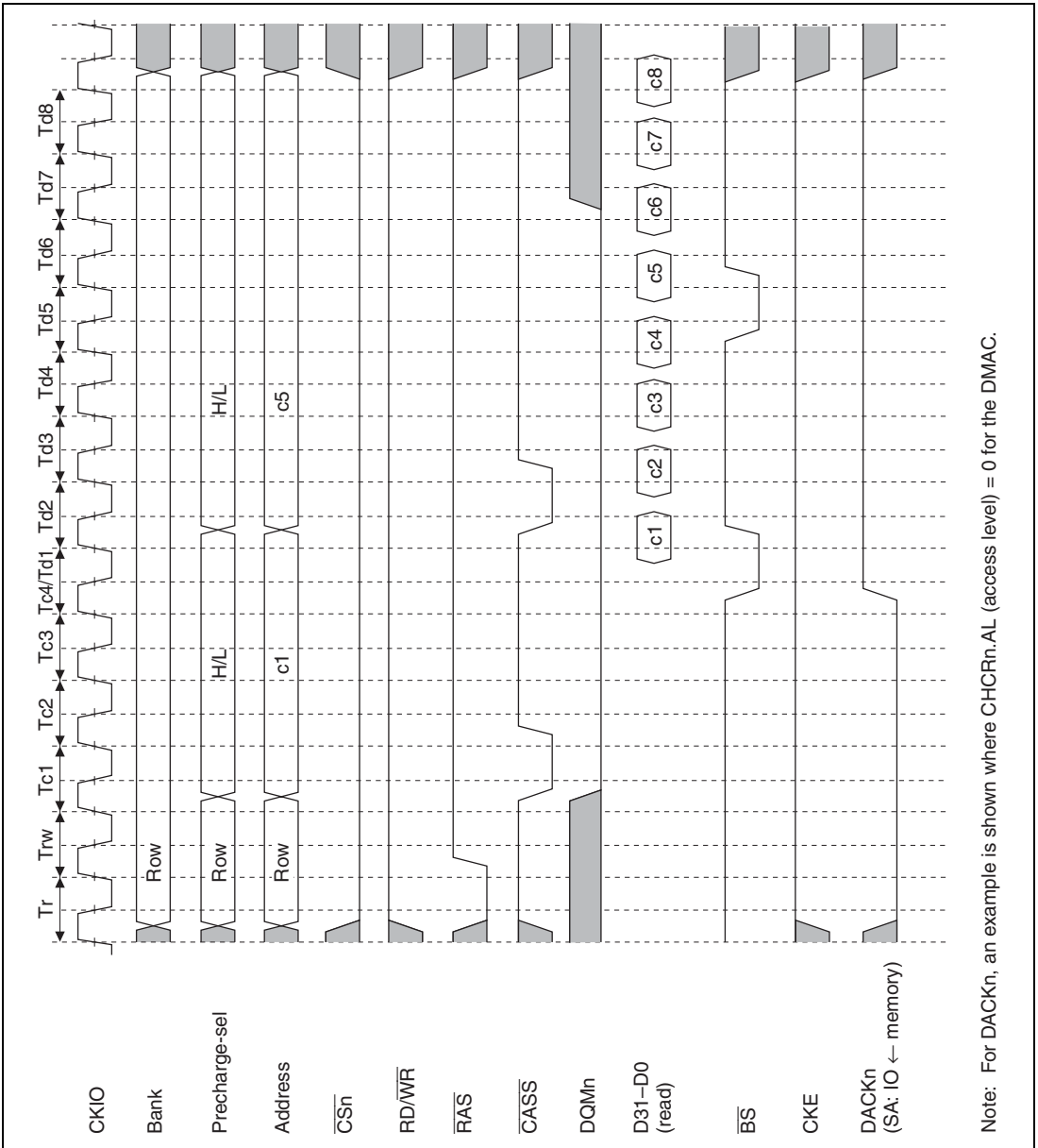
There is a limit on t_{RAS} , the time for placing each bank in the active state. If there is no guarantee that there will not be a cache hit and another row address will be accessed within the period in which this value is maintained by program execution, it is necessary to set auto-refresh and set the refresh cycle to no more than the maximum value of t_{RAS} . In this way, it is possible to observe the restrictions on the maximum active state time for each bank. If auto-refresh is not used, measures must be taken in the program to ensure that the banks do not remain active for longer than the prescribed time.

A burst read cycle without auto-precharge is shown in figure 13.28, a burst read cycle for the same row address in figure 13.29, and a burst read cycle for different row addresses in figure 13.30. Similarly, a burst write cycle without auto-precharge is shown in figure 13.31, a burst write cycle for the same row address in figure 13.32, and a burst write cycle for different row addresses in figure 13.33.

When synchronous DRAM is read, there is a 2-cycle latency for the DMQn signal that performs the byte specification. As a result, when the READ command is issued in figure 13.28, if the Tc cycle is executed immediately, the DMQn signal specification for Td1 cycle data output cannot be carried out. Therefore, the CAS latency should not be set to 1.

When RAS down mode is set, if only accesses to the respective banks in area 3 are considered, as long as accesses to the same row address continue, the operation starts with the cycle in figure 13.28 or 13.31, followed by repetition of the cycle in figure 13.29 or 13.32. An access to a different area during this time has no effect. If there is an access to a different row address in the

bank active state, after this is detected the bus cycle in figure 13.30 or 13.33 is executed instead of that in figure 13.29 or 13.32. In RAS down mode, too, a PALL command is issued before a refresh cycle or before bus release due to bus arbitration.



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.28 Burst Read Timing

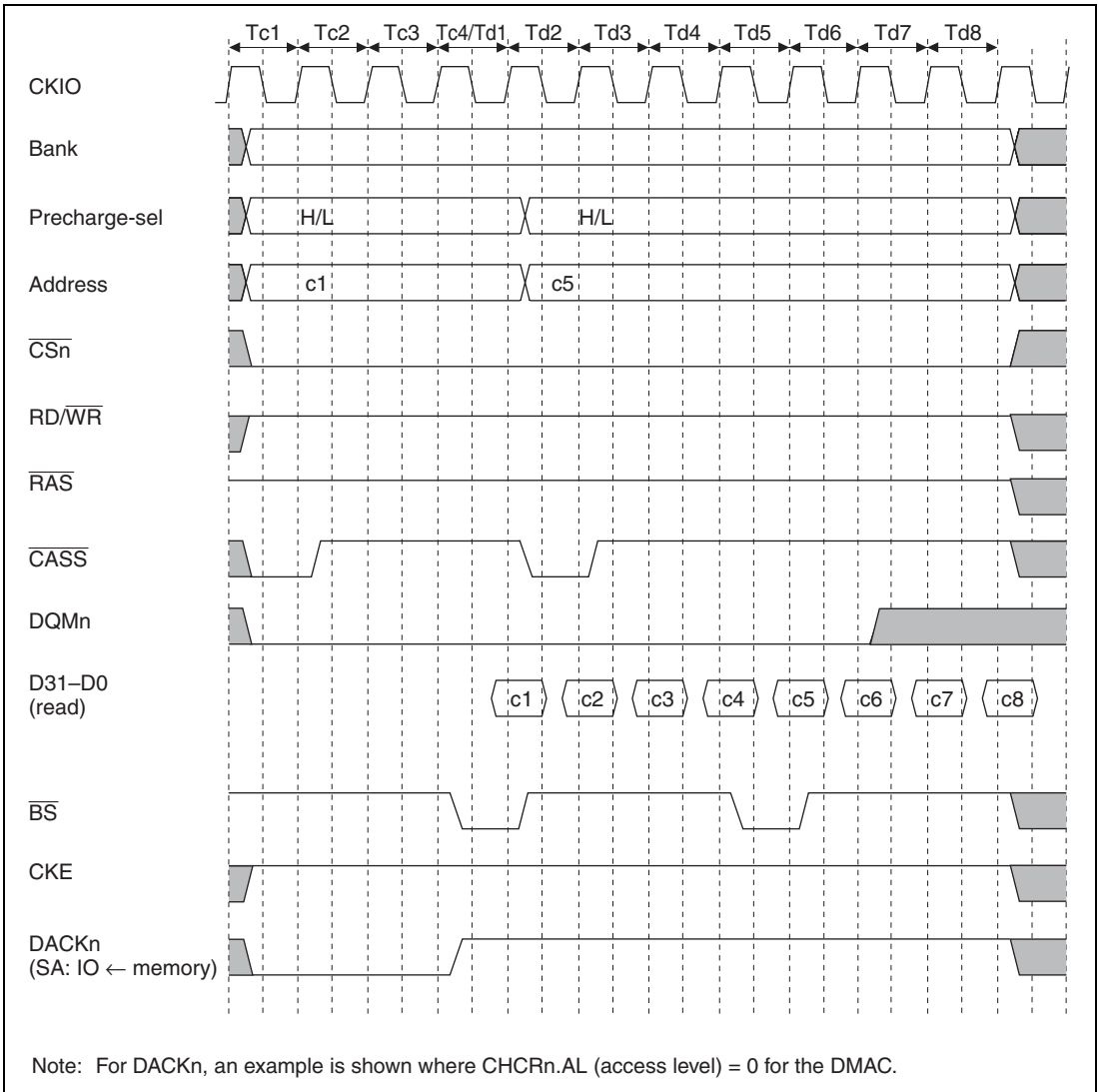


Figure 13.29 Burst Read Timing (RAS Down, Same Row Address)

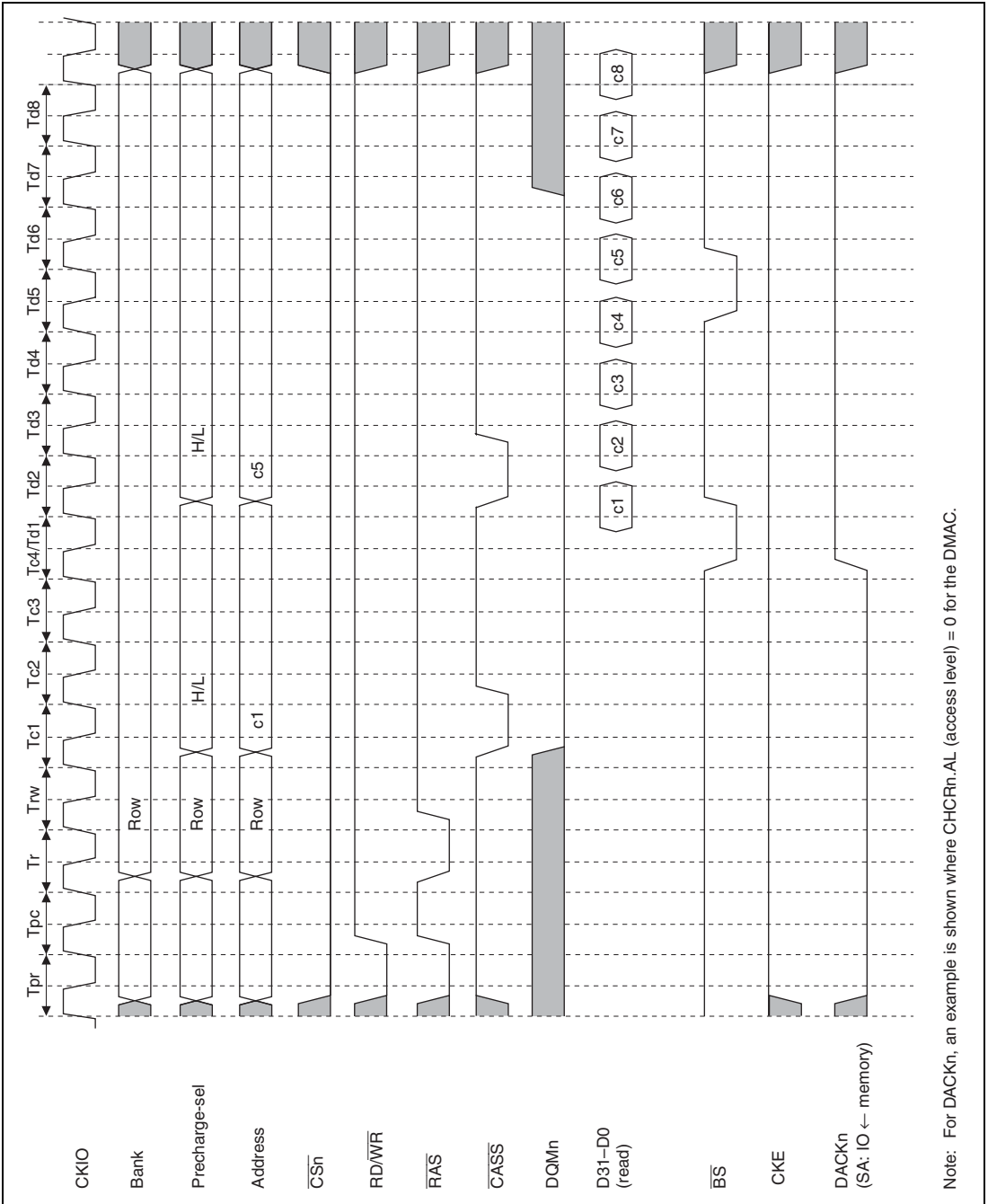
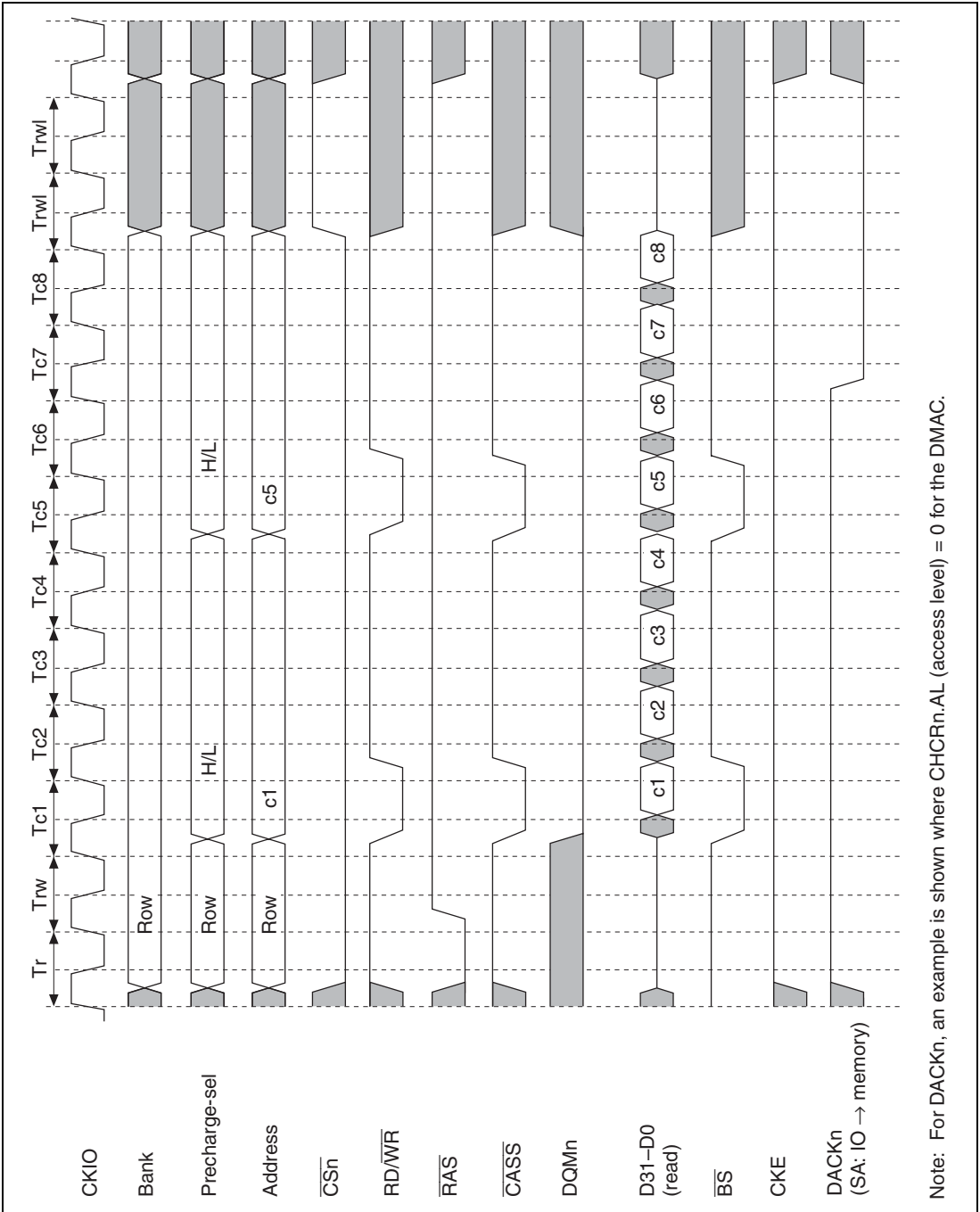


Figure 13.30 Burst Read Timing (RAS Down, Different Row Addresses)



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.31 Burst Write Timing

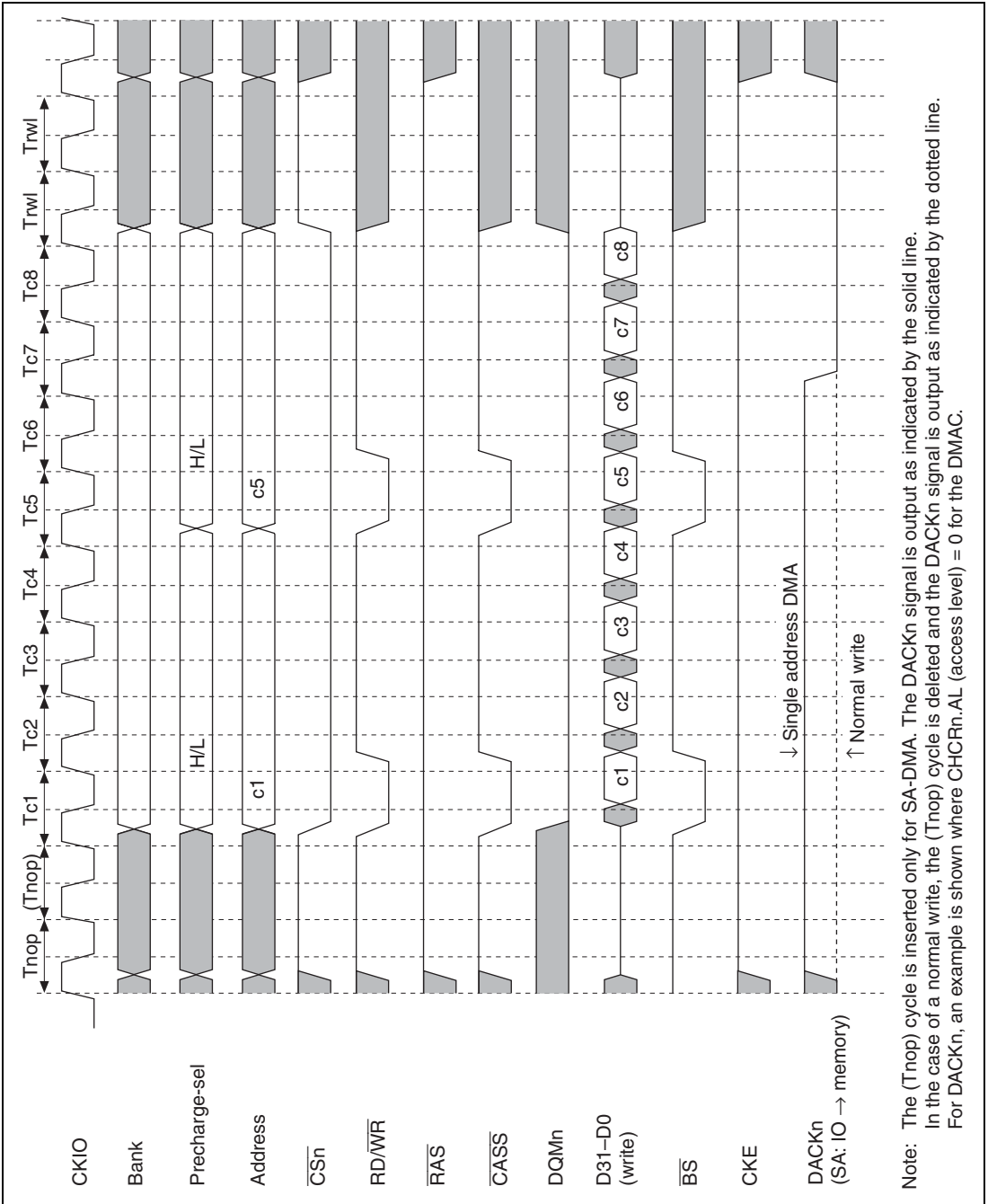


Figure 13.32 Burst Write Timing (Same Row Address)

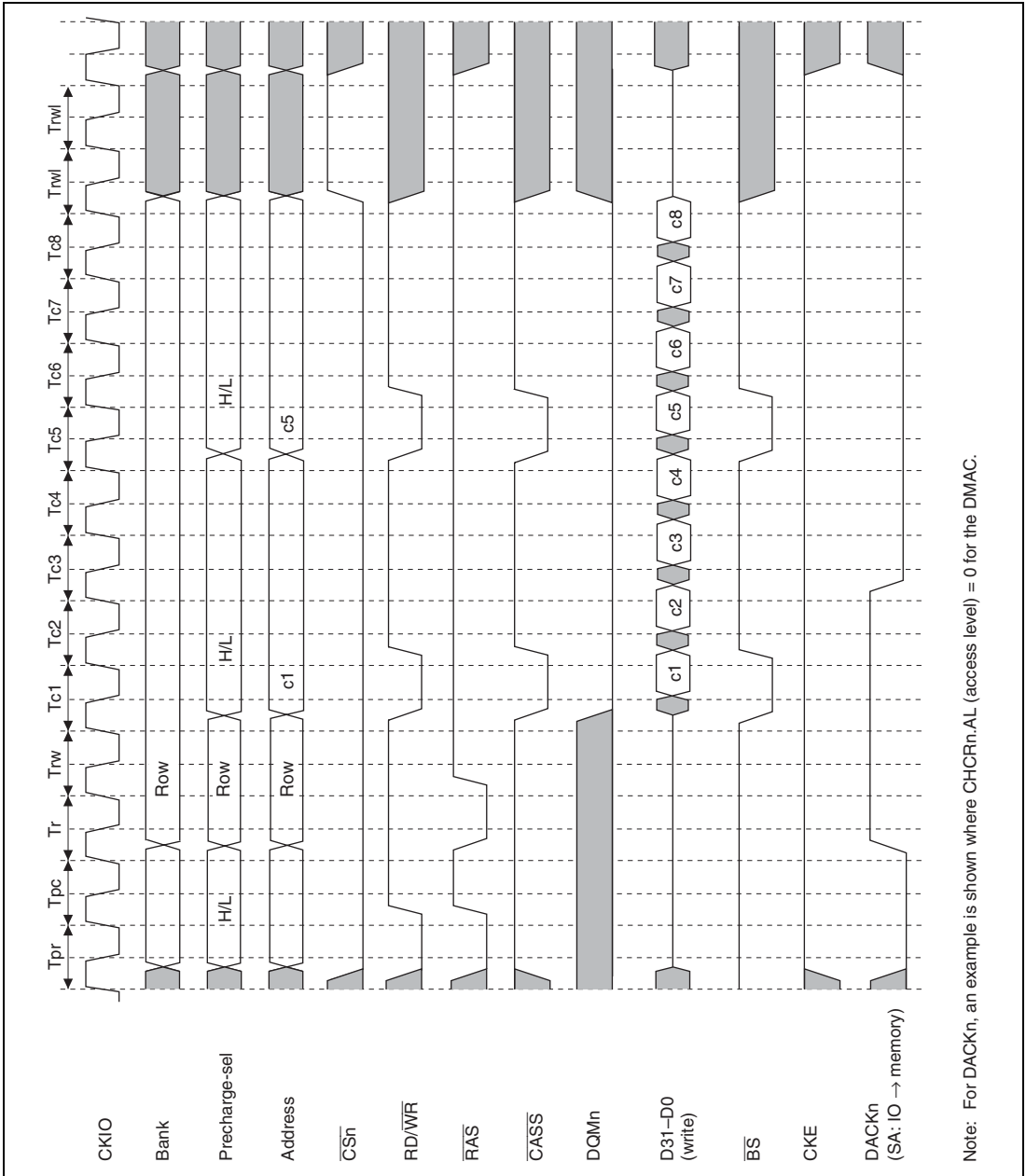


Figure 13.33 Burst Write Timing (Different Row Addresses)

Pipelined Access: When the RASD bit is set to 1 in MCR, pipelined access is performed between an access by the CPU and an access by the DMAC, or in the case of consecutive accesses by the DMAC, to provide faster access to synchronous DRAM. As synchronous DRAM is internally divided into two or four banks, after a READ or WRIT command is issued for one bank it is possible to issue a PRE, ACTV, or other command during the CAS latency cycle or data latch cycle, or during the data write cycle, and so shorten the access cycle.

When a read access is followed by another read access to the same row address, after a READ command has been issued, another READ command is issued before the end of the data latch cycle, so that there is read data on the data bus continuously. When an access is made to another row address and the bank is different, the PRE command or ACTV command can be issued during the CAS latency cycle or data latch cycle. If there are consecutive access requests for different row addresses in the same bank, the PRE command cannot be issued until the last-but-one data latch cycle. If a read access is followed by a write access, it may be possible to issue a PRE or ACTV command, depending on the bank and row address, but since the write data is output at the same time as the WRIT command, the PRE, ACTV, and WRIT commands are issued in such a way that one or two empty cycles occur automatically on the data bus. Similarly, with a read access following a write access, or a write access following a write access, the PRE, ACTV, READ, or WRIT command is issued during the data write cycle for the preceding access; however, in the case of different row addresses in the same bank, a PRE command cannot be issued, and so in this case the PRE command is issued following the number of Trwl cycles specified by the TRWL bits in MCR, after the end of the last data write cycle.

Figure 13.34 shows a burst read cycle for a different bank and row address following a preceding burst read cycle.

Pipelined access is enabled only for consecutive access to area 3, and will be discontinued in the event of an access to another area. Pipelined access is also discontinued in the event of a refresh cycle, or bus release due to bus arbitration. The cases in which pipelined access is available are shown in table 13.16. In this table, “DMAC dual” indicates transfer in DMAC dual address mode, and “DMAC single”, transfer in DMAC single address mode.

Table 13.16 Cycles in Which Pipelined Access Can Be Used

| Preceding Access | | Following Access | | | | | |
|------------------|-------|------------------|-------|-----------|-------|-------------|-------|
| | | CPU | | DMAC Dual | | DMAC Single | |
| | | Read | Write | Read | Write | Read | Write |
| CPU | Read | X | X | O | X | O | O |
| | Write | X | X | O | X | O | O |
| DMAC dual | Read | X | X | X | X | X | X |
| | Write | O | O | O | X | O | O |
| DMAC single | Read | O | O | O | X | O | O |
| | Write | O | O | O | X | O | O |

Legend:

O: Pipelined access possible

X: Pipelined access not possible

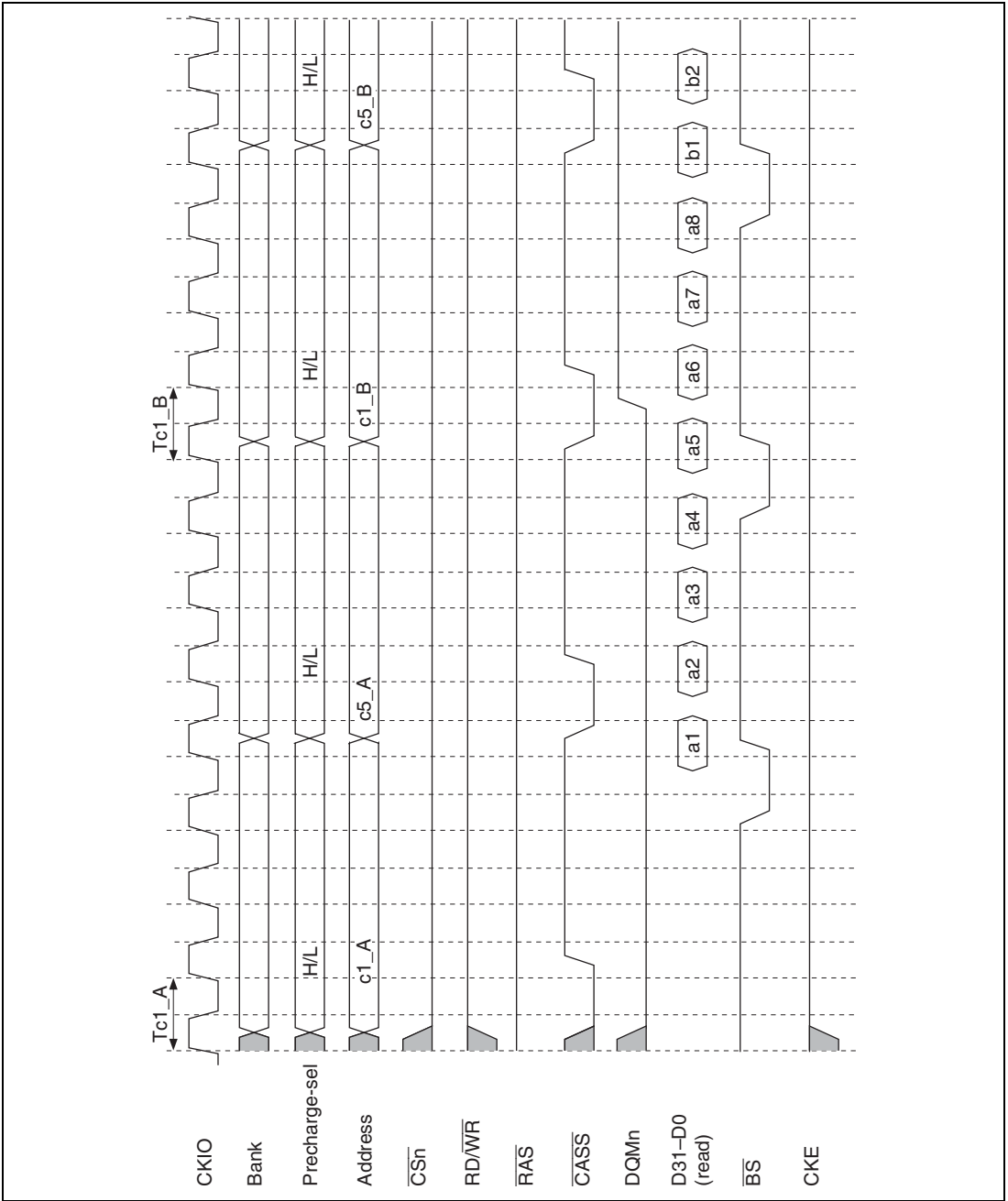


Figure 13.34 Burst Read Cycle for Different Bank and Row Address Following Preceding Burst Read Cycle

Refreshing: The bus state controller is provided with a function for controlling synchronous DRAM refreshing. Auto-refreshing can be performed by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in MCR. If synchronous DRAM is not accessed for a long period, self-refresh mode, in which the power consumption for data retention is low, can be activated by setting both the RMODE bit and the RFSH bit to 1.

- Auto-Refreshing

Refreshing is performed at intervals determined by the input clock selected by bits CKS2–CKS0 in RTCSR, and the value set in RTCOR. The value of bits CKS2–CKS0 in RTCOR should be set so as to satisfy the refresh interval specification for the synchronous DRAM used. First make the settings for RTCOR, RTCNT, and the RMODE and RFSH bits in MCR, then make the CKS2–CKS0 setting last of all. When the clock is selected by CKS2–CKS0, RTCNT starts counting up from the value at that time. The RTCNT value is constantly compared with the RTCOR value, and if the two values are the same, a refresh request is generated and an auto-refresh is performed. At the same time, RTCNT is cleared to zero and the count-up is restarted.

Figure 13.36 shows the auto-refresh cycle timing.

First, an REF command is issued in the TRr cycle. After the TRr cycle, new command output cannot be performed for the duration of the number of cycles specified by bits TRAS2–TRAS0 in MCR plus the number of cycles specified by bits TRC2–TRC0 in MCR. The TRAS2–TRAS0 and TRC2–TRC0 bits must be set so as to satisfy the synchronous DRAM refresh cycle time specification (active/active command delay time).

Auto-refreshing is performed in normal operation, in sleep mode, and in the case of a manual reset. When both areas 2 and 3 are set to the synchronous DRAM, auto-refreshing of area 2 is performed subsequent to area 3.

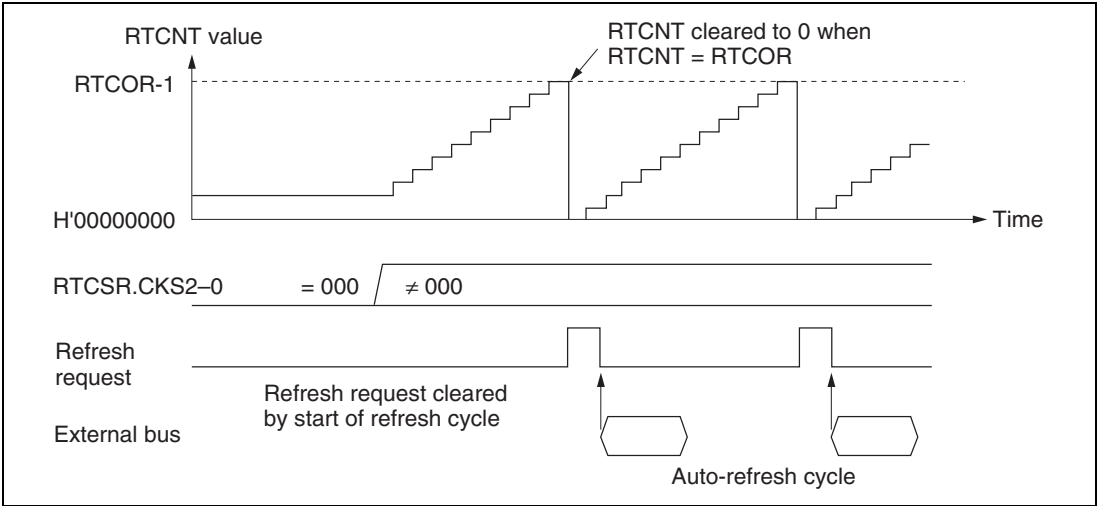


Figure 13.35 Auto-Refresh Operation

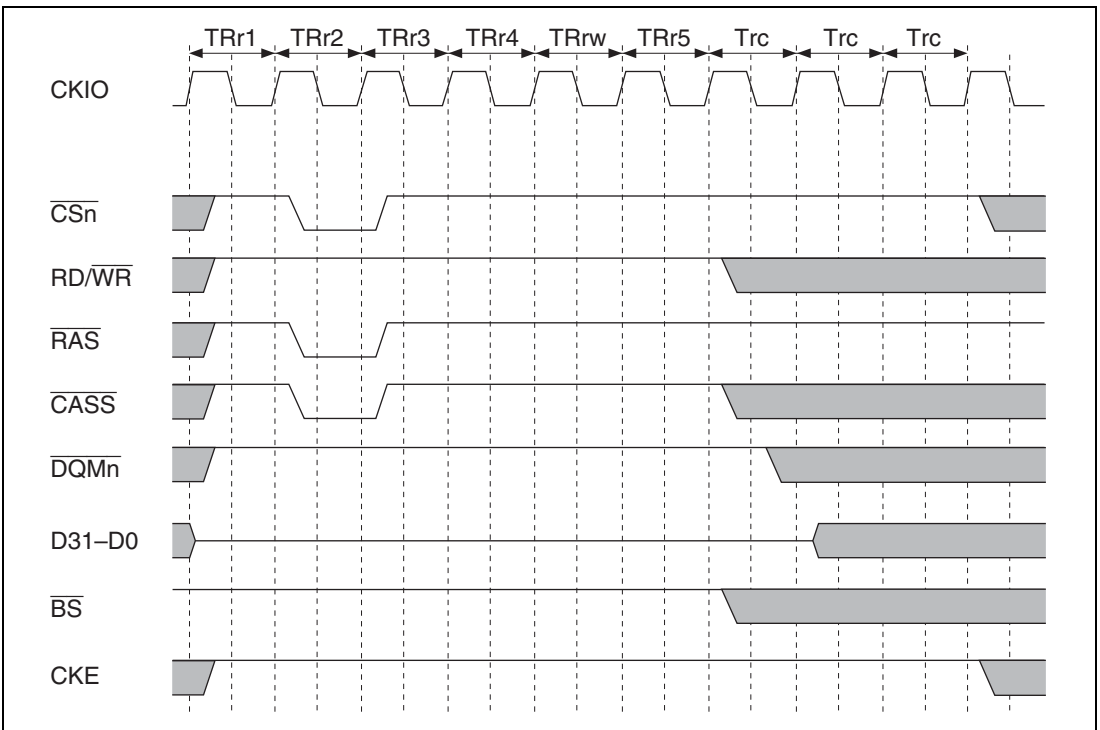


Figure 13.36 Synchronous DRAM Auto-Refresh Timing

- Self-Refreshing

Self-refresh mode is a kind of standby mode in which the refresh timing and refresh addresses are generated within the synchronous DRAM. Self-refreshing is activated by setting both the RMODE bit and the RFSH bit to 1. The self-refresh state is maintained while the CKE signal is low. Synchronous DRAM cannot be accessed while in the self-refresh state. Self-refresh mode is cleared by clearing the RMODE bit to 0. After self-refresh mode has been cleared, command issuance is disabled for the number of cycles specified by bits TRC2–TRC0 in MCR. Self-refresh timing is shown in figure 13.37. Settings must be made so that self-refresh clearing and data retention are performed correctly, and auto-refreshing is performed at the correct intervals. When self-refreshing is activated from the state in which auto-refreshing is set, or when exiting standby mode other than through a power-on reset, auto-refreshing is restarted if RFSH is set to 1 and RMODE is cleared to 0 when self-refresh mode is cleared. If the transition from clearing of self-refresh mode to the start of auto-refreshing takes time, this time should be taken into consideration when setting the initial value of RTCNT. Making the RTCNT value 1 less than the RTCOR value will enable refreshing to be started immediately. After self-refreshing has been set, the self-refresh state continues even if the chip standby state is entered using this LSI standby function, and is maintained even after recovery from standby mode other than through a power-on reset.

In the case of a power-on reset, the bus state controller's registers are initialized, and therefore the self-refresh state is cleared.

Self-refreshing is performed in normal operation, in sleep mode, in standby mode, and in the case of a manual reset.

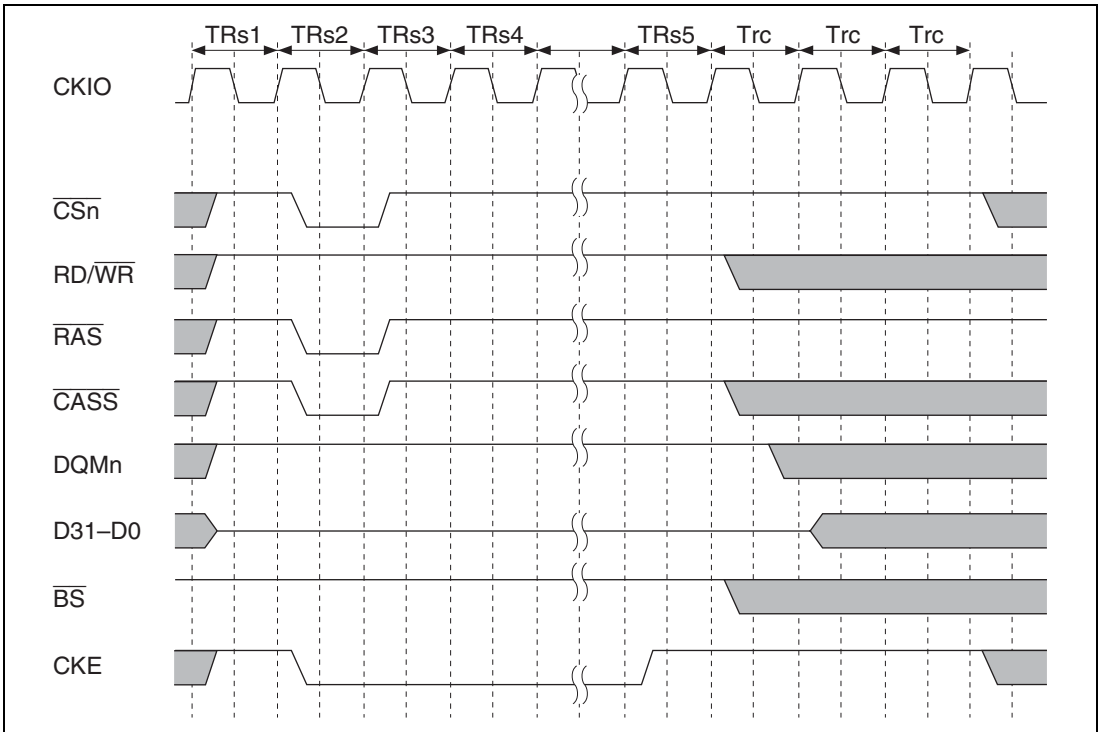


Figure 13.37 Synchronous DRAM Self-Refresh Timing

- Relationship between Refresh Requests and Bus Cycle Requests

If a refresh request is generated during execution of a bus cycle, execution of the refresh is deferred until the bus cycle is completed. Refresh operations are deferred during multiple bus cycles generated because the data bus width is smaller than the access size (for example, when performing longword access to 8-bit bus width memory) and during a 32-byte transfer such as a cache fill or write-back, and also between read and write cycles during execution of a TAS instruction, and between read and write cycles when DMAC dual address transfer is executed. If a refresh request occurs when the bus has been released by the bus arbiter, refresh execution is deferred until the bus is acquired. If a match between RTCNT and RTCOR occurs while a refresh is waiting to be executed, so that a new refresh request is generated, the previous refresh request is eliminated. In order for refreshing to be performed normally, care must be taken to ensure that no bus cycle or bus mastership occurs that is longer than the refresh interval. When a refresh request is generated, the $\overline{\text{BACK}}$ pin is negated (driven high). Therefore, normal refreshing can be performed by having the $\overline{\text{BACK}}$ pin monitored by a bus master other than this LSI requesting the bus, or the bus arbiter, and returning the bus to this LSI.

Power-On Sequence: In order to use synchronous DRAM, mode setting must first be performed after powering on. To perform synchronous DRAM initialization correctly, the bus state controller registers must first be set, followed by a write to the synchronous DRAM mode register. In synchronous DRAM mode register setting, the address signal value at that time is latched by a combination of the $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and $\text{RD}/\overline{\text{WR}}$ signals. If the value to be set is X, the bus state controller provides for value X to be written to the synchronous DRAM mode register by performing a write to address $\text{H'FF900000} + \text{X}$ for area 2 synchronous DRAM, and to address $\text{H'FF940000} + \text{X}$ for area 3 synchronous DRAM. In this operation the data is ignored, but the mode write is performed as a byte-size access. To set burst read/burst write, CAS latency 1 to 3, wrap type = sequential, and burst length 4, 8*, supported by this LSI, arbitrary data is written by byte-size access to the following addresses.

| Bus Width | | CAS Latency | Area 2 | Area 3 |
|-----------|----|-------------|------------|------------|
| 32 | 4 | 1 | H'FF900048 | H'FF940048 |
| | | 2 | H'FF900088 | H'FF940088 |
| | | 3 | H'FF9000C8 | H'FF9400C8 |
| 32 | 8* | 1 | H'FF90004C | H'FF94004C |
| | | 2 | H'FF90008C | H'FF94008C |
| | | 3 | H'FF9000CC | H'FF9400CC |

Note: * SH7751R only

The value set in MCR.MRSET is used to select whether a precharge all banks command or a mode register setting command is issued. The timing for the precharge all banks command is shown in figure 13.38 (1), and the timing for the mode register setting command in figure 13.38 (2).

Before mode register, a 200 μs idle time (depending on the memory manufacturer) must be guaranteed after the power required for the synchronous DRAM is turned on. If the reset signal pulse width is greater than this idle time, there is no problem in making the precharge all banks setting immediately.

First, a precharge all banks (PALL) command is issued in the TRp1 cycle by performing a write to address $\text{H'FF900000} + \text{X}$ or $\text{H'FF940000} + \text{X}$ while $\text{MCR.MRSET} = 0$. Next, the number of dummy auto-refresh cycles specified by the manufacturer (usually 8) or more must be executed. This is achieved automatically while various kinds of initialization are being performed after auto-refresh setting, but a way of carrying this out more dependably is to change the RTCOR register value to set a short refresh request generation interval just while these dummy cycles are being executed. With simple read or write access, the address counter in the synchronous DRAM used for auto-refreshing is not initialized, and so the cycle must always be an auto-refresh cycle. After

auto-refreshing has been executed at least the prescribed number of times, a mode register setting command is issued in the TMw1 cycle by setting MCR.MRSET to 1 and performing a write to address H'FF900000 + X or H'FF940000 + X.

Synchronous DRAM mode register setting should be executed once only after power-on reset and before synchronous DRAM access, and no subsequent changes should be made.

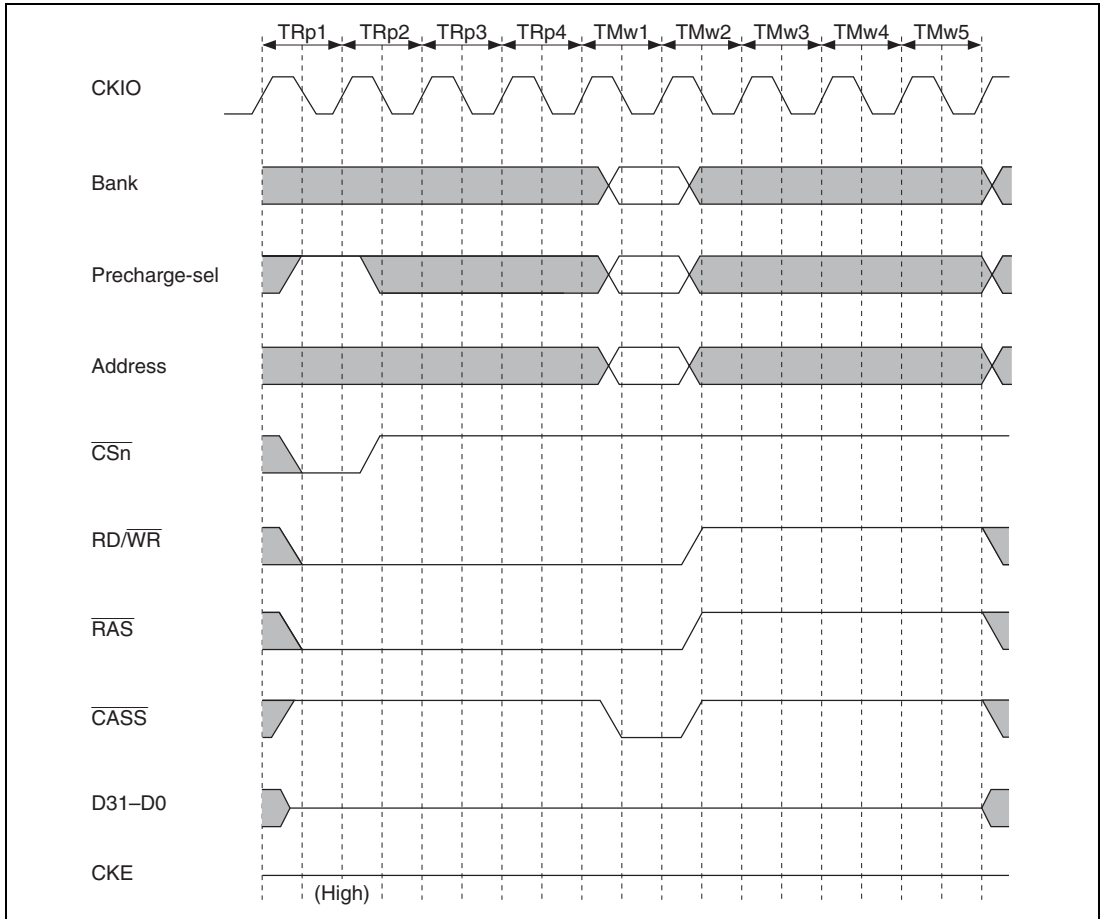


Figure 13.38 (1) Synchronous DRAM Mode Write Timing (PALL)

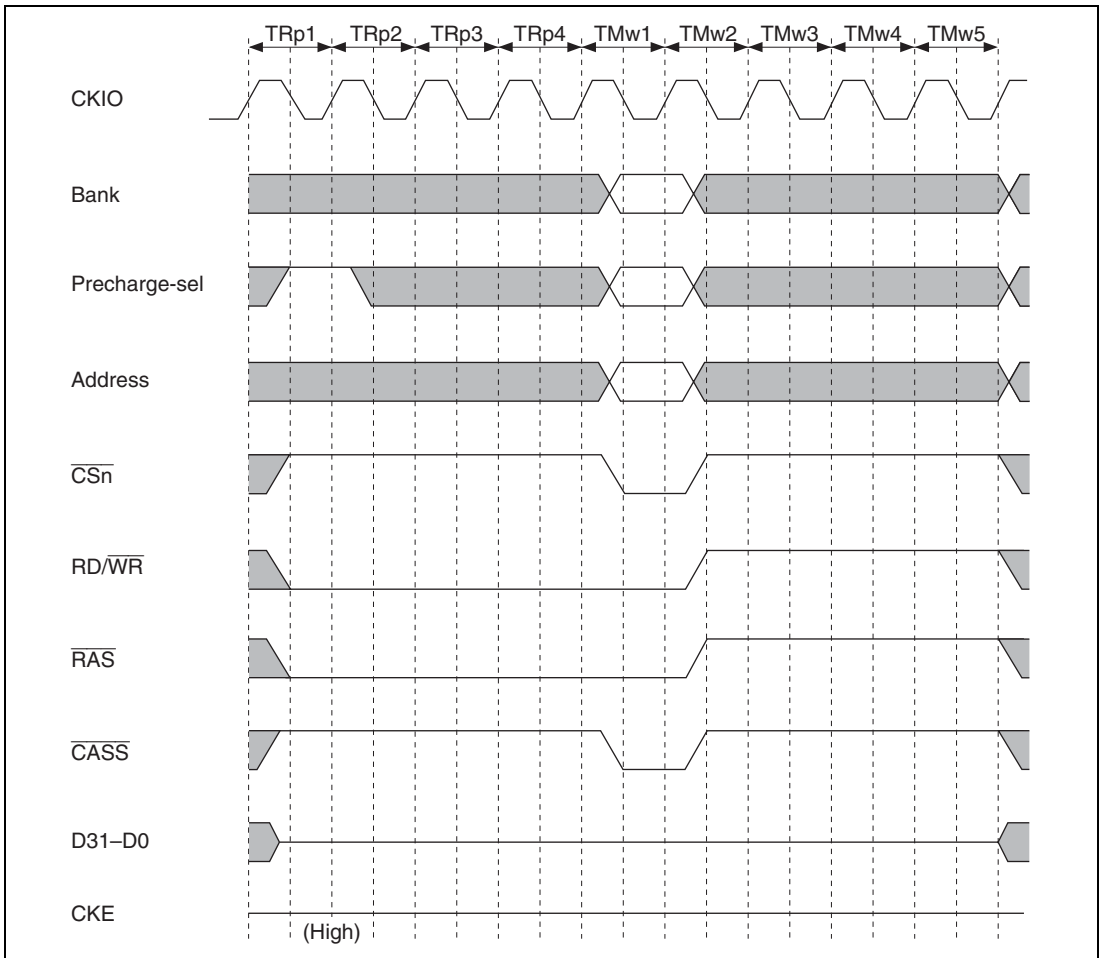


Figure 13.38 (2) Synchronous DRAM Mode Write Timing (Mode Register Setting)

Changing the Burst Length (SH7751R Only): When synchronous DRAM is connected with the 32-bit memory bus of the SH7751R, a burst length of either 4 or 8 can be selected by the setting of the SDBL bit of the BCR3 register. For more details, see the description of the BCR3 register.

- Burst Read

Figure 13.39 is the timing chart for burst-read operations. For the example shown below, we assume that two synchronous DRAMs of $512k \times 16 \text{ bits} \times 2 \text{ banks}$ are connected and are used with a 32-bit data width and a burst length of 8. Following the T_r cycle, during which an ACTV command is output, a READA command is issued during cycle T_{c1} . During the T_{d1} to T_{d8} cycles, the read data are accepted on the rising edges of the external command clock

(CKIO). Tpc is the cycle used to wait for auto-precharging, which is triggered by the READA command, to be completed in the synchronous DRAM. During this cycle, no new command that accesses the same bank can be issued. In this LSI, the number of Tpc cycles is determined by the setting of the TPC2 to TPC0 bits of the MCR, and no command that operates on the synchronous DRAM is issued during these cycles.

Figure 13.39 shows an example of the basic timing of a burst-read. To allow the connection of a lower-speed DRAM, the cycle's period can be extended by the settings of the bits in WCR2 and MCR. The number of cycles from cycle Tr on which the ACTV command is output to cycle Tc1 on which the READA command is output can be specified by the RCD1 and RCD0 bits in MCR: the number of cycles is 2, 3, or 4 for the setting value of 1, 2, or 3, respectively. When two or more cycles are specified, the Trw cycle, which is for the issuing of NOP commands to the synchronous DRAM, is inserted between the Tr and Tc cycles. The number of cycles from cycle Tc1 on which the READA command is output until cycle Td1, in which the first part of the data to be read is received, can be set by the bits A2W2 to A2W0 and A3W2 to A3W0 of WCR2. These independently select a number of cycles between 1 and 5 for areas 2 and 3. Note that this number of cycles is equal to the number of CAS latency cycles of the synchronous DRAM.

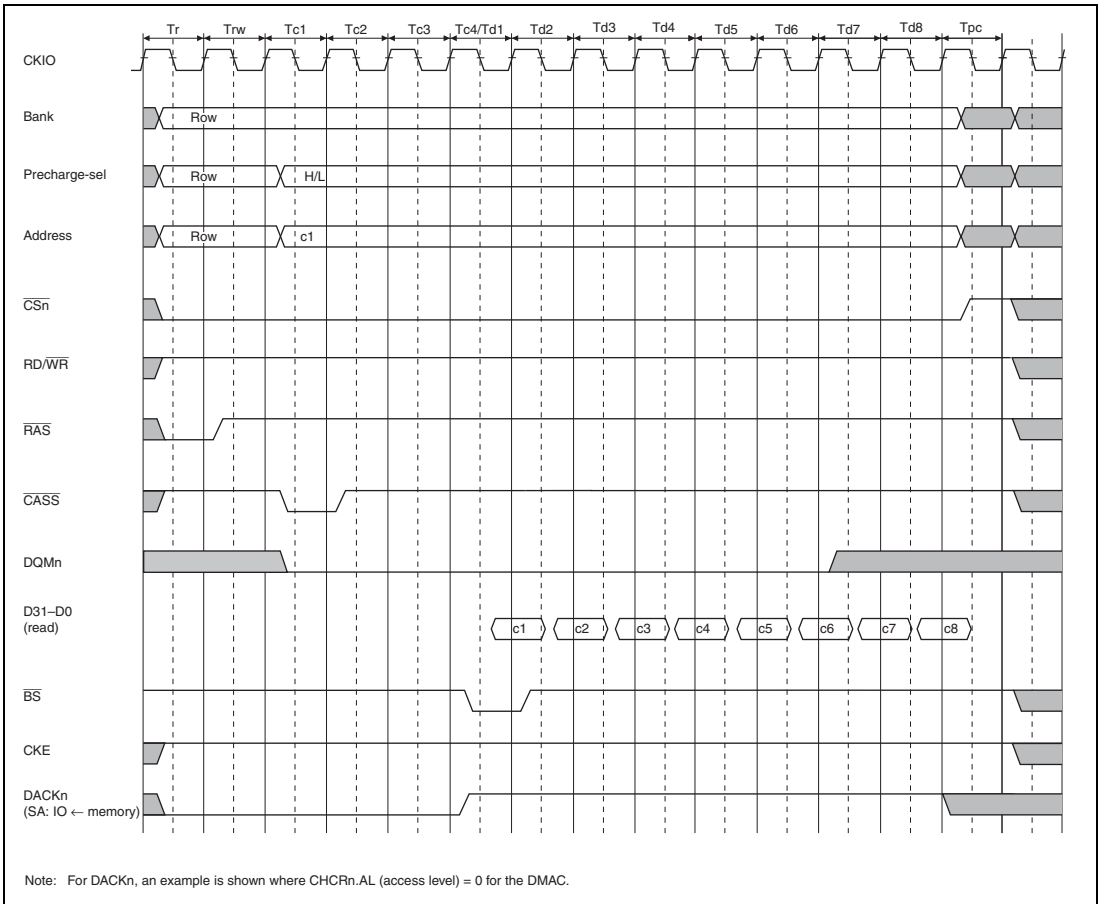


Figure 13.39 Basic Timing of a Burst Read from Synchronous DRAM (Burst Length = 8)

In a cycle of access to synchronous DRAM, the \overline{BS} signal is asserted for one clock cycle at the beginning of a bus cycle. Data are accessed in the following sequence: in the fill operation for a cache miss, the data between the 32-bit boundaries that include the missed data are first read; after that, the data between 32-byte boundaries that include the missed data are read in a wraparound way.

- Burst Write

Figure 13.40 is the timing chart for a burst-write operation with a burst length of 8. In this LSI, a burst write takes place when a copy-back of the cache or a 32-byte transfer of data by the DMAC takes place. In a burst-write operation, a WRITA command that include auto precharging, is issued during the Tc1 cycle that follows the Tr cycle in which the ACTV command is output. During the write cycle, the data to be written is output along with the write command. With a write command that includes an auto precharge, precharging is of the relevant bank of the synchronous DRAM and takes place on completion of the write command, so no new command that accesses the same bank can be issued until precharging has been completed. For this reason, the Trw1 cycles are added as a period of waiting for precharging to start after the write command has been issued. This is additional to the precharge-waiting cycle as used in read access. The Trw1 cycles delay the issuing of new commands to the same bank. The setting of the TRWL2 to TRWL0 bits of MCR selects the number of Trw1 cycles. The data between 32-byte boundaries are written in a wraparound way.

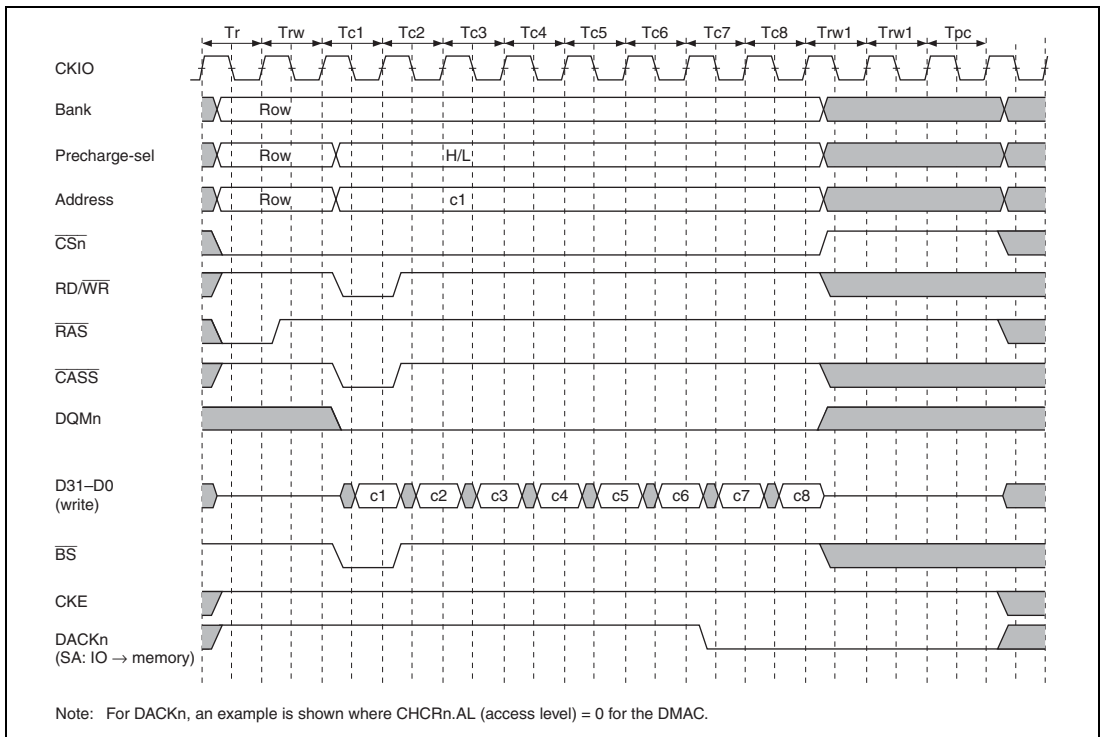


Figure 13.40 Basic Timing of a Burst Write to Synchronous DRAM

13.3.6 Burst ROM Interface

Setting bits A0BST2–A0BST0, A5BST2–A5BST0, and A6BST2–A6BST0 in BCR1 to a non-zero value allows burst ROM to be connected to areas 0, 5, and 6. The burst ROM interface provides high-speed access to ROM that has a burst access function. The timing for burst access to burst ROM is shown in figure 13.41. Two wait cycles are set. Basically, access is performed in the same way as for SRAM interface, but when the first cycle ends, only the address is changed before the next access is executed. When 8-bit ROM is connected, the number of consecutive accesses can be set as 4, 8, 16, or 32 with bits A0BST2–A0BST0, A5BST2–A5BST0, or A6BST2–A6BST0. When 16-bit ROM is connected, 4, 8, or 16 can be set in the same way. When 32-bit ROM is connected, 4 or 8 can be set.

$\overline{\text{RDY}}$ pin sampling is always performed when one or more wait states are set.

The second and subsequent access cycles also comprise two cycles when a burst ROM setting is made and the wait specification is 0. The timing in this case is shown in figure 13.42.

In a burst ROM interface write operation is performed as SRAM interface.

In 32-byte transfer, a total of 32 bytes are transferred consecutively according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on the data at the 32-byte boundary. The bus is not released during this period.

Figure 13.43 shows the timing when a burst ROM setting is made, and setup/hold is specified in WCR3.

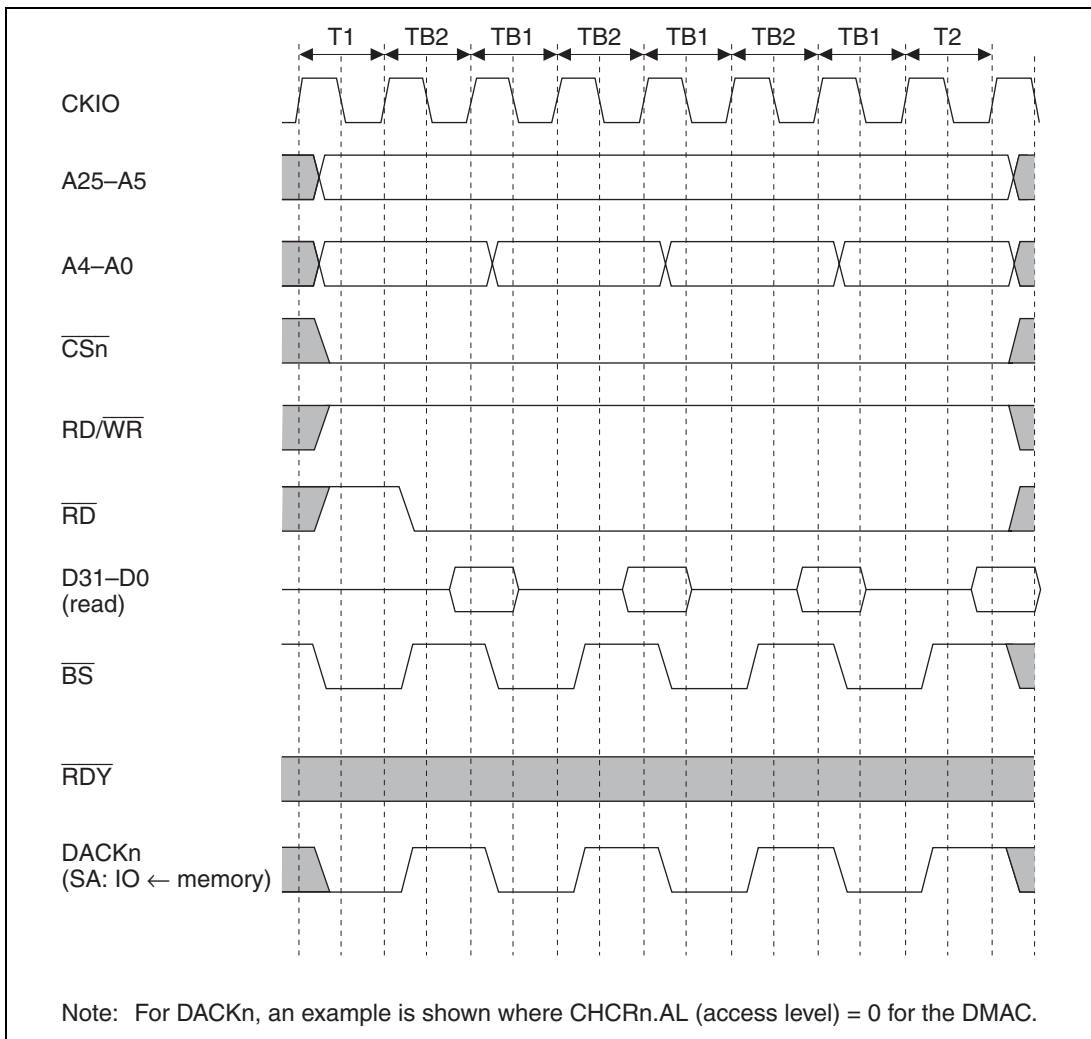


Figure 13.41 Burst ROM Basic Access Timing

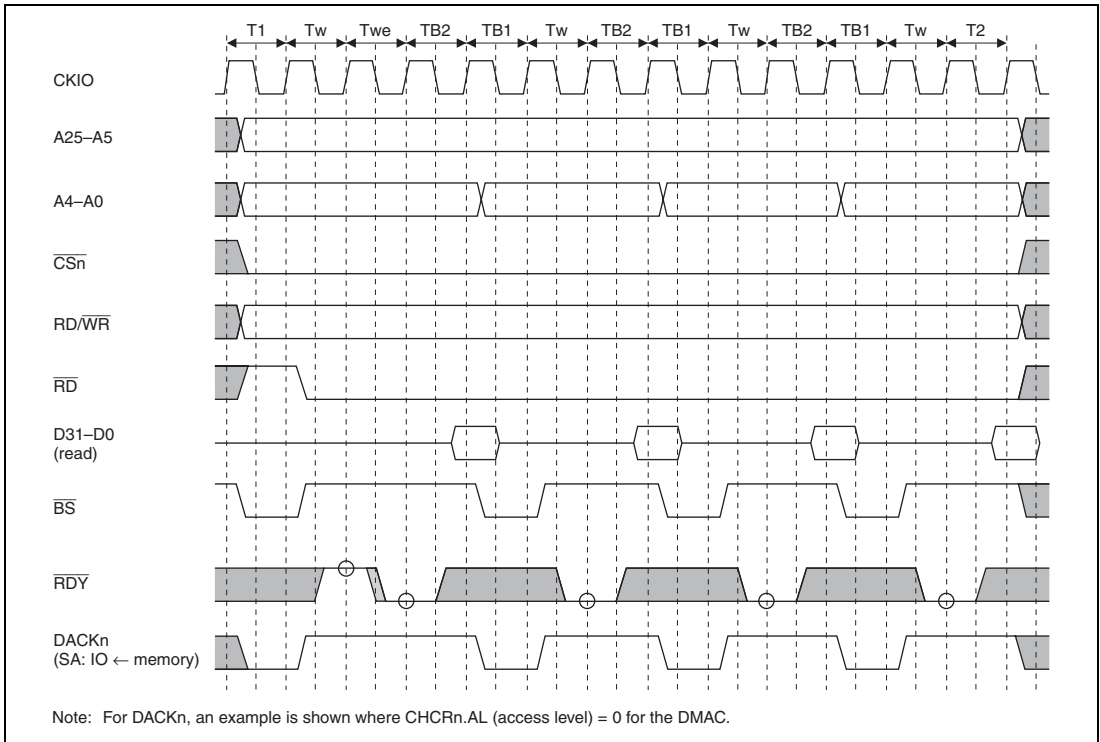


Figure 13.42 Burst ROM Wait Access Timing

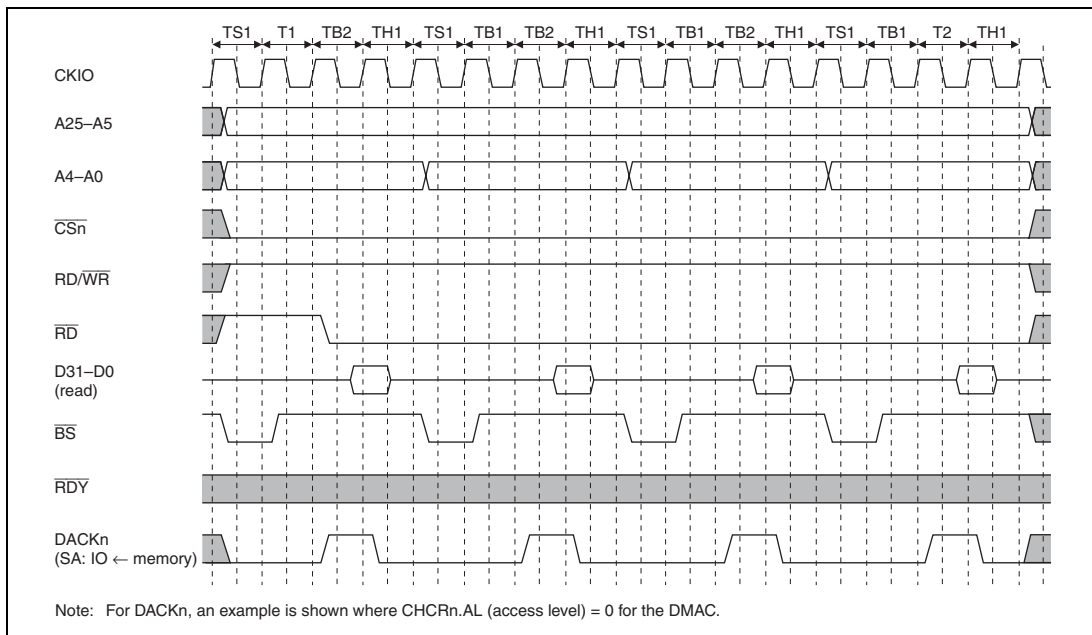


Figure 13.43 Burst ROM Wait Access Timing

13.3.7 PCMCIA Interface

In this LSI, setting the A56PCM bit in BCR1 to 1 makes the bus interface for external memory space areas 5 and 6 an IC memory card interface or I/O card interface as stipulated in JEIDA specification version 4.2 (PCMCIA2.1).

Figure 13.44 shows an example of PCMCIA card connection to this LSI. To enable active insertion of the PCMCIA cards (i.e. insertion or removal while system power is being supplied), a 3-state buffer must be connected between this LSI bus interface and the PCMCIA cards.

As operation in big endian mode is not explicitly stipulated in the JEIDA/PCMCIA standard, this LSI supports only little-endian mode setting and the little-endian mode PCMCIA interface.

When the MMU is on, PCMCIA interface can be set in MMU page units, and there is a choice of 8-bit common memory, 16-bit common memory, 8-bit attribute memory, 16-bit attribute memory, 8-bit I/O space, 16-bit I/O space, or dynamic bus sizing. See section 3, Memory Management Unit (MMU), for details of the setting method. When the MMU is off, the setting of bits SA2 to SA0 of PTEA is always used for access.

| SA2 | SA1 | SA0 | Description |
|-----|-----|-----|-------------------------------|
| 0 | 0 | 0 | Reserved (Setting prohibited) |
| | | 1 | Dynamic I/O bus sizing |
| | 1 | 0 | 8-bit I/O space |
| | | 1 | 16-bit I/O space |
| 1 | 0 | 0 | 8-bit common memory |
| | | 1 | 16-bit common memory |
| | 1 | 0 | 8-bit attribute memory |
| | | 1 | 16-bit attribute memory |

When the MMU is on, wait cycles in a bus access can be set in MMU page units. See section 3, Memory Management Unit (MMU), for details of the setting method. When the MMU is off, access is always performed according to the setting of the TC bit in PTEA. When TC is cleared to 0, bits A5W2–A5W0 in wait control register 2 (WCR2) and bits A5PCW1–A5PCW0, A5TED2–A5TED0, and A5TEH2–A5TEH0 in the PCMCIA control register (PCR) are selected. When TC is set to 1, bits A6W2–A6W0 in WCR2 and bits A6PCW1–A6PCW0, A6TED2–A6TED0, and A6TEH2–A6TEH0 in PCR are selected.

Access to a PCMCIA interface area by the DMAC is always performed using the DMAC's CHCRn.SSAn, CHCRn.DSAn, CHCRn.STC, and CHCRn.DTC values.

AnPCW1–AnPCW0 specify the number of wait states to be inserted in a low-speed bus cycle; a value of 0, 15, 30, or 50 can be set, and this value is added to the number of wait states for insertion specified by WCR2. AnTED2–AnTED0 can be set to a value from 0 to 15, enabling the address, \overline{CS} , $\overline{CE2A}$, $\overline{CE2B}$, and \overline{REG} setup times with respect to the \overline{RD} and $\overline{WE1}$ signals to be secured. AnTEH2–AnTEH0 can also be set to a value from 0 to 15, enabling the address, \overline{CS} , $\overline{CE2A}$, $\overline{CE2B}$, and \overline{REG} data hold times with respect to the \overline{RD} and $\overline{WE1}$ signals to be secured.

Wait cycles between cycles are set with bits A5IW2–A5IW0 and A6IW2–A6IW0 in wait control register 1 (WCR1). The inter-cycle write cycles selected depend only on the area accessed (area 5 or 6): when area 5 is accessed, bits A5IW2–A5IW0 are selected, and when area 6 is accessed, bits A6IW2–A6IW0 are selected.

In 32-byte transfer, a total of 32 bytes are transferred consecutively according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed in wraparound mode on the data at the 32-byte boundary. The bus is not released during this operation.

Table 13.17 Relationship between Address and CE When Using PCMCIA Interface

| Bus Width (Bits) | Read/Write | Access Size (Bits)*1 | Access | | | | | | | | | |
|------------------|------------|----------------------|----------|------------|--------|------------|-----|----|------------------|------------------|------------|------------|
| | | | Odd/Even | IOIS16 | Access | CE2 | CE1 | A0 | D15–D8 | D7–D0 | | |
| 8 | Read | 8 | Even | Don't care | — | 1 | 0 | 0 | Invalid | Read data | | |
| | | | Odd | Don't care | — | 1 | 0 | 1 | Invalid | Read data | | |
| | | 16 | Even | Don't care | First | 1 | 0 | 0 | Invalid | Lower read data | | |
| | | | Even | Don't care | Second | 1 | 0 | 1 | Invalid | Upper read data | | |
| | | | Odd | Don't care | — | — | — | — | — | — | | |
| | | | Odd | Don't care | — | — | — | — | — | — | | |
| | Write | 8 | Even | Don't care | — | 1 | 0 | 0 | Invalid | Write data | | |
| | | | Odd | Don't care | — | 1 | 0 | 1 | Invalid | Write data | | |
| | | 16 | Even | Don't care | First | 1 | 0 | 0 | Invalid | Lower write data | | |
| | | | Even | Don't care | Second | 1 | 0 | 1 | Invalid | Upper write data | | |
| | | | Odd | Don't care | — | — | — | — | — | — | | |
| | | | Odd | Don't care | — | — | — | — | — | — | | |
| 16 | Read | 8 | Even | Don't care | — | 1 | 0 | 0 | Invalid | Read data | | |
| | | | Odd | Don't care | — | 0 | 1 | 1 | Read data | Invalid | | |
| | | 16 | Even | Don't care | — | 0 | 0 | 0 | Upper read data | Lower read data | | |
| | | | Odd | Don't care | — | — | — | — | — | — | | |
| | | | Write | 8 | Even | Don't care | — | 1 | 0 | 0 | Invalid | Write data |
| | | | | | Odd | Don't care | — | 0 | 1 | 1 | Write data | Invalid |
| | 16 | Even | | Don't care | — | 0 | 0 | 0 | Upper write data | Lower write data | | |
| | | Odd | | Don't care | — | — | — | — | — | — | | |

| Bus Width (Bits) | Access Read/Write | Access Size (Bits)*1 | Odd/Even | IOIS16 | Access | CE2 | CE1 | A0 | Access | |
|----------------------|-------------------|----------------------|----------|--------|--------|-----|-----|------------------|------------------|------------------|
| | | | | | | | | | D15–D8 | D7–D0 |
| Dynamic bus sizing*2 | Read | 8 | Even | 0 | — | 1 | 0 | 0 | Invalid | Read data |
| | | | Odd | 0 | — | 0 | 1 | 1 | Read data | Invalid |
| | | 16 | Even | 0 | — | 0 | 0 | 0 | Upper read data | Lower read data |
| | | | Odd | 0 | — | — | — | — | — | — |
| | Write | 8 | Even | 0 | — | 1 | 0 | 0 | Invalid | Write data |
| | | | Odd | 0 | — | 0 | 1 | 1 | Write data | Invalid |
| | | 16 | Even | 0 | — | 0 | 0 | 0 | Upper write data | Lower write data |
| | | | Odd | 0 | — | — | — | — | — | — |
| | Read | 8 | Even | 1 | — | 1 | 0 | 0 | Invalid | Read data |
| | | | Odd | 1 | First | 0 | 1 | 1 | Ignored | Invalid |
| | | | Odd | 1 | Second | 1 | 0 | 1 | Invalid | Read data |
| | | 16 | Even | 1 | First | 0 | 0 | 0 | Invalid | Lower read data |
| Even | | | 1 | Second | 1 | 0 | 1 | Invalid | Upper read data | |
| Odd | | | 1 | — | — | — | — | — | — | |
| Write | 8 | Even | 1 | — | 1 | 0 | 0 | Invalid | Write data | |
| | | Odd | 1 | First | 0 | 1 | 1 | Invalid | Write data | |
| | | Odd | 1 | Second | 1 | 0 | 1 | Invalid | Write data | |
| | 16 | Even | 1 | First | 0 | 0 | 0 | Upper write data | Lower write data | |
| | | Even | 1 | Second | 1 | 0 | 1 | Invalid | Upper write data | |
| | | Odd | 1 | — | — | — | — | — | — | |

Notes: 1. In 32-bit/64-bit/32-byte transfer, the above accesses are repeated, with address incrementing performed automatically according to the bus width, until the transfer data size is reached.

2. PCMCIA I/O card interface only

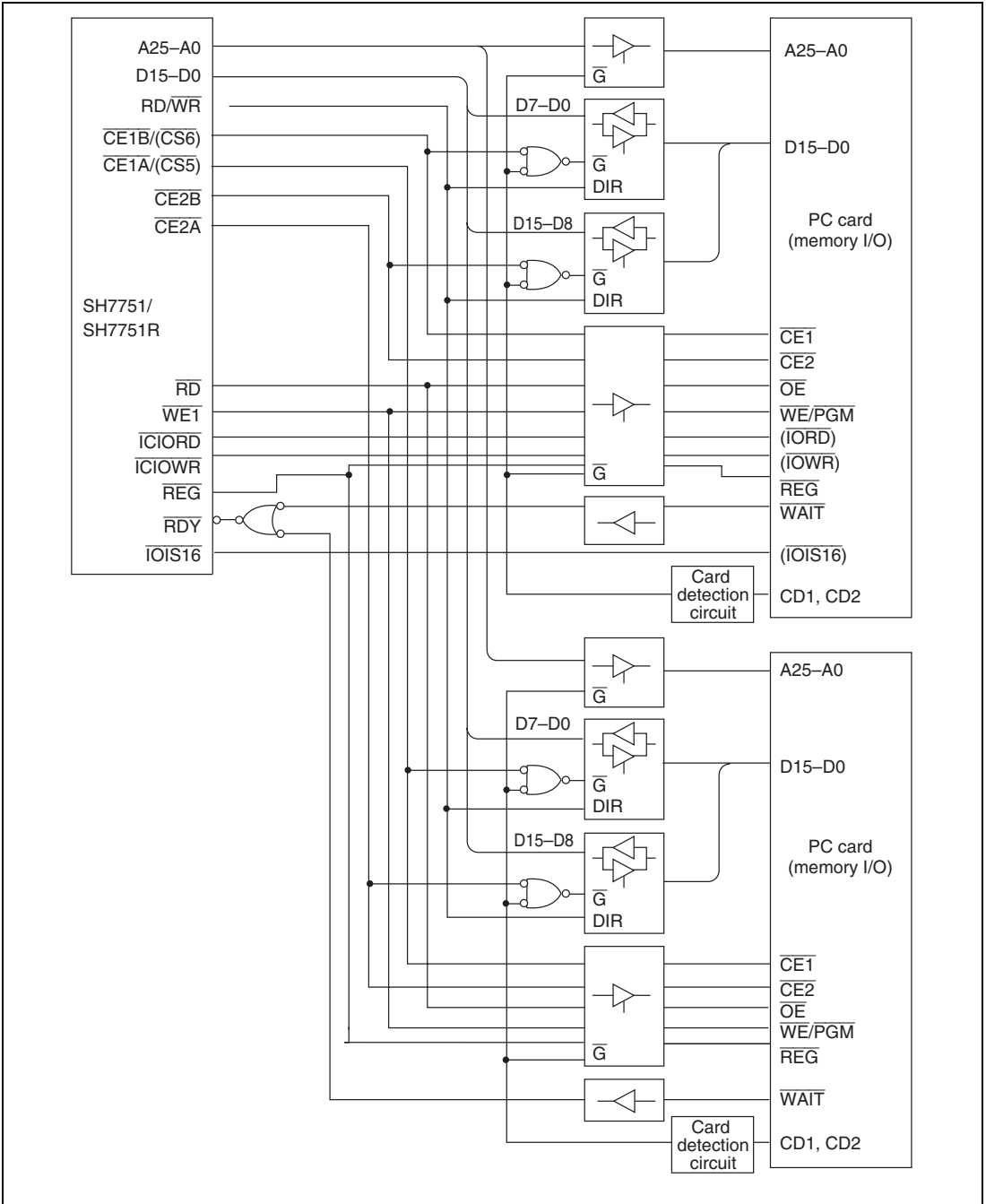


Figure 13.44 Example of PCMCIA Interface

Memory Card Interface Basic Timing: Figure 13.45 shows the basic timing for the PCMCIA memory card interface, and figure 13.46 shows the wait timing for the PCMCIA memory card interface.

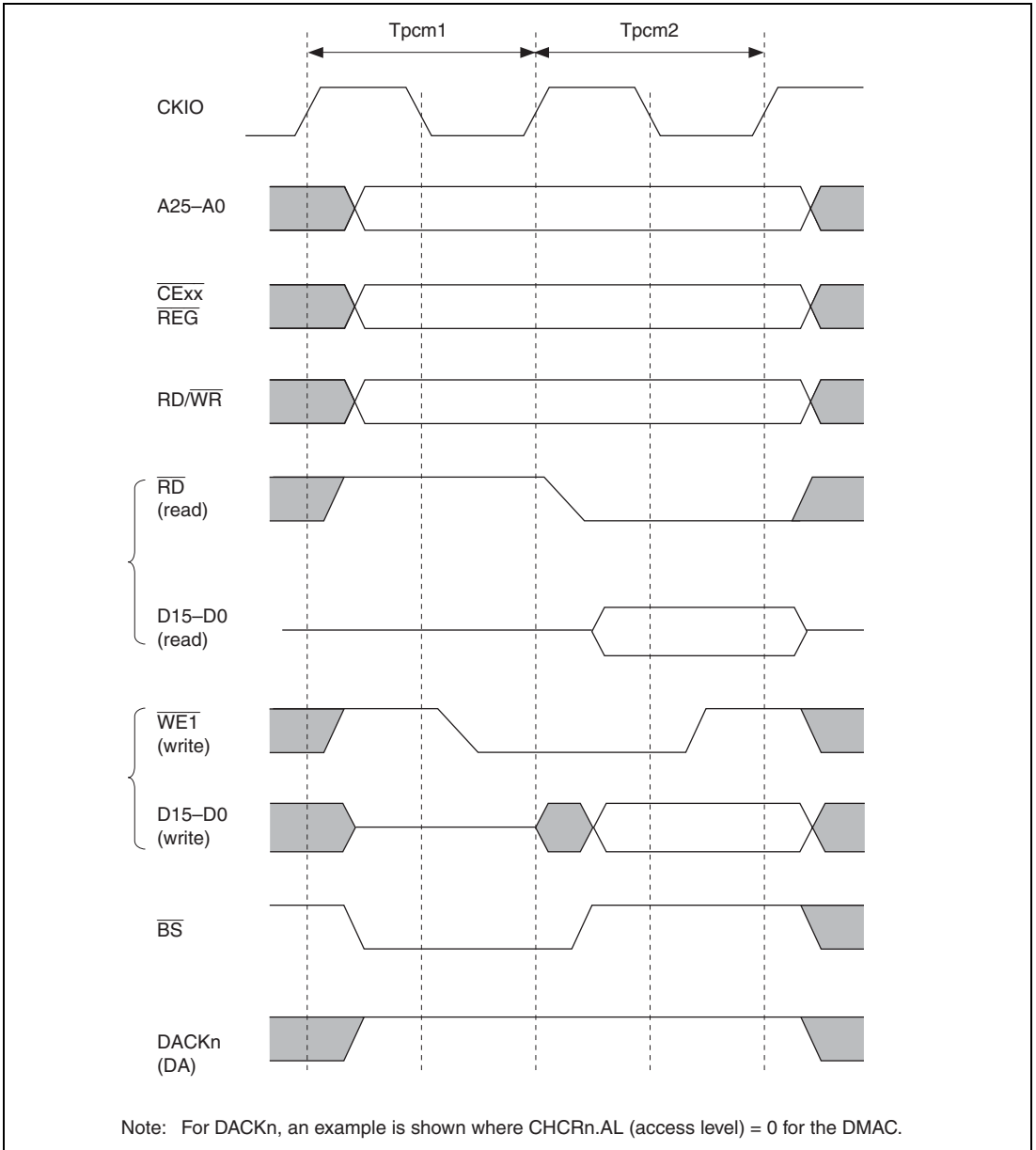


Figure 13.45 Basic Timing for PCMCIA Memory Card Interface

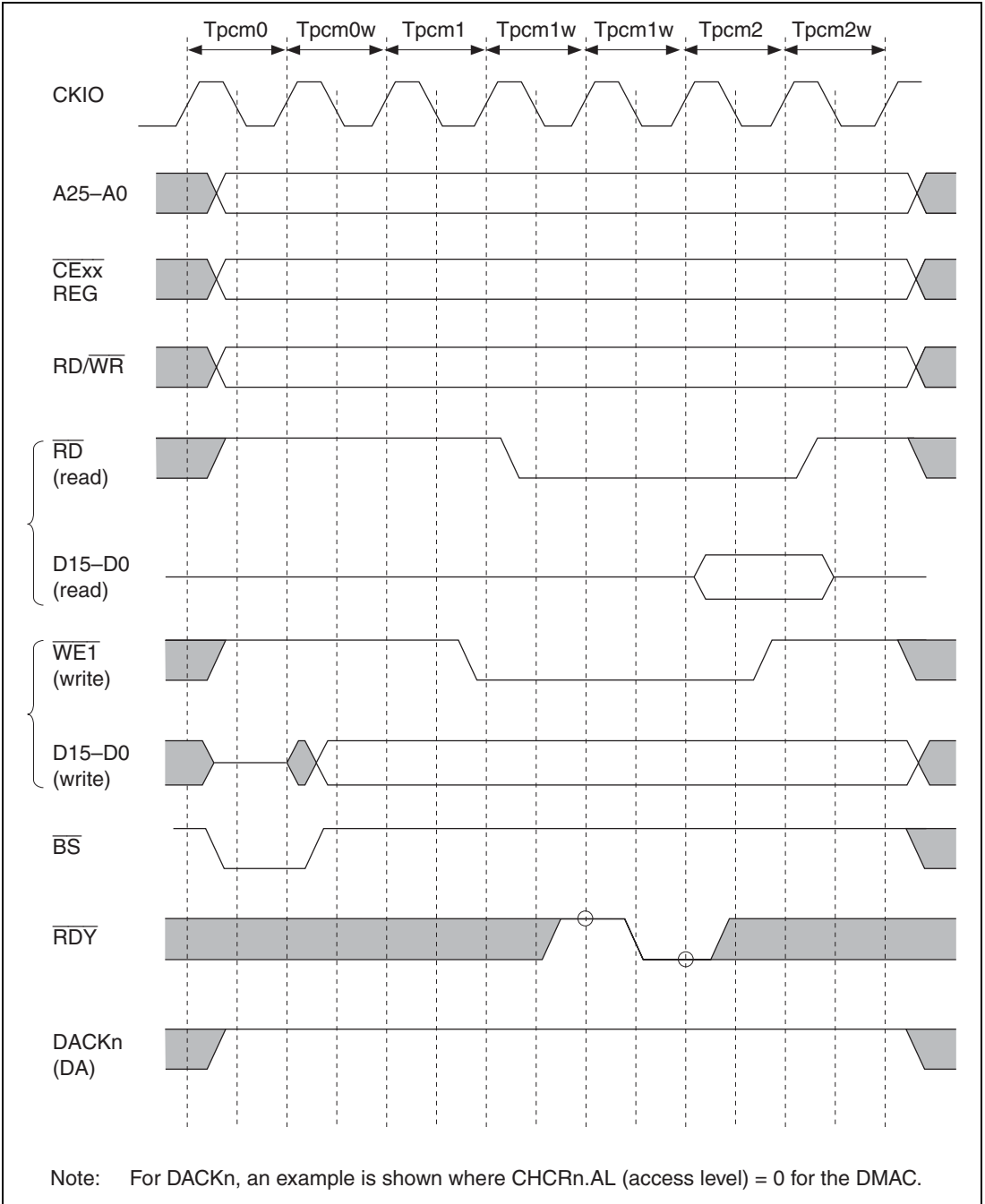


Figure 13.46 Wait Timing for PCMCIA Memory Card Interface

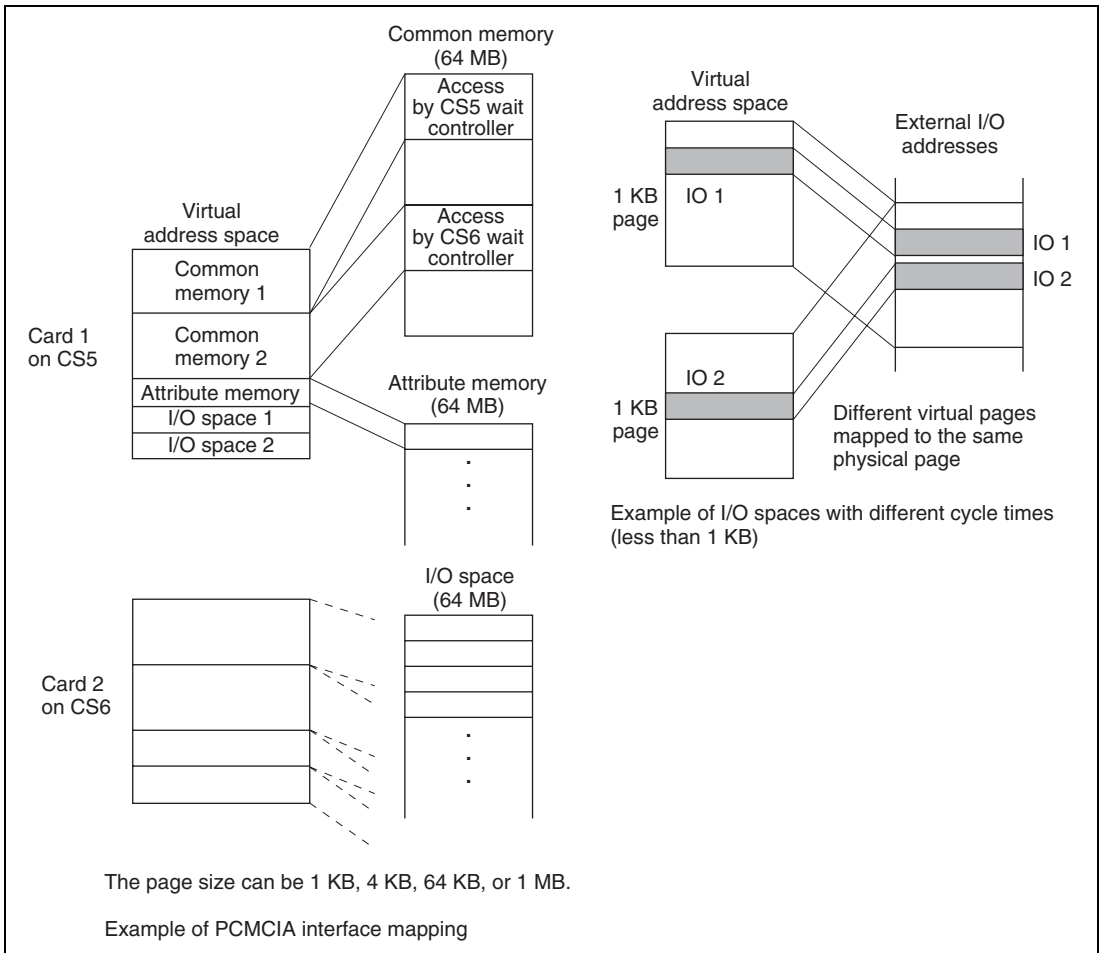


Figure 13.47 PCMCIA Space Allocation

I/O Card Interface Timing: Figures 13.48 and 13.49 show the timing for the PCMCIA I/O card interface.

When an I/O card interface access is made to a PCMCIA card, dynamic sizing of the I/O bus width is possible using the $\overline{\text{IOIS16}}$ pin. When a 16-bit bus width is set, if the $\overline{\text{IOIS16}}$ signal is high during a word-size I/O bus cycle, the I/O port is recognized as being 8 bits in width. In this case, a data access for only 8 bits is performed in the I/O bus cycle being executed, followed automatically by a data access for the remaining 8 bits. Dynamic bus sizing is also performed in the case of byte-size access to address $2n + 1$.

Figure 13.50 shows the basic timing for dynamic bus sizing.

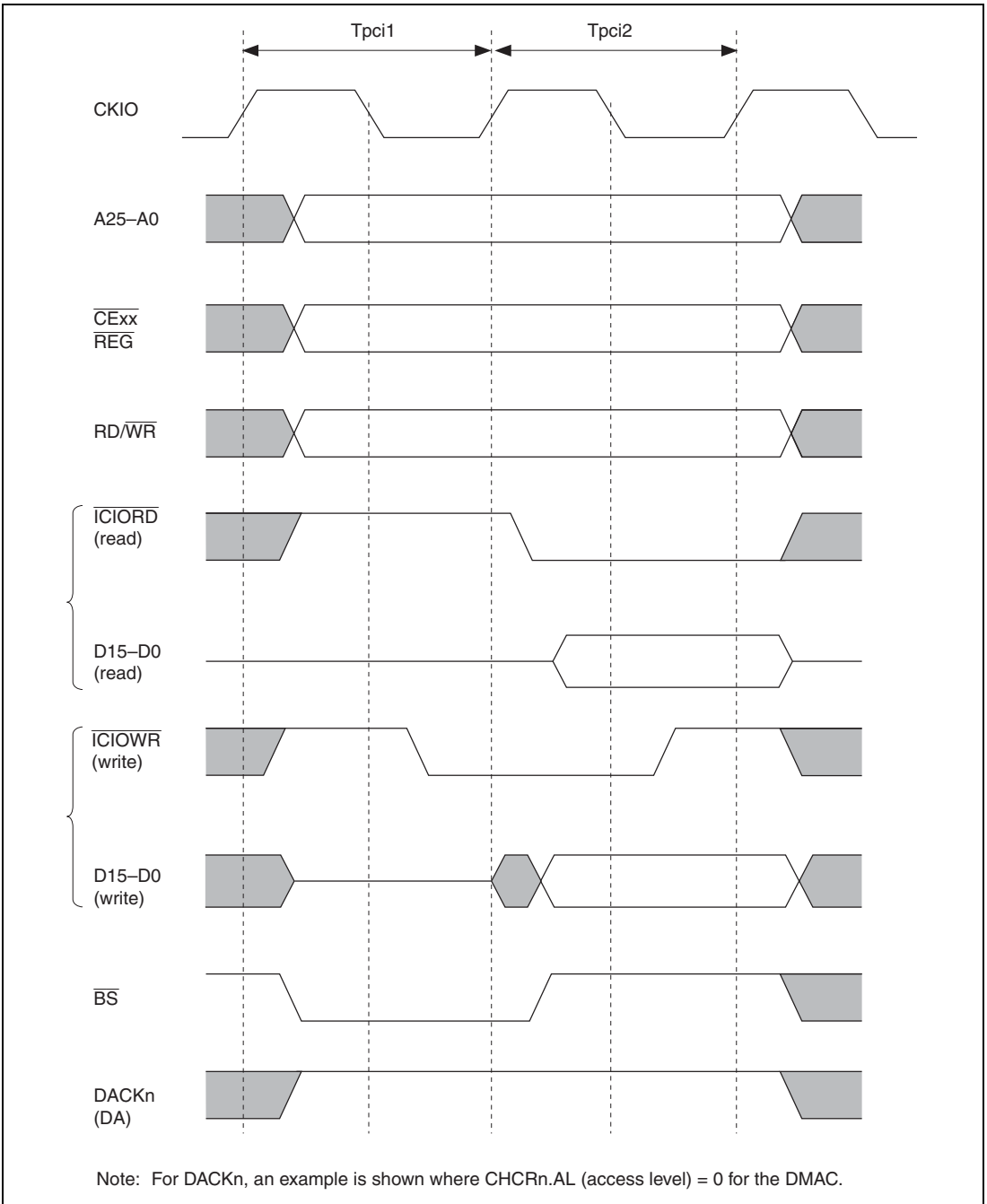


Figure 13.48 Basic Timing for PCMCIA I/O Card Interface

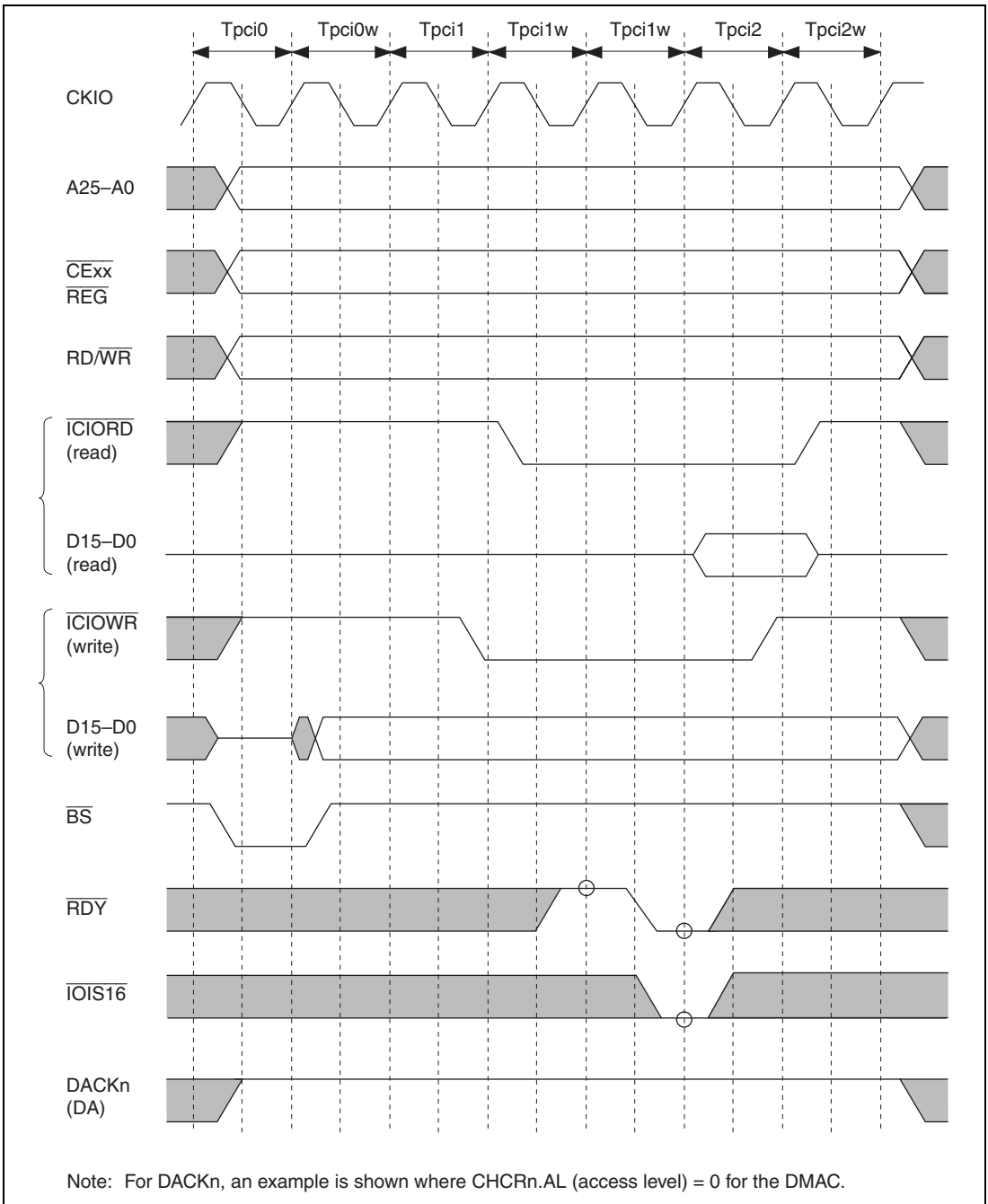


Figure 13.49 Wait Timing for PCMCIA I/O Card Interface

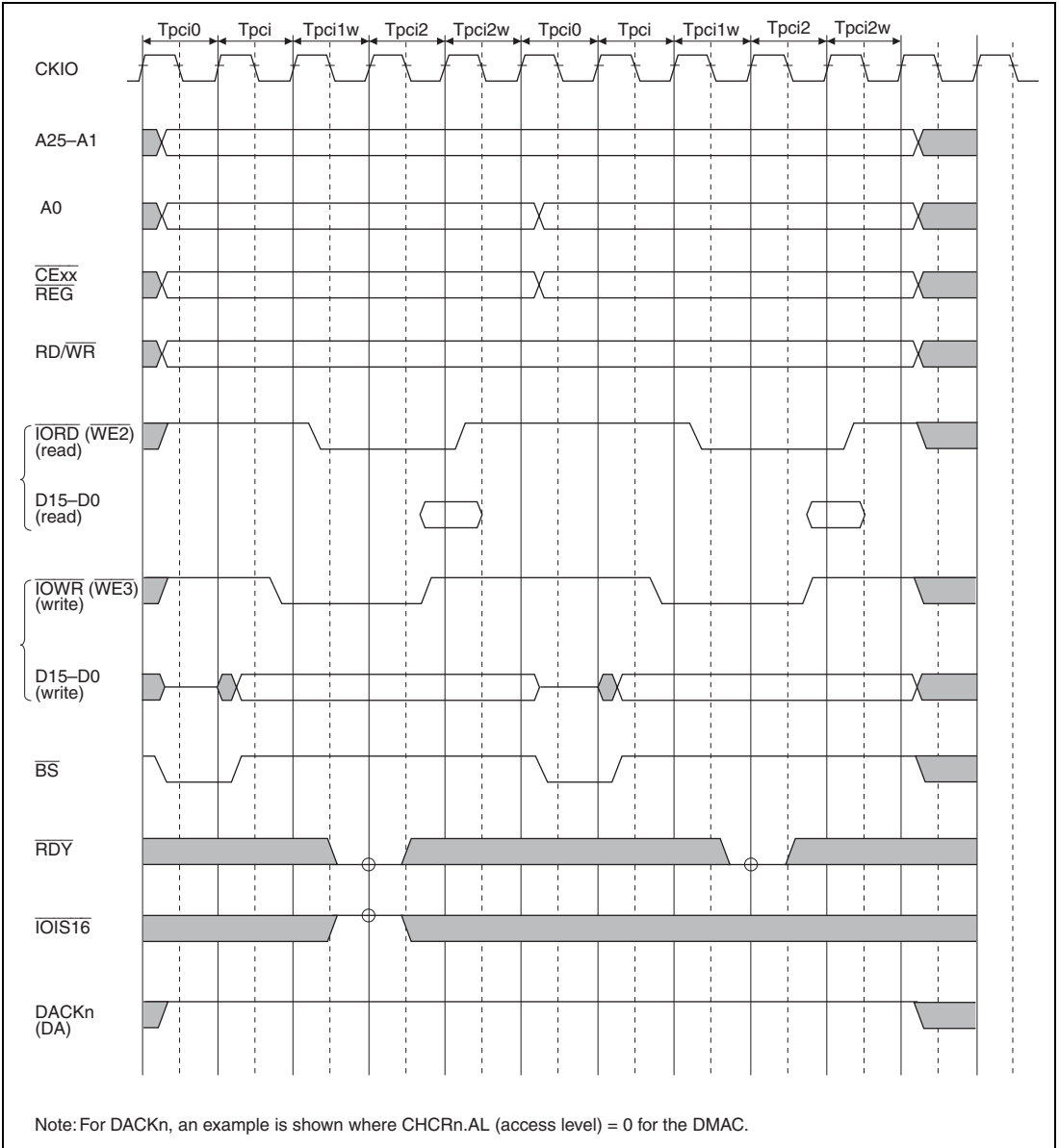


Figure 13.50 Dynamic Bus Sizing Timing for PCMCIA I/O Card Interface

13.3.8 MPX Interface

If the MD6 pin is cleared to 0 in a power-on reset by means of the $\overline{\text{RESET}}$ pin, the MPX interface is selected for area 0. The MPX interface is selected for areas 1 to 6 by means of the MPX bit in BCR1 and MEMMODE, A4MPX, and A1MPX in BCR3. The MPX interface offers a multiplexed address/data type bus protocol, and permits easy connection to an external memory controller chip that uses a single 32-bit multiplexed address/data bus. A bus cycle consists of an address phase and a data phase, with address information output to D25–D0 and the access size output to D31–D29 in the address phase. The $\overline{\text{BS}}$ signal is asserted for one cycle to indicate the address phase. The $\overline{\text{CSn}}$ signal is asserted at the rise of Tm1 and negated after the end of the last data transfer in the data phase. Therefore, a negation period does not occur in the case of minimum pitch access. The $\overline{\text{FRAME}}$ signal is asserted at the rise of Tm1 and negated when the last data transfer cycle starts in the data phase. Therefore, an external device for the MPX interface must hold the address information and access size output in the address phase within itself, and peripheral function data input/output for the data phase. For details of access sizes and data alignment, see section 13.3.1, Endian/Access Size and Data Alignment.

Values output to address pins A25–A0 are not guaranteed.

In 32-byte transfer, a total of 32 bytes are transferred consecutively according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on 32-byte boundary data. If the access size exceeds the set bus width in this case, burst access is performed with a number of data cycles following one address output. The bus is not released during this period.

| D31 | D30 | D29 | Access Size |
|-----|-----|-----|---------------|
| 0 | 0 | 0 | Byte |
| | | 1 | Word |
| | 1 | 0 | Longword |
| | | 1 | Quadword |
| 1 | X | X | 32-byte burst |

Legend: X: Don't care

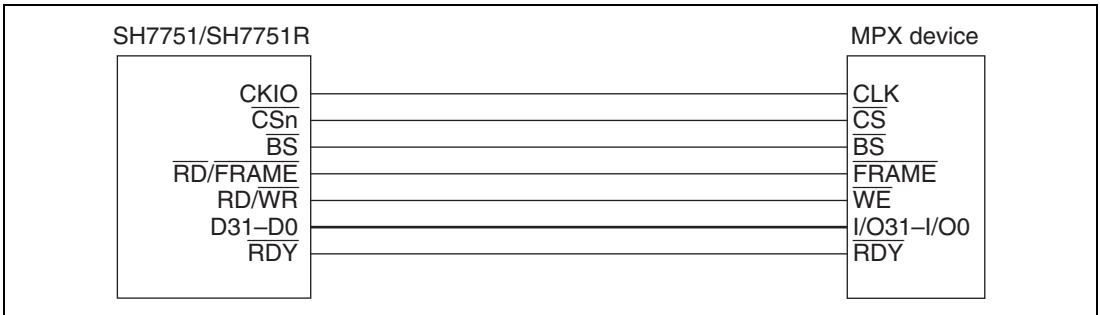


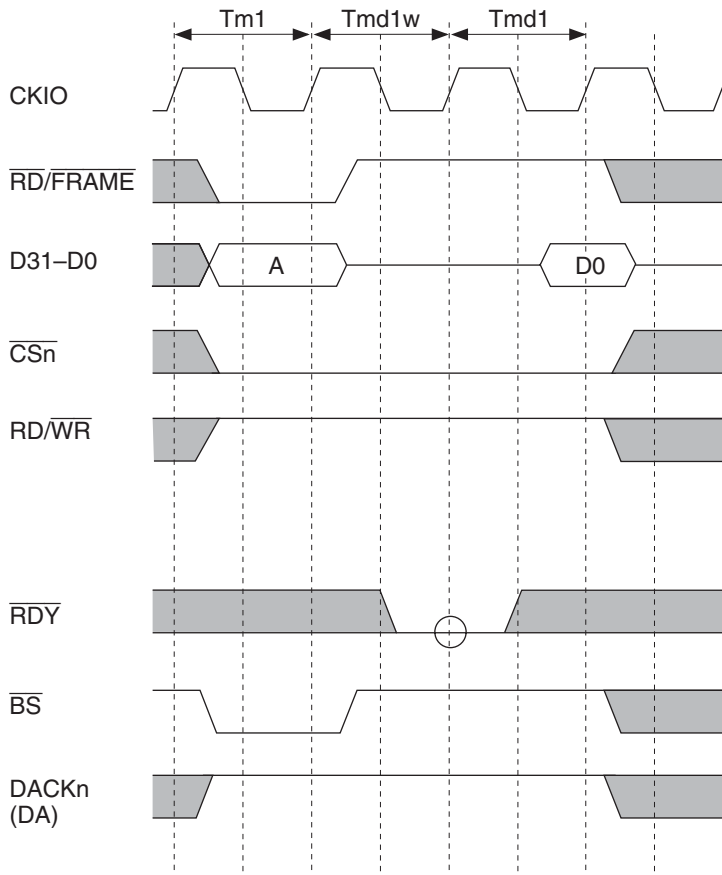
Figure 13.51 Example of 32-Bit Data Width MPX Connection

The MPX interface timing is shown below.

When the MPX interface is used for areas 1 to 6, a bus size of 32 bit should be specified in BCR2.

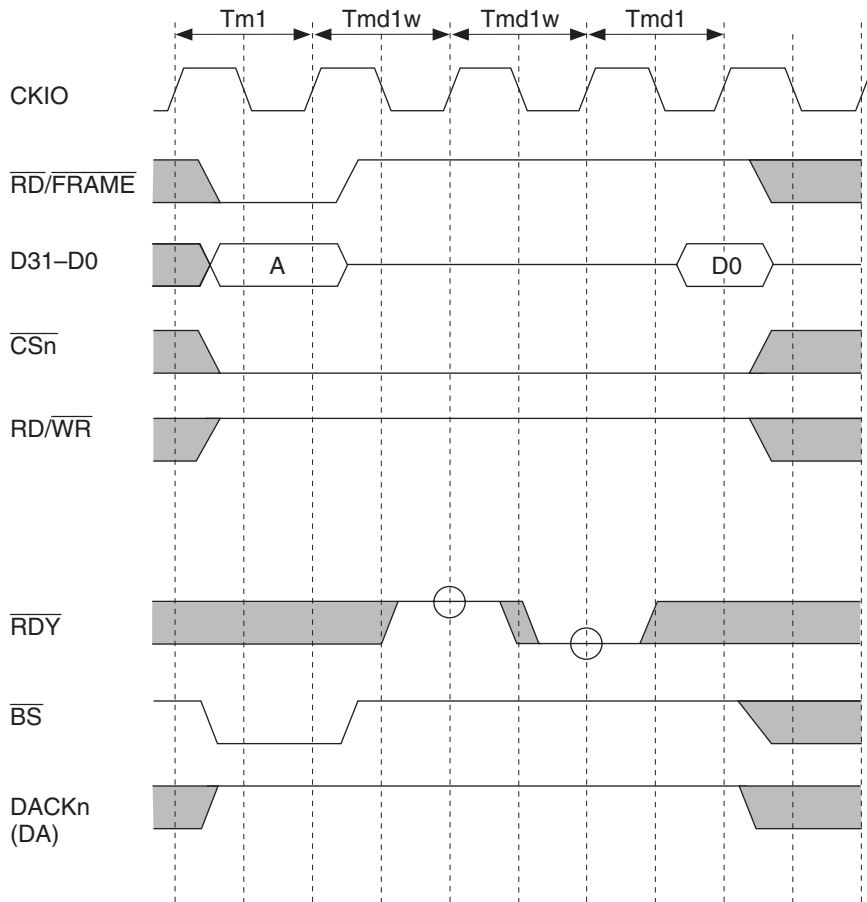
For wait control, waits specified by WCR2 and wait insertion by means of the $\overline{\text{RDY}}$ pin can be used.

In a read, one wait cycle is automatically inserted after address output, even if WCR2 is cleared to 0.



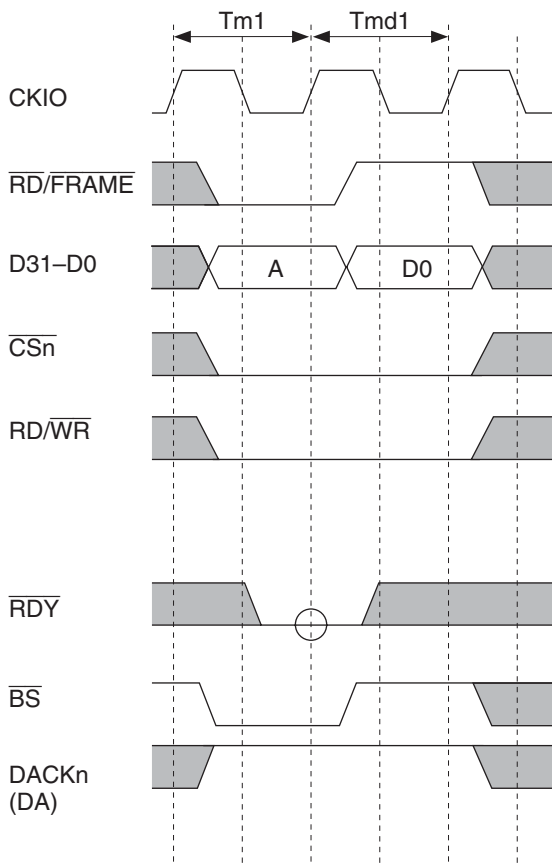
Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.52 MPX Interface Timing 1 (Single Read Cycle, AnW = 0, No External Wait)



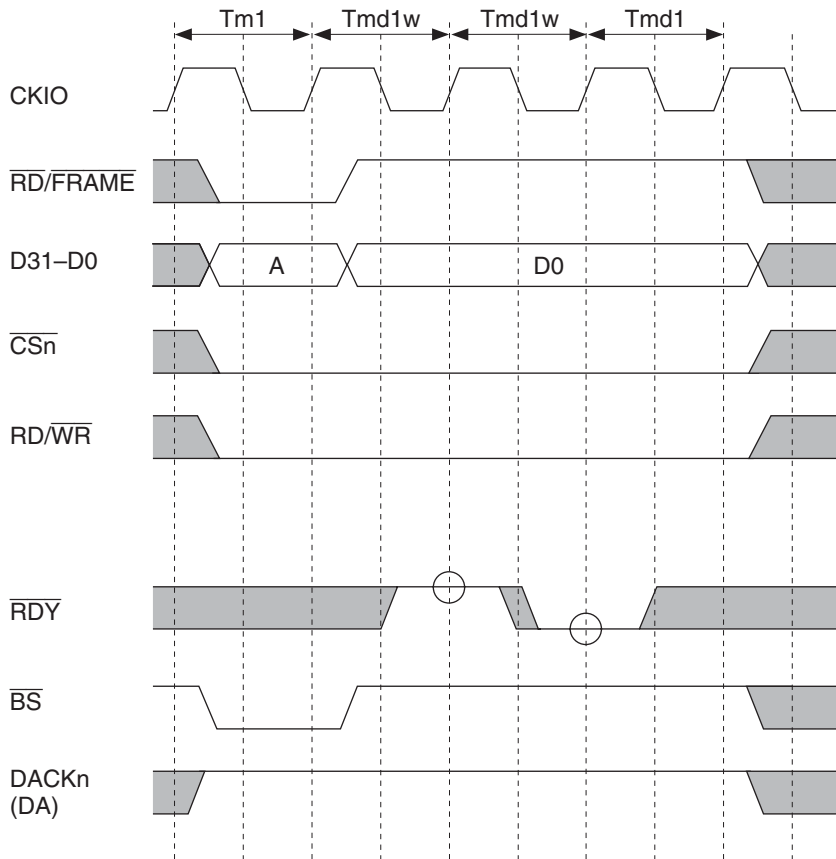
Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.53 MPX Interface Timing 2 (Single Read, AnW = 0, One External Wait Inserted)



Note: For DACK n , an example is shown where CHCR n .AL (access level) = 0 for the DMAC.

Figure 13.54 MPX Interface Timing 3 (Single Write Cycle, AnW = 0, No External Wait)



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.55 MPX Interface Timing 4
(Single Write, AnW = 1, One External Wait Inserted)

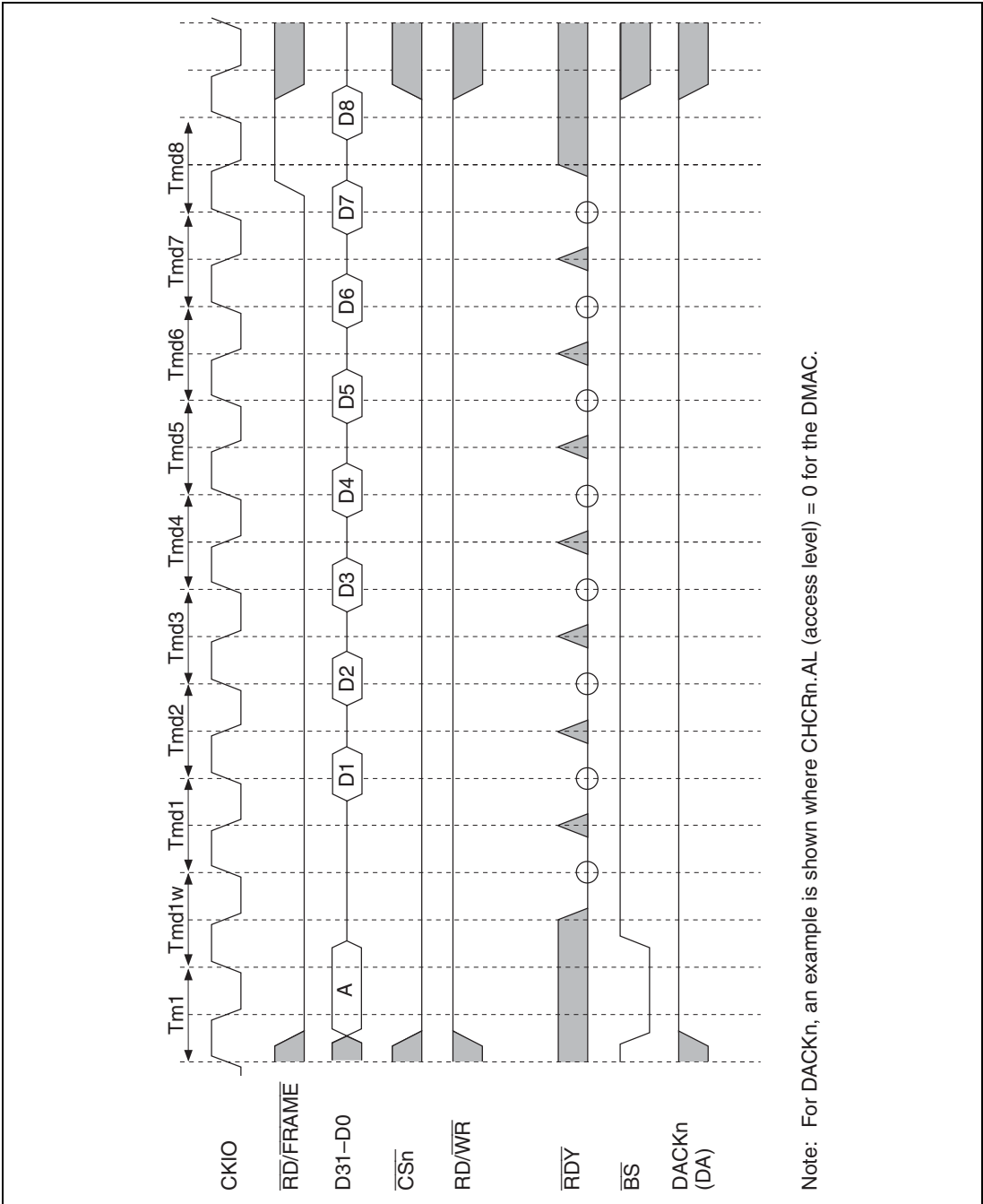


Figure 13.56 MPX Interface Timing 5 (Burst Read Cycle, AnW = 0, No External Wait)

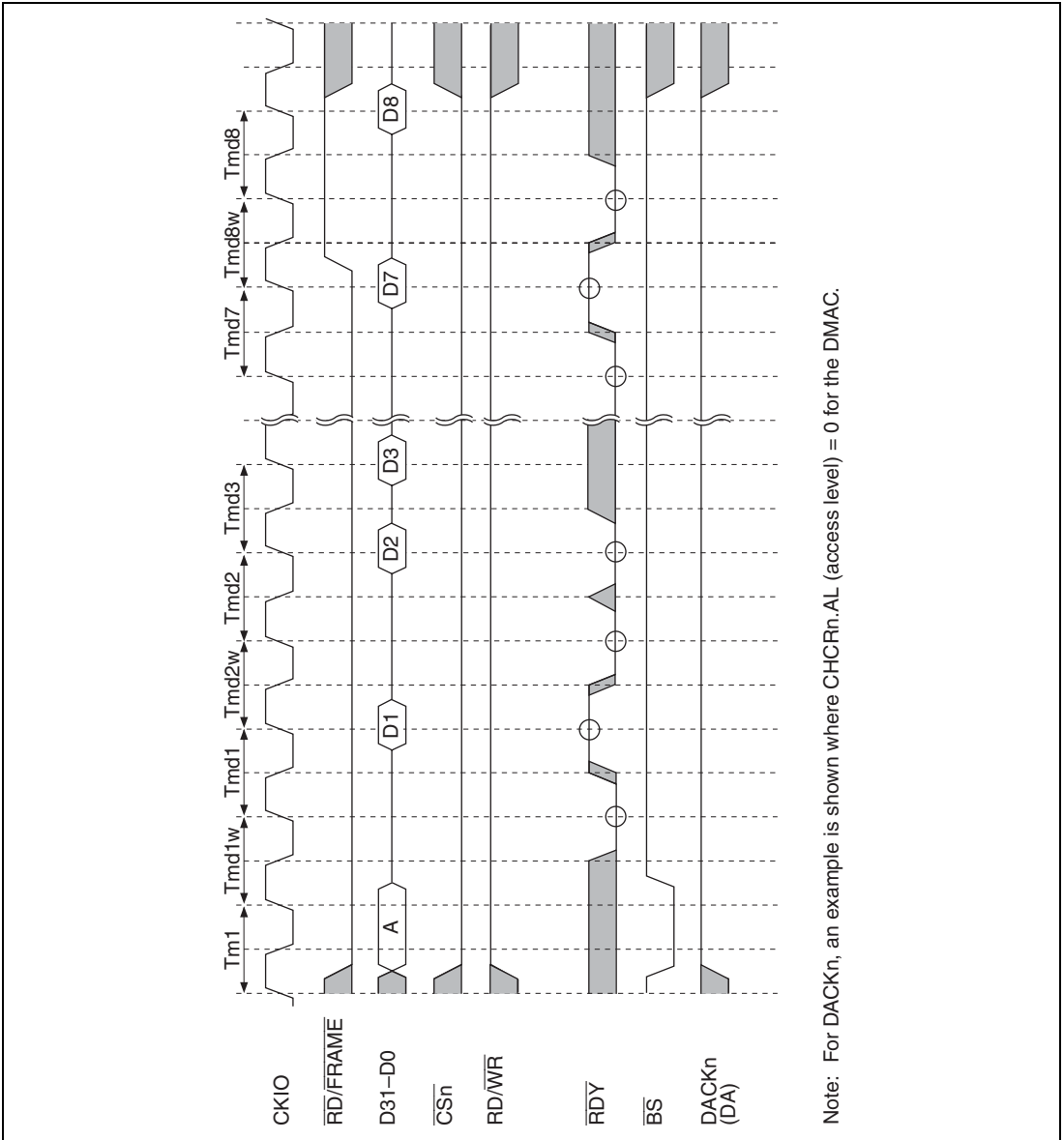


Figure 13.57 MPX Interface Timing 6 (Burst Read Cycle, AnW = 0, External Wait Control)

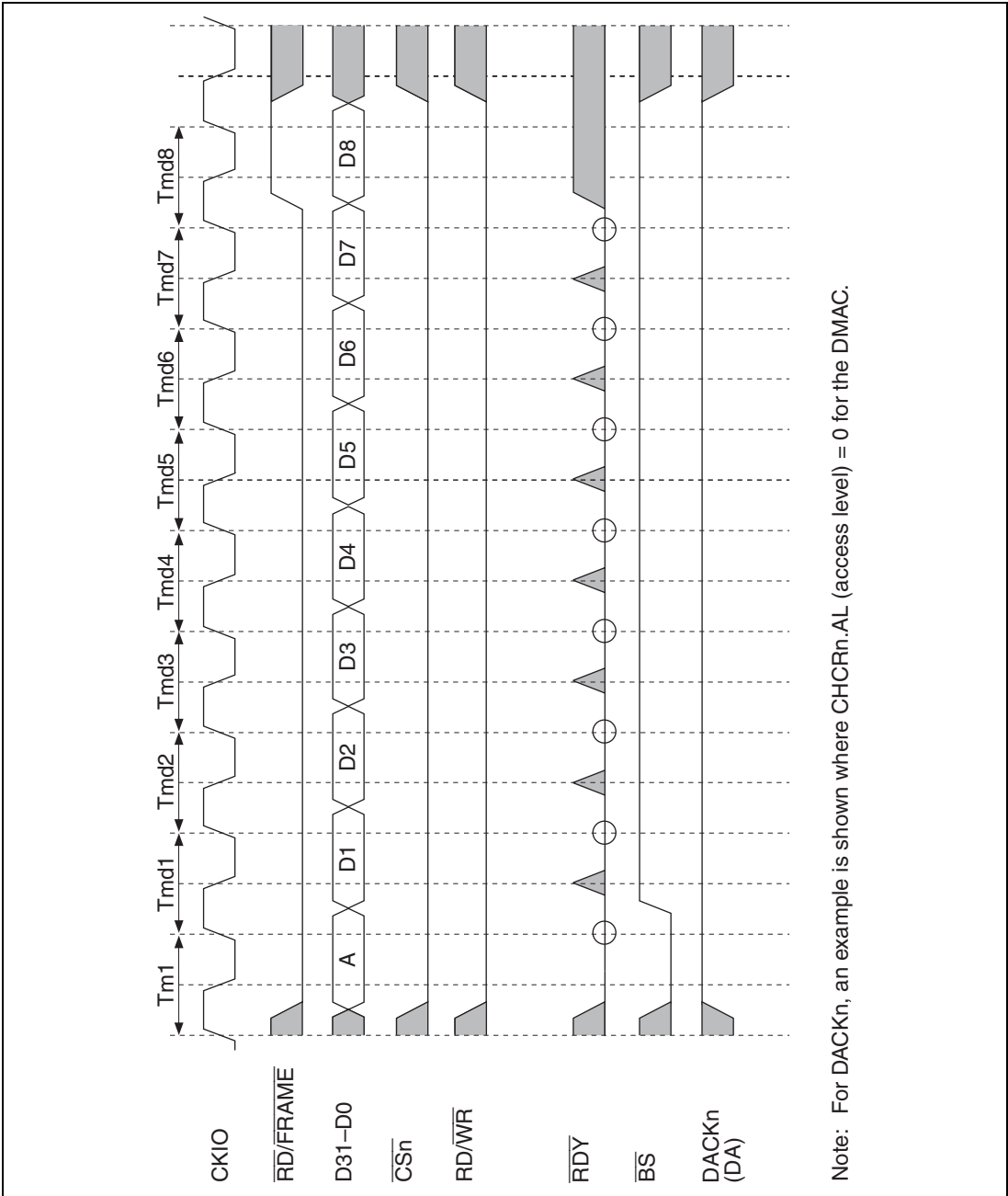
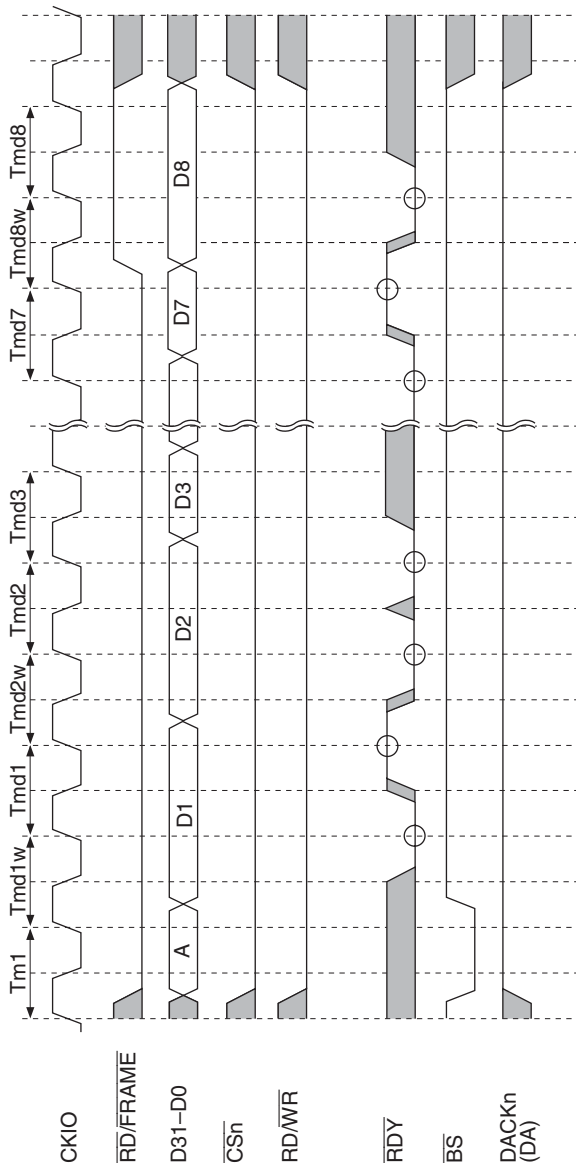
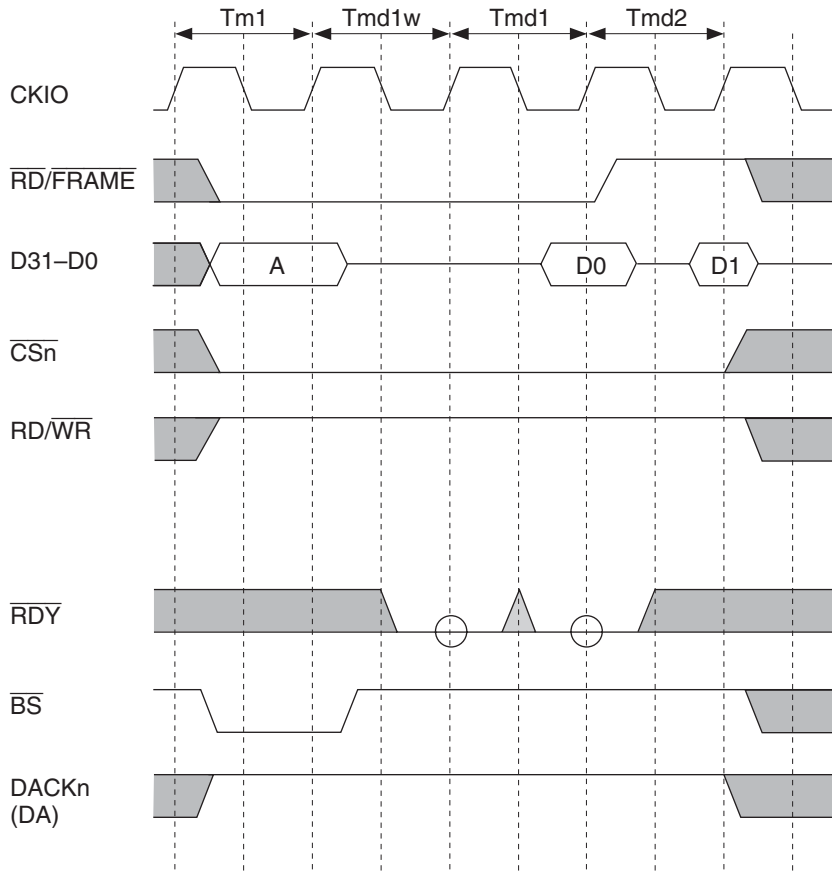


Figure 13.58 MPX Interface Timing 7 (Burst Write Cycle, AnW = 0, No External Wait)



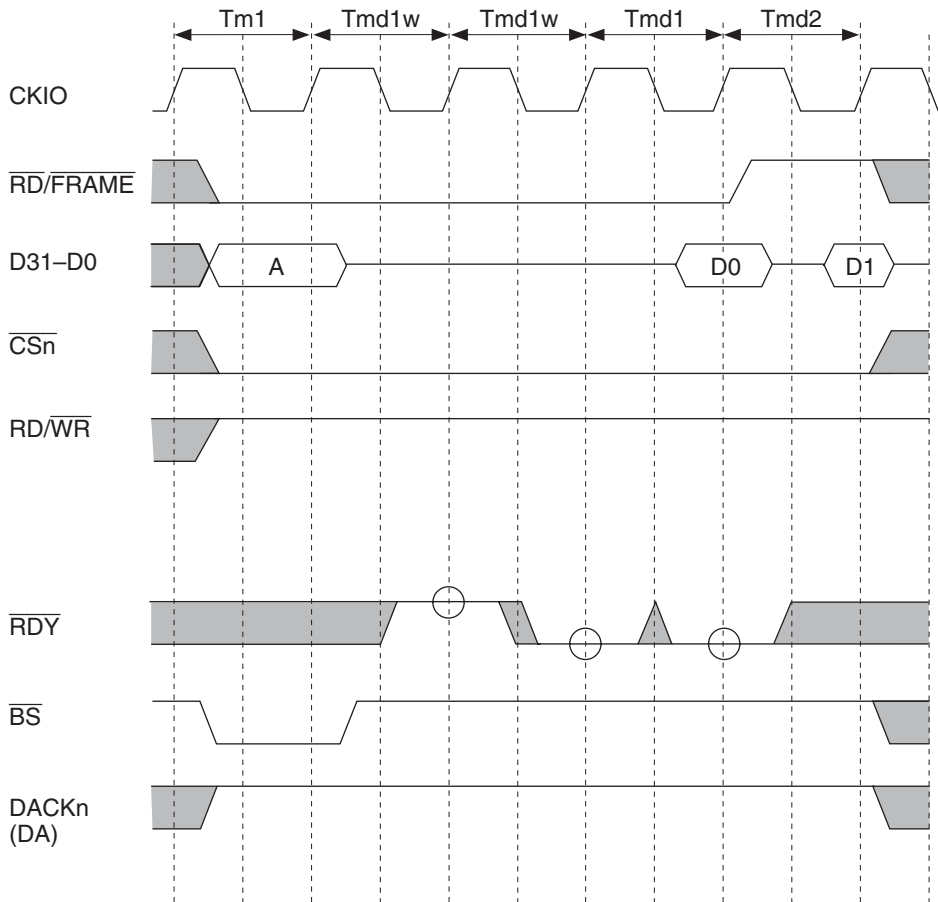
Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.59 MPX Interface Timing 8
 (Burst Write Cycle, AnW = 1, External Wait Control)



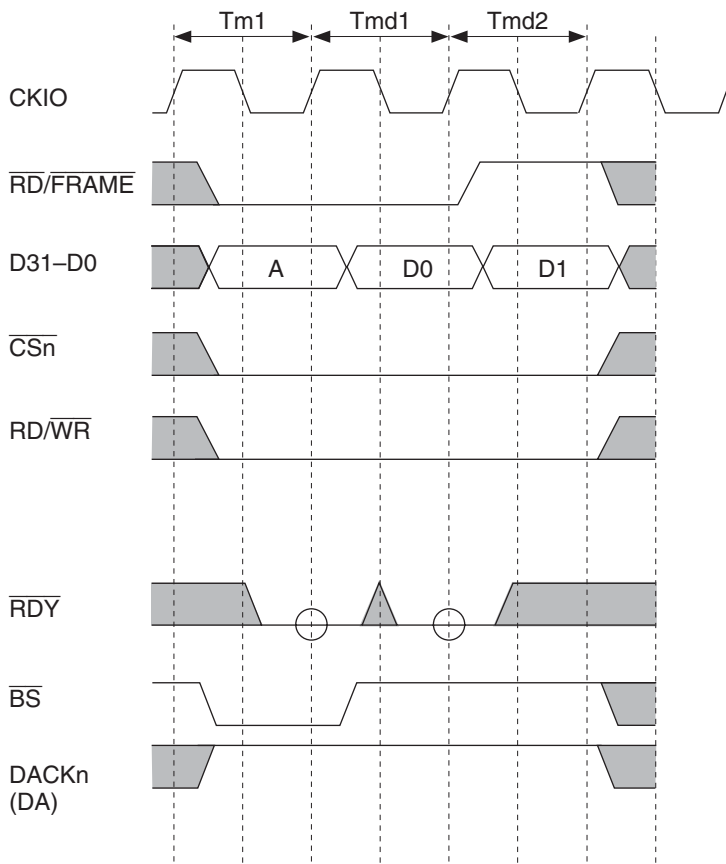
Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.60 MPX Interface Timing 9
(Burst Read Cycle, AnW = 0, No External Wait, Bus Width: 32 Bits,
Transfer Data Size: 64 Bits)



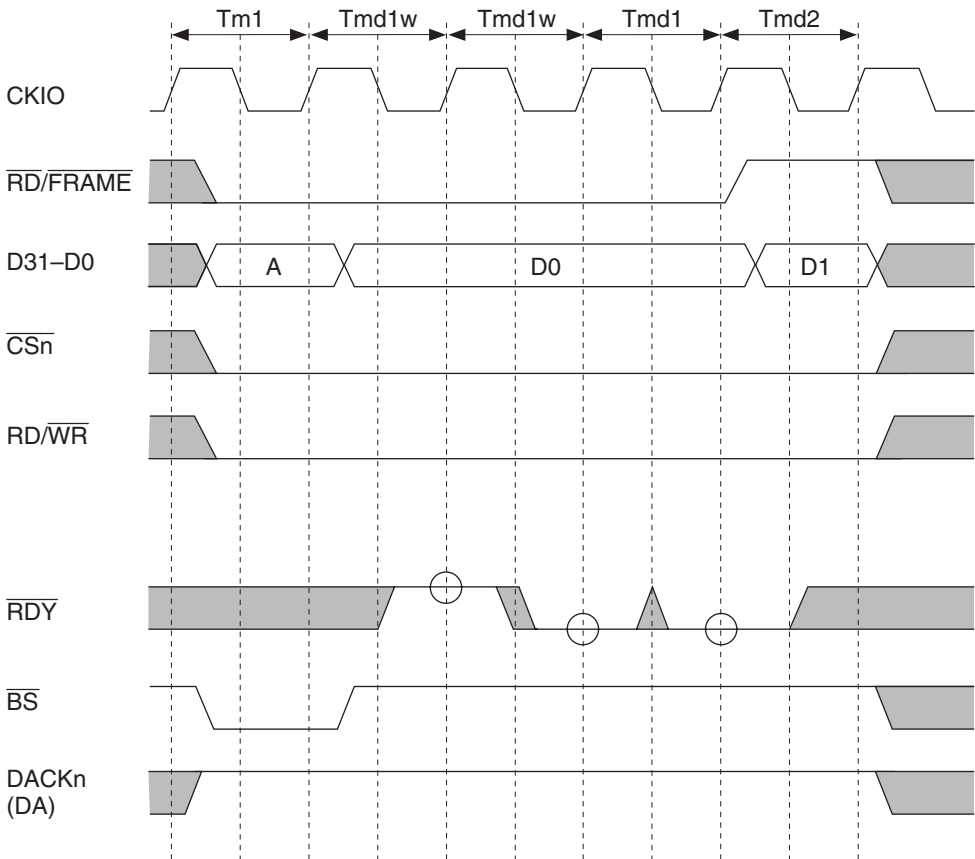
Note: For DACK_n, an example is shown where CHCR_n.AL (access level) = 0 for the DMAC.

Figure 13.61 MPX Interface Timing 10
(Burst Read Cycle, AnW = 0, One External Wait Inserted, Bus Width: 32 Bits,
Transfer Data Size: 64 Bits)



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.62 MPX Interface Timing 11
(Burst Write Cycle, AnW = 0, No External Wait, Bus Width: 32 Bits,
Transfer Data Size: 64 Bits)



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.63 MPX Interface Timing 12
(Burst Write Cycle, AnW = 1, One External Wait Inserted, Bus Width: 32 Bits,
Transfer Data Size: 64 Bits)

13.3.9 Byte Control SRAM Interface

The byte control SRAM interface is a memory interface that outputs a byte select strobe (\overline{WE}_n) in both read and write bus cycles. It has 16 bit data pins, and can be connected to SRAM which has an upper byte select strobe and lower byte select function such as UB and LB.

Areas 1 and 4 can be designated as byte control SRAM interface. However, when these areas are set to MPX mode, MPX mode has priority.

The byte control SRAM interface write timing is the same as for the normal SRAM interface.

In read operations, the \overline{WE}_n pin timing is different. In a read access, only the \overline{WE} signal for the byte being read is asserted. Assertion is synchronized with the fall of the CKIO clock, as for the \overline{WE} signal, while negation is synchronized with the rise of the CKIO clock, using the same timing as the \overline{RD} signal.

32-byte transfer is performed consecutively for a total of 32 bytes according to the set bus width. The first access is performed on the data for which there was an access request. The remaining accesses are performed in wrap-around fashion on the data at the 32-byte boundary. The bus is not released during this period.

Figure 13.64 shows an example of byte control SRAM connection to this LSI, and figures 13.65 to 13.67 show examples of byte control SRAM read cycles.

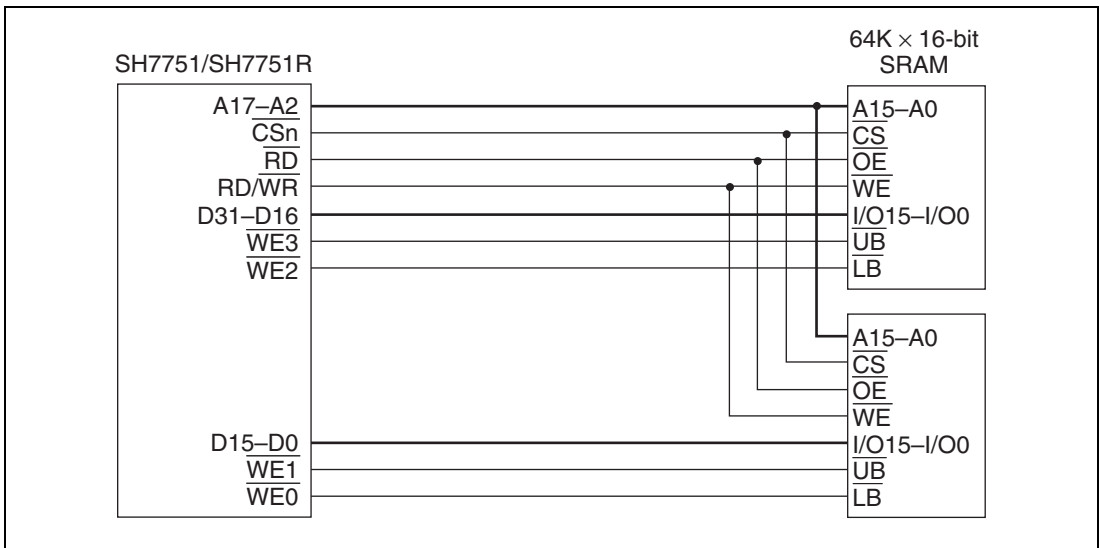
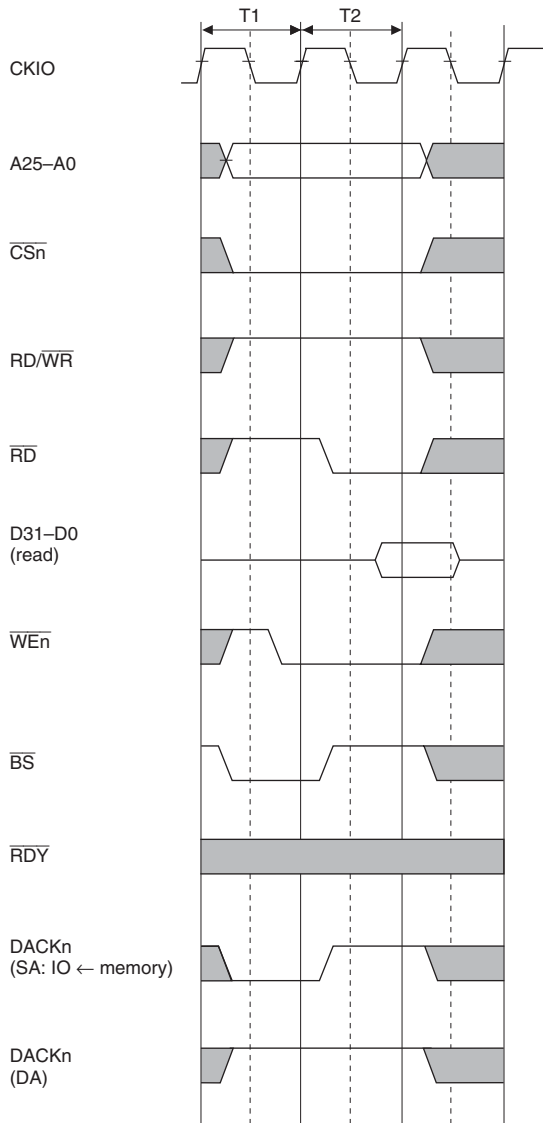


Figure 13.64 Example of 32-Bit Data Width Byte Control SRAM



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

Figure 13.65 Byte Control SRAM Basic Read Cycle (No Wait)

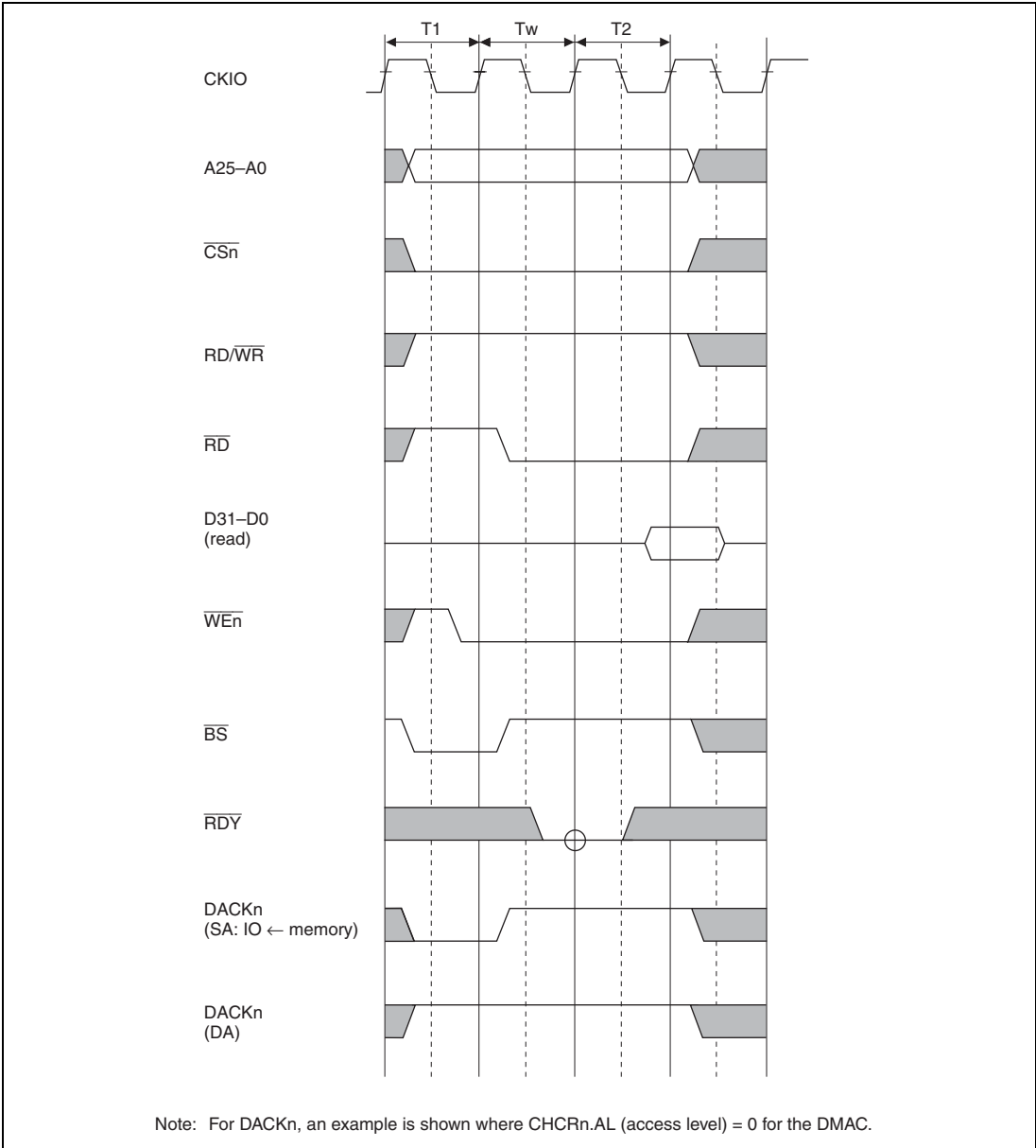
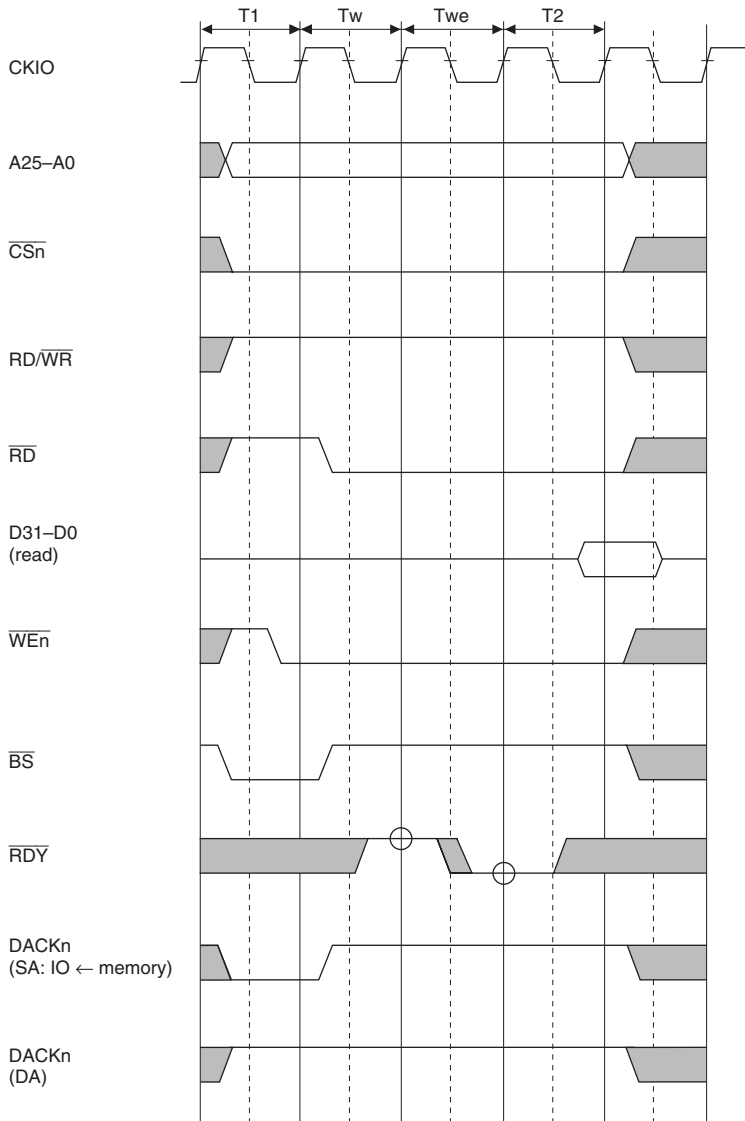


Figure 13.66 Byte Control SRAM Basic Read Cycle (One Internal Wait Cycle)



Note: For DACKn, an example is shown where CHCRn.AL (access level) = 0 for the DMAC.

**Figure 13.67 Byte Control SRAM Basic Read Cycle
(One Internal Wait + One External Wait)**

13.3.10 Waits between Access Cycles

A problem associated with higher external memory bus operating frequencies is that data buffer turn-off on completion of a read from a low-speed device may be too slow, causing a collision with the data in the next access, and so resulting in lower reliability or incorrect operation. To avoid this problem, a data collision prevention feature has been provided. This memorizes the preceding access area and the kind of read/write, and if there is a possibility of a bus collision when the next access is started, inserts a wait cycle before the access cycle to prevent a data collision. Wait cycle insertion consists of inserting idle cycles between access cycles, as shown in section 13.2.5, Wait Control Register 1 (WCR1). When this LSI performs consecutive write cycles, the data transfer direction is fixed (from this LSI to other memory) and there is no problem. With read accesses to the same area, also, in principle data is output from the same data buffer, and wait cycle insertion is not performed. If there is originally space between accesses, according to the setting of bits AnIW2–AnIW0 ($n = 0$ to 6) in WCR1, the number of idle cycles inserted is the specified number of idle cycles minus the number of empty cycles.

When bus arbitration is performed, the bus is released after waits are inserted between cycles.

In single address mode DMA transfer, when data transfer is performed from an I/O device to memory the data on the bus is determined by the speed of the I/O device. With a low-speed I/O device, an inter-cycle idle wait equivalent to the output buffer turn-off time must be inserted. Even with high-speed memory, when DMA transfer is considered, it may be necessary to insert an inter-cycle wait to adjust to the speed of a low-speed device, preventing the memory from being used at full speed.

Bits DMAIW2–DMAIW0 in wait control register 1 (WCR1) allow an inter-cycle wait setting to be made when transferring data from an I/O device to memory using single address mode DMA transfer. From 0 to 15 waits can be inserted. The number of waits specified by DMAIW2–DMAIW0 are inserted in single address DMA transfers to all areas.

In dual address mode DMA transfer, the normal inter-cycle wait specified by AnIW2–AnIW0 ($n = 0$ to 6) is inserted.

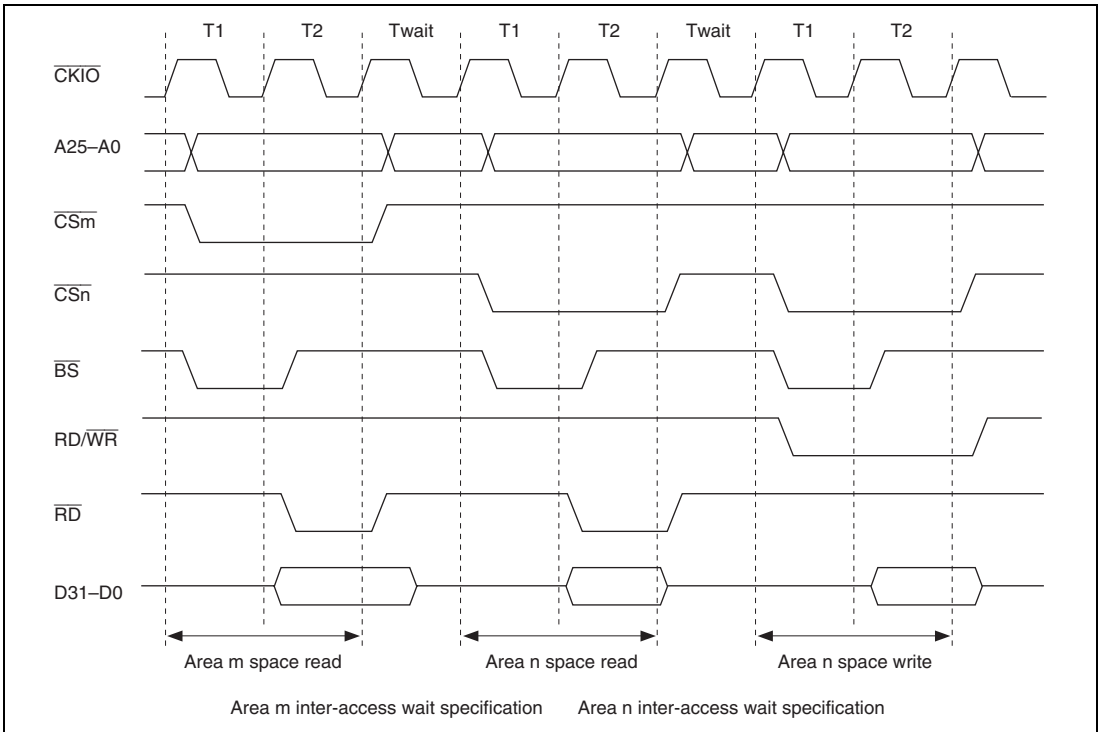


Figure 13.68 Waits between Access Cycles

13.3.11 Bus Arbitration

This LSI is provided with a bus arbitration function that grants the bus to an external device when it makes a bus request.

There are two bus arbitration modes: master mode, and slave mode. In master mode the bus is held on a constant basis, and is released to another device in response to a bus request. In slave mode the bus is not held on a constant basis; a bus request is issued each time an external bus cycle occurs, and the bus is released again at the end of the access.

Master mode and slave mode can be specified by the external mode pins. See appendix C, Mode Pin Settings, for the external mode pin settings. In master mode and slave mode, the bus goes to the high-impedance state when not being held. Instead of a slave mode chip. In the following description, an external device that issues bus requests is also referred to as a slave.

This LSI has three internal bus masters: the CPU, DMAC, and PCIC. When synchronous DRAM or DRAM is connected and refresh control is performed, refresh requests constitute a fourth bus master. In addition to these are bus requests from external devices in master mode. If requests

occur simultaneously, priority is given, in high-to-low order, to a bus request from an external device, a refresh request, the DMAC, and the CPU. See section 13.3.15, Notes on Usage.

To prevent incorrect operation of connected devices when the bus is transferred between master and slave, all bus control signals are negated before the bus is released. When mastership of the bus is received, also, bus control signals begin driving the bus from the negated state. Since signals are driven to the same value by the master and slave exchanging the bus, output buffer collisions can be avoided.

Bus transfer is executed between bus cycles.

When the bus release request signal ($\overline{\text{BREQ}}$) is asserted, this LSI releases the bus as soon as the currently executing bus cycle ends, and outputs the bus use permission signal ($\overline{\text{BACK}}$). However, bus release is not performed during multiple bus cycles generated because the data bus width is smaller than the access size (for example, when performing longword access to 8-bit bus width memory) or during a 32-byte transfer such as a cache fill or write-back. In addition, bus release is not performed between read and write cycles during execution of a TAS instruction, or between read and write cycles when DMAC dual address transfer is executed. When $\overline{\text{BREQ}}$ is negated, $\overline{\text{BACK}}$ is negated and use of the bus is resumed. See appendix D, Pin Functions, for the pin states when the bus is released.

When a refresh request is generated, this LSI performs a refresh operation as soon as the currently executing bus cycle ends. However, refresh operations are deferred during multiple bus cycles generated because the data bus width is smaller than the access size (for example, when performing longword access to 8-bit bus width memory) and during a 32-byte transfer such as a cache fill or write-back, and also between read and write cycles during execution of a TAS instruction, and between read and write cycles when DMAC dual address transfer is executed. Refresh operations are also deferred in the bus-released state.

If the synchronous DRAM interface is set to the RAS down mode the PALL command is issued before a refresh cycle occurs or before the bus is released by bus arbitration.

As the CPU in this LSI is connected to cache memory by a dedicated internal bus, reading from cache memory can still be carried out when the bus is being used by another bus master inside or outside this LSI. When writing from the CPU, an external write cycle is generated when write-through has been set for the cache in this LSI, or when an access is made to a cache-off area. There is consequently a delay until the bus is returned.

When this LSI wants to take back the bus in response to an internal memory refresh request, it negates $\overline{\text{BACK}}$. On receiving the $\overline{\text{BACK}}$ negation, the device that asserted the external bus release request negates $\overline{\text{BREQ}}$ to release the bus. The bus is thereby returned to this LSI, which then carries out the necessary processing.

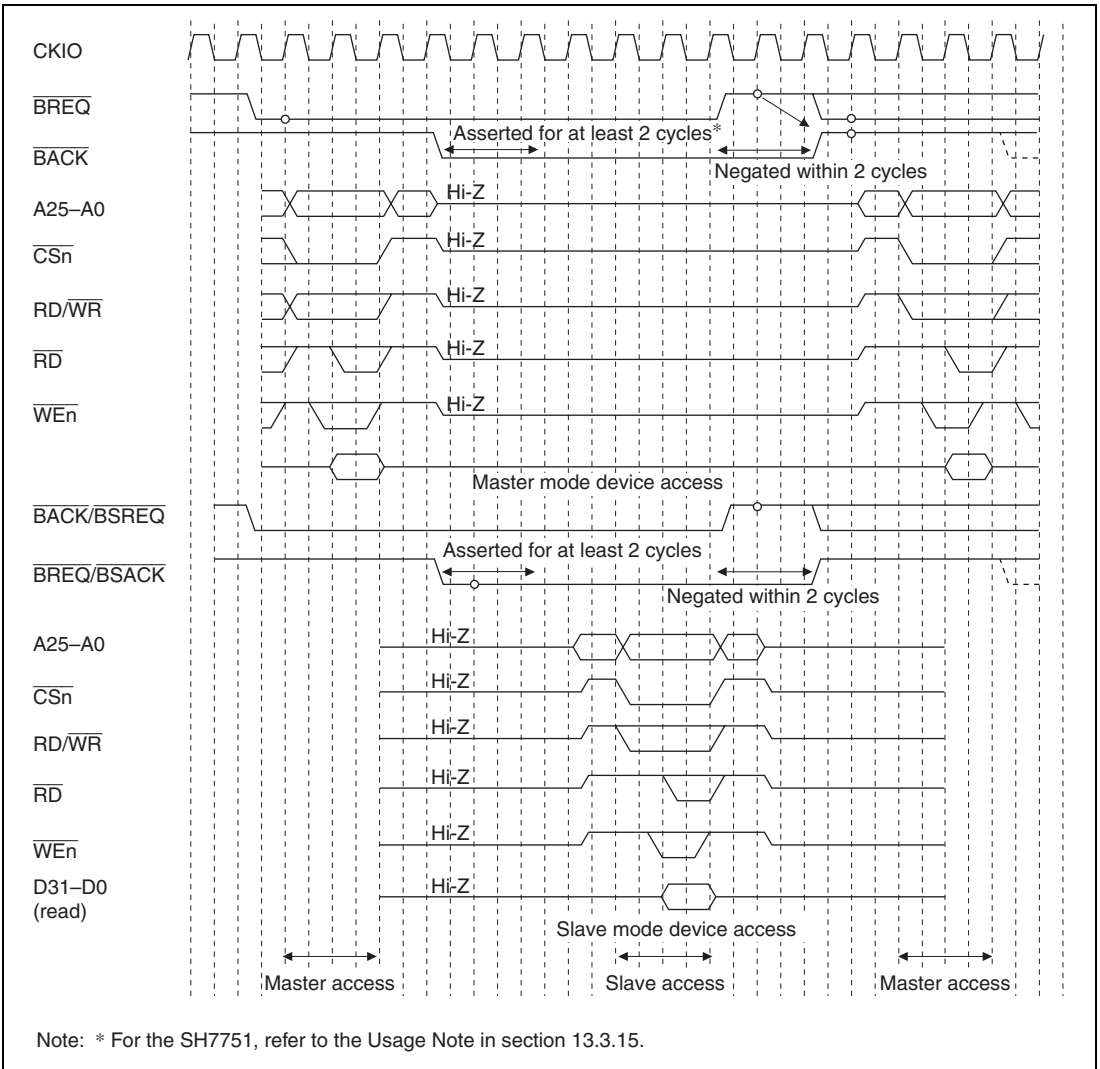


Figure 13.69 Arbitration Sequence

13.3.12 Master Mode

The master mode processor holds the bus itself unless it receives a bus request.

On receiving an assertion (low level) of the bus request signal ($\overline{\text{BREQ}}$) from off-chip, the master mode processor releases the bus and asserts (drives low) the bus use permission signal ($\overline{\text{BACK}}$) as soon as the currently executing bus cycle ends. If a bus release request due to a refresh request has not been issued, on receiving the $\overline{\text{BREQ}}$ negation (high level) indicating that the slave has released the bus, the processor negates (drives high) the $\overline{\text{BACK}}$ signal and resumes use of the bus.

If a bus request is issued due to a memory refresh request in the bus-released state, the processor negates the bus use permission signal ($\overline{\text{BACK}}$), and on receiving the $\overline{\text{BREQ}}$ negation indicating that the slave has released the bus, resumes use of the bus.

When the bus is released, all bus interface related output signals and input/output signals go to the high-impedance state, except for the synchronous DRAM interface CKE signal and bus arbitration $\overline{\text{BACK}}$ signal, and DACK0 and DACK1 which control DMA transfers.

With DRAM, the bus is released after precharging is completed. With synchronous DRAM, also, a precharge command is issued for the active bank and the bus is released after precharging is completed.

The actual bus release sequence is as follows.

First, the bus use permission signal is asserted in synchronization with the rising edge of the clock. The address bus and data bus go to the high-impedance state in synchronization after this $\overline{\text{BACK}}$ assertion. At the same time, the bus control signals ($\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RAS1}}$, $\overline{\text{WEn}}$, $\overline{\text{RD}}$, $\text{RD}/\overline{\text{WR}}$, $\overline{\text{CE2A}}$, and $\overline{\text{CE2B}}$) go to the high-impedance state. These bus control signals are negated no later than one cycle before going to high-impedance. Bus request signal sampling is performed on the rising edge of the clock.

The sequence for re-acquiring the bus from the slave is as follows.

As soon as $\overline{\text{BREQ}}$ negation is detected on the rising edge of the clock, $\overline{\text{BACK}}$ is negated and from next rising edge of the clock, bus control signal driving is started. Driving of the address bus and data bus starts at the next rising edge of an in-phase clock. The bus control signals are asserted and the bus cycle is actually started, at the earliest, at the clock rising edge at which the address and data signals are driven.

In order to reacquire the bus and start execution of a refresh operation or bus access, the $\overline{\text{BREQ}}$ signal must be negated for at least two cycles.

If a refresh request is generated when $\overline{\text{BACK}}$ has been asserted and the bus has been released, the $\overline{\text{BACK}}$ signal is negated even while the $\overline{\text{BREQ}}$ signal is asserted to request the slave to relinquish the bus. When this LSI is used in master mode, consecutive bus accesses may be attempted to reduce the overhead due to arbitration in the case of a slave designed independently by the user. When connecting a slave for which the total duration of consecutive accesses exceeds the refresh cycle, the design should provide for the bus to be released as soon as possible after negation of the $\overline{\text{BACK}}$ signal is detected.

13.3.13 Slave Mode

In slave mode, the bus is normally in the released state, and an external device cannot be accessed unless the bus is acquired through execution of the bus arbitration sequence. In a reset, also, the bus-released state is established and the bus arbitration sequence is started from the reset vector fetch.

To acquire the bus, the slave device asserts (drives low) the $\overline{\text{BSREQ}}$ signal in synchronization with the rising edge of the clock. The bus use permission $\overline{\text{BSACK}}$ signal is sampled for assertion (low level) in synchronization with the rising edge of the clock. When $\overline{\text{BSACK}}$ assertion is detected, the bus control signals are driven at the negated level after two cycles. The bus cycle is started at the next rising edge of the clock. The last signal negated at the end of the access cycle is synchronized with the rising edge of the clock. When the bus cycle ends, the $\overline{\text{BSREQ}}$ signal is negated and the release of the bus is reported to the master. On the next rising edge of the clock, the control signals are set to high-impedance.

In order for the slave mode processor to begin access, the $\overline{\text{BSACK}}$ signal must be asserted for at least two cycles.

For a slave access cycle in DRAM or synchronous DRAM, the bus is released on completion of precharging, as in the case of the master.

Refresh control is left to the master mode device, and any refresh control settings made in slave mode are ignored.

Do not use DRAM/synchronous DRAM RAS down mode in slave mode.

Synchronous DRAM mode register settings should be made by the master mode device. Do not use the DMAC's DDT mode in slave mode.

13.3.14 Cooperation between Master and Slave

To enable system resources to be controlled in a harmonious fashion by master and slave, their respective roles must be clearly defined. Before DRAM or synchronous DRAM is used, initialization operations must be carried out. Responsibility must also be assigned when a standby operation is performed to implement the power-down state.

The design of this LSI provides for all control, including initialization, refreshing, and standby control, to be carried out by the master mode device.

If this LSI is specified as the master in a power-on reset, it will not accept bus requests from the slave until the $\overline{\text{BREQ}}$ enable bit (BCR1.BREQEN) is set to 1.

To ensure that the slave processor does not access memory requiring initialization before use, such as DRAM and synchronous DRAM, until initialization is completed, write 1 to the $\overline{\text{BREQ}}$ enable bit after initialization ends.

Before setting self-refresh mode in standby mode, etc., write 0 to the $\overline{\text{BREQ}}$ enable bit to invalidate the $\overline{\text{BREQ}}$ signal from the slave. Write 1 to the $\overline{\text{BREQ}}$ enable bit only after the master has performed the necessary processing (refresh settings, etc.) for exiting self-refresh mode.

13.3.15 Notes on Usage

Refresh: Auto refresh operations stop when a transition is made to standby mode, hardware standby mode, or deep-sleep mode. If the memory system requires refresh operations, set the memory in the self-refresh state prior to making the transition to standby mode, hardware standby mode, or deep-sleep mode.

Bus Arbitration: On transition to standby mode or deep-sleep mode, the processor in master mode does not release bus privileges. In systems performing bus arbitration, make the transition to standby mode or deep-sleep mode only after setting the bus privilege release enable bit (BCR1.BREQEN) to 0 for the processor in master mode. If the bus privilege release enable bit remains set to 1, operation cannot be guaranteed when the transition is made to standby mode or deep-sleep mode.

Simultaneous Use of Refresh and Bus Arbitration: With the SH7751, when performing bus arbitration using the external device and $\overline{\text{BREQ}}$ signal, the following two failures may occur.

- When a $\overline{\text{BREQ}}$ signal is input from the external device while using DMA transfer or target transfer by the PCIC, and DRAM/synchronous DRAM is set to CAS-before-RAS refresh and auto-refresh in master mode (MD7 = 1), bus arbitration may not be performed correctly and this LSI may hang up.

- When a $\overline{\text{BREQ}}$ signal is input from the external device while DRAM/synchronous DRAM is set to CAS-before-RAS refresh and auto-refresh in master mode (MD7 = 1), assertion of the $\overline{\text{BACK}}$ signal (low-level) in response to the $\overline{\text{BREQ}}$ signal may be for only one cycle at CKIO.

Both above phenomena can be avoided by not using the $\overline{\text{BREQ}}$ signal. If the $\overline{\text{BREQ}}$ signal is to be used, disable refresh operations during normal operation. If refresh operations are required, carry them out at one time with the BREQEN bit in BCR1 cleared to 0.

Synchronous DRAM Mode Register Settings (SH7751 Only): The following conditions must be satisfied when setting the synchronous DRAM mode register.

- The DMAC must not be activated until synchronous DRAM mode register setting is completed.*¹
- Register setting for the on-chip peripheral modules*² must not be performed until synchronous DRAM mode register setting is completed.*³

Notes: 1. If a conflict occurs between synchronous DRAM mode register setting and memory access using the DMAC, neither operation can be guaranteed.

2. This applies to the following on-chip peripheral modules: CPG, RTC, INTC, TMU, SCI, SCIF, and H-UDI.

3. If synchronous DRAM mode register setting is performed immediately following write access to the on-chip peripheral modules*², the values written to the on-chip peripheral modules cannot be guaranteed. Note that following power-on, synchronous DRAM mode register settings should be performed before accessing synchronous DRAM. After making mode register settings, do not change them.

Section 14 Direct Memory Access Controller (DMAC)

14.1 Overview

The SH7751 includes an on-chip four-channel direct memory access controller (DMAC). The SH7751R has an on-chip eight-channel DMAC. The DMAC can be used in place of the CPU to perform high-speed data transfers among external devices equipped with DACK (DMA transfer end notification), external memories, memory-mapped external devices, and on-chip peripheral modules (TMU, RTC, SCI, SCIF, CPG, and INTC). Using the DMAC reduces the burden on the CPU and increases the operating efficiency of the chip. When using the SH7751R, see section 14.6, Configuration of the DMAC (SH7751R), section 14.7, Register Descriptions (SH7751R), and section 14.8, Operation (SH7751R).

14.1.1 Features

The DMAC has the following features.

- Four channels (SH7751), eight channels (SH7751R)
- Physical address space
- Choice of 8-bit, 16-bit, 32-bit, 64-bit, or 32-byte transfer data length
- Maximum of 16 M (16,777,216) transfers
- Choice of single or dual address mode
 - Single address mode: Either the transfer source or the transfer destination (external device) is accessed by a DACK signal while the other is accessed by address. One data transfer is completed in one bus cycle.
 - Dual address mode: Both the transfer source and transfer destination are accessed by address. Values set in DMAC internal registers indicate the accessed address for both the transfer source and the transfer destination. Two bus cycles are required for one data transfer.
- Choice of bus mode: cycle steal mode or burst mode
- Two types of DMAC channel priority ranking:
 - Fixed priority mode: Channel priorities are permanently fixed.
 - Round robin mode: Sets the lowest priority for the channel that last received an execution request.
- An interrupt request can be sent to the CPU on completion of the specified number of transfers.

- The following kinds of DMAC transfer activation requests are provided
 - External request
 - (1) Normal DMA mode

Two $\overline{\text{DREQ}}$ pins. Low level detection or falling edge detection can be specified. External requests can only be accepted on channel 0 and channel 1.
 - (2) On-demand data transfer mode (DDT mode)

The SH7551 performs interfacing between an external device and the DMAC using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID [1:0], and D[31:0] pins. The SH7551R performs interfacing between an external device and the DMAC using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID [2:0], and D[31:0] pins. External requests can be accepted on all eight channels.

Channel 0 does not have a request queue, but channels 1 to 3 in the SH7751 and channels 1 to 7 in the SH7751R each have four request queues.
 - On-chip peripheral modules request

Transfer requests from the SCI, SCF, and TMU. These can be accepted on all channels.
 - Auto-request

A transfer request is generated automatically within the DMAC.
- Channel functions: Transfer modes that can be set are different for each channel.
 - (1) Normal DMA mode
 - Channel 0: Single or dual address mode. External requests are accepted.
 - Channel 1: Single or dual address mode. External requests are accepted.
 - Channel 2: Dual address mode only
 - Channel 3: Dual address mode only
 - Channel 4 (SH7751R only): Dual address mode only
 - Channel 5 (SH7751R only): Dual address mode only
 - Channel 6 (SH7751R only): Dual address mode only
 - Channel 7 (SH7751R only): Dual address mode only
 - (2) DDT mode
 - Channel 0: Single or dual address mode. External requests are accepted.
 - Channel 1: Single or dual address mode. External requests are accepted.
 - Channel 2: Single or dual address mode. External requests are accepted.
 - Channel 3: Single or dual address mode. External requests are accepted.
 - Channel 4 (SH7751R only): Single or dual address mode. External requests are accepted.
 - Channel 5 (SH7751R only): Single or dual address mode. External requests are accepted.

- Channel 6 (SH7751R only): Single or dual address mode. External requests are accepted.
- Channel 7 (SH7751R only): Single or dual address mode. External requests are accepted.
- In DDT mode, data transfer is carried out by the SH7751 using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID [1:0], and D[31:0] signals to perform handshaking between the external device and the DMAC, and data transfer is carried out by the SH7751R using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID [2:0], and D[31:0] signals to perform handshaking between the external device and the DMAC.
- Request-queue clear for each channel (SH7751R only)
Request queues can be cleared on a channel-by-channel basis in either of the following two ways.
 - Clearing a request queue by DTR format
The request queues of the relevant channel are cleared when it receives DTR.SZ = 110, DTR.ID = 00, DTR.MD = 11, and DTR.COUNT [7:4]* = [1–8].
 - Using software to clear the request queue
The request queues of the relevant channel are cleared by writing a 1 to the CHCRn.QCL bit (request-queue clear bit) of each channel.

Note: * DTR.COUNT [7:4] (DTR [23:20]): Sets the port as not used. In DDT mode on the SH7751, an external device and the DMAC perform handshaking using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID[1:0], and D[31:0] signals during data transfer. On the SH7751R, the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID[2:0], and D[31:0] signals are used for handshaking during data transfer between an external device and the DMAC.

14.1.2 Block Diagram (SH7751)

Figure 14.1 shows a block diagram of the DMAC.

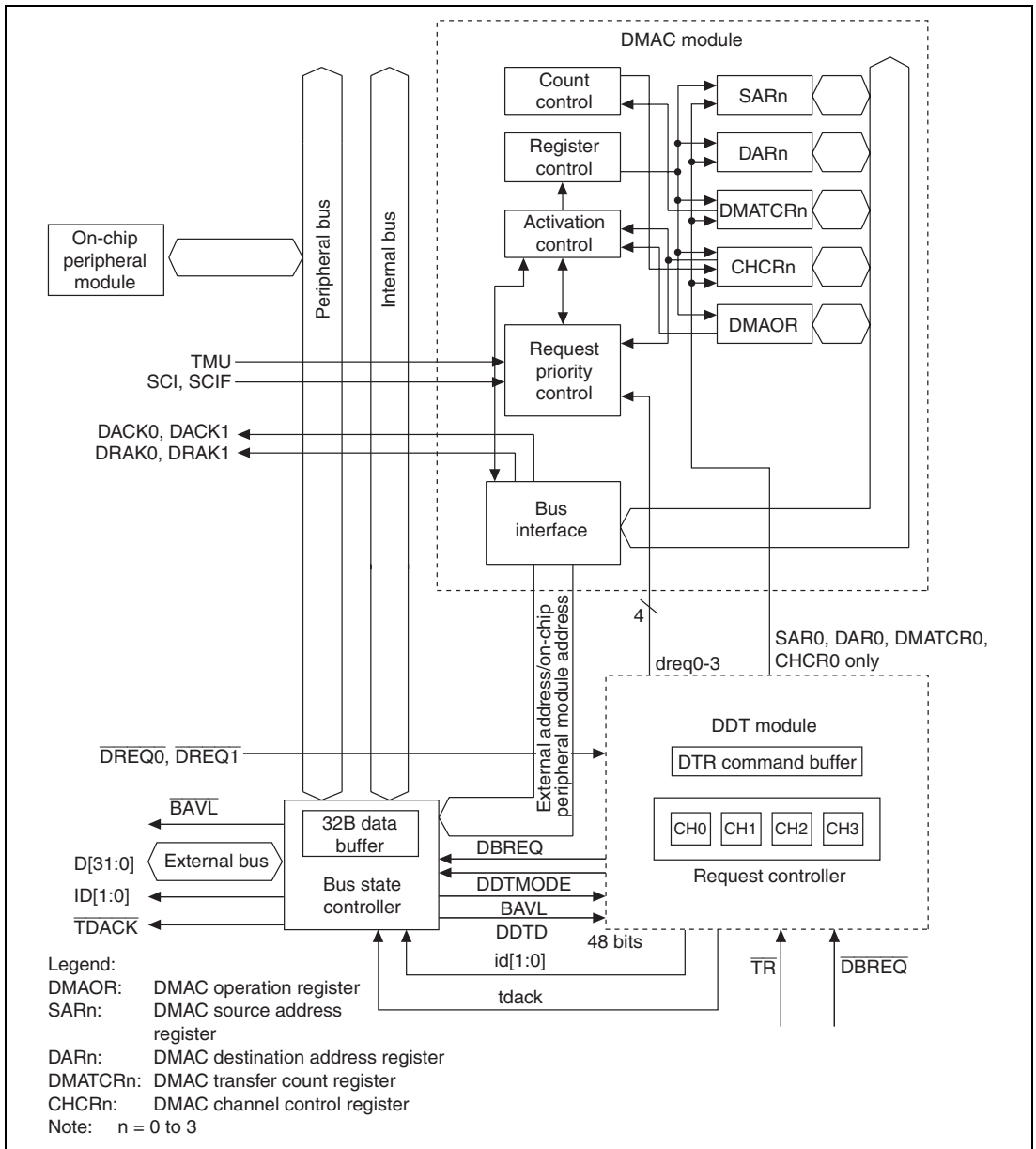


Figure 14.1 Block Diagram of DMAC

14.1.3 Pin Configuration (SH7751)

Tables 14.1 and 14.2 show the DMAC pins.

Table 14.1 DMAC Pins

| Channel | Pin Name | Abbreviation | I/O | Function |
|---------|--|---------------------------|--------|---|
| 0 | DMA transfer request | $\overline{\text{DREQ0}}$ | Input | DMA transfer request input from external device to channel 0 |
| | $\overline{\text{DREQ}}$ acceptance confirmation | DRAK0 | Output | Acceptance of request for DMA transfer from channel 0 to external device Notification to external device of start of execution |
| | DMA transfer end notification | DACK0 | Output | Strobe output to external device of DMA transfer request from channel 0 to external device |
| 1 | DMA transfer request | $\overline{\text{DREQ1}}$ | Input | DMA transfer request input from external device to channel 1 |
| | $\overline{\text{DREQ}}$ acceptance confirmation | DRAK1 | Output | Acceptance of request for DMA transfer from channel 1 to external device Notification to external device of start of execution |
| | DMA transfer end notification | DACK1 | Output | Strobe output to external device of DMA transfer request from channel 1 to external device |

Table 14.2 DMAC Pins in DDT Mode

| Pin Name | Abbreviation | I/O | Function |
|-----------------------------|--------------------------------------|--------|---|
| Data bus request | $\overline{\text{DBREQ}}$ (DREQ0) | Input | Data bus release request from external device for DTR format input |
| Data bus available | $\overline{\text{BAVL}}$ (DRAK0) | Output | Data bus release notification Data bus can be used 2 cycles after BAVL is asserted |
| Transfer request signal | $\overline{\text{TR}}$ (DREQ1) | Input | If asserted 2 cycles after $\overline{\text{BAVL}}$ assertion, DTR format is sent Only $\overline{\text{TR}}$ asserted: DMA request $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ asserted simultaneously: Direct request to channel 2 |
| DMAC strobe | $\overline{\text{TDACK}}$ (DACK0) | Output | Reply strobe signal for external device from DMAC |
| Channel number notification | ID [1:0] (DRAK1, DACK1) | Output | Notification of channel number to external device at same time as $\overline{\text{TDACK}}$ output (ID [1] = DRAK1, ID [0] = DACK1) |

14.1.4 Register Configuration (SH7751)

Table 14.3 summarizes the DMAC registers. The DMAC has a total of 17 registers: four registers are allocated to each channel, and an additional control register is shared by all four channels.

Table 14.3 DMAC Registers

| Channel | Name | Abbreviation | Read/Write | Initial Value | P4 Address | Area 7 Address | Access Size |
|-----------------|------------------------------------|--------------|------------|---------------|-------------|----------------|-------------|
| 0 | DMA source address register 0 | SAR0 | R/W | Undefined | H'FFFA00000 | H'1FA00000 | 32 |
| | DMA destination address register 0 | DAR0 | R/W | Undefined | H'FFFA00004 | H'1FA00004 | 32 |
| | DMA transfer count register 0 | DMATCR0 | R/W | Undefined | H'FFFA00008 | H'1FA00008 | 32 |
| | DMA channel control register 0 | CHCR0 | R/W* | H'00000000 | H'FFFA0000C | H'1FA0000C | 32 |
| 1 | DMA source address register 1 | SAR1 | R/W | Undefined | H'FFFA00010 | H'1FA00010 | 32 |
| | DMA destination address register 1 | DAR1 | R/W | Undefined | H'FFFA00014 | H'1FA00014 | 32 |
| | DMA transfer count register 1 | DMATCR1 | R/W | Undefined | H'FFFA00018 | H'1FA00018 | 32 |
| | DMA channel control register 1 | CHCR1 | R/W* | H'00000000 | H'FFFA0001C | H'1FA0001C | 32 |
| 2 | DMA source address register 2 | SAR2 | R/W | Undefined | H'FFFA00020 | H'1FA00020 | 32 |
| | DMA destination address register 2 | DAR2 | R/W | Undefined | H'FFFA00024 | H'1FA00024 | 32 |
| | DMA transfer count register 2 | DMATCR2 | R/W | Undefined | H'FFFA00028 | H'1FA00028 | 32 |
| | DMA channel control register 2 | CHCR2 | R/W* | H'00000000 | H'FFFA0002C | H'1FA0002C | 32 |
| 3 | DMA source address register 3 | SAR3 | R/W | Undefined | H'FFFA00030 | H'1FA00030 | 32 |
| | DMA destination address register 3 | DAR3 | R/W | Undefined | H'FFFA00034 | H'1FA00034 | 32 |
| | DMA transfer count register 3 | DMATCR3 | R/W | Undefined | H'FFFA00038 | H'1FA00038 | 32 |
| | DMA channel control register 3 | CHCR3 | R/W* | H'00000000 | H'FFFA0003C | H'1FA0003C | 32 |
| Common register | DMA operation register | DMAOR | R/W* | H'00000000 | H'FFFA00040 | H'1FA00040 | 32 |

Notes: Longword access should be used for all control registers. If a different access width is used, reads will return all 0s and writes will not be possible.

* Bit 1 of CHCR0–CHCR3 and bits 2 and 1 of DMAOR can only be written with 0 after being read as 1, to clear the flags.

14.2 Register Descriptions

14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3)

| | | | | | | | | |
|----------------|-----|-------|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | | | | | | | 0 |
| Initial value: | — | | | | | | | — |
| R/W: | R/W | | | | | | | R/W |

DMA source address registers 0–3 (SAR0–SAR3) are 32-bit readable/writable registers that specify the source address of a DMA transfer. These registers have a counter feedback function, and during a DMA transfer they indicate the next source address. In single address mode, the SAR value is ignored when a device with DACK has been specified as the transfer source.

Specify a 16-bit, 32-bit, 64-bit, or 32-byte boundary address when performing a 16-bit, 32-bit, 64-bit, or 32-byte data transfer, respectively. If a different address is specified, an address error will be detected and the DMAC will halt.

The initial value of these registers after a power-on or manual reset is undefined. They retain their values in standby mode, sleep mode, and deep sleep mode.

14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)

| | | | | | | | | |
|----------------|-----|-------|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | | | | | | | 0 |
| | | | | | | | | |
| Initial value: | — | | | | | | | — |
| R/W: | R/W | | | | | | | R/W |

DMA destination address registers 0–3 (DAR0–DAR3) are 32-bit readable/writable registers that specify the destination address of a DMA transfer. These registers have a counter feedback function, and during a DMA transfer they indicate the next destination address. In single address mode, the DAR value is ignored when a device with DACK has been specified as the transfer destination.

Specify a 16-bit, 32-bit, 64-bit, or 32-byte boundary address when performing a 16-bit, 32-bit, 64-bit, or 32-byte data transfer, respectively. If a different address is specified, an address error will be detected and the DMAC will halt.

The initial value of these registers after a power-on or manual reset is undefined. They retain their values in standby mode, sleep mode, and deep sleep mode.

- Notes:
1. When a 16-bit, 32-bit, 64-bit, or 32-byte boundary address is specified, take care with the setting of bit 0, bits 1–0, bits 2–0, or bits 4–0, respectively. If an address specification that ignores boundary considerations is made, the DMAC will detect an address error and halt operation on all channels (DMAOR: address error flag AE = 1). The DMAC will also detect an address error and halt if an area 7 address is specified in an external data bus transfer, or if the address of a nonexistent on-chip peripheral module is specified.
 2. External addresses are 29 bits in length. SAR[31:29] and DAR[31:29] are not used in DMA transfer, and it is recommended that they both be set to 000.

14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3)

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | | | | | | | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | | | | |
| 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 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMA transfer count registers 0–3 (DMATCR0–DMATCR3) are 32-bit readable/writable registers that specify the transfer count for the corresponding channel (byte count, word count, longword count, quadword count, or 32-byte count). Specifying H'000001 gives a transfer count of 1, while H'000000 gives the maximum setting, 16,777,216 (16M) transfers. During DMAC operation, the remaining number of transfers is shown.

Bits 31–24 of these registers are reserved; they are always read as 0, and should only be written with 0.

The initial value of these registers after a power-on or manual reset is undefined. They retain their values in standby mode, sleep mode, and deep sleep mode.

14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)

| | | | | | | | | |
|----------------|------|------|------|-----|------|-------|-------|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | SSA2 | SSA1 | SSA0 | STC | DSA2 | DSA1 | DSA0 | DTC |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | DS | RL | AM | AL |
| Initial value: | 0 | 0 | 0 | 0 | — | — | — | — |
| R/W: | R | R | R | R | R/W | (R/W) | R/W | (R/W) |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | 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 |
| | TM | TS2 | TS1 | TS0 | — | IE | TE | DE |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R | R/W | R/(W) | R/W |

Notes: The TE bit can only be written with 0 after being read as 1, to clear the flag.

The RL, AM, AL, and DS bits may be absent, depending on the channel.

DMA channel control registers 0–3 (CHCR0–CHCR3) are 32-bit readable/writable registers that specify the operating mode, transfer method, etc., for each channel. Bits 31–28 and 27–24 indicate the source address and destination address, respectively; these settings are only valid when the transfer involves the CS5 or CS6 space and the relevant space has been specified as a PCMCIA interface space. In other cases, these bits should be cleared to 0. For details of the PCMCIA interface, see section 13.3.7, PCMCIA Interface.

Bits 18 and 16 are not present in CHCR2 and CHCR3. In CHCR2 and CHCR3, these bits cannot be modified (for write, a write value of 0 should always be used) and are always read as 0.

These registers are initialized to H'00000000 by a power-on or manual reset. They retain their values in standby mode, sleep mode, and deep sleep mode.

Bits 31 to 29—Source Address Space Attribute Specification (SSA2–SSA0): These bits specify the space attribute for PCMCIA interface area access.

| Bit 31: SSA2 | Bit 30: SSA1 | Bit 29: SSA0 | Description |
|--------------|--------------|--------------|---|
| 0 | 0 | 0 | Reserved in PCMCIA access (Initial value) |
| | | 1 | Dynamic bus sizing I/O space |
| | 1 | 0 | 8-bit I/O space |
| | | 1 | 16-bit I/O space |
| 1 | 0 | 0 | 8-bit common memory space |
| | | 1 | 16-bit common memory space |
| | 1 | 0 | 8-bit attribute memory space |
| | | 1 | 16-bit attribute memory space |

Bit 28—Source Address Wait Control Select (STC): Specifies CS5 or CS6 space wait control for PCMCIA interface area access. This bit selects the wait control register in the BSC that performs area 5 and 6 wait cycle control.

| Bit 28: STC | Description |
|-------------|---|
| 0 | CS5 space wait cycle selection (Initial value) Settings of bits A5W2–A5W0 in wait control register 2 (WCR2), and bits A5PCW1–A5PCW0, A5TED2–A5TED0, and A5TEH2–A5TEH0 in the PCMCIA control register (PCR), are selected |
| 1 | CS6 space wait cycle selection Settings of bits A6W2–A6W0 in wait control register 2 (WCR2), and bits A6PCW1–A6PCW0, A6TED2–A6TED0, and A6TEH2–A6TEH0 in the PCMCIA control register (PCR), are selected |

Note: For details, see section 13.3.7, PCMCIA Interface.

Bits 27 to 25—Destination Address Space Attribute Specification (DSA2–DSA0): These bits specify the space attribute for PCMCIA interface area access.

| Bit 27: DSA2 | Bit 26: DSA1 | Bit 25: DSA0 | Description |
|--------------|--------------|--------------|---|
| 0 | 0 | 0 | Reserved in PCMCIA access (Initial value) |
| | | 1 | Dynamic bus sizing I/O space |
| | 1 | 0 | 8-bit I/O space |
| | | 1 | 16-bit I/O space |
| 1 | 0 | 0 | 8-bit common memory space |
| | | 1 | 16-bit common memory space |
| | 1 | 0 | 8-bit attribute memory space |
| | | 1 | 16-bit attribute memory space |

Bit 24—Destination Address Wait Control Select (DTC): Specifies CS5 or CS6 space wait cycle control for PCMCIA interface area access. This bit selects the wait control register in the BSC that performs area 5 and 6 wait cycle control.

| Bit 24: DTC | Description |
|-------------|---|
| 0 | CS5 space wait cycle selection (Initial value) Settings of bits A5W2–A5W0 in wait control register 2 (WCR2), and bits A5PCW1–A5PCW0, A5TED2–A5TED0, and A5TEH2–A5TEH0 in the PCMCIA control register (PCR), are selected |
| 1 | CS6 space wait cycle selection Settings of bits A6W2–A6W0 in wait control register 2 (WCR2), and bits A6PCW1–A6PCW0, A6TED2–A6TED0, and A6TEH2–A6TEH0 in the PCMCIA control register (PCR), are selected |

Note: For details, see section 13.3.7, PCMCIA Interface.

Bits 23 to 20—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 19— $\overline{\text{DREQ}}$ Select (DS): Specifies either low level detection or falling edge detection as the sampling method for the $\overline{\text{DREQ}}$ pin used in external request mode.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. In DDT mode, it is valid in CHCR0–CHCR3.

| Bit 19: DS | Description |
|------------|-------------------------------------|
| 0 | Low level detection (Initial value) |
| 1 | Falling edge detection |

Notes: Level detection burst mode when TM = 1 and DS = 0

Edge detection burst mode when TM = 1 and DS = 1

Bit 18—Request Check Level (RL): Selects whether the DRAK signal (that notifies an external device of the acceptance of $\overline{\text{DREQ}}$) is an active-high or active-low output.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. It is invalid in DDT mode.

| Bit 18: RL | Description |
|------------|---|
| 0 | DRAK is an active-high output (Initial value) |
| 1 | DRAK is an active-low output |

Bit 17—Acknowledge Mode (AM): In dual address mode, selects whether DACK is output in the data read cycle or write cycle. In single address mode, DACK is always output regardless of the setting of this bit.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. In DDT mode, it is valid in CHCR1–CHCR3. (DDT mode: TDACK)

| Bit 17: AM | Description |
|------------|--|
| 0 | DACK is output in read cycle (Initial value) |
| 1 | DACK is output in write cycle |

Bit 16—Acknowledge Level (AL): Specifies the DACK (acknowledge) signal as active-high or active-low.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. It is invalid in DDT mode.

| Bit 16: AL | Description |
|------------|------------------------------------|
| 0 | Active-high output (Initial value) |
| 1 | Active-low output |

Bits 15 and 14—Destination Address Mode 1 and 0 (DM1, DM0): These bits specify incrementing/decrementing of the DMA transfer destination address. The specification of these bits is ignored when data is transferred from external memory to an external device in single address mode.

| Bit 15: DM1 | Bit 14: DM0 | Description |
|-------------|-------------|--|
| 0 | 0 | Destination address fixed (Initial value) |
| | 1 | Destination address incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +8 in 64-bit transfer, +32 in 32-byte burst transfer) |
| 1 | 0 | Destination address decremented (−1 in 8-bit transfer, −2 in 16-bit transfer, −4 in 32-bit transfer, −8 in 64-bit transfer, −32 in 32-byte burst transfer) |
| | 1 | Setting prohibited |

Bits 13 and 12—Source Address Mode 1 and 0 (SM1, SM0): These bits specify incrementing/decrementing of the DMA transfer source address. The specification of these bits is ignored when data is transferred from an external device to external memory in single address mode.

| Bit 13: SM1 | Bit 12: SM0 | Description |
|-------------|-------------|---|
| 0 | 0 | Source address fixed (Initial value) |
| | 1 | Source address incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +8 in 64-bit transfer, +32 in 32-byte burst transfer) |
| 1 | 0 | Source address decremented (−1 in 8-bit transfer, −2 in 16-bit transfer, −4 in 32-bit transfer, −8 in 64-bit transfer, −32 in 32-byte burst transfer) |
| | 1 | Setting prohibited |

Bits 11 to 8—Resource Select 3 to 0 (RS3–RS0): These bits specify the transfer request source.

| Bit 11: RS3 | Bit 10: RS2 | Bit 9: RS1 | Bit 8: RS0 | Description | |
|----------------|----------------|---------------|---------------|---|--|
| 0 | 0 | 0 | 0 | External request, dual address mode*1*3 (external address space → external address space) (Initial value) | |
| | | | 1 | Setting prohibited | |
| | | 1 | 0 | External request, single address mode External address space → external device*1*3 | |
| | | | 1 | External request, single address mode External device → external address space*1*3 | |
| | 1 | 0 | 0 | Auto-request (external address space → external address space)*2 | |
| | | | 1 | Auto-request (external address space → on-chip peripheral module)*2 | |
| | | 1 | 0 | Auto-request (on-chip peripheral module → external address space)*2 | |
| | | | 1 | Setting prohibited | |
| 1 | 0 | 0 | 0 | SCI transmit-data-empty interrupt transfer request (external address space → SCTDR1)*2 | |
| | | | 1 | SCI receive-data-full interrupt transfer request (SCRDR1 → external address space)*2 | |
| | | 1 | 0 | SCIF transmit-data-empty interrupt transfer request (external address space → SCFTDR2)*2 | |
| | | | 1 | SCIF receive-data-full interrupt transfer request (SCFRDR2 → external address space)*2 | |
| | | 1 | 0 | 0 | TMU channel 2 (input capture interrupt, external address space → external address space)*2 |
| | | | | 1 | TMU channel 2 (input capture interrupt) (external address space → on-chip peripheral module)*2 |
| | 1 | 0 | 0 | TMU channel 2 (input capture interrupt) (on-chip peripheral module → external address space)*2 | |
| | | | 1 | Setting prohibited | |

Notes: 1. External request specifications are valid only for channels 0 and 1. Requests are not accepted for channels 2 and 3 in normal DMA mode.

2. Dual address mode

3. In DDT mode, an external request specification is possible for channels 0, 1, 2, and 3.

Bit 7—Transmit Mode (TM): Specifies the bus mode for transfer.

| Bit 7: TM | Description |
|-----------|----------------------------------|
| 0 | Cycle steal mode (Initial value) |
| 1 | Burst mode |

Bits 6 to 4—Transmit Size 2 to 0 (TS2–TS0): These bits specify the transfer data size. In access to external memory, the specification is treated as an access size as described in section 13.3, Operation. In access to a register, the specification is treated as a register access size.

| Bit 6: TS2 | Bit 5: TS1 | Bit 4: TS0 | Description |
|------------|------------|------------|--|
| 0 | 0 | 0 | Quadword size (64-bit) specification (Initial value) |
| | | 1 | Byte size (8-bit) specification |
| | 1 | 0 | Word size (16-bit) specification |
| | | 1 | Longword size (32-bit) specification |
| 1 | 0 | 0 | 32-byte block transfer specification |

Bit 3—Reserved: This bit is always read as 0, and should only be written with 0.

Bit 2—Interrupt Enable (IE): When this bit is set to 1, an interrupt request (DMTE) is generated after the number of data transfers specified in DMATCR (when TE = 1).

| Bit 2: IE | Description |
|-----------|---|
| 0 | Interrupt request not generated after number of transfers specified in DMATCR (Initial value) |
| 1 | Interrupt request generated after number of transfers specified in DMATCR |

Bit 1—Transfer End (TE): This bit is set to 1 after the number of transfers specified in DMATCR. If the IE bit is set to 1 at this time, an interrupt request (DMTE) is generated.

If data transfer ends before TE is set to 1 (for example, due to an NMI interrupt, address error, or clearing of the DE bit or the DME bit in DMAOR), the TE bit is not set to 1. When this bit is 1, the transfer enabled state is not entered even if the DE bit is set to 1.

| Bit 1: TE | Description |
|-----------|---|
| 0 | Number of transfers specified in DMATCR not completed (Initial value) [Clearing conditions] <ul style="list-style-type: none"> When 0 is written to TE after reading TE = 1 In a power-on or manual reset, and in standby mode |
| 1 | Number of transfers specified in DMATCR completed |

Bit 0—DMAC Enable (DE): Enables operation of the corresponding channel.

| Bit 0: DE | Description |
|-----------|--|
| 0 | Operation of corresponding channel is disabled (Initial value) |
| 1 | Operation of corresponding channel is enabled |

When auto-request is specified (with RS3–RS0), transfer is begun when this bit is set to 1. In the case of an external request or on-chip peripheral module request, transfer is begun when a transfer request is issued after this bit is set to 1. Transfer can be suspended midway by clearing this bit to 0.

Even if the DE bit has been set, transfer is not enabled when TE is 1, when DME in DMAOR is 0, or when the NMIF or AE bit in DMAOR is 1.

14.2.5 DMA Operation Register (DMAOR)

| | | | | | | | | |
|----------------|-----|----|----|----|----|-------|-------|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | DDT | — | — | — | — | — | PR1 | PR0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R | R | R | R | R | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | 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/W |

Note: The AE and NMIF bits can only be written with 0 after being read as 1, to clear the flags.

DMAOR is a 32-bit readable/writable register that specifies the DMAC transfer mode.

DMAOR is initialized to H'00000000 by a power-on or manual reset. They retain their values in standby mode and deep sleep mode.

Bits 31 to 16—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 15—On-Demand Data Transfer (DDT): Specifies on-demand data transfer mode.

| Bit 15: DDT | Description |
|-------------|---------------------------------|
| 0 | Normal DMA mode (Initial value) |
| 1 | On-demand data transfer mode |

Note: $\overline{\text{BAVL}}$ (DRAK0) is an active-high output in normal DMA mode. When the DDT bit is set to 1, the $\overline{\text{BAVL}}$ pin function is enabled and this pin becomes an active-low output.

Bits 14 to 10—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 9 and 8—Priority Mode 1 and 0 (PR1, PR0): These bits determine the order of priority for channel execution when transfer requests are made for a number of channels simultaneously.

| Bit 9: PR1 | Bit 8: PR0 | Description |
|------------|------------|---------------------------------------|
| 0 | 0 | CH0 > CH1 > CH2 > CH3 (Initial value) |
| | 1 | CH0 > CH2 > CH3 > CH1 |
| 1 | 0 | CH2 > CH0 > CH1 > CH3 |
| | 1 | Round robin mode |

Bits 7 to 3—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 2—Address Error Flag (AE): Indicates that an address error has occurred during DMA transfer. If this bit is set during data transfer, transfers on all channels are suspended, and an interrupt request (DMAE) is generated. The CPU cannot write 1 to AE. This bit can only be cleared by writing 0 after reading 1.

| Bit 2: AE | Description |
|-----------|--|
| 0 | No address error, DMA transfer enabled (Initial value) [Clearing condition] When 0 is written to AE after reading AE = 1 |
| 1 | Address error, DMA transfer disabled [Setting condition] When an address error is caused by the DMAC |

Bit 1—NMI Flag (NMIF): Indicates that NMI has been input. This bit is set regardless of whether or not the DMAC is operating. If this bit is set during data transfer, transfers on all channels are suspended. The CPU cannot write 1 to NMIF. This bit can only be cleared by writing 0 after reading 1.

| Bit 1: NMIF | Description |
|-------------|--|
| 0 | No NMI input, DMA transfer enabled (Initial value) [Clearing condition] When 0 is written to NMIF after reading NMIF = 1 |
| 1 | NMI input, DMA transfer disabled [Setting condition] When an NMI interrupt is generated |

Bit 0—DMAC Master Enable (DME): Enables activation of the entire DMAC. When the DME bit and the DE bit of the CHCR register for the corresponding channel are set to 1, that channel is enabled for transfer. If this bit is cleared during data transfer, transfers on all channels are suspended.

Even if the DME bit has been set, transfer is not enabled when TE is 1 or DE is 0 in CHCR, or when the NMI or AE bit in DMAOR is 1.

| Bit 0: DME | Description |
|------------|--|
| 0 | Operation disabled on all channels (Initial value) |
| 1 | Operation enabled on all channels |

14.3 Operation

When a DMA transfer request is issued, the DMAC starts the transfer according to the predetermined channel priority order. It ends the transfer when the transfer end conditions are satisfied. Transfers can be requested in three modes: auto-request, external request, and on-chip peripheral module request. There are two modes for DMA transfer: single address mode and dual address mode. Either burst mode or cycle steal mode can be selected as the bus mode.

14.3.1 DMA Transfer Procedure

After the desired transfer conditions have been set in the DMA source address register (SAR), DMA destination address register (DAR), DMA transfer count register (DMATCR), DMA channel control register (CHCR), and DMA operation register (DMAOR), the DMAC transfers data according to the following procedure:

1. The DMAC checks to see if transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0).
2. When a transfer request is issued and transfer has been enabled, the DMAC transfers one transfer unit of data (determined by the setting of TS2–TS0). In auto-request mode, the transfer begins automatically when the DE bit and DME bit are set to 1. The DMATCR value is decremented by 1 for each transfer. The actual transfer flow depends on the address mode and bus mode.
3. When the specified number of transfers have been completed (when the DMATCR value reaches 0), the transfer ends normally. If the IE bit in CHCR is set to 1 at this time, a DMTE interrupt request is sent to the CPU.
4. If a DMAC address error or NMI interrupt occurs, the transfer is suspended. Transfer is also suspended when the DE bit in CHCR or the DME bit in DMAOR is cleared to 0. In the event of an address error, a DMAE interrupt request is forcibly sent to the CPU.

Figure 14.2 shows a flowchart of this procedure.

Note: If a transfer request is issued while transfer is disabled, the transfer enable wait state (transfer suspended state) is entered. Transfer is started when subsequently enabled (by setting DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0).

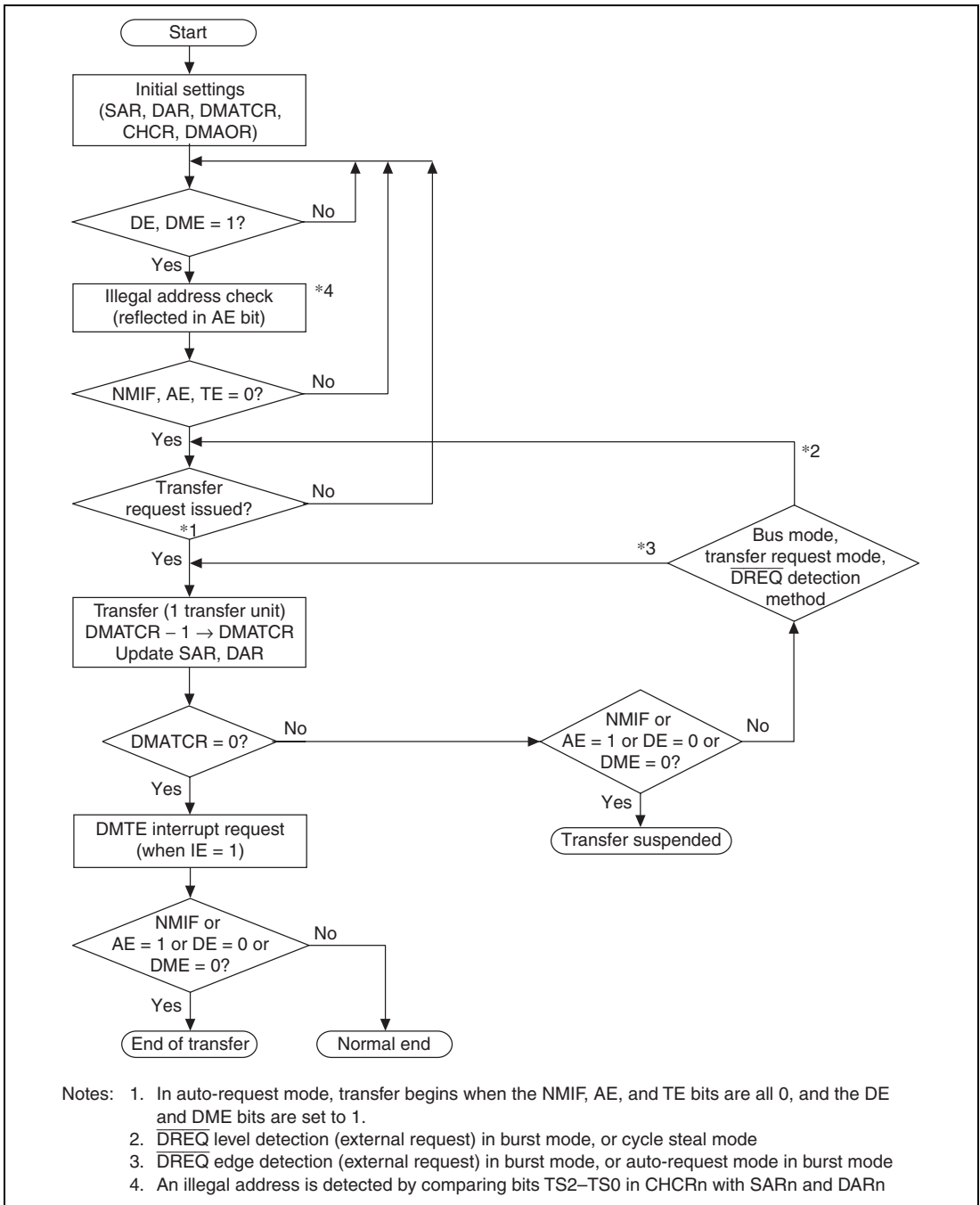


Figure 14.2 DMAC Transfer Flowchart

14.3.2 DMA Transfer Requests

DMA transfer requests are basically generated at either the data transfer source or destination, but they can also be issued by external devices or 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 peripheral module request. The transfer request mode is selected by means of bits RS3–RS0 in 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 bit in CHCR0–CHCR3 and the DME bit in the DMA operation register (DMAOR) are set to 1, the transfer begins (so long as the TE bit in CHCR0–CHCR3 and the NMIF and AE bits in DMAOR are all 0).

External Request Mode: In this mode a transfer is performed in response to a transfer request signal ($\overline{\text{DREQ}}$) from an external device. One of the modes shown in table 14.4 should be chosen according to the application system. If DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), transfer starts when $\overline{\text{DREQ}}$ is input. The DS bit in CHCR0/CHCR1 is used to select either falling edge detection or low level detection for the $\overline{\text{DREQ}}$ signal (level detection when DS = 0, edge detection when DS = 1).

The source of the transfer request does not have to be the data transfer source or destination.

$\overline{\text{DREQ}}$ is accepted after a power-on reset if TE = 0, NMIF = 0, and AE = 0, but transfer is not executed if DMA transfer is not enabled (DE = 0 or DME = 0).

In this case, DMA transfer is started when enabled (by setting DE = 1 and DME = 1).

Table 14.4 Selecting External Request Mode with RS Bits

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

- External Request Acceptance Conditions

- When at least one of DMAOR.DME and CHCR.DE is 0, and DMAOR.NMIF, DMAOR.AE, and CHCR.TE are all 0, if an external request ($\overline{\text{DREQ}}$: edge-detected) is input it will be held inside the DMAC until DMA transfer is either executed or canceled. Since DMA transfer is not enabled in this case (DME = 0 or DE = 0), DMA transfer is not initiated. DMA transfer is started after it is enabled (DME = 1, DE = 1, DMAOR.NMIF = 0, DMAOR.AE = 0, CHCR.TE = 0).
- When DMA transfer is enabled (DME = 1, DE = 1, DMAOR.NMIF = 0, DMAOR.AE = 0, CHCR.TE = 0), if an external request ($\overline{\text{DREQ}}$) is input, DMA transfer is started.
- An external request ($\overline{\text{DREQ}}$) will be ignored if input when CHCR.TE = 1, DMAOR.NMIF = 1, DMAOR.AE = 1, during a power-on reset or manual reset, in deep sleep mode, standby mode, or while the DMAC is in the module standby state.
- A previously input external request will be canceled by the occurrence of an NMI interrupt (DMAOR.NMIF = 1) or address error (DMAOR.AE = 1), or by a power-on reset or manual reset.

- Usage Notes

- An external request ($\overline{\text{DREQ}}$) is detected by a low level or falling edge. Ensure that the external request ($\overline{\text{DREQ}}$) signal is held high when there is no DMA transfer request from an external device after a power-on reset or manual reset.
When DMA transfer is restarted, check whether a DMA transfer request is being held.
- With $\overline{\text{DREQ}}$ edge detection, an accepted external request can be canceled by first negating $\overline{\text{DREQ}}$, enabling a change of setting from CHCR.DS = 1 to CHCR.DS = 0, and then asserting $\overline{\text{DREQ}}$ after setting CHCR.DS to 1 again.

On-Chip Peripheral Module Request Mode: In this mode a transfer is performed in response to a transfer request signal (interrupt request signal) from an on-chip peripheral module. As shown in table 14.5, there are seven transfer request signals: input capture interrupts from the timer unit (TMU), and receive-data-full interrupts (RXI) and transmit-data-empty interrupts (TXI) from the two serial communication interfaces (SCI, SCIF). If DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), transfer starts when a transfer request signal is input.

The source of the transfer request does not have to be the data transfer source or destination. However, when the transfer request is set to RXI (transfer request by SCI/SCIF receive-data-full interrupt), the transfer source must be the SCI/SCIF's receive data register (SCRDR1/SCFRDR2). When the transfer request is set to TXI (transfer request by SCI/SCIF transmit-data-empty interrupt), the transfer destination must be the SCI/SCIF's transmit data register (SCTDR1/SCFTDR2).

Table 14.5 Selecting On-Chip Peripheral Module Request Mode with RS Bits

| RS3 | RS2 | RS1 | RS0 | DMAC Transfer Request Source | DMAC Transfer Request Signal | Transfer Source | Transfer Destination | Bus Mode |
|-----|-----|-----|-----|------------------------------|---|--------------------|----------------------|------------------------|
| 1 | 0 | 0 | 0 | SCI transmitter | SCTDR1 (SCI transmit-data-empty transfer request) | External* | SCTDR1 | Cycle steal mode |
| | | | 1 | SCI receiver | SCRDR1 (SCI receive-data-full transfer request) | SCRDR1 | External* | Cycle steal mode |
| | | 1 | 0 | SCIF transmitter | SCFTDR2 (SCIF transmit-data-empty transfer request) | External* | SCFTDR2 | Cycle steal mode |
| | | | 1 | SCIF receiver | SCFRDR2 (SCIF receive-data-full transfer request) | SCFRDR2 | External* | Cycle steal mode |
| | 1 | 0 | 0 | TMU channel 2 | Input capture occurrence | External* | External* | Burst/cycle steal mode |
| | | | 1 | TMU channel 2 | Input capture occurrence | External* | On-chip peripheral | Burst/cycle steal mode |
| | | | 1 | TMU channel 2 | Input capture occurrence | On-chip peripheral | External* | Burst/cycle steal mode |

Legend:

TMU: Timer unit

SCI: Serial communication interface

SCIF: Serial communication interface with FIFO

Notes: SCI/SCIF burst transfer setting is prohibited.

If input capture interrupt acceptance is set for multiple channels and DE =1 for each channel, processing will be executed on the highest-priority channel in response to a single input capture interrupt.

A DMA transfer request by means of an input capture interrupt can be canceled by setting TCR2.ICPE1 = 0 and TCR2.ICPE0 = 0 in the TMU.

* External memory or memory-mapped external device

To output a transfer request from an on-chip peripheral module, set the DMA transfer request enable bit for that module and output a transfer request signal.

For details, see sections 12, Timer Unit (TMU), 15, Serial Communication Interface (SCI), and 16, Serial Communication Interface with FIFO (SCIF).

When a DMA transfer corresponding to a transfer request signal from an on-chip peripheral module shown in table 14.5 is carried out, the signal is discontinued automatically. This occurs every transfer in cycle steal mode, and in the last transfer in burst mode.

14.3.3 Channel Priorities

If the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority system, either in a fixed mode or round robin mode. The mode is selected with priority bits PR1 and PR0 in the DMA operation register (DMAOR).

Fixed Mode: In this mode, the relative channel priorities remain fixed. The following priority orders are available in fixed mode:

- CH0 > CH1 > CH2 > CH3
- CH0 > CH2 > CH3 > CH1
- CH2 > CH0 > CH1 > CH3

The priority order is selected with bits PR1 and PR0 in DMAOR.

Round Robin Mode: In round robin mode, each time the transfer of one transfer unit (byte, word, longword, quadword, or 32 bytes) ends on a given channel, that channel is assigned the lowest priority level. This is illustrated in figure 14.3. The order of priority in round robin mode immediately after a reset is CH0 > CH1 > CH2 > CH3.

Note: In round robin mode, if no transfer request is accepted for any channel during DMA transfer, the priority order becomes CH0 > CH1 > CH2 > CH3.

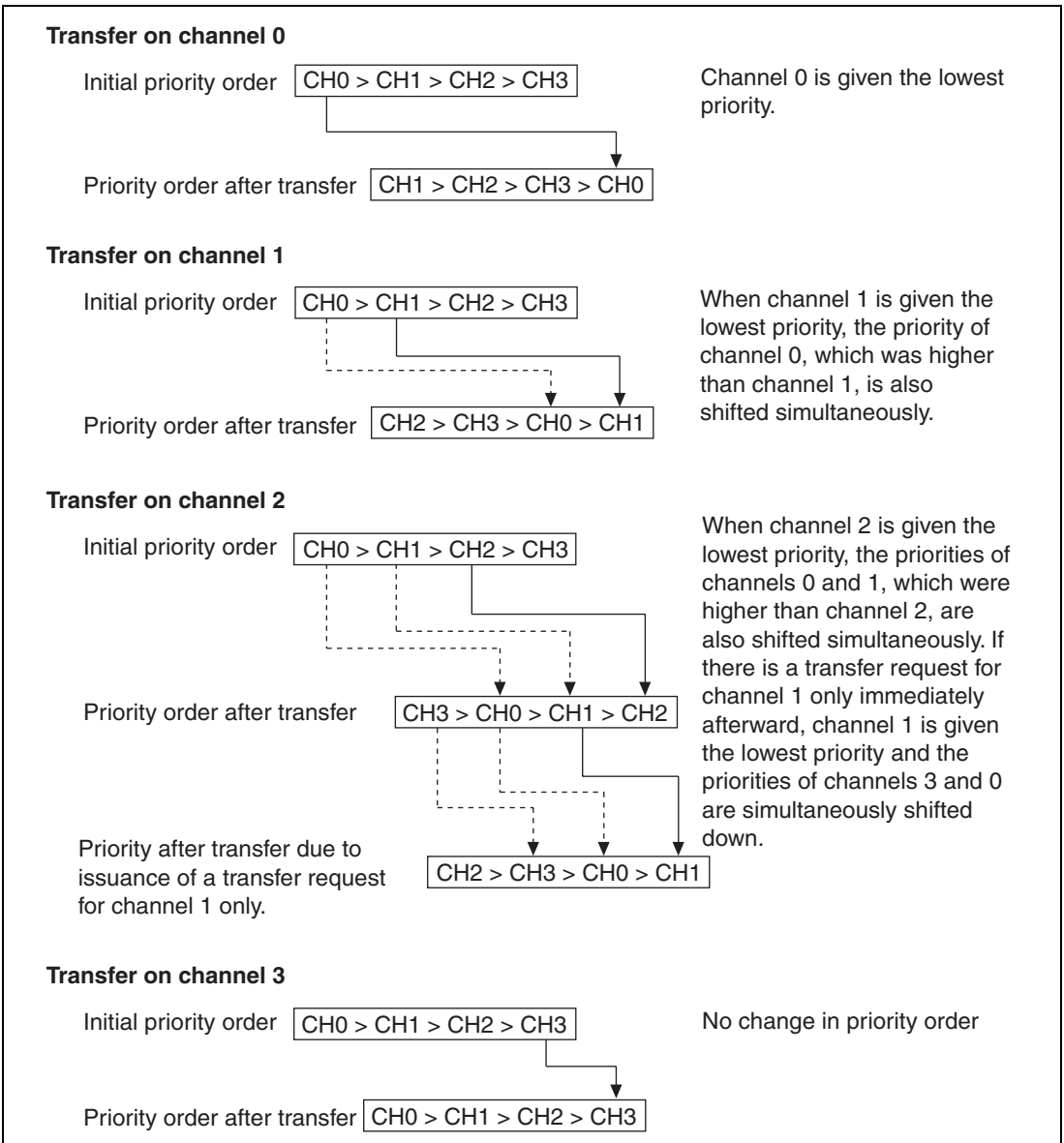


Figure 14.3 Round Robin Mode

Figure 14.4 shows the changes in priority levels when transfer requests are issued simultaneously for channels 0 and 3, and channel 1 receives a transfer request during a transfer on channel 0. The operation of the DMAC in this case is as follows.

1. Transfer requests are issued simultaneously for channels 0 and 3.
2. Since channel 0 has a higher priority level than channel 3, the channel 0 transfer is executed first (channel 3 is on transfer standby).
3. A transfer request is issued for channel 1 during the channel 0 transfer (channels 1 and 3 are on transfer standby).
4. At the end of the channel 0 transfer, channel 0 shifts to the lowest priority level.
5. At this point, channel 1 has a higher priority level than channel 3, so the channel 1 transfer is started (channel 3 is on transfer standby).
6. At the end of the channel 1 transfer, channel 1 shifts to the lowest priority level.
7. The channel 3 transfer is started.
8. At the end of the channel 3 transfer, the channel 3 and channel 2 priority levels are lowered, giving channel 3 the lowest priority.

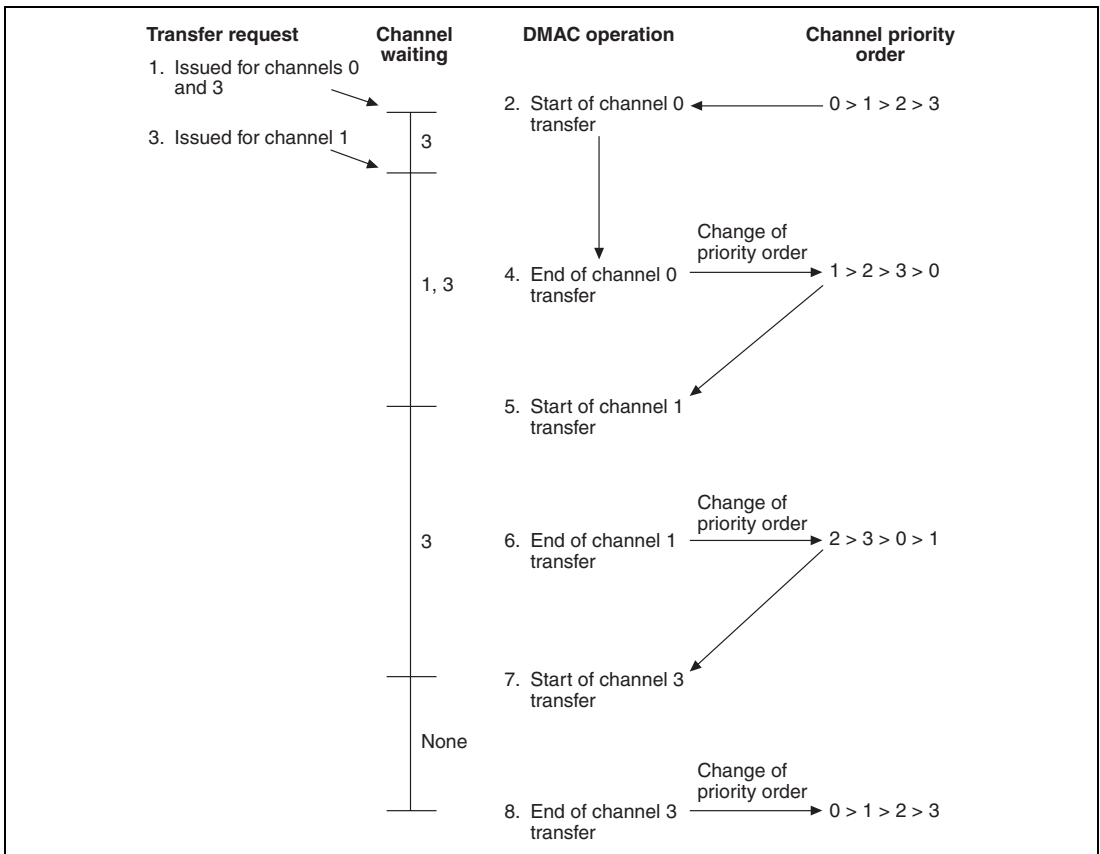


Figure 14.4 Example of Changes in Priority Order in Round Robin Mode

14.3.4 Types of DMA Transfer

The DMAC supports the transfers shown in table 14.6. It can operate in single address mode, in which either the transfer source or the transfer destination is accessed using the acknowledge signal, or in dual address mode, in which both the transfer source and transfer destination addresses are output. The actual transfer operation timing depends on the bus mode, which can be either burst mode or cycle steal mode.

Table 14.6 Supported DMA Transfers

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

Address Modes

Single Address Mode: In single address mode, both the transfer source and the transfer destination are external; one is accessed by the DACK signal and the other by an address. In this mode, the DMAC performs a DMA transfer in one bus cycle by simultaneously outputting the external device strobe signal (DACK) to either the transfer source or transfer destination external device to access it, while outputting an address to the other side of the transfer. Figure 14.5 shows an example of a transfer between external memory and an external device with DACK in which the external device outputs data to the data bus and that data is written to external memory in the same bus cycle.

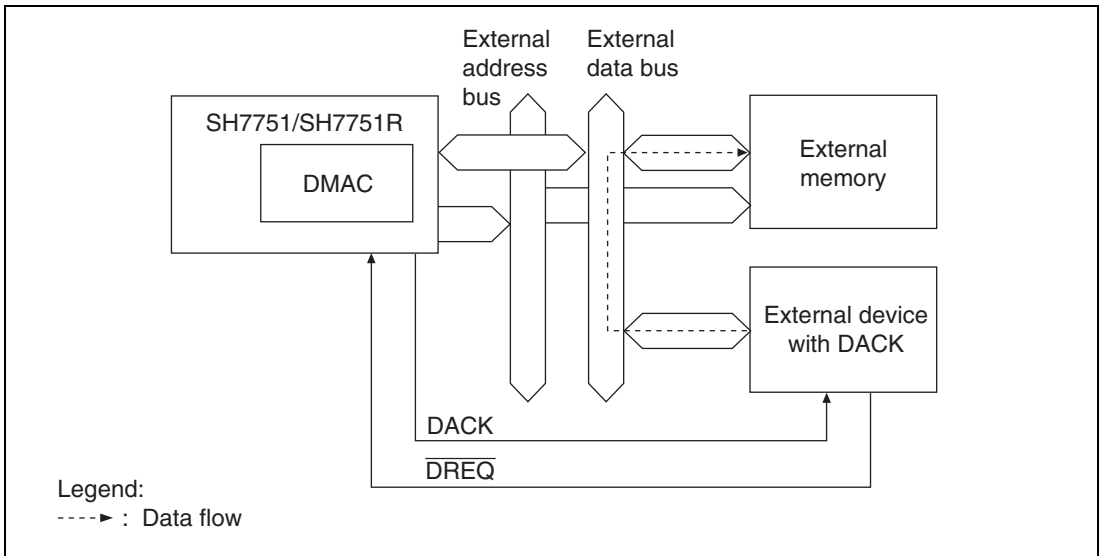
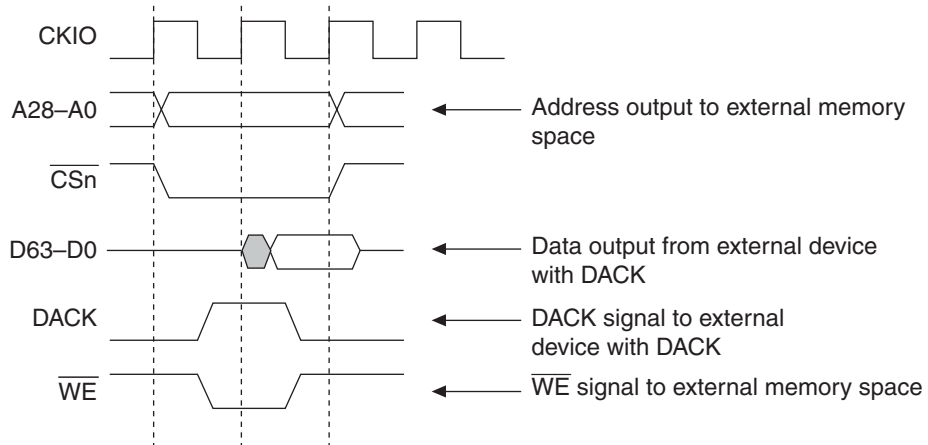


Figure 14.5 Data Flow in Single Address Mode

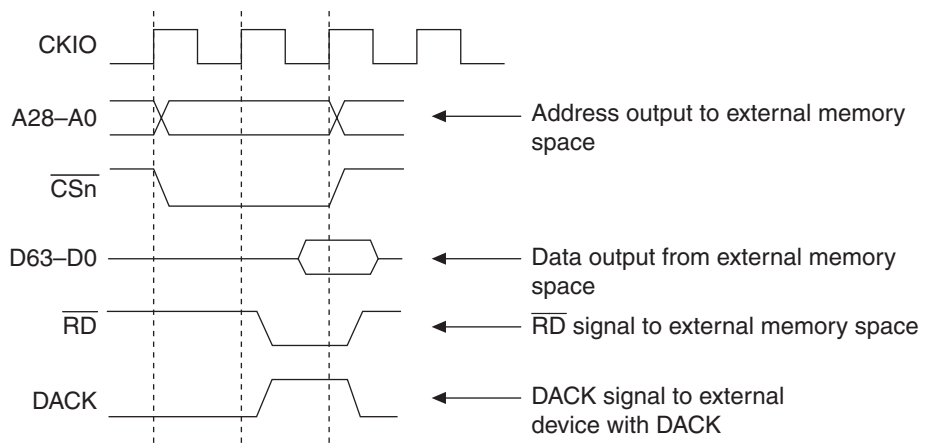
Two types of transfer are possible in single address mode: (1) transfer between an external device with DACK and a memory-mapped external device, and (2) transfer between an external device with DACK and external memory. Only the external request signal ($\overline{\text{DREQ}}$) is used in both these cases.

Figure 14.6 shows the transfer timing for single address mode.

The access timing depends on the type of external memory. For details, see the descriptions of the memory interfaces in section 13, Bus State Controller (BSC).



(a) From external device with DACK to external memory space



(b) From external memory space to external device with DACK

Figure 14.6 DMA Transfer Timing in Single Address Mode

Dual Address Mode: Dual address mode is used to access both the transfer source and the transfer destination by address. The transfer source and destination can be accessed by either on-chip peripheral module or external address.

In dual address mode, data is read from the transfer source in the data read cycle, and written to the transfer destination in the data write cycle, so that the transfer is executed in two bus cycles. The transfer data is temporarily stored in the data buffer in the bus state controller (BSC).

In a transfer between external memories such as that shown in figure 14.7, data is read from external memory into the BSC's data buffer in the read cycle, then written to the other external memory in the write cycle. Figure 14.8 shows the timing for this operation.

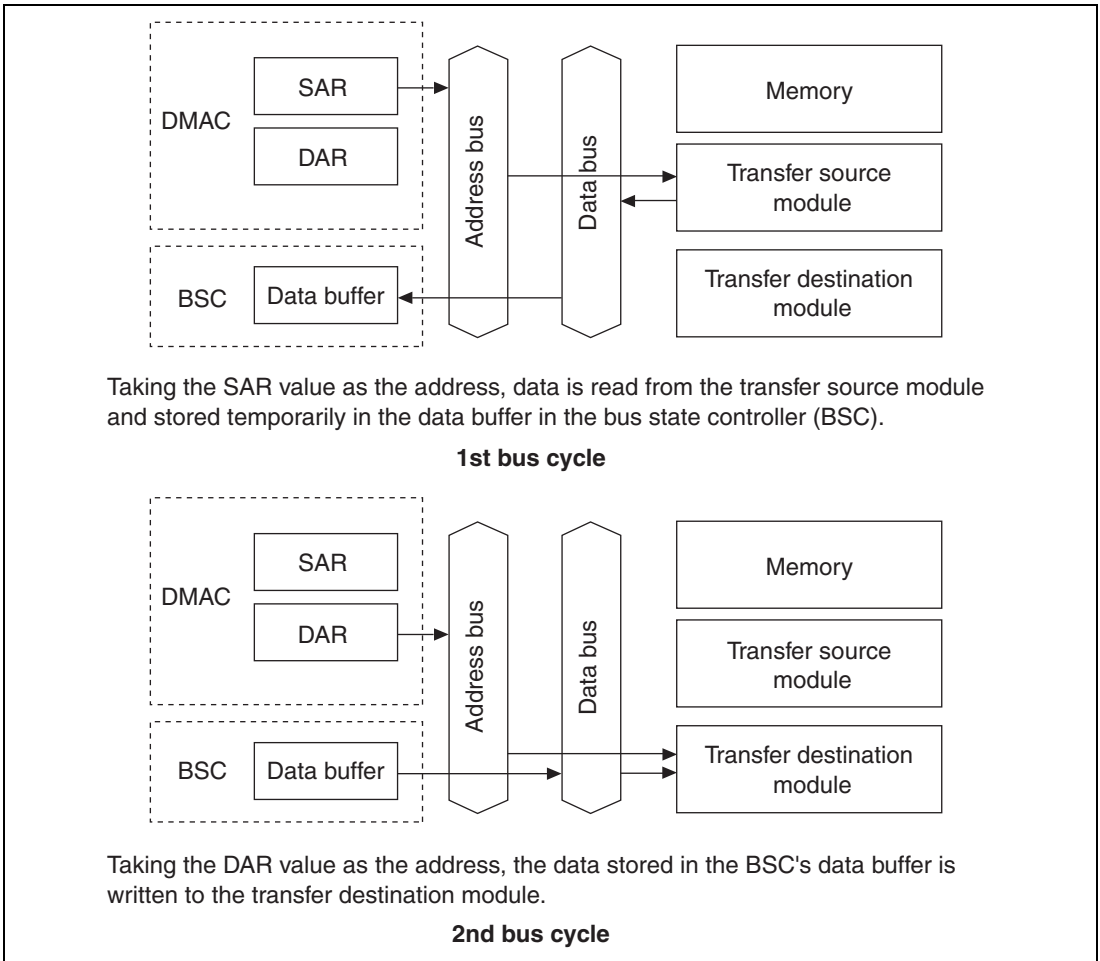


Figure 14.7 Operation in Dual Address Mode

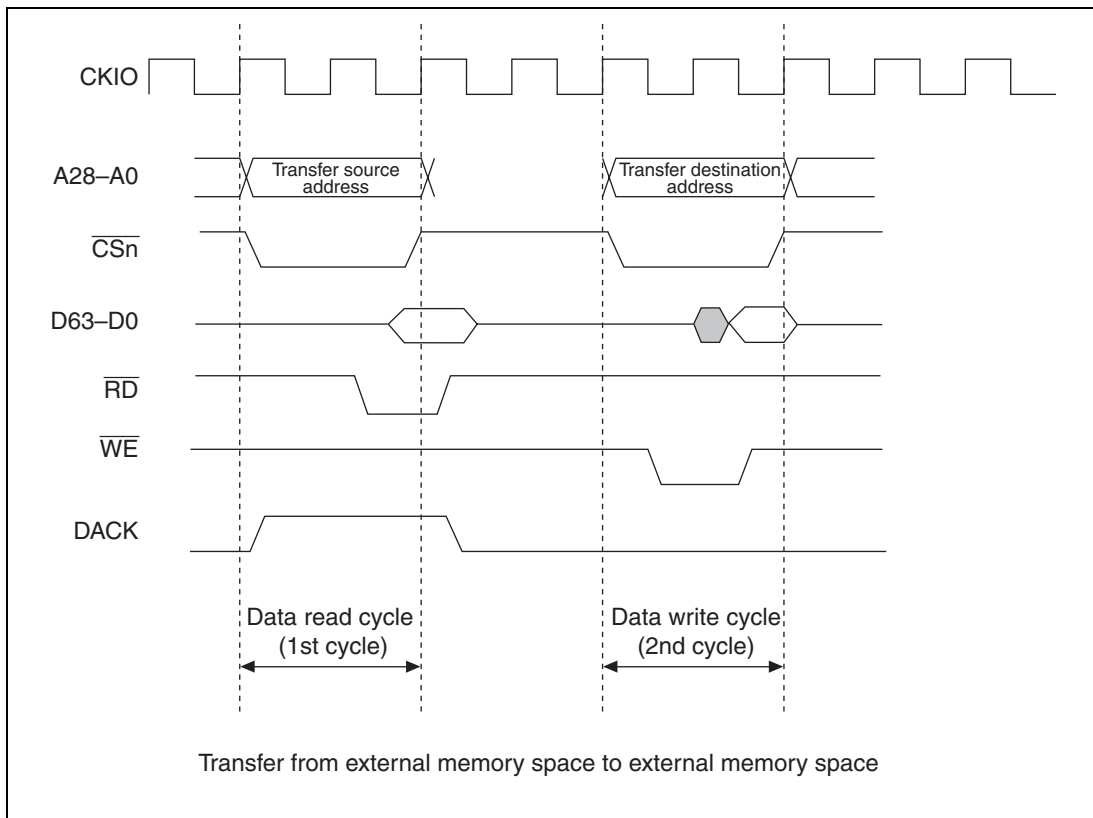


Figure 14.8 Example of Transfer Timing in Dual Address Mode

Bus Modes

There are two bus modes, cycle steal mode and burst mode, selected with the TM bit in CHCR0-CHCR3.

Cycle Steal Mode: In cycle steal mode, the DMAC releases the bus to the CPU at the end of each transfer-unit (8-bit, 16-bit, 32-bit, 64-bit, or 32-byte) transfer. When the next transfer request is issued, the DMAC reacquires the bus from the CPU and carries out another transfer-unit transfer. At the end of this transfer, the bus is again given to the CPU. This is repeated until the transfer end condition is satisfied.

Cycle steal mode can be used with all categories of transfer request source, transfer source, and transfer destination.

Figure 14.9 shows an example of DMA transfer timing in cycle steal mode. The transfer conditions in this example are dual address mode and $\overline{\text{DREQ}}$ level detection.

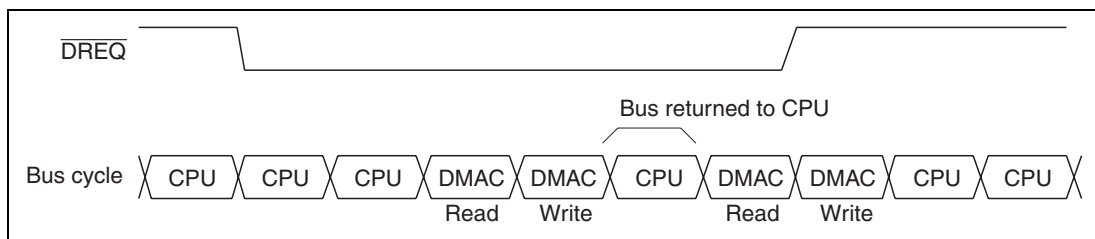


Figure 14.9 Example of DMA Transfer in Cycle Steal Mode

Burst Mode: In burst mode, once the DMAC has acquired the bus it holds the bus and transfers data continuously until the transfer end condition is satisfied. Bus release by means of $\overline{\text{BREQ}}$ and refresh requests conform to the DMAC burst mode transfer priority specification in bus control register 1 (BCRL.DMABST). With $\overline{\text{DREQ}}$ low level detection in external request mode, however, when $\overline{\text{DREQ}}$ is driven high the bus passes to another bus master after the end of the DMAC transfer request that has already been accepted, even if the transfer end condition has not been satisfied.

Figure 14.10 shows an example of DMA transfer timing in burst mode. The transfer conditions in this example are single address mode and $\overline{\text{DREQ}}$ level detection (CHCRn.DS = 0, CHCRn.TM = 1).

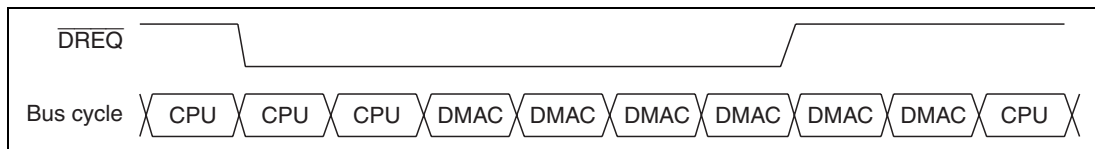


Figure 14.10 Example of DMA Transfer in Burst Mode

Note: Burst mode can be set regardless of the transfer size. A 32-byte block transfer burst mode setting can also be made.

Relationship between DMA Transfer Type, Request Mode, and Bus Mode

Table 14.7 shows the relationship between the type of DMA transfer, the request mode, and the bus mode.

Table 14.7 Relationship between DMA Transfer Type, Request Mode, and Bus Mode

| Address Mode | Type of Transfer | Request Mode | Bus Mode | Transfer Size (Bits) | Usable Channels |
|--------------|---|---|-------------------|--------------------------|----------------------------|
| Single | External device with DACK and external memory | External | B/C | 8/16/32/64/32B | 0, 1 (2, 3)* ⁶ |
| | External device with DACK and memory-mapped external device | External | B/C | 8/16/32/64/32B | 0, 1 (2, 3)* ⁶ |
| Dual | External memory and external memory | Internal* ¹ , external* ⁷ | B/C | 8/16/32/64/32B | 0, 1, 2, 3* ^{5*6} |
| | External memory and memory-mapped external device | Internal* ¹ , external* ⁷ | B/C | 8/16/32/64/32B | 0, 1, 2, 3* ^{5*6} |
| | Memory-mapped external device and memory-mapped external device | Internal* ¹ , external* ⁷ | B/C | 8/16/32/64/32B | 0, 1, 2, 3* ^{5*6} |
| | External memory and on-chip peripheral module | Internal* ² | B/C* ³ | 8/16/32/64* ⁴ | 0, 1, 2, 3* ^{5*6} |
| | Memory-mapped external device and on-chip peripheral module | Internal* ² | B/C* ³ | 8/16/32/64* ⁴ | 0, 1, 2, 3* ^{5*6} |

Legend:

32B: 32-byte burst transfer

B: Burst

C: Cycle steal

External: External request

Internal: Auto request, on-chip peripheral module request

- Notes: 1. External request, auto-request, or on-chip peripheral module request (TMU input capture interrupt request) possible. In the case of an on-chip peripheral module request, it is not possible to specify external memory data transfer with the SCI (SCIF) as the transfer request source.
2. Auto-request, or on-chip peripheral module request possible. If the transfer request source is the SCI (SCIF), either the transfer source must be SCRDR1 (SCFRDR2) or the transfer destination must be SCTDR1 (SCFTDR2).
3. When the transfer request source is the SCI (SCIF), only cycle steal mode can be used.
4. Access size permitted for the on-chip peripheral module register that is the transfer source or transfer destination.

5. When the transfer request is an external request, only channels 0 and 1 can be used.
6. In DDT mode, transfer requests can be accepted for all channels from external devices capable of DTR format output.
7. See tables 14.8 and 14.9 for the transfer sources and transfer destinations in DMA transfer by means of an external request.

(a) Normal DMA Mode

Table 14.8 shows the memory interfaces that can be specified for the transfer source and transfer destination in DMA transfer initiated by an external request supported by this LSI in normal DMA mode.

Table 14.8 External Request Transfer Sources and Destinations in Normal DMA Mode

| Transfer Direction (Settable Memory Interface) | | Address Mode | Usable DMAC Channels |
|--|------------------------------|--------------|----------------------|
| Transfer Source | Transfer Destination | | |
| 1 | Synchronous DRAM | Single | 0, 1 |
| 2 | External device with DACK | Single | 0, 1 |
| 3 | SRAM-type, DRAM | Single | 0, 1 |
| 4 | External device with DACK | Single | 0, 1 |
| 5 | Synchronous DRAM | * Dual | 0, 1 |
| 6 | SRAM-type, MPX, PCMCIA | * Dual | 0, 1 |
| 7 | SRAM-type, DRAM, PCMCIA, MPX | * Dual | 0, 1 |
| 8 | SRAM-type, MPX, PCMCIA | * Dual | 0, 1 |

Notes: "SRAM-type" in the table indicates an SRAM, byte control SRAM, or burst ROM setting.

Memory interfaces on which transfer is possible in single address mode are SRAM, byte control SRAM, burst ROM, DRAM, and synchronous DRAM.

When performing dual address mode transfer, make the DACK output setting for the SRAM, byte control SRAM, burst ROM, PCMCIA, or MPX interface.

* DACK output setting in dual address mode transfer

(b) DDT Mode

Table 14.9 shows the memory interfaces that can be specified for the transfer source and transfer destination in DMA transfer initiated by an external request supported by this LSI in DDT mode.

Table 14.9 External Request Transfer Sources and Destinations in DDT Mode

| Transfer Direction (Settable Memory Interface) | | Address Mode | Usable DMAC Channels | |
|--|------------------------------|--------------------------------|----------------------|------------|
| Transfer Source | Transfer Destination | | | |
| 1 | Synchronous DRAM | External device with DACK | Single | 0, 1, 2, 3 |
| 2 | External device with DACK | Synchronous DRAM | Single | 0, 1, 2, 3 |
| 3 | Synchronous DRAM | SRAM-type, MPX, PCMCIA | * Dual | 1, 2, 3 |
| 4 | SRAM-type, MPX, PCMCIA | * Synchronous DRAM | Dual | 1, 2, 3 |
| 5 | SRAM-type, DRAM, PCMCIA, MPX | SRAM-type, MPX, PCMCIA | * Dual | 1, 2, 3 |
| 6 | SRAM-type, MPX, PCMCIA | * SRAM-type, DRAM, PCMCIA, MPX | Dual | 1, 2, 3 |

Notes: "SRAM-type" in the table indicates an SRAM, byte control SRAM, or burst ROM setting.

The only memory interface on which single address mode transfer is possible in DDT mode is synchronous DRAM.

When performing dual address mode transfer, make the DACK output setting for the SRAM, byte control SRAM, burst ROM, PCMCIA, or MPX interface.

* DACK output setting in dual address mode transfer

Bus Mode and Channel Priority Order

When, for example, channel 1 is transferring data in burst mode, and a transfer request is issued to channel 0, which has a higher priority, the channel 0 transfer is started immediately.

If fixed mode has been set for the priority levels ($CH0 > CH1$), transfer on channel 1 is continued after transfer on channel 0 is completely finished, whether cycle steal mode or burst mode is set for channel 0.

If round robin mode has been set for the priority levels, transfer on channel 1 is restarted after one transfer unit of data is transferred on channel 0, whether cycle steal mode or burst mode is set for channel 0. Channel execution alternates in the order: channel 1 → channel 0 → channel 1 → channel 0.

An example of round robin mode operation is shown in figure 14.11.

Since channel 1 is in burst mode (in the case of edge sensing) regardless of whether fixed mode or round robin mode is set for the priority order, the bus is not released to the CPU until channel 1 transfer ends.

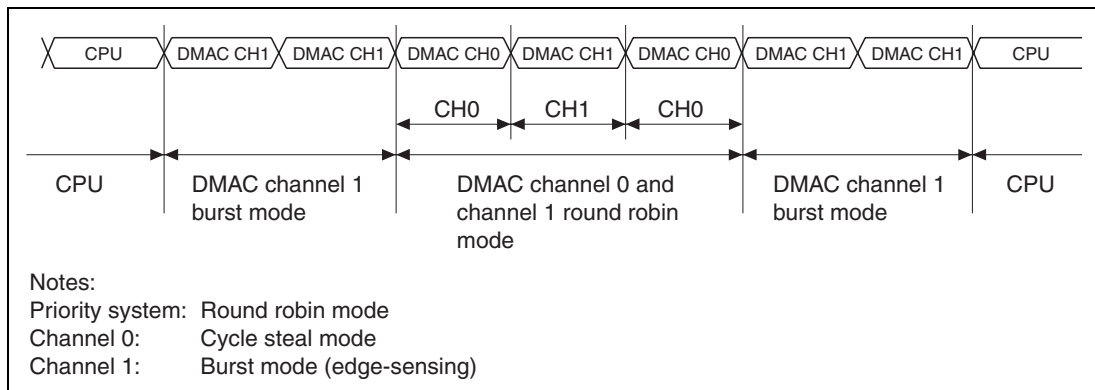


Figure 14.11 Bus Handling with Two DMAC Channels Operating

Note: When channel 1 is in level-sensing burst mode with the settings shown in figure 14.11, the bus is passed to the CPU during a break in requests.

14.3.5 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sampling 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 (BSC) just as it is when the CPU is the bus master. See section 13, Bus State Controller (BSC), for details.

$\overline{\text{DREQ}}$ Pin Sampling Timing: In external request mode, the $\overline{\text{DREQ}}$ pin is sampled at the rising edge of CKIO clock pulses. When $\overline{\text{DREQ}}$ input is detected, a DMAC bus cycle is generated and DMA transfer executed after four CKIO cycles at the earliest.

With $\overline{\text{DREQ}}$ falling edge detection, as the signal passes via an asynchronous circuit the DMAC recognizes $\overline{\text{DREQ}}$ two cycles (CKIO) later (one cycle (CKIO) later in the case of low level detection).

The second and subsequent $\overline{\text{DREQ}}$ sampling operations are performed one cycle after the start of the first DMAC transfer bus cycle (in the case of single address mode).

DRAK is output for one cycle only, once each time $\overline{\text{DREQ}}$ is detected, regardless of the transfer mode or $\overline{\text{DREQ}}$ detection method. In the case of burst mode edge detection, $\overline{\text{DREQ}}$ is sampled in the first cycle only, and so DRAK is output in the first cycle only.

Operation: Figures 14.12 to 14.22 show the timing in each mode.

1. Cycle Steal Mode

In cycle steal mode, The $\overline{\text{DREQ}}$ sampling timing differs for dual address mode and single address mode, and for level detection and edge detection of $\overline{\text{DREQ}}$.

For example, in figure 14.12 (cycle steal mode, dual address mode, level detection), DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation. The second sampling operation is performed one cycle after the start of the first DMAC transfer write cycle. If $\overline{\text{DREQ}}$ is not detected at this time, sampling is executed in every subsequent cycle.

In figure 14.13 (cycle steal mode, dual address mode, edge detection), DMAC transfer begins, at the earliest, five CKIO cycles after the first sampling operation. The second sampling operation begins from the cycle in which the first DMAC transfer read cycle ends. If $\overline{\text{DREQ}}$ is not detected at this time, sampling is executed in every subsequent cycle.

For details of the timing for various memory accesses, see section 13, Bus State Controller (BSC).

Figure 14.18 shows the case of cycle steal mode, single address mode, and level detection. In this case, too, transfer is started, at the earliest, four CKIO cycles after the first $\overline{\text{DREQ}}$ sampling operation. The second sampling operation is performed one cycle after the start of the first DMAC transfer bus cycle.

Figure 14.19 shows the case of cycle steal mode, single address mode, and edge detection. In this case, transfer is started, at the earliest, five CKIO cycles after the first $\overline{\text{DREQ}}$ sampling operation. The second sampling begins one cycle after the first assertion of DRAK.

In single address mode, the DACK signal is output every DMAC transfer cycle.

2. Burst Mode, Dual Address Mode, Level Detection

$\overline{\text{DREQ}}$ sampling timing in burst mode using dual address mode and level detection is virtually the same as for cycle steal mode.

For example, in figure 14.14, DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation. The second sampling operation is performed one cycle after the start of the first DMAC transfer write cycle.

In the case of dual address mode transfer initiated by an external request, the DACK signal can be output in either the read cycle or the write cycle of the DMAC transfer according to the specification of the AM bit in CHCR.

3. Burst Mode, Single Address Mode, Level Detection

$\overline{\text{DREQ}}$ sampling timing in burst mode using single address mode and level detection is shown in figure 14.20.

In the example shown in figure 14.20, DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation, and the second sampling operation begins one cycle after the start of the first DMAC transfer bus cycle.

In single address mode, the DACK signal is output every DMAC transfer cycle.

In figure 14.22, with a 32-byte data size, 32-bit bus width, and SDRAM: row hit write, DMAC transfer begins, at the earliest, six CKIO cycles after the first sampling operation. The second sampling operation begins one cycle after DACK is asserted for the first DMAC transfer.

4. Burst Mode, Dual Address Mode, Edge Detection

In burst mode using dual address mode and edge detection, $\overline{\text{DREQ}}$ sampling is performed in the first cycle only.

For example, in the case shown in figure 14.15, DMAC transfer begins, at the earliest, five CKIO cycles after the first sampling operation. DMAC transfer then continues until the end of the number of data transfers set in DMATCR. $\overline{\text{DREQ}}$ is not sampled during this time, and therefore DRAK is output in the first cycle only.

In the case of dual address mode transfer initiated by an external request, the DACK signal can be output in either the read cycle or the write cycle of the DMAC transfer according to the specification of the AM bit in CHCR.

5. Burst Mode, Single Address Mode, Edge Detection

In burst mode using single address mode and edge detection, $\overline{\text{DREQ}}$ sampling is performed only in the first cycle.

For example, in the case shown in figure 14.21, DMAC transfer begins, at the earliest, five cycles after the first sampling operation. DMAC transfer then continues until the end of the number of data transfers set in DMATCR. $\overline{\text{DREQ}}$ is not sampled during this time, and therefore DRAK is output in the first cycle only.

In single address mode, the DACK signal is output every DMAC transfer cycle.

Suspension of DMA Transfer in Case of $\overline{\text{DREQ}}$ Level Detection

With $\overline{\text{DREQ}}$ level detection in burst mode or cycle steal mode, and in dual address mode or single address mode, the external device for which DMA transfer is being executed can judge from the rising edge of CKIO that DRAK has been asserted, and can suspend DMA transfer by negating $\overline{\text{DREQ}}$. In this case, the next DRAK signal is not output.

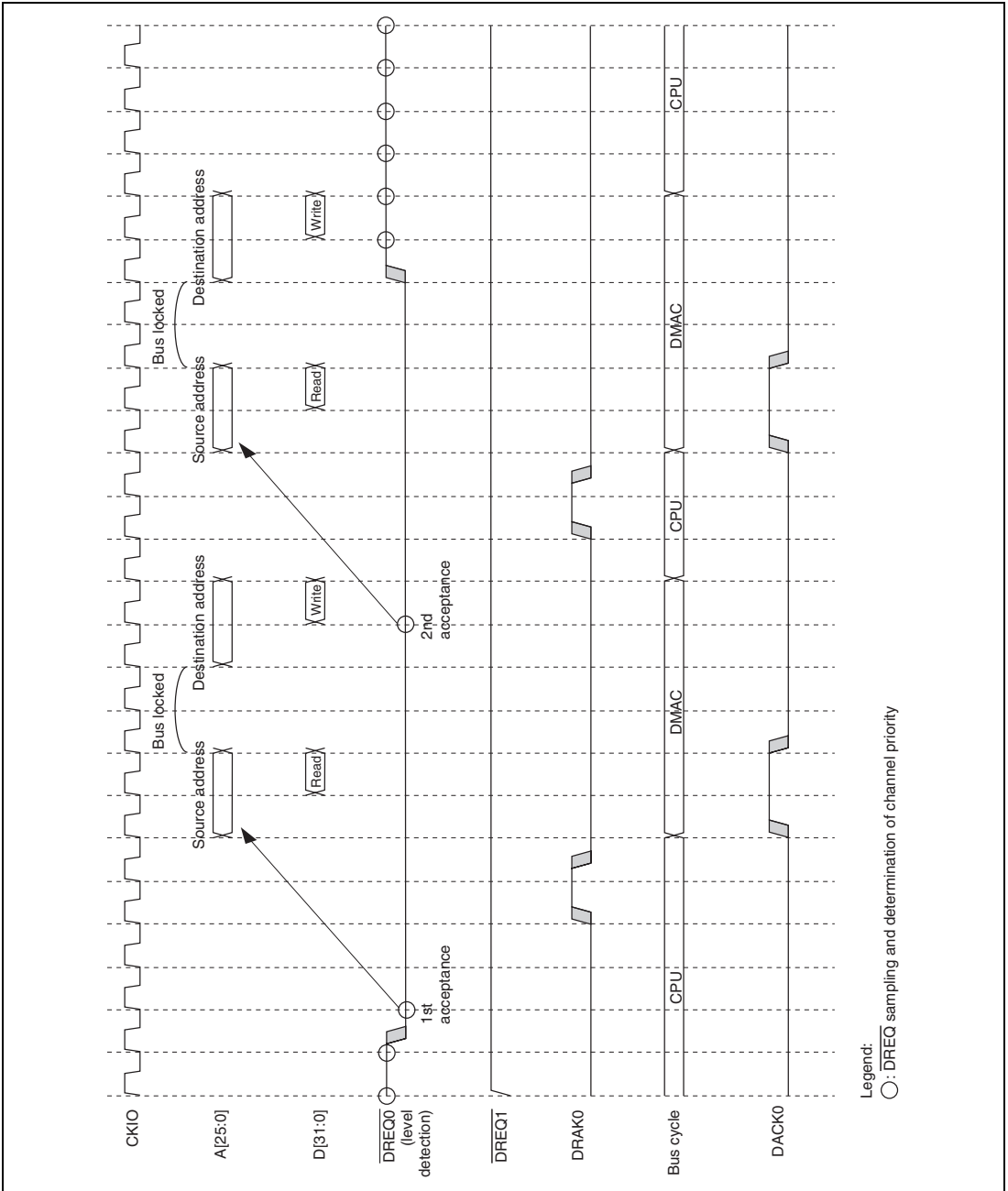


Figure 14.12 Dual Address Mode/Cycle Steal Mode
External Bus → External Bus/DREQ (Level Detection), DACK (Read Cycle)

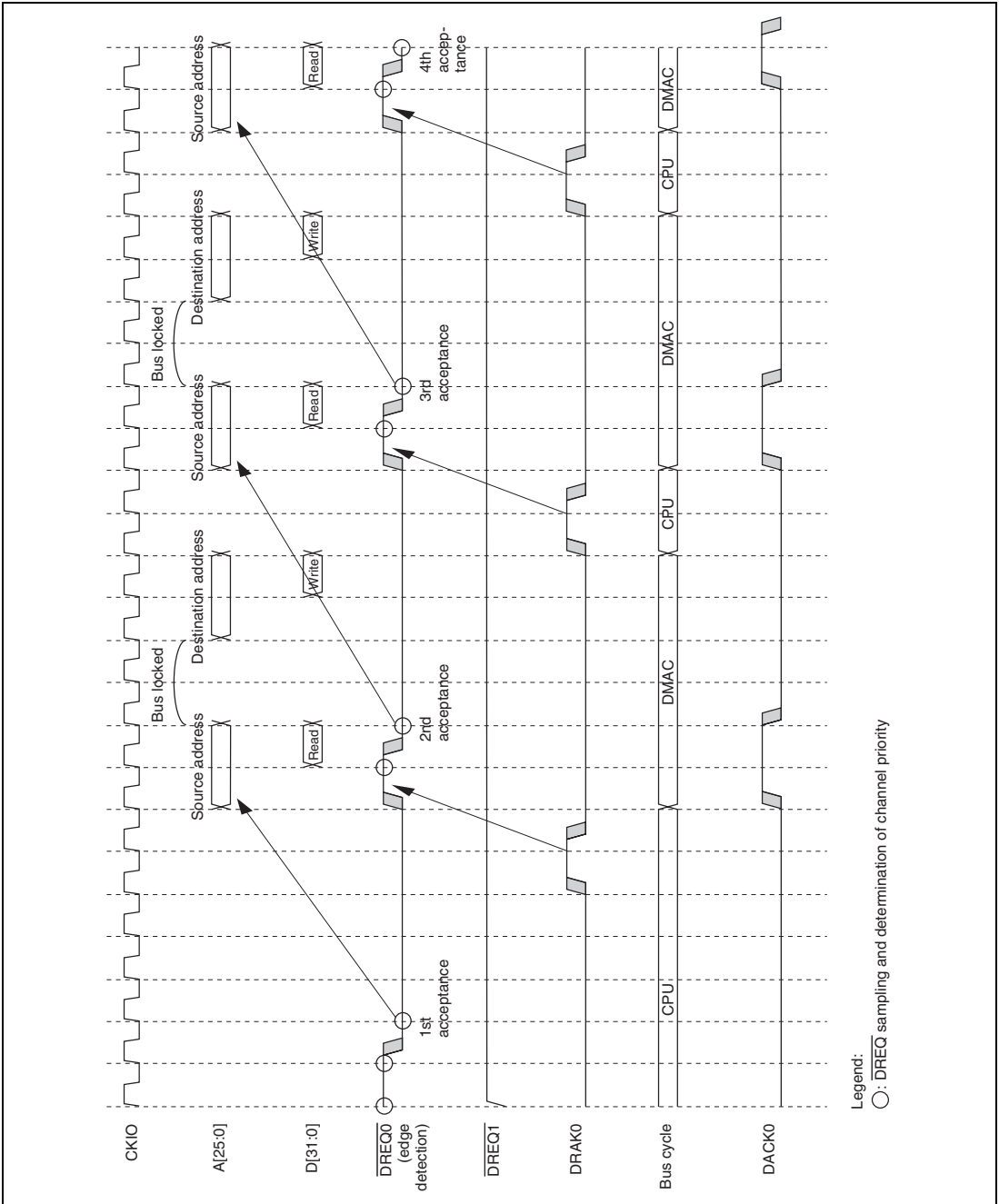


Figure 14.13 Dual Address Mode/Cycle Steal Mode
 External Bus → External Bus/ $\overline{\text{DREQ}}$ (Edge Detection), $\overline{\text{DACK}}$ (Read Cycle)

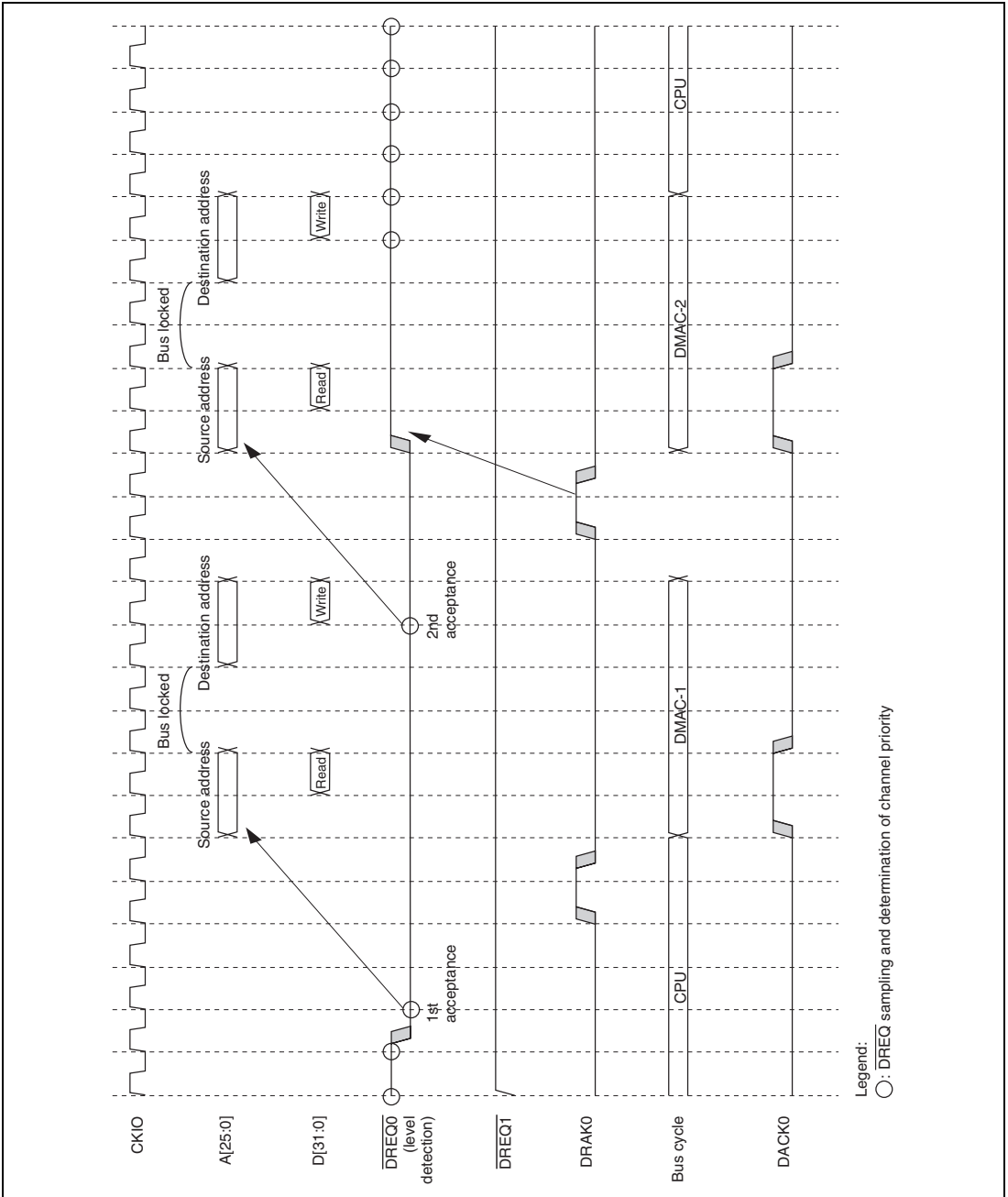
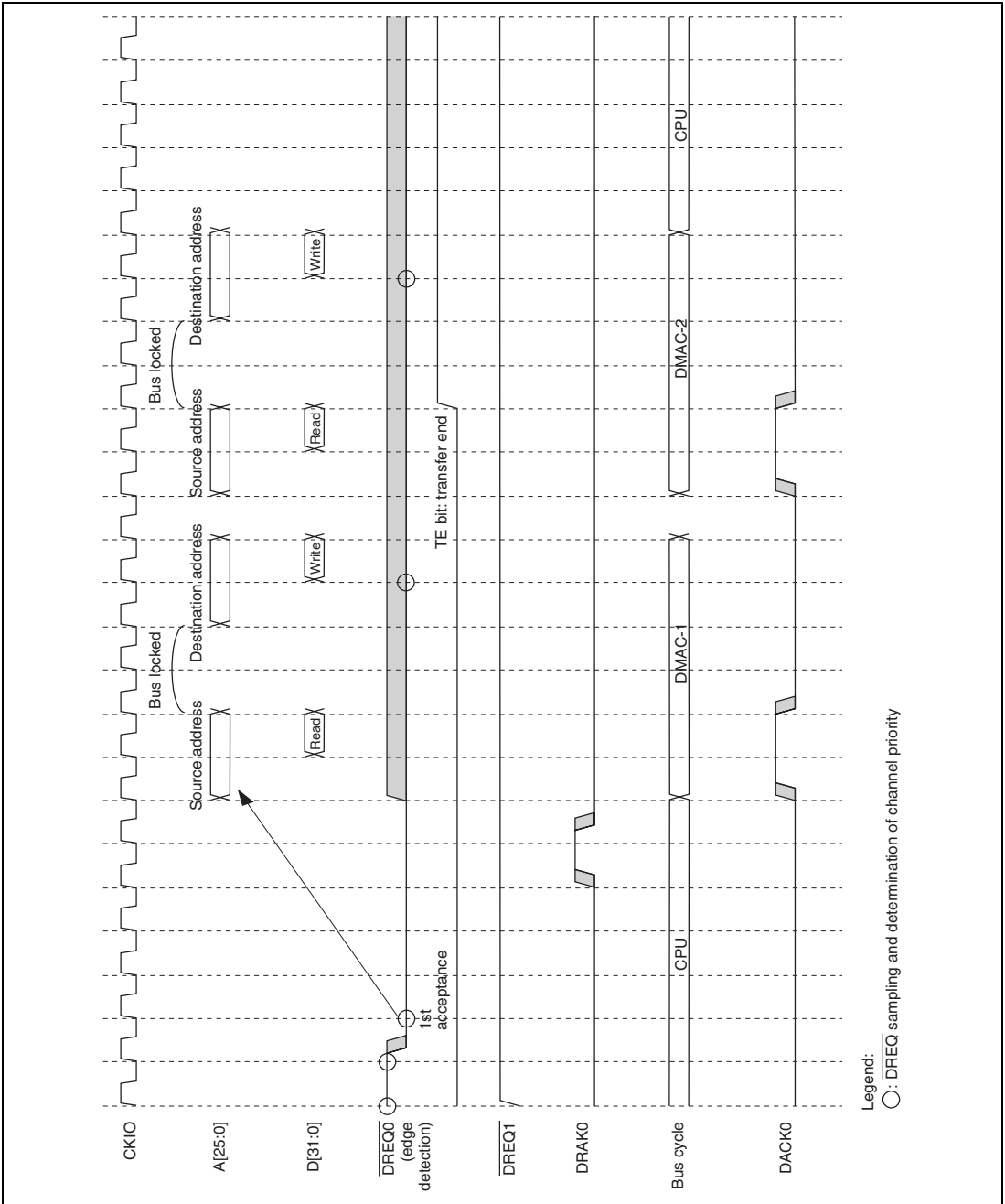
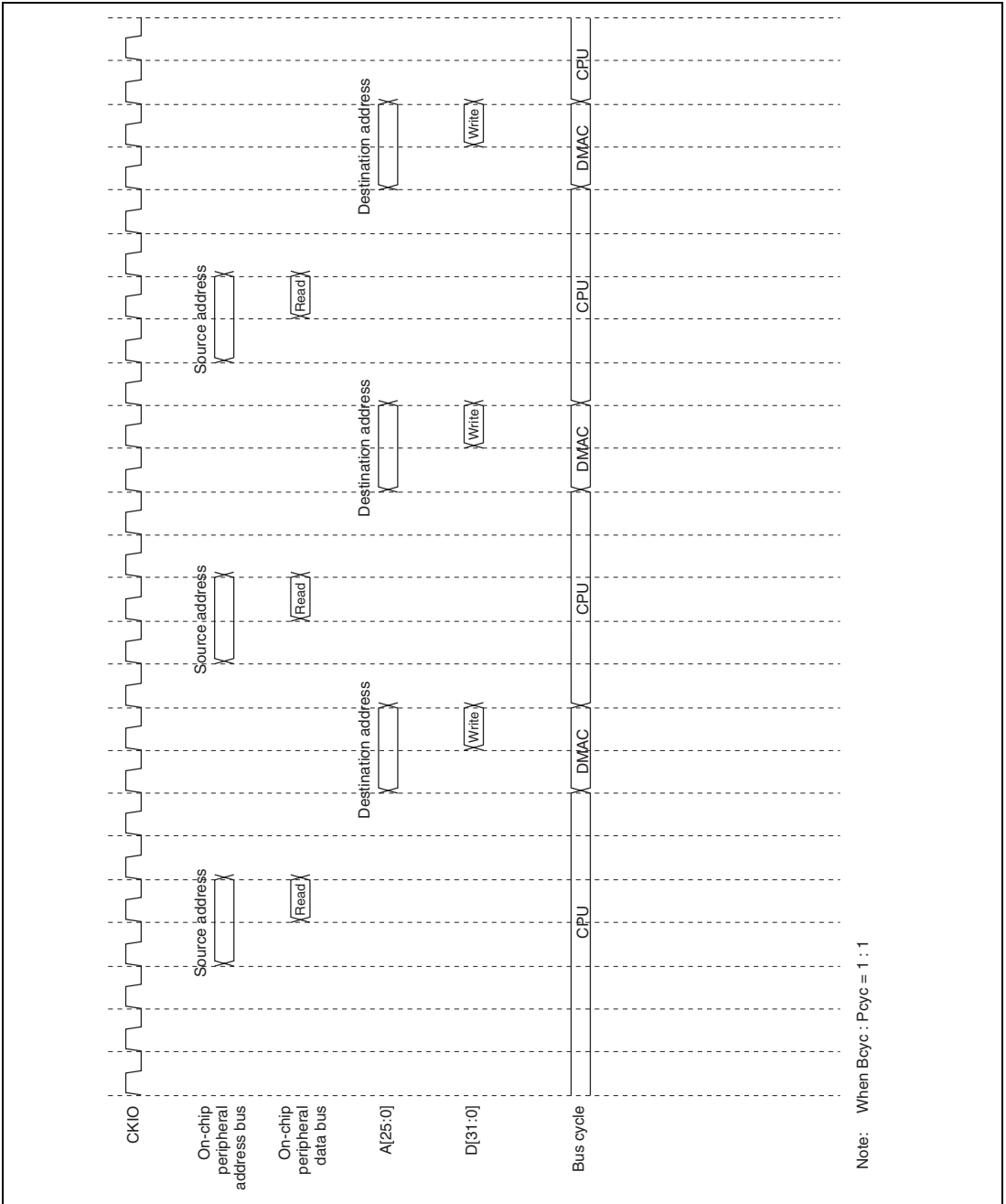


Figure 14.14 Dual Address Mode/Burst Mode
 External Bus → External Bus/ $\overline{\text{DREQ}}$ (Level Detection), DACK (Read Cycle)

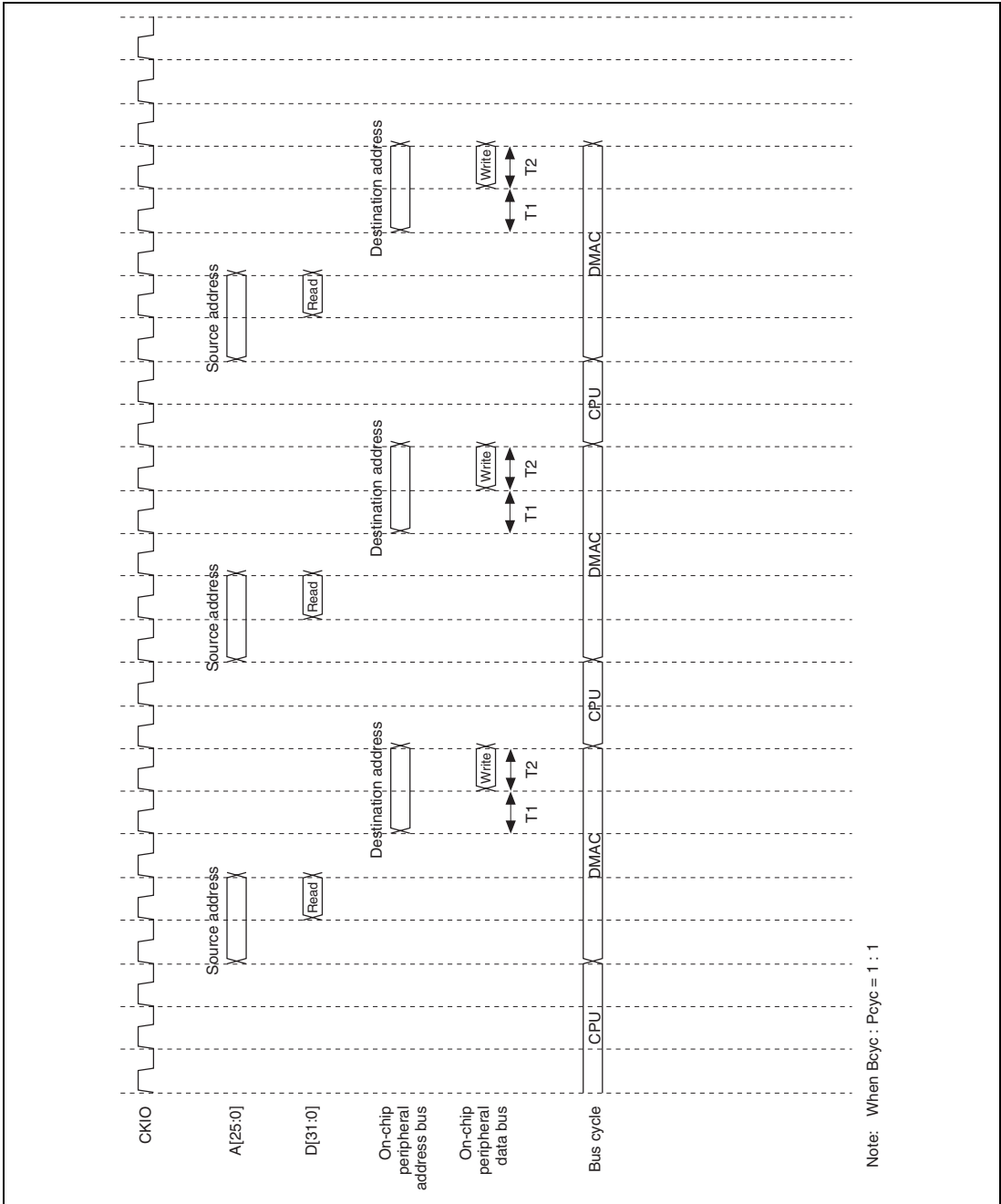


Legend:
 ○ : DREQ sampling and determination of channel priority

Figure 14.15 Dual Address Mode/Burst Mode
External Bus → External Bus/DREQ (Edge Detection), DACK (Read Cycle)



**Figure 14.16 Dual Address Mode/Cycle Steal Mode
On-Chip SCI (Level Detection) → External Bus**



**Figure 14.17 Dual Address Mode/Cycle Steal Mode
External Bus → On-Chip SCI (Level Detection)**

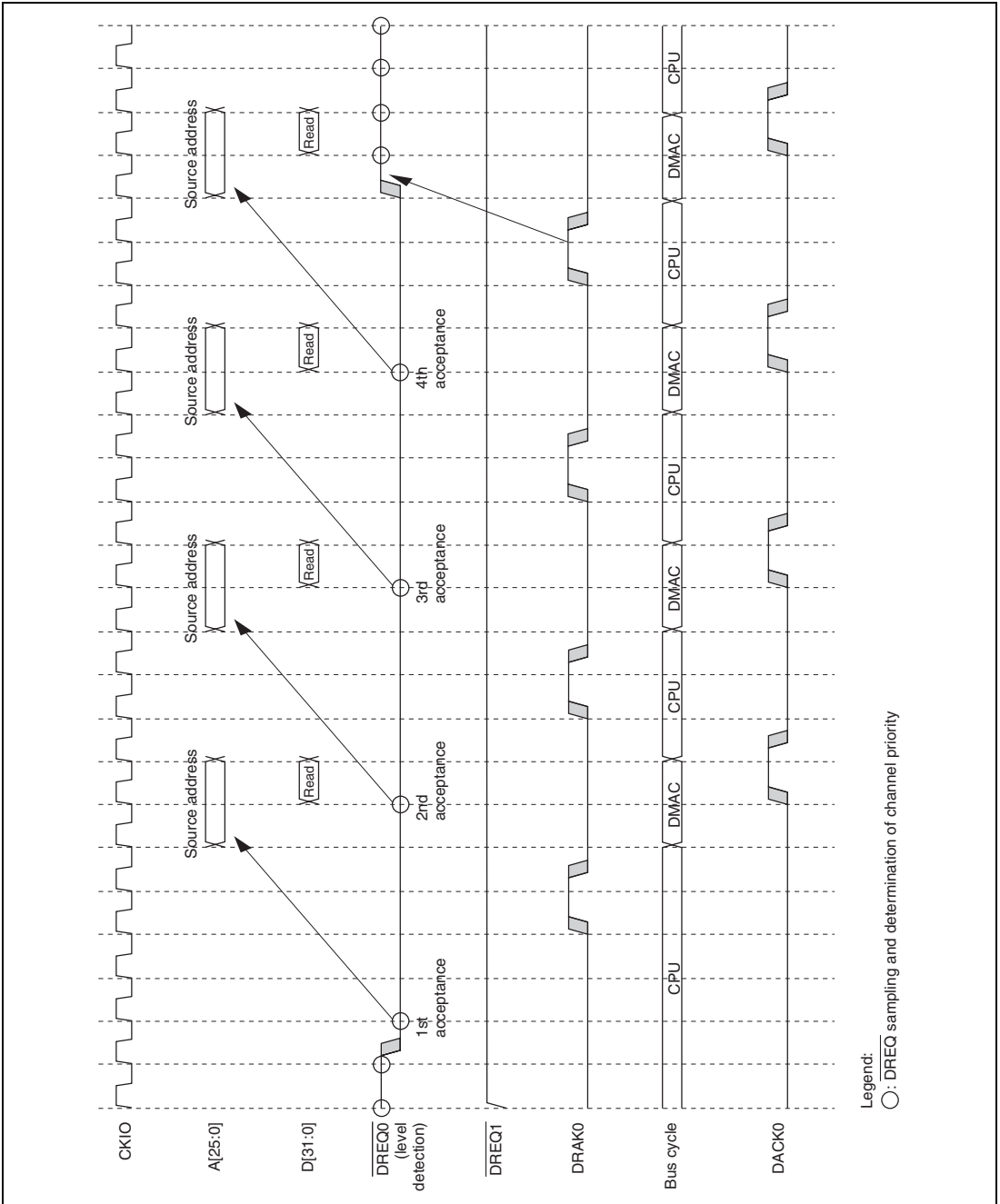
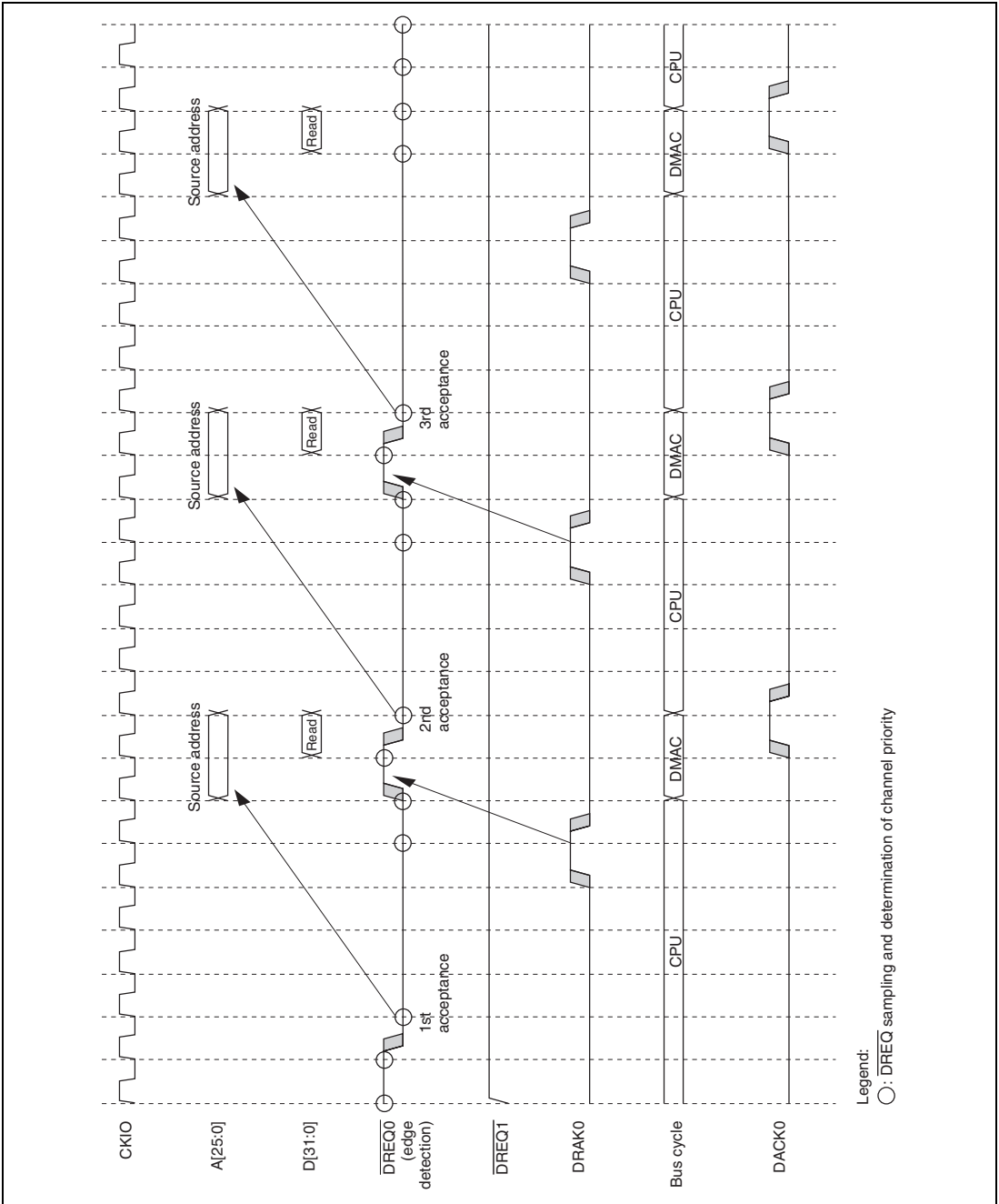


Figure 14.18 Single Address Mode/Cycle Steal Mode
External Bus → External Bus/DREQ (Level Detection)



**Figure 14.19 Single Address Mode/Cycle Steal Mode
External Bus → External Bus/DREQ (Edge Detection)**

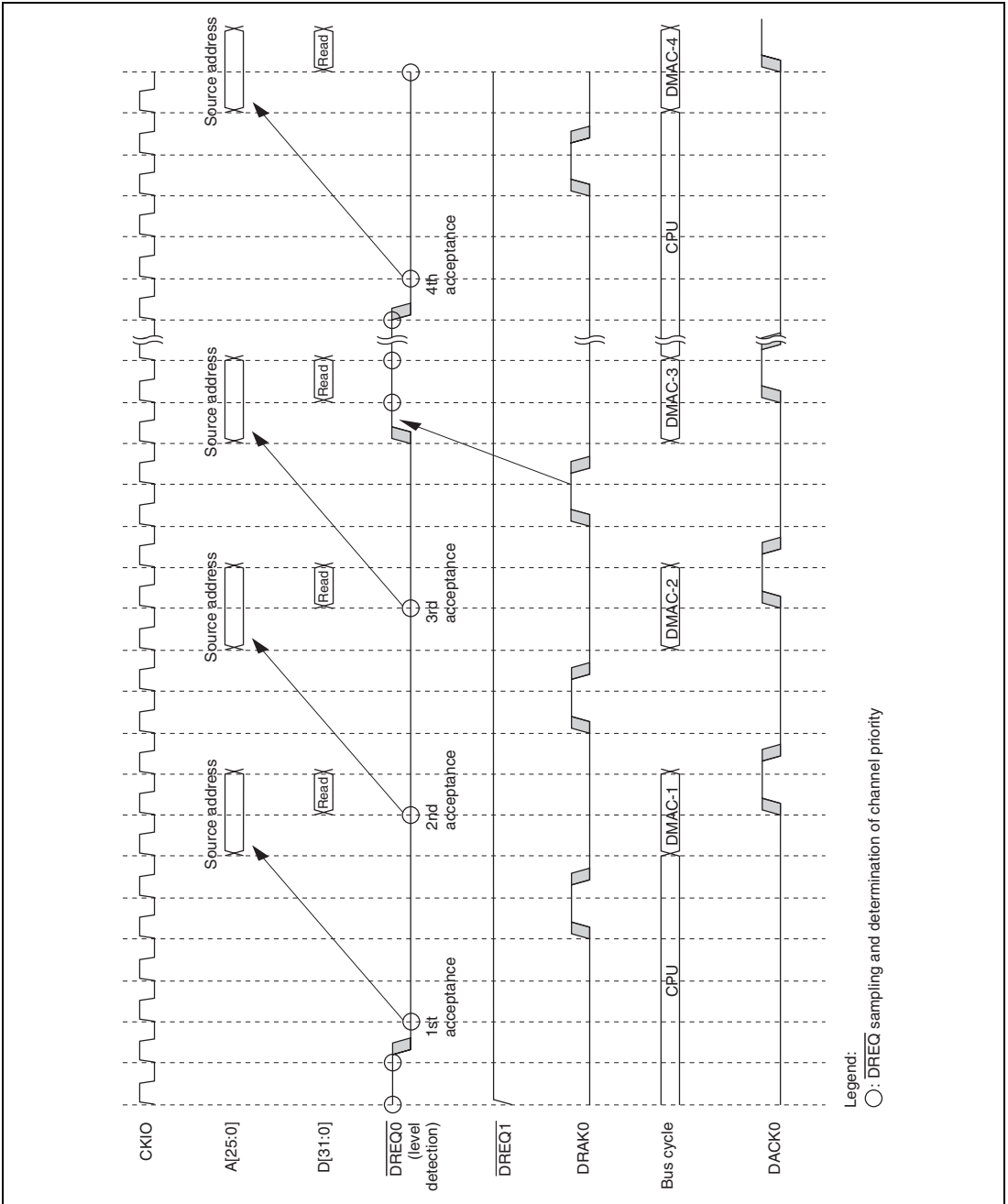


Figure 14.20 Single Address Mode/Burst Mode
External Bus → External Bus/DREQ (Level Detection)

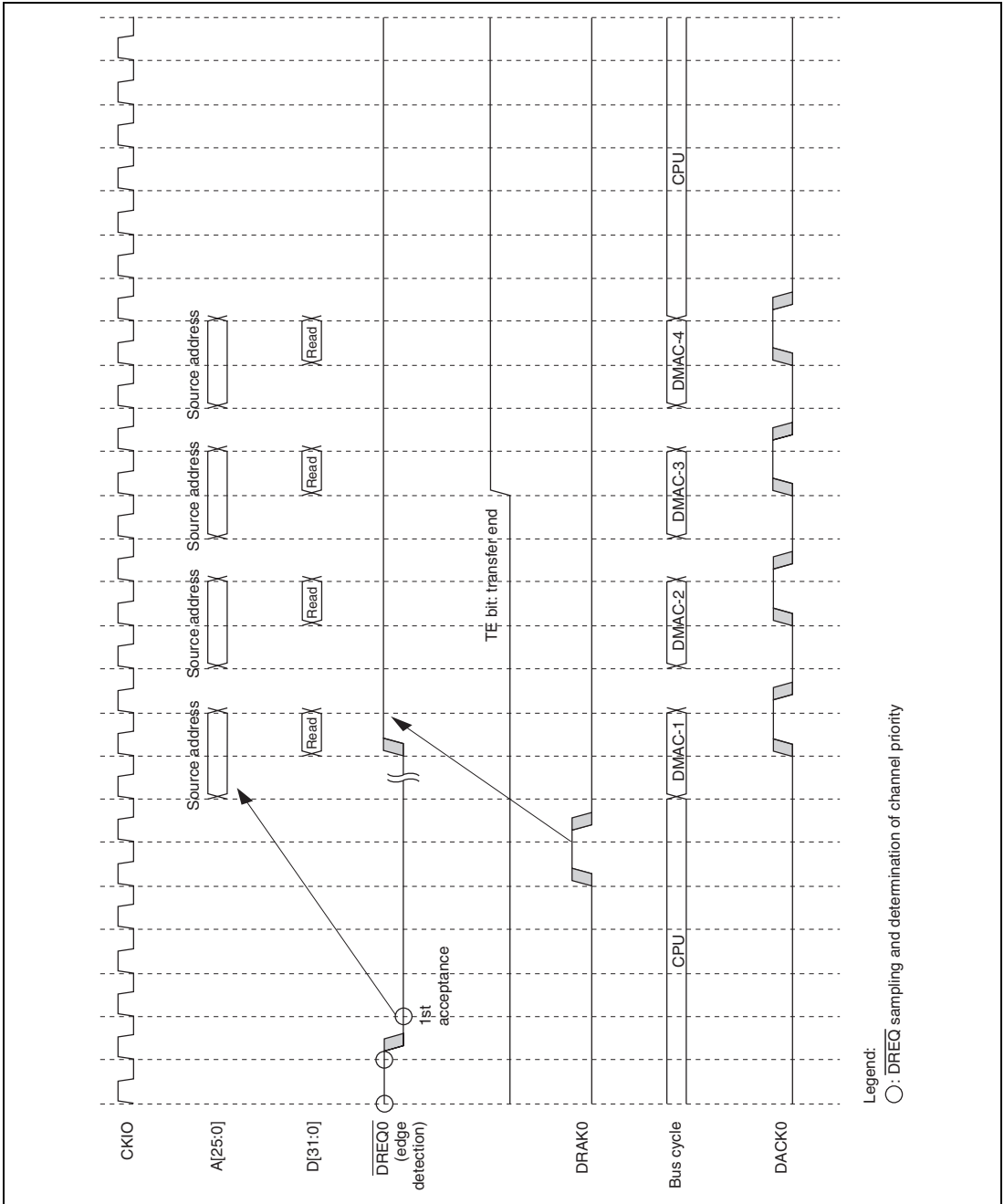


Figure 14.21 Single Address Mode/Burst Mode
External Bus → External Bus/DREQ (Edge Detection)

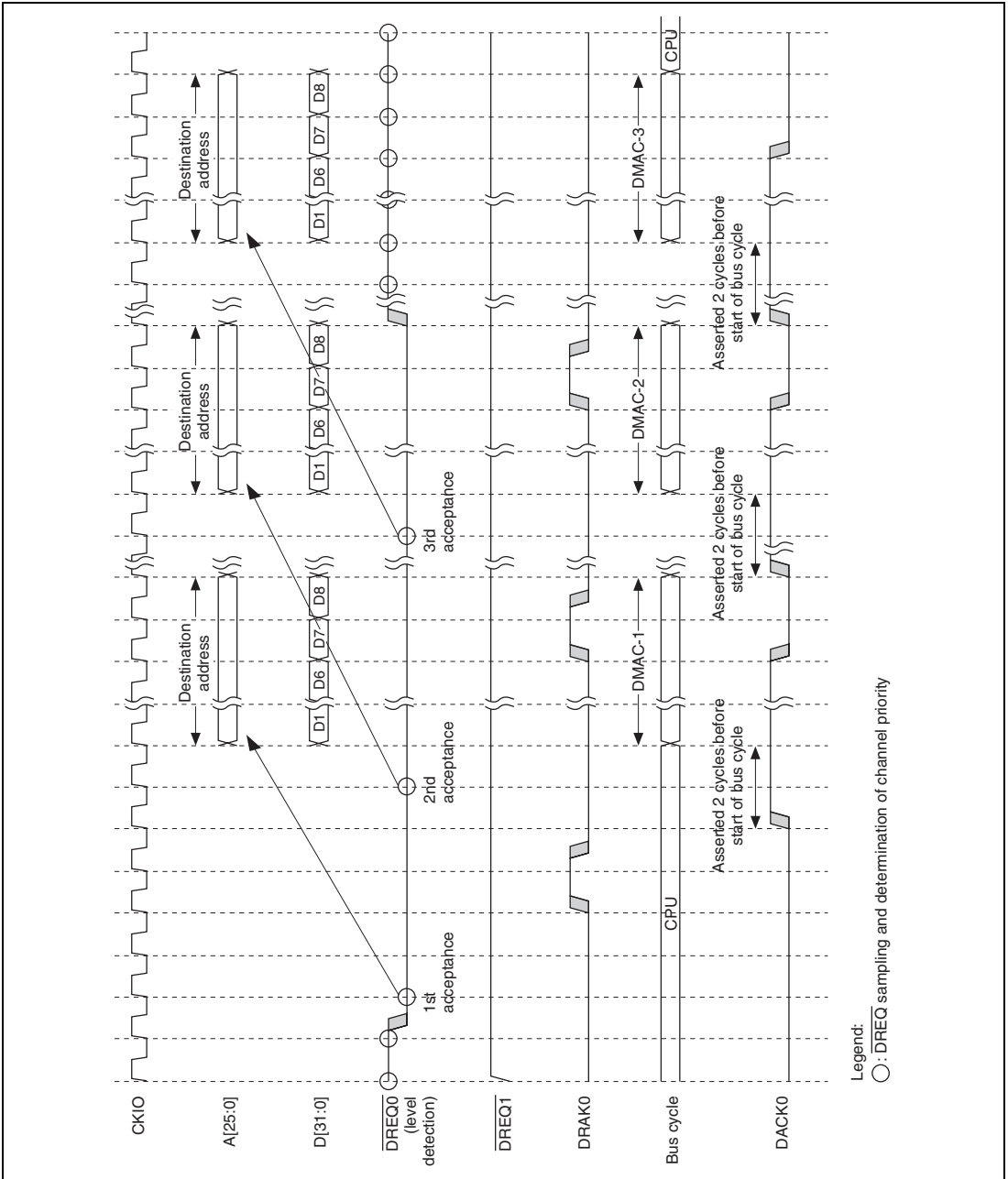


Figure 14.22 Single Address Mode/Burst Mode
External Bus → External Bus/DREQ (Level Detection)/32-Byte Block Transfer
(Bus Width: 32 Bits, SDRAM: Row Hit Write)

14.3.6 Ending DMA Transfer

The conditions for ending DMA transfer are different for ending on individual channels and for ending on all channels together. Except for the case where transfer ends when the value in the DMA transfer count register (DMATCR) reaches 0, the following conditions apply to ending transfer.

1. Cycle Steal Mode (External Request, On-Chip Peripheral Module Request, Auto-Request)

When a transfer end condition is satisfied, acceptance of DMAC transfer requests is suspended. The DMAC completes transfer for the transfer requests accepted up to the point at which the transfer end condition was satisfied, then stops.

In cycle steal mode, the operation is the same for both edge and level transfer request detection.

2. Burst Mode, Edge Detection (External Request, On-Chip Peripheral Module Request, Auto-Request)

The delay between the point at which a transfer end condition is satisfied and the point at which the DMAC actually stops is the same as in cycle steal mode. In burst mode with edge detection, only the first transfer request activates the DMAC, but the timing of stop request ($DE = 0$ in CHCR, $DME = 0$ in DMAOR) sampling is the same as the transfer request sampling timing shown in 4 and 5 under Operation in section 14.3.5, Number of Bus Cycle States and \overline{DREQ} Pin Sampling Timing. Therefore, a transfer request is regarded as having been issued until a stop request is detected, and the corresponding processing is executed before the DMAC stops.

3. Burst Mode, Level Detection (External Request)

The delay between the point at which a transfer end condition is satisfied and the point at which the DMAC actually stops is the same as in cycle steal mode. As in the case of burst mode with edge detection, the timing of stop request ($DE = 0$ in CHCR, $DME = 0$ in DMAOR) sampling is the same as the transfer request sampling timing shown in 2 and 3 under Operation in section 14.3.5, Number of Bus Cycle States and \overline{DREQ} Pin Sampling Timing. Therefore, a transfer request is regarded as having been issued until a stop request is detected, and the corresponding processing is executed before the DMAC stops.

4. Transfer Suspension Bus Timing

Transfer suspension is executed on completion of processing for one transfer unit. In dual address mode transfer, write cycle processing is executed even if a transfer end condition is satisfied during the read cycle, and the transfers covered in 1, 2, and 3 above are also executed before operation is suspended.

Conditions for Ending Transfer on Individual Channels: Transfer ends on the corresponding channel when either of the following conditions is satisfied:

- The value in the DMA transfer count register (DMATCR) reaches 0.
- The DE bit in the DMA channel control register (CHCR) is cleared to 0.

1. End of transfer when DMATCR = 0

When the DMATCR value reaches 0, DMA transfer ends on the corresponding channel and the transfer end flag (TE) in CHCR is set. If the interrupt enable bit (IE) is set at this time, an interrupt (DMTE) request is sent to the CPU.

Transfer ending when DMATCR = 0 does not follow the procedures described in 1, 2, 3, and 4 in section 14.3.6.

2. End of transfer when DE = 0 in CHCR

When the DMA enable bit (DE) in CHCR is cleared, DMA transfer is suspended on the corresponding channel. The TE bit is not set in this case. Transfer ending in this case follows the procedures described in 1, 2, 3, and 4 in section 14.3.6.

Conditions for Ending Transfer Simultaneously on All Channels: Transfer ends on all channels simultaneously when either of the following conditions is satisfied:

- The address error bit (AE) or NMI flag (NMIF) in the DMA operation register (DMAOR) is set.
- The DMA master enable bit (DME) in DMAOR is cleared to 0.

1. End of transfer when AE = 1 in DMAOR

If the AE bit in DMAOR is set to 1 due to an address error, DMA transfer is suspended on all channels in accordance with the conditions in 1, 2, 3, and 4 in section 14.3.6, and the bus is passed to the CPU. Therefore, when AE is set to 1, the values in the DMA source address register (SAR), DMA destination address register (DAR), and DMA transfer count register (DMATCR) indicate the addresses for the DMA transfer to be performed next and the remaining number of transfers. The TE bit is not set in this case. Before resuming transfer, it is necessary to make a new setting for the channel that caused the address error, then write 0 to the AE bit after first reading 1 from it. Acceptance of external requests is suspended while AE is set to 1, so a DMA transfer request must be reissued when resuming transfer. Acceptance of internal requests is also suspended, so when resuming transfer, the DMA transfer request enable bit for the relevant on-chip peripheral module must be cleared to 0 before the new setting is made.

2. End of transfer when NMIF = 1 in DMAOR

If the NMIF bit in DMAOR is set to 1 due to an NMI interrupt, DMA transfer is suspended on all channels in accordance with the conditions in 1, 2, 3, and 4 in section 14.3.6, and the bus is passed to the CPU. Therefore, when NMIF is set to 1, the values in the DMA source address register (SAR), DMA destination address register (DAR), and DMA transfer count register (DMATCR) indicate the addresses for the DMA transfer to be performed next and the remaining number of transfers. The TE bit is not set in this case. Before resuming transfer after NMI interrupt handling is completed, 0 must be written to the NMIF bit after first reading 1 from it. As in the case of AE being set to 1, acceptance of external requests is suspended while NMIF is set to 1, so a DMA transfer request must be reissued when resuming transfer. Acceptance of internal requests is also suspended, so when resuming transfer, the DMA transfer request enable bit for the relevant on-chip peripheral module must be cleared to 0 before the new setting is made.

3. End of transfer when DME = 0 in DMAOR

If the DME bit in DMAOR is cleared to 0, DMA transfer is suspended on all channels in accordance with the conditions in 1, 2, 3, and 4 in section 14.3.6, and the bus is passed to the CPU. The TE bit is not set in this case. When DME is cleared to 0, the values in the DMA source address register (SAR), DMA destination address register (DAR), and DMA transfer count register (DMATCR) indicate the addresses for the DMA transfer to be performed next and the remaining number of transfers. When resuming transfer, DME must be set to 1. Operation will then be resumed from the next transfer.

14.4 Examples of Use

14.4.1 Examples of Transfer between External Memory and an External Device with DACK

Examples of transfer of data in external memory to an external device with DACK using DMAC channel 1 are considered here.

Table 14.10 shows the transfer conditions and the corresponding register settings.

Table 14.10 Conditions for Transfer between External Memory and an External Device with DACK, and Corresponding Register Settings

| Transfer Conditions | Register | Set Value |
|--|----------|--------------------|
| Transfer source: external memory | SAR1 | H'0C000000 |
| Transfer source: external device with DACK | DAR1 | (Accessed by DACK) |
| Number of transfers: 32 | DMATCR1 | H'00000020 |
| Transfer source address: decremented | CHCR1 | H'000022A5 |
| Transfer destination address: (setting invalid) | | |
| Transfer request source: external pin ($\overline{\text{DREQ1}}$) edge detection | | |
| Bus mode: burst | | |
| Transfer unit: word | | |
| No interrupt request at end of transfer | | |
| Channel priority order: 2 > 0 > 1 > 3 | DMAOR | H'00000201 |

14.5 On-Demand Data Transfer Mode (DDT Mode)

14.5.1 Operation

Setting the DDT bit to 1 in DMAOR causes a transition to on-demand data transfer mode (DDT mode). In DDT mode, it is possible to transfer to channel 0 to 3 via the data bus and DDT module, and simultaneously issue a transfer request, using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID [1:0], DTR.ID, and DTR.MD signals between an external device and the DMAC. Figure 14.23 shows a block diagram of the DMAC, DDT, BSC, and an external device (with $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, ID [1:0], DTR.ID, and DTR.MD pins).

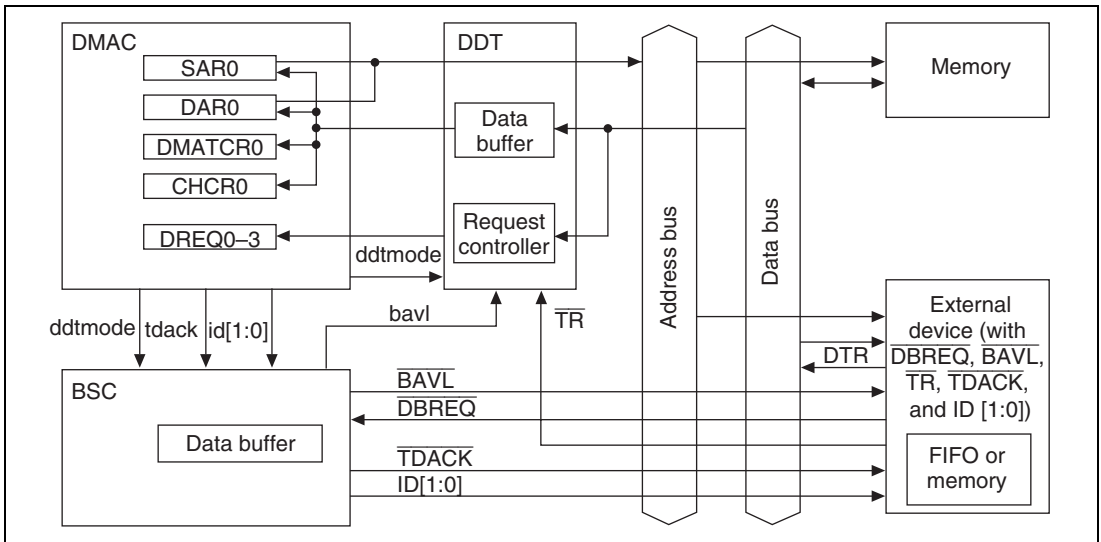


Figure 14.23 On-Demand Transfer Mode Block Diagram

After first making the normal DMA transfer settings for DMAC channels 0 to 3 using the CPU, a transfer request is output from an external device using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, DTR.ID [1:0], and DTR.MD [1:0] signals (handshake protocol using the data bus). A transfer request can also be issued simply by asserting $\overline{\text{TR}}$, without using the external bus (handshake protocol without use of the data bus). For channel 2, after making the DMA transfer settings in the normal way, a transfer request can be issued directly from an external device (with $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, DTR.ID [1:0], and DTR.MD [1:0] pins) by asserting $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ simultaneously.

In DDT mode, there is a choice of five modes for performing DMA transfer.

1. Normal data transfer mode (channel 0)

$\overline{\text{BAVL}}$ (the data bus available signal) is asserted in response to $\overline{\text{DBREQ}}$ (the data bus request signal) from an external device. Two CKIO-synchronous cycles after $\overline{\text{BAVL}}$ is asserted, the external data bus drives the data transfer setting command (DTR command) in synchronization with $\overline{\text{TR}}$ (the transfer request signal). The initial settings are then made in the DMAC channel 0 control register, and the DMA transfer is processed.

2. Normal data transfer mode (except channel 1 to channel 3)

In this mode, the data transfer settings are made in the DMAC from the CPU, and DMA transfer requests only are performed from the external device.

As in 1 above, $\overline{\text{DBREQ}}$ is asserted from the external device and the external bus is secured, then the DTR command is driven.

The transfer request channel can be specified by means of the two ID bits in the DTR command.

3. Handshake protocol using the data bus (valid for channel 0 only)

This mode is only valid for channel 0.

After the initial settings have been made in the DMAC channel 0 control register, the DDT module asserts a data transfer request for the DMAC by setting the DTR command ID = 00, MD = 00, and SZ \neq 101, 110 and driving the DTR command.

4. Handshake protocol without use of the data bus

The DDT module includes a function for recording the previously asserted request channel. By using this function, it is possible to assert a transfer request for the channel for which a request was asserted immediately before, by asserting $\overline{\text{TR}}$ only from an external device after a transfer request has once been made to the channel for which an initial setting has been made in the DMAC control register (DTR command and data transfer setting by the CPU in the DMAC).

5. Direct data transfer mode (valid for channel 2 only)

A data transfer request can be asserted for channel 2 by asserting $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ simultaneously from an external device after the initial settings have been made in the DMAC channel 2 control register.

14.5.2 Pins in DDT Mode

Figure 14.24 shows the system configuration in DDT mode.

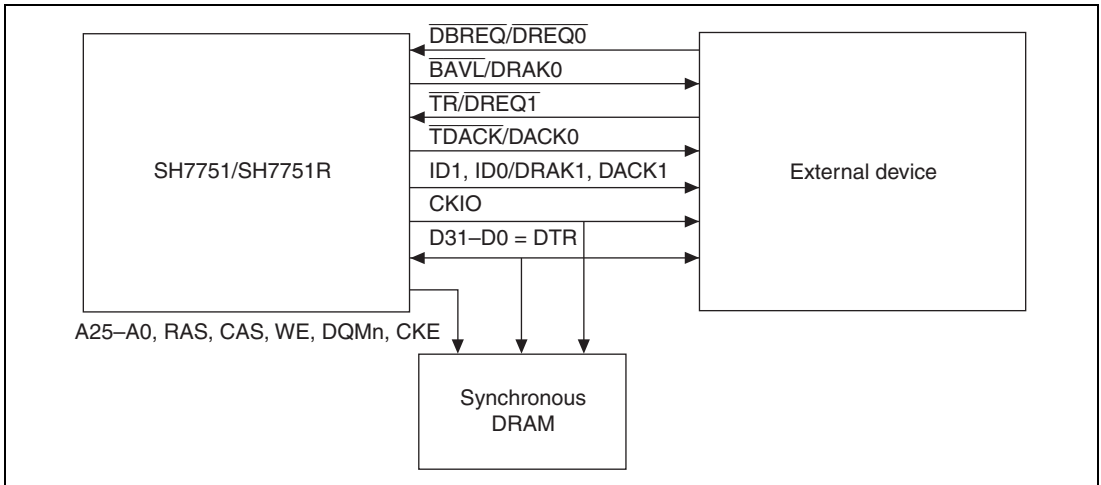


Figure 14.24 System Configuration in On-Demand Data Transfer Mode

- **$\overline{\text{DBREQ}}$** : Data bus release request signal for transmitting the data transfer request format (DTR format) or a DMA request from an external device to the DMAC

If there is a wait for release of the data bus, an external device can have the data bus released by asserting $\overline{\text{DBREQ}}$. When $\overline{\text{DBREQ}}$ is accepted, the BSC asserts $\overline{\text{BAVL}}$.

- **$\overline{\text{BAVL}}$** : Data bus D31–D0 release signal

Assertion of $\overline{\text{BAVL}}$ means that the data bus will be released two cycles later.

- **$\overline{\text{TR}}$** : Transfer request signal

Assertion of $\overline{\text{TR}}$ has the following different meanings.

- In normal data transfer mode (channel 0, except channel 0), $\overline{\text{TR}}$ is asserted, and at the same time the DTR format is output, two cycles after $\overline{\text{BAVL}}$ is asserted.
- In the case of the handshake protocol without use of the data bus, asserting $\overline{\text{TR}}$ enables a transfer request to be issued for the channel for which a transfer request was made immediately before. This function can be used only when $\overline{\text{BAVL}}$ is not asserted two cycles earlier.
- In the case of direct data transfer mode (valid only for channel 2), a direct transfer request can be made to channel 2 by asserting $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ simultaneously.

- 11: Channel 3

Bits 25 and 24: Transfer Request Mode (MD1, MD0)

- 00: Handshake protocol (data bus used)
- 01: Setting prohibited
- 10: Request queue clear specification
- 11: Setting prohibited

Bits 23 to 0: Reserved

Notes: 1. In channels 1 to 3, only the ID field is valid.

2. In channel 0, the MD field is valid. Set MD = 00. If 01, 10, or 11 is set, the DMAC will halt with an address error.
3. In edge-sense burst mode, DMA transfer is executed continuously. In level-sense burst mode and cycle steal mode, a handshake protocol is used to transfer each unit of data.
4. When specifying data transfer requests using a handshake protocol for channel 0, set DTR.ID = 00, DTR.MD = 00, and DTR.SZ ≠ 101, 110 for the DTR format. Use the MOV instruction to make settings in the DMAC's SAR0, DAR0, CHCR0, and DMATCR0 registers. Either single address mode or dual address mode can be used as the transfer mode.

Select one of the following settings: CHCR0.RS3—RS0 = 0000, 0010, 0011.

Operation is not guaranteed if the DTR format data settings are DTR.ID = 00, DTR.MD = 00, and DTR.SZ ≠ 101, 110.

Usable SZ, ID, and MD Combination in DDT Mode

Table 14.11 shows the usable combination of SZ, ID, and MD in DDT mode of this LSI.

Table 14.11 Usable SZ, ID, and MD Combination in DDT Mode

| SZ [2:0] | ID [1:0] | MD [1:0] | Function |
|----------|----------|----------|-----------------------------------|
| 000 | 00 | 00 | Request for transfer to channel 0 |
| 110 | 00 | 10 | Request queue clear |
| 111 | 00 | 00 | Transfer end |
| X | 01 | X | Request for transfer to channel 1 |
| X | 10 | X | Request for transfer to channel 2 |
| X | 11 | X | Request for transfer to channel 3 |

Legend: X: Don't care

Note: Don't set values other than those shown in the above table.

14.5.3 Transfer Request Acceptance on Each Channel

On channel 0, a DMA data transfer request can be made by means of the DTR format. No further transfer requests are accepted between DTR format acceptance and the end of the data transfer.

On channels 1 to 3, output a transfer request from an external device by means of the DTR format (ID = 01, 10, or 11) after making DMAC control register settings in the same way as in normal DMA mode. Each of channels 1 to 3 has a request queue that can accept up to four transfer requests. When a request queue is full, the fifth and subsequent transfer requests will be ignored, and so transfer requests must not be output. When CHCR.TE = 1 when a transfer request remains in the request queue and a transfer is completed, the request queue retains it. When another transfer request is sent at that time, the transfer request is added to the request queue if the request queue is vacant.

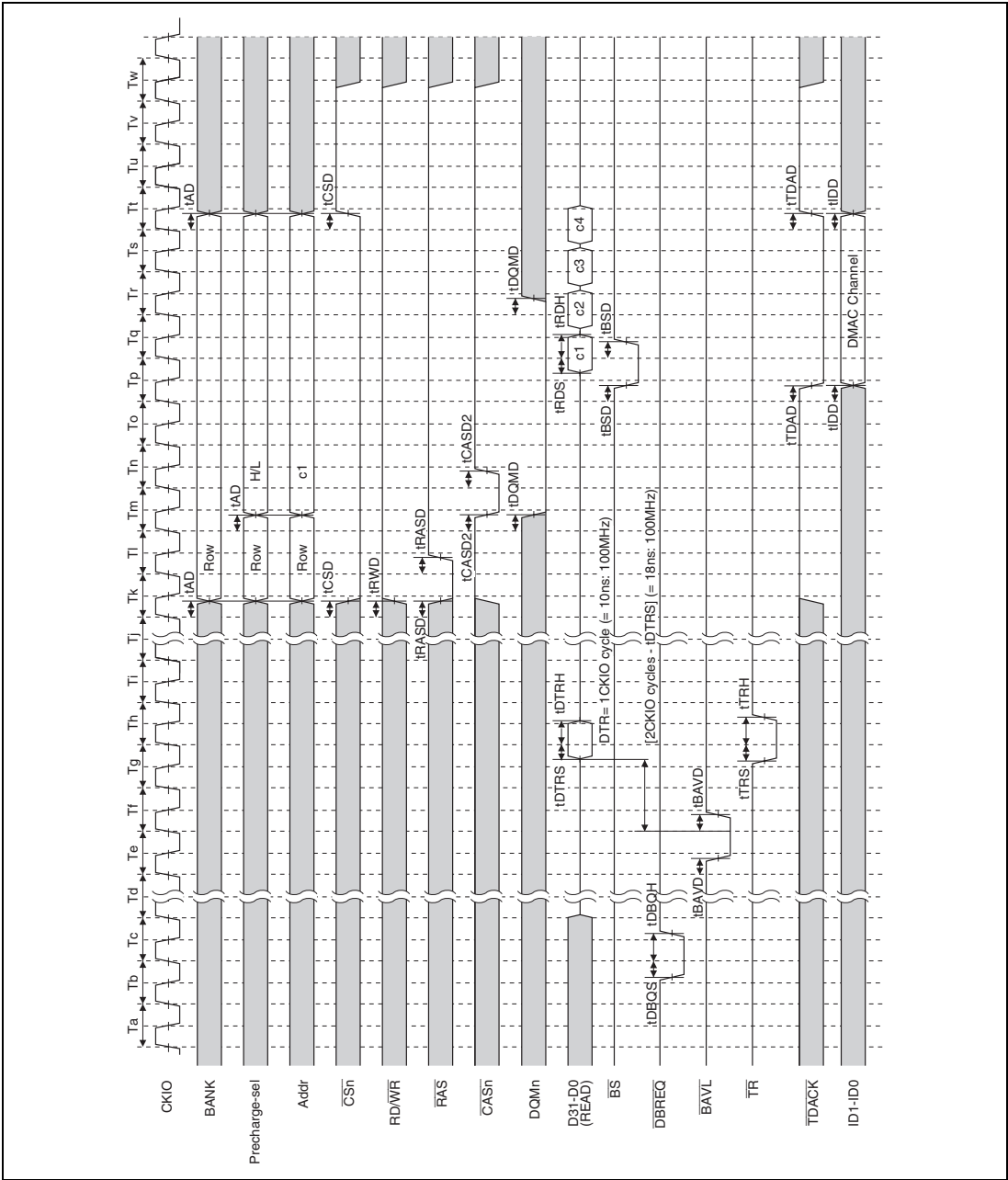


Figure 14.26 Single Address Mode/Synchronous DRAM → External Device Longword Transfer SDRAM Auto-Precharge Read Bus Cycle, Burst (RCD = 1, CAS latency = 3, TPC = 3)

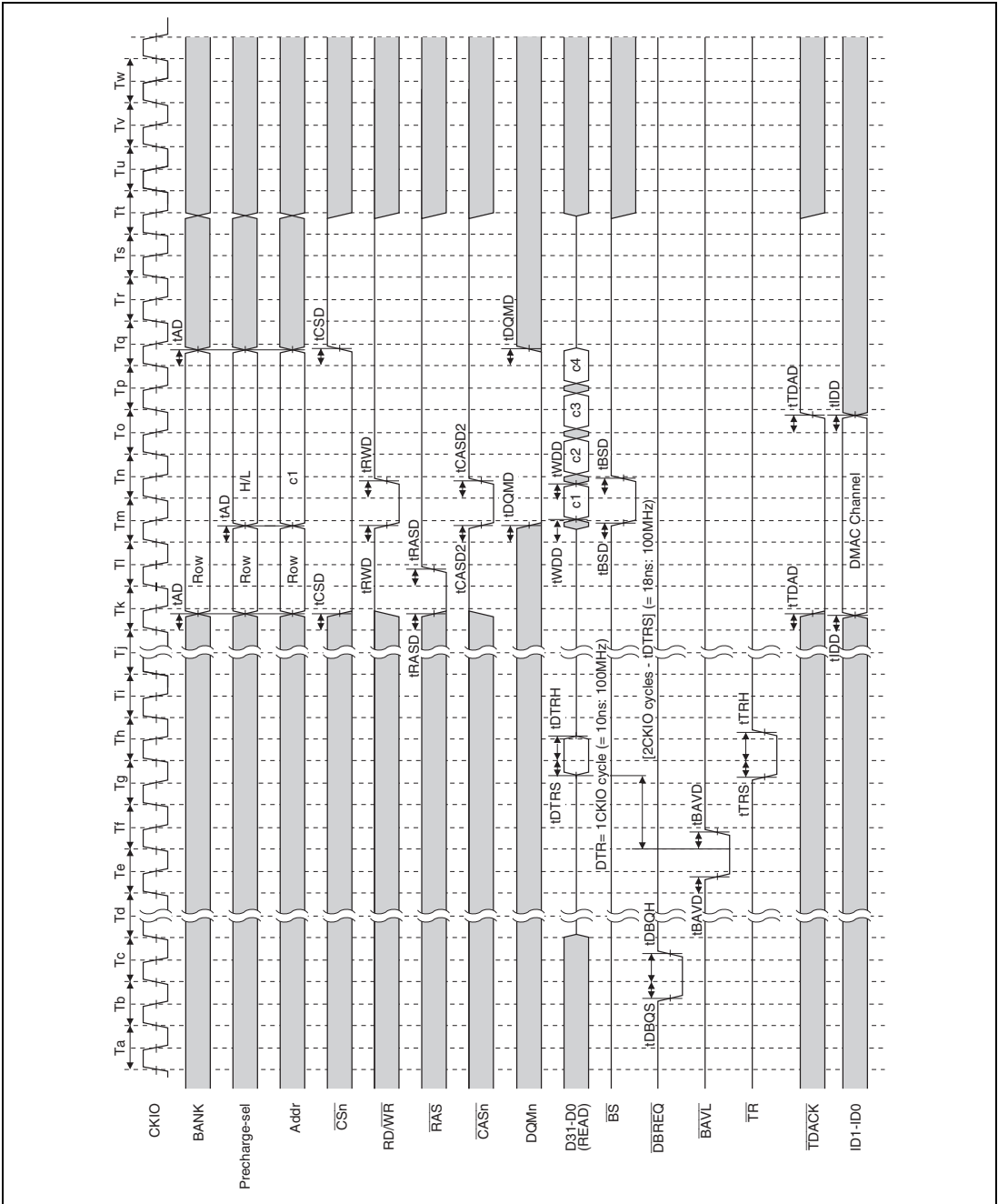


Figure 14.27 Single Address Mode/External Device → Synchronous DRAM Longword Transfer SDRAM Auto-Precharge Write Bus Cycle, Burst (RCD = 1, TRWL = 2, TPC = 1)

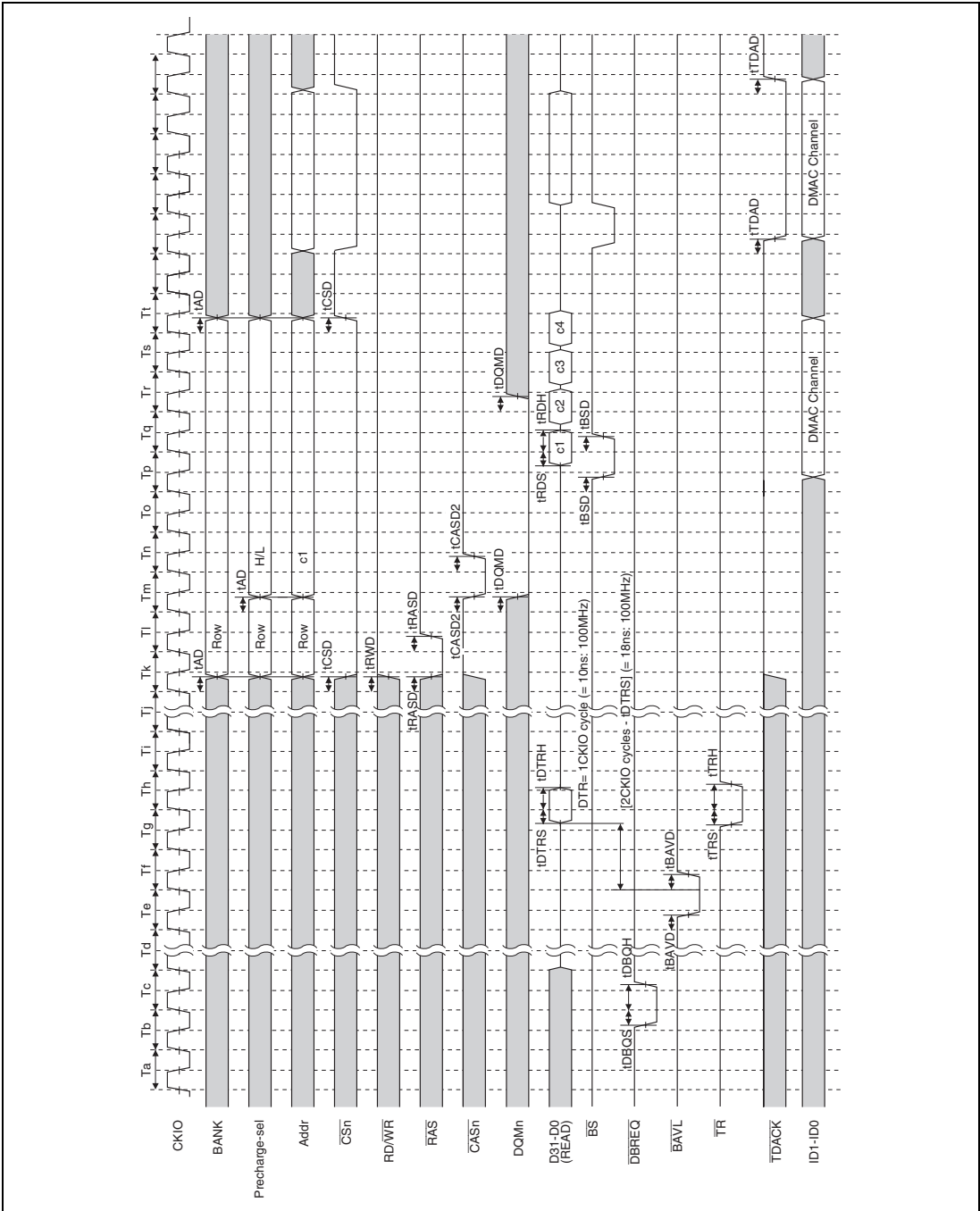


Figure 14.28 Dual Address Mode/Synchronous DRAM → SRAM Longword Transfer

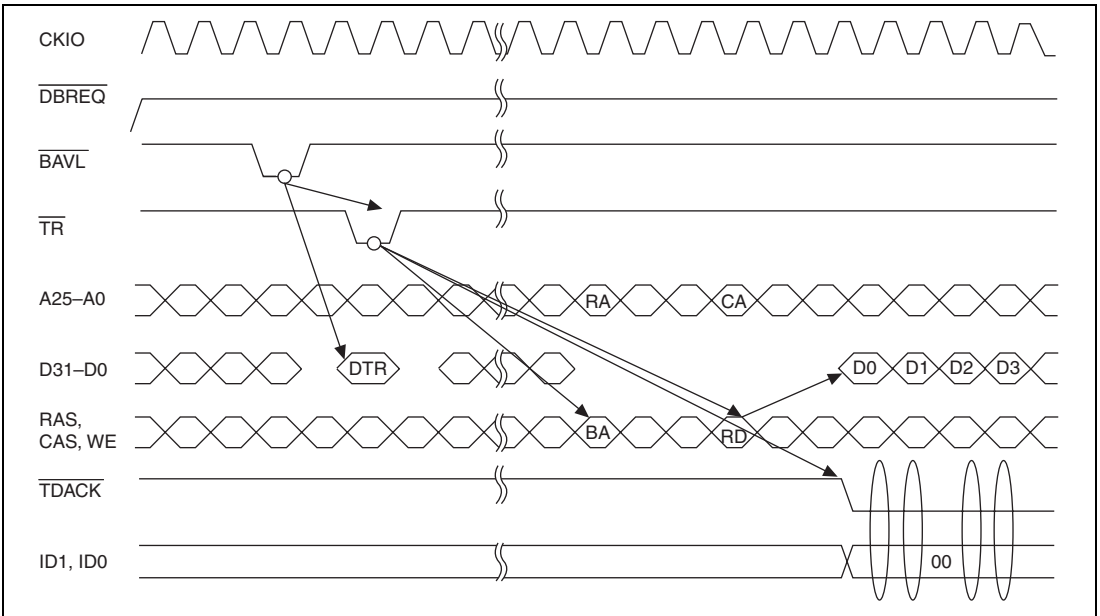


Figure 14.29 Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer

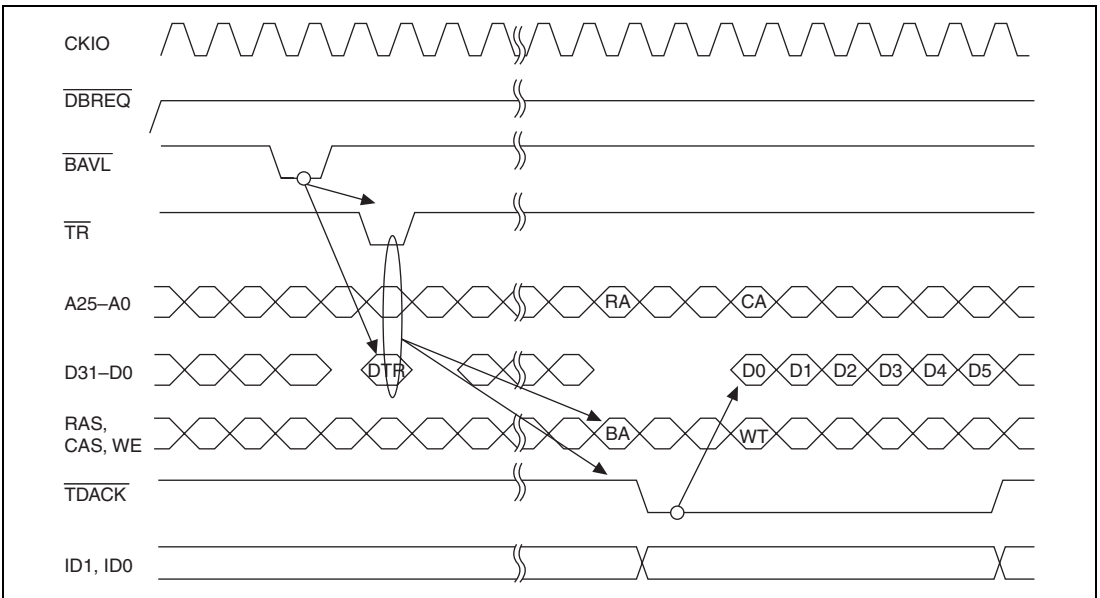


Figure 14.30 Single Address Mode/Burst Mode/External Device → External Bus 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer

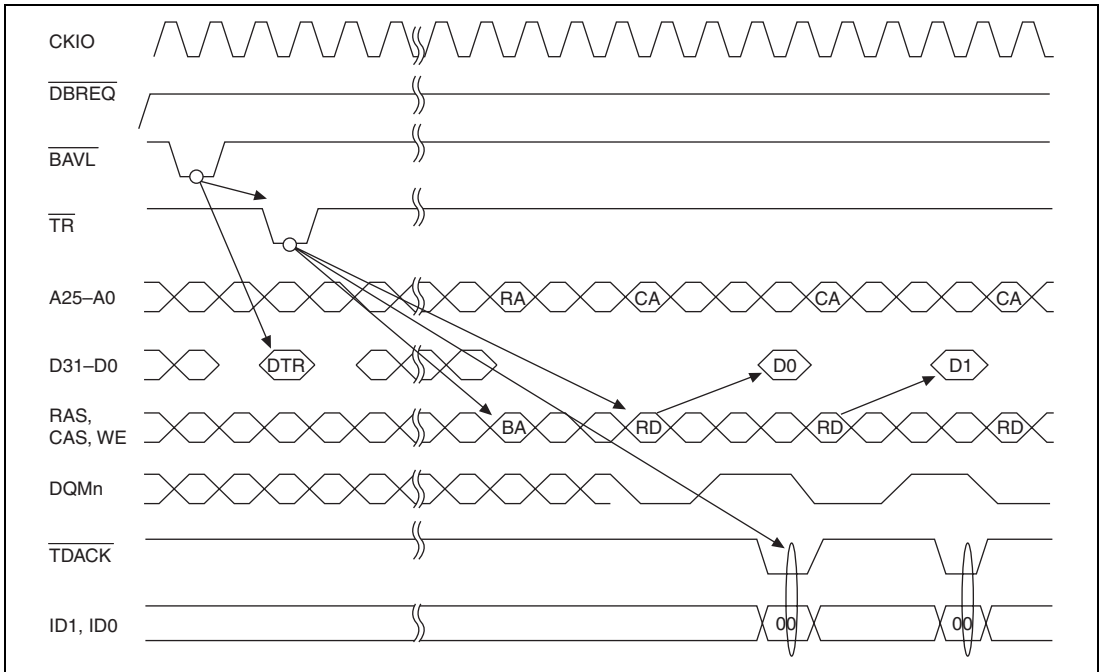


Figure 14.31 Single Address Mode/Burst Mode/External Bus → External Device 32-Bit Transfer/Channel 0 On-Demand Data Transfer

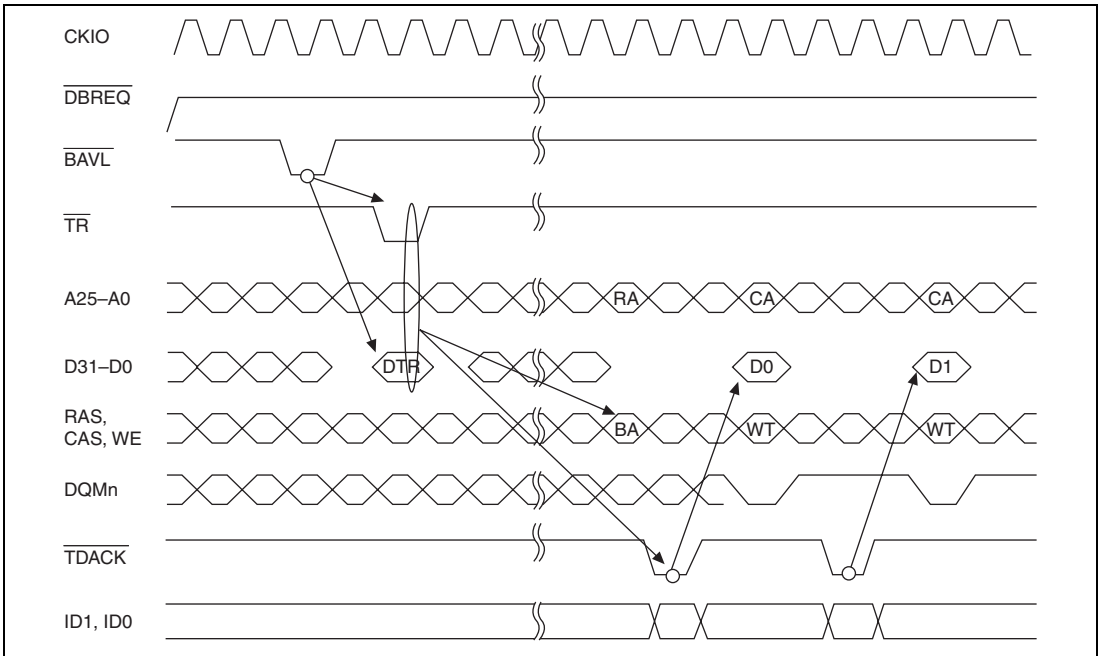
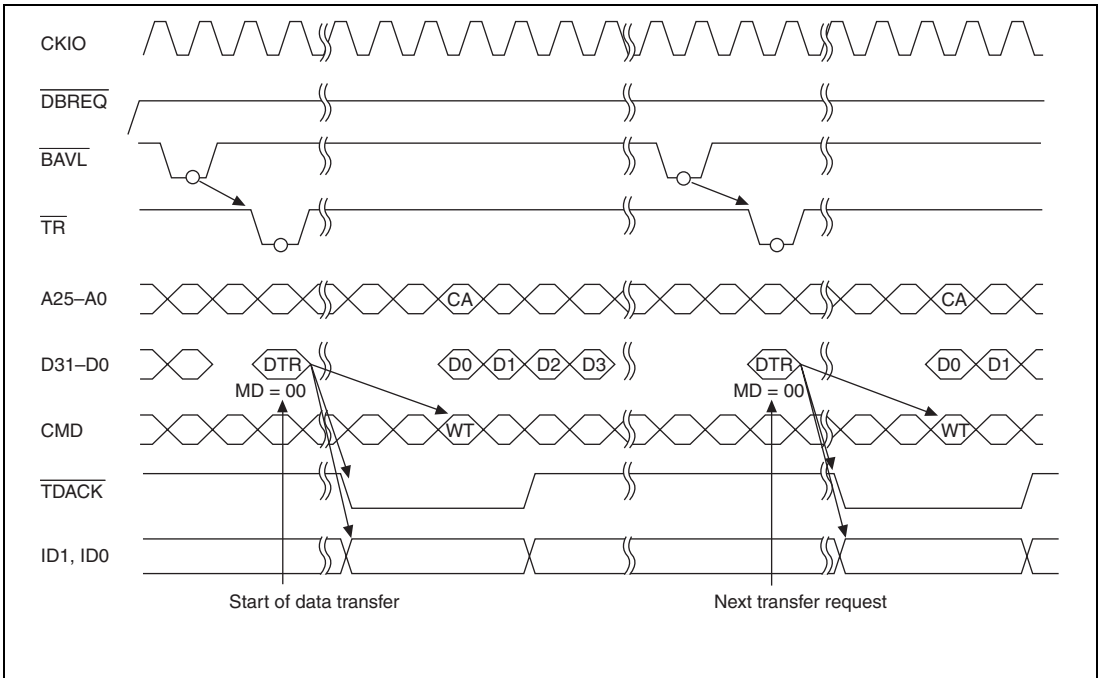
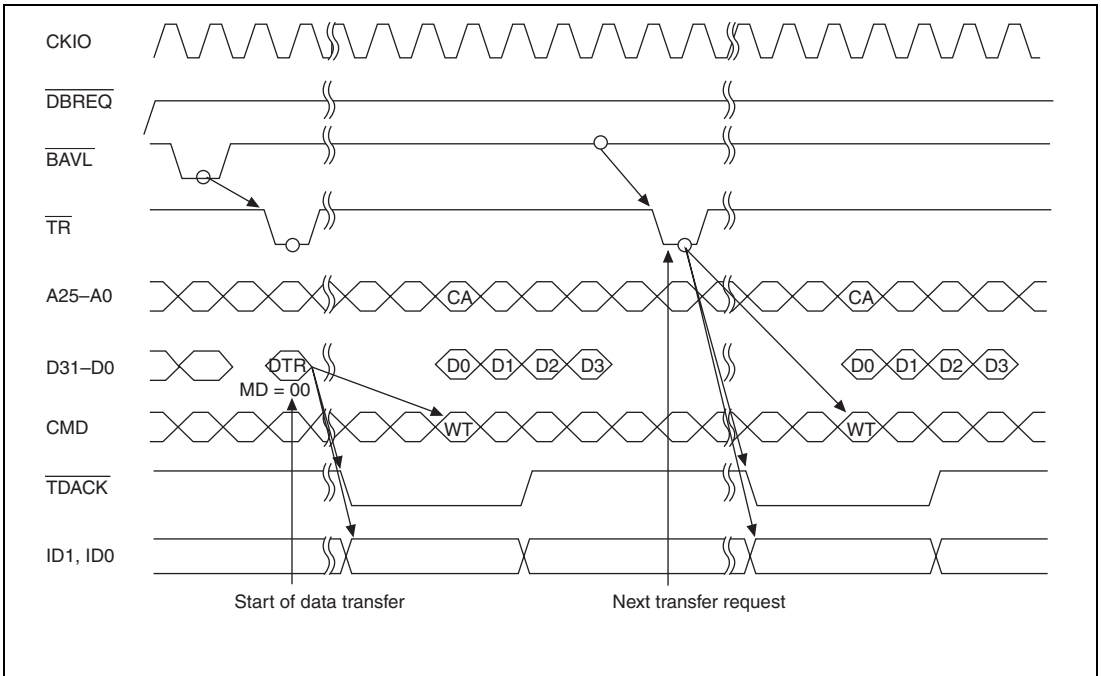


Figure 14.32 Single Address Mode/Burst Mode/External Device → External Bus 32-Bit Transfer/Channel 0 On-Demand Data Transfer



**Figure 14.33 Handshake Protocol Using Data Bus
(Channel 0 On-Demand Data Transfer)**



**Figure 14.34 Handshake Protocol without Use of Data Bus
(Channel 0 On-Demand Data Transfer)**

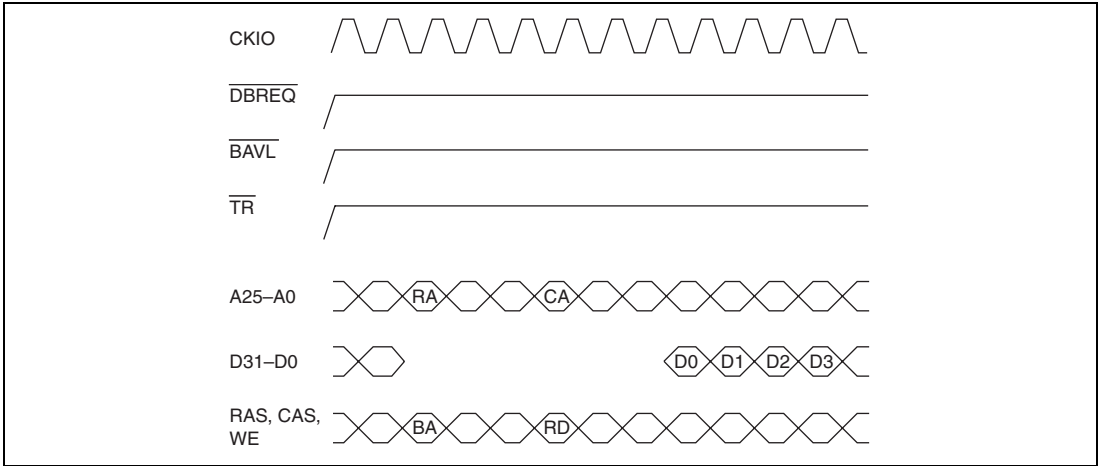


Figure 14.35 Read from Synchronous DRAM Precharge Bank

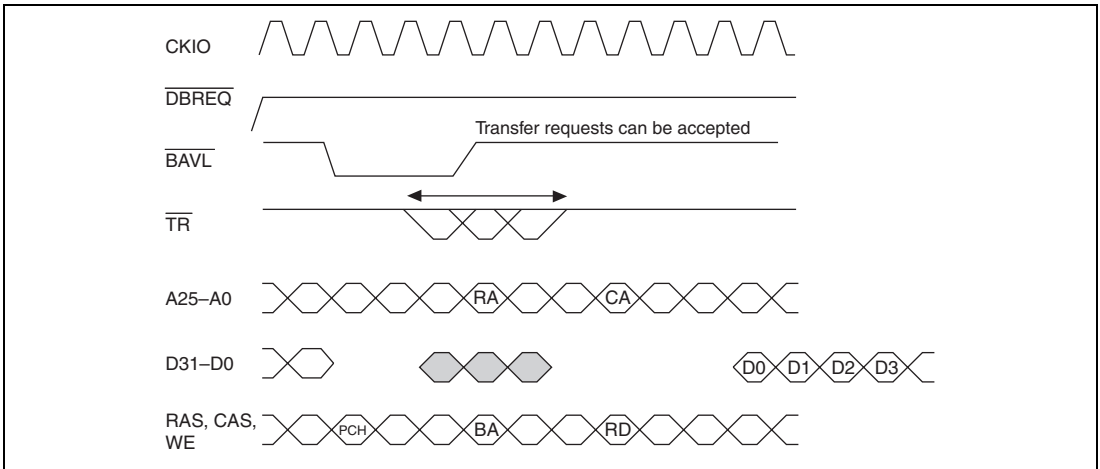


Figure 14.36 Read from Synchronous DRAM Non-Precharge Bank (Row Miss)

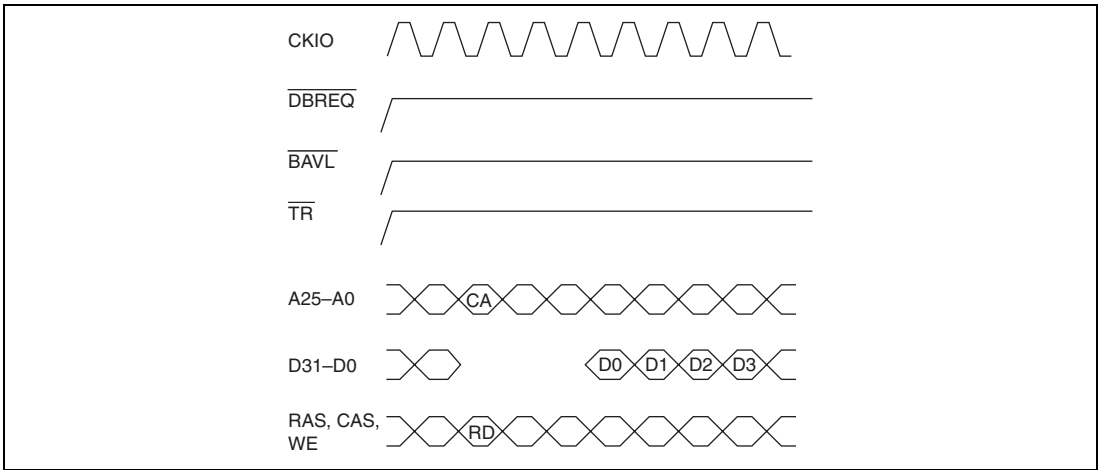


Figure 14.37 Read from Synchronous DRAM (Row Hit)

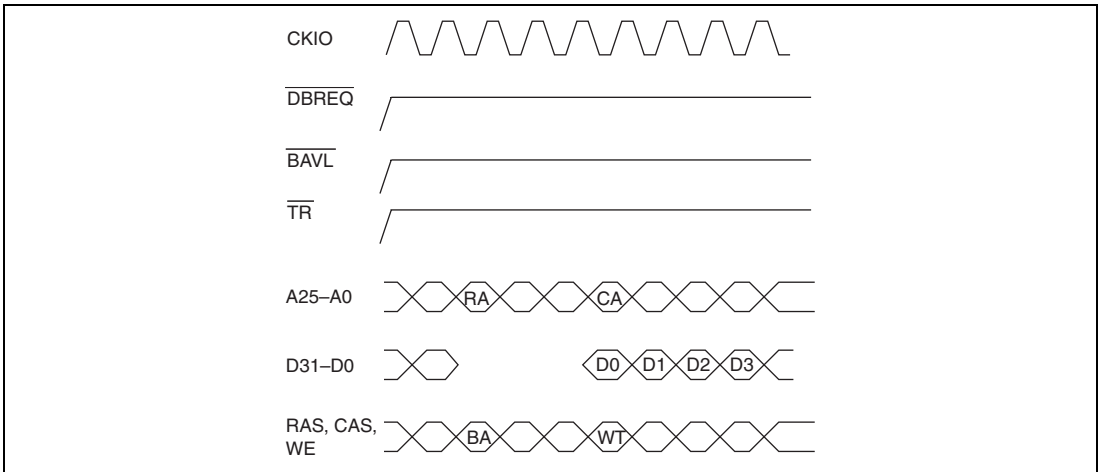


Figure 14.38 Write to Synchronous DRAM Precharge Bank

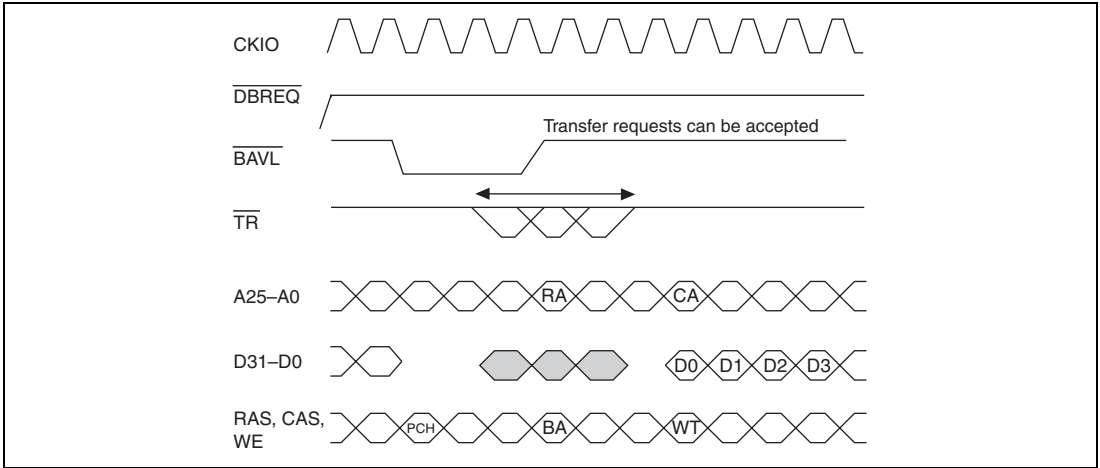


Figure 14.39 Write to Synchronous DRAM Non-Precharge Bank (Row Miss)

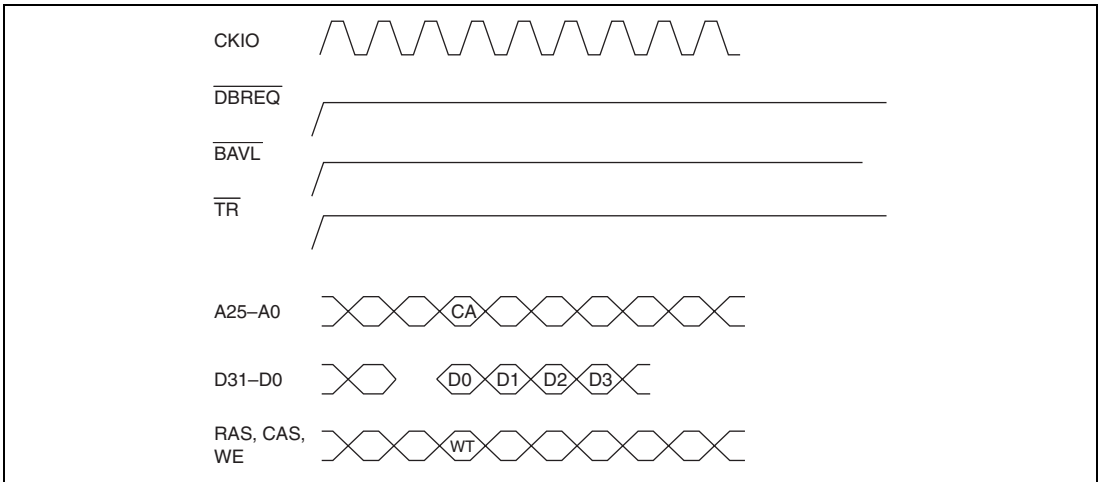


Figure 14.40 Write to Synchronous DRAM (Row Hit)

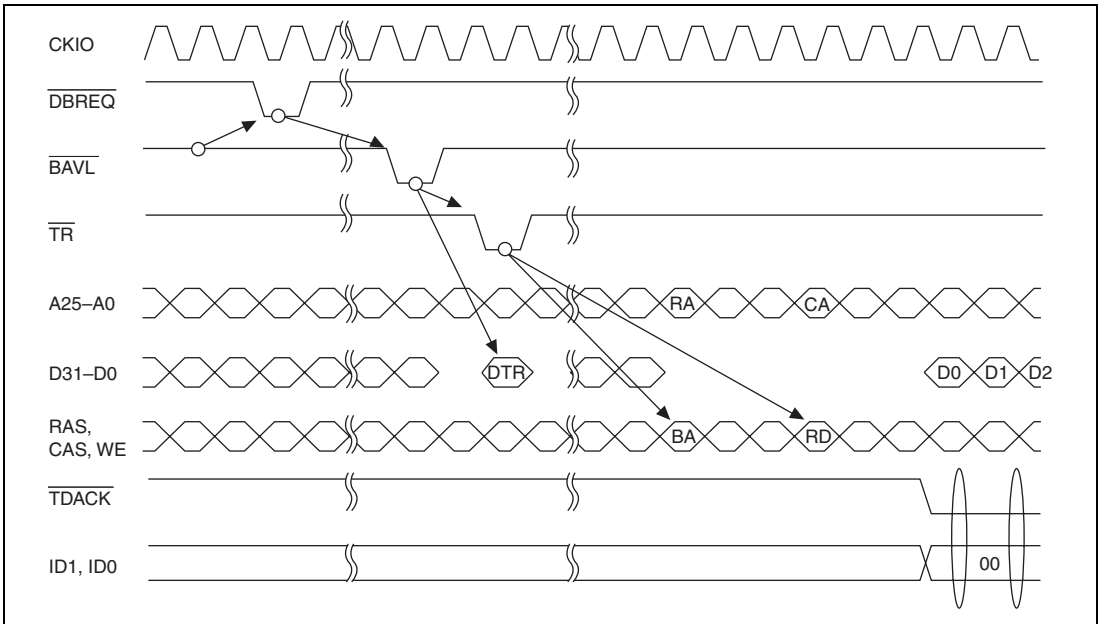
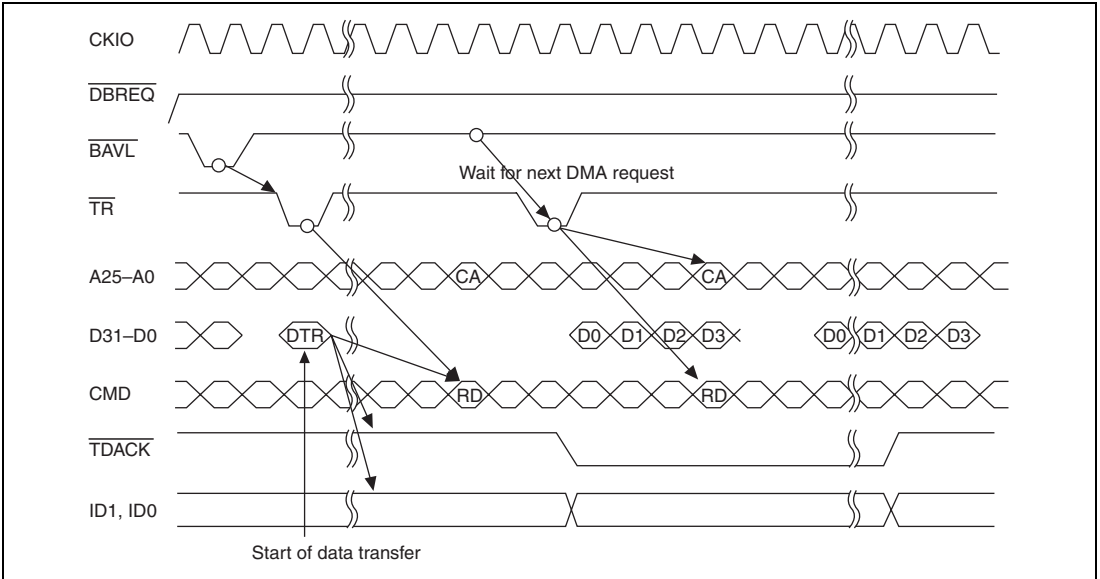
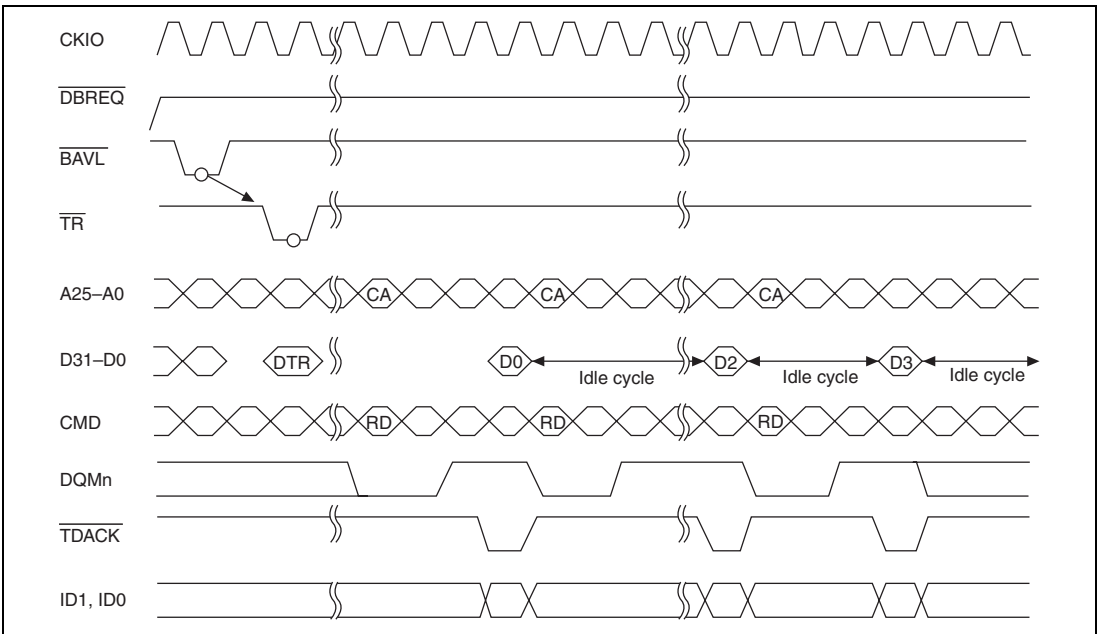


Figure 14.41 Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer



**Figure 14.44 Single Address Mode/Burst Mode/Level Detection/
External Bus → External Device Data Transfer**



**Figure 14.45 Single Address Mode/Burst Mode/Edge Detection/Byte, Word, Longword,
Quadword/External Bus → External Device Data Transfer**

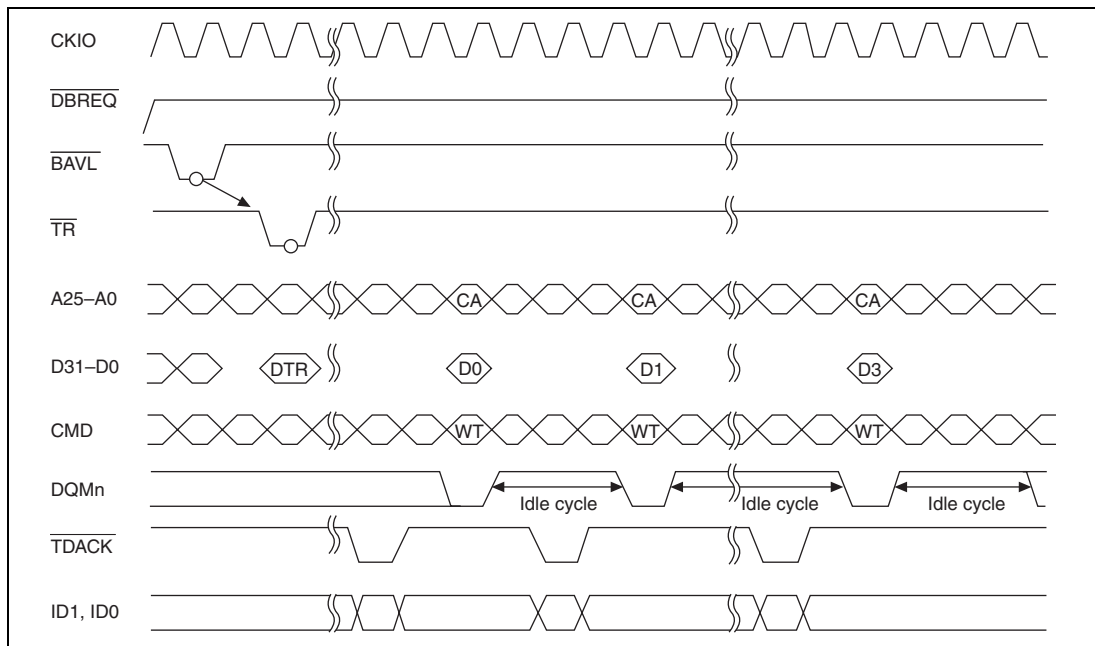


Figure 14.46 Single Address Mode/Burst Mode/Edge Detection/Byte, Word, Longword, Quadword/External Device → External Bus Data Transfer

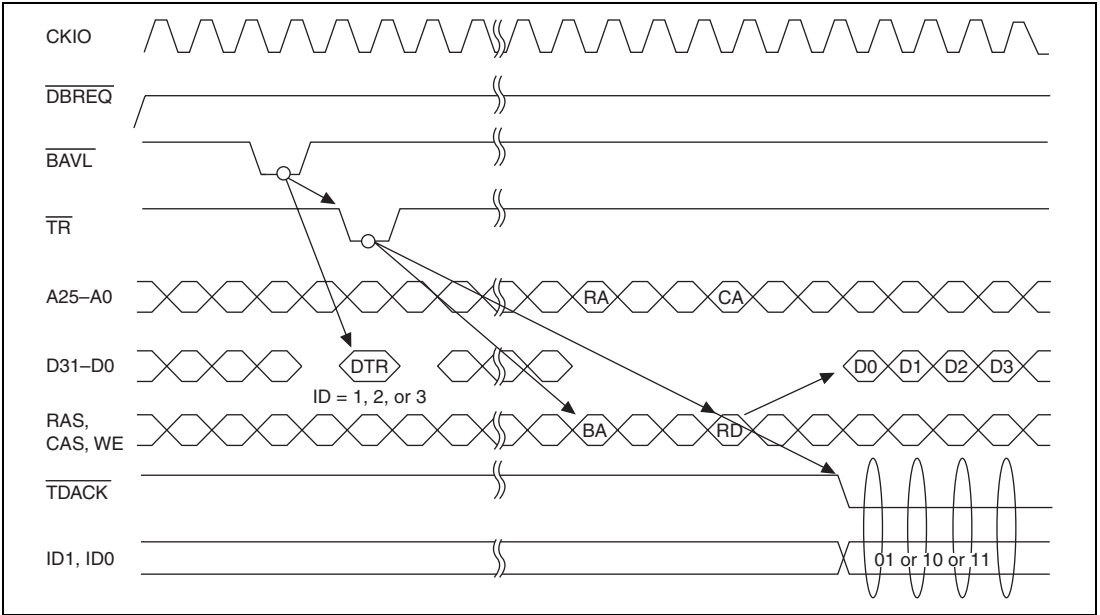
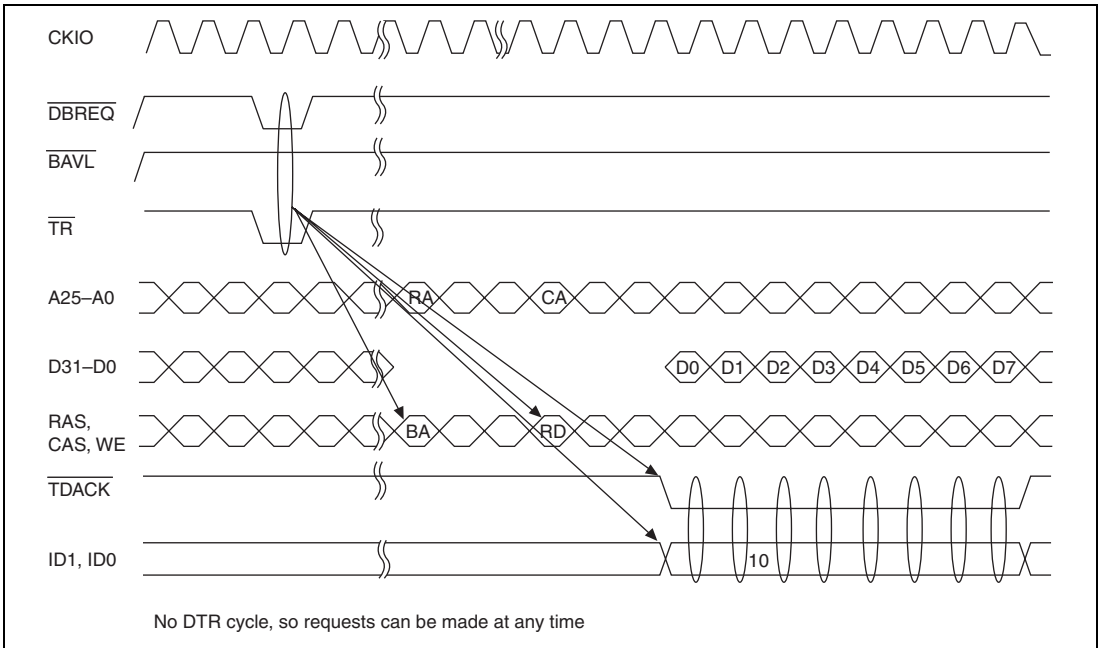


Figure 14.47 Single Address Mode/Burst Mode/32-Byte Block Transfer/DMA Transfer Request to Channels 1-3 Using Data Bus



**Figure 14.48 Single Address Mode/Burst Mode/32-Byte Block Transfer/
External Bus → External Device Data Transfer/
Direct Data Transfer Request to Channel 2 without Using Data Bus**

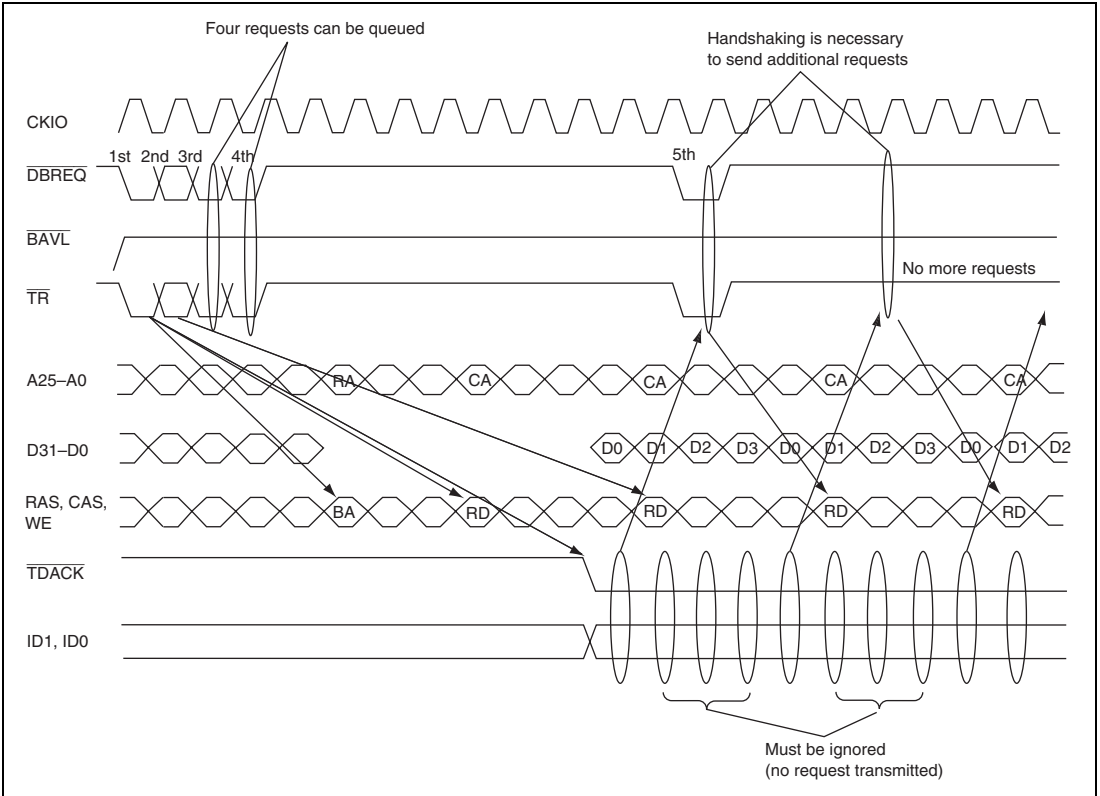


Figure 14.49 Single Address Mode/Burst Mode/External Bus → External Device Data Transfer/Direct Data Transfer Request to Channel 2

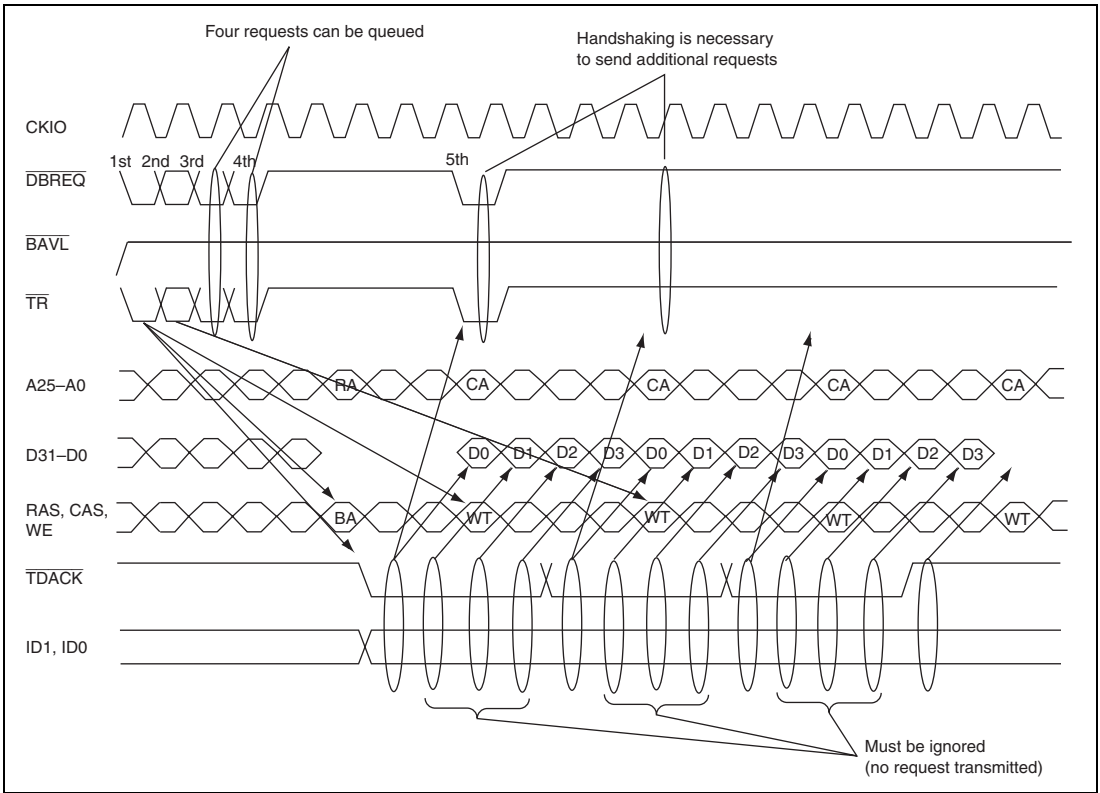


Figure 14.50 Single Address Mode/Burst Mode/External Device → External Bus Data Transfer/Direct Data Transfer Request to Channel 2

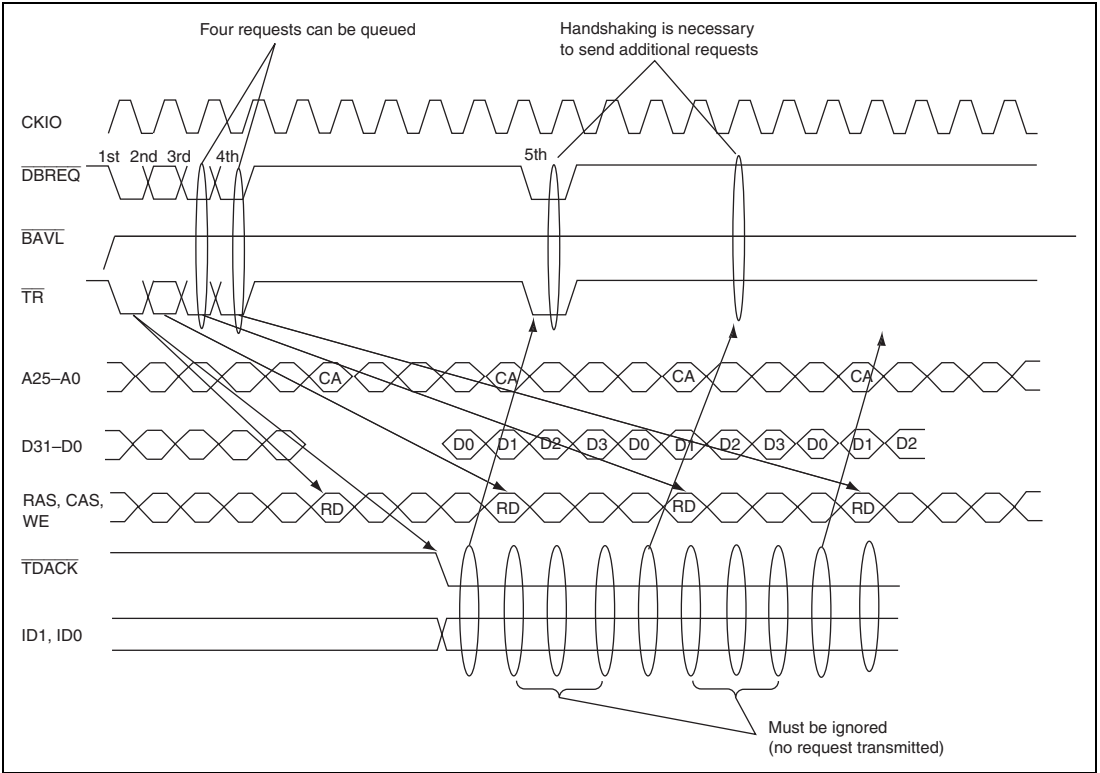


Figure 14.51 Single Address Mode/Burst Mode/External Bus → External Device Data Transfer (Active Bank Address)/Direct Data Transfer Request to Channel 2

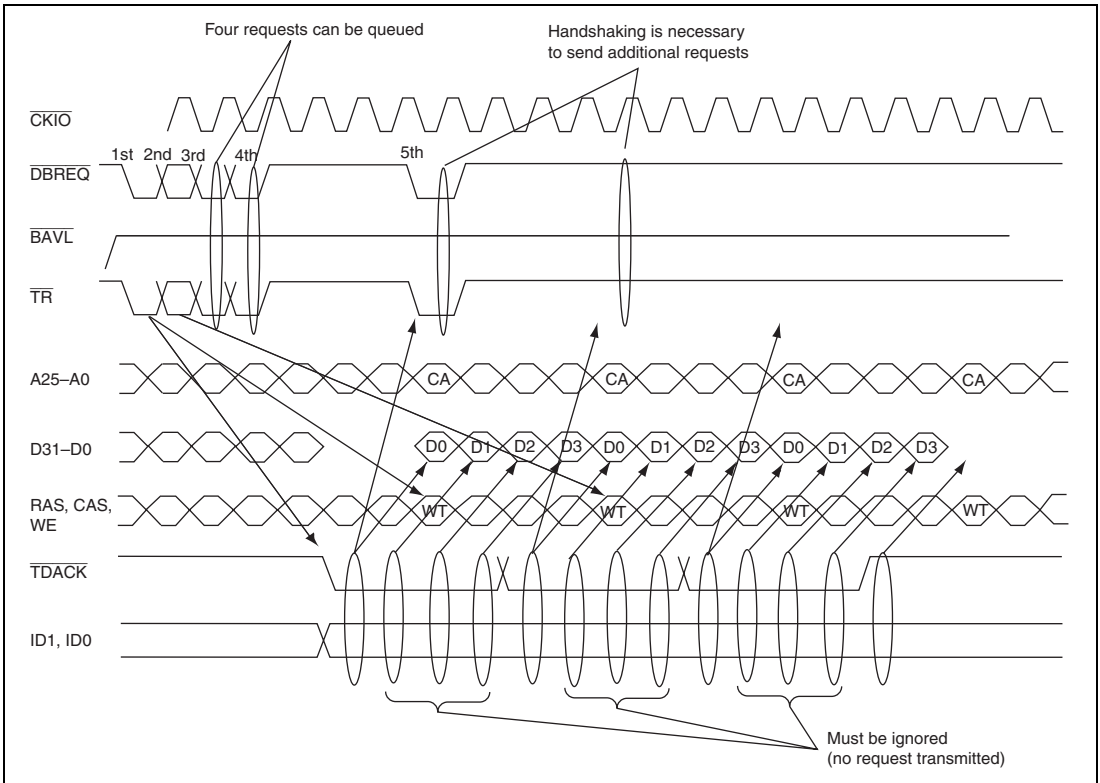


Figure 14.52 Single Address Mode/Burst Mode/External Device → External Bus Data Transfer (Active Bank Address)/Direct Data Transfer Request to Channel 2

14.5.4 Notes on Use of DDT Module

1. Normal data transfer mode (channel 0)

Set DTR.ID = 00 and DTR.MD = 00. If a setting of MD = 01, 10, or 11 is made, the DMAC will halt with an address error. In this case, the error can be cleared by reading DMAOR.AE = 1, then writing AE = 0.

2. Normal data transfer mode (channel 1 to channel 3)

If a setting of DTR.ID = 01, 10, or 11 is made, DTR.MD will be ignored.

3. Handshake protocol using the data bus (valid on channel 0 only)

- a. The handshake protocol using the data bus can be executed only on channel 0. (The DTR format must be set to DTR.ID = 00, DTR.MD = 00, and DTR.SZ ≠ 101, 110. Operation is not guaranteed if the DTR format data settings are DTR.ID = 00, DTR.MD = 00, and DTR.SZ ≠ 101, 110.)
- b. If, during execution of the handshake protocol using the data bus for channel 0, a request is input for one of channels 1 to 3, and after that DMA transfer is executed settings of DTR.ID = 00, DTR.MD = 00, and DTR.SZ ≠ 101, 110 are input in the handshake protocol using the data bus, a transfer request will be asserted for channel 0.
- c. If \overline{TR} only is asserted by means of the handshake protocol without use of the data bus and a DMA transfer request is input when channel 0 DMA transfer has ended and CHCR0.TE = 1, the DMAC will freeze. Before issuing a DMA transfer request, the TE flag must be cleared by writing CHCR0.TE = 0 after reading CHCR0.TE = 1.

4. Handshake protocol without use of the data bus

- a. With the handshake protocol without use of the data bus, a DMA transfer request can be input to the DMAC again for the channel for which transfer was requested immediately before by asserting \overline{TR} only.
- b. When using the handshake protocol without use of the data bus, first make the necessary settings in the DMAC control registers.
- c. When not using the handshake protocol without use of the data bus, if \overline{TR} only is asserted without outputting DTR, a request will be issued for the channel for which DMA transfer was requested immediately before. Also, if the first DMA transfer request after a power-on reset is input by asserting \overline{TR} only, it will be ignored and the DMAC will not operate.

5. Direct data transfer mode (valid on channel 2 only)

- a. If a DMA transfer request for channel 2 is input by simultaneous assertion of \overline{DBREQ} and \overline{TR} during DMA transfer execution with the handshake protocol without use of the data bus, it will be accepted if there is space in the DDT channel 2 request queue.
- b. In direct data transfer mode (with \overline{DBREQ} and \overline{TR} asserted simultaneously), \overline{DBREQ} is not interpreted as a bus arbitration signal, and therefore the \overline{BAVL} signal is never asserted.

6. Request queue transfer request acceptance
 - a. The DDT has four request queues for each of channels 1 to 3. When these request queues are full, a DMA transfer request from an external device will be ignored.
 - b. If a DMA transfer request for channel 0 is input during execution of a channel 0 DMA bus cycle, the DDT will ignore that request. Confirm that channel 0 DMA transfer has finished (burst mode) or that a DMA bus cycle is not in progress (cycle steal mode).
7. DTR format
 - a. The DDT module processes DTR.ID, DTR.MD, and DTR.SZ as follows.

When DTR.ID = 00

 - MD = 00, SZ ≠ 101, 110: Handshake protocol using the data bus
 - MD ≠ 00, SZ = 111: CHCR0.DE = 0 setting (DMA transfer end request)
 - MD = 10, SZ = 110: DDT request queue clear

When DTR.ID ≠ 00

 - Transfer request to channels 1—3 (items other than ID ignored)

Note: Do not use setting values other than the above.
8. Data transfer end request
 - a. A data transfer end request (DTR.ID = 00, MD ≠ 00, SZ = 111) cannot be accepted during channel 0 DMA transfer. Therefore, if edge detection and burst mode are set for channel 0, transfer cannot be ended midway.
 - b. When a transfer end request (DTR.ID = 00, MD ≠ 00, SZ = 111) is accepted, the values set in CHCR0, SAR0, DAR0, and DMATCR0 are retained. In this case, execution cannot be restarted from an external device. To restart execution, set CHCR0.DE = 1 with an MOV instruction.
9. Request queue clearance
 - a. When settings of DTR.ID = 00, DTR.MD = 10, and SZ = 110 are accepted by the DDT in normal data transfer mode, DDT channel 0 requests and channel 1 to 3 request queues are all cleared. All external requests held on the DMAC side are also cleared.
 - b. In case 3-c, the DMAC freeze state can be cleared.
 - c. When settings of DMAOR.DDT = 1, DTR.ID = 00, DTR.MD = 10, and SZ = 110 are accepted by the DDT in case 11, the DMAC freeze state can be cleared.
10. $\overline{\text{DBREQ}}$ assertion
 - a. After $\overline{\text{DBREQ}}$ is asserted, do not assert $\overline{\text{DBREQ}}$ again until $\overline{\text{BAVL}}$ is asserted, as this will result in a discrepancy between the number of $\overline{\text{DBREQ}}$ and $\overline{\text{BAVL}}$ assertions.
 - b. The $\overline{\text{BAVL}}$ assertion period due to $\overline{\text{DBREQ}}$ assertion is one cycle.

If a row address miss occurs in a read or write in the non-precharged bank during synchronous DRAM access, $\overline{\text{BAVL}}$ is asserted for a number of cycles in accordance with the RAS precharge interval set in BSC.MCR.TCP.

- c. It takes one cycle for $\overline{\text{DBREQ}}$ to be accepted by the DMAC after being asserted by an external device. If a row address miss occurs at this time in a read or write in the non-precharged bank during synchronous DRAM access, and $\overline{\text{BAVL}}$ is asserted, the $\overline{\text{DBREQ}}$ signal asserted by the external device is ignored. Therefore, $\overline{\text{BAVL}}$ is not asserted again due to this signal.

11. Clearing DDT mode

Check that DMA transfer is not in progress on any channel before setting the DMAOR.DDT bit. If the DMAOR.DDT setting is changed from 1 to 0 during DMA transfer in DDT mode, the DMAC will freeze.

This also applies when switching from normal DMA mode (DMAOR.DDT = 0) to DDT mode.

12. Confirming DMA transfer requests and number of transfers executed

The channel associated with a DMA bus cycle being executed in response to a DMA transfer request can be confirmed by determining the level of external pins ID1 and ID0 at the rising edge of the CKIO clock while $\overline{\text{TDACK}}$ is asserted.

(ID = 00: channel 0; ID = 01: channel 1; ID = 10: channel 2; ID = 11: channel 3)

14.6 Configuration of the DMAC (SH7751R)

14.6.1 Block Diagram of the DMAC

Figure 14.53 is a block diagram of the DMAC in the SH7751R.

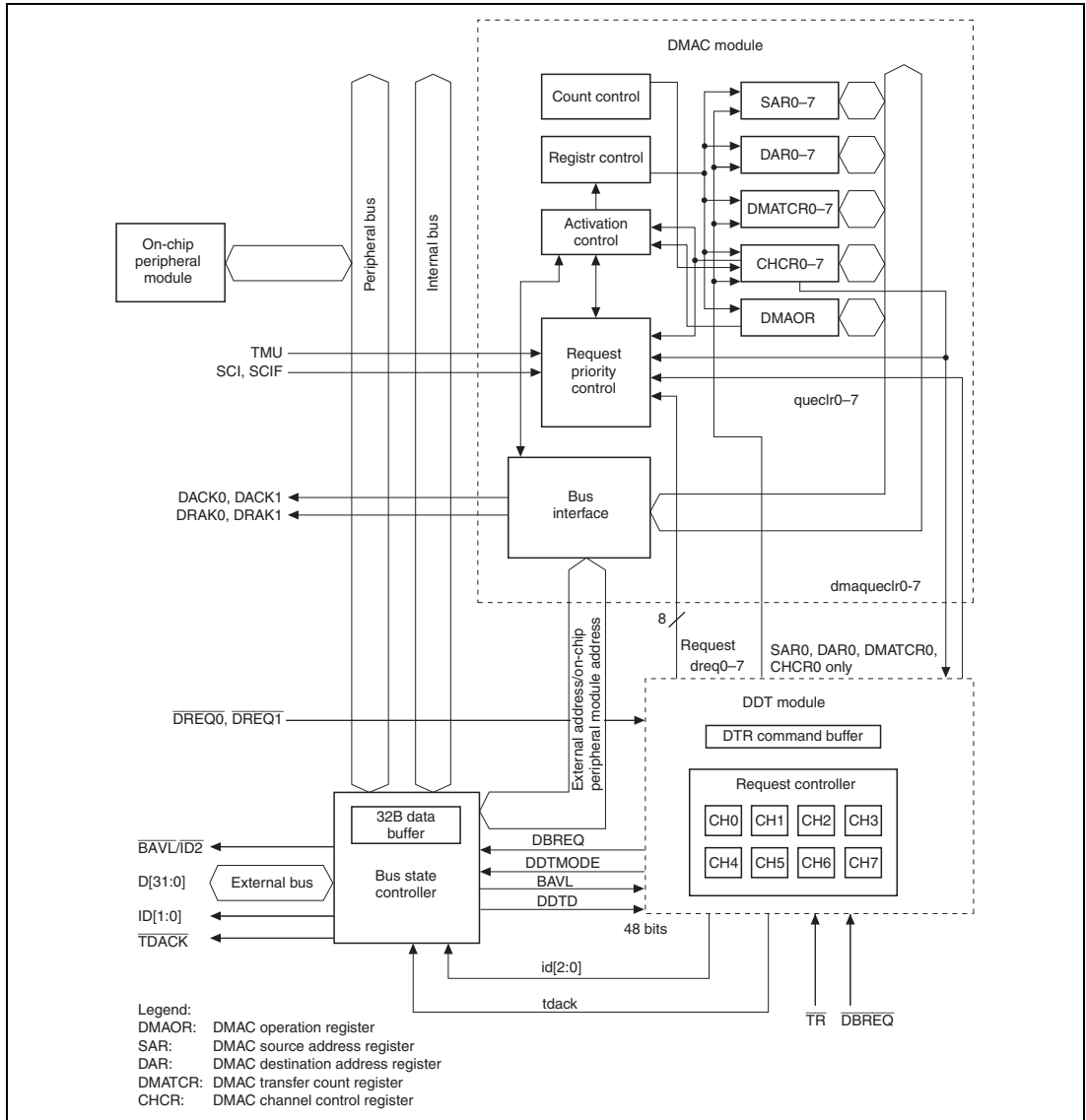


Figure 14.53 Block Diagram of the DMAC

14.6.2 Pin Configuration (SH7751R)

Tables 14.12 and 14.13 show the pin configuration of the DMAC.

Table 14.12 DMAC Pins

| Channel | Pin Name | Abbreviation | I/O | Function |
|---------|--|---------------------------|--------|---|
| 0 | DMA transfer request | $\overline{\text{DREQ0}}$ | Input | DMA transfer request input from external device to channel 0 |
| | $\overline{\text{DREQ}}$ acceptance confirmation | DRAK0 | Output | Acceptance of request for DMA transfer from channel 0 to external device Notification to external device of start of execution |
| | DMA transfer end notification | DACK0 | Output | Strobe output to external device of DMA transfer request from channel 0 to external device |
| 1 | DMA transfer request | $\overline{\text{DREQ1}}$ | Input | DMA transfer request input from external device to channel 1 |
| | $\overline{\text{DREQ}}$ acceptance confirmation | DRAK1 | Output | Acceptance of request for DMA transfer from channel 1 to external device Notification to external device of start of execution |
| | DMA transfer end notification | DACK1 | Output | Strobe output to external device of DMA transfer request from channel 1 to external device |

Table 14.13 DMAC Pins in DDT Mode

| Pin Name | Abbreviation | I/O | Function |
|-----------------------------|---|--------|---|
| Data bus request | $\overline{\text{DBREQ}}$ (DREQ0) | Input | Data bus release request from external device for DTR format input |
| Data bus available | $\overline{\text{BAVL}}/\overline{\text{ID2}}$ (DRAK0) | Output | Data bus release notification Data bus can be used 2 cycles after BAVL is asserted Notification of channel number to external device at same time as $\overline{\text{TDACK}}$ output |
| Transfer request signal | $\overline{\text{TR}}$ (DREQ1) | Input | If asserted 2 cycles after $\overline{\text{BAVL}}$ assertion, DTR format is sent Only $\overline{\text{TR}}$ asserted: DMA request $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ asserted simultaneously: Direct request to channel 2 |
| DMAC strobe | $\overline{\text{TDACK}}$ (DACK0) | Output | Reply strobe signal for external device from DMAC |
| Channel number notification | ID[1:0] (DRAK1, DACK1) | Output | Notification of channel number to external device at same time as $\overline{\text{TDACK}}$ output (ID [1] = DRAK1, ID [0] = DACK1) |

Requests for DMA transfer from external devices are normally accepted only on channel 0 ($\overline{\text{DREQ0}}$) and channel 1 ($\overline{\text{DREQ1}}$). In DDT mode, the BAVL pin functions as both the data-bus-available pin and channel-number-notification ($\overline{\text{ID2}}$) pin.

14.6.3 Register Configuration (SH7751R)

Table 14.14 shows the configuration of the DMAC's registers. The DMAC of the SH7751R has a total of 33 registers: four registers are assigned to each channel, and there is a control register for the overall control of the DMAC.

Table 14.14 Register Configuration

| Channel | Name | Abbreviation | Read/Write | Initial Value | P4 Address | Area 7 Address | Access Size |
|---------|------------------------------------|--------------|------------|---------------|-------------|----------------|-------------|
| 0 | DMA source address register 0 | SAR0 | R/W | Undefined | H'FFFA00000 | H'1FA00000 | 32 |
| | DMA destination address register 0 | DAR0 | R/W | Undefined | H'FFFA00004 | H'1FA00004 | 32 |
| | DMA transfer count register 0 | DMATCR0 | R/W | Undefined | H'FFFA00008 | H'1FA00008 | 32 |
| | DMA channel control register 0 | CHCR0 | R/W* | H'00000000 | H'FFFA0000C | H'1FA0000C | 32 |
| 1 | DMA source address register 1 | SAR1 | R/W | Undefined | H'FFFA00010 | H'1FA00010 | 32 |
| | DMA destination address register 1 | DAR1 | R/W | Undefined | H'FFFA00014 | H'1FA00014 | 32 |
| | DMA transfer count register 1 | DMATCR1 | R/W | Undefined | H'FFFA00018 | H'1FA00018 | 32 |
| | DMA channel control register 1 | CHCR1 | R/W* | H'00000000 | H'FFFA0001C | H'1FA0001C | 32 |
| 2 | DMA source address register 2 | SAR2 | R/W | Undefined | H'FFFA00020 | H'1FA00020 | 32 |
| | DMA destination address register 2 | DAR2 | R/W | Undefined | H'FFFA00024 | H'1FA00024 | 32 |
| | DMA transfer count register 2 | DMATCR2 | R/W | Undefined | H'FFFA00028 | H'1FA00028 | 32 |
| | DMA channel control register 2 | CHCR2 | R/W* | H'00000000 | H'FFFA0002C | H'1FA0002C | 32 |
| 3 | DMA source address register 3 | SAR3 | R/W | Undefined | H'FFFA00030 | H'1FA00030 | 32 |
| | DMA destination address register 3 | DAR3 | R/W | Undefined | H'FFFA00034 | H'1FA00034 | 32 |
| | DMA transfer count register 3 | DMATCR3 | R/W | Undefined | H'FFFA00038 | H'1FA00038 | 32 |
| | DMA channel control register 3 | CHCR3 | R/W* | H'00000000 | H'FFFA0003C | H'1FA0003C | 32 |
| Common | DMA operation register | DMAOR | R/W* | H'00000000 | H'FFFA00040 | H'1FA00040 | 32 |

| Channel | Name | Abbreviation | Read/Write | Initial Value | P4 Address | Area 7 Address | Access Size |
|---------|------------------------------------|--------------|------------|---------------|-------------|----------------|-------------|
| 4 | DMA source address register 4 | SAR4 | R/W | Undefined | H'FFFA00050 | H'1FA00050 | 32 |
| | DMA destination address register 4 | DAR4 | R/W | Undefined | H'FFFA00054 | H'1FA00054 | 32 |
| | DMA transfer count register 4 | DMATCR4 | R/W | Undefined | H'FFFA00058 | H'1FA00058 | 32 |
| | DMA channel control register 4 | CHCR4 | R/W* | H'00000000 | H'FFFA0005C | H'1FA0005C | 32 |
| 5 | DMA source address register 5 | SAR5 | R/W | Undefined | H'FFFA00060 | H'1FA00060 | 32 |
| | DMA destination address register 5 | DAR5 | R/W | Undefined | H'FFFA00064 | H'1FA00064 | 32 |
| | DMA transfer count register 5 | DMATCR5 | R/W | Undefined | H'FFFA00068 | H'1FA00068 | 32 |
| | DMA channel control register 5 | CHCR5 | R/W* | H'00000000 | H'FFFA0006C | H'1FA0006C | 32 |
| 6 | DMA source address register 6 | SAR6 | R/W | Undefined | H'FFFA00070 | H'1FA00070 | 32 |
| | DMA destination address register 6 | DAR6 | R/W | Undefined | H'FFFA00074 | H'1FA00074 | 32 |
| | DMA transfer count register 6 | DMATCR6 | R/W | Undefined | H'FFFA00078 | H'1FA00078 | 32 |
| | DMA channel control register 6 | CHCR6 | R/W* | H'00000000 | H'FFFA0007C | H'1FA0007C | 32 |
| 7 | DMA source address register 7 | SAR7 | R/W | Undefined | H'FFFA00080 | H'1FA00080 | 32 |
| | DMA destination address register 7 | DAR7 | R/W | Undefined | H'FFFA00084 | H'1FA00084 | 32 |
| | DMA transfer count register 7 | DMATCR7 | R/W | Undefined | H'FFFA00088 | H'1FA00088 | 32 |
| | DMA channel control register 7 | CHCR7 | R/W* | H'00000000 | H'FFFA0008C | H'1FA0008C | 32 |

Notes: Longword access should be used for all control registers. If a different access width is used, reads will return all 0s and writes will not be possible.

* Bit 1 of CHCR0–CHCR7 and bits 2 and 1 of DMAOR can only be written with 0 after being read as 1, to clear the flags.

14.7 Register Descriptions (SH7751R)

14.7.1 DMA Source Address Registers 0–7 (SAR0–SAR7)

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMA source address registers 0–7 (SAR0–SAR7) are 32-bit readable/writable registers that specify the source address for a DMA transfer. The functions of these registers are the same as on the SH7751. For more information, see section 14.2.1, DMA Source Address Registers 0–3 (SAR0–SAR3).

14.7.2 DMA Destination Address Registers 0–7 (DAR0–DAR7)

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMA destination address registers 0–7 (DAR0–DAR7) are 32-bit readable/writable registers that specify the destination address for a DMA transfer. The functions of these registers are the same as on the SH7751. For more information, see section 14.2.2, DMA Destination Address Registers 0–3 (DAR0–DAR3).

14.7.3 DMA Transfer Count Registers 0–7 (DMATCR0–DMATCR7)

| | | | | | | | | | | | | | | | | |
|----------------|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | | | | | | | | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | — | — | — | — | — | — | — | — |
| R/W: | R | R | R | R | R | R | R | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

DMA transfer count registers 0–7 (DMATCR0–DMATCR7) are 32-bit readable/writable registers that specify the number of transfers in transfer operations for the corresponding channel (bytecount, word count, longword count, quadword count, or 32-byte count). Functions of these registers are the same as the transfer-count registers of the SH7751. For more information, see section 14.2.3, DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3).

14.7.4 DMA Channel Control Registers 0–7 (CHCR0–CHCR7)

| | | | | | | | | | | | | | | | | |
|----------------|------|------|------|-----|------|------|------|-----|----|----|----|----|-----|-------|-----|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | SSA2 | SSA1 | SSA0 | STC | DSA2 | DSA1 | DSA0 | DTC | — | — | — | — | DS | RL | AM | AL |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R | R | R | R | R/W | (R/W) | R/W | (R/W) |

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-------|-----|-------|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DM1 | DM0 | SM1 | SM0 | RS3 | RS2 | RS1 | RS0 | TM | TS2 | TS1 | TS0 | QCL | IE | TE | DE |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | (R/W) | R/W | (R/W) | R/W |

DMA channel control registers 0–7 (CHCR0–CHCR7) are 32-bit readable/writable registers that specify the operating mode, transfer method, etc., for each channel. Bits 31–28 and 27–24 correspond to the source address and destination address, respectively; these settings are only valid when the transfer involves the CS5 or CS6 space and the relevant space has been specified as a PCMCIA-interface space. In other cases, these bits should be cleared to 0. For more information about the PCMCIA interface, see section 13.3.7, PCMCIA Interface.

No function is assigned to bits 18 and 16 of the CHCR2–CHCR7 registers. Writing to these bits of the CHCR2–CHCR7 registers is invalid. If, however, a value is written to these bits, it should always be 0. These bits are always read as 0.

These registers are initialized to H'00000000 by a power-on or manual reset. Their values are retained in standby, sleep, and deep-sleep modes.

Bits 31 to 29—Source Address Space Attribute Specification (SSA2–SSA0): These bits specify the space attribute for PCMCIA access. These bits are only valid in the case of page mapping to PCMCIA connected to areas 5 and 6. For details of the settings, see the description of the SSA2–SSA0 bits in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 28—Source Address Wait Control Select (STC): Specifies CS5 or CS6 space wait control for PCMCIA access. This bit selects the wait control register in the BSC that performs area 5 and 6 wait cycle control. For details of the settings, see the description of the STC bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bits 27 to 25—Destination Address Space Attribute Specification (DSA2–DSA0): These bits specify the space attribute for PCMCIA access. These bits are only valid in the case of page mapping to PCMCIA connected to areas 5 and 6. For details of the settings, see the description of the DSA2–DSA0 bits in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 24—Destination Address Wait Control Select (DTC): Specifies CS5 or CS6 space wait cycle control for PCMCIA access. This bit selects the wait control register in the BSC that performs area 5 and 6 wait cycle control. For details of the settings, see the description of the DTC bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bits 23 to 20—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 19— $\overline{\text{DREQ}}$ Select (DS): Specifies either low level detection or falling edge detection as the sampling method for the $\overline{\text{DREQ}}$ pin used in external request mode.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. In DDT mode, it is valid in CHCR0–CHCR7. For details of the settings, see the description of the DS bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 18—Request Check Level (RL): Selects whether the DRAK signal (that notifies an external device of the acceptance of $\overline{\text{DREQ}}$) is an active-high or active-low output.

This bit is valid only in CHCR0 and CHCR1 in normal mode, and is invalid in DDT mode. For details of the settings, see the description of the RL bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 17—Acknowledge Mode (AM): In dual address mode, selects whether DACK is output in the data read cycle or write cycle. In single address mode, DACK is always output regardless of the setting of this bit.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. In DDT mode, it is valid in CHCR0–CHCR7. (DDT mode: $\overline{\text{TDACK}}$) For details of the settings, see the description of the AM bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 16—Acknowledge Level (AL): Specifies the DACK (acknowledge) signal as active-high or active-low.

This bit is valid only in CHCR0 and CHCR1 in normal mode, and is invalid in DDT mode. For details of the settings, see the description of the AL bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bits 15 and 14—Destination Address Mode 1 and 0 (DM1, DM0): These bits specify incrementing/decrementing of the DMA transfer destination address. The specification of these bits is ignored when data is transferred from external memory to an external device in single address mode. For details of the settings, see the description of the DM1 and DM0 bits in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bits 13 and 12—Source Address Mode 1 and 0 (SM1, SM0): These bits specify incrementing/decrementing of the DMA transfer source address. The specification of these bits is ignored when data is transferred from an external device to external memory in single address mode. For details of the settings, see the description of the SM1 and SM0 bits in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bits 11 to 8—Resource Select 3 to 0 (RS3–RS0): These bits specify the transfer request source. For details of the settings, see the description of the RS3–RS0 bits in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 7—Transmit Mode (TM): Specifies the bus mode for transfer. For details of the settings, see the description of the TM bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bits 6 to 4—Transmit Size 2 to 0 (TS2–TS0): These bits specify the transfer data size. For details of the settings, see the description of the TS2–TS0 bits in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 3—Request Queue Clear (QCL): Writing a 1 to this bit clears the request queues of the corresponding channel as well as any external requests that have already been accepted. This bit is only functional when DMAOR.DDT = 1 and DMAOR.DBL = 1.

CHCR Bit 3

| QCL | Description |
|-----|--|
| 0 | This bit is always read as 0. (Initial value) Writing a 0 to this bit is invalid. |
| 1 | When DMAOR.DBL = 1, writing a 1 to this bit clears the request queues on the DDT side and any external requests stored in the DMAC. The written value is not retained. |

Bit 2—Interrupt Enable (IE): When this bit is set to 1, an interrupt request (DMTE) is generated after the number of data transfers specified in DMATCR (when TE = 1). For details of the settings, see the description of the IE bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 1—Transfer End (TE): This bit is set to 1 after the number of transfers specified in DMATCR. If the IE bit is set to 1 at this time, an interrupt request (DMTE) is generated.

If data transfer ends before TE is set to 1 (for example, due to an NMI interrupt, address error, or clearing of the DE bit or the DME bit in DMAOR), the TE bit is not set to 1. When this bit is 1, the transfer enabled state is not entered even if the DE bit is set to 1. For details of the settings, see the description of the TE bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

Bit 0—DMAC Enable (DE): Enables operation of the corresponding channel. For details of the settings, see the description of the DE bit in section 14.2.4, DMA Channel Control Registers 0–3 (CHCR0–CHCR3).

14.7.5 DMA Operation Register (DMAOR)

| | | | | | | | | | | | | | | | | |
|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| | | | | | | | | | | | | | | | | |
|----------------|-----|-----|----|----|----|----|-----|-----|---|---|---|---|---|-------|-------|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DDT | DBL | — | — | — | — | PR1 | PR0 | — | — | — | — | — | AE | NMIF | DME |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R | R | R | R | R/W | R/W | R | R | R | R | R | R/(W) | R/(W) | R/W |

DMAOR is a 32-bit readable/writable register that specifies the DMAC transfer mode.

DMAOR is initialized to H'00000000 by a power-on or manual reset. They retain their values in standby mode and deep sleep mode.

Bits 31 to 16—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 15—On-Demand Data Transfer (DDT): Specifies on-demand data transfer mode. For details of the settings, see the description of the DDT bit in section 14.2.5, DMA Operation Register (DMAOR).

Bit 14—Number of DDT-Mode Channels (DBL): Selects the number of channels that are able to accept external requests in DDT mode.

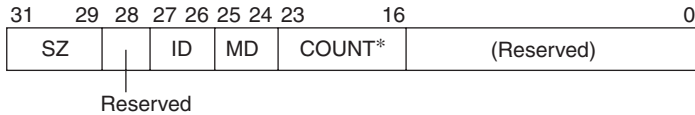
| Bit 14: DBL | Description |
|-------------|--|
| 0 | Four DDT-mode channels (Initial value) |
| 1 | Eight DDT-mode channels |

Note: When DMAOR.DBL = 0, channels 4 to 7 do not accept external requests.

When DMAOR.DBL = 1, one channel can be selected from among channels 0–7 by the combination of DTR.SZ and DTR.ID in the DTR format (see figure 14.54). Table 14.15 shows the channel selection by DTR format in the DDT mode.

Table 14.15 Channel Selection by DTR Format (DMAOR.DBL = 1)

| DTR.ID[1:0] | DTR.SZ[2:0] ≠ 101 | DTR.SZ[2:0] = 101 |
|-------------|-------------------|-------------------|
| 00 | CH0 | CH4 |
| 01 | CH1 | CH5 |
| 10 | CH2 | CH6 |
| 11 | CH3 | CH7 |



Note: * These bits are valid when request queue clear is specified (with no transfer count function).

Figure 14.54 DTR Format (Transfer Request Format) (SH7751R)

Bits 13 to 10—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 9 and 8—Priority Mode 1 and 0 (PR1, PR0): These bits determine the order of priority for channel execution when transfer requests are made for a number of channels simultaneously.

| DMAOR Bit 9 | DMAOR Bit 8 | Description |
|----------------|----------------|---|
| PR1 | PR0 | |
| 0 | 0 | CH0 > CH1 > CH2 > CH3 > CH4 > CH5 > CH6 > CH7 (Initial value) |
| 0 | 1 | CH0 > CH2 > CH3 > CH4 > CH5 > CH6 > CH7 > CH1 |
| 1 | 0 | CH2 > CH0 > CH1 > CH3 > CH4 > CH5 > CH6 > CH7 |
| 1 | 1 | Round robin mode |

Bits 7 to 3—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 2—Address Error Flag (AE): Indicates that an address error has occurred during DMA transfer. If this bit is set during data transfer, transfers on all channels are suspended, and an interrupt request (DMAE) is generated. The CPU cannot write 1 to AE. This bit can only be cleared by writing 0 after reading 1. For details of the settings, see the description of the AE bit in section 14.2.5, DMA Operation Register (DMAOR).

Bit 1—NMI Flag (NMIF): Indicates that NMI has been input. This bit is set regardless of whether or not the DMAC is operating. If this bit is set during data transfer, transfers on all channels are suspended. The CPU cannot write 1 to NMIF. This bit can only be cleared by writing 0 after reading 1. For details of the settings, see the description of the NMIF bit in section 14.2.5, DMA Operation Register (DMAOR).

Bit 0—DMAC Master Enable (DME): Enables activation of the entire DMAC. When the DME bit and the DE bit of the CHCR register for the corresponding channel are set to 1, that channel is enabled for transfer. If this bit is cleared during data transfer, transfers on all channels are suspended.

Even if the DME bit has been set, transfer is not enabled when TE is 1 or DE is 0 in CHCR, or when the NMI or AE bit in DMAOR is 1. For details of the settings, see the description of the DME bit in section 14.2.5, DMA Operation Register (DMAOR).

14.8 Operation (SH7751R)

Operation specific to the SH7751R is described here. For details of operation, see section 14.3, Operation.

14.8.1 Channel Specification for a Normal DMA Transfer

In normal DMA transfer mode, the DMAC always operates with eight channels, and external requests are only accepted on channel 0 ($\overline{\text{DREQ0}}$) and channel 1 ($\overline{\text{DREQ1}}$).

After setting the registers of the channels in use, including CHCR, SAR, DAR, and DMATCR, DMA transfer is started on receiving a DMA transfer request in the transfer-enabled state (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), in the order of predetermined priority. The transfer ends when the transfer-end condition is satisfied. There are three modes for transfer requests: auto-request, external request, and on-chip peripheral module request. The addressing modes for DMA transfer are the single-address mode and the dual-address mode. Bus mode is selectable between burst mode and cycle steal mode.

14.8.2 Channel Specification for DDT-Mode DMA Transfer

For DMA transfer in DDT mode, the DMAOR.DBL setting selects either four or eight channels. External requests are accepted on channels 0–3 when DMAOR.DBL = 0, and on channels 0–7 when DMAOR.DBL = 1. For further information on these settings, see the entry on the DBL bit in section 14.7.5, DMA Operation Register (DMAOR).

14.8.3 Transfer Channel Notification in DDT Mode

When the DMAC is set up for four-channel external request acceptance in DDT mode (DMAOR.DBL = 0), the ID [1:0] bits are used to notify the external device of the DMAC channel that is to be used. For more details, see section 14.5, On-Demand Data Transfer Mode (DDT Mode).

When the DMAC is set up for eight-channel external request acceptance in DDT mode (DMAOR.DBL = 1), the ID [1:0] bits and the simultaneous (on the timing of $\overline{\text{TDACK}}$ assertion) assertion of $\overline{\text{ID2}}$ from the $\overline{\text{BAVL}}$ (data bus available) pin are used to notify the external device of the DMAC channel that is to be used (see table 14.16, Notification of Transfer Channel in Eight-Channel DDT Mode).

When the DMAC is set up for eight-channel external request acceptance in DDT mode (DMAOR.DBL = 1), it is important to note that the $\overline{\text{BAVL}}$ pin has the two functions as shown in table 14.17.

Table 14.16 Notification of Transfer Channel in Eight-Channel DDT Mode

| $\overline{\text{BAVL}}/\overline{\text{ID2}}$ | ID[1:0] | Transfer Channel |
|--|---------|------------------|
| 1 | 00 | CH0 |
| | 01 | CH1 |
| | 10 | CH2 |
| | 11 | CH3 |
| 0 | 00 | CH4 |
| | 01 | CH5 |
| | 10 | CH6 |
| | 11 | CH7 |

Table 14.17 Function of $\overline{\text{BAVL}}$

| Function of $\overline{\text{BAVL}}$ | |
|--------------------------------------|--|
| $\overline{\text{TDACK}}$ = High | Bus available |
| $\overline{\text{TDACK}}$ = Low | Notification of channel number ($\overline{\text{ID2}}$) |

14.8.4 Clearing Request Queues by DTR Format

In DDT mode, the request queues of any channel can be cleared by using DTR.ID, DTR.MD, DTR.SZ, and DTR.COUNT [7:4] in a DTR format. This function is only available when DMAOR.DBL = 1. Table 14.18 shows the DTR format settings for clearing request queues.

Table 14.18 DTR Format for Clearing Request Queues

| DMAOR.DBL | DTR.ID | DTR.MD | DTR.SZ | DTR.COUNT[7:4] | Description |
|-----------|-------------------------------|--------|--------|----------------|---|
| 0 | 00 | 10 | 110 | * | Clear the request queues of all channels (1–7). |
| | | 11 | | | Clear the CH0 request-accepted flag Setting prohibited |
| 1 | 00 | 10 | 110 | * | Clear the request queues of all channels (1–7). |
| | | 11 | | | Clear the CH0 request-accepted flag. |
| | | 0001 | | | Clear the CH0 request-accepted flag |
| | | 0010 | | | Clear the CH1 request queues. |
| | | 0011 | | | Clear the CH2 request queues. |
| | | 0100 | | | Clear the CH3 request queues. |
| | | 0101 | | | Clear the CH4 request queues. |
| | | 0110 | | | Clear the CH5 request queues. |
| 0111 | Clear the CH6 request queues. | | | | |
| | | | 1000 | | Clear the CH7 request queues. |

Note: (SH7751R) DTR.SZ = DTR[31:29], DTR.ID = DTR[27:26], DTR.MD = DTR[25:24],
DTR.COUNT[7:4] = DTR[23:20]

14.8.5 Interrupt-Request Codes

When the number of transfers specified in DMATCR has been finished and the interrupt request is enabled (CHCR.IE = 1), a transfer-end interrupt request can be sent to the CPU from each channel. Table 14.19 lists the interrupt-request codes that are associated with these transfer-end interrupts.

Table 14.19 DMAC Interrupt-Request Codes

| Source of the Interrupt | Description | INTEVT Code | Priority |
|-------------------------|----------------------------|-------------|-----------------------|
| DMTE0 | CH0 transfer-end interrupt | H'640 | High ↑ ↓ Low |
| DMTE1 | CH1 transfer-end interrupt | H'660 | |
| DMTE2 | CH2 transfer-end interrupt | H'680 | |
| DMTE3 | CH3 transfer-end interrupt | H'6A0 | |
| DMTE4 | CH4 transfer-end interrupt | H'780 | |
| DMTE5 | CH5 transfer-end interrupt | H'7A0 | |
| DMTE6 | CH6 transfer-end interrupt | H'7C0 | |
| DMTE7 | CH7 transfer-end interrupt | H'7E0 | |
| DMAE | Address error interrupt | H'6C0 | Low |

DMTE4–DMTE7: These codes are not used in the SH7751.

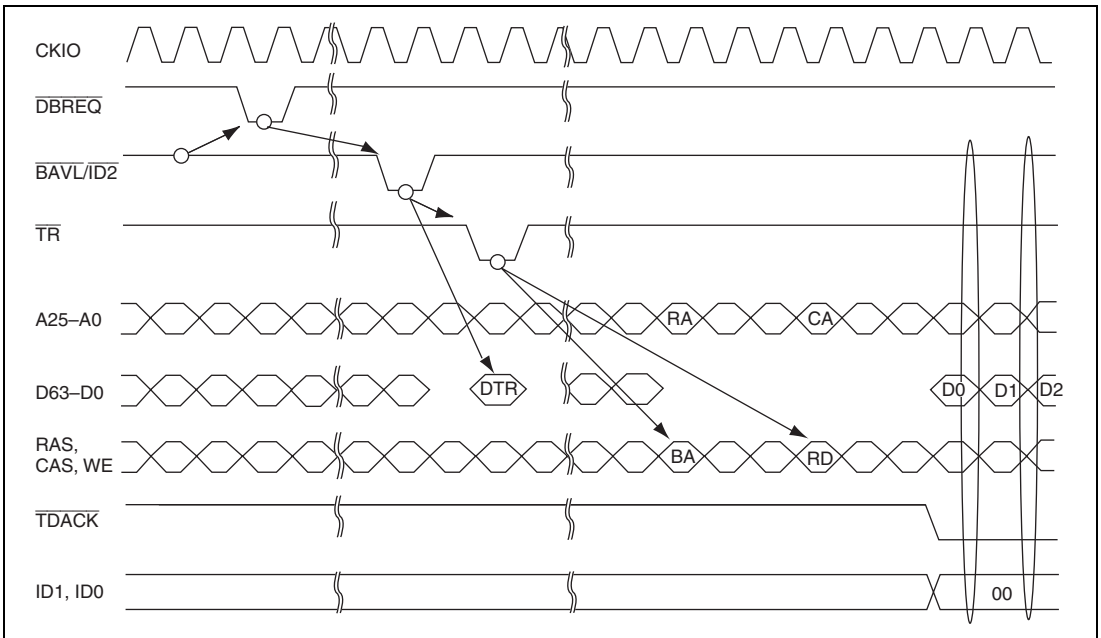


Figure 14.55 Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer

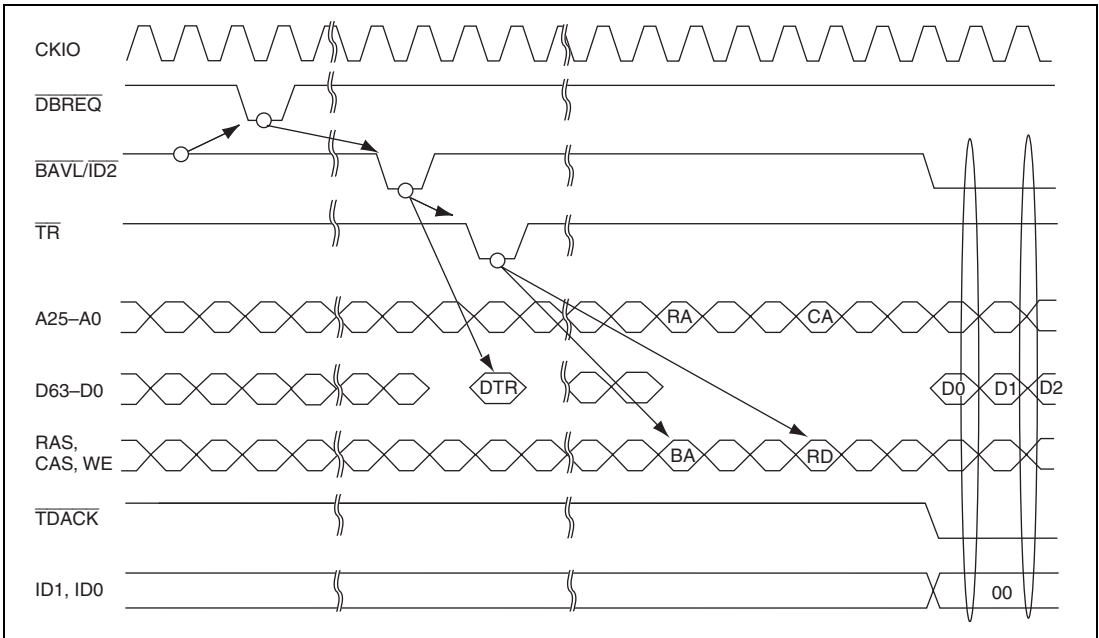


Figure 14.56 Single Address Mode/Cycle Steal Mode/External Bus → External Device/32-Byte Block Transfer/On-Demand Data Transfer on Channel 4

14.9 Usage Notes

1. When modifying SAR0–SAR3, DAR0–DAR3, DMATCR0–DMATCR3, and CHCR0–CHCR3 in the SH7751 or when modifying SAR0–SAR7, DAR0–DAR7, DMATCR0–DMATCR7, and CHCR0–CHCR7 in the SH7751R, first clear the DE bit for the relevant channel to 0.
2. The NMIF bit in DMAOR is set when an NMI interrupt is input even if the DMAC is not operating.

Confirmation method when DMA transfer is not executed correctly:

With the SH7751, read the NMIF, AE, and DME bits in DMAOR, the DE and TE bits in CHCR0–CHCR3, and DMATCR0–DMATCR3.

With the SH7751R, read the NMIF, AE, and DME bits in DMAOR, the DE and TE bits in CHCR0–CHCR7, and DMATCR0–DMATCR7. If NMIF was set before the transfer, the DMATCR transfer count will remain at the set value. If NMIF was set during the transfer, when the DE bit is 1 and the TE bit is 0 in CHCR0–CHCR3 in the SH7751 or CHCR0–CHCR7 in the SH7751R, the DMATCR value will indicate the remaining number of transfers.

Also, the next addresses to be accessed can be found by reading SAR0–SAR3 and DAR0–DAR3 in the SH7751 or SAR0–SAR7 and DAR0–DAR7 in the SH7751R. If the AE bit has been set, an address error has occurred. Check the set values in CHCR, SAR, and DAR.

3. Check that DMA transfer is not in progress before making a transition to the module standby state, standby mode, or deep sleep mode.
Either check that TE = 1 in the SH7751's CHCR0–CHCR3 or in the SH7751R's CHCR0–CHCR7, or clear DME to 0 in DMAOR to terminate DMA transfer. When DME is cleared to 0 in DMAOR, transfer halts at the end of the currently executing DMA bus cycle. Note, therefore, that transfer may not end immediately, depending on the transfer data size. DMA operation is not guaranteed if the module standby state, standby mode, or deep sleep mode is entered without confirming that DMA transfer has ended.
4. Do not specify a DMAC, CCN, BSC, UBC, or PCIC control register as the DMAC transfer source or destination.
5. When activating the DMAC, make the SAR, DAR, and DMATCR register settings for the relevant channel before setting DE to 1 in CHCR, or make the register settings with DE cleared to 0 in CHCR, then set DE to 1. It does not matter whether setting of the DME bit to 1 in DMAOR is carried out first or last. To operate the relevant channel, DME and DE must both be set to 1. The DMAC may not operate normally if the SAR, DAR, and DMATCR settings are not made (with the exception of the unused register in single address mode).
6. After the DMATCR count reaches 0 and DMA transfer ends normally, always write 0 to DMATCR even when executing the maximum number of transfers on the same channel.

7. When falling edge detection is used for external requests, keep the external request pin high when making DMAC settings.
8. When using the DMAC in single address mode, set an external address as the address. All channels will halt due to an address error if an on-chip peripheral module address is set.

Section 15 Serial Communication Interface (SCI)

15.1 Overview

This LSI is equipped with a single-channel serial communication interface (SCI) and a single-channel serial communication interface with built-in FIFO registers (SCI with FIFO: SCIF).

The SCI can handle both asynchronous and synchronous serial communication.

The SCI supports a smart card interface. This is a serial communication function supporting a subset of the ISO/IEC 7816-3 (identification cards) standard. For details, see section 17, Smart Card Interface.

The SCIF is a dedicated asynchronous communication serial interface with built-in 16-stage FIFO registers for both transmission and reception. For details, see section 16, Serial Communication Interface with FIFO (SCIF).

15.1.1 Features

SCI features are listed below.

- Choice of synchronous or asynchronous serial communication mode

- Asynchronous mode

Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A multiprocessor communication function is also provided that enables serial data communication with a number of processors.

There is a choice of 12 serial data transfer formats.

Data length: 7 or 8 bits

Stop bit length: 1 or 2 bits

Parity: Even/odd/none

Multiprocessor bit: 1 or 0

Receive error detection: Parity, overrun, and framing errors

Break detection: A break can be detected by reading the RxD pin level directly from the serial port register (SCSPTR1) when a framing error occurs.

— Synchronous mode

Serial data communication is synchronized with a clock. Serial data communication can be carried out with other chips that have a synchronous communication function.

There is a single serial data transfer format.

Data length: 8 bits

Receive error detection: Overrun errors

- Full-duplex communication capability

The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.

- On-chip baud rate generator allows any bit rate to be selected.

- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK pin

- Four interrupt sources

There are four interrupt sources—transmit-data-empty, transmit-end, receive-data-full, and receive-error—that can issue requests independently. The transmit-data-empty interrupt and receive-data-full interrupt can activate the DMA controller (DMAC) to execute a data transfer.

- When not in use, the SCI can be stopped by halting its clock supply to reduce power consumption.

15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the SCI.

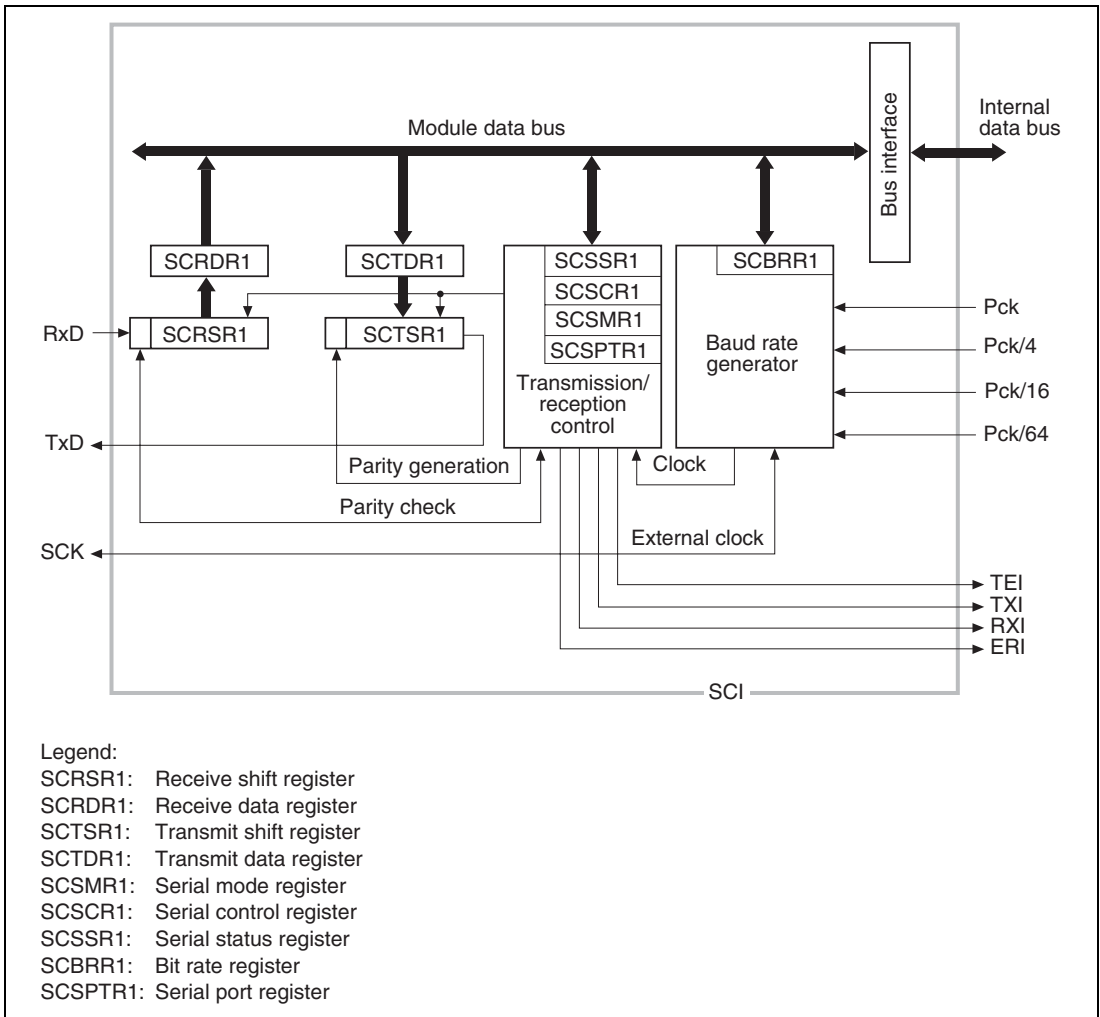


Figure 15.1 Block Diagram of SCI

15.1.3 Pin Configuration

Table 15.1 shows the SCI pin configuration.

Table 15.1 SCI Pins

| Pin Name | Abbreviation | I/O | Function |
|-------------------|--------------|--------|----------------------|
| Serial clock pin | SCK | I/O | Clock input/output |
| Receive data pin | RxD | Input | Receive data input |
| Transmit data pin | TxD | Output | Transmit data output |

Note: They are made to function as serial pins by performing SCI operation settings with the TE, RE, CKEI, and CKE0 bits in SCSCR1 and the C/\bar{A} bit in SCSMR1. Break state transmission and detection, can be set in the SCI's SCSPTR1 register.

15.1.4 Register Configuration

The SCI has the internal registers shown in table 15.2. These registers are used to specify asynchronous mode or synchronous mode, the data format, and the bit rate, and to perform transmitter/receiver control.

With the exception of the serial port register, the SCI registers are initialized in standby mode and in the module standby state as well as after a power-on reset or manual reset. When recovering from standby mode or the module standby state, the registers must be set again.

Table 15.2 SCI Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|-------------------------|--------------|---------------------|--------------------|------------|----------------|-------------|
| Serial mode register | SCSMR1 | R/W | H'00 | H'FFE00000 | H'1FE00000 | 8 |
| Bit rate register | SCBRR1 | R/W | H'FF | H'FFE00004 | H'1FE00004 | 8 |
| Serial control register | SCSCR1 | R/W | H'00 | H'FFE00008 | H'1FE00008 | 8 |
| Transmit data register | SCTDR1 | R/W | H'FF | H'FFE0000C | H'1FE0000C | 8 |
| Serial status register | SCSSR1 | R/(W)* ¹ | H'84 | H'FFE00010 | H'1FE00010 | 8 |
| Receive data register | SCRDR1 | R | H'00 | H'FFE00014 | H'1FE00014 | 8 |
| Serial port register | SCSPTR1 | R/W | H'00* ² | H'FFE0001C | H'1FE0001C | 8 |

Notes: 1. Only 0 can be written, to clear flags.

2. The value of bits 2 and 0 is undefined

15.2 Register Descriptions

15.2.1 Receive Shift Register (SCRSR1)

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

SCRSR1 is the register used to receive serial data.

The SCI sets serial data input from the RxD pin in SCRSR1 in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to SCRDR1 automatically.

SCRSR1 cannot be directly read or written to by the CPU.

15.2.2 Receive Data Register (SCRDR1)

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

SCRDR1 is the register that stores received serial data.

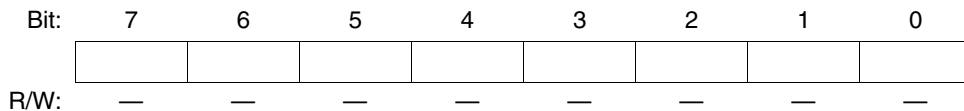
When the SCI has received one byte of serial data, it transfers the received data from SCRSR1 to SCRDR1 where it is stored, and completes the receive operation. SCRSR1 is then enabled for reception.

Since SCRSR1 and SCRDR1 function as a double buffer in this way, it is possible to receive data continuously.

SCRDR1 is a read-only register, and cannot be written to by the CPU.

SCRDR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

15.2.3 Transmit Shift Register (SCTSR1)



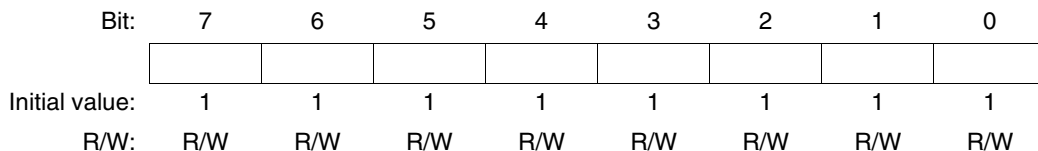
SCTSR1 is the register used to transmit serial data.

To perform serial data transmission, the SCI first transfers transmit data from SCTDR1 to SCTSR1, then sends the data to the TxD pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCTDR1 to SCTSR1, and transmission started, automatically. However, data transfer from SCTDR1 to SCTSR1 is not performed if the TDRE flag in the serial status register (SCSSR1) is set to 1.

SCTSR1 cannot be directly read or written to by the CPU.

15.2.4 Transmit Data Register (SCTDR1)



SCTDR1 is an 8-bit register that stores data for serial transmission.

When the SCI detects that SCTSR1 is empty, it transfers the transmit data written in SCTDR1 to SCTSR1 and starts serial transmission. Continuous serial transmission can be carried out by writing the next transmit data to SCTDR1 during serial transmission of the data in SCTSR1.

SCTDR1 can be read or written to by the CPU at all times.

SCTDR1 is initialized to H'FF by a power-on reset or manual reset, in standby mode, and in the module standby state.

15.2.5 Serial Mode Register (SCSMR1)

| | | | | | | | | |
|----------------|--------------|-----|-----|--------------|------|-----|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | C/ \bar{A} | CHR | PE | O/ \bar{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 |

SCSMR1 is an 8-bit register used to set the SCI's serial transfer format and select the baud rate generator clock source.

SCSMR1 can be read or written to by the CPU at all times.

SCSMR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

Bit 7—Communication Mode (C/ \bar{A}): Selects asynchronous mode or synchronous mode as the SCI operating mode.

| Bit 7: C/ \bar{A} | Description |
|---------------------|-----------------------------------|
| 0 | Asynchronous mode (Initial value) |
| 1 | Synchronous mode |

Bit 6—Character Length (CHR): Selects 7 or 8 bits as the data length in asynchronous mode. In synchronous mode, a fixed data length of 8 bits is used regardless of the CHR setting,

| Bit 6: CHR | Description |
|------------|----------------------------|
| 0 | 8-bit data (Initial value) |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of SCTDR1 is not transmitted.

Bit 5—Parity Enable (PE): In asynchronous mode, selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. In synchronous mode, parity bit addition and checking is not performed, regardless of the PE bit setting.

| Bit 5: PE | Description |
|-----------|---|
| 0 | Parity bit addition and checking disabled (Initial value) |
| 1 | Parity bit addition and checking enabled* |

Note: * When the PE bit is set to 1, the parity (even or odd) specified by the O/\bar{E} bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/\bar{E} bit.

Bit 4—Parity Mode (O/\bar{E}): Selects either even or odd parity for use in parity addition and checking. The O/\bar{E} bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking, in asynchronous mode. The O/\bar{E} bit setting is invalid in synchronous mode, and when parity addition and checking is disabled in asynchronous mode.

| Bit 4: O/\bar{E} | Description |
|--------------------|---|
| 0 | Even parity* ¹ (Initial value) |
| 1 | Odd parity* ² |

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is even.

2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is odd.

Bit 3—Stop Bit Length (STOP): Selects 1 or 2 bits as the stop bit length in asynchronous mode. The STOP bit setting is only valid in asynchronous mode. If synchronous mode is set, the STOP bit setting is invalid since stop bits are not added.

| Bit 3: STOP | Description |
|-------------|--|
| 0 | 1 stop bit* ¹ (Initial value) |
| 1 | 2 stop bits* ² |

Notes: 1. In transmission, a single 1-bit (stop bit) is added to the end of a transmit character before it is sent.

2. In transmission, two 1-bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, 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 it is 0, it is treated as the start bit of the next transmit character.

Bit 2—Multiprocessor Mode (MP): Selects a multiprocessor format. When a multiprocessor format is selected, the PE bit and O/\bar{E} bit parity settings are invalid. The MP bit setting is only valid in asynchronous mode; it is invalid in synchronous mode.

For details of the multiprocessor communication function, see section 15.3.3, Multiprocessor Communication Function.

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

Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock source for the on-chip baud rate generator. The clock source can be selected from Pck, Pck/4, Pck/16, and Pck/64, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 15.2.9, Bit Rate Register (SCBRR1).

| Bit 1: CKS1 | Bit 0: CKS0 | Description |
|-------------|-------------|---------------------------|
| 0 | 0 | Pck clock (Initial value) |
| | 1 | Pck/4 clock |
| 1 | 0 | Pck/16 clock |
| | 1 | Pck/64 clock |

Note: Pck: Peripheral clock

15.2.6 Serial Control Register (SCSCR1)

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

The SCSCR1 register performs enabling or disabling of SCI transfer operations, serial clock output in asynchronous mode, and interrupt requests, and selection of the serial clock source.

SCSCR1 can be read or written to by the CPU at all times.

SCSCR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables transmit-data-empty interrupt (TXI) request generation when serial transmit data is transferred from SCTDR1 to SCTS1 and the TDRE flag in SCSSR1 is set to 1.

| Bit 7: TIE | Description |
|------------|---|
| 0 | Transmit-data-empty interrupt (TXI) request disabled* (Initial value) |
| 1 | Transmit-data-empty interrupt (TXI) request enabled |

Note: * TXI interrupt requests can be cleared by reading 1 from the TDRE flag, then clearing it to 0, or by clearing the TIE bit to 0.

Bit 6—Receive Interrupt Enable (RIE): Enables or disables receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request generation when serial receive data is transferred from SCRSR1 to SCRDR1 and the RDRF flag in SCSSR1 is set to 1.

| Bit 6: RIE | Description |
|------------|---|
| 0 | Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request disabled* (Initial value) |
| 1 | Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request enabled |

Note: * RXI and ERI interrupt requests can be cleared by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or by clearing the RIE bit to 0.

Bit 5—Transmit Enable (TE): Enables or disables the start of serial transmission by the SCI.

| Bit 5: TE | Description |
|-----------|---|
| 0 | Transmission disabled* ¹ (Initial value) |
| 1 | Transmission enabled* ² |

Notes: 1. The TDRE flag in SCSSR1 is fixed at 1.

2. In this state, serial transmission is started when transmit data is written to SCTDR1 and the TDRE flag in SCSSR1 is cleared to 0.

SCSMR1 setting must be performed to decide the transmit format before setting the TE bit to 1.

Bit 4—Receive Enable (RE): Enables or disables the start of serial reception by the SCI.

| Bit 4: RE | Description |
|-----------|--|
| 0 | Reception disabled* ¹ (Initial value) |
| 1 | Reception enabled* ² |

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.
 2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in synchronous mode.
 SCSMR1 setting must be performed to decide the receive format before setting the RE bit to 1.

Bit 3—Multiprocessor Interrupt Enable (MPIE): Enables or disables multiprocessor interrupts. The MPIE bit setting is only valid in asynchronous mode when the MP bit in SCSMR1 is set to 1.

The MPIE bit setting is invalid in synchronous mode or when the MP bit is cleared to 0.

| Bit 3: MPIE | Description |
|-------------|--|
| 0 | Multiprocessor interrupts disabled (normal reception performed) (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • When the MPIE bit is cleared to 0 • When data with MPB = 1 is received |
| 1 | Multiprocessor interrupts enabled* |

Note: * When receive data including MPB = 1 is received, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCSCR1 are set to 1) and FER and ORER flag setting is enabled.

Bit 2—Transmit-End interrupt Enable (TEIE): Enables or disables transmit-end interrupt (TEI) request generation when there is no valid transmit data in SCTDR1 at the time for MSB data transmission.

| Bit 2: TEIE | Description |
|-------------|--|
| 0 | Transmit-end interrupt (TEI) request disabled* (Initial value) |
| 1 | Transmit-end interrupt (TEI) request enabled* |

Note: * TEI interrupt requests can be cleared by reading 1 from the TDRE flag in SCSSR1, then clearing it to 0 and clearing the TEND flag to 0, or by clearing the TEIE bit to 0.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin. The combination of the CKE1 and CKE0 bits determines whether the SCK pin functions as the serial clock output pin or the serial clock input pin.

The setting of the CKE0 bit, however, is only valid for internal clock operation (CKE1 = 0) in asynchronous mode. The CKE0 bit setting is invalid in synchronous mode and in the case of external clock operation (CKE1 = 1). The CKE1 and CKE0 bits must be set before determining the SCI's operating mode with SCSMR1.

For details of clock source selection, see table 15.9 in section 15.3, Operation.

| Bit 1: CKE1 | Bit 0: CKE0 | Description | |
|-------------|-------------|-------------------|--|
| 0 | 0 | Asynchronous mode | Internal clock/SCK pin functions as input pin (input signal ignored)* ¹ |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output* ¹ |
| | 1 | Asynchronous mode | Internal clock/SCK pin functions as clock output* ² |
| | | Synchronous mode | Internal clock/SCK pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK pin functions as clock input* ³ |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK pin functions as clock input* ³ |
| | | Synchronous mode | External clock/SCK pin functions as serial clock input |

Notes: 1. Initial value

2. Outputs a clock of the same frequency as the bit rate.

3. Inputs a clock with a frequency 16 times the bit rate.

15.2.7 Serial Status Register (SCSSR1)

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

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

SCSSR1 is an 8-bit register containing status flags that indicate the operating status of the SCI, and multiprocessor bits.

SCSSR1 can be read or written to by the CPU at all times. However, 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER. Also note that in order to clear these flags they must be read as 1 beforehand. The TEND flag and MPB flag are read-only flags and cannot be modified.

SCSSR1 is initialized to H'84 by a power-on reset or manual reset, in standby mode, and in the module standby state.

Bit 7—Transmit Data Register Empty (TDRE): Indicates that data has been transferred from SCTDR1 to SCTSR1 and the next serial transmit data can be written to SCTDR1.

| Bit 7: TDRE | Description |
|-------------|---|
| 0 | Valid transmit data has been written to SCTDR1 [Clearing conditions] <ul style="list-style-type: none"> • When 0 is written to TDRE after reading TDRE = 1 • When data is written to SCTDR1 by the DMAC |
| 1 | There is no valid transmit data in SCTDR1 (Initial value) [Setting conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When the TE bit in SCSCR1 is 0 • When data is transferred from SCTDR1 to SCTSR1 and data can be written to SCTDR1 |

Bit 6—Receive Data Register Full (RDRF): Indicates that the received data has been stored in SCRDR1.

| Bit 6: RDRF | Description |
|-------------|--|
| 0 | There is no valid receive data in SCRDR1 (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to RDRF after reading RDRF = 1 • When data in SCRDR1 is read by the DMAC |
| 1 | There is valid receive data in SCRDR1 [Setting condition] When serial reception ends normally and receive data is transferred from SCRSR1 to SCRDR1 |

Note: SCRDR1 and the RDRF flag are not affected and retain their previous values when an error is detected during reception or when the RE bit in SCSCR1 is cleared to 0.

If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error will occur and the receive data will be lost.

Bit 5—Overrun Error (ORER): Indicates that an overrun error occurred during reception, causing abnormal termination.

| Bit 5: ORER | Description |
|-------------|---|
| 0 | Reception in progress, or reception has ended normally* ¹ (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to ORER after reading ORER = 1 |
| 1 | An overrun error occurred during reception* ² [Setting condition] When the next serial reception is completed while RDRF = 1 |

Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCSCR1 is cleared to 0.

2. The receive data prior to the overrun error is retained in SCRDR1, and the data received subsequently is lost. Serial reception cannot be continued while the ORER flag is set to 1. In synchronous mode, serial transmission cannot be continued either.

Bit 4—Framing Error (FER): Indicates that a framing error occurred during reception in asynchronous mode, causing abnormal termination.

| Bit 4: FER | Description |
|------------|---|
| 0 | Reception in progress, or reception has ended normally* ¹ (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to FER after reading FER = 1 |
| 1 | A framing error occurred during reception [Setting condition] <p>When the SCI checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0*²</p> |

- Notes:
1. The FER flag is not affected and retains its previous state when the RE bit in SCSCR1 is cleared to 0.
 2. In 2-stop-bit mode, only the first stop bit is checked for a value of 1; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to SCRDR1 but the RDRF flag is not set. Serial reception cannot be continued while the FER flag is set to 1.

Bit 3—Parity Error (PER): Indicates that a parity error occurred during reception with parity addition in asynchronous mode, causing abnormal termination.

| Bit 3: PER | Description |
|------------|---|
| 0 | Reception in progress, or reception has ended normally* ¹ (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to PER after reading PER = 1 |
| 1 | A parity error occurred during reception* ² [Setting condition] <p>When, in reception, the number of 1-bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/\bar{E} bit in SCSMR1</p> |

- Notes:
1. The PER flag is not affected and retains its previous state when the RE bit in SCSCR1 is cleared to 0.
 2. If a parity error occurs, the receive data is transferred to SCRDR1 but the RDRF flag is not set. Serial reception cannot be continued while the PER flag is set to 1.

Bit 2—Transmit End (TEND): Indicates that there is no valid data in SCTDR1 when the last bit of the transmit character is sent, and transmission has been ended.

The TEND flag is read-only and cannot be modified.

| Bit 2: TEND | Description |
|-------------|--|
| 0 | Transmission is in progress [Clearing conditions] <ul style="list-style-type: none"> • When 0 is written to TDRE after reading TDRE = 1 • When data is written to SCTDR1 by the DMAC |
| 1 | Transmission has been ended (Initial value) [Setting conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When the TE bit in SCSCR1 is 0 • When TDRE = 1 on transmission of the last bit of a 1-byte serial transmit character |

Bit 1—Multiprocessor Bit (MPB): This bit is read-only and cannot be written to. The read value is undefined.

Note: This bit is prepared for storing a multi-processor bit in the received data when the receipt is carried out with a multi-processor format in asynchronous mode, however, this does not function correctly in this LSI. Do not use the read value from this bit.

Bit 0—Multiprocessor Bit Transfer (MPBT): When transmission is performed using a multiprocessor format in asynchronous mode, MPBT stores the multiprocessor bit to be added to the transmit data.

The MPBT bit setting is invalid in synchronous mode, when a multiprocessor format is not used, and when the operation is not transmission.

Unlike transmit data, the MPBT bit is not double-buffered, so it is necessary to check whether transmission has been completed before changing its value.

| Bit 0: MPBT | Description |
|-------------|---|
| 0 | Data with a 0 multiprocessor bit is transmitted (Initial value) |
| 1 | Data with a 1 multiprocessor bit is transmitted |

15.2.8 Serial Port Register (SCSPTR1)

| | | | | | | | | |
|----------------|-----|---|---|---|--------|--------|--------|--------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | EIO | — | — | — | SPB1IO | SPB1DT | SPB0IO | SPB0DT |
| Initial value: | 0 | 0 | 0 | 0 | 0 | — | 0 | — |
| R/W: | R/W | — | — | — | R/W | R/W | R/W | R/W |

SCSPTR1 is an 8-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface (SCI) pins. Input data can be read from the RxD pin, output data written to the TxD pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. SCK pin data reading and output data writing can be performed by means of bits 3 and 2. Bit 7 controls enabling and disabling of the RXI interrupt.

SCSPTR1 can be read or written to by the CPU at all times. All SCSPTR1 bits except bits 2 and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 2 and 0 is undefined. SCSPTR1 is not initialized in the module standby state or standby mode.

Bit 7—Error Interrupt Only (EIO): When the EIO bit is 1, an RXI interrupt request is not sent to the CPU even if the RIE bit is set to 1. When the DMAC is used, this setting means that only ERI interrupts are handled by the CPU. The DMAC transfers read data to memory or another peripheral module. This bit specifies enabling or disabling of the RXI interrupt.

| Bit 7: EIO | Description |
|------------|--|
| 0 | When the RIE bit is 1, RXI and ERI interrupts are sent to INTC (Initial value) |
| 1 | When the RIE bit is 1, only ERI interrupts are sent to INTC |

Bits 6 to 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Serial Port Clock Port I/O (SPB1IO): Specifies serial port SCK pin input/output. When the SCK pin is actually set as a port output pin and outputs the value set by the SPB1DT bit, the $\overline{C/A}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1 should be cleared to 0.

| Bit 3: SPB1IO | Description |
|---------------|---|
| 0 | SPB1DT bit value is not output to the SCK pin (Initial value) |
| 1 | SPB1DT bit value is output to the SCK pin |

Bit 2—Serial Port Clock Port Data (SPB1DT): Specifies the serial port SCK pin input/output data. Input or output is specified by the SPB1IO bit (see the description of bit 3, SPB1IO, for details). When output is specified, the value of the SPB1DT bit is output to the SCK pin. The SCK pin value is read from the SPB1DT bit regardless of the value of the SPB1IO bit. The initial value of this bit after a power-on or manual reset is undefined.

| Bit 2: SPB1DT | Description |
|---------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Bit 1—Serial Port Break I/O (SPB0IO): Specifies the serial port TxD pin output condition. When the TxD pin is actually set as a port output pin and outputs the value set by the SPB0DT bit, the TE bit in SCSCR1 should be cleared to 0.

| Bit 1: SPB0IO | Description |
|---------------|---|
| 0 | SPB0DT bit value is not output to the TxD pin (Initial value) |
| 1 | SPB0DT bit value is output to the TxD pin |

Bit 0—Serial Port Break Data (SPB0DT): Specifies the serial port RxD pin input data and TxD pin output data. The TxD pin output condition is specified by the SPB0IO bit (see the description of bit 1, SPB0IO, for details). When the TxD pin is designated as an output, the value of the SPB0DT bit is output to the TxD pin. The RxD pin value is read from the SPB0DT bit regardless of the value of the SPB0IO bit. The initial value of this bit after a power-on or manual reset is undefined.

| Bit 0: SPB0DT | Description |
|---------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

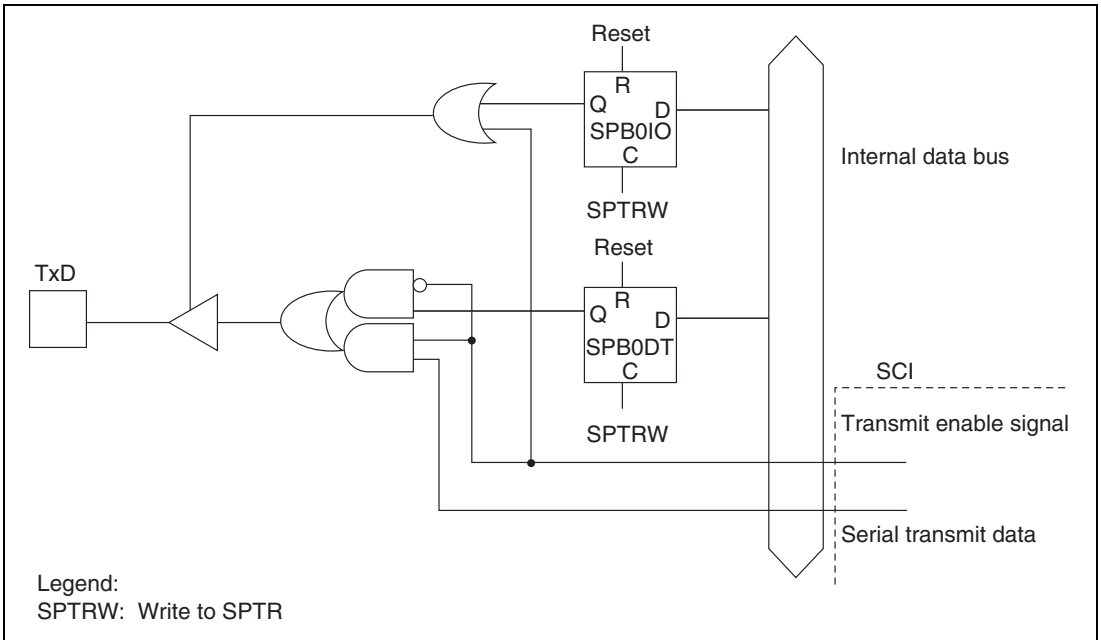


Figure 15.3 TxD Pin

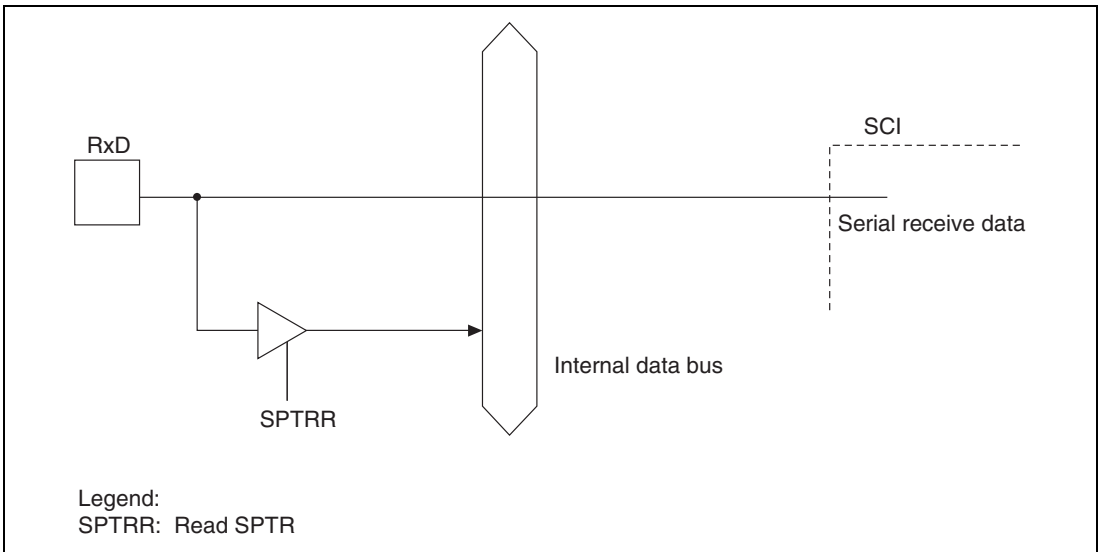


Figure 15.4 RxD Pin

15.2.9 Bit Rate Register (SCBRR1)

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

SCBRR1 is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SCSMR1.

SCBRR1 can be read or written to by the CPU at all times.

SCBRR1 is initialized to H'FF by a power-on reset or manual reset, in standby mode, and in the module standby state.

The SCBRR1 setting is found from the following equations.

Asynchronous mode:

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

Synchronous mode:

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

Where B: Bit rate (bits/s)

N: SCBRR1 setting for baud rate generator ($0 \leq N \leq 255$)

Pck: Peripheral module operating frequency (MHz)

n: Baud rate generator input clock ($n = 0$ to 3)

(See the table below for the relation between n and the clock.)

| n | Clock | SCSMR1 Setting | |
|---|--------|----------------|------|
| | | CKS1 | CKS0 |
| 0 | Pck | 0 | 0 |
| 1 | Pck/4 | 0 | 1 |
| 2 | Pck/16 | 1 | 0 |
| 3 | Pck/64 | 1 | 1 |

The bit rate error in asynchronous mode is found from the following equation:

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

Table 15.3 shows sample SCBRR1 settings in asynchronous mode, and table 15.4 shows sample SCBRR1 settings in synchronous mode.

Pck (MHz)

| Bit Rate (bits/s) | 6 | | | 6.144 | | | 7.37288 | | | 8 | | |
|----------------------|---|-----|-----------|-------|-----|-----------|---------|-----|-----------|---|-----|-----------|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 106 | -0.44 | 2 | 108 | 0.08 | 2 | 130 | -0.07 | 2 | 141 | 0.03 |
| 150 | 2 | 77 | 0.16 | 2 | 79 | 0.00 | 2 | 95 | 0.00 | 2 | 103 | 0.16 |
| 300 | 1 | 155 | 0.16 | 1 | 159 | 0.00 | 1 | 191 | 0.00 | 1 | 207 | 0.16 |
| 600 | 1 | 77 | 0.16 | 1 | 79 | 0.00 | 1 | 95 | 0.00 | 1 | 103 | 0.16 |
| 1200 | 0 | 155 | 0.16 | 0 | 159 | 0.00 | 0 | 191 | 0.00 | 0 | 207 | 0.16 |
| 2400 | 0 | 77 | 0.16 | 0 | 79 | 0.00 | 0 | 95 | 0.00 | 0 | 103 | 0.16 |
| 4800 | 0 | 38 | 0.16 | 0 | 39 | 0.00 | 0 | 47 | 0.00 | 0 | 51 | 0.16 |
| 9600 | 0 | 19 | -2.34 | 0 | 19 | 0.00 | 0 | 23 | 0.00 | 0 | 25 | 0.16 |
| 19200 | 0 | 9 | -2.34 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 12 | 0.16 |
| 31250 | 0 | 5 | 0.00 | 0 | 5 | 2.40 | 0 | 6 | 5.33 | 0 | 7 | 0.00 |
| 38400 | 0 | 4 | -2.34 | 0 | 4 | 0.00 | 0 | 5 | 0.00 | 0 | 6 | -6.99 |

Pck (MHz)

| Bit Rate (bits/s) | 9.8304 | | | 10 | | | 12 | | | 12.288 | | |
|----------------------|--------|-----|-----------|----|-----|-----------|----|-----|-----------|--------|-----|-----------|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 174 | -0.26 | 2 | 177 | -0.25 | 2 | 212 | 0.03 | 2 | 217 | 0.08 |
| 150 | 2 | 127 | 0.00 | 2 | 129 | 0.16 | 2 | 155 | 0.16 | 2 | 159 | 0.00 |
| 300 | 1 | 255 | 0.00 | 2 | 64 | 0.16 | 2 | 77 | 0.16 | 2 | 79 | 0.00 |
| 600 | 1 | 127 | 0.00 | 1 | 129 | 0.16 | 1 | 155 | 0.16 | 1 | 159 | 0.00 |
| 1200 | 0 | 255 | 0.00 | 1 | 64 | 0.16 | 1 | 77 | 0.16 | 1 | 79 | 0.00 |
| 2400 | 0 | 127 | 0.00 | 0 | 129 | 0.16 | 0 | 155 | 0.16 | 0 | 159 | 0.00 |
| 4800 | 0 | 63 | 0.00 | 0 | 64 | 0.16 | 0 | 77 | 0.16 | 0 | 79 | 0.00 |
| 9600 | 0 | 31 | 0.00 | 0 | 32 | -1.36 | 0 | 38 | 0.16 | 0 | 39 | 0.00 |
| 19200 | 0 | 15 | 0.00 | 0 | 15 | 1.73 | 0 | 19 | 0.16 | 0 | 19 | 0.00 |
| 31250 | 0 | 9 | -1.70 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 11 | 2.40 |
| 38400 | 0 | 7 | 0.00 | 0 | 7 | 1.73 | 0 | 9 | -2.34 | 0 | 9 | 0.00 |

| Bit Rate (bits/s) | Pck (MHz) | | | | | | | | | | | |
|----------------------|-----------|-----|-----------|----|-----|-----------|---------|-----|-----------|----|-----|-----------|
| | 14.7456 | | | 16 | | | 19.6608 | | | 20 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 64 | 0.70 | 3 | 70 | 0.03 | 3 | 86 | 0.31 | 3 | 88 | -0.25 |
| 150 | 2 | 191 | 0.00 | 2 | 207 | 0.16 | 2 | 255 | 0.00 | 3 | 64 | 0.16 |
| 300 | 2 | 95 | 0.00 | 2 | 103 | 0.16 | 2 | 127 | 0.00 | 2 | 129 | 0.16 |
| 600 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 1200 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 2400 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 4800 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 9600 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 19200 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | -1.36 |
| 31250 | 0 | 14 | -1.70 | 0 | 15 | 0.00 | 0 | 19 | -1.70 | 0 | 19 | 0.00 |
| 38400 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |

| Bit Rate (bits/s) | Pck (MHz) | | | | | | | | | | | |
|----------------------|-----------|-----|-----------|---|-----|-----------|---|-----|-----------|---|-----|-----------|
| | 24 | | 24.576 | | | 28.7 | | | 30 | | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 106 | -0.44 | 3 | 108 | 0.08 | 3 | 126 | 0.31 | 3 | 132 | 0.13 |
| 150 | 3 | 77 | 0.16 | 3 | 79 | 0.00 | 3 | 92 | 0.46 | 3 | 97 | -0.35 |
| 300 | 2 | 155 | 0.16 | 2 | 159 | 0.00 | 2 | 186 | -0.08 | 2 | 194 | 0.16 |
| 600 | 2 | 77 | 0.16 | 2 | 79 | 0.00 | 2 | 92 | 0.46 | 2 | 97 | -0.35 |
| 1200 | 1 | 155 | 0.16 | 1 | 159 | 0.00 | 1 | 186 | -0.08 | 1 | 194 | 0.16 |
| 2400 | 1 | 77 | 0.16 | 1 | 79 | 0.00 | 1 | 92 | 0.46 | 1 | 97 | -0.35 |
| 4800 | 0 | 155 | 0.16 | 0 | 159 | 0.00 | 0 | 186 | -0.08 | 0 | 194 | -1.36 |
| 9600 | 0 | 77 | 0.16 | 0 | 79 | 0.00 | 0 | 92 | 0.46 | 0 | 97 | -0.35 |
| 19200 | 0 | 38 | 0.16 | 0 | 39 | 0.00 | 0 | 46 | -0.61 | 0 | 48 | -0.35 |
| 31250 | 0 | 23 | 0.00 | 0 | 24 | -1.70 | 0 | 28 | -1.03 | 0 | 29 | 0.00 |
| 38400 | 0 | 19 | -2.34 | 0 | 19 | 0.00 | 0 | 22 | 1.55 | 0 | 23 | 1.73 |

Legend:

Blank: No setting is available.

—: A setting is available but error occurs.

Table 15.4 Examples of Bit Rates and SCBRR1 Settings in Synchronous Mode

| Bit Rate (bits/s) | Pck (MHz) | | | | | | | | | |
|-------------------|-----------|-----|---|-----|----|-----|------|-----|----|-----|
| | 4 | | 8 | | 16 | | 28.7 | | 30 | |
| | n | N | n | N | n | N | n | N | n | N |
| 10 | — | — | — | — | — | — | — | — | — | — |
| 250 | 2 | 249 | 3 | 124 | 3 | 249 | — | — | — | — |
| 500 | 2 | 124 | 2 | 249 | 3 | 124 | 3 | 223 | 3 | 233 |
| 1k | 1 | 249 | 2 | 124 | 2 | 249 | 3 | 111 | 3 | 116 |
| 2.5k | 1 | 99 | 1 | 199 | 2 | 99 | 2 | 178 | 2 | 187 |
| 5k | 0 | 199 | 1 | 99 | 1 | 199 | 2 | 89 | 2 | 93 |
| 10k | 0 | 99 | 0 | 199 | 1 | 99 | 1 | 178 | 1 | 187 |
| 25k | 0 | 39 | 0 | 79 | 0 | 159 | 1 | 71 | 1 | 74 |
| 50k | 0 | 19 | 0 | 39 | 0 | 79 | 0 | 143 | 0 | 149 |
| 100k | 0 | 9 | 0 | 19 | 0 | 39 | 0 | 71 | 0 | 74 |
| 250k | 0 | 3 | 0 | 7 | 0 | 15 | — | — | 0 | 29 |
| 500k | 0 | 1 | 0 | 3 | 0 | 7 | — | — | 0 | 14 |
| 1M | 0 | 0* | 0 | 1 | 0 | 3 | — | — | — | — |
| 2M | | | 0 | 0* | 0 | 1 | — | — | — | — |

Legend:

Blank: No setting is available.

—: A setting is available but error occurs.

Notes: As far as possible, the setting should be made so that the error is within 1%.

* Continuous transmission/reception is not possible.

Table 15.5 shows the maximum bit rate for various frequencies in asynchronous mode. Tables 15.6 and 15.7 show the maximum bit rates with external clock input.

Table 15.5 Maximum Bit Rate for Various Frequencies with Baud Rate Generator (Asynchronous Mode)

| Pck (MHz) | Maximum Bit Rate (bits/s) | Settings | |
|-----------|---------------------------|----------|---|
| | | 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 |
| 8 | 250000 | 0 | 0 |
| 9.8304 | 307200 | 0 | 0 |
| 12 | 375000 | 0 | 0 |
| 14.7456 | 460800 | 0 | 0 |
| 16 | 500000 | 0 | 0 |
| 19.6608 | 614400 | 0 | 0 |
| 20 | 625000 | 0 | 0 |
| 24 | 750000 | 0 | 0 |
| 24.576 | 768000 | 0 | 0 |
| 28.7 | 896875 | 0 | 0 |
| 30 | 937500 | 0 | 0 |

Table 15.6 Maximum Bit Rate with External Clock Input (Asynchronous Mode)

| Pck (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 |
| 8 | 2.0000 | 125000 |
| 9.8304 | 2.4576 | 153600 |
| 12 | 3.0000 | 187500 |
| 14.7456 | 3.6864 | 230400 |
| 16 | 4.0000 | 250000 |
| 19.6608 | 4.9152 | 307200 |
| 20 | 5.0000 | 312500 |
| 24 | 6.0000 | 375000 |
| 24.576 | 6.1440 | 384000 |
| 28.7 | 7.1750 | 448436 |
| 30 | 7.5000 | 468750 |

Table 15.7 Maximum Bit Rate with External Clock Input (Synchronous Mode)

| Pck (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bits/s) |
|------------------|-----------------------------------|----------------------------------|
| 8 | 1.3333 | 1333333.3 |
| 16 | 2.6667 | 2666666.7 |
| 24 | 4.0000 | 4000000.0 |
| 28.7 | 4.7833 | 4783333.3 |
| 30 | 5.0000 | 5000000.0 |

15.3 Operation

15.3.1 Overview

The SCI can carry out serial communication in two modes: asynchronous mode in which synchronization is achieved character by character, and synchronous mode in which synchronization is achieved with clock pulses.

Selection of asynchronous or synchronous mode and the transmission format is made using SCSMR1 as shown in table 15.8. The SCI clock source is determined by a combination of the C/\bar{A} bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1, as shown in table 15.9.

- Asynchronous mode
 - Data length: Choice of 7 or 8 bits
 - Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
 - Detection of framing, parity, and overrun errors, and breaks, during reception
 - Choice of internal or external clock as SCI clock source
 - When internal clock is selected: The SCI operates on the baud rate generator clock and a clock with the same frequency as the bit rate can be output.
 - When external clock is selected: A clock with a frequency of 16 times the bit rate must be input (the on-chip baud rate generator is not used).
- Synchronous mode
 - Transfer format: Fixed 8-bit data
 - Detection of overrun errors during reception
 - Choice of internal or external clock as SCI clock source
 - When internal clock is selected: The SCI operates on the baud rate generator clock and a serial clock is output off-chip.
 - When external clock is selected: The on-chip baud rate generator is not used, and the SCI operates on the input serial clock.

Table 15.8 SCSMR1 Settings for Serial Transfer Format Selection

| SCSMR1 Settings | | | | | | SCI Transfer Format | | | | |
|------------------------|---------------|--------------|--------------|----------------|----------------------------|---------------------|----------------------------|---------------|--------------------|--------|
| Bit 7: C/ \bar{A} | Bit 6: CHR | Bit 2: MP | Bit 5: PE | Bit 3: STOP | Mode | Data Length | Multi- processor Bit | Parity Bit | Stop Bit Length | |
| 0 | 0 | 0 | 0 | 0 | Asynchronous mode | 8-bit data | No | No | 1 bit | |
| | | | | 1 | | | | | 2 bits | |
| | | | | 0 | | | | | 1 bit | |
| | | | | 1 | | | | | 2 bits | |
| | 1 | 0 | 0 | 0 | (multiprocessor format) | 7-bit data | | | No | 1 bit |
| | | | | | | | | | 1 | 2 bits |
| | | | | | | | | | 1 | 1 bit |
| | | | | | | | | | 1 | 2 bits |
| 0 | 1 | * | * | 0 | Asynchronous mode | 8-bit data | Yes | No | 1 bit | |
| | | | | 1 | | | | | 2 bits | |
| | | | | 0 | | | | | 7-bit data | 1 bit |
| | | | | 1 | | | | | 2 bits | |
| 1 | * | * | * | * | Synchronous mode | 8-bit data | No | | None | |

Note: An asterisk in the table means "Don't care."

Table 15.9 SCSMR1 and SCSCR1 Settings for SCI Clock Source Selection

| SCSMR1 Bit 7: C/A | SCSCR1 Setting | | Mode | SCI Transmit/Receive Clock | |
|-------------------------|----------------|----------------|----------------------|----------------------------|---|
| | Bit 1: CKE1 | Bit 0: CKE0 | | Clock Source | SCK Pin Function |
| 0 | 0 | 0 | Asynchronous mode | Internal | SCI does not use SCK pin |
| | | 1 | | | Outputs clock with same frequency as bit rate |
| | 1 | 0 | | External | Inputs clock with frequency of 16 times the bit rate |
| | | 1 | | | |
| 1 | 0 | 0 | Synchronous mode | Internal | Outputs serial clock |
| | | 1 | | | |
| | 1 | 0 | | External | Inputs serial clock |
| | | 1 | | | |

15.3.2 Operation in Asynchronous Mode

In asynchronous mode, characters are sent or received, each preceded by a start bit indicating the start of communication and followed by one or two stop bits indicating the end of communication. Serial communication is thus carried out with synchronization established on a character-by-character basis.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 15.5 shows the general format for asynchronous serial communication.

In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication.

One serial communication character consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, the SCI performs synchronization at the falling edge of the start bit in reception. The SCI samples the data on the eighth pulse of a clock with a frequency of 16 times the length of one bit, so that the transfer data is latched at the center of each bit.

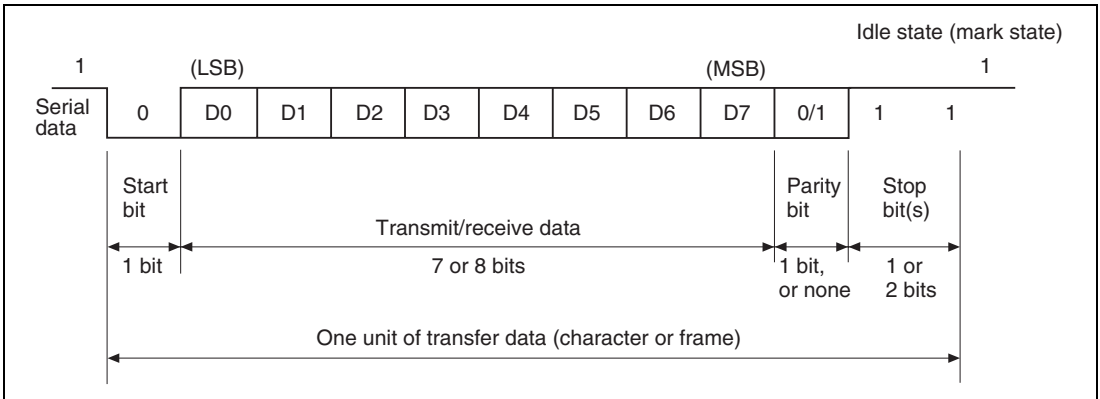


Figure 15.5 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

Data Transfer Format

Table 15.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SCSMR1 setting.

Table 15.10 Serial Transfer Formats (Asynchronous Mode)

| SCSMR1 Settings | | | | Serial Transfer Format and Frame Length | | | | | | | | | | | | |
|-----------------|----|----|------|---|------------|---|---|---|---|---|---|------|------|------|------|--|
| CHR | PE | MP | STOP | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 0 | 0 | 0 | 0 | S | 8-bit data | | | | | | | | STOP | | | |
| 0 | 0 | 0 | 1 | S | 8-bit data | | | | | | | | STOP | STOP | | |
| 0 | 1 | 0 | 0 | S | 8-bit data | | | | | | | | P | STOP | | |
| 0 | 1 | 0 | 1 | S | 8-bit data | | | | | | | | P | STOP | STOP | |
| 1 | 0 | 0 | 0 | S | 7-bit data | | | | | | | STOP | | | | |
| 1 | 0 | 0 | 1 | S | 7-bit data | | | | | | | STOP | STOP | | | |
| 1 | 1 | 0 | 0 | S | 7-bit data | | | | | | | P | STOP | | | |
| 1 | 1 | 0 | 1 | S | 7-bit data | | | | | | | P | STOP | STOP | | |
| 0 | * | 1 | 0 | S | 8-bit data | | | | | | | | MPB | STOP | | |
| 0 | * | 1 | 1 | S | 8-bit data | | | | | | | | MPB | STOP | STOP | |
| 1 | * | 1 | 0 | S | 7-bit data | | | | | | | MPB | STOP | | | |
| 1 | * | 1 | 1 | S | 7-bit data | | | | | | | MPB | STOP | STOP | | |

Legend:

S: Start bit

STOP: Stop bit

P: Parity bit

MPB: Multiprocessor bit

Note: An asterisk in the table means “Don't care.”

Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK pin can be selected as the SCI's serial clock, according to the setting of the C/\bar{A} bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1. For details of SCI clock source selection, see table 15.9.

When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is at the center of each transmit data bit, as shown in figure 15.6.

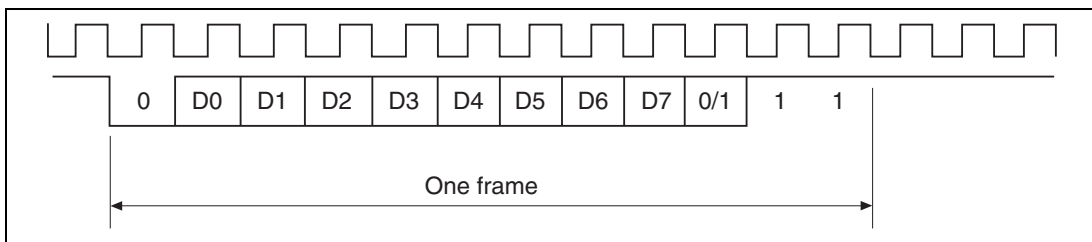


Figure 15.6 Relation between Output Clock and Transfer Data Phase (Asynchronous Mode)

Data Transfer Operations

SCI Initialization (Asynchronous Mode): Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR1 to 0, then initialize the SCI as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and SCTSR1 is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of SCRDR1.

When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 15.7 shows a sample SCI initialization flowchart.

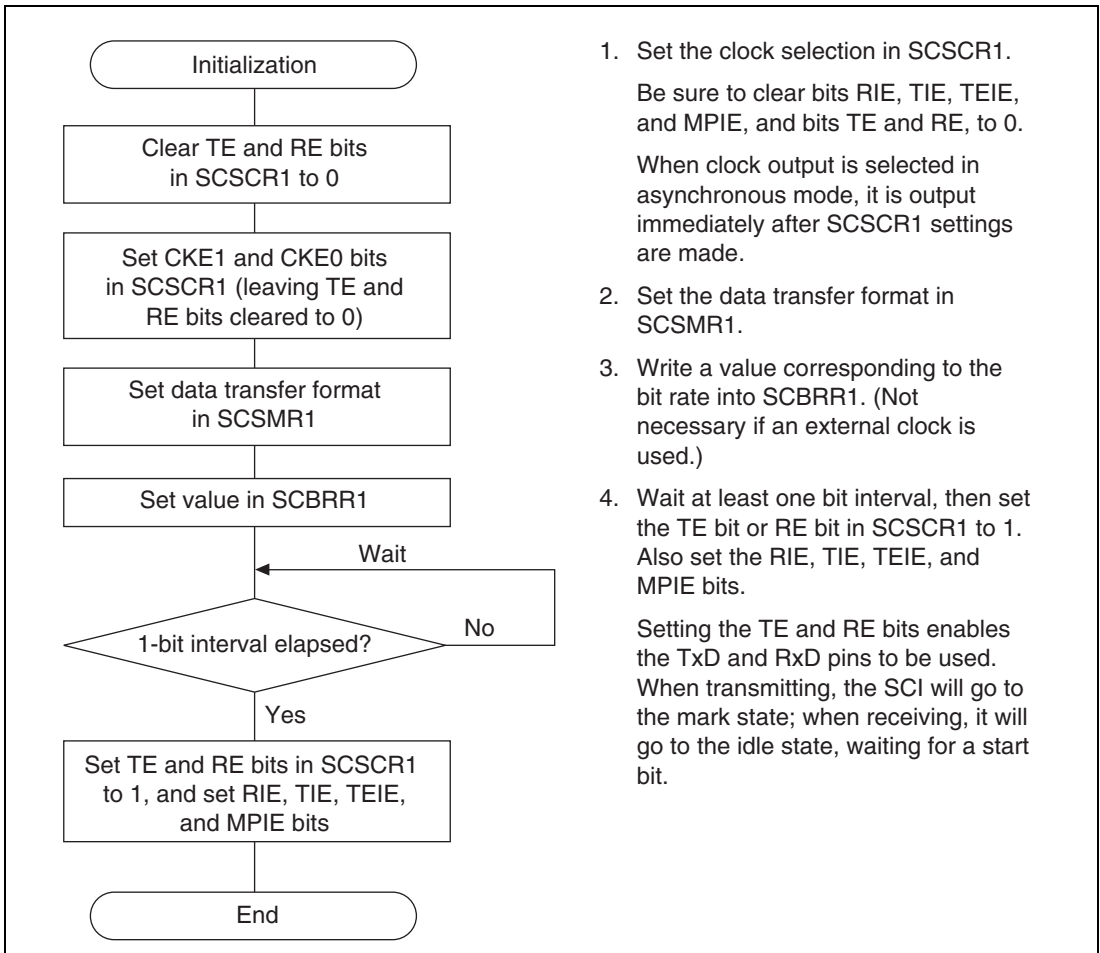
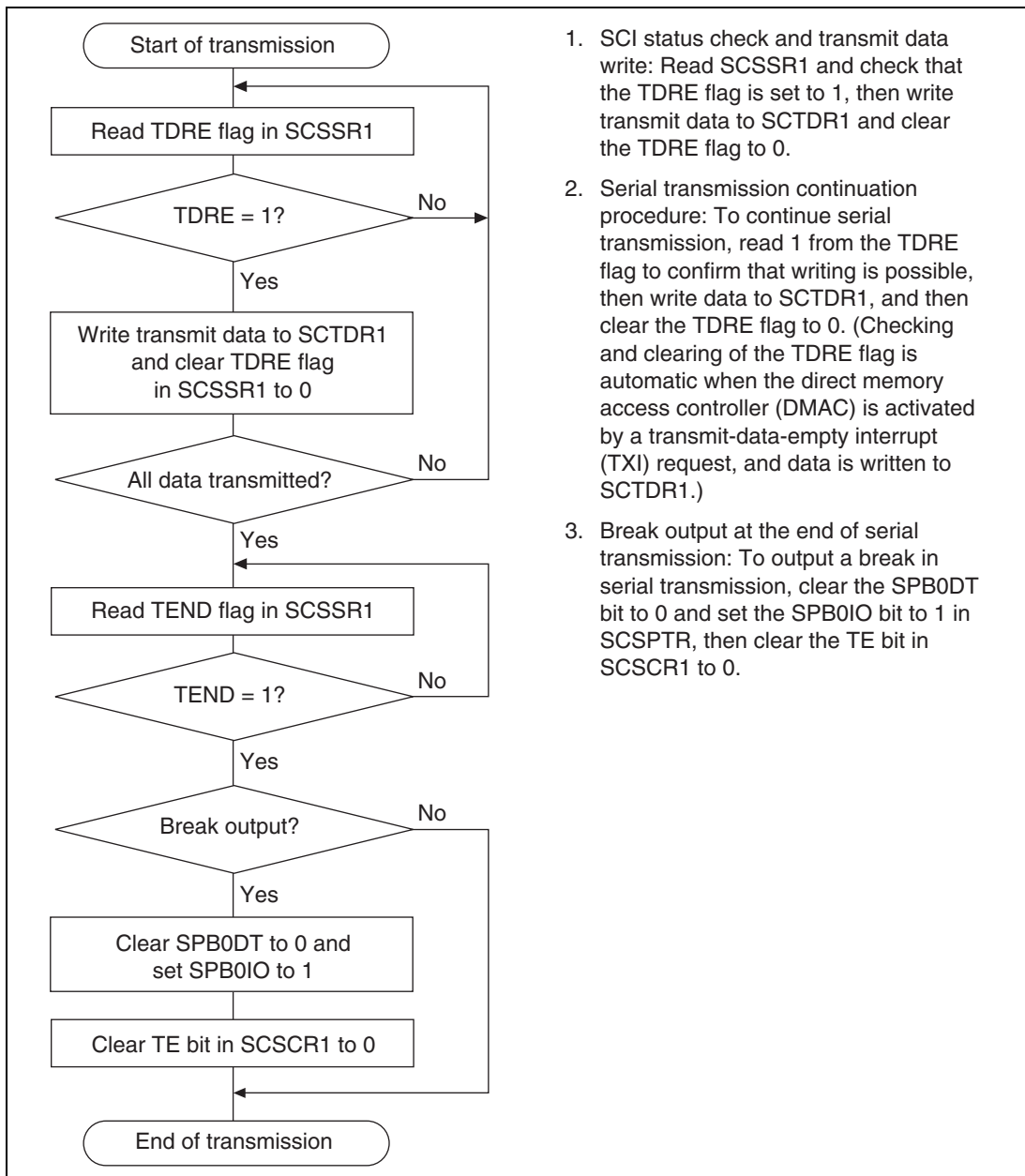


Figure 15.7 Sample SCI Initialization Flowchart

Serial Data Transmission (Asynchronous Mode): Figure 15.8 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCI for transmission.



1. SCI status check and transmit data write: Read SCSSR1 and check that the TDRE flag is set to 1, then write transmit data to SCTDR1 and clear the TDRE flag to 0.
2. Serial transmission continuation procedure: To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1, and then clear the TDRE flag to 0. (Checking and clearing of the TDRE flag is automatic when the direct memory access controller (DMAC) is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1.)
3. Break output at the end of serial transmission: To output a break in serial transmission, clear the SPB0DT bit to 0 and set the SPB0IO bit to 1 in SCSPTR, then clear the TE bit in SCSSR1 to 0.

Figure 15.8 Sample Serial Transmission Flowchart

In serial transmission, the SCI operates as described below.

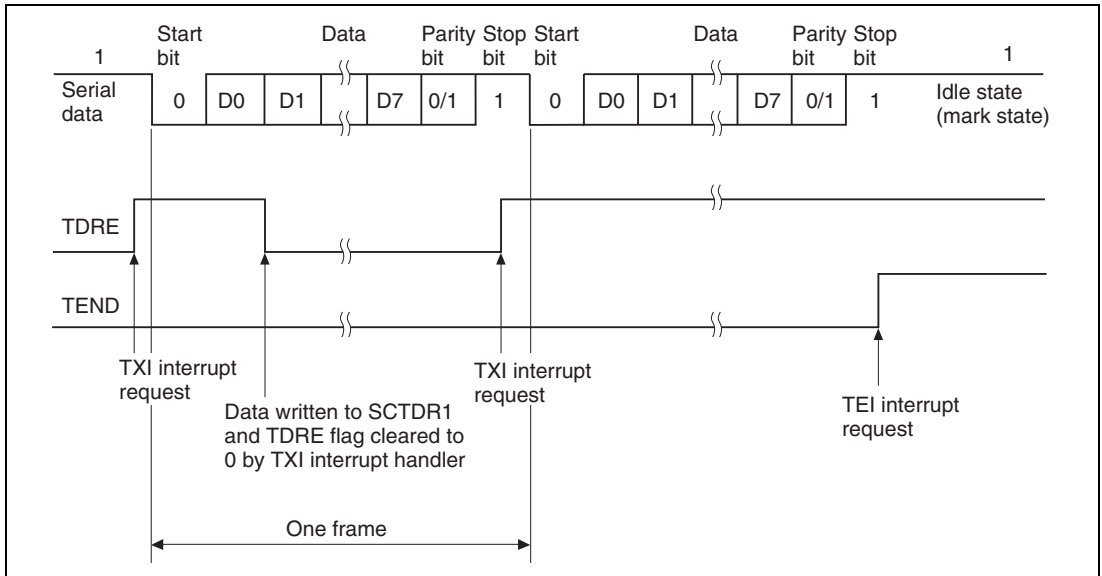
1. The SCI monitors the TDRE flag in SCSSR1. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
2. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

- a. Start bit: One 0-bit is output.
 - b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
 - c. Parity bit or multiprocessor bit: One parity bit (even or odd parity), or one multiprocessor bit is output. (A format in which neither a parity bit nor a multiprocessor bit is output can also be selected.)
 - d. Stop bit(s): One or two 1-bits (stop bits) are output.
 - e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCI checks the TDRE flag at the timing for sending the stop bit. If the TDRE flag is cleared to 0, data is transferred from SCTDR1 to SCTSR1, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SCSSR1 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output continuously. If the TEIE bit in SCSCR1 is set to 1 at this time, a TEI interrupt request is generated.

Figure 15.9 shows an example of the operation for transmission in asynchronous mode.



**Figure 15.9 Example of Transmit Operation in Asynchronous Mode
(Example with 8-Bit Data, Parity, One Stop Bit)**

Serial Data Reception (Asynchronous Mode): Figure 15.10 shows a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling the SCI for reception.

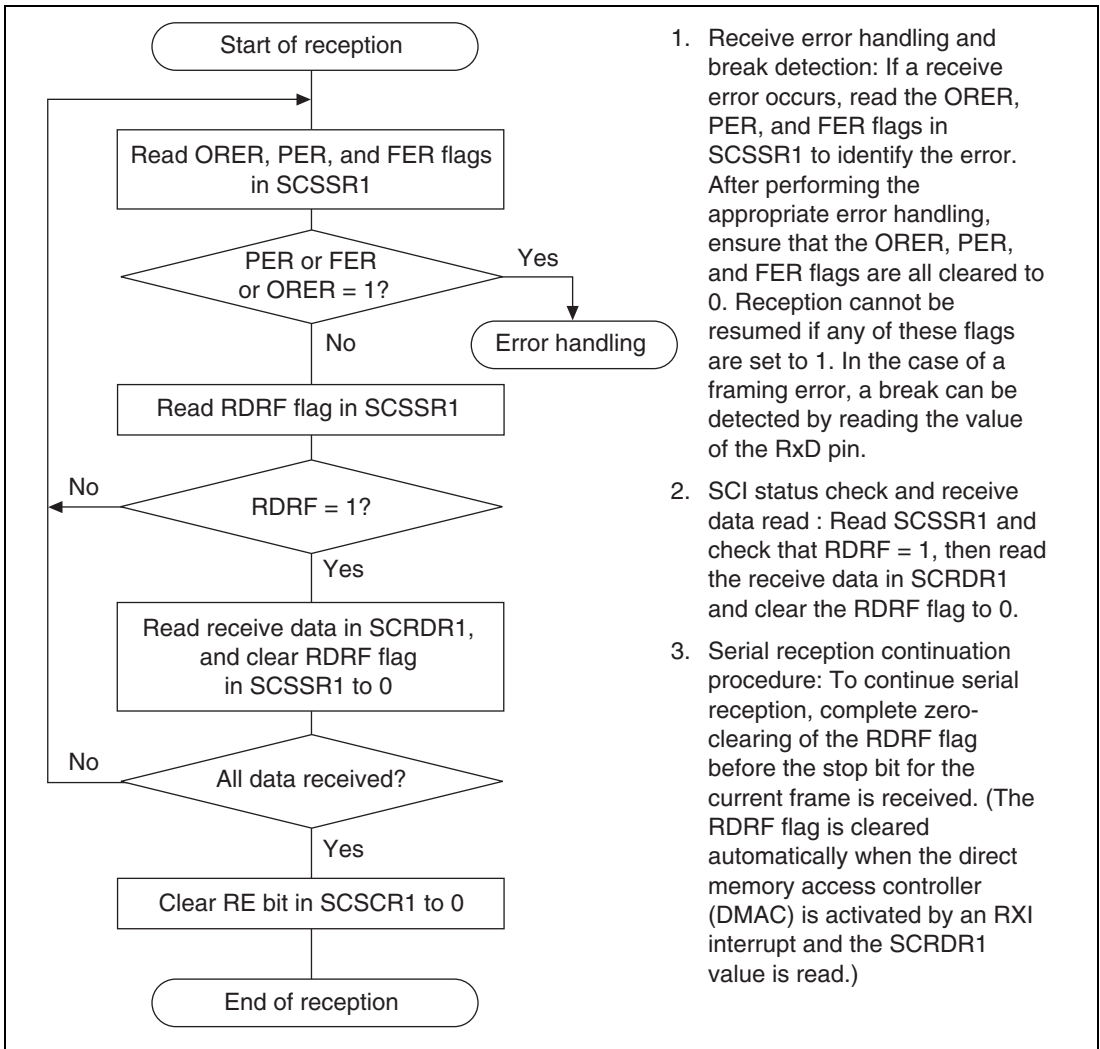


Figure 15.10 Sample Serial Reception Flowchart (1)

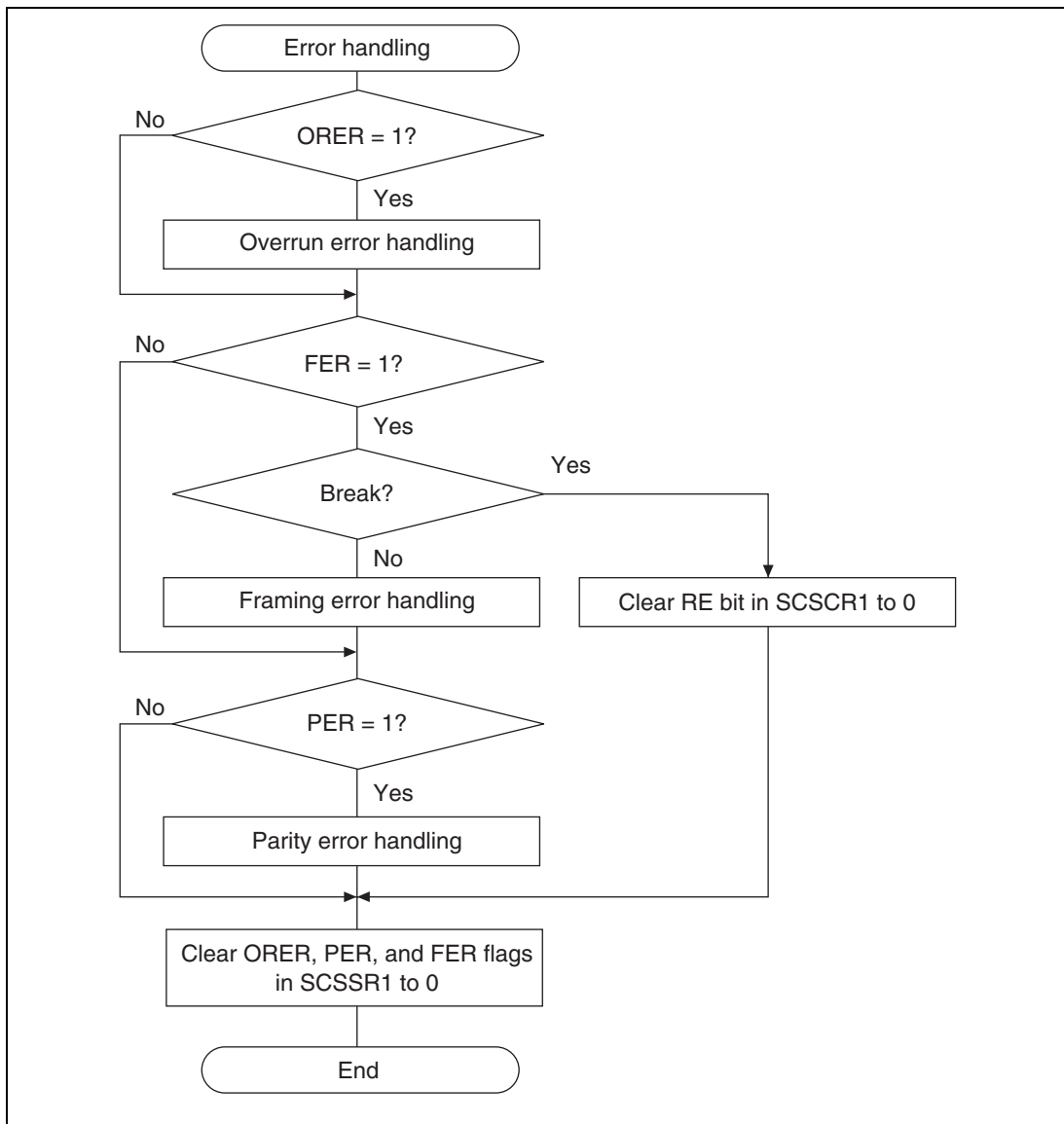


Figure 15.10 Sample Serial Reception Flowchart (2)

In serial reception, the SCI operates as described below.

1. The SCI monitors the transmission line, and if a 0 start bit is detected, performs internal synchronization and starts reception.
2. The received data is stored in SCRSR1 in LSB-to-MSB order.
3. The parity bit and stop bit are received.

After receiving these bits, the SCI carries out the following checks.

- a. Parity check: The SCI checks whether the number of 1-bits in the receive data agrees with the parity (even or odd) set in the O/\bar{E} bit in SCSMR1.
- b. Stop bit check: The SCI checks whether the stop bit is 1. If there are two stop bits, only the first is checked.
- c. Status check: The SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from SCRSR1 to SCRDR1.

If all the above checks are passed, the RDRF flag is set to 1, and the receive data is stored in SCRDR1.

If a receive error is detected in the error check, the operation is as shown in table 15.11.

Note: No further receive operations can be performed when a receive error has occurred. Also note that the RDRF flag is not set to 1 in reception, and so the error flags must be cleared to 0.

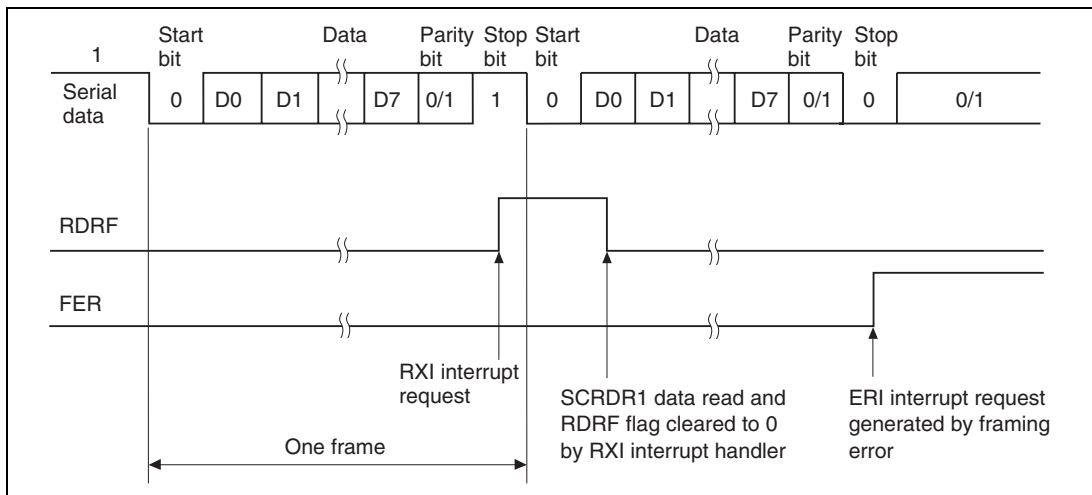
4. If the EIO bit in SCSPTR1 is cleared to 0 and the RIE bit in SCSCR1 is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated.

If the RIE bit in SCSCR1 is set to 1 when the ORER, PER, or FER flag changes to 1, a receive-error interrupt (ERI) request is generated. A receive-data-full request is always output to the DMAC when the RDRF flag changes to 1.

Table 15.11 Receive Error Conditions

| Receive Error | Abbreviation | Condition | Data Transfer |
|---------------|--------------|---|---|
| Overrun error | ORER | Reception of next data is completed while RDRF flag in SCSSR1 is set to 1 | Receive data is not transferred from SCRSR1 to SCRDR1 |
| Framing error | FER | Stop bit is 0 | Receive data is transferred from SCRSR1 to SCRDR1 |
| Parity error | PER | Received data parity differs from that (even or odd) set in SCSMR1 | Receive data is transferred from SCRSR1 to SCRDR1 |

Figure 15.11 shows an example of the operation for reception in asynchronous mode.



**Figure 15.11 Example of SCI Receive Operation
(Example with 8-Bit Data, Parity, One Stop Bit)**

15.3.3 Multiprocessor Communication Function

The multiprocessor communication function performs serial communication using a multiprocessor format, in which a multiprocessor bit is added to the transfer data, in asynchronous mode. Use of this function enables data transfer to be performed among a number of processors sharing a serial transmission line.

When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code.

The serial communication cycle consists of two cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with the multiprocessor bit set to 1. It then sends transmit data as data with the multiprocessor bit cleared to 0.

The receiving station skips the data until data with the multiprocessor bit set to 1 is sent.

When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip the data until data with a 1 multiprocessor bit is again received. In this way, data communication is carried out among a number of processors.

Figure 15.12 shows an example of inter-processor communication using a multiprocessor format.

Note: Even when this LSI has received data with a 0 multiprocessor bit that was meant to be sent to another station, the RDRF flag in SCSSR1 is set to 1. When the RDRF flag in SCSSR1 is set to 1, the exception handling routine reads the MPIE bit in SCSSR1, and skips the receive data if the MPIE bit is 1. Skipping of unnecessary data is achieved by collaborative operation with the exception handling routine.

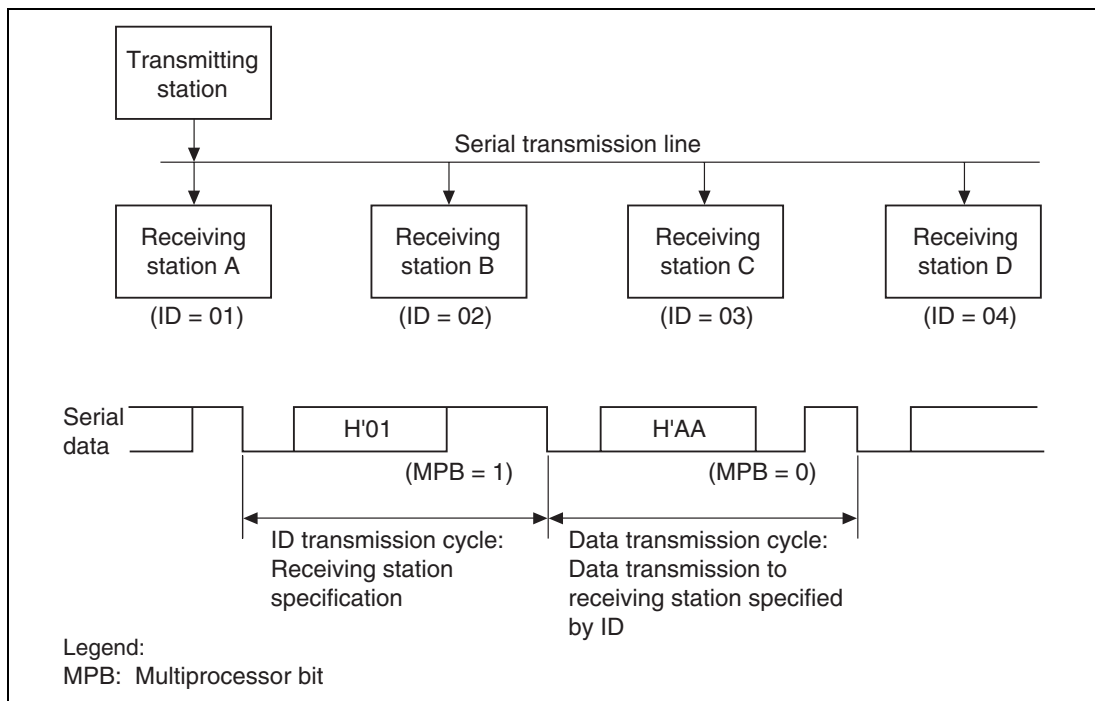


Figure 15.12 Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)

Data Transfer Formats

There are four data transfer formats. When the multiprocessor format is specified, the parity bit specification is invalid. For details, see table 15.10.

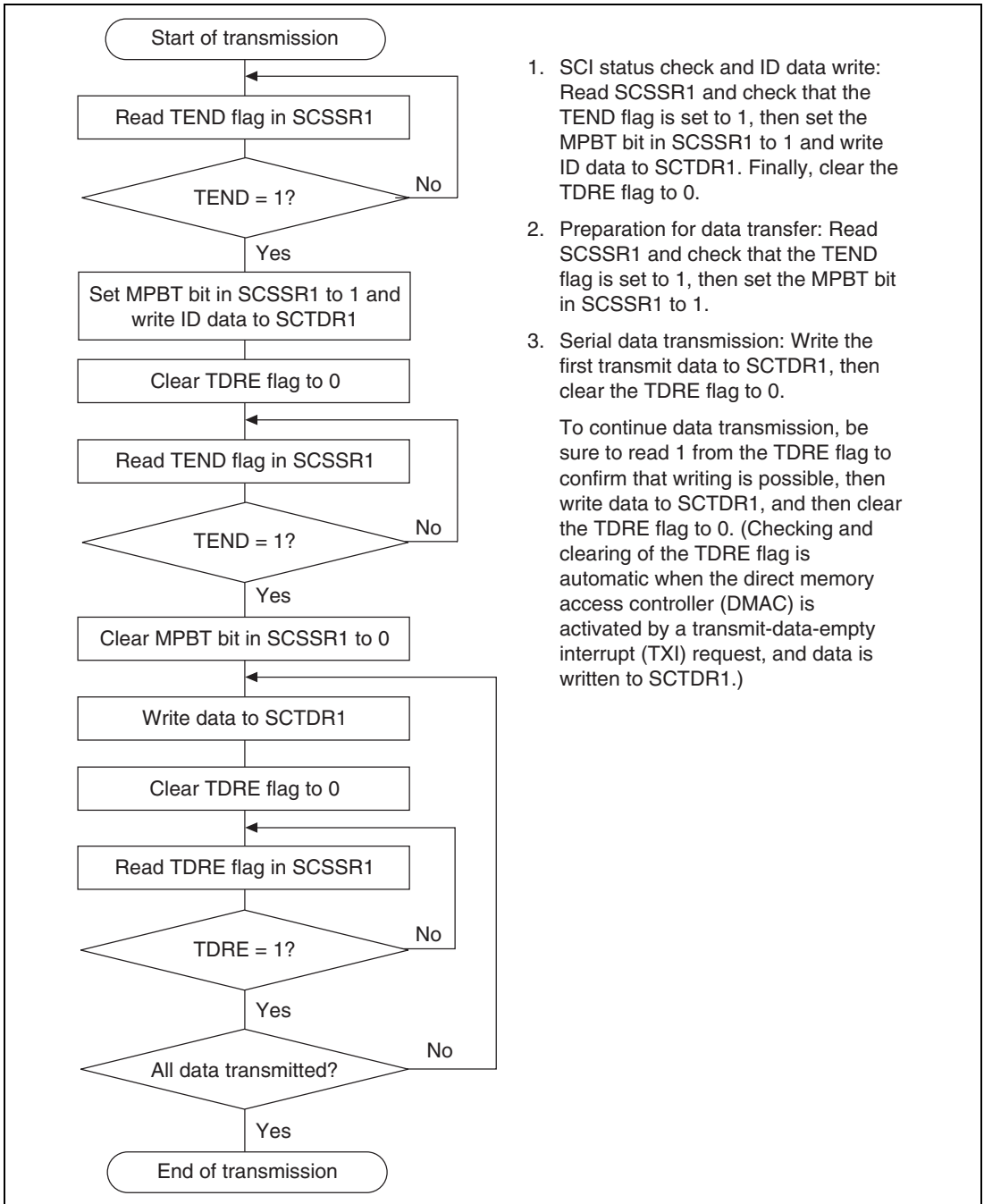
Clock

See the description under Clock in section 15.3.2, Operation in Asynchronous Mode.

Data Transfer Operations

Multiprocessor Serial Data Transmission: Figure 15.13 shows a sample flowchart for multiprocessor serial data transmission.

Use the following procedure for multiprocessor serial data transmission after enabling the SCI for transmission.



1. SCI status check and ID data write: Read SCSSR1 and check that the TEND flag is set to 1, then set the MPBT bit in SCSSR1 to 1 and write ID data to SCTDR1. Finally, clear the TDRE flag to 0.
2. Preparation for data transfer: Read SCSSR1 and check that the TEND flag is set to 1, then set the MPBT bit in SCSSR1 to 1.
3. Serial data transmission: Write the first transmit data to SCTDR1, then clear the TDRE flag to 0.

To continue data transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1, and then clear the TDRE flag to 0. (Checking and clearing of the TDRE flag is automatic when the direct memory access controller (DMAC) is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1.)

Figure 15.13 Sample Multiprocessor Serial Transmission Flowchart

In serial transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SCSSR1. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
2. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmission.

The serial transmit data is sent from the TxD pin in the following order.

- a. Start bit: One 0-bit is output.
 - b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
 - c. Multiprocessor bit: One multiprocessor bit (MPBT value) is output.
 - d. Stop bit(s): One or two 1-bits (stop bits) are output.
 - e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCI checks the TDRE flag at the timing for sending the stop bit. If the TDRE flag is set to 1, the TEND flag in SCSSR1 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output. If the TEIE bit in SCSCR1 is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.
 4. The SCI monitors the TDRE flag. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
 5. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE bit) in SCSCR1 is set to 1 at this time, a transmit-data-empty interrupt (TXI) request is generated.

The order of transmission is the same as in step 2.

Figure 15.14 shows an example of SCI operation for transmission using a multiprocessor format.

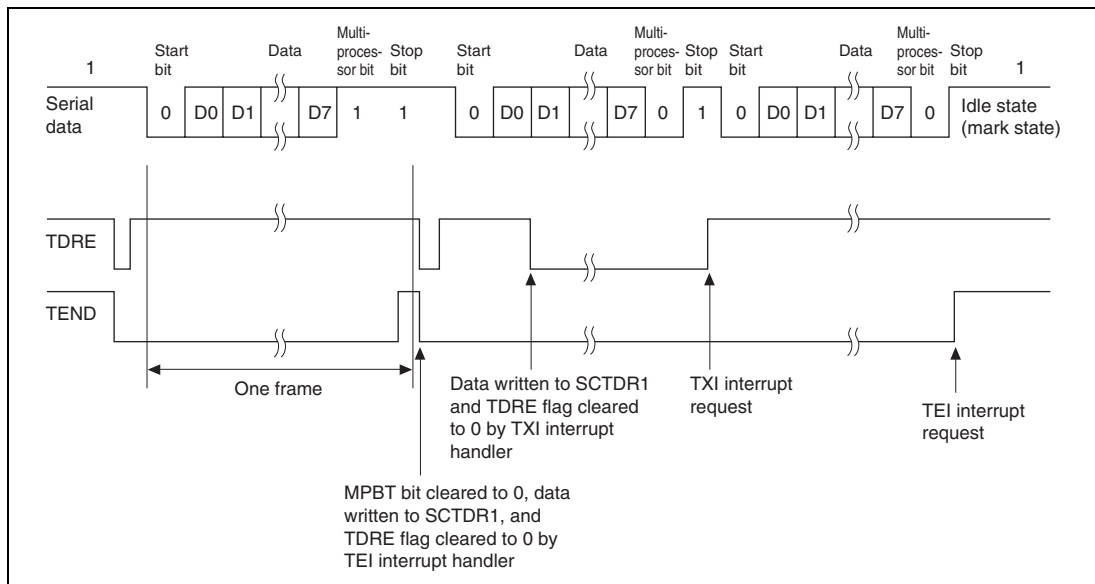


Figure 15.14 Example of SCI Transmit Operation (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)

Multiprocessor Serial Data Reception

- Method for determining whether an interrupt generated during receive operation is a multiprocessor interrupt

When an interrupt such as RXI occurs during receive operation using the on-chip SCI multiprocessor communication function, check the state of the MPIE bit in the SCSCR1 register as part of the interrupt handling routine.

- If the MPIE bit in the SCSCR1 register is set to 1
Ignore the received data.

Data with the multiprocessor bit (MPB) set to 0 and intended for another station was received, and the RDRF bit in the SCSCR1 register was set to 1. Therefore, clear the RDRF bit in the SCSCR1 register to 0.

- If the MPIE bit in the SCSCR1 register is cleared to 0

A multiprocessor interrupt indicating that data (ID) with the multiprocessor bit (MPB) set to 1 was received, or a receive data full interrupt (RXI) occurred when data with the multiprocessor bit (MPB) set to 0 and intended for this station was received.

- Method for determining whether received data is ID or data

Do not use the MPB bit in the SCSSR1 register for software processing.

When using software processing to determine whether received data is ID (MPB = 1) or data (MPB = 0), use a procedure such as saving a user-defined flag in memory to indicate receive start.

Figure 15.15 shows a flowchart of a sample software workaround.

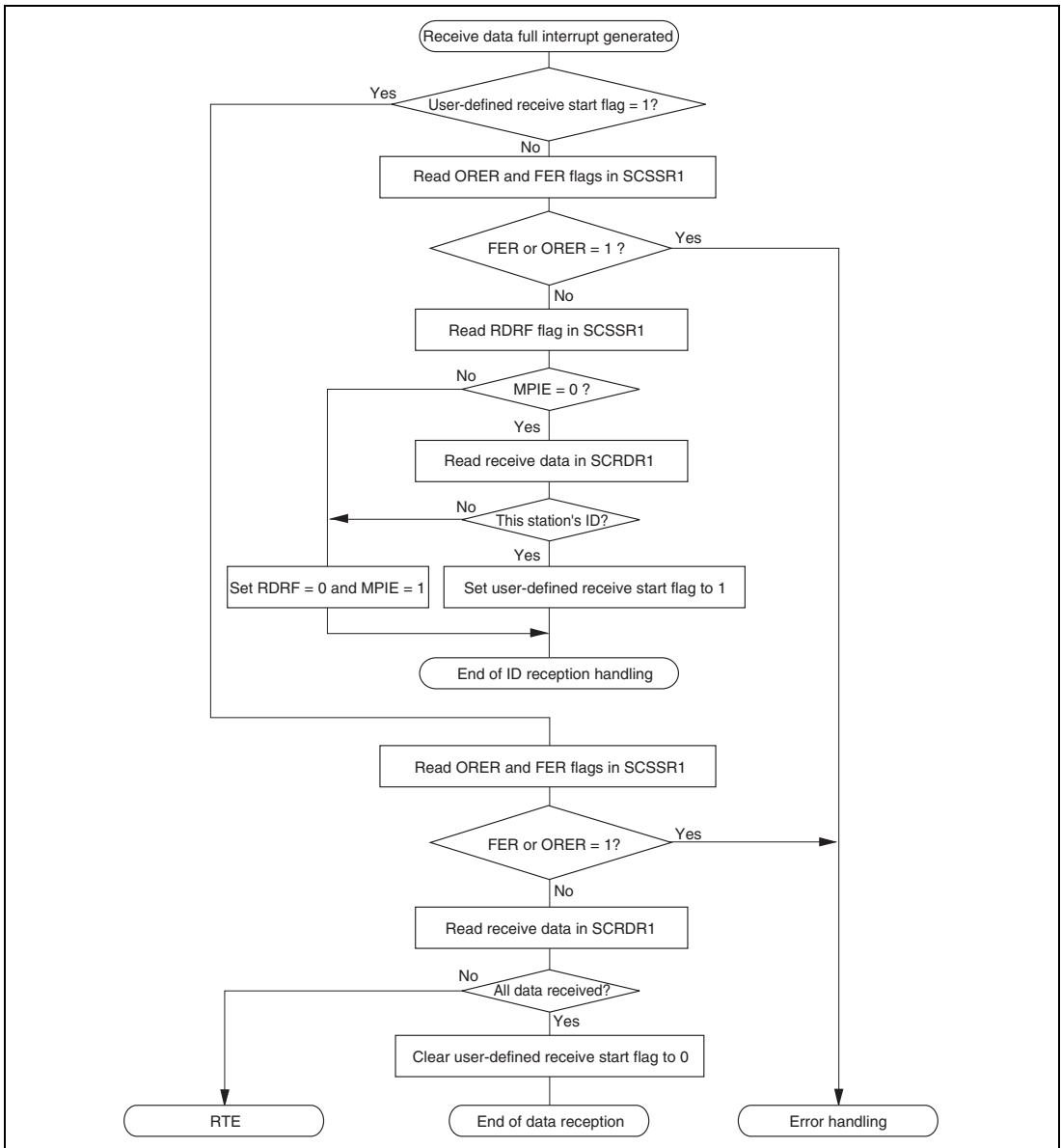


Figure 15.15 Sample Flowchart of Multiprocessor Serial Reception with Interrupt Generation

Figure 15.16 shows a sample flowchart of multiprocessor serial reception. To perform multiprocessor serial reception, first enable the SCI for data reception and then follow the procedure shown below.

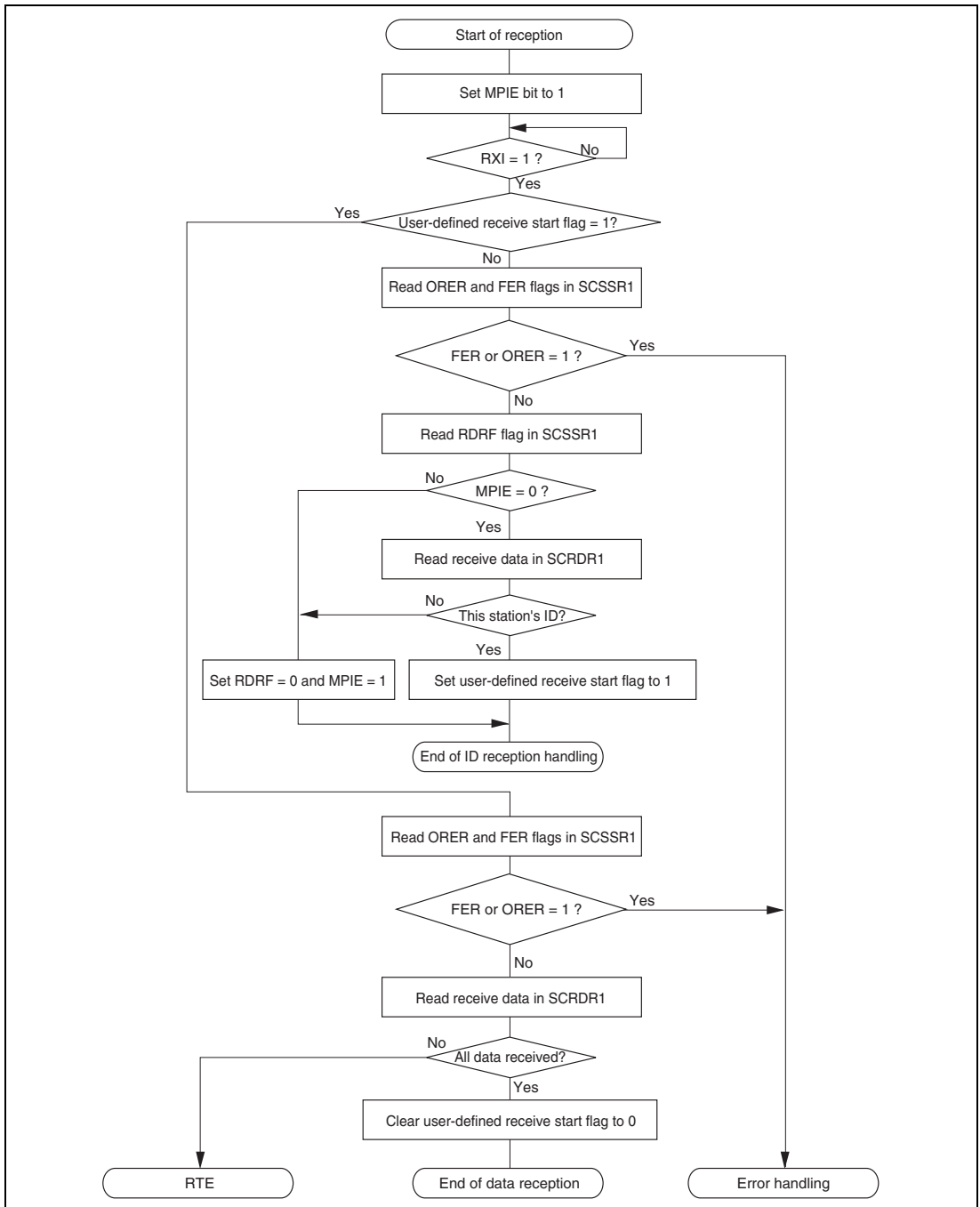


Figure 15.16 Sample Multiprocessor Serial Reception Flowchart (1)

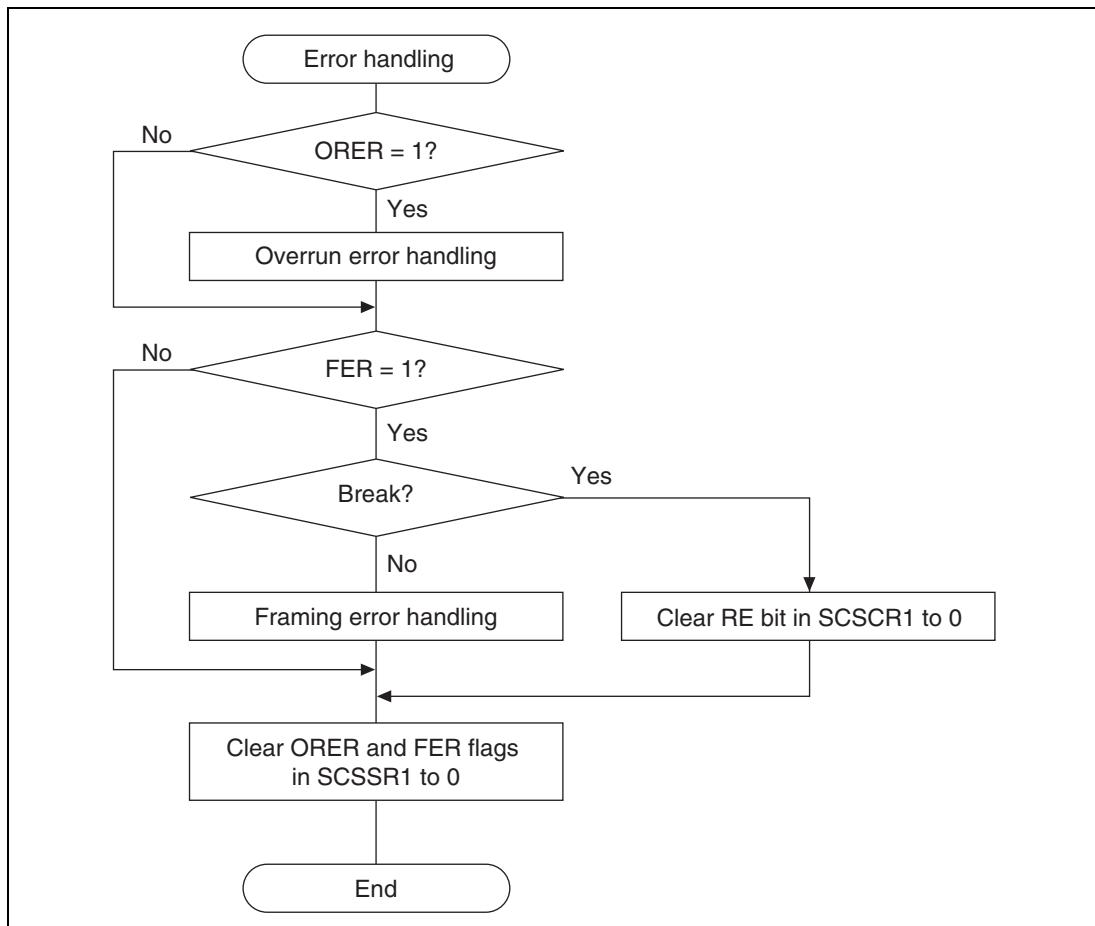
**Figure 15.16 Sample Multiprocessor Serial Reception Flowchart (2)**

Figure 15.17 shows an example of SCI operation for multiprocessor format reception.

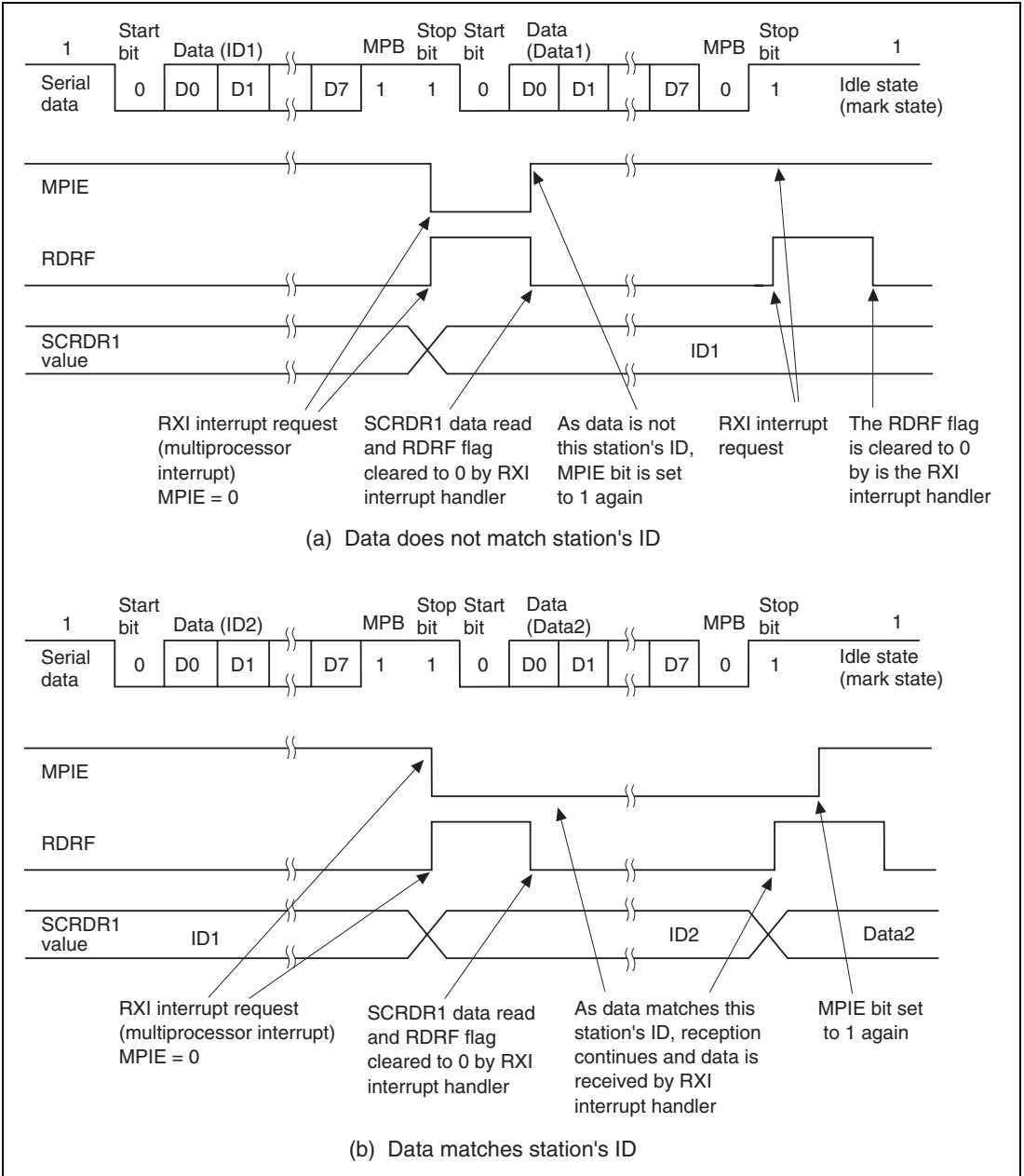


Figure 15.17 Example of SCI Receive Operation
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)

In multiprocessor mode serial reception, the SCI operates as described below.

1. The SCI monitors the transmission line, and if a 0 start bit is detected, performs internal synchronization and starts reception.
2. The received data is stored in SCRSR1 in LSB-to-MSB order.
3. If the MPIE bit is 1, MPIE is cleared to 0 when a 1 is received in the multiprocessor bit position. If the multiprocessor bit is 0, the MPIE bit is not changed.
4. If the MPIE bit is 0, RDRF is checked at the stop bit position, and if RDRF is 1 the overrun error bit is set. If the stop bit is not 0, the framing error bit is set. If RDRF is 0, the value in SCRSR1 is transferred to SCRDR1, and if the stop bit is 0, RDRF is set to 1.

15.3.4 Operation in Synchronous Mode

In synchronous mode, data is transmitted or received in synchronization with clock pulses, making it suitable for high-speed serial communication.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 15.18 shows the general format for synchronous serial communication.

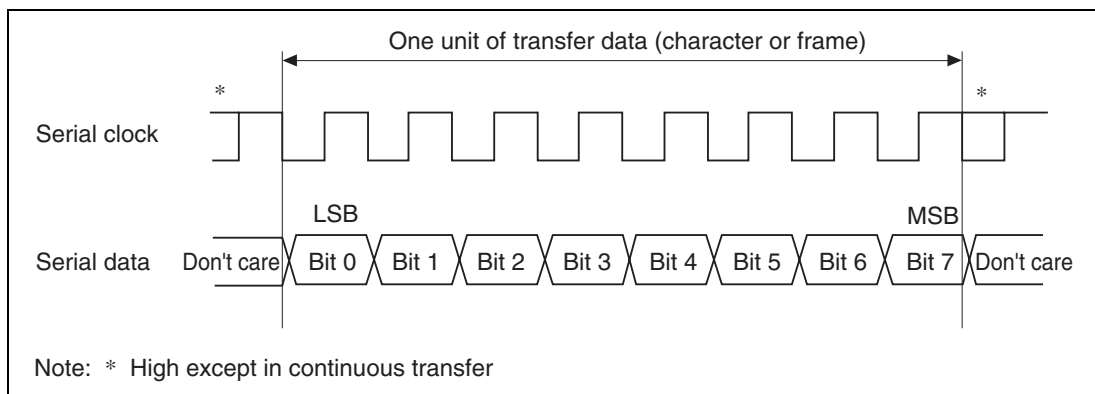


Figure 15.18 Data Format in Synchronous Communication

In synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. Data confirmation is guaranteed at the rising edge of the serial clock.

In serial communication, one character consists of data output starting with the LSB and ending with the MSB. After the MSB is output, the transmission line holds the MSB state.

In synchronous mode, the SCI receives data in synchronization with the falling edge of the serial clock.

Data Transfer Format

A fixed 8-bit data format is used. No parity or multiprocessor bits are added.

Clock

Either an internal clock generated by the on-chip baud rate generator or an external serial clock input at the SCK pin can be selected, according to the setting of the C/\bar{A} bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1. For details of SCI clock source selection, see table 15.9.

When the SCI is operated on an internal clock, the serial clock is output from the SCK pin.

Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. In reception only, if an on-chip clock source is selected, clock pulses are output while RE = 1. When the last data is received, RE should be cleared to 0 before the end of bit 7.

Data Transfer Operations

SCI Initialization (Synchronous Mode): Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR1 to 0, then initialize the SCI as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and SCTSR1 is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of SCRDR1.

Figure 15.19 shows a sample SCI initialization flowchart.

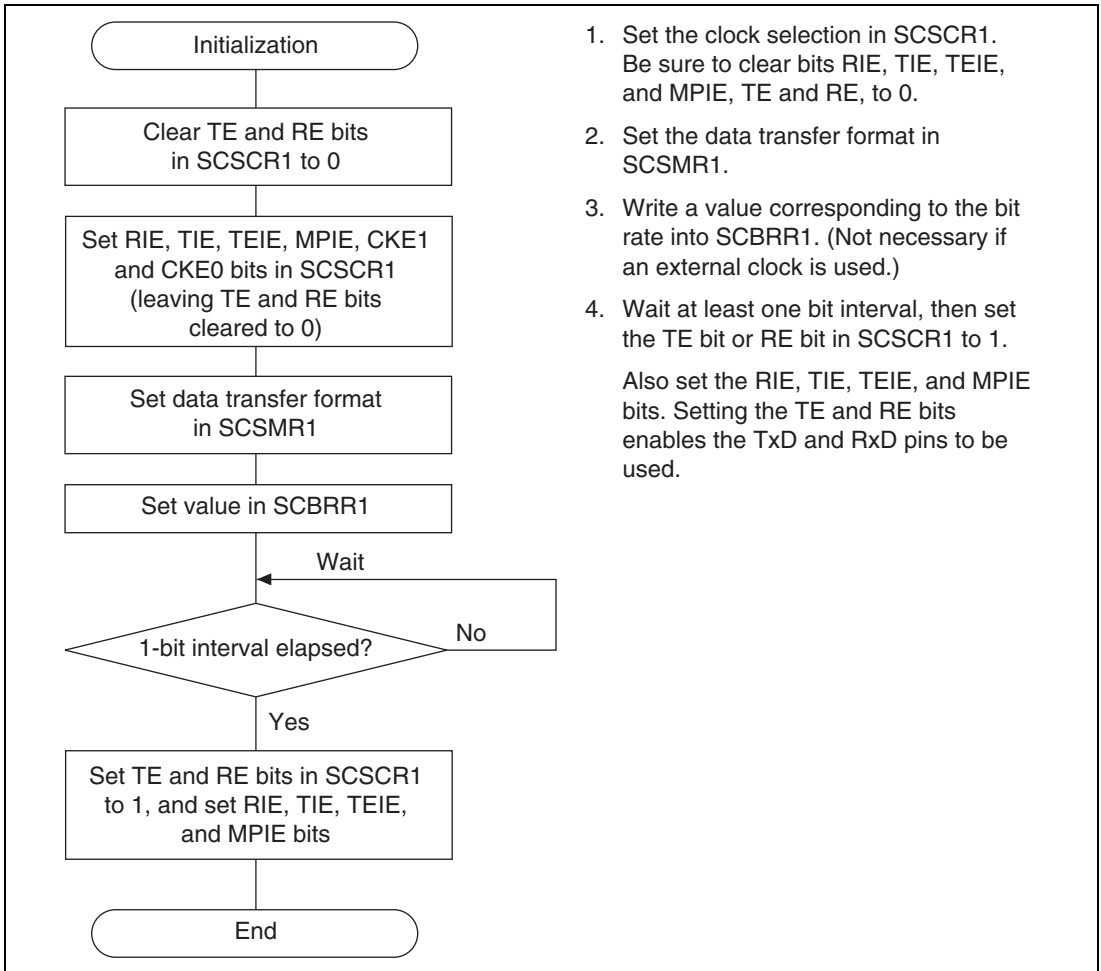
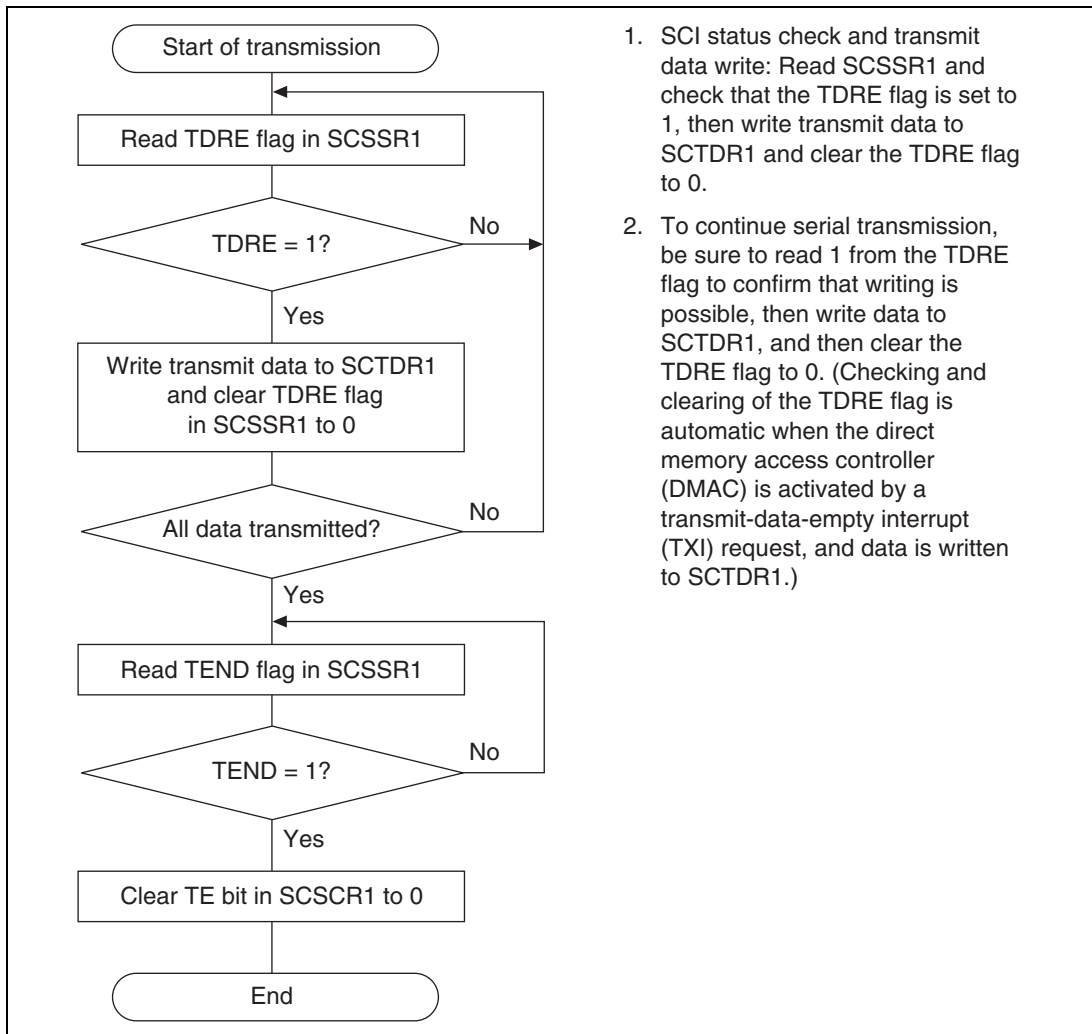


Figure 15.19 Sample SCI Initialization Flowchart

Serial Data Transmission (Synchronous Mode): Figure 15.20 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCI for transmission.



1. SCI status check and transmit data write: Read SCSSR1 and check that the TDRE flag is set to 1, then write transmit data to SCTDR1 and clear the TDRE flag to 0.
2. To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1, and then clear the TDRE flag to 0. (Checking and clearing of the TDRE flag is automatic when the direct memory access controller (DMAC) is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1.)

Figure 15.20 Sample Serial Transmission Flowchart

In serial transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SCSSR1. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
2. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) request is generated.

When clock output mode has been set, the SCI outputs 8 serial clock pulses. When use of an external clock has been specified, data is output synchronized with the input clock.

The serial transmit data is sent from the TxD pin starting with the LSB (bit 0) and ending with the MSB (bit 7).

3. The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).

If the TDRE flag is cleared to 0, data is transferred from SCTDR1 to SCTSR1, and serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SCSSR1 is set to 1, the MSB (bit 7) is sent, and the TxD pin maintains its state.

If the TEIE bit in SCSCR1 is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.

4. After completion of serial transmission, the SCK pin is fixed high.

Figure 15.21 shows an example of SCI operation in transmission.

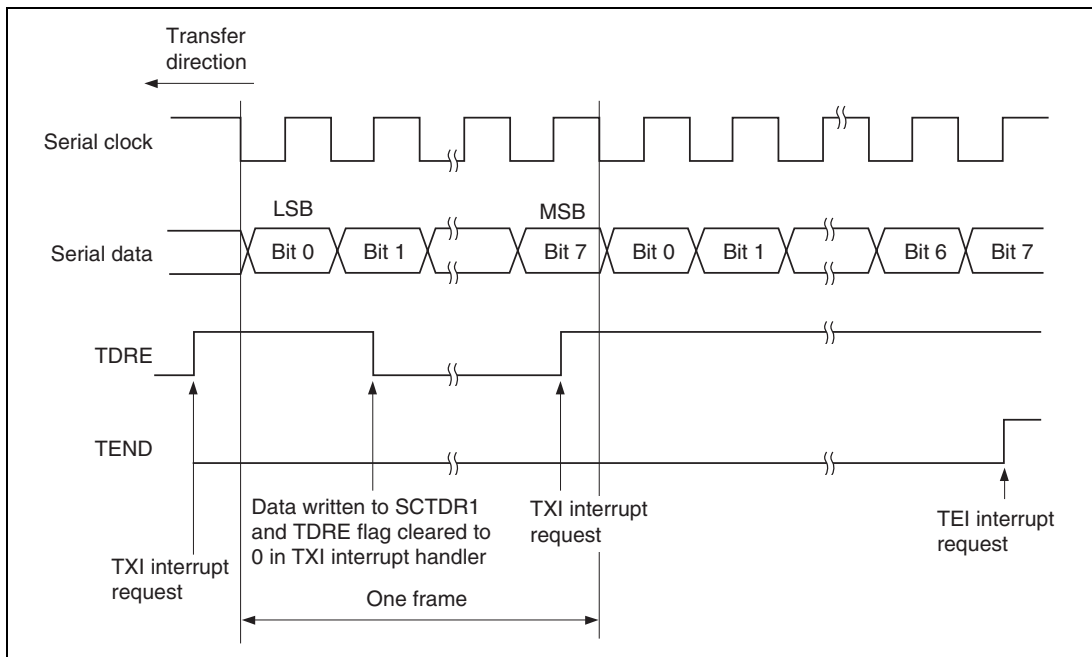
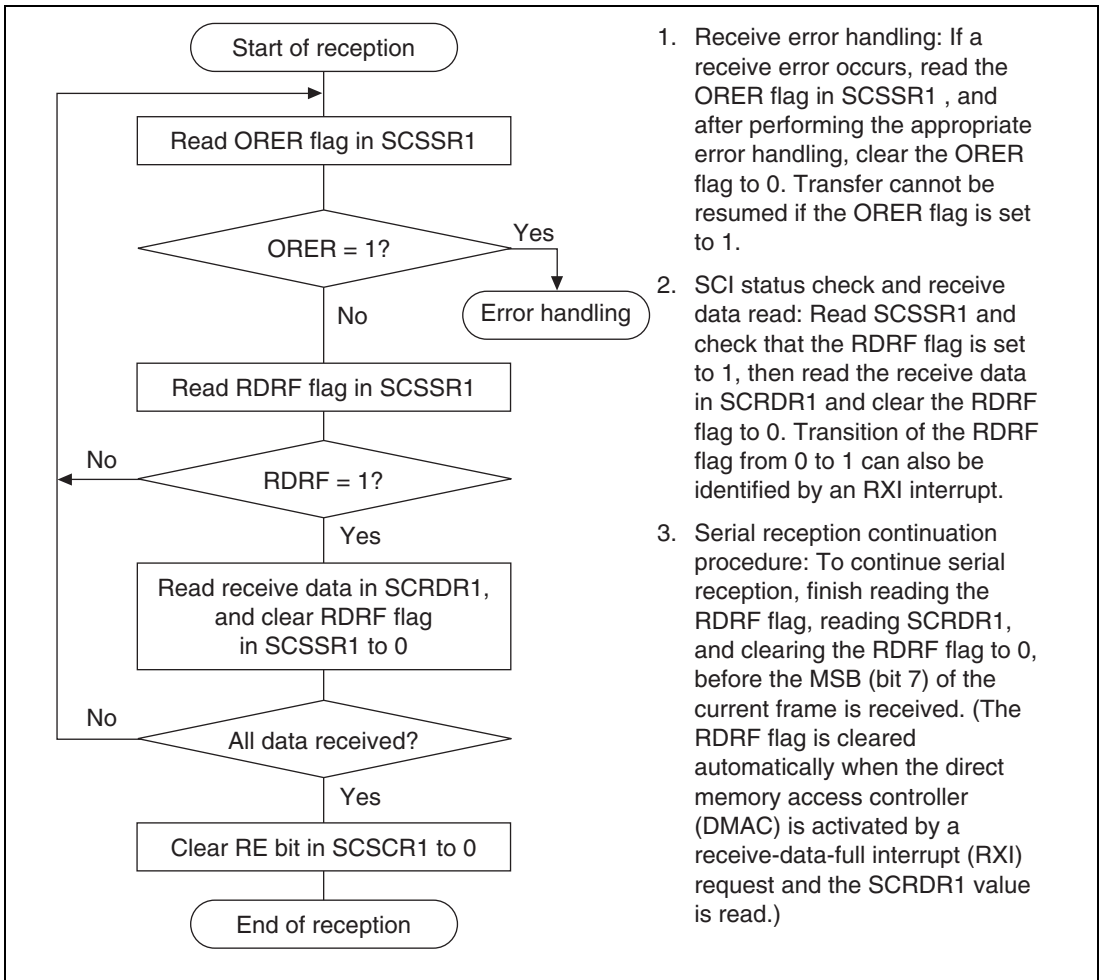


Figure 15.21 Example of SCI Transmit Operation

Serial Data Reception (Synchronous Mode): Figure 15.22 shows a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling the SCI for reception.

When changing the operating mode from asynchronous to synchronous, be sure to check that the ORER, PER, and FER flags are all cleared to 0. The RDRF flag will not be set if the FER or PER flag is set to 1, and neither transmit nor receive operations will be possible.



1. Receive error handling: If a receive error occurs, read the ORER flag in SCSSR1, and after performing the appropriate error handling, clear the ORER flag to 0. Transfer cannot be resumed if the ORER flag is set to 1.
2. SCI status check and receive data read: Read SCSSR1 and check that the RDRF flag is set to 1, then read the receive data in SCRDR1 and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
3. Serial reception continuation procedure: To continue serial reception, finish reading the RDRF flag, reading SCRDR1, and clearing the RDRF flag to 0, before the MSB (bit 7) of the current frame is received. (The RDRF flag is cleared automatically when the direct memory access controller (DMAC) is activated by a receive-data-full interrupt (RXI) request and the SCRDR1 value is read.)

Figure 15.22 Sample Serial Reception Flowchart (1)

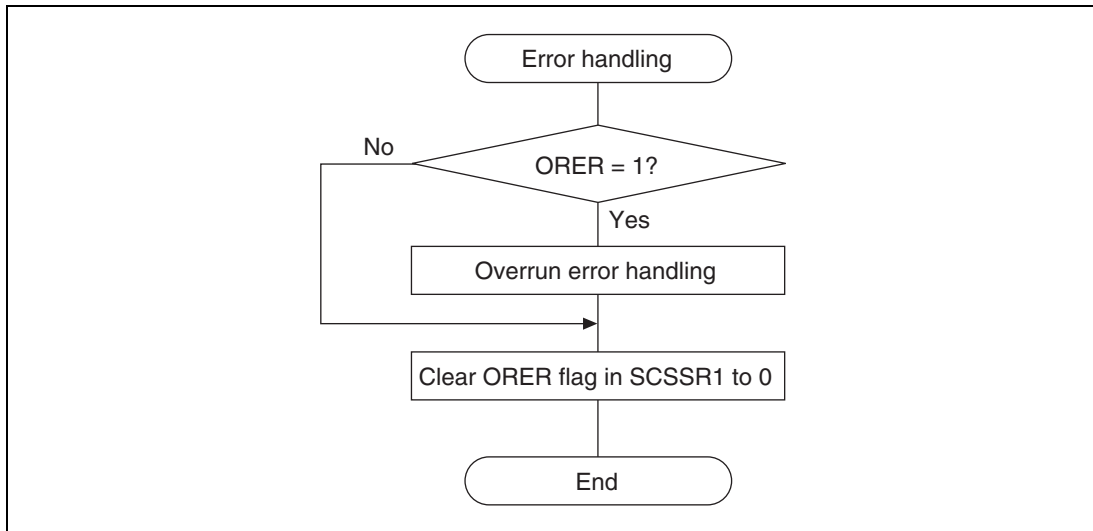


Figure 15.22 Sample Serial Reception Flowchart (2)

In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with serial clock input or output.
2. The received data is stored in SCRSR1 in LSB-to-MSB order.

After reception, the SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from SCRSR1 to SCRDR1.

If this check is passed, the RDRF flag is set to 1, and the receive data is stored in SCRDR1. If a receive error is detected in the error check, the operation is as shown in table 15.11.

Neither transmit nor receive operations can be performed subsequently when a receive error has been found in the error check.

Also, as the RDRF flag is not set to 1 when receiving, the flag must be cleared to 0.

3. If the RIE bit in SCRSR1 is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated. If the RIE bit in SCRSR1 is set to 1 when the ORER flag changes to 1, a receive-error interrupt (ERI) request is generated.

Figure 15.23 shows an example of SCI operation in reception.

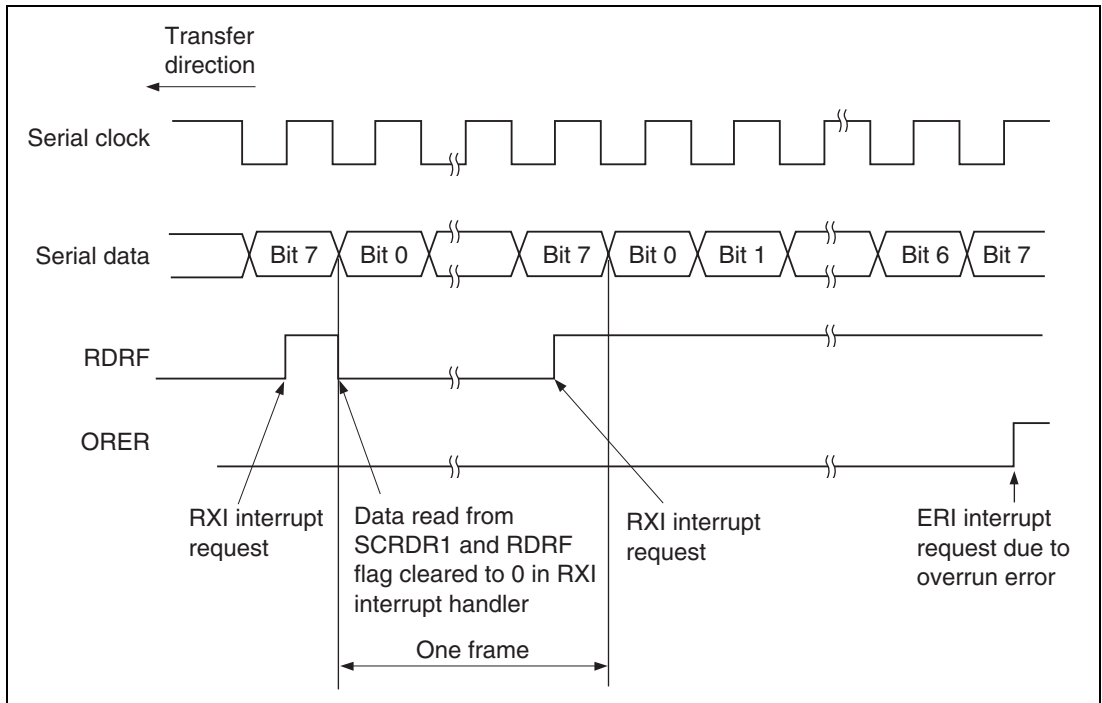
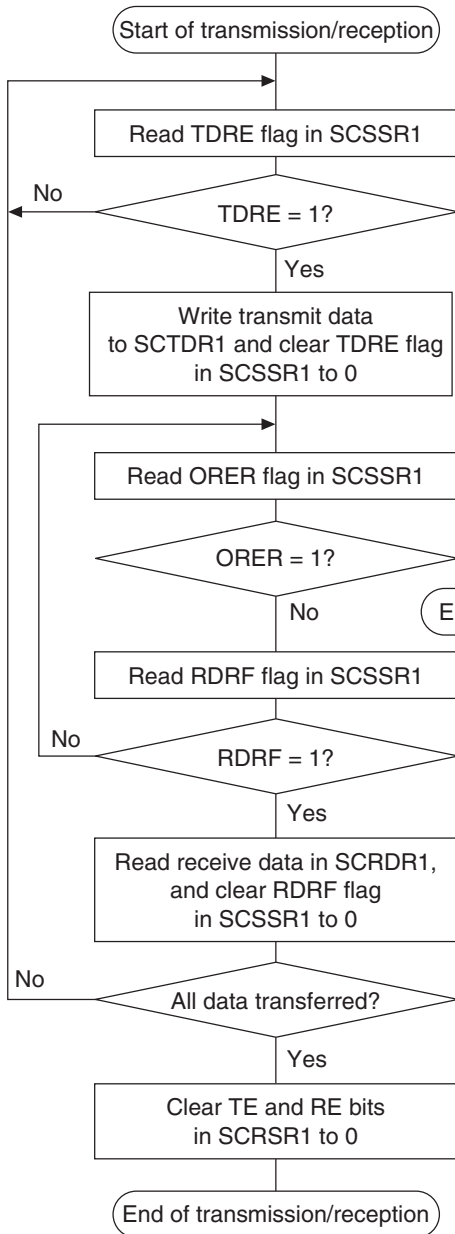


Figure 15.23 Example of SCI Receive Operation

Simultaneous Serial Data Transmission and Reception (Synchronous Mode): Figure 15.24 shows a sample flowchart for simultaneous serial transmit and receive operations.

Use the following procedure for simultaneous serial data transmit and receive operations after enabling the SCI for transmission and reception.



1. SCI status check and transmit data write:
Read SCSSR1 and check that the TDRE flag is set to 1, then write transmit data to SCTDR1 and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.
2. Receive error handling:
If a receive error occurs, read the ORER flag in SCSSR1, and after performing the appropriate error handling, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.
3. SCI status check and receive data read:
Read SCSSR1 and check that the RDRF flag is set to 1, then read the receive data in SCRDR1 and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
4. Serial transmission/reception continuation procedure:
To continue serial transmission/reception, finish reading the RDRF flag, reading SCRDR1, and clearing the RDRF flag to 0, before the MSB (bit 7) of the current frame is received. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1 and clear the TDRE flag to 0.

(Checking and clearing of the TDRE flag is automatic when the DMAC is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1. Similarly, the RDRF flag is cleared automatically when the DMAC is activated by a receive-data-full interrupt (RXI) request and the SCRDR1 value is read.)

Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1.

Figure 15.24 Sample Flowchart for Serial Data Transmission and Reception

15.4 SCI Interrupt Sources and DMAC

The SCI has four interrupt sources: the transmit-end interrupt (TEI) request, receive-error interrupt (ERI) request, receive-data-full interrupt (RXI) request, and transmit-data-empty interrupt (TXI) request.

Table 15.12 shows the interrupt sources and their relative priorities. Individual interrupt sources can be enabled or disabled with the TIE, RIE, and TEIE bits in SCSSR1, and the EIO bit in SCSPTR1. Each kind of interrupt request is sent to the interrupt controller independently.

When the TDRE flag in the serial status register (SCSSR1) is set to 1, a TDR-empty request is generated separately from the interrupt request. A TDR-empty request can activate the direct memory access controller (DMAC) to perform data transfer. The TDRE flag is cleared to 0 automatically when a write to the transmit data register (SCTDR1) is performed by the DMAC.

When the RDRF flag in SCSSR1 is set to 1, an RDR-full request is generated separately from the interrupt request. An RDR-full request can activate the DMAC to perform data transfer.

The RDRF flag is cleared to 0 automatically when a receive data register (SCRDR1) read is performed by the DMAC.

When the ORER, FER, or PER flag in SCSSR1 is set to 1, an ERI interrupt request is generated. The DMAC cannot be activated by an ERI interrupt request. When receive data processing is to be carried out by the DMAC and receive error handling is to be performed by means of an interrupt to the CPU, set the RIE bit to 1 and also set the EIO bit in SCSPTR1 to 1 so that an interrupt error occurs only for a receive error. If the EIO bit is cleared to 0, interrupts to the CPU will be generated even during normal data reception.

When the TEND flag in SCSSR1 is set to 1, a TEI interrupt request is generated. The DMAC cannot be activated by a TEI interrupt request.

A TXI interrupt indicates that transmit data can be written, and a TEI interrupt indicates that the transmit operation has ended.

Table 15.12 SCI Interrupt Sources

| Interrupt Source | Description | DMAC Activation | Priority on Reset Release |
|------------------|-------------------------------------|-----------------|---------------------------|
| ERI | Receive error (ORER, FER, or PER) | Not possible | High |
| RXI | Receive data register full (RDRF) | Possible | ↑ ↓ |
| TXI | Transmit data register empty (TDRE) | Possible | |
| TEI | Transmit end (TEND) | Not possible | Low |

See section 5, Exceptions, for the priority order and relation to non-SCI interrupts.

15.5 Usage Notes

The following points should be noted when using the SCI.

SCTDR1 Writing and the TDRE Flag: The TDRE flag in SCSSR1 is a status flag that indicates that transmit data has been transferred from SCTDR1 to SCTSR1. When the SCI transfers data from SCTDR1 to SCTSR1, the TDRE flag is set to 1.

Data can be written to SCTDR1 regardless of the state of the TDRE flag. However, if new data is written to SCTDR1 when the TDRE flag is cleared to 0, the data stored in SCTDR1 will be lost since it has not yet been transferred to SCTSR1. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to SCTDR1.

Simultaneous Multiple Receive Errors: If a number of receive errors occur at the same time, the state of the status flags in SCSSR1 is as shown in table 15.13. If there is an overrun error, data is not transferred from SCRSR1 to SCRDR1, and the receive data is lost.

Table 15.13 SCSSR1 Status Flags and Transfer of Receive Data

| Receive Errors | SCSSR1 Status Flags | | | | Receive Data Transfer |
|--|---------------------|------|-----|-----|-----------------------|
| | RDRF | ORER | FER | PER | SCRSR1 → SCRDR1 |
| 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 |

Legend:

O: Receive data is transferred from SCRSR1 to SCRDR1.

X: Receive data is not transferred from SCRSR1 to SCRDR1.

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 the FER flag is set and the parity error flag (PER) may also be set. Note that the SCI receiver continues to operate in the break state, so if the FER flag is cleared to 0 it will be set to 1 again.

Sending a Break Signal: The input/output condition and level of the TxD pin are determined by bits SPB0IO and SPB0DT in the serial port register (SCSPTR1). This feature can be used to send a break signal.

After the serial transmitter is initialized, the TxD pin function is not selected and the value of the SPB0DT bit substitutes for the mark state until the TE bit is set to 1 (i.e. transmission is enabled). The SPB0IO and SPB0DT bits should therefore be set to 1 (designating output and high level) beforehand.

To send a break signal during serial transmission, clear the SPB0DT bit to 0 (designating low level), then clear the TE bit to 0 (halting transmission). When the TE bit is cleared to 0, the transmitter is initialized regardless of its current state, and the TxD pin becomes an output port outputting the value 0.

Handling of TEND Flag and TE Bit: The TEND flag is set to 1 when the stop bit of the final data segment is transmitted. If the TE bit is cleared immediately after confirming that the TEND flag was set, transmission may not complete properly because stop bit transmission processing is still underway. Therefore, wait at least 0.5 serial clock cycles (1.5 cycles if two stop bits are used) after confirming that the TEND flag was set before clearing the TE bit.

Receive Error Flags and Transmit Operations (Synchronous Mode Only): Transmission cannot be started when a receive error flag (ORER, PER, or FER) is set to 1, even if the TDRE flag is set to 1. Be sure to clear the receive error flags to 0 before starting transmission.

Note also that the receive error flags are not cleared to 0 by clearing the RE bit to 0.

Receive Data Sampling Timing and Receive Margin in Asynchronous Mode: The SCI operates on a base clock with a frequency of 16 times the bit rate. In reception, the SCI synchronizes internally with the fall of the start bit, which it samples on the base clock. Receive data is latched at the rising edge of the eighth base clock pulse. The timing is shown in figure 15.25.

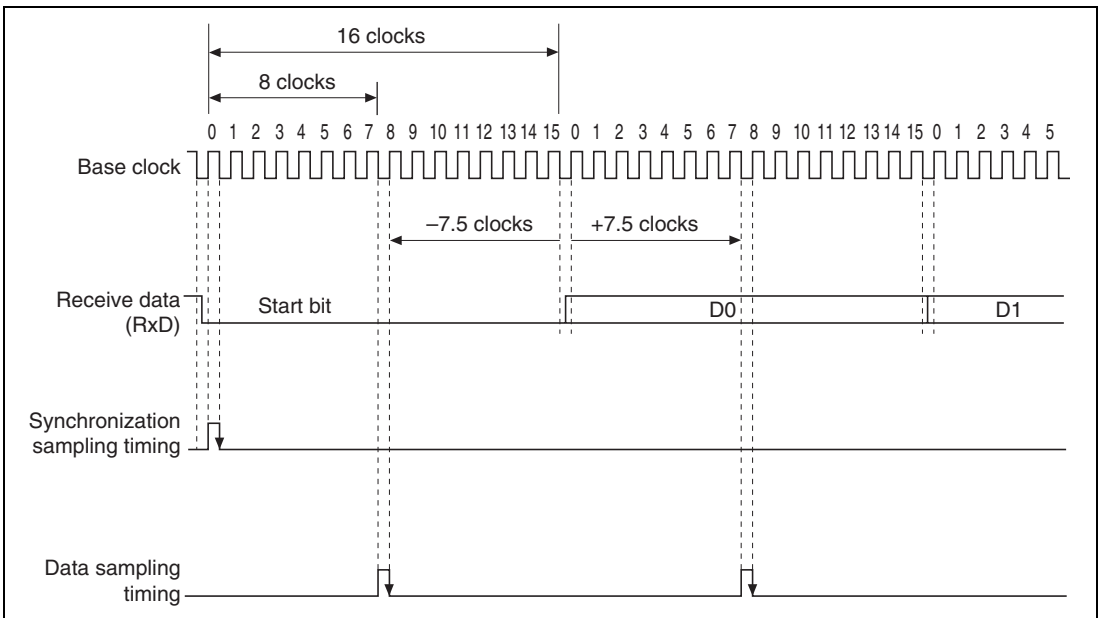


Figure 15.25 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

$$M = \left| \left(0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \cdot 100\% \quad \dots \quad (1)$$

- M: Receive margin (%)
 N: Ratio of clock frequency to bit rate ($N = 16$)
 D: Clock duty cycle ($D = 0$ to 1.0)
 L: Frame length ($L = 9$ to 12)
 F: Absolute deviation of clock frequency

From equation (1), if $F = 0$ and $D = 0.5$, the receive margin is 46.875%, as given by equation (2).

When $D = 0.5$ and $F = 0$:

$$M = (0.5 - 1/(2 \times 16)) \times 100\% = 46.875\% \dots\dots\dots (2)$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

When Using the DMAC: When an external clock source is used as the serial clock, the transmit clock should not be input until at least 5 peripheral operating clock cycles after SCTDR1 is updated by the DMAC. Incorrect operation may result if the transmit clock is input within 4 cycles after SCTDR1 is updated. (See figure 15.26)

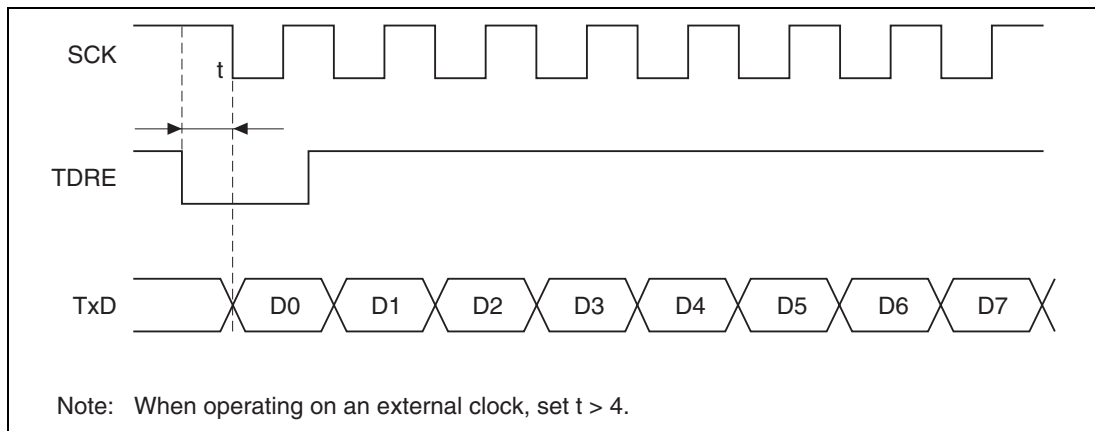


Figure 15.26 Example of Synchronous Transmission by DMAC

When SCRDR1 is read by the DMAC, be sure to set the SCI receive-data-full interrupt (RXI) as the activation source with bits RS3 to RS0 in CHCR.

When Using Synchronous External Clock Mode:

- Do not set TE or RE to 1 until at least 4 peripheral operating clock cycles after external clock SCK has changed from 0 to 1.
- Only set both TE and RE to 1 when external clock SCK is 1.

- In reception, note that if RE is cleared to 0 from 2.5 to 3.5 peripheral operating clock cycles after the rising edge of the RxD D7 bit SCK input, RDRF will be set to 1 but copying to SCRDR1 will not be possible.

When Using Synchronous Internal Clock Mode: In reception, note that if RE is cleared to zero 1.5 peripheral operating clock cycles after the rising edge of the RxD D7 bit SCK output, RDRF will be set to 1 but copying to SCRDR1 will not be possible.

When Using DMAC: When using the DMAC for transmission/reception, make a setting to suppress output of RXI and TXI interrupt requests to the interrupt controller. Even if a setting is made to output interrupt requests, interrupt requests to the interrupt controller will be cleared by the DMAC independently of the interrupt handling program.

Section 16 Serial Communication Interface with FIFO (SCIF)

16.1 Overview

This LSI is equipped with a single-channel serial communication interface with built-in FIFO buffers (Serial Communication Interface with FIFO: SCIF). The SCIF can perform asynchronous serial communication.

Sixteen-stage FIFO registers are provided for both transmission and reception, enabling fast, efficient, and continuous communication.

16.1.1 Features

SCIF features are listed below.

- Asynchronous serial communication

Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA).

There is a choice of 8 serial data transfer formats.

— Data length: 7 or 8 bits

— Stop bit length: 1 or 2 bits

— Parity: Even/odd/none

— Receive error detection: Parity, framing, and overrun errors

— Break detection: If the receive data following that in which a framing error occurred is also at the space “0” level, and there is a frame error, a break is detected. When a framing error occurs, a break can also be detected by reading the RxD2 pin level directly from the serial port register (SCSPTR2).

- Full-duplex communication capability

The transmitter and receiver are independent units, enabling transmission and reception to be performed simultaneously.

The transmitter and receiver both have a 16-stage FIFO buffer structure, enabling fast and continuous serial data transmission and reception.

- On-chip baud rate generator allows any bit rate to be selected.

- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK2 pin
- Four interrupt sources
There are four interrupt sources—transmit-FIFO-data-empty, break, receive-FIFO-data-full, and receive-error—that can issue requests independently.
- The DMA controller (DMAC) can be activated to execute a data transfer by issuing a DMA transfer request in the event of a transmit-FIFO-data-empty or receive-FIFO-data-full interrupt.
- When not in use, the SCIF can be stopped by halting its clock supply to reduce power consumption.
- Modem control functions ($\overline{\text{RTS2}}$ and $\overline{\text{CTS2}}$) are provided.
- The amount of data in the transmit/receive FIFO registers, and the number of receive errors in the receive data in the receive FIFO register, can be ascertained.
- A timeout error (DR) can be detected during reception.

16.1.2 Block Diagram

Figure 16.1 shows a block diagram of the SCIF.

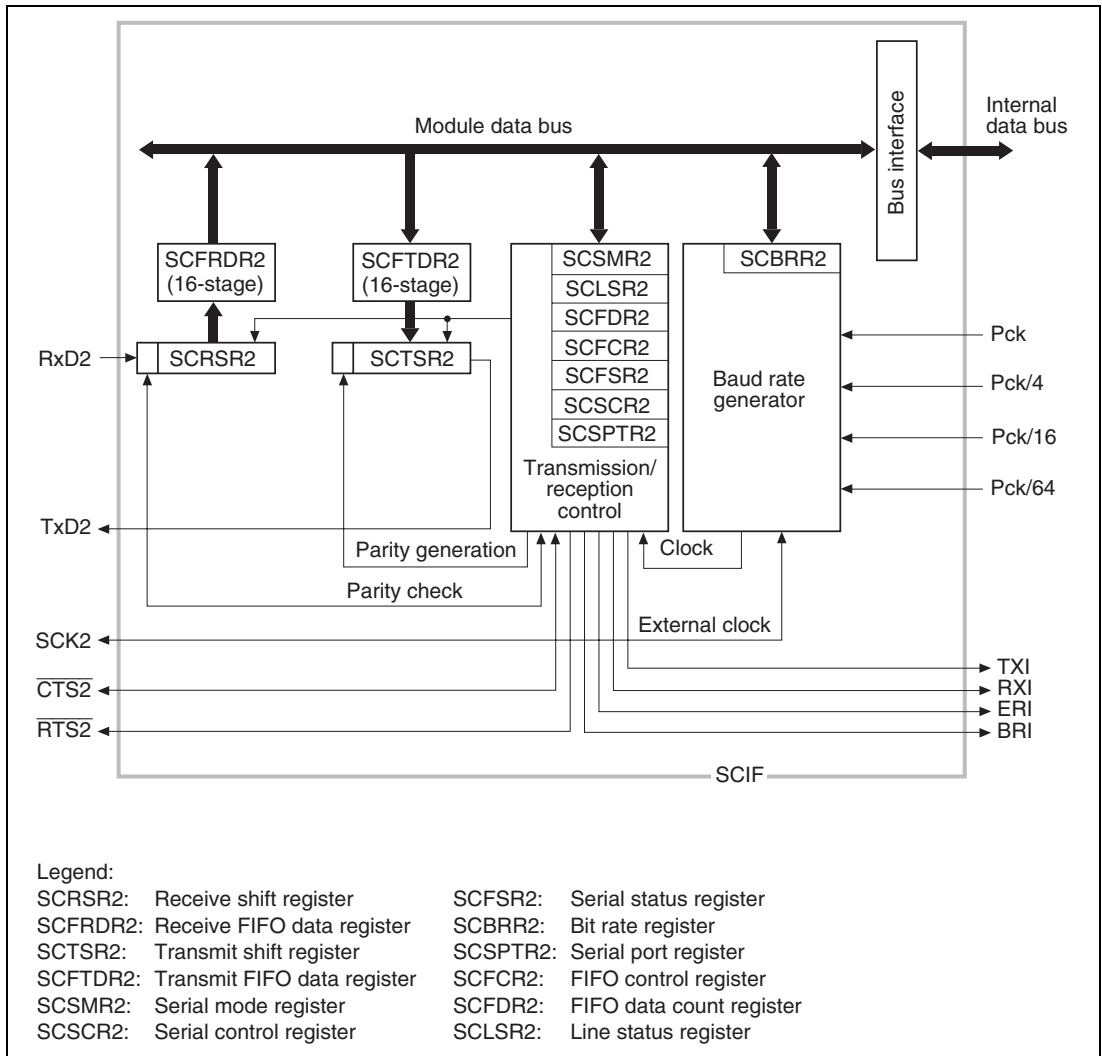


Figure 16.1 Block Diagram of SCIF

16.1.3 Pin Configuration

Table 16.1 shows the SCIF pin configuration.

Table 16.1 SCIF Pins

| Pin Name | Abbreviation | I/O | Function |
|-------------------|-------------------------------|--------|----------------------|
| Serial clock pin | MD0/SCK2 | I/O | Clock input/output |
| Receive data pin | MD2/RxD2 | Input | Receive data input |
| Transmit data pin | MD1/TxD2 | Output | Transmit data output |
| Modem control pin | MD7/ $\overline{\text{CTS2}}$ | I/O | Transmission enabled |
| Modem control pin | MD8/ $\overline{\text{RTS2}}$ | I/O | Transmission request |

Note: These pins function as the MD0, MD1, MD2, MD7, and MD8 mode input pins after a power-on reset. These pins are made to function as serial pins by performing SCIF operation settings with the TE, RE, CKE1, and CKE0 bits in SCSCR2 and the MCE bit in SCFCR2. Break state transmission and detection can be set in the SCIF's SCSPTR2 register.

16.1.4 Register Configuration

The SCIF has the internal registers shown in table 16.2. These registers are used to specify the data format and bit rate, and to perform transmitter/receiver control.

Table 16.2 SCIF Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|-----------------------------|--------------|---------------------|----------------------|------------|----------------|-------------|
| Serial mode register | SCSMR2 | R/W | H'0000 | H'FFE80000 | H'IFE80000 | 16 |
| Bit rate register | SCBRR2 | R/W | H'FF | H'FFE80004 | H'IFE80004 | 8 |
| Serial control register | SCSCR2 | R/W | H'0000 | H'FFE80008 | H'IFE80008 | 16 |
| Transmit FIFO data register | SCFTDR2 | W | Undefined | H'FFE8000C | H'IFE8000C | 8 |
| Serial status register | SCFSR2 | R/(W)* ¹ | H'0060 | H'FFE80010 | H'IFE80010 | 16 |
| Receive FIFO data register | SCFRDR2 | R | Undefined | H'FFE80014 | H'IFE80014 | 8 |
| FIFO control register | SCFCR2 | R/W | H'0000 | H'FFE80018 | H'IFE80018 | 16 |
| FIFO data count register | SCFDR2 | R | H'0000 | H'FFE8001C | H'IFE8001C | 16 |
| Serial port register | SCSPTR2 | R/W | H'0000* ² | H'FFE80020 | H'IFE80020 | 16 |
| Line status register | SCLSR2 | R/(W)* ³ | H'0000 | H'FFE80024 | H'IFE80024 | 16 |

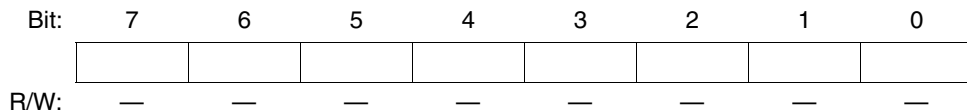
Notes: 1. Only 0 can be written, to clear flags. Bits 15 to 8, 3, and 2 are read-only, and cannot be modified.

2. The value of bits 6, 4, 2, and 0 is undefined.

3. Only 0 can be written, to clear flags. Bits 15 to 1 are read-only, and cannot be modified.

16.2 Register Descriptions

16.2.1 Receive Shift Register (SCRSR2)

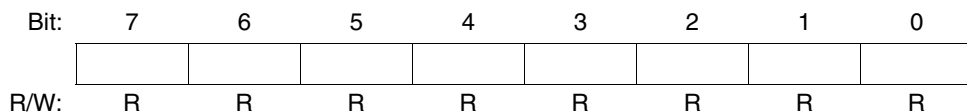


SCRSR2 is the register used to receive serial data.

The SCIF sets serial data input from the RxD2 pin in SCRSR2 in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to the receive FIFO register, SCFRDR2, automatically.

SCRSR2 cannot be directly read or written to by the CPU.

16.2.2 Receive FIFO Data Register (SCFRDR2)



SCFRDR2 is a 16-stage FIFO register that stores received serial data.

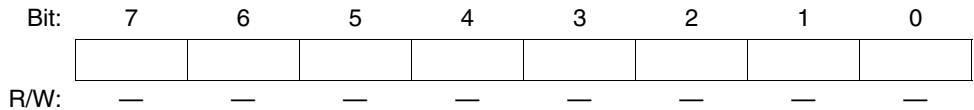
When the SCIF has received one byte of serial data, it transfers the received data from SCRSR2 to SCFRDR2 where it is stored, and completes the receive operation. SCRSR2 is then enabled for reception, and consecutive receive operations can be performed until the receive FIFO register is full (16 data bytes).

SCFRDR2 is a read-only register, and cannot be written to by the CPU.

If a read is performed when there is no receive data in the receive FIFO register, an undefined value will be returned. When the receive FIFO register is full of receive data, subsequent serial data is lost.

The contents of SCFRDR2 are undefined after a power-on reset or manual reset.

16.2.3 Transmit Shift Register (SCTSR2)



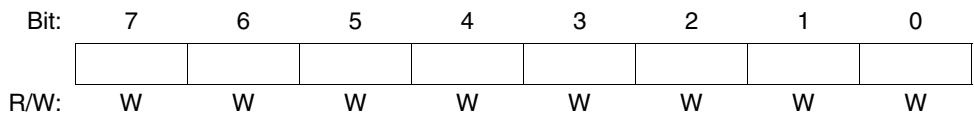
SCTSR2 is the register used to transmit serial data.

To perform serial data transmission, the SCIF first transfers transmit data from SCFTDR2 to SCTSR2, then sends the data to the TxD2 pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCFTDR2 to SCTSR2, and transmission started, automatically.

SCTSR2 cannot be directly read or written to by the CPU.

16.2.4 Transmit FIFO Data Register (SCFTDR2)



SCFTDR2 is a 16-stage FIFO register that stores 8-bit data for serial transmission.

If SCTSR2 is empty when transmit data has been written to SCFTDR2, the SCIF transfers the transmit data written in SCFTDR2 to SCTSR2 and starts serial transmission.

SCFTDR2 is a write-only register, and cannot be read by the CPU.

The next data cannot be written when SCFTDR2 is filled with 16 bytes of transmit data. Data written in this case is ignored.

The contents of SCFTDR2 are undefined after a power-on reset or manual reset.

16.2.5 Serial Mode Register (SCSMR2)

| | | | | | | | | |
|----------------|----|-----|-----|--------------|------|----|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CHR | PE | O/ \bar{E} | STOP | — | CKS1 | CKS0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | R/W | R/W | R/W | R | R/W | R/W |

SCSMR2 is a 16-bit register used to set the SCIF's serial transfer format and select the baud rate generator clock source.

SCSMR2 can be read or written to by the CPU at all times.

SCSMR2 is initialized to H'0000 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 7—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 6—Character Length (CHR): Selects 7 or 8 bits as the asynchronous mode data length.

| Bit 6: CHR | Description |
|------------|----------------------------|
| 0 | 8-bit data (Initial value) |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of SCFTDR2 is not transmitted.

Bit 5—Parity Enable (PE): Selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception.

| Bit 5: PE | Description |
|-----------|---|
| 0 | Parity bit addition and checking disabled (Initial value) |
| 1 | Parity bit addition and checking enabled* |

Note: * When the PE bit is set to 1, the parity (even or odd) specified by the O/ \bar{E} bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/ \bar{E} bit.

Bit 4—Parity Mode (O/\bar{E}): Selects either even or odd parity for use in parity addition and checking. The O/\bar{E} bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking. The O/\bar{E} bit setting is invalid when parity addition and checking is disabled.

| Bit 4: O/\bar{E} | Description |
|--------------------|---|
| 0 | Even parity* ¹ (Initial value) |
| 1 | Odd parity* ² |

- Notes:
1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is even.
 2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is odd.

Bit 3—Stop Bit Length (STOP): Selects 1 or 2 bits as the stop bit length.

| Bit 3: STOP | Description |
|-------------|--|
| 0 | 1 stop bit* ¹ (Initial value) |
| 1 | 2 stop bits* ² |

- Notes:
1. In transmission, a single 1-bit (stop bit) is added to the end of a transmit character before it is sent.
 2. In transmission, two 1-bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, 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 it is 0, it is treated as the start bit of the next transmit character.

Bit 2—Reserved: This bit is always read as 0, and should only be written with 0.

Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock source for the on-chip baud rate generator. The clock source can be selected from Pck, Pck/4, Pck/16, and Pck/64, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 16.2.8, Bit Rate Register (SCBRR2).

| Bit 1: CKS1 | Bit 0: CKS0 | Description |
|-------------|-------------|---------------------------|
| 0 | 0 | Pck clock (Initial value) |
| | 1 | Pck/4 clock |
| 1 | 0 | Pck/16 clock |
| | 1 | Pck/64 clock |

Note: Pck: Peripheral clock

16.2.6 Serial Control Register (SCSCR2)

| | | | | | | | | |
|----------------|-----|-----|-----|-----|------|----|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TIE | RIE | TE | RE | REIE | — | 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 | R/W | R/W |

The SCSCR2 register performs enabling or disabling of SCIF transfer operations, serial clock output, and interrupt requests, and selection of the serial clock source.

SCSCR2 can be read or written to by the CPU at all times.

SCSCR2 is initialized to H'0000 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 8, and 2—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables transmit-FIFO-data-empty interrupt (TXI) request generation when serial transmit data is transferred from SCFTDR2 to SCTSR2, the number of data bytes in the transmit FIFO register falls to or below the transmit trigger set number, and the TDFE flag in the serial status register (SCFSR2) is set to 1.

| Bit 7: TIE | Description |
|------------|--|
| 0 | Transmit-FIFO-data-empty interrupt (TXI) request disabled* (Initial value) |
| 1 | Transmit-FIFO-data-empty interrupt (TXI) request enabled |

Note: * TXI interrupt requests can be cleared by writing transmit data exceeding the transmit trigger set number to SCFTDR2 after reading 1 from the TDFE flag, then clearing it to 0, or by clearing the TIE bit to 0.

Bit 6—Receive Interrupt Enable (RIE): Enables or disables generation of a receive-data-full interrupt (RXI) request when the RDF flag or DR flag in SCFSR2 is set to 1, a receive-error interrupt (ERI) request when the ER flag in SCFSR2 is set to 1, and a break interrupt (BRI) request when the BRK flag in SCFSR2 or the ORER flag in SCLSR2 is set to 1.

| Bit 6: RIE | Description |
|------------|---|
| 0 | Receive-data-full interrupt (RXI) request, receive-error interrupt (ERI) request, and break interrupt (BRI) request disabled* (Initial value) |
| 1 | Receive-data-full interrupt (RXI) request, receive-error interrupt (ERI) request, and break interrupt (BRI) request enabled |

Note: * An RXI interrupt request can be cleared by reading 1 from the RDF or DR flag, then clearing the flag to 0, or by clearing the RIE bit to 0. ERI and BRI interrupt requests can be cleared by reading 1 from the ER, BRK, or ORER flag, then clearing the flag to 0, or by clearing the RIE and REIE bits to 0.

Bit 5—Transmit Enable (TE): Enables or disables the start of serial transmission by the SCIF.

| Bit 5: TE | Description |
|-----------|---------------------------------------|
| 0 | Transmission disabled (Initial value) |
| 1 | Transmission enabled* |

Note: * Serial transmission is started when transmit data is written to SCFTDR2 in this state. Serial mode register (SCSMR2) and FIFO control register (SCFCR2) settings must be made, the transmission format decided, and the transmit FIFO reset, before the TE bit is set to 1.

Bit 4—Receive Enable (RE): Enables or disables the start of serial reception by the SCIF.

| Bit 4: RE | Description |
|-----------|--|
| 0 | Reception disabled* ¹ (Initial value) |
| 1 | Reception enabled* ² |

Notes: 1. Clearing the RE bit to 0 does not affect the DR, ER, BRK, RDF, FER, PER, and ORER flags, which retain their states.
 2. Serial transmission is started when a start bit is detected in this state. Serial mode register (SCSMR2) and FIFO control register (SCFCR2) settings must be made, the reception format decided, and the receive FIFO reset, before the RE bit is set to 1.

Bit 3—Receive Error Interrupt Enable (REIE): Enables or disables generation of receive-error interrupt (ERI) and break interrupt (BRI) requests. The REIE bit setting is valid only when the RIE bit is 0.

| Bit 3: REIE | Description |
|-------------|--|
| 0 | Receive-error interrupt (ERI) and break interrupt (BRI) requests disabled* (Initial value) |
| 1 | Receive-error interrupt (ERI) and break interrupt (BRI) requests enabled |

Note: * Receive-error interrupt (ERI) and break interrupt (BRI) requests can be cleared by reading 1 from the ER, BRK, or ORER flag, then clearing the flag to 0, or by clearing the RIE and REIE bits to 0. When REIE is set to 1, ERI and BRI interrupt requests will be generated even if RIE is cleared to 0. In DMAC transfer, this setting is made if the interrupt controller is to be notified of ERI and BRI interrupt requests.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1 and CKE0): These bits select the SCIF clock source and enable/disable clock output from the SCK2 pin. The combination of CKE1 and CKE0 determine whether the SCK2 pin functions as serial clock output pin or the serial clock input pin.

Note, however, that the setting of the CKE0 bit is valid only when CKE1 = 0 (internal clock operation). When CKE1 = 1 (external clock), CKE0 is ignored. Also, be sure to set CKE1 and CKE0 prior to determining the SCIF operating mode with SCSMR2.

| Bit 1: CKE1 | Bit 0: CKE0 | Description |
|-------------|-------------|---|
| 0 | 0 | Internal clock/SCK pin functions as port (Initial value) |
| | 1 | Internal clock/SCK2 pin functions as clock output* ¹ |
| 1 | 0 | External clock/SCK2 pin functions as clock input* ² |
| | 1 | External clock/SCK2 pin functions as clock input* ² |

Notes: 1. Outputs a clock with a frequency 16 times the bit rate.
 2. Inputs a clock with a frequency 16 times the bit rate.

16.2.7 Serial Status Register (SCFSR2)

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|------|------|------|------|------|------|------|------|
| | PER3 | PER2 | PER1 | PER0 | FER3 | FER2 | FER1 | FER0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|--------|--------|--------|--------|-----|-----|--------|--------|
| | ER | TEND | TDFE | BRK | FER | PER | RDF | DR |
| Initial value: | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/(W)* | R/(W)* |

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

SCFSR2 is a 16-bit register. The lower 8 bits consist of status flags that indicate the operating status of the SCIF, and the upper 8 bits indicate the number of receive errors in the data in the receive FIFO register.

SCFSR2 can be read or written to by the CPU at all times. However, 1 cannot be written to flags ER, TEND, TDFE, BRK, RDF, and DR. Also note that in order to clear these flags they must be read as 1 beforehand. The FER flag and PER flag are read-only flags and cannot be modified.

SCFSR2 is initialized to H'0060 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 12—Number of Parity Errors (PER3–PER0): These bits indicate the number of data bytes in which a parity error occurred in the receive data stored in SCFRDR2.

After the ER bit in SCFSR2 is set, the value indicated by bits 15 to 12 is the number of data bytes in which a parity error occurred.

If all 16 bytes of receive data in SCFRDR2 have parity errors, the value indicated by bits PER3 to PER0 will be 0.

Bits 11 to 8—Number of Framing Errors (FER3–FER0): These bits indicate the number of data bytes in which a framing error occurred in the receive data stored in SCFRDR2.

After the ER bit in SCFSR2 is set, the value indicated by bits 11 to 8 is the number of data bytes in which a framing error occurred.

If all 16 bytes of receive data in SCFRDR2 have framing errors, the value indicated by bits FER3 to FER0 will be 0.

Bit 7—Receive Error (ER): Indicates that a framing error or parity error occurred during reception.*

Note: * The ER flag is not affected and retains its previous state when the RE bit in SCSCR2 is cleared to 0. When a receive error occurs, the receive data is still transferred to SCFRDR2, and reception continues.

The FER and PER bits in SCFSR2 can be used to determine whether there is a receive error in the data read from SCFRDR2.

| Bit 7: ER | Description |
|-----------|---|
| 0 | No framing error or parity error occurred during reception (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset or manual reset • When 0 is written to ER after reading ER = 1 |
| 1 | A framing error or parity error occurred during reception [Setting conditions] <ul style="list-style-type: none"> • When the SCIF checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0* • When, in reception, the number of 1-bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/\bar{E} bit in SCSMR2 |

Note: * In 2-stop-bit mode, only the first stop bit is checked for a value of 1; the second stop bit is not checked.

Bit 6—Transmit End (TEND): Indicates that there is no valid data in SCFTDR2 when the last bit of the transmit character is sent, and transmission has been ended.

| Bit 6: TEND | Description |
|-------------|---|
| 0 | Transmission is in progress [Clearing conditions] <ul style="list-style-type: none">• When transmit data is written to SCFTDR2, and 0 is written to TEND after reading TEND = 1• When data is written to SCFTDR2 by the DMAC |
| 1 | Transmission has been ended (Initial value) [Setting conditions] <ul style="list-style-type: none">• Power-on reset or manual reset• When the TE bit in SCSCR2 is 0• When there is no transmit data in SCFTDR2 on transmission of the last bit of a 1-byte serial transmit character |

Bit 5—Transmit FIFO Data Empty (TDFE): Indicates that data has been transferred from SCFTDR2 to SCTSR2, the number of data bytes in SCFTDR2 has fallen to or below the transmit trigger data number set by bits TTRG1 and TTRG0 in the FIFO control register (SCFCR2), and new transmit data can be written to SCFTDR2.

| Bit 5: TDFE | Description |
|-------------|--|
| 0 | <p>A number of transmit data bytes exceeding the transmit trigger set number have been written to SCFTDR2</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> • When transmit data exceeding the transmit trigger set number is written to SCFTDR2 after reading TDFE = 1, and 0 is written to TDFE • When transmit data exceeding the transmit trigger set number is written to SCFTDR2 by the DMAC |
| 1 | <p>The number of transmit data bytes in SCFTDR2 does not exceed the transmit trigger set number (Initial value)</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> • Power-on reset or manual reset • When the number of SCFTDR2 transmit data bytes falls to or below the transmit trigger set number as the result of a transmit operation* |

Note: * As SCFTDR2 is a 16-byte FIFO register, the maximum number of bytes that can be written when TDFE = 1 is 16 - (transmit trigger set number). Data written in excess of this will be ignored.

The number of data bytes in SCFTDR2 is indicated by the upper bits of SCFDR2.

Bit 4—Break Detect (BRK): Indicates that a receive data break signal has been detected.

| Bit 4: BRK | Description |
|------------|---|
| 0 | <p>A break signal has not been received (Initial value)</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> • Power-on reset or manual reset • When 0 is written to BRK after reading BRK = 1 |
| 1 | <p>A break signal has been received*</p> <p>[Setting condition]</p> <p>When data with a framing error is received, followed by the space "0" level (low level) for at least one frame length</p> |

Note: * When a break is detected, the receive data (H'00) following detection is not transferred to SCFRDR2. When the break ends and the receive signal returns to mark "1", receive data transfer is resumed.

Bit 3—Framing Error (FER): Indicates whether or not a framing error has been found in the data that is to be read from the receive FIFO data register (SCFRDR2).

| Bit 3: FER | Description |
|------------|---|
| 0 | There is no framing error in the receive data that is to be read from SCFRDR2 (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset or manual reset • When there is no framing error in the data that is to be read next from SCFRDR2 |
| 1 | There is a framing error in the receive data that is to be read from SCFRDR2 [Setting condition] When there is a framing error in the data that is to be read next from SCFRDR2 |

Bit 2—Parity Error (PER): Indicates whether or not a parity error has been found in the data that is to be read from the receive FIFO data register (SCFRDR2).

| Bit 2: PER | Description |
|------------|---|
| 0 | There is no parity error in the receive data that is to be read from SCFRDR2 (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset or manual reset • When there is no parity error in the data that is to be read next from SCFRDR2 |
| 1 | There is a parity error in the receive data that is to be read from SCFRDR2 [Setting condition] When there is a parity error in the data that is to be read next from SCFRDR2 |

Bit 1—Receive FIFO Data Full (RDF): Indicates that the received data has been transferred from SCRSR2 to SCFRDR2, and the number of receive data bytes in SCFRDR2 is equal to or greater than the receive trigger number set by bits RTRG1 and RTRG0 in the FIFO control register (SCFCR2).

| Bit 1: RDF | Description |
|------------|---|
| 0 | <p>The number of receive data bytes in SCFRDR2 is less than the receive trigger set number (Initial value)</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> • Power-on reset or manual reset • When SCFRDR2 is read until the number of receive data bytes in SCFRDR2 falls below the receive trigger set number after reading RDF = 1, and 0 is written to RDF • When SCFRDR2 is read by the DMAC until the number of receive data bytes in SCFRDR2 falls below the receive trigger set number |
| 1 | <p>The number of receive data bytes in SCFRDR2 is equal to or greater than the receive trigger set number</p> <p>[Setting condition]</p> <p>When SCFRDR2 contains at least the receive trigger set number of receive data bytes*</p> |

Note: * SCFRDR2 is a 16-byte FIFO register. When RDF = 1, at least the receive trigger set number of data bytes can be read. If all the data in SCFRDR2 is read and another read is performed, the data value will be undefined. The number of receive data bytes in SCFRDR2 is indicated by the lower bits of SCFDR2.

Bit 0—Receive Data Ready (DR): Indicates that there are fewer than the receive trigger set number of data bytes in SCFRDR2, and no further data has arrived for at least 15 etu after the stop bit of the last data received.

| Bit 0: DR | Description |
|-----------|---|
| 0 | Reception is in progress or has ended normally and there is no receive data left in SCFRDR2 (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset or manual reset • When all the receive data in SCFRDR2 has been read after reading DR = 1, and 0 is written to DR • When all the receive data in SCFRDR2 has been read by the DMAC |
| 1 | No further receive data has arrived [Setting condition] When SCFRDR2 contains fewer than the receive trigger set number of receive data bytes, and no further data has arrived for at least 15 etu after the stop bit of the last data received* |

Note: * Equivalent to 1.5 frames with an 8-bit, 1-stop-bit format.
etu: Elementary time unit (time for transfer of 1 bit)

16.2.8 Bit Rate Register (SCBRR2)

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

SCBRR2 is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SCSMR2.

SCBRR2 can be read or written to by the CPU at all times.

SCBRR2 is initialized to H'FF by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

The SCBRR2 setting is found from the following equation.

Asynchronous mode:

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

Where B: Bit rate (bits/s)

N: SCBRR2 setting for baud rate generator ($0 \leq N \leq 255$)

Pck: Peripheral module operating frequency (MHz)

n: Baud rate generator input clock ($n = 0$ to 3)

(See the table below for the relation between n and the clock.)

| n | Clock | SCSMR2 Setting | |
|---|--------|----------------|------|
| | | CKS1 | CKS0 |
| 0 | Pck | 0 | 0 |
| 1 | Pck/4 | 0 | 1 |
| 2 | Pck/16 | 1 | 0 |
| 3 | Pck/64 | 1 | 1 |

The bit rate error in asynchronous mode is found from the following equation:

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

16.2.9 FIFO Control Register (SCFCR2)

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-----|--------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | RSTRG2 | RSTRG1 | RSTRG0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RTRG1 | RTRG0 | TTRG1 | TTRG0 | MCE | TFRST | RFRST | LOOP |
| 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 |

SCFCR2 performs data count resetting and trigger data number setting for the transmit and receive FIFO registers, and also contains a loopback test enable bit.

SCFCR2 can be read or written to by the CPU at all times.

SCFCR2 is initialized to H'0000 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 11—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 10, 9 and 8— $\overline{\text{RTS2}}$ Output Active Trigger (RSTRG2, RSTG1, and RSTG0): These bits output the high level to the $\overline{\text{RTS2}}$ signal when the number of received data stored in the receive FIFO data register (SCFRDR2) exceeds the trigger number, as shown in the table below.

| Bit 10: RSTRG2 | Bit 9: RSTRG1 | Bit 8: RSTRG0 | $\overline{\text{RTS2}}$ Output Active Trigger |
|----------------|---------------|---------------|--|
| 0 | 0 | 0 | 15 (Initial value) |
| | | 1 | 1 |
| | 1 | 0 | 4 |
| | | 1 | 6 |
| 1 | 0 | 0 | 8 |
| | | 1 | 10 |
| | 1 | 0 | 12 |
| | | 1 | 14 |

Bits 7 and 6—Receive FIFO Data Number Trigger (RTRG1, RTRG0): These bits are used to set the number of receive data bytes that sets the receive data full (RDF) flag in the serial status register (SCFSR2).

The RDF flag is set when the number of receive data bytes in SCFRDR2 is equal to or greater than the trigger set number shown in the following table.

| Bit 7: RTRG1 | Bit 6: RTRG0 | Receive Trigger Number |
|--------------|--------------|------------------------|
| 0 | 0 | 1 (Initial value) |
| | 1 | 4 |
| 1 | 0 | 8 |
| | 1 | 14 |

Bits 5 and 4—Transmit FIFO Data Number Trigger (TTRG1, TTRG0): These bits are used to set the number of remaining transmit data bytes that sets the transmit FIFO data register empty (TDFE) flag in the serial status register (SCFSR2). The TDFE flag is set when the number of transmit data bytes in SCFTDR2 is equal to or less than the trigger set number shown in the following table.

| Bit 5: TTRG1 | Bit 4: TTRG0 | Transmit Trigger Number | |
|--------------|--------------|-------------------------|-----------------|
| 0 | 0 | 8 (8) | (Initial value) |
| | 1 | 4 (12) | |
| 1 | 0 | 2 (14) | |
| | 1 | 1 (15) | |

Note: Figures in parentheses are the number of empty bytes in SCFTDR2 when the flag is set.

Bit 3—Modem Control Enable (MCE): Enables the $\overline{\text{CTS2}}$ and $\overline{\text{RTS2}}$ modem control signals.

| Bit 3: MCE | Description | |
|------------|-------------------------|-----------------|
| 0 | Modem signals disabled* | (Initial value) |
| 1 | Modem signals enabled | |

Note: * $\overline{\text{CTS2}}$ is fixed at active-0 regardless of the input value, and $\overline{\text{RTS2}}$ output is also fixed at 0.

Bit 2—Transmit FIFO Data Register Reset (TFRST): Invalidates the transmit data in the transmit FIFO data register and resets it to the empty state.

| Bit 2: TFRST | Description | |
|--------------|---------------------------|-----------------|
| 0 | Reset operation disabled* | (Initial value) |
| 1 | Reset operation enabled | |

Note: * A reset operation is performed in the event of a power-on reset or manual reset.

Bit 1—Receive FIFO Data Register Reset (RFRST): Invalidates the receive data in the receive FIFO data register and resets it to the empty state.

| Bit 1: RFRST | Description | |
|--------------|---------------------------|-----------------|
| 0 | Reset operation disabled* | (Initial value) |
| 1 | Reset operation enabled | |

Note: * A reset operation is performed in the event of a power-on reset or manual reset.

Bit 0—Loopback Test (LOOP): Internally connects the transmit output pin (TxD2) and receive input pin (RxD2), and the $\overline{\text{RTS2}}$ pin and $\overline{\text{CTS2}}$ pin, enabling loopback testing.

| Bit 0: LOOP | Description |
|-------------|--|
| 0 | Loopback test disabled (Initial value) |
| 1 | Loopback test enabled |

16.2.10 FIFO Data Count Register (SCFDR2)

SCFDR2 is a 16-bit register that indicates the number of data bytes stored in SCFTDR2 and SCFRDR2.

The upper 8 bits show the number of transmit data bytes in SCFTDR2, and the lower 8 bits show the number of receive data bytes in SCFRDR2.

SCFDR2 can be read by the CPU at all times.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|----|----|----|----|----|----|----|----|
| | — | — | — | T4 | T3 | T2 | T1 | T0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

These bits show the number of untransmitted data bytes in SCFTDR2. A value of H'00 indicates that there is no transmit data, and a value of H'10 indicates that SCFTDR2 is full of transmit data.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|----|----|----|----|----|
| | — | — | — | R4 | R3 | R2 | R1 | R0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

These bits show the number of receive data bytes in SCFRDR2. A value of H'00 indicates that there is no receive data, and a value of H'10 indicates that SCFRDR2 is full of receive data.

16.2.11 Serial Port Register (SCSPTR2)

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RTSIO | RTSDT | CTSIO | CTSDT | SCKIO | SCKDT | SPB2IO | SPB2DT |
| Initial value: | 0 | — | 0 | — | 0 | — | 0 | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SCSPTR2 is a 16-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface with FIFO (SCIF) pins. Input data can be read from the RxD2 pin, output data written to the TxD2 pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. Data can be read from, and output data written to, the SCK2 pin by means of bits 3 and 2. Data can be read from, and output data written to, the $\overline{\text{CTS2}}$ pin by means of bits 5 and 4. Data can be read from, and output data written to, the $\overline{\text{RTS2}}$ pin by means of bits 6 and 7.

SCSPTR2 can be read or written to by the CPU at all times. All SCSPTR2 bits except bits 6, 4, 2, and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 6, 4, 2, and 0 is undefined. SCSPTR2 is not initialized in standby mode or in the module standby state.

Bits 15 to 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Serial Port RTS Port I/O (RTSIO): Specifies the serial port $\overline{\text{RTS2}}$ pin input/output condition. When the $\overline{\text{RTS2}}$ pin is actually set as a port output pin and outputs the value set by the RTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

| Bit 7: RTSIO | Description |
|--------------|---|
| 0 | RTSDT bit value is not output to $\overline{\text{RTS2}}$ pin (Initial value) |
| 1 | RTSDT bit value is output to $\overline{\text{RTS2}}$ pin |

Bit 6—Serial Port RTS Port Data (RTSDT): Specifies the serial port $\overline{\text{RTS2}}$ pin input/output data. Input or output is specified by the RTSIO bit (see the description of bit 7, RTSIO, for details). In output mode, the RTSDT bit value is output to the $\overline{\text{RTS2}}$ pin. The $\overline{\text{RTS2}}$ pin value is read from the RTSDT bit regardless of the value of the RTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 6: RTSDT | Description |
|--------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Bit 5—Serial Port CTS Port I/O (CTSIO): Specifies the serial port $\overline{\text{CTS2}}$ pin input/output condition. When the $\overline{\text{CTS2}}$ pin is actually set as a port output pin and outputs the value set by the CTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

| Bit 5: CTSIO | Description |
|--------------|---|
| 0 | CTSDT bit value is not output to $\overline{\text{CTS2}}$ pin (Initial value) |
| 1 | CTSDT bit value is output to $\overline{\text{CTS2}}$ pin |

Bit 4—Serial Port CTS Port Data (CTSDT): Specifies the serial port $\overline{\text{CTS2}}$ pin input/output data. Input or output is specified by the CTSIO bit (see the description of bit 5, CTSIO, for details). In output mode, the CTSDT bit value is output to the $\overline{\text{CTS2}}$ pin. The $\overline{\text{CTS2}}$ pin value is read from the CTSDT bit regardless of the value of the CTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 4: CTSDT | Description |
|--------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Bit 3—Serial Port Clock Port I/O (SCKIO): Sets the I/O for the SCK2 pin serial port. To actually set the SCK2 pin as the port output pin and output the value set in the SCKDT bit, set the CKE1 and CKE0 bits of the SCSCR2 register to 0.

| Bit 3: SCKIO | Description |
|--------------|---|
| 0 | Shows that the value of the SCKDT bit is not output to the SCK2 pin (Initial value) |
| 1 | Shows that the value of the SCKDT bit is output to the SCK2 pin. |

Bit 2—Serial Port Clock Port Data (SCKDT): Specifies the I/O data for the SCK2 pin serial port. The SCKIO bit specified input or output. (See bit 3: SCKIO, for details.) When set for output, the value of the SCKDT bit is output to the SCK2 pin. Regardless of the value of the SCKIO bit, the value of the SCK2 pin is fetched from the SCKDT bit. The initial value after a power-on reset or manual reset is undefined.

| Bit 2: SCKDT | Description |
|--------------|------------------------------|
| 0 | Shows I/O data level is LOW |
| 1 | Shows I/O data level is HIGH |

Bit 1—Serial Port Break I/O (SPB2IO): Specifies the serial port TxD2 pin output condition. When the TxD2 pin is actually set as a port output pin and outputs the value set by the SPB2DT bit, the TE bit in SCSCR2 should be cleared to 0.

| Bit 1: SPB2IO | Description |
|---------------|--|
| 0 | SPB2DT bit value is not output to the TxD2 pin (Initial value) |
| 1 | SPB2DT bit value is output to the TxD2 pin |

Bit 0—Serial Port Break Data (SPB2DT): Specifies the serial port Rx/D2 pin input data and TxD2 pin output data. The TxD2 pin output condition is specified by the SPB2IO bit (see the description of bit 1, SPB2IO, for details). When the TxD2 pin is designated as an output, the value of the SPB2DT bit is output to the TxD2 pin. The Rx/D2 pin value is read from the SPB2DT bit regardless of the value of the SPB2IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 0: SPB2DT | Description |
|---------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

SCIF I/O port block diagrams are shown in figures 16.2 to 16.6.

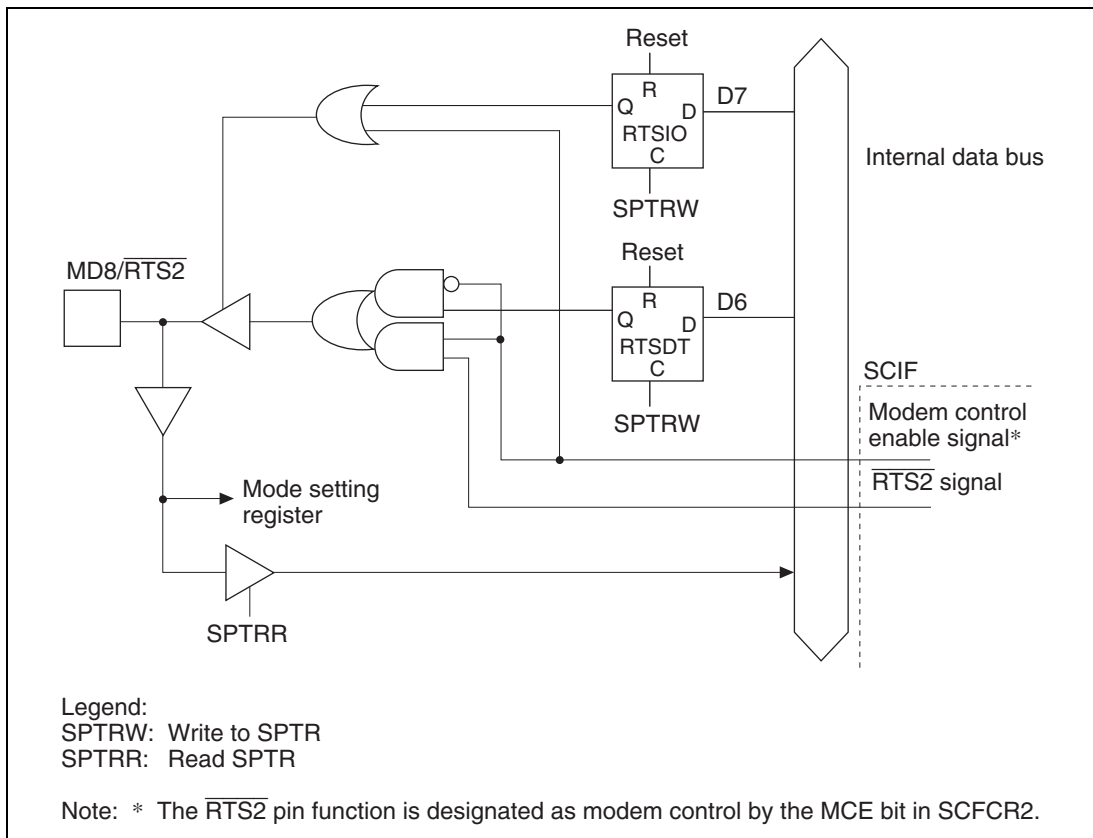


Figure 16.2 MD8/ $\overline{\text{RTS2}}$ Pin

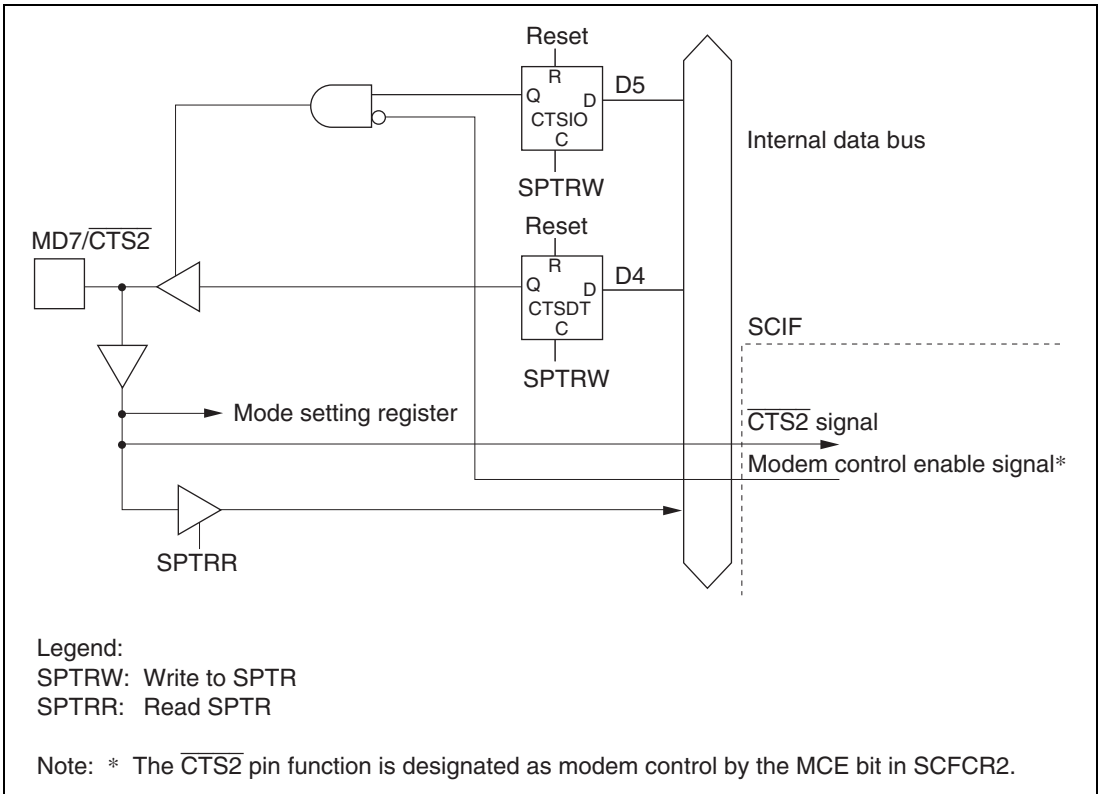


Figure 16.3 MD7/ $\overline{\text{CTS2}}$ Pin

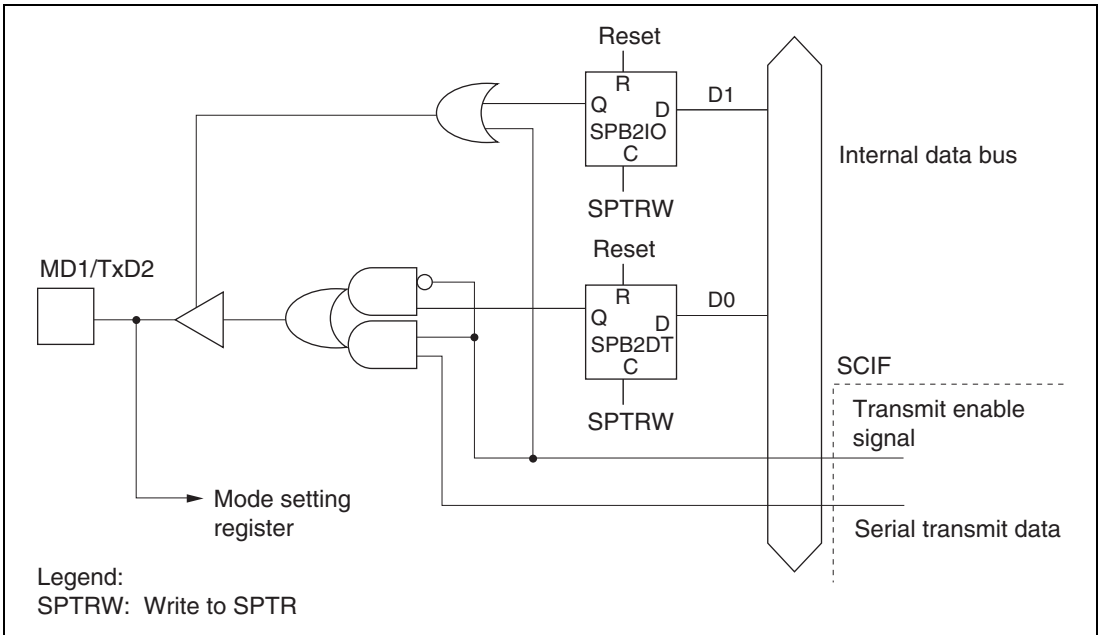


Figure 16.4 MD1/TxD2 Pin

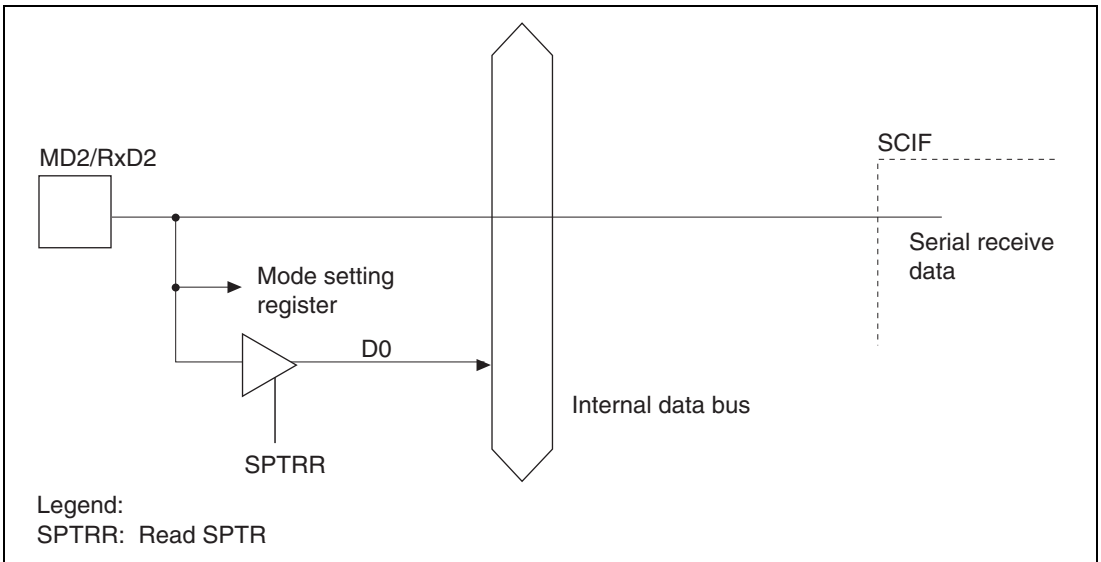


Figure 16.5 MD2/RxD2 Pin

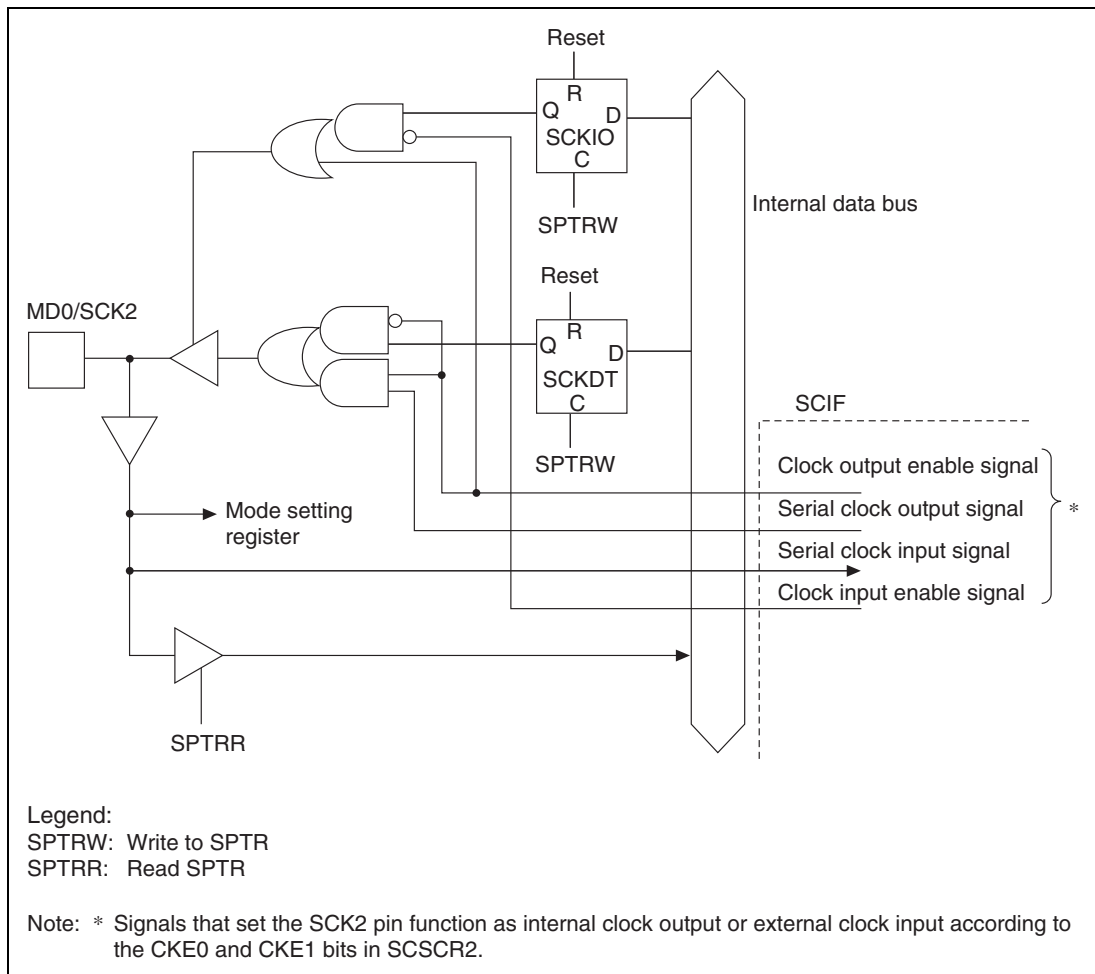


Figure 16.6 MD0/SCK2 Pin

16.2.12 Line Status Register (SCLSR2)

| | | | | | | | | |
|----------------|----|----|----|----|----|----|---|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | ORER |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | (R/W)* |

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

Bits 15 to 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 0—Overrun Error (ORER): Indicates that an overrun error occurred during reception, causing abnormal termination.

| Bit 0: ORER | Description |
|-------------|--|
| 0 | Reception in progress, or reception has ended normally* ¹ (Initial value) [Clearing conditions] <ul style="list-style-type: none"> • Power-on reset or manual reset • When 0 is written to ORER after reading ORER = 1 |
| 1 | An overrun error occurred during reception* ² [Setting condition] <p>When the next serial reception is completed while the receive FIFO is full</p> |

- Notes:
1. The ORER flag is not affected and retains its previous state when the RE bit in SCSCR2 is cleared to 0.
 2. The receive data prior to the overrun error is retained in SCFRDR2, and the data received subsequently is lost. Serial reception cannot be continued while the ORER flag is set to 1.

16.3 Operation

16.3.1 Overview

The SCIF can carry out serial communication in asynchronous mode, in which synchronization is achieved character by character. See section 15.3.2, Operation in Asynchronous Mode, for details.

Sixteen-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed. $\overline{\text{RTS2}}$ and $\overline{\text{CTS2}}$ signals are also provided as modem control signals.

The transmission format is selected using the serial mode register (SCSMR2), as shown in table 16.3. The SCIF clock source is determined by the CKE1 bit in the serial control register (SCSCR2), as shown in table 16.4.

- Data length: Choice of 7 or 8 bits
- Choice of parity addition and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing errors, parity errors, receive-FIFO-data-full state, overrun errors, receive-data-ready state, and breaks, during reception
- Indication of the number of data bytes stored in the transmit and receive FIFO registers
- Choice of internal or external clock as SCIF clock source

When internal clock is selected: The SCIF operates on the baud rate generator clock, and a clock with a frequency of 16 times the bit rate must be output

When external clock is selected: A clock with a frequency of 16 times the bit rate must be input (the on-chip baud rate generator is not used).

Table 16.3 SCSMR2 Settings for Serial Transfer Format Selection

| SCSMR2 Settings | | | | SCIF Transfer Format | | | |
|-----------------|--------------|----------------|-------------------|----------------------|-----------------------|---------------|--------------------|
| Bit 6: CHR | Bit 5: PE | Bit 3: STOP | Mode | Data Length | Multiprocessor Bit | Parity Bit | Stop Bit Length |
| 0 | 0 | 0 | Asynchronous mode | 8-bit data | No | No | 1 bit |
| | | 1 | | | | | 2 bits |
| | 1 | 0 | | | | Yes | 1 bit |
| | | 1 | | | | 2 bits | |
| 1 | 0 | 0 | Asynchronous mode | 7-bit data | No | No | 1 bit |
| | | 1 | | | | | 2 bits |
| | 1 | 0 | | | | Yes | 1 bit |
| | | 1 | | | | 2 bits | |

Table 16.4 SCSCR2 Settings for SCIF Clock Source Selection

| SCSCR2 Setting | | | SCIF Transmit/Receive Clock | |
|----------------|-------------|-------------------|-----------------------------|--|
| Bit 1: CKE1 | Bit 0: CKE0 | Mode | Clock Source | SCK2 Pin Function |
| 0 | 0 | Asynchronous mode | Internal | SCIF does not use SCK2 pin |
| | 1 | | | Output clock with frequency of 16 times the bit rate |
| 1 | 0 | Asynchronous mode | External | Inputs clock with frequency of 16 times the bit rate |
| | 1 | | | |

16.3.2 Serial Operation

Data Transfer Format

Table 16.5 shows the data transfer formats that can be used. Any of 8 transfer formats can be selected according to the SCSMR2 settings.

Table 16.5 Serial Transfer Formats

| SCSMR2 Settings | | | Serial Transfer Format and Frame Length | | | | | | | | | | | | | |
|-----------------|----|------|---|------------|---|---|---|---|---|---|------|------|------|------|--|--|
| CHR | PE | STOP | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | | |
| 0 | 0 | 0 | S | 8-bit data | | | | | | | | STOP | | | | |
| 0 | 0 | 1 | S | 8-bit data | | | | | | | | STOP | STOP | | | |
| 0 | 1 | 0 | S | 8-bit data | | | | | | | | P | STOP | | | |
| 0 | 1 | 1 | S | 8-bit data | | | | | | | | P | STOP | STOP | | |
| 1 | 0 | 0 | S | 7-bit data | | | | | | | STOP | | | | | |
| 1 | 0 | 1 | S | 7-bit data | | | | | | | STOP | STOP | | | | |
| 1 | 1 | 0 | S | 7-bit data | | | | | | | P | STOP | | | | |
| 1 | 1 | 1 | S | 7-bit data | | | | | | | P | STOP | STOP | | | |

Legend:

S: Start bit

STOP: Stop bit

P: Parity bit

Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK2 pin can be selected as the SCIF's serial clock, according to the setting of the CKE1 bit in SCSCR2. For details of SCIF clock source selection, see table 16.4.

When an external clock is input at the SCK2 pin, the clock frequency should be 16 times the bit rate used.

When operating using the internal clock, the clock can be output via the SCK2 pin. The frequency of this clock is 16 times the bit rate.

Data Transfer Operations

SCIF Initialization: Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR2 to 0, then initialize the SCIF as described below.

When the transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, SCTSR2 is initialized. Note that clearing the TE and RE bits to 0 does not change the contents of SCFSR2, SCFTDR2, or SCFRDR2. The TE bit should be cleared to 0 after all transmit data has been sent and the TEND flag in SCFSR2 has been set. TEND can also be cleared to 0 during transmission, but the data being transmitted will go to the mark state after the clearance. Before setting TE again to start transmission, the TFRST bit in SCFCR2 should first be set to 1 to reset SCFTDR2.

When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 16.7 shows a sample SCIF initialization flowchart.

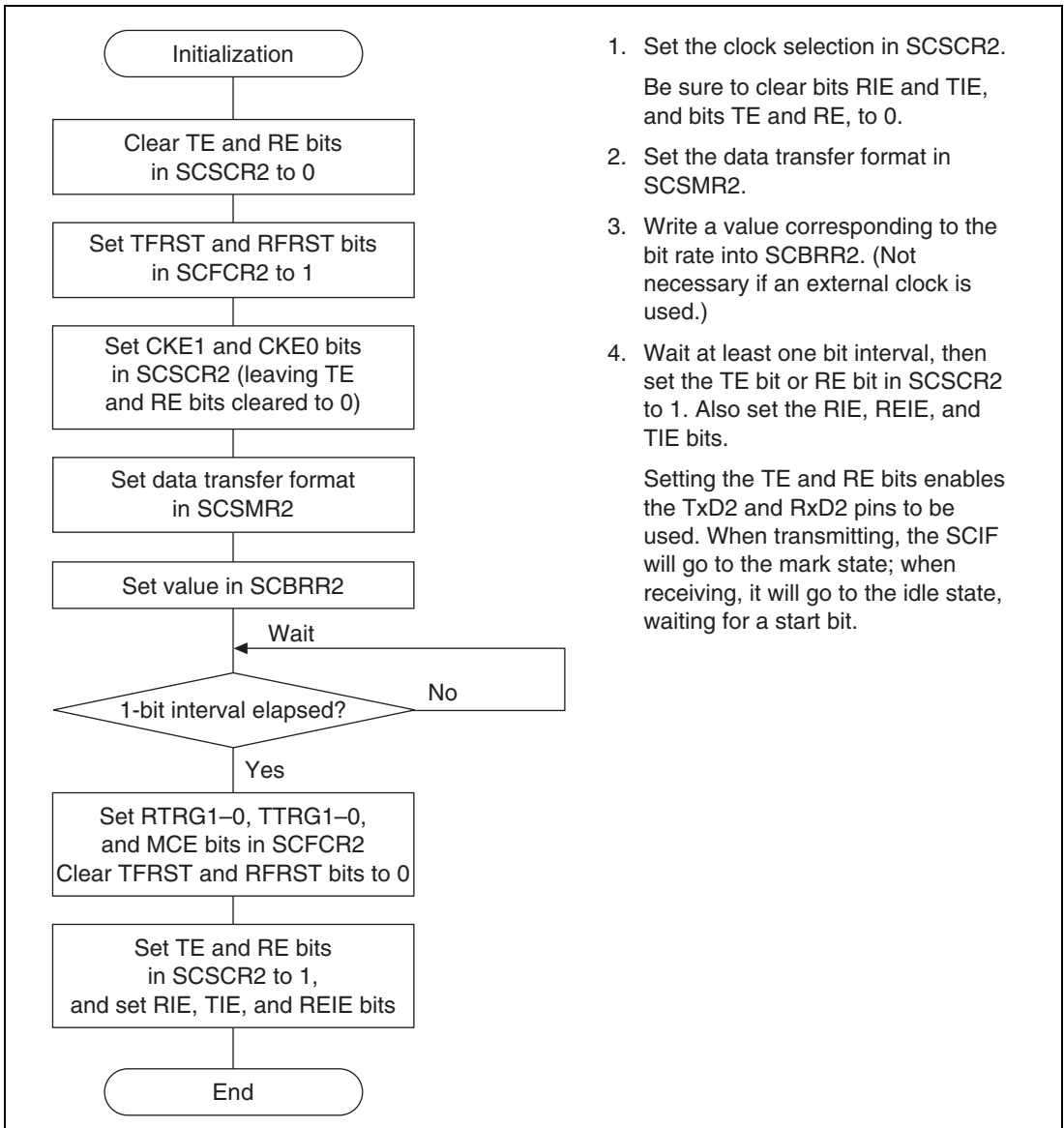
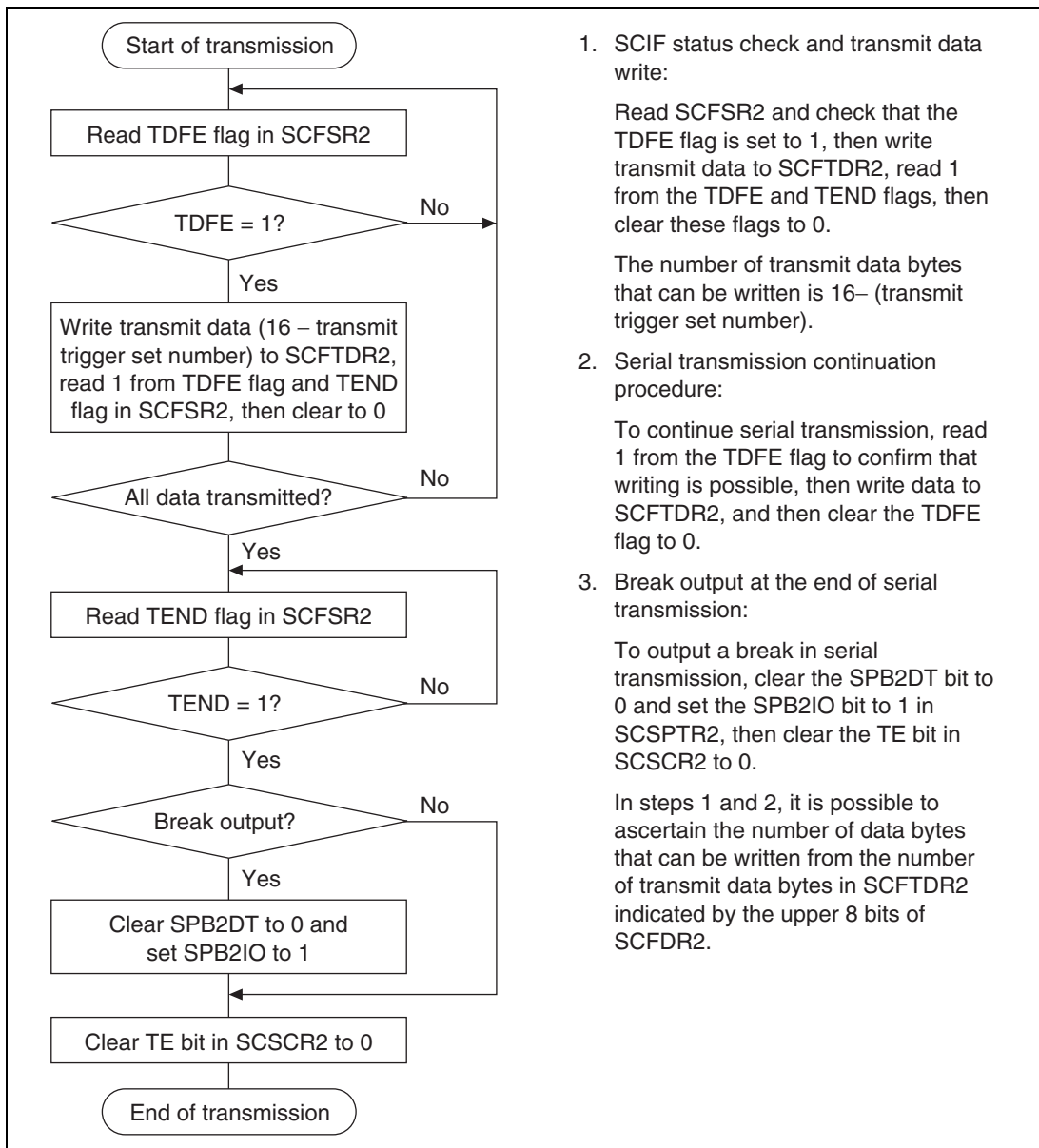


Figure 16.7 Sample SCIF Initialization Flowchart

Serial Data Transmission: Figure 16.8 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCIF for transmission.



1. SCIF status check and transmit data write:

Read SCFSR2 and check that the TDFE flag is set to 1, then write transmit data to SCFTDR2, read 1 from the TDFE and TEND flags, then clear these flags to 0.

The number of transmit data bytes that can be written is 16– (transmit trigger set number).

2. Serial transmission continuation procedure:

To continue serial transmission, read 1 from the TDFE flag to confirm that writing is possible, then write data to SCFTDR2, and then clear the TDFE flag to 0.

3. Break output at the end of serial transmission:

To output a break in serial transmission, clear the SPB2DT bit to 0 and set the SPB2IO bit to 1 in SCSPTR2, then clear the TE bit in SCSCR2 to 0.

In steps 1 and 2, it is possible to ascertain the number of data bytes that can be written from the number of transmit data bytes in SCFTDR2 indicated by the upper 8 bits of SCFDR2.

Figure 16.8 Sample Serial Transmission Flowchart

In serial transmission, the SCIF operates as described below.

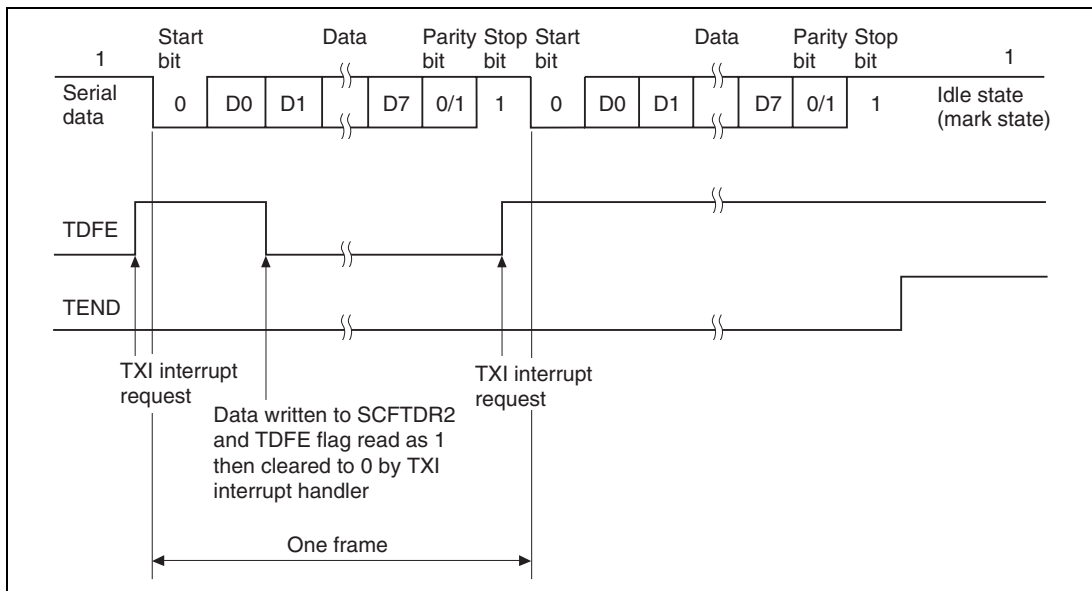
1. When data is written into SCFTDR2, the SCIF transfers the data from SCFTDR2 to SCTSR2 and starts transmitting. Confirm that the TDFE flag in the serial status register (SCFSR2) is set to 1 before writing transmit data to SCFTDR2. The number of data bytes that can be written is at least 16– (transmit trigger set number).
2. When data is transferred from SCFTDR2 to SCTSR2 and transmission is started, consecutive transmit operations are performed until there is no transmit data left in SCFTDR2. When the number of transmit data bytes in SCFTDR2 falls to or below the transmit trigger number set in the FIFO control register (SCFCR2), the TDFE flag is set. If the TIE bit in SCSCR2 is set to 1 at this time, a transmit-FIFO-data-empty interrupt (TXI) request is generated.

The serial transmit data is sent from the TxD2 pin in the following order.

- a. Start bit: One 0-bit is output.
 - b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
 - c. Parity bit: One parity bit (even or odd parity) is output. (A format in which a parity bit is not output can also be selected.)
 - d. Stop bit(s): One or two 1-bits (stop bits) are output.
 - e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCIF checks the SCFTDR2 transmit data at the timing for sending the stop bit. If data is present, the data is transferred from SCFTDR2 to SCTSR2, the stop bit is sent, and then serial transmission of the next frame is started.

If there is no transmit data, the TEND flag in SCFSR2 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output.

Figure 16.9 shows an example of the operation for transmission in asynchronous mode.



**Figure 16.9 Example of Transmit Operation
(Example with 8-Bit Data, Parity, One Stop Bit)**

4. When modem control is enabled, transmission can be stopped and restarted in accordance with the $\overline{\text{CTS2}}$ input value. When $\overline{\text{CTS2}}$ is set to 1, if transmission is in progress, the line goes to the mark state after transmission of one frame. When $\overline{\text{CTS2}}$ is set to 0, the next transmit data is output starting from the start bit.

Figure 16.10 shows an example of the operation when modem control is used.

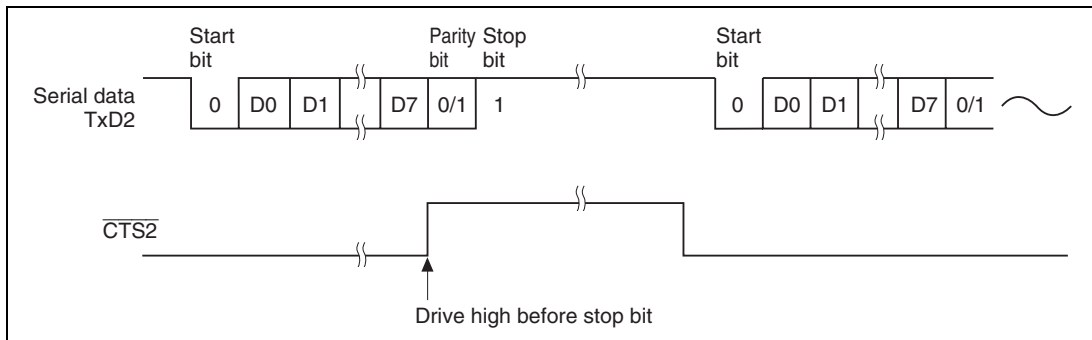
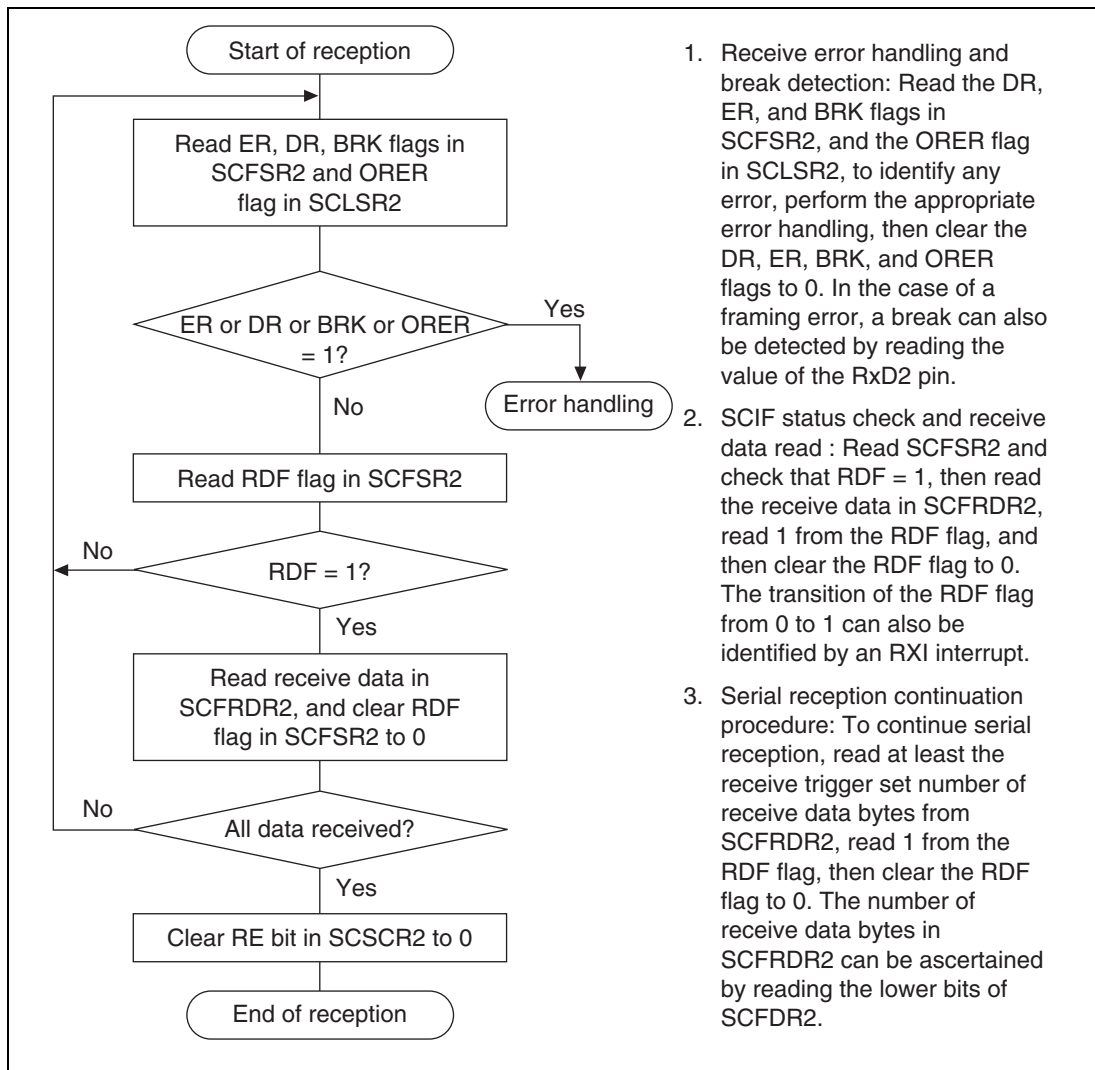


Figure 16.10 Example of Operation Using Modem Control ($\overline{\text{CTS2}}$)

Serial Data Reception: Figure 16.11 shows a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling the SCIF for reception.



1. Receive error handling and break detection: Read the DR, ER, and BRK flags in SCFSR2, and the ORER flag in SCLSR2, to identify any error, perform the appropriate error handling, then clear the DR, ER, BRK, and ORER flags to 0. In the case of a framing error, a break can also be detected by reading the value of the RxD2 pin.
2. SCIF status check and receive data read : Read SCFSR2 and check that RDF = 1, then read the receive data in SCFRDR2, read 1 from the RDF flag, and then clear the RDF flag to 0. The transition of the RDF flag from 0 to 1 can also be identified by an RXI interrupt.
3. Serial reception continuation procedure: To continue serial reception, read at least the receive trigger set number of receive data bytes from SCFRDR2, read 1 from the RDF flag, then clear the RDF flag to 0. The number of receive data bytes in SCFRDR2 can be ascertained by reading the lower bits of SCFDR2.

Figure 16.11 Sample Serial Reception Flowchart (1)

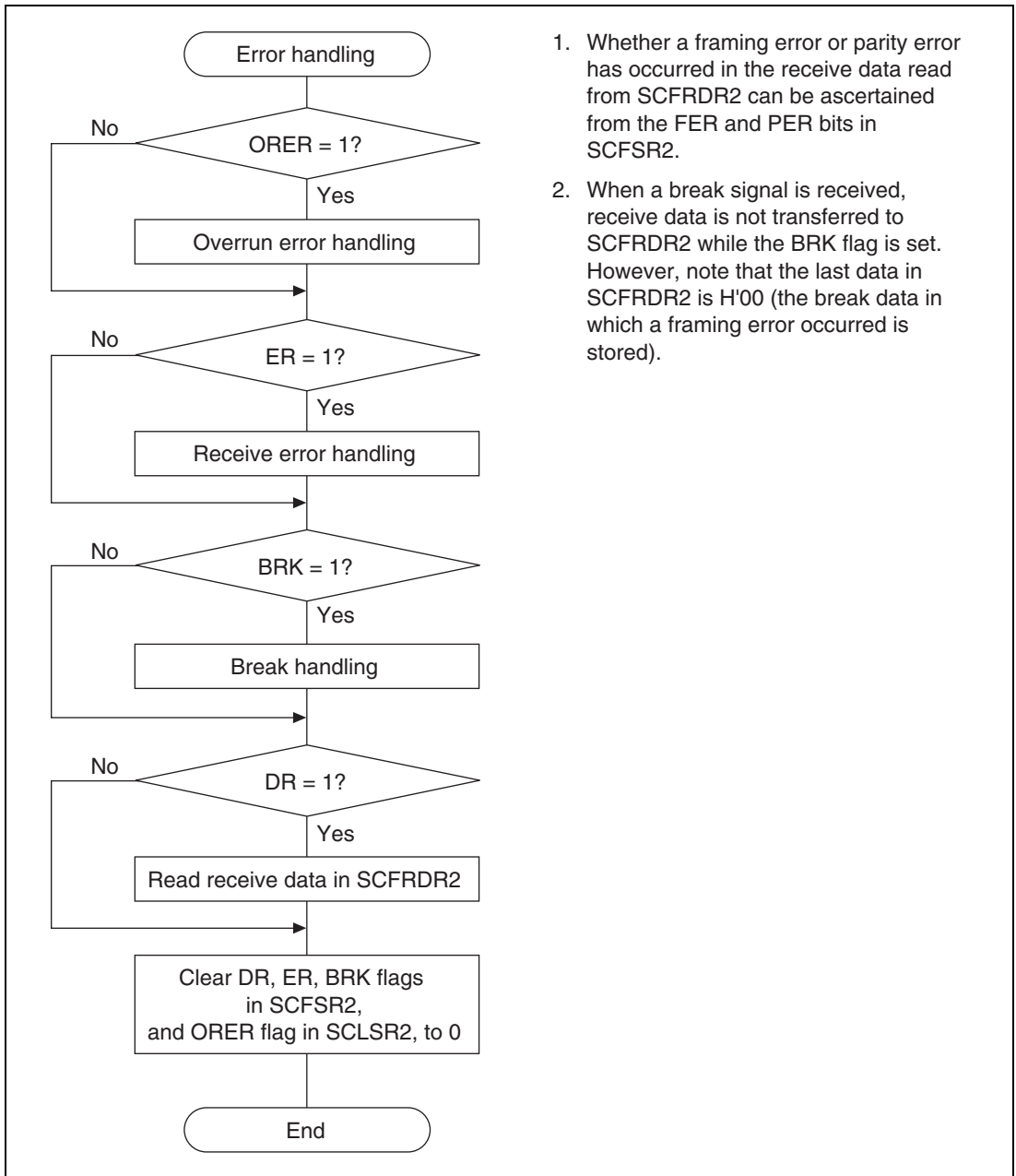


Figure 16.11 Sample Serial Reception Flowchart (2)

In serial reception, the SCIF operates as described below.

1. The SCIF monitors the transmission line, and if a 0 start bit is detected, performs internal synchronization and starts reception.
2. The received data is stored in SCRSR2 in LSB-to-MSB order.
3. The parity bit and stop bit are received.

After receiving these bits, the SCIF carries out the following checks.

- a. Stop bit check: The SCIF checks whether the stop bit is 1. If there are two stop bits, only the first is checked.
- b. The SCIF checks whether receive data can be transferred from the receive shift register (SCRSR2) to SCFRDR2.
- c. Overrun error check: The SCIF checks that the ORER flag is 0, indicating that no overrun error has occurred.
- d. Break check: The SCIF checks that the BRK flag is 0, indicating that the break state is not set.

If all the b, c, and d checks are passed, the receive data is stored in SCFRDR2.

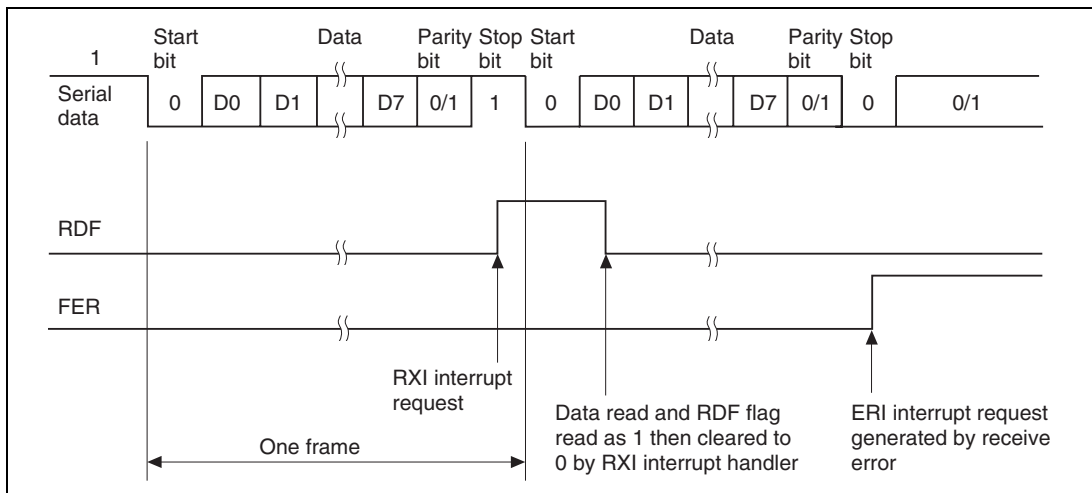
Note: Reception continues when parity error, framing error occurs.

4. If the RIE bit in SCSCR2 is set to 1 when the RDF or DR flag changes to 1, a receive-FIFO-data-full interrupt (RXI) request is generated.

If the RIE bit or REIE bit in SCSCR2 is set to 1 when the ER flag changes to 1, a receive-error interrupt (ERI) request is generated.

If the RIE bit or REIE bit in SCSCR2 is set to 1 when the BRK or ORER flag changes to 1, a break reception interrupt (BRI) request is generated.

Figure 16.12 shows an example of the operation for reception.



**Figure 16.12 Example of SCIF Receive Operation
(Example with 8-Bit Data, Parity, One Stop Bit)**

- When modem control is enabled, the $\overline{\text{RTS2}}$ signal is output when SCFRDR2 is empty. When $\overline{\text{RTS2}}$ is 0, reception is possible. When $\overline{\text{RTS2}}$ is 1, this indicates that SCFRDR2 contains a number of data bytes equal to or greater than the $\overline{\text{RTS2}}$ output active trigger set number. The $\overline{\text{RTS2}}$ output active trigger value is specified by bits 10 to 8 in the FIFO control register (SCFCR2), described in section 16.2.9, FIFO Control Register (SCFCR2). $\overline{\text{RTS2}}$ also goes to 1 when bit 4 (RE) in SCSCR2 is 0.

Figure 16.13 shows an example of the operation when modem control is used.

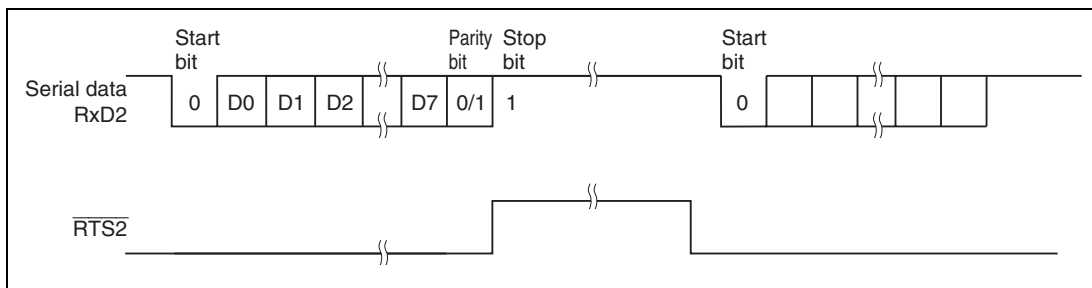


Figure 16.13 Example of Operation Using Modem Control ($\overline{\text{RTS2}}$)

16.4 SCIF Interrupt Sources and the DMAC

The SCIF has four interrupt sources: transmit-FIFO-data-empty interrupt (TXI) request, receive-error interrupt (ERI) request, receive-FIFO-data-full interrupt (RXI) request, and break interrupt (BRI) request.

Table 16.6 shows the interrupt sources and their order of priority. The interrupt sources are enabled or disabled by means of the TIE, RIE, and REIE bits in SCSCR2. A separate interrupt request is sent to the interrupt controller for each of these interrupt sources.

When transmission/reception is carried out using the DMAC, output of interrupt requests to the interrupt controller can be inhibited by clearing the RIE bit in SCSCR2 to 0. By setting the REIE bit to 1 while the RIE bit is cleared to 0, it is possible to output ERI and BRI interrupt requests, but not RXI interrupt requests.

When the TDFE flag in the serial status register (SCFSR2) is set to 1, a transmit-FIFO-data-empty request is generated separately from the interrupt request. A transmit-FIFO-data-empty request can activate the DMAC to perform data transfer.


When the RDF flag or DR flag in SCFSR2 is set to 1, a receive-FIFO-data-full request is generated separately from the interrupt request. A receive-FIFO-data-full request can activate the DMAC to perform data transfer.

When using the DMAC for transmission/reception, set and enable the DMAC before making the SCIF settings. See section 14, Direct Memory Access Controller (DMAC), for details of the DMAC setting procedure.

When the BRK flag in SCFSR2 or the ORER flag in the line status register (SCLSR2) is set to 1, a BRI interrupt request is generated.

The TXI interrupt indicates that transmit data can be written, and the RXI interrupt indicates that there is receive data in SCFRDR2.

Table 16.6 SCIF Interrupt Sources

| Interrupt Source | Description | DMAC Activation | Priority on Reset Release |
|------------------|--|-----------------|---|
| ERI | Interrupt initiated by receive error flag (ER) | Not possible | High |
| RXI | Interrupt initiated by receive FIFO data full flag (RDF) or receive data ready flag (DR) | Possible |  |
| BRI | Interrupt initiated by break flag (BRK) or overrun error flag (ORER) | Not possible | |
| TXI | Interrupt initiated by transmit FIFO data empty flag (TDFE) | Possible | |
| | | | |

See section 5, Exceptions, for priorities and the relationship with non-SCIF interrupts.

16.5 Usage Notes

Note the following when using the SCIF.

SCFTDR2 Writing and the TDFE Flag: The TDFE flag in the serial status register (SCFSR2) is set when the number of transmit data bytes written in the transmit FIFO data register (SCFTDR2) has fallen to or below the transmit trigger number set by bits TTRG1 and TTRG0 in the FIFO control register (SCFCR2). After TDFE is set, transmit data up to the number of empty bytes in SCFTDR2 can be written, allowing efficient continuous transmission.

However, if the number of data bytes written in SCFTDR2 is equal to or less than the transmit trigger number, the TDFE flag will be set to 1 again after being read as 1 and cleared to 0. TDFE clearing should therefore be carried out when SCFTDR2 contains more than the transmit trigger number of transmit data bytes.

The number of transmit data bytes in SCFTDR2 can be found from the upper 8 bits of the FIFO data count register (SCFDR2).

SCFRDR2 Reading and the RDF Flag: The RDF flag in the serial status register (SCFSR2) is set when the number of receive data bytes in the receive FIFO data register (SCFRDR2) has become equal to or greater than the receive trigger number set by bits RTRG1 and RTRG0 in the FIFO control register (SCFCR2). After RDF is set, receive data equivalent to the trigger number can be read from SCFRDR2, allowing efficient continuous reception.

However, if the number of data bytes in SCFRDR2 is equal to or greater than the trigger number, the RDF flag will be set to 1 again if it is cleared to 0. RDF should therefore be cleared to 0 after being read as 1 after all the receive data has been read.

The number of receive data bytes in SCFRDR2 can be found from the lower 8 bits of the FIFO data count register (SCFDR2).

Break Detection and Processing: Break signals can be detected by reading the RxD2 pin directly when a framing error (FER) is detected. In the break state the input from the RxD2 pin consists of all 0s, so the FER flag is set and the parity error flag (PER) may also be set.

Although the SCIF stops transferring receive data to SCFRDR2 after receiving a break, the receive operation continues.

Sending a Break Signal: The input/output condition and level of the TxD2 pin are determined by bits SPB2IO and SPB2DT in the serial port register (SCSPTR2). This feature can be used to send a break signal.

After the serial transmitter is initialized, the TxD2 pin function is not selected and the value of the SPB2DT bit substitutes for the mark state until the TE bit is set to 1 (i.e. transmission is enabled). The SPB2IO and SPB2DT bits should therefore be set to 1 (designating output and high level) beforehand.

To send a break signal during serial transmission, clear the SPB2DT bit to 0 (designating low level), then clear the TE bit to 0 (halting transmission). When the TE bit is cleared to 0, the transmitter is initialized, regardless of its current state, and 0 is output from the TxD2 pin.

Receive Data Sampling Timing and Receive Margin: The SCIF operates on a base clock with a frequency of 16 times the bit rate. In reception, the SCIF synchronizes internally with the fall of the start bit, which it samples on the base clock. Receive data is latched at the rising edge of the eighth base clock pulse. The timing is shown in figure 16.14.

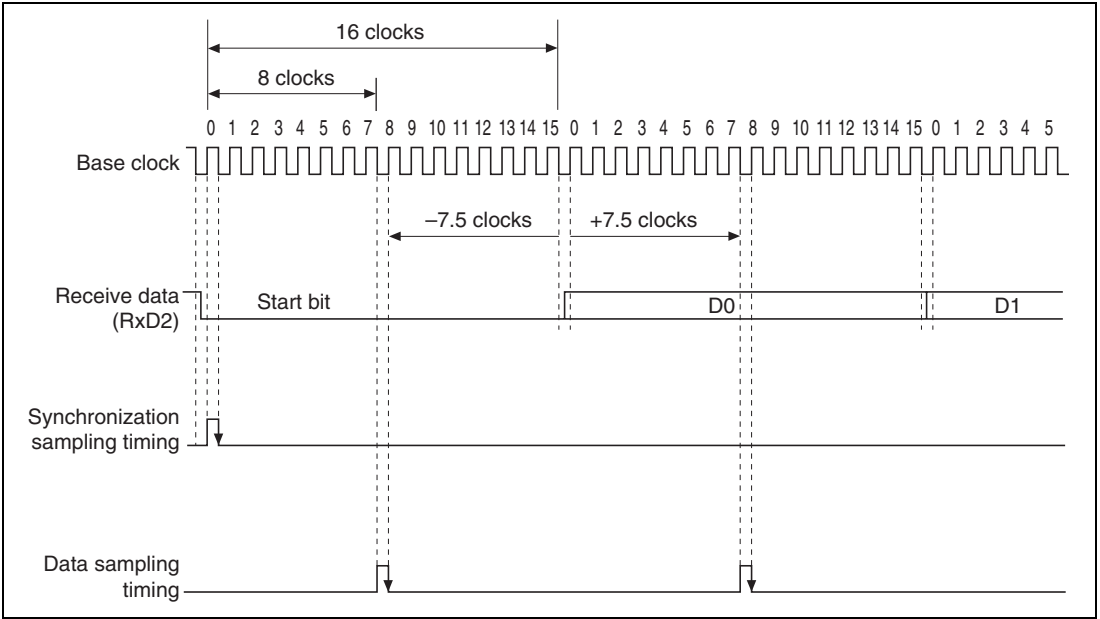


Figure 16.14 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

$$M = \left| \left(0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\% \dots\dots\dots (1)$$

Legend:

- M: Receive margin (%)
- N: Ratio of clock frequency to bit rate (N = 16)
- D: Clock duty cycle (D = 0 to 1.0)
- L: Frame length (L = 9 to 12)
- F: Absolute deviation of clock frequency

From equation (1), if F = 0 and D = 0.5, the receive margin is 46.875%, as given by equation (2).

When D = 0.5 and F = 0:

$$M = \left(0.5 - 1 / (2 \times 16) \right) \times 100\% = 46.875\% \dots\dots\dots (2)$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

When Using the DMAC: When using the DMAC for transmission/reception, inhibit output of RXI and TXI interrupt requests to the interrupt controller. If interrupt request output is enabled, interrupt requests to the interrupt controller will be cleared by the DMAC without regard to the interrupt handler.

Serial Ports: Note that, when the SCIF pin value is read using a serial port, the value read will be the value two peripheral clock cycles earlier.

Section 17 Smart Card Interface

17.1 Overview

The serial communication interface (SCI) supports a subset of the ISO/IEC 7816-3 (identification cards) standard as an extended function.

Switching between the normal serial communication interface and the smart card interface is carried out by means of a register setting.

17.1.1 Features

Features of the smart card interface are listed below.

- Asynchronous mode
 - Data length: 8 bits
 - Parity bit generation and checking
 - Transmission of error signal (parity error) in receive mode
 - Error signal detection and automatic data retransmission in transmit mode
 - Direct convention and inverse convention both supported
- On-chip baud rate generator allows any bit rate to be selected
- Three interrupt sources

There are three interrupt sources—transmit-data-empty, receive-data-full, and transmit/receive error—that can issue requests independently.

The transmit-data-empty interrupt and receive-data-full interrupt can activate the DMA controller (DMAC) to execute data transfer.

17.1.2 Block Diagram

Figure 17.1 shows a block diagram of the smart card interface.

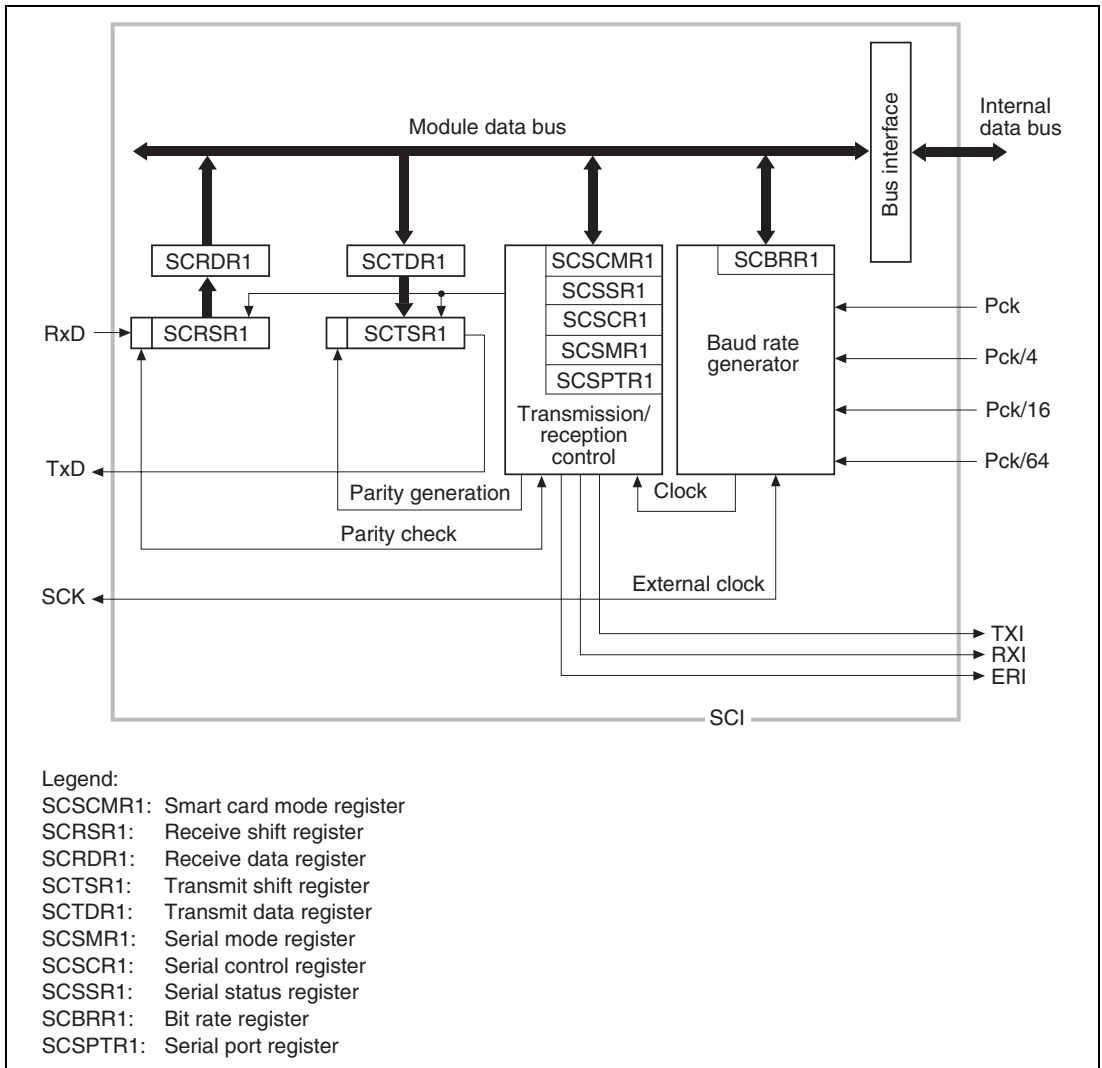


Figure 17.1 Block Diagram of Smart Card Interface

17.1.3 Pin Configuration

Table 17.1 shows the smart card interface pin configuration.

Table 17.1 Smart Card Interface Pins

| Pin Name | Abbreviation | I/O | Function |
|-------------------|--------------|--------|----------------------|
| Serial clock pin | SCK | I/O | Clock input/output |
| Receive data pin | RxD | Input | Receive data input |
| Transmit data pin | TxD | Output | Transmit data output |

17.1.4 Register Configuration

The smart card interface has the internal registers shown in table 17.2. Details of the SCBRR1, SCTDR1, SCRDR1, and SCSPTR1 registers are the same as for the normal SCI function: see the register descriptions in section 15, Serial Communication Interface (SCI).

With the exception of the serial port register, the smart card interface registers are initialized in standby mode and in the module standby state as well as by a power-on reset or manual reset. When recovering from standby mode or the module standby state, the registers must be set again.

Table 17.2 Smart Card Interface Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|--------------------------|--------------|---------------------|--------------------|------------|----------------|-------------|
| Serial mode register | SCSMR1 | R/W | H'00 | H'FFE00000 | H'1FE00000 | 8 |
| Bit rate register | SCBRR1 | R/W | H'FF | H'FFE00004 | H'1FE00004 | 8 |
| Serial control register | SCSCR1 | R/W | H'00 | H'FFE00008 | H'1FE00008 | 8 |
| Transmit data register | SCTDR1 | R/W | H'FF | H'FFE0000C | H'1FE0000C | 8 |
| Serial status register | SCSSR1 | R/(W)* ¹ | H'84 | H'FFE00010 | H'1FE00010 | 8 |
| Receive data register | SCRDR1 | R | H'00 | H'FFE00014 | H'1FE00014 | 8 |
| Smart card mode register | SCSCMR1 | R/W | H'00 | H'FFE00018 | H'1FE00018 | 8 |
| Serial port register | SCSPTR1 | R/W | H'00* ² | H'FFE0001C | H'1FE0001C | 8 |

Notes: 1. Only 0 can be written, to clear flags.
2. The value of bits 2 and 0 is undefined.

17.2 Register Descriptions

Only registers that have been added, and bit functions that have been modified, for the smart card interface are described here.

17.2.1 Smart Card Mode Register (SCSCMR1)

SCSCMR1 is an 8-bit readable/writable register that selects the smart card interface function. SCSCMR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

| | | | | | | | | |
|----------------|---|---|---|---|------|------|---|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value: | — | — | — | — | 0 | 0 | — | 0 |
| R/W: | — | — | — | — | R/W | R/W | — | R/W |

Bits 7 to 4 and 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Smart Card Data Transfer Direction (SDIR): Selects the serial/parallel conversion format.

| Bit 3: SDIR | Description |
|-------------|---|
| 0 | SCTDR1 contents are transmitted LSB-first (Initial value) Receive data is stored in SCRDR1 LSB-first |
| 1 | SCTDR1 contents are transmitted MSB-first Receive data is stored in SCRDR1 MSB-first |

Bit 2—Smart Card Data Invert (SINV): Specifies inversion of the data logic level. This function is used together with the bit 3 function for communication with an inverse convention card. The SINV bit does not affect the logic level of the parity bit. For parity-related setting procedures, see section 17.3.4, Register Settings.

| Bit 2: SINV | Description |
|-------------|--|
| 0 | SCTDR1 contents are transmitted as they are (Initial value) Receive data is stored in SCRDR1 as it is |
| 1 | SCTDR1 contents are inverted before being transmitted Receive data is stored in SCRDR1 in inverted form |

Bit 0—Smart Card Interface Mode Select (SMIF): Enables or disables the smart card interface function.

| Bit 0: SMIF | Description |
|-------------|---|
| 0 | Smart card interface function is disabled (Initial value) |
| 1 | Smart card interface function is enabled |

17.2.2 Serial Mode Register (SCSMR1)

Bit 7 of SCSMR1 has a different function in smart card interface mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-------------------|-----|-----|--------------|------|-----|------|------|
| | GM(C/ \bar{A}) | CHR | PE | O/ \bar{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—GSM Mode (GM): Sets the smart card interface function to GSM mode.

With the normal smart card interface, this bit is cleared to 0. Setting this bit to 1 selects GSM mode, an additional mode for controlling the timing for setting the TEND flag that indicates completion of transmission, and the type of clock output used. The details of the additional clock output control mode are specified by the CKE1 and CKE0 bits in the serial control register (SCSCR1). In GSM mode, the pulse width is guaranteed when SCK start/stop specifications are made by CKE1 and CKE0.

| Bit 7: GM | Description |
|-----------|---|
| 0 | Normal smart card interface mode operation (Initial value) <ul style="list-style-type: none"> The TEND flag is set 12.5 etu after the beginning of the start bit Clock output on/off control only |
| 1 | GSM mode smart card interface mode operation <ul style="list-style-type: none"> The TEND flag is set 11.0 etu after the beginning of the start bit Clock output on/off and fixed-high/fixed-low control (set in SCSCR1) |

Note: etu: Elementary time unit (time for transfer of 1 bit)

Bits 6 to 0: Operate in the same way as for the normal SCI. See section 15, Serial Communication Interface (SCI), for details. With the smart card interface, the following settings should be used: CHR = 0, PE = 1, STOP = 1, MP = 0.

17.2.3 Serial Control Register (SCSCR1)

Bits 1 and 0 of SCSCR1 have a different function in smart card interface mode.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TIE | RIE | TE | RE | — | — | 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 |

Bits 7 to 4: Operate in the same way as for the normal SCI. See section 15, Serial Communication Interface (SCI), for details.

Bits 3 and 2—Reserved: Not used with the smart card interface.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits specify the function of the SCK pin. In smart card interface mode, an internal clock is always used as the clock source. In smart card interface mode, it is possible to specify a fixed high level or fixed low level for the clock output, in addition to the usual switching between enabling and disabling of the clock output.

| GM | CKE1 | CKE0 | SCK Pin Function |
|----|------|------|-----------------------------------|
| 0 | 0 | 0 | Port I/O pin |
| | | 1 | Clock output as SCK output pin |
| | 1 | 0 | Invalid setting: must not be used |
| | | 1 | Invalid setting: must not be used |
| 1 | 0 | 0 | Output pin with output fixed low |
| | | 1 | Clock output as output pin |
| | 1 | 0 | Output pin with output fixed high |
| | | 1 | Clock output as output pin |

17.2.4 Serial Status Register (SCSSR1)

Bit 4 of SCSSR1 has a different function in smart card interface mode. Coupled with this, the setting conditions for bit 2 (TEND) are also different.

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

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

Bits 7 to 5: Operate in the same way as for the normal SCI. See section 15, Serial Communication Interface (SCI), for details.

Bit 4—Error Signal Status (ERS): In smart card interface mode, bit 4 indicates the status of the error signal sent back from the receiving side during transmission. Framing errors are not detected in smart card interface mode.

| Bit 4: ERS | Description |
|------------|--|
| 0 | Normal reception, no error signal (Initial value) [Clearing conditions] <ul style="list-style-type: none"> Power-on reset, manual reset, standby mode, or module standby When 0 is written to ERS after reading ERS = 1 |
| 1 | An error signal has been sent from the receiving side indicating detection of a parity error [Setting condition] When the low level of the error signal is detected |

Note: Clearing the TE bit in SCSSR1 to 0 does not affect the ERS flag, which retains its previous state.

Bit 3—Parity Error (PER): Operates in the same way as for the normal SCI. See section 15, Serial Communication Interface (SCI), for details.

Bit 2—Transmit End (TEND): The setting conditions for the TEND flag are as follows.

| Bit 2: TEND | Description |
|-------------|---|
| 0 | Transmission in progress [Clearing condition] When 0 is written to TDRE after reading TDRE = 1 |
| 1 | Transmission has been ended (Initial value) [Setting conditions] <ul style="list-style-type: none"> • Power-on reset, manual reset, standby mode, or module standby • When the TE bit in SCSCR1 is 0 and the FER/ERS bit is also 0 • When the GM bit in SCSMR1 is 0, and TDRE = 1 and FER/ERS = 0 (normal transmission) 2.5 etu after transmission of a 1-byte serial character • When the GM bit in SCSMR1 is 1, and TDRE = 1 and FER/ERS = 0 (normal transmission) 1.0 etu after transmission of a 1-byte serial character |

Note: etu: Elementary time unit (time for transfer of 1 bit)

Bits 1 and 0—Reserved: Not used with the smart card interface.

17.3 Operation

17.3.1 Overview

The main functions of the smart card interface are as follows.

- One frame consists of 8-bit data plus a parity bit.
- In transmission, a guard time of at least 2 etu (elementary time unit: the time for transfer of one bit) is left between the end of the parity bit and the start of the next frame.
- If a parity error is detected during reception, a low error signal level is output for a 1-etu period 10.5 etu after the start bit.
- If an error signal is detected during transmission, the same data is transmitted automatically after the elapse of 2 etu or longer.
- Only asynchronous communication is supported; there is no synchronous communication function.

17.3.2 Pin Connections

Figure 17.2 shows a schematic diagram of smart card interface related pin connections.

In communication with an IC card, since both transmission and reception are carried out on a single data transmission line, the TxD pin and RxD pin should be connected outside the chip. The data transmission line should be pulled up on the V_{CC} power supply side with a resistor.

When the clock generated on the smart card interface is used by an IC card, the SCK pin output is input to the CLK pin of the IC card. No connection is needed if the IC card uses an internal clock.

Chip port output is used as the reset signal.

Other pins must normally be connected to the power supply or ground.

Note: If an IC card is not connected, and both TE and RE are set to 1, closed transmission/reception is possible, enabling self-diagnosis to be carried out.

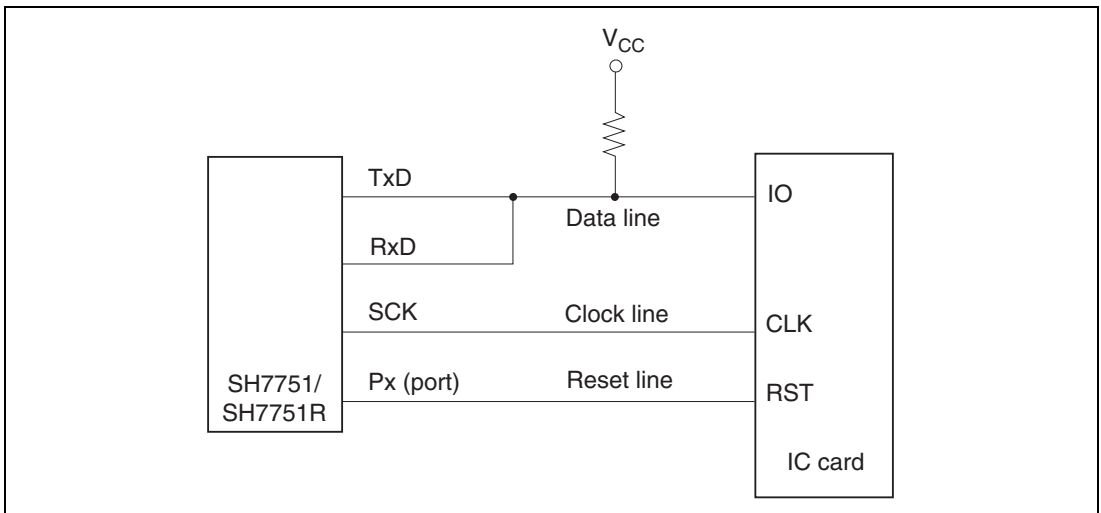


Figure 17.2 Schematic Diagram of Smart Card Interface Pin Connections

17.3.3 Data Format

Figure 17.3 shows the smart card interface data format. In reception in this mode, a parity check is carried out on each frame, and if an error is detected an error signal is sent back to the transmitting side to request retransmission of the data. If an error signal is detected during transmission, the same data is retransmitted.

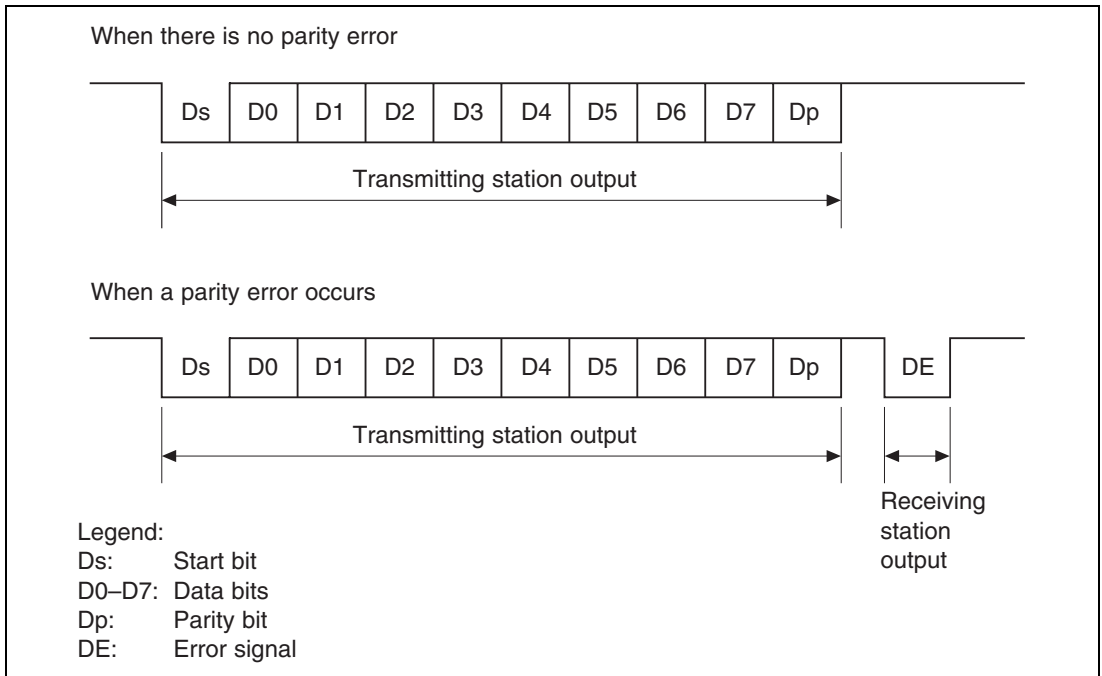


Figure 17.3 Smart Card Interface Data Format

The operation sequence is as follows.

1. When the data line is not in use it is in the high-impedance state, and is fixed high with a pull-up resistor.
2. The transmitting station starts transmission of one frame of data. The data frame starts with a start bit (Ds, low-level), followed by 8 data bits (D0 to D7) and a parity bit (Dp).
3. With the smart card interface, the data line then returns to the high-impedance state. The data line is pulled high with a pull-up resistor.
4. The receiving station carries out a parity check.

If there is no parity error and the data is received normally, the receiving station waits for reception of the next data.

If a parity error occurs, however, the receiving station outputs an error signal (DE, low-level) to request retransmission of the data. After outputting the error signal for the prescribed length of time, the receiving station places the signal line in the high-impedance state again. The signal line is pulled high again by a pull-up resistor.

5. If the transmitting station does not receive an error signal, it proceeds to transmit the next data frame.

If it receives an error signal, however, it returns to step 2 and retransmits the erroneous data.

17.3.4 Register Settings

Table 17.3 shows a bit map of the registers used by the smart card interface. Bits indicated as 0 or 1 must be set to the value shown. The setting of other bits is described below.

Table 17.3 Smart Card Interface Register Settings

| Register | Bit | | | | | | | |
|----------|-------|-------|-------|-------------|--------|--------|--------|--------|
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| SCSMR1 | GM | 0 | 1 | O/\bar{E} | 1 | 0 | CKS1 | CKS0 |
| SCBRR1 | BRR7 | BRR6 | BRR5 | BRR4 | BRR3 | BRR2 | BRR1 | BRR0 |
| SCSCR1 | TIE | RIE | TE | RE | 0 | 0 | CKE1 | CKE0 |
| SCTDR1 | TDR7 | TDR6 | TDR5 | TDR4 | TDR3 | TDR2 | TDR1 | TDR0 |
| SCSSR1 | TDRE | RDRF | ORER | FER/ERS | PER | TEND | 0 | 0 |
| SCRDR1 | RDR7 | RDR6 | RDR5 | RDR4 | RDR3 | RDR2 | RDR1 | RDR0 |
| SCSCMR1 | — | — | — | — | SDIR | SINV | — | SMIF |
| SCSPTR1 | EIO | — | — | — | SPB1IO | SPB1DT | SPB0IO | SPB0DT |

Note: A dash indicates an unused bit.

Serial Mode Register (SCSMR1) Settings: The GM bit is used to select the timing of TEND flag setting, and, together with the CKE1 and CKE0 bits in the serial control register (SCSCR1), to select the clock output state.

The O/\bar{E} bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

Bits CKS1 and CKS0 select the clock source of the on-chip baud rate generator. See section 17.3.5, Clock.

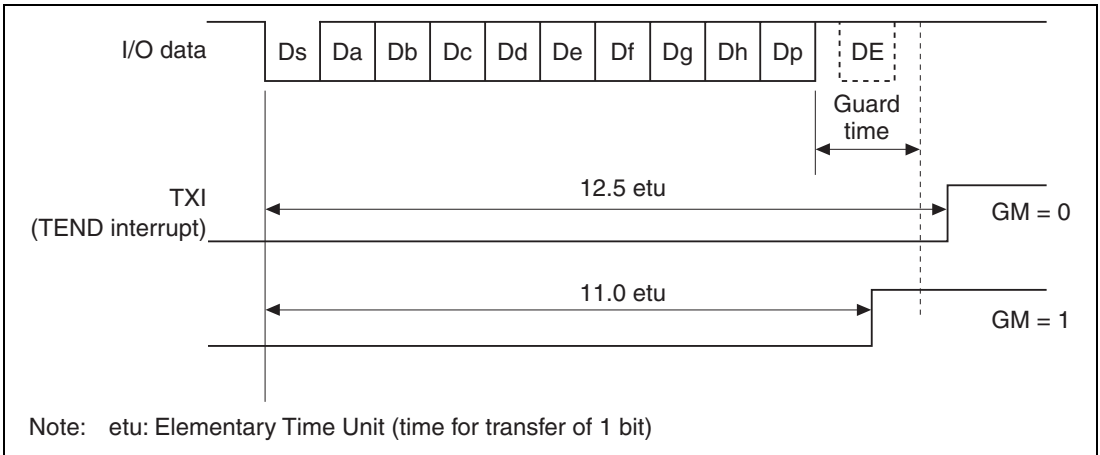


Figure 17.4 TEND Generation Timing

Bit Rate Register (SCBRR1) Setting: SCBRR1 is used to set the bit rate. See section 17.3.5, Clock, for the method of calculating the value to be set.

Serial Control Register (SCSCR1) Settings: The function of the TIE, RIE, TE, and RE bits is the same as for the normal SCI. See section 15, Serial Communication Interface (SCI), for details.

The CKE1 and CKE0 bits specify the clock output state. See section 17.3.5, Clock, for details.

Smart Card Mode Register (SCSCMR1) Settings: The SDIR bit and SINV bit are both cleared to 0 if the IC card is of the direct convention type, and both set to 1 if of the inverse convention type.

The SMIF bit is set to 1 when the smart card interface is used.

Figure 17.5 shows examples of register settings and the waveform of the start character for the two types of IC card (direct convention and inverse convention).

With the direct convention type, the logic 1 level corresponds to state Z and the logic 0 level to state A, and transfer is performed in LSB-first order. The start character data in this case is H'3B. The parity bit is 1 since even parity is stipulated for the smart card.

With the inverse convention type, the logic 1 level corresponds to state A and the logic 0 level to state Z, and transfer is performed in MSB-first order. The start character data in this case is H'3F. The parity bit is 0, corresponding to state Z, since even parity is stipulated for the smart card.

Inversion specified by the SINV bit applies only to the data bits, D7 to D0. For parity bit inversion, the $O\bar{E}$ bit in SCSMR1 is set to odd parity mode. (This applies to both transmission and reception).

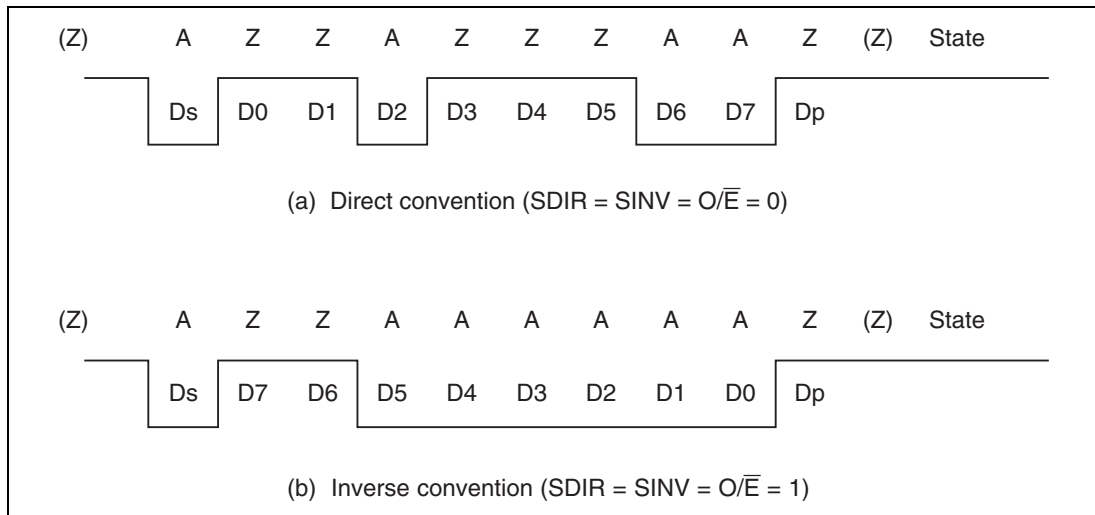


Figure 17.5 Sample Start Character Waveforms

17.3.5 Clock

Only an internal clock generated by the on-chip baud rate generator can be used as the transmit/receive clock for the smart card interface. The bit rate is set with the bit rate register (SCBRR1) and the CKS1 and CKS0 bits in the serial mode register (SCSMR1). The equation for calculating the bit rate is shown below. Table 17.5 shows some sample bit rates.

If clock output is selected with CKE0 set to 1, a clock with a frequency of 372 times the bit rate is output from the SCK pin.

$$B = \frac{Pck}{1488 \times 2^{2n-1} \times (N+1)} \times 10^6$$

Where: N = Value set in SCBRR1 ($0 \leq N \leq 255$)

B = Bit rate (bits/s)

Pck = Peripheral module operating frequency (MHz)

n = 0 to 3 (See table 17.4)

Table 17.4 Values of n and Corresponding CKS1 and CKS0 Settings

| n | CKS1 | CKS0 |
|---|------|------|
| 0 | 0 | 0 |
| 1 | 0 | 1 |
| 2 | 1 | 0 |
| 3 | 1 | 1 |

Table 17.5 Examples of Bit Rate B (bits/s) for Various SCBRR1 Settings (When n = 0)

| N | Pck (MHz) | | | | | | |
|---|-----------|---------|---------|---------|---------|---------|---------|
| | 7.1424 | 10.00 | 10.7136 | 14.2848 | 25.0 | 33.0 | 50.0 |
| 0 | 9600.0 | 13440.9 | 14400.0 | 19200.0 | 33602.2 | 44354.8 | 67204.3 |
| 1 | 4800.0 | 6720.4 | 7200.0 | 9600.0 | 16801.1 | 22177.4 | 33602.2 |
| 2 | 3200.0 | 4480.3 | 4800.0 | 6400.0 | 11200.7 | 14784.9 | 22401.4 |

Note: Bit rates are rounded to one decimal place.

The method of calculating the value to be set in the bit rate register (SCBRR1) from the peripheral module operating frequency and bit rate is shown below. Here, N is an integer in the range $0 \leq N \leq 255$, and the smaller error is specified.

$$N = \frac{Pck}{1488 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Table 17.6 Examples of SCBRR1 Settings for Bit Rate B (bits/s) (When n = 0)

| Bits/s | Pck (MHz) | | | | | | | | | | | | | |
|--------|-----------|-------|-------|-------|---------|-------|---------|-------|-------|-------|-------|-------|-------|-------|
| | 7.1424 | | 10.00 | | 10.7136 | | 14.2848 | | 25.00 | | 33.00 | | 50.00 | |
| | N | Error | N | Error | N | Error | N | Error | N | Error | N | Error | N | Error |
| 9600 | 0 | 0.00 | 1 | 30.00 | 1 | 25.00 | 1 | 8.99 | 3 | 14.27 | 4 | 8.22 | 6 | 0.01 |

Table 17.7 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode)




| Pck (MHz) | Maximum Bit Rate (bits/s) | N | n |
|-----------|---------------------------|---|---|
| 7.1424 | 19200 | 0 | 0 |
| 10.00 | 26882 | 0 | 0 |
| 10.7136 | 28800 | 0 | 0 |
| 16.00 | 43010 | 0 | 0 |
| 20.00 | 53763 | 0 | 0 |
| 25.0 | 67204 | 0 | 0 |
| 30.0 | 80645 | 0 | 0 |
| 33.0 | 88710 | 0 | 0 |
| 50.0 | 67204 | 0 | 0 |

The bit rate error is given by the following equation:

$$\text{Error (\%)} = \left\{ \frac{\text{Pck}}{1488 \times 2^{2n-1} \times B \times (N+1)} \right\} \times 10^6 - 1 \times 100$$

Table 17.8 shows the relationship between the smart card interface transmit/receive clock register settings and the output state.

Table 17.8 Register Settings and SCK Pin State

| Setting | Register Values | | | | SCK Pin | |
|-----------------|-----------------|----|------|------|---|--|
| | SMIF | GM | CKE1 | CKE0 | Output | State |
| 1* ¹ | 1 | 0 | 0 | 0 | Port | Determined by setting of SPB1IO and SPB1DT bits in SCSPTR1 |
| | 1 | 0 | 0 | 1 |  | SCK (serial clock) output state |
| 2* ² | 1 | 1 | 0 | 0 | Low output | Low-level output state |
| | 1 | 1 | 0 | 1 |  | SCK (serial clock) output state |
| 3* ² | 1 | 1 | 1 | 0 | High output | High-level output state |
| | 1 | 1 | 1 | 1 |  | SCK (serial clock) output state |

Notes: 1. The SCK output state changes as soon as the CKE0 bit setting is changed. Clear the CKE1 bit to 0.

2. Stopping and starting the clock by changing the CKE0 bit setting does not affect the clock duty cycle.

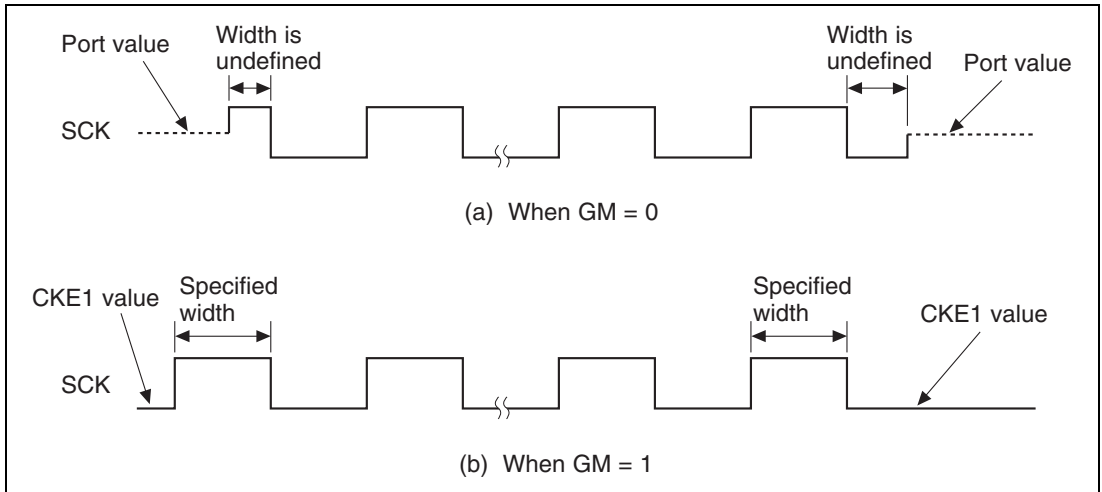


Figure 17.6 Difference in Clock Output According to GM Bit Setting

17.3.6 Data Transfer Operations

Initialization: Before transmitting and receiving data, the smart card interface must be initialized as described below. Initialization is also necessary when switching from transmit mode to receive mode, or vice versa. Figure 17.7 shows a sample initialization processing flowchart.

1. Clear the TE and RE bits in the serial control register (SCSCR1) to 0.
2. Clear error flags FER/ERS, PER, and ORER in the serial status register (SCSSR1) to 0.
3. Set the GM bit, parity bit (O/\bar{E}), and baud rate generator select bits (CKS1 and CKS0) in the serial mode register (SCSMR1). Clear the CHR and MP bits to 0, and set the STOP and PE bits to 1.
4. Set the SMIF, SDIR, and SINV bits in the smart card mode register (SCSCMR1).
When the SMIF bit is set to 1, the TxD pin and RxD pin both go to the high-impedance state.
5. Set the value corresponding to the bit rate in the bit rate register (SCBRR1).
6. Set the clock source select bits (CKE1 and CKE0) in SCSCR1. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0.
If the CKE0 bit is set to 1, the clock is output from the SCK pin.
7. Wait at least one bit interval, then set the TIE, RIE, TE, and RE bits in SCSCR1. Do not set the TE bit and RE bit at the same time, except for self-diagnosis.

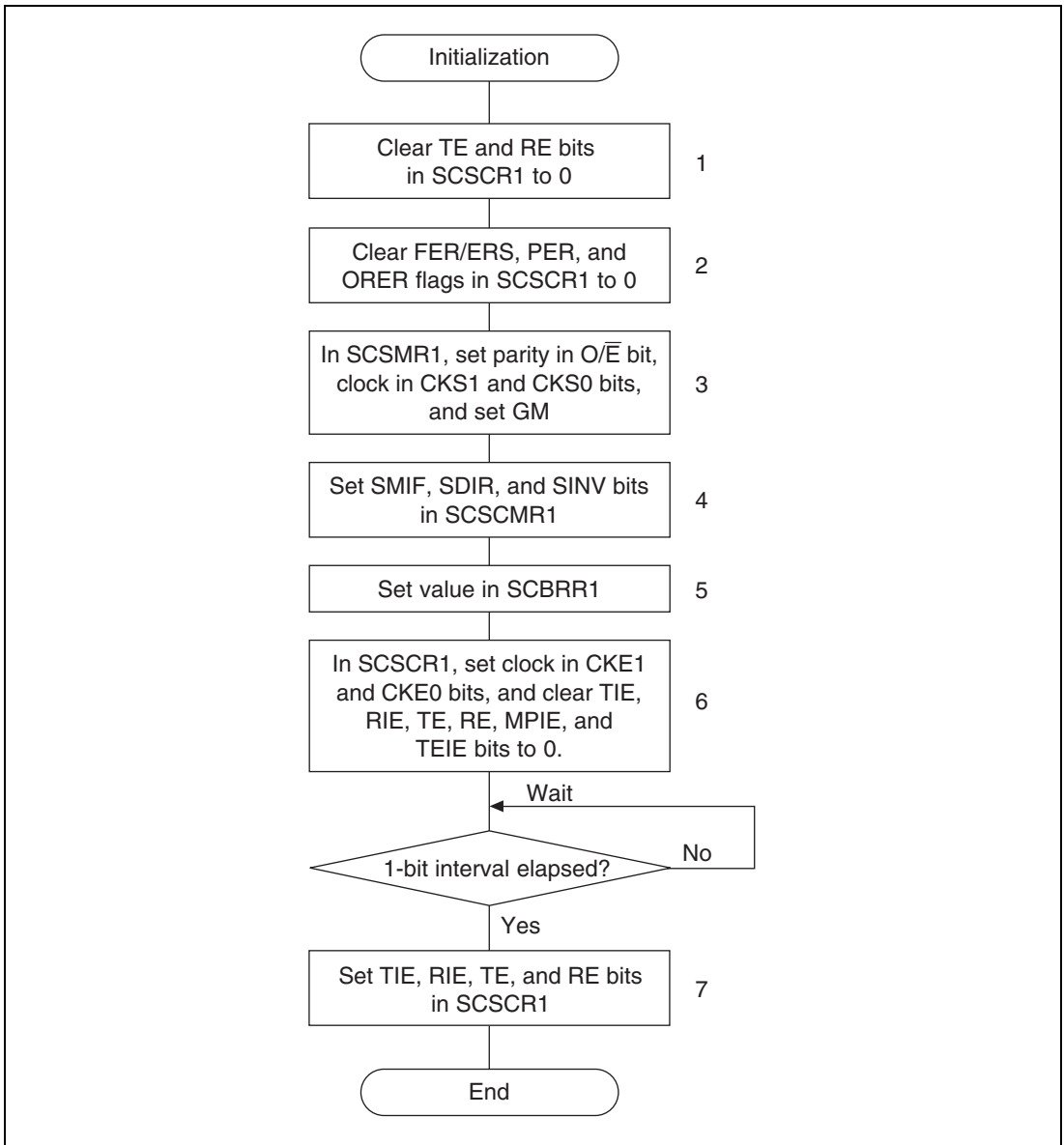


Figure 17.7 Sample Initialization Flowchart

Serial Data Transmission: As data transmission in smart card mode involves error signal sampling and retransmission processing, the processing procedure is different from that for the normal SCI. Figure 17.8 shows a sample transmission processing flowchart.

1. Perform smart card interface mode initialization as described in Initialization above.
2. Check that the FER/ERS error flag in SCSSR1 is cleared to 0.
3. Repeat steps 2 and 3 until it can be confirmed that the TEND flag in SCSSR1 is set to 1.
4. Write the transmit data to SCTDR1, clear the TDRE flag to 0, and perform the transmit operation. The TEND flag is cleared to 0.
5. To continue transmitting data, go back to step 2.
6. To end transmission, clear the TE bit to 0.

With the above processing, interrupt handling is possible.

If transmission ends and the TEND flag is set to 1 while the TIE bit is set to 1 and interrupt requests are enabled, a transmit-data-empty interrupt (TXI) request will be generated. If an error occurs in transmission and the ERS flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a transmit/receive-error interrupt (ERI) request will be generated. See Interrupt Operation below for details.

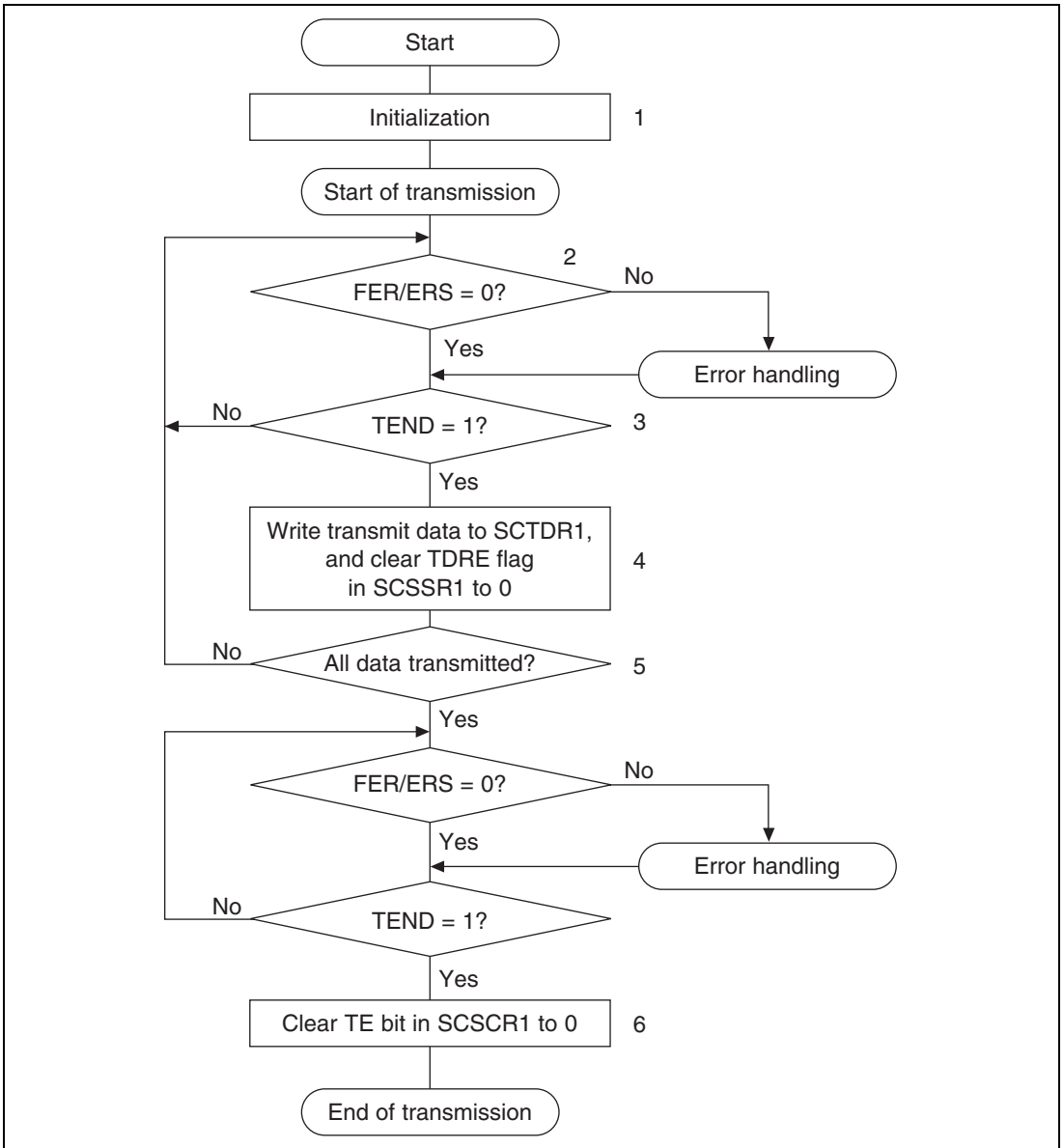


Figure 17.8 Sample Transmission Processing Flowchart

Serial Data Reception: Data reception in smart card mode uses the same processing procedure as for the normal SCI. Figure 17.9 shows a sample reception processing flowchart.

1. Perform smart card interface mode initialization as described in Initialization above.
2. Check that the ORER flag and PER flag in SCSSR1 are cleared to 0. If either is set, perform the appropriate receive error handling, then clear both the ORER and the PER flag to 0.
3. Repeat steps 2 and 3 until it can be confirmed that the RDRF flag is set to 1.
4. Read the receive data from SCRDR1.
5. To continue receiving data, clear the RDRF flag to 0 and go back to step 2.
6. To end reception, clear the RE bit to 0.

With the above processing, interrupt handling is possible.

If reception ends and the RDRF flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a receive-data-full interrupt (RXI) request will be generated. If an error occurs in reception and either the ORER flag or the PER flag is set to 1, a transmit/receive-error interrupt (ERI) request will be generated.

See Interrupt Operation below for details.

If a parity error occurs during reception and the PER flag is set to 1, the received data is still transferred to SCRDR1, and therefore this data can be read.

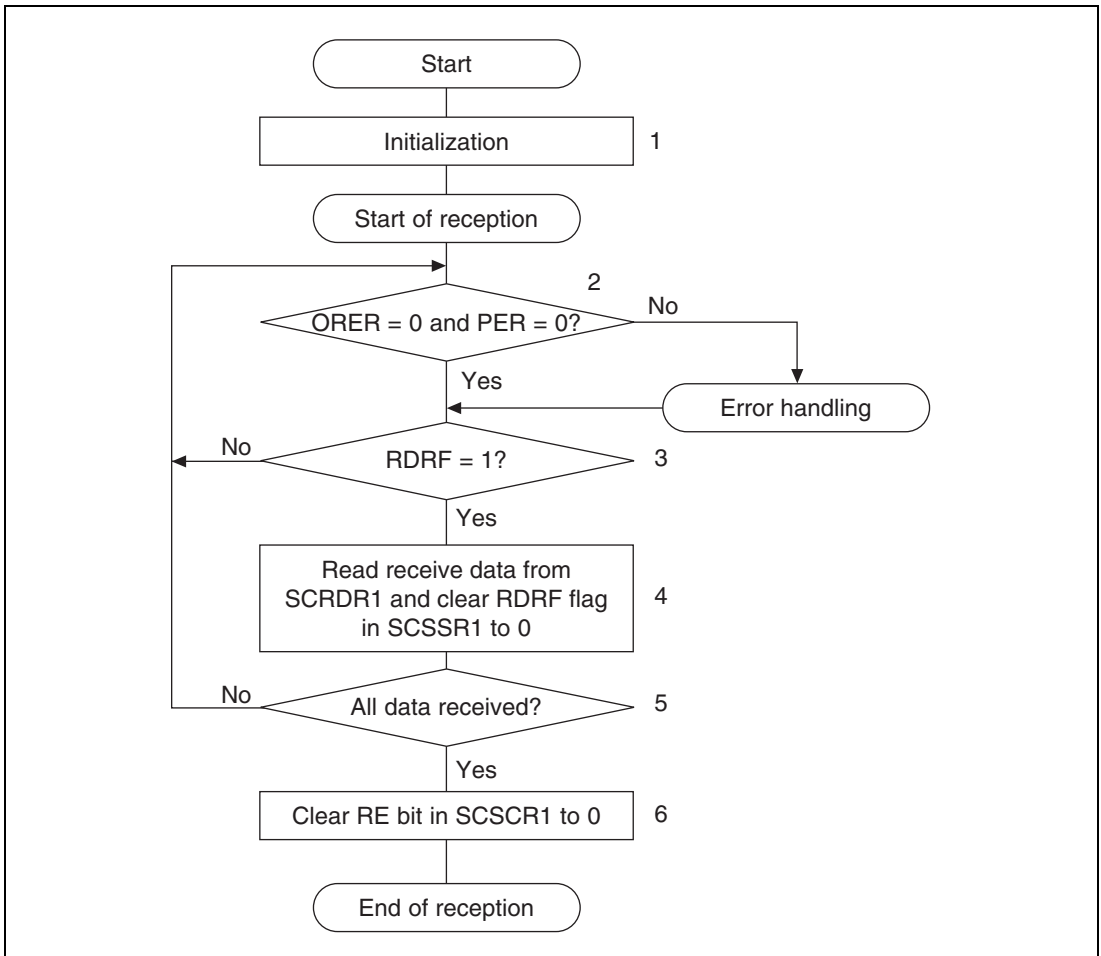


Figure 17.9 Sample Reception Processing Flowchart

Mode Switching Operation: When switching from receive mode to transmit mode, first confirm that the receive operation has been completed, then start from initialization, clearing RE to 0 and setting TE to 1. The RDRF flag or the PER and ORER flags can be used to check that the receive operation has been completed.

When switching from transmit mode to receive mode, first confirm that the transmit operation has been completed, then start from initialization, clearing TE to 0 and setting RE to 1. The TEND flag can be used to check that the transmit operation has been completed.

Interrupt Operation: There are three interrupt sources in smart card interface mode, generating transmit-data-empty interrupt (TXI) requests, transmit/receive-error interrupt (ERI) requests, and receive-data-full interrupt (RXI) requests. The transmit-end interrupt (TEI) request cannot be used in this mode.

When the TEND flag in SCSSR1 is set to 1, a TXI interrupt request is generated.

When the RDRF flag in SCSSR1 is set to 1, an RXI interrupt request is generated.

When any of flags ORER, PER, and FER/ERS in SCSSR1 is set to 1, an ERI interrupt request is generated. The relationship between the operating states and interrupt sources is shown in table 17.9.

Table 17.9 Smart Card Mode Operating States and Interrupt Sources

| Operating State | | Flag | Mask Bit | Interrupt Source |
|-----------------|------------------|-----------|----------|------------------|
| Transmit mode | Normal operation | TEND | TIE | TXI |
| | Error | FER/ERS | RIE | ERI |
| Receive mode | Normal operation | RDRF | RIE | RXI |
| | Error | PER, ORER | RIE | ERI |

Data Transfer Operation by DMAC: In smart card mode, as with the normal SCI, transfer can be carried out using the DMAC. In a transmit operation, when the TEND flag in SCSSR1 is set to 1, a TXI interrupt is requested. If the TXI request is designated beforehand as a DMAC activation source, the DMAC will be activated by the TXI request, and transfer of the transmit data will be carried out. The TEND flag is automatically cleared to 0 when data transfer is performed by the DMAC. In the event of an error, the SCI retransmits the same data automatically. The TEND flag remains cleared to 0 during this time, and the DMAC is not activated. Thus, the number of bytes specified by the SCI and DMAC are transmitted automatically, including retransmission following an error. However, the ERS flag is not cleared automatically when an error occurs, and therefore the RIE bit should be set to 1 beforehand so that an ERI request will be generated in the event of an error, and the ERS flag will be cleared.

In a receive operation, an RXI interrupt request is generated when the RDRF flag in SCSSR1 is set to 1. If the RXI request is designated beforehand as a DMAC activation source, the DMAC will be activated by the RXI request, and transfer of the receive data will be carried out. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DMAC. If an error occurs, an error flag is set but the RDRF flag is not. The DMAC is not activated, but instead, an ERI interrupt request is sent to the CPU. The error flag must therefore be cleared.

When performing data transfer using the DMAC, it is essential to set and enable the DMAC before carrying out SCI settings. For details of the DMAC setting procedures, see section 14, Direct Memory Access Controller (DMAC).

17.4 Usage Notes

The following points should be noted when using the SCI as a smart card interface.

(1) Receive Data Sampling Timing and Receive Margin

In asynchronous mode, the SCI operates on a base clock with a frequency of 372 times the transfer rate. In reception, the SCI synchronizes internally with the fall of the start bit, which it samples on the base clock. Receive data is latched at the rising edge of the 186th base clock pulse. The timing is shown in figure 17.10.

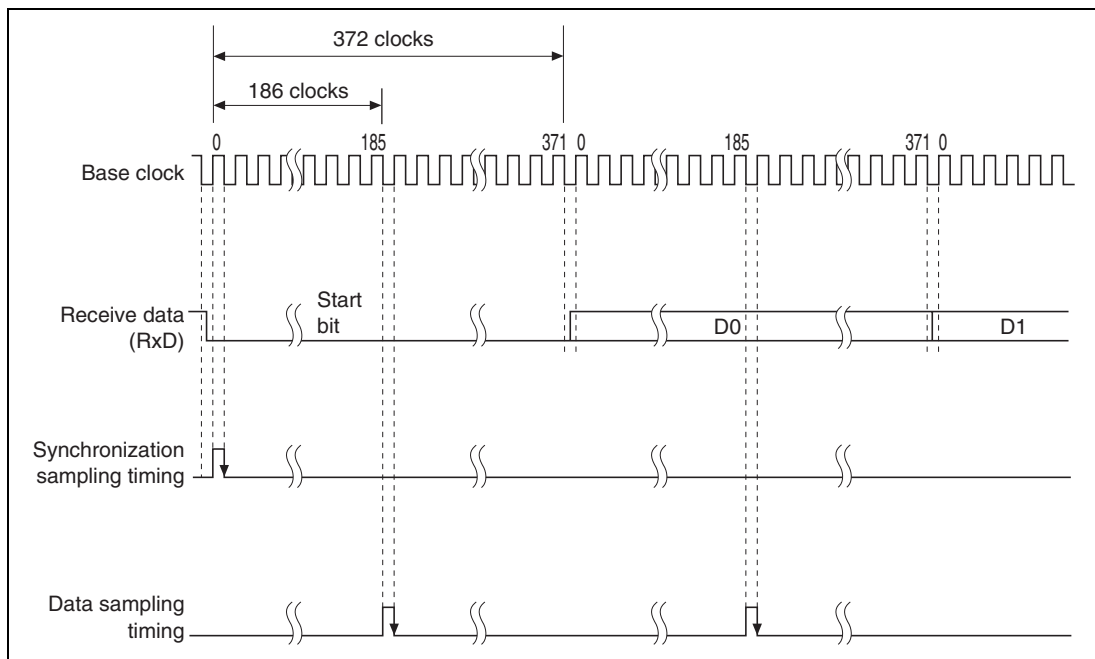


Figure 17.10 Receive Data Sampling Timing in Smart Card Mode

The receive margin in smart card mode can therefore be expressed as shown in the following equation.

$$M = \left| \left(0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \cdot 100\%$$

Legend:

M: Receive margin (%)

N: Ratio of clock frequency to bit rate (N = 372)

D: Clock duty cycle (D = 0 to 1.0)

L: Frame length (L = 10)

F: Absolute deviation of clock frequency

From the above equation, if F = 0 and D = 0.5, the receive margin is 49.866%, as given by the following equation.

When D = 0.5 and F = 0:

$$M = (0.5 - 1/2 \times 372) \times 100\% = 49.866\%$$

(2) Retransfer Operations

Retransfer operations are performed by the SCI in receive mode and transmit mode as described below.

Retransfer Operation when SCI is in Receive Mode: Figure 17.11 illustrates the retransfer operation when the SCI is in receive mode.

1. If an error is found when the received parity bit is checked, the PER bit in SCSSR1 is automatically set to 1. If the RIE bit in SCSCR1 is enabled at this time, an ERI interrupt request is generated. The PER bit in SCSSR1 should be cleared to 0 before the next parity bit is sampled.
2. The RDRF bit in SCSSR1 is not set for a frame in which an error has occurred.
3. If an error is found when the received parity bit is checked, the PER bit in SCSSR1 is not set to 1.
4. If no error is found when the received parity bit is checked, the receive operation is judged to have been completed normally, and the RDRF bit in SCSSR1 is automatically set to 1. If the RIE bit in SCSCR1 is enabled at this time, an RXI interrupt request is generated.
5. When a normal frame is received, the pin retains the high-impedance state at the timing for error signal transmission.

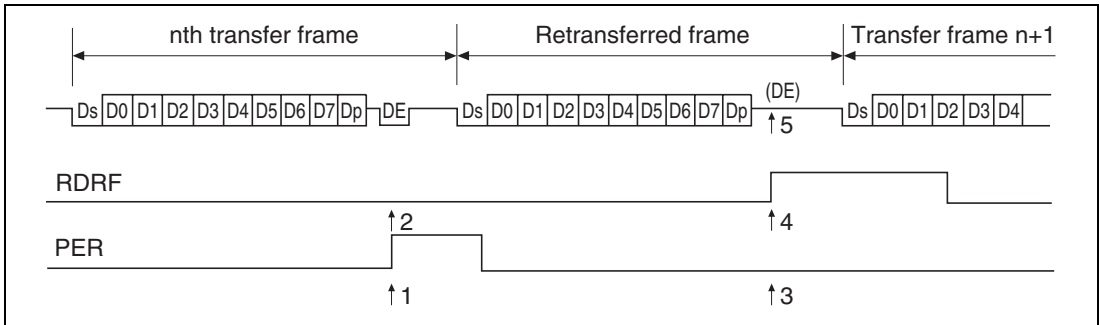


Figure 17.11 Retransfer Operation in SCI Receive Mode

Retransfer Operation when SCI is in Transmit Mode: Figure 17.12 illustrates the retransfer operation when the SCI is in transmit mode.

1. If an error signal is sent back from the receiving side after transmission of one frame is completed, the FER/ERS bit in SCSSR1 is set to 1. If the RIE bit in SCSCR1 is enabled at this time, an ERI interrupt request is generated. The FER/ERS bit in SCSSR1 should be cleared to 0 before the next parity bit is sampled.
2. The TEND bit in SCSSR1 is not set for a frame for which an error signal indicating an error is received.
3. If an error signal is not sent back from the receiving side, the FER/ERS bit in SCSSR1 is not set.
4. If an error signal is not sent back from the receiving side, transmission of one frame, including a retransfer, is judged to have been completed, and the TEND bit in SCSSR1 is set to 1. If the TIE bit in SCSCR1 is enabled at this time, a TXI interrupt request is generated.

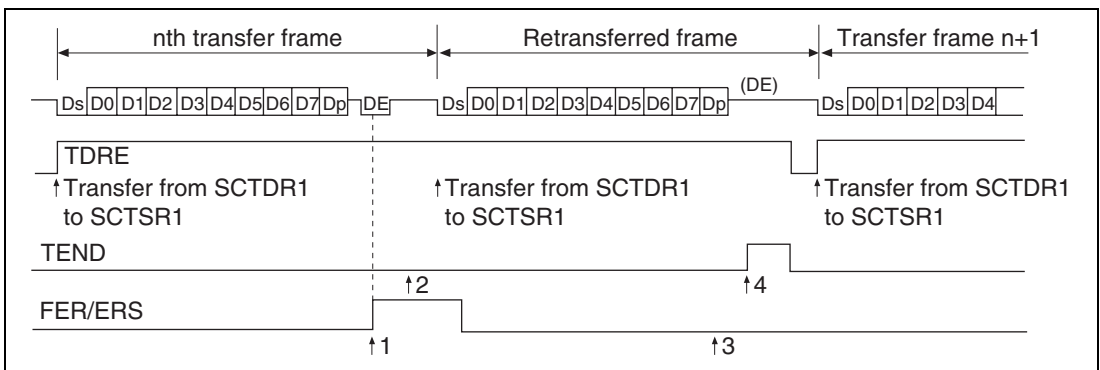


Figure 17.12 Retransfer Operation in SCI Transmit Mode

(3) Standby Mode and Clock

When switching between smart card interface mode and standby mode, the following procedures should be used to maintain the clock duty cycle.

Switching from Smart Card Interface Mode to Standby Mode:

1. Set the SBP1IO and SBP1DT bits in SCSPTR1 to the values for the fixed output state in standby mode.
2. Write 0 to the TE and RE bits in the serial control register (SCSCR1) to stop transmit/receive operations. At the same time, set the CKE1 bit to the value for the fixed output state in standby mode.
3. Write 0 to the CKE0 bit in SCSCR1 to stop the clock.
4. Wait for one serial clock cycle. During this period, the duty cycle is preserved and clock output is fixed at the specified level.
5. Write H'00 to the serial mode register (SCSMR1) and smart card mode register (SCSMR1).
6. Make the transition to the standby state.

Returning from Standby Mode to Smart Card Interface Mode:

7. Clear the standby state.
8. Set the CKE1 bit in SCSCR1 to the value for the fixed output state at the start of standby (the current SCK pin state).
9. Set smart card interface mode and output the clock. Clock signal generation is started with the normal duty cycle.

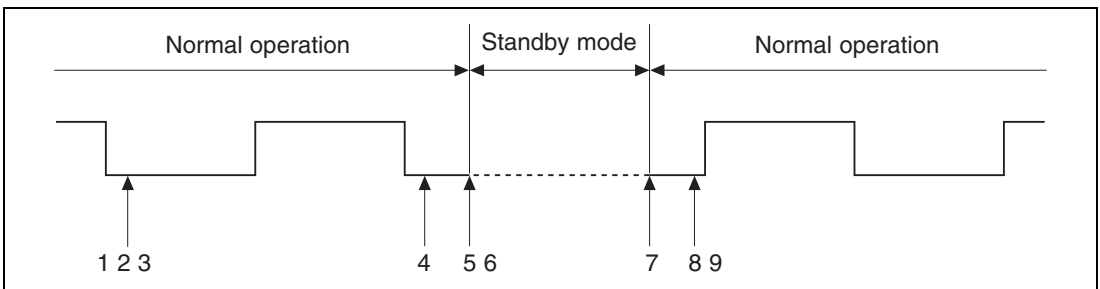


Figure 17.13 Procedure for Stopping and Restarting the Clock

(4) Power-On and Clock

The following procedure should be used to secure the clock duty cycle after powering on.

1. The initial state is port input and high impedance. Use pull-up or pull-down resistors to fix the potential.
2. Fix at the output specified by the CKE1 bit in the serial control register (SCSCR1).
3. Set the serial mode register (SCSMR1) and smart card mode register (SCSCMR1), and switch to smart card mode operation.
4. Set the CKE0 bit in SCSCR1 to 1 to start clock output.

Section 18 I/O Ports

18.1 Overview

This LSI has a 32-bit general-purpose I/O port, SCI I/O port, and SCIF I/O port.

18.1.1 Features

The features of the general-purpose I/O port are as follows:

- Available only in PCI-disabled mode.
- 32-bit I/O port with input/output direction independently specifiable for each bit.
- Pull-up can be specified independently for each bit.
- The 32 bits of the general-purpose I/O port are divided into 16-bit port A and 16-bit port B. Interrupts can be input to 16-bit port A.
- Use or non-use of the I/O port can be selected with the PORTEN bit in bus control register 2 (BCR2). (Do not set PORTEN = 1 when in PCI-enabled mode.)

The features of the SCI I/O port are as follows:

- Data can be output when the I/O port is designated for output and SCI enabling has not been set. This allows break function transmission.
- The RxD pin value can be read at all times, allowing break state detection.
- SCK pin control is possible when the I/O port is designated for output and SCI enabling has not been set.
- The SCK pin value can be read at all times.

The features of the SCIF I/O port are as follows:

- Data can be output when the I/O port is designated for output and SCIF enabling has not been set. This allows break function transmission.
- The RxD2 pin value can be read at all times, allowing break state detection.
- SCK2, $\overline{\text{CTS2}}$, and $\overline{\text{RTS2}}$ pin control is possible when the I/O port is designated for output and SCIF enabling has not been set.
- The SCK2, $\overline{\text{CTS2}}$, and $\overline{\text{RTS2}}$ pin values can be read at all times.

18.1.2 Block Diagrams

Figure 18.1 is a block diagram of the 16-bit general-purpose I/O port A with interrupt function.

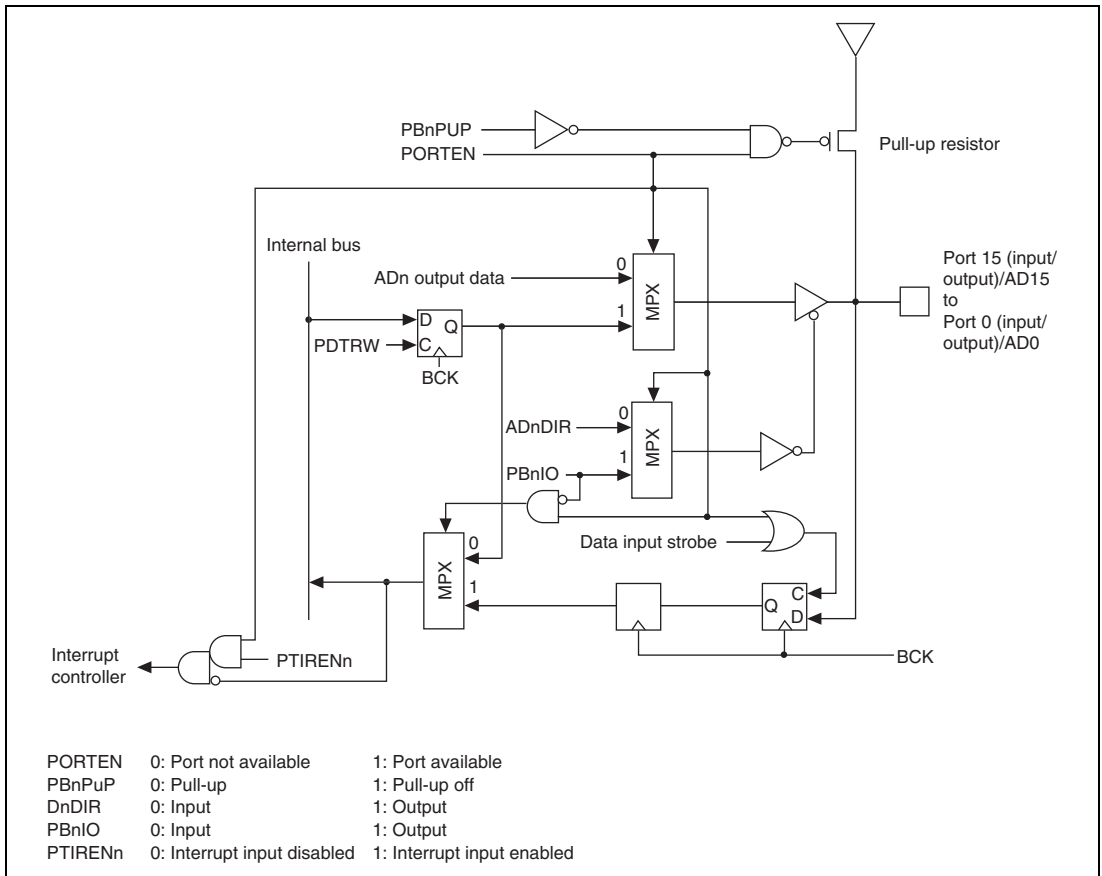


Figure 18.1 16-Bit Port A

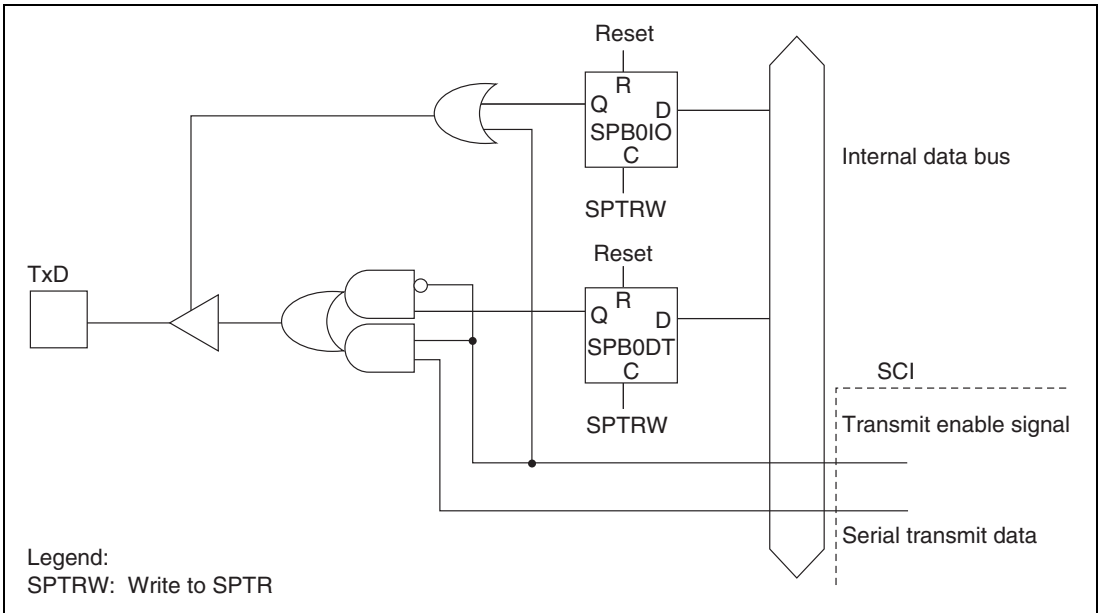


Figure 18.4 TxD Pin

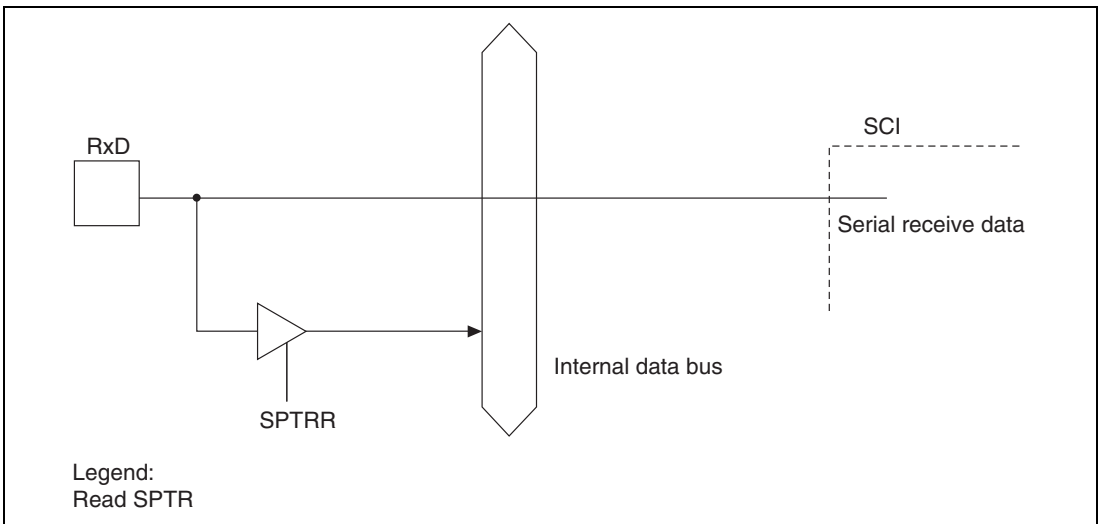


Figure 18.5 RxD Pin

SCIF I/O port block diagrams are shown in figures 18.6 to 18.10.

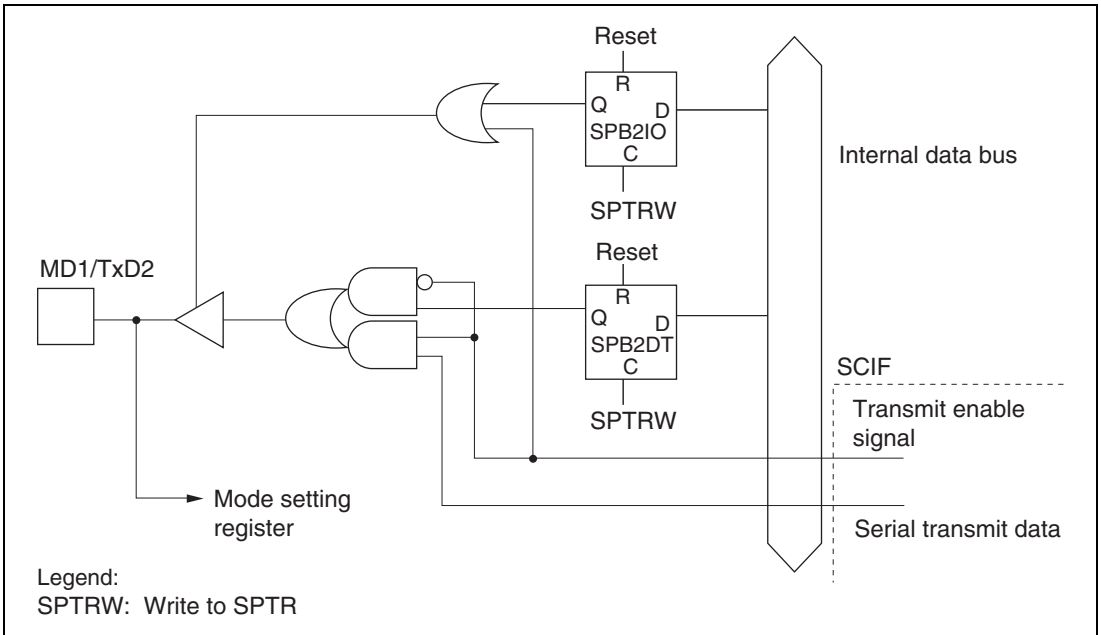


Figure 18.6 MD1/TxD2 Pin

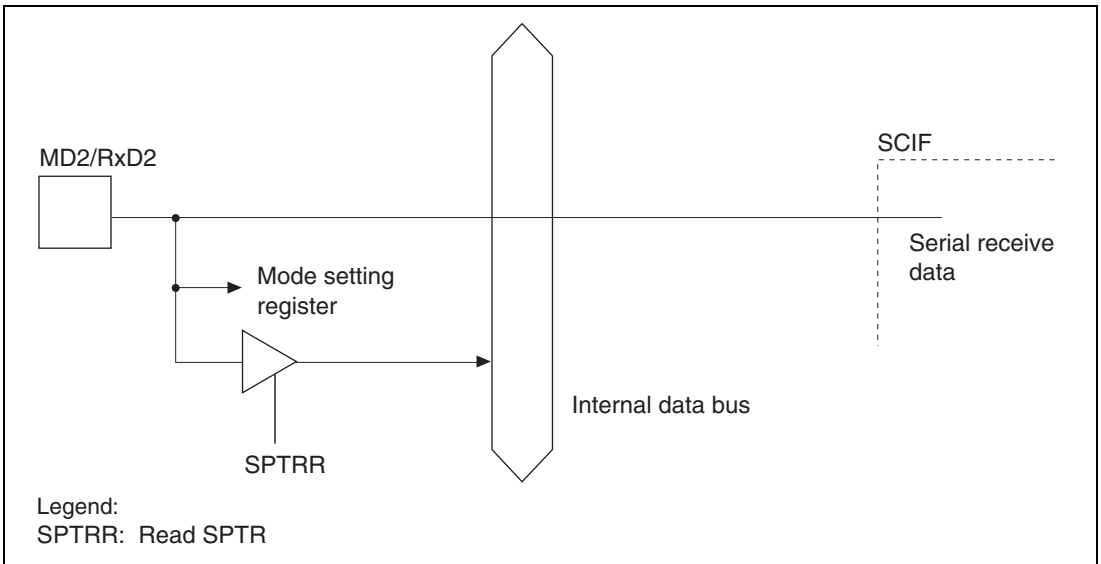


Figure 18.7 MD2/RxD2 Pin

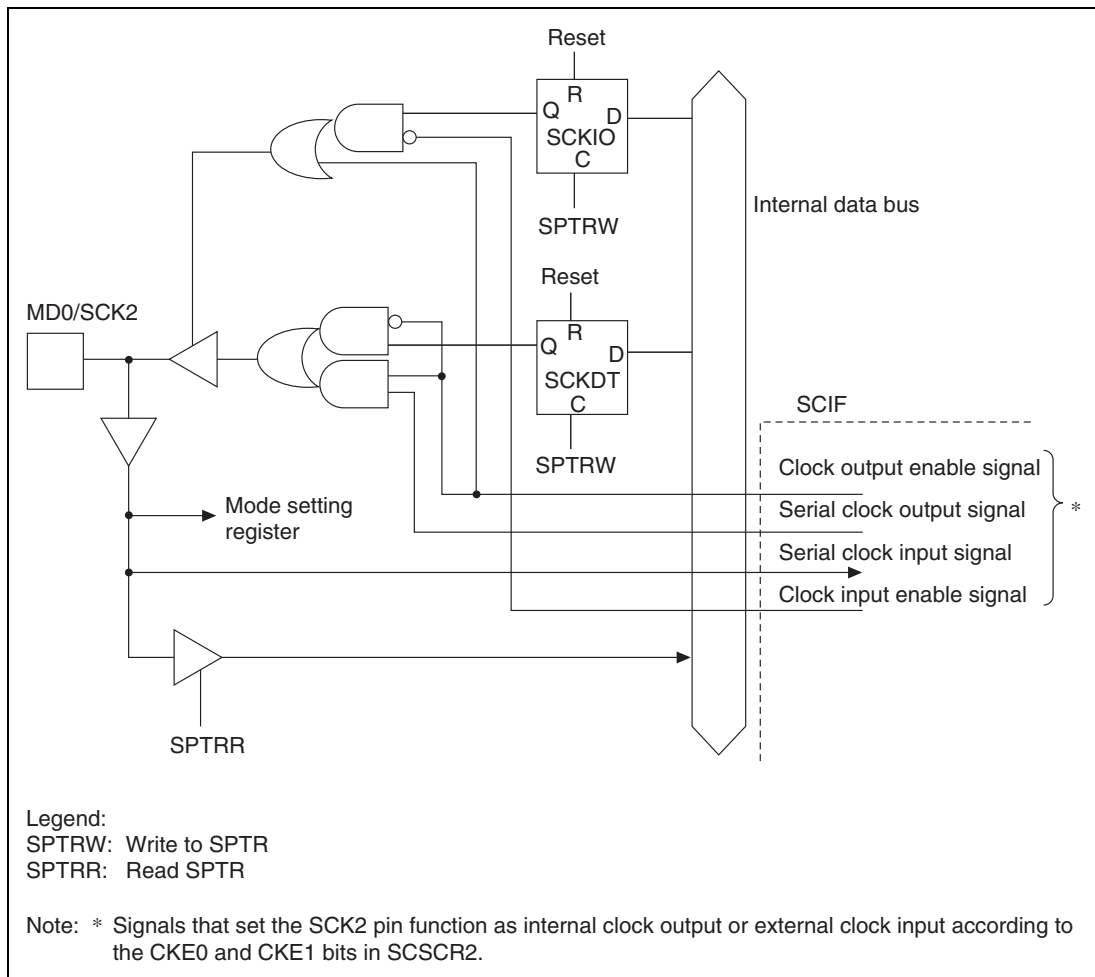


Figure 18.8 MD0/SCK2 Pin

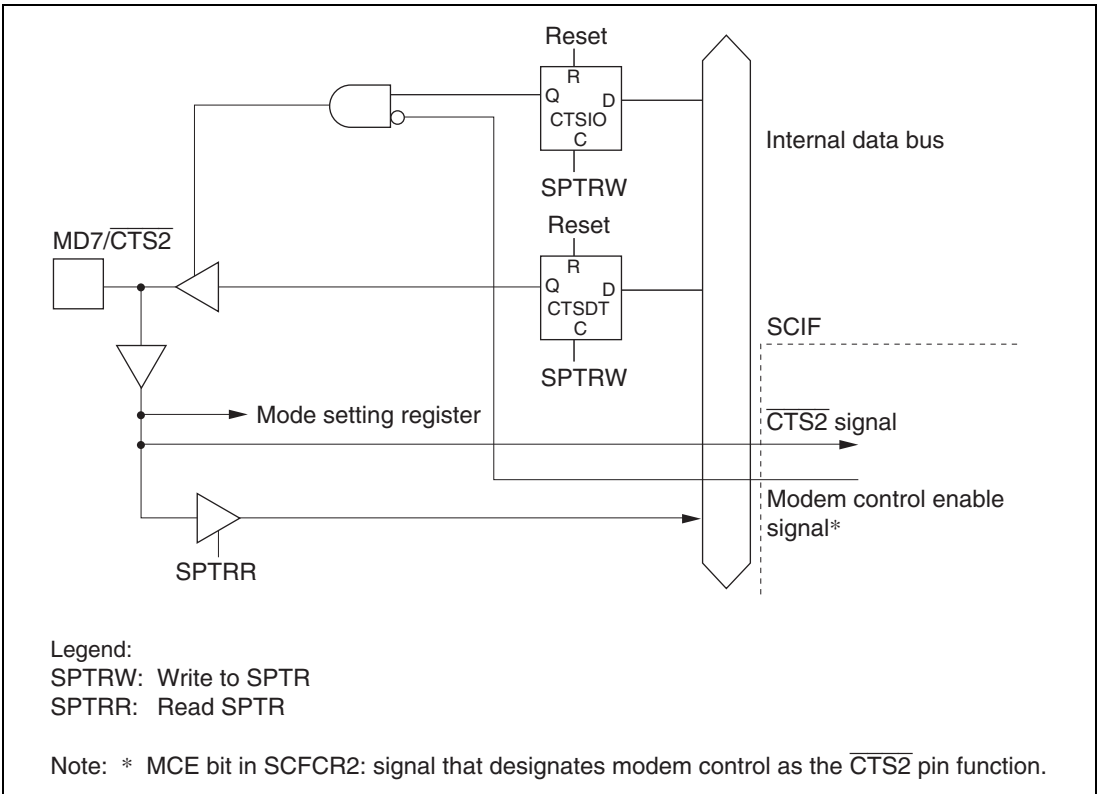
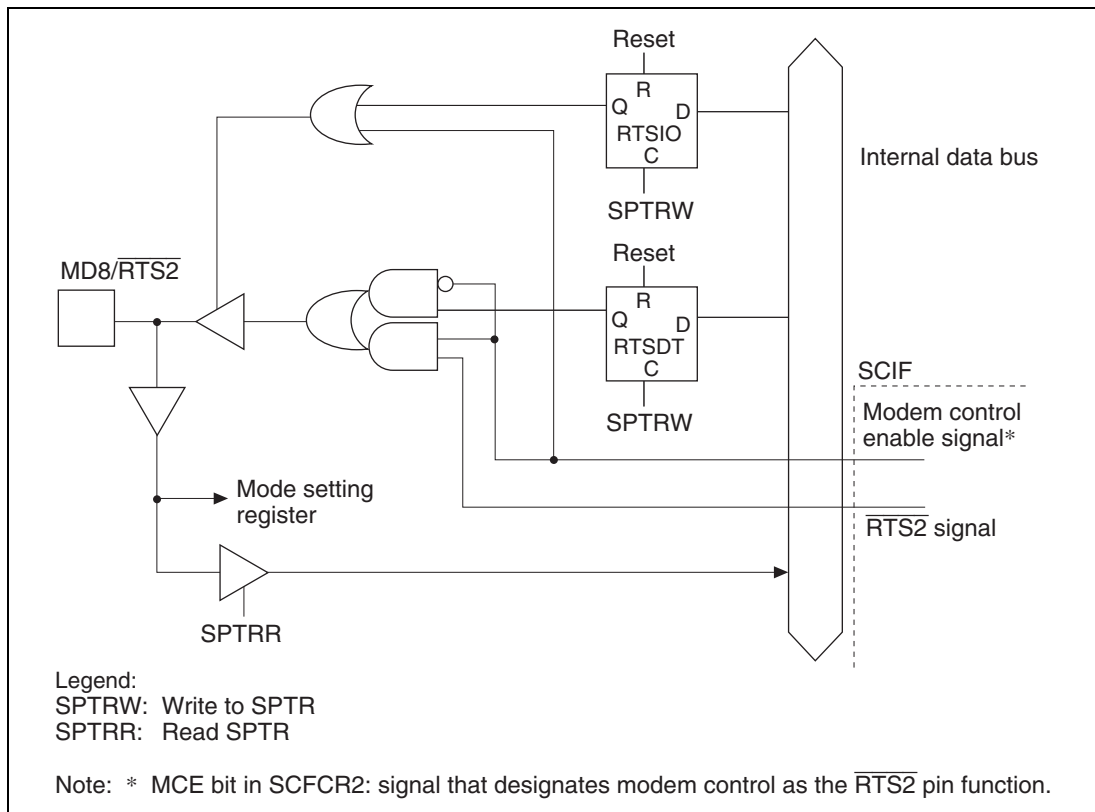


Figure 18.9 MD7/ $\overline{\text{CTS2}}$ Pin

Figure 18.10 MD8/ $\overline{\text{RTS2}}$ Pin

18.1.3 Pin Configuration

Table 18.1 shows the 32-bit general-purpose I/O port pin configuration.

Table 18.1 32-Bit General-Purpose I/O Port Pins

| Pin Name | Signal | I/O | Function |
|-------------|-------------|-----|----------|
| Port 31 pin | AD31/PORT31 | I/O | I/O port |
| Port 30 pin | AD30/PORT30 | I/O | I/O port |
| Port 29 pin | AD29/PORT29 | I/O | I/O port |
| Port 28 pin | AD28/PORT28 | I/O | I/O port |
| Port 27 pin | AD27/PORT27 | I/O | I/O port |
| Port 26 pin | AD26/PORT26 | I/O | I/O port |
| Port 25 pin | AD25/PORT25 | I/O | I/O port |

| Pin Name | Signal | I/O | Function |
|-------------|-------------|------|---------------------------|
| Port 24 pin | AD24/PORT24 | I/O | I/O port |
| Port 23 pin | AD23/PORT23 | I/O | I/O port |
| Port 22 pin | AD22/PORT22 | I/O | I/O port |
| Port 21 pin | AD21/PORT21 | I/O | I/O port |
| Port 20 pin | AD20/PORT20 | I/O | I/O port |
| Port 19 pin | AD19/PORT19 | I/O | I/O port |
| Port 18 pin | AD18/PORT18 | I/O | I/O port |
| Port 17 pin | AD17/PORT17 | I/O | I/O port |
| Port 16 pin | AD16/PORT16 | I/O | I/O port |
| Port 15 pin | AD15/PORT15 | I/O* | I/O port / GPIO interrupt |
| Port 14 pin | AD14/PORT14 | I/O* | I/O port / GPIO interrupt |
| Port 13 pin | AD13/PORT13 | I/O* | I/O port / GPIO interrupt |
| Port 12 pin | AD12/PORT12 | I/O* | I/O port / GPIO interrupt |
| Port 11 pin | AD11/PORT11 | I/O* | I/O port / GPIO interrupt |
| Port 10 pin | AD10/PORT10 | I/O* | I/O port / GPIO interrupt |
| Port 9 pin | AD9/PORT9 | I/O* | I/O port / GPIO interrupt |
| Port 8 pin | AD8/PORT8 | I/O* | I/O port / GPIO interrupt |
| Port 7 pin | AD7/PORT7 | I/O* | I/O port / GPIO interrupt |
| Port 6 pin | AD6/PORT6 | I/O* | I/O port / GPIO interrupt |
| Port 5 pin | AD5/PORT5 | I/O* | I/O port / GPIO interrupt |
| Port 4 pin | AD4/PORT4 | I/O* | I/O port / GPIO interrupt |
| Port 3 pin | AD3/PORT3 | I/O* | I/O port / GPIO interrupt |
| Port 2 pin | AD2/PORT2 | I/O* | I/O port / GPIO interrupt |
| Port 1 pin | AD1/PORT1 | I/O* | I/O port / GPIO interrupt |
| Port 0 pin | AD0/PORT0 | I/O* | I/O port / GPIO interrupt |

Note: * When port pins are used as GPIO interrupts, they must be set to input mode. The input setting can be made in the PCTRA register.

Table 18.2 shows the SCI I/O port pin configuration.

Table 18.2 SCI I/O Port Pins

| Pin Name | Abbreviation | I/O | Function |
|-------------------|--------------|--------|----------------------|
| Serial clock pin | SCK | I/O | Clock input/output |
| Receive data pin | RxD | Input | Receive data input |
| Transmit data pin | TxD | Output | Transmit data output |

Note: They are made to function as serial pins by performing SCI operation settings with the TE, RE, CKEI, and CKE0 bits in SCSCR1 and the C/ \bar{A} bit in SCSMR1. Break state transmission and detection can be performed by means of a setting in the SCI's SCSPTR1 register.

Table 18.3 shows the SCIF I/O port pin configuration.

Table 18.3 SCIF I/O Port Pins

| Pin Name | Abbreviation | I/O | Function |
|-------------------|-------------------------------|--------|----------------------|
| Serial clock pin | MD0/SCK2 | I/O | Clock input/output |
| Receive data pin | MD2/RxD2 | Input | Receive data input |
| Transmit data pin | MD1/TxD2 | Output | Transmit data output |
| Modem control pin | MD7/ $\overline{\text{CTS2}}$ | I/O | Transmission enabled |
| Modem control pin | MD8/ $\overline{\text{RTS2}}$ | I/O | Transmission request |

Note: These pins function as the MD0, MD1, MD2, MD7, and MD8 mode input pins after a power-on reset. These pins are made to function as serial pins by performing SCIF operation settings with the TE, RE, CKE1, and CKE0 bits in SCSCR2 and the MCE bit in SCFCR2. Break state transmission and detection can be set in the SCIF's SCSPTR2 register.

18.1.4 Register Configuration

The 32-bit general-purpose I/O port, SCI I/O port, and SCIF I/O port have seven registers, as shown in table 18.4.

Table 18.4 I/O Port Registers

| Name | Abbreviation | R/W | Initial Value* | P4 Address | Area 7 Address | Access Size |
|---------------------------------|--------------|-----|----------------|------------|----------------|-------------|
| Port control register A | PCTRA | R/W | H'00000000 | H'FF80002C | H'1F80002C | 32 |
| Port data register A | PDTRA | R/W | Undefined | H'FF800030 | H'1F800030 | 16 |
| Port control register B | PCTRB | R/W | H'00000000 | H'FF800040 | H'1F800040 | 32 |
| Port data register B | PDTRB | R/W | Undefined | H'FF800044 | H'1F800044 | 16 |
| GPIO interrupt control register | GPIOIC | R/W | H'00000000 | H'FF800048 | H'1F800048 | 16 |
| Serial port register | SCSPTR1 | R/W | Undefined | H'FFE0001C | H'1FE0001C | 8 |
| Serial port register | SCSPTR2 | R/W | Undefined | H'FFE80020 | H'1FE80020 | 16 |

Note: * Initialized by a power-on reset.

18.2 Register Descriptions

18.2.1 Port Control Register A (PCTRA)

Port control register A (PCTRA) is a 32-bit readable/writable register that controls the input/output direction and pull-up for each bit in the 16-bit port A (port 15 pin to port 0 pin). As the initial value of port data register A (PDTRA) is undefined, all the bits in the 16-bit port A should be set to output with PCTRA after writing a value to the PDTRA register.

PCTRA is initialized to H'00000000 by a power-on reset. It is not initialized by a manual reset or in standby mode, and retains its contents.

| | | | | | | | | |
|----------------|---------|--------|---------|--------|---------|--------|---------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | PB15PUP | PB15IO | PB14PUP | PB14IO | PB13PUP | PB13IO | PB12PUP | PB12IO |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | PB11PUP | PB11IO | PB10PUP | PB10IO | PB9PUP | PB9IO | PB8PUP | PB8IO |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PB7PUP | PB7IO | PB6PUP | PB6IO | PB5PUP | PB5IO | PB4PUP | PB4IO |
| 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 |
| | PB3PUP | PB3IO | PB2PUP | PB2IO | PB1PUP | PB1IO | PB0PUP | PB0IO |
| 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 2n + 1 (n = 0–15)—Port Pull-Up Control (PBnPUP): Specifies whether each bit in the 16-bit port A is to be pulled up with a built-in resistor. Pull-up is automatically turned off for a port pin set to output by bit PBnIO.

| Bit 2n + 1: PBnPUP | Description |
|--------------------|--|
| 0 | Bit m (m = 0–15) of 16-bit port A is pulled up (Initial value) |
| 1 | Bit m (m = 0–15) of 16-bit port A is not pulled up |

Bit 2n (n = 0–15)—Port I/O Control (PBnIO): Specifies whether each bit in the 16-bit port A is an input or an output.

| Bit 2n: PBnIO | Description |
|---------------|---|
| 0 | Bit m (m = 0–15) of 16-bit port A is an input (Initial value) |
| 1 | Bit m (m = 0–15) of 16-bit port A is an output |

18.2.2 Port Data Register A (PDTRA)

Port data register A (PDTRA) is a 16-bit readable/writable register used as a data latch for each bit in the 16-bit port A. When a bit is set as an output, the value written to the PDTRA register is output from the external pin. When a value is read from the PDTRA register while a bit is set as an input, the external pin value sampled on the external bus clock is read. When a bit is set as an output, the value written to the PDTRA register is read.

PDTRA is not initialized by a power-on or manual reset, or in standby mode, and retains its contents.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PB15DT | PB14DT | PB13DT | PB12DT | PB11DT | PB10DT | PB9DT | PB8DT |
| 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 |
| | PB7DT | PB6DT | PB5DT | PB4DT | PB3DT | PB2DT | PB1DT | PB0DT |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

18.2.3 Port Control Register B (PCTRB)

Port control register B (PCTRB) is a 32-bit readable/writable register that controls the input/output direction and pull-up for each bit in the 16-bit port B (port 31 pin to port 16 pin). As the initial value of port data register B (PDTRB) is undefined, each bit in the 16-bit port B should be set to output with PCTRB after writing a value to the PDTRB register.

PCTRB is initialized to H'00000000 by a power-on reset. It is not initialized by a manual reset or in standby mode, and retains its contents.

| | | | | | | | | |
|----------------|---------|--------|---------|--------|---------|--------|---------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | PB31PUP | PB31IO | PB30PUP | PB30IO | PB29PUP | PB29IO | PB28PUP | PB28IO |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | PB27PUP | PB27IO | PB26PUP | PB26IO | PB25PUP | PB25IO | PB24PUP | PB24IO |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PB23PUP | PB23IO | PB22PUP | PB22IO | PB21PUP | PB21IO | PB20PUP | PB20IO |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PB19PUP | PB19IO | PB18PUP | PB18IO | PB17PUP | PB17IO | PB16PUP | PB16IO |
| 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 2n + 1 (n = 0–15)—Port Pull-Up Control (PBnPUP): Specifies whether each bit in the 16-bit port B is to be pulled up with a built-in resistor. Pull-up is automatically turned off for a port pin set to output by bit PBnIO.

| Bit 2n + 1: PBnPUP | Description |
|--------------------|---|
| 0 | Bit m (m = 16–31) of 16-bit port B is pulled up (Initial value) |
| 1 | Bit m (m = 16–31) of 16-bit port B is not pulled up |

Bit 2n (n = 0–15)—Port I/O Control (PBnIO): Specifies whether each bit in the 16-bit port B is an input or an output.

| Bit 2n: PBnIO | Description |
|---------------|--|
| 0 | Bit m (m = 16–31) of 16-bit port B is an input (Initial value) |
| 1 | Bit m (m = 16–31) of 16-bit port B is an output |

18.2.4 Port Data Register B (PDTRB)

Port data register B (PDTRB) is a 16-bit readable/writable register used as a data latch for each bit in the 16-bit port B. When a bit is set as an output, the value written to the PDTRB register is output from the external pin. When a value is read from the PDTRB register while a bit is set as an input, the external pin value sampled on the external bus clock is read. When a bit is set as an output, the value written to the PDTRB register is read.

PDTRB is not initialized by a power-on or manual reset, or in standby mode, and retains its contents.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PB31DT | PB30DT | PB29DT | PB28DT | PB27DT | PB26DT | PB25DT | PB24DT |
| 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 |
| | PB23DT | PB22DT | PB21DT | PB20DT | PB19DT | PB18DT | PB17DT | PB16DT |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

18.2.5 GPIO Interrupt Control Register (GPIOIC)

The GPIO interrupt control register (GPIOIC) is a 16-bit readable/writable register that performs 16-bit interrupt input control.

GPIOIC is initialized to H'00000000 by a power-on reset. It is not initialized by a manual reset or in standby mode, and retains its contents.

GPIO interrupts are active-low level interrupts. Bit-by-bit masking is possible, and the OR of all the bits set as GPIO interrupts is used for interrupt detection. Which bits interrupts are input to can be identified by reading the PDTRA register.

| | | | | | | | | |
|----------------|----------|----------|----------|----------|----------|----------|---------|---------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PTIREN15 | PTIREN14 | PTIREN13 | PTIREN12 | PTIREN11 | PTIREN10 | PTIREN9 | PTIREN8 |
| 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 |
| | PTIREN7 | PTIREN6 | PTIREN5 | PTIREN4 | PTIREN3 | PTIREN2 | PTIREN1 | PTIREN0 |
| 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 n (n = 0–15)—Port Interrupt Enable (PTIRENn): Specifies whether interrupt input is performed for each bit.

| Bit n: PTIRENn | Description |
|----------------|--|
| 0 | Port m (m = 0–15) of 16-bit port A is used as a normal I/O port (Initial value) |
| 1 | Port m (m = 0–15) of 16-bit port A is used as a GPIO interrupt* |

Note: * When using an interrupt, set the corresponding port to input in the PCTRA register before making the PTIRENn setting.

18.2.6 Serial Port Register (SCSPTR1)

| | | | | | | | | |
|----------------|-----|---|---|---|--------|--------|--------|--------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | EIO | — | — | — | SPB1IO | SPB1DT | SPB0IO | SPB0DT |
| Initial value: | 0 | 0 | 0 | 0 | 0 | — | 0 | — |
| R/W: | R/W | — | — | — | R/W | R/W | R/W | R/W |

The serial port register (SCSPTR1) is an 8-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface (SCI) pins. Input data can be read from the RxD pin, output data written to the TxD pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. SCK pin data reading and output data writing can be performed by means of bits 3 and 2. Bit 7 controls enabling and disabling of the RXI interrupt.

SCSPTR1 can be read or written to by the CPU at all times. All SCSPTR1 bits except bits 2 and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 2 and 0 is undefined. SCSPTR1 is not initialized in the module standby state or standby mode.

Bit 7—Error Interrupt Only (EIO): See section 15.2.8, Serial Port Register (SCSPTR1).

Bits 6 to 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Serial Port Clock Port I/O (SPB1IO): Specifies serial port SCK pin input/output. When the SCK pin is actually set as a port output pin and outputs the value set by the SPB1DT bit, the $\overline{C/A}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1 should be cleared to 0.

| Bit 3: SPB1IO | Description |
|---------------|---|
| 0 | SPB1DT bit value is not output to the SCK pin (Initial value) |
| 1 | SPB1DT bit value is output to the SCK pin |

Bit 2—Serial Port Clock Port Data (SPB1DT): Specifies the serial port SCK pin input/output data. Input or output is specified by the SPB1IO bit (see the description of bit 3, SPB1IO, for details). When output is specified, the value of the SPB1DT bit is output to the SCK pin. The SCK pin value is read from the SPB1DT bit regardless of the value of the SPB1IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 2: SPB1DT | Description |
|---------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Bit 1—Serial Port Break I/O (SPB0IO): Specifies the serial port TxD pin output condition. When the TxD pin is actually set as a port output pin and outputs the value set by the SPB0DT bit, the TE bit in SCSCR1 should be cleared to 0.

| Bit 1: SPB0IO | Description |
|---------------|---|
| 0 | SPB0DT bit value is not output to the TxD pin (Initial value) |
| 1 | SPB0DT bit value is output to the TxD pin |

Bit 0—Serial Port Break Data (SPB0DT): Specifies the serial port RxD pin input data and TxD pin output data. The TxD pin output condition is specified by the SPB0IO bit (see the description of bit 1, SPB0IO, for details). When the TxD pin is designated as an output, the value of the SPB0DT bit is output to the TxD pin. The RxD pin value is read from the SPB0DT bit regardless of the value of the SPB0IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 0: SPB0DT | Description |
|---------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

18.2.7 Serial Port Register (SCSPTR2)

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RTSIO | RTSDT | CTSIO | CTSDT | SCKIO | SCKDT | SPB2IO | SPB2DT |
| Initial value: | 0 | — | 0 | — | 0 | — | 0 | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The serial port register (SCSPTR2) is a 16-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface with FIFO (SCIF) pins. Input data can be read from the RxD2 pin, output data written to the TxD2 pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. SCK2 pin data reading and output data writing can be performed by means of bits 3 and 2. $\overline{CTS2}$ pin data reading and output data writing can be performed by means of bits 5 and 4, and $\overline{RTS2}$ pin data reading and output data writing by means of bits 7 and 6.

SCSPTR2 can be read or written to by the CPU at all times. All SCSPTR2 bits except bits 6, 4, 2, and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 6, 4, 2, and 0 is undefined. SCSPTR2 is not initialized in standby mode or in the module standby state.

Bits 15 to 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Serial Port RTS Port I/O (RTSIO): Specifies serial port $\overline{RTS2}$ pin input/output. When the $\overline{RTS2}$ pin is actually set as a port output pin and outputs the value set by the RTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

| Bit 7: RTSIO | Description |
|--------------|--|
| 0 | RTSDT bit value is not output to the $\overline{RTS2}$ pin (Initial value) |
| 1 | RTSDT bit value is output to the $\overline{RTS2}$ pin |

Bit 6—Serial Port RTS Port Data (RTSDT): Specifies the serial port $\overline{\text{RTS2}}$ pin input/output data. Input or output is specified by the RTSIO pin (see the description of bit 7, RTSIO, for details). When the $\overline{\text{RTS2}}$ pin is designated as an output, the value of the RTSDT bit is output to the $\overline{\text{RTS2}}$ pin. The $\overline{\text{RTS2}}$ pin value is read from the RTSDT bit regardless of the value of the RTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 6: RTSDT | Description |
|--------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Bit 5—Serial Port CTS Port I/O (CTSIO): Specifies serial port $\overline{\text{CTS2}}$ pin input/output. When the $\overline{\text{CTS2}}$ pin is actually set as a port output pin and outputs the value set by the CTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

| Bit 5: CTSIO | Description |
|--------------|---|
| 0 | CTSDT bit value is not output to the $\overline{\text{CTS2}}$ pin (Initial value) |
| 1 | CTSDT bit value is output to the $\overline{\text{CTS2}}$ pin |

Bit 4—Serial Port CTS Port Data (CTSDT): Specifies the serial port $\overline{\text{CTS2}}$ pin input/output data. Input or output is specified by the CTSIO pin (see the description of bit 5, CTSIO, for details). When the $\overline{\text{CTS2}}$ pin is designated as an output, the value of the CTSDT bit is output to the $\overline{\text{CTS2}}$ pin. The $\overline{\text{CTS2}}$ pin value is read from the CTSDT bit regardless of the value of the CTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 4: CTSDT | Description |
|--------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Bit 3—Serial Port Clock Port I/O (SCKIO): Sets the I/O for the SCK2 pin serial port. To actually set the SCK2 pin as the port output pin and output the value set in the SCKDT bit, set the CKE1 and CKE0 bits of the SCSCR2 register to 0.

| Bit 3: SCKIO | Description |
|--------------|---|
| 0 | Shows that the value of the SCKDT bit is not output to the SCK2 pin (Initial value) |
| 1 | Shows that the value of the SCKDT bit is output to the SCK2 pin |

Bit 2—Serial Port Clock Port Data (SCKDT): Specifies the I/O data for the SCK2 pin serial port. The SCKIO bit specified input or output. (See bit 3: SCKIO, for details.) When set for output, the value of the SCKDT bit is output to the SCK2 pin. Regardless of the value of the SCKIO bit, the value of the SCK2 pin is fetched from the SCKDT bit. The initial value after a power-on reset or manual reset is undefined.

| Bit 2: SCKDT | Description |
|--------------|------------------------------|
| 0 | Shows I/O data level is LOW |
| 1 | Shows I/O data level is HIGH |

Bit 1—Serial Port Break I/O (SPB2IO): Specifies the serial port TxD2 pin output condition. When the TxD2 pin is actually set as a port output pin and outputs the value set by the SPB2DT bit, the TE bit in SCSCR2 should be cleared to 0.

| Bit 1: SPB2IO | Description |
|---------------|--|
| 0 | SPB2DT bit value is not output to the TxD2 pin (Initial value) |
| 1 | SPB2DT bit value is output to the TxD2 pin |

Bit 0—Serial Port Break Data (SPB2DT): Specifies the serial port RxD2 pin input data and TxD2 pin output data. The TxD2 pin output condition is specified by the SPB2IO bit (see the description of bit 1, SPB2IO, for details). When the TxD2 pin is designated as an output, the value of the SPB2DT bit is output to the TxD2 pin. The RxD2 pin value is read from the SPB2DT bit regardless of the value of the SPB2IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

| Bit 0: SPB2DT | Description |
|---------------|---------------------------------|
| 0 | Input/output data is low-level |
| 1 | Input/output data is high-level |

Section 19 Interrupt Controller (INTC)

19.1 Overview

The interrupt controller (INTC) ascertains the priority of interrupt sources and controls interrupt requests to the CPU. The INTC registers set the order of priority of each interrupt, allowing the user to handle interrupt requests according to user-set priority.

19.1.1 Features

The INTC has the following features.

- Fifteen interrupt priority levels can be set
By setting the five interrupt priority registers, the priorities of on-chip peripheral module interrupts can be selected from 15 levels for different request sources.
- NMI noise canceler function
The NMI input level bit indicates the NMI pin state. The pin state can be checked by reading this bit in the interrupt exception handler, enabling it to be used as a noise canceler.
- NMI request masking when SR.BL bit is set
It is possible to select whether or not NMI requests are to be masked when the SR.BL bit is set.

19.1.2 Block Diagram

Figure 19.1 shows a block diagram of the INTC.

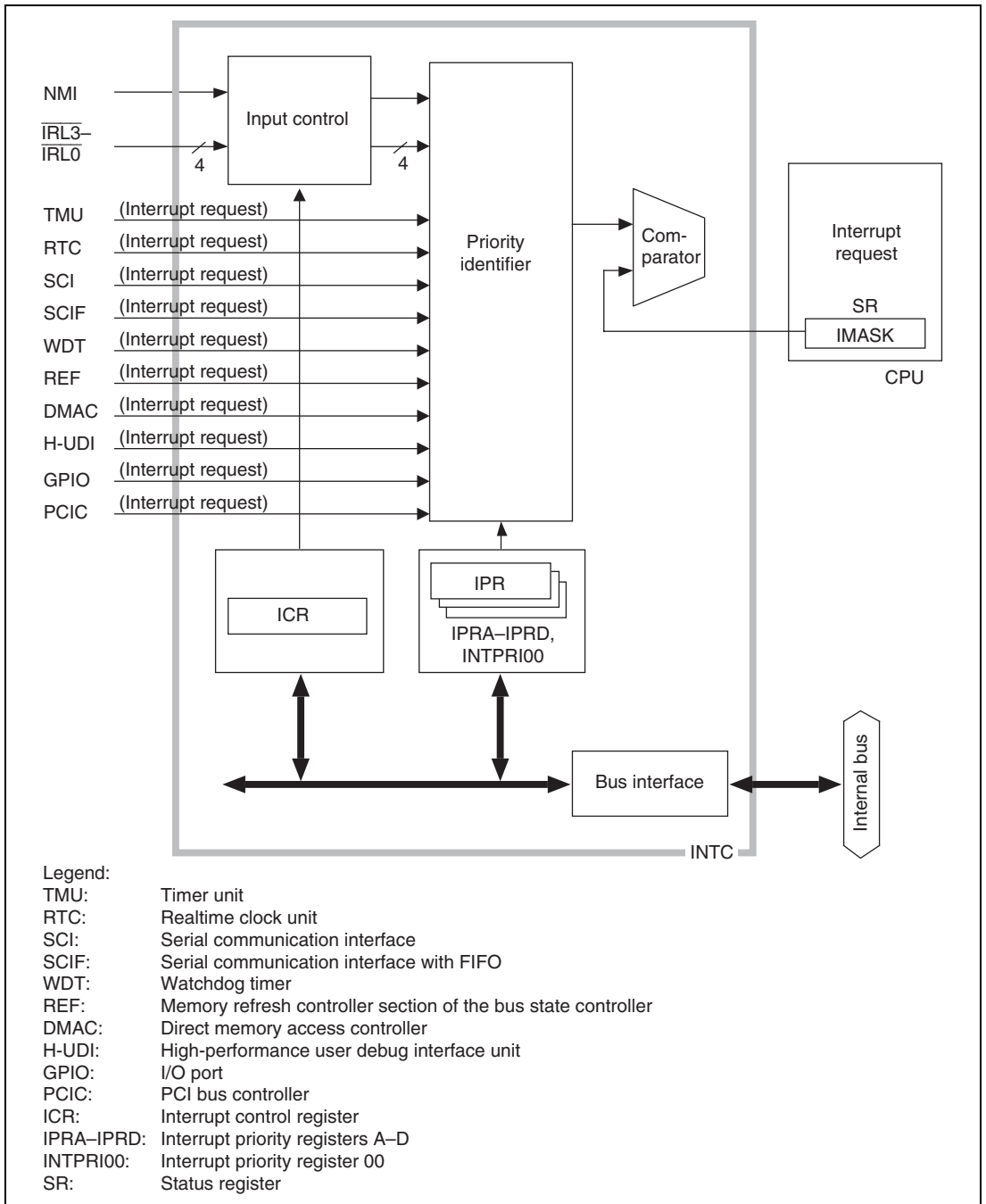


Figure 19.1 Block Diagram of INTC

19.1.3 Pin Configuration

Table 19.1 shows the INTC pin configuration.

Table 19.1 INTC Pins

| Pin Name | Abbreviation | I/O | Function |
|---------------------------------|---|-------|--|
| Nonmaskable interrupt input pin | NMI | Input | Input of nonmaskable interrupt request signal |
| Interrupt input pins | $\overline{\text{IRL3}}\text{--}\overline{\text{IRL0}}$ | Input | Input of interrupt request signals (maskable by IMASK in SR) |

19.1.4 Register Configuration

The INTC has the registers shown in table 19.2.

Table 19.2 INTC Registers

| Name | Abbreviation | R/W | Initial Value* ¹ | P4 Address | Area 7 Address | Access Size |
|----------------------------------|--------------|-----|-----------------------------|------------|----------------|-------------|
| Interrupt control register | ICR | R/W | * ² | H'FFD00000 | H'1FD00000 | 16 |
| Interrupt priority register A | IPRA | R/W | H'0000 | H'FFD00004 | H'1FD00004 | 16 |
| Interrupt priority register B | IPRB | R/W | H'0000 | H'FFD00008 | H'1FD00008 | 16 |
| Interrupt priority register C | IPRC | R/W | H'0000 | H'FFD0000C | H'1FD0000C | 16 |
| Interrupt priority register D | IPRD | R/W | H'DA74 | H'FFD00010 | H'1FD00010 | 16 |
| Interrupt priority register 00 | INTPRI00 | R/W | H'00000000 | H'FE080000 | H'1E080000 | 32 |
| Interrupt request register 00 | INTREQ00 | R | H'00000000 | H'FE080020 | H'1E080020 | 32 |
| Interrupt mask register 00 | INTMSK00 | R/W | H'000003FF | H'FE080040 | H'1E080040 | 32 |
| Interrupt mask clear register 00 | INTMSKCLR00 | W | — | H'FE080060 | H'1E080060 | 32 |

Notes: 1. Initialized by a power-on reset or manual reset.

2. H'8000 when the NMI pin is high, H'0000 when the NMI pin is low.

19.2 Interrupt Sources

There are three types of interrupt sources: NMI, IRL, and on-chip peripheral modules. Each interrupt has a priority level (16–0), with level 16 as the highest and level 1 as the lowest. When level 0 is set, the interrupt is masked and interrupt requests are ignored.

19.2.1 NMI Interrupt

The NMI interrupt has the highest priority level of 16. It is always accepted unless the BL bit in the status register in the CPU is set to 1. In sleep or standby mode, the interrupt is accepted even if the BL bit is set to 1.

A setting can also be made to have the NMI interrupt accepted even if the BL bit is set to 1.

Input from the NMI pin is edge-detected. The NMI edge select bit (NMIE) in the interrupt control register (ICR) is used to select either rising or falling edge. When the NMIE bit in the ICR register is modified, the NMI interrupt is not detected for a maximum of 6 bus clock cycles after the modification.

NMI interrupt exception handling does not affect the interrupt mask level bits (IMASK) in the status register (SR).

19.2.2 IRL Interrupts

IRL interrupts are input by level at pins $\overline{\text{IRL3}}\text{--}\overline{\text{IRL0}}$. The priority level is the level indicated by pins $\overline{\text{IRL3}}\text{--}\overline{\text{IRL0}}$. An $\overline{\text{IRL3}}\text{--}\overline{\text{IRL0}}$ value of 0 (0000) indicates the highest-level interrupt request (interrupt priority level 15). A value of 15 (1111) indicates no interrupt request (interrupt priority level 0).

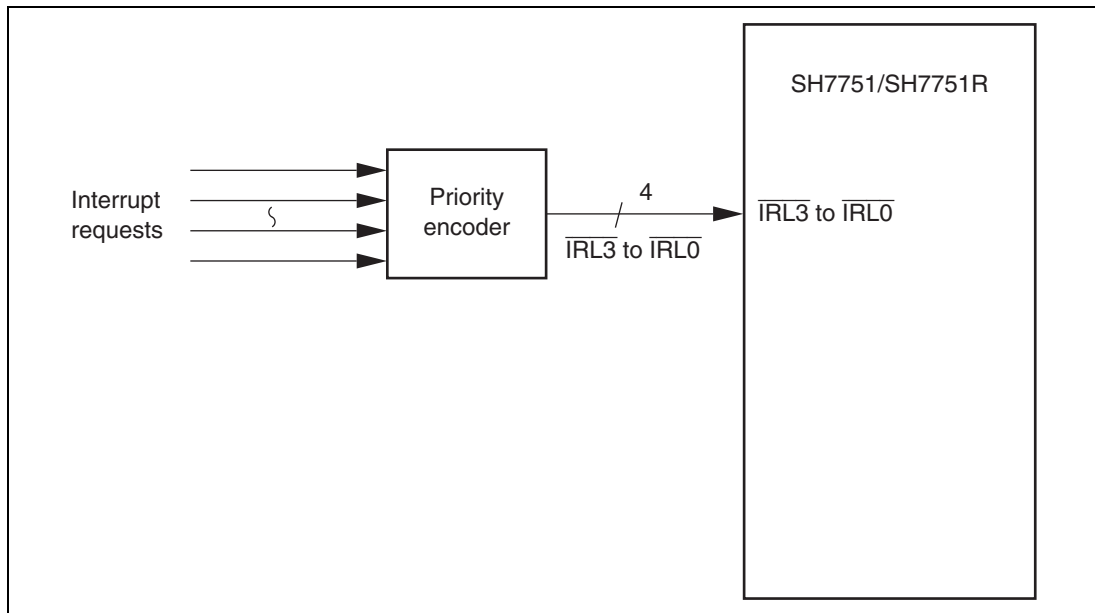


Figure 19.2 Example of IRL Interrupt Connection

Table 19.3 $\overline{\text{IRL3}}\text{--}\overline{\text{IRL0}}$ Pins and Interrupt Levels

| $\overline{\text{IRL3}}$ | $\overline{\text{IRL2}}$ | $\overline{\text{IRL1}}$ | $\overline{\text{IRL0}}$ | Interrupt Priority Level | Interrupt Request |
|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|----------------------------|
| 0 | 0 | 0 | 0 | 15 | Level 15 interrupt request |
| | | | 1 | 14 | Level 14 interrupt request |
| | | 1 | 0 | 13 | Level 13 interrupt request |
| | | | 1 | 12 | Level 12 interrupt request |
| | 1 | 0 | 0 | 11 | Level 11 interrupt request |
| | | | 1 | 10 | Level 10 interrupt request |
| | | 1 | 0 | 9 | Level 9 interrupt request |
| | | | 1 | 8 | Level 8 interrupt request |
| 1 | 0 | 0 | 0 | 7 | Level 7 interrupt request |
| | | | 1 | 6 | Level 6 interrupt request |
| | | 1 | 0 | 5 | Level 5 interrupt request |
| | | | 1 | 4 | Level 4 interrupt request |
| | 1 | 0 | 0 | 3 | Level 3 interrupt request |
| | | | 1 | 2 | Level 2 interrupt request |
| | | 1 | 0 | 1 | Level 1 interrupt request |
| | | | 1 | 0 | No interrupt request |

A noise-cancellation feature is built in, and the IRL interrupt is not detected unless the levels sampled at every bus clock cycle remain unchanged for three consecutive cycles, so that no transient level on the $\overline{\text{IRL}}$ pin change is detected. In standby mode, as the bus clock is stopped, noise cancellation is performed using the 32.768 kHz clock for the RTC instead. When the RTC is not used, therefore, interruption by means of IRL interrupts cannot be performed in standby mode.

The priority level of the IRL interrupt must not be lowered unless the interrupt is accepted and the interrupt handling starts. However, the priority level can be changed to a higher one.

The interrupt mask bits (IMASK) in the status register (SR) are not affected by IRL interrupt handling.

Pins $\overline{\text{IRL0}}\text{--}\overline{\text{IRL3}}$ can be used for four independent interrupt requests by setting the IRLM bit to 1 in the ICR register.

19.2.3 On-Chip Peripheral Module Interrupts

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

- High-performance user debug interface unit (H-UDI)
- Direct memory access controller (DMAC)
- Timer unit (TMU)
- Realtime clock (RTC)
- Serial communication interface (SCI)
- Serial communication interface with FIFO (SCIF)
- Bus state controller (BSC)
- Watchdog timer (WDT)
- I/O port (GPIO)
- PCI bus controller (PCIC)

Not every interrupt source is assigned a different interrupt vector, bus sources are reflected in the interrupt event register (INTEVT), so it is easy to identify sources by using the INTEVT register value as a branch offset in the exception handling routine.

A priority level from 15 to 0 can be set for each module by means of interrupt priority registers A to D (IPRA–IPRD) and interrupt priority register 00 (INTPRI00).

The interrupt mask bits (IMASK) in the status register (SR) are not affected by on-chip peripheral module interrupt handling.

On-chip peripheral module interrupt source flag and interrupt enable flag updating should only be carried out when the BL bit in the status register (SR) is set to 1. To prevent acceptance of an erroneous interrupt from an interrupt source that should have been updated, first read the on-chip peripheral register containing the relevant flag, then clear the BL bit to 0. Furthermore, in case of an interrupt of TMU channels 3 and 4 and PCIC, read the interrupt factor register 00 (INTREQ00). This will secure the necessary timing internally. When updating a number of flags, there is no problem if only the register containing the last flag updated is read.

If flag updating is performed while the BL bit is cleared to 0, the program may jump to the interrupt handling routine when the INTEVT register value is 0. In this case, interrupt handling is initiated due to the timing relationship between the flag update and interrupt request recognition within the chip. Processing can be continued without any problem by executing an RTE instruction.

19.2.4 Interrupt Exception Handling and Priority

Table 19.4 lists the codes for the interrupt event register (INTEVT), and the order of interrupt priority. Each interrupt source is assigned a unique INTEVT code. The start address of the interrupt handler is common to each interrupt source. This is why, for instance, the value of INTEVT is used as an offset at the start of the interrupt handler and branched to in order to identify the interrupt source.

The order of priority of the on-chip peripheral modules is specified as desired by setting priority levels from 0 to 15 in interrupt priority registers A to D (IPRA–IPRD) and interrupt priority register 00 (INTPRI00). The order of priority of the on-chip peripheral modules is set to 0 by a reset.

When the priorities for multiple interrupt sources are set to the same level and such interrupts are generated simultaneously, they are handled according to the default priority order shown in table 19.4.

Updating of interrupt priority registers A to D, and INTPRI00 should only be carried out when the BL bit in the status register (SR) is set to 1. To prevent erroneous interrupt acceptance, first read one of the interrupt priority registers, then clear the BL bit to 0. This will secure the necessary timing internally.

TICPI2: Input capture interrupt
ATI: Alarm interrupt
PRI: Periodic interrupt
CUI: Carry-up interrupt
ERI: Receive-error interrupt
RXI: Receive-data-full interrupt
TXI: Transmit-data-empty interrupt
TEI: Transmit-end interrupt
BRI: Break interrupt request
ITI: Interval timer interrupt
RCMI: Compare-match interrupt
ROVI: Refresh counter overflow interrupt
H-UDI: H-UDI interrupt
GPIOI: I/O port interrupt
DMTE0–DMTE7: DMAC transfer end interrupts
DMAE: DMAC address error interrupt
PCISERR: PCIC SERR error interrupt
PCIERR: PCIC error interrupt
PCIPWDWN: PCIC power-down request interrupt
PCIPWON: PCIC power-ON request interrupt
PCIDMA0 to 3: PCIC DMA transfer end interrupts
Note: * SH7751R only

19.3 Register Descriptions

19.3.1 Interrupt Priority Registers A to D (IPRA–IPRD)

Interrupt priority registers A to D (IPRA–IPRD) are 16-bit readable/writable registers that set priority levels from 0 to 15 for on-chip peripheral module interrupts. IPRA to IPRC are initialized to H'0000 and IPRD is to H'DA74 by a reset. They are not initialized in standby mode.

IPRA to IPRC

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

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

IPRD

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

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

Table 19.5 shows the relationship between the interrupt request sources and the IPRA–IPRD register bits.

Table 19.5 Interrupt Request Sources and IPRA–IPRD Registers

| Register | Bits | | | |
|-------------------------------|-------|-------------------|------|------------------------|
| | 15–12 | 11–8 | 7–4 | 3–0 |
| Interrupt priority register A | TMU0 | TMU1 | TMU2 | RTC |
| Interrupt priority register B | WDT | REF* ¹ | SCI1 | Reserved* ² |
| Interrupt priority register C | GPIO | DMAC | SCIF | H-UDI |
| Interrupt priority register D | IRL0 | IRL1 | IRL2 | IRL3 |

Notes: 1. REF is the memory refresh unit in the bus state controller (BSC). See section 13, Bus State Controller (BSC), for details.

2. Reserved bits: These bits are always read as 0 and should always be written with 0.

As shown in table 19.5, four on-chip peripheral modules are assigned to each register. Interrupt priority levels are established by setting a value from H'F (1111) to H'0 (0000) in each of the four-bit groups: 15–12, 11–8, 7–4, and 3–0. Setting H'F designates priority level 15 (the highest level), and setting H'0 designates priority level 0 (requests are masked).

19.3.2 Interrupt Control Register (ICR)

The interrupt control register (ICR) is a 16-bit register that sets the input signal detection mode for external interrupt input pin NMI and indicates the input signal level at the NMI pin. This register is initialized by a power-on reset or manual reset. It is not initialized in standby mode.

| | | | | | | | | |
|----------------|------|-----|----|----|----|----|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Bit name: | NMIL | MAI | — | — | — | — | NMIB | NMIE |
| Initial value: | 0/1* | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | — | — | — | — | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Bit name: | IRLM | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | — | — | — | — | — | — | — |

Note: * 1 when NMI pin input is high, 0 when low.

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

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

Bit 14—NMI Interrupt Mask (MAI): Specifies whether or not all interrupts are to be masked while the NMI pin input level is low, irrespective of the CPU's SR.BL bit.

| Bit 14: MAI | Description |
|-------------|--|
| 0 | Interrupts enabled even while NMI pin is low (Initial value) |
| 1 | Interrupts disabled while NMI pin is low* |

Note: * NMI interrupts are accepted in normal operation and in sleep mode.

In standby mode, all interrupts are masked, and standby is not cleared, while the NMI pin is low.

Bit 9—NMI Block Mode (NMIB): Specifies whether an NMI request is to be held pending or detected immediately while the SR.BL bit is set to 1.

| Bit 9: NMIB | Description |
|-------------|---|
| 0 | NMI interrupt requests held pending while SR.BL bit is set to 1 (Initial value) |
| 1 | NMI interrupt requests detected while SR.BL bit is set to 1 |

Notes: 1. If interrupt requests are enabled while SR.BL = 1, the previous exception information will be lost, and so must be saved beforehand.

2. This bit is cleared automatically by NMI acceptance.

Bit 8—NMI Edge Select (NMIE): Specifies whether the falling or rising edge of the interrupt request signal to the NMI pin is detected.

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

Bit 7—IRL Pin Mode (IRLM): Specifies whether pins $\overline{\text{IRL3}}$ – $\overline{\text{IRL0}}$ are to be used as level-encoded interrupt requests or as four independent interrupt requests.

| Bit 7: IRLM | Description |
|-------------|---|
| 0 | $\overline{\text{IRL}}$ pins used as level-encoded interrupt requests (Initial value) |
| 1 | $\overline{\text{IRL}}$ pins used as four independent interrupt requests (level-sense IRQ mode) |

Bits 13 to 10 and 6 to 0—Reserved: These bits are always read as 0, and should only be written with 0.

19.3.3 Interrupt Priority Level Setting Register 00 (INTPRI00)

The interrupt priority level setting register (INTPRI00) sets the order of priority (levels 15 to 0) of the internal peripheral module interrupts. The INTPRI00 register is a 32-bit read/write register. It is initialized to H'00000000 at a reset. It is not initialized in standby mode.

| | | | | | | | | |
|----------------|--------------------------|--------------------------|--------------------------|-----|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit: | 31 | 30 | 29 | ... | 19 | 18 | 17 | 16 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | ... | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 15 | 14 | 13 | ... | 3 | 2 | 1 | 0 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | ... | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 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 |

Table 19.6 shows the relationship between interrupt request sources and the respective bits of the INTPRI00 register.

Table 19.6 Interrupt Request Sources and INTPRI00 Register

| Register | Bits | | | | | | | |
|---|----------|----------|----------|----------|----------|---------|---------|---------|
| | 31 to 28 | 27 to 24 | 23 to 20 | 19 to 16 | 15 to 12 | 11 to 8 | 7 to 4 | 3 to 0 |
| Interrupt priority level setting register | Reserved | Reserved | Reserved | Reserved | TMU ch4 | TMU ch3 | PCI (1) | PCI (0) |

Note: Reserved bits: These bits always read as 0, and should only be written with 0.

As shown in table 19.6, 8 combinations of internal peripheral modules are assigned to one register. Values of H'F (1111) to H'0 (0000) can be set in each 4 bits, allowing the order levels of the corresponding interrupts to be set. H'F is priority level 15 (highest level) while H'0 is priority level 0 (request mask).

Reserved: These bits are always read as 0, and should only be written with 0.

19.3.4 Interrupt Factor Register 00 (INTREQ00)

The interrupt factor register 00 (INTREQ00) shows which interrupt have been requested of the INTC. Even when the interrupts are masked with INTPRI00 and INTMSK00, the bits in this register are not affected. INTREQ00 is a 32-bit read-only register.

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|---|---|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | | | | ... | | | | |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | ... | R | R | R | R |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

Bits 31 to 0—Interrupt Request: These bits indicate the existence of an interrupt request corresponding to each bit. For the correspondence between bits and interrupt sources, see section 19.3.7, INTREQ00, INTMSK00, and INTMSKCLR00 Bit Allocation.

| Bits 31 to 0 | Description |
|--------------|---|
| 0 | Shows no corresponding interrupt request (Initial value) |
| 1 | Shows existence of corresponding interrupt request |

19.3.5 Interrupt Mask Register 00 (INTMSK00)

The interrupt mask register 00 (INTMSK00) specifies whether or not to mask individual interrupts each time they are requested. The INTMSK00 register is a 32-bit register. It is initialized to H'000003FF at a reset. The values are retained in standby mode.

To clear each interrupt mask, write 1 to the corresponding bit of the INTMSKCLR00 register. The values in INTMSK00 do not change if you write 0 to it.

| | | | | | | | | |
|----------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | ... | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 1 | 1 |
| R/W: | R | R | R | ... | R | R | R/W | R/W |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 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 31 to 0—Interrupt Masks: These bits indicate the existence of an interrupt request corresponding to each bit. For the correspondence between bits and interrupt sources, see section 19.3.7, INTREQ00, INTMSK00, and INTMSKCLR00 Bit Allocation.

| Bits 31 to 0 | Description |
|--------------|--|
| 0 | Accept corresponding interrupt request |
| 1 | Mask corresponding interrupt request |

19.3.6 Interrupt Mask Clear Register 00 (INTMSKCLR00)

The interrupt mask clear register 00 (INTMSKCLR00) clears the masks for each request of the corresponding interrupt. INTMSKCLR00 is a 32-bit write-only register.

| | | | | | | | | |
|----------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | ... | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | — | — | — | ... | — | — | — | — |
| R/W: | W | W | W | ... | W | W | W | W |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | W | W | W | W | W | W | W | W |

Bits 31 to 0—Interrupt Mask Clear: These bits indicate the existence of an interrupt request corresponding to each bit. For the correspondence between bits and interrupt sources, see section 19.3.7, INTREQ00, INTMSK00, and INTMSKCLR00 Bit Allocation.

| Bits 31 to 0 | Description |
|--------------|--|
| 0 | Do not change corresponding interrupt mask |
| 1 | Clear corresponding interrupt mask |

19.3.7 INTREQ00, INTMSK00, and INTMSKCLR00 Bit Allocation

The following shows the relationship between individual bits in the register and interrupt factors.

Table 19.7 Bit Allocation

| Bit No. | Module | Interrupt |
|----------|----------|-----------|
| 31 to 10 | Reserved | Reserved |
| 9 | TMU | TUNI4 |
| 8 | TMU | TUNI3 |
| 7 | PCI | PCIERR |
| 6 | PCI | PCIPWDWN |
| 5 | PCI | PCIPWON |
| 4 | PCI | PCIDMA0 |
| 3 | PCI | PCIDMA1 |
| 2 | PCI | PCIDMA2 |
| 1 | PCI | PCIDMA3 |
| 0 | PCI | PCISERR |

19.4 INTC Operation

19.4.1 Interrupt Operation Sequence

The sequence of operations when an interrupt is generated is described below. Figure 19.3 shows a flowchart of the operations.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest-priority interrupt from the interrupt requests sent, according to the priority levels set in interrupt priority registers A to D (IPRA–IPRD) and interrupt priority register 00 (INTPRI00). Lower-priority interrupts are held pending. If two of these interrupts have the same priority level, or if multiple interrupts occur within a single module, the interrupt with the highest priority according to table 19.4, Interrupt Exception Handling Sources and Priority Order, is selected.
3. The priority level of the interrupt selected by the interrupt controller is compared with the interrupt mask bits (IMASK) in the status register (SR) of the CPU. If the request priority level is higher than the level in bits IMASK, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. The CPU accepts an interrupt at a break between instructions.
5. The interrupt source code is set in the interrupt event register (INTEVT).
6. The status register (SR) and program counter (PC) are saved to SSR and SPC, respectively.
7. The block bit (BL), mode bit (MD), and register bank bit (RB) in SR are set to 1.
8. The CPU jumps to the start address of the interrupt handler (the sum of the value set in the vector base register (VBR) and H'00000600).

The interrupt handler may branch with the INTEVT register value as its offset in order to identify the interrupt source. This enables it to branch to the handling routine for the particular interrupt source.

- Notes:
1. The interrupt mask bits (IMASK) in the status register (SR) are not changed by acceptance of an interrupt in this LSI.
 2. The interrupt source flag should be cleared in the exception handling routine. To ensure that an interrupt request that should have been cleared is not inadvertently accepted again, read the interrupt source flag after it has been cleared, then wait for the interval shown in table 19.8 (Time for priority decision and SR mask bit comparison) before clearing the BL bit or executing an RTE instruction.
 3. Depending on the interrupt factor, the interrupt mask (INTMSK00) must be cleared for each factor using the INTMSKCLR00 register. See section 19.3.5, Interrupt Mask

Register 00 (INTMSK00), and section 19.3.6, Interrupt Mask Clear Register 00 (INTMSKCLR00), for details.

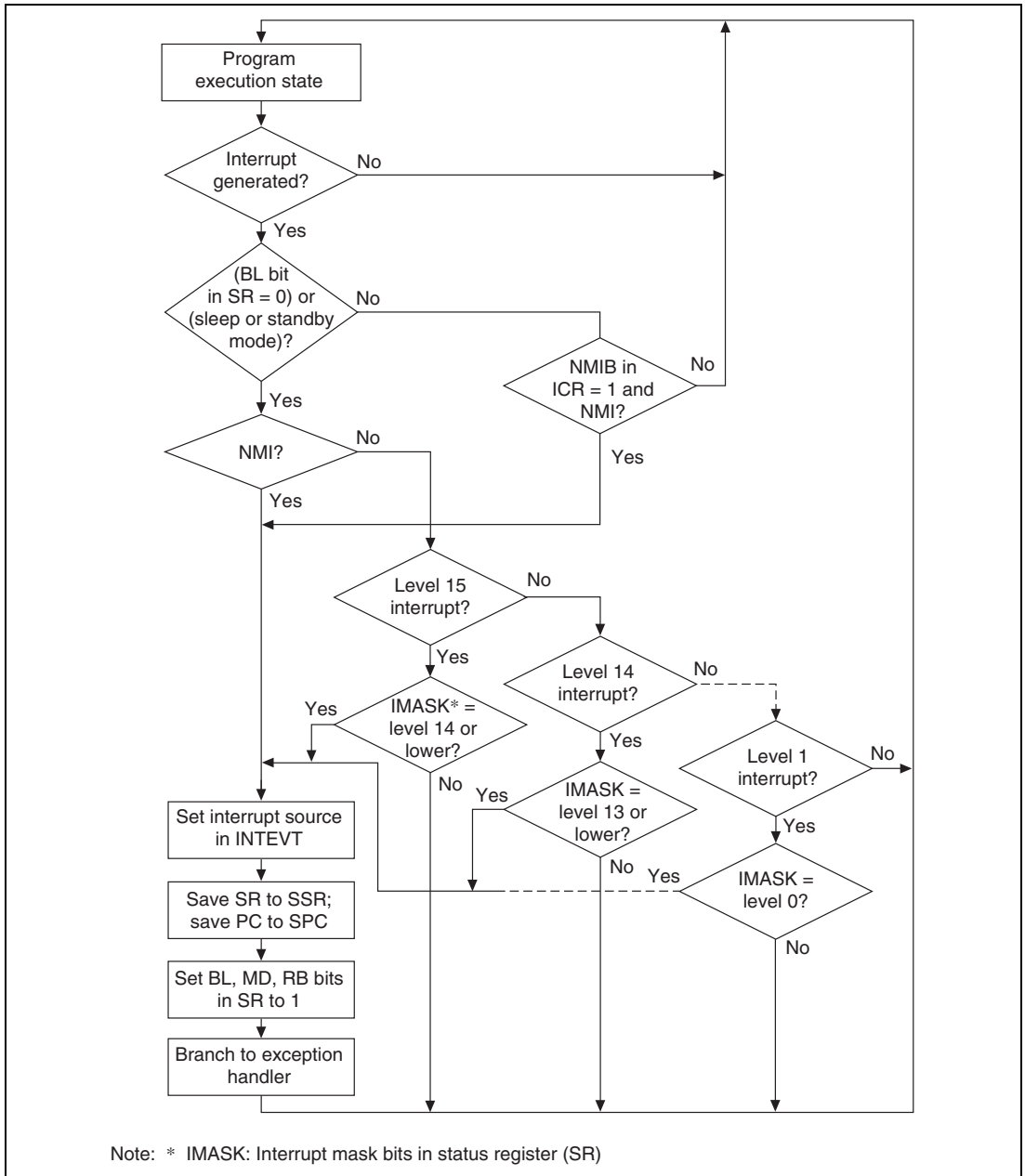


Figure 19.3 Interrupt Operation Flowchart

19.4.2 Multiple Interrupts

When handling multiple interrupts, interrupt handling should include the following procedures:

1. Branch to a specific interrupt handler corresponding to a code set in the INTEVT register. The code in INTEVT can be used as a branch-offset for branching to the specific handler.
2. Clear the interrupt source in the corresponding interrupt handler.
3. Save SPC and SSR to the stack.
4. Clear the BL bit in SR, and set the accepted interrupt level in the interrupt mask bits in SR.
5. Handle the interrupt.
6. Set the BL bit in SR to 1.
7. Restore SSR and SPC from memory.
8. Execute the RTE instruction.

When these procedures are followed in order, an interrupt of higher priority than the one being handled can be accepted after clearing BL in step 4. This enables the interrupt response time to be shortened for urgent processing.

19.4.3 Interrupt Masking with MAI Bit

By setting the MAI bit to 1 in the ICR register, it is possible to mask interrupts while the NMI pin is low, irrespective of the BL and IMASK bits in the SR register.

- In normal operation and sleep mode
All interrupts are masked while the NMI pin is low. However, an NMI interrupt only is generated by a transition at the NMI pin.
- In standby mode
All interrupts are masked while the NMI pin is low, and an NMI interrupt is not generated by a transition at the NMI pin. Therefore, standby cannot be cleared by an NMI interrupt while the MAI bit is set to 1.

19.5 Interrupt Response Time

The time from generation of an interrupt request until interrupt exception handling is performed and fetching of the first instruction of the exception handler is started (the interrupt response time) is shown in table 19.8.

Table 19.8 Interrupt Response Time

| Item | Number of States | | | Notes | |
|---|---------------------------------|---------------------------------|---------------------------------|--------------------------------|--------------------------|
| | NMI | RL | Peripheral Modules | | |
| Time for priority decision and SR mask bit comparison* | 1lcyc + 4Bcyc | 1lcyc + 7Bcyc | 1lcyc + 2Bcyc | | |
| Wait time until end of sequence being executed by CPU | $S - 1 (\geq 0) \times$ lcyc | $S - 1 (\geq 0) \times$ lcyc | $S - 1 (\geq 0) \times$ lcyc | | |
| Time from interrupt exception handling (save of SR and PC) until fetch of first instruction of exception handler is started | $4 \times$ lcyc | $4 \times$ lcyc | $4 \times$ lcyc | | |
| Response time | Total | 5lcyc + 4Bcyc + (S - 1)lcyc | 5lcyc + 7Bcyc + (S - 1)lcyc | 5lcyc + 2Bcyc + (S - 1)lcyc | |
| | Minimum case | 13lcyc | 19lcyc | 9lcyc | When lcyc: Bcyc = 2:1 |
| | Maximum case | 36 + S lcyc | 60 + S lcyc | 20 + S lcyc | When lcyc: Bcyc = 8:1 |

Legend:

lcyc: One cycle of internal clock supplied to CPU, etc.

Bcyc: One CKIO cycle

S: Latency of instruction

Note: * In the SH7751, this includes the case where the mask bit (IMASK) in SR is changed and a new interrupt is generated.

19.6 Usage Notes

19.6.1 NMI Interrupts (SH7751 Only)

When multiple NMI interrupts are input to the NMI pin within a set period of time (which is dependent on the internal state of the CPU and the external bus state), subsequent interrupts may not be accepted.

Note that this problem does not occur when sufficient time^{*1} is provided between NMI interrupt inputs or with non-NMI interrupts such as IRL interrupts.

Workarounds: Any of the following methods may be used to avoid the above problem.

(1) Allow sufficient time between NMI interrupt inputs, as described in note 1, below.

Note that it may not be possible to assure the above interval between NMI interrupt inputs if hazard is input to NMI, and that this may cause the device to malfunction. Design the external circuits so that no hazard is input via NMI.^{*2}

(2) Do not use NMI interrupts. Use IRL interrupts instead.

(3) Workaround using software

The above problem can be avoided by inserting the following lines of code^{*3*4} into the NMI exception handling routine.

- Notes:
1. If SR.BL is cleared to 0 so that one or more instructions may be executed between the handling of two NMI interrupts.
 2. When changing the level of the NMI input, ensure that the high and low durations are at least 5 CKIO cycles. Also ensure that no noise pulses occur before or after level changes.
 3. If the NMI exception handling routine contains code that changes the value of the SR.BL bit, the code listed below should be inserted before the point at which the change is made.
 4. Registers R0 to R3 in the code sample can be changed to any general register. Also, the necessary register save and restore instructions should be inserted before and after the code listed below, as appropriate.

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; R0 : tmp
;; R1 : Original SR
;; R2 : Original ICR
;; R3 : ICR Address
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
NMIH:
; (1) Set SR.IMASK = H'F
    stc    SR, R1 ;                Store SR
    mov    R1,R0
    or     #H'F0,R0
    ldc    R0, SR
; (2) Reverse ICR.NMIE
    mov.l  #ICR, R3
    mov.w  @R3, R2 ;                Store ICR
    mov.w  #H'0100, R0
    xor    R2, R0
    mov.w  R0, @R3 ;                Write ICR.NMIE inverted (dummy)
    bra    NMIH1
    nop
    .pool
    .align 4
NMIH2:
;
    mov.w  @R3, R0 ;                dummy read
    mov.w  R2, @R3 ;                Write ICR.NMIE
    stc    SR, R0
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR
    ldc    R0, SR

```


Section 20 User Break Controller (UBC)

20.1 Overview

The user break controller (UBC) provides functions that simplify program debugging. When break conditions are set in the UBC, a user break interrupt is generated according to the contents of the bus cycle generated by the CPU. This function makes it easy to design an effective self-monitoring debugger, enabling programs to be debugged with the chip alone, without using an in-circuit emulator.

20.1.1 Features

The UBC has the following features.

- Two break channels (A and B)
User break interrupts can be generated on independent conditions for channels A and B, or on sequential conditions (sequential break setting: channel A → channel B).
- The following can be set as break compare conditions:
 - Address (selection of 32-bit virtual address and ASID for comparison):
Address: All bits compared/lower 10 bits masked/lower 12 bits masked/lower 16 bits masked/lower 20 bits masked/all bits masked
ASID: All bits compared/all bits masked
 - Data (channel B only, 32-bit mask capability)
 - Bus cycle: Instruction access/operand access
 - Read/write
 - Operand size: Byte/word/longword/quadword
- An instruction access cycle break can be effected before or after the instruction is executed.

20.1.2 Block Diagram

Figure 20.1 shows a block diagram of the UBC.

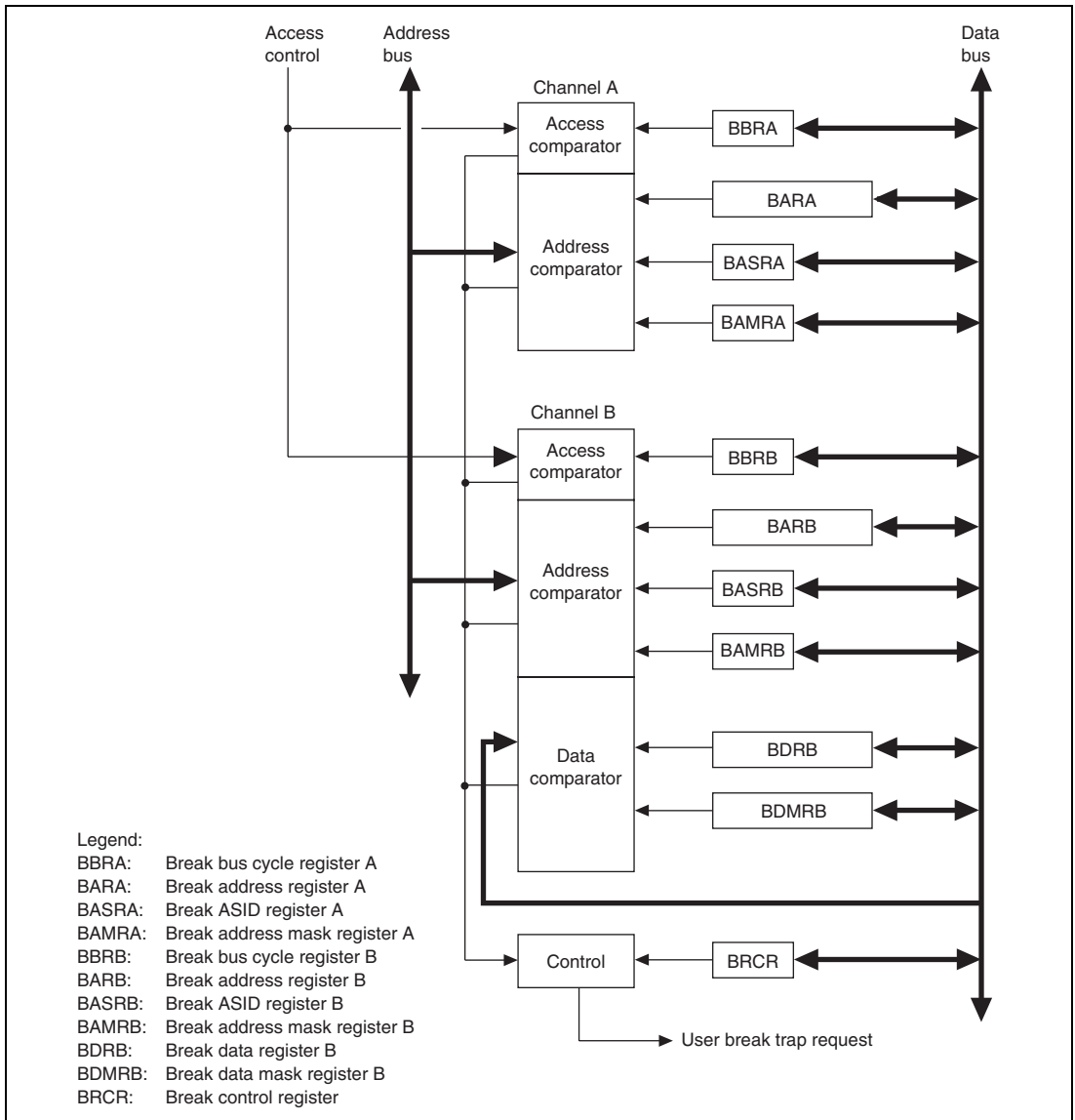


Figure 20.1 Block Diagram of User Break Controller

Table 20.1 shows the UBC registers.

Table 20.1 UBC Registers

| Name | Abbreviation | R/W | Initial Value | P4 Address | Area 7 Address | Access Size |
|-------------------------------|---------------------|------------|----------------------|-------------------|-----------------------|--------------------|
| Break address register A | BARA | R/W | Undefined | H'FF200000 | H'1F200000 | 32 |
| Break address mask register A | BAMRA | R/W | Undefined | H'FF200004 | H'1F200004 | 8 |
| Break bus cycle register A | BBRA | R/W | H'0000 | H'FF200008 | H'1F200008 | 16 |
| Break ASID register A | BASRA | R/W | Undefined | H'FF000014 | H'1F000014 | 8 |
| Break address register B | BARB | R/W | Undefined | H'FF20000C | H'1F20000C | 32 |
| Break address mask register B | BAMRB | R/W | Undefined | H'FF200010 | H'1F200010 | 8 |
| Break bus cycle register B | BBRB | R/W | H'0000 | H'FF200014 | H'1F200014 | 16 |
| Break ASID register B | BASRB | R/W | Undefined | H'FF000018 | H'1F000018 | 8 |
| Break data register B | BDRB | R/W | Undefined | H'FF200018 | H'1F200018 | 32 |
| Break data mask register B | BDMRB | R/W | Undefined | H'FF20001C | H'1F20001C | 32 |
| Break control register | BRCR | R/W | H'0000* | H'FF200020 | H'1F200020 | 16 |

Note: * Some bits are not initialized. See section 20.2.12, Break Control Register (BRCR), for details.

20.2 Register Descriptions

20.2.1 Access to UBC Registers

The access size must be the same as the control register size. If the sizes are different, a write will not be effected in a UBC register write operation, and a read operation will return an undefined value. UBC register contents cannot be transferred to a floating-point register using a floating-point memory load instruction.

When a UBC register is updated, use either of the following methods to make the updated value valid:

1. Execute an RTE instruction after the memory store instruction that updated the register. The updated value will be valid from the RTE instruction jump destination onward.
2. Execute instructions requiring 5 states for execution after the memory store instruction that updated the register. As the CPU executes two instructions in parallel and a minimum of 0.5 state is required for execution of one instruction, 11 instructions must be inserted. The updated value will be valid from the 6th state onward.

20.2.2 Break Address Register A (BARA)

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | BAA31 | BAA30 | BAA29 | BAA28 | BAA27 | BAA26 | BAA25 | BAA24 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BAA23 | BAA22 | BAA21 | BAA20 | BAA19 | BAA18 | BAA17 | BAA16 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | BAA15 | BAA14 | BAA13 | BAA12 | BAA11 | BAA10 | BAA9 | BAA8 |
| 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 |
| | BAA7 | BAA6 | BAA5 | BAA4 | BAA3 | BAA2 | BAA1 | BAA0 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Undefined

Break address register A (BARA) is a 32-bit readable/writable register that specifies the virtual address used in the channel A break conditions. BARA is not initialized by a power-on reset or manual reset.

Bits 31 to 0—Break Address A31 to A0 (BAA31–BAA0): These bits hold the virtual address (bits 31–0) used in the channel A break conditions.

20.2.3 Break ASID Register A (BASRA)

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | BASA7 | BASA6 | BASA5 | BASA4 | BASA3 | BASA2 | BASA1 | BASA0 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Undefined

Break ASID register A (BASRA) is an 8-bit readable/writable register that specifies the ASID used in the channel A break conditions. BASRA is not initialized by a power-on reset or manual reset.

Bits 7 to 0—Break ASID A7 to A0 (BASA7–BASA0): These bits hold the ASID (bits 7–0) used in the channel A break conditions.

20.2.4 Break Address Mask Register A (BAMRA)

| | | | | | | | | |
|----------------|---|---|---|---|-------|-------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | BAMA2 | BASMA | BAMA1 | BAMA0 |
| Initial value: | 0 | 0 | 0 | 0 | * | * | * | * |
| R/W: | R | R | R | R | R/W | R/W | R/W | R/W |

Note: * Undefined

Break address mask register A (BAMRA) is an 8-bit readable/writable register that specifies which bits are to be masked in the break ASID set in BASRA and the break address set in BARA. BAMRA is not initialized by a power-on reset or manual reset.

Bits 7 to 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 2—Break ASID Mask A (BASMA): Specifies whether all bits of the channel A break ASID (BASA7–BASA0) are to be masked.

| Bit 2: BASMA | Description |
|--------------|---|
| 0 | All BASRA bits are included in break conditions |
| 1 | No BASRA bits are included in break conditions |

Bits 3, 1, and 0—Break Address Mask A2 to A0 (BAMA2–BAMA0): These bits specify which bits of the channel A break address (BAA31–BAA0) set in BARA are to be masked.

| Bit 3: BAMA2 | Bit 1: BAMA1 | Bit 0: BAMA0 | Description |
|--------------|--------------|--------------|--|
| 0 | 0 | 0 | All BARA bits are included in break conditions |
| | | 1 | Lower 10 bits of BARA are masked, and not included in break conditions |
| | 1 | 0 | Lower 12 bits of BARA are masked, and not included in break conditions |
| | | 1 | All BARA bits are masked, and not included in break conditions |
| 1 | 0 | 0 | Lower 16 bits of BARA are masked, and not included in break conditions |
| | | 1 | Lower 20 bits of BARA are masked, and not included in break conditions |
| | 1 | * | Reserved (cannot be set) |

Legend: * Don't care

20.2.5 Break Bus Cycle Register A (BBRA)

| | | | | | | | | |
|----------------|----|------|------|------|------|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | SZA2 | IDA1 | IDA0 | RWA1 | RWA0 | SZA1 | SZA0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Break bus cycle register A (BBRA) is a 16-bit readable/writable register that sets three conditions—(1) instruction access/operand access, (2) read/write, and (3) operand size—from among the channel A break conditions.

BBRA is initialized to H'0000 by a power-on reset. It retains its value in standby mode.

Bits 15 to 7—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 5 and 4—Instruction Access/Operand Access Select A (IDA1, IDA0): These bits specify whether an instruction access cycle or an operand access cycle is used as the bus cycle in the channel A break conditions.

| Bit 5: IDA1 | Bit 4: IDA0 | Description |
|-------------|-------------|---|
| 0 | 0 | Condition comparison is not performed (Initial value) |
| | 1 | Instruction access cycle is used as break condition |
| 1 | 0 | Operand access cycle is used as break condition |
| | 1 | Instruction access cycle or operand access cycle is used as break condition |

Bits 3 and 2—Read/Write Select A (RWA1, RWA0): These bits specify whether a read cycle or write cycle is used as the bus cycle in the channel A break conditions.

| Bit 3: RWA1 | Bit 2: RWA0 | Description |
|-------------|-------------|---|
| 0 | 0 | Condition comparison is not performed (Initial value) |
| | 1 | Read cycle is used as break condition |
| 1 | 0 | Write cycle is used as break condition |
| | 1 | Read cycle or write cycle is used as break condition |

Bits 6, 1, and 0—Operand Size Select A (SZA2–SZA0): These bits select the operand size of the bus cycle used as a channel A break condition.

| Bit 6: SZA2 | Bit 1: SZA1 | Bit 0: SZA0 | Description |
|-------------|-------------|-------------|--|
| 0 | 0 | 0 | Operand size is not included in break conditions (Initial value) |
| | | 1 | Byte access is used as break condition |
| | 1 | 0 | Word access is used as break condition |
| | | 1 | Longword access is used as break condition |
| 1 | 0 | 0 | Quadword access is used as break condition |
| | | 1 | Reserved (cannot be set) |
| | 1 | * | Reserved (cannot be set) |

Legend: * Don't care

20.2.6 Break Address Register B (BARB)

BARB is the channel B break address register. The bit configuration is the same as for BARA.

20.2.7 Break ASID Register B (BASRB)

BASRB is the channel B break ASID register. The bit configuration is the same as for BASRA.

20.2.8 Break Address Mask Register B (BAMRB)

BAMRB is the channel B break address mask register. The bit configuration is the same as for BAMRA.

20.2.9 Break Data Register B (BDRB)

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | BDB31 | BDB30 | BDB29 | BDB28 | BDB27 | BDB26 | BDB25 | BDB24 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BDB23 | BDB22 | BDB21 | BDB20 | BDB19 | BDB18 | BDB17 | BDB16 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | BDB15 | BDB14 | BDB13 | BDB12 | BDB11 | BDB10 | BDB9 | BDB8 |
| 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 |
| | BDB7 | BDB6 | BDB5 | BDB4 | BDB3 | BDB2 | BDB1 | BDB0 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Undefined

Break data register B (BDRB) is a 32-bit readable/writable register that specifies the data (bits 31–0) to be used in the channel B break conditions. BDRB is not initialized by a power-on reset or manual reset.

Bits 31 to 0—Break Data B31 to B0 (BDB31–BDB0): These bits hold the data (bits 31–0) to be used in the channel B break conditions.

20.2.10 Break Data Mask Register B (BDMRB)

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | BDMB31 | BDMB30 | BDMB29 | BDMB28 | BDMB27 | BDMB26 | BDMB25 | BDMB24 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BDMB23 | BDMB22 | BDMB21 | BDMB20 | BDMB19 | BDMB18 | BDMB17 | BDMB16 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | BDMB15 | BDMB14 | BDMB13 | BDMB12 | BDMB11 | BDMB10 | BDMB9 | BDMB8 |
| 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 |
| | BDMB7 | BDMB6 | BDMB5 | BDMB4 | BDMB3 | BDMB2 | BDMB1 | BDMB0 |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Undefined

Break data mask register B (BDMRB) is a 32-bit readable/writable register that specifies which bits of the break data set in BDRB are to be masked. BDMRB is not initialized by a power-on reset or manual reset.

Bits 31 to 0—Break Data Mask B31 to B0 (BDMB31–BDMB0): These bits specify whether the corresponding bit of the channel B break data (BDB31–BDB0) set in BDRB is to be masked.

| Bit 31–0: BDMBn | Description |
|-----------------|---|
| 0 | Channel B break data bit BDBn is included in break conditions |
| 1 | Channel B break data bit BDBn is masked, and not included in break conditions |

Notes: n = 31 to 0

When the data bus value is included in the break conditions, the operand size should be specified. When byte size is specified, set the same data in bits 15–8 and 7–0 of BDRB and BDMRB.

20.2.11 Break Bus Cycle Register B (BBRB)

BBRB is the channel B bus break register. The bit configuration is the same as for BBRA.

20.2.12 Break Control Register (BRCR)

| | | | | | | | | |
|----------------|------|------|----|----|-----|------|---|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | CMFA | CMFB | — | — | — | PCBA | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | * | 0 | 0 |
| R/W: | R/W | R/W | R | R | R | R/W | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DBEB | PCBB | — | — | SEQ | — | — | UBDE |
| Initial value: | * | * | 0 | 0 | * | 0 | 0 | 0 |
| R/W: | R/W | R/W | R | R | R/W | R | R | R/W |

Note: * Undefined

The break control register (BRCR) is a 16-bit readable/writable register that specifies (1) whether channels A and B are to be used as two independent channels or in a sequential condition, (2) whether the break is to be effected before or after instruction execution, (3) whether the BDRB register is to be included in the channel B break conditions, and (4) whether the user break debug function is to be used. BRCR also contains condition match flags. The CMFA, CMFB, and UBDE bits in BRCR are initialized to 0 by a power-on reset, but retain their value in standby mode. The value of the PCBA, DBEB, PCBB, and SEQ bits is undefined after a power-on reset or manual reset, so these bits should be initialized by software as necessary.

Bit 15—Condition Match Flag A (CMFA): Set to 1 when a break condition set for channel A is satisfied. This flag is not cleared to 0 (to confirm that the flag is set again after once being set, it should be cleared with a write).

| Bit 15: CMFA | Description |
|--------------|--|
| 0 | Channel A break condition is not matched (Initial value) |
| 1 | Channel A break condition match has occurred |

Bit 14—Condition Match Flag B (CMFB): Set to 1 when a break condition set for channel B is satisfied. This flag is not cleared to 0 (to confirm that the flag is set again after once being set, it should be cleared with a write).

| Bit 14: CMFB | Description |
|--------------|--|
| 0 | Channel B break condition is not matched (Initial value) |
| 1 | Channel B break condition match has occurred |

Bits 13 to 11—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 10—Instruction Access Break Select A (PCBA): Specifies whether a channel A instruction access cycle break is to be effected before or after the instruction is executed. This bit is not initialized by a power-on reset or manual reset.

| Bit 10: PCBA | Description |
|--------------|---|
| 0 | Channel A PC break is effected before instruction execution |
| 1 | Channel A PC break is effected after instruction execution |

Bits 9 and 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Data Break Enable B (DBEB): Specifies whether the data bus condition is to be included in the channel B break conditions. This bit is not initialized by a power-on reset or manual reset.

| Bit 7: DBEB | Description |
|-------------|--|
| 0 | Data bus condition is not included in channel B conditions |
| 1 | Data bus condition is included in channel B conditions |

Note: When the data bus is included in the break conditions, bits IDB1–0 in break bus cycle register B (BBRB) should be set to 10 or 11.

Bit 6—PC Break Select B (PCBB): Specifies whether a channel B instruction access cycle break is to be effected before or after the instruction is executed. This bit is not initialized by a power-on reset or manual reset.

| Bit 6: PCBB | Description |
|-------------|---|
| 0 | Channel B PC break is effected before instruction execution |
| 1 | Channel B PC break is effected after instruction execution |

Bits 5 and 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Sequence Condition Select (SEQ): Specifies whether the conditions for channels A and B are to be independent or sequential. This bit is not initialized by a power-on reset or manual reset.

| Bit 3: SEQ | Description |
|------------|--|
| 0 | Channel A and B comparisons are performed as independent conditions |
| 1 | Channel A and B comparisons are performed as sequential conditions (channel A → channel B) |

Bits 2 and 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 0—User Break Debug Enable (UBDE): Specifies whether the user break debug function (see section 20.4, User Break Debug Support Function) is to be used.

| Bit 0: UBDE | Description |
|-------------|---|
| 0 | User break debug function is not used (Initial value) |
| 1 | User break debug function is used |

20.3 Operation

20.3.1 Explanation of Terms Relating to Accesses

An instruction access is an access that obtains an instruction. For example, the fetching of an instruction from the branch destination when a branch instruction is executed is an instruction access. An operand access is any memory access for the purpose of instruction execution. For example, the access to address $PC+disp \times 2+4$ in the instruction `MOV.W@(disp,PC), Rn` is an operand access. As the term “data” is used to distinguish data from an address, the term “operand access” is used in this section.

In this LSI, all operand accesses are treated as either read accesses or write accesses. The following instructions require special attention:

- `PREF`, `OCBP`, and `OCBWB` instructions: Treated as read accesses.
- `MOVCA.L` and `OCBI` instructions: Treated as write accesses.
- `TAS.B` instruction: Treated as one read access and one write access.

The operand accesses for the `PREF`, `OCBP`, `OCBWB`, and `OCBI` instructions are accesses with no access data.

This LSI handles all operand accesses as having a data size. The data size can be byte, word, longword, or quadword. The operand data size for the `PREF`, `OCBP`, `OCBWB`, `MOVCA.L`, and `OCBI` instructions is treated as longword.

20.3.2 Explanation of Terms Relating to Instruction Intervals

In this section, “1 (2, 3, ...) instruction(s) after...”, as a measure of the distance between two instructions, is defined as follows. A branch is counted as an interval of two instructions.

- Example of sequence of instructions with no branch:
 - 100 Instruction A (0 instructions after instruction A)
 - 102 Instruction B (1 instruction after instruction A)
 - 104 Instruction C (2 instructions after instruction A)
 - 106 Instruction D (3 instructions after instruction A)

- Example of sequence of instructions with a branch (however, the example of a sequence of instructions with no branch should be applied when the branch destination of a delayed branch instruction is the instruction itself + 4):

```

100 Instruction A: BT/S L200 (0 instructions after instruction A)
102 Instruction B (1 instruction after instruction A, 0 instructions after instruction B)
L200 200 Instruction C (3 instructions after instruction A, 2 instructions after instruction B)
202 Instruction D (4 instructions after instruction A, 3 instructions after instruction
    B)

```

20.3.3 User Break Operation Sequence

The sequence of operations from setting of break conditions to user break exception handling is described below.

1. Specify pre- or post-execution breaking in the case of an instruction access, inclusion or exclusion of the data bus value in the break conditions in the case of an operand access, and use of independent or sequential channel A and B break conditions, in the break control register (BRCR). Set the break addresses in the break address registers for each channel (BARA, BARB), the ASIDs corresponding to the break space in the break ASID registers (BASRA, BASRB), and the address and ASID masking methods in the break address mask registers (BAMRA, BAMRB). If the data bus value is to be included in the break conditions, also set the break data in the break data register (BDRB) and the data mask in the break data mask register (BDMRB).
2. Set the break bus conditions in the break bus cycle registers (BBRA, BBRB). If even one of the BBRA/BBRB instruction access/operand access select (ID bit) and read/write select groups (RW bit) is set to 00, a user break interrupt will not be generated on the corresponding channel. Make the BBRA and BBRB settings after all other break-related register settings have been completed. If breaks are enabled with BBRA/BBRB while the break address, data, or mask register, or the break control register is in the initial state after a reset, a break may be generated inadvertently.
3. The operation when a break condition is satisfied depends on the BL bit (in the CPU's SR register). When the BL bit is 0, exception handling is started and the condition match flag (CMFA/CMFB) for the respective channel is set for the matched condition. When the BL bit is 1, the condition match flag (CMFA/CMFB) for the respective channel is set for the matched condition but exception handling is not started.

The condition match flags (CMFA, CMFB) are set by a branch condition match, but are not reset. Therefore, a memory store instruction should be used on the BRCR register to clear the

flags to 0. See section 20.3.6, Condition Match Flag Setting, for the exact setting conditions for the condition match flags.

4. When sequential condition mode has been selected, and the channel B condition is matched after the channel A condition has been matched, a break is effected at the instruction at which the channel B condition was matched. See section 20.3.8, Contiguous A and B Settings for Sequential Conditions, for the operation when the channel A condition match and channel B condition match occur close together. With sequential conditions, only the channel B condition match flag is set. When sequential condition mode has been selected, if it is wished to clear the channel A match when the channel A condition has been matched but the channel B condition has not yet been matched, this can be done by writing 0 to the SEQ bit in the BRRC register.

20.3.4 Instruction Access Cycle Break

1. When an instruction access/read/word setting is made in the break bus cycle register (BBRA/BBRB), an instruction access cycle can be used as a break condition. In this case, breaking before or after execution of the relevant instruction can be selected with the PCBA/PCBB bit in the break control register (BRCC). When an instruction access cycle is used as a break condition, clear the LSB of the break address registers (BARA, BARB) to 0. A break will not be generated if this bit is set to 1.
2. When a pre-execution break is specified, the break is effected when it is confirmed that the instruction is to be fetched and executed. Therefore, overrun-fetched instructions (instructions that are fetched but not executed when a branch or exception occurs) cannot be used in a break. However, if a TLB miss or TLB protection violation exception occurs at the time of the fetch of instructions subject to a break, the break exception handling is carried out first. The instruction TLB exception handling is performed when the instruction is re-executed (see section 5.4, Exception Types and Priorities). Also, since a delayed branch instruction and the delay slot instruction are executed as a single instruction, if a pre-execution break is specified for a delay slot instruction, the break will be effected before execution of the delayed branch instruction. However, a pre-execution break cannot be specified for the delay slot instruction for an RTE instruction.
3. With a post-execution break, the instruction set as a break condition is executed, then a break interrupt is generated before the next instruction is executed. When a post-execution break is set for a delayed branch instruction, the delay slot is executed and the break is effected before execution of the instruction at the branch destination (when the branch is made) or the instruction two instructions ahead of the branch instruction (when the branch is not made).

- When an instruction access cycle is set for channel B, break data register B (BDRB) is ignored in judging whether there is an instruction access match. Therefore, a break condition specified by the DBEB bit in BR CR is not executed.

20.3.5 Operand Access Cycle Break

- In the case of an operand access cycle break, the bits included in address bus comparison vary as shown below according to the data size specification in the break bus cycle register (BBRA/BBRB).

| Data Size | Address Bits Compared |
|---------------------------------|--|
| Quadword (100) | Address bits A31–A3 |
| Longword (011) | Address bits A31–A2 |
| Word (010) | Address bits A31–A1 |
| Byte (001) | Address bits A31–A0 |
| Not included in condition (000) | In quadword access, address bits A31–A3 In longword access, address bits A31–A2 In word access, address bits A31–A1 In byte access, address bits A31–A0 |

- When data bus is included in break conditions in channel B

When a data value is included in the break conditions, set the DBEB bit in the break control register (BR CR) to 1. In this case, break data register B (BDRB) and break data mask register B (BDMRB) settings are necessary in addition to the address condition. A user break interrupt is generated when all three conditions—address, ASID, and data—are matched. When a quadword access occurs, the 64-bit access data is divided into an upper 32 bits and lower 32 bits, and interpreted as two 32-bit data units. A break is generated if either of the 32-bit data units satisfies the data match condition.

Set the IDB1–0 bits in break bus cycle register B (BBRB) to 10 or 11. When byte data is specified, the same data should be set in the two bytes comprising bits 15–8 and bits 7–0 in break data register B (BDRB) and break data mask register B (BDMRB). When word or byte is set, bits 31–16 of BDRB and BDMRB are ignored.

- When the DBEB bit in the break control register (BR CR) is set to 1, a break is not generated by an operand access with no access data (an operand access in a PREF, OCBP, OCBWB, or OCBI instruction).

20.3.6 Condition Match Flag Setting

1. Instruction access with post-execution condition, or operand access

The flag is set when execution of the instruction that causes the break is completed. As an exception to this, however, in the case of an instruction with more than one operand access the flag may be set on detection of the match condition alone, without waiting for execution of the instruction to be completed.

Example 1:

100 BT L200 (branch performed)

102 Instruction (operand access break on channel A) → flag not set

Example 2:

110 FADD (FPU exception)

112 Instruction (operand access break on channel A) → flag not set

2. Instruction access with pre-execution condition

The flag is set when the break match condition is detected.

Example 1:

110 Instruction (pre-execution break on channel A) → flag set

112 Instruction (pre-execution break on channel B) → flag not set

Example 2:

110 Instruction (pre-execution break on channel B, instruction access TLB miss) → flag set

20.3.7 Program Counter (PC) Value Saved

1. When instruction access (pre-execution) is set as a break condition, the program counter (PC) value saved to SPC in user break interrupt handling is the address of the instruction at which the break condition match occurred. In this case, a user break interrupt is generated and the fetched instruction is not executed.
2. When instruction access (post-execution) is set as a break condition, the program counter (PC) value saved to SPC in user break interrupt handling is the address of the instruction to be executed after the instruction at which the break condition match occurred. In this case, the fetched instruction is executed, and a user break interrupt is generated before execution of the next instruction.
3. When an instruction access (post-execution) break condition is set for a delayed branch instruction, the delay slot instruction is executed and a user break is effected before execution of the instruction at the branch destination (when the branch is made) or the instruction two instructions ahead of the branch instruction (when the branch is not made). In this case, the PC

value saved to SPC is the address of the branch destination (when the branch is made) or the instruction following the delay slot instruction (when the branch is not made).

4. When operand access (address only) is set as a break condition, the address of the instruction to be executed after the instruction at which the condition match occurred is saved to SPC.
5. When operand access (address + data) is set as a break condition, execution of the instruction at which the condition match occurred is completed. A user break interrupt is generated before execution of instructions from one instruction later to four instructions later. It is not possible to specify at which instruction, from one later to four later, the interrupt will be generated. The start address of the instruction after the instruction for which execution is completed at the point at which user break interrupt handling is started is saved to SPC. If an instruction between one instruction later and four instructions later causes another exception, control is performed as follows. Designating the exception caused by the break as exception 1, and the exception caused by an instruction between one instruction later and four instructions later as exception 2, memory updating and register updating that essentially cannot be performed by exception 2 cannot be performed is guaranteed irrespective of the existence of exception 1. The program counter value saved is the address of the first instruction for which execution is suppressed. Whether exception 1 or exception 2 is used for the exception jump destination and the value written to the exception register (EXPEVT/INTEVT) is not guaranteed. However, if exception 2 is from a source not synchronized with an instruction (external interrupt or peripheral module interrupt), exception 1 is used for the exception jump destination and the value written to the exception register (EXPEVT/INTEVT).

20.3.8 Contiguous A and B Settings for Sequential Conditions

When channel A match and channel B match timings are close together, a sequential break may not be guaranteed. Rules relating to the guaranteed range are given below.

1. Instruction access matches on both channel A and channel B

| | |
|---|---|
| Instruction B is 0 instructions after instruction A | Equivalent to setting the same address. Do not use this setting |
| Instruction B is 1 instruction after instruction A | Sequential operation is not guaranteed |
| Instruction B is 2 or more instructions after instruction A | Sequential operation is guaranteed |

2. Instruction access match on channel A, operand access match on channel B

| | |
|---|--|
| Instruction B is 0 or 1 instruction after instruction A | Sequential operation is not guaranteed |
|---|--|

| | |
|---|------------------------------------|
| Instruction B is 2 or more instructions after instruction A | Sequential operation is guaranteed |
|---|------------------------------------|

3. Operand access match on channel A, instruction access match on channel B

| | |
|--|--|
| Instruction B is 0 to 3 instructions after instruction A | Sequential operation is not guaranteed |
|--|--|

| | |
|---|------------------------------------|
| Instruction B is 4 or more instructions after instruction A | Sequential operation is guaranteed |
|---|------------------------------------|

4. Operand access matches on both channel A and channel B

Do not make a setting such that a single operand access will match the break conditions of both channel A and channel B. There are no other restrictions. For example, sequential operation is guaranteed even if two accesses within a single instruction match channel A and channel B conditions in turn.

20.3.9 Usage Notes

1. Do not execute a post-execution instruction access break for the SLEEP instruction.
2. Do not make an operand access break setting between 1 and 3 instructions before a SLEEP instruction.
3. The value of the BL bit referenced in a user break exception depends on the break setting, as follows.
 - a. Pre-execution instruction access break: The BL bit value before the executed instruction is referenced.
 - b. Post-execution instruction access break: The OR of the BL bit values before and after the executed instruction is referenced.
 - c. Operand access break (address/data): The BL bit value after the executed instruction is referenced.
 - d. In the case of an instruction that modifies the BL bit

| SL.BL | Pre-Execution Instruction Access | Post-Execution Instruction Access | Pre-Execution Instruction Access | Post-Execution Instruction Access | Operand Access (Address/Data) |
|-------|----------------------------------|-----------------------------------|----------------------------------|-----------------------------------|-------------------------------|
| 0 → 0 | A | A | A | A | A |
| 1 → 0 | M | M | M | M | A |
| 0 → 1 | A | M | A | M | M |
| 1 → 1 | M | M | M | M | M |

Legend:

A: Accepted

M: Masked

- e. In the case of an RTE delay slot
 The BL bit value before execution of a delay slot instruction is the same as the BL bit value before execution of an RTE instruction. The BL bit value after execution of a delay slot instruction is the same as the first BL bit value for the first instruction executed on returning by means of an RTE instruction (the same as the value of the BL bit in SSR before execution of the RTE instruction).
 - f. If an interrupt or exception is accepted with the BL bit cleared to 0, the value of the BL bit before execution of the first instruction of the exception handling routine is 1.
4. If channels A and B both match independently at virtually the same time, and, as a result, the SPC value is the same for both user break interrupts, only one user break interrupt is generated, but both the CMFA bit and the CMFB bit are set. For example:
 - 110 Instruction (post-execution instruction break on channel A) → SPC = 112, CMFA = 1
 - 112 Instruction (pre-execution instruction break on channel B) → SPC = 112, CMFB = 1
 5. The PCBA or PCBB bit in BRCCR is valid for an instruction access break setting.
 6. When the SEQ bit in BRCCR is 1, the internal sequential break state is initialized by a channel B condition match. For example: A → A → B (user break generated) → B (no break generated)
 7. In the event of contention between a re-execution type exception and a post-execution break in a multistep instruction, the re-execution type exception is generated. In this case, the CMF bit may or may not be set to 1 when the break condition occurs.
 8. A post-execution break is classified as a completion type exception. Consequently, in the event of contention between a completion type exception and a post-execution break, the post-execution break is suppressed in accordance with the priorities of the two events. For example,

in the case of contention between a TRAPA instruction and a post-execution break, the user break is suppressed. However, in this case, the CMF bit is set by the occurrence of the break condition.

20.4 User Break Debug Support Function

The user break debug support function enables the processing used in the event of a user break exception to be changed. When a user break exception occurs, if the UBDE bit is set to 1 in the BRCR register, the DBR register value will be used as the branch destination address instead of [VBR + offset]. The value of R15 is saved in the SGR register regardless of the value of the UBDE bit in the BRCR register or the kind of exception event. A flowchart of the user break debug support function is shown in figure 20.2.

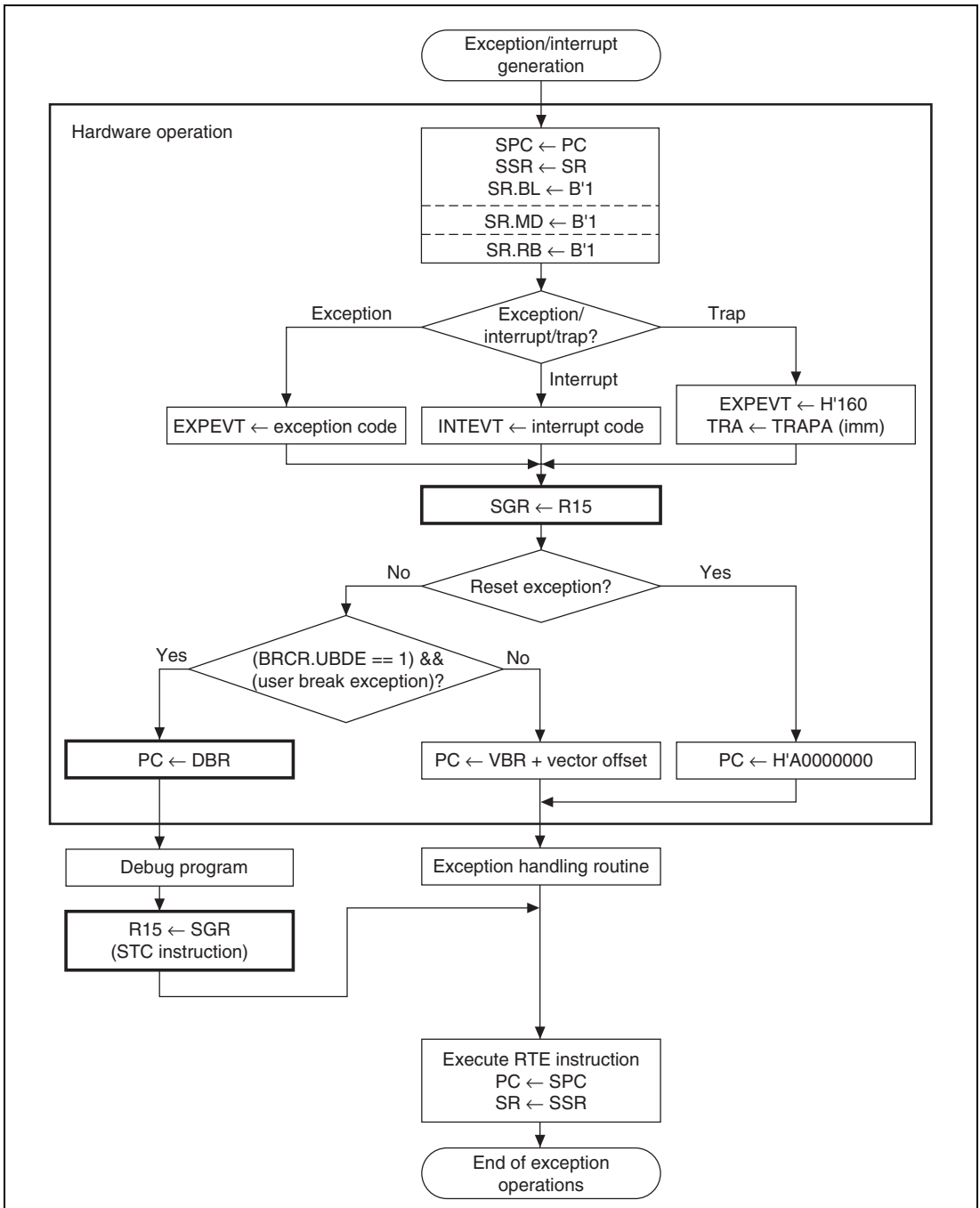


Figure 20.2 User Break Debug Support Function Flowchart

20.5 Examples of Use

Instruction Access Cycle Break Condition Settings

- Register settings: BASRA = H'80 / BARA = H'00000404 / BAMRA = H'00 / BBRA = H'0014 / BASRB = H'70 / BARB = H'00008010 / BAMRB = H'01 / BBRB = H'0014 / BDRB = H'00000000 / BDMRB = H'00000000 / BRCR = H'0400

Conditions set: Independent channel A/channel B mode

— Channel A: ASID: H'80 / address: H'00000404 / address mask: H'00

Bus cycle: instruction access (post-instruction-execution), read (operand size not included in conditions)

— Channel B: ASID: H'70 / address: H'00008010 / address mask: H'01

Data: H'00000000 / data mask: H'00000000

Bus cycle: instruction access (pre-instruction-execution), read (operand size not included in conditions)

A user break is generated after execution of the instruction at address H'00000404 with ASID = H'80, or before execution of an instruction at addresses H'00008000–H'000083FE with ASID = H'70.

- Register settings: BASRA = H'80 / BARA = H'00037226 / BAMRA = H'00 / BBRA = H'0016 / BASRB = H'70 / BARB = H'0003722E / BAMRB = H'00 / BBRB = H'0016 / BDRB = H'00000000 / BDMRB = H'00000000 / BRCR = H'0008

Conditions set: Channel A → channel B sequential mode

— Channel A: ASID: H'80 / address: H'00037226 / address mask: H'00

Bus cycle: instruction access (pre-instruction-execution), read, word

— Channel B: ASID: H'70 / address: H'0003722E / address mask: H'00

Data: H'00000000 / data mask: H'00000000

Bus cycle: instruction access (pre-instruction-execution), read, word

The instruction at address H'00037266 with ASID = H'80 is executed, then a user break is generated before execution of the instruction at address H'0003722E with ASID = H'70.

- Register settings: BASRA = H'80 / BARA = H'00027128 / BAMRA = H'00 / BBRA = H'001A / BASRB = H'70 / BARB = H'00031415 / BAMRB = H'00 / BBRB = H'0014 / BDRB = H'00000000 / BDMRB = H'00000000 / BRCR = H'0000

Conditions set: Independent channel A/channel B mode

— Channel A: ASID: H'80 / address: H'00027128 / address mask: H'00

Bus cycle: CPU, instruction access (pre-instruction-execution), write, word

— Channel B: ASID: H'70 / address: H'00031415 / address mask: H'00

Data: H'00000000 / data mask: H'00000000

Bus cycle: CPU, instruction access (pre-instruction-execution), read (operand size not included in conditions)

A user break interrupt is not generated on channel A since the instruction access is not a write cycle.

A user break interrupt is not generated on channel B since instruction access is performed on an even address.

Operand Access Cycle Break Condition Settings

- Register settings: BASRA = H'80 / BARA = H'00123456 / BAMRA = H'00 / BBRA = H'0024 / BASRB = H'70 / BARB = H'000ABCDE / BAMRB = H'02 / BBRB = H'002A / BDRB = H'0000A512 / BDMRB = H'00000000 / BR CR = H'0080

Conditions set: Independent channel A/channel B mode

— Channel A: ASID: H'80 / address: H'00123456 / address mask: H'00

Bus cycle: operand access, read (operand size not included in conditions)

— Channel B: ASID: H'70 / address: H'000ABCDE / address mask: H'02

Data: H'0000A512 / data mask: H'00000000

Bus cycle: operand access, write, word

Data break enabled

On channel A, a user break interrupt is generated in the event of a longword read at address H'00123454, a word read at address H'00123456, or a byte read at address H'00123456, with ASID = H'80.

On channel B, a user break interrupt is generated when H'A512 is written by word access to any address from H'000AB000 to H'000ABFFE with ASID = H'70.

20.6 User Break Controller Stop Function

This function stops the clock supplied to the user break controller and is used to minimize power dissipation when the chip is operating. Note that, if you use this function, you cannot use the user break controller.

20.6.1 Transition to User Break Controller Stopped State

Setting the MSTP5 bit of the STBCR2 (inside the CPG) to 1 stops the clock supply and causes the user break controller to enter the stopped state. Follow steps (1) to (5) below to set the MSTP5 bit to 1 and enter the stopped state.

- (1) Initialize BBRA and BBRB to 0;
- (2) Initialize BRCCR to 0;
- (3) Make a dummy read of BRCCR;
- (4) Read STBCR2, then set the MSTP5 bit in the read data to 1 and write back.
- (5) Make two dummy reads of STBCR2.

Make sure that, if an exception or interrupt occurs while performing steps (1) to (5), you do not change the values of these registers in the exception handling routine.

Do not read or write the following registers while the user break controller clock is stopped: BARA, BAMRA, BBRA, BARB, BAMRB, BBRB, BDRB, BDMRB, and BRCCR. If these registers are read or written, the value cannot be guaranteed.

20.6.2 Cancelling the User Break Controller Stopped State

The clock supply can be restarted by setting the MSTP5 bit of STBCR2 (inside the CPG) to 0. The user break controller can then be operated again. Follow steps (6) and (7) below to clear the MSTP5 bit to 0 to cancel the stopped state.

- (6) Read STBCR2, then clear the MSTP5 bit in the read data to 0 and write the modified data back;
- (7) Make two dummy reads of STBGR2.

As with the transition to the stopped state, if an exception or interrupt occurs while processing steps (6) and (7), make sure that the values in these registers are not changed in the exception handling routine.

20.6.3 Examples of Stopping and Restarting the User Break Controller

The following are example programs:

```
; Transition to user break controller stopped state
; (1) Initialize BBRA and BBRB to 0.
    mov     #0, R0
    mov.l   #BBRA, R1
    mov.w   R0, @R1
    mov.l   #BBRB, R1
    mov.w   R0, @R1
; (2) Initialize BRCCR to 0.
    mov.l   #BRCCR, R1
    mov.w   R0, @R1
; (3) Dummy read BRCCR.
    mov.w   @R1, R0
; (4) Read STBCR2, then set MSTP5 bit in the read data to 1 and write
    it back
    mov.l   #STBCR2, R1
    mov.b   @R1, R0
    or      #H'1, R0
    mov.b   R0, @R1
; (5) Twice dummy read STBCR2.
    mov.b   @R1, R0
    mov.b   @R1, R0
; Canceling user break controller stopped state
; (6) Read STBCR2, then clear MSTP5 bit in the read data to 0 and write
    it back
    mov.l   #STBCR2, R1
    mov.b   @R1, R0
    and     #H'FE, R0
    mov.b   R0, @R1
; (7) Twice dummy read STBCR2.
    mov.b   @R1, R0
    mov.b   @R1, R0
```


Section 21 High-performance User Debug Interface (H-UDI)

21.1 Overview

21.1.1 Features

The high-performance user debug interface (H-UDI) is a serial input/output interface supporting a subset of the JTAG, IEEE 1149.1, IEEE Standard Test Access Port and Boundary-Scan Architecture. This LSI H-UDI support boundary-scan, and is used for emulator connection. The functions of this interface should not be used when using an emulator. Refer to the emulator manual for the method of connecting the emulator. The H-UDI uses six pins (TCK, TMS, TDI, TDO, $\overline{\text{TRST}}$, and $\overline{\text{ASEBRK/BRKACK}}$). In this LSI, six dedicated emulator pins have been added (AUDSYNC, AUDCK, and AUDATA3 to AUDATA0). The pin functions and serial transfer protocol conform to the JTAG specifications.

21.1.2 Block Diagram

Figure 21.1 shows a block diagram of the H-UDI. The TAP (test access port) controller and control registers are reset independently of the chip reset pin by driving the $\overline{\text{TRST}}$ pin low or setting TMS to 1 and applying TCK for at least five clock cycles. The other circuits are reset and initialized in an ordinary reset. The H-UDI circuit has six internal registers: SDBPR, SDBSR, SDIR, SDINT, SDDRH, and SDDRL (these last two together designated SDDR). The SDBPR register supports the JTAG bypass mode, SDBSR is a shift register forming a JTAG boundary scan, SDIR is the command register, SDDR is the data register, and SDINT is the H-UDI interrupt register. SDIR can be accessed directly from the TDI and TDO pins.

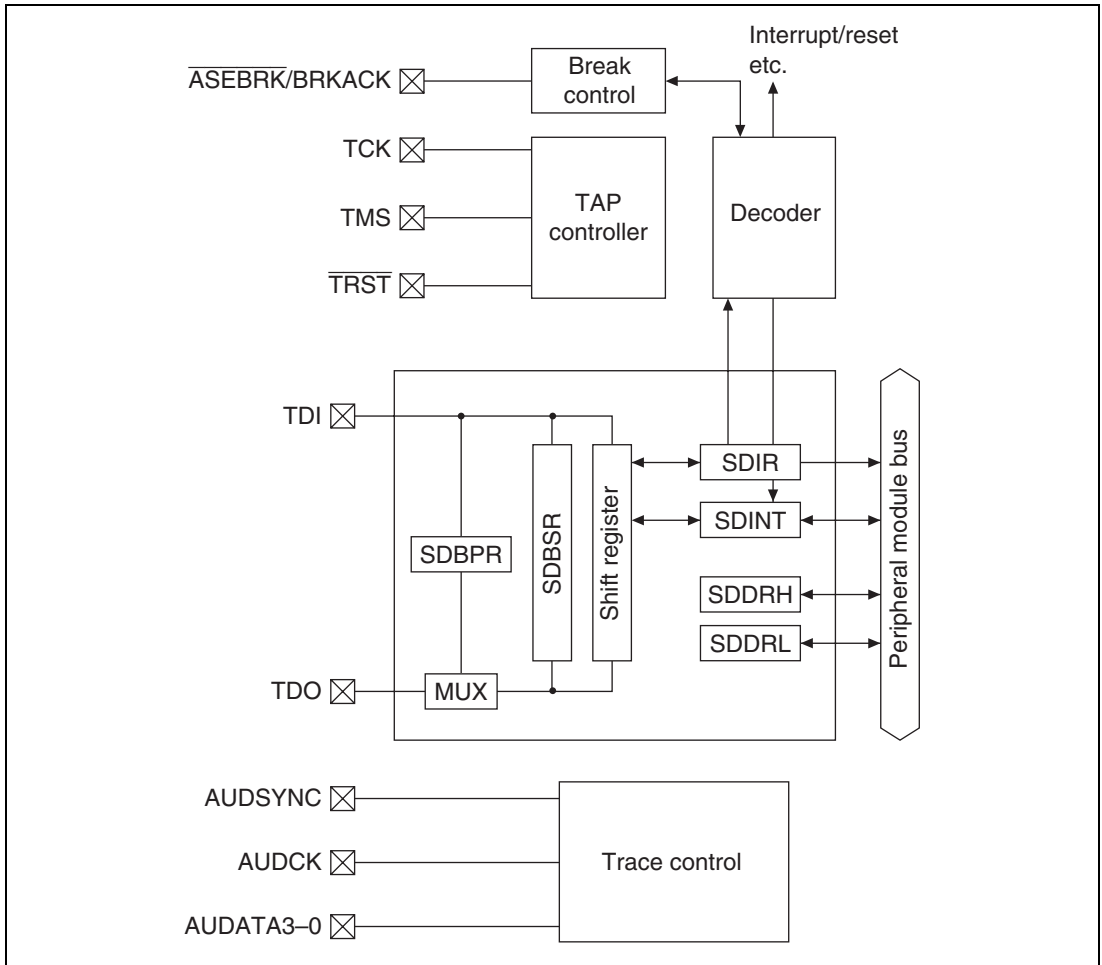


Figure 21.1 Block Diagram of H-UDI Circuit

21.1.3 Pin Configuration

Table 21.1 shows the H-UDI pin configuration.

Table 21.1 H-UDI Pins

| Pin Name | Abbreviation | I/O | Function | When Not Used |
|-----------------|---|--------------|--|---------------------------------|
| Clock pin | TCK | Input | Same as the JTAG serial clock input pin. Data is transferred from data input pin TDI to the H-UDI circuit, and data is read from data output pin TDO, in synchronization with this signal. | Open* ¹ |
| Mode pin | TMS | Input | The mode select input pin. Changing this signal in synchronization with TCK determines the meaning of the data input from TDI. The protocol conforms to the JTAG (IEEE Std 1149.1) specification. | Open* ¹ |
| Reset pin | $\overline{\text{TRST}}$ | Input | The input pin that resets the H-UDI. This signal is received asynchronously with respect to TCK, and effects a reset of the JTAG interface circuit when low. $\overline{\text{TRST}}$ must be driven low for a certain period when powering on, regardless of whether or not JTAG is used. This differs from the IEEE specification. | * ² , * ³ |
| Data input pin | TDI | Input | The data input pin. Data is sent to the H-UDI circuit by changing this signal in synchronization with TCK. | Open* ¹ |
| Data output pin | TDO | Output | The data output pin. Data is sent to the H-UDI circuit by reading this signal in synchronization with TCK. | Open |
| Emulator pin | $\overline{\text{ASEBRK/BRKACK}}$ | Input/output | Dedicated emulator pin | Open* ¹ |
| | AUDSYNC AUDCK AUDATA3– AUDATA0 | Output | Dedicated emulator pin | Open |

- Notes:
1. Pulled up inside the chip. When designing a board that allows use of an emulator, or when using interrupts and resets via the H-UDI, there is no problem in connecting a pullup resistance externally.
 2. When designing a board that enables the use of an emulator, or when using interrupts and resets via the H-UDI, drive $\overline{\text{TRST}}$ low for a period overlapping $\overline{\text{RESET}}$ at power-on, and also provide for control by $\overline{\text{TRST}}$ alone.

3. Fixed to the ground or connected to the same signal line as $\overline{\text{RESET}}$, or to a signal line that behaves in the same way. However, there is a problem when this pin is fixed to the ground. $\overline{\text{TRST}}$ is pulled up in the chip so, when this pin is fixed to the ground via external connection, a minute current will flow. The size of this current is determined by the rating of the pull-up resistor. Although this current has no effect on the chip's operation, unnecessary current will be dissipated.

The maximum frequency of TCK (TMS, TDI, TDO) is 20 MHz. Make the TCK or this LSI CPG setting so that the TCK frequency is lower than that of this LSI's peripheral module clock.

21.1.4 Register Configuration

Table 21.2 shows the H-UDI registers. Except for SDBPR and SDBSR, these registers are mapped in the control register space and can be referenced by the CPU.

Table 21.2 H-UDI Registers

| Name | Abbreviation | CPU Side | | | | | H-UDI Side | | |
|---------------------------|----------------|----------------------|-------------------|----------------|--------------------------------|-----------------------|--------------------------------|----|--|
| | | P4 R/W Address | Area 7 Address | Access Size | Initial Value* ¹ | Access R/W Size | Initial Value* ¹ | | |
| Instruction register | SDIR | R | H'FFF00000 | H'1FF00000 | 16 | H'FFFF | R/W | 32 | H'FFFFFFFD (Fixed value* ²) |
| Data register H | SDDR/ SDDRH | R/W | H'FFF00008 | H'1FF00008 | 32/16 | Undefined | — | — | — |
| Data register L | SDDRL | R/W | H'FFF0000A | H'1FF0000A | 16 | Undefined | — | — | — |
| Bypass register | SDBPR | — | — | — | — | Undefined | R/W | 1 | — |
| Interrupt factor register | SDINT | R/W | H'FFF00014 | H'1FF00014 | 16 | H'0000 | W* ³ | 32 | H'00000000 |
| Boundary scan register | SDBSR | — | — | — | — | Undefined | R/W | — | Undefined |

Notes: 1. Initialized when the $\overline{\text{TRST}}$ pin goes low or when the TAP is in the Test-Logic-Reset state.

2. The value read from H-UDI is fixed (H'FFFFFFFD).

3. 1 can be written to the LSB using the H-UDI interrupt command.

21.2 Register Descriptions

21.2.1 Instruction Register (SDIR)

The instruction register (SDIR) is a 16-bit register that can only be read by the CPU. In the initial state, bypass mode is set. The value (command) is set from the serial input pin (TDI). SDIR is initialized by the TRST pin or in the TAP Test-Logic-Reset state. When this register is written to from the H-UDI, writing is possible regardless of the CPU mode. Operation is undefined if a reserved command is set in this register.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | TI7 | TI6 | TI5 | TI4 | TI3 | TI2 | TI1 | TI0 |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W: | R | R | R | R | R | R | R | R |

Bits 15 to 8—Test Instruction Bits (TI7–TI0)

| Bit 15: TI7 | Bit 14: TI6 | Bit 13: TI5 | Bit 12: TI4 | Bit 11: TI3 | Bit 10: TI2 | Bit 9: TI1 | Bit 8: TI0 | Description |
|------------------|----------------|----------------|----------------|----------------|----------------|---------------|---------------|-----------------------------|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | EXTEST |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | SAMPLE/PRELOAD |
| 0 | 1 | 1 | 0 | — | — | — | — | H-UDI reset negate |
| 0 | 1 | 1 | 1 | — | — | — | — | H-UDI reset assert |
| 1 | 0 | 1 | — | — | — | — | — | H-UDI interrupt |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Bypass mode (Initial value) |
| Other than above | | | | | | | | Reserved |

Bits 7 to 0—Reserved: These bits are always read as 1, and should only be written with 1.

21.2.2 Data Register (SDDR)

The data register (SDDR) is a 32-bit register, comprising the two 16-bit registers SDDRH and SDDRL, that can be read and written to by the CPU. The value in this register is initialized by $\overline{\text{TRST}}$, but not by a CPU reset.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | | | | | | | | |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | | | | | | |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | | | | |
| 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 |
| | | | | | | | | |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Undefined

Bits 31 to 0—DR Data: These bits store the SDDR value.

21.2.3 Bypass Register (SDBPR)

The bypass register (SDBPR) is a one-bit register that cannot be accessed by the CPU. When bypass mode is set in SDIR, SDBPR is connected between the TDI pin and TDO pin of the H-UDI.

21.2.4 Interrupt Factor Register (SDINT)

The interrupt factor register (SDINT) is a 16-bit register that can be read/written from the CPU. When a (H-UDI interrupt) command is set in the SDIR (Update-IR) via the H-UDI pin, the INTREQ bit is set to 1. While SDIR has the “H-UDI interrupt” command, the SDINT register is connected between H-UDI pins TDI and TDO, and can be read as a 32-bit register. The high 16 bits are 0 and the low 16 bits are SDINT.

Only 0 can be written to the INTREQ bit from the CPU. While this bit is 1, the interrupt request continues to be generated, and must therefore be cleared to 0 by the interrupt handler. This register is initialized by $\overline{\text{TRST}}$ or when in the Test Logic Reset state.

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

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|---|---|--------|
| | — | — | — | — | — | — | — | INTREQ |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R/W |

Bits 15 to 1—Reserved: These bits always read as 0, and should only be written with 0.

Bit 0—Interrupt Request Bit (INTREQ): Shows the existence of an interrupt request from the “H-UDI interrupt” command. The interrupt request can be cleared by writing 0 to this bit from the CPU. When 1 is written to this bit, the existing value is retained.

21.2.5 Boundary Scan Register (SDBSR)

The boundary scan register (SDBSR) is a shift register that is placed on the pads to control the chip's I/O pins. This register can perform a boundary scan test equivalent to the JTAG (IEEE Std 1149.1) standard using EXTEST, SAMPLE, and PRELOAD commands. Table 21.3 shows the relationship between this LSI pins and the boundary scan register.

Table 21.3 Structure of Boundary Scan Register

| No. | Pin name | Type |
|--------|----------------------|------|
| to TDO | | |
| 418 | $\overline{CS0}$ | OUT |
| 417 | $\overline{CS0}$ | CTL |
| 416 | $\overline{CS1}$ | OUT |
| 415 | $\overline{CS1}$ | CTL |
| 414 | $\overline{CS4}$ | OUT |
| 413 | $\overline{CS4}$ | CTL |
| 412 | $\overline{CS5}$ | OUT |
| 411 | $\overline{CS5}$ | CTL |
| 410 | $\overline{CS6}$ | OUT |
| 409 | $\overline{CS6}$ | CTL |
| 408 | \overline{BS} | OUT |
| 407 | \overline{BS} | CTL |
| 406 | $\overline{WE0/REG}$ | OUT |
| 405 | $\overline{WE0/REG}$ | CTL |
| 404 | $\overline{WE1}$ | OUT |
| 403 | $\overline{WE1}$ | CTL |
| 402 | D0 | OUT |
| 401 | D0 | CTL |
| 400 | D0 | IN |
| 399 | D1 | OUT |
| 398 | D1 | CTL |
| 397 | D1 | IN |
| 396 | D2 | OUT |
| 395 | D2 | CTL |
| 394 | D2 | IN |
| 393 | D3 | OUT |
| 392 | D3 | CTL |
| 391 | D3 | IN |
| 390 | D4 | OUT |
| 389 | D4 | CTL |

| No. | Pin name | Type |
|-----|----------|------|
| 388 | D4 | IN |
| 387 | D5 | OUT |
| 386 | D5 | CTL |
| 385 | D5 | IN |
| 384 | D6 | OUT |
| 383 | D6 | CTL |
| 382 | D6 | IN |
| 381 | D7 | OUT |
| 380 | D7 | CTL |
| 379 | D7 | IN |
| 378 | D8 | OUT |
| 377 | D8 | CTL |
| 376 | D8 | IN |
| 375 | D9 | OUT |
| 374 | D9 | CTL |
| 373 | D9 | IN |
| 372 | D10 | OUT |
| 371 | D10 | CTL |
| 370 | D10 | IN |
| 369 | D11 | OUT |
| 368 | D11 | CTL |
| 367 | D11 | IN |
| 366 | D12 | OUT |
| 365 | D12 | CTL |
| 364 | D12 | IN |
| 363 | D13 | OUT |
| 362 | D13 | CTL |
| 361 | D13 | IN |
| 360 | D14 | OUT |
| 359 | D14 | CTL |
| 358 | D14 | IN |
| 357 | D15 | OUT |
| 356 | D15 | CTL |

| No. | Pin name | Type |
|-----|---|------|
| 355 | D15 | IN |
| 354 | $\overline{\text{CAS0}}/\text{DQM0}$ | OUT |
| 353 | $\overline{\text{CAS0}}/\text{DQM0}$ | CTL |
| 352 | $\overline{\text{CAS1}}/\text{DQM1}$ | OUT |
| 351 | $\overline{\text{CAS1}}/\text{DQM1}$ | CTL |
| 350 | $\text{RD}/\overline{\text{WR}}$ | OUT |
| 349 | $\text{RD}/\overline{\text{WR}}$ | CTL |
| 348 | $\overline{\text{RD}}/\text{CASS}/\text{FRAME}$ | OUT |
| 347 | $\overline{\text{RD}}/\text{CASS}/\text{FRAME}$ | CTL |
| 346 | CKE | OUT |
| 345 | CKE | CTL |
| 344 | $\overline{\text{RAS}}$ | OUT |
| 343 | $\overline{\text{RAS}}$ | CTL |
| 342 | $\overline{\text{CS2}}$ | OUT |
| 341 | $\overline{\text{CS2}}$ | CTL |
| 340 | $\overline{\text{CS3}}$ | OUT |
| 339 | $\overline{\text{CS3}}$ | CTL |
| 338 | A0 | OUT |
| 337 | A0 | CTL |
| 336 | A1 | OUT |
| 335 | A1 | CTL |
| 334 | A2 | OUT |
| 333 | A2 | CTL |
| 332 | A3 | OUT |
| 331 | A3 | CTL |
| 330 | A4 | OUT |
| 329 | A4 | CTL |
| 328 | A5 | OUT |
| 327 | A5 | CTL |
| 326 | A6 | OUT |
| 325 | A6 | CTL |
| 324 | A7 | OUT |
| 323 | A7 | CTL |

| No. | Pin name | Type |
|-----|-------------------------------|------|
| 322 | A8 | OUT |
| 321 | A8 | CTL |
| 320 | A9 | OUT |
| 319 | A9 | CTL |
| 318 | A10 | OUT |
| 317 | A10 | CTL |
| 316 | A11 | OUT |
| 315 | A11 | CTL |
| 314 | A12 | OUT |
| 313 | A12 | CTL |
| 312 | A13 | OUT |
| 311 | A13 | CTL |
| 310 | A14 | OUT |
| 309 | A14 | CTL |
| 308 | A15 | OUT |
| 307 | A15 | CTL |
| 306 | A16 | OUT |
| 305 | A16 | CTL |
| 304 | A17 | OUT |
| 303 | A17 | CTL |
| 302 | $\overline{\text{CAS2/DQM2}}$ | OUT |
| 301 | $\overline{\text{CAS2/DQM2}}$ | CTL |
| 300 | $\overline{\text{CAS3/DQM3}}$ | OUT |
| 299 | $\overline{\text{CAS3/DQM3}}$ | CTL |
| 298 | D16 | OUT |
| 297 | D16 | CTL |
| 296 | D16 | IN |
| 295 | D17 | OUT |
| 294 | D17 | CTL |
| 293 | D17 | IN |
| 292 | D18 | OUT |
| 291 | D18 | CTL |
| 290 | D18 | IN |

| No. | Pin name | Type |
|-----|----------|------|
| 289 | D19 | OUT |
| 288 | D19 | CTL |
| 287 | D19 | IN |
| 286 | D20 | OUT |
| 285 | D20 | CTL |
| 284 | D20 | IN |
| 283 | D21 | OUT |
| 282 | D21 | CTL |
| 281 | D21 | IN |
| 280 | D22 | OUT |
| 279 | D22 | CTL |
| 278 | D22 | IN |
| 277 | D23 | OUT |
| 276 | D23 | CTL |
| 275 | D23 | IN |
| 274 | D24 | OUT |
| 273 | D24 | CTL |
| 272 | D24 | IN |
| 271 | D25 | OUT |
| 270 | D25 | CTL |
| 269 | D25 | IN |
| 268 | D26 | OUT |
| 267 | D26 | CTL |
| 266 | D26 | IN |
| 265 | D27 | OUT |
| 264 | D27 | CTL |
| 263 | D27 | IN |
| 262 | D28 | OUT |
| 261 | D28 | CTL |
| 260 | D28 | IN |
| 259 | D29 | OUT |
| 258 | D29 | CTL |
| 257 | D29 | IN |

| No. | Pin name | Type |
|-----|--------------------------------|------|
| 256 | D30 | OUT |
| 255 | D30 | CTL |
| 254 | D30 | IN |
| 253 | D31 | OUT |
| 252 | D31 | CTL |
| 251 | D31 | IN |
| 250 | A18 | OUT |
| 249 | A18 | CTL |
| 248 | A19 | OUT |
| 247 | A19 | CTL |
| 246 | A20 | OUT |
| 245 | A20 | CTL |
| 244 | A21 | OUT |
| 243 | A21 | CTL |
| 242 | A22 | OUT |
| 241 | A22 | CTL |
| 240 | A23 | OUT |
| 239 | A23 | CTL |
| 238 | A24 | OUT |
| 237 | A24 | CTL |
| 236 | A25 | OUT |
| 235 | A25 | CTL |
| 234 | $\overline{\text{WE2/ICIORD}}$ | OUT |
| 233 | $\overline{\text{WE2/ICIORD}}$ | CTL |
| 232 | $\overline{\text{WE3/ICIOWR}}$ | OUT |
| 231 | $\overline{\text{WE3/ICIOWR}}$ | CTL |
| 230 | SLEEP | IN |
| 229 | $\overline{\text{PCIGNT4}}$ | OUT |
| 228 | $\overline{\text{PCIGNT4}}$ | CTL |
| 227 | $\overline{\text{PCIGNT3}}$ | OUT |
| 226 | $\overline{\text{PCIGNT3}}$ | CTL |
| 225 | $\overline{\text{PCIGNT2}}$ | OUT |
| 224 | $\overline{\text{PCIGNT2}}$ | CTL |

| No. | Pin name | Type |
|-----|------------------------------------|------|
| 223 | $\overline{\text{PCIREQ4}}$ | OUT |
| 222 | $\overline{\text{PCIREQ4}}$ | CTL |
| 221 | $\overline{\text{PCIREQ4}}$ | IN |
| 220 | $\overline{\text{PCIREQ3/MD10}}$ | OUT |
| 219 | $\overline{\text{PCIREQ3/MD10}}$ | CTL |
| 218 | $\overline{\text{PCIREQ3/MD10}}$ | IN |
| 217 | $\overline{\text{PCIREQ2/MD9}}$ | OUT |
| 216 | $\overline{\text{PCIREQ2/MD9}}$ | CTL |
| 215 | $\overline{\text{PCIREQ2/MD9}}$ | IN |
| 214 | IDSEL | IN |
| 213 | $\overline{\text{INTA}}$ | OUT |
| 212 | $\overline{\text{INTA}}$ | CTL |
| 211 | $\overline{\text{PCIRST}}$ | OUT |
| 210 | $\overline{\text{PCIRST}}$ | CTL |
| 209 | PCICLK | IN |
| 208 | $\overline{\text{PCIGNT1/REQOUT}}$ | OUT |
| 207 | $\overline{\text{PCIGNT1/REQOUT}}$ | CTL |
| 206 | $\overline{\text{PCIREQ1/GNTIN}}$ | OUT |
| 205 | $\overline{\text{PCIREQ1/GNTIN}}$ | CTL |
| 204 | $\overline{\text{PCIREQ1/GNTIN}}$ | IN |
| 203 | $\overline{\text{SERR}}$ | OUT |
| 202 | $\overline{\text{SERR}}$ | CTL |
| 201 | $\overline{\text{SERR}}$ | IN |
| 200 | AD31 | OUT |
| 199 | AD31 | CTL |
| 198 | AD31 | IN |
| 197 | AD30 | OUT |
| 196 | AD30 | CTL |
| 195 | AD30 | IN |
| 194 | AD29 | OUT |
| 193 | AD29 | CTL |
| 192 | AD29 | IN |
| 191 | AD28 | OUT |

| No. | Pin name | Type |
|-----|-----------------------------|------|
| 190 | AD28 | CTL |
| 189 | AD28 | IN |
| 188 | AD27 | OUT |
| 187 | AD27 | CTL |
| 186 | AD27 | IN |
| 185 | AD26 | OUT |
| 184 | AD26 | CTL |
| 183 | AD26 | IN |
| 182 | AD25 | OUT |
| 181 | AD25 | CTL |
| 180 | AD25 | IN |
| 179 | AD24 | OUT |
| 178 | AD24 | CTL |
| 177 | AD24 | IN |
| 176 | C/ $\overline{\text{BE}}_3$ | OUT |
| 175 | C/ $\overline{\text{BE}}_3$ | CTL |
| 174 | C/ $\overline{\text{BE}}_3$ | IN |
| 173 | AD23 | OUT |
| 172 | AD23 | CTL |
| 171 | AD23 | IN |
| 170 | AD22 | OUT |
| 169 | AD22 | CTL |
| 168 | AD22 | IN |
| 167 | AD21 | OUT |
| 166 | AD21 | CTL |
| 165 | AD21 | IN |
| 164 | AD20 | OUT |
| 163 | AD20 | CTL |
| 162 | AD20 | IN |
| 161 | AD19 | OUT |
| 160 | AD19 | CTL |
| 159 | AD19 | IN |
| 158 | AD18 | OUT |

| No. | Pin name | Type |
|-----|----------|------|
| 157 | AD18 | CTL |
| 156 | AD18 | IN |
| 155 | AD17 | OUT |
| 154 | AD17 | CTL |
| 153 | AD17 | IN |
| 152 | AD16 | OUT |
| 151 | AD16 | CTL |
| 150 | AD16 | IN |
| 149 | C/BE2 | OUT |
| 148 | C/BE2 | CTL |
| 147 | C/BE2 | IN |
| 146 | PCIFRAME | OUT |
| 145 | PCIFRAME | CTL |
| 144 | PCIFRAME | IN |
| 143 | IRDY | OUT |
| 142 | IRDY | CTL |
| 141 | IRDY | IN |
| 140 | TRDY | OUT |
| 139 | TRDY | CTL |
| 138 | TRDY | IN |
| 137 | DEVSEL | OUT |
| 136 | DEVSEL | CTL |
| 135 | DEVSEL | IN |
| 134 | PCISTOP | OUT |
| 133 | PCISTOP | CTL |
| 132 | PCISTOP | IN |
| 131 | PCILOCK | OUT |
| 130 | PCILOCK | CTL |
| 129 | PCILOCK | IN |
| 128 | PERR | OUT |
| 127 | PERR | CTL |
| 126 | PERR | IN |
| 125 | PAR | OUT |

| No. | Pin name | Type |
|-----|----------------|------|
| 124 | PAR | CTL |
| 123 | PAR | IN |
| 122 | C/BE $\bar{1}$ | OUT |
| 121 | C/BE $\bar{1}$ | CTL |
| 120 | C/BE $\bar{1}$ | IN |
| 119 | AD15 | OUT |
| 118 | AD15 | CTL |
| 117 | AD15 | IN |
| 116 | AD14 | OUT |
| 115 | AD14 | CTL |
| 114 | AD14 | IN |
| 113 | AD13 | OUT |
| 112 | AD13 | CTL |
| 111 | AD13 | IN |
| 110 | AD12 | OUT |
| 109 | AD12 | CTL |
| 108 | AD12 | IN |
| 107 | AD11 | OUT |
| 106 | AD11 | CTL |
| 105 | AD11 | IN |
| 104 | AD10 | OUT |
| 103 | AD10 | CTL |
| 102 | AD10 | IN |
| 101 | AD9 | OUT |
| 100 | AD9 | CTL |
| 99 | AD9 | IN |
| 98 | AD8 | OUT |
| 97 | AD8 | CTL |
| 96 | AD8 | IN |
| 95 | C/BE $\bar{0}$ | OUT |
| 94 | C/BE $\bar{0}$ | CTL |
| 93 | C/BE $\bar{0}$ | IN |
| 92 | AD7 | OUT |

| No. | Pin name | Type |
|-----|---------------------------------|------|
| 91 | AD7 | CTL |
| 90 | AD7 | IN |
| 89 | AD6 | OUT |
| 88 | AD6 | CTL |
| 87 | AD6 | IN |
| 86 | AD5 | OUT |
| 85 | AD5 | CTL |
| 84 | AD5 | IN |
| 83 | AD4 | OUT |
| 82 | AD4 | CTL |
| 81 | AD4 | IN |
| 80 | AD3 | OUT |
| 79 | AD3 | CTL |
| 78 | AD3 | IN |
| 77 | AD2 | OUT |
| 76 | AD2 | CTL |
| 75 | AD2 | IN |
| 74 | AD1 | OUT |
| 73 | AD1 | CTL |
| 72 | AD1 | IN |
| 71 | AD0 | OUT |
| 70 | AD0 | CTL |
| 69 | AD0 | IN |
| 68 | $\overline{\text{IRL0}}$ | IN |
| 67 | $\overline{\text{IRL1}}$ | IN |
| 66 | $\overline{\text{IRL2}}$ | IN |
| 65 | $\overline{\text{IRL3}}$ | IN |
| 64 | NMI | IN |
| 63 | $\overline{\text{BACK/BSREQ}}$ | OUT |
| 62 | $\overline{\text{BACK/BSREQ}}$ | CTL |
| 61 | $\overline{\text{BREQ/BSACK}}$ | IN |
| 60 | MD6/ $\overline{\text{IOIS16}}$ | IN |
| 59 | $\overline{\text{RDY}}$ | IN |

| No. | Pin name | Type |
|-----|----------|------|
| 58 | TXD | OUT |
| 57 | TXD | CTL |
| 56 | TXD | IN |
| 55 | MD2/RXD2 | IN |
| 54 | RXD | IN |
| 53 | TCLK | OUT |
| 52 | TCLK | CTL |
| 51 | TCLK | IN |
| 50 | RTS2/MD8 | OUT |
| 49 | RTS2/MD8 | CTL |
| 48 | RTS2/MD8 | IN |
| 47 | SCK | OUT |
| 46 | SCK | CTL |
| 45 | SCK | IN |
| 44 | MD1/TXD2 | OUT |
| 43 | MD1/TXD2 | CTL |
| 42 | MD1/TXD2 | IN |
| 41 | MD0/SCK2 | OUT |
| 40 | MD0/SCK2 | CTL |
| 39 | MD0/SCK2 | IN |
| 38 | MD7/CTS2 | OUT |
| 37 | MD7/CTS2 | CTL |
| 36 | MD7/CTS2 | IN |
| 35 | AUDSYNC | OUT |
| 34 | AUDSYNC | CTL |
| 33 | AUDCK | OUT |
| 32 | AUDCK | CTL |
| 31 | AUDATA0 | OUT |
| 30 | AUDATA0 | CTL |
| 29 | AUDATA1 | OUT |
| 28 | AUDATA1 | CTL |
| 27 | AUDATA2 | OUT |
| 26 | AUDATA2 | CTL |

| No. | Pin name | Type |
|-----|-------------------------------|------|
| 25 | AUDATA3 | OUT |
| 24 | AUDATA3 | CTL |
| 23 | MD3/ $\overline{\text{CE2A}}$ | OUT |
| 22 | MD3/ $\overline{\text{CE2A}}$ | CTL |
| 21 | MD3/ $\overline{\text{CE2A}}$ | IN |
| 20 | MD4/ $\overline{\text{CE2B}}$ | OUT |
| 19 | MD4/ $\overline{\text{CE2B}}$ | CTL |
| 18 | MD4/ $\overline{\text{CE2B}}$ | IN |
| 17 | MD5 | OUT |
| 16 | MD5 | CTL |
| 15 | MD5 | IN |
| 14 | DACK0 | OUT |
| 13 | DACK0 | CTL |
| 12 | DACK1 | OUT |
| 11 | DACK1 | CTL |
| 10 | DRAK0 | OUT |
| 9 | DRAK0 | CTL |
| 8 | DRAK1 | OUT |
| 7 | DRAK1 | CTL |
| 6 | STATUS0 | OUT |
| 5 | STATUS0 | CTL |
| 4 | STATUS1 | OUT |
| 3 | STATUS1 | CTL |
| 2 | $\overline{\text{DREQ0}}$ | IN |
| 1 | $\overline{\text{DREQ1}}$ | IN |

from TDI

Note: CTL is a low-active signal. The relevant pin is driven to the OUT state when CTL is set LOW.

21.3 Operation

21.3.1 TAP Control

Figure 21.2 shows the internal states of the TAP control circuit. These conform to the state transitions specified by JTAG.

- The transition condition is the TMS value at the rising edge of TCK.
- The TDI value is sampled at the rising edge of TCK, and shifted at the falling edge.
- The TDO value changes at the falling edge of TCK. When not in the Shift-DR or Shift-IR state, TDO is in the high-impedance state.
- In a transition to $\overline{\text{TRST}} = 0$, a transition is made to the Test-Logic-Reset state asynchronously with respect to TCK.

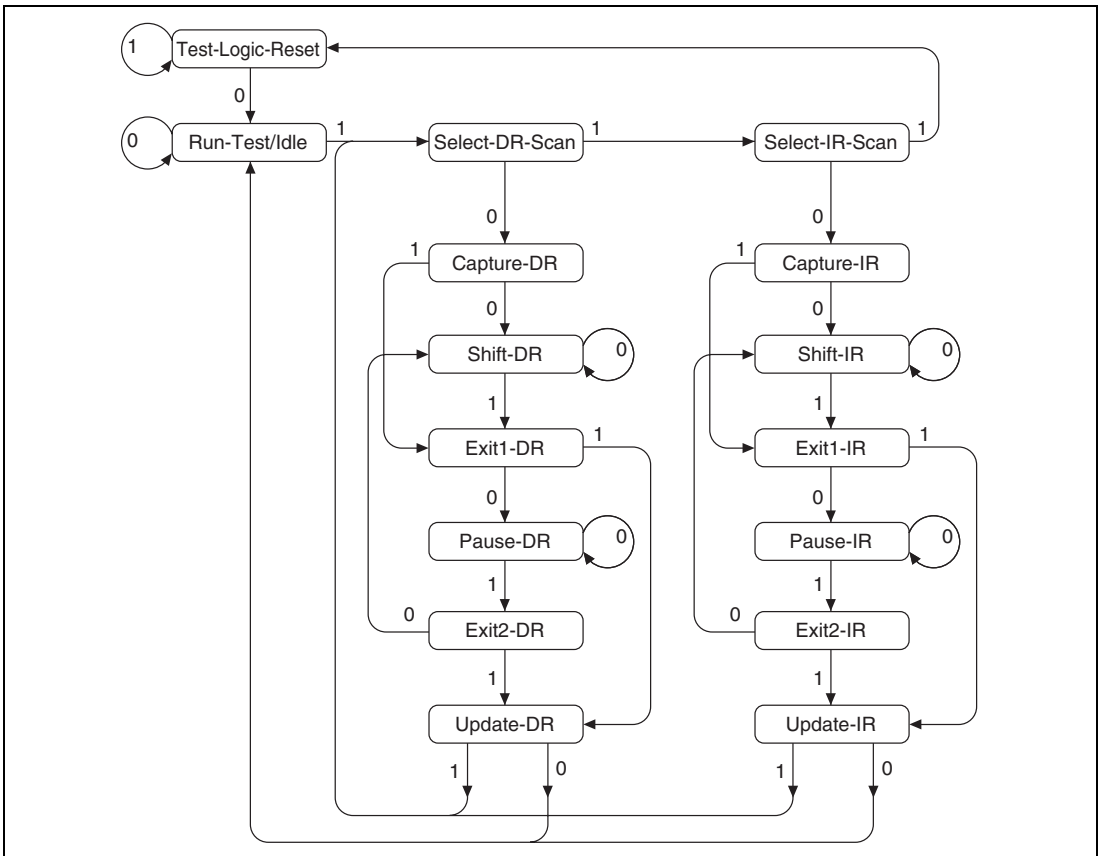


Figure 21.2 TAP Control State Transition Diagram

21.3.2 H-UDI Reset

A power-on reset is effected by an SDIR command. A reset is effected by sending a H-UDI reset assert command, and then sending a H-UDI reset negate command, from the H-UDI pin (see figure 21.3). The interval required between the H-UDI reset assert command and the H-UDI reset negate command is the same as the length of time the reset pin is held low in order to effect a power-on reset.

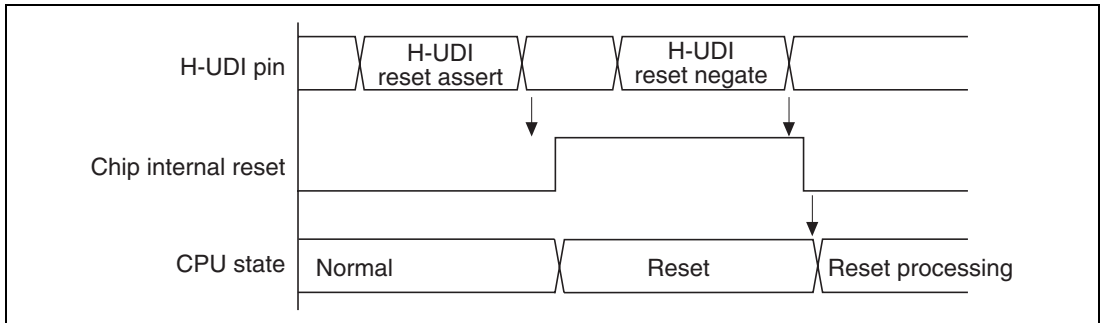


Figure 21.3 H-UDI Reset

21.3.3 H-UDI Interrupt

The H-UDI interrupt function generates an interrupt by setting a command value in SDIR from the H-UDI. The H-UDI interrupt is of general exception/interrupt operation type, with a branch to an address based on VBR and return effected by means of an RTE instruction. The exception code stored in control register INTEVT in this case is H'600. The priority of the H-UDI interrupt can be controlled with bits 3 to 0 of control register IPRC.

The H-UDI interrupt request signal is asserted when, after the command is set (Update-IR), the INTREQ bit of the SDINT register is set to 1. The interrupt request signal is not negated until 0 is written to the INTREQ bit by software, and there is therefore no risk of the interrupt request being unexpectedly missed. While the H-UDI interrupt command is set in SDIR, the SDINT register is connected between TDI and TDO.

21.3.4 Boundary Scan (EXTEST, SAMPLE/PRELOAD, BYPASS)

In this LSI, setting a command from the H-UDI in SDIR can place the H-UDI pins in the boundary scan mode. However, the following limitations apply.

1. Clock-related signals (EXTAL, EXTAL2, XTAL, XTAL2, and CKIO) are excluded from the boundary scan.
2. Reset-related signals ($\overline{\text{RESET}}$, $\overline{\text{MRESET}}$, and CA) are excluded from the boundary scan.
3. H-UDI signals (TCK, TDI, TDO, TMS, and $\overline{\text{TRST}}$) are excluded from the boundary scan.
4. With EXTEST, assert the $\overline{\text{MRESET}}$ pin (Low), negate the $\overline{\text{RESET}}$ pin (High), and assert the CA pin (High). With SAMPLE/PRELOAD, assert the CA pin (High).
5. When executing a boundary scan (EXTEST, SAMPLE/PRELOAD, and BYPASS), supply a clock signal to the EXTAL pin. The allowed range of input clock frequencies is from 1 to 33.3 MHz. Execute the boundary scan after t_{OSCI} (the power-on oscillation-stabilization time) has elapsed. The clock signal need not be supplied to the EXTAL pin after t_{OSCI} has elapsed. For details on t_{OSCI} (the power-on oscillation-stabilization time), see section 23, Electrical Characteristics.

21.4 Usage Notes

1. SDIR Command

Once an SDIR command is set, it does not change until another command is written from the H-UDI, unless initialized by asserting $\overline{\text{TRST}}$ or the TAP is set in the Test-Logic-Reset state.

2. SDIR Commands in Sleep Mode

Sleep mode is cleared by an H-UDI interrupt or H-UDI reset, and these exception requests are accepted in this mode. In standby mode, neither an H-UDI interrupt nor an H-UDI reset is accepted.

3. In standby mode, the H-UDI function cannot be used. Furthermore, TCK must be retained at a high level when entering the standby mode in order to retain the TAP state before and after standby mode.
4. The H-UDI is used for emulator connection. Therefore, H-UDI functions cannot be used when an emulator is used.
5. When the SH7751 is in bypass mode, the bypass register (SDBPR) is not fixed in the Capture-DR state. (It is cleared to 0 in the SH7751R.)

Section 22 PCI Controller (PCIC)

22.1 Overview

The PCI Controller (PCIC) controls the PCI bus and transfers data between memory connected to the external bus and a PCI device connected to the PCI bus. The ability for PCI devices to be connected directly not only facilitates the design of systems using PCI buses but also enables systems to be more compact and capable of high-speed data transfer.

22.1.1 Features

The PCIC has the following features:

- Supports a subset of PCI version 2.1.
- Compatible with PCI bus operating speeds of 33 MHz/66 MHz.
- Compatible with 32-bit PCI bus.
- Up to four PCI master devices running at 33 MHz or one PCI master device at 66 MHz can be connected.
- Arbitration control is available as a PCI host function.
- Can operate as master or target.
- When operating as master, PIO and DMA transfer are available.
- Four DMA transfer channels.
- Six 32-bit x 16 longword internal FIFO (one for target reading, one for target writing, and four for DMA transfer).
- Asynchronous operation of BSC bus clock and PCI bus clock available, and CKIO can be used as PCI bus clock.
- SRAM, DRAM, SDRAM, and MPX* can be used as external memory for PCI bus data transfers.
- 32-bit or 16-bit memory data bus for data transfers with PCI bus (32-bit bus when connected to SDRAM).
- Support for big endian and little endian local bus (PCI bus operates with little endian, while internal bus for peripheral modules operates with big endian).

Note: * MPX is only supported by the SH7751R and is not supported by the SH7751.

22.1.2 Block Diagram

Figure 22.1 is a block diagram of the PCIC.

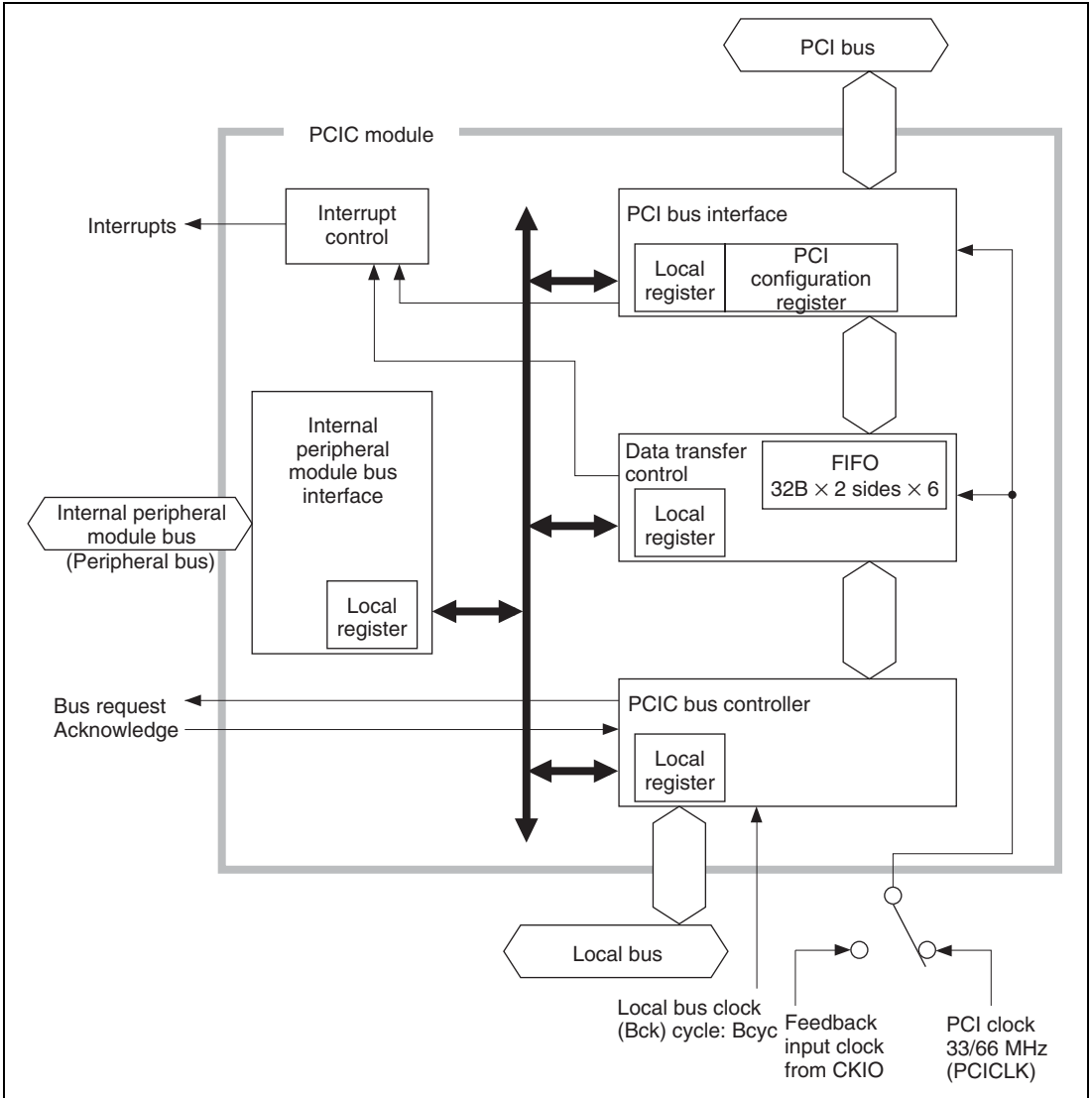


Figure 22.1 PCIC Block Diagram

22.1.3 Pin Configuration

Table 22.1 shows the configuration of I/O pins of the PCIC.

Table 22.1 Pin Configuration

| No. | Pin Name | PCI Standard Signal Name | Function | I/O Type | Pull-up Resistor ^{*)} | I/O Status in Operating Modes | | | | Remarks |
|-----|--------------------|--------------------------|---------------------------------|----------|--------------------------------|-------------------------------|--------|----------|--------|---------------------------|
| | | | | | | Host | | Non-host | | |
| | | | | | | Master | Target | Master | Target | |
| 1 | PCICLK | CLK | PCI input clock (33 MHz/66 MHz) | in | | I | I | I | I | |
| 2 | PCIRST | — | Reset output | out | | O | O | — | — | |
| 3 | AD31 to AD0 | AD[31:0] | Address/data | t/s | | I/O | I/O | I/O | I/O | Low level output at reset |
| 4 | C/BE3 to C/BE0 | C/BE[3:0] | Command/byte enable | t/s | | O | I | O | I | Low level output at reset |
| 5 | PAR | PAR | Parity | t/s | | I/O | I/O | I/O | I/O | Low level output at reset |
| 6 | PCIFRAME | FRAME | Bus cycle | s/t/s | Yes | O | I | O | I | |
| 7 | IRDY | IRDY | Initiator ready | s/t/s | Yes | O | I | O | I | |
| 8 | TRDY | TRDY | Target ready | s/t/s | Yes | I | O | I | O | |
| 9 | PCISTOP | STOP | Transaction stop | s/t/s | Yes | I | O | I | O | |
| 10 | PCILOCK | LOCK | Exclusive access control | s/t/s | Yes | O | I | O | I | |
| 11 | DEVSEL | DEVSEL | Device select | s/t/s | Yes | I | O | I | O | |
| 12 | PCIREQ1/ GNTIN | REQ1 | Bus request (host function) | t/s | Yes | I | I | — | — | |
| | | GNT | Bus grant | t/s | Yes | — | — | I | | |
| 13 | PCIGNT1/ REQOUT | GNT1 | Bus grant (host function) | t/s | No | O | O | — | — | |
| | | REQ | Bus request | t/s | No | — | — | O | | |
| 14 | PERR | PERR | Parity error | s/t/s | Yes | I/O | O | I/O | O | |
| 15 | SERR | SERR | System error | o/d | Yes | O | O | O | O | |
| 16 | INTA | INTA | Interrupt (async) | o/d | Yes | — | — | O | O | |

| No. | Pin Name | PCI Standard Signal Name | Function | I/O Type | Pull-up Resistor ^{*1} | I/O Status in Operating Modes | | | | Remarks |
|-----|-----------------------|--------------------------|--------------------------------|----------|--------------------------------|-------------------------------|--------|----------|--------|----------------|
| | | | | | | Host | | Non-host | | |
| | | | | | | Master | Target | Master | Target | |
| 17 | PCIREQ2/ MD9 | REQ2 | Bus request (host function) | t/s | Yes | | | — | — | |
| | | | PCI clock switch (BCLK/PCICLK) | in | | | | | | * ² |
| 18 | PCIREQ3/ MD10 | REQ3 | Bus request (host function) | t/s | Yes | | | — | — | |
| | | | Host bridge function ON/OFF | in | | | | | | * ² |
| 19 | PCIREQ4 | REQ4 | Bus request (host function) | t/s | Yes | | | — | — | |
| 20 | PCIGNT4 to PCIGNT2 | GNT4 to GNT2 | Bus grant (host function) | t/s | | O | O | — | — | |
| 21 | IDSEL | IDSEL | Config device select | in | | — | — | | | * ³ |

Legend:

in: Input

out: Output

s/t/s: Sustained try state

o/d: Open drain

t/s: Try state

Notes: 1. Terminal provided with a pull-up resistor.

2. The values of external pins are sampled in a power-on reset by means of the $\overline{\text{RESET}}$ pin.

3. Pull down this pin to low level when IDSEL is not in use. If a configuration access to an external PCI device occurs while IDSEL is high level, the PCIC itself may respond.

22.1.4 Register Configuration

The PCIC has the PCI configuration registers and PCI control registers shown in table 22.2, 22.3 and 22.4. Also, the PCI bus address space is allocated to the internal bus for the peripheral modules, making it possible to access the PCI bus by program IO (PIO). Not only do these registers control the PCI bus but also enable high-speed data transfers between the PCI device and memory on the SH-4 external data bus (hereinafter, the SH-4 external data bus is referred to as the local bus to distinguish it from the PCI bus).

Table 22.2 List of PCI Configuration Registers

| Name | Abbreviation | PCI R/W | PP-Bus R/W | Initial Value | PCI Configuration Address | P4 Address | Area 7 Address | Access Size |
|-------------------------------|--------------|------------------------|-------------------------|---------------|---------------------------|--------------------------------|--------------------------------|-------------|
| PCI configuration register 0 | PCICONF0 | R | R | *1 | H'00 | H'FE200000 | H'1E200000 | 32 |
| PCI configuration register 1 | PCICONF1 | R/W | R/W | H'02900080 | H'04 | H'FE200004 | H'1E200004 | 32 |
| PCI configuration register 2 | PCICONF2 | R | R/W[31:8] R (other) | H'xxxxxx*2 | H'08 | H'FE200008 | H'1E200008 | 32 |
| PCI configuration register 3 | PCICONF3 | R/W[15:8] R (other) | R/W[15:8] R (other) | H'00000000 | H'0C | H'FE20000C | H'1E20000C | 32 |
| PCI configuration register 4 | PCICONF4 | R/W | R/W | H'00000001 | H'10 | H'FE200010 | H'1E200010 | 32 |
| PCI configuration register 5 | PCICONF5 | R/W | R/W | H'00000000 | H'14 | H'FE200014 | H'1E200014 | 32 |
| PCI configuration register 6 | PCICONF6 | R/W | R/W | H'00000000 | H'18 | H'FE200018 | H'1E200018 | 32 |
| PCI configuration register 7 | PCICONF7 | R | R | H'00000000 | H'1C | H'FE20001C | H'1E20001C | 32 |
| PCI configuration register 8 | PCICONF8 | R | R | H'00000000 | H'20 | H'FE200020 | H'1E200020 | 32 |
| PCI configuration register 9 | PCICONF9 | R | R | H'00000000 | H'24 | H'FE200024 | H'1E200024 | 32 |
| PCI configuration register 10 | PCICONF10 | R | R | H'00000000 | H'28 | H'FE200028 | H'1E200028 | 32 |
| PCI configuration register 11 | PCICONF11 | R | R/W | H'xxxxxxxx | H'2C | H'FE20002C | H'1E20002C | 32 |
| PCI configuration register 12 | PCICONF12 | R | R | H'00000000 | H'30 | H'FE200030 | H'1E200030 | 32 |
| PCI configuration register 13 | PCICONF13 | R | R | H'00000040 | H'34 | H'FE200034 | H'1E200034 | 32 |
| PCI configuration register 14 | PCICONF14 | R | R | H'00000000 | H'38 | H'FE200038 | H'1E200038 | 32 |
| PCI configuration register 15 | PCICONF15 | R/W[7:0] R (other) | R/W[7:0] R (other) | H'0000100 | H'3C | H'FE20003C | H'1E20003C | 32 |
| PCI configuration register 16 | PCICONF16 | R R | R/W[18:16] R (other) | H'00010001 | H'40 | H'FE200040 | H'1E200040 | 32 |
| PCI configuration register 17 | PCICONF17 | R/W[1:0] R (other) | R/W[1:0] R (other) | H'00000000 | H'44 | H'FE200044 | H'1E200044 | 32 |
| Reserved | — | R | R | H'00000000 | H'48 to H'FC | H'FE200048 to H'FE2000FC | H'1E200048 to H'1E2000FC | 32 |

Legend: x: Undefined value

Notes: 1. Varies with the logic versions of the chip.

2. H'35051054 for the SH7751; H'350E1054 for the SH7751R.

Table 22.3 PCI Configuration Register Configuration

| PCI Configu- ration Address | P4 Address | Area 7 Address | PCI Configuration Register | | | | PCI R/W | PP-Bus R/W |
|--------------------------------------|--------------------------|--------------------------|-------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|------------------------|-------------------------|
| | | | 31 to 24 | 23 to 16 | 15 to 8 | 7 to 0 | | |
| H'00 | H'FE200000 | H'1E200000 | Device ID | Device ID | Vendor ID | Vendor ID | R | R |
| H'04 | H'FE200004 | H'1E200004 | Status | Status | Command | Command | R/W | R/W |
| H'08 | H'FE200008 | H'1E200008 | Class code | Class code | Class code | Revision ID | R | R/W[31:8] R (other) |
| H'0C | H'FE20000C | H'1E20000C | BIST | Header type | PCI latency timer | Cache line size | R/W[15:8] R (other) | R/W[15:8] R (other) |
| H'10 | H'FE200010 | H'1E200010 | Base address (I/O area) | Base address (I/O area) | Base address (I/O area) | Base address (I/O area) | R/W | R/W |
| H'14 | H'FE200014 | H'1E200014 | Base address (local address area 0) | Base address (local address area 0) | Base address (local address area 0) | Base address (local address area 0) | R/W | R/W |
| H'18 | H'FE200018 | H'1E200018 | Base address (local address area 1) | Base address (local address area 1) | Base address (local address area 1) | Base address (local address area 1) | R/W | R/W |
| H'1C | H'FE20001C | H'1E20001C | Reserved | Reserved | Reserved | Reserved | R | R |
| H'20 | H'FE200020 | H'1E200020 | Reserved | Reserved | Reserved | Reserved | R | R |
| H'24 | H'FE200024 | H'1E200024 | Reserved | Reserved | Reserved | Reserved | R | R |
| H'28 | H'FE200028 | H'1E200028 | Reserved | Reserved | Reserved | Reserved | R | R |
| H'2C | H'FE20002C | H'1E20002C | Subsystem ID | Subsystem ID | Subsystem vendor ID | Subsystem vendor ID | R | R/W |
| H'30 | H'FE200030 | H'1E200030 | Reserved | Reserved | Reserved | Reserved | R | R |
| H'34 | H'FE200034 | H'1E200034 | Reserved | Reserved | Reserved | Extended function pointer | R | R |
| H'38 | H'FE200038 | H'1E200038 | Reserved | Reserved | Reserved | Reserved | R | R |
| H'3C | H'FE20003C | H'1E20003C | Max_Lat | Min_Gnt | Interrupt pin | Interrupt line | R/W[7:0] R (other) | R/W[7:0] R (other) |
| H'40 | H'FE200040 | H'1E200040 | Power management related | Power management related | Power management related | Power management related | R | R/W[18:16] R (other) |
| H'44 | H'FE200044 | H'1E200044 | Power management related | Power management related | Power management related | Power management related | R/W[1:0] R (other) | R/W[1:0] R (other) |
| H'48 to H'0FC | H'FE200048 to H'FE2000FC | H'1E200048 to H'1E2000FC | Reserved | Reserved | Reserved | Reserved | R | R |

Table 22.4 List of PCIC Local Registers

| Name | Abbreviation | PCI R/W | PP-Bus R/W | Initial Value | PCI I/O Address (SH7751/SH7751R) | P4 Address | Area 7 Address | Access Size |
|--|--------------|---------|------------|---------------|---------------------------------------|--------------------------------|--------------------------------|-------------|
| PCI control register | PCICR | R | R/W | H'000000*0 | H'100/ H'00 | H'FE200100 | H'1E200100 | 32 |
| Local space register 0 for PCI | PCILSRO | R | R/W | H'00000000 | H'104/ H'04 | H'FE200104 | H'1E200104 | 32 |
| Local space register 1 for PCI | PCILSR1 | R | R/W | H'00000000 | H'108/ H'08 | H'FE200108 | H'1E200108 | 32 |
| Local address register 0 for PCI | PCILAR0 | R/W | R/W | H'00000000 | H'10C/ H'0C | H'FE20010C | H'1E20010C | 32 |
| Local address register 1 for PCI | PCILAR1 | R/W | R/W | H'00000000 | H'110/ H'10 | H'FE200110 | H'1E210110 | 32 |
| PCI interrupt register | PCIINT | R/W | R/W | H'00000000 | H'114/ H'14 | H'FE200114 | H'1E200114 | 32 |
| PCI interrupt mask register | PCIINTM | R/W | R/W | H'00000000 | H'118/ H'18 | H'FE200118 | H'1E200118 | 32 |
| Error address data register for PCI | PCIALR | R | R | H'xxxxxxx | H'11C/ H'1C | H'FE20011C | H'1E20011C | 32 |
| Error command data register for PCI | PCICLR | R | R | H'0000000x | H'120/ H'20 | H'FE200120 | H'1E200120 | 32 |
| Reserved | — | — | — | H'00000000 | H'124 to H'12C/ H'24 to H'2C | H'FE200124 to H'FE20012C | H'1E200124 to H'1E20012C | 32 |
| PCI arbiter interrupt register | PCIAINT | R/W | R/W | H'00000000 | H'130/ H'30 | H'FE200130 | H'1E200130 | 32 |
| PCI arbiter interrupt mask register | PCIAINTM | R/W | R/W | H'00000000 | H'134/ H'34 | H'FE200134 | H'1E200134 | 32 |
| Error bus master data register for PCI | PCIBMLR | R | R | H'00000000 | H'138/ H'38 | H'FE200138 | H'1E200138 | 32 |
| Reserved | — | — | — | H'00000000 | H'13C/ H'3C | H'FE20013C | H'1E20013C | 32 |
| DMA transfer arbitration register for PCI | PCIDMABT | R/W | R/W | H'00000000 | H'140/ H'40 | H'FE200140 | H'1E200140 | 32 |
| Reserved | — | — | — | H'00000000 | H'144 to H'17C/ H'44 to H'7C | H'FE200144 to H'FE20017C | H'1E200144 to H'1E20017C | 32 |
| DMA transfer PCI address register 0 for PCI | PCIDPA0 | R/W | R/W | H'00000000 | H'180/ H'80 | H'FE200180 | H'1E200180 | 32 |
| DMA transfer local bus starting address register 0 for PCI | PCIDLA0 | R/W | R/W | H'00000000 | H'184/ H'84 | H'FE200184 | H'1E200184 | 32 |

| Name | Abbreviation | PCI R/W | PP-Bus R/W | Initial Value | PCI I/O Address (SH7751/SH7751R) | P4 Address | Area 7 Address | Access Size |
|--|--------------|---------|------------|---------------|----------------------------------|------------|----------------|-------------|
| DMA transfer count register 0 for PCI | PCIDTC0 | R/W | R/W | H'00000000 | H'188/ H'88 | H'FE200188 | H'1E200188 | 32 |
| DMA control register 0 for PCI | PCIDCR0 | R/W | R/W | H'00000000 | H'18C/ H'8C | H'FE20018C | H'1E20018C | 32 |
| DMA PCI address register 1 for PCI | PCIDPA1 | R/W | R/W | H'00000000 | H'190/ H'90 | H'FE200190 | H'1E200190 | 32 |
| DMA transfer local bus starting address register 1 for PCI | PCIDLA1 | R/W | R/W | H'00000000 | H'194/ H'94 | H'FE200194 | H'1E200194 | 32 |
| DMA transfer count register 1 for PCI | PCIDTC1 | R/W | R/W | H'00000000 | H'198/ H'98 | H'FE200198 | H'1E200198 | 32 |
| DMA control register 1 for PCI | PCIDCR1 | R/W | R/W | H'00000000 | H'19C/ H'9C | H'FE20019C | H'1E20019C | 32 |
| DMA transfer PCI address register 2 for PCI | PCIDPA2 | R/W | R/W | H'00000000 | H'1A0/ H'A0 | H'FE2001A0 | H'1E2001A0 | 32 |
| DMA transfer local bus starting address register 2 for PCI | PCIDLA2 | R/W | R/W | H'00000000 | H'1A4/ H'A4 | H'FE2001A4 | H'1E2001A4 | 32 |
| DMA transfer count register 2 for PCI | PCIDTC2 | R/W | R/W | H'00000000 | H'1A8/ H'A8 | H'FE2001A8 | H'1E2001A8 | 32 |
| DMA control register 2 for PCI | PCIDCR2 | R/W | R/W | H'00000000 | H'1AC/ H'AC | H'FE2001AC | H'1E2001AC | 32 |
| DMA transfer PCI address register 3 for PCI | PCIDPA3 | R/W | R/W | H'00000000 | H'1B0/ H'B0 | H'FE2001B0 | H'1E2001B0 | 32 |
| DMA transfer local bus starting address register 3 for PCI | PCIDLA3 | R/W | R/W | H'00000000 | H'1B4/ H'B4 | H'FE2001B4 | H'1E2001B4 | 32 |
| DMA transfer count register 3 for PCI | PCIDTC3 | R/W | R/W | H'00000000 | H'1B8/ H'B8 | H'FE2001B8 | H'1E2001B8 | 32 |
| DMA control register 3 for PCI | PCIDCR3 | R/W | R/W | H'00000000 | H'1BC/ H'BC | H'FE2001BC | H'1E2001BC | 32 |
| PIO address register | PCIPAR | — | R/W | H'80xxxxxx | — | H'FE2001C0 | H'1E2001C0 | 32 |
| Memory space base register | PCIMBR | — | R/W | H'xx000000 | — | H'FE2001C4 | H'1E2001C4 | 32 |
| IO space base register | PCIOBR | — | R/W | H'xxxx0000 | — | H'FE2001C8 | H'1E2001C8 | 32 |
| PCI power management interrupt register | PCIPINT | — | R/W | H'00000000 | — | H'FE2001CC | H'1E2001CC | 32 |
| PCI power management interrupt mask register | PCIPINTM | — | R/W | H'00000000 | — | H'FE2001D0 | H'1E2001D0 | 32 |
| PCI clock control register | PCICLKR | — | R/W | H'00000000 | — | H'FE2001D4 | H'1E2001D4 | 32 |

| Name | Abbreviation | PCI R/W | PP-Bus R/W | Initial Value | PCI I/O Address (SH7751/SH7751R) | P4 Address | Area 7 Address | Access Size |
|---|--------------|---------|------------|---------------|----------------------------------|--------------------------|--------------------------|-------------|
| Reserved | — | | | H'00000000 | — | H'FE2001D8 to H'FE2001DC | H'1E2001D8 to H'1E2001DC | 32 |
| PCI bus control register 1 | PCIBCR1 | — | R/W | H'*0000000 | — | H'FE2001E0 | H'1E2001E0 | 32 |
| PCI bus control register 2 | PCIBCR2 | — | R/W | H'0000*FFC | — | H'FE2001E4 | H'1E2001E4 | 32 |
| PCI wait control register 1 | PCIWCR1 | — | R/W | H'77777777 | — | H'FE2001E8 | H'1E2001E8 | 32 |
| PCI wait control register 2 | PCIWCR2 | — | R/W | H'FFFFFFF | — | H'FE2001EC | H'1E2001EC | 32 |
| PCI wait control register 3 | PCIWCR3 | — | R/W | H'07777777 | — | H'FE2001F0 | H'1E2001F0 | 32 |
| PCIC discrete memory control register | PCIMCR | — | R/W | H'00000000 | — | H'FE2001F4 | H'1E2001F4 | 32 |
| PCIC bus control register 3* ¹ | PCIBCR3 | — | R/W | H'00000001 | — | H'FE2001F8 | H'1E2001F8 | 32 |
| Reserved | — | | | H'00000000 | | H'FE2001FC | H'1E2001FC | 32 |
| Port control register | PCIPCTR | — | R/W | H'00000000 | — | H'FE200200 | H'1E200200 | 32 |
| Port data register | PCIPDTR | — | R/W | H'00000000 | — | H'FE200204 | H'1E200204 | 32 |
| Reserved | — | | | H'00000000 | — | H'FE200208 to H'FE20021C | H'1E200208 to H'1E20021C | 32 |
| PIO data register | PCIPDR | — | R/W | H'xxxxxxx | — | H'FE200220 | H'1E200220 | 32 |

Notes: * The values of some external pins are sampled in a power-on reset by means of the RESET pin.

x indicates “undefined.”

1. PCIC bus control register 3 is provided only in the SH7751R. The relevant areas of the SH7751 are reserved areas.

22.2 PCIC Register Descriptions

22.2.1 PCI Configuration Register 0 (PCICONF0)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|---------|---------|---------|---------|---------|---------|--------|--------|
| | DEVID15 | DEVID14 | DEVID13 | DEVID12 | DEVID11 | DEVID10 | DEVID9 | DEVID8 |
| Initial value: | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | DEVID7 | DEVID6 | DEVID5 | DEVID4 | DEVID3 | DEVID2 | DEVID1 | DEVID0 |
| Initial value: | 0 | 0 | 0 | 0 | 0/1* | 1 | 0/1* | 1/0* |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | VNDID15 | VNDID14 | VNDID13 | VNDID12 | VNDID11 | VNDID10 | VNDID9 | VNDID8 |
| Initial value: | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | VNDID7 | VNDID6 | VNDID5 | VNDID4 | VNDID3 | VNDID2 | VNDID1 | VNDID0 |
| Initial value: | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Note: * These values differ between SH7751 and SH7751R.

PCI configuration register 0 (PCICONF0) is a 32-bit read-only register that includes the device ID and vendor ID PCI configuration registers stipulated in the PCI local bus specifications. The SH7751 ID (H'3505) or the SH7751R ID (H'350E) is read from bits 31 to 16; the vendor ID (H'1054*) is read from bits 15 to 0.

All bits of the PCICONF0 are fixed in hardware.

Bits 31 to 16—DEVID15 to 0: These bits specify the device ID of the SH7751 or SH7751R allocated by the PCI device vendor. H'3505 (fixed in hardware) for the SH7751, and H'350E (fixed in hardware) for the SH7751R.

Bits 15 to 0—DNVID15 to 0: These bits specify the PCI device maker (vendor ID). (H'1054*: fixed in hardware)

Note: * The vendor ID H'1054 specifies Hitachi, Ltd., but the SH7751 and SH7751R are now products of Renesas Electronics Corp. For information on these products, contact Renesas Electronics Corp.

22.2.2 PCI Configuration Register 1 (PCICONF1)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|------|------|------|------|------|------|------|------|
| | DPE | SSE | RMA | RTA | STA | DEV1 | DEV0 | DPD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| PCI-R/W: | R/WC | R/WC | R/WC | R/WC | R/WC | R | R | R/WC |
| PP Bus-R/W: | R/WC | R/WC | R/WC | R/WC | R/WC | R | R | R/WC |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | FBBC | UDF | 66M | PM | — | — | — | — |
| Initial value: | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R/W | R/W | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | PBBE | SER |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R/W |
| PP Bus-R/W: | R | R | R | R | R | R | R | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | WCC | PER | VPS | MWIE | SPC | BUM | MES | IOS |
| Initial value: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R | R | R | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R | R | R | R/W | R/W | R/W |

Note: Cleared by writing WC: 1. (Writing of 0 is ignored.)

PCI configuration register 1 (PCICONF1) is a 32-bit read/partial-write register that includes the status and command PCI configuration registers stipulated in the PCI local bus specifications. The status is read from bits 31 to 16 (status register) in the event of an error on the PCI bus. Bits 15 to 0 (command register) contain the settings required for initiating transfers on the PCI bus.

Bits 31 to 27, 24, 8 to 6, and 2 to 0 can be written to from both the PP and PCI buses. However, bits 31 to 27 and 24 are write-clear bits that are cleared when 1 is written to them. Bits 22 and 21 can be written to from the PP bus. Other bits are fixed in hardware.

The PCICONF1 register is initialized to H'02900080 at a power-on reset or software reset.

Always write to this register before initiating transfers on the PCI bus.

Bit 31—Parity Error Detection Status (DPE): Indicates the detection of a parity error in read data when the PCIC is operating as the master, or a parity error in write data when the PCIC is operating as a target.

| Bit 31: DPE | Description |
|-------------|---|
| 0 | No parity error detected by device (Initial value) |
| 1 | Parity error detected by device Set this bit regardless of the parity error response bit (bit 6) on the device |

Bit 30—System Error Output Status (SSE): Indicates the $\overline{\text{SERR}}$ assert operation of the PCIC.

| Bit 30: SSE | Description |
|-------------|--|
| 0 | Device not asserting $\overline{\text{SERR}}$ (Initial value) |
| 1 | Device asserting $\overline{\text{SERR}}$ (Value retained until cleared) |

Bit 29—Master abort receive status (RMA): Indicates the termination of transaction by master abort when the PCIC is operating as the master.

| Bit 29: RMA | Description |
|-------------|---|
| 0 | No transaction termination using bus master abort (Initial value) |
| 1 | Detection by bus master of transaction termination by bus master abort However, in the case of a master abort in a special cycle, notify the master devices that are not set |

Bit 28—Target Abort Receive Status (RTA): Indicates the termination of transaction by master abort when the PCIC is operating as the master.

| Bit 28: RTA | Description |
|-------------|--|
| 0 | No transaction termination using target abort (Initial value) |
| 1 | Detection by bus master of transaction termination by target abort |

Bit 27—Target Abort Execution Status (STA): Indicates the termination of transaction by target abort when the PCIC is operating as the target.

| Bit 27: STA | Description |
|-------------|---|
| 0 | No transaction termination using target abort by target device (Initial value) |
| 1 | Transaction termination by target abort by target device. Notification by target device |

Bits 26 and 25— $\overline{\text{DEVSEL}}$ Timing Status (DEV1 and 0): These bits indicate the $\overline{\text{DEVSEL}}$ response timing when the PCIC is operating as a target.

| Bit 26: DEV1 | Bit 25: DEV0 | Description |
|--------------|--------------|------------------------------|
| 0 | 0 | High-speed (not supported) |
| | 1 | Medium speed (Initial value) |
| 1 | 0 | Low speed (not supported) |
| | 1 | Reserved |

Bit 24—Data Parity Status (DPD): Indicates the $\overline{\text{PERR}}$ assert operation or the detection of $\overline{\text{PERR}}$ when the PCIC is operating as the master. This bit is set only when the parity error response bit (bit 6) is 1.

| Bit 24: DPD | Description |
|-------------|--|
| 0 | Data parity not detected (Initial value) |
| 1 | Data parity occurred |

Bit 23—High-Speed Back-To-Back Status (FBBC): Shows whether a high-speed back-to-back transfer to a different target can be accepted when the PCIC is operating as a target.

| Bit 23: FBBC | Description |
|--------------|--|
| 0 | The target does not have a high-speed back-to-back transaction function for use with other targets |
| 1 | The target has a high-speed back-to-back transaction function for use with other targets (Initial value) |

Bit 22—User Defined Function System (UDF): Shows whether user defined functions are supported.

| Bit 22: UDF | Description |
|-------------|---|
| 0 | This device does not support user functions (Initial value) |
| 1 | This device supports user functions |

Bit 21—66 MHz Operating Status (66M): Shows whether 66 MHz operation is supported.

| Bit 21: 66M | Description |
|-------------|---|
| 0 | This device supports 33 MHz operation (Initial value) |
| 1 | This device supports 66 MHz operation |

Bit 20—PCI Power Management (PM): Shows whether the PCI power management is supported.

| Bit 20: PM | Description |
|------------|--|
| 0 | Power management not supported |
| 1 | Power management supported (Initial value) |

Bits 19 to 10—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 9—High-Speed Back-To-Back Control (PBBE): Selects whether or not to allow high-speed back-to-back control with different targets when privileged as the master.

| Bit 9: PBBE | Description |
|-------------|--|
| 0 | Allows high-speed back-to-back control only with same target (Initial value) |
| 1 | Allows high-speed back-to-back control with different target (Not supported) |

Bit 8— $\overline{\text{SERR}}$ Output Control (SER): Controls the $\overline{\text{SERR}}$ output.

| Bit 8: SER | Description |
|------------|---|
| 0 | $\overline{\text{SERR}}$ output disabled (Hi-Z) (Initial value) |
| 1 | $\overline{\text{SERR}}$ output enabled |

Bit 7—Wait Cycle Control (WCC): Controls the address/data stepping. When WCC=1, address and data are output in master write operations, only address is output in master read operations, and only data is output in target read operations, at least in two clocks.

| Bit 7: WCC | Description |
|------------|--|
| 0 | Disable address/data stepping control |
| 1 | Enable address/data stepping control (Initial value) |

Bit 6—Parity Error Response (PER): Controls the device response when a parity error is detected or a parity error report is received. $\overline{\text{PERR}}$ is asserted only when PER = 1.

| Bit 6: PER | Description |
|------------|---|
| 0 | Ignore detected parity errors (Initial value) |
| 1 | Respond to detected parity error |

Bit 5—VGA Pallet Snoop Control (VPS)

| Bit 5: VPS | Description |
|------------|---|
| 0 | VGA-compatible device (Initial value) |
| 1 | The device does not respond to pallet register writes (not supported) |

Bit 4—Memory Write and Invalidate Control (MWIE): Controls the issuance of memory and invalidate command when the PCIC is operating as the master.

| Bit 4: MWIE | Description |
|-------------|---|
| 0 | The device uses memory write (Initial value) |
| 1 | The device can execute memory write and invalidate commands (not supported) |

Bit 3—Special Cycle Control (SPC): Shows whether special cycles are supported when the PCIC is operating as a target.

| Bit 3: SPC | Description | |
|------------|---------------------------------------|-----------------|
| 0 | Ignore special cycle | (Initial value) |
| 1 | Monitor special cycle (not supported) | |

Bit 2—PCI Bus Master Control (BUM): Controls the bus master operation.

| Bit 2: BUM | Description | |
|------------|------------------------------|-----------------|
| 0 | Disable bus master operation | (Initial value) |
| 1 | Enable bus master operation | |

Bit 1—Memory Space Control (MES): Controls the access to the memory space when the PCIC is operating as a target. When this bit is 0, all memory transfers to the PCIC are terminated by master abort.

| Bit 1: MES | Description | |
|------------|--------------------------------|-----------------|
| 0 | Disable access to memory space | (Initial value) |
| 1 | Enable access to memory space | |

Bit 0—I/O Space Control (IOS): Controls the access to the I/O space when the PCIC is operating as a target. When this bit is 0, all I/O transfers to the PCIC are terminated by master abort.

| Bit 0: IOS | Description | |
|------------|-----------------------------|-----------------|
| 0 | Disable access to I/O space | (Initial value) |
| 1 | Enable access to I/O space | |

22.2.3 PCI Configuration Register 2 (PCICONF2)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| | CLASS23 | CLASS22 | CLASS21 | CLASS20 | CLASS19 | CLASS18 | CLASS17 | CLASS16 |
| Initial value: | — | — | — | — | — | — | — | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | CLASS15 | CLASS14 | CLASS13 | CLASS12 | CLASS11 | CLASS10 | CLASS9 | CLASS8 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | CLASS7 | CLASS6 | CLASS5 | CLASS4 | CLASS3 | CLASS2 | CLASS1 | CLASS0 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-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 |
| | REVID7 | REVID6 | REVID5 | REVID4 | REVID3 | REVID2 | REVID1 | REVID0 |
| Initial value: | * | * | * | * | * | * | * | * |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Note: * Initial values vary with the logic versions of the chip.

The PCI configuration register 2 (PCICONF2) is a 32-bit read/partial-write register that includes the class code and revision ID PCI configuration registers stipulated in the PCI local bus specifications. Bits 31 to 8 (class code) set the device functions. The chip logic version can be read from bits 7 to 0 (revision ID).

Bits 31 to 8 can be written to from the PP bus. Bits 7 to 0 are fixed in hardware.

The PCICONF2 register class codes are not initialized at a reset. They must be initialized while CFINIT (bit 0) of the PCI control registers (PCICR) is cleared.

Bits 31 to 24—Base Class Code (CLASS23 to 16): These bits indicate the base class code. For details of setting values, refer to table 22.5.

Table 22.5 List of CLASS23 to 16 Base Class Codes (CLASS23 to 16)

| CLASS23 to 16 Base Class | Meaning |
|-------------------------------------|---|
| H'00 | Device designed prior to class code being defined |
| H'01 | High-capacity storage controller |
| H'02 | Network controller |
| H'03 | Display controller |
| H'04 | Multimedia device |
| H'05 | Memory controller |
| H'06 | Bridge device |
| H'07 | Simple communication device |
| H'08 | Basic peripheral device |
| H'09 | Input device |
| H'0A | Docking station |
| H'0B | Processor |
| H'0C | Serial bus controller |
| H'0D to H'FE | Reserved |
| H'FF | Device not categorized in defined class |

Bits 23 to 16—Sub Class Codes (CLASS15 to 8): Shows the subclass code. For details, please see appendix D, Pin Functions of the PCI Local Bus Specifications, Revision 2.1.

Bits 15 to 8—Register Level Programming Interface (CLASS7 to 0): Shows the register level programming interface. For details, please see appendix D, Pin Functions of the PCI Local Bus Specifications, Revision 2.1.

Bits 7 to 0—Revision ID (REVID7 to 0): Shows the PCIC revision. The initial value differs according to the logic version of the chip.

22.2.4 PCI Configuration Register 3 (PCICONF3)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| | BIST7 | BIST6 | BIST5 | BIST4 | BIST3 | BIST2 | BIST1 | BIST0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | HEAD7 | HEAD6 | HEAD5 | HEAD4 | HEAD3 | HEAD2 | HEAD1 | HEAD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | LAT7 | LAT6 | LAT5 | LAT4 | LAT3 | LAT2 | LAT1 | LAT0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-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 |
| | CACHE7 | CACHE6 | CACHE5 | CACHE4 | CACHE3 | CACHE2 | CACHE1 | CACHE0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI configuration register 3 (PCICONF3) is a 32-bit read/partial-write register that includes the BIST function, header type, latency timer, and cache line size PCI configuration registers stipulated in the PCI local bus specification. The BIST function is read from bits 31 to 24, the header type from bits 23 to 16, the cache line size from bits 7 to 0. The guaranteed time for the PCIC to occupy the PCI bus when the PCIC is master is set in bits 15-8 (latency timer).

Bits 15 to 8 can be written to. Other bits are fixed in hardware.

The PCICONF3 register is initialized to H'00000000 at a power-on reset and software reset.

Bit 31—BIST7: BIST function support

| Bit 31: BIST7 | Description | |
|---------------|------------------------------------|-----------------|
| 0 | Function not supported | (Initial value) |
| 1 | Function supported (not supported) | |

Bit 30—BIST6: Used to control the BIST starting.

| Bit 30: BIST6 | Description | |
|---------------|---------------------------|-----------------|
| 0 | Execution completed | (Initial value) |
| 1 | Executing (not supported) | |

Bits 29 and 28—BIST5 and 4: These bits always return 0 when read.**Bits 27 to 24—BIST3 to 0:** BIST status on completion of operation.

| Bits 27 to 24: BIST3 to 0 | Description | |
|------------------------------|-----------------------------|-----------------|
| H'0 | Passed test | (Initial value) |
| H'1 to H'F | Test failed (not supported) | |

Bit 23—Multifunction Status (HEAD7): Shows whether the device is a multi-function unit or a single-function unit.

| Bit 23: HEAD7 | Description | |
|---------------|--|-----------------|
| 0 | Single-function device | (Initial value) |
| 1 | Device has between 2 and 8 functions (not supported) | |

Bits 22 to 16—Configuration Layout Type (HEAD6 to 0): These bits indicate the layout type of the configuration register.

| Bits 22 to 16: HEAD6 to 0 | Description | |
|------------------------------|---|-----------------|
| H'00 | Type 00h layout supported | (Initial value) |
| H'01 | Type 01h layout supported (not supported) | |
| H'02 to H'3F | Reserved | |

Bits 15 to 8—Latency Timer Register (LAT7 to 0): These bits specify the latency time of the PCI bus when the PCIC is operating as the master.

Bits 7 to 0—Cache Line Size (CACHE7 to 0): Not supported. Memory target is set cache-disabled, and SDONE and $\overline{\text{SBO}}$ are ignored.

22.2.5 PCI Configuration Register 4 (PCICONF4)

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | BASE31 | BASE30 | BASE29 | BASE28 | BASE27 | BASE26 | BASE25 | BASE24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BASE23 | BASE22 | BASE21 | BASE20 | BASE19 | BASE18 | BASE17 | BASE16 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W* | R/W* | R/W* | R/W* |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W* | R/W* | R/W* | R/W* |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | BASE15 | BASE14 | BASE13 | BASE12 | BASE11 | BASE10 | BASE9 | BASE8 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W* | R/W* | R/W* | R/W* | R/W* | R/W* | R/W* | R/W* |
| PP Bus-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 |
| | BASE7 | BASE6 | BASE5 | BASE4 | BASE3 | BASE2 | — | ASI |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Note: * These bits are read-only in the SH7751 and can be read from and written to in the SH7751R.

PCI configuration register 4 (PCICONF4) is a 32-bit read/partial-write register that accommodates the I/O-space base address register, which is one of the PCI configuration registers that are

stipulated in the PCI's local-bus specifications. PCICONF4 holds the higher-order bits of the address used when a device on the PCI bus uses I/O transfer commands to access a local register in the PCIC.

In the SH7751, the 12 higher-order bits (bits 31 to 8) are set; in the SH7751R, the 24 higher-order bits are set. As the I/O space for the PCI bus, allocate 1 Mbyte of space for the SH7751 and 256 bytes of space for the SH7751R.

In the SH7751, bits 30 to 20 are writable, and bits 19 to 2 and 0 are fixed by the hardware.

In the SH7751R, bits 31 to 8 are writable, and bits 7 to 2 and 0 are fixed by the hardware.

The PCICONF4 register is initialized to H'00000001 at a power-on reset and software reset.

Always write to this register prior to executing I/O transfers (accessing the local registers in the PCIC) to or from the PCIC from the PCI bus.

Bits 31 to 8—Base Address of the I/O Space (BASE 31 to 8): Sets the base address of the local registers (I/O space) in the PCIC. In the SH7751, bits 19 to 8 are fixed to H'000 in hardware.

Bits 7 to 2—Base Address of the I/O Space (BASE 7 to 2): Fixed at H'00 in hardware.

Bit 1—Reserved: This bit always returns 0 when read. Always write 0 to this bit.

Bit 0—Address Space Indicator (ASI): Shows whether the base address specified by this register is an I/O space or memory space.

| Bit 0: ASI | Description |
|------------|---------------------------|
| 0 | Memory space |
| 1 | I/O space (Initial value) |

22.2.6 PCI Configuration Register 5 (PCICONF5)

| | | | | | | | | |
|----------------|---------|---------|---------|---------|---------|----------|----------|---------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | BASE031 | BASE030 | BASE029 | BASE028 | BASE027 | BASE026 | BASE025 | BASE024 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BASE023 | BASE022 | BASE021 | BASE020 | BASE019 | BASE018 | BASE017 | BASE016 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | BASE015 | BASE014 | BASE013 | BASE012 | BASE011 | BASE010 | BASE09 | BASE08 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | BASE07 | BASE06 | BASE05 | BASE04 | LA0PREF | LA0TYPE1 | LA0TYPE0 | LA0ASI |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI configuration register 5 (PCICONF5) is a 32-bit read/partial-write register that accommodates the memory space base address PCI configuration register stipulated in the PCI local bus specifications. This register holds the high bits (12 max. in bits 31 to 20) of the address used when a device on the PCI bus accesses local memory on the SH local bus using memory transfer commands. Allocate at least the capacity set in the local space register 0 (PCILSR0) as PCI bus memory space.

Bits 19 to 0 are fixed in hardware. Of writable bits 31 to 20, those that hold valid values differ according to the value set in PCILSR0.

Table 22.6 Memory Space Base Address Register (BASE0)

| PCILSR0 [28:20] Register Value | Required Address Space | BASE0[31:20] Valid Writable Bits |
|-----------------------------------|---------------------------|-------------------------------------|
| B'0_0000_0000 | 1 MB | Bits 31 to 20 |
| B'0_0000_0001 | 2 MB | Bits 31 to 21 |
| B'0_0000_0011 | 4 MB | Bits 31 to 22 |
| ⋮ | ⋮ | ⋮ |
| B'0_1111_1111 | 256 MB | Bits 31 to 28 |
| B'1_1111_1111 | 512 MB | Bits 31 to 29 |

The PCICONF5 register is initialized to H'00000000 at a power-on reset and software reset.

Always write to this register before transferring data to and from the PCIC memory from the PCI bus.

Bits 31 to 20—Base Address of the Memory Space 0 (BASE0 31 to 20): These bits specify the base address of the local address space 0 (this LSI external bus space).

Bits 19 to 4—Base Address of the Memory Space 0 (BASE0 19 to 4): Fixed at H'0000 in hardware.

Bit 3—Pre-fetch Control (LA0PREF): Shows availability of prefetching of the local address space 0.

| Bit 3: LA0PREF | Description |
|----------------|-----------------------------------|
| 0 | Prefetch disabled (Initial value) |
| 1 | Prefetch enabled (not supported) |

Bits 2 and 1—LA0TYPE1 and 0: In the case of I/O space, can be set as the base address. Shows the memory type of the local address space 0.

| Bit 2: LA0TYPE1 | Bit 1: LA0TYPE0 | Description |
|-----------------|-----------------|--|
| 0 | 0 | Base address can be set to 32-bit width, 32-bit space (Initial value) |
| | 1 | Base address can be set to 32-bit width, less than 1MB space (not supported) |
| 1 | 0 | Base address is 64-bit width (not supported) |
| | 1 | Reserved |

Bit 0—LA0ASI: Shows whether the base address specified by this register is an I/O space or memory space.

| Bit 0: LA0ASI | Description |
|---------------|------------------------------|
| 0 | Memory space (Initial value) |
| 1 | I/O space |

22.2.7 PCI Configuration Register 6 (PCICONF6)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|---------|---------|---------|---------|---------|----------|----------|---------|
| | BASE131 | BASE130 | BASE129 | BASE128 | BASE127 | BASE126 | BASE125 | BASE124 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BASE123 | BASE122 | BASE121 | BASE120 | BASE119 | BASE118 | BASE117 | BASE116 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | BASE115 | BASE114 | BASE113 | BASE112 | BASE111 | BASE110 | BASE19 | BASE18 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | BASE17 | BASE16 | BASE15 | BASE14 | LA1PREF | LA1TYPE1 | LA1TYPE0 | LA1ASI |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI configuration register 6 (PCICONF6) is a 32-bit read/partial-write register that accommodates the memory space base address PCI configuration register stipulated in the PCI

local bus specifications. This register contains the most significant bits (maximum 12 in bits 31 to 20) of the address used when a device on the PCI bus accesses local memory on the SH local bus using memory transfer commands. Minimally, allocate the capacity set in the local space register 1 (PCILSR1) to PCI bus memory space.

Bits 19 to 0 are fixed in hardware. The number of valid bits of those that can be written to (bit 31 to 20) differs according to the value set in PCILSR1.

Table 22.7 Memory Space Base Address Register (BASE1)

| PCILSR1 [28:20] Register Value | Required Address Space | Valid BASE1 [31:20] Write Bits |
|-----------------------------------|---------------------------|-----------------------------------|
| B'0_0000_0000 | 1 MB | Bits 31 to 20 |
| B'0_0000_0001 | 2 MB | Bits 31 to 21 |
| B'0_0000_0011 | 4 MB | Bits 31 to 22 |
| : | : | : |
| B'0_1111_1111 | 256 MB | Bits 31 to 28 |
| B'1_1111_1111 | 512 MB | Bits 31 to 29 |

The PCICONF6 register is initialized to H'00000000 at a power-on reset and software reset.

Always write to this register prior to transferring data to or from the PCIC memory from the PCI bus.

Bits 31 to 20—Base Address of the Memory Space 1 (BASE1 31 to 20): Specifies the base address of the local address space 1 (this LSI external bus space).

Bits 19 to 4—Base Address of the Memory Space 1 (BASE1 19 to 4): Fixed at H'0000 in hardware.

Bit 3—Pre-fetch Control (LA1PREF): Shows whether the local address space 1 can be pre-fetched.

| Bit 3: LA1PREF | Description | |
|----------------|----------------------------------|-----------------|
| 0 | Prefetch disabled | (Initial value) |
| 1 | Prefetch enabled (not supported) | |

Bits 2 and 1—Memory Type (LA1TYPE1 to 0): These bits indicate the memory type of the local address space 1.

| Bit 2: LA1TYPE1 | Bit 1: LA1TYPE0 | Description |
|-----------------|-----------------|--|
| 0 | 0 | The base address can be set to 32-bit width, 32-bit space (Initial value) |
| | 1 | The base address can be set to 32-bit width, but less than 1MB (not supported) |
| 1 | 0 | The base address has 64-bit width (not supported) |
| | 1 | Reserved |

Bit 0—Address Space Indicator (LA1ASI): Shows whether the base address specified by this register is an I/O space or memory space.

| Bit 0: LA1ASI | Description |
|---------------|------------------------------|
| 0 | Memory space (Initial value) |
| 1 | I/O space |

22.2.8 PCI Configuration Register 7 (PCICONF7) to PCI Configuration Register 10 (PCICONF10)

| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
|----------------|----|----|----|-----|----|----|---|---|
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | | R | R | R | R |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Bits 31 to 0—Reserved: These bits are always read as 0.

22.2.9 PCI Configuration Register 11 (PCICONF11)

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|-------|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | SSID15 | SSID14 | SSID13 | SSID12 | SSID11 | SSID10 | SSID9 | SSID8 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | SSID7 | SSID6 | SSID5 | SSID4 | SSID3 | SSID2 | SSID1 | SSID0 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | SVID15 | SVID14 | SVID13 | SVID12 | SVID11 | SVID10 | SVID9 | SVID8 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-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 |
| | SVID7 | SVID6 | SVID5 | SVID4 | SVID3 | SVID2 | SVID1 | SVID0 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The PCI configuration register 11 (PCICONF11) is a 32-bit read/write register that accommodates the subsystem ID and subsystem vendor ID PCI configuration registers stipulated in the PCI local bus specifications. The register contains the ID of the add-in board that this LSI is installed on its subsystem (bits 31 to 16) as well as the subsystem vendor ID (bits 15 to 0).

All bits can be written to from the PP bus.

The PCICONF11 register is not initialized at a reset. Always initialize this register while the CFINIT bit (bit 0) of the PCICR register is cleared.

Bits 31 to 16—SSID15 to 0: Specifies the subsystem ID.

Bits 15 to 0—SVID15 to 0: Specifies the PCI subsystem vendor ID.

22.2.10 PCI Configuration Register 12 (PCICONF12)

| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
|----------------|----|----|----|-----|----|----|---|---|
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Bits 31 to 0—Reserved: These bits are always read as 0.

22.2.11 PCI Configuration Register 13 (PCICONF13)

| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
|----------------|----|----|----|-----|----|----|---|---|
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| | CAPPTR7 | CAPPTR6 | CAPPTR5 | CAPPTR4 | CAPPTR3 | CAPPTR2 | CAPPTR1 | CAPPTR0 |
| Initial value: | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI configuration register 13 (PCICONF13) is a 32-bit read-only register that accommodates the extended function pointer PCI configuration register stipulated in the PCI power management specifications. The address offset of the extended function is read from bits 7 to 0.

All bits are fixed in hardware.

Bits 31 to 8—Reserved: These bits are always read as 0.

Bits 7 to 0—CAPPTR7 to 0: These bits specify the address offset of the extended functions (power management). The initial value is H'40 (fixed).

22.2.12 PCI Configuration Register 14 (PCICONF14)

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|---|---|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Bits 31 to 0—Reserved: These bits are always read as 0.

22.2.13 PCI Configuration Register 15 (PCICONF15)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | MLAT7 | MLAT6 | MLAT5 | MLAT4 | MLAT3 | MLAT2 | MLAT1 | MLAT0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | MGNT7 | MGNT6 | MGNT5 | MGNT4 | MGNT3 | MGNT2 | MGNT1 | MGNT0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | IPIN7 | IPIN6 | IPIN5 | IPIN4 | IPIN3 | IPIN2 | IPIN1 | IPIN0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ILIN7 | ILIN6 | ILIN5 | ILIN4 | ILIN3 | ILIN2 | ILIN1 | ILIN0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The PCI configuration register 15 (PCICONF15) is a 32-bit read/partial-write register that accommodates the maximum latency, minimum grant, interrupt pin, and interrupt line PCI configuration registers stipulated in the PCI local bus specifications. The interrupt pins used by this LSI is read from bits 15 to 8. Bits 7 to 0 indicate to which of the interrupt request signal lines of an interrupt controller the interrupt line is connected.

Bits 31 to 8 are fixed in hardware. Bits 7 to 0 can be written to from both the PP bus and PCI bus.

The PCICONF15 register is initialized to H'00000100 at a power-on reset and software reset.

Bits 31 to 24—Designation of Maximum Latency (MLAT7 to 0): These bits specify the maximum time from the time the PCI master device demands bus privileges and to the time it obtains the privileges (not supported).

Bits 23 to 16—Minimum Latency Specification (MGNT 7 to 0): Specify the burst interval required by the PCI device (not supported).

Bits 15 to 8—Interrupt Pin Specification (IPIN7 to 0)

Bits 15 to 8:

| IPIN7 to 0 | Description | |
|-------------------|-------------------------------|-----------------|
| H'01 | $\overline{\text{INTA}}$ used | (Initial value) |
| H'02 | $\overline{\text{INTB}}$ used | |
| H'03 | $\overline{\text{INTC}}$ used | |
| H'04 | $\overline{\text{INTD}}$ used | |
| H'05 to H'FF | Reserved bits | |

Bits 7 to 0—Interrupt Line Specification (ILIN7 to 0): Specifies an interrupt line of a system to which interrupt output used by the PCIC is connected.

22.2.14 PCI Configuration Register 16 (PCICONF16)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|---------|---------|---------|---------|---------|--------|--------|--------|
| | PMESPT4 | PMESPT3 | PMESPT2 | PMESPT1 | PMESPT0 | D2SPT | D1SPT | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | DS1 | — | PMECLK | VER2 | VER1 | VER0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | NIP7 | NIP6 | NIP5 | NIP4 | NIP3 | NIP2 | NIP1 | NIP0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CAPID7 | CAPID6 | CAPID5 | CAPID4 | CAPID3 | CAPID2 | CAPID1 | CAPID0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI configuration register 16 (PCICONF16) is a 32-bit read/partial-write register that accommodates the power management function (PMC), next-item pointer, and extended function ID power management registers stipulated in the PCI power management specifications.

PCICONF16 is valid only when the PCIC is functioning not as the host. The power management related functions are read from bits 31 to 16 (PMC), the address offset of the next function in the extended function list is read from bits 15 to 8 (next item pointer), and the power management ID (H'01) is read from bits 7 to 0 (extended function ID).

Bits 18 to 16 can be written to from the PP bus only. Other bits are fixed in hardware.

The PCICONF16 register is initialized to H'00010001 at a power-on reset and a software reset.

Bits 31 to 27—PME Support (PMESPT4 to 0): Not supported. Defines the function state supporting $\overline{\text{PME}}$ output.

Bit 26—D2 Support (D2SPT): Not supported. Specifies whether D2 state is supported.

Bit 25—D1 Support (D1SPT): Not supported. Specifies whether D1 state is supported.

Bits 24 to 22—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 21—DSI: Specifies whether bit-device-specific initialization is required.

Bit 20—Reserved: This bit always returns 0 when read. Always write 0 to this bit.

Bit 19—PME Clock (PMECLK): Not supported. Specifies whether a clock is required for $\overline{\text{PME}}$ support.

Bits 18 to 16—Version (VER2 to 0): Specify the version of power management specifications.

Bits 15 to 8—Next Item Pointer (NIP7 to 0): Specify the offset to the next extended function register

Bits 7 to 0—Extended Function ID (CAPID7 to 0): Extended function (Capability Identifier) ID. These bits always return H'01 when read.

22.2.15 PCI Configuration Register 17 (PCICONF17)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|-------|----------|----------|----------|----------|----------|----------|--------|
| | DATA7 | DATA6 | DATA5 | DATA4 | DATA3 | DATA2 | DATA1 | DATA0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PMEST | DTATSCL1 | DTATSCL0 | DATASEL3 | DATASEL2 | DATASEL1 | DATASEL0 | PMEEN |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | PWRST1 | PWRST0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R/W | R/W |
| PP Bus-R/W: | R | R | R | R | R | R | R/W | R/W |

The PCI configuration register 17 (PCICONF17) is a 32-bit read/partial-write register that accommodates the power management control/status (PMCSR), bridge-compatible PMCSR extended (PMCSR_BSE), and data power management registers stipulated in the PCI power management specifications. PCICONF17 is valid only when the PCIC is operating not as the host. Bits 31 to 24 (data) and bits 23 to 16 (PMCSR_BSE) are not supported. The power management status is read from bits 15 to 0 (PMCSR).

Bits 1 and 0 can be written to from both the PP bus and the PCI bus. Other bits are fixed in hardware.

PCICONF17 is initialized to H'00000000 at a power-on reset and software reset.

When B'11 is written to bits 1 and 0 and a transition is made to power state D3 (power down mode), PCIC operation as a master target is disabled, regardless of the setting of bits 2 to 0 of the PCICONF1 (bus master control, memory and I/O space access control) (these bits are masked). When B'00 is written to bits 1 and 0 and a transition is made to power state D0 (normal operating mode), the mask is canceled.

Bits 31 to 24—DATA (DATA7 to 0): Not supported. Data field for power management.

Bits 23 to 16—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 15—PME Status (PMEST): Not supported. Shows the status of the $\overline{\text{PME}}$ bit. This bit is set when the signal is output.

Bits 14 and 13—Data Scale (DTATSCL1 to 0): Not supported. These bits specify the scaling value for the data field value.

Bits 12 to 9—Data Select (DATASEL3 to 0): Not supported. Select the value to be output to the data field.

Bit 8— $\overline{\text{PME}}$ Enable (PMEEN): Not supported. Controls the $\overline{\text{PME}}$ signal output.

Bits 7 to 2—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bits 1 and 0—Power State (PWRST1 and 0): Specifies the power state. No state transition is effected when a non-supported state is specified. (Normal termination, no error output.)

| Bit 1: PWRST1 | Bit 0: PWRST0 | Description |
|---------------|---------------|--|
| 0 | 0 | D0 state (Initial value, normal state) |
| | 1 | D1 state (not supported) |
| 1 | 0 | D2 state (not supported) |
| | 1 | D3 state (power down mode) |

22.2.16 Reserved Area

Reserved area.

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|---|---|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

Note: PCI configuration addresses H'48 to H'FC are reserved.

Bits 31 to 0—Reserved: These bits always return 0 when read.

22.2.17 PCI Control Register (PCICR)

| | | | | | | | | |
|----------------|--------|-------|------|------|------|------|--------|----------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | TRDSGL | BYTESWAP |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R/W | R/W |
| PP Bus-R/W: | R | R | R | R | R | R | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCIPUP | BMABT | MD10 | MD9 | SERR | INTA | RSTCTL | CFINIT |
| Initial value: | 0 | 0 | 0/1* | 0/1* | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R | R | R/W | R/W | R/W | R/W |

Note: * The value of the external pin is sampled in a power-on reset by means of the $\overline{\text{RESET}}$ pin.

The PCI control register (PCICR) is a 32-bit register that monitors the status of the mode pin at initialization and controls the basic operation of the PCIC. Bits 5 (MD10) and 4 (MD9) are read-only bits from the PP bus. Other bits are read/write bits. Bits 9 (TRDSGL) and 8 (BYTESWAP) are read/write bits from the PCI bus. Other bits are read-only.

In PCIC host operation, a software reset can be applied to the PCI bus by means of bit 1 (RSTCTL) of PCICR. When a software reset is executed, the $\overline{\text{PCIRST}}$ pin is asserted and the internal state of the PCIC is initialized.

The PCICR register is initialized at a power-on reset to H'000000*0 (bits 7 and 6 are initialized to B'00, and bits 5 and 4 sample the value of mode pins 9 and 10). At a software reset, bit 1 (RSTCTL) is not initialized. All other bits are initialized in the same way as at a software reset.

This register can be written to only when bits 31 to 24 are H'A5.

Always set bit 0 (CFINIT) to 1 on completion of PCIC register initialization.

Bits 31 to 10—Reserved: These bits are always read as 0. When writing, write H'A5 to bits 31 to 24, and 0 to others.

Bit 9—Target Read Single Buffer (TRDSGL): This bit specifies whether one target read buffer (32 bytes) or two target read buffers (64 bytes) are used for target memory read access to the PCIC. When two target read buffers faces are used, the data from two buffers are read via the local bus in advanced.

| Bit 9: TRDSGL | Description | |
|---------------|-------------------------------|-----------------|
| 0 | Use 2 target read buffers | (Initial value) |
| 1 | Use 1 target read buffer only | |

Bit 8—Data Byte Swap (BYTESWAP): Specifies whether the data byte is swapped when the PCIC performs PIO transfer.

| Bit 8: BYTESWAP | Description | |
|-----------------|-------------------------------|-----------------|
| 0 | Send data as-is | (Initial value) |
| 1 | Swap data byte before sending | |

Note: For details, refer to section 22.4, Endians.

Bit 7—PCI Signal Pull-up (PCIUP): Controls the pull-up resistance of the PCI signal. Regarding the pins that are subject to pull-up, refer to table 22.1. Regarding the pull-up control provided when the $\overline{\text{PCIQE2}}/\text{MD9}$, $\overline{\text{PCIQE3}}/\text{MD10}$ or $\overline{\text{PCIQE4}}$ is used as a port, refer to the section on port control register (PCIPCTR).

| Bit 7: PCIUP | Description | |
|--------------|-------------|-----------------|
| 0 | Pull-up | (Initial value) |
| 1 | No pull-up | |

Bit 6—Bus Master Arbitration (BMABT): Controls the PCI bus arbitration mode of the PCIC when the PCIC is operating as the host. When the PCIC is non-host, the value of this bit is ignored.

| Bit 6: BMABT | Description |
|--------------|--|
| 0 | Fixed priority order (device 0 (PCIC) > device 1 > device 2 > device 3 > device 4) (Initial value) |
| 1 | Pseudo round-ribbon (The priority level of the device with bus privileges is set lowest at the next access.) |

Bit 5—Mode 10 Pin Monitor (MD10): Monitors the $\overline{\text{PCIREQ3}}$ /MD10 pin value in a power-on reset by means of the $\overline{\text{RESET}}$ pin.

| Bit 5: MD10 | Description |
|-------------|--|
| 0 | Host bridge function (arbitration) enabled |
| 1 | Host bridge function disabled |

Bit 4—Mode 9 Pin Monitor (MD9): Monitors the $\overline{\text{PCIREQ2}}$ /MD9 pin value in a power-on reset by means of the $\overline{\text{RESET}}$ pin.

| Bit 4: MD9 | Description |
|------------|--|
| 0 | PCICLK used as PCI clock |
| 1 | Feedback input clock from CKIO used as PCI clock |

Bit 3— $\overline{\text{SERR}}$ Output (SERR): Software control of $\overline{\text{SERR}}$ output. This bit is valid only when bit 8 (SER) of the PCICONFI register is “1”. When “1” is written to this bit, $\overline{\text{SERR}}$ is asserted for 1 clock. This bit always returns “0” when read. Used when the PCIC is not the host. If used when the PCIC is the host, an $\overline{\text{SERR}}$ assert interrupt is generated to the CPU.

| Bit 3: SERR | Description |
|-------------|---|
| 0 | $\overline{\text{SERR}}$ pin at Hi-Z (driven to High by pull-up resistor) (Initial value) |
| 1 | Assert $\overline{\text{SERR}}$ (Low output) |

Bit 2— $\overline{\text{INTA}}$ Output (INTA): Software control of $\overline{\text{INTA}}$ (valid only when PCIC is not host)

| Bit 2: INTA | Description |
|-------------|---|
| 0 | $\overline{\text{INTA}}$ pin at Hi-Z (driven to High by pull-up resistor) (Initial value) |
| 1 | Assert $\overline{\text{INTA}}$ (Low output) |

Bit 1—PCIRST Output Control (RSTCTL): Controls the $\overline{\text{PCIRST}}$ output. This field is reset only at a power-on reset. Do not use the field when the PCIC is non-host.

| Bit 1: PCIRST | Description |
|---------------|---|
| 0 | Negate $\overline{\text{PCIRST}}$ (High output) (Initial value) |
| 1 | Assert $\overline{\text{PCIRST}}$ (Low output) |

Bit 0—PCIC Internal Register Initialization Control Bit (CFINIT): After the SH initializes the PCI registers, setting this bit enables access from the PCI bus. During initialization, no bus privileges are granted to other devices on the PCI bus while operating as the host. When operating not as the host, a retry is returned without the access from the PCI bus being accepted.

| Bit 0: CFINIT | Description |
|---------------|-------------------------------------|
| 0 | Initialization busy (Initial value) |
| 1 | Initialization complete |

22.2.18 PCI Local Space Register [1:0] (PCILSR [1:0])

| | | | | | | | | |
|----------------|----|----|----|--------|--------|--------|--------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | PLSR28 | PLSR27 | PLSR26 | PLSR25 | PLSR24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|--------|--------|--------|--------|----|----|----|----|
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | PLSR23 | PLSR22 | PLSR21 | PLSR20 | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R | R | R | R |

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

| | | | | | | | | |
|----------------|---|---|---|---|---|---|---|---|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI local space register [1:0] (PCILSR [1:0]) specifies the capacities of the two local address spaces (address space 0 and address space 1) supported when a device on the PCI bus performs a memory read/memory write of the PCIC using target transfers. This is a 32-bit register that can be read and written from the PP bus, or read only from the PCI bus.

The PCILSR [1:0] register is initialized to H'00000000 at a power-on reset and software reset.

Always write to this register before performing target transfers to specify the capacity of the address space being used. Specify the value “(capacity –1) bytes” in bits 28 to 20. For example, to secure a 32MB space, set the value H'01F00000.

If you specify all zeros, a 1MB space is reserved. You can specify an address space up to 512MB. Refer to table 22.6 in section 22.2.6, PCI Configuration Register 5 (PCICONF5).

Bits 31 to 29—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bits 28 to 20—Capacities of the Local Address Spaces 0, 1 (PLSR28 to 20): These bits specify the capacities of the address space 0 and address space 1 in bytes. Specifying (capacity –1) bytes. A 1MB space is secured if all zeros are specified.

Bits 19 to 0—Reserved: These bits always return 0 when read. Always write 0 to these bits.

22.2.19 PCI Local Address Register [1:0] (PCILAR [1:0])

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | LAR28 | LAR27 | LAR26 | LAR25 | LAR24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | LAR23 | LAR22 | LAR21 | LAR20 | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI local address register [1:0] (PCILAR [1:0]) specifies the starting address (external address of local bus) of the two local address spaces (address space 0 and address space 1) supported when performing memory read/memory write operations due to target transfers to the PCIC. It is a 32-bit register that can be read and written from the PP bus and is read-only from the PCI bus.

The PCILAR [1:0] register is initialized to H'00000000 at a power-on reset and software reset.

The valid bits of the local address specified by this register vary according to the capacity of the address space specified in the PCILSR [1:0] register. In other words, set 0 in the least significant address bit which corresponds to the capacity set by PCILSR0, 1, and set the starting address only

in the most significant address bit. For example, when the capacity of the local address space is set to 32MB (PCILSR: H'01F00000), bits 28 to 25 of the local address are valid. Only the value set in these bits is used as the physical address of the local address space.

Always write to this register prior to target transfers. Specify the starting address (physical address) of the memory installed on the local bus according to the address space being used. Bits 28 to 26 of the PCI local address register 0 select the local address area. Bits 25 to 20 show the address within that area.

Bits 31 to 29—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bits 28 to 20—Local Address (LAR28 to 20): Specify bits 28 to 20 of the starting address of the local address space.

Bits 19 to 0—Reserved: These bits always return 0 when read. Always write 0 to these bits.

22.2.20 PCI Interrupt Register (PCIINT)

| | | | | | | | | |
|----------------|--------------|-----------------|----------------|---------------|-----------------|-----------------|----------------|----------------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | M_LOCK ON | T_TGT_A BORT | — | — | — | — | TGT_RET RY | MST_DIS |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/WC | R/WC | R | R | R | R | R/WC | R/WC |
| PP Bus-R/W: | R/WC | R/WC | R | R | R | R | R/WC | R/WC |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ADPER R | SERR_D ET | T_DPER R_WT | T_PERR DET | M_TGT_A BORT | M_MST_ ABORT | M_DPER R_WT | M_DPER R_RD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/WC | R/WC | R/WC | R/WC | R/WC | R/WC | R/WC | R/WC |
| PP Bus-R/W: | R/WC | R/WC | R/WC | R/WC | R/WC | R/WC | R/WC | R/WC |

Note: WC: Cleared by writing “1”. (Writing of 0 is ignored.)

The PCI interrupt register (PCIINT) is a 32-bit register that saves the error source when an error occurs on the PCI bus as a result of the PCIC attempting to invoke a transfer on the PCI bus, or when the PCIC is the PCI master or PCI target. This register can be read from both the PP bus and PCI bus. Also, 1 can be written from either the PP bus or PCI bus to perform a write-clear in which the detection bit is cleared to its initial value (0).

The PCIINT register is initialized to H'00000000 at a power-on reset or software reset.

When an error occurs, the bit corresponding to the error content is set to 1. Each interrupt detection bit can be cleared to its initial status (0) by writing 1 to it. (Write clear)

Note that the error detection bits can be set even when the interrupt is masked.

The error source holding circuit can only store one error source. For this reason, any second or subsequent error factors are not stored if errors occur consecutively.

Bits 31 to 16—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 15—Unlocked Transfer Detection Interrupt (M_LOCKON): When the PCIC is master, an unlocked PIO transfer was performed when the I-specified target was locked.

Bit 14—Target Target Abort Interrupt (T_TGT_ABORT): Indicates the termination of transaction by target abort when the PCIC is a target. Target abort is generated when the 2 least significant address bits (bits 1, 0) and byte enable constitute an illegal combination (illegal byte enable) during I/O transfer.

Bits 13 to 10—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 9—Target Memory Read Retry Timeout Interrupt (TGT_RETRY): When the PCIC is target, the master did not attempt a retry within the prescribed number of PCI bus clocks (2^{15}) (detected only in the case of memory read operations).

Bit 8—Master Function Disable Error Interrupt (MST_DIS): Indicates that an attempt was made to conduct a master operation (PIO transfer, DMA transfer) when bit 2 (BUM) of the PCICONF1 was set to 0 to prohibit bus master operations.

Bit 7—Address Parity Error Detection Interrupt (ADRPERR): Address parity error detected. Detects only when bit 6 (PER) and bit 8 (SER) of the PCICONF1 are both 1.

Bit 6— $\overline{\text{SERR}}$ Detection Interrupt (SERR_DET): When the PCIC is host, assertion of the $\overline{\text{SERR}}$ signal was detected.

Bit 5—Target Write Data Parity Error Interrupt (T_DPERR_WT): When the PCIC is target, a data parity error was detected while receiving a target write transfer (only detected when PCICONFI bit 6 (PER) is 1).

Bit 4—Target Read $\overline{\text{PERR}}$ Detection Interrupt (T_PERR_DET): When the PCIC is target, $\overline{\text{PERR}}$ was detected when receiving a target read transfer. Detects only when bit 6 (SER) of the PCICONF1 is 1.

Bit 3—Master Target Abort Interrupt (M_TGT_ABORT): When the PCIC is master. Indicates the termination of transaction by target abort.

Bit 2—Master Master Abort Interrupt (M_MST_ABORT): When the PCIC is master. Indicates the termination of transaction by master abort.

Bit 1—Master Write $\overline{\text{PERR}}$ Detection Interrupt (M_DPERR_WT): When the PCIC is master. $\overline{\text{PERR}}$ received from the target while writing data to the target. Detects only when bit 6 (PER) of the PCICONF1 is 1.

Bit 0—Master Read Data Parity Error Interrupt (M_DPERR_RD): When the PCIC is master, a parity error was detected during a data read from the target. Detects only when bit 6 (PER) of the PCICONF1 is 1.

22.2.21 PCI Interrupt Mask Register (PCIINTM)

| | | | | | | | | |
|----------------|--------------|-----------------|----------------|----------------|-----------------|-----------------|----------------|----------------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | M_LOCK ON | T_TGT_A BORT | — | — | — | — | TGT_RET RY | MST_DIS |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R | R | R | R | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R | R | R | R | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ADRP R | SERR_D ET | T_DPER R_WT | T_PERR_ DET | M_TGT_A BORT | M_MST_ ABORT | M_DPER R_WT | M_DPER R_RD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The PCI interrupt mask register (PCIINTM) sets the respective interrupt masks for the interrupts generated when errors occur in PCI transfers. It is a 32-bit read/write register that can be accessed from both the PP bus and PCI bus. When set to 0, the respective interrupt is disabled, and enabled when set to 1.

The PCIINTM register is initialized to H'00000000 at a power-on reset and software reset.

Bits 31 to 16—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 15—Unlocked Transfer Detection Interrupt Mask (M_LOCKON)

Bit 14—Target Target Abort Interrupt Mask (T_TGT_ABORT)

Bits 13 to 10—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bit 9—Target Retry Timeout Interrupt Mask (TGT_RETRY)

Bit 8—Master Function Disable Error Interrupt Mask (MST_DIS)

Bit 7—Address Parity Error Detection Interrupt Mask (ADRPERR)

Bit 6— $\overline{\text{SERR}}$ Detection Interrupt Mask (SERR_DET)

Bit 5—Target Write Data Parity Error Interrupt Mask (T_DPERR_WT)

Bit 4—Target Read $\overline{\text{PERR}}$ Detection Interrupt Mask (T_PERR_DET)

Bit 3—Master Target Abort Interrupt Mask (M_TGT_ABORT)

Bit 2—Master Master Abort Interrupt Mask (M_MST_ABORT)

Bit 1—Master Write Data Parity Error Interrupt Mask (M_DPERR_WT)

Bit 0—Master Read Data Parity Error Interrupt Mask (M_DPERR_RD)

22.2.22 PCI Address Data Register at Error (PCIALR)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| | ALOG31 | ALOG30 | ALOG29 | ALOG28 | ALOG27 | ALOG26 | ALOG25 | ALOG24 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | ALOG23 | ALOG22 | ALOG21 | ALOG20 | ALOG19 | ALOG18 | ALOG17 | ALOG16 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | ALOG15 | ALOG14 | ALOG13 | ALOG12 | ALOG11 | ALOG10 | ALOG9 | ALOG8 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ALOG7 | ALOG6 | ALOG5 | ALOG4 | ALOG3 | ALOG2 | ALOG1 | ALOG0 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI address data register at error (PCIALR) stores the PCI address data (ALOG [31:0]) of errors that occur on the PCI bus. It is a 32-bit register that can be read from both the PP bus and PCI bus.

The PCIALR register is not initialized at a power-on reset or software reset. The initial value is undefined. A valid value is retained only when one of the PCIINT register bits is set to 1.

The error source holding circuit can only store one error source. For this reason, any second or subsequent error factors are not stored if errors occur consecutively.

Bits 31 to 0—Address Log (ALOG31 to 0): PIC address data (value of A/D line) at time of error. (Initial value is undefined.)

22.2.23 PCI Command Data Register at Error (PCICLR)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|--------|---------|---------|---------|---------|---------|---------|---------|
| | MSTPIO | MSTDMA0 | MSTDMA1 | MSTDMA2 | MSTDMA3 | TGT | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMDLOG3 | CMDLOG2 | CMDLOG1 | CMDLOG0 |
| Initial value: | 0 | 0 | 0 | 0 | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI command data register at error (PCICLR) stores the type of transfer (MSTPIO, MSTDMA0, MSTDMA1, MSTDMA2, MSTDMA3, or TGT) when an error occurs on the PCI bus, and the PCI command (CMDLOG [3:0]). It is a 32-bit register that can be read from both the PP bus and PCI bus.

Although bits 31 to 26 of the PCICLR register are initialized at a power-on reset and a software reset, bits 3 through 0 are not initialized. When an error is detected, 1 is set in one of bits 31 to 26, and the relevant command value is retained in bits 3 to 0.

A valid value is retained only when one of the PCIINT register bits is set to 1.

The error source holding circuit can only store one error source. For this reason, any second or subsequent error factors are not stored if errors occur consecutively.

Bit 31—PIO Error (MSTPIO): Error occurred in PIO transfer.

Bit 30—DMA0 Error (MSTDMA0): Error occurred in DMA channel 0 transfer.

Bit 29—DMA1 Error (MSTDMA1): Error occurred in DMA channel 1 transfer.

Bit 28—DMA2 Error (MSTDMA2): Error occurred in DMA channel 2 transfer.

Bit 27—DMA3 Error (MSTDMA3): Error occurred in DMA channel 3 transfer.

Bit 26—Target Error (TGT): Error occurred in target read or target write transfer.

Bits 25 to 4—Reserved: These bits are always read as 0.

Bits 3 to 0—Command Log (CMDLOG3 to 0): These bits retain the PCI transfer command information (value of C/BE line) upon detection of an error. (Initial value is undefined.)

22.2.24 PCI Arbiter Interrupt Register (PCIAINT)

| | | | | | | | | |
|----------------|----|----|----------|-----------|-----------|-----------|----------|----------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | MST_BRKN | TGT_BUSTO | MST_BUSTO | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R/WC | R/WC | R/WC | R | R | R |
| PP Bus-R/W: | R | R | R/WC | R/WC | R/WC | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | TGT_ABORT | MST_ABORT | DPERR_WT | DPERR_RD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R/WC | R/WC | R/WC | R/WC |
| PP Bus-R/W: | R | R | R | R | R/WC | R/WC | R/WC | R/WC |

Note: Cleared by writing WC:1. (Writing of 0 is ignored.)

The PCI arbiter interrupt register (PCIAINT) is a 32-bit register that stores the sources of PCI bus errors occurring during transfers by another PCI master device when the PCIC is operating as the host with the arbitration function. The register can be read from both the PP bus and the PCI bus. Also, each interrupt detection bit can be cleared to its initial status (0) by writing 1 to it from either the PP bus or the PCI bus. (Write clear)

The PCIAINT register is initialized to H'00000000 at a power-on reset or software reset.

When an error is detected, the bit corresponding to the error type is set to 1. Each interrupt detection bit can be cleared to its initial status (0) by writing 1 to it. (Write clear)

The error detection bits are set even when the interrupts are masked.

Bits 31 to 14—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 13—Master Broken Interrupt (MST_BRKN): Detects when the master granted with bus privileges does not start a transaction ($\overline{\text{FRAME}}$ not asserted) within 16 clocks. For the SH7751, see 22.12, Usage Notes.

Bit 12—Target Bus Timeout Interrupt (TGT_BUSTO): Neither $\overline{\text{TRDY}}$ nor $\overline{\text{STOP}}$ are not returned within 16 clocks in the case of the first data transfer, or within 8 clocks in the case of second and subsequent data transfers. For the SH7751, see 22.12, Usage Notes.

Bit 11—Master Bus Timeout Interrupt (MST_BUSTO): Indicates the detection that $\overline{\text{IRDY}}$ was not asserted within 8 clock cycles in a transaction initiated by a device including PCIC.

Bits 10 to 4—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 3—Target Abort Interrupt (TGT_ABORT): Indicates the termination of transaction by target abort when a device other than the PCIC is operating as the bus master.

Bit 2—Master Abort Interrupt (MST_ABORT): Indicates the termination of transaction by master abort when a device other than the PCIC is operating as the bus master.

Bit 1—Write Data Parity Error Interrupt (DPERR_WT): Indicates the detection of the assertion of $\overline{\text{PERR}}$ in a data write operation when a device other than the PCIC is operating as the bus master.

Bit 0—Read Data Parity Error Interrupt (DPERR_RD): Indicates the detection of the assertion of $\overline{\text{PERR}}$ in a data read operation when a device other than the PCIC is operating as the bus master.

22.2.25 PCI Arbiter Interrupt Mask Register (PCIAINTM)

| | | | | | | | | |
|----------------|----|----|----------|-----------|-----------|-----------|----------|----------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | MST_BRKN | TGT_BUSTO | MST_BUSTO | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R/W | R/W | R/W | R | R | R |
| PP Bus-R/W: | R | R | R/W | R/W | R/W | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | TGT_ABORT | MST_ABORT | DPERR_WT | DPERR_RD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R | R | R | R | R/W | R/W | R/W | R/W |

The PCI arbiter interrupt mask register (PCIAINTM) sets interrupt masks for the individual interrupts that occur due to errors generated during PCI transfers performed by other PCI devices when the PCIC is operating as the host with the arbitration function. It is a 32-bit register that is readable and writable from both the peripheral bus and the PCI bus. Each bit is set to 0 to disable the respective interrupt, or 1 to enable that interrupt.

The PCIINTM register is initialized to H'00000000 at a power-on reset or software reset.

Bits 31 to 14—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 13—Master Broken Interrupt Mask (MST_BRKN)

Bit 12—Target Bus Timeout Interrupt Mask (TGT_BUSTO)

Bit 11—Master Bus Timeout Interrupt Mask (MST_BUSTO)

Bits 10 to 4—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 3—Target Abort Interrupt Mask (TGT_ABORT)

Bit 2—Master Abort interrupt Mask (MST_ABORT)

Bit 1—Read Data Parity Error Interrupt Mask (DPERR_WT)

Bit 0—Write Data Parity Error Interrupt Mask (DPERR_RD)

22.2.26 PCI Error Bus Master Data Register (PCIBMLR)

| | | | | | | | | |
|----------------|----|----|----|--------|--------|--------|--------|--------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | REQ4ID | REQ3ID | REQ2ID | REQ1ID | REQ0ID |
| Initial value: | 0 | 0 | 0 | — | — | — | — | — |
| PCI-R/W: | R | R | R | R | R | R | R | R |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |

The PCI error bus master data register (PCIBMLR) stores the device number of the bus master at the time an error occurred in PCI transfer by another PCI device when the PCIC was operating as the host with the arbitration function. It is a 32-bit register than can be read from both the PP bus and PCI bus.

The PCIINTM register is initialized to H'00000000 at a power-on reset or software reset. A valid value is retained only when one of the PCIAINT register bits is set to 1.

The bus master data holding circuit can only store data for one master. For this reason, no bus master data is stored for any second or subsequent errors if errors occur consecutively.

Bits 31 to 5—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 4—REQ4 Error (REQ4ID): Error occurred when device 4 (REQ4) was bus master.

Bit 3—REQ3 Error (REQ3ID): Error occurred when device 3 (REQ3) was bus master.

Bit 2—REQ2 Error (REQ2ID): Error occurred when device 2 (REQ2) was bus master.

Bit 1—REQ1 Error (REQ1ID): Error occurred when device 1 (REQ1) was bus master.

Bit 0—REQ0 Error (REQ0ID): Error occurred when device 0 (REQ0) was bus master.

22.2.27 PCI DMA Transfer Arbitration Register (PCIDMABT)

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|---|-------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | DMABT |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R | R/W |
| PP Bus-R/W: | R | R | R | R | R | R | R | R/W |

The PCI DMA transfer arbitration register (PCIDMABT) is a register that controls the arbitration mode in the case of DMA transfers. Two types of DMA arbitration mode can be selected: priority-fixed and pseudo round-robin. This 32-bit read/write register can be accessed from both the PP bus and PCI bus.

The PCIDMABT register is initialized to H'00000000 at a power-on reset or software reset.

Always write to this register to specify the DMA transfer arbitration mode prior to starting DMA transfers.

Bits 31 to 1—Reserved: These bits always returns 0 when read. Always write 0 to these bits when writing.

Bit 0—DMA Arbitration Mode (DMABT): Controls the DMA arbitration mode.

| Bit 0: DMABT | Description |
|--------------|---|
| 0 | Priority-fixed (Channel 0 > Channel 1 > Channel 2 > Channel 3) (Initial value) |
| 1 | Pseudo round-robin |

22.2.28 PCI DMA Transfer PCI Address Register [3:0] (PCIDPA [3:0])

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| | PDPA31 | PDPA30 | PDPA29 | PDPA28 | PDPA27 | PDPA26 | PDPA25 | PDPA24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | PDPA23 | PDPA22 | PDPA21 | PDPA20 | PDPA19 | PDPA18 | PDPA17 | PDPA16 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PDPA15 | PDPA14 | PDPA13 | PDPA12 | PDPA11 | PDPA10 | PDPA9 | PDPA8 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-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 |
| | PDPA7 | PDPA6 | PDPA5 | PDPA4 | PDPA3 | PDPA2 | PDPA1 | PDPA0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The DMA transfer PCI address register [3:0] (PCIDPA [3:0]) specifies the starting address at the PCI when performing DMA transfers. This 32-bit read/write register can be accessed from both the PP bus and PCI bus.

The PCIDPA register is initialized to H'00000000 at a power-on reset and a software reset.

The transfer address of a byte boundary or character boundary can be set, but the 2 least significant bits of this register are ignored, and the data of the longword boundary is transferred.

Before starting a DMA transfer, be sure to write to this register. After a DMA transfer starts, the value in the register is not retained. Always re-set the register value before starting a new DMA transfer after a DMA transfer has been completed.

Bits 31 to 0—DMA Transfer PCI Starting Address (PDPA31 to 0): Set the PCI starting address for DMA transfer.

22.2.29 PCI DMA Transfer Local Bus Start Address Register [3:0] (PCIDLA [3:0])

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| | — | — | — | PDLA28 | PDLA27 | PDLA26 | PDLA25 | PDLA24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | PDLA23 | PDLA22 | PDLA21 | PDLA20 | PDLA19 | PDLA18 | PDLA17 | PDLA16 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PDLA15 | PDLA14 | PDLA13 | PDLA12 | PDLA11 | PDLA10 | PDLA9 | PDLA8 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-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 |
| | PDLA7 | PDLA6 | PDLA5 | PDLA4 | PDLA3 | PDLA2 | PDLA1 | PDLA0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The DMA transfer local bus start address register [3:0] (PCIDLA [3:0]) specifies the starting address at the local bus when performing DMA transfers.

This 32-bit read/write register can be accessed from both the PP bus and PCI bus.

The PCIDLA register is initialized to H'00000000 at a power-on reset and a software reset.

The transfer address of a byte boundary or character boundary can be set, but the 2 least significant bits of the register are ignored, and the data of the longword boundary is transferred. Note that the local bus starting address set in this register is the external address of the SH bus.

Always write to this register prior to starting DMA transfers. After a DMA transfer starts, the register value is not retained. Always re-set this register before starting a new DMA transfer after a DMA transfer has completed.

Bits 31 to 29—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bits 28 to 0—DMA Transfer Local Bus Starting Address (PDLA28 to 0): These bits set the starting address of the local bus (external address of SH bus) for DMA transfer. Bits 28 to 26 indicate the local bus area.

22.2.30 PCI DMA Transfer Counter Register [3:0] (PCIDTC [3:0])

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|----|----|----|----|----|----|-------|-------|
| | — | — | — | — | — | — | PTC25 | PTC24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R | R/W | R/W |
| PP Bus-R/W: | R | R | R | R | R | R | R/W | R/W |

| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | PTC23 | PTC22 | PTC21 | PTC20 | PTC19 | PTC18 | PTC17 | PTC16 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|-------|-------|-------|-------|-------|-------|------|------|
| | PTC15 | PTC14 | PTC13 | PTC12 | PTC11 | PTC10 | PTC9 | PTC8 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-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 |
|----------------|------|------|------|------|------|------|------|------|
| | PTC7 | PTC6 | PTC5 | PTC4 | PTC3 | PTC2 | PTC1 | PTC0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The DMA transfer counter register [3:0] (PCIDTC [3:0]) specifies the number of bytes for DMA transfers. This 32-bit read/write register can be accessed from both the PP bus and PCI bus. When read during a DMA transfer, it returns the remaining number of bytes in the DMA transfer.

The PCIDTC register is initialized to H'00000000 at a power-on reset and a software reset.

Bits 25 to 0 are used to specify the number of transfer bytes. When set to H'00000000, the maximum 64MB transfer is performed. Since the transfer data size corresponds only to longword data, the 2 least significant bits are ignored.

Always write to this register prior to starting a DMA transfer. Please re-set this register when starting a new DMA transfer after a DMA transfer completes.

Bits 31 to 26—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bits 25 to 0—DMA Transfer Byte Count (PTC25 to 0): Specify the number of bytes in DMA transfer. The maximum number of transfer bits are 64 MB (when set to H'00000000).

22.2.31 PCI DMA Control Register [3:0] (PCIDCR [3:0])

| | | | | | | | | |
|----------------|-------|-------|--------|-----|--------|---------|---------|---------|
| Bit: | 31 | 30 | 29 | ... | 19 | 18 | 17 | 16 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | ... | R | R | R | R |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | ALNMD10 | ALMMD9 | DMAST |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R | R | R | R | R | R/W | R/W | R |
| PP Bus-R/W: | R | R | R | R | R | R/W | R/W | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DMAIM | DMAIS | LAHOLD | — | IOSEL0 | DIR | DMASTOP | DMASTRT |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | R/W | R/WC | R/W | R | R/W | R/W | R/W | R/W |
| PP Bus-R/W: | R/W | R/WC | R/W | R | R/W | R/W | R/W | R/W |

Note: Cleared by writing WC:1. (Writing of 0 is ignored.)

The DMA transfer control register [3:0] (PCIDCR [3:0]) specifies the operating mode of the respective channels and the method of transfer, etc. This 32-bit read/write register can be accessed from the PP bus and PCI bus.

The PCIDCR register is initialized to H'00000000 at a power-on reset and software reset.

Writing 1 to bit 0 (DMASTRT) starts DMA transfer. Always re-set the value in this register before starting a new DMA transfer after completion of a DMA transfer.

When setting the DMASTOP bit, do not write 1 to the DMASTART bit. Also, write the same setting at the start of transfer to the DMAIM, DMAIS, LAHOLD, IOSEL and DIR bits.

Example: Starting transfer with PCIDCR = H'00000085
 Forced DMA termination PCIDCR = H'00000086

If DMA is forcibly terminated with a value other than the setting used in the transfer being performed, data accuracy is not guaranteed.

Bits 31 to 11—Reserved: These bits always return 0 when read. Always write 0 to these bits.

Bits 10 and 9—Alignment Mode (ALNMD): Sets data alignment when local bus is big endian

| Bit 10: ALNMD10 | Bit 9: ALNMD9 | Description |
|-----------------|---------------|--|
| 0 | 0 | Byte boundary mode (Initial value) |
| | 1 | W/LW boundary mode 1 (LW data is sent as byte × 4) |
| 1 | 0 | W/LW boundary mode 2 (LW data is sent as word × 2) |
| | 1 | W/LW boundary mode 3 (LW data is sent as longword) |

Legend:

W: Word

LW: Longword

Note: For details, refer to section 22.4, Endians.

Bit 8—DMA Transfer End Status (DMAST): Indicates the DMA transfer end status.

| Bit 8: DMAST | Description |
|--------------|---|
| 0 | Normal termination (Initial value) |
| 1 | Abnormal termination (Error detection or forced DMA transfer termination) |

Bit 7—DMA Transfer Termination Interrupt Mask (DMAIM): Specifies the DMA transfer termination interrupt mask.

| Bit 7: DMAIM | Description |
|--------------|------------------------------------|
| 0 | Interrupt disabled (Initial value) |
| 1 | Interrupt enabled |

Bit 6—DMA Transfer Termination Interrupt Status (DMAIS): Indicates the DMA transfer termination interrupt status. The interrupt status is set even when the interrupt mask is set.

| Bit 6: DMAIS | Description | |
|--------------|-------------|--|
| When writing | 0 | Ignored |
| | 1 | Status clear |
| When reading | 0 | Interrupt not detected (Initial value) |
| | 1 | Interrupt detected |

Bit 5—Local Address Control (LAHOLD): Local address control during DMA transfer

| Bit 5: LAHOLD | Description |
|---------------|--|
| 0 | Incremented (Initial value) |
| 1 | High address fixed (Address A[4:0] is incremented) |

Bit 4—Reserved: This bit always returns 0 when read. Always write 0 to this bit.

Bit 3—PCI Address Space Type (IOSEL): Type of PCI address space during transfer

| Bit 3: IOSEL | Description |
|--------------|------------------------------|
| 0 | Memory space (Initial value) |
| 1 | I/O space |

Bit 2—Transfer Direction (DIR): Transfer direction during DMA transfer

| Bit 2: DIR | Description |
|------------|---|
| 0 | Transfer from PCI bus to local bus (SH bus) (Initial value) |
| 1 | Transfer from local bus (SH bus) to PCI bus |

Bit 1—Forced DMA Transfer Termination (DMASTOP): Forced termination of DMA transfer

| Bit 1: DMASTOP | Description | |
|----------------|-------------|--|
| When writing | 0 | Writing of 0 is ignored. |
| | 1 | Forced termination of DMA transfer |
| When reading | | When DMA transfer stops due to forced DMA transfer termination, 1 is set |

Bit 0—DMA Transfer Start Control (DMASTRT): Controls the starting of DMA transfer.

| Bit 0: DMASTRT | Description | |
|----------------|-------------|---------------------------------|
| When writing | 0 | Ignored |
| | 1 | Start |
| When reading | 0 | End of transfer (Initial value) |
| | 1 | Busy (in transfer) |

22.2.32 PIO Address Register (PCIPAR)

| | | | | | | | | |
|----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | CFGEN | — | — | — | — | — | — | — |
| Initial value: | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | BUSNO23 | BUSNO22 | BUSNO21 | BUSNO20 | BUSNO19 | BUSNO18 | BUSNO17 | BUSNO16 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | DEVNO15 | DEVNO14 | DEVNO13 | DEVNO12 | DEVNO11 | FNCNO10 | FNCNO9 | FNCNO8 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-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 |
| | REGADR7 | REGADR6 | REGADR5 | REGADR4 | REGADR3 | REGADR2 | — | — |
| Initial value: | — | — | — | — | — | — | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R | R |

The PIO address register (PCIPAR) is used when issuing configuration cycles on the PCI bus when the PCIC is host. The PCIC supports the configuration mechanism 1 stipulated in the PCI local bus specifications. This register is equivalent to the configuration register of configuration mechanism 1. This register is equivalent to the CONFIG_ADDRESS of configuration mechanism 1. The check that the issuance of the PCI configuration cycle is enabled, and access the PCI configuration space, this register contains the PCI bus No., device No., Function No., and LW (longword) boundary of the configuration register. This 32-bit read/write register can be accessed from the PP bus.

Bit 31 (CFGEN) is set in hardware and none of the other bits of the PCIPAR register are initialized at a power-on reset or software reset.

Always write to this register prior to accessing the PCI configuration space. After setting a value in this register, generate the configuration cycle by reading or writing to the PIO data register (PCIPDR).

Also, a special cycle is issued by setting H'8000FF00 in this register and writing to the PCIPDR.

Bit 31—Configuration Cycle Generate Enable (CFGEN): Indicates the configuration cycle generation enable.

Bits 30 to 24—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bits 23 to 16—PCI Bus No. (BUSNO): These bits specify the No. of the PCI bus subject to configuration access. Bus No. D indicates the bus connected with the PCIC. The bus No. is expressed with 8 bits, and its maximum value is 255.

Bits 15 to 11—Device No. (DEVNO): These bits specify the No. of the device subject to configuration access. The device No. is expressed with 5 bits, and takes a value from bits 0 to 31. In place of IDSEL, one of bits 31 to 16 of the A/D line, corresponding to the device No. set in this field, is driven to “1”. The following table shows the relationship between the device No. and IDSEL (A/D [31 to 16]). When the device No. is 10h or greater, A/D [31 to 16]) are all zeros.

| DEVNO | IDSEL | DEVNO | IDSEL | DEVNO | IDSEL | DEVNO | IDSEL |
|-------|------------|-------|------------|-------|------------|-------|------------|
| H'0 | AD[16] = 1 | H'4 | AD[20] = 1 | H'8 | AD[24] = 1 | H'C | AD[28] = 1 |
| H'1 | AD[17] = 1 | H'5 | AD[21] = 1 | H'9 | AD[25] = 1 | H'D | AD[29] = 1 |
| H'2 | AD[18] = 1 | H'6 | AD[22] = 1 | H'A | AD[26] = 1 | H'E | AD[30] = 1 |
| H'3 | AD[19] = 1 | H'7 | AD[23] = 1 | H'B | AD[27] = 1 | H'F | AD[31] = 1 |

Bits 10 to 8: Function No. (FNCNO): These bits specify the No. of the function subject to configuration access. The function No. is expressed with 3 bits, and takes a value of 0 to 7.

Bits 7 to 2—Configuration Register Address (REGADR): These bits set the register subject to configuration access with a longword boundary.

Bits 1 and 0—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

22.2.33 Memory Space Base Register (PCIMBR)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | MBR31 | MBR30 | MBR29 | MBR28 | MBR27 | MBR26 | MBR25 | MBR24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | LOCK |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R/W |

The memory space base register (PCIMBR) specifies the most significant 8 bits of the address of the PCI memory space when performing a memory read/write operation using PIO transfers. It also specifies locked transfers. This 32-bit read/write register can be accessed from the PP bus.

All bits of the PCIMBR register are initialized to 0 at a power-on reset. They are not initialized at a software reset.

Setting bit 0 (LOCK) to 1 locks the memory space for PIO transfers while the bit remains set. A locked transfer consists of the combined read and write operations. Do not attempt to perform other PIO transfers during the locked combination of read and write operations.

Always write to this register prior to performing memory read/write operations by PIO transfer.

Bits 31 to 24—Memory Space Base Address (MBR31 to 24): Sets the base address for the PCI memory space in PIO transfers. (Initial value is undefined.)

Bits 23 to 1—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 0—Lock Transfer (LOCK): Specifies the locking of the memory space during PIO transfer.

| Bit 0: LOCK | Description |
|--------------------|----------------------------|
| 0 | Not locked (Initial value) |
| 1 | Locked |

22.2.34 I/O Space Base Register (PCIIOBR)

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| | IOBR31 | IOBR30 | IOBR29 | IOBR28 | IOBR27 | IOBR26 | IOBR25 | IOBR24 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | IOBR23 | IOBR22 | IOBR21 | IOBR20 | IOBR19 | IOBR18 | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | LOCK |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R/W |

The I/O space base register (PCIIOBR) specifies the most significant 14 bits of the address of the PCI I/O space when performing I/O read and I/O write operations by PIO transfer. It also specifies locked transfers. This 32-bit read/write register can be accessed from the PP bus.

All bits of the PCIIOBR register are initialized to 0 at a power-on reset. They are not initialized at a software reset.

Setting bit 0 (LOCK) to 1 locks the I/O space for PIO transfers while the bit remains set. A locked transfer consists of the combined read and write operations. Do not attempt to perform other PIO transfers during the locked combination of read and write operations.

Always write to this register prior to I/O space read and I/O space write operations by PIO transfer.

Bits 31 to 18—I/O Space Base Address (IOBR31 to 18): Sets the base register for the PCI I/O space in PIO transfers.

Bits 17 to 1—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 0—Lock Transfer (LOCK): Specifies the locking of the I/O space during PIO transfer.

| Bit 0: LOCK | Description |
|-------------|----------------------------|
| 0 | Not locked (Initial value) |
| 1 | Locked |

22.2.35 PCI Power Management Interrupt Register (PCIPINT)

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|--------------|--------------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | ... | — | — | — | — |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | PWRST_ D3 | PWRST_ D0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R/WC | R/WC |

Note: Cleared by setting WC: 1. (Writing of 0 is ignored.)

The PCI power management interrupt register (PCIPINT) controls the power management interrupts. It provides the interrupt bits for a transition to the power state D3 (power down mode) and recovery to the power state D0 (normal state). This 32-bit read/write register can be accessed from the PP bus.

The PCIPINT register is initialized to H'00000000 at a power-on reset. It is not initialized at a software reset. When an interrupt is detected, the bit corresponding to the content of that interrupt is set to 1. Each interrupt detection bit can be cleared to 0 by writing 1 to it (write clear).

The power state D0 interrupt is not generated at a power-on reset.

Bits 31 to 2—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 1—Power state D3 (PWRST_D3): Transition request to power-down mode interrupt for this LSI.

Bit 0—Power state D0 (PWRST_D0): Restore from power-down mode interrupt for this LSI.

Note: The power states D3, D0 are not masked even when the interrupt mask bit is set ON.

22.2.36 PCI Power Management Interrupt Mask Register (PCIPINTM)

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|--------------|--------------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | ... | — | — | — | — |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | DPERR_ WT | DPERR_ RD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R/W | R/W |

The PCI power management interrupt mask register (PCIPINTM) sets the interrupt mask for the power management interrupts. This 32-bit read/write register can be accessed from the PP bus.

The PCIPINTM register is initialized to H'00000000 at a power-on reset. It is not initialized at a software reset.

Interrupt masks can be set for both the interrupt for a transition to the power state D3 (power down mode) and recovery to the power state D0 (normal status). Setting the respective bit to 0 disables the interrupt and setting it to 1 enables the interrupt.

Bits 31 to 2—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 1—Power State D3 (DPERR_WT): Transition request to power-down mode interrupt mask for this LSI.

Bit 0—Power State D0 (DPERR_RD): Restore from power-down mode interrupt mask for this LSI.

22.2.37 PCI Clock Control Register (PCICLKR)

| | | | | | | | | |
|----------------|----|----|----|-----|----|----|----------------|--------------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | ... | — | — | — | — |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | PCICLKS TOP | BCLKST OP |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R/W | R/W |

The PCI clock control register (PCICLKR) controls the stopping of the local bus clock (BCLK) in the PCIC and the PCI bus clock. This 32-bit read/write register can be accessed from the PP bus.

The PCICLKR register is initialized to H'00000000 at a power-on reset. It is not initialized at a software reset.

When the PCI bus clock is input from the external input pin $\overline{\text{PCICLK}}$, the PCI bus clock can be stopped by setting the PCICLKSTOP bit to 1. Likewise, the local bus clock can be stopped by setting the BCLKSTOP bit to 1.

When the PCI bus clock is input via the CKIO pin, setting BCLKSTOP to 1 stops both the Bck in the PCIC and the feedback input clock from CKIO.

Writing to this register is valid only when bits 31 to 24 are H'A5.

Bits 31 to 2—Reserved: These bits are always read as 0. When writing, always write H'A5 to bits 31 to 24, and 0 to the other bits. Always write 0 to these bits when writing.

Bit 1—PCICLK Stop Control (PCICLKSTOP): Controls the stopping of the clock input via the PCICLK pin.

| Bit 1: PCICLKSTOP | Description | |
|-------------------|--------------------------------|----------|
| 0 | PCICLK input enabled value) | (Initial |
| 1 | Stop PCICLK input | |

Bit 0—BCLK Stop Control (BCLKSTOP): Controls the stopping of the Bck input clock and CKIO input clock in the PCIC.

| Bit 0: BCLKSTOP | Description | |
|-----------------|-----------------------------|----------|
| 0 | Bck input enabled value) | (Initial |
| 1 | Stop Bck input | |

22.2.38 PCIC-BSC Registers

PCIC Bus Control Register 1 (PCIBCR1)

PCIC Bus Control Register 2 (PCIBCR2)

PCIC Bus Control Register 3 (PCIBCR3)*¹

PCIC Wait Control Register 1 (PCIWCR1)

PCIC Wait Control Register 2 (PCIWCR2)

PCIC Wait Control register 3 (PCIWCR3)

PCIC Discrete Memory Control Register (PCIMCR)

Because PCI bus data is stored, in the PCIC, in memory on the local bus, the PCIC is equipped with an internal bus controller (PCIC-BSC). The PCIC-BSC performs the same type of control as the slave function of the bus controller (BSC). However, the PCIC-BSC returns bus rights to the BSC after each data transfer of up to 32 bytes of data. There are six registers in the PCIC-BSC: PCIBCR1 (equivalent to the BCR1 of the BSC), PCIBCR2 (equivalent to the BCR2 of the BSC), PCIBCR3 (equivalent to the BCR3 of the BSC)*¹, PCIWCR1 (equivalent to the WCR1 of the BSC), PCIWCR2 (equivalent to the WCR2 of the BSC), PCIWCR3 (equivalent to the WCR3 of the BSC), and PCIMCR (equivalent to the MCR of the BSC). Each is a 32-bit register. BCR2 and BCR3 are 16-bit registers, but PCIBCR2 and PCIBCR3 should be accessed by longword access. The low 16 bits of PCIBCR2 and PCIBCR3 corresponds to the 16 bits of these registers, respectively. See section 13, Bus State Controller (BSC), for details of the initial values, etc.

- The PCIC-BSC performs the same operations as the slave mode of the BSC. Therefore, the MATER bit of the PCI bus control register 1 (PCIBCR1) shows the slave status.
- Because the PCIC-BSC operates in slave mode, the bus privilege is handed to the BSC once per bus cycle.
- The external memory capable of data transfers to the PCI bus is SRAM, DRAM, synchronous DRAM, and MPX*².
- The memory data width is 32-bit or 16-bit only (only 32-bit in the case of synchronous DRAM).
- Do not specify other external memory types (burst ROM, MPX, byte control SRAM or PCMCIA) as the external memory for data transfers with the PCI bus.
- Because the PCIC-BSC operates in slave mode, the RAS-down mode of DRAM and SDRAM is not available.
- The local bus supports both big and little endian. However, the PCI bus supports only little endian.

The PCI-BSC does not support mode register setting of synchronous DRAM nor refreshing of synchronous DRAM or DRAM. These must be executed by the BSC.

Also, do not implement any settings that are not allowed in slave mode in the PCIC-BSC registers. This is because bit 30: master/slave flag (MASTER) of the PCIBCR1 is fixed Low, regardless of the value of the external master/slave setting pin (MD7) at a power-on reset, and the PCIC-BSC therefore is set in slave mode.

In the case of external memory not used for data transfers with the PCI bus, make the same settings as the corresponding bus state controller register.

These registers are initialized at a power-on reset, but not by a software reset.

- Notes:
1. This register is provided only in the SH7751R, not provided in the SH7751.
 2. MPX is supported only in the SH7751R, not supported in the SH7751.

22.2.39 Port Control Register (PCIPCTR)

| | | | | | | | | |
|----------------|----|----|--------|-------|--------|---------|---------|---------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | PORT2EN | PORT1EN | PORT0EN |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | PB2PUP | PB2IO | PB1PUP | PB1IO | PB0PUP | PB0IO |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

The port control register (PCIPCTR) selects whether to enable or disable port function allocation for pins for unwanted PCI bus arbitration when the PCIC is used in non-host mode. It also specifies the swithing ON/OFF of pin pull-up resistances and between input and output. This 32-bit read/write register can be accessed from the PP bus.

The PCIPCTR register is initialized to H'00000000 at a power-on reset. It is not initialized at a software reset.

When the PCIC is operating as host, the port function cannot be used if the arbitration function is enabled.

Bits 31 to 19—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 18—Port 2 Enable (PORT2EN): Provides the enable control for the port 2.

| Bit 18: PORT2EN | Description |
|-----------------|---|
| 0 | Do not use pins $\overline{\text{PCIGNT4}}$ or $\overline{\text{PCIREQ4}}$ as ports (Initial value) |
| 1 | Use pins $\overline{\text{PCIGNT4}}$ or $\overline{\text{PCIREQ4}}$ as ports |

Bit 17—Port 1 Enable (PORT1EN): Provides the enable control for the port 1.

| Bit 17: PORT1EN | Description |
|-----------------|---|
| 0 | Do not use pins $\overline{\text{PCIGNT3}}$ or $\overline{\text{PCIREQ3}}$ as ports (Initial value) |
| 1 | Use pins $\overline{\text{PCIGNT3}}$ or $\overline{\text{PCIREQ3}}$ as ports |

Bit 16—Port 0 Enable (PORT0EN): Provides the enable control for the port 0.

| Bit 16: PORT0EN | Description |
|-----------------|---|
| 0 | Do not use pins $\overline{\text{PCIGNT2}}$ or $\overline{\text{PCIREQ2}}$ as ports (Initial value) |
| 1 | Use pins $\overline{\text{PCIGNT2}}$ or $\overline{\text{PCIREQ2}}$ as ports |

Bits 15 to 6—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 5—Port 2 Pull-up Resistance Control (PB2PUP): Controls pull-up resistance when $\overline{\text{PCIREQ4}}$ pin is used as port.

| Bit 5: PB2PUP | Description |
|---------------|---|
| 0 | Pull-up $\overline{\text{PCIREQ4}}$ pin (Initial value) |
| 1 | Do not pull-up $\overline{\text{PCIREQ4}}$ pin |

Bit 4—Port 2 Input/Output Control (PB2IO): Controls input or output when $\overline{\text{PCIREQ4}}$ is used as a port.

| Bit 4: PB2IO | Description |
|--------------|---|
| 0 | Set $\overline{\text{PCIREQ4}}$ pin for input (Initial value) |
| 1 | Set $\overline{\text{PCIREQ4}}$ pin for output |

Bit 3—Port 1 Pull-up Resistance Control (PB1PUP): Controls pull-up resistance when $\overline{\text{PCIREQ3}}$ pin is used as port.

| Bit 3: PB1PUP | Description |
|---------------|---|
| 0 | Pull-up $\overline{\text{PCIREQ3}}$ pin (Initial value) |
| 1 | Do not pull-up $\overline{\text{PCIREQ3}}$ pin |

Bit 2—Port 1 Input/Output Control (PB1IO): Controls input or output when $\overline{\text{PCIREQ3}}$ is used as a port.

| Bit 2: PB1IO | Description |
|--------------|---|
| 0 | Set $\overline{\text{PCIREQ3}}$ pin for input (Initial value) |
| 1 | Set $\overline{\text{PCIREQ3}}$ pin for output |

Bit 1—Port 0 Pull-up Resistance Control (PB0PUP): Controls pull-up resistance when $\overline{\text{PCIREQ2}}$ pin is used as port.

| Bit 1: PB0PUP | Description |
|---------------|---|
| 0 | Pull-up $\overline{\text{PCIREQ2}}$ pin (Initial value) |
| 1 | Do not pull-up $\overline{\text{PCIREQ2}}$ pin |

Bit 0—Port 0 Input/Output Control (PB0IO): Controls input or output when $\overline{\text{PCIREQ2}}$ is used as a port.

| Bit 0: PB0IO | Description |
|--------------|---|
| 0 | Set $\overline{\text{PCIREQ2}}$ pin for input (Initial value) |
| 1 | Set $\overline{\text{PCIREQ2}}$ pin for output |

22.2.40 Port Data Register (PCIPDTR)

| | | | | | | | | |
|----------------|----|----|-------|-------|-------|-------|-------|-------|
| Bit: | 31 | 30 | 29 | ... | 11 | 10 | 9 | 8 |
| | — | — | — | ... | — | — | — | — |
| Initial value: | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | ... | — | — | — | — |
| PP Bus-R/W: | R | R | R | ... | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | PB5DT | PB4DT | PB3DT | PB2DT | PB1DT | PB0DT |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

The port data register (PCIPDTR) inputs and outputs the port data when allocation of the port function to the unwanted PCI bus arbitration pins is enabled when the PCIC is operating in non-host mode. This 32-bit read/write register can be accessed from the PP bus.

The PCIPDTR register is initialized to H'00000000 at a power-on reset. It is not initialized at a software reset.

Data is output in sync with the local bus clock. Input data is fetched at the rising edge of the local bus clock.

Bits 31 to 6—Reserved: These bits always return 0 when read. Always write 0 to these bits when writing.

Bit 5—Port 2 Output Data (PB5DT): Output data when $\overline{\text{PCIGNT4}}$ pin is used as port. ($\overline{\text{PCIGNT4}}$ pin is output-only.)

Bit 4—Port 2 Input/Output Data (PB4DT): Receives input data and sets output data when the $\overline{\text{PCIREQ4}}$ pin is used as a port.

Bit 3—Port 1 Output Data (PB3DT): Output data when $\overline{\text{PCIGNT3}}$ pin is used as port. ($\overline{\text{PCIGNT3}}$ pin is output-only.)

Bit 2—Port 1 Input/Output Data (PB2DT): Receives input data and sets output data when the $\overline{\text{PCIREQ3}}$ pin is used as a port.

Bit 1—Port 0 Output Data (PB1DT): Output data when $\overline{\text{PCIGNT2}}$ pin is used as port.
($\overline{\text{PCIGNT2}}$ pin is output-only.)

Bit 0—Port 0 Input/Output Data (PB0DT): Receives input data and sets output data when the $\overline{\text{PCIREQ2}}$ pin is used as a port.

22.2.41 PIO Data Register (PCIPDR)

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | PPDA31 | PPDA30 | PPDA29 | PPDA28 | PPDA27 | PPDA26 | PPDA25 | PPDA24 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | PPDA23 | PPDA22 | PPDA21 | PPDA20 | PPDA19 | PPDA18 | PPDA17 | PPDA16 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PPDA15 | PPDA14 | PPDA13 | PPDA12 | PPDA11 | PPDA10 | PPDA9 | PPDA8 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-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 |
| | PPDA7 | PPDA6 | PPDA5 | PPDA4 | PPDA3 | PPDA2 | PPDA1 | PPDA0 |
| Initial value: | — | — | — | — | — | — | — | — |
| PCI-R/W: | — | — | — | — | — | — | — | — |
| PP Bus-R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The PIO data register (PCIPDR) sets the data for read/write in the PCI configuration cycle. This 32-bit read/write register can be accessed from the PP bus.

The PCIPDR register is not initialized at a power-on reset or software reset. The initial value is undefined.

Always write to this register before accessing the PCI configuration space. Always read/write to this register after setting the value in the PIO address register (PCIPAR).

The configuration cycle on the PCI bus can be generated by reading/writing to this register.

Bits 31 to 0—PIO Configuration Data (PPDA31 to 0): Read/write register for configuration data in PIO transfers.

The configuration cycle on the PCI bus can be generated by reading/writing to this register.

22.3 Description of Operation

22.3.1 Operating Modes

The external mode pins (MD9 and MD10) select whether the PCIC operates as the host on the PCI bus and also select the bus clock for the PCI bus. The mode selection signals input via the external mode pins are fetched on negation of a power-on reset.

Table 22.8 Operating Modes

| MD9 | MD10 | Operating Modes |
|-----|------|--|
| 0 | 0 | The PCIC host functions are enabled and the external input via the PCICLK pin is the operating clock for the PCI bus |
| | 1 | The PCIC host functions are enabled and this LSI bus clock (feedback input clock from CKIO pin) is the operating clock for the PCI bus |
| 1 | 0 | The PCIC host functions are disabled (non-host) and the input clock from the PCICLK pin is selected as the clock for the PCI bus |
| | 1 | PCIC-disabled mode. In this mode, PCIC operation is disabled |

Note: In PCIC-disabled mode, do not attempt to access the PCIC local registers.

In this section, the clock resulting from the above mode switching is known as the PCI bus clock.

22.3.2 PCI Commands

Table 22.9 lists the PCI commands and shows the PCIC support.

Table 22.9 PCI Command Support

| Command | Host Operation | | Non-Host Operation | | Remarks |
|-----------------------------|----------------|--------|--------------------|--------|---|
| | Master | Target | Master | Target | |
| Memory read | O | O | O | O | |
| Memory read line | X | Δ | X | Δ | When the target, operates as memory read |
| Memory read multiple | X | Δ | X | Δ | When the target, operates as memory read |
| Memory write | O | O | O | O | |
| Memory write and invalidate | X | Δ | X | Δ | When the target, operates as memory write |
| I/O read | O | O | O | O | |
| I/O write | O | O | O | O | |
| Configuration read | O | — | — | O | |
| Configuration write | O | — | — | O | |
| Interrupt acknowledge cycle | X | X | X | X | |
| Special cycle | O | — | — | X | |
| Dual address cycle | X | X | X | X | |

Legend:

O: Supported

Δ: Limited support

X, —: Not issued by PCIC or no response from PCIC

When PCIC Operates as Master: The PCIC supports the memory read command, memory write command, I/O read command, and I/O write command. When the host functions are enabled, the configuration command and special cycle can also be used.

When PCIC Operates as Target: The PCIC receives the memory read command, memory write command, I/O read command, and I/O write command. The memory read line command and memory read multiple command function as memory reads, while the memory write invalidate command functions as a memory write. When operating in non-host mode, the PCIC accepts the configuration command.

22.3.3 PCIC Initialization

After a power-on reset, the configuration register initialization bit (CFINIT) of the PCI control register (PCICR) is cleared. At this point, if the PCIC is operating as the PCI bus host, the bus privileges are permanently granted to the PCIC, and no device arbitration is performed on the PCI bus. When the PCIC is not operating as host, retries are returned without accepting access from PCI devices connected to the PCI bus.

The PCIC's internal configuration registers and local registers must be initialized while the CFINIT bit is cleared to 0. On completion of initialization, set the CFINIT bit to 1. When operating as host, arbitration is enabled; when operating as non-host, the PCIC can be accessed from the PCI bus.

Regardless of whether or not the PCIC is operating as host, external PCI devices cannot be accessed from the PCIC while the CFINIT bit is cleared. If the PCIC's internal configuration registers and local registers are initialized correctly, the PCIC will operate correctly. However, we recommend first setting the CFINIT bit to 1.

When the PCIC is operating as the host, arbitration is enabled. When operating as non-host, the PCIC can be accessed from the PCI bus.

Regardless of whether the PCIC is operating as the host or non-host, external PCI devices cannot be accessed from the PCIC while the CFINIT bit is being cleared. Set the CFINIT bit to 1 before accessing an external PCIC device.

Be sure to initialize the following 13 registers while the CFINIT bit is being cleared: configuration registers 1, 2, 11 (PCICONF1, 2, 11) for PCI, local space registers 0, 1 (PCILSR0, 1) for PCI, local address registers 0, 1 (PCILAR0, 1) for PCI, PCI bus control registers 1, 2 (PCIBCR1, 2) for PCIC-BSC, PCI weight control registers 1, 2, 3 (PCIWCR1, 2, 3), and PCI-specific memory control register (PCIMCR). Since the PCIC-BSC is fixed in sleep mode at a power-on reset regardless of the value of the external pin (MD7) for master/slave designation, do not make a PCIC-BSC register setting that is prohibited in the sleep mode.

Also, as the BSC has BCR1.BREQEN bits that enable an external request and a bus request from the PCIC to be accepted, BCR1.BREQEN should be set to 1 when the PCIC is used.

While 1 is being set in the CFINIT bit, the registers for the PCIC-BSC (PCIBCR1, 2, PCIWCR1, 2, 3, PCIMCR) cannot be written to. The data transfer accuracy between the PCI bus and local bus cannot be guaranteed if an attempt is made to write to any of these registers during this period.

22.3.4 Local Register Access

Only longword (32-bit) access of the PCIC's internal local registers and configuration registers from the CPU is supported.

(It is possible to use PIO transfers to perform byte, word, and longword access of the memory space and I/O space on the PCI bus.)

If an attempt is made to access these registers using other than the prescribed access size, zero is returned when reading and writing is ignored. The same is true if you attempt to access the reserved areas in the register area in the PCIC.

Some of the configuration registers and local registers can be accessed both from the CPU and from the PCI device(s). Therefore, arbitration is performed for both types of access and either the CPU or PCI device access made to wait according to the access timing.

In the read bus cycle from the CPU, the internal bus cycle for the peripheral module is made to wait until the data is actually ready. In the write bus cycle, the bus cycle of the internal bus for peripheral modules ends with the data having been written to the interface (register located immediately after the PCIC input) register on the internal bus for peripheral modules, but the data is not actually written to the local register(s) or PCI bus until the following clock cycle. If it is necessary to check that the data has actually been written, read the register to which the data was to have been written. This is because the read cycle must be after the write cycle has completed.

When accessing from a PCI device, the PCI bus cycle is caused to wait until the read or write operation has actually completed.

The internal bus for peripheral modules used for read/write operations from the CPU operates only with big endians.

22.3.5 Host Functions

The PCIC has the following PCI bus host functions (host devices):

- Inter-PCI device arbitration function
- Configuration register access function
- Special cycle generation function
- Reset output function
- Clock output function

Inter-PCI Device Arbitration: The PCI bus arbitration circuit in the PCIC can be used when the PCIC is operating as the host device. The arbitration circuit can be connected to up to four external PCI devices (devices that can operate as master devices) that request bus privileges.

If multiple bus privilege requests are made simultaneously by the PCI devices, the bus privilege is granted in the predetermined order of priority. There are two orders of priority: fixed, and pseudo round robin. The mode is selected by setting the bus master arbitration mode control bit (BMABT) of the PCI control register (PCICR).

- Priority-fixed mode (BMABT = 0)

In priority-fixed mode, the priority order of bus privilege requests is fixed and cannot be changed. The order is as follows:

PCIC (device 0) > device 1 > device 2 > device 3 > device 4

That is, the PCIC has the highest order of priority and device 4 has the lowest. When bus privilege requests occur simultaneously, the device with the highest order of priority takes precedence. Here, device 1 is the PCI device using bus privilege request pins $\overline{\text{PCIREQ1}}$ and $\overline{\text{PCIGNT1}}$, device 2 uses $\overline{\text{PCIREQ2}}$ and $\overline{\text{PCIGNT2}}$, device 3 uses $\overline{\text{PCIREQ3}}$ and $\overline{\text{PCIGNT3}}$, and device 4 uses $\overline{\text{PCIREQ4}}$ and $\overline{\text{PCIGNT4}}$. When the PCIC is operating as the host device, no bus privilege request signals are output from the PCIC to the PCI bus arbitration circuit.

- Pseudo round-robin mode (BMABT = 1)

In pseudo round-robin mode, when a device takes the bus privilege, the priority order of that device becomes lowest.

In the initial state, the priority order is set to the same as in the fixed mode. Here, device 1 outputs a bus privilege request, after which the priority order changes to ...

PCIC > device 2 > device 3 > device 4 > device 1.

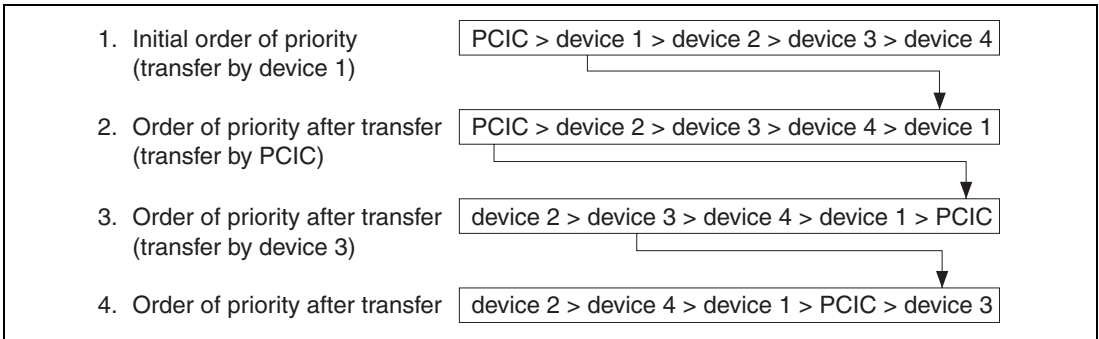
If the PCIC then outputs a bus privilege request and takes the bus privilege, the priority order changes to ...

Device 2 > device 3 > device 4 > device 1 > PCIC.

Likewise, if device 3 outputs a bus privilege request and takes the bus privilege, the priority order becomes ...

Device 2 > device 4 > device 1 > PCIC > device 3.

In this way, the priority order of the master device that takes the bus privilege always changes to lowest after the data transfer is completed.



When the PCIC is operating as the host device, the PCIC performs the PCI bus parking (bus drive when not in use).

When 3 or fewer master devices are connected, set the level of the unused pins of $\overline{\text{PCIREQ}}[4:1]$ high.

In non-host mode, the PCI bus arbitration function of the PCIC is disabled. PCI bus arbitration is performed according to the specifications of the connected PCI bus arbiter. For details, see section 22.3.6, PCI Bus Arbitration in Non-host Mode.

Configuration Register Access: The configuration register of external PCI devices can be accessed when the PCIC is operating as the host device. The PIO address register (PCIPAR) and PIO data register (PCIPDR) are used to generate a configuration read/write transfer for accessing the configuration register.

The PCIC supports the configuration mechanism stipulated in the PCI local bus spec.

First, specify in the PCIPAR the address of the configuration register of the external PCI device to be accessed. See section 22.2, PCIC Register Descriptions, for how to set the PCIPAR.

Next, read data from the PCIPDR or write data to the PCIPDR. Only longword (32-bit) access of the PCIPDR is supported.

Special Cycle Generation: When the PCIC operates as the host device, a special cycle is generated by setting H'8000FF00 in the PCIPAR and writing to the PCIPDR.

Reset Output: When the PCIC is operating as the host device, $\overline{\text{PCIRST}}$ can be used to reset the PCI bus. See section 22.5, Resetting, for details of $\overline{\text{PCIRST}}$.

Clock Output: When the PCIC is operating as the host device and the bus clock (CKIO pin) is selected as the PCI bus clock, not only does the PCIC's PCI bus clock operate using the CKIO clock but the CKIO clock can also be used as the PCI bus clock. Thus, there is no requirement for an external PCI clock oscillation circuit.

When using the CKIO clock, please note the limitations on CKIO clock frequency, stability, and load capacitance that can be connected to the CKIO pin. Check the clock oscillation circuit and electrical characteristics in section 10, Clock Oscillation Circuits, and section 23, Electrical Characteristics.

22.3.6 PCI Bus Arbitration in Non-host Mode

When operating in non-host mode, the PCI bus arbitration function in the PCIC is disabled and PCI bus arbitration is performed according to the specifications of the externally connected PCI bus arbiter.

In this case, the PCIC must request PCI bus privileges from the PCI bus arbiter (system host device). The $\overline{\text{PCIGNT1/REQOUT}}$ pins are used for the bus request signals, and the $\overline{\text{PCIREQ1/GNTIN}}$ pins are used for the bus grant signals. When the bus grant signals are asserted when the bus request signals are not asserted, the PCIC performs bus parking.

Also, when the PCIC is used as a target device that does not request bus privileges, the $\overline{\text{PCIREQ1/GNTIN}}$ pins must be fixed at the high level.

22.3.7 PIO Transfers

PIO transfer is a data transfer mode in which a peripheral bus is used to access the memory space and I/O space of the PCI bus.

The following commands are supported in PIO transfer mode:

- Memory read, memory write, I/O read, and I/O write
- Locked transfer (High-speed back-to-back transfers are not supported.)

In PIO transfer mode, only single transfers are supported. 32-byte burst transfers are not supported.

In memory transfers and I/O transfers, the supported, so generate byte enable signals ($\overline{\text{BE[3:0]}}$) to match the respective access sizes and output these signals to the PCI bus. Access sizes are byte, word, and longword.

Locked transfers are supported only in the case of memory transfers and I/O transfers. High-speed back-to-back transfers are not supported.

Memory Transfers: This section describes how PIO transfers are used to access memory space. 16MB between H'FD000000 and H'FDFFFFFF of area P4 (H'1D000000 to H'1DFFFFFF in area 7) is allocated as PCI memory address space. This space is used as the least significant 24 bits of the PCI address. However, in memory transfers, the two low bits of the PCI address are ignored, and B'00 is output to the PCI bus. The most significant 8 bits (MBR [31:24]) of the memory space base register (PCIMBR) are used as the most significant bits of the PCI address. These two addresses are combined to specify a 32-bit PCI address.

To transfer to the memory space, first specify the most significant 8 bits of the PCI address in the PCIMBR, then access the PCI memory address space. If within the 16MB space, the PCI memory address space can be consecutively accessed simply by setting the PCIMBR once. If it is necessary to access an address space over the 16MB, set PCIMBR again.

When performing locked transfers in memory transfer mode, set the PCIMBR memory space lock specification bit (LOCK). While the LOCK bit is set, the memory space is locked.

Note the following when performing LOCK transfers:

- A LOCK transfer consists of one read transfer and one write transfer. Always start with the read transfer. The system will operate correctly if you start with a write transfer, but the resource LOCK will not be established. Also, the system will operate correctly if you perform two LOCK read transfers, but the LOCK will be released at the next LOCK write transfer.
- The minimum resource for which the LOCK is guaranteed is a 16-byte block. However, the system will operate correctly even if LOCK transfers are made to addresses other than where the LOCK is established.
- You cannot access other targets while a target is LOCKed (from the LOCK read until the LOCK write).
 - PIO LOCK access of another target ends normally and transfers on the PCI bus are also generated.
 - Unlocked PIO transfer requests invoked between a LOCK read and LOCK write end normally, but no transfers are generated on the PCI bus.
 - DMA transfers are postponed until the LOCK transfer ends.

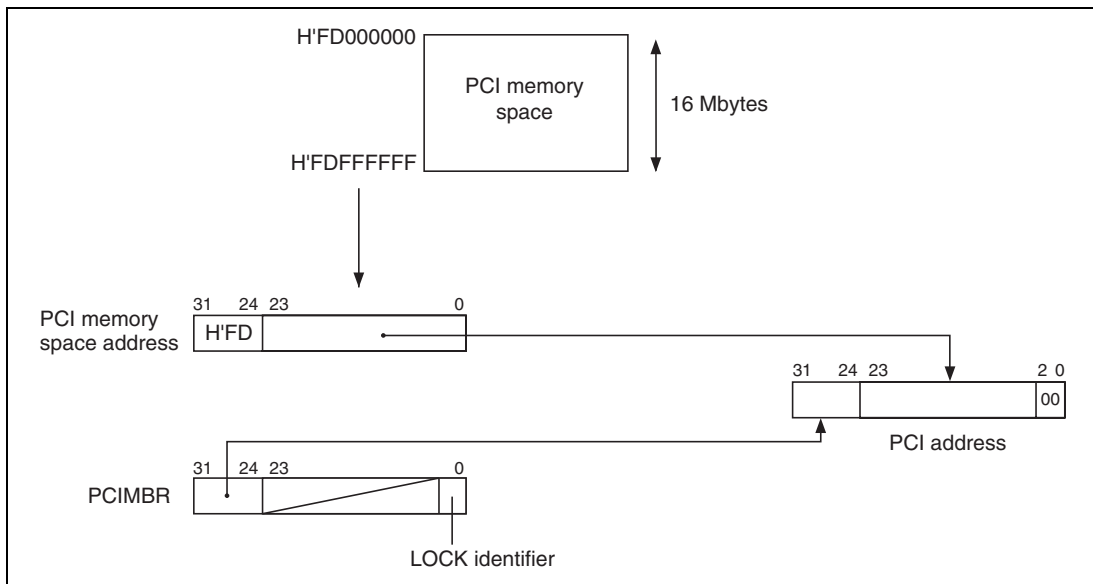


Figure 22.2 PIO Memory Space Access

I/O Transfers: This section describes how to access I/O space using PIO transfers. The 256KB from H'FE240000 to H'FE27FFFF of area P4 (H'1E240000 to H'1E27FFFF in area 7) is allocated as PCI I/O address space. This space is used for the least significant 18 bits of the PCI address. The most significant 14 bits (IOBR [31:18]) of the I/O space base register (PCIIOBR) are used as the most significant 14 bits of the PCI address. These two addresses are combined to specify the 32-bit PCI address.

For transfers to the I/O space, first specify the most significant 14 bits of the PCI address in PCIIOBR, then access the PCI I/O address space. If within the 256KB space, you can access the PCI I/O address space consecutively simply by setting the PCIIOBR once. If it is necessary to access another address space beyond 256KB, set PCIIOBR again.

When performing locked transfers in I/O transfers, set the I/O space lock specification bit (LOCK) in the PCIIOBR. The I/O space is locked while the LOCK bit is set. The same precautions apply to LOCK I/O transfers as to LOCK memory transfers.

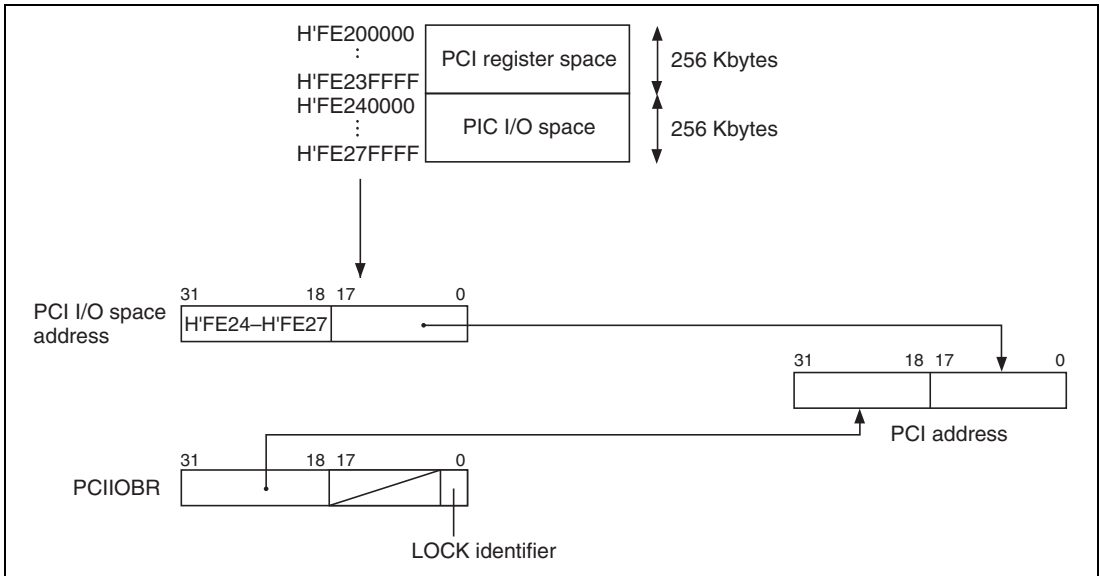


Figure 22.3 PIO I/O Space Access

PIO Transfer Error: An error on the PCI bus that occurs in a transfer during a PIO write operation is not detected. When an error is generated during a PIO read operation, the PIO transfer is forcibly terminated to prevent effects on the DMA transfer and target transfer. However, accuracy of the read data is not guaranteed.

22.3.8 Target Transfers

The following commands are available for transferring data in target transfers.

- Memory read and memory write
- I/O read and I/O write (access to PCIC local registers)
- Configuration read, configuration write
- Locked transfer is supported.
- High-speed back-to-back, is not supported.

When the PCIC is operating in non-host mode, no response is made on reception of special cycle commands.

Memory Read/Memory Write Commands: In the case of memory read and memory write commands, both single transfers and burst transfers are supported on the PCI bus. Data on the PCI bus is always longword data, but BE[3:0] can be used to control the valid byte lane. In the case of memory read, longword data is always read from the local bus and output to the PCI bus. In the

case of memory write, the internal control allows only the writing of valid byte lane data to the local bus. Only the linear mode is supported for addressing for burst transfers, and the 2 least significant bits of the PCI address are regarded as B'00.

If a memory read line command or memory read multiple command is received, they operate as memory reads. Similarly, when a memory write invalidate command is received, it functions as a memory write.

Data must be set in the following registers prior to performing target transfers using memory read or memory write commands: PCI configuration register 5 (PCICNF5), PCI configuration register 6 (PCICNF6), PCI local space register 0 (PCILSR [0]), PCI local space register 1 (PCILSR [1]), PCI local address register 0 (PCILAR [0]), and PCI local address register 1 (PCILAR [1]).

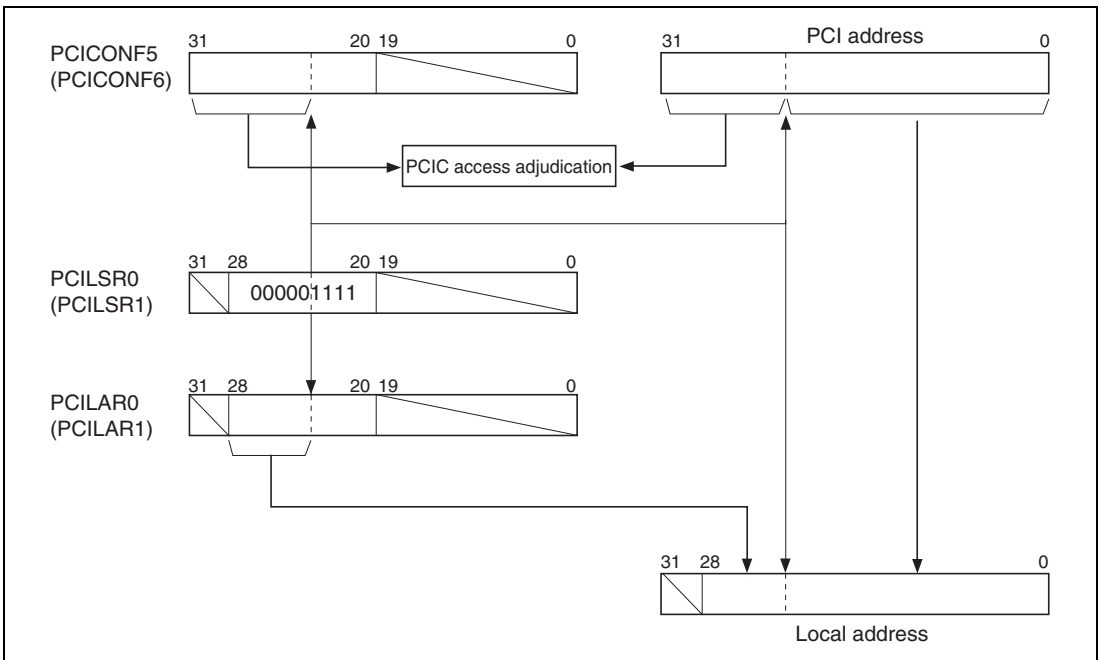


Figure 22.4 Local Address Space Accessing Method

The PCIC supports two local address spaces (address space 0 and address space 1).

A certain range of the address space on the PCI bus corresponds to the local address space.

The local address space 0 is controlled by the PCICNF5, PCILAR0 and PCISR0. Figure 22.4 shows the method of accessing the local address space.

The PCICONF5 indicates the starting address of the memory space used by the PCI device. The PCILAR0 specifies the starting address of the local address space 0. The PCILSR0 expresses the size of the memory used by the PCI device. Regarding the method of setting each register, refer to section 22.2, PCIC Register Descriptions.

For the PCICONF5 and PCILAR0, the most significant address bit that is higher than the memory size set in the PCILSR0 becomes valid. The most significant address bit of the PCICONF5 and the PCI address output from an external PCI device are compared for the purpose of determining whether the access is made to the PCIC. When the addresses correspond, the access to the PCIC is recognized, and a local address is generated from the most significant address bit of the PCILAR0 and the least significant bit of the PCI address output from the external PCI device. The PCI command is executed for this local address.

If the most significant address bit of the PCI address output from the external PCI device does not correspond with the most significant address bit of the PCICONF5, the PCIC does not respond to the PCI command.

Address space 1 is, like address space 0, controlled by the PCICONF6, PCILSR1, and PCILAR1.

In this way, it is possible to set two address spaces. In systems with two or less local bus areas that can be accessed from the PCI bus, separate address spaces can be allocated to each of them.

To make it possible to access two or more areas from the PCI bus, set the address spaces so that multiple areas are covered. In this case, we can assume that the address space includes areas for which no memory is installed. Note that, in this case, it is not possible to disable target transfers to areas for which no memory is installed.

Note: See 22.3.11 (2), Target Read/Write Cycle Timing.

I/O-Read and I/O-Write Commands: The local registers of the PCIC are accessed by means of a target transfer triggered by an I/O-read or I/O-write command. In the SH7751, accessing the local registers by means of I/O transfer is made possible by setting a base address that specifies 1 Mbyte of I/O space* in PCI configuration register 4 (PCICONF4). In the SH7751R, a base address that specifies 256 bytes of I/O space should be set.

I/O-read and I/O-write commands only supports single transfers. The values of the byte-enable signals (\overline{BE} [3:0]) are ignored, and longword accesses are carried out inside the PCIC. When executing an I/O-read and I/O-write commands transfer, specify B'0000 as the \overline{BE} [3:0] value.

Note that some of the local registers are not accessible from the PCI bus. For details, see section 22.2, PCIC Register Descriptions.

Note: * In version 2.1 of the PCI specifications the I/O space for PCI devices is defined as being no more than 256 bytes. As a result, when the SH7751 is used in a PCI non-host device, for example on an add-in card, it may be identified as an unusable device during device configuration because it requires an I/O space larger than 256 bytes.

Configuration-Read and Configuration-Write Commands: When the PCIC operates as a non-host device, the configuration registers of the PCIC are accessed by using configuration-read and configuration-write commands.

Configuration access only supports single transfers. In the SH7751, the values of the byte-enable signals (\overline{BE} [3:0]) are ignored, and longword accesses are carried out inside the PCIC*. In the SH7751R, the values of \overline{BE} [3:0] are enabled. When executing a configuration-write operation, specify B'0000 as the \overline{BE} [3:0] value.

Note: * Version 2.1 of the PCI specifications specifies that any combination of byte-enable signal (\overline{BE} [3:0]) values must be allowed when accepting a configuration access. As a result, when byte or word access is specified by the combination of \overline{BE} [3:0], the remaining portion of the data in the longword unit is also overwritten by the write operation.

Locked Transfer: Locked transfers are supported, but the locked space becomes the whole memory of the PCIC in the case of memory transfers, and becomes the whole register space in the case of I/O transfers or configuration transfers. While the memory is locked, retry is returned for all memory accesses of the PCIC from other PCI devices. Register access is, however, accepted. Similarly, while the registers are locked, retry is returned for all I/O accesses or configuration accesses of the PCIC from another PCI device, but memory access is accepted.

22.3.9 DMA Transfers

DMA transfers allow the high-speed transfer of data between devices connected to the local bus and PCI bus when the PCIC has bus privileges as master. The following commands are supported in the case of DMA transfers:

- Memory read, memory write, I/O read, and I/O write
(Locked transfers are not supported.)
(High-speed back-to-back transfers are not supported.)

There are four DMA channels. In each channel, a maximum of 64MB can be set for each transfer, the number of transfer bytes and the starting address for the transfer being set at a longword boundary.

In DMA transfers, all transferred data is handled in long word units, so the number of transfer bytes and the low 2 bits of the transfer initial address are ignored and B'0000 is always output for $\overline{BE}[3:0]$. Also, in DMA transfers, because burst transfers are effected using linear addressing, the low 2 bits of the output PCI address are always B'00.

Note that locked transfers are not supported in the case of DMA transfers.

Starting DMA Transfer: The following registers exist to control DMA transfers: PCI DMA transfer arbitration register (PCIDMABT) and, for four channels, the PCI DMA transfer PCI address register [3:0] (PCIDPA [3:0]), PCI DMA transfer local bus starting address register [3:0] (PCIDLA [3:0]), PCI DMA transfer count register [3:0] (PCIDTC [3:0]), and PCI DMA control register [3:0] (PCIDCR [3:0]).

Set the arbitration mode in PCIDMABT prior to starting the DMA transfer. Also select the DMA channel to be used, set the PCI bus starting address and local bus starting address in the appropriate PCIDPA and PCIDLA for the selected channel, respectively, set the number of bytes in the transfer in PCIDTC, set the DMA transfer mode in the PCIDCR, and specify a transfer start request.

The transfer starting address and the number of bytes in the transfer can be set on byte or word boundaries, but because the least significant two bits of these registers are ignored, the transfer is performed in longword units. Also, note that the local bus starting address set in PCIDLA is the physical address.

PCIDPA, PCIDLA, and PCIDTC are updated during data transfer. If another DMA transfer is to be performed on completion of one DMA transfer, new values must be set in these registers.

The registers controlling DMA transfers can be set from both CPU and PCI device. Note that the DMA channel allocated to the CPU and PCI device must be predetermined when configuring the system.

When performing DMA transfers, the address of the local bus and the size of data to be transferred can be set to a 32-byte boundary to ensure that data transfers on the local bus are as efficient as possible.

PCIDCR can be used to control the abortion of DMA transfers, the direction of DMA transfers, to select PCI commands (memory/I/O) whether to update the PCI address, whether to update the local address, whether to use transfer termination interrupts, and, when the local bus is big endian, the method of alignment.

Figure 22.5 shows an example of DMA transfer control register settings.

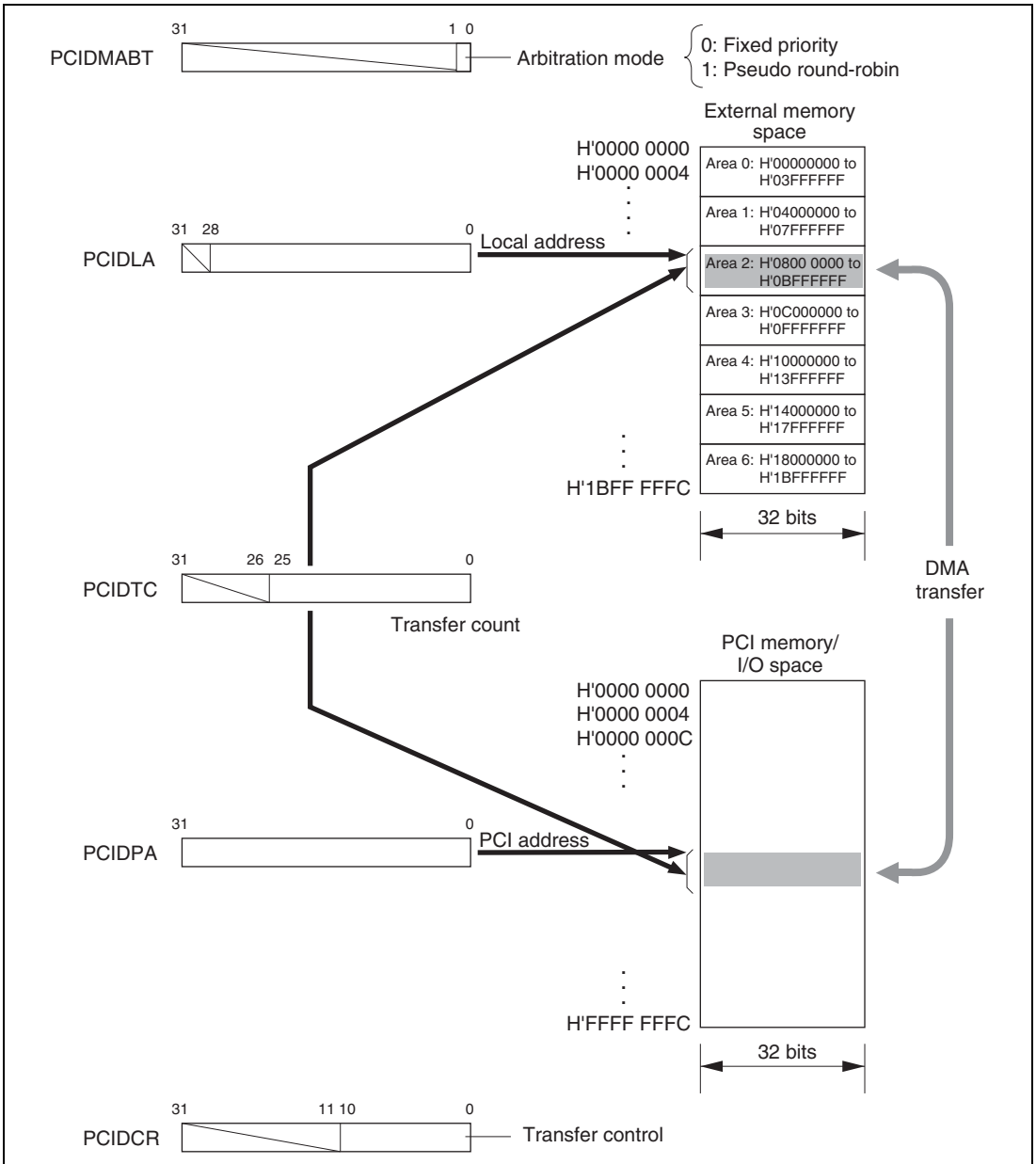


Figure 22.5 Example of DMA Transfer Control Register Settings

DMA Transfer End: The following describes the status on termination of a DMA transfer.

- Normal termination

DMA transfer ends after the set number of bytes has been transferred. In the case of normal termination, the DMA end status bit (DMAST) of the PCIDCR and the DMA transfer start control bit (DMASTART) are cleared, and the DMA transfer termination interrupt status bit (DMAIS) is set.

If the DMA transfer interrupt mask bit (DMAIM) is set to 1, the DMA transfer termination interrupt is issued.

Note that the DMAIS bit is set even if the DMAIM bit is set to 0. The DMAIS bit is maintained until it is cleared. Therefore, the DMAIS bit must be cleared before starting the next DMA transfer.

- Abnormal termination

The DMA transfer may terminate abnormally if an error on the PCI bus is detected during data transfer or the DMA transfer is forcibly terminated.

- Error in data transfer

When an error occurs during DMA transfer, the DMA transfer is forcibly terminated on the channel in which the error occurred. There is no effect on data transfers on other channels.

- Forced termination of DMA transfer

When the PCIDCR and DMASTOP bits for a channel are set, data transfer on that channel is forcibly terminated. However, when the DMASTOP bit is set, do not write 1 to the DMASTRT bit. Also, in control bits other than the DMASTOP bit, write the value at the time of transfer started.

In the case of an abnormal termination, the DMA termination status bit (DMAST) in the PCIDCR is set when the cause of that abnormal termination (error detection or forced termination of DMA transfer) occurs. After the data transfer terminates, the DMA transfer start control bit (DMASTART) is cleared and the DMA transfer termination interrupt status bit (DMAIS) is set.

If the DMA transfer interrupt mask bit (DMAIM) is set to 1, the DMA transfer termination interrupt is issued.

In the event of an abnormal termination, the transferred data is not guaranteed.

Figure 22.6 shows an example of DMA transfer flowchart.

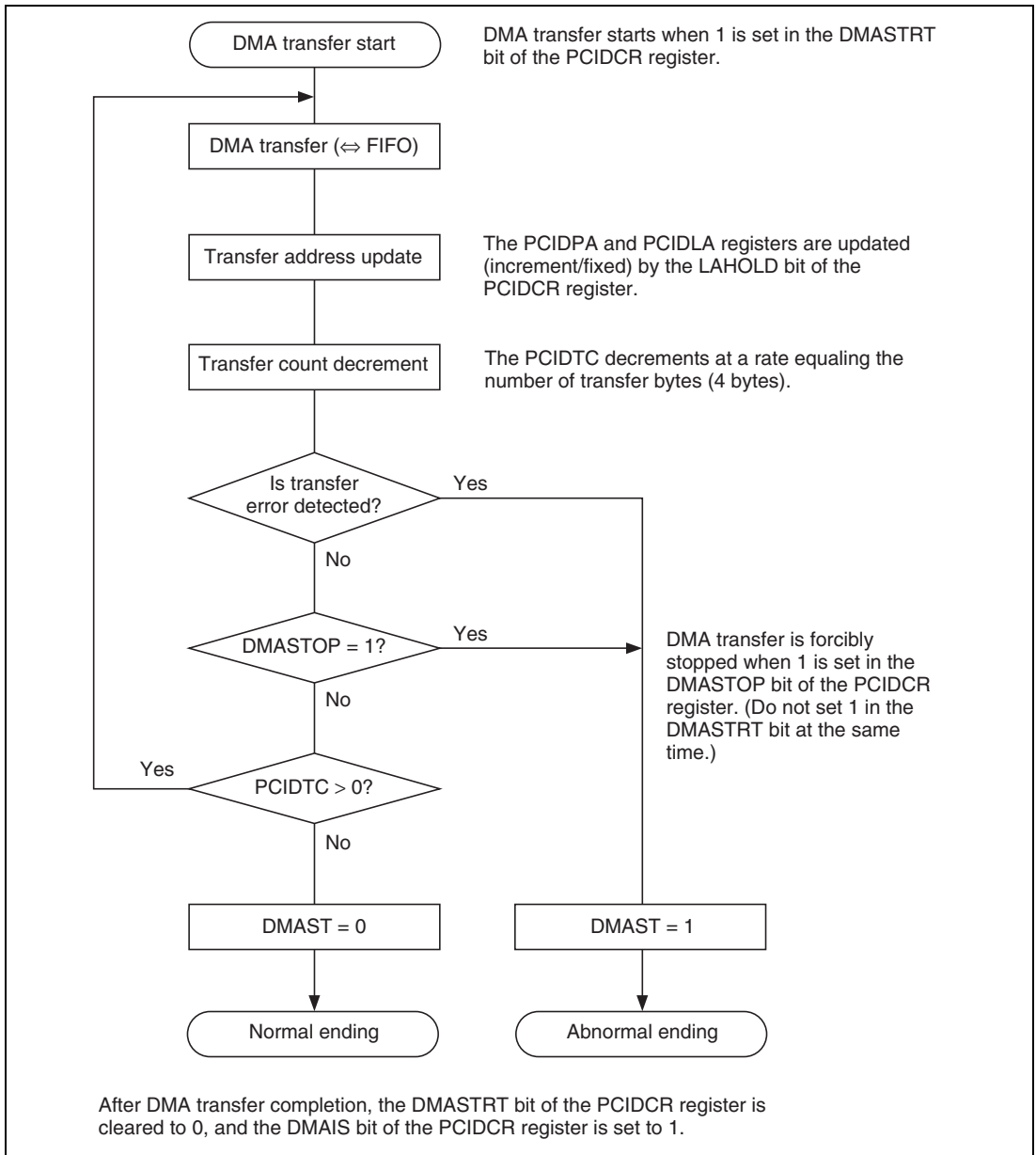


Figure 22.6 Example of DMA Transfer Flowchart

- Termination by software reset

When the RSTCTL bit of the PCICR is asserted, the PCIC is reset and DMA transfers are forcibly terminated. Note, however, that when transfers are terminated by a software reset, the PCIDCR is also reset and the DMA transfer control registers are all cleared.

DMA Arbitration: If transfer requests are made simultaneously on multiple DMA channels in the PCI, transfer arbitration is required. There are two modes that can be selected to determine order of priority of the DMA transfers on the four channels: fixed order of priority and pseudo round-robin. The mode is selected using the DMABT bit of the PCI's DMA transfer arbitration register (PCIDMABT).

For arbitration to be performed in such a way as to maintain high-speed data transfer, there are 4 FIFOs (32-byte × 2 buffer structure) for the four DMA transfer channels. The FIFOs have a 2-buffer structure, enabling one buffer to be accessed from the PCI bus while the other is being accessed from the local bus. Depending on the direction of the transfer, the input port of the FIFO for DMA transfers. Transfers are possible in both directions between the local bus and PCI bus by selecting the transfer direction.

The arbitration circuit monitors the data transfer requests (data write requests to the FIFO when the FIFO is empty and read requests from the FIFO when it is full) 4 DMA transfer channels to control the data transfers. For each transfer request, a transfer of up to 32 bytes of data is performed.

If a DMA transfer request occurs at the same time as a PIO transfer request, the PIO transfer takes precedence over transfers on the four DMA channels, regardless of the specified mode of DMA transfer priority order.

Fixed Priority Mode (DMABT = 0): In fixed priority mode, the order of priority of data transfer requests is fixed and cannot be changed. The order is as follows:

Channel 0 DMA transfer > channel 1 DMA transfer > channel 2 DMA transfer >
channel 3 DMA transfer

DMA transfer on channel 0. Take the highest priority and channel 3 DMA transfers take the lowest priority. When data transfer requests occur simultaneously, the data transfer with the highest priority takes precedence.

Let's look at data transfers from the local bus to the PCI bus in fixed priority mode. The arbitration circuit monitors the transfer requests from the respective data transfer control circuits and writes data read from the local bus to the data transfer FIFO that not only is empty but also has the highest priority.

On the other hand, it checks if transfer data exists in the respective FIFOs and reads that data from the data transfer FIFO in which there is data and which has the highest priority, and outputs that data to the PCI bus.

For example, if channel 1 FIFO is empty, the arbitration circuit writes the data from the local bus into the channel 1 FIFO. Next, if data of 32 bytes or more is in the channel 1 FIFO, it outputs that data to the PCI bus.

If data has been written to both buffers of the channel 1 FIFO, the channel 1 FIFO is busy while data is output from one of those buffers to the PCI bus. While it is busy, data is written from the local bus to the channel 2 FIFO, which has the next highest order of priority. When all data has been output from the channel 1 FIFO to the PCI bus, data is output from the channel 2 FIFO, which still contains data, to the PCI bus.

Thus, in fixed priority mode, execution alternates between the two data transfers with the highest priority.

That is, if DMA transfers are performed simultaneously on 4 channels, the data transfers start with alternation between channels 1 and 2 and then move to alternating between 2 and 3 when all the data in channel 1 has been transferred. Likewise, execution moves to alternation between channels 3 and 4 on completion of channel 2.

This pattern is the same when data is transferred from the PCI bus to the local bus.

Pseudo round-robin mode (DMABT = 1): In pseudo round-robin mode, as each time data is transferred, the order of priority is changed so that the priority level of the completed data transfer becomes the lowest.

Regarding pseudo round-robin mode operations, refer to section 22.3.5, Host Functions.

22.3.10 Transfer Contention within PCIC

No contention occurs in the PCIC in the case of PIO transfer requests from the CPU and memory reads/memory writes due to target transfers. This is because PIO transfers use an internal bus for peripheral modules, and this operates independently of the local bus that has memory accessed by external PCI devices. Contention can occur in the PCIC in the case of PIO transfer requests from the CPU and IO reads/IO writes due to target transfers (PCIC local register access). In this case, however, arbitration is performed in the PCIC such that priority is given to register access by the external PCI device that has the PCI bus rights.

22.3.11 PCI Bus Basic Interface

The PCI interface of the MCU supports a subset of version 2.1 of the PCI specifications and enables connection to a device with a PCI bus interface.

While the PCIC is set in host mode, or while set in non-host mode, operation differs according to whether or not bus parking is performed, and whether or not the PCI bus arbiter function is enabled or not.

In host mode, the AD, PAR, $\overline{C/\overline{BE}}$ signal lines are driven by the PCIC when transfers are not being performed on the PCI bus (bus parking). When the PCIC subsequently starts transfers as master, these signal lines continue to be driven until the end of the address phase. However, in non-host mode, the master performing parking is determined according to the GNT output by the external arbiter. When the master performing parking is not the same master as that starting the subsequent transfer, a high impedance state of at least one clock is generated prior to the address phase.

In host mode, the arbiters in the PCICs and the REQ and GNT between PCICs are connected internally. Here, pins $\overline{PCIREQ1/GNTIN}$, $\overline{PCIREQ2/MD9}$, $\overline{PCIREQ3/MD10}$, and $\overline{PCIREQ4}$ function as the REQ inputs from the external masters 1 to 4. Similarly, $\overline{PCIGNT1/REQOUT}$, $\overline{PCIGNT2}$, $\overline{PCIGNT3}$, and $\overline{PCIGNT4}$ function as the GNT outputs to external masters 1 to 4. Including the PCIC, arbitration of up to five masters is possible.

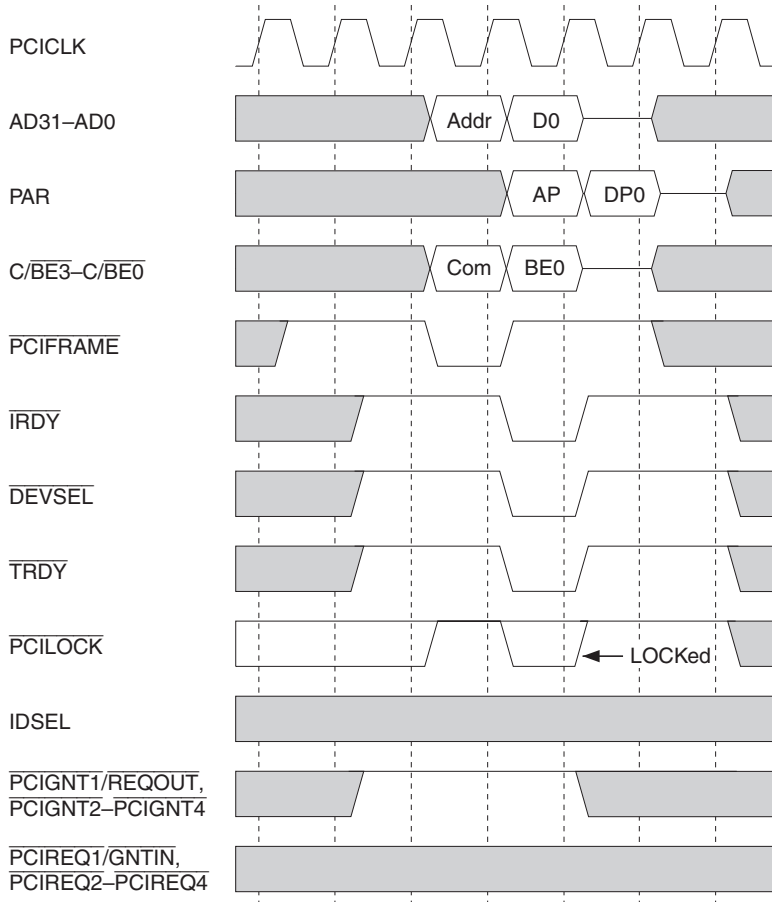
In non-host mode, pins $\overline{PCIREQ1/GNTIN}$ functions as the GNT input of the PCIC, while $\overline{PCIGNT1/REQOUT}$ functions as the REQ output of the PCIC.

Master Read/Write Cycle Timing: Figures 22.7 is an example of a single-write cycle in host mode. Figure 22.8 is an example of a single read cycle in host mode. Figure 22.9 is an example of a burst write cycle in non-host mode. And Figure 22.10 is an example of a burst read cycle in non-host mode. Note that the response speed of \overline{DEVSEL} and \overline{TRDY} differs according to the connected target device.

In PIO transfers, always use single read/write cycles.

The issuing of configuration transfers is only possible in host mode.

LOCK transfers are possible only using PIO transfers.



Legend:

Addr: PCI space address

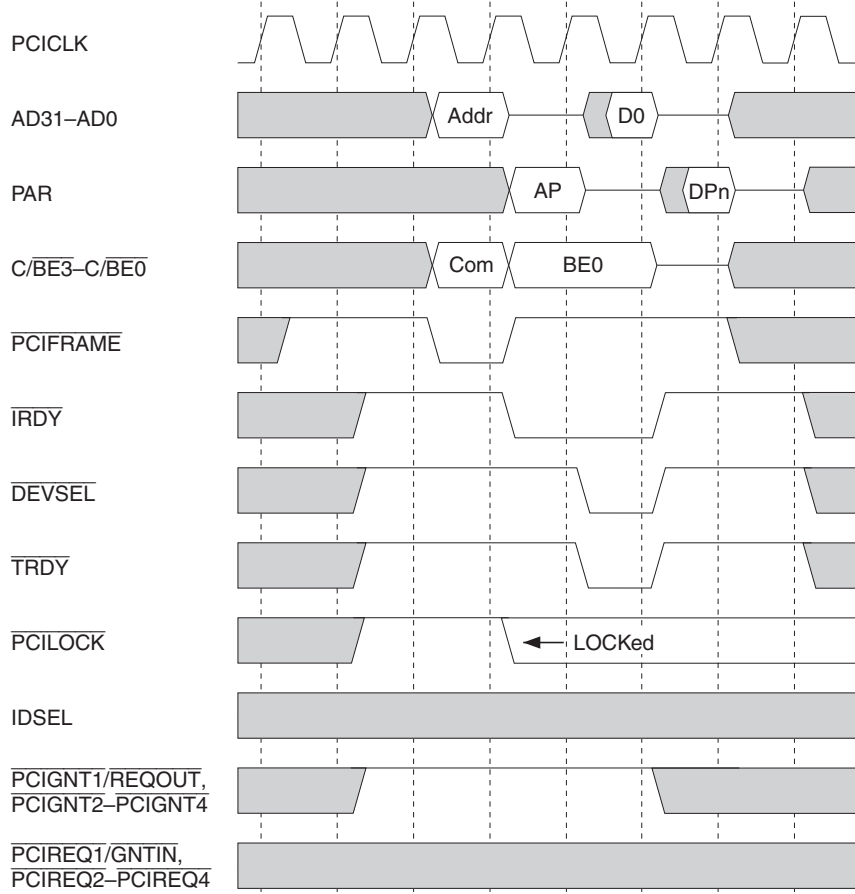
Dn: nth data

AP: Address parity

DPn: nth data parity

Com: Command

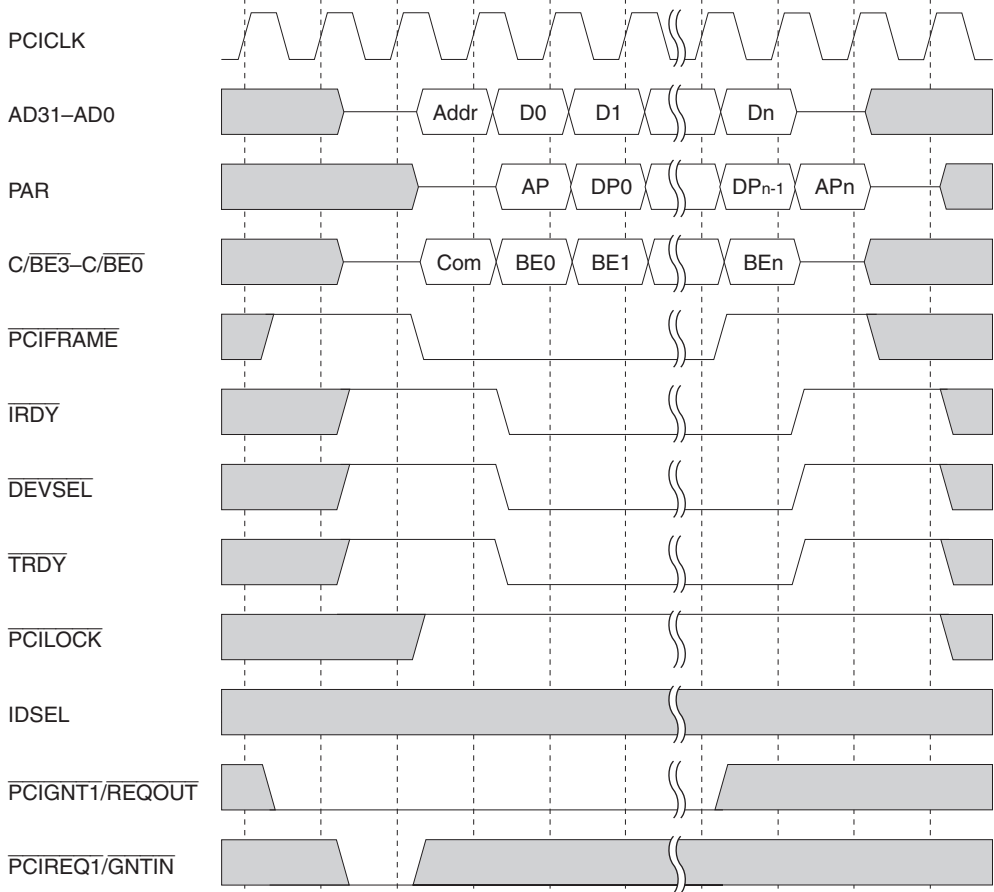
BE_n: nth data byte enable**Figure 22.7 Master Write Cycle in Host Mode (Single)**



Legend:

Addr: PCI space address
 Dn: nth data
 AP: Address parity
 DPn: nth data parity
 Com: Command
 BE_n: nth data byte enable

Figure 22.8 Master Read Cycle in Host Mode (Single)



Legend:

Addr: PCI space address

Dn: nth data

AP: Address parity

DPn: nth data parity

Com: Command

BEn: nth data byte enable

Figure 22.9 Master Memory Write Cycle in Non-Host Mode (Burst)

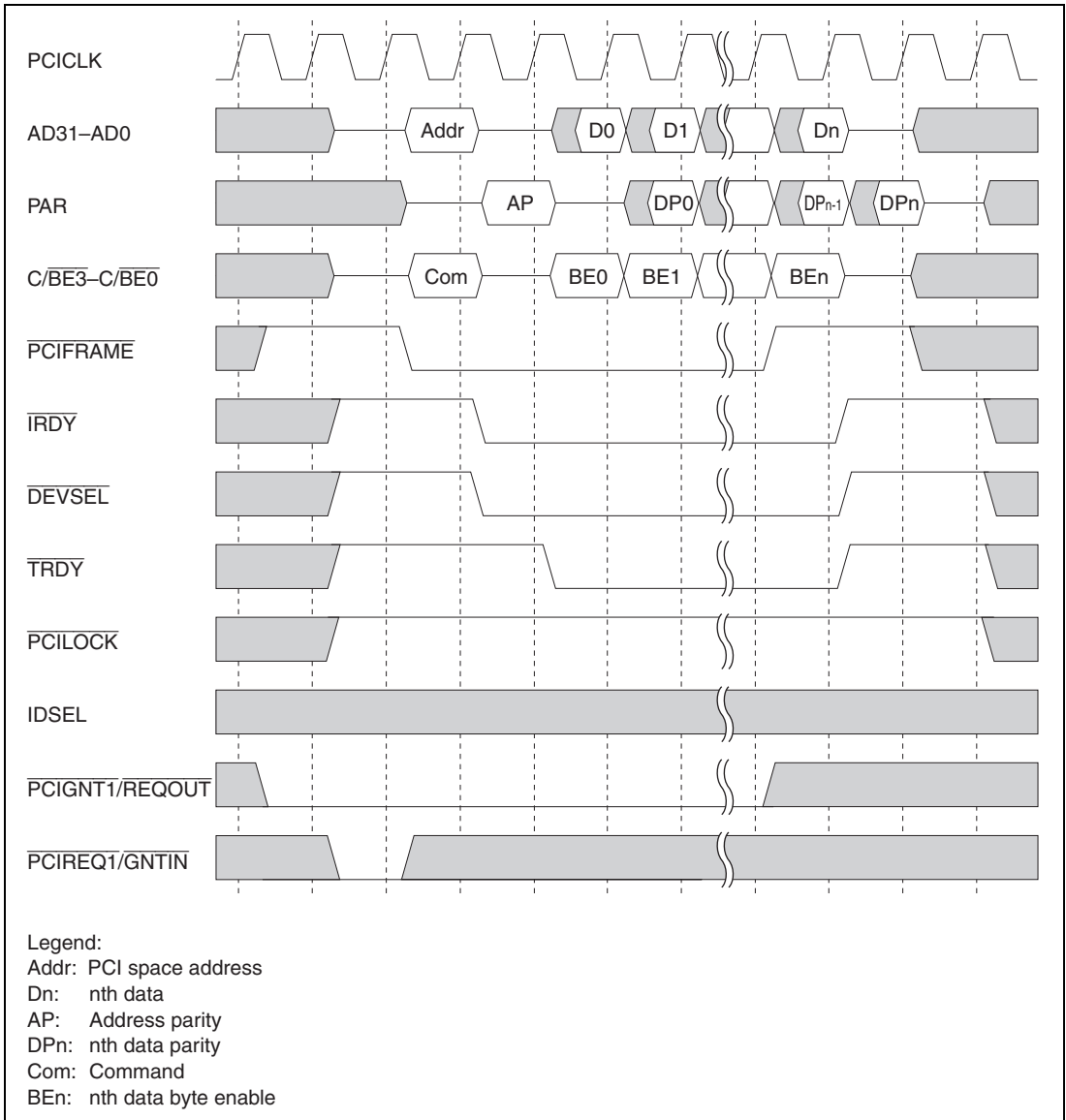


Figure 22.10 Master Memory Read Cycle in Non-Host Mode (Burst)

Target Read/Write Cycle Timing: The PCIC responds to target memory read accesses from an external master by retries until 8 longword data are prepared in the PCIC's internal FIFO. That is, it always responds to the first target read with a retry.

Also, if a target memory write access is made, the PCIC responds to all subsequent target memory accesses with a retry until the write data is completely written to local memory. Thus, the content of the data is guaranteed when data written to the target is immediately subject to a target read operation.

The following restrictions apply to the SH7751. With the SH7751R, in the following case the values of data are discarded for a target read that is executed immediately after a target write because the data read in an earlier read operation that was carried out by a different PCI device are discarded.

[Restrictions]

In a system in which access is made to the same address*¹ in local memory by two or more PCI devices, the data cannot be guaranteed when a target read is performed immediately after a target write.

The possibility of an error occurs when the target read immediately after the target write gets bus privileges at the point the data is ready for a target read by a different PCI device prior to the target write. In this case, the data prior to the target write is read. If such transfers are likely to occur, implement either (a) or (b) below.

- (a) If using the data that has been read, perform two read operations and use only the data from the second read operation.
- (b) If not using the data that has been read (if you are performing the read operation in order to determine the timing for actually writing data to the destination), be sure that the read address*² immediately after writing is different from the write address.

- Notes: 1. Address matching AD[31:2] in the address phase.
2. The address that does not correspond to the address AD[31:2] on a longword boundary.

Only single transfers are supported in the case of target accesses of the configuration space and I/O space. If there is a burst access request, the external master is disconnected on completion of the first transfer.

Note that the $\overline{\text{DEVSEL}}$ response speed is fixed at 2 clocks (Medium) in the case of target access of the PCIC.

Figure 22.11 shows an example target single read cycle in non-host mode. Figure 22.12 shows an example target single write cycle in non-host mode. Figure 22.13 is an example of a target burst read cycle in host mode. And Figure 22.14 is an example target burst write cycle in host mode.

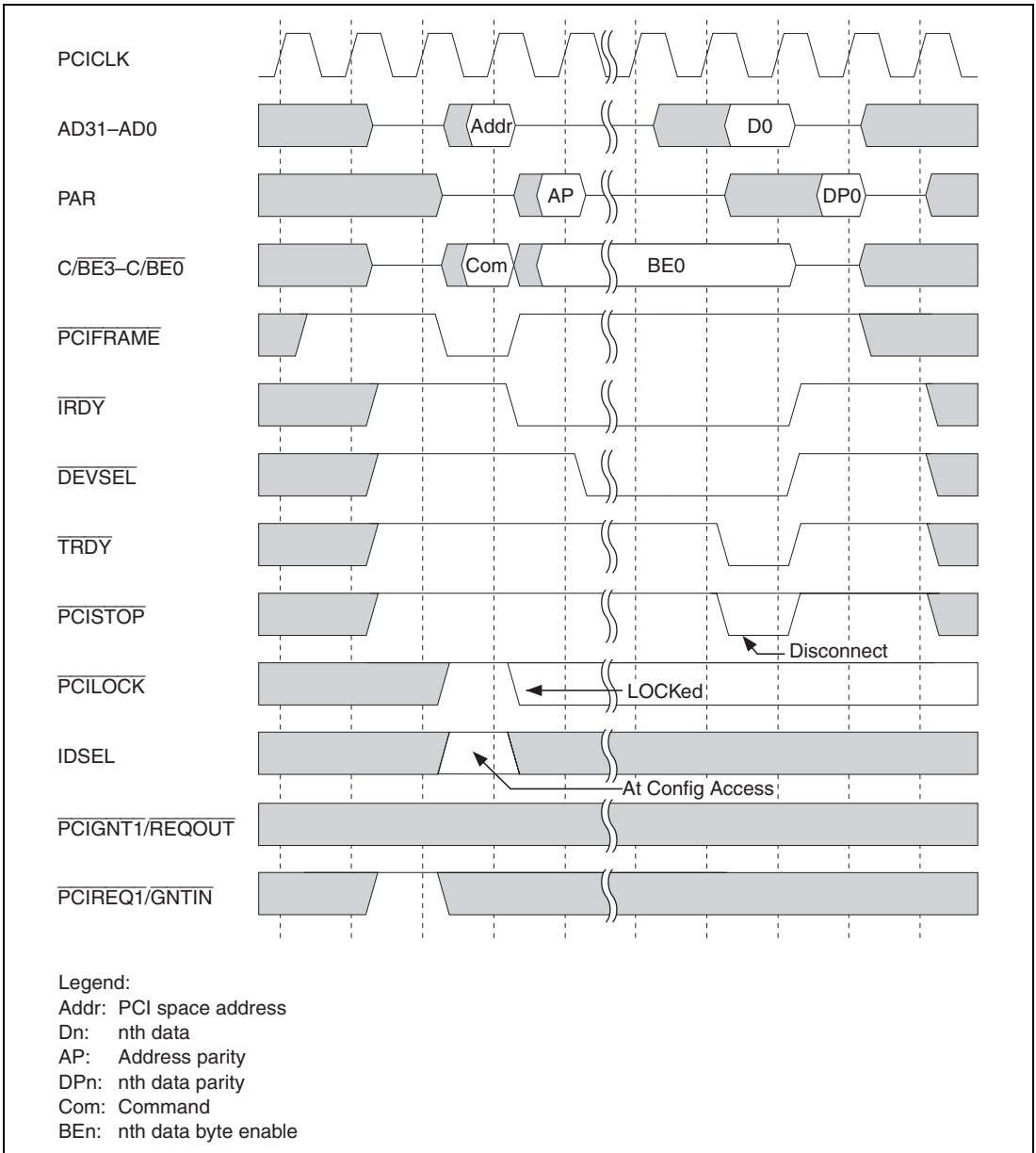


Figure 22.11 Target Read Cycle in Non-Host Mode (Single)

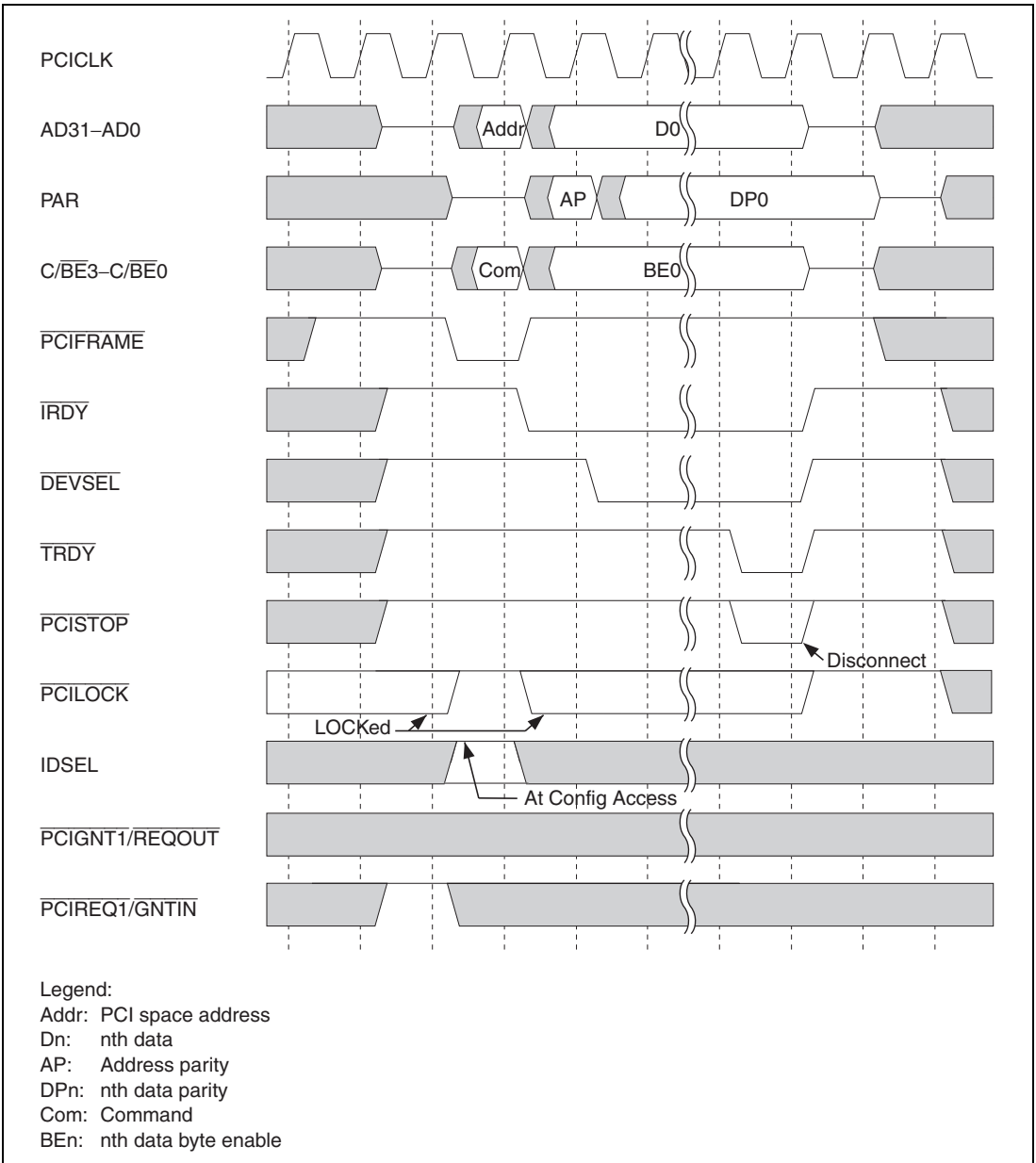


Figure 22.12 Target Write Cycle in Non-Host Mode (Single)

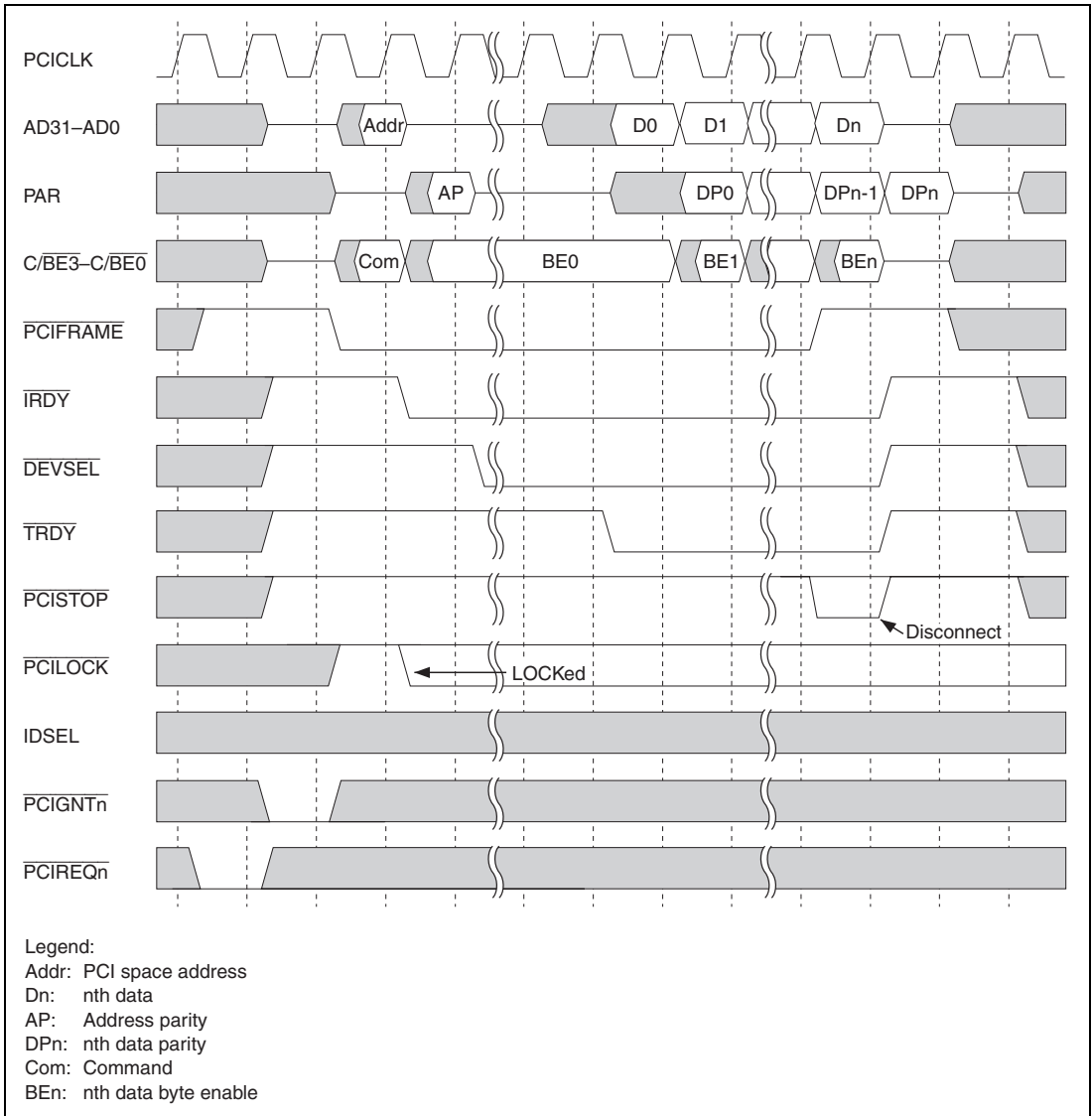


Figure 22.13 Target Memory Read Cycle in Host Mode (Burst)

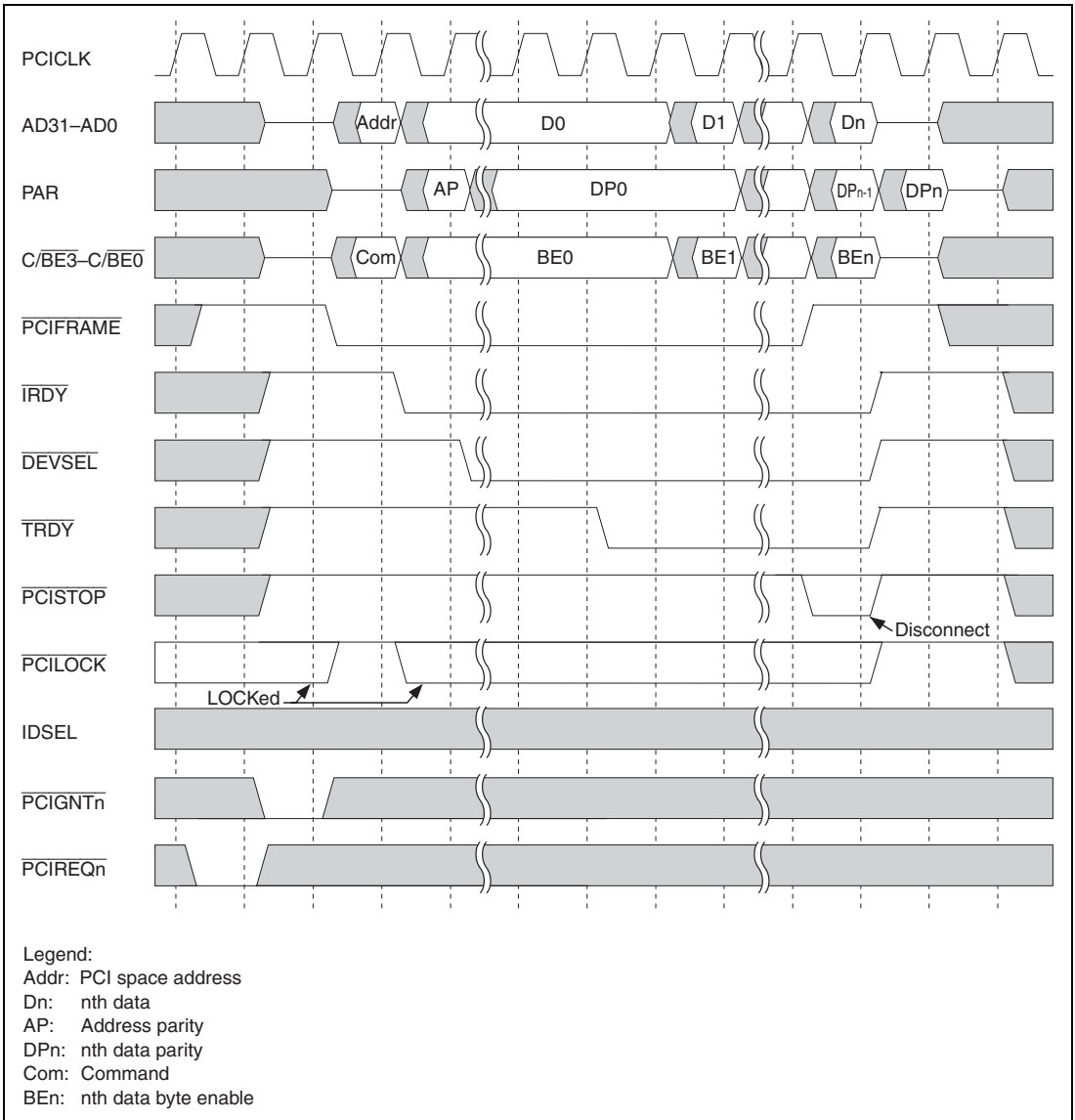


Figure 22.14 Target Memory Write Cycle in Host Mode (Burst)

Address/Data Stepping Timing: By writing 1 to the WCC bit (bit 7 of the PCICONF1), a wait (stepping) of one clock can be inserted when the PCIC is driving the AD bus. As a result, the PCIC drives the AD bus over 2 clocks. This function can be used when there is a heavy load on the PCI bus and the AD bus does not achieve the stipulated logic level in one clock.

When the PCIC operates as the host, it is recommended to use this function for the issuance of configuration transfers.

Figure 22.15 is an example of burst memory write cycle with stepping. Figure 22.16 is an example of target burst read cycle with stepping.

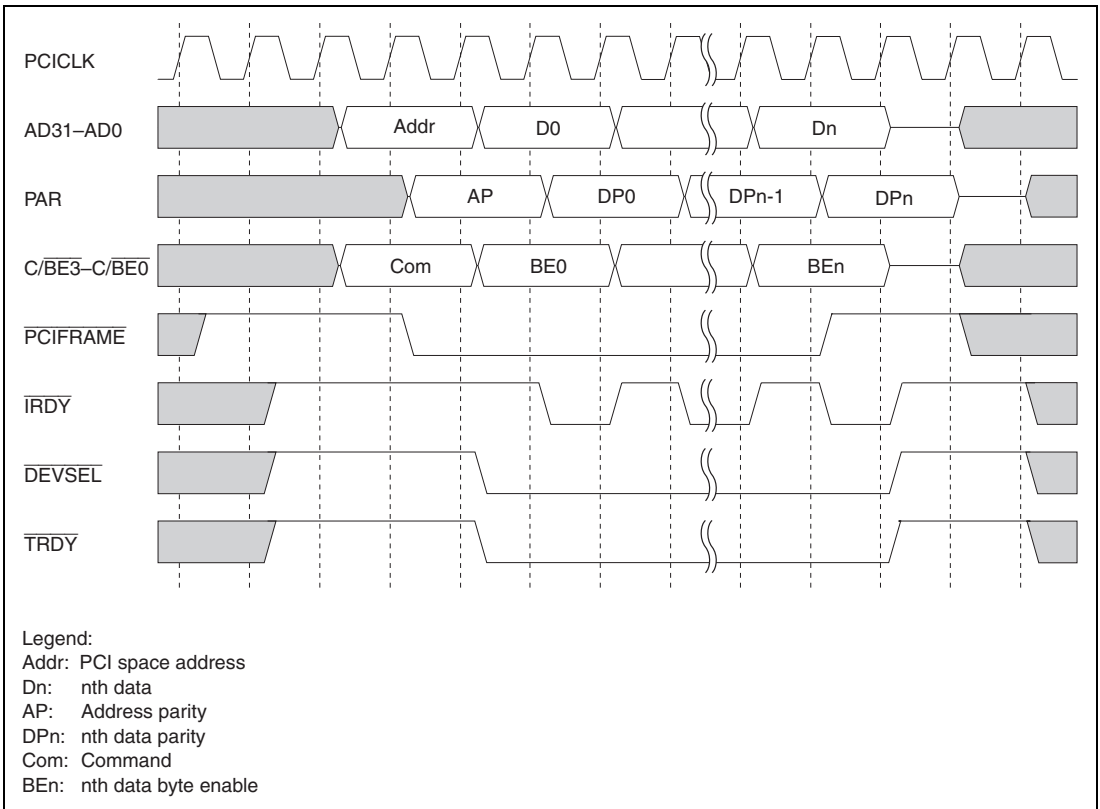


Figure 22.15 Master Memory Write Cycle in Host Mode (Burst, With Stepping)

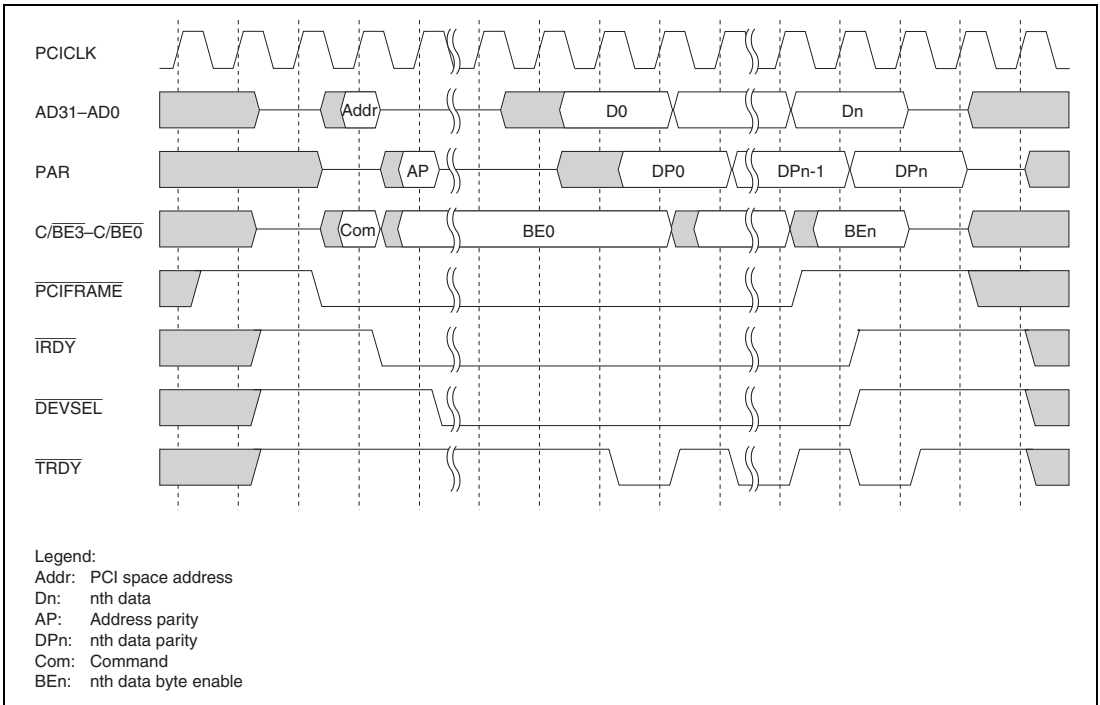


Figure 22.16 Target Memory Read Cycle in Host Mode (Burst, With Stepping)

22.4 Endians

22.4.1 Internal Bus (Peripheral Bus) Interface for Peripheral Modules

The internal bus (peripheral bus) for the peripheral modules that write data from CPU to the PCIC registers operates in big endians. On the other hand, PCI bus operates in little endian. Therefore, big/little endian conversion is required in PIO transfer, as shown in figure 22.17. The PCIC supports two endian conversion modes, the BYTESWAP bit of the PCI control register (PCICR) switching between these modes.

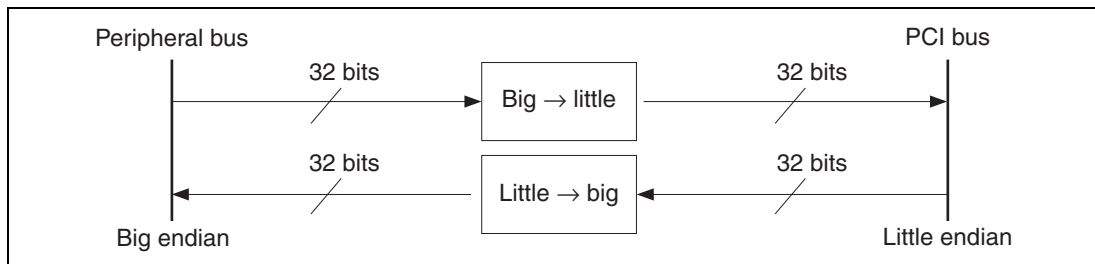


Figure 22.17 Endian Conversion Modes for Peripheral Bus

1. Byte data boundary mode: Big/little endian conversion is performed on the assumption that all data is on byte boundaries. (BYTESWAP = 1)
2. Word/longword (W/LW) boundary mode: Big/little endian conversion is performed according to the size of data accessed. (BYTESWAP = 0)

Table 22.10 shows the access sizes supported by the conversion modes at the destination of peripheral bus access. The local registers in the PCIC are always accessed in the word/longword boundary mode regardless of the transfer mode.

Figure 22.18 shows the data alignment between peripheral bus and PCI bus in each boundary mode.

Table 22.10 Access Size

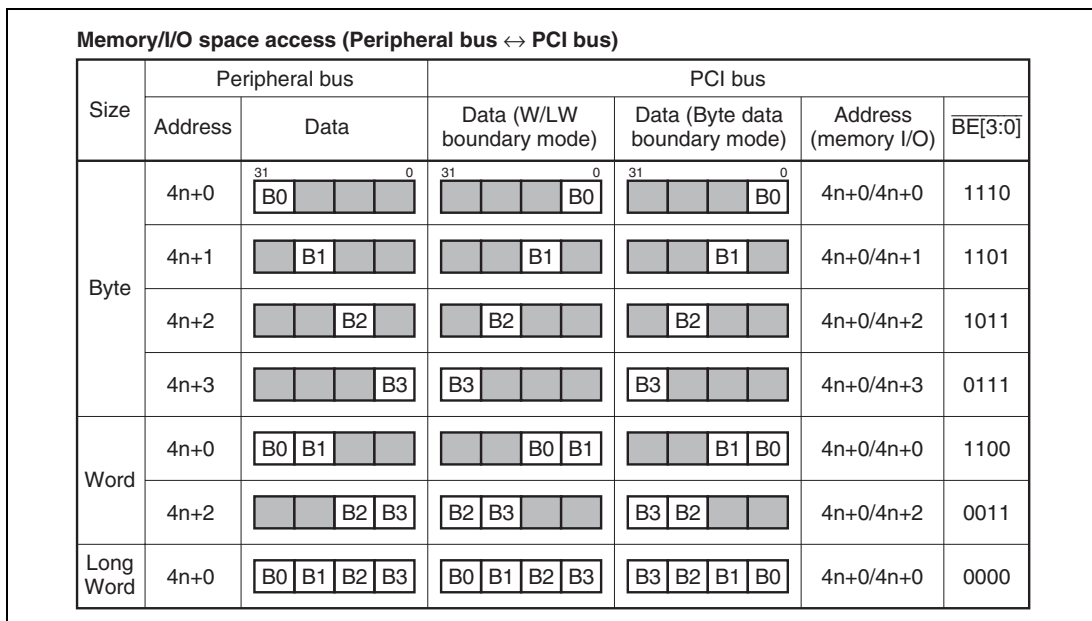
| Access Destination | Access Size | Transfer Mode | | |
|---------------------|------------------------|--------------------|-------------------------|-----|
| | | W/LW Boundary Mode | Byte Data Boundary Mode | |
| PCI external device | Memory space | B, W, LW | Yes | Yes |
| | I/O space | B, W, LW | Yes | Yes |
| | Configuration register | LW | Yes | Yes |
| PCIC register | LW | Yes | W/LW boundary mode | |

Legend:

B: Byte

W: Word

LW: Longword

**Figure 22.18 Peripheral Bus ↔ PCI Bus Data Alignment**

22.4.2 Endian Control for Local Bus

Big and little endians are supported on the local bus, determined at power-on reset by the external endian specification pin (MD5). Therefore, when transferring data between the local bus and the PCI bus, when the local bus is set for big endian, big/little endian conversion is therefore required. Figure 22.19 shows the block diagram of the local bus endian control. An endian conversion circuit is provided between the local bus and the FIFO. For details of the endian control, refer to section 22.4.3, Endian Control in DMA Transfers, and section 22.4.4, Endian Control in Target Transfers (Memory Read/Memory Write).

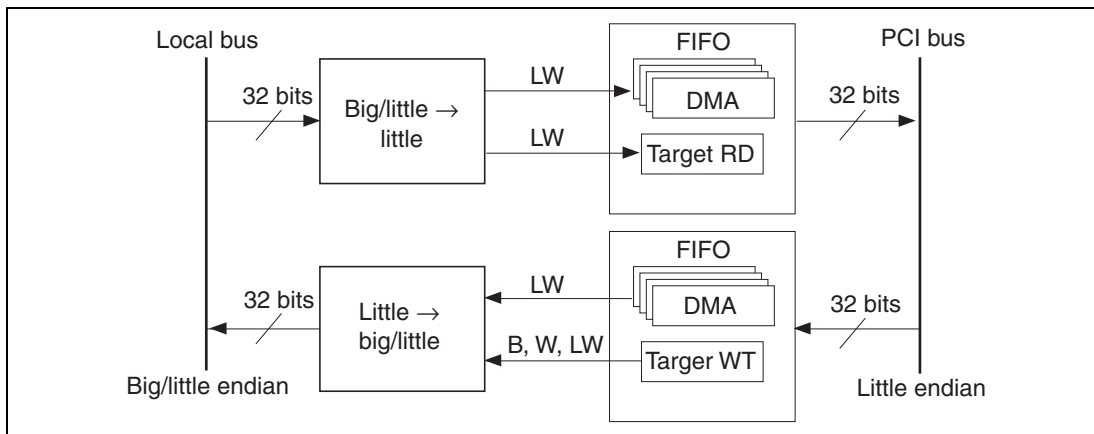


Figure 22.19 Endian Control for Local Bus

22.4.3 Endian Control in DMA Transfers

Although only the longword access size is supported in DMA transfers (see table 22.11), the endian conversion mode can be selected from the following four types depending on whether the longword data consists of four byte data units or two word data units.

The conversion mode can be switched by the setting of bits 10 and 9 (ALNMD) of the DMA control registers (PCIDCR0 to 3) for PCI.

1. Byte data boundary mode: Big/little endian conversion is performed on the assumption that all data is on a byte boundary. (ALNMD = b'00)
2. Word/longword (W/LW) boundary mode 1: Longword data is transferred as byte data x 4. (ALNMD = b'01)
3. Word/longword (W/LW) boundary mode 2: Longword data is transferred as word data x 2. (ALNMD = b'10)

4. Word/longword (W/LW) boundary mode 3: Longword data is transferred as longword data x 1. (ALNMD = b'11)

Only longword access size is supported in the case of DMA transfers.

Figure 22.20 shows the data alignment in the respective boundary modes in DMA transfers.

Table 22.11 DMA Transfer Access Size and Endian Conversion Mode

| Local Bus Endian | Data Transfer Direction | Access Size | Endian Conversion Mode | |
|------------------|-------------------------|-------------|-----------------------------|-------------------------|
| | | | W/LW Boundary Mode (1 to 3) | Byte Data Boundary Mode |
| Big endian | Local bus ↔ PCI bus | LW | Yes | Yes |
| Little endian | Local bus ↔ PCI bus | LW | Conversion not required | Conversion not required |

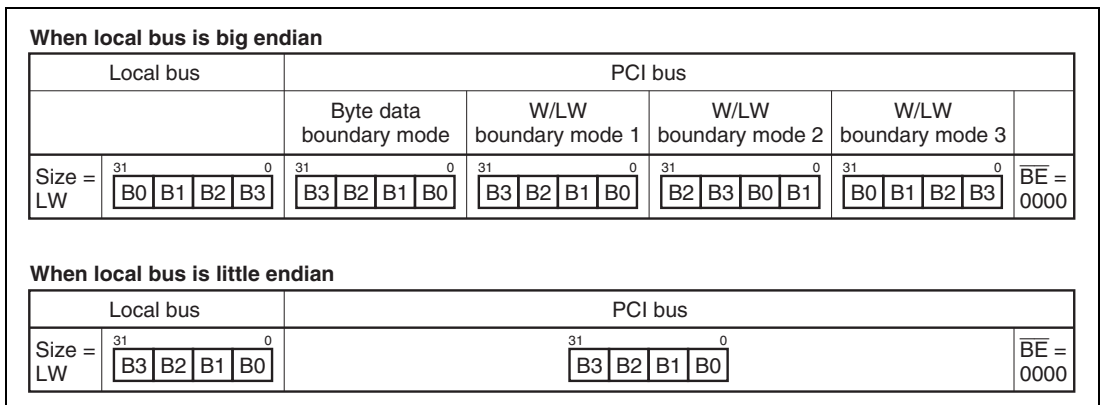


Figure 22.20 Data Alignment at DMA Transfer

22.4.4 Endian Control in Target Transfers (Memory Read/Memory Write)

In target transfers, for memory read and memory write that perform data transfer between the local bus and the PCI bus, big/little endian conversion is required in the same way as for DMA transfers when the local bus is set for big endians. Word/longword boundary modes are not supported in the case of target transfers. As shown in table 22.12, the byte data boundary mode is used, for all transfers.

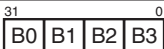
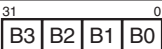
The access sizes supported in the case of target transfers are as follows: For target reads (local bus to PCI bus), longword only. For target writes (PCI bus to local bus), longword/word/byte. In target write operations, the byte, word and longword data in the PCIC are transferred to the local bus in one or two transfer operations depending on the type of the byte enable signal of the PCI bus.

For example, when $C/\overline{BE} = B'1010$, byte access to the local bus is generated twice. When $C/\overline{BE} = B'1000$, byte access and word access are each generated once.

Table 22.12 Target Transfer Access Size and Endian Conversion Mode

| Local Bus Endian | Data Transfer Direction | Access Size | Endian Conversion Mode | |
|------------------|-------------------------|-------------|-----------------------------|-------------------------|
| | | | W/LW Boundary Mode (1 to 3) | Byte Data Boundary Mode |
| Big endian | Target read | LW | No | Yes |
| | Target write | B, W, LW | No | Yes |
| Little endian | Target read | LW | Conversion not required | Conversion not required |
| | Target write | B, W, LW | Conversion not required | Conversion not required |

Target memory read transfers (local bus → PCI bus) when local bus is big endian

| Size | Local bus | PCI bus | \overline{BE} |
|------|---|---|-----------------|
| LW |  |  | H'0 to H'F |

Target memory write transfers (local bus ← PCI bus) when local bus is big endian



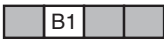

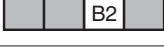
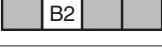


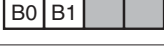
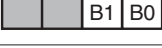
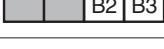
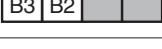
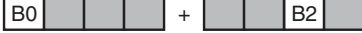






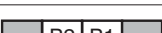

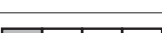

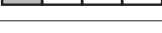

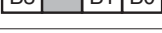
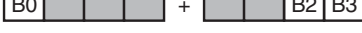

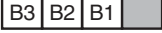


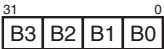
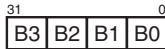
| Size | Local bus | PCI bus | \overline{BE} |
|-------|---|---|-----------------|
| B |  |  | 1110 |
| B |  |  | 1101 |
| B |  |  | 1011 |
| B |  |  | 0111 |
| W |  |  | 1100 |
| W |  |  | 0011 |
| B + B |  |  | 1010 |
| B + B |  |  | 0101 |
| B + B |  |  | 0110 |
| B + B |  |  | 1001 |
| W + B |  |  | 1000 |
| W + B |  |  | 0100 |
| B + W |  |  | 0010 |
| B + W |  |  | 0001 |
| — | — |  | 1111 |
| LW |  |  | 0000 |

Figure 22.21 (1) Data Alignment at Target Memory Transfer (Big-Endian Local Bus)

Target memory read transfers (local bus → PCI bus) when local bus is little endian

| Size | Local bus | PCI bus | \overline{BE} |
|------|---|---|-----------------|
| LW |  |  | H'0 to H'F |

Target memory write transfers (local bus ← PCI bus) when local bus is little endian





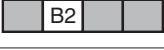
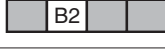


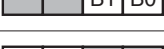
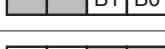

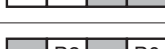


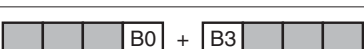

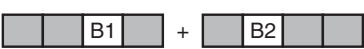
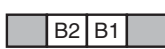
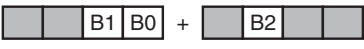
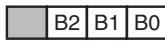



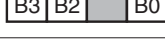

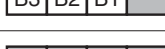
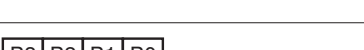
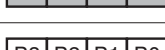
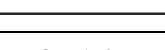


| Size | Local bus | PCI bus | \overline{BE} |
|-------|---|---|-----------------|
| B |  |  | 1110 |
| B |  |  | 1101 |
| B |  |  | 1011 |
| B |  |  | 0111 |
| W |  |  | 1100 |
| W |  |  | 0011 |
| B + B |  |  | 1010 |
| B + B |  |  | 0101 |
| B + B |  |  | 0110 |
| B + B |  |  | 1001 |
| W + B |  |  | 1000 |
| W + B |  |  | 0100 |
| B + W |  |  | 0010 |
| B + W |  |  | 0001 |
| — | — |  | 1111 |
| LW |  |  | 0000 |

Figure 22.21 (2) Data Alignment at Target Memory Transfer (Little-Endian Local Bus)

22.4.5 Endian Control in Target Transfers (I/O Read/I/O Write)

The access size is fixed at longword when accessing the PCIC local register using I/O read or I/O write commands. Addresses are specified using 4-byte boundaries, and $\overline{BE}[3:0]$ is specified as B'0000.

The data alignment in target transfers (I/O read and I/O write) is shown in figure 22.22.

| Target I/O read transfer data alignment (local register →PCI bus) | | | | |
|---|---------|---|---|-----------------|
| Size | Address | Local register | PCI bus | \overline{BE} |
| LW | 4n | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | H'0000 |

| Target I/O write transfer data alignment (PCI bus →local register) | | | | |
|--|---------|---|---|-----------------|
| Size | Address | Local register | PCI bus | \overline{BE} |
| LW | 4n | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | H'0000 |

Figure 22.22 Data Alignment at Target I/O Transfer (Both Big Endian and Little Endian)

22.4.6 Endian Control in Target Transfers (Configuration Read/Configuration Write)

The data alignment when accessing the PCIC configuration register using the target configuration read and configuration write commands is shown in figure 22.23.

In the SH7751 the access size is fixed at longword. The $\overline{BE}[3:0]$ value is ignored. In the SH7751R all \overline{BE} combinations are valid.

Target configuration read transfer data alignment (configuration register → PCI bus)

| Configuration register | PCI bus | BE |
|---|---|------------|
| $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | H'0 to H'F |

SH7751 target configuration write transfer data alignment (PCI bus → configuration register)

| Configuration register | PCI bus | BE |
|---|---|------------|
| $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | H'0 to H'F |

SH7751R target configuration transfer data alignment (PCI bus → configuration register)

| Configuration register | PCI bus | BE |
|---|---|------|
| $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | 0000 |
| $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \text{B1} & \square \\ \hline \end{array}$ | 0001 |
| $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \square & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \square & \text{B0} \\ \hline \end{array}$ | 0010 |
| $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \text{B2} & \square & \square \\ \hline \end{array}$ | 0011 |
| $\begin{array}{ c c c c } \hline \text{B3} & \square & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \square & \text{B1} & \text{B0} \\ \hline \end{array}$ | 0100 |
| $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \text{B0} \\ \hline \end{array}$ | 0100 |
| $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \square \\ \hline \end{array}$ | 0101 |
| $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \square \\ \hline \end{array}$ | 0110 |
| $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \text{B3} & \square & \square & \square \\ \hline \end{array}$ | 0111 |
| $\begin{array}{ c c c c } \hline \square & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \text{B2} & \text{B1} & \text{B0} \\ \hline \end{array}$ | 1000 |
| $\begin{array}{ c c c c } \hline \square & \text{B2} & \text{B1} & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \text{B2} & \text{B1} & \square \\ \hline \end{array}$ | 1001 |
| $\begin{array}{ c c c c } \hline \square & \text{B2} & \square & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \text{B2} & \square & \text{B0} \\ \hline \end{array}$ | 1010 |
| $\begin{array}{ c c c c } \hline \square & \text{B2} & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \text{B2} & \square & \square \\ \hline \end{array}$ | 1011 |
| $\begin{array}{ c c c c } \hline \square & \square & \text{B1} & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \square & \text{B1} & \text{B0} \\ \hline \end{array}$ | 1100 |
| $\begin{array}{ c c c c } \hline \square & \square & \square & \text{B0} \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \square & \square & \text{B0} \\ \hline \end{array}$ | 1101 |
| $\begin{array}{ c c c c } \hline \square & \square & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \square & \square & \square \\ \hline \end{array}$ | 1110 |
| $\begin{array}{ c c c c } \hline \square & \square & \square & \square \\ \hline \end{array}$ | $\begin{array}{ c c c c } \hline \square & \square & \square & \square \\ \hline \end{array}$ | 1111 |

**Figure 22.23 Data Alignment at Target Configuration Transfer
(Both Big Endian and Little Endian)**

22.5 Resetting

This section describes the resetting supported by the PCIC.

Power-On Reset when Host: A reset ($\overline{\text{PCIRST}}$) can be output to the PCI bus when the PCIC is host. The $\overline{\text{PCIRST}}$ pin is asserted when a power-on reset is generated at the $\overline{\text{RESET}}$ pin or when a software reset is generated by setting 1 in the $\overline{\text{PCIRST}}$ output control bit (RSTCTL) of the PCI control register (PCICR).

Reset Input in Non-Host Mode: The PCIC has no dedicated reset input pin. A reset signal from the PCI bus can be connected to the $\overline{\text{RESET}}$ pin and a power-on reset applied to this LSI, but the following point must be noted:

In the PCI standard, the reset ($\overline{\text{RST}}$) signal must be asserted for a minimum of 1msec, check the time required for the power-on reset of this LSI (see section 23, Electrical Characteristics), and design the timing of power-on resets so that it satisfies the conditions of the reset period for both.

Manual Reset: The PCIC does not support the input of manual reset signals via the $\overline{\text{MRESET}}$ pin. No initialization therefore occurs by manual resets.

Software Reset: Software resets are generated by setting 1 in the $\overline{\text{PCIRST}}$ output control bit (RSTCTL) of the PCI control register (PCICR). The $\overline{\text{PCIRST}}$ pin is asserted at the same time as the PCIC is reset. While a software reset is asserted, the PCIC registers cannot be accessed. Assertion requires a minimum of 1ms. Software resets are canceled by setting a 0 to the RSTCTL bit.

It is not possible to set 0 in the RSTCTL bit and set other bits of the PCICR at the same time. After setting 0 in the RSTCTL bit, set other bits of the PCICR.

Note that not all PCIC registers are reset at a software reset. See section 22.2, PCIC Register Descriptions, for details of which registers are reset. Use software clears as required for any registers that are not cleared by the software reset.

Note that, since software resets cannot be asserted while the PCI bus clock is stopped, software resets must be asserted when the PCI bus clock (PCICLK or CKIO) is being input.

Note that data cannot be guaranteed if a software reset is used while a data transfer is in progress.

22.6 Interrupts

22.6.1 Interrupts from PCIC to CPU

There are 8 interrupts, as shown in the following, that can be generated by the PCIC for the CPU.

The interrupt controller also controls the individual interrupt priority levels and interrupt masks, etc. See the section 19, Interrupt Controller (INTC), for details.

Table 22.13 Interrupts

| Interrupt Source | Function | INTPRI00 | Priority |
|------------------|------------------------------|----------|----------|
| PCISERR | SERR error interrupt | [3:0] | High |
| PCIERR | ERR error interrupt | [7:4] | High |
| PCIPWDWN | Power-down request interrupt | | |
| PCIPWON | Power-on request interrupt | | |
| PCIDMA0 | DMA0 transfer end interrupt | | |
| PCIDMA1 | DMA1 transfer end interrupt | | |
| PCIDMA2 | DMA2 transfer end interrupt | | |
| PCIDMA3 | DMA3 transfer end interrupt | | |

System Error ($\overline{\text{SERR}}$) Interrupt (PCISERR): This interrupt shows detection of the $\overline{\text{SERR}}$ pin being asserted. This interrupt is generated only when the PCIC is operating as host.

When the PCIC is operating as non-host, the SERR bit in the PCI control register (PCICR) is used to notify the host device of the system error (assertion of $\overline{\text{SERR}}$ pin).

The $\overline{\text{SERR}}$ pin can be asserted when the SERR bit is asserted and when an address parity error is detected in a target transfer.

When the SER bit of the PCI configuration register 1 (PCICONF1) is set to 0, the $\overline{\text{SERR}}$ pin is not asserted.

Error Interrupt (PCIERR): Shows error detection by the PCIC. The error interrupt is asserted when either of the following errors is detected:

- Interrupts detected by PCI interrupt register (PCIINT)
- Interrupts detected by PCI arbiter interrupt register (PCIAINT)

The interrupts that can be detected by these two registers can also be masked. The PCI interrupt mask register (PCIINTM) masks the PCIINT interrupts, and the PCI arbiter interrupt mask register (PCIAINTM) masks the PCIAINT interrupts. See section 22.2, PCIC Register Descriptions, for details.

The following are also set in relation to error interrupts: of the PCI configuration register 1 (PCICONF1), the parity error output status (DPE) the system error output status (SSE), the master abort reception status (RMA), the target abort reception status (RTA), the target abort execution status (STA) and the data parity status (DPD).

DMA Channel 0 Transfer Termination Interrupt (PCIDMA0): The DMA termination interrupt status (DMAIS) bit of the DMA control register 0 (PCIDCR0) is set. The interrupt mask is set by the DMA termination interrupt mask (DMAIM) bit of the same register.

DMA Channel 1 Transfer Termination Interrupt (PCIDMA1): The DMA termination interrupt status (DMAIS) bit of the DMA control register 1 (PCIDCR1) is set. The interrupt mask is set by the DMA termination interrupt mask (DMAIM) bit of the same register.

DMA Channel 2 Transfer Termination Interrupt (PCIDMA2): The DMA termination interrupt status (DMAIS) bit of the DMA control register 2 (PCIDCR2) is set. The interrupt mask is set by the DMA termination interrupt mask (DMAIM) bit of the same register.

DMA Channel 3 Transfer Termination Interrupt (PCIDMA3): The DMA termination interrupt status (DMAIS) bit of the DMA control register 3 (PCIDCR3) is set. The interrupt mask is set by the DMA termination interrupt mask (DMAIM) bit of the same register.

Power Management Interrupt (Transition Request to Normal Status) (PCIPWON): The power state D0 (PWRST_D0) bit of the PCI power management interrupt register (PCIPINT) is set. The power state D0 interrupt mask can be set using the power state D0 (PWRST_D0) bit of the PCI power management interrupt mask register (PCIPINTM).

Power Management Interrupt (Transition Request to Power-Down Mode) (PCIPWDWN): The power state D3 (PWRST_D3) bit of the PCI power management interrupt register (PCIPINT) is set. The power state D3 interrupt mask can be set using the power state D3 (PWRST_D3) bit of the PCI power management interrupt mask register (PCIPINTM).

22.6.2 Interrupts from External PCI Devices

To receive interrupt signals from external PCI devices, etc., while the PCIC is operating as the host device, use the $\overline{\text{IRL}}$ [3:0] pin. The PCIC has no dedicated external interrupt input pin.

22.6.3 $\overline{\text{INTA}}$

When the PCIC is operating as a non-host device, the $\overline{\text{INTA}}$ output can be used for interrupts to the host device. $\overline{\text{INTA}}$ can be asserted (Low output)/negated (High output) using the $\overline{\text{INTA}}$ output soft control bit (INTA) of the PCI control register (PCICR). $\overline{\text{INTA}}$ is open collector output.

22.7 Error Detection

The PCIC can store error information generated on the PCI bus. The address information (ALOG [31:0]) at the time of the error is stored in the PCI error address data register (PCIALR). The PCI error command information register (PCICLR) stores the type of transfer (MSTPIO, MSTDMA0, MSTDMA1, MSTDMA2, MSTDMA3, TGT) at the time of the error, and the PCI command (CMDLOG [3:0]). When the PCIC is operating as host, the PCI error bus master information register (PCIBMLR) stores the bus master information (REQ4ID, REQ3ID, REQ2ID, REQ1ID, REQ0ID) at the time of the error.

The error information storage circuit can only store information for one error. Therefore, when errors occur consecutively, no information is stored for the second or subsequent errors.

Error information is cleared by resets.

22.8 PCIC Clock

Three clocks are used with the PCIC. The peripheral module clock (Pck) is used for PCIC register access and PIO transfers. The bus clock (Bck) is used for local bus control. The PCI bus clock is used for PCI bus operation.

The peripheral module clock and PCI bus clock do not need to be in sync, and there is no particular limit on the frequency ratio. However, in PIO transfers and when registers are being accessed, etc., circuits operating with the peripheral module clock and circuits operating with the PCI bus clock and circuits that synchronize both clocks are used, so the transfer speed depends on the frequency of the peripheral module clock as well.

The bus clock (Bck) and PCI bus clock do not need to be in sync. However, the PCI bus clock should be set to the same frequency as the bus clock (Bck) or lower.

The maximum PCI bus clock is 66 MHz.

Either of the following can be selected using MD9 as the PCI bus clock: the CKIO feedback input clock and the clock input from the external input pin (PCICLK).

External Input Pin (PCICLK) Operating Mode: In this mode the PCI bus clock is input from outside. This mode requires the provision of an external oscillation module for the PCI.

CKIO Operating Mode: In this mode, the clock output from the CKIO pin is used as the PCI bus clock. The feedback input from the CKIO pin is used as the PCI bus clock.

This mode can only be used when the PCIC is operating as the host bridge. It cannot be used in non-host mode.

When using this mode, note the CKIO load capacitance and only use it within the prescribed load stated in the manual. Note, too, that the clock frequency of CKIO cannot be guaranteed until the PLL oscillation stabilizes after a power-on reset or the clock frequency is changed. Also, in standby mode, the clock stops. This mode should only be employed after checking that these points do not cause any problems from the viewpoint of the system configuration.

In CKIO operating mode, the maximum Bck frequency is 66 MHz.

When not using the PCICLK pin, fix the pin level high.

66 MHz Compatibility: The PCIC is not necessarily fully compatible with the 66 MHz bus standard of the PCI. For details, see section 23, Electrical Characteristics. In the electrical characteristics of the PCI bus-related pins, the permissible delay on the board is extremely short. For this reason, the on-board load capacitance and impedance matching should be considered before connecting to a 66MHz-compatible PCI device. Note, too, that only one PCI device can be connected.

In the PCI standard, there are two methods for checking if a PCI device can operate at 66 MHz: checking the 66 MHz operating status in the configuration register 1, and monitoring the M66ENB pin in the PCI bus standard. The PCIC supports the 66 MHz operating status (66M) bit of the configuration register 1 (PCICONF1). The PCIC does not have a special pin for directly monitoring the M66ENB pin. Also, there is no control output pin for switching between 33 MHz and 66 MHz when an external oscillator is used. A special external circuit is required to effect these controls.

22.9 Power Management

22.9.1 Power Management Overview

The PCIC supports the PCI power management (version 1.0 compatible) configuration registers. These are as follows:

- Support for the PCI power management control configuration register;
- Support for the power-down/restore request interrupts from hosts on the PCI bus.

There are three configuration registers for PCI power management control. PCI configuration register 13 shows the address offset (CAPPTR) of the configuration registers for power management. In the PCIC, this offset is fixed at CAPPTR = H'40. PCI configuration register 16 and PCI configuration register 17 are power management registers. They support two states: power state D0 (normal) and power state D3 (power down mode).

The PCIC detects when the power state (PWRST) bit of the PCI configuration register 17 changes (when it is written to from an external PCI device), and issues a power management interrupt. To control the power management interrupts, there are a PCI power management interrupt register (PCIPINT) and PCI power management interrupt mask register (PCIPINTM). Of the power management interrupts, the power state D3 (PWRST_D3) interrupt detects a transition from the power state D0 to D3, while power state D0 (PWRST_D0) interrupt detects a transition from the power state D3 to D0. Interrupt masks can be set for each interrupt.

No power state D0 interrupt is generated at a power-on reset.

The following cautions should be noted when the PCIC is operating in non-host mode and a power down interrupt is received from the host:

In PCI power management (version 1.0 compatible), the PCI bus clock stops within a minimum of 16 clocks after the host device has instructed a transition to power state D3. After detecting a power state D3 (power down) interrupt, do not, therefore, attempt to read or write to local registers that can be accessed from the CPU and PCI bus. Because these registers operate using the PCI bus clock, the read/write cycle for these registers will not be completed if the clock stops.

22.9.2 Stopping the Clock

Power savings can be achieved by stopping the bus clock used by the PCIC and the PCI bus clock. The PCI clock control register (PCICLKR) is provided for controlling the PCIC clock. Regarding the control register for stopping the peripheral module clock (Pck) in the PCIC, refer to section 9, Power-Down Modes.

The method of stopping the clock differs according to the operating mode of the PCI bus clock. See table 22.14.

Table 22.14 Method of Stopping Clock per Operating Mode

| | | PCIC | | | | | |
|-------------------------|----------------------------|-----------------------|--|---|--|--|------------------|
| | | LSI (Other than PCIC) | Master | | Slave | | |
| | | | PCICLK Operation | CKIO Operation | PCICLK Operation | | |
| Clock operating status | Normal operation/ sleep | Bck | Normal operation | Normal operation | Normal operation | Normal operation | |
| | | Pck | Normal operation | Normal operation | Normal operation | Normal operation | |
| | | PCICLK | Not used | Normal operation | Not used | Normal operation | |
| | Deep sleep | Bck | Stopped | Stopped | Stopped | Stopped | Stopped |
| | | Pck | Normal operation | Normal operation | Normal operation | Normal operation | Normal operation |
| | | PCICLK | Not used | Normal operation | Not used | Normal operation | |
| | Standby | Bck | Stopped | Stopped | Stopped | Stopped | Stopped |
| | | Pck | Stopped | Stopped | Stopped | Stopped | Stopped |
| | | PCICLK | Not used | Stopped | Not used | Stopped | |
| Transition/ Recovery | Deep sleep | Transition | Sleep command | Bck stopped from LSI | Bck and PCICLK stopped from LSI | PCI command + interrupt (PCIC → LSI) + Bck restarted from LSI | |
| | Recovery 1 | Recovery 1 | Not used | PME interrupt (connected to IRL) + Bck restarted from LSI | PME interrupt (connected to IRL) + Bck and PCICLK restarted from LSI | PCI command + interrupt (PCIC → LSI) + Bck restarted from LSI | |
| | Recovery 2 | Recovery 2 | NMI, IRL, and RESET on-chip peripheral interrupt | NMI, IRL, RESET + Bck restarted from LSI | NMI, IRL, RESET + Bck and PCICLK restarted from LSI | NMI, IRL, RESET + Bck restarted from LSI + wait for PCI command (recovery) | |

| | | PCIC | | | | |
|-------------------------|---------|-----------------------|--|----------------------------------|--|--|
| | | Master | | | Slave | |
| | | LSI (Other than PCIC) | PCICLK Operation | CKIO Operation | PCICLK Operation | |
| Transition/ Recovery | Standby | Transition | Standby command | Standby command | PCICLK stopped from LSI + standby command | PCI command + interrupt (PCIC → LSI) + standby command |
| | | Recovery 1 | Not used | PME interrupt (connected to IRL) | PME interrupt (connected to IRL) + PCICLK restarted from LSI | Power-on reset |
| | | Recovery 2 | NMI, IRL, and RESET on-chip peripheral interrupt | NMI, IRL, and RESET | NMI, IRL, RESET + PCICLK restarted from LSI | NMI, IRL, RESET + wait for PCI command (recovery) |

Notes: Recovery 1: Recovery from PCI bus

Recovery 2: Recovery from other than PCI bus

External Input Pin (PCICLK) Operating Mode: The PCI bus clock can be stopped by writing 1 to the PCICLKSTOP bit. The bus clock can be stopped by writing 1 to the BCLKSTOP bit. It requires a minimum of 2 clocks of the PCI bus clock for the clock to actually stop after writing to PCICLK (setting the PCICLKSTOP bit to 1). It takes a similar time for the clock to restart.

Bus Clock (CKIO) Operating Mode: Both the PCI bus clock and bus clock can be stopped by writing 1 to the BCLKSTOP bit. It requires a minimum of 2 clocks of the bus clock for the clock to actually stop after writing to PCICLK (setting the BCLKSTOP bit to 1). It takes a similar time for the clock to restart.

While the PCI bus clock is stopped, it is not possible to access the local registers that can be accessed both from the peripheral module internal bus and from the PCI bus. Neither writing nor reading can be performed correctly.

Also, the following cautions must be observed when stopping the bus clock and PCI bus clock while the PCI is in use:

- When operating as host device

The clock must be stopped only after stopping the operation of external PCI devices connected to the PCI bus. If you stop the clock prior to stopping the external devices, access from those external devices will cause a hang-up. Stop the clock only after checking that there is no problem in respect to the system configuration.

One method of stopping the operation of external PCI devices is to use the PCI power management, as discussed above. Stop the clock after switching the external PCI devices to power state D3 (power-down mode). In this case, all external PCI devices must support PCI power management.

- When operating in non-host mode

When operating in non-host mode, the PCI bus clock must be in external input operating mode (PCICLK). In this case, the host device is responsible for stopping and restarting the PCI bus clock, so it is not necessary to stop the clock using PCICLK of the PCIC. Make sure that the CPU receives the interrupt in accordance with the power management sequence.

22.9.3 Compatibility with Standby and Sleep

To stop all the PCIC's internal clocks, the SLEEP command must be used to transit to standby mode. When operating in external input pin (PCICLK) operating mode, set the PCICLKSTOP bit to 1 to stop the PCI bus clock, transit to standby, then, after recovering from standby, clear the PCICLKSTOP bit to 0 to prevent hazards occurring in the PCI bus clock.

When using the standby command in systems using the PCI bus, first check that the system does not hang up if the clock is stopped.

Note that the PCIC clock does not stop after transiting to sleep mode.

22.10 Port Functions

When the PCIC is operating in non-host mode, the arbitration pin of the PCI bus can be used as a port. When using the host functions (arbitration), the port functions cannot be used. The following six pins can be used: PCIREQ2, PCIREQ3, PCIREQ4, PCIGNT2, PCIGNT3, and PCIGNT4. The three pins PCIREQ2, PCIREQ3, and PCIREQ4 can be used as I/O ports. The three pins PCIGNT2, PCIGNT3, and PCIGNT4 can be used as output ports. Data is output in synchronous with the bus clock (CKIO). Input data is fetched at the rising edge of the bus clock.

Port control is performed by the port control register (PCIPCTR) and port data register (PCIPDTR). PCIPCTR controls the existence of the port function, the switching ON/OFF of the pull-up resistance, and the switching between input and output. PCIPDTR performs the input/output of port data.

22.11 Version Management

The PCIC version management is written in the revision ID (8 bits) of the PCI configuration register 2 (PCICONF2).

22.12 Usage Notes

22.12.1 Notes on Arbiter Interrupt Usage (SH7751 Only)

When the PCIC function of the SH7751 is employed as a host with an arbitration function, care must be exercised as follows with regard to the target bus timeout interrupt and master bus timeout interrupt in the PCI arbiter interrupt register (PCIAINT).

Description: On the SH7751, notification of violations of the 16-clock rule or 8-clock rule for external PCI devices (target latency and master data latency clock cycle limitations under the PCI 2.1 specification) are provided by setting bit 12 (target bus timeout interrupt) or bit 11 (master bus timeout interrupt) in the PCI arbiter interrupt register (PCIAINT) of the PCIC. However, on the SH7751 these clock cycle limitations are set to one clock cycle fewer than the values defined in the PCI 2.1 specification.

In other words, in the timings described in 1. and 2. below, even though the target latency or master data latency of the external PCI device does not violate the 16-clock rule or 8-clock rule according to the PCI 2.1 specification, the SH7751 judges that a 16-clock rule or 8-clock rule violation has occurred and sets to 1 bit 12 (target bus timeout interrupt) or bit 11 (master bus timeout interrupt) in the PCI arbiter interrupt register (PCIAINT).

1. Target latency: A target bus timeout interrupt occurs (see figures 22.24 and 22.25).
During the first data transfer, the external PCI device functioning as the target asserts $\overline{\text{TRDY}}$ or $\overline{\text{STOP}}$ at the sixteenth clock cycle after the data transfer request from the master device ($\overline{\text{FRAME}}$ asserted). Alternately, during the second or a subsequent data transfer, it asserts $\overline{\text{TRDY}}$ or $\overline{\text{STOP}}$ at the eighth clock cycle after the immediately preceding data phase.
2. Master data latency: A master bus timeout interrupt occurs (see figures 22.26 and 22.27).
The external PCI device functioning as the master acquires the bus and asserts $\overline{\text{FRAME}}$, then asserts $\overline{\text{IRDY}}$ at the eighth clock cycle during the first data transfer. Alternately, during the second or a subsequent data transfer, it asserts $\overline{\text{IRDY}}$ at the eighth clock cycle after the immediately preceding data phase.

Workarounds: When the PCIC function of the SH7751 is employed as a host with an arbitration function, and an external device is connected that employs the full number of clock cycles permitted under the 16-clock rule or 8-clock rule, use the PCI arbiter interrupt mask register (PCIAINTM) to mask the bus timeout interrupts in the PCI arbiter interrupt register (PCIAINT).

1. If the problem concerns target latency, clear to 0 bit 12 (target bus timeout interrupt mask) in the PCI arbiter interrupt mask register (PCIAINTM) to mask the target bus timeout interrupt.
2. If the problem concerns master data latency, clear to 0 bit 11 (master bus timeout interrupt mask) in the PCI arbiter interrupt mask register (PCIAINTM) to mask the master bus timeout interrupt.

Note that if the above interrupts are masked, no interrupt will occur when the 16-clock rule or 8-clock rule of PCI 2.1 specification is violated, even if the violation is detected.

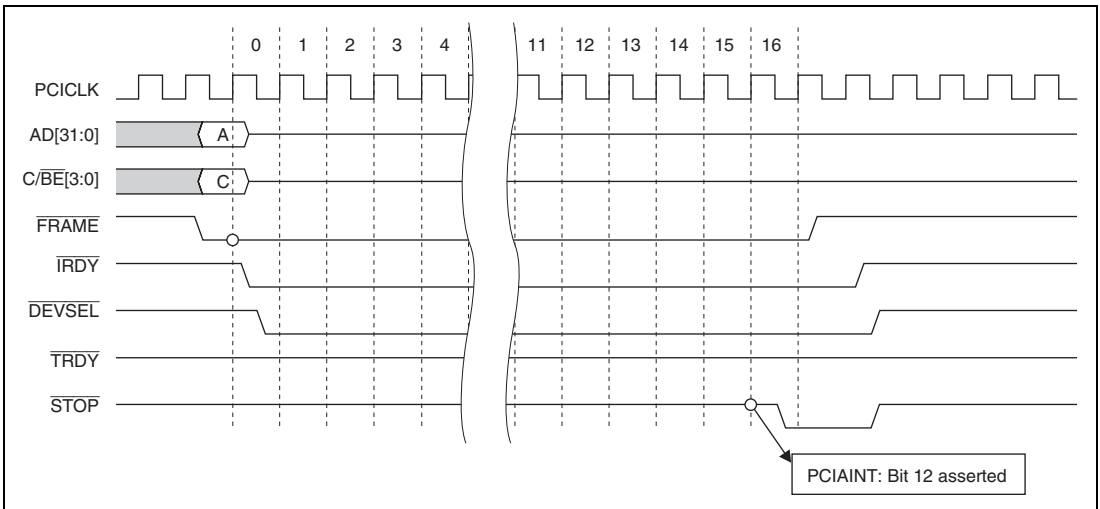


Figure 22.24 Target Bus Timeout Interrupt Generation Example 1
(Example in which the Target Device Asserts $\overline{\text{STOP}}$ at the Sixteenth Clock Cycle after $\overline{\text{FRAME}}$ Was Asserted)

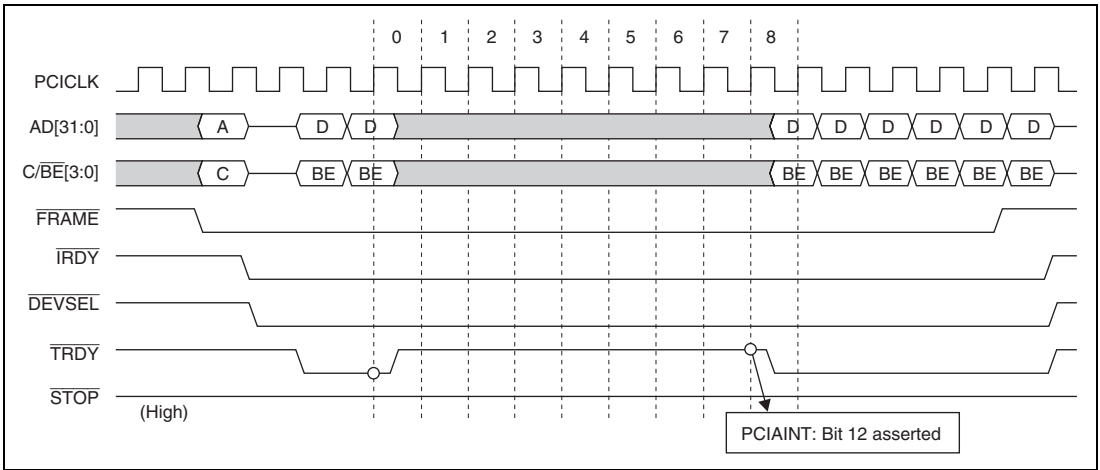


Figure 22.25 Target Bus Timeout Interrupt Generation Example 2
 (Example in which the Target Device Takes 8 Clock Cycles to Prepare for the Third Data Transfer)

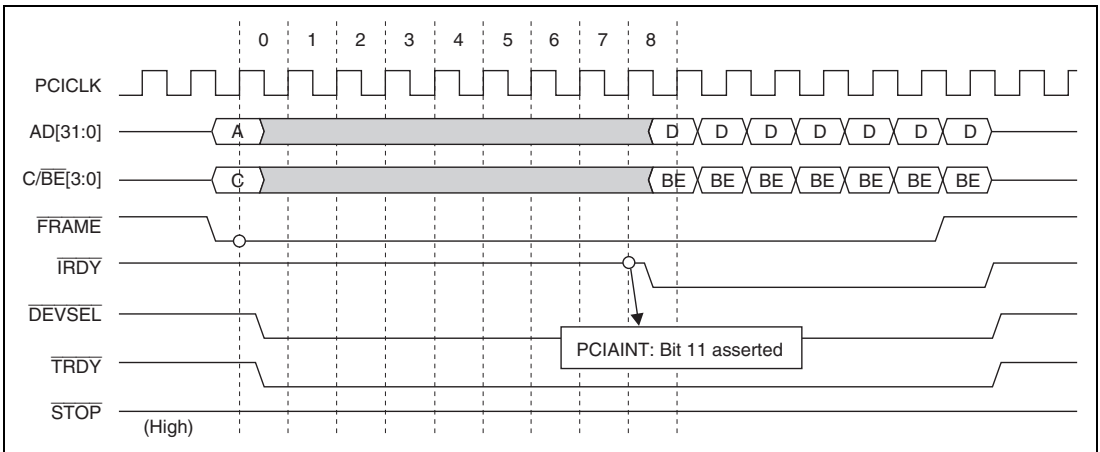


Figure 22.26 Master Bus Timeout Interrupt Generation Example 1
 (Example in which the Master Device Prepares the Data and Asserts $\overline{\text{IRDY}}$ at the Eighth Clock Cycle after $\overline{\text{FRAME}}$ Was Asserted)

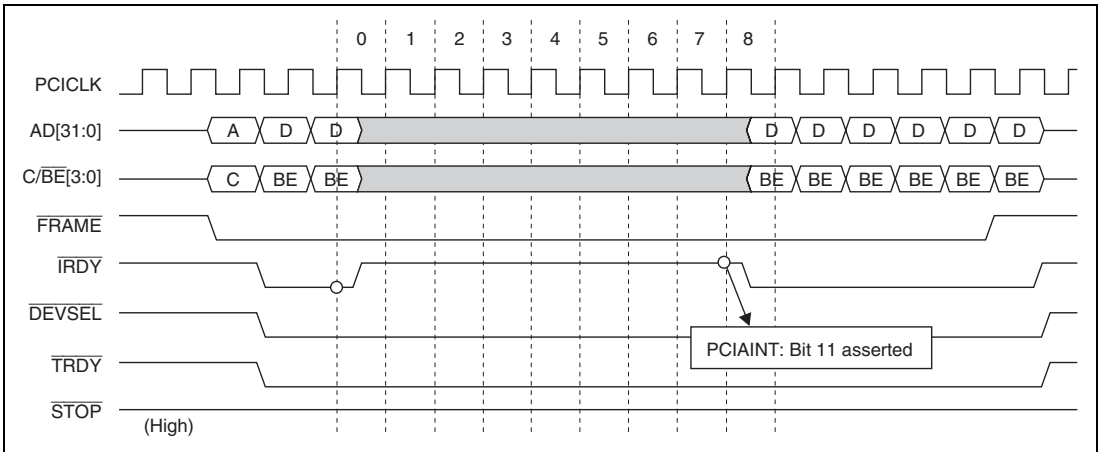


Figure 22.27 Master Bus Timeout Interrupt Generation Example 2
(Example in which the Master Device Takes 8 Clock Cycles to Prepare for the Third Data Transfer following the Second Data Phase)

22.12.2 Notes on I/O Read and I/O Write Commands (SH7751 Only)

See I/O-Read and I/O-Write Commands in 22.3.8.

22.12.3 Notes on Configuration-Read and Configuration-Write Commands (SH7751 Only)

See Configuration-Read and Configuration-Write Commands in 22.3.8.

22.12.4 Notes on Target Read/Write Cycle Timing (SH7751 Only)

See Target Read/Write Cycle Timing in 22.3.11.

22.12.5 Notes on Parity Error Detection during Master Access

Data error may not be detected while $\overline{\text{TRDY}}$ is asserted when the PCIC makes read access as a master.

Phenomenon

If all the following conditions are satisfied, data parity error cannot be detected. In such a case, the operation is the same as when no error is detected since PERR is not asserted and no detection bit is provided.

- PCIC (Master) operating conditions
 1. The PER bit in PCI configuration register 1 is set to 1 (to respond to detected parity error).
 2. Master memory read cycle
- External PCI device (target) operating conditions
 1. Target initiated disconnect (with data): \overline{STOP} asserted

Influence on System

When a target initiated disconnect (with data) occurs, a parity error is not detected in the data phase in which disconnection has occurred, and \overline{PERR} is not asserted. If not all the above conditions are satisfied, a parity error can be properly determined. This phenomenon may be inconvenient during master read access (target data parity error).

Preventive Measures

There are no preventive measures against this inconvenience that can be taken using the PCIC.

Section 23 Electrical Characteristics

23.1 Absolute Maximum Ratings

Table 23.1 Absolute Maximum Ratings

| Item | Symbol | Value | Unit |
|------------------------------------|-----------------------------|------------------------------------|------|
| I/O, RTC, CPG power supply voltage | $V_{DDQ'}$ | -0.3 to 4.2 | V |
| | $V_{DD-RTC'}$ | -0.3 to 4.6* ¹ | |
| | V_{DD-CPG} | | |
| Internal power supply voltage | $V_{DD'}$, $V_{DD-PLL1/2}$ | -0.3 to 2.5 | V |
| | | -0.3 to 2.1* ¹ | |
| Input voltage | V_{in} | -0.3 to $V_{DDQ} + 0.3$ | V |
| Operating temperature | T_{opr} | -20 to 75, -40 to 85* ² | °C |
| Storage temperature | T_{stg} | -55 to 125 | °C |

Notes: The LSI may be permanently damaged if the maximum ratings are exceeded.

The LSI may be permanently damaged if any of the VSS pins are not connected to GND.

For the powering-on and powering-off sequences, see Appendix G, Power-On and Power-Off Procedures.

1. HD6417751R only.
2. HD6417751RBA240HV only.

23.2 DC Characteristics

Table 23.2 DC Characteristics (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV)

$T_a = -20$ to $+75^\circ\text{C}^{*3}$

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions | |
|----------------------|---|--------------|----------------------|-----|-----------------|---|-------------------------------|
| Power supply voltage | V_{DDQ} | 3.0 | 3.3 | 3.6 | V | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| | V_{DD-CPG} V_{DD-RTC} | | | | | | |
| | V_{DD} | 1.4 | 1.5 | 1.6 | | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| | $V_{DD-PLL1/2}$ | | | | | | |
| Current dissipation | Normal operation | I_{DD} | — | 255 | 660 | mA | lck = 240 MHz |
| | | | — | 140 | 180 | | |
| | Sleep mode | | — | — | 400 | μA | $T_a = 25^\circ\text{C}^{*1}$ |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}^{*1}$ |
| Current dissipation | Normal operation | I_{DDQ} | — | 100 | 145 | mA | Bck = 120 MHz |
| | | | — | 60 | 115 | | |
| | Sleep mode | | — | — | 400 | μA | $T_a = 25^\circ\text{C}^{*1}$ |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}^{*1}$ |
| Current dissipation | Standby mode | I_{DD-RTC} | — | 15 | 25 | μA | RTC on *2 |
| | | | — | 3 | 5 | | RTC off |
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IH} | $V_{DDQ} \times 0.9$ | — | $V_{DDQ} + 0.3$ | V | |
| | PCICLK | | $V_{DDQ} \times 0.6$ | — | $V_{DDQ} + 0.3$ | | |
| | Other PCI input pins | | $V_{DDQ} \times 0.5$ | — | $V_{DDQ} + 0.3$ | | |
| | Other input pins | | 2.0 | — | $V_{DDQ} + 0.3$ | | |

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|-----------------------------|---|--------------------|------|-----|-----------------------------|------------------|--|
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IL} | -0.3 | — | $V_{\text{DDQ}} \times 0.1$ | V | |
| | PCICKL | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| | Other PCI input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.3$ | | |
| | Other input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| Input leak current | All input pins | $ I_{\text{in}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Three-state leak current | I/O, all output pins (off state) | $ I_{\text{sti}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Output voltage | PCI pins | V_{OH} | 2.4 | — | — | V | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -4$ mA |
| | Other output pins | | 2.4 | — | — | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -2$ mA |
| | PCI pins | V_{OL} | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 4$ mA |
| | Other output pins | | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 2$ mA |
| Pull-up resistance | All pins | R_{pull} | 20 | 60 | 180 | $\text{k}\Omega$ | |
| Pin capacitance | All pins | C_{L} | — | — | 10 | pF | |

Notes: Connect $V_{\text{DD-RTC}}$ and $V_{\text{DD-CPG}}$ to V_{DDQ} , $V_{\text{DD-PLL1/2}}$ to V_{DD} , and $V_{\text{SS-CPG}}$, $V_{\text{SS-PLL1/2}}$, and $V_{\text{SS-RTC}}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

The current dissipation values are for $V_{\text{IH min}} = V_{\text{DDQ}} - 0.5$ V and $V_{\text{IL max}} = 0.5$ V with all output pins unloaded.

I_{DD} is the sum of the V_{DD} and $V_{\text{DD-PLL1/2}}$ currents.

I_{DDQ} is the sum of the V_{DDQ} , $V_{\text{DD-RTC}}$, and $V_{\text{DD-CPG}}$ currents.

1. RCR2.RTCEN must be set to 1 to reduce the leak current in the standby mode (There is no need to input a clock from EXTAL2).
2. To reduce the leakage current in standby mode, the RTC must be turned on (RCR2.TRCEN = 1 and clock is input to EXTAL2).
3. $T_{\text{a}} = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.3 DC Characteristics (HD6417751RF240 (V)) $T_a = -20$ to $+75^\circ\text{C}$

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions | |
|----------------------|---|--------------|----------------------|-----|-----------------|---|-------------------------------|
| Power supply voltage | V_{DDQ} V_{DD-CPG} V_{DD-RTC} | 3.0 | 3.3 | 3.6 | V | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| | V_{DD} $V_{DD-PLL1/2}$ | 1.4 | 1.5 | 1.6 | | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| Current dissipation | Normal operation | I_{DD} | — | 255 | 660 | mA | Ick = 240 MHz |
| | | | — | 140 | 180 | | |
| | Sleep mode | I_{DDQ} | — | — | 400 | μA | $T_a = 25^\circ\text{C}^{*1}$ |
| | | | — | — | 800 | | |
| Current dissipation | Normal operation | I_{DDQ} | — | 70 | 100 | mA | Bck = 84 MHz |
| | | | — | 42 | 80 | | |
| | Sleep mode | I_{DDQ} | — | — | 400 | μA | $T_a = 25^\circ\text{C}^{*1}$ |
| | | | — | — | 800 | | |
| Current dissipation | Standby mode | I_{DD-RTC} | — | 15 | 25 | μA | RTC on ^{*2} |
| | | | — | 3 | 5 | | RTC off |
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IH} | $V_{DDQ} \times 0.9$ | — | $V_{DDQ} + 0.3$ | V | |
| | PCICLK | | $V_{DDQ} \times 0.6$ | — | $V_{DDQ} + 0.3$ | | |
| | Other PCI input pins | | $V_{DDQ} \times 0.5$ | — | $V_{DDQ} + 0.3$ | | |
| | Other input pins | | 2.0 | — | $V_{DDQ} + 0.3$ | | |

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|--------------------------|--|--------------------|------|-----|-----------------------------|------------------|--|
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / BRKACK , $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IL} | -0.3 | — | $V_{\text{DDQ}} \times 0.1$ | V | |
| | PCICKL | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| | Other PCI input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.3$ | | |
| | Other input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| Input leak current | All input pins | $ I_{\text{in}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Three-state leak current | I/O, all output pins (off state) | $ I_{\text{sti}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Output voltage | PCI pins | V_{OH} | 2.4 | — | — | V | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -4$ mA |
| | Other output pins | | 2.4 | — | — | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -2$ mA |
| | PCI pins | V_{OL} | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 4$ mA |
| | Other output pins | | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 2$ mA |
| Pull-up resistance | All pins | R_{pull} | 20 | 60 | 180 | $\text{k}\Omega$ | |
| Pin capacitance | All pins | C_{L} | — | — | 10 | pF | |

Notes: Connect $V_{\text{DD-RTC}}$ and $V_{\text{DD-CPG}}$ to V_{DDQ} , $V_{\text{DD-PLL1/2}}$ to V_{DD} , and $V_{\text{SS-CPG}}$, $V_{\text{SS-PLL1/2}}$, and $V_{\text{SS-RTC}}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

The current dissipation values are for $V_{\text{IH min}} = V_{\text{DDQ}} - 0.5$ V and $V_{\text{IL max}} = 0.5$ V with all output pins unloaded.

I_{DD} is the sum of the V_{DD} and $V_{\text{DD-PLL1/2}}$ currents.

I_{DDQ} is the sum of the V_{DDQ} , $V_{\text{DD-RTC}}$, and $V_{\text{DD-CPG}}$ currents.

1. RCR2.RTCEN must be set to 1 to reduce the leak current in the standby mode (There is no need to input a clock from EXTAL2).
2. To reduce the leakage current in standby mode, the RTC must be turned on (RCR2.TRCEN = 1 and clock is input to EXTAL2).

Table 23.4 DC Characteristics (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV*³) $T_a = -20$ to $+75^\circ\text{C}$ *⁴

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions | |
|----------------------|--|--------------|----------------------|-----|-----------------|---|---|
| Power supply voltage | V_{DDQ} V_{DD-CPG} V_{DD-RTC} | 3.0 | 3.3 | 3.6 | V | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| | V_{DD} $V_{DD-PLL1/2}$ | 1.35 | 1.5 | 1.6 | | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| Current dissipation | Normal operation | I_{DD} | — | 210 | 550 | mA | Ick = 200 MHz |
| | | | — | 115 | 150 | | |
| | Sleep mode | — | — | — | 400 | μA | $T_a = 25^\circ\text{C}$ * ¹ |
| | | | | | | | $T_a > 50^\circ\text{C}$ * ¹ |
| Current dissipation | Normal operation | I_{DDQ} | — | 85 | 120 | mA | Bck = 100 MHz |
| | | | — | 50 | 95 | | |
| | Sleep mode | — | — | — | 400 | μA | $T_a = 25^\circ\text{C}$ * ¹ |
| | | | | | | | $T_a > 50^\circ\text{C}$ * ¹ |
| Standby mode | — | — | — | 800 | μA | $T_a = 25^\circ\text{C}$ * ¹ | |
| | | | | | | $T_a > 50^\circ\text{C}$ * ¹ | |
| Current dissipation | Standby mode | I_{DD-RTC} | — | 15 | 25 | μA | RTC on* ² |
| | | | — | 3 | 5 | | RTC off |
| Input voltage | RESET, NMI, TRST, ASEBRK/ BRKACK, MRESET, SLEEP, CA | V_{IH} | $V_{DDQ} \times 0.9$ | — | $V_{DDQ} + 0.3$ | V | |
| | PCICLK | | $V_{DDQ} \times 0.6$ | — | $V_{DDQ} + 0.3$ | | |
| | Other PCI input pins | | $V_{DDQ} \times 0.5$ | — | $V_{DDQ} + 0.3$ | | |
| | Other input pins | | 2.0 | — | $V_{DDQ} + 0.3$ | | |

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|--------------------------|---|--------------------|------|-----|-----------------------------|---------------|--|
| Input voltage | $\overline{\text{RESET}}$, NMI , TRST , $\text{ASEBRK}/$ BRKACK , MRESET , SLEEP , CA | V_{IL} | -0.3 | — | $V_{\text{DDQ}} \times 0.1$ | V | |
| | PCICLK | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| | Other PCI input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.3$ | | |
| | Other input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| Input leak current | All input pins | $ I_{\text{in}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Three-state leak current | I/O, all output pins (off state) | $ I_{\text{sti}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Output voltage | PCI pins | V_{OH} | 2.4 | — | — | V | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -4$ mA |
| | Other output pins | | 2.4 | — | — | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -2$ mA |
| | PCI pins | V_{OL} | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 4$ mA |
| | Other output pins | | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 2$ mA |
| Pull-up resistance | All pins | R_{pull} | 20 | 60 | 180 | k Ω | |
| Pin capacitance | All pins | C_{L} | — | — | 10 | pF | |

Notes: Connect $V_{\text{DD-RTC}}$ and $V_{\text{DD-CPG}}$ to V_{DDQ} , $V_{\text{DD-PLL1/2}}$ to V_{DD1} and $V_{\text{SS-CPG}}$, $V_{\text{SS-PLL1/2}}$, and $V_{\text{SS-RTC}}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

The current dissipation values are for $V_{\text{IH min}} = V_{\text{DDQ}} - 0.5$ V and $V_{\text{IL max}} = 0.5$ V with all output pins unloaded.

I_{DD} is the sum of the V_{DD} and $V_{\text{DD-PLL1/2}}$ currents.

I_{DDQ} is the sum of the V_{DDQ} , $V_{\text{DD-RTC}}$, and $V_{\text{DD-CPG}}$ currents.

1. RCR2.RTCEN must be set to 1 to reduce the leak current in the standby mode (There is no need to input a clock from EXTAL2).
2. To reduce the leakage current in standby mode, the RTC must be turned on (RCR2.TRCEN = 1 and clock is input to EXTAL2).
3. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
4. $T_{\text{a}} = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.5 DC Characteristics (HD6417751RF200 (V)) $T_a = -20$ to $+75^\circ\text{C}$

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions | |
|----------------------|---|----------------------|----------------------|-----------------|-----------------|---|-------------------------------|
| Power supply voltage | V_{DDQ} V_{DD-CPG} V_{DD-RTC} | 3.0 | 3.3 | 3.6 | V | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| | V_{DD} $V_{DD-PLL1/2}$ | 1.35 | 1.5 | 1.6 | | Normal mode, sleep mode, deep-sleep mode, standby mode | |
| Current dissipation | Normal operation | I_{DD} | — | 210 | 550 | mA | I _{clk} = 200 MHz |
| | | | — | 115 | 150 | | |
| | Sleep mode | I_{DDQ} | — | — | 400 | μA | $T_a = 25^\circ\text{C}^{*1}$ |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}^{*1}$ |
| Current dissipation | Normal operation | I_{DDQ} | — | 70 | 100 | mA | B _{ck} = 84 MHz |
| | | | — | 42 | 80 | | |
| | Sleep mode | I_{DDQ} | — | — | 400 | μA | $T_a = 25^\circ\text{C}^{*1}$ |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}^{*1}$ |
| Current dissipation | Standby mode | I_{DD-RTC} | — | 15 | 25 | μA | RTC on ^{*2} |
| | | | — | 3 | 5 | | RTC off |
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IH} | $V_{DDQ} \times 0.9$ | — | $V_{DDQ} + 0.3$ | V | |
| | PCICLK | $V_{DDQ} \times 0.6$ | — | $V_{DDQ} + 0.3$ | | | |
| | Other PCI input pins | $V_{DDQ} \times 0.5$ | — | $V_{DDQ} + 0.3$ | | | |
| | Other input pins | 2.0 | — | $V_{DDQ} + 0.3$ | | | |

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|--------------------------|--|--------------------|------|-----|-----------------------------|------------------|--|
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / BRKACK , $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IL} | -0.3 | — | $V_{\text{DDQ}} \times 0.1$ | V | |
| | PCICKL | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| | Other PCI input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.3$ | | |
| | Other input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| Input leak current | All input pins | $ I_{\text{in}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Three-state leak current | I/O, all output pins (off state) | $ I_{\text{sti}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Output voltage | PCI pins | V_{OH} | 2.4 | — | — | V | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -4$ mA |
| | Other output pins | | 2.4 | — | — | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -2$ mA |
| | PCI pins | V_{OL} | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 4$ mA |
| | Other output pins | | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 2$ mA |
| Pull-up resistance | All pins | R_{pull} | 20 | 60 | 180 | $\text{k}\Omega$ | |
| Pin capacitance | All pins | C_{L} | — | — | 10 | pF | |

Notes: Connect $V_{\text{DD-RTC}}$ and $V_{\text{DD-CPG}}$ to V_{DDQ} , $V_{\text{DD-PLL1/2}}$ to V_{DD} , and $V_{\text{SS-CPG}}$, $V_{\text{SS-PLL1/2}}$, and $V_{\text{SS-RTC}}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

The current dissipation values are for $V_{\text{IH min}} = V_{\text{DDQ}} - 0.5$ V and $V_{\text{IL max}} = 0.5$ V with all output pins unloaded.

I_{DD} is the sum of the V_{DD} and $V_{\text{DD-PLL1/2}}$ currents.

I_{DDQ} is the sum of the V_{DDQ} , $V_{\text{DD-RTC}}$, and $V_{\text{DD-CPG}}$ currents.

1. RCR2.RTCEN must be set to 1 to reduce the leak current in the standby mode (There is no need to input a clock from EXTAL2).
2. To reduce the leakage current in standby mode, the RTC must be turned on (RCR2.TRCEN = 1 and clock is input to EXTAL2).

Table 23.6 DC Characteristics (HD6417751BP167 (V)) $T_a = -20$ to $+75^\circ\text{C}$

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions | |
|----------------------|--|--------------|----------------------|-----|-----------------|---|--|
| Power supply voltage | V_{DDQ} | 3.0 | 3.3 | 3.6 | V | Normal mode, sleep mode, standby mode | |
| | V_{DD-CPG} V_{DD-RTC} | 1.6 | 1.8 | 2.0 | | Normal mode, sleep mode, standby mode | |
| Current dissipation | Normal operation | I_{DD} | — | 420 | 750 | mA | I _{ck} = 167 MHz |
| | | | — | 100 | 130 | | |
| | Standby mode | I_{DD} | — | — | 400 | μA | $T_a = 25^\circ\text{C}$ (RTC on)* |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}$ (RTC on)* |
| Current dissipation | Normal operation | I_{DDQ} | — | 70 | 100 | mA | I _{ck} = 167 MHz, B _{ck} = 84 MHz |
| | | | — | 40 | 80 | | |
| | Standby mode | I_{DDQ} | — | — | 400 | μA | $T_a = 25^\circ\text{C}$ (RTC on)* |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}$ (RTC on)* |
| Current dissipation | Standby mode | I_{DD-RTC} | — | — | 25 | μA | RTC on |
| | | | — | — | 5 | | RTC off |
| Input voltage | RESET, NMI, TRST, ASEBRK/ BRKACK, MRESET, SLEEP, CA | V_{IH} | $V_{DDQ} \times 0.9$ | — | $V_{DDQ} + 0.3$ | V | |
| | PCICLK | | $V_{DDQ} \times 0.6$ | — | $V_{DDQ} + 0.3$ | | |
| | Other PCI input pins | | $V_{DDQ} \times 0.5$ | — | $V_{DDQ} + 0.3$ | | |
| | Other input pins | | 2.0 | — | $V_{DDQ} + 0.3$ | | |

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|--------------------------|---|--------------------|------|-----|-----------------------------|------------------|--|
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IL} | -0.3 | — | $V_{\text{DDQ}} \times 0.1$ | V | |
| | PCICKL | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| | Other PCI input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.3$ | | |
| | Other input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| Input leak current | All input pins | $ I_{\text{in}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Three-state leak current | I/O, all output pins (off state) | $ I_{\text{sti}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Output voltage | PCI pins | V_{OH} | 2.4 | — | — | V | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -4$ mA |
| | Other output pins | | 2.4 | — | — | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -2$ mA |
| | PCI pins | V_{OL} | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 4$ mA |
| | Other output pins | | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 2$ mA |
| Pull-up resistance | All pins | R_{pull} | 20 | 60 | 180 | $\text{k}\Omega$ | |
| Pin capacitance | All pins | C_{L} | — | — | 10 | pF | |

Notes: Connect $V_{\text{DD-RTC}}$ and $V_{\text{DD-CPG}}$ to V_{DDQ} , $V_{\text{DD-PLL1/2}}$ to V_{DD} , and $V_{\text{SS-CPG}}$, $V_{\text{SS-PLL1/2}}$, and $V_{\text{SS-RTC}}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

The current dissipation values are for $V_{\text{IH min}} = V_{\text{DDQ}} - 0.5$ V and $V_{\text{IL max}} = 0.5$ V with all output pins unloaded.

I_{DD} is the sum of the V_{DD} and $V_{\text{DD-PLL1/2}}$ currents.

I_{DDQ} is the sum of the V_{DDQ} , $V_{\text{DD-RTC}}$, and $V_{\text{DD-CPG}}$ currents.

* To reduce the leakage current in standby mode, the RTC must be turned on (RCR2.TRCEN = 1 and clock is input to EXTAL2).

Table 23.7 DC Characteristics (HD6417751F167 (V)) $T_a = -20$ to $+75^\circ\text{C}$

| Item | Symbol | Min | Typ | Max | Unit | Test Conditions | |
|----------------------|---|--------------|----------------------|-----|-----------------|---|--|
| Power supply voltage | V_{DDQ} | 3.0 | 3.3 | 3.6 | V | Normal mode, sleep mode, standby mode | |
| | V_{DD-CPG} V_{DD-RTC} | 1.6 | 1.8 | 2.0 | | Normal mode, sleep mode, standby mode | |
| Current dissipation | Normal operation | I_{DD} | — | 420 | 750 | mA | I _{ck} = 167 MHz |
| | | | — | 100 | 130 | | |
| | Standby mode | I_{DD} | — | — | 400 | μA | $T_a = 25^\circ\text{C}$ (RTC on)* |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}$ (RTC on)* |
| Current dissipation | Normal operation | I_{DDQ} | — | 70 | 100 | mA | I _{ck} = 167 MHz, B _{ck} = 84 MHz |
| | | | — | 40 | 80 | | |
| | Standby mode | I_{DDQ} | — | — | 400 | μA | $T_a = 25^\circ\text{C}$ (RTC on)* |
| | | | — | — | 800 | | $T_a > 50^\circ\text{C}$ (RTC on)* |
| Current dissipation | Standby mode | I_{DD-RTC} | — | — | 25 | μA | RTC on |
| | | | — | — | 5 | | RTC off |
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IH} | $V_{DDQ} \times 0.9$ | — | $V_{DDQ} + 0.3$ | V | |
| | | | $V_{DDQ} \times 0.6$ | — | $V_{DDQ} + 0.3$ | | |
| | | | $V_{DDQ} \times 0.5$ | — | $V_{DDQ} + 0.3$ | | |
| | | | 2.0 | — | $V_{DDQ} + 0.3$ | | |
| | PCICLK | | | | | | |
| | Other PCI input pins | | | | | | |
| | Other input pins | | | | | | |

| Item | | Symbol | Min | Typ | Max | Unit | Test Conditions |
|-----------------------------|---|--------------------|------|-----|-----------------------------|------------------|--|
| Input voltage | $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$ / $\overline{\text{BRKACK}}$, $\overline{\text{MRESET}}$, $\overline{\text{SLEEP}}$, CA | V_{IL} | -0.3 | — | $V_{\text{DDQ}} \times 0.1$ | V | |
| | PCICKL | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| | Other PCI input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.3$ | | |
| | Other input pins | | -0.3 | — | $V_{\text{DDQ}} \times 0.2$ | | |
| Input leak current | All input pins | $ I_{\text{in}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Three-state leak current | I/O, all output pins (off state) | $ I_{\text{sti}} $ | — | — | 1 | μA | $V_{\text{in}} = 0.5$ to V_{DDQ} -0.5 V |
| Output voltage | PCI pins | V_{OH} | 2.4 | — | — | V | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -4$ mA |
| | Other output pins | | 2.4 | — | — | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OH}} = -2$ mA |
| | PCI pins | V_{OL} | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 4$ mA |
| | Other output pins | | — | — | 0.55 | | $V_{\text{DDQ}} = 3.0$ V, $I_{\text{OL}} = 2$ mA |
| Pull-up resistance | All pins | R_{pull} | 20 | 60 | 180 | $\text{k}\Omega$ | |
| Pin capacitance | All pins | C_{L} | — | — | 10 | pF | |

Notes: Connect $V_{\text{DD-RTC}}$ and $V_{\text{DD-CPG}}$ to V_{DDQ} , $V_{\text{DD-PLL1/2}}$ to V_{DD} , and $V_{\text{SS-CPG}}$, $V_{\text{SS-PLL1/2}}$, and $V_{\text{SS-RTC}}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

The current dissipation values are for $V_{\text{IH min}} = V_{\text{DDQ}} - 0.5$ V and $V_{\text{IL max}} = 0.5$ V with all output pins unloaded.

I_{DD} is the sum of the V_{DD} and $V_{\text{DD-PLL1/2}}$ currents.

I_{DDQ} is the sum of the V_{DDQ} , $V_{\text{DD-RTC}}$, and $V_{\text{DD-CPG}}$ currents.

* To reduce the leakage current in standby mode, the RTC must be turned on (RCR2.TRCEN = 1 and clock is input to EXTAL2).

Table 23.8 Permissible Output Currents

| Item | Symbol | Min | Typ | Max | Unit |
|---|-------------------|-----|-----|-----|------|
| Permissible output low current (per pin; other than PCI pins) | I_{OL} | — | — | 2 | mA |
| Permissible output low current (per pin; PCI pins) | I_{OL} | — | — | 4 | |
| Permissible output low current (total) | ΣI_{OL} | — | — | 120 | |
| Permissible output high current (per pin; other than PCI pins) | $-I_{OH}$ | — | — | 2 | mA |
| Permissible output high current (per pin; PCI pins) | $-I_{OH}$ | — | — | 4 | |
| Permissible output high current (total) | $\Sigma(-I_{OH})$ | — | — | 40 | |

Note: To protect chip reliability, do not exceed the output current values in table 23.8.

23.3 AC Characteristics

In principle, this LSI's input should be synchronous. Unless specified otherwise, ensure that the setup time and hold times for each input signal are observed.

Table 23.9 Clock Timing (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV)

| Item | | Symbol | Min | Typ | Max | Unit | Notes |
|---------------------|----------------------|--------|-----|-----|-----|------|-------|
| Operating frequency | CPU, FPU, cache, TLB | f | 1 | — | 240 | MHz | |
| | External bus | | 1 | — | 120 | | |
| | Peripheral modules | | 1 | — | 60 | | |

Table 23.10 Clock Timing (HD6417751RF240 (V))

| Item | | Symbol | Min | Typ | Max | Unit | Notes |
|---------------------|----------------------|--------|-----|-----|-----|------|-------|
| Operating frequency | CPU, FPU, cache, TLB | f | 1 | — | 240 | MHz | |
| | External bus | | 1 | — | 84 | | |
| | Peripheral modules | | 1 | — | 60 | | |

Table 23.11 Clock Timing (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV*)

| Item | | Symbol | Min | Typ | Max | Unit | Notes |
|---------------------|----------------------|--------|-----|-----|-----|------|-------|
| Operating frequency | CPU, FPU, cache, TLB | f | 1 | — | 200 | MHz | |
| | External bus | | 1 | — | 100 | | |
| | Peripheral modules | | 1 | — | 50 | | |

Note: * This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.

Table 23.12 Clock Timing (HD6417751RF200 (V))

| Item | | Symbol | Min | Typ | Max | Unit | Notes |
|---------------------|----------------------|--------|-----|-----|-----|------|-------|
| Operating frequency | CPU, FPU, cache, TLB | f | 1 | — | 200 | MHz | |
| | External bus | | 1 | — | 84 | | |
| | Peripheral modules | | 1 | — | 50 | | |

Table 23.13 Clock Timing (HD6417751BP167 (V), HD6417751F167 (V))

| Item | | Symbol | Min | Typ | Max | Unit | Notes |
|---------------------|----------------------|--------|-----|-----|-----|------|-------|
| Operating frequency | CPU, FPU, cache, TLB | f | 1 | — | 167 | MHz | |
| | External bus | | 1 | — | 84 | | |
| | Peripheral modules | | 1 | — | 42 | | |

23.3.1 Clock and Control Signal Timing

Table 23.14 Clock and Control Signal Timing (HD6417751RBP240 (V), HD6417751RBG240 (V), HD6417751RBA240HV)

$V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C^* , $C_L = 30$ pF

| Item | Symbol | Min | Max | Unit | Figure |
|---|------------------------------|----------|------|---------------|------------------------|
| EXTAL clock input frequency | PLL1 6-times/PLL2 operation | f_{EX} | 16 | 34 | MHz |
| | PLL1 12-times/PLL2 operation | | 14 | 20 | |
| | PLL1/PLL2 not operating | | 1 | 34 | |
| EXTAL clock input cycle time | t_{EXCyc} | 30 | 1000 | ns | 23.1 |
| EXTAL clock input low-level pulse width | t_{EXL} | 3.5 | — | ns | 23.1 |
| EXTAL clock input high-level pulse width | t_{EXH} | 3.5 | — | ns | 23.1 |
| EXTAL clock input rise time | t_{EXr} | — | 4 | ns | 23.1 |
| EXTAL clock input fall time | t_{EXf} | — | 4 | ns | 23.1 |
| CKIO clock output | PLL1/PLL2 operating | f_{OP} | 25 | 120 | MHz |
| | PLL1/PLL2 not operating | | 1 | 34 | |
| CKIO clock output cycle time | t_{cyc} | 8.3 | 1000 | ns | 23.2(1) |
| CKIO clock output low-level pulse width | t_{CKOL1} | 1 | — | ns | 23.2(1) |
| CKIO clock output high-level pulse width | t_{CKOH1} | 1 | — | ns | 23.2(1) |
| CKIO clock output rise time | t_{CKOr} | — | 3 | ns | 23.2(1) |
| CKIO clock output fall time | t_{CKOf} | — | 3 | ns | 23.2(1) |
| CKIO clock output low-level pulse width | t_{CKOL2} | 3 | — | ns | 23.2(2) |
| CKIO clock output high-level pulse width | t_{CKOH2} | 3 | — | ns | 23.2(2) |
| Power-on oscillation settling time | t_{OSC1} | 10 | — | ms | 23.3, 23.5 |
| Power-on oscillation settling time/mode settling | t_{OSCMD} | 10 | — | ms | 23.3, 23.5 |
| MD reset setup time | t_{MDRS} | 3 | — | t_{cyc} | |
| MD reset hold time | t_{MDRH} | 20 | — | ns | 23.3, 23.5 |
| RESET assert time | t_{RESW} | 20 | — | t_{cyc} | 23.3, 23.4, 23.5, 23.6 |
| PLL synchronization settling time | t_{PLL} | 200 | — | μs | 23.9, 23.10 |
| Standby return oscillation settling time 1 | t_{OSC2} | 3 | — | ms | 23.4, 23.6 |
| Standby return oscillation settling time 2 | t_{OSC3} | 3 | — | ms | 23.7 |
| Standby return oscillation settling time 3 | t_{OSC4} | 3 | — | ms | 23.8 |
| Standby return oscillation settling time 1* ¹ | t_{OSC2} | 2 | — | ms | |
| Standby return oscillation settling time 2* ¹ | t_{OSC3} | 2 | — | ms | |
| Standby return oscillation settling time 3* ¹ | t_{OSC4} | 2 | — | ms | |
| IRL interrupt determination time (RTC used, standby mode) | t_{IRLSTB} | — | 200 | μs | 23.10 |
| TRST reset hold time | t_{TRSTRH} | 0 | — | ns | 23.3, 23.5 |

- Notes: When a crystal resonator is connected to EXTAL and XTAL, the maximum frequency is 34 MHz. When a 3rd overtone crystal resonator is used, an external tank circuit is necessary. As there is feedback from the CKIO pin when PLL2 is operating, the load capacitance connected to the CKIO pin should be a maximum of 50 pF.
1. When the oscillation settling time of the crystal resonator is 1 ms or less.
 2. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.15 Clock and Control Signal Timing (HD6417751RF240 (V))
 $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C , $C_L = 30$ pF

| Item | | Symbol | Min | Max | Unit | Figure |
|---|------------------------------|--------------|------|------|---------------|------------------------|
| EXTAL clock input frequency | PLL1 6-times/PLL2 operation | f_{EX} | 16 | 34 | MHz | |
| | PLL1 12-times/PLL2 operation | | 16 | 20.0 | | |
| | PLL1/PLL2 not operating | | 1 | 34 | | |
| EXTAL clock input cycle time | | t_{EXCyc} | 30 | 1000 | ns | 23.1 |
| EXTAL clock input low-level pulse width | | t_{EXL} | 3.5 | — | ns | 23.1 |
| EXTAL clock input high-level pulse width | | t_{EXH} | 3.5 | — | ns | 23.1 |
| EXTAL clock input rise time | | t_{EXr} | — | 4 | ns | 23.1 |
| EXTAL clock input fall time | | t_{EXf} | — | 4 | ns | 23.1 |
| CKIO clock output | PLL1/PLL2 operating | f_{OP} | 25 | 84 | MHz | |
| | PLL1/PLL2 not operating | | 1 | 34 | | |
| CKIO clock output cycle time | | t_{cyc} | 11.9 | 1000 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | | t_{CKOL1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output high-level pulse width | | t_{CKOH1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output rise time | | t_{CKOr} | — | 3 | ns | 23.2 (1) |
| CKIO clock output fall time | | t_{CKOf} | — | 3 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | | t_{CKOL2} | 3 | — | ns | 23.2 (2) |
| CKIO clock output high-level pulse width | | t_{CKOH2} | 3 | — | ns | 23.2 (2) |
| Power-on oscillation settling time | | t_{OSC1} | 10 | — | ms | 23.3, 23.5 |
| Power-on oscillation settling time/mode settling | | t_{OSCMD} | 10 | — | ms | 23.3, 23.5 |
| MD reset setup time | | t_{MDRS} | 3 | — | t_{cyc} | |
| MD reset hold time | | t_{MDRH} | 20 | — | ns | 23.3, 23.5 |
| $\overline{\text{RESET}}$ assert time | | t_{RESW} | 20 | — | t_{cyc} | 23.3, 23.4, 23.5, 23.6 |
| PLL synchronization settling time | | t_{PLL} | 200 | — | μs | 23.9, 23.10 |
| Standby return oscillation settling time 1 | | t_{OSC2} | 3 | — | ms | 23.4, 23.6 |
| Standby return oscillation settling time 2 | | t_{OSC3} | 3 | — | ms | 23.7 |
| Standby return oscillation settling time 3 | | t_{OSC4} | 3 | — | ms | 23.8 |
| Standby return oscillation settling time 1* | | t_{OSC2} | 2 | — | ms | |
| Standby return oscillation settling time 2* | | t_{OSC3} | 2 | — | ms | |
| Standby return oscillation settling time 3* | | t_{OSC4} | 2 | — | ms | |
| IRL interrupt determination time (RTC used, standby mode) | | t_{IRLSTB} | — | 200 | μs | 23.10 |
| $\overline{\text{TRST}}$ reset hold time | | t_{TRSTRH} | 0 | — | ns | 23.3, 23.5 |

Notes: When a crystal resonator is connected to EXTAL and XTAL, the maximum frequency is 34 MHz. When a 3rd overtone crystal resonator is used, an external tank circuit is necessary.

As there is feedback from the CKIO pin when PLL2 is operating, the load capacitance connected to the CKIO pin should be a maximum of 50 pF.

- * When the oscillation settling time of the crystal resonator is 1 ms or less.

Table 23.16 Clock and Control Signal Timing (HD6417751RBP200 (V), HD6417751RBG200 (V), HD6417751RBA240HV^{※2})
 $V_{DDQ} = 3.0 \text{ to } 3.6 \text{ V}$, $V_{DD} = 1.5 \text{ V}$, $T_a = -20 \text{ to } 75^\circ\text{C}^{\ast 3}$, $C_L = 30 \text{ pF}$

| Item | | Symbol | Min | Max | Unit | Figure |
|---|------------------------------|--------------|-----|------|---------------|------------------------|
| EXTAL clock input frequency | PLL1 6-times/PLL2 operation | f_{EX} | 16 | 34 | MHz | |
| | PLL1 12-times/PLL2 operation | | 14 | 17 | | |
| | PLL1/PLL2 not operating | | 1 | 34 | | |
| EXTAL clock input cycle time | | t_{EXcyc} | 30 | 1000 | ns | 23.1 |
| EXTAL clock input low-level pulse width | | t_{EXL} | 3.5 | — | ns | 23.1 |
| EXTAL clock input high-level pulse width | | t_{EXH} | 3.5 | — | ns | 23.1 |
| EXTAL clock input rise time | | t_{EXr} | — | 4 | ns | 23.1 |
| EXTAL clock input fall time | | t_{EXf} | — | 4 | ns | 23.1 |
| CKIO clock output | PLL1/PLL2 operating | f_{OP} | 25 | 100 | MHz | |
| | PLL1/PLL2 not operating | | 1 | 34 | | |
| CKIO clock output cycle time | | t_{cyc} | 10 | 1000 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | | t_{CKOL1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output high-level pulse width | | t_{CKOH1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output rise time | | t_{CKOr} | — | 3 | ns | 23.2 (1) |
| CKIO clock output fall time | | t_{CKOf} | — | 3 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | | t_{CKOL2} | 3 | — | ns | 23.2 (2) |
| CKIO clock output high-level pulse width | | t_{CKOH2} | 3 | — | ns | 23.2 (2) |
| Power-on oscillation settling time | | t_{OSC1} | 10 | — | ms | 23.3, 23.5 |
| Power-on oscillation settling time/mode settling | | t_{OSCMD} | 10 | — | ms | 23.3, 23.5 |
| MD reset setup time | | t_{MDRS} | 3 | — | t_{cyc} | |
| MD reset hold time | | t_{MDRH} | 20 | — | ns | 23.3, 23.5 |
| RESET assert time | | t_{RESW} | 20 | — | t_{cyc} | 23.3, 23.4, 23.5, 23.6 |
| PLL synchronization settling time | | t_{PLL} | 200 | — | μs | 23.9, 23.10 |
| Standby return oscillation settling time 1 | | t_{OSC2} | 5 | — | ms | 23.4, 23.6 |
| Standby return oscillation settling time 2 | | t_{OSC3} | 5 | — | ms | 23.7 |
| Standby return oscillation settling time 3 | | t_{OSC4} | 5 | — | ms | 23.8 |
| Standby return oscillation settling time 1 ^{※1} | | t_{OSC2} | 2 | — | ms | |
| Standby return oscillation settling time 2 ^{※1} | | t_{OSC3} | 2 | — | ms | |
| Standby return oscillation settling time 3 ^{※1} | | t_{OSC4} | 2 | — | ms | |
| IRL interrupt determination time (RTC used, standby mode) | | t_{IRLSTB} | — | 200 | μs | 23.10 |
| TRST reset hold time | | t_{TRSTRH} | 0 | — | ns | 23.3, 23.5 |

- Notes: When a crystal resonator is connected to EXTAL and XTAL, the maximum frequency is 34 MHz. When a 3rd overtone crystal resonator is used, an external tank circuit is necessary. As there is feedback from the CKIO pin when PLL2 is operating, the load capacitance connected to the CKIO pin should be a maximum of 50 pF.
1. When the oscillation settling time of the crystal resonator is 1 ms or less.
 2. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
 3. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.17 Clock and Control Signal Timing (HD6417751RF200 (V))
 $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C , $C_L = 30$ pF

| Item | Symbol | Min | Max | Unit | Figure |
|---|------------------------------|----------|------|---------------|------------------------|
| EXTAL clock input frequency | PLL1 6-times/PLL2 operation | f_{EX} | 16 | 34 | MHz |
| | PLL1 12-times/PLL2 operation | | 14 | 17 | |
| | PLL1/PLL2 not operating | | 1 | 34 | |
| EXTAL clock input cycle time | t_{EXCyc} | 30 | 1000 | ns | 23.1 |
| EXTAL clock input low-level pulse width | t_{EXL} | 3.5 | — | ns | 23.1 |
| EXTAL clock input high-level pulse width | t_{EXH} | 3.5 | — | ns | 23.1 |
| EXTAL clock input rise time | t_{EXr} | — | 4 | ns | 23.1 |
| EXTAL clock input fall time | t_{EXf} | — | 4 | ns | 23.1 |
| CKIO clock output | PLL1/PLL2 operating | f_{OP} | 25 | 84 | MHz |
| | PLL1/PLL2 not operating | | 1 | 34 | MHz |
| CKIO clock output cycle time | t_{cyc} | 11.9 | 1000 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | t_{CKOL1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output high-level pulse width | t_{CKOH1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output rise time | t_{CKOr} | — | 3 | ns | 23.2 (1) |
| CKIO clock output fall time | t_{CKOf} | — | 3 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | t_{CKOL2} | 3 | — | ns | 23.2 (2) |
| CKIO clock output high-level pulse width | t_{CKOH2} | 3 | — | ns | 23.2 (2) |
| Power-on oscillation settling time | t_{OSC1} | 10 | — | ms | 23.3, 23.5 |
| Power-on oscillation settling time/mode settling | t_{OSCMD} | 10 | — | ms | 23.3, 23.5 |
| MD reset setup time | t_{MDRS} | 3 | — | t_{cyc} | |
| MD reset hold time | t_{MDRH} | 20 | — | ns | 23.3, 23.5 |
| $\overline{\text{RESET}}$ assert time | t_{RESW} | 20 | — | t_{cyc} | 23.3, 23.4, 23.5, 23.6 |
| PLL synchronization settling time | t_{PLL} | 200 | — | μs | 23.9, 23.10 |
| Standby return oscillation settling time 1 | t_{OSC2} | 5 | — | ms | 23.4, 23.6 |
| Standby return oscillation settling time 2 | t_{OSC3} | 5 | — | ms | 23.7 |
| Standby return oscillation settling time 3 | t_{OSC4} | 5 | — | ms | 23.8 |
| Standby return oscillation settling time 1* | t_{OSC2} | 2 | — | ms | |
| Standby return oscillation settling time 2* | t_{OSC3} | 2 | — | ms | |
| Standby return oscillation settling time 3* | t_{OSC4} | 2 | — | ms | |
| IRL interrupt determination time (RTC used, standby mode) | t_{IRLSTB} | — | 200 | μs | 23.10 |
| $\overline{\text{TRST}}$ reset hold time | t_{TRSTRH} | 0 | — | ns | 23.3, 23.5 |

Notes: When a crystal resonator is connected to EXTAL and XTAL, the maximum frequency is 34 MHz. When a 3rd overtone crystal resonator is used, an external tank circuit is necessary.

As there is feedback from the CKIO pin when PLL2 is operating, the load capacitance connected to the CKIO pin should be a maximum of 50 pF.

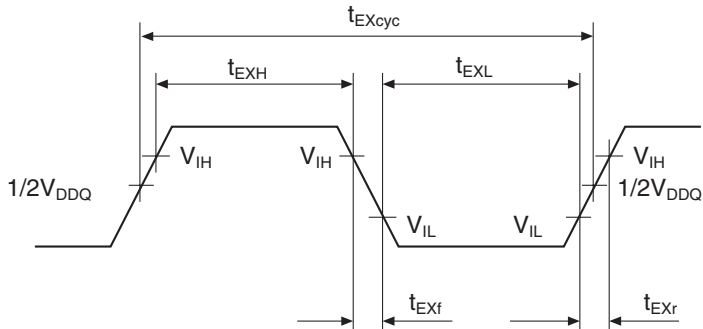
- * When the oscillation settling time of the crystal resonator is 1 ms or less.

Table 23.18 Clock and Control Signal Timing (HD6417751BP167 (V), HD6417751F167 (V))
 $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.8$ V, $T_a = -20$ to 75°C , $C_L = 30$ pF

| Item | | | Symbol | Min | Max | Unit | Figure |
|---|-------------------------|---------------------------|--------------|------|------|---------------|------------------------|
| EXTAL clock input frequency | PLL1/PLL2 operating | 1/2 divider operating | f_{EX} | 30 | 56 | MHz | |
| | | 1/2 divider not operating | f_{EX} | 15 | 28 | | |
| | PLL1/PLL2 not operating | 1/2 divider operating | f_{EX} | 2 | 56 | | |
| | | 1/2 divider not operating | f_{EX} | 1 | 28 | | |
| EXTAL clock input cycle time | | | t_{EXcyc} | 17.8 | 1000 | ns | 23.1 |
| EXTAL clock input low-level pulse width | | | t_{EXL} | 3.5 | — | ns | 23.1 |
| EXTAL clock input high-level pulse width | | | t_{EXH} | 3.5 | — | ns | 23.1 |
| EXTAL clock input rise time | | | t_{EXr} | — | 4 | ns | 23.1 |
| EXTAL clock input fall time | | | t_{EXf} | — | 4 | ns | 23.1 |
| CKIO clock output | PLL2 operating | | f_{OP} | 30 | 84 | MHz | |
| | PLL2 not operating | | f_{OP} | 1 | 84 | MHz | |
| CKIO clock output cycle time | | | t_{cyc} | 11.9 | 1000 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | | | t_{CKOL1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output high-level pulse width | | | t_{CKOH1} | 1 | — | ns | 23.2 (1) |
| CKIO clock output rise time | | | t_{CKOr} | — | 3 | ns | 23.2 (1) |
| CKIO clock output fall time | | | t_{CKOf} | — | 3 | ns | 23.2 (1) |
| CKIO clock output low-level pulse width | | | t_{CKOL2} | 3 | — | ns | 23.2 (2) |
| CKIO clock output high-level pulse width | | | t_{CKOH2} | 3 | — | ns | 23.2 (2) |
| Power-on oscillation settling time | | | t_{OSC1} | 10 | — | ms | 23.3, 23.5 |
| Power-on oscillation settling time/mode settling | | | t_{OSCMD} | 10 | — | ms | 23.3, 23.5 |
| MD reset setup time | | | t_{MDRS} | 3 | — | t_{cyc} | |
| MD reset hold time | | | t_{MDRH} | 20 | — | ns | 23.3, 23.5 |
| RESET assert time | | | t_{RESW} | 20 | — | t_{cyc} | 23.3, 23.4, 23.5, 23.6 |
| PLL synchronization settling time | | | t_{PLL} | 200 | — | μs | 23.9, 23.10 |
| Standby return oscillation settling time 1 | | | t_{OSC2} | 10 | — | ms | 23.4, 23.6 |
| Standby return oscillation settling time 2 | | | t_{OSC3} | 5 | — | ms | 23.7 |
| Standby return oscillation settling time 3 | | | t_{OSC4} | 5 | — | ms | 23.8 |
| IRL interrupt determination time (RTC used, standby mode) | | | t_{IRLSTB} | — | 200 | μs | 23.10 |
| TRST reset hold time | | | t_{TRSTRH} | 0 | — | ns | 23.3, 23.5 |

Notes: When a crystal resonator is connected to EXTAL and XTAL, the maximum frequency is 28 MHz. When a 3rd overtone crystal resonator is used, an external tank circuit is necessary.

As there is feedback from the CKIO pin when PLL2 is operating, the load capacitance connected to the CKIO pin should be a maximum of 50 pF.



Note: When the clock is input from the EXTAL pin

Figure 23.1 EXTAL Clock Input Timing

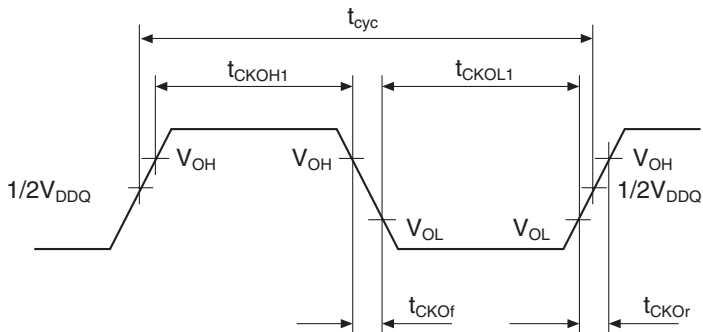


Figure 23.2 (1) CKIO Clock Output Timing

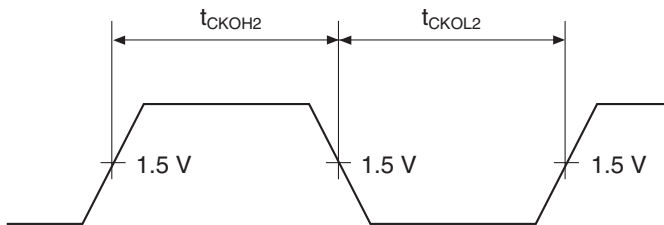


Figure 23.2 (2) CKIO Clock Output Timing

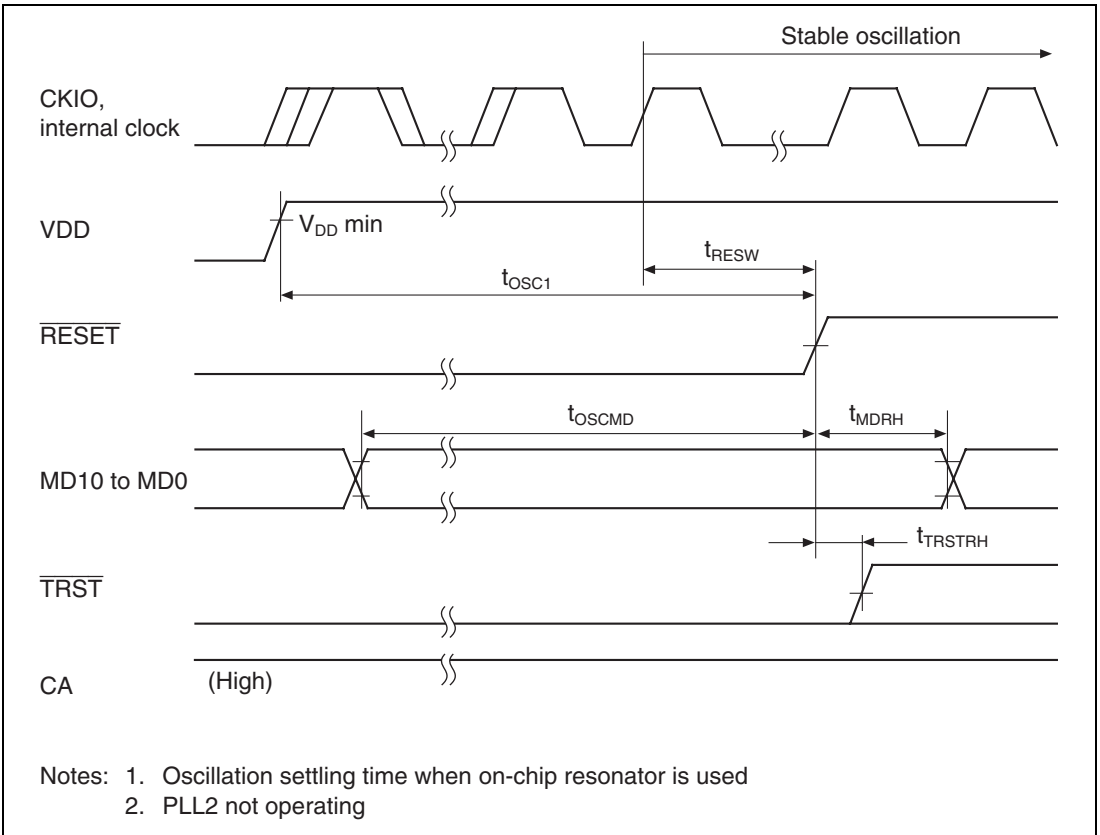


Figure 23.3 Power-On Oscillation Settling Time

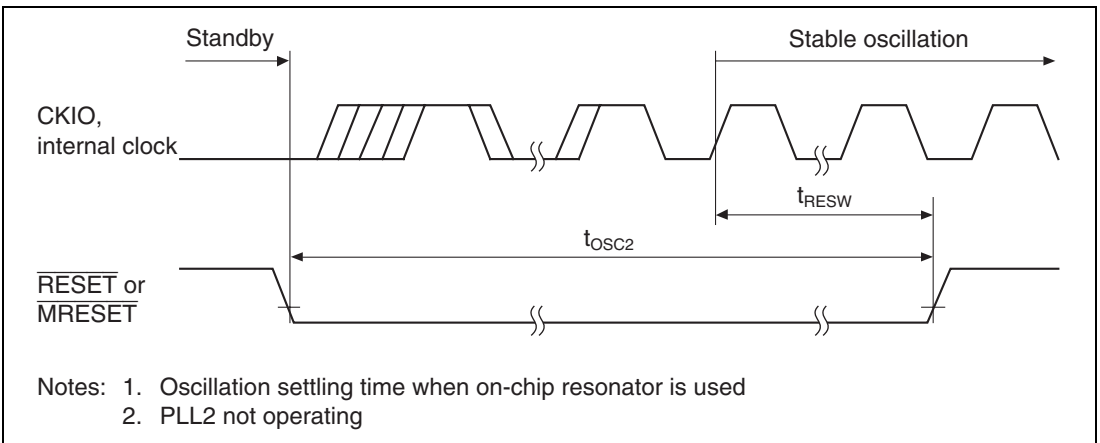


Figure 23.4 Standby Return Oscillation Settling Time (Return by $\overline{\text{RESET}}$ or $\overline{\text{MRESET}}$)

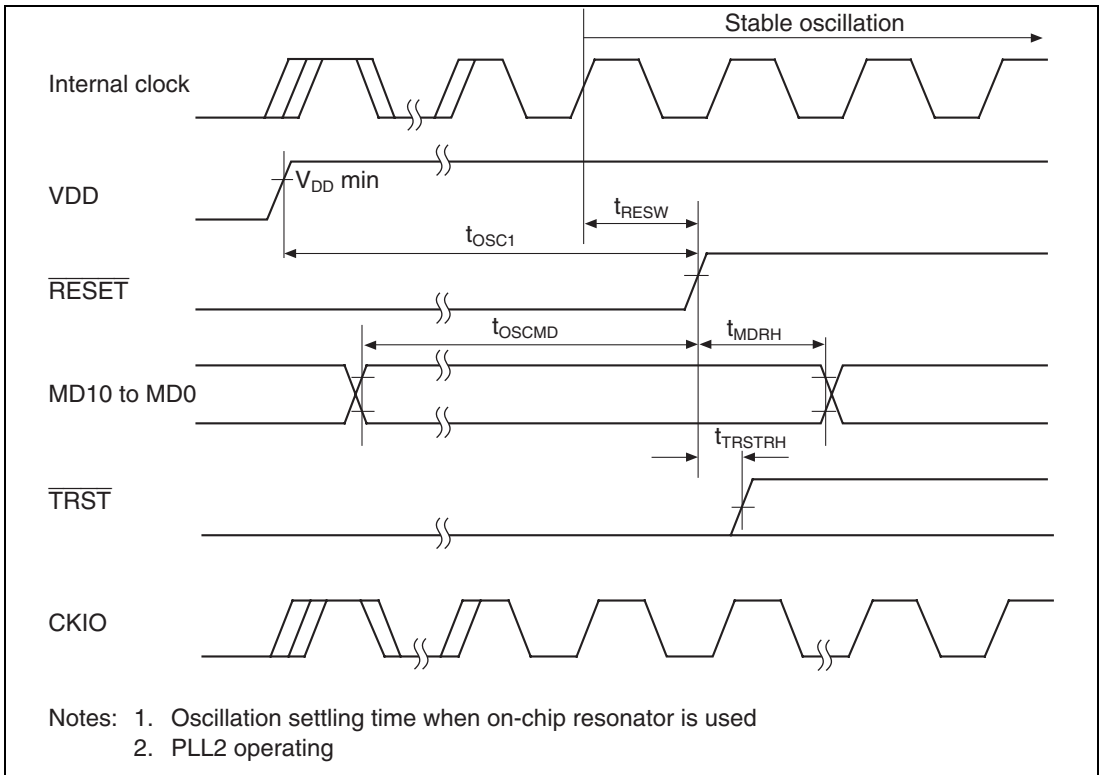


Figure 23.5 Power-On Oscillation Settling Time

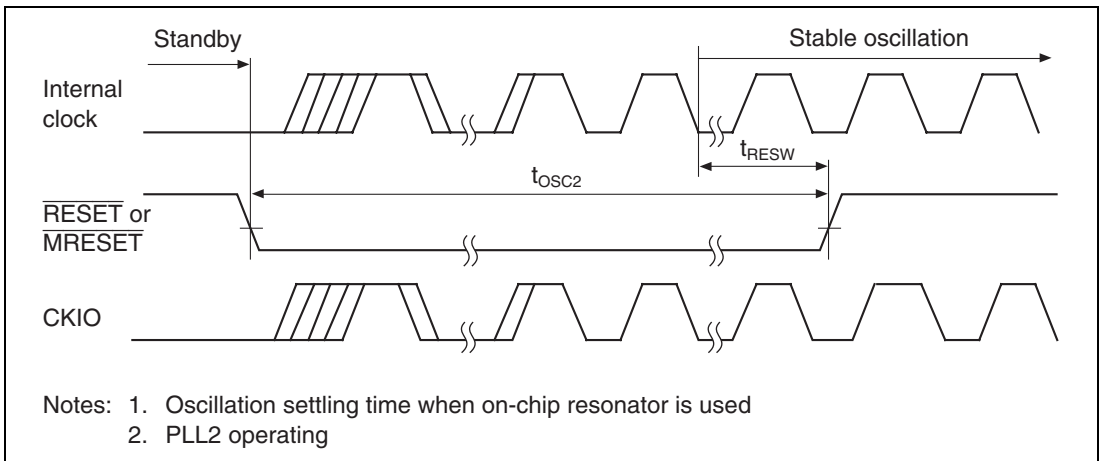


Figure 23.6 Standby Return Oscillation Settling Time (Return by $\overline{\text{RESET}}$ or $\overline{\text{MRESET}}$)

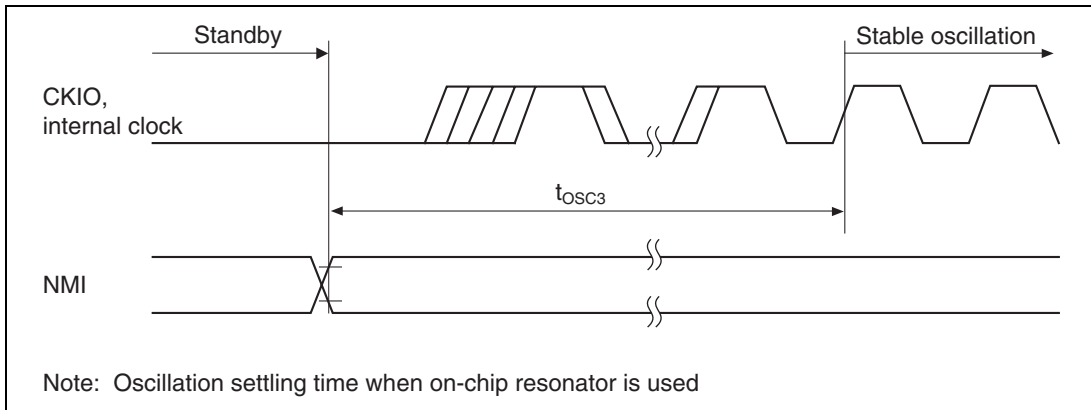


Figure 23.7 Standby Return Oscillation Settling Time (Return by NMI)

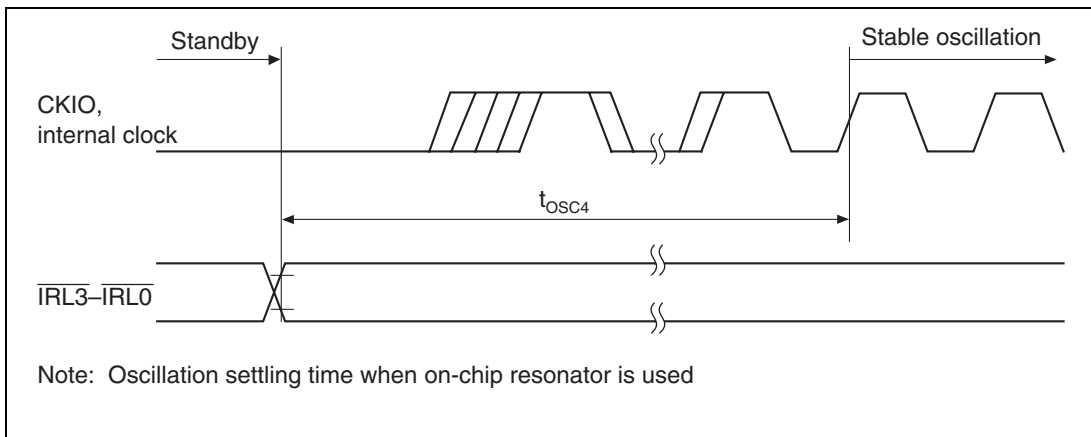


Figure 23.8 Standby Return Oscillation Settling Time (Return by $\overline{IRL3}$ - $\overline{IRL0}$)

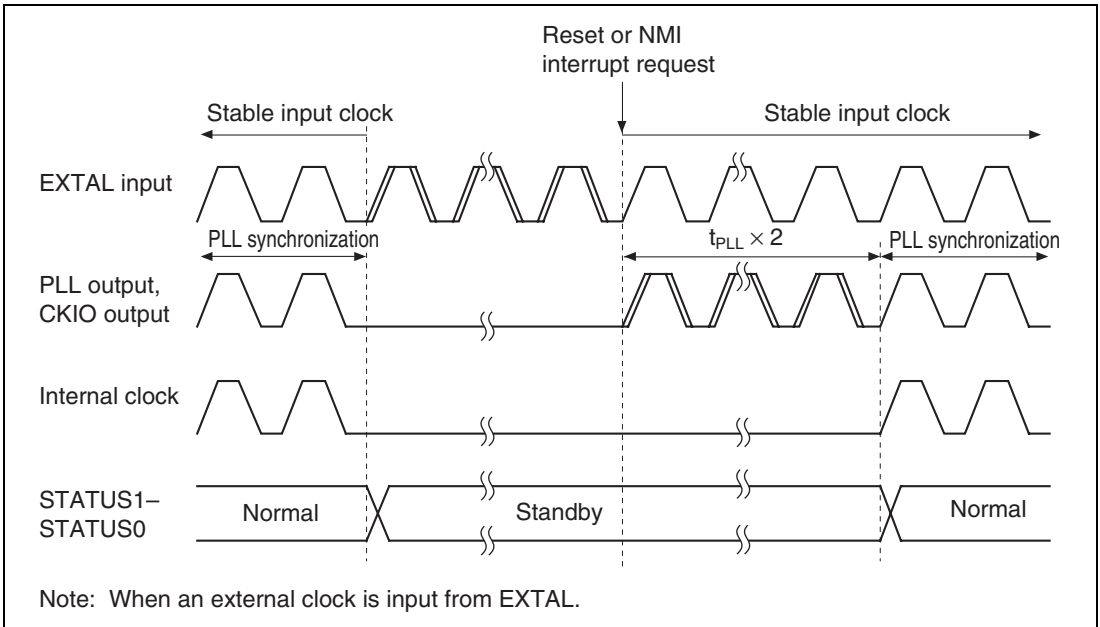


Figure 23.9 PLL Synchronization Settling Time in Case of $\overline{\text{RESET}}$, $\overline{\text{MRESET}}$ or NMI Interrupt

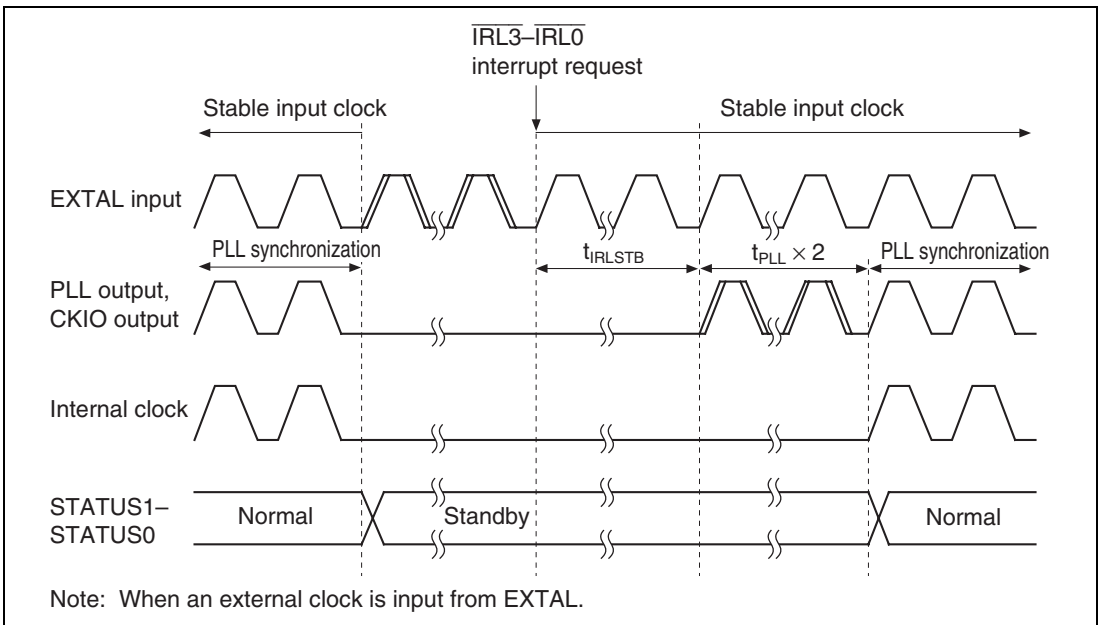


Figure 23.10 PLL Synchronization Settling Time in Case of IRL Interrupt

23.3.2 Control Signal Timing

Table 23.19 Control Signal Timing

| Item | Symbol | HD6417751 RBP240 (V) | | HD6417751 RBP200 (V) | | HD6417751 RBA240HV*2 | | HD6417751 RF240 (V) | | HD6417751 RF200 (V) | | Unit | Figure |
|--|--------------------|-------------------------|-----|-------------------------|-----|-------------------------|-----|------------------------|-----|------------------------|-----|------------------|---------------|
| | | Min | Max | Min | Max | Min | Max | Min | Max | Min | Max | | |
| BREQ setup time | t_{BREQS} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | 3.5 | — | ns | 23.11 |
| BREQ hold time | t_{BREQH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | 23.11 |
| BACK delay time | t_{BACKD} | — | 5.3 | — | 5.3 | — | 6 | — | 6 | — | 6 | ns | 23.11 |
| Bus tri-state delay time | t_{BOFF1} | — | 12 | — | 12 | — | 12 | — | 12 | — | 12 | ns | 23.11 |
| Bus tri-state delay time to standby mode | t_{BOFF2} | — | 2 | — | 2 | — | 2 | — | 2 | — | 2 | t_{cyc} | 23.12 (2) |
| Bus buffer on time | t_{BON1} | — | 12 | — | 12 | — | 12 | — | 12 | — | 12 | ns | 23.11 |
| Bus buffer on time from standby | t_{BON2} | — | 2 | — | 2 | — | 2 | — | 2 | — | 2 | t_{cyc} | 23.12 (2) |
| STATUS 0/1 delay time | t_{STD1} | — | 6 | — | 6 | — | 6 | — | 6 | — | 6 | ns | 23.12 (1) |
| | t_{STD2} | — | 2 | — | 2 | — | 2 | — | 2 | — | 2 | t_{cyc} | 23.12 (1) (2) |
| | t_{STD3} | — | 2 | — | 2 | — | 2 | — | 2 | — | 2 | t_{cyc} | 23.12 (2) |

Notes: 1. $V_{\text{DDQ}} = 3.0$ to 3.6 V, $V_{\text{DD}} = 1.5$ V, $T_a = -20$ to 75°C^{*3} , $C_L = 30$ pF, PLL2 on

2. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.

3. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.20 Control Signal Timing

| | | HD6417751BP167 (V) HD6417751F167 (V) | | | |
|--|--------------------|---|-----|------------------|---------------|
| | | * | | | |
| Item | Symbol | Min | Max | Unit | Figure |
| $\overline{\text{BREQ}}$ setup time | t_{BREQS} | 3.5 | — | ns | 23.11 |
| $\overline{\text{BREQ}}$ hold time | t_{BREQH} | 1.5 | — | ns | 23.11 |
| $\overline{\text{BACK}}$ delay time | t_{BACKD} | — | 8 | ns | 23.11 |
| Bus tri-state delay time | t_{BOFF1} | — | 12 | ns | 23.11 |
| Bus tri-state delay time to standby mode | t_{BOFF2} | — | 2 | t_{cyc} | 23.12 (2) |
| Bus buffer on time | t_{BON1} | — | 12 | ns | 23.11 |
| Bus buffer on time from standby | t_{BON2} | — | 2 | t_{cyc} | 23.12 (2) |
| STATUS 0/1 delay time | t_{STD1} | — | 6 | ns | 23.12 (1) |
| | t_{STD2} | — | 2 | t_{cyc} | 23.12 (1) (2) |
| | t_{STD3} | — | 2 | t_{cyc} | 23.12 (2) |

Note: * $V_{\text{DDQ}} = 3.0$ to 3.6 V, $V_{\text{DD}} = 1.8$ V, $T_a = -20$ to 75°C , $C_L = 30$ pF, PLL2 on

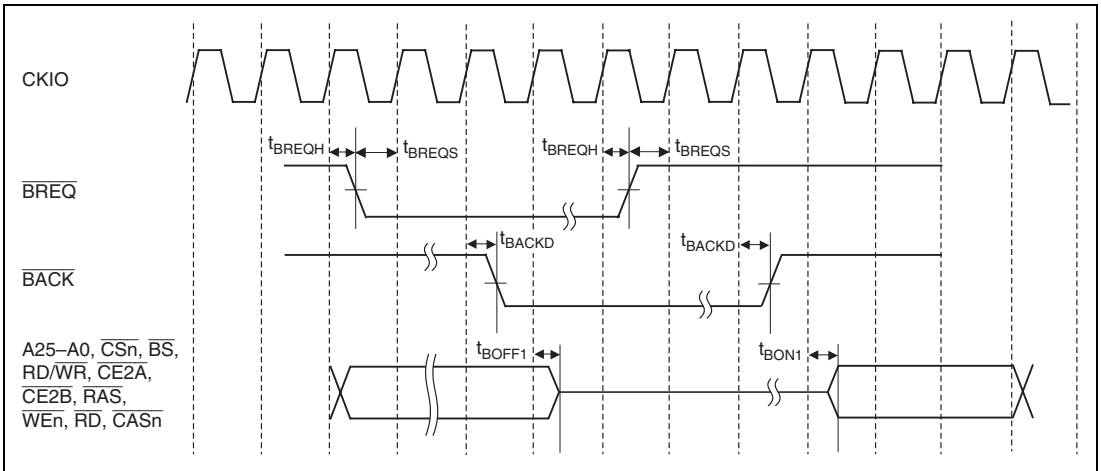


Figure 23.11 Control Signal Timing

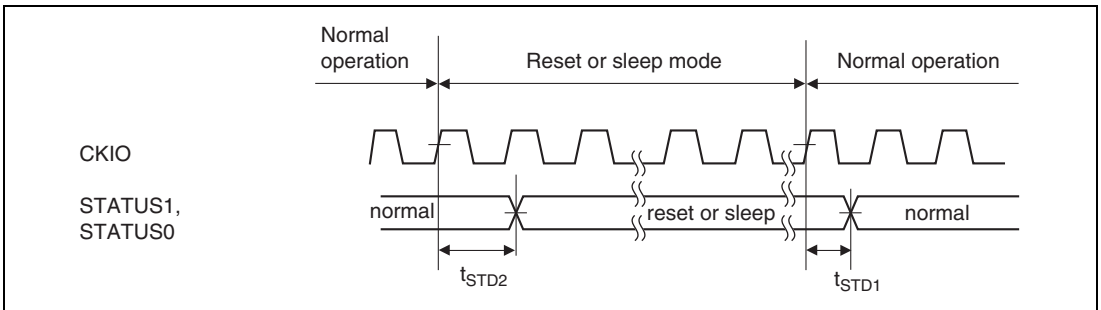


Figure 23.12 (1) Pin Drive Timing for Reset or Sleep Mode

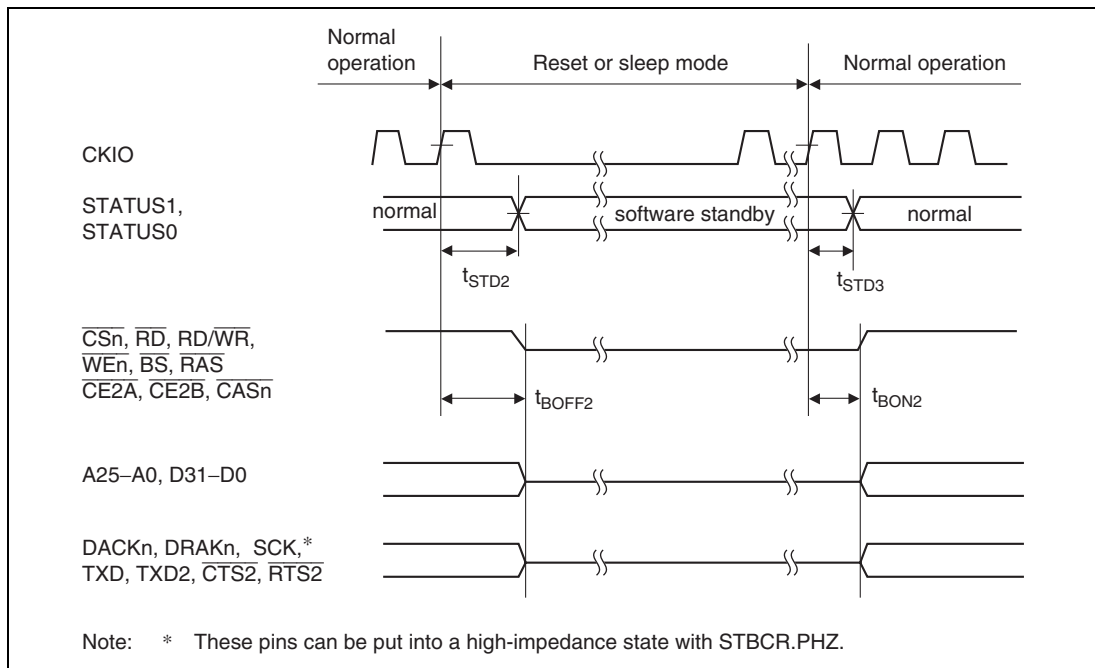


Figure 23.12 (2) Pin Drive Timing for Software Standby Mode

23.3.3 Bus Timing

Table 23.21 Bus Timing (1)

| Item | Symbol | HD6417751 RBP240 (V) | | HD6417751 RBP200 (V) | | HD6417751 RBG240 (V) | | HD6417751 RBG200 (V) | | Unit | Notes |
|--|--------------|-------------------------|-----|-------------------------|-----|-------------------------|-----|-------------------------|-----|------|-------------------------------|
| | | Min | Max | Min | Max | Min | Max | Min | Max | | |
| Address delay time | t_{AD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| \overline{BS} delay time | t_{BSD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| \overline{CS} delay time | t_{CSD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| \overline{RW} delay time | t_{RWD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| \overline{RD} delay time | t_{RSD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| Read data setup time | t_{RDS} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | |
| Read data hold time | t_{RDH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | |
| \overline{WE} delay time (falling edge) | t_{WEDF} | — | 5.3 | — | 5.3 | — | 6 | — | 6 | ns | Relative to CKIO falling edge |
| \overline{WE} delay time | t_{WED1} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| Write data delay time | t_{WDD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| \overline{RDY} setup time | t_{RDYS} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | |
| \overline{RDY} hold time | t_{RDYH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | |
| \overline{RAS} delay time | t_{RASD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| \overline{CAS} delay time 1 | t_{CASD1} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | DRAM |
| \overline{CAS} delay time 2 | t_{CASD2} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | SDRAM |
| CKE delay time | t_{CKED} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | SDRAM |
| DQM delay time | t_{DQMD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | SDRAM |
| \overline{FRAME} delay time | t_{FMD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | MPX |
| $\overline{IOIS16}$ setup time | t_{IO16S} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | PCMCIA |
| $\overline{IOIS16}$ hold time | t_{IO16H} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | PCMCIA |
| \overline{ICIORW} delay time (falling edge) | t_{ICWSDF} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | PCMCIA |
| \overline{ICIORW} delay time | t_{ICRSD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | PCMCIA |

| Item | Symbol | HD6417751 RBP240 (V) | | HD6417751 RBP200 (V) | | HD6417751 RBG240 (V) | | HD6417751 RBG200 (V) | | Unit | Notes |
|-----------------------------------|--------------|-------------------------|-----|-------------------------|-----|-------------------------|-----|-------------------------|-----|------|--|
| | | Min | Max | Min | Max | Min | Max | Min | Max | | |
| DACK delay time | t_{DACK} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| DACK delay time (falling edge) | t_{DACKDF} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | Relative to CKIO falling edge |
| DTR setup time | t_{DTRS} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | |
| DTR hold time | t_{DTRH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | |
| DBREQ setup time | t_{DBQS} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | |
| DBREQ hold time | t_{DBQH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | |
| TR setup time | t_{TRS} | 2.0 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | |
| TR hold time | t_{TRH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | |
| BAVL delay time | t_{BAVD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| TDACK delay time | t_{TDAD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |
| ID1, ID0 delay time | t_{IDD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | |

- Notes: 1. $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to $+75^\circ\text{C}^{*3}$, $C_L = 30$ pF, PLL2 on
2. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
3. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.22 Bus Timing (2)

HD6417751BP167 (V)
HD6417751F167 (V)

*

| Item | Symbol | Min | Max | Unit | Notes |
|---|--------------|-----|-----|------|-------------------------------------|
| Address delay time | t_{AD} | 1.0 | 8 | ns | |
| \overline{BS} delay time | t_{BSD} | 1.0 | 8 | ns | |
| \overline{CS} delay time | t_{CSD} | 1.0 | 8 | ns | |
| \overline{RW} delay time | t_{RWD} | 1.0 | 8 | ns | |
| \overline{RD} delay time | t_{RSD} | 1.0 | 8 | ns | |
| Read data setup time | t_{RDS} | 3.5 | — | ns | |
| Read data hold time | t_{RDH} | 1.5 | — | ns | |
| \overline{WE} delay time (falling edge) | t_{WEDF} | 1.0 | 8 | ns | Relative to CKIO falling edge |
| \overline{WE} delay time | t_{WED1} | 1.0 | 8 | ns | |
| Write data delay time | t_{WDD} | 1.0 | 8 | ns | |
| \overline{RDY} setup time | t_{RDYS} | 3.5 | — | ns | |
| \overline{RDY} hold time | t_{RDYH} | 1.5 | — | ns | |
| \overline{RAS} delay time | t_{RASD} | 1.0 | 8 | ns | |
| \overline{CAS} delay time 1 | t_{CASD1} | 1.0 | 8 | ns | DRAM |
| \overline{CAS} delay time 2 | t_{CASD2} | 1.0 | 8 | ns | SDRAM |
| CKE delay time | t_{CKED} | 1.0 | 8 | ns | SDRAM |
| DQM delay time | t_{DQMD} | 1.0 | 8 | ns | SDRAM |
| \overline{FRAME} delay time | t_{FMD} | 1.0 | 8 | ns | MPX |
| $\overline{IOIS16}$ setup time | t_{IO16S} | 3.5 | — | ns | PCMCIA |
| $\overline{IOIS16}$ hold time | t_{IO16H} | 1.5 | — | ns | PCMCIA |
| \overline{ICIOR} delay time (falling edge) | t_{ICWSDF} | 1.0 | 8 | ns | PCMCIA |
| \overline{ICIOR} delay time | t_{ICRSDF} | 1.0 | 8 | ns | PCMCIA |
| DACK delay time | t_{DACD} | 1.0 | 8 | ns | |
| DACK delay time (falling edge) | t_{DACDF} | 1.0 | 8 | ns | Relative to CKIO falling edge |
| DTR setup time | t_{DTRS} | 3.5 | — | ns | |

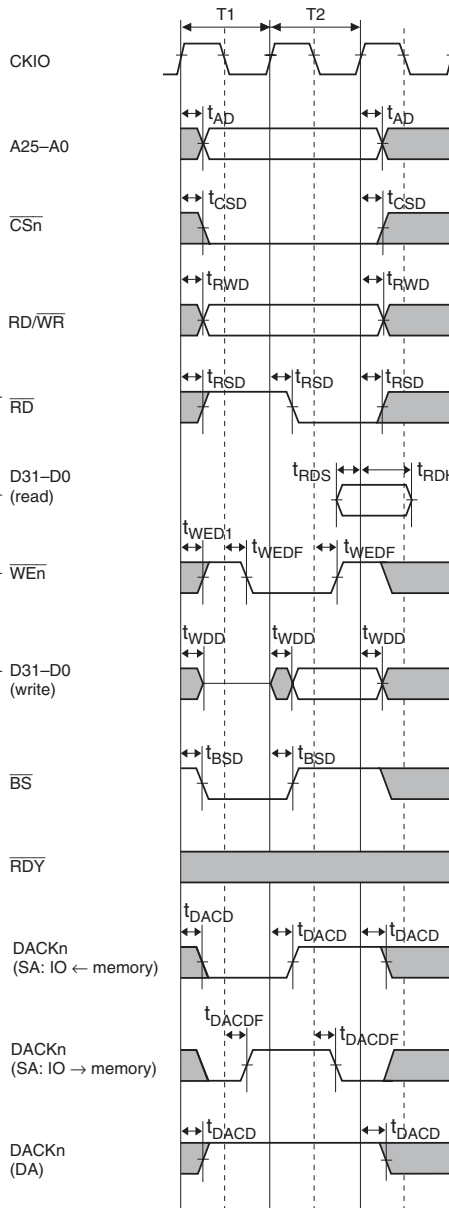
HD6417751BP167 (V)

HD6417751F167 (V)

*

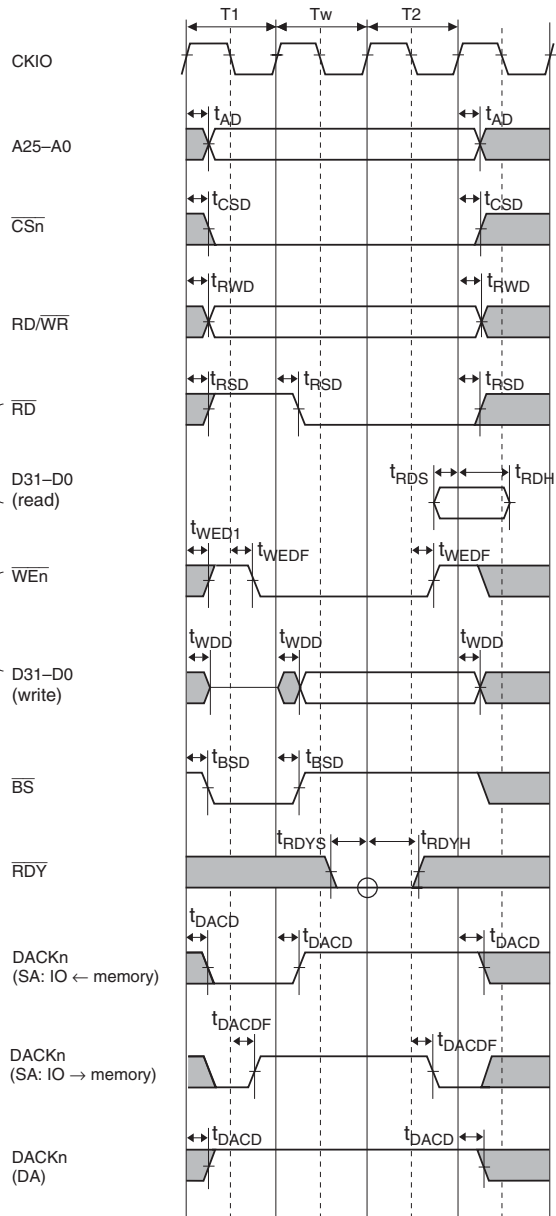
| Item | Symbol | Min | Max | Unit | Notes |
|-------------------------------|------------|-----|-----|------|-------|
| DTR hold time | t_{DTRH} | 1.5 | — | ns | |
| \overline{DBREQ} setup time | t_{DBQS} | 3.5 | — | ns | |
| \overline{DBREQ} hold time | t_{DBQH} | 1.5 | — | ns | |
| \overline{TR} setup time | t_{TRS} | 3.5 | — | ns | |
| \overline{TR} hold time | t_{TRH} | 1.5 | — | ns | |
| \overline{BAVL} delay time | t_{BAVD} | 1.0 | 8 | ns | |
| \overline{TDACK} delay time | t_{TDAD} | 1.0 | 8 | ns | |
| ID1, ID0 delay time | t_{IDD} | 1.0 | 8 | ns | |

Note: * $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.8$ V, $T_a = -20$ to $+75^\circ\text{C}$, $C_L = 30$ pF, PLL2 on



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.13 SRAM Bus Cycle: Basic Bus Cycle (No Wait)



Legend:

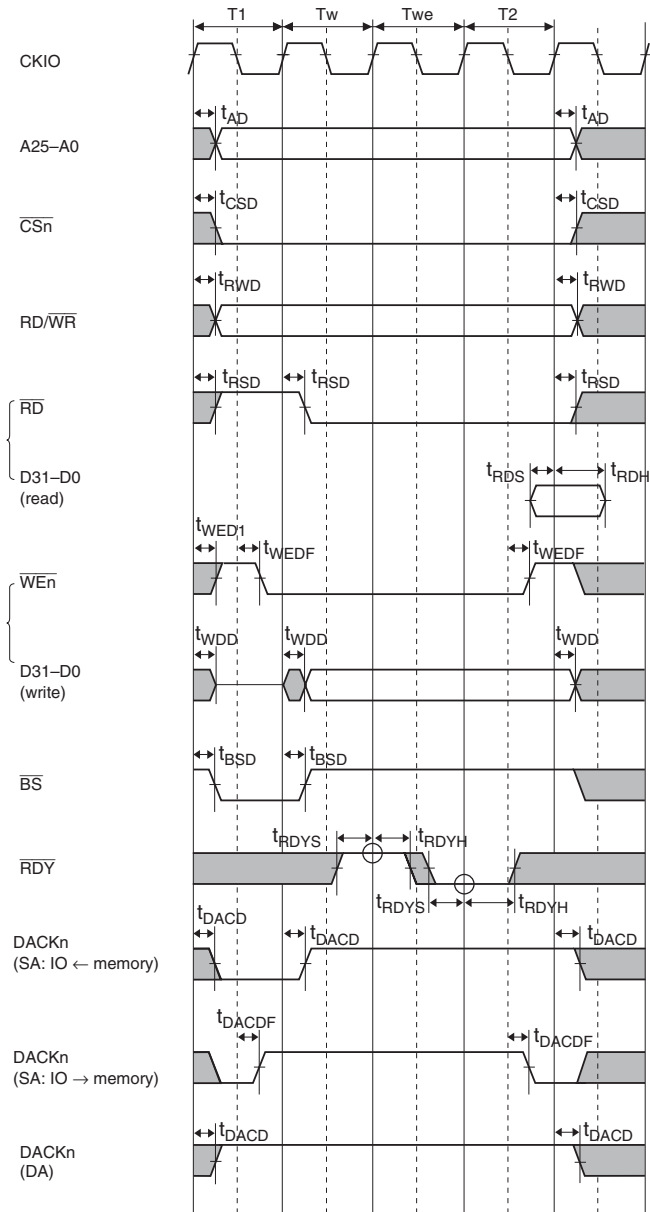
IO: DACK device

SA: Single address DMA transfer

DA: Dual address DMA transfer

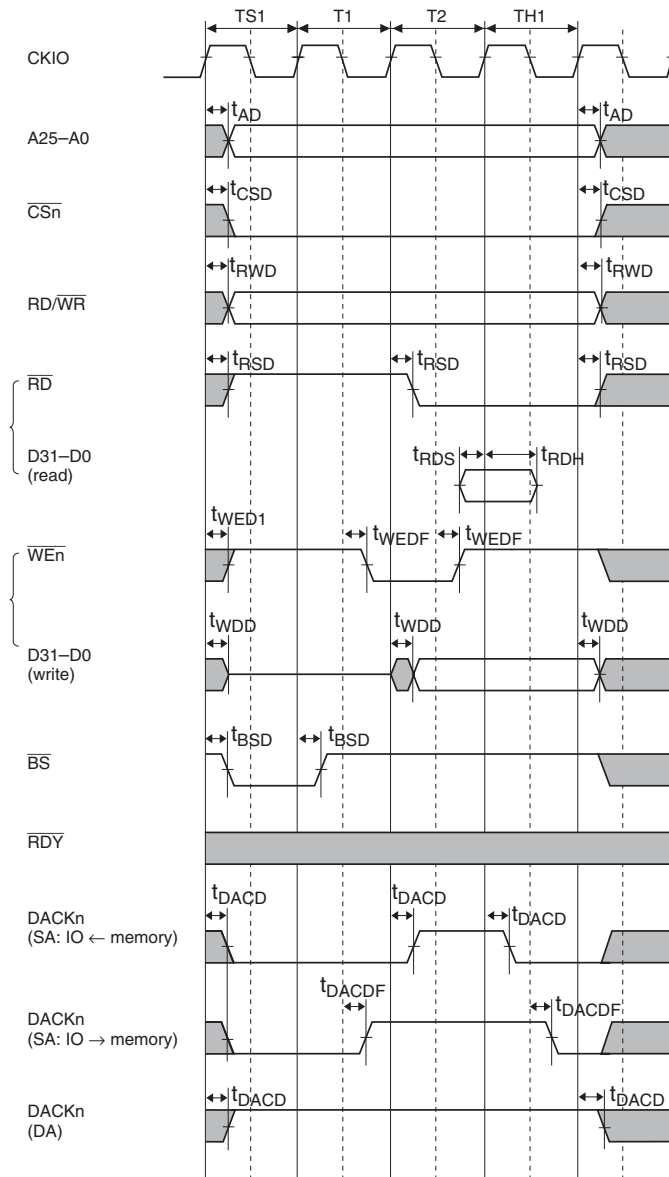
DACK set to active-high

Figure 23.14 SRAM Bus Cycle: Basic Bus Cycle (One Internal Wait)



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.15 SRAM Bus Cycle: Basic Bus Cycle (One Internal Wait + One External Wait)



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.16 SRAM Bus Cycle: Basic Bus Cycle (No Wait, Address Setup/Hold Time Insertion, AnS = 1, AnH = 1)

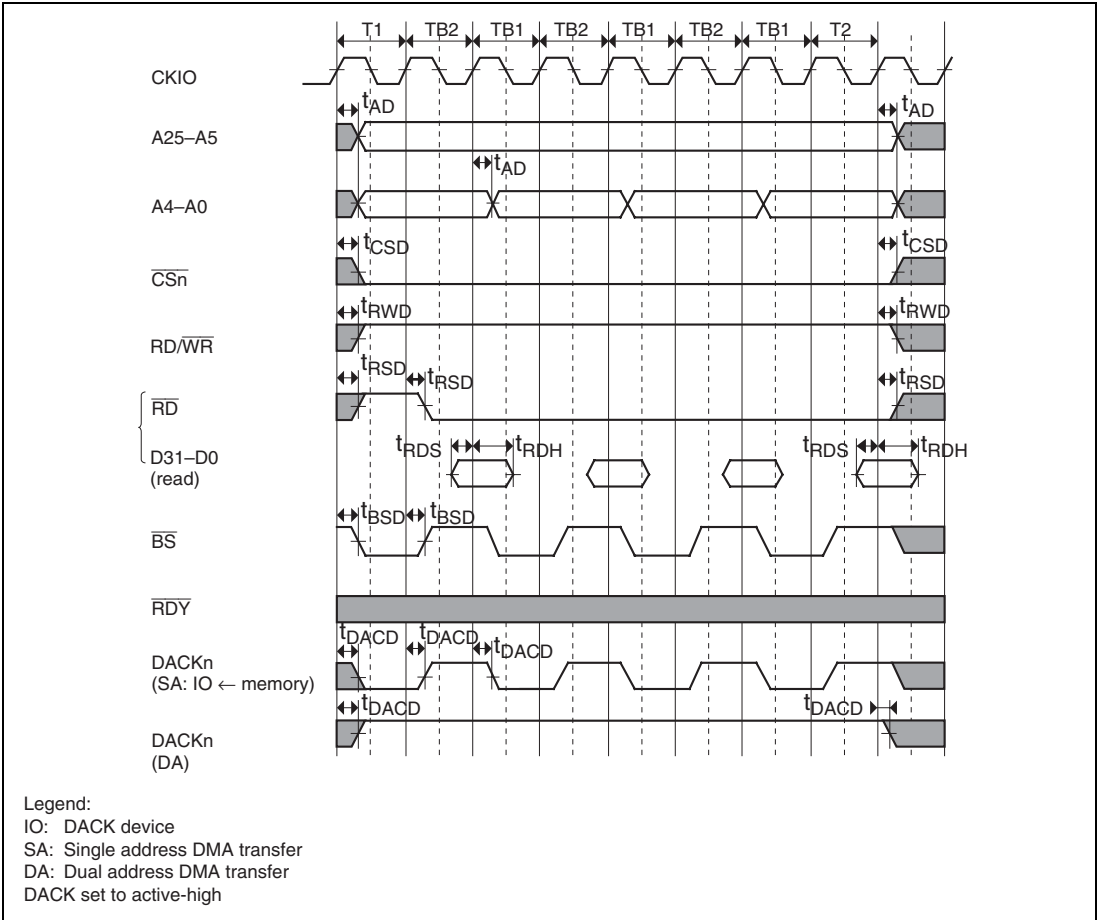


Figure 23.17 Burst ROM Bus Cycle (No Wait)

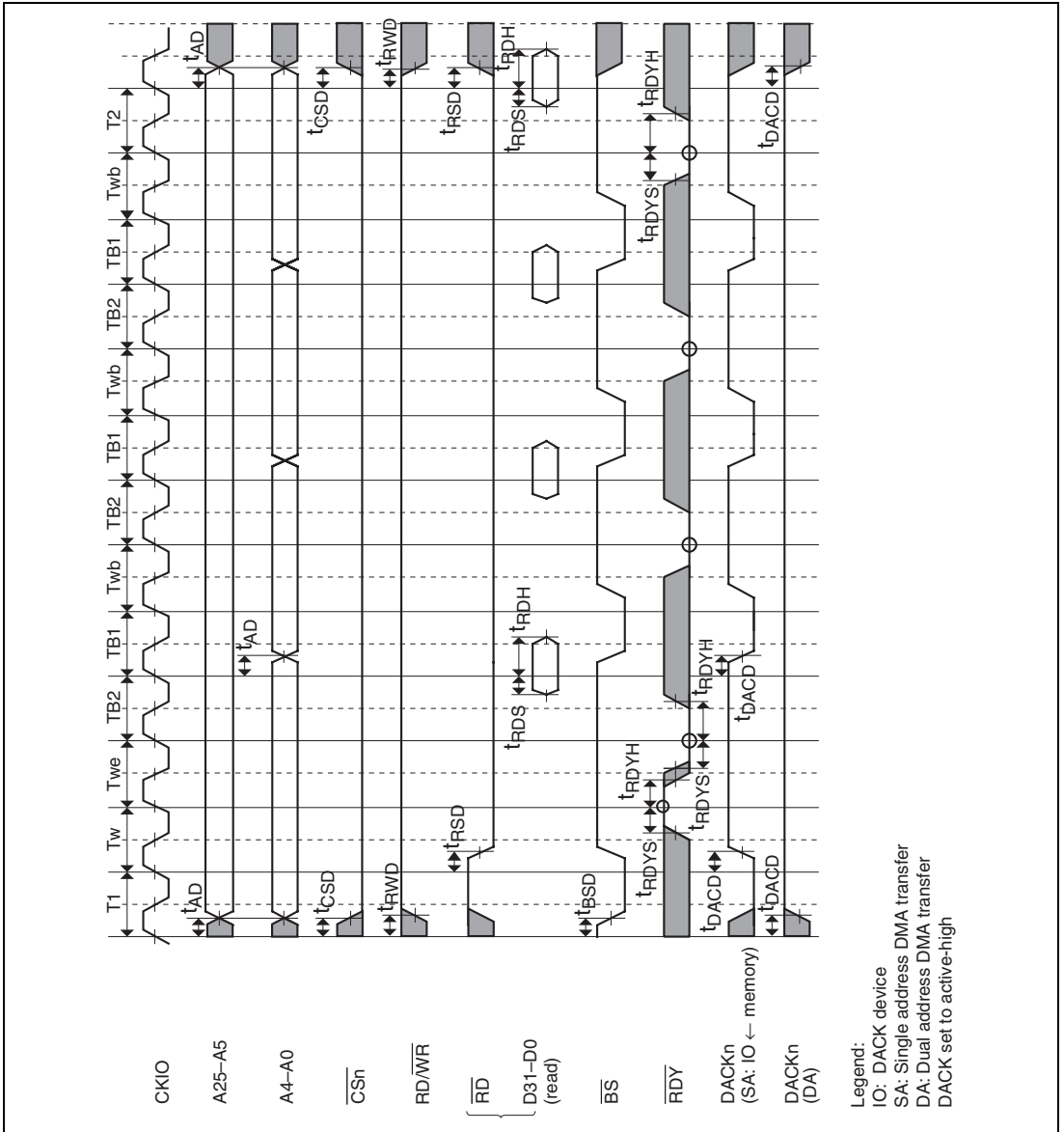


Figure 23.18 Burst ROM Bus Cycle

(1st Data: One Internal Wait + One External Wait; 2nd/3rd/4th Data: One Internal Wait)

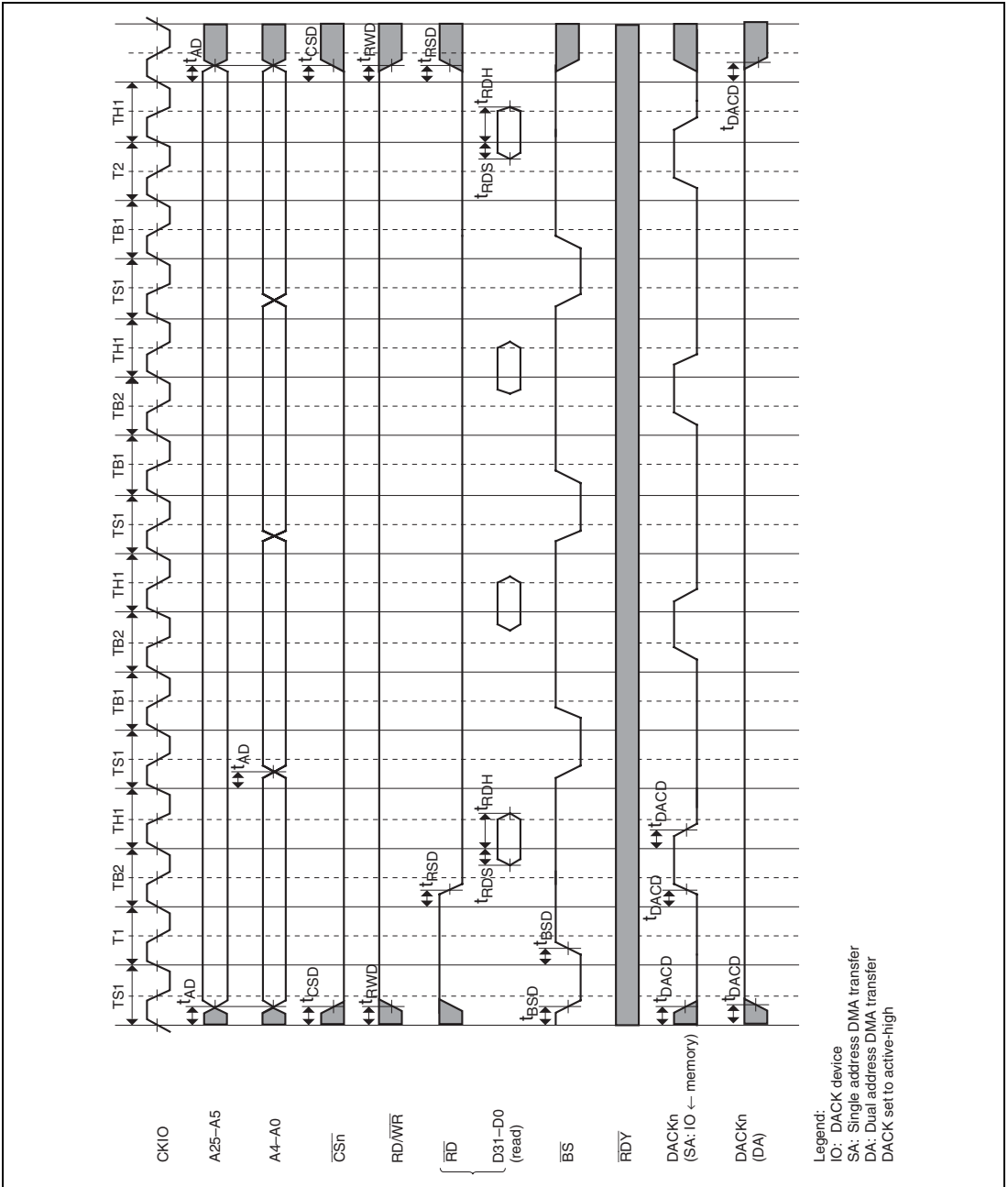


Figure 23.19 Burst ROM Bus Cycle
 (No Wait, Address Setup/Hold Time Insertion, AnS = 1, AnH = 1)

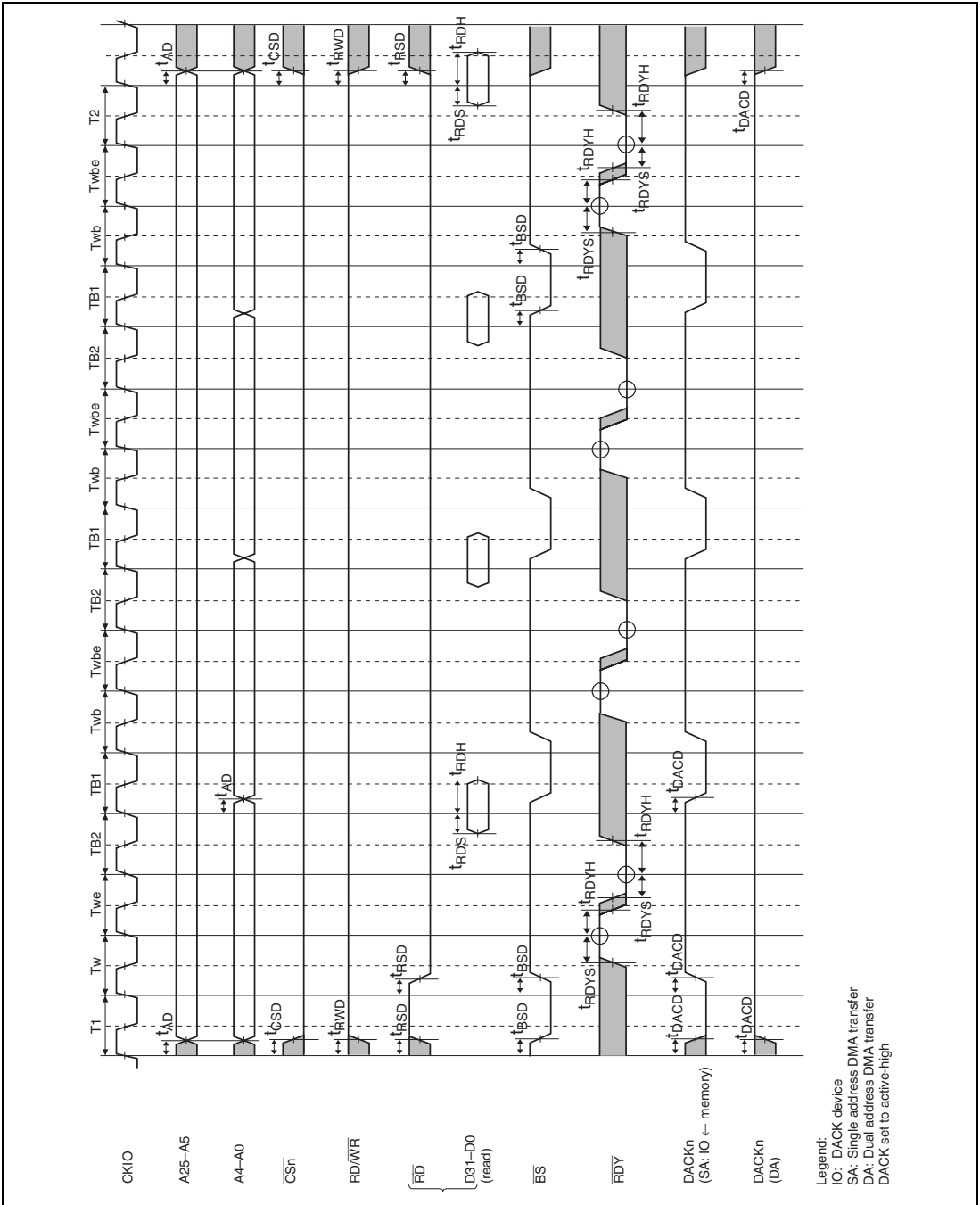


Figure 23.20 Burst ROM Bus Cycle (One Internal Wait + One External Wait)

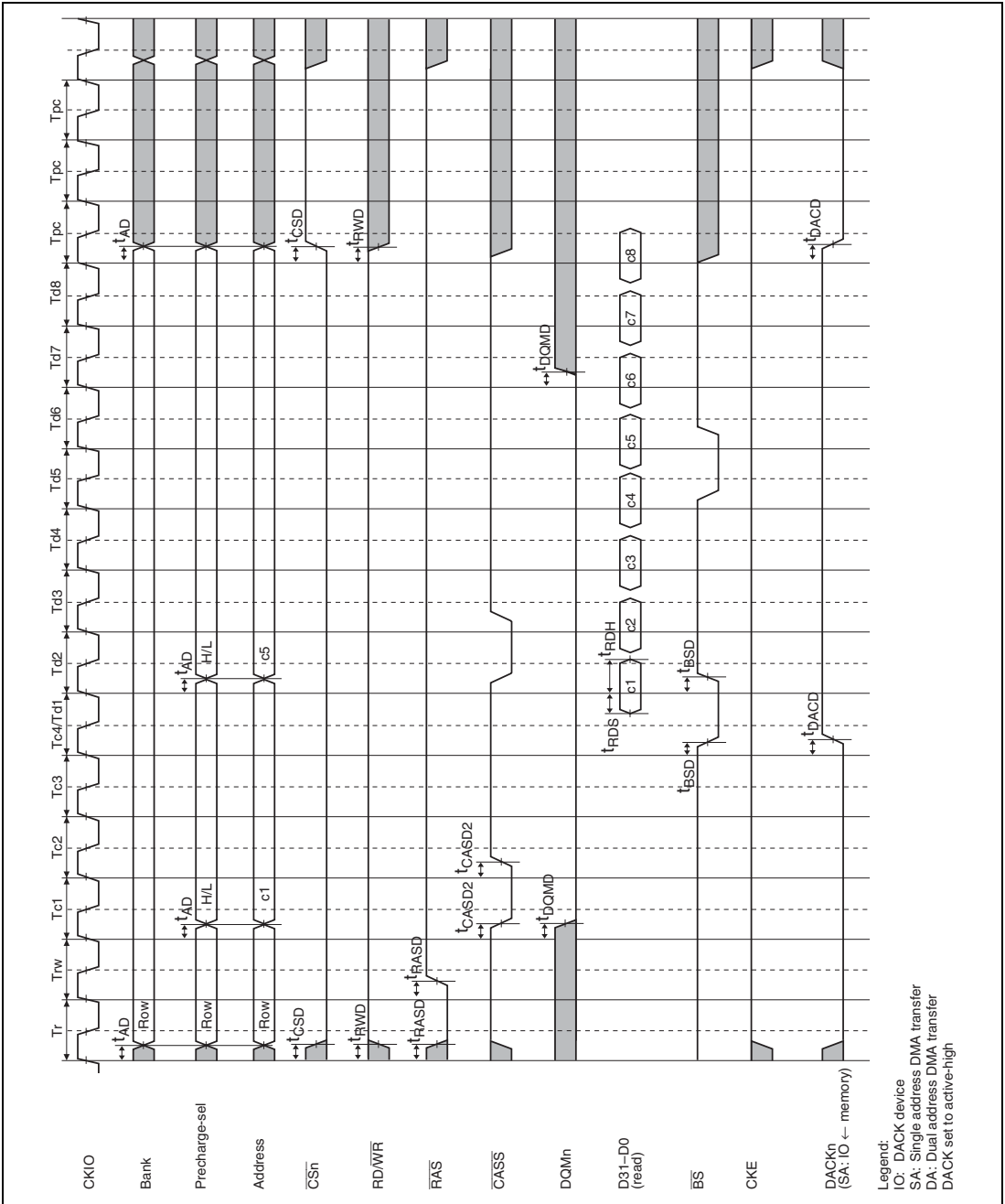


Figure 23.22 Synchronous DRAM Auto-Precharge Read Bus Cycle: Burst
(RCD [1:0] = 01, CAS Latency = 3, TPC [2:0] = 011)

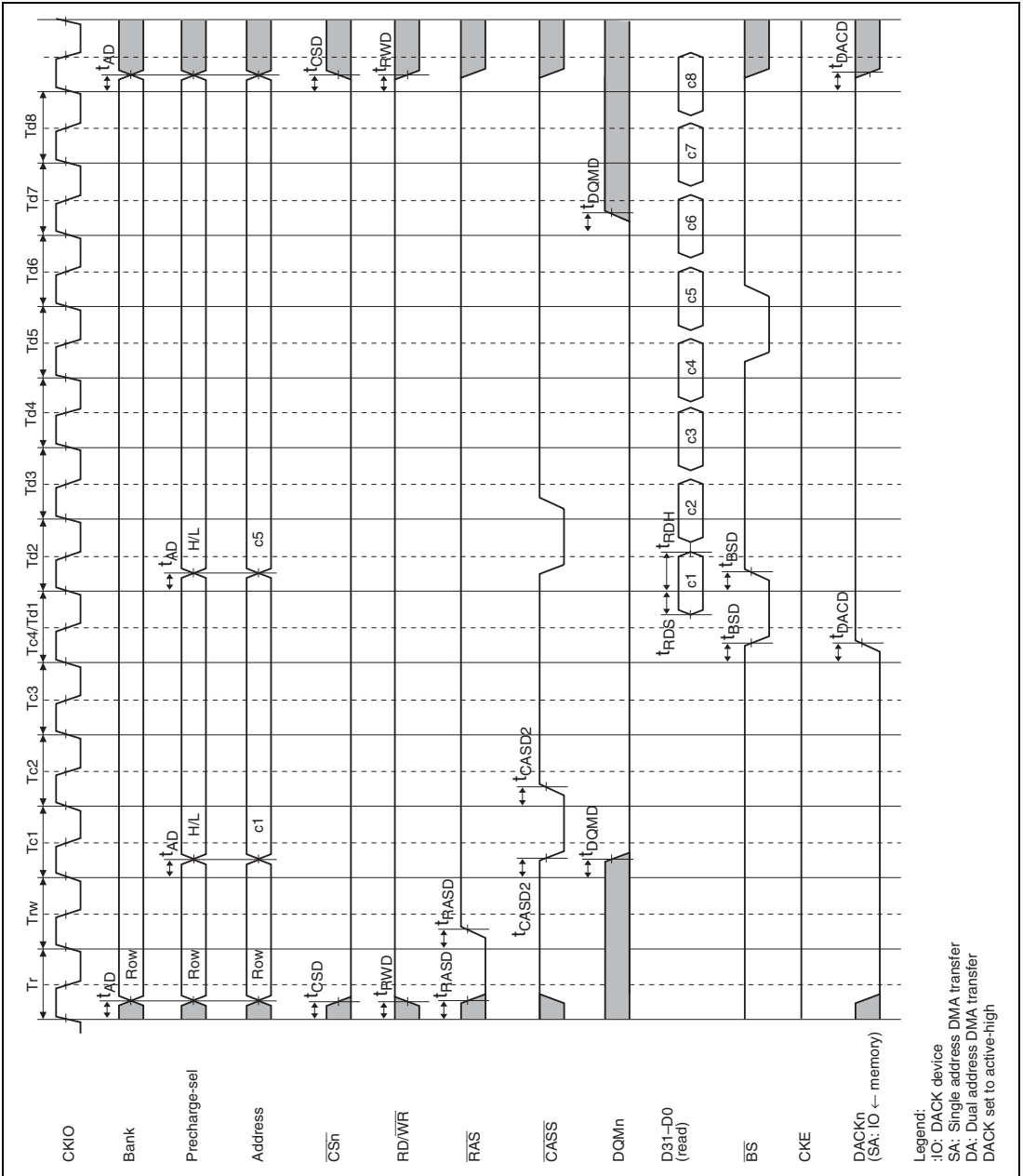


Figure 23.23 Synchronous DRAM Normal Read Bus Cycle: ACT + READ Commands, Burst (RASD = 1, RCD [1:0] = 01, CAS Latency = 3)

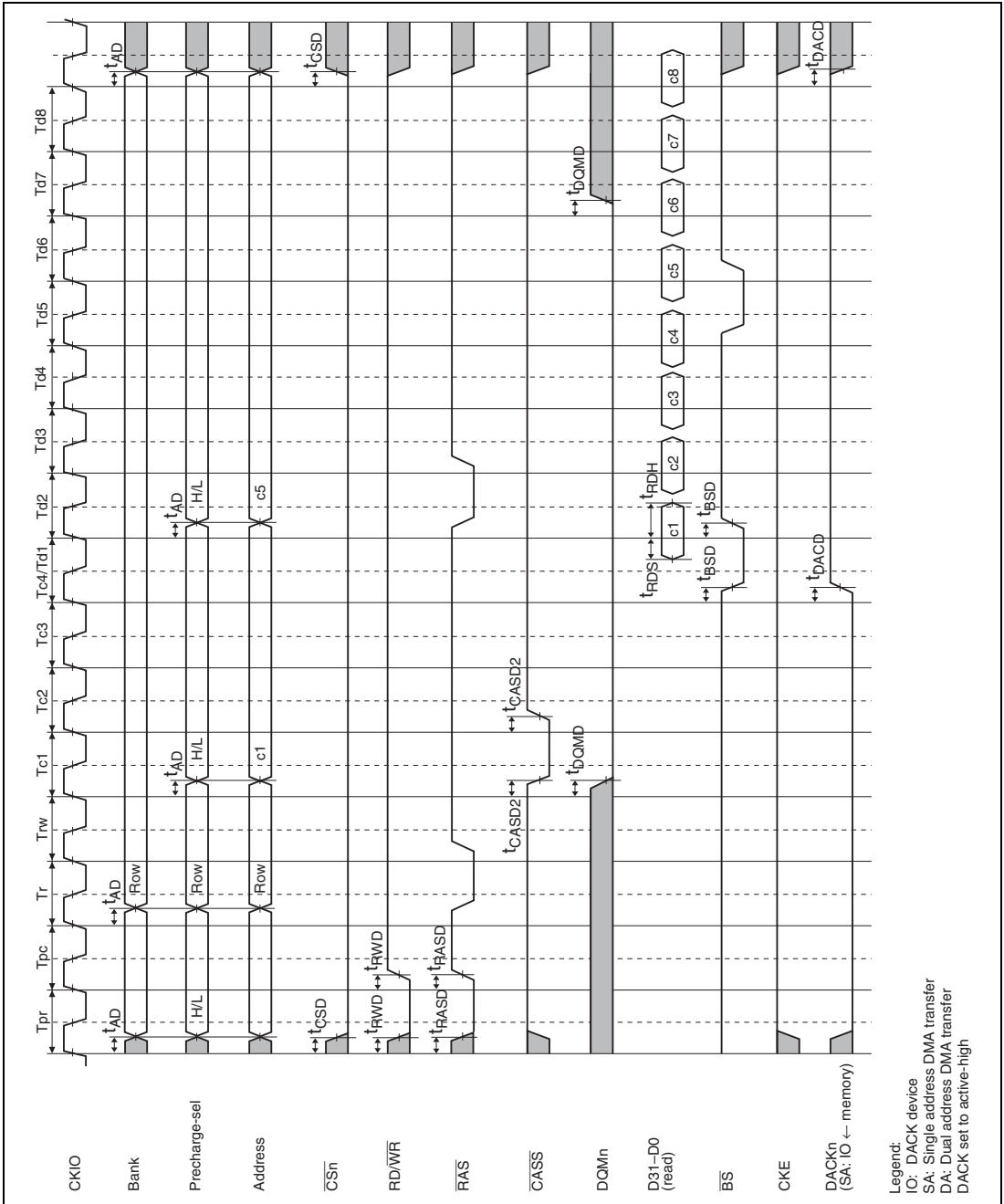


Figure 23.24 Synchronous DRAM Normal Read Bus Cycle: PRE + ACT + READ Commands, Burst (RASD = 1, RCD [1:0] = 01, TPC [2:0] = 001, CAS Latency = 3)

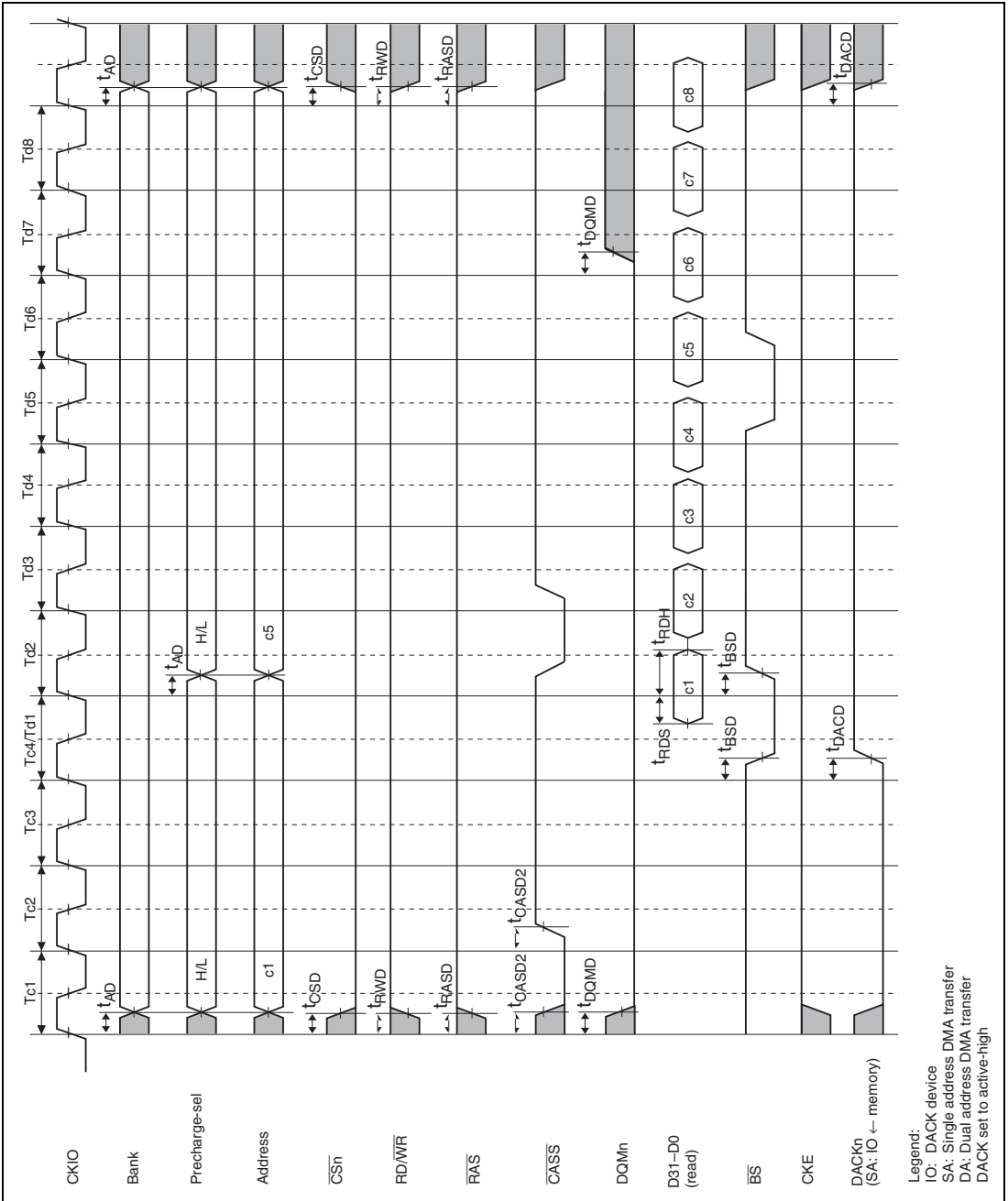


Figure 23.25 Synchronous DRAM Normal Read Bus Cycle: READ Command, Burst (RASD = 1, CAS Latency = 3)

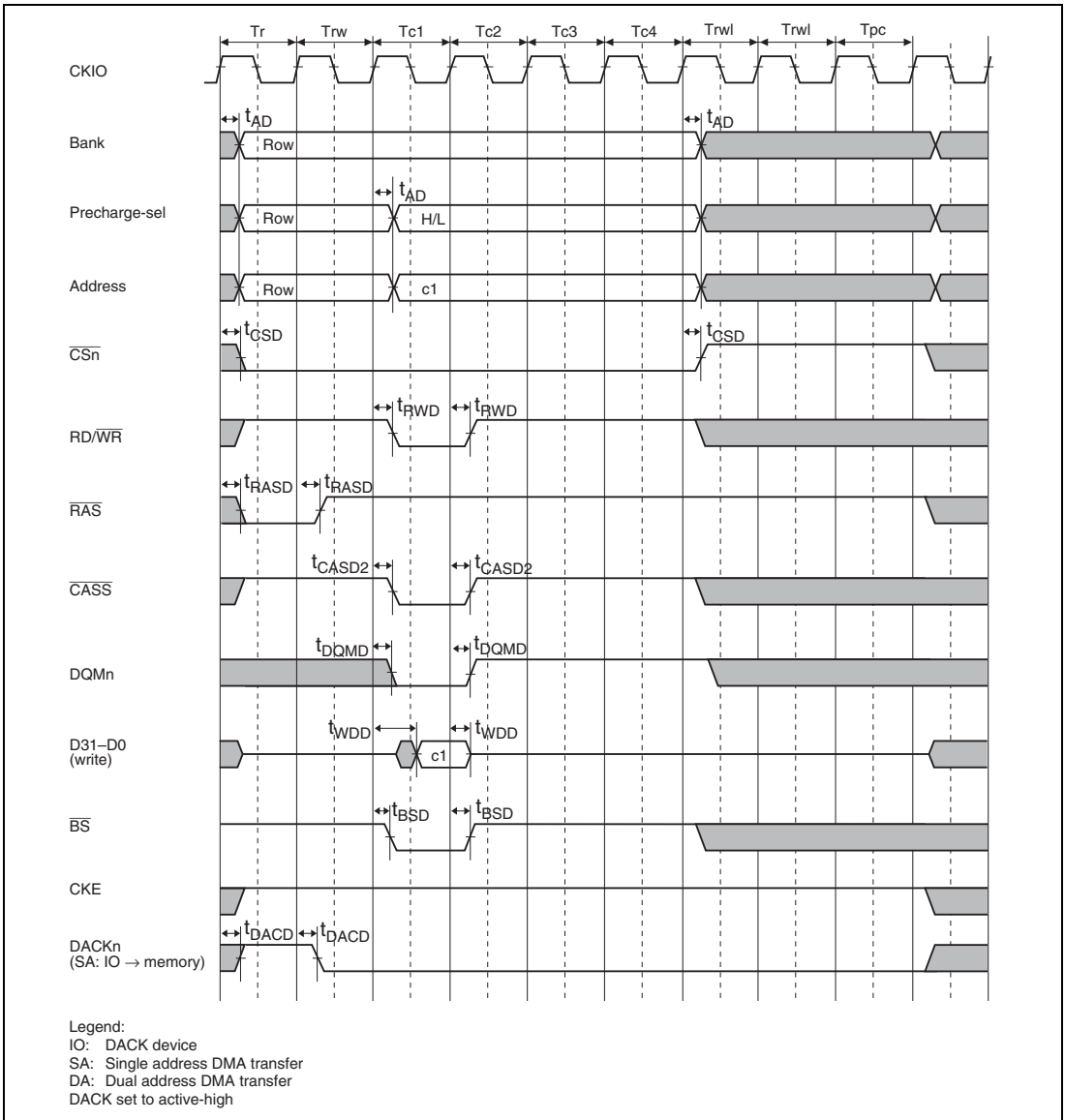


Figure 23.26 Synchronous DRAM Auto-Precharge Write Bus Cycle: Single
(RCD [1:0] = 01, TPC [2:0] = 001, TRWL [2:0] = 010)

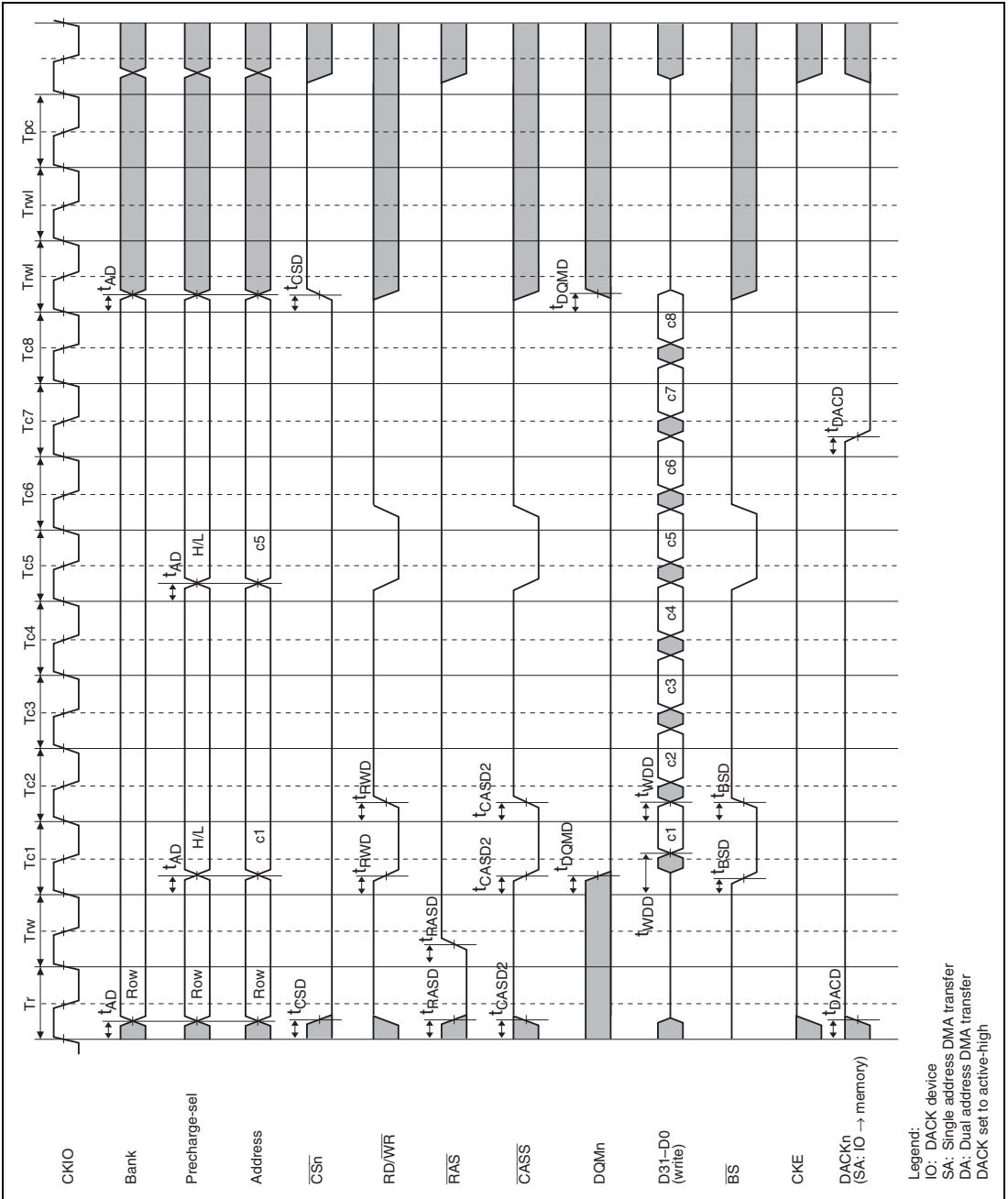


Figure 23.27 Synchronous DRAM Auto-Precharge Write Bus Cycle: Burst
 (RCD [1:0] = 01, TPC [2:0] = 001, TRWL [2:0] = 010)

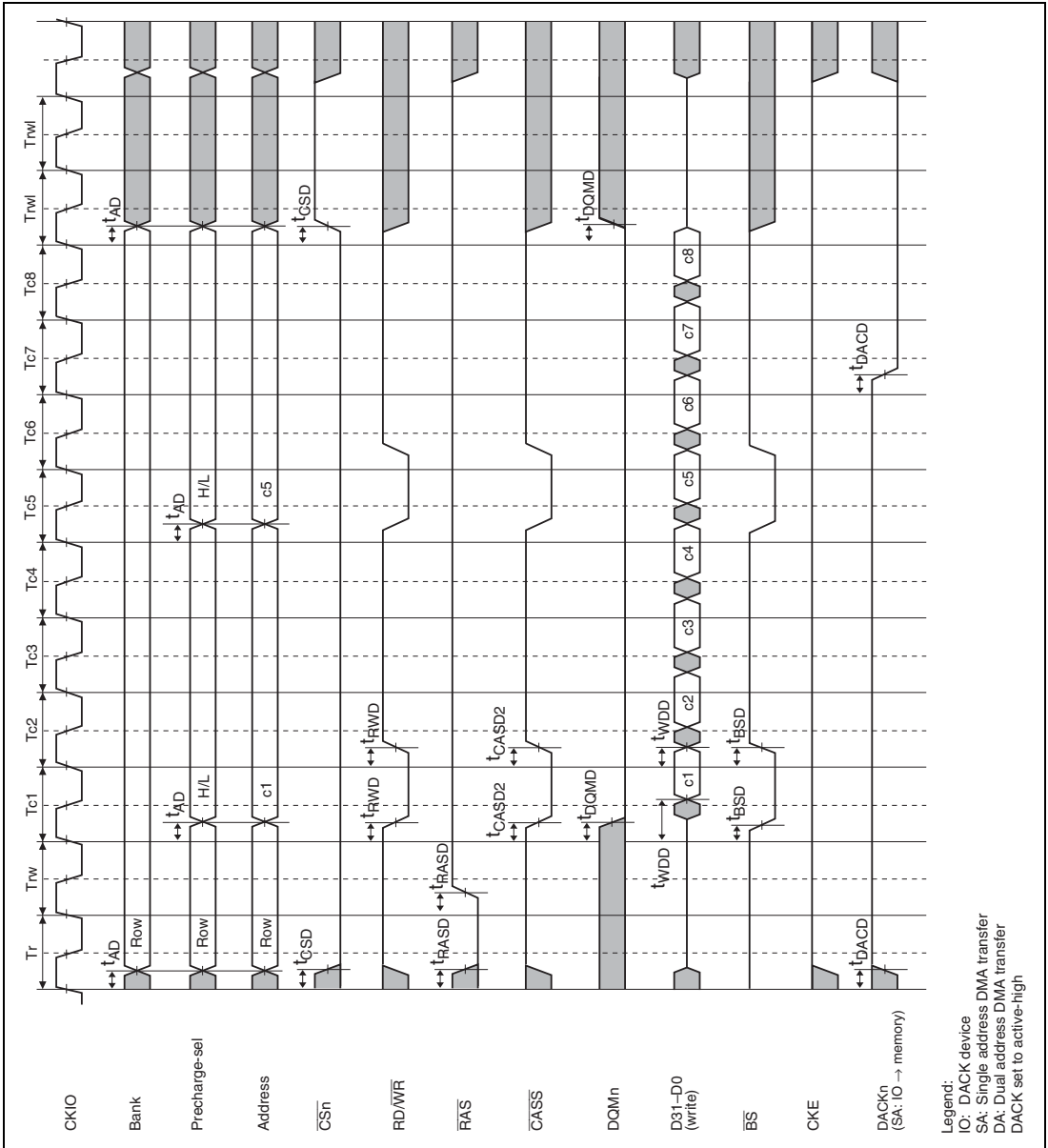


Figure 23.28 Synchronous DRAM Normal Write Bus Cycle: ACT + WRITE Commands, Burst (RASD = 1, RCD [1:0] = 01, TRWL [2:0] = 010)

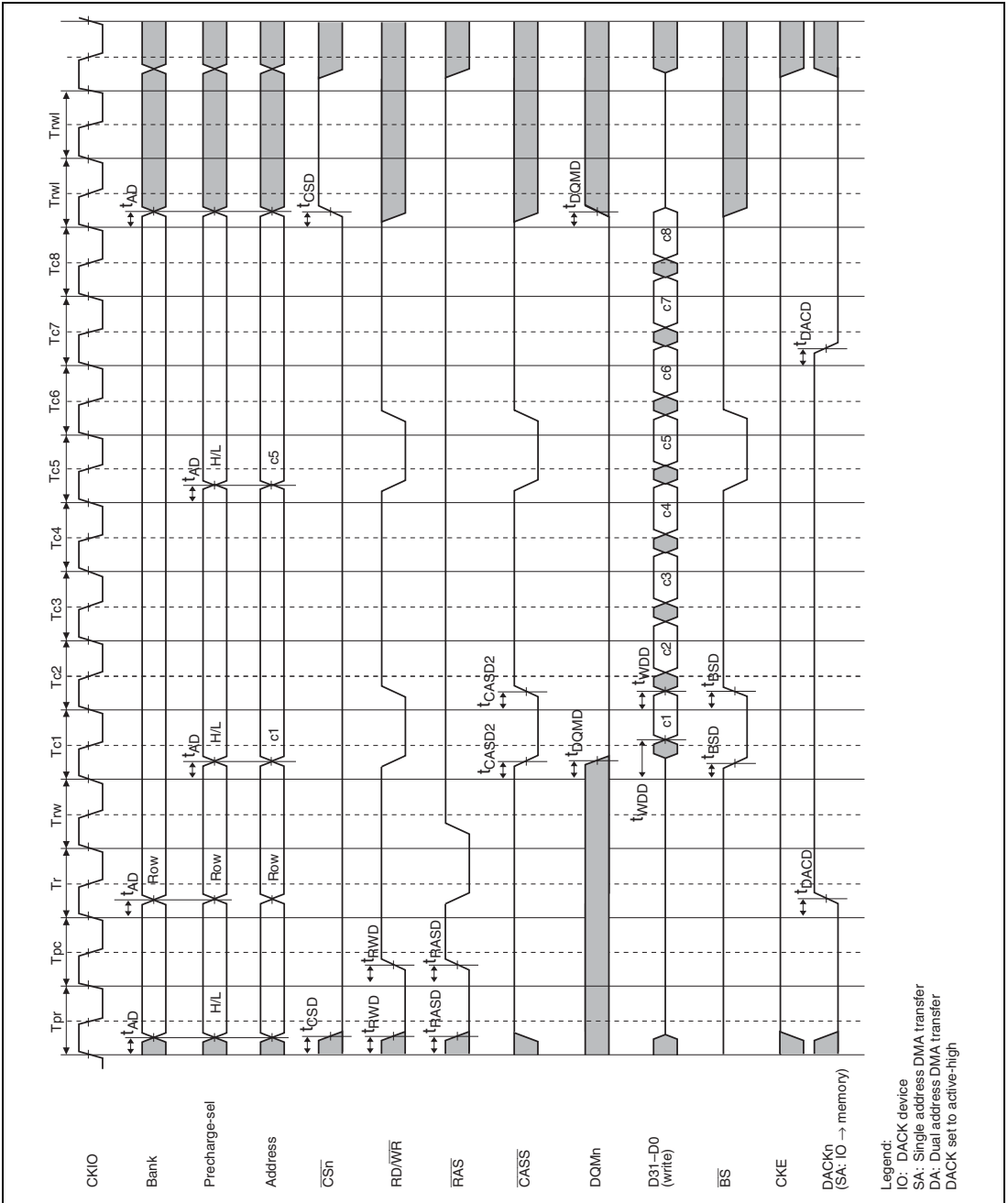


Figure 23.29 Synchronous DRAM Normal Write Bus Cycle: PRE + ACT + WRITE Commands, Burst (RASD = 1, RCD [1:0] = 01, TPC [2:0] = 001, TRWL [2:0] = 010)

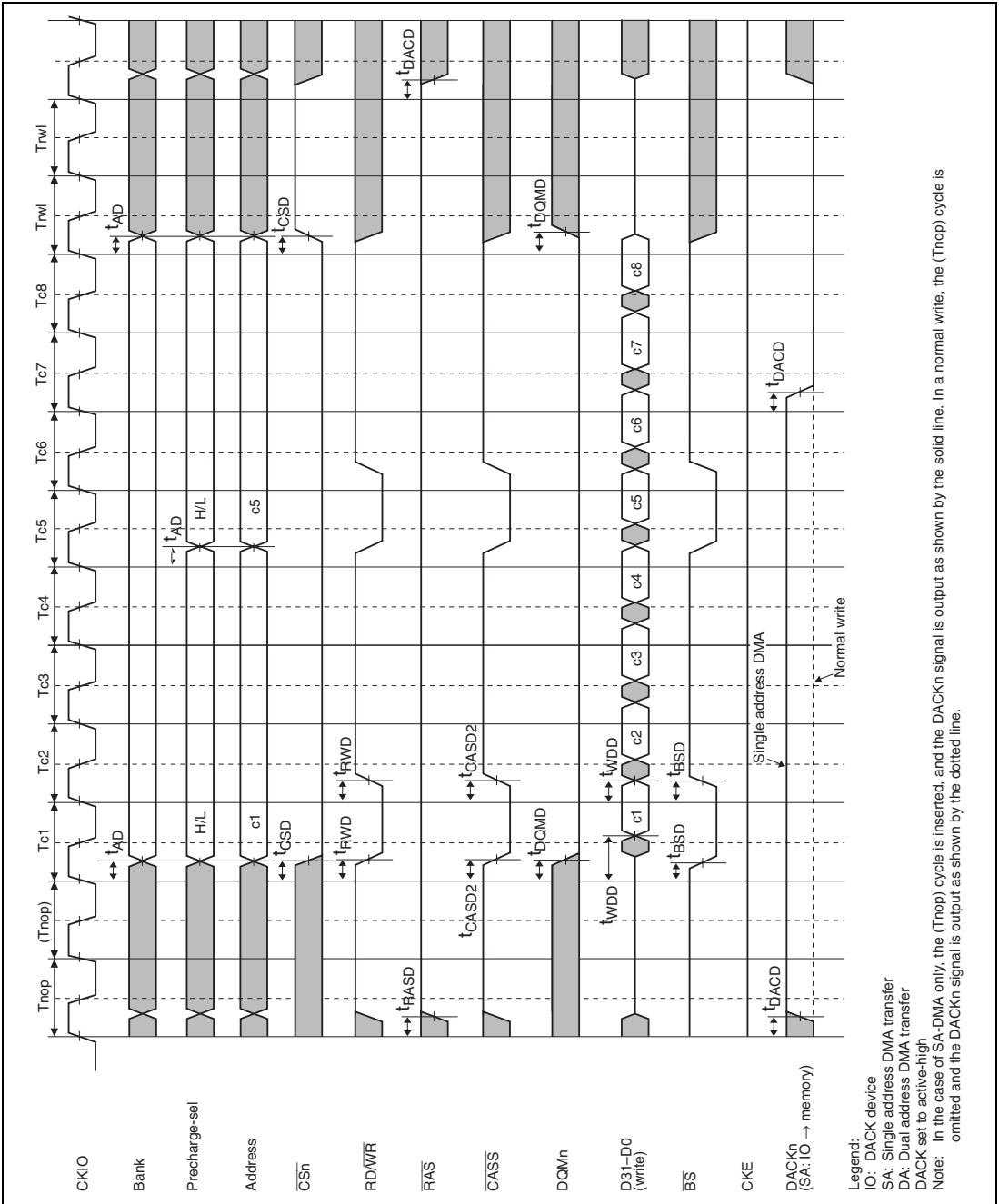
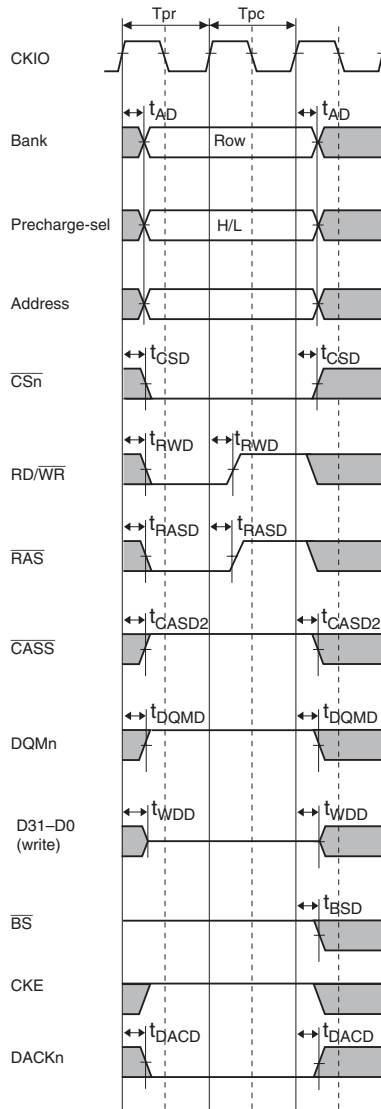


Figure 23.30 Synchronous DRAM Normal Write Bus Cycle: WRITE Command, Burst (RASD = 1, TRWL [2:0] = 010)



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.31 Synchronous DRAM Bus Cycle: Precharge Command (TPC [2:0] = 001)

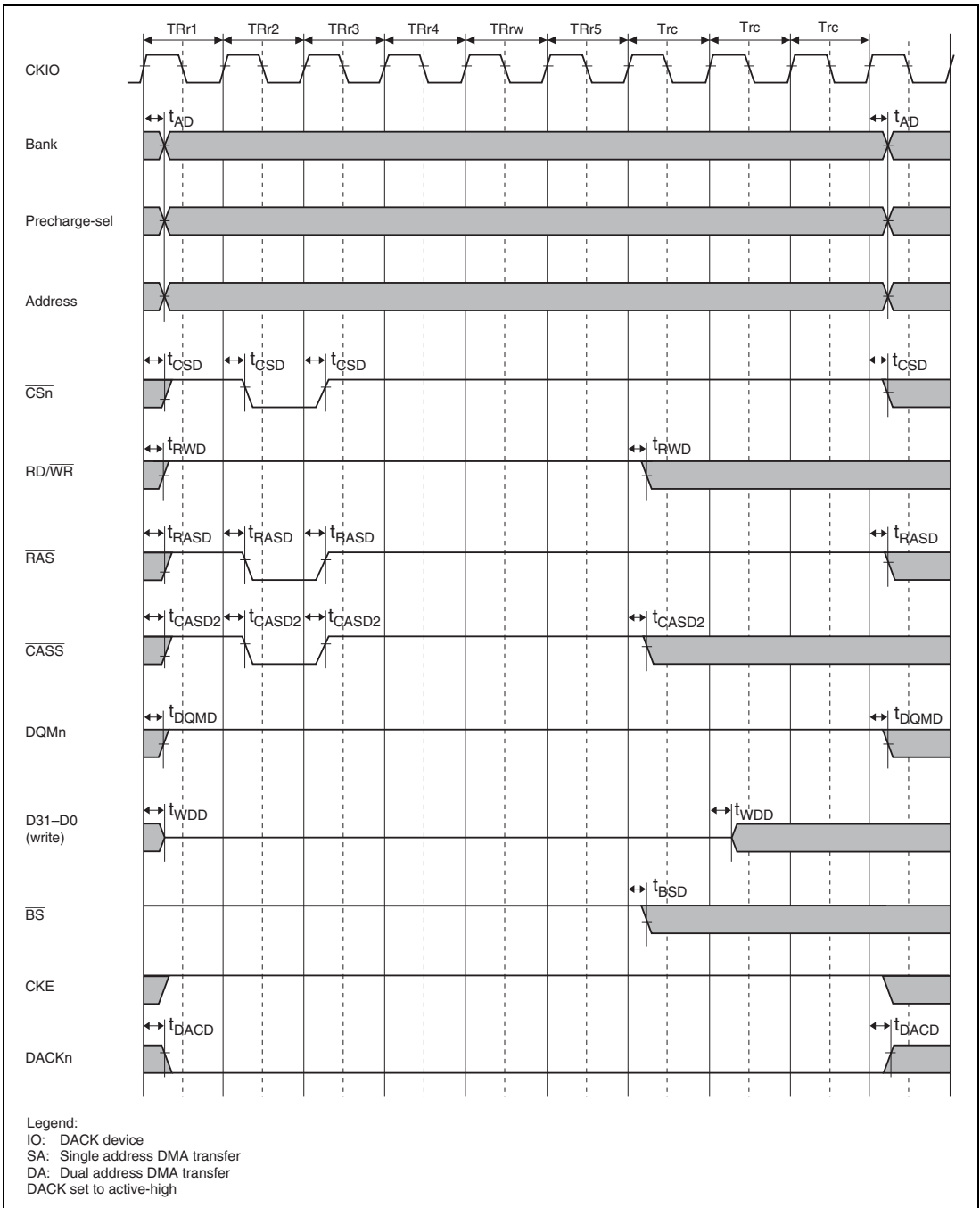


Figure 23.32 Synchronous DRAM Bus Cycle: Auto-Refresh ($TRAS = 1$, $TRC [2:0] = 001$)

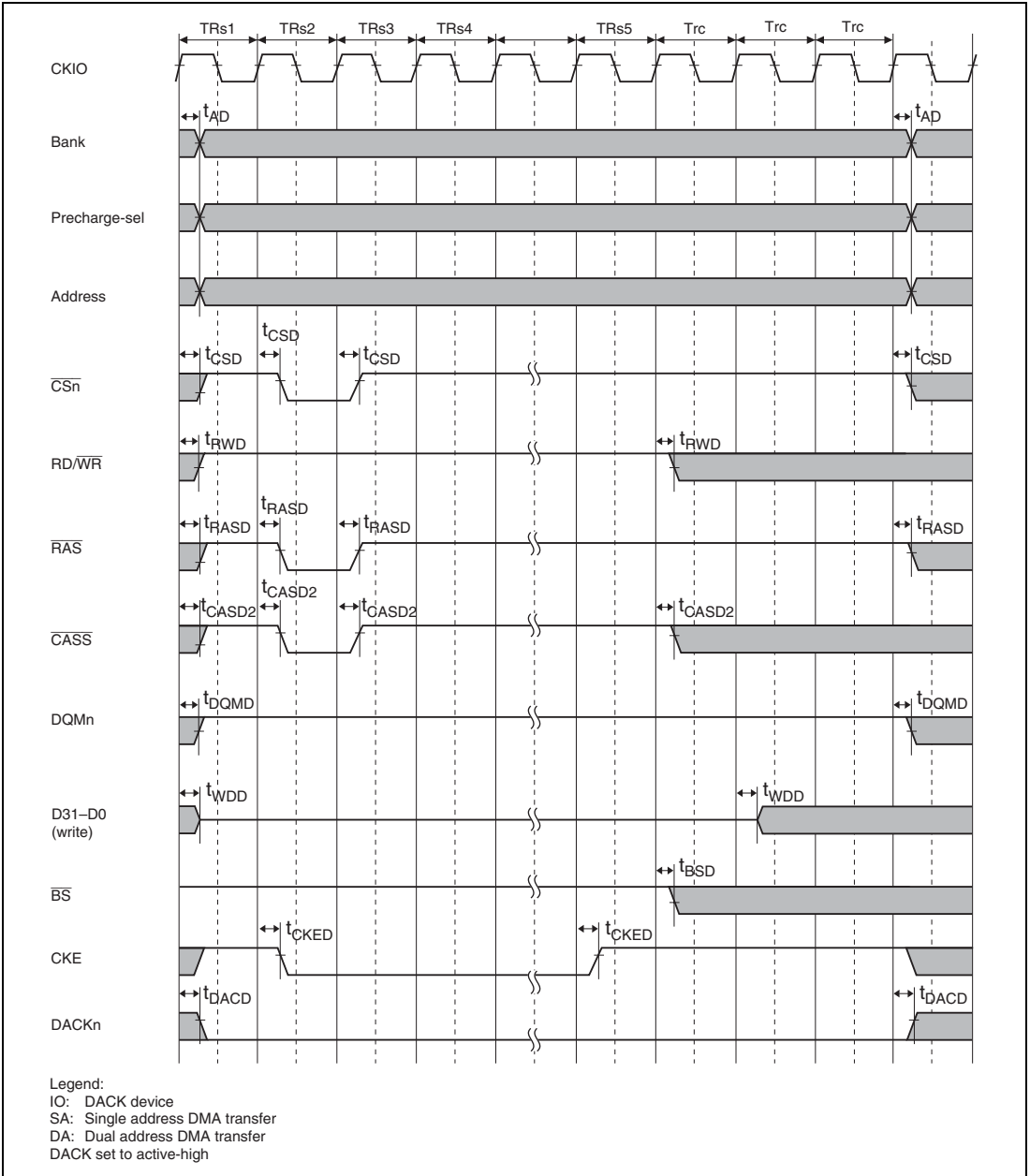


Figure 23.33 Synchronous DRAM Bus Cycle: Self-Refresh (TRC [2:0] = 001)

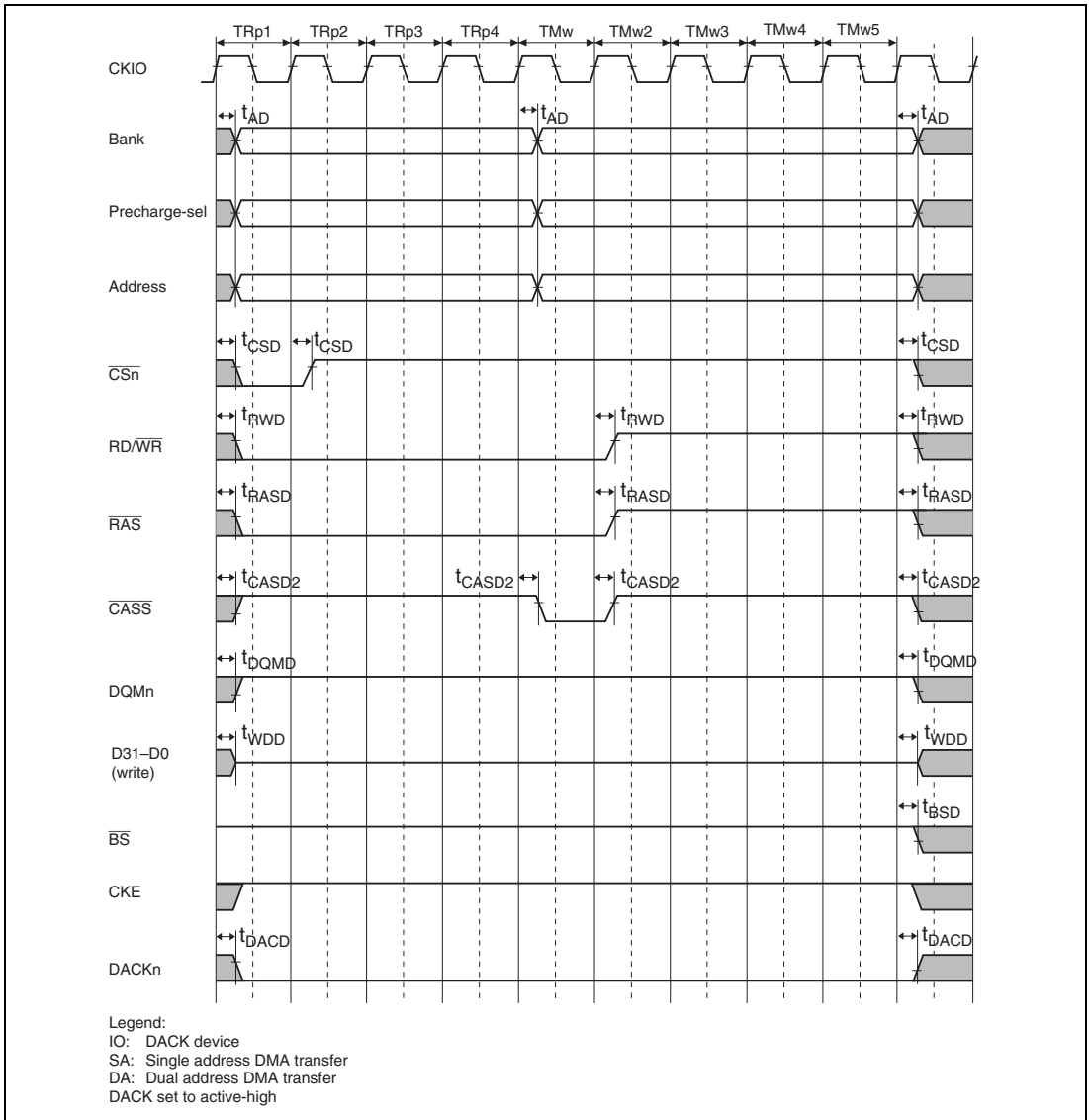


Figure 23.34 (a) Synchronous DRAM Bus Cycle: Mode Register Setting (PALL)

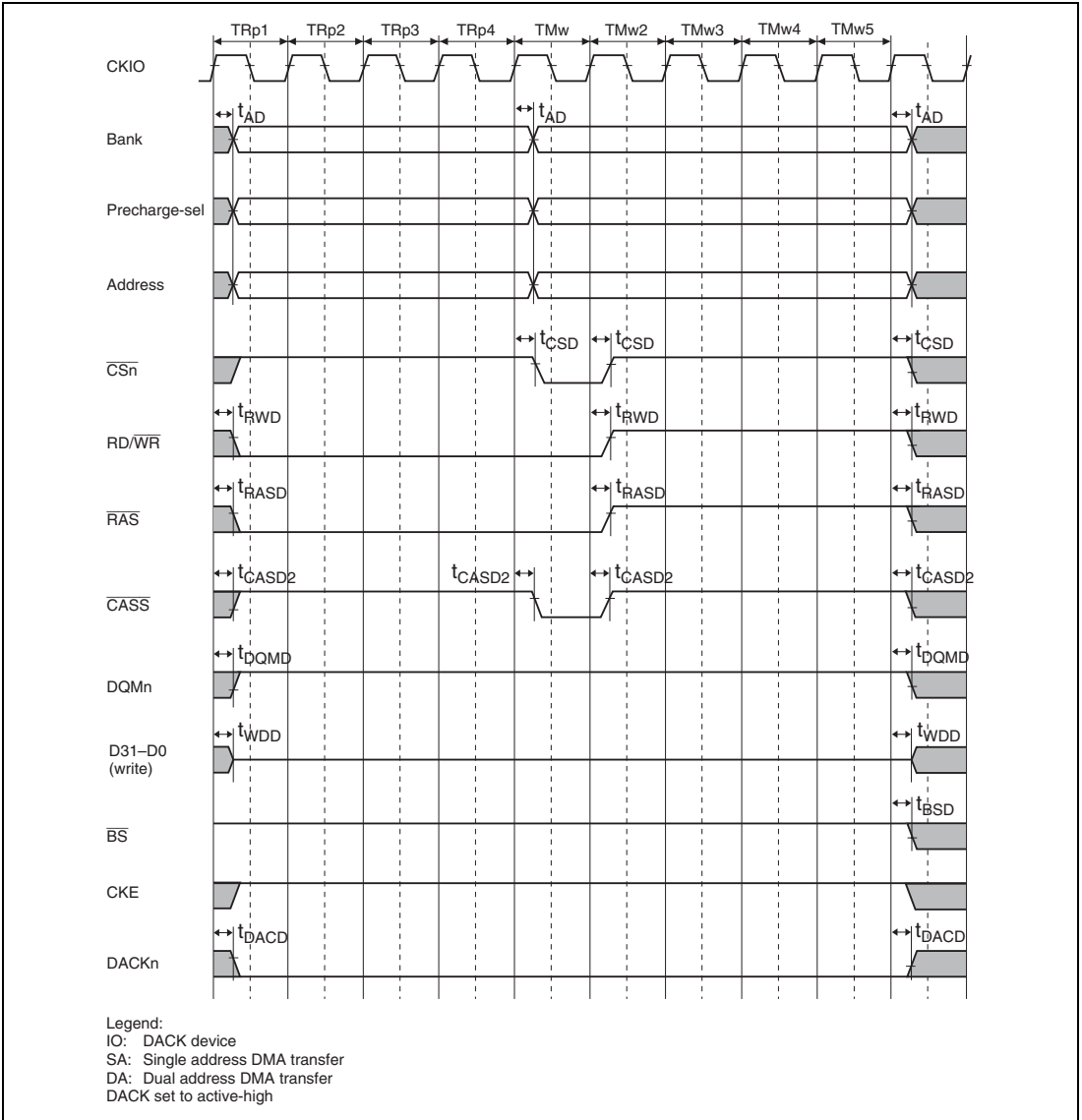


Figure 23.34 (b) Synchronous DRAM Bus Cycle: Mode Register Setting (SET)

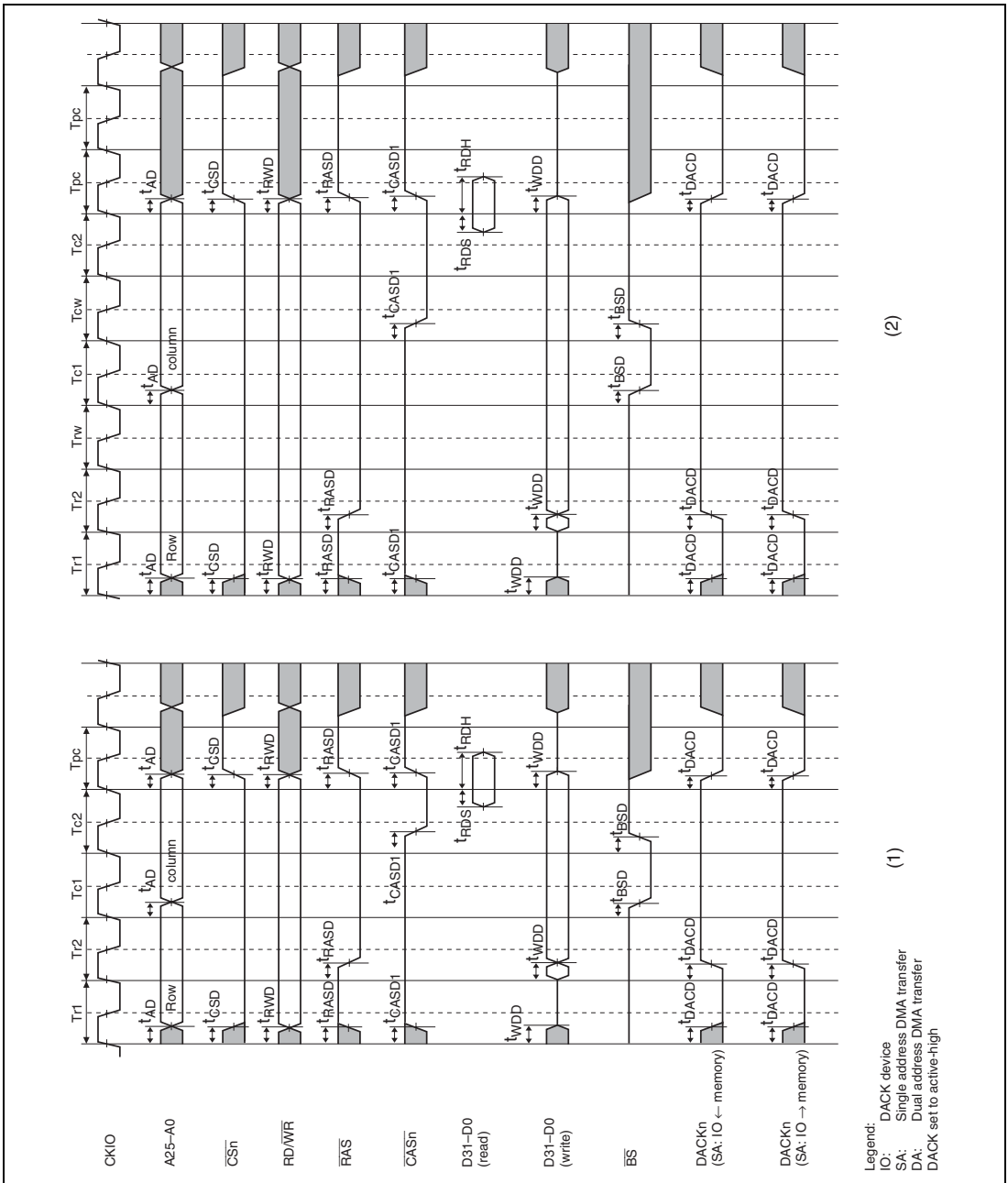


Figure 23.35 DRAM Bus Cycles

(1) RCD [1:0] = 00, AnW [2:0] = 000, TPC [2:0] = 001

(2) RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 010

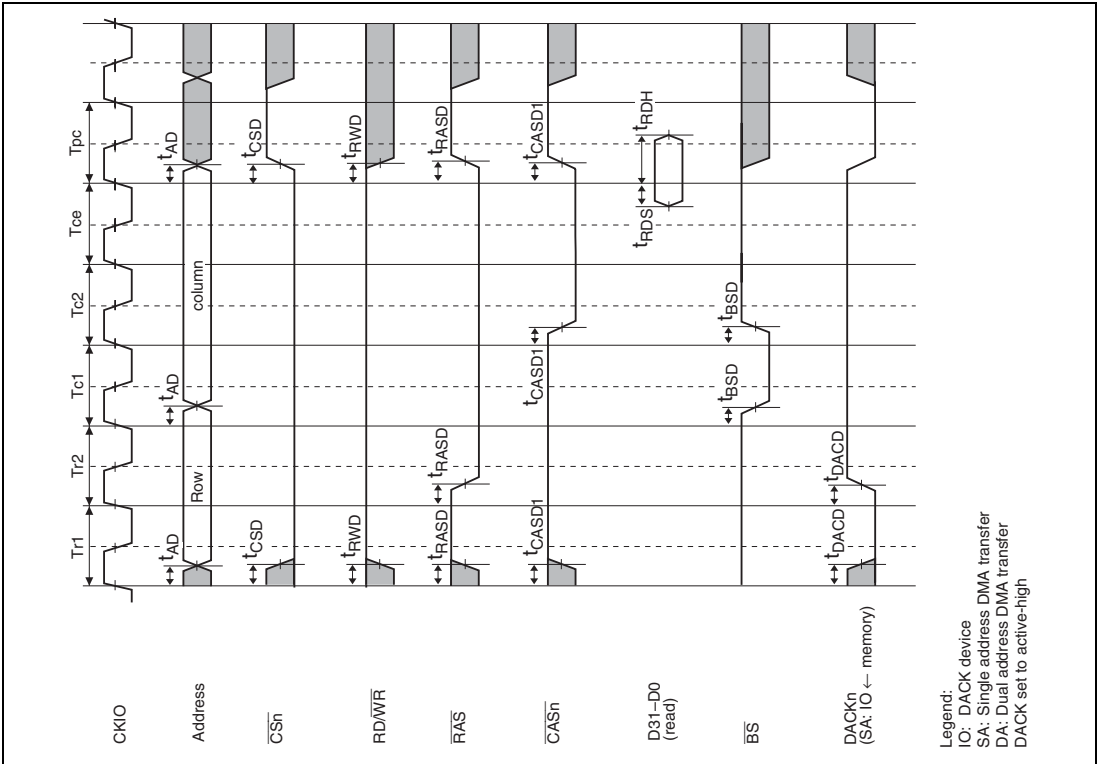


Figure 23.36 DRAM Bus Cycle
 (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000, TRC [2:0] = 001)

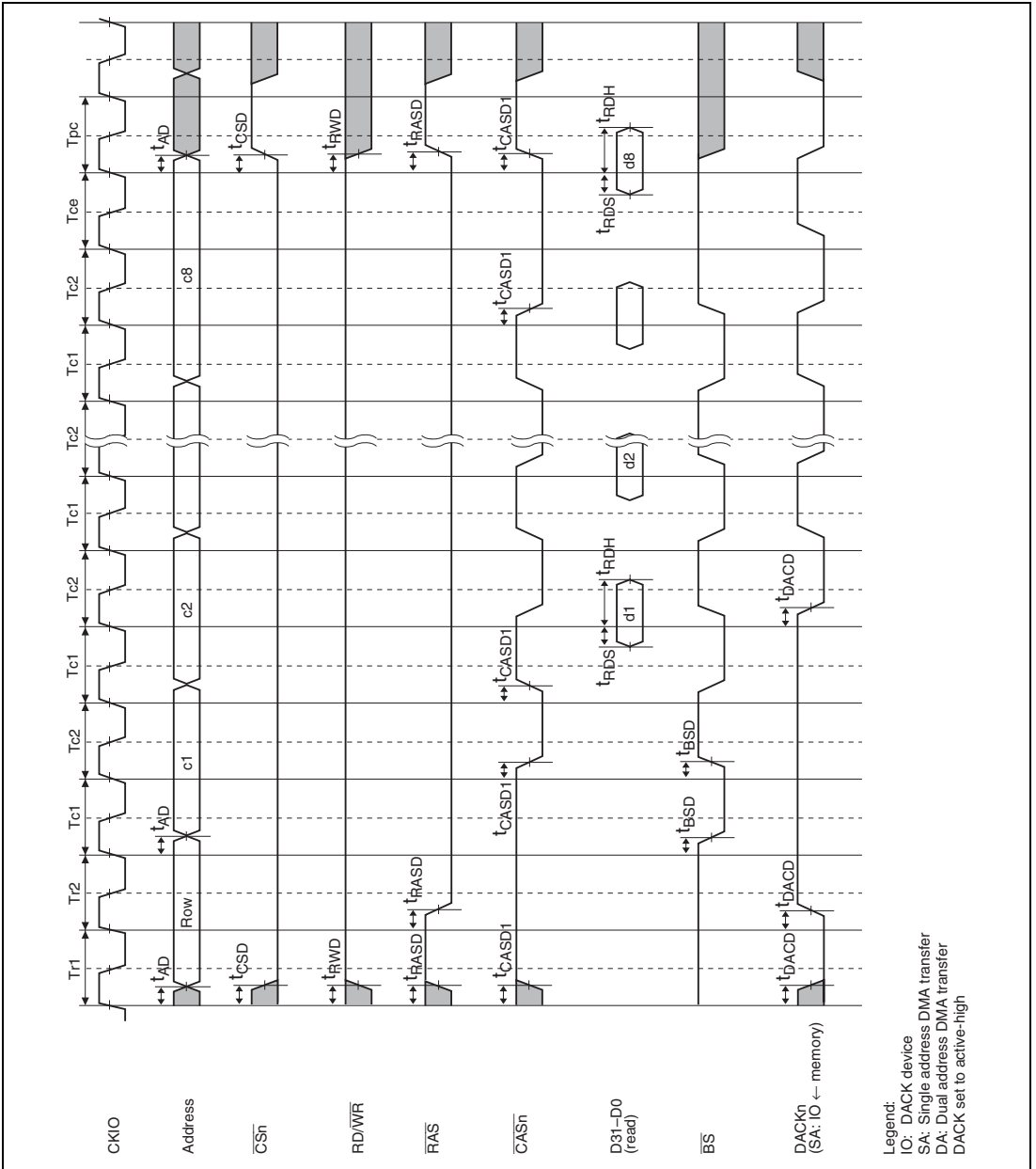
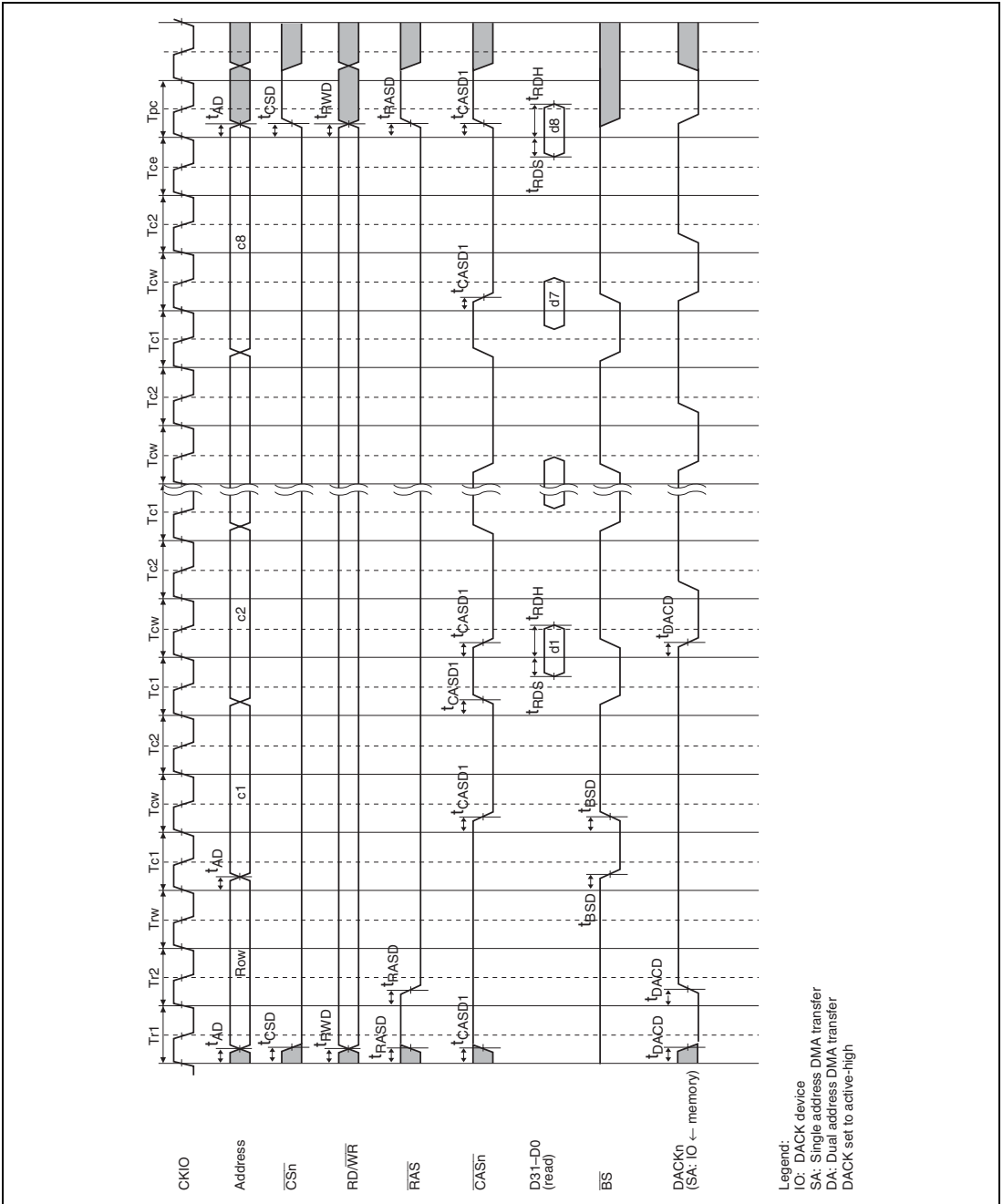
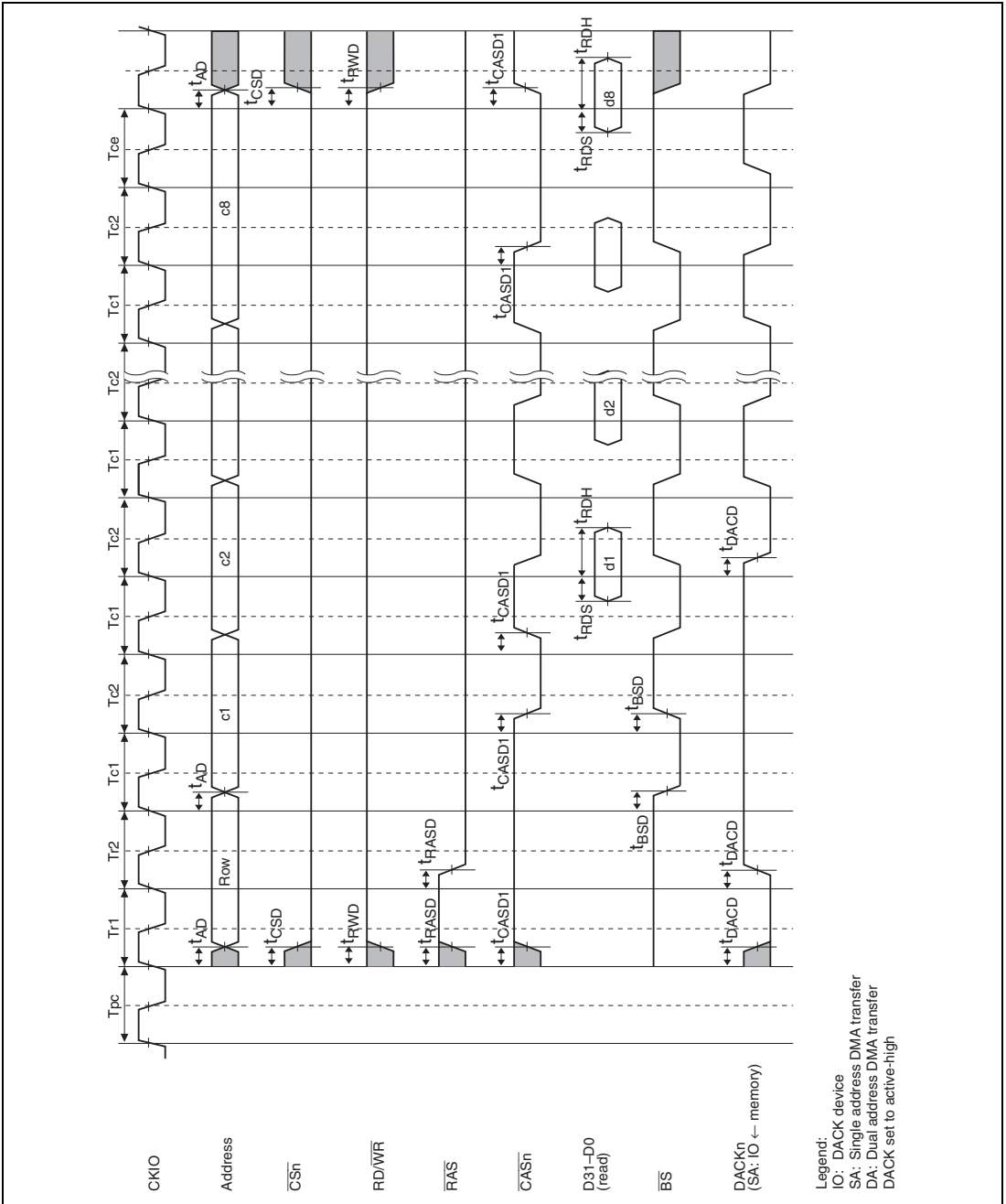


Figure 23.37 DRAM Bus Cycle
(EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000, TPC [2:0] = 001)



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.38 DRAM Burst Bus Cycle
 (EDO Mode, RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 001)



**Figure 23.40 DRAM Burst Bus Cycle: RAS Down Mode State
 (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000)**

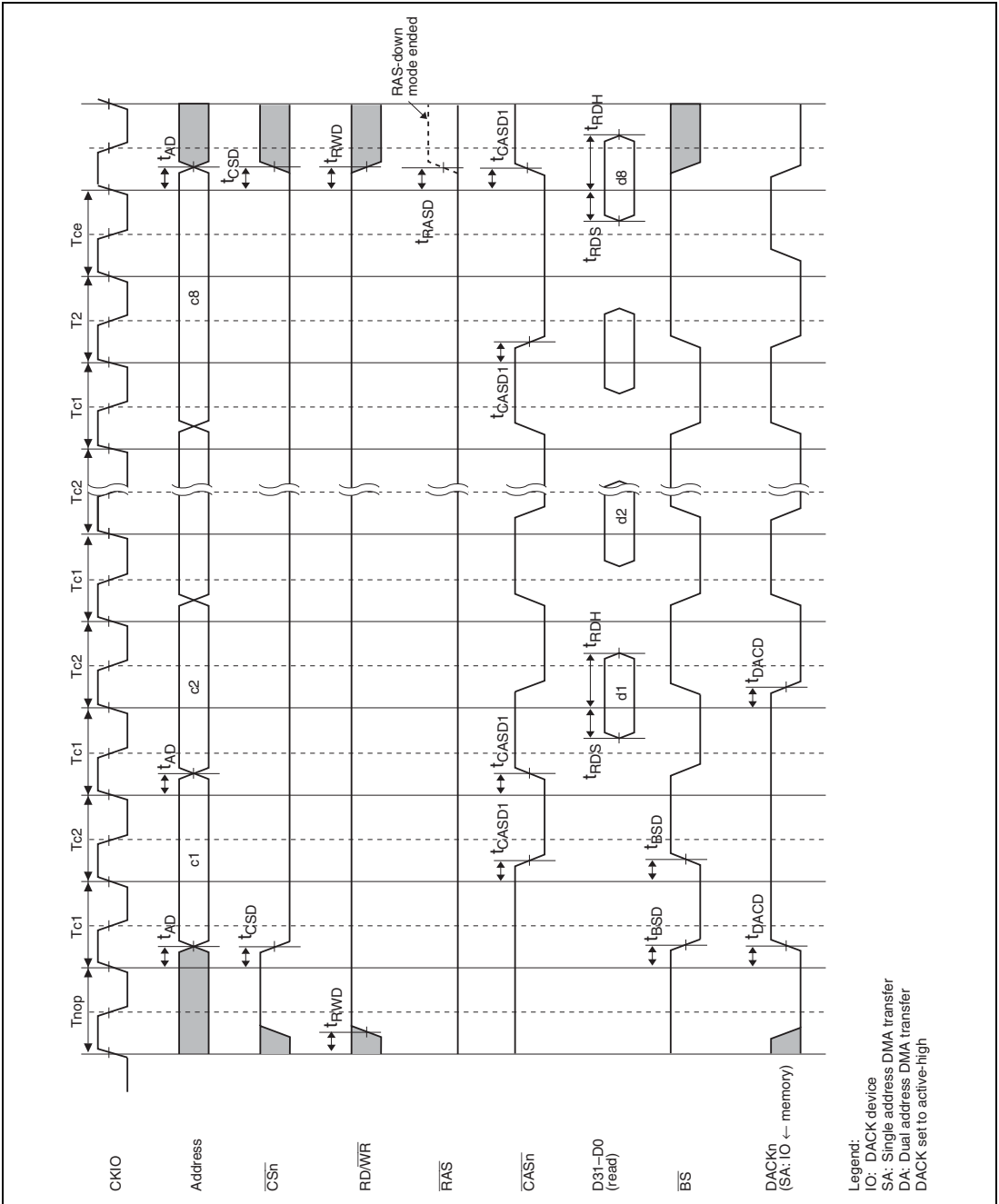
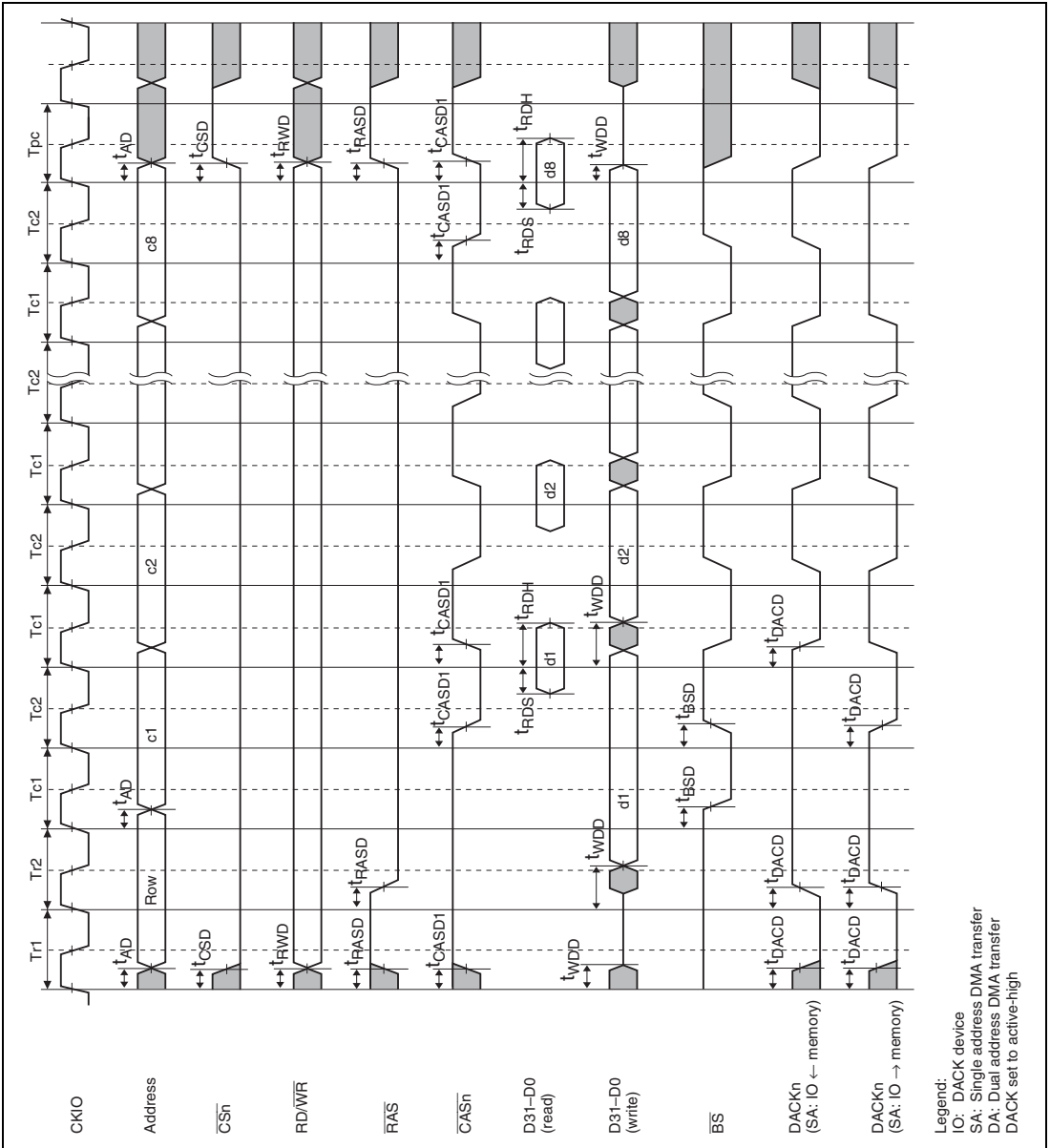


Figure 23.41 DRAM Burst Bus Cycle: RAS Down Mode Continuation (EDO Mode, RCD [1:0] = 00, AnW [2:0] = 000)



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.42 DRAM Burst Bus Cycle
(Fast Page Mode, RCD [1:0] = 00, AnW [2:0] = 000, TPC [2:0] = 001)

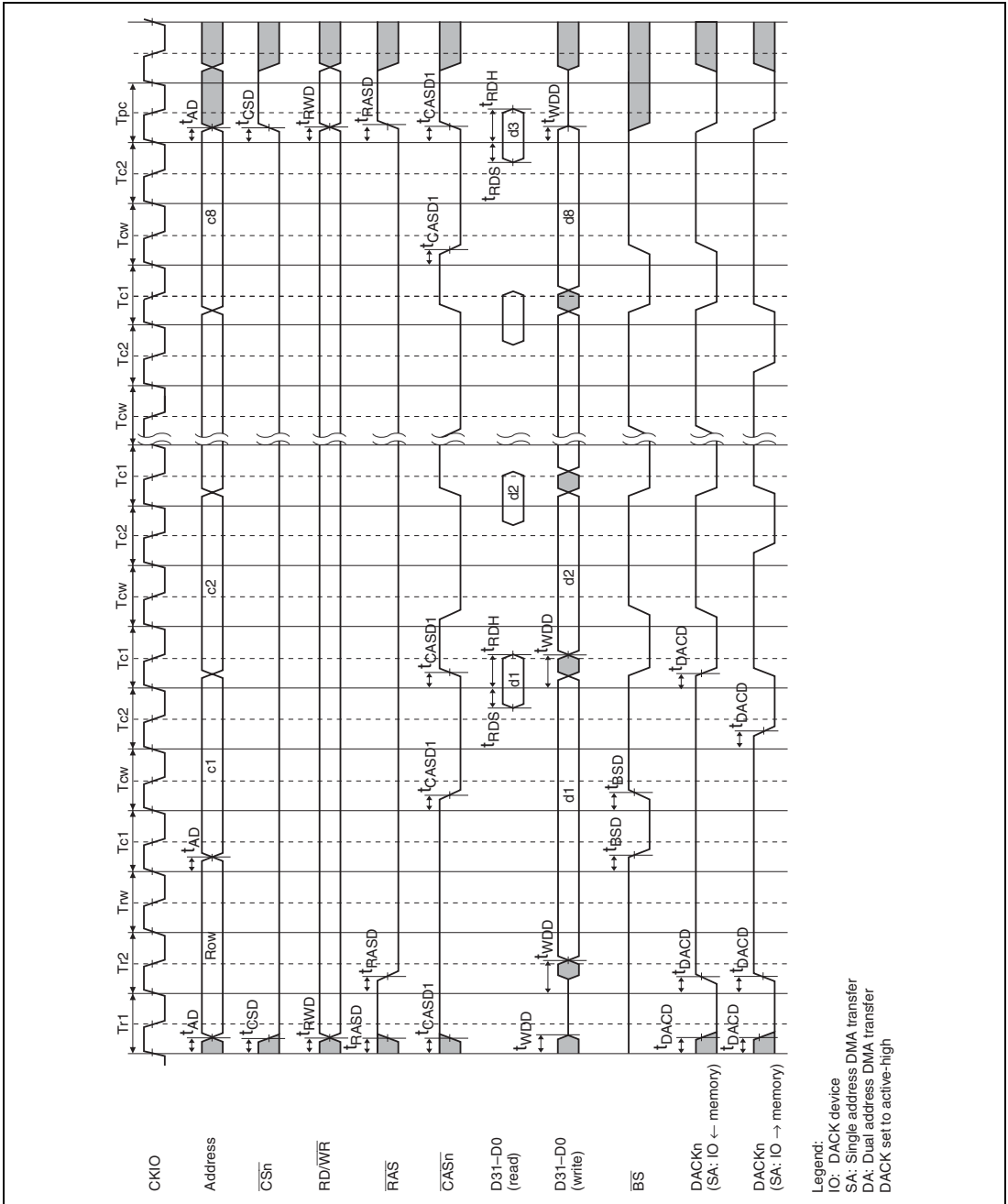
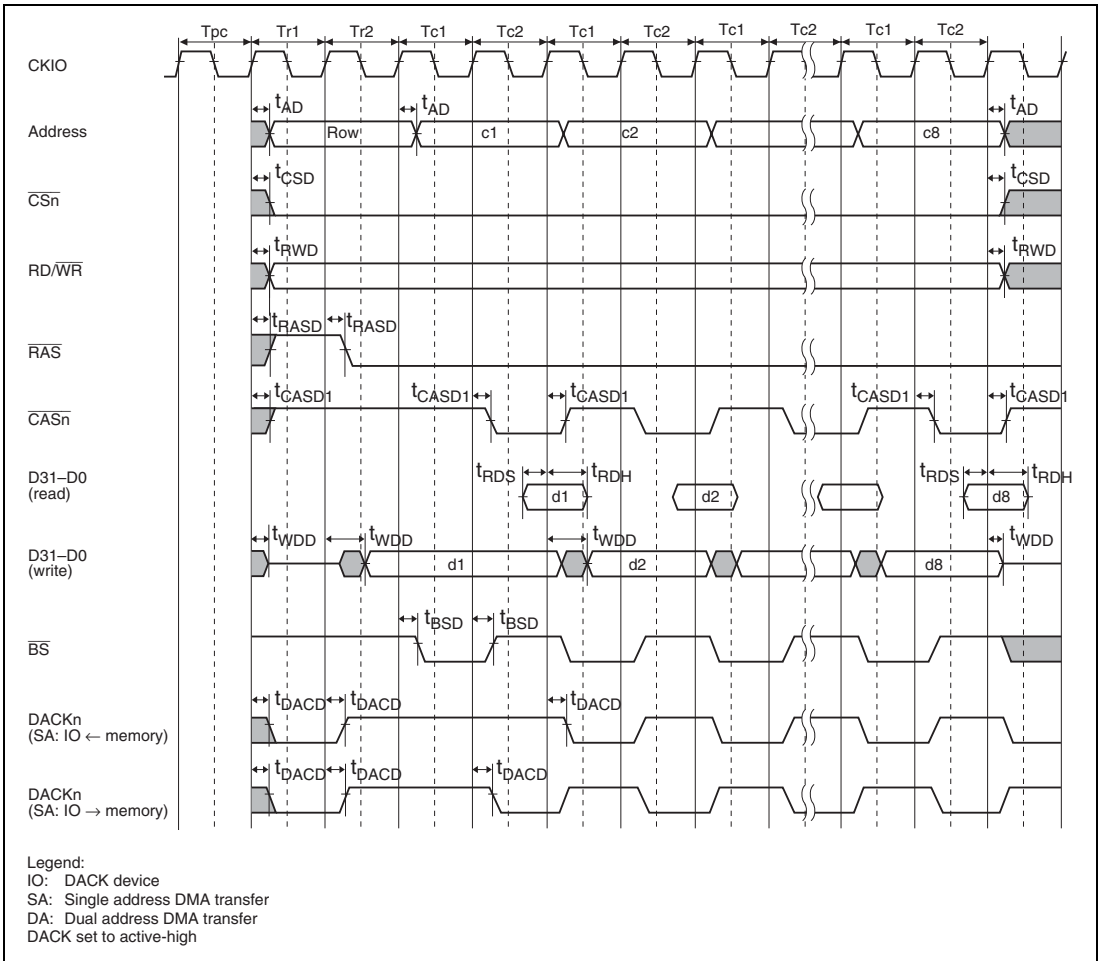


Figure 23.43 DRAM Burst Bus Cycle

(Fast Page Mode, RCD [1:0] = 01, AnW [2:0] = 001, TPC [2:0] = 001)



**Figure 23.45 DRAM Burst Bus Cycle: RAS Down Mode State
(Fast Page Mode, RCD [1:0] = 00, AnW [2:0] = 000)**

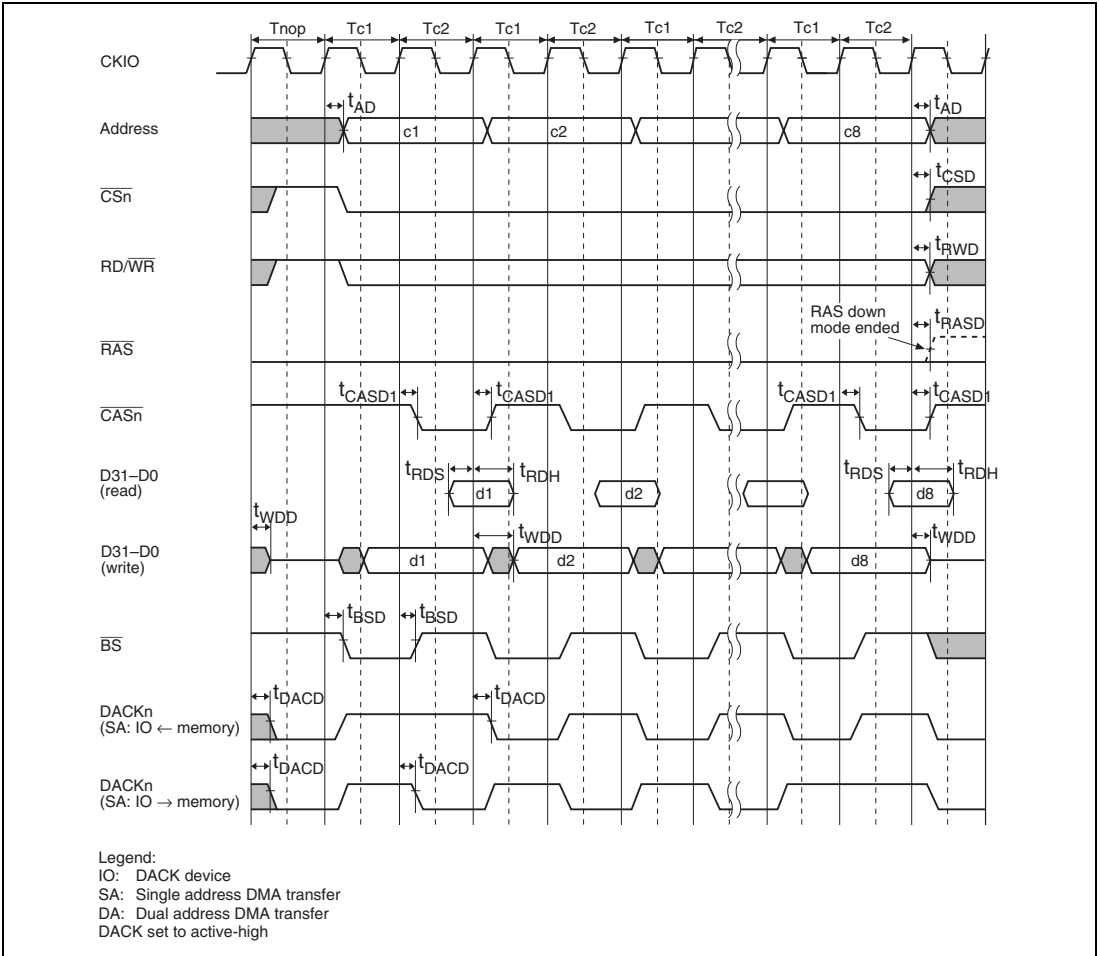


Figure 23.46 DRAM Burst Bus Cycle: RAS Down Mode Continuation
 (Fast Page Mode, RCD [1:0] = 00, AnW [2:0] = 000)

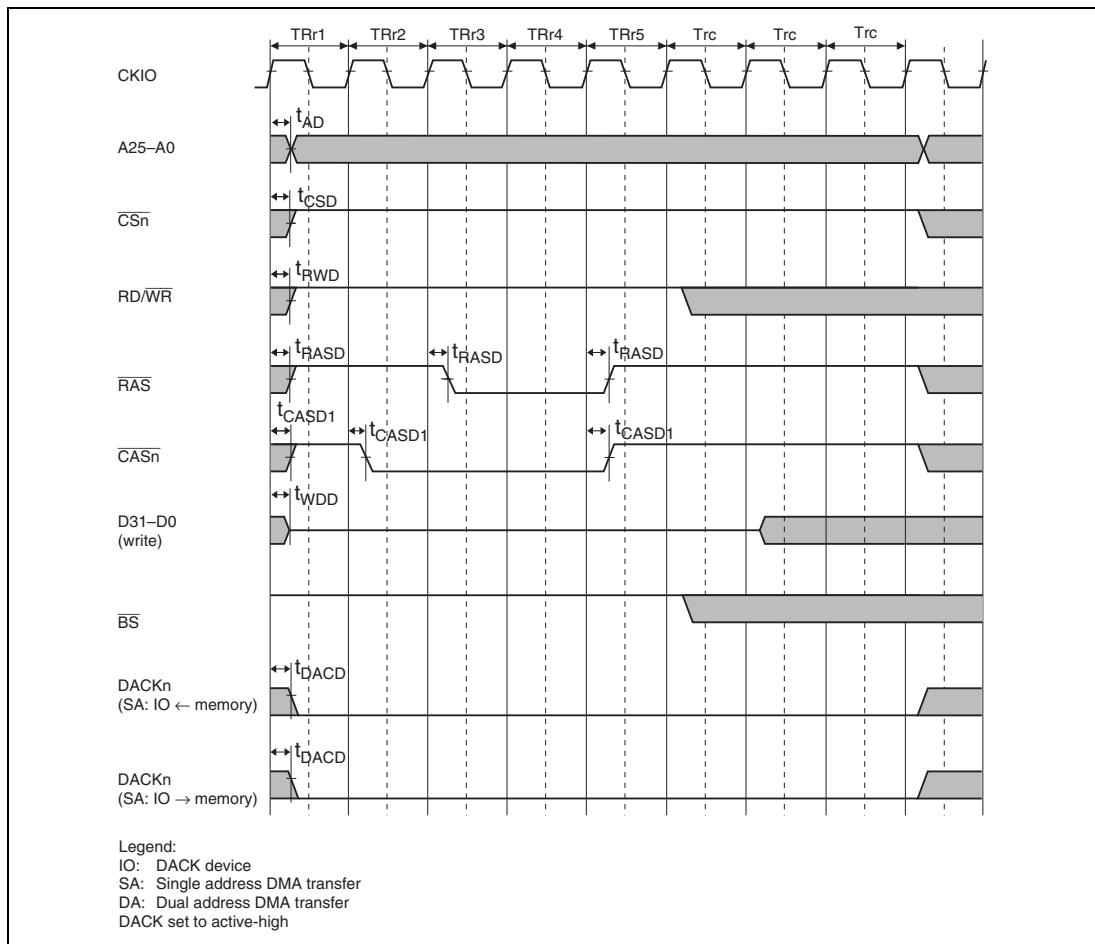
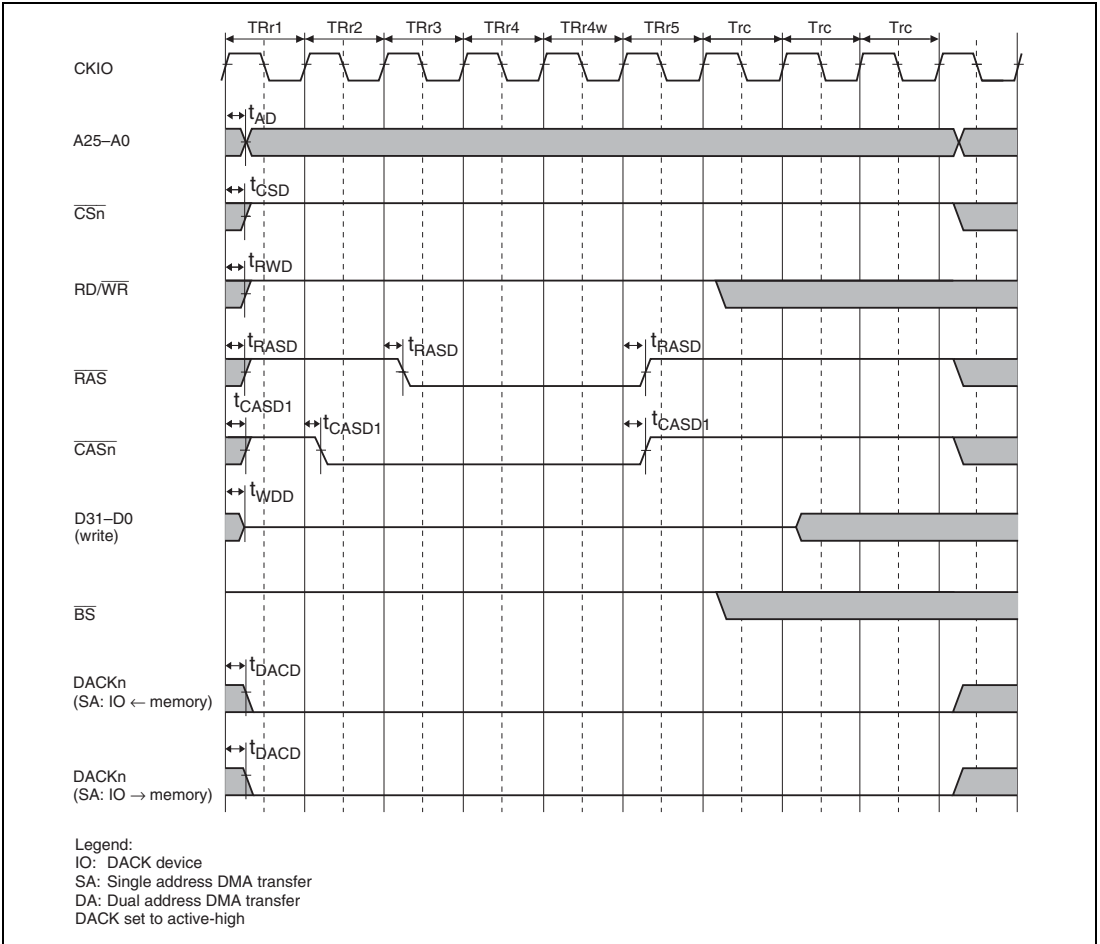


Figure 23.47 DRAM Bus Cycle: DRAM CAS-Before-RAS Refresh
($TRAS[2:0] = 000$, $TRC[2:0] = 001$)



**Figure 23.48 DRAM Bus Cycle: DRAM CAS-Before-RAS Refresh
 (TRAS [2:0] = 001, TRC [2:0] = 001)**

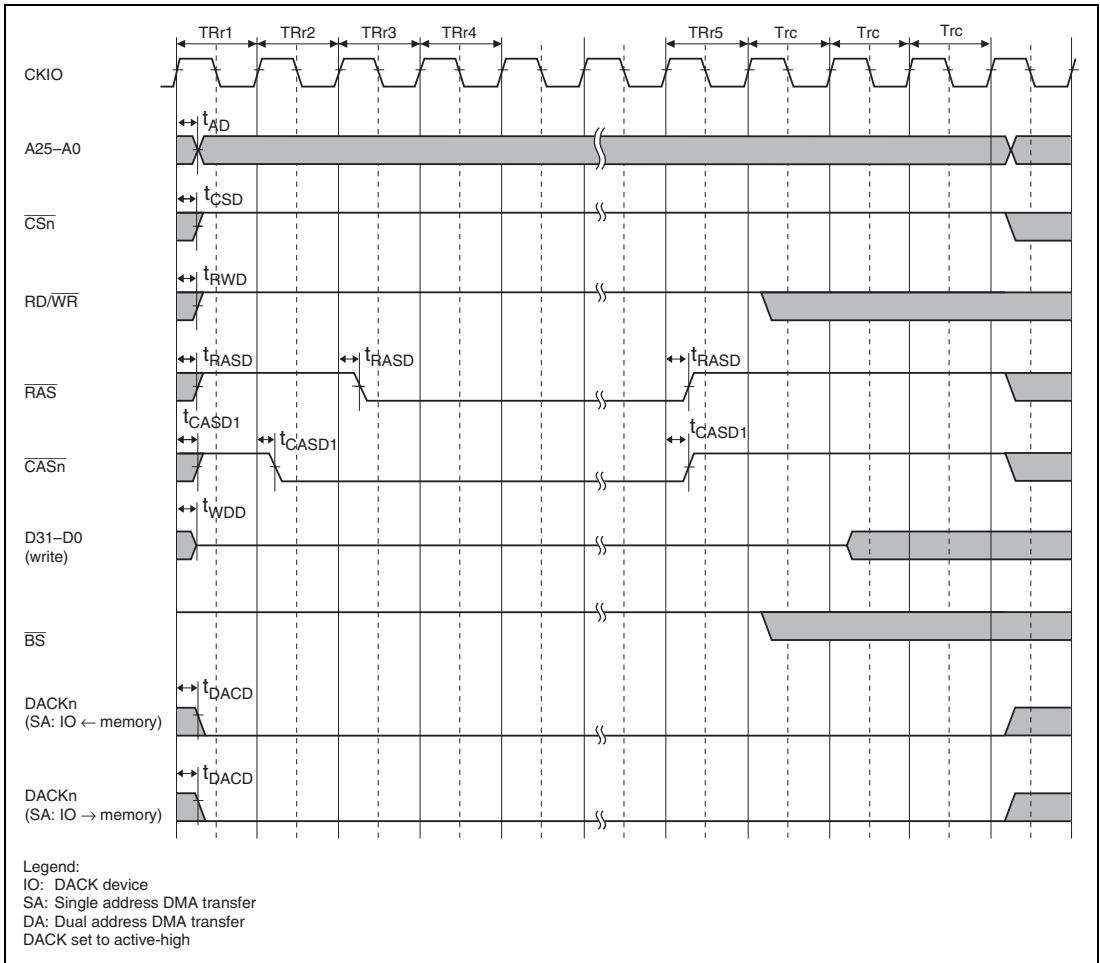


Figure 23.49 DRAM Bus Cycle: DRAM Self-Refresh (TRC [2:0] = 001)

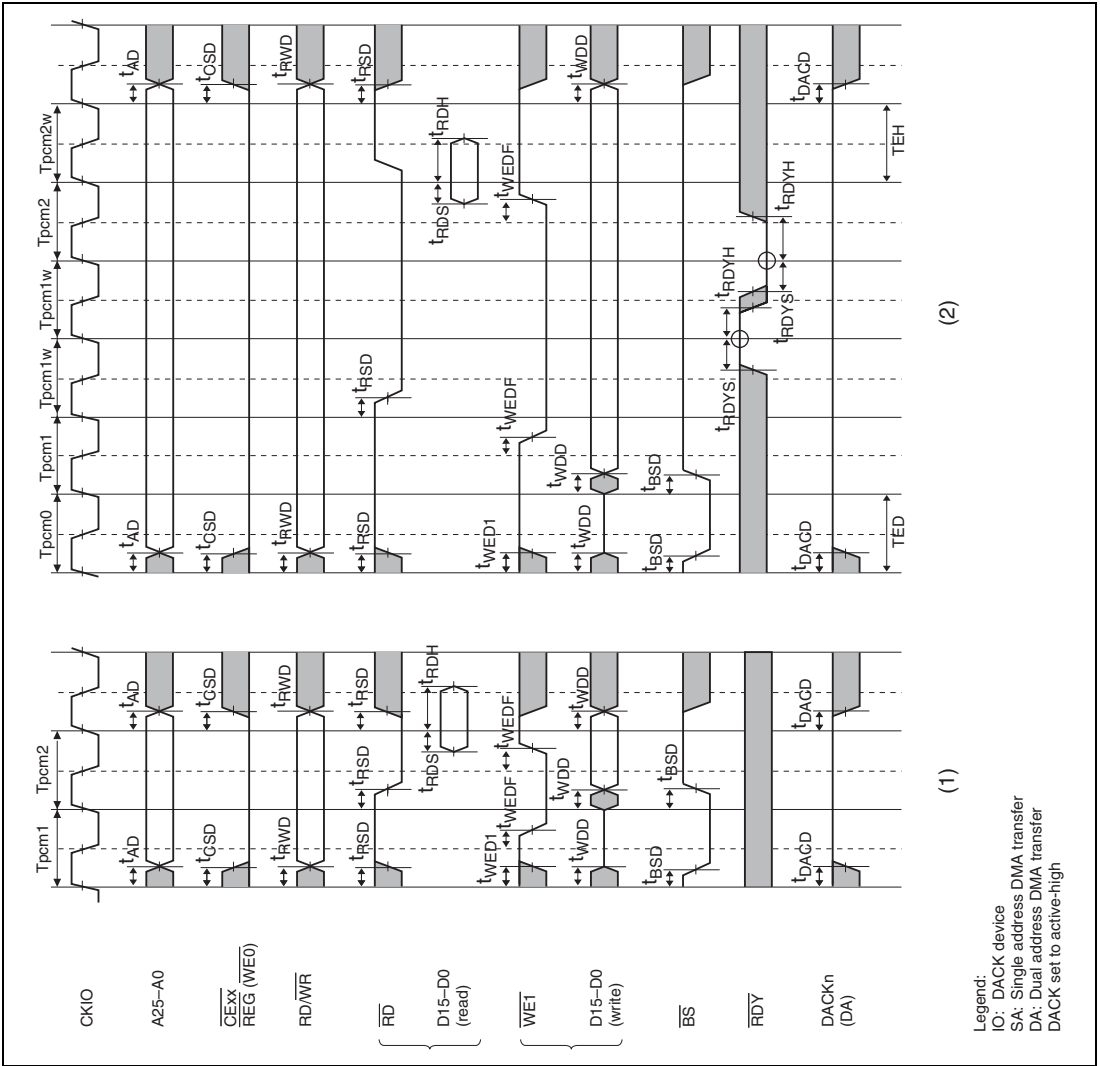


Figure 23.50 PCMCIA Memory Bus Cycle

(1) TED [2:0] = 000, TEH [2:0] = 000, No Wait

(2) TED [2:0] = 001, TEH [2:0] = 001, One Internal Wait + One External Wait

Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

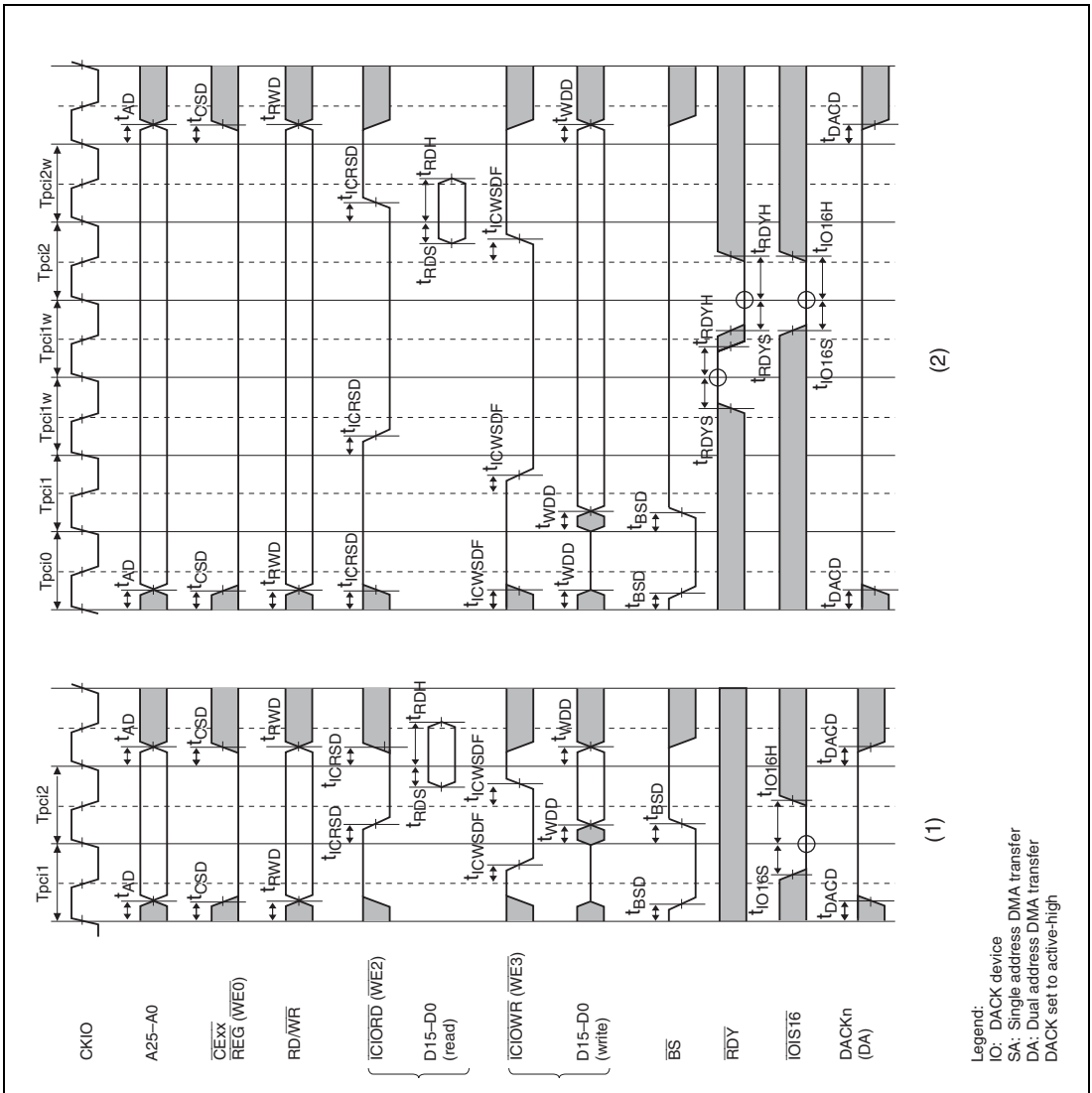


Figure 23.51 PCMCIA I/O Bus Cycle

(1) TED [2:0] = 000, TEH [2:0] = 000, No Wait

(2) TED [2:0] = 001, TEH [2:0] = 001, One Internal Wait + One External Wait

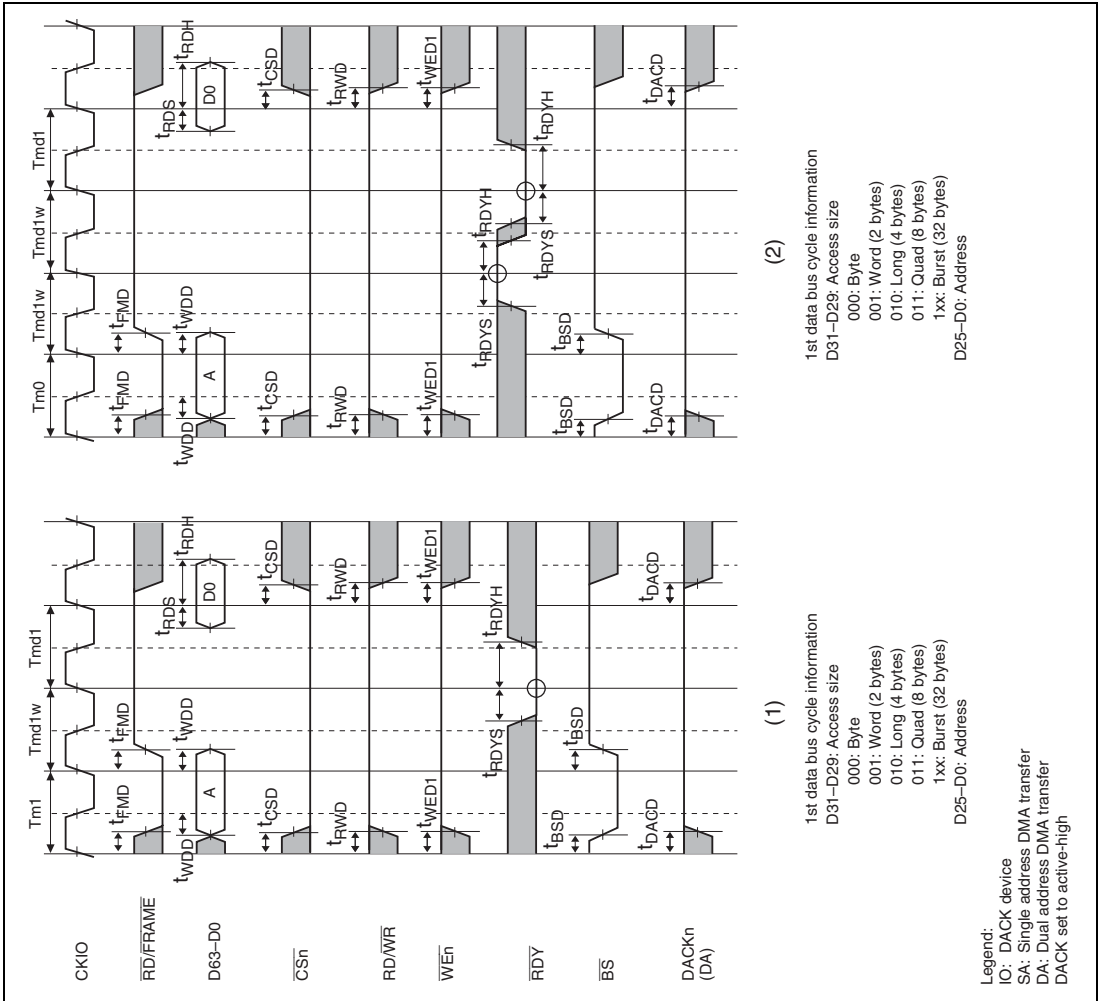


Figure 23.53 MPX Basic Bus Cycle: Read
(1) 1st Data (One Internal Wait)
(2) 1st Data (One Internal Wait + One External Wait)

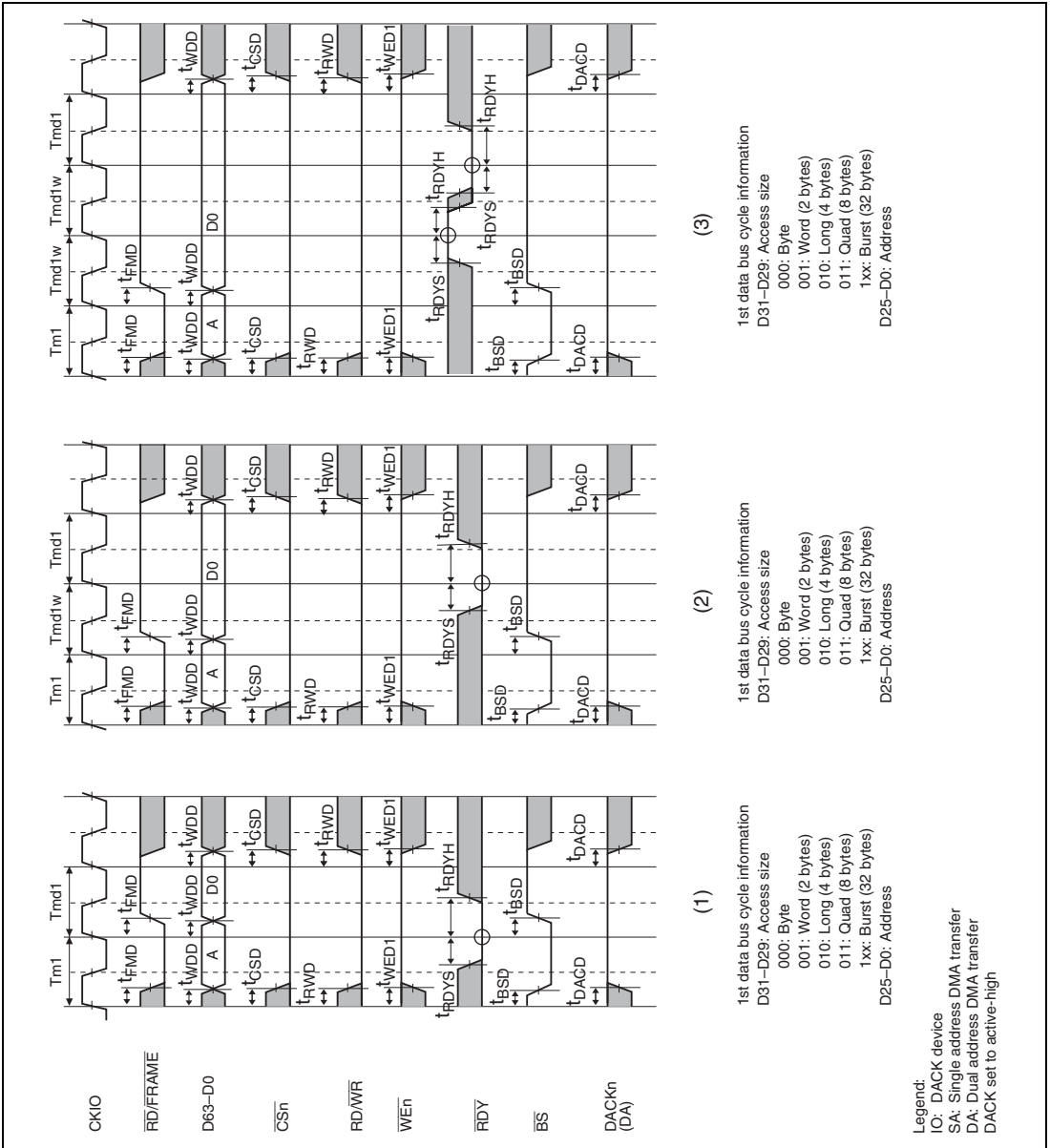


Figure 23.54 MPX Basic Bus Cycle: Write

(1) 1st Data (No Wait)

(2) 1st Data (One Internal Wait)

(3) 1st Data (One Internal Wait + One External Wait)

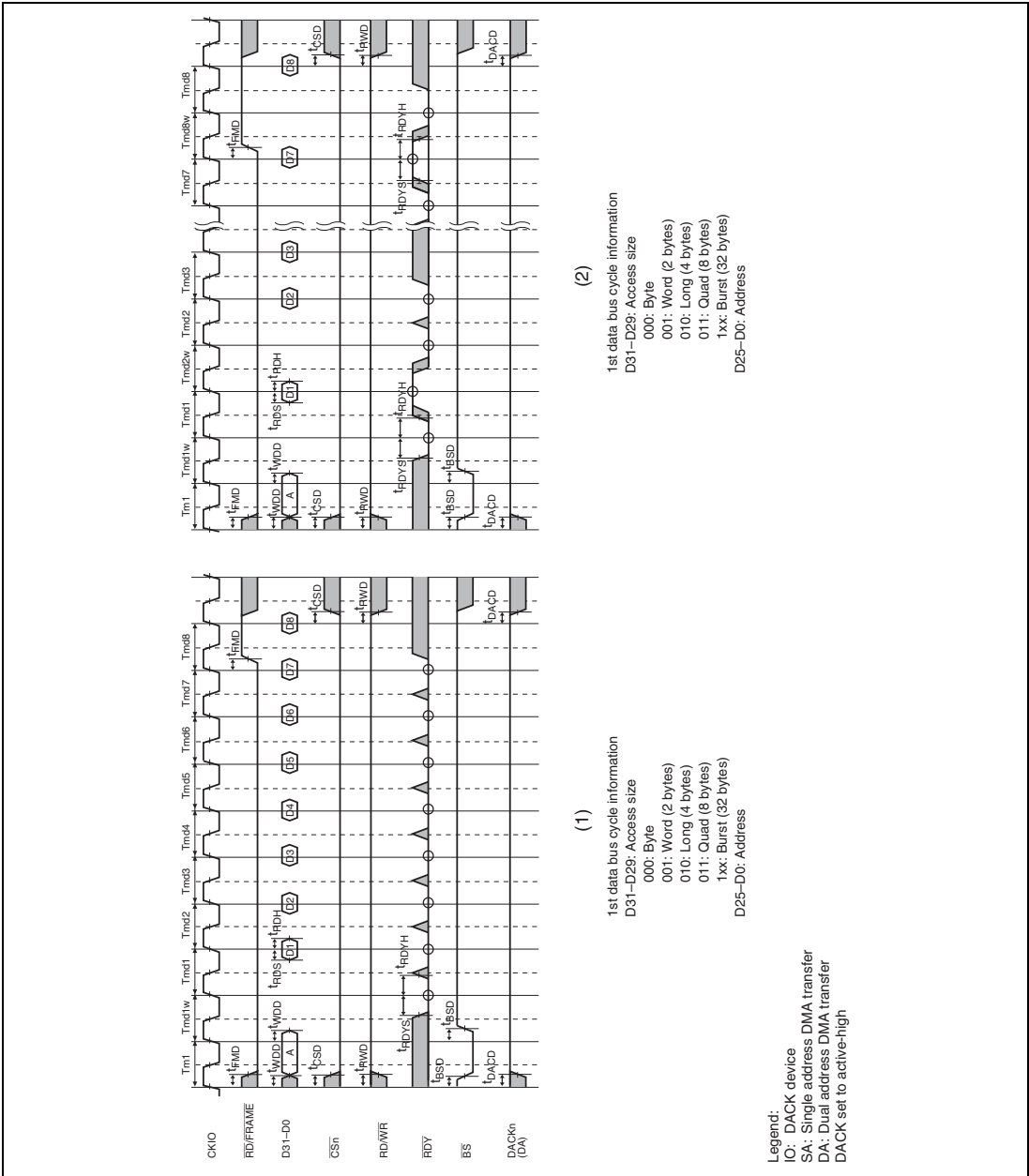


Figure 23.55 MPX Bus Cycle: Burst Read

- (1) 1st Data (One Internal Wait), 2nd to 8th Data (No Internal Wait)**
- (2) 1st Data (No Internal Wait), 2nd to 8th Data (No Internal Wait + External Wait Control)**

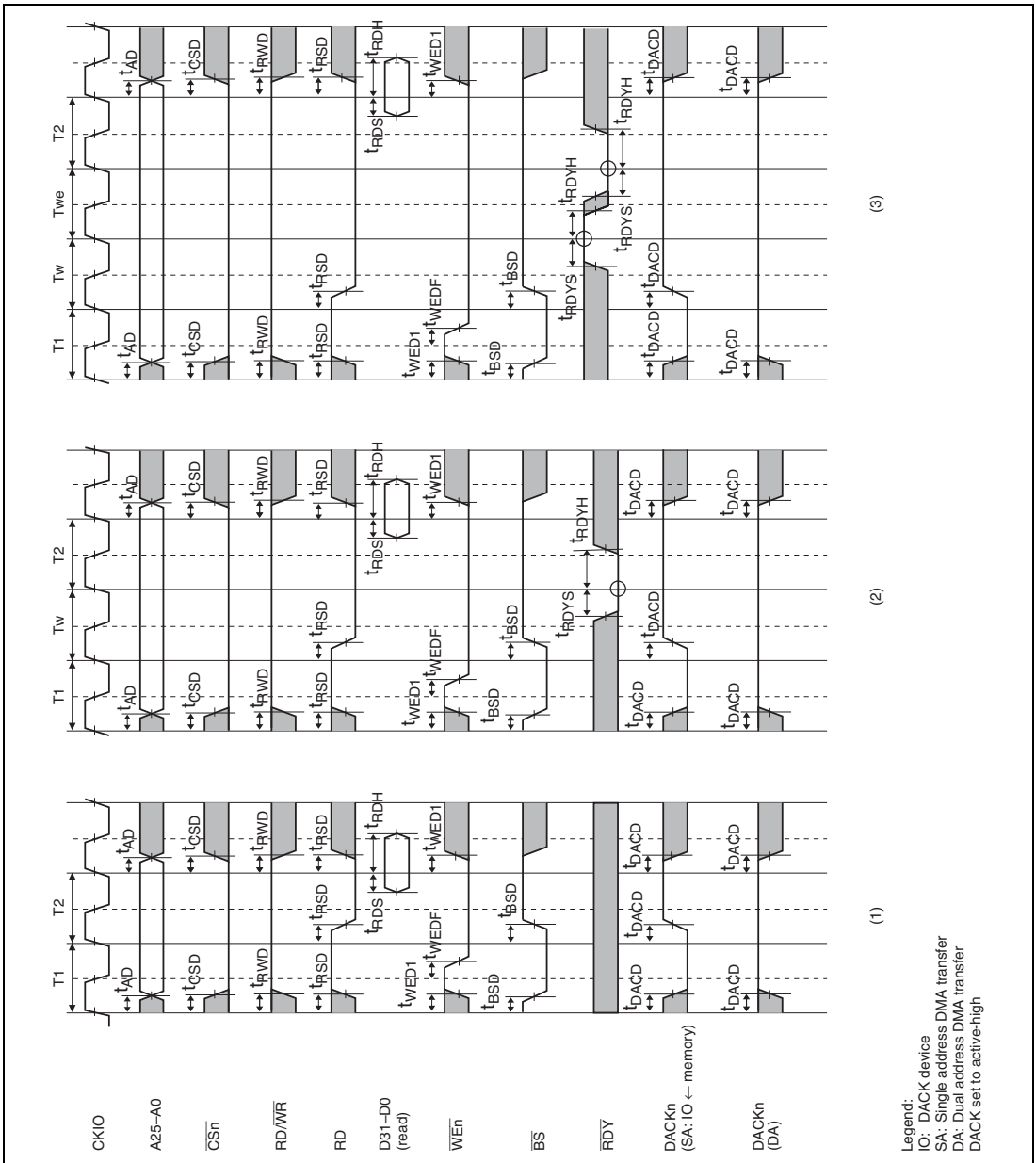
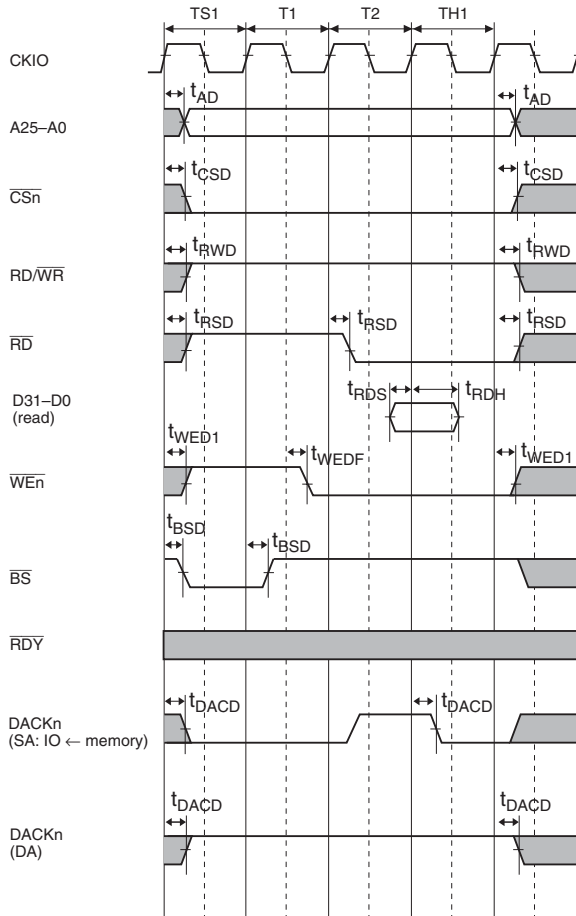


Figure 23.57 Memory Byte Control SRAM Bus Cycles
(1) Basic Read Cycle (No Wait)
(2) Basic Read Cycle (One Internal Wait)
(3) Basic Read Cycle (One Internal Wait + One External Wait)



Legend:
 IO: DACK device
 SA: Single address DMA transfer
 DA: Dual address DMA transfer
 DACK set to active-high

Figure 23.58 Memory Byte Control SRAM Bus Cycle: Basic Read Cycle (No Wait, Address Setup/Hold Time Insertion, AnS [0] = 1, AnH [1:0] = 01)

23.3.4 Peripheral Module Signal Timing

Table 23.23 Peripheral Module Signal Timing (1)

| Module | Item | Symbol | HD6417751 RBP240 (V) | | HD6417751 RBP200 (V) | | HD6417751 RBG240 (V) | | HD6417751 RBG200 (V) | | Unit | Figure | Notes |
|--------------------------------------|---------------------------------------|--------------|-------------------------|-----|-------------------------|-----|-------------------------|-----|-------------------------|-----|--------------------|--------|-------|
| | | | Min | Max | Min | Max | Min | Max | Min | Max | | | |
| | | | *2 | | *2 | | *2 | | *2 | | | | |
| TMU, RTC | Timer clock pulse width (high) | t_{TCLKWH} | 4 | — | 4 | — | 4 | — | 4 | — | Pcyc* ¹ | 23.59 | |
| | Timer clock pulse width (low) | t_{TCLKWL} | 4 | — | 4 | — | 4 | — | 4 | — | Pcyc* ¹ | 23.59 | |
| | Timer clock rise time | t_{TCLKr} | — | 0.8 | — | 0.8 | — | 0.8 | — | 0.8 | Pcyc* ¹ | 23.59 | |
| | Timer clock fall time | t_{TCLKf} | — | 0.8 | — | 0.8 | — | 0.8 | — | 0.8 | Pcyc* ¹ | 23.59 | |
| | Oscillation settling time | t_{ROSC} | — | 3 | — | 3 | — | 3 | — | 3 | s | 23.60 | |
| SCI | Input clock cycle (asynchronous) | t_{Scyc} | 4 | — | 4 | — | 4 | — | 4 | — | Pcyc* ¹ | 23.61 | |
| | Input clock cycle (synchronous) | t_{Scyc} | 6 | — | 6 | — | 6 | — | 6 | — | Pcyc* ¹ | 23.61 | |
| | Input clock pulse width | t_{SCKW} | 0.4 | 0.6 | 0.4 | 0.6 | 0.4 | 0.6 | 0.4 | 0.6 | t_{Scyc} | 23.61 | |
| | Input clock rise time | t_{SCKr} | — | 0.8 | — | 0.8 | — | 0.8 | — | 0.8 | Pcyc* ¹ | 23.61 | |
| | Input clock fall time | t_{SCKf} | — | 0.8 | — | 0.8 | — | 0.8 | — | 0.8 | Pcyc* ¹ | 23.61 | |
| | Transfer data delay time | t_{TXD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | 23.62 | |
| | Receive data setup time (synchronous) | t_{RXS} | 16 | — | 16 | — | 16 | — | 16 | — | ns | 23.62 | |
| Receive data hold time (synchronous) | t_{RXH} | 16 | — | 16 | — | 16 | — | 16 | — | ns | 23.62 | | |

| Module | Item | Symbol | HD6417751 RBP240 (V) | | HD6417751 RBP200 (V) | | HD6417751 RBG240 (V) | | HD6417751 RBG200 (V) | | Unit | Figure | Notes |
|-----------|--------------------------------|----------------------|-------------------------|-----|-------------------------|-----|-------------------------|-----|-------------------------|-----|------------------|--------------|----------------------|
| | | | Min | Max | Min | Max | Min | Max | Min | Max | | | |
| | | | *2 | | *2 | | *2 | | *2 | | | | |
| I/O ports | Output data delay time | t_{PORTD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | 23.63 | |
| | Input data setup time | t_{PORTS} | 2 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | 23.63 | |
| | Input data hold time | t_{PORTH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | 23.63 | |
| DMAC | DREQ \bar{n} setup time | t_{DROS} | 2 | — | 2.5 | — | 3.5 | — | 3.5 | — | ns | 23.64 | |
| | DREQ \bar{n} hold time | t_{DRGH} | 1.5 | — | 1.5 | — | 1.5 | — | 1.5 | — | ns | 23.64 | |
| | DRAK n delay time | t_{DRAKD} | 1.5 | 5.3 | 1.5 | 5.3 | 1.5 | 6 | 1.5 | 6 | ns | 23.64 | |
| INTC | NMI pulse width (high) | t_{NMIH} | 5 | — | 5 | — | 5 | — | 5 | — | t_{cyc} | 23.69 | Normal or sleep mode |
| | | | 30 | — | 30 | — | 30 | — | 30 | — | ns | 23.69 | Standby mode |
| | NMI pulse width (low) | t_{NMIL} | 5 | — | 5 | — | 5 | — | 5 | — | t_{cyc} | 23.69 | Normal or sleep mode |
| | | | 30 | — | 30 | — | 30 | — | 30 | — | ns | 23.69 | Standby mode |
| H-UDI | Input clock cycle | t_{TCKcyc} | 50 | — | 50 | — | 50 | — | 50 | — | ns | 23.65, 23.67 | |
| | Input clock pulse width (high) | t_{TCKH} | 15 | — | 15 | — | 15 | — | 15 | — | ns | 23.65 | |
| | Input clock pulse width (low) | t_{TCKL} | 15 | — | 15 | — | 15 | — | 15 | — | ns | 23.65 | |
| | Input clock rise time | t_{TCKr} | — | 10 | — | 10 | — | 10 | — | 10 | ns | 23.65 | |
| | Input clock fall time | t_{TCKf} | — | 10 | — | 10 | — | 10 | — | 10 | ns | 23.65 | |
| | ASEBRK setup time | t_{ASEBRKS} | 10 | — | 10 | — | 10 | — | 10 | — | t_{cyc} | 23.66 | |
| | ASEBRK hold time | t_{ASEBRKH} | 10 | — | 10 | — | 10 | — | 10 | — | t_{cyc} | 23.66 | |

| | | HD6417751 | | HD6417751 | | | | | | | | | |
|--------|------------------------|--------------|-----|------------|-----|-----------|-----|-----------|-----|-----|--------------------|--------|-------|
| | | RBP240 (V) | | RBP200 (V) | | | | | | | | | |
| | | HD6417751 | | HD6417751 | | | | | | | | | |
| | | RBG240 (V) | | RBG200 (V) | | | | | | | | | |
| | | HD6417751 | | HD6417751 | | HD6417751 | | HD6417751 | | | | | |
| | | RBA240HV | | RBA240HV*3 | | RF240 (V) | | RF200 (V) | | | | | |
| | | *2 | | *2 | | *2 | | *2 | | | | | |
| Module | Item | Symbol | Min | Max | Min | Max | Min | Max | Min | Max | Unit | Figure | Notes |
| H-UDI | TDI/TMS setup time | t_{TDIS} | 15 | — | 15 | — | 15 | — | 15 | — | ns | 23.67 | |
| | TDI/TMS hold time | t_{TDIH} | 15 | — | 15 | — | 15 | — | 15 | — | ns | 23.67 | |
| | TDO delay time | t_{TDO} | 0 | 10 | 0 | 10 | 0 | 10 | 0 | 10 | ns | 23.67 | |
| | ASE-PINBRK pulse width | t_{PINBRK} | 2 | — | 2 | — | 2 | — | 2 | — | Pcyc* ¹ | 23.68 | |

- Notes:
1. Pcyc: P clock cycles
 2. $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to $+75^\circ\text{C}^{*4}$, $C_L = 30$ pF, PLL2 on
 3. This is the case when the device in use is an HD6417751RBA240HV running at 200 MHz.
 4. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.24 Peripheral Module Signal Timing (2)

| | | HD6417751BP167 (V) HD6417751F167 (V) | | | | | |
|-----------|---------------------------------------|---|-----|-----|--------------------|--------|-------|
| | | *2 | | | | | |
| Module | Item | Symbol | Min | Max | Unit | Figure | Notes |
| TMU, RTC | Timer clock pulse width (high) | t_{CLKWH} | 4 | — | Pcyc* ¹ | 23.59 | |
| | Timer clock pulse width (low) | t_{CLKWL} | 4 | — | Pcyc* ¹ | 23.59 | |
| | Timer clock rise time | t_{CLKr} | — | 0.8 | Pcyc* ¹ | 23.59 | |
| | Timer clock fall time | t_{CLKf} | — | 0.8 | Pcyc* ¹ | 23.59 | |
| | Oscillation settling time | t_{ROSC} | — | 3 | s | 23.60 | |
| SCI | Input clock cycle (asynchronous) | t_{Scyc} | 4 | — | Pcyc* ¹ | 23.61 | |
| | Input clock cycle (synchronous) | t_{Scyc} | 6 | — | Pcyc* ¹ | 23.61 | |
| | Input clock pulse width | t_{SCKW} | 0.4 | 0.6 | t_{Scyc} | 23.61 | |
| | Input clock rise time | t_{SCKr} | — | 0.8 | Pcyc* ¹ | 23.61 | |
| | Input clock fall time | t_{SCKf} | — | 0.8 | Pcyc* ¹ | 23.61 | |
| | Transfer data delay time | t_{TXD} | — | 30 | ns | 23.62 | |
| | Receive data setup time (synchronous) | t_{RXS} | 0.8 | — | Pcyc* ¹ | 23.62 | |
| | Receive data hold time (synchronous) | t_{RXH} | 0.8 | — | Pcyc* ¹ | 23.62 | |
| I/O ports | Output data delay time | t_{PORTD} | — | 8 | ns | 23.63 | |
| | Input data setup time | t_{PORTS} | 3.5 | — | ns | 23.63 | |
| | Input data hold time | t_{PORTH} | 1.5 | — | ns | 23.63 | |
| DMAC | DREQn setup time | t_{DRQS} | 3.5 | — | ns | 23.64 | |
| | DREQn hold time | t_{DRQH} | 1.5 | — | ns | 23.64 | |
| | DRAKn delay time | t_{DRAKD} | — | 8 | ns | 23.64 | |

| | | HD6417751BP167 (V) HD6417751F167 (V) | | | | | |
|--------|--------------------------------|---|-----|-----|-------------|-----------------|----------------------|
| | | *2 | | | | | |
| Module | Item | Symbol | Min | Max | Unit | Figure | Notes |
| INTC | NMI pulse width (high) | t_{NMIH} | 5 | — | t_{cyc} | 23.69 | Normal or sleep mode |
| | | | 30 | — | ns | 23.69 | Standby mode |
| | NMI pulse width (low) | t_{NMIL} | 5 | — | t_{cyc} | 23.69 | Normal or sleep mode |
| | | | 30 | — | ns | 23.69 | Standby mode |
| H-UDI | Input clock cycle | t_{TCKcyc} | 50 | — | ns | 23.65, 23.67 | |
| | Input clock pulse width (high) | t_{TCKH} | 15 | — | ns | 23.65 | |
| | Input clock pulse width (low) | t_{TCKL} | 15 | — | ns | 23.65 | |
| | Input clock rise time | t_{TCKr} | — | 10 | ns | 23.65 | |
| | Input clock fall time | t_{TCKf} | — | 10 | ns | 23.65 | |
| | ASEBRK $\bar{}$ setup time | $t_{ASEBRKS}$ | 10 | — | t_{cyc} | 23.66 | |
| | ASEBRK hold time | $t_{ASEBRKH}$ | 10 | — | t_{cyc} | 23.66 | |
| | TDI/TMS setup time | t_{TDIS} | 15 | — | ns | 23.67 | |
| | TDI/TMS hold time | t_{TDIH} | 15 | — | ns | 23.67 | |
| | TDO delay time | t_{TDO} | 0 | 10 | ns | 23.67 | |
| | ASE-PINBRK pulse width | t_{PINBRK} | 2 | — | $Pcyc^{*1}$ | 23.68 | |

Notes: 1. $Pcyc$: P clock cycles

2. $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.8$ V, $T_a = -20$ to 75°C , $C_L = 30$ pF, PLL2 on

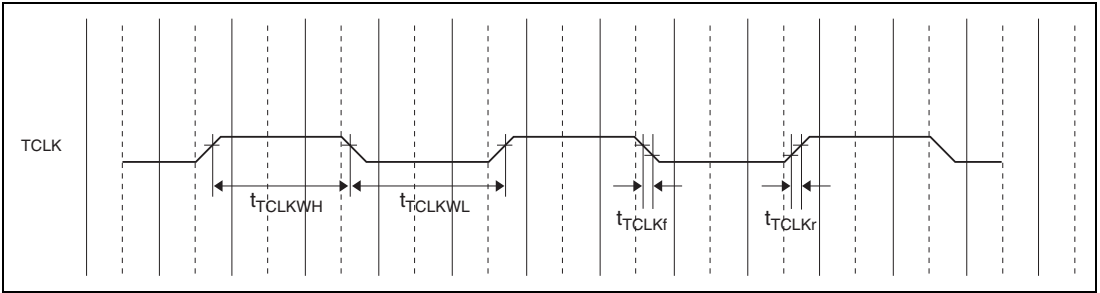


Figure 23.59 TCLK Input Timing

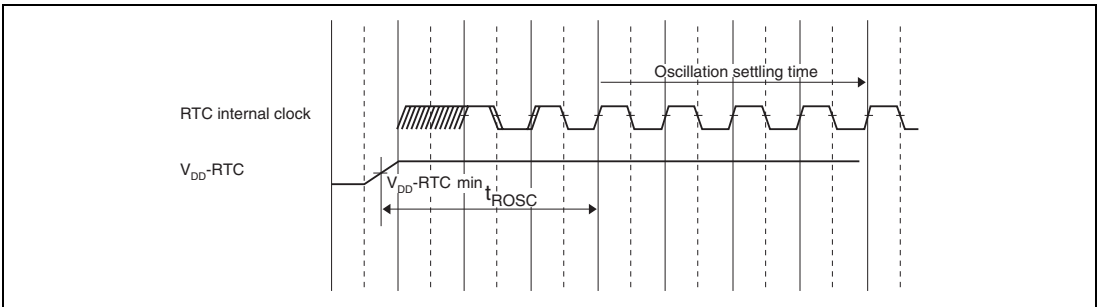


Figure 23.60 RTC Oscillation Settling Time at Power-On

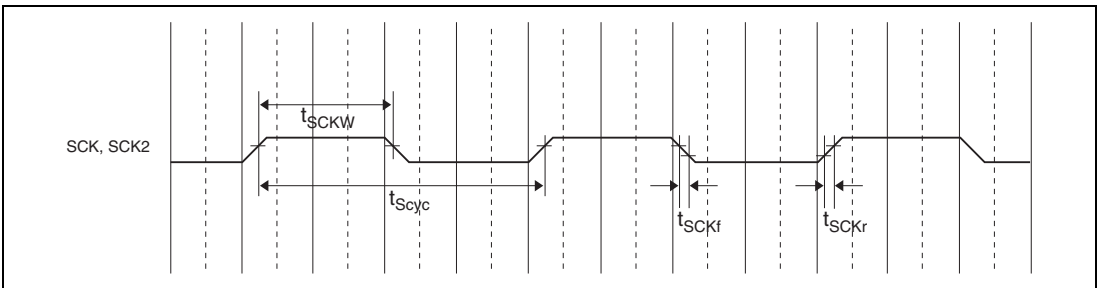


Figure 23.61 SCK Input Clock Timing

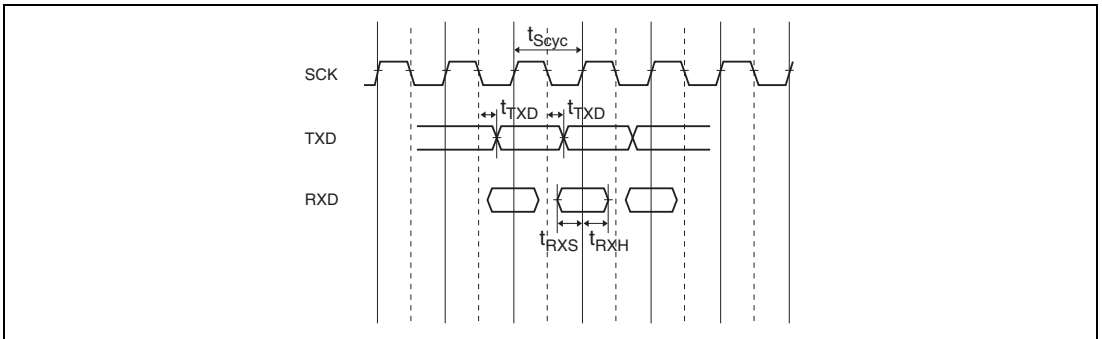


Figure 23.62 SCI I/O Synchronous Mode Clock Timing

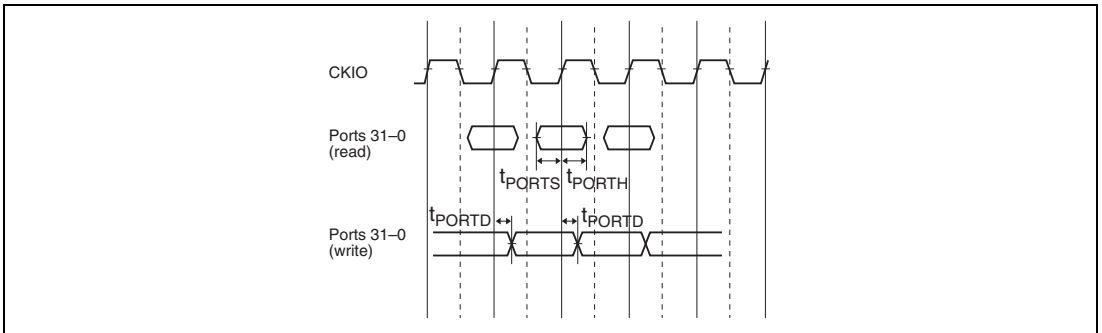


Figure 23.63 I/O Port Input/Output Timing

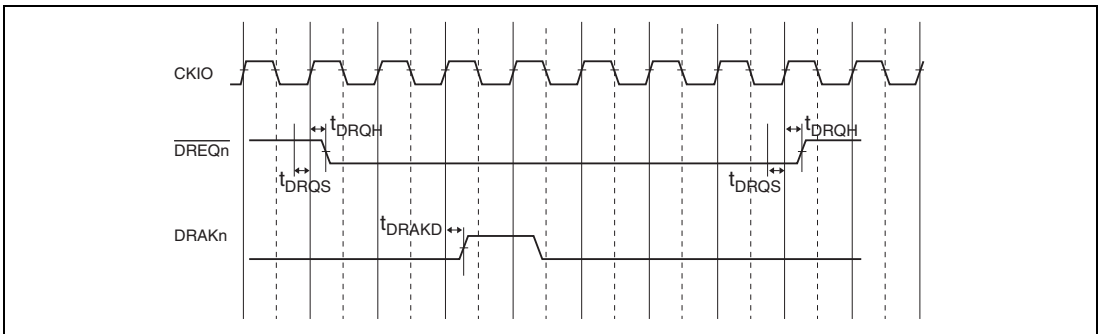


Figure 23.64 (a) $\overline{\text{DREQ}}$ /DRAK Timing

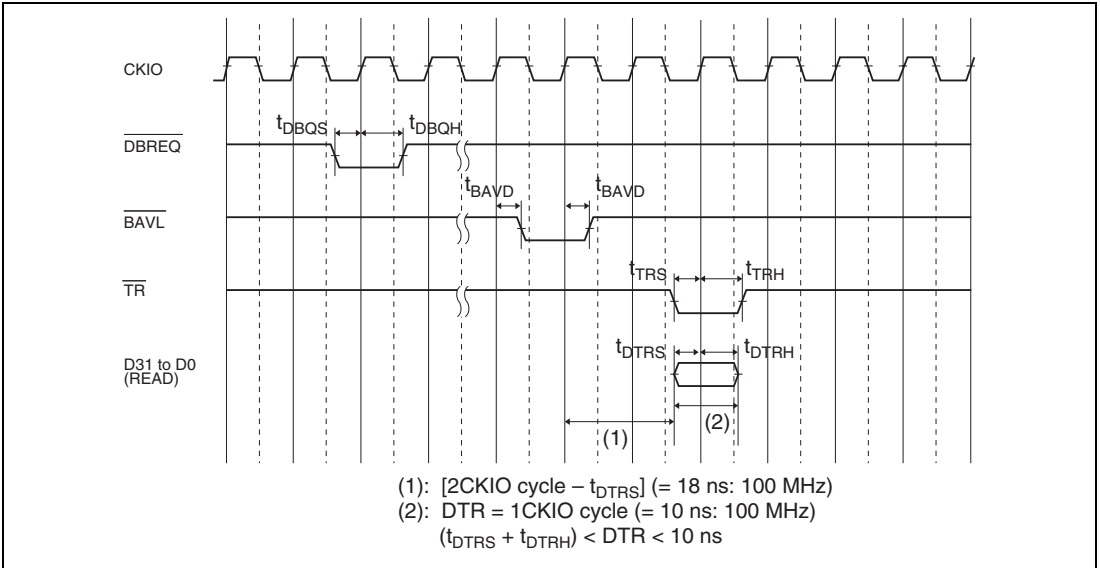


Figure 23.64 (b) $\overline{\text{DBREQ}}/\overline{\text{TR}}$ Input Timing and $\overline{\text{BAVL}}$ Output Timing

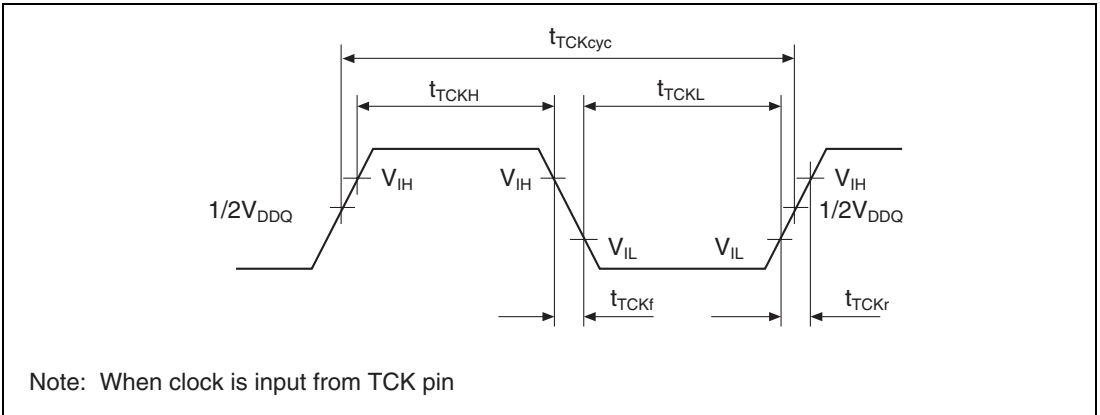


Figure 23.65 TCK Input Timing

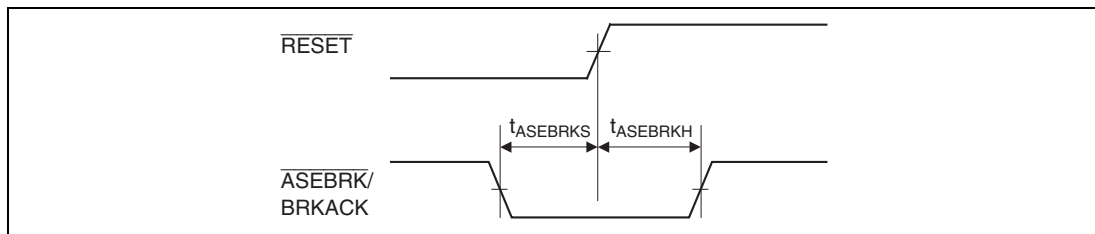
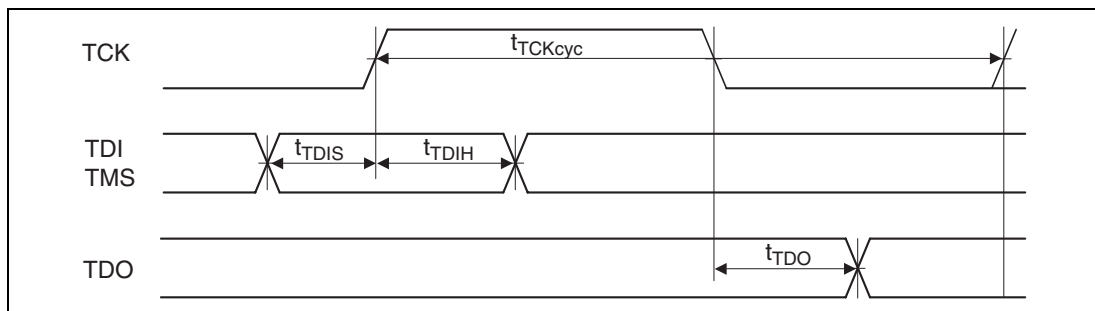
Figure 23.66 $\overline{\text{RESET}}$ Hold Timing

Figure 23.67 H-UDI Data Transfer Timing

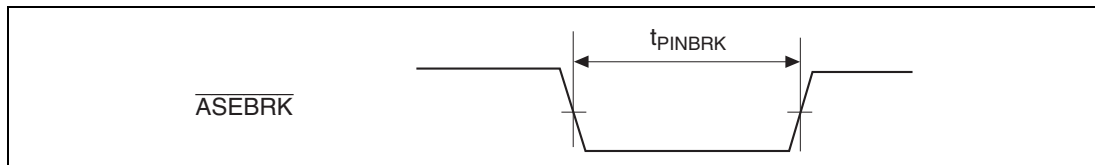


Figure 23.68 Pin Break Timing

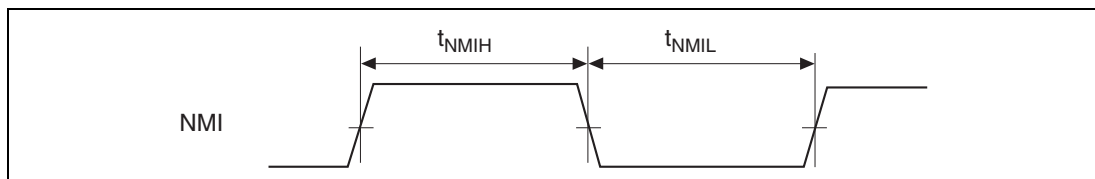


Figure 23.69 NMI Input Timing

Table 23.25 PCIC Signal Timing (in PCIREQ/PCIGNT Non-Port Mode) (1)

HD6417751RBP240 (V), HD6417751RBP200 (V), HD6417751RBG240 (V),
 HD6417751RBG200 (V), HD6417751RBA240HV, HD6417751RF240 (V), HD6417751RF200
 (V): $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C^{*2} , $C_L = 30$ pF

| Pin | Item | Symbol | 33 MHz | | 66 MHz | | Unit | Figure |
|--------------------------------|-------------------------------------|---------------|--------------------------|-----|-----------------------------|-----|------|--------|
| | | | Min | Max | Min | Max | | |
| PCICLK | Clock cycle | t_{PCICYC} | 30 | — | 15 | 30 | ns | 23.70 |
| | Clock pulse width (high) | $t_{PCIHIGH}$ | 11 | — | 6 | — | ns | 23.70 |
| | Clock pulse width (low) | t_{PCILOW} | 11 | — | 6 | — | ns | 23.70 |
| | Clock rise time | t_{PCIr} | — | 4 | — | 1.5 | ns | 23.70 |
| | Clock fall time | t_{PCIf} | — | 4 | — | 1.5 | ns | 23.70 |
| PCIRST | Output data delay time | t_{PCIVAL} | — | 10 | — | 8 | ns | 23.71 |
| IDSEL | Input hold time | t_{PCIH} | 1.5 | — | 1.5 | — | ns | 23.72 |
| | Input setup time | t_{PCISU} | 3.0 (3.5 ^{*1}) | — | 3.0 (3.5 ^{*1}) | — | ns | 23.72 |
| AD31–AD0 | Output data delay time | t_{PCIVAL} | — | 10 | — | 8 | ns | 23.71 |
| C/BE3–C/BE0 | Tri-state drive delay time | t_{PCION} | — | 10 | — | 10 | ns | 23.71 |
| PAR | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |
| PCIFRAME | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |
| IRDY | Input hold time | t_{PCIH} | 1.5 | — | 1.5 | — | ns | 23.72 |
| TRDY | Input setup time | t_{PCISU} | 3.0 (3.5 ^{*1}) | — | 3.0 (3.5 ^{*1}) | — | ns | 23.72 |
| PCISTOP | Input setup time | t_{PCISU} | 3.0 (3.5 ^{*1}) | — | 3.0 (3.5 ^{*1}) | — | ns | 23.72 |
| PCILOCK | | | | | | | | |
| DEVSEL | | | | | | | | |
| PERR | | | | | | | | |
| PCIREQ1/ GNTIN | Output data delay time | t_{PCIVAL} | — | 10 | — | 8 | ns | 23.71 |
| | Tri-state drive delay time | t_{PCION} | — | 10 | — | 10 | ns | 23.71 |
| PCIREQ2/MD9 | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |
| PCIREQ3/ MD10 | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |
| | Input hold time | t_{PCIH} | 1.5 | — | 1.5 | — | ns | 23.72 |
| PCIREQ4/ PCIGNT1/ REQOUT | Input setup time | t_{PCISU} | 3.0 (3.5 ^{*1}) | — | 3.0 (3.5 ^{*1}) | — | ns | 23.72 |
| | Input setup time | t_{PCISU} | 3.0 (3.5 ^{*1}) | — | 3.0 (3.5 ^{*1}) | — | ns | 23.72 |
| PCIGNT4– PCIGNT1 | Input setup time | t_{PCISU} | 3.0 (3.5 ^{*1}) | — | 3.0 (3.5 ^{*1}) | — | ns | 23.72 |
| SERR | Tri-state drive delay time | t_{PCION} | — | 10 | — | 10 | ns | 23.71 |
| INTA | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |

Notes: 1. HD6417751RF240 (V), HD6417751RF200 (V)
 2. $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.26 PCIC Signal Timing (in PCIREQ/PCIGNT Non-Port Mode) (2)

HD6417751BP167 (V), HD6417751F167 (V): $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.8$ V, $T_a = -20$ to 75°C ,
 $C_L = 30$ pF

| Pin | Item | Symbol | 33 MHz | | 66 MHz | | Unit | Figure |
|--------------------------------|-------------------------------------|---------------|---------------|-----|------------|-----|------|--------|
| | | | Min | Max | Min | Max | | |
| PCICLK | Clock cycle | t_{PCICYC} | 30 | — | 15 | 30 | ns | 23.70 |
| | Clock pulse width (high) | $t_{PCIHIGH}$ | 11 | — | 6 | — | ns | 23.70 |
| | Clock pulse width (low) | t_{PCILOW} | 11 | — | 6 | — | ns | 23.70 |
| | Clock rise time | t_{PCIR} | — | 4 | — | 1.5 | ns | 23.70 |
| | Clock fall time | t_{PCIF} | — | 4 | — | 1.5 | ns | 23.70 |
| PCIRST | Output data delay time | t_{PCIVAL} | — | 10 | — | 10 | ns | 23.71 |
| IDSEL | Input hold time | t_{PCIH} | 1 | — | 1 | — | ns | 23.72 |
| | Input setup time | t_{PCISU} | 3.0 (3.5*) | — | 3.0 (3.5*) | — | ns | 23.72 |
| AD31–AD0 | Output data delay time | t_{PCIVAL} | — | 10 | — | 10 | ns | 23.71 |
| C/BE3–C/BE0 | Tri-state drive delay time | t_{PCION} | — | 10 | — | 10 | ns | 23.71 |
| PAR | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |
| PCIFRAME | | | | | | | | |
| IRDY | Input hold time | t_{PCIH} | 1 | — | 1 | — | ns | 23.72 |
| TRDY | | | | | | | | |
| PCISTOP | Input setup time | t_{PCISU} | 3.0 (3.5*) | — | 3.0 (3.5*) | — | ns | 23.72 |
| PCIOCK | | | | | | | | |
| DEVSEL | | | | | | | | |
| PERR | | | | | | | | |
| PCIREQ1/ GNTIN | Output data delay time | t_{PCIVAL} | — | 10 | — | 10 | ns | 23.71 |
| | Tri-state drive delay time | t_{PCION} | — | 10 | — | 10 | ns | 23.71 |
| PCIREQ2/ MD9 | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |
| PCIREQ3/ MD10 | Input hold time | t_{PCIH} | 1 | — | 1 | — | ns | 23.72 |
| PCIREQ4/ PCIGNT1/ REQOUT | Input setup time | t_{PCISU} | 3.0 (3.5*) | — | 3.0 (3.5*) | — | ns | 23.72 |
| PCIGNT4– PCIGNT1 | | | | | | | | |
| SERR | Tri-state drive delay time | t_{PCION} | — | 10 | — | 10 | ns | 23.71 |
| INTA | Tri-state high-impedance delay time | t_{PCIOFF} | — | 12 | — | 12 | ns | 23.71 |

Note: * HD6417751F167 (V)

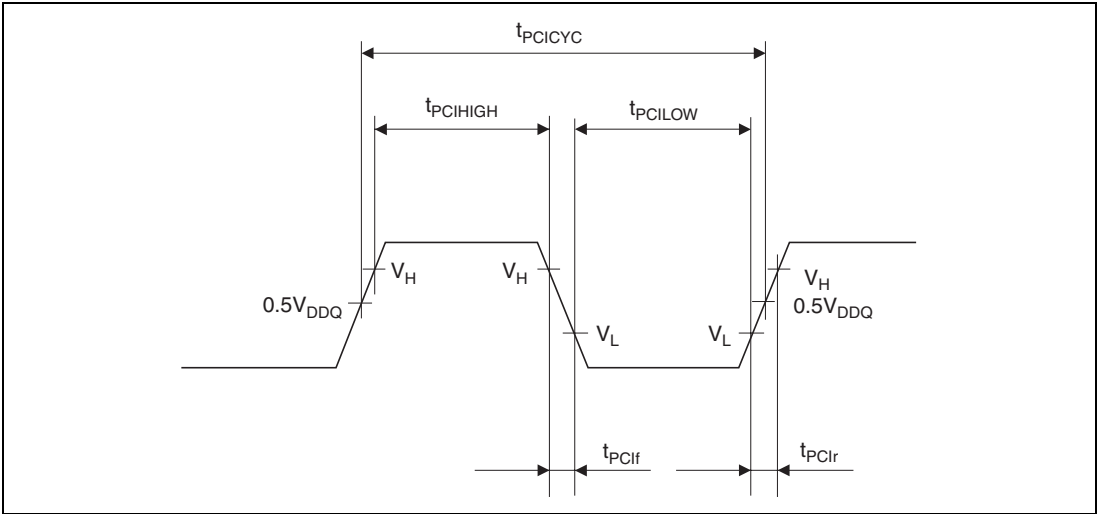


Figure 23.70 PCI Clock Input Timing

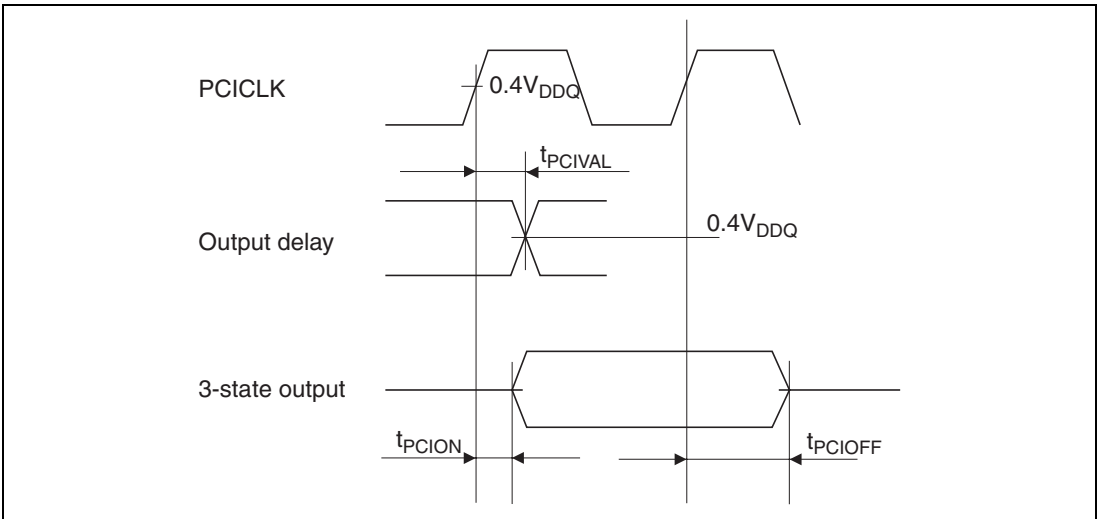


Figure 23.71 Output Signal Timing

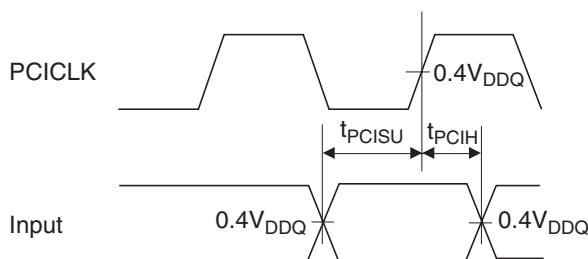


Figure 23.72 Output Signal Timing

Table 23.27 PCIC Signal Timing
(With PCIREQ/PCIGNT Port Settings in Non-Host Mode) (1)

HD6417751RBP240 (V), HD6417751RBP200 (V), HD6417751RBG240 (V),
HD6417751RBG200 (V), HD6417751RBA240HV, HD6417751RF240 (V), HD6417751RF200
(V): $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.5$ V, $T_a = -20$ to 75°C^* , $C_L = 30$ pF

| Pin | Item | Symbol | Min | Max | Unit | Figure |
|-----------------|------------------------|----------------|-----|-----|------|--------|
| PCIREQ2/MD9 | Output data delay time | $t_{PCIPORTD}$ | — | 10 | ns | 23.73 |
| PCIREQ3/MD10 | | | | | | |
| PCIREQ4 | Input hold time | $t_{PCIPORTH}$ | 1.5 | — | ns | 23.73 |
| | Input setup time | $t_{PCIPORTS}$ | 3.5 | — | ns | 23.73 |
| PCIGNT4–PCIGNT1 | Output data delay time | $t_{PCIPORTD}$ | — | 10 | ns | 23.73 |

Note: * $T_a = -40$ to 85°C for the HD6417751RBA240HV.

Table 23.28 PCIC Signal Timing
(With PCIREQ/PCIGNT Port Settings in Non-Host Mode) (2)

HD6417751BP167 (V), HD6417751F167 (V): $V_{DDQ} = 3.0$ to 3.6 V, $V_{DD} = 1.8$ V,
 $T_a = -20$ to 75°C , $C_L = 30$ pF

| Pin | Item | Symbol | Min | Max | Unit | Figure |
|-----------------|------------------------|----------------|-----|-----|------|--------|
| PCIREQ2/MD9 | Output data delay time | $t_{PCIPORTD}$ | — | 10 | ns | 23.73 |
| PCIREQ3/MD10 | | | | | | |
| PCIREQ4 | Input hold time | $t_{PCIPORTH}$ | 1.5 | — | ns | 23.73 |
| | Input setup time | $t_{PCIPORTS}$ | 3.5 | — | ns | 23.73 |
| PCIGNT4–PCIGNT1 | Output data delay time | $t_{PCIPORTD}$ | — | 10 | ns | 23.73 |

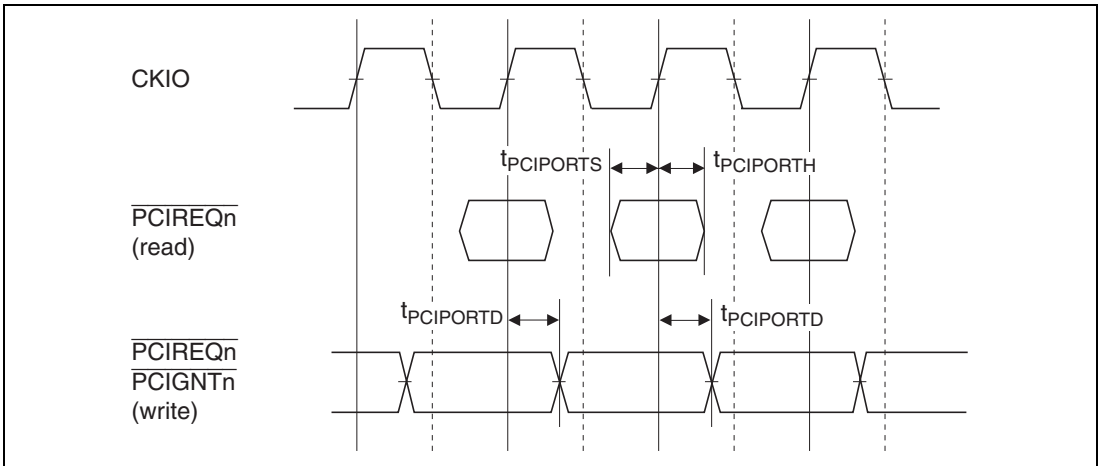


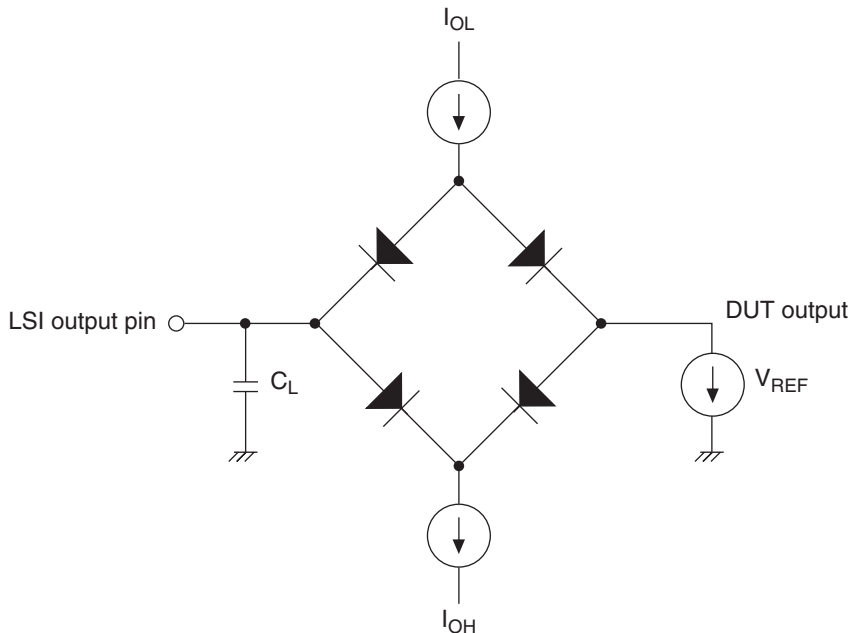
Figure 23.73 I/O Port Input/Output Timing

23.3.5 AC Characteristic Test Conditions

The AC characteristic test conditions are as follows:

- Input/output signal reference level: 1.5 V ($V_{DDQ} = 3.3 \pm 0.3$ V)
- Input pulse level: V_{SSQ} to 3.0 V (V_{SSQ} to V_{DDQ} for $\overline{\text{RESET}}$, $\overline{\text{TRST}}$, $\overline{\text{NMI}}$, and $\overline{\text{ASEBRK/BRKACK}}$)
- Input rise/fall time: 1 ns

The output load circuit is shown in figure 23.74



- Notes:
1. C_L is the total value, including the capacitance of the test jig, etc.
The capacitance of each pin is set to 30 pF.
 2. I_{OL} and I_{OH} values are as shown in table 23.10, Permissible Output Currents.

Figure 23.74 Output Load Circuit

23.3.6 Change in Delay Time Based on Load Capacitance

Figure 23.75 is a chart showing the changes in the delay time (reference data) when a load capacitance equal to or larger than the stipulated value (30 pF) is connected to the LSI pins. When connecting an external device with a load capacitance exceeding the regulation, use the chart in figure 23.75 as reference for system design.

Note that if the load capacitance to be connected exceeds the range shown in figure 23.75 the graph will not be a straight line.

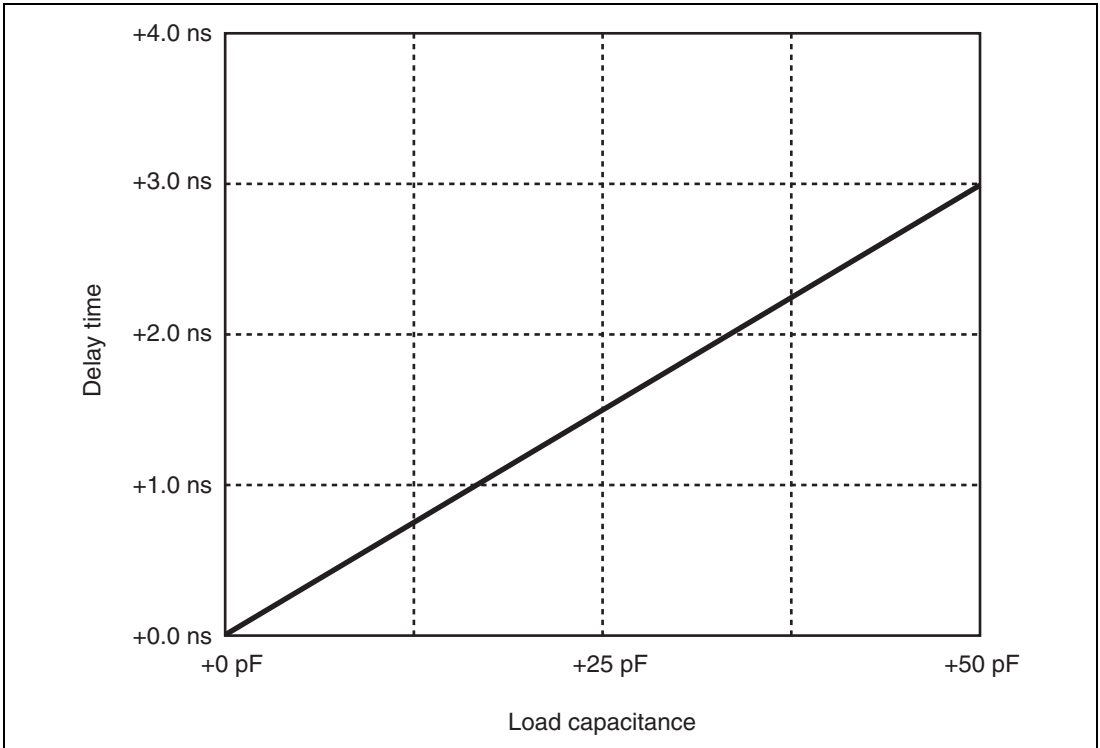


Figure 23.75 Load Capacitance–Delay Time

Appendix A Address List

Table A.1 Address List

| Module | Register | P4 Address | Area 7 Address* ¹ | Size | Power-On Reset | Manual Reset | Sleep | Stand-by | Synchronization Clock |
|--------|-----------------|----------------------------------|----------------------------------|-----------------|--|--------------|-------|----------|-----------------------|
| PCIC | PCIMEM | H'FD00 0000 to H'FDFF FFFF | H'FD00 0000 to H'FDFF FFFF | 8, 16, 32 | According to PCI memory space | | | | Pck |
| INTC | INTPRI00 | H'FE08 0000 | H'1E08 0000 | 32 | H'0000 0000 | Held | Held | Held | Pck |
| INTC | INTREQ00 | H'FE08 0020 | H'1E08 0020 | 32 | H'0000 0000 | Held | Held | Held | Pck |
| INTC | INTMSK00 | H'FE08 0040 | H'1E08 0040 | 32 | H'0000 03FF | Held | Held | Held | Pck |
| INTC | INTMSKCLR 00 | H'FE08 0060 | H'1E08 0060 | 32 | Write-only | | | | Pck |
| CPG | CLKSTP00 | H'FE0A 0000 | H'1E0A 0000 | 32 | H'0000 0000 | Held | Held | Held | Pck |
| CPG | CLKSTPCLR 00 | H'FE0A 0008 | H'1E0A 0008 | 32 | Write-only | | | | Pck |
| TMU | TSTR2 | H'FE10 0004 | H'1E10 0004 | 8 | H'00 | Held | Held | Held | Pck |
| TMU | TCOR3 | H'FE10 0008 | H'1E10 0008 | 32 | H'FFFF FFFF | Held | Held | Held | Pck |
| TMU | TCNT3 | H'FE10 000C | H'1E10 000C | 32 | H'FFFF FFFF | Held | Held | Held | Pck |
| TMU | TCR3 | H'FE10 0010 | H'1E10 0010 | 16 | H'0000 | Held | Held | Held | Pck |
| TMU | TCOR4 | H'FE10 0014 | H'1E10 0014 | 32 | H'FFFF FFFF | Held | Held | Held | Pck |
| TMU | TCNT4 | H'FE10 0018 | H'1E10 0018 | 32 | H'FFFF FFFF | Held | Held | Held | Pck |
| TMU | TCR4 | H'FE10 001C | H'1E10 001C | 16 | H'0000 | Held | Held | Held | Pck |
| PCIC | PCICONF0 | H'FE20 0000 | H'1E20 0000 | 32 | H'35051054 (SH7751)/ H'350E1054 (SH7751R) | | | | Pck |
| PCIC | PCICONF1 | H'FE20 0004 | H'1E20 0004 | 32 | H'02900080 | Held | Held | Held | Pck |
| PCIC | PCICONF2 | H'FE20 0008 | H'1E20 0008 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCICONF3 | H'FE20 000C | H'1E20 000C | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF4 | H'FE20 0010 | H'1E20 0010 | 32 | H'00000001 | Held | Held | Held | Pck |
| PCIC | PCICONF5 | H'FE20 0014 | H'1E20 0014 | 32 | H'00000000 | Held | Held | Held | Pck |

| Module | Register | P4 Address | Area 7 Address* ¹ | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|-----------|-------------|---------------------------------|------|-------------------|-----------------|-------|--------------|------------------------------|
| PCIC | PCICONF6 | H'FE20 0018 | H'1E20 0018 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF7 | H'FE20 001C | H'1E20 001C | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF8 | H'FE20 0020 | H'1E20 0020 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF9 | H'FE20 0024 | H'1E20 0024 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF10 | H'FE20 0028 | H'1E20 0028 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICON111 | H'FE20 002C | H'1E20 002C | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCICONF12 | H'FE20 0030 | H'1E20 0030 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF13 | H'FE20 0034 | H'1E20 0034 | 32 | H'00000040 | Held | Held | Held | Pck |
| PCIC | PCICONF14 | H'FE20 0038 | H'1E20 0038 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICONF15 | H'FE20 003C | H'1E20 003C | 32 | H'00000100 | Held | Held | Held | Pck |
| PCIC | PCICONF16 | H'FE20 0040 | H'1E20 0040 | 32 | H'00010001 | Held | Held | Held | Pck |
| PCIC | PCICONF17 | H'FE20 0044 | H'1E20 0044 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICR | H'FE20 0100 | H'1E20 0100 * ² | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCILSR0 | H'FE20 0104 | H'1E20 0104 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCILSR1 | H'FE20 0108 | H'1E20 0108 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCILAR0 | H'FE20 010C | H'1E20 010C | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCILAR1 | H'FE20 0110 | H'1E20 0110 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIINT | H'FE20 0114 | H'1E20 0114 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIINTM | H'FE20 0118 | H'1E20 0118 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIALR | H'FE20 011C | H'1E20 011C | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCICLR | H'FE20 0120 | H'1E20 0120 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCIAINT | H'FE20 0130 | H'1E20 0130 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIAINTM | H'FE20 0134 | H'1E20 0134 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIBLLR | H'FE20 0138 | H'1E20 0138 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCIDMABT | H'FE20 0140 | H'1E20 0140 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDPA0 | H'FE20 0180 | H'1E20 0180 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDLA0 | H'FE20 0184 | H'1E20 0184 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDTC0 | H'FE20 0188 | H'1E20 0188 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDCR0 | H'FE20 018C | H'1E20 018C | 32 | H'00000000 | Held | Held | Held | Pck |

| Module | Register | P4 Address | Area 7 Address*1 | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|----------|----------------------------------|----------------------------------|-----------------|----------------------------|-----------------|-------|--------------|------------------------------|
| PCIC | PCIDPA1 | H'FE20 0190 | H'1E20 0190 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDLA1 | H'FE20 0194 | H'1E20 0194 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDTC1 | H'FE20 0198 | H'1E20 0198 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDCR1 | H'FE20 019C | H'1E20 019C | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDPA2 | H'FE20 01A0 | H'1E20 01A0 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDLA2 | H'FE20 01A4 | H'1E20 01A4 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDTC2 | H'FE20 01A8 | H'1E20 01A8 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDCR2 | H'FE20 01AC | H'1E20 01AC | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDPA3 | H'FE20 01B0 | H'1E20 01B0 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDLA3 | H'FE20 01B4 | H'1E20 01B4 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDTC3 | H'FE20 01B8 | H'1E20 01B8 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIDCR3 | H'FE20 01BC | H'1E20 01BC | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIPAR | H'FE20 01C0 | H'1E20 01C0 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCIMBR | H'FE20 01C4 | H'1E20 01C4 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCIIOBR | H'FE20 01C8 | H'1E20 01C8 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCIPINT | H'FE20 01CC | H'1E20 01CC | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIPINTM | H'FE20 01D0 | H'1E20 01D0 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCICLKR | H'FE20 01D4 | H'1E20 01D4 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIBCR1 | H'FE20 01E0 | H'1E20 01E0 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIBCR2 | H'FE20 01E4 | H'1E20 01E4 | 32 | H'00003FFC | Held | Held | Held | Pck |
| PCIC | PCIBCR3 | H'FE20 01F8 | H'1E20 01F8 | 32 | H'0000 0001 | Held | Held | Held | Pck |
| PCIC | PCIWCR1 | H'FE20 01E8 | H'1E20 01E8 | 32 | H'7777 7777 | Held | Held | Held | Pck |
| PCIC | PCIWCR2 | H'FE20 01EC | H'1E20 01EC | 32 | H'FFFE EFFF | Held | Held | Held | Pck |
| PCIC | PCIWCR3 | H'FE20 01F0 | H'1E20 01F0 | 32 | H'0777 7777 | Held | Held | Held | Pck |
| PCIC | PCIMCR | H'FE20 01F4 | H'1E20 01F4 | 32 | H'0000 0000 | Held | Held | Held | Pck |
| PCIC | PCIPCTR | H'FE20 0200 | H'1E20 0200 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIPDTR | H'FE20 0204 | H'1E20 0204 | 32 | H'00000000 | Held | Held | Held | Pck |
| PCIC | PCIPDR | H'FE20 0220 | H'1E20 0220 | 32 | Undefined | Held | Held | Held | Pck |
| PCIC | PCIIO | H'FE24 0000 to H'FE27 FFFF | H'1E24 0000 to H'1E27 FFFF | 8, 16, 32 | According to PCI I/O space | | | | Pck |

| Module | Register | P4 Address | Area 7 Address* ¹ | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|----------|-------------|---------------------------------|------|----------------------|-----------------|-------|--------------|------------------------------|
| CCN | PTEH | H'FF00 0000 | H'1F00 0000 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | PTL | H'FF00 0004 | H'1F00 0004 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | TTB | H'FF00 0008 | H'1F00 0008 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | TEA | H'FF00 000C | H'1F00 000C | 32 | Undefined | Held | Held | Held | lck |
| CCN | MMUCR | H'FF00 0010 | H'1F00 0010 | 32 | H'0000 0000 | H'0000 0000 | Held | Held | lck |
| CCN | BASRA | H'FF00 0014 | H'1F00 0014 | 8 | Undefined | Held | Held | Held | lck |
| CCN | BASRB | H'FF00 0018 | H'1F00 0018 | 8 | Undefined | Held | Held | Held | lck |
| CCN | CCR | H'FF00 001C | H'1F00 001C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | lck |
| CCN | TRA | H'FF00 0020 | H'1F00 0020 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | EXPEVT | H'FF00 0024 | H'1F00 0024 | 32 | H'0000 0000 | H'0000 0020 | Held | Held | lck |
| CCN | INTEVT | H'FF00 0028 | H'1F00 0028 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | PTEA | H'FF00 0034 | H'1F00 0034 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | QACR0 | H'FF00 0038 | H'1F00 0038 | 32 | Undefined | Undefined | Held | Held | lck |
| CCN | QACR1 | H'FF00 003C | H'1F00 003C | 32 | Undefined | Undefined | Held | Held | lck |
| | | | | | | | | | |
| UBC | BARA | H'FF20 0000 | H'1F20 0000 | 32 | Undefined | Held | Held | Held | lck |
| UBC | BAMRA | H'FF20 0004 | H'1F20 0004 | 8 | Undefined | Held | Held | Held | lck |
| UBC | BBRA | H'FF20 0008 | H'1F20 0008 | 16 | H'0000 | Held | Held | Held | lck |
| UBC | BARB | H'FF20 000C | H'1F20 000C | 32 | Undefined | Held | Held | Held | lck |
| UBC | BAMRB | H'FF20 0010 | H'1F20 0010 | 8 | Undefined | Held | Held | Held | lck |
| UBC | BBRB | H'FF20 0014 | H'1F20 0014 | 16 | H'0000 | Held | Held | Held | lck |
| UBC | BDRB | H'FF20 0018 | H'1F20 0018 | 32 | Undefined | Held | Held | Held | lck |
| UBC | BDMRB | H'FF20 001C | H'1F20 001C | 32 | Undefined | Held | Held | Held | lck |
| UBC | BRCR | H'FF20 0020 | H'1F20 0020 | 16 | H'0000* ² | Held | Held | Held | lck |
| | | | | | | | | | |
| BSC | BCR1 | H'FF80 0000 | H'1F80 0000 | 32 | H'0000 0000 | Held | Held | Held | Bck |
| BSC | BCR2 | H'FF80 0004 | H'1F80 0004 | 16 | H'3FFC | Held | Held | Held | Bck |
| BSC | BCR3 | H'FF80 0050 | H'1F80 0050 | 16 | H'0000 | Held | Held | Held | Bck |
| BSC | BCR4 | H'FE0A 00F0 | H'1E0A 00F0 | 32 | H'0000 0000 | Held | Held | Held | Bck |
| BSC | WCR1 | H'FF80 0008 | H'1F80 0008 | 32 | H'7777 7777 | Held | Held | Held | Bck |
| BSC | WCR2 | H'FF80 000C | H'1F80 000C | 32 | H'FFFE EFFF | Held | Held | Held | Bck |

| Module | Register | P4 Address | Area 7 Address*1 | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|----------|-------------|---------------------|------|-------------------|-----------------|-------|--------------|------------------------------|
| BSC | WCR3 | H'FF80 0010 | H'1F80 0010 | 32 | H'0777 7777 | Held | Held | Held | Bck |
| BSC | MCR | H'FF80 0014 | H'1F80 0014 | 32 | H'0000 0000 | Held | Held | Held | Bck |
| BSC | PCR | H'FF80 0018 | H'1F80 0018 | 16 | H'0000 | Held | Held | Held | Bck |
| BSC | RTCSPR | H'FF80 001C | H'1F80 001C | 16 | H'0000 | Held | Held | Held | Bck |
| BSC | RTCNT | H'FF80 0020 | H'1F80 0020 | 16 | H'0000 | Held | Held | Held | Bck |
| BSC | RTCOR | H'FF80 0024 | H'1F80 0024 | 16 | H'0000 | Held | Held | Held | Bck |
| BSC | RFCR | H'FF80 0028 | H'1F80 0028 | 16 | H'0000 | Held | Held | Held | Bck |
| BSC | PCTRA | H'FF80 002C | H'1F80 002C | 32 | H'0000 0000 | Held | Held | Held | Bck |
| BSC | PDTRA | H'FF80 0030 | H'1F80 0030 | 16 | Undefined | Held | Held | Held | Bck |
| BSC | PCTRB | H'FF80 0040 | H'1F80 0040 | 32 | H'0000 0000 | Held | Held | Held | Bck |
| BSC | PDTRB | H'FF80 0044 | H'1F80 0044 | 16 | Undefined | Held | Held | Held | Bck |
| BSC | GPIOIC | H'FF80 0048 | H'1F80 0048 | 16 | H'0000 0000 | Held | Held | Held | Bck |
| BSC | SDMR2 | H'FF90 xxxx | H'1F90 xxxx | 8 | Write-only | | | | Bck |
| BSC | SDMR3 | H'FF94 xxxx | H'1F94 xxxx | 8 | | | | | Bck |
| DMAC | SAR0 | H'FFA0 0000 | H'1FA0 0000 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR0 | H'FFA0 0004 | H'1FA0 0004 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR0 | H'FFA0 0008 | H'1FA0 0008 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR0 | H'FFA0 000C | H'1FA0 000C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | SAR1 | H'FFA0 0010 | H'1FA0 0010 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR1 | H'FFA0 0014 | H'1FA0 0014 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR1 | H'FFA0 0018 | H'1FA0 0018 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR1 | H'FFA0 001C | H'1FA0 001C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | SAR2 | H'FFA0 0020 | H'1FA0 0020 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR2 | H'FFA0 0024 | H'1FA0 0024 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR2 | H'FFA0 0028 | H'1FA0 0028 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR2 | H'FFA0 002C | H'1FA0 002C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | SAR3 | H'FFA0 0030 | H'1FA0 0030 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR3 | H'FFA0 0034 | H'1FA0 0034 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR3 | H'FFA0 0038 | H'1FA0 0038 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR3 | H'FFA0 003C | H'1FA0 003C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | DMAOR | H'FFA0 0040 | H'1FA0 0040 | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |

| Module | Register | P4 Address | Area 7 Address* ¹ | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|----------|-------------|---------------------------------|------------------------|-------------------|-----------------|-------|--------------|------------------------------|
| DMAC | SAR4 | H'FFA0 0050 | H'1FA0 0050 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR4 | H'FFA0 0054 | H'1FA0 0054 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR4 | H'FFA0 0058 | H'1FA0 0058 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR4 | H'FFA0 005C | H'1FA0 005C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | SAR5 | H'FFA0 0060 | H'1FA0 0060 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR5 | H'FFA0 0064 | H'1FA0 0064 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR5 | H'FFA0 0068 | H'1FA0 0068 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR5 | H'FFA0 006C | H'1FA0 006C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | SAR6 | H'FFA0 0070 | H'1FA0 0070 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR6 | H'FFA0 0074 | H'1FA0 0074 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR6 | H'FFA0 0078 | H'1FA0 0078 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR6 | H'FFA0 007C | H'1FA0 007C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| DMAC | SAR7 | H'FFA0 0080 | H'1FA0 0080 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DAR7 | H'FFA0 0084 | H'1FA0 0084 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | DMATCR7 | H'FFA0 0088 | H'1FA0 0088 | 32 | Undefined | Undefined | Held | Held | Bck |
| DMAC | CHCR7 | H'FFA0 008C | H'1FA0 008C | 32 | H'0000 0000 | H'0000 0000 | Held | Held | Bck |
| CPG | FRQCR | H'FFC0 0000 | H'1FC0 0000 | 16 | * ² | Held | Held | Held | Pck |
| CPG | STBCR | H'FFC0 0004 | H'1FC0 0004 | 8 | H'00 | Held | Held | Held | Pck |
| CPG | WTCNT | H'FFC0 0008 | H'1FC0 0008 | 8/16 * ³ | H'00 | Held | Held | Held | Pck |
| CPG | WTCSR | H'FFC0 000C | H'1FC0 000C | 8/16 * ³ | H'00 | Held | Held | Held | Pck |
| CPG | STBCR2 | H'FFC0 0010 | H'1FC0 0010 | 8 | H'00 | Held | Held | Held | Pck |
| RTC | R64CNT | H'FFC8 0000 | H'1FC8 0000 | 8 | Held | Held | Held | Held | Pck |
| RTC | RSECCNT | H'FFC8 0004 | H'1FC8 0004 | 8 | Held | Held | Held | Held | Pck |
| RTC | RMINCNT | H'FFC8 0008 | H'1FC8 0008 | 8 | Held | Held | Held | Held | Pck |
| RTC | RHRCNT | H'FFC8 000C | H'1FC8 000C | 8 | Held | Held | Held | Held | Pck |
| RTC | RWKCNT | H'FFC8 0010 | H'1FC8 0010 | 8 | Held | Held | Held | Held | Pck |
| RTC | RDAYCNT | H'FFC8 0014 | H'1FC8 0014 | 8 | Held | Held | Held | Held | Pck |
| RTC | RMONCNT | H'FFC8 0018 | H'1FC8 0018 | 8 | Held | Held | Held | Held | Pck |
| RTC | RYRCNT | H'FFC8 001C | H'1FC8 001C | 16 | Held | Held | Held | Held | Pck |

| Module | Register | P4 Address | Area 7 Address* ¹ | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|----------|-------------|---------------------------------|------|----------------------|----------------------|-------|--------------------|------------------------------|
| RTC | RSECAR | H'FFC8 0020 | H'1FC8 0020 | 8 | Held* ² | Held | Held | Held | Pck |
| RTC | RMINAR | H'FFC8 0024 | H'1FC8 0024 | 8 | Held* ² | Held | Held | Held | Pck |
| RTC | RHRAR | H'FFC8 0028 | H'1FC8 0028 | 8 | Held* ² | Held | Held | Held | Pck |
| RTC | RWKAR | H'FFC8 002C | H'1FC8 002C | 8 | Held* ² | Held | Held | Held | Pck |
| RTC | RDAYAR | H'FFC8 0030 | H'1FC8 0030 | 8 | Held* ² | Held | Held | Held | Pck |
| RTC | RMONAR | H'FFC8 0034 | H'1FC8 0034 | 8 | Held* ² | Held | Held | Held | Pck |
| RTC | RCR1 | H'FFC8 0038 | H'1FC8 0038 | 8 | H'00* ² | H'00* ² | Held | Held | Pck |
| RTC | RCR2 | H'FFC8 003C | H'1FC8 003C | 8 | H'09* ² | H'00* ² | Held | Held | Pck |
| RTC | RCR3 | H'FFC8 0050 | H'1FC8 0050 | 8 | H'00 | Held | Held | Held | Pck |
| RTC | RYRAR | H'FFC8 0054 | H'1FC8 0054 | 16 | Undefined | Held | Held | Held | Pck |
| INTC | ICR | H'FFD0 0000 | H'1FD0 0000 | 16 | H'0000* ² | H'0000* ² | Held | Held | Pck |
| INTC | IPRA | H'FFD0 0004 | H'1FD0 0004 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| INTC | IPRB | H'FFD0 0008 | H'1FD0 0008 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| INTC | IPRC | H'FFD0 000C | H'1FD0 000C | 16 | H'0000 | H'0000 | Held | Held | Pck |
| INTC | IPRD | H'FFD0 0010 | H'1FD0 0010 | 16 | H'DA74 | H'DA74 | Held | Held | Pck |
| TMU | TOCR | H'FFD8 0000 | H'1FD8 0000 | 8 | H'00 | H'00 | Held | Held | Pck |
| TMU | TSTR | H'FFD8 0004 | H'1FD8 0004 | 8 | H'00 | H'00 | Held | H'00* ² | Pck |
| TMU | TCOR0 | H'FFD8 0008 | H'1FD8 0008 | 32 | H'FFFF FFFF | H'FFFF FFFF | Held | Held | Pck |
| TMU | TCNT0 | H'FFD8 000C | H'1FD8 000C | 32 | H'FFFF FFFF | H'FFFF FFFF | Held | Held | Pck |
| TMU | TCR0 | H'FFD8 0010 | H'1FD8 0010 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| TMU | TCOR1 | H'FFD8 0014 | H'1FD8 0014 | 32 | H'FFFF FFFF | H'FFFF FFFF | Held | Held | Pck |
| TMU | TCNT1 | H'FFD8 0018 | H'1FD8 0018 | 32 | H'FFFF FFFF | H'FFFF FFFF | Held | Held | Pck |
| TMU | TCR1 | H'FFD8 001C | H'1FD8 001C | 16 | H'0000 | H'0000 | Held | Held | Pck |
| TMU | TCOR2 | H'FFD8 0020 | H'1FD8 0020 | 32 | H'FFFF FFFF | H'FFFF FFFF | Held | Held | Pck |
| TMU | TCNT2 | H'FFD8 0024 | H'1FD8 0024 | 32 | H'FFFF FFFF | H'FFFF FFFF | Held | Held | Pck |
| TMU | TCR2 | H'FFD8 0028 | H'1FD8 0028 | 16 | H'0000 | H'0000 | Held | Held | Pck |

| Module | Register | P4 Address | Area 7 Address* ¹ | Size | Power-On Reset | Manual Reset | Sleep | Stand- by | Synchro nization Clock |
|--------|----------|-------------|---------------------------------|------|----------------------|----------------------|-------|--------------------|------------------------------|
| TMU | TCPR2 | H'FFD8 002C | H'1FD8 002C | 32 | Held | Held | Held | Held | Pck |
| SCI | SCSMR1 | H'FFE0 0000 | H'1FE0 0000 | 8 | H'00 | H'00 | Held | H'00 | Pck |
| SCI | SCBRR1 | H'FFE0 0004 | H'1FE0 0004 | 8 | H'FF | H'FF | Held | H'FF | Pck |
| SCI | SCSCR1 | H'FFE0 0008 | H'1FE0 0008 | 8 | H'00 | H'00 | Held | H'00 | Pck |
| SCI | SCTDR1 | H'FFE0 000C | H'1FE0 000C | 8 | H'FF | H'FF | Held | H'FF | Pck |
| SCI | SCSSR1 | H'FFE0 0010 | H'1FE0 0010 | 8 | H'84 | H'84 | Held | H'84 | Pck |
| SCI | SCRDR1 | H'FFE0 0014 | H'1FE0 0014 | 8 | H'00 | H'00 | Held | H'00 | Pck |
| SCI | SCSCMR1 | H'FFE0 0018 | H'1FE0 0018 | 8 | H'00 | H'00 | Held | H'00 | Pck |
| SCI | SCSPTR1 | H'FFE0 001C | H'1FE0 001C | 8 | H'00* ² | H'00* ² | Held | H'00* ² | Pck |
| SCIF | SCSMR2 | H'FFE8 0000 | H'1FE8 0000 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| SCIF | SCBRR2 | H'FFE8 0004 | H'1FE8 0004 | 8 | H'FF | H'FF | Held | Held | Pck |
| SCIF | SCSCR2 | H'FFE8 0008 | H'1FE8 0008 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| SCIF | SCFTDR2 | H'FFE8 000C | H'1FE8 000C | 8 | Undefined | Undefined | Held | Held | Pck |
| SCIF | SCFSR2 | H'FFE8 0010 | H'1FE8 0010 | 16 | H'0060 | H'0060 | Held | Held | Pck |
| SCIF | SCFRDR2 | H'FFE8 0014 | H'1FE8 0014 | 8 | Undefined | Undefined | Held | Held | Pck |
| SCIF | SCFCR2 | H'FFE8 0018 | H'1FE8 0018 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| SCIF | SCFDR2 | H'FFE8 001C | H'1FE8 001C | 16 | H'0000 | H'0000 | Held | Held | Pck |
| SCIF | SCSPTR2 | H'FFE8 0020 | H'1FE8 0020 | 16 | H'0000* ² | H'0000* ² | Held | Held | Pck |
| SCIF | SCLSR2 | H'FFE8 0024 | H'1FE8 0024 | 16 | H'0000 | H'0000 | Held | Held | Pck |
| H-UDI | SDIR | H'FFF0 0000 | H'1FF0 0000 | 16 | H'FFFF* ² | Held | Held | Held | Pck |
| H-UDI | SDDR | H'FFF0 0008 | H'1FF0 0008 | 32 | Held | Held | Held | Held | Pck |
| Hi-UDI | SDINT | H'FFF0 0014 | H'1FF0 0014 | 16 | H'0000 | Held | Held | Held | Pck |

- Notes:
1. With control registers, the above addresses in the physical page number field can be accessed by means of a TLB setting. When these addresses are set directly without using the TLB, operations are limited.
 2. Includes undefined bits. See the descriptions of the individual modules.
 3. Use word-size access when writing. Perform the write with the upper byte set to H'5A or H'A5, respectively. Byte- and longword-size writes cannot be used.
Use byte-size access when reading.

Appendix B Package Dimensions

The package dimension that is shown in the Renesas Semiconductor Package Data Book has priority.

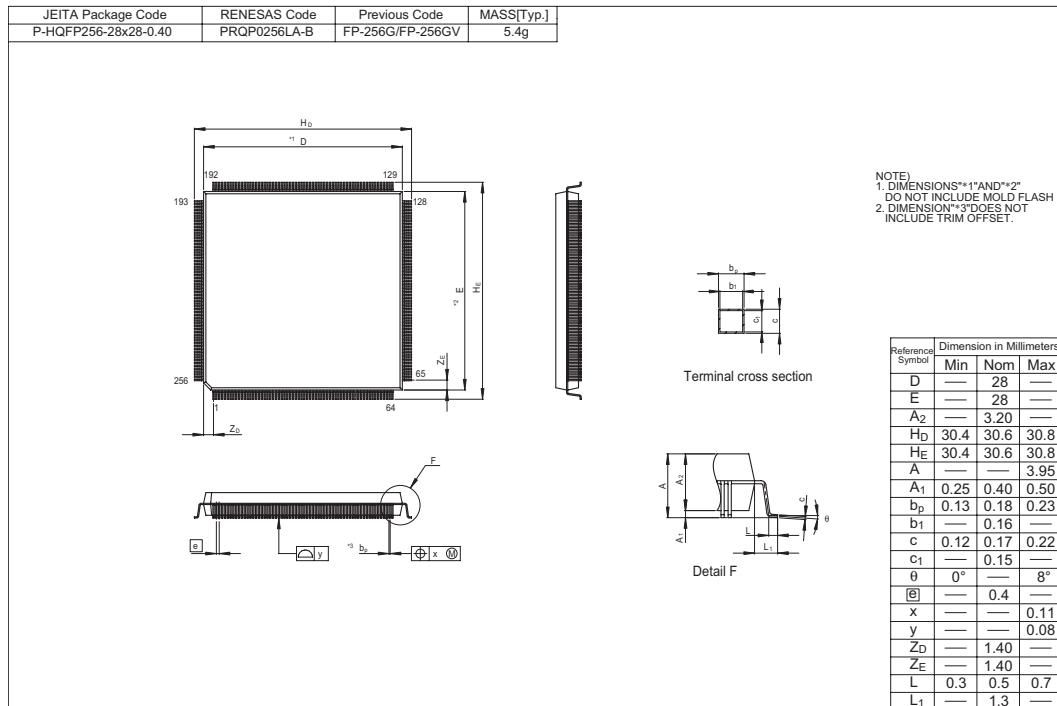


Figure B.1 Package Dimensions (256-pin QFP)

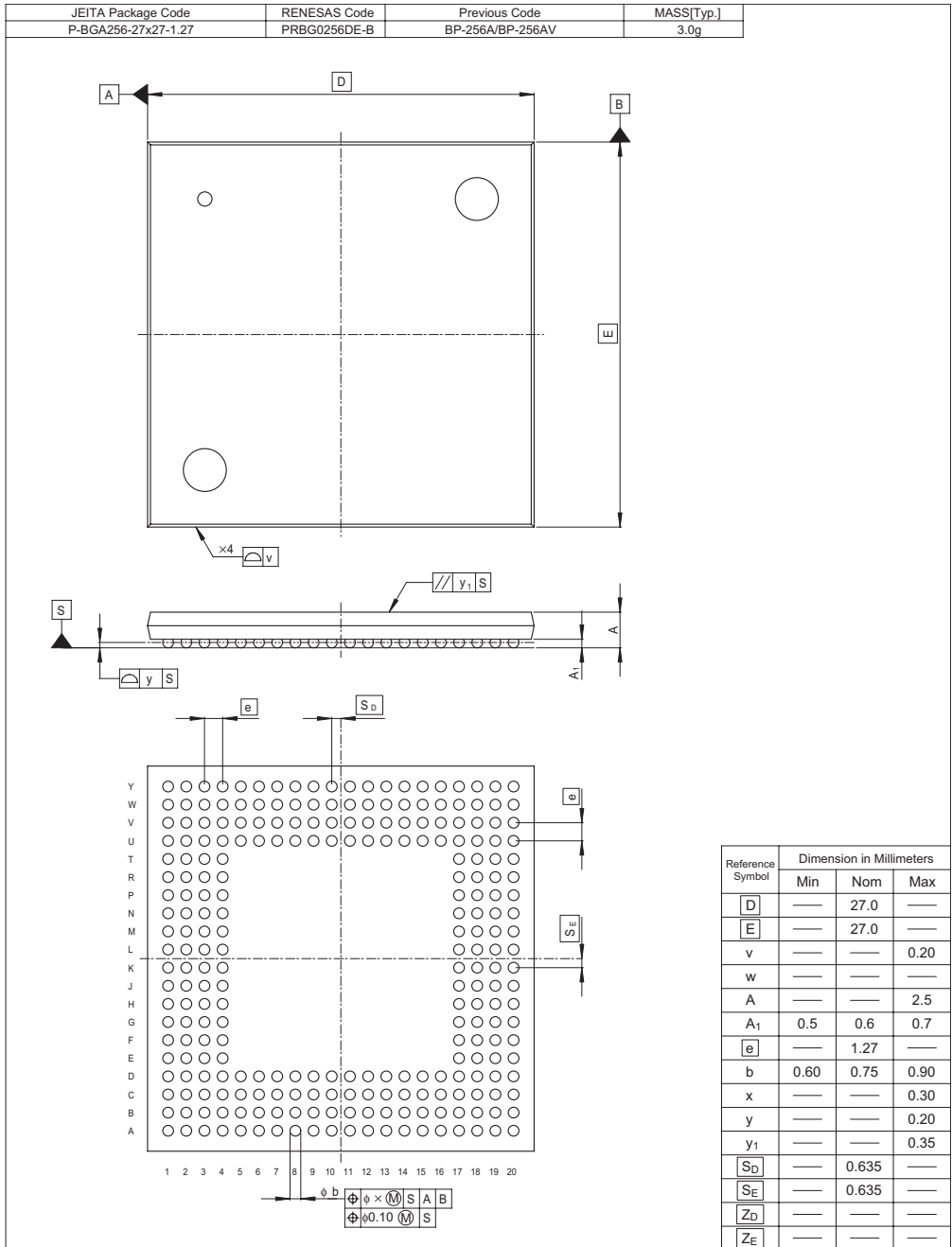


Figure B.2 Package Dimensions
(256-pin BGA: Devices Other than HD6417751RBA240HV)

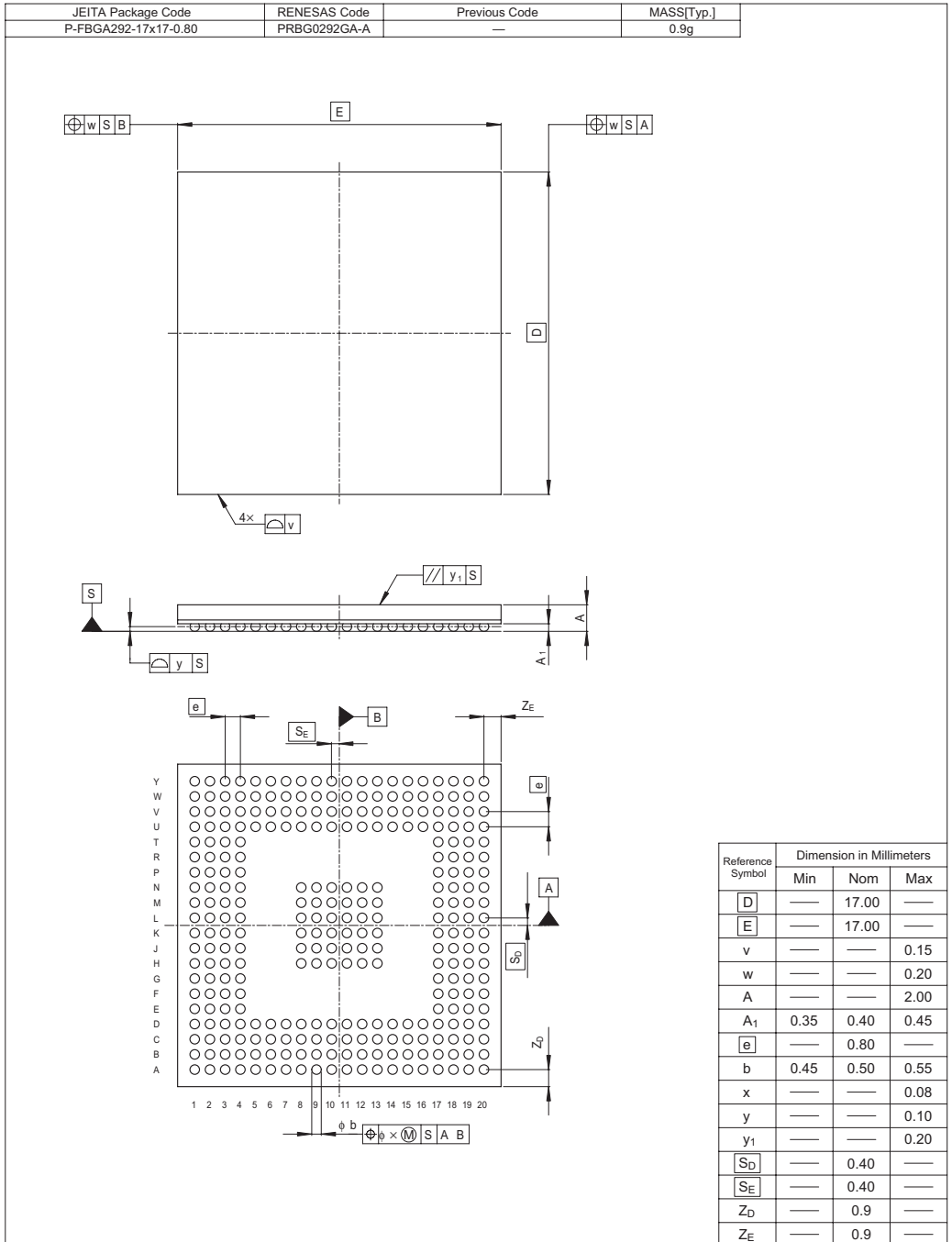


Figure B.3 Package Dimensions (292-pin BGA)

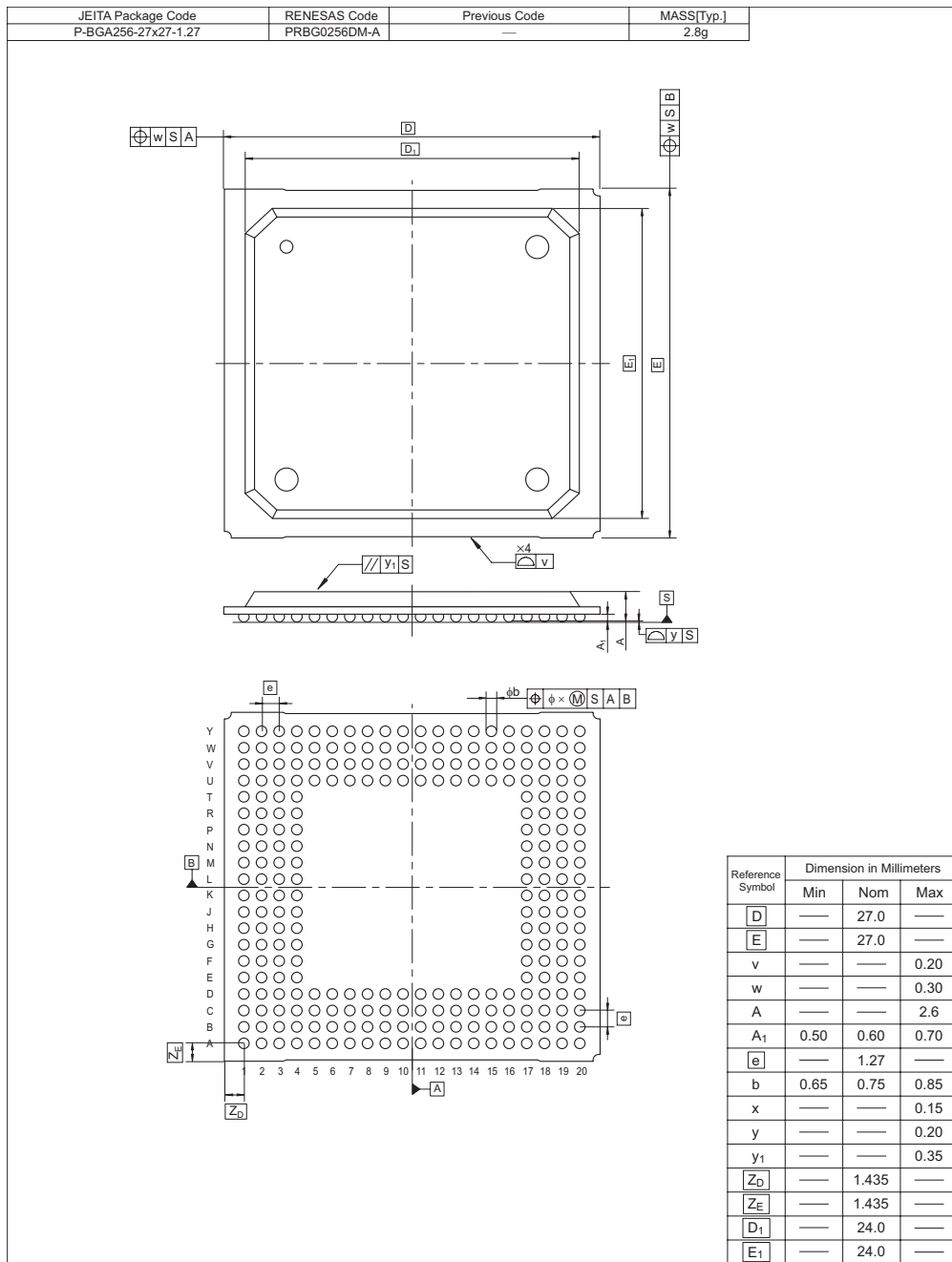


Figure B.4 Package Dimensions
(256-pin BGA: HD6417751RBA240HV)

Appendix C Mode Pin Settings

The MD10–MD0 pin values are input in the event of a power-on reset via the $\overline{\text{RESET}}$ pin.

Clock Modes

Table C.1 Clock Operating Modes (SH7751)

| Clock Operating Mode | External Pin Combination | | | 1/2 Frequency Divider | PLL1 | PLL2 | Frequency (vs. Input Clock) | | | |
|----------------------|--------------------------|-----|-----|-----------------------|------|------|-----------------------------|-----------|-------------------------|---------------------|
| | MD2 | MD1 | MD0 | | | | CPU Clock | Bus Clock | Peripheral Module Clock | FRQCR Initial Value |
| 0 | 0 | 0 | 0 | Off | On | On | 6 | 3/2 | 3/2 | H'0E1A |
| 1 | | | 1 | Off | On | On | 6 | 1 | 1 | H'0E23 |
| 2 | | 1 | 0 | On | On | On | 3 | 1 | 1/2 | H'0E13 |
| 3 | | | 1 | Off | On | On | 6 | 2 | 1 | H'0E13 |
| 4 | 1 | 0 | 0 | On | On | On | 3 | 3/2 | 3/4 | H'0E0A |
| 5 | | | 1 | Off | On | On | 6 | 3 | 3/2 | H'0E0A |
| 6 | | 1 | 0 | Off | Off | Off | 1 | 1/2 | 1/2 | H'0808 |

- Notes:
1. The multiplication factor of 1/2 frequency divider is solely determined by the clock operating mode.
 2. For the ranges input clock frequency, see the description of the EXTAL clock input frequency (f_{EX}) and the CKIO clock output (f_{OP}) in section 23.3.1, Clock and Control Signal Timing.

Table C.2 Clock Operating Modes (SH7751R)

| Clock Operating Mode | External Pin Combination | | | | | Frequency (vs. Input Clock) | | | |
|----------------------|--------------------------|-----|-----|----------|------|-----------------------------|-----------|-------------------------|---------------------|
| | MD2 | MD1 | MD0 | PLL1 | PLL2 | CPU Clock | Bus Clock | Peripheral Module Clock | FRQCR Initial Value |
| 0 | 0 | 0 | 0 | On (×12) | On | 12 | 3 | 3 | H'0E1A |
| 1 | | | 1 | On (×12) | On | 12 | 3/2 | 3/2 | H'0E2C |
| 2 | | 1 | 0 | On (×6) | On | 6 | 2 | 1 | H'0E13 |
| 3 | | | 1 | On (×12) | On | 12 | 4 | 2 | H'0E13 |
| 4 | 1 | 0 | 0 | On (×6) | On | 6 | 3 | 3/2 | H'0E0A |
| 5 | | | 1 | On (×12) | On | 12 | 6 | 3 | H'0E0A |
| 6 | | 1 | 0 | OFF (×6) | OFF | 1 | 1/2 | 1/2 | H'0808 |

- Notes: 1. The multiplication factor of PLL1 is solely determined by the clock operating mode.
 2. For the ranges input clock frequency, see the description of the EXTAL clock input frequency (f_{EX}) and the CKIO clock output (f_{OP}) in section 23.3.1, Clock and Control Signal Timing.

Table C.3 Area 0 Memory Map and Bus Width

| Pin Value | | | Memory Type | Bus Width |
|-----------|-----|-----|---------------------------|---------------------------|
| MD6 | MD4 | MD3 | | |
| 0 | 0 | 0 | Reserved (Cannot be used) | Reserved (Cannot be used) |
| | | 1 | Reserved (Cannot be used) | Reserved (Cannot be used) |
| | 1 | 0 | Reserved (Cannot be used) | Reserved (Cannot be used) |
| | | 1 | MPX interface | 32 bits |
| 1 | 0 | 0 | Reserved (Cannot be used) | Reserved (Cannot be used) |
| | | 1 | SRAM interface | 8 bits |
| | 1 | 0 | SRAM interface | 16 bits |
| | | 1 | SRAM interface | 32 bits |

Table C.4 Endian

| Pin Value | |
|-----------|---------------|
| MD5 | Endian |
| 0 | Big endian |
| 1 | Little endian |

Table C.5 Master/Slave

| Pin Value | |
|-----------|--------------|
| MD7 | Master/Slave |
| 0 | Slave |
| 1 | Master |

Table C.6 Clock Input

| Pin Value | |
|-----------|----------------------|
| MD8 | Clock Input |
| 0 | External input clock |
| 1 | Crystal resonator |

Table C.7 PCI Mode

| Mode | Pin Value | | Mode |
|------|-----------|-----|--|
| | MD10 | MD9 | |
| 0 | 0 | 0 | PCI host with external clock input |
| 1 | 0 | 1 | PCI host with feedback input clock from CKIO |
| 2 | 1 | 0 | PCI non-host with external clock input |
| 3 | 1 | 1 | PCI disabled |

Note: When exiting standby mode or hardware standby mode using a power-on reset, do not change the PCI mode.

Appendix D Pin Functions

D.1 Pin States

**Table D.1 Pin States in Reset, Power-Down State, and Bus-Released State
(PCI Enable, Disable Common)**

| Pin Name | I/O | Reset (Power-On) | | Reset (Manual) | | Standby | Bus Released | Hard- ware Standby | Notes |
|--------------------------|-----|---------------------|------------------|----------------------------------|------------------|----------------------------------|----------------------------------|--------------------------|-----------------|
| | | Master | Slave | Master | Slave | | | | |
| D0–D31 | I/O | Z | Z | Z* ¹⁴ | Z* ¹⁴ | Z* ¹⁴ | Z* ¹⁴ | Z | |
| A2–A17, A0–A25 | O | Z | Z | Z* ¹³ O* ⁷ | Z* ¹³ | Z* ¹³ O* ⁵ | Z* ¹³ | Z | |
| RESET | I | I | I | I | I | I | I | I | |
| BACK/BSREQ | O | H | H | H | H | H | O | Z | |
| BREQ/BSACK | I | PI | PI | I* ¹² | I* ¹² | I* ¹² | I | I | |
| BS | O | H | PZ | H | Z* ¹³ | Z* ¹³ H* ⁵ | Z* ¹³ | Z | |
| CKE | O | H | H | O* ⁴ | H | L | O* ⁴ | Z | |
| CS6–CS0 | O | H | PZ | H | Z* ¹³ | Z* ¹³ H* ⁵ | Z* ¹³ | Z | |
| RAS | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| RD/CASS/FRAME | O | H | PZ | O* ⁴ | Z | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| RD/ \overline{WR} | O | H | PZ | H | Z* ¹³ | Z* ¹³ H* ⁵ | Z* ¹³ | Z | |
| RDY | I | PI | PI | I* ¹² | I* ¹² | I* ¹² | I* ¹² | I | |
| CAS3/DQM3 | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| CAS2/DQM2 | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| CAS1/DQM1 | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| CAS0/DQM0 | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| WE3/IOICWR | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| WE2/IOICRD | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| WE1 | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| WE0/REG | O | H | PZ | O* ⁴ | Z* ¹³ | Z* ¹³ O* ³ | Z* ¹³ O* ³ | Z | |
| DACK1–DACK0 | O | L | L | L | L | Z* ¹¹ O* ⁶ | O | Z | DMAC |
| MD7/CTS2 | I/O | I* ¹⁷ | I* ¹⁷ | I* ¹¹ | I* ¹¹ | I* ¹¹ O* ⁶ | I* ¹¹ O | Z | SCIF |
| MD6/ $\overline{IOIS16}$ | I | I* ¹⁷ | I* ¹⁷ | I* ¹² | I* ¹² | I* ¹² | I* ¹² | I | PCMClA (I/O) |
| MD5 | I | I* ¹⁷ | I* ¹⁷ | Z* ¹³ | Z* ¹³ | Z* ¹³ | Z* ¹³ | Z | |

| Pin Name | I/O | Reset (Power-On) | | Reset (Manual) | | Standby | Bus Released | Hard- ware Standby | Notes |
|---------------------|-------------------|---------------------|------------------|--------------------|--------------------|---|--------------------|--------------------------|--------|
| | | Master | Slave | Master | Slave | | | | |
| MD4/CE2B | I/O* ¹ | I* ¹⁷ | I* ¹⁷ | Z* ¹³ H | Z* ¹³ | Z* ¹³ H* ⁵ | Z* ¹³ | Z | PCMCIA |
| MD3/CE2A | I/O* ² | I* ¹⁷ | I* ¹⁷ | Z* ¹³ H | Z* ¹³ | Z* ¹³ H* ⁵ | Z* ¹³ | Z | PCMCIA |
| CKIO | O | O | O | ZO* ⁸ | ZO* ⁸ | ZO* ⁸ | ZO* ⁸ | Z | |
| STATUS1– STATUS0 | O | O | O | O | O | O | O | ZO* ⁹ | |
| IRL3–IRL0 | I | PI | PI | I* ¹² | I* ¹² | I* ¹² | I* ¹² | I | INTC |
| NMI | I | PI | PI | I* ¹² | I* ¹² | I* ¹² | I* ¹² | I | INTC |
| DREQ1–DREQ0 | I | PI | PI | I* ¹¹ | I* ¹¹ | I* ¹¹ | I* ¹¹ | I | DMAC |
| DRAK1–DRAK0 | O | L | L | L | L | Z* ¹¹ O* ⁶ | O | Z | DMAC |
| MD0/SCK2 | I/O | I* ¹⁷ | I* ¹⁷ | I* ¹¹ | I* ¹¹ | I* ¹¹ Z* ¹¹ O* ⁶ | I* ¹¹ O | I | SCIF |
| RXD | I | PI | PI | I* ¹¹ | I* ¹¹ | I* ¹¹ | I* ¹¹ | I | SCI |
| SCK | I/O | PI | PI | I* ¹¹ | I* ¹¹ | I* ¹¹ Z* ¹¹ O* ⁶ | I* ¹¹ O | Z | SCI |
| MD1/TXD2 | I/O | I* ¹⁷ | I* ¹⁷ | Z* ¹¹ | Z* ¹¹ | Z* ¹¹ O* ⁶ | Z* ¹¹ O | Z | SCIF |
| MD2/RXD2 | I | I* ¹⁷ | I* ¹⁷ | I* ¹¹ | I* ¹¹ | I* ¹¹ | I* ¹¹ | I | SCIF |
| TxD | I/O | PI | PI | Z* ¹¹ O | Z* ¹¹ O | Z* ¹¹ O* ⁶ | O | Z | SCI |
| MD8/RTS2 | I/O | I* ¹⁷ | I* ¹⁷ | I* ¹¹ | I* ¹¹ | I* ¹¹ Z* ¹¹ O* ⁶ | I* ¹¹ O | Z | SCIF |
| TCLK | I/O | PI | PI | I* ¹¹ | I* ¹¹ | I* ¹¹ O | I* ¹¹ O | Z | TMU |
| TDO | O | O | O | O | O | O | O | Z | H-UDI |
| TMS | I | PI | PI | PI | PI | PI | PI | I | H-UDI |
| TCK | I | PI | PI | PI | PI | PI | PI | I | H-UDI |
| TDI | I | PI | PI | PI | PI | PI | PI | I | H-UDI |
| TRST | I | PI | PI | PI | PI | PI | PI | I | H-UDI |
| MRESET | I | PI | PI | PI | PI | PI | PI | I | |
| SLEEP | I | PI | PI | I* ¹² | I* ¹² | I* ¹² | I* ¹² | I | |
| CA | I | I | I | I | I | I | I | I | |

Table D.2 Pin States in Reset, Power-Down State, and Bus-Released State (PCI Enable)

| Pin Name | I/O | Reset (Power On) | | Reset (Manual) | | Standby | | Reset (Software) | | Hard- ware Standby | Notes |
|-----------------|-----|---------------------|------------------|--------------------------------|--|------------------|--|---------------------|--------------------------------|--------------------------|--|
| | | Host | Non- Host | Host | Non- Host | Host | Non- Host | Host | Non- Host | | |
| AD31–AD31 | I/O | L | Z | IOZ | IOZ | K | Z | L | Z | Z | |
| CBE3–CBE0 | I/O | L | Z | IOZ | IOZ | K | Z | L | Z | Z | |
| PAR | I/O | L | Z | IOZ | IOZ | K | Z | L | Z | Z | |
| SERR | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| PERR | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| PCILOCK | I/O | PZ | PZ | IZ* ¹⁰ | IZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| PCISTOP | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| DEVSEL | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| TRDY | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| IRDY | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| PCIFRAME | I/O | PZ | PZ | IOZ* ¹ ₀ | IOZ* ¹⁰ | Z* ¹⁰ | Z* ¹⁰ | PZ | PZ | Z | |
| PCIREQ4 | I/O | PI | PZ | Z* ¹⁰ | Z* ¹⁰ (IO* ^{11,*16}) | I* ¹⁰ | Z* ¹⁰ (IO* ^{10,*16}) | PI | PZ (IO* ^{10,*16}) | Z | Values in paren- thesis are when using PORT |
| PCIREQ2/ MD9 | I/O | I* ¹⁷ | I* ¹⁷ | Z* ¹⁰ | Z* ¹⁰ (IO* ^{11,*16}) | I* ¹⁰ | Z* ¹⁰ (IO* ^{10,*16}) | PI | PZ (IO* ^{10,*16}) | Z | Values in paren- thesis are when using PORT |

| Pin Name | I/O | Reset (Power On) | | Reset (Manual) | | Standby | | Reset (Software) | | Hard- ware Standby | Notes |
|---------------------|-----|---------------------|------------------|-----------------------|--|-----------------------|--|---------------------|--------------------------------|--------------------------|--|
| | | Host | Non- Host | Host | Non- Host | Host | Non- Host | Host | Non- Host | | |
| PCIREQ3/ MD10 | I/O | I ^{*17} | I ^{*17} | Z ^{*10} | Z ^{*10} (IO ^{*11,*16}) | I ^{*10} | Z ^{*10} (IO ^{*10,*16}) | PI | PZ (IO ^{*10,*16}) | Z | Values in paren- thesis are when using PORT |
| PCIREQ1/ GNTIN | I | PI | PI | I ^{*10} | I ^{*10} | I ^{*10} | I ^{*10} | PI | PI | Z | |
| PCIGNT4- PCIGNT2 | O | Z | Z | O | Z (K) | K | Z (K) | Z | Z (K) | Z | Values in paren- thesis are when using PORT |
| PCIGNT1/ REQOUT | O | Z | Z | O | O | K | K | Z | H | Z | |
| PCICLK | I | I | I | I | I | I | I | I | I | Z | |
| PCIRST | O | L | L | K | K | K | K | L | L | Z | |
| IDSEL | I | PI | I | PI | I | PI | I | PI | I | Z | |
| INTA | O | PZ | PZ | ODK _{*10} | ODK _{*10} | ODK _{*10} | ODK _{*10} | PZ | PZ | Z | |

Table D.3 Pin States in Reset, Power-Down State, and Bus-Released State (PCI Disable)

| Pin Name | I/O | Reset (Power-On) | | Reset (Manual) | | Standby | Bus Released | Hard- ware Standby | Notes |
|-----------------|-----|---------------------|------------------|-------------------|----------|-------------------------|-------------------------|--------------------------|---|
| | | Master | Slave | Master | Slave | | | | |
| AD31–AD0 | I/O | Z | Z | Z (K) | Z (K) | Z* ¹⁵ (K) | Z* ¹⁵ (K) | Z | Values in parenthesis are when using PORT |
| CBE3–CBE0 | — | Z | Z | Z | Z | Z | Z | Z | |
| PAR | O | Z | Z | Z | Z | Z | Z | Z | |
| SERR | — | Z | Z | Z | Z | Z | Z | Z | |
| PERR | — | Z | Z | Z | Z | Z | Z | Z | |
| PCLOCK | — | Z | Z | Z | Z | Z | Z | Z | |
| PCISTOP | — | Z | Z | Z | Z | Z | Z | Z | |
| DEVSEL | — | Z | Z | Z | Z | Z | Z | Z | |
| TRDY | — | Z | Z | Z | Z | Z | Z | Z | |
| IRDY | — | Z | Z | Z | Z | Z | Z | Z | |
| PCIFRAME | — | Z | Z | Z | Z | Z | Z | Z | |
| PCIREQ4 | — | Z | Z | Z | Z | Z | Z | Z | |
| PCIREQ2/MD9 | I/O | I* ¹⁷ | I* ¹⁷ | Z | Z | Z | Z | Z | |
| PCIREQ3/MD10 | I/O | I* ¹⁷ | I* ¹⁷ | Z | Z | Z | Z | Z | |
| PCIREQ1 | — | Z | Z | Z | Z | Z | Z | Z | |
| PCIGNT4–PCIGNT2 | O | Z | Z | Z | Z | Z | Z | Z | |
| PCIGNT1 | O | Z | Z | Z | Z | Z | Z | Z | |
| PCICLK | — | Z | Z | Z | Z | Z | Z | Z | |
| PCIRST | O | Z | Z | Z | Z | Z | Z | Z | |
| IDSEL | — | Z | Z | Z | Z | Z | Z | Z | |
| INTA | — | Z | Z | Z | Z | Z | Z | Z | |

Legend:

I: Input

O: Output

H: High-level output

L: Low-level output
 Z: High-impedance
 K: Output state held
 IZ/IOZ: Response to access from PCI
 PZ: Pulled up with a built-in pull-up resistance
 PI: Input pulled up with a built-in pull-up resistance
 ODK: Open-drain output state held

- Notes:
1. Output when area 5 PCMCIA is used.
 2. Output when area 6 PCMCIA is used.
 3. Z (I) or O (refresh), depending on register setting (BCR1.HIZCNT).
 4. Depends on refresh operation.
 5. Z (I) or H (state held), depending on register setting (BCR1.HIZMEM).
 6. Z or O, depending on register setting (STBCR.PHZ).
 7. Output when refreshing is set.
 8. Z or O, depending on register setting (FRQCR.CKOEN).
 9. Z or O, depending on register setting (STBCR.STHZ).
 10. Pullup, depending on register setting (PCICR.PCIPUP).
 11. Pullup, depending on register setting (STBCR.PPU).
 12. Pullup, depending on register setting (BCR1.IPUP).
 13. Pullup, depending on register setting (BCR1.OPUP).
 14. Pullup, depending on register setting (BCR1.DPUP).
 15. Pullup, depending on register setting (BCR2.PORTEN).
 16. Pullup, depending on register setting (PCIPCTR.PB2PUP to PCIPCTR.PB4PUP).
 17. Pullup by on-chip pullup resistor. Note that this cannot be used for pullup of the mode pin during a power-on reset. Pullup or pulldown should be performed externally to this LSI.

D.2 Handling of Unused Pins

- When RTC is not used
 - EXTAL2: Pull up to 3.3 V
 - XTAL2: Leave unconnected
 - VDD-RTC: Power supply
 - VSS-RTC: Power supply
- When PLL1 is not used
 - VDD-PLL1: Power supply
 - VSS-PLL1: Power supply

- When PLL2 is not used
 - VDD-PLL2: Power supply
 - VSS-PLL2: Power supply
- When on-chip crystal oscillator is not used
 - XTAL: Leave unconnected
 - VDD-CPG: Power supply
 - VSS-CPG: Power supply

Table D.4 Handling of Pins When PCI Is Not Used

| Pin Name | I/O | Handling |
|---|-----|---|
| AD31–AD0 | I/O | Pull up to 3.3 V* |
| $\overline{\text{CBE3}}\text{--}\overline{\text{CBE0}}$ | I/O | Pull up to 3.3 V |
| PAR | I/O | Pull up to 3.3 V |
| $\overline{\text{SERR}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{PERR}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{PCILOCK}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{PCISTOP}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{DEVSEL}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{TRDY}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{IRDY}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{PCIFRAME}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{PCIREQ4}}\text{--}\overline{\text{PCIREQ2}}$ | I/O | Pull up to 3.3 V |
| $\overline{\text{PCIREQ1}}$ | I | Pull up to 3.3 V |
| $\overline{\text{PCIGNT4}}\text{--}\overline{\text{PCIGNT2}}$ | O | Pull up to 3.3 V |
| $\overline{\text{PCIGNT1}}$ | O | Pull up to 3.3 V |
| $\overline{\text{PCICLK}}$ | I | Pull up to 3.3 V |
| $\overline{\text{PCIRST}}$ | O | Leave unconnected |
| IDSEL | I | Pull down to low level when IDSEL is not in use |
| $\overline{\text{INTA}}$ | O | Leave unconnected |

Note: * When not used as a general-purpose I/O port.

D.3 Note on Pin Processing

To prevent unwanted effects on other pins when using external pull-up or pull-down resistors, use independent pull-up or pull-down resistors for individual pins.

Appendix E Synchronous DRAM Address Multiplexing Tables

(1) **BUS 32** (16M: 512k × 16b × 2) × 2 *
AMX 0 **AMXEXT 0** **16M, column-addr-8bit** **4MB**

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|-------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A14 | | | | |
| A13 | A21 | A21 | A11 | BANK selects bank address |
| A12 | A20 | H/L | A10 | Address precharge setting |
| A11 | A19 | 0 | A9 | Address |
| A10 | A18 | 0 | A8 | |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(2) **BUS 32** (16M: 512k × 16b × 2) × 2 *
AMX 0 **AMXEXT 1** **16M, column-addr-8bit** **4MB**

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A14 | | | | |
| A13 | A20 | A20 | A11 | BANK selects bank address |
| A12 | A21 | H/L | A10 | Address precharge setting |
| A11 | A19 | 0 | A9 | Address |
| A10 | A18 | 0 | A8 | |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(3) **BUS 32** (16M: $1\text{M} \times 8\text{b} \times 2$) $\times 4$ *
AMX 1 **AMXEXT 0** **16M, column-addr-9bit** **8MB**

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A14 | | | | |
| A13 | A22 | A22 | A11 | BANK selects bank address |
| A12 | A21 | H/L | A10 | Address precharge setting |
| A11 | A20 | 0 | A9 | Address |
| A10 | A19 | A10 | A8 | |
| A9 | A18 | A9 | A7 | |
| A8 | A17 | A8 | A6 | |
| A7 | A16 | A7 | A5 | |
| A6 | A15 | A6 | A4 | |
| A5 | A14 | A5 | A3 | |
| A4 | A13 | A4 | A2 | |
| A3 | A12 | A3 | A1 | |
| A2 | A11 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(4) **BUS 32** (16M: 1M × 8b × 2) × 4 *
AMX 1 **AMXEXT 1** **16M, column-addr-9bit** **8MB**

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A14 | | | | |
| A13 | A21 | A21 | A11 | BANK selects bank address |
| A12 | A22 | H/L | A10 | Address precharge setting |
| A11 | A20 | 0 | A9 | Address |
| A10 | A19 | A10 | A8 | |
| A9 | A18 | A9 | A7 | |
| A8 | A17 | A8 | A6 | |
| A7 | A16 | A7 | A5 | |
| A6 | A15 | A6 | A4 | |
| A5 | A14 | A5 | A3 | |
| A4 | A13 | A4 | A2 | |
| A3 | A12 | A3 | A1 | |
| A2 | A11 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(5) **BUS 32** (64M: 1M × 16b × 4) × 2 *
AMX 2 64M, column-addr-8bit 16MB

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A16 | | | | |
| A15 | A23 | A23 | A13 | BANK selects bank address |
| A14 | A22 | A22 | A12 | |
| A13 | A21 | 0 | A11 | Address precharge setting |
| A12 | A20 | H/L | A10 | Address |
| A11 | A19 | 0 | A9 | |
| A10 | A18 | 0 | A8 | |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

- (6) **BUS 32** (64M: $2M \times 8b \times 4$) $\times 4$ *
AMX 3 64M, column-addr-9bit 32MB

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function | |
|-----|------------------|-----------|----------------------------------|---------------------------|--|
| | RAS Cycle | CAS Cycle | | | |
| A16 | | | | | |
| A15 | A24 | A24 | A13 | BANK selects bank address | |
| A14 | A23 | A23 | A12 | | |
| A13 | A22 | 0 | A11 | Address precharge setting | |
| A12 | A21 | H/L | A10 | Address | |
| A11 | A20 | 0 | A9 | | |
| A10 | A19 | A10 | A8 | | |
| A9 | A18 | A9 | A7 | | |
| A8 | A17 | A8 | A6 | | |
| A7 | A16 | A7 | A5 | | |
| A6 | A15 | A6 | A4 | | |
| A5 | A14 | A5 | A3 | | |
| A4 | A13 | A4 | A2 | | |
| A3 | A12 | A3 | A1 | | |
| A2 | A11 | A2 | A0 | | |
| A1 | Not used | | | | |
| A0 | Not used | | | | |

(7) **BUS 32** (64M: 512k × 32b × 4) × 1 ***AMX 4** 64M, column-addr-8bit 8MB

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A15 | | | | |
| A14 | A22 | A22 | A12 | BANK selects bank address |
| A13 | A21 | A21 | A11 | |
| A12 | A20 | H/L | A10 | Address precharge setting |
| A11 | A19 | 0 | A9 | Address |
| A10 | A18 | 0 | A8 | |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

- (8) **BUS 32** (64M: 1M × 32b × 2) × 1 *
AMX 5 64M, column-addr-8bit 8MB

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A15 | | | | |
| A14 | A22 | A22 | A12 | BANK selects bank address |
| A13 | A21 | 0 | A11 | |
| A12 | A20 | H/L | A10 | Address precharge setting |
| A11 | A19 | 0 | A9 | Address |
| A10 | A18 | 0 | A8 | |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(9) **BUS 32** (64M: $4M \times 4b \times 4$) $\times 8$ *
 (128M: $4M \times 8b \times 4$) $\times 4$ *
AMX 6 64M, column-addr-10bit

64MB

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A15 | A25 | A25 | A13 | BANK selects bank address |
| A14 | A24 | A24 | A12 | |
| A13 | A23 | 0 | A11 | |
| A12 | A22 | H/L | A10 | Address precharge setting |
| A11 | A21 | A11 | A9 | Address |
| A10 | A20 | A10 | A8 | |
| A9 | A19 | A9 | A7 | |
| A8 | A18 | A8 | A6 | |
| A7 | A17 | A7 | A5 | |
| A6 | A16 | A6 | A4 | |
| A5 | A15 | A5 | A3 | |
| A4 | A14 | A4 | A2 | |
| A3 | A13 | A3 | A1 | |
| A2 | A12 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(10) BUS 32 (256M: 4M × 16b × 4) × 2 *
AMX 6 AMXEXT1 256M, column-addr-9bit 64MB

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A16 | A25 | A25 | A14 | BANK selects bank address |
| A15 | A24 | A24 | A13 | |
| A14 | A23 | 0 | A12 | |
| A13 | A22 | 0 | A11 | |
| A12 | A21 | H/L | A10 | Address precharge setting |
| A11 | A20 | 0 | A9 | Address |
| A10 | A19 | A10 | A8 | |
| A9 | A18 | A9 | A7 | |
| A8 | A17 | A8 | A6 | |
| A7 | A16 | A7 | A5 | |
| A6 | A15 | A6 | A4 | |
| A5 | A14 | A5 | A3 | |
| A4 | A13 | A4 | A2 | |
| A3 | A12 | A3 | A1 | |
| A2 | A11 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

(11) BUS 32 (16M: 256k × 32b × 2) × 1 ***AMX 7 16M, column-addr-8bit 2MB**

| | LSI Address Pins | | Synchronous DRAM Address Pins | Function |
|-----|------------------|-----------|----------------------------------|---------------------------|
| | RAS Cycle | CAS Cycle | | |
| A13 | | | | |
| A12 | A20 | A20 | A10 | BANK selects bank address |
| A11 | A19 | H/L | A9 | Address precharge setting |
| A10 | A18 | 0 | A8 | Address |
| A9 | A17 | A9 | A7 | |
| A8 | A16 | A8 | A6 | |
| A7 | A15 | A7 | A5 | |
| A6 | A14 | A6 | A4 | |
| A5 | A13 | A5 | A3 | |
| A4 | A12 | A4 | A2 | |
| A3 | A11 | A3 | A1 | |
| A2 | A10 | A2 | A0 | |
| A1 | Not used | | | |
| A0 | Not used | | | |

Note: * Example configurations of synchronous DRAM

Appendix F Instruction Prefetching and Its Side Effects

The SH7751 Group is provided with an internal buffer for holding pre-read instructions, and always performs pre-reading. Therefore, program code must not be located in the last 20-byte area of any memory space. If program code is located in these areas, the memory area will be exceeded and a bus access for instruction pre-reading may be initiated. A case in which this is a problem is shown below.

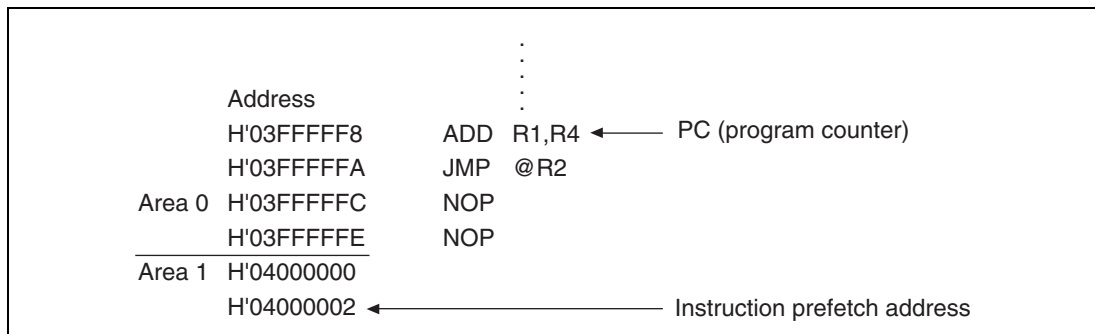


Figure F.1 Instruction Prefetch

Figure F.1 presupposes a case in which the instruction (ADD) indicated by the program counter (PC) and the address H'04000002 instruction prefetch are executed simultaneously. It is also assumed that the program branches to an area other than area 1 after executing the following JMP instruction and delay slot instruction.

In this case, the program flow is unpredictable, and a bus access (instruction prefetch) to area 1 may be initiated.

Instruction Prefetch Side Effects

1. It is possible that an external bus access caused by an instruction prefetch may result in misoperation of an external device, such as a FIFO, connected to the area concerned.
2. If there is no device to reply to an external bus request caused by an instruction prefetch, hangup will occur.

Remedies

1. These illegal instruction fetches can be avoided by using the MMU.
2. The problem can be avoided by not locating program code in the last 20 bytes of any area.

Appendix G Power-On and Power-Off Procedures

G.1 Power-On Stipulations

1. Supply power to power supply V_{DDQ} and to I/O, RTC, CPG, PLL1, and PLL2 simultaneously.
2. Perform input to the signal lines ($\overline{\text{RESET}}$, $\overline{\text{MRESET}}$, MD0 to MD10, external clock, etc.) after or at the same time power is supplied to V_{DDQ} . Applying input to signal lines before power is supplied to V_{DDQ} could damage the product.
 - Drive the $\overline{\text{RESET}}$ signal low when power is first supplied to V_{DDQ} .
3. Apply power such that the voltage of power supply V_{DD} is less than 1.2 V until the voltage of power supply V_{DDQ} reaches 2 V. Note that the on-chip PLL circuit (PLL2) may not operate correctly if this condition is not met.
4. It is recommended to apply power first to power supply V_{DDQ} and then to power supply V_{DD} .
5. In addition to 1., 2., 3., and 4. above, also observe the stipulations in G.3. Furthermore:
 - There are no time restrictions on the power-on sequence for power supply V_{DDQ} and power supply V_{DD} with regard to the LSI alone. Refer to figure G.1. Nevertheless, it is recommended that the power-on sequence be completed in as short a time as possible.
 - When the LSI is mounted on a board and connected to other elements, ensure that $-0.3 \text{ V} < V_{in} < V_{DDQ} + 0.3 \text{ V}$. In addition, the time limit for the rise of either power supply V_{DDQ} or power supply V_{DD} from $V_{DDQ} \geq 1.0 \text{ V}$ or $V_{DD} \geq 0.5 \text{ V}$, respectively, to above the minimum values in the LSI's guaranteed operation voltage range ($V_{DDQ}(\text{min.})$ and $V_{DD}(\text{min.})$) is 100 ms (max.), as shown in figure G.2. The product may be damaged if this time limit is exceeded. It is recommended that the power-on sequence be completed in as short a time as possible.

G.2 Power-Off Stipulations

1. Power off power supply V_{DDQ} and I/O, RTC, CPG, PLL1, and PLL2 simultaneously.
2. There are no timing restrictions for the $\overline{\text{RESET}}$ and $\overline{\text{MRESET}}$ signal lines at power-off.
3. Cut off the input signal level for signal lines other than $\overline{\text{RESET}}$ and $\overline{\text{MRESET}}$ in the same sequence as power supply V_{DDQ} .
4. It is recommended to first power off power supply V_{DD} and then power supply V_{DDQ} .
5. In addition to 1., 2., 3., and 4. above, also observe the stipulations in G.3. Furthermore:
 - There are no time restrictions on the power-off sequence for power supply V_{DDQ} and power supply V_{DD} with regard to the LSI alone. Refer to figure G.2. Nevertheless, it is recommended that the power-off sequence be completed in as short a time as possible.

- When the LSI is mounted on a board and connected to other elements, ensure that $-0.3 \text{ V} < V_{in} < V_{DDQ} + 0.3 \text{ V}$. In addition, the time limit for the fall of power supply V_{DDQ} and power supply V_{DD} from the minimum values in the LSI's guaranteed operation voltage range ($V_{DDQ}(\text{min.})$ and $V_{DD}(\text{min.})$) to $V_{DDQ} \geq 1.0 \text{ V}$ or $V_{DD} \geq 0.5 \text{ V}$, respectively, is 150 ms (max.), as shown in figure G.3. The product may be damaged if this time limit is exceeded. It is recommended that the power-off sequence be completed in as short a time as possible.

Notes: 1. Note on Power-On

If the below conditions (A) are not met during power-on, PLL2 may not oscillate correctly and CKIO may not be output properly.

Conditions (A):

V_{DDQ} (V_{DDQ} , V_{DD-CPG} , V_{DD-RTC}) is 2.0 V or above when V_{DD} (V_{DD} , $V_{DD-PLL1}$, $V_{DD-PLL2}$) is 1.2 V or above.

2. Workarounds

Any of methods (1) to (3) below may be used to avoid the problem by stopping PLL2 oscillation temporarily.

- (1) As shown in figure G.1, select mode 6*¹ immediately after power-on, select the desired clock mode once the above conditions (A) are satisfied, and cancel the power-on reset.
- (2) After starting with clock operation mode 6*¹ selected, change FRQCR to specify the desired frequency clock.

Note: It is not possible to use frequency divider 1 when this method is employed.

- (3) Temporarily stop PLL2 by writing 0 to FRQCR.PLL2EN. After maintaining FRQCR.PLL2EN as 0 for 1 μs or more, write 1 to FRQCR.PLL2EN to restart PLL2.

Note: If this method is used, the clock output from CKIO cannot be guaranteed until the above operations are completed. If abnormal signal output is produced, the frequency is higher than normal. Therefore, it is possible that unwanted noise may be generated from the clock line or, if the LSI's CKIO pin is used to supply a clock to another device, the clock may not be supplied correctly to the external device. When using this method, it is recommended that sufficient verification testing be performed on the actual system.

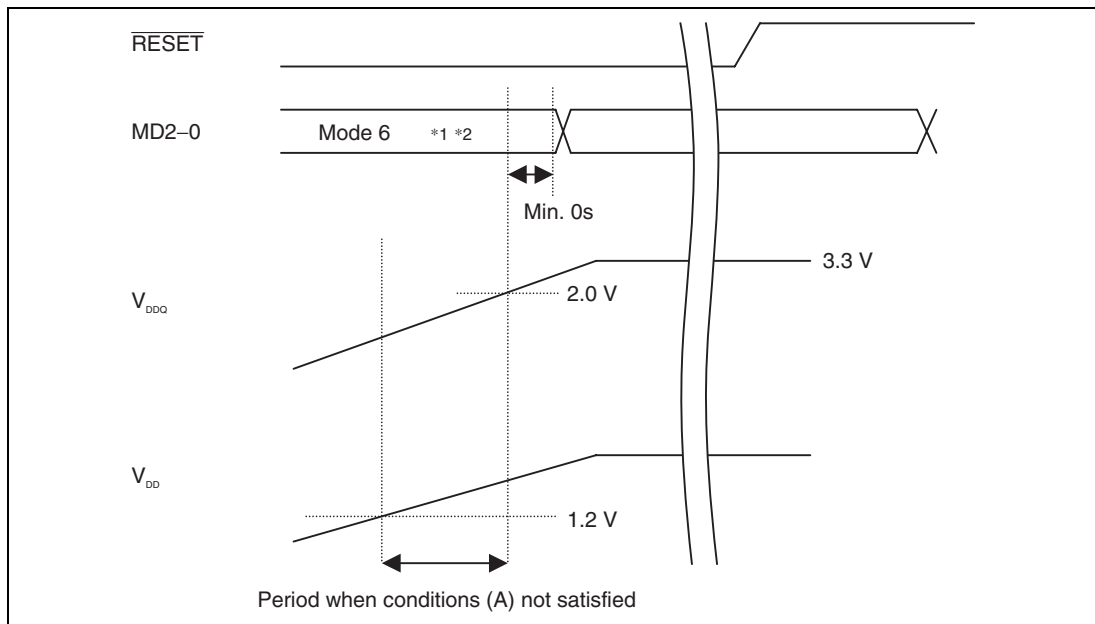


Figure G.1 Method for Temporarily Selecting Clock Operation Mode 6

Notes: 1. Clock operation mode 6

(I) SH7751

- (1) External pin combination: MD0 = low, MD1 = high, MD2 = high
- (2) Frequency dividers 1 and 2 = off, PLL1 = off, PLL2 = off
- (3) Frequencies (relative to input clock):
 - CPU clock = 1
 - Bus clock = 1/2
 - Peripheral module clock = 1/2
- (4) Input clock frequency range = 1 to 66.7 MHz

(II) SH7751R

- (1) External pin combination: MD0 = low, MD1 = high, MD2 = high
- (2) PLL1 = off (x6), PLL2 = off
- (3) Frequencies (relative to input clock):
 - CPU clock = 1
 - Bus clock = 1/2
 - Peripheral module clock = 1/2
- (4) Input clock frequency range = 1 to 34 MHz

2. Input to the MD should be high-level and follow the voltage level of the I/O, PLL, RTC, and CPG power supplies.

G.3 Common Stipulations for Power-On and Power-Off

1. Always ensure that $V_{DDQ} = V_{DD-CPG} = V_{DD-RTC} = V_{DD-PLL1/2}$ *

Refer to 9.9.5, Hardware Standby Mode Timing, regarding V_{DD-RTC} in hardware standby mode.

2. Ensure that $-0.3 \text{ V} < V_{DD} < V_{DDQ} + 0.3 \text{ V}$.
3. Ensure that $V_{SS} = V_{SSQ} = V_{SS-PLL1/2} = V_{SS-CPG} = V_{SS-RTC} = \text{GND} (0 \text{ V})$.

The product may be damaged if conditions 1., 2., and 3. above are not satisfied.

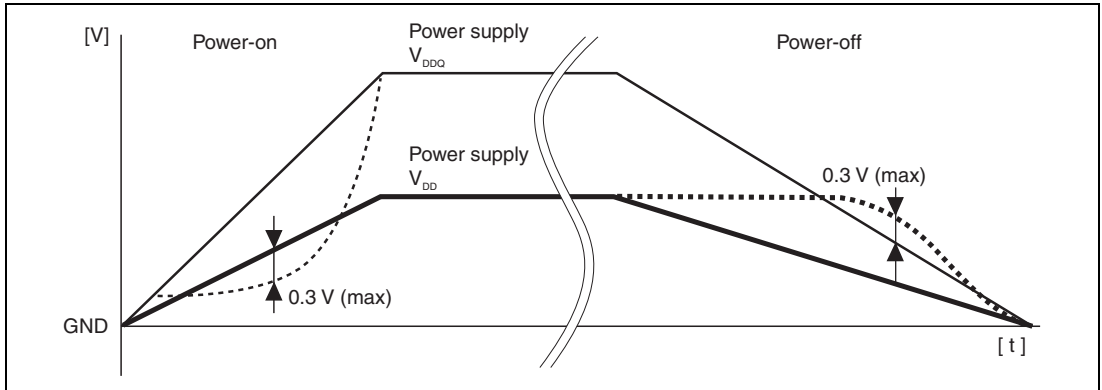


Figure G.2 Power-On Procedure 1

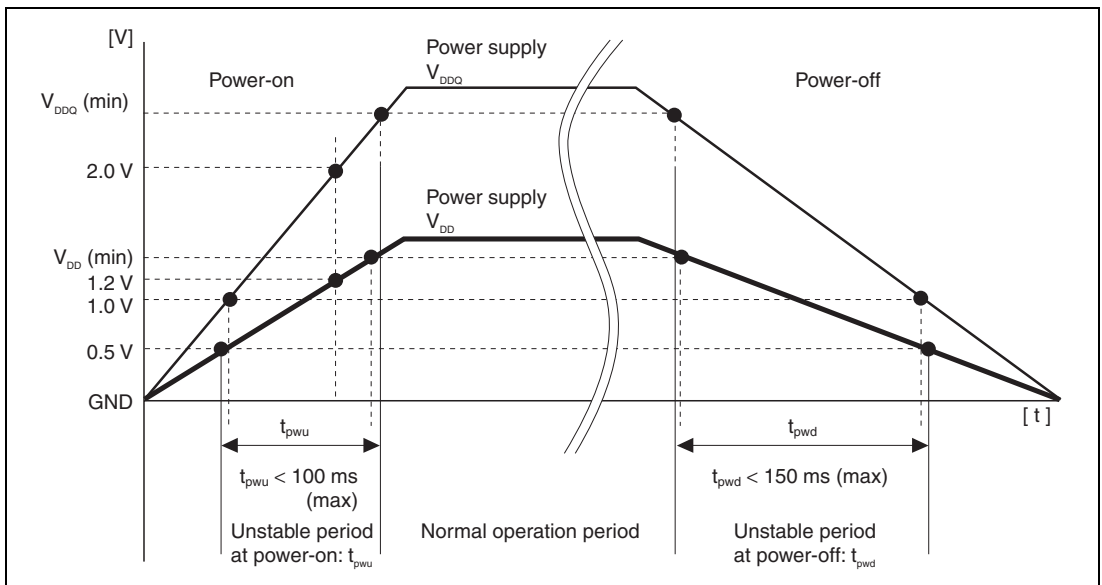


Figure G.3 Power-On Procedure 2

Appendix H Product Lineup

Table H.1 SH7751/SH7751R Product Lineup

| Product Name | Voltage | Operating Frequency | Operating Temperature* ¹ | Part Number* ² | Package |
|--------------|---------|---------------------|-------------------------------------|---------------------------|-------------|
| SH7751 | 1.8 V | 167 MHz | -20 to 75°C | HD6417751BP167 (V) | 256-pin BGA |
| | | | | HD6417751F167 (V) | 256-pin QFP |
| SH7751R | 1.5 V | 240 MHz | -20 to 75°C | HD6417751RBP240 (V) | 256-pin BGA |
| | | | | HD6417751RF240 (V) | 256-pin QFP |
| | | | | HD6417751RBG240 (V) | 292-pin BGA |
| | | 200 MHz | HD6417751RBP200 (V) | 256-pin BGA | |
| | | | HD6417751RF200 (V) | 256-pin QFP | |
| | | | HD6417751RBG200 (V) | 292-pin BGA | |

- Notes: 1. Contact a Renesas sales office regarding product versions with specifications for a wider temperature range (-40 to +85°C). The wide temperature range (-40 to +85°C) is the standard specification for the HD6417751RBA240HV.
2. All listed products are available in lead-free versions. Lead-free products have a "V" appended at the end of the part number.

Appendix I Version Registers

The configuration of the registers related to the product version is shown below.

Table I.1 Register Configuration

| Name | Abbreviation | Read/Write | Initial value | P4 Address | Area 7 Address | Access Size |
|----------------------------|--------------|------------|---------------|------------|----------------|-------------|
| Processor version register | PVR | R | * | H'FF000030 | H'1F000030 | 32 |
| Product register | PRR | R | * | H'FF000044 | H'1F000044 | 32 |

Note: * Refer to table below.

PVR and PRR Initial Values

| Product Name | PVR | PRR |
|--------------|------------|------------|
| SH7751 | H'041100xx | H'xxxxxxxx |
| SH7751R | H'040500xx | H'0000011x |

Legend: x: Undefined

1. Processor Version Register (PVR) Initial Value Example for SH7751R

| | | | | | | | | | | | | | | | | |
|----------------|---------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | Version information | | | | | | | | | | | | | | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| R/W: | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| | | | | | | | | | | | | | | | | |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Version information | | | | | | | | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | — | — | — | — | — | — | — | — |
| R/W: | R | R | R | R | R | R | R | R | — | — | — | — | — | — | — | — |

2. Product Register (PRR) Initial Value Example for SH7751R

| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|----------------|---------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| | Version information | | | | | | | | | | | | | | | |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W: | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Version information | | | | | | | | | | | | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | — | — | — | — |
| R/W: | R | R | R | R | R | R | R | R | R | R | R | R | — | — | — | — |

Renesas 32-Bit RISC Microcomputer
SH7751 Group, SH7751R Group
User's Manual: Hardware

Publication Date: 1st Edition, April, 2000
Rev.4.01, September 24, 2013

Published by: Renesas Electronics Corporation



SALES OFFICES

Renesas Electronics Corporation

<http://www.renesas.com>

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2690 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 D. Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhichunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Laviered or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

SH7751 Group, SH7751R Group
User's Manual: Hardware