REJ09B0074-0700

Everywhere you imagine. RENESAS

The revision list can be viewed directly by clicking the title page. The revision list summarizes the locations of revisions and additions. Details should always be checked by referring to the relevant text.

16

# H8S/2218 Group, H8S/2212 Group
## Hardware Manual

Hardware Manual

Renesas 16-Bit Single-Chip Microcomputer
H8S Family/H8S/2200 Series

| | |
|---|---|
| H8S/2218 | HD64F2218 |
| | HD64F2218U |
| | HD6432217 |
| H8S/2218C | HD64F2218CU |
| | HD64F2217CU |
| H8S/2212 | HD64F2212 |
| | HD64F2212U |
| | HD64F2211 |
| | HD64F2211U |
| | HD6432211 |
| | HD6432210 |
| | HD6432210S |
| H8S/2212C | HD64F2212CU |
| | HD64F2211CU |
| | HD64F2210CU |

Rev.7.00
Revision date: Dec. 24, 2008

Renesas Technology
www.renesas.com

## Notes regarding these materials

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

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

# General Precautions in the Handling of MPU/MCU Products

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

1. Handling of Unused Pins

   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions may occur due to the false recognition of the pin state as an input signal. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.

   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

# Configuration of This Manual

This manual comprises the following items:

1. General Precautions on Handling of Product
2. Configuration of This Manual
3. Preface
4. Main Revisions for This Edition (only for revised versions)

The list of revisions is a summary of points that have been revised or added to earlier versions. This does not include all of the revised contents. For details, see the actual locations in this manual.

5. Contents
6. Overview
7. Description of Functional Modules
   - CPU and System-Control Modules
   - On-Chip Peripheral Modules

     The configuration of the functional description of each module differs according to the module. However, the generic style includes the following items:

   i) Feature
   ii) Input/Output Pin
   iii) Register Description
   iv) Operation
   v) Usage Note

When designing an application system that includes this LSI, take notes into account. Each section includes notes in relation to the descriptions given, and usage notes are given, as required, as the final part of each section.

8. List of Registers
9. Electrical Characteristics
10. Appendix
11. Index

# Preface

This LSI is a microcomputer (MCU) made up of the H8S/2000 CPU with Renesas Technology's original architecture as its core, and the peripheral functions required to configure a system.

The H8S/2000 CPU has an internal 32-bit configuration, sixteen 16-bit general registers, and a simple and optimized instruction set for high-speed operation. The H8S/2000 CPU can handle a 16-Mbyte linear address space.

This LSI is equipped with ROM and RAM, a direct memory access controller (DMAC), a bus master, a 16-bit timer pulse unit (TPU), a watchdog timer (WDT), a realtime clock (RTC), a universal serial bus (USB), two types of serial communication interfaces (SCIs), an A/D converter, and I/O ports as on-chip peripheral modules for system configuration.

A single-power flash memory (F-ZTAT™*) version and masked ROM version are available for this LSI's ROM. The F-ZTAT version provides flexibility as it can be reprogrammed in no time to cope with all situations from the early stages of mass production to full-scale mass production. This is particularly applicable to application devices with specifications that will most probably change.

This manual describes this LSI's hardware.

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

Target Users: This manual was written for users who will be using the H8S/2218 Group and H8S/2212 Group in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logical circuits, and microcomputers.

Objective: This manual was written to explain the hardware functions and electrical characteristics of the H8S/2218 Group and H8S/2212 Group to the target users. Refer to the H8S/2600 Series, H8S/2000 Series Software Manual for a detailed description of the instruction set.

Notes on reading this manual:

- In order to understand the overall functions of the chip
  Read the manual according to the contents. This manual can be roughly categorized into parts on the CPU, system control functions, peripheral functions and electrical characteristics.
- In order to understand the details of the CPU's functions
  Read the H8S/2600 Series, H8S/2000 Series Software Manual.

- In order to understand the details of a register when its name is known

  Read the index that is the final part of the manual to find the page number of the entry on the register. The addresses, bits, and initial values of the registers are summarized in section 21, List of Registers.

  Examples:

  | | |
  |---|---|
  | Register name: | The following notation is used for cases when the same or a similar function, e.g. 16-bit timer pulse unit or serial communication, is implemented on more than one channel: XXX_N (XXX is the register name and N is the channel number) |
  | Bit order: | The MSB is on the left and the LSB is on the right. |
  | Number notation: | Binary is B'xxxx, hexadecimal is H'xxxx. |
  | Signal notation: | An overbar is added to a low-active signal: $\overline{\text{xxxx}}$ |

Related Manuals: The latest versions of all related manuals are available from our web site. Please ensure you have the latest versions of all documents you require. http://www.renesas.com/

H8S/2218 Group, H8S/2212 Group Manuals:

| Document Title | Document No. |
|---|---|
| H8S/2218 Group, H8S/2212 Group Hardware Manual | This manual |
| H8S/2600 Series, H8S/2000 Series Software Manual | REJ09B0139 |

User's Manuals for Development Tools:

| Document Title | Document No. |
|---|---|
| H8S, H8/300 Series C/C++ Compiler, Assembler, Optimizing Linkage Editor Compiler Package Ver. 6.01 User's Manual | REJ10B0161 |
| H8S, H8/300 Series Simulator/Debugger (for Windows) User's Manual | REJ10B0211 |
| H8S, H8/300 Series High-performance Embedded Workshop, High-performance Debugging Interface Tutorial | ADE-702-231 |
| High-performance Embedded Workshop User's Manual | ADE-702-201 |

Application Notes:

| Document Title | Document No. |
|---|---|
| H8S, H8/300 Series C/C++ Compiler Package Application Note | REJ05B0464 |

# Main Revisions for This Edition

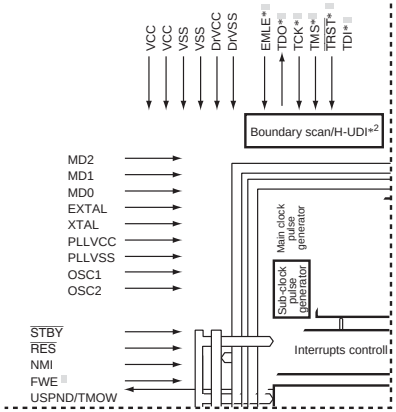
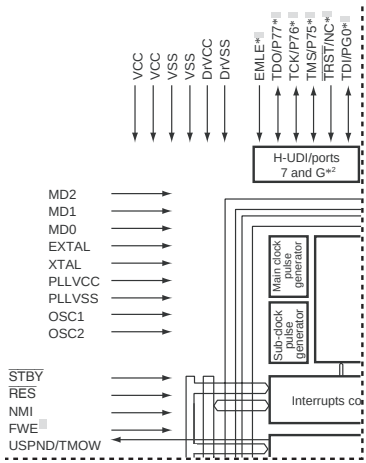
| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 1.1 Overview<br>• On-chip memory | 1 | Table amended<br>H8S/2218 Group |

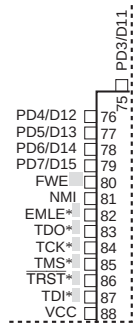
| ROM | Part No. | ROM | RAM | Remarks |
|---|---|---|---|---|
| Flash memory Version | HD64F2218 | 128 kbytes | 12 kbytes | SCI boot mode |
| | HD64F2218U | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2218CU | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2217CU | 64 kbytes | 12 kbytes | USB boot mode |
| Masked ROM Version | HD6432217 | 64 kbytes | 8 kbytes | — |

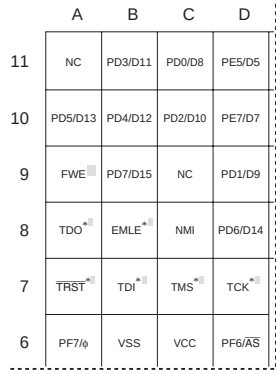
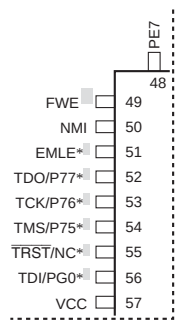
| | | |
|---|---|---|
| | 2 | Table amended<br>H8S/2212 Group |

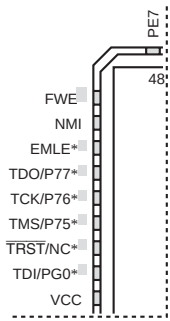
| ROM | Part No. | ROM | RAM | Remarks |
|---|---|---|---|---|
| Flash memory Version | HD64F2212 | 128 kbytes | 12 kbytes | SCI boot mode |
| | HD64F2212U | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2212CU | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2211 | 64 kbytes | 8 kbytes | SCI boot mode |
| | HD64F2211U | 64 kbytes | 8 kbytes | USB boot mode |
| | HD64F2211CU | 64 kbytes | 8 kbytes | USB boot mode |
| | HD64F2210CU | 32 kbytes | 8 kbytes | USB boot mode |
| Masked ROM Version | HD6432211 | 64 kbytes | 8 kbytes | — |
| | HD6432210 | 32 kbytes | 4 kbytes | — |
| | HD6432210S | 32 kbytes | 4 kbytes | — |

| | | |
|---|---|---|
| 1.2 Internal Block Diagram | 3 | Description amended<br>The internal block diagram of the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU is shown in figure 1.1. The internal block diagram of the HD6432217 is shown in figure 1.2. The internal block diagram of the HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU is shown in figure 1.3. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 1.2 Internal Block Diagram<br>Figure 1.1 Internal Block Diagram of HD64F2218, HD64F2218U, HD64F2218CU and HD642217CU | 3 | Title and figure amended<br>  <br>Note *1 shown below deleted<br>Notes: 1. The FWE pin is provided only in the HD64F2218 and HD64F2218U. |
| Figure 1.2 Internal Block Diagram of HD6432217 | 4 | Note amended<br>1. The FWE pin is provided only in the flash memory version. |
| Figure 1.3 Internal Block Diagram of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU | 5 | Title and figure amended<br>  <br>Note *1 shown below deleted<br>Notes: 1. The FWE pin is provided only in the HD64F2212, HD64F2212U, HD64F2211 andHD64F2211U. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 1.2 Internal Block Diagram<br>Figure 1.4 Internal Block Diagram of HD6432211, HD6432210 and HD6432210S | 6 | Note amended<br>1. The FWE pin is provided only in the flash memory version. |
| 1.3 Pin Arrangements | 7 | Description amended<br>The pin arrangements of the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU are shown in figures 1.5 and 1.6. The pin arrangements of the HD6432217 are shown in figures 1.7 and 1.8. The pin arrangements of the HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU are shown in figures 1.9 and 1.11. |
| Figure 1.5 Pin Arrangements of HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU (TFP-100G, TFP-100GV) | | Title and figure amended<br>  <br>Note *1 shown below deleted<br>Notes: 1. The FWE pin is provided only in the HD64F2218, andHD64F2218U. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| Figure 1.6 Pin Arrangements of HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU (BP-112, BP-112V) | 8 | Title and figure amended<br>  <br><br>Note *1 shown below deleted<br>Notes: 1. The FWE pin is provided only in the HD64F2218, andHD64F2218U. |
| Figure 1.7 Pin Arrangements of HD6432217 (TFP-100G, TFP-100GV) | 9 | Note amended<br>1. The FWE pin is provided only in the flash memory version. |
| Figure 1.8 Pin Arrangements of HD6432217 (BP-112, BP-112V) | 10 | Note amended<br>1. The FWE pin is provided only in the flash memory version. |
| Figure 1.9 Pin Arrangements of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU (FP-64E, FP-64EV) | 11 | Title and figure amended<br>  <br><br>Note *1 shown below deleted<br>Notes: 1. The FWE pin is provided only in the HD64F2212, HD64F2212U, HD64F2211 andHD64F2211U. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 1.3 Pin Arrangement<br>Figure 1.10 Pin Arrangements of HD6432211, HD6432210 and HD6432210S (FP-64E, FP-64EV) | 12 | Note amended<br>1. The FWE pin is provided only in the flash memory version. |
| Figure 1.11 Pin Arrangements of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU (TNP-64B, TNP-64BV) | 13 | Title and figure amended<br>  <br><br>Note *1 shown below deleted<br>Notes: 1. The FWE pin is provided only in the HD64F2212, HD64F2212U, HD64F2211 andHD64F2211U. |
| Figure 1.12 Pin Arrangements of HD6432211, HD6432210 and HD6432210S (TNP-64B, TNP-64BV) | 14 | Note amended<br>1. The FWE pin is provided only in the flash memory version. |
| 1.5 Pin Functions | 26 | Table amended |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| Boundary scan (Supported only by the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU) | TMS | 85 | C7 | 54 | Input | Control signal input pin for the boundary scan |
| | TCK | 84 | D7 | 53 | Input | Clock input pin for the boundary scan |
| | TDO | 83 | A8 | 52 | Output | Data output pin for the boundary scan |
| | TDI | 87 | B7 | 56 | Input | Data input pin for the boundary scan |
| | $\overline{\text{TRST}}$ | 86 | A7 | 55 | Input | Reset pin for the TAP controller |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 3.4 Memory Map in Each Operating Mode<br>Figure 3.1 Memory Map in Each Operating Mode for HD64F2218, HD64F2218U and HD64F2218CU | 77 | Title amended |
| Figure 3.2 Memory Map in Each Operating Mode for HD64F2217CU | 78 | Figure added |
| Figure 3.4 Memory Map in Each Operating Mode for HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU, HD64F2210CU, HD6432211, HD6432210 and HD6432210S | 80 | Figure replaced |
| 4.3 Reset | 83 | Notes amended<br>Notes: TRST in the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU, which incorporate a boundary scan function, should be brought low when power is on. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 12.3.7 Serial Status Register (SSR)<br>• Normal Serial Communication Interface Mode (When SMIF in SCMR is 0) | 377 to 379 | Note added |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TDRE | 1 | R/(W)*1 | Transmit Data Register Empty<br>……<br>[Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1*2<br>…… |
| 6 | RDRF | 0 | R/(W)*1 | Receive Data Register Full<br>……<br>[Clearing conditions]<br>• When 0 is written to RDRF after reading RDRF = 1*2<br>…… |
| 5 | ORER | 0 | R/(W)*1 | Overrun Error<br>……<br>[Clearing condition]<br>• When 0 is written to ORER after reading ORER = 1*2<br>…… |
| 4 | FER | 0 | R/(W)*1 | Framing Error<br>……<br>[Clearing condition]<br>• When 0 is written to FER after reading FER = 1*2<br>…… |
| 3 | PER | 0 | R/(W)*1 | Parity Error<br>……<br>[Clearing condition]<br>• When 0 is written to PER after reading PER = 1*2<br>…… |

Note: 1. The write value should always be 0 to clear the flag.
2. To clear the flag by the CPU on the HD6432210S, reread the flag after writing 0 to it.

| | | |
|---|---|---|
| • Smart Card Interface Mode (When SMIF in SCMR is 1) | 380 to 383 | Note added |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TDRE | 1 | R/(W)*1 | Transmit Data Register Empty<br>……<br>[Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1*2<br>…… |
| 6 | RDRF | 0 | R/(W)*1 | Receive Data Register Full<br>……<br>[Clearing conditions]<br>• When 0 is written to RDRF after reading RDRF = 1*2<br>…… |
| 5 | ORER | 0 | R/(W)*1 | Overrun Error<br>……<br>[Clearing condition]<br>• When 0 is written to ORER after reading ORER = 1*2<br>…… |
| 4 | ERS | 0 | R/(W)*1 | Error Signal Status<br>……<br>[Clearing condition]<br>• When 0 is written to ERS after reading ERS = 1*2<br>…… |
| 3 | PER | 0 | R/(W)*1 | Parity Error<br>……<br>[Clearing condition]<br>• When 0 is written to PER after reading PER = 1*2<br>…… |

Note: 1. The write value should always be 0 to clear the flag.
2. To clear the flag by the CPU on the HD6432210S, reread the flag after writing 0 to it.

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 12.3.11 Bit Rate Register (BRR)<br>• Table 12.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode) | 401 | Table amended |

| Bit Rate (bps) | Operating Frequency φ (MHz) 2 | | | ····· | 24 | |
|---|---|---|---|---|---|---|
| | n | N | n | | n | N |
| 110 | 3 | 70 | — | ····· | | |
| 250 | 2 | 124 | 2 | ····· | | |
| 500 | 1 | 249 | 2 | ····· | — | — |
| 1 k | 1 | 124 | 1 | ····· | — | — |
| 2.5 k | 0 | 199 | 1 | ····· | 2 | 149 |
| 5 k | 0 | 99 | 0 | ····· | 2 | 74 |
| 10 k | 0 | 49 | 0 | ····· | 1 | 149 |
| 25 k | 0 | 19 | 0 | ····· | 0 | 239 |
| 50 k | 0 | 9 | 0 | ····· | 0 | 119 |
| 100 k | 0 | 4 | 0 | ····· | 0 | 59 |
| 250 k | 0 | 1 | 0 | ····· | 0 | 23 |
| 500 k | 0 | 0* | 0 | ····· | 0 | 11 |
| 1 M | | | 0 | ····· | 0 | 5 |
| 2 M | | | | ····· | 0 | 2 |
| 2.5 M | | | | ····· | | |
| 4 M | | | | ····· | | |
| 5 M | | | | ····· | | |
| 6 M | | | | ····· | 0 | 0* |

| | | |
|---|---|---|
| 12.4.2. Receive Data Sampling Timing and Reception Margin in Asynchronous Mode<br>Figure 12.6 Receive Data Sampling Timing in Asynchronous Mode | 405 | Note added<br>Note: * In this example the value of the ABCS bit in SEMRA_0 is 0. When ABCS is set to 1, the basic clock frequency is eight times the bit rate and the receive data is sampled at the fourth rising edge of the basic clock. |
| Section 13 Boundary Scan Function | 449 | Description amended<br>The HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU incorporate a boundary scan function, which is a serial I/O interface based on the JTAG (Joint Test Action Group, IEEEStd.1149.1 and IEEE Standard Test Access Port and Boundary Scan Architecture). |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 13.3.2 IDCODE Register (IDCODE) | 454 | Description amended<br>...The HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU output fixed codes H'002A200F from the TDO.<br>... |
| Table 13.3 IDCODE Register Configuration | | Table amended<br>**Bits**<br>HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU codes<br>Contents |
| 14.3.5 USB FIFO Clear Register 0 (UFCLR0) | 475 to 476 | Bit Table amended<br>**Bit** / **Bit Name** / **Initial Value** / **R/W** / **Description**<br>7, 6 — All 0 R Reserved<br>These bits are always rea modified.<br>5 EP2CLR 0 W EP2 Clear*<br>0: Performs no operation.<br>1: Clears EP2 OUT FIFO.<br>Note added<br>Note:* When DMA writes are enabled (EP2T1 set to 1 and EP2T0 set to 0 or 1 in UDMAR), it is not possible to clear the data in the FIFO by writing 1 to EP2CLR. To clear the data in the FIFO, disable DMA transfers (clear EP2T1 and EP2T0 in UDMAR to 0) and then write 1 to EP2CLR. |
| 14.8.16 Clearing the FIFO when DMA Transfer Is Enabled | 533 | Description added<br>When DMA transfer is enabled (EP2T1 = 1 and EP2T0 = 0 or 1 in UDMAR) at endpoint 2, it is not possible to clear OUTFIFO in EP2. It is necessary to disable DMA transfer (EP2T1 = 0 and EP2T0 = 0 in UDMAR) before clearing the FIFO. |
| Section 16 RAM | 551 | Description amended<br>The HD64F2218, HD64F2218U, and HD64F2218CU have 12 kbytes of on-chip high-speed static RAM. The HD6432217, HD64F2211, HD64F2211U, and HD64F2211CU have 8 kbytes of on-chip high-speed static RAM. The HD6432210 and HD6432210S have 4 kbytes of on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| Section 16 RAM | 551 | Table amended |

| Product Class | | ROM Type | RAM Size | RAM Address |
|---|---|---|---|---|
| H8S/2218 Group | HD64F2218<br>HD64F2218U<br>HD64F2218CU<br>HD64F2217CU | Flash memory Version | 12 kbytes | H'FFC000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD6432217 | Masked ROM Version | 8 kbytes | H'FFD000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| H8S/2212 Group | HDF64F2212<br>HDF64F2212U<br>HDF64F2212CU | Flash memory Version | 12 kbytes | H'FFC000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD64F2211<br>HD64F2211U<br>HD64F2211CU<br>HD64F2210CU | | 8 kbytes | H'FFD000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD6432211 | Masked ROM Version | 8 kbytes | H'FFD000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD6432210<br>HD6432210S | | 4 kbytes | H'FFE000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |

| | | |
|---|---|---|
| 17.1 Features | 553 | Table amended |

- Size:

| Product Class | | ROM Size | ROM Address |
|---|---|---|---|
| H8S/2218 Group | HD64F2218, HD64F2218U<br>HD64F2218CU | 128 kbytes | H'000000 to H'01FFFF (Modes 6 and 7) |
| | HD64F2217CU | 64 kbytes | H'000000 to H'00FFFF (Modes 6 and 7) |
| H8S/2212 Group | HD64F2212, HD64F2212U<br>HD64F2212CU | 128 kbytes | H'000000 to H'01FFFF (Mode 7) |
| | HD64F2211, HD64F2211U<br>HD64F2211CU | 64 kbytes | H'000000 to H'00FFFF (Mode 7) |
| | HD64F2210CU | 32 kbytes | H'000000 to H'007FFF (Mode 7) |

- Two flash memory operating modes

Description amended

— Boot mode

SCI boot mode: HD64F2218, HD64F2212, and HD64F2211
USB boot mode: HD64F2218U, HD64F2218CU, HD64F2217CU, HD64F2212U, HD64F2212CU, HD64F2211U, HD64F2211CU and HD64F2210CU

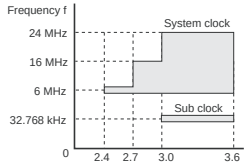
| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 17.1 Features | 554 | Note amended |
| Figure 17.1 Block Diagram of Flash Memory | | Note: * 128 kbytes in the HD64F2218, HD64F2218U, HD64F2218CU, HD64F2212, HD64F2212U and HD64F2212CU;<br>64 kbytes in the HD64F2217CU, HD64F2211, HD64F2211U and HD64F2211CU.<br>32 kbytes in the HD64F2210CU |
| 17.3 Block Configuration | 558 | Description amended<br>Figure 17.5 shows the block configuration of 128-kbyte flash memory in the HD64F2218, HD64F2218U, HD64F2218CU, HDHD64F2212, HD64F2212U and HD64F2212CU. |
| Figure 17.5 Flash Memory Block Configuration (HD64F2218, HD64F2218U, HD64F2218CU, HD64F2212, HD64F2212U, HD64F2212CU) | | Title amended |
| Figure 17.6 Flash Memory Block Configuration (HD64F2217CU, HD64F2211, HD64F2211U, HD64F2211CU) | 559 | Description and title amended<br>Figure 17.6 shows the block configuration of 64-kbyte flash memory in the HD64F2217CU, HD64F2211, HD64F2211U and HD64F2211CU. |
| Figure 17.7 Flash Memory Block Configuration (HD64F2210CU) | 560 | Figure and description added |
| 17.4 Input/Output Pins<br>Table 17.2 Pin Configuration | 561 | Table amended |

| Pin Name | I/O | Function | |
|---|---|---|---|
| USD+, USD– | Input/output | USB data input/output | HD64F2218U, HD64F2218CU, HD64F2217CU, HD64F2212U, HD64F2212CU, HD64F2211U |
| VBUS | Input | USB cable connect/cut detect | |
| $\overline{\text{UBPM}}$ | Input | USB bus power mode/self power mode select | |
| USPND | Output | USB suspend output | |
| P36 (PUPD+) | Output | D+ pull-up control | |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 17.6 On-Board Programming Modes<br>Table 17.3 Setting On-Board Programming Modes | 567 | Table amended<br>Mode<br>SCI boot mode (HD64F2218, HD64F2212, HD64F2211)<br>USB boot mode (HD64F2218U, HD64F2218CU, HD64F2217CU, HD64F2212U, HD64F2212CU, HD64F2211U, HD64F2211CU, HD64F2210CU) |
| 17.6.1 SCI Boot Mode (HD64F2218, HD64F2212, and HD64F2211) | 569 | Description amended<br>5. In boot mode, a part of the on-chip RAM area (four kbytes) is used by the boot program. The area to which the programming control program is transferred from the host is 8 kbytes (H'FFC000 to H'FFDFFF) in the HD64F2218 and HD64F2212 and 4 kbytes (H'FFD000 to H'FFDFFF) in the HD64F2211 . The boot program area cannot be used until the execution state in boot mode switches to the programming control program. |
| 17.6.2 USB Boot Mode (HD64F2218U, HD64F2212U, and HD64F2211U) | 573 | Description amended<br>4. In boot mode, the 4-kbyte on-chip RAM area H'FFE000 to H'FFEFBF is used by the boot program. The programming control program is transferred from the host stored in the 8-kbyte area H'FFC000 to H'FFDFFF in the HD64F2218U, HD64F2218CU, HD64F2212U, and HD64F2212CU and the 4-kbyte area H'FFD000 to H'FFDFFF in the HD64F2211U, HD64F2211CU and HD64F2210CU. |

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| 22.2 Power Supply Voltage and Operating Frequency Range<br>Figure 22.1 Power Supply Voltage and Operating Ranges | 658 | Figure amended |

(1) Mask ROM versions (except for HD6432210S)



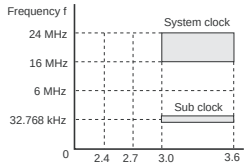

Condition A: Vcc = PLLVcc = DrVcc = 2.4 to 3.6V
Vref = 2.4V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 6 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

Condition B: Vcc = PLLVcc = DrVcc = 2.7 to 3.6V
Vref = 2.7V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 6 to 16 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

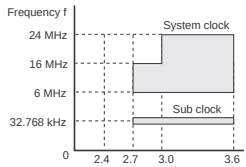
Condition C: Vcc = PLLVcc = DrVcc = 3.0 to 3.6V
Vref = 3.0V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 6 to 24 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

Condition D: Vcc = PLLVcc = DrVcc = 3.0 to 3.6V
Vref = 3.0V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 16 to 24 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

(2) Masked ROM version (HD6432210S)




(3) F-ZTAT versions (except for H8S/2218C, H8S/2212C)

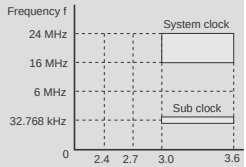



Condition A: None

Condition B: Vcc = PLLVcc = DrVcc = 2.7 to 3.6V
Vref = 2.7V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 6 to 16 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

Condition C: Vcc = PLLVcc = DrVcc = 3.0 to 3.6V
Vref = 3.0V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 6 to 24 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

(4) F-ZTAT versions (H8S/2218C, H8S/2212C)




Condition D: Vcc = PLLVcc = DrVcc = 3.0 to 3.6V
Vref = 3.0V to Vcc
Vss = PLLVss = DrVss = 0V
f = 32.768 kHz, 6 to 24 MHz
Ta = -20 to +75 (Regular specifications)
Ta = -40 to +85 (Wide-range specifications)

(5) When using the on-chip USB

| Item | Page | Revision (See Manual for Details) |
|---|---|---|
| B. Product Model Lineup | 689 to 690 | Table amended |

| Product Class | | Part No. | Model Name | Marking | Package (code) |
|---|---|---|---|---|---|
| H8S/2218 Group | Flash memory Version | HD64F2218 | HD64F2218TF24 | F2218TF24 | 100-pin TQFP (TFP-100G, TFP-100GV) |
| | | | HD64F2218BR24 | 64F2218BR24 | 112-pin P-LFBGA (BP-112, BP-112V) |
| | | HD64F2218CU | HD64F2218CUTF24 | F2218CUTF24 | 100-pin TQFP (TFP-100GV) |
| | | | HD64F2218CUBR24 | 64F2218CUBR24 | 112-pin P-LFBGA (BP-112V) |
| | | HD64F2217CU | HD64F2217CUTF24 | F2217CUTF24 | 100-pin TQFP (TFP-100GV) |
| | | | HD64F2217CUBR24 | 64F2217CUBR24 | 112-pin P-LFBGA (BP-112V) |
| | Masked ROM | HD6432217 | HD6432217(***)TF | 2217(***)TF | 100-pin TQFP (TFP-100G, TFP-100GV) |
| H8S/2212 Group | Flash memory Version | HD64F2212 | HD64F2212FP24 | 2212FP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2212NP24 | F2212NP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD64F2212U | HD64F2212UFP24 | 2212UFP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2212UNP24 | F2212UNP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD64F2212CU | HD64F2212CUFP24 | 2212CUFP24 | 64-pin LQFP (FP-64EV) |
| | | | HD64F2212CUNP24 | F2212CUNP24 | 64-pin VQFN (TNP-64BV) |
| | | HD64F2211 | HD64F2211FP24 | 2211FP24 | 64-pin LQFP |
| H8S/2212 Group | Flash memory Version | HD64F2211U | HD64F2211UFP24 | 2211UFP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2211UNP24 | F2211UNP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD64F2211CU | HD64F2211CUFP24 | 2211CUFP24 | 64-pin LQFP (FP-64EV) |
| | | | HD64F2211CUNP24 | F2211CUNP24 | 64-pin VQFN (TNP-64BV) |
| | | HD64F2210CU | HD64F2210CUFP24 | 2210CUFP24 | 64-pin LQFP (FP-64EV) |
| | | | HD64F2210CUNP24 | F2210CUNP24 | 64-pin VQFN (TNP-64BV) |

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

# Contents


Section 1 Overview ..... 1
1.1 Overview ..... 1
1.2 Internal Block Diagram ..... 3
1.3 Pin Arrangements ..... 7
1.4 Pin Functions in Each Operating Mode ..... 15
1.5 Pin Functions ..... 21

Section 2 CPU ..... 31
2.1 Features ..... 31
2.1.1 Differences between H8S/2600 CPU and H8S/2000 CPU ..... 32
2.1.2 Differences from H8/300 CPU ..... 33
2.1.3 Differences from H8/300H CPU ..... 33
2.2 CPU Operating Modes ..... 34
2.2.1 Normal Mode ..... 34
2.2.2 Advanced Mode ..... 36
2.3 Address Space ..... 38
2.4 Register Configuration ..... 39
2.4.1 General Registers ..... 40
2.4.2 Program Counter (PC) ..... 41
2.4.3 Extended Control Register (EXR) ..... 41
2.4.4 Condition-Code Register (CCR) ..... 42
2.4.5 Initial Register Values ..... 43
2.5 Data Formats ..... 43
2.5.1 General Register Data Formats ..... 43
2.5.2 Memory Data Formats ..... 45
2.6 Instruction Set ..... 45
2.6.1 Table of Instructions Classified by Function ..... 47
2.6.2 Basic Instruction Formats ..... 57
2.7 Addressing Modes and Effective Address Calculation ..... 58
2.7.1 Register Direct—Rn ..... 58
2.7.2 Register Indirect—@ERn ..... 58
2.7.3 Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn) ..... 59
2.7.4 Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn ..... 59
2.7.5 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32 ..... 59
2.7.6 Immediate—#xx:8, #xx:16, or #xx:32 ..... 60
2.7.7 Program-Counter Relative—@(d:8, PC) or @(d:16, PC) ..... 60
2.7.8 Memory Indirect—@@aa:8 ..... 61
2.7.9 Effective Address Calculation ..... 62

2.8 Processing States ..... 64
2.9 Usage Notes ..... 66
2.9.1 Note on TAS Instruction Usage ..... 66
2.9.2 STM/LTM Instruction Usage ..... 66
2.9.3 Note on Bit Manipulation Instructions ..... 66
2.9.4 Accessing Registers Containing Write-Only Bits ..... 68

Section 3 MCU Operating Modes ..... 71
3.1 Operating Mode Selection ..... 71
3.2 Register Descriptions ..... 72
3.2.1 Mode Control Register (MDCR) ..... 72
3.2.2 System Control Register (SYSCR) ..... 72
3.3 Operating Mode Descriptions ..... 74
3.3.1 Mode 4 (Supported Only by the H8S/2218 Group) ..... 74
3.3.2 Mode 5 (Supported Only by the H8S/2218 Group) ..... 74
3.3.3 Mode 6 (Supported Only by the H8S/2218 Group) ..... 75
3.3.4 Mode 7 ..... 75
3.3.5 Pin Functions ..... 76
3.4 Memory Map in Each Operating Mode ..... 77

Section 4 Exception Handling ..... 81
4.1 Exception Handling Types and Priority ..... 81
4.2 Exception Sources and Exception Vector Table ..... 81
4.3 Reset ..... 83
4.3.1 Reset Types ..... 83
4.3.2 Reset Exception Handling ..... 84
4.3.3 Interrupts after Reset ..... 86
4.3.4 State of On-Chip Peripheral Modules after Reset Release ..... 86
4.4 Traces ..... 87
4.5 Interrupts ..... 87
4.6 Trap Instruction ..... 88
4.7 Stack Status after Exception Handling ..... 89
4.8 Notes on Use of the Stack ..... 90

Section 5 Interrupt Controller ..... 91
5.1 Features ..... 91
5.2 Input/Output Pins ..... 93
5.3 Register Descriptions ..... 93
5.3.1 Interrupt Priority Registers A to G, J, K, M (IPRA to IPRG, IPRJ, IPRK, IPRM) ..... 94
5.3.2 IRQ Enable Register (IER) ..... 95

5.3.3 IRQ Sense Control Registers H and L (ISCRH, ISCRL) ... 96
5.3.4 IRQ Status Register (ISR) ... 98
5.4 Interrupt Sources ... 99
5.4.1 External Interrupts ... 99
5.4.2 Internal Interrupts ... 100
5.5 Interrupt Exception Handling Vector Table ... 101
5.6 Interrupt Control Modes and Interrupt Operation ... 103
5.6.1 Interrupt Control Mode 0 ... 103
5.6.2 Interrupt Control Mode 2 ... 105
5.6.3 Interrupt Exception Handling Sequence ... 107
5.6.4 Interrupt Response Times ... 108
5.6.5 DMAC Activation by Interrupt ... 109
5.7 Usage Notes ... 112
5.7.1 Contention between Interrupt Generation and Disabling ... 112
5.7.2 Instructions that Disable Interrupts ... 113
5.7.3 Times when Interrupts Are Disabled ... 113
5.7.4 Interrupts during Execution of EEPMOV Instruction ... 113
5.7.5 IRQ Interrupt ... 113
5.7.6 NMI Interrupt Usage Notes ... 114

Section 6 Bus Controller ... 115
6.1 Features ... 115
6.2 Input/Output Pins ... 117
6.3 Register Descriptions ... 118
6.3.1 Bus Width Control Register (ABWCR) ... 118
6.3.2 Access State Control Register (ASTCR) ... 119
6.3.3 Wait Control Registers H and L (WCRH, WCRL) ... 120
6.3.4 Bus Control Register H (BCRH) ... 124
6.3.5 Bus Control Register L (BCRL) ... 125
6.3.6 Pin Function Control Register (PFCR) ... 126
6.4 Bus Control ... 127
6.4.1 Area Divisions ... 127
6.4.2 Bus Specifications ... 128
6.4.3 Bus Interface for Each Area ... 129
6.4.4 Chip Select Signals ... 130
6.5 Basic Timing ... 131
6.5.1 On-Chip Memory (ROM, RAM) Access Timing ... 131
6.5.2 On-Chip Peripheral Module Access Timing ... 132
6.5.3 External Address Space Access Timing ... 133
6.6 Basic Bus Interface ... 134
6.6.1 Data Size and Data Alignment (Supported Only by the H8S/2218 Group) ... 134

6.6.2 Valid Strobes ... 135
6.6.3 Basic Timing ... 136
6.6.4 Wait Control ... 145
6.7 Burst ROM Interface ... 147
6.7.1 Basic Timing ... 147
6.7.2 Wait Control ... 149
6.8 Idle Cycle ... 149
6.9 Bus Release ... 153
6.9.1 Bus Release Usage Note ... 154
6.10 Bus Arbitration ... 155
6.10.1 Operation ... 155
6.10.2 Bus Transfer Timing ... 155
6.10.3 External Bus Release Usage Note ... 156
6.11 Resets and the Bus Controller ... 156

Section 7 DMA Controller (DMAC) ... 157
7.1 Features ... 157
7.2 Register Configuration ... 159
7.3 Register Descriptions ... 161
7.3.1 Memory Address Registers (MAR) ... 161
7.3.2 I/O Address Register (IOAR) ... 161
7.3.3 Execute Transfer Count Register (ETCR) ... 162
7.3.4 DMA Control Register (DMACR) ... 163
7.3.5 DMA Band Control Register (DMABCR) ... 169
7.4 Operation ... 177
7.4.1 Transfer Modes ... 177
7.4.2 Sequential Mode ... 178
7.4.3 Idle Mode ... 181
7.4.4 Repeat Mode ... 183
7.4.5 Normal Mode ... 186
7.4.6 Block Transfer Mode ... 189
7.4.7 DMAC Activation Sources ... 194
7.4.8 Basic DMAC Bus Cycles ... 196
7.4.9 DMAC Bus Cycles (Dual Address Mode) ... 197
7.4.10 DMAC Multi-Channel Operation ... 202
7.4.11 Relation between the DMAC and External Bus Requests ... 203
7.4.12 NMI Interrupts and DMAC ... 203
7.4.13 Forced Termination of DMAC Operation ... 204
7.4.14 Clearing Full Address Mode ... 205
7.5 Interrupts ... 206
7.6 Usage Notes ... 207

7.6.1 DMAC Register Access during Operation ..... 207
7.6.2 Module Stop ..... 208
7.6.3 Medium-Speed Mode ..... 208
7.6.4 Activation Source Acceptance ..... 209
7.6.5 Internal Interrupt after End of Transfer ..... 209
7.6.6 Channel Re-Setting ..... 209

Section 8 I/O Ports ..... 211
8.1 Port 1 ..... 216
8.1.1 Port 1 Data Direction Register (P1DDR) ..... 216
8.1.2 Port 1 Data Register (P1DR) ..... 217
8.1.3 Port 1 Register (PORT1) ..... 217
8.1.4 Pin Functions ..... 218
8.2 Port 3 ..... 223
8.2.1 Port 3 Data Direction Register (P3DDR) ..... 223
8.2.2 Port 3 Data Register (P3DR) ..... 224
8.2.3 Port 3 Register (PORT3) ..... 224
8.2.4 Port 3 Open-Drain Control Register (P3ODR) ..... 225
8.2.5 Pin Functions ..... 225
8.3 Port 4 ..... 227
8.3.1 Port 4 Register (PORT4) ..... 227
8.3.2 Pin Function ..... 227
8.4 Port 7 ..... 228
8.4.1 Port 7 Data Direction Register (P7DDR) ..... 228
8.4.2 Port 7 Data Register (P7DR) ..... 229
8.4.3 Port 7 Register (PORT7) ..... 230
8.4.4 Pin Functions ..... 231
8.5 Port 9 ..... 232
8.5.1 Port 9 Register (PORT9) ..... 232
8.5.2 Pin Function ..... 232
8.6 Port A ..... 233
8.6.1 Port A Data Direction Register (PADDR) ..... 233
8.6.2 Port A Data Register (PADR) ..... 234
8.6.3 Port A Register (PORTA) ..... 234
8.6.4 Port A Pull-Up MOS Control Register (PAPCR) ..... 235
8.6.5 Port A Open-Drain Control Register (PAODR) ..... 235
8.6.6 Pin Functions ..... 236
8.6.7 Port A Input Pull-Up MOS States ..... 238
8.7 Port B (H8S/2218 Group Only) ..... 239
8.7.1 Port B Data Direction Register (PBDDR) ..... 239
8.7.2 Port B Data Register (PBDR) ..... 240

8.7.3 Port B Register (PORTB) ... 240
8.7.4 Port B Pull-Up MOS Control Register (PBPCR) ... 241
8.7.5 Pin Functions ... 242
8.7.6 Port B Input Pull-Up MOS States ... 244
8.8 Port C (H8S/2218 Group Only) ... 245
8.8.1 Port C Data Direction Register (PCDDR) ... 245
8.8.2 Port C Data Register (PCDR) ... 246
8.8.3 Port C Register (PORTC) ... 246
8.8.4 Port C Pull-Up MOS Control Register (PCPCR) ... 247
8.8.5 Pin Functions ... 247
8.8.6 Port C Input Pull-Up MOS States ... 249
8.9 Port D (H8S/2218 Group Only) ... 250
8.9.1 Port D Data Direction Register (PDDDR) ... 250
8.9.2 Port D Data Register (PDDR) ... 251
8.9.3 Port D Register (PORTD) ... 251
8.9.4 Port D Pull-Up MOS Control Register (PDPCR) ... 252
8.9.5 Pin Functions ... 252
8.9.6 Port D Input Pull-Up MOS States ... 254
8.10 Port E ... 255
8.10.1 Port E Data Direction Register (PEDDR) ... 255
8.10.2 Port E Data Register (PEDR) ... 256
8.10.3 Port E Register (PORTE) ... 256
8.10.4 Port E Pull-Up MOS Control Register (PEPCR) ... 257
8.10.5 Pin Functions ... 257
8.10.6 Port E Input Pull-Up MOS States ... 260
8.11 Port F ... 261
8.11.1 Port F Data Direction Register (PFDDR) ... 262
8.11.2 Port F Data Register (PFDR) ... 263
8.11.3 Port F Register (PORTF) ... 263
8.11.4 Clock Output Control Register (OUTCR) ... 264
8.11.5 Pin Functions ... 264
8.12 Port G ... 267
8.12.1 Port G Data Direction Register (PGDDR) ... 268
8.12.2 Port G Data Register (PGDR) ... 269
8.12.3 Port G Register (PORTG) ... 269
8.12.4 Pin Functions ... 270
8.13 Handling of Unused Pins ... 271

Section 9 16-Bit Timer Pulse Unit (TPU) ... 273
9.1 Features ... 273
9.2 Input/Output Pins ... 277

9.3 Register Descriptions ..... 278
9.3.1 Timer Control Register (TCR) ..... 279
9.3.2 Timer Mode Register (TMDR) ..... 282
9.3.3 Timer I/O Control Register (TIOR) ..... 284
9.3.4 Timer Interrupt Enable Register (TIER) ..... 293
9.3.5 Timer Status Register (TSR) ..... 294
9.3.6 Timer Counter (TCNT) ..... 297
9.3.7 Timer General Register (TGR) ..... 297
9.3.8 Timer Start Register (TSTR) ..... 297
9.3.9 Timer Synchro Register (TSYR) ..... 298
9.4 Interface to Bus Master ..... 299
9.4.1 16-Bit Registers ..... 299
9.4.2 8-Bit Registers ..... 299
9.5 Operation ..... 301
9.5.1 Basic Functions ..... 301
9.5.2 Synchronous Operation ..... 307
9.5.3 Buffer Operation ..... 309
9.5.4 PWM Modes ..... 313
9.5.5 Phase Counting Mode ..... 317
9.6 Interrupts ..... 322
9.6.1 Interrupt Source and Priority ..... 322
9.6.2 DMAC Activation ..... 323
9.6.3 A/D Converter Activation ..... 323
9.7 Operation Timing ..... 324
9.7.1 Input/Output Timing ..... 324
9.7.2 Interrupt Signal Timing ..... 327
9.8 Usage Notes ..... 331

Section 10 Watchdog Timer (WDT) ..... 339
10.1 Features ..... 339
10.2 Register Descriptions ..... 340
10.2.1 Timer Counter (TCNT) ..... 340
10.2.2 Timer Control/Status Register (TCSR) ..... 340
10.2.3 Reset Control/Status Register (RSTCSR) ..... 342
10.3 Operation ..... 343
10.3.1 Watchdog Timer Mode ..... 343
10.3.2 Timing of Setting of Watchdog Timer Overflow Flag (WOVF) ..... 344
10.3.3 Interval Timer Mode ..... 344
10.3.4 Timing of Setting of Overflow Flag (OVF) ..... 345
10.4 Interrupts ..... 345
10.5 Usage Notes ..... 346

10.5.1 Notes on Register Access ..... 346
10.5.2 Contention between Timer Counter (TCNT) Write and Increment ..... 347
10.5.3 Changing Value of CKS2 to CKS0 ..... 348
10.5.4 Switching between Watchdog Timer Mode and Interval Timer Mode ..... 348
10.5.5 Internal Reset in Watchdog Timer Mode ..... 348
10.5.6 OVF Flag Clearing in Interval Timer Mode ..... 348

Section 11 Realtime Clock (RTC) ..... 349
11.1 Features ..... 349
11.2 Input/Output Pin ..... 350
11.3 Register Descriptions ..... 350
11.3.1 Second Data Register (RSECDR) ..... 350
11.3.2 Minute Data Register (RMINDR) ..... 351
11.3.3 Hour Data Register (RHRDR) ..... 352
11.3.4 Day-of-Week Data Register (RWKDR) ..... 353
11.3.5 RTC Control Register 1 (RTCCR1) ..... 354
11.3.6 RTC Control Register 2 (RTCCR2) ..... 355
11.3.7 Clock Source Select Register (RTCCSR) ..... 356
11.3.8 Extended Module Stop Register (EXMDLSTP) ..... 357
11.4 Operation ..... 358
11.4.1 Initial Settings of Registers after Power-On and Resetting Procedure ..... 358
11.4.2 Time Data Reading Procedure ..... 359
11.5 Interrupt Source ..... 360
11.6 Operating State in Each Mode ..... 361
11.7 Usage Notes ..... 362

Section 12 Serial Communication Interface ..... 363
12.1 Features ..... 363
12.1.1 Block Diagram ..... 365
12.2 Input/Output Pins ..... 367
12.3 Register Descriptions ..... 367
12.3.1 Receive Shift Register (RSR) ..... 368
12.3.2 Receive Data Register (RDR) ..... 368
12.3.3 Transmit Data Register (TDR) ..... 368
12.3.4 Transmit Shift Register (TSR) ..... 368
12.3.5 Serial Mode Register (SMR) ..... 369
12.3.6 Serial Control Register (SCR) ..... 373
12.3.7 Serial Status Register (SSR) ..... 377
12.3.8 Smart Card Mode Register (SCMR) ..... 383
12.3.9 Serial Extended Mode Register A_0 (SEMRA_0) ..... 384
12.3.10 Serial Extended Mode Register B_0 (SEMRB_0) ..... 386

12.3.11 Bit Rate Register (BRR) ..... 395
12.4 Operation in Asynchronous Mode ..... 403
12.4.1 Data Transfer Format ..... 404
12.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode 405
12.4.3 Clock ..... 406
12.4.4 SCI Initialization (Asynchronous Mode) ..... 407
12.4.5 Data Transmission (Asynchronous Mode) ..... 408
12.4.6 Serial Data Reception (Asynchronous Mode) ..... 410
12.5 Multiprocessor Communication Function ..... 413
12.5.1 Multiprocessor Serial Data Transmission ..... 415
12.5.2 Multiprocessor Serial Data Reception ..... 416
12.6 Operation in Clocked Synchronous Mode ..... 419
12.6.1 Clock ..... 419
12.6.2 SCI Initialization (Clocked Synchronous Mode) ..... 420
12.6.3 Serial Data Transmission (Clocked Synchronous Mode) ..... 421
12.6.4 Serial Data Reception (Clocked Synchronous Mode) ..... 424
12.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode) ..... 425
12.7 Operation in Smart Card Interface ..... 427
12.7.1 Pin Connection Example ..... 427
12.7.2 Data Format (Except for Block Transfer Mode) ..... 428
12.7.3 Clock ..... 429
12.7.4 Block Transfer Mode ..... 429
12.7.5 Receive Data Sampling Timing and Reception Margin ..... 430
12.7.6 Initialization ..... 431
12.7.7 Serial Data Transmission (Except for Block Transfer Mode) ..... 432
12.7.8 Serial Data Reception (Except for Block Transfer Mode) ..... 435
12.7.9 Clock Output Control ..... 436
12.8 SCI Select Function (Clocked Synchronous Mode) ..... 438
12.9 Interrupts ..... 440
12.9.1 Interrupts in Normal Serial Communication Interface Mode ..... 440
12.9.2 Interrupts in Smart Card Interface Mode ..... 441
12.10 Usage Notes ..... 441
12.10.1 Module Stop Mode Setting ..... 441
12.10.2 Break Detection and Processing (Asynchronous Mode Only) ..... 441
12.10.3 Mark State and Break Detection (Asynchronous Mode Only) ..... 442
12.10.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only) ..... 442
12.10.5 Restrictions on Use of DMAC ..... 442
12.10.6 Operation in Case of Mode Transition ..... 443
12.10.7 Switching from SCK Pin Function to Port Pin Function: ..... 447

Section 13 Boundary Scan Function ..... 449
13.1 Features ..... 449
13.2 Pin Configuration ..... 451
13.3 Register Descriptions ..... 452
13.3.1 Instruction Register (INSTR) ..... 452
13.3.2 IDCODE Register (IDCODE) ..... 454
13.3.3 BYPASS Register (BYPASS) ..... 454
13.3.4 Boundary Scan Register (BSCANR) ..... 454
13.4 Boundary Scan Function Operation ..... 462
13.4.1 TAP Controller ..... 462
13.5 Usage Notes ..... 463

Section 14 Universal Serial Bus (USB) ..... 465
14.1 Features ..... 465
14.2 Input/Output Pins ..... 467
14.3 Register Descriptions ..... 467
14.3.1 USB Control Register (UCTLR) ..... 468
14.3.2 USB DMAC Transfer Request Register (UDMAR) ..... 471
14.3.3 USB Device Resume Register (UDRR) ..... 472
14.3.4 USB Trigger Register 0 (UTRG0) ..... 473
14.3.5 USB FIFO Clear Register 0 (UFCLR0) ..... 475
14.3.6 USB Endpoint Stall Register 0 (UESTL0) ..... 476
14.3.7 USB Endpoint Stall Register 1 (UESTL1) ..... 477
14.3.8 USB Endpoint Data Register 0s (UEDR0s) ..... 477
14.3.9 USB Endpoint Data Register 0i (UEDR0i) ..... 477
14.3.10 USB Endpoint Data Register 0o (UEDR0o) ..... 478
14.3.11 USB Endpoint Data Register 3 (UEDR3) ..... 478
14.3.12 USB Endpoint Data Register 1 (UEDR1) ..... 478
14.3.13 USB Endpoint Data Register 2 (UEDR2) ..... 479
14.3.14 USB Endpoint Receive Data Size Register 0o (UESZ0o) ..... 479
14.3.15 USB Endpoint Receive Data Size Register 2 (UESZ2) ..... 479
14.3.16 USB Interrupt Flag Register 0 (UIFR0) ..... 480
14.3.17 USB Interrupt Flag Register 1 (UIFR1) ..... 482
14.3.18 USB Interrupt Flag Register 3 (UIFR3) ..... 483
14.3.19 USB Interrupt Enable Register 0 (UIER0) ..... 484
14.3.20 USB Interrupt Enable Register 1 (UIER1) ..... 485
14.3.21 USB Interrupt Enable Register 3 (UIER3) ..... 485
14.3.22 USB Interrupt Select Register 0 (UISR0) ..... 486
14.3.23 USB Interrupt Select Register 1 (UISR1) ..... 486
14.3.24 USB Interrupt Select Register 3 (UISR3) ..... 487
14.3.25 USB Data Status Register (UDSR) ..... 487

14.3.26 USB Configuration Value Register (UCVR) ... 488
14.3.27 USB Test Register 0 (UTSTR0) ... 489
14.3.28 USB Test Register 1 (UTSTR1) ... 490
14.3.29 USB Test Registers 2 and A to F (UTSTR2, UTSTRA to UTSTRF) ... 492
14.3.30 Module Stop Control Register B (MSTPCRB) ... 492
14.3.31 Extended Module Stop Register (EXMDLSTP) ... 493
14.4 Interrupt Sources ... 494
14.5 Communication Operation ... 497
14.5.1 Initialization ... 497
14.5.2 USB Cable Connection/Disconnection ... 498
14.5.3 Suspend and Resume Operations ... 502
14.5.4 Control Transfer ... 506
14.5.5 Interrupt-In Transfer (Endpoint 3) ... 512
14.5.6 Bulk-In Transfer (Dual FIFOs) (Endpoint 1) ... 513
14.5.7 Bulk-Out Transfer (Dual FIFOs) (Endpoint 2) ... 515
14.5.8 Processing of USB Standard Commands and Class/Vendor Commands ... 516
14.5.9 Stall Operations ... 517
14.6 DMA Transfer Specifications ... 520
14.6.1 DMAC Transfer by USB Request ... 520
14.6.2 DMA Transfer by Auto-Request ... 522
14.7 USB External Circuit Example ... 525
14.8 Usage Notes ... 527
14.8.1 Emulator Usage Notes ... 527
14.8.2 Bus Interface ... 527
14.8.3 Operating Frequency ... 527
14.8.4 Setup Data Reception ... 528
14.8.5 FIFO Clear ... 528
14.8.6 $\overline{\text{IRQ6}}$ Interrupt ... 528
14.8.7 Data Register Overread or Overwrite ... 528
14.8.8 Reset ... 529
14.8.9 EP0 Interrupt Sources Assignment ... 529
14.8.10 Level Shifter for VBUS and $\overline{\text{IRQx}}$ Pins ... 530
14.8.11 USB Endpoint Data Read and Write ... 530
14.8.12 Restrictions on Entering and Canceling Power-Down Mode ... 530
14.8.13 USB External Circuit Example ... 532
14.8.14 Pin Processing when USB Not Used ... 533
14.8.15 Notes on TR Interrupt ... 533
14.8.16 Clearing the FIFO when DMA Transfer Is Enabled ... 533

Section 15 A/D Converter ... 535
15.1 Features ... 535

15.2 Input/Output Pins ... 537
15.3 Register Descriptions ... 537
15.3.1 A/D Data Registers A to D (ADDRA to ADDRD) ... 538
15.3.2 A/D Control/Status Register (ADCSR) ... 538
15.3.3 A/D Control Register (ADCR) ... 540
15.4 Interface to Bus Master ... 541
15.5 Operation ... 542
15.5.1 Single Mode ... 542
15.5.2 Scan Mode ... 543
15.5.3 Input Sampling and A/D Conversion Time ... 544
15.5.4 External Trigger Input Timing ... 545
15.6 Interrupts ... 546
15.7 A/D Conversion Precision Definitions ... 546
15.8 Usage Notes ... 548
15.8.1 Module Stop Mode Setting ... 548
15.8.2 Permissible Signal Source Impedance ... 548
15.8.3 Influences on Absolute Precision ... 548
15.8.4 Range of Analog Power Supply and Other Pin Settings ... 549
15.8.5 Notes on Board Design ... 549

Section 16 RAM ... 551

Section 17 Flash Memory (F-ZTAT Version) ... 553
17.1 Features ... 553
17.2 Mode Transitions ... 555
17.3 Block Configuration ... 558
17.4 Input/Output Pins ... 561
17.5 Register Descriptions ... 561
17.5.1 Flash Memory Control Register 1 (FLMCR1) ... 562
17.5.2 Flash Memory Control Register 2 (FLMCR2) ... 563
17.5.3 Erase Block Register 1 (EBR1) ... 564
17.5.4 Erase Block Register 2 (EBR2) ... 564
17.5.5 RAM Emulation Register (RAMER) ... 565
17.5.6 Serial Control Register X (SCRX) ... 566
17.6 On-Board Programming Modes ... 567
17.6.1 SCI Boot Mode (HD64F2218, HD64F2212, and HD64F2211) ... 567
17.6.2 USB Boot Mode (HD64F2218U, HD64F2212U, and HD64F2211U) ... 571
17.6.3 Programming/Erasing in User Program Mode ... 576
17.7 Flash Memory Emulation in RAM ... 577
17.8 Flash Memory Programming/Erasing ... 579
17.8.1 Program/Program-Verify ... 579

17.8.2 Erase/Erase-Verify ... 581
17.9 Program/Erase Protection ... 583
17.9.1 Hardware Protection ... 583
17.9.2 Software Protection ... 583
17.9.3 Error Protection ... 583
17.10 Interrupt Handling when Programming/Erasing Flash Memory ... 584
17.11 Programmer Mode ... 585
17.12 Power-Down States for Flash Memory ... 586
17.13 Flash Memory Programming and Erasing Precautions ... 587
17.14 Note on Switching from F-ZTAT Version to Masked ROM Version ... 592

Section 18 Masked ROM ... 593
18.1 Features ... 593

Section 19 Clock Pulse Generator ... 595
19.1 Register Descriptions ... 596
19.1.1 System Clock Control Register (SCKCR) ... 596
19.1.2 Low Power Control Register (LPWRCR) ... 597
19.2 System Clock Oscillator ... 600
19.2.1 Connecting a Crystal Resonator ... 600
19.2.2 Inputting External Clock ... 601
19.3 Duty Adjustment Circuit ... 602
19.4 Medium-Speed Clock Divider ... 602
19.5 Bus Master Clock Selection Circuit ... 602
19.6 Subclock Oscillator ... 603
19.6.1 Connecting 32.768-kHz Crystal Resonator ... 603
19.6.2 Handling Pins when Subclock Not Required ... 603
19.7 Subclock Waveform Generation Circuit ... 604
19.8 PLL Circuit for USB ... 604
19.9 Usage Notes ... 605
19.9.1 Note on Crystal Resonator ... 605
19.9.2 Note on Board Design ... 605
19.9.3 Note on Switchover of External Clock ... 605

Section 20 Power-Down Modes ... 607
20.1 Register Descriptions ... 611
20.1.1 Standby Control Register (SBYCR) ... 611
20.1.2 Timer Control/Status Register (TCSR_1) ... 613
20.1.3 Module Stop Control Registers A to C (MSTPCRA to MSTPCRC) ... 614
20.1.4 Extended Module Stop Register (EXMDLSTP) ... 617
20.2 Medium-Speed Mode ... 617

20.3 Sleep Mode ... 618
20.3.1 Transition to Sleep Mode ... 618
20.3.2 Exiting Sleep Mode ... 618
20.4 Software Standby Mode ... 619
20.4.1 Transition to Software Standby Mode ... 619
20.4.2 Clearing Software Standby Mode ... 619
20.4.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode ... 620
20.4.4 Software Standby Mode Application Example ... 620
20.5 Hardware Standby Mode ... 621
20.5.1 Transition to Hardware Standby Mode ... 621
20.5.2 Clearing Hardware Standby Mode ... 621
20.5.3 Hardware Standby Mode Timing ... 622
20.5.4 Hardware Standby Mode Timings ... 622
20.6 Module Stop Mode ... 623
20.7 Watch Mode ... 624
20.7.1 Transition to Watch Mode ... 624
20.7.2 Exiting Watch Mode ... 624
20.8 Subsleep Mode ... 625
20.8.1 Transition to Sleep Mode ... 625
20.8.2 Exiting Subsleep Mode ... 625
20.9 Subactive Mode ... 626
20.9.1 Transition to Subactive Mode ... 626
20.9.2 Exiting Subactive Mode ... 626
20.10 Direct Transitions ... 627
20.10.1 Direct Transitions from High-Speed Mode to Subactive Mode ... 627
20.10.2 Direct Transitions from Subactive Mode to High-Speed Mode ... 627
20.11 φ Clock Output Disabling Function ... 627
20.12 Usage Notes ... 628
20.12.1 I/O Port Status ... 628
20.12.2 Current Dissipation during Oscillation Stabilization Wait Period ... 628
20.12.3 Flash Memory Module Stop ... 628
20.12.4 DMAC Module Stop ... 628
20.12.5 On-Chip Peripheral Module Interrupt ... 628
20.12.6 Entering Subactive/Watch Mode and DMAC and DTC Module Stop ... 629
20.12.7 Writing to MSTPCR ... 629

Section 21 List of Registers ... 631
21.1 Register Addresses (Address Order) ... 632
21.2 Register Bits ... 640
21.3 Register States in Each Operating Mode ... 649

Section 22 Electrical Characteristics ..... 657
22.1 Absolute Maximum Ratings ..... 657
22.2 Power Supply Voltage and Operating Frequency Range ..... 658
22.3 DC Characteristics ..... 659
22.4 AC Characteristics ..... 663
22.4.1 Clock Timing ..... 664
22.4.2 Control Signal Timing ..... 666
22.4.3 Bus Timing ..... 668
22.4.4 Timing of On-Chip Supporting Modules ..... 675
22.5 USB Characteristics ..... 680
22.6 A/D Conversion Characteristics ..... 682
22.7 Flash Memory Characteristics ..... 683
22.8 Usage Note ..... 684

Appendix ..... 685
A. I/O Port States in Each Processing State ..... 685
B. Product Model Lineup ..... 689
C. Package Dimensions ..... 691

Index ..... 695

# Figures

## Section 1 Overview


Figure 1.1 Internal Block Diagram of HD64F2218, HD64F2218U, HD64F2218CU and HD642217CU ..... 3
Figure 1.2 Internal Block Diagram of HD6432217 ..... 4
Figure 1.3 Internal Block Diagram of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU ..... 5
Figure 1.4 Internal Block Diagram of HD6432211, HD6432210 and HD6432210S ..... 6
Figure 1.5 Pin Arrangements of HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU (TFP-100G, TFP-100GV) ..... 7
Figure 1.6 Pin Arrangements of HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU (BP-112, BP-112V) ..... 8
Figure 1.7 Pin Arrangements of HD6432217 (TFP-100G, TFP-100GV) ..... 9
Figure 1.8 Pin Arrangements of HD6432217 (BP-112, BP-112V) ..... 10
Figure 1.9 Pin Arrangements of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU (FP-64E, FP-64EV) ..... 11
Figure 1.10 Pin Arrangements of HD6432211, HD6432210 and HD6432210S (FP-64E, FP-64EV) ..... 12
Figure 1.11 Pin Arrangements of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU (TNP-64B, TNP-64BV) ..... 13
Figure 1.12 Pin Arrangements of HD6432211, HD6432210 and HD6432210S (TNP-64B, TNP-64BV) ..... 14


## Section 2 CPU


Figure 2.1 Exception Vector Table (Normal Mode) ..... 35
Figure 2.2 Stack Structure in Normal Mode ..... 35
Figure 2.3 Exception Vector Table (Advanced Mode) ..... 36
Figure 2.4 Stack Structure in Advanced Mode ..... 37
Figure 2.5 Memory Map ..... 38
Figure 2.6 CPU Registers ..... 39
Figure 2.7 Usage of General Registers ..... 40
Figure 2.8 Stack ..... 41
Figure 2.9 General Register Data Formats (1) ..... 44
Figure 2.9 General Register Data Formats (2) ..... 44
Figure 2.10 Memory Data Formats ..... 45
Figure 2.11 Instruction Formats (Examples) ..... 57
Figure 2.12 Branch Address Specification in Memory Indirect Mode ..... 61
Figure 2.13 State Transitions ..... 65
Figure 2.14 Flowchart of Method for Accessing Registers Containing Write-Only Bits ..... 69

Section 3 MCU Operating Modes
Figure 3.1 Memory Map in Each Operating Mode for HD64F2218, HD64F2218U and HD64F2218CU ..... 77
Figure 3.2 Memory Map in Each Operating Mode for HD64F2217CU ..... 78
Figure 3.3 Memory Map in Each Operating Mode for HD6432217 ..... 79
Figure 3.4 Memory Map in Each Operating Mode for HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU, HD64F2210CU, HD6432211, HD6432210 and HD6432210S ..... 80

Section 4 Exception Handling
Figure 4.1 Reset Sequence (Mode 4) ..... 85
Figure 4.2 Reset Sequence (Modes 6 and 7) ..... 86
Figure 4.3 Stack Status after Exception Handling ..... 89
Figure 4.4 Operation when SP Value Is Odd ..... 90

Section 5 Interrupt Controller
Figure 5.1 Block Diagram of Interrupt Controller ..... 92
Figure 5.2 Block Diagram of Interrupts IRQn ..... 99
Figure 5.3 Timing of Setting IRQnF ..... 100
Figure 5.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0 ..... 104
Figure 5.5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2 ..... 106
Figure 5.6 Interrupt Exception Handling ..... 107
Figure 5.7 Interrupt Control for DMAC ..... 110
Figure 5.8 Contention between Interrupt Generation and Disabling ..... 112

Section 6 Bus Controller
Figure 6.1 Block Diagram of Bus Controller ..... 116
Figure 6.2 Overview of Area Divisions ..... 127
Figure 6.3 $\overline{\text{CSn}}$ Signal Output Timing (n = 0 to 5) ..... 130
Figure 6.4 On-Chip Memory Access Cycle ..... 131
Figure 6.5 Pin States during On-Chip Memory Access ..... 132
Figure 6.6 On-Chip Peripheral Module Access Cycle ..... 132
Figure 6.7 Pin States during On-Chip Peripheral Module Access ..... 133
Figure 6.8 Access Sizes and Data Alignment Control (8-Bit Access Space) ..... 134
Figure 6.9 Access Sizes and Data Alignment Control (16-Bit Access Space) ..... 135
Figure 6.10 Bus Timing for 8-Bit 2-State Access Space ..... 136
Figure 6.11 Bus Timing for 8-Bit 3-State Access Space (Except Area 6) ..... 137
Figure 6.12 Bus Timing for Area 6 and RTC ..... 138
Figure 6.13 Bus Timing for 16-Bit 2-State Access Space (1) (Even Address Byte Access) ..... 139

Figure 6.14 Bus Timing for 16-Bit 2-State Access Space (2) (Odd Address Byte Access) ..... 140
Figure 6.15 Bus Timing for 16-Bit 2-State Access Space (3) (Word Access)..... 141
Figure 6.16 Bus Timing for 16-Bit 3-State Access Space (1) (Even Address Byte Access)..... 142
Figure 6.17 Bus Timing for 16-Bit 3-State Access Space (2) (Odd Address Byte Access) ..... 143
Figure 6.18 Bus Timing for 16-Bit 3-State Access Space (3) (Word Access)..... 144
Figure 6.19 Example of Wait State Insertion Timing..... 146
Figure 6.20 Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 1)..... 148
Figure 6.21 Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 0)..... 148
Figure 6.22 Example of Idle Cycle Operation (1) ..... 150
Figure 6.23 Example of Idle Cycle Operation (2) ..... 151
Figure 6.24 Relationship between Chip Select ($\overline{CS}$) and Read ($\overline{RD}$)..... 152
Figure 6.25 Bus-Released State Transition Timing..... 154

## Section 7 DMA Controller (DMAC)

Figure 7.1 Block Diagram of DMAC ..... 158
Figure 7.2 Operation in Sequential Mode..... 179
Figure 7.3 Example of Sequential Mode Setting Procedure..... 180
Figure 7.4 Operation in Idle Mode ..... 181
Figure 7.5 Example of Idle Mode Setting Procedure..... 182
Figure 7.6 Operation in Repeat mode..... 184
Figure 7.7 Example of Repeat Mode Setting Procedure..... 185
Figure 7.8 Operation in Normal Mode ..... 187
Figure 7.9 Example of Normal Mode Setting Procedure..... 188
Figure 7.10 Operation in Block Transfer Mode (BLKDIR = 0)..... 190
Figure 7.11 Operation in Block Transfer Mode (BLKDIR = 1)..... 191
Figure 7.12 Operation Flow in Block Transfer Mode ..... 192
Figure 7.13 Example of Block Transfer Mode Setting Procedure..... 193
Figure 7.14 Example of DMA Transfer Bus Timing..... 196
Figure 7.15 Example of Short Address Mode Transfer..... 197
Figure 7.16 Example of Full Address Mode (Cycle Steal) Transfer ..... 198
Figure 7.17 Example of Full Address Mode (Burst Mode) Transfer..... 199
Figure 7.18 Example of Full Address Mode (Block Transfer Mode) Transfer ..... 200
Figure 7.19 Example of $\overline{DREQ}$ Level Activated Normal Mode Transfer ..... 201
Figure 7.20 Example of Multi-Channel Transfer ..... 202
Figure 7.21 Example of Procedure for Continuing Transfer on Channel Interrupted by NMI Interrupt ..... 204
Figure 7.22 Example of Procedure for Forcibly Terminating DMAC Operation..... 204
Figure 7.23 Example of Procedure for Clearing Full Address Mode ..... 205
Figure 7.24 Block Diagram of Transfer End/Transfer Break Interrupt ..... 206
Figure 7.25 DMAC Register Update Timing ..... 207
Figure 7.26 Contention between DMAC Register Update and CPU Read..... 208

Section 9 16-Bit Timer Pulse Unit (TPU)


Figure 9.1 Block Diagram of TPU ..... 274
Figure 9.2 16-Bit Register Access Operation [Bus Master ↔ TCNT (16 Bits)] ..... 299
Figure 9.3 8-Bit Register Access Operation [Bus Master ↔ TCR (Upper 8 Bits)] ..... 300
Figure 9.4 8-Bit Register Access Operation [Bus Master ↔ TMDR (Lower 8 Bits)] ..... 300
Figure 9.5 8-Bit Register Access Operation [Bus Master ↔ TCR and TMDR (16 Bits)] ..... 300
Figure 9.6 Example of Counter Operation Setting Procedure ..... 301
Figure 9.7 Free-Running Counter Operation ..... 302
Figure 9.8 Periodic Counter Operation ..... 303
Figure 9.9 Example of Setting Procedure for Waveform Output by Compare Match ..... 303
Figure 9.10 Example of 0 Output/1 Output Operation ..... 304
Figure 9.11 Example of Toggle Output Operation ..... 304
Figure 9.12 Example of Input Capture Operation Setting Procedure ..... 305
Figure 9.13 Example of Input Capture Operation ..... 306
Figure 9.14 Example of Synchronous Operation Setting Procedure ..... 307
Figure 9.15 Example of Synchronous Operation ..... 308
Figure 9.16 Compare Match Buffer Operation ..... 309
Figure 9.17 Input Capture Buffer Operation ..... 309
Figure 9.18 Example of Buffer Operation Setting Procedure ..... 310
Figure 9.19 Example of Buffer Operation (1) ..... 311
Figure 9.20 Example of Buffer Operation (2) ..... 312
Figure 9.21 Example of PWM Mode Setting Procedure ..... 314
Figure 9.22 Example of PWM Mode Operation (1) ..... 315
Figure 9.23 Example of PWM Mode Operation (2) ..... 315
Figure 9.24 Example of PWM Mode Operation (3) ..... 316
Figure 9.25 Example of Phase Counting Mode Setting Procedure ..... 317
Figure 9.26 Example of Phase Counting Mode 1 Operation ..... 318
Figure 9.27 Example of Phase Counting Mode 2 Operation ..... 319
Figure 9.28 Example of Phase Counting Mode 3 Operation ..... 320
Figure 9.29 Example of Phase Counting Mode 4 Operation ..... 321
Figure 9.30 Count Timing in Internal Clock Operation ..... 324
Figure 9.31 Count Timing in External Clock Operation ..... 324
Figure 9.32 Output Compare Output Timing ..... 325
Figure 9.33 Input Capture Input Signal Timing ..... 325
Figure 9.34 Counter Clear Timing (Compare Match) ..... 326
Figure 9.35 Counter Clear Timing (Input Capture) ..... 326
Figure 9.36 Buffer Operation Timing (Compare Match) ..... 327
Figure 9.37 Buffer Operation Timing (Input Capture) ..... 327
Figure 9.38 TGI Interrupt Timing (Compare Match) ..... 328
Figure 9.39 TGI Interrupt Timing (Input Capture) ..... 328
Figure 9.40 TCIV Interrupt Setting Timing ..... 329

Figure 9.41 TCIU Interrupt Setting Timing ..... 329
Figure 9.42 Timing for Status Flag Clearing by CPU ..... 330
Figure 9.43 Timing for Status Flag Clearing by DMAC Activation ..... 330
Figure 9.44 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode ..... 331
Figure 9.45 Contention between TCNT Write and Clear Operations ..... 332
Figure 9.46 Contention between TCNT Write and Increment Operations ..... 332
Figure 9.47 Contention between TGR Write and Compare Match ..... 333
Figure 9.48 Contention between Buffer Register Write and Compare Match ..... 334
Figure 9.49 Contention between TGR Read and Input Capture ..... 334
Figure 9.50 Contention between TGR Write and Input Capture ..... 335
Figure 9.51 Contention between Buffer Register Write and Input Capture ..... 336
Figure 9.52 Contention between Overflow and Counter Clearing ..... 336
Figure 9.53 Contention between TCNT Write and Overflow ..... 337

## Section 10 Watchdog Timer (WDT)

Figure 10.1 Block Diagram of WDT ..... 340
Figure 10.2 Operation in Watchdog Timer Mode ..... 343
Figure 10.3 Timing of WOVF Setting ..... 344
Figure 10.4 Operation in Interval Timer Mode ..... 344
Figure 10.5 Timing of OVF Setting ..... 345
Figure 10.6 Format of Data Written to TCNT and TCSR ..... 346
Figure 10.7 Format of Data Written to RSTCSR (Example of WDT0) ..... 347
Figure 10.8 Contention between TCNT Write and Increment ..... 347

## Section 11 Realtime Clock (RTC)

Figure 11.1 Block Diagram of RTC ..... 349
Figure 11.2 Definition of Time Expression ..... 354
Figure 11.3 Initial Setting Procedure ..... 358
Figure 11.4 Example: Reading of Inaccurate Time Data ..... 359
Figure 11.5 Initializing Procedure in Using RTC Interrupt ..... 361
Figure 11.6 Example of RTC Interrupt Handling Routine ..... 361

## Section 12 Serial Communication Interface

Figure 12.1 Block Diagram of SCI_0 ..... 365
Figure 12.2 Block Diagram of SCI_2 ..... 366
Figure 12.3 Examples of Base Clock when Average Transfer Rate Is Selected (1) ..... 388
Figure 12.3 Examples of Base Clock when Average Transfer Rate Is Selected (2) ..... 389
Figure 12.3 Examples of Base Clock when Average Transfer Rate Is Selected (3) ..... 390
Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (1) ..... 391
Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (2) ..... 392
Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (3) ..... 393

Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (4) ..... 394
Figure 12.5 Data Format in Asynchronous Communication
(Example with 8-Bit Data, Parity, Two Stop Bits) ..... 403
Figure 12.6 Receive Data Sampling Timing in Asynchronous Mode ..... 405
Figure 12.7 Relationship between Output Clock and Transfer Data Phase
(Asynchronous Mode) ..... 406
Figure 12.8 Sample SCI Initialization Flowchart ..... 407
Figure 12.9 Example of Operation in Transmission in Asynchronous Mode
(Example with 8-Bit Data, Parity, One Stop Bit) ..... 408
Figure 12.10 Sample Serial Data Transmission Flowchart ..... 409
Figure 12.11 Example of SCI Operation in Reception
(Example with 8-Bit Data, Parity, One Stop Bit) ..... 410
Figure 12.12 Sample Serial Data Reception Flowchart (1) ..... 411
Figure 12.12 Sample Serial Data Reception Flowchart (2) ..... 412
Figure 12.13 Example of Communication Using Multiprocessor Format
(Transmission of Data H'AA to Receiving Station A) ..... 414
Figure 12.14 Sample Multiprocessor Serial Data Transmission Flowchart ..... 415
Figure 12.15 Example of SCI Operation in Reception
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit) ..... 416
Figure 12.16 Sample Multiprocessor Serial Data Reception Flowchart (1) ..... 417
Figure 12.16 Sample Multiprocessor Serial Data Reception Flowchart (2) ..... 418
Figure 12.17 Data Format in Synchronous Communication (For LSB-First) ..... 419
Figure 12.18 Sample SCI Initialization Flowchart ..... 420
Figure 12.19 Sample SCI Transmission Operation in Clocked Synchronous Mode ..... 422
Figure 12.20 Sample Serial Data Transmission Flowchart ..... 423
Figure 12.21 Example of SCI Operation in Reception ..... 424
Figure 12.22 Sample Serial Data Reception Flowchart ..... 425
Figure 12.23 Sample Flowchart of Simultaneous Serial Data Transmit
and Receive Operations ..... 426
Figure 12.24 Schematic Diagram of Smart Card Interface Pin Connections ..... 427
Figure 12.25 Normal Smart Card Interface Data Format ..... 428
Figure 12.26 Direct Convention (SDIR = SINV = O/$\overline{E}$ = 0) ..... 428
Figure 12.27 Inverse Convention (SDIR = SINV = O/$\overline{E}$ = 1) ..... 429
Figure 12.28 Receive Data Sampling Timing in Smart Card Mode
(Using Clock of 372 Times the Transfer Rate) ..... 430
Figure 12.29 Retransfer Operation in SCI Transmit Mode ..... 433
Figure 12.30 TEND Flag Generation Timing in Transmission Operation ..... 433
Figure 12.31 Example of Transmission Processing Flow ..... 434
Figure 12.32 Retransfer Operation in SCI Receive Mode ..... 435
Figure 12.33 Example of Reception Processing Flow ..... 436
Figure 12.34 Timing for Fixing Clock Output Level ..... 436

Figure 12.35 Clock Halt and Restart Procedure ..... 437
Figure 12.36 Example of Communication Using the SCI Select Function ..... 438
Figure 12.37 Example of Communication Using the SCI Select Function ..... 439
Figure 12.38 Example of Clocked Synchronous Transmission by DMAC ..... 442
Figure 12.39 Sample Flowchart for Mode Transition during Transmission ..... 444
Figure 12.40 Port Pin State of Asynchronous Transmission Using Internal Clock ..... 444
Figure 12.41 Port Pin State of Synchronous Transmission Using Internal Clock ..... 445
Figure 12.42 Sample Flowchart for Mode Transition during Reception ..... 446
Figure 12.43 Operation when Switching from SCK Pin Function to Port Pin Function ..... 447
Figure 12.44 Operation when Switching from SCK Pin Function to Port Pin Function (Example of Preventing Low-Level Output) ..... 448

## Section 13 Boundary Scan Function

Figure 13.1 Block Diagram of Boundary Scan Function ..... 450
Figure 13.2 Boundary Scan Register Configuration ..... 455
Figure 13.3 TAP Controller Status Transition ..... 462
Figure 13.4 Recommended Reset Signal Design ..... 463
Figure 13.5 Serial Data Input/Output ..... 463

## Section 14 Universal Serial Bus (USB)

Figure 14.1 Block Diagram of USB ..... 466
Figure 14.2 USB Initialization ..... 497
Figure 14.3 USB Cable Connection (When USB Module Stop or Power-Down Mode Is not Used) ..... 498
Figure 14.4 USB Cable Connection (When USB Module Stop or Power-Down Mode Is Used) ..... 499
Figure 14.5 USB Cable Disconnection (When USB Module Stop or Power-Down Mode Is not Used) ..... 500
Figure 14.6 USB Cable Disconnection (When USB Module Stop or Power-Down Mode Is Used) ..... 501
Figure 14.7 Example Flowchart of Suspend and Resume Operations ..... 502
Figure 14.8 Example Flowchart of Suspend and Resume Interrupt Processing ..... 503
Figure 14.9 Example Flowchart of Suspend and Remote-Wakeup Operations ..... 504
Figure 14.10 Example Flowchart of Remote-Wakeup Interrupt Processing ..... 505
Figure 14.11 Control Transfer Stage Configuration ..... 506
Figure 14.12 Setup Stage Operation ..... 507
Figure 14.13 Data Stage Operation (Control-In) ..... 508
Figure 14.14 Data Stage Operation (Control-Out) ..... 509
Figure 14.15 Status Stage Operation (Control-In) ..... 510
Figure 14.16 Status Stage Operation (Control-Out) ..... 511
Figure 14.17 EP3 Interrupt-In Transfer Operation ..... 512

Figure 14.18 EP1 Bulk-In Transfer Operation ..... 514
Figure 14.19 EP2 Bulk-Out Transfer Operation ..... 515
Figure 14.20 Forcible Stall by Firmware ..... 518
Figure 14.21 Automatic Stall by USB Function Module ..... 519
Figure 14.22 EP1PKTE Operation in UTRG0 ..... 521
Figure 14.23 EP2RDFN Operation in UTRG0 ..... 522
Figure 14.24 EP1PKTE Operation in UTRG0 (Auto-Request) ..... 523
Figure 14.25 EP2RDFN Operation in UTRG0 (Auto-Request) ..... 524
Figure 14.26 USB External Circuit in Bus-Powered Mode ..... 525
Figure 14.27 USB External Circuit in Self-Powered Mode ..... 526
Figure 14.28 Flowchart ..... 531
Figure 14.29 Timing Chart ..... 532
Figure 14.30 TR Interrupt Flag Set Timing ..... 533

## Section 15 A/D Converter

Figure 15.1 Block Diagram of A/D Converter ..... 536
Figure 15.2 Access to ADDR (When Reading H'AA40) ..... 541
Figure 15.3 A/D Conversion Timing (Single-Chip Mode, Channel 1 Selected) ..... 542
Figure 15.4 A/D Conversion Timing (Scan Mode, Channels AN0 to AN2 Selected) ..... 543
Figure 15.5 A/D Conversion Timing ..... 544
Figure 15.6 External Trigger Input Timing ..... 545
Figure 15.7 A/D Conversion Precision Definitions (1) ..... 547
Figure 15.8 A/D Conversion Precision Definitions (2) ..... 547
Figure 15.9 Example of Analog Input Circuit ..... 548
Figure 15.10 Analog Input Pin Equivalent Circuit ..... 549

## Section 17 Flash Memory (F-ZTAT Version)

Figure 17.1 Block Diagram of Flash Memory ..... 554
Figure 17.2 Flash Memory State Transitions ..... 555
Figure 17.3 Boot Mode (Sample) ..... 556
Figure 17.4 User Program Mode (Sample) ..... 557
Figure 17.5 Flash Memory Block Configuration (HD64F2218, HD64F2218U, HD64F2218CU, HD64F2212, HD64F2212U, HD64F2212CU) ..... 558
Figure 17.6 Flash Memory Block Configuration (HD64F2217CU, HD64F2211, HD64F2211U, HD64F2211CU) ..... 559
Figure 17.7 Flash Memory Block Configuration (HD64F2210CU) ..... 560
Figure 17.8 System Configuration in SCI Boot Mode ..... 568
Figure 17.9 System Configuration Diagram when Using USB Boot Mode ..... 572
Figure 17.10 Programming/Erasing Flowchart Example in User Program Mode ..... 576
Figure 17.11 Flowchart for Flash Memory Emulation in RAM ..... 577
Figure 17.12 Example of RAM Overlap Operation ..... 578

Figure 17.13 Program/Program-Verify Flowchart ..... 580
Figure 17.14 Erase/Erase-Verify Flowchart ..... 582
Figure 17.15 Memory Map in Programmer Mode ..... 585
Figure 17.16 Power-On/Off Timing (Boot Mode) ..... 589
Figure 17.17 Power-On/Off Timing (User Program Mode) ..... 590
Figure 17.18 Mode Transition Timing
(Example: Boot Mode → User Mode ↔ User Program Mode) ..... 591

## Section 18 Masked ROM

Figure 18.1 Block Diagram of On-Chip Masked ROM (64 kbytes) ..... 593

## Section 19 Clock Pulse Generator

Figure 19.1 Block Diagram of Clock Pulse Generator ..... 595
Figure 19.2 Connection of Crystal Resonator (Example) ..... 600
Figure 19.3 Crystal Resonator Equivalent Circuit ..... 600
Figure 19.4 External Clock Input (Examples) ..... 601
Figure 19.5 External Clock Input Timing ..... 602
Figure 19.6 Example Connection of 32.768-kHz Quartz Oscillator ..... 603
Figure 19.7 Equivalence Circuit for 32.768-kHz Oscillator ..... 603
Figure 19.8 Pin Handling when Subclock Not Required ..... 603
Figure 19.9 Example of PLL Circuit ..... 604
Figure 19.10 Note on Board Design of Oscillator Circuit ..... 605
Figure 19.11 Example of External Clock Switching Circuit ..... 606
Figure 19.12 Example of External Clock Switchover Timing ..... 606

## Section 20 Power-Down Modes

Figure 20.1 Mode Transition Diagram ..... 609
Figure 20.2 Example of Flash Memory Module Stop Mode Usage ..... 616
Figure 20.3 Medium-Speed Mode Transition and Clearance Timing ..... 618
Figure 20.4 Software Standby Mode Application Example ..... 621
Figure 20.5 Hardware Standby Mode Timing (Example) ..... 622
Figure 20.6 Timing of Transition to Hardware Standby Mode ..... 622
Figure 20.7 Timing of Recovery from Hardware Standby Mode ..... 623

## Section 2 Electrical Characteristics

Figure 22.1 Power Supply Voltage and Operating Ranges ..... 658
Figure 22.2 Output Load Circuit ..... 663
Figure 22.3 System Clock Timing ..... 665
Figure 22.4 Oscillation Stabilization Timing ..... 665
Figure 22.5 Reset Input Timing ..... 667
Figure 22.6 Interrupt Input Timing ..... 667

Figure 22.7 Basic Bus Timing (Two-State Access) ..... 670
Figure 22.8 Basic Bus Timing (Three-State Access) ..... 671
Figure 22.9 Basic Bus Timing (Three-State Access with One Wait State) ..... 672
Figure 22.10 Burst ROM Access Timing (Two-State Access) ..... 673
Figure 22.11 External Bus Release Timing ..... 674
Figure 22.12 I/O Port Input/Output Timing ..... 677
Figure 22.13 TPU Input/Output Timing ..... 677
Figure 22.14 TPU Clock Input Timing ..... 677
Figure 22.15 SCK Clock Input Timing ..... 678
Figure 22.16 SCI Input/Output Timing (Clock Synchronous Mode) ..... 678
Figure 22.17 A/D Converter External Trigger Input Timing ..... 678
Figure 22.18 Boundary Scan TCK Input Timing ..... 678
Figure 22.19 Boundary Scan $\overline{\text{TRST}}$ Input Timing (At Reset Hold) ..... 678
Figure 22.20 Boundary Scan Data Transmission Timing ..... 679
Figure 22.21 Data Signal Timing ..... 681
Figure 22.22 Test Load Circuit ..... 681

## Appendix

Figure C.1 TFP-100G and TFP-100GV Package Dimensions ..... 691
Figure C.2 BP-112 and BP-112V Package Dimensions ..... 692
Figure C.3 FP-64E and FP-64EV Package Dimensions ..... 693
Figure C.4 TNP-64B and TNP-64BV Package Dimensions ..... 694

# Tables

## Section 1 Overview


Table 1.1 Pin Functions in Each Operating Mode for H8S/2218 Group ..... 15
Table 1.2 Pin Functions in Each Operating Mode for H8S/2212 Group ..... 18


## Section 2 CPU


Table 2.1 Instruction Classification ..... 46
Table 2.2 Operation Notation ..... 47
Table 2.3 Data Transfer Instructions ..... 48
Table 2.4 Arithmetic Operations Instructions ..... 49
Table 2.5 Logic Operations Instructions ..... 50
Table 2.6 Shift Instructions ..... 51
Table 2.7 Bit Manipulation Instructions ..... 52
Table 2.8 Branch Instructions ..... 54
Table 2.9 System Control Instruction ..... 55
Table 2.10 Block Data Transfer Instruction ..... 56
Table 2.11 Addressing Modes ..... 58
Table 2.12 Absolute Address Access Ranges ..... 60
Table 2.13 Effective Address Calculation ..... 62


## Section 3 MCU Operating Modes


Table 3.1 MCU Operating Mode Selection ..... 71
Table 3.2 Pin Functions in Each Operating Mode ..... 76


## Section 4 Exception Handling


Table 4.1 Exception Types and Priority ..... 81
Table 4.2 Exception Handling Vector Table ..... 82
Table 4.3 Reset Types ..... 84
Table 4.4 Status of CCR and EXR after Trace Exception Handling ..... 87
Table 4.5 Status of CCR and EXR after Trap Instruction Exception Handling ..... 88


## Section 5 Interrupt Controller


Table 5.1 Pin Configuration ..... 93
Table 5.2 Interrupt Sources, Vector Addresses, and Interrupt Priorities ..... 101
Table 5.3 Interrupt Control Modes ..... 103
Table 5.4 Interrupt Response Times ..... 108
Table 5.5 Number of States in Interrupt Handling Routine Execution Statuses ..... 109
Table 5.6 Interrupt Source Selection and Clearing Control ..... 111

## Section 6 Bus Controller


Table 6.1 Pin Configuration ..... 117
Table 6.2 Bus Specifications for Each Area (Basic Bus Interface) ..... 129
Table 6.3 Data Buses Used and Valid Strobes ..... 135
Table 6.4 Pin States in Idle Cycle ..... 152
Table 6.5 Pin States in Bus Released State ..... 153


## Section 7 DMA Controller (DMAC)


Table 7.1 Short Address Mode and Full Address Mode (For 1 Channel: Example of Channel 0) ..... 160
Table 7.2 DMAC Transfer Modes ..... 177
Table 7.3 Register Functions in Sequential Mode ..... 178
Table 7.4 Register Functions in Idle Mode ..... 181
Table 7.5 Register Functions in Repeat Mode ..... 183
Table 7.6 Register Functions in Normal Mode ..... 186
Table 7.7 Register Functions in Block Transfer Mode ..... 189
Table 7.8 DMAC Activation Sources ..... 194
Table 7.9 DMAC Channel Priority Order ..... 202
Table 7.10 Interrupt Source Priority Order ..... 206


## Section 8 I/O Ports


Table 8.1 Port Functions of H8S/2218 Group ..... 211
Table 8.2 Port Functions of H8S/2212 Group ..... 214
Table 8.3 P17 Pin Function ..... 218
Table 8.4 P16 Pin Function ..... 218
Table 8.5 P15 Pin Function ..... 218
Table 8.6 P14 Pin Function ..... 219
Table 8.7 P13 Pin Function ..... 219
Table 8.8 P12 Pin Function ..... 219
Table 8.9 P11 Pin Function ..... 220
Table 8.10 P10 Pin Function ..... 220
Table 8.11 P17 Pin Function ..... 220
Table 8.12 P16 Pin Function ..... 221
Table 8.13 P15 Pin Function ..... 221
Table 8.14 P14 Pin Function ..... 221
Table 8.15 P13 Pin Function ..... 222
Table 8.16 P12 Pin Function ..... 222
Table 8.17 P11 Pin Function ..... 222
Table 8.18 P10 Pin Function ..... 222
Table 8.19 P36 Pin Function ..... 225
Table 8.20 P32 Pin Function ..... 226

Table 8.21 P31 Pin Function ..... 226
Table 8.22 P30 Pin Function ..... 226
Table 8.23 P74 Pin Function ..... 231
Table 8.24 P71 Pin Function ..... 231
Table 8.25 P70 Pin Function ..... 231
Table 8.26 P77 Pin Function ..... 231
Table 8.27 P76 Pin Function ..... 232
Table 8.28 P75 Pin Function ..... 232
Table 8.29 PA3 Pin Function ..... 236
Table 8.30 PA2 Pin Function ..... 236
Table 8.31 PA1 Pin Function ..... 237
Table 8.32 PA0 Pin Function ..... 237
Table 8.33 PA3 Pin Function ..... 237
Table 8.34 PA2 Pin Function ..... 238
Table 8.35 PA1 Pin Function ..... 238
Table 8.36 Input Pull-Up MOS States (Port A) ..... 238
Table 8.37 PB7 Pin Function ..... 242
Table 8.38 PB6 Pin Function ..... 242
Table 8.39 PB5 Pin Function ..... 242
Table 8.40 PB4 Pin Function ..... 243
Table 8.41 PB3 Pin Function ..... 243
Table 8.42 PB2 Pin Function ..... 243
Table 8.43 PB1 Pin Function ..... 243
Table 8.44 PB0 Pin Function ..... 244
Table 8.45 Input Pull-Up MOS States (Port B) ..... 244
Table 8.46 PC7 Pin Function ..... 247
Table 8.47 PC6 Pin Function ..... 247
Table 8.48 PC5 Pin Function ..... 248
Table 8.49 PC4 Pin Function ..... 248
Table 8.50 PC3 Pin Function ..... 248
Table 8.51 PC2 Pin Function ..... 248
Table 8.52 PC1 Pin Function ..... 248
Table 8.53 PC0 Pin Function ..... 249
Table 8.54 Input Pull-Up MOS States (Port C) ..... 249
Table 8.55 PD7 Pin Function ..... 252
Table 8.56 PD6 Pin Function ..... 252
Table 8.57 PD5 Pin Function ..... 252
Table 8.58 PD4 Pin Function ..... 253
Table 8.59 PD3 Pin Function ..... 253
Table 8.60 PD2 Pin Function ..... 253
Table 8.61 PD1 Pin Function ..... 253

Table 8.62 PD0 Pin Function ... 253
Table 8.63 Input Pull-Up MOS States (Port D) ... 254
Table 8.64 PE7 Pin Function ... 257
Table 8.65 PE6 Pin Function ... 257
Table 8.66 PE5 Pin Function ... 258
Table 8.67 PE4 Pin Function ... 258
Table 8.68 PE3 Pin Function ... 258
Table 8.69 PE2 Pin Function ... 258
Table 8.70 PE1 Pin Function ... 259
Table 8.71 PE0 Pin Function ... 259
Table 8.72 PE7 Pin Function ... 259
Table 8.73 PE6 Pin Function ... 259
Table 8.74 PE5 Pin Function ... 259
Table 8.75 PE4 Pin Function ... 260
Table 8.76 PE3 Pin Function ... 260
Table 8.77 PE2 Pin Function ... 260
Table 8.78 PE1 Pin Function ... 260
Table 8.79 PE0 Pin Function ... 260
Table 8.80 Input Pull-Up MOS States (Port E) ... 261
Table 8.81 PF7 Pin Function ... 264
Table 8.82 PF6 Pin Function ... 264
Table 8.83 PF5 Pin Function ... 265
Table 8.84 PF4 Pin Function ... 265
Table 8.85 PF3 Pin Function ... 265
Table 8.86 PF2 Pin Function ... 265
Table 8.87 PF1 Pin Function ... 266
Table 8.88 PF0 Pin Function ... 266
Table 8.89 PF7 Pin Function ... 266
Table 8.90 PF3 Pin Function ... 267
Table 8.91 PF0 Pin Function ... 267
Table 8.92 PG4 Pin Function ... 270
Table 8.93 PG3 Pin Function ... 270
Table 8.94 PG2 Pin Function ... 270
Table 8.95 PG1 Pin Function ... 270
Table 8.96 PG1 Pin Function ... 271
Table 8.97 PG0 Pin Function ... 271
Table 8.98 Examples of Ways to Handle Unused Input Pins ... 272

## Section 9 16-Bit Timer Pulse Unit (TPU)

Table 9.1 TPU Functions ... 275
Table 9.2 Pin Configuration ... 277

Table 9.3 CCLR2 to CCLR0 (channel 0) ..... 280
Table 9.4 CCLR2 to CCLR0 (channels 1 and 2) ..... 280
Table 9.5 TPSC2 to TPSC0 (channel 0) ..... 281
Table 9.6 TPSC2 to TPSC0 (channel 1) ..... 281
Table 9.7 TPSC2 to TPSC0 (channel 2) ..... 282
Table 9.8 MD3 to MD0 ..... 283
Table 9.9 TIORH_0 (channel 0) ..... 285
Table 9.10 TIORH_0 (channel 0) ..... 286
Table 9.11 TIORL_0 (channel 0) ..... 287
Table 9.12 TIORL_0 (channel 0) ..... 288
Table 9.13 TIOR_1 (channel 1) ..... 289
Table 9.14 TIOR_1 (channel 1) ..... 290
Table 9.15 TIOR_2 (channel 2) ..... 291
Table 9.16 TIOR_2 (channel 2) ..... 292
Table 9.17 Register Combinations in Buffer Operation ..... 309
Table 9.18 PWM Output Registers and Output Pins ..... 314
Table 9.19 Phase Counting Mode Clock Input Pins ..... 317
Table 9.20 Up/Down-Count Conditions in Phase Counting Mode 1 ..... 318
Table 9.21 Up/Down-Count Conditions in Phase Counting Mode 2 ..... 319
Table 9.22 Up/Down-Count Conditions in Phase Counting Mode 3 ..... 320
Table 9.23 Up/Down-Count Conditions in Phase Counting Mode 4 ..... 321
Table 9.24 TPU Interrupts ..... 322

## Section 10 Watchdog Timer (WDT)

Table 10.1 WDT Interrupt Source ..... 345

## Section 11 Realtime Clock (RTC)

Table 11.1 Pin Configuration ..... 350
Table 11.2 Interrupt Source ..... 360
Table 11.3 Operating State in Each Mode ..... 362

## Section 12 Serial Communication Interface

Table 12.1 Pin Configuration ..... 367
Table 12.2 Relationships between the N Setting in BRR and Bit Rate B ..... 395
Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode) ..... 396
Table 12.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode) ..... 399
Table 12.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode) ..... 400
Table 12.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode) ..... 401
Table 12.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode) ..... 401
Table 12.8 BRR Settings for Various Bit Rates
(Smart Card Interface Mode, when n = 0 and S = 372) ..... 402

Table 12.9 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode) ..... 402
Table 12.10 Serial Transfer Formats (Asynchronous Mode) ..... 404
Table 12.11 SSR Status Flags and Receive Data Handling ..... 411
Table 12.12 SCI Interrupt Sources ..... 440
Table 12.13 Interrupt Sources in Smart Card Interface Mode ..... 441

## Section 13 Boundary Scan Function

Table 13.1 Pin Configuration ..... 451
Table 13.2 Instruction Configuration ..... 452
Table 13.3 IDCODE Register Configuration ..... 454
Table 13.4 Correspondence between LSI Pins and Boundary Scan Register ..... 455

## Section 14 Universal Serial Bus (USB)

Table 14.1 Pin Configuration ..... 467
Table 14.2 Relationship between UTSTR0 Setting and Pin Output ..... 490
Table 14.3 Relationship between Pin Input and UTSTR1 Monitoring Value ..... 491
Table 14.4 Interrupt Sources ..... 495
Table 14.5 Command Decoding by Firmware ..... 516

## Section 15 A/D Converter

Table 15.1 Pin Configuration ..... 537
Table 15.2 Analog Input Channels and Corresponding ADDR Registers ..... 538
Table 15.3 A/D Conversion Time (Single Mode) ..... 545
Table 15.4 A/D Conversion Time (Scan Mode) ..... 545
Table 15.5 A/D Converter Interrupt Source ..... 546
Table 15.6 Analog Pin Specifications ..... 549

## Section 17 Flash Memory (F-ZTAT Version)

Table 17.1 Differences between Boot Mode and User Program Mode ..... 555
Table 17.2 Pin Configuration ..... 561
Table 17.3 Setting On-Board Programming Modes ..... 567
Table 17.4 Boot Mode Operation ..... 570
Table 17.5 System Clock Frequencies for which Automatic Adjustment of LSI Bit Rate Is Possible ..... 570
Table 17.6 Enumeration Information ..... 571
Table 17.7 USB Boot Mode Operation ..... 574
Table 17.8 Flash Memory Operating States ..... 586
Table 17.9 Registers Present in F-ZTAT Version but Absent in Masked ROM Version ..... 592

## Section 19 Clock Pulse Generator

Table 19.1 Damping Resistance Value ..... 600

Table 19.2 Crystal Resonator Characteristics ..... 600
Table 19.3 External Clock Input Conditions ..... 601
Table 19.4 External Clock Input Conditions when Duty Adjustment Circuit Is not Used ..... 602

## Section 20 Power-Down Modes

Table 20.1 LSI Internal States in Each Mode ..... 608
Table 20.2 Transition Conditions of Power-Down Modes ..... 610
Table 20.3 Oscillation Stabilization Time Settings ..... 620
Table 20.4 ϕ Pin State in Each Processing State ..... 627

## Section 22 Electrical Characteristics

Table 22.1 Absolute Maximum Ratings ..... 657
Table 22.2 DC Characteristics ..... 659
Table 22.3 Permissible Output Currents ..... 662
Table 22.4 Clock Timing ..... 664
Table 22.5 Control Signal Timing ..... 666
Table 22.6 Bus Timing ..... 668
Table 22.7 Timing of On-Chip Supporting Modules ..... 675
Table 22.8 USB Characteristics (USD+ and USD- pins) when On-Chip USB Transceiver Is Used ..... 680
Table 22.9 A/D Conversion Characteristics ..... 682
Table 22.10 Flash Memory Characteristics ..... 683

# Section 1 Overview

## 1.1 Overview

- High-speed H8S/2000 central processing unit with 16-bit architecture
  - Upward-compatible with H8/300 and H8/300H CPUs on an object level
  - Sixteen 16-bit general registers
  - 65 basic instructions
- Various peripheral functions
  - DMA controller (DMAC)
  - 16-bit timer-pulse unit (TPU)
  - Watchdog timer (WDT)
  - Realtime clock (RTC)
  - Serial communication interface (SCI)
  - Boundary scan
  - Universal serial bus (USB)
  - 10-bit A/D converter
  - High-performance user debugging interface (H-UDI)
  - Clock pulse generator
- On-chip memory

  H8S/2218 Group

| ROM | Part No. | ROM | RAM | Remarks |
|---|---|---|---|---|
| Flash memory Version | HD64F2218 | 128 kbytes | 12 kbytes | SCI boot mode |
| | HD64F2218U | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2218CU | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2217CU | 64 kbytes | 12 kbytes | USB boot mode |
| Masked ROM Version | HD6432217 | 64 kbytes | 8 kbytes | — |

H8S/2212 Group

| ROM | Part No. | ROM | RAM | Remarks |
|---|---|---|---|---|
| Flash memory Version | HD64F2212 | 128 kbytes | 12 kbytes | SCI boot mode |
| | HD64F2212U | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2212CU | 128 kbytes | 12 kbytes | USB boot mode |
| | HD64F2211 | 64 kbytes | 8 kbytes | SCI boot mode |
| | HD64F2211U | 64 kbytes | 8 kbytes | USB boot mode |
| | HD64F2211CU | 64 kbytes | 8 kbytes | USB boot mode |
| | HD64F2210CU | 32 kbytes | 8 kbytes | USB boot mode |
| Masked ROM Version | HD6432211 | 64 kbytes | 8 kbytes | — |
| | HD6432210 | 32 kbytes | 4 kbytes | — |
| | HD6432210S | 32 kbytes | 4 kbytes | — |

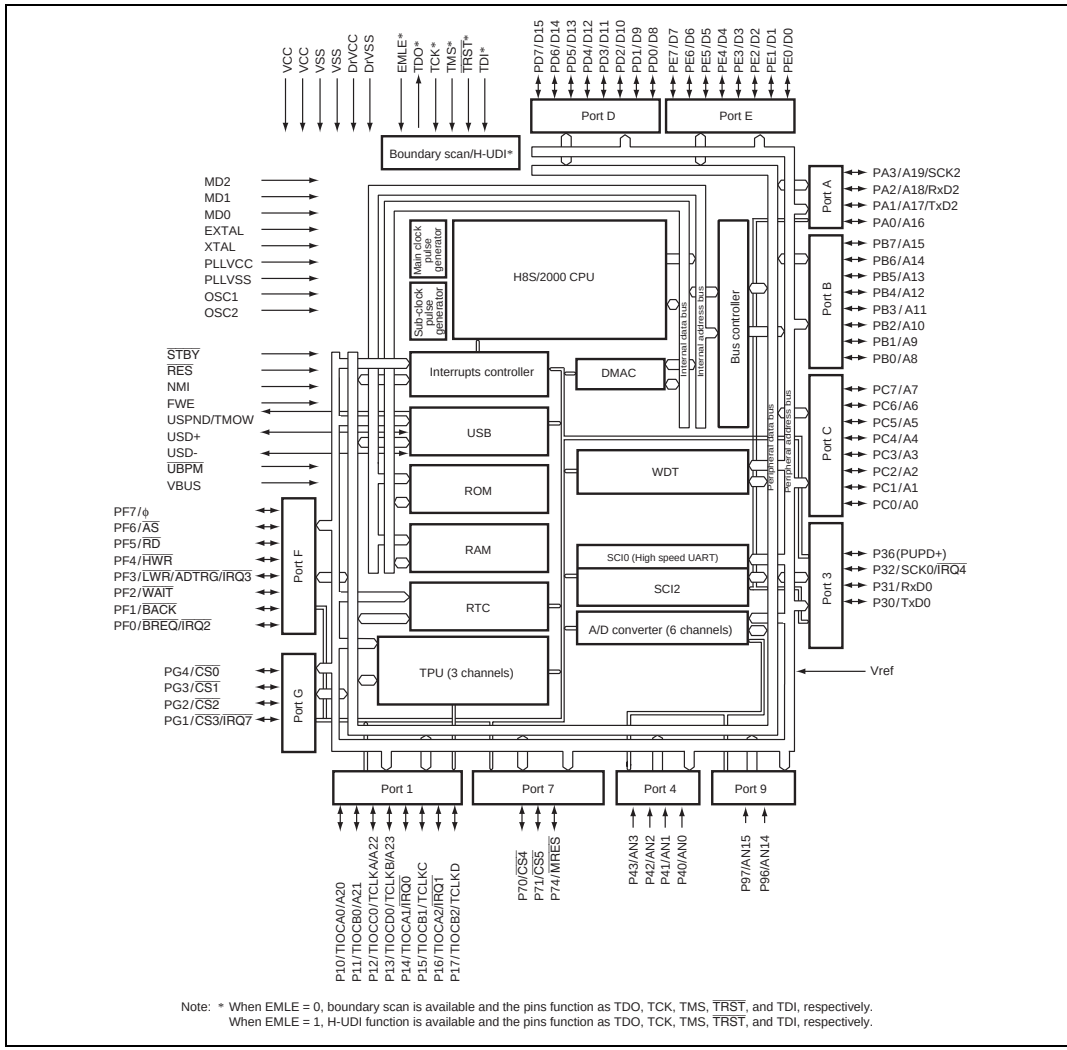
- General I/O ports
  I/O pins: 69 for the H8S/2218 Group, 37 for the H8S/2212 Group
- Supports various power-down states
- Compact package

| Package | Code* | Body Size | Pin Pitch | Remarks |
|---|---|---|---|---|
| TQFP-100 | TFP-100G, TFP-100GV | 12.0 × 12.0 mm | 0.4 mm | H8S/2218 Group |
| P-LFBGA-112 | BP-112, BP-112V | 10.0 × 10.0 mm | 0.8 mm | |
| LQFP-64 | FP-64E, FP-64EV | 10.0 × 10.0 mm | 0.5 mm | H8S/2212 Group |
| VQFN-64 | TNP-64B, TNP-64BV | 8.0 × 8.0 mm | 0.4 mm | |

Note: * A V appended to the end of the package code indicates a lead-free version.

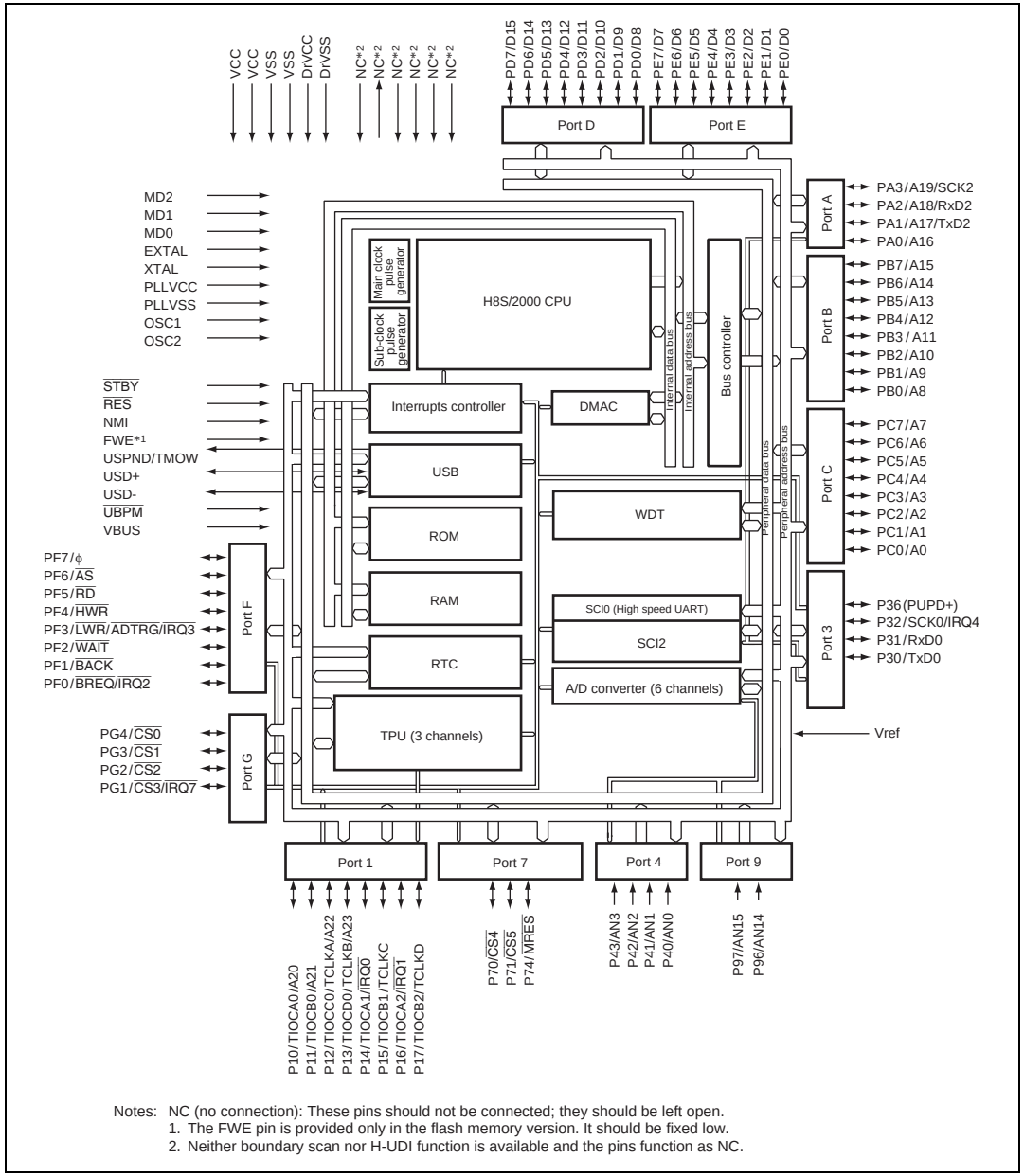
## 1.2 Internal Block Diagram

The internal block diagram of the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU is shown in figure 1.1. The internal block diagram of the HD6432217 is shown in figure 1.2. The internal block diagram of the HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU is shown in figure 1.3. The internal block diagram of the HD6432211, HD6432210 and HD6432210S is shown in figure 1.4.



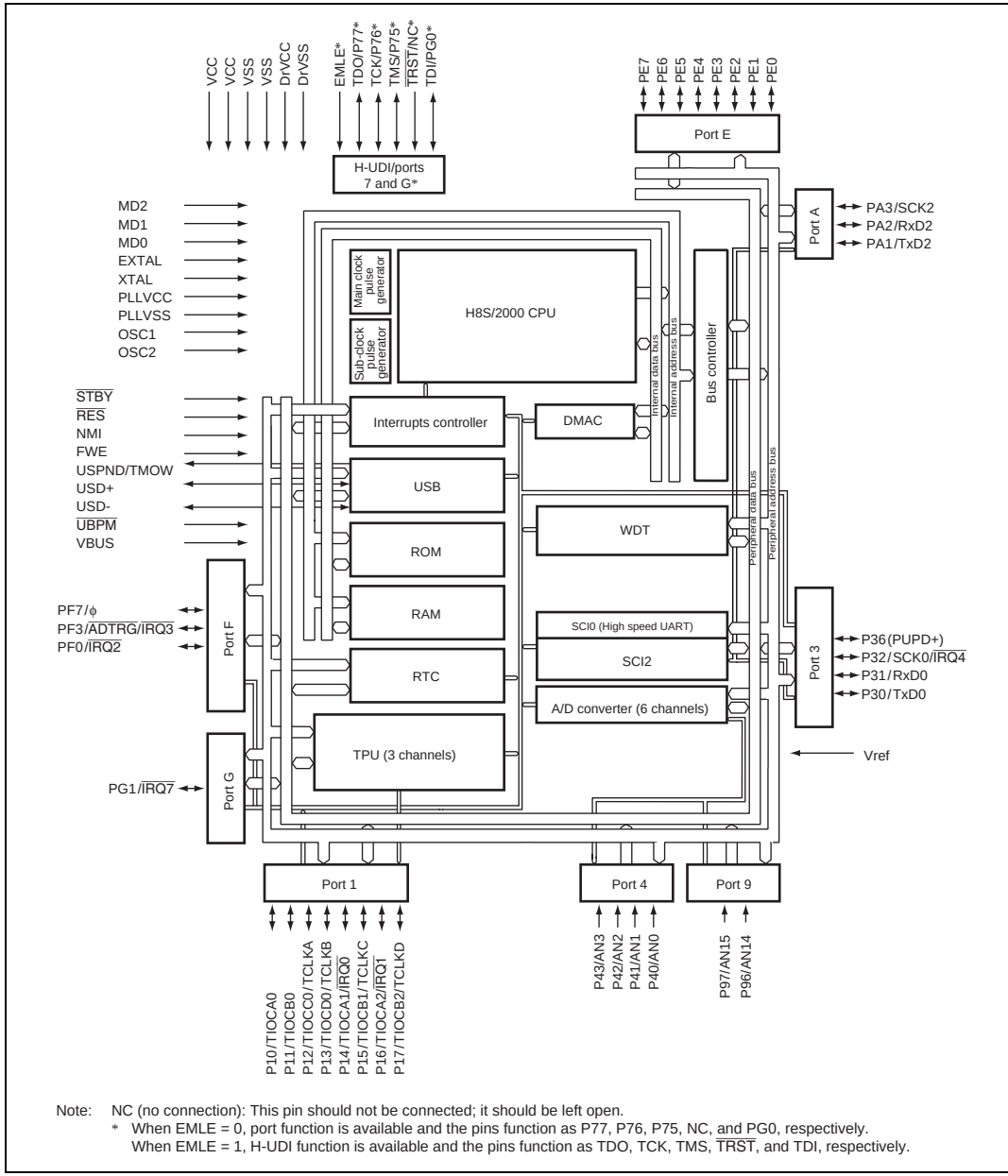

Note: * When EMLE = 0, boundary scan is available and the pins function as TDO, TCK, TMS, TRST, and TDI, respectively.
When EMLE = 1, H-UDI function is available and the pins function as TDO, TCK, TMS, TRST, and TDI, respectively.

**Figure 1.1 Internal Block Diagram of HD64F2218, HD64F2218U, HD64F2218CU and HD642217CU**



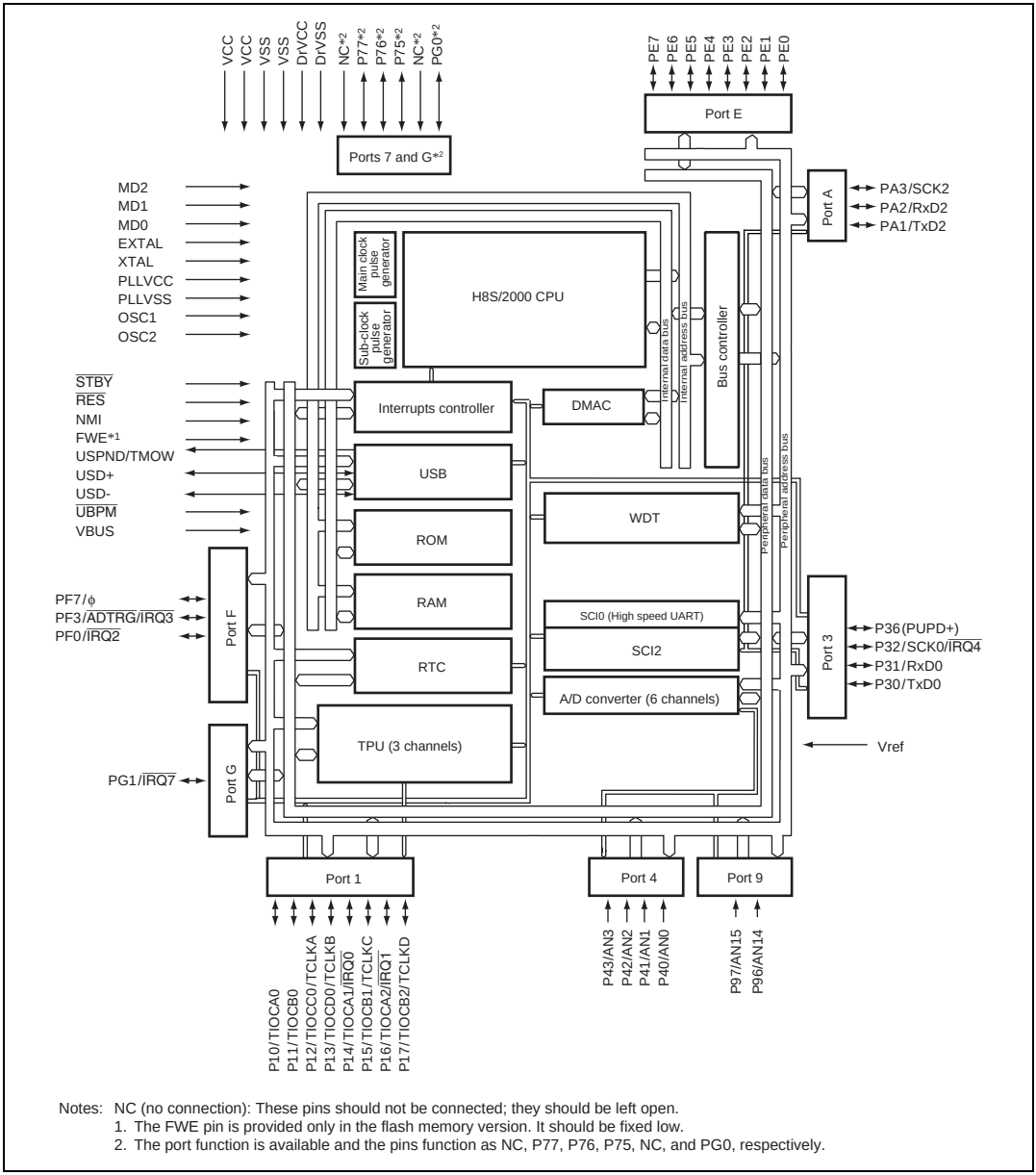

Notes: NC (no connection): These pins should not be connected; they should be left open.
1. The FWE pin is provided only in the flash memory version. It should be fixed low.
2. Neither boundary scan nor H-UDI function is available and the pins function as NC.

**Figure 1.2 Internal Block Diagram of HD6432217**




Note: NC (no connection): This pin should not be connected; it should be left open.
* When EMLE = 0, port function is available and the pins function as P77, P76, P75, NC, and PG0, respectively.
When EMLE = 1, H-UDI function is available and the pins function as TDO, TCK, TMS, TRST, and TDI, respectively.

**Figure 1.3 Internal Block Diagram of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU**

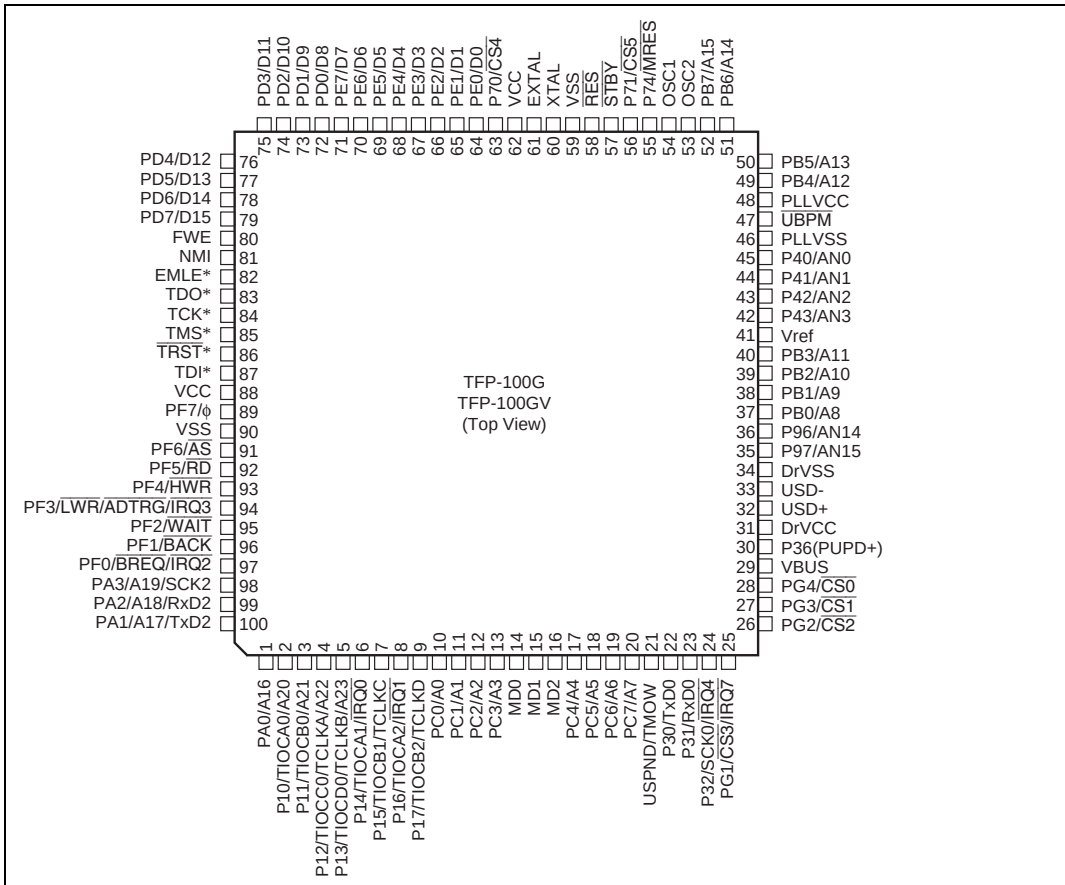



Notes: NC (no connection): These pins should not be connected; they should be left open.
1. The FWE pin is provided only in the flash memory version. It should be fixed low.
2. The port function is available and the pins function as NC, P77, P76, P75, NC, and PG0, respectively.

**Figure 1.4 Internal Block Diagram of HD6432211, HD6432210 and HD6432210S**

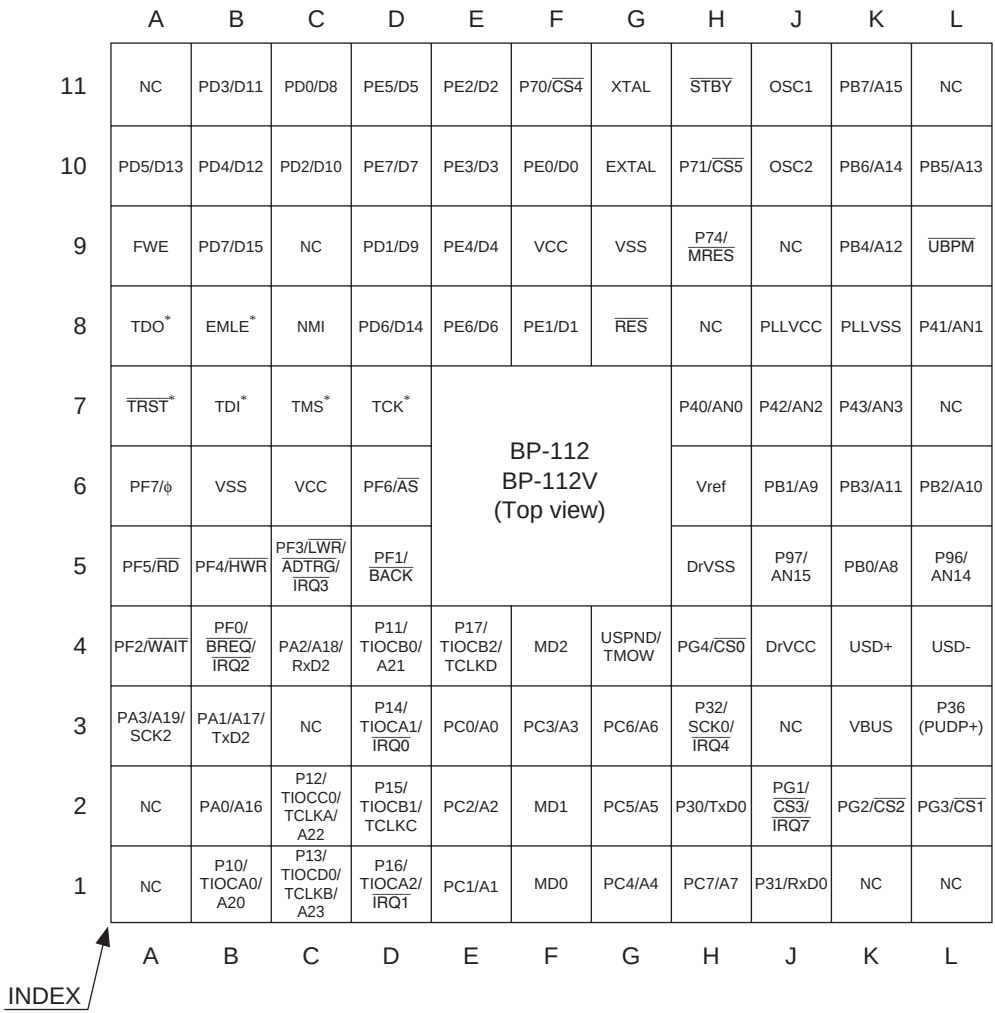
## 1.3 Pin Arrangements

The pin arrangements of the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU are shown in figures 1.5 and 1.6. The pin arrangements of the HD6432217 are shown in figures 1.7 and 1.8. The pin arrangements of the HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU are shown in figures 1.9 and 1.11. The pin arrangements of the HD6432211, HD6432210 and HD6432210S is shown in figures 1.10 and 1.12.




Note: * When EMLE = 0, boundary scan is available and the pins function as TDO, TCK, TMS, $\overline{\text{TRST}}$, and TDI, respectively.
When EMLE = 1, H-UDI function is available and the pins function as TDO, TCK, TMS, $\overline{\text{TRST}}$, and TDI, respectively.

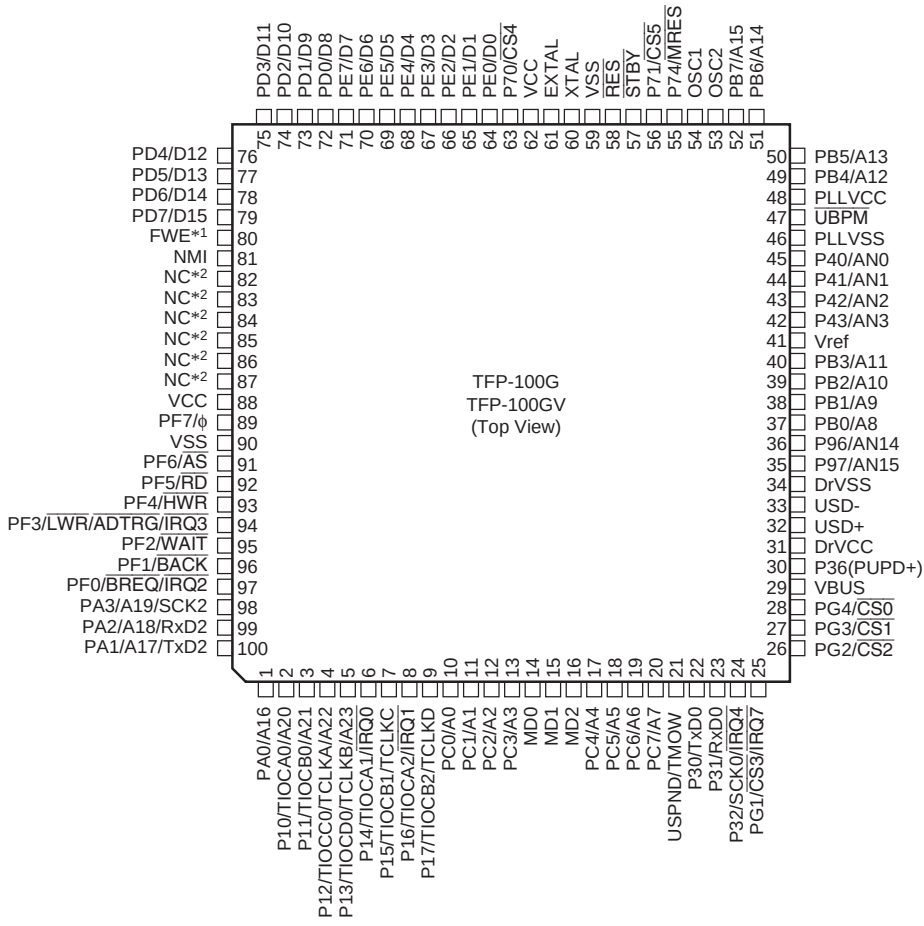
**Figure 1.5 Pin Arrangements of HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU (TFP-100G, TFP-100GV)**




Notes: NC (no connection): These pins should not be connected; they should be left open.

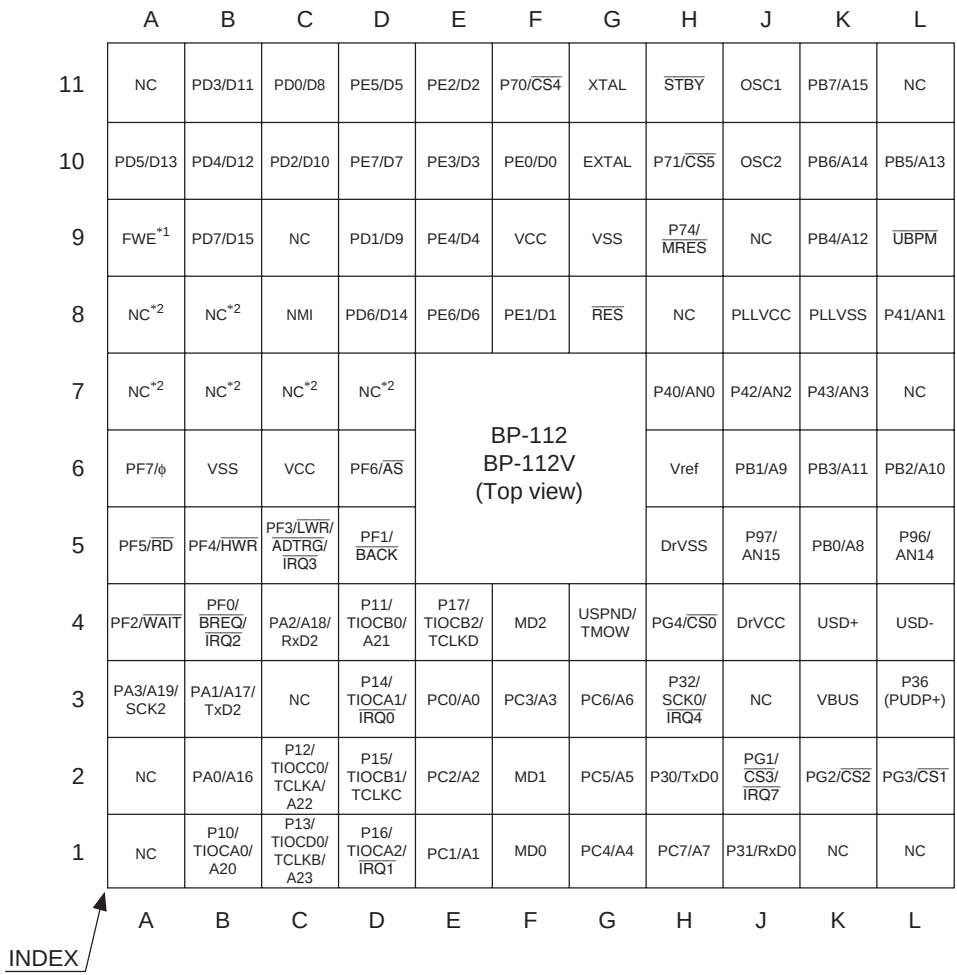
* When EMLE = 0, boundary scan is available and the pins function as TDO, TCK, TMS, TRST, and TDI, respectively.
When EMLE = 1, H-UDI function is available and the pins function as TDO, TCK, TMS, TRST, and TDI, respectively.

**Figure 1.6 Pin Arrangements of HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU (BP-112, BP-112V)**




Notes: NC (no connection): These pins should not be connected; they should be left open.
1. The FWE pin is provided only in the flash memory version. It should be fixed low.
2. Neither boundary scan nor H-UDI function is available and the pins function as NC.

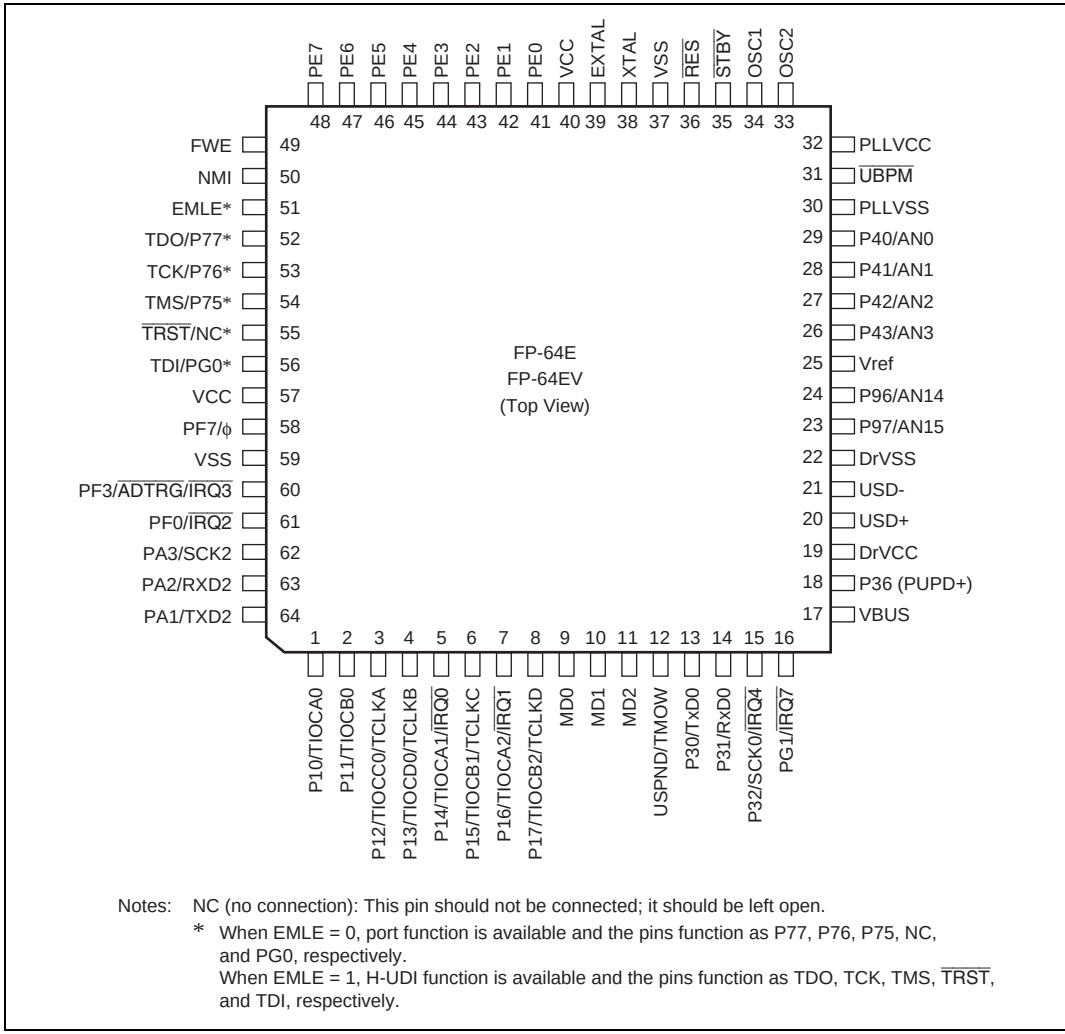
**Figure 1.7 Pin Arrangements of HD6432217 (TFP-100G, TFP-100GV)**




Notes: NC (no connection): These pins should not be connected; they should be left open.

1. The FWE pin is provided only in the flash memory version. It should be fixed low.
2. Neither boundary scan nor H-UDI function is available and the pins function as NC.

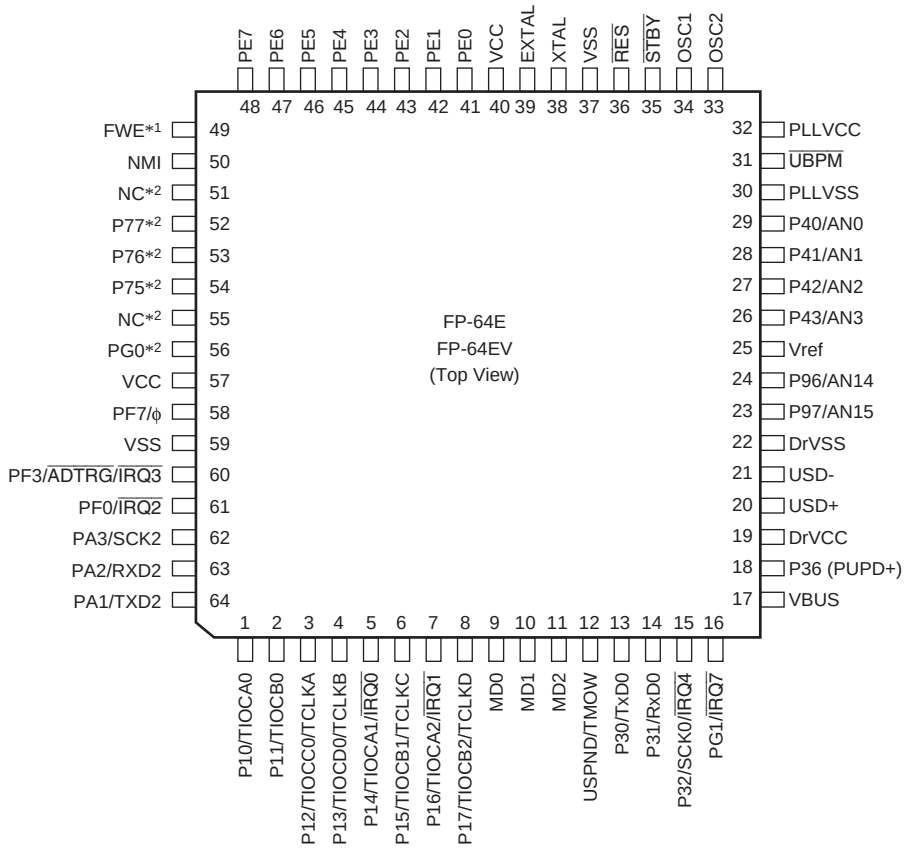
**Figure 1.8 Pin Arrangements of HD6432217 (BP-112, BP-112V)**




Notes: NC (no connection): This pin should not be connected; it should be left open.

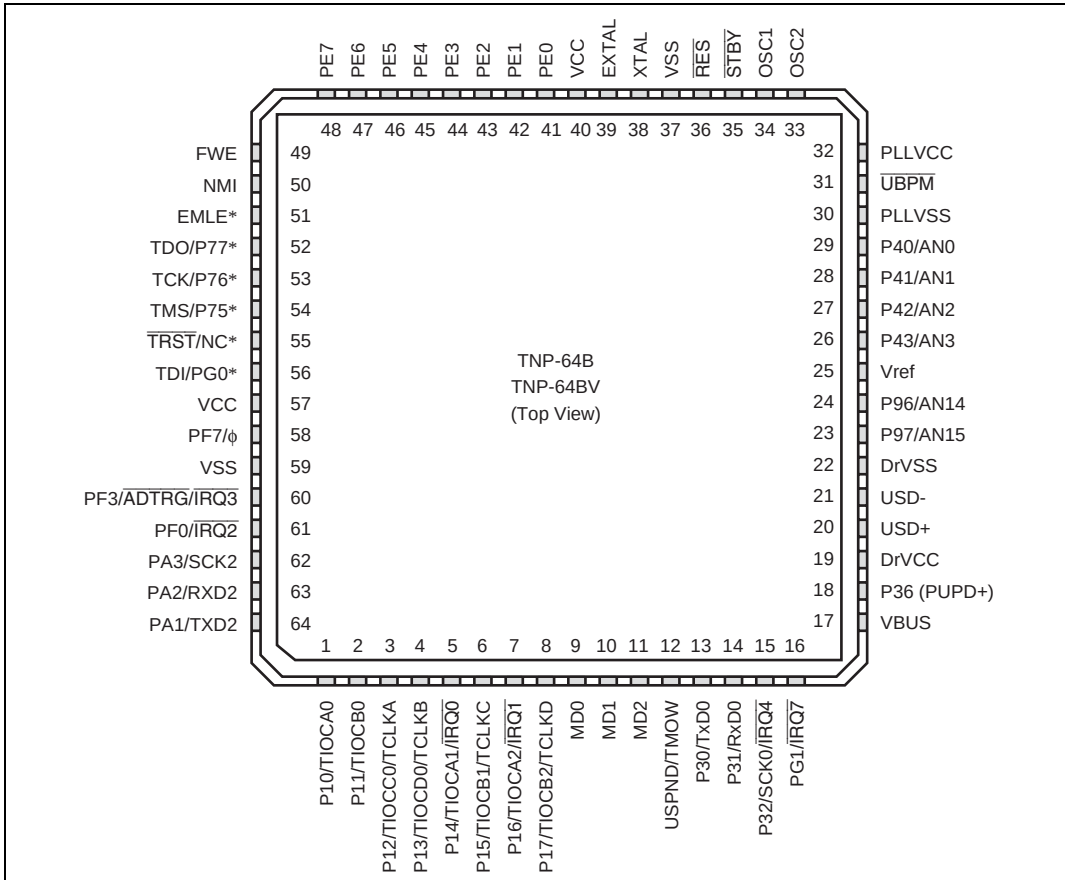
* When EMLE = 0, port function is available and the pins function as P77, P76, P75, NC, and PG0, respectively.
When EMLE = 1, H-UDI function is available and the pins function as TDO, TCK, TMS, $\overline{\text{TRST}}$, and TDI, respectively.

**Figure 1.9 Pin Arrangements of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU (FP-64E, FP-64EV)**




Notes: NC (no connection): These pins should not be connected; they should be left open.
1. The FWE pin is provided only in the flash memory version. It should be fixed low.
2. The port function is available and the pins function as NC, P77, P76, P75, NC, and PG0, respectively.

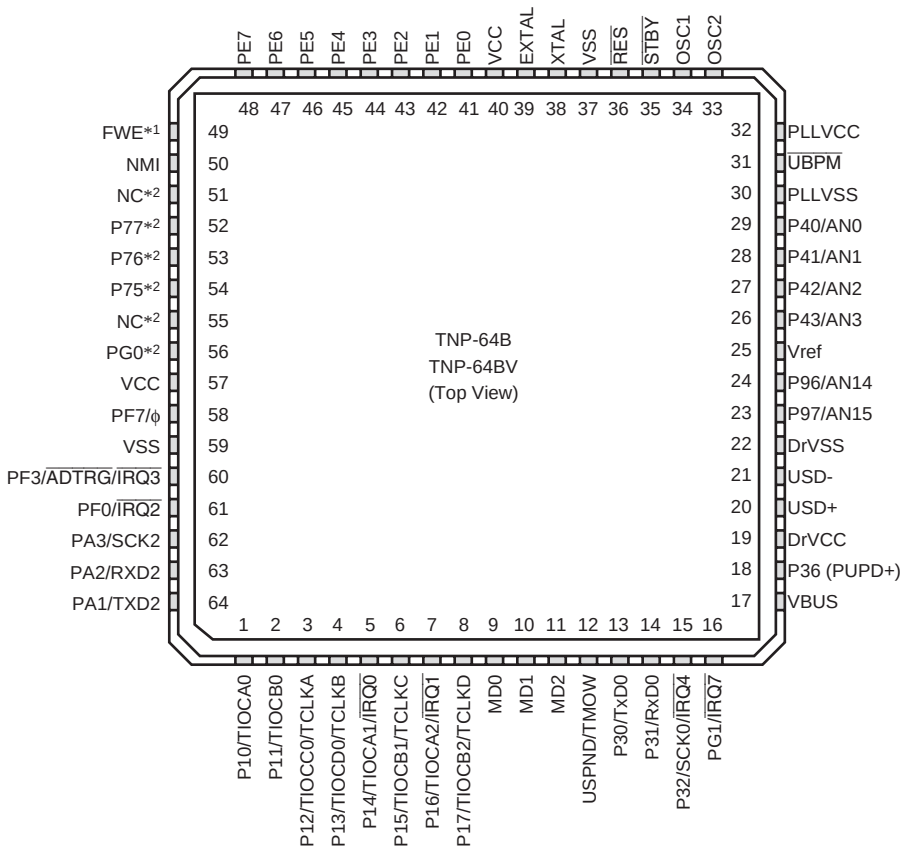
**Figure 1.10 Pin Arrangements of HD6432211, HD6432210 and HD6432210S (FP-64E, FP-64EV)**




Notes: NC (no connection): This pin should not be connected; it should be left open.

* When EMLE = 0, the port function is enabled (P77, P76, P75, NC, and PG0).
When EMLE = 1, the H-UDI function is enabled (TDO, TCK, TMS, TRST, and TDI).

**Figure 1.11 Pin Arrangements of HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU and HD64F2210CU (TNP-64B, TNP-64BV)**




Notes: NC (no connection): These pins should not be connected; they should be left open.
1. The FWE pin is provided only in the flash memory version. It should be fixed low.
2. The port function is enabled (P77, P76, P75, NC, and PG0).

**Figure 1.12 Pin Arrangements of HD6432211, HD6432210 and HD6432210S (TNP-64B, TNP-64BV)**

## 1.4 Pin Functions in Each Operating Mode

Table 1.1 shows the pin functions in each operating mode for the H8S/2218 Group, and table 1.2 shows that for the H8S/2212 Group.

**Table 1.1 Pin Functions in Each Operating Mode for H8S/2218 Group**

| Pin No. | | Pin Name* | | | |
|---|---|---|---|---|---|
| TFP-100G, TFP-100GV | BP-112, BP-112V | Modes 4, 5 | Mode 6 | Mode 7 | Programmer Mode |
| 1 | B2 | PA0/A16 | PA0/A16 | PA0 | NC |
| 2 | B1 | P10/TIOCA0/A20 | P10/TIOCA0/A20 | P10/TIOCA0 | A2 |
| 3 | D4 | P11/TIOCB0/A21 | P11/TIOCB0/A21 | P11/TIOCB0 | A3 |
| 4 | C2 | P12/TIOCC0/TCLKA/A22 | P12/TIOCC0/TCLKA/A22 | P12/TIOCC0/TCLKA | A4 |
| 5 | C1 | P13/TIOCD0/TCLKB/A23 | P13/TIOCD0/TCLKB/A23 | P13/TIOCD0/TCLKB | A5 |
| 6 | D3 | P14/TIOCA1/$\overline{\text{IRQ0}}$ | P14/TIOCA1/$\overline{\text{IRQ0}}$ | P14/TIOCA1/$\overline{\text{IRQ0}}$ | VSS |
| 7 | D2 | P15/TIOCB1/TCLKC | P15/TIOCB1/TCLKC | P15/TIOCB1/TCLKC | $\overline{\text{WE}}$ |
| 8 | D1 | P16/TIOCA2/$\overline{\text{IRQ1}}$ | P16/TIOCA2/$\overline{\text{IRQ1}}$ | P16/TIOCA2/$\overline{\text{IRQ1}}$ | VSS |
| 9 | E4 | P17/TIOCB2/TCLKD | P17/TIOCB2/TCLKD | P17/TIOCB2/TCLKD | $\overline{\text{CE}}$ |
| 10 | E3 | A0 | PC0/A0 | PC0 | NC |
| 11 | E1 | A1 | PC1/A1 | PC1 | NC |
| 12 | E2 | A2 | PC2/A2 | PC2 | NC |
| 13 | F3 | A3 | PC3/A3 | PC3 | NC |
| 14 | F1 | MD0 | MD0 | MD0 | VSS |
| 15 | F2 | MD1 | MD1 | MD1 | VSS |
| 16 | F4 | MD2 | MD2 | MD2 | VSS |
| 17 | G1 | A4 | PC4/A4 | PC4 | NC |
| 18 | G2 | A5 | PC5/A5 | PC5 | NC |
| 19 | G3 | A6 | PC6/A6 | PC6 | NC |
| 20 | H1 | A7 | PC7/A7 | PC7 | NC |
| 21 | G4 | USPND/TMOW | USPND/TMOW | USPND/TMOW | NC |
| 22 | H2 | P30/TxD0 | P30/TxD0 | P30/TxD0 | A10 |
| 23 | J1 | P31/RxD0 | P31/RxD0 | P31/RxD0 | A11 |
| 24 | H3 | P32/SCK0/$\overline{\text{IRQ4}}$ | P32/SCK0/$\overline{\text{IRQ4}}$ | P32/SCK0/$\overline{\text{IRQ4}}$ | A12 |
| 25 | J2 | PG1/$\overline{\text{CS3}}$/$\overline{\text{IRQ7}}$ | PG1/$\overline{\text{CS3}}$/$\overline{\text{IRQ7}}$ | PG1/$\overline{\text{IRQ7}}$ | A15 |

| Pin No. | | Pin Name* | | | |
|---|---|---|---|---|---|
| **TFP-100G, TFP-100GV** | **BP-112, BP-112V** | **Modes 4, 5** | **Mode 6** | **Mode 7** | **Programmer Mode** |
| 26 | K2 | PG2/$\overline{CS2}$ | PG2/$\overline{CS2}$ | PG2 | NC |
| 27 | L2 | PG3/$\overline{CS1}$ | PG3/$\overline{CS1}$ | PG3 | NC |
| 28 | H4 | PG4/$\overline{CS0}$ | PG4/$\overline{CS0}$ | PG4 | NC |
| 29 | K3 | VBUS | VBUS | VBUS | VSS |
| 30 | L3 | P36 (PUDP+) | P36 (PUDP+) | P36 (PUDP+) | A16 |
| 31 | J4 | DrVCC | DrVCC | DrVCC | VCC |
| 32 | K4 | USD+ | USD+ | USD+ | NC |
| 33 | L4 | USD- | USD- | USD- | NC |
| 34 | H5 | DrVSS | DrVSS | DrVSS | VSS |
| 35 | J5 | P97/AN15 | P97/AN15 | P97/AN15 | A7 |
| 36 | L5 | P96/AN14 | P96/AN14 | P96/AN14 | A6 |
| 37 | K5 | PB0/A8 | PB0/A8 | PB0 | NC |
| 38 | J6 | PB1/A9 | PB1/A9 | PB1 | NC |
| 39 | L6 | PB2/A10 | PB2/A10 | PB2 | NC |
| 40 | K6 | PB3/A11 | PB3/A11 | PB3 | NC |
| 41 | H6 | Vref | Vref | Vref | VCC |
| 42 | K7 | P43/AN3 | P43/AN3 | P43/AN3 | A14 |
| 43 | J7 | P42/AN2 | P42/AN2 | P42/AN2 | A13 |
| 44 | L8 | P41/AN1 | P41/AN1 | P41/AN1 | A9 |
| 45 | H7 | P40/AN0 | P40/AN0 | P40/AN0 | A8 |
| 46 | K8 | PLLVSS | PLLVSS | PLLVSS | VSS |
| 47 | L9 | $\overline{UBPM}$ | $\overline{UBPM}$ | $\overline{UBPM}$ | A17 |
| 48 | J8 | PLLVCC | PLLVCC | PLLVCC | VCC |
| 49 | K9 | PB4/A12 | PB4/A12 | PB4 | NC |
| 50 | L10 | PB5/A13 | PB5/A13 | PB5 | NC |
| 51 | K10 | PB6/A14 | PB6/A14 | PB6 | NC |
| 52 | K11 | PB7/A15 | PB7/A15 | PB7 | NC |
| 53 | J10 | OSC2 | OSC2 | OSC2 | NC |
| 54 | J11 | OSC1 | OSC1 | OSC1 | VCC |
| 55 | H9 | P74/$\overline{MRES}$ | P74/$\overline{MRES}$ | P74/$\overline{MRES}$ | NC |

| Pin No. | | Pin Name* | | | |
|---|---|---|---|---|---|
| **TFP-100G, TFP-100GV** | **BP-112, BP-112V** | **Modes 4, 5** | **Mode 6** | **Mode 7** | **Programmer Mode** |
| 56 | H10 | P71/$\overline{CS5}$ | P71/$\overline{CS5}$ | P71 | NC |
| 57 | H11 | $\overline{STBY}$ | $\overline{STBY}$ | $\overline{STBY}$ | VCC |
| 58 | G8 | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ | $\overline{RES}$ |
| 59 | G9 | VSS | VSS | VSS | VSS |
| 60 | G11 | XTAL | XTAL | XTAL | XTAL |
| 61 | G10 | EXTAL | EXTAL | EXTAL | EXTAL |
| 62 | F9 | VCC | VCC | VCC | VCC |
| 63 | F11 | P70/$\overline{CS4}$ | P70/$\overline{CS4}$ | P70 | NC |
| 64 | F10 | PE0/D0 | PE0/D0 | PE0 | D0 |
| 65 | F8 | PE1/D1 | PE1/D1 | PE1 | D1 |
| 66 | E11 | PE2/D2 | PE2/D2 | PE2 | D2 |
| 67 | E10 | PE3/D3 | PE3/D3 | PE3 | D3 |
| 68 | E9 | PE4/D4 | PE4/D4 | PE4 | D4 |
| 69 | D11 | PE5/D5 | PE5/D5 | PE5 | D5 |
| 70 | E8 | PE6/D6 | PE6/D6 | PE6 | D6 |
| 71 | D10 | PE7/D7 | PE7/D7 | PE7 | D7 |
| 72 | C11 | D8 | D8 | PD0 | NC |
| 73 | D9 | D9 | D9 | PD1 | NC |
| 74 | C10 | D10 | D10 | PD2 | NC |
| 75 | B11 | D11 | D11 | PD3 | NC |
| 76 | B10 | D12 | D12 | PD4 | NC |
| 77 | A10 | D13 | D13 | PD5 | NC |
| 78 | D8 | D14 | D14 | PD6 | NC |
| 79 | B9 | D15 | D15 | PD7 | NC |
| 80 | A9 | FWE | FWE | FWE | FWE |
| 81 | C8 | NMI | NMI | NMI | VCC |
| 82 | B8 | EMLE/NC | EMLE/NC | EMLE/NC | VSS |
| 83 | A8 | TDO/NC | TDO/NC | TDO/NC | NC |
| 84 | D7 | TCK/NC | TCK/NC | TCK/NC | VCC |
| 85 | C7 | TMS/NC | TMS/NC | TMS/NC | VCC |

| Pin No. | | Pin Name* | | | |
|---|---|---|---|---|---|
| TFP-100G, TFP-100GV | BP-112, BP-112V | Modes 4, 5 | Mode 6 | Mode 7 | Programmer Mode |
| 86 | A7 | $\overline{\text{TRST}}$/NC | $\overline{\text{TRST}}$/NC | $\overline{\text{TRST}}$/NC | $\overline{\text{RES}}$ |
| 87 | B7 | TDI/NC | TDI/NC | TDI/NC | VSS |
| 88 | C6 | VCC | VCC | VCC | VCC |
| 89 | A6 | PF7/$\phi$ | PF7/$\phi$ | PF7/$\phi$ | NC |
| 90 | B6 | VSS | VSS | VSS | VSS |
| 91 | D6 | $\overline{\text{AS}}$ | $\overline{\text{AS}}$ | PF6 | NC |
| 92 | A5 | $\overline{\text{RD}}$ | $\overline{\text{RD}}$ | PF5 | NC |
| 93 | B5 | $\overline{\text{HWR}}$ | $\overline{\text{HWR}}$ | PF4 | NC |
| 94 | C5 | PF3/$\overline{\text{LWR}}$/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$ | PF3/$\overline{\text{LWR}}$/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$ | PF3/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$ | VCC |
| 95 | A4 | PF2/$\overline{\text{WAIT}}$ | PF2/$\overline{\text{WAIT}}$ | PF2 | NC |
| 96 | D5 | PF1/$\overline{\text{BACK}}$ | PF1/$\overline{\text{BACK}}$ | PF1 | NC |
| 97 | B4 | PF0/$\overline{\text{BREQ}}$/$\overline{\text{IRQ2}}$ | PF0/$\overline{\text{BREQ}}$/$\overline{\text{IRQ2}}$ | PF0/$\overline{\text{IRQ2}}$ | VCC |
| 98 | A3 | PA3/A19/SCK2 | PA3/A19/SCK2 | PA3/SCK2 | A1 |
| 99 | C4 | PA2/A18/RxD2 | PA2/A18/RxD2 | PA2/RxD2 | A0 |
| 100 | B3 | PA1/A17/TxD2 | PA1/A17/TxD2 | PA1/TxD2 | $\overline{\text{OE}}$ |

Note: * The NC should be left open.

**Table 1.2 Pin Functions in Each Operating Mode for H8S/2212 Group**

| Pin No. | Pin Name* | |
|---|---|---|
| FP-64E, FP-64EV, TNP-64B, TNP-64BV | Mode 7 | Programmer Mode |
| 1 | P10/TIOCA0 | A2 |
| 2 | P11/TIOCB0 | A3 |
| 3 | P12/TIOCC0/TCLKA | A4 |
| 4 | P13/TIOCD0/TCLKB | A5 |
| 5 | P14/TIOCA1/$\overline{\text{IRQ0}}$ | VSS |
| 6 | P15/TIOCB1/TCLKC | $\overline{\text{WE}}$ |
| 7 | P16/TIOCA2/$\overline{\text{IRQ1}}$ | VSS |
| 8 | P17/TIOCB2/TCLKD | $\overline{\text{CE}}$ |
| 9 | MD0 | VSS |

| Pin No. | | Pin Name* |
|---|---|---|
| FP-64E, FP-64EV, TNP-64B, TNP-64BV | Mode 7 | Programmer Mode |
| 10 | MD1 | VSS |
| 11 | MD2 | VSS |
| 12 | USPND/TMOW | NC |
| 13 | P30/TxD0 | A10 |
| 14 | P31/RxD0 | A11 |
| 15 | P32/SCK0/$\overline{\text{IRQ4}}$ | A12 |
| 16 | PG1/$\overline{\text{IRQ7}}$ | A15 |
| 17 | VBUS | VSS |
| 18 | P36(PD+) | A16 |
| 19 | DrVCC | VCC |
| 20 | USD+ | NC |
| 21 | USD- | NC |
| 22 | DrVSS | VSS |
| 23 | P97/AN15 | A7 |
| 24 | P96/AN14 | A6 |
| 25 | Vref | VCC |
| 26 | P43/AN3 | A14 |
| 27 | P42/AN2 | A13 |
| 28 | P41/AN1 | A9 |
| 29 | P40/AN0 | A8 |
| 30 | PLLVSS | VSS |
| 31 | $\overline{\text{UBPM}}$ | A17 |
| 32 | PLLVCC | VCC |
| 33 | OSC2 | NC |
| 34 | OSC1 | VCC |
| 35 | $\overline{\text{STBY}}$ | VCC |
| 36 | $\overline{\text{RES}}$ | $\overline{\text{RES}}$ |
| 37 | VSS | VSS |
| 38 | XTAL | XTAL |

| Pin No. | | Pin Name* |
|---|---|---|
| FP-64E, FP-64EV, TNP-64B, TNP-64BV | Mode 7 | Programmer Mode |
| 39 | EXTAL | EXTAL |
| 40 | VCC | VCC |
| 41 | PE0 | D0 |
| 42 | PE1 | D1 |
| 43 | PE2 | D2 |
| 44 | PE3 | D3 |
| 45 | PE4 | D4 |
| 46 | PE5 | D5 |
| 47 | PE6 | D6 |
| 48 | PE7 | D7 |
| 49 | FWE | FWE |
| 50 | NMI | VCC |
| 51 | EMLE/NC | VSS |
| 52 | TDO/P77 | NC |
| 53 | TCK/P76 | VCC |
| 54 | TMS/P75 | VCC |
| 55 | $\overline{\text{TRST}}$/NC | $\overline{\text{RES}}$ |
| 56 | TDI/PG0 | VSS |
| 57 | VCC | VCC |
| 58 | PF7/$\phi$ | NC |
| 59 | VSS | VSS |
| 60 | PF3/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$ | VCC |
| 61 | PF0/$\overline{\text{IRQ2}}$ | VCC |
| 62 | PA3/SCK2 | A1 |
| 63 | PA2/RxD2 | A0 |
| 64 | PA1/TxD2 | $\overline{\text{OE}}$ |

Note: * The NC should be left open.

## 1.5 Pin Functions

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| Power supply | VCC | 62<br>88 | F9<br>C6 | 40<br>57 | Input | Power supply pins. Connect all these pins to the system power supply. |
| | VSS | 59<br>90 | G9<br>B6 | 37<br>59 | Input | Ground pins. Connect all these pins to the system power supply (0 V). |
| | PLLVCC | 48 | J8 | 32 | Input | Power supply pin for an on-chip PLL oscillator. Connect this pin to the system power supply. |
| | PLLVSS | 46 | K8 | 30 | Input | Ground pin for an on-chip PLL oscillator |
| Clock | XTAL | 60 | G11 | 38 | Input | For connection to a crystal resonator. For examples of crystal resonator connection and external clock input, see section 19, Clock Pulse Generator. |
| | EXTAL | 61 | G10 | 39 | Input | For connection to a crystal resonator. An external clock can be supplied from the EXTAL pin. For examples of crystal resonator connection and external clock input, see section 19, Clock Pulse Generator. |
| | OSC1<br>OSC2 | 54<br>53 | J11<br>J10 | 34<br>33 | Input | For connection to a 32.768-kHz crystal resonator. For examples of crystal resonator connection, see section 19, Clock Pulse Generator. |
| | $\phi$ | 89 | A6 | 58 | Output | Supplies the system clock to external devices. |
| Operating mode control | MD2<br>MD1<br>MD0 | 16<br>15<br>14 | F4<br>F2<br>F1 | 11<br>10<br>9 | Input | Set the operating mode. Inputs at these pins cannot be modified during operation.<br>Sets the operating mode. Inputs at these pins should not be changed during operation. Except for mode changing, be sure to fix the levels of the mode pins (MD2 to MD0) by pulling them down or pulling them up until the power turns off. |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| System control | $\overline{\text{RES}}$* | 58 | G8 | 36 | Input | Reset pin. When this pin is driven low, the chip is reset. |
| | $\overline{\text{STBY}}$* | 57 | H11 | 35 | Input | When this pin is driven low, a transition is made to hardware standby mode. |
| | $\overline{\text{MRES}}$ | 55 | H9 | — | Input | When this pin is driven low, a transition is made to manual reset mode. (Supported only by the H8S/2218 Group) |
| | $\overline{\text{BREQ}}$ | 97 | B4 | — | Input | Used by an external bus master to issue a bus request to this LSI (Supported only by the H8S/2218 Group) |
| | $\overline{\text{BACK}}$ | 96 | D5 | — | Output | Indicates that the bus has been released to an external bus master. (Supported only by the H8S/2218 Group) |
| | FWE | 80 | A9 | 49 | Input | Pin for use by flash memory. This pin is only used in the flash memory version. In the masked ROM version, it should be connected to the system power supply (0 V). |
| | EMLE | 82 | B8 | 51 | Input | Emulator enable<br>When E10A is not used, connect this pin to the system power supply (0 V). When E10A is used, this pin should be fixed high. |
| Interrupts | NMI* | 81 | C8 | 50 | Input | Nonmaskable interrupt pin. If this pin is not used, it should be fixed high. |
| | $\overline{\text{IRQ7}}$ | 25 | J2 | 16 | Input | These pins request a maskable interrupt. |
| | $\overline{\text{IRQ4}}$ | 24 | H3 | 15 | | |
| | $\overline{\text{IRQ3}}$ | 94 | C5 | 60 | | |
| | $\overline{\text{IRQ2}}$ | 97 | B4 | 61 | | |
| | $\overline{\text{IRQ1}}$ | 8 | D1 | 7 | | |
| | $\overline{\text{IRQ0}}$ | 6 | D3 | 5 | | |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| Address bus | A23 | 5 | C1 | — | Output | These pins output an address. (Supported only by the H8S/2218 Group) |
| | A22 | 4 | C2 | — | | |
| | A21 | 3 | D4 | — | | |
| | A20 | 2 | B1 | — | | |
| | A19 | 98 | A3 | — | | |
| | A18 | 99 | C4 | — | | |
| | A17 | 100 | B3 | — | | |
| | A16 | 1 | B2 | — | | |
| | A15 | 52 | K11 | — | | |
| | A14 | 51 | K10 | — | | |
| | A13 | 50 | L10 | — | | |
| | A12 | 49 | K9 | — | | |
| | A11 | 40 | K6 | — | | |
| | A10 | 39 | L6 | — | | |
| | A9 | 38 | J6 | — | | |
| | A8 | 37 | K5 | — | | |
| | A7 | 20 | H1 | — | | |
| | A6 | 19 | G3 | — | | |
| | A5 | 18 | G2 | — | | |
| | A4 | 17 | G1 | — | | |
| | A3 | 13 | F3 | — | | |
| | A2 | 12 | E2 | — | | |
| | A1 | 11 | E1 | — | | |
| | A0 | 10 | E3 | — | | |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| Data bus | D15 | 79 | B9 | — | I/O | These pins constitute a bi-directional data bus. (Supported only by the H8S/2218 Group) |
| | D14 | 78 | D8 | — | | |
| | D13 | 77 | A10 | — | | |
| | D12 | 76 | B10 | — | | |
| | D11 | 75 | B11 | — | | |
| | D10 | 74 | C10 | — | | |
| | D9 | 73 | D9 | — | | |
| | D8 | 72 | C11 | — | | |
| | D7 | 71 | D10 | — | | |
| | D6 | 70 | E8 | — | | |
| | D5 | 69 | D11 | — | | |
| | D4 | 68 | E9 | — | | |
| | D3 | 67 | E10 | — | | |
| | D2 | 66 | E11 | — | | |
| | D1 | 65 | F8 | — | | |
| | D0 | 64 | F10 | — | | |
| Bus control | $\overline{CS5}$ | 56 | H10 | — | Output | Signals for selecting areas 5 to 0 in the external address space. (Supported only by the H8S/2218 Group) |
| | $\overline{CS4}$ | 63 | F11 | — | | |
| | $\overline{CS3}$ | 25 | J2 | — | | |
| | $\overline{CS2}$ | 26 | K2 | — | | |
| | $\overline{CS1}$ | 27 | L2 | — | | |
| | $\overline{CS0}$ | 28 | H4 | — | | |
| | $\overline{AS}$ | 91 | D6 | — | Output | When this pin is low, it indicates that address output on the address bus is enabled. (Supported only by the H8S/2218 Group) |
| | $\overline{RD}$ | 92 | A5 | — | Output | When this pin is low, it indicates that the external address space can be read. (Supported only by the H8S/2218 Group) |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| Bus control | $\overline{\text{HWR}}$ | 93 | B5 | — | Output | A strobe signal that writes to external address space and indicates that the upper half (D15 to D8) of the data bus is enabled. (Supported only by the H8S/2218 Group) |
| | $\overline{\text{LWR}}$ | 94 | C5 | — | Output | A strobe signal that writes to external address space and indicates that the lower half (D7 to D0) of the data bus is enabled. (Supported only by the H8S/2218 Group) |
| | $\overline{\text{WAIT}}$ | 95 | A4 | — | Input | Requests insertion of a wait state in the bus cycle when accessing external 3-state address space. (Supported only by the H8S/2218 Group) |
| 16-bit timer pulse unit (TPU) | TCLKA | 4 | C2 | 3 | Input | TPU external clock input pins |
| | TCLKB | 5 | C1 | 4 | | |
| | TCLKC | 7 | D2 | 6 | | |
| | TCLKD | 9 | E4 | 8 | | |
| | TIOCA0 | 2 | B1 | 1 | I/O | The TGRA_0 to TGRD_0 input capture input/output compare output/PWM output pins |
| | TIOCB0 | 3 | D4 | 2 | | |
| | TIOCC0 | 4 | C2 | 3 | | |
| | TIOCD0 | 5 | C1 | 4 | | |
| | TIOCA1 | 6 | D3 | 5 | I/O | The TGRA_1 and TGRB_1 input capture input/output compare output/PWM output pins |
| | TIOCB1 | 7 | D2 | 6 | | |
| | TIOCA2 | 8 | D1 | 7 | I/O | The TGRA_2 and TGRB_2 input capture input/output compare output/PWM output pins |
| | TIOCB2 | 9 | E4 | 8 | | |
| Realtime clock (RTC) | TMOW | 21 | G4 | 12 | Output | The divided clock output pin |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| Serial communication interface (SCI) | TxD2 | 100 | B3 | 64 | Output | Data output pins |
| | TxD0 | 22 | H2 | 13 | | |
| | RxD2 | 99 | C4 | 63 | Input | Data input pins |
| | RxD0 | 23 | J1 | 14 | | |
| | SCK2 | 98 | A3 | 62 | I/O | Clock input/output pins |
| | SCK0 | 24 | H3 | 15 | | |
| A/D converter | AN15 | 35 | J5 | 23 | Input | Analog input pins for the A/D converter |
| | AN14 | 36 | L5 | 24 | | |
| | AN3 | 42 | K7 | 26 | | |
| | AN2 | 43 | J7 | 27 | | |
| | AN1 | 44 | L8 | 28 | | |
| | AN0 | 45 | H7 | 29 | | |
| | $\overline{\text{ADTRG}}$ | 94 | C5 | 60 | Input | Pin for input of an external trigger to start A/D conversion |
| | Vref | 41 | H6 | 25 | Input | The reference voltage input pin for the A/D converter. When the A/D converter is not used, this pin should be connected to the system power supply (VCC). |
| Boundary scan (Supported only by the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU) | TMS | 85 | C7 | 54 | Input | Control signal input pin for the boundary scan |
| | TCK | 84 | D7 | 53 | Input | Clock input pin for the boundary scan |
| | TDO | 83 | A8 | 52 | Output | Data output pin for the boundary scan |
| | TDI | 87 | B7 | 56 | Input | Data input pin for the boundary scan |
| | $\overline{\text{TRST}}$ | 86 | A7 | 55 | Input | Reset pin for the TAP controller |
| USB | DrVCC | 31 | J4 | 19 | Input | Power supply pin for the on-chip transceiver. Connect this pin to the system power supply. |
| | DrVSS | 34 | H5 | 22 | Input | Ground pin for the on-chip transceiver. |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| USB | USD+* | 32 | K4 | 20 | I/O | USB data I/O pin |
| | USD-* | 33 | L4 | 21 | | |
| | VBUS* | 29 | K3 | 17 | Input | Connection/disconnection detecting input pin for the USB cable |
| | USPND | 21 | G4 | 12 | Output | USB suspend output<br>This pin is driven high when a transition is made to suspend state. |
| | $\overline{\text{UBPM}}$ | 47 | L9 | 31 | Input | Bus power/self power mode setting Input<br>When the USB is used in bus power mode, this input pin must be fixed low.<br>When the USB is used in self power mode, this input pin must be fixed high. |
| | P36 (PUPD+) | 30 | L3 | 18 | I/O | Use as D+ signal pull-up control pin. |
| I/O port | P17 | 9 | E4 | 8 | I/O | 8-bit I/O pins |
| | P16 | 8 | D1 | 7 | | |
| | P15 | 7 | D2 | 6 | | |
| | P14 | 6 | D3 | 5 | | |
| | P13 | 5 | C1 | 4 | | |
| | P12 | 4 | C2 | 3 | | |
| | P11 | 3 | D4 | 2 | | |
| | P10 | 2 | B1 | 1 | | |
| | P36 | 30 | L3 | 18 | I/O | 4-bit I/O pins |
| | P32 | 24 | H3 | 15 | | (Use P36 as D+ signal pull-up control pin of USB.) |
| | P31 | 23 | J1 | 14 | | |
| | P30 | 22 | H2 | 13 | | |
| | P43 | 42 | K7 | 26 | Input | 4-bit input pins |
| | P42 | 43 | J7 | 27 | | |
| | P41 | 44 | L8 | 28 | | |
| | P40 | 45 | H7 | 29 | | |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| I/O port | P77 | — | — | 52 | I/O | 3-bit I/O pins |
| | P76 | — | — | 53 | | |
| | P75 | — | — | 54 | | |
| | P74 | 55 | H9 | — | | |
| | P71 | 56 | H10 | — | | |
| | P70 | 63 | F11 | — | | |
| | P97 | 35 | J5 | 23 | Input | 2-bit input pins |
| | P96 | 36 | L5 | 24 | | |
| | PA3 | 98 | A3 | 62 | I/O | 4-bit I/O pins for the H8S/2218 Group. 3-bit I/O pins for the H8S/2212 Group. |
| | PA2 | 99 | C4 | 63 | | |
| | PA1 | 100 | B3 | 64 | | |
| | PA0 | 1 | B2 | — | | |
| | PB7 | 52 | K11 | — | I/O | 8-bit I/O pins (Supported only by the H8S/2218 Group) |
| | PB6 | 51 | K10 | — | | |
| | PB5 | 50 | L10 | — | | |
| | PB4 | 49 | K9 | — | | |
| | PB3 | 40 | K6 | — | | |
| | PB2 | 39 | L6 | — | | |
| | PB1 | 38 | J6 | — | | |
| | PB0 | 37 | K5 | — | | |
| | PC7 | 20 | H1 | — | I/O | 8-bit I/O pins (Supported only by the H8S/2218 Group) |
| | PC6 | 19 | G3 | — | | |
| | PC5 | 18 | G2 | — | | |
| | PC4 | 17 | G1 | — | | |
| | PC3 | 13 | F3 | — | | |
| | PC2 | 12 | E2 | — | | |
| | PC1 | 11 | E1 | — | | |
| | PC0 | 10 | E3 | — | | |

| Type | Symbol | Pin No. TFP-100G, TFP-100GV | Pin No. BP-112, BP-112V | Pin No. FP-64E, FP-64EV, TNP-64B, TNP-64BV | I/O | Function |
|---|---|---|---|---|---|---|
| I/O port | PD7 | 79 | B9 | — | I/O | 8-bit I/O pins (Supported only by the H8S/2218 Group) |
| | PD6 | 78 | D8 | — | | |
| | PD5 | 77 | A10 | — | | |
| | PD4 | 76 | B10 | — | | |
| | PD3 | 75 | B11 | — | | |
| | PD2 | 74 | C10 | — | | |
| | PD1 | 73 | D9 | — | | |
| | PD0 | 72 | C11 | — | | |
| | PE7 | 71 | D10 | 48 | I/O | 8-bit I/O pins |
| | PE6 | 70 | E8 | 47 | | |
| | PE5 | 69 | D11 | 46 | | |
| | PE4 | 68 | E9 | 45 | | |
| | PE3 | 67 | E10 | 44 | | |
| | PE2 | 66 | E11 | 43 | | |
| | PE1 | 65 | F8 | 42 | | |
| | PE0 | 64 | F10 | 41 | | |
| | PF7 | 89 | A6 | 58 | I/O | 8-bit I/O pins for the H8S/2218 Group. 3-bit I/O pins for the H8S/2212 Group. |
| | PF6 | 91 | D6 | — | | |
| | PF5 | 92 | A5 | — | | |
| | PF4 | 93 | B5 | — | | |
| | PF3 | 94 | C5 | 60 | | |
| | PF2 | 95 | A4 | — | | |
| | PF1 | 96 | D5 | — | | |
| | PF0 | 97 | B4 | 61 | | |
| | PG4 | 28 | H4 | — | I/O | 4-bit I/O pins for the H8S/2218 Group. 2-bit I/O pins for the H8S/2212 Group. |
| | PG3 | 27 | L2 | — | | |
| | PG2 | 26 | K2 | — | | |
| | PG1 | 25 | J2 | 16 | | |
| | PG0 | — | — | 56 | | |

Note: * Anti-noise measures should be taken to prevent malfunction.

# Section 2 CPU

The H8S/2000 CPU is a high-speed central processing unit with an internal 32-bit architecture that is upward compatible with the H8/300 and H8/300H CPUs. The H8S/2000 CPU has sixteen 16-bit general registers, can address a 16-Mbyte linear address space, and is ideal for realtime control.

This section describes the H8S/2000 CPU. The usable modes and address spaces differ depending on the product. For details on each product, refer to section 3, MCU Operating Modes.

## 2.1 Features

- Upward-compatible with H8/300 and H8/300H CPUs
  - Can execute H8/300 and H8/300H CPU object programs
- General-register architecture
  - Sixteen 16-bit general registers also usable as sixteen 8-bit registers or eight 32-bit registers
- Sixty-five basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
- Eight addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
  - Register indirect with post-increment or pre-decrement [@ERn+ or @–ERn]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Memory indirect [@@aa:8]
- 16-Mbyte address space
  - Program: 16 Mbytes
  - Data: 16 Mbytes
- High-speed operation
  - All frequently-used instructions execute in one or two states
  - 8/16/32-bit register-register add/subtract: 1 state
  - 8 × 8-bit register-register multiply: 12 states
  - 16 ÷ 8-bit register-register divide: 12 states

— 16 × 16-bit register-register multiply: 20 states
— 32 ÷ 16-bit register-register divide: 20 states

- Two CPU operating modes
  — Normal mode*
  — Advanced mode

Note: * Normal mode is not available in this LSI.

- Power-down state
  — Transition to power-down state by SLEEP instruction
  — CPU clock speed selection

### 2.1.1 Differences between H8S/2600 CPU and H8S/2000 CPU

The differences between the H8S/2600 CPU and the H8S/2000 CPU are as shown below.

- Register configuration
  The MAC register is supported only by the H8S/2600 CPU.
- Basic instructions
  The four instructions MAC, CLRMAC, LDMAC, and STMAC are supported only by the H8S/2600 CPU.
- The number of execution states of the MULXU and MULXS instructions

| | | Execution States | |
|---|---|---|---|
| **Instruction** | **Mnemonic** | **H8S/2600** | **H8S/2000** |
| MULXU | MULXU.B Rs, Rd | 3 | 12 |
| | MULXU.W Rs, ERd | 4 | 20 |
| MULXS | MULXS.B Rs, Rd | 4 | 13 |
| | MULXS.W Rs, ERd | 5 | 21 |

In addition, there are differences in address space, CCR and EXR register functions, power-down modes, etc., depending on the model.

### 2.1.2 Differences from H8/300 CPU

In comparison to the H8/300 CPU, the H8S/2000 CPU has the following enhancements.

- More general registers and control registers
  - Eight 16-bit extended registers, and one 8-bit and two 32-bit control registers, have been added.
- Extended address space
  - Normal mode supports the same 64-kbyte address space as the H8/300 CPU.
  - Advanced mode supports a maximum 16-Mbyte address space.
- Enhanced addressing
  - The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Signed multiply and divide instructions have been added.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

### 2.1.3 Differences from H8/300H CPU

In comparison to the H8/300H CPU, the H8S/2000 CPU has the following enhancements.

- Additional control register
  - One 8-bit control registers have been added.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

## 2.2 CPU Operating Modes

The H8S/2000 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte total address space. The mode is selected by the mode pins.

### 2.2.1 Normal Mode

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

- Address Space

  A maximum address space of 64 kbytes can be accessed.
- Extended Registers (En)

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

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

  In normal mode the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The exception vector table in normal mode is shown in figure 2.1. For details of the exception vector table, see section 4, Exception Handling.

  The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In normal mode the operand is a 16-bit (word) operand, providing a 16-bit branch address. Branch addresses can be stored in the top area from H'0000 to H'00FF. Note that this area is also used for the exception vector table.
- Stack Structure

  When the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.2. EXR is not pushed onto the stack in interrupt control mode 0. For details, see section 4, Exception Handling.

Note: Normal mode is not available in this LSI.

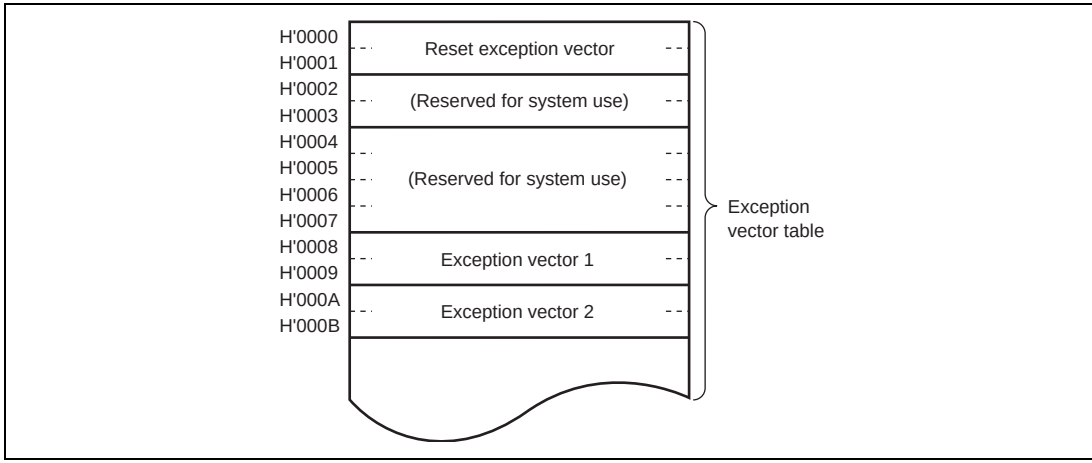
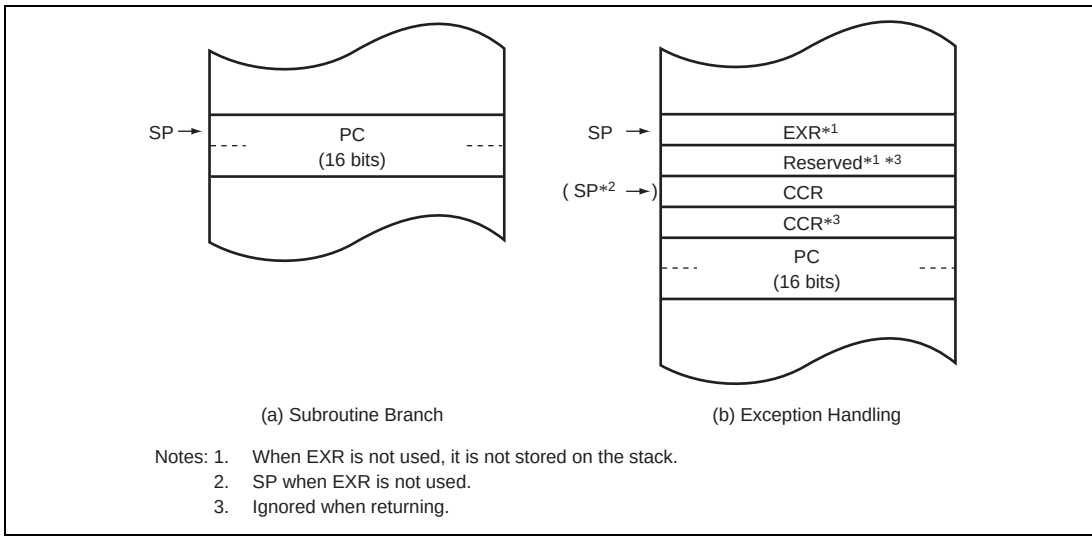



**Figure 2.1 Exception Vector Table (Normal Mode)**




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

**Figure 2.2 Stack Structure in Normal Mode**

### 2.2.2 Advanced Mode

- Address Space

  Linear access is provided to a 16-Mbyte maximum address space.
- Extended Registers (En)

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

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

  In advanced mode the top area starting at H'00000000 is allocated to the exception vector table in units of 32 bits. In each 32 bits, the upper 8 bits are ignored and a branch address is stored in the lower 24 bits (figure 2.3). For details of the exception vector table, see section 4, Exception Handling.

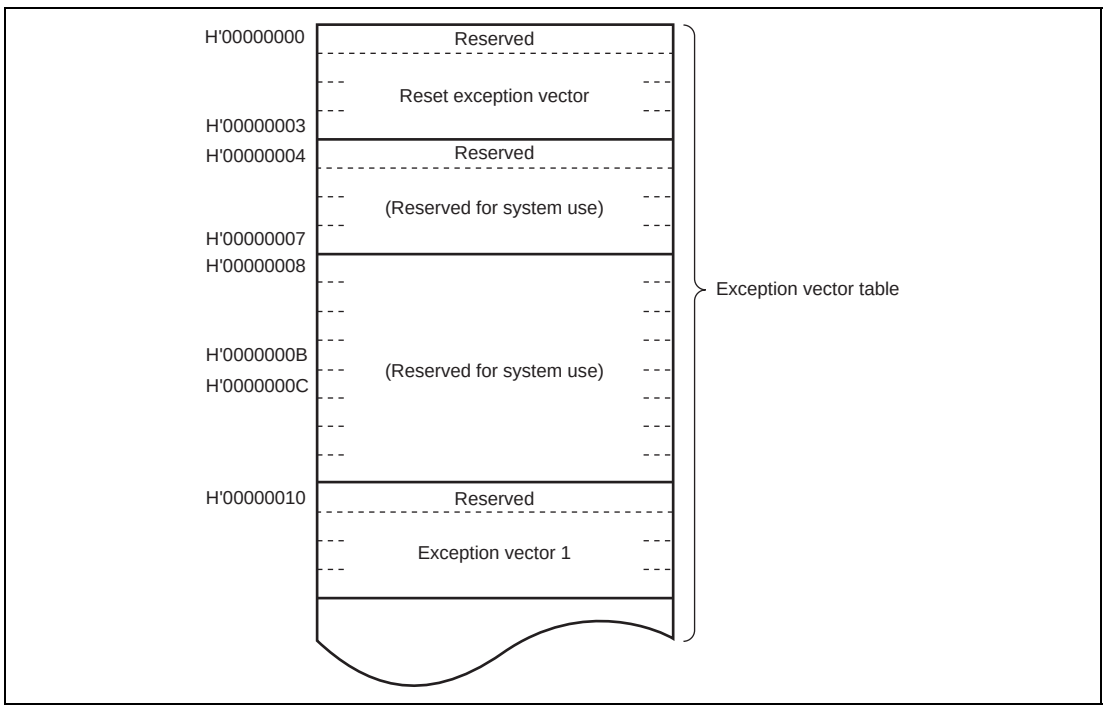



**Figure 2.3 Exception Vector Table (Advanced Mode)**

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In advanced mode the operand is a 32-bit longword operand, providing a 32-bit branch address. The upper 8 bits of these 32 bits are a reserved area that is regarded as H'00. Branch addresses can be stored in the area from H'00000000 to H'000000FF. Note that the first part of this range is also the exception vector table.

- Stack Structure

  In advanced mode, when the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.4. When EXR is invalid, it is not pushed onto the stack. For details, see section 4, Exception Handling.

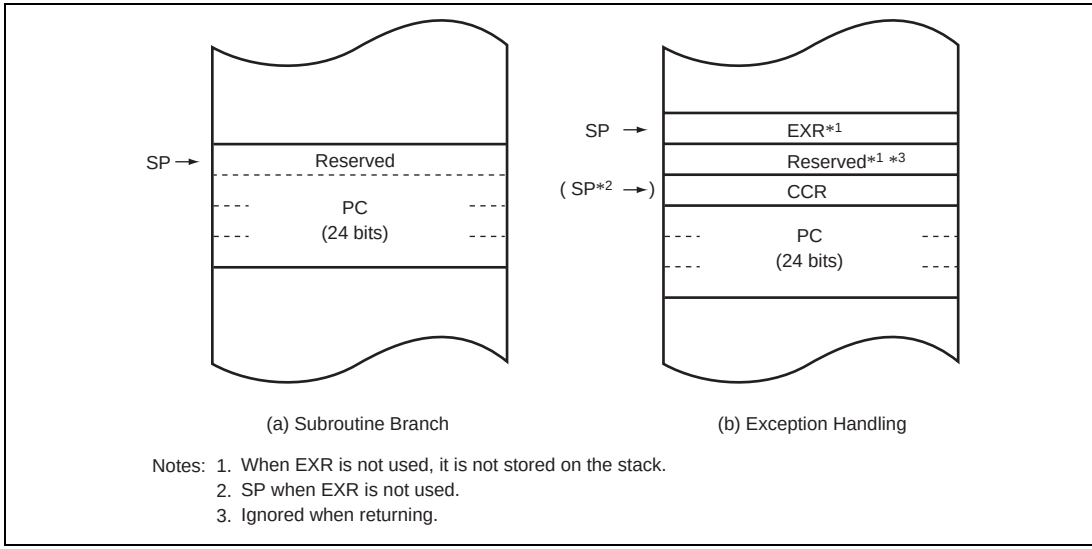



**Figure 2.4 Stack Structure in Advanced Mode**

## 2.3 Address Space

Figure 2.5 shows a memory map of the H8S/2000 CPU. The H8S/2000 CPU provides linear access to a maximum 64-kbyte address space in normal mode, and a maximum 16-Mbyte (architecturally 4-Gbyte) address space in advanced mode. The usable modes and address spaces differ depending on the product. For details on each product, refer to section 3, MCU Operating Modes.

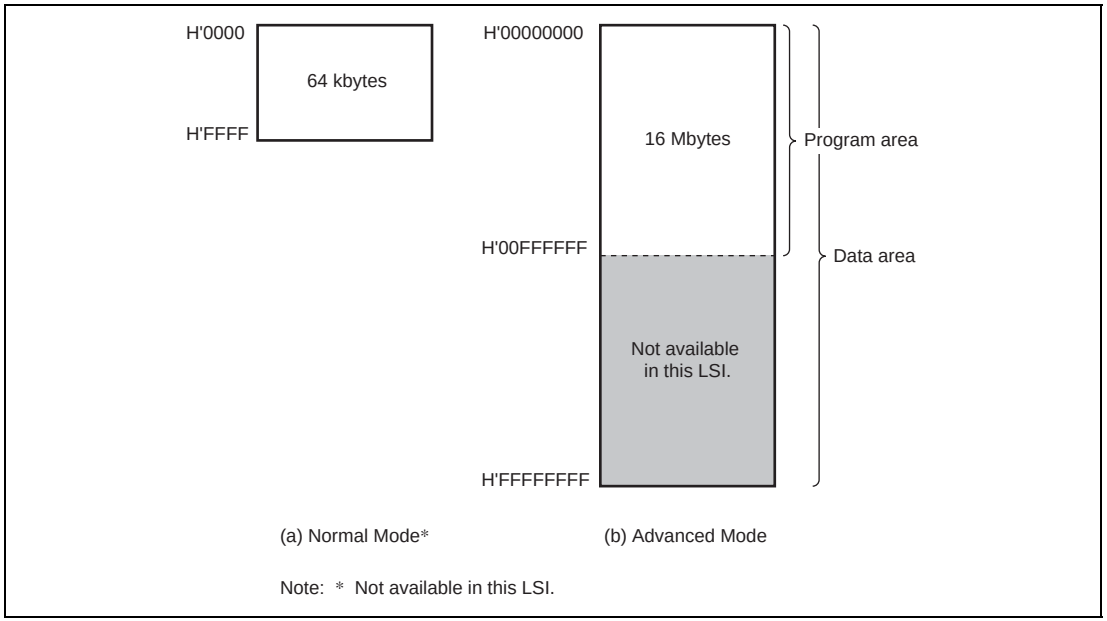



**Figure 2.5 Memory Map**

## 2.4 Register Configuration

The H8S/2000 CPU has the internal registers shown in figure 2.6. There are two types of registers: general registers and control registers. Control registers are a 24-bit program counter (PC), an 8-bit extended control register (EXR), and an 8-bit condition code register (CCR).

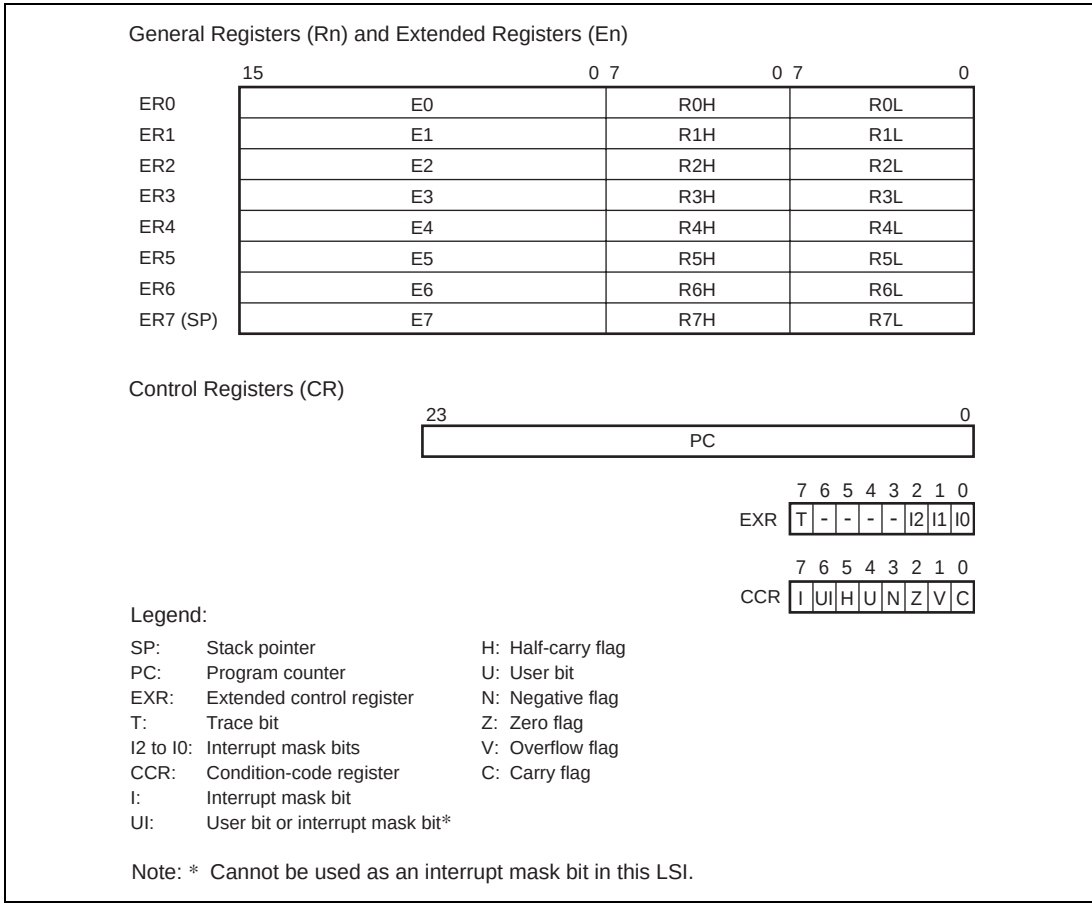



**Figure 2.6 CPU Registers**

### 2.4.1 General Registers

The H8S/2000 CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.7 illustrates the usage of the general registers. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

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

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

The usage of each register can be selected independently.

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2.8 shows the stack.

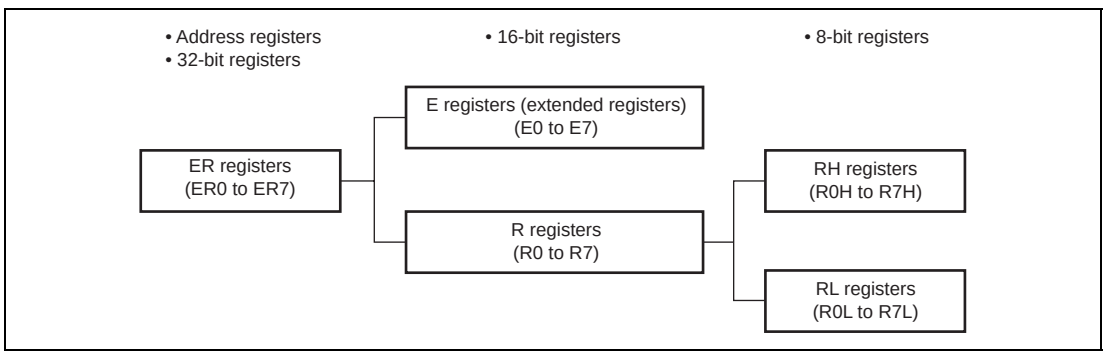



**Figure 2.7 Usage of General Registers**

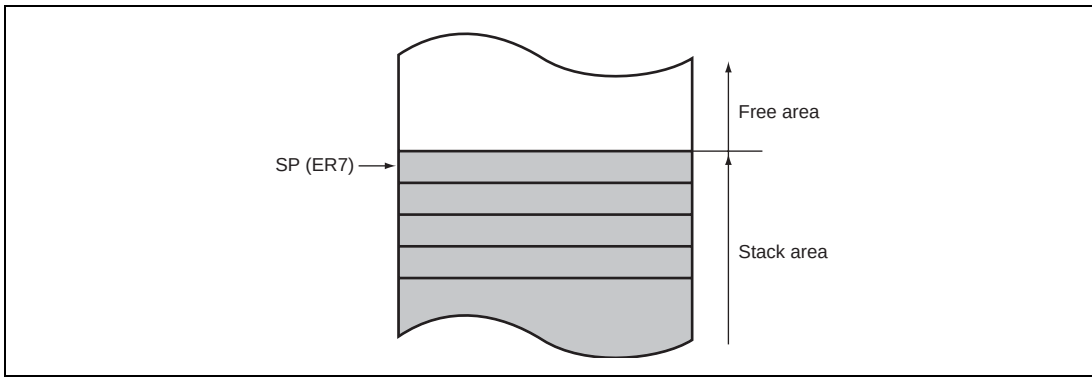



**Figure 2.8 Stack**

### 2.4.2 Program Counter (PC)

This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is two bytes (one word), so the least significant PC bit is ignored. (When an instruction is fetched, the least significant PC bit is regarded as 0.)

### 2.4.3 Extended Control Register (EXR)

EXR is an 8-bit register that manipulates the LDC, STC, ANDC, ORC, and XORC instructions. When these instructions except for the STC instruction is executed, all interrupts including NMI will be masked for three states after execution is completed.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | T | 0 | R/W | Trace Bit<br>When this bit is set to 1, a trace exception is generated each time an instruction is executed. When this bit is cleared to 0, instructions are executed in sequence. |
| 6 to 3 | – | All 1 | – | Reserved<br>These bits are always read as 1. |
| 2 | I2 | 1 | R/W | These bits designate the interrupt mask level (0 to 7). For details, refer to section 5, Interrupt Controller. |
| 1 | I1 | | | |
| 0 | I0 | | | |

### 2.4.4 Condition-Code Register (CCR)

This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

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

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | I | 1 | R/W | Interrupt Mask Bit<br>Masks interrupts other than NMI when set to 1. NMI is accepted regardless of the I bit setting. The I bit is set to 1 by hardware at the start of an exception-handling sequence. For details, refer to section 5, Interrupt Controller. |
| 6 | UI | undefined | R/W | User Bit or Interrupt Mask Bit<br>Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit cannot be used as an interrupt mask bit in this LSI. |
| 5 | H | undefined | R/W | Half-Carry Flag<br>When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise. |
| 4 | U | undefined | R/W | User Bit<br>Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions. |
| 3 | N | undefined | R/W | Negative Flag<br>Stores the value of the most significant bit of data as a sign bit. |
| 2 | Z | undefined | R/W | Zero Flag<br>Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 1 | V | undefined | R/W | Overflow Flag<br>Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times. |
| 0 | C | undefined | R/W | Carry Flag<br>Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:<br>• Add instructions, to indicate a carry<br>• Subtract instructions, to indicate a carry<br>• Shift and rotate instructions, to indicate a carry<br>They carry flag is also used as a bit accumulator by bit manipulation instructions. |

### 2.4.5 Initial Register Values

Reset exception handling loads the CPU's program counter (PC) from the vector table, clears the trace bit in EXR to 0, and sets the interrupt mask bits in CCR and EXR to 1. The other CCR bits and the general registers are not initialized. In particular, the stack pointer (ER7) is not initialized.

The stack pointer should therefore be initialized by an MOV.L instruction executed immediately after a reset.

## 2.5 Data Formats

The H8S/2000 CPU can process 1-bit, 4-bit (BCD), 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

### 2.5.1 General Register Data Formats

Figure 2.9 shows the data formats in general registers.

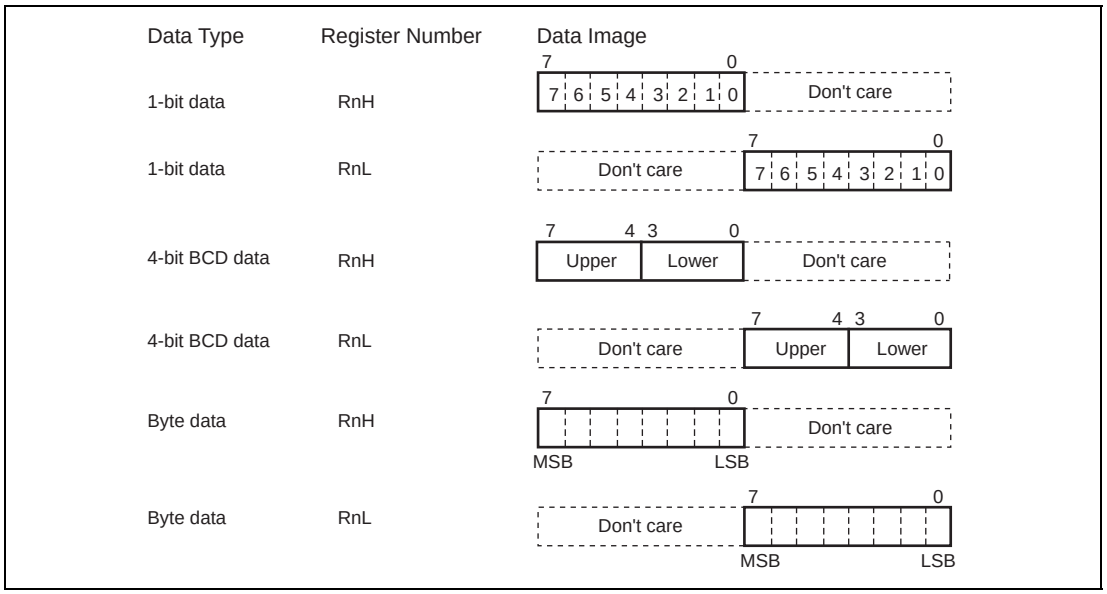



**Figure 2.9 General Register Data Formats (1)**

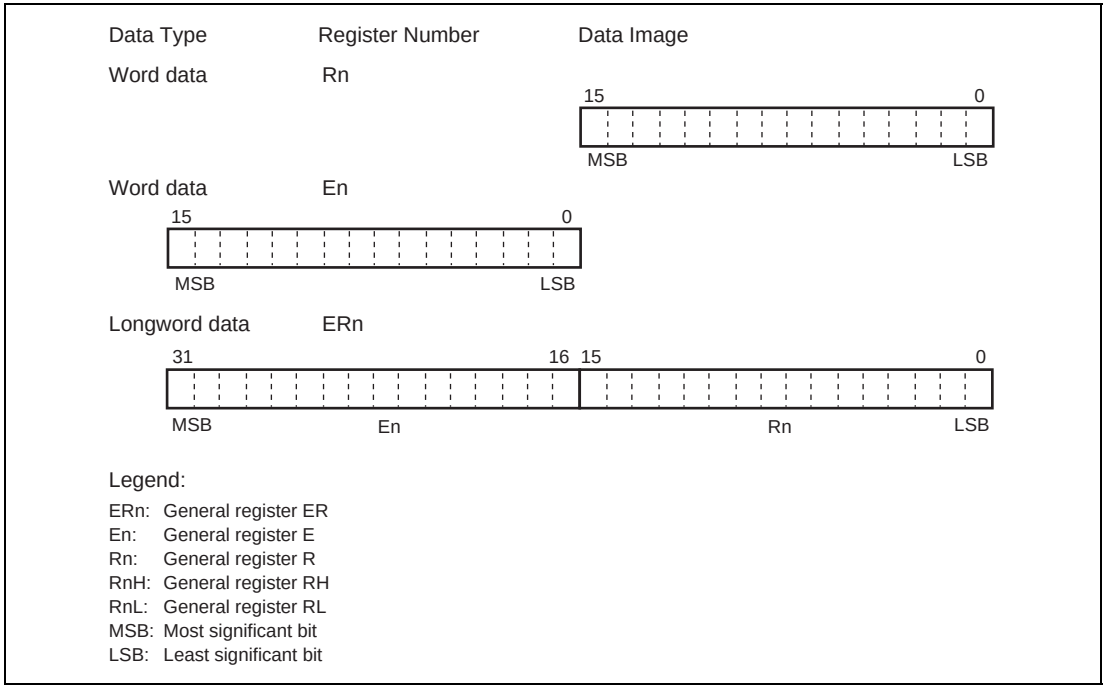



**Figure 2.9 General Register Data Formats (2)**

### 2.5.2 Memory Data Formats

Figure 2.10 shows the data formats in memory. The H8S/2000 CPU can access word data and longword data in memory, but word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, no address error occurs but the least significant bit of the address is regarded as 0, so the access starts at the preceding address. This also applies to instruction fetches.

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

**Figure 2.10 Memory Data Formats**

## 2.6 Instruction Set

The H8S/2000 CPU has 65 types of instructions. The instructions are classified by function in table 2.1.

**Table 2.1 Instruction Classification**

| Function | Instructions | Size | Types |
|---|---|---|---|
| Data transfer | MOV | B/W/L | 5 |
| | POP*[1], PUSH*[1] | W/L | |
| | LDM*[5], STM*[5] | L | |
| | MOVFPE*[3], MOVTPE*[3] | B | |
| Arithmetic operations | ADD, SUB, CMP, NEG | B/W/L | 19 |
| | ADDX, SUBX, DAA, DAS | B | |
| | INC, DEC | B/W/L | |
| | ADDS, SUBS | L | |
| | MULXU, DIVXU, MULXS, DIVXS | B/W | |
| | EXTU, EXTS | W/L | |
| | TAS*[4] | B | |
| Logic operations | AND, OR, XOR, NOT | B/W/L | 4 |
| Shift | SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR | B/W/L | 8 |
| Bit manipulation | BSET, BCLR, BNOT, BTST, BLD, BILD, BST, BIST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR | B | 14 |
| Branch | Bcc*[2], JMP, BSR, JSR, RTS | – | 5 |
| System control | TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP | – | 9 |
| Block data transfer | EEPMOV | – | 1 |
| | | Total: | 65 |

Legend: B: Byte size
W: Word size
L: Longword size

Notes: 1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @-SP. POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @-SP.
2. Bcc is the general name for conditional branch instructions.
3. Cannot be used in this LSI.
4. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.
5. ER7 is used as a stack pointer in STM and LDM instructions. ER7, therefore, should not be used as a saving (STM) or restoring (LDM) register.

### 2.6.1 Table of Instructions Classified by Function

Tables 2.3 to 2.10 summarizes the instructions in each functional category. The notation used in tables 2.3 to 2.10 is defined below.

**Table 2.2 Operation Notation**

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

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

**Table 2.3 Data Transfer Instructions**

| Instruction | Size*[1] | Function |
|---|---|---|
| MOV | B/W/L | (EAs) → Rd, Rs → (EAd)<br>Moves data between two general registers or between a general register and memory, or moves immediate data to a general register. |
| MOVFPE | B | Cannot be used in this LSI. |
| MOVTPE | B | Cannot be used in this LSI. |
| POP | W/L | @SP+ → Rn<br>Pops a general register from the stack. POP.W Rn is identical to MOV.W @SP+, Rn. POP.L ERn is identical to MOV.L @SP+, ERn |
| PUSH | W/L | Rn → @-SP<br>Pushes a general register onto the stack. PUSH.W Rn is identical to MOV.W Rn, @-SP. PUSH.L ERn is identical to MOV.L ERn, @-SP. |
| LDM*[2] | L | @SP+ → Rn (register list)<br>Pops two or more general registers from the stack. |
| STM*[2] | L | Rn (register list) → @-SP<br>Pushes two or more general registers onto the stack. |

Notes: 1. Size refers to the operand size.
B: Byte
W: Word
L: Longword
2. ER7 is used as a stack pointer in STM and LDM instructions. ER7, therefore, should not be used as a saving (STM) or restoring (LDM) register.

**Table 2.4 Arithmetic Operations Instructions**

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

| Instruction | Size*[1] | Function |
|---|---|---|
| NEG | B/W/L | 0 – Rd → Rd<br>Takes the two's complement (arithmetic complement) of data in a general register. |
| EXTU | W/L | Rd (zero extension) → Rd<br>Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by padding with zeros on the left. |
| EXTS | W/L | Rd (sign extension) → Rd<br>Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by extending the sign bit. |
| TAS*[2] | B | @ERd – 0, 1 → (<bit 7> of @ERd)<br>Tests memory contents, and sets the most significant bit (bit 7) to 1. |

Notes: 1. Size refers to the operand size.
B: Byte
W: Word
L: Longword
2. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

**Table 2.5 Logic Operations Instructions**

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

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

**Table 2.6 Shift Instructions**

| Instruction | Size* | Function |
|---|---|---|
| SHAL | B/W/L | Rd (shift) → Rd |
| SHAR | | Performs an arithmetic shift on general register contents. 1-bit or 2 bit shift is possible. |
| SHLL | B/W/L | Rd (shift) → Rd |
| SHLR | | Performs an logical shift on general register contents. 1-bit or 2 bit shift is possible. |
| ROTL | B/W/L | Rd (rotate) → Rd |
| ROTR | | Rotates general register contents. 1-bit or 2 bit rotation is possible. |
| ROTXL | B/W/L | Rd (rotate) → Rd |
| ROTXR | | Rotates general register contents through the carry flag. 1-bit or 2 bit rotation is possible. |

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

**Table 2.7 Bit Manipulation Instructions**

| Instruction | Size* | Function |
|---|---|---|
| BSET | B | 1 → (<bit-No.> of <EAd>)<br>Sets a specified bit in a general register or memory operand to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| BCLR | B | 0 → (<bit-No.> of <EAd>)<br>Clears a specified bit in a general register or memory operand to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| BNOT | B | ~ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>)<br>Inverts a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| BTST | B | ~ (<bit-No.> of <EAd>) → Z<br>Tests a specified bit in a general register or memory operand and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register. |
| BAND | B | C ∧ (<bit-No.> of <EAd>) → C<br>ANDs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| BIAND | B | C ∧ ~ (<bit-No.> of <EAd>) → C<br>ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data. |
| BOR | B | C ∨ (<bit-No.> of <EAd>) → C<br>ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| BIOR | B | C ∨ ~ (<bit-No.> of <EAd>) → C<br>ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data. |

Note: * Size refers to the operand size.
B: Byte

| Instruction | Size* | Function |
|---|---|---|
| BXOR | B | C ⊕ (<bit-No.> of <EAd>) → C<br>Exclusive-ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag. |
| BIXOR | B | C ⊕ ~ (<bit-No.> of <EAd>) → C<br>Exclusive-ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data. |
| BLD | B | (<bit-No.> of <EAd>) → C<br>Transfers a specified bit in a general register or memory to the carry flag. |
| BILD | B | ~ (<bit-No.> of <EAd>) → C<br>Transfers the inverse of a specified bit in a general register or memory operand to the carry flag. The bit number is specified by 3-bit immediate data. |
| BST | B | C → (<bit-No.> of <EAd>)<br>Transfers the carry flag value to a specified bit in a general register or memory operand. |
| BIST | B | ~ C → (<bit-No.> of <EAd>)<br>Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data. |

Note:* Size refers to the operand size.
B: Byte

**Table 2.8 Branch Instructions**

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

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

| | | |
|---|---|---|
| JMP | – | Branches unconditionally to a specified address. |
| BSR | – | Branches to a subroutine at a specified address |
| JSR | – | Branches to a subroutine at a specified address |
| RTS | – | Returns from a subroutine |

**Table 2.9 System Control Instruction**

| Instruction | Size* | Function |
|---|---|---|
| TRAPA | – | Starts trap-instruction exception handling. |
| RTE | – | Returns from an exception-handling routine. |
| SLEEP | – | Causes a transition to a power-down state. |
| LDC | B/W | (EAs) → CCR, (EAs) → EXR<br>Moves the source operand contents or immediate data to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid. |
| STC | B/W | CCR → (EAd), EXR → (EAd)<br>Transfers CCR or EXR contents to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid. |
| ANDC | B | CCR ∧ #IMM → CCR, EXR ∧ #IMM → EXR<br>Logically ANDs the CCR or EXR contents with immediate data. |
| ORC | B | CCR ∨ #IMM → CCR, EXR ∨ #IMM → EXR<br>Logically ORs the CCR or EXR contents with immediate data. |
| XORC | B | CCR ⊕ #IMM → CCR, EXR ⊕ #IMM → EXR<br>Logically exclusive-ORs the CCR or EXR contents with immediate data. |
| NOP | – | PC + 2 → PC<br>Only increments the program counter. |

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

**Table 2.10 Block Data Transfer Instruction**

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

### 2.6.2 Basic Instruction Formats

The H8S/2000 CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op), a register field (r), an effective address extension (EA), and a condition field (cc).

Figure 2.11 shows examples of instruction formats.

- Operation Field
  Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.
- Register Field
  Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.
- Effective Address Extension
  8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.
- Condition Field
  Specifies the branching condition of Bcc instructions.

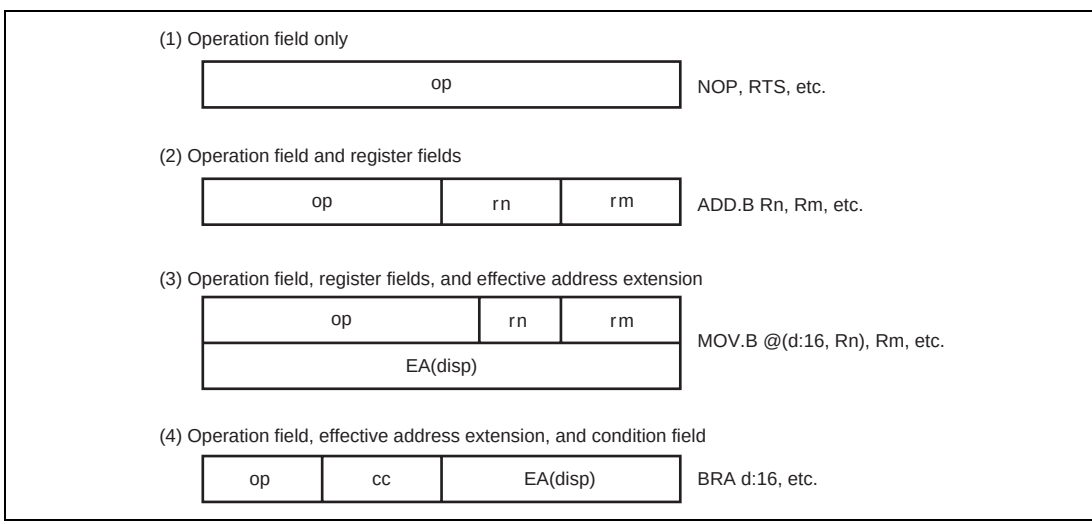



**Figure 2.11 Instruction Formats (Examples)**

## 2.7 Addressing Modes and Effective Address Calculation

The H8S/2000 CPU supports the eight addressing modes listed in table 2.11. Each instruction uses a subset of these addressing modes. Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

**Table 2.11 Addressing Modes**

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

### 2.7.1 Register Direct—Rn

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

### 2.7.2 Register Indirect—@ERn

The register field of the instruction code specifies an address register (ERn) which contains the address of the operand on memory. If the address is a program instruction address, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

### 2.7.3 Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn)

A 16-bit or 32-bit displacement contained in the instruction is added to an address register (ERn) specified by the register field of the instruction, and the sum gives the address of a memory operand. A 16-bit displacement is sign-extended when added.

### 2.7.4 Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn

**Register Indirect with Post-Increment—@ERn+:** The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word transfer instruction, or 4 for longword transfer instruction. For word or longword transfer instruction, the register value should be even.

**Register Indirect with Pre-Decrement—@-ERn:** The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the result becomes the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word transfer instruction, or 4 for longword transfer instruction. For word or longword transfer instruction, the register value should be even.

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

The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24), or 32 bits long (@aa:32). Table 2.12 indicates the accessible absolute address ranges.

To access data, the absolute address should be 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) long. For an 8-bit absolute address, the upper 24 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address the upper 16 bits are a sign extension. A 32-bit absolute address can access the entire address space.

A 24-bit absolute address (@aa:24) indicates the address of a program instruction. The upper 8 bits are all assumed to be 0 (H'00).

**Table 2.12 Absolute Address Access Ranges**

| Absolute Address | | Normal Mode* | Advanced Mode |
|---|---|---|---|
| Data address | 8 bits (@aa:8) | H'FF00 to H'FFFF | H'FFFF00 to H'FFFFFF |
| | 16 bits (@aa:16) | H'0000 to H'FFFF | H'000000 to H'007FFF, H'FF8000 to H'FFFFFF |
| | 32 bits (@aa:32) | | H'000000 to H'FFFFFF |
| Program instruction address | 24 bits (@aa:24) | | |

Note: * Not available in this LSI.

### 2.7.6 Immediate—#xx:8, #xx:16, or #xx:32

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

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

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

This mode is used in the Bcc and BSR instructions. An 8-bit or 16-bit displacement contained in the instruction is sign-extended and added to the 24-bit PC contents to generate a branch address.

Only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00). The PC value to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is –126 to +128 bytes (–63 to +64 words) or –32766 to +32768 bytes (–16383 to +16384 words) from the branch instruction. The resulting value should be an even number.

### 2.7.8 Memory Indirect—@@aa:8

This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand. This memory operand contains a branch address.

The upper bits of the absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode*, H'000000 to H'0000FF in advanced mode). In normal mode the memory operand is a word operand and the branch address is 16 bits long. In advanced mode the memory operand is a longword operand, the first byte of which is assumed to be H'00.

Note that the first part of the address range is also the exception vector area. For further details, refer to section 4, Exception Handling.

If an odd address is specified in word or longword memory access, or as a branch address, the least significant bit is regarded as 0, causing data to be accessed or instruction code to be fetched at the address preceding the specified address. (For further information, see section 2.5.2, Memory Data Formats.)

Note: * Not available in this LSI.

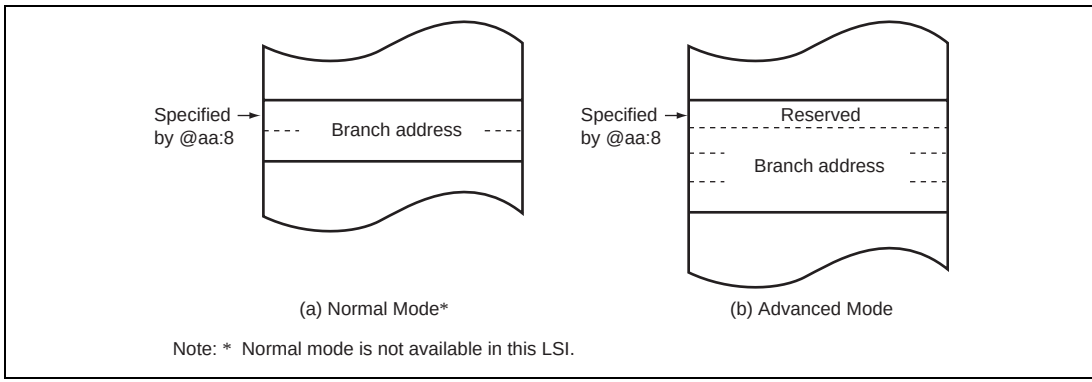



**Figure 2.12 Branch Address Specification in Memory Indirect Mode**

### 2.7.9 Effective Address Calculation

Table 2.13 indicates how effective addresses are calculated in each addressing mode. In normal mode the upper 8 bits of the effective address are ignored in order to generate a 16-bit address.

**Table 2.13 Effective Address Calculation**

| No | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 1 | Register direct (Rn)<br>op \| rm \| rn | | Operand is general register contents. |
| 2 | Register indirect (@ERn)<br>op \| r | 31 … 0: General register contents | 31 … 24 23 … 0: Don't care |
| 3 | Register indirect with displacement @(d:16,ERn) or @(d:32,ERn)<br>op \| r \| disp | 31 … 0: General register contents<br>31 … 0: Sign extension \| disp | 31 … 24 23 … 0: Don't care |
| 4 | Register indirect with post-increment or pre-decrement<br>•Register indirect with post-increment @ERn+<br>op \| r | 31 … 0: General register contents<br>1, 2, or 4 | 31 … 24 23 … 0: Don't care |
| | •Register indirect with pre-decrement @-ERn<br>op \| r | 31 … 0: General register contents<br>1, 2, or 4 | 31 … 24 23 … 0: Don't care |

| Operand Size | Offset |
|---|---|
| Byte | 1 |
| Word | 2 |
| Longword | 4 |

| No | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|---|---|---|---|
| 5 | Absolute address<br>@aa:8<br>op abs<br>@aa:16<br>op abs<br>@aa:24<br>op abs<br>@aa:32<br>op<br>abs | | 31 24 23 8 7 0<br>Don't care H'FFFF<br>31 24 23 16 15 0<br>Don't care Sign extension<br>31 24 23 0<br>Don't care<br>31 24 23 0<br>Don't care |
| 6 | Immediate<br>#xx:8/#xx:16/#xx:32<br>op IMM | | Operand is immediate data. |
| 7 | Program-counter relative<br>@(d:8,PC)/@(d:16,PC)<br>op disp | 23 0<br>PC contents<br>23 0<br>Sign extension disp | 31 24 23 0<br>Don't care |
| 8 | Memory indirect @@aa:8<br>•Normal mode*<br>op abs<br>•Advanced mode<br>op abs | 31 8 7 0<br>H'000000 abs<br>15 0<br>Memory contents<br>31 8 7 0<br>H'000000 abs<br>31 0<br>Memory contents | 31 24 23 16 15 0<br>Don't care H'00<br>31 24 23 0<br>Don't care |

Note: * Normal mode is not available in this LSI.

## 2.8 Processing States

The H8S/2000 CPU has five main processing states: the reset state, exception handling state, program execution state, bus-released state, and power-down state. Figure 2.13 indicates the state transitions.

- Reset State

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

  The reset state can also be entered by a watchdog timer overflow.

- Exception-Handling State

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

- Program Execution State

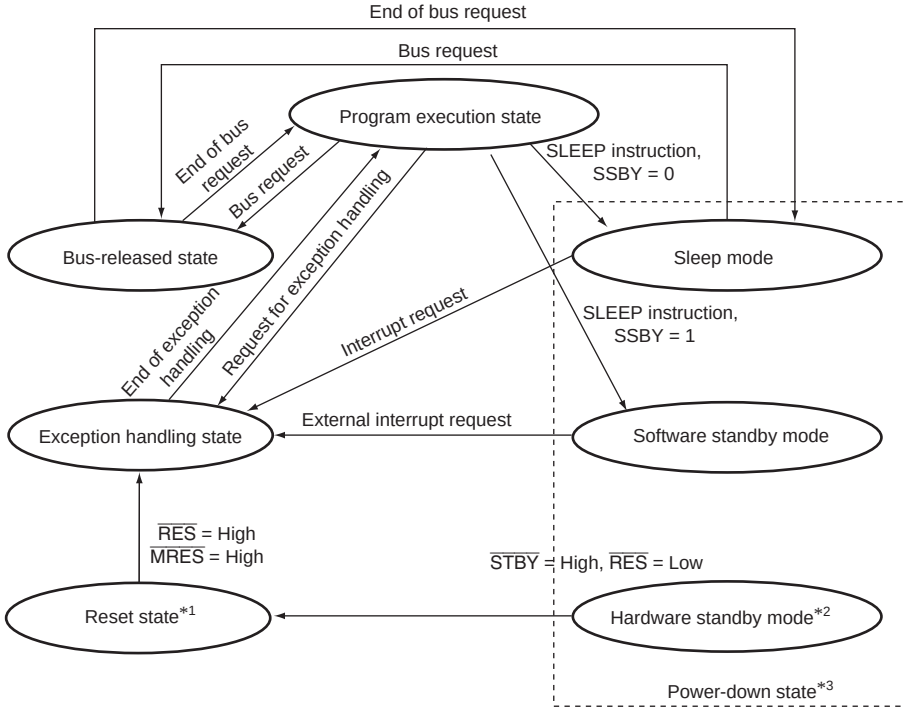
  In this state the CPU executes program instructions in sequence.

- Bus-Released State

  In a product which has a bus master other than the CPU, such as a direct memory access controller (DMAC) and a data transfer controller (DTC), the bus-released state occurs when the bus has been released in response to a bus request from a bus master other than the CPU. While the bus is released, the CPU halts operations.

- Power-Down State

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




Notes: 1. From any state except hardware standby mode, a transition to the reset state occurs whenever $\overline{RES}$ goes low. A transition can also be made to the reset state when the watchdog timer overflows. From any state except hardware standby mode and power-on reset state, a transition to the manual reset state occurs whenever $\overline{MRES}$ goes low.
2. From any state, a transition to hardware standby mode occurs when $\overline{STBY}$ goes low.
3. Apart from these states, there are also the watch mode, subactive mode, and the subsleep mode. See section 20, Power-Down Modes.

**Figure 2.13 State Transitions**

## 2.9 Usage Notes

### 2.9.1 Note on TAS Instruction Usage

Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction. The TAS instruction is not generated by the Renesas Technology H8S and H8/300 Series C/C++ compilers. If the TAS instruction is used as a user-defined intrinsic function, ensure that only register ER0, ER1, ER4, or ER5 is used.

### 2.9.2 STM/LTM Instruction Usage

With the STM or LDM instruction, the ER7 register is used as the stack pointer, and thus cannot be used as a register that allows save (STM) or restore (LDM) operation.

With a single STM or LDM instruction, two to four registers can be saved or restored. The available registers are as follows:

For two registers: ER0 and ER1, ER2 and ER3, or ER4 and ER5

For three registers: ER0 to ER2, or ER4 to ER6

For four registers: ER0 to ER3

For the Renesas Technology H8S or H8/300 Series C/C++ Compiler, the STM/LDM instruction including ER7 is not created.

### 2.9.3 Note on Bit Manipulation Instructions

Using bit manipulation instructions on registers containing write-only bits can result in the bits that should have been manipulated not being manipulated as intended or in the wrong bits being manipulated.

Reading data from a register containing write-only bits may return fixed or undefined values. Consequently, bit manipulation instructions that use the read values to perform operations (BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, and BILD) will not work properly.

In addition, bit manipulation instructions that write data following operations based on the data values read (BSET, BCLR, BNOT, BST, and BIST) may change the values of bits unrelated to the intended bit manipulation. Therefore, caution is necessary when using bit manipulation instructions on registers containing write-only bits.

The instructions BSET, BCLR, BNOT, BST, and BIST perform the following operations in the order shown:

1. Read data in byte units
2. Perform bit manipulation on the read data according to the instruction
3. Write data in byte units

Example: Using the BCLR instruction to clear pin 14 only of P1DDR for port 1

P1DDR is an 8-bit register that contains write-only bits. It is used to specify the I/O setting of the individual pins in port 1. Reading produces invalid data. Attempting to read from P1DDR returns undefined values.

In this example, the BCLR instruction is used to set pin 14 as an input port. Let us assume that pins 17 to 14 are presently set as output pins and pins 13 to 10 are set as input pins. Thus, the value of P1DDR is initially H'F0.

| | **P17** | **P16** | **P15** | **P14** | **P13** | **P12** | **P11** | **P10** |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

To change pin 14 from an output pin to an input pin, the value of bit 4 in P1DDR must be changed from 1 to 0 (H'F0 to H'E0). Now assume that the BCLR instruction is used to clear bit 4 in P1DDR to 0.

```
BCLR    #4, @P1DDR
```

However, using the above bit manipulation instruction on the write-only register P1DDR can cause problems, as described below.

The BCLR instruction first reads data from P1DDR in byte units, but in this case the read values are undefined. These undefined values can be 0 or 1 for each bit in the register, but there is no way of telling which. Since all of the bits in P1DDR are write-only, undefined values are returned for all of the bits when the register is read. In this example the value of P1DDR is H'F0, but we will assume that the value returned when the register was read is H'F8, which would give bit 3 a value of 1.

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read value | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

The BCLR instruction performs bit manipulation on the read value, which is H'F8 in this example. It clears bit 4 to 0.

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| After bit manipulation | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

Following bit manipulation the data is written to P1DDR and the BCLR instruction terminates.

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Input | Output | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| Write value | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

The contents of P1DDR should have been overwritten with a value of H'E0, but in fact a value of H'E8 was written to the register. This changed pin 13, which should have been an input pin, to an output pin. In this example we assumed that the value of bit 1 in P1DDR was 1. However, since the values of bits 7 to 0 in P1DDR are all undefined when read, there is the possibility that individual bit values could be changed from 0 to 1 or from 1 to 0. To prevent this from happening, the recommendations in section 2.9.4, Accessing Registers Containing Write-Only Bits, should be followed when changing the values of registers containing write-only bits.

In addition, the BCLR instruction can be used to clear flags in internal I/O registers to 0. In such cases it is not necessary to read the relevant flag beforehand so long as it is clear that it has been set to 1 by an interrupt processing routine or the like.

### 2.9.4 Accessing Registers Containing Write-Only Bits

Using data transfer instructions or bit manipulation instructions on registers containing write-only bits can result in undefined values being read. To prevent the reading of undefined values, the procedure described below should be used to access registers containing write-only bits.

In order to write to a register containing write-only bits, set aside a work area in memory (in on-chip RAM, for example) and write the data to be manipulated to it. After accessing and manipulating the data in the work area in memory, write the resulting data to the register containing write-only bits.

Figure 2.14 Example Flowchart of Method for Accessing Registers Containing Write-Only Bits

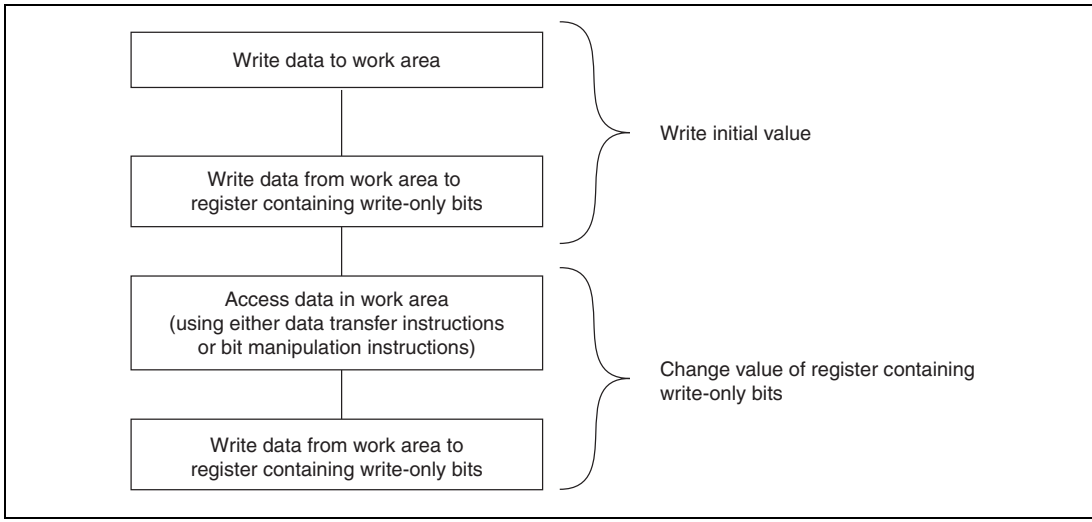



**Figure 2.14 Flowchart of Method for Accessing Registers Containing Write-Only Bits**

Example: Clearing pin 14 only of P1DDR for port 1

P1DDR is an 8-bit register that contains write-only bits. It is used to specify the I/O setting of the individual pins in port 1. Reading produces invalid data. Attempting to read from P1DDR returns undefined values.

In this example, the BCLR instruction is used to set pin 14 as an input port. To start, the initial value H'F0 to be written to P1DDR is written ahead of time to the work area (RAM0) in memory.

```
MOV.B    #H'F0,  R0L
MOV.B    R0L,    @RAM0
MOV.B    R0L,    @P1DDR
```

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | | | | | | | | |
| RAM0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |

To change pin 14 from an output pin to an input pin, the value of bit 4 in P1DDR must be changed from 1 to 0 (H'F0 to H'E0). Here the BCLR instruction will be used to clear bit 4 in P1DDR to 0.

```
BCLR    #4,     @RAM0
```

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| | | | | | | | | |
| RAM0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

Since RAM0 is a read/write area of memory, performing the above bit manipulation using the BCLR instruction causes only bit 4 in RAM0 to be cleared to 0. The value of RAM0 is then written to P1DDR.

```
MOV.B   @RAM0,  R0L
MOV.B   R0L,    @P1DDR
```

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Input | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| | | | | | | | | |
| RAM0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

By using the above procedure to access registers containing write-only bits, it is possible to create programs that are not dependent on the type of instructions used.

# Section 3 MCU Operating Modes

## 3.1 Operating Mode Selection

This LSI supports four operating modes (modes 4 to 7). These modes enable selection of the CPU operating mode, enabling/disabling of on-chip ROM, and the initial bus width setting, by setting the mode pins (MD2 to MD0) as show in table 3.1. Modes 4 to 6 are external extended modes that allow access to the external memory and peripheral devices. In external extended mode, 8-bit or 16-bit address space can be set for each area depending on the bus controller setting after program execution starts. If 16-bit access is selected for any one area, 16-bit bus mode is set; if 8-bit access is selected for all areas, 8-bit bus mode is set. In mode 7, the external address space cannot be used. Do not change the mode pin settings during operation. Only mode 7 is available in the H8S/2212 Group.

**Table 3.1 MCU Operating Mode Selection**

| MCU Operating Mode | MD2 | MD1 | MD0 | CPU Operating Mode | Description | On-chip ROM | External Data Bus | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | Initial Value | Maximum Value |
| 4 | 1 | 0 | 0 | Advanced mode | On-chip ROM disabled, extended mode | Disabled | 16 bits | 16 bits |
| 5 | 1 | 0 | 1 | Advanced mode | On-chip ROM disabled, extended mode | Disabled | 8 bits | 16 bits |
| 6 | 1 | 1 | 0 | Advanced mode | On-chip ROM enabled, extended mode | Enabled | 8 bits | 16 bits |
| 7 | 1 | 1 | 1 | Advanced mode | Single-chip mode | Enabled | – | – |

Note: When using the E6000 emulator:

- Mode 7 is not available in the H8S/2218 Group. (The E6000 emulator does not support mode 7.)
- Note following restrictions to use the RTC and USB in mode 6.
  Specify PFCR so that A9 and A8 are output on the PB1 and PB0 pins.
  Set H'FF in PCDDR so that A7 to A0 are output on the PC7 to PC0 pins.

## 3.2 Register Descriptions

The following registers are related to the operating mode.

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

### 3.2.1 Mode Control Register (MDCR)

MDCR is used to monitor the current operating mode of this LSI. MDCR should not be modified.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are always read as undefined value and cannot be modified. |
| 3 | FWE | —*[1] | R | Flash Programming Enable<br>Reflects the input level at the FWE pin. This bit functions same as the FWE bit in the FLMCR1 register. |
| 2 | MDS2 | —*[1] | R | Mode Select 2 to 0<br>These bits indicate the input levels at pins MD2 to MD0 (the current operating mode). Bits MDS2 to MDS0 correspond to MD2 to MD0. MDS2 to MDS0 are read-only bits and they cannot be written to. The mode pin (MD2 to MD0) input levels are latched into these bits when MDCR is read.<br>These latches are canceled by a power-on reset, but maintained at manual reset*[2]. |
| 1 | MDS1 | —*[1] | R | |
| 0 | MDS0 | —*[1] | R | |

Notes: 1. Determined by the FWE and MD2 to MD0 pin settings.
2. Supported only by the H8S/2218 Group.

### 3.2.2 System Control Register (SYSCR)

SYSCR is used to select the interrupt control mode and the detected edge for NMI, select the $\overline{\text{MRES}}$ input pin* enable or disable, and enables or disables on-chip RAM.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | – | 0 | R/W | Reserved<br>The write value should always be 0. |
| 6 | – | 0 | – | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 5<br>4 | INTM1<br>INTM0 | 0<br>0 | R/W<br>R/W | These bits select the control mode of the interrupt controller. For details of the interrupt control modes, see section 5.6, Interrupt Control Modes and Interrupt Operation.<br>00: Interrupt control mode 0<br>01: Setting prohibited<br>10: Interrupt control mode 2<br>11: Setting prohibited |
| 3 | NMIEG | 0 | R/W | NMI Edge Select<br>Selects the valid edge of the NMI interrupt input.<br>0: An interrupt is requested at the falling edge of NMI input<br>1: An interrupt is requested at the rising edge of NMI input |
| 2 | MRESE | 0 | R/W | Manual Reset Select<br>Enables or disables the $\overline{\text{MRES}}$ pin* input.<br>0: Manual reset is disabled<br>1: Manual reset is enabled<br>The $\overline{\text{MRES}}$ input pin* can be used. |
| 1 | – | 0 | – | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 0 | RAME | 1 | R/W | RAM Enable<br>Enables or disables the on-chip RAM. The RAME bit is initialized when the reset status is released.<br>0: On-chip RAM is disabled<br>1: On-chip RAM is enabled |

Note: * Supported only by the H8S/2218 Group.

## 3.3 Operating Mode Descriptions

### 3.3.1 Mode 4 (Supported Only by the H8S/2218 Group)

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled.

Pins P13 to P10, and ports A, B, and C function as an address bus, ports D and E function as a data bus, and part of port F carries bus control signals.

Pins P13 to P11 function as input ports immediately after a reset. Pin 10 and ports A and B function as address (A20 to A8) outputs immediately after a reset. Address (A23 to A21) output can be enabled or disabled by bits AE3 to AE0 in the pin function control register (PFCR) regardless of the corresponding data direction register (DDR) values. Pins for which address output is disabled among pins P13 to P10 and in ports A and B become port outputs when the corresponding DDR bits are set to 1.

Port C always has an address (A7 to A0) output function.

The initial bus mode after a reset is 16 bits, with 16-bit access to all areas. However, note that if 8-bit access is designated by the bus controller for all areas, the bus mode switches to 8 bits.

### 3.3.2 Mode 5 (Supported Only by the H8S/2218 Group)

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled.

Pins P13 to P10, and ports A, B, and C function as an address bus, ports D and E function as a data bus, and part of port F carries bus control signals.

Pins P13 to P11 function as input ports immediately after a reset. Pin 10 and ports A and B function as address (A20 to A8) outputs immediately after a reset. Address (A23 to A21) output can be enabled or disabled by bits AE3 to AE0 in the pin function control register (PFCR) regardless of the corresponding data direction register (DDR) values. Pins for which address output is disabled among pins P13 to P10 and in ports A and B become port outputs when the corresponding DDR bits are set to 1.

Port C always has an address (A7 to A0) output function.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, note that if 16-bit access is designated by the bus controller for any area, the bus mode switches to 16 bits and port E becomes a data bus.

### 3.3.3 Mode 6 (Supported Only by the H8S/2218 Group)

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled.

Pins P13 to P10, and ports A, B and C function as input ports immediately after a reset. Address (A23 to A8) output can be enabled or disabled by bits AE3 to AE0 in the pin function control register (PFCR) regardless of the corresponding data direction register (DDR) values. Pins for which address output is disabled among pins P13 to P10 and in ports A and B become port outputs when the corresponding DDR bits are set to 1.

Port C is an input port immediately after a reset. Addresses A7 to A0 are output by setting the corresponding DDR bits to 1.

Ports D and E function as a data bus, and part of port F carries data bus signals.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, note that if 16-bit access is designated by the bus controller for any area, the bus mode switches to 16 bits and port E becomes a data bus.

### 3.3.4 Mode 7

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled, but external addresses cannot be accessed.

All I/O ports are available for use as input-output ports.

### 3.3.5 Pin Functions

The pin functions of ports 1, and A to F vary depending on the operating mode. Table 3.2 shows their functions in each operating mode.

**Table 3.2 Pin Functions in Each Operating Mode**

| Port | | Mode 4 | Mode 5 | Mode 6 | Mode 7 |
|---|---|---|---|---|---|
| Port 1 | P13 to P11 | P*/A | P*/A | P*/A | P |
| | P10 | P/A* | P/A* | P*/A | P |
| Port A | PA3 to PA0 | P/A* | P/A* | P*/A | P |
| Port B | | P/A* | P/A* | P*/A | P |
| Port C | | A | A | P*/A | P |
| Port D | | D | D | D | P |
| Port E | | P/D* | P*/D | P*/D | P |
| Port F | PF7 | P/C* | P/C* | P/C* | P*/C |
| | PF6 to PF4 | C | C | C | P |
| | PF3 | P/C* | P*/C | P*/C | |
| | PF2 to PF0 | P*/C | P*/C | P*/C | |

Legend:

P: I/O port

A: Address bus output

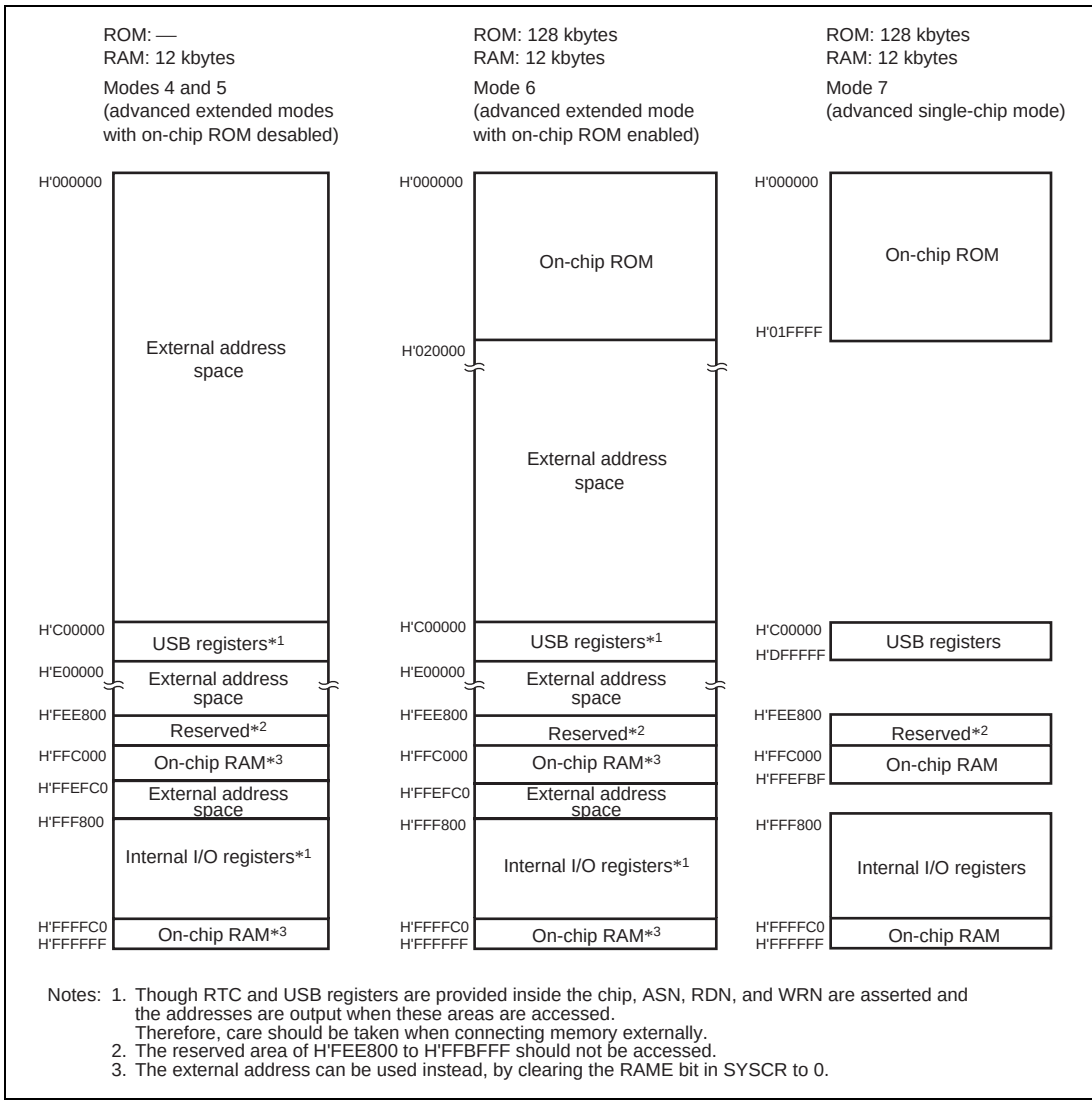
D: Data bus I/O

C: Control signals, clock I/O

*: After reset

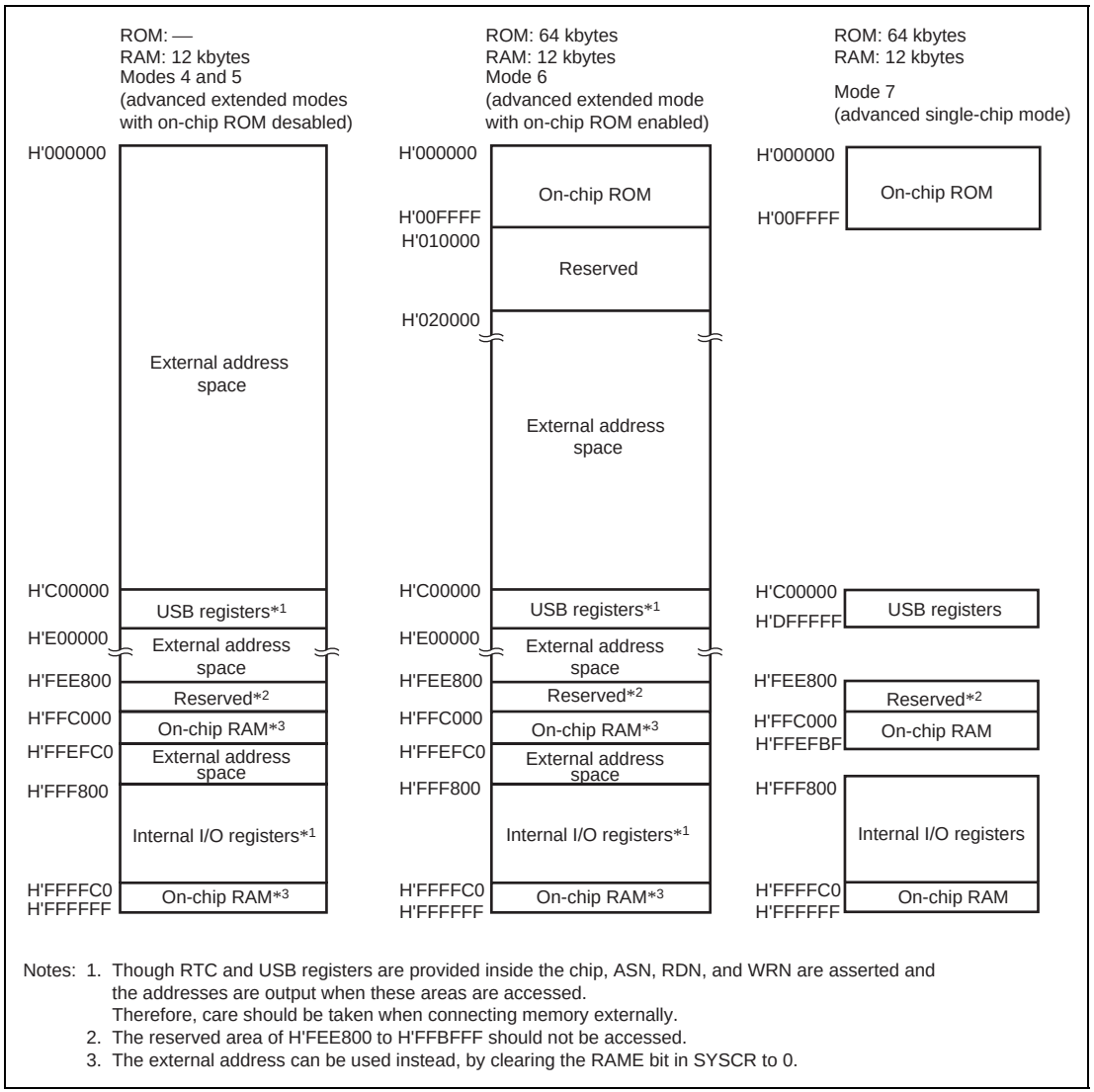
## 3.4 Memory Map in Each Operating Mode

Figures 3.1 to 3.4 show the memory map in each operation mode, respectively.



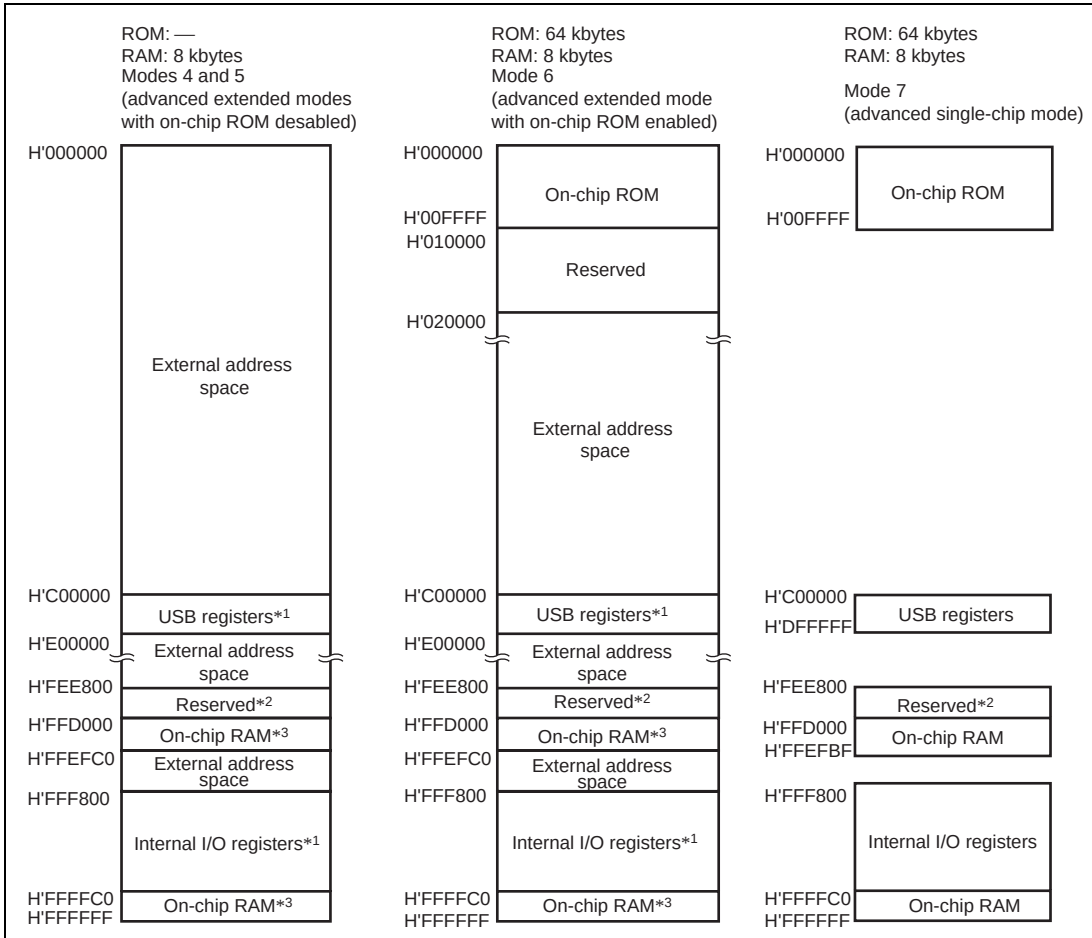

Notes: 1. Though RTC and USB registers are provided inside the chip, ASN, RDN, and WRN are asserted and the addresses are output when these areas are accessed.
Therefore, care should be taken when connecting memory externally.
2. The reserved area of H'FEE800 to H'FFBFFF should not be accessed.
3. The external address can be used instead, by clearing the RAME bit in SYSCR to 0.

**Figure 3.1 Memory Map in Each Operating Mode for HD64F2218, HD64F2218U and HD64F2218CU**




Notes: 1. Though RTC and USB registers are provided inside the chip, ASN, RDN, and WRN are asserted and the addresses are output when these areas are accessed.
Therefore, care should be taken when connecting memory externally.
2. The reserved area of H'FEE800 to H'FFBFFF should not be accessed.
3. The external address can be used instead, by clearing the RAME bit in SYSCR to 0.

**Figure 3.2 Memory Map in Each Operating Mode for HD64F2217CU**




Notes: 1. Though RTC and USB registers are provided inside the chip, ASN, RDN, and WRN are asserted and the addresses are output when these areas are accessed. Therefore, care should be taken when connecting memory externally.
2. The reserved area of H'FEE800 to H'FFCFFF should not be accessed.
3. The external address can be used instead, by clearing the RAME bit in SYSCR to 0.

**Figure 3.3 Memory Map in Each Operating Mode for HD6432217**

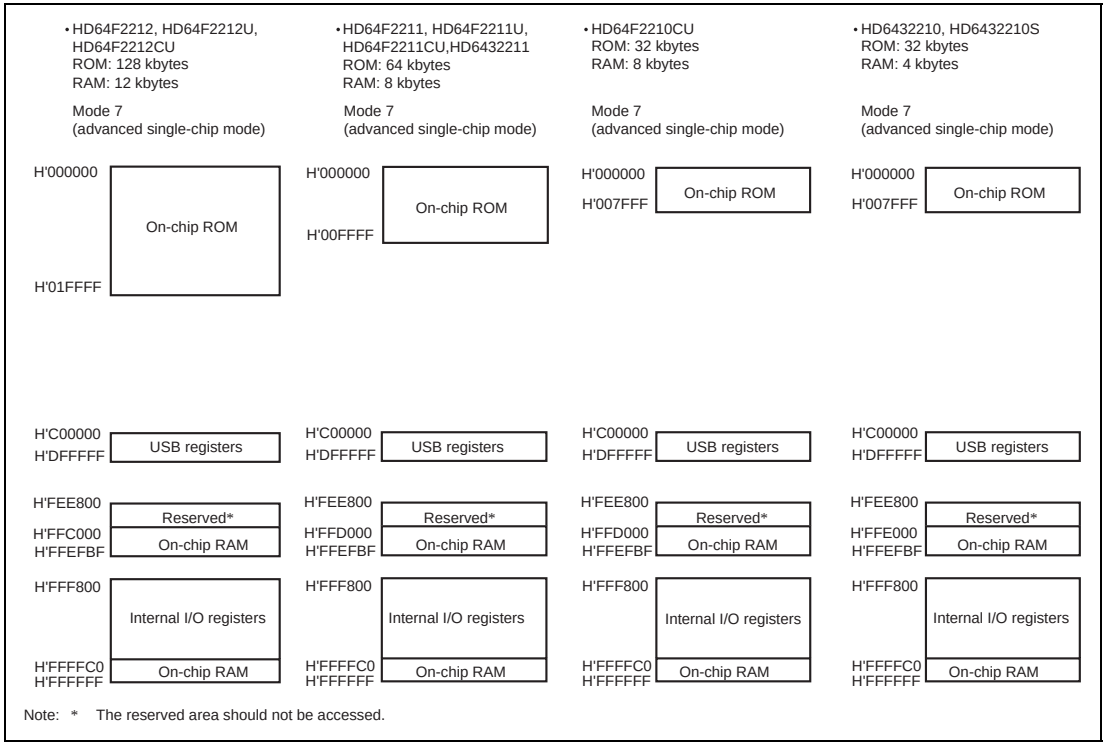



**Figure 3.4 Memory Map in Each Operating Mode for HD64F2212, HD64F2212U, HD64F2212CU, HD64F2211, HD64F2211U, HD64F2211CU, HD64F2210CU, HD6432211, HD6432210 and HD6432210S**

# Section 4 Exception Handling

## 4.1 Exception Handling Types and Priority

As table 4.1 indicates, exception handling may be caused by a reset, trace, trap instruction, or interrupt. Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Exception sources, the stack structure, and operation of the CPU vary depending on the interrupt control mode. For details on the interrupt control mode, refer to section 5, Interrupt Controller.

**Table 4.1 Exception Types and Priority**

| Priority | Exception Type | Start of Exception Handling |
|---|---|---|
| High | Reset | Starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin, or when the watchdog timer overflows. The CPU enters the reset state when the $\overline{\text{RES}}$ pin is low. The CPU enters the manual reset state when the $\overline{\text{MRES}}$ pin* is low. |
| | Trace | Starts when execution of the current instruction or exception handling ends, if the trace (T) bit in the EXR is set to 1. This is enabled only in trace interrupt control mode 2. Trace exception processing is not performed after RTE instruction execution. |
| | Interrupt | Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued. Note that after executing the ANDC, ORC, XORC, or LDC instruction or at the completion of reset exception processing, no interrupt is detected. |
| Low | Trap instruction (TRAPA) | Started by execution of a trap instruction (TRAPA). Trap exception processing is always accepted in program execution state. |

Note: * Supported only by the H8S/2218 Group.

## 4.2 Exception Sources and Exception Vector Table

Different vector addresses are assigned to different exception sources. Table 4.2 lists the exception sources and their vector addresses. Since the usable modes differ depending on the product, for details on each product, refer to section 3, MCU Operating Modes.

**Table 4.2 Exception Handling Vector Table**

| Exception Source | | Vector Number | Vector Address*1 | |
|---|---|---|---|---|
| | | | Normal Mode*4 | Advanced Mode |
| Power-on reset | | 0 | H'0000 to H'0001 | H'0000 to H'0003 |
| Manual reset | | 1 | H'0002 to H'0003 | H'0004 to H'0007 |
| Reserved for system use | | 2 | H'0004 to H'0005 | H'0008 to H'000B |
| | | 3 | H'0006 to H'0007 | H'000C to H'000F |
| | | 4 | H'0008 to H'0019 | H'0010 to H'0013 |
| Trace | | 5 | H'000A to H'000B | H'0014 to H'0017 |
| Direct transitions*2 | | 6 | H'000C to H'000D | H'0018 to H'001B |
| External interrupt (NMI) | | 7 | H'000E to H'000F | H'001C to H'001F |
| Trap instruction | #0 | 8 | H'0010 to H'0011 | H'0020 to H'0023 |
| | #1 | 9 | H'0012 to H'0013 | H'0024 to H'0027 |
| | #2 | 10 | H'0014 to H'0015 | H'0028 to H'002B |
| | #3 | 11 | H'0016 to H'0017 | H'002C to H'002F |
| Reserved for system use | | 12 | H'0018 to H'0019 | H'0030 to H'0033 |
| | | 13 | H'001A to H'001B | H'0034 to H'0037 |
| | | 14 | H'001C to H'001D | H'0038 to H'003B |
| | | 15 | H'001E to H'001F | H'003C to H'003F |
| External interrupt | IRQ0 | 16 | H'0020 to H'0021 | H'0040 to H'0043 |
| External interrupt | IRQ1 | 17 | H'0022 to H'0023 | H'0044 to H'0047 |
| External interrupt | IRQ2 | 18 | H'0024 to H'0025 | H'0048 to H'004B |
| External interrupt | IRQ3 | 19 | H'0026 to H'0027 | H'004C to H'004F |
| External interrupt | IRQ4 | 20 | H'0028 to H'0029 | H'0050 to H'0053 |
| RTC interrupt | IRQ5 | 21 | H'002A to H'002B | H'0054 to H'0057 |
| USB interrupt | IRQ6 | 22 | H'002C to H'002D | H'0058 to H'005B |
| External interrupt | IRQ7 | 23 | H'002E to H'002F | H'005C to H'005F |
| Internal interrupt*3 | | 24 | H'0030 to H'0031 | H'0060 to H'0063 |
| | | ⎪ | ⎪ | ⎪ |
| | | 127 | H'00FE to H'00FF | H'01FC to H'01FF |

Notes: 1. Lower 16 bits of the address.
2. For direct transfer, see section 20.10, Direct Transitions.
3. For details of internal interrupt vectors, see section 5.5, Interrupt Exception Handling Vector Table.
4. Not available in this LSI.

## 4.3 Reset

A reset has the highest exception priority.

When the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin goes low, all processing halts and this LSI enters the reset state. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-up or hold the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin low for at least 20 states during operation.

A reset initializes the internal state of the CPU and the registers of on-chip peripheral modules.

This LSI can also be reset by overflow of the watchdog timer. For details, see section 10, Watchdog Timer (WDT).

Immediately after a reset, interrupt control mode 0 is set.

Notes: $\overline{\text{TRST}}$ in the HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU, which incorporate a boundary scan function, should be brought low when power is on. For details, see section 13, Boundary Scan Function.

* Supported only by the H8S/2218 Group.

### 4.3.1 Reset Types

A reset can be of either of two types for the H8S/2218 Group: a power-on reset or a manual reset. A reset for the H8S/2212 Group is power-on reset. Reset types are shown in table 4.3. A power-on reset should be used when powering on.

The internal state of the CPU is initialized by either type of reset. A power-on reset also initializes all the registers in the on-chip peripheral modules, while a manual reset initializes all the registers in the on-chip peripheral modules except for the bus controller and I/O ports, which retain their previous states.

With a manual reset, since the on-chip peripheral modules are initialized, ports used as on-chip peripheral module I/O pins are switched to I/O ports controlled by DDR and DR.

**Table 4.3 Reset Types**

| | Reset Transition Condition | | Internal State | |
|---|---|---|---|---|
| **Type** | **$\overline{\text{MRES}}$** | **$\overline{\text{RES}}$** | **CPU** | **On-Chip Peripheral Modules** |
| Power-on reset | × | Low | Initialized | Initialized |
| Manual reset | Low | High | Initialized | Initialized, except for bus controller and I/O ports |

Legend:
×: Don't care

A reset caused by the watchdog timer can also be of either of two types: a power-on reset or a manual reset.

When the $\overline{\text{MRES}}$ pin* is used, $\overline{\text{MRES}}$ pin* input must be enabled by setting the MRESE bit to 1 in SYSCR.

Note:* Supported only by the H8S/2218 Group.

### 4.3.2 Reset Exception Handling

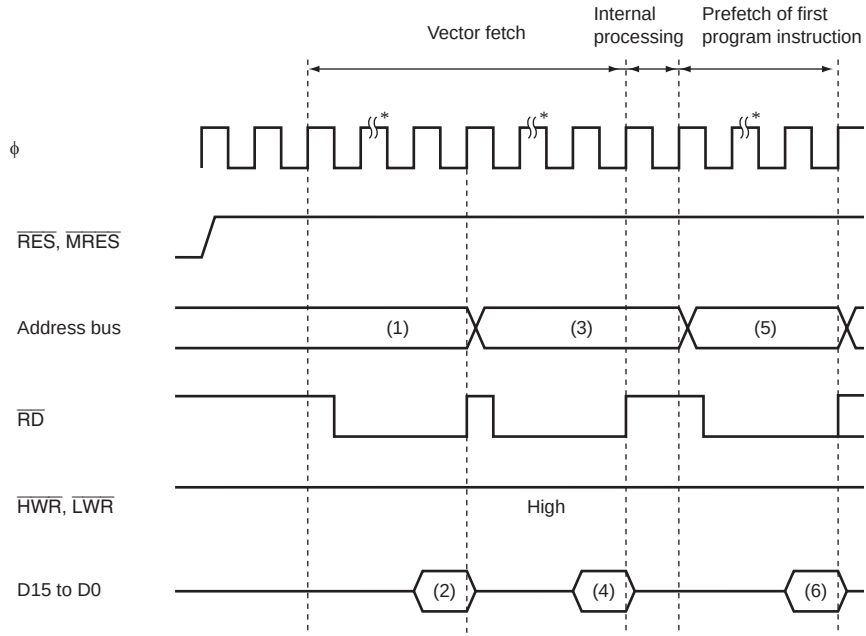
When the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin goes low, this LSI enters the reset. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-up. To reset the chip during operation, hold the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin low for at least 20 states.

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

1. The internal state of the CPU and the registers of the on-chip peripheral modules are initialized, the T bit is cleared to 0 in EXR, and the I bit is set to 1 in EXR and CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Note: * Supported only by the H8S/2218 Group.

Figures 4.1 and 4.2 show examples of the reset sequence.




(1) (3) Reset exception handling vector address (for power-on reset, (1) = H'000000, (3) = H'000002; for manual reset, (1) = H'000004, (3) = H'000006)
(2) (4) Start address (contents of reset exception handling vector address)
(5) Start address ((5) = (2) (4))
(6) First program instruction

Note: * Three program wait states are inserted.

**Figure 4.1 Reset Sequence (Mode 4)**

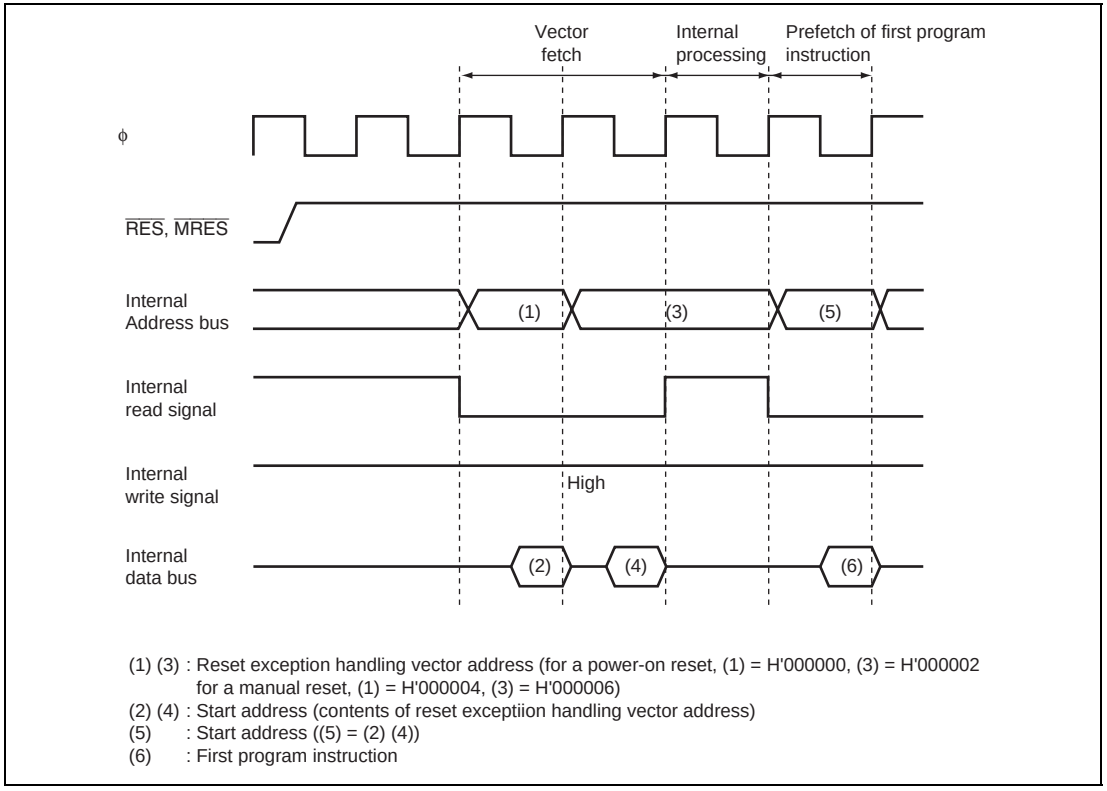



**Figure 4.2 Reset Sequence (Modes 6 and 7)**

### 4.3.3 Interrupts after Reset

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

### 4.3.4 State of On-Chip Peripheral Modules after Reset Release

After reset release, MSTPCRA, MSTPCRB, and MSTPCRC are initialized and all modules except the DMAC enter module stop mode. Consequently, on-chip peripheral module registers cannot be read from or written to. Register reading and writing is enabled when module stop mode is exited.

## 4.4 Traces

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

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

Trace exception handling is not carried out after execution of the RTE instruction.

Interrupts are accepted even within the trace exception handling routine.

**Table 4.4 Status of CCR and EXR after Trace Exception Handling**

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

Legend:
1: Set to 1
0: Cleared to 0
–: Retains value prior to execution.

## 4.5 Interrupts

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

The interrupt exception handling is as follows:

1. The values in the program counter (PC), condition code register (CCR), and extended control register (EXR) are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared.
3. A vector address corresponding to the interrupt source is generated, the start address is loaded from the vector table to the PC, and program execution starts from that address.

## 4.6 Trap Instruction

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

The trap instruction exception handling is as follows:

1. The values in the program counter (PC), condition code register (CCR), and extended control register (EXR) are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared.
3. A vector address corresponding to the interrupt source is generated, the start address is loaded from the vector table to the PC, and program execution starts from that address.

The TRAPA instruction fetches a start address from a vector table entry corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 4.5 shows the status of CCR and EXR after execution of trap instruction exception handling.

**Table 4.5 Status of CCR and EXR after Trap Instruction Exception Handling**

| | CCR | | EXR | |
|---|---|---|---|---|
| **Interrupt Control Mode** | **I** | **UI** | **I2 to I0** | **T** |
| 0 | 1 | – | – | – |
| 2 | 1 | – | – | 0 |

Legend:
1: Set to 1
0: Cleared to 0
–: Retains value prior to execution.

## 4.7 Stack Status after Exception Handling

Figure 4.3 shows the stack after completion of trap instruction exception handling and interrupt exception handling.

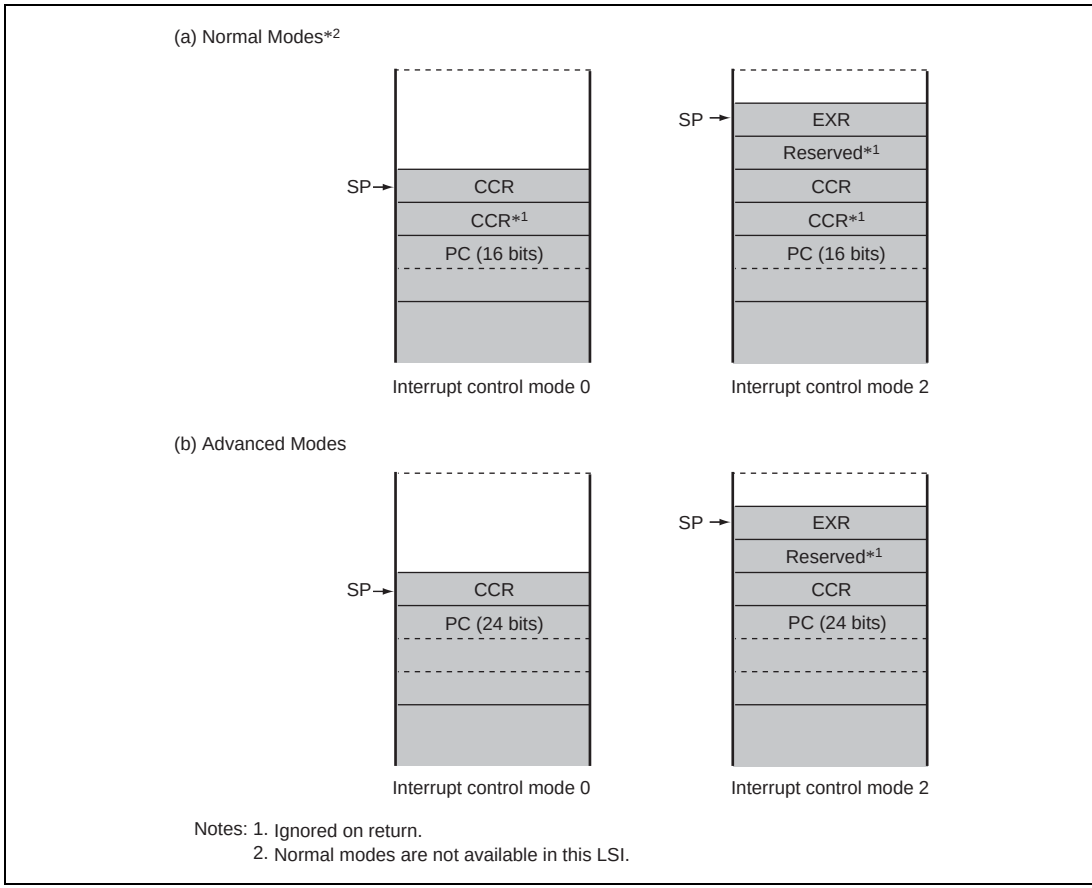



**Figure 4.3 Stack Status after Exception Handling**

## 4.8 Notes on Use of the Stack

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

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

Use the following instructions to restore registers:

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

Setting SP to an odd value may lead to a malfunction. Figure 4.4 shows an example of what happens when the SP value is odd.

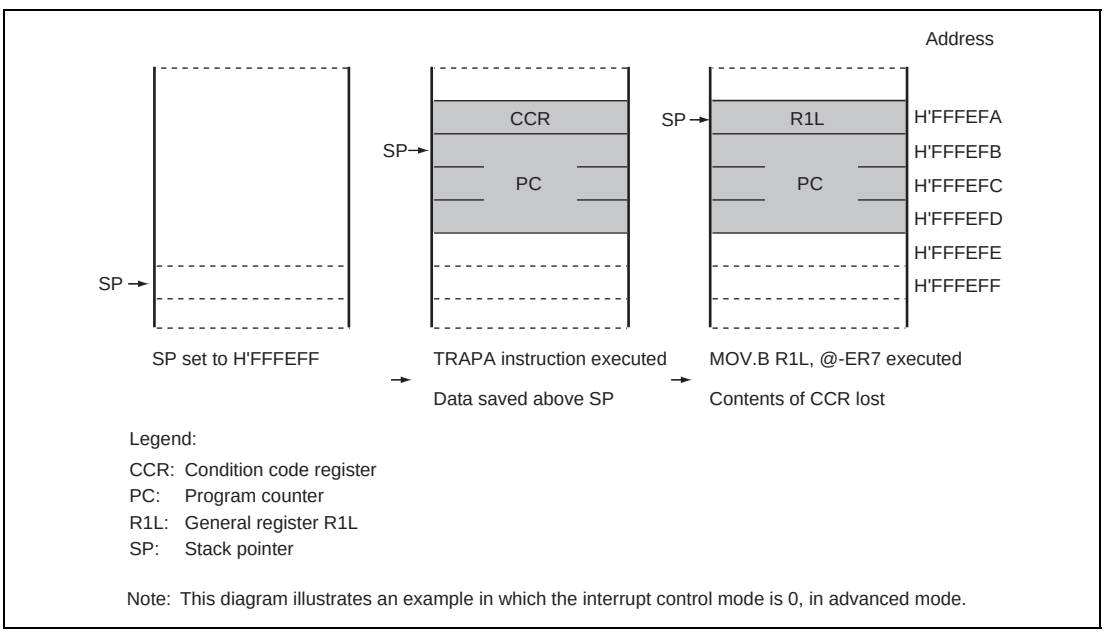



**Figure 4.4 Operation when SP Value Is Odd**

# Section 5 Interrupt Controller

## 5.1 Features

- Two interrupt control modes
  - Any of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).
- Priorities settable with IPR
  - An interrupt priority register (IPR) is provided for setting interrupt priorities. Eight priority levels can be set for each module for all interrupts except NMI. NMI is assigned the highest priority level of 8, and can be accepted at all times.
- Independent vector addresses
  - All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.
- Seven external interrupts (NMI, $\overline{\text{IRQ7}}$, and $\overline{\text{IRQ4}}$ to $\overline{\text{IRQ0}}$)
  - NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge can be selected for NMI. Falling edge, rising edge, or both edge detection, or level sensing, can be selected for $\overline{\text{IRQ7}}$ and $\overline{\text{IRQ4}}$ to $\overline{\text{IRQ0}}$. $\overline{\text{IRQ6}}$ is an interrupt only for the on-chip USB. $\overline{\text{IRQ5}}$ is an interrupt only for the on-chip RTC.
- DMAC control
  - DMAC activation is performed by means of interrupts.

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

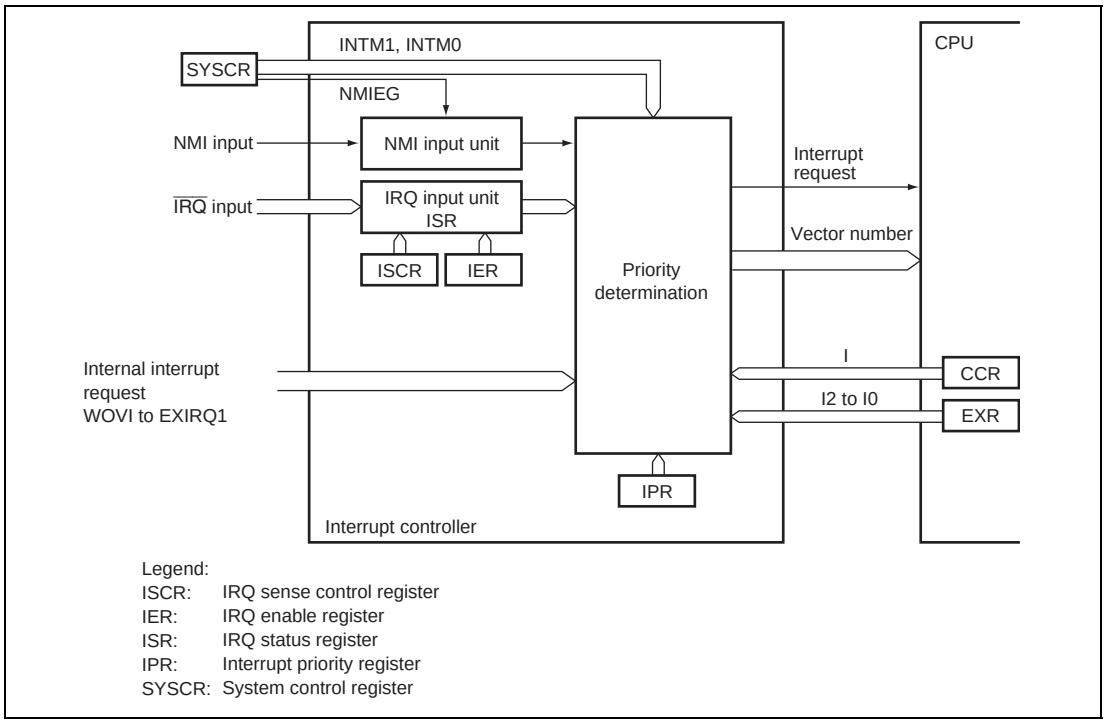



**Figure 5.1 Block Diagram of Interrupt Controller**

## 5.2 Input/Output Pins

Table 5.1 summarizes the pins of the interrupt controller.

**Table 5.1 Pin Configuration**

| Name | I/O | Function |
|---|---|---|
| NMI | Input | Nonmaskable external interrupt<br>Rising or falling edge can be selected |
| $\overline{\text{IRQ7}}$ | Input | Maskable external interrupts |
| $\overline{\text{IRQ4}}$ | Input | Rising, falling, or both edges, or level sensing can be selected ($\overline{\text{IRQ6}}$ is an interrupt signal only for the on-chip USB. $\overline{\text{IRQ5}}$ is an interrupt signal only for the on-chip RTC.) |
| $\overline{\text{IRQ3}}$ | Input | |
| $\overline{\text{IRQ2}}$ | Input | |
| $\overline{\text{IRQ1}}$ | Input | |
| $\overline{\text{IRQ0}}$ | Input | |

## 5.3 Register Descriptions

The interrupt controller has the following registers. For details on the system control register, refer to section 3.2.2, System Control Register (SYSCR).

- System control register (SYSCR)
- IRQ sense control register H (ISCRH)
- IRQ sense control register L (ISCRL)
- IRQ enable register (IER)
- IRQ status register (ISR)
- Interrupt priority register A (IPRA)
- Interrupt priority register B (IPRB)
- Interrupt priority register C (IPRC)
- Interrupt priority register D (IPRD)
- Interrupt priority register E (IPRE)
- Interrupt priority register F (IPRF)
- Interrupt priority register G (IPRG)
- Interrupt priority register J (IPRJ)
- Interrupt priority register K (IPRK)
- Interrupt priority register M (IPRM)

### 5.3.1 Interrupt Priority Registers A to G, J, K, M (IPRA to IPRG, IPRJ, IPRK, IPRM)

The IPR registers set priorities (levels 7 to 0) for interrupts other than NMI.

The correspondence between interrupt sources and IPR settings is shown in section 5.5, Interrupt Exception Handling Vector Table. Setting a value in the range from H'0 to H'7 in the 3-bit groups of bits 6 to 4 and 2 to 0 sets the priority of the corresponding interrupt.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | – | 0 | – | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 6 | IPR6 | 1 | R/W | These bits set the priority of the corresponding interrupt source.<br>000: Priority level 0 (Lowest)<br>001: Priority level 1<br>010: Priority level 2<br>011: Priority level 3<br>100: Priority level 4<br>101: Priority level 5<br>110: Priority level 6<br>111: Priority level 7 (Highest) |
| 5 | IPR5 | 1 | R/W | |
| 4 | IPR4 | 1 | R/W | |
| 3 | – | 0 | – | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 2 | IPR2 | 1 | R/W | These bits set the priority of the corresponding interrupt source.<br>000: Priority level 0 (Lowest)<br>001: Priority level 1<br>010: Priority level 2<br>011: Priority level 3<br>100: Priority level 4<br>101: Priority level 5<br>110: Priority level 6<br>111: Priority level 7 (Highest) |
| 1 | IPR1 | 1 | R/W | |
| 0 | IPR0 | 1 | R/W | |

### 5.3.2 IRQ Enable Register (IER)

IER controls enabling and disabling of interrupt requests IRQ7 to IRQ0.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | IRQ7E | 0 | R/W | IRQ7 Enable<br>The IRQ7 interrupt request is enabled when this bit is 1. |
| 6 | IRQ6E | 0 | R/W | IRQ6 Enable*[1]<br>The IRQ6 interrupt request is enabled when this bit is 1. |
| 5 | IRQ5E | 0 | R/W | IRQ5 Enable*[2]<br>The IRQ5 interrupt request is enabled when this bit is 1. |
| 4 | IRQ4E | 0 | R/W | IRQ4 Enable<br>The IRQ4 interrupt request is enabled when this bit is 1. |
| 3 | IRQ3E | 0 | R/W | IRQ3 Enable<br>The IRQ3 interrupt request is enabled when this bit is 1. |
| 2 | IRQ2E | 0 | R/W | IRQ2 Enable<br>The IRQ2 interrupt request is enabled when this bit is 1. |
| 1 | IRQ1E | 0 | R/W | IRQ1 Enable<br>The IRQ1 interrupt request is enabled when this bit is 1. |
| 0 | IRQ0E | 0 | R/W | IRQ0 Enable<br>The IRQ0 interrupt request is enabled when this bit is 1. |

Notes: 1. IRQ6 is an interrupt only for the on-chip USB.
2. IRQ5 is an interrupt only for the on-chip RTC.

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

The ISCR registers select the source that generates an interrupt request at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 15 | IRQ7SCB | 0 | R/W | IRQ7 Sense Control B |
| 14 | IRQ7SCA | 0 | R/W | IRQ7 Sense Control A<br>00: Interrupt request generated at $\overline{\text{IRQ7}}$ input low level<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ7}}$ input<br>10: Interrupt request generated rising edge of $\overline{\text{IRQ7}}$ input<br>11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ7}}$ input |
| 13 | IRQ6SCB | 0 | R/W | IRQ6*[1] Sense Control B |
| 12 | IRQ6SCA | 0 | R/W | IRQ6*[1] Sense Control A<br>00: Setting prohibited when using on-chip USB suspend or resume interrupt<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ6}}$ input<br>1x: Setting prohibited |
| 11 | IRQ5SCB | 0 | R/W | IRQ5*[2] Sense Control B |
| 10 | IRQ5SCA | 0 | R/W | IRQ5*[2] Sense Control A<br>00: Setting prohibited when using RTC interrupt<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ5}}$ input<br>1x: Setting prohibited |
| 9 | IRQ4SCB | 0 | R/W | IRQ4 Sense Control B |
| 8 | IRQ4SCA | 0 | R/W | IRQ4 Sense Control A<br>00: Interrupt request generated at $\overline{\text{IRQ4}}$ input low level<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ4}}$ input<br>10: Interrupt request generated at rising edge of $\overline{\text{IRQ4}}$ input<br>11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ4}}$ input |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | IRQ3SCB | 0 | R/W | IRQ3 Sense Control B |
| 6 | IRQ3SCA | 0 | R/W | IRQ3 Sense Control A<br>00: Interrupt request generated at $\overline{\text{IRQ3}}$ input low level<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ3}}$ input<br>10: Interrupt request generated at rising edge of $\overline{\text{IRQ3}}$ input<br>11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ3}}$ input |
| 5 | IRQ2SCB | 0 | R/W | IRQ2 Sense Control B |
| 4 | IRQ2SCA | 0 | R/W | IRQ2 Sense Control A<br>00: Interrupt request generated at $\overline{\text{IRQ2}}$ input low level<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ2}}$ input<br>10: Interrupt request generated at rising edge of $\overline{\text{IRQ2}}$ input<br>11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ2}}$ input |
| 3 | IRQ1SCB | 0 | R/W | IRQ1 Sense Control B |
| 2 | IRQ1SCA | 0 | R/W | IRQ1 Sense Control A<br>00: Interrupt request generated at $\overline{\text{IRQ1}}$ input low level<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ1}}$ input<br>10: Interrupt request generated at rising edge of $\overline{\text{IRQ1}}$ input<br>11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ1}}$ input |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 1 | IRQ0SCB | 0 | R/W | IRQ0 Sense Control B |
| 0 | IRQ0SCA | 0 | R/W | IRQ0 Sense Control A<br>00: Interrupt request generated at $\overline{\text{IRQ0}}$ input low level<br>01: Interrupt request generated at falling edge of $\overline{\text{IRQ0}}$ input<br>10: Interrupt request generated at rising edge of $\overline{\text{IRQ0}}$ input<br>11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ0}}$ input |

Legend:
×: Don't care
Notes: 1. IRQ6 is an interrupt only for the on-chip USB.
2. IRQ5 is an interrupt only for the on-chip RTC.

### 5.3.4 IRQ Status Register (ISR)

ISR indicates the status of IRQ7 to IRQ0 interrupt requests.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | IRQ7F | 0 | R/(W)* | [Setting condition]<br>When the interrupt source selected by the ISCR registers occurs<br>[Clearing conditions]<br>• Cleared by reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag<br>• When interrupt exception handling is executed when low-level detection is set and , $\overline{\text{IRQn}}$ input is high<br>• When IRQn interrupt exception handling is executed when falling, rising, or both-edge detection is set |
| 6 | IRQ6F | 0 | R/(W)* | |
| 5 | IRQ5F | 0 | R/(W)* | |
| 4 | IRQ4F | 0 | R/(W)* | |
| 3 | IRQ3F | 0 | R/(W)* | |
| 2 | IRQ2F | 0 | R/(W)* | |
| 1 | IRQ1F | 0 | R/(W)* | |
| 0 | IRQ0F | 0 | R/(W)* | |

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

## 5.4 Interrupt Sources

### 5.4.1 External Interrupts

There are seven external interrupts: NMI, IRQ7, and IRQ4 to IRQ0. These interrupts can be used to restore this LSI from software standby mode. Though IRQ5 is only for the on-chip RTC and IRQ6 is only for the on-chip USB, the interrupts can be used to restore this LSI from software standby mode. IRQ5 and IRQ6 are functionally same as IRQ7 and IRQ4 to IRQ0.

**NMI Interrupt:** NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode or the status of the CPU interrupt mask bits. The NMIEG bit in SYSCR can be used to select whether an interrupt is requested at a rising edge or a falling edge on the NMI pin.

**IRQ7 to IRQ0 Interrupts:** Interrupts IRQ7 to IRQ0 are requested by an input signal at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$. Interrupts IRQ7 to IRQ0 have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$
- Enabling or disabling of interrupt requests IRQ7 to IRQ0 can be selected with IER.
- The interrupt priority level can be set with IPR.
- The status of interrupt requests IRQ7 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

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

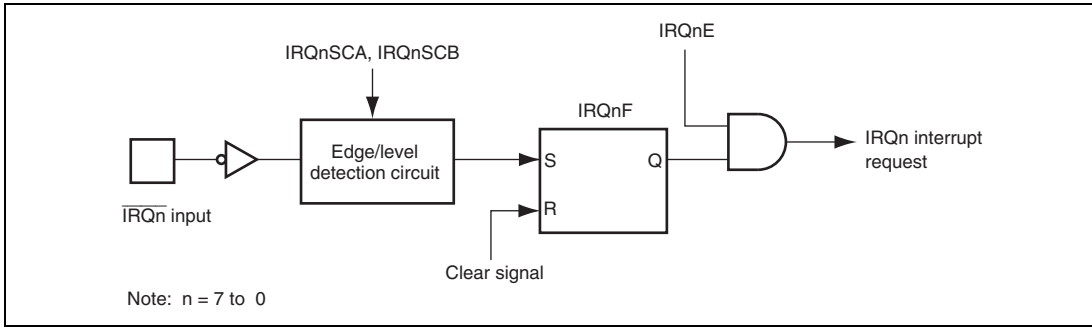



**Figure 5.2 Block Diagram of Interrupts IRQn**

The set timing for IRQnF is shown in figure 5.3.

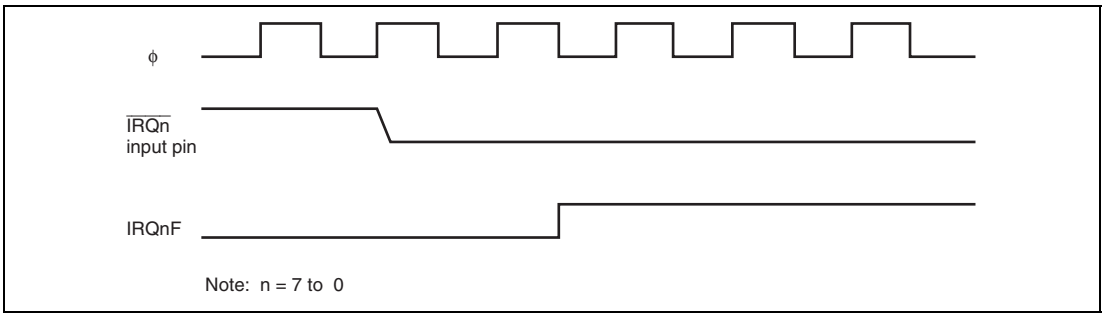



**Figure 5.3 Timing of Setting IRQnF**

The detection of IRQn interrupts does not depend on whether the relevant pin has been set for input or output. However, when a pin is used as an external interrupt input pin, do not clear the corresponding DDR to 0; and use the pin as an I/O pin for another function. IRQnF interrupt request flag is set when the setting condition is satisfied, regardless of IER settings. Accordingly, refer to only necessary flags.

### 5.4.2 Internal Interrupts

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

- For each on—chip peripheral module there are flags that indicate the interrupt request status, and enable bits that select enabling or disabling of these interrupts. If both of these are set to 1 for a particular interrupt source, an interrupt request is issued to the interrupt controller.
- The interrupt priority level can be set by means of IPR.
- The DMAC can be activated by a TPU, SCI, or other interrupt request.
- When the DMAC is activated by an interrupt request, it is not affected by the interrupt control mode or CPU interrupt mask bit.

## 5.5 Interrupt Exception Handling Vector Table

Table 5.2 shows interrupt exception handling sources, vector addresses, and interrupt priorities.

For default priorities, the lower the vector number, the higher the priority. Priorities among modules can be set by means of the IPR. Modules set at the same priority will conform to their default priorities. Priorities within a module are fixed.

**Table 5.2 Interrupt Sources, Vector Addresses, and Interrupt Priorities**

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address* | | Priority |
|---|---|---|---|---|---|
| | | | Advanced Mode | IPR | |
| External pins | NMI | 7 | H'001C | | High |
| | IRQ0 | 16 | H'0040 | IPRA6 to IPRA4 | |
| | IRQ1 | 17 | H'0044 | IPRA2 to IPRA0 | |
| | IRQ2 | 18 | H'0048 | IPRB6 to IPRB4 | |
| | IRQ3 | 19 | H'004C | | |
| | IRQ4 | 20 | H'0050 | IPRB2 to IPRB0 | |
| RTC | IRQ5 | 21 | H'0054 | | |
| USB | IRQ6 | 22 | H'0058 | IPRC6 to IPRC4 | |
| External pins | IRQ7 | 23 | H'005C | | |
| Watchdog Timer | WOVI | 25 | H'0064 | IPRD6 to IPRD4 | |
| A/D | ADI | 28 | H'0070 | IPRE2 to IPRE0 | |
| TPU channel 0 | TGI0A | 32 | H'0080 | IPRF6 to IPRF4 | |
| | TGI0B | 33 | H'0084 | | |
| | TGI0C | 34 | H'0088 | | |
| | TGI0D | 35 | H'008C | | |
| | TGI0V | 36 | H'0090 | | |
| TPU channel 1 | TGI1A | 40 | H'00A0 | IPRF2 to IPRF0 | |
| | TGI1B | 41 | H'00A4 | | |
| | TGI1V | 42 | H'00A8 | | |
| | TGI1U | 43 | H'00AC | | Low |

| Interrupt Source | Origin of Interrupt Source | Vector Number | Vector Address* Advanced Mode | IPR | Priority |
|---|---|---|---|---|---|
| TPU channel 2 | TGI2A | 44 | H'00B0 | IPRG6 to IPRG4 | High |
| | TGI2B | 45 | H'00B4 | | |
| | TGI2V | 46 | H'00B8 | | |
| | TGI2U | 47 | H'00BC | | |
| DMAC | DEND0A | 72 | H'0120 | IPRJ6 to IPRJ4 | |
| | DEND0B | 73 | H'0124 | | |
| | DEND1A | 74 | H'0128 | | |
| | DEND1B | 75 | H'012C | | |
| SCI channel 0 | ERI0 | 80 | H'0140 | IPRJ2 to IPRJ0 | |
| | RXI0 | 81 | H'0144 | | |
| | TXI0 | 82 | H'0148 | | |
| | TEI0 | 83 | H'014C | | |
| SCI channel 2 | ERI2 | 88 | H'0160 | IPRK2 to IPRK0 | |
| | RXI2 | 89 | H'0164 | | |
| | TXI2 | 90 | H'0168 | | |
| | TEI2 | 91 | H'016C | | |
| USB | EXIRQ0 | 104 | H'01A0 | IPRM6 to IPRM4 | |
| | EXIRQ1 | 105 | H'01A4 | | Low |

Note: * Lower 16 bits of the start address.

## 5.6 Interrupt Control Modes and Interrupt Operation

The interrupt controller has two modes: interrupt control mode 0 and interrupt control mode 2.

Interrupt operations differ depending on the interrupt control mode. The interrupt control mode is selected by SYSCR. Table 5.3 shows the differences between interrupt control mode 0 and interrupt control mode 2.

**Table 5.3 Interrupt Control Modes**

| Interrupt Control Mode | Priority Setting Register | Interrupt Mask Bits | Description |
|---|---|---|---|
| 0 | Default | I | The priority of interrupt sources are fixed at the default settings.<br>Interrupt sources except for NMI is marked by the I bit. |
| 2 | IPR | I2 to I0 | 8-level interrupt mask control is performed by bits I2 to I0.<br>8 priority levels other than NMI can be set with IPR. |

### 5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests except for NMI is masked by the I bit of CCR in the CPU. Figure 5.4 shows a flowchart of the interrupt acceptance operation in this case.

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

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

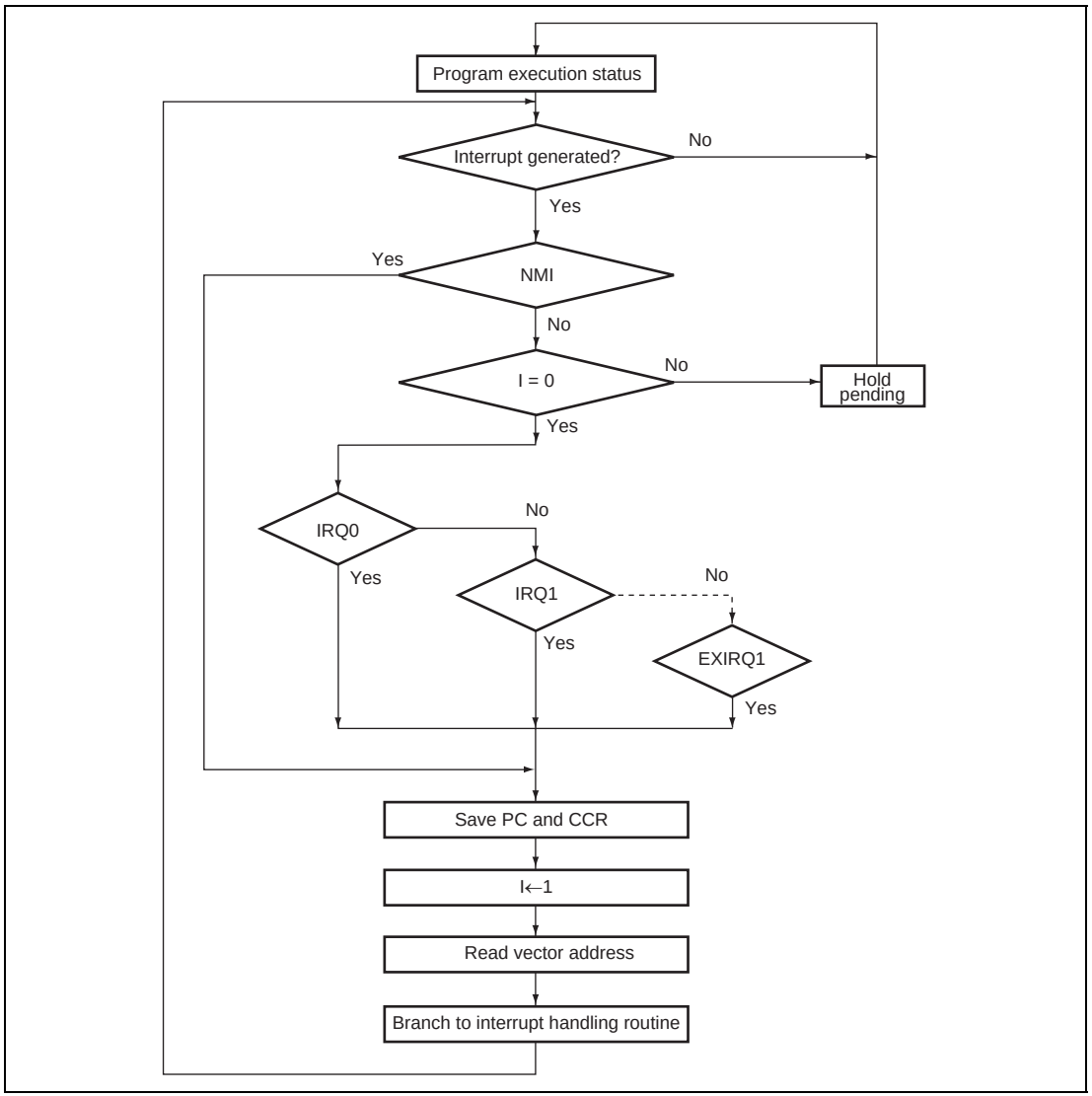



**Figure 5.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

### 5.6.2 Interrupt Control Mode 2

In interrupt control mode 2, mask control is done in eight levels for interrupt requests except for NMI by comparing the EXR interrupt mask level (I2 to I0 bits) in the CPU and the IPR setting.

Figure 5.5 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. When interrupt requests are sent to the interrupt controller, the interrupt with the highest priority according to the interrupt priority levels set in IPR is selected, and lower-priority interrupt requests are held pending. If a number of interrupt requests with the same priority are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 5.2 is selected.
3. Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. An interrupt request with a priority no higher than the mask level set at that time is held pending, and only an interrupt request with a priority higher than the interrupt mask level is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC, CCR, and EXR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
6. The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority level of the accepted interrupt.

   If the accepted interrupt is NMI, the interrupt mask level is set to H'7.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

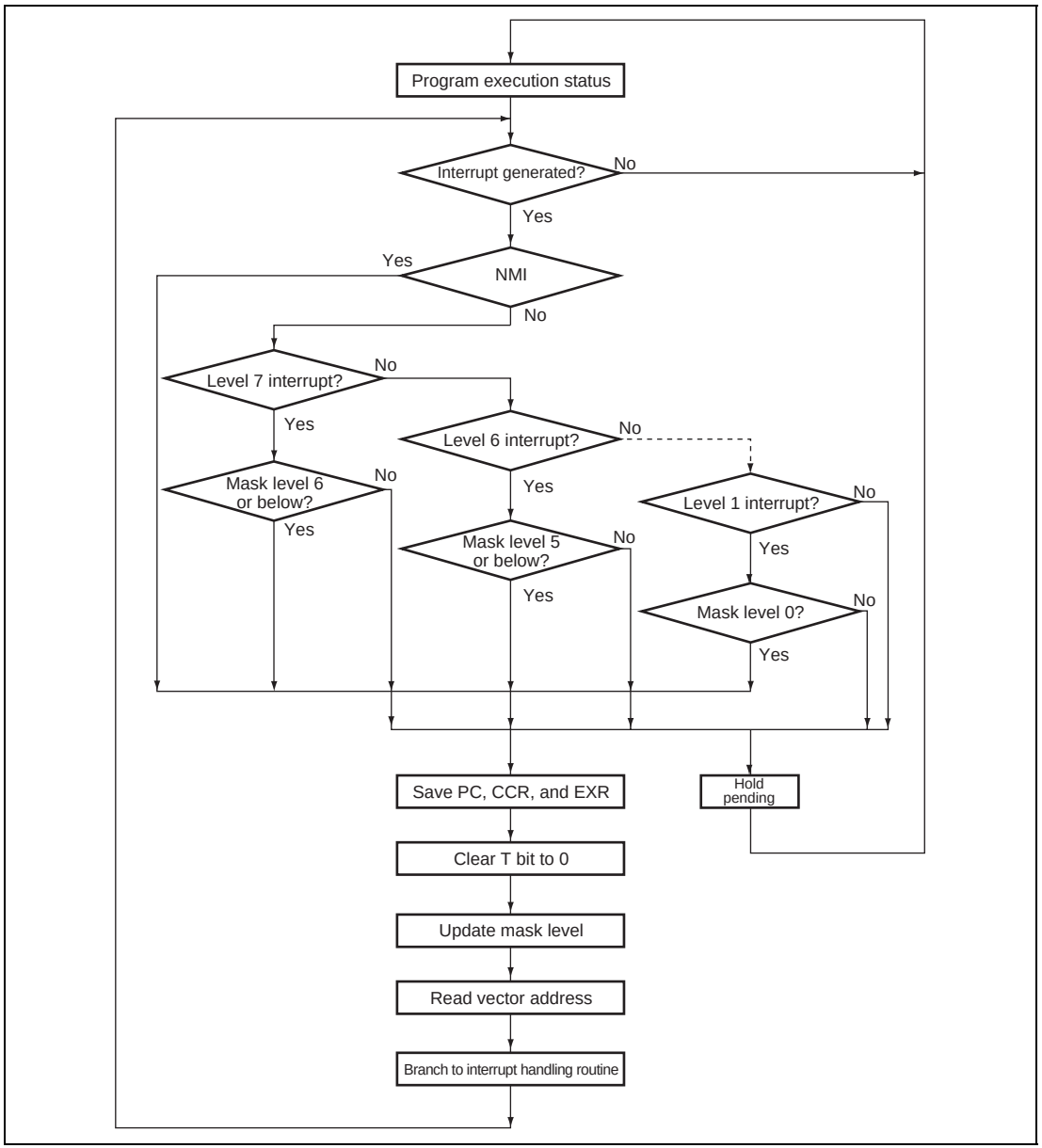



**Figure 5.5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2**

### 5.6.3 Interrupt Exception Handling Sequence

Figure 5.6 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.

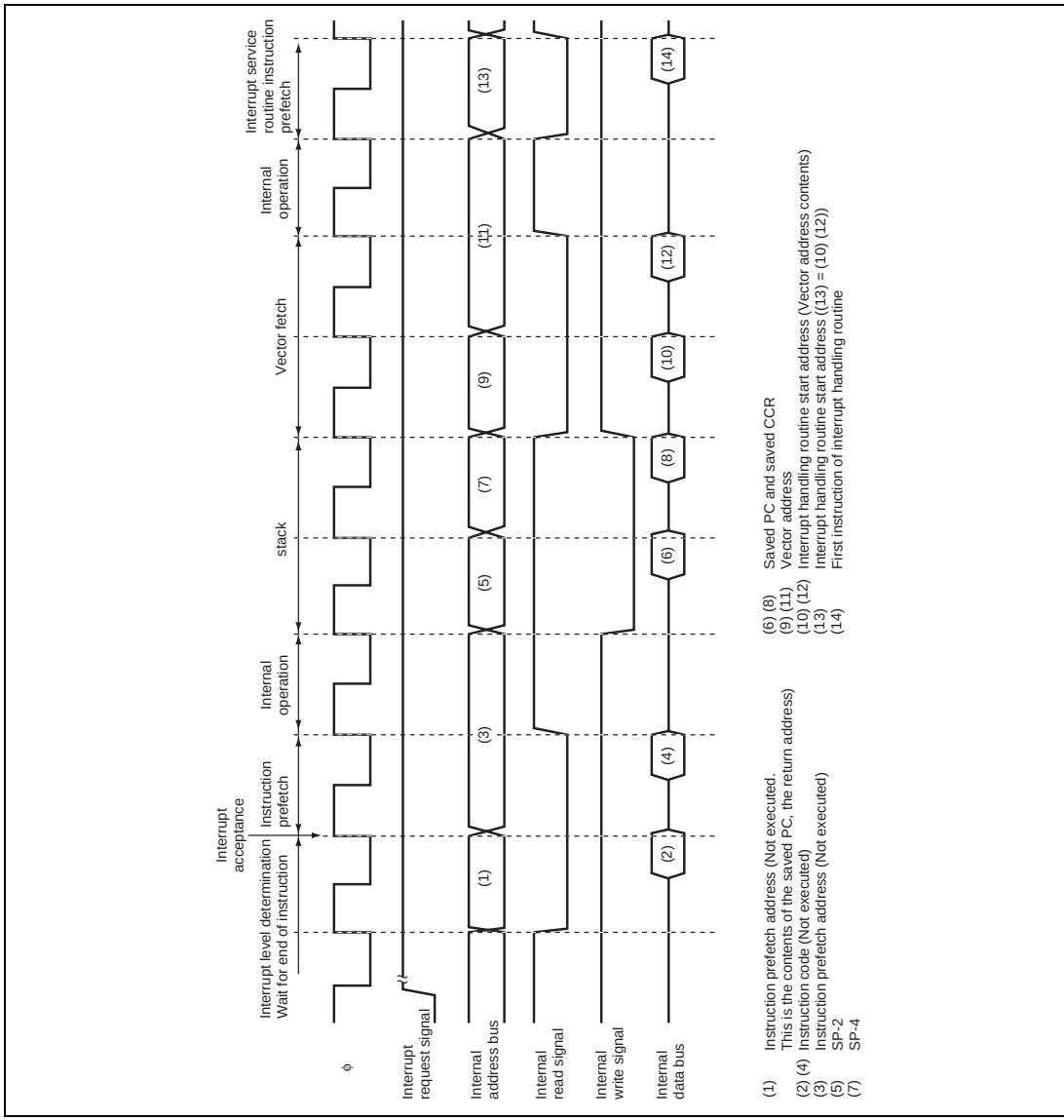



**Figure 5.6 Interrupt Exception Handling**

### 5.6.4 Interrupt Response Times

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

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

**Table 5.4 Interrupt Response Times**

| | | Normal Mode*[5] | | Advanced Mode | |
|---|---|---|---|---|---|
| No. | Execution State | Interrupt Control Mode 0 | Interrupt Control Mode 2 | Interrupt Control Mode 0 | Interrupt Control Mode 2 |
| 1 | Interrupt priority determination*[1] | 3 | 3 | 3 | 3 |
| 2 | Number of wait states until executing instruction ends*[2] | 1 to 19+2·$S_I$ | 1 to 19+2·$S_I$ | 1 to 19+2·$S_I$ | 1 to 19+2·$S_I$ |
| 3 | PC, CCR, EXR stack save | 2·$S_K$ | 3·$S_K$ | 2·$S_K$ | 3·$S_K$ |
| 4 | Vector fetch | $S_I$ | $S_I$ | 2·$S_I$ | 2·$S_I$ |
| 5 | Instruction fetch*[3] | 2·$S_I$ | 2·$S_I$ | 2·$S_I$ | 2·$S_I$ |
| 6 | Internal processing*[4] | 2 | 2 | 2 | 2 |
| Total | (using on-chip memory) | 11 to 31 | 12 to 32 | 12 to 32 | 13 to 33 |

Notes: 1. Two states in case of internal interrupt.
2. Refers to MULXS and DIVXS instructions.
3. Prefetch after interrupt acceptance and interrupt handling routine prefetch.
4. Internal processing after interrupt acceptance and internal processing after vector fetch.
5. Not available in this LSI.

**Table 5.5 Number of States in Interrupt Handling Routine Execution Statuses**

| | | Object of Access | | | | |
|---|---|---|---|---|---|---|
| | | | External Device | | | |
| | | | 8-Bit Bus | | 16-Bit Bus | |
| Symbol | | Internal Memory | 2-State Access | 3-State Access | 2-State Access | 3-State Access |
| Instruction fetch | $S_I$ | 1 | 4 | 6 + 2m | 2 | 3 + m |
| Branch address read | $S_J$ | | | | | |
| Stack manipulation | $S_K$ | | | | | |

Legend:
m: Number of wait states in an external device access.

### 5.6.5 DMAC Activation by Interrupt

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

- Interrupt request to CPU
- Activation request to DMAC
- Selection of a number of the above

For details of interrupt requests that can be used with to activate the DMAC, see section 7, DMA Controller (DMAC).

Figure 5.7 shows a block diagram of the interrupt controller of DMAC.

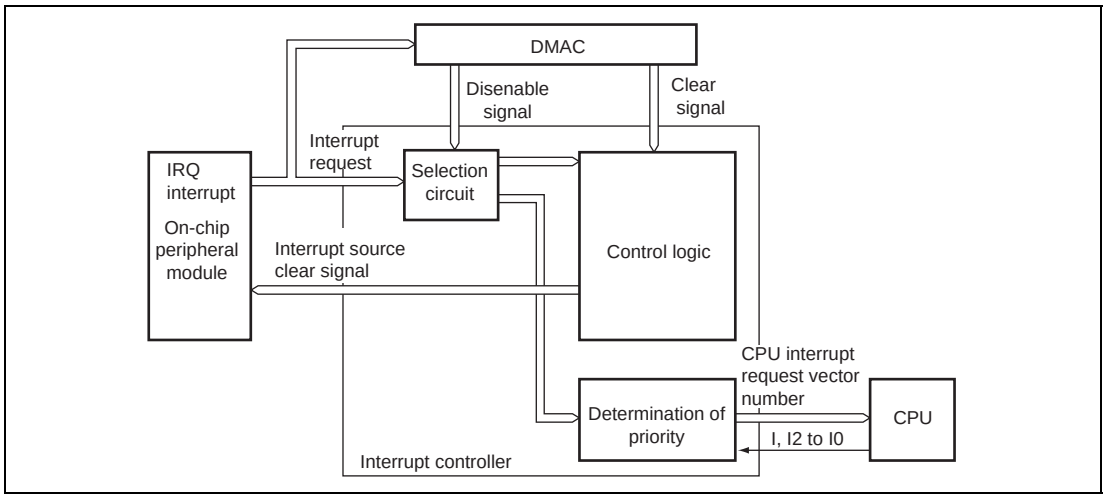



**Figure 5.7 Interrupt Control for DMAC**

**Selection of Interrupt Source:** An activation factor is directly input to each channel of the DMAC. The activation factors for each channel of the DMAC are selected by the DTF3 to DTF0 bits of DMACR. The DTA bit of DMABCR can be used to select whether the selected activation factors are managed by the DMAC. By setting the DTA bit to 1, the interrupt factor which was the activation factor for that DMAC cannot act as the CPU interrupt factor.

Interrupt factors other than the interrupts managed by the DMAC is CPU interrupt request.

**Determination of Priority:** The activation source is directly input to each channel of DMAC.

**Operation Order:** If the same interrupt is selected as the DMAC activation factor or CPU interrupt factor, these operate independently. They operate in accordance with the respective operating states and bus priorities.

Table 5.6 shows the interrupt factor clear control and selection of interrupt factors by specification of the DTA bit of DMAC's DMABCR.

**Table 5.6 Interrupt Source Selection and Clearing Control**

| Settings | | |
|---|---|---|
| **DMAC** | **Interrupt Sources Selection/Clearing Control** | |
| **DTA** | **DMAC** | **CPU** |
| 0 | Δ | O |
| 1 | O | X |

Legend:

O: The relevant interrupt is used. Interrupt source clearing is performed.
(The CPU should clear the source flag in the interrupt handling routine.)

Δ: The relevant interrupt is used. The interrupt source is not cleared.

X: The relevant bit cannot be used.

**Notes on Use:** The SCI interrupt source is cleared when the DMAC reads or writes to the prescribed register, and is not dependent upon the DTA bit.

## 5.7 Usage Notes

### 5.7.1 Contention between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupts, the disabling becomes effective after execution of the instruction.

When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored.

The same also applies when an interrupt source flag is cleared to 0.

Figure 5.8 shows an example in which the TGIEA bit in the TPU's TIER_0 is cleared to 0.

The above contention will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

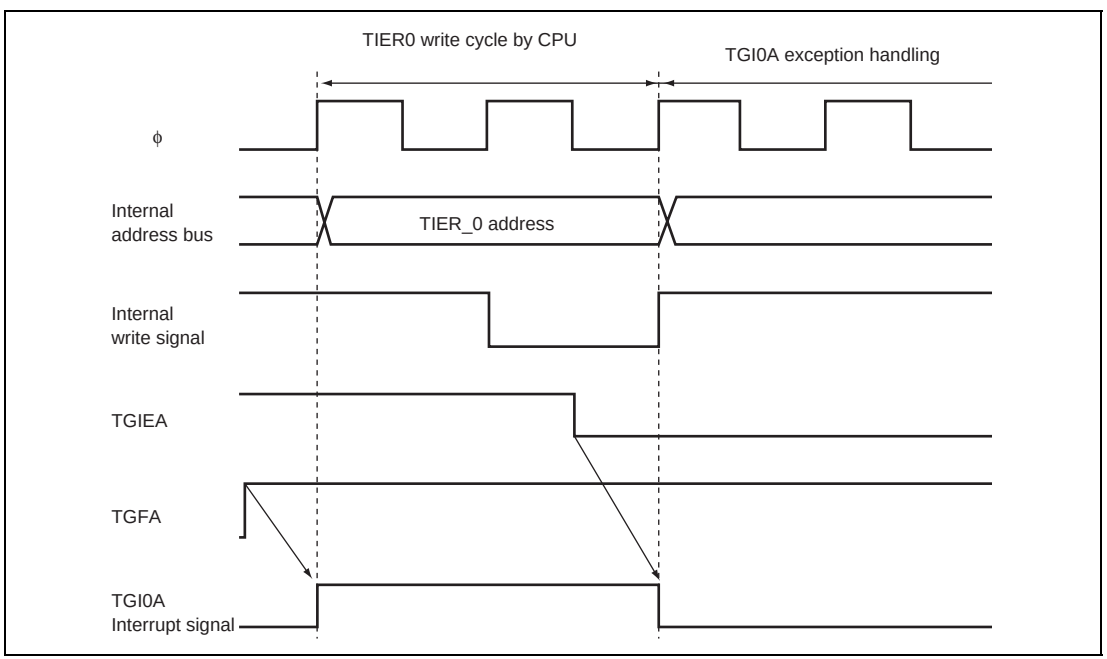



**Figure 5.8 Contention between Interrupt Generation and Disabling**

### 5.7.2 Instructions that Disable Interrupts

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

### 5.7.3 Times when Interrupts Are Disabled

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

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

### 5.7.4 Interrupts during Execution of EEPMOV Instruction

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

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

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at a break in the transfer cycle. The PC value saved on the stack in this case is the address of the next instruction.

Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

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

### 5.7.5 IRQ Interrupt

When operating by clock input, acceptance of input to an IRQ is synchronized with the clock. In software standby mode and watch mode, the input is accepted asynchronously. For details on the input conditions, see section 22.4.2, Control Signal Timing.

### 5.7.6 NMI Interrupt Usage Notes

The NMI interrupt is part of the exception processing performed cooperatively by the LSI's internal interrupt controller and the CPU when the system is operating normally under the specified electrical conditions. No operations, including NMI interrupts, are guaranteed when operation is not normal (runaway status) due to software problems or abnormal input to the LSI's pins. In such cases, the LSI may be restored to the normal program execution state by applying an external reset.

# Section 6 Bus Controller

This LSI has a built-in bus controller (BSC) that manages the external address space divided into eight areas. The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters: the CPU and DMA controller (DMAC).

## 6.1 Features

- Manages external address space in area units
  - Manages the external space as 8 areas of 2-Mbytes
  - Bus specifications can be set independently for each area
  - Burst ROM interface can be set
- Basic bus interface*[1]
  - Chip select ($\overline{CS0}$ to $\overline{CS5}$) can be output for areas 0 to 5*[2]
  - 8-bit access or 16-bit access can be selected for each area
  - 2-state access or 3-state access can be selected for each area
  - Program wait states can be inserted for each area
- Burst ROM interface*[2]
  - Burst ROM interface can be selected for area 0
  - One or two states can be selected for the burst cycle
- Idle cycle insertion*[2]
  - Idle cycle can be inserted between consecutive read accesses to different areas
  - Idle cycle can be inserted before a write access to an external area immediately after a read access to an external area
- Bus arbitration
  - The on-chip bus arbiter arbitrates bus mastership among CPU and DMAC
- Other features
  - External bus release function*[2]

Notes: 1. Chip select $\overline{CS6}$ in area 6 is for the on-chip USB. Therefore it cannot be used as an external area. 8-bit bus mode, 3-state access, and no program wait state should be set for area 6. Access to the RTC related registers (address: H'FFFF40 to H'FFFF5F) follows the specification of area 7. 8-bit access, 3-state access, and no program wait state should be set for area 7.

2. These functions are not available in the H8S/2212 Group.

Figure 6.1 shows a block diagram of the bus controller.

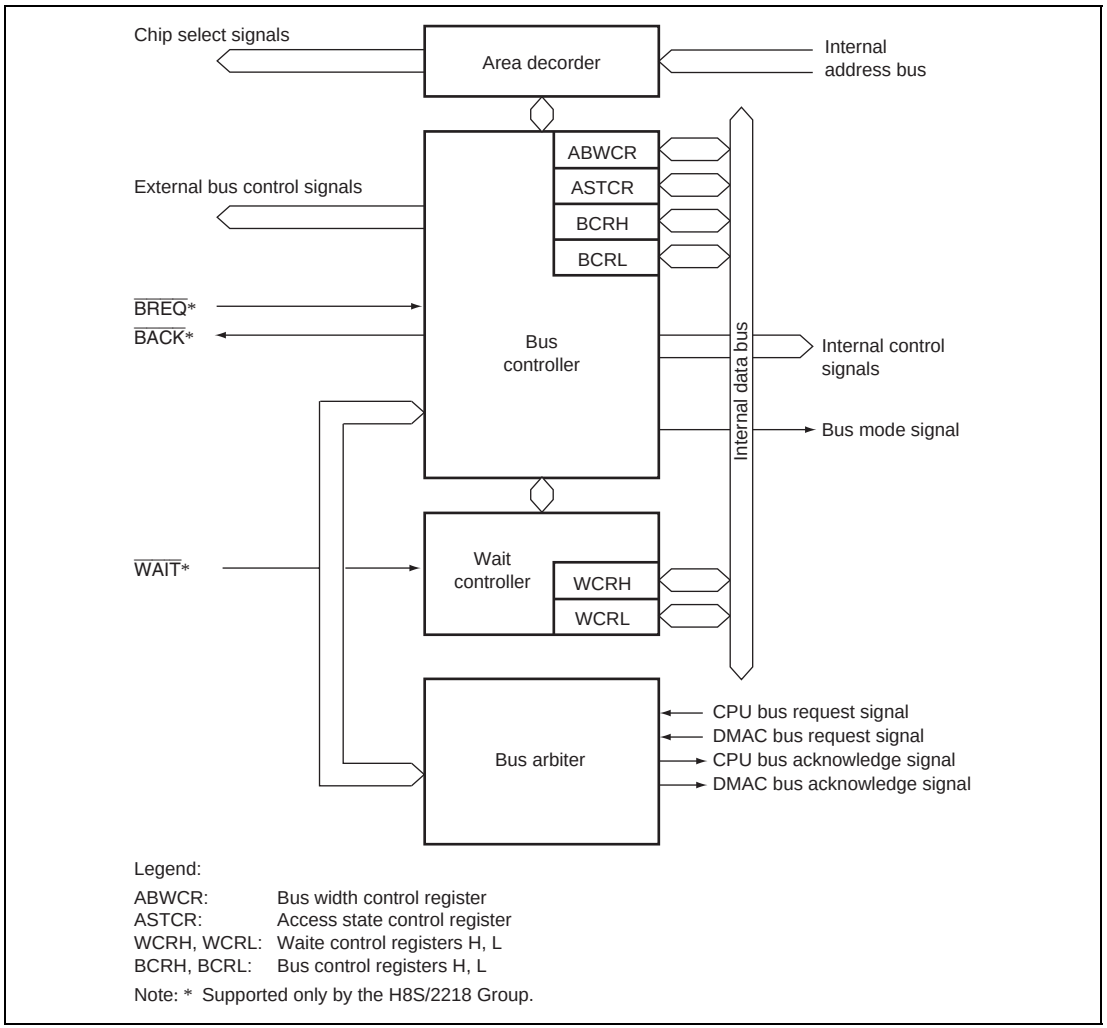



**Figure 6.1 Block Diagram of Bus Controller**

## 6.2 Input/Output Pins

Table 6.1 summarizes the pins of the bus controller.

These pins are supported only by the H8S/2218 Group.

**Table 6.1 Pin Configuration**

| Name | Symbol | I/O | Function |
|---|---|---|---|
| Address strove | $\overline{AS}$ | Output | Strobe signal indicating that address output on address bus is enabled. |
| Read | $\overline{RD}$ | Output | Strobe signal indicating that external space is being read. |
| High write | $\overline{HWR}$ | Output | Strobe signal indicating that external space is to be written, and upper half (D15 to D8) of data bus is enabled. |
| Low write | $\overline{LWR}$ | Output | Strobe signal indicating that external space is to be written, and lower half (D7 to D0) of data bus is enabled. |
| Chip select 0 to 5 | $\overline{CS0}$ to $\overline{CS5}$ | Output | Strobe signal indicating that areas 0 to 5 are selected. |
| Wait | $\overline{WAIT}$ | Input | Wait request signal when accessing external 3-state access space. |
| Bus request | $\overline{BREQ}$ | Input | Request signal that releases bus to external device. |
| Bus request acknowledge | $\overline{BACK}$ | Output | Acknowledge signal indicating that bus has been released. |

## 6.3 Register Descriptions

The following shows the registers of the bus controller.

- Bus width control register (ABWCR)
- Access state control register (ASTCR)
- Wait control register H (WCRH)
- Wait control register L (WCRL)
- Bus control register H (BCRH)
- Bus control register L (BCRL )
- Pin function control register (PFCR)

### 6.3.1 Bus Width Control Register (ABWCR)

ABWCR designates each area for either 8-bit access or 16-bit access.

ABWCR sets the data bus width for the external memory space. The bus width for on-chip memory and internal I/O registers except for the on-chip USB and RTC is fixed regardless of the settings in ABWCR.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | ABW7*[2] | 1/0*[1] | R/W | Area 7 to 0 Bus Width Control: |
| 6 | ABW6*[2] | 1/0*[1] | R/W | These bits select whether the corresponding area is to be designated for 8-bit access or 16-bit access. |
| 5 | ABW5 | 1/0*[1] | R/W | |
| 4 | ABW4 | 1/0*[1] | R/W | 0: Area n is designated for 16-bit access |
| 3 | ABW3 | 1/0*[1] | R/W | 1: Area n is designated for 8-bit access |
| 2 | ABW2 | 1/0*[1] | R/W | Legend: n = 7 to 0 |
| 1 | ABW1 | 1/0*[1] | R/W | |
| 0 | ABW0 | 1/0*[1] | R/W | |

Notes: 1. In modes 5 to 7, initial value of each bit is 1. In mode 4, initial value of each bit is 0. These bits should be set to 1 in the H8S/2212 Group.
2. The on-chip USB and on-chip RTC are allocated to area 6 and area 7, respectively. Therefore, these bits should be set to 1.

### 6.3.2 Access State Control Register (ASTCR)

ASTCR designates each area as either a 2-state access space or a 3-state access space.

ASTCR sets the number of access states for the external memory space. The number of access states for on-chip memory and internal I/O registers except for the on-chip USB is fixed regardless of the settings in ASTCR.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | AST7* | 1 | R/W | Area 7 to 0 Access State Control: |
| 6 | AST6* | 1 | R/W | These bits select whether the corresponding area is to be designated as a 2-state access space or a 3-state access space. Wait state insertion is enabled or disabled at the same time. |
| 5 | AST5 | 1 | R/W | |
| 4 | AST4 | 1 | R/W | |
| 3 | AST3 | 1 | R/W | 0: Area n is designated for 2-state access |
| 2 | AST2 | 1 | R/W | Wait state insertion in area n external space is disabled |
| 1 | AST1 | 1 | R/W | 1: Area n is designated for 3-state access |
| 0 | AST0 | 1 | R/W | Wait state insertion in area n external space is enabled |
| | | | | Legend: n = 7 to 0 |

Note: * The on-chip USB and on-chip RTC are allocated to area 6 and area 7, respectively. Therefore, these bits should be set to 1.

### 6.3.3 Wait Control Registers H and L (WCRH, WCRL)

WCRH and WCRL select the number of program wait states for each area.

Program waits are not inserted in the case of on-chip memory or internal I/O registers except for the on-chip USB.

- WCRH

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | W71* | 1 | R/W | Area 7 Wait Control 1 and 0 |
| 6 | W70* | 1 | R/W | These bits select the number of program wait states when area 7 in external space is accessed while the AST7 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 7 is accessed<br>01: 1 program wait state inserted when external space area 7 is accessed<br>10: 2 program wait states inserted when external space area 7 is accessed<br>11: 3 program wait states inserted when external space area 7 is accessed |
| 5 | W61* | 1 | R/W | Area 6 Wait Control 1 and 0 |
| 4 | W60* | 1 | R/W | These bits select the number of program wait states when area 6 in external space is accessed while the AST6 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 6 is accessed<br>01: 1 program wait state inserted when external space area 6 is accessed<br>10: 2 program wait states inserted when external space area 6 is accessed<br>11: 3 program wait states inserted when external space area 6 is accessed |

Note: * The on-chip USB and on-chip RTC are allocated to area 6 and area 7, respectively. Therefore, these bits should be set to 0.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | W51 | 1 | R/W | Area 5 Wait Control 1 and 0 |
| 2 | W50 | 1 | R/W | These bits select the number of program wait states when area 5 in external space is accessed while the AST5 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 5 is accessed<br>01: 1 program wait state inserted when external space area 5 is accessed<br>10: 2 program wait states inserted when external space area 5 is accessed<br>11: 3 program wait states inserted when external space area 5 is accessed |
| 1 | W41 | 1 | R/W | Area 4 Wait Control 1 and 0 |
| 0 | W40 | 1 | R/W | These bits select the number of program wait states when area 4 in external space is accessed while the AST4 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 4 is accessed<br>01: 1 program wait state inserted when external space area 4 is accessed<br>10: 2 program wait states inserted when external space area 4 is accessed<br>11: 3 program wait states inserted when external space area 4 is accessed |

- WCRL

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | W31 | 1 | R/W | Area 3 Wait Control 1 and 0 |
| 6 | W30 | 1 | R/W | These bits select the number of program wait states when area 3 in external space is accessed while the AST3 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 3 is accessed<br>01: 1 program wait state inserted when external space area 3 is accessed<br>10: 2 program wait states inserted when external space area 3 is accessed<br>11: 3 program wait states inserted when external space area 3 is accessed |
| 5 | W21 | 1 | R/W | Area 2 Wait Control 1 and 0 |
| 4 | W20 | 1 | R/W | These bits select the number of program wait states when area 2 in external space is accessed while the AST2 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 2 is accessed<br>01: 1 program wait state inserted when external space area 2 is accessed<br>10: 2 program wait states inserted when external space area 2 is accessed<br>11: 3 program wait states inserted when external space area 2 is accessed |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | W11 | 1 | R/W | Area 1 Wait Control 1 and 0 |
| 2 | W10 | 1 | R/W | These bits select the number of program wait states when area 1 in external space is accessed while the AST1 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 1 is accessed<br>01: 1 program wait state inserted when external space area 1 is accessed<br>10: 2 program wait states inserted when external space area 1 is accessed<br>11: 3 program wait states inserted when external space area 1 is accessed |
| 1 | W01 | 1 | R/W | Area 0 Wait Control 1 and 0 |
| 0 | W00 | 1 | R/W | These bits select the number of program wait states when area 0 in external space is accessed while the AST0 bit in ASTCR is set to 1.<br>00: Program wait not inserted when external space area 0 is accessed<br>01: 1 program wait state inserted when external space area 0 is accessed<br>10: 2 program wait states inserted when external space area 0 is accessed<br>11: 3 program wait states inserted when external space area 0 is accessed |

### 6.3.4 Bus Control Register H (BCRH)

BCRH selects enabling or disabling of idle cycle insertion, and the memory interface for area 0. This register should be set initial value and not be modified in the H8S/2212 Group.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | ICIS1 | 1 | R/W | Idle Cycle Insert 1:<br>Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read cycles are performed in different areas.<br>0: Idle cycle not inserted in case of successive external read cycles in different areas<br>1: Idle cycle inserted in case of successive external read cycles in different areas |
| 6 | ICIS0 | 1 | R/W | Idle Cycle Insert 0:<br>Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read and write cycles are performed.<br>0: Idle cycle not inserted in case of successive external read and write cycles<br>1: Idle cycle inserted in case of successive external read and write cycles |
| 5 | BRSTRM | 0 | R/W | Burst ROM enable:<br>Selects whether area 0 is used as a burst ROM interface.<br>0: Area 0 is basic bus interface<br>1: Area 0 is burst ROM interface |
| 4 | BRSTS1 | 1 | R/W | Burst Cycle Select 1:<br>Selects the number of burst cycles for the burst ROM interface.<br>0: Burst cycle comprises 1 state<br>1: Burst cycle comprises 2 states |
| 3 | BRSTS0 | 0 | R/W | Burst Cycle Select 0:<br>Selects the number of words that can be accessed in a burst ROM interface burst access.<br>0: Max. 4 words in burst access<br>1: Max. 8 words in burst access |
| 2 to 0 | – | All 0 | R/W | Reserved<br>The write value should always be 0. |

### 6.3.5 Bus Control Register L (BCRL)

BCRL performs selection of the external bus-released state protocol, and enabling or disabling of $\overline{\text{WAIT}}$ pin input.

The functions selected by this register are available only in the H8S/2218 Group. This register should not be modified in the H8S/2212 Group.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | BRLE* | 0 | R/W | Bus Release Enable<br>Enables or disables external bus release.<br>0: External bus release is disabled. $\overline{\text{BREQ}}$ and $\overline{\text{BACK}}$ can be used as I/O ports.<br>1: External bus release is enabled. |
| 6 | – | 0 | R/W | Reserved<br>The write value should always be 0. |
| 5 | – | 0 | – | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 4 | – | 0 | R/W | Reserved<br>The write value should always be 0. |
| 3 | – | 1 | R/W | Reserved<br>The write value should always be 1. |
| 2, 1 | – | All 0 | R/W | Reserved<br>The write value should always be 0. |
| 0 | WAITE* | 0 | R/W | WAIT Pin Enable<br>Selects enabling or disabling of wait input by the $\overline{\text{WAIT}}$ pin.<br>0: Wait input by $\overline{\text{WAIT}}$ pin disabled. $\overline{\text{WAIT}}$ pin can be used as I/O port.<br>1: Wait input by $\overline{\text{WAIT}}$ pin enabled. |

Note: * These bits should be set to 0 in the H8S/2212 Group.

### 6.3.6 Pin Function Control Register (PFCR)

PFCR performs address output control in external extended mode. When using the USB with the emulator (E6000), enable the A8 and A9 output by setting AE3 to AE0 to 0010.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | R/W | Reserved<br>The write value should always be 0. |
| 3 | AE3 | 1/0* | R/W | Address Output Enable 3 to 0<br>These bits select enabling or disabling of address outputs A8 to A23 in ROMless extended mode and modes with ROM.<br>When a pin is enabled for address output, the address is output regardless of the corresponding DDR setting. When a pin is disabled for address output, it becomes an output port when the corresponding DDR bit is set to 1.<br>0000: A8 to A23 output disabled (initial value of mode 6 and 7)<br>0001: A8 output enabled; A9 to A23 output disabled<br>0010: A8, A9 output enabled; A10 to A23 output disabled<br>0011: A8 to A10 output enabled; A11 to A23 output disabled<br>0100: A8 to A11 output enabled; A12 to A23 output disabled<br>0101: A8 to A12 output enabled; A13 to A23 output disabled<br>0110: A8 to A13 output enabled; A14 to A23 output disabled<br>0111: A8 to A14 output enabled; A15 to A23 output disabled<br>1000: A8 to A15 output enabled; A16 to A23 output disabled<br>1001: A8 to A16 output enabled; A17 to A23 output disabled<br>1010: A8 to A17 output enabled; A18 to A23 output disabled<br>1011: A8 to A18 output enabled; A19 to A23 output disabled<br>1100: A8 to A19 output enabled; A20 to A23 output disabled<br>1101: A8 to A20 output enabled; A21 to A23 output disabled (initial value of modes 4 and 5)<br>1110: A8 to A21 output enabled; A22, A23 output disabled<br>1111: A8 to A23 output enabled |
| 2 | AE2 | 1/0* | R/W | |
| 1 | AE1 | 0 | R/W | |
| 0 | AE0 | 1/0* | R/W | |

Note: * In modes 4 and 5, initial value of each bit is 1. In modes 6 and 7, initial value of each bit is 0.

## 6.4 Bus Control

### 6.4.1 Area Divisions

In advanced mode, the bus controller partitions the 16 Mbytes address space into eight areas, 0 to 7, in 2-Mbyte units, and performs bus control for external space in area units. In normal mode*, it controls a 64-kbyte address space comprising part of area 0.

Figure 6.2 shows an outline of the memory map.

Chip select signals ($\overline{CS0}$ to $\overline{CS5}$) can be output for areas 0 to 5.

Note: * Not available in this LSI.

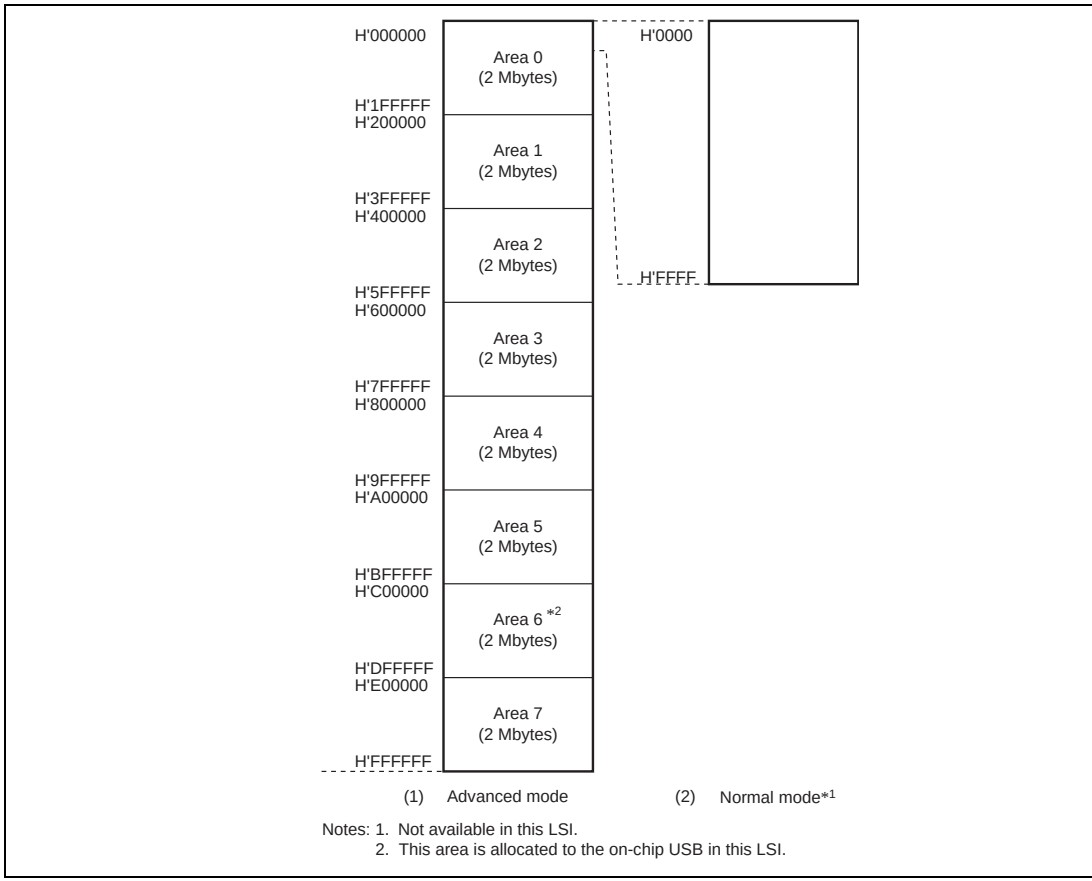



**Figure 6.2 Overview of Area Divisions**

### 6.4.2 Bus Specifications

The external space bus specifications consist of three elements: bus width, number of access states, and number of program wait states.

The bus width and number of access states for memory and internal I/O registers except for the on-chip USB and RTC are fixed, and are not affected by the bus controller.

**Bus Width:** A bus width of 8 or 16 bits can be selected with ABWCR. An area for which an 8-bit bus is selected functions as an 8-bit access space, and an area for which a 16-bit bus is selected functions as a16-bit access space.

If all areas are designated for 8-bit access, 8-bit bus mode is set; if any area is designated for 16-bit access, 16-bit bus mode is set. When the burst ROM interface is designated, 16-bit bus mode is always set. 8-bit bus mode should be set for area 6 and area 7 in this LSI.

**Number of Access States:** Two or three access states can be selected with ASTCR.

An area for which 2-state access is selected functions as a 2-state access space, and an area for which 3-state access is selected functions as a 3-state access space.

With the burst ROM interface, the number of access states may be determined without regard to ASTCR.

When 2-state access space is designated, wait insertion is disabled.

Area 6 and area 7 should be set to function as a 3-state access space in this LSI.

**Number of Program Wait States:** When 3-state access space is designated by ASTCR, the number of program wait states to be inserted automatically is selected with WCRH and WCRL.

From 0 to 3 program wait states can be selected.

The number of program wait states in area 6 and area 7 should be set to 0 in this LSI.

**Table 6.2 Bus Specifications for Each Area (Basic Bus Interface)**

| ABWCR | ASTCR | WCRH, WCRL | | Bus Specifications (Basic Bus Interface) | | |
|---|---|---|---|---|---|---|
| ABWn | ASTn | Wn1 | Wn0 | Bus Width | Number of Access States | Number of Program Wait States |
| 0 | 0 | — | — | 16 | 2 | 0 |
| | 1 | 0 | 0 | | 3 | 0 |
| | | | 1 | | | 1 |
| | | 1 | 0 | | | 2 |
| | | | 1 | | | 3 |
| 1 | 0 | — | — | 8 | 2 | 0 |
| | 1 | 0 | 0 | | 3 | 0 |
| | | | 1 | | | 1 |
| | | 1 | 0 | | | 2 |
| | | | 1 | | | 3 |

### 6.4.3 Bus Interface for Each Area

The initial state of each area is basic bus interface, 3-state access space. The initial bus width is selected according to the operating mode. The bus specifications described here cover basic items only, and the sections on each memory interface (section 6.6, Basic Bus Interface and section 6.7, Burst ROM Interface) should be referred to for further details. Note that the ROM is always enabled and no external extended mode in the H8S/2212 Group.

**Area 0:** Area 0 includes on-chip ROM, and in ROM-disabled extended mode, all of area 0 is external space. In ROM-enabled extended mode, the space excluding on-chip ROM is external space.

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

Either basic bus interface or burst ROM interface can be selected for area 0.

**Areas 1 to 6:** In external extended mode, all of areas 1 to 6 is external space. When area 1 to 5 external space is accessed, the $\overline{CS1}$ to $\overline{CS5}$ pin signals respectively can be output. Only the basic bus interface can be used for areas 1 to 5. Area 6 is only for the on-chip USB. For details, see section 14, Universal Serial Bus (USB).

**Area 7:** Area 7 includes the on-chip RAM and internal l/O registers. In external extended mode, the space excluding the reserved area (for details, see section 3.4, Memory Map in Each Operating Mode) the on-chip RAM and internal l/O registers except on-chip RTC, is external space. The on-chip RAM is enabled when the RAME bit in the system control register (SYSCR) is set to 1; when

the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding space becomes external space.

Only the basic bus interface can be used for the area 7.

### 6.4.4 Chip Select Signals

In the H8S/2218 Group chip select signals ($\overline{CS0}$ to $\overline{CS5}$) can be output to areas 0 to 5, the signal being driven low when the corresponding external space area is accessed. Figure 6.3 shows an example of $\overline{CSn}$ (n = 0 to 5) output timing. Enabling or disabling of the $\overline{CSn}$ signal is performed by setting the data direction register (DDR) for the port corresponding to the particular $\overline{CSn}$ pin.

In ROM-disabled extended mode, the $\overline{CS0}$ pin is placed in the output state after a power-on reset. Pins $\overline{CS1}$ to $\overline{CS5}$ are placed in the input state after a power-on reset, and so the corresponding DDR should be set to 1 when outputting signals $\overline{CS1}$ to $\overline{CS5}$.

In ROM-enabled extended mode, pins $\overline{CS0}$ to $\overline{CS5}$ are all placed in the input state after a power-on reset, and so the corresponding DDR should be set to 1 when outputting signals $\overline{CS0}$ to $\overline{CS5}$. For details, see section 8, I/O Ports.

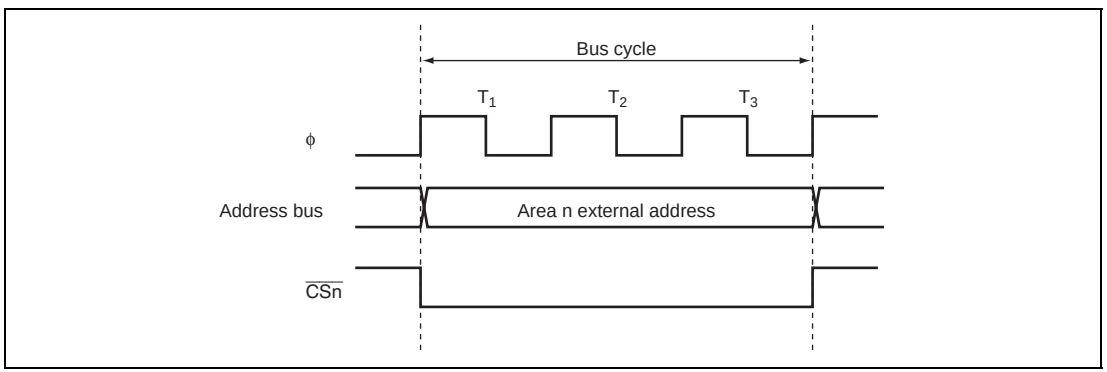



**Figure 6.3 $\overline{CSn}$ Signal Output Timing (n = 0 to 5)**

## 6.5 Basic Timing

The CPU is driven by a system clock (ϕ), denoted by the symbol ϕ. The period from one rising edge of ϕ to the next is referred to as a "state." The memory cycle or bus cycle consists of one, two, or three states. Different methods are used to access on-chip memory, on-chip peripheral modules, and the external address space.

### 6.5.1 On-Chip Memory (ROM, RAM) Access Timing

On-chip memory is accessed in one state. The data bus is 16 bits wide, permitting both byte and word transfer instruction. Figure 6.4 shows the on-chip memory access cycle. Figure 6.5 shows the pin states.

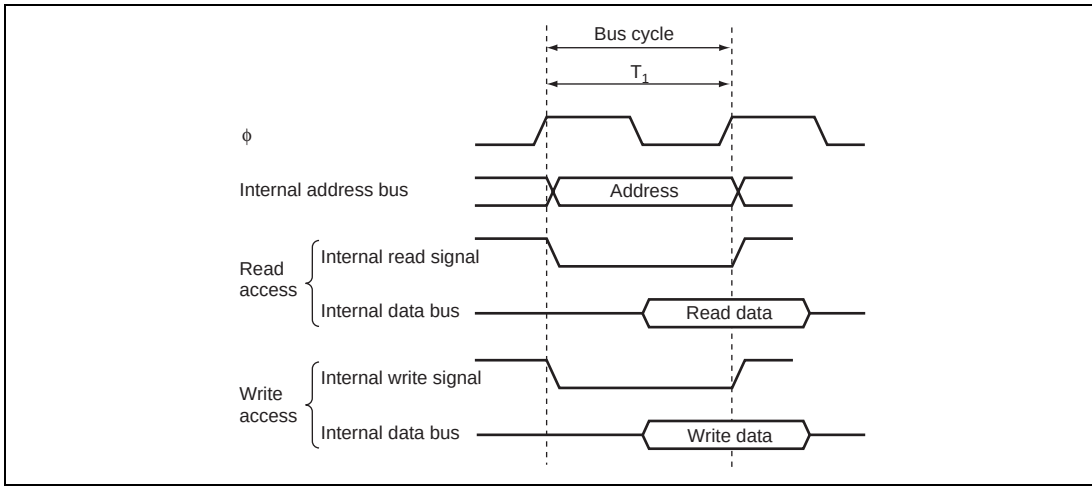



**Figure 6.4 On-Chip Memory Access Cycle**

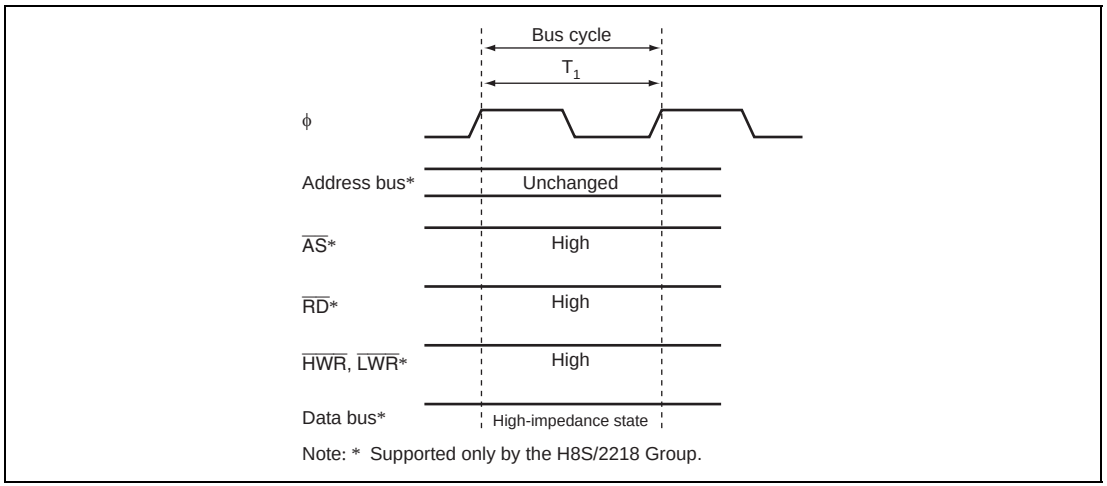



**Figure 6.5 Pin States during On-Chip Memory Access**

### 6.5.2 On-Chip Peripheral Module Access Timing

The on-chip peripheral modules are accessed in two states except on-chip USB and RTC. The data bus is either 8 bits or 16 bits wide, depending on the particular internal I/O register being accessed. Figure 6.6 shows the access timing for the on-chip peripheral modules. Figure 6.7 shows the pin states.

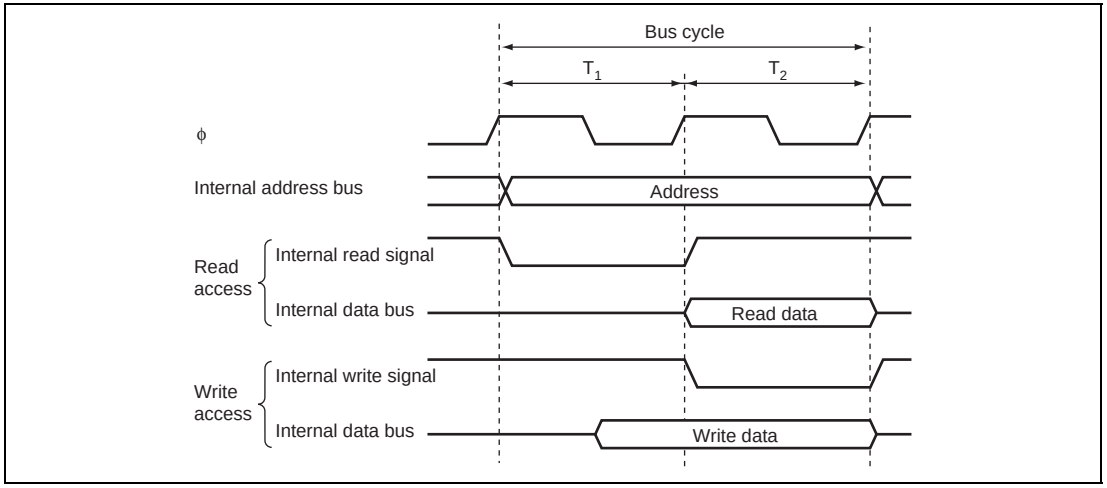



**Figure 6.6 On-Chip Peripheral Module Access Cycle**

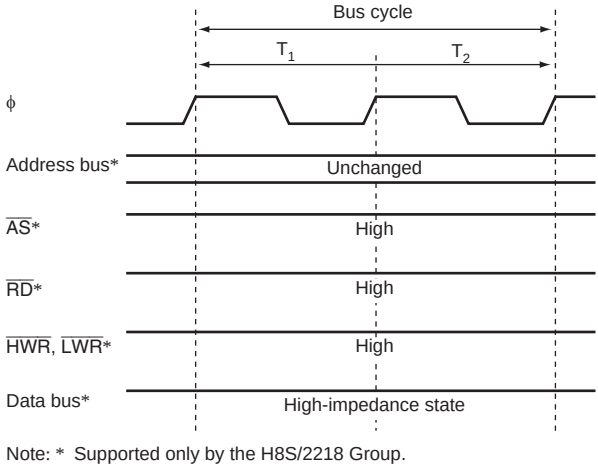



**Figure 6.7 Pin States during On-Chip Peripheral Module Access**

### 6.5.3 External Address Space Access Timing

The external address space is accessed with an 8-bit or 16-bit data bus width in a two-state or three-state bus cycle. In three-state access, wait states can be inserted. For further details, refer to section 6.6.3, Basic Timing.

## 6.6 Basic Bus Interface

The basic bus interface enables direct connection of ROM, SRAM, and so on.

### 6.6.1 Data Size and Data Alignment (Supported Only by the H8S/2218 Group)

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

**8-Bit Access Space:** Figure 6.8 illustrates data alignment control for the 8-bit access space. With the 8-bit access space, the upper data bus (D15 to D8) is always used for accesses. The amount of data that can be accessed at one time is one byte: a word transfer instruction is performed as two-byte accesses, and a longword transfer instruction, as four-byte accesses.

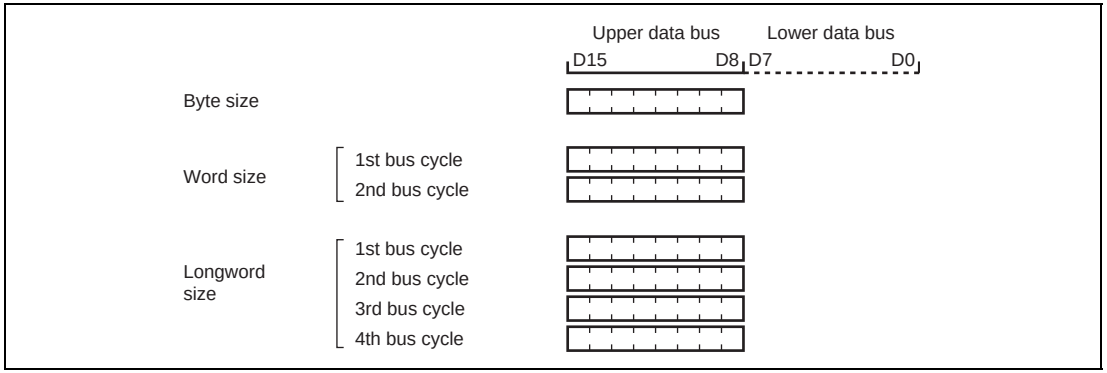



**Figure 6.8 Access Sizes and Data Alignment Control (8-Bit Access Space)**

**16-Bit Access Space:** Figure 6.9 illustrates data alignment control for the 16-bit access space. With the 16-bit access space, the upper data bus (D15 to D8) and lower data bus (D7 to D0) are used for accesses. The amount of data that can be accessed at one time is one byte or one word, and a longword transfer instruction is executed as two word transfer instructions.

In byte access, whether the upper or lower data bus is used is determined by whether the address is even or odd. The upper data bus is used for an even address, and the lower data bus for an odd address.

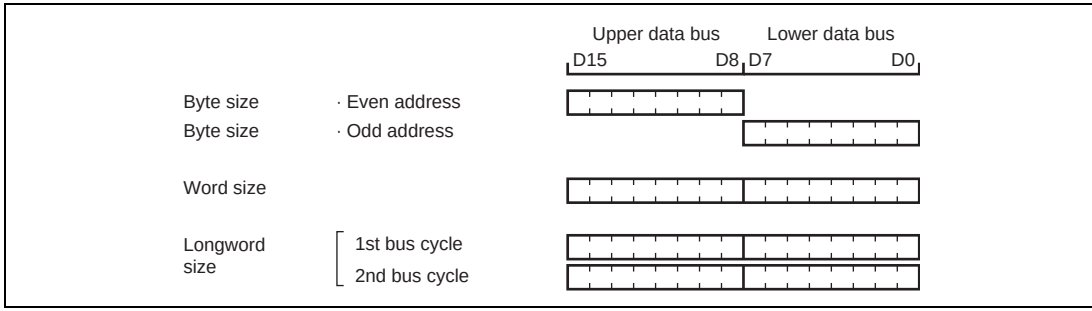



**Figure 6.9 Access Sizes and Data Alignment Control (16-Bit Access Space)**

### 6.6.2 Valid Strobes

Table 6.3 shows the data buses used and valid strobes for the access spaces in the H8S/2218 Group.

In a read, the $\overline{RD}$ signal is valid without discrimination between the upper and lower halves of the data bus.

In a write, the $\overline{HWR}$ signal is valid for the upper half of the data bus, and the $\overline{LWR}$ signal for the lower half.

The $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$ signals are not available in the H8S/2212 Group.

**Table 6.3 Data Buses Used and Valid Strobes**

| Area | Access Size | Read/ Write | Address | Valid Strobe | Upper Data Bus (D15 to D8) | Lower Data Bus (D7 to D0) |
|---|---|---|---|---|---|---|
| 8-bit access space | Byte | Read | – | $\overline{RD}$ | Valid | Invalid |
| | | Write | – | $\overline{HWR}$ | | Hi-Z |
| 16-bit access space | Byte | Read | Even | $\overline{RD}$ | Valid | Invalid |
| | | | Odd | | Invalid | Valid |
| | | Write | Even | $\overline{HWR}$ | Valid | Hi-Z |
| | | | Odd | $\overline{LWR}$ | Hi-Z | Valid |
| | Word | Read | – | $\overline{RD}$ | Valid | Valid |
| | | Write | – | $\overline{HWR}$, $\overline{LWR}$ | | |

Notes:
Hi-Z: High impedance.
Invalid: Input state: input value is ignored.

### 6.6.3 Basic Timing

**8-Bit 2-State Access Space:** Figure 6.10 shows the bus timing for an 8-bit 2-state access space in the H8S/2218 Group. When an 8-bit access space is accessed, the upper half (D15 to D8) of the data bus is used.

Wait states cannot be inserted.

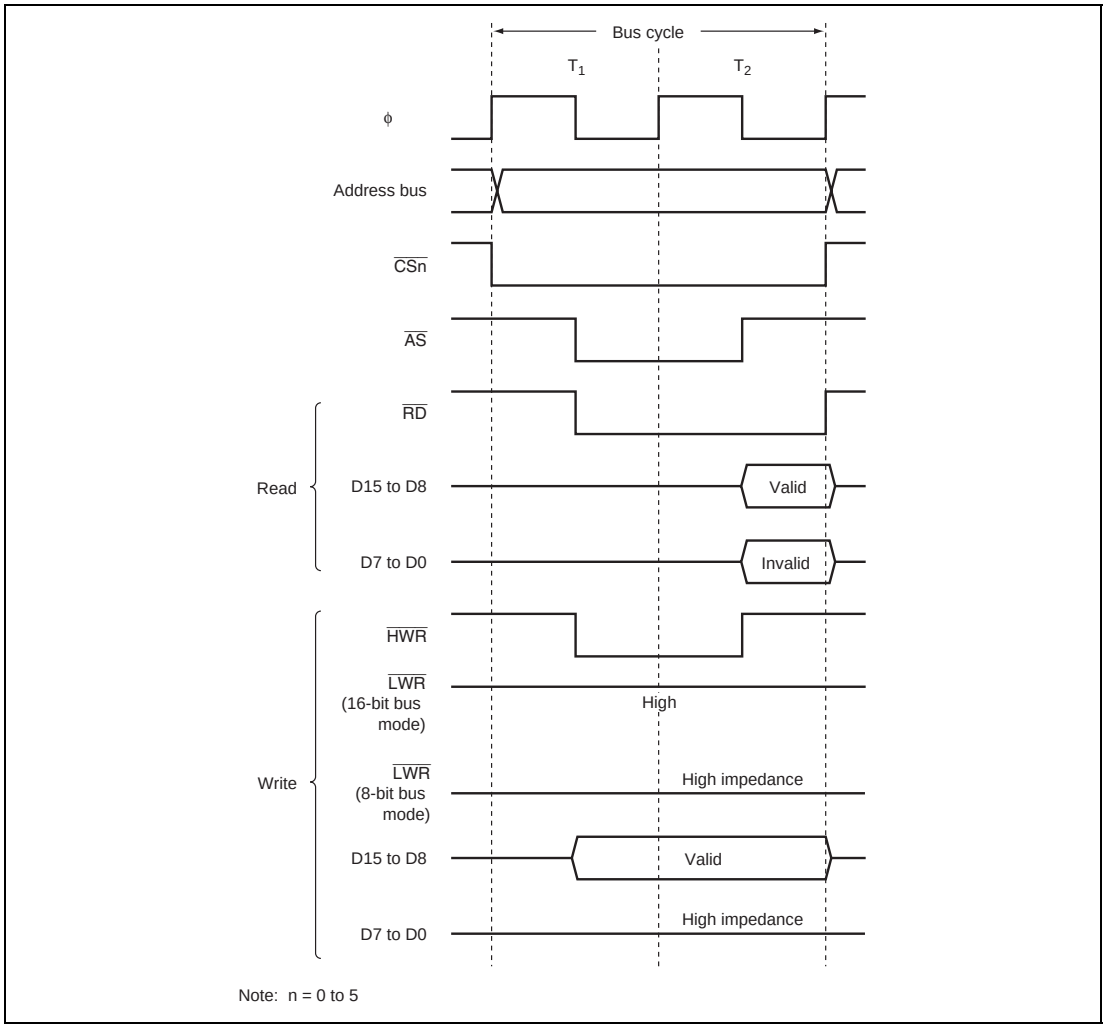



**Figure 6.10 Bus Timing for 8-Bit 2-State Access Space**

**8-Bit 3-State Access Space (Except Area 6):** Figure 6.11 shows the bus timing for an 8-bit 3-state access space in the H8S/2218 Group. When an 8-bit access space is accessed, the upper half (D15 to D8) of the data bus is used.

Wait states can be inserted.

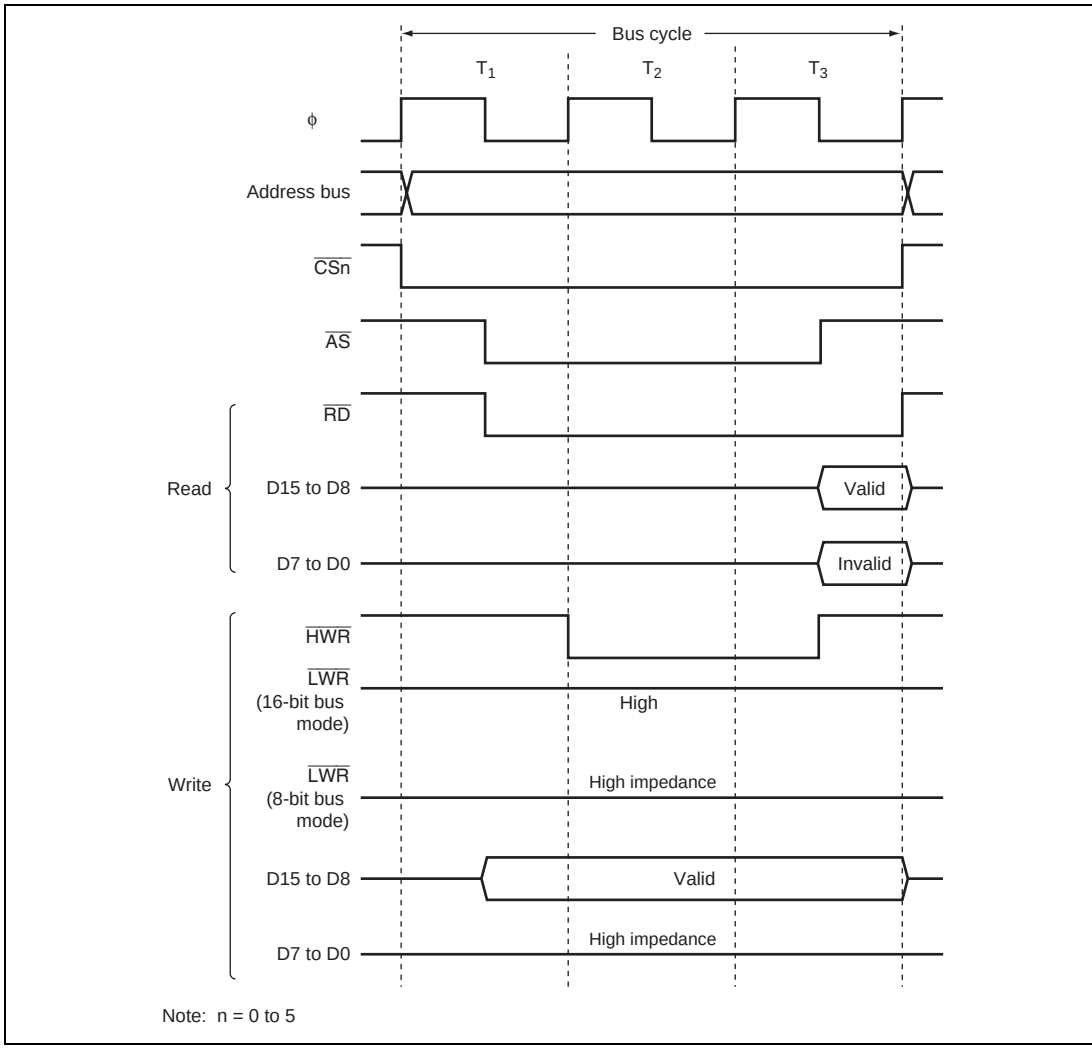



**Figure 6.11 Bus Timing for 8-Bit 3-State Access Space (Except Area 6)**

**8-Bit 3-State Access Space (Area 6 and RTC):** Figure 6.12 shows the bus timing for area 6 and RTC area (address = H'FFFF40 to H'FFFF5F). When the areas are accessed, the data bus cannot be used.

Wait states cannot be inserted.

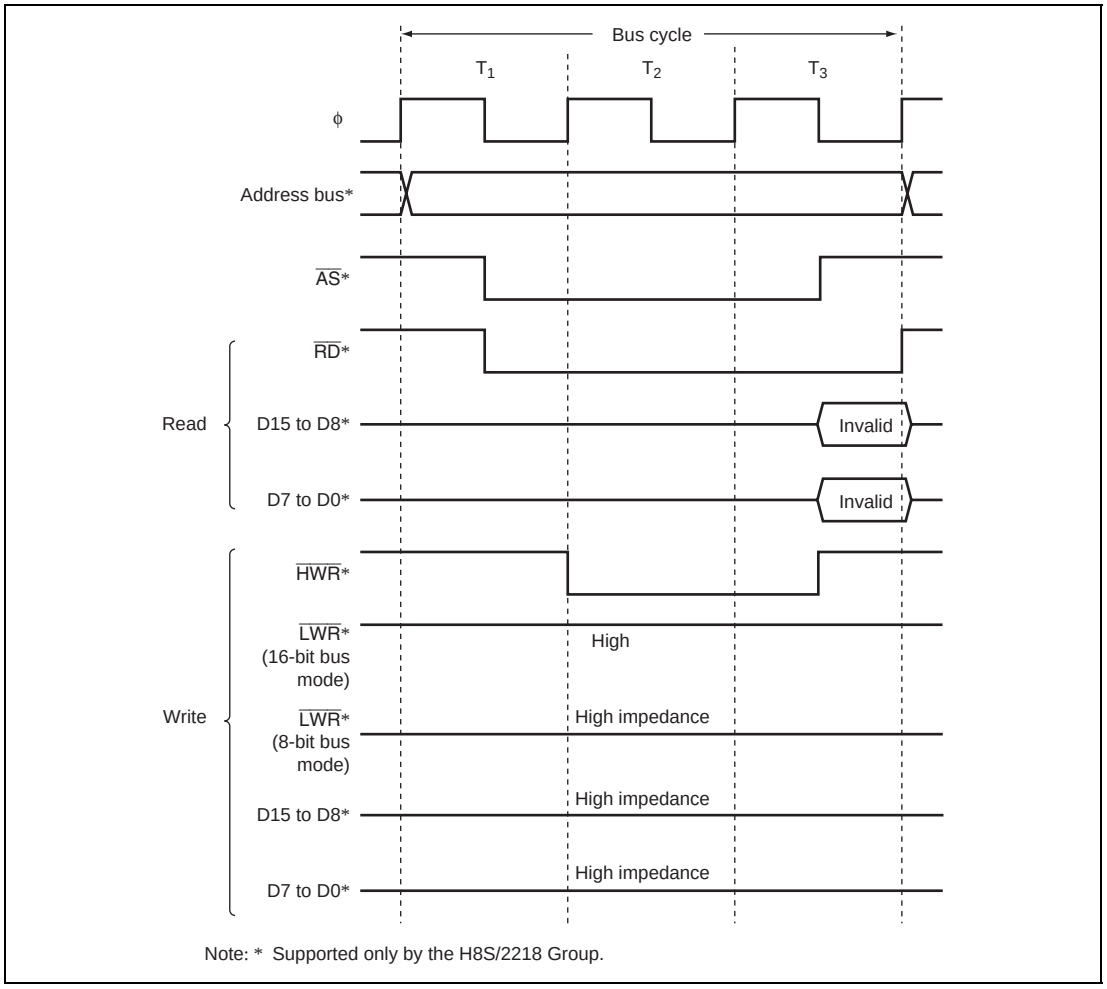



**Figure 6.12 Bus Timing for Area 6 and RTC**

**16-Bit 2-State Access Space:** Figures 6.13 to 6.15 show bus timings for a 16-bit 2-state access space in the H8S/2218 Group. When a 16-bit access space is accessed, the upper half (D15 to D8) of the data bus is used for the even address, and the lower half (D7 to D0) for the odd address.

Wait states cannot be inserted.

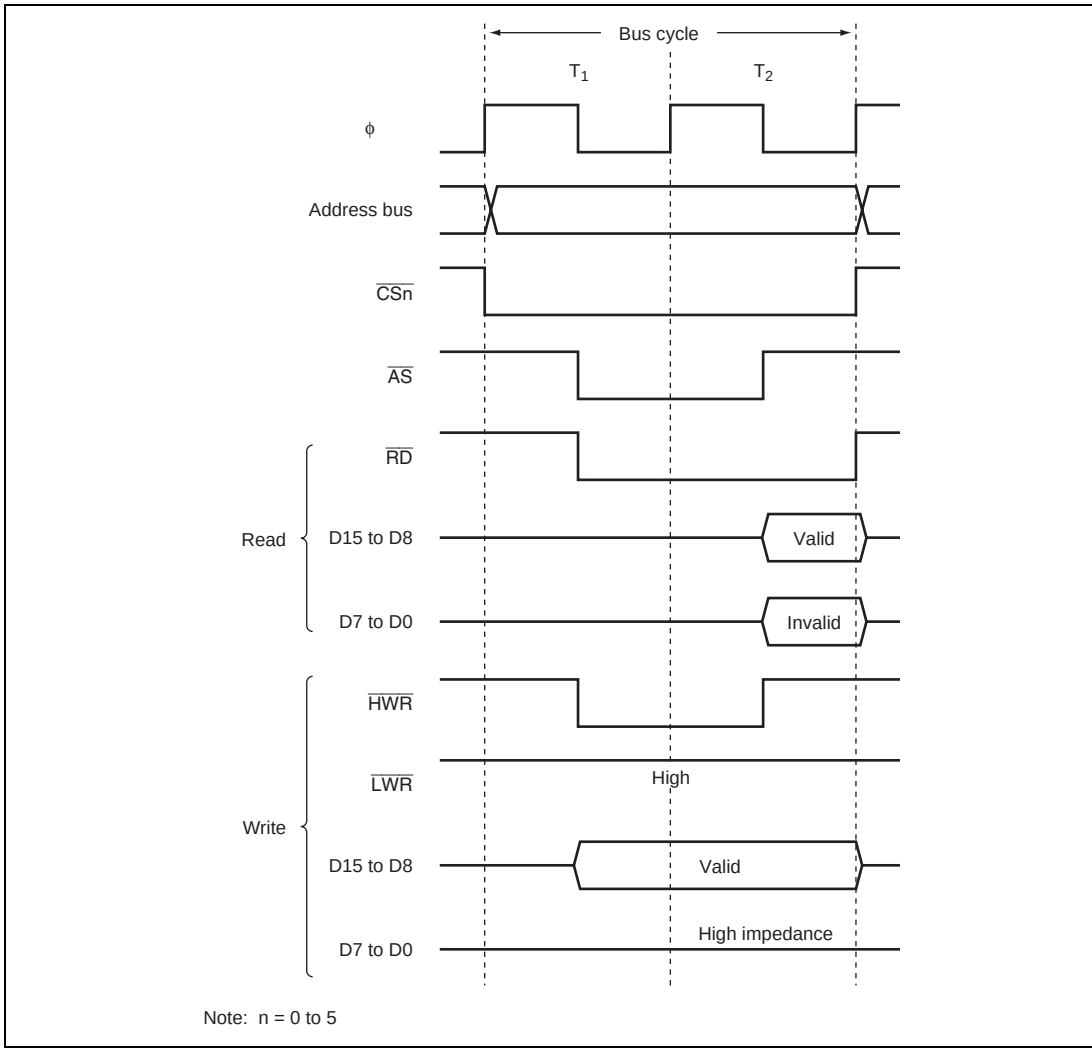



**Figure 6.13 Bus Timing for 16-Bit 2-State Access Space (1) (Even Address Byte Access)**

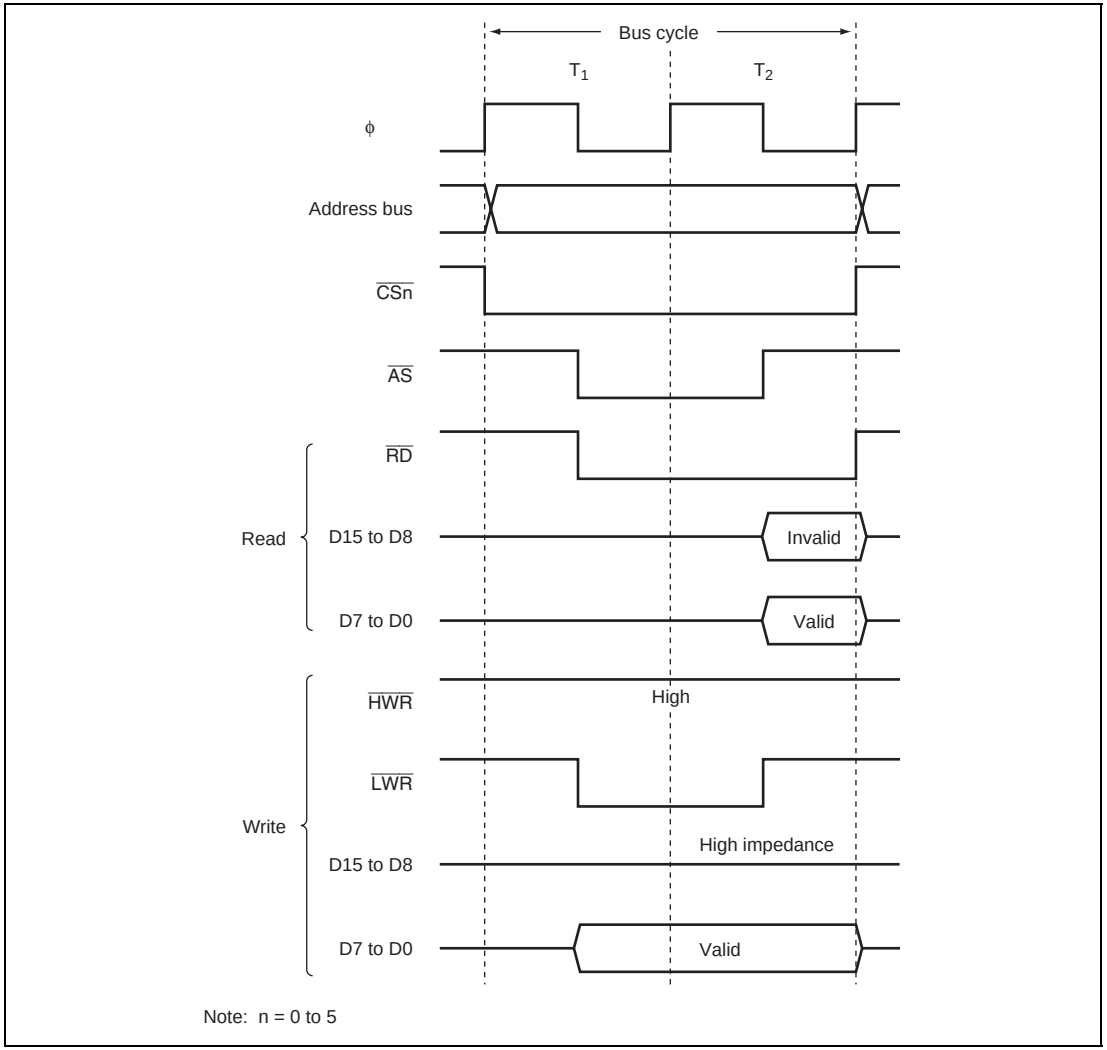



**Figure 6.14 Bus Timing for 16-Bit 2-State Access Space (2) (Odd Address Byte Access)**

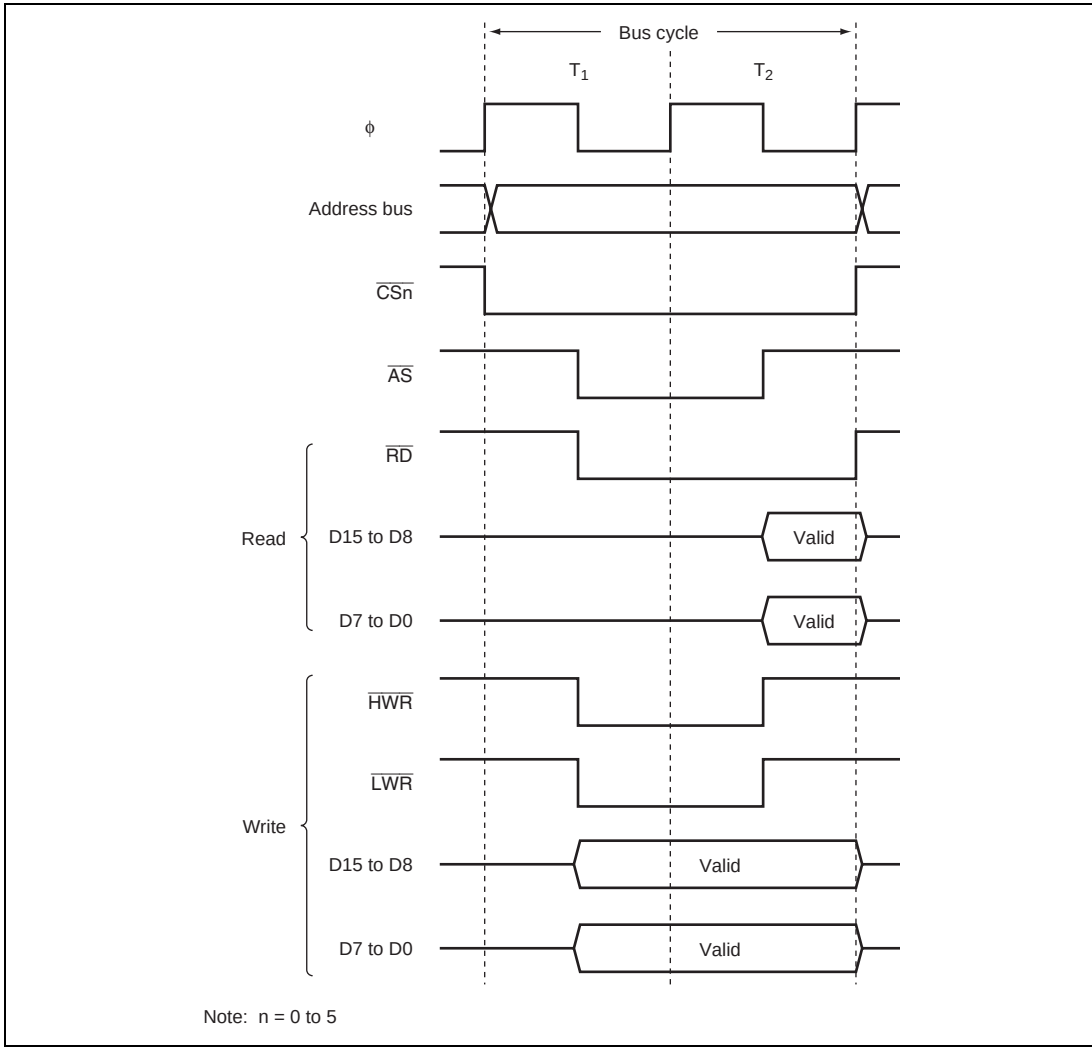



**Figure 6.15 Bus Timing for 16-Bit 2-State Access Space (3) (Word Access)**

**16-Bit 3-State Access Space:** Figures 6.16 to 6.18 show bus timings for a 16-bit 3-state access space in the H8S/2218 Group. When a 16-bit access space is accessed , the upper half (D15 to D8) of the data bus is used for the even address, and the lower half (D7 to D0) for the odd address.

Wait states can be inserted.

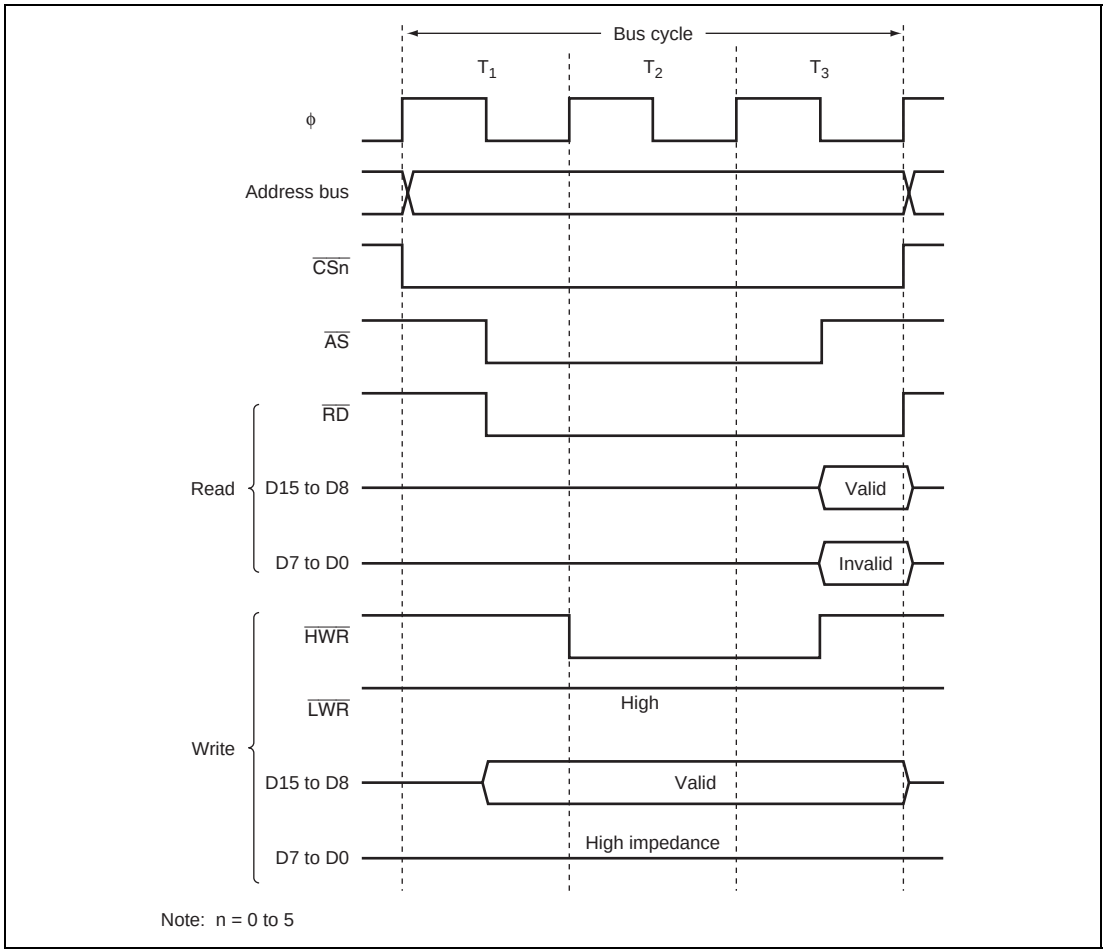



**Figure 6.16 Bus Timing for 16-Bit 3-State Access Space (1) (Even Address Byte Access)**

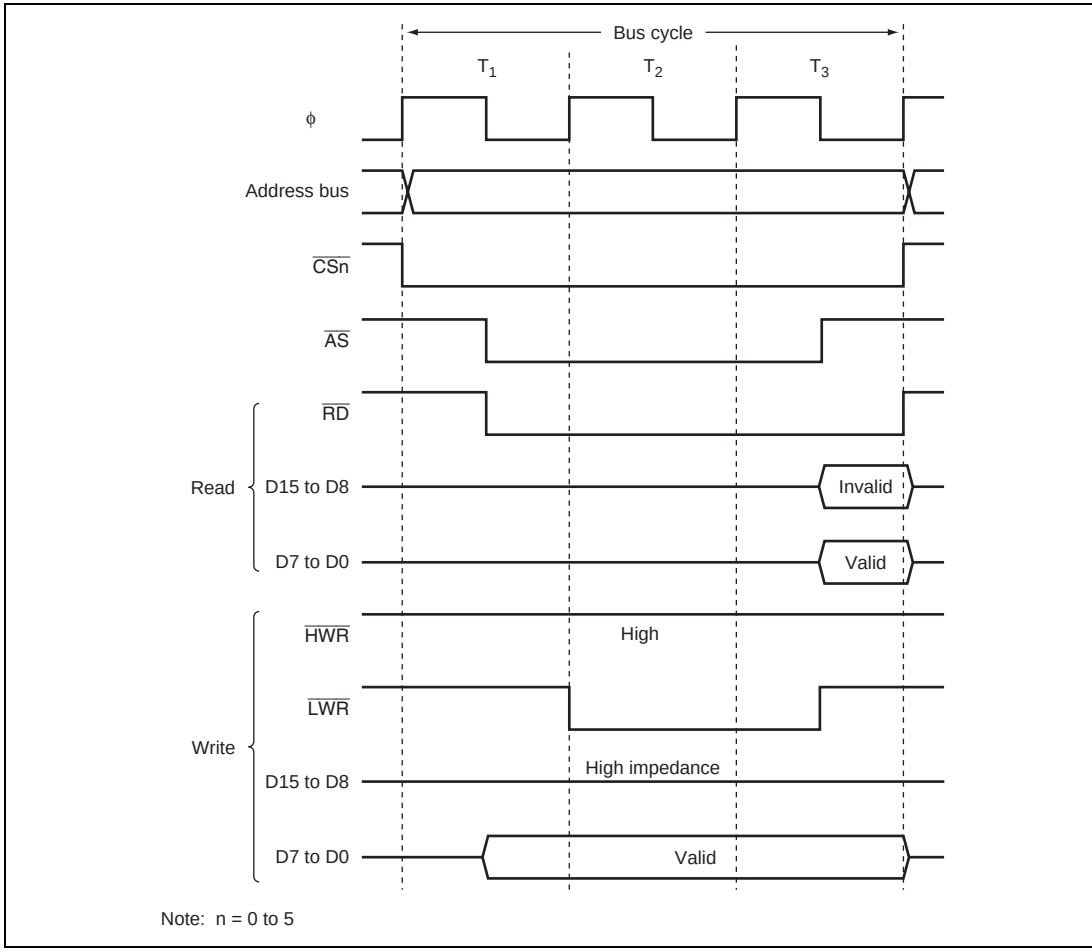



**Figure 6.17 Bus Timing for 16-Bit 3-State Access Space (2) (Odd Address Byte Access)**

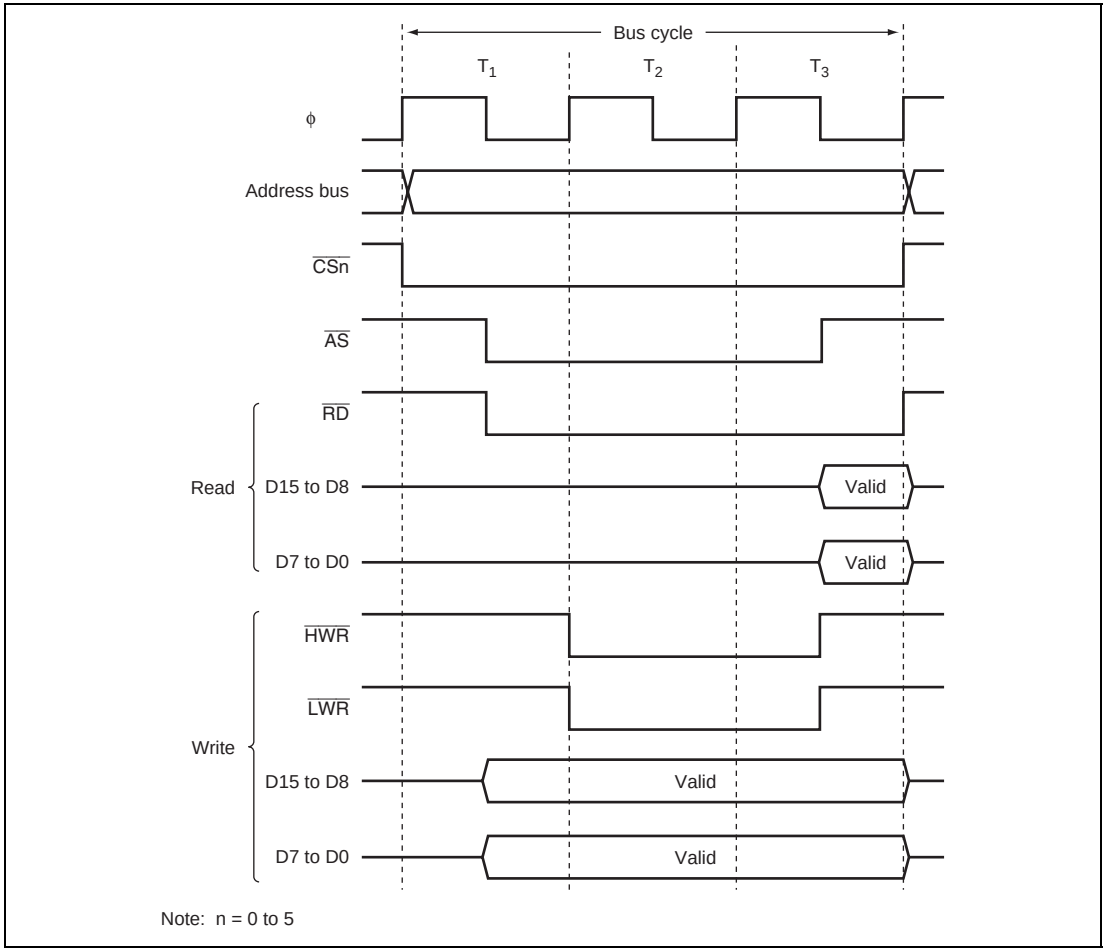



**Figure 6.18 Bus Timing for 16-Bit 3-State Access Space (3) (Word Access)**

### 6.6.4 Wait Control

When accessing external space, this LSI can extend the bus cycle by inserting one or more wait states (Tw). There are two ways of inserting wait states: program wait insertion and pin wait insertion using the $\overline{\text{WAIT}}$ pin.

**Program Wait Insertion:** From 0 to 3 wait states can be inserted automatically between the $T_2$ state and $T_3$ state on an individual area basis in 3-state access space, according to the settings of WCRH and WCRL.

**Pin Wait Insertion:** Setting the WAITE bit in BCRH to 1 enables wait insertion by means of the $\overline{\text{WAIT}}$ pin in the H8S/2218 Group. When external space is accessed in this state, program wait insertion is first carried out according to the settings in WCRH and WCRL. Then, if the $\overline{\text{WAIT}}$ pin is low at the falling edge of $\phi$ in the last $T_2$ or $T_W$ state, a $T_W$ state is inserted. If the $\overline{\text{WAIT}}$ pin is held low, $T_W$ states are inserted until it goes high.

Figure 6.19 shows an example of wait state insertion timing.

In the H8S/2212 Group, the WAITE bit in BCRH should not be set to 1.

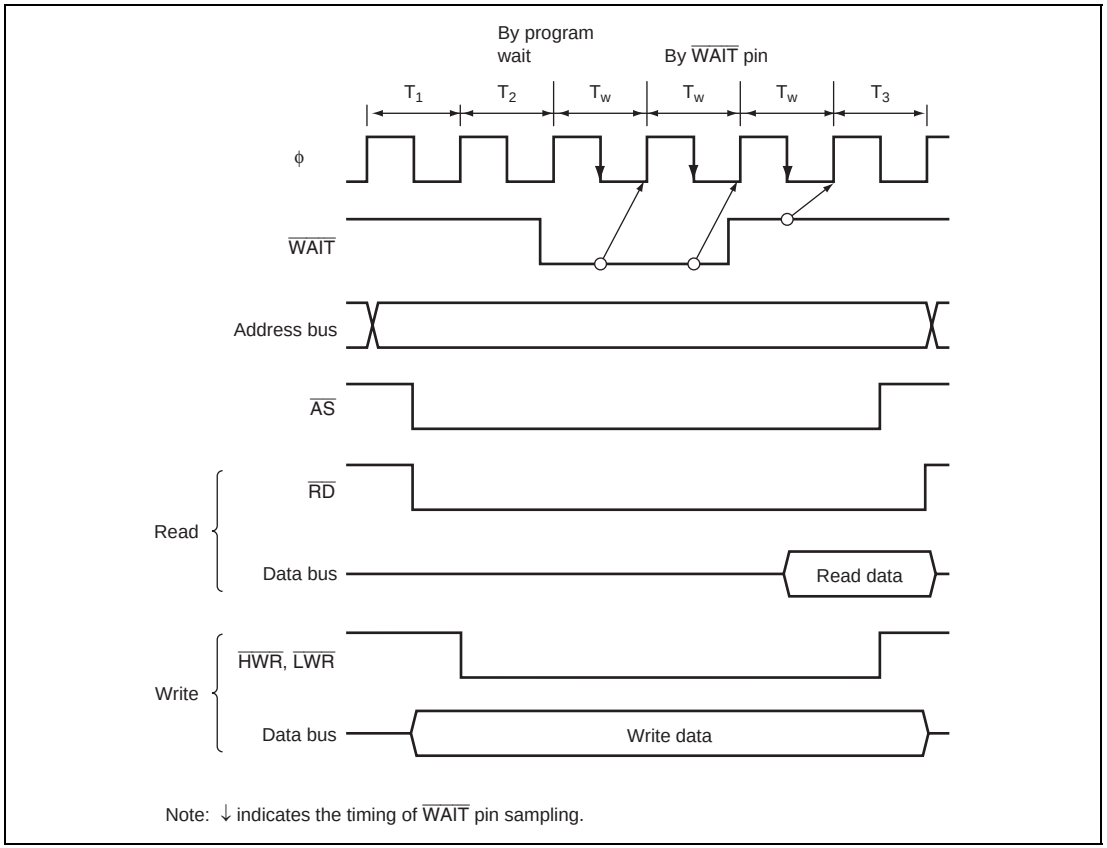



**Figure 6.19 Example of Wait State Insertion Timing**

## 6.7 Burst ROM Interface

With the H8S/2218 Group, external space area 0 can be designated as burst ROM space, and burst ROM interfacing can be performed. The burst ROM space interface enables 16-bit configuration ROM with burst access capability to be accessed at high speed.

Area 0 can be designated as burst ROM space by means of the BRSTRM bit in BCRH.

Consecutive burst accesses of a maximum of 4 words or 8 words can be performed for CPU instruction fetches only. One or two states can be selected for burst access.

### 6.7.1 Basic Timing

The number of states in the initial cycle (full access) of the burst ROM interface is in accordance with the setting of the AST0 bit in ASTCR. Also, when the AST0 bit is set to 1, wait state insertion is possible. One or two states can be selected for the burst cycle, according to the setting of the BRSTS1 bit in BCRH. Wait states cannot be inserted. When area 0 is designated as burst ROM space, it becomes 16-bit access space regardless of the setting of the ABW0 bit in ABWCR.

When the BRSTS0 bit in BCRH is cleared to 0, burst access of up to 4 words is performed; when the BRSTS0 bit is set to 1, burst access of up to 8 words is performed.

The basic access timing for burst ROM space is shown in figures 6.20 and 6.21. The timing shown in figure 6.20 is for the case where the AST0 and BRSTS1 bits are both set to 1, and that in figure 6.21 is for the case where both these bits are cleared to 0.

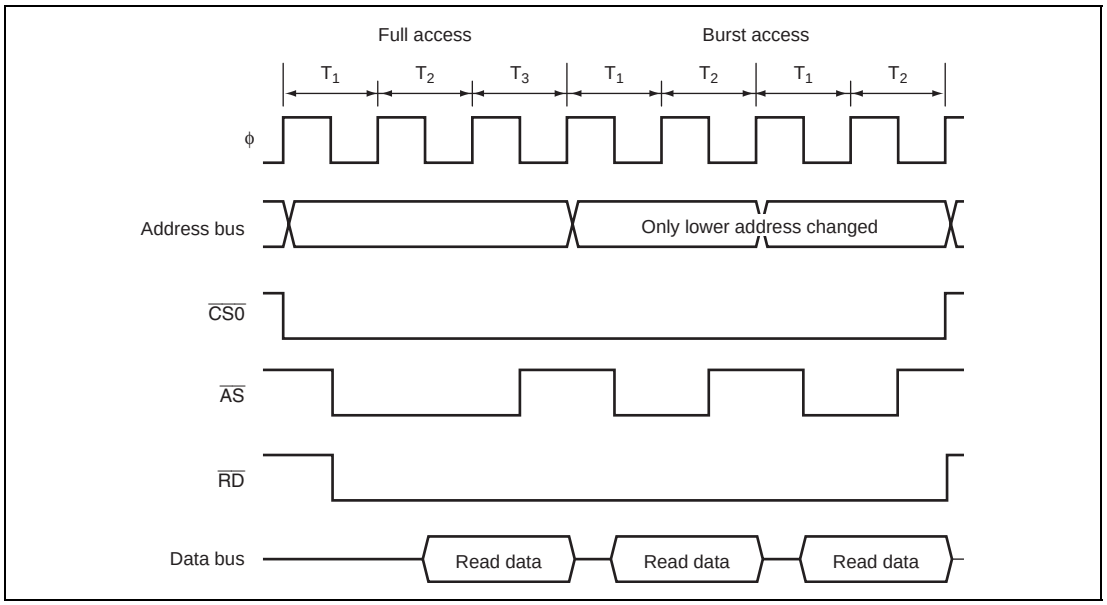



**Figure 6.20 Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 1)**

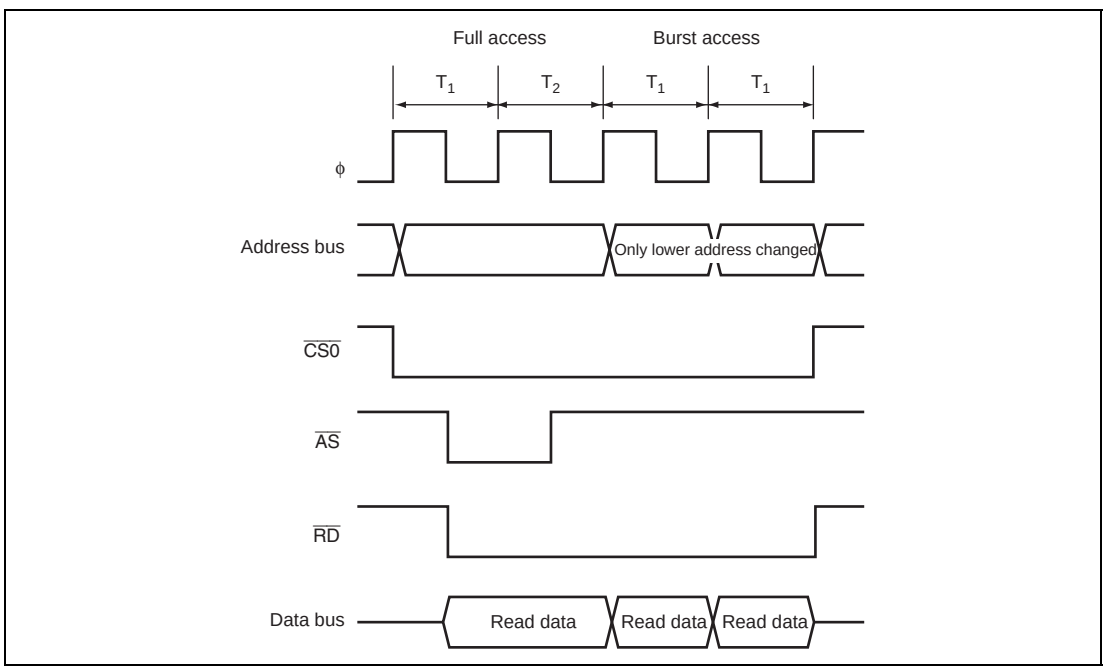



**Figure 6.21 Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 0)**

### 6.7.2 Wait Control

As with the basic bus interface, either program wait insertion or pin wait insertion using the $\overline{\text{WAIT}}$ pin can be used in the initial cycle (full access) of the burst ROM interface. See section 6.6.4, Wait Control.

Wait states cannot be inserted in a burst cycle.

## 6.8 Idle Cycle

When the H8S/2218 Group accesses external space, it can insert a 1-state idle cycle ($T_I$) between bus cycles in the following two cases: (1) when read accesses between different areas occur consecutively, and (2) when a write cycle occurs immediately after a read cycle. By inserting an idle cycle it is possible, for example, to avoid data collisions between ROM, with a long output floating time, and high-speed memory, I/O interfaces, and so on.

**Consecutive Reads between Different Areas:** If consecutive reads between different areas occur while the ICIS1 bit in BCRH is set to 1, an idle cycle is inserted at the start of the second read cycle.

Figure 6.22 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a read cycle from SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data collision is prevented.

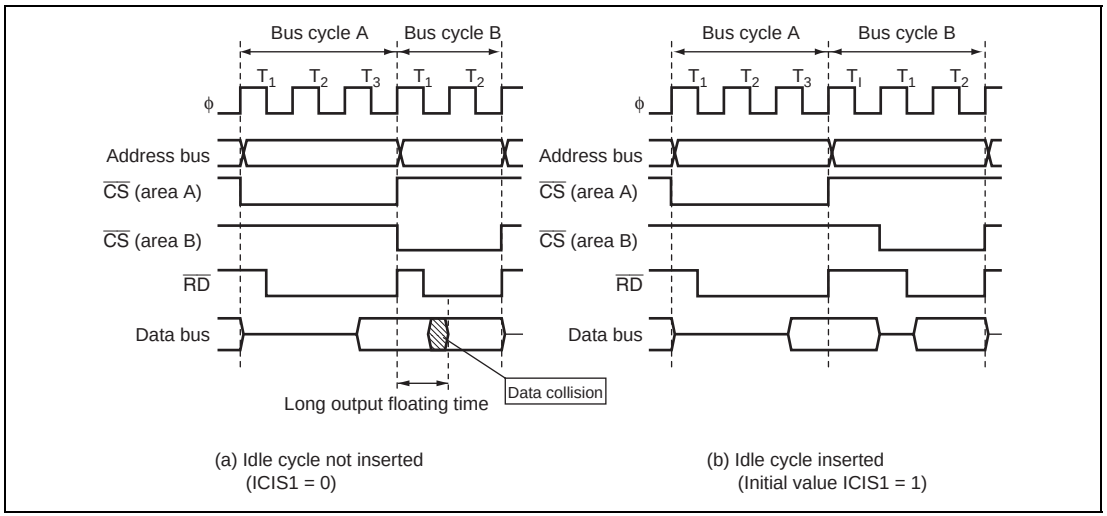



**Figure 6.22 Example of Idle Cycle Operation (1)**

**Write after Read:** If an external write occurs after an external read while the ICIS0 bit in BCRH is set to 1, an idle cycle is inserted at the start of the write cycle.

Figure 6.23 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data collision is prevented.

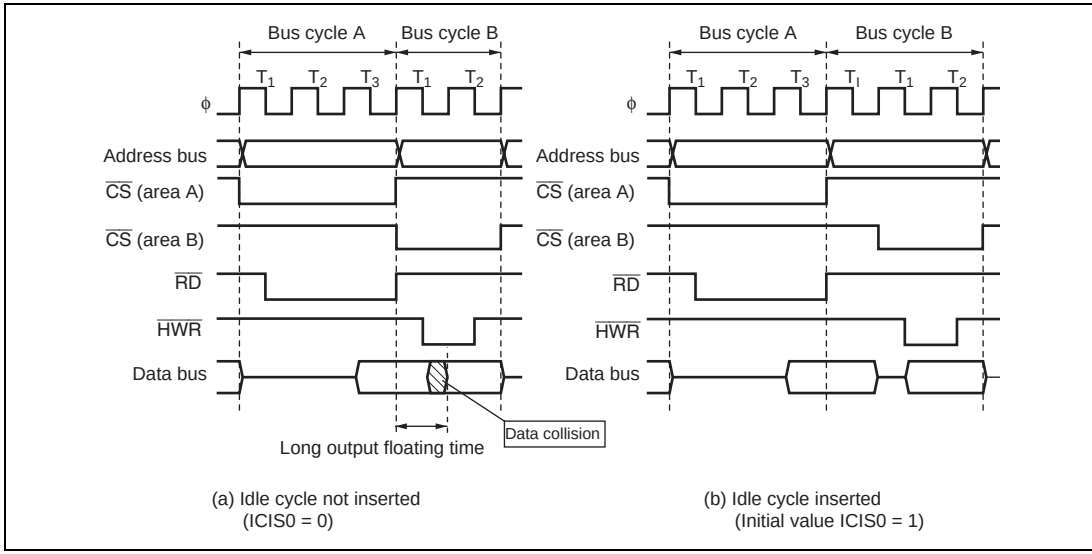



**Figure 6.23 Example of Idle Cycle Operation (2)**

**Relationship between Chip Select ($\overline{CS}$) Signal and Read ($\overline{RD}$) Signal:** Depending on the system's load conditions, the $\overline{RD}$ signal may lag behind the $\overline{CS}$ signal. An example is shown in figure 6.24.

In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the bus cycle A $\overline{RD}$ signal and the bus cycle B $\overline{CS}$ signal.

Setting idle cycle insertion, as in (b), however, will prevent any overlap between the $\overline{RD}$ and $\overline{CS}$ signals.

In the initial state after reset release, idle cycle insertion (b) is set.

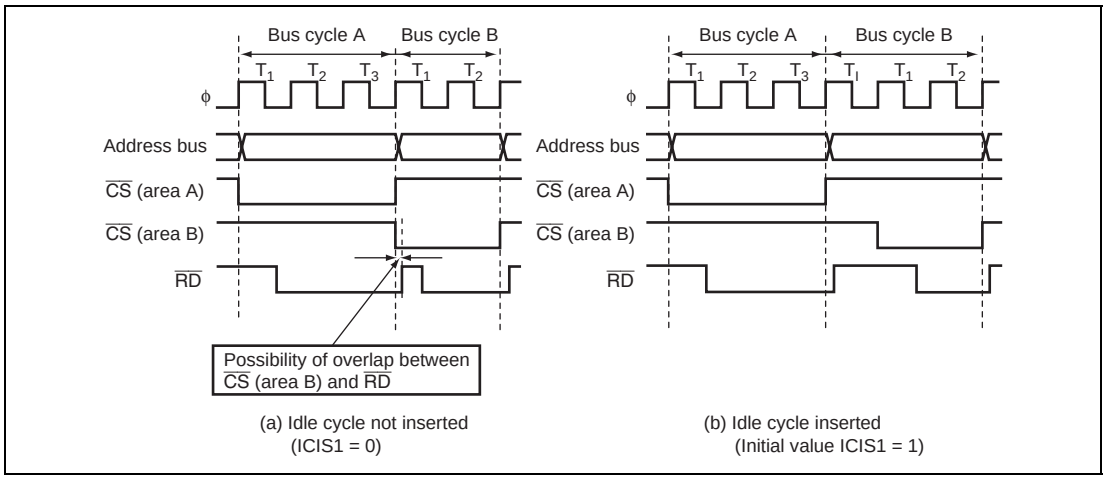



**Figure 6.24 Relationship between Chip Select ($\overline{CS}$) and Read ($\overline{RD}$)**

Table 6.4 shows pin states in an idle cycle.

**Table 6.4 Pin States in Idle Cycle**

| Pins | Pin State |
|---|---|
| A23 to A0 | Contents of next bus cycle |
| D15 to D0 | High impedance |
| $\overline{CSn}$ | High |
| $\overline{AS}$ | High |
| $\overline{RD}$ | High |
| $\overline{HWR}$ | High |
| $\overline{LWR}$ | High |

## 6.9 Bus Release

The H8S/2218 Group can release the external bus in response to a bus request from an external device. In the external bus released state, the internal bus master continues to operate as long as there is no external access.

In external extended mode, the bus can be released to an external device by setting the BRLE bit in BCRL to 1. Driving the $\overline{\text{BREQ}}$ pin low issues an external bus request to this LSI. When the $\overline{\text{BREQ}}$ pin is sampled, at the prescribed timing the $\overline{\text{BACK}}$ pin is driven low, and the address bus, data bus, and bus control signals are placed in the high-impedance state, establishing the external bus-released state.

In the external bus released state, an internal bus master can perform accesses using the internal bus. When an internal bus master wants to make an external access, it temporarily defers activation of the bus cycle, and waits for the bus request from the external bus master to be dropped.

When the $\overline{\text{BREQ}}$ pin is driven high, the $\overline{\text{BACK}}$ pin is driven high at the prescribed timing and the external bus released state is terminated.

In the event of simultaneous external bus release request and external access request generation, the order of priority is as follows:

(High) External bus release > Internal bus master external access (Low)

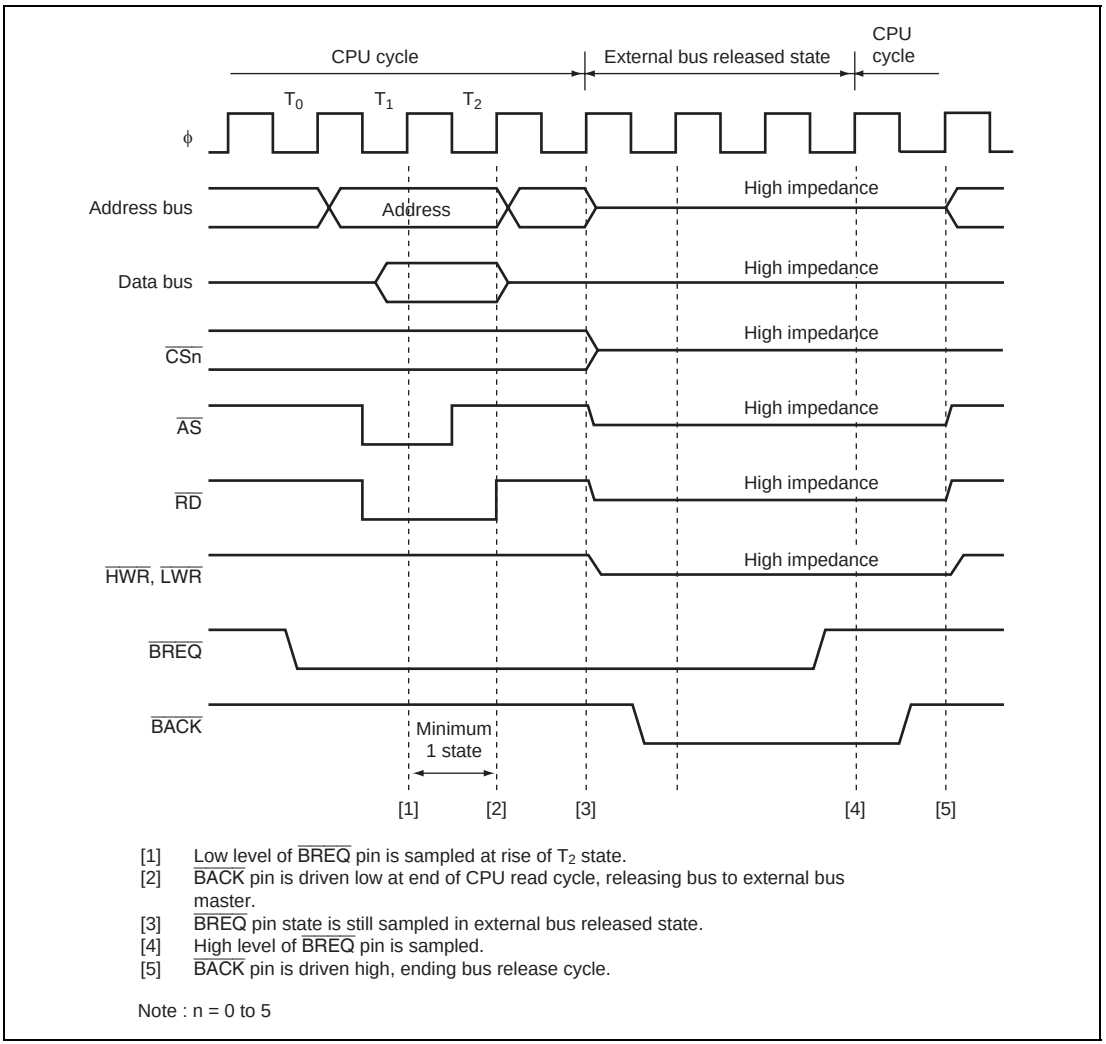
Table 6.5 shows pin states in the external bus released state.

In the H8S/2212 Group, the BRLE bit in BCRL should not be set to 1.

**Table 6.5 Pin States in Bus Released State**

| Pins | Pin State |
|---|---|
| A23 to A0 | High impedance |
| D15 to D0 | High impedance |
| $\overline{\text{CSn}}$ | High impedance |
| $\overline{\text{AS}}$ | High impedance |
| $\overline{\text{RD}}$ | High impedance |
| $\overline{\text{HWR}}$ | High impedance |
| $\overline{\text{LWR}}$ | High impedance |

Figure 6.25 shows the timing for transition to the bus-released state.




[1] Low level of $\overline{\text{BREQ}}$ pin is sampled at rise of $T_2$ state.
[2] $\overline{\text{BACK}}$ pin is driven low at end of CPU read cycle, releasing bus to external bus master.
[3] $\overline{\text{BREQ}}$ pin state is still sampled in external bus released state.
[4] High level of $\overline{\text{BREQ}}$ pin is sampled.
[5] $\overline{\text{BACK}}$ pin is driven high, ending bus release cycle.

Note : n = 0 to 5

**Figure 6.25 Bus-Released State Transition Timing**

### 6.9.1 Bus Release Usage Note

When MSTPCR is set to H'FFFFFF and transmitted to sleep mode, the external bus release does not function. To activate the external bus release in sleep mode, do not set MSTPCR to H'FFFFFF.

## 6.10 Bus Arbitration

This LSI has a bus arbiter that arbitrates bus master operations.

There are two bus masters, the CPU and DMAC, which perform read/write operations when they have possession of the bus. Each bus master requests the bus by means of a bus request signal. The bus arbiter determines priorities at the prescribed timing, and permits use of the bus by means of a bus request acknowledge signal. The selected bus master then takes possession of the bus and begins its operation.

### 6.10.1 Operation

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

The order of priority of the bus masters is as follows:

(High) DMAC > CPU (Low)

An internal bus access by an internal bus master, and external bus release, can be executed in parallel in the H8S/2218 Group.

In the event of simultaneous external bus release request, and internal bus master external access request generation, the order of priority is as follows:

(High) External bus release > Internal bus master external access (Low)

The H8S/2212 Group does not have the external bus release function.

### 6.10.2 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority than that of the bus master that has acquired the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific times at which each bus master can relinquish the bus.

**CPU:** The CPU is the lowest-priority bus master, and if a bus request is received from the DMAC, the bus arbiter transfers the bus to the bus master that issued the request. The timing for transfer of the bus is as follows:

- The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the operations.
- If the CPU is in sleep mode, it transfers the bus immediately.

**DMAC:** The DMAC sends the bus arbiter a request for the bus when an activation request is generated.

In the case of a USB request in short address mode or normal mode, and in cycle steal mode, the DMAC releases the bus after a single transfer.

In block transfer mode, it releases the bus after transfer of one block, and in burst mode, after completion of the transfer.

### 6.10.3 External Bus Release Usage Note

External bus release can be performed on completion of an external bus cycle in the H8S/2218 Group. The $\overline{CS}$ signal remains low until the end of the external bus cycle. Therefore, when external bus release is performed, the $\overline{CS}$ signal may change from the low level to the high-impedance state.

## 6.11 Resets and the Bus Controller

In a power-on reset, this LSI, including the bus controller, enters the reset state at that point, and an executing bus cycle is discontinued.

In a manual reset*, the bus controller's registers and internal state are maintained, and an executing external bus cycle is completed. In this case, $\overline{WAIT}$ input is ignored and write data is not guaranteed.

Note: * Supported only by the H8S/2218 Group.

# Section 7 DMA Controller (DMAC)

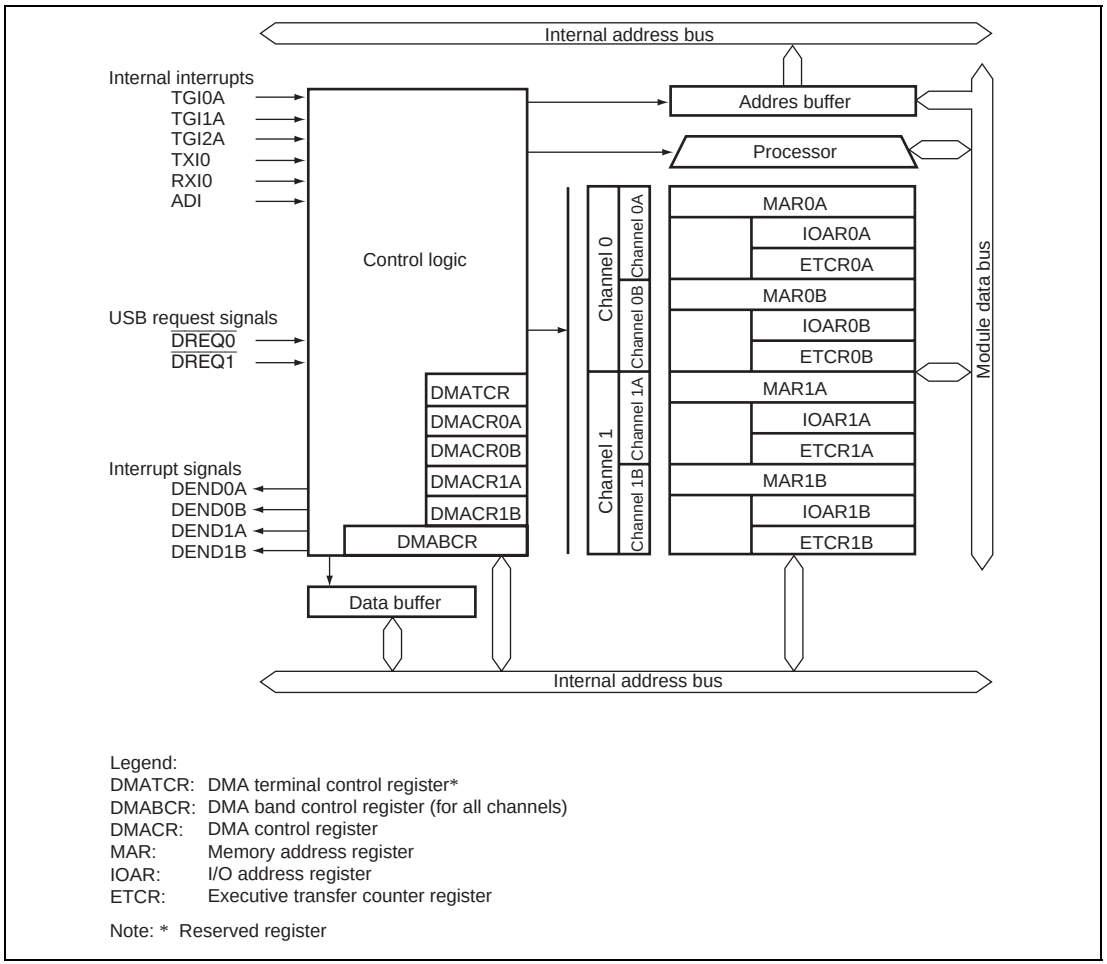
This LSI has a built-in DMA controller (DMAC) which can carry out data transfer on up to 4 channels.

## 7.1 Features

The features of the DMAC are listed below.

- Choice of short address mode or full address mode
  (1) Short address mode
    — Maximum of 4 channels can be used
    — Choice of dual address mode
    — In dual address mode, one of the two addresses, transfer source and transfer destination, is specified as 24 bits and the other as16 bits
    — Choice of sequential mode, idle mode, or repeat mode for dual address mode
  (2) Full address mode
    — Maximum of 2 channels can be used
    — Transfer source and transfer destination address specified as 24 bits
    — Choice of normal mode or block transfer mode
- 16-Mbyte address space can be specified directly
- Byte or word can be set as the transfer unit
- Activation sources: internal interrupt, USB request, auto-request (depending on transfer mode)
  — 16-bit timer-pulse unit (TPU) compare match/input capture interrupts
  — Serial communication interface (SCI_0) transmission complete interrupt, reception complete interrupt
  — A/D conversion end Interrupt
  — USB request
  — Auto-request
- Module stop mode can be set

A block diagram of the DMAC is shown in figure 7.1.




Legend:
DMATCR: DMA terminal control register*
DMABCR: DMA band control register (for all channels)
DMACR: DMA control register
MAR: Memory address register
IOAR: I/O address register
ETCR: Executive transfer counter register

Note: * Reserved register

**Figure 7.1 Block Diagram of DMAC**

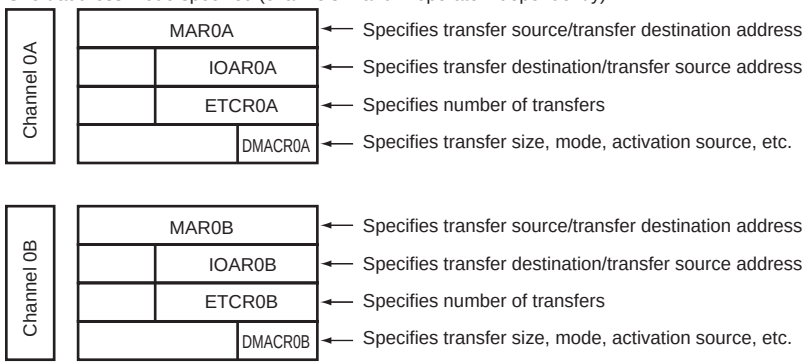
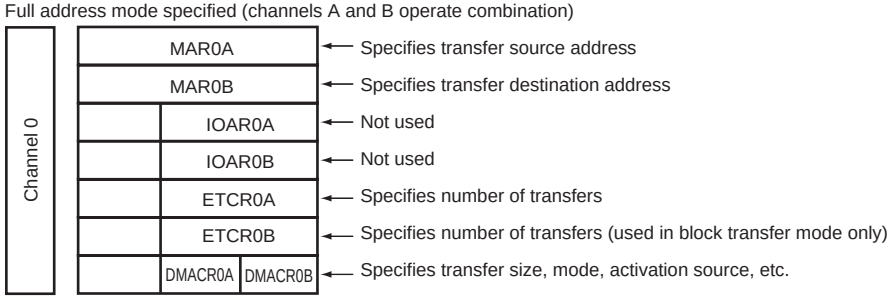
## 7.2 Register Configuration

The DMAC registers are listed below.

- Memory address register 0A (MAR0A)
- I/O address register 0A (IOAR0A)
- Transfer count register 0A (ETCR0A)
- Memory address register 0B (MAR0B)
- I/O address register 0B (IOAR0B)
- Transfer count register 0B (ETCR0B)
- Memory address register 1A (MAR1A)
- I/O address register 1A (IOAR1A)
- Transfer count register 1A (ETCR1A)
- Memory address register 1B (MAR1B)
- I/O address register 1B (IOAR1B)
- Transfer count register 1B (ETCR1B)
- DMA control register 0A (DMACR0A)
- DMA control register 0B (DMACR0B)
- DMA control register 1A (DMACR1A)
- DMA control register 1B (DMACR1B)
- DMA band control register (DMABCR)

The DMAC register functions differs depending on the address modes: short address mode and full address mode. The DMAC register functions are described in each address mode. Short address mode or full address mode can be selected for channels 1 and 0 independently by means of bits FAE1 and FAE0.

**Table 7.1 Short Address Mode and Full Address Mode (For 1 Channel: Example of Channel 0)**

| FAE0 | Description |
|---|---|
| 0 | Short address mode specified (channels A and B operate independently)<br>  |
| 1 | Full address mode specified (channels A and B operate combination)<br>  |

## 7.3 Register Descriptions

### 7.3.1 Memory Address Registers (MAR)

- Short Address Mode

  MAR is a 32-bit readable/writable register that specifies the transfer source address or destination address. The upper 8 bits of MAR are reserved: they are always read as 0, and cannot be modified. Whether MAR functions as the source address register or as the destination address register can be selected by means of the DTDIR bit in DMACR.

  MAR is incremented or decremented each time a byte or word transfer is executed, so that the address specified by MAR is constantly updated. For details, see section 7.3.4, DMA Control Register (DMACR). MAR is not initialized by a reset or in standby mode.

- Full Address Mode

  MAR is a 32-bit readable/writable register; MARA functions as the transfer source address register, and MARB as the destination address register.

  MAR is composed of two 16-bit registers, MARH and MARL. The upper 8 bits of MARH are reserved: they are always read as 0, and cannot be modified. MAR is incremented or decremented each time a byte or word transfer is executed, so that the source or destination memory address can be updated automatically. For details, see section 7.3.4, DMA Control Register (DMACR). MAR is not initialized by a reset or in standby mode.

### 7.3.2 I/O Address Register (IOAR)

- Short Address Mode

  IOAR is a 16-bit readable/writable register that specifies the lower 16 bits of the transfer source address or destination address. The upper 8 bits of the transfer address are automatically set to H'FF. Whether IOAR functions as the source address register or as the destination address register can be selected by means of the DTDIR bit in DMACR.

  IOAR is not incremented or decremented each time a transfer is executed, so that the address specified by IOAR is fixed. IOAR is not initialized by a reset or in standby mode.

- Full Address Mode:

  IOAR is not used in full address mode transfer.

### 7.3.3 Execute Transfer Count Register (ETCR)

- Short Address Mode

  ETCR is a 16-bit readable/writable register that specifies the number of transfers. The setting of this register is different for sequential mode and idle mode on the one hand, and for repeat mode on the other. ETCR is not initialized by a reset or in standby mode.

  — Sequential Mode and Idle Mode

  In sequential mode and idle mode, ETCR functions as a 16-bit transfer counter (with a count range of 1 to 65,536). ETCR is decremented by 1 each time a transfer is performed, and when the count reaches H'0000, the DTE bit in DMABCR is cleared, and transfer ends.

  — Repeat Mode

  In repeat mode, ETCR functions as an 8-bit transfer counter ETCRL (with a count range of 1 to 256) and transfer number storage register ETCRH. ETCRL is decremented by 1 each time a transfer is performed, and when the count reaches H'00, ETCRL is loaded with the value in ETCRH. At this point, MAR is automatically restored to the value it had when the count was started. The DTE bit in DMABCR is not cleared, and so transfers can be performed repeatedly until the DTE bit is cleared by the user.

- Full Address Mode

  ETCR is a 16-bit readable/writable register that specifies the number of transfers. The function of this register is different in normal mode and in block transfer mode. ETCR is not initialized by a reset or in standby mode.

  — Normal Mode

  (a) ETCRA

  In normal mode, ETCRA functions as a 16-bit transfer counter. ETCRA is decremented by 1 each time a transfer is performed, and transfer ends when the count reaches H'0000.

  (b) ETCRB

  ETCRB is not used in normal mode.

  — Block Transfer Mode

  (a) ETCRA

  In block transfer mode, ETCRAL functions as an 8-bit block size counter and ETCRAH holds the block size. ETCRAL is decremented each time a 1-byte or 1-word transfer is performed, and when the count reaches H'00, ETCRAL is loaded with the value in ETCRAH. So by setting the block size in ETCRAH and ETCRAL, it is possible to repeatedly transfer blocks consisting of any desired number of bytes or words.

  (b) ETCRB

  ETCRB functions in block transfer mode, as a 16-bit block transfer counter. ETCRB is decremented by 1 each time a block is transferred, and transfer ends when the count reaches H'0000.

### 7.3.4 DMA Control Register (DMACR)

DMACR controls the operation of each DMAC channel.

- Short Address Mode (common to DMACRA and DMACRB)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | DTSZ | 0 | R/W | Data Transfer Size<br>Selects the size of data to be transferred at one time.<br>0: Byte-size transfer<br>1: Word-size transfer |
| 6 | DTID | 0 | R/W | Data Transfer Increment/Decrement<br>Selects incrementing or decrementing of MAR every data transfer in sequential mode or repeat mode.<br>In idle mode, MAR is neither incremented nor decremented.<br>0: MAR is incremented after a data transfer<br>• When DTSZ = 0, MAR is incremented by 1 after a transfer<br>• When DTSZ = 1, MAR is incremented by 2 after a transfer<br>1: MAR is decremented after a data transfer<br>• When DTSZ = 0, MAR is decremented by 1 after a transfer<br>• When DTSZ = 1, MAR is decremented by 2 after a transfer |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 5 | RPE | 0 | R/W | Repeat Enable<br>Used in combination with the DTIE bit in DMABCR to select the mode (sequential, idle, or repeat) in which transfer is to be performed.<br>RPE DTIE<br>0 0: Transfer in sequential mode (no transfer end interrupt)<br>0 1: Transfer in sequential mode (with transfer end interrupt)<br>1 0: Transfer in repeat mode (no transfer end interrupt)<br>1 1: Transfer in idle mode (with transfer end interrupt)<br>Note: For details of operation in sequential, idle, and repeat mode, see section 7.4.2, Sequential Mode, section 7.4.3, Idle Mode, and section 7.4.4, Repeat Mode. |
| 4 | DTDIR | 0 | R/W | Data Transfer Direction<br>Specifies the data transfer direction (source or destination).<br>0: Transfer with MAR as source address and IOAR as destination address<br>1: Transfer with IOAR as source address and MAR as destination address |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | DTF3 | 0 | R/W | Data Transfer Factor |
| 2 | DTF2 | 0 | R/W | These bits select the data transfer factor (activation source). |
| 1 | DTF1 | 0 | R/W | 0000: — |
| 0 | DTF0 | 0 | R/W | 0001: Activated by A/D conversion end interrupt |
| | | | | 0010: — |
| | | | | 0011: — |
| | | | | 0100: Activated by SCI channel 0 transmission complete interrupt |
| | | | | 0101: Activated by SCI channel 0 reception complete interrupt |
| | | | | 0110: — |
| | | | | 0111: — |
| | | | | 1000: Activated by TPU channel 0 compare match/input capture A interrupt |
| | | | | 1001: Activated by TPU channel 1 compare match/input capture A interrupt |
| | | | | 1010: Activated by TPU channel 2 compare match/input capture A interrupt |
| | | | | 1011: — |
| | | | | 1100: — |
| | | | | 1101: — |
| | | | | 1110: — |
| | | | | 1111: — |

The same factor can be selected for more than one channel. In this case, activation starts with the highest-priority channel according to the relative channel priorities. For relative channel priorities, see section 7.4.10, DMAC Multi-Channel Operation.

- Full Address Mode (DMACRA)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 15 | DTSZ | 0 | R/W | Data Transfer Size<br>Selects the size of data to be transferred at one time.<br>0: Byte-size transfer<br>1: Word-size transfer |
| 14 | SAID | 0 | R/W | Source Address Increment/Decrement |
| 13 | SAIDE | 0 | R/W | Source Address Increment/Decrement Enable<br>These bits specify whether source address register MARA is to be incremented, decremented, or left unchanged, when data transfer is performed.<br>00: MARA is fixed<br>01: MARA is incremented after a data transfer<br>• When DTSZ = 0, MARA is incremented by 1 after a transfer<br>• When DTSZ = 1, MARA is incremented by 2 after a transfer<br>10: MARA is fixed<br>11: MARA is decremented after a data transfer<br>• When DTSZ = 0, MARA is decremented by 1 after a transfer<br>• When DTSZ = 1, MARA is decremented by 2 after a transfer |
| 12 | BLKDIR | 0 | R/W | Block Direction |
| 11 | BLKE | 0 | R/W | Block Enable<br>These bits specify whether normal mode or block transfer mode is to be used. If block transfer mode is specified, the BLKDIR bit specifies whether the source side or the destination side is to be the block area.<br>00: Transfer in normal mode<br>01: Transfer in block transfer mode, destination side is block area<br>10: Transfer in normal mode<br>11: Transfer in block transfer mode, source side is block area<br>For operation in normal mode and block transfer mode, see section 7.4, Operation. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 10 to 8 | — | All 0 | R/W | Reserved<br>Although these bits are readable/writable, only 0 should be written to. |

- Full Address Mode (DMACRB)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | 0 | R/W | Reserved<br>Although this bit is readable/writable, only 0 should be written to. |
| 6 | DAID | 0 | R/W | Destination Address Increment/Decrement |
| 5 | DAIDE | 0 | R/W | Destination Address Increment/Decrement Enable<br>These bits specify whether destination address register MARB is to be incremented, decremented, or left unchanged, when data transfer is performed.<br>00: MARB is fixed<br>01: MARB is incremented after a data transfer<br>• When DTSZ = 0, MARB is incremented by 1 after a transfer<br>• When DTSZ = 1, MARB is incremented by 2 after a transfer<br>10: MARB is fixed<br>11: MARB is decremented after a data transfer<br>• When DTSZ = 0, MARB is decremented by 1 after a transfer<br>• When DTSZ = 1, MARB is decremented by 2 after a transfer |
| 4 | — | 0 | R/W | Reserved<br>Although this bit is readable/writable, only 0 should be written to. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | DTF3 | 0 | R/W | Data Transfer Factor |
| 2 | DTF2 | 0 | R/W | These bits select the data transfer factor (activation source). |
| 1 | DTF1 | 0 | R/W | In normal mode: |
| 0 | DTF0 | 0 | R/W | 0000: —<br>0001: —<br>0010: —<br>0011: Activated by $\overline{\text{DREQ}}$ signal's low level input from USB (USB request)<br>010×: —<br>0110: Auto-request (cycle steal)<br>0111: Auto-request (burst)<br>1×××: —<br>In block transfer mode:<br>0000: —<br>0001: Activated by A/D conversion end interrupt<br>0010: —<br>0011: —<br>0100: Activated by SCI channel 0 transmission complete interrupt<br>0101: Activated by SCI channel 0 reception complete interrupt<br>0110: —<br>0111: —<br>1000: Activated by TPU channel 0 compare match/input capture A interrupt<br>1001: Activated by TPU channel 1 compare match/input capture A interrupt<br>1010: Activated by TPU channel 2 compare match/input capture A interrupt<br>1011: —<br>11××: —<br>The same factor can be selected for more than one channel. In this case, activation starts with the highest-priority channel according to the relative channel priorities. For relative channel priorities, see section 7.4.10, DMAC Multi-Channel Operation.<br>Legend:<br>×: Don't care |

### 7.3.5 DMA Band Control Register (DMABCR)

DMABCR controls the operation of each DMAC channel.

- Short Address Mode

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 15 | FAE1 | 0 | R/W | Full Address Enable 1<br>Specifies whether channel 1 is to be used in short address mode or full address mode.<br>In short address mode, channels 1A and 1B are used as independent channels.<br>0: Short address mode<br>1: Full address mode |
| 14 | FAE0 | 0 | R/W | Full Address Enable 0<br>Specifies whether channel 0 is to be used in short address mode or full address mode.<br>In short address mode, channels 0A and 0B are used as independent channels.<br>0: Short address mode<br>1: Full address mode |
| 13, 12 | — | — | R/W | Reserved<br>The write value should always be 0. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 11 | DTA1B | 0 | R/W | Data Transfer Acknowledge |
| 10 | DTA1A | 0 | R/W | These bits enable or disable clearing, when DMA transfer is performed, of the internal interrupt source selected by the data transfer factor setting.<br>When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting is cleared automatically by DMA transfer. When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting does not issue an interrupt request to the CPU.<br>When DTE = 1 and DTA = 0, the internal interrupt source selected by the data transfer factor setting is not cleared when a transfer is performed, and can issue an interrupt request to the CPU in parallel. In this case, the interrupt source should be cleared by the CPU.<br>When DTE = 0, the internal interrupt source selected by the data transfer factor setting issues an interrupt request to the CPU regardless of the DTA bit setting.<br>0: Clearing of selected internal interrupt source at time of DMA transfer is disabled<br>1: Clearing of selected internal interrupt source at time of DMA transfer is enabled |
| 9 | DTA0B | 0 | R/W | |
| 8 | DTA0A | 0 | R/W | |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | DTE1B | 0 | R/W | Data Transfer Enable |
| 6 | DTE1A | 0 | R/W | When DTE = 0, data transfer is disabled and the activation source selected by the data transfer factor setting is ignored. If the activation source is an internal interrupt, an interrupt request is issued to the CPU. If the DTIE bit is set to 1when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU.<br>The conditions for the DTE bit being cleared to 0 are as follows:<br>• When initialization is performed<br>• When the specified number of transfers have been completed in a transfer mode other than repeat mode<br>• When 0 is written to the DTE bit to forcibly abort the transfer, or for a similar reason<br>When DTE = 1, data transfer is enabled and the DMAC waits for a request by the activation source selected by the data transfer factor setting. When a request is issued by the activation source, DMA transfer is executed. The condition for the DTE bit being set to 1 is as follows:<br>• When 1 is written to the DTE bit after the DTE bit is read as 0<br>0: Data transfer disabled<br>1: Data transfer enabled |
| 5 | DTE0B | 0 | R/W | |
| 4 | DTE0A | 0 | R/W | |
| 3 | DTIE1B | 0 | R/W | Data Transfer End Interrupt Enable |
| 2 | DTIE1A | 0 | R/W | These bits enable or disable an interrupt to the CPU when transfer ends. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU.<br>A transfer end interrupt can be canceled either by clearing the DTIE bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the transfer counter and address register again, and then setting the DTE bit to 1.<br>0: Transfer end interrupt disabled<br>1: Transfer end interrupt enabled |
| 1 | DTIE0B | 0 | R/W | |
| 0 | DTIE0A | 0 | R/W | |

- Full Address Mode

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 15 | FAE1 | 0 | R/W | Full Address Enable 1<br>Specifies whether channel 1 is to be used in short address mode or full address mode.<br>In full address mode, channels 1A and 1B are used together as a single channel.<br>0: Short address mode<br>1: Full address mode |
| 14 | FAE0 | 0 | R/W | Full Address Enable 0<br>Specifies whether channel 0 is to be used in short address mode or full address mode.<br>In full address mode, channels 0A and 0B are used together as a single channel.<br>0: Short address mode<br>1: Full address mode |
| 13,12 | — | All 0 | R/W | Reserved<br>Although these bits are readable/writable, only 0 should be written to. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| | | | | Data Transfer Acknowledge<br>Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the data transfer factor setting.<br>When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting is cleared automatically by DMA transfer. When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting does not issue an interrupt request to the CPU.<br>When DTE = 1 and DTA = 0, the internal interrupt source selected by the data transfer factor setting is not cleared when a transfer is performed, and can issue an interrupt request to the CPU in parallel. In this case, the interrupt source should be cleared by the CPU transfer.<br>When DTE = 0, the internal interrupt source selected by the data transfer factor setting issues an interrupt request to the CPU regardless of the DTA bit setting.<br>The state of the DTME bit does not affect the above operations. |
| 11 | DTA1 | 0 | R/W | Data transfer acknowledge 1<br>Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1 data transfer factor setting.<br>0: Clearing of selected internal interrupt source at time of DMA transfer is disabled<br>1: Clearing of selected internal interrupt source at time of DMA transfer is enabled |
| 10 | – | 0 | R/W | Reserved<br>Although this bit is readable/writable, only 0 should be written to. |
| 9 | DTA0 | 0 | R/W | Data Transfer Acknowledge 0<br>Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0 data transfer factor setting.<br>0: Clearing of selected internal interrupt source at time of DMA transfer is disabled<br>1: Clearing of selected internal interrupt source at time of DMA transfer is enabled |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 8 | – | 0 | R/W | Reserved<br>Although this bit is readable/writable, only 0 should be written to. |
| | | | | Data Transfer Master Enable<br>Together with the DTE bit, this bit controls enabling or disabling of data transfer on the relevant channel. When both the DTME bit and the DTE bit are set to 1, transfer is enabled for the channel. If the relevant channel is in the middle of a burst mode transfer when an NMI interrupt is generated, the DTME bit is cleared, the transfer is interrupted, and bus mastership passes to the CPU. When the DTME bit is subsequently set to 1 again, the interrupted transfer is resumed. In block transfer mode, however, the DTME bit is not cleared by an NMI interrupt, and transfer is not interrupted.<br>The conditions for the DTME bit being cleared to 0 are as follows:<br>• When initialization is performed<br>• When NMI is input in burst mode<br>• When 0 is written to the DTME bit<br>The condition for DTME being set to 1 is as follows:<br>• When 1 is written to DTME after DTME is read as 0 |
| 7 | DTME1 | 0 | R/W | Data Transfer Master Enable 1<br>Enables or disables data transfer on channel 1<br>0: Data transfer disabled. In burst mode, cleared to 0 by an NMI interrupt<br>1: Data transfer enabled |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| | | | | Data Transfer Enable<br>When DTE = 0, data transfer is disabled and the activation source selected by the data transfer factor setting is ignored. If the activation source is an internal interrupt, an interrupt request is issued to the CPU. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU.<br>The conditions for the DTE bit being cleared to 0 are as follows:<br>• When initialization is performed<br>• When the specified number of transfers have been completed<br>• When 0 is written to the DTE bit to forcibly abort the transfer, or for a similar reason<br>When DTE = 1 and DTME = 1, data transfer is enabled and the DMAC waits for a request by the activation source selected by the data transfer factor setting. When a request is issued by the activation source, DMA transfer is executed.<br>The condition for the DTE bit being set to 1 is as follows:<br>• When 1 is written to the DTE bit after the DTE bit is read as 0 |
| 6 | DTE1 | 0 | R/W | Data Transfer Enable 1<br>Enables or disables data transfer on channel 1.<br>0: Data transfer disabled<br>1: Data transfer enabled |
| 5 | DTME0 | 0 | R/W | Data Transfer Master Enable 0<br>Enables or disables data transfer on channel 0.<br>0: Data transfer disabled. In burst mode, cleared to 0 by an NMI interrupt<br>1: Data transfer enabled |
| 4 | DTE0 | 0 | R/W | Data Transfer Enable 0<br>Enables or disables data transfer on channel 0.<br>0: Data transfer disabled<br>1: Data transfer enabled |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| | | | | Data Transfer Interrupt Enable B<br>Enables or disables an interrupt to the CPU when transfer is interrupted. If the DTIEB bit is set to 1 when DTME = 0, the DMAC regards this as indicating a break in the transfer, and issues a transfer break interrupt request to the CPU. A transfer break interrupt can be canceled either by clearing the DTIEB bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the DTME bit to 1. |
| 3 | DTIE1B | 0 | R/W | Data Transfer Interrupt Enable 1B<br>Enables or disables the channel 1 transfer break interrupt.<br>0: Transfer break interrupt disabled<br>1: Transfer break interrupt enabled |
| | | | | Data Transfer End Interrupt Enable A<br>Enables or disables an interrupt to the CPU when transfer ends. If the DTIEA bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU. A transfer end interrupt can be canceled either by clearing the DTIEA bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the DTE bit to 1. |
| 2 | DTIE1A | 0 | R/W | Data Transfer End Interrupt Enable 1A<br>Enables or disables the channel 1 transfer end interrupt.<br>0: Transfer end interrupt disabled<br>1: Transfer end interrupt enabled |
| 1 | DTIE0B | 0 | R/W | Data Transfer Interrupt Enable 0B<br>Enables or disables the channel 0 transfer break interrupt.<br>0: Transfer break interrupt disabled<br>1: Transfer break interrupt enabled |
| 0 | DTIE0A | 0 | R/W | Data Transfer End Interrupt Enable 0A<br>Enables or disables the channel 0 transfer end interrupt.<br>0: Transfer end interrupt disabled<br>1: Transfer end interrupt enabled |

## 7.4 Operation

### 7.4.1 Transfer Modes

Table 7.2 lists the DMAC modes.

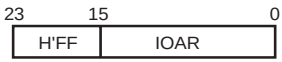
**Table 7.2 DMAC Transfer Modes**

| Transfer Mode | | | Transfer Source | Remarks |
|---|---|---|---|---|
| Short address mode | Dual address mode | (1) Sequential mode<br>(2) Idle mode<br>(3) Repeat Mode | • TPU channel 0 to 2 compare match/input capture A interrupt<br>• SCI transmission complete interrupt<br>• SCI reception complete interrupt<br>• A/D conversion end interrupt | • Up to 4 channels can operate independently |
| Full address mode | | (4) Normal mode | • USB request<br>• Auto-request | • Max. 2-channel operation, combining channels A and B<br>• With auto-request, burst mode transfer or cycle steal transfer can be selected |
| | | (5) Block transfer mode | • TPU channel 0 to 2 compare match/input capture A interrupt<br>• SCI transmission complete interrupt<br>• SCI reception complete interrupt<br>• A/D conversion end interrupt | |

### 7.4.2 Sequential Mode

Sequential mode can be specified by clearing the RPE bit in DMACR to 0. In sequential mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCR. One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR. Table 7.3 summarizes register functions in sequential mode.

**Table 7.3 Register Functions in Sequential Mode**

| | Function | | | |
|---|---|---|---|---|
| **Register** | **DTDIR = 0** | **DTDIR = 1** | **Initial Setting** | **Operation** |
| | Source address register | Destination address register | Start address of transfer destination or transfer source | Incremented/ decremented every transfer |
|  | Destination address register | Source address register | Start address of transfer source or transfer destination | Fixed |
| | Transfer counter | | Number of transfers | Decremented every transfer, transfer ends when count reaches H'0000 |

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is incremented or decremented by 1 or 2 each time a byte or word is transferred. IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF. Figure 7.2 illustrates operation in sequential mode.

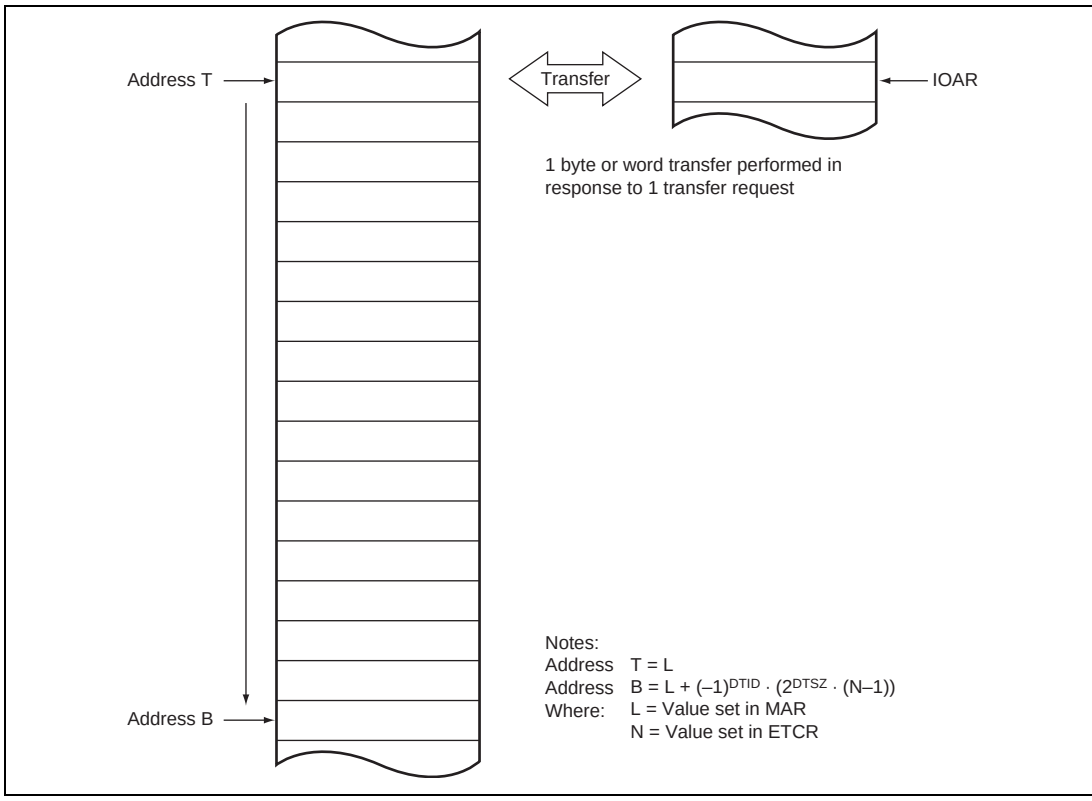
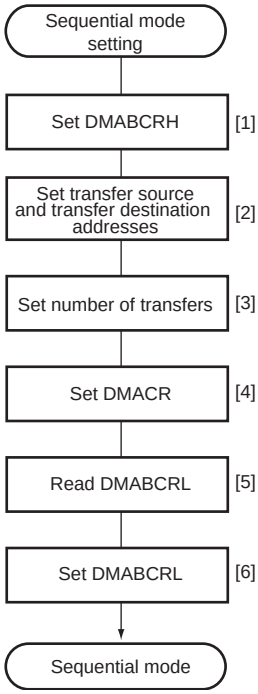



**Figure 7.2 Operation in Sequential Mode**

The number of transfers is specified as 16 bits in ETCR. ETCR is decremented by 1 each time a transfer is executed, and when its value reaches H'0000, the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU. The maximum number of transfers, when H'0000 is set in ETCR, is 65,536. Transfer requests (activation sources) consist of A/D conversion end interrupt, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 2 compare match/input capture A interrupts. Figure 7.3 shows an example of the setting procedure for sequential mode.




[1] Set each bit in DMABCRH.
- Clear the FAE bit to 0 to select short address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address and transfer destination address in MAR and IOAR.

[3] Set the number of transfers in ETCR.

[4] Set each bit in DMACR.
- Set the transfer data size with the DTSZ bit.
- Specify whether MAR is to be incremented or decremented with the DTID bit.
- Clear the RPE bit to 0 to select sequential mode.
- Specify the transfer direction with the DTDIR bit.
- Select the activation source with bits DTF3 to DTF0.

[5] Read the DTE bit in DMABCRL as 0.

[6] Set each bit in DMABCRL.
- Specify enabling or disabling of transfer andinterrupts with the DTIE bit.
- Set the DTE bit to 1 to enable transfer.

**Figure 7.3 Example of Sequential Mode Setting Procedure**

### 7.4.3 Idle Mode

Idle mode can be specified by setting the RPE bit and DTIE bit in DMACR to 1. In idle mode, one byte or word is transferred in response to a single transfer request, and this is executed the number of times specified in ETCR. One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR. Table 7.4 summarizes register functions in idle mode.

**Table 7.4 Register Functions in Idle Mode**

| Register | Function | | Initial Setting | Operation |
|---|---|---|---|---|
| | DTDIR = 0 | DTDIR = 1 | | |
| 23 MAR 0 | Source address register | Destination address register | Start address of transfer destination or transfer source | Fixed |
| 23 H'FF 15 IOAR 0 | Destination address register | Source address register | Start address of transfer source or transfer destination | Fixed |
| 15 ETCR 0 | Transfer counter | | Number of transfers | Decremented every transfer, transfer ends when count reaches H'0000 |

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is neither incremented nor decremented each time a byte or word is transferred. IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF. Figure 7.4 illustrates operation in idle mode.

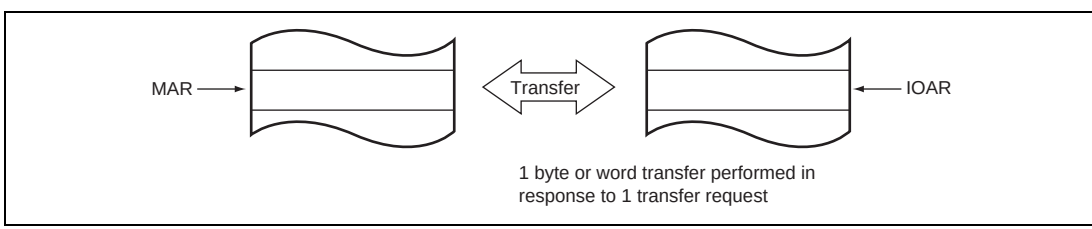



**Figure 7.4 Operation in Idle Mode**

The number of transfers is specified as 16 bits in ETCR. ETCR is decremented by 1 each time a transfer is executed, and when its value reaches H'0000, the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU. The maximum number of transfers, when H'0000 is set in ETCR, is 65,536.

Transfer requests (activation sources) consist of A/D conversion end interrupt, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 2 compare match/input capture A interrupts. Figure 7.5 shows an example of the setting procedure for idle mode.

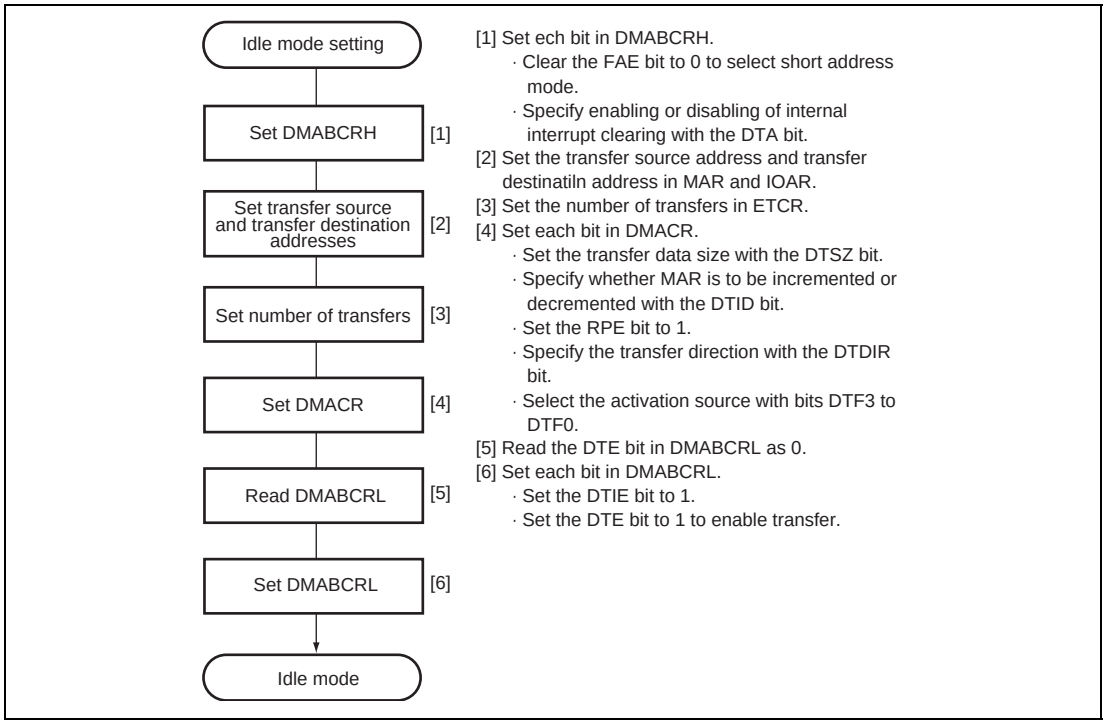
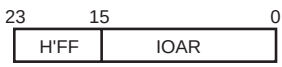
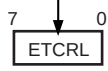



**Figure 7.5 Example of Idle Mode Setting Procedure**

### 7.4.4 Repeat Mode

Repeat mode can be specified by setting the RPE bit in DMACR to 1, and clearing the DTIE bit to 0. In repeat mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCR. On completion of the specified number of transfers, MAR and ETCRL are automatically restored to their original settings and operation continues. One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR. Table 7.5 summarizes register functions in repeat mode.

**Table 7.5 Register Functions in Repeat Mode**

| Register | Function | | Initial Setting | Operation |
|---|---|---|---|---|
| | DTDIR = 0 | DTDIR = 1 | | |
| | Source address register | Destination address register | Start address of transfer destination or transfer source | Incremented/decremented every transfer. Initial setting is restored when value reaches H'0000 |
|  | Destination address register | Source address register | Start address of transfer source or transfer destination | Fixed |
| | Holds number of transfers | | Number of transfers | Fixed |
|  | Transfer counter | | Number of transfers | Decremented every transfer. Loaded with ETCRH value when count reaches H'00 |

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is incremented or decremented by 1 or 2 each time a byte or word is transferred. IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF. The number of transfers is specified as 8 bits by ETCRH and ETCRL. The maximum number of transfers, when H'00 is set in both ETCRH and ETCRL, is 256.

In repeat mode, ETCRL functions as the transfer counter, and ETCRH is used to hold the number of transfers. ETCRL is decremented by 1 each time a transfer is executed, and when its value reaches H'00, it is loaded with the value in ETCRH. At the same time, the value set in MAR is restored in accordance with the values of the DTSZ and DTID bits in DMACR. The MAR restoration operation is as shown below.

$$MAR = MAR - (-1)^{DTID} \cdot 2^{DTSZ} \cdot ETCRH$$

The same value should be set in ETCRH and ETCRL.

In repeat mode, operation continues until the DTE bit is cleared. To end the transfer operation, therefore, you should clear the DTE bit to 0. A transfer end interrupt request is not sent to the CPU. By setting the DTE bit to 1 again after it has been cleared, the operation can be restarted from the transfer after that terminated when the DTE bit was cleared. Figure 7.6 illustrates operation in repeat mode.

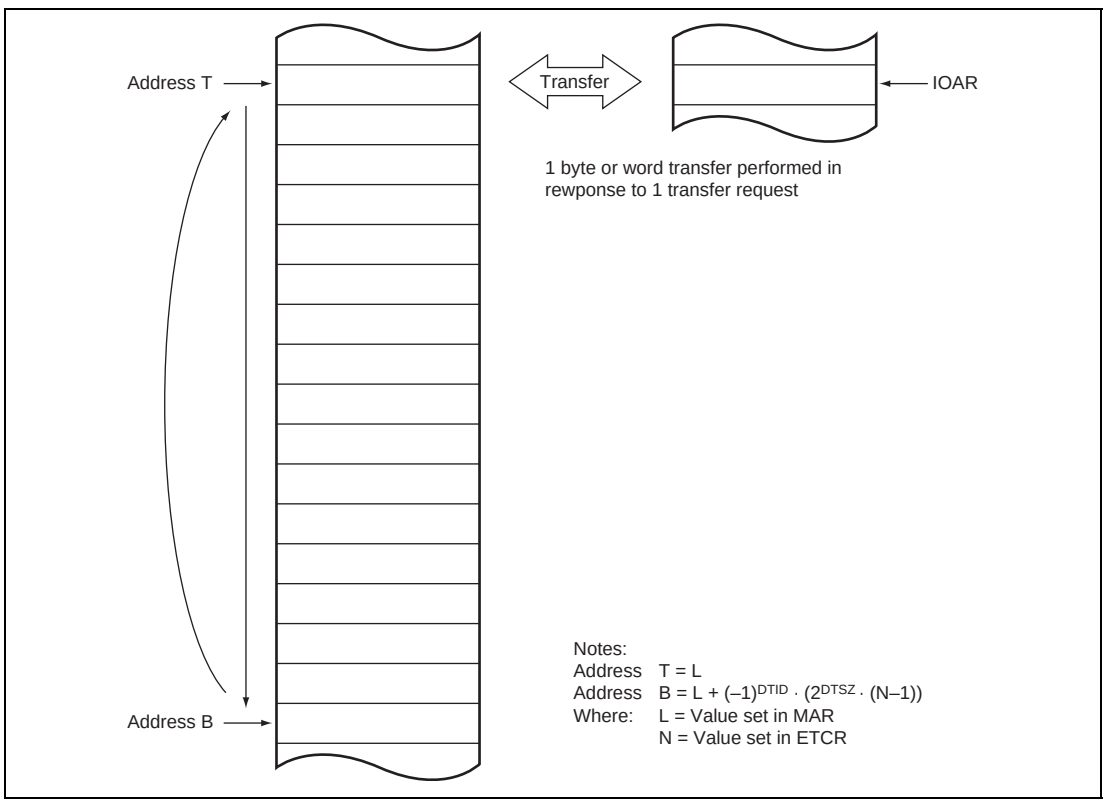



**Figure 7.6 Operation in Repeat mode**

Transfer requests (activation sources) consist of A/D conversion end interrupt, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 2 compare match/input capture A interrupts. Figure 7.7 shows an example of the setting procedure for repeat mode.

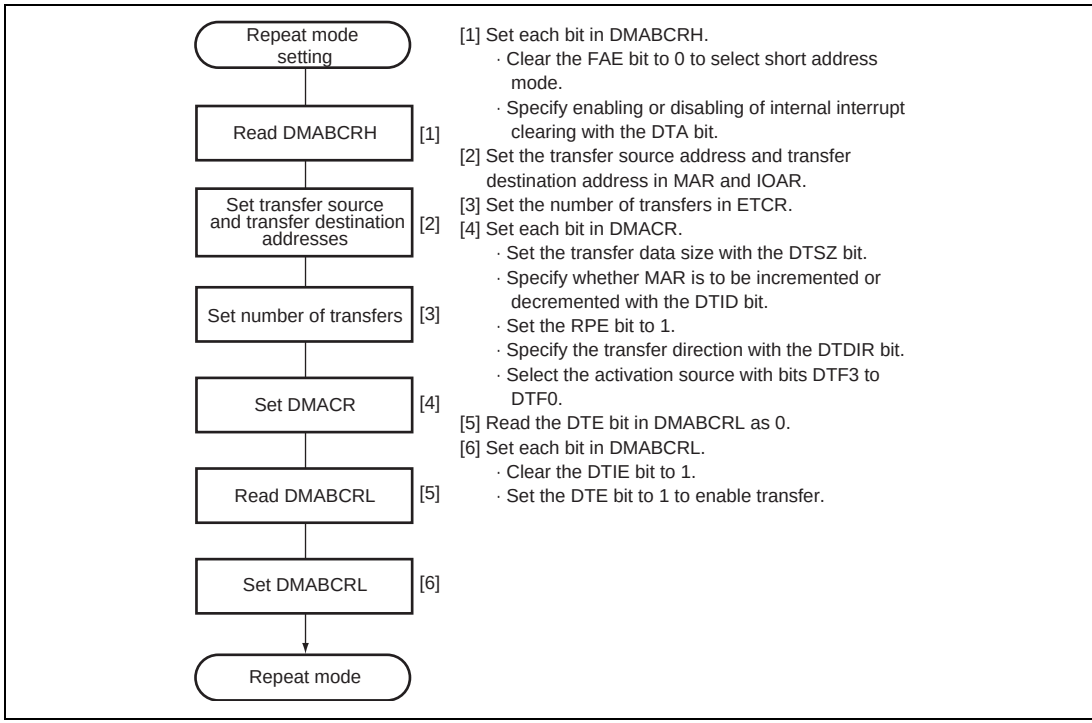
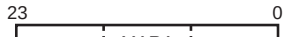



**Figure 7.7 Example of Repeat Mode Setting Procedure**

### 7.4.5 Normal Mode

In normal mode, transfer is performed with channels A and B used in combination. Normal mode can be specified by setting the FAE bit in DMABCR to 1 and clearing the BLKE bit in DMACRA to 0. In normal mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCRA. The transfer source is specified by MARA, and the transfer destination by MARB. Table 7.6 summarizes register functions in normal mode.

**Table 7.6 Register Functions in Normal Mode**

| Register | Function | Initial Setting | Operation |
|---|---|---|---|
|  | Source address register | Start address of transfer source | Incremented/decremented every transfer, or fixed |
| | Destination address register | Start address of transfer destination | Incremented/decremented every transfer, or fixed |
| | Transfer counter | Number of transfers | Decremented every transfer; transfer ends when count reaches H'0000 |

MARA and MARB specify the start addresses of the transfer source and transfer destination, respectively, as 24 bits. MAR can be incremented or decremented by 1 or 2 each time a byte or word is transferred, or can be fixed. Incrementing, decrementing, or holding a fixed value can be set separately for MARA and MARB.

The number of transfers is specified by ETCRA as 16 bits. ETCRA is decremented each time a transfer is performed, and when its value reaches H'0000 the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU. The maximum number of transfers, when H'0000 is set in ETCRA, is 65,536.

Figure 7.8 illustrates operation in normal mode.

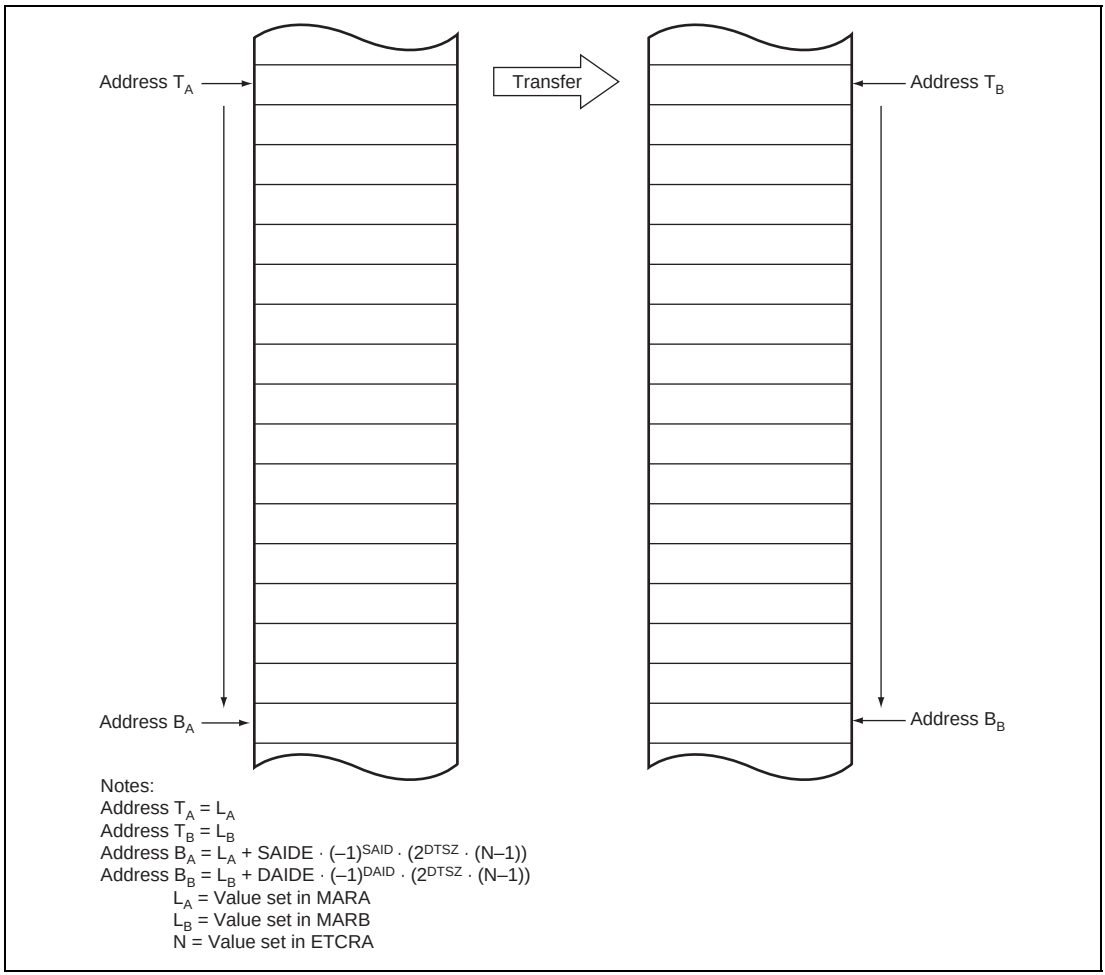
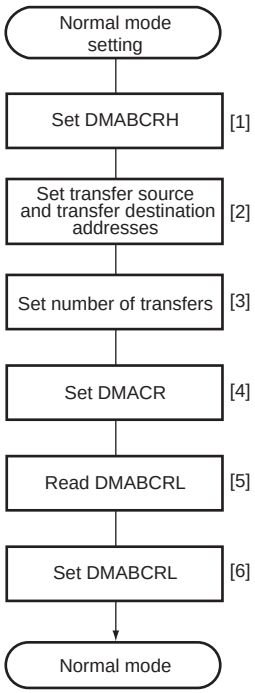



**Figure 7.8 Operation in Normal Mode**

Transfer requests (activation sources) are external requests and auto-requests. With auto-request, the DMAC is only activated by register setting, and the specified number of transfers are performed automatically. With auto-request, cycle steal mode or burst mode can be selected. In cycle steal mode, the bus is released to another bus master each time a transfer is performed. In burst mode, the bus is held continuously until transfer ends. For setting details, see section 7.3.4, DMA Controller Register (DMACR).

Figure 7.9 shows an example of the setting procedure for normal mode.




[1] Set each bit in DMABCRH.
- Set the FAE bit to 1 to select full address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address in MARA, and the transfer destination address in MARB.
[3] Set the number of transfers in ETCRA.
[4] Set each bit in DMACRA and DMACRB.
- Set the transfer data size with the DTSZ bit.
- Specify whether MARA is to be incremented, decremented, or fixed, with the SAID and SAIDE bits.
- Clear the BLKE bit to 0 to select normal mode.
- Specify whether MARB is to be incremented, decremented, or fixed, with the DAID and DAIDE bits.
- Select the activation source with bits DTF3 to DTF0.

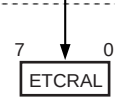
[5] Read the DTE = 0 and DTME = 0 in DMABCRL.
[6] Set each bit in DMABCRL.
- Specify enabling or desabling of transfer end interrupts with the DTIE bit.
- Set both the DTME bit and the DTE bit to 1 to enable transfer.

**Figure 7.9 Example of Normal Mode Setting Procedure**

### 7.4.6 Block Transfer Mode

In block transfer mode, transfer is performed with channels A and B used in combination. Block transfer mode can be specified by setting the FAE bit in DMABCR and the BLKE bit in DMACRA to 1. In block transfer mode, a transfer of the specified block size is carried out in response to a single transfer request, and this is executed the specified number of times. The transfer source is specified by MARA, and the transfer destination by MARB. Either the transfer source or the transfer destination can be selected as a block area (an area composed of a number of bytes or words). Table 7.7 summarizes register functions in block transfer mode.

**Table 7.7 Register Functions in Block Transfer Mode**

| Register | Function | Initial Setting | Operation |
|---|---|---|---|
| | Source address register | Start address of transfer source | Incremented/decremented every transfer, or fixed |
| | Description address register | Start address of transfer destination | Incremented/decremented every transfer, or fixed |
| | Holds block size | Block size | Fixed |
|  | Block size counter | Block size | decremented every transfer; ETCRH value copied when count reaches H'00 |
| | Block transfer counter | Number of block transfers | Decremented every block transfer; transfer ends when count reaches H'0000 |

MARA and MARB specify the start addresses of the transfer source and transfer destination, respectively, as 24 bits. MAR can be incremented or decremented by 1 or 2 each time a byte or word is transferred, or can be fixed. Incrementing, decrementing, or holding a fixed value can be set separately for MARA and MARB. Whether a block is to be designated for MARA or for MARB is specified by the BLKDIR bit in DMACRA.

To specify the number of transfers, if M is the size of one block (where M = 1 to 256) and N transfers are to be performed (where N = 1 to 65,536), M is set in both ETCRAH and ETCRAL, and N in ETCRB.

Figure 7.10 illustrates operation in block transfer mode when MARB is designated as a block area.

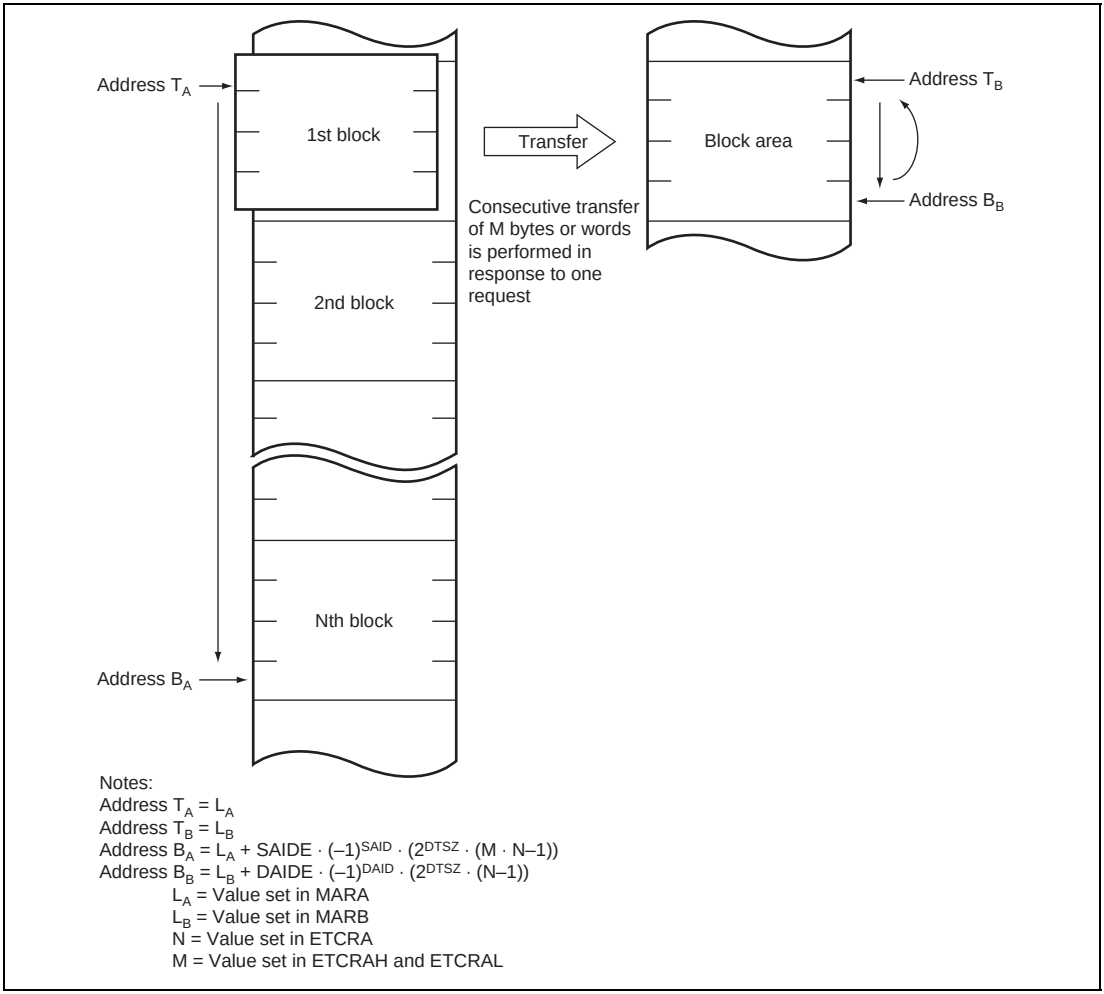



**Figure 7.10 Operation in Block Transfer Mode (BLKDIR = 0)**

Figure 7.11 illustrates operation in block transfer mode when MARA is designated as a block area.

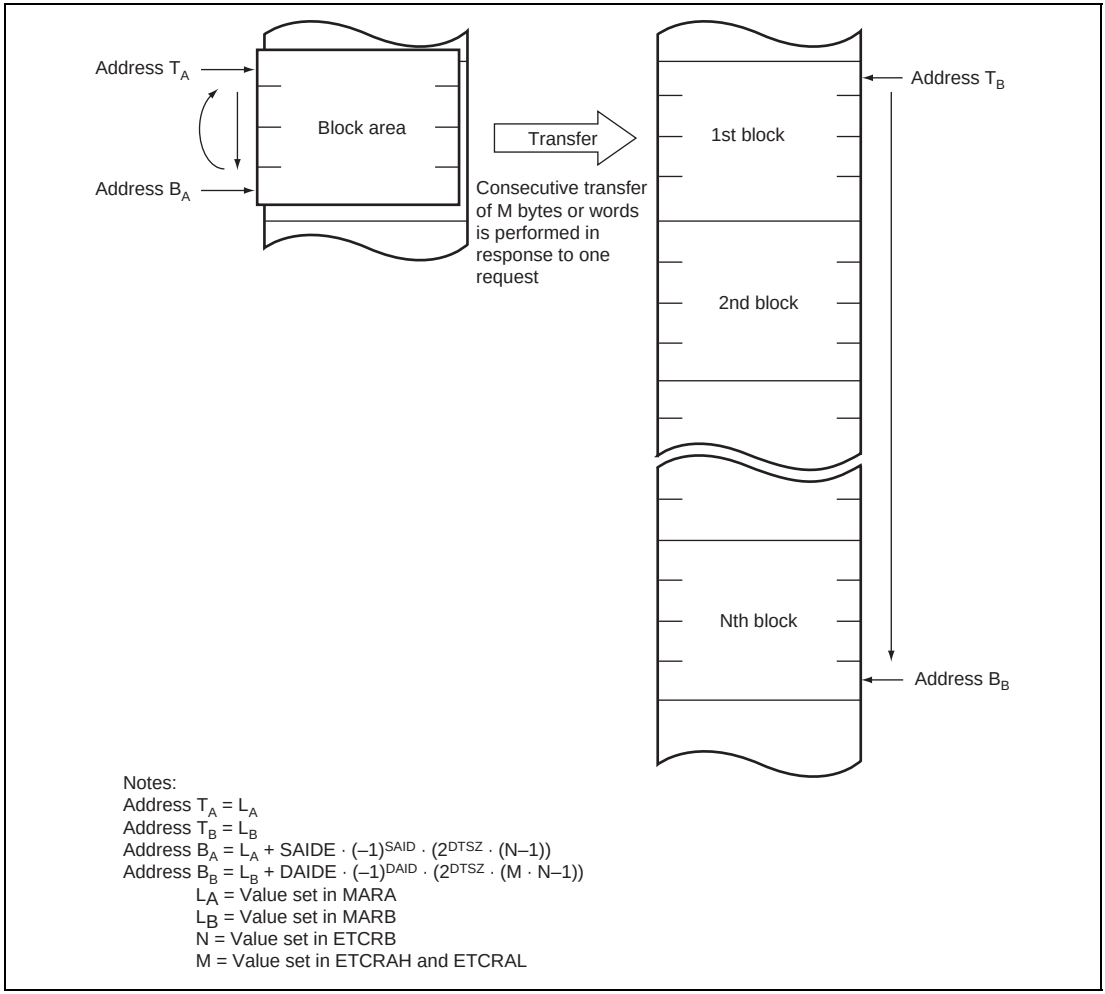



**Figure 7.11 Operation in Block Transfer Mode (BLKDIR = 1)**

ETCRAL is decremented by 1 each time a byte or word transfer is performed. In response to a single transfer request, burst transfer is performed until the value in ETCRAL reaches H'00. ETCRAL is then loaded with the value in ETCRAH. At this time, the value in the MAR register for which a block designation has been given by the BLKDIR bit in DMACRA is restored in accordance with the DTSZ, SAID/DAID, and SAIDE/DAIDE bits in DMACR.

ETCRB is decremented by 1 every block transfer, and when the count reaches H'0000 the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this point, an interrupt request is sent to the CPU. Figure 7.12 shows the operation flow in block transfer mode.

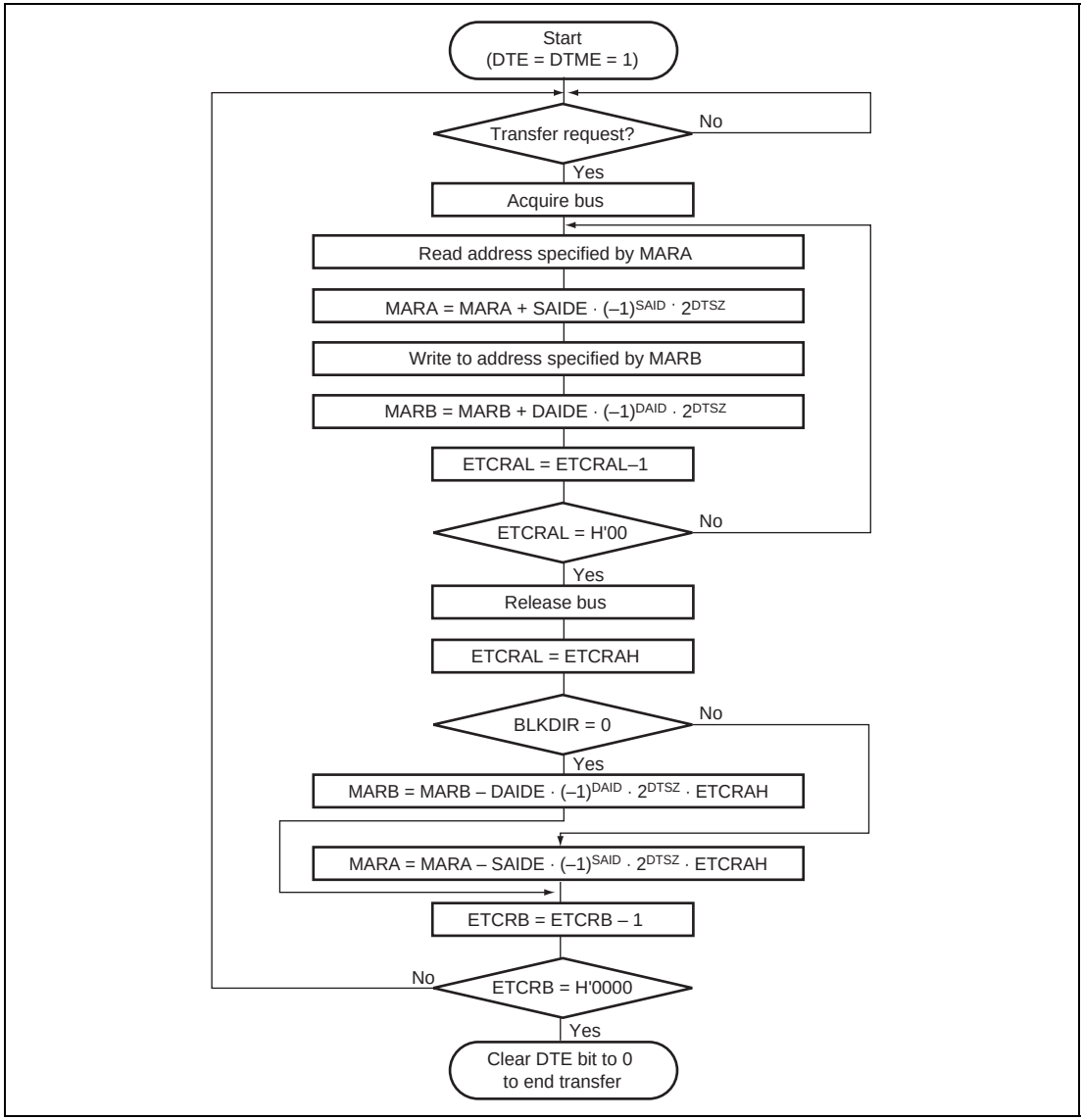
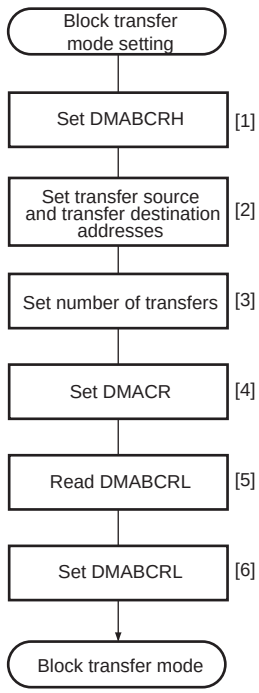



**Figure 7.12 Operation Flow in Block Transfer Mode**

Transfer requests (activation sources) consist of A/D conversion end interrupt, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 2 compare match/input capture A interrupts. For details, see section 7.3.4, DMA Control Register (DMACR). Figure 7.13 shows an example of the setting procedure for block transfer mode.




[1] Set each bit in DMABCRH.
- Set the FAE bit to 1 to select full address mode.
- Specify enabling or disabling of internal interrupt clearing with the DTA bit.

[2] Set the transfer source address in MARA, and the transfer destination address in MARB.

[3] Set the transfer source address in ETCRAH and ETCRAL. Set the number of transfers in ETCRB.

[4] Set each bit in DMACRA and DMACRB.
- Set the transfer data size with the DTSZ bit.
- Specify whether MARA is to be incremented, decremented, or fixed, with the SAID and SAIDE bits.
- Set the BLKE bit to 1 to select block transfer mode.
- Specify whether the transfer source or the transfer destination is a block area with the BLKDIR bit.
- Specify whether MARB is to be incremented, decremented, or fixed, with the DAID and DAIDE bits.
- Select the activation source with bits DTF3 to DTF0.

[5] Read the DTE = 0 and DTME = 0 in DMABCRL.

[6] Set each bit in DMABCRL.
- Specify enabling or desabling of transfer end interrupts to the CPU with the DTIE bit.
- Set both the DTME bit and the DTE bit to 1 to enable transfer.

**Figure 7.13 Example of Block Transfer Mode Setting Procedure**

### 7.4.7 DMAC Activation Sources

DMAC activation sources consist of internal interrupts, external requests, and auto-requests. The activation sources that can be specified depend on the transfer mode, as shown in table 7.8.

**Table 7.8 DMAC Activation Sources**

| | | | Full Address Mode | |
|---|---|---|---|---|
| **Activation Source** | | **Short Address Mode** | **Normal Mode** | **Block Transfer Mode** |
| Internal Interrupt | ADI | ○ | × | ○ |
| | TXI0 | ○ | × | ○ |
| | RXI0 | ○ | × | ○ |
| | TGI0A | ○ | × | ○ |
| | TGI1A | ○ | × | ○ |
| | TGI2A | ○ | × | ○ |
| USB request | Low level input of the $\overline{\text{DREQ}}$ signal | × | ○ | × |
| Auto-request | | × | ○ | × |

Legend:
○ : Can be specified
×: Cannot be specified

**Activation by Internal Interrupt:** An interrupt request selected as a DMAC activation source can be sent simultaneously to the CPU. For details, see section 5, Interrupt Controller.

With activation by an internal interrupt, the DMAC accepts the request independently of the interrupt controller. Consequently, interrupt controller priority settings are not accepted.

If the DMAC is activated by an interrupt request that is not used as a CPU interrupt source (DTA = 1), the interrupt source flag is cleared automatically by the DMA transfer. With ADI, TXI, and RXI interrupts, however, the interrupt source flag is not cleared unless the prescribed register is accessed in a DMA transfer. If the same interrupt is used as an activation source for more than one channel, the interrupt request flag is cleared when the highest-priority channel is activated first. Transfer requests for other channels are held pending in the DMAC, and activation is carried out in order of priority.

When DTE = 0, such as after completion of a transfer, a request from the selected activation source is not sent to the DMAC, regardless of the DTA bit. In this case, the relevant interrupt request is sent to the CPU. In case of overlap with a CPU interrupt source (DTA = 0), the interrupt request flag is not cleared by the DMAC.

**Activation by USB Request:** The USB request ($\overline{\text{DREQ}}$ signal) is specified as a DMAC activation source. The USB request is generated by the level sense. In full-address normal mode, the USB request is carried out as follows.

While the $\overline{\text{DREQ}}$ signal is kept high, the DMAC waits for the transfer request. While the $\overline{\text{DREQ}}$ signal is kept low, the DMAC releases the bus each time a byte is transferred and the transfer is performed continuously. When the $\overline{\text{DREQ}}$ signal is driven high during the transfer, the transfer is halted and the DMAC waits for the transfer request.

**Activation by Auto-Request:** Auto-request activation is performed by register setting only, and transfer continues to the end. With auto-request activation, cycle steal mode or burst mode can be selected.

In cycle steal mode, the DMAC releases the bus to another bus master each time a byte or word is transferred. DMA and CPU cycles usually alternate. In burst mode, the DMAC keeps possession of the bus until the end of the transfer, and transfer is performed continuously.

### 7.4.8 Basic DMAC Bus Cycles

An example of the basic DMAC bus cycle timing is shown in figure 7.14. In this example, word-size transfer is performed from 16-bit, 2-state access space to 8-bit, 3-state access space. When the bus is transferred from the CPU to the DMAC, a source address read and destination address write are performed. The bus is not released in response to another bus request, etc., between these read and write operations. As with CPU cycles, DMA cycles conform to the bus controller settings.

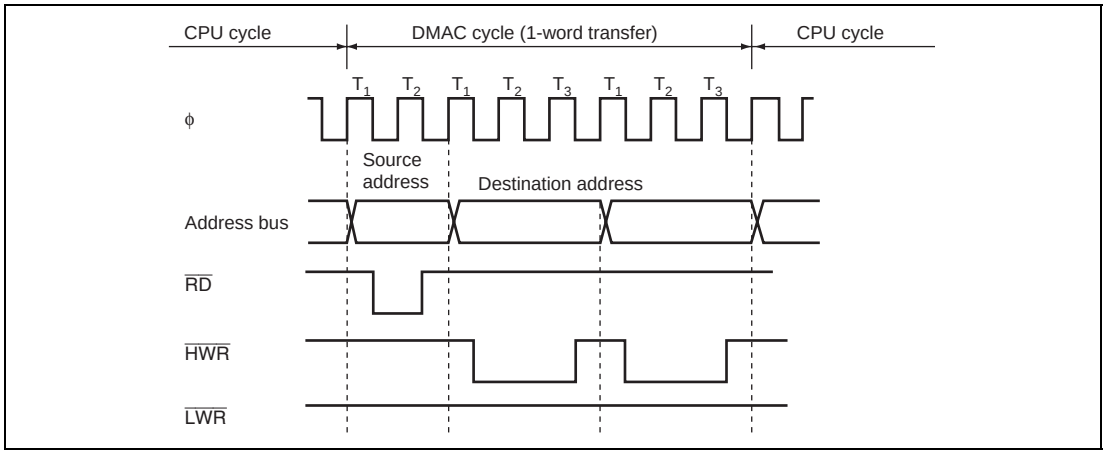



**Figure 7.14 Example of DMA Transfer Bus Timing**

The address is not output to the external address bus in an access to on-chip memory or an internal I/O register.

### 7.4.9 DMAC Bus Cycles (Dual Address Mode)

**Short Address Mode:** Figure 7.15 shows a transfer example in which $\overline{\text{TEND}}$* output is enabled and byte-size short address mode transfer (sequential/idle/repeat mode) is performed from external 8-bit, 2-state access space to internal I/O space.

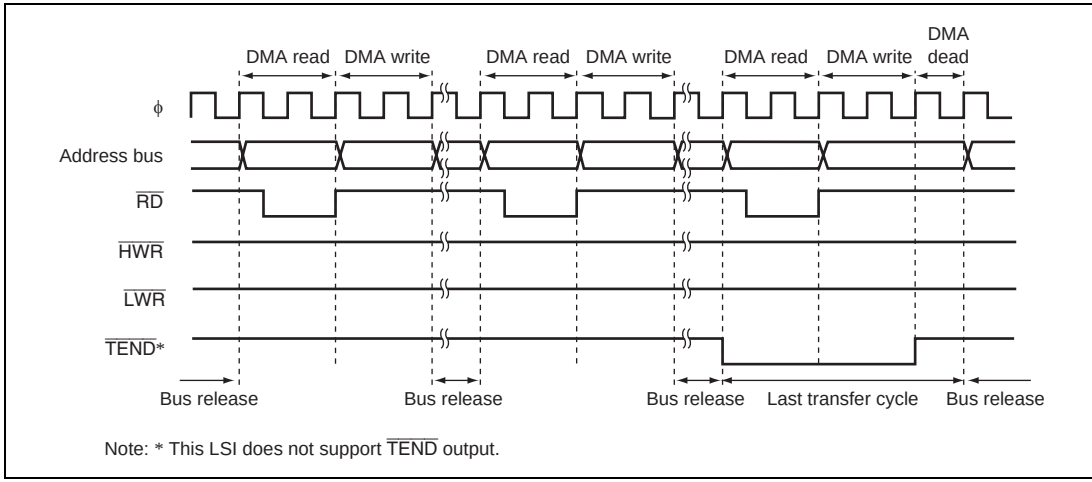



**Figure 7.15 Example of Short Address Mode Transfer**

A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released one or more bus cycles are inserted by the CPU.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

In repeat mode, when $\overline{\text{TEND}}$* output is enabled, $\overline{\text{TEND}}$* output goes low in the transfer cycle in which the transfer counter reaches 0.

Note: * This LSI does not support $\overline{\text{TEND}}$ output.

**Full Address Mode (Cycle Steal Mode):** Figure 7.16 shows a transfer example in which $\overline{\text{TEND}}$* output is enabled and word-size full address mode transfer (cycle steal mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

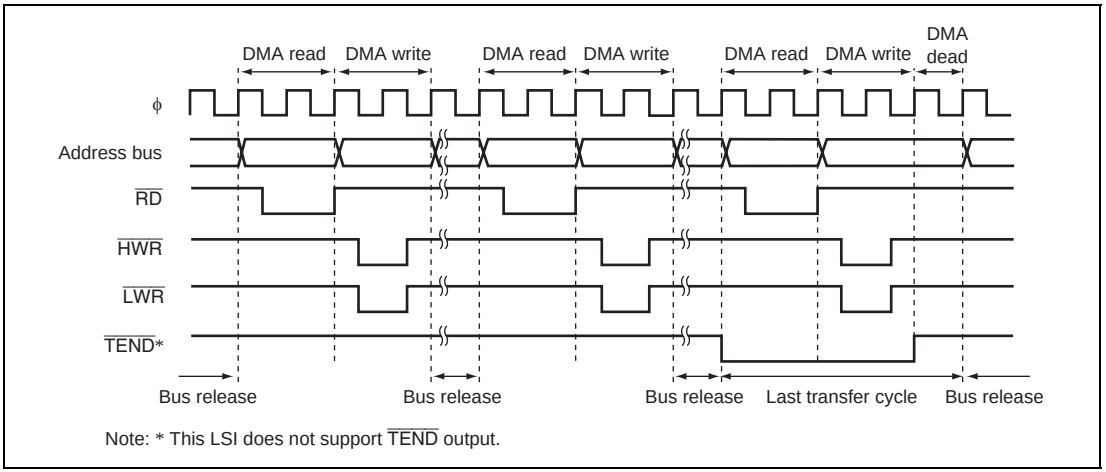



**Figure 7.16 Example of Full Address Mode (Cycle Steal) Transfer**

A one-byte or one-word transfer is performed, and after the transfer the bus is released. While the bus is released one bus cycle is inserted by the CPU.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

Note: * This LSI does not support $\overline{\text{TEND}}$ output.

**Full Address Mode (Burst Mode):** Figure 7.17 shows a transfer example in which $\overline{\text{TEND}}$* output is enabled and word-size full address mode transfer (burst mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

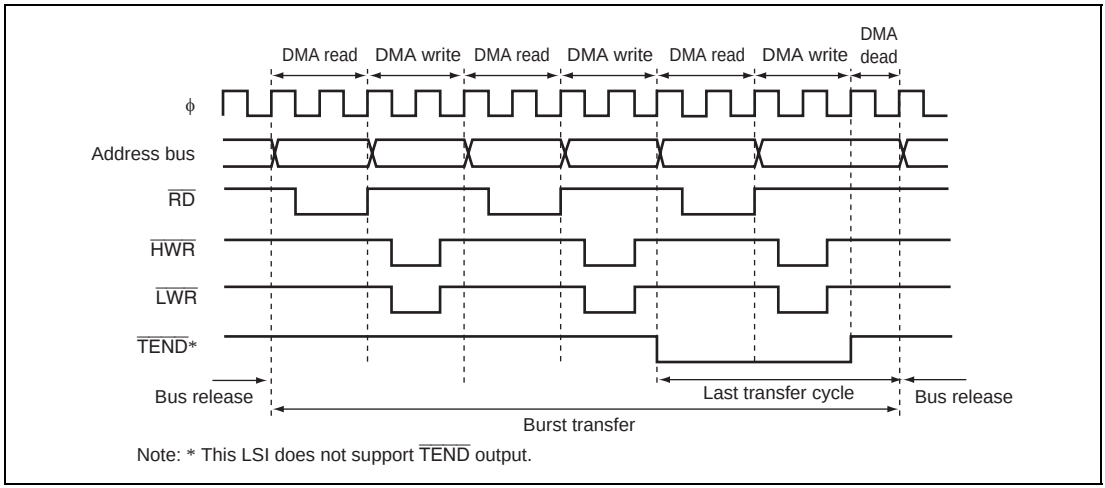



**Figure 7.17 Example of Full Address Mode (Burst Mode) Transfer**

In burst mode, one-byte or one-word transfers are executed consecutively until transfer ends. In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

If a request from another higher-priority channel is generated after burst transfer starts, that channel has to wait until the burst transfer ends.

If an NMI is generated while a channel designated for burst transfer is in the transfer enabled state, the DTME bit is cleared and the channel is placed in the transfer disabled state. If burst transfer has already been activated inside the DMAC, the bus is released on completion of a one-byte or one-word transfer within the burst transfer, and burst transfer is suspended. If the last transfer cycle of the burst transfer has already been activated inside the DMAC, execution continues to the end of the transfer even if the DTME bit is cleared.

Note: * This LSI does not support $\overline{\text{TEND}}$ output.

**Full Address Mode (Block Transfer Mode):** Figure 7.18 shows a transfer example in which $\overline{\text{TEND}}$* output is enabled and word-size full address mode transfer (block transfer mode) is performed from internal 16-bit, 1-state access space to external 16-bit, 2-state access space.

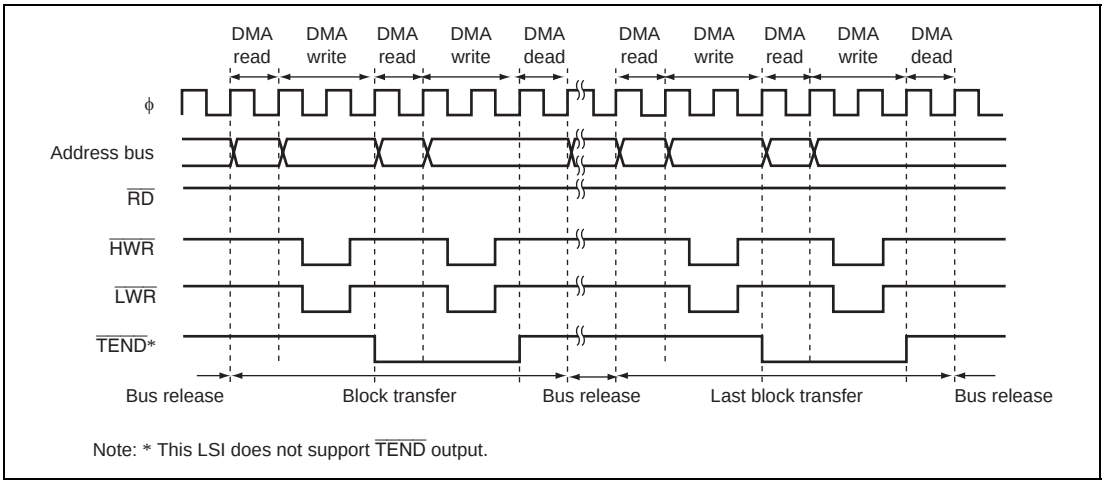



**Figure 7.18 Example of Full Address Mode (Block Transfer Mode) Transfer**

A one-block transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released, one or more bus cycles are inserted by the CPU.

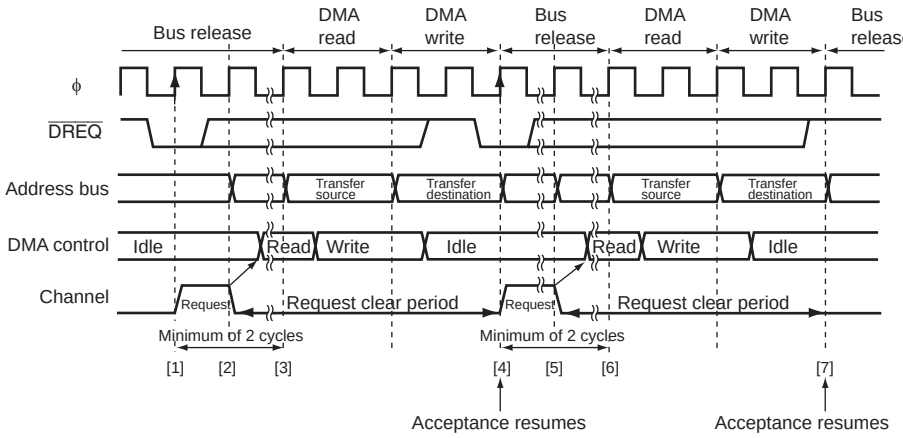
In the transfer end cycle of each block (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

One block is transmitted without interruption. NMI generation does not affect block transfer operation.

Note: * This LSI does not support $\overline{\text{TEND}}$ output.

**$\overline{\text{DREQ}}$ Signal Level Activation Timing (Normal Mode):** Set the DTA bit for the channel for which the $\overline{\text{DREQ}}$ signal is selected to 1.

Figure 7.19 shows an example of $\overline{\text{DREQ}}$ level activated normal mode transfer.




[1] Acceptance after transfer enabling; the $\overline{\text{DREQ}}$ signal low level is sampled on the rising edge of f, and the request is held.
[2] [5] The request is cleared at the next bus break, and activation is started in the DMAC.
[3] [6] Start of DMA cycle.
[4] [7] Acceptance is resumed after the write cycle is completed.
(As in [1], the $\overline{\text{DREQ}}$ signal low level is sampled on the rising edge of φ, and the request is held.)

**Figure 7.19 Example of $\overline{\text{DREQ}}$ Level Activated Normal Mode Transfer**

$\overline{\text{DREQ}}$ signal sampling is performed every cycle, with the rising edge of the next φ cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the $\overline{\text{DREQ}}$ signal low level is sampled while acceptance by means of the $\overline{\text{DREQ}}$ signal is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. Acceptance resumes after the end of the write cycle, $\overline{\text{DREQ}}$ signal low level sampling is performed again, and this operation is repeated until the transfer ends.

Note: The $\overline{\text{DREQ}}$ signal of this chip is an internal signal of chip, so it is not output from the pin.

### 7.4.10 DMAC Multi-Channel Operation

The DMAC channel priority order is: channel 0 > channel 1, and channel A > channel B. Table 7.9 summarizes the priority order for DMAC channels.

**Table 7.9 DMAC Channel Priority Order**

| Short Address Mode | Full Address Mode | Priority |
|---|---|---|
| Channel 0A | Channel 0 | High |
| Channel 0B | | |
| Channel 1A | Channel 1 | |
| Channel 1B | | Low |

If transfer requests are issued simultaneously for more than one channel, or if a transfer request for another channel is issued during a transfer, when the bus is released the DMAC selects the highest-priority channel from among those issuing a request according to the priority order shown in table 7.9. During burst transfer, or when one block is being transferred in block transfer, the channel will not be changed until the end of the transfer. Figure 7.20 shows a transfer example in which transfer requests are issued simultaneously for channels 0A, 0B, and 1.

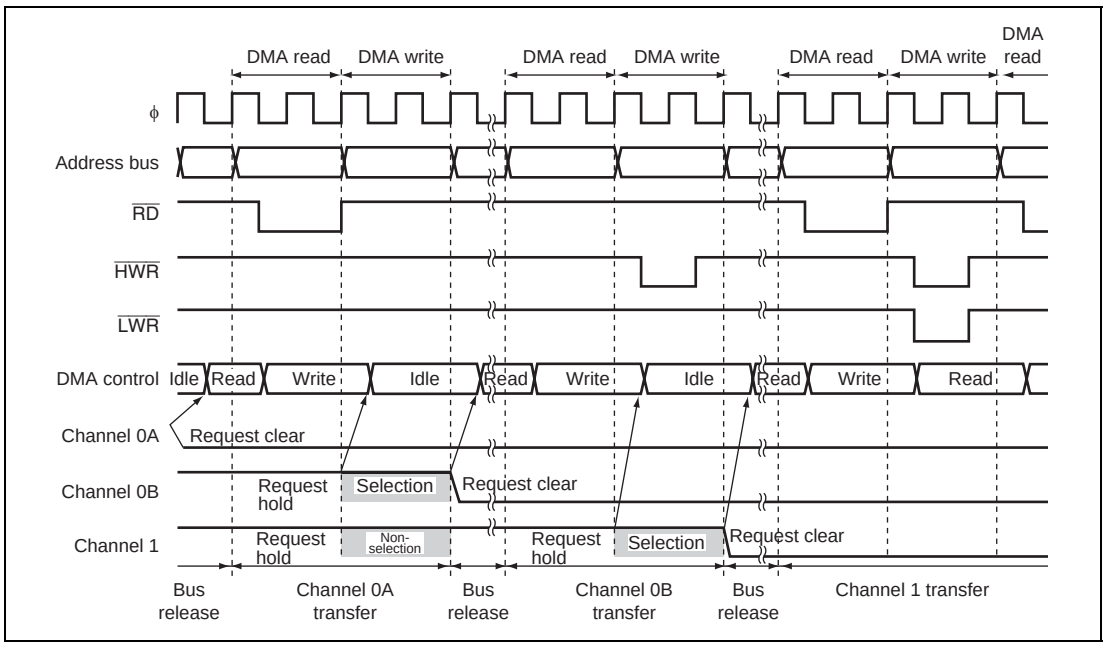



**Figure 7.20 Example of Multi-Channel Transfer**

### 7.4.11 Relation between the DMAC and External Bus Requests

There can be no break between a DMA cycle read and a DMA cycle write. This means that an external bus release cycle is not generated between the external read and external write in a DMA cycle.

In the case of successive read and write cycles, such as in burst transfer or block transfer, an external bus released state may be inserted after a write cycle.

When DMA cycle reads or writes are accesses to on-chip memory or internal I/O registers, these DMA cycles can be executed at the same time as refresh cycles or external bus release. However, simultaneous operation may not be possible when a write buffer is used.

### 7.4.12 NMI Interrupts and DMAC

When an NMI interrupt is requested, burst mode transfer in full address mode is interrupted. An NMI interrupt does not affect the operation of the DMAC in other modes.

In full address mode, transfer is enabled for a channel when both the DTE bit and the DTME bit are set to 1. With burst mode setting, the DTME bit is cleared when an NMI interrupt is requested.

If the DTME bit is cleared during burst mode transfer, the DMAC discontinues transfer on completion of the 1-byte or 1-word transfer in progress, then releases the bus, which passes to the CPU.

The channel on which transfer was interrupted can be restarted by setting the DTME bit to 1 again. Figure 7.21 shows the procedure for continuing transfer when it has been interrupted by an NMI interrupt on a channel designated for burst mode transfer.

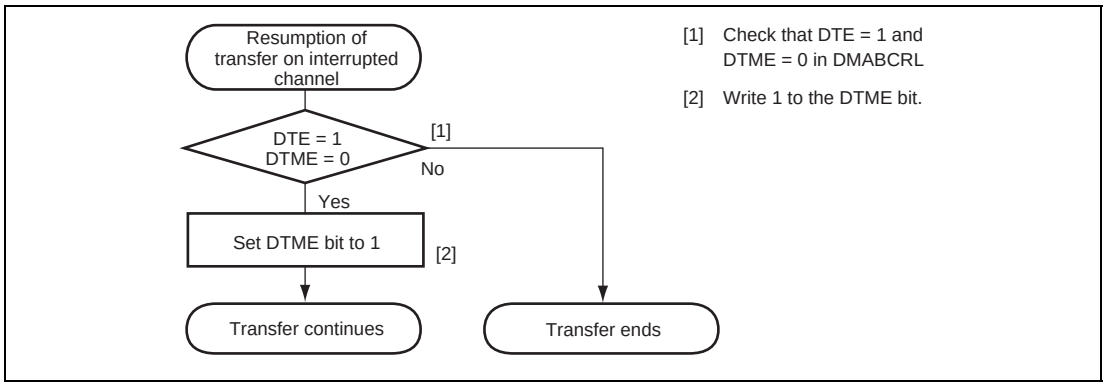



**Figure 7.21 Example of Procedure for Continuing Transfer on Channel Interrupted by NMI Interrupt**

### 7.4.13 Forced Termination of DMAC Operation

If the DTE bit for the channel currently operating is cleared to 0, the DMAC stops on completion of the 1-byte or 1-word transfer in progress. DMAC operation resumes when the DTE bit is set to 1 again. In full address mode, the same applies to the DTME bit. Figure 7.22 shows the procedure for forcibly terminating DMAC operation by software.

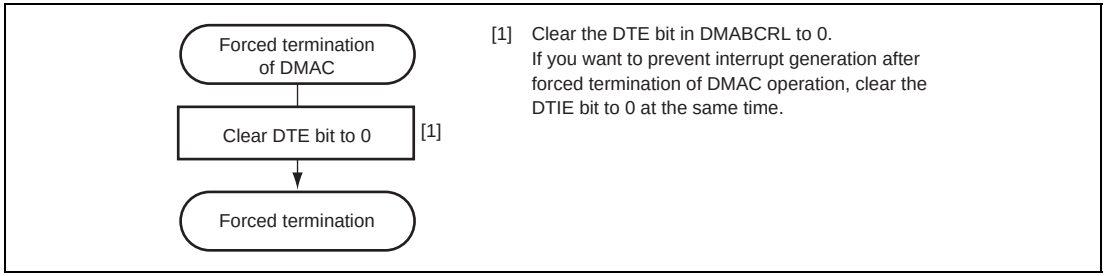



**Figure 7.22 Example of Procedure for Forcibly Terminating DMAC Operation**

### 7.4.14 Clearing Full Address Mode

Figure 7.23 shows the procedure for releasing and initializing a channel designated for full address mode. After full address mode has been cleared, the channel can be set to another transfer mode using the appropriate setting procedure.

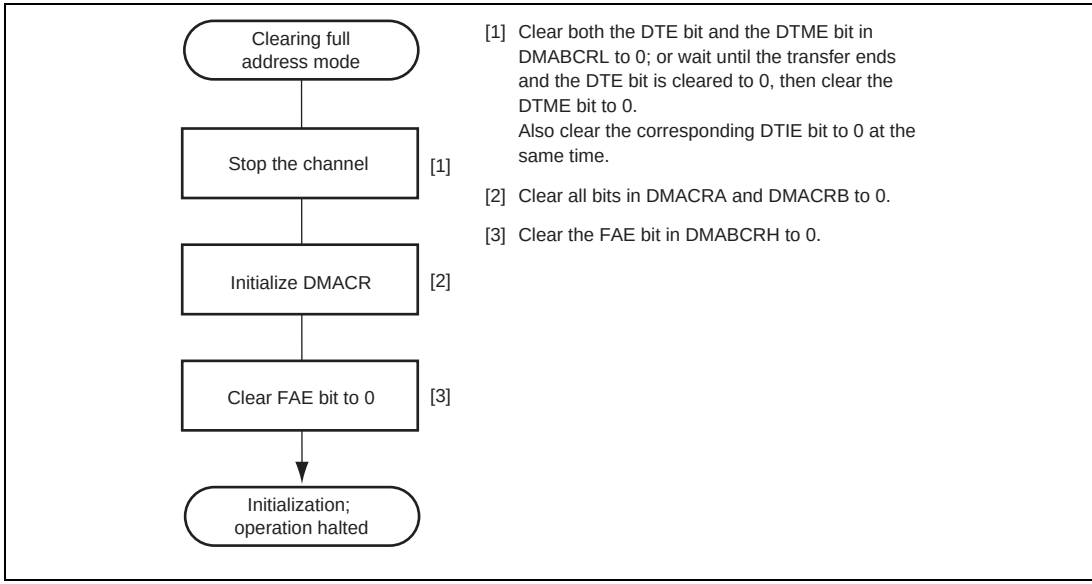



**Figure 7.23 Example of Procedure for Clearing Full Address Mode**

## 7.5 Interrupts

The sources of interrupts generated by the DMAC are transfer end and transfer break. Table 7.10 shows the interrupt sources and their priority order.

**Table 7.10 Interrupt Source Priority Order**

| Interrupt Name | Interrupt Source | | Interrupt Priority Order |
|---|---|---|---|
| | Short Address Mode | Full Address Mode | |
| DEND0A | Interrupt due to end of transfer on channel 0A | Interrupt due to end of transfer on channel 0 | High |
| DEND0B | Interrupt due to end of transfer on channel 0B | Interrupt due to break in transfer on channel 0 | |
| DEND1A | Interrupt due to end of transfer on channel 1A | Interrupt due to end of transfer on channel 1 | |
| DEND1B | Interrupt due to end of transfer on channel 1B | Interrupt due to break in transfer on channel 1 | Low |

Enabling or disabling of each interrupt source is set by means of the DTIE bit for the corresponding channel in DMABCR, and interrupts from each source are sent to the interrupt controller independently. The relative priority of transfer end interrupts on each channel is decided by the interrupt controller, as shown in table 7.10.

Figure 7.24 shows a block diagram of a transfer end/transfer break interrupt. An interrupt is always generated when the DTIE bit is set to 1 while DTE bit is cleared to 0.

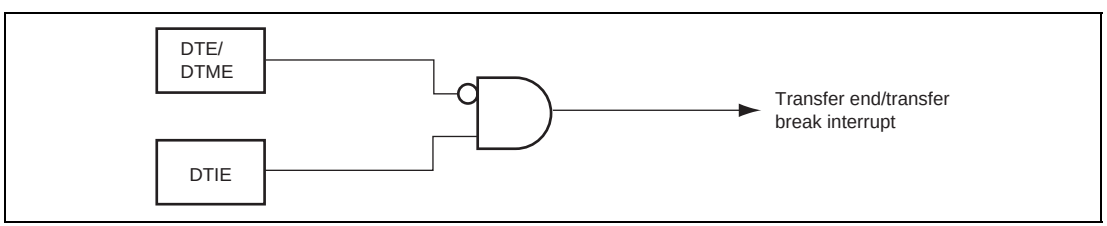



**Figure 7.24 Block Diagram of Transfer End/Transfer Break Interrupt**

In full address mode, a transfer break interrupt is generated when the DTME bit is cleared to 0 while DTIEB bit is set to 1. In both short address mode and full address mode, DMABCR should be set so as to prevent the occurrence of a combination that constitutes a condition for interrupt generation during setting.

## 7.6 Usage Notes

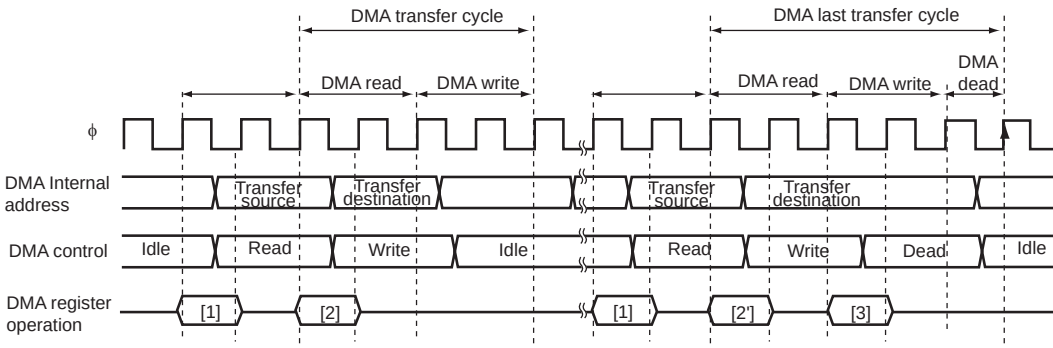
### 7.6.1 DMAC Register Access during Operation

Except for forced termination, the operating (including transfer waiting state) channel setting should not be changed. The operating channel setting should only be changed when transfer is disabled. Also, the DMAC register should not be written to in a DMA transfer.

DMAC register reads during operation (including the transfer waiting state) are described below.

1. DMAC control starts one cycle before the bus cycle, with output of the internal address. Consequently, MAR is updated in the bus cycle before DMAC transfer.

   Figure 7.25 shows an example of the update timing for DMAC registers in dual address transfer mode.




[1] Transfer source address register MAR operation (incremented/decremented/fixed)
Transfer counter ETCR operation (decremented)
Block size counter ETCR operation (decremented in block transfer mode)

[2] Transfer destination address register MAR operation (incremented/decremented/fixed)

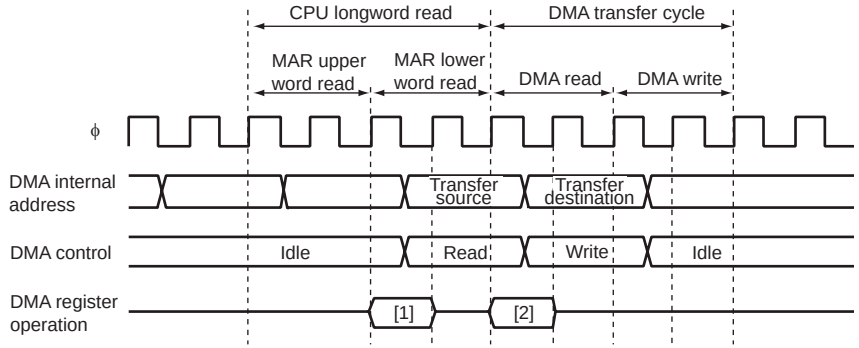
[2'] Transfer destination address register MAR operation (incremented/decremented/fixed)
Block transfer counter ETCR operation (decremented, in last transfer cycle of a block in block transfer mode)

[3] Transfer address register MAR restore operation (in block or repeat transfer mode)
Transfer counter ETCR restore (in repeat transfer mode)
Block size counter ETCR restore (in block transfer mode)

Note: The MAR operation is post-incrementing/decrementing of the DMA internal address value.

**Figure 7.25 DMAC Register Update Timing**

2. If a DMAC transfer cycle occurs immediately after a DMAC register read cycle, the DMAC register is read as shown in figure 7.26.




Note: The lower word of MAR is the updated value after the operation in [1].

**Figure 7.26 Contention between DMAC Register Update and CPU Read**

### 7.6.2 Module Stop

When the MSTPA7 bit in MSTPCR is set to 1, the DMAC clock stops, and the module stop state is entered. However, 1 cannot be written to the MSTPA7 bit if any of the DMAC channels is enabled. This setting should therefore be made when DMAC operation is stopped.

When the DMAC clock stops, DMAC register accesses can no longer be made. Since the following DMAC register settings are valid even in the module stop state, they should be invalidated, if necessary, before a module stop.

- Transfer end/suspend interrupt (DTE = 0 and DTIE = 1)
  For details, refer to section 20, Power-Down Modes.

### 7.6.3 Medium-Speed Mode

When the DTA bit is 0, internal interrupt signals specified as DMAC transfer sources are edge-detected. In medium-speed mode, the DMAC operates on a medium-speed clock, while on-chip peripheral modules operate on a high-speed clock.

Consequently, if the period in which the relevant interrupt source is cleared by the CPU or another DMAC channel, and the next interrupt is generated, is less than one state with respect to the DMAC clock (bus master clock), edge detection may not be possible and the interrupt may be ignored.

### 7.6.4 Activation Source Acceptance

At the start of activation source acceptance, a low level is detected in both $\overline{\text{DREQ}}$ signal falling edge sensing and low level sensing. Similarly, in the case of an internal interrupt, the interrupt request is detected. Therefore, a request is accepted from an internal interrupt or $\overline{\text{DREQ}}$ pin low level that occurs before execution of the DMABCRL write to enable transfer.

When the DMAC is activated, take any necessary steps to prevent an internal interrupt or $\overline{\text{DREQ}}$ signal low level remaining from the end of the previous transfer, etc.

### 7.6.5 Internal Interrupt after End of Transfer

When the DTE bit is cleared to 0 by the end of transfer or an abort, the selected internal interrupt request will be sent to the CPU even if DTA is set to 1.

Also, if internal DMAC activation has already been initiated when operation is aborted, the transfer is executed but flag clearing is not performed for the selected internal interrupt even if DTA is set to 1.

An internal interrupt request following the end of transfer or an abort should be handled by the CPU as necessary.

### 7.6.6 Channel Re-Setting

To reactivate a number of channels when multiple channels are enabled, use exclusive handling of transfer end interrupts, and perform DMABCR control bit operations exclusively. Note, in particular, that in cases where multiple interrupts are generated between reading and writing of DMABCR, and a DMABCR operation is performed during new interrupt handling, the DMABCR write data in the original interrupt handling routine will be incorrect, and the write may invalidate the results of the operations by the multiple interrupts. Ensure that overlapping DMABCR operations are not performed by multiple interrupts, and that there is no separation between read and write operations by the use of a bit-manipulation instruction. Also, when the DTE and DTME bits are cleared by the DMAC or are written with 0, they must first be read while cleared to 0 before the CPU can write a 1 to them.

# Section 8 I/O Ports

Table 8.1 and table 8.2 summarize the port functions of the H8S/2218 Group and H8S/2212 Group respectively. The pins of each port also have other functions such as input/output or external interrupt input pins of on-chip peripheral modules. Each I/O port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, and a port register (PORT) used to read the pin states. The input-only ports do not have DR and DDR.

Ports A to E have an on-chip input pull-up MOS and a input pull-up MOS control register (PCR) to control the on/off state of the input pull-up MOS. Ports 3 and A include an open-drain control register (ODR) that controls the on/off state of the output buffer PMOS.

All the I/O ports can drive a single TTL load and 30-pF capacitive load.

**Table 8.1 Port Functions of H8S/2218 Group**

| Port | Description | Modes 4 and 5 | Mode 6 | Mode 7 | Input/Output Type |
|---|---|---|---|---|---|
| Port 1 | General I/O port also functioning as TPU I/O pins, interrupt input pins, and address bus output pins | P17/TIOCB2/TCLKD | | | Schmitt trigger input ($\overline{\text{IRQ1}}$, $\overline{\text{IRQ0}}$) |
| | | P16/TIOCA2/$\overline{\text{IRQ1}}$ | | | |
| | | P15/TIOCB1/TCLKC | | | |
| | | P14/TIOCA1/$\overline{\text{IRQ0}}$ | | | |
| | | P13/TIOCD0/TCLKB/A23 | | P13/TIOCD0/TCLKB | |
| | | P12/TIOCC0/TCLKA/A22 | | P12/TIOCC0/TCLKA | |
| | | P11/TIOCB0/A21 | | P11/TIOCB0 | |
| | | P10/TIOCA0/A20 | | P10/TIOCA0 | |
| Port 3 | General I/O port also functioning as SCI_0 I/O pins and interrupt input pins | P36 | | | Open-drain output |
| | | P32/SCK0/$\overline{\text{IRQ4}}$ | | | Schmitt trigger input ($\overline{\text{IRQ4}}$) |
| | | P31/RxD0 | | | |
| | | P30/TxD0 | | | |
| Port 4 | General input port also functioning as A/D converter analog input pins | P43/AN3 | | | |
| | | P42/AN2 | | | |
| | | P41/AN1 | | | |
| | | P40/AN0 | | | |
| Port 7 | General I/O port also functioning as bus control output pins and manual reset input pins | P74/$\overline{\text{MRES}}$ | | P74/$\overline{\text{MRES}}$ | |
| | | P71/$\overline{\text{CS5}}$ | | P71 | |
| | | P70/$\overline{\text{CS4}}$ | | P70 | |

| Port | Description | Modes 4 and 5 | Mode 6 | Mode 7 | Input/Output Type |
|---|---|---|---|---|---|
| Port 9 | General input port also functioning as A/D converter analog input pins | P97/AN15<br>P96/AN14 | | | |
| Port A | General I/O port also functioning as SCI_2 I/O pins and address bus output pins | PA3/A19/SCK2<br>PA2/A18/RxD2<br>PA1/A17/TxD2<br>PA0/A16 | | PA3/SCK2<br>PA2/RxD2<br>PA1/TxD2<br>PA0 | On-chip input pull-up MOS<br>Open-drain output |
| Port B | General I/O port also functioning as address bus output pins | PB7/A15<br>PB6/A14<br>PB5/A13<br>PB4/A12<br>PB3/A11<br>PB2/A10<br>PB1/A9<br>PB0/A8 | | PB7<br>PB6<br>PB5<br>PB4<br>PB3<br>PB2<br>PB1<br>PB0 | On-chip input pull-up MOS |
| Port C | General I/O port also functioning as address bus output pins | A7 | When DDR = 0: PC7<br>When DDR = 1: A7 | PC7 | On-chip input pull-up MOS |
| | | A6 | When DDR = 0: PC6<br>When DDR = 1: A6 | PC6 | |
| | | A5 | When DDR = 0: PC5<br>When DDR = 1: A5 | PC5 | |
| | | A4 | When DDR = 0: PC4<br>When DDR = 1: A4 | PC4 | |
| | | A3 | When DDR = 0: PC73<br>When DDR = 1: A3 | PC3 | |
| | | A2 | When DDR = 0: PC2<br>When DDR = 1: A2 | PC2 | |
| | | A1 | When DDR = 0: PC1<br>When DDR = 1: A1 | PC1 | |
| | | A0 | When DDR = 0: PC0<br>When DDR = 1: A0 | PC0 | |
| Port D | General I/O port also functioning as data bus I/O pins | D15<br>D14<br>D13<br>D12<br>D11<br>D10<br>D9<br>D8 | | PD7<br>PD6<br>PD5<br>PD4<br>PD3<br>PD2<br>PD1<br>PD0 | On-chip input pull-up MOS |

| Port | Description | Modes 4 and 5 | Mode 6 | Mode 7 | Input/Output Type |
|---|---|---|---|---|---|
| Port E | General I/O port also functioning as data bus I/O pins | 8-bit bus mode: PE7<br>16-bit bus mode: D7 | | PE7 | On-chip input pull-up MOS |
| | | 8-bit bus mode: PE6<br>16-bit bus mode: D6 | | PE6 | |
| | | 8-bit bus mode: PE5<br>16-bit bus mode: D5 | | PE5 | |
| | | 8-bit bus mode: PE4<br>16-bit bus mode: D4 | | PE4 | |
| | | 8-bit bus mode: PE3<br>16-bit bus mode: D3 | | PE3 | |
| | | 8-bit bus mode: PE2<br>16-bit bus mode: D2 | | PE2 | |
| | | 8-bit bus mode: PE1<br>16-bit bus mode: D1 | | PE1 | |
| | | 8-bit bus mode: PE0<br>16-bit bus mode: D0 | | PE0 | |
| Port F | General I/O port also functioning as bus control signal I/O pins and interrupt input pins | When DDR = 0: PF7<br>When DDR = 1 (after reset): $\phi$ | | When DDR = 0 (after reset): PF7<br>When DDR = 1: $\phi$ | Schmitt trigger input ($\overline{\text{IRQ3}}$, $\overline{\text{IRQ2}}$) |
| | | $\overline{\text{AS}}$ | | PF6 | |
| | | $\overline{\text{RD}}$ | | PF5 | |
| | | $\overline{\text{HWR}}$ | | PF4 | |
| | | 8-bit bus mode: PF3/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$<br>16-bit bus mode: $\overline{\text{LWR}}$ | | PF3/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$ | |
| | | When WAITE = 0 (after reset): PF2<br>When WAITE = 1: $\overline{\text{WAIT}}$ | | PF2 | |
| | | When BRLE = 0 (after reset): PF1<br>When BRLE = 1: $\overline{\text{BACK}}$ | | PF1 | |
| | | When BRLE = 0 (after reset): PF0/$\overline{\text{IRQ2}}$<br>When BRLE = 1: $\overline{\text{BREQ}}$/$\overline{\text{IRQ2}}$ | | PF0/$\overline{\text{IRQ2}}$ | |

| Port | Description | Modes 4 and 5 | Mode 6 | Mode 7 | Input/Output Type |
|---|---|---|---|---|---|
| Port G | General I/O port also functioning as bus control output pins and interrupt Input pins | When DDR = 0 (after reset in mode 6): PG4<br>When DDR = 1 (after reset in modes 4, 5): $\overline{CS0}$ | | PG4 | Schmitt trigger input<br>($\overline{IRQ7}$) |
| | | When DDR = 0: PG3<br>When DDR = 1: $\overline{CS1}$ | | PG3 | |
| | | When DDR = 0: PG2<br>When DDR = 1: $\overline{CS2}$ | | PG2 | |
| | | When DDR = 0: PG1/$\overline{IRQ7}$<br>When DDR = 1: $\overline{CS3}$/$\overline{IRQ7}$ | | PG1/$\overline{IRQ7}$ | |

**Table 8.2 Port Functions of H8S/2212 Group**

| Port | Description | Mode 7 | Input/Output Type |
|---|---|---|---|
| Port 1 | General I/O port also functioning as TPU I/O pins and interrupt input pins | P17/TIOCB2/TCLKD<br>P16/TIOCA2/$\overline{IRQ1}$<br>P15/TIOCB1/TCLKC<br>P14/TIOCA1/$\overline{IRQ0}$<br>P13/TIOCD0/TCLKB<br>P12/TIOCC0/TCLKA<br>P11/TIOCB0<br>P10/TIOCA0 | Schmitt trigger input<br>($\overline{IRQ1}$, $\overline{IRQ0}$) |
| Port3 | General I/O port also functioning as SCI_0 I/O pins and interrupt input pins | P36<br>P32/SCK0/$\overline{IRQ4}$<br>P31/RxD0<br>P30/TxD0 | Open-drain output<br>Schmitt trigger input<br>($\overline{IRQ4}$) |
| Port 4 | General input port also functioning as A/D converter analog input pins | P43/AN3<br>P42/AN2<br>P41/AN1<br>P40/AN0 | |
| Port 7 | General I/O port | P77*<br>P76*<br>P75* | |
| Port 9 | General input port also functioning as A/D converter analog input pins | P97/AN15<br>P96/AN14 | |

| Port | Description | Mode 7 | Input/Output Type |
|---|---|---|---|
| Port A | General I/O port also functioning as SCI_2 I/O pins | PA3/SCK2<br>PA2/RxD2<br>PA1/TxD2 | On-chip input pull-up MOS<br>Open-drain output |
| Port E | General I/O port | PE7<br>PE6<br>PE5<br>PE4<br>PE3<br>PE2<br>PE1<br>PE0 | On-chip input pull-up MOS |
| Port F | General I/O port also functioning as interrupt input pins | When DDR = 0 (after reset): PF7<br>When DDR = 1: $\phi$<br>PF3/$\overline{\text{ADTRG}}$/$\overline{\text{IRQ3}}$<br>PF0/$\overline{\text{IRQ2}}$ | Schmitt trigger input<br>($\overline{\text{IRQ3}}$, $\overline{\text{IRQ2}}$) |
| Port G | General I/O port also functioning as interrupt input pins | PG1/$\overline{\text{IRQ7}}$<br>PG0* | Schmitt trigger input<br>($\overline{\text{IRQ7}}$) |

Note: * These pins are available only when EMLE = 0. These pins are not available when the H-UDI is used.

## 8.1 Port 1

In the H8S/2218 Group, the port 1 is an 8-bit I/O port also functioning as address bus pins, TPU I/O pins, and external interrupt input pins. In the H8S/2212 Group, the port 1 is an 8-bit I/O port also functioning as TPU I/O pins and external interrupt input pins. The port 1 has the following registers.

- Port 1 data direction register (P1DDR)
- Port 1 data register (P1DR)
- Port 1 register (PORT1)

### 8.1.1 Port 1 Data Direction Register (P1DDR)

P1DDR specifies input or output for the pins of the port 1.

Since P1DDR is a write-only register, the bit manipulation instructions must not be used to write P1DDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P17DDR | 0 | W | (H8S/2218 Group) |
| 6 | P16DDR | 0 | W | Modes 4 to 6:<br>If address output is enabled by the setting of bits AE3 to AE0 in PFCR, pins P13 to P10 are address outputs. Pins P17 to P14, and pins P13 to P10 when address output is disabled, are output ports when the corresponding P1DDR bits are set to 1, and input ports when the corresponding P1DDR bits are cleared to 0.<br>Mode 7:<br>Setting a P1DDR bit to 1 makes the corresponding port 1 pin an output port, while clearing the bit to 0 makes the pin an input port.<br>(H8S/2212 Group)<br>Setting a P1DDR bit to 1 makes the corresponding port 1 pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 5 | P15DDR | 0 | W | |
| 4 | P14DDR | 0 | W | |
| 3 | P13DDR | 0 | W | |
| 2 | P12DDR | 0 | W | |
| 1 | P11DDR | 0 | W | |
| 0 | P10DDR | 0 | W | |

### 8.1.2 Port 1 Data Register (P1DR)

P1DR stores output data for the port 1 pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P17DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 6 | P16DR | 0 | R/W | |
| 5 | P15DR | 0 | R/W | |
| 4 | P14DR | 0 | R/W | |
| 3 | P13DR | 0 | R/W | |
| 2 | P12DR | 0 | R/W | |
| 1 | P11DR | 0 | R/W | |
| 0 | P10DR | 0 | R/W | |

### 8.1.3 Port 1 Register (PORT1)

PORT1 indicates the pin states of the port 1.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P17 | —* | R | If the port 1 is read while P1DDR bits are set to 1, the P1DR value is read. If the port 1 is read while P1DDR bits are cleared to 0, the pin states are read. |
| 6 | P16 | —* | R | |
| 5 | P15 | —* | R | |
| 4 | P14 | —* | R | |
| 3 | P13 | —* | R | |
| 2 | P12 | —* | R | |
| 1 | P11 | —* | R | |
| 0 | P10 | —* | R | |

Note: * Determined by the states of pins P17 to P10.

### 8.1.4 Pin Functions

**Pin Functions of H8S/2218 Group**

Port 1 pins also function as address bus (A23 to A20) output pins, TPU I/O pins, and external interrupt input ($\overline{\text{IRQ0}}$ and $\overline{\text{IRQ1}}$) pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.3 P17 Pin Function**

| TPU Channel 2 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P17DDR | — | 0 | 1 |
| Pin Function | TIOCB2 output pin | P17 input pin | P17 output pin |
| | | TIOCB2 input pin | |
| | TCLKD input pin | | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.4 P16 Pin Function**

| TPU Channel 2 Setting*[1] | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P16DDR | — | 0 | 1 |
| Pin Function | TIOCA2 output pin | P16 input pin | P16 output pin |
| | | TIOCA2 input pin | |
| | $\overline{\text{IRQ1}}$ input pin*[2] | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. When this pin is used as an external interrupt pin, this pin must not be used for another function.

**Table 8.5 P15 Pin Function**

| TPU Channel 1 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P15DDR | — | 0 | 1 |
| Pin Function | TIOCB1 output pin | P15 input pin | P15 output pin |
| | | TIOCB1 input pin | |
| | TCLKC input pin | | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.6 P14 Pin Function**

| TPU Channel 1 Setting*[1] | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P14DDR | — | 0 | 1 |
| Pin Function | TIOCA1 output pin | P14 input pin | P14 output pin |
| | | TIOCA1 input pin | |
| | $\overline{\text{IRQ0}}$ input pin*[2] | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. When this pin is used as an external interrupt pin, this pin must not be used for another function.

**Table 8.7 P13 Pin Function**

| AE3 to AE0*[2] | Other than B'1111 | | | B'1111 |
|---|---|---|---|---|
| TPU Channel 0 Setting*[1] | Output Setting | Input Setting or Initial Value | | — |
| P13DDR | — | 0 | 1 | — |
| Pin Function | TIOCD0 output pin | P13 input pin | P13 output pin | A23 output pin*[2] |
| | | TIOCD0 input pin | | |
| | TCLKB input pin | | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. Valid in modes 4, 5, and 6.

**Table 8.8 P12 Pin Function**

| AE3 to AE0*[2] | Other than B'1111 | | | B'1111 |
|---|---|---|---|---|
| TPU Channel 0 Setting*[1] | Output Setting | Input Setting or Initial Value | | — |
| P12DDR | — | 0 | 1 | — |
| Pin Function | TIOCC0 output pin | P12 input pin | P12 output pin | A22 output pin*[2] |
| | | TIOCC0 input pin | | |
| | TCLKA input pin | | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. Valid in modes 4, 5, and 6.

**Table 8.9 P11 Pin Function**

| AE3 to AE0*[2] | Other than B'1110 to B'1111 | | | B'1110 to B'1111 |
|---|---|---|---|---|
| TPU Channel 0 Setting*[1] | Output Setting | Input Setting or Initial Value | | — |
| P11DDR | — | 0 | 1 | — |
| Pin Function | TIOCB0 output pin | P11 input pin | P11 output pin | A21 output pin*[2] |
| | | TIOCB0 input pin | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. Valid in modes 4, 5, and 6.

**Table 8.10 P10 Pin Function**

| AE3 to AE0*[2] | Other than B'1101 to B'1111 | | | B'1101 to B'1111 |
|---|---|---|---|---|
| TPU Channel 0 Setting*[1] | Output Setting | Input Setting or Initial Value | | — |
| P10DDR | — | 0 | 1 | — |
| Pin Function | TIOCA0 output pin | P10 input pin | P10 output pin | A20 output pin*[2] |
| | | TIOCA0 input pin | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. Valid in modes 4, 5, and 6.

**Pin Functions of H8S/2212 Group**

Port 1 pins also function as TPU I/O pins and external interrupt input ($\overline{\text{IRQ0}}$ and $\overline{\text{IRQ1}}$) pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.11 P17 Pin Function**

| TPU Channel 2 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P17DDR | — | 0 | 1 |
| Pin Function | TIOCB2 output pin | P17 input pin | P17 output pin |
| | | TIOCB2 input pin | |
| | TCLKD input pin | | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.12 P16 Pin Function**

| TPU Channel 2 Setting*[1] | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P16DDR | — | 0 | 1 |
| Pin Function | TIOCA2 output pin | P16 input pin | P16 output pin |
| | | TIOCA2 input pin | |
| | $\overline{\text{IRQ1}}$ input pin*[2] | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. When this pin is used as an external interrupt pin, this pin must not be used for another function.

**Table 8.13 P15 Pin Function**

| TPU Channel 1 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P15DDR | — | 0 | 1 |
| Pin Function | TIOCB1 output pin | P15 input pin | P15 output pin |
| | | TIOCB1 input pin | |
| | TCLKC input pin | | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.14 P14 Pin Function**

| TPU Channel 1 Setting*[1] | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P14DDR | — | 0 | 1 |
| Pin Function | TIOCA1 output pin | P14 input pin | P14 output pin |
| | | TIOCA1 input pin | |
| | $\overline{\text{IRQ0}}$ input pin*[2] | | |

Notes: 1. For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).
2. When this pin is used as an external interrupt pin, this pin must not be used for another function.

**Table 8.15 P13 Pin Function**

| TPU Channel 0 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P13DDR | — | 0 | 1 |
| Pin Function | TIOCD0 output pin | P13 input pin | P13 output pin |
| | | TIOCD0 input pin | |
| | TCLKB input pin | | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.16 P12 Pin Function**

| TPU Channel 0 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P12DDR | — | 0 | 1 |
| Pin Function | TIOCC0 output pin | P12 input pin | P12 output pin |
| | | TIOCC0 input pin | |
| | TCLKA input pin | | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.17 P11 Pin Function**

| TPU Channel 0 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P11DDR | — | 0 | 1 |
| Pin Function | TIOCB0 output pin | P11 input pin | P11 output pin |
| | | TIOCB0 input pin | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

**Table 8.18 P10 Pin Function**

| TPU Channel 0 Setting* | Output Setting | Input Setting or Initial Value | |
|---|---|---|---|
| P10DDR | — | 0 | 1 |
| Pin Function | TIOCA0 output pin | P10 input pin | P10 output pin |
| | | TIOCA0 input pin | |

Note: * For details on the TPU channel setting, refer to section 9, 16-Bit Timer Pulse Unit (TPU).

## 8.2 Port 3

The port 3 is a 4-bit I/O port also functioning as the SCI I/O pins and external interrupt input ($\overline{IRQ4}$) pins. The port 3 of the H8S/2218 Group has the same function as that of the H8S/2212 Group. The port 3 has the following registers.

- Port 3 data direction register (P3DDR)
- Port 3 data register (P3DR)
- Port 3 register (PORT3)
- Port 3 open-drain control register (P3ODR)

### 8.2.1 Port 3 Data Direction Register (P3DDR)

P3DDR specifies input or output for the pins of the port 3.

Since P3DDR is a write-only register, the bit manipulation instructions must not be used to write P3DDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | Undefined | — | Reserved<br>This bit is undefined and cannot be modified. |
| 6 | P36DDR | 0 | W | Setting a P3DDR bit to 1 makes the corresponding port 3 pin an output pin, while clearing the bit to 0 makes the pin an input pin. |
| 5 to 3 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 2 | P32DDR | 0 | W | Setting a P3DDR bit to 1 makes the corresponding port 3 pin an output pin, while clearing the bit to 0 makes the pin an input pin. |
| 1 | P31DDR | 0 | W | |
| 0 | P30DDR | 0 | W | |

### 8.2.2 Port 3 Data Register (P3DR)

P3DR stores output data for the port 3 pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | Undefined | — | Reserved<br>This bit is undefined and cannot be modified. |
| 6 | P36DR | 0 | R/W | Stores output data for a pin that functions as a general output port. |
| 5 to 3 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 2 | P32DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 1 | P31DR | 0 | R/W | |
| 0 | P30DR | 0 | R/W | |

### 8.2.3 Port 3 Register (PORT3)

PORT3 indicates the pin states of the port 3.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | Undefined | — | Reserved<br>This bit is undefined. |
| 6 | P36 | —* | R | If the port 3 is read while P3DDR bits are set to 1, the P3DR value is read. If the port 3 is read while P3DDR bits are cleared to 0, the pin states are read. |
| 5 to 3 | — | Undefined | — | Reserved<br>These bits are undefined. |
| 2 | P32 | —* | R | If the port 3 is read while P3DDR bits are set to 1, the P3DR value is read. If the port 3 is read while P3DDR bits are cleared to 0, the pin states are read. |
| 1 | P31 | —* | R | |
| 0 | P30 | —* | R | |

Note: * Determined by the states of pins P36 and P32 to P30.

### 8.2.4 Port 3 Open-Drain Control Register (P3ODR)

P3ODR controls the PMOS on/off state for each port 3 pin.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | Undefined | — | Reserved<br>This bit is undefined and cannot be modified. |
| 6 | P36ODR | 0 | R/W | Setting a P3ODR bit to 1 makes the corresponding port 3 pin an NMOS open-drain output pin, while clearing the bit to 0 makes the pin a CMOS output pin. |
| 5 to 3 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 2 | P32ODR | 0 | R/W | Setting a P3ODR bit to 1 makes the corresponding port 3 pin an NMOS open-drain output pin, while clearing the bit to 0 makes the pin a CMOS output pin. |
| 1 | P31ODR | 0 | R/W | |
| 0 | P30ODR | 0 | R/W | |

### 8.2.5 Pin Functions

Port 3 pins also function as SCI I/O pins and external interrupt input ($\overline{\text{IRQ4}}$) pins. The correspondence between the register specification and the pin functions is shown below. The P36 pin must be used as the D+ pull-up control output pin of the USB. For details, refer to section 14, Universal Serial Bus (USB).

**Table 8.19 P36 Pin Function**

| P36DDR | 0 | 1 |
|---|---|---|
| Pin Function | P36 input pin | P36 output pin<br>(D+ pull-up control output pin of USB) |

**Table 8.20 P32 Pin Function**

| CKE1 in SCR_0 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ in SMR_0 | 0 | | | 1 | — |
| CKE0 in SCR_0 | 0 | | 1 | — | — |
| P32DDR | 0 | 1 | — | — | — |
| Pin Function | P32 input pin | P32 output pin | SCK0 output pin | SCK0 output pin | SCK0 input pin |
| | $\overline{IRQ4}$ input pin* | | | | |

Note: * When this pin is used as an external interrupt pin, this pin must not be used for another function.

**Table 8.21 P31 Pin Function**

| RE in SCR_0 | 0 | | 1 |
|---|---|---|---|
| P31DDR | 0 | 1 | — |
| Pin Function | P31 input pin | P31 output pin | RxD0 input pin |

**Table 8.22 P30 Pin Function**

| TE in SCR_0 | 0 | | 1 |
|---|---|---|---|
| P30DDR | 0 | 1 | — |
| Pin Function | P30 input pin | P30 output pin | TxD0 output pin |

## 8.3 Port 4

The port 4 is a 4-bit input port also functioning as A/D converter analog input pins. The port 4 of the H8S/2218 Group has the same function as that of the H8S/2212 Group. The port 4 has the following register.

- Port 4 register (PORT4)

### 8.3.1 Port 4 Register (PORT4)

PORT4 indicates the pin states of the port 4.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are undefined. |
| 3 | P43 | —* | R | The pin states are always read when these bits are read. |
| 2 | P42 | —* | R | |
| 1 | P41 | —* | R | |
| 0 | P40 | —* | R | |

Note: * Determined by the states of pins P43 to P40.

### 8.3.2 Pin Function

The port 4 also functions as A/D converter analog input (AN3 to AN0) pins.

## 8.4 Port 7

In the H8S/2218 Group, the port 7 is a 3-bit I/O port also functioning as bus control output pins and manual reset input pins. In the H8S/2212 Group, the port 7 is a 3-bit I/O port also functioning as H-UDI pins. The port 7 has the following registers.

- Port 7 data direction register (P7DDR)
- Port 7 data register (P7DR)
- Port 7 register (PORT7)

### 8.4.1 Port 7 Data Direction Register (P7DDR)

P7DDR specifies input or output for the pins of the port 7.

Since P7DDR is a write-only register, the bit manipulation instructions must not be used to write P7DDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P77DDR | 0 | W | (H8S/2218 Group)<br>Reserved<br>These bits are undefined and cannot be modified.<br>(H8S/2212 Group)<br>When EMLE = 1: Pins P77 to P75 function as the H-UDI pins (TDO, TMS, TCK).<br>When EMLE = 0: If a P7DDR bit is set to 1, pins P77 to P75 function as output ports. If a P7DDR bit is cleared to 0, pins P77 to P75 function as input ports. |
| 6 | P76DDR | 0 | W | |
| 5 | P75DDR | 0 | W | |
| 4 | P74DDR | 0 | W | (H8S/2218 Group)<br>Setting a P7DDR bit to 1 makes the corresponding port 7 pin an output pin, while clearing the bit to 0 makes the pin an input pin.<br>(H8S/2212 Group)<br>Reserved<br>This bit is undefined and cannot be modified. |
| 3, 2 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 1 | P71DDR | 0 | W | (H8S/2218 Group) |
| 0 | P70DDR | 0 | W | Setting a P7DDR bit to 1 makes the corresponding port 7 pin an output pin, while clearing the bit to 0 makes the pin an input pin.<br>(H8S/2212 Group)<br>Reserved<br>These bits are undefined and cannot be modified. |

### 8.4.2 Port 7 Data Register (P7DR)

P7DR stores output data for the port 7 pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P77DR | 0 | R/W | (H8S/2218 Group) |
| 6 | P76DR | 0 | R/W | Reserved<br>These bits are undefined and cannot be modified. |
| 5 | P75DR | 0 | R/W | (H8S/2212 Group)<br>Store output data for the port 7 pins. |
| 4 | P74DR | 0 | R/W | (H8S/2218 Group)<br>Stores output data for the port 7 pins.<br>(H8S/2212 Group)<br>Reserved<br>This bit is undefined and cannot be modified. |
| 3, 2 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 1 | P71DR | 0 | R/W | (H8S/2218 Group) |
| 0 | P70DR | 0 | R/W | Store output data for the port 7 pins.<br>(H8S/2212 Group)<br>Reserved<br>These bits are undefined and cannot be modified. |

### 8.4.3 Port 7 Register (PORT7)

PORT7 indicates the pin states of the port 7.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P77 | —* | — | (H8S/2218 Group) |
| 6 | P76 | —* | — | Reserved |
| 5 | P75 | —* | — | These bits are undefined and cannot be modified.<br>(H8S/2212 Group)<br>If P7DDR bits are set to 1, the P7DR value is read. If P7DDR bits are cleared to 0, the pin states are read. |
| 4 | P74 | —* | R | (H8S/2218 Group)<br>If the port 7 is read while P7DDR bits are set to 1, the P7DR value is read. If the port 7 is read while P7DDR bits are cleared to 0, the pin states are read.<br>(H8S/2212 Group)<br>Reserved<br>This bit is undefined and cannot be modified. |
| 3, 2 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 1 | P71 | —* | R | (H8S/2218 Group) |
| 0 | P70 | —* | R | If the port 7 is read while P7DDR bits are set to 1, the P7DR value is read. If the port 7 is read while P7DDR bits are cleared to 0, the pin states are read.<br>(H8S/2212 Group)<br>Reserved<br>These bits are undefined and cannot be modified. |

Note: * Determined by the states of pins P77 to P74, P71, and P70.

### 8.4.4 Pin Functions

**Pin Functions of H8S/2218 Group**

Port 7 pins also function as bus control output pins and manual reset input pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.23 P74 Pin Function**

| MRESE | 0 | | 1 |
|---|---|---|---|
| P74DDR | 0 | 1 | — |
| Pin Function | P74 input pin | P74 output pin | $\overline{\text{MRES}}$ input pin |

**Table 8.24 P71 Pin Function**

| Operating Mode | Modes 4 to 6 | | Mode 7 | |
|---|---|---|---|---|
| P71DDR | 0 | 1 | 0 | 1 |
| Pin Function | P71 input pin | $\overline{\text{CS5}}$ output pin | P71 input pin | P71 output pin |

**Table 8.25 P70 Pin Function**

| Operating Mode | Modes 4 to 6 | | Mode 7 | |
|---|---|---|---|---|
| P70DDR | 0 | 1 | 0 | 1 |
| Pin Function | P70 input pin | $\overline{\text{CS4}}$ output pin | P70 input pin | P70 output pin |

**Pin Functions of H8S/2212 Group**

Port 7 pins also function as H-UDI pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.26 P77 Pin Function**

| EMLE | 0 | | 1 |
|---|---|---|---|
| P77DDR | 0 | 1 | — |
| Pin Function | P77 input pin | P77 output pin | TDO output pin |

**Table 8.27 P76 Pin Function**

| EMLE | 0 | | 1 |
|---|---|---|---|
| P76DDR | 0 | 1 | — |
| Pin Function | P76 input pin | P76 output pin | TCK input pin |

**Table 8.28 P75 Pin Function**

| EMLE | 0 | | 1 |
|---|---|---|---|
| P75DDR | 0 | 1 | — |
| Pin Function | P75 input pin | P75 output pin | TMS input pin |

## 8.5 Port 9

The port 9 is a 2-bit input port also functioning as A/D converter analog input pins. The port 9 of the H8S/2218 Group has the same function as that of the H8S/2212 Group.

- Port 9 register (PORT9)

### 8.5.1 Port 9 Register (PORT9)

PORT9 indicates the pin states of the port 9.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | P97 | —* | R | The pin states are always read when these bits are read. |
| 6 | P96 | —* | R | |
| 5 to 0 | — | Undefined | — | Reserved<br>These bits are undefined. |

Note: * Determined by the states of pins P97 and P96.

### 8.5.2 Pin Function

The port 9 also functions as A/D converter analog input (AN15 and AN14) pins.

## 8.6 Port A

In the H8S/2218 Group, the port A is a 4-bit I/O port also functioning as address bus (A19 to A16) output pins and SCI I/O pins. In the H8S/2212 Group, the port A is a 3-bit I/O port also functioning as SCI I/O pins. The port A has the following registers.

- Port A data direction register (PADDR)
- Port A data register (PADR)
- Port A register (PORTA)
- Port A pull-up MOS control register (PAPCR)
- Port A open-drain control register (PAODR)

### 8.6.1 Port A Data Direction Register (PADDR)

PADDR specifies input or output for the pins of the port A.

Since PADDR is a write-only register, the bit manipulation instructions must not be used to write PADDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 3 | PA3DDR | 0 | W | (H8S/2218 Group)<br>Modes 4 to 6:<br>If address output is enabled by the setting of bits AE3 to AE0 in PFCR, the corresponding port A pins are address outputs. When address output is disabled, setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port.<br>Mode 7:<br>Setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port.<br>(H8S/2212 Group)<br>Setting a PADDR bit to 1 makes the corresponding port A pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 2 | PA2DDR | 0 | W | |
| 1 | PA1DDR | 0 | W | |
| 0 | PA0DDR* | 0 | W | |

Note: * Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.6.2 Port A Data Register (PADR)

PADR stores output data for the port A pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 3 | PA3DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 2 | PA2DR | 0 | R/W | |
| 1 | PA1DR | 0 | R/W | |
| 0 | PA0DR* | 0 | R/W | |

Note: * Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.6.3 Port A Register (PORTA)

PORTA indicates the pin states of the port A.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are undefined. |
| 3 | PA3 | —*[1] | R | If the port A is read while PADDR bits are set to 1, the PADR value is read. If the port A is read while PADDR bits are cleared to 0, the pin states are read. |
| 2 | PA2 | —*[1] | R | |
| 1 | PA1 | —*[1] | R | |
| 0 | PA0*[2] | —*[1] | R | |

Notes: 1. Determined by the states of pins PA3 to PA0.
2. Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read.

### 8.6.4 Port A Pull-Up MOS Control Register (PAPCR)

PAPCR controls the on/off state of the port A input pull-up MOS. PAPCR is valid for port input and SCI input pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 3 | PA3PCR | 0 | R/W | When a pin functions as an input port, setting the corresponding bit to 1 turns on the input pull-up MOS for that pin. |
| 2 | PA2PCR | 0 | R/W | |
| 1 | PA1PCR | 0 | R/W | |
| 0 | PA0PCR* | 0 | R/W | |

Note: * Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.6.5 Port A Open-Drain Control Register (PAODR)

PAODR specifies an output type of the port A. PAODR is valid for port output and SCI output pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 3 | PA3ODR | 0 | R/W | Setting a PAODR bit to 1 makes the corresponding port A pin an NMOS open-drain output pin, while clearing the bit to 0 makes the pin a CMOS output pin. |
| 2 | PA2ODR | 0 | R/W | |
| 1 | PA1ODR | 0 | R/W | |
| 0 | PA0ODR* | 0 | R/W | |

Note: * Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.6.6 Pin Functions

#### Pin Functions of H8S/2218 Group

Port A pins also function as address bus (A19 to A16) output pins and SCI_2 I/O pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.29 PA3 Pin Function**

| Operating Mode | Modes 4 to 6 | | | | | | Mode 7 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| AE3 to AE0 | B'11×× | Other than B'11×× | | | | | — | | | | |
| CKE1 in SCR_2 | — | 0 | | | | 1 | 0 | | | | 1 |
| C/$\overline{A}$ in SMR_2 | — | 0 | | | 1 | — | 0 | | | 1 | — |
| CKE0 in SCR_2 | — | 0 | | 1 | — | — | 0 | | 1 | — | — |
| PA3DDR | — | 0 | 1 | — | — | — | 0 | 1 | — | — | — |
| Pin Function | A19 output pin | PA3 input pin | PA3 output pin | SCK2 output pin | SCK2 output pin | SCK2 input pin | PA3 input pin | PA3 output pin | SCK2 output pin | SCK2 output pin | SCK2 input pin |

**Table 8.30 PA2 Pin Function**

| Operating mode | Modes 4 to 6 | | | | Mode 7 | | |
|---|---|---|---|---|---|---|---|
| AE3 to AE0 | B'1011 or B'11×× | Other than B'1011 or B'11×× | | | — | | |
| RE in SCR_2 | — | 0 | | 1 | 0 | | 1 |
| PA2DDR | — | 0 | 1 | — | 0 | 1 | — |
| Pin Function | A18 output pin | PA2 input pin | PA2 output pin | RxD2 input pin | PA2 input pin | PA2 output pin | RxD2 input pin |

Legend:
×: Don't care.

**Table 8.31 PA1 Pin Function**

| Operating mode | Modes 4 to 6 | | | | Mode 7 | | |
|---|---|---|---|---|---|---|---|
| AE3 to AE0 | B'101× or B'11×× | Other than B'101× or B'11×× | | | — | | |
| TE in SCR_2 | — | 0 | | 1 | 0 | | 1 |
| PA1DDR | — | 0 | 1 | — | 0 | 1 | — |
| Pin Function | A17 output pin | PA1 input pin | PA1 output pin | TxD2 output pin | PA1 input pin | PA1 output pin | TxD2 output pin |

**Table 8.32 PA0 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | Other than B'0××× or B'1000 | B'0××× or B'1000 | | — | |
| PA0DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A16 output pin | PA0 input pin | PA0 output pin | PA0 input pin | PA0 output pin |

Legend:
×: Don't care.

**Pin Functions of H8S/2212 Group**

Port A pins also function as SCI_2 I/O pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.33 PA3 Pin Function**

| CKE1 in SCR_2 | 0 | | | | 1 |
|---|---|---|---|---|---|
| C/$\overline{A}$ in SMR_2 | 0 | | | 1 | — |
| CKE0 in SCR_2 | 0 | | 1 | — | — |
| PA3DDR | 0 | 1 | — | — | — |
| Pin Function | PA3 input pin | PA3 output pin | SCK2 output pin | SCK2 output pin | SCK2 input pin |

**Table 8.34 PA2 Pin Function**

| RE in SCR_2 | 0 | | 1 |
|---|---|---|---|
| PA2DDR | 0 | 1 | — |
| Pin Function | PA2 input pin | PA2 output pin | RxD2 input pin |

**Table 8.35 PA1 Pin Function**

| TE in SCR_2 | 0 | | 1 |
|---|---|---|---|
| PA1DDR | 0 | 1 | — |
| Pin Function | PA1 input pin | PA1 output pin | TxD2 output pin |

### 8.6.7 Port A Input Pull-Up MOS States

The port A has an on-chip input pull-up MOS function that can be controlled by software. The input pull-up MOS can be specified as the on or off state for individual bits.

Table 8.36 summarizes the input pull-up MOS states.

**Table 8.36 Input Pull-Up MOS States (Port A)**

| Pins | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| Address output, port output, SCI output | Off | | Off | | |
| Port input, SCI input | | | On/Off | | |

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when PADDR = 0 and PAPCR = 1; otherwise off.

## 8.7 Port B (H8S/2218 Group Only)

The port B is an 8-bit I/O port also functioning as address bus (A15 to A8) output pins. The port B has the following registers.

Note: When the USB is used while the E6000 emulator is used, the AE3 to AE0 bits in PFCR must be set so that the PB1 and PB0 pins output addresses A9 and A8. This note applies to both the H8S/2218 Group and H8S/2212 Group.

- Port B data direction register (PBDDR)
- Port B data register (PBDR)
- Port B register (PORTB)
- Port B pull-up MOS control register (PBPCR)

### 8.7.1 Port B Data Direction Register (PBDDR)

PBDDR specifies input or output for the pins of the port B.

Since PBDDR is a write-only register, the bit manipulation instructions must not be used to write PBDDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PB7DDR | 0 | W | Modes 4 to 6:<br>If address output is enabled by the setting of bits AE3 to AE0 in PFCR, the corresponding port B pins are address outputs. When address output is disabled, setting a PBDDR bit to 1 makes the corresponding port B pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 6 | PB6DDR | 0 | W | |
| 5 | PB5DDR | 0 | W | |
| 4 | PB4DDR | 0 | W | |
| 3 | PB3DDR | 0 | W | |
| 2 | PB2DDR | 0 | W | |
| 1 | PB1DDR | 0 | W | Mode 7:<br>Setting a PBDDR bit to 1 makes the corresponding port B pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 0 | PB0DDR | 0 | W | |

### 8.7.2 Port B Data Register (PBDR)

PBDR stores output data for the port B pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PB7DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 6 | PB6DR | 0 | R/W | |
| 5 | PB5DR | 0 | R/W | |
| 4 | PB4DR | 0 | R/W | |
| 3 | PB3DR | 0 | R/W | |
| 2 | PB2DR | 0 | R/W | |
| 1 | PB1DR | 0 | R/W | |
| 0 | PB0DR | 0 | R/W | |

### 8.7.3 Port B Register (PORTB)

PORTB indicates the pin states of the port B.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PB7 | —* | R | If the port B is read while PBDDR bits are set to 1, the PBDR value is read. If the port B is read while PBDDR bits are cleared to 0, the pin states are read. |
| 6 | PB6 | —* | R | |
| 5 | PB5 | —* | R | |
| 4 | PB4 | —* | R | |
| 3 | PB3 | —* | R | |
| 2 | PB2 | —* | R | |
| 1 | PB1 | —* | R | |
| 0 | PB0 | —* | R | |

Note: * Determined by the states of pins PB7 to PB0.

### 8.7.4 Port B Pull-Up MOS Control Register (PBPCR)

PBPCR controls the on/off state of the port B input pull-up MOS. PBPCR is valid for port input pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PB7PCR | 0 | R/W | When a pin functions as an input port, setting the corresponding bit to 1 turns on the input pull-up MOS for that pin. |
| 6 | PB6PCR | 0 | R/W | |
| 5 | PB5PCR | 0 | R/W | |
| 4 | PB4PCR | 0 | R/W | |
| 3 | PB3PCR | 0 | R/W | |
| 2 | PB2PCR | 0 | R/W | |
| 1 | PB1PCR | 0 | R/W | |
| 0 | PB0PCR | 0 | R/W | |

### 8.7.5 Pin Functions

Port B pins also function as address bus (A15 to A9) output pins. The correspondence between the register specification and the pin functions is shown below.

Note: When using the USB with the emulator (E6000), set A9 and A8 as address bus output pins.

**Table 8.37 PB7 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | B'1×××| Other than B'1××× | | — | |
| PB7DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A15 output pin | PB7 input pin | PB7 output pin | PB7 input pin | PB7 output pin |

**Table 8.38 PB6 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | B'0111 or B'1××× | Other than B'0111 or B'1××× | | — | |
| PB6DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A14 output pin | PB6 input pin | PB6 output pin | PB6 input pin | PB6 output pin |

**Table 8.39 PB5 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | B'011× or B'1××× | Other than B'011× or B'1××× | | — | |
| PB5DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A13 output pin | PB5 input pin | PB5 output pin | PB5 input pin | PB5 output pin |

**Table 8.40 PB4 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | Other than B'0100 or B'00×× | B'0100 or B'00×× | | — | |
| PB4DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A12 output pin | PB4 input pin | PB4 output pin | PB4 input pin | PB4 output pin |

**Table 8.41 PB3 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | Other than B'00×× | B'00×× | | — | |
| PB3DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A11 output pin | PB3 input pin | PB3 output pin | PB3 input pin | PB3 output pin |

**Table 8.42 PB2 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | Other than B'0010 or B'000× | B'0010 or B'000× | | — | |
| PB2DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A10 output pin | PB2 input pin | PB2 output pin | PB2 input pin | PB2 output pin |

**Table 8.43 PB1 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | Other than B'000× | B'000× | | — | |
| PB1DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A9 output pin | PB1 input pin | PB1 output pin | PB1 input pin | PB1 output pin |

**Table 8.44 PB0 Pin Function**

| Operating mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| AE3 to AE0 | Other than B'0000 | B'0000 | | — | |
| PB0DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A8 output pin | PB0 input pin | PB0 output pin | PB0 input pin | PB0 output pin |

Legend:

×: Don't care.

### 8.7.6 Port B Input Pull-Up MOS States

The port B has an on-chip input pull-up MOS function that can be controlled by software. The input pull-up MOS can be specified as the on or off state for individual bits.

Table 8.45 summarizes the input pull-up MOS states.

**Table 8.45 Input Pull-Up MOS States (Port B)**

| Pins | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| Address output, port output | Off | | Off | | |
| Port input | | | On/Off | | |

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when PBDDR = 0 and PBPCR = 1; otherwise off.

## 8.8 Port C (H8S/2218 Group Only)

The port C is an 8-bit I/O port also functioning as address bus (A7 to A0) output pins. The port C has the following registers.

Note: When the RTC and USB are used while the E6000 emulator is used, the PC7DDR to PC0DDR bits in PCDDR must be set so that the PC7 to PC0 pins output addresses A7 to A0. This note applies to both the H8S/2218 Group and H8S/2212 Group.

- Port C data direction register (PCDDR)
- Port C data register (PCDR)
- Port C register (PORTC)
- Port C pull-up MOS control register (PCPCR)

### 8.8.1 Port C Data Direction Register (PCDDR)

PCDDR specifies input or output for the pins of the port C.

Since PCDDR is a write-only register, the bit manipulation instructions must not be used to write PCDDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PC7DDR | 0 | W | Modes 4 and 5:<br>Port C pins are address output pins. |
| 6 | PC6DDR | 0 | W | |
| 5 | PC5DDR | 0 | W | Mode 6:<br>Setting a PCDDR bit to 1 makes the corresponding port C pin an address output pin, while clearing the bit to 0 makes the pin an input port. |
| 4 | PC4DDR | 0 | W | |
| 3 | PC3DDR | 0 | W | |
| 2 | PC2DDR | 0 | W | Mode 7:<br>Setting a PCDDR bit to 1 makes the corresponding port C pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 1 | PC1DDR | 0 | W | |
| 0 | PC0DDR | 0 | W | |

### 8.8.2 Port C Data Register (PCDR)

PCDR stores output data for the port C pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PC7DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 6 | PC6DR | 0 | R/W | |
| 5 | PC5DR | 0 | R/W | |
| 4 | PC4DR | 0 | R/W | |
| 3 | PC3DR | 0 | R/W | |
| 2 | PC2DR | 0 | R/W | |
| 1 | PC1DR | 0 | R/W | |
| 0 | PC0DR | 0 | R/W | |

### 8.8.3 Port C Register (PORTC)

PORTC indicates the pin states of the port C.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PC7 | —* | R | If the port C is read while PCDDR bits are set to 1, the PCDR value is read. If the port C is read while PCDDR bits are cleared to 0, the pin states are read. |
| 6 | PC6 | —* | R | |
| 5 | PC5 | —* | R | |
| 4 | PC4 | —* | R | |
| 3 | PC3 | —* | R | |
| 2 | PC2 | —* | R | |
| 1 | PC1 | —* | R | |
| 0 | PC0 | —* | R | |

Note: * Determined by the states of pins PC7 to PC0.

### 8.8.4 Port C Pull-Up MOS Control Register (PCPCR)

PCPCR controls the on/off state of the port C input pull-up MOS.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PC7PCR | 0 | R/W | When a pin functions as an input port, setting the corresponding bit to 1 turns on the input pull-up MOS for that pin. |
| 6 | PC6PCR | 0 | R/W | |
| 5 | PC5PCR | 0 | R/W | |
| 4 | PC4PCR | 0 | R/W | |
| 3 | PC3PCR | 0 | R/W | |
| 2 | PC2PCR | 0 | R/W | |
| 1 | PC1PCR | 0 | R/W | |
| 0 | PC0PCR | 0 | R/W | |

### 8.8.5 Pin Functions

Port C pins also function as address bus (A7 to A0) output pins. The correspondence between the register specification and the pin functions is shown below.

Note: When using the RTC and USB with the emulator (E6000), set A7 to A0 as address bus output pins.

**Table 8.46 PC7 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC7DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A7 output pin | PC7 input pin | A7 output pin | PC7 input pin | PC7 output pin |

**Table 8.47 PC6 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC6DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A6 output pin | PC6 input pin | A6 output pin | PC6 input pin | PC6 output pin |

**Table 8.48 PC5 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC5DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A5 output pin | PC5 input pin | A5 output pin | PC5 input pin | PC5 output pin |

**Table 8.49 PC4 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC4DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A4 output pin | PC4 input pin | A4 output pin | PC4 input pin | PC4 output pin |

**Table 8.50 PC3 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC3DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A3 output pin | PC3 input pin | A3 output pin | PC3 input pin | PC3 output pin |

**Table 8.51 PC2 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC2DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A2 output pin | PC2 input pin | A2 output pin | PC2 input pin | PC2 output pin |

**Table 8.52 PC1 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC1DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A1 output pin | PC1 input pin | A1 output pin | PC1 input pin | PC1 output pin |

**Table 8.53 PC0 Pin Function**

| Operating Mode | Modes 4 and 5 | Mode 6 | | Mode 7 | |
|---|---|---|---|---|---|
| PC0DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | A0 output pin | PC0 input pin | A0 output pin | PC0 input pin | PC0 output pin |

### 8.8.6 Port C Input Pull-Up MOS States

The port C has an on-chip input pull-up MOS function that can be controlled by software. The input pull-up MOS can be used in modes 6 and 7, and can be specified as the on or off state for individual bits.

Table 8.54 summarizes the input pull-up MOS states.

**Table 8.54 Input Pull-Up MOS States (Port C)**

| Pins | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| Address output (modes 4 and 5), port output (modes 6 and 7) | Off | | Off | | |
| Port input (modes 6 and 7) | | | On/Off | | |

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when PCDDR = 0 and PCPCR = 1; otherwise off.

## 8.9 Port D (H8S/2218 Group Only)

The port D is an 8-bit I/O port also functioning as data bus (D15 to D8) I/O pins. The port D has the following registers.

- Port D data direction register (PDDDR)
- Port D data register (PDDR)
- Port D register (PORTD)
- Port D pull-up MOS control register (PDPCR)

### 8.9.1 Port D Data Direction Register (PDDDR)

PDDDR specifies input or output for the pins of the port D.

Since PDDDR is a write-only register, the bit manipulation instructions must not be used to write PDDDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PD7DDR | 0 | W | Modes 4 to 6:<br>Port D pins automatically function as data input/output pins.<br>Mode 7:<br>Setting a PDDDR bit to 1 makes the corresponding port D pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 6 | PD6DDR | 0 | W | |
| 5 | PD5DDR | 0 | W | |
| 4 | PD4DDR | 0 | W | |
| 3 | PD3DDR | 0 | W | |
| 2 | PD2DDR | 0 | W | |
| 1 | PD1DDR | 0 | W | |
| 0 | PD0DDR | 0 | W | |

### 8.9.2 Port D Data Register (PDDR)

PDDR stores output data for the port D pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PD7DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 6 | PD6DR | 0 | R/W | |
| 5 | PD5DR | 0 | R/W | |
| 4 | PD4DR | 0 | R/W | |
| 3 | PD3DR | 0 | R/W | |
| 2 | PD2DR | 0 | R/W | |
| 1 | PD1DR | 0 | R/W | |
| 0 | PD0DR | 0 | R/W | |

### 8.9.3 Port D Register (PORTD)

PORTD indicates the pin states of the port D.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PD7 | —* | R | If the port D is read while PDDDR bits are set to 1, the PDDR value is read. If the port D is read while PDDDR bits are cleared to 0, the pin states are read.<br>Note: After accessing EXMDLSTP or the RTC register (address range: H'FFFF40 to H'FFFF5F), you must perform a dummy read to the external address space (such as H'FFEF00 to H'FF7FF) outside the range H'FFFF40 to H'FFFF5F before reading PORTD. |
| 6 | PD6 | —* | R | |
| 5 | PD5 | —* | R | |
| 4 | PD4 | —* | R | |
| 3 | PD3 | —* | R | |
| 2 | PD2 | —* | R | |
| 1 | PD1 | —* | R | |
| 0 | PD0 | —* | R | |

Note: * Determined by the states of pins PD7 to PD0.

### 8.9.4 Port D Pull-Up MOS Control Register (PDPCR)

PDPCR controls the on/off state of the port D input pull-up MOS.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PD7PCR | 0 | R/W | When a pin functions as an input port, setting the corresponding bit to 1 turns on the input pull-up MOS for that pin. |
| 6 | PD6PCR | 0 | R/W | |
| 5 | PD5PCR | 0 | R/W | |
| 4 | PD4PCR | 0 | R/W | |
| 3 | PD3PCR | 0 | R/W | |
| 2 | PD2PCR | 0 | R/W | |
| 1 | PD1PCR | 0 | R/W | |
| 0 | PD0PCR | 0 | R/W | |

### 8.9.5 Pin Functions

Port D pins also function as data bus (D15 to D8) I/O pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.55 PD7 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD7DDR | — | 0 | 1 |
| Pin Function | D15 input/output pin | PD7 input pin | PD7 output pin |

**Table 8.56 PD6 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD6DDR | — | 0 | 1 |
| Pin Function | D14 input/output pin | PD6 input pin | PD6 output pin |

**Table 8.57 PD5 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD5DDR | — | 0 | 1 |
| Pin Function | D13 input/output pin | PD5 input pin | PD5 output pin |

**Table 8.58 PD4 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD4DDR | — | 0 | 1 |
| Pin Function | D12 input/output pin | PD4 input pin | PD4 output pin |

**Table 8.59 PD3 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD3DDR | — | 0 | 1 |
| Pin Function | D11 input/output pin | PD3 input pin | PD3 output pin |

**Table 8.60 PD2 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD2DDR | — | 0 | 1 |
| Pin Function | D10 input/output pin | PD2 input pin | PD2 output pin |

**Table 8.61 PD1 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD1DDR | — | 0 | 1 |
| Pin Function | D9 input/output pin | PD1 input pin | PD1 output pin |

**Table 8.62 PD0 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PD0DDR | — | 0 | 1 |
| Pin Function | D8 input/output pin | PD0 input pin | PD0 output pin |

### 8.9.6 Port D Input Pull-Up MOS States

The port D has an on-chip input pull-up MOS function that can be controlled by software. The input pull-up MOS can be used in mode 7, and can be specified as the on or off state for individual bits.

Table 8.63 summarizes the input pull-up MOS states.

**Table 8.63 Input Pull-Up MOS States (Port D)**

| Pins | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| Data input/output (modes 4 to 6), port output (mode 7) | Off | | Off | | |
| Port input (mode 7) | | | On/Off | | |

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when PDDDR = 0 and PDPCR = 1; otherwise off.

## 8.10 Port E

The port E is an 8-bit I/O port also functioning as data bus (D7 to D0) I/O pins. The port E has the following registers.

- Port E data direction register (PEDDR)
- Port E data register (PEDR)
- Port E register (PORTE)
- Port E pull-up MOS control register (PEPCR)

### 8.10.1 Port E Data Direction Register (PEDDR)

PEDDR specifies input or output for the pins of the port E.

Since PEDDR is a write-only register, the bit manipulation instructions must not be used to write PEDDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PE7DDR | 0 | W | (H8S/2218 Group) |
| 6 | PE6DDR | 0 | W | Modes 4 to 6: |
| 5 | PE5DDR | 0 | W | When 8-bit bus mode is selected, port E functions as an I/O port. Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 4 | PE4DDR | 0 | W | |
| 3 | PE3DDR | 0 | W | |
| 2 | PE2DDR | 0 | W | When 16-bit bus mode is selected, the input/output direction settings in PEDDR are ignored, and port E pins automatically function as data input/output pins. For details on 8-bit/16-bit bus mode, refer to section 6, Bus Controller. |
| 1 | PE1DDR | 0 | W | |
| 0 | PE0DDR | 0 | W | |
| | | | | Mode 7:<br>Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port. |
| | | | | (H8S/2212 Group) |
| | | | | Setting a PEDDR bit to 1 makes the corresponding port E pin an output port, while clearing the bit to 0 makes the pin an input port. |

### 8.10.2 Port E Data Register (PEDR)

PEDR stores output data for the port E pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PE7DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 6 | PE6DR | 0 | R/W | |
| 5 | PE5DR | 0 | R/W | |
| 4 | PE4DR | 0 | R/W | |
| 3 | PE3DR | 0 | R/W | |
| 2 | PE2DR | 0 | R/W | |
| 1 | PE1DR | 0 | R/W | |
| 0 | PE0DR | 0 | R/W | |

### 8.10.3 Port E Register (PORTE)

PORTE indicates the pin states of the port E.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PE7 | —* | R | If the port E is read while PEDDR bits are set to 1, the PEDR value is read. If the port E is read while PEDDR bits are cleared to 0, the pin states are read. |
| 6 | PE6 | —* | R | |
| 5 | PE5 | —* | R | |
| 4 | PE4 | —* | R | |
| 3 | PE3 | —* | R | |
| 2 | PE2 | —* | R | |
| 1 | PE1 | —* | R | |
| 0 | PE0 | —* | R | |

Note: * Determined by the states of pins PE7 to PE0.

### 8.10.4 Port E Pull-Up MOS Control Register (PEPCR)

PEPCR controls the on/off state of the port E input pull-up MOS.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PE7PCR | 0 | R/W | When a pin functions as an input port, setting the corresponding bit to 1 turns on the input pull-up MOS for that pin. |
| 6 | PE6PCR | 0 | R/W | |
| 5 | PE5PCR | 0 | R/W | |
| 4 | PE4PCR | 0 | R/W | |
| 3 | PE3PCR | 0 | R/W | |
| 2 | PE2PCR | 0 | R/W | |
| 1 | PE1PCR | 0 | R/W | |
| 0 | PE0PCR | 0 | R/W | |

### 8.10.5 Pin Functions

#### Pin Functions of H8S/2218 Group

Port E pins also function as data bus (D7 to D0) I/O pins. The correspondence between the register specification and the pin function is shown below.

**Table 8.64 PE7 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE7DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE7 input pin | PE7 output pin | D7 input/output pin | PE7 input pin | PE7 output pin |

**Table 8.65 PE6 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE6DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE6 input pin | PE6 output pin | D6 input/output pin | PE6 input pin | PE6 output pin |

**Table 8.66 PE5 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE5DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE5 input pin | PE5 output pin | D5 input/output pin | PE5 input pin | PE5 output pin |

**Table 8.67 PE4 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE4DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE4 input pin | PE4 output pin | D4 input/output pin | PE4 input pin | PE4 output pin |

**Table 8.68 PE3 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE3DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE3 input pin | PE3 output pin | D3 input/output pin | PE3 input pin | PE3 output pin |

**Table 8.69 PE2 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE2DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE2 input pin | PE2 output pin | D2 input/output pin | PE2 input pin | PE2 output pin |

**Table 8.70 PE1 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE1DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE1 input pin | PE1 output pin | D1 input/output pin | PE1 input pin | PE1 output pin |

**Table 8.71 PE0 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 8-bit bus mode | | 16-bit bus mode | — | |
| PE0DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PE0 input pin | PE0 output pin | D0 input/output pin | PE0 input pin | PE0 output pin |

**Pin Functions of H8S/2212 Group**

The port E function as a general I/O port. The correspondence between the register specification and the pin function is shown below.

**Table 8.72 PE7 Pin Function**

| PE7DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE7 input pin | PE7 output pin |

**Table 8.73 PE6 Pin Function**

| PE6DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE6 input pin | PE6 output pin |

**Table 8.74 PE5 Pin Function**

| PE5DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE5 input pin | PE5 output pin |

**Table 8.75 PE4 Pin Function**

| PE4DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE4 input pin | PE4 output pin |

**Table 8.76 PE3 Pin Function**

| PE3DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE3 input pin | PE3 output pin |

**Table 8.77 PE2 Pin Function**

| PE2DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE2 input pin | PE2 output pin |

**Table 8.78 PE1 Pin Function**

| PE1DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE1 input pin | PE1 output pin |

**Table 8.79 PE0 Pin Function**

| PE0DDR | 0 | 1 |
|---|---|---|
| Pin Function | PE0 input pin | PE0 output pin |

### 8.10.6 Port E Input Pull-Up MOS States

The port E has an on-chip input pull-up MOS function that can be controlled by software. The input pull-up MOS can be used in modes 4 to 6 and in 8-bit bus mode, or in mode 7, and can be specified as the on or off state for individual bits.

Table 8.80 summarizes the input pull-up MOS states.

**Table 8.80 Input Pull-Up MOS States (Port E)**

| Pins | Power-On Reset | Hardware Standby Mode | Manual Reset | Software Standby Mode | In Other Operations |
|---|---|---|---|---|---|
| Data input/output (16-bit bus mode in modes 4 to 6), port output (8-bit bus mode in modes 4 to 6, mode 7) | Off | | Off | | |
| Port input (8-bit bus mode in modes 4 to 6, mode 7) | | | On/Off | | |

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when PEDDR = 0 and PEPCR = 1; otherwise off.

## 8.11 Port F

In the H8S/2218 Group, the port F is an 8-bit I/O port also functioning as external interrupt input ($\overline{\text{IRQ2}}$, $\overline{\text{IRQ3}}$) pins, bus control signal I/O pins, and system clock output pins. In the H8S/2212 Group, the port F is a 3-bit I/O port also functioning as external interrupt input ($\overline{\text{IRQ2}}$, $\overline{\text{IRQ3}}$) pins and system clock output pins. The port F has the following registers.

- Port F data direction register (PFDDR)
- Port F data register (PFDR)
- Port F register (PORTF)

### 8.11.1 Port F Data Direction Register (PFDDR)

PFDDR specifies input or output for the pins of the port F.

Since PFDDR is a write-only register, the bit manipulation instructions must not be used to write PFDDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PF7DDR | 1/0*[1] | W | (H8S/2218 Group) |
| 6 | PF6DDR*[2] | 0 | W | Modes 4 to 6:<br>Pin PF7 functions as the φ output pin when the PF7DDR bit is set to 1, and as an input port when the bit is cleared to 0. Pins PF6 to PF3 are automatically designated as bus control output pins. Pins PF2 to PF0 are made bus control input/output pins by bus controller settings. Otherwise, setting a PFDDR bit to 1 makes the corresponding pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 5 | PF5DDR*[2] | 0 | W | |
| 4 | PF4DDR*[2] | 0 | W | |
| 3 | PF3DDR | 0 | W | |
| 2 | PF2DDR*[2] | 0 | W | |
| 1 | PF1DDR*[2] | 0 | W | |
| 0 | PF0DDR | 0 | W | |
| | | | | Mode 7<br>Setting a PFDDR bit to 1 makes the corresponding port F pin PF6 to PF0 an output port, or in the case of pin PF7, the φ output pin. Clearing the bit to 0 makes the pin an input port.<br>(H8S/2212 Group)<br>Setting a PFDDR bit to 1 makes the corresponding port F pin PF6 to PF0 an output port, or in the case of pin PF7, the φ output pin. Clearing the bit to 0 makes the pin an input port. |

Notes: 1. The initial value becomes 1 in modes 4 to 6 and 0 in mode 7.
2. Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.11.2 Port F Data Register (PFDR)

PFDR stores output data for the port F pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PF7DR | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 6 | PF6DR* | 0 | R/W | |
| 5 | PF5DR* | 0 | R/W | |
| 4 | PF4DR* | 0 | R/W | |
| 3 | PF3DR | 0 | R/W | |
| 2 | PF2DR* | 0 | R/W | |
| 1 | PF1DR* | 0 | R/W | |
| 0 | PF0DR | 0 | R/W | |

Note: * Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.11.3 Port F Register (PORTF)

PORTF indicates the pin states of the port F.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PF7 | —*[1] | R | If the port F is read while PFDDR bits are set to 1, the PFDR value is read. If the port F is read while PFDDR bits are cleared to 0, the pin states are read. |
| 6 | PF6*[2] | —*[1] | R | |
| 5 | PF5*[2] | —*[1] | R | |
| 4 | PF4*[2] | —*[1] | R | |
| 3 | PF3 | —*[1] | R | |
| 2 | PF2*[2] | —*[1] | R | |
| 1 | PF1*[2] | —*[1] | R | |
| 0 | PF0 | —*[1] | R | |

Notes: 1. Determined by the states of pins PF7 to PF0.
2. Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read.

### 8.11.4 Clock Output Control Register (OUTCR)

OUTCR specifies the clock frequency output from the PF7 pin.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 3 | — | Undefined | — | Reserved<br>The write value should always be 0. |
| 2 | PF7OUT2 | 0 | R/W | PF7 Pin Output Select 2 to 0 |
| 1 | PF7OUT1 | 0 | R/W | 000: Main oscillation clock |
| 0 | PF7OUT0 | 0 | R/W | 001: Outputs 1/2 of main oscillation clock<br>010: Outputs 1/3 of main oscillation clock<br>011: Outputs 1/4 of main oscillation clock<br>1xx: Reserved<br>This function is not supported by the E6000 emulator.<br>$\phi$ in section 22, Electrical Characteristics is in the case when PF7OUT2 to PF7OUT0 = 000. |

### 8.11.5 Pin Functions

#### Pin Functions of H8S/2218 Group

The port F is an 8-bit I/O port. Port F pins also function as external interrupt input ($\overline{IRQ2}$, $\overline{IRQ3}$) pins, bus control signal I/O pins, and system clock output ($\phi$) pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.81 PF7 Pin Function**

| PF7DDR | 0 | 1 | | | |
|---|---|---|---|---|---|
| PF7OUT2 to PF7OUT0 | — | B'000 | B'001 | B'010 | B'011 |
| Pin Function | PF7 input pin | $\phi$ output pin | $\phi$/2 output pin | $\phi$/3 output pin | $\phi$/4 output pin |

**Table 8.82 PF6 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PF6DDR | — | 0 | 1 |
| Pin Function | $\overline{AS}$ output pin | PF6 input pin | PF6 output pin |

**Table 8.83 PF5 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PF5DDR | — | 0 | 1 |
| Pin Function | $\overline{RD}$ output pin | PF5 input pin | PF5 output pin |

**Table 8.84 PF4 Pin Function**

| Operating Mode | Modes 4 to 6 | Mode 7 | |
|---|---|---|---|
| PF4DDR | — | 0 | 1 |
| Pin Function | $\overline{HWR}$ output pin | PF4 input pin | PF4 output pin |

**Table 8.85 PF3 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| Bus Mode | 16-bit bus mode | 8-bit bus mode | | — | |
| PF3DDR | — | 0 | 1 | 0 | 1 |
| Pin Function | $\overline{LWR}$ output pin | PF3 input pin | PF3 output pin | PF3 input pin | PF3 output pin |
| | | $\overline{ADTRG}$ input pin*[1] | | | |
| | | $\overline{IRQ3}$ input pin*[2] | | | |

Notes: 1. $\overline{ADTRG}$ input pin when TRGS0 = TRGS1 = 1.

2. When this pin is used as an external interrupt input pin, this pin must not be used as an I/O pin for another function.

**Table 8.86 PF2 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| WAITE | 0 | | 1 | — | |
| PF2DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PF2 input pin | PF2 output pin | $\overline{WAIT}$ input pin | PF2 input pin | PF2 output pin |

**Table 8.87 PF1 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| BRLE | 0 | | 1 | — | |
| PF1DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PF1 input pin | PF1 output pin | $\overline{\text{BACK}}$ output pin | PF1 input pin | PF1 output pin |

**Table 8.88 PF0 Pin Function**

| Operating Mode | Modes 4 to 6 | | | Mode 7 | |
|---|---|---|---|---|---|
| BRLE | 0 | | 1 | — | |
| PF0DDR | 0 | 1 | — | 0 | 1 |
| Pin Function | PF0 input pin | PF0 output pin | $\overline{\text{BREQ}}$ input pin | PF0 input pin | PF0 output pin |
| | $\overline{\text{IRQ2}}$ input pin* | | | | |

Note: * When this pin is used as an external interrupt input pin, this pin must not be used as an I/O pin for another function.

**Pin Functions of H8S/2212 Group**

The port F is a 3-bit I/O port. Port F pins also function as external interrupt input ($\overline{\text{IRQ2}}$, $\overline{\text{IRQ3}}$) pins and system clock output ($\phi$) pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.89 PF7 Pin Function**

| PF7DDR | 0 | 1 | | | |
|---|---|---|---|---|---|
| PF7OUT2 to PF7OUT0 | — | B'000 | B'001 | B'010 | B'011 |
| Pin Function | PF7 input pin | $\phi$ output pin | $\phi$/2 output pin | $\phi$/3 output pin | $\phi$/4 output pin |

**Table 8.90 PF3 Pin Function**

| PF3DDR | 0 | 1 |
|---|---|---|
| Pin Function | PF3 input pin | PF3 output pin |
| | $\overline{\text{ADTRG}}$ input pin*[1] | |
| | $\overline{\text{IRQ3}}$ input pin*[2] | |

Notes: 1. $\overline{\text{ADTRG}}$ input pin when TRGS0 = TRGS1 = 1.
2. When this pin is used as an external interrupt input pin, this pin must not be used as an I/O pin for another function.

**Table 8.91 PF0 Pin Function**

| PF0DDR | 0 | 1 |
|---|---|---|
| Pin Function | PF0 input pin | PF0 output pin |
| | $\overline{\text{IRQ2}}$ input pin* | |

Note: * When this pin is used as an external interrupt input pin, this pin must not be used as an I/O pin for another function.

## 8.12 Port G

In the H8S/2218 Group, the port G is a 4-bit I/O port also functioning as external interrupt input ($\overline{\text{IRQ7}}$) pins and bus control output ($\overline{\text{CS0}}$ to $\overline{\text{CS3}}$) pins. In the H8S/2212 Group, the port G is a 2-bit I/O port also functioning as external interrupt input ($\overline{\text{IRQ7}}$) pins and H-UDI (TDI) pins. The port G has the following registers.

- Port G data direction register (PGDDR)
- Port G data register (PGDR)
- Port G register (PORTG)

### 8.12.1 Port G Data Direction Register (PGDDR)

PGDDR specifies input or output for the pins of the port G.

Since PGDDR is a write-only register, the bit manipulation instructions must not be used to write PGDDR. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 5 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 4 | PG4DDR*2 | 0/1*1 | W | (H8S/2218 Group)<br>Modes 4 to 6:<br>Setting a PGDDR bit to 1 makes the PG4 to PG1 pins bus control signal output pins, while clearing the bit to 0 makes the pins input ports.<br>Mode 7:<br>Setting a PGDDR bit to 1 makes the corresponding port G pin an output port, while clearing the bit to 0 makes the pin an input port.<br>(H8S/2212 Group)<br>Setting a PG1DDR bit to 1 makes the corresponding port G pin an output port, while clearing the bit to 0 makes the pin an input port. |
| 3 | PG3DDR*2 | 0 | W | |
| 2 | PG2DDR*2 | 0 | W | |
| 1 | PG1DDR | 0 | W | |
| 0 | PG0DDR*3 | 0 | W | (H8S/2212 Group)<br>When EMLE = 1: Pin PG0 function as the H-UDI (TDI) pin.<br>When EMLE = 0: If a PG0DDR bit is set to 1, pin PG0 function as output ports. If a PG0DDR bit is cleared to 0, pin PG0 function as input ports. |

Notes: 1. The initial value becomes 1 in modes 4 and 5 and 0 in modes 6 and 7.
2. Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.
3. Reserved in the H8S/2218 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.12.2 Port G Data Register (PGDR)

PGDR stores output data for the port G pins.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 5 | — | Undefined | — | Reserved<br>These bits are undefined and cannot be modified. |
| 4 | PG4DR*[1] | 0 | R/W | Store output data for a pin that functions as a general output port. |
| 3 | PG3DR*[1] | 0 | R/W | |
| 2 | PG2DR*[1] | 0 | R/W | |
| 1 | PG1DR | 0 | R/W | |
| 0 | PG0DR*[2] | 0 | R/W | |

Notes: 1. Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.
2. Reserved in the H8S/2218 Group. If this bit is read, an undefined value will be read. This bit cannot be modified.

### 8.12.3 Port G Register (PORTG)

PORTG indicates the pin states of the port G.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 5 | — | Undefined | — | Reserved<br>These bits are undefined. |
| 4 | PG4*[2] | —*[1] | R | If the port G is read while PGDDR bits are set to 1, the PGDR value is read. If the port G is read while PGDDR bits are cleared to 0, the pin states are read. |
| 3 | PG3*[2] | —*[1] | R | |
| 2 | PG2*[2] | —*[1] | R | |
| 1 | PG1 | —*[1] | R | |
| 0 | PG0*[3] | —*[1] | R | |

Notes: 1. Determined by the states of pins PG4 to PG0.
2. Reserved in the H8S/2212 Group. If this bit is read, an undefined value will be read.
3. Reserved in the H8S/2218 Group. If this bit is read, an undefined value will be read.

### 8.12.4 Pin Functions

**Pin Functions of H8S/2218 Group**

Port G pins also function as external interrupt input ($\overline{IRQ7}$) pins and bus control signal output ($\overline{CS0}$ to $\overline{CS3}$) pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.92 PG4 Pin Function**

| Operating Mode | Modes 4 to 6 | | Mode 7 | |
|---|---|---|---|---|
| PG4DDR | 0 | 1 | 0 | 1 |
| Pin Function | PG4 input pin | $\overline{CS0}$ output pin | PG4 input pin | PG4 output pin |

**Table 8.93 PG3 Pin Function**

| Operating Mode | Modes 4 to 6 | | Mode 7 | |
|---|---|---|---|---|
| PG3DDR | 0 | 1 | 0 | 1 |
| Pin Function | PG3 input pin | $\overline{CS1}$ output pin | PG3 input pin | PG3 output pin |

**Table 8.94 PG2 Pin Function**

| Operating Mode | Modes 4 to 6 | | Mode 7 | |
|---|---|---|---|---|
| PG2DDR | 0 | 1 | 0 | 1 |
| Pin Function | PG2 input pin | $\overline{CS2}$ output pin | PG2 input pin | PG2 output pin |

**Table 8.95 PG1 Pin Function**

| Operating Mode | Modes 4 to 6 | | Mode 7 | |
|---|---|---|---|---|
| PG1DDR | 0 | 1 | 0 | 1 |
| Pin Function | PG1 input pin | $\overline{CS3}$ output pin | PG1 input pin | PG1 output pin |
| | $\overline{IRQ7}$ input pin* | | | |

Note: * When this pin is used as an external interrupt input pin, this pin must not be used as an I/O pin for another function.

**Pin Functions of H8S/2212 Group**

Port G pins also function as external interrupt input ($\overline{\text{IRQ7}}$) pins and H-UDI (TDI) pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.96 PG1 Pin Function**

| PG1DDR | 0 | 1 |
|---|---|---|
| Pin Function | PG1 input pin | PG1 output pin |
| | $\overline{\text{IRQ7}}$ input pin* | |

Note: * When this pin is used as an external interrupt input pin, this pin must not be used as an I/O pin for another function.

**Table 8.97 PG0 Pin Function**

| EMLE | 0 | | 1 |
|---|---|---|---|
| PG0DDR | 0 | 1 | — |
| Pin Function | PG0 input pin | PG0 output pin | TDI input pin |

## 8.13 Handling of Unused Pins

Unused input pins should be fixed high or low. Generally, the input pins of CMOS products are high-impedance. Leaving unused pins open can cause the generation of intermediate levels due to peripheral noise induction. This can result in shoot-through current inside the device and cause it to malfunction. Table 8.98 lists examples of ways to handle unused pins. For the handling of dedicated boundary scan pins that are unused, see section 13.2, Pin Configuration, and section 13.5, Usage Notes. Pins marked NC should be left open. For the handling of dedicated USB pins that are unused, see section 14.8.14, Pin Processing when USB Not Used.

**Table 8.98 Examples of Ways to Handle Unused Input Pins**

| Port Name | Pin Handling Example |
|---|---|
| Port 1 | Connect each pin to Vcc (pull-up) or to Vss (pull-down) via a resistor. |
| Port 3 | |
| Port 4 | Connect each pin to AVcc (pull-up) or to AVss (pull-down) via a resistor. |
| Port 7 | Connect each pin to Vcc (pull-up) or to Vss (pull-down) via a resistor. |
| Port 9 | Connect each pin to AVcc (pull-up) or to AVss (pull-down) via a resistor. |
| Port A | Connect each pin to Vcc (pull-up) or to Vss (pull-down) via a resistor. |
| Port B* | * Ports B, C, and D apply to the H8S/2218 Group only. |
| Port C* | |
| Port D* | |
| Port E | |
| Port F | |
| Port G | |

# Section 9 16-Bit Timer Pulse Unit (TPU)

This LSI has an on-chip 16-bit timer pulse unit (TPU) that comprises three 16-bit timer channels. The function list of the 16-bit timer unit and its block diagram are shown in table 9.1 and figure 9.1, respectively.

## 9.1 Features

- Maximum 8-pulse input/output
- Selection of 8 counter input clocks for each channel
- The following operations can be set for each channel

  Waveform output at compare match, input capture function, counter clear operation, simultaneous writing to multiple timer counters (TCNT), simultaneous clearing using compare match or input capture, simultaneous input/output for individual registers using counter synchronous operation, PWM output using user-defined duty, up to 7-phase PWM output by combination with synchronous operation
- Buffer operation settable for channel 0
- Phase counting mode settable independently for each of channels 1 and 2
- Fast access via internal 16-bit bus
- 13 interrupt sources
- Automatic transfer of register data
- A/D converter conversion start trigger can be generated
- Module stop mode can be set
- Baud rate clock for the SCI0 can be generated

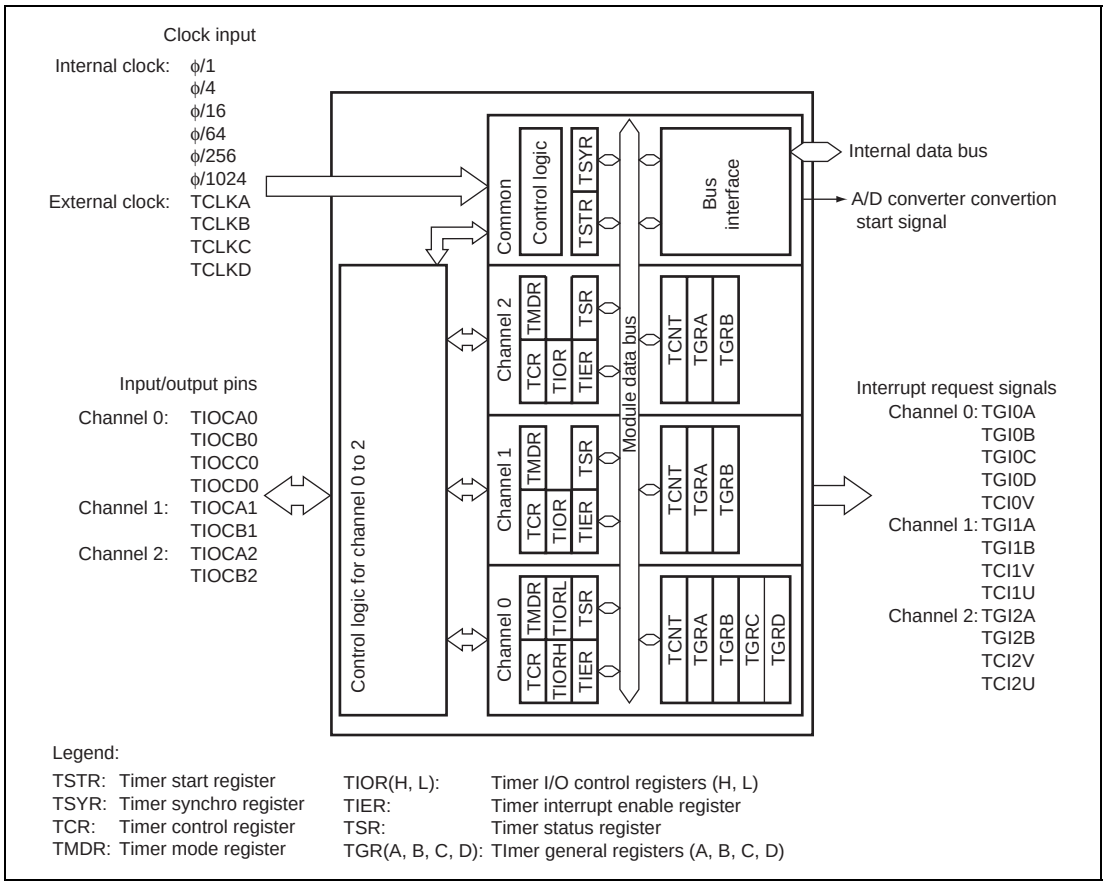



**Figure 9.1 Block Diagram of TPU**

**Table 9.1 TPU Functions**

| Item | | Channel 0 | Channel 1 | Channel 2 |
|---|---|---|---|---|
| Count clock | | φ/1<br>φ/4<br>φ/16<br>φ/64<br>TCLKA<br>TCLKB<br>TCLKC<br>TCLKD | φ/1<br>φ/4<br>φ/16<br>φ/64<br>φ/256<br>TCLKA<br>TCLKB | φ/1<br>φ/4<br>φ/16<br>φ/64<br>φ/1024<br>TCLKA<br>TCLKB<br>TCLKC |
| General registers | | TGRA_0<br>TGRB_0 | TGRA_1<br>TGRB_1 | TGRA_2<br>TGRB_2 |
| General registers/buffer registers | | TGRC_0<br>TGRD_0 | – | – |
| I/O pins | | TIOCA0<br>TIOCB0<br>TIOCC0<br>TIOCD0 | TIOCA1<br>TIOCB1 | TIOCA2<br>TIOCB2 |
| Counter clear function | | TGR compare match or input capture | TGR compare match or input capture | TGR compare match or input capture |
| Compare match output | 0 output | O | O | O |
| | 1 output | O | O | O |
| | Toggle output | O | O | O |
| Input capture function | | O | O | O |
| Synchronous operation | | O | O | O |
| PWM mode | | O | O | O |
| Phase counting mode | | – | O | O |
| Buffer operation | | O | – | – |

| Item | Channel 0 | Channel 1 | Channel 2 |
|---|---|---|---|
| DMAC activation | TGRA_0 compare match or input capture | TGRA_1 compare match or input capture | TGRA_2 compare match or input capture |
| A/D converter trigger | TGRA_0 compare match or input capture | TGRA_1 compare match or input capture | TGRA_2 compare match or input capture |
| Interrupt sources | 5 sources<br>• Compare match or input capture 0A<br>• Compare match or input capture 0B<br>• Compare match or input capture 0C<br>• Compare match or input capture 0D<br>• Overflow | 4 sources<br>• Compare match or input capture 1A<br>• Compare match or input capture 1B<br>• Overflow<br>• Underflow | 4 sources<br>• Compare match or input capture 2A<br>• Compare match or input capture 2B<br>• Overflow<br>• Underflow |

Legend:

O: Possible

–: Not possible

## 9.2 Input/Output Pins

**Table 9.2 Pin Configuration**

| Channel | Symbol | I/O | Function |
|---|---|---|---|
| All | TCLKA | Input | External clock A input pin<br>(Channel 1 phase counting mode A phase input) |
| | TCLKB | Input | External clock B input pin<br>(Channel 1 phase counting mode B phase input) |
| | TCLKC | Input | External clock C input pin<br>(Channel 2 phase counting mode A phase input) |
| | TCLKD | Input | External clock D input pin<br>(Channel 2 phase counting mode B phase input) |
| 0 | TIOCA0 | I/O | TGRA_0 input capture input/output compare output/PWM output pin |
| | TIOCB0 | I/O | TGRB_0 input capture input/output compare output/PWM output pin |
| | TIOCC0 | I/O | TGRC_0 input capture input/output compare output/PWM output pin |
| | TIOCD0 | I/O | TGRD_0 input capture input/output compare output/PWM output pin |
| 1 | TIOCA1 | I/O | TGRA_1 input capture input/output compare output/PWM output pin |
| | TIOCB1 | I/O | TGRB_1 input capture input/output compare output/PWM output pin |
| 2 | TIOCA2 | I/O | TGRA_2 input capture input/output compare output/PWM output pin |
| | TIOCB2 | I/O | TGRA_2 input capture input/output compare output/PWM output pin |

## 9.3 Register Descriptions

The TPU has the following registers.

- Timer control register_0 (TCR_0)
- Timer mode register_0 (TMDR_0)
- Timer I/O control register H_0 (TIORH_0)
- Timer I/O control register L_0 (TIORL_0)
- Timer interrupt enable register_0 (TIER_0)
- Timer status register_0 (TSR_0)
- Timer counter_0 (TCNT_0)
- Timer general register A_0 (TGRA_0)
- Timer general register B_0 (TGRB_0)
- Timer general register C_0 (TGRC_0)
- Timer general register D_0 (TGRD_0)
- Timer control register_1 (TCR_1)
- Timer mode register_1 (TMDR_1)
- Timer I/O control register _1 (TIOR_1)
- Timer interrupt enable register_1 (TIER_1)
- Timer status register_1 (TSR_1)
- Timer counter_1 (TCNT_1)
- Timer general register A_1 (TGRA_1)
- Timer general register B_1 (TGRB_1)
- Timer control register_2 (TCR_2)
- Timer mode register_2 (TMDR_2)
- Timer I/O control register_2 (TIOR_2)
- Timer interrupt enable register_2 (TIER_2)
- Timer status register_2 (TSR_2)
- Timer counter_2 (TCNT_2)
- Timer general register A_2 (TGRA_2)
- Timer general register B_2 (TGRB_2)

Common Registers

- Timer start register (TSTR)
- Timer synchro register (TSYR)

### 9.3.1 Timer Control Register (TCR)

The TCR registers control the TCNT operation for each channel. The TPU has a total of three TCR registers, one for each channel (channel 0 to 2). TCR register settings should be made only when TCNT operation is stopped.

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 7 | CCLR2 | 0 | R/W | Counter Clear 2 to 0 |
| 6 | CCLR1 | 0 | R/W | These bits select the TCNTcounter clearing source. See tables 9.3 and 9.4 for details. |
| 5 | CCLR0 | 0 | R/W | |
| 4 | CKEG1 | 0 | R/W | Clock Edge 1 and 0 |
| 3 | CKEG0 | 0 | R/W | These bits select the input clock edge. When the internal clock is counted using both edges, the input clock frequency is halved (e.g., $\phi$/4 both edges = $\phi$/2 rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority. Internal clock edge selection is valid when the input clock is $\phi$/4 or slower. If $\phi$/1 is selected as the input clock, this setting is ignored and count at falling edge of $\phi$ is selected.<br>00: Count at rising edge<br>01: Count at falling edge<br>1×: Count at both edges<br>Legend:<br>×: Don't care |
| 2 | TPSC2 | 0 | R/W | Time Prescaler 2 to 0 |
| 1 | TPSC1 | 0 | R/W | These bits select the TCNT counter clock. The clock source can be selected independently for each channel. See tables 9.5 to 9.7 for details. |
| 0 | TPSC0 | 0 | R/W | |

**Table 9.3 CCLR2 to CCLR0 (channel 0)**

| | Bit 7 | Bit 6 | Bit 5 | |
|---|---|---|---|---|
| **Channel** | **CCLR2** | **CCLR1** | **CCLR0** | **Description** |
| 0 | 0 | 0 | 0 | TCNT clearing disabled |
| | | | 1 | TCNT cleared by TGRA compare match/input capture |
| | | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*[1] |
| | 1 | 0 | 0 | TCNT clearing disabled |
| | | | 1 | TCNT cleared by TGRC compare match/input capture*[2] |
| | | 1 | 0 | TCNT cleared by TGRD compare match/input capture*[2] |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*[1] |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register. TCNT is not cleared because the buffer register setting has priority, and compare match/input capture dose not occur.

**Table 9.4 CCLR2 to CCLR0 (channels 1 and 2)**

| | Bit 7 | Bit 6 | Bit 5 | |
|---|---|---|---|---|
| **Channel** | **Reserved*[2]** | **CCLR1** | **CCLR0** | **Description** |
| 1, 2 | 0 | 0 | 0 | TCNT clearing disabled |
| | | | 1 | TCNT cleared by TGRA compare match/input capture |
| | | 1 | 0 | TCNT cleared by TGRB compare match/input capture |
| | | | 1 | TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation*[1] |

Notes: 1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
2. Bit 7 is reserved in channels 1 and 2. It is always read as 0 and cannot be modified.

**Table 9.5 TPSC2 to TPSC0 (channel 0)**

| | Bit 2 | Bit 1 | Bit 0 | |
|---|---|---|---|---|
| **Channel** | **TPSC2** | **TPSC1** | **TPSC0** | **Description** |
| 0 | 0 | 0 | 0 | Internal clock: counts on $\phi$/1 |
| | | | 1 | Internal clock: counts on $\phi$/4 |
| | | 1 | 0 | Internal clock: counts on $\phi$/16 |
| | | | 1 | Internal clock: counts on $\phi$/64 |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | | 1 | External clock: counts on TCLKB pin input |
| | | 1 | 0 | External clock: counts on TCLKC pin input |
| | | | 1 | External clock: counts on TCLKD pin input |

**Table 9.6 TPSC2 to TPSC0 (channel 1)**

| | Bit 2 | Bit 1 | Bit 0 | |
|---|---|---|---|---|
| **Channel** | **TPSC2** | **TPSC1** | **TPSC0** | **Description** |
| 1 | 0 | 0 | 0 | Internal clock: counts on $\phi$/1 |
| | | | 1 | Internal clock: counts on $\phi$/4 |
| | | 1 | 0 | Internal clock: counts on $\phi$/16 |
| | | | 1 | Internal clock: counts on $\phi$/64 |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | | 1 | External clock: counts on TCLKB pin input |
| | | 1 | 0 | Internal clock: counts on $\phi$/256 |
| | | | 1 | Setting prohibited |

Note: This setting is ignored when channel 1 is in phase counting mode.

**Table 9.7 TPSC2 to TPSC0 (channel 2)**

| Channel | Bit 2 TPSC2 | Bit 1 TPSC1 | Bit 0 TPSC0 | Description |
|---|---|---|---|---|
| 2 | 0 | 0 | 0 | Internal clock: counts on $\phi$/1 |
| | | | 1 | Internal clock: counts on $\phi$/4 |
| | | 1 | 0 | Internal clock: counts on $\phi$/16 |
| | | | 1 | Internal clock: counts on $\phi$/64 |
| | 1 | 0 | 0 | External clock: counts on TCLKA pin input |
| | | | 1 | External clock: counts on TCLKB pin input |
| | | 1 | 0 | External clock: counts on TCLKC pin input |
| | | | 1 | Internal clock: counts on $\phi$/1024 |

Note: This setting is ignored when channel 1 is in phase counting mode.

### 9.3.2 Timer Mode Register (TMDR)

The TMDR registers are used to set the operating mode for each channel. The TPU has three TMDR registers, one for each channel. TMDR register settings should be made only when TCNT operation is stopped.

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 1 | — | Reserved<br>These bits are always read as 1 and cannot be modified. |
| 5 | BFB | 0 | R/W | Buffer Operation B<br>Specifies whether TGRB is to operate in the normal way, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register. TGRD input capture/output compare is not generation. In channels 1 and 2, which have no TGRD, bit 5 is reserved. It is always read as 0 and cannot be modified.<br>0: TGRB operates normally<br>1: TGRB and TGRD used together for buffer operation |

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 4 | BFA | 0 | R/W | Buffer Operation A<br>Specifies whether TGRA is to operate in the normal way, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated. In channels 1 and 2, which have no TGRC, bit 4 is reserved. It is always read as 0 and cannot be modified.<br>0: TGRA operates normally<br>1: TGRA and TGRC used together for buffer operation |
| 3 | MD3 | 0 | R/W | Modes 3 to 0 |
| 2 | MD2 | 0 | R/W | These bits are used to set the timer operating mode. |
| 1 | MD1 | 0 | R/W | MD3 is a reserved bit. In a write, the write value should always be 0. See table 9.8, for details. |
| 0 | MD0 | 0 | R/W | |

**Table 9.8 MD3 to MD0**

| Bit 3 | Bit2 | Bit 1 | Bit 0 | |
|---|---|---|---|---|
| **MD3*[1]** | **MD2*[2]** | **MD1** | **MD0** | **Description** |
| 0 | 0 | 0 | 0 | Normal operation |
| | | | 1 | Reserved |
| | | 1 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | 1 | 0 | 0 | Phase counting mode 1 |
| | | | 1 | Phase counting mode 2 |
| | | 1 | 0 | Phase counting mode 3 |
| | | | 1 | Phase counting mode 4 |
| 1 | × | × | × | — |

Legend:

×: Don't care

Notes: 1. MD3 is reserved bit. In a write, it should be written with 0.

2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

### 9.3.3 Timer I/O Control Register (TIOR)

The TIOR registers control the TGR registers. The TPU has eight TIOR registers, two each for channels 0, and one each for channels 1 and 2. Care is required since TIOR is affected by the TMDR setting. The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified. When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

- **TIORH_0, TIOR_1, TIOR_2**

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 7 | IOB3 | 0 | R/W | I/O Control B3 to B0 |
| 6 | IOB2 | 0 | R/W | Specify the function of TGRB. |
| 5 | IOB1 | 0 | R/W | |
| 4 | IOB0 | 0 | R/W | |
| 3 | IOA3 | 0 | R/W | I/O Control A3 to A0 |
| 2 | IOA2 | 0 | R/W | Specify the function of TGRA. |
| 1 | IOA1 | 0 | R/W | |
| 0 | IOA0 | 0 | R/W | |

- **TIORL_0**

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 7 | IOD3 | 0 | R/W | I/O Control D3 to D0 |
| 6 | IOD2 | 0 | R/W | Specify the function of TGRD. |
| 5 | IOD1 | 0 | R/W | |
| 4 | IOD0 | 0 | R/W | |
| 3 | IOC3 | 0 | R/W | I/O Control C3 to C0 |
| 2 | IOC2 | 0 | R/W | Specify the function of TGRC. |
| 1 | IOC1 | 0 | R/W | |
| 0 | IOC0 | 0 | R/W | |

**Table 9.9 TIORH_0 (channel 0)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **TGRB_0** | |
| **IOB3** | **IOB2** | **IOB1** | **IOB0** | **Function** | **TIOCB0 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | 0 | 0 | 0 | Input capture register | Capture input source is TIOCB0 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCB0 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCB0 pin<br>Input capture at both edges |
| | 1 | × | × | | Setting prohibited |

Legend:

×: Don’t care

**Table 9.10 TIORH_0 (channel 0)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** | **TGRA_0** | |
| **IOA3** | **IOA2** | **IOA1** | **IOA0** | **Function** | **TIOCA0 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | 0 | 0 | 0 | Input capture register | Capture input source is TIOCA0 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCA0 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCA0 pin<br>Input capture at both edges |
| | 1 | × | × | | Setting prohibited |

Legend:
×: Don’t care

**Table 9.11 TIORL_0 (channel 0)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **TGRD_0 Function** | |
| **IOD3** | **IOD2** | **IOD1** | **IOD0** | | **TIOCD0 Pin Function** |
| 0 | 0 | 0 | 0 | Output Compare register* | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | 0 | 0 | 0 | Input capture register* | Capture input source is TIOCD0 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCD0 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCD0 pin<br>Input capture at both edges |
| | 1 | × | × | | Setting prohibited |

Legend:

×: Don't care

Note: * When the BFB bit in TMDR_0 is set to 1 and TGRD_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

**Table 9.12 TIORL_0 (channel 0)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 3** | **Bit 2** | **Bit 1** | **Bit 1** | **TGRC_0** | |
| **IOC3** | **IOC2** | **IOC1** | **IOC0** | **Function** | **TIOCC0 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register* | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | 0 | 0 | 0 | Input capture register* | Capture input source is TIOCC0 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCC0 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCC0 pin<br>Input capture at both edges |
| | 1 | × | × | | Setting prohibited |

Legend:

×: Don't care

Note: * When the BFA bit in TMDR_0 is set to 1and TGRC_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

**Table 9.13 TIOR_1 (channel 1)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **TGRB_1** | |
| **IOB3** | **IOB2** | **IOB1** | **IOB0** | **Function** | **TIOCB1 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | 0 | 0 | 0 | Input capture register | Capture input source is TIOCB1 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCB1 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCB1 pin<br>Input capture at both edges |
| | 1 | × | × | | Setting prohibited |

Legend:

×: Don’t care

**Table 9.14 TIOR_1 (channel 1)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** | **TGRA_1** | |
| **IOA3** | **IOA2** | **IOA1** | **IOA0** | **Function** | **TIOCA1 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | 0 | 0 | 0 | Input capture register | Capture input source is TIOCA1 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCA1 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCA1 pin<br>Input capture at both edges |
| | 1 | × | × | | Setting prohibited |

Legend:
×: Don’t care

**Table 9.15 TIOR_2 (channel 2)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **TGRB_2** | |
| **IOB3** | **IOB2** | **IOB1** | **IOB0** | **Function** | **TIOCB2 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | × | 0 | 0 | Input capture register | Capture input source is TIOCB2 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCB2 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCB2 pin<br>Input capture at both edges |

Legend:

×: Don’t care

**Table 9.16 TIOR_2 (channel 2)**

| | | | | Description | |
|---|---|---|---|---|---|
| **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** | **TGRA_2** | |
| **IOA3** | **IOA2** | **IOA1** | **IOA0** | **Function** | **TIOCA2 Pin Function** |
| 0 | 0 | 0 | 0 | Output compare register | Output disabled |
| | | | 1 | | Initial output is 0 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 0 output<br>1 output at compare match |
| | | | 1 | | Initial output is 0 output<br>Toggle output at compare match |
| | 1 | 0 | 0 | | Output disabled |
| | | | 1 | | Initial output is 1 output<br>0 output at compare match |
| | | 1 | 0 | | Initial output is 1 output<br>1 output at compare match |
| | | | 1 | | Initial output is 1 output<br>Toggle output at compare match |
| 1 | × | 0 | 0 | Input capture register | Capture input source is TIOCA2 pin<br>Input capture at rising edge |
| | | | 1 | | Capture input source is TIOCA2 pin<br>Input capture at falling edge |
| | | 1 | × | | Capture input source is TIOCA2 pin<br>Input capture at both edges |

Legend:

×: Don’t care

### 9.3.4 Timer Interrupt Enable Register (TIER)

The TIER registers control enabling or disabling of interrupt requests for each channel. The TPU has three TIER registers, one for each channel.

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 7 | TTGE | 0 | R/W | A/D Conversion Start Request Enable<br>Enables or disables generation of A/D conversion start requests by TGRA input capture/compare match.<br>0: A/D conversion start request generation disabled<br>1: A/D conversion start request generation enabled |
| 6 | – | 1 | – | Reserved<br>This bit is always read as 1 and cannot be modified. |
| 5 | TCIEU | 0 | R/W | Underflow Interrupt Enable<br>Enables or disables interrupt requests (TCU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1 and 2. In channel 0, bit 5 is reserved.<br>0: Interrupt requests (TCIU) by TCFU disabled<br>1: Interrupt requests (TCIU) by TCFU enabled |
| 4 | TCIEV | 0 | R/W | Overflow Interrupt Enable<br>Enables or disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1.<br>0: Interrupt requests (TCIV) by TCFV disabled<br>1: Interrupt requests (TCIV) by TCFV enabled |
| 3 | TGIED | 0 | R/W | TGR Interrupt Enable D<br>Enables or disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channel 0. In channels 1 and 2, bit 3 is reserved. It is always read as 0 and cannot be modified.<br>0: Interrupt requests (TGID) by TGFD disabled<br>1: Interrupt requests (TGID) by TGFD enabled. |
| 2 | TGIEC | 0 | R/W | TGR Interrupt Enable C<br>Enables or disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channel 0. In channels 1 and 2, bit 2 is reserved. It is always read as 0 and cannot be modified.<br>0: Interrupt requests (TGIC) by TGFC disabled<br>1: Interrupt requests (TGIC) by TGFC enabled |

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 1 | TGIEB | 0 | R/W | TGR Interrupt Enable B<br>Enables or disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1.<br>0: Interrupt requests (TGIB) by TGFB disabled<br>1: Interrupt requests (TGIB) by TGFB enabled |
| 0 | TGIEA | 0 | R/W | TGR Interrupt Enable A<br>Enables or disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1.<br>0: Interrupt requests (TGIA) by TGFA disabled<br>1: Interrupt requests (TGIA) by TGFA enabled |

### 9.3.5 Timer Status Register (TSR)

The TSR registers indicate the status of each channel. The TPU has three TSR registers, one for each channel.

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 7 | TCFD | 1 | R | Count Direction Flag<br>Status flag that shows the direction in which TCNT counts in channel 1 and 2. In channel 0, bit 7 is reserved. It is always read as 0 and cannot be modified.<br>0: TCNT counts down<br>1: TCNT counts up |
| 6 | – | 1 | – | Reserved<br>This bit is always read as 1 and cannot be modified. |
| 5 | TCFU | 0 | R/(W)* | Underflow Flag<br>Status flag that indicates that TCNT underflow has occurred when channels 1 and 2 are set to phase counting mode. The write value should always be 0 to clear this flag. In channel 0, bit 5 is reserved.<br>[Setting condition]<br>When the TCNT value underflows (change from H'0000 to H'FFFF)<br>[Clearing condition]<br>When 0 is written to TCFU after reading TCFU = 1 |

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 4 | TCFV | 0 | R/(W)* | Overflow Flag<br>Status flag that indicates that TCNT overflow has occurred. The write value should always be 0 to clear this flag.<br>[Setting condition]<br>When the TCNT value overflows (change from H'FFFF to H'0000)<br>[Clearing condition]<br>When 0 is written to TCFV after reading TCFV = 1 |
| 3 | TGFD | 0 | R/(W)* | Input Capture/Output Compare Flag D<br>Status flag that indicates the occurrence of TGRD input capture or compare match in channel 0. The write value should always be 0 to clear this flag. In channels 1 and 2, bit 3 is reserved. It is always read as 0 and cannot be modified.<br>[Setting conditions]<br>• When TCNT = TGRD while TGRD is functioning as output compare register<br>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register<br>[Clearing condition]<br>When 0 is written to TGFD after reading TGFD = 1 |
| 2 | TGFC | 0 | R/(W)* | Input Capture/Output Compare Flag C<br>Status flag that indicates the occurrence of TGRC input capture or compare match in channel 0. The write value should always be 0 to clear this flag. In channels 1 and 2, bit 2 is reserved. It is always read as 0 and cannot be modified.<br>[Setting conditions]<br>• When TCNT = TGRC while TGRC is functioning as output compare register<br>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register<br>[Clearing condition]<br>When 0 is written to TGFC after reading TGFC = 1 |

| Bit | Bit Name | Initial value | R/W | Description |
|---|---|---|---|---|
| 1 | TGFB | 0 | R/(W)* | Input Capture/Output Compare Flag B<br>Status flag that indicates the occurrence of TGRB input capture or compare match. The write value should always be 0 to clear this flag.<br>[Setting conditions]<br>• When TCNT = TGRB while TGRB is functioning as output compare register<br>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register<br>[Clearing condition]<br>When 0 is written to TGFB after reading TGFB = 1 |
| 0 | TGFA | 0 | R/(W)* | Input Capture/Output Compare Flag A<br>Status flag that indicates the occurrence of TGRA input capture or compare match. The write value should always be 0 to clear this flag.<br>[Setting conditions]<br>• When TCNT = TGRA while TGRA is functioning as output compare register<br>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register<br>[Clearing conditions]<br>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1<br>• When 0 is written to TGFA after reading TGFA = 1 |

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

### 9.3.6 Timer Counter (TCNT)

The TCNT registers are 16-bit counters. The TPU has three TCNT counters, one for each channel. The TCNT counters are initialized to H'0000 by a reset, and in hardware standby mode. The TCNT counters cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

### 9.3.7 Timer General Register (TGR)

The TGR registers are 16-bit registers with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channel 0 and two each for channels 1 and 2. TGRC and TGRD for channel 0 can also be designated for operation as buffer registers. The TGR registers cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit. TGR buffer register combinations are TGRA—TGRC and TGRB—TGRD.

### 9.3.8 Timer Start Register (TSTR)

TSTR is an 8-bit readable/writable register that selects operation/stoppage for channels 0 to 2. When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 3 | – | All 0 | – | Reserved<br>The write value should always be 0. |
| 2 | CST2 | 0 | R/W | Counter Start 2 to 0 (CST2 to CST0) |
| 1 | CST1 | 0 | R/W | These bits select operation or stoppage for TCNT. |
| 0 | CST0 | 0 | R/W | If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained.<br>If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value.<br>0: TCNT_2 to TCNT_0 count operation is stopped<br>1: TCNT_2 to TCNT_0 performs count operation |

### 9.3.9 Timer Synchro Register (TSYR)

TSYR selects independent operation or synchronous operation for the channel 0 to 2 TCNT counters. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 3 | – | All 0 | – | Reserved<br>The write value should always be 0. |
| 2 | SYNC2 | 0 | R/W | Timer Synchro 2 to 0 |
| 1 | SYNC1 | 0 | R/W | These bits select whether operation is independent of or synchronized with other channels.<br>When synchronous operation is selected, synchronous presetting of multiple channels, and synchronous clearing through counter clearing on another channel are possible. To set synchronous operation, the SYNC bits for at least two channels must be set to 1. To set synchronous clearing, in addition to the SYNC bit, the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.<br>0: TCNT_2 to TCNT_0 operates independently (TCNT presetting /clearing is unrelated to other channels)<br>1: TCNT_2 to TCNT_0 performs synchronous operation TCNT synchronous presetting/synchronous clearing is possible |
| 0 | SYNC0 | 0 | R/W | |

## 9.4 Interface to Bus Master

### 9.4.1 16-Bit Registers

TCNT and TGR are 16-bit registers. As the data bus to the bus master is 16 bits wide, these registers can be read and written to in 16-bit units.

These registers cannot be read from or written to in 8-bit units; 16-bit access must always be used.

An example of 16-bit register access operation is shown in figure 9.2.

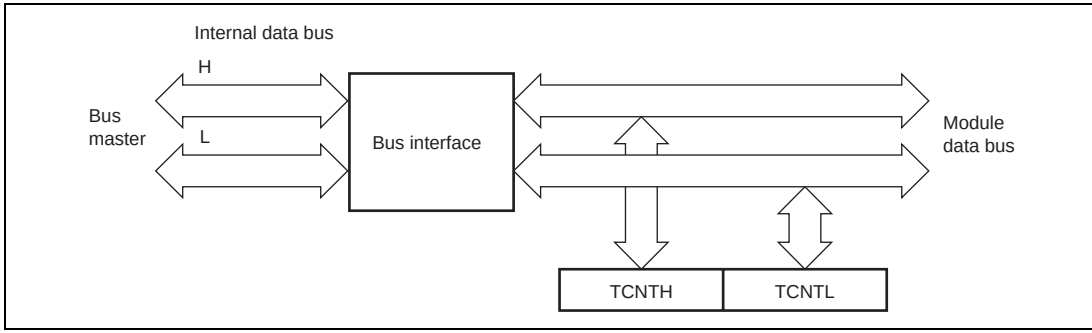



**Figure 9.2 16-Bit Register Access Operation [Bus Master ↔ TCNT (16 Bits)]**

### 9.4.2 8-Bit Registers

Registers other than TCNT and TGR are 8-bit. As the data bus to the CPU is 16 bits wide, these registers can be read and written to in 16-bit units. They can also be read and written to in 8-bit units.

Examples of 8-bit register access operation are shown in figures 9.3 to 9.5.

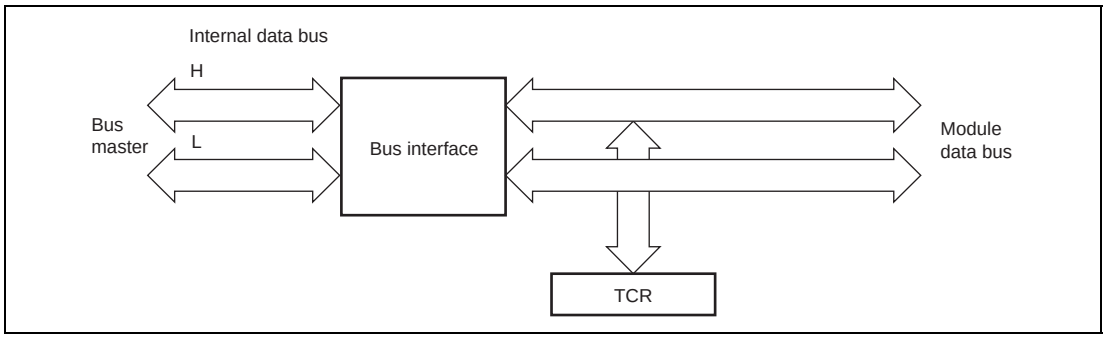



**Figure 9.3 8-Bit Register Access Operation [Bus Master ↔ TCR (Upper 8 Bits)]**

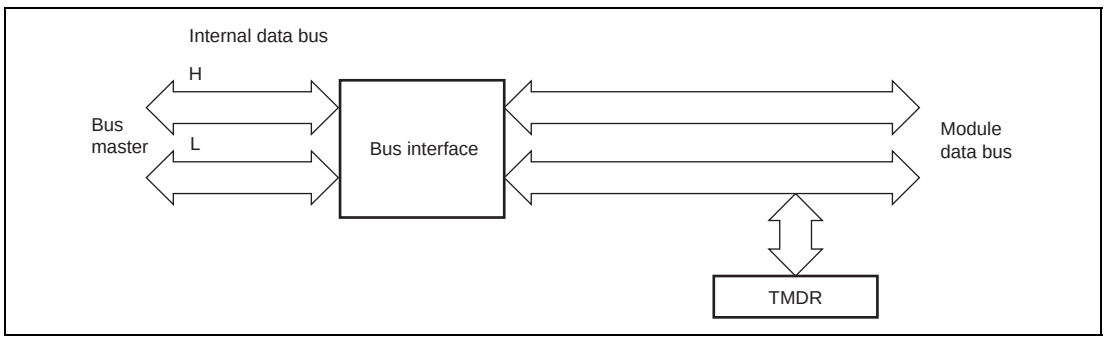



**Figure 9.4 8-Bit Register Access Operation [Bus Master ↔ TMDR (Lower 8 Bits)]**

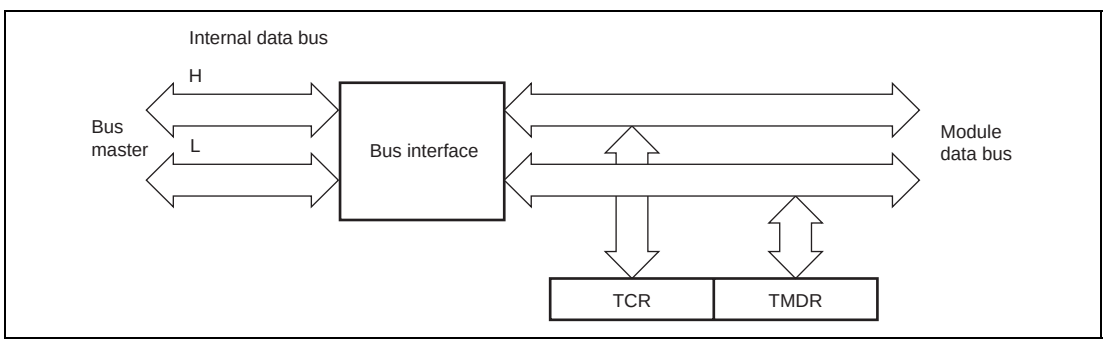



**Figure 9.5 8-Bit Register Access Operation [Bus Master ↔ TCR and TMDR (16 Bits)]**

## 9.5 Operation

### 9.5.1 Basic Functions

Each channel has a TCNT and TGR. TCNT performs up-counting, and is also capable of free-running operation, synchronous counting, and external event counting. Each TGR can be used as an input capture register or output compare register.

**Counter Operation:** When one of bits CST0 to CST2 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

1. Example of count operation setting procedure
   Figure 9.6 shows an example of the count operation setting procedure.

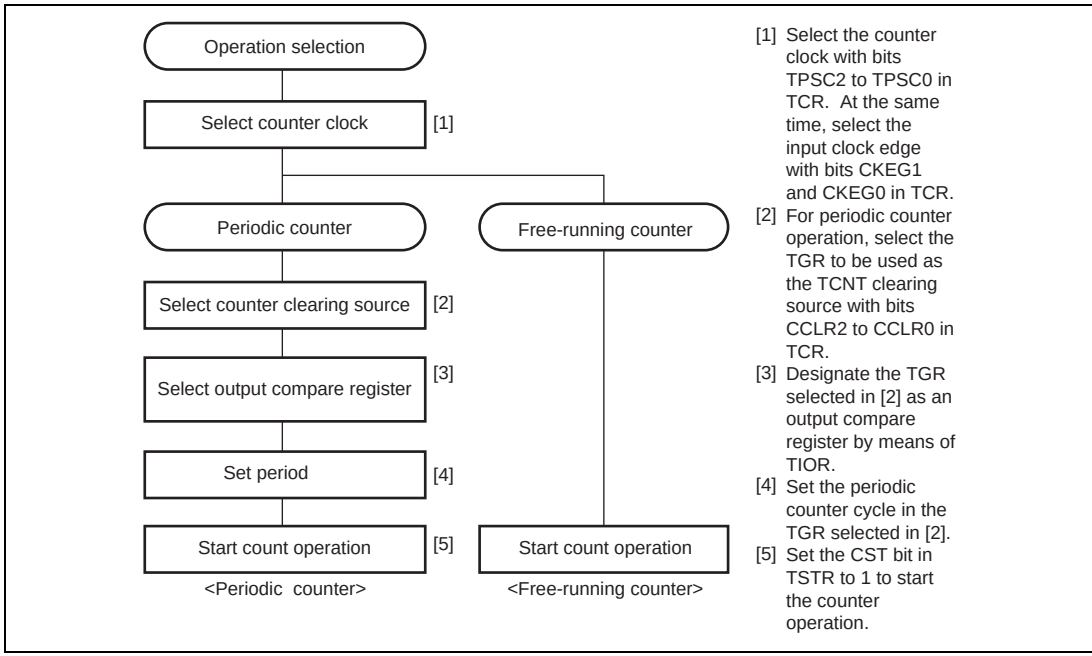



**Figure 9.6 Example of Counter Operation Setting Procedure**

2. Free-running count operation and periodic count operation

   Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000. Figure 9.7 illustrates free-running counter operation.

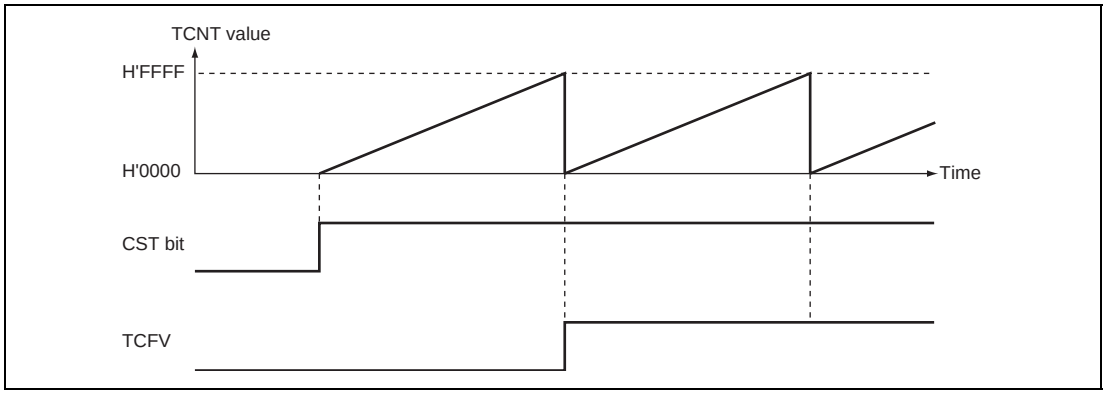



**Figure 9.7 Free-Running Counter Operation**

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts up-count operation as periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000. If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000. Figure 9.8 illustrates periodic counter operation.

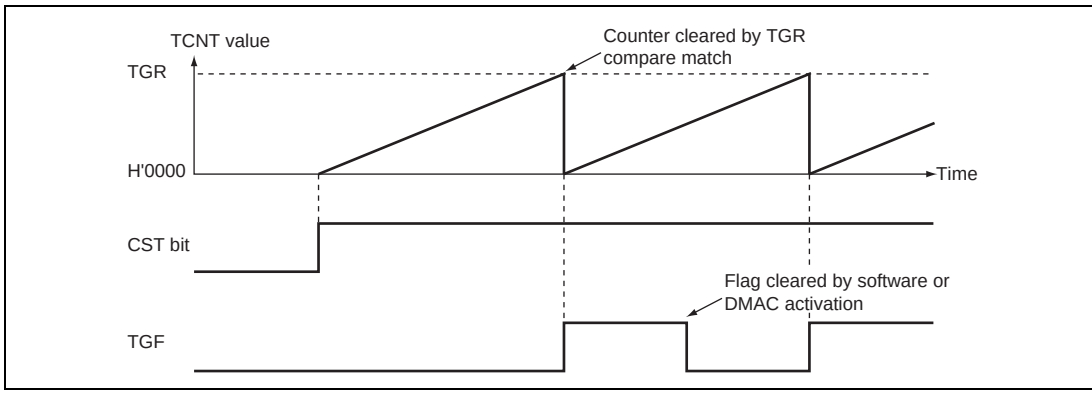



**Figure 9.8 Periodic Counter Operation**

**Waveform Output by Compare Match:** The TPU can perform 0, 1, or toggle output from the corresponding output pin using compare match.

1. Example of setting procedure for waveform output by compare match

   Figure 9.9 shows an example of the setting procedure for waveform output by compare match.

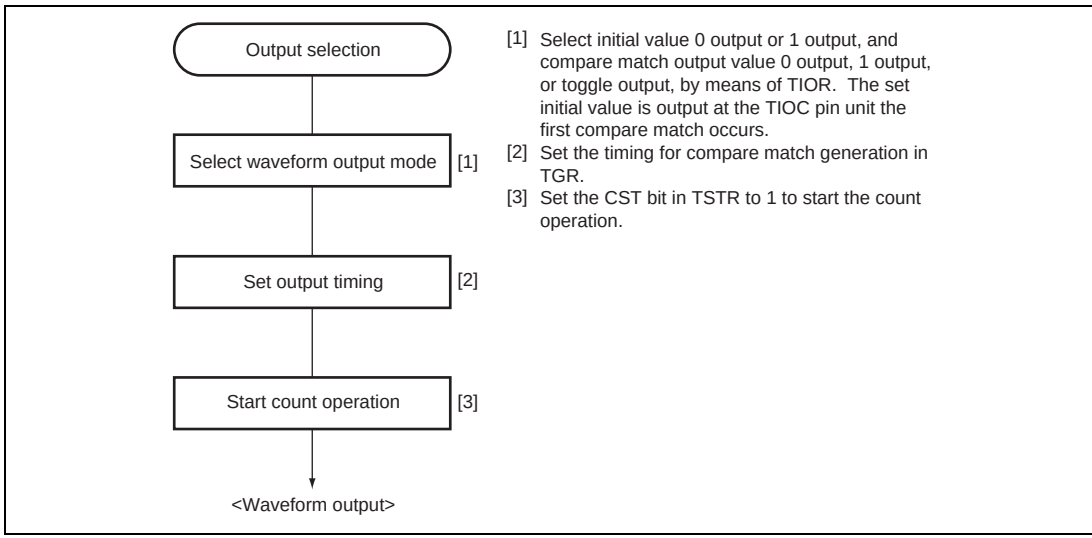



**Figure 9.9 Example of Setting Procedure for Waveform Output by Compare Match**

2. Examples of waveform output operation

Figure 9.10 shows an example of 0 output/1 output. In this example TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level coincide, the pin level does not change.

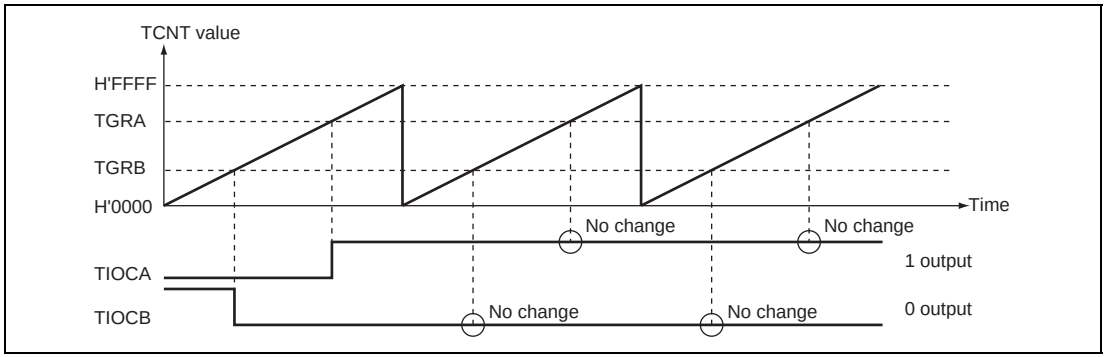



**Figure 9.10 Example of 0 Output/1 Output Operation**

Figure 9.11 shows an example of toggle output.

In this example TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.

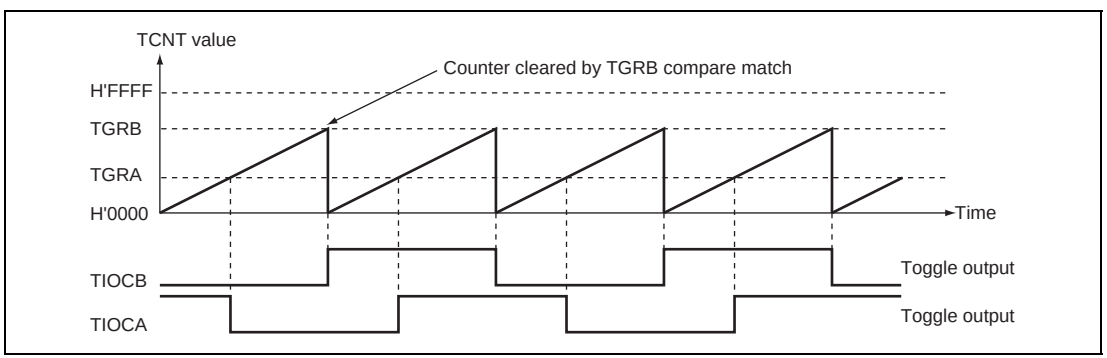



**Figure 9.11 Example of Toggle Output Operation**

**Input Capture Function:** The TCNT value can be transferred to TGR on detection of the TIOC pin input edge. Rising edge, falling edge, or both edges can be selected as the detected edge.

1. Example of input capture operation setting procedure

   Figure 9.12 shows an example of the input capture operation setting procedure.

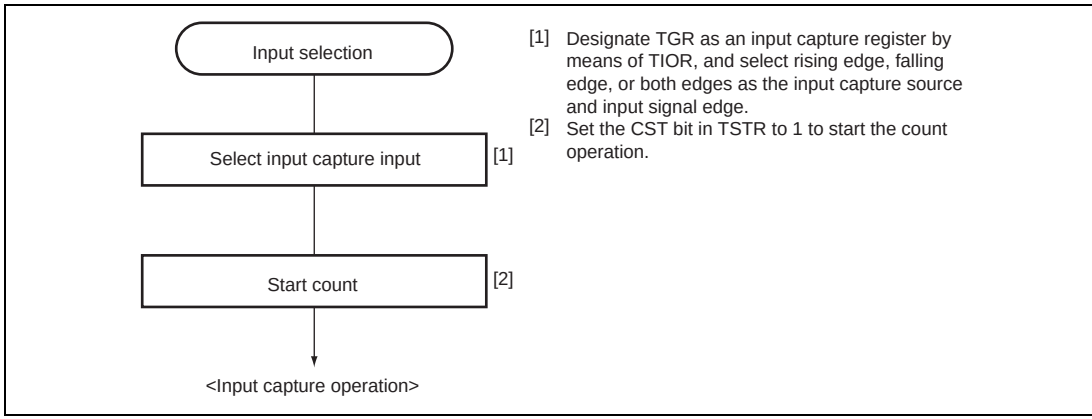



**Figure 9.12 Example of Input Capture Operation Setting Procedure**

2. Example of input capture operation

Figure 9.13 shows an example of input capture operation. In this example both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.

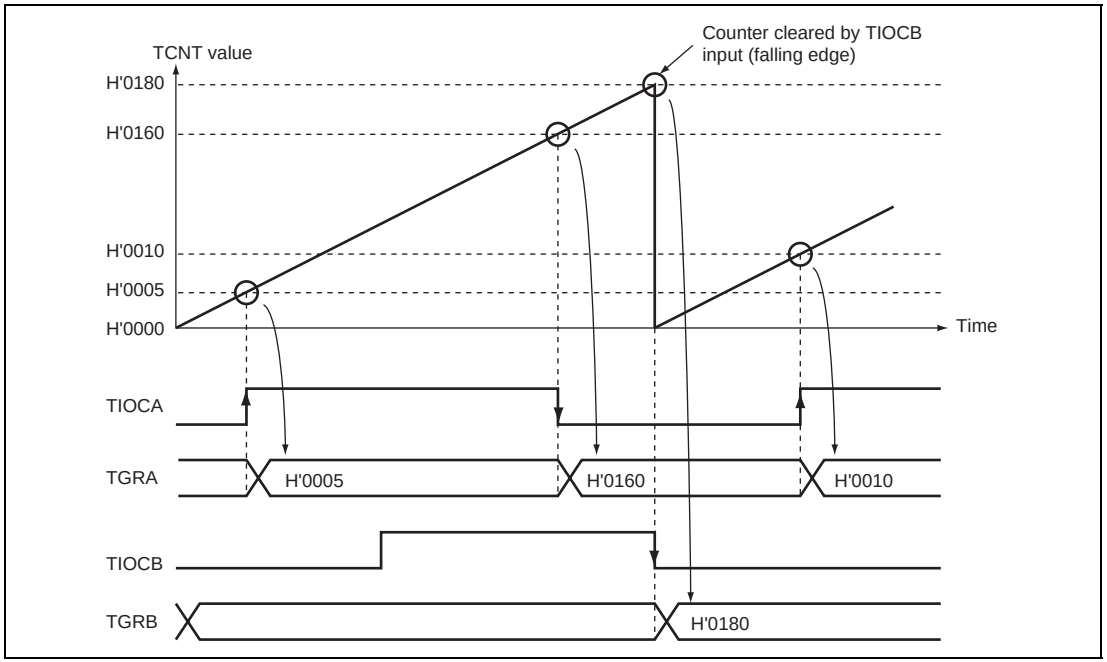
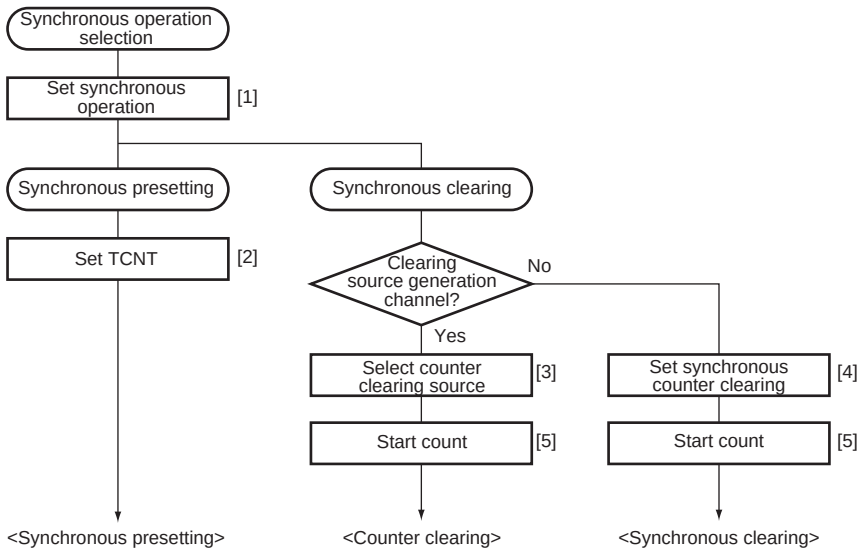



**Figure 9.13 Example of Input Capture Operation**

### 9.5.2 Synchronous Operation

In synchronous operation, the values in a number of TCNT counters can be rewritten simultaneously (synchronous presetting). Also, a number of TCNT counters can be cleared simultaneously by making the appropriate setting in TCR (synchronous clearing). Synchronous operation enables TGR to be incremented with respect to a single time base. Channels 0 to 2 can all be designated for synchronous operation.

**Example of Synchronous Operation Setting Procedure:** Figure 9.14 shows an example of the synchronous operation setting procedure.




[1] Set to 1 the SYNC bits in TSYR corresponding to the channels to be designated for synchronous operation.
[2] When the TCNT counter of any of the channels designated for synchronous operation is written to, the same value is simultaneously written to the other TCNT counters.
[3] Use bits CCLR2 to CCLR0 in TCR to specify TCNT clearing by input capture/output compare, etc.
[4] Use bits CCLR2 to CCLR0 in TCR to designate synchronous clearing for the counter clearing source.
[5] Set to 1 the CST bits in TSTR for the relevant channels, to start the count operation.

**Figure 9.14 Example of Synchronous Operation Setting Procedure**

**Example of Synchronous Operation:** Figure 9.15 shows an example of synchronous operation. In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGRB_0 compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source. Three-phase PWM waveforms are output from pins TIOC0A, TIOC1A, and TIOC2A. At this time, synchronous presetting, and synchronous clearing by TGRB_0 compare match, is performed for channel 0 to 2 TCNT counters, and the data set in TGRB_0 is used as the PWM cycle. For details of PWM modes, see section 9.5.4, PWM Modes.

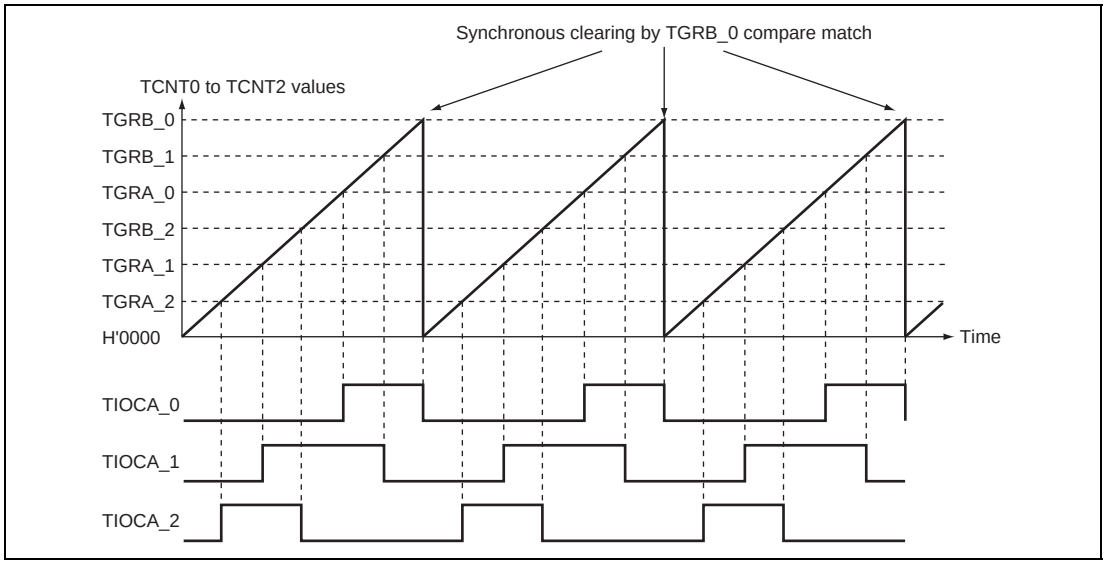



**Figure 9.15 Example of Synchronous Operation**

### 9.5.3 Buffer Operation

Buffer operation, provided for channel 0, enables TGRC and TGRD to be used as buffer registers. Buffer operation differs depending on whether TGR has been designated as an input capture register or as a compare match register. Table 9.17 shows the register combinations used in buffer operation.

**Table 9.17 Register Combinations in Buffer Operation**

| Channel | Timer General Register | Buffer Register |
|---|---|---|
| 0 | TGRA_0 | TGRC_0 |
| | TGRB_0 | TGRD_0 |

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register. This operation is illustrated in figure 9.16.

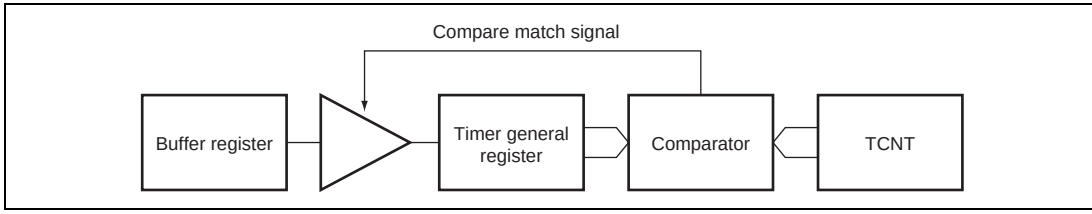



**Figure 9.16 Compare Match Buffer Operation**

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in the timer general register is transferred to the buffer register. This operation is illustrated in figure 9.17.

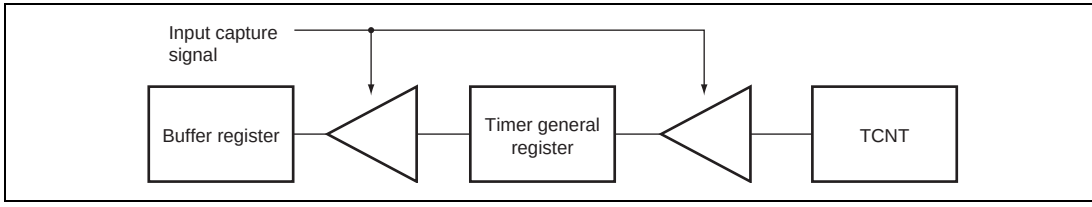



**Figure 9.17 Input Capture Buffer Operation**

**Example of Buffer Operation Setting Procedure:** Figure 9.18 shows an example of the buffer operation setting procedure.

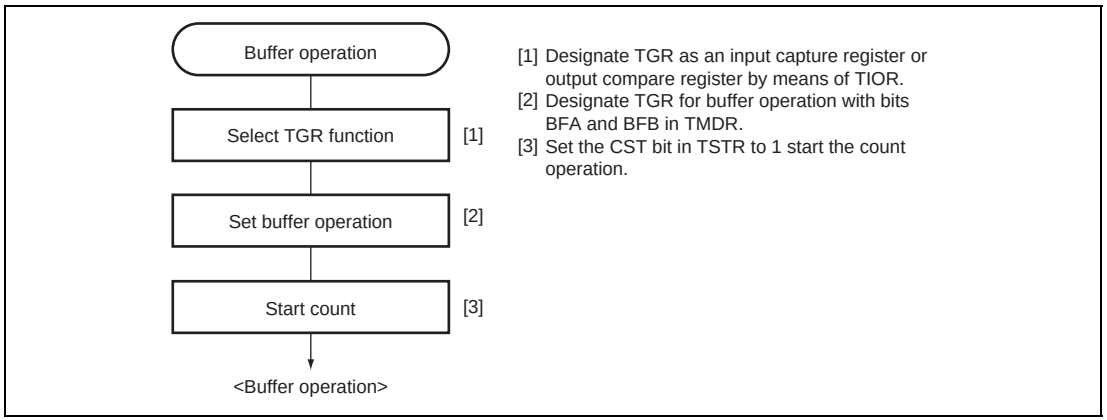



**Figure 9.18 Example of Buffer Operation Setting Procedure**

**Examples of Buffer Operation**

1. When TGR is an output compare register

   Figure 9.19 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B. As buffer operation has been set, when compare match A occurs the output changes and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs. For details of PWM modes, see section 9.5.4, PWM Modes.

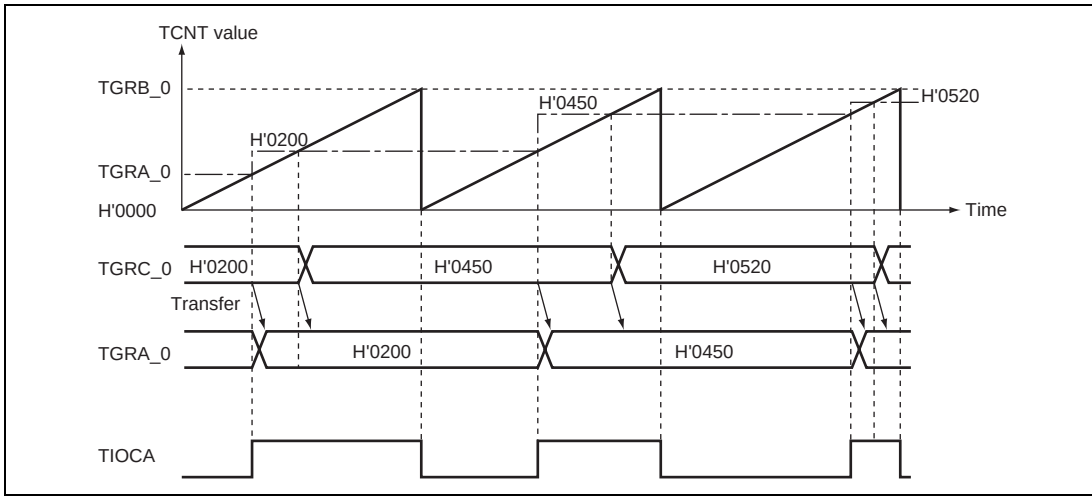



**Figure 9.19 Example of Buffer Operation (1)**

2. When TGR is an input capture register

   Figure 9.20 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC. Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge. As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.

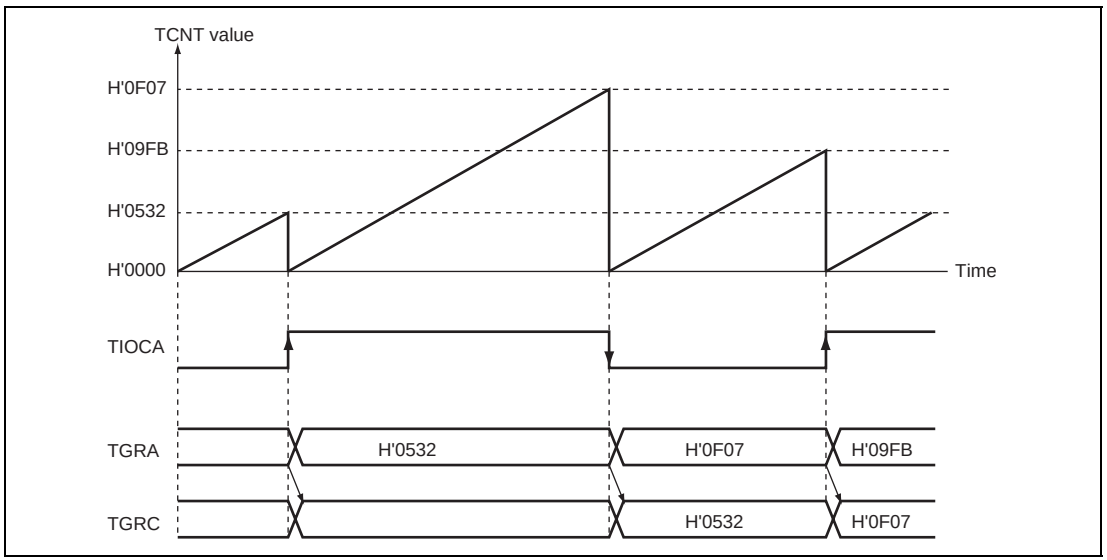



**Figure 9.20 Example of Buffer Operation (2)**

### 9.5.4 PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0, 1, or toggle output can be selected as the output level in response to compare match of each TGR. Settings of TGR registers can output a PWM waveform in the range of 0 % to 100 % duty. Designating TGR compare match as the counter clearing source enables the period to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible. There are two PWM modes, as described below.

- PWM mode 1

  PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The output specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR is output from the TIOCA and TIOCC pins at compare matches A and C, and the output specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR is output at compare matches B and D. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs. In PWM mode 1, a maximum 4-phase PWM output is possible.
- PWM mode 2

  PWM output is generated using one TGR as the cycle register and the others as duty registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronization register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty registers are identical, the output value does not change when a compare match occurs. In PWM mode 2, a maximum 7-phase PWM output is possible by combined use with synchronous operation. The correspondence between PWM output pins and registers is shown in table 9.18.

**Table 9.18 PWM Output Registers and Output Pins**

| Channel | Registers | Output Pins: PWM Mode 1 | Output Pins: PWM Mode 2 |
|---|---|---|---|
| 0 | TGRA_0 | TIOCA0 | TIOCA0 |
| | TGRB_0 | | TIOCB0 |
| | TGRC_0 | TIOCC0 | TIOCC0 |
| | TGRD_0 | | TIOCD0 |
| 1 | TGRA_1 | TIOCA1 | TIOCA1 |
| | TGRB_1 | | TIOCB1 |
| 2 | TGRA_2 | TIOCA2 | TIOCA2 |
| | TGRB_2 | | TIOCB2 |

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the period is set.

**Example of PWM Mode Setting Procedure:** Figure 9.21 shows an example of the PWM mode setting procedure.

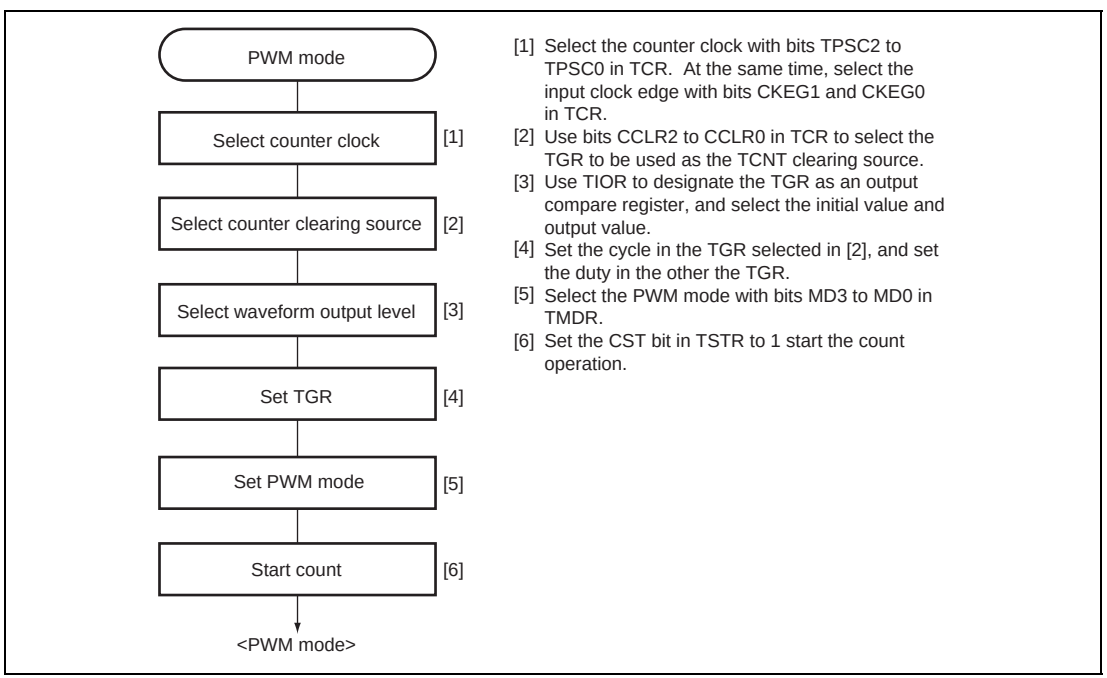



**Figure 9.21 Example of PWM Mode Setting Procedure**

**Examples of PWM Mode Operation:** Figure 9.22 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value. In this case, the value set in TGRA is used as the period, and the values set in TGRB registers as the duty.

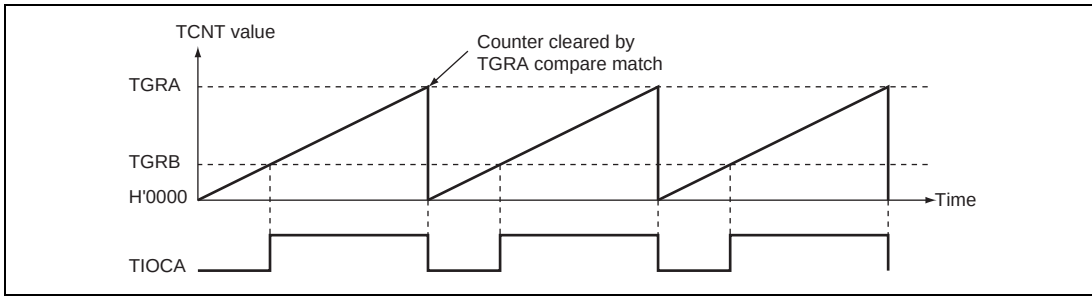



**Figure 9.22 Example of PWM Mode Operation (1)**

Figure 9.23 shows an example of PWM mode 2 operation. In this example, synchronous operation is designated for channels 0 and 1, TGRB_1 compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGRA_0 to TGRD_0, TGRA_1), to output a 5-phase PWM waveform. In this case, the value set in TGRB_1 is used as the cycle, and the values set in the other TGRs as the duty.

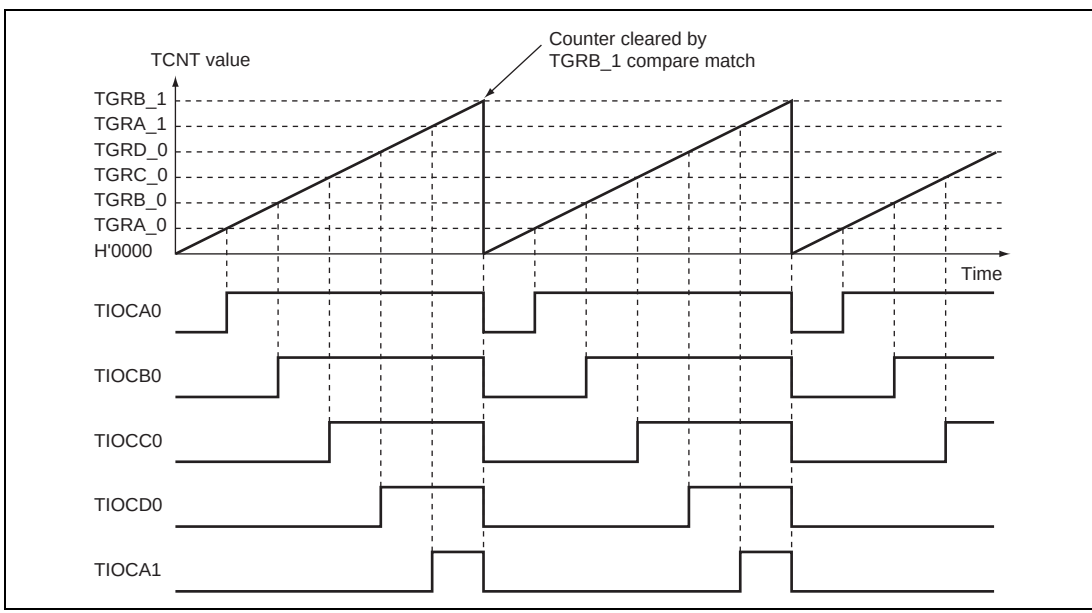



**Figure 9.23 Example of PWM Mode Operation (2)**

Figure 9.24 shows examples of PWM waveform output with 0% duty and 100% duty in PWM mode.

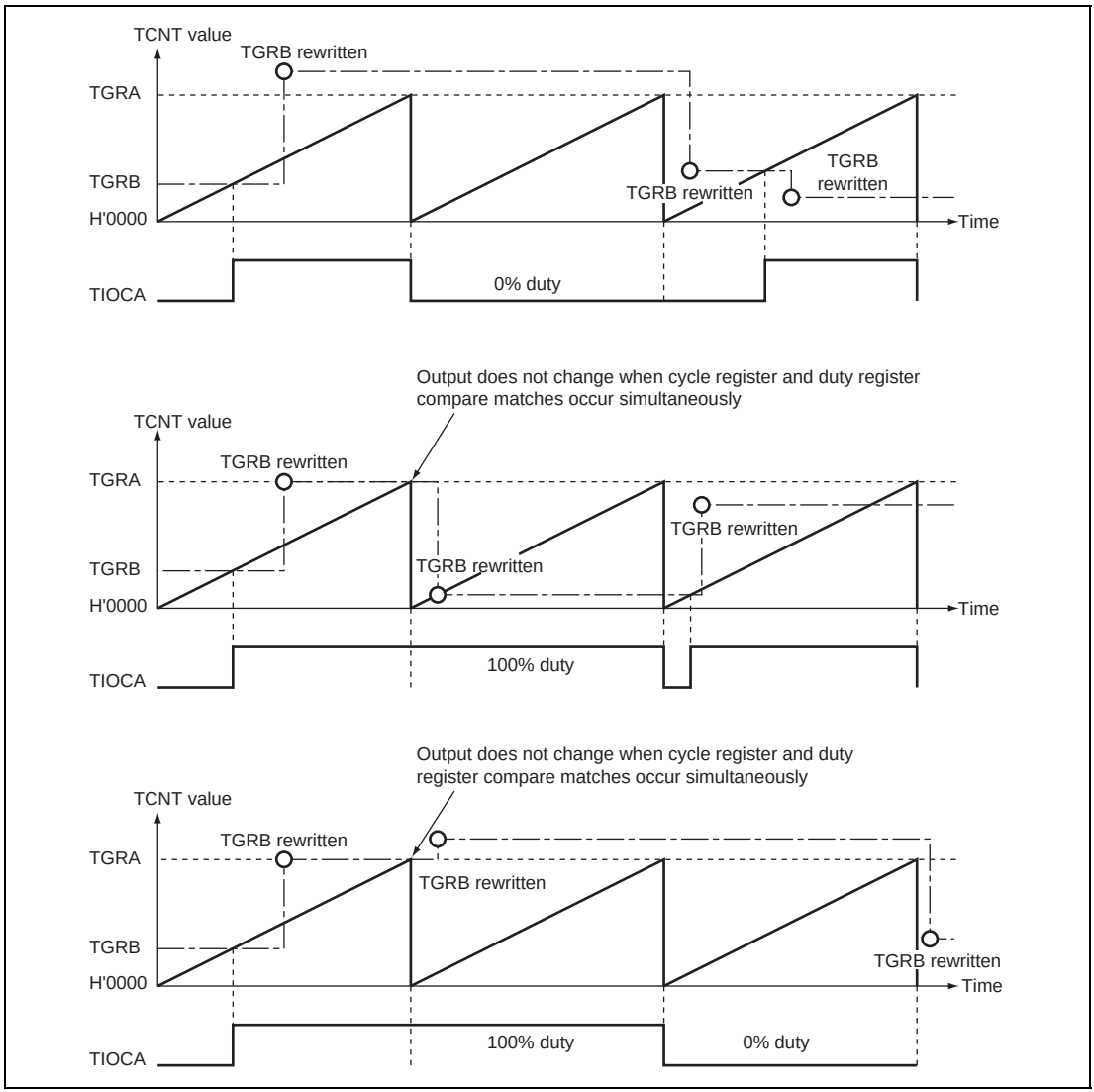



**Figure 9.24 Example of PWM Mode Operation (3)**

### 9.5.5 Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1 and 2. When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used. This can be used for two-phase encoder pulse input. When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set. The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down. Table 9.19 shows the correspondence between external clock pins and channels.

**Table 9.19 Phase Counting Mode Clock Input Pins**

| | External Clock Pins | |
|---|---|---|
| **Channels** | **A-Phase** | **B-Phase** |
| When channel 1 is set to phase counting mode | TCLKA | TCLKB |
| When channel 2 is set to phase counting mode | TCLKC | TCLKD |

**Example of Phase Counting Mode Setting Procedure:** Figure 9.25 shows an example of the phase counting mode setting procedure.

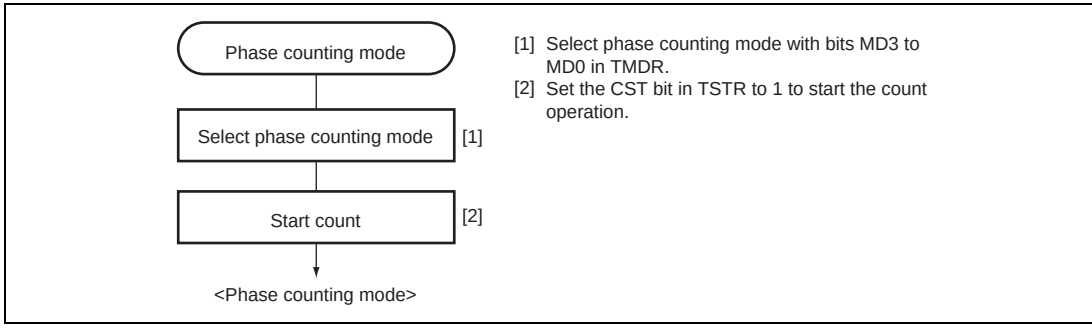



**Figure 9.25 Example of Phase Counting Mode Setting Procedure**

**Examples of Phase Counting Mode Operation:** In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

1. Phase counting mode 1

   Figure 9.26 shows an example of phase counting mode 1 operation, and table 9.20 summarizes the TCNT up/down-count conditions.

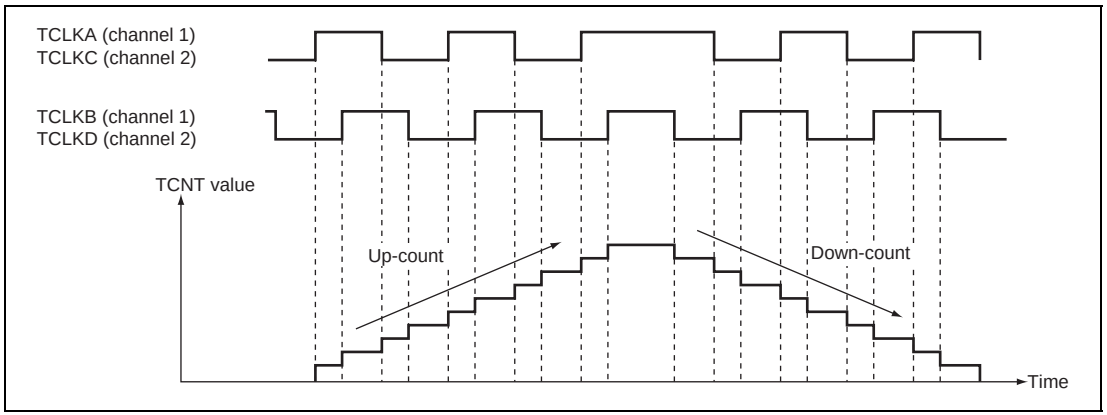



**Figure 9.26 Example of Phase Counting Mode 1 Operation**

**Table 9.20 Up/Down-Count Conditions in Phase Counting Mode 1**

| TCLKA (Channel 1) / TCLKC (Channel 2) | TCLKB (Channel 1) / TCLKD (Channel 2) | Operation |
|---|---|---|
| High level | ↑ | Up-count |
| Low level | ↓ | |
| ↑ | Low level | |
| ↓ | High level | |
| High level | ↓ | Down-count |
| Low level | ↑ | |
| ↑ | High level | |
| ↓ | Low level | |

Legend:
↑ : Rising edge
↓ : Falling edge

2. Phase counting mode 2

Figure 9.27 shows an example of phase counting mode 2 operation, and table 9.21 summarizes the TCNT up/down-count conditions.

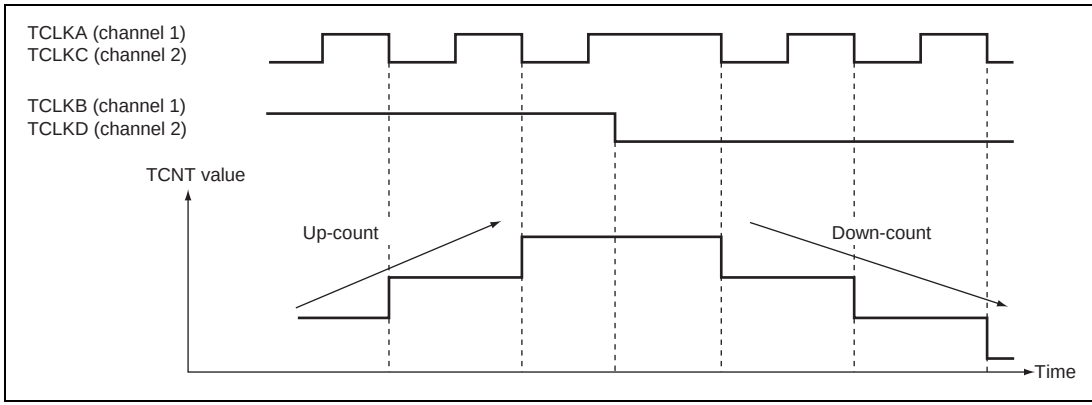



**Figure 9.27 Example of Phase Counting Mode 2 Operation**

**Table 9.21 Up/Down-Count Conditions in Phase Counting Mode 2**

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|---|---|---|
| High level | ↑ | Don't care |
| Low level | ↓ | Don't care |
| ↑ | Low level | Don't care |
| ↓ | High level | Up-count |
| High level | ↓ | Don't care |
| Low level | ↑ | Don't care |
| ↑ | High level | Don't care |
| ↓ | Low level | Down-count |

Legend:
↑ : Rising edge
↓ : Falling edge

3. Phase counting mode 3

Figure 9.28 shows an example of phase counting mode 3 operation, and table 9.22 summarizes the TCNT up/down-count conditions.

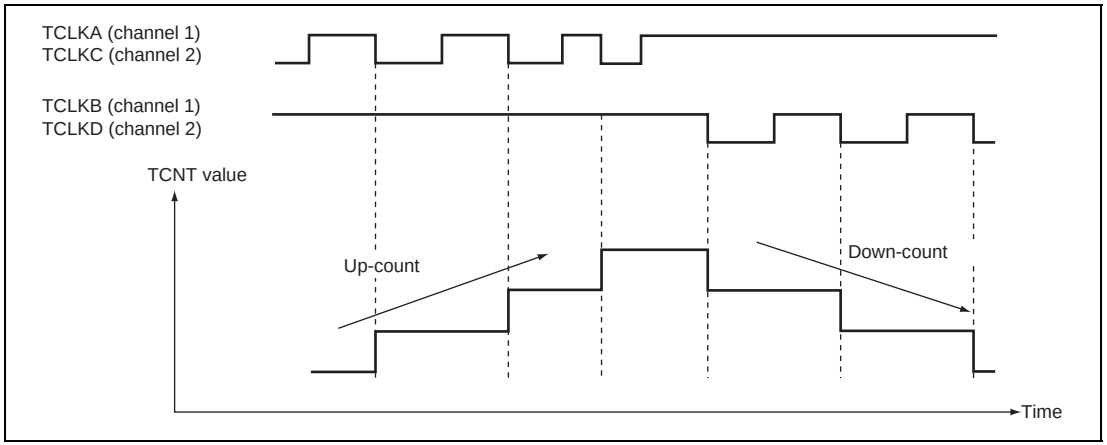



**Figure 9.28 Example of Phase Counting Mode 3 Operation**

**Table 9.22 Up/Down-Count Conditions in Phase Counting Mode 3**

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|---|---|---|
| High level | ↑ | Don't care |
| Low level | ↓ | Don't care |
| ↑ | Low level | Don't care |
| ↓ | High level | Up-count |
| High level | ↓ | Down-count |
| Low level | ↑ | Don't care |
| ↑ | High level | Don't care |
| ↓ | Low level | Don't care |

Legend:
↑ : Rising edge
↓ : Falling edge

4. Phase counting mode 4

   Figure 9.29 shows an example of phase counting mode 4 operation, and table 9.23 summarizes the TCNT up/down-count conditions.

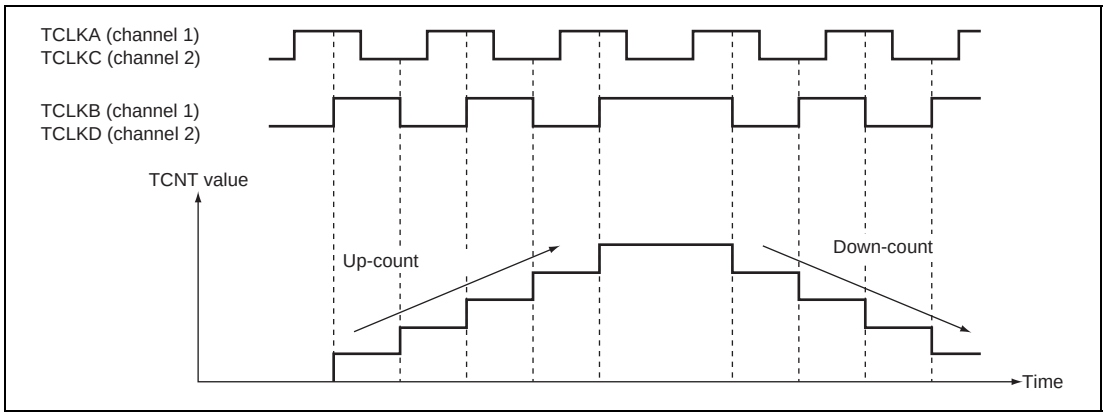



**Figure 9.29 Example of Phase Counting Mode 4 Operation**

**Table 9.23 Up/Down-Count Conditions in Phase Counting Mode 4**

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|---|---|---|
| High level | ↑ | Up-count |
| Low level | ↓ | |
| ↑ | Low level | Don't care |
| ↓ | High level | |
| High level | ↓ | Down-count |
| Low level | ↑ | |
| ↑ | High level | Don't care |
| ↓ | Low level | |

Legend:
↑ : Rising edge
↓ : Falling edge

## 9.6 Interrupts

### 9.6.1 Interrupt Source and Priority

There are three kinds of TPU interrupt source: TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disabled bit, allowing generation of interrupt request signals to be enabled or disabled individually. When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0. Relative channel priorities can be changed by the interrupt controller, but the priority order within a channel is fixed. For details, see section 5, Interrupt Controller. Table 9.24 lists the TPU interrupt sources.

**Table 9.24 TPU Interrupts**

| Channel | Name | Interrupt Source | Interrupt Flag | DMAC Activation | Priority* |
|---|---|---|---|---|---|
| 0 | TGI0A | TGRA_0 input capture/compare match | TGFA | Possible | High |
| | TGI0B | TGRB_0 input capture/compare match | TGFB | Not possible | |
| | TGI0C | TGRC_0 input capture/compare match | TGFC | Not possible | |
| | TGI0D | TGRD_0 input capture/compare match | TGFD | Not possible | |
| | TCI0V | TCNT_0 overflow | TCFV | Not possible | |
| 1 | TGI1A | TGRA_1 input capture/compare match | TGFA | Possible | |
| | TGI1B | TGRB_1 input capture/compare match | TGFB | Not possible | |
| | TCI1V | TCNT_1 overflow | TCFV | Not possible | |
| | TCI1U | TCNT_1 underflow | TCFU | Not possible | |
| 2 | TGI2A | TGRA_2 input capture/compare match | TGFA | Possible | |
| | TGI2B | TGRB_2 input capture/compare match | TGFB | Not possible | |
| | TCI2V | TCNT_2 overflow | TCFV | Not possible | |
| | TCI2U | TCNT_2 underflow | TCFU | Not possible | Low |

Note: * This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

**Input Capture/Compare Match Interrupt:** An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1 by the occurrence of a TGR input capture/compare match on a particular channel. The interrupt request is cleared by clearing the TGF flag to 0. The TPU has 8 input capture/compare match interrupts, four each for channel 0, and two each for channels 1 and 2.

**Overflow Interrupt:** An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to 1 by the occurrence of TCNT overflow on a channel. The interrupt request is cleared by clearing the TCFV flag to 0. The TPU has three overflow interrupts, one for each channel.

**Underflow Interrupt:** An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has two underflow interrupts, one each for channels 1 and 2.

### 9.6.2 DMAC Activation

The DMAC can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 7, DMA Controller (DMAC). With the TPU, a total of three TGRA input capture/compare match interrupts can be used as DMAC activation sources, one for each channel.

### 9.6.3 A/D Converter Activation

The A/D converter can be activated by the TGRA input capture/compare match for a channel. If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started. In the TPU, a total of three TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

## 9.7 Operation Timing

### 9.7.1 Input/Output Timing

**TCNT Count Timing:** Figure 9.30 shows TCNT count timing in internal clock operation, and figure 9.31 shows TCNT count timing in external clock operation.

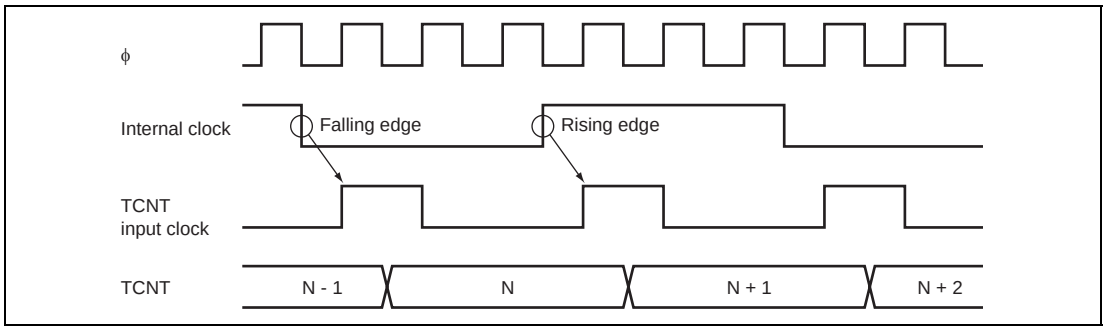



**Figure 9.30 Count Timing in Internal Clock Operation**

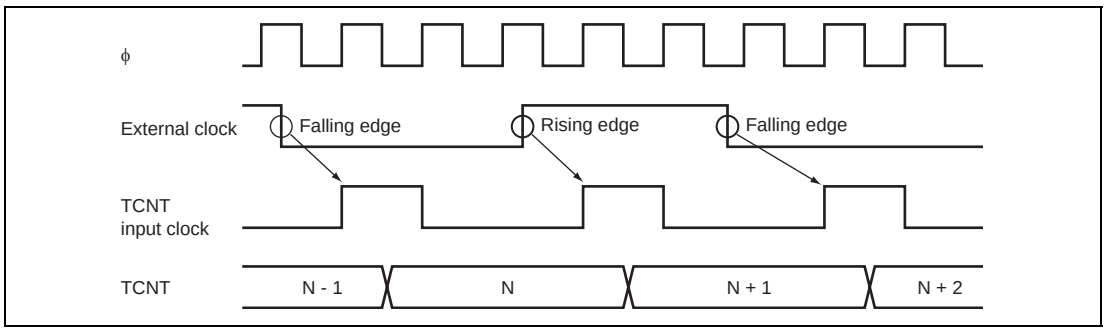



**Figure 9.31 Count Timing in External Clock Operation**

**Output Compare Output Timing:** A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin. After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated. Figure 9.32 shows output compare output timing.

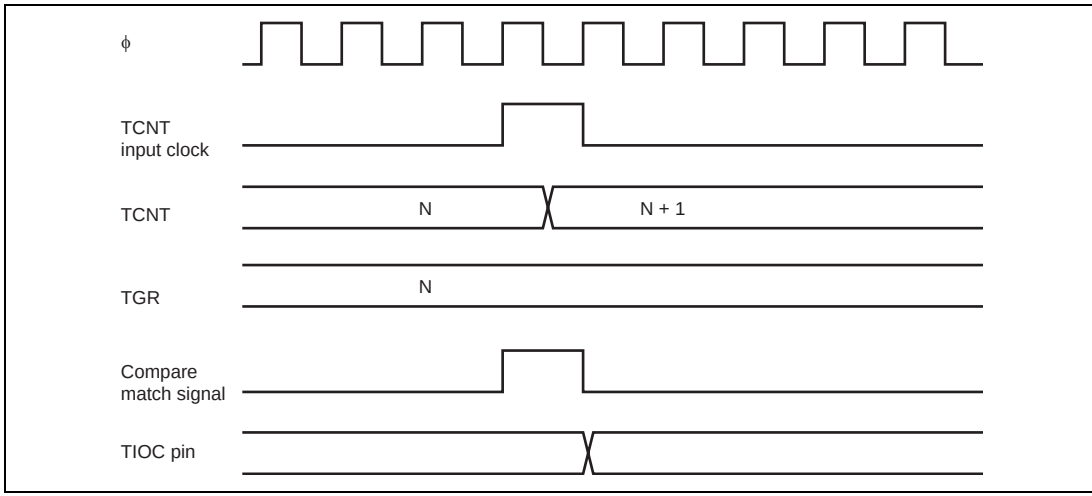



**Figure 9.32 Output Compare Output Timing**

**Input Capture Signal Timing:** Figure 9.33 shows input capture signal timing.

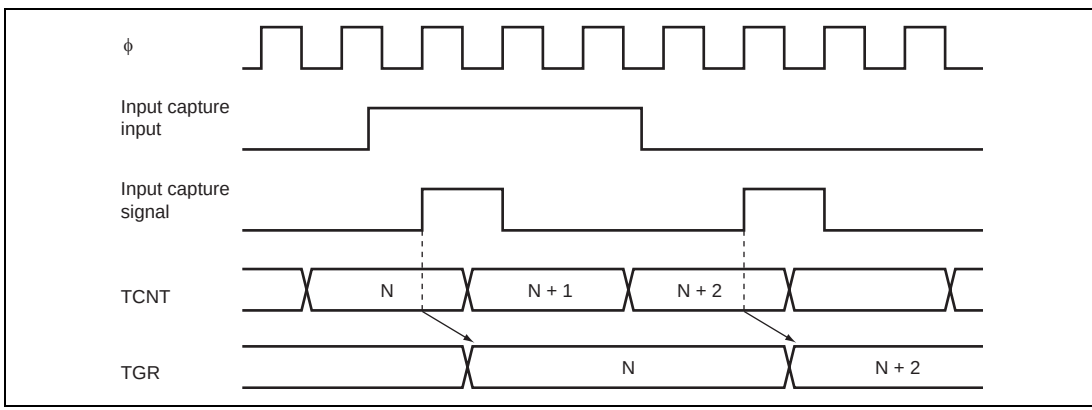



**Figure 9.33 Input Capture Input Signal Timing**

**Timing for Counter Clearing by Compare Match/Input Capture:** Figure 9.34 shows the timing when counter clearing by compare match occurrence is specified, and figure 9.35 shows the timing when counter clearing by input capture occurrence is specified.

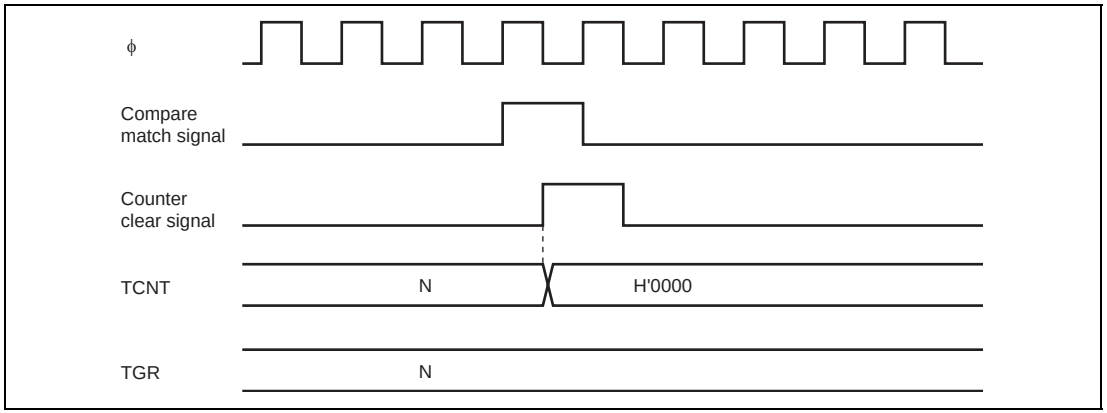



**Figure 9.34 Counter Clear Timing (Compare Match)**

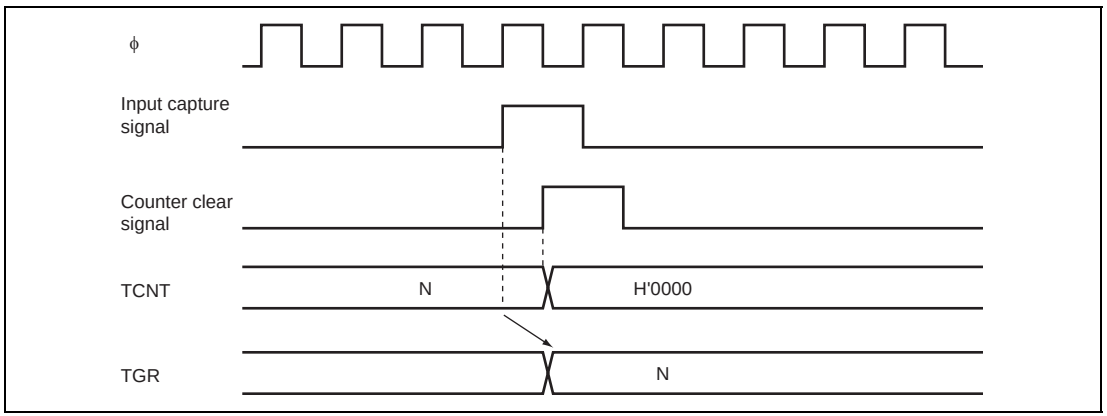



**Figure 9.35 Counter Clear Timing (Input Capture)**

**Buffer Operation Timing:** Figures 9.36 and 9.37 show the timing in buffer operation.

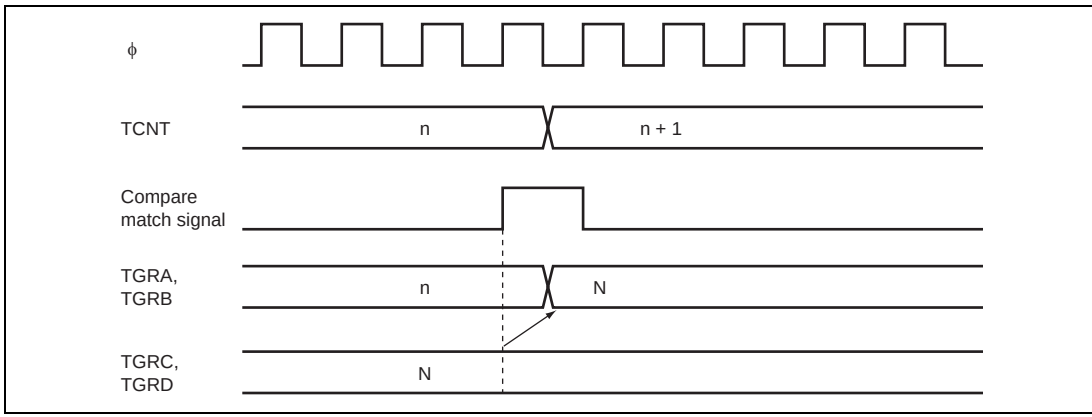



**Figure 9.36 Buffer Operation Timing (Compare Match)**

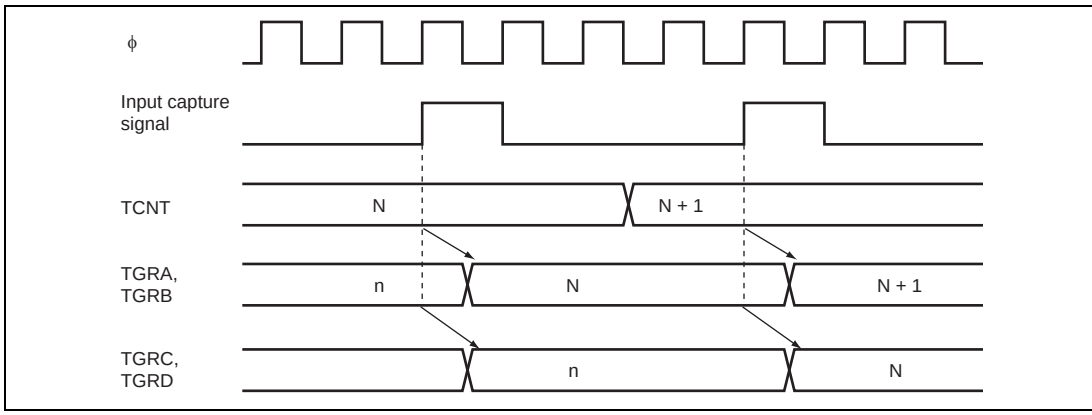



**Figure 9.37 Buffer Operation Timing (Input Capture)**

### 9.7.2 Interrupt Signal Timing

**TGF Flag Setting Timing in Case of Compare Match:** Figure 9.38 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and TGI interrupt request signal timing.

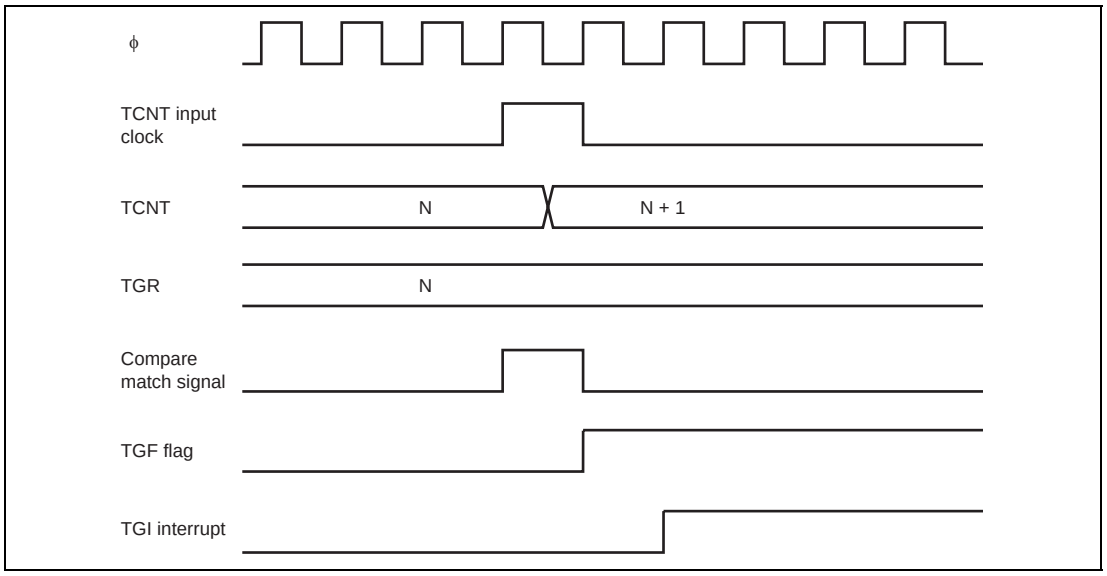



**Figure 9.38 TGI Interrupt Timing (Compare Match)**

**TGF Flag Setting Timing in Case of Input Capture:** Figure 9.39 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and TGI interrupt request signal timing.

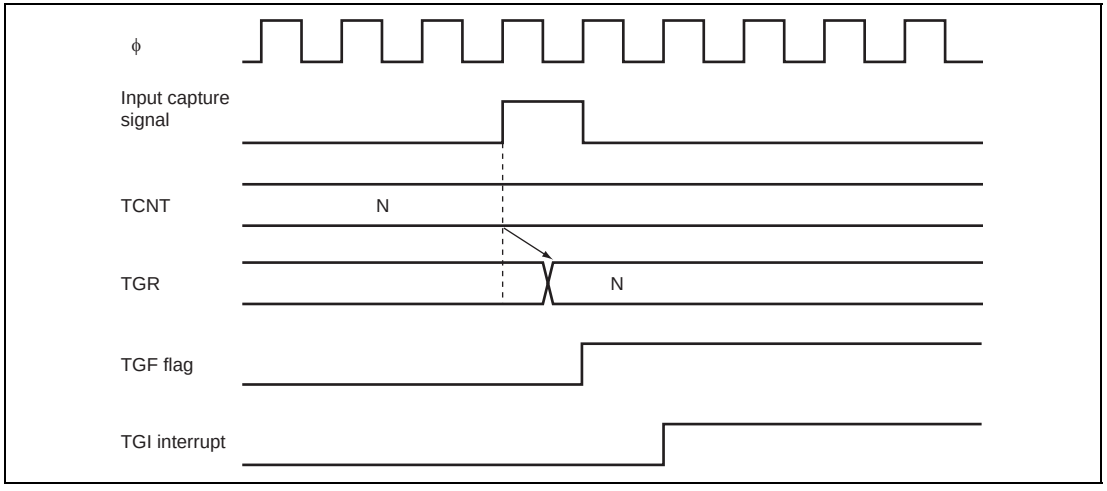



**Figure 9.39 TGI Interrupt Timing (Input Capture)**

**TCFV Flag/TCFU Flag Setting Timing:** Figure 9.40 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and TCIV interrupt request signal timing. Figure 9.41 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and TCIU interrupt request signal timing.

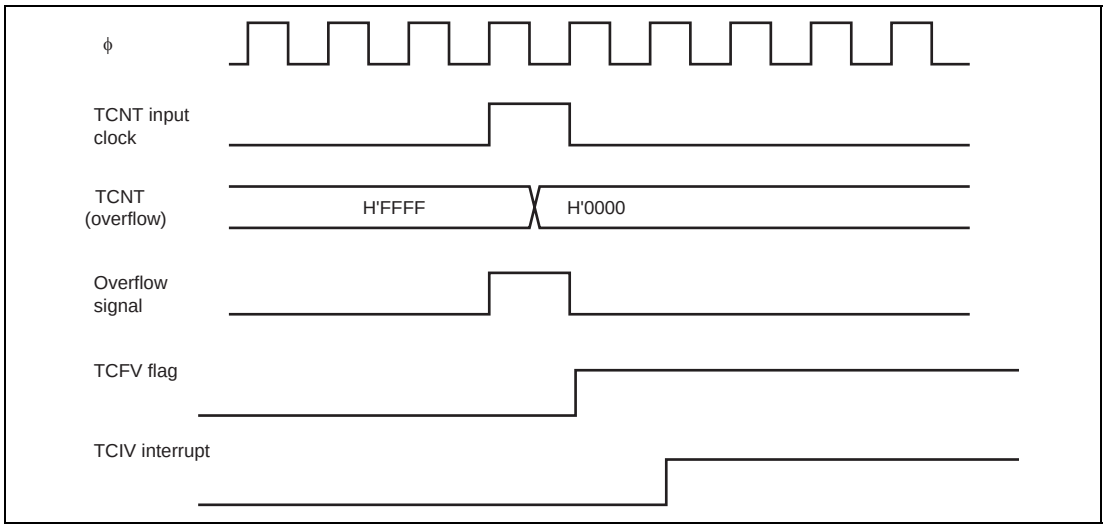



**Figure 9.40 TCIV Interrupt Setting Timing**

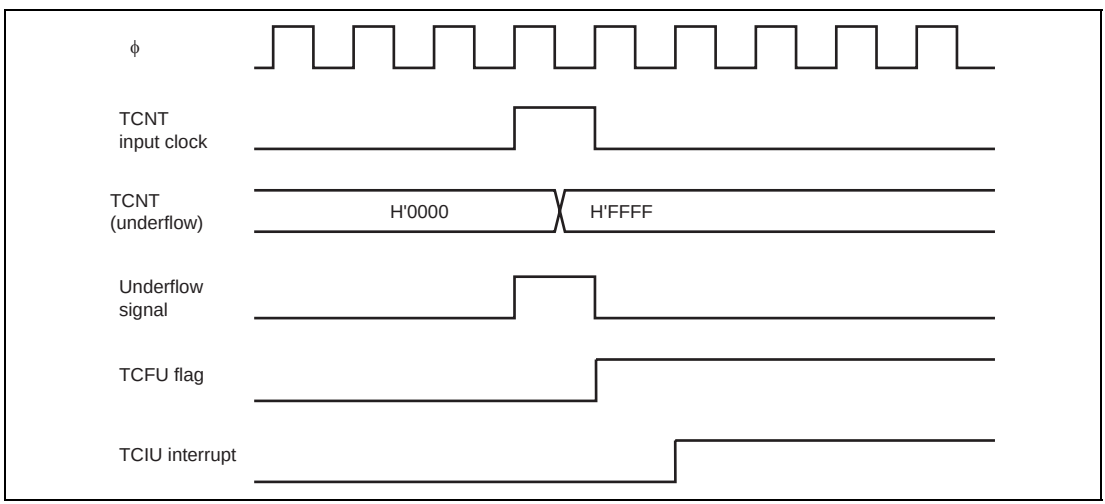



**Figure 9.41 TCIU Interrupt Setting Timing**

**Status Flag Clearing Timing:** After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DMAC is activated, the flag is cleared automatically. Figure 9.42 shows the timing for status flag clearing by the CPU, and figure 9.43 shows the timing for status flag clearing by the DMAC.

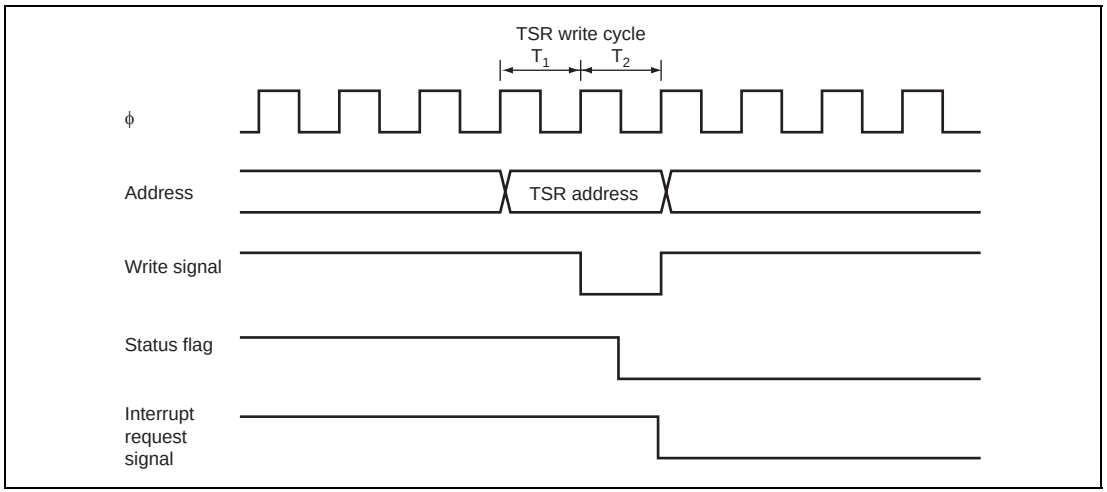



**Figure 9.42 Timing for Status Flag Clearing by CPU**

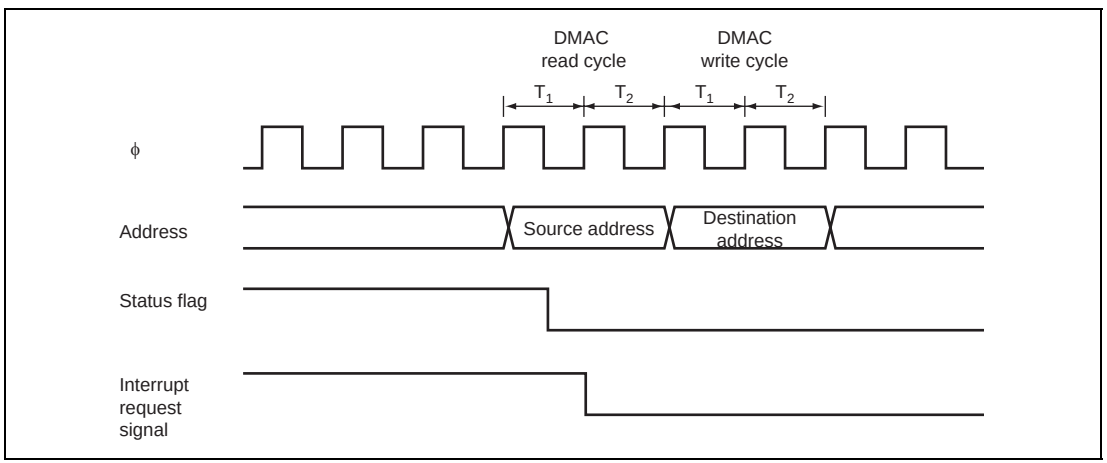



**Figure 9.43 Timing for Status Flag Clearing by DMAC Activation**

## 9.8 Usage Notes

**Input Clock Restrictions:** The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width. In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 9.44 shows the input clock conditions in phase counting mode.

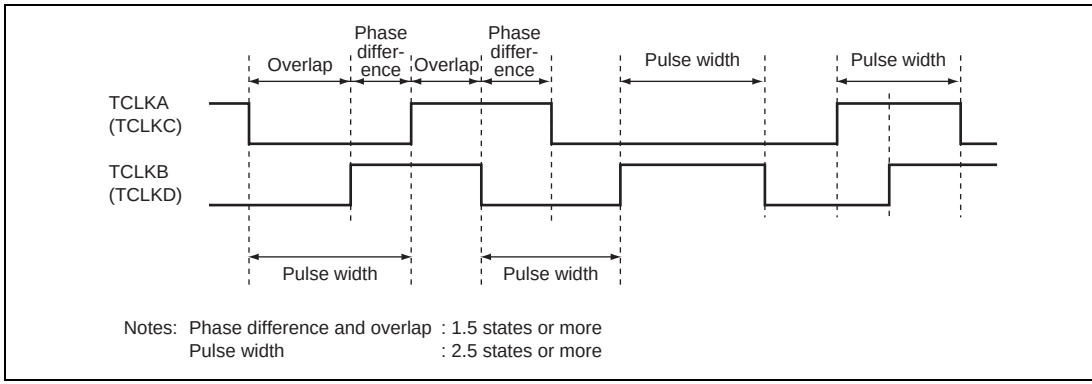



**Figure 9.44 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode**

**Caution on Period Setting:** When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

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

Where f : Counter frequency
$\phi$ : Operating frequency
N : TGR set value

**Contention between TCNT Write and Clear Operations:** If the counter clear signal is generated in the $T_2$ state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed. Figure 9.45 shows the timing in this case.

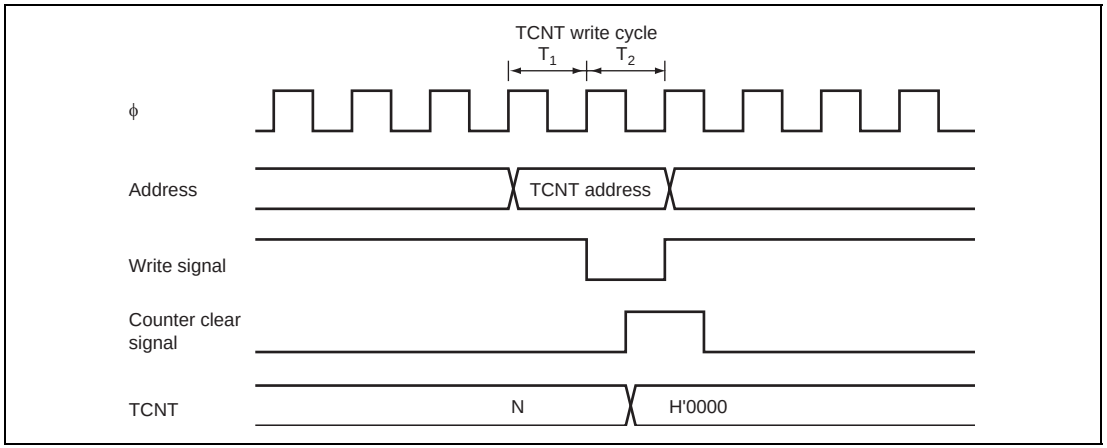



**Figure 9.45 Contention between TCNT Write and Clear Operations**

**Contention between TCNT Write and Increment Operations:** If incrementing occurs in the $T_2$ state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented. Figure 9.46 shows the timing in this case.

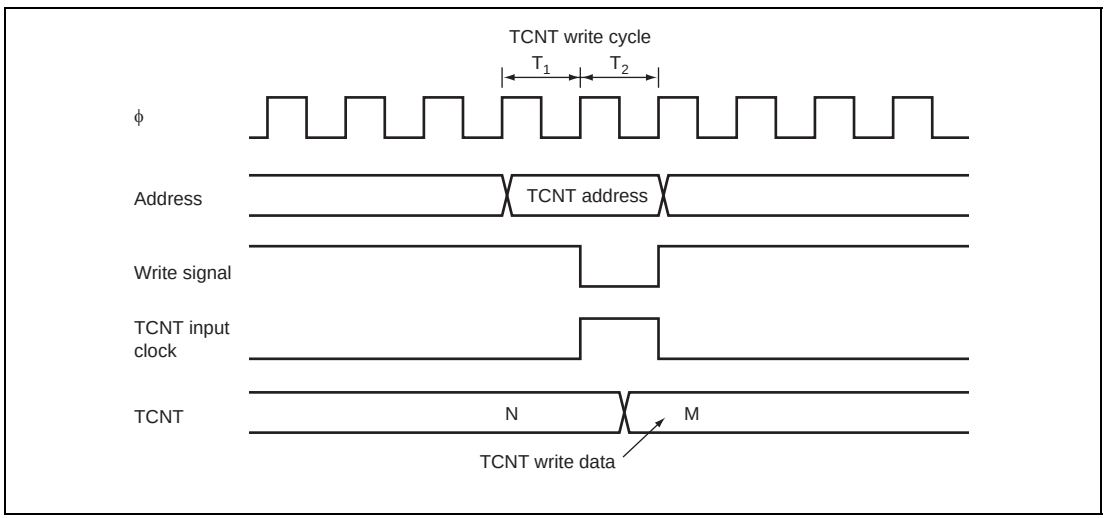



**Figure 9.46 Contention between TCNT Write and Increment Operations**

**Contention between TGR Write and Compare Match:** If a compare match occurs in the $T_2$ state of a TGR write cycle, the TGR write takes precedence and the compare match signal is inhibited. A compare match does not occur even if the same value as before is written. Figure 9.47 shows the timing in this case.

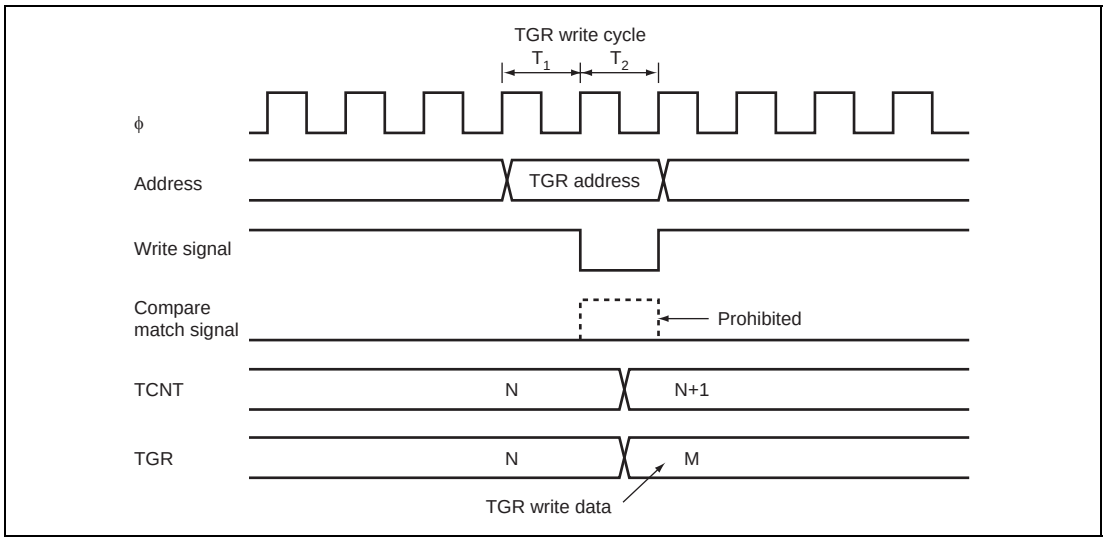



**Figure 9.47 Contention between TGR Write and Compare Match**

**Contention between Buffer Register Write and Compare Match:** If a compare match occurs in the $T_2$ state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the data prior to the write. Figure 9.48 shows the timing in this case.

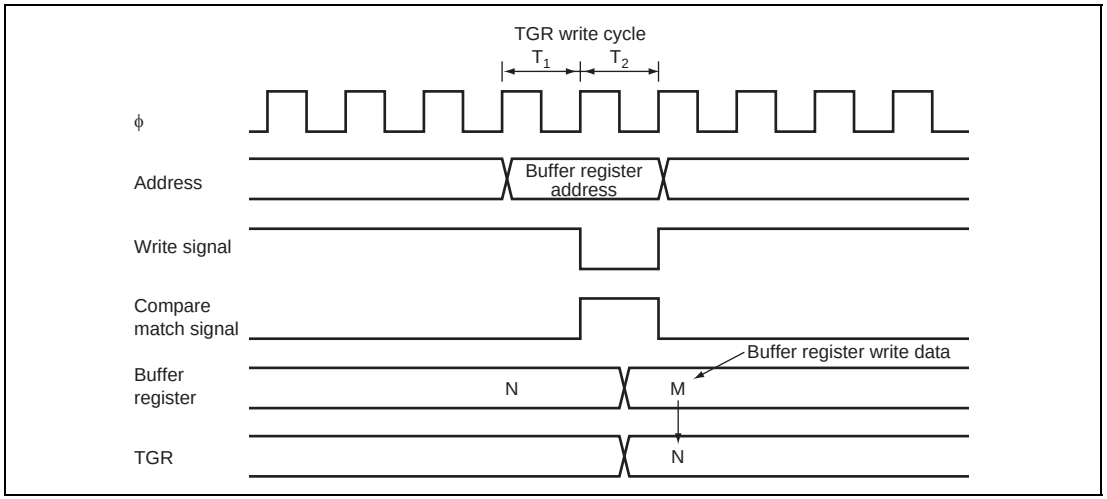



**Figure 9.48 Contention between Buffer Register Write and Compare Match**

**Contention between TGR Read and Input Capture:** If the input capture signal is generated in the $T_1$ state of a TGR read cycle, the data that is read will be the data after input capture transfer. Figure 9.49 shows the timing in this case.

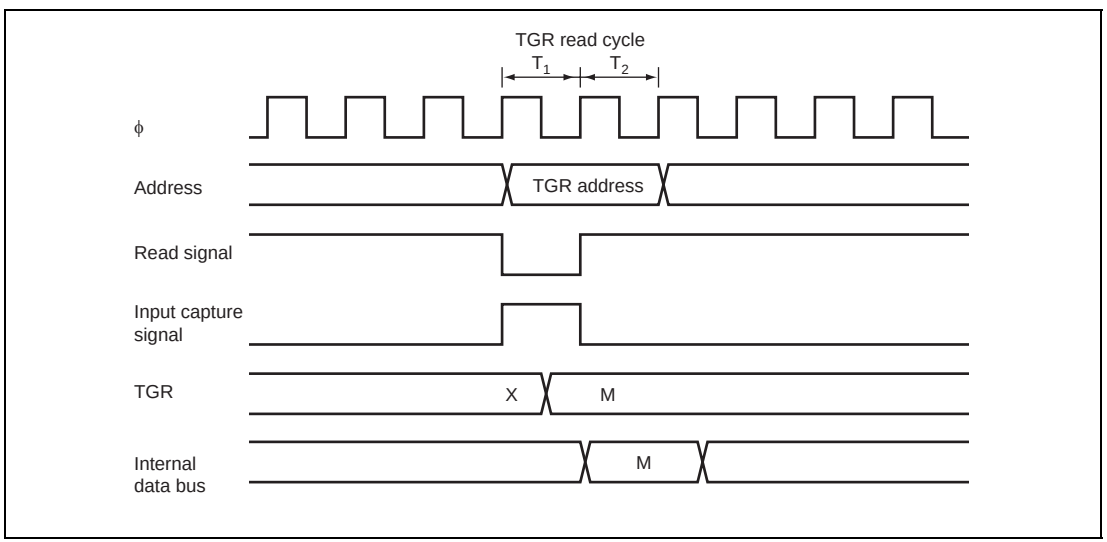



**Figure 9.49 Contention between TGR Read and Input Capture**

**Contention between TGR Write and Input Capture:** If the input capture signal is generated in the $T_2$ state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed. Figure 9.50 shows the timing in this case.

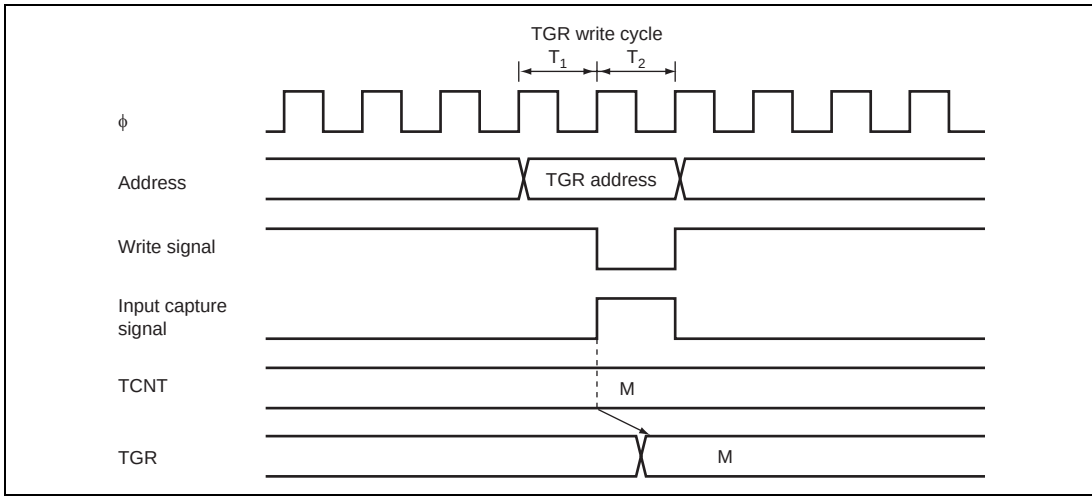



**Figure 9.50 Contention between TGR Write and Input Capture**

**Contention between Buffer Register Write and Input Capture:** If the input capture signal is generated in the $T_2$ state of a buffer register write cycle, the buffer operation takes precedence and the write to the buffer register is not performed. Figure 9.51 shows the timing in this case.

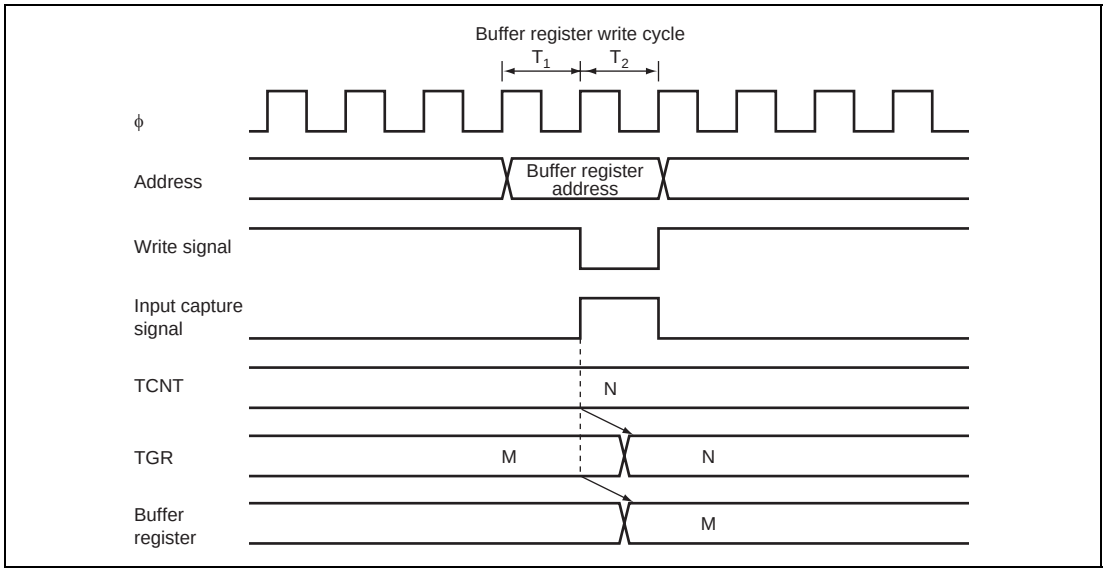



**Figure 9.51 Contention between Buffer Register Write and Input Capture**

**Contention between Overflow/Underflow and Counter Clearing:** If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence. Figure 9.52 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.

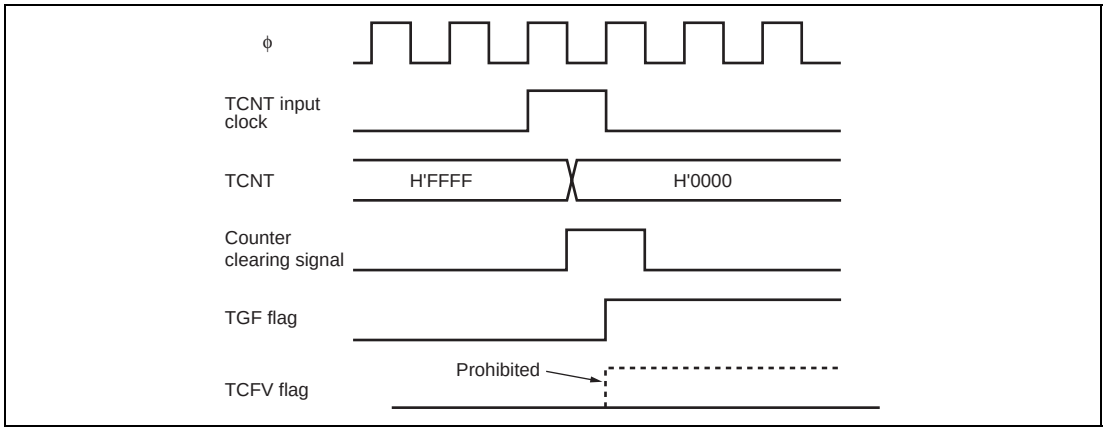



**Figure 9.52 Contention between Overflow and Counter Clearing**

**Contention between TCNT Write and Overflow/Underflow:** If there is an up-count or down-count in the $T_2$ state of a TCNT write cycle, and overflow/underflow occurs, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set. Figure 9.53 shows the operation timing when there is contention between TCNT write and overflow.

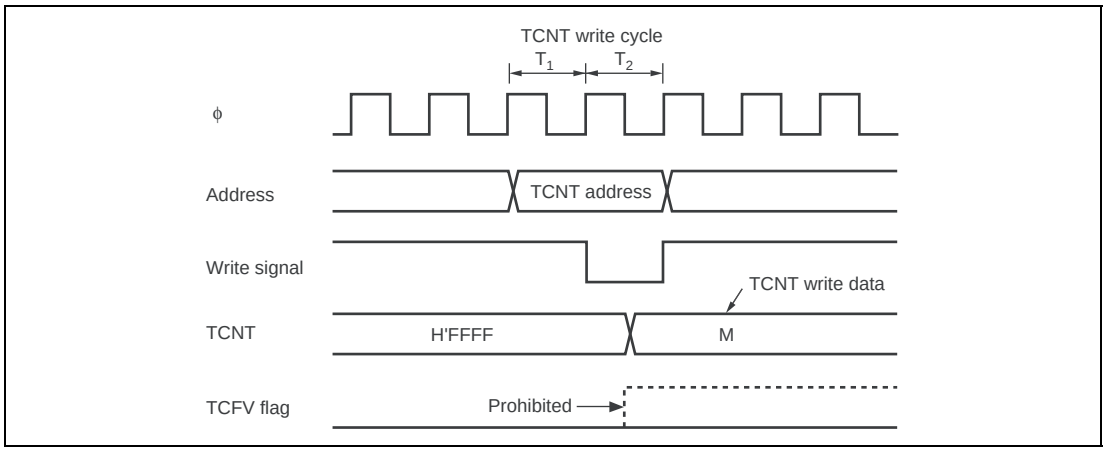



**Figure 9.53 Contention between TCNT Write and Overflow**

**Multiplexing of I/O Pins:** In this LSI, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

**Interrupts in Module Stop Mode:** If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC activation source. Interrupts should therefore be disabled before entering module stop mode.

**Module Stop Mode Setting:** TPU operation can be disabled or enabled using the module stop control register. The initial setting is for TPU operation to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 20, Power-Down Modes.

# Section 10 Watchdog Timer (WDT)

The watchdog timer (WDT) is an 8-bit timer that can generate an internal reset signal for this LSI if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow. When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

The block diagram of the WDT is shown in figure 10.1.

## 10.1 Features

- Selectable from eight counter input clocks
- Switchable between watchdog timer mode and interval timer mode

In watchdog timer mode:

- If the counter overflows, it is possible to select whether this LSI is internally reset or not.

In interval timer mode:

- If the counter overflows, the WDT generates an interval timer interrupt (WOVI).

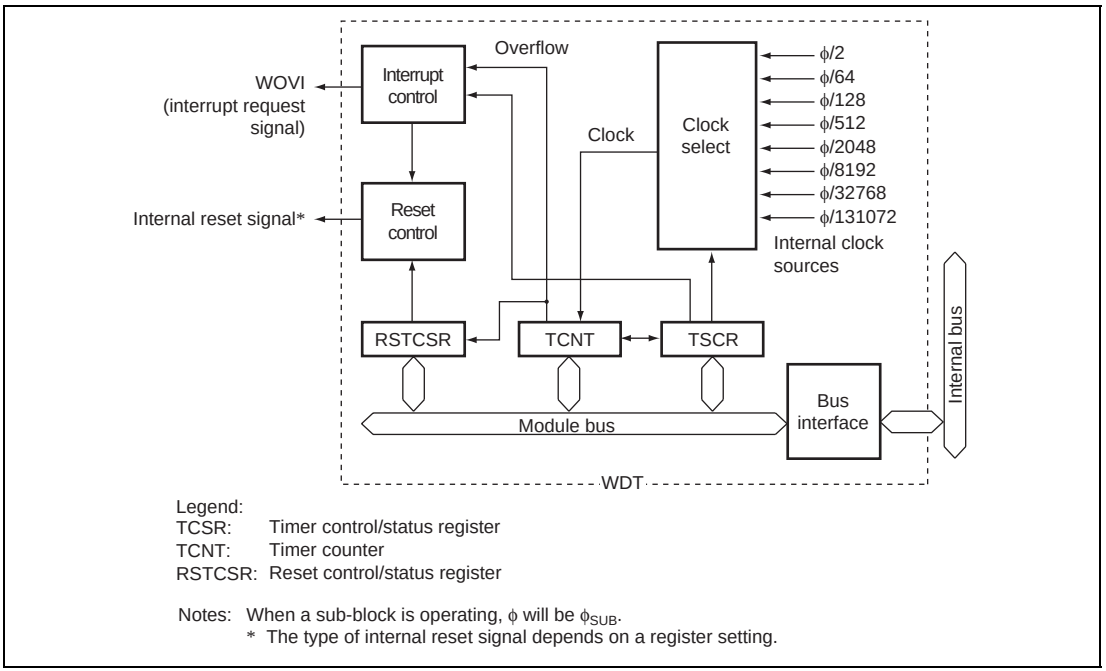



**Figure 10.1 Block Diagram of WDT**

## 10.2 Register Descriptions

The WDT has the following three registers. For details, refer to section 21, List of Registers. To prevent accidental overwriting, TCSR, TCNT, and RSTCSR have to be written to by a different method to normal registers. For details, refer to section 10.5.1, Notes on Register Access.

- Timer counter (TCNT)
- Timer control/status register (TCSR)
- Reset control/status register (RSTCSR)

### 10.2.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 by a reset, when the TME bit in TCSR is cleared to 0.

### 10.2.2 Timer Control/Status Register (TCSR)

TCSR is an 8-bit readable/writable register. Its functions include selecting the clock source to be input to TCNT, and selecting the timer mode.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | OVF | 0 | R/(W)* | Overflow Flag<br>Indicates that TCNT has overflowed. Only a write of 0 is permitted, to clear the flag.<br>[Setting condition]<br>When TCNT overflows (changes from H'FF to H'00)<br>When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset.<br>[Clearing condition]<br>Cleared by reading TCSR when OVF = 1, then writing 0 to OVF<br>When polling CVF when the interval timer interrupt has been prohibited, OVF = 1 status should be read two or more times. |
| 6 | WT/$\overline{IT}$ | 0 | R/W | Timer Mode Select<br>Selects whether the WDT is used as a watchdog timer or interval timer.<br>0: Interval timer mode<br>1: Watchdog timer mode |
| 5 | TME | 0 | R/W | Timer Enable<br>When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00. |
| 4, 3 | — | All 1 | — | Reserved<br>These bits are always read as 1 and cannot be modified. |
| 2 | CKS2 | 0 | R/W | Clock Select 0 to 2<br>Selects the clock source to be input to TCNT. The overflow frequency for $\phi$ = 16 MHz is enclosed in parentheses. The overflow period is the time from when TCNT starts counting up from H'00 until overflow occurs.<br>000: Clock $\phi$/2 (frequency: 32.0 $\mu$s)<br>001: Clock $\phi$/64 (frequency: 1.0 ms)<br>010: Clock $\phi$/128 (frequency: 2.0 ms)<br>011: Clock $\phi$/512 (frequency: 8.2 ms)<br>100: Clock $\phi$/2048 (frequency: 32.8 ms)<br>101: Clock $\phi$/8192 (frequency: 131.1 ms)<br>110: Clock $\phi$/32768 (frequency: 524.3 ms)<br>111: Clock $\phi$/131072 (frequency: 2.1 s) |
| 1 | CKS1 | 0 | R/W | |
| 0 | CKS0 | 0 | R/W | |

Note: * The write value should always be 0 to clear this flag.

### 10.2.3 Reset Control/Status Register (RSTCSR)

RSTCSR is an 8-bit readable/writable register that controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal. RSTCSR is initialized to H'1F by a reset signal from the $\overline{\text{RES}}$ pin, and not by the WDT internal reset signal caused by overflows.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | WOVF | 0 | R/(W)*[1] | Watchdog Overflow Flag<br>This bit is set when TCNT overflows in watchdog timer mode. This bit cannot be set in interval timer mode, and the write value should always be 0.<br>[Setting condition]<br>Set when TCNT overflows (changed from H'FF to H'00) in watchdog timer mode<br>[Clearing condition]<br>Cleared by reading RSTCSR when WOVF = 1, and then writing 0 to WOVF |
| 6 | RSTE | 0 | R/W | Reset Enable<br>Specifies whether or not a reset signal is generated in the chip if TCNT overflows during watchdog timer operation.<br>0: Reset signal is not generated even if TCNT overflows<br>(Though this LSI is not reset, TCNT and TCSR in WDT are reset)<br>1: Reset signal is generated if TCNT overflows |
| 5 | RSTS | 0 | R/W | Reset Select<br>Selects the type of internal reset generated if TCNT overflows during watchdog timer operation.<br>0: Power-on reset*[2]<br>1: Manual reset*[3] |
| 4 to 0 | — | All 1 | — | Reserved<br>These bits are always read as 1 and cannot be modified. |

Notes: 1. The write value should always be 0 to clear this flag.
2. Bear in mind that the USB register is not initialized by a power-on reset trigged by the WDT. For details, see section 14.8.8, Reset.
3. Supported only by the H8S/2218 Group.

## 10.3 Operation

### 10.3.1 Watchdog Timer Mode

To use the WDT as a watchdog timer, set the WT/$\overline{IT}$ bit in TCSR and the TME bit to 1.

TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally be writing H'00) before overflows occurs.

When the RSTE bit of the RSTCSR is set to 1, and if the TCNT overflows, an internal reset signal for this LSI is issued. In this case, select power-on reset or manual reset* by setting the RSTS bit of the RSTCSR to 0.

If a reset caused by a signal input to the $\overline{RES}$ pin occurs at the same time as a reset caused by a WDT overflow, the $\overline{RES}$ pin reset has priority and the WOVF bit in RSTCSR is cleared to 0. The internal reset signal is output for 518 states.

When the TCNT overflows in watchdog timer mode, the WOVF bit of the RSTCSR is set to 1. If the RSTE bit of the RSTCSR has been set to 1, an internal reset signal for the entire LSI is generated at TCNT overflow.

Note: * Supported only by the H8S/2218 Group.

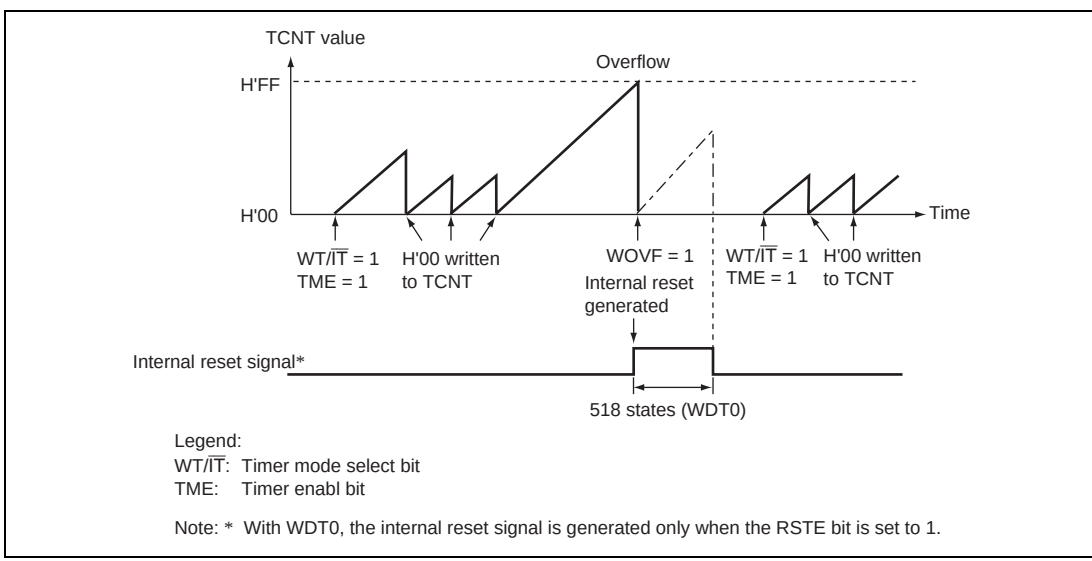



**Figure 10.2 Operation in Watchdog Timer Mode**

### 10.3.2 Timing of Setting of Watchdog Timer Overflow Flag (WOVF)

With WDT, the WOVF bit in RSTCSR is set to 1 if TCNT overflows in watchdog timer mode. If TCNT overflows while the RSTE bit in RSTCSR is set to 1, an internal reset signal is generated for the entire chip. This timing is illustrated in figure 10.3.

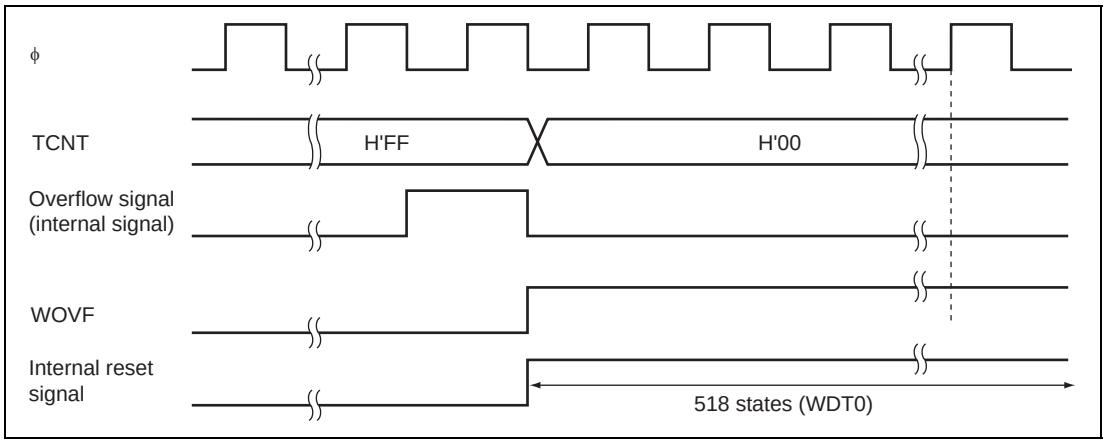



**Figure 10.3 Timing of WOVF Setting**

### 10.3.3 Interval Timer Mode

To use the WDT as an interval timer, clear bit WT/$\overline{IT}$ in TCSR to 0 and set bit TME to 1. When the interval timer is operating, an interval timer interrupt (WOVI) is generated each time the TCNT overflows. Therefore, an interrupt can be generated at intervals.

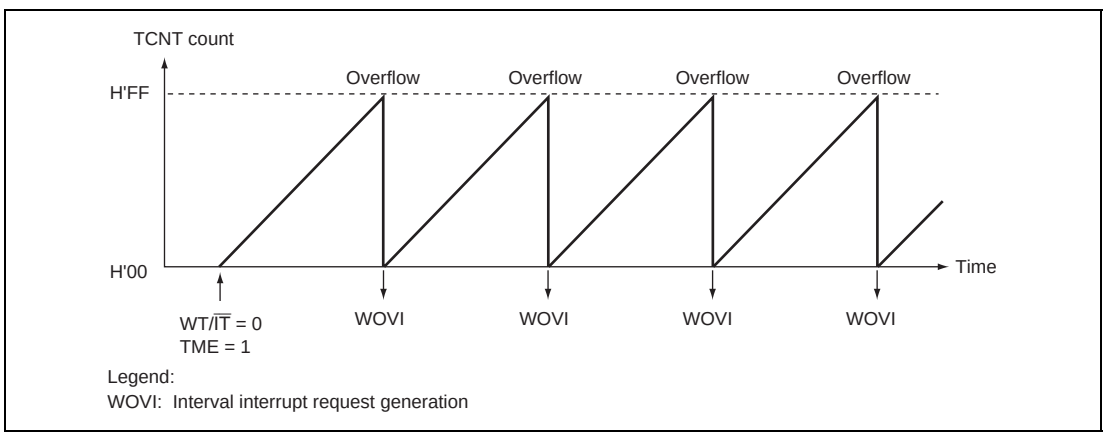



**Figure 10.4 Operation in Interval Timer Mode**

### 10.3.4 Timing of Setting of Overflow Flag (OVF)

The OVF flag is set to 1 if TCNT overflows during interval timer operation. At the same time, an interval timer interrupt (WOVI) is requested. This timing is shown in figure 10.5.

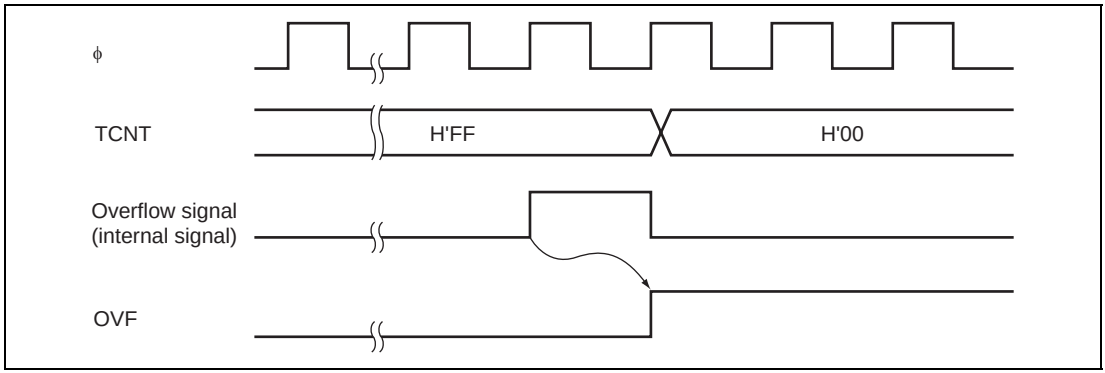



**Figure 10.5 Timing of OVF Setting**

## 10.4 Interrupts

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR. OVF must be cleared to 0 in the interrupt handling routine.

**Table 10.1 WDT Interrupt Source**

| Name | Interrupt Source | Interrupt Flag |
|---|---|---|
| WOVI | TCNT overflow | WOVF |

## 10.5 Usage Notes

### 10.5.1 Notes on Register Access

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

**Writing to TCNT and TCSR:** These registers must be written to by a word transfer instruction. They cannot be written to with byte transfer instructions. Figure 10.6 shows the format of data written to TCNT and TCSR. TCNT and TCSR both have the same write address. For a write to TCNT, the upper byte of the written word must contain H'5A and the lower byte must contain the write data. For a write to TCSR, the upper byte of the written word must contain H'A5 and the lower byte must contain the write data. This transfers the write data from the lower byte to TCNT or TCSR.

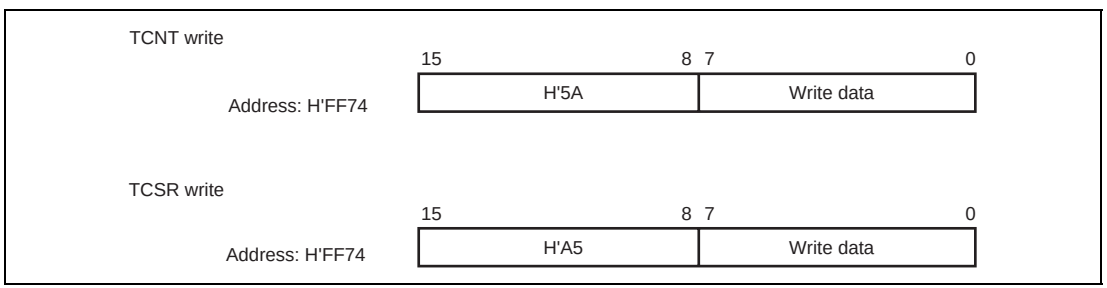



**Figure 10.6 Format of Data Written to TCNT and TCSR**

**Writing to RSTCSR:** RSTCSR must be written to by a word transfer to address H'FF76. It cannot be written to with byte instructions. Figure 10.7 shows the format of data written to RSTCSR. The method of writing 0 to the WOVF bit differs from that for writing to the RSTE and RSTS bits.

To write 0 to the WOVF bit, the upper byte of the written word must contain H'A5 and the lower byte must contain H'00. This clears the WOVF bit to 0, but has no effect on the RSTE and RSTS bits. To write to the RSTE and RSTS bits, the upper byte must contain H'5A and the lower byte must contain the write data. This writes the values in bits 6 and 5 of the lower byte into the RSTE and RSTS bits, but has no effect on the WOVF bit.

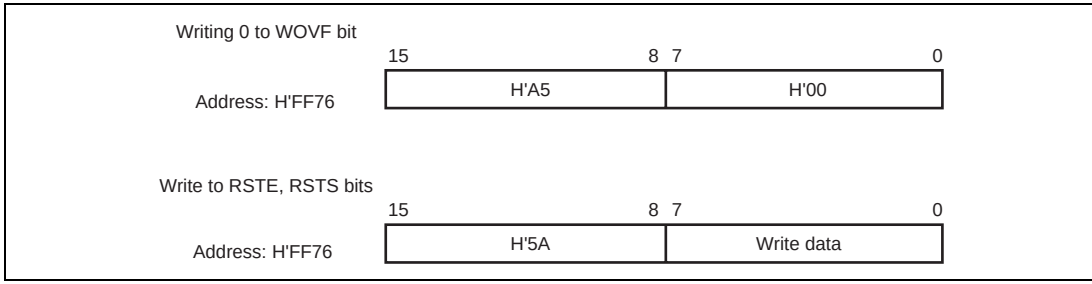



**Figure 10.7 Format of Data Written to RSTCSR (Example of WDT0)**

**Reading from TCNT, TCSR, and RSTCSR:** TCNT, TCSR, and RSTCSR are read by using the same method as for the general registers. TCSR, TCNT, and RSTCSR are allocated in addresses H'FF74, H'FF75, and H'FF77 respectively.

### 10.5.2 Contention between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the $T_2$ state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 10.8 shows this operation.

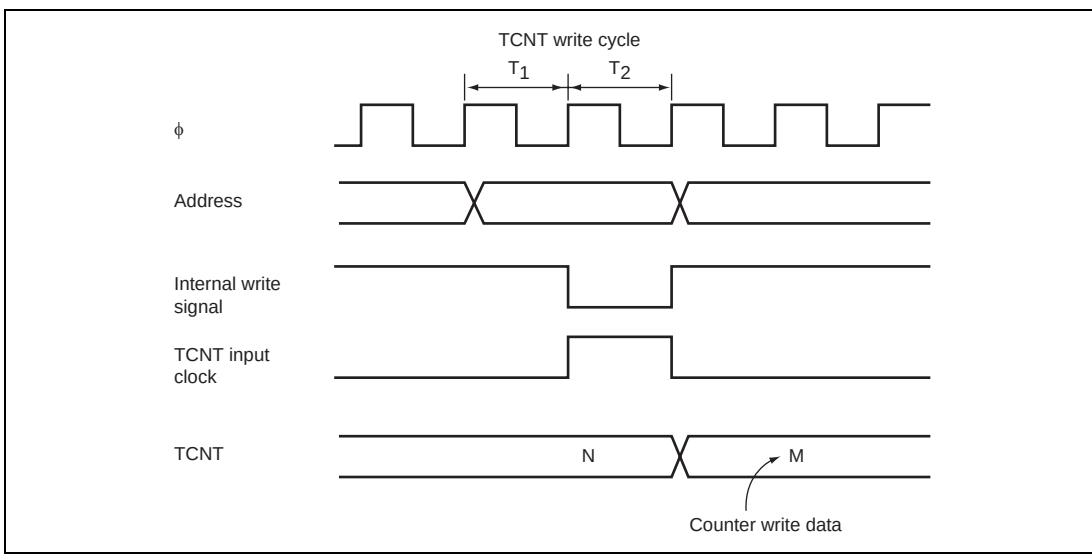



**Figure 10.8 Contention between TCNT Write and Increment**

### 10.5.3 Changing Value of CKS2 to CKS0

If bits CKS0 to CKS2 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. Software must be used to stop the watchdog timer (by clearing the TME bit to 0) before changing the value of bits CKS0 to CKS2.

### 10.5.4 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer while the WDT is operating, errors could occur in the incrementation. Software must be used to stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

### 10.5.5 Internal Reset in Watchdog Timer Mode

This LSI is not reset internally if TCNT overflows while the RSTE bit is cleared to 0 during watchdog timer operation, however TCNT and TCSR of the WDT are reset.

TCNT, TCSR, or RSTCR cannot be written to for 132 states following an overflow. During this period, any attempt to read the WOVF flag is not acknowledged. Accordingly, wait 132 states after overflow to write 0 to the WOVF flag for clearing.

### 10.5.6 OVF Flag Clearing in Interval Timer Mode

When the OVF flag setting conflicts with the OVF flag reading in interval timer mode, writing 0 to the OVF bit may not clear the flag even though the OVF bit has been read while it is 1. If there is a possibility that the OVF flag setting and reading will conflict, such as when the OVF flag is polled with the interval timer interrupt disabled, read the OVF bit while it is 1 at least twice before writing 0 to the OVF bit to clear the flag.

# Section 11 Realtime Clock (RTC)

The realtime clock (RTC) is a timer used to count time ranging from a second to a week. Figure 11.1 shows the block diagram of the RTC.

## 11.1 Features

- Counts seconds, minutes, hours, and day-of-week
- Start/stop function
- Reset function
- Readable/writable counter of seconds, minutes, hours, and day-of-week with BCD codes
- Periodic (seconds, minutes, hours, days, and weeks) interrupts
- 8-bit free running counter
- Selection of clock source
- 8-bit bus (3 cycle access timing) connected to the external bus
  RTC register is allocated to a part of area 7 of external address (H'FFFF48 to H'FFFF4F)

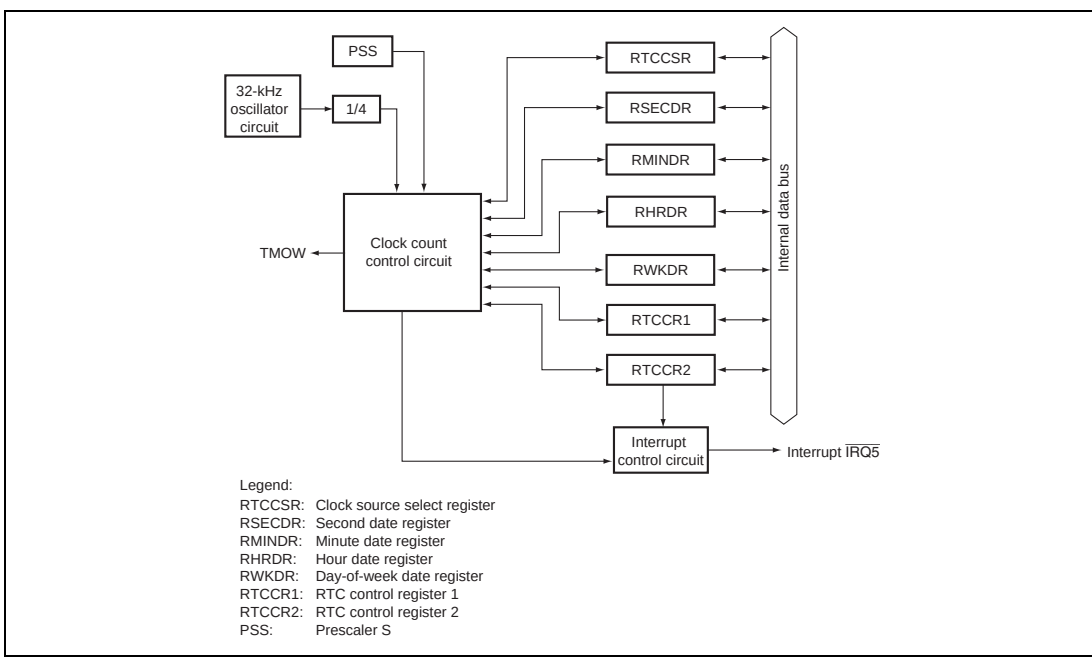



**Figure 11.1 Block Diagram of RTC**

## 11.2 Input/Output Pin

Table 11.1 shows the RTC input/output pin.

**Table 11.1 Pin Configuration**

| Name | Abbreviation | I/O | Function |
|---|---|---|---|
| Clock output | TMOW | Output | RTC divided clock output |

## 11.3 Register Descriptions

The RTC has the following registers.

- Second data register (RSECDR)
- Minute data register (RMINDR)
- Hour data register (RHRDR)
- Day-of-week data register (RWKDR)
- RTC control register 1 (RTCCR1)
- RTC control register 2 (RTCCR2)
- Clock source select register (RTCCSR)
- Extended module stop register (EXMDLSTP)

### 11.3.1 Second Data Register (RSECDR)

RSECDR counts the BCD-coded second value. This register is initialized to H'00 by a $\overline{\text{STBY}}$ input or the RST bit in RTCCR1, but not initialized by a $\overline{\text{RES}}$ input. The setting range is decimal 00 to 59. For more information on reading seconds, minutes, hours, and day-of-week, see section 11.4.2, Time Data Reading Procedure.

| Bit | Bit Name | Initial Value* | R/W | Description |
|---|---|---|---|---|
| 7 | BSY | — | R | RTC Busy<br>This bit is set to 1 when the RTC is updating (operating) the values of second, minute, hour, and day-of-week data registers. When this bit is 0, the values of second, minute, hour, and day-of-week data registers must be adopted. |
| 6 | SC12 | — | R/W | Counting Ten's Position of Seconds |
| 5 | SC11 | — | R/W | Counts on 0 to 5 for 60-second counting. |
| 4 | SC10 | — | R/W | |
| 3 | SC03 | — | R/W | Counting One's Position of Seconds |
| 2 | SC02 | — | R/W | Counts on 0 to 9 once per second. When a carry is generated, 1 is added to the ten's position. |
| 1 | SC01 | — | R/W | |
| 0 | SC00 | — | R/W | |

Note: * Initial value after $\overline{\text{RES}}$.

### 11.3.2 Minute Data Register (RMINDR)

RMINDR counts the BCD-coded minute value on the carry generated once per minute by the RSECDR counting. This register is initialized to H'00 a $\overline{\text{STBY}}$ input or the RST bit in RTCCR1, but not initialized by a $\overline{\text{RES}}$ input. The setting range is decimal 00 to 59.

| Bit | Bit Name | Initial Value* | R/W | Description |
|---|---|---|---|---|
| 7 | BSY | — | R | RTC Busy<br>This bit is set to 1 when the RTC is updating (operating) the values of second, minute, hour, and day-of-week data registers. When this bit is 0, the values of second, minute, hour, and day-of-week data registers must be adopted. |
| 6 | MN12 | — | R/W | Counting Ten's Position of Minutes |
| 5 | MN11 | — | R/W | Counts on 0 to 5 for 60-minute counting. |
| 4 | MN10 | — | R/W | |
| 3 | MN03 | — | R/W | Counting One's Position of Minutes |
| 2 | MN02 | — | R/W | Counts on 0 to 9 once per minute. When a carry is generated, 1 is added to the ten's position. |
| 1 | MN01 | — | R/W | |
| 0 | MN00 | — | R/W | |

Note: * Initial value after $\overline{\text{RES}}$.

### 11.3.3 Hour Data Register (RHRDR)

RHRDR counts the BCD-coded hour value on the carry generated once per hour by RMINDR. This register is initialized to H'00 by a $\overline{\text{STBY}}$ input or the RST bit in RTCCR1, but not initialized by a $\overline{\text{RES}}$ input. The setting range is either decimal 0 to 11 or 0 to 23 by the selection of the 12/24 bit in RTCCR1.

| Bit | Bit Name | Initial Value* | R/W | Description |
|---|---|---|---|---|
| 7 | BSY | — | R | RTC Busy<br>This bit is set to 1 when the RTC is updating (operating) the values of second, minute, hour, and day-of-week data registers. When this bit is 0, the values of second, minute, hour, and day-of-week data registers must be adopted. |
| 6 | — | 0 | — | Reserved<br>This bit is always read as 0. |
| 5 | HR11 | — | R/W | Counting Ten's Position of Hours |
| 4 | HR10 | — | R/W | Counts on 0 to 2 for ten's position of hours. |
| 3 | HR03 | — | R/W | Counting One's Position of Hours |
| 2 | HR02 | — | R/W | Counts on 0 to 9 once per hour. When a carry is generated, 1 is added to the ten's position. |
| 1 | HR01 | — | R/W | |
| 0 | HR00 | — | R/W | |

Note: * Initial value after $\overline{\text{RES}}$.

### 11.3.4 Day-of-Week Data Register (RWKDR)

RWKDR counts the BCD-coded day-of-week value on the carry generated once per day by RHRDR. This register is initialized to H'00 by a $\overline{\text{STBY}}$ input or the RST bit in RTCCR1, but not initialized by a $\overline{\text{RES}}$ input. The setting range is decimal 0 to 6 using bits WK2 to WK0.

| Bit | Bit Name | Initial Value* | R/W | Description |
|---|---|---|---|---|
| 7 | BSY | — | R | RTC Busy<br>This bit is set to 1 when the RTC is updating (operating) the values of second, minute, hour, and day-of-week data registers. When this bit is 0, the values of second, minute, hour, and day-of-week data registers must be adopted. |
| 6 to 3 | — | All 0 | — | Reserved<br>These bits are always read as 0. |
| 2 | WK2 | — | R/W | Day-of-Week Counting |
| 1 | WK1 | — | R/W | Day-of-week is indicated with a binary code. |
| 0 | WK0 | — | R/W | 000: Sunday<br>001: Monday<br>010: Tuesday<br>011: Wednesday<br>100: Thursday<br>101: Friday<br>110: Saturday<br>111: Reserved (Setting prohibited) |

Note: * Initial value after $\overline{\text{RES}}$.

### 11.3.5 RTC Control Register 1 (RTCCR1)

RTCCR1 controls start/stop and reset of the clock timer. Bits 7 to 5 of this register are initialized to H'00 by a $\overline{\text{STBY}}$ input or the RST bit in RTCCR1, but not initialized by a $\overline{\text{RES}}$ input. For the definition of time expression, see figure 11.2.

| Bit | Bit Name | Initial Value* | R/W | Description |
|---|---|---|---|---|
| 7 | RUN | — | R/W | RTC Operation Start<br>0: Stops RTC and free running counter operation<br>1: Starts RTC and free running counter operation |
| 6 | 12/24 | — | R/W | Operating Mode<br>0: RTC operates in 12-hour mode. RHRDR counts on 0 to 11.<br>1: RTC operates in 24-hour mode. RHRDR counts on 0 to 23. |
| 5 | PM | — | R/W | A.M./P.M.<br>0: Indicates a.m. when RTC is in the 12-hour mode.<br>1: Indicates p.m. when RTC is in the 12-hour mode. |
| 4 | RST | 0 | R/W | Reset<br>0: Normal operation<br>1: Resets registers and control circuits except RTCCSR and this bit. Clear this bit to 0 after having been set to 1. |
| 3 to 0 | — | All 0 | — | Reserved<br>These bits are always read as 0. |

Note: * Initial value after $\overline{\text{RES}}$.

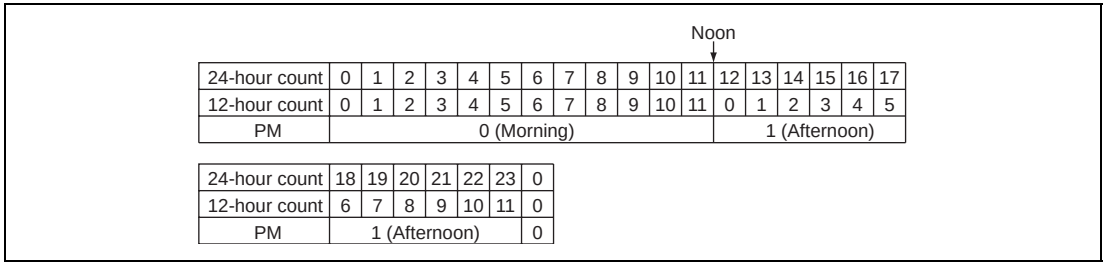



**Figure 11.2 Definition of Time Expression**

### 11.3.6 RTC Control Register 2 (RTCCR2)

RTCCR2 controls RTC periodic interrupts of weeks, days, hours, minutes, and seconds. This register is initialized to H'00 by a $\overline{\text{STBY}}$ input or the RST bit in RTCCR1, but not initialized by a $\overline{\text{RES}}$ input. Enabling interrupts of weeks, days, hours, minutes, and seconds sets the IRRTA flag to 1 in the interrupt flag register 1 (IRR1) when an interrupt occurs. It also controls an overflow interrupt of a free running counter when RTC operates as a free running counter.

| Bit | Bit Name | Initial Value* | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 0 | — | Reserved<br>These bits are always read as 0. |
| 5 | FOIE | — | R/W | Free Running Counter Overflow Interrupt Enable<br>0: Disables an overflow interrupt<br>1: Enables an overflow interrupt |
| 4 | WKIE | — | R/W | Week Periodic Interrupt Enable<br>0: Disables a week periodic interrupt<br>1: Enables a week periodic interrupt |
| 3 | DYIE | — | R/W | Day Periodic Interrupt Enable<br>0: Disables a day periodic interrupt<br>1: Enables a day periodic interrupt |
| 2 | HRIE | — | R/W | Hour Periodic Interrupt Enable<br>0: Disables an hour periodic interrupt<br>1: Enables an hour periodic interrupt |
| 1 | MNIE | — | R/W | Minute Periodic Interrupt Enable<br>0: Disables a minute periodic interrupt<br>1: Enables a minute periodic interrupt |
| 0 | SEIE | — | R/W | Second Periodic Interrupt Enable<br>0: Disables a second periodic interrupt<br>1: Enables a second periodic interrupt |

Note: * Initial value after $\overline{\text{RES}}$.

### 11.3.7 Clock Source Select Register (RTCCSR)

RTCCSR selects clock source. This register is initialized to H'08 by a $\overline{\text{STBY}}$ input or $\overline{\text{RES}}$ input. A free running counter controls start/stop of counter operation by the RUN bit in RTCCR1. When a clock other than 32.768 MHz is selected, the RTC is disabled and operates as an 8-bit free running counter. An interrupt can be generated by setting 1 to the FOIE bit in RTCCR2 and enabling an overflow interrupt of the free running counter. A clock in which the system clock is divided by 32, 16, 8, or 4 is output in high-speed mode, medium-speed mode, sleep mode, subactive mode, or subsleep mode.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | 0 | — | Reserved<br>This bit is always read as 0. |
| 6 | RCS6 | 0 | R/W | Clock Output Selection |
| 5 | RCS5 | 0 | R/W | Selects a clock output from the TMOW pin when the TMOWE bit in UCTLR is set to 1.<br>00: $\phi$/4<br>01: $\phi$/8<br>10: $\phi$/16<br>11: $\phi$/32 |
| 4 | — | 0 | — | Reserved<br>This bit is always read as 0. |
| 3 | RCS3 | 1 | R/W | Clock Source Selection |
| 2 | RCS2 | 0 | R/W | 0000: $\phi$/8····· Free running counter operation |
| 1 | RCS1 | 0 | R/W | 0001: $\phi$/32····· Free running counter operation |
| 0 | RCS0 | 0 | R/W | 0010: $\phi$/128····· Free running counter operation<br>0011: $\phi$/256····· Free running counter operation<br>0100: $\phi$/512····· Free running counter operation<br>0101: $\phi$/2048····· Free running counter operation<br>0110: $\phi$/4096····· Free running counter operation<br>0111: $\phi$/8192····· Free running counter operation<br>1000: 32.768 kHz·····RTC operation |

### 11.3.8 Extended Module Stop Register (EXMDLSTP)

EXMDLSTP controls the clock supply of the RTC and USB.

Note: When reading pin states using the port D register (PORTD), after accessing EXMDLSTP (address range: H'FFFF40 to H'FFFF5F), you must perform a dummy read to the external address space (such as H'FFEF00 to H'FF7FF) outside the range H'FFFF40 to H'FFFF5F before reading PORTD.

| Bit | Bit Name | Initial Value | R/W | Module |
|---|---|---|---|---|
| 7 to 2 | — | Undefined | — | Reserved<br>These bits are always read as undefined values. These bits should not to be modified. |
| 1 | RTCSTOP | 0 | R/W | RTC Module Stop<br>0: RTC module stop cancelled<br>1: RTC module stop |
| 0 | USBSTOP1 | 0 | R/W | USB Module Stop<br>0: USB module stop partly cancelled<br>1: USB module completely stop |

## 11.4 Operation

### 11.4.1 Initial Settings of Registers after Power-On and Resetting Procedure

The RTC registers that store second, minute, hour, day-of week, operating mode, and A.M./P.M. data are not reset by an $\overline{\text{STBY}}$ input. Therefore, all registers must be set to their initial values after power-on and $\overline{\text{STBY}}$ input. Figure 11.3 shows the initial setting and resetting procedures of the RTC. Once the register setting are made, the RTC provides an accurate time as long as power is supplied regardless of a $\overline{\text{RES}}$ input.

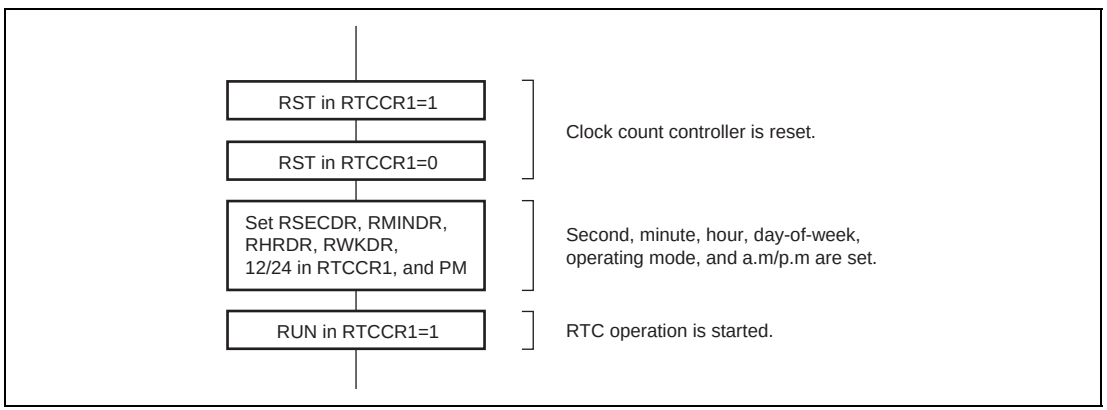



**Figure 11.3 Initial Setting Procedure**

### 11.4.2 Time Data Reading Procedure

When the seconds, minutes, hours, or day-of-week datum is updated while time data is being read, the data obtained may not be correct, and so the time data must be read again. Figure 11.4 shows an example in which correct data is not obtained. In this example, since only RSECDR is read after data update, about 1-minute inconsistency occurs.

To avoid reading in this timing, the following processing must be performed.

1. Check the setting of the BSY bit, and when the BSY bit changes from 1 to 0, read from the second, minute, hour, and day-of-week registers. When about 62.5 ms is passed after the BSY bit is set to 1, the registers are updated, and the BSY bit is cleared to 0.
2. Making use of interrupts, read from the second, minute, hour, and day-of week registers after the IRQ5F flag in ISR is set to 1 and the BSY bit is confirmed to be 0.
3. Read from the second, minute, hour, and day-of week registers twice in a row, and if there is no change in the read data, the read data is used.

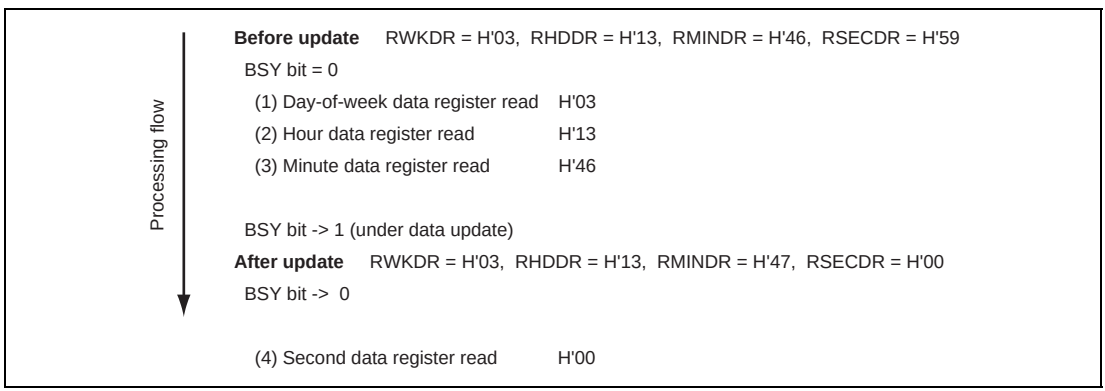



**Figure 11.4 Example: Reading of Inaccurate Time Data**

## 11.5 Interrupt Source

The RTC interrupt sources are listed in table 11.2. There are five kinds of RTC interrupts: week interrupts, day interrupts, hour interrupts, minute interrupts, and second interrupts.

When using an interrupt, initiate the RTC last after other registers (include ISCRH and IER of interrupt controller) are set. Do not set multiple interrupt enable bits in RTCCR2 simultaneously to 1.

When an interrupt request of the RTC occurs, the IRQ5F flag in ISR is set to 1. When clearing the flag, write 0 after reading the flag = 1.

Figure 11.5 shows the initializing setting procedure in using the RTC interrupt and figure 11.6 shows an example of the RTC interrupt handling routine.

**Table 11.2 Interrupt Source**

| Interrupt Name | Interrupt Source | Interrupt Enable Bit |
|---|---|---|
| Overflow interrupt | Occurs when the free running counter is overflown. | FOIE |
| Week periodic interrupt | Occurs every week when the day-of-week date register value becomes 0. | WKIE |
| Day periodic interrupt | Occurs every day when the day-of-week date register is counted. | DYIE |
| Hour periodic interrupt | Occurs every hour when the hour date register is counted. | HRIE |
| Minute periodic interrupt | Occurs every minute when the minute date register is counted. | MNIE |
| Second periodic interrupt | Occurs every second when the second date register is counted. | SEIE |

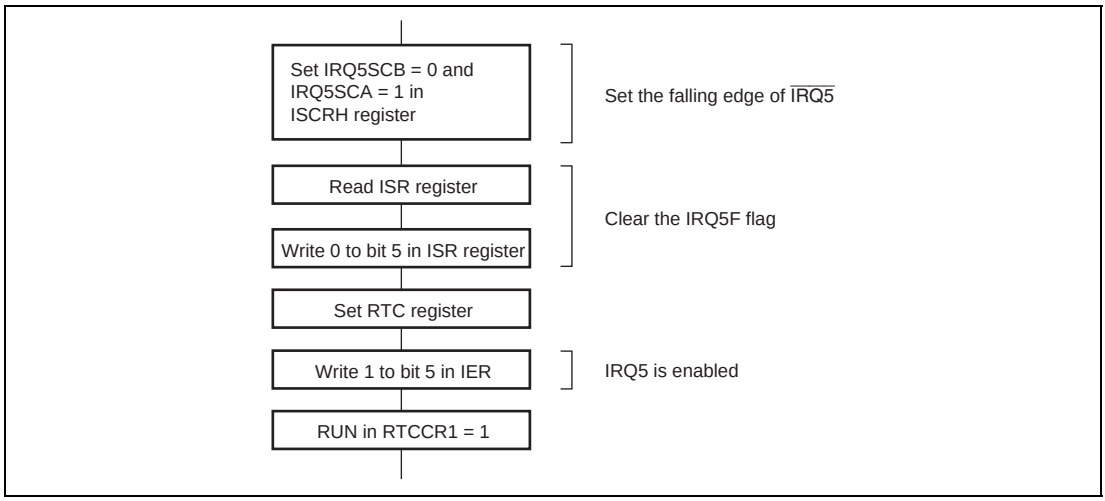



**Figure 11.5 Initializing Procedure in Using RTC Interrupt**

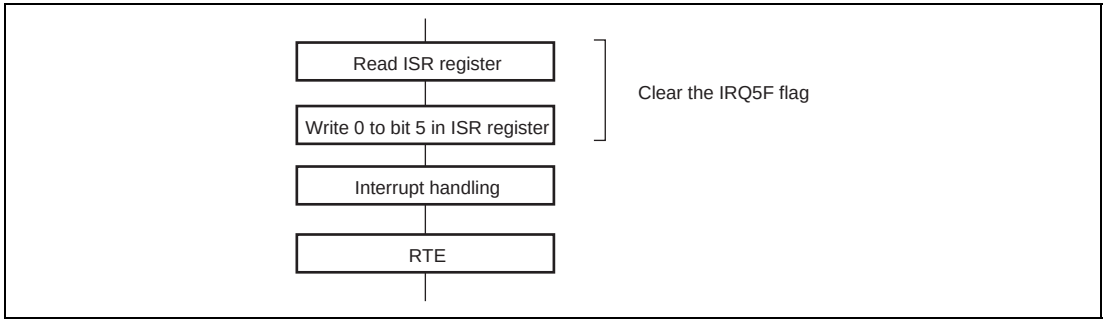



**Figure 11.6 Example of RTC Interrupt Handling Routine**

## 11.6 Operating State in Each Mode

Table 11.3 shows the operating state in each mode when the RTC is set for clock operation and free running timer operation. The clock operation is performed continuously even in low power mode. Therefore, when the clock operation is unnecessary, cancel it by EXMDLSTP.

**Table 11.3 Operating State in Each Mode**

| Function | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hard-ware Standby |
|---|---|---|---|---|---|---|---|---|---|
| Clock operation | Subclock operation | Subclock operation | Subclock operation | Halted (Retained) | Subclock operation | Subclock operation | Subclock operation | Subclock operation | Halted (Reset) |
| Free running timer operation | Operating | Operating | Operating | Halted (Retained) | Halted (Retained) | Halted (Retained) | Halted (Retained) | Halted (Retained) | Halted (Reset) |

## 11.7 Usage Notes

### (1) Notes on Using the Emulator

In the E6000 emulator the RTC module is mounted on an external extended board. Since it must be accessed as an external module, the limitations listed below apply. These limitations do not apply to the E10A or to product chips.

- RTC operation is not supported in the H8S/2218 Group's mode 7 (single-chip mode).
- When using the RTC module in the H8S/2218 Group's mode 6 (on-chip ROM-enabled mode) or the H8S/2212 Group's mode 7 (single-chip mode), A7 to A0 are input pins in the initial status. Therefore, A7 to A0 must be set as output pins by setting PC7DDR to PC0DDR to H'FF before accessing the RTC module.
- The above setting is not necessary when using the RTC module in the H8S/2218 Group's modes 4 and 5 (on-chip ROM-disabled mode) because A7 to A0 are output pins.

### (2) Bus Interface

The bus interface of the module conforms to the bus specifications for external area 7. Consequently, before accessing the RTC module, area 7 must be specified as having an 8-bit bus width and 3-state access using the bus controller register.

### (3) Method for Reading Pin States Using the Port D Register (PORTD)

First access EXMDLSTP or the RTC register (address range: H'FFFF40 to H'FFFF5F). Then, you must perform a dummy read to the external address space (such as H'FFEF00 to H'FF7FF) outside the range H'FFFF40 to H'FFFF5F before reading PORTD.

# Section 12 Serial Communication Interface

This LSI has two independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clocked synchronous serial communication. Asynchronous serial data communication can be carried out using standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or an Asynchronous Communication Interface Adapter (ACIA). A function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface based on ISO/IEC 7816-3 (Identification Card) as an enhanced asynchronous communication function.

## 12.1 Features

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

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

  Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.

  On-chip baud rate generator allows any bit rate to be selected

  External clock can be selected as a transfer clock source (except for in Smart Card interface mode).
- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode 7-bit data)
- Four interrupt sources

  Transmit-end, transmit-data-empty, receive-data-full, and receive error — that can issue requests.

  The transmit-data-empty interrupt and receive data full interrupts can be used to activate the direct memory access controller (DMAC).
- Module stop mode can be set

Asynchronous Mode

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

- Average transfer rate generator (SCI_0):
  921.569 kbps, 720 kbps, 460.784 kbps, or 115.196 kbps can be selected at 16 MHz
  921.053 kbps, 720 kbps, 460.526 kbps, or 115.132 kbps can be selected at 24 MHz
- A transfer rate clock can be input from the TPU (SCI_0)
- Communication between multiple processors is supported

Clocked Synchronous Mode

- Data length: 8 bits
- Receive error detection: Overrun errors detected
- SCI select function (SCI_0): TxD0 = high-impedance and SCK0 = fixed high-level input can selected when $\overline{\text{IRQ7}}$ = 1)
- Serial data communication can be carried out with other chips that have a synchronous communication function

Smart Card Interface

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on detection of a error signal during transmission
- Both direct convention and inverse convention are supported

### 12.1.1 Block Diagram

Figure 12.1 shows the block diagram of the SCI_0. Figure 12.2 shows the block diagram of the SCI_2.

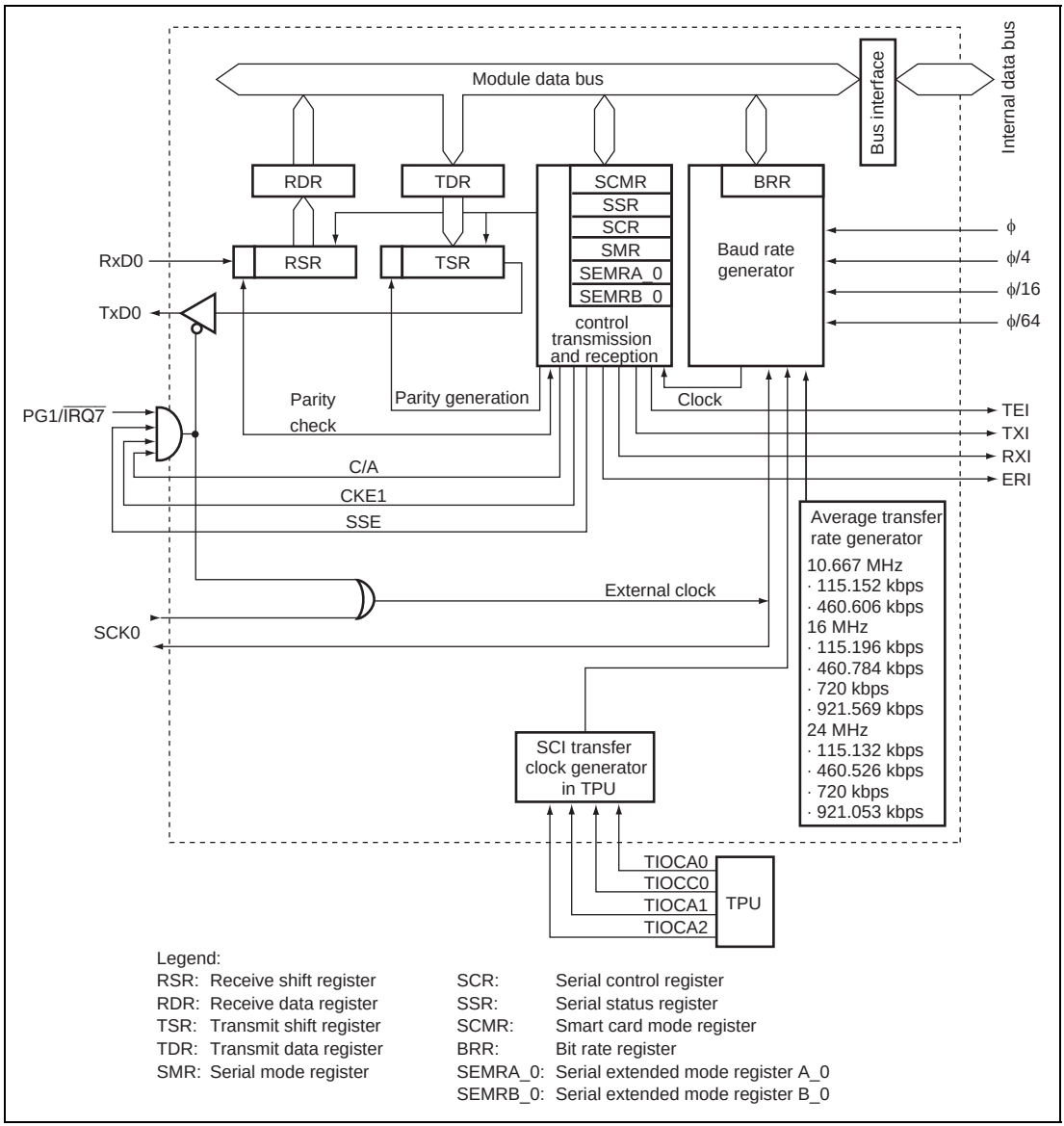



**Figure 12.1 Block Diagram of SCI_0**

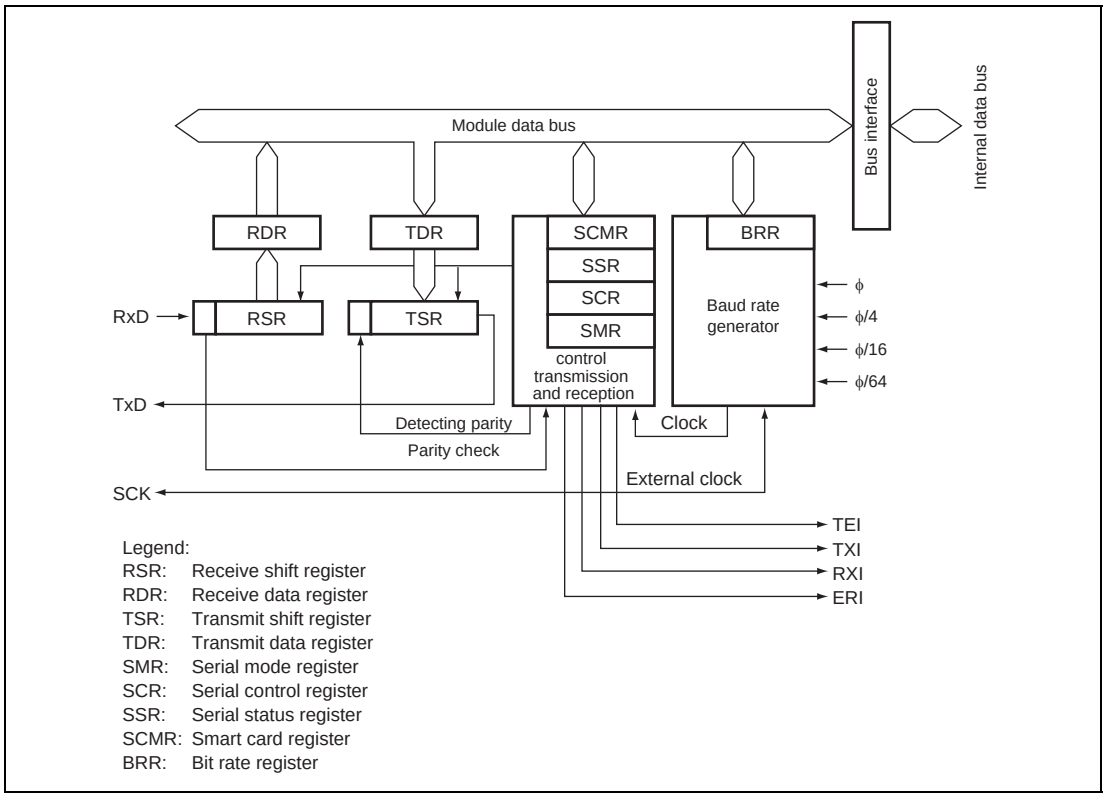



**Figure 12.2 Block Diagram of SCI_2**

## 12.2 Input/Output Pins

Table 12.1 shows the serial pins for each SCI channel.

**Table 12.1 Pin Configuration**

| Channel | Pin Name* | I/O | Function |
|---|---|---|---|
| 0 | SCK0 | I/O | SCI_0 clock input/output |
| | RxD0 | Input | SCI_0 receive data input |
| | TxD0 | Output | SCI_0 transmit data output |
| 2 | SCK2 | I/O | SCI_2 clock input/output |
| | RxD2 | Input | SCI_2 receive data input |
| | TxD2 | Output | SCI_2 transmit data output |

Note: * Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

## 12.3 Register Descriptions

The SCI has the following registers for each channel. Some bits in the serial mode register (SMR), serial status register (SSR), and serial control register (SCR) have different functions in different modes—normal serial communication interface mode and smart card interface mode; therefore, the bits are described separately for each mode in the corresponding register sections.

- Receive shift register (RSR)
- Receive data register (RDR)
- Transmit data register (TDR)
- Transmit shift register (TSR)
- Serial mode register (SMR)
- Serial control register (SCR)
- Serial status register (SSR)
- Smart card mode register (SCMR)
- Serial extended mode register A_0 (SEMRA_0) (only for channel 0)
- Serial extended mode register B_0 (SEMRB_0) (only for channel 0)
- Bit rate register (BRR)

### 12.3.1 Receive Shift Register (RSR)

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

### 12.3.2 Receive Data Register (RDR)

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

### 12.3.3 Transmit Data Register (TDR)

TDR is an 8-bit register that stores data for transmission. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structure of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR during serial transmission, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read from or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR only once after confirming that the TDRE bit in SSR is set to 1. TDR is initialized to H'FF by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, or module stop mode.

### 12.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first transfers transmit data from TDR to TSR, then sends the data to the TxD pin. TSR cannot be directly accessed by the CPU.

### 12.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock source. Some bits in SMR have different functions in normal mode and smart card interface mode.

- Normal Serial Communication Interface Mode (When SMIF in SCMR is 0)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | C/$\overline{A}$ | 0 | R/W | Communication Mode<br>0: Asynchronous mode<br>1: Clocked synchronous mode |
| 6 | CHR | 0 | R/W | Character Length (enabled only in asynchronous mode)<br>0: Selects 8 bits as the data length<br>1: Selects 7 bits as the data length. LSB-first is fixed and the MSB of TDR is not transmitted in transmission<br>In clocked synchronous mode, a fixed data length of 8 bits is used. |
| 5 | PE | 0 | R/W | Parity Enable (enabled only in asynchronous mode)<br>When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. For a multiprocessor format, parity bit addition and checking are not performed regardless of the PE bit setting. |
| 4 | O/$\overline{E}$ | 0 | R/W | Parity Mode (enabled only when the PE bit is 1 in asynchronous mode)<br>0: Selects even parity<br>1: Selects odd parity |
| 3 | STOP | 0 | R/W | Stop Bit Length (enabled only in asynchronous mode)<br>Selects the stop bit length in transmission.<br>0: 1 stop bit<br>1: 2 stop bits<br>In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit character. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 2 | MP | 0 | R/W | Multiprocessor Mode (enabled only in asynchronous mode)<br>When this bit is set to 1, the multiprocessor communication function is enabled. The PE bit and O/$\overline{E}$ bit settings are invalid in multiprocessor mode. For details, see section 12.5, Multi Processor Communication Function. |
| 1 | CKS1 | 0 | R/W | Clock Select 0 and 1:<br>These bits select the clock source for the baud rate generator.<br>00: $\phi$ clock (n = 0)<br>01: $\phi$/4 clock (n = 1)<br>10: $\phi$/16 clock (n = 2)<br>11: $\phi$/64 clock (n = 3)<br>For the relationship between the bit rate register setting and the baud rate, see section 12.3.11, Bit Rate Register (BRR). n is the decimal representation of the value of n in BRR (see section 12.3.11, Bit Rate Register (BRR)). |
| 0 | CKS0 | 0 | R/W | |

- Smart Card Interface Mode (When SMIF in SCMR is 1)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | GM | 0 | R/W | GSM Mode<br>Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 etu from the start and the clock output control function is appended. For details, see section 12.7.9, Clock Output Control.<br>0: Normal smart card interface mode operation (initial value)<br>(1) The TEND flag is generated 12.5 etu (11.5 etu in the block transfer mode) after the beginning of the start bit.<br>(2) Clock output on/off control only.<br>1: GSM mode operation in smart card interface mode<br>(1) The TEND flag is generated 11.0 etu after the beginning of the start bit.<br>(2) In addition to clock output on/off control, high/how fixed control is supported (set using SCR). |
| 6 | BLK | 0 | R/W | Setting this bit to 1 allows block transfer mode operation. For details, see section 12.7.4, Block Transfer Mode.<br>0: Normal smart card interface mode operation (initial value)<br>(1) Error signal transmission, detection, and automatic data retransmission are performed.<br>(2) The TXI interrupt is generated by the TEND flag.<br>(3) The TEND flag is set 12.5 etu (11.0 etu in the GSM mode) after transmission starts.<br>1: Operation in block transfer mode<br>(1) Error signal transmission, detection, and automatic data retransmission are not performed.<br>(2) The TXI interrupt is generated by the TDRE flag.<br>(3) The TEND flag is set 11.5 etu (11.0 etu in the GSM mode) after transmission starts. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 5 | PE | 0 | R/W | Parity Enable<br>When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. Set this bit to 1 in smart card interface mode. |
| 4 | O/$\overline{E}$ | 0 | R/W | Parity Mode (valid only when the PE bit is 1)<br>0: Selects even parity<br>1: Selects odd parity<br>For details on the usage of this bit in smart card interface mode, see section 12.7.2, Data Format (Except for Block Transfer Mode). |
| 3 | BCP1 | 0 | R/W | Basic Clock Pulse 1 and 0 |
| 2 | BCP0 | 0 | R/W | These bits select the number of basic clock cycles in a 1-bit data transfer time in smart card interface mode.<br>00: 32 clock cycles (S = 32)<br>01: 64 clock cycles (S = 64)<br>10: 372 clock cycles (S = 372)<br>11: 256 clock cycles (S = 256)<br>For details, see section 12.7.5, Receive Data Sampling Timing and Reception Margin. S is described in section 12.3.11, Bit Rate Register (BRR). |
| 1 | CKS1 | 0 | R/W | Clock Select 1 and 0 |
| 0 | CKS0 | 0 | R/W | These bits select the clock source for the baud rate generator.<br>00: $\phi$ clock (n = 0)<br>01: $\phi$/4 clock (n = 1)<br>10: $\phi$/16 clock (n = 2)<br>11: $\phi$/64 clock (n = 3)<br>For the relation between the bit rate register setting and the baud rate, see section 12.3.11, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 12.3.11, Bit Rate Register (BRR)). |

### 12.3.6 Serial Control Register (SCR)

SCR is a register that enables or disables SCI transfer operations and interrupt requests, and is also used to selection of the transfer clock source. For details on interrupt requests, refer to section 12.9, Interrupts. Some bits in SCR have different functions in normal mode and smart card interface mode.

- Normal Serial Communication Interface Mode (When SMIF in SCMR is 0)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TIE | 0 | R/W | Transmit Interrupt Enable<br>When this bit is set to 1, the TXI interrupt request is enabled.<br>TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or clearing the TIE bit to 0. |
| 6 | RIE | 0 | R/W | Receive Interrupt Enable<br>When this bit is set to 1, RXI and ERI interrupt requests are enabled.<br>RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or clearing the RIE bit to 0. |
| 5 | TE | 0 | R/W | Transmit Enable<br>When this bit s set to 1, transmission is enabled.<br>In this state, serial transmission is started when transmit data is written to TDR and the TDRE flag in SSR is cleared to 0. SMR setting must be performed to decide the transfer format before setting the TE bit to 1. The TDRE flag in SSR is fixed at 1 if transmission is disabled by clearing this bit to 0. |
| 4 | RE | 0 | R/W | Receive Enable<br>When this bit is set to 1, reception is enabled.<br>Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in clocked synchronous mode. SMR setting must be performed to decide the transfer format before setting the RE bit to 1.<br>Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | MPIE | 0 | R/W | Multiprocessor Interrupt Enable (enabled only when the MP bit in SMR is 1 in asynchronous mode)<br>When this bit is set to 1, receive data in which the multiprocessor bit is 0 is skipped, and setting of the RDRF, FER, and ORER status flags in SSR is prohibited. On receiving data in which the multiprocessor bit is 1, this bit is automatically cleared and normal reception is resumed. For details, refer to section 12.5, Multiprocessor Communication Function.<br>When receive data including MPB = 0 is received, receive data transfer from RSR to RDR, receive error detection, and setting of the RDRF, FER, and ORER flags in SSR, is not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR are set to 1) and FER and ORER flag setting is enabled. |
| 2 | TEIE | 0 | R/W | Transmit End Interrupt Enable<br>This bit is set to 1, TEI interrupt request is enabled. TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or clearing the TEIE bit to 0. |
| 1 | CKE1 | 0 | R/W | Clock Enable 0 and 1 |
| 0 | CKE0 | 0 | R/W | Selects the clock source and SCK pin function.<br>Asynchronous mode<br>00: Internal baud rate generator<br>SCK pin functions as I/O port<br>01: Internal baud rate generator<br>Outputs a clock of the same frequency as the bit rate from the SCK pin.<br>1×: External clock<br>Inputs a clock with a frequency 16 times the bit rate from the SCK pin.<br>Clocked synchronous mode<br>0×: Internal clock (SCK pin functions as clock output)<br>1×: External clock (SCK pin functions as clock input) |

Legend:

×: Don't care

- Smart Card Interface Mode (When SMIF in SCMR is 1)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TIE | 0 | R/W | Transmit Interrupt Enable<br>When this bit is set to 1, TXI interrupt request is enabled.<br>TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0, or clearing the TIE bit to 0. |
| 6 | RIE | 0 | R/W | Receive Interrupt Enable<br>When this bit is set to 1, RXI and ERI interrupt requests are enabled.<br>RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF, FER, PER, or ORER flag in SSR, then clearing the flag to 0, or clearing the RIE bit to 0. |
| 5 | TE | 0 | R/W | Transmit Enable<br>When this bit s set to 1, transmission is enabled.<br>In this state, serial transmission is started when transmit data is written to TDR and the TDRE flag in SSR is cleared to 0.<br>SMR setting must be performed to decide the transfer format before setting the TE bit to 1. When this bit is cleared to 0, the transmission operation is disabled, and the TDRE flag is fixed at 1. |
| 4 | RE | 0 | R/W | Receive Enable<br>When this bit is set to 1, reception is enabled.<br>Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in clocked synchronous mode.<br>SMR setting must be performed to decide the reception format before setting the RE bit to 1.<br>Clearing the RE bit to 0 does not affect the RDRF, FER,PER, and ORER flags, which retain their states. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | MPIE | 0 | R/W | Multiprocessor Interrupt Enable (enabled only when the MP bit in SMR is 1 in asynchronous mode)<br>Write 0 to this bit in Smart Card interface mode.<br>When receive data including MPB = 0 is received, receive data transfer from RSR to RDR, receive error detection, and setting of the RERF, FER, and ORER flags in SSR, are not performed.<br>When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR are set to 1) and FER and ORER flag setting are enabled. |
| 2 | TEIE | 0 | R/W | Transmit End Interrupt Enable<br>Write 0 to this bit in Smart Card interface mode.<br>TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or clearing the TEIE bit to 0. |
| 1 | CKE1 | 0 | R/W | Clock Enable 0 and 1 |
| 0 | CKE0 | 0 | | Enables or disables clock output from the SCK pin. The clock output can be dynamically switched in GSM mode. For details, refer to section 12.7.9, Clock Output Control.<br>When the GM bit in SMR is 0:<br>00: Output disabled (SCK pin can be used as an I/O port pin)<br>01: Clock output<br>1×: Reserved<br>When the GM bit in SMR is 1:<br>00: Output fixed low<br>01: Clock output<br>10: Output fixed high<br>11: Clock output |

Legend:

×: Don’t care

### 12.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER; they can only be cleared. Some bits in SSR have different functions in normal mode and smart card interface mode.

- Normal Serial Communication Interface Mode (When SMIF in SCMR is 0)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TDRE | 1 | R/(W)*[1] | Transmit Data Register Empty<br>Displays whether TDR contains transmit data.<br>[Setting conditions]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR<br>[Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1*[2]<br>• When the DMAC is activated by a TXI interrupt request and writes data to TDR |
| 6 | RDRF | 0 | R/(W)*[1] | Receive Data Register Full<br>Indicates that the received data is stored in RDR.<br>[Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR<br>[Clearing conditions]<br>• When 0 is written to RDRF after reading RDRF = 1*[2]<br>• When the DMAC is activated by an RXI interrupt and transferred data from RDR<br>RDR and the RDRF flag are not affected and retain their previous values when the RE bit in SCR is cleared to 0.<br>The RDRF flag is not affected and retains their previous values when the RE bit in SCR is cleared to 0. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 5 | ORER | 0 | R/(W)*[1] | Overrun Error<br>[Setting condition]<br>• When the next serial reception is completed while RDRF = 1<br>The receive data prior to the overrun error is retained in RDR, and the data received subsequently is lost. Also, subsequent serial reception cannot be continued while the ORER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.<br>[Clearing condition]<br>• When 0 is written to ORER after reading ORER = 1*[2]<br>The ORER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0. |
| 4 | FER | 0 | R/(W)*[1] | Framing Error<br>[Setting condition]<br>• When the stop bit is 0<br>In 2-stop-bit mode, only the first stop bit is checked for a value of 0; the second stop bits not checked. If a framing error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the FER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.<br>[Clearing condition]<br>• When 0 is written to FER after reading FER = 1*[2]<br>The FER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | PER | 0 | R/(W)*[1] | Parity Error<br>[Setting condition]<br>• When a parity error is detected during reception<br>If a parity error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.<br>[Clearing condition]<br>• When 0 is written to PER after reading PER = 1*[2]<br>The PER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0. |
| 2 | TEND | 1 | R | Transmit End<br>[Setting conditions]<br>• When the TE bit in SCR is 0<br>• When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character<br>[Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC is activated by a TXI interrupt and writes data to TDR |
| 1 | MPB | 0 | R | Multiprocessor Bit<br>MPB stores the multiprocessor bit in the receive data. When the RE bit in SCR is cleared to 0 its previous state is retained. This bit retains its previous state when the RE bit in SCR is cleared to 0. |
| 0 | MPBT | 0 | R/W | Multiprocessor Bit Transfer<br>MPBT stores the multiprocessor bit to be added to the transmit data. |

Note: 1. The write value should always be 0 to clear the flag.
2. To clear the flag by the CPU on the HD6432210S, reread the flag after writing 0 to it.

- Smart Card Interface Mode (When SMIF in SCMR is 1)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TDRE | 1 | R/(W)*[1] | Transmit Data Register Empty<br>Indicates whether TDR contains transmit data.<br>[Setting conditions]<br>• When the TE bit in SCR is 0<br>• When data is transferred from TDR to TSR and data can be written to TDR<br>[Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1*[2]<br>• When the DMAC is activated by a TXI interrupt request and writes data to TDR |
| 6 | RDRF | 0 | R/(W)*[1] | Receive Data Register Full<br>Indicates that the received data is stored in RDR.<br>[Setting condition]<br>When serial reception ends normally and receive data is transferred from RSR to RDR<br>[Clearing conditions]<br>• When 0 is written to RDRF after reading RDRF = 1*[2]<br>• When the DMAC is activated by an RXI interrupt and transferred data from RDR<br>The RDRF flag is not affected and retains their previous values when the RE bit in SCR is cleared to 0.<br>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 | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 5 | ORER | 0 | R/(W)*[1] | Overrun Error<br>Indicates that an overrun error occurred during reception, causing abnormal termination.<br>[Setting condition]<br>• When the next serial reception is completed while RDRF = 1<br>The receive data prior to the overrun error is retained in RDR, and the data received subsequently is lost. Also, subsequent serial cannot be continued while the ORER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.<br>[Clearing condition]<br>• When 0 is written to ORER after reading ORER = 1*[2]<br>The ORER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0. |
| 4 | ERS | 0 | R/(W)*[1] | Error Signal Status<br>Indicates that the status of an error, signal 1 returned from the reception side at reception<br>[Setting condition]<br>• When the low level of the error signal is sampled<br>[Clearing condition]<br>• When 0 is written to ERS after reading ERS = 1*[2]<br>The ERS flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | PER | 0 | R/(W)*[1] | Parity Error<br>Indicates that a parity error occurred during reception using parity addition in asynchronous mode, causing abnormal termination.<br>[Setting condition]<br>• When a parity error is detected during reception<br>If a parity error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. In clocked synchronous mode, serial transmission cannot be continued, either.<br>[Clearing condition]<br>• When 0 is written to PER after reading PER = 1*[2]<br>The PER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0. |
| 2 | TEND | 1 | R | Transmit End<br>This bit is set to 1 when no error signal has been sent back from the receiving end and the next transmit data is ready to be transferred to TDR.<br>[Setting conditions]<br>• When the TE bit in SCR is 0 and the ERS bit is also 0<br>• When the ERS bit is 0 and the TDRE bit is 1 after the specified interval following transmission of 1-byte data.<br>The timing of bit setting differs according to the register setting as follows:<br>When GM = 0 and BLK = 0, 12.5 etu after transmission starts<br>When GM = 0 and BLK = 1, 11.5 etu after transmission starts<br>When GM = 1 and BLK = 0, 11.0 etu after transmission starts<br>When GM = 1 and BLK = 1, 11.0 etu after transmission starts<br>[Clearing conditions]<br>• When 0 is written to TDRE after reading TDRE = 1<br>• When the DMAC is activated by a TXI interrupt and transfers transmission data to TDR |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 1 | MPB | 0 | R | Multiprocessor Bit<br>This bit is not used in Smart Card interface mode. |
| 0 | MPBT | 0 | R/W | Multiprocessor Bit Transfer<br>Write 0 to this bit in Smart Card interface mode. |

Note: 1. The write value should always be 0 to clear the flag.
2. To clear the flag by the CPU on the HD6432210S, reread the flag after writing 0 to it.

### 12.3.8 Smart Card Mode Register (SCMR)

SCMR is a register that selects the transfer format. In this LSI, Smart Card interface mode cannot be specified.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 1 | — | Reserved<br>These bits are always read as 1. |
| 3 | DIR | 0 | R/W | Data Transfer Direction<br>Selects the serial/parallel conversion format.<br>0: LSB-first in transfer<br>1: MSB-first in transfer<br>The bit setting is valid only when the transfer data format is 8 bits. |
| 2 | INV | 0 | R/W | Data Invert<br>Specifies inversion of the data logic level. The SINV bit does not affect the logic level of the parity bit. To invert the parity bit, invert the O/$\overline{E}$ bit in SMR.<br>0: TDR contents are transmitted as they are. Receive data is stored as it is in RDR<br>1: TDR contents are inverted before being transmitted. Receive data is stored in inverted form in RDR |
| 1 | — | 1 | — | Reserved<br>This bit is always read as 1. |
| 0 | SMIF | 0 | R/W | Smart Card Interface Mode Select<br>When this bit is set to 1, smart card interface mode is selected.<br>0: Normal asynchronous or clocked synchronous mode<br>1: Smart card interface mode |

### 12.3.9 Serial Extended Mode Register A_0 (SEMRA_0)

SEMRA_0 extends the functions of SCI_0. SEMR0 enables selection of the SCI_0 select function in synchronous mode, base clock setting in asynchronous mode, and also clock source selection and automatic transfer rate setting. Figure 12.3 shows an example of the internal base clock when an average transfer rate is selected and figure 12.4 shows as example of the setting when the TPU clock input is selected.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | SSE | 0 | R/W | SCI_0 Select Enable<br>Allows selection of the SCI0 select function when an external clock is input in synchronous mode.<br>The SSE setting is valid when external clock input is used (CKE1 = 1 in SCR) in synchronous mode (C/$\overline{A}$ = 1 in SMR).<br>0: SCI_0 select function disabled<br>1: SCI_0 select function enabled<br>When the SCI_0 select function is enabled, if 1 is input to the PG1/$\overline{IRQ7}$ pin, TxD0 output goes to the high-impedance state, SCK0 input is fixed high. |
| 6 | TCS2 | 0 | R/W | TPU Clock Select |
| 5 | TCS1 | 0 | R/W | When the TPU clock is input (ACS3 to ACS0 = B'0100) as the clock source in asynchronous mode, serial transfer clock is generated depending on the combination of the TPU clock. |
| 4 | TCS0 | 0 | R/W | |

| | Base Clock | Clock Enable | TCLKA | TCLKB | TCLKC |
|---|---|---|---|---|---|
| 000 | TIOCA1 | TIOCA2 | Base clock written in the left column | Pin input | Pin input |
| 001 | TIOCA0 \| TIOCC0 | TIOCA1 | Pin input | Base clock written in the left column | Pin input |
| 010 | TIOCA0 | TIOCA1 & TIOCA2 | Pin input | Base clock written in the left column | Pin input |
| 011 | TIOCA0 \| TIOCC0 | TIOCA1 & TIOCA2 | Pin input | Base clock written in the left column | Pin input |
| 1×× | Reserved (Setting prohibited) | | | | |

Legend:

&: AND (logical multiplication)

I : OR (logical addition)

Note: The functions of bits 6 to 4 are not supported by the E6000 emulator. Figure 12.4 shows the setting examples.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | ABCS | 0 | R/W | Asynchronous Base Clock Select<br>Selects the 1-bit-interval base clock in asynchronous mode. The ABCS setting is valid in asynchronous mode (C/$\overline{A}$ = 0 in SMR).<br>0: SCI_0 operates on base clock with frequency of 16 times transfer rate<br>1: SCI_0 operates on base clock with frequency of 8 times transfer rate |
| 2 | ACS2 | 0 | R/W | Asynchronous Clock Source Select 2 to 0<br>These bits select the clock source in asynchronous mode depending on the combination with the bit 7 (ACS3) in SEMRB_0 (serial extended mode register B_0). When an average transfer rate is selected, the base clock is set automatically regardless of the ABCS value. Note that average transfer rates support only 10.667 MHz, 16 MHz, and 24 MHz, and not support other operating frequencies.<br>Set ACS3 to ACS0 when inputting the external clock (the CKE1 bit in the SCR register is 1) in asynchronous mode (the C/$\overline{A}$ bit in the SMR register is 0). Figures 12.3 and 12.4 show the setting examples.<br>ACS 3210<br>0000: External clock input<br>0001: 115.152 kbps average transfer rate (for $\phi$ = 10.667 MHz only) is selected (SCI_0 operates on base clock with frequency of 16 times transfer rate)<br>0010: 460.606 kbps average transfer rate (for $\phi$ = 10.667 MHz only) is selected (SCI_0 operates on base clock with frequency of eight times transfer rate)<br>0011: 921.569 kbps average transfer rate (for $\phi$ = 16 MHz only) is selected (SCI_0 operates on base clock with frequency of eight times transfer rate)<br>0100: TPU clock input<br>The signal generated by TIOCA0, TIOCC0, TIOCA1, and TIOCA2, which are the compare match outputs for TPU_0 to TPU_2 or PWM outputs, is used as a base clock. Note that $\overline{IRQ0}$ and $\overline{IRQ1}$ cannot be used since TIOCA1 and TIOCA2 are used as outputs. |
| 1 | ACS1 | 0 | R/W | |
| 0 | ACS0 | 0 | R/W | |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 2 | ACS2 | 0 | R/W | 0101: 115.196 kbps average transfer rate (for ϕ = 16 MHz only) is selected (SCI_0 operates on base clock with frequency of 16 times transfer rate) |
| 1 | ACS1 | 0 | R/W | |
| 0 | ACS0 | 0 | R/W | 0110: 460.784 kbps average transfer rate (for ϕ = 16 MHz only) is selected (SCI_0 operates on base clock with frequency of eight times transfer rate) |
| | | | | 0111: 720 kbps average transfer rate (for ϕ = 16 MHz only) is selected (SCI_0 operates on base clock with frequency of eight times transfer rate) |
| | | | | 1000: 115.132 kbps average transfer rate (for ϕ = 24 MHz only) is selected* (SCI_0 operates on base clock with frequency of 16 times transfer rate) |
| | | | | 1001: 460.526 kbps average transfer rate (for ϕ = 24 MHz only) is selected* (SCI_0 operates on base clock with frequency of 16 times transfer rate) |
| | | | | 1010: 720 kbps average transfer rate (for ϕ = 24 MHz only) is selected* (SCI_0 operates on base clock with frequency of eight times transfer rate) |
| | | | | 1011: 921.053 kbps average transfer rate (for ϕ = 24 MHz only) is selected* (SCI_0 operates on base clock with frequency of eight times transfer rate) |
| | | | | 11××: Reserved (Setting prohibited) |

Note: * The average transfer rate select functions for 24 MHz only (ACS3 to ACS0 = B'10XX) are not supported by the E6000 emulator.

### 12.3.10 Serial Extended Mode Register B_0 (SEMRB_0)

SEMRB_0 enables clock source selection with the combination of SEMRA_0, automatic transfer rate setting, and control of port 1 pins (P16, P14, P12, and P10) at the transfer clock generation by TPU.

Note: SEMRB_0 is not supported by the E6000 emulator.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | ACS3 | 0 | R/W | Asynchronous Clock Source Select<br>Selects the clock source in asynchronous mode depending on the combination with the ACS2 to ACS0 (bits 2 to 0 in SEMRA_0). For details, see section 12.3.9, Serial Extended Mode Register A_0 (SEMRA_0). |
| 6 to 4 | — | Undefined | — | Reserved<br>The write value should always be 0. |
| 3 | TIOCA2E | 1 | R/W | TIOCA2 Output Enable<br>Controls the TIOCA2 output on the P16 pin.<br>When the TIOCA2 in TPU is output to generate the transfer clock, P16 is used as other function pin by setting this bit to 0.<br>0: Disables output of TIOCA2 in TPU<br>1: Enables output of TIOCA2 in TPU |
| 2 | TIOCA1E | 1 | R/W | TIOCA1 Output Enable<br>Controls the TIOCA1 output on the P14 pin.<br>When the TIOCA1 in TPU is output to generate the transfer clock, P14 is used as other function pin by setting this bit to 0.<br>0: Disables output of TIOCA1 in TPU<br>1: Enables output TIOCA1 in TPU |
| 1 | TIOCC0E | 1 | R/W | TIOCC0 Output Enable<br>Controls the TIOCC0 output on the P12 pin.<br>When the TIOCC0 in TPU is output to generate the transfer clock, P12 is used as other function pin by setting this bit to 0.<br>0: Disables output of TIOCC0 in TPU<br>1: Enables output of TIOCC0 in TPU |
| 0 | TIOCA0E | 1 | R/W | TIOCA0 Output Enable<br>Controls the TIOCA0 output on the P10 pin.<br>When the TIOCA0 in TPU is output to generate the transfer clock, P10 is used as other function pin by setting this bit to 0.<br>0: Disables output of TIOCA0 in TPU<br>1: Enables output of TIOCA0 in TPU |

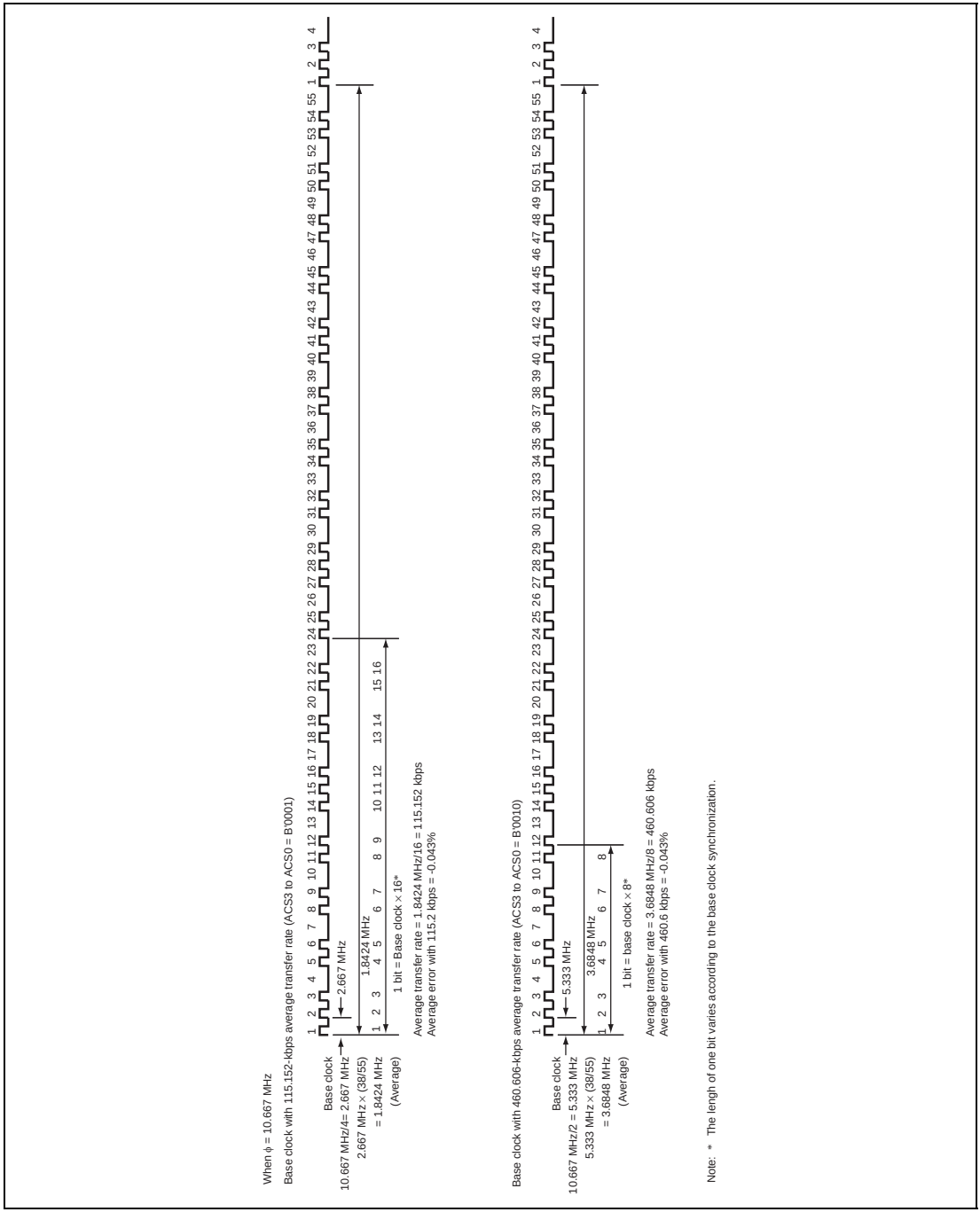



**Figure 12.3 Examples of Base Clock when Average Transfer Rate Is Selected (1)**

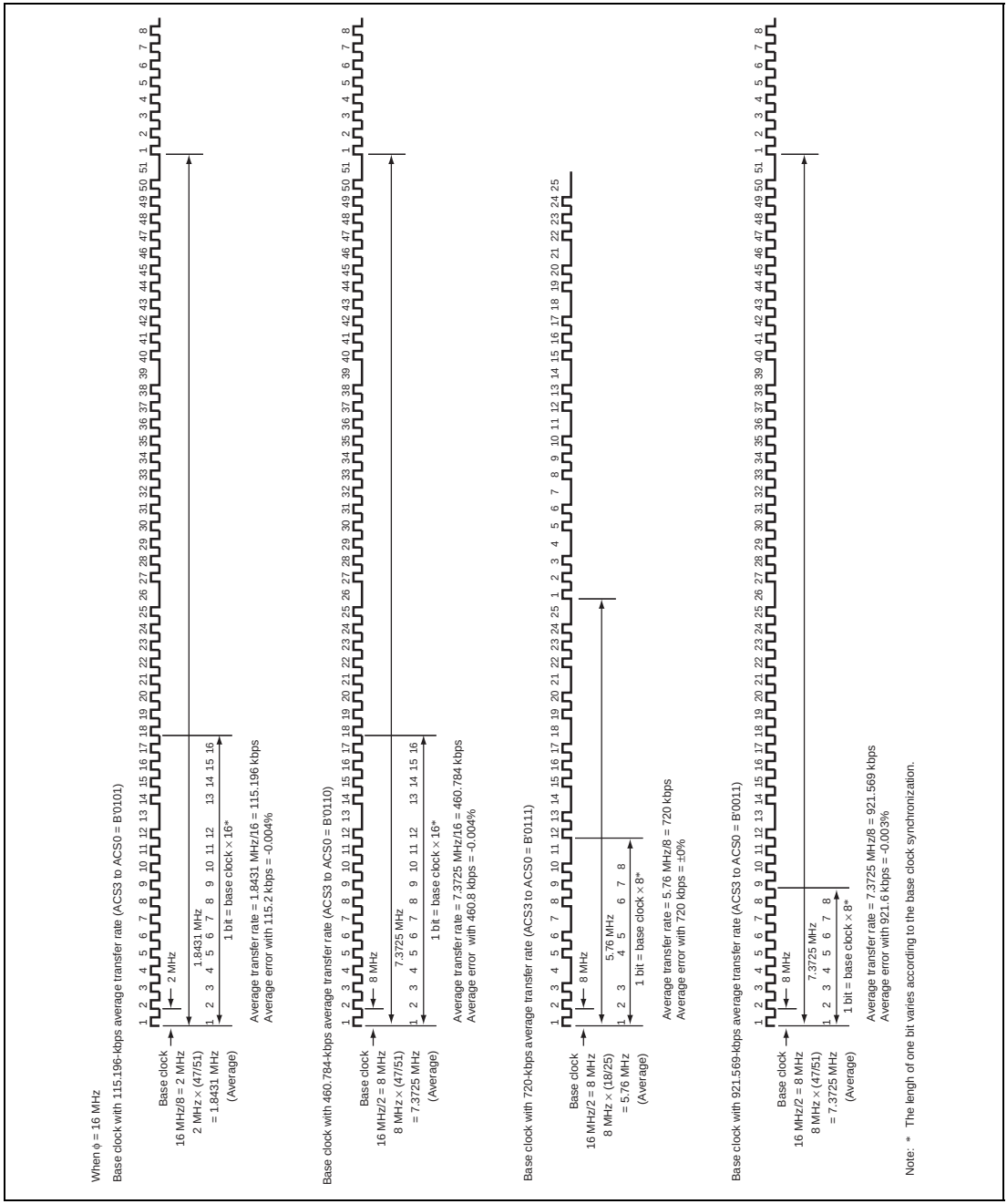



**Figure 12.3 Examples of Base Clock when Average Transfer Rate Is Selected (2)**

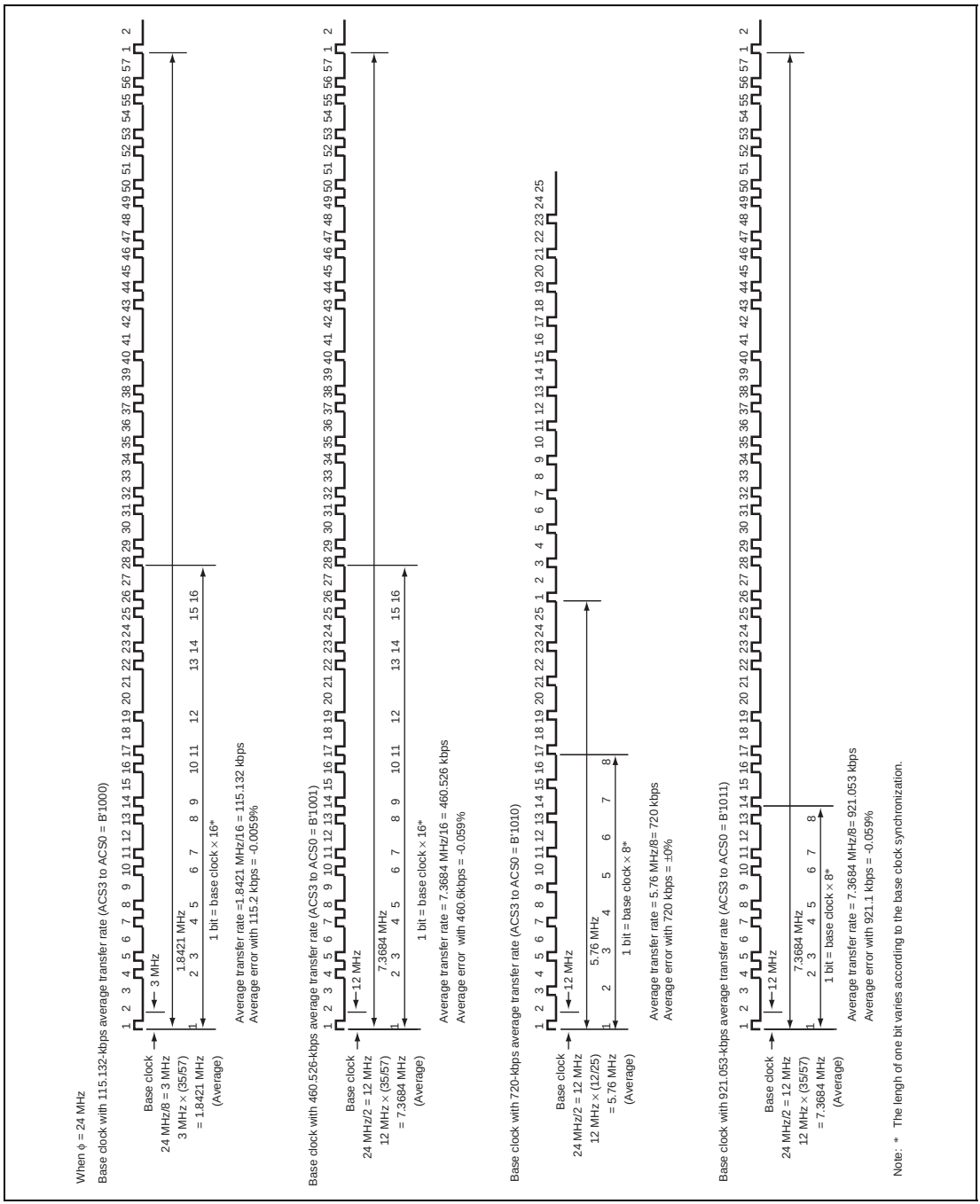



**Figure 12.3 Examples of Base Clock when Average Transfer Rate Is Selected (3)**

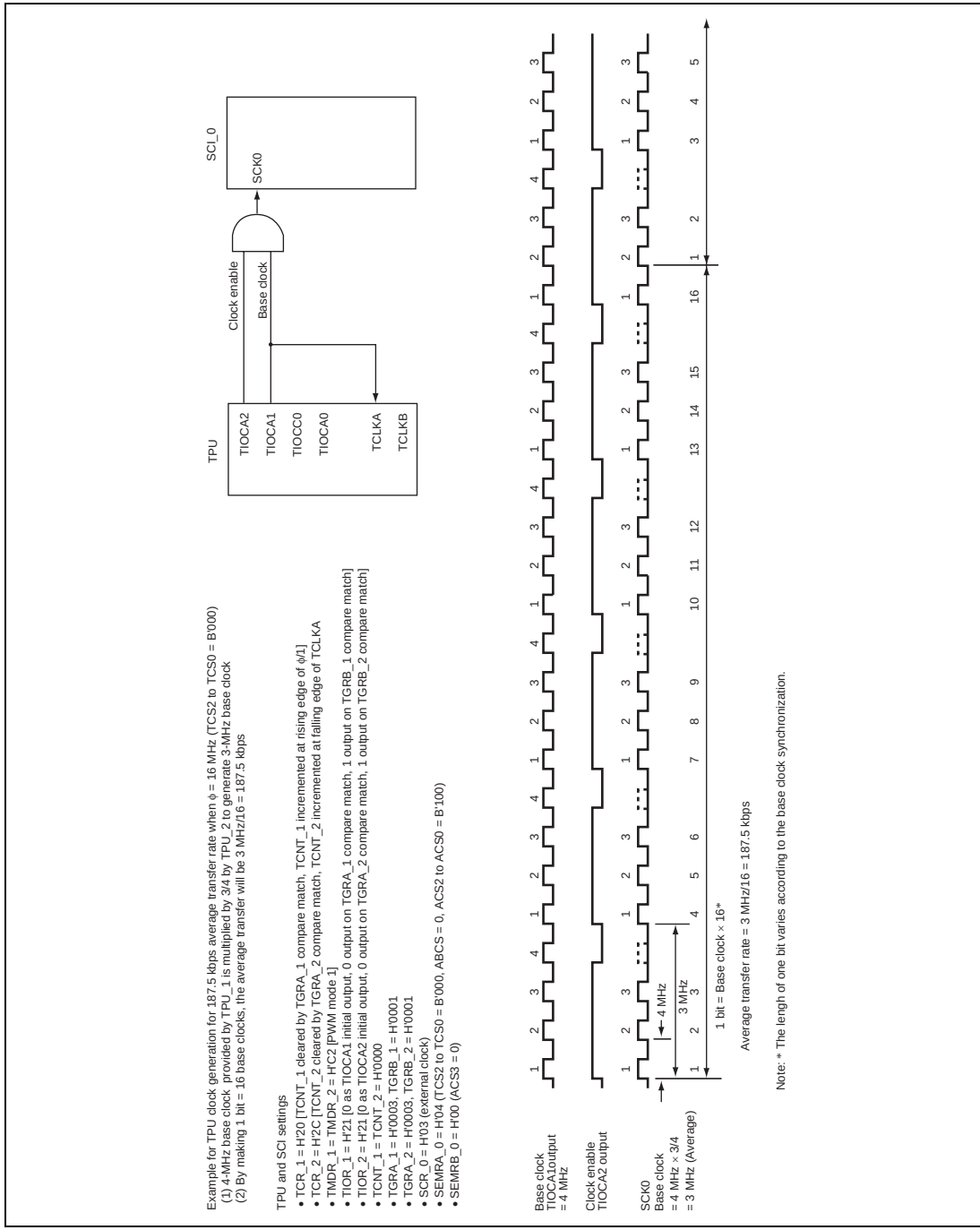



**Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (1)**

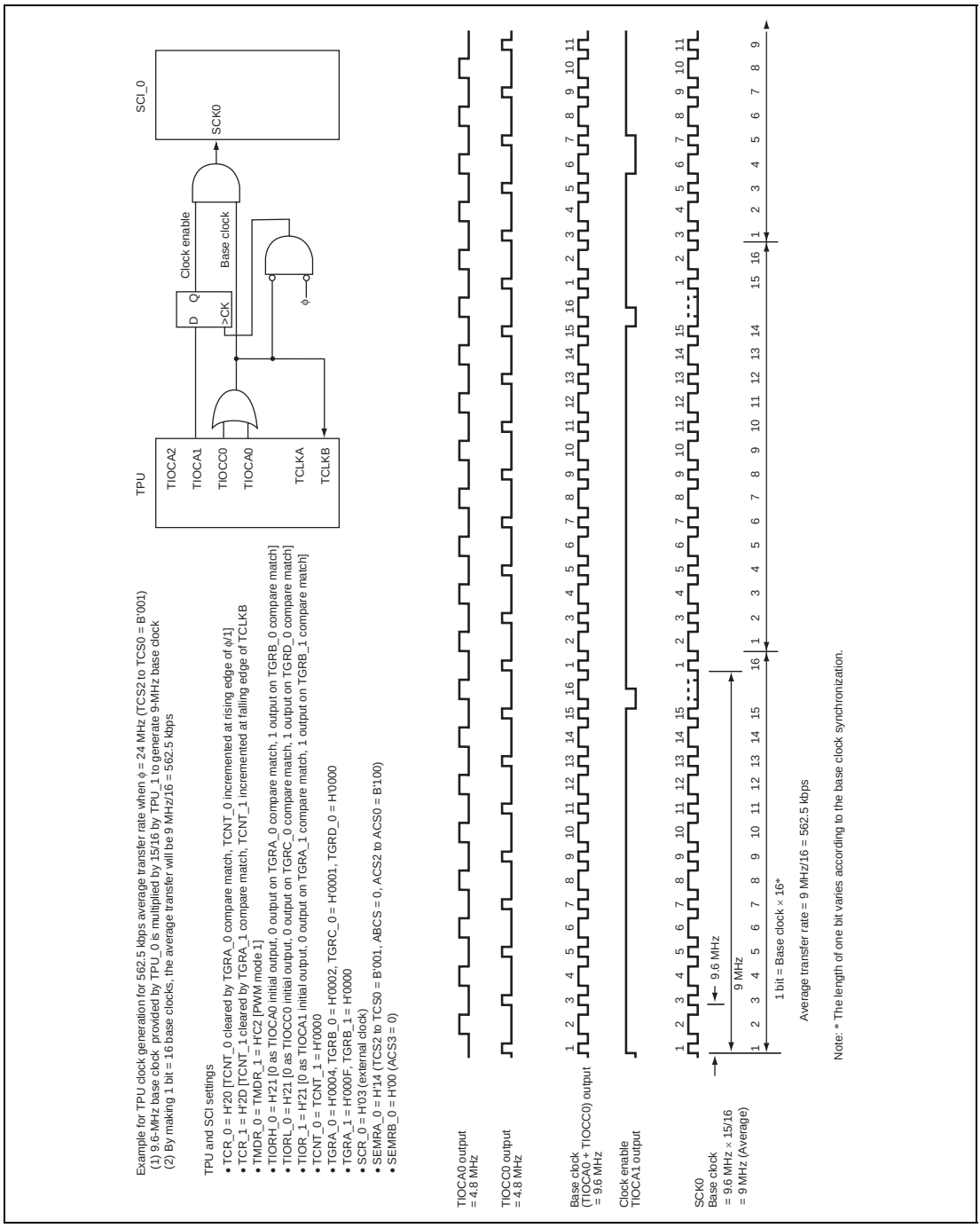



**Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (2)**

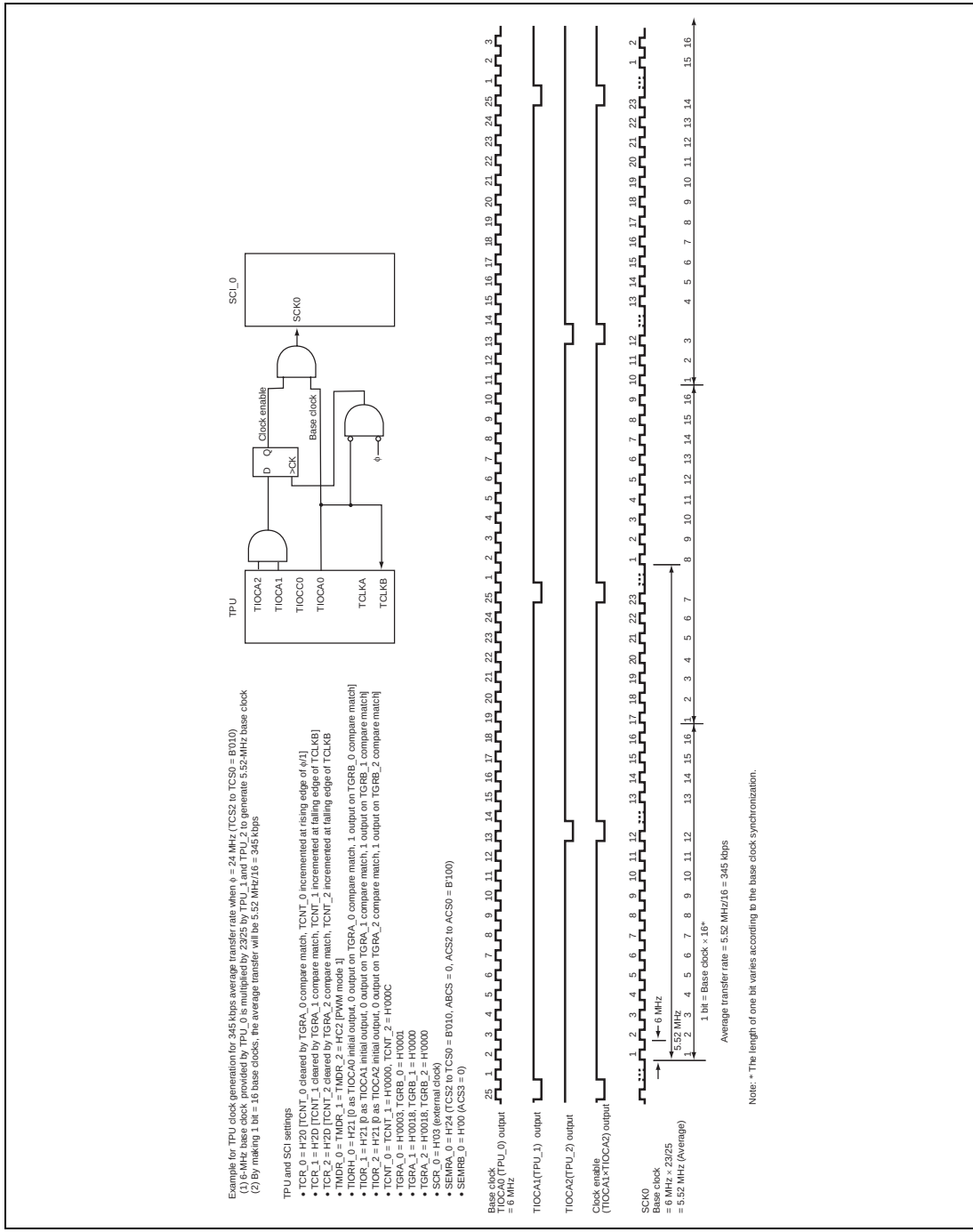



**Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (3)**

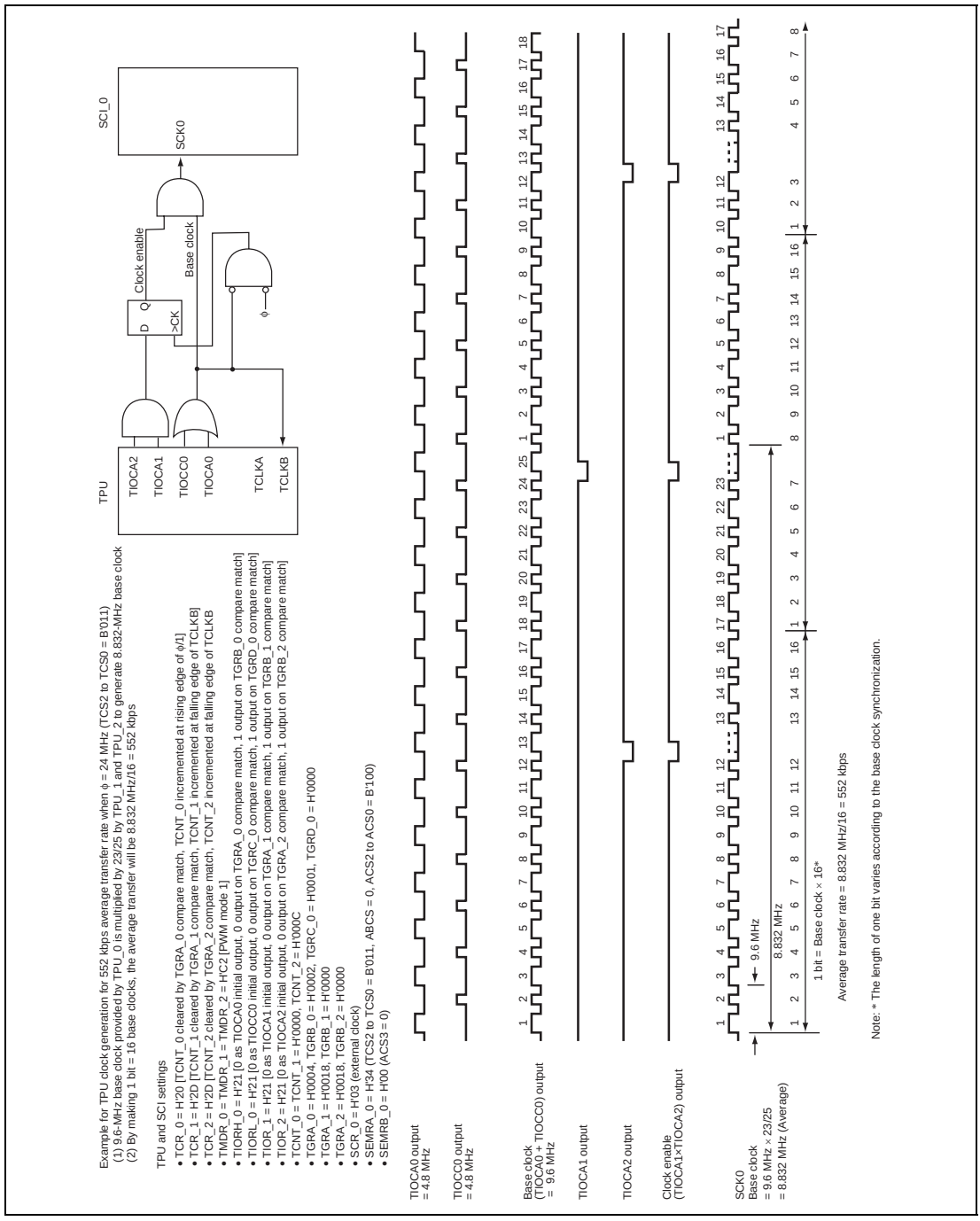



**Figure 12.4 Example of Average Transfer Rate Setting when TPU Clock Is Input (4)**

### 12.3.11 Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 12.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode, clocked synchronous mode, and Smart Card interface mode. The initial value of BRR is H'FF, and it can be read from or written to by the CPU at all times.

**Table 12.2 Relationships between the N Setting in BRR and Bit Rate B**

| Mode | ABCS | Bit Rate | Error |
|---|---|---|---|
| Asynchronous mode | 0 | $B = \frac{\phi \times 10^6}{64 \times 2^{2n-1} \times (N + 1)}$ | $\text{Error (\%)} = \left\{\frac{\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N + 1)} - 1\right\} \times 100$ |
| | 1 | $B = \frac{\phi \times 10^6}{32 \times 2^{2n-1} \times (N + 1)}$ | $\text{Error (\%)} = \left\{\frac{\phi \times 10^6}{B \times 32 \times 2^{2n-1} \times (N + 1)} - 1\right\} \times 100$ |
| Clocked synchronous mode | × | $B = \frac{\phi \times 10^6}{8 \times 2^{2n-1} \times (N + 1)}$ | — |
| Smart Card interface mode | × | $B = \frac{\phi \times 10^6}{S \times 2^{2n+1} \times (N + 1)}$ | $\text{Error (\%)} = \left\{\frac{\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N + 1)} - 1\right\} \times 100$ |

Legend:

B: Bit rate (bps)

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

$\phi$: Operating frequency (MHz)

n, S: Determined by the SMR settings shown in the following tables.

×: Don't care

| SMR Setting | | | |
|---|---|---|---|
| CKS1 | CKS0 | Clock Source | n |
| 0 | 0 | $\phi$ | 0 |
| 0 | 1 | $\phi$/4 | 1 |
| 1 | 0 | $\phi$/16 | 2 |
| 1 | 1 | $\phi$/64 | 3 |

| SMR Setting | | |
|---|---|---|
| BCP1 | BCP0 | S |
| 0 | 0 | 32 |
| 0 | 1 | 64 |
| 1 | 0 | 372 |
| 1 | 1 | 256 |

Table 12.3 shows sample N settings in BRR in normal asynchronous mode. Table 12.4 shows the maximum bit rate for each frequency in normal asynchronous mode. Table 12.6 shows sample N settings in BRR in clocked synchronous mode. Table 12.8 shows sample N settings in BRR in Smart Card interface mode. In Smart Card interface mode, S (the number of basic clock periods in a 1-bit transfer interval) can be selected. For details, see section 12.7.5, Receive Data Sampling

Timing and Reception Margin. Tables 12.5 and 12.7 show the maximum bit rates with external clock input.

When the ABCS bit in SCI_0's serial extended mode register A_0 (SEMRA_0) is set to 1 in asynchronous mode, the maximum bit rates are twice those shown in table 12.3.

**Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode)**

| Bit Rate | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | | | 2.097152 | | | 2.4576 | | | 3 | | |
| (bps) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 1 | 141 | 0.03 | 1 | 148 | –0.04 | 1 | 174 | –0.26 | 1 | 212 | 0.03 |
| 150 | 1 | 103 | 0.16 | 1 | 108 | 0.21 | 1 | 127 | 0.00 | 1 | 155 | 0.16 |
| 300 | 0 | 207 | 0.16 | 0 | 217 | 0.21 | 0 | 255 | 0.00 | 1 | 77 | 0.16 |
| 600 | 0 | 103 | 0.16 | 0 | 108 | 0.21 | 0 | 127 | 0.00 | 0 | 155 | 0.16 |
| 1200 | 0 | 51 | 0.16 | 0 | 54 | –0.70 | 0 | 63 | 0.00 | 0 | 77 | 0.16 |
| 2400 | 0 | 25 | 0.16 | 0 | 26 | 1.14 | 0 | 31 | 0.00 | 0 | 38 | 0.16 |
| 4800 | 0 | 12 | 0.16 | 0 | 13 | –2.48 | 0 | 15 | 0.00 | 0 | 19 | –2.34 |
| 9600 | — | — | — | 0 | 6 | –2.48 | 0 | 7 | 0.00 | 0 | 9 | –2.34 |
| 19200 | — | — | — | — | — | — | 0 | 3 | 0.00 | 0 | 4 | –2.34 |
| 31250 | 0 | 1 | 0.00 | — | — | — | — | — | — | 0 | 2 | 0.00 |
| 38400 | — | — | — | — | — | — | 0 | 1 | 0.00 | — | — | — |

| Bit Rate | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 3.6864 | | | 4 | | | 4.9152 | | | 5 | | |
| (bps) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 64 | 0.70 | 2 | 70 | 0.03 | 2 | 86 | 0.31 | 2 | 88 | –0.25 |
| 150 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 300 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 600 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 1200 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 2400 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 4800 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | –1.36 |
| 9600 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |
| 19200 | 0 | 5 | 0.00 | — | — | — | 0 | 7 | 0.00 | 0 | 7 | 1.73 |
| 31250 | — | — | — | 0 | 3 | 0.00 | 0 | 4 | –1.70 | 0 | 4 | 0.00 |
| 38400 | 0 | 2 | 0.00 | — | — | — | 0 | 3 | 0.00 | 0 | 3 | 1.73 |

| Bit Rate | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 6 | | | 6.144 | | | 7.3728 | | | 8 | | |
| (bps) | 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 | 7 | 0.00 |
| 38400 | 0 | 4 | –2.34 | 0 | 4 | 0.00 | 0 | 5 | 0.00 | — | — | — |

| Bit Rate (bps) | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 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 | –2.34 | 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 (bps) | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 14 | | | 14.7456 | | | 16 | | | 17.2032 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 248 | –0.17 | 3 | 64 | 0.70 | 3 | 70 | 0.03 | 3 | 75 | 0.48 |
| 150 | 2 | 181 | 0.16 | 2 | 191 | 0.00 | 2 | 207 | 0.16 | 2 | 223 | 0.00 |
| 300 | 2 | 90 | 0.16 | 2 | 95 | 0.00 | 2 | 103 | 0.16 | 2 | 111 | 0.00 |
| 600 | 1 | 181 | 0.16 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 223 | 0.00 |
| 1200 | 1 | 90 | 0.16 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 111 | 0.00 |
| 2400 | 0 | 181 | 0.16 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 223 | 0.00 |
| 4800 | 0 | 90 | 0.16 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 111 | 0.00 |
| 9600 | 0 | 45 | –0.93 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 55 | 0.00 |
| 19200 | 0 | 22 | –0.93 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 27 | 0.00 |
| 31250 | 0 | 13 | 0.00 | 0 | 14 | –1.70 | 0 | 15 | 0.00 | 0 | 16 | 1.20 |
| 38400 | – | – | – | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 13 | 0.00 |

| Bit Rate (bps) | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 18 | | | 19.6608 | | | 20 | | | 24 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 79 | –0.12 | 3 | 86 | 0.31 | 3 | 88 | –0.25 | 3 | 106 | –0.44 |
| 150 | 2 | 233 | 0.16 | 2 | 255 | 0.00 | 3 | 64 | 0.16 | 3 | 77 | 0.16 |
| 300 | 2 | 116 | 0.16 | 2 | 127 | 0.00 | 2 | 129 | 0.16 | 2 | 155 | 0.16 |
| 600 | 1 | 233 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 | 2 | 77 | 0.16 |
| 1200 | 1 | 116 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 | 1 | 155 | 0.16 |
| 2400 | 0 | 233 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 | 1 | 77 | 0.16 |
| 4800 | 0 | 116 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 | 0 | 155 | 0.16 |
| 9600 | 0 | 58 | –0.69 | 0 | 63 | 0.00 | 0 | 64 | 0.16 | 0 | 77 | 0.16 |
| 19200 | 0 | 28 | 1.02 | 0 | 31 | 0.00 | 0 | 32 | –1.36 | 0 | 38 | 0.16 |
| 31250 | 0 | 17 | 0.00 | 0 | 19 | –1.70 | 0 | 19 | 0.00 | 0 | 23 | 0.00 |
| 38400 | 0 | 14 | –2.34 | 0 | 15 | 0.00 | 0 | 15 | 1.73 | 0 | 19 | –2.34 |

Note: This table shows bit rates when the ABCS bit in SEMRA_0 is cleared to 0.
When the ABCS bit in SEMR0 is set to 1, the bit rates are twice those shown in this table.
In this LSI, operating frequency ϕ must be 6 MHz or greater.

**Table 12.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode)**

| ϕ (MHz) | Maximum Bit Rate (kbps) | | n | N | ϕ (MHz) | Maximum Bit Rate (kbps) | | n | N |
|---|---|---|---|---|---|---|---|---|---|
| | ABCS = 0 | ABCS = 1 | | | | ABCS = 0 | ABCS = 1 | | |
| 2 | 62.5 | 125.0 | 0 | 0 | 9.8304 | 307.2 | 614.4 | 0 | 0 |
| 2.097152 | 65.536 | 131.027 | 0 | 0 | 10 | 312.5 | 625.0 | 0 | 0 |
| 2.4576 | 76.8 | 153.6 | 0 | 0 | 12 | 375.0 | 750.0 | 0 | 0 |
| 3 | 93.75 | 187.5 | 0 | 0 | 12.288 | 384.0 | 768.0 | 0 | 0 |
| 3.6864 | 115.2 | 230.4 | 0 | 0 | 14 | 437.5 | 875.0 | 0 | 0 |
| 4 | 125.0 | 250.0 | 0 | 0 | 14.7456 | 460.8 | 921.6 | 0 | 0 |
| 4.9152 | 153.6 | 307.2 | 0 | 0 | 16 | 500.0 | 1000.0 | 0 | 0 |
| 5 | 156.25 | 312.5 | 0 | 0 | 17.2032 | 537.6 | 1075.2 | 0 | 0 |
| 6 | 187.5 | 375.0 | 0 | 0 | 18 | 562.5 | 1125.0 | 0 | 0 |
| 6.144 | 192.0 | 384.0 | 0 | 0 | 19.6608 | 614.4 | 1228.8 | 0 | 0 |
| 7.3728 | 230.4 | 460.8 | 0 | 0 | 20 | 625.0 | 1250.0 | 0 | 0 |
| 8 | 250.0 | 500.0 | 0 | 0 | 24 | 750.0 | 1500.0 | 0 | 0 |

**Table 12.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)**

| ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (kbps) ABCS = 0 | Maximum Bit Rate (kbps) ABCS = 1 | ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (kbps) ABCS = 0 | Maximum Bit Rate (kbps) ABCS = 1 |
|---|---|---|---|---|---|---|---|
| 2 | 0.5000 | 31.25 | 62.5 | 9.8304 | 2.4576 | 153.6 | 307.2 |
| 2.097152 | 0.5243 | 327.68 | 65.536 | 10 | 2.5000 | 156.25 | 312.5 |
| 2.4576 | 0.6144 | 38.4 | 76.8 | 12 | 3.0000 | 187.5 | 375.0 |
| 3 | 0.7500 | 46.875 | 93.75 | 12.288 | 3.0720 | 192.0 | 384.0 |
| 3.6864 | 0.9216 | 57.6 | 115.2 | 14 | 3.5000 | 218.75 | 437.0 |
| 4 | 1.0000 | 62.5 | 125.0 | 14.7456 | 3.6864 | 230.4 | 460.8 |
| 4.9152 | 1.2288 | 76.8 | 153.6 | 16 | 4.0000 | 250.0 | 500.0 |
| 5 | 1.2500 | 78.125 | 156.25 | 17.2032 | 4.3008 | 268.8 | 537.6 |
| 6 | 1.5000 | 93.75 | 187.5 | 18 | 4.5000 | 281.25 | 562.5 |
| 6.144 | 1.5360 | 96.0 | 192.0 | 19.6608 | 4.9152 | 307.2 | 614.4 |
| 7.3728 | 1.8432 | 115.2 | 230.4 | 20 | 5.0000 | 312.5 | 625.0 |
| 8 | 2.0000 | 125.0 | 250.0 | 24 | 6.0000 | 375.0 | 750.0 |

Note: In this LSI, operating frequency ϕ must be 6 MHz or greater.

**Table 12.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)**

| Bit Rate (bps) | Operating Frequency φ (MHz) | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | | 4 | | 6 | | 8 | | 10 | | 16 | | 20 | | 24 | |
| | n | N | n | N | n | N | n | N | n | N | n | N | n | N | n | N |
| 110 | 3 | 70 | — | — | | | | | | | | | | | | |
| 250 | 2 | 124 | 2 | 249 | | | 3 | 124 | — | — | 3 | 249 | | | | |
| 500 | 1 | 249 | 2 | 124 | | | 2 | 249 | — | — | 3 | 124 | — | — | — | — |
| 1 k | 1 | 124 | 1 | 249 | | | 2 | 124 | — | — | 2 | 249 | — | — | — | — |
| 2.5 k | 0 | 199 | 1 | 99 | 1 | 149 | 1 | 199 | 1 | 249 | 2 | 99 | 2 | 124 | 2 | 149 |
| 5 k | 0 | 99 | 0 | 199 | 1 | 74 | 1 | 99 | 1 | 124 | 1 | 199 | 1 | 249 | 2 | 74 |
| 10 k | 0 | 49 | 0 | 99 | 0 | 149 | 0 | 199 | 0 | 249 | 1 | 99 | 1 | 124 | 1 | 149 |
| 25 k | 0 | 19 | 0 | 39 | 0 | 59 | 0 | 79 | 0 | 99 | 0 | 159 | 0 | 199 | 0 | 239 |
| 50 k | 0 | 9 | 0 | 19 | 0 | 29 | 0 | 39 | 0 | 49 | 0 | 79 | 0 | 99 | 0 | 119 |
| 100 k | 0 | 4 | 0 | 9 | 0 | 14 | 0 | 19 | 0 | 24 | 0 | 39 | 0 | 49 | 0 | 59 |
| 250 k | 0 | 1 | 0 | 3 | 0 | 5 | 0 | 7 | 0 | 9 | 0 | 15 | 0 | 19 | 0 | 23 |
| 500 k | 0 | 0* | 0 | 1 | 0 | 2 | 0 | 3 | 0 | 4 | 0 | 7 | 0 | 9 | 0 | 11 |
| 1 M | | | 0 | 0* | | | 0 | 1 | | | 0 | 3 | 0 | 4 | 0 | 5 |
| 2 M | | | | | | | 0 | 0* | | | 0 | 1 | | | 0 | 2 |
| 2.5 M | | | | | | | | | 0 | 0* | | | 0 | 1 | | |
| 4 M | | | | | | | | | | | 0 | 0* | | | | |
| 5 M | | | | | | | | | | | | | 0 | 0* | | |
| 6 M | | | | | | | | | | | | | | | 0 | 0* |

Legend:

Blank: Cannot be set.

—: Can be set, but there will be a degree of error.

*: Continuous transfer is not possible.

**Table 12.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)**

| φ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (Mbps) | φ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (Mbps) |
|---|---|---|---|---|---|
| 2 | 0.333 | 0.333 | 14 | 2.333 | 2.333 |
| 4 | 0.667 | 0.667 | 16 | 2.667 | 2.667 |
| 6 | 1.000 | 1.000 | 18 | 3.000 | 3.000 |
| 8 | 1.333 | 1.333 | 20 | 3.333 | 3.333 |
| 10 | 1.667 | 1.667 | 24 | 4.000 | 4.000 |
| 12 | 2.000 | 2.000 | | | |

Note: In this LSI, operating frequency φ must be 6 MHz or greater.

**Table 12.8 BRR Settings for Various Bit Rates (Smart Card Interface Mode, when n = 0 and S = 372)**

| | Operating Frequency ϕ (MHz) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 5.00 | | 7.00 | | 7.1424 | | 10.00 | | 10.7136 | | 13.00 | |
| Bit Rate (bps) | N | Error (%) | N | Error (%) | N | Error (%) | N | Error (%) | N | Error (%) | N | Error (%) |
| 6720 | 0 | 0.01 | 1 | 30.00 | 1 | 28.57 | 1 | 0.01 | 1 | 7.14 | 2 | 13.33 |
| 9600 | 0 | 30.00 | 0 | 1.99 | 0 | 0.00 | 1 | 30.00 | 1 | 25.00 | 1 | 8.99 |

| | Operating Frequency ϕ (MHz) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 14.2848 | | 16.00 | | 18.00 | | 20.00 | | 24.00 | |
| Bit Rate (bps) | N | Error (%) | N | Error (%) | N | Error (%) | N | Error (%) | N | Error (%) |
| 6720 | 2 | 4.76 | 2 | 6.67 | 3 | 0.01 | 3 | 0.01 | 4 | 3.99 |
| 9600 | 1 | 0.00 | 1 | 12.01 | 2 | 15.99 | 2 | 6.66 | 2 | 12.01 |

**Table 12.9 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode)**

| | Maximum Bit Rate (bps) | | | | | |
|---|---|---|---|---|---|---|
| ϕ (MHz) | S = 32 | S = 64 | S = 256 | S = 372 | n | N |
| 5.00 | 78125 | 39063 | 9766 | 6720 | 0 | 0 |
| 6.00 | 93750 | 46875 | 11719 | 8065 | 0 | 0 |
| 7.00 | 109375 | 54688 | 13672 | 9409 | 0 | 0 |
| 7.1424 | 111600 | 55800 | 13950 | 9600 | 0 | 0 |
| 10.00 | 156250 | 78125 | 19531 | 13441 | 0 | 0 |
| 10.7136 | 167400 | 83700 | 20925 | 14400 | 0 | 0 |
| 13.00 | 203125 | 101563 | 25391 | 17473 | 0 | 0 |
| 14.2848 | 223200 | 111600 | 27900 | 19200 | 0 | 0 |
| 16.00 | 250000 | 125000 | 31250 | 21505 | 0 | 0 |
| 18.00 | 281250 | 140625 | 35156 | 24194 | 0 | 0 |
| 20.00 | 312500 | 156250 | 39063 | 26882 | 0 | 0 |
| 24.00 | 375000 | 187500 | 46875 | 32258 | 0 | 0 |

Note: In this LSI, operating frequency ϕ must be 6 MHz or greater.

## 12.4 Operation in Asynchronous Mode

Figure 12.5 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally stop bits (high level). In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line. When the transmission line goes to the space state (low level), the SCI recognizes a start bit and starts serial communication. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex. Both the transmitter and the receiver also have a double-buffered structure, so data can be read from or written during transmission or reception, enabling continuous data transfer.

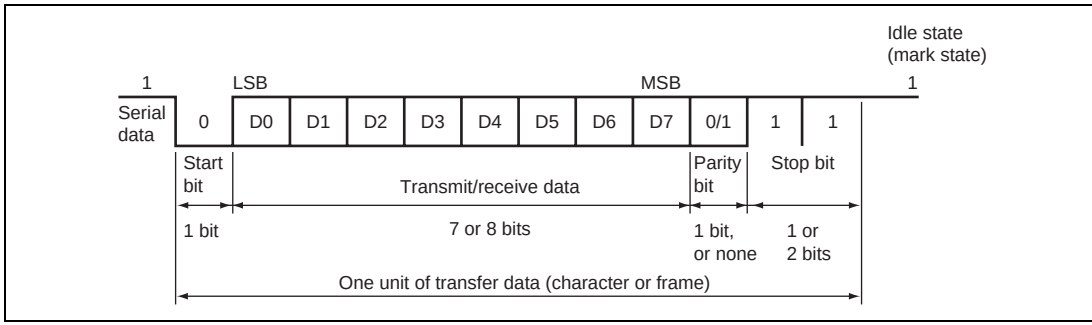



**Figure 12.5 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)**

### 12.4.1 Data Transfer Format

Table 12.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting. For details on the multiprocessor bit, refer to section 12.5, Multiprocessor Communication Function.

**Table 12.10 Serial Transfer Formats (Asynchronous Mode)**

| SMR 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

### 12.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the transfer rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the basic clock as shown in Figure 12.6. Thus, the reception margin in asynchronous mode is given by formula (1) below.

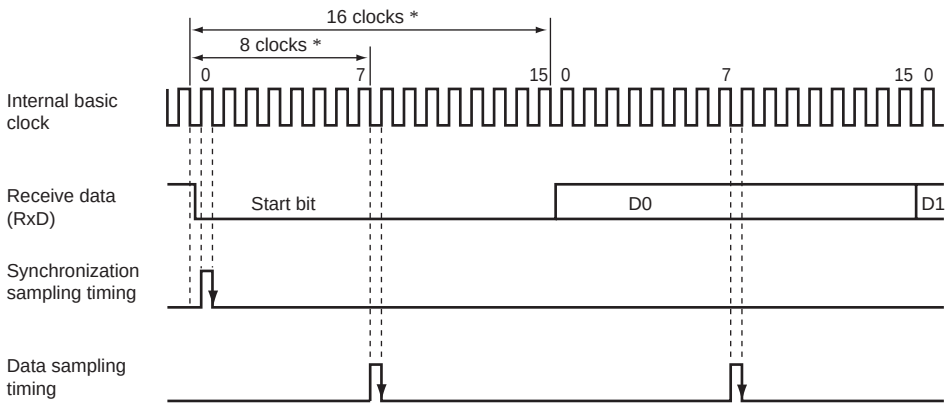
$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)\,F - \frac{|\,D - 0.5\,|}{N}\,(1 + F)\right| \times 100\ [\%] \qquad \text{... Formula (1)}$$

Where M: Reception margin
N: Ratio of bit rate to clock (N = 16 if ABCS = 0, N = 8 if ABCS = 1)
D: Clock duty (D = 0 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute value of clock rate deviation

Assuming values of F (absolute value of clock rate deviation) = 0, D (clock duty) = 0.5, and N (ratio of bit rate to clock) = 16 in formula (1), the reception margin can be given by the formula.

$M = \{0.5 - 1/(2 \times 16)\} \times 100\ [\%] = 46.875\%$

However, this is only the computed value, and a margin of 20% to 30% should be allowed for in system design.




Note: * In this example the value of the ABCS bit in SEMRA_0 is 0. When ABCS is set to 1, the basic clock frequency is eight times the bit rate and the receive data is sampled at the fourth rising edge of the basic clock.

**Figure 12.6 Receive Data Sampling Timing in Asynchronous Mode**

Note: * Figure 12.6 shows an example when the ABCS bit of SEMRA_0 is cleared to 0. When ABCS is set to 1, the clock frequency of basic clock is 8 times the bit rate and the receive data is sampled at the rising edge of the 4th pulse of the basic clock.

### 12.4.3 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/$\overline{A}$ bit in SMR and the CKE0 and CKE1 bits in SCR. When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used. When an external clock is selected, the basic clock of average transfer rate can be selected according to the ACS2 to ACS0 bit setting of SEMR_0.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin by setting CKE1 = 0 and CKE0 = 1. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 12.7.

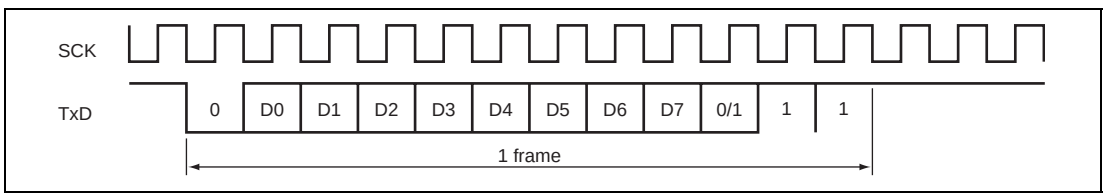
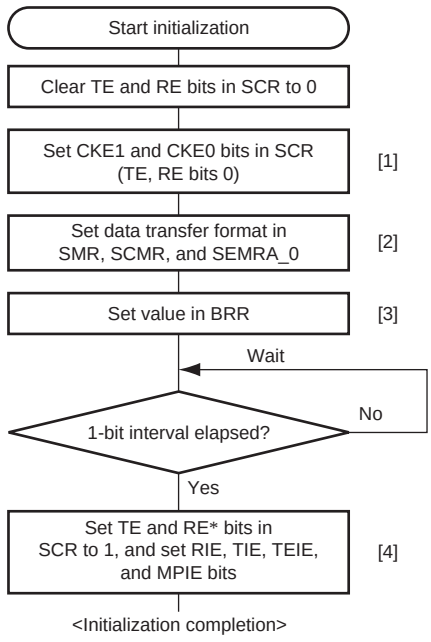



**Figure 12.7 Relationship between Output Clock and Transfer Data Phase (Asynchronous Mode)**

### 12.4.4 SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 12.8. When the operating mode, or transfer format, is changed for example, 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. Note that clearing the RE bit to 0 does not initialize the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization.




[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock is selected in asynchronous mode, it is output immediately after SCR settings are made.

[2] Set the data transfer format in SMR, SCMR, and SEMRA_0.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock or average transfer rate clock by ACS2 to ACS0 is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits.

Setting the TE and RE bits enables use of the TxD and RxD pins.

Note: * Perform this set operation with the RxD pin in the 1 state. If the RE bit is set to 1 with the RxD pin in the 0 state, it may be misinterpreted as a start bit.

**Figure 12.8 Sample SCI Initialization Flowchart**

### 12.4.5 Data Transmission (Asynchronous Mode)

Figure 12.9 shows an example of operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR. If the flag is cleared to 0, the SCI recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit data empty interrupt request (TXI) is generated. Continuous transmission is possible because the TXI interrupt routine writes next transmit data to TDR before transmission of the current transmit data has been completed.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit or multiprocessor bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the "mark state" is entered, in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

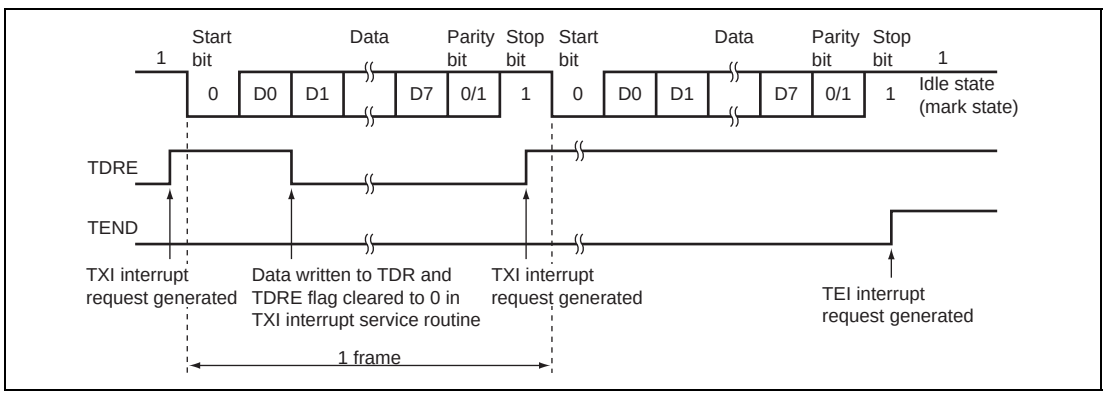
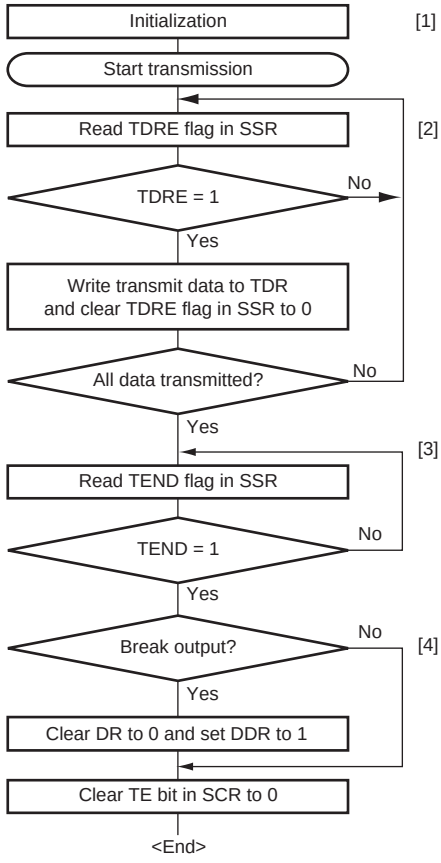



**Figure 12.9 Example of Operation in Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)**

Figure 12.10 shows a sample flowchart for transmission in asynchronous mode.




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then 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 TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port corresponding to the TxD pin to 1, clear DR to 0, then clear the TE bit in SCR to 0.

**Figure 12.10 Sample Serial Data Transmission Flowchart**

### 12.4.6 Serial Data Reception (Asynchronous Mode)

Figure 12.11 shows an example of operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line. If a start bit is detected, the SCI performs internal synchronization, receives receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error occurs (when reception of the next data is completed while the RDRF flag is still set to 1), the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error is detected (when the stop bit is 0), the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception is completed successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Continuous reception is possible because the RXI interrupt routine reads the receive data transferred to RDR before reception of the next receive data has been completed.

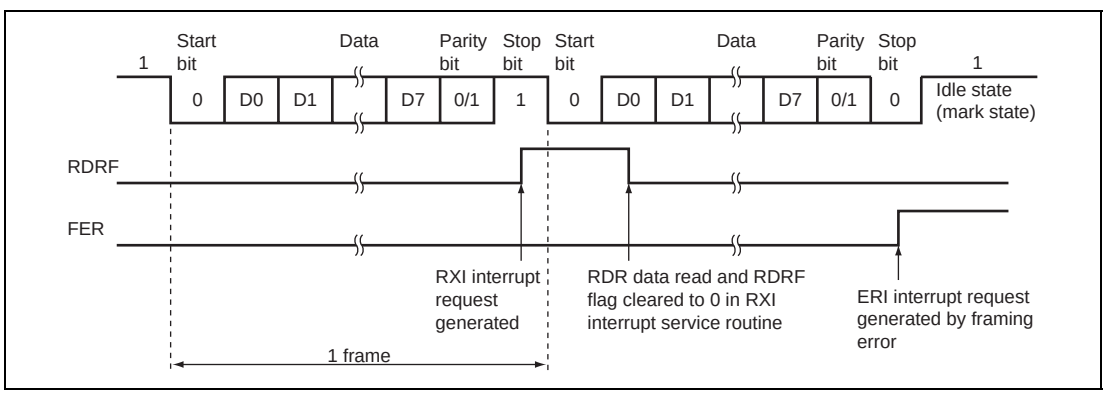


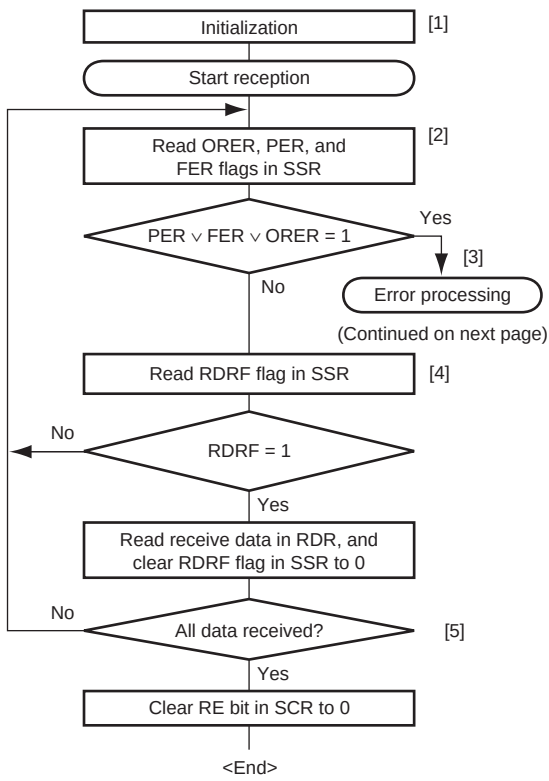

**Figure 12.11 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit)**

Table 12.11 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 12.12 shows a sample flow chart for serial data reception.

**Table 12.11 SSR Status Flags and Receive Data Handling**

| SSR Status Flag | | | | | |
|---|---|---|---|---|---|
| **RDRF*** | **ORER** | **FER** | **PER** | **Receive Data** | **Receive Error Type** |
| 1 | 1 | 0 | 0 | Lost | Overrun error |
| 0 | 0 | 1 | 0 | Transferred to RDR | Framing error |
| 0 | 0 | 0 | 1 | Transferred to RDR | Parity error |
| 1 | 1 | 1 | 0 | Lost | Overrun error + framing error |
| 1 | 1 | 0 | 1 | Lost | Overrun error + parity error |
| 0 | 0 | 1 | 1 | Transferred to RDR | Framing error + parity error |
| 1 | 1 | 1 | 1 | Lost | Overrun error + framing error + parity error |

Note: * The RDRF flag retains the state it had before data reception.




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing and break detection:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.

[4] SCI status check and receive data read:
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the end bit for the current frame is received, reading the RDRF flag and RDR, and clearing the RDRF flag to 0 should be finished. The RDRF flag is cleared automatically when DMAC is activated by a reception complete interrupt (RXI) and the RDR value is read.

**Figure 12.12 Sample Serial Data Reception Flowchart (1)**

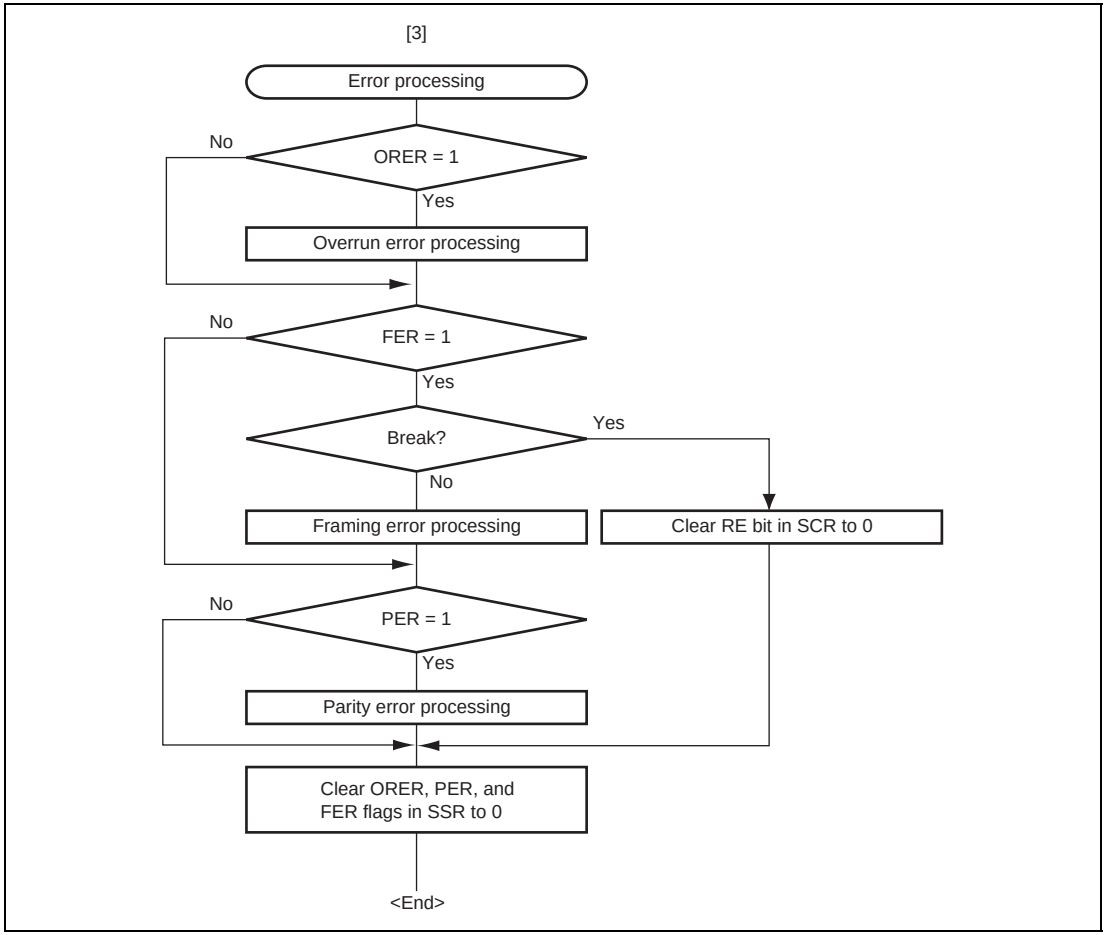



**Figure 12.12 Sample Serial Data Reception Flowchart (2)**

## 12.5 Multiprocessor Communication Function

Use of the multiprocessor communication function enables data transfer between a number of processors sharing communication lines by asynchronous serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data. When multiprocessor communication is performed, each receiving station is addressed by a unique ID code. The serial communication cycle consists of two component cycles; an ID transmission cycle that 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. If the multiprocessor bit is 1, the cycle is an ID transmission cycle; if the multiprocessor bit is 0, the cycle is a data transmission cycle. Figure 12.13 shows an example of inter-processor communication using the multiprocessor format. The transmitting station first sends the ID code of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added. 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 do not match continue to skip data until data with a 1 multiprocessor bit is again received.

The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set to 1, transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status flags, RDRF, FER, and ORER to 1, are inhibited until data with a 1 multiprocessor bit is received. On reception of a receive character with a 1 multiprocessor bit, the MPB bit in SSR is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt is generated.

When the multiprocessor format is selected, the parity bit setting is rendered invalid. All other bit settings are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.

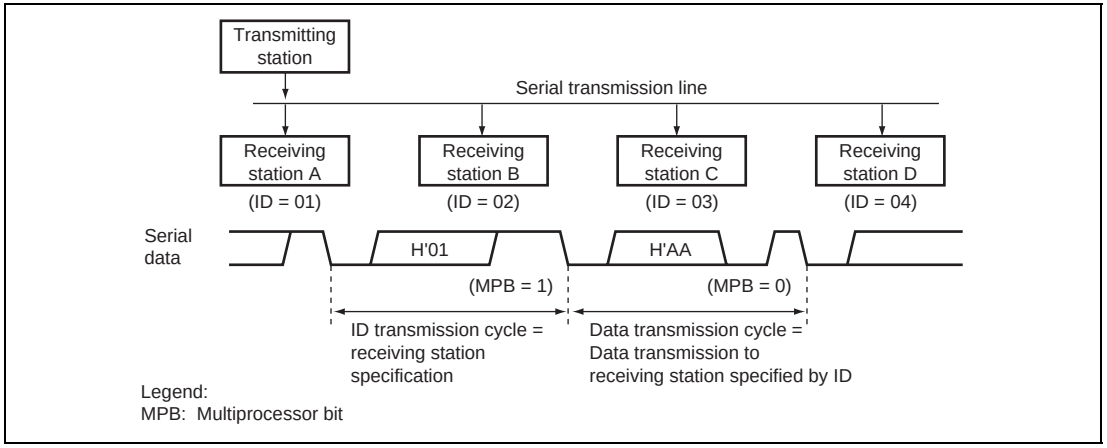



**Figure 12.13 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)**

### 12.5.1 Multiprocessor Serial Data Transmission

Figure 12.14 shows a sample flowchart for multiprocessor serial data transmission. For an ID transmission cycle, set the MPBT bit in SSR to 1 before transmission. For a data transmission cycle, clear the MPBT bit in SSR to 0 before transmission. All other SCI operations are the same as those in asynchronous mode.

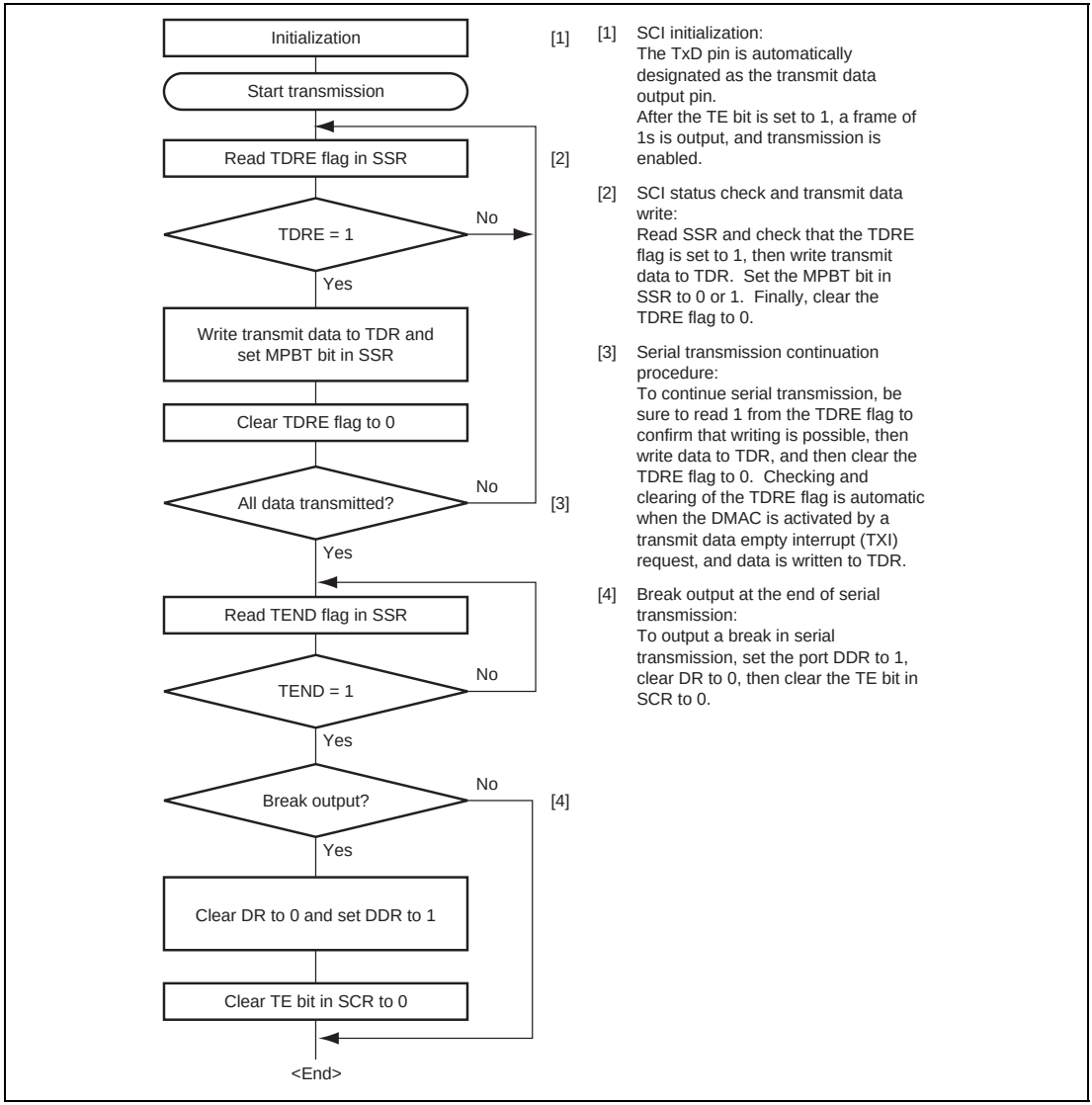



**Figure 12.14 Sample Multiprocessor Serial Data Transmission Flowchart**

### 12.5.2 Multiprocessor Serial Data Reception

Figure 12.16 shows a sample flowchart for multiprocessor serial data reception. If the MPIE bit in SCR is set to 1, data is skipped until data with a 1 multiprocessor bit is sent. On receiving data with a 1 multiprocessor bit, the receive data is transferred to RDR. An RXI interrupt request is generated at this time. All other SCI operations are the same as in asynchronous mode. Figure 12.15 shows an example of SCI operation for multiprocessor format reception.

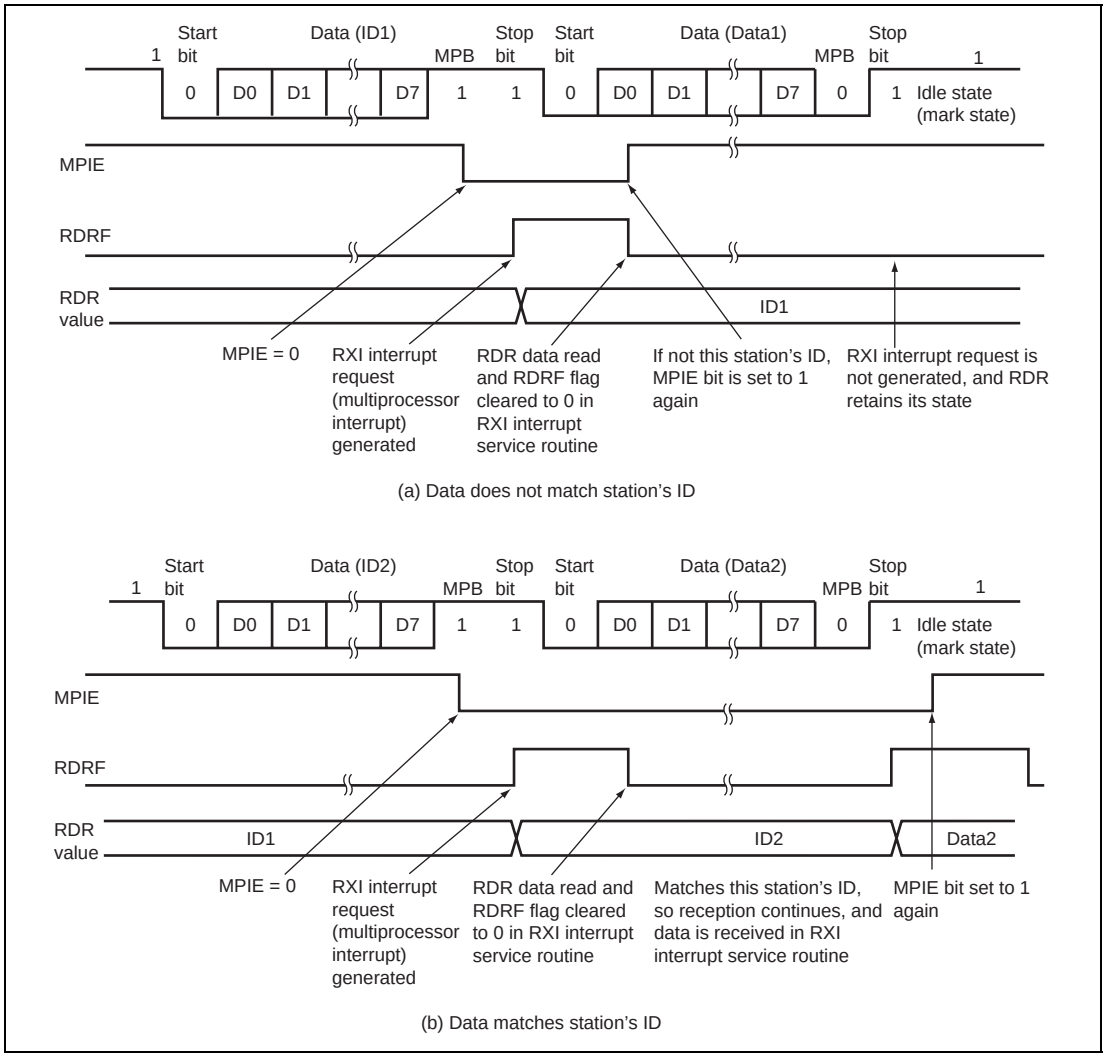
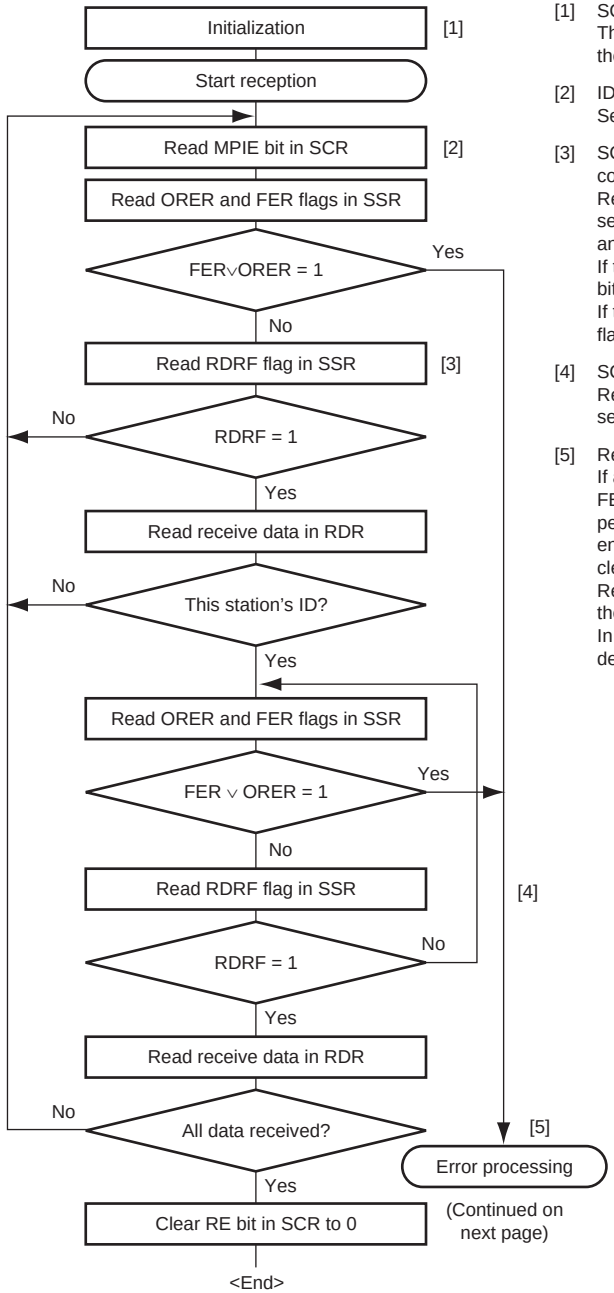



**Figure 12.15 Example of SCI Operation in Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**




[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] ID reception cycle:
Set the MPIE bit in SCR to 1.

[3] SCI status check, ID reception and comparison:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID.
If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0.
If the data is this station's ID, clear the RDRF flag to 0.

[4] SCI status check and data reception:
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.

[5] Receive error processing and break detection:
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER and FER flags are all cleared to 0.
Reception cannot be resumed if either of these flags is set to 1.
In the case of a framing error, a break can be detected by reading the RxD pin value.

**Figure 12.16 Sample Multiprocessor Serial Data Reception Flowchart (1)**

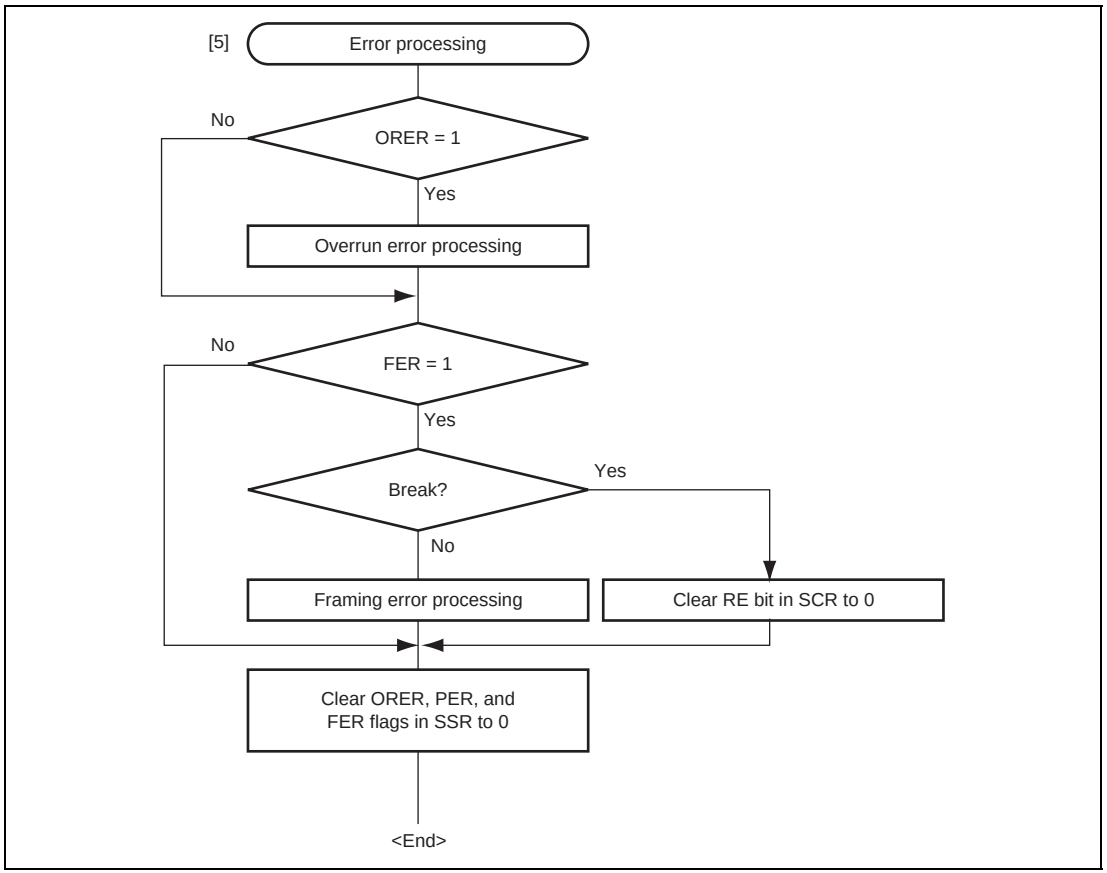



**Figure 12.16 Sample Multiprocessor Serial Data Reception Flowchart (2)**

## 12.6 Operation in Clocked Synchronous Mode

Figure 12.17 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received synchronous with clock pulses. In clocked synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. In clocked synchronous mode, the SCI receives data in synchronous with the rising edge of the serial clock. After 8-bit data is output, the transmission line holds the MSB state. In clocked synchronous mode, no parity or multiprocessor bit is added. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication through the use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so data can be read from or written during transmission or reception, enabling continuous data transfer.

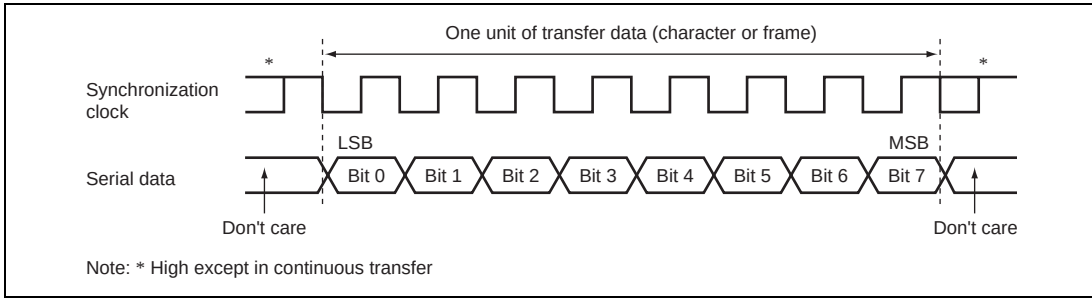


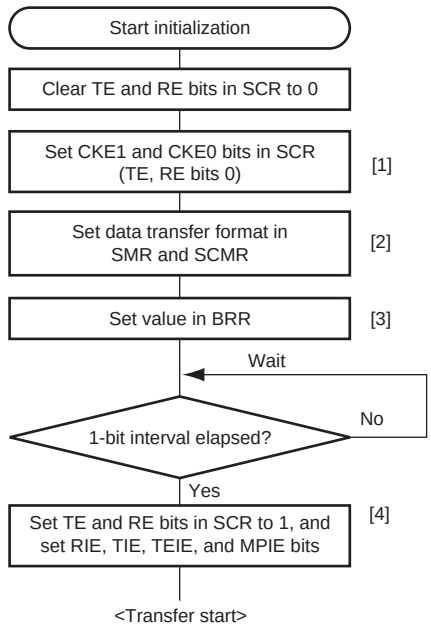

**Figure 12.17 Data Format in Synchronous Communication (For LSB-First)**

### 12.6.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK pin can be selected, according to the setting of CKE0 and CKE1 bits in SCR. 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.

### 12.6.2 SCI Initialization (Clocked Synchronous Mode)

Before transmitting and receiving data, the TE and RE bits in SCR should be cleared to 0, then the SCI should be initialized as described in a sample flowchart in figure 12.18. When the operating mode, or transfer format, is changed for example, 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. 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 RDR.




[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, TE and RE, to 0.

[2] Set the data transfer format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1.
Also set the RIE, TIE TEIE, and MPIE bits.
Setting the TE and RE bits enables the TxD and RxD pins to be used.

Note: In simultaneous transmit and receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

**Figure 12.18 Sample SCI Initialization Flowchart**

### 12.6.3 Serial Data Transmission (Clocked Synchronous Mode)

Figure 12.19 shows an example of SCI operation for transmission in clocked synchronous mode. In serial transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if the flag is 0, the SCI recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a transmit data empty interrupt (TXI) is generated. Continuous transmission is possible because the TXI interrupt routine writes the next transmit data to TDR before transmission of the current transmit data has been completed.
3. 8-bit data is sent from the TxD pin synchronized with the output clock when output clock mode has been specified, and synchronized with the input clock when use of an external clock has been specified.
4. The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).
5. If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TDRE flag maintains the output state of the last bit. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated. The SCK pin is fixed high.

Figure 12.20 shows a sample flow chart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure that the receive error flags are cleared to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.

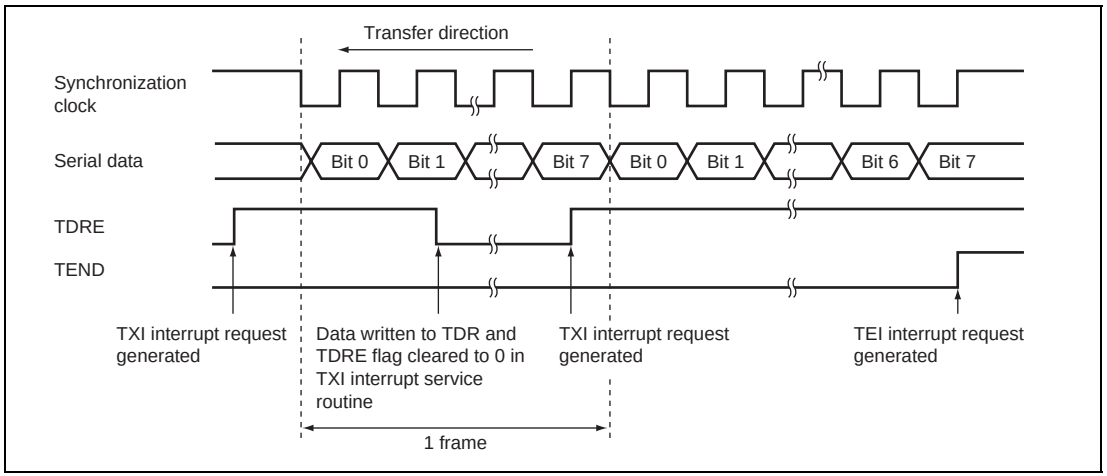
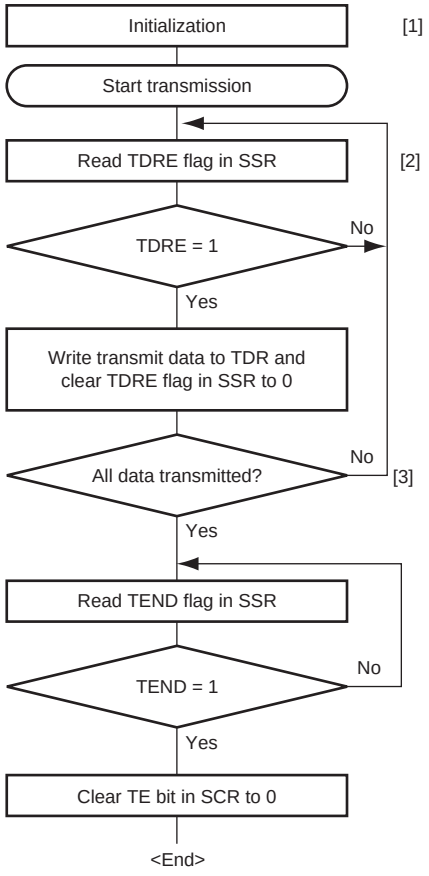



**Figure 12.19 Sample SCI Transmission Operation in Clocked Synchronous Mode**




[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0.
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 TDR.

**Figure 12.20 Sample Serial Data Transmission Flowchart**

### 12.6.4 Serial Data Reception (Clocked Synchronous Mode)

Figure 12.21 shows an example of SCI operation for reception in clocked synchronous mode. In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization synchronous with a synchronous clock input or output, starts receiving data, and stores the received data in RSR.
2. If an overrun error occurs (when reception of the next data is completed while the RDRF flag in SSR is still set to 1), the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated, receive data is not transferred to RDR, and the RDRF flag remains to be set to 1.
3. If reception is completed successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Continuous reception is possible because the RXI interrupt routine reads the receive data transferred to RDR before reception of the next receive data has finished.

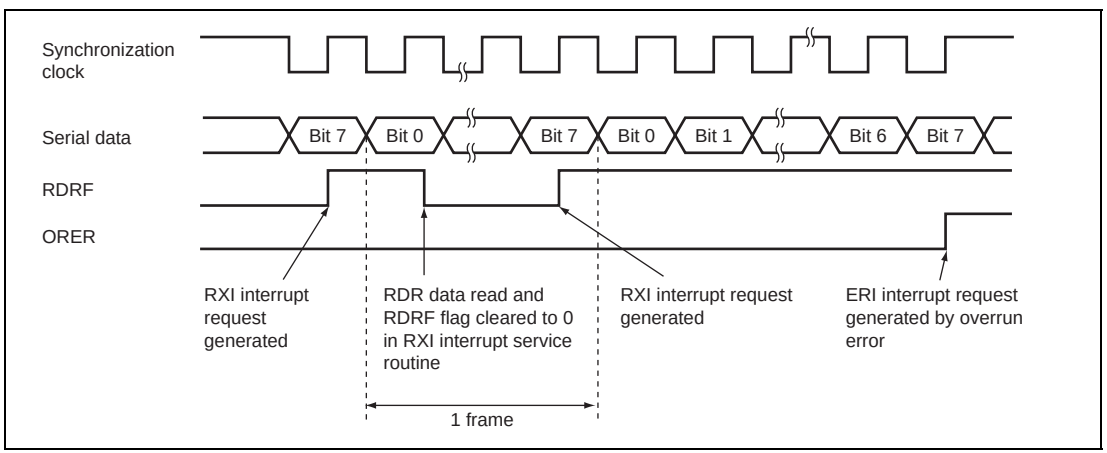



**Figure 12.21 Example of SCI Operation in Reception**

Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 12.22 shows a sample flow chart for serial data reception.

When the internal clock is selected during reception, the synchronization clock will be output until an overrun error occurs or the RE bit is cleared. To receive data in frame units, a dummy data of one frame must be transmitted simultaneously.

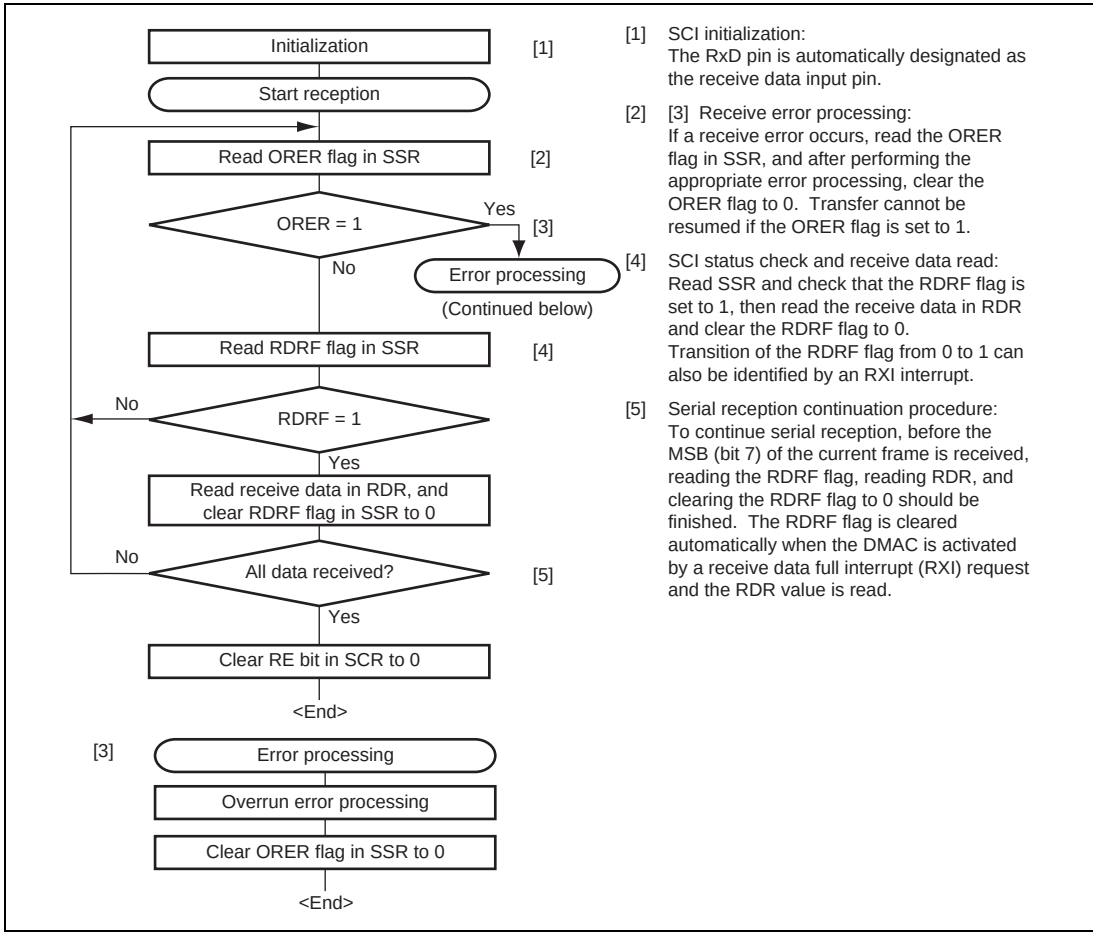
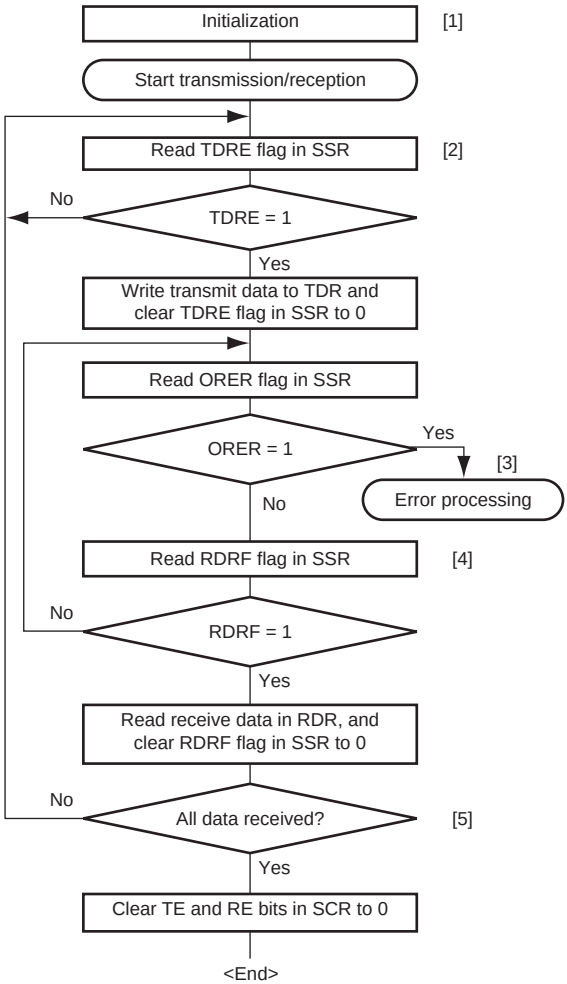



**Figure 12.22 Sample Serial Data Reception Flowchart**

### 12.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode)

Figure 12.23 shows a sample flowchart for simultaneous serial transmit and receive operations. The following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags are set to 1, clear TE to 0. Then simultaneously set TE and RE to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI has finished reception, clear RE to 0. Then after checking that the RDRF and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set TE and RE to 1 with a single instruction.




Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

[1] SCI initialization:
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.

[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.
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 TDR. Also, the RDRF flag is cleared automatically when the DMAC is activated by a receive data full interrupt (RXI) request and the RDR value is read.

**Figure 12.23 Sample Flowchart of Simultaneous Serial Data Transmit and Receive Operations**

## 12.7 Operation in Smart Card Interface

The SCI supports an IC card (Smart Card) interface that conforms to ISO/IEC 7816-3 (Identification Card) as a serial communication interface extension function. Switching between the normal serial communication interface and the Smart Card interface mode is carried out by means of a register setting.

### 12.7.1 Pin Connection Example

Figure 12.24 shows an example of connection with the Smart Card. In communication with an IC card, as both transmission and reception are carried out on a single data transmission line, the TxD pin and RxD pin should be connected to the LSI pin. The data transmission line should be pulled up to the $V_{CC}$ power supply with a resistor. If an IC card is not connected, and the TE and RE bits are both set to 1, closed transmission/reception is possible, enabling self-diagnosis to be carried out. 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. This LSI port output is used as the reset signal.

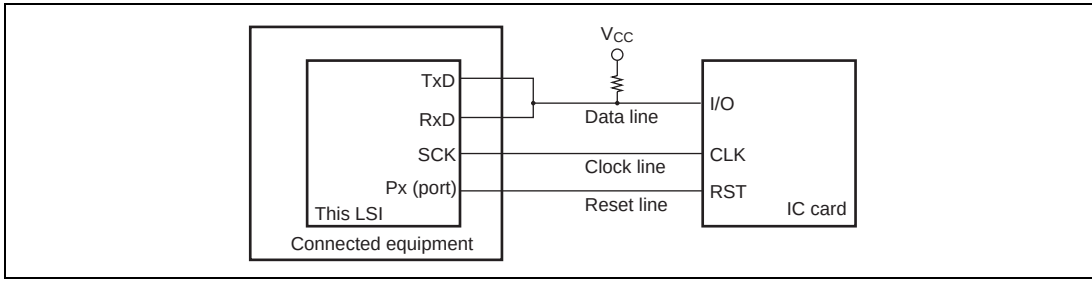



**Figure 12.24 Schematic Diagram of Smart Card Interface Pin Connections**

### 12.7.2 Data Format (Except for Block Transfer Mode)

Figure 12.25 shows the transfer data format in Smart Card interface mode.

- One frame consists of 8-bit data plus a parity bit in asynchronous mode.
- 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 one etu period, 10.5 etu after the start bit.
- If an error signal is sampled during transmission, the same data is retransmitted automatically after a delay of 2 etu or longer.

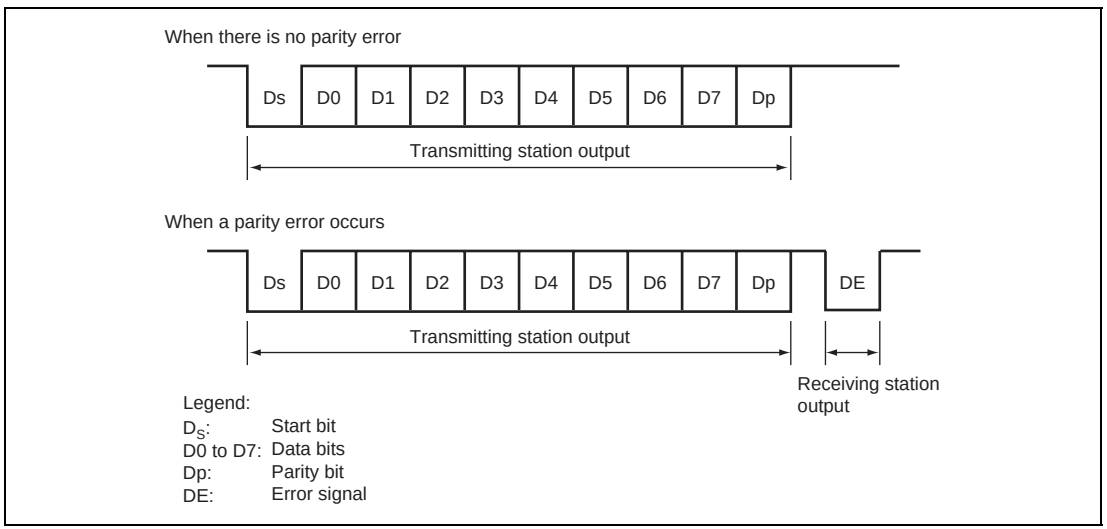



**Figure 12.25 Normal Smart Card Interface Data Format**

Data transfer with other types of IC cards (direct convention and inverse convention) are performed as described in the following.

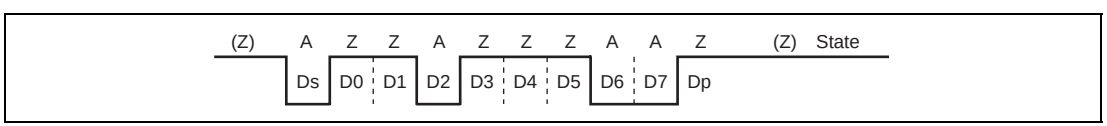



**Figure 12.26 Direct Convention (SDIR = SINV = O/$\overline{E}$ = 0)**

With the direction convention type IC and the above sample start character, 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 above is H'3B. For the direct convention type, clear the SDIR and SINV

bits in SCMR to 0. According to Smart Card regulations, clear the O/$\overline{E}$ bit in SMR to 0 to select even parity mode.

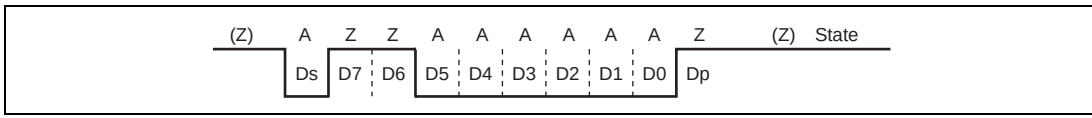



**Figure 12.27 Inverse Convention (SDIR = SINV = O/$\overline{E}$ = 1)**

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 for the above is H'3F. For the inverse convention type, set the SDIR and SINV bits in SCMR to 1. According to Smart Card regulations, even parity mode is the logic 0 level of the parity bit, and corresponds to state Z. In this LSI, the SINV bit inverts only data bits D0 to D7. Therefore, set the O/$\overline{E}$ bit in SMR to 1 to invert the parity bit for both transmission and reception.

### 12.7.3 Clock

As a transmit/receive clock, only an internal clock which is generated by an on-chip baud rate generator can be used. When clock output is selected by setting CKE0 to 1, a clock with a frequency S times the bit rate is output from the SCK pin.

Note: Symbol S is the value of S described in section 12.3.11, Bit Rate Register (BRR).

### 12.7.4 Block Transfer Mode

Operation in block transfer mode is the same as that in the normal Smart Card interface mode, except for the following points.

- In reception, though the parity check is performed, no error signal is output even if an error is detected. However, the PER bit in SSR is set to 1 and must be cleared before receiving the parity bit of the next frame.
- In transmission, a guard time of at least 1 etu is left between the end of the parity bit and the start of the next frame.
- In transmission, because retransmission is not performed, the TEND flag is set to 1, 11.5 etu after transmission start.
- As with the normal Smart Card interface, the ERS flag indicates the error signal status, but since error signal transfer is not performed, this flag is always cleared to 0.

### 12.7.5 Receive Data Sampling Timing and Reception Margin

In Smart Card interface mode an internal clock generated by the on-chip baud rate generator can only be used as a transmission/reception clock. In this mode, the SCI operates on a basic clock with a frequency of 32, 64, 372, or 256 times the transfer rate (fixed to 16 times in normal asynchronous mode) as determined by bits BCP1 and BCP0. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. As shown in figure 12.28, by sampling receive data at the rising-edge of the 16th, 32nd, 186th, or 128th pulse of the basic clock, data can be latched at the middle of the bit. The reception margin is given by the following formula.

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)\,F - \frac{|\,D - 0.5\,|}{N}\,(1 + F) \right| \times 100\ [\%]$$

Where M: Reception margin (%)
N: Ratio of bit rate to clock (N = 32, 64, 372, and 256)
D: Clock duty (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock frequency deviation

Assuming values of F = 0, D = 0.5 and N = 372 in the above formula, the reception margin formula is as follows.

$$M = (0.5 - 1/2 \times 372) \times 100\%$$
$$= 49.866\%$$

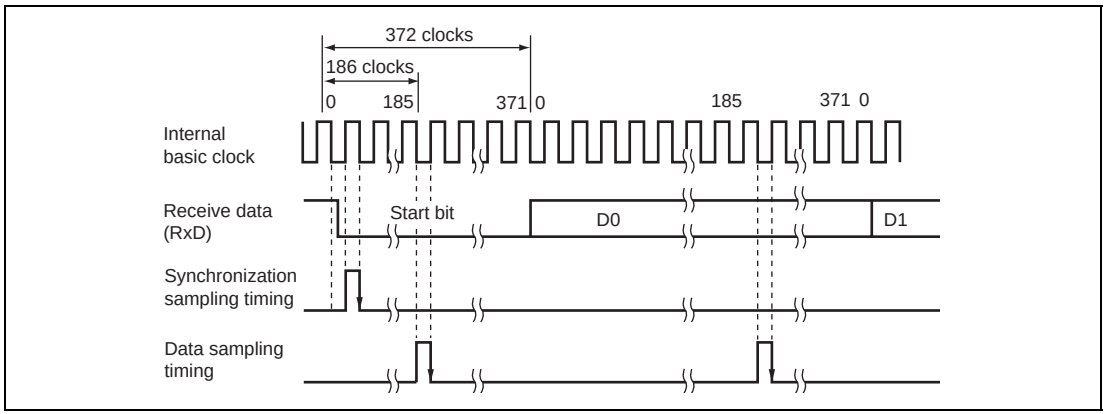



**Figure 12.28 Receive Data Sampling Timing in Smart Card Mode (Using Clock of 372 Times the Transfer Rate)**

### 12.7.6 Initialization

Before transmitting and receiving data, initialize the SCI as described below. Initialization is also necessary when switching from transmit mode to receive mode, or vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Clear the error flags ERS, PER, and ORER in SSR to 0.
3. Set the GM, BLK, O/$\overline{E}$, BCP0, BCP1, CKS0, CKS1 bits in SMR. Set the PE bit to 1.
4. Set the SMIF, SDIR, and SINV bits in SCMR.
   When the SMIF bit is set to 1, the TxD and RxD pins are both switched from ports to SCI pins, and are placed in the high-impedance state.
5. Set the value corresponding to the bit rate in BRR.
6. Set the CKE0 and CKE1 bits in SCR. 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 SCR. Do not set the TE bit and RE bit at the same time, except for self-diagnosis.

To switch from receive mode to transmit mode, after checking that the SCI has finished reception, initialize the SCI, and set RE to 0 and TE to 1. Whether SCI has finished reception or not can be checked with the RDRF, PER, or ORER flags. To switch from transmit mode to receive mode, after checking that the SCI has finished transmission, initialize the SCI, and set TE to 0 and RE to 1. Whether SCI has finished transmission or not can be checked with the TEND flag.

### 12.7.7 Serial Data Transmission (Except for Block Transfer Mode)

As data transmission in Smart Card interface mode involves error signal sampling and retransmission processing, the operations are different from those in normal serial communication interface mode (except for block transfer mode). Figure 12.29 illustrates the retransfer operation when the SCI is in transmit mode.

1. If an error signal is sent back from the receiving end after transmission of one frame is complete, the ERS bit in SSR is set to 1. If the RIE bit in SCR is enabled at this time, an ERI interrupt request is generated. The ERS bit in SSR should be cleared to 0 by the time the next parity bit is sampled.
2. The TEND bit in SSR is not set for a frame in which an error signal indicating an abnormality is received. Data is retransferred from TDR to TSR, and retransmitted automatically.
3. If an error signal is not sent back from the receiving end, the ERS bit in SSR is not set. Transmission of one frame, including a retransfer, is judged to have been completed, and the TEND bit in SSR is set to 1. If the TIE bit in SCR is enabled at this time, a TXI interrupt request is generated. Writing transmit data to TDR transfers the next transmit data.

Figure 12.31 shows a flowchart for transmission. A sequence of transmit operations can be performed automatically by specifying the DMAC to be activated with a TXI interrupt source. In a transmit operation, the TDRE flag is set to 1 at the same time as the TEND flag in SSR is set, and a TXI interrupt will be generated if the TIE bit in SCR has been set to 1. 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 TDRE and TEND flags are automatically cleared to 0 when data is transferred by the DMAC. In the event of an error, the SCI retransmits the same data automatically. During this period, the TEND flag remains cleared to 0 and the DMAC is not activated. Therefore, the SCI and DMAC will automatically transmit the specified number of bytes in the event of an error, including retransmission. However, the ERS flag is not cleared automatically when an error occurs, and so 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. When using the DMAC for data transmission or reception, always make DMAC settings and enable the DMAC before making SCI settings. For details on DMAC settings, see section 7, DMA Controller (DMAC).

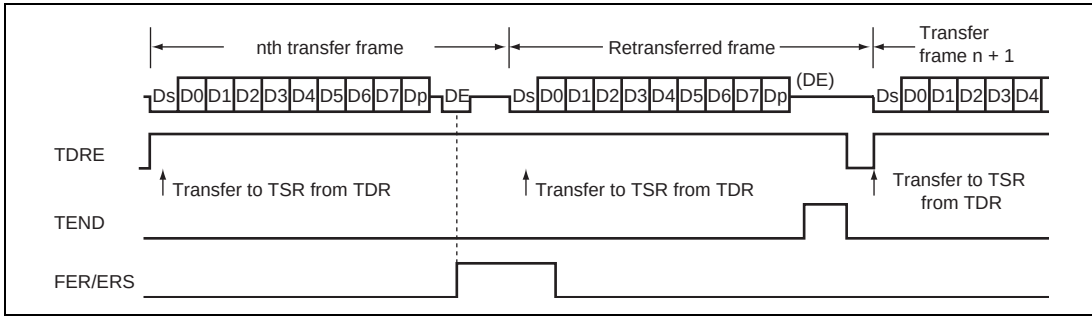



**Figure 12.29 Retransfer Operation in SCI Transmit Mode**

The timing for setting the TEND flag depends on the value of the GM bit in SMR. The TEND flag set timing is shown in figure 12.30.

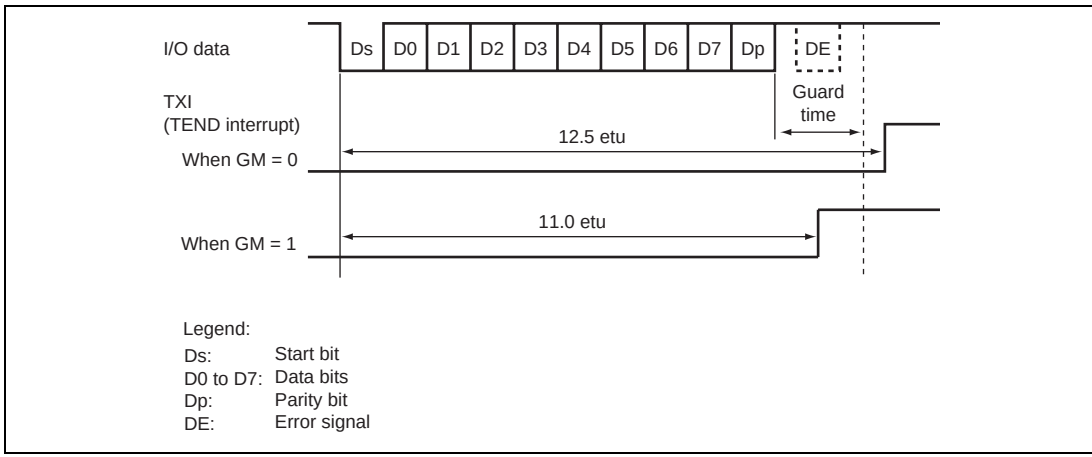



**Figure 12.30 TEND Flag Generation Timing in Transmission Operation**

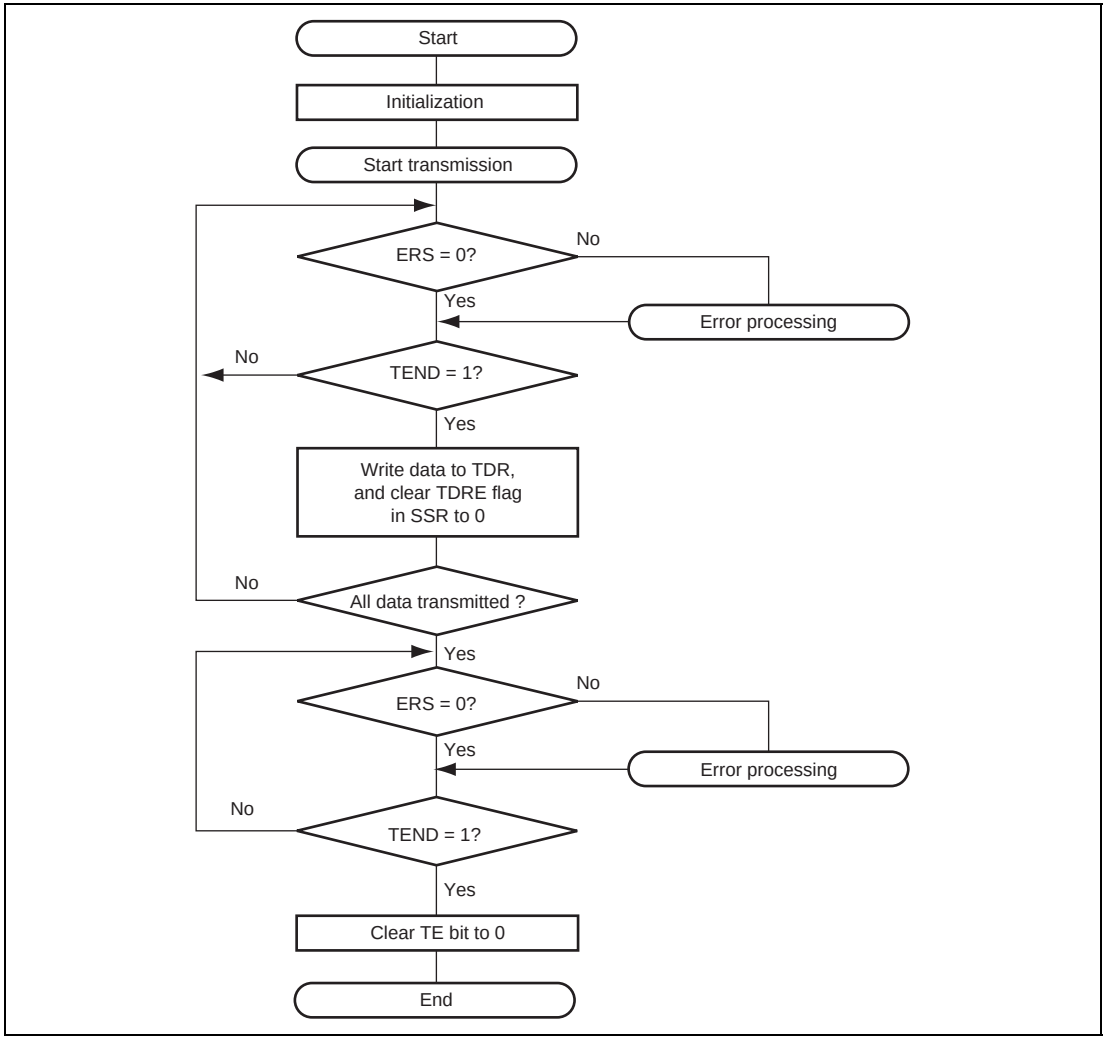



**Figure 12.31 Example of Transmission Processing Flow**

### 12.7.8 Serial Data Reception (Except for Block Transfer Mode)

Data reception in Smart Card interface mode uses the same operation procedure as for normal serial communication interface mode. Figure 12.32 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 SSR is automatically set to 1. If the RIE bit in SCR is set at this time, an ERI interrupt request is generated. The PER bit in SSR should be kept cleared to 0 until the next parity bit is sampled.
2. The RDRF bit in SSR is not set for a frame in which an error has occurred.
3. If no error is found when the received parity bit is checked, the PER bit in SSR is not set to 1, the receive operation is judged to have been completed normally, and the RDRF flag in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an RXI interrupt request is generated.

Figure 12.33 shows a flowchart for reception. A sequence of receive operations can be performed automatically by specifying the DMAC to be activated using an RXI interrupt source. In a receive operation, an RXI interrupt request is generated when the RDRF flag in SSR 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 the receive data will be transferred. The RDRF flag is cleared to 0 automatically when data is transferred by the DMAC. If an error occurs in receive mode and the ORER or PER flag is set to 1, a transfer error interrupt (ERI) request will be generated. Hence, so the error flag must be cleared to 0. In the event of an error, the DMAC is not activated and receive data is skipped. Therefore, receive data is transferred for only the specified number of bytes in the event of an error. Even when a parity error occurs in receive mode and the PER flag is set to 1, the data that has been received is transferred to RDR and can be read from there.

Note: For details on receive operations in block transfer mode, refer to section 12.4, Operation in Asynchronous Mode.

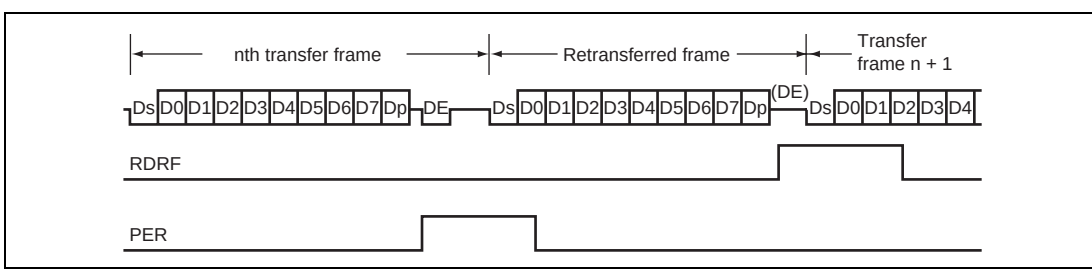



**Figure 12.32 Retransfer Operation in SCI Receive Mode**

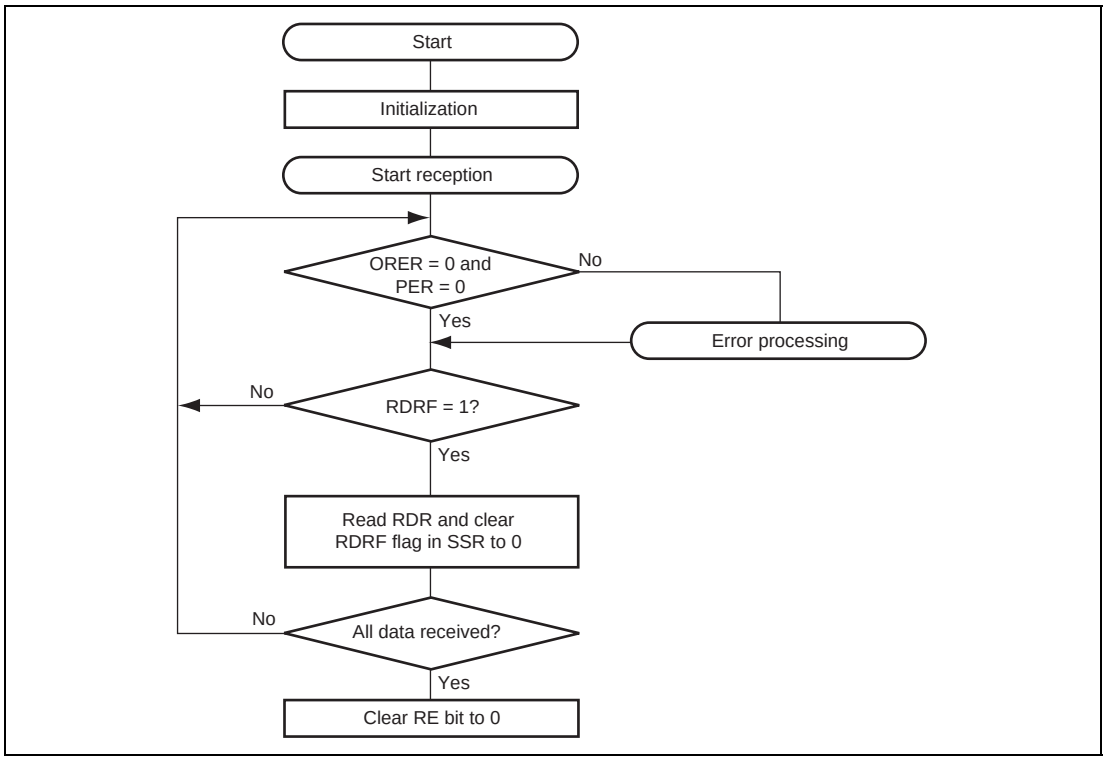



**Figure 12.33 Example of Reception Processing Flow**

### 12.7.9 Clock Output Control

When the GM bit in SMR is set to 1, the clock output level can be fixed with bits CKE0 and CKE1 in SCR. At this time, the minimum clock pulse width can be made the specified width. Figure 12.34 shows the timing for fixing the clock output level. In this example, GM is set to 1, CKE1 is cleared to 0, and the CKE0 bit is controlled.

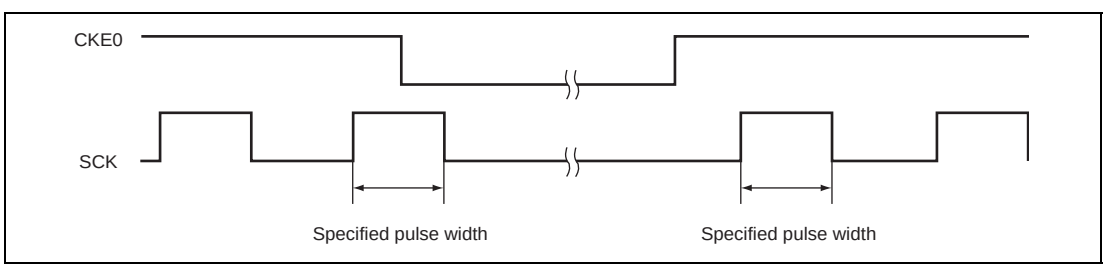



**Figure 12.34 Timing for Fixing Clock Output Level**

When turning on the power or switching between Smart Card interface mode and software standby mode, the following procedures should be followed in order to maintain the clock duty.

**Powering On:** To secure clock duty from power-on, the following switching procedure should be followed.

1. The initial state is port input and high impedance. Use a pull-up resistor or pull-down resistor to fix the potential.
2. Fix the SCK pin to the specified output level with the CKE1 bit in SCR.
3. Set SMR and SCMR, and switch to smart card mode operation.
4. Set the CKE0 bit in SCR to 1 to start clock output.

**When changing from smart card interface mode to software standby mode:**

1. Set the data register (DR) and data direction register (DDR) corresponding to the SCK pin to the value for the fixed output state in software standby mode.
2. Write 0 to the TE bit and RE bit in the serial control register (SCR) to halt transmit/receive operation. At the same time, set the CKE1 bit to the value for the fixed output state in software standby mode.
3. Write 0 to the CKE0 bit in SCR to halt the clock.
4. Wait for one serial clock period.
   During this interval, clock output is fixed at the specified level, with the duty preserved.
5. Make the transition to the software standby state.

**When returning to smart card interface mode from software standby mode:**

1. Exit the software standby state.
2. Write 1 to the CKE0 bit in SCR and output the clock. Signal generation is started with the normal duty.

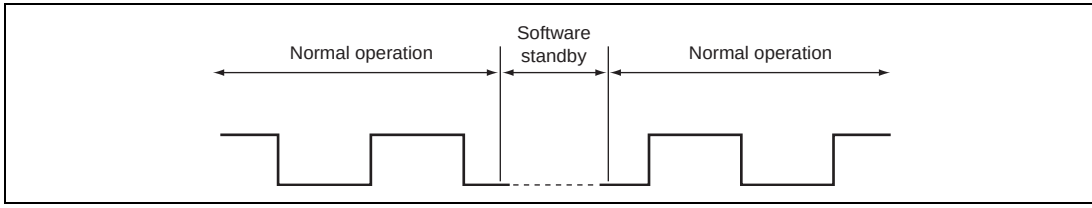



**Figure 12.35 Clock Halt and Restart Procedure**

## 12.8 SCI Select Function (Clocked Synchronous Mode)

The SCI_0 supports the SCI select function which allows clock synchronous communication between master LSI and one of multiple slave LSI. Figure 12.36 shows an example of communication using the SCI select function. Figure 12.37 shows the operation.

The master LSI can communicate with slave LSI_A by bringing $\overline{\text{SEL_A}}$ and $\overline{\text{SEL_B}}$ signals low and high, respectively. In this case, the TxD0_B pin of the slave LSI_B is brought high-impedance state and the internal SCK0_A signal is fixed high. This halts the communication operation of slave LSI_B. The master LSI can communicate with slave LSI_B by bringing the $\overline{\text{SEL_A}}$ and $\overline{\text{SEL_B}}$ signals high and low, respectively.

The slave LSI detects the selection by receiving the low level input from the $\overline{\text{IRQ7}}$ pin and immediately executes data transmission/reception processing.

Note: The selection signals ($\overline{\text{SEL_A}}$ and $\overline{\text{SEL_B}}$) of the LSI must be switched while the serial clock (M_SCK) is high after the end bit of the transmit data has been send. Note that one selection signal can be brought low at the same time.

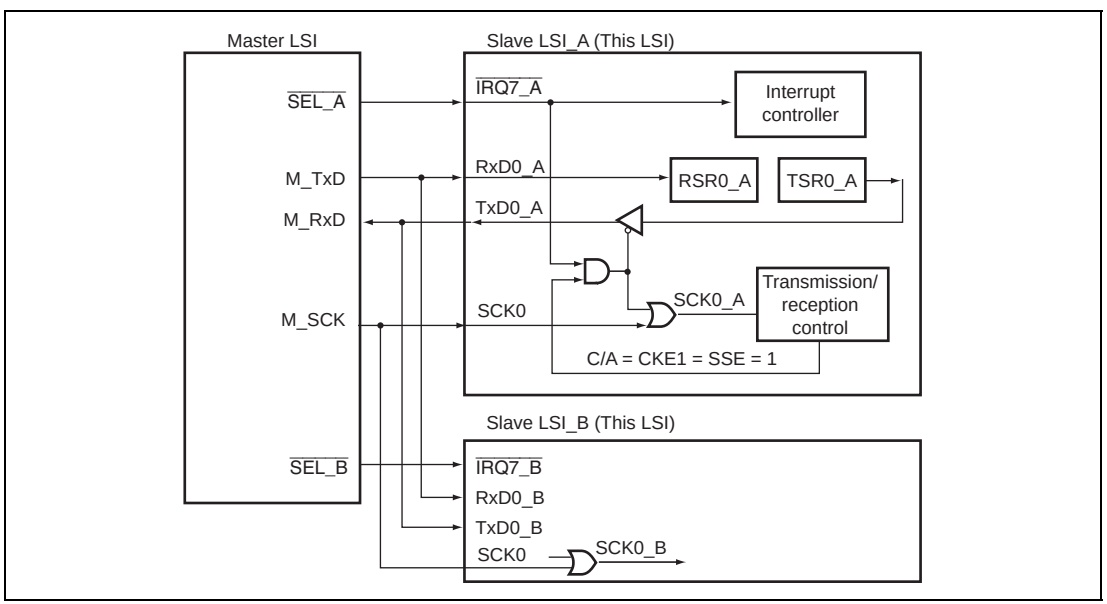



**Figure 12.36 Example of Communication Using the SCI Select Function**

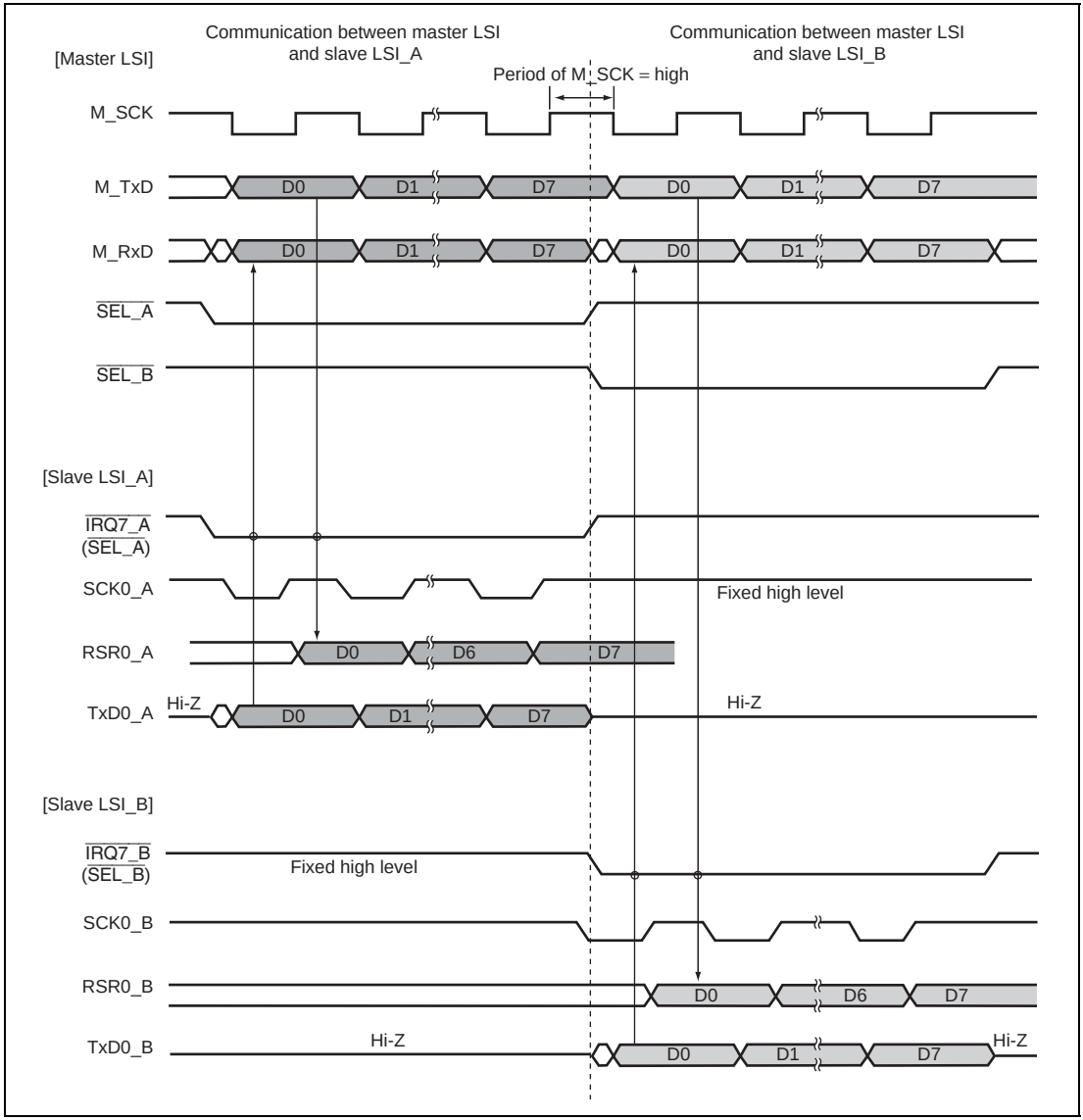



**Figure 12.37 Example of Communication Using the SCI Select Function**

## 12.9 Interrupts

### 12.9.1 Interrupts in Normal Serial Communication Interface Mode

Table 12.12 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt can activate the DMAC to perform data transfer. The TDRE flag is cleared to 0 automatically when data is transferred by the DMAC.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt request can activate the DMAC to transfer data. The RDRF flag is cleared to 0 automatically when data is transferred by the DMAC.

A TEI interrupt is requested when the TEND flag is set to 1 and the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, if the TDRE and TEND flags are cleared simultaneously by the TXI interrupt routine, the SCI cannot branch to the TEI interrupt routine later.

**Table 12.12 SCI Interrupt Sources**

| Channel | Name | Interrupt Source | Interrupt Flag | DMAC Activation | Priority* |
|---|---|---|---|---|---|
| 0 | ERI0 | Receive Error | ORER, FER, PER | Not possible | High |
| | RXI0 | Receive Data Full | RDRF | Possible | |
| | TXI0 | Transmit Data Empty | TDRE | Possible | |
| | TEI0 | Transmission End | TEND | Not possible | |
| 2 | ERI2 | Receive Error | ORER, FER, PER | Not possible | |
| | RXI2 | Receive Data Full | RDRF | Not possible | |
| | TXI2 | Transmit Data Empty | TDRE | Not possible | |
| | TEI2 | Transmission End | TEND | Not possible | Low |

Note: * This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

### 12.9.2 Interrupts in Smart Card Interface Mode

Table 12.13 shows the interrupt sources in Smart Card interface mode. The transmit end interrupt (TEI) request cannot be used in this mode.

Note: In case of block transfer mode, see section 12.9.1, Interrupts in Normal Serial Communication Interface Mode.

**Table 12.13 Interrupt Sources in Smart Card Interface Mode**

| Channel | Name | Interrupt Source | Interrupt Flag | DMAC Activation | Priority* |
|---|---|---|---|---|---|
| 0 | ERI0 | Receive Error, detection | ORER, PER, ERS | Not possible | High |
| | RXI0 | Receive Data Full | RDRF | Possible | |
| | TXI0 | Transmit Data Empty | TEND | Possible | |
| 2 | ERI2 | Receive Error, detection | ORER, PER, ERS | Not possible | |
| | RXI2 | Receive Data Full | RDRF | Not possible | |
| | TXI2 | Transmit Data Empty | TEND | Not possible | Low |

Note: * Indicates the initial state immediately after a reset.
Priorities in channels can be changed by the interrupt controller.

## 12.10 Usage Notes

### 12.10.1 Module Stop Mode Setting

SCI operation can be disabled or enabled using the module stop control register. The initial setting is for SCI operation to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 20, Power-Down Modes.

### 12.10.2 Break Detection and Processing (Asynchronous Mode Only)

When framing error detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, setting the FER flag, and possibly the PER flag. Note that as the SCI continues the receive operation after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

### 12.10.3 Mark State and Break Detection (Asynchronous Mode Only)

When TE is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line at mark state until TE is set to 1, set both DDR and DR to 1. As TE is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set PCR to 1 and PDR to 0, and then clear TE to 0. When TE is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

### 12.10.4 Receive Error Flags and Transmit Operations (Clocked 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 cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

### 12.10.5 Restrictions on Use of DMAC

- When an external clock source is used as the serial clock, the transmit clock should not be input until at least 5 ϕ clock cycles after TDR is updated by the DMAC. Misoperation may occur if the transmit clock is input within 4 ϕ clocks after TDR is updated. (figure 12.38)
- When RDR is read by the DMAC, be sure to set the activation source to the relevant SCI reception end interrupt (RXI).

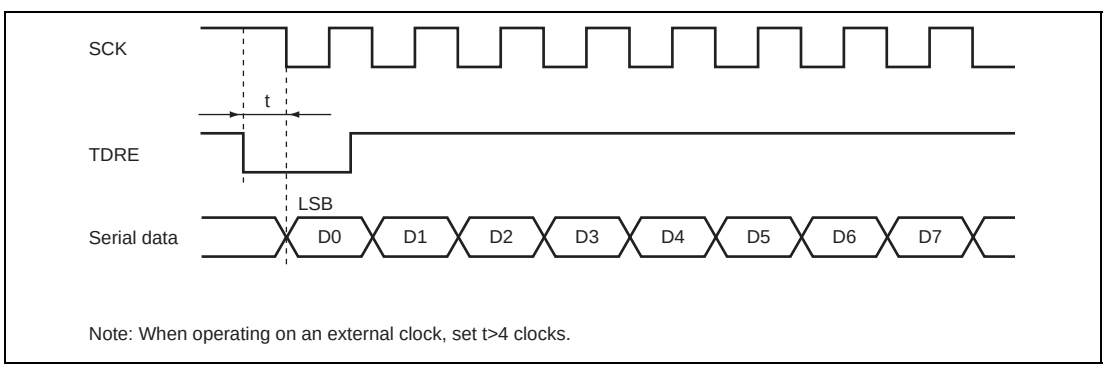



**Figure 12.38 Example of Clocked Synchronous Transmission by DMAC**

### 12.10.6 Operation in Case of Mode Transition

- Transmission

  Operation should be stopped (by clearing TE, TIE, and TEIE to 0) before making a module stop mode, software standby mode, watch mode, subactive mode, or subsleep mode transition. TSR, TDR, and SSR are reset. The output pin states in module stop mode, software standby mode, watch mode, subactive mode, or subsleep mode depend on the port settings, and becomes high-level output after the relevant mode is cleared. If a transition is made during transmission, the data being transmitted will be undefined. When transmitting without changing the transmit mode after the relevant mode is cleared, transmission can be started by setting TE to 1 again, and performing the following sequence:

  SSR read -> TDR write -> TDRE clearance. To transmit with a different transmit mode after clearing the relevant mode, the procedure must be started again from initialization. Figure 12.39 shows a sample flowchart for mode transition during transmission. Port pin states are shown in figures 12.40 and 12.41.

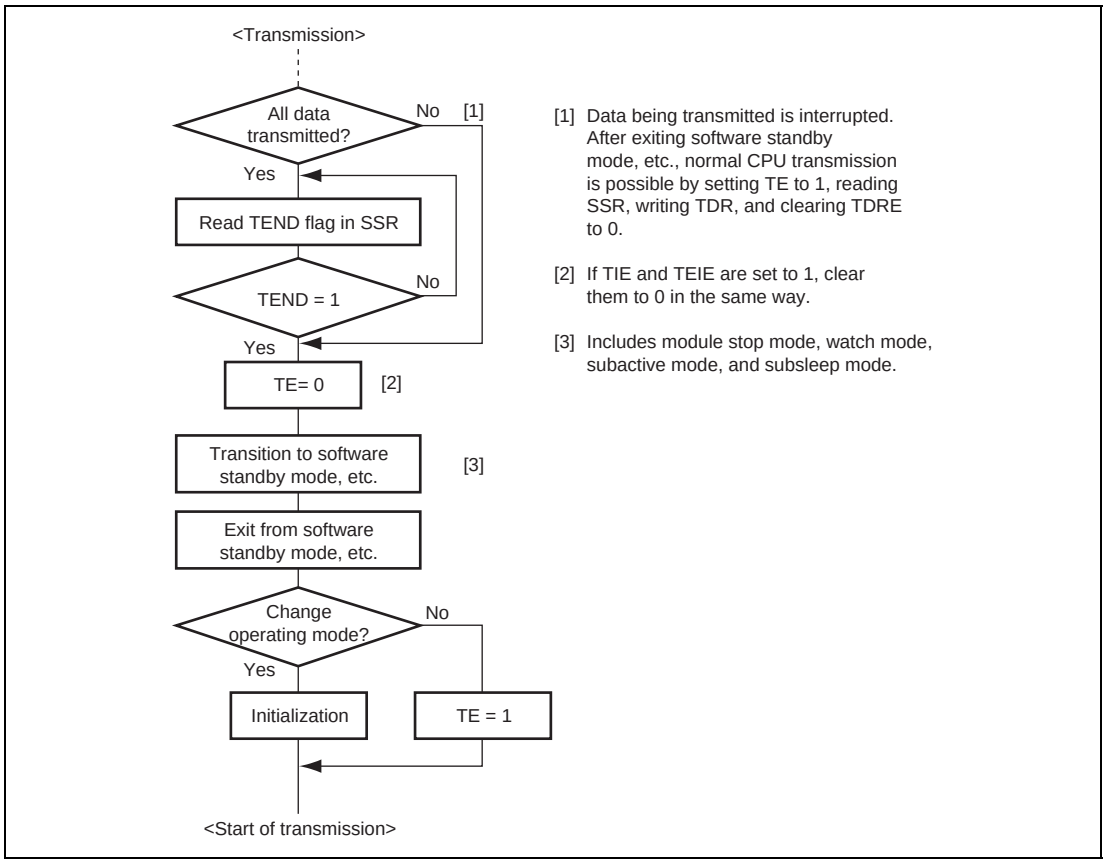



**Figure 12.39 Sample Flowchart for Mode Transition during Transmission**

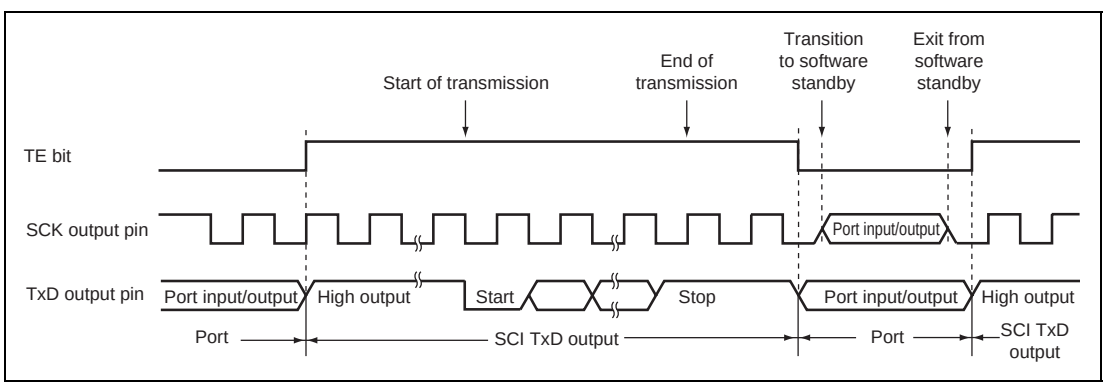



**Figure 12.40 Port Pin State of Asynchronous Transmission Using Internal Clock**

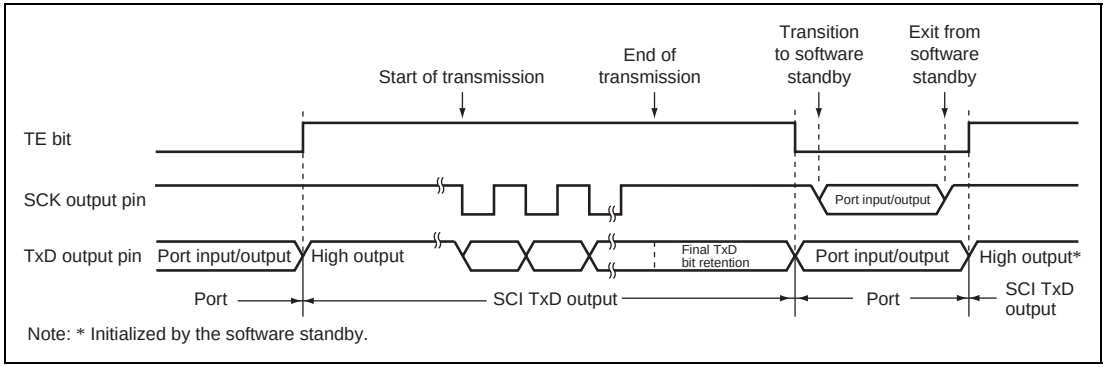



**Figure 12.41 Port Pin State of Synchronous Transmission Using Internal Clock**

- Reception

  Receive operation should be stopped (by clearing RE to 0) before making a module stop mode, software standby mode, watch mode, subactive mode, or subsleep mode transition. RSR, RDR, and SSR are reset. If a transition is made without stopping operation, the data being received will be invalid.

  To continue receiving without changing the reception mode after the relevant mode is cleared, set RE to 1 before starting reception. To receive with a different receive mode, the procedure must be started again from initialization.

  Figure 12.42 shows a sample flowchart for mode transition during reception.

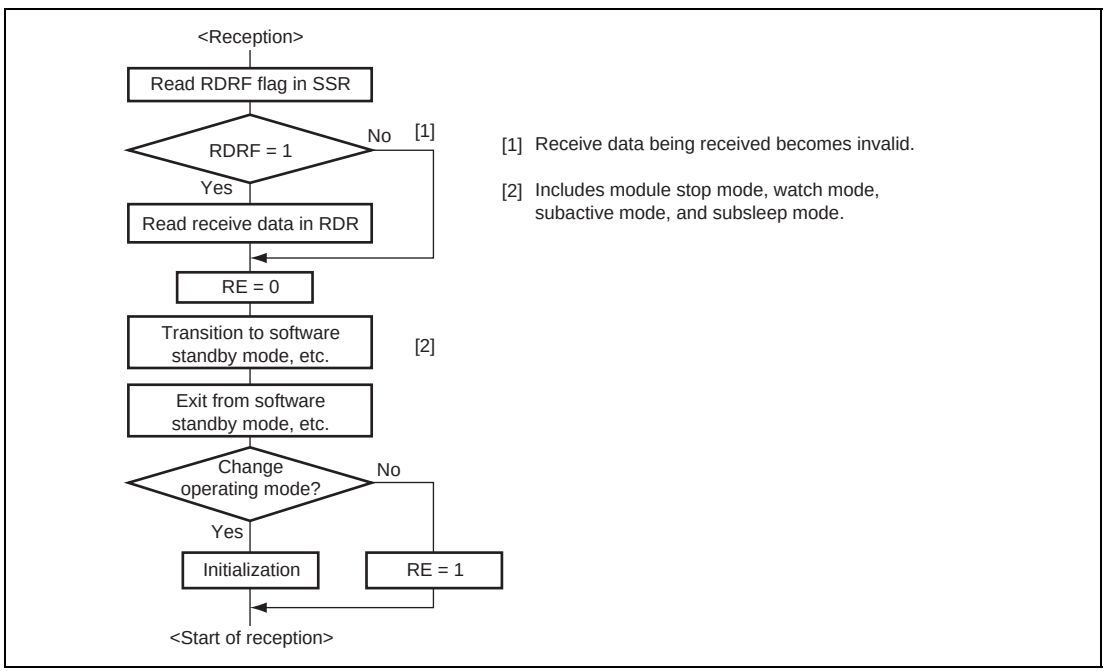



**Figure 12.42 Sample Flowchart for Mode Transition during Reception**

### 12.10.7 Switching from SCK Pin Function to Port Pin Function:

When switching the SCK pin function to the output port function (high-level output) by making the following settings while DDR = 1, DR = 1, C/$\overline{A}$ = 1, CKE1 = 0, CKE0 = 0, and TE = 1 (synchronous mode), low-level output occurs for one half-cycle.

1. End of serial data transmission
2. TE bit = 0
3. C/$\overline{A}$ bit = 0 ... switchover to port output
4. Occurrence of low-level output (see figure 12.43)

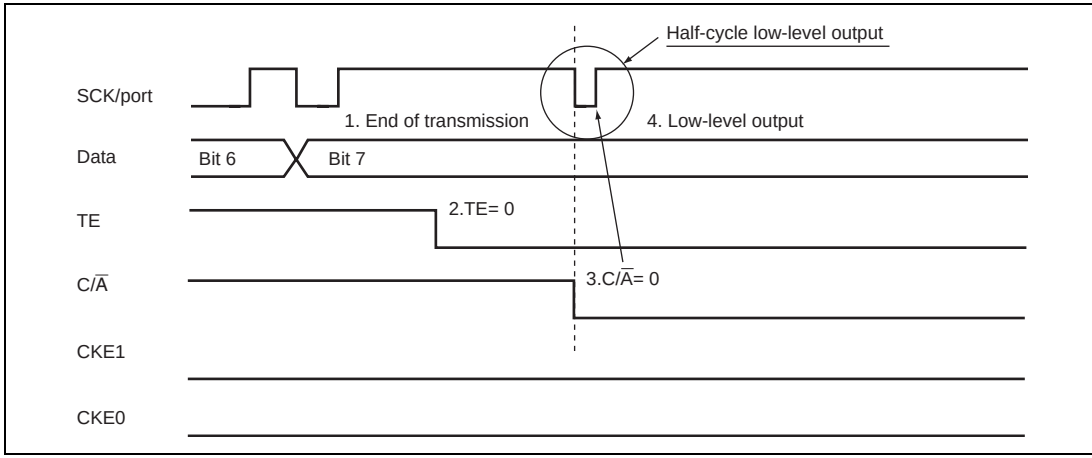



**Figure 12.43 Operation when Switching from SCK Pin Function to Port Pin Function**

**Sample Procedure for Avoiding Low-Level Output:** As this sample procedure temporarily places the SCK pin in the input state, the SCK/port pin should be pulled up beforehand with an external circuit.
With DDR = 1, DR = 1, C/$\overline{A}$ = 1, CKE1 = 0, CKE0 = 0, and TE = 1, make the following settings in the order shown.

1. End of serial data transmission
2. TE bit = 0
3. CKE1 bit = 1
4. C/$\overline{A}$ bit = 0 ... switchover to port output
5. CKE1 bit = 0

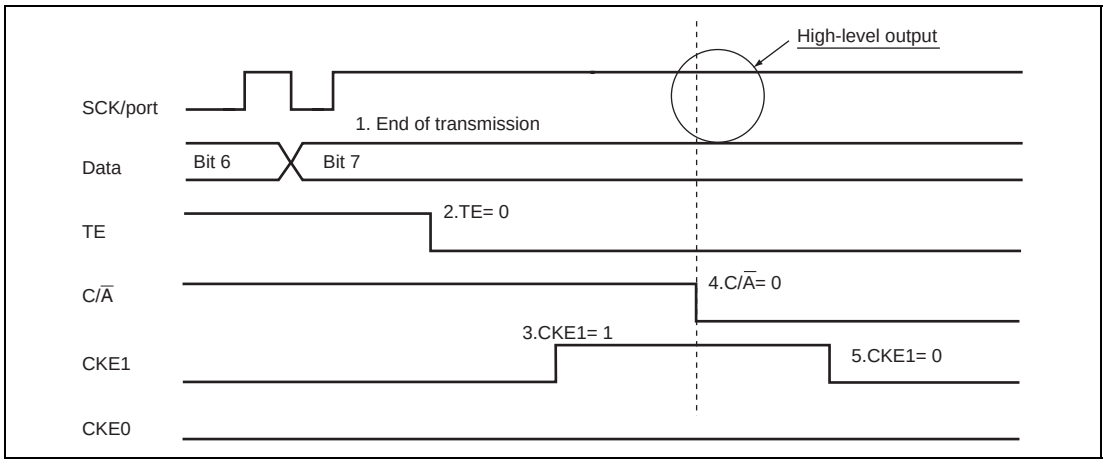


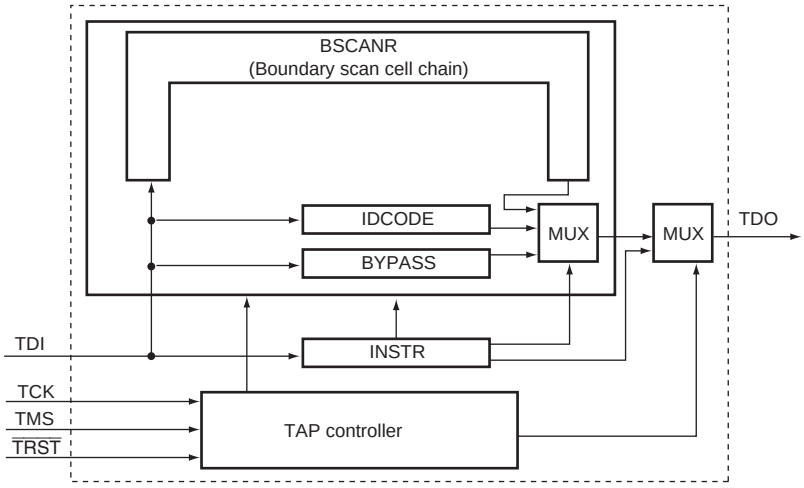

**Figure 12.44 Operation when Switching from SCK Pin Function to Port Pin Function (Example of Preventing Low-Level Output)**

# Section 13 Boundary Scan Function

The HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU incorporate a boundary scan function, which is a serial I/O interface based on the JTAG (Joint Test Action Group, IEEEStd.1149.1 and IEEE Standard Test Access Port and Boundary Scan Architecture). Figure 13.1 shows the block diagram of the boundary scan function.

## 13.1 Features

- Five test signals
  - TCK, TDI, TDO, TMS, $\overline{\text{TRST}}$
- Six test modes supported
  - BYAPASS, SAMPLE/PRELOAD, EXTEST, CLAMP, HIGHZ, IDCODE
- Boundary scan function cannot be performed on the following pins.
  - Power supply pins: VCC, VSS, Vref, PLLVCC, PLLVSS, DrVCC, DrVSS
  - Clock signals: EXTAL, XTAL, OSC2, OSC1
  - Analog signals: P40 to P43, P96, P97, USD+, USD-
  - Boundary scan signals: TCK, TDI, TDO, TMS, $\overline{\text{TRST}}$
  - H-UDI control signal: EMLE




Legend:
BSCANR: Boundary scan register
IDCODE: IDCODE register
BYPASS: BYPASS register
INSTR: Instruction register
TAP: Test access port

**Figure 13.1 Block Diagram of Boundary Scan Function**

## 13.2 Pin Configuration

Table 13.1 shows the I/O pins used in the boundary scan function.

**Table 13.1 Pin Configuration**

| Pin Name | I/O | Function |
|---|---|---|
| TMS | Input | Test Mode Select<br>Controls the TAP controller which is a 16-state Finite State Machine.<br>The TMS input value at the rising edge of TCK determines the status transition direction on the TAP controller.<br>The TMS is fixed high when the boundary scan function is not used.<br>The protocol is based on JTAG standard (IEEE Std.1149.1).<br>This pin has a pull-up resistor. |
| TCK | Input | Test Clock<br>A clock signal for the boundary scan function.<br>When the boundary scan function is used, input a clock of 50% duty to this pin.<br>This pin has a pull-up resistor. |
| TDI | Input | Test Data Input<br>A data input signal for the boundary scan function.<br>Data input from the TDI is latched at the rising edge of TCK.<br>TDI is fixed high when the boundary scan function is not used.<br>This pin has a pull-up register. |
| TDO | Output | Test Data Output<br>A data output signal for the boundary scan function. Data output from the TDO changes at the falling edge of TCK. The output driver of the TDO is driven only when it is necessary only in Shift-IR or Shift-DR states, and is brought to the high-impedance state when not necessary. |
| $\overline{\text{TRST}}$ | Input | Test Reset<br>Asynchronously resets the TAP controller when $\overline{\text{TRST}}$ is brought low.<br>The user must apply power-on reset signal specific to the boundary scan function when the power is supplied. (For details on signal design, refer to section 13.5, Usage Notes.)<br>This pin has a pull-up resister. |

## 13.3 Register Descriptions

The boundary scan function has the following registers. These registers cannot be accessed by the CPU.

- Instruction register (INSTR)
- IDCODE register (IDCODE)
- BYPASS register (BYPASS)
- Boundary scan register (BSCANR)

### 13.3.1 Instruction Register (INSTR)

INSTR is a 3-bit register. At initialization, this register is specified to IDCODE mode. When $\overline{\text{TRST}}$ is pulled low, or when the TAP controller is in the Test-Logic-Reset state, INSTR is initialized. INSTR can be written by the serial data input from the TDI. If more than three bits of instruction is input from the TDI, INSTR stores the last three bits of serial data.

If a command reserved in INSTR is used, the correct operation cannot be guaranteed.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 2 | TI2 | 1 | — | Test Instruction Bits |
| 1 | TI1 | 0 | — | Instruction configuration is shown in table 13.2. |
| 0 | TI0 | 1 | — | |

**Table 13.2 Instruction Configuration**

| Bit 2 | Bit1 | Bit 0 | |
|---|---|---|---|
| **TI2** | **TI1** | **TI0** | **Instruction** |
| 0 | 0 | 0 | EXTEST |
| 0 | 0 | 1 | SAMPLE/PRELOAD |
| 0 | 1 | 0 | CLAMP |
| 0 | 1 | 1 | HIGHZ |
| 1 | 0 | 0 | Reserved |
| 1 | 0 | 1 | IDCODE (initial value) |
| 1 | 1 | 0 | Reserved |
| 1 | 1 | 1 | BYPASS |

**EXTEST:** The EXTEST instruction is used to test external circuits when this LSI is installed on the print circuit board. If this instruction is executed, output pins are used to output test data (specified by the SAMPLE/PRELOAD instruction) from the boundary scan register to the print circuit board, and input pins are used to input test results.

**SAMPLE/PRELOAD:** The SAMPLE/PRELOAD instruction is used to input data from the LSI internal circuits to the boundary scan register, output data from scan path, and reload the data to the scan path. While this instruction is executed, input signals are directly input to the LSI and output signals are also directly output to the external circuits. The LSI system circuit is not affected by this instruction.

In SAMPLE operation, the boundary scan register latches the snap shot of data transferred from input pins to internal circuit or data transferred from internal circuit to output pins. The latched data is read from the scan path. The scan register latches the snap data at the rising edge of the TCK in Capture-DR state. The scan register latches snap shot without affecting the LSI normal operation.

In PRELOAD operation, initial value is written from the scan path to the parallel output latch of the boundary scan register prior to the EXTEST instruction execution. If the EXTEST is executed without executing this RELOAD operation, undefined values are output from the beginning to the end (transfer to the output latch) of the EXTEST sequence. (In EXTEST instruction, output parallel latches are always output to the output pins.)

**CLAMP:** When the CLAMP instruction is selected output pins output the boundary scan register value which was specified by the SAMPLE/PRELOAD instruction in advance. While the CLAMP instruction is selected, the status of boundary scan register is maintained regardless of the TAP controller state. BYPASS is connected between TDI and TDO, the same operation as BYPASS instruction can be achieved.

**HIGHZ:** When the HIGHZ instruction is selected, all outputs enter high-impedance state. While this instruction is selected, the status of boundary scan register is maintained regardless of the TAP controller state. BYPASS resistor is connected between TDI and TDO, the same operation as BYPASS instruction can be achieved.

**IDCODE:** When the IDCODE instruction is selected, IDCODE register value is output to the TDO in Shift-DR state of TAP controller. In this case, IDCODE register value is output from the LSB. During this instruction execution, test circuit does not affect the system circuit. INSTR is initialized by the IDCODE instruction in Test-Logic-Reset state of TAP controller.

**BYPASS:** The BYPASS instruction is a standard instruction necessary to operate bypass register. The BYPASS instruction improves the serial data transfer speed by bypassing the scan path. During this instruction execution, test circuit does not affect the system circuit.

### 13.3.2 IDCODE Register (IDCODE)

IDCODE register is a 32-bit register. If INSTR is set to IDCODE mode, IDCODE is connected between TDI and TDO. The HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU output fixed codes H'002A200F from the TDO. Serial data cannot be written to IDCODE register through TDI. Table 13.3 shows the IDCODE register configuration.

**Table 13.3 IDCODE Register Configuration**

| Bits | 31 to 28 | 27 to 12 | 11 to 1 | 0 |
|---|---|---|---|---|
| HD64F2218, HD64F2218U, HD64F2218CU and HD64F2217CU codes | 0000 | 0000 0010 1010 0010 | 0000 0000 111 | 1 |
| Contents | Version (4 bits) | Part No. (16 bits) | Product No. (11 bits) | Fixed code (1 bit) |

### 13.3.3 BYPASS Register (BYPASS)

BYPASS is a 1-bit register. If INSTR is specified to BYPASS mode, CLAMP mode, or HIGHZ mode, BYPASS is connected between TDI and TDO.

### 13.3.4 Boundary Scan Register (BSCANR)

BSCAN is a 199-bit shift register assigned on the pins to control input/output pins.

The I/O pins consists of three bits (IN, Control, OUT), input pins 1 bit (IN), and output pins 1 bit (OUT) of shift registers. The boundary scan test based on the JTAG standard can be performed by using instructions listed in table 13.2. Table 13.4 shows the correspondence between the LSI pins and boundary scan registers. (In table 13.4, Control indicates the high active pin. By specifying Control to high, the pin is driven by OUT. ) Figure 13.2 shows the boundary scan register configuration example.

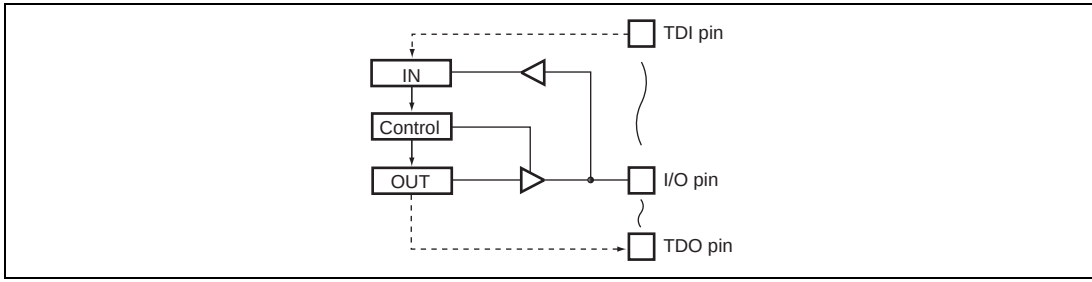



**Figure 13.2 Boundary Scan Register Configuration**

**Table 13.4 Correspondence between LSI Pins and Boundary Scan Register**

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| | | From TDI | | |
| 89 | A6 | PF7/$\phi$ | IN | 198 |
| | | | Control | 197 |
| | | | OUT | 196 |
| 91 | D6 | PF6/$\overline{AS}$ | IN | 195 |
| | | | Control | 194 |
| | | | OUT | 193 |
| 92 | A5 | PF5/$\overline{RD}$ | IN | 192 |
| | | | Control | 191 |
| | | | OUT | 190 |
| 93 | B5 | PF4/$\overline{HWR}$ | IN | 189 |
| | | | Control | 188 |
| | | | OUT | 187 |
| 94 | C5 | PF3/$\overline{LWR}$/$\overline{ADTRG}$/$\overline{IRQ3}$ | IN | 186 |
| | | | Control | 185 |
| | | | OUT | 184 |
| 95 | A4 | PF2/$\overline{WAIT}$ | IN | 183 |
| | | | Control | 182 |
| | | | OUT | 181 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 96 | D5 | PF1/$\overline{\text{BACK}}$ | IN | 180 |
| | | | Control | 179 |
| | | | OUT | 178 |
| 97 | B4 | PF0/$\overline{\text{BREQ}}$/$\overline{\text{IRQ2}}$ | IN | 177 |
| | | | Control | 176 |
| | | | OUT | 175 |
| 98 | A3 | PA3/A19/SCK2 | IN | 174 |
| | | | Control | 173 |
| | | | OUT | 172 |
| 99 | C4 | PA2/A18/RxD2 | IN | 171 |
| | | | Control | 170 |
| | | | OUT | 169 |
| 100 | B3 | PA1/A17/TxD2 | IN | 168 |
| | | | Control | 167 |
| | | | OUT | 166 |
| 1 | B2 | PA0/A16 | IN | 165 |
| | | | Control | 164 |
| | | | OUT | 163 |
| 2 | B1 | P10/TIOCA0/A20 | IN | 162 |
| | | | Control | 161 |
| | | | OUT | 160 |
| 3 | D4 | P11/TIOCB0/A21 | IN | 159 |
| | | | Control | 158 |
| | | | OUT | 157 |
| 4 | C2 | P12/TIOCC0/TCLKA/A22 | IN | 156 |
| | | | Control | 155 |
| | | | OUT | 154 |
| 5 | C1 | P13/TIOCD0/TCLKB/A23 | IN | 153 |
| | | | Control | 152 |
| | | | OUT | 151 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 6 | D3 | P14/TIOCA1/$\overline{\text{IRQ0}}$ | IN | 150 |
| | | | Control | 149 |
| | | | OUT | 148 |
| 7 | D2 | P15/TIOCB1/TCLKC | IN | 147 |
| | | | Control | 146 |
| | | | OUT | 145 |
| 8 | D1 | P16/TIOCA2/$\overline{\text{IRQ1}}$ | IN | 144 |
| | | | Control | 143 |
| | | | OUT | 142 |
| 9 | E4 | P17/TIOCB2/TCLKD | IN | 141 |
| | | | Control | 140 |
| | | | OUT | 139 |
| 10 | E3 | PC0/A0 | IN | 138 |
| | | | Control | 137 |
| | | | OUT | 136 |
| 11 | E1 | PC1/A1 | IN | 135 |
| | | | Control | 134 |
| | | | OUT | 133 |
| 12 | E2 | PC2/A2 | IN | 132 |
| | | | Control | 131 |
| | | | OUT | 130 |
| 13 | F3 | PC3/A3 | IN | 129 |
| | | | Control | 128 |
| | | | OUT | 127 |
| 14 | F1 | MD0 | IN | 126 |
| 15 | F2 | MD1 | IN | 125 |
| 16 | F4 | MD2 | IN | 124 |
| 17 | G1 | PC4/A4 | IN | 123 |
| | | | Control | 122 |
| | | | OUT | 121 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 18 | G2 | PC5/A5 | IN | 120 |
| | | | Control | 119 |
| | | | OUT | 118 |
| 19 | G3 | PC6/A6 | IN | 117 |
| | | | Control | 116 |
| | | | OUT | 115 |
| 20 | H1 | PC7/A7 | IN | 114 |
| | | | Control | 113 |
| | | | OUT | 112 |
| 21 | G4 | USPND/TMOW | OUT | 111 |
| 22 | H2 | P30/TxD0 | IN | 110 |
| | | | Control | 109 |
| | | | OUT | 108 |
| 23 | J1 | P31/RxD0 | IN | 107 |
| | | | Control | 106 |
| | | | OUT | 105 |
| 24 | H3 | P32/SCK0/$\overline{\text{IRQ4}}$ | IN | 104 |
| | | | Control | 103 |
| | | | OUT | 102 |
| 25 | J2 | PG1/$\overline{\text{CS3}}$/$\overline{\text{IRQ7}}$ | IN | 101 |
| | | | Control | 100 |
| | | | OUT | 99 |
| 26 | K2 | PG2/$\overline{\text{CS2}}$ | IN | 98 |
| | | | Control | 97 |
| | | | OUT | 96 |
| 27 | L2 | PG3/$\overline{\text{CS1}}$ | IN | 95 |
| | | | Control | 94 |
| | | | OUT | 93 |
| 28 | H4 | PG4/$\overline{\text{CS0}}$ | IN | 92 |
| | | | Control | 91 |
| | | | OUT | 90 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 29 | K3 | VBUS | IN | 89 |
| 30 | L3 | P36 | IN | 88 |
| | | | Control | 87 |
| | | | OUT | 86 |
| 37 | K5 | PB0/A8 | IN | 85 |
| | | | Control | 84 |
| | | | OUT | 83 |
| 38 | J6 | PB1/A9 | IN | 82 |
| | | | Control | 81 |
| | | | OUT | 80 |
| 39 | L6 | PB2/A10 | IN | 79 |
| | | | Control | 78 |
| | | | OUT | 77 |
| 40 | K6 | PB3/A11 | IN | 76 |
| | | | Control | 75 |
| | | | OUT | 74 |
| 47 | L9 | $\overline{\text{UBPM}}$ | IN | 73 |
| 49 | K9 | PB4/A12 | IN | 72 |
| | | | Control | 71 |
| | | | OUT | 70 |
| 50 | L10 | PB5/A13 | IN | 69 |
| | | | Control | 68 |
| | | | OUT | 67 |
| 51 | K10 | PB6/A14 | IN | 66 |
| | | | Control | 65 |
| | | | OUT | 64 |
| 52 | K11 | PB7/A15 | IN | 63 |
| | | | Control | 62 |
| | | | OUT | 61 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 55 | H9 | P74/$\overline{\text{MRES}}$ | IN | 60 |
| | | | Control | 59 |
| | | | OUT | 58 |
| 56 | H10 | P71/$\overline{\text{CS5}}$ | IN | 57 |
| | | | Control | 56 |
| | | | OUT | 55 |
| 57 | H11 | $\overline{\text{STBY}}$ | IN | 54 |
| 58 | G8 | $\overline{\text{RES}}$ | IN | 53 |
| 63 | F11 | P70/$\overline{\text{CS4}}$ | IN | 52 |
| | | | Control | 51 |
| | | | OUT | 50 |
| 64 | F10 | PE0/D0 | IN | 49 |
| | | | Control | 48 |
| | | | OUT | 47 |
| 65 | F8 | PE1/D1 | IN | 46 |
| | | | Control | 45 |
| | | | OUT | 44 |
| 66 | E11 | PE2/D2 | IN | 43 |
| | | | Control | 42 |
| | | | OUT | 41 |
| 67 | E10 | PE3/D3 | IN | 40 |
| | | | Control | 39 |
| | | | OUT | 38 |
| 68 | E9 | PE4/D4 | IN | 37 |
| | | | Control | 36 |
| | | | OUT | 35 |
| 69 | D11 | PE5/D5 | IN | 34 |
| | | | Control | 33 |
| | | | OUT | 32 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 70 | E8 | PE6/D6 | IN | 31 |
| | | | Control | 30 |
| | | | OUT | 29 |
| 71 | D10 | PE7/D7 | IN | 28 |
| | | | Control | 27 |
| | | | OUT | 26 |
| 72 | C11 | PD0/D8 | IN | 25 |
| | | | Control | 24 |
| | | | OUT | 23 |
| 73 | D9 | PD1/D9 | IN | 22 |
| | | | Control | 21 |
| | | | OUT | 20 |
| 74 | C10 | PD2/D10 | IN | 19 |
| | | | Control | 18 |
| | | | OUT | 17 |
| 75 | B11 | PD3/D11 | IN | 16 |
| | | | Control | 15 |
| | | | OUT | 14 |
| 76 | B10 | PD4/D12 | IN | 13 |
| | | | Control | 12 |
| | | | OUT | 11 |
| 77 | A10 | PD5/D13 | IN | 10 |
| | | | Control | 9 |
| | | | OUT | 8 |
| 78 | D8 | PD6/D14 | IN | 7 |
| | | | Control | 6 |
| | | | OUT | 5 |
| 79 | B9 | PD7/D15 | IN | 4 |
| | | | Control | 3 |
| | | | OUT | 2 |

| TFP-100G TFP-100GV Pin No. | BP-112 BP-112V Pin No. | Pin Name | I/O | Bit Name |
|---|---|---|---|---|
| 80 | A9 | FWE | IN | 1 |
| 81 | C8 | NMI | IN | 0 |
| | | to TDO | | |

## 13.4 Boundary Scan Function Operation

### 13.4.1 TAP Controller

Figure 13.3 shows the TAP controller status transition diagram, based on the JTAG standard.

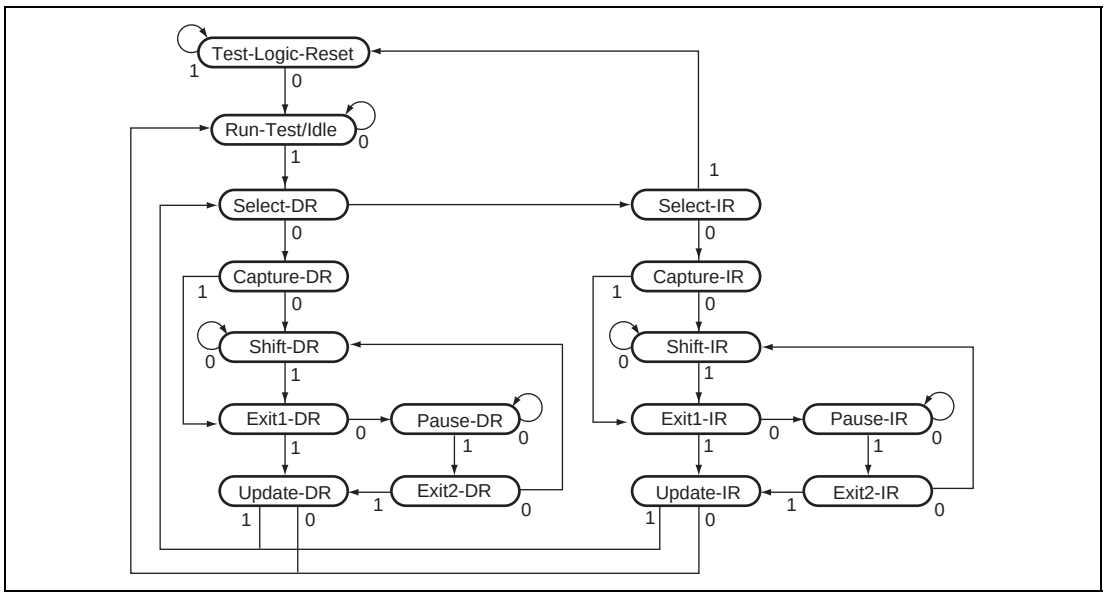



**Figure 13.3 TAP Controller Status Transition**

Note: The transition condition is the TMS value at the rising edge of TCK. The TDI value is sampled at the rising edge of the TCK and shifted at the falling edge of the TCK. The TDO value changes at the falling edge of the TCK. In addition, TDO is high-impedance state in a state other than Shift-DR or Shift-IR state. If $\overline{\text{TRST}}$ is 0, Test-Logic-Reset state is entered asynchronously with the TCK.

## 13.5 Usage Notes

1. When using the boundary scan function, clear $\overline{\text{TRST}}$ to 0 at power-on and after the $t_{RESW}$ time has elapsed set $\overline{\text{TRST}}$ to 1 and set TCK, TMS, and TDI appropriately. During normal operation when the boundary scan function is not used, set TCK, TMS, and TDI to Hi-Z, clear $\overline{\text{TRST}}$ to 0 at power-on, and after the $t_{RESW}$ time has elapsed set $\overline{\text{TRST}}$ to 1 or to Hi-Z. These pins are pulled up internally, so care must be taken in standby mode because breakthrough current flow can occur if there is a potential difference between the pin input voltage value when set to 1 and the power supply voltage Vcc.
2. The following must be noted on the power-on reset signal applied to the $\overline{\text{TRST}}$ pin.
   - Reset signal must be applied at power-on.
   - $\overline{\text{TRST}}$ must be separated in order not to affect the system operation.
   - $\overline{\text{TRST}}$ must be separated from the system circuitry in order not to affect the system operation.
   - System circuitry must also be separated from the $\overline{\text{TRST}}$ in order not to affect $\overline{\text{TRST}}$ operation as shown in figure 13.4.

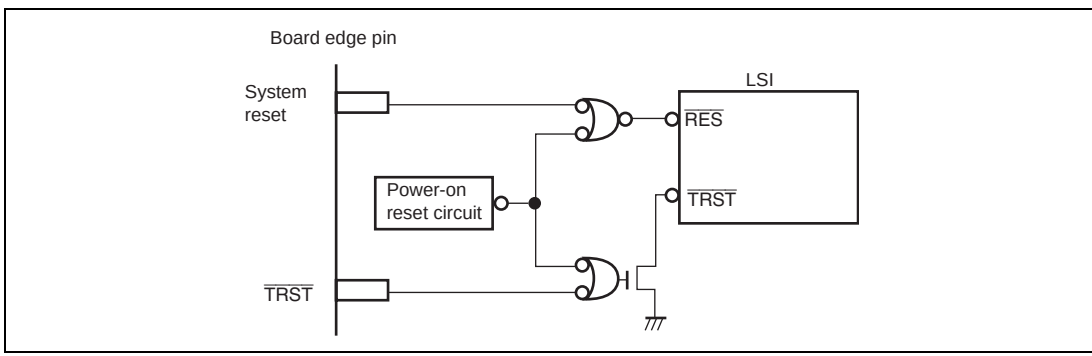



**Figure 13.4 Recommended Reset Signal Design**

3. TCK clock speed should be slower than system clock frequency.
4. In serial communication, data is input or output from the LSB as shown in figure 13.5.

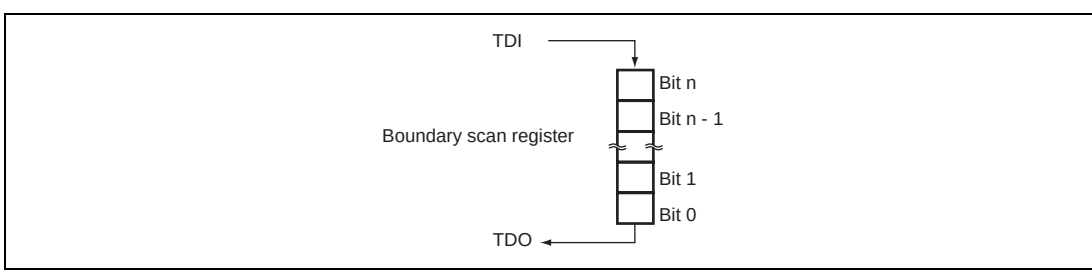



**Figure 13.5 Serial Data Input/Output**

5. If a pin with pull-up function is SAMPLEed with pull-up function enabled, the corresponding IN register is set to 1. In this case, the corresponding Control register must be cleared to 0.
6. If a pin with open-drain function is SAMPLEed while its open-drain function is enabled and while the corresponding OUT register is set to 1, the corresponding Control register is cleared to 0 (the pin status is Hi-Z). If the pin is SAMPLEed while the corresponding OUT register is cleared to 0, the corresponding Control register is set to 1 (the pin status is 0).
7. If EXTEST, CLAMP, or HIGHZ state is entered, this LSI enters guarded mode such as hardware standby mode ($\overline{RES}$ = $\overline{STBY}$ = 0). Before entering normal operating mode from EXTEST, CLAMP, or HIGHZ state, specify $\overline{RES}$, $\overline{STBY}$, FWE, and MD2 to MD0 pin to the designated mode.
8. The EMLE pin must be cleared to 0. When the pin is set to 1, this chip functions as High-performance user debugging interface (H-UDI).

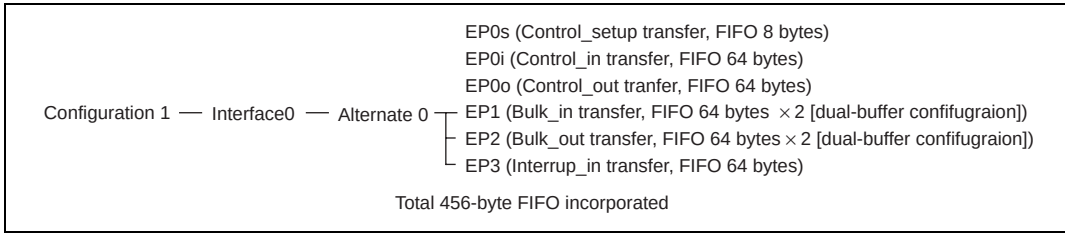
| EMLE Pin | Chip State |
|---|---|
| 0 | Normal operation, boundary scan function |
| 1 | High-performance user debugging interface (H-UDI) |

# Section 14 Universal Serial Bus (USB)

This LSI incorporates a USB function module complying with USB standard version 1.1. Figure 14.1 shows the block diagram of the USB.

## 14.1 Features

- USB standard version 2.0 full speed mode (12 Mbps) support
- Bus-powered mode or self-powered mode is selectable via the USB specific pin ($\overline{\text{UBPM}}$)
- On-chip PLL circuit to generate the USB operation clock (24 MHz × 2 = 48 MHz, 16 MHz × 3 = 48 MHz)
- On-chip bus transceiver
- Standard commands are processed automatically by hardware
  - Only Set_Descriptor, Get_Descriptor, Class/VendorCommand, and SynchFrame commands should be processed by software
- Current Configuration value can be checked by Set_Configuration interrupt
- Three transfer modes supported (Control, Bulk, Interrupt)
- Configuration of four endpoints; EP0, EP1, EP2, and EP3




- 16 kinds of interrupts
  - Suspend/resume interrupt source can be assigned for $\overline{\text{IRQ6}}$
  - Each interrupt source except the suspend/resume interrupt source can be assigned for $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ via registers
- DMA transfer interface
  - DMA transfer is enabled for the Bulk transfer data of EP1 and EP2
- 8-bit bus (3 cycle access timing) connected to the external bus interface
  - Internal registers are addressed to a part of area 6 of external address (H'C00000 to H'DFFFFF)
  - Address H'C00100 to H'DFFFFF is for USB reserved area and thus access prohibited.

Note: In this section, power-down mode represents watch, subactive, subsleep, and software standby modes.

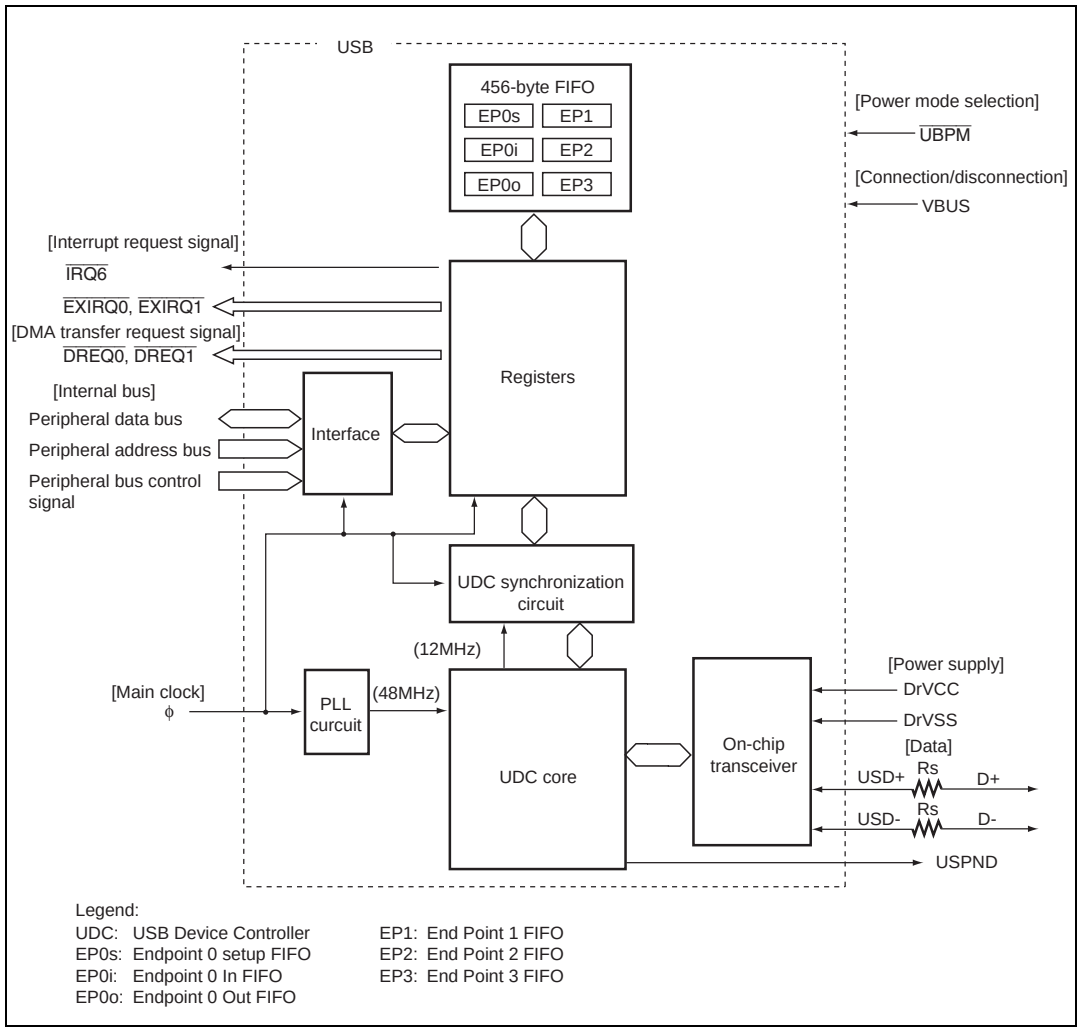



**Figure 14.1 Block Diagram of USB**

## 14.2 Input/Output Pins

Table 14.1 shows the USB pin configuration.

**Table 14.1 Pin Configuration**

| Pin Name | I/O | Function |
|---|---|---|
| USD+ | I/O | I/O pin for USB data |
| USD- | | |
| DrVCC | Input | USB internal transceiver power supply pin |
| DrVSS | Input | USB internal transceiver ground pin |
| VBUS | Input | USB cable connection/disconnection detection signal pin |
| $\overline{\text{UBPM}}$ | Input | USB bus-powered/self-powered mode set pin<br>When USB is used in bus-powered mode, $\overline{\text{UBPM}}$ must be fixed low.<br>When USB is used in self-powered mode, $\overline{\text{UBPM}}$ must be fixed high. |
| USPND | Output | USB suspend output pin<br>When USB enters the suspend state, USPND is set to high. |

## 14.3 Register Descriptions

The USB has the following registers.

- USB control register (UCTLR)
- USB DMAC transfer request register (UDMAR)
- USB device resume register (UDRR)
- USB trigger register 0 (UTRG0)
- USB FIFO clear register 0 (UFCLR0)
- USB endpoint stall register 0 (UESTL0)
- USB endpoint stall register 1 (UESTL1)
- USB endpoint data register 0s (UEDR0s) [for Setup data reception]
- USB endpoint data register 0i (UEDR0i) [for Control_in data transmission]
- USB endpoint data register 0o (UEDR0o) [for Control_out data reception]
- USB endpoint data register 3 (UEDR3) [for Interrupt_in data transmission]
- USB endpoint data register 1 (UEDR1) [for Bulk_in data transmission]
- USB endpoint data register 2 (UEDR2) [for Bulk_out data reception]
- USB endpoint receive data size register 0o (UESZ0o) [for Control _out data reception]
- USB endpoint receive data size register 2 (UESZ2) [for Bulk_out data reception]

- USB interrupt flag register 0 (UIFR0)
- USB interrupt flag register 1 (UIFR1)
- USB interrupt flag register 3 (UIFR3)
- USB interrupt enable register 0 (UIER0)
- USB interrupt enable register 1 (UIER1)
- USB interrupt enable register 3 (UIER3)
- USB interrupt select register 0 (UISR0)
- USB interrupt select register 1 (UISR1)
- USB interrupt select register 3 (UISR3)
- USB data status register (UDSR)
- USB Configuration value register (UCVR)
- USB test register 0 (UTSTR0)
- USB test register 1 (UTSTR1)
- USB test registers 2 and A to F (UTSTR2, UTSTRA to UTSTRF)
- Module stop control register B (MSTPCRB)
- Extended module stop register (EXMDLSTP)

### 14.3.1 USB Control Register (UCTLR)

UCTLR is used to select the USB operation clock and control the USB module internal reset. UCTLR can be read from or written to even when the USB module stop 2 bit (MSTPB0) in MSTPCRB is 1. For details on UCTLR setting procedure, refer to section 14.5, Communication Operation.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | 0 | R/W | Reserved<br>The write value should always be 0. |
| 6 | TMOWE | 0 | R/W | TMOW Pin Enable<br>0: The USPND/TMOW pin outputs USPND of USB.<br>1: The USPND/TMOW pin outputs TMOW of RTC. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 5 | UCKS3 | 0 | R/W | USB Operation Clock Select 3 to 0 |
| 4 | UCKS2 | | | These bits control the on-chip PLL, which generates the USB operation clock (48 MHz). When UCKS3 to UCKS0 are 0000, the PLL circuit stops and thus the USB operation clock must be selected according to the clock source. |
| 3 | UCKS1 | | | |
| 2 | UCKS0 | | | |
| | | | | The on-chip PLL circuit starts operating after the USB module stop 2 bit has been cancelled. In addition, the USB operation clock is supplied to the UDC core after the USB operating clock stabilization time has been passed. The completion timing of the USB operating clock stabilization time can be detected by the CK48READY flag in UIFR3. |
| | | | | UCKS0 to UCKS3 muse be written while the USB module stop 2 bit (MSTPB0) is 1. |
| | | | | 0000: USB operation clock stops (PLL stops) |
| | | | | 0001: Reserved |
| | | | | 001×: Reserved |
| | | | | 010×: Reserved |
| | | | | 0110: Uses a clock (48 MHz) generated by doubling the 24-MHz main oscillation by the PLL. |
| | | | | 0111: Uses a clock (48 MHz) generated by tripling the 16-MHz main oscillation by the PLL. |
| | | | | 1×××: Reserved |
| | | | | The USB operating clock stabilization time is 2 ms. |
| | | | | Legend: |
| | | | | ×: Don't care |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 1 | UIFRST | 1 | R/W | USB Interface Software Reset<br>Controls USB module internal reset. When the UIFRST bit is set to 1, the USB internal modules other than UCTLR, UIER3, and the CK48READY bit in UIFR3 are all reset. At initialization, the UIFRST bit must be cleared to 0 after the USB operating clock (48 MHz) stabilization time has passed following the clearing of the USB module stop 2 bit.<br>0: Sets the USB internal modules to the operating state. (At initialization, this bit must be cleared after the USB operating clock stabilization time has passed.)<br>1: Sets the USB internal modules other than UCTLR, UIER3, and the CK48READY bit in UIFR3 to the reset state.<br>If the UIFRST bit is set to 1 after it is cleared to 0, the UDCRST bit should also be set to 1 simultaneously. |
| 0 | UDCRST | 1 | R/W | UDC Core Software Reset<br>Controls reset of the UDC core in the USB module. When the UDCRST bit is set to 1, the UDC core is reset and the USB bus synchronization operation stops. At initialization, UDCRST must be cleared to 0 after D+ pull-up by the port (P36) control following the clearing of the UIFRST bit. In the suspend state, to maintain the internal state of the UDC core, enter power-down mode after setting the USB module stop 2 bit with the UDCRST bit to be maintained to 0. After VBUS disconnection detection, UDCRST must be set to 1.<br>0: Sets the UDC core in the USB module to operating state. (At initialization, UDCRST must be cleared to 0 after D+ pull-up by the port control following the clearing of the UIFRST bit.)<br>1: Sets the UDC core in the USB module to reset state. (In the suspend state, UDCRST must not be set to 1; after VBUS disconnection detection, UDCRST must be set to 1.) |

### 14.3.2 USB DMAC Transfer Request Register (UDMAR)

UDMAR is set when data transfer by means of a USB request of the on-chip DMAC is performed for data registers UEDR1 and UEDR2 corresponding to EP1 and EP2 respectively used for Bulk transfer. For the DMAC transfer, set $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ separately. If $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ usage overlaps, the USB cannot operate correctly. For details on DMAC transfer, refer to section 14.6, DMA Transfer Specifications.

Note: As the $\overline{\text{DREQ}}$ signal is not used in the data transfer by auto request of the on-chip DMAC, set UDMAR to H'00.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 3 | EP2T1 | 0 | R/W | EP2 DMAC Transfer Request Select 1, 0 |
| 2 | EP2T0 | | | 00: Does not request EP2 DMAC transfer |
| | | | | 01: Reserved |
| | | | | 10: Requests EP2 DMAC transfer by $\overline{\text{DREQ0}}$ |
| | | | | 11: Requests EP2 DMAC transfer by $\overline{\text{DREQ1}}$ |
| 1 | EP1T1 | 0 | R/W | EP1 DMAC Transfer Request Select 1, 0 |
| 0 | EP1T0 | | | 00: Does not request EP1 DMAC transfer |
| | | | | 01: Reserved |
| | | | | 10: Requests EP1 DMAC transfer by $\overline{\text{DREQ0}}$ |
| | | | | 11: Requests EP1 DMAC transfer by $\overline{\text{DREQ1}}$ |

### 14.3.3 USB Device Resume Register (UDRR)

UDRR indicates the enabled or disabled state of remote wakeup by the host, and executes the remote wakeup of the USB modules in the suspend state.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 2 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 1 | RWUPs | 0 | R | Remote Wakeup Status<br>Indicates the enabled or disabled state of remote wakeup by the host. This bit is a status bit and cannot be written to. If the remote wakeup from the host is disabled by Device_Remote_Wakeup through the Set_Feature/Clear _Feature request, this bit is cleared to 0. If the remote wakeup is enabled, this bit is set to 1.<br>0: Remote wakeup disabled state<br>1: Remote wakeup enabled state |
| 0 | DVR | 0 | W | Device Resume<br>Cancels the suspend state (executes the remote wakeup). This bit can be written to 1 and is always read as 0. Before executing the remote wakeup, power-down mode or USB module stop mode must be cancelled to provide a clock for the USB module.<br>0: Performs no operation<br>1: Cancels the suspend state (executes the remote wakeup) |

### 14.3.4 USB Trigger Register 0 (UTRG0)

UTRG0 is a one-shot register to generate triggers to the FIFO for each endpoint EP0 to EP3. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 5 | EP2RDFN | 0 | W | EP2 Read Complete<br>0: Performs no operation.<br>1: Writes 1 to this bit after reading data for EP2 OUT FIFO. EP2 has a dual-FIFO configuration. This trigger is generated to the currently effective FIFO. |
| 4 | EP1PKTE | 0 | W | EP1 Packet Enable<br>0: Performs no operation.<br>1: Generates a trigger to enable the transmission to EP1 IN FIFO. EP1 has a dual-FIFO configuration. This trigger is generated to the currently effective FIFO. |
| 3 | EP3PKTE | 0 | W | EP3 Packet Enable<br>0: Performs no operation.<br>1: Generates a trigger to enable the transmission to EP3 IN FIFO. |
| 2 | EP0oRDFN | 0 | W | EP0o Read Complete<br>0: Performs no operation.<br>1: Writes 1 to this bit after reading data for EP0o OUT FIFO. This trigger enables EP0o to receive the next packet. |
| 1 | EP0iPKTE | 0 | W | EP0i Packet Enable<br>0: Performs no operation.<br>1: Generates a trigger to enable the transmission to EP0i IN FIFO. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 0 | EP0sRDFN | 0 | W | EP0s Read Complete<br>0: Performs no operation. A NAK handshake is returned in response to transmit/receive requests in the data stage until 1 is written to this bit.<br>1: Writes 1 to this bit after reading data for EP0s command FIFO. After receiving the setup command, this trigger enables the next packet in the data stage to be received by EP0i and EP0o. EP0s can always be overwritten and receive data regardless of this trigger. |

### 14.3.5 USB FIFO Clear Register 0 (UFCLR0)

UFCLR0 is a one-shot register used to clear the FIFO for each endpoint EP0 to EP3. Writing 1 to a bit clears the data in the corresponding FIFO.

For IN FIFO, writing 1 to a bit in UFCLR0 clears the data for which the corresponding PKTE bit in UTRG0 is not set to 1 after data write, or data that is validated by setting the corresponding PKTE bit in UTRG0.

For OUT FIFO, writing 1 to a bit in UFCLR0 clears data that has not been fixed during reception or received data for which the corresponding RDFN bit is not set to 1. Accordingly, care must be taken not to clear data that is currently being received or transmitted. EP1 and EP2, having a dual-FIFO configuration, are cleared by entire FIFOs. Note that this trigger does not clear the corresponding interrupt flag. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 5 | EP2CLR | 0 | W | EP2 Clear*<br>0: Performs no operation.<br>1: Clears EP2 OUT FIFO. |
| 4 | EP1CLR | 0 | W | EP1 Clear<br>0: Performs no operation.<br>1: Clears EP1 IN FIFO. |
| 3 | EP3CLR | 0 | W | EP3 Clear<br>0: Performs no operation.<br>1: Clears EP3 IN FIFO. |
| 2 | EP0oCLR | 0 | W | EP0o Clear<br>0: Performs no operation.<br>1: Clears EP0o OUT FIFO. |
| 1 | EP0iCLR | 0 | W | EP0i Clear<br>0: Performs no operation.<br>1: Clears EP0i IN FIFO. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 0 | — | 0 | R | Reserved<br>This bit is always read as 0 and cannot be modified. |

Note:* When DMA writes are enabled (EP2T1 set to 1 and EP2T0 set to 0 or 1 in UDMAR), it is not possible to clear the data in the FIFO by writing 1 to EP2CLR. To clear the data in the FIFO, disable DMA transfers (clear EP2T1 and EP2T0 in UDMAR to 0) and then write 1 to EP2CLR.

### 14.3.6 USB Endpoint Stall Register 0 (UESTL0)

UESTL0 is used to forcibly stall each endpoint EP0 to EP3. When the bit is set to 1, the corresponding endpoint returns a stall handshake to the host, following from the next transfer.

The stall bit for endpoint 0 is cleared automatically on reception of 8-byte command data for which decoding is performed by the function, and thus the EP0STL bit is cleared to 0. When the SetupTS flag in UIFR0 is set to 1, a write of 1 to the EP0STL bit is ignored. For details, refer to section 14.5.9, Stall Operations.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 5 | EP2STL | 0 | R/W | EP2 Stall<br>0: Cancels the EP2 stall state.<br>1: Sets the EP2 stall state. |
| 4 | EP1STL | 0 | R/W | EP1 Stall<br>0: Cancels the EP1 stall state.<br>1: Sets the EP1 stall state. |
| 3 | EP3STL | 0 | R/W | EP3 Stall<br>0: Cancels the EP3 stall state.<br>1: Sets the EP3 stall state. |
| 2, 1 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 0 | EP0STL | 0 | R/W | EP0 Stall<br>0: Cancels the EP0 stall state.<br>1: Sets the EP0 stall state. |

### 14.3.7 USB Endpoint Stall Register 1 (UESTL1)

UESTL1 is used to control stall cancellation mode for all endpoints.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | SCME | 0 | R/W | Reserved<br>The write value should always be 0. |
| 6 to 0 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |

### 14.3.8 USB Endpoint Data Register 0s (UEDR0s)

UEDR0s stores the setup command for endpoint 0 (for Control_out transfer). UEDR0s stores 8-byte command data sent from the host in setup stage.

For details on the USB operation when data for the next setup stage is received while data in UEDR0s is being read, refer to section 14.8, Usage Notes.

UEDR0s is a byte register to which 4-byte address area is assigned. Accordingly, UEDR0s allows the user to read 2-byte or 4-byte data continuously by word transfer or longword transfer.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 0 | D7 to D0 | — | R | These bits store the setup command for Control_out transfer |

### 14.3.9 USB Endpoint Data Register 0i (UEDR0i)

UEDR0i is a data register for endpoint 0 (for Control_in transfer). UEDR0i stores data to be sent to the host. The number of data items to be written continuously must be the maximum packet size or less.

UEDR0i is a byte register to which 4-byte address area is assigned. Accordingly, UEDR0i allows the user to write 2-byte or 4-byte data continuously by word transfer or longword transfer. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 0 | D7 to D0 | All 0 | W | These bits store data for Control_in transfer |

### 14.3.10 USB Endpoint Data Register 0o (UEDR0o)

UEDR0o is a data register for endpoint 0 (for Control_out transfer). UEDR0o stores data received from the host. The number of data items to be read must be the number of bytes specified by UESZ0o.

When 1 byte is read from UEDR0o, UESZ0o is decremented by1.

UEDR0o is a byte register to which 4-byte address area is assigned. Accordingly, UEDR0o allows the user to read 2-byte or 4-byte data continuously by word transfer or longword transfer.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 0 | D7 to D0 | — | R | These bits store data for Control_out transfer |

### 14.3.11 USB Endpoint Data Register 3 (UEDR3)

UEDR3 is a data register for endpoint 3 (for Interrupt_in transfer). UEDR3 stores data to be sent to the host. The number of data items to be written continuously must be the maximum packet size or less.

UEDR3 is a byte register to which 4-byte address area is assigned. Accordingly, UEDR3 allows the user to write 2-byte or 4-byte data continuously by word transfer or longword transfer. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 0 | D7 to D0 | All 0 | W | These bits store data for Interrupt_in transfer |

### 14.3.12 USB Endpoint Data Register 1 (UEDR1)

UEDR1 is a data register for endpoint 1 (for Bulk_in transfer). UEDR1 stores data to be sent to the host. The number of data items to be written continuously must be the maximum packet size or less.

UEDR1 is a byte register to which 4-byte address area is assigned. Accordingly, UEDR1 allows the user to write 2-byte or 4-byte data continuously by word transfer or longword transfer. For details, see section 2.9.4, Accessing Registers Containing Write-Only Bits.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 0 | D7 to D0 | All 0 | W | These bits store data for Bulk_in transfer |

### 14.3.13 USB Endpoint Data Register 2 (UEDR2)

UEDR2 is a data register for endpoint 2 (for Bulk_out transfer). UEDR2 stores data received from the host. The number of data items to be read must be the number of bytes specified by UESZ2.

When 1 byte is read from UEDR2, UESZ2 is decremented by1.

UEDR2 is a byte register to which 4-byte address area is assigned. Accordingly, UEDR2 allows the user to read 2-byte or 4-byte data continuously by word transfer or longword transfer.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 0 | D7 to D0 | — | R | These bits store data for Bulk_out transfer |

### 14.3.14 USB Endpoint Receive Data Size Register 0o (UESZ0o)

UESZ0o is a receive data size register for endpoint 0 (for Control_out transfer). UESZ0o indicates the number of bytes of data to be received from the host.

Note that UESZ0o is decremented by 1 every time when 1 byte is read from UEDR0o.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | — | R | Reserved |
| 6 to 0 | D6 to D0 | — | R | These bits indicate the size of data to be received in Control_out transfer |

### 14.3.15 USB Endpoint Receive Data Size Register 2 (UESZ2)

UESZ2 is a receive data size register for endpoint 2 (for Bulk_out transfer). UESZ2 indicates the number of bytes of data to be received from the host.

Note that UESZ2 is decremented by 1 every time when 1 byte is read from UEDR2.

The FIFO for endpoint 2 (for Bulk_out transfer) has a dual-FIFO configuration. The data size indicated by this register refers to the currently selected FIFO.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | — | — | R | Reserved |
| 6 to 0 | D6 to D0 | — | R | These bits indicate the size of data to be received in Bulk_out transfer |

### 14.3.16 USB Interrupt Flag Register 0 (UIFR0)

UIFR0 is an interrupt flag register indicating the setup command reception, EP0 and EP3 transmission/reception, and bus reset state. If the corresponding bit is set to 1, the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ interrupt is requested to the CPU. A bit in this register can be cleared by writing 0 to it. Writing 1 to a bit is invalid and causes no operation. Consequently, to clear a flag, write 0 to the corresponding bit and 1 to all the other bits. (For example, write H'DF to clear bit 5.) The bit-clear instruction is a read/modify/write instruction, so if a new flag is set between the read and write operations, there is a danger that it may be cleared erroneously. Therefore, do not use the bit-clear instruction to clear bits in this interrupt flag resister.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | BRST | 0 | R/(W)* | Bus Reset<br>Set to 1 when the bus reset signal is detected on the USB bus. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$.<br>Note that BRST is also set to 1 if D+ is not pulled-up during USB cable connection. |
| 6 | — | 0 | R | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 5 | EP3TR | 0 | R/(W)* | EP3 Transfer Request<br>Set to 1 if there is no valid data in the FIFO when an IN token is sent from the host to EP3. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 4 | EP3TS | 0 | R/(W)* | EP3 Transmit Complete<br>Set to 1 if the data written in EP3 is transmitted to the host normally and the ACK handshake is returned. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 3 | EP0oTS | 0 | R/(W)* | EP0o Receive Complete<br>Set to 1 if EP0o receives data from the host normally and returns the ACK handshake to the host. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 2 | EP0iTR | 0 | R/(W)* | EP0i Transfer Request<br>Set to 1 if there is no valid data in the FIFO when an IN token is sent from the host to EP0i. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 1 | EP0iTS | 0 | R/(W)* | EP0i Transmit Complete<br>Set to 1 if the data written in EP0i is transmitted to the host normally and the ACK handshake is returned. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 0 | SetupTS | 0 | R/(W)* | Setup Command Receive Complete<br>Set to 1 if EP0s normally receives 8-byte command data to be decoded by the function from the host and returns the ACK handshake to the host. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |

Note:* The write value should always be 0 to clear this flag.

### 14.3.17 USB Interrupt Flag Register 1 (UIFR1)

UIFR1 is an interrupt flag register indicating the EP1 and EP2 status. If the corresponding bit is set to 1, the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ interrupt is requested to the CPU. EP1TR flags can be cleared by writing 0 to them. Writing 1 to them is invalid and causes no operation. Consequently, to clear a flag, write 0 to the corresponding bit and 1 to all the other bits. (For example, write H'FD to clear bit 1.) The bit-clear instruction is a read/modify/write instruction, so if a new flag is set between the read and write operations, there is a danger that it may be cleared erroneously. Therefore, do not use the bit-clear instruction to clear bits in this interrupt flag resister. However, EP1EMPTY, EP2READY, and EP1ALLEMPTYs are status bits to indicate the EP1, EP2, and FIFO state respectively, and cannot be cleared.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 3 | EP1ALL EMPTYs | 1 | R | EP1 FIFO All Empty Status<br>EP1 FIFO has a dual-FIFO configuration. This bit is set to 1 if there is no valid data in both FIFOs. This corresponds to the negative-electrode signal for the EP1DE bit in UDSR.<br>An interrupt cannot be required by EP1ALLEMPTY. |
| 2 | EP2READY | 0 | R | EP2 Data Ready<br>EP2 FIFO has a dual-FIFO configuration. This bit is set to 1 if there is valid data at least in either of FIFOs. This bit is cleared to 0 if there is no valid data in both FIFOs. This bit is a status bit and cannot be cleared. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 1 | EP1TR | 0 | R/(W)* | EP1 Transfer Request<br>Set to 1 if there is no valid data in both FIFOs when an IN token is sent from the host to EP1. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 0 | EP1EMPTY | 1 | R | EP1 FIFO Empty<br>EP1 FIFO has a dual-FIFO configuration. This bit is set to 1 if there is no valid data at least in either of FIFOs. This bit is cleared to 0 if there is valid data in both FIFOs. This bit is a status bit and cannot be cleared. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |

Note:* The write value should always be 0 to clear this flag.

### 14.3.18 USB Interrupt Flag Register 3 (UIFR3)

UIFR3 is an interrupt flag register indicating the USB status. If the corresponding bit is set to 1, the corresponding $\overline{\text{EXIRQ0}}$, $\overline{\text{EXIRQ1}}$, or $\overline{\text{IRQ6}}$ interrupt is requested to the CPU. VBUSi, SPRSi, SETC, SOF, and CK48READY flags can be cleared by writing 0 to them. Writing 1 to them is invalid and causes no operation. Consequently, to clear a flag, write 0 to the corresponding bit and 1 to all the other bits. (For example, write H'DF to clear bit 5.) The bit-clear instruction is a read/modify/write instruction, so if a new flag is set between the read and write operations, there is a danger that it may be cleared erroneously. Therefore, do not use the bit-clear instruction to clear bits in this interrupt flag resister. VBUSs and SPRSs are status bits and cannot be cleared.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | CK48READY | 0 | R/(W)* | USB Operating Clock (48 MHz) Stabilization Detection<br>Set to 1 when the USB operating clock (48 MHz) stabilization time has been automatically counted after USB module stop mode cancellation. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$.<br>CK48READY can also operate in the USB interface software reset state (the UIFRST bit in UCTLR is set to 1).<br>Refer to the UCKS3 to UCKS0 bits in section 14.3.1, USB Control Register (UCTLR). |
| 6 | SOF | 0 | R/(W)* | Start of Frame Packet Detection<br>Set to 1 if the Start of Frame (SOF) packet is detected. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 5 | SETC | 0 | R/(W)* | Set_Configuration Command Detection<br>Set to 1 if the Set_Configuration command is detected. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |
| 4 | — | 0 | R | Reserved<br>This bit is always read as 0 and cannot be modified. |
| 3 | SPRSs | 0 | R | Suspend/Resume Status<br>SPRSs indicates the suspend/resume status. However, an interrupt cannot be requested by SPRSs.<br>0: Indicates that the bus is in the normal state.<br>1: Indicates that the bus is in the suspend state. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 2 | SPRSi | 0 | R/(W)* | Suspend/Resume Interrupt<br>Set to 1 if a transition from normal state to suspend state or suspend state to normal state has occurred. The corresponding interrupt output is $\overline{\text{IRQ6}}$. This bit can be used to cancel power-down mode at resuming. |
| 1 | VBUSs | 0 | R | VBUS Status<br>VBUSs is a status bit to indicate the VBUS state by the USB cable connection or disconnection. However, an interrupt cannot be requested by VBUSs.<br>0: Indicates that the VBUS (USB cable) bus is disconnected.<br>1: Indicates that the VBUS (USB cable) bus is connected. |
| 0 | VBUSi | 0 | R/(W)* | VBUS Interrupt<br>Set to 1 if a VBUS state changes by the USB cable connection or disconnection. The corresponding interrupt output is $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. |

Note:* The write value should always be 0 to clear this flag.

### 14.3.19 USB Interrupt Enable Register 0 (UIER0)

UIER0 enables the interrupt request indicated in the interrupt flag register 0 (UIFR0). When an interrupt flag is set while the corresponding bit in UIER0 is set to 1, an interrupt is requested by asserting the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. Either $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ must be selected by the interrupt select register 0 (UISR0).

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | BRSTE | 0 | R/W | Enables the BRST interrupt. |
| 6 | — | 0 | R | Reserved<br>This bit is always read as 0. |
| 5 | EP3TRE | 0 | R/W | Enables the EP3TR interrupt. |
| 4 | EP3TSE | 0 | R/W | Enables the EP3TS interrupt. |
| 3 | EP0oTSE | 0 | R/W | Enables the EP0oTS interrupt. |
| 2 | EP0iTRE | 0 | R/W | Enables the EP0iTR interrupt. |
| 1 | EP0iTSE | 0 | R/W | Enables the EP0iTS interrupt. |
| 0 | SetupTSE | 0 | R/W | Enables the SetupTS interrupt. |

### 14.3.20 USB Interrupt Enable Register 1 (UIER1)

UIER1 enables the interrupt request indicated in the interrupt flag register 1 (UIFR1). When an interrupt flag is set while the corresponding bit in UIER1 is set to 1, an interrupt is requested by asserting the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. Either $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ must be selected by the interrupt select register 1 (UISR1).

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R | Reserved<br>These bits are always read as 0. |
| 3 | — | 0 | R/W | Reserved<br>The write value should always be 0. |
| 2 | EP2READYE | 0 | R/W | Enables the EP2READY interrupt. |
| 1 | EP1TRE | 0 | R/W | Enables the EP1TR interrupt. |
| 0 | EP1EMPTYE | 0 | R/W | Enables the EP1EMPTYE interrupt. |

### 14.3.21 USB Interrupt Enable Register 3 (UIER3)

UIER3 enables the interrupt request indicated in the interrupt flag register 3 (UIFR3). This register is readable/writable while the USB module stop 2 bit (MSTPB0) in MSTPCRB is 1.

When an interrupt flag is set while the corresponding bit in UIER3 is set to 1, an interrupt is requested by asserting the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. Either $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ must be selected by the interrupt select register 3 (UISR3). Note, however, that the SPRSiE bit is an interrupt enable bit specific to the $\overline{\text{IRQ6}}$ pin and cannot be selected by UISR3.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | CK48READYE | 1 | R/W | Enables the CK48READY interrupt. |
| 6 | SOFE | 0 | R/W | Enables the SOF interrupt. |
| 5 | SETCE | 0 | R/W | Enables the SETC interrupt. |
| 4, 3 | — | All 0 | R | Reserved<br>These bits are always read as 0. |
| 2 | SPRSiE | 0 | R/W | Enables the SPRSi interrupt. (only for $\overline{\text{IRQ6}}$) |
| 1 | — | 0 | R | Reserved<br>This bit is always read as 0. |
| 0 | VBUSiE | 0 | R/W | Enables the VBUSi interrupt. |

### 14.3.22 USB Interrupt Select Register 0 (UISR0)

UISR0 sets $\overline{\text{EXIRQ}}$ to output interrupt request indicated in the interrupt flag register 0 (UIFR0). When a bit in UIER0 corresponding to the UISR0 bit is cleared to 0, an interrupt request is output from $\overline{\text{EXIRQ0}}$. When a bit in UIER0 corresponding to the UISR0 bit is set to 1, an interrupt request is output from $\overline{\text{EXIRQ1}}$.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | BRSTS | 0 | R/W | Selects the BRST interrupt. |
| 6 | — | 0 | R | Reserved<br>This bit is always read as 0. |
| 5 | EP3TRS | 0 | R/W | Selects the EP3TR interrupt. |
| 4 | EP3TSS | 0 | R/W | Selects the EP3TS interrupt |
| 3 | EP0oTSS | 0 | R/W | Selects the EP0oTS interrupt. |
| 2 | EP0iTRS | 0 | R/W | Selects the EP0iTR interrupt. |
| 1 | EP0iTSS | 0 | R/W | Selects the EP0iTS interrupt. |
| 0 | SetupTSS | 0 | R/W | Selects the SetupTS interrupt. |

### 14.3.23 USB Interrupt Select Register 1 (UISR1)

UISR1 sets $\overline{\text{EXIRQ}}$ to output interrupt request indicated in the interrupt flag register 1 (UIFR1). When a bit in UIER1 corresponding to the UISR1 bit is cleared to 0, an interrupt request is output from $\overline{\text{EXIRQ0}}$. When a bit in UIER1 corresponding to the UISR1 bit is set to 1, an interrupt request is output from $\overline{\text{EXIRQ1}}$.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R | Reserved<br>These bits are always read as 0. |
| 3 | — | 0 | R/W | Reserved<br>The write value should always be 0. |
| 2 | EP2READYS | 0 | R/W | Selects the EP2READY interrupt. |
| 1 | EP1TRS | 0 | R/W | Selects the EP1TR interrupt. |
| 0 | EP1EMPTYS | 0 | R/W | Selects the EP1EMPTY interrupt. |

### 14.3.24 USB Interrupt Select Register 3 (UISR3)

UISR3 sets $\overline{\text{EXIRQ}}$ to output interrupt request indicated in the interrupt flag register 3 (UIFR3). When a bit in UIER3 corresponding to the UISR3 bit is cleared to 0, an interrupt request is output from $\overline{\text{EXIRQ0}}$. When a bit in UIER3 corresponding to the UISR3 bit is set to 1, an interrupt request is output from $\overline{\text{EXIRQ1}}$.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | CK48READYS | 0 | R/W | Selects the CK48READY interrupt. |
| 6 | SOFS | 0 | R/W | Selects the SOF interrupt. |
| 5 | SETCS | 0 | R/W | Selects the SETC interrupt. |
| 4 to 1 | — | All 0 | R | Reserved<br>These bits are always read as 0. |
| 0 | VBUSiS | 0 | R/W | Selects the VBUSi interrupt. |

### 14.3.25 USB Data Status Register (UDSR)

UDSR indicates whether the IN FIFO data registers (EP1, and EP3) contain valid data or not. A bit in USDR is set when data written to the corresponding IN FIFO becomes valid after the corresponding PKTE bit in UTRG is set to 1. A bit in USDR is cleared when all valid data is sent to the host. For EP1, having a dual-FIFO configuration, the corresponding bit in USDR is cleared to 0 and FIFO becomes empty.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 3 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 2 | EP1DE | 0 | R | EP1 Data Enable<br>0: Indicates that the EP1 contains no valid data.<br>1: Indicates that the EP1 contains valid data.<br>EP1DE corresponds to the negative-electrode signal for EP1ALLEMPTYs in UIFR1. |
| 1 | EP3DE | 0 | R | EP3 Data Enable<br>0: Indicates that the EP3 contains no valid data.<br>1: Indicates that the EP3 contains valid data. |
| 0 | EP0iDE | 0 | R | EP0i Data Enable<br>0: Indicates that the EP0i contains no valid data.<br>1: Indicates that the EP0i contains valid data. |

### 14.3.26 USB Configuration Value Register (UCVR)

UCVR stores the Configuration value when the Set_Configuration command is received from the host.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 5 | CNFV0 | 0 | R | Configuration Value 0<br>Stores the Configuration value when the Set_Configuration command is received. CNFV0 is modified when the SETC bit in UIFR3 is set to 1. |
| 4 to 0 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |

### 14.3.27 USB Test Register 0 (UTSTR0)

UTSTR0 controls the on-chip transceiver output signals. Setting the PTSTE bit to 1 after setting UIFRST and UDCRST in UCTLR to 0 specifies the transceiver output signals (USD+ and USD-) arbitrarily. Table 14.2 shows the relationship between UTSTR0 setting and pin output.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PTSTE | 0 | R/W | Pin Test Enable<br>Enables the test control for the on-chip transceiver output pins (USD+ and USD-) and USPND pin. |
| 6 to 4 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 3 | SUSPEND | 0 | R/W | On-Chip Transceiver Output Signal Setting<br>SUSPEND: Sets the USPND pin signal of the on-chip transceiver.<br>$\overline{OE}$: Sets the output enable ($\overline{OE}$) signal of the on-chip transceiver.<br>FSE0: Sets the single-ended 0 (FSE0) signal of the on-chip transceiver.<br>VPO: Sets the USD+ (VPO) signal of the on-chip transceiver. |
| 2 | $\overline{OE}$ | 1 | R/W | |
| 1 | FSE0 | 0 | R/W | |
| 0 | VPO | 0 | R/W | |

**Table 14.2 Relationship between UTSTR0 Setting and Pin Output**

| Register Setting | | | Pin Output | Pin Input | Register Setting | | | | Pin Output | |
|---|---|---|---|---|---|---|---|---|---|---|
| UCTLR/ TMOWE | PTSTE | SUSPEND | USPND/ TMOW | VBUS | PTSTE | $\overline{OE}$ | FSE0 | VPO | USD+ | USD- |
| 1 | × | × | — | 0 | × | × | × | × | Hi-Z | Hi-Z |
| 0 | 0 | × | — | 1 | 0 | × | × | × | — | — |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| | | | | 1 | 1 | 0 | 1 | × | 0 | 0 |
| | | | | 1 | 1 | 1 | × | × | Hi-Z | Hi-Z |

Legend:

×: Don't care

—: Cannot be controlled. Indicates state in normal operation according to the USB operation and port settings.

### 14.3.28 USB Test Register 1 (UTSTR1)

UTSTR1 allows the USB control pin and on-chip transceiver input signals to be monitored. Table 14.3 shows the relationship between pin input and UTSTR1 monitoring value.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | VBUS | —* | R | On-Chip Transceiver Input Signal Monitor |
| 6 | $\overline{UBPM}$ | —* | R | VBUS: Monitors the VBUS pin.<br>$\overline{UBPM}$: Monitors the $\overline{UBPM}$ pin. |
| 5 to 3 | — | All 0 | R | Reserved<br>These bits are always read as 0 and cannot be modified. |
| 2 | RCV | —* | R | On-Chip Transceiver Input Signal Monitor |
| 1 | VP | —* | R | RCV: Monitors the differential input level (RCV) signal of the on-chip transceiver. |
| 0 | VM | —* | R | VP: Monitors the USD+ (VP) signal of the on-chip transceiver.<br>VM: Monitors the USD- (VM) signal of the on-chip transceiver. |

Note: * Determined by the state of pins. VBUS, $\overline{UBPM}$, USD+, USD-

**Table 14.3 Relationship between Pin Input and UTSTR1 Monitoring Value**

| Pin Input | | UTSTR1 Monitoring Value | | Register Setting | | Pin Input | | | UTSTR1 Monitoring Value | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| VBUS | $\overline{\text{UBPM}}$ | VBUS | $\overline{\text{UBPM}}$ | UTSTR0/ PTSTE | UTSTR0/ SUSPEND | VBUS | USD+ | USD- | RCV | VP | VM |
| 0/1 | × | 0/1 | × | × | × | 0 | × | × | 0 | 0 | 0 |
| × | 0/1 | × | 0/1 | 0 | × | 1 | 0 | 0 | × | 0 | 0 |
| | | | | 0 | × | 1 | 0 | 1 | 0 | 0 | 1 |
| | | | | 0 | × | 1 | 1 | 0 | 1 | 1 | 0 |
| | | | | 0 | × | 1 | 1 | 1 | × | 1 | 1 |
| | | | | 1 | 0 | 1 | 0 | 0 | × | 0 | 0 |
| | | | | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| | | | | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| | | | | 1 | 0 | 1 | 1 | 1 | × | 1 | 1 |
| | | | | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| | | | | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| | | | | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| | | | | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

Legend:

×: Don’t care

0/1: Combination for pin input = UTSTR1 monitoring value.

### 14.3.29 USB Test Registers 2 and A to F (UTSTR2, UTSTRA to UTSTRF)

UTSTR2 and UTSRTA to UTSRTF are test registers and cannot be written to.

### 14.3.30 Module Stop Control Register B (MSTPCRB)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | MSTPB7 | 1 | R/W | Module Stop |
| 6 | MSTPB6 | 1 | | For details, refer to section 20.1.3, Module Stop Control Registers A to C (MSTPCRA to MSTPCRC). |
| 5 | MSTPB5 | 1 | | |
| 4 | MSTPB4 | 1 | | |
| 3 | MSTPB3 | 1 | | |
| 2 | MSTPB2 | 1 | | |
| 1 | MSTPB1 | 1 | | |
| 0 | MSTPB0 | 1 | R/W | USB Module Stop 2<br>0: Cancels the stop state of the USB module completely.<br>A clock is provided for the USB module completely. Before clearing this bit, make sure to clear the USBSTOP1 bit in EXMDLSTP. After this bit has been cleared, the internal PLL circuit starts operation. Registers in the USB module must be accessed after the USB operating clock stabilization time (the CK48READY bit in UIFR3 is set to 1) has passed.<br>1: Places the USB module partly in the stop state.<br>The internal PLL circuit and the most of the clocks in the USB module stop operation. However, register values in the USB module are maintained. |

### 14.3.31 Extended Module Stop Register (EXMDLSTP)

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 2 | — | Undefined | — | Reserved<br>These bits are always read as an undefined value and cannot be modified. |
| 1 | RTCSTOP | 0 | R/W | RTC Module Stop<br>0: Cancels the RTC module stop.<br>1: Sets the RTC module stop. |
| 0 | USBSTOP1 | 0 | R/W | USB Module Stop 1<br>0: Cancels the stop state of the USB module partly. A clock is provided for the USB module partly. After this bit has been cleared, only the UCTLR and UIER3 registers in the USB module can be accessed. To access the other registers, clear the MSTPB0 bit in MSTPCRB to 0.<br>1: Places the USB module completely in the stop state. The clocks in the USB module stop operation completely. However, register values in the USB module are maintained. |

Notes: 1. For details on USB module stop mode cancellation procedure, refer to section 14.5, Communication Operation.

2. When reading pin states using the port D register (PORTD), after accessing EXMDLSTP (address range: H'FFFF40 to H'FFFF5F), you must perform a dummy read to the external address space (such as H'FFEF00 to H'FF7FF) outside the range H'FFFF40 to H'FFFF5F before reading PORTD.

## 14.4 Interrupt Sources

This module has three interrupt signals. Table 14.4 shows the interrupt sources and their corresponding interrupt request signals. The $\overline{\text{EXIRQ}}$ interrupt signals are activated at low level. The $\overline{\text{EXIRQ}}$ interrupt requests can only be detected at low level (specified as level sensitive). The suspend/resume interrupt request $\overline{\text{IRQ6}}$ must be specified to be detected at the falling edge (falling-edge sensitive) by the interrupt controller register.

**Table 14.4 Interrupt Sources**

| Register | Bit | Transfer Mode | Interrupt Source | Description | Interrupt Request Signal | DMAC Activation by USB Request*[5] |
|---|---|---|---|---|---|---|
| UIFR0 | 0 | Control transfer (EP0) | SetupTS*[1] | Setup command receive complete | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 1 | | EP0iTS*[1] | EP0i transfer complete | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 2 | | EP0iTR*[1] | EP0i transfer request | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 3 | | EP0oTS*[1] | EP0o receive complete | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 4 | Interrupt_in transfer (EP3) | EP3TS | EP3 transfer complete | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 5 | | EP3TR | EP3 transfer request | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 6 | — | Reserved | — | — | — |
| | 7 | (Status) | BRST | Bus reset | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| UIFR1 | 0 | Bulk_in transfer (EP1) | EP1EMPTY | EP1 FIFO empty | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | $\overline{\text{DREQ0}}$ or $\overline{\text{DREQ1}}$*[2] |
| | 1 | | EP1TR | EP1 transfer request | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 2 | Bulk_out transfer (EP2) | EP2READY | EP2 data ready | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | $\overline{\text{DREQ0}}$ or $\overline{\text{DREQ1}}$*[3] |
| | 3 | Bulk_in transfer (EP1) | (EP1ALLEMPTYs) | EP1 FIFO all empty status | × | × |
| | 4 | — | Reserved | — | — | — |
| | 5 | | | | | |
| | 6 | | | | | |
| | 7 | | | | | |

| Register | Bit | Transfer Mode | Interrupt Source | Description | Interrupt Request Signal | DMAC Activation by USB Request*5 |
|---|---|---|---|---|---|---|
| UIFR3 | 0 | — (Status) | VBUSi | VBUS interrupt | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 1 | | (VBUSs) | VBUS status | × | × |
| | 2 | | SPRSi | Suspend/resume interrupt | $\overline{\text{IRQ6}}$ *4 | × |
| | 3 | | (SPRSs) | Suspend/resume status | × | × |
| | 4 | | Reserved | — | — | — |
| | 5 | | SETC | Set_Configuration detection | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 6 | | SOF | Start of Frame packet detection | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |
| | 7 | | CK48READY | USB operating clock stabilization detection | $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ | × |

Notes: 1. EP0 interrupts must be assigned to the same interrupt request signal.
2. An EP1 DMA transfer by a USB request is specified by the EP1T1 and EP1T0 bits in UDMAR.
3. An EP2 DMA transfer by a USB request is specified by the EP2T1 and EP2T0 bits in UDMAR.
4. The suspend/resume interrupt request $\overline{\text{IRQ6}}$ must be specified to be detected at the falling edge (IRQ6SCB and IRQ6SCA in ISCRH = 01) by the interrupt controller register.
5. The $\overline{\text{DREQ}}$ signal is not used for auto-request. The CPU can activate the DMAC using any flags and interrupts.

- $\overline{\text{EXIRQ0}}$ signal

  The $\overline{\text{EXIRQ0}}$ signal requests interrupt sources for which the corresponding bits in interrupt select registers 0 to 3 (UISR0 to UISR3) are cleared to 0. The $\overline{\text{EXIRQ0}}$ is driven low if a corresponding bit in the interrupt flag register is set to 1.
- $\overline{\text{EXIRQ1}}$ signal

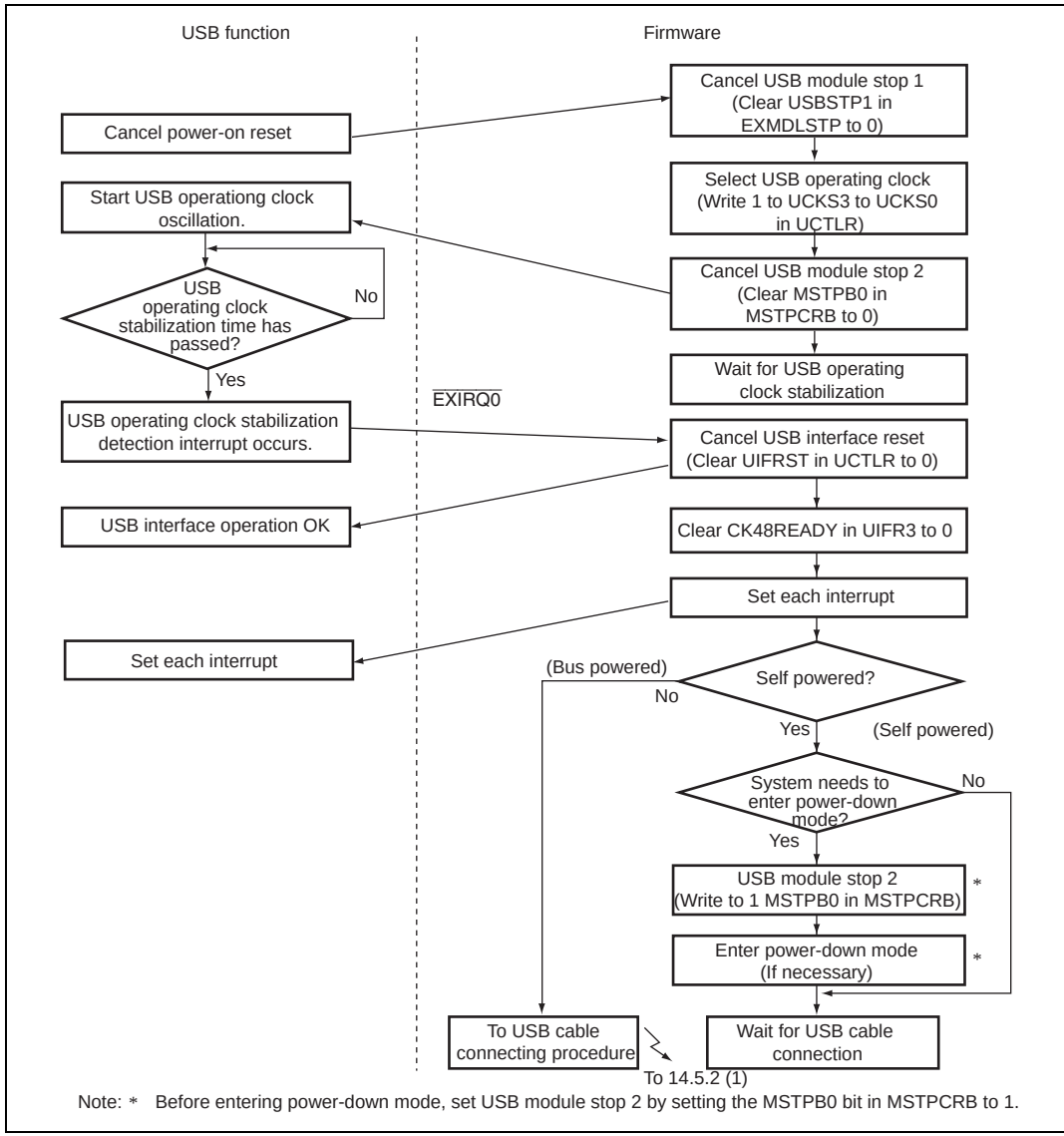
  The $\overline{\text{EXIRQ1}}$ signal requests interrupt sources for which the corresponding bits in interrupt select registers 0 to 3 (UISR0 to UISR3) are cleared to 0. The $\overline{\text{EXIRQ1}}$ is driven low if a corresponding bit in the interrupt flag register is set to 1.
- $\overline{\text{IRQ6}}$ signal

  The $\overline{\text{IRQ6}}$ signal is specific to the suspend/resume interrupt request. The falling edge of the $\overline{\text{IRQ6}}$ signal is output at the transition from the suspend state or from the resume state.

## 14.5 Communication Operation

### 14.5.1 Initialization

The USB must be initialized as described in the flowchart in figure 14.2.



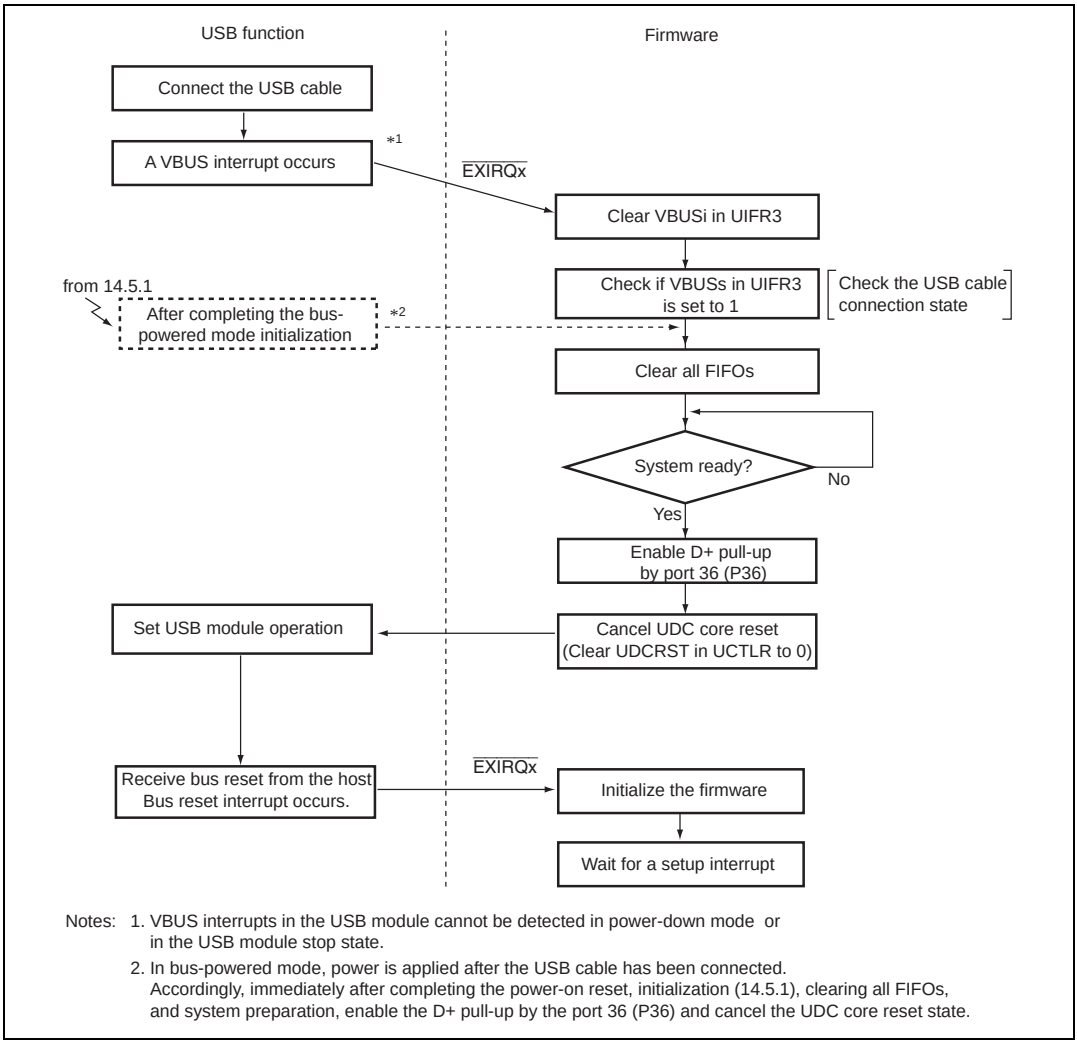

Note: * Before entering power-down mode, set USB module stop 2 by setting the MSTPB0 bit in MSTPCRB to 1.

**Figure 14.2 USB Initialization**

### 14.5.2 USB Cable Connection/Disconnection

**(1) USB Cable Connection (When USB module stop or power-down mode is not used)**

If the USB cable enters the connection state from the disconnection state in an application (self powered) where USB module stop or power-down mode is not used, perform the operation as shown in figure 14.3. In bus-powered mode, perform the operation according to note 2 in figure 14.3.



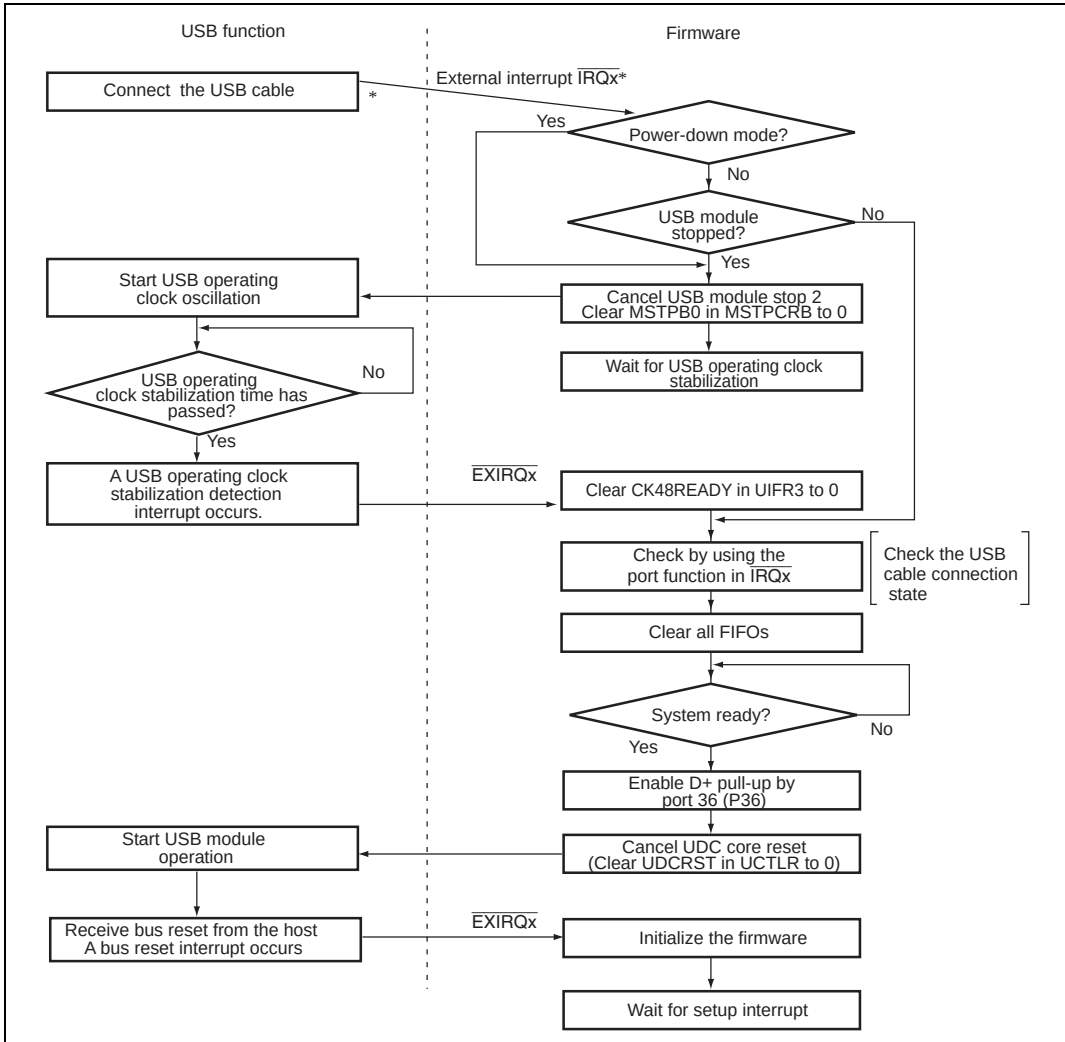

Notes: 1. VBUS interrupts in the USB module cannot be detected in power-down mode or in the USB module stop state.

2. In bus-powered mode, power is applied after the USB cable has been connected. Accordingly, immediately after completing the power-on reset, initialization (14.5.1), clearing all FIFOs, and system preparation, enable the D+ pull-up by the port 36 (P36) and cancel the UDC core reset state.

**Figure 14.3 USB Cable Connection (When USB Module Stop or Power-Down Mode Is not Used)**

**(2) USB Cable Connection (When USB module stop or power-down mode is used)**

If the USB cable enters the connection state from the disconnection state in an application (self powered) where USB module stop or power-down mode is used, perform the operation as shown in figure 14.4.




Note: * A VBUS interrupts in the USB module cannot be detected in power-down mode or in the USB module stop state. Accordingly, in an application (self powered) where power-down mode or USB module stop is used,VBUS interrupts of the USB must be detected via the external interrupt pin IRQx.
In this case, the IRQx pin must be specified as both-edge sensitive. When IRQx is used, VBUS interrupts in the USB module need not to be used.

**Figure 14.4 USB Cable Connection (When USB Module Stop or Power-Down Mode Is Used)**

**(3) USB Cable Disconnection (When USB module stop or power-down mode is not used)**

If the USB cable enters the disconnection state from the connection state in an application (self powered) where USB module stop or power-down mode is not used, perform the operation as shown in figure 14.5. In bus-powered mode, the power is automatically turned off when the USB cable is disconnected and the following processing is not required.

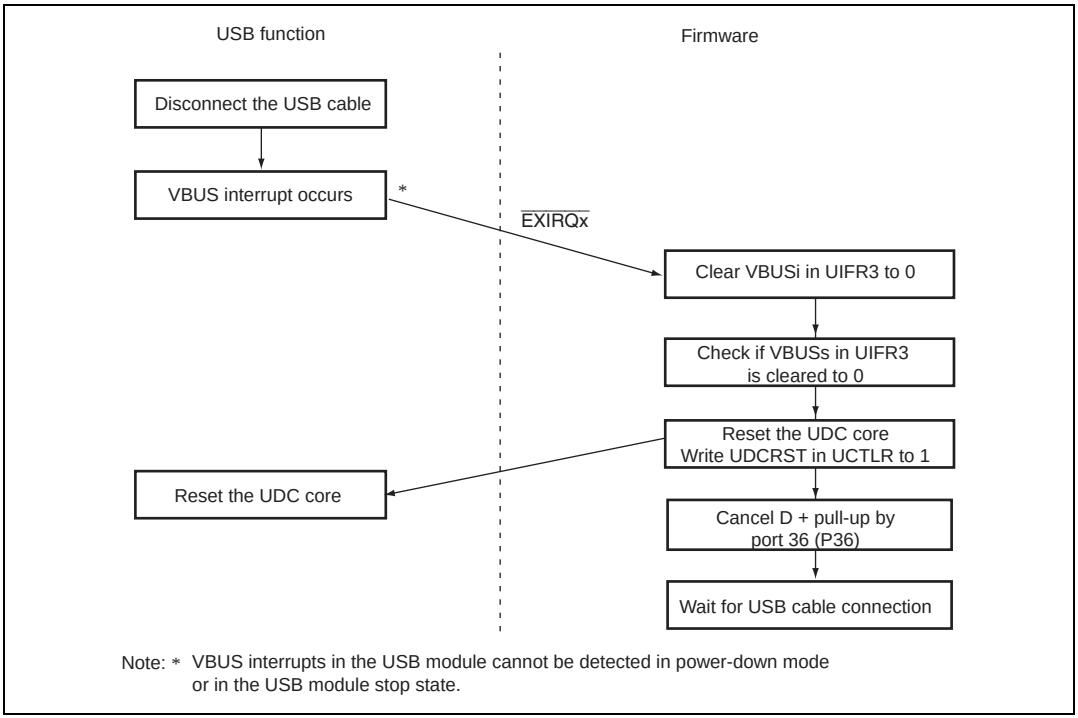
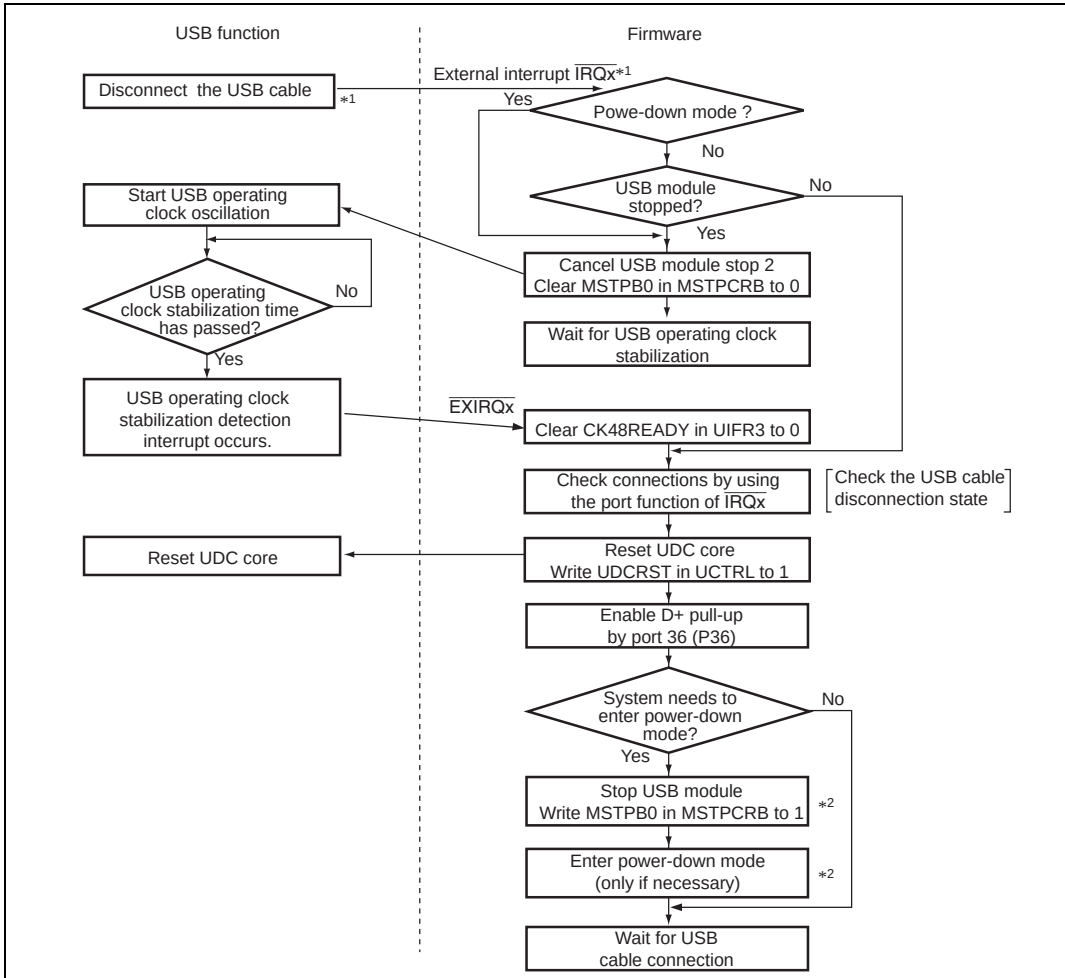



**Figure 14.5 USB Cable Disconnection (When USB Module Stop or Power-Down Mode Is not Used)**

**(4) USB Cable Disconnection (When USB module stop or power-down mode is used)**

If the USB cable enters the disconnection state from the connection state in an application (self powered) where USB module stop or power-down mode is used, perform the operation as shown in figure 14.6.




Notes: 1. VBUS interrupts in the USB module cannot be detected in power-down mode or in the USB module stop state. Accordingly, in an application (self powered) where power-down mode or USB module stop is used , VBUS interrupts of the USB must bedetected via the external interrupt pin IRQx. In this case, the IRQx pin must be specified as both edge sensitive. When IRQx is used, VBUS interrupts in the USB module need not to be used.

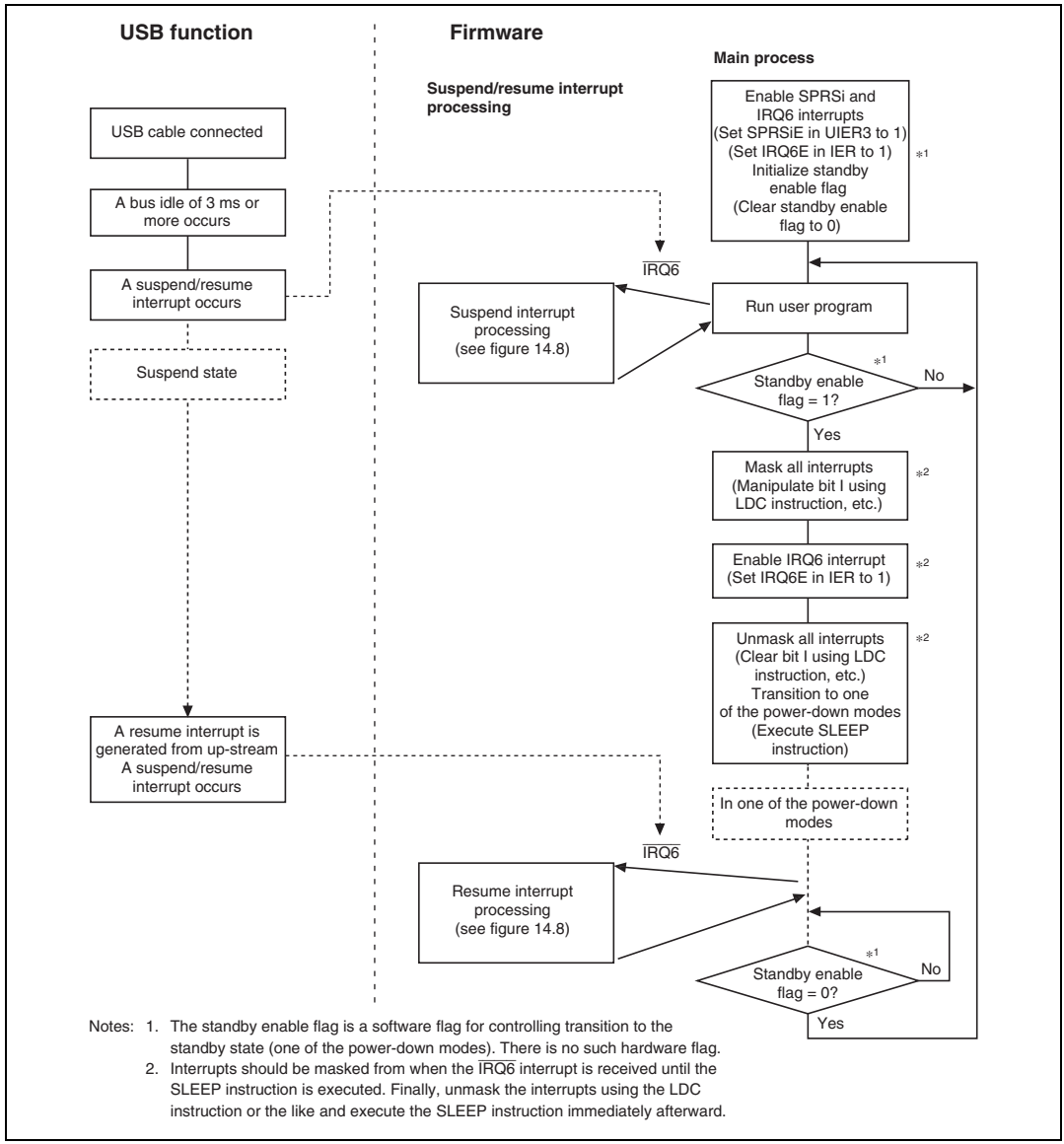
2. Before entering power-down mode, make sure to set USB module stop2 (the MSTPB0 bit of MSTPCRB = 1).

**Figure 14.6 USB Cable Disconnection (When USB Module Stop or Power-Down Mode Is Used)**

### 14.5.3 Suspend and Resume Operations

#### (1) Suspend and Resume Operations

Figures 14.7 and 14.8 are flowcharts of the suspend and resume operations. If the USB bus enters the suspend state from a non-suspend state, or if it enters a non-suspend state from the suspend state due to a resume signal from up-stream, perform the operations shown below.

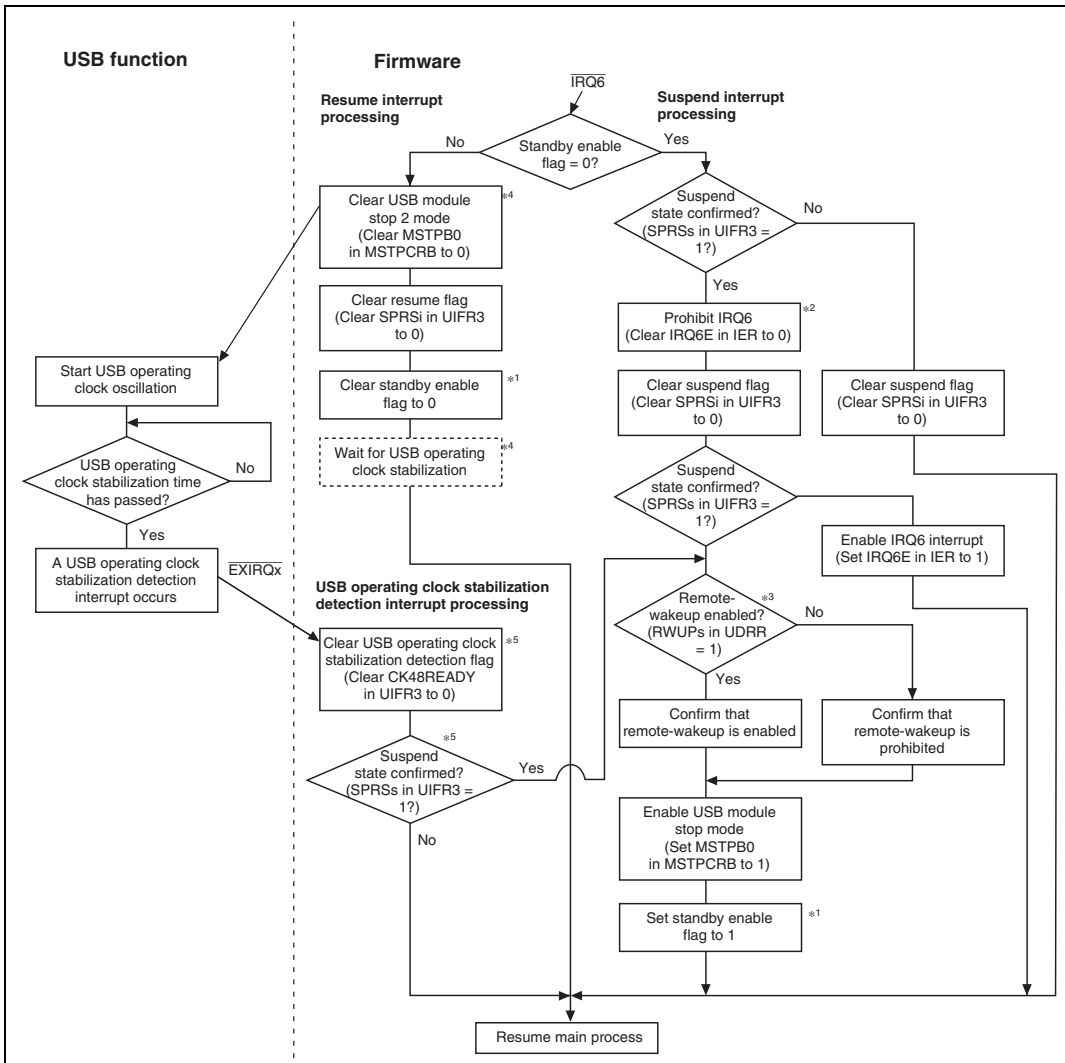



Notes: 1. The standby enable flag is a software flag for controlling transition to the standby state (one of the power-down modes). There is no such hardware flag.
2. Interrupts should be masked from when the $\overline{\text{IRQ6}}$ interrupt is received until the SLEEP instruction is executed. Finally, unmask the interrupts using the LDC instruction or the like and execute the SLEEP instruction immediately afterward.

**Figure 14.7 Example Flowchart of Suspend and Resume Operations**

### (2) Suspend and Resume Interrupt Processing

Figure 14.8 is a flowchart of suspend and resume interrupt processing.

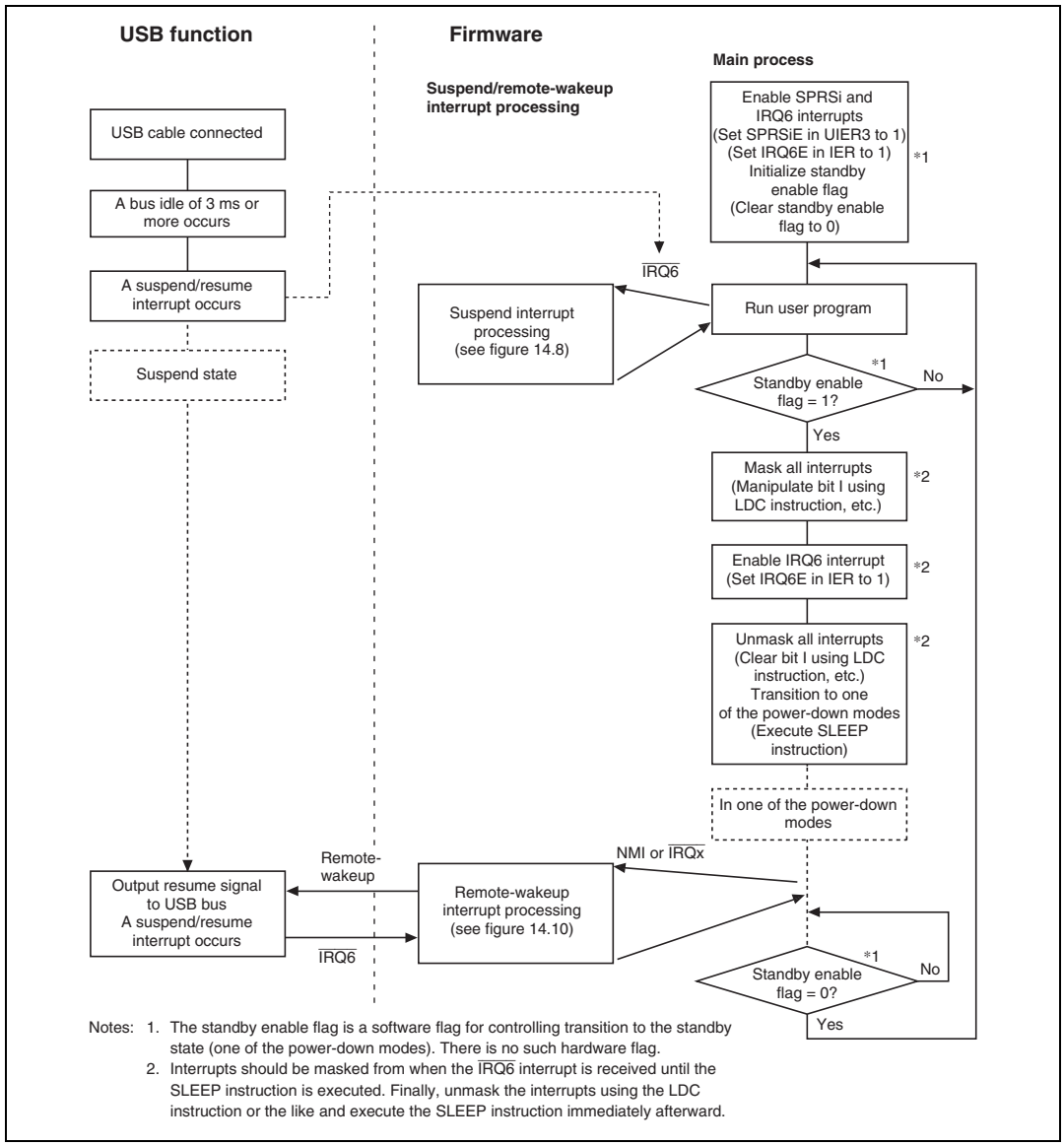



Notes: 1. The standby enable flag is a software flag for controlling transition to the standby state (one of the power-down modes). There is no such hardware flag.
2. Interrupts should be masked from when the $\overline{\text{IRQ6}}$ interrupt is received until the SLEEP instruction is executed. Finally, unmask the interrupts using the LDC instruction or the like and execute the SLEEP instruction immediately afterward.
3. The remote-wakeup function cannot be used unless it is enabled by the host. Accordingly, the remote-wakeup function cannot be used unless it is enabled by the host. Accordingly, make sure to check RWUPs in UDRR before using the remote-wakeup function. However, it is not necessary to confirm that the remote-wakeup function is enabled by the host if the application does not make use of this function.
4. When resuming using the remote wakeup function, the USB module stop state must already be cleared.
5. Return to the main process and wait for the USB operating clock stabilization detection interrupt. When resuming by means of remote-wakeup the USB operating clock has already stabilized, so this step is not necessary.

**Figure 14.8 Example Flowchart of Suspend and Resume Interrupt Processing**

### (3) Suspend and Remote-Wakeup Operations

Figures 14.9 and 14.10 are flowcharts of the suspend and remote-wakeup operations. If the USB bus enters a non-suspend state from the suspend state due to a remote-wakeup signal from this function, perform the operations shown below.




Notes: 1. The standby enable flag is a software flag for controlling transition to the standby state (one of the power-down modes). There is no such hardware flag.
2. Interrupts should be masked from when the IRQ6 interrupt is received until the SLEEP instruction is executed. Finally, unmask the interrupts using the LDC instruction or the like and execute the SLEEP instruction immediately afterward.

**Figure 14.9 Example Flowchart of Suspend and Remote-Wakeup Operations**

### (4) Remote-Wakeup Interrupt Processing

Figure 14.10 is a flowchart of remote-wakeup interrupt processing.

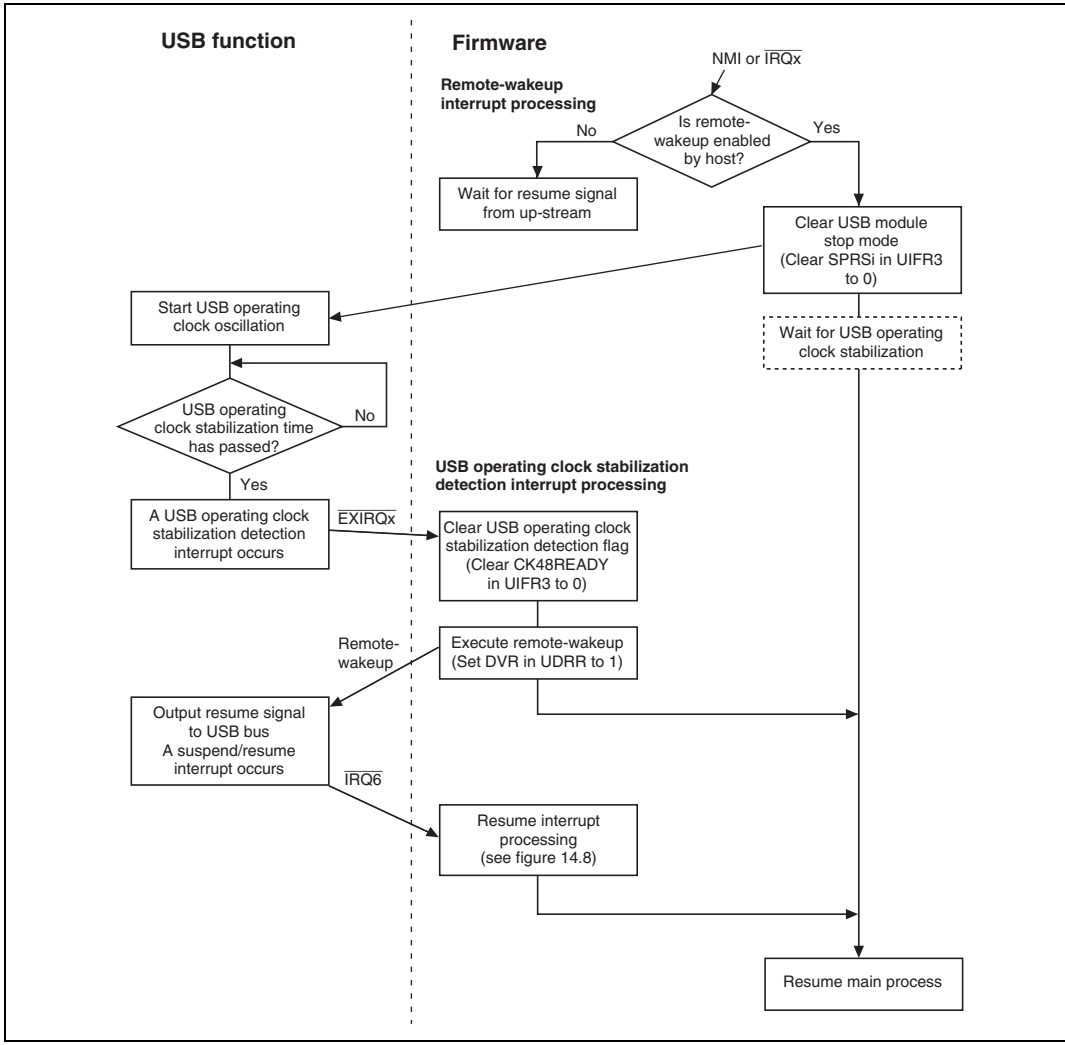



**Figure 14.10 Example Flowchart of Remote-Wakeup Interrupt Processing**

### 14.5.4 Control Transfer

The control transfer consists of three stages; setup, data (sometimes omitted), and status, as shown in figure 14.11. The data stage consists of multiple bus transactions. Figures 14.12 to 14.16 show operation flows in each stage.

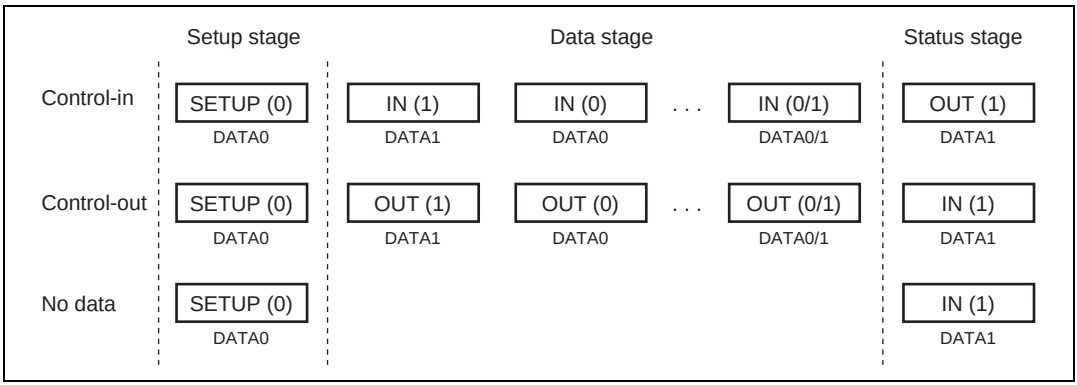
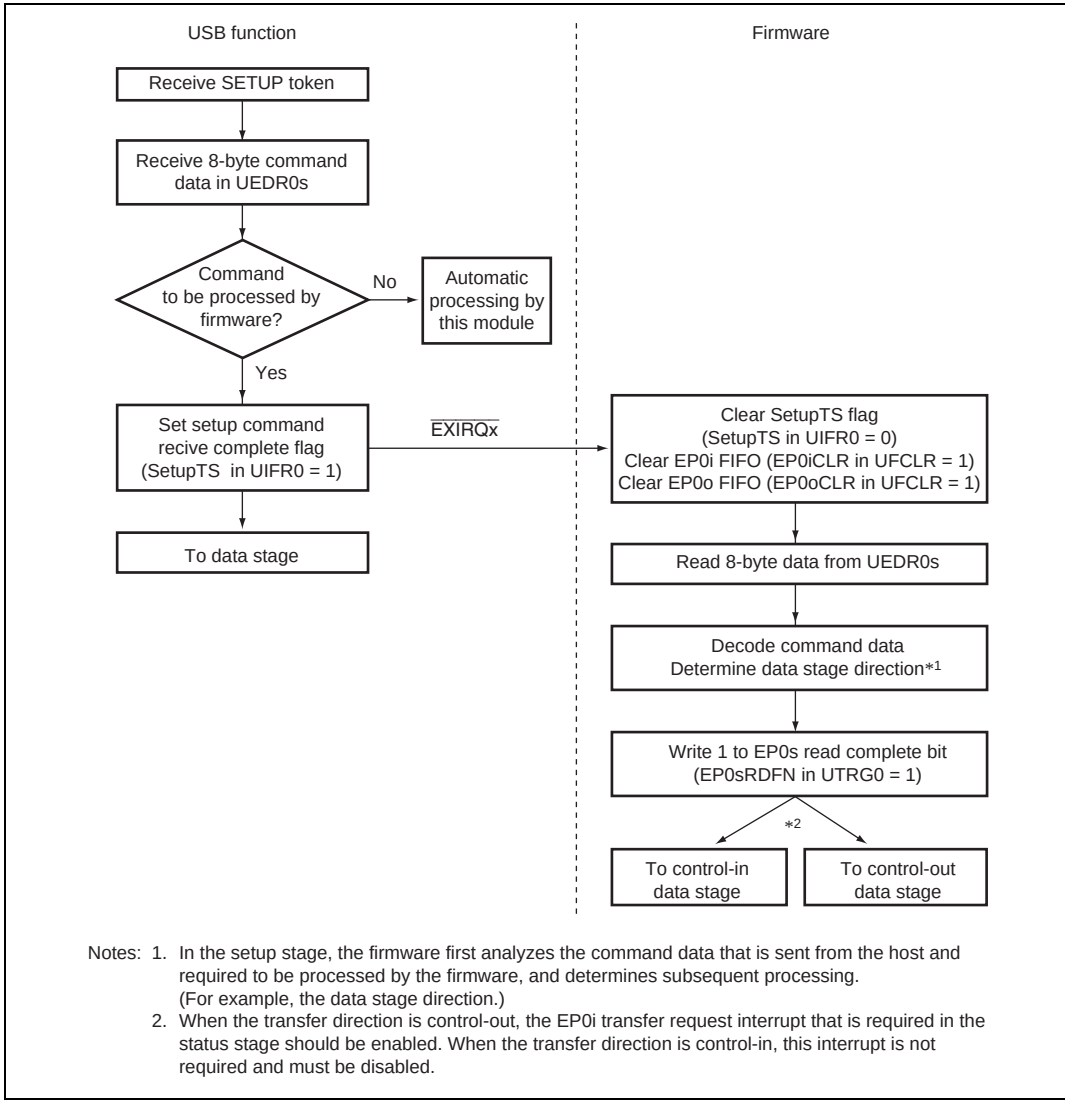



**Figure 14.11 Control Transfer Stage Configuration**

### (1) Setup Stage



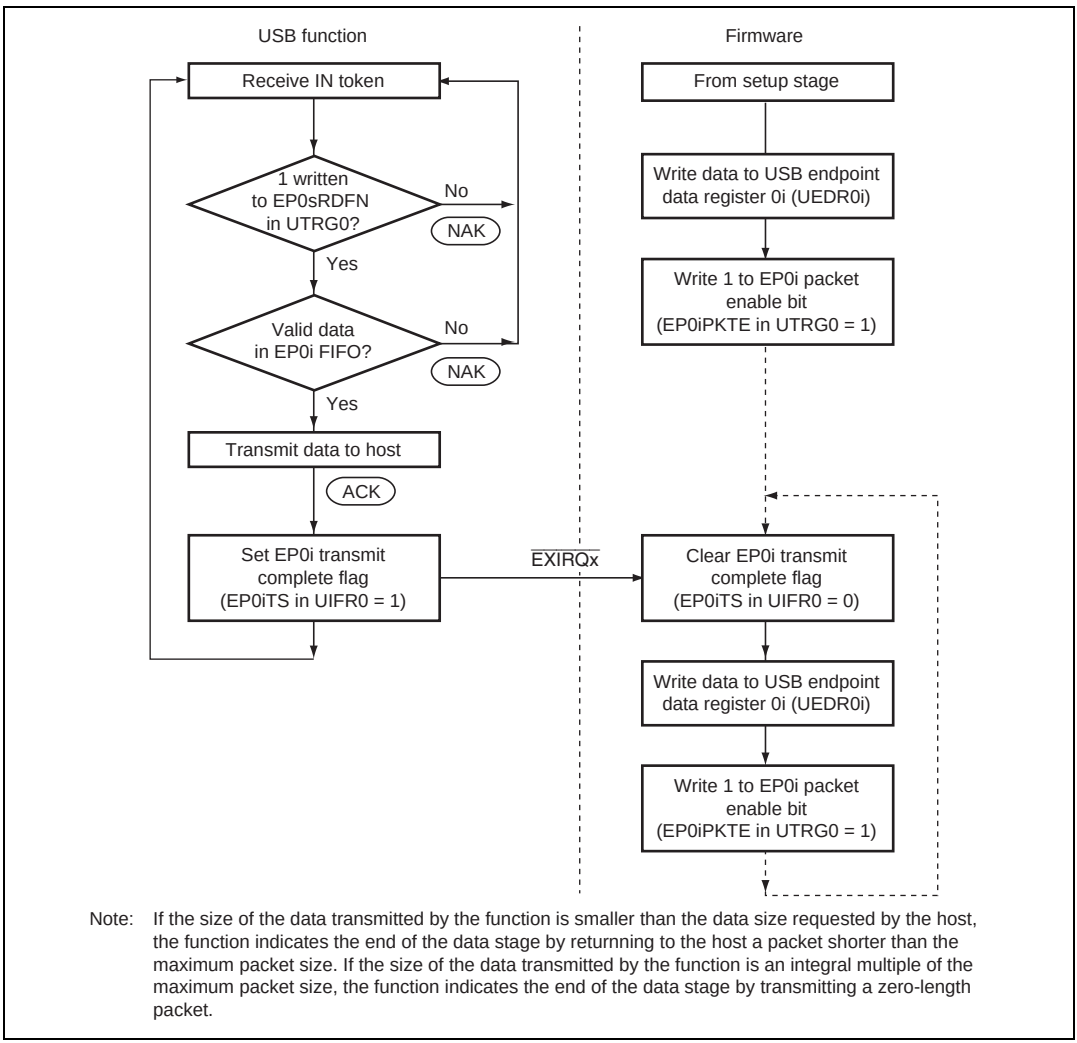

Notes: 1. In the setup stage, the firmware first analyzes the command data that is sent from the host and required to be processed by the firmware, and determines subsequent processing. (For example, the data stage direction.)
2. When the transfer direction is control-out, the EP0i transfer request interrupt that is required in the status stage should be enabled. When the transfer direction is control-in, this interrupt is not required and must be disabled.

**Figure 14.12 Setup Stage Operation**

**(2) Data Stage (Control-In)**

The firmware first analyzes the command data that is sent from the host in the setup stage, and determines the subsequent data stage direction. If the result of command data analysis is that the data stage is in-transfer, one packet of data to be sent to the host is written to the FIFO. If there is more data to be sent, this data is written to the FIFO after the data written first has been sent to the host (EP0iTS in UIFR0 is set to 1).

The end of the data stage is identified when the host transmits an OUT token and the status stage is entered.




Note: If the size of the data transmitted by the function is smaller than the data size requested by the host, the function indicates the end of the data stage by returnning to the host a packet shorter than the maximum packet size. If the size of the data transmitted by the function is an integral multiple of the maximum packet size, the function indicates the end of the data stage by transmitting a zero-length packet.

**Figure 14.13 Data Stage Operation (Control-In)**

**(3) Data Stage (Control-Out)**

The firmware first analyzes the command data that is sent from the host in the setup stage, and determines the subsequent data stage direction. If the result of command data analysis is that the data stage is out-transfer, data from the host is waited for, and after data is received (EP0oTS in UIFR0 is set to 1), data is read from the FIFO. Next, the firmware writes 1 to the EP0o read complete bit, empties the receive FIFO, and waits for reception of the next data.

The end of the data stage is identified when the host transmits an IN token and the status stage is entered.

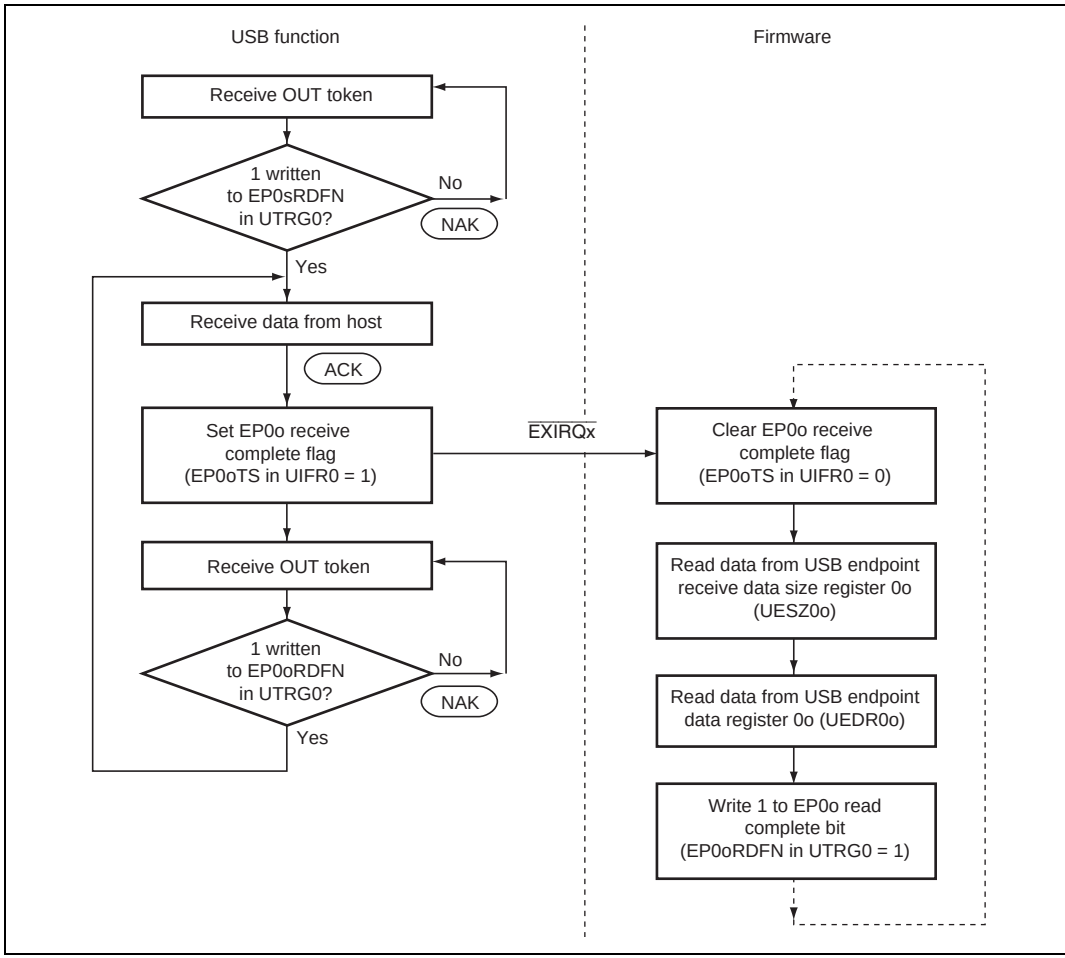



**Figure 14.14 Data Stage Operation (Control-Out)**

**(4) Status Stage (Control-In)**

The control-in status stage starts with an OUT token from the host. The firmware receives 0-byte data from the host, and ends control transfer.

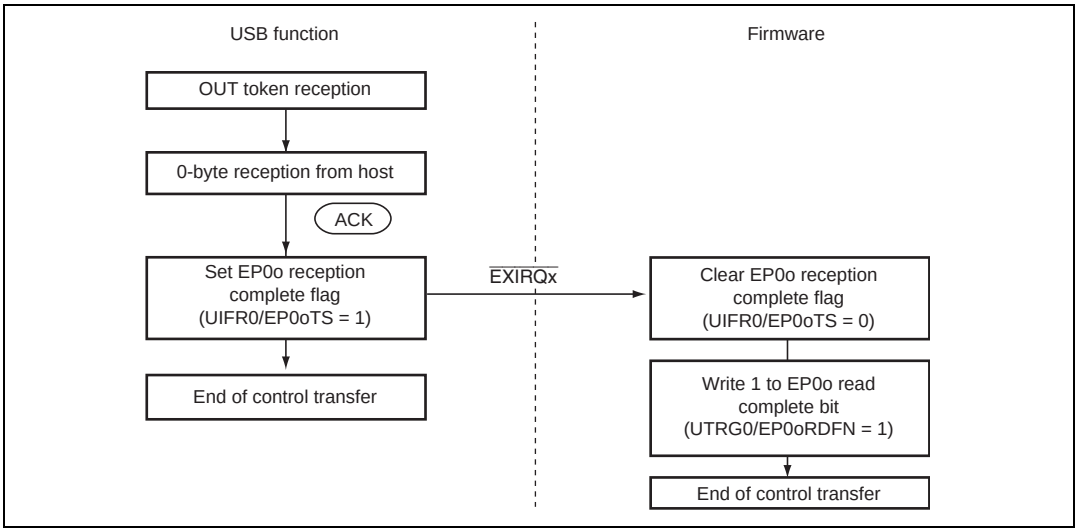



**Figure 14.15 Status Stage Operation (Control-In)**

**(5) Status Stage (Control-Out)**

The control-out status stage starts with an IN token from the host. When an IN-token is received at the start of the status stage, there is not yet any data in the EP0i FIFO, and so an EP0i transfer request interrupt is generated. The firmware recognizes from this interrupt that the status stage has started. Next, in order to transmit 0-byte data to the host, 1 is written to the EP0i packet enable bit but no data is written to the EP0i FIFO. As a result, the next IN token causes 0-byte data to be transmitted to the host, and control transfer ends.

After the firmware has finished all processing relating to the data stage, 1 should be written to the EP0i packet enable bit.

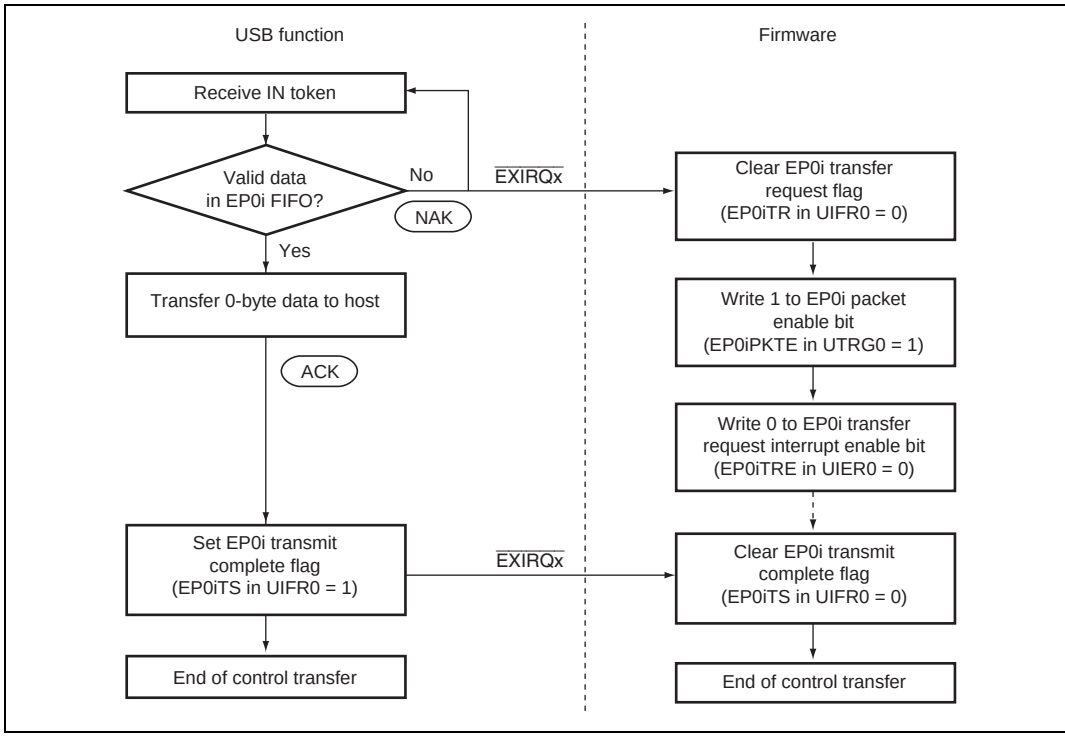
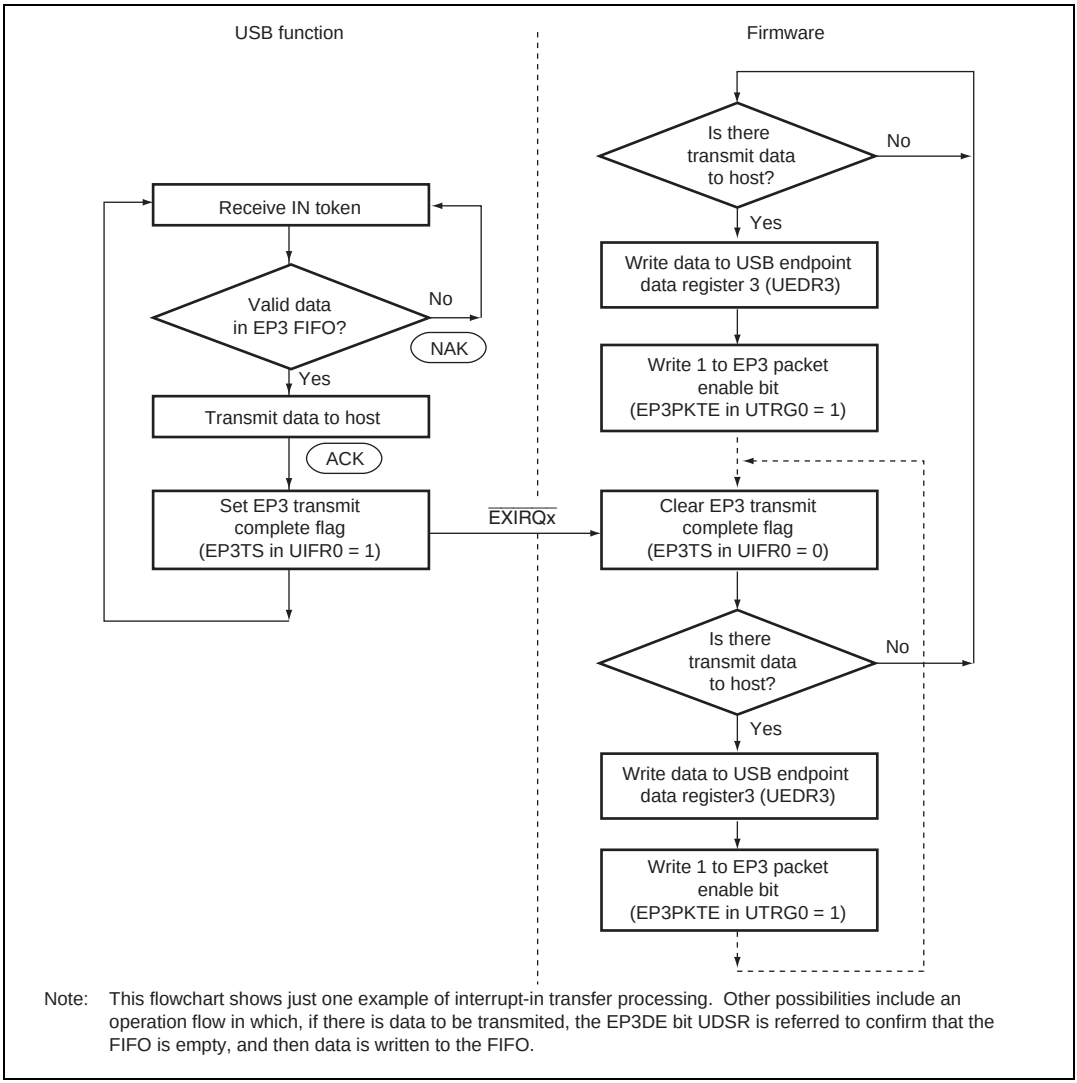



**Figure 14.16 Status Stage Operation (Control-Out)**

### 14.5.5 Interrupt-In Transfer (Endpoint 3)




Note: This flowchart shows just one example of interrupt-in transfer processing. Other possibilities include an operation flow in which, if there is data to be transmited, the EP3DE bit UDSR is referred to confirm that the FIFO is empty, and then data is written to the FIFO.

**Figure 14.17 EP3 Interrupt-In Transfer Operation**

### 14.5.6 Bulk-In Transfer (Dual FIFOs) (Endpoint 1)

EP1 has two 64-byte FIFOs, but the user can transmit data and write transmit data without being aware of this dual-FIFO configuration. However, one data write should be performed for one FIFO. For example, even if both FIFOs are empty, it is not possible to perform EP1PKTE at one time after consecutively writing 128 bytes of data. EP1PKTE must be performed for each 64- byte write.

When transmitting data to the host by bulk-in transfer, first enable the EP1 FIFO empty interrupt by writing 1 to EP1EMPTYE in UIER1. At first, both EP1 FIFOs are empty, and so an EP1 FIFO empty interrupt is generated immediately. The data to be transmitted is written to the data register using this interrupt. After the first transmit data write for one FIFO, the other FIFO is empty, and so the next transmit data can be written to the other FIFO immediately. When both FIFOs are full, EP1EMPTY is cleared to 0. If at least one FIFO is empty, UIFR1/EP1EMPTY is set to 1. When ACK is returned from the host after data transmission is completed, the FIFO used in the data transmission becomes empty. If the other FIFO contains valid transmit data at this time, transmission can be continued.

When transmission of all data has been completed, write 0 to UIER1/EP1EMPTYE and disable $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ interrupt requests.

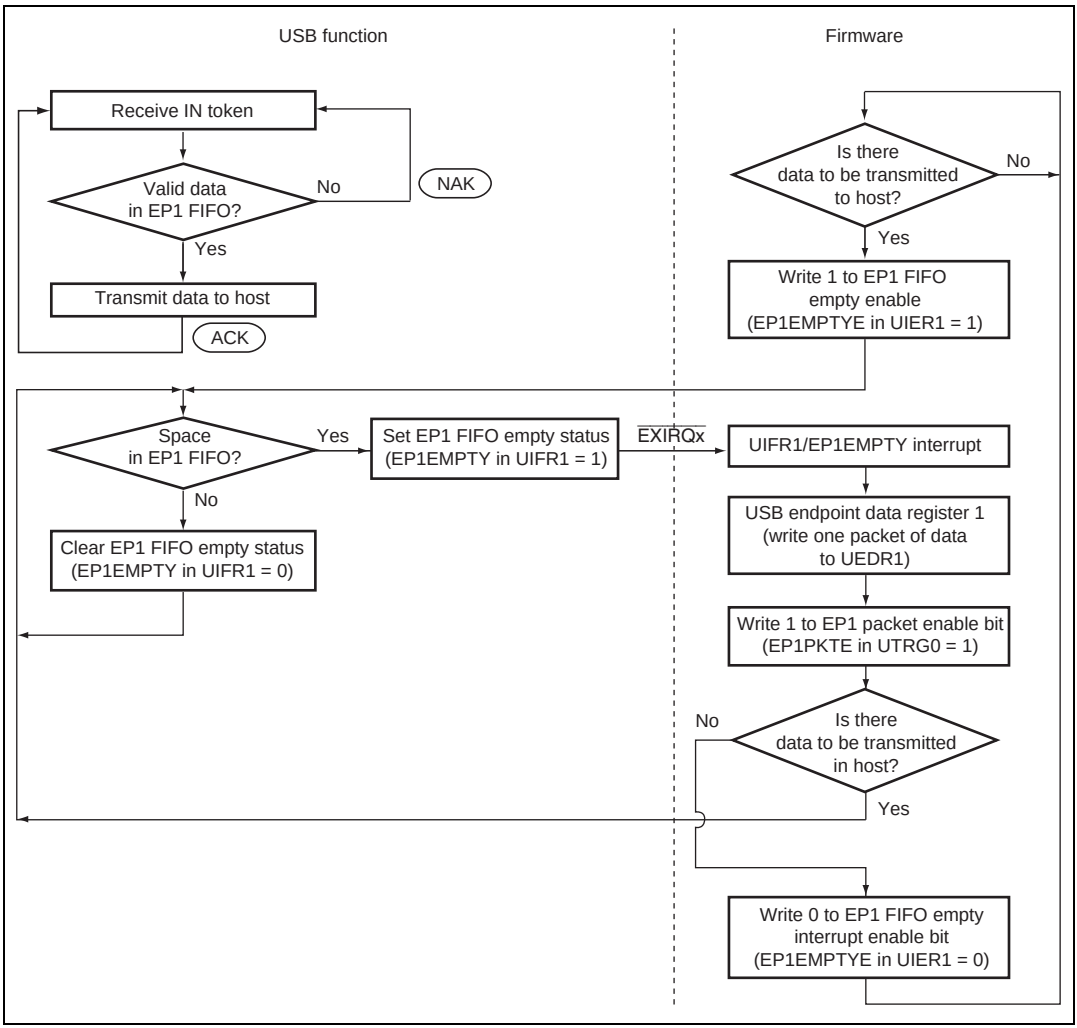



**Figure 14.18 EP1 Bulk-In Transfer Operation**

### 14.5.7 Bulk-Out Transfer (Dual FIFOs) (Endpoint 2)

EP2 has two 64-byte FIFOs, but the user can receive data and read receive data without being aware of this dual-FIFO configuration.

When one FIFO is full after reception is completed, the UIFR1/EP2READY bit is set. After the first receive operation into one of the FIFOs when both FIFOs are empty, the other FIFO is empty, and so the next packet can be received immediately. When both FIFOs are full, NAK is returned to the host automatically. When reading of the receive data is completed following data reception, 1 is written to the UTRG0/EP2RDFN bit. This operation empties the FIFO that has just been read, and makes it ready to receive the next packet.

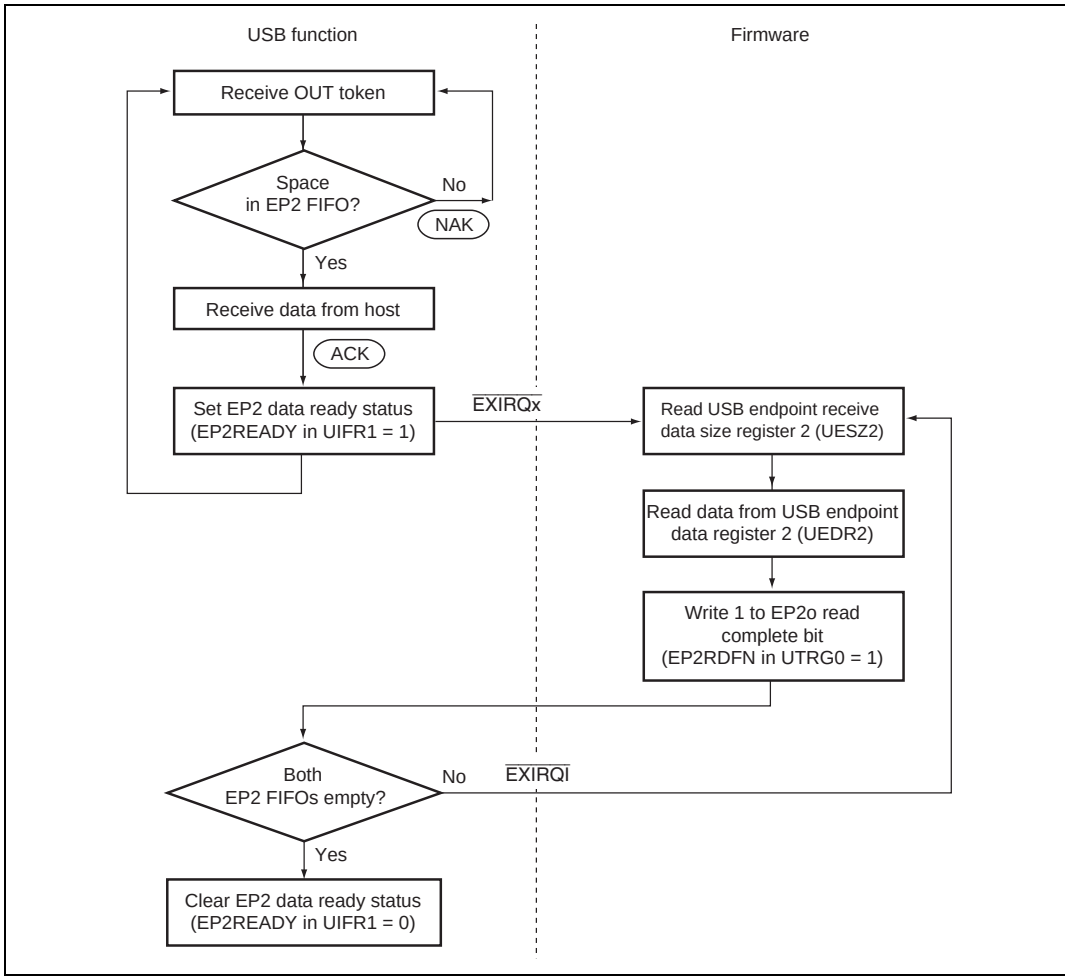



**Figure 14.19 EP2 Bulk-Out Transfer Operation**

### 14.5.8 Processing of USB Standard Commands and Class/Vendor Commands

**(1) Processing of Commands Transmitted by Control Transfer**

A command transmitted from the host by control transfer may require decoding and execution of command processing by the firmware. Whether or not command decoding is required by the firmware is indicated in table 14.5 below.

**Table 14.5 Command Decoding by Firmware**

| Decoding not Necessary by Firmware | Decoding Necessary by Firmware |
|---|---|
| Clear Feature | Get Descriptor |
| Get Configuration | Synch Frame |
| Get Interface | Set Descriptor |
| Get Status | Class/Vendor command |
| Set Address | |
| Set Configuration | |
| Set Feature | |
| Set Interface | |

If decoding is not necessary by the firmware, command decoding and data stage and status stage processing are performed automatically. No processing is necessary by the user. An interrupt is not generated in this case.

If decoding is necessary by the firmware, the USB function module stores the command in the EP0s FIFO. After normal reception is completed, the SetupTS flag in UIER0 is set and an interrupt request is generated from the $\overline{\text{EXIRQ}}$x pin. In the interrupt routine, eight bytes of data must be read from the EP0s data register (UEDR0s) and decoded by the firmware. The necessary data stage and status stage processing should then be carried out according to the result of the decoding operation.

### 14.5.9 Stall Operations

**(1) Overview**

This section describes stall operations in the USB function module. There are two cases in which the USB function module stall function is used:

A. When the firmware forcibly stalls an endpoint for some reason

B. When a stall is performed automatically within the USB function module due to a USB specification violation.

The USB function module has internal status bits that hold the status (stall or non-stall) of each endpoint. When a transaction is sent from the host, the module refers these internal status bits and determines whether to return a stall to the host. These bits cannot be cleared by the firmware; they must be cleared with a Clear Feature command from the host. However, the internal status bit for EP0 is cleared automatically at the reception of the setup command.

**(2) Forcible Stall by Firmware**

The firmware uses the UESTL register to issue a stall request for the USB function module. When the firmware wishes to stall a specific endpoint, it sets the corresponding EPnSTL bit (1-1 in figure 14.20). The internal status bits are not changed at this time.

When a transaction is sent from the host for the endpoint for which the EPnSTL bit was set, the USB function module refers the internal status bit, and if this is not set, refers the corresponding EPnSTL bit (1-2 in figure 14.20). If the corresponding EPnSTL bit is not set, the internal status bit is not changed and the transaction is accepted. If the corresponding EPnSTL bit is set, the USB function module sets the internal status bit and returns a stall handshake to the host (1-3 in figure 14.20).

Once an internal status bit is set, it remains set until cleared by a Clear Feature command from the host, without regarding to EPnSTL. Even after a bit is cleared by the Clear Feature command (3-1 in figure 14.20), the USB function module continues to return a stall handshake while the EPnSTL bit is set, since the internal status bit is set each time a transaction is executed for the corresponding endpoint (1-2 in figure 14.20). To clear a stall, therefore, it is necessary for the corresponding EPnSTL bit to be cleared by the firmware, and also for the internal status bit to be cleared with a Clear Feature command (2-1 to 2-3 in figure 14.20).

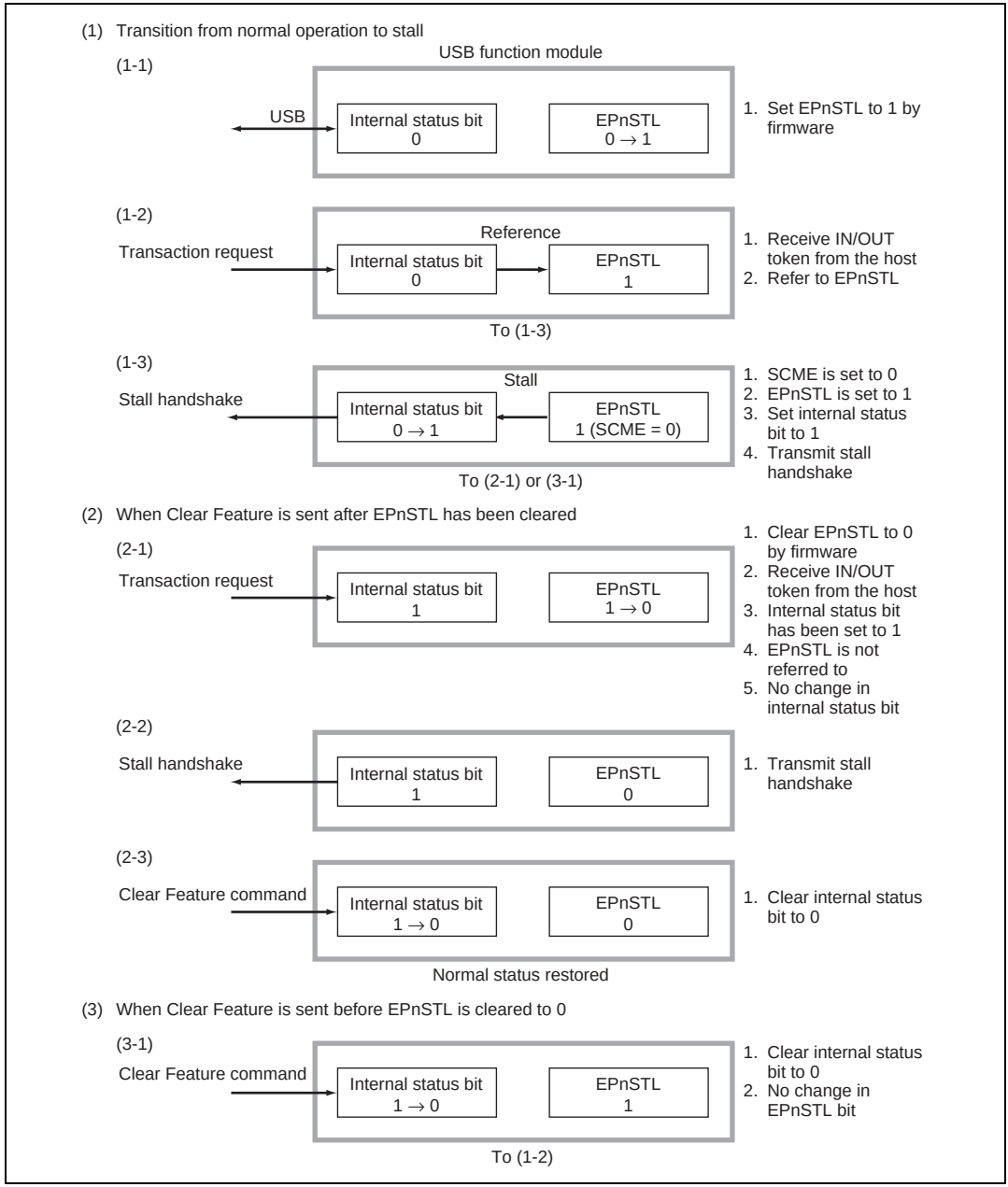



**Figure 14.20 Forcible Stall by Firmware**

### (3) Automatic Stall by USB Function Module

When a stall setting is made with the Set Feature command, when the information of this module differs from that returned to the host by the Get Descriptor, or in the event of a USB specification violation, the USB function module automatically sets the internal status bit for the corresponding endpoint without regarding to EPnSTL, and returns a stall handshake (1-1 in figure 14.21).

Once an internal status bit is set, it remains set until cleared by a Clear Feature command from the host, without regarding to EPnSTL. After a bit is cleared by the Clear Feature command, EPnSTL is referred (3-1 in figure 14.21). The USB function module continues to return a stall handshake while the internal status bit is set, since the internal status bit is set even if a transaction is executed for the corresponding endpoint (2-1 and 2-2 in figure 14.21). To clear a stall, therefore, the internal status bit must be cleared with a Clear Feature command (3-1 in figure 14.21). If set by the firmware, EPnSTL should also be cleared (2-1 in figure 14.21).

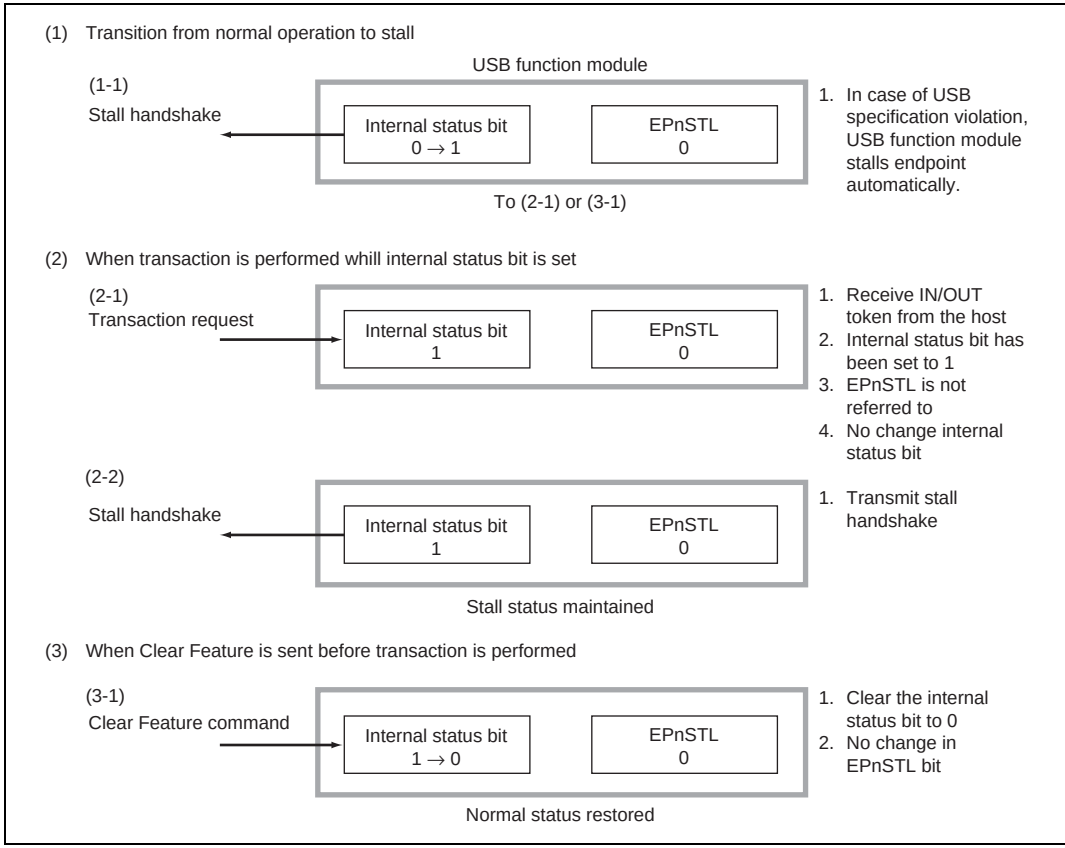



**Figure 14.21 Automatic Stall by USB Function Module**

## 14.6 DMA Transfer Specifications

Two methods of USB request and auto request are available for the DMA transfer of USB data.

### 14.6.1 DMAC Transfer by USB Request

#### (1) Overview

Only normal mode in full address mode (cycle steal mode) supports the transfer by a USB request of the on-chip DMAC. Endpoints that can be transferred by the on-chip DMAC are EP1 and EP2 in Bulk transfer (corresponding registers are UEDR1 and UEDR2). In DMA transfer, the USB module must be accessed as an external device in area 6. The USB module cannot be accessed as a device with external ACK (single-address transfer cannot be performed). 0-byte data transfer to EP2 is ignored even if the DMA transfer is enabled by setting the EP2T1 bit in UDMAR to 1.

#### (2) On-Chip DMAC Settings

The on-chip DMAC must be specified as follows: A USB request ($\overline{\text{DREQ}}$ signal is used), activated by low-level input, byte size, full-address mode transfer, and the DTA bit in DMABCR = 1. After completing the DMA transfer of specified times, the DMAC automatically stops. Note, however, that the USB module keeps the $\overline{\text{DREQ}}$ signal low while data to be transferred by the on-chip DMAC remains regardless of the DMAC status.

#### (3) EP1 DMA Transfer

The EP1T1 bit in UDMAR enables the DMA transfer. The EP1T0 bit in UDMAR specifies the $\overline{\text{DREQ}}$ signal to be used by the DMA transfer. When 1 is written to the EP1T1 bit, the $\overline{\text{DREQ}}$ signal is driven low if at least one of EP1 data FIFOs is empty; the $\overline{\text{DREQ}}$ signal is driven high if both EP1 data FIFOs are full.

**(a) EP1PKTE in UTRG0**

When DMA transfer is performed on EP1 transmit data, the USB module automatically performs the same processing as writing 1 to EP1PKTE if one data FIFO (64 bytes) becomes full. Accordingly, to transfer data of integral multiples of 64 bytes, the user needs not to write 1 to EP1PKTE. To transfer data of less than 64 bytes, the user must write 1 to EP1PKTE using the DMA transfer end interrupt of the on-chip DMAC. If the user writes 1 to EP1PKTE in cases other than the case when data of less than 64 bytes is transferred, excess transfer occurs and correct operation cannot be guaranteed.

Figure 14.22 shows an example for transmitting 150 bytes of data from EP1 to the host. In this case, internal processing as the same as writing 1 to EP1PKTE is automatically performed twice. This kind of internal processing is performed when the currently selected data FIFO becomes full.

Accordingly, this processing is automatically performed only when 64-byte data is sent. This processing is not performed automatically when data less than 64 bytes is sent.

**(b) EP1 DMA transfer procedure**

1. Set the bits EP1T1 and EP1T0 in UDMAR.
2. Set DMAC (specifies the number of transfers in DMAC to transmit 150 bytes of data).
3. Activate DMAC.
4. Perform DMA transfer.
5. Write 1 to the EP1PKTE bit in UTRG0 by a DMA transfer end interrupt.

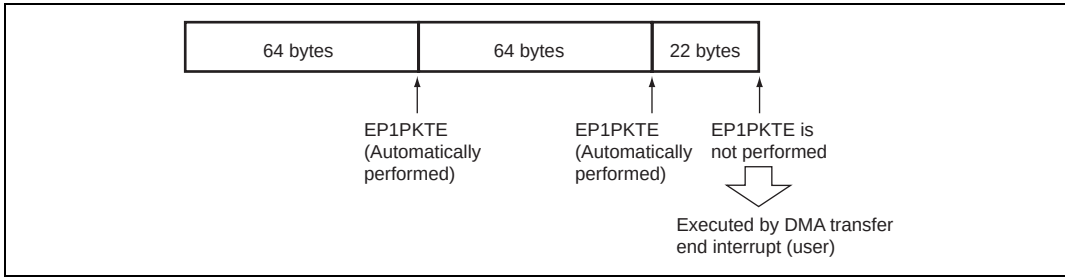



**Figure 14.22 EP1PKTE Operation in UTRG0**

**(4) EP2 DMA Transfer**

The EP2T1 bit in UDMAR enables the DMA transfer. The EP2T0 bit in the UDMAR specifies the $\overline{\text{DREQ}}$ signal to be used by the DMA transfer. When 1 is written to the EP2T1 bit, the $\overline{\text{DREQ}}$ signal is driven low if at least one of EP2 data FIFOs is full (ready state); the $\overline{\text{DREQ}}$ signal is driven high if both EP2 data FIFOs are empty when all receive data items are read.

**(a) EP2RDFN in UTRG0**

When DMA transfer is performed on EP2 receive data, do not write 1 to EP2RDFN after one data FIFO (64 bytes) has been read. In data transfer other than DMA transfer, the next data cannot be read after one data FIFO (64 bytes) has been read unless 1 is written to EP2RDFN. While in DMA transfer, the USB module automatically performs the same processing as writing 1 to EP2RDFN if the currently selected data FIFO becomes empty. Accordingly, in DMA transfer, the user needs not to write 1 to EP2RDFN. If the user writes 1 to EP2RDFN in DMA transfer, excess transfer occurs and correct operation cannot be guaranteed.

Figure 14.23 shows an example of EP2 receiving 150 bytes of data from the host. In this case, internal processing as the same as writing 1 to EP2RDFN is automatically performed three times. This kind of internal processing is performed when the currently selected data FIFO becomes empty. Accordingly, this processing is automatically performed both when 64-byte data is sent and when data less than 64 bytes is sent.

**(b) EP2 DMA transfer procedure**

Perform DMAC transfer in 1 packet units. After setting the EP2READY flag, check the size of data received from the host and then set the data size as the number of DMAC transfers.

1. Set the bits EP2T1 and EP2T0 in UDMAR.
2. Wait for the EP2READY flag in UIFR1 to be set.
3. Set DMAC.
   Read the value in UESZ2 and specifies the size of received data (not more than 64 bytes) as the number of transfers.
4. Activate DMAC.
5. Perform DMA transfer (not more than 64 bytes).
6. Wait for DMA transfer end.
7. Repeat steps 2 to 6.

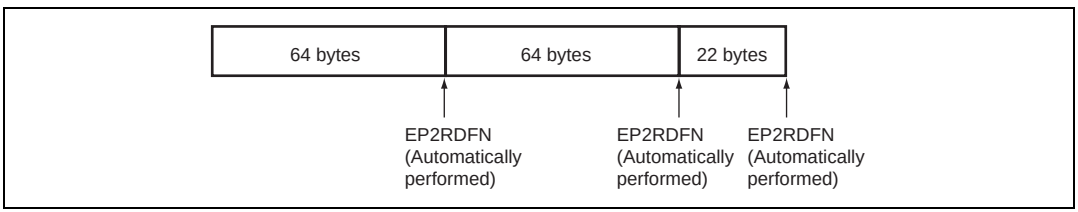



**Figure 14.23 EP2RDFN Operation in UTRG0**

### 14.6.2 DMA Transfer by Auto-Request

**(1) Overview**

Burst mode transfer or ycle steal transfer can be selected for the on-chip DMAC auto-request transfer. Endpoints that can be transferred by the on-chip DMAC are all registers (UEDR0s, UEDR0i, UEDR0o, UEDR1, UEDR2, UEDR3). Confirm flags and interrupts corresponding to each data register before activating the DMA. As UDMAR is not used in auto-request mode, set UDMAR to H'00.

**(2) On-Chip DMAC Settings**

The on-chip DMAC must be specified as follows: Auto-request, byte size, full-address mode transfer, and number of transfers equal to or less than the maximum packet size of the data register. After completing the DMAC transfers of specified time, the DMAC automatically stops.

### (3) EP0, EP1, or EP3 DMA Transfer

#### (a) EPnPKTE Bits of UTRG0 (n = 0i, 1, or 3)

Note that 1 is not automatically written to EPnPKTE in case of auto-request transfer. Always write 1 to EPnPKTE by the CPU. The following example shows when 150-byte data is transmitted from EP1 to the host. In this case, 1 should be written to EP2PKTE three times as shown in figure 14.24.

#### (b) EP1 DMA Transfer Procedure

The DMAC transfer unit should be one packet. Therefore, set the number of transfers so that it is equal to or less than the maximum packet size of each endpoint.

1. Confirm that UIFR1/EP1EMPTY flag is 1.
2. DMAC settings for EP1 data transfer (such as auto-request and address setting).
3. Set the number of transfers for 64 bytes (the maximum packet size or less) in the DMAC.
4. Activate the DMAC (write 1 to DTE after reading DTE as 0).
5. DMA transfer.
6. Write 1 to the UTRG0/EP1PKTE bit after the DMA transfer is completed.
7. Repeat steps 1 to 6 above.
8. Confirm that UIFR1/EP1EMPTY flag is 1.
9. Set the number of transfer for 22 bytes in the DMAC.
10. Activate the DMAC (write 1 to DTE after reading DTE as 0).
11. DMA transfer.
12. Write 1 to the UTRG0/EP1PKTE bit after the DMA transfer is completed.

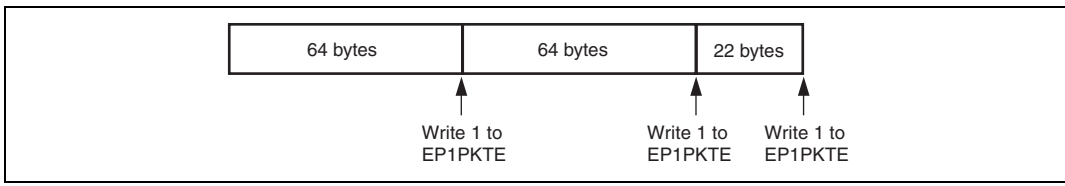



**Figure 14.24 EP1PKTE Operation in UTRG0 (Auto-Request)**

### (4) EP0o or EP2 DMA Transfer

#### (a) EPnRDFN Bits of UTRG0 (n = 0o or 2)

Note that 1 is not automatically written to EPnRDFN in case of auto-request transfer. Always write 1 to EPnRDFN by the CPU. The following example shows when EP2 receives 150-byte data from the host. In this case, 1 should be written to EP2RDFN three times as shown in figure 14.25.

**(b) EP2 DMA Transfer Procedure**

The DMAC transfer unit should be one packet. Therefore, set the number of transfers so that it is equal to or less than the maximum packet size of each endpoint.

1. Wait for the UIFR1/EP2READY flag to be set.
2. DMAC settings for EP2 data transfer (such as auto-request and address setting). Read value of UESZ2 and specify number of transfers to match size of received data (64 bytes or less).
3. Activate the DMAC (write 1 to DTE after reading DTE as 0).
4. DMA transfer (transfer of 64 bytes or less).
5. Write 1 to the UTRG0/EP2RDFN bit after the DMA transfer is completed.
6. Repeat steps 1 to 5 above.

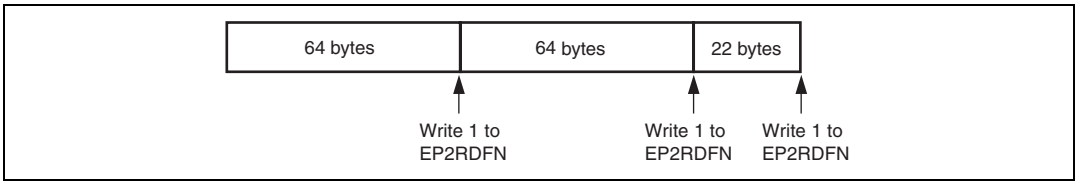
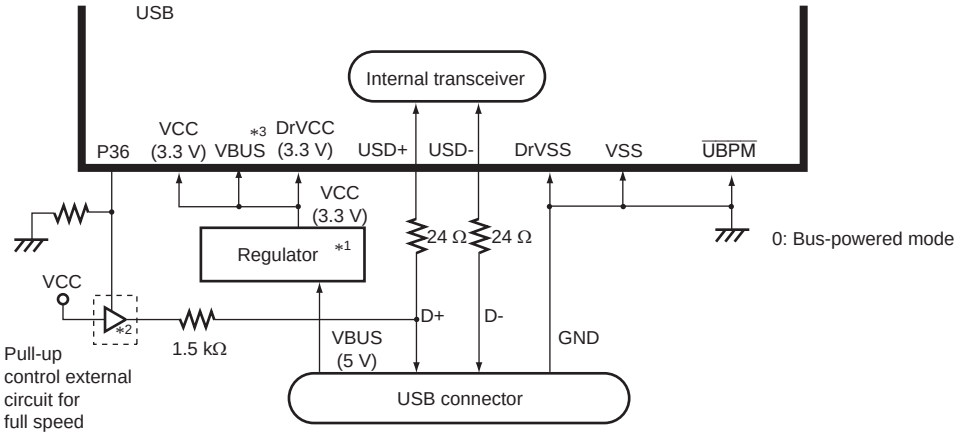



**Figure 14.25 EP2RDFN Operation in UTRG0 (Auto-Request)**

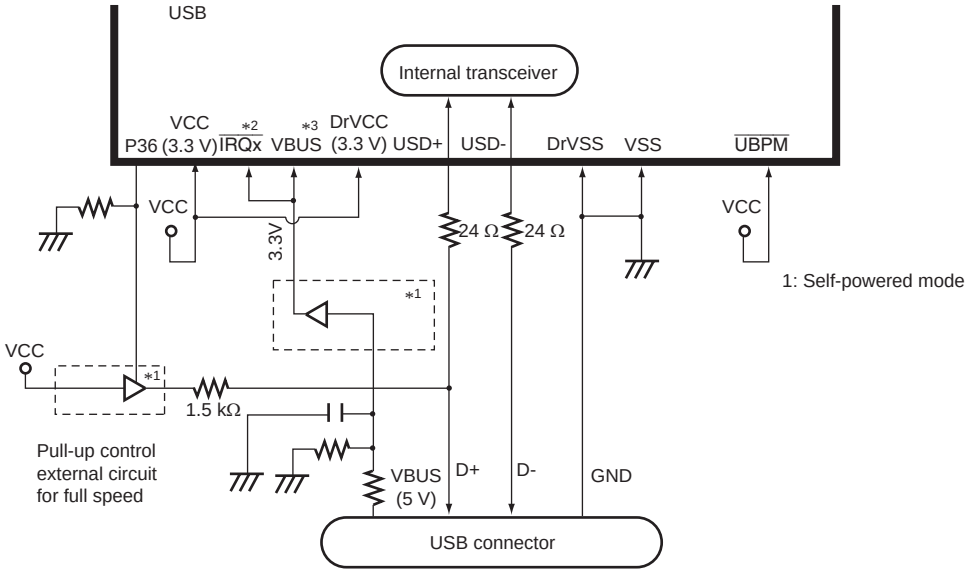
## 14.7 USB External Circuit Example

Figures 14.26 and 14.27 show the USB external circuit examples of this LIS.




Notes: 1. Step-down to the operating voltage VCC (3.3 V) of this LSI.
2. To protect the LSI, voltage applicable IC such as HD74LV-A series must be used even when the system power is turned off.
3. Prevent the VBUS pin from being affected by noise while USB is in communication.

**Figure 14.26 USB External Circuit in Bus-Powered Mode**




Notes: 1. To protect the LSI, voltage applicable IC such as HD74LV-A series must be used even when the system power is turned off.
2. To cancel power-down mode by detecting the USB cable connection, theVBUS signal must be connected to the IRQx pin. Note that power-down mode state cannot be canceled by the USB interrupt EXIRQx.
3. Prevent the VBUS pin from being affected by noise while USB is in communication.

**Figure 14.27 USB External Circuit in Self-Powered Mode**

## 14.8 Usage Notes

### 14.8.1 Emulator Usage Notes

1. If UEDR0o and UEDR2 are displayed using the I/O register window function, or the like, the EP0o FIFO or EP2 FIFO read pointer will not operate properly, preventing UEDR0o, UESZ0o, UEDR2, and UESZ2 from being read correctly. Therefore, UEDR0o and UEDR2 should not be displayed.
2. In the E6000, since the USB module is mounted on the external extended board and accessed as an external module, there are some limitations as shown below. These limitations do not apply to the E10A or to product chips.

- USB operation is not supported in the H8S/2218 Group's mode 7 (single-chip mode).
- When using the USB module in the H8S/2218 Group's mode 6 (on-chip ROM-enabled mode) or the H8S/2212 Group's mode 7 (single-chip mode), $\overline{CS6}$ and A9 to A0 are input pins in the initial status. Therefore, $\overline{CS6}$ and A9 to A0 must be set as output pins (= B'0010) by setting P72DDR to 1, AE3 to AE0 to B'0010, and PC7DDR to PC0DDR to H'FF before accessing the USB module.
- When using the USB module in the H8S/2218 Group's modes 4 and 5 (on-chip ROM-disabled mode), $\overline{CS6}$ and A9 to A8 must be set as output pins by setting P72DDR to 1 and AE3 to AE0 to B'0010.

### 14.8.2 Bus Interface

The USB module's interface is based on the bus specifications of external area 6. Accordingly, before accessing the USB module, area 6 must be specified as having an 8-bit bus width and 3-state access using the bus controller register.

Address H'C00100 to H'DFFFFF is for USB reserved area and thus access prohibited.

### 14.8.3 Operating Frequency

The main clock of this LSI must be 24 MHz or 16 MHz. This 24-MHz main clock, used as base clock, is doubled in the on-chip PLL circuit or this 16-MHz main clock, also used as base clock, is tripled in the on-chip PLL circuit, to generate the 48-MHz USB operating clock. Since the USB module does not support medium-speed mode, sleep mode, watch mode, subactive mode, and subsleep mode, make sure to use full-speed mode.

### 14.8.4 Setup Data Reception

The following must be noted for the EP0s FIFO used to receive 8-byte setup data. The USB is designed to always receive setup commands. Accordingly, write from the UDC has higher priority than read from the LSI. If the reception of the next setup command starts while the LSI is reading data after completing reception, this data read from the LSI is forcibly cancelled and the next setup command write starts. After the next setup command write, data read from the LSI is thus undefined. Read operation is forcibly disabled because data cannot be guaranteed if DP-RAM used as FIFO accesses the same address for write and read.

### 14.8.5 FIFO Clear

If the USB cable is disconnected during communication, old data may be contained in the FIFO. Accordingly, FIFOs must be cleared immediately after USB cable connection. In addition, after bus reset, all FIFOs must also be cleared. Note, however, that FIFOs that are currently used for data transfer to or from the host must not be cleared.

### 14.8.6 $\overline{\text{IRQ6}}$ Interrupt

A suspend/resume interrupt requested by $\overline{\text{IRQ6}}$ must be specified as falling-edge sensitive.

### 14.8.7 Data Register Overread or Overwrite

When the CPU reads or writes to data registers, the following must be noted:

- Transmit data registers (UEDR0i, UEDR3, UEDR1)

  Data to be written to the transmit data registers must be within the maximum packet size. For the transmit data register of EP1 having a dual-FIFO configuration, data to be written at any time must be within the maximum packet size. In this case, after a data write, the FIFO is switched to the other FIFO, enabling an further data write, when the PKTE bit in UTRG0 is set to 1. Accordingly, data of size corresponding to two FIFOs must not be written to the transmit data registers at a time.
- Receive data registers (UEDR0o, UEDR2)

  Receive data registers must not read a data size that is greater than the effective size of the read data item. In other words, receive data registers must not read data with data size larger than that specified by the receive data size register. For the receive data register of EP2 having a dual-FIFO configuration, data to be read at any time must be within the maximum packet size. In this case, after reading the currently selected FIFO, set the RDFN bit in UTRG to 1. This switches the FIFO to the other FIFO and updates the receive data size, enabling the next data read. In addition, if there is no receive data in a FIFO, data must not be read. Otherwise, the

pointer that controls the internal module FIFO is updated and correct operation cannot be guaranteed.

### 14.8.8 Reset

— The manual reset during USB communication operations must not be executed, since the LSI may stop with the state of USD+ and USD- pins maintained.

This USB module uses synchronous reset for some registers. The reset state of these registers must be cancelled after the clock oscillation stabilization time has passed. At initialization, reset must be cancelled using the following procedure:

1. Cancel the USB module stop 1: Clear the USBSTOP1 bit in EXMDLSTP to 0.
2. Select the USB operating clock: Write 1 to the UCKS3 to UCKS0 bits in UCTLR.
3. Cancel the USB module stop 2: Clear the MSTPB0 bit in MSTPCRB to 0.
4. Wait for the USB operating clock stabilization: Wait until the CK48READY bit in UIFR3 is set to 1.
5. Cancel the USB interface reset state: Clear the UIFRST bit in UCTLR to 0.
6. Cancel the UDC core reset state: Clear the UDCRST bit in UCTLR to 0.

For details, see the flowcharts in section 14.5.1, Initialization and section 14.5.2, USB Cable Connection/Disconnection.

— The USB registers are not initialized when the watchdog timer (WDT) triggers a power-on reset. Therefore, the USB may not operate properly after a power-on reset is triggered by the WDT due to CPU runaway or a similar cause. (If a power-on reset is triggered by input of a power-on reset signal from the $\overline{\text{RES}}$ pin, the USB registers are initialized and there is no problem.) Consequently, an initialization routine should be used to write the initial values listed below to the following three registers, thereby ensuring that all the USB registers are properly initialized, immediately following a reset.

UCTLR = H'03, UIER3 = H'80, UIFR3 = H'00

### 14.8.9 EP0 Interrupt Sources Assignment

EP0 interrupt sources assigned to bits 3 to 0 in UIFR0 must be assigned to the same interrupt signal ($\overline{\text{EXIRQx}}$) by setting UISR0. There are no other restrictions on interrupt sources.

### 14.8.10 Level Shifter for VBUS and $\overline{\text{IRQx}}$ Pins

The VBUS and $\overline{\text{IRQx}}$ pins of this USB module must be connected to the USB connector's VBUS pin via a level shifter. This is because the USB module has a circuit that operates by detecting USB cable connection or disconnection.

Even if the power of the device incorporating this USB module is turned off, 5-V power is applied to the USB connector's VBUS pin while the USB cable is connected to the device set. To protect the LSI from destruction, use a level shifter such as the HD74LV-A Series, which allows voltage application to the pin even when the power is off.

### 14.8.11 USB Endpoint Data Read and Write

To write data to an USB endpoint data register (UEDR0i, UEDR1, or UEDR3) on the transmit side using a CPU word or longword transfer instruction, the size of data to be written must be smaller than the size of data that is to be transmitted.

For example, when 7-byte data is transferred to the host, 8-byte data is sent to the host if data is written twice by the longword transfer instructions or if data is written four times by the word transfer instructions. To write 7-byte data correctly, data must be written once by a longword transfer instruction, once by a word transfer instruction, and once by a byte transfer instruction, or data must be written three times by a word transfer instruction and once by a byte transfer instruction.

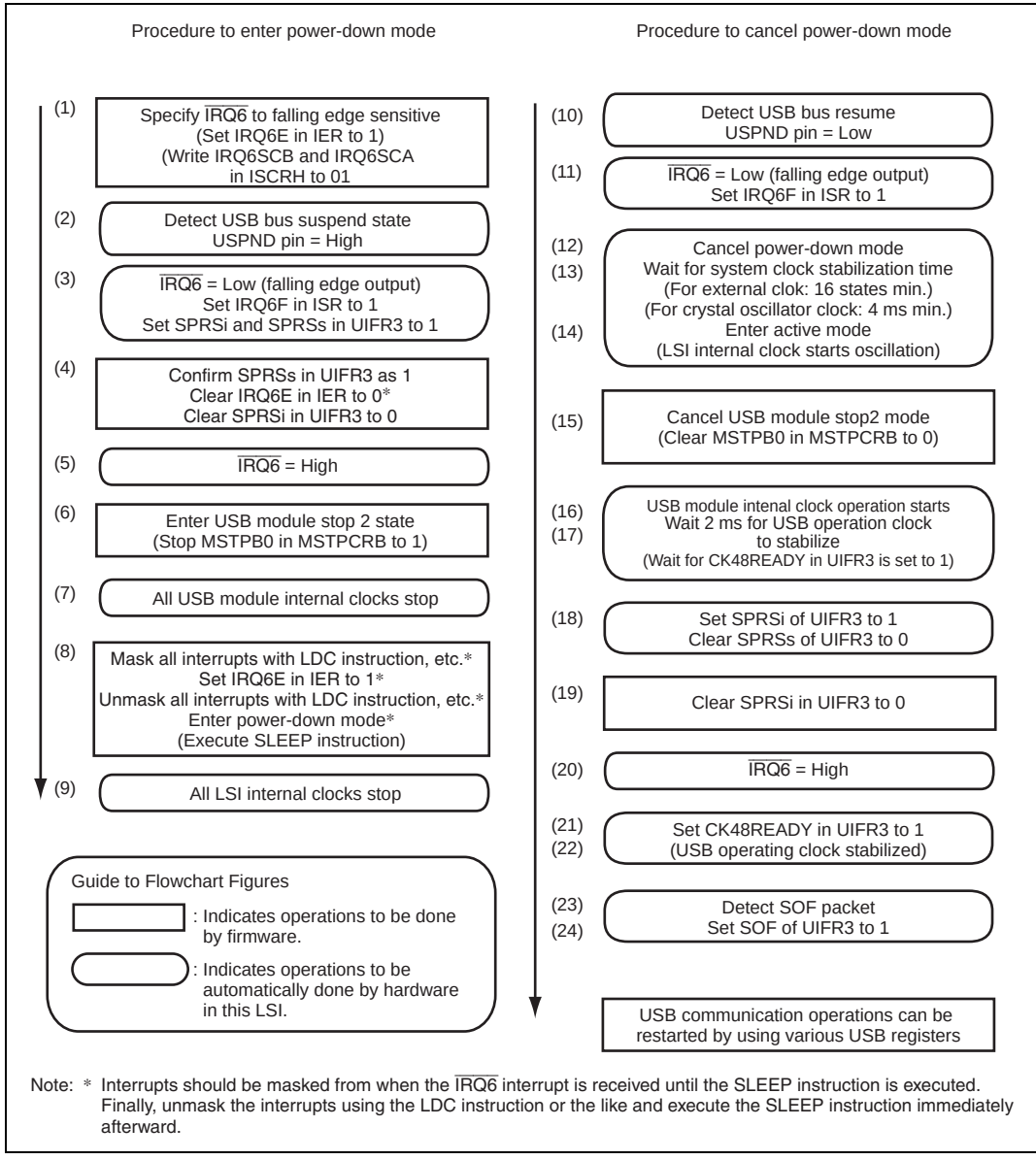
To read data from the USB endpoint data register (UEDR0o or UEDR2) on the receive side, the correct size of data must be read. In this case, the data size is specified by the USB endpoint receive size register (UESZ0o or UESZ2).

To execute DMA transfer on data in the USB endpoint data register using the on-chip DMAC, byte transfer musts be used. In word transfer, odd-byte data cannot be transferred. Word transfer is thus disabled.

### 14.8.12 Restrictions on Entering and Canceling Power-Down Mode

Before entering the power-down mode, set the USB module stop 2 state. The UDC core must not be reset.

To access the USB module after canceling power-down mode, cancel the USB module stop 2 state and wait for the USB operating clock (48 MHz) stabilization time.




Note: * Interrupts should be masked from when the IRQ6 interrupt is received until the SLEEP instruction is executed. Finally, unmask the interrupts using the LDC instruction or the like and execute the SLEEP instruction immediately afterward.

**Figure 14.28 Flowchart**

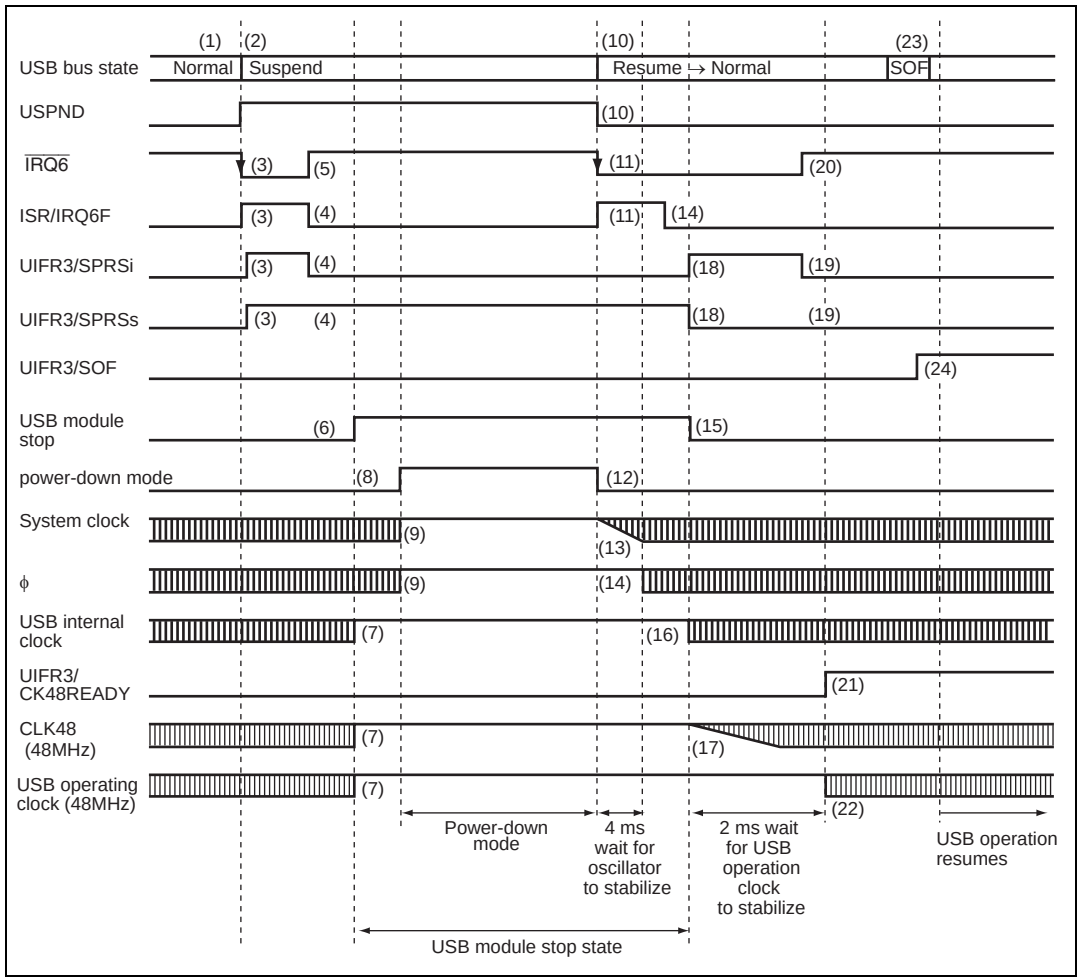



**Figure 14.29 Timing Chart**

### 14.8.13 USB External Circuit Example

The USB external circuit examples are used for reference only. In actual board design, carefully check the system operation.

In addition, the USB external circuit examples cannot guarantee the correct system operation. The user must individually take measures against external surges or ESD noise by incorporating protective diodes or other components if necessary.

### 14.8.14 Pin Processing when USB Not Used

Pin processing should be performed as follows.

DrVCC = VCC, DrVSS = 0 V, USD+ = USD– = USPND = open state, VBUS = $\overline{\text{UBPM}}$ = 0 V

### 14.8.15 Notes on TR Interrupt

Note the following when using the transfer request interrupt (TR interrupt) for IN transfer to EP0i, EP1, or EP3.

The TR interrupt flag is set if the FIFO for the target EP has no data when the IN token is sent from the USB host. However, at the timing shown in figure 14.30, multiple TR interrupts occur successively. Take appropriate measures against malfunction in such a case.

Note: This module determines whether to return NAK if the FIFO of the target EP has no data when receiving the IN token, but the TR interrupt flag is set only after a NAK handshake is sent. If the next IN token is sent before PKTE of UTRG0 is written to, the TR interrupt flag is set again.

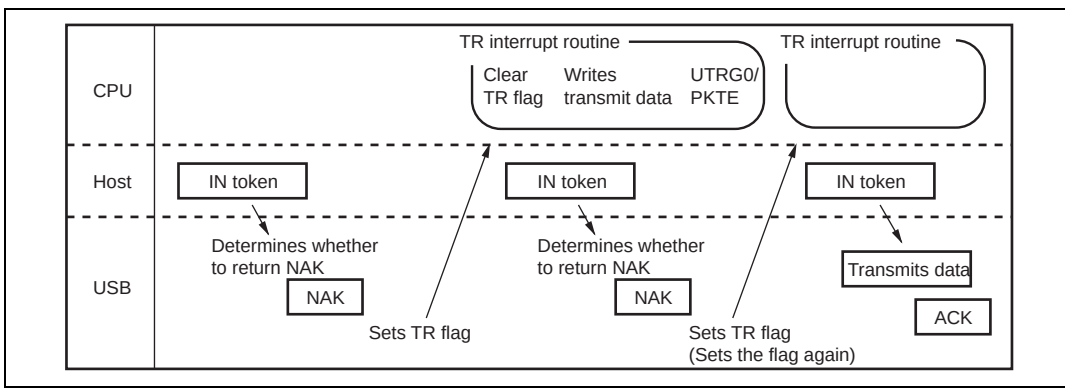



**Figure 14.30 TR Interrupt Flag Set Timing**

### 14.8.16 Clearing the FIFO when DMA Transfer Is Enabled

When DMA transfer is enabled (EP2T1 = 1 and EP2T0 = 0 or 1 in UDMAR) at endpoint 2, it is not possible to clear OUTFIFO in EP2. It is necessary to disable DMA transfer (EP2T1 = 0 and EP2T0 = 0 in UDMAR) before clearing the FIFO.

# Section 15 A/D Converter

This LSI includes a successive approximation type 10-bit A/D converter that allows up to six analog input channels to be selected. The block diagram of the A/D converter is shown in figure 15.1.

## 15.1 Features

- 10-bit resolution
- Six input channels
- Conversion time: 8.1 µs per channel (at 16-MHz operation), 10.7 µs per channel (at 24-MHz operation), 21.8 µs per channel (at 6-MHz operation)
- Two operating modes
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels
- Four data registers
  - Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three methods conversion start
  - Software
  - Timer (TPU) conversion start trigger
  - External trigger signal ($\overline{\text{ADTRG}}$)
- Interrupt request
  - An A/D conversion end interrupt request (ADI) can be generated
- Module stop mode can be set
- Settable analog conversion voltage range

  Analog conversion voltage range settable using the reference voltage pin (Vref) as the reference voltage

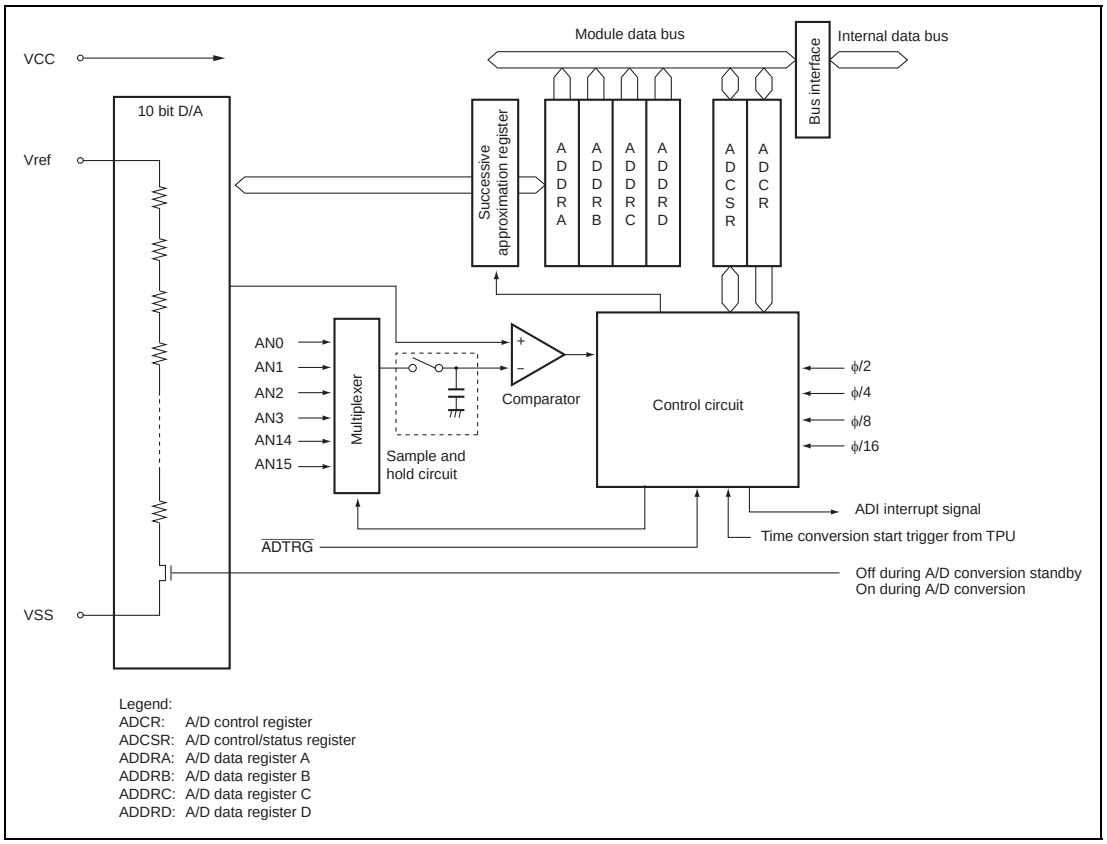



**Figure 15.1 Block Diagram of A/D Converter**

## 15.2 Input/Output Pins

Table 15.1 summarizes the input pins used by the A/D converter. The AN0 to AN3 and AN14 to AN15 pins are analog input pins. The VCC and VSS pins are the power supply pins for the analog block in the A/D converter. The Vref pin is the reference voltage pin for the A/D conversion.

**Table 15.1 Pin Configuration**

| Pin Name | Symbol | I/O | Function |
|---|---|---|---|
| Power supply pin | VCC | Input | Analog block power supply and reference voltage (also used for digital block) |
| Ground pin | VSS | Input | Analog block ground and reference voltage (also used for digital block) |
| Reference voltage pin | Vref | Input | Reference voltage pin for A/D conversion |
| Analog input pin 0 | AN0 | Input | Analog input pins |
| Analog input pin 1 | AN1 | Input | |
| Analog input pin 2 | AN2 | Input | |
| Analog input pin 3 | AN3 | Input | |
| Analog input pin 14 | AN14 | Input | |
| Analog input pin 15 | AN15 | Input | |
| A/D external trigger input pin | $\overline{\text{ADTRG}}$ | Input | External trigger input pin for starting A/D conversion |

## 15.3 Register Descriptions

The A/D converter has the following registers.

- A/D data register A (ADDRA)
- A/D data register B (ADDRB)
- A/D data register C (ADDRC)
- A/D data register D (ADDRD)
- A/D control/status register (ADCSR)
- A/D control register (ADCR)

### 15.3.1 A/D Data Registers A to D (ADDRA to ADDRD)

There are four 16-bit read-only ADDR registers; ADDRA to ADDRD, used to store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 15.2.

The converted 10-bit data is stored in bits 6 to 15. The lower 6 bits are always read as 0.

The data bus between the CPU and the A/D converter is 8 bits wide. The upper byte can be read directly from the CPU, however the lower byte should be read via a temporary register. The temporary register contents are transferred from the ADDR when the upper byte data is read. When reading the ADDR, read the upper byte before the lower byte, or read in word unit. The initial value of the ADDR is H'0000.

**Table 15.2 Analog Input Channels and Corresponding ADDR Registers**

| Analog Input Channel | A/D Data Register to Be Stored the Results of A/D Conversion |
|---|---|
| AN0 | ADDRA |
| AN1 | ADDRB |
| AN2, AN14 | ADDRC |
| AN3, AN15 | ADDRD |

### 15.3.2 A/D Control/Status Register (ADCSR)

ADCSR controls A/D conversion operations.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | ADF | 0 | R/(W)* | A/D End Flag<br>A status flag that indicates the end of A/D conversion.<br>[Setting conditions]<br>• When A/D conversion ends in single mode<br>• When A/D conversion ends on all channels specified in scan mode<br>[Clearing conditions]<br>• When 0 is written after reading ADF = 1<br>• When DMAC is activated by an ADI interrupt and ADDR is read |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 6 | ADIE | 0 | R/W | A/D Interrupt Enable<br>A/D conversion end interrupt (ADI) request enabled when 1 is set. |
| 5 | ADST | 0 | R/W | A/D Start<br>Clearing this bit to 0 stops A/D conversion, and the A/D converter enters the wait state.<br>Setting this bit to 1 starts A/D conversion. It can be set to 1 by software, the timer conversion start trigger, and the A/D external trigger ($\overline{\text{ADTRG}}$). In single mode, this bit is cleared to 0 automatically when conversion on the specified channel is complete. In scan mode, conversion continues sequentially on the specified channels until this bit is cleared to 0 by software, a reset, a transition to standby mode, or module stop mode. |
| 4 | SCAN | 0 | R/W | Scan Mode<br>Selects single mode or scan mode as the A/D conversion operating mode.<br>0: Single mode<br>1: Scan mode |
| 3 | CH3 | 0 | R/W | Channel Select 3 to 0<br>Select analog input channels. |
| 2 | CH2 | 0 | R/W | |
| 1 | CH1 | 0 | R/W | |
| 0 | CH0 | 0 | R/W | |

| When SCAN = 0 | When SCAN = 1 |
|---|---|
| 0000: AN0 | 0000: AN0 |
| 0001: AN1 | 0001: AN0 to AN1 |
| 0010: AN2 | 0010: AN0 to AN2 |
| 0011: AN3 | 0011: AN0 to AN3 |
| 01××: Setting prohibited | 01xx: Setting prohibited |
| 10××: Setting prohibited | 1xxx: Setting prohibited |
| 11××: Setting prohibited | |
| 1110: AN14 | |
| 1111: AN15 | |
| Legend: | |
| ×: Don't care | |

Note: * The write value should always be 0 to clear this flag.

### 15.3.3 A/D Control Register (ADCR)

The ADCR enables A/D conversion started by an external trigger signal.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | TRGS1 | 0 | R/W | Timer Trigger Select 1 and 0 |
| 6 | TRGS0 | 0 | R/W | Enables the start of A/D conversion by a trigger signal. Only set bits TRGS1 and TRGS0 while conversion is stopped (ADST = 0).<br>00: A/D conversion start by software<br>01: A/D conversion start by TPU<br>10: Setting prohibited<br>11: A/D conversion start by external trigger pin ($\overline{\text{ADTRG}}$) |
| 5, 4 | — | All 1 | — | Reserved<br>These bits are always read as 1 cannot be modified. |
| 3 | CKS1 | 0 | R/W | Clock Select 1 and 0 |
| 2 | CKS0 | 0 | R/W | These bits specify the A/D conversion time. The conversion time should be changed only when ADST = 0.<br>00: Conversion time = 530 states (Max.)<br>01: Conversion time = 266 states (Max.)<br>10: Conversion time = 134 states (Max.)<br>11: Conversion time = 68 states (Max.)<br>The conversion time setting should exceed the conversion time shown in section 22.6, A/D Converter Characteristics. |
| 1, 0 | — | All 1 | — | Reserved<br>These bits are always read as 1, and only 1 should be written to them. |

## 15.4 Interface to Bus Master

ADDRA to ADDRD are 16-bit registers. As the data bus to the bus master is 8 bits wide, the bus master accesses to the upper byte of the registers directly while to the lower byte of the registers via the temporary register (TEMP).

Data in ADDR is read in the following way: When the upper-byte data is read, the upper-byte data will be transferred to the CPU and the lower-byte data will be transferred to TEMP. Then, when the lower-byte data is read, the lower-byte data will be transferred to the CPU.

When data in ADDR is read, the data should be read from the upper byte and lower byte in the order. When only the upper-byte data is read, the data is guaranteed. However, when only the lower-byte data is read, the data is not guaranteed.

Figure 15.2 shows data flow when accessing to ADDR.

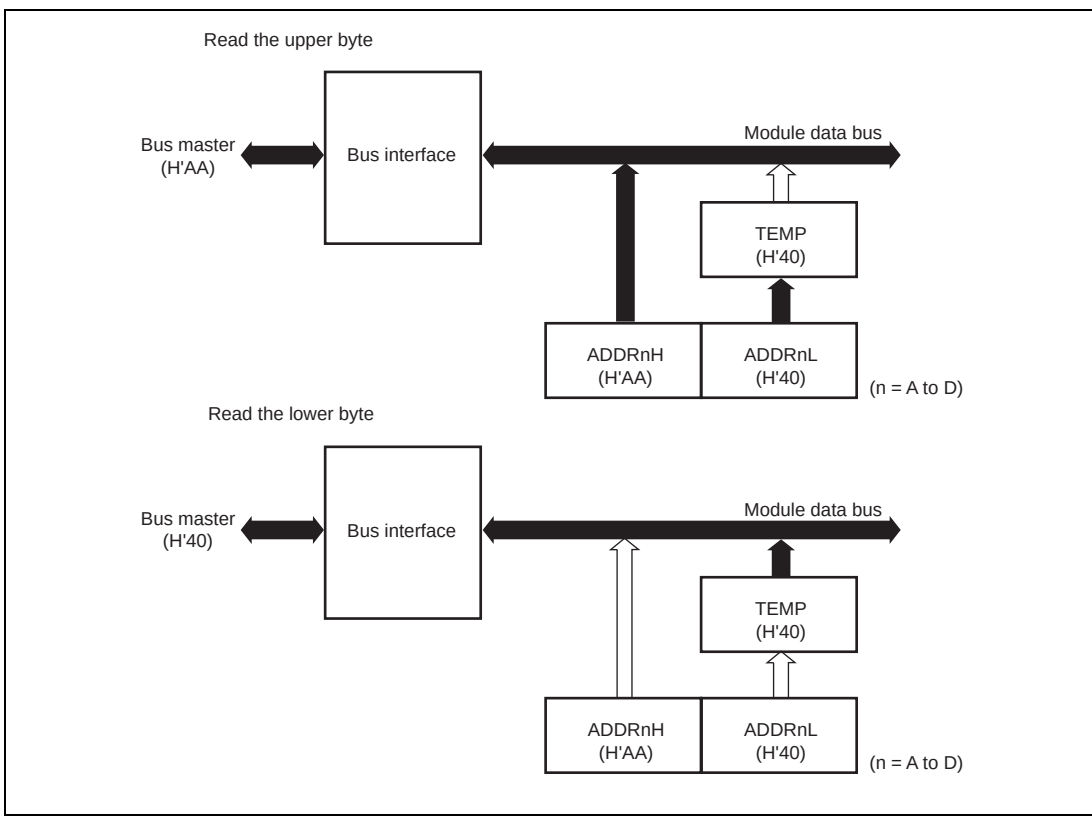



**Figure 15.2 Access to ADDR (When Reading H'AA40)**

## 15.5 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes; single mode and scan mode. When changing the operating mode or analog input channel, in order to prevent incorrect operation, first clear the bit ADST to 0 in ADCSR. The ADST bit can be set at the same time as the operating mode or analog input channel is changed.

### 15.5.1 Single Mode

In single mode, A/D conversion is to be performed only once on the specified single channel. The operations are as follows.

1. A/D conversion is started when the ADST bit is set to 1, according to software or external trigger input.
2. When A/D conversion is completed, the result is transferred to the corresponding A/D data register to the channel.
3. On completion of conversion, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion. When A/D conversion ends, the ADST bit is automatically cleared to 0 and the A/D converter enters the wait state.

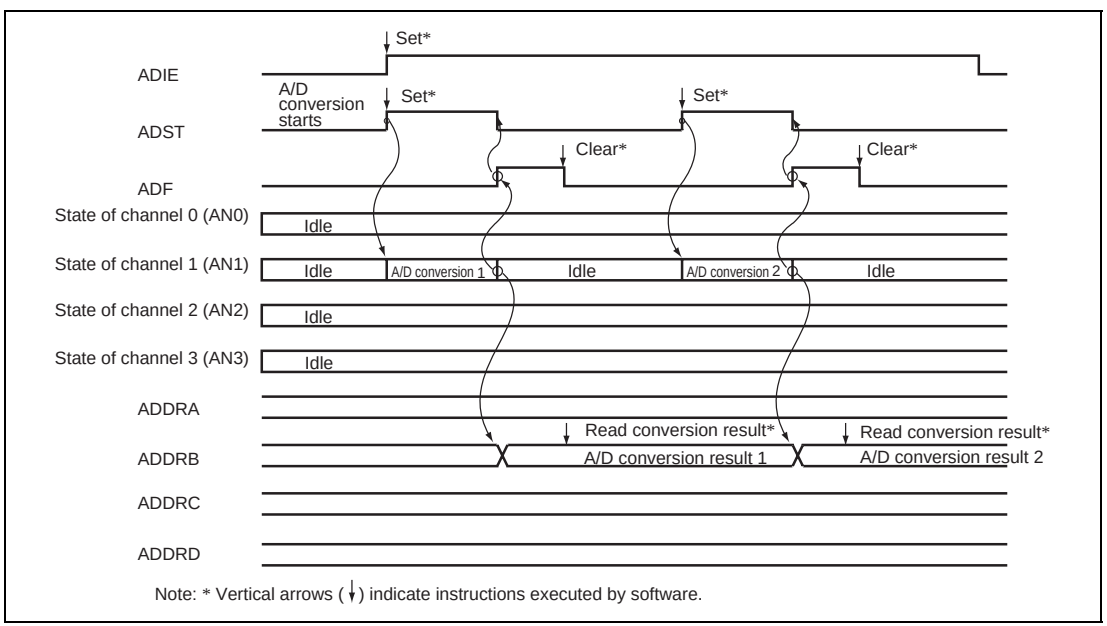
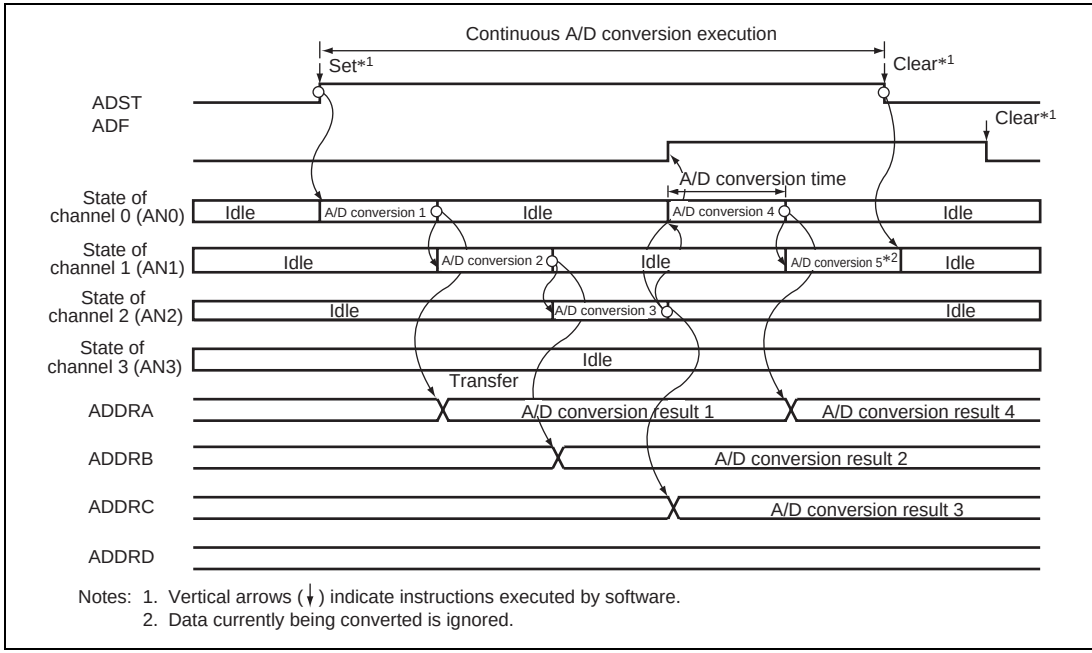



**Figure 15.3 A/D Conversion Timing (Single-Chip Mode, Channel 1 Selected)**

### 15.5.2 Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the specified channels (four channels maximum). The operations are as follows.

1. When the ADST bit is set to 1 by software, TPU or external trigger input, A/D conversion starts on the first channel in the group (AN0 when CH3 and CH2 = 00, AN4 when CH3 and CH2 = 01, or AN8 when CH3 and CH2 = 10).
2. When A/D conversion for each channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.
3. When conversion of all the selected channels is completed, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt is requested after A/D conversion ends. Conversion of the first channel in the group starts again.
4. Steps 2 to 3 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops.




Notes: 1. Vertical arrows (↓) indicate instructions executed by software.
2. Data currently being converted is ignored.

**Figure 15.4 A/D Conversion Timing (Scan Mode, Channels AN0 to AN2 Selected)**

### 15.5.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input when the A/D conversion start delay time ($t_D$) has passed after the ADST bit is set to 1, then starts conversion. Figure 15.5 shows the A/D conversion timing. Tables 15.3 and 15.4 show the A/D conversion time.

As indicated in figure 15.5, the A/D conversion time ($t_{CONV}$) includes $t_D$ and the input sampling time ($t_{SPL}$). The length of $t_D$ varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 15.4.

In scan mode, the values given in table 15.4 apply to the first conversion time. The values given in table 15.3 apply to the second and subsequent conversions.

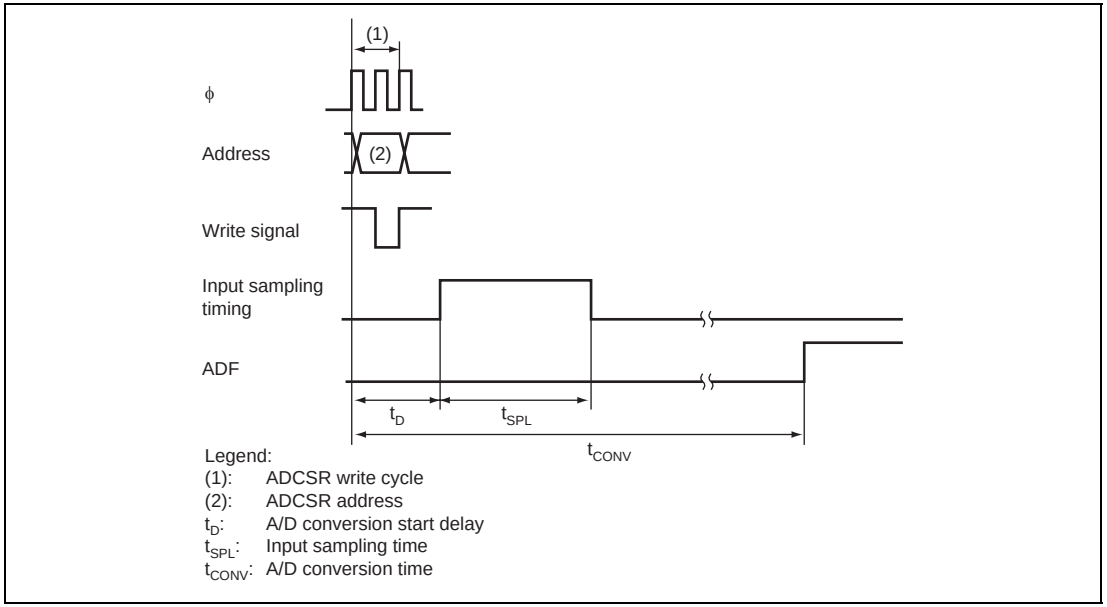



**Figure 15.5 A/D Conversion Timing**

**Table 15.3 A/D Conversion Time (Single Mode)**

| | | CKS1 = 0 | | | | | | CKS1 = 1 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | CKS0 = 0 | | | CKS0 = 1 | | | CKS0 = 0 | | | CKS0 = 1 | | |
| **Item** | **Symbol** | **Min.** | **Typ.** | **Max.** | **Min.** | **Typ.** | **Max.** | **Min.** | **Typ.** | **Max.** | **Min.** | **Typ.** | **Max.** |
| A/D conversion start delay | $t_D$ | 18 | — | 33 | 10 | — | 17 | 6 | — | 9 | 4 | — | 5 |
| Input sampling time | $t_{SPL}$ | — | 127 | — | — | 63 | — | — | 31 | — | — | 15 | — |
| A/D conversion time | $t_{CONV}$ | 515 | — | 530 | 259 | — | 266 | 131 | — | 134 | 67 | — | 68 |

Note: All values represent the number of states.

**Table 15.4 A/D Conversion Time (Scan Mode)**

| CKS1 | CKS0 | Conversion Time (State) |
|---|---|---|
| 0 | 0 | 512 (Fixed) |
| | 1 | 256 (Fixed) |
| 1 | 0 | 128 (Fixed) |
| | 1 | 64 (Fixed) |

### 15.5.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS0 and TRGS1 bits are set to 11 in ADCR, external trigger input is enabled at the $\overline{\text{ADTRG}}$ pin. A falling edge at the $\overline{\text{ADTRG}}$ pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, in both single and scan modes, are the same as when the bit ADST has been set to 1 by software. Figure 15.6 shows the timing.

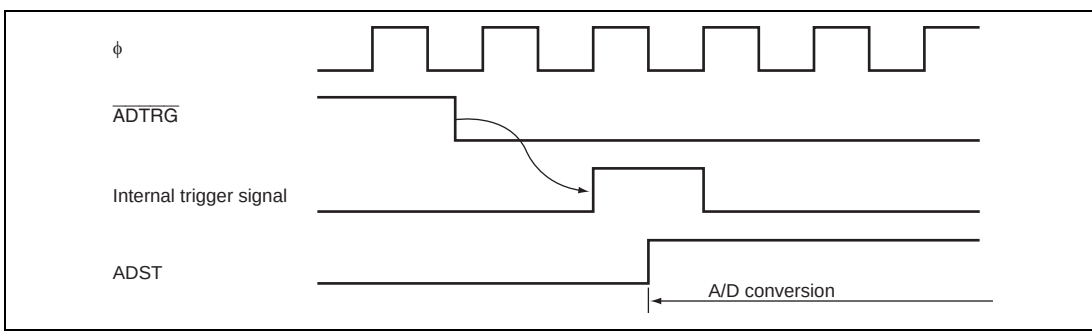



**Figure 15.6 External Trigger Input Timing**

## 15.6 Interrupts

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 enables ADI interrupt requests while the bit ADF in ADCSR is set to 1 after A/D conversion is completed. The DMAC can be activated by an ADI interrupt.

**Table 15.5 A/D Converter Interrupt Source**

| Name | Interrupt Source | Interrupt Source Flag | DMAC Activation |
|---|---|---|---|
| ADI | A/D conversion completed | ADF | Possible |

## 15.7 A/D Conversion Precision Definitions

This LSI's A/D conversion precision definitions are given below.

- Resolution

  The number of A/D converter digital output codes
- Quantization error

  The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 15.7).
- Offset error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'0000000000 (H'000) to B'0000000001 (H'001) (see figure 15.8).
- Full-scale error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 (H'3FE) to B'1111111111 (H'3FF) (see figure 15.8).
- Nonlinearity error

  The error with respect to the ideal A/D conversion characteristic between zero voltage and full-scale voltage. Does not include offset error, full-scale error, or quantization error (see figure 15.8).
- Absolute precision

  The deviation between the digital value and the analog input value. Includes offset error, full-scale error, quantization error, and nonlinearity error.

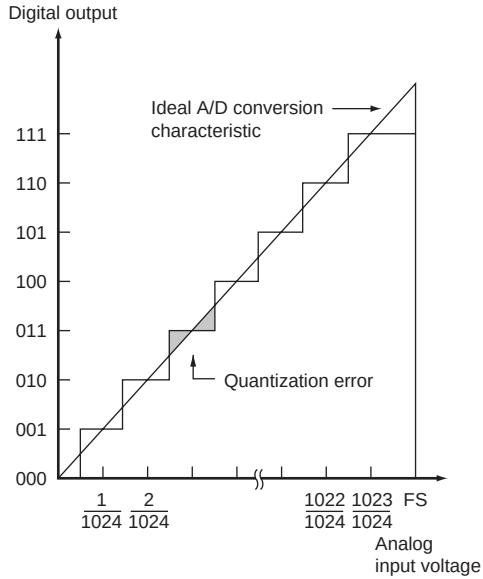



**Figure 15.7 A/D Conversion Precision Definitions (1)**

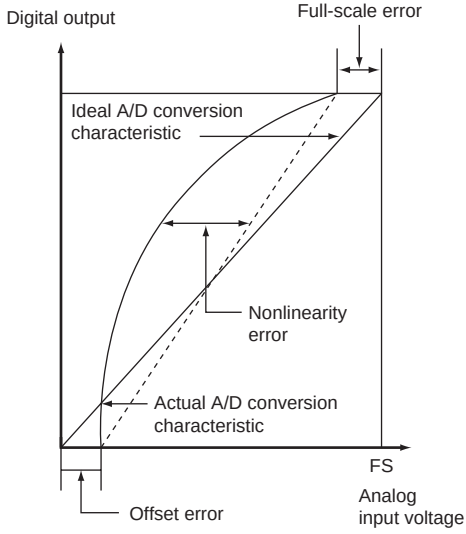



**Figure 15.8 A/D Conversion Precision Definitions (2)**

## 15.8 Usage Notes

### 15.8.1 Module Stop Mode Setting

Operation of the A/D converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the A/D converter to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 20, Power-Down Modes.

### 15.8.2 Permissible Signal Source Impedance

This LSI's analog input is designed such that conversion precision is guaranteed for an input signal for which the signal source impedance is 5 kΩ or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 5 kΩ, charging may be insufficient and it may not be possible to guarantee A/D conversion precision. However, for A/D conversion in single mode with a large capacitance provided externally, the input load will essentially comprise only the internal input resistance of 10 kΩ, and the signal source impedance is ignored. However, as a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (e.g., 5 mV/Ωs or greater) (see figure 15.9). When converting a high-speed analog signal, a low-impedance buffer should be inserted.

### 15.8.3 Influences on Absolute Precision

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute precision. Be sure to make the connection to an electrically stable GND such as AVSS.

Care is also required to insure that filter circuits do not communicate with digital signals on the mounting board (i.e., acting as antennas).

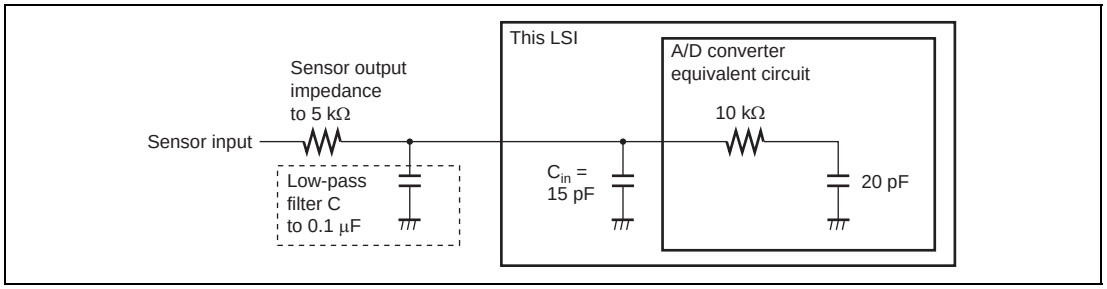



**Figure 15.9 Example of Analog Input Circuit**

### 15.8.4 Range of Analog Power Supply and Other Pin Settings

If the conditions below are not met, the reliability of the device may be adversely affected.

- Analog input voltage range
  The voltage applied to analog input pin ANn during A/D conversion should be in the range VSS ≤ ANn ≤ Vref.
- Vref input range
  The analog reference voltage input at the Vref pin set is the range Vref ≤ Vcc.

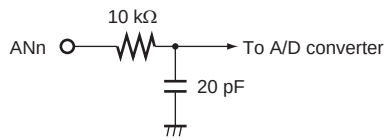
### 15.8.5 Notes on Board Design

Careful consideration is required in board design for noise countermeasures and in order to prevent damage due to abnormal voltage, such as an excessive surge at the analog input pins (AN0 to AN3 or AN14 to AN15) and analog reference voltage pin (Vref).

**Table 15.6 Analog Pin Specifications**

| Item | Min. | Max. | Unit |
|---|---|---|---|
| Analog input capacitance | — | 20 | pF |
| Permissible signal source impedance | — | 5* | kΩ |

Note: * Vcc = 2.7 to 3.6 V




Note: Values are reference values.

**Figure 15.10 Analog Input Pin Equivalent Circuit**

# Section 16 RAM

The HD64F2218, HD64F2218U, and HD64F2218CU have 12 kbytes of on-chip high-speed static RAM. The HD6432217, HD64F2211, HD64F2211U, and HD64F2211CU have 8 kbytes of on-chip high-speed static RAM. The HD6432210 and HD6432210S have 4 kbytes of on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data.

The on-chip RAM can be enabled or disabled by means of the RAM enable bit (RAME) in the system control register (SYSCR). For details on SYSCR, refer to section 3.2.2, System Control Register (SYSCR).

| Product Class | | ROM Type | RAM Size | RAM Address |
|---|---|---|---|---|
| H8S/2218 Group | HD64F2218<br>HD64F2218U<br>HD64F2218CU | Flash memory Version | 12 kbytes | H'FFC000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD64F2217CU | | | |
| | HD6432217 | Masked ROM Version | 8 kbytes | H'FFD000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| H8S/2212 Group | HDF64F2212<br>HDF64F2212U<br>HDF64F2212CU | Flash memory Version | 12 kbytes | H'FFC000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD64F2211<br>HD64F2211U<br>HD64F2211CU | | 8 kbytes | H'FFD000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD64F2210CU | | | |
| | HD6432211 | Masked ROM Version | 8 kbytes | H'FFD000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |
| | HD6432210<br>HD6432210S | | 4 kbytes | H'FFE000 to H'FFEFBF<br>H'FFFFC0 to H'FFFFFF |

# Section 17 Flash Memory (F-ZTAT Version)

The features of the on-chip flash memory are summarized below. The block diagram of the flash memory is shown in figure 17.1.

## 17.1 Features

- Size:

| Product Class | | ROM Size | ROM Address |
|---|---|---|---|
| H8S/2218 Group | HD64F2218, HD64F2218U<br>HD64F2218CU | 128 kbytes | H'000000 to H'01FFFF<br>(Modes 6 and 7) |
| | HD64F2217CU | 64 kbytes | H'000000 to H'00FFFF<br>(Modes 6 and 7) |
| H8S/2212 Group | HD64F2212, HD64F2212U<br>HD64F2212CU | 128 kbytes | H'000000 to H'01FFFF<br>(Mode 7) |
| | HD64F2211, HD64F2211U<br>HD64F2211CU | 64 kbytes | H'000000 to H'00FFFF<br>(Mode 7) |
| | HD64F2210CU | 32 kbytes | H'000000 to H'007FFF<br>(Mode 7) |

- Programming/erase methods
  - The flash memory is programmed 128 bytes at a time. Erase is performed in single-block units. The flash memory is configured as follows: 32 kbytes × 2 blocks, 28 kbytes × 1 block, 16 kbytes × 8 blocks, 8 kbytes × 1 block, and 1 kbyte × 4 blocks. To erase the entire flash memory, each block must be erased in turn.
- Reprogramming capability
  - Flash memory can be reprogrammed a minimum of 100 times.
- Two flash memory operating modes
  - Boot mode

    SCI boot mode: HD64F2218, HD64F2212, and HD64F2211

    USB boot mode: HD64F2218U, HD64F2218CU, HD64F2217CU, HD64F2212U, HD64F2212CU, HD64F2211U, HD64F2211CU and HD64F2210CU
  - User program mode

    On-board programming/erasing can be done in boot mode in which the boot program built into the chip is started for erase or programming of the entire flash memory. In normal user program mode, individual blocks can be erased or programmed.

- Automatic bit rate adjustment
  - With data transfer in SCI boot mode, this LSI's bit rate can be automatically adjusted to match the transfer bit rate of the host.
- Programming/erasing protection
  - Sets hardware protection, software protection, and error protection against flash memory programming/erasing.
- Programmer mode
  - Flash memory can be programmed/erased in programmer mode, using a PROM programmer, as well as in on-board programming mode.
- Flash memory emulation in RAM
  - Flash memory programming can be emulated in real time by overlapping a part of RAM onto flash memory.

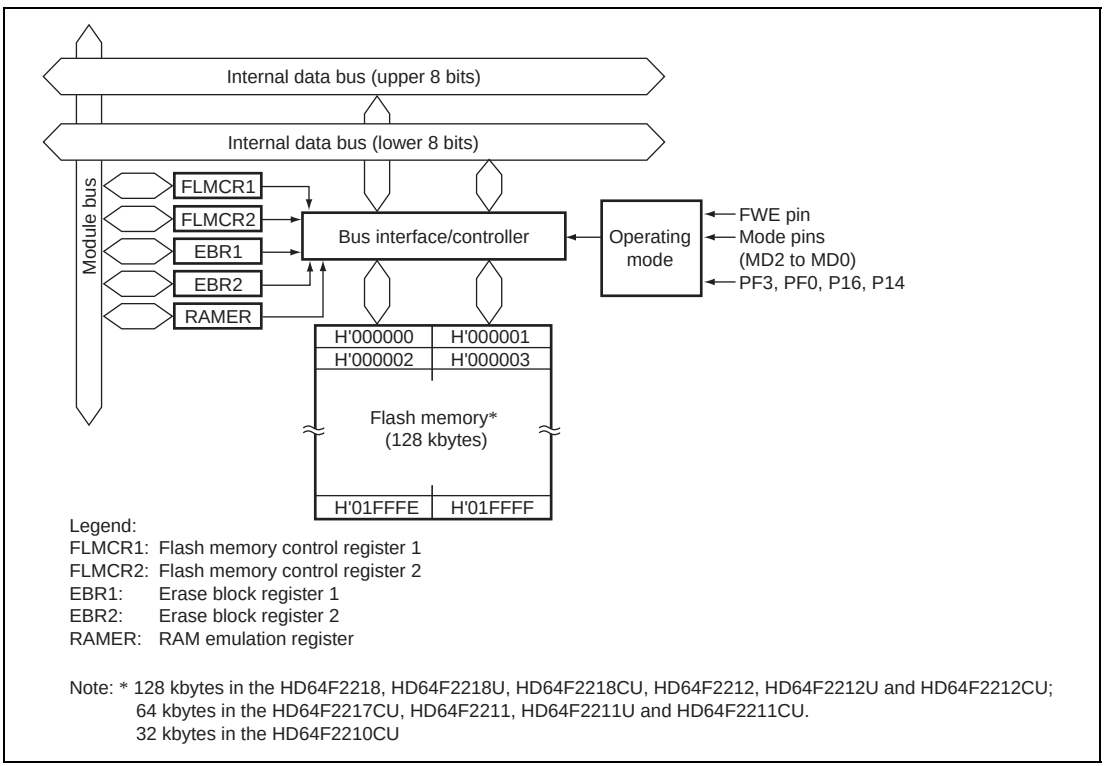
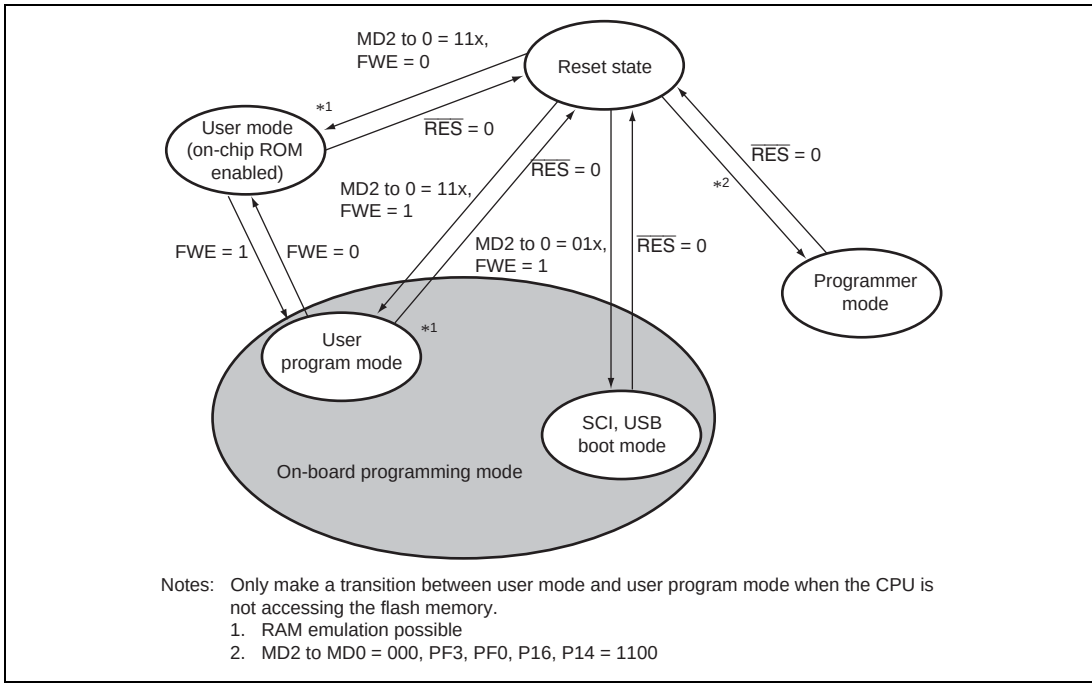



**Figure 17.1 Block Diagram of Flash Memory**

## 17.2 Mode Transitions

When the mode pins and the FWE pin are set in the reset state and a reset-start is executed, this LSI enters an operating mode as shown in figure 17.2. In user mode, flash memory can be read but not programmed or erased. The boot and user program modes are provided as modes to write and erase the flash memory.

The differences between boot mode and user program mode are shown in table 17.1. Boot mode and user program mode operations are shown in figures 17.3 and 17.4, respectively.




Notes: Only make a transition between user mode and user program mode when the CPU is not accessing the flash memory.
1. RAM emulation possible
2. MD2 to MD0 = 000, PF3, PF0, P16, P14 = 1100

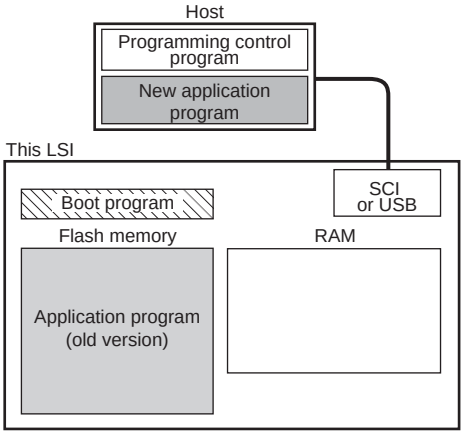
**Figure 17.2 Flash Memory State Transitions**

**Table 17.1 Differences between Boot Mode and User Program Mode**

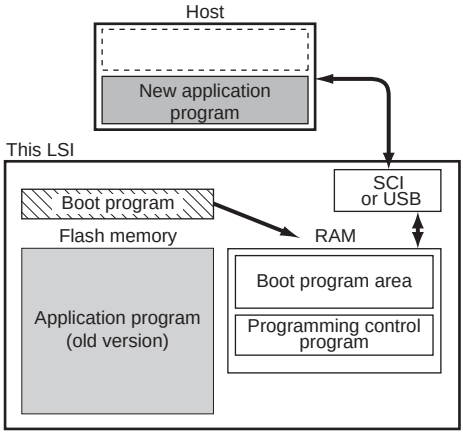
| | SCI, USB Boot Mode | User Program Mode | User Mode |
|---|---|---|---|
| Total erase | Yes | Yes | No |
| Block erase | No | Yes | No |
| Programming control program* | Program/program-verify | Erase/erase-verify<br>Program/program-verify<br>Emulation | — |

Note: * To be provided by the user, in accordance with the recommended algorithm.

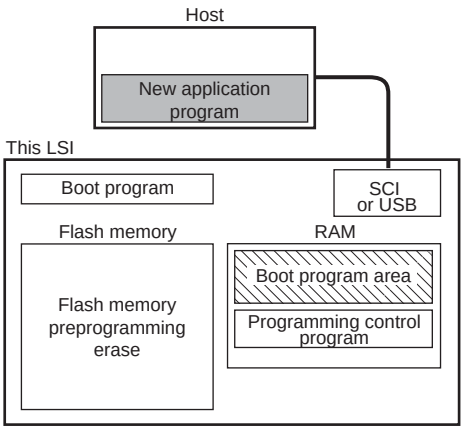
1. Initial state
   The old program version or data remains written in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.



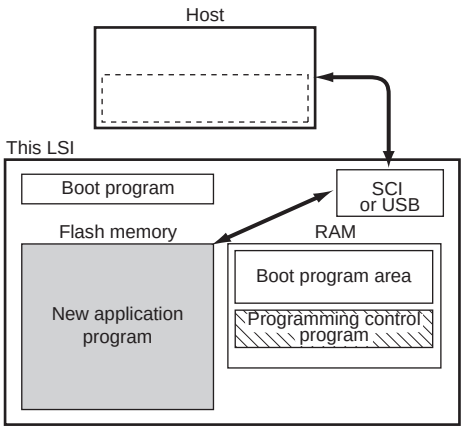

2. Programming control program transfer
   When boot mode is entered, the boot program in this LSI (originally incorporated in the chip) is started and the programming control program in the host is transferred to RAM via SCI or USB communication. The boot program required for flash memory erasing is automatically transferred to the RAM boot program area.




3. Flash memory initialization
   The erase program in the boot program area (in RAM) is executed, and the flash memory is initialized (to H'FF). In boot mode, total flash memory erasure is performed, without regard to blocks.




4. Writing new application program
   The programming control program transferred from the host to RAM is executed, and the new application program in the host is written into the flash memory.

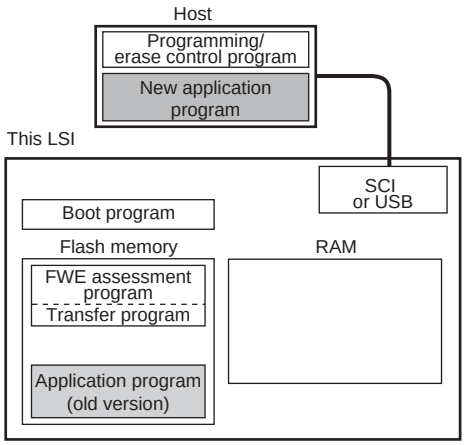


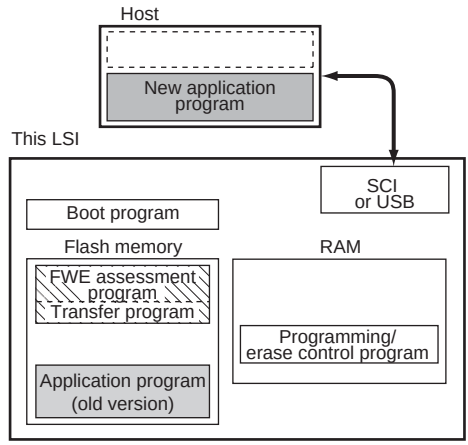

Program execution state

**Figure 17.3 Boot Mode (Sample)**

1. Initial state
   The FWE assessment program that confirms that user program mode has been entered, and the program that will transfer the programming/erase control program from flash memory to on-chip RAM should be written into the flash memory by the user beforehand. The programming/erase control program should be prepared in the host or in the flash memory.

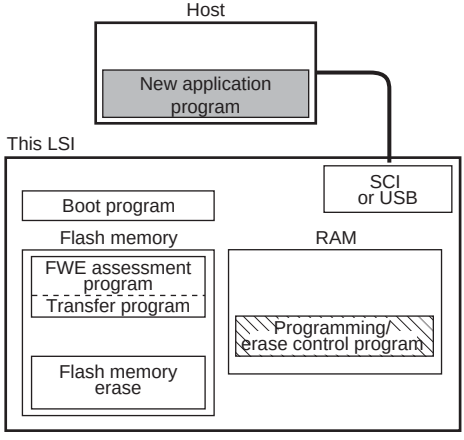
2. Programming/erase control program transfer
   When user program mode is entered, user software confirms this fact, executes transfer program in the flash memory, and transfers the programming/erase control program to RAM.

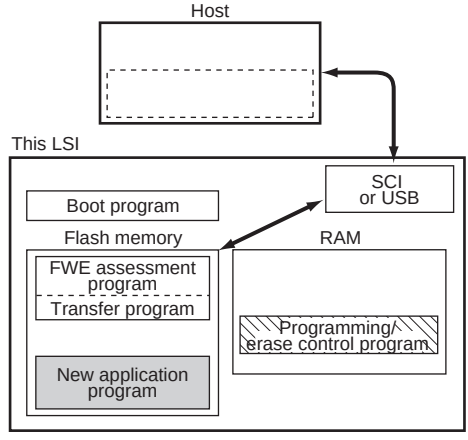



3. Flash memory initialization
   The programming/erase program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.

4. Writing new application program
   Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.




Program execution state

**Figure 17.4 User Program Mode (Sample)**

## 17.3 Block Configuration

Figure 17.5 shows the block configuration of 128-kbyte flash memory in the HD64F2218, HD64F2218U, HD64F2218CU, HD64F2212, HD64F2212U and HD64F2212CU. The thick lines indicate erasing units, the narrow lines indicate programming units, and the values are addresses. The flash memory is divided into one kbyte (four blocks), 28 kbytes (one block), 16 kbytes (one block), eight kbytes (two blocks), and 32 kbytes (two blocks). Erasing is performed in these divided units. Programming is performed in 128-byte units starting from an address whose lower eight bits are H'00 or H'80.

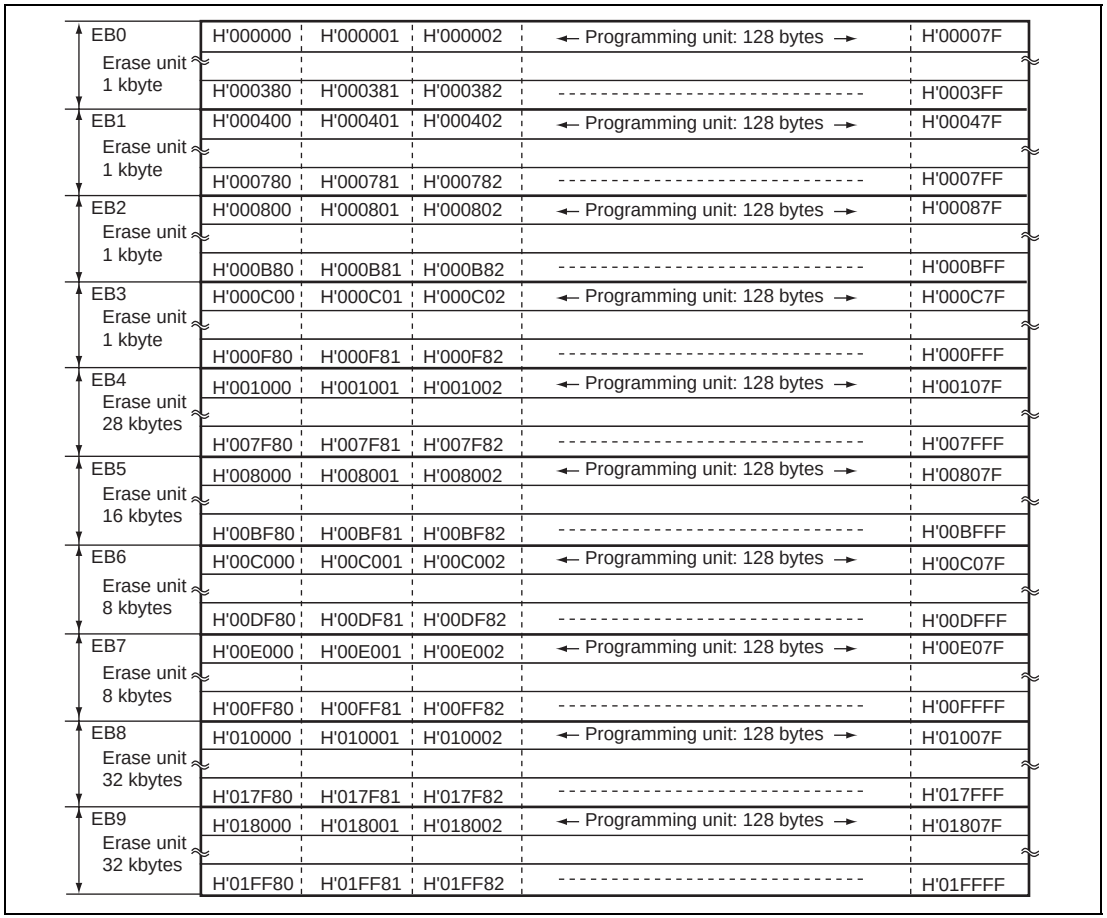



**Figure 17.5 Flash Memory Block Configuration**
**(HD64F2218, HD64F2218U, HD64F2218CU, HD64F2212, HD64F2212U, HD64F2212CU)**

Figure 17.6 shows the block configuration of 64-kbyte flash memory in the HD64F2217CU, HD64F2211, HD64F2211U and HD64F2211CU. The thick lines indicate erasing units, the narrow lines indicate programming units, and the values are addresses. The flash memory is divided into one kbyte (four blocks), 28 kbytes (one block), and 16 kbytes (one block), eight kbytes (two blocks). Erasing is performed in these divided units. Programming is performed in 128-byte units starting from an address whose lower eight bits are H'00 or H'80.

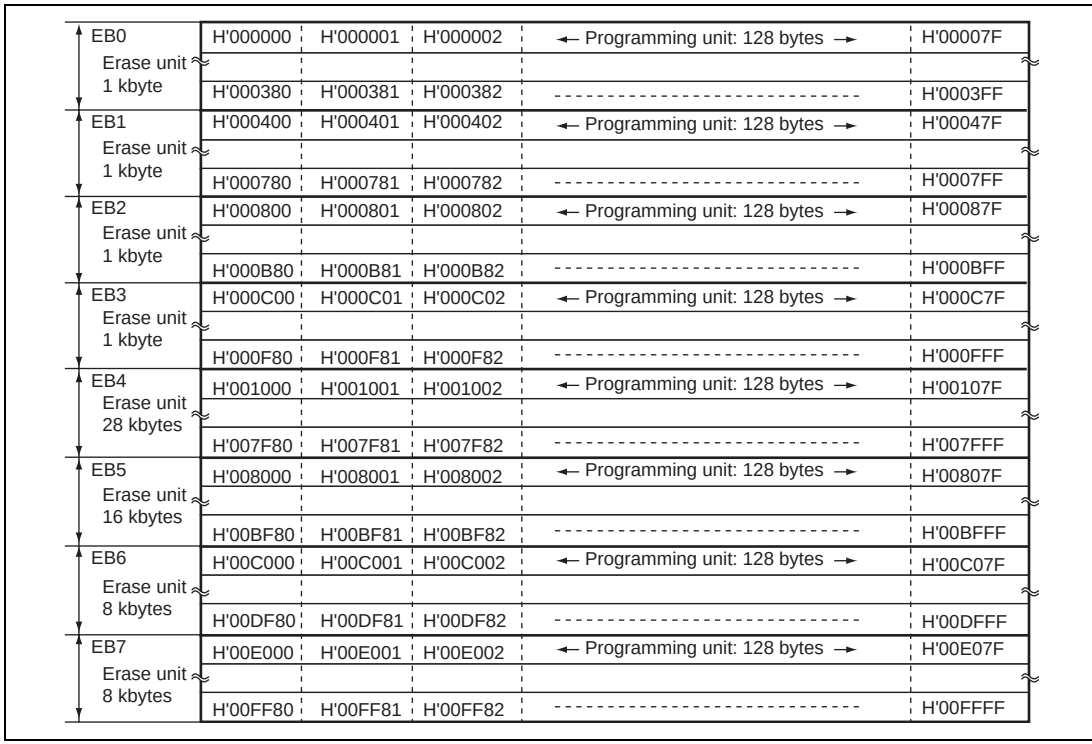



**Figure 17.6 Flash Memory Block Configuration (HD64F2217CU, HD64F2211, HD64F2211U, HD64F2211CU)**

Figure 17.7 shows the block configuration of the 32 Kbyte flash memory in the HD64F2210CU. The thick lines indicate erase blocks, the narrow lines indicate programming units, and the values in the boxes are addresses. The flash memory is divided into units one Kbyte (four blocks) or 28 Kbytes (one block) in size, and erasing is performed in these units. Programming is performed in 128-byte units starting from an address whose lower eight bits are H'00 or H'80.

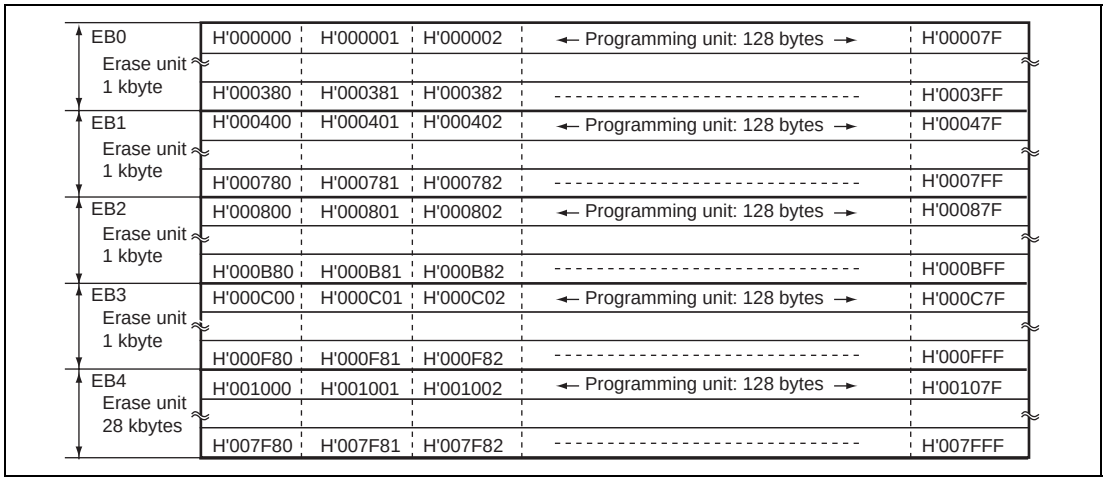



**Figure 17.7 Flash Memory Block Configuration (HD64F2210CU)**

## 17.4 Input/Output Pins

The flash memory is controlled by means of the pins shown in table 17.2.

**Table 17.2 Pin Configuration**

| Pin Name | I/O | Function | |
|---|---|---|---|
| $\overline{\text{RES}}$ | Input | Reset | All |
| FWE | Input | Flash program/erase protection by hardware | |
| MD2, MD1, MD0 | Input | Sets this LSI's operating mode | |
| PF3, PF0, P16, P14 | Input | Sets this LSI's operating mode in programmer mode | |
| EMLE | Input | Emulator enable | |
| TxD2 | Output | Serial transmit data output | HD64F2218, HD64F2212, HD64F2211 |
| RxD2 | Input | Serial receive data input | |
| USD+, USD− | Input/output | USB data input/output | HD64F2218U, HD64F2218CU, HD64F2217CU, HD64F2212U, HD64F2212CU, HD64F2211U |
| VBUS | Input | USB cable connect/cut detect | |
| $\overline{\text{UBPM}}$ | Input | USB bus power mode/self power mode select | |
| USPND | Output | USB suspend output | |
| P36 (PUPD+) | Output | D+ pull-up control | |

## 17.5 Register Descriptions

The flash memory has the following registers. For details on register addresses and register states during each processing, refer to section 21, List of Registers.

- Flash memory control register 1 (FLMCR1)
- Flash memory control register 2 (FLMCR2)
- Erase block register 1 (EBR1)
- Erase block register 2 (EBR2)
- RAM emulation register (RAMER)
- Serial control register X ( SCRX)

The masked ROM version is not equipped with the above registers. Attempting to read them with produce an undetermined value, and writing to them is invalid.

### 17.5.1 Flash Memory Control Register 1 (FLMCR1)

FLMCR1 is a register that makes the flash memory transit to program mode, program-verify mode, erase mode, or erase-verify mode. For details on register setting, refer to section 17.8, Flash Memory Programming/Erasing.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | FWE | —* | R | Flash Write Enable<br>Reflects the input level at the FWE pin. It is set to 1 when a low level is input to the FWE pin, and cleared to 0 when a high level is input. |
| 6 | SWE1 | 0 | R/W | Software Write Enable<br>When this bit is set to 1, flash memory programming/erasing is enabled. When this bit is cleared to 0, other FLMCR1 register bits and all EBR1, EBR2 bits cannot be set.<br>[Setting condition]<br>When FWE = 1 |
| 5 | ESU1 | 0 | R/W | Erase Setup<br>When this bit is set to 1, the flash memory transits to the erase setup state. When it is cleared to 0, the erase setup state is cancelled. Set this bit to 1 before setting the E1 bit in FLMCR1.<br>[Setting condition]<br>When FWE = 1 and SWE1 = 1 |
| 4 | PSU1 | 0 | R/W | Program Setup<br>When this bit is set to 1, the flash memory transits to the program setup state. When it is cleared to 0, the program setup state is cancelled. Set this bit to 1 before setting the P1 bit in FLMCR1.<br>[Setting condition]<br>When FWE = 1 and SWE1 = 1 |
| 3 | EV1 | 0 | R/W | Erase-Verify<br>When this bit is set to 1, the flash memory transits to erase-verify mode. When it is cleared to 0, erase-verify mode is cancelled.<br>[Setting condition]<br>When FWE = 1 and SWE1 = 1 |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 2 | PV1 | 0 | R/W | Program-Verify<br>When this bit is set to 1, the flash memory transits to program-verify mode. When it is cleared to 0, program-verify mode is cancelled.<br>[Setting condition]<br>When FWE = 1 and SWE1 = 1 |
| 1 | E1 | 0 | R/W | Erase<br>When this bit is set to 1 while the SWE1 and ESU1 bits are 1, the flash memory transits to erase mode. When it is cleared to 0, erase mode is cancelled.<br>[Setting condition]<br>When FWE = 1, SWE1 = 1, and ESU1 = 1 |
| 0 | P1 | 0 | R/W | Program<br>When this bit is set to 1 while the SWE1 and PSU1 bits are 1, the flash memory transits to program mode. When it is cleared to 0, program mode is cancelled.<br>[Setting condition]<br>When FWE = 1, SWE1 = 1, and PSU1 = 1 |

Note: * Set according to the FWE pin state.

### 17.5.2 Flash Memory Control Register 2 (FLMCR2)

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

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | FLER | 0 | R | Indicates that an error has occurred during an operation on flash memory (programming or erasing). When FLER is set to 1, flash memory goes to the error-protection state.<br>See section 17.9.3, Error Protection, for details. |
| 6 to 0 | — | All 0 | — | Reserved<br>These bits are always read as 0. |

### 17.5.3 Erase Block Register 1 (EBR1)

EBR1 specifies the flash memory erase area block. EBR1 is initialized to H'00 when the SWE1 bit in FLMCR is 0. Do not set more than one bit at a time, as this will cause all the bits in EBR1 and EBR2 to be automatically cleared to 0.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | EB7 | 0 | R/W | When this bit is set to 1, 8 kbytes of EB7 (H'00E000 to H'00FFFF) are to be erased. |
| 6 | EB6 | 0 | R/W | When this bit is set to 1, 8 kbytes of EB6 (H'00C000 to H'00DFFF) are to be erased. |
| 5 | EB5 | 0 | R/W | When this bit is set to 1, 16 kbytes of EB5 (H'008000 to H'00BFFF) are to be erased. |
| 4 | EB4 | 0 | R/W | When this bit is set to 1, 28 kbytes of EB4 (H'001000 to H'007FFF) are to be erased. |
| 3 | EB3 | 0 | R/W | When this bit is set to 1, 1 kbyte of EB3 (H'000C00 to H'000FFF) is to be erased. |
| 2 | EB2 | 0 | R/W | When this bit is set to 1, 1 kbyte of EB2 (H'000800 to H'000BFF) is to be erased. |
| 1 | EB1 | 0 | R/W | When this bit is set to 1, 1 kbyte of EB1 (H'000400 to H'0007FF) is to be erased. |
| 0 | EB0 | 0 | R/W | When this bit is set to 4, 1 kbyte of EB0 (H'000000 to H'0003FF) is to be erased. |

### 17.5.4 Erase Block Register 2 (EBR2)

EBR2 specifies the flash memory erase area block. EBR2 is initialized to H'00 when the SWE1 bit in FLMCR1 is 0. Do not set more than one bit at a time, as this will cause all the bits in EBR1 and EBR2 to be automatically cleared to 0.

Note: These registers are reserved on the HD64F2211 and HD64F2211U. Only H'00 should be written to them.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 2 | — | All 0 | R/W | Reserved<br>The write value should always be 0. |
| 1 | EB9 | 0 | R/W | When this bit is set to 1, 32 kbytes of EB9 (H'018000 to H'01FFFF) are to be erased. |
| 0 | EB8 | 0 | R/W | When this bit is set to 1, 32 kbytes of EB8 (H'010000 to H'017FFF) are to be erased. |

### 17.5.5 RAM Emulation Register (RAMER)

RAMER specifies the area of flash memory to be overlapped with part of RAM when emulating real-time flash memory programming. RAMER settings should be made in user mode or user program mode. To ensure correct operation of the emulation function, the ROM for which RAM emulation is performed should not be accessed immediately after this register has been modified. Normal execution of an access immediately after register modification is not guaranteed. For details, refer to section 17.7, Flash Memory Emulation in RAM.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R/W | Reserved<br>These bits always read as 0. The write value should always be 0. |
| 3 | RAMS | 0 | R/W | RAM Select<br>Specifies selection or non-selection of flash memory emulation in RAM. When RAMS = 1, the flash memory is overlapped with part of RAM, and all flash memory block are program/erase-protected. |
| 2 | RAM2 | 0 | R/W | Flash Memory Area Selection<br>When the RAMS bit is set to 1, selects one of the following flash memory areas to overlap the RAM area. The areas correspond with 1-kbyte erase blocks.<br>000: H'000000 to H'0003FF (EB0)<br>001: H'000400 to H'0007FF (EB1)<br>010: H'000800 to H'000BFF (EB2)<br>011: H'000C00 to H'000FFF (EB3)<br>100: Setting prohibited<br>101: Setting prohibited<br>110: Setting prohibited<br>111: Setting prohibited |
| 1 | RAM1 | 0 | R/W | |
| 0 | RAM0 | 0 | R/W | |

### 17.5.6 Serial Control Register X (SCRX)

SCRX performs register access control.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R/W | Reserved<br>The write value should always be 0. |
| 3 | FLSHE | 0 | R/W | Flash Memory Control Register Enable:<br>Controls CPU access to the flash memory control registers (FLMCR1, FLMCR2, EBR1, and EBR2). Setting the FLSHE bit to 1 enables read/write access to the flash memory control registers. If FLSHE is cleared to 0, the flash memory control registers are deselected. In this case, the flash memory control register contents are retained.<br>0: Flash control registers deselected in area H'FFFFA8 to H'FFFFAC<br>1: Flash control registers selected in area H'FFFFA8 to H'FFFFAC |
| 2 to 0 | — | All 0 | R/W | Reserved<br>The write value should always be 0. |

## 17.6 On-Board Programming Modes

When pins are set to on-board programming mode and a reset-start is executed, a transition is made to the on-board programming state in which program/erase/verify operations can be performed on the on-chip flash memory. There are two on-board programming modes: boot mode and user program mode. The pin settings for transition to each of these modes are shown in table 17.3. For a diagram of the transitions to the various flash memory modes, see figure 17.2.

**Table 17.3 Setting On-Board Programming Modes**

| Mode | | EMLE | FWE | MD2 | MD1 | MD0 |
|---|---|---|---|---|---|---|
| SCI boot mode (HD64F2218, HD64F2212, HD64F2211) | Advanced: single-chip mode | 0 | 1 | 0 | 1 | × |
| USB boot mode (HD64F2218U, HD64F2218CU, HD64F2217CU, HD64F2212U, HD64F2212CU, HD64F2211U, HD64F2211CU, HD64F2210CU) | Advanced: single-chip mode 24 MHz system clock | 0 | 1 | 0 | 1 | 0 |
| | Advanced: single-chip mode 16 MHz system clock | 0 | 1 | 0 | 1 | 1 |
| User program mode | Advanced: on-chip ROM extended mode (MCU operating mode 6) | 0 | 1 | 1 | 1 | 0 |
| | Advanced: Single-chip mode (MCU operating mode 7) | 0 | 1 | 1 | 1 | 1 |

### 17.6.1 SCI Boot Mode (HD64F2218, HD64F2212, and HD64F2211)

When a reset-start is executed after the LSI's pins have been set to boot mode, the boot program built into the LSI is started and the programming control program prepared in the host is serially transmitted to the LSI via the SCI. In the LSI, the programming control program received via the SCI is written into the programming control program area in on-chip RAM. After the transfer is completed, control branches to the start address of the programming control program area and the programming control program execution state is entered (flash memory programming is performed). The system configuration in boot mode is shown in Figure 17.8.

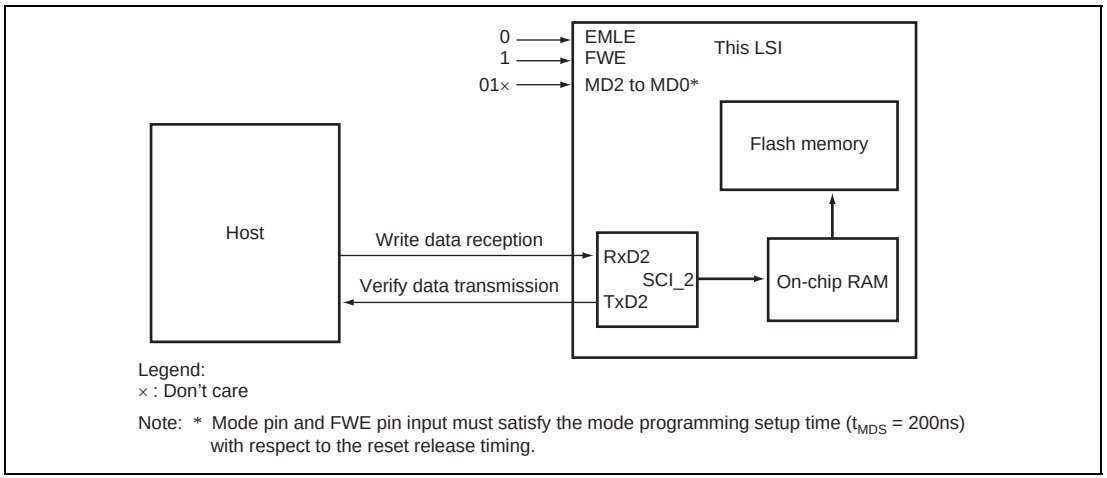



**Figure 17.8 System Configuration in SCI Boot Mode**

Table 17.4 shows the boot mode operations between reset end and branching to the programming control program.

1. When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. Prepare a programming control program in accordance with the description in section 17.8, Flash Memory Programming/Erasing. In boot mode, if any data has been programmed into the flash memory (if all data is not 1), all flash memory blocks are erased. Boot mode is for use in enforced exit when user program mode is unavailable, such as the first time on-board programming is performed, or if the program activated in user program mode is accidentally erased.
2. The SCI_2 should be set to asynchronous mode, and the transfer format as follows: 8-bit data, 1 stop bit, and no parity.
3. When the boot program is initiated, the chip measures the low-level period of asynchronous SCI communication data (H'00) transmitted continuously from the host. The chip then calculates the bit rate of transmission from the host, and adjusts the SCI_2 bit rate to match that of the host. The reset should end with the RxD pin high. The RxD and TxD pins should be pulled up on the board if necessary. After the reset ends, it takes approximately 100 states before the chip is ready to measure the low-level period.
4. After matching the bit rates, the chip transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the chip. If reception could not be performed normally, initiate boot mode again by a reset. Depending on the host's transfer bit rate and system clock frequency of this LSI, there will be a discrepancy between the bit rates of the host and the chip. To operate the SCI properly, set the host's transfer bit rate and system clock frequency of this LSI within the ranges listed in table 17.5.

5. In boot mode, a part of the on-chip RAM area (four kbytes) is used by the boot program. The area to which the programming control program is transferred from the host is 8 kbytes (H'FFC000 to H'FFDFFF) in the HD64F2218 and HD64F2212 and 4 kbytes (H'FFD000 to H'FFDFFF) in the HD64F2211. The boot program area cannot be used until the execution state in boot mode switches to the programming control program.
6. Before branching to the programming control program, the chip terminates transfer operations by the SCI_2 (by clearing the RE and TE bits in SCR to 0), but the adjusted bit rate value remains set in BRR. Therefore, the programming control program can still use it for transfer of write data or verify data with the host. The TxD pin is high. The contents of the CPU general registers are undefined immediately after branching to the programming control program. These registers must be initialized at the beginning of the programming control program, since the stack pointer (SP), in particular, is used implicitly in subroutine calls, etc.
7. Boot mode can be cleared by a reset. End the reset* after driving the reset pin low, waiting at least 20 states, and then setting the FWE pin and the mode (MD) pins. Boot mode is also cleared when a WDT overflow occurs.
8. Do not change the MD pin input levels in boot mode. If the mode pin input levels are changed (for example, from low to high) during a reset, the state of ports with multiplexed address functions and bus control output pins ($\overline{AS}$, $\overline{RD}$, $\overline{WR}$) will change according to the change in the microcomputer's operating mode . Therefore, care must be taken to make pin settings to prevent these pins from becoming output signal pins during a reset, or to prevent collision with signals outside the microcomputer.

9. All interrupts are disabled during programming or erasing of the flash memory.

Note:* Mode pin and FWE pin input must satisfy the mode programming setup time ($t_{MDS}$ = 200 ns) with respect to the reset release timing.

**Table 17.4 Boot Mode Operation**

| Item | Host Operation | LSI Operation |
|---|---|---|
| | | Branches to boot program at reset-start |
| Bit rate adjustment | Continuously transmits data H'00 at specified bit rate | Measures low-level period of receive data H'00 |
| | | Calculates bit rate and sets it in BRR of SCI_2 |
| | Transmits data H'55 when data H'00 is received error-free | Transmits data H'00 to host as adjustment end indication |
| | | Transmits data H'AA to host when data H'55 is received |
| Transmits number of bytes (N) of programming control program | Transmits number of bytes (N) of programming control program to be transferred as 2-byte data (low-order byte following high-order byte) | Echobacks the 2-byte data received as verification data |
| Transmits 1-byte of programming control program (repeated for N times) | Transmits 1-byte of programming control program | Echobacks received data to host and also transfers it to RAM |
| Flash memory erase | | Checks flash memory data, erases all flash memory blocks in case of written data existing, and transmits data H'AA to host. (If erase could not be done, transmits data H'FF to host and aborts operation) |
| Programming control program execution | | Branches to programming control program transferred to on-chip RAM and starts execution |

**Table 17.5 System Clock Frequencies for which Automatic Adjustment of LSI Bit Rate Is Possible**

| Host Bit Rate | System Clock Frequency Range of LSI |
|---|---|
| 19,200 bps | 8 to 24 MHz |
| 9,600 bps | 6 to 24 MHz |
| 4,800 bps | 6 to 24 MHz |

### 17.6.2 USB Boot Mode (HD64F2218U, HD64F2212U, and HD64F2211U)

- Features
  - Selection of bus-powered mode or self-powered mode
  - Supports the USB operating clock generation by 16 MHz system clock with PLL3 multiplication (FWE = 1, MD2 to MD0 = 011) or 24 MHz system clock with PLL2 multiplication (FWE = 1, MD2 to MD0 = 010)
  - D+ pull up control connection supported for P36 pin only
  - See table 17.6 for enumeration information

**Table 17.6 Enumeration Information**

| | | |
|---|---|---|
| USB Standard | Ver.1.1 | |
| Transfer modes | Control (in, out), Bulk (in, out) | |
| Maximum power | Self power mode ($\overline{\text{UBPM}}$ pin = 1) | 100 mA |
| | Bus power mode ($\overline{\text{UBPM}}$ pin = 0) | 500 mA |
| Endpoint configuration | EP0 Control (in, out) 64 bytes<br>Configuration 1<br>└ Interface Number 0<br>└ Alternate Setting 0<br>├ EP1 Bulk (in) 64 bytes<br>└ EP2 Bulk (out) 64 bytes | |

- Notes on USB Boot Mode Execution
  - Specify 16 MHz or 24 MHz system clock and the FWE and MD2 to MD0 pins correctly.
  - Use the P36 pin for D+ pull-up control connection.
  - To ensure stable power supply during flash memory programming/erasing, do not use cable connection via a bus powered HUB.
  - Note in particular that, in the worst case, the LSI may be permanently damaged if the USB cable is detached during flash memory programming/erasing.
  - A transition is not made to power-down modes even if the USB bus enters suspend mode when in bus power mode.

- Overview

  When a reset start preformed after the pins of this LSI have been set to boot mode, a boot program incorporated in the microcomputer beforehand is activated, and the prepared programming control program is transmitted sequentially to the host using the USB. With this LSI, the programming control program received by the USB is written to a programming control program area in on-chip RAM. After transfer is completed, control branches to the start address of the programming control program area, and the programming control program execution state is established (flash memory programming is performed). Figure 17.9 shows a system configuration diagram when using USB boot mode.

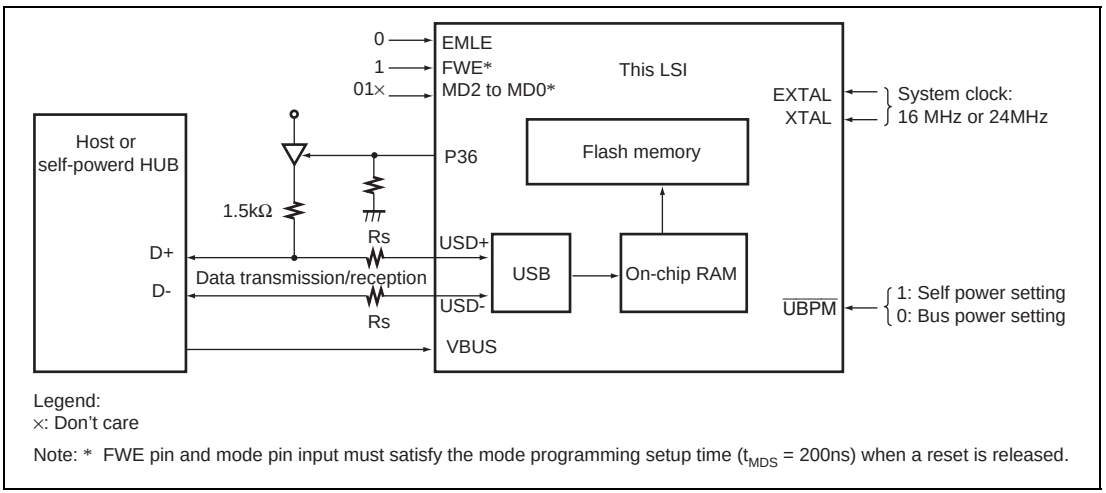



**Figure 17.9 System Configuration Diagram when Using USB Boot Mode**

Table 17.7 shows operations from reset release in USB boot mode until processing branches to the programming control program.

1. When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. Prepare a programming control program in accordance with the description in section 17.8, Flash Memory Programming/Erasing. In boot mode, if any data has been programmed into the flash memory (if all data is not 1), all flash memory blocks are erased. Boot mode is for use in enforced exit when user program mode is unavailable, such a the first time on-board programming control program, or performed, or if the program activated in user program mode is accidentally erased.
2. When the boot program is activated, enumeration with respect to the host is carried out. Enumeration information is shown in table 17.6. When enumeration is completed, transmit a single H'55 byte from the host. If reception has not been preformed normally, restart boot mode by means of a reset.

3. Set the frequency for transmission from the host as a numeric value in units of MHz × 100 (ex: 16.00 MHz → H'0640, 24.00 MHz → H'0960).
4. In boot mode, the 4-kbyte on-chip RAM area H'FFE000 to H'FFEFBF is used by the boot program. The programming control program is transferred from the host stored in the 8-kbyte area H'FFC000 to H'FFDFFF in the HD64F2218U, HD64F2218CU, HD64F2212U, and HD64F2212CU and the 4-kbyte area H'FFD000 to H'FFDFFF in the HD64F2211U, HD64F2211CU and HD64F2210CU. The boot program area cannot be used until program execution switches to the programming control program. Also note that the boot program remains in RAM even after control passes to the programming control program.
5. When a branch is made to the programming control program, the USB remains connected and can be used immediately for transmission/reception of write data or verify data between the programming control program and the host. The contents of CPU general registers are undefined after a branch to the programming control program. Note, in particular, that since the stack pointer is used implicitly in subroutine calls ad the like, it should be initialized at the start of the programming control program.
6. Boot mode is by means of a reset. Drive the reset pin low, wait for the elapse of at least 20 states, then set the FWE pin and mode pins to release the reset.* Boot mode is also exited in the event of a WDT overflow reset.
7. Do not change the input level of the mode pins while in boot mode. In the input level of a mode pin is changed (from low to high) during a reset, the states of ports with a dual function as address output s, and bus control output signals ($\overline{AS}$, $\overline{RD}$, $\overline{WR}$), will change due to switching of the operating mode. Either make pin settings so that these pins do not become output signal pins during a reset, or take precautions to prevent collisions with external signals.
8. Interrupt cannot be used during flash memory programming or erasing.

Note: * FWE pin and mode pin input must satisfy the mode programming set up time ($t_{MDS}$ = 200 ns) when a reset is released.

**Table 17.7 USB Boot Mode Operation**

| Item | Host Operation | Operation of this LSI |
|---|---|---|
| | | Branches to boot program after reset start |
| Start of USB boot mode | Transmits one H'55 byte on completion of USB enumeration | |
| | | Transmits one H'AA byte to host on reception of H'55 |
| Transfer clock information | Transmits frequency (2 bytes), number of multiplication | |
| | Classification (1 byte), multiplication ratio (1 byte) | |
| | With 16 MHz system clock, H'0640, H'01, H'01 are transmitted | |
| | With 24 MHz system clock, H'0960, H'01, H'01 are transmitted | If received data are within respective ranges, transmits H'AA to host |
| | | If any received data is out-of-ranges, transmits H'FF to host and halts operation |
| Transfer number of bytes (N) of programming control program | Performs 2-byte transfer number of bytes (N) of programming control program | |
| | | If received number of bytes is within ranges, transmits H'AA to host |
| | | If received number of bytes is out-of- ranges, transmits H'FF to host and halts operation |

| Item | Host Operation | Operation of this LSI |
|---|---|---|
| Transfer of programming control program and sum value | Transmits programming control program in N-byte divisions. | |
| | | Transfers received data to on-chip RAM |
| | Transmits sum value (two's complement of sum total of programming control program (1 byte)) | |
| | | Calculates sum total of received sum value and 1 byte units of programming control program transferred to on-chip RAM |
| | | If sum is 0, transmits H'AA to host |
| | | If sum is not 0, transmits H'FF to host halts operation |
| Memory erase | | Starts total erase of flash memory |
| | Transmits total erase status command (H'3A) | |
| | | Transmits H'11 to host if total erase processing is being executed when total erase status command is received |
| | | Transmits H'06 to host if total erase of all blocks has been completed when total erase status command is received |
| | Retransmits total erase status command (H'3A) when H'11 is received | If erase cannot be performed when total erase status command is received, transmits H'EE to host and halts operation |
| Execution of programming control program | | Branches to programming control program transferred to on-chip RAM and starts execution. |

### 17.6.3 Programming/Erasing in User Program Mode

On-board programming/erasing of an individual flash memory block can also be performed in user program mode by branching to a user program/erase control program. The user must set branching conditions and provide on-board FWE control and supply of programming data, and storing a program/erase control program in part of the program area as necessary. The flash memory must contain the user program/erase control program or a program which provides the user program/erase control program from external memory. Because the flash memory itself cannot be read during programming/erasing, transfer the user program/erase control program to on-chip RAM, as like in boot mode. Figure 17.10 shows a sample procedure for programming/erasing in user program mode. Prepare a user program/erase control program in accordance with the description in section 17.8, Flash Memory Programming/Erasing.

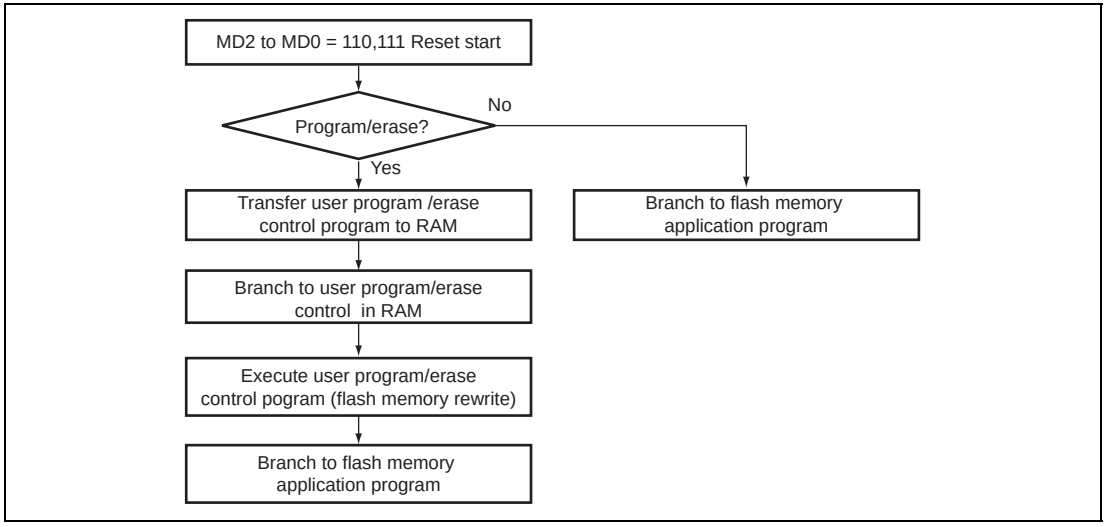



**Figure 17.10 Programming/Erasing Flowchart Example in User Program Mode**

## 17.7 Flash Memory Emulation in RAM

Making a setting in the RAM emulation register (RAMER) enables part of RAM to be overlapped onto the flash memory area so that data to be written to flash memory can be emulated in RAM in real time. Emulation can be performed in user mode or user program mode. Figure 17.11 shows an example of emulation of real-time flash memory programming.

1. Set RAMER to overlap part of RAM onto the area for which real-time programming is required.
2. Emulation is performed using the overlapping RAM.
3. After the program data has been confirmed, the RAMS bit is cleared, thus releasing RAM overlap.
4. The data written in the overlapping RAM is written into the flash memory space (EB0).

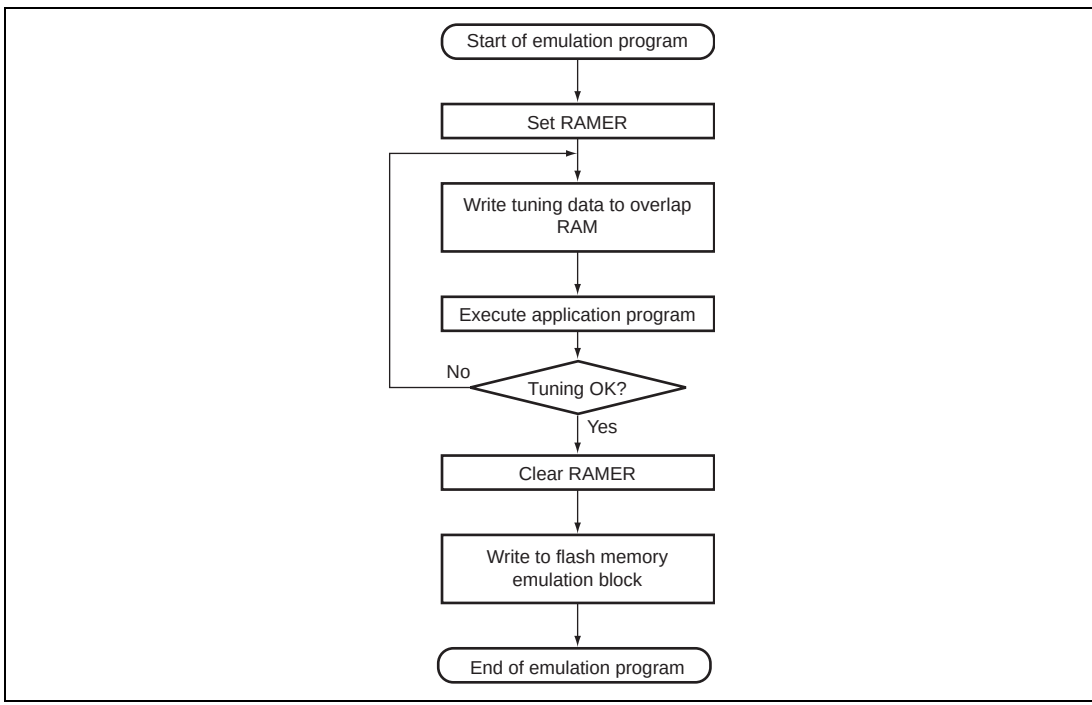



**Figure 17.11 Flowchart for Flash Memory Emulation in RAM**

An example in which flash memory block area EB0 is overlapped is shown in Figure 17.12.

1. The RAM area to be overlapped is fixed at a 1-kbyte area in the range of H'FFD000 to H'FFD3FF.
2. The flash memory area to overlap is selected by RAMER from a 1-kbyte area among one of the EB0 to EB3 blocks.
3. The overlapped RAM area can be accessed from both the flash memory addresses and RAM addresses.
4. When the RAMS bit in RAMER is set to 1, program/erase protection is enabled for all flash memory blocks (emulation protection). In this state, setting the P1 or E1 bit in FLMCR1 to 1 does not cause a transition to program mode or erase mode.
5. A RAM area cannot be erased by execution of software in accordance with the erase algorithm.
6. Block area EB0 contains the vector table. When performing RAM emulation, the vector table is needed in the overlap RAM.

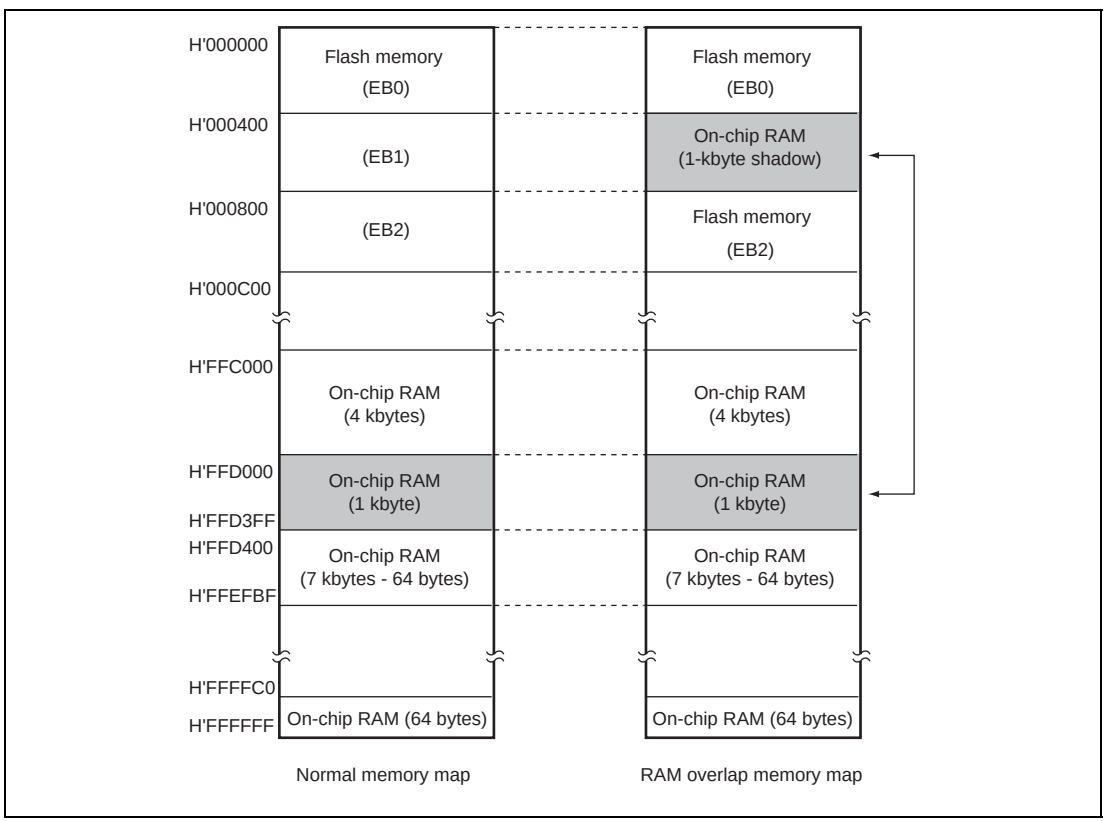



**Figure 17.12 Example of RAM Overlap Operation**

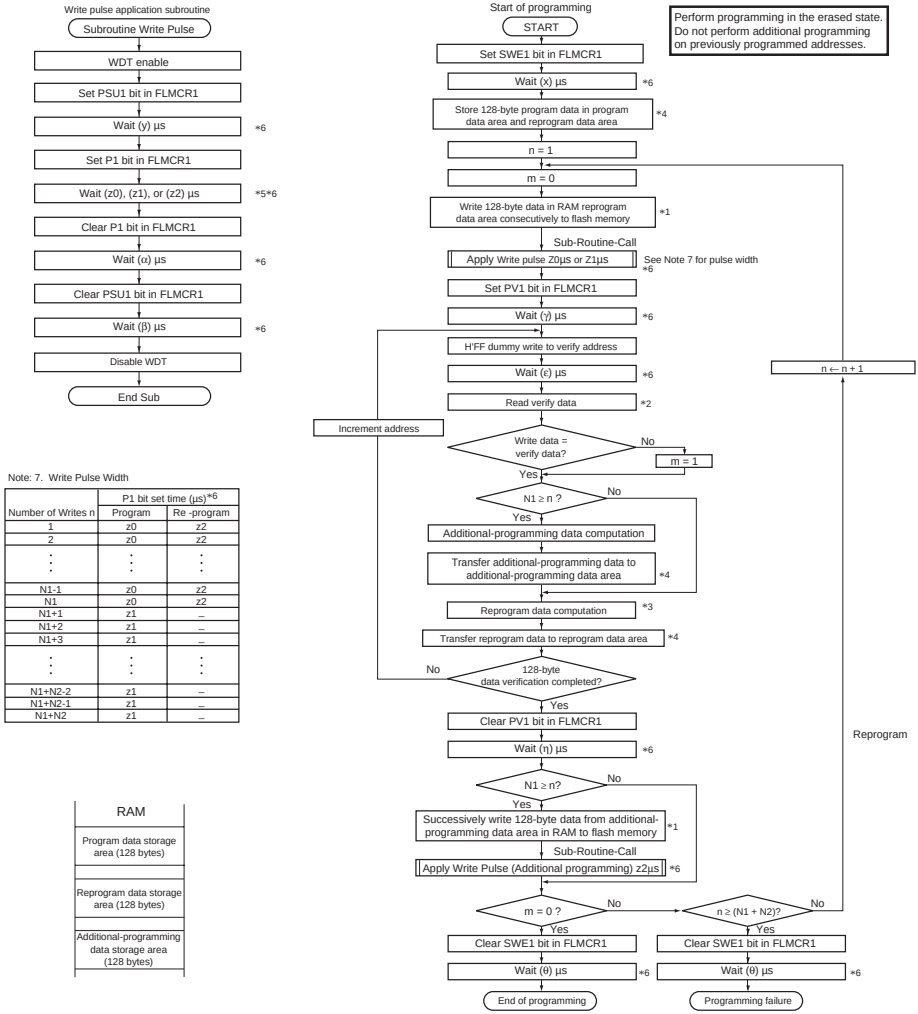
## 17.8 Flash Memory Programming/Erasing

A software method, using the CPU, is employed to program and erase flash memory in the on-board programming modes. Depending on the FLMCR1 setting, the flash memory operates in one of the following four modes: program mode, erase mode, program-verify mode, and erase-verify mode. The programming control program in boot mode and the user program/erase control program in user program mode use these operating modes in combination to perform programming/erasing. Flash memory programming and erasing should be performed in accordance with the descriptions in section 17.8.1, Program/Program-Verify and section 17.8.2, Erase/Erase-Verify, respectively.

### 17.8.1 Program/Program-Verify

When writing data or programs to the flash memory, the program/program-verify flowchart shown in Figure 17.13 should be followed. Performing programming operations according to this flowchart will enable data or programs to be written to the flash memory without subjecting the chip to voltage stress or sacrificing program data reliability.

1. Programming must be done to an empty address. Do not reprogram an address to which programming has already been performed.
2. Programming should be carried out 128 bytes at a time. A 128-byte data transfer must be performed even if writing fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
3. Prepare the following data storage areas in RAM: a 128-byte programming data area, a 128-byte reprogramming data area, and a 128-byte additional-programming data area. Perform reprogramming data computation and additional programming data computation according to Figure 17.13.
4. Consecutively transfer 128 bytes of data in byte units from the reprogramming data area or additional-programming data area to the flash memory. The program address and 128-byte data are latched in the flash memory. The lower 8 bits of the start address in the flash memory destination area must be H'00 or H'80.
5. The time during which the P1 bit is set to 1 is the programming time. Figure 17.13 shows the allowable programming times.
6. The watchdog timer (WDT) is set to prevent overprogramming due to program runaway, etc. An overflow cycle of approximately $(y + z1 + \alpha + \beta)$ μs is allowed.
7. For a dummy write to a verify address, write 1-byte data H'FF to an address whose lower 1 bit is B'0. Verify data can be read in words from the address to which a dummy write was performed.
8. The maximum number of repetitions of the program/program-verify sequence to the same bit is (N1 + N2).




Note: 7. Write Pulse Width

| Number of Writes n | P1 bit set time (µs)*6 | |
|---|---|---|
| | Program | Re -program |
| 1 | z0 | z2 |
| 2 | z0 | z2 |
| ⋮ | ⋮ | ⋮ |
| N1-1 | z0 | z2 |
| N1 | z0 | z2 |
| N1+1 | z1 | – |
| N1+2 | z1 | – |
| N1+3 | z1 | – |
| ⋮ | ⋮ | ⋮ |
| N1+N2-2 | z1 | – |
| N1+N2-1 | z1 | – |
| N1+N2 | z1 | – |

Notes: 1. Data transfer is performed by byte transfer. The lower 8 bits of the first address written to must be H'00 or H'80.
A 128-byte data transfer must be performed even if writing fewer than 128 bytes; in this case, H'FF data must be written to the extra addresses.
2. Verify data is read in 16-bit (word) units.
3. Reprogram data is determined by the operation shown in the table below (comparison between the data stored in the program data area and the verify data). Bits for which the reprogram data is 0 are programmed in the next reprogramming loop. Therefore, even bits for which programming has been completed will be subjected to programming once again if the result of the subsequent verify operation is NG.
4. A 128-byte area for storing program data, a 128-byte area for storing reprogram data, and a 128-byte area for storing additional data must be provided in RAM. The contents of the reprogram data area and additional data area are modified as programming proceeds.
5. A write pulse of z0 or z1 is applied according to the progress of the programming operation. See note 7 for details of the pulse widths. When writing of additional-programming data is executed, a z2 write pulse should be applied. Reprogram data X' means reprogram data when the write pulse is applied.
6. x, y, z0, z1, z2, α, β, γ, ε, η, θ, N1, and N2 are shown in section 22.7, Flash Memory Characteristics.

Reprogram Data Computation Table

| Original Data (D) | Verify Data (V) | Reprogram Data (X) | Comments |
|---|---|---|---|
| 0 | 0 | 1 | Programming completed |
| 0 | 1 | 0 | Programming incomplete; reprogram |
| 1 | 0 | 1 | |
| 1 | 1 | 1 | Still in erased state; no action |

Additional-Programming Data Computation Table

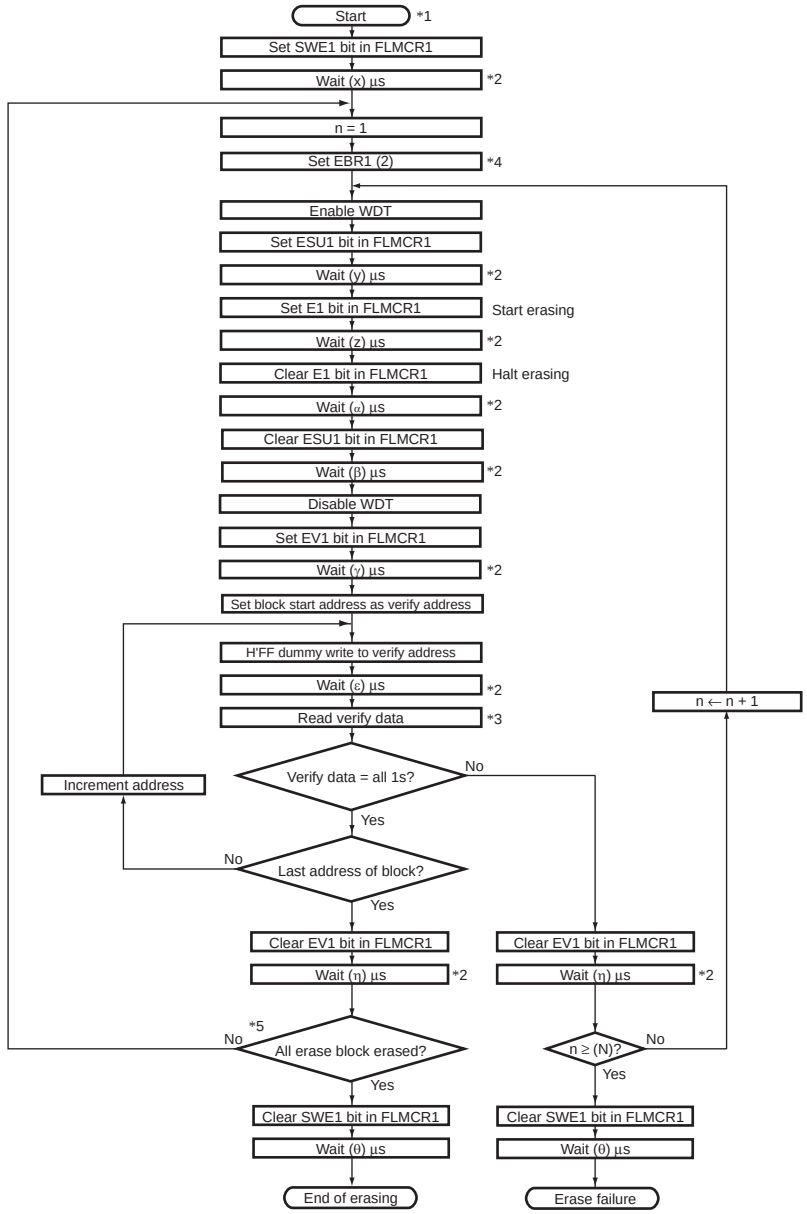
| Reprogram Data (X') | Verify Data (V) | Additional-Programming Data (Y) | Comments |
|---|---|---|---|
| 0 | 0 | 0 | Additional programming to be executed |
| 0 | 1 | 1 | Additional programming not to be executed |
| 1 | 0 | 1 | Additional programming not to be executed |
| 1 | 1 | 1 | Additional programming not to be executed |

**Figure 17.13 Program/Program-Verify Flowchart**

### 17.8.2 Erase/Erase-Verify

When erasing flash memory, the erase/erase-verify flowchart shown in Figure 17.14 should be followed.

1. Prewriting (setting erase block data to all 0s) is not necessary.
2. Erasing is performed in block units. Make only a single-bit specification in the erase block register 1, 2 (EBR1, EBR2). To erase multiple blocks, each block must be erased in turn.
3. The time during which the E1 bit is set to 1 is the flash memory erase time.
4. The watchdog timer (WDT) is set to prevent overprogramming due to program runaway, etc. An overflow cycle of approximately (y+z+α+β) ms is allowed.
5. For a dummy write to a verify address, write 1-byte data H'FF to an address whose lower 1 bit is B'0. Verify data can be read in longwords from the address to which a dummy write was performed.
6. If the read data is unerased, set erase mode again, and repeat the erase/erase-verify sequence as before. The maximum number of repetitions of the erase/erase-verify sequence is N.




Notes: 1. Pre-write (clearing data in the block to be erased to 0) isn not required.
2. x, y, z, α, β, γ, ε, η, θ, and N are shown in section 22.7, Flash Memory Characteristics.
3. Veryfy data is read in 16 bits.
4. Only 1 bit in the EBR register must be set. Two or more bits in EBR cannot be set.
5. Erasure is performed in block units. To erase multiple blocks, each block must be erased sequentially.

**Figure 17.14 Erase/Erase-Verify Flowchart**

## 17.9 Program/Erase Protection

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

### 17.9.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted because of a transition to reset or standby mode. Flash memory control register 1 (FLMCR1), flash memory control register 2 (FLMCR2), erase block register 1 (EBR1), and erase block register 2 (EBR2) are initialized. In a reset via the $\overline{\text{RES}}$ pin, the reset state is not entered unless the $\overline{\text{RES}}$ pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width specified in the AC Characteristics section.

### 17.9.2 Software Protection

Software protection can be implemented against programming/erasing of all flash memory blocks by clearing the SWE1 bit in FLMCR1. When software protection is in effect, setting the P1 or E1 bit in FLMCR1 does not cause a transition to program mode or erase mode. By setting the erase block register 1 (EBR1), and erase block register 2 (EBR2), erase protection can be set for individual blocks. When EBR1 and EBR2 are set to H'00, erase protection is set for all blocks.

### 17.9.3 Error Protection

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

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

Setting Conditions of FLER Bit (Erase Protection)

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

The FLMCR1, FLMCR2, EBR1 and EBR2 settings are retained, but program mode or erase mode is aborted at the point at which the error occurred. Program mode or erase mode cannot be re-entered by re-setting the P1 or E1 bit. However, PV1 and EV1 bit setting is enabled, and a transition can be made to verify mode.

## 17.10 Interrupt Handling when Programming/Erasing Flash Memory

All interrupts, including NMI interrupt is disabled when flash memory is being programmed or erased (when the P1 or E1 bit is set in FLMCR1), and while the boot program is executing in boot mode*[1], to give priority to the program or erase operation. There are three reasons for this:

1. Interrupt during programming or erasing might cause a violation of the programming or erasing algorithm, with the result that normal operation could not be assured.
2. In the interrupt exception handling sequence during programming or erasing, the vector would not be read correctly*[2] , possibly resulting in CPU runaway.
3. If interrupt occurred during boot program execution, it would not be possible to execute the normal boot mode sequence.

Notes: 1. Interrupt requests must be disabled inside and outside the CPU until the programming control program has completed programming.

2. The vector may not be read correctly in this case for the following two reasons:
   - If flash memory is read while being programmed or erased (while the P1 or E1 bit is set in FLMCR1), correct read data will not be obtained (undetermined values will be returned).
   - If the interrupt entry in the vector table has not been programmed yet, interrupt exception handling will not be executed correctly.

## 17.11 Programmer Mode

In programmer mode, a PROM programmer can perform programming/erasing via a socket adapter, just like for a discrete flash memory. Use a PROM programmer that supports the Renesas Technology 128-kbyte or 64-kbyte flash memory on-chip MCU device type. Memory map in programmer mode is shown in Figure 17.15.

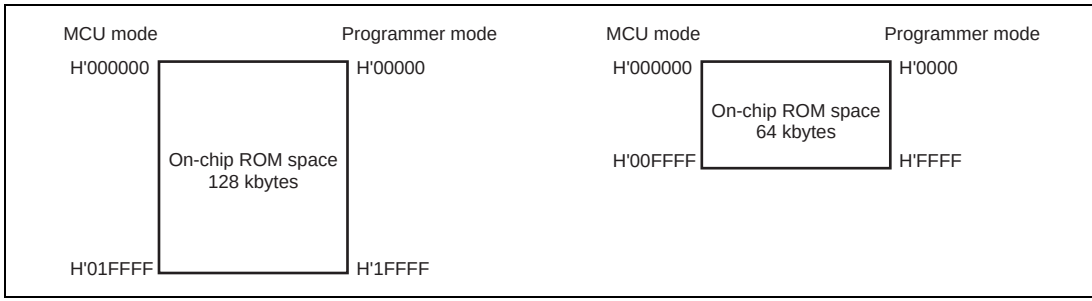



**Figure 17.15 Memory Map in Programmer Mode**

## 17.12 Power-Down States for Flash Memory

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

- Normal operating mode
  The flash memory can be read and written to.
- Standby mode
  All flash memory circuits are halted.
- Power-down state
  The flash memory can be read when part of the power supply circuit is halted and the LSI operates by subclocks.

Table 17.8 shows the correspondence between the operating modes of this LSI and the flash memory. When the flash memory returns to normal operation from a power-down state, a power supply circuit stabilization period is needed. When the flash memory returns to its normal operating state from watch mode or standby mode, bits STS2 to STS0 in SBYCR must be set to provide a wait time of at least 100 μs; when returns from flash memory module stop mode, the software wait state should be set.

**Table 17.8 Flash Memory Operating States**

| LSI Operating State | Flash Memory Operating State |
|---|---|
| Active mode<br>Sleep mode | Normal operating mode |
| Watch mode<br>Standby mode<br>Flash memory module stop mode | Standby mode<br>(Before entering to the normal operation mode, wait time of at least 100 μs is required.) |
| Subactive mode<br>Subsleep mode | Power-down mode (read only) |

## 17.13 Flash Memory Programming and Erasing Precautions

Precautions concerning the use of on-board programming mode, the RAM emulation function, and PROM mode are summarized below.

- Use the specified voltages and timing for programming and erasing.

  Applied voltages in excess of the rating can permanently damage the device. Use a PROM programmer that supports the Renesas Technology microcomputer device type with on-chip flash memory (FZTAT128V3A, FZTAT64V3A).

  Do not select the HN27C4096 setting for the PROM programmer, and only use the specified socket adapter. Failure to observe these points may result in damage to the device.
- Powering on and off

  Do not apply a high level to the FWE pin until VCC has stabilized. Also, drive the FWE pin low before turning off VCC. When applying or disconnecting VCC power, fix the FWE pin low and place the flash memory in the hardware protection state. The power-on and power-off timing requirements should also be satisfied in the event of a power failure and subsequent recovery.
- FWE application/disconnection

  FWE application should be carried out when MCU operation is in a stable condition. If MCU operation is not stable, fix the FWE pin low and set the protection state. The following points must be observed concerning FWE application and disconnection to prevent unintentional programming or erasing of flash memory:
  - Apply FWE when the VCC voltage has stabilized within its rated voltage range.
  - In boot mode, apply and disconnect FWE during a reset.
  - In user program mode, FWE can be switched between high and low level regardless of the reset state. FWE input can also be switched during execution of a program in flash memory.
  - Do not apply FWE if program runaway has occurred.
  - Disconnect FWE only when the SWE1, ESU1, PSU1, EV1, PV1, P1, and E1 bits in FLMCR1 are cleared. Make sure that the SWE1, ESU1, PSU1, EV1, PV1, P1, and E1 bits are not set by mistake when applying or disconnecting FWE.
- Do not apply a constant high level to the FWE pin.

  Apply a high level to the FWE pin only when programming or erasing flash memory. A system configuration in which a high level is constantly applied to the FWE pin should be avoided. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

- Use the recommended algorithm when programming and erasing flash memory.

  The recommended algorithm enables programming and erasing to be carried out without subjecting the device to voltage stress or sacrificing program data reliability. When setting the P1 or E1 bit in FLMCR1, the watchdog timer should be set beforehand as a precaution against program runaway, etc.
- Do not set or clear the SWE1 bit during execution of a program in flash memory.

  Wait at least θ µs* after clearing the SWE1 bit before executing a program or reading data in flash memory. When the SWE1 bit is set, data in flash memory can be rewritten, but access flash memory only for verify operations (verification during programming/erasing). Also, do not clear the SWE1 bit during programming, erasing, or verifying. Similarly, when using emulation by RAM with a high level applied to the FWE pin, the SWE1 bit should be cleared before executing a program or reading data in flash memory. However, read/write accesses can be performed in the RAM area overlapping the flash memory space regardless of whether the SWE1 bit is set or cleared.

Note: * Refer to section 22.7, Flash Memory Characteristics.

- Do not use interrupts while flash memory is being programmed or erased.

  All interrupt requests, including NMI, should be disabled during FWE application to give priority to program/erase operations.
- Do not perform additional programming. Erase the memory before reprogramming.

  In on-board programming, perform only one programming operation on a 128-byte programming unit block. In programmer mode, too, perform only one programming operation on a 128-byte programming unit block. Programming should be carried out with the entire programming unit block erased.
- Before programming, check that the chip is correctly mounted in the PROM programmer.

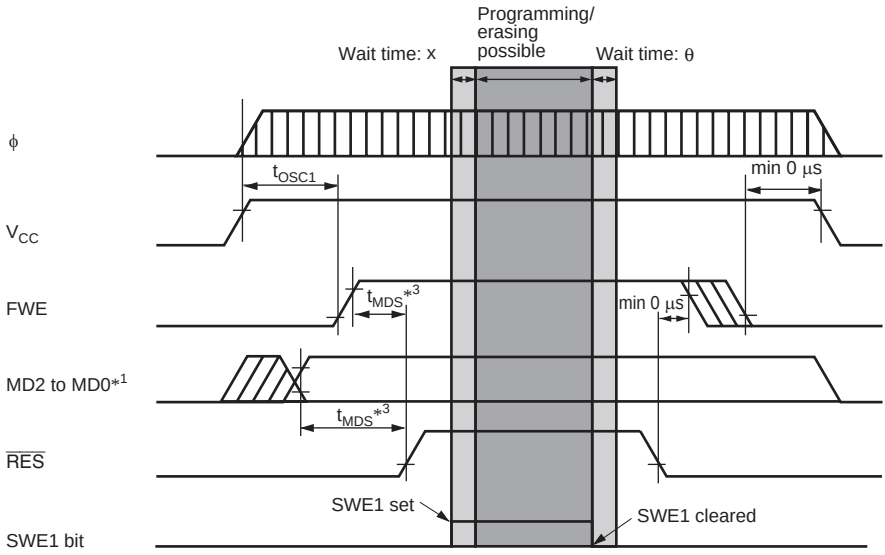
  Overcurrent damage to the device can result if the index marks on the PROM programmer socket, socket adapter, and chip are not correctly aligned.
- Do not touch the socket adapter or chip during programming.

  Touching either of these can cause contact faults and write errors.
- The reset state must be entered after powering on

  Apply the reset signal for at least 100 µs during the oscillation setting period.
- When a reset is applied during operation, this should be done while the SWE1 pin is low.

  Wait at least θ µs* after clearing the SWE1 bit before applying the reset.

Note: * Refer to section 22.7, Flash Memory Characteristics.

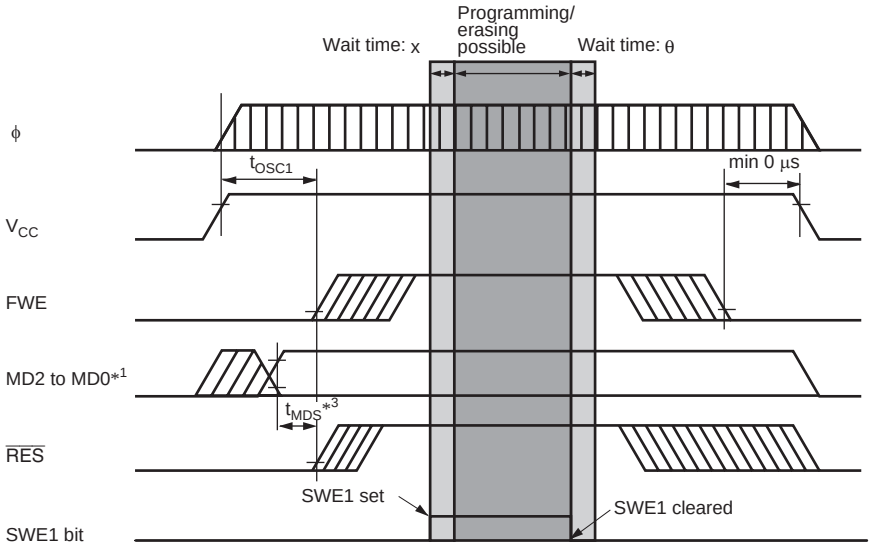



Period during which flash memory access is prohibited
(x: Wait time after setting SWE1 bit)*2

Period during which flash memory can be programmed
(Execution of program in flash memory prohibited, and data reads other than verify operations prohibited)

Notes: 1. Except when switching modes, the level of the mode pins (MD2 to MD0) must be fixed until power-off by pulling the pins up or down.
2. See section 22.7, Flash Memory Characteristics.
3. Mode programming setup time $t_{MDS}$ (min) = 200 ns.

**Figure 17.16 Power-On/Off Timing (Boot Mode)**

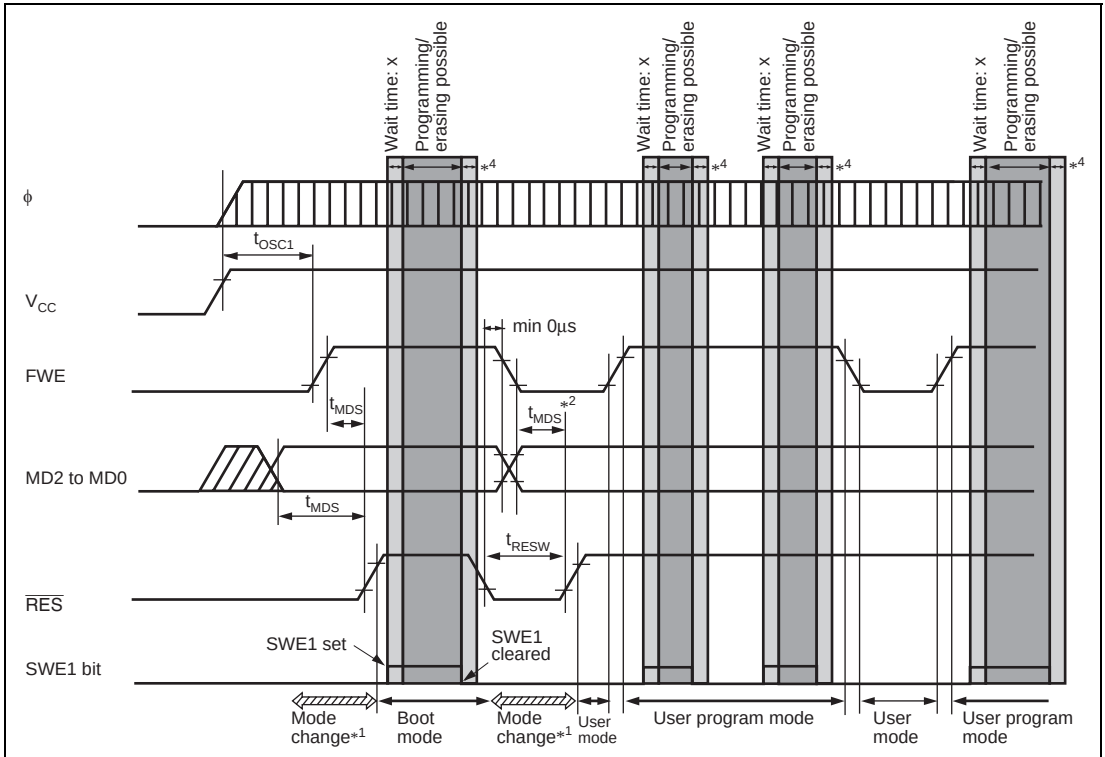



Period during which flash memory access is prohibited
(x: Wait time after setting SWE1 bit)*2

Period during which flash memory can be programmed
(Execution of program in flash memory prohibited, and data reads other than verify operations prohibited)

Notes: 1. Except when switching modes, the level of the mode pins (MD2 to MD0) must be fixed until power-off by pulling the pins up or down.
2. See section 22.7, Flash Memory Characteristics.
3. Mode programming setup time $t_{MDS}$ (min) = 200 ns.

**Figure 17.17 Power-On/Off Timing (User Program Mode)**




Notes: 1. When entering boot mode or making a transition from boot mode to another mode, mode switching must be carried out by means of $\overline{RES}$ input
2. When making a transition from boot mode to another mode, a mode programming setup time $t_{MDS}$ (min) of 200 ns is necessary with respect to $\overline{RES}$ clearance timing.
3. See section 22.7, Flash Memory Characteristics.
4. Wait time: θ.

**Figure 17.18 Mode Transition Timing**
**(Example: Boot Mode → User Mode ↔ User Program Mode)**

## 17.14 Note on Switching from F-ZTAT Version to Masked ROM Version

The masked ROM version does not have the internal registers for flash memory control that are provided in the F-ZTAT version. Table 17.9 lists the registers that are present in the F-ZTAT version but not in the masked ROM version. If a register listed in table 17.9 is read in the masked ROM version, an undefined value will be returned. Therefore, if application software developed on the F-ZTAT version is switched to a masked ROM version product, it must be modified to ensure that the registers in table 17.9 have no effect.

**Table 17.9 Registers Present in F-ZTAT Version but Absent in Masked ROM Version**

| Register | Abbreviation | Address |
|---|---|---|
| Flash memory control register 1 | FLMCR1 | H'FFA8 |
| Flash memory control register 2 | FLMCR2 | H'FFA9 |
| Erase block register 1 | EBR1 | H'FFAA |
| Erase block register 2 | EBR2 | H'FFAB |
| RAM emulation register | RAMER | H'FEDB |
| Serial control register x | SCRX | H'FDB4 |

# Section 18 Masked ROM

This LSI incorporates a masked ROM which has the following features.

## 18.1 Features

- Size

| Product Class | | ROM Size | ROM Address (Modes 6 and 7) |
|---|---|---|---|
| H8S/2218 Group | HD6432217 | 64 kbytes | H'000000 to H'00FFFF |
| H8S/2212 Group | HD6432211 | 64 kbytes | H'000000 to H'00FFFF |
| | HD6432210, HD6432210S | 32 kbytes | H'000000 to H'007FFF |

- Connected to the bus master through 16-bit data bus, enabling one-state access to both byte data and word data.

Figure 18.1 shows a block diagram of the on-chip masked ROM.

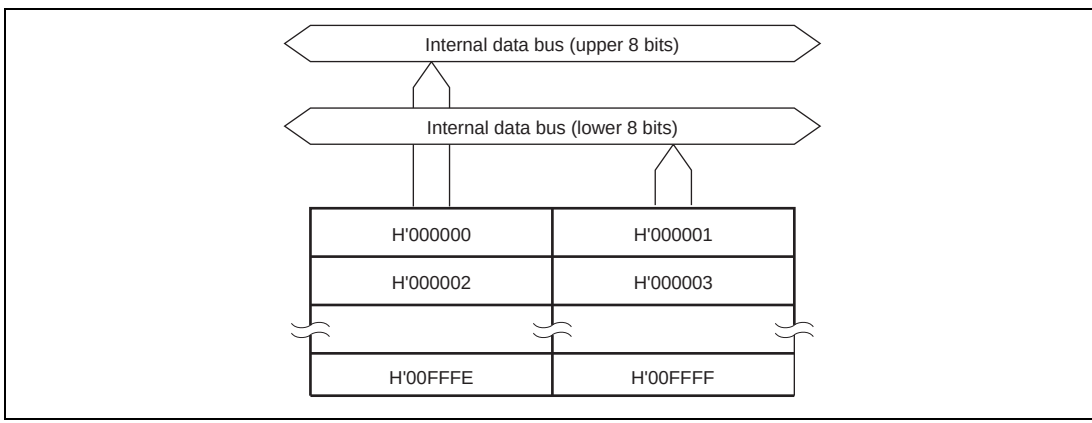



**Figure 18.1 Block Diagram of On-Chip Masked ROM (64 kbytes)**

# Section 19 Clock Pulse Generator

This LSI has an on-chip clock pulse generator that generates the system clock (ϕ), the bus master clock, and internal clocks. The clock pulse generator consists of a main clock oscillator, duty adjustment circuit, clock select circuit, medium-speed clock divider, bus master clock selection circuit, subclock oscillator, waveform shaping circuit, PLL (Phase Locked Loop) circuit, and USB operating clock selection circuit. A block diagram of clock pulse generator is shown in figure 19.1.

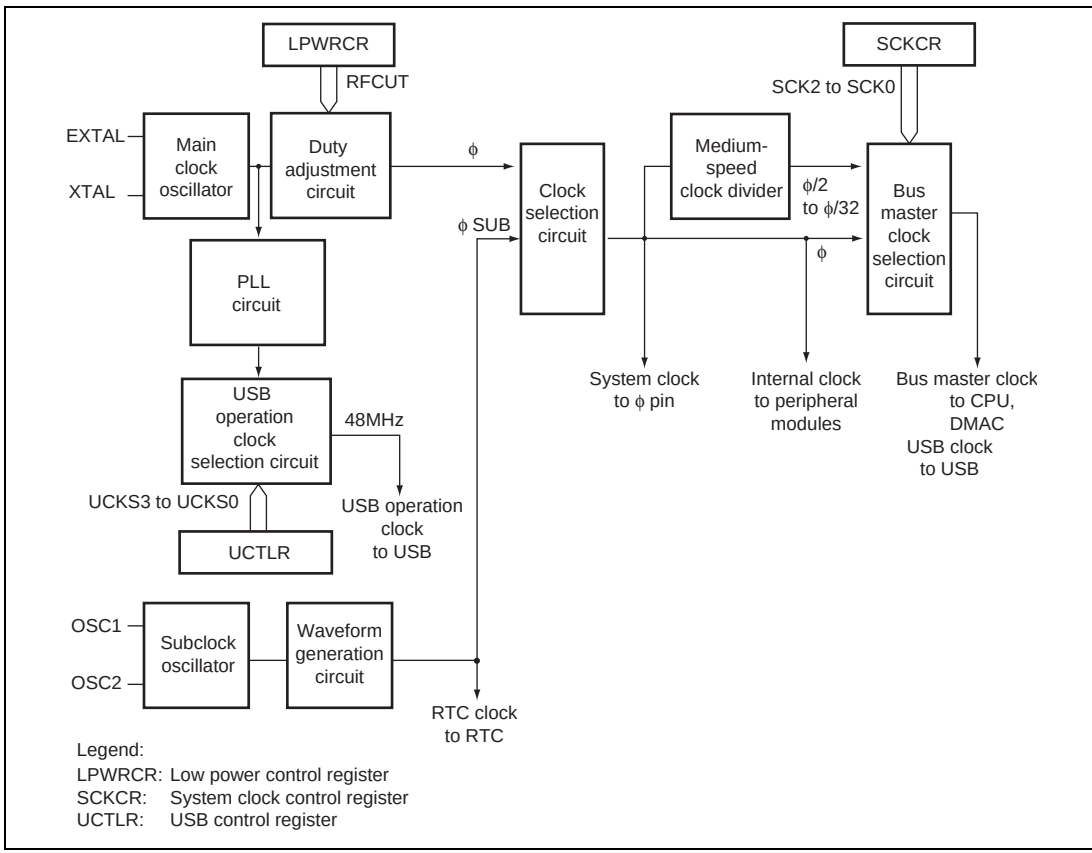



**Figure 19.1 Block Diagram of Clock Pulse Generator**

The frequency of the main clock oscillator can be changed by software by means of settings in the low-power control register (LPWRCR) and system clock control register (SCKCR). PLL 48-MHz clock can be selected by software by means of setting the USB control register (UCTLR). For details, refer to section 14, Universal Serial Bus (USB).

## 19.1 Register Descriptions

The on-chip clock pulse generator has the following registers.

- System clock control register (SCKCR)
- Low-power control register (LPWRCR)

### 19.1.1 System Clock Control Register (SCKCR)

SCKCR controls ϕ clock output and medium-speed mode.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PSTOP | 0 | R/W | ϕ Clock Output Disable<br>Controls ϕ output. The operation of this bit changes depending on the operating mode. For details, see section 20.11, ϕ Clock Output Disabling Function.<br>0: ϕ output, fixed high, or high impedance<br>1: Fixed high or high impedance |
| 6 | — | 0 | R/W | Reserved<br>Although this bit is readable/writable, only 0 should be written to. |
| 5, 4 | — | All 0 | — | Reserved<br>These bits are always read as 0. |
| 3 | — | 0 | R/W | Reserved<br>Although this bit is readable/writable, only 0 should be written to. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 2 | SCK2 | 0 | R/W | System Clock Select 2 to 0 |
| 1 | SCK1 | 0 | R/W | These bits select the bus master clock. To operate in subactive mode or watch mode, clear the SCK2 to SCK0 bits to 0. |
| 0 | SCK0 | 0 | R/W | 000: High-speed mode<br>001: Medium-speed clock is $\phi$/2<br>010: Medium-speed clock is $\phi$/4<br>011: Medium-speed clock is $\phi$/8<br>100: Medium-speed clock is $\phi$/16<br>101: Medium-speed clock is $\phi$/32<br>11×: Setting prohibited |

Legend:
×: Don't care

### 19.1.2 Low Power Control Register (LPWRCR)

LPWRCR performs power-down mode control, selects sampling frequency for eliminating noise, performs subclock oscillator control, and selects whether or not built-in feedback resistance and duty adjustment circuit of the system clock generator used.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | DTON | 0 | R/W | Direct Transition ON Flag<br>0: When the SLEEP instruction is executed in high-speed mode or medium-speed mode, operation shifts to sleep mode, software standby mode, or watch mode*.<br>When the SLEEP instruction is executed in subactive mode, operation shifts to subsleep mode or watch mode.<br>1: When the SLEEP instruction is executed in high-speed mode or medium-speed mode, operation shifts directly to subactive mode, or shifts to sleep mode or software standby mode.<br>When the SLEEP instruction is executed in subactive mode, operation shifts directly to high-speed mode, or shifts to subsleep mode. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 6 | LSON | 0 | R/W | Low Speed ON Flag<br>0: When the SLEEP instruction is executed in high-speed mode or medium-speed mode, operation shifts to sleep mode, software standby mode, or watch mode*.<br>When the SLEEP instruction is executed in subactive mode, operation shifts to watch mode* or shifts directly to high-speed mode.<br>Operation shifts to high-speed mode when watch mode is cancelled.<br>1: When the SLEEP instruction is executed in high-speed mode*, operation shifts to watch mode or subactive mode*.<br>When the SLEEP instruction is executed in subactive mode, operation shifts to subsleep mode or watch mode.<br>Operation shifts to subactive mode when watch mode is cancelled. |
| 5 | NESEL | 0 | R/W | Noise Elimination Sampling Frequency Select<br>This bit selects the sampling frequency of the subclock ($\phi_{SUB}$) generated by the subclock oscillator is sampled by the clock ($\phi$) generated by the system clock oscillator<br>0: Sampling using 1/32 x $\phi$<br>1: Sampling using 1/4 x $\phi$ |
| 4 | SUBSTP | 0 | R/W | Subclock Enable<br>This bit enables/disables subclock generation. This bit should be set to 1 when subclock is not used.<br>0: Enables subclock generation.<br>1: Disables subclock generation. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 3 | RFCUT | 0 | R/W | Built-in Feedback Resistor Control<br>Selects whether the oscillator's built-in feedback resistor and duty adjustment circuit are used with external clock input. This bit should not be accessed when a crystal oscillator is used.<br>After this bit is set when using external clock input, a transition should initially be made to software standby mode. Switching between use and non-use of the oscillator's built-in feedback resistor and duty adjustment circuit is performed when the transition is made to software standby mode.<br>0: Main clock oscillator's built-in feedback resistor and duty adjustment circuit are used<br>1: Main clock oscillator's built-in feedback resistor and duty adjustment circuit are not used |
| 2 | — | 0 | R/W | Reserved<br>This bit can be read from or written to, but the write value should always 0. |
| 1 | STC1 | 0 | R/W | Frequency Multiplication Factor |
| 0 | STC0 | 0 | R/W | Specify the frequency multiplication factor of the PLL circuit incorporated into the evaluation chip. The specified frequency multiplication factor is valid after a transition to software standby mode.<br>With this LSI, the STC1 and STC0 bits must both be set to 1. After a reset, the STC1 and STC0 bits are both cleared to 0, and so they must be set to 1.<br>00: × 1<br>01: × 2 (Setting prohibited)<br>10: × 4 (Setting prohibited)<br>11: PLL is bypassed |

Note: * When watch mode or subactive mode is entered, set high-speed mode.

## 19.2 System Clock Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

### 19.2.1 Connecting a Crystal Resonator

A crystal resonator can be connected as shown in the example in figure 19.2. Select the damping resistance $R_d$ according to table 19.1. An AT-cut parallel-resonance crystal should be used.

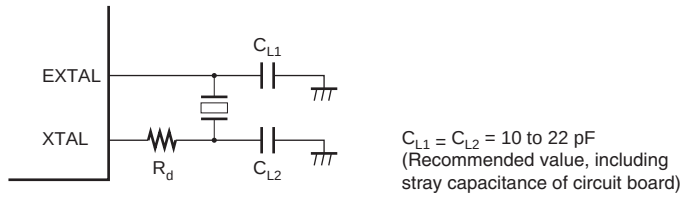



**Figure 19.2 Connection of Crystal Resonator (Example)**

**Table 19.1 Damping Resistance Value**

| Frequency (MHz) | 6 | 8 | 10 | 13 | 16 | 20 | 24 |
|---|---|---|---|---|---|---|---|
| $R_d$ (Ω) | 300 | 200 | 100 | 0 | 0 | 0 | 0 |

Figure 19.3 shows the equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 19.2.

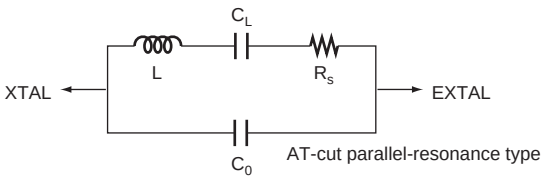



**Figure 19.3 Crystal Resonator Equivalent Circuit**

**Table 19.2 Crystal Resonator Characteristics**

| Frequency (MHz) | 6 | 8 | 10 | 13 | 16 | 20 | 24 |
|---|---|---|---|---|---|---|---|
| $R_S$ max (Ω) | 100 | 80 | 60 | 60 | 50 | 40 | 40 |
| $C_0$ max (pF) | 7 | 7 | 7 | 7 | 7 | 7 | 7 |

### 19.2.2 Inputting External Clock

An external clock signal can be input as shown in an example in figure 19.4. If the XTAL pin is left open, make sure that stray capacitance is no more than 10 pF. When complementary clock input to XTAL pin, the external clock input should be fixed high in standby mode, subactive mode, subsleep mode, or watch mode.

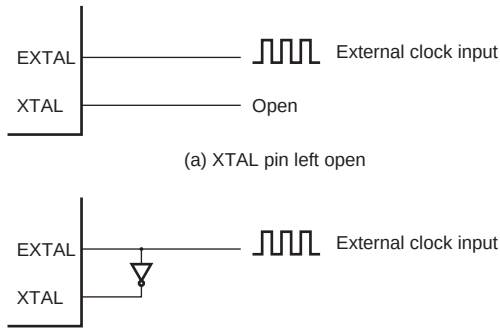



**Figure 19.4 External Clock Input (Examples)**

Table 19.3 shows the input conditions for the external clock.

**Table 19.3 External Clock Input Conditions**

| | | VCC= 2.4 to 3.6V | | VCC = 2.7 to 3.6V | | VCC = 3.0 to 3.6V | | | Test |
|---|---|---|---|---|---|---|---|---|---|
| Item | Symbol | min | max | Min | max | min | max | Unit | Conditions |
| External clock input low pulse width | $t_{EXL}$ | 65 | — | 25 | — | 15.5 | — | ns | Figure 19.5 |
| External clock input high pulse width | $t_{EXH}$ | 65 | — | 25 | — | 15.5 | — | ns | |
| External clock rise time | $t_{EXr}$ | — | 15 | — | 6.25 | — | 5.25 | ns | |
| External clock fall time | $t_{EXf}$ | — | 15 | — | 6.25 | — | 5.25 | ns | |
| Clock low pulse width level | $t_{CL}$ | 0.35 | 0.65 | 0.4 | 0.6 | 0.4 | 0.6 | tcyc | Figure 22.3 |
| Clock high pulse width level | $t_{CH}$ | 0.35 | 0.65 | 0.4 | 0.6 | 0.4 | 0.6 | tcyc | |

The external clock input conditions when the duty adjustment circuit is not used are shown in table 19.4. When the duty adjustment circuit is not used, note that the maximum operating frequency depends on the external clock input waveform. For example, if $t_{EXL} = t_{EXH} = 20.8$ ns and $t_{EXr} = t_{EXf} = 5.25$ ns, the maximum operating frequency becomes 19.2 MHz depending on the clcok cycle time of 52.1 ns.

**Table 19.4 External Clock Input Conditions when Duty Adjustment Circuit Is not Used**

| Item | Symbol | min | max | Min | max | min | max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|---|---|
| External clock input low pulse width | $t_{EXL}$ | 80 | — | 31.25 | — | 20.8 | — | ns | Figure 19.5 |
| External clock input high pulse width | $t_{EXH}$ | 80 | — | 31.25 | — | 20.8 | — | ns | |
| External clock rise time | $t_{EXr}$ | — | 15 | — | 6.25 | — | 5.25 | ns | |
| External clock fall time | $t_{EXf}$ | — | 15 | — | 6.25 | — | 5.25 | ns | |

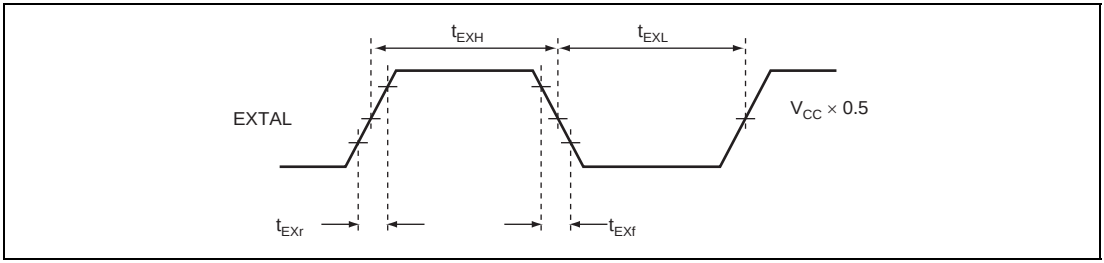



**Figure 19.5 External Clock Input Timing**

## 19.3 Duty Adjustment Circuit

When the oscillator frequency is 5 MHz or higher, the duty adjustment circuit adjusts the duty cycle of the clock signal from the oscillator to generate the system clock (ϕ).

## 19.4 Medium-Speed Clock Divider

The medium-speed clock divider divides the system clock to generate ϕ/2, ϕ/4, ϕ/8, ϕ/16, and ϕ/32.

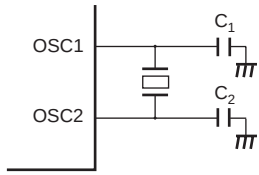
## 19.5 Bus Master Clock Selection Circuit

The bus master clock selection circuit selects the clock supplied to the bus master by setting the bits SCK2 to SCK0 in SCKCR. The bus master clock can be selected from high-speed mode, or medium-speed clocks (ϕ/2, ϕ/4, ϕ/8, ϕ/16, ϕ/32).

## 19.6 Subclock Oscillator

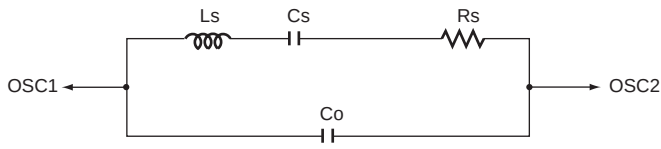
### 19.6.1 Connecting 32.768-kHz Crystal Resonator

supply a clock to the subclock divider, connect a 32.768-kHz crystal resonator, as shown in figure 19.6. Figure 19.7 shows the equivalence circuit for a 32.768-kHz oscillator.




Note: $C_1$ and $C_2$ are reference values including the floating capacitance of the board.

**Figure 19.6 Example Connection of 32.768-kHz Quartz Oscillator**




Co = 1.5 pF (typ.)
Rs = 14 kΩ (typ.)
fw = 32.768 kHz
Type name = C001R (SEIKO EPSON)

**Figure 19.7 Equivalence Circuit for 32.768-kHz Oscillator**

### 19.6.2 Handling Pins when Subclock Not Required

If no subclock is required, connect the OSC1 pin to Vss and leave OSC2 open, as shown in figure 19.8. Set the SUBSTP bit of LPWRCR to 1. If this setting is not made, transition to power-down mode may not complete properly in some cases.

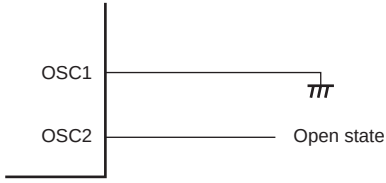



**Figure 19.8 Pin Handling when Subclock Not Required**

## 19.7 Subclock Waveform Generation Circuit

To eliminate noise from the subclock input to OSCI, the subclock is sampled using the dividing clock ϕ. The sampling frequency is set using the NESEL bit of LPWRCR. For details, see section 19.1.2, Low Power Control Register (LPWRCR).

No sampling is performed in subactive mode, subsleep mode, or watch mode.

## 19.8 PLL Circuit for USB

The PLL circuit has the function of doubling or tripling the 16-MHz or 24-MHz clock from the main oscillator to generate the 48-MHz USB operating clock.

When the PLL circuit is used, set the UCKS3 to UCKS0 bits of UCTLR. For details, refer to section 14, Universal Serial Bus (USB).

When the PLL circuit is not used, connect the PLVCC pin to Vcc and the PLLVSS pin to the ground (Vss). Figure 19.9 shows examples of external circuits peripheral to the PLL.

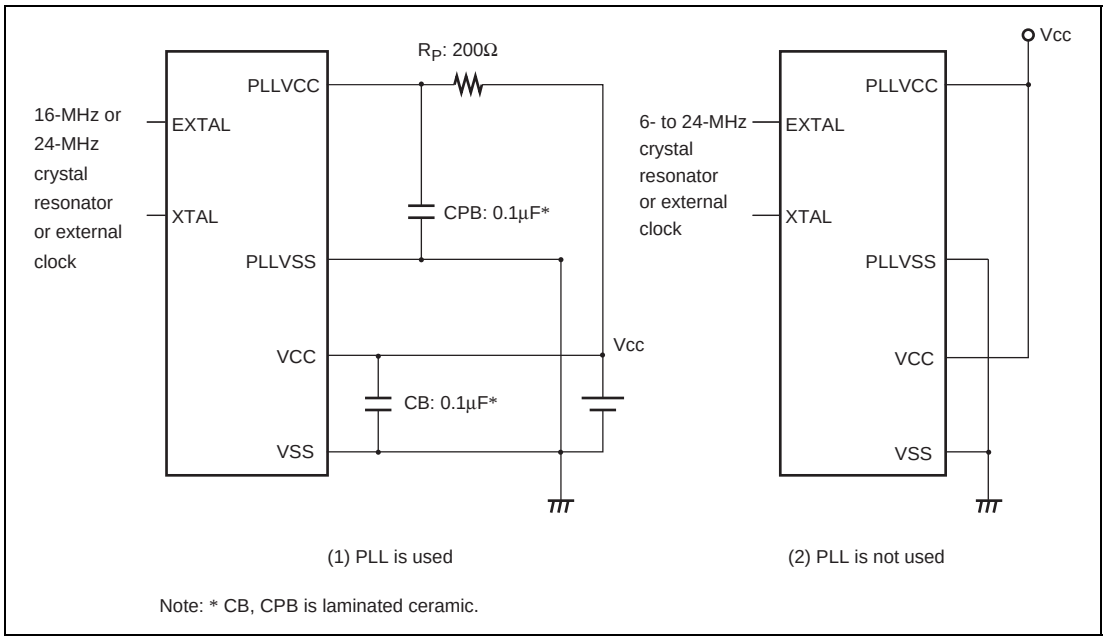



**Figure 19.9 Example of PLL Circuit**

## 19.9 Usage Notes

### 19.9.1 Note on Crystal Resonator

Since various characteristics related to the crystal resonator are closely linked to the user's board design, thorough evaluation is necessary on the user's part, using the resonator connection examples shown in this section as a guide. As the resonator circuit ratings will depend on the floating capacitance of the resonator and the mounting circuit, the ratings should be determined in consultation with the resonator manufacturer. The design must ensure that a voltage exceeding the maximum rating is not applied to the oscillator pin.

### 19.9.2 Note on Board Design

When designing the board, place the crystal resonator and its load capacitors as close as possible to the XTAL or OSC1 and EXTAL or OSC2 pins. Other signal lines should be routed away from the oscillator circuit to prevent induction from interfering with correct oscillation. See figure 19.10.

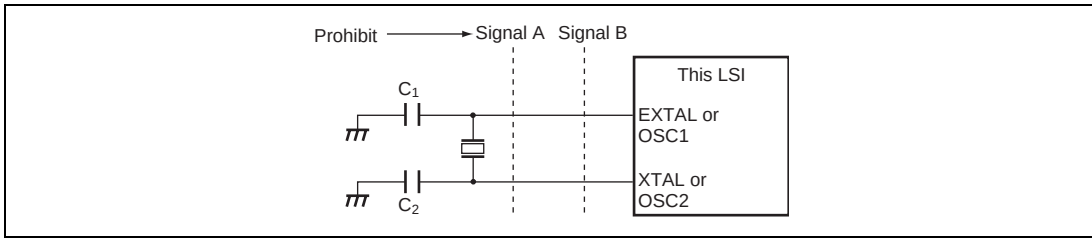



**Figure 19.10 Note on Board Design of Oscillator Circuit**

### 19.9.3 Note on Switchover of External Clock

When two or more external clocks (e.g. 16 MHz and 13 MHz) are used as the system clock, switchover of the input clock should be carried out in software standby mode.

An example of an external clock switching circuit is shown in figure 19.11, and an example of the external clock switchover timing in figure 19.12.

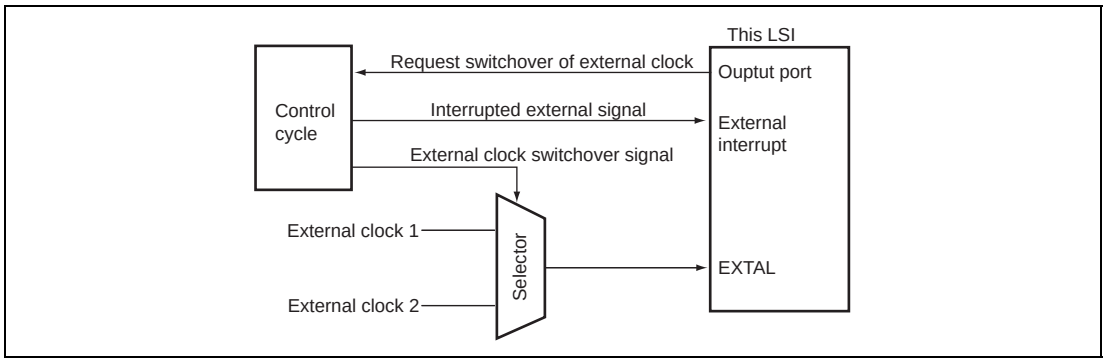
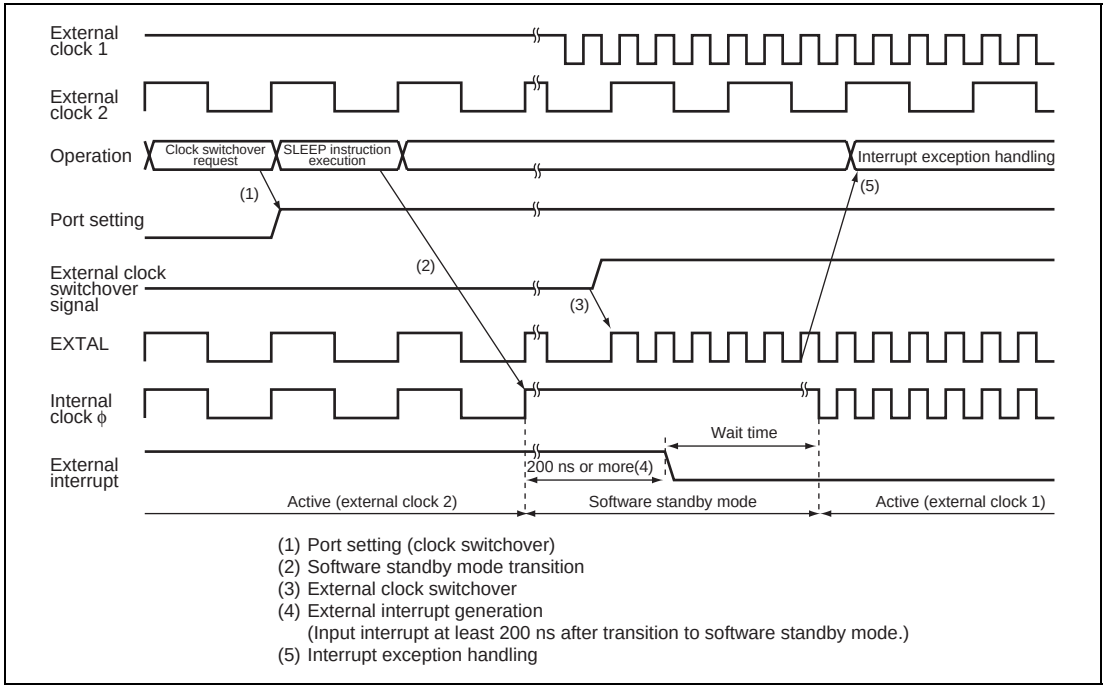



**Figure 19.11 Example of External Clock Switching Circuit**




(1) Port setting (clock switchover)
(2) Software standby mode transition
(3) External clock switchover
(4) External interrupt generation
(Input interrupt at least 200 ns after transition to software standby mode.)
(5) Interrupt exception handling

**Figure 19.12 Example of External Clock Switchover Timing**

# Section 20 Power-Down Modes

In addition to the normal program execution state, this LSI has five power-down modes in which operation of the CPU and oscillator is halted and power dissipation is reduced. Low-power operation can be achieved by individually controlling the CPU, on-chip supporting modules, and so on.

This LSI's operating modes are high-speed mode and five power down modes:

1. Medium-speed mode
2. Subactive mode
3. Sleep mode
4. Subsleep mode
5. Watch mode
6. Module stop mode
7. Software standby mode
8. Hardware standby mode

1. to 5. are power-down modes. Sleep mode is CPU states, medium-speed mode is a CPU and bus master state, subactive mode is a CPU, bus master, and on-chip peripheral function state, and module stop mode is an internal peripheral function (including bus masters other than the CPU) state. Some of these states can be combined.

After a reset, the LSI is in high-speed mode, flash memory, and module stop mode (other than DMAC).

Table 20.1 and table 20.2 show the LSI internal states in each mode and the transition conditions of power-down modes respectively. Figure 20.1 shows the diagram of mode transition.

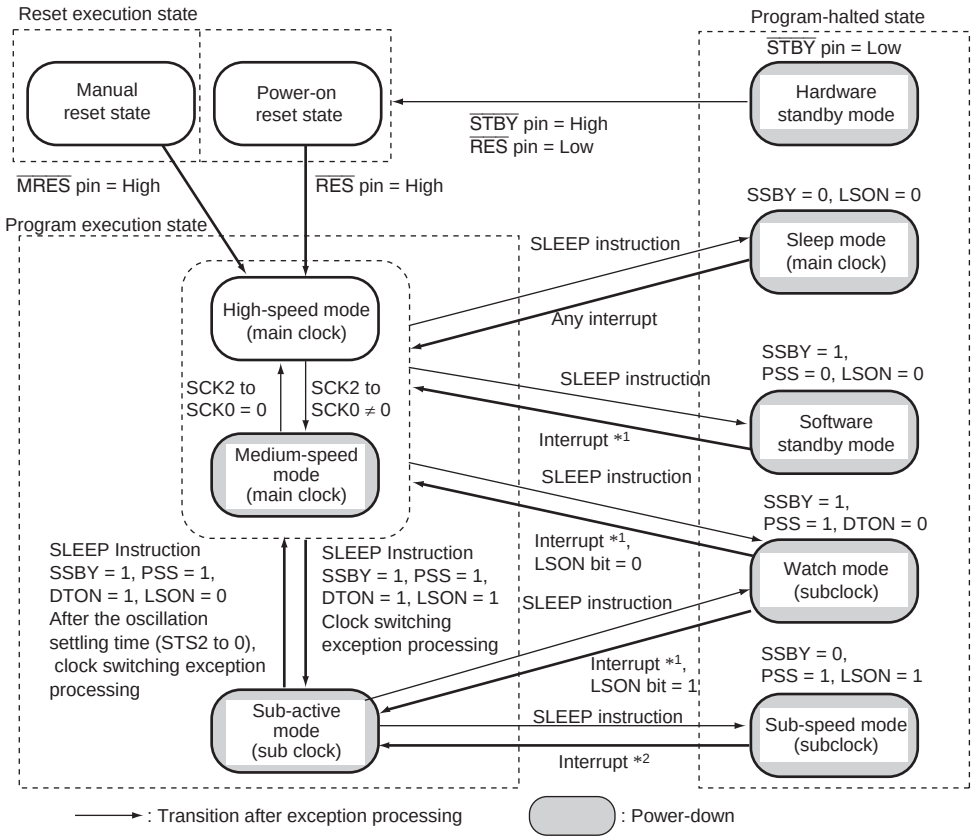
## Table 20.1 LSI Internal States in Each Mode

| Function | | | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby |
|---|---|---|---|---|---|---|---|---|---|---|---|
| System clock pulse generator | | | Functioning | Functioning | Functioning | Functioning | Halted | Halted | Halted | Halted | Halted |
| Subclock pulse generator | | | Functioning/halted | Functioning/halted | Functioning/halted | Functioning/halted | Functioning | Functioning | Functioning | Functioning/halted | Halted |
| CPU | Instructions | | Functioning | Medium-speed operation | Halted | Functioning | Halted | Subclock operation | Halted | Halted | Halted |
| | Registers | | | | Retained | | Retained | | Retained | Retained | Undefined |
| RAM | | | Functioning | Functioning | Functioning | Functioning | Retained | Functioning | Retained | Retained | Retained |
| I/O | | | Functioning | Functioning | Functioning | Functioning | Retained | Functioning | Functioning | Halted | High impedance |
| External interrupts | NMI<br>IRQ0 to IRQ4, IRQ7 | | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Halted |
| Peripheral functions | DMAC | | Functioning | Medium-speed operation | Functioning | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (reset) |
| | WDT | | Functioning | Functioning | Functioning | Functioning | Halted (retained) | Subclock operation | Subclock operation | Halted (retained) | Halted (reset) |
| | RTC | Clock operation | Subclock operation | Subclock operation | Subclock operation | Halted (retained) | Subclock operation | Subclock operation | Subclock operation | Subclock functioning/ halted | Halted (reset) |
| | | Free-running timer operation | Functioning | Functioning | Functioning | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (reset) |
| | TPU | | Functioning | Functioning | Functioning | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (reset) |
| | SCI | | | | | | | | | | |
| | A/D | | Functioning | Functioning | Functioning | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) |
| | USB | | Functioning | Function not guaranteed | Functioning | Halted (retained) | Function not guaranteed.<br>Always select module stop mode. | | | Halted (retained) | Halted (reset) |
| | | PLL circuit | | | | Halted | | | | Halted | |

Notes: "Halted (retained)" means that internal register values are retained. The internal state is "operation suspended."

"Halted (reset)" means that internal register values and internal states are initialized.

In module stop mode, only modules for which a stop setting has been made are halted (reset or retained).




Notes: When a transition is made between modes by means of an interrupt, the transition cannot be made on interrupt source generation alone. Ensure that interrupt handling is performed after accepting the interrupt request.
From any state except hardware standby mode, a transition to the power-on reset state occurs when $\overline{\text{RES}}$ is driven low. From any state except hardware standby mode and power-on reset, a transition to the manual reset state occurs when $\overline{\text{MRES}}$ is driven low.
From any state, a transition to hardware standby mode occurs when $\overline{\text{STBY}}$ is driven low.
Always select high-speed mode before making a transition to watch mode or subactive mode.

1. NMI and IRQ0 to IRQ4, IRQ7, RTC interrupt, and USB suspend/resume interrupt
2. NMI and IRQ0 to IRQ4, IRQ7, RTC interrupt, and WDT interrupt

**Figure 20.1 Mode Transition Diagram**

**Table 20.2 Transition Conditions of Power-Down Modes**

| Pre-Transition State | Status of Control Bit at Transition | | | | State after Transition Invoked by SLEEP Command | State after Transition Back from Power-Down Mode Invoked by Interrupt |
|---|---|---|---|---|---|---|
| | SSBY | PSS | LSON | DTON | | |
| High-speed/ Medium-speed | 0 | × | 0 | × | Sleep | High-speed/Medium-speed |
| | 0 | × | 1 | × | — | — |
| | 1 | 0 | 0 | × | Software standby | High-speed/Medium-speed |
| | 1 | 0 | 1 | × | — | — |
| | 1 | 1 | 0 | 0 | Watch | High-speed |
| | 1 | 1 | 1 | 0 | Watch | Subactive |
| | 1 | 1 | 0 | 1 | — | — |
| | 1 | 1 | 1 | 1 | Subactive | — |
| Subactive | 0 | 0 | × | × | — | — |
| | 0 | 1 | 0 | × | — | — |
| | 0 | 1 | 1 | × | Sub sleep | Subactive |
| | 1 | 0 | × | × | — | — |
| | 1 | 1 | 0 | 0 | Watch | High-speed |
| | 1 | 1 | 1 | 0 | Watch | Subactive |
| | 1 | 1 | 0 | 1 | High-speed | — |
| | 1 | 1 | 1 | 1 | — | — |

Legend:

×: Don't care

—: Do not set

## 20.1 Register Descriptions

The registers relating to the power down mode are shown below. For details on the low power control register (LPWRCR), refer to section 19.1.2, Low Power Control Register (LPWRCR). For details on the system clock control register (SCKCR), refer to section 19.1.1, System Clock Control Register (SCKCR).

- Standby control register (SBYCR)
- System clock control register (SCKCR)
- Low power control register (LPWRCR)
- Timer control/status register (TCSR_1)
- Module stop control register A (MSTPCRA)
- Module stop control register B (MSTPCRB)
- Module stop control register C (MSTPCRC)
- Extended module stop register (EXMDLSTP)

### 20.1.1 Standby Control Register (SBYCR)

SBYCR performs software standby mode control.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | SSBY | 0 | R/W | Software Standby<br>This bit specifies the transition mode after executing the SLEEP instruction<br>0: Shifts to sleep mode when the SLEEP instruction is executed in high-speed mode or medium-speed mode.<br>Shifts to subsleep mode when the SLEEP instruction is executed in subactive mode.<br>1: Shifts to software standby mode, subactive mode, or watch mode when the SLEEP instruction is executed in high-speed mode or medium-speed mode.<br>Shifts to watch mode or high-speed mode when the SLEEP instruction is executed in subactive mode.<br>This bit does not change when clearing software standby mode by using external interrupts and shifting to normal operation. 0 should be written to this bit for clearing. |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 6 | STS2 | 0 | R/W | Standby Timer Select 2 to 0 |
| 5 | STS1 | 0 | R/W | These bits select the MCU wait time for clock stabilization when cancel software standby mode, watch mode, or subactive mode by an external interrupt. With a crystal oscillator (tables 20.3 and 20.4), select a wait time of $t_{OSC2}$ ms (oscillation stabilization time) or more, depending on the operating frequency. With an external clock, there are no specific wait requirements. However, in the F-ZTAT version a standby time of 16 wait states cannot be used with an external clock. In this case, select a wait time of 100 μs or more.<br>000: Standby time = 8192 states<br>001: Standby time = 16384 states<br>010: Standby time = 32768 states<br>011: Standby time = 65536 states<br>100: Standby time = 131072 states<br>101: Standby time = 262144 states<br>110: Standby time = 2048 states<br>111: Standby time = 16 states |
| 4 | STS0 | 0 | R/W | |
| 3 | OPE | 1 | R/W | Output Port Enable<br>This bit selects whether address bus and bus control signals ($\overline{CS0}$ to $\overline{CS7}$, $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$) are brought to high impedance state or retained in software standby mode, watch mode, or direct transition.<br>0: High impedance state<br>1: Retained |
| 2 to 0 | — | All 0 | — | Reserved<br>These bits are always read as 0, and cannot be modified. |

### 20.1.2 Timer Control/Status Register (TCSR_1)

TCSR_1 controls the operation in power-down mode transition.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 5 | — | All 0 | — | Reserved<br>The write value should always be 0. |
| 4 | PSS | 0 | R/W | Prescaler Select<br>0: When the SLEEP instruction is executed in high-speed mode or medium-speed mode, operation shifts to sleep mode or software standby mode.<br>1: When the SLEEP instruction is executed in high-speed mode or medium-speed mode, operation shifts to sleep mode, watch mode, or subactive mode.<br>When the SLEEP instruction is executed in subactive mode, operation shifts to subsleep mode, watch mode, or high-speed mode<br>TCSR_1 differs from other registers in being more difficult to write to. The procedures for writing to and reading this register are given below.<br>Write:<br>TCSR_1 must be written to by a word transfer instruction. The upper byte of the written word must contain H′A5 and the lower byte must contain the write data. (When the PSS bit is set to 1, the upper byte of the written word must contain H'A510.)<br>Read:<br>TCSR_1 is read by the same procedure as for the general registers. |
| 3 to 0 | — | All 0 | — | Reserved<br>The write value should always be 0. |

### 20.1.3 Module Stop Control Registers A to C (MSTPCRA to MSTPCRC)

MSTPCR, comprising three 8-bit readable/writable registers, performs module stop mode control. Setting a bit to 1, causes the corresponding module to enter module stop mode, while clearing the bit to 0 clears the module stop mode.

**MSTPCRA**

| Bit | Bit Name | Initial Value | R/W | Module |
|---|---|---|---|---|
| 7 | MSTPA7 | 0 | R/W | DMA controller (DMAC) |
| 6 | MSTPA6* | 0 | R/W | — |
| 5 | MSTPA5 | 1 | R/W | 16-bit timer pulse unit (TPU) |
| 4 | MSTPA4* | 1 | R/W | — |
| 3 | MSTPA3* | 1 | R/W | — |
| 2 | MSTPA2* | 1 | R/W | — |
| 1 | MSTPA1 | 1 | R/W | A/D converter |
| 0 | MSTPA0* | 1 | R/W | — |

**MSTPCRB**

| Bit | Bit Name | Initial Value | R/W | Module |
|---|---|---|---|---|
| 7 | MSTPB7 | 1 | R/W | Serial communication interface 0 (SCI_0) |
| 6 | MSTPB6* | 1 | R/W | — |
| 5 | MSTPB5 | 1 | R/W | Serial communication interface 2 (SCI_2) |
| 4 | MSTPB4* | 1 | R/W | — |
| 3 | MSTPB3* | 1 | R/W | — |
| 2 | MSTPB2* | 1 | R/W | — |
| 1 | MSTPB1* | 1 | R/W | — |
| 0 | MSTPB0 | 1 | R/W | USB |

**MSTPCRC**

| Bit | Bit Name | Initial Value | R/W | Module |
|---|---|---|---|---|
| 7 | MSTPC7* | 1 | R/W | — |
| 6 | MSTPC6* | 1 | R/W | — |
| 5 | MSTPC5* | 1 | R/W | — |
| 4 | MSTPC4* | 1 | R/W | — |
| 3 | MSTPC3* | 1 | R/W | — |
| 2 | MSTPC2* | 1 | R/W | — |
| 1 | MSTPC1 | 0 | R/W | Flash Memory (This bit is reserved in the masked ROM version; setting is disabled.)<br>Note: Setting of the flash memory module stop mode should be carried out while the programs in the on-chip RAM and external memory are executed. If the flash memory is stopped with the programming in the flash memory, the program after setting module stop mode stops and enters to the deadlock state.<br>Figure 20.2 shows the example of using module stop mode. |
| 0 | MSTPC0* | 1 | R/W | — |

Note: * MSTPA6 are readable/writable bits with an initial value of 0 and should always be written with 1.
MSTPA4 to MSTPA2, MSTPA0, MSTPB6, MSTPB4 to MSTPB1, MSTPC7 to MSTPC2, MSTPC0 are readable/writable bits with an initial value of 1 and should always be written with 1.

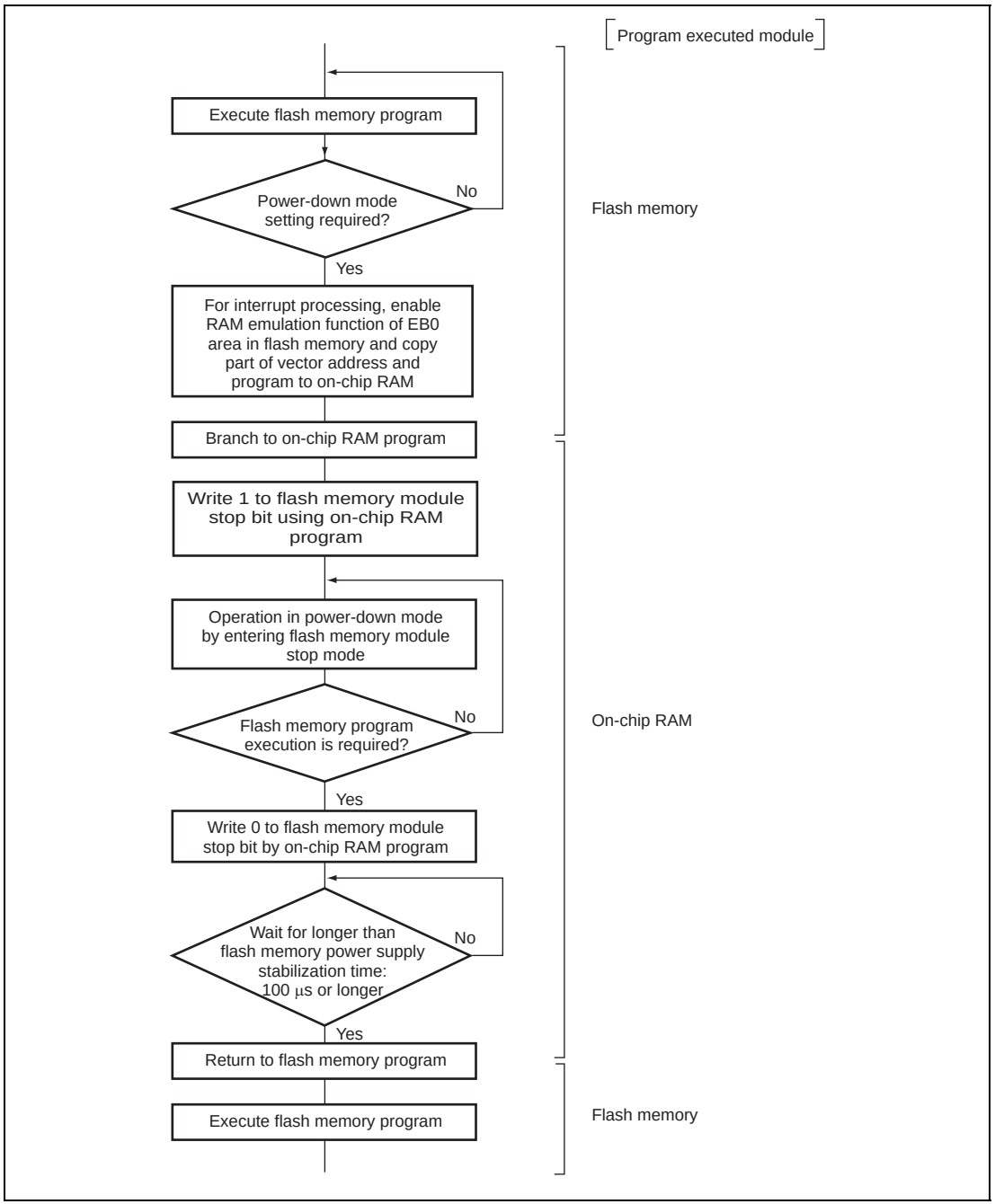



**Figure 20.2 Example of Flash Memory Module Stop Mode Usage**

### 20.1.4 Extended Module Stop Register (EXMDLSTP)

EXMDLSTP controls the clock supply of the RTC and USB, performs module stop mode control. Setting a bit to 1, causes the corresponding module to enter module stop mode, while clearing the bit to 0 clears the module stop mode.

| Bit | Bit Name | Initial Value | R/W | Module |
|---|---|---|---|---|
| 7 to 2 | — | Undefined | — | Reserved<br>Read is undefined. These bits should not to be modified. |
| 1 | RTCSTOP | 0 | R/W | RTC |
| 0 | USBSTOP1 | 0 | R/W | USB |

## 20.2 Medium-Speed Mode

When the SCK2 to SCK0 bits in SCKCR are set to 1, the operating mode changes to medium-speed mode as soon as the current bus cycle ends. In medium-speed mode, the CPU operates on the operating clock ($\phi/2$, $\phi/4$, $\phi/8$, $\phi/16$, or $\phi/32$) specified by the SCK2 to SCK0 bits. The bus masters other than the CPU (DMAC) also operate in medium-speed mode. On-chip supporting modules other than the bus masters always operate on the high-speed clock ($\phi$).

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if $\phi/4$ is selected as the operating clock, on-chip memory is accessed in 4 states, and internal I/O registers in 8 states.

Medium-speed mode is cleared by clearing all of bits SCK2 to SCK0 to 0. A transition is made to high-speed mode and medium-speed mode is cleared at the end of the current bus cycle.

If a SLEEP instruction is executed when the SSBY bit in SBYCR, the LSON bit in LPWRCR are cleared to 0, a transition is made to sleep mode. When sleep mode is cleared by an interrupt, medium-speed mode is restored.

When the SLEEP instruction is executed with the SSBY bit = 1 and the LSON bit, and the PSS bit in TCSR_1 are cleared to 0, operation shifts to the software standby mode. When software standby mode is cleared by an external interrupt, medium-speed mode is restored.

When the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin is set low and medium-speed mode is cancelled, operation shifts to the reset state. The same applies in the case of a reset caused by overflow of the watchdog timer.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Figure 20.3 shows the timing for transition to and clearance of medium-speed mode.

Note: * Supported only by the H8S/2218 Group.

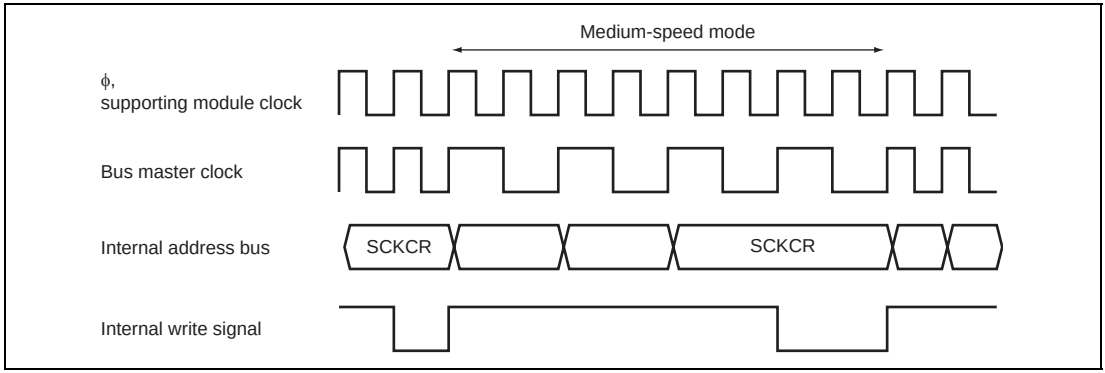



**Figure 20.3 Medium-Speed Mode Transition and Clearance Timing**

## 20.3 Sleep Mode

### 20.3.1 Transition to Sleep Mode

When the SLEEP instruction is executed when the SSBY bit in SBYCR and the LSON bit in LPWRCR are cleared to 0, the CPU enters the sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other supporting modules do not stop.

### 20.3.2 Exiting Sleep Mode

Sleep mode is exited by any interrupt, or signals at the $\overline{\text{RES}}$, $\overline{\text{MRES}}$*, or $\overline{\text{STBY}}$ pin.

- Exiting Sleep Mode by Interrupts
  When an interrupt occurs, sleep mode is exited and interrupt exception processing starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.
- Exiting Sleep Mode by $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* Pin
  Setting the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin level Low selects the reset state. After the stipulated reset input duration, driving the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin High starts the CPU performing reset exception processing.
- Exiting Sleep Mode by $\overline{\text{STBY}}$ Pin
  When the $\overline{\text{STBY}}$ pin level is driven Low, a transition is made to hardware standby mode.

Note: * Supported only by the H8S/2218 Group.

## 20.4 Software Standby Mode

### 20.4.1 Transition to Software Standby Mode

A transition is made to software standby mode when the SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1 and the LSON bit in LPWRCR and the PSS bit in TCSR_1 are cleared to 0. In this mode, the CPU, on-chip supporting modules, and oscillator all stop. However, the contents of the CPU's internal registers, RAM data, and the states of on-chip supporting modules other than the A/D converter, and the states of I/O ports, are retained. In this mode the oscillator stops, and therefore power dissipation is significantly reduced.

### 20.4.2 Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, $\overline{\text{IRQ7}}$ pin, or $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ4}}$ pins), RTC interrupt ($\overline{\text{IRQ5}}$ signal), or USB suspend/resume interrupt ($\overline{\text{IRQ6}}$ signal), or by means of the $\overline{\text{RES}}$ pin, $\overline{\text{MRES}}$ pin*, or $\overline{\text{STBY}}$ pin.

- Clearing with an interrupt

  When an NMI or IRQ0 to IRQ7 interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SBYCR, stable clocks are supplied to the entire chip, software standby mode is cleared, and interrupt exception handling is started.

  When clearing software standby mode with an IRQ0 to IRQ7 interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ5 is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side.
- Clearing with the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin

  When the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire chip. Note that the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin must be held low until clock oscillation stabilizes. When the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin goes high, the CPU begins reset exception handling.
- Clearing with the $\overline{\text{STBY}}$ pin

  When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Note: * Supported only by the H8S/2218 Group.

### 20.4.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode

Bits STS2 to STS0 in SBYCR should be set as described below.

- Using a Crystal Oscillator:
  Set bits STS2 to STS0 so that the standby time is at least $t_{OSC2}$ ms (the oscillation stabilization time).
  Table 20.3 shows the standby times for different operating frequencies and settings of bits STS2 to STS0.
- Using an External Clock
  Set bits STS2 to STS0 as any value. Usually, minimum value is recommended. A standby time of 100 μs or longer (flash memory power supply stabilization time) should be used in the F-ZTAT version.

**Table 20.3 Oscillation Stabilization Time Settings**

| STS2 | STS1 | STS0 | Standby Time | 24MHz | 20MHz | 16MHz | 13MHz | 10MHz | 8MHz | 6MHz | 4MHz | 2MHz | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 8192 states | 0.34 | 0.41 | 0.51 | 0.63 | 0.82 | 1.0 | 1.4 | 2.0 | 4.1 | ms |
| | | 1 | 16384 states | 0.68 | 0.82 | 1.0 | 1.3 | 1.6 | 2.0 | 2.7 | 4.1 | 8.2 | |
| | 1 | 0 | 32768 states | 1.4 | 1.6 | 2.0 | 2.5 | 3.3 | 4.1 | 5.5 | 8.2 | 16.4 | |
| | | 1 | 65536 states | 2.7 | 3.3 | 4.1 | 5.0 | 6.6 | 8.2 | 10.9 | 16.4 | 32.8 | |
| 1 | 0 | 0 | 131072 states | 5.5 | 6.6 | 8.2 | 10.1 | 13.1 | 16.4 | 21.8 | 32.8 | 65.5 | |
| | | 1 | 262144 states | 10.9 | 13.1 | 16.4 | 20.2 | 26.2 | 32.8 | 43.7 | 65.5 | 131.1 | |
| | 1 | 0 | 2048 states | 0.09 | 0.10 | 0.13 | 0.16 | 0.20 | 0.26 | 0.34 | 0.51 | 1.0 | |
| | | 1 | 16 states | 0.67 | 0.80 | 1.0 | 1.2 | 1.6 | 2.0 | 2.7 | 4.0 | 8.0 | μs |

: Recommended time setting (For conditions, see $t_{OSC2}$ in table 22.4.)

### 20.4.4 Software Standby Mode Application Example

Figure 20.4 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in SYSCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.

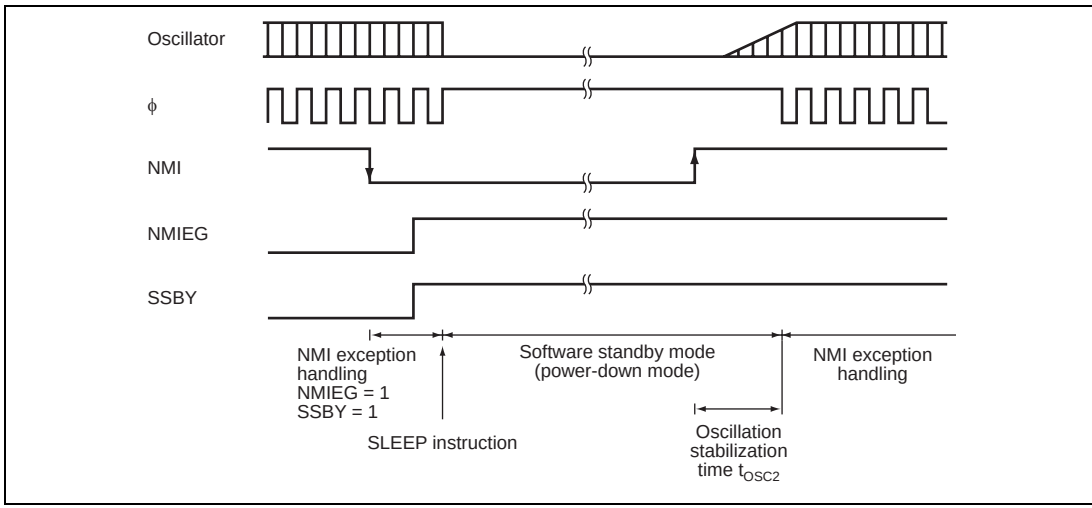



**Figure 20.4 Software Standby Mode Application Example**

## 20.5 Hardware Standby Mode

### 20.5.1 Transition to Hardware Standby Mode

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power dissipation. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the $\overline{\text{STBY}}$ pin low.

Do not change the state of the mode pins (MD2 to MD0) while this LSI is in hardware standby mode.

### 20.5.2 Clearing Hardware Standby Mode

Hardware standby mode is cleared by means of the $\overline{\text{STBY}}$ pin and the $\overline{\text{RES}}$ pin. When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, the reset state is set and clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until the clock oscillator stabilizes (at least $t_{osc1}$—the oscillation stabilization time—when using a crystal oscillator). When the $\overline{\text{RES}}$ pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

### 20.5.3 Hardware Standby Mode Timing

Figure 20.5 shows an example of hardware standby mode timing.

When the $\overline{\text{STBY}}$ pin is driven low after the $\overline{\text{RES}}$ pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the $\overline{\text{STBY}}$ pin high, waiting for the oscillation stabilization time, then changing the $\overline{\text{RES}}$ pin from low to high.

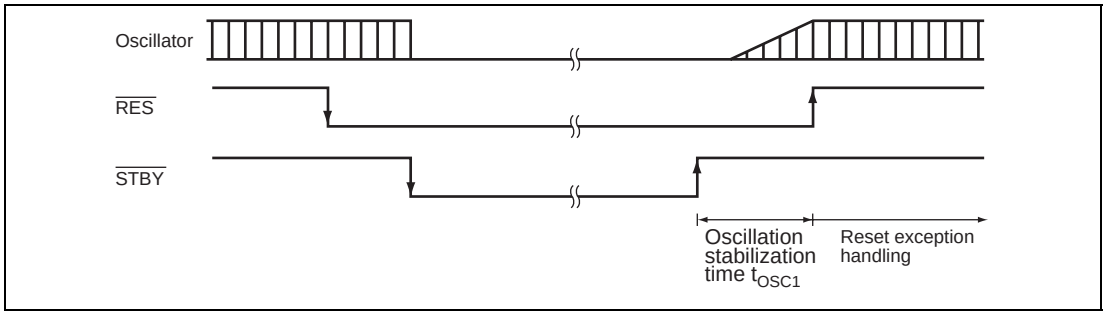



**Figure 20.5 Hardware Standby Mode Timing (Example)**

### 20.5.4 Hardware Standby Mode Timings

**Timing of Transition to Hardware Standby Mode:**

1. To retain RAM contents with the RAME bit set to 1 in SYSCR
   Drive the $\overline{\text{RES}}$ signal low at least 10 states before the $\overline{\text{STBY}}$ signal goes low, as shown in figure 20.6. After $\overline{\text{STBY}}$ has gone low, $\overline{\text{RES}}$ has to wait for at least 0 ns before becoming high.

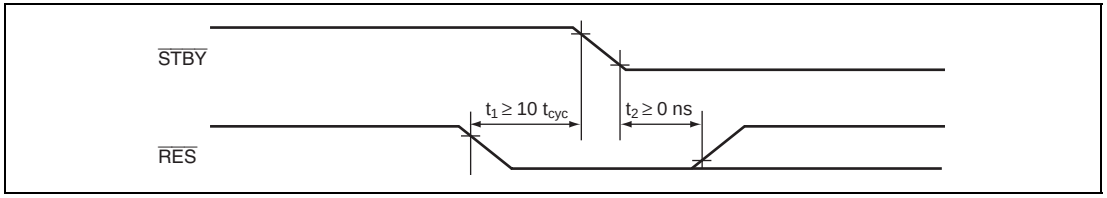



**Figure 20.6 Timing of Transition to Hardware Standby Mode**

2. To retain RAM contents with the RAME bit cleared to 0 in SYSCR, or when RAM contents do not need to be retained
   $\overline{\text{RES}}$ does not have to be driven low as in the above case.

**Timing of Recovery from Hardware Standby Mode:**

Drive the $\overline{RES}$ signal low approximately 100 ns or more before $\overline{STBY}$ goes high to execute a power-on reset.

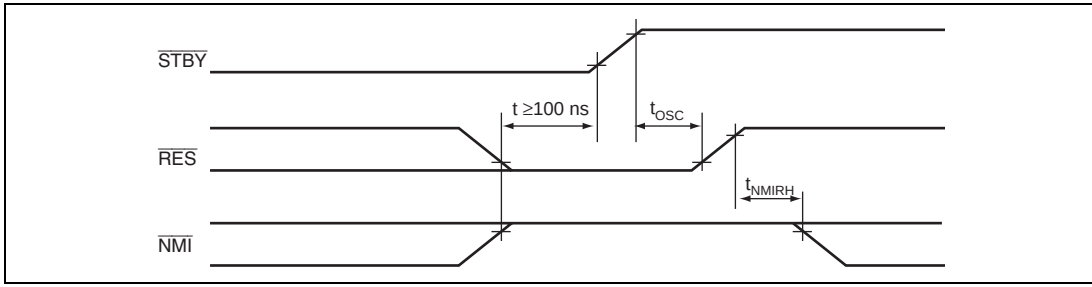



**Figure 20.7 Timing of Recovery from Hardware Standby Mode**

## 20.6 Module Stop Mode

Module stop mode can be set for individual on-chip supporting modules.

When the corresponding MSTP bit in MSTPCR is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

When the corresponding MSTP bit is cleared to 0, module stop mode is cleared and the module starts operating at the end of the bus cycle. In module stop mode, the internal states of modules other than the A/D converter are retained.

After reset clearance, all modules other than DMAC and flash memory are in module stop mode.

When an on-chip supporting module is in module stop mode, read/write access to its registers is disabled.

When a transition is made to sleep mode with all modules stopped, the bus controller and I/O ports also stop operating, enabling current dissipation to be further reduced.

## 20.7 Watch Mode

### 20.7.1 Transition to Watch Mode

CPU operation makes a transition to watch mode when the SLEEP instruction is executed in high-speed mode or subactive mode with SBYCR SSBY=1, LPWRCR DTON = 0, and TCSR_1 PSS = 1.

In watch mode, the CPU is stopped and peripheral modules other than RTC are also stopped. The contents of the CPU's internal registers, the data in internal RAM, and the statuses of the internal peripheral modules (excluding the A/D converter) and I/O ports are retained. To make a transition to watch mode, bits SCK2 to SCK0 in SCKCR must be set to 0.

### 20.7.2 Exiting Watch Mode

Watch mode is exited by any interrupt (WOVI interrupt, NMI pin, or $\overline{\text{IRQ0}}$, to $\overline{\text{IRQ7}}$), or signals at the $\overline{\text{RES}}$, $\overline{\text{MRES}}$*, or $\overline{\text{STBY}}$ pin.

- Exiting Watch Mode by Interrupts

  When an interrupt occurs, watch mode is exited and a transition is made to high-speed mode or medium-speed mode when the LPWRCR LSON bit = 0 or to subactive mode when the LSON bit = 1. When a transition is made to high-speed mode, a stable clock is supplied to all LSI circuits and interrupt exception processing starts after the time set in SBYCR STS2 to STS0 has elapsed. In case of IRQ0, to IRQ7 interrupts, no transition is made from watch mode if the corresponding enable bit/pin function switching bit has been cleared to 0, and, in the case of interrupts from the internal peripheral modules, the interrupt enable register has been set to disable the reception of that interrupt, or is masked by the CPU.

  See section 20.4.3, Setting Oscillation Stablization Time after Clearing Software Standby Mode, for how to set the oscillation settling time when making a transition from watch mode to high-speed mode.
- Exiting Watch Mode by $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin

  For exiting watch mode by the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin, see section 20.4.2, Clearing Software Standby Mode.
- Exiting Watch Mode by $\overline{\text{STBY}}$ pin

  When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.

Note: * Supported only by the H8S/2218 Group.

## 20.8 Subsleep Mode

### 20.8.1 Transition to Sleep Mode

When the SLEEP instruction is executed with the SBYCR SSBY bit = 0, LPWRCR LSON bit = 1, and TCSR_1 PSS bit = 1, CPU operation shifts to subsleep mode.

In subsleep mode, the CPU is stopped. Peripheral modules other WDT and RTC are also stopped. The contents of the CPU's internal registers, the data in internal RAM, and the statuses of the internal peripheral modules (excluding the A/D converter) and I/O ports are retained.

### 20.8.2 Exiting Subsleep Mode

Subsleep mode is exited by an interrupt (interrupts from internal peripheral modules, NMI pin, or $\overline{\text{IRQ0}}$, to $\overline{\text{IRQ7}}$), or signals at the $\overline{\text{RES}}$, $\overline{\text{MRES}}$*, or $\overline{\text{STBY}}$ pin.

- Exiting Subsleep Mode by Interrupts
  When an interrupt occurs, subsleep mode is exited and interrupt exception processing starts.
  In case of $\overline{\text{IRQ0}}$, to $\overline{\text{IRQ7}}$interrupts, subsleep mode is not cancelled if the corresponding enable bit/pin function switching bit has been cleared to 0, and, in the case of interrupts from the internal peripheral modules, the interrupt enable register has been set to disable the reception of that interrupt, or is masked by the CPU.
- Exiting Subsleep Mode by $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin
  For exiting subsleep mode by the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin, see section 20.4.2, Clearing Software Standby Mode.
- Exiting Subsleep Mode by $\overline{\text{STBY}}$ Pin
  When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.

Note: * Supported only by the H8S/2218 Group.

## 20.9 Subactive Mode

### 20.9.1 Transition to Subactive Mode

When the SLEEP instruction is executed in high-speed mode with the SBYCR SSBY bit = 1, LPWRCR DTON bit = 1, LSON bit = 1, and TCSR_1 PSS bit = 1, CPU operation shifts to subactive mode. When an interrupt occurs in watch mode, and if the LSON bit of LPWRCR is 1, a transition is made to subactive mode. And if an interrupt occurs in subsleep mode, a transition is made to subactive mode.

In subactive mode, the CPU operates at low speed on the subclock, and the program is executed step by step. Peripheral modules other than WDT and RTC are also stopped.

When operating the CPU in subactive mode, the SCKCR SCK2 to SCK0 bits must be set to 0.

### 20.9.2 Exiting Subactive Mode

Subactive mode is exited by the SLEEP instruction or the $\overline{\text{RES}}$, $\overline{\text{MRES}}$*, or $\overline{\text{STBY}}$ pin.

- Exiting Subactive Mode by SLEEP Instruction
  When the SLEEP instruction is executed with the SBYCR SSBY bit = 1, LPWRCR DTON bit = 0, and TCSR_1 PSS bit = 1, the CPU exits subactive mode and a transition is made to watch mode. When the SLEEP instruction is executed with the SBYCR SSBY bit = 0, LPWRCR LSON bit = 1, and TCSR_1 PSS bit = 1, a transition is made to subsleep mode. Finally, when the SLEEP instruction is executed with the SBYCR SSBY bit = 1, LPWRCR DTON bit = 1, LSON bit = 0, and TCSR_1 PSS bit = 1, a direct transition is made to high-speed mode (SCK0 to SCK2 all 0).
- Exiting Subactive Mode by $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin
  For exiting subactive mode by the $\overline{\text{RES}}$ or $\overline{\text{MRES}}$* pin, see section 20.4.2, Clearing Software Standby Mode.
- Exiting Subactive Mode by $\overline{\text{STBY}}$ Pin
  When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.

Note: * Supported only by the H8S/2218 Group.

## 20.10 Direct Transitions

There are three modes, high-speed, medium-speed, and subactive, in which the CPU executes programs. When a direct transition is made, there is no interruption of program execution when shifting between high-speed and subactive modes. Direct transitions are enabled by setting the LPWRCR DTON bit to 1, then executing the SLEEP instruction. After a transition, direct transition interrupt exception processing starts.

### 20.10.1 Direct Transitions from High-Speed Mode to Subactive Mode

Execute the SLEEP instruction in high-speed mode when the SBYCR SSBY bit = 1, LPWRCR LSON bit = 1, and DTON bit = 1, and TSCR_1 PSS bit = 1 to make a transition to subactive mode.

### 20.10.2 Direct Transitions from Subactive Mode to High-Speed Mode

Execute the SLEEP instruction in subactive mode when the SBYCR SSBY bit = 1, LPWRCR LSON bit = 0, and DTON bit = 1, and TSCR_1 PSS bit = 1 to make a direct transition to high-speed mode after the time set in SBYCR STS2 to STS0 has elapsed.

## 20.11 φ Clock Output Disabling Function

Output of the φ clock can be controlled by means of the PSTOP bit in SCKCR, and DDR for the corresponding port. When the PSTOP bit is set to 1, the φ clock stops at the end of the bus cycle, and φ output goes high. φ clock output is enabled when the PSTOP bit is cleared to 0. When DDR for the corresponding port is cleared to 0, φ clock output is disabled and input port mode is set. Table 20.4 shows the state of the φ pin in each processing state.

**Table 20.4 φ Pin State in Each Processing State**

| Register Settings | | High-Speed Mode, Medium-Speed Mode, Subactive Mode | Sleep Mode, Subsleep Mode | Software Standby Mode, Watch Mode, Direct Transition | Hardware Standby Mode |
|---|---|---|---|---|---|
| DDR | PSTOP | | | | |
| 0 | × | High impedance | High impedance | High impedance | High impedance |
| 1 | 0 | φ output | φ output | Fixed high | High impedance |
| 1 | 1 | Fixed high | Fixed high | Fixed high | High impedance |

Legend:
×: Don't care

## 20.12 Usage Notes

### 20.12.1 I/O Port Status

In software standby mode or watch mode, I/O port states are retained. In addition, if the OPE bit is set to 1, the address bus and bus control signal output are retained. Therefore, there is no reduction in current dissipation for the output current when a high-level signal is output.

### 20.12.2 Current Dissipation during Oscillation Stabilization Wait Period

Current dissipation increases during the oscillation stabilization wait period.

### 20.12.3 Flash Memory Module Stop

Setting of the flash memory module stop mode should be carried out while the programs in the on-chip RAM and external memory are executed. For details, see section 20.1.3, Module Stop Control Registers A to C (MSTPCRA to MSTPCRC).

### 20.12.4 DMAC Module Stop

Depending on the operating status of the DMAC, the MSTPA7 bit may not be set to 1. Setting of the DMAC module stop mode should be carried out only when the DMAC is not activated.

For details, section 7, DMA Controller (DMAC).

### 20.12.5 On-Chip Peripheral Module Interrupt

- Module stop mode

  Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or DMAC activation source. Interrupts should therefore be disabled before setting module stop mode.
- Subactive Mode/Watch Mode

  On-chip peripheral modules (DMAC and TPU) that stop operation in subactive mode cannot clear interrupts in subactive mode. Therefore, if subactive mode is entered when an interrupt is requested, CPU interrupt factors cannot be cleared.

  Interrupts should therefore before executing the SLEEP instruction and entering subactive or watch mode.

### 20.12.6 Entering Subactive/Watch Mode and DMAC and DTC Module Stop

To enter subactive or watch mode, set DMAC to module stop (write 1 to the MSTPA7 bit) and reading the MSTPA7 bit as 1 before transiting mode. After transiting from subactive mode to active mode, clear module stop.

When a DMAC activation source is generated in subactive mode, the DMAC is activated when module stop is cleared following the transition to active mode.

### 20.12.7 Writing to MSTPCR

MSTPCR should only be written to by the CPU.

# Section 21 List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register Addresses (address order)
- Registers are listed from the lower allocation addresses.
- Registers are classified by functional modules.
- The access size is indicated.

2. Register Bits
- Bit configurations of the registers are described in the same order as the Register Addresses (address order) above.
- Reserved bits are indicated by "—" in the bit name column.
- The bit number in the bit-name column indicates that the whole register is allocated as a counter or for holding data.
- 16-bit or 24-bit registers are indicated from the bit on the MSB side.

3. Register States in Each Operating Mode
- Register states are described in the same order as the Register Addresses (address order) above.
- The register states described here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

## 21.1 Register Addresses (Address Order)

The data bus width indicates the numbers of bits by which the register is accessed.

The number of access states indicates the number of states based on the specified reference clock.

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| USB reserved area | — | — | H'C00000 to H'C0007F | — | — | USB |
| USB control register | UCTLR | 8 | H'C00080 | 8 | 3 | |
| USB test register A | UTSTRA | 8 | H'C00081 | 8 | 3 | |
| USB DMAC transfer request register | UDMAR | 8 | H'C00082 | 8 | 3 | |
| USB device resume register | UDRR | 8 | H'C00083 | 8 | 3 | |
| USB trigger register 0 | UTRG0 | 8 | H'C00084 | 8 | 3 | |
| USB FIFO clear register 0 | UFCLR0 | 8 | H'C00086 | 8 | 3 | |
| USB endpoint stall register 0 | UESTL0 | 8 | H'C00088 | 8 | 3 | |
| USB endpoint stall register 1 | UESTL1 | 8 | H'C00089 | 8 | 3 | |
| USB endpoint data register 0s | UEDR0s | 8 | H'C00090 to H'C00093 | 8 | 3 | |
| USB endpoint data register 0i | UEDR0i | 8 | H'C00094 to H'C00097 | 8 | 3 | |
| USB endpoint data register 0o | UEDR0o | 8 | H'C00098 to H'C0009B | 8 | 3 | |
| USB endpoint data register 3 | UEDR3 | 8 | H'C0009C to H'C0009F | 8 | 3 | |
| USB endpoint data register 1 | UEDR1 | 8 | H'C000A0 to H'C000A3 | 8 | 3 | |
| USB endpoint data register 2 | UEDR2 | 8 | H'C000A4 to H'C000A7 | 8 | 3 | |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| USB endpoint receive data size register 0o | UESZ0o | 8 | H'C000BC | 8 | 3 | USB |
| USB endpoint receive data size register 2 | UESZ2 | 8 | H'C000BD | 8 | 3 | |
| USB interrupt flag register 0 | UIFR0 | 8 | H'C000C0 | 8 | 3 | |
| USB interrupt flag register 1 | UIFR1 | 8 | H'C000C1 | 8 | 3 | |
| USB interrupt flag register 3 | UIFR3 | 8 | H'C000C3 | 8 | 3 | |
| USB interrupt enable register 0 | UIER0 | 8 | H'C000C4 | 8 | 3 | |
| USB interrupt enable register 1 | UIER1 | 8 | H'C000C5 | 8 | 3 | |
| USB interrupt enable register 3 | UIER3 | 8 | H'C000C7 | 8 | 3 | |
| USB interrupt selection register 0 | UISR0 | 8 | H'C000C8 | 8 | 3 | |
| USB interrupt selection register 1 | UISR1 | 8 | H'C000C9 | 8 | 3 | |
| USB interrupt selection register 3 | UISR3 | 8 | H'C000CB | 8 | 3 | |
| USB data status register | UDSR | 8 | H'C000CC | 8 | 3 | |
| USB configuration value register | UCVR | 8 | H'C000CF | 8 | 3 | |
| USB test register 0 | UTSTR0 | 8 | H'C000F0 | 8 | 3 | |
| USB test register 1 | UTSTR1 | 8 | H'C000F1 | 8 | 3 | |
| USB test register 2 | UTSTR2 | 8 | H'C000F2 | 8 | 3 | |
| USB test register B | UTSTRB | 8 | H'C000FB | 8 | 3 | |
| USB test register C | UTSTRC | 8 | H'C000FC | 8 | 3 | |
| USB test register D | UTSTRD | 8 | H'C000FD | 8 | 3 | |
| USB test register E | UTSTRE | 8 | H'C000FE | 8 | 3 | |
| USB test register F | UTSTRF | 8 | H'C000FF | 8 | 3 | |
| USB reserved area | — | — | H'C00100 to H'DFFFFF | — | — | |
| Serial control register X | SCRX | 8 | H'FDB4 | 8 | 2 | FLASH |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| Standby control register | SBYCR | 8 | H'FDE4 | 8 | 2 | SYSTEM |
| System control register | SYSCR | 8 | H'FDE5 | 8 | 2 | |
| System clock control register | SCKCR | 8 | H'FDE6 | 8 | 2 | |
| Mode control register | MDCR | 8 | H'FDE7 | 8 | 2 | |
| Module stop control register A | MSTPCRA | 8 | H'FDE8 | 8 | 2 | |
| Module stop control register B | MSTPCRB | 8 | H'FDE9 | 8 | 2 | |
| Module stop control register C | MSTPCRC | 8 | H'FDEA | 8 | 2 | |
| Pin function control register | PFCR | 8 | H'FDEB | 8 | 2 | BSC |
| Low power control register | LPWRCR | 8 | H'FDEC | 8 | 2 | SYSTEM |
| Clock output control register | OUTCR | 8 | H'FDEF | 8 | 2 | PORT |
| Serial extended mode register A_0 | SEMRA_0 | 8 | H'FDF8 | 8 | 2 | SCI_0 |
| Serial extended mode register B_0 | SEMRB_0 | 8 | H'FDF9 | 8 | 2 | |
| IRQ sense control register H | ISCRH | 8 | H'FE12 | 8 | 2 | INT |
| IRQ sense control register L | ISCRL | 8 | H'FE13 | 8 | 2 | |
| IRQ enable register | IER | 8 | H'FE14 | 8 | 2 | |
| IRQ status register | ISR | 8 | H'FE15 | 8 | 2 | |
| Port 1 data direction register | P1DDR | 8 | H'FE30 | 8 | 2 | PORT |
| Port 3 data direction register | P3DDR | 8 | H'FE32 | 8 | 2 | |
| Port 7 data direction register | P7DDR | 8 | H'FE36 | 8 | 2 | |
| Port A data direction register | PADDR | 8 | H'FE39 | 8 | 2 | |
| Port B data direction register | PBDDR | 8 | H'FE3A | 8 | 2 | |
| Port C data direction register | PCDDR | 8 | H'FE3B | 8 | 2 | |
| Port D data direction register | PDDDR | 8 | H'FE3C | 8 | 2 | |
| Port E data direction register | PEDDR | 8 | H'FE3D | 8 | 2 | |
| Port F data direction register | PFDDR | 8 | H'FE3E | 8 | 2 | |
| Port G data direction register | PGDDR | 8 | H'FE3F | 8 | 2 | |
| Port A pull-up MOS control register | PAPCR | 8 | H'FE40 | 8 | 2 | |
| Port B pull-up MOS control register | PBPCR | 8 | H'FE41 | 8 | 2 | |
| Port C pull-up MOS control register | PCPCR | 8 | H'FE42 | 8 | 2 | |
| Port D pull-up MOS control register | PDPCR | 8 | H'FE43 | 8 | 2 | |
| Port E pull-up MOS control register | PEPCR | 8 | H'FE44 | 8 | 2 | |
| Port 3 open drain control register | P3ODR | 8 | H'FE46 | 8 | 2 | |
| Port A open drain control register | PAODR | 8 | H'FE47 | 8 | 2 | |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| Timer start register | TSTR | 8 | H'FEB0 | 16 | 2 | TPU |
| Timer synchro register | TSYR | 8 | H'FEB1 | 16 | 2 | |
| Interrupt priority register A | IPRA | 8 | H'FEC0 | 8 | 2 | INT |
| Interrupt priority register B | IPRB | 8 | H'FEC1 | 8 | 2 | |
| Interrupt priority register C | IPRC | 8 | H'FEC2 | 8 | 2 | |
| Interrupt priority register D | IPRD | 8 | H'FEC3 | 8 | 2 | |
| Interrupt priority register E | IPRE | 8 | H'FEC4 | 8 | 2 | |
| Interrupt priority register F | IPRF | 8 | H'FEC5 | 8 | 2 | |
| Interrupt priority register G | IPRG | 8 | H'FEC6 | 8 | 2 | |
| Interrupt priority register J | IPRJ | 8 | H'FEC9 | 8 | 2 | |
| Interrupt priority register K | IPRK | 8 | H'FECA | 8 | 2 | |
| Interrupt priority register M | IPRM | 8 | H'FECC | 8 | 2 | |
| Bus width control register | ABWCR | 8 | H'FED0 | 8 | 2 | BSC |
| Access state control register | ASTCR | 8 | H'FED1 | 8 | 2 | |
| Wait control register H | WCRH | 8 | H'FED2 | 8 | 2 | |
| Wait control register L | WCRL | 8 | H'FED3 | 8 | 2 | |
| Bus control register H | BCRH | 8 | H'FED4 | 8 | 2 | |
| Bus control register L | BCRL | 8 | H'FED5 | 8 | 2 | |
| RAM emulation register | RAMER | 8 | H'FEDB | 8 | 2 | FLASH |
| Memory address register 0A H | MAR0AH | 16 | H'FEE0 | 16 | 2 | DMAC |
| Memory address register 0A L | MAR0AL | 16 | H'FEE2 | 16 | 2 | |
| I/O address register 0A | IOAR0A | 16 | H'FEE4 | 16 | 2 | |
| Transfer count register 0A | ETCR0A | 16 | H'FEE6 | 16 | 2 | |
| Memory address register 0B H | MAR0BH | 16 | H'FEE8 | 16 | 2 | |
| Memory address register 0B L | MAR0BL | 16 | H'FEEA | 16 | 2 | |
| I/O address register 0B | IOAR0B | 16 | H'FEEC | 16 | 2 | |
| Transfer count register 0B | ETCR0B | 16 | H'FEEE | 16 | 2 | |
| Memory address register 1A H | MAR1AH | 16 | H'FEF0 | 16 | 2 | |
| Memory address register 1A L | MAR1AL | 16 | H'FEF2 | 16 | 2 | |
| I/O address register 1A | IOAR1A | 16 | H'FEF4 | 16 | 2 | |
| Transfer count register 1A | ETCR1A | 16 | H'FEF6 | 16 | 2 | |
| Memory address register 1B H | MAR1BH | 16 | H'FEF8 | 16 | 2 | |
| Memory address register 1B L | MAR1BL | 16 | H'FEFA | 16 | 2 | |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| I/O address register 1B | IOAR1B | 16 | H'FEFC | 16 | 2 | DMAC |
| Transfer count register 1B | ETCR1B | 16 | H'FEFE | 16 | 2 | |
| Port 1 data register | P1DR | 8 | H'FF00 | 8 | 2 | PORT |
| Port 3 data register | P3DR | 8 | H'FF02 | 8 | 2 | |
| Port 7 data register | P7DR | 8 | H'FF06 | 8 | 2 | |
| Port A data register | PADR | 8 | H'FF09 | 8 | 2 | |
| Port B data register | PBDR | 8 | H'FF0A | 8 | 2 | |
| Port C data register | PCDR | 8 | H'FF0B | 8 | 2 | |
| Port D data register | PDDR | 8 | H'FF0C | 8 | 2 | |
| Port E data register | PEDR | 8 | H'FF0D | 8 | 2 | |
| Port F data register | PFDR | 8 | H'FF0E | 8 | 2 | |
| Port G data register | PGDR | 8 | H'FF0F | 8 | 2 | |
| Timer control register_0 | TCR_0 | 8 | H'FF10 | 16 | 2 | TPU_0 |
| Timer mode register_0 | TMDR_0 | 8 | H'FF11 | 16 | 2 | |
| Timer I/O control register H_0 | TIORH_0 | 8 | H'FF12 | 16 | 2 | |
| Timer I/O control register L_0 | TIORL_0 | 8 | H'FF13 | 16 | 2 | |
| Timer interrupt enable register_0 | TIER_0 | 8 | H'FF14 | 16 | 2 | |
| Timer status register_0 | TSR_0 | 8 | H'FF15 | 16 | 2 | |
| Timer counter_0 | TCNT_0 | 16 | H'FF16 | 16 | 2 | |
| Timer general register A_0 | TGRA_0 | 16 | H'FF18 | 16 | 2 | |
| Timer general register B_0 | TGRB_0 | 16 | H'FF1A | 16 | 2 | |
| Timer general register C_0 | TGRC_0 | 16 | H'FF1C | 16 | 2 | |
| Timer general register D_0 | TGRD_0 | 16 | H'FF1E | 16 | 2 | |
| Timer control register_1 | TCR_1 | 8 | H'FF20 | 16 | 2 | TPU_1 |
| Timer mode register_1 | TMDR_1 | 8 | H'FF21 | 16 | 2 | |
| Timer I/O control register _1 | TIOR_1 | 8 | H'FF22 | 16 | 2 | |
| Timer interrupt enable register _1 | TIER_1 | 8 | H'FF24 | 16 | 2 | |
| Timer status register_1 | TSR_1 | 8 | H'FF25 | 16 | 2 | |
| Timer counter_1 | TCNT_1 | 16 | H'FF26 | 16 | 2 | |
| Timer general register A_1 | TGRA_1 | 16 | H'FF28 | 16 | 2 | |
| Timer general register B_1 | TGRB_1 | 16 | H'FF2A | 16 | 2 | |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| Timer control register_2 | TCR_2 | 8 | H'FF30 | 16 | 2 | TPU_2 |
| Timer mode register_2 | TMDR_2 | 8 | H'FF31 | 16 | 2 | |
| Timer I/O control register 2 | TIOR_2 | 8 | H'FF32 | 16 | 2 | |
| Timer interrupt enable register 2 | TIER_2 | 8 | H'FF34 | 16 | 2 | |
| Timer status register_2 | TSR_2 | 8 | H'FF35 | 16 | 2 | |
| Timer counter_2 | TCNT_2 | 16 | H'FF36 | 16 | 2 | |
| Timer general register A_2 | TGRA_2 | 16 | H'FF38 | 16 | 2 | |
| Timer general register B_2 | TGRB_2 | 16 | H'FF3A | 16 | 2 | |
| Extended module stop register | EXMDLSTP | 8 | H'FF40 | 8 | 2 | SYSTEM |
| Second data register/ free running counter data register | RSECDR | 8 | H'FF48 | 8 | 2 | RTC |
| Minute data register | RMINDR | 8 | H'FF49 | 8 | 2 | |
| Hour data register | RHRDR | 8 | H'FF4A | 8 | 2 | |
| Day-of-week data register | RWKDR | 8 | H'FF4B | 8 | 2 | |
| RTC control register 1 | RTCCR1 | 8 | H'FF4C | 8 | 2 | |
| RTC control register 2 | RTCCR2 | 8 | H'FF4D | 8 | 2 | |
| Clock source select register | RTCCSR | 8 | H'FF4F | 8 | 2 | |
| DMA control register 0A | DMACR0A | 8 | H'FF62 | 16 | 2 | DMAC |
| DMA control register 0B | DMACR0B | 8 | H'FF63 | 16 | 2 | |
| DMA control register 1A | DMACR1A | 8 | H'FF64 | 16 | 2 | |
| DMA control register 1B | DMACR1B | 8 | H'FF65 | 16 | 2 | |
| DMA band control register | DMABCR | 16 | H'FF66 | 16 | 2 | |
| Timer control/status register | TCSR | 8 | H'FF74 | 16 | 2 | WDT |
| Timer counter | TCNT | 8 | H'FF74 (write) | 16 | 2 | |
| Timer counter | TCNT | 8 | H'FF75 (read) | 16 | 2 | |
| Reset control/status register | RSTCSR | 8 | H'FF76 (write) | 16 | 2 | |
| Reset control/status register | RSTCSR | 8 | H'FF77 (read) | 16 | 2 | |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| Serial mode register_0 | SMR_0 | 8 | H'FF78 | 8 | 2 | SCI_0 |
| Bit rate register_0 | BRR_0 | 8 | H'FF79 | 8 | 2 | |
| Serial control register_0 | SCR_0 | 8 | H'FF7A | 8 | 2 | |
| Transmit data register_0 | TDR_0 | 8 | H'FF7B | 8 | 2 | |
| Serial status register_0 | SSR_0 | 8 | H'FF7C | 8 | 2 | |
| Receive data register_0 | RDR_0 | 8 | H'FF7D | 8 | 2 | |
| Smart card mode register_0 | SCMR_0 | 8 | H'FF7E | 8 | 2 | |
| Serial mode register_2 | SMR_2 | 8 | H'FF88 | 8 | 2 | SCI_2 |
| Bit rate register_2 | BRR_2 | 8 | H'FF89 | 8 | 2 | |
| Serial control register_2 | SCR_2 | 8 | H'FF8A | 8 | 2 | |
| Transmit data register_2 | TDR_2 | 8 | H'FF8B | 8 | 2 | |
| Serial status register_2 | SSR_2 | 8 | H'FF8C | 8 | 2 | |
| Receive data register_2 | RDR_2 | 8 | H'FF8D | 8 | 2 | |
| Smart card mode register_2 | SCMR_2 | 8 | H'FF8E | 8 | 2 | |
| A/D data register AH | ADDRAH | 8 | H'FF90 | 8 | 2 | A/D |
| A/D data register AL | ADDRAL | 8 | H'FF91 | 8 | 2 | |
| A/D data register BH | ADDRBH | 8 | H'FF92 | 8 | 2 | |
| A/D data register BL | ADDRBL | 8 | H'FF93 | 8 | 2 | |
| A/D data register CH | ADDRCH | 8 | H'FF94 | 8 | 2 | |
| A/D data register CL | ADDRCL | 8 | H'FF95 | 8 | 2 | |
| A/D data register DH | ADDRDH | 8 | H'FF96 | 8 | 2 | |
| A/D data register DL | ADDRDL | 8 | H'FF97 | 8 | 2 | |
| A/D control/status register | ADCSR | 8 | H'FF98 | 8 | 2 | |
| A/D control register | ADCR | 8 | H'FF99 | 8 | 2 | |
| Timer control/status register | TCSR_1 | 8 | H'FFA2 | 16 | 2 | SYSTEM |
| Flash memory control register 1 | FLMCR1 | 8 | H'FFA8 | 8 | 2 | FLASH |
| Flash memory control register 2 | FLMCR2 | 8 | H'FFA9 | 8 | 2 | |
| Erase block register 1 | EBR1 | 8 | H'FFAA | 8 | 2 | |
| Erase block register 2 | EBR2 | 8 | H'FFAB | 8 | 2 | |

| Register Name | Abbreviation | Number of Bits | Address | Data Bus Width | Number of Access States | Module |
|---|---|---|---|---|---|---|
| Port 1 register | PORT1 | 8 | H'FFB0 | 8 | 2 | PORT |
| Port 3 register | PORT3 | 8 | H'FFB2 | 8 | 2 | |
| Port 4 register | PORT4 | 8 | H'FFB3 | 8 | 2 | |
| Port 7 register | PORT7 | 8 | H'FFB6 | 8 | 2 | |
| Port 9 register | PORT9 | 8 | H'FFB8 | 8 | 2 | |
| Port A register | PORTA | 8 | H'FFB9 | 8 | 2 | |
| Port B register | PORTB | 8 | H'FFBA | 8 | 2 | |
| Port C register | PORTC | 8 | H'FFBB | 8 | 2 | |
| Port D register | PORTD | 8 | H'FFBC | 8 | 2 | |
| Port E register | PORTE | 8 | H'FFBD | 8 | 2 | |
| Port F register | PORTF | 8 | H'FFBE | 8 | 2 | |
| Port G register | PORTG | 8 | H'FFBF | 8 | 2 | |

## 21.2 Register Bits

Register bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, so 16-bit registers are shown as two lines and 32-bit registers as four lines.

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| UCTLR | — | USPNDE | UCKS3 | UCKS2 | UCKS1 | UCKS0 | UIFRST | UDCRST | USB |
| UTSTRA | — | — | — | — | — | — | — | — | |
| UDMAR | — | — | — | — | EP2T1 | EP2T0 | EP1T1 | EP1T0 | |
| UDRR | — | — | — | — | — | — | RWUPs | DVR | |
| UTRG0 | — | — | EP2RDFN | EP1PKTE | EP3PKTE | EP0oRDFN | EP0iPKTE | EP0sRDFN | |
| UFCLR0 | — | — | EP2CLR | EP1CLR | EP3CLR | EP0oCLR | EP0iCLR | — | |
| UESTL0 | — | — | EP2STL | EP1STL | EP3STL | — | — | EP0STL | |
| UESTL1 | SCME | — | — | — | — | — | — | — | |
| UEDR0s | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR0i | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR0o | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR3 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR1 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR2 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UESZ0o | — | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UESZ2 | — | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UIFR0 | BRST | — | EP3TR | EP3TS | EP0oTS | EP0iTR | EP0iTS | SetupTS | |
| UIFR1 | — | — | — | — | EP1ALL EMPTYs | EP2 READY | EP1TR | EP1 EMPTY | |
| UIFR3 | CK48 READY | SOF | SETC | — | SPRSs | SPRSi | VBUSs | VBUSi | |
| UIER0 | BRSTE | — | EP3TRE | EP3TSE | EP0oTSE | EP0iTRE | EP0iTSE | SetupTSE | |
| UIER1 | — | — | — | — | — | EP2 READYE | EP1TRE | EP1 EMPTYE | |
| UIER3 | CK48 READYE | SOFE | SETCE | — | — | SPRSiE | — | VBUSiE | |
| UISR0 | BRSTS | — | EP3TRS | EP3TSS | EP0oTSS | EP0iTRS | EP0iTSS | SetupTSS | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| UISR1 | — | — | — | — | — | EP2 READYS | EP1TRS | EP1 EMPTYS | USB |
| UISR3 | CK48 READYS | SOFS | SETCS | — | — | — | — | VBUSiS | |
| UDSR | — | — | — | — | — | EP1DE | EP3DE | EP0iDE | |
| UCVR | — | — | CNFV0 | — | — | — | — | — | |
| UTSTR0 | PTSTE | — | — | — | SUSPEND | $\overline{OE}$ | FSE0 | VPO | |
| UTSTR1 | VBUS | $\overline{UBPM}$ | — | — | — | RCV | VP | VM | |
| UTSTR2 | — | — | — | — | — | — | — | — | |
| UTSTRB | — | — | — | — | — | — | — | — | |
| UTSTRC | — | — | — | — | — | — | — | — | |
| UTSTRD | — | — | — | — | — | — | — | — | |
| UTSTRE | — | — | — | — | — | — | — | — | |
| UTSTRF | — | — | — | — | — | — | — | — | |
| SCRX | — | — | — | — | FLSHE | — | — | — | FLASH |
| SBYCR | SSBY | STS2 | STS1 | STS0 | OPE | — | — | — | SYSTEM |
| SYSCR | — | — | INTM1 | INTM0 | NMIEG | MRESE | — | RAME | |
| SCKCR | PSTOP | — | — | — | — | SCK2 | SCK1 | SCK0 | |
| MDCR | — | — | — | — | FWE | MDS2 | MDS1 | MDS0 | |
| MSTPCRA | MSTPA7 | MSTPA6 | MSTPA5 | MSTPA4 | MSTPA3 | MSTPA2 | MSTPA1 | MSTPA0 | |
| MSTPCRB | MSTPB7 | MSTPB6 | MSTPB5 | MSTPB4 | MSTPB3 | MSTPB2 | MSTPB1 | MSTPB0 | |
| MSTPCRC | MSTPC7 | MSTPC6 | MSTPC5 | MSTPC4 | MSTPC3 | MSTPC2 | MSTPC1 | MSTPC0 | |
| PFCR | — | — | — | — | AE3 | AE2 | AE1 | AE0 | BSC |
| LPWRCR | DTON | LSON | NESEL | SUBSTP | RFCUT | — | STC1 | STC0 | SYSTEM |
| OUTCR | — | — | — | — | — | PF7OUT2 | PF7OUT1 | PF7OUT0 | PORT |
| SEMRA_0 | SSE | TCS2 | TCS1 | TCS0 | ABCS | ACS2 | ACS1 | ACS0 | SCI_0 |
| SEMRB_0 | ACS3 | — | — | — | TIOCA2E | TIOCA1E | TIOCC0E | TIOCA0E | |
| ISCRH | IRQ7SCB | IRQ7SCA | IRQ6SCB | IRQ6SCA | IRQ5SCB | IRQ5SCA | IRQ4SCB | IRQ4SCA | INT |
| ISCRL | IRQ3SCB | IRQ3SCA | IRQ2SCB | IRQ2SCA | IRQ1SCB | IRQ1SCA | IRQ0SCB | IRQ0SCA | |
| IER | IRQ7E | IRQ6E | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E | |
| ISR | IRQ7F | IRQ6F | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| P1DDR | P17DDR | P16DDR | P15DDR | P14DDR | P13DDR | P12DDR | P11DDR | P10DDR | PORT |
| P3DDR | — | P36DDR | — | — | — | P32DDR | P31DDR | P30DDR | |
| P7DDR | P77DDR | P76DDR | P75DDR | P74DDR | — | — | P71DDR | P70DDR | |
| PADDR | — | — | — | — | PA3DDR | PA2DDR | PA1DDR | PA0DDR | |
| PBDDR | PB7DDR | PB6DDR | PB5DDR | PB4DDR | PB3DDR | PB2DDR | PB1DDR | PB0DDR | |
| PCDDR | PC7DDR | PC6DDR | PC5DDR | PC4DDR | PC3DDR | PC2DDR | PC1DDR | PC0DDR | |
| PDDDR | PD7DDR | PD6DDR | PD5DDR | PD4DDR | PD3DDR | PD2DDR | PD1DDR | PD0DDR | |
| PEDDR | PE7DDR | PE6DDR | PE5DDR | PE4DDR | PE3DDR | PE2DDR | PE1DDR | PE0DDR | |
| PFDDR | PF7DDR | PF6DDR | PF5DDR | PF4DDR | PF3DDR | PF2DDR | PF1DDR | PF0DDR | |
| PGDDR | — | — | — | PG4DDR | PG3DDR | PG2DDR | PG1DDR | PG0DDR | |
| PAPCR | — | — | — | — | PA3PCR | PA2PCR | PA1PCR | PA0PCR | |
| PBPCR | PB7PCR | PB6PCR | PB5PCR | PB4PCR | PB3PCR | PB2PCR | PB1PCR | PB0PCR | |
| PCPCR | PC7PCR | PC6PCR | PC5PCR | PC4PCR | PC3PCR | PC2PCR | PC1PCR | PC0PCR | |
| PDPCR | PD7PCR | PD6PCR | PD5PCR | PD4PCR | PD3PCR | PD2PCR | PD1PCR | PD0PCR | |
| PEPCR | PE7PCR | PE6PCR | PE5PCR | PE4PCR | PE3PCR | PE2PCR | PE1PCR | PE0PCR | |
| P3ODR | — | P36ODR | — | — | — | P32ODR | P31ODR | P30ODR | |
| PAODR | — | — | — | — | PA3ODR | PA2ODR | PA1ODR | PA0ODR | |
| TSTR | — | — | — | — | — | CST2 | CST1 | CST0 | TPU |
| TSYR | — | — | — | — | — | SYNC2 | SYNC1 | SYNC0 | |
| IPRA | — | IPRA6 | IPRA5 | IPRA4 | — | IPRA2 | IPRA1 | IPRA0 | INT |
| IPRB | — | IPRB6 | IPRB5 | IPRB4 | — | IPRB2 | IPRB1 | IPRB0 | |
| IPRC | — | IPRC6 | IPRC5 | IPRC4 | — | — | — | — | |
| IPRD | — | IPRD6 | IPRD5 | IPRD4 | — | — | — | — | |
| IPRE | — | — | — | — | — | IPRE2 | IPRE1 | IPRE0 | |
| IPRF | — | IPRF6 | IPRF5 | IPRF4 | — | IPRF2 | IPRF1 | IPRF0 | |
| IPRG | — | IPRG6 | IPRG5 | IPRG4 | — | — | — | — | |
| IPRJ | — | IPRJ6 | IPRJ5 | IPRJ4 | — | IPRJ2 | IPRJ1 | IPRJ0 | |
| IPRK | — | — | — | — | — | IPRK2 | IPRK1 | IPRK0 | |
| IPRM | — | IPRM6 | IPRM5 | IPRM4 | — | — | — | — | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| ABWCR | ABW7 | ABW6 | ABW5 | ABW4 | ABW3 | ABW2 | ABW1 | ABW0 | BSC |
| ASTCR | AST7 | AST6 | AST5 | AST4 | AST3 | AST2 | AST1 | AST0 | |
| WCRH | W71 | W70 | W61 | W60 | W51 | W50 | W41 | W40 | |
| WCRL | W31 | W30 | W21 | W20 | W11 | W10 | W01 | W00 | |
| BCRH | ICIS1 | ICIS0 | BRSTRM | BRSTS1 | BRSTS0 | RMTS2 | RMTS1 | RMTS0 | |
| BCRL | BRLE | — | — | — | — | — | — | WAITE | |
| RAMER | — | — | — | — | RAMS | — | RAM1 | RAM0 | FLASH |
| MAR0A | — | — | — | — | — | — | — | — | DMAC |
| | Bit 23 | Bit 22 | Bit 21 | Bit 20 | Bit 19 | Bit 18 | Bit 17 | Bit 16 | |
| | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| IOAR0A | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| ETCR0A | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| MAR0B | — | — | — | — | — | — | — | — | |
| | Bit 23 | Bit 22 | Bit 21 | Bit 20 | Bit 19 | Bit 18 | Bit 17 | Bit 16 | |
| | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| IOAR0B | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| ETCR0B | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| MAR1A | — | — | — | — | — | — | — | — | |
| | Bit 23 | Bit 22 | Bit 21 | Bit 20 | Bit 19 | Bit 18 | Bit 17 | Bit 16 | |
| | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| IOAR1A | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| ETCR1A | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| MAR1B | — | — | — | — | — | — | — | — | DMAC |
| | Bit 23 | Bit 22 | Bit 21 | Bit 20 | Bit 19 | Bit 18 | Bit 17 | Bit 16 | |
| | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| IOAR1B | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| ETCR1B | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| P1DR | P17DR | P16DR | P15DR | P14DR | P13DR | P12DR | P11DR | P10DR | PORT |
| P3DR | — | P36DR | — | — | — | P32DR | P31DR | P30DR | |
| P7DR | P77DR | P76DR | P75DR | P74DR | — | — | P71DR | P70DR | |
| PADR | — | — | — | — | PA3DR | PA2DR | PA1DR | PA0DR | |
| PBDR | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR | |
| PCDR | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR | |
| PDDR | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR | |
| PEDR | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR | |
| PFDR | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR | |
| PGDR | — | — | — | PG4DR | PG3DR | PG2DR | PG1DR | PG0DR | |
| TCR_0 | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU_0 |
| TMDR_0 | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 | |
| TIORH_0 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | |
| TIORL_0 | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 | |
| TIER_0 | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA | |
| TSR_0 | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA | |
| TCNT_0 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRA_0 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRB_0 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| TGRC_0 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | TPU_0 |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRD_0 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TCR_1 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU_1 |
| TMDR_1 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | |
| TIOR_1 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | |
| TIER_1 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | |
| TSR_1 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | |
| TCNT_1 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRA_1 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRB_1 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TCR_2 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | TPU_2 |
| TMDR_2 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | |
| TIOR_2 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | |
| TIER_2 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | |
| TSR_2 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | |
| TCNT_2 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRA_2 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| TGRB_2 | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | |
| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| EXMDLSTP | — | — | — | — | — | — | RTCSTOP | USBSTOP1 | SYSTEM |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| RSECDR | BSY | SC12 | SC11 | SC10 | SC03 | SC02 | SC01 | SC00 | RTC |
| RMINDR | BSY | MN12 | MN11 | MN10 | MN03 | MN02 | MN01 | MN00 | |
| RHRDR | BSY | — | HR11 | HR10 | HR03 | HR02 | HR01 | HR00 | |
| RWKDR | BSY | — | — | — | — | WK2 | WK1 | KWK0 | |
| RTCCR1 | RUN | 12/24 | PM | RST | — | — | — | — | |
| RTCCR2 | — | — | FOIE | WKIE | DYIE | HRIE | MNIE | SEIE | |
| RTCCSR | — | RCS6 | RCS5 | — | RCS3 | RCS2 | RCS1 | RCS0 | |
| DMACR0A*[1] | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | DMAC |
| DMACR0A*[2] | DTSZ | SAID | SAIDE | BLKDIR | BLKE | — | — | — | |
| DMACR0B*[1] | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | |
| DMACR0B*[2] | — | DAID | DAIDE | — | DTF3 | DTF2 | DTF1 | DTF0 | |
| DMACR1A*[1] | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | |
| DMACR1A*[2] | DTSZ | SAID | SAIDE | BLKDIR | BLKE | — | — | — | |
| DMACR1B*[1] | DTSZ | DTID | RPE | DTDIR | DTF3 | DTF2 | DTF1 | DTF0 | |
| DMACR1B*[2] | — | DAID | DAIDE | — | DTF3 | DTF2 | DTF1 | DTF0 | |
| DMABCR*[1] | FAE1 | FAE0 | — | — | DTA1B | DTA1A | DTA0B | DTA0A | |
| | DTE1B | DTE1A | DTE0B | DTE0A | DTIE1B | DTIE1A | DTIE0B | DTIE0A | |
| DMABCR*[2] | FAE1 | FAE0 | — | — | DTA1 | — | DTA0 | — | |
| | DTME1 | DTE1 | DTME0 | DTE0 | DTIE1B | DTIE1A | DTIE0B | DTIE0A | |
| TCSR | OVF | WT/$\overline{IT}$ | TME | — | — | CKS2 | CKS1 | CKS0 | WDT |
| TCNT | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| RSTCSR | WOVF | RSTE | RSTS | — | — | — | — | — | |
| SMR_0 | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI_0 |
| SMR_0*[3] | GM | BLK | PE | O/$\overline{E}$ | BCP1 | BCP0 | CKS1 | CKS0 | |
| BRR_0 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| SCR_0 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | |
| TDR_0 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| SSR_0 | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT | |
| SSR_0*[3] | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT | |
| RDR_0 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| SCMR_0 | — | — | — | — | SDIR | SINV | — | SMIF | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| SMR_2 | C/$\overline{A}$ | CHR | PE | O/$\overline{E}$ | STOP | MP | CKS1 | CKS0 | SCI_2 |
| SMR_2*[3] | GM | BLK | PE | O/$\overline{E}$ | BCP1 | BCP0 | CKS1 | CKS0 | |
| BRR_2 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| SCR_2 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | |
| TDR_2 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| SSR_2 | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT | |
| SSR_2*[3] | TDRE | RDRF | ORER | ERS | PER | TEND | MPB | MPBT | |
| RDR_2 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| SCMR_2 | — | — | — | — | SDIR | SINV | — | SMIF | |
| ADDRAH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | A/D |
| ADDRAL | AD1 | AD0 | — | — | — | — | — | — | |
| ADDRBH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | |
| ADDRBL | AD1 | AD0 | — | — | — | — | — | — | |
| ADDRCH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | |
| ADDRCL | AD1 | AD0 | — | — | — | — | — | — | |
| ADDRDH | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | |
| ADDRDL | AD1 | AD0 | — | — | — | — | — | — | |
| ADCSR | ADF | ADIE | ADST | SCAN | — | CH2 | CH1 | CH0 | |
| ADCR | TRGS1 | TRGS0 | — | — | CKS1 | CKS0 | — | — | |
| TCSR_1 | — | — | — | PSS | — | — | — | — | SYSTEM |
| FLMCR1 | FWE | SWE1 | ESU1 | PSU1 | EV1 | PV1 | E1 | P1 | FLASH |
| FLMCR2 | FLER | — | — | — | — | — | — | — | |
| EBR1 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 | |
| EBR2 | — | — | — | — | — | — | EB9 | EB8 | |

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| PORT1 | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | PORT |
| PORT3 | — | P36 | — | — | — | P32 | P31 | P30 | |
| PORT4 | — | — | — | — | P43 | P42 | P41 | P40 | |
| PORT7 | P77 | P76 | P75 | P74 | — | — | P71 | P70 | |
| PORT9 | P97 | P96 | — | — | — | — | — | — | |
| PORTA | — | — | — | — | PA3 | PA2 | PA1 | PA0 | |
| PORTB | PB7 | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 | |
| PORTC | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 | |
| PORTD | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 | |
| PORTE | PE7 | PE6 | PE5 | PE4 | PE3 | PE2 | PE1 | PE0 | |
| PORTF | PF7 | PF6 | PF5 | PF4 | PF3 | PF2 | PF1 | PF0 | |
| PORTG | — | — | — | PG4 | PG3 | PG2 | PG1 | PG0 | |

Notes: 1. Short address mode
2. Full address mode
3. Smart card interface

## 21.3 Register States in Each Operating Mode

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UCTLR | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | USB |
| UTSTRA | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UDMAR | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UDRR | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTRG0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UFCLR0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UESTL0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UESTL1 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UEDR0s | —* | — | — | — | — | — | — | — | — | — | — | |
| UEDR0i | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UEDR0o | —* | — | — | — | — | — | — | — | — | — | — | |
| UEDR3 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UEDR1 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UEDR2 | —* | — | — | — | — | — | — | — | — | — | — | |
| UESZ0o | —* | — | — | — | — | — | — | — | — | — | — | |
| UESZ2 | —* | — | — | — | — | — | — | — | — | — | — | |
| UIFR0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UIFR1 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UIFR3 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UIER0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UIER1 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UIER3 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UISR0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UISR1 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UISR3 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UDSR | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UCVR | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UTSTR0 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | USB |
| UTSTR1 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTSTR2 | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTSTRB | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTSTRC | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTSTRD | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTSTRE | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| UTSTRF | Initialized* | — | — | — | — | — | — | — | — | — | Initialized | |
| SCRX | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | FLASH |
| SBYCR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | SYSTEM |
| SYSCR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| SCKCR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| MDCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| MSTPCRA | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| MSTPCRB | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| MSTPCRC | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| PFCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | BSC |
| LPWRCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | SYSTEM |
| OUTCR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | PORT |
| SEMRA_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | SCI_0 |
| SEMRB_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| ISCRH | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | INT |
| ISCRL | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IER | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| ISR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| P1DDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | PORT |
| P3DDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| P7DDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PADDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PBDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PCDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | PORT |
| PEDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PFDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PGDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PAPCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PBPCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PCPCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PDPCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PEPCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| P3ODR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PAODR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| TSTR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | TPU |
| TSYR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRA | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | INT |
| IPRB | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRC | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRD | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRE | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRF | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRG | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRJ | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRK | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| IPRM | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| ABWCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | BSC |
| ASTCR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| WCRH | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| WCRL | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| BCRH | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| BCRL | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| RAMER | Initialized | — | — | — | — | — | — | — | — | — | Initialized | FLASH |

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAR0A | — | — | — | — | — | — | — | — | — | — | — | DMAC |
| IOAR0A | — | — | — | — | — | — | — | — | — | — | — | |
| ETCR0A | — | — | — | — | — | — | — | — | — | — | — | |
| MAR0B | — | — | — | — | — | — | — | — | — | — | — | |
| IOAR0B | — | — | — | — | — | — | — | — | — | — | — | |
| ETCR0B | — | — | — | — | — | — | — | — | — | — | — | |
| MAR1A | — | — | — | — | — | — | — | — | — | — | — | |
| IOAR1A | — | — | — | — | — | — | — | — | — | — | — | |
| ETCR1A | — | — | — | — | — | — | — | — | — | — | — | |
| MAR1B | — | — | — | — | — | — | — | — | — | — | — | |
| IOAR1B | — | — | — | — | — | — | — | — | — | — | — | |
| ETCR1B | — | — | — | — | — | — | — | — | — | — | — | |
| P1DR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | PORT |
| P3DR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| P7DR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PADR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PBDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PCDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PDDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PEDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PFDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| PGDR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| TCR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | TPU_0 |
| TMDR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIORH_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIORL_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIER_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TSR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TCNT_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRA_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRB_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRC_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRD_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TCR_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | TPU_1 |
| TMDR_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIOR_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIER_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TSR_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TCNT_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRA_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRB_1 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TCR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | TPU_2 |
| TMDR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIOR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TIER_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TSR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TCNT_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRA_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TGRB_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| EXMDLSTP | Initialized | — | — | — | — | — | — | — | — | — | Initialized | SYSTEM |
| RSECDR | — | — | — | — | — | — | — | — | — | — | Initialized | RTC |
| RMINDR | — | — | — | — | — | — | — | — | — | — | Initialized | |
| RHRDR | — | — | — | — | — | — | — | — | — | — | Initialized | |
| RWKDR | — | — | — | — | — | — | — | — | — | — | Initialized | |
| RTCCR1 | — | — | — | — | — | — | — | — | — | — | Initialized | |
| RTCCR2 | — | — | — | — | — | — | — | — | — | — | Initialized | |
| RTCCSR | Initialized | — | — | — | — | — | — | — | — | — | Initialized | |
| DMACR0A | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | DMAC |
| DMACR0B | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| DMACR1A | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| DMACR1B | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| DMABCR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TCSR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | WDT |
| TCNT | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| RSTCSR | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | SCI_0 |
| BRR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| SCR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TDR_0 | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| SSR_0 | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| RDR_0 | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| SCMR_0 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| SMR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | SCI_2 |
| BRR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| SCR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| TDR_2 | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| SSR_2 | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| RDR_2 | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| SCMR_2 | Initialized | Initialized | — | — | — | — | — | — | — | — | Initialized | |
| ADDRAH | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | A/D |
| ADDRAL | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADDRBH | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADDRBL | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADDRCH | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADDRCL | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADDRDH | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADDRDL | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADCSR | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| ADCR | Initialized | Initialized | — | — | — | Initialized | Initialized | Initialized | Initialized | Initialized | Initialized | |
| TCSR_1 | Initialized | Initialized | — | — | — | — | — | — | — | Initialized | Initialized | SYSTEM |
| FLMCR1 | Initialized | — | — | — | — | — | — | — | — | Initialized | Initialized | FLASH |
| FLMCR2 | Initialized | — | — | — | — | — | — | — | — | Initialized | Initialized | |
| EBR1 | Initialized | — | — | — | — | — | — | — | — | Initialized | Initialized | |
| EBR2 | Initialized | — | — | — | — | — | — | — | — | Initialized | Initialized | |

| Register Name | Power-on Reset | Manual Reset | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Software Standby | Hardware Standby | Module |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PORT1 | — | — | — | — | — | — | — | — | — | — | — | PORT |
| PORT3 | — | — | — | — | — | — | — | — | — | — | — | |
| PORT4 | — | — | — | — | — | — | — | — | — | — | — | |
| PORT7 | — | — | — | — | — | — | — | — | — | — | — | |
| PORT9 | — | — | — | — | — | — | — | — | — | — | — | |
| PORTA | — | — | — | — | — | — | — | — | — | — | — | |
| PORTB | — | — | — | — | — | — | — | — | — | — | — | |
| PORTC | — | — | — | — | — | — | — | — | — | — | — | |
| PORTD | — | — | — | — | — | — | — | — | — | — | — | |
| PORTE | — | — | — | — | — | — | — | — | — | — | — | |
| PORTF | — | — | — | — | — | — | — | — | — | — | — | |
| PORTG | — | — | — | — | — | — | — | — | — | — | — | |

Notes: —: is not initialized.

* The USB registers are no initialized by a power-on reset triggered by the WDT.

# Section 22 Electrical Characteristics

## 22.1 Absolute Maximum Ratings

Table 22.1 lists the absolute maximum ratings.

**Table 22.1 Absolute Maximum Ratings**

| Item | Symbol | Value | Unit |
|---|---|---|---|
| Power supply voltage | $V_{CC}$, $PLLV_{CC}$, $DrV_{CC}$ | –0.3 to +4.3 | V |
| Input voltage | $V_{in}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Reference voltage | $V_{ref}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Analog input voltage | $V_{AN}$ | –0.3 to $V_{CC}$ +0.3 | V |
| Operating temperature | $T_{opr}$ | Regular specifications: –20 to +75 | °C |
| | | Wide-range specifications: –40 to +85* | °C |
| Storage temperature | $T_{stg}$ | –55 to +125 | °C |

Caution: Permanent damage to the chip may result if absolute maximum rating are exceeded.

Note: * The operating temperature ranges for flash memory programming/erasing are $T_a$ = –20°C to +75°C.

## 22.2 Power Supply Voltage and Operating Frequency Range

Power supply voltage and operating frequency ranges (shaded areas) are shown in figure 22.1.

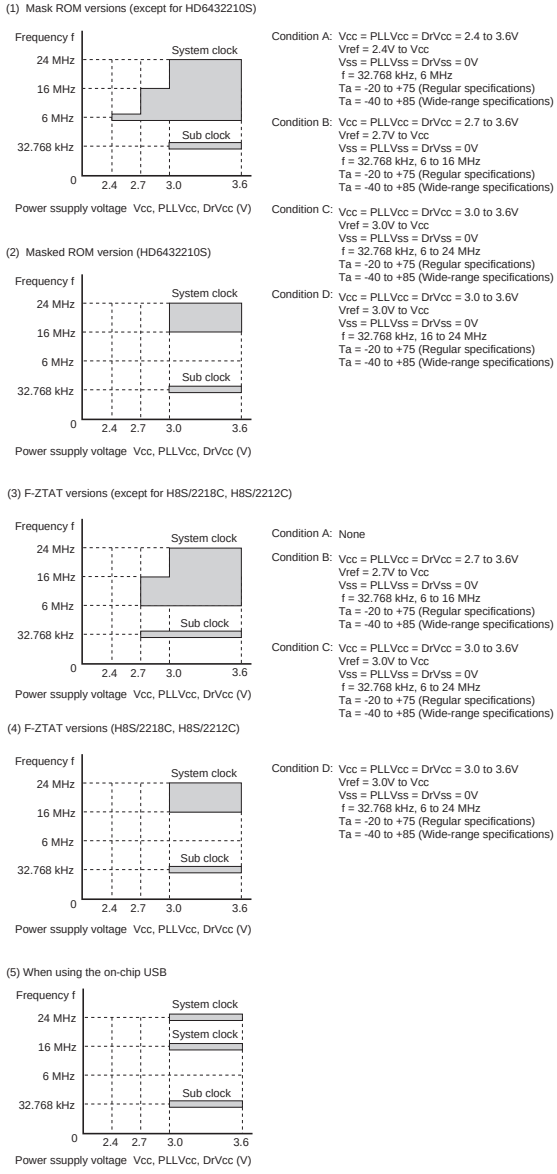



**Figure 22.1 Power Supply Voltage and Operating Ranges**

## 22.3 DC Characteristics

Table 22.2 lists the DC characteristics. Table 22.3 lists the permissible output currents.

**Table 22.2 DC Characteristics**

Condition A: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.4 V to 3.6 V, Vref = 2.4 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.7 V to 3.6 V, Vref = 2.7 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| Item | | Symbol | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Schmitt trigger input voltage | $\overline{IRQ0}$ to $\overline{IRQ4}$ | $V_T^-$ | $V_{CC} \times 0.2$ | — | — | V | |
| | $\overline{IRQ7}$ | $V_T^+$ | — | — | $V_{CC} \times 0.8$ | V | |
| | | $V_T^+ - V_T^-$ | $V_{CC} \times 0.05$ | — | — | V | |
| Input high voltage | $\overline{RES}$, $\overline{STBY}$, NMI, MD2 to MD0, $\overline{TRST}$, TCK, TMS, TDI, EMLE, VBUS, $\overline{UBPM}$, FWE*[4] | $V_{IH}$ | $V_{CC} \times 0.9$ | — | $V_{CC}$ + 0.3 | V | |
| | EXTAL, ports 1, 3, 4, 7, 9, and A to G | | $V_{CC} \times 0.8$ | — | $V_{CC}$ + 0.3 | V | |

| Item | | Symbol | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Input low voltage | $\overline{RES}$, $\overline{STBY}$, MD2 to MD0, $\overline{TRST}$, TCK, TMS, TDI, $\overline{EMLE}$, VBUS, $\overline{UBPM}$, FWE*[4] | $V_{IL}$ | –0.3 | — | $V_{CC} \times 0.1$ | V | |
| | EXTAL, NMI, ports 1, 3, 4, 7, 9, and A to G | | –0.3 | — | $V_{CC} \times 0.2$ | V | |
| Output high voltage | All output pins | $V_{OH}$ | $V_{CC}$ – 0.5 | — | — | V | $I_{OH}$ = –200 µA |
| | | | $V_{CC}$ – 1.0 | — | — | V | $I_{OH}$ = –1 mA |
| Output low voltage | All output pins | $V_{OL}$ | — | — | 0.4 | V | $I_{OL}$ = 0.8 mA |
| Input leakage current | $\overline{RES}$, VBUS, $\overline{UBPM}$, $\overline{STBY}$, NMI, $\overline{EMLE}$, MD2 to MD0, FWE*[4], ports 4, 9 | $\lvert I_{in} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5 V |
| Three-state leakage current (off state) | Ports 1, 3, 7, and A to G | $\lvert I_{TSI} \rvert$ | — | — | 1.0 | µA | $V_{in}$ = 0.5 to $V_{CC}$ – 0.5 V |
| Input pull-up MOS current | Ports A to E TDI, TCK, TMS, $\overline{TRST}$ | $-I_P$ | 10 | — | 300 | µA | $V_{in}$ = 0 V |
| Input capacitance | $\overline{RES}$, NMI | $C_{in}$ | — | — | 30 | pF | $V_{in}$ = 0 V<br>f = 1 MHz |
| | All input pins other than $\overline{RES}$, NMI | | — | — | 15 | pF | $T_a$ = 25°C |
| Current dissipation*[1] | Normal operation | $I_{CC}$*[2] | — | 22<br>$V_{CC}$ = 3.3 V | 35<br>$V_{CC}$ = 3.6 V | mA | f = 16 MHz |
| | (USB halts) | | — | 31<br>$V_{CC}$ = 3.3 V | 50<br>$V_{CC}$ = 3.6 V | mA | f = 24 MHz |

| Item | | Symbol | Min. | Typ. | Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Current dissipation*[1] | Normal operation (USB operates) | $I_{CC}$*[2] | — | 30<br>$V_{CC}$ = 3.3 V | 45<br>$V_{CC}$ = 3.6 V | mA | f = 16 MHz, When PLL3 is used |
| | | | — | 41<br>$V_{CC}$ = 3.3 V | 60<br>$V_{CC}$ = 3.6 V | mA | f = 24 MHz, When PLL2 is used |
| | Sleep mode | | — | 16<br>$V_{CC}$ = 3.3 V | 30<br>$V_{CC}$ = 3.6 V | mA | f = 16 MHz, When USB and PLL are halted |
| | | | — | 22<br>$V_{CC}$ = 3.3 V | 45<br>$V_{CC}$ = 3.6 V | mA | f = 24 MHz, When USB and PLL are halted |
| | All modules other than flash memory stopped | | — | 16<br>$V_{CC}$ = 3.3 V | — | mA | f = 16 MHz (reference value) |
| | | | — | 24<br>$V_{CC}$ = 3.3 V | — | mA | f = 24 MHz (reference value) |
| | Subactive mode | | — | 45 | 180 | µA | Vcc = 3.3 V, EMLE = 0 When crystal resonator (32.768 kHz) is used |
| | | | — | 30*[5] | — | | |
| | Subsleep mode | | — | 35 | 100 | µA | |
| | | | — | 20*[5] | — | | |
| | Watch mode | | — | 5 | 40 | µA | |
| | Standby mode*[3] | | — | 1.0 | 10 | µA | $T_a \leq 50°C$ 32.768 kHz RTC halted EMLE = 0 |
| | | | — | — | 50 | µA | $50°C < T_a$ 32.768 kHz RTC halted EMLE = 0 |
| Reference power supply current | During A/D conversion | $AI_{CC}$ | — | 1.3 | 2.5 | mA | $V_{ref}$ = 3.3 V |
| | Idle | | — | 0.01 | 5.0 | µA | |
| RAM standby voltage | | $V_{RAM}$ | 2.0 | — | — | V | |

Notes: If the A/D converter is not used, the Vref pin should not be open. Even if the A/D converter is not used, connect the Vref pin to Vcc.

1. Current dissipation values are for $V_{IH}$ min. = $V_{CC}$ – 0.2 V and $V_{IL}$ max. = 0.2 V, with all output pins unloaded and the on-chip MOS pull-up transistors in the off state.
2. $I_{CC}$ depends on $V_{CC}$ and f as follows (Reference):
   $I_{CC}$ max. = 5 (mA) + 0.52 (mA/(MHz x V)) × $V_{CC}$ × f (normal operation, USB halted)
   $I_{CC}$ max. = 9 (mA) + 0.60 (mA/(MHz x V)) × $V_{CC}$ × f (normal operation, USB operated)
   $I_{CC}$ max. = 1 (mA) + 0.51 (mA/(MHz x V)) × $V_{CC}$ × f (sleep mode)
3. The values are for $V_{RAM}$ < $V_{CC}$ < 2.7 V, $V_{IH}$ min. = $V_{CC}$ × 0.9, and $V_{IL}$ max. = 0.3 V.
4. The FWE pin is effective only in the F-ZTAT version.
5. Reference value when setting the flash memory module stop mode is carried out while the on-chip RAM program is executed. The value is effective in the F-ZTAT version.

### Table 22.3 Permissible Output Currents

Condition A: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.4 V to 3.6 V, Vref = 2.4 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.7 V to 3.6 V, Vref = 2.7 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

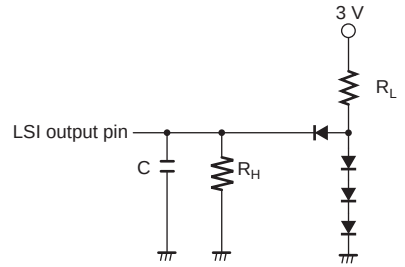
Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| Item | | Symbol | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Permissible output low current (per pin) | All output pins | $I_{OL}$ | — | — | 1.0 | mA |
| Permissible output low current (total) | Total of all output pins | $\sum I_{OL}$ | — | — | 60 | mA |
| Permissible output high current (per pin) | All output pins | $-I_{OH}$ | — | — | 1.0 | mA |
| Permissible output high current (total) | Total of all output pins | $\sum -I_{OH}$ | — | — | 30 | mA |

Note: To protect chip reliability, do not exceed the output current values in table 22.3.

## 22.4 AC Characteristics

Figure 22.2 shows, the test conditions for the AC characteristics.




C=30 pF
$R_L$= 2.4 kΩ
$R_H$=12 kΩ
Input/output timing measurement levels
• Low level : 1.3 V (2.4 V ≤ Vcc < 2.7 V)
: 0.8 V (2.7 V ≤ Vcc ≤ 3.6 V)
• High level : 1.3 V (2.4 V ≤ Vcc < 2.7 V)
: 2.0 V (2.7 V ≤ Vcc ≤ 3.6 V)

**Figure 22.2 Output Load Circuit**

### 22.4.1 Clock Timing

Table 22.4 lists the clock timing

**Table 22.4 Clock Timing**

Condition A: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.4 V to 3.6 V, Vref = 2.4 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.7 V to 3.6 V, Vref = 2.7 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | Condition C | | Condition D | | | Test |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** | **Conditions** |
| Clock cycle time | $t_{cyc}$ | 166.6 | | 62.5 | 166.6 | 41.6 | 166.6 | 41.6 | 62.5 | ns | |
| Clock high pulse width | $t_{CH}$ | 50 | — | 20 | — | 13 | — | 13 | — | ns | |
| Clock low pulse width | $t_{CL}$ | 50 | — | 20 | — | 13 | — | 13 | — | ns | |
| Clock rise time | $t_{Cr}$ | — | 25 | — | 10 | — | 7 | — | 7 | ns | |
| Clock fall time | $t_{Cf}$ | — | 25 | — | 10 | — | 7 | — | 7 | ns | |
| Oscillation stabilization time at reset (crystal) | $t_{OSC1}$ | 40 | — | 20 | — | 20 | — | 20 | — | ms | Figure 22.4 |
| Oscillation stabilization time in software standby (crystal) | $t_{OSC2}$ | 16 | — | 8 | — | 8 | — | 8 | — | ms | Figures 20.4, 19.2<br>$C_{L1}$ = $C_{L2}$ = 10 to 22 pF |
| | | 16 | — | 8 | — | 4 | — | 4 | — | ms | Figures 20.4, 19.2<br>$C_{L1}$ = $C_{L2}$ = 10 to 15 pF |

| Item | Symbol | Condition A Min. | Condition A Max. | Condition B Min. | Condition B Max. | Condition C Min. | Condition C Max. | Condition D Min. | Condition D Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|---|---|---|---|
| External clock output stabilization delay time | $t_{DEXT}$ | 1000 | — | 500 | — | 500 | — | 500 | — | µs | Figure 22.4 |
| Subclock stabilization time | $t_{OSC3}$ | — | 4 | — | 2 | — | 2 | — | 2 | s | |
| Subclock oscillator frequency | $f_{SUB}$ | 32.768 | | 32.768 | | 32.768 | | 32.768 | | kHz | |
| Subclock ($\phi_{SUB}$) cycle time | $f_{SUB}$ | 30.5 | | 30.5 | | 30.5 | | 30.5 | | µs | |

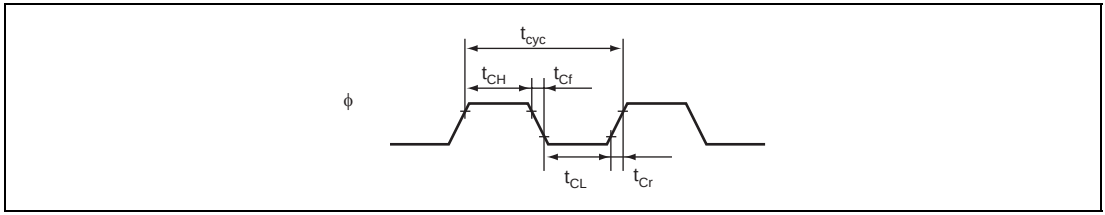



**Figure 22.3 System Clock Timing**

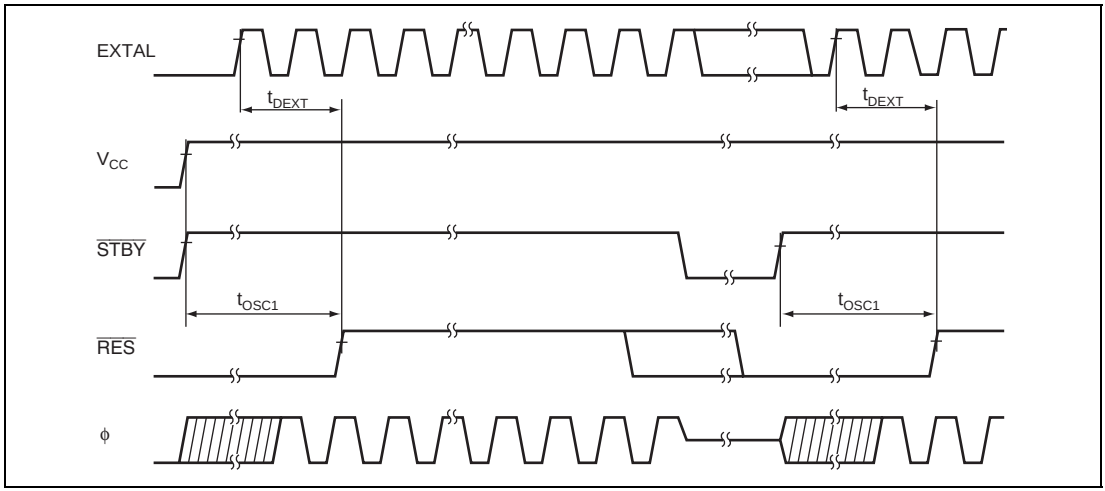



**Figure 22.4 Oscillation Stabilization Timing**

### 22.4.2 Control Signal Timing

Table 22.5 lists the control signal timing.

**Table 22.5 Control Signal Timing**

Condition A: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.4 V to 3.6 V, Vref = 2.4 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.7 V to 3.6 V, Vref = 2.7 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| | | Condition A | | Condition B, C, D | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** | **Test Conditions** |
| $\overline{RES}$ setup time | $t_{RESS}$ | 350 | — | 250 | — | ns | Figure 22.5 |
| $\overline{RES}$ pulse width | $t_{RESW}$ | 20 | — | 20 | — | $t_{cyc}$ | |
| $\overline{MRES}$ setup time | $t_{MRESS}$ | 350 | — | 250 | — | ns | |
| $\overline{MRES}$ pulse width | $t_{MRESW}$ | 20 | — | 20 | — | $t_{cyc}$ | |
| NMI setup time | $t_{NMIS}$ | 350 | — | 250 | — | ns | Figure 22.6 |
| NMI hold time | $t_{NMIH}$ | 10 | — | 10 | — | ns | |
| NMI pulse width (exiting software standby mode) | $t_{NMIW}$ | 300 | — | 200 | — | ns | |
| $\overline{IRQ}$ setup time | $t_{IRQS}$ | 350 | — | 250 | — | ns | |
| $\overline{IRQ}$ hold time | $t_{IRQH}$ | 10 | — | 10 | — | ns | |
| $\overline{IRQ}$ pulse width (exiting software standby mode) | $t_{IRQW}$ | 300 | — | 200 | — | ns | |

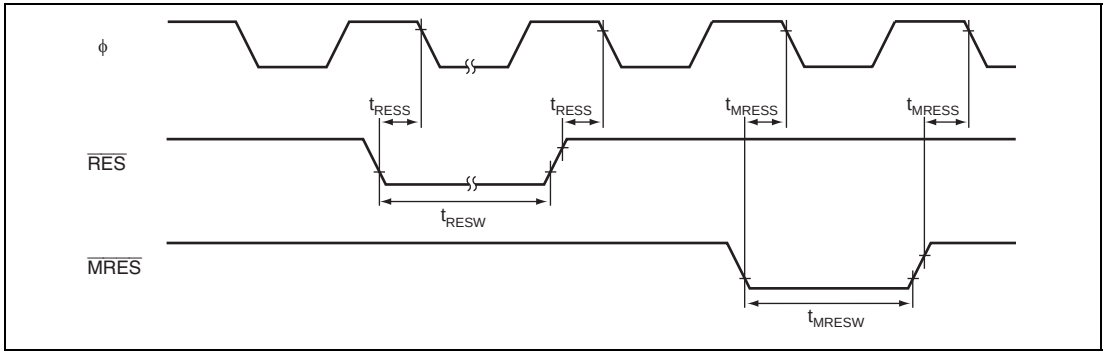



**Figure 22.5 Reset Input Timing**

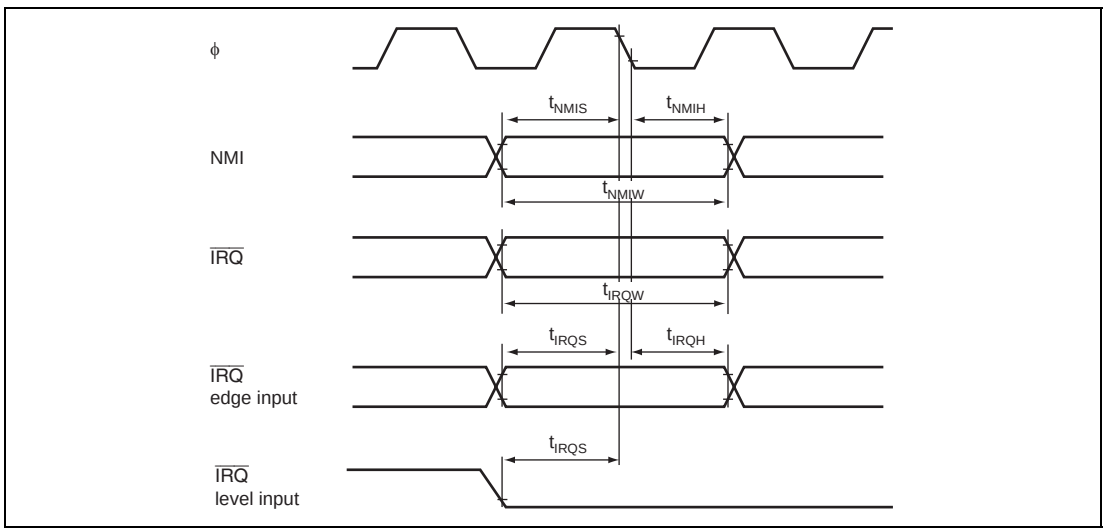



**Figure 22.6 Interrupt Input Timing**

### 22.4.3 Bus Timing

Table 22.6 shows, Bus Timing.

**Table 22.6 Bus Timing**

Condition A: $V_{CC}$ = PLL $V_{CC}$ =Dr $V_{CC}$ =2.4 V to 3.6 V, Vref=2.4 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ =Dr $V_{CC}$ =2.7 V to 3.6 V, Vref=2.7 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ =Dr $V_{CC}$ =3.0 V to 3.6 V, Vref=3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| | | Condition A | | Condition B | | Condition C, D | | | Test |
|---|---|---|---|---|---|---|---|---|---|
| **Item** | **Symbol** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** | **Conditions** |
| Address delay time | $t_{AD}$ | — | 90 | — | 50 | — | 30 | ns | Figures 22.7, 22.8, 22.10 |
| Address setup time | $t_{AS}$ | 0.5 × $t_{cyc}$ – 60 | — | 0.5 × $t_{cyc}$ – 30 | — | 0.5 × $t_{cyc}$ – 20 | — | ns | |
| Address hold time | $t_{AH}$ | 0.5 × $t_{cyc}$ – 30 | — | 0.5 × $t_{cyc}$ – 15 | — | 0.5 × $t_{cyc}$ – 8 | — | ns | |
| $\overline{CS}$ delay time | $t_{CSD}$ | — | 90 | — | 50 | — | 30 | ns | Figures 22.7, 22.8 |
| $\overline{AS}$ delay time | $t_{ASD}$ | — | 90 | — | 50 | — | 25 | ns | Figures 22.7, 22.8, 22.10 |
| $\overline{RD}$ delay time 1 | $t_{RSD1}$ | — | 90 | — | 50 | — | 25 | ns | Figures 22.7, 22.8 |
| $\overline{RD}$ delay time 2 | $t_{RSD2}$ | — | 90 | — | 50 | — | 25 | ns | Figures 22.7, 22.8, 22.10 |
| Read data setup time | $t_{RDS}$ | 50 | — | 30 | — | 20 | — | ns | |
| Read data hold time | $t_{RDH}$ | 0 | — | 0 | — | 0 | — | ns | |
| Read data access time 2 | $t_{ACC2}$ | — | 1.5 × $t_{cyc}$ – 90 | — | 1.5 × $t_{cyc}$ – 65 | — | 1.5 × $t_{cyc}$ – 35 | ns | Figure 22.7 |

| Item | Symbol | Condition A Min. | Condition A Max. | Condition B Min. | Condition B Max. | Condition C, D Min. | Condition C, D Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|---|---|
| Read data access time 3 | $t_{ACC3}$ | — | $2.0 \times t_{cyc} - 90$ | — | $2.0 \times t_{cyc} - 65$ | — | $2.0 \times t_{cyc} - 40$ | ns | Figures 22.7, 22.10 |
| Read data access time 4 | $t_{ACC4}$ | — | $2.5 \times t_{cyc} - 90$ | — | $2.5 \times t_{cyc} - 65$ | — | $2.5 \times t_{cyc} - 35$ | ns | Figure 22.8 |
| Read data access time 5 | $t_{ACC5}$ | — | $3.0 \times t_{cyc} - 90$ | — | $3.0 \times t_{cyc} - 65$ | — | $3.0 \times t_{cyc} - 40$ | ns | |
| $\overline{WR}$ delay time 1 | $t_{WRD1}$ | — | 90 | — | 50 | — | 20 | ns | Figure 22.8 |
| $\overline{WR}$ delay time 2 | $t_{WRD2}$ | — | 90 | — | 50 | — | 25 | ns | Figures 22.7, 22.8 |
| $\overline{WR}$ pulse width 1 | $t_{WSW1}$ | $1.0 \times t_{cyc} - 60$ | — | $1.0 \times t_{cyc} - 30$ | — | $1.0 \times t_{cyc} - 20$ | — | ns | Figure 22.7 |
| $\overline{WR}$ pulse width 2 | $t_{WSW2}$ | $1.5 \times t_{cyc} - 60$ | — | $1.5 \times t_{cyc} - 30$ | — | $1.5 \times t_{cyc} - 20$ | — | ns | Figure 22.8 |
| Write data delay time | $t_{WDD}$ | — | 100 | — | 50 | — | 30 | ns | Figures 22.7, 22.8 |
| Write data setup time | $t_{WDS}$ | $0.5 \times t_{cyc} - 80$ | — | $0.5 \times t_{cyc} - 30$ | — | $0.5 \times t_{cyc} - 20$ | — | ns | Figure 22.8 |
| Write data hold time | $t_{WDH}$ | $0.5 \times t_{cyc} - 60$ | — | $0.5 \times t_{cyc} - 15$ | — | $0.5 \times t_{cyc} - 10$ | — | ns | Figures 22.7, 22.8 |
| $\overline{WAIT}$ setup time | $t_{WTS}$ | 90 | — | 50 | — | 25 | — | ns | Figure 22.9 |
| $\overline{WAIT}$ hold time | $t_{WTH}$ | 10 | — | 10 | — | 5 | — | ns | |
| $\overline{BREQ}$ setup time | $t_{BRQS}$ | 90 | — | 50 | — | 25 | — | ns | Figure 22.11 |
| $\overline{BACK}$ delay time | $t_{BACD}$ | — | 90 | — | 50 | — | 35 | ns | |
| Bus-floating time | $t_{BZD}$ | — | 160 | — | 80 | — | 50 | ns | |

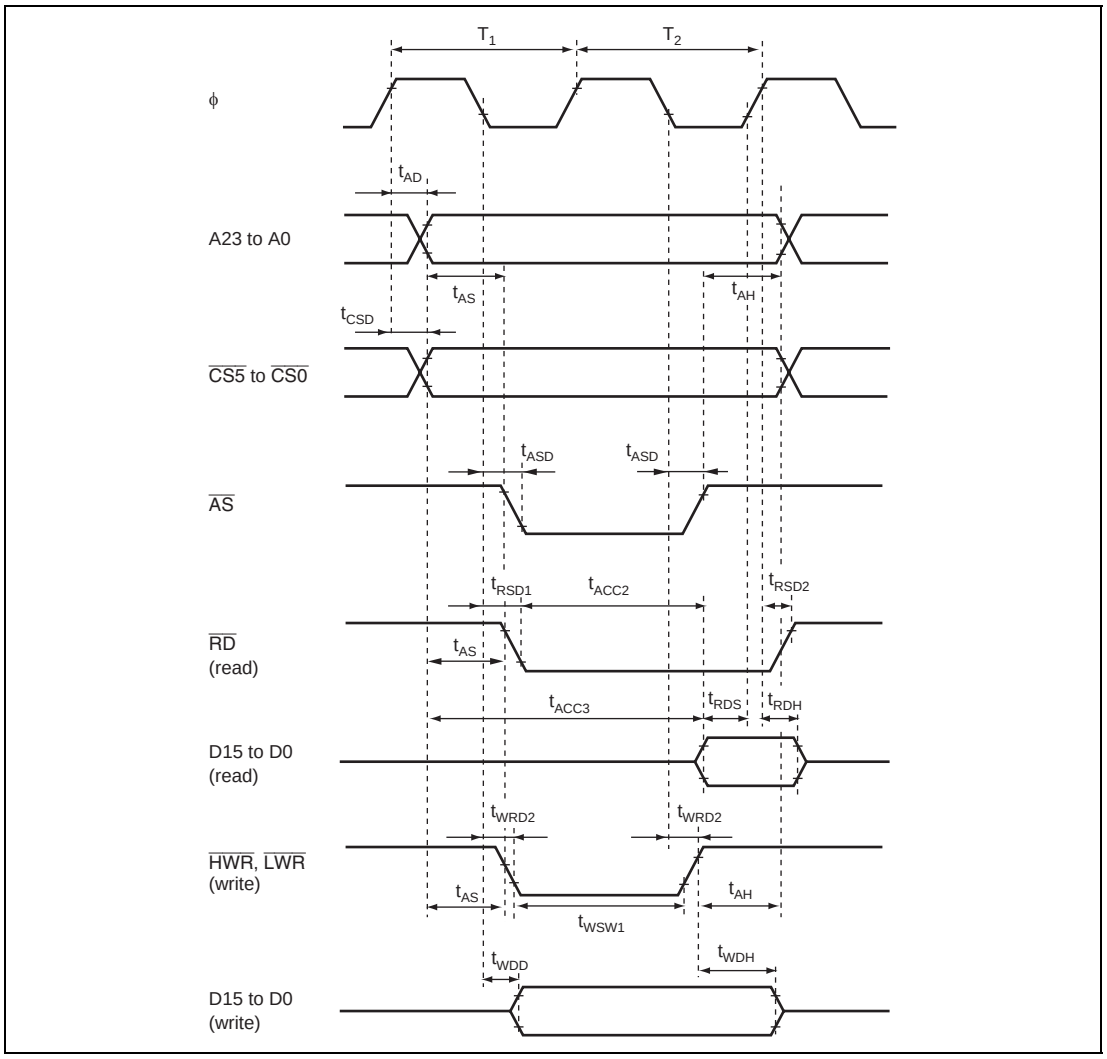



**Figure 22.7 Basic Bus Timing (Two-State Access)**

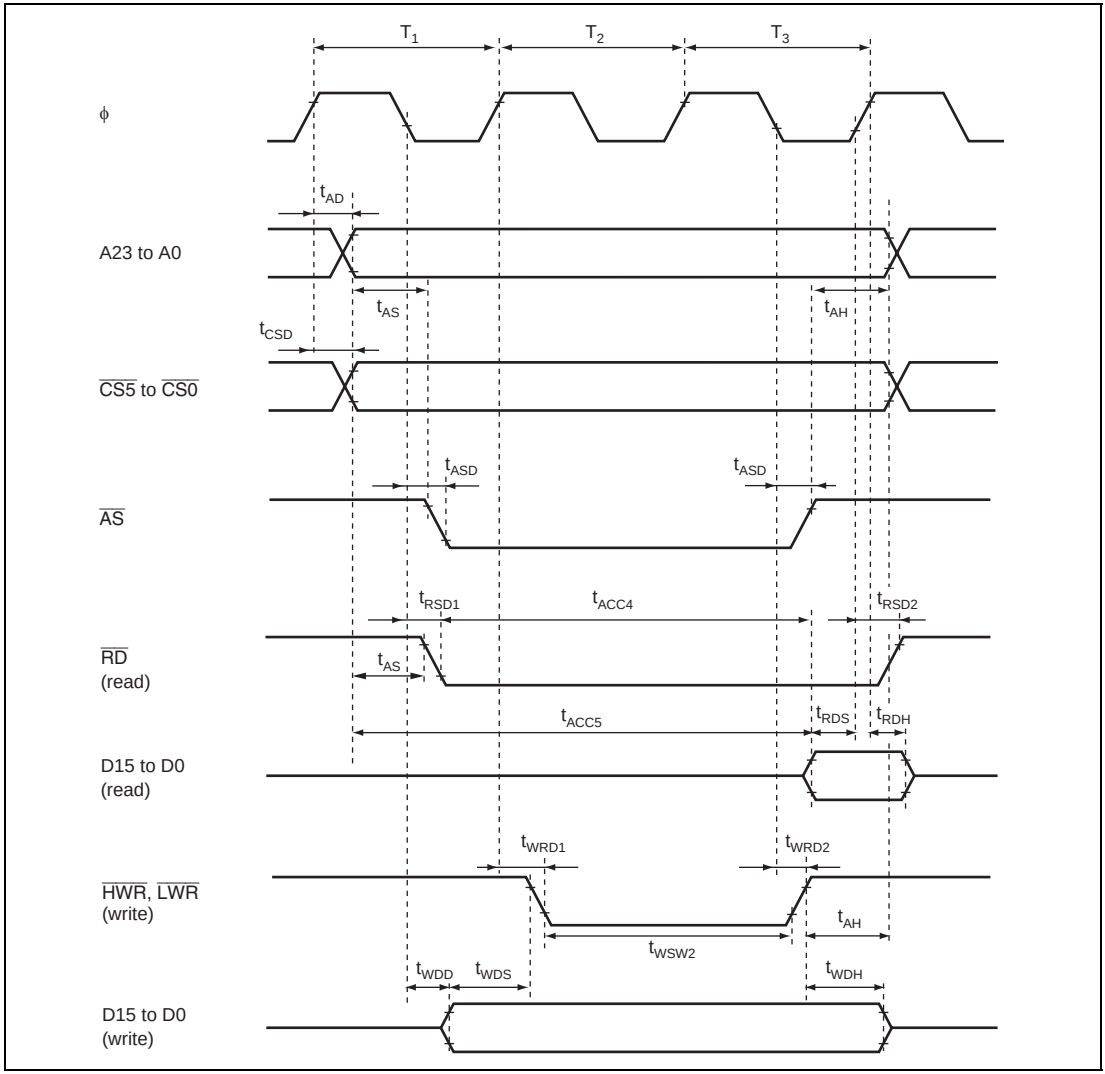



**Figure 22.8 Basic Bus Timing (Three-State Access)**

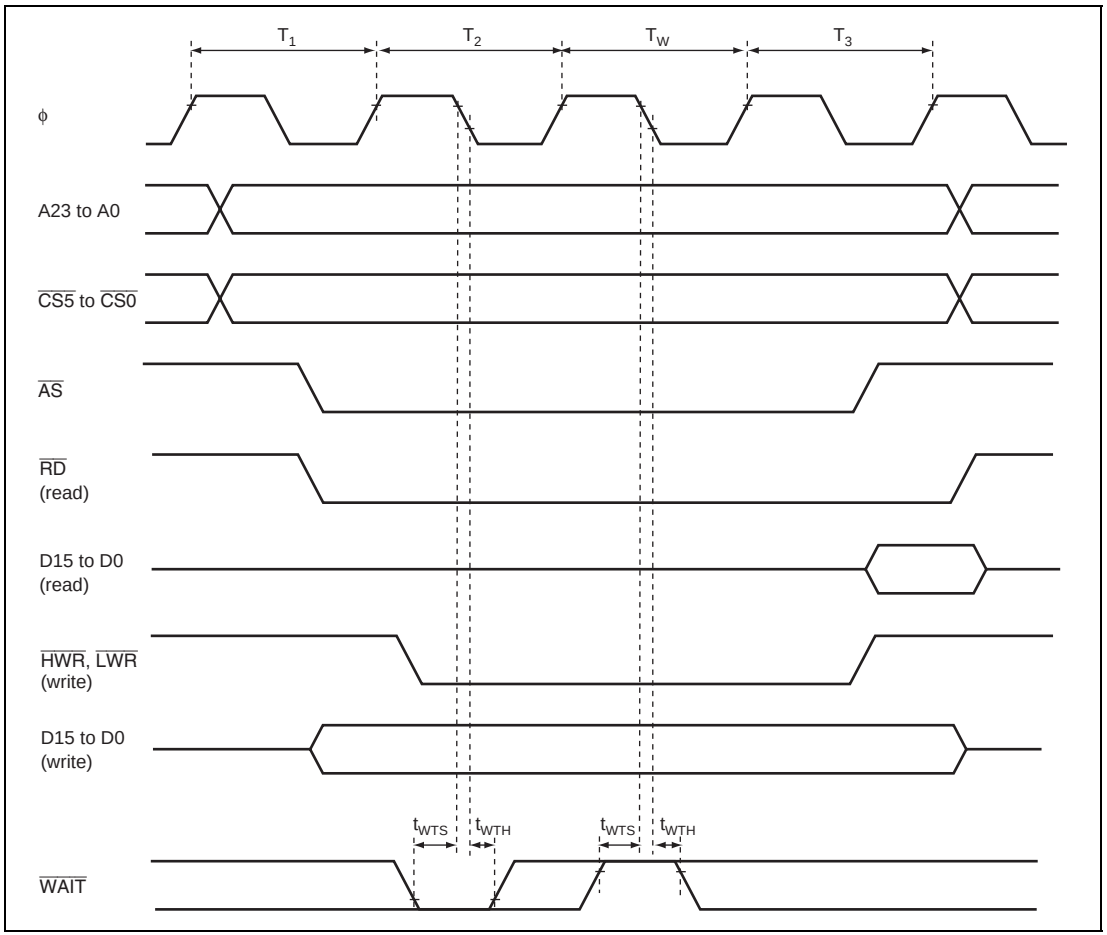



**Figure 22.9 Basic Bus Timing (Three-State Access with One Wait State)**

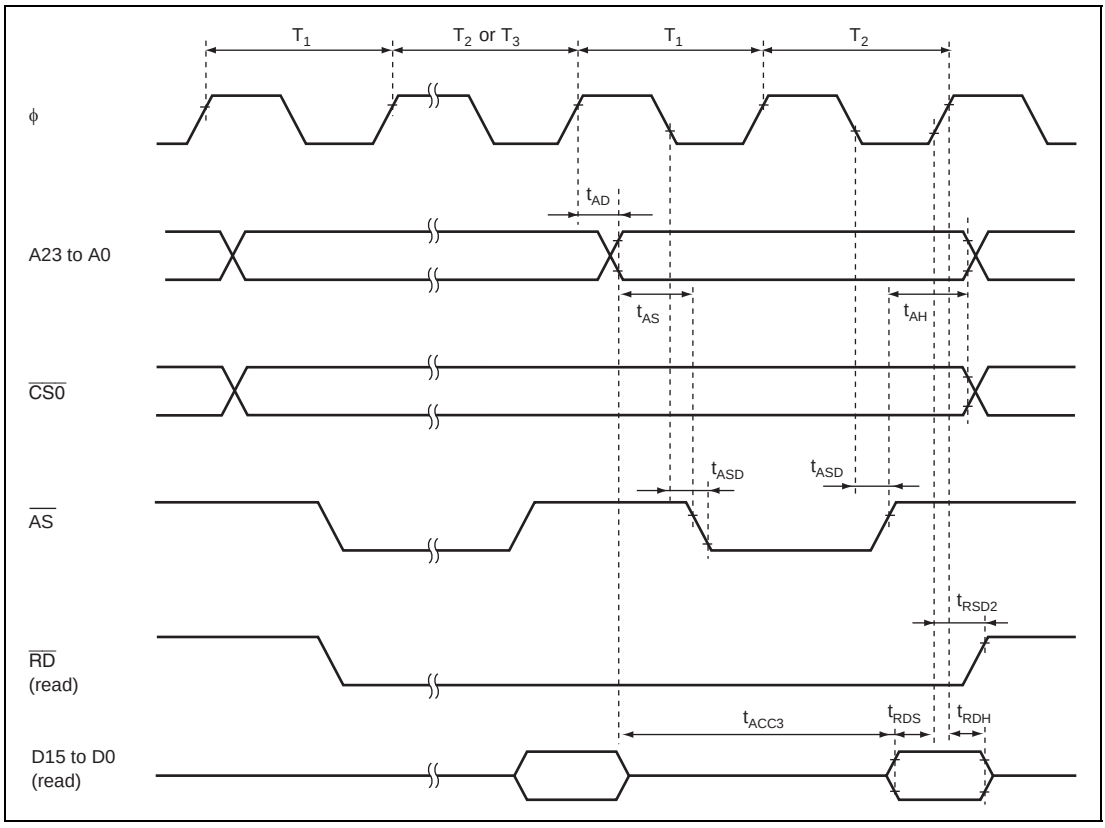



**Figure 22.10 Burst ROM Access Timing (Two-State Access)**

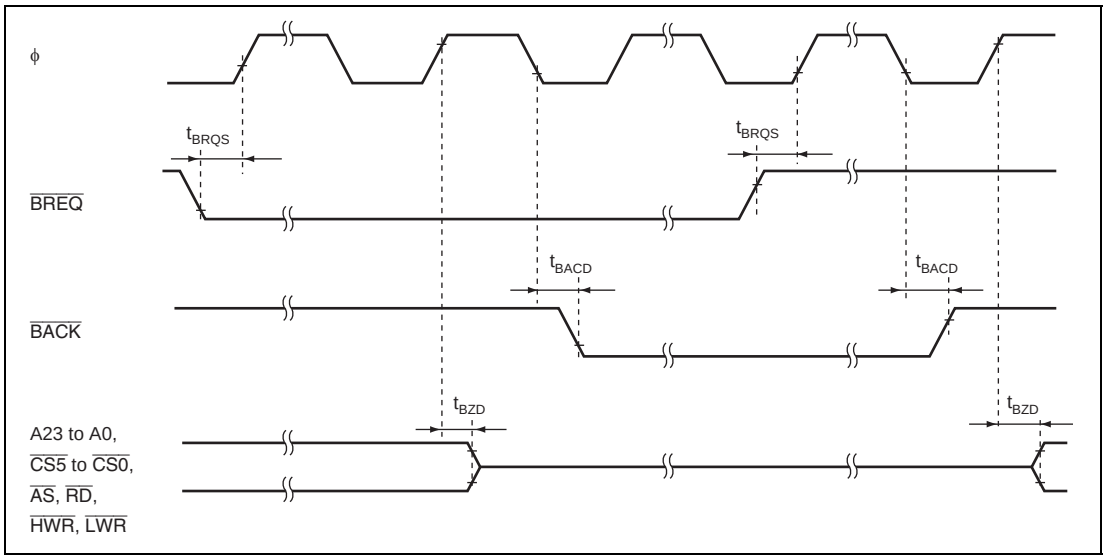



**Figure 22.11 External Bus Release Timing**

### 22.4.4 Timing of On-Chip Supporting Modules

Table 22.7 lists the timing of on-chip supporting modules.

**Table 22.7 Timing of On-Chip Supporting Modules**

Condition A: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.4 V to 3.6 V, Vref = 2.4 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.7 V to 3.6 V, Vref = 2.7 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = PLL$V_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| | | | Condition A | | Condition B | | Condition C, D | | | Test |
|---|---|---|---|---|---|---|---|---|---|---|
| **Item** | | **Symbol** | **Min.** | **Max.** | **Min.** | **Max.** | **Min.** | **Max.** | **Unit** | **Conditions** |
| I/O port | Output data delay time | $t_{PWD}$ | — | 150 | — | 60 | — | 40 | ns | Figure 22.12 |
| | Input data setup time | $t_{PRS}$ | 80 | — | 50 | — | 30 | — | | |
| | Input data hold time | $t_{PRH}$ | 50 | — | 50 | — | 30 | — | | |
| TPU | Timer output delay time | $t_{TOCD}$ | — | 150 | — | 60 | — | 40 | ns | Figure 22.13 |
| | Timer input setup time | $t_{TICS}$ | 60 | — | 40 | — | 30 | — | | |
| | Timer clock input setup time | $t_{TCKS}$ | 60 | — | 40 | — | 30 | — | ns | Figure 22.14 |
| | Timer clock pulse width — Single edge | $t_{TCKWH}$ | 1.5 | — | 1.5 | — | 1.5 | — | $t_{cyc}$ | |
| | Timer clock pulse width — Both edges | $t_{TCKWL}$ | 2.5 | — | 2.5 | — | 2.5 | — | | |

| Item | | | Symbol | Condition A Min. | Condition A Max. | Condition B Min. | Condition B Max. | Condition C,D Min. | Condition C,D Max. | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SCI | Input clock cycle | Asynchronous | $t_{Scyc}$ | 4 | — | 4 | — | 4 | — | $t_{cyc}$ | Figure 22.15 |
| | | Synchronous | | 6 | — | 6 | — | 6 | — | | |
| | Input clock pulse width | | $t_{SCKW}$ | 0.4 | 0.6 | 0.4 | 0.6 | 0.4 | 0.6 | $t_{Scyc}$ | |
| | Input clock rise time | | $t_{SCKr}$ | — | 1.5 | — | 1.5 | — | 1.5 | $t_{cyc}$ | |
| | Input clock fall time | | $t_{SCKf}$ | — | 1.5 | — | 1.5 | — | 1.5 | | |
| | Transmit data delay time | | $t_{TXD}$ | — | 150 | — | 60 | — | 40 | ns | Figure 22.16 |
| | Receive data setup time (synchronous) | | $t_{RXS}$ | 150 | — | 60 | — | 40 | — | | |
| | Receive data hold time (synchronous) | | $t_{RXH}$ | 150 | — | 60 | — | 40 | — | | |
| A/D converter | Trigger input setup time | | $t_{TRGS}$ | 60 | — | 40 | — | 30 | — | ns | Figure 22.17 |
| Boundary scan | TCK cycle time | | $t_{Tcyc}$ | 166.6 | — | 62.5 | — | 41.6 | — | ns | Figure 22.18 |
| | TCK high level pulse width | | $t_{TCKH}$ | 0.4 | 0.6 | 0.4 | 0.6 | 0.4 | 0.6 | $t_{Tcyc}$ | |
| | TCK low level pulse width | | $t_{TCKL}$ | 0.4 | 0.6 | 0.4 | 0.6 | 0.4 | 0.6 | $t_{Tcyc}$ | |
| | $\overline{\text{TRST}}$ pulse width | | $t_{TRSW}$ | 20 | — | 20 | — | 20 | — | $t_{Tcyc}$ | Figure 22.19 |
| | $\overline{\text{TRST}}$ setup time | | $t_{TRSS}$ | 350 | — | 250 | — | 250 | — | ns | |
| | TDI setup time | | $t_{TDIS}$ | 80 | — | 30 | — | 20 | — | ns | Figure 22.20 |
| | TDI hold time | | $t_{TDIH}$ | 10 | — | 10 | — | 10 | — | | |
| | TMS setup time | | $t_{TMSS}$ | 80 | — | 30 | — | 20 | — | | |
| | TMS hold time | | $t_{TMSH}$ | 10 | — | 10 | — | 10 | — | | |
| | TDO delay time | | $t_{TDOD}$ | — | 100 | — | 40 | — | 35 | | |

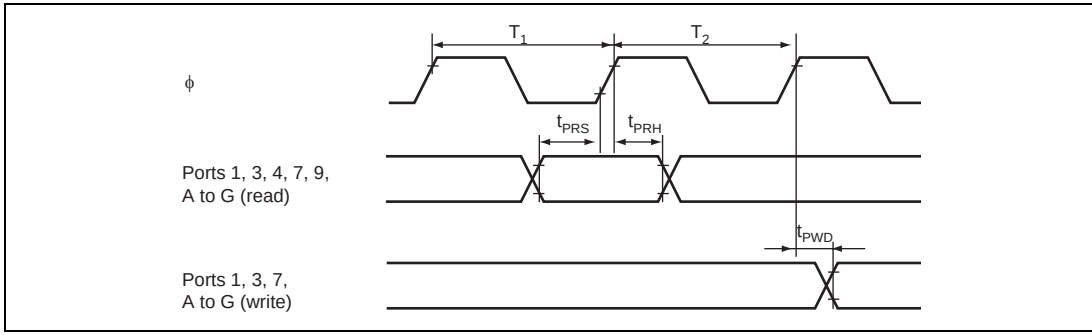



**Figure 22.12 I/O Port Input/Output Timing**

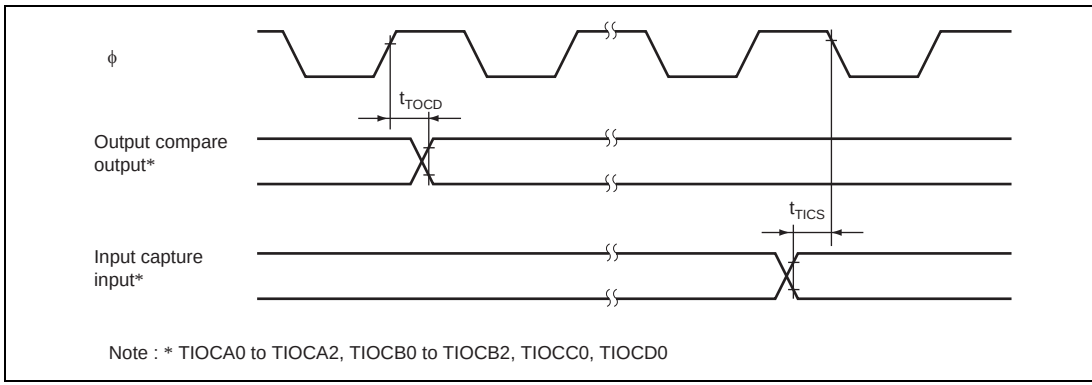



**Figure 22.13 TPU Input/Output Timing**

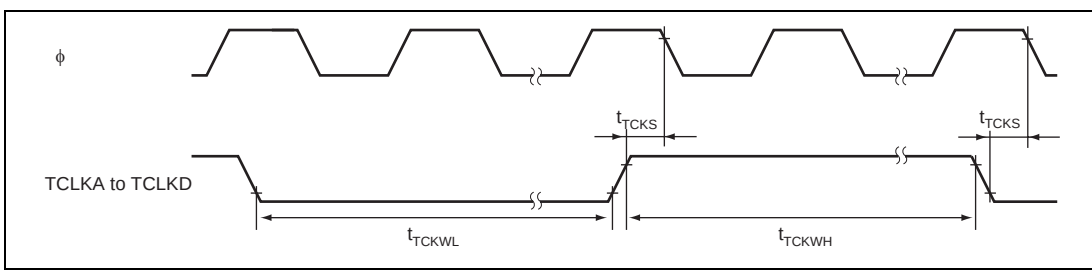



**Figure 22.14 TPU Clock Input Timing**

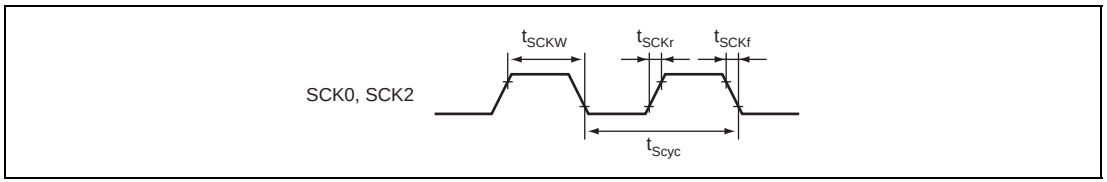



**Figure 22.15 SCK Clock Input Timing**

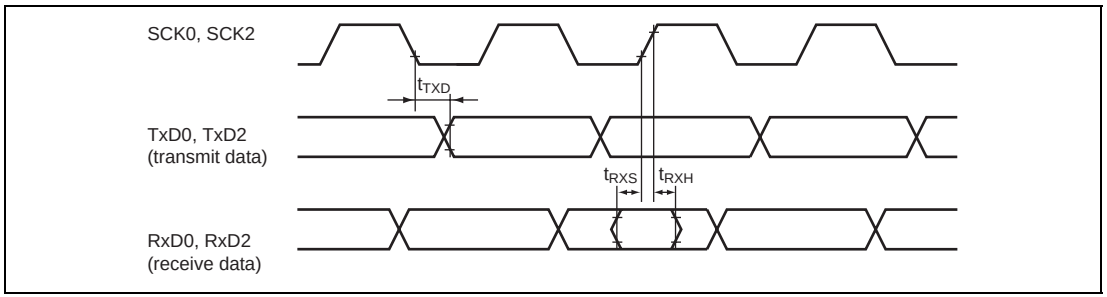



**Figure 22.16 SCI Input/Output Timing (Clock Synchronous Mode)**

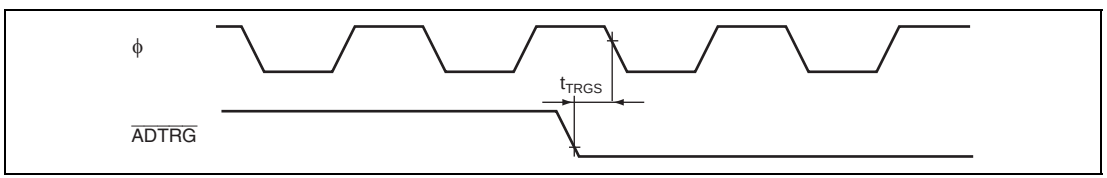



**Figure 22.17 A/D Converter External Trigger Input Timing**

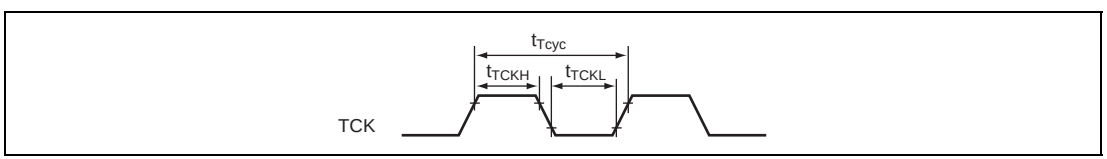



**Figure 22.18 Boundary Scan TCK Input Timing**

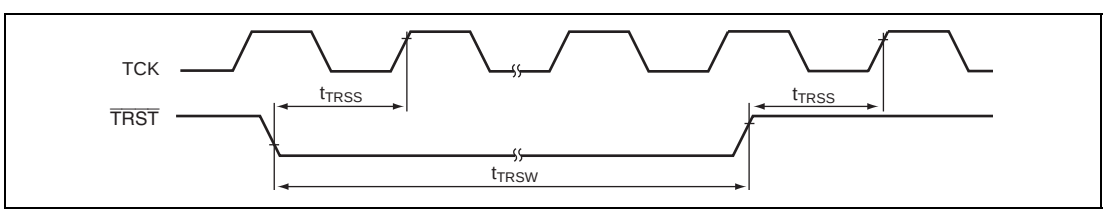



**Figure 22.19 Boundary Scan $\overline{TRST}$ Input Timing (At Reset Hold)**

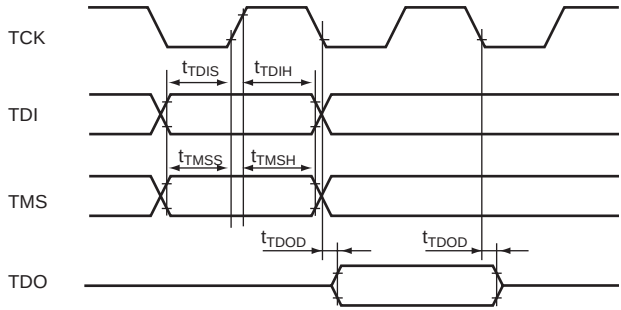



**Figure 22.20 Boundary Scan Data Transmission Timing**

## 22.5 USB Characteristics

Table 22.8 lists the USB characteristics (USD+ and USD- pins) when the on-chip USB transceiver is used.

**Table 22.8 USB Characteristics (USD+ and USD- pins) when On-Chip USB Transceiver Is Used**

Conditions: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, $V_{SS}$ = $PLLV_{SS}$ = $DrV_{SS}$ = 0 V, f = 16 MHz, 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| Item | | Symbol | Min. | Max. | Unit | Test Condition | |
|---|---|---|---|---|---|---|---|
| Input characteristics | Input high level voltage | $V_{IH}$ | 2.0 | — | V | | Figures 22.21, 22.22 |
| | Input low level voltage | $V_{IL}$ | — | 0.8 | V | | |
| | Differential input sense | $V_{DI}$ | 0.2 | — | V | \| (D+)-(D-)\| | |
| | Differential common mode range | $V_{CM}$ | 0.8 | 2.5 | V | | |
| Output characteristics | Output high level voltage | $V_{OH}$ | 2.8 | — | V | $I_{OH}$ = -200 μA | |
| | Output low level voltage | $V_{OL}$ | — | 0.3 | V | $I_{OL}$ = 2 mA | |
| | Crossover voltage | $V_{CRS}$ | 1.3 | 2.0 | V | | |
| | Rise time | $t_R$ | 4 | 20 | ns | | |
| | Fall time | $t_F$ | 4 | 20 | ns | | |
| | Rise time/fall time matching | $t_{RFM}$ | 90 | 111.11 | % | ($T_R/T_F$ ) | |
| | Output resistance | $Z_{DRV}$ | 28 | 44 | Ω | Including Rs = 24 Ω | |

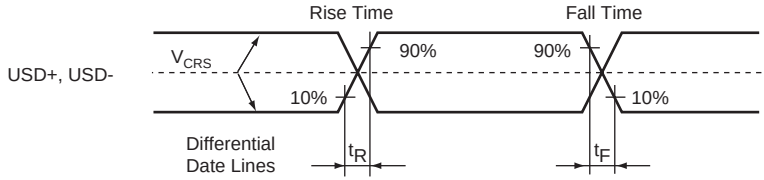



**Figure 22.21 Data Signal Timing**

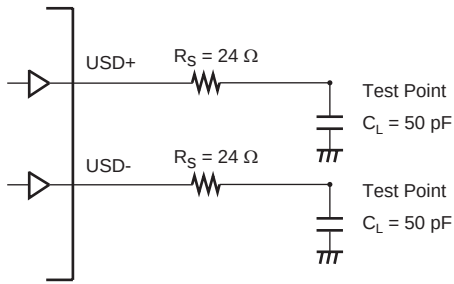



**Figure 22.22 Test Load Circuit**

## 22.6 A/D Conversion Characteristics

Table 22.9 lists the A/D conversion characteristics.

**Table 22.9 A/D Conversion Characteristics**

Condition A: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.4 V to 3.6 V, Vref = 2.4 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition B: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 2.7 V to 3.6 V, Vref = 2.7 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 16 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition C: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 6 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

Condition D: $V_{CC}$ = PLL $V_{CC}$ = Dr $V_{CC}$ = 3.0 V to 3.6 V, Vref = 3.0 V to $V_{CC}$, $V_{SS}$ = $PLLV_{SS}$ = Dr $V_{SS}$ = 0 V, f = 32.768 kHz, 16 MHz to 24 MHz, $T_a$ = –20°C to +75°C (regular specifications), $T_a$ = –40°C to +85°C (wide-range specifications)

| | Condition A | | | Condition B, C, D | | | |
|---|---|---|---|---|---|---|---|
| **Item** | **Min.** | **Typ.** | **Max.** | **Min.** | **Typ.** | **Max.** | **Unit** |
| Resolution | 10 | 10 | 10 | 10 | 10 | 10 | bits |
| Conversion time | 21.8 | — | — | 8.1 | — | — | µs |
| Analog input capacitance | — | — | 20 | — | — | 20 | pF |
| Permissible signal-source impedance | — | — | 5 | — | — | 5 | kΩ |
| Nonlinearity error | — | — | ±6.0 | — | — | ±6.0 | LSB |
| Offset error | — | — | ±4.0 | — | — | ±4.0 | LSB |
| Full-scale error | — | — | ±4.0 | — | — | ±4.0 | LSB |
| Quantization | — | — | ±0.5 | — | — | ±0.5 | LSB |
| Absolute accuracy | — | — | ±8.0 | — | — | ±6.0 | LSB |

## 22.7 Flash Memory Characteristics

Table 22.10 lists the flash memory characteristics.

**Table 22.10 Flash Memory Characteristics**

Conditions: $V_{CC}$ = $PLLV_{CC}$ = $DrV_{CC}$ = 3.0 V to 3.6 V, Vref = 2.7 V to $V_{CC}$,
$V_{SS}$ = $PLLV_{SS}$ = $DrV_{SS}$ = 0 V,
$T_a$ = –20 to +75°C (Programming/erasing operating temperature range)

| Item | | Symbol | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Programming time*1 *2 *4 | | $t_P$ | — | 10 | 200 | ms/128 bytes |
| Erase time*1 *3 *5 | | $t_E$ | — | 50 | 1000 | ms/block |
| Reprogramming count | | $N_{WEC}$ | 100*6 | 10000*7 | — | Times |
| Data retention time*8 | | $t_{DRP}$ | 10 | — | — | Years |
| Programming | Wait time after PSU1 bit setting*1 | y | 50 | 50 | — | µs |
| | Wait time after P1 bit setting*1 *4 | z0 | 28 | 30 | 32 | µs |
| | | z1 | 198 | 200 | 202 | µs |
| | | z2 | 8 | 10 | 12 | µs |
| | Wait time after P1 bit clear*1 | α | 5 | 5 | — | µs |
| | Wait time after PSU1 bit clear*1 | β | 5 | 5 | — | µs |
| | Wait time after PV1 bit setting*1 | γ | 4 | 4 | — | µs |
| | Wait time after H'FF dummy write*1 | ε | 2 | 2 | — | µs |
| | Wait time after PV1 bit clear*1 | η | 2 | 2 | — | µs |
| | Maximum programming count*1 *4 | N1 | — | — | 6*4 | Times |
| | | N2 | — | — | 994*4 | Times |
| Common | Wait time after SWE1 bit setting*1 | x | 1 | 1 | — | µs |
| | Wait time after SWE1 bit clear*1 | θ | 100 | 100 | — | µs |
| Erase | Wait time after ESU1 bit setting*1 | y | 100 | 100 | — | µs |
| | Wait time after E1 bit setting*1 *5 | z | 10 | 10 | 100 | ms |
| | Wait time after E1 bit clear*1 | α | 10 | 10 | — | µs |
| | Wait time after ESU1 bit clear*1 | β | 10 | 10 | — | µs |
| | Wait time after EV1 bit setting*1 | γ | 20 | 20 | — | µs |
| | Wait time after H'FF dummy write*1 | ε | 2 | 2 | — | µs |
| | Wait time after EV1 bit clear*1 | η | 4 | 4 | — | µs |
| | Maximum erase count*1 *5 | N | — | — | 100 | Times |

Notes: 1. Make each time setting in accordance with the program/program-verify flowchart or erase/erase-verify flowchart.

2. Programming time per 128 bytes (Shows the total period for which the P-bit in the flash memory control register (FLMCR1) is set. It does not include the programming verification time.)

3. Block erase time (Shows the total period for which the E1-bit FLMCR1 is set. It does not include the erase verification time.)
4. Maximum programming time value
   $t_p$ (max.) = Wait time after P1 bit set (z) × maximum programming count (N1 + N2)
   = (Z0 + Z2) × 6 + Z1 × 994
5. Maximum erasure time value
   $t_E$ (max.) = Wait time after E1 bit set (z) × maximum erasure count (N)
6. Minimum times that guarantee all characteristics after programming. (The guaranteed range is 1 to the minimum value.)
7. Reference value when the temperature is 25°C. (It is reference that reprogramming is normally enabled up to this value.)
8. Data hold characteristics when reprogramming is performed within the range of specifications including the minimum value.

## 22.8 Usage Note

**General Notice during Design for Printed Circuit Board:** Measures for radiation noise caused by the transient current in this LSI should be taken into consideration. The examples of the measures are shown below.

- To use a multilayer printed circuit board which includes layers for Vcc and GND.
- To mount by-pass capacitors (approximately 0.1 μF) between the Vcc and GND (Vss) pins, and the PLLVcc and PLLGND pins, of this LSI.

**Characteristics of F-ZTAT and Masked ROM Versions:** Though the F-ZTAT version and the masked ROM version satisfy electrical characteristics described in this manual, the actual value of electrical characteristics, operating margin, and noise margin may differ due to the differences of production process, on-chip ROM, and layout patterning.
When the system has been evaluated with the F-ZTAT version, the equivalent evaluation should be implemented to the masked ROM version when shifted to the masked ROM version.

# Appendix

## A. I/O Port States in Each Processing State

| Port Name / Pin Name | MCU Operating Mode | Power-on Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode or Watch Mode | Bus Right Release State | Program Execution State or Sleep Mode |
|---|---|---|---|---|---|---|---|
| P17 to P14 | 4 to 7 | T | keep | T | keep | keep | I/O port |
| P13/A23<br>P12/A22<br>P11/A21 | 7 | T | keep | T | keep | keep | I/O port |
| Address output selected by AEn bit | 4 to 6 | T | keep | T | [OPE=0]<br>T<br>[OPE=1]<br>keep | T | Address output |
| Port selection | 4 to 6 | T | keep | T | keep | keep | I/O port |
| P10/A20 | 7 | T | keep | T | keep | keep | I/O port |
| Address output selected by AEn bit | 4 and 5 | L | keep | T | [OPE=0]<br>T<br>[OPE=1]<br>keep | T | Address output |
| | 6 | T | | | | | |
| Port selection | 4 to 6 | T*[1] | keep | T | keep | keep | I/O port |
| Port 3 | 4 to 7 | T | keep | T | keep | keep | I/O port |
| Port 4 | 4 to 7 | T | T | T | T | T | Input port |
| P77 to P75*[3] | 7 | T | keep | T | keep | keep | I/O port |
| P74*[2] | 4 to 7 | T | keep | T | keep | keep | I/O port |
| P71/$\overline{CS5}$*[2] | 7 | T | keep | T | keep | keep | I/O port |
| P70/$\overline{CS4}$*[2] | 4 to 6 | T | keep | T | [DDR•OPE=0]<br>T<br>[DDR•OPE=1]<br>H | T | [DDR=0]<br>Input port<br>[DDR=1]<br>$\overline{CS5}$, $\overline{CS4}$ |
| Port 9 | 4 to 7 | T | T | T | [DAOEn=1]<br>keep<br>[DAOEn=0]<br>T | keep | Input port |

| Port Name / Pin Name | | MCU Operating Mode | Power-on Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode or Watch Mode | Bus Right Release State | Program Execution State or Sleep Mode |
|---|---|---|---|---|---|---|---|---|
| Port A | | 7 | T | keep | T | keep | keep | I/O port |
| Address output selected by AEn bit | | 4 and 5 | L | keep | T | [OPE=0]<br>T<br>[OPE=1]<br>keep | T | Address output |
| | | 6 | T | | | | | |
| Port selection | | 4 to 6 | T*[1] | keep | T | keep | keep | I/O port |
| Port B*[2] | | 7 | T | keep | T | keep | keep | I/O port |
| Address output selected by AEn bit | | 4 and 5 | L | keep | T | [OPE=0]<br>T<br>[OPE=1]<br>keep | T | Address output |
| | | 6 | T | | | | | |
| Port selection | | 4 to 6 | T*[1] | keep | T | keep | keep | I/O port |
| Port C*[2] | | 4 and5 | L | keep | T | [OPE=0]<br>T<br>[OPE=1]<br>keep | T | Address output |
| | | 6 | T | keep | T | [DDR•OPE=0]<br>T<br>[DDR•OPE=1]<br>keep | T | [DDR=0]<br>Input port<br>[DDR=1]<br>Address output |
| | | 7 | T | keep | T | keep | keep | I/O port |
| Port D*[2] | | 4 to 6 | T | T | T | T | T | Data bus |
| | | 7 | T | keep | T | keep | keep | I/O port |
| Port E | 8-bit bus | 4 to 6 | T | keep | T | keep | keep | I/O port |
| | 16-bit bus | 4 to 6 | T | T | T | T | T | Data bus |
| | | 7 | T | keep | T | keep | keep | I/O port |

| Port Name<br>Pin Name | MCU Operating Mode | Power-on Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode or Watch Mode | Bus Right Release State | Program Execution State or Sleep Mode |
|---|---|---|---|---|---|---|---|
| PF7/ϕ | 4 to 6 | Clock output | [DDR=0]<br>Input port<br>[DDR=1]<br>Clock output | T | [DDR=0]<br>Input port<br>[DDR=1]<br>H | [DDR=0]<br>Input port<br>[DDR=1]<br>Clock output | [DDR=0]<br>input port<br>[DDR=1]<br>Clock output |
| | 7 | T | keep | T | [DDR=0]<br>Input port<br>[DDR=1]<br>H | [DDR=0]<br>Input port<br>[DDR=1]<br>Clock output | [DDR=0]<br>Input port<br>[DDR=1]<br>Clock output |
| PF6/$\overline{AS}$*2<br>PF5/$\overline{RD}$*2<br>PF4/$\overline{HWR}$*2 | 4 to 6 | H | H | T | [OPE=0]<br>T<br>[OPE=1]<br>H | T | $\overline{AS}$, $\overline{RD}$, $\overline{HWR}$ |
| | 7 | T | keep | T | keep | keep | I/O port |
| PF3/$\overline{LWR}$ | 7 | T | keep | T | keep | keep | I/O port |
| 8-bit bus | 4 to 6 | (Mode 4)<br>H | keep | T | keep | keep | I/O port |
| 16-bit bus | 4 to 6 | (Modes 5 ,6)<br>T | H | T | [OPE=0]<br>T<br>[OPE=1]<br>H | T | $\overline{LWR}$ |
| PF2/$\overline{WAIT}$*2 | 4 to 6 | T | keep | T | [WAITE=0]<br>keep<br>[WAITE=1]<br>T | [WAITE=0]<br>keep<br>[WAITE=1]<br>T | [WAITE=0]<br>I/O port<br>[WAITE=1]<br>$\overline{WAIT}$ |
| | 7 | T | keep | T | keep | keep | I/O port |
| PF1/$\overline{BACK}$*2 | 4 to 6 | T | keep | T | [BRLE=0]<br>keep<br>[BRLE=1]<br>H | L | [BRLE=0]<br>I/O port<br>[BRLE=1]<br>$\overline{BACK}$ |
| | 7 | T | keep | T | keep | keep | I/O port |

| Port Name<br>Pin Name | MCU Operating Mode | Power-on Reset | Manual Reset | Hardware Standby Mode | Software Standby Mode or Watch Mode | Bus Right Release State | Program Execution State or Sleep Mode |
|---|---|---|---|---|---|---|---|
| PF0/$\overline{\text{BREQ}}$ | 4 to 6 | T | keep | T | [BRLE=0]<br>keep<br>[BRLE=1]<br>T | T | [BRLE=0]<br>I/O port<br>[BRLE=1]<br>$\overline{\text{BREQ}}$ |
| | 7 | T | keep | T | keep | keep | I/O port |
| PG4/$\overline{\text{CS0}}$*2 | 4 and 5 | H | keep | T | [DDR•OPE=0]<br>T<br>[DDR•OPE=1]<br>H | T | [DDR=0]<br>I/O port<br>[DDR=1]<br>$\overline{\text{CS0}}$ |
| | 6 | T | | | | | (When sleep mode)H |
| | 7 | T | keep | T | keep | keep | I/O port |
| PG3/$\overline{\text{CS1}}$*2<br>PG2/$\overline{\text{CS2}}$*2<br>PG1/$\overline{\text{CS3}}$ | 4 to 6 | T | keep | T | [DDR•OPE=0]<br>T<br>[DDR•OPE=1]<br>H | T | [DDR=0]<br>I/O port<br>[DDR=1]<br>$\overline{\text{CS1}}$ to $\overline{\text{CS3}}$ |
| | 7 | T | keep | T | keep | keep | I/O port |
| PG0*3 | 4 to 7 | T | keep | T | keep | keep | I/O port |

Legend:

H: High level

L: Low level

T: High impedance

keep: Input port level is high impedance, and output port level is retained.

DDR: Data direction register

OPE: Output port enable

WAITE: Wait port enable

BRLE: Bus release enable

Notes: 1. L (address input) in mode 4 or 5

2. Supported only by the H8S/2218 Group.

3. Supported only by the H8S/2212 Group.

# B. Product Model Lineup

| Product Class | | Part No. | Model Name | Marking | Package (code) |
|---|---|---|---|---|---|
| H8S/2218 Group | Flash memory Version | HD64F2218 | HD64F2218TF24 | F2218TF24 | 100-pin TQFP (TFP-100G, TFP-100GV) |
| | | | HD64F2218BR24 | 64F2218BR24 | 112-pin P-LFBGA (BP-112, BP-112V) |
| | | HD64F2218U | HD64F2218UTF24 | F2218UTF24 | 100-pin TQFP (TFP-100G, TFP-100GV) |
| | | | HD64F2218UBR24 | 64F2218UBR24 | 112-pin P-LFBGA (BP-112, BP-112V) |
| | | HD64F2218CU | HD64F2218CUTF24 | F2218CUTF24 | 100-pin TQFP (TFP-100GV) |
| | | | HD64F2218CUBR24 | 64F2218CUBR24 | 112-pin P-LFBGA (BP-112V) |
| | | HD64F2217CU | HD64F2217CUTF24 | F2217CUTF24 | 100-pin TQFP (TFP-100GV) |
| | | | HD64F2217CUBR24 | 64F2217CUBR24 | 112-pin P-LFBGA (BP-112V) |
| | Masked ROM Version | HD6432217 | HD6432217(***)TF | 2217(***)TF | 100-pin TQFP (TFP-100G, TFP-100GV) |
| | | | HD6432217(***)BR | 2217(***)BR | 112-pin P-LFBGA (BP-112, BP-112V) |
| H8S/2212 Group | Flash memory Version | HD64F2212 | HD64F2212FP24 | 2212FP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2212NP24 | F2212NP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD64F2212U | HD64F2212UFP24 | 2212UFP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2212UNP24 | F2212UNP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD64F2212CU | HD64F2212CUFP24 | 2212CUFP24 | 64-pin LQFP (FP-64EV) |
| | | | HD64F2212CUNP24 | F2212CUNP24 | 64-pin VQFN (TNP-64BV) |
| | | HD64F2211 | HD64F2211FP24 | 2211FP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2211NP24 | F2211NP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |

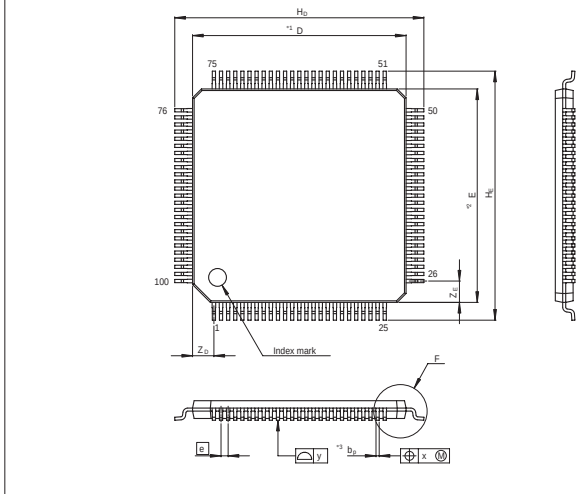
| Product Class | | Part No. | Model Name | Marking | Package (code) |
|---|---|---|---|---|---|
| H8S/2212 Group | Flash memory Version | HD64F2211U | HD64F2211UFP24 | 2211UFP24 | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD64F2211UNP24 | F2211UNP24 | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD64F2211CU | HD64F2211CUFP24 | 2211CUFP24 | 64-pin LQFP (FP-64EV) |
| | | | HD64F2211CUNP24 | F2211CUNP24 | 64-pin VQFN (TNP-64BV) |
| | | HD64F2210CU | HD64F2210CUFP24 | 2210CUFP24 | 64-pin LQFP (FP-64EV) |
| | | | HD64F2210CUNP24 | F2210CUNP24 | 64-pin VQFN (TNP-64BV) |
| | Masked ROM Version | HD6432211 | HD6432211(***)FP | 2211(***)FP | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD6432211(***)NP | 2211(***)NP | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD6432210 | HD6432210(***)FP | 2210(***)FP | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD6432210(***)NP | 2210(***)NP | 64-pin VQFN (TNP-64B, TNP-64BV) |
| | | HD6432210S | HD6432210S(***)FP | 2210S(***)FP | 64-pin LQFP (FP-64E, FP-64EV) |
| | | | HD6432210S(***)NP | 2210S(***)NP | 64-pin VQFN (TNP-64B, TNP-64BV) |

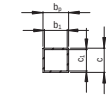
Legend:
***: ROM code

# C. Package Dimensions

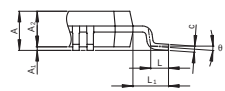
The package dimension that is shown in the Renesas Semiconductor Package Data Book has priority.

| JEITA Package Code | RENESAS Code | Previous Code | MASS[Typ.] |
|---|---|---|---|
| P-TQFP100-12x12-0.40 | PTQP0100LC-A | TFP-100G/TFP-100GV | 0.4g |




Terminal cross section



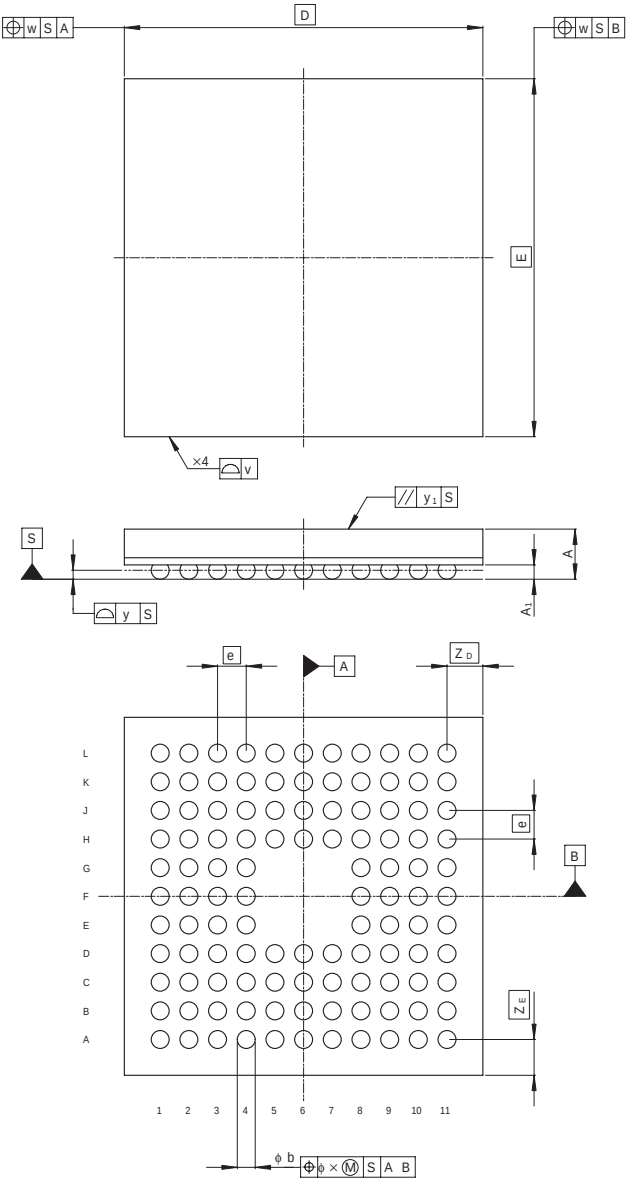

Detail F

NOTE)
1. DIMENSIONS"*1"AND"*2" DO NOT INCLUDE MOLD FLASH
2. DIMENSION"*3"DOES NOT INCLUDE TRIM OFFSET.

| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | — | 12 | — |
| E | — | 12 | — |
| $A_2$ | — | 1.00 | — |
| $H_D$ | 13.8 | 14.0 | 14.2 |
| $H_E$ | 13.8 | 14.0 | 14.2 |
| A | — | — | 1.20 |
| $A_1$ | 0.00 | 0.10 | 0.20 |
| $b_p$ | 0.13 | 0.18 | 0.23 |
| $b_1$ | — | 0.16 | — |
| c | 0.12 | 0.17 | 0.22 |
| $c_1$ | — | 0.15 | — |
| θ | 0° | — | 8° |
| e | — | 0.4 | — |
| x | — | — | 0.07 |
| y | — | — | 0.10 |
| $Z_D$ | — | 1.2 | — |
| $Z_E$ | — | 1.2 | — |
| L | 0.4 | 0.5 | 0.6 |
| $L_1$ | — | 1.0 | — |

**Figure C.1 TFP-100G and TFP-100GV Package Dimensions**

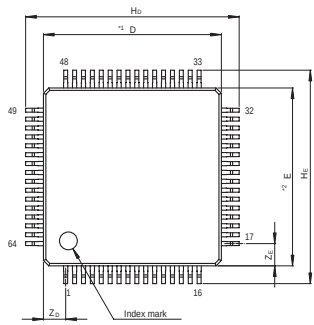
| JEITA Package Code | RENESAS Code | Previous Code | MASS[Typ.] |
|---|---|---|---|
| P-LFBGA112-10x10-0.80 | PLBG0112GA-A | BP-112/BP-112V | 0.3g |

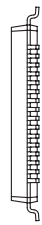


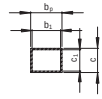

| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | — | 10.00 | — |
| E | — | 10.00 | — |
| v | — | — | 0.15 |
| w | — | — | 0.20 |
| A | — | — | 1.40 |
| $A_1$ | 0.35 | 0.40 | 0.45 |
| e | — | 0.80 | — |
| b | 0.45 | 0.50 | 0.55 |
| x | — | — | 0.08 |
| y | — | — | 0.10 |
| $y_1$ | — | — | 0.2 |
| $S_D$ | — | — | — |
| $S_E$ | — | — | — |
| $Z_D$ | — | 1.00 | — |
| $Z_E$ | — | 1.00 | — |

**Figure C.2 BP-112 and BP-112V Package Dimensions**

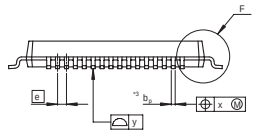
| JEITA Package Code | RENESAS Code | Previous Code | MASS[Typ.] |
|---|---|---|---|
| P-LQFP64-10x10-0.50 | PLQP0064KC-A | FP-64E/FP-64EV | 0.4g |

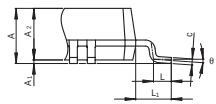





Terminal cross section



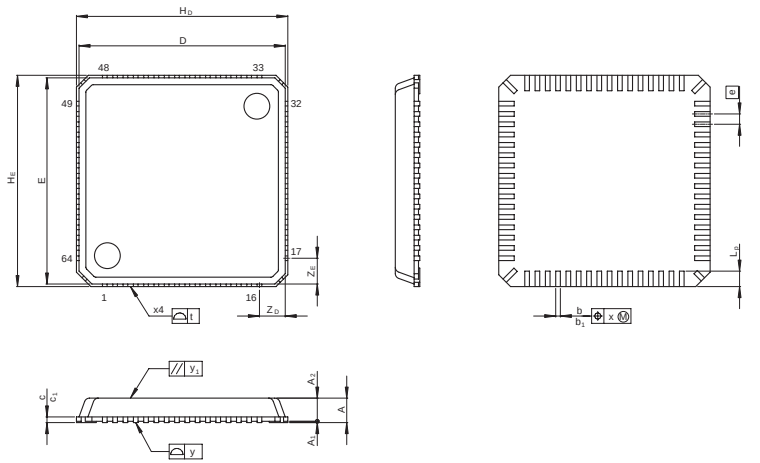

Detail F

NOTE)
1. DIMENSIONS"*1"AND"*2"
DO NOT INCLUDE MOLD FLASH
2. DIMENSION"*3"DOES NOT
INCLUDE TRIM OFFSET.

| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | — | 10 | — |
| E | — | 10 | — |
| $A_2$ | — | 1.45 | — |
| $H_D$ | 11.8 | 12.0 | 12.2 |
| $H_E$ | 11.8 | 12.0 | 12.2 |
| A | — | — | 1.70 |
| $A_1$ | 0.00 | 0.10 | 0.20 |
| $b_p$ | 0.17 | 0.22 | 0.27 |
| $b_1$ | — | 0.20 | — |
| c | 0.12 | 0.17 | 0.22 |
| $c_1$ | — | 0.15 | — |
| θ | 0° | — | 8° |
| e | — | 0.5 | — |
| x | — | — | 0.08 |
| y | — | — | 0.10 |
| $Z_D$ | — | 1.25 | — |
| $Z_E$ | — | 1.25 | — |
| L | 0.3 | 0.5 | 0.7 |
| $L_1$ | — | 1.0 | — |

**Figure C.3 FP-64E and FP-64EV Package Dimensions**

| JEITA Package Code | RENESAS Code | Previous Code | MASS[Typ.] |
|---|---|---|---|
| P-VQFN64-8x8-0.40 | PVQN0064LB-A | TNP-64B/TNP-64BV | 0.12g |




| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | — | 8.0 | — |
| E | — | 8.0 | — |
| $A_2$ | — | 0.89 | — |
| A | — | — | 0.95 |
| $A_1$ | 0.005 | 0.02 | 0.04 |
| b | 0.13 | 0.18 | 0.23 |
| $b_1$ | — | 0.16 | — |
| e | — | 0.4 | — |
| $L_p$ | 0.50 | 0.60 | 0.70 |
| x | — | — | 0.05 |
| y | — | — | 0.05 |
| $y_1$ | — | — | 0.2 |
| t | — | — | 0.2 |
| $H_D$ | — | 8.2 | — |
| $H_E$ | — | 8.2 | — |
| $Z_D$ | — | 1.0 | — |
| $Z_E$ | — | 1.0 | — |
| c | 0.17 | 0.22 | 0.25 |
| $c_1$ | — | 0.20 | — |

**Figure C.4 TNP-64B and TNP-64BV Package Dimensions**

# Index

16-Bit Timer Pulse Unit ..... 273

A/D Conversion Time ..... 544
A/D Converter ..... 535
A/D Converter Activation ..... 323
Absolute Address ..... 59
Address Space ..... 38
Addressing Mode ..... 58
ADI ..... 546
Advanced Mode ..... 36
Arithmetic Operations Instructions ..... 49
Asynchronous Mode ..... 403

Bcc ..... 46, 54
Bit Manipulation Instructions ..... 52
Bit rate ..... 395
Block Data Transfer Instruction ..... 56
Boot Mode ..... 567
Boundary Scan ..... 449
Branch Instructions ..... 54
Break ..... 441
Buffer Operation ..... 309
Bulk-In Transfer ..... 513
Bulk-Out Transfer ..... 515
Bus Arbitration ..... 155
Bus cycle ..... 133

Clock Pulse Generator ..... 595
Condition Field ..... 57
Condition-Code Register ..... 42
Control Transfer ..... 506
CPU Operating Modes ..... 34

Data reading procedure ..... 359
Data Transfer Instructions ..... 48
DMA Transfer Specifications ..... 520

Effective Address ..... 58, 62
Effective Address Extension ..... 57
Emulation ..... 577
Erase/Erase-Verify ..... 581
Erasing units ..... 558, 559
Error Protection ..... 583
Exception Handling ..... 81
Extended Control Register (EXR) ..... 41
External Trigger ..... 545

Flash Memory ..... 553
Framing error ..... 410
Free-running count operation ..... 302

General Registers ..... 40

Hardware Protection ..... 583

Immediate ..... 60
Input Capture Function ..... 305
Instruction Set ..... 45
Internal bus masters ..... 115
Interrupt Control Mode ..... 103
Interrupt Controller ..... 91
Interrupt Exception Handling Vector Table ..... 101
Interrupt Mask Bit ..... 42
Interrupt mask level ..... 41
Interrupt priority register ..... 91
Interrupt-In Transfer ..... 512
Interrupts ..... 87
Interval Timer Mode ..... 344

Logic Operations Instructions ..... 50

Mark State ..... 442
Masked ROM ..... 593
Memory cycle ..... 131
Memory Indirect ..... 61

Memory Map ..... 77

NMI Interrupt ..... 99
Normal Mode ..... 34

On-Board Programming ..... 567
Operating Mode Selection ..... 71
Operation Field ..... 57
Overflow ..... 345
Overrun error ..... 410

Parity error ..... 410
Periodic count operation ..... 302
Phase Counting Mode ..... 317
PLL Circuit ..... 604
Processing of USB Standard Commands and Class/Vendor Commands ..... 516
Program Counter ..... 41
Program/Program-Verify ..... 579
Program-Counter Relative ..... 60
Programmer Mode ..... 585
Programming units ..... 558, 559
Programming/Erasing in User Program Mode ..... 576
PWM Modes ..... 313

Realtime Clock (RTC) ..... 349
Register
ABWCR ..... 118, 635, 643, 651
ADCR ..... 540, 638, 647, 654
ADCSR ..... 538, 638, 647, 654
ADDR ..... 538, 638, 647, 654
ASTCR ..... 119, 635, 643, 651
BCRH ..... 124, 635, 643, 651
BCRL ..... 125, 635, 643, 651
BRR ..... 395, 638, 646, 654
BSCANR ..... 454
BYPASS ..... 454
DMACR ..... 163, 637, 646, 653
EBR1 ..... 564, 638, 647, 654
EBR2 ..... 564, 638, 647, 654
ETCR ..... 162, 635, 643, 652
FLMCR1 ..... 562, 638, 647, 654
FLMCR2 ..... 563, 638, 647, 654
IDCODE ..... 454
IER ..... 95, 634, 641, 650
INSTR ..... 452
IOAR ..... 161, 635, 643, 652
IPR ..... 94, 635, 642, 651
ISCR ..... 96, 634, 641, 650
ISR ..... 98, 634, 641, 650
LPWRCR ..... 597, 634, 641, 650
MAR ..... 161, 635, 643, 652
MDCR ..... 72, 634, 641, 650
MSTPCR ..... 614, 634, 641, 650
P1DDR ..... 216, 634, 642, 650
P1DR ..... 217, 636, 644, 652
P3DDR ..... 223, 634, 642, 650
P3DR ..... 224, 636, 644, 652
P3ODR ..... 225, 634, 642, 651
P7DDR ..... 228, 634, 642, 650
P7DR ..... 229, 636, 644, 652
PADDR ..... 233, 634, 642, 650
PADR ..... 234, 636, 644, 652
PAODR ..... 235, 634, 642, 651
PAPCR ..... 235, 634, 642, 651
PBDDR ..... 239, 634, 642, 650
PBDR ..... 240, 636, 644, 652
PBPCR ..... 241, 634, 642, 651
PCDDR ..... 245, 634, 642, 650
PCDR ..... 246, 636, 644, 652
PCPCR ..... 247, 634, 642, 651
PDDDR ..... 250, 634, 642, 651
PDDR ..... 251, 636, 644, 652
PDPCR ..... 252, 634, 642, 651
PEDDR ..... 255, 634, 642, 651
PEDR ..... 256, 636, 644, 652
PEPCR ..... 257, 634, 642, 651
PFCR ..... 126, 634, 641, 650
PFDDR ..... 262, 634, 642, 651
PFDR ..... 263, 636, 644, 652
PGDDR ..... 268, 634, 642, 651

PGDR ..... 269, 636, 644, 652
PORT1 ..... 217, 639, 648, 655
PORT3 ..... 224, 639, 648, 655
PORT4 ..... 227, 639, 648, 655
PORT7 ..... 230, 639, 648, 655
PORT9 ..... 232, 639, 648, 655
PORTA ..... 234, 639, 648, 655
PORTB ..... 240, 639, 648, 655
PORTC ..... 246, 639, 648, 655
PORTD ..... 251, 639, 648, 655
PORTE ..... 256, 639, 648, 655
PORTF ..... 263, 639, 648, 655
PORTG ..... 269, 639, 648, 655
RAMER ..... 565, 635, 643, 651
RDR ..... 368, 638, 646, 654
RHRDR ..... 352
RMINDR ..... 351, 637, 646, 653
RSECDR ..... 637, 646, 653
RSTCSR ..... 342, 637, 646, 653
RTCCR1 ..... 354, 637, 646, 653
RTCCR2 ..... 355, 637, 646, 653
RTCCSR ..... 356, 637, 646, 653
RWKDR ..... 353, 637, 646, 653
SBYCR ..... 611, 634, 641, 650
SCKCR ..... 596, 634, 641, 650
SCMR ..... 383, 638, 646, 654
SCR ..... 373, 638, 646, 654
SCRX ..... 566, 633, 641, 650
SEMRA_0 ..... 384, 634, 641, 650
SEMRB_0 ..... 386, 634, 641, 650
SMR ..... 369, 638, 646, 654
SSR ..... 377, 638, 646, 654
SYSCR ..... 72, 634, 641, 650
TCNT ..... 297, 340, 636, 637, 644, 646, 652, 653
TCR ..... 279, 636, 644, 652
TCSR ..... 340, 637, 646, 653
TDR ..... 368, 638, 646, 654
TGR ..... 297, 636, 644, 652
TIER ..... 293, 636, 644, 652
TIOR ..... 284, 636, 644, 652
TMDR ..... 282, 636, 644, 652
TSR ..... 294, 636, 644, 652
TSTR ..... 297, 635, 642, 651
TSYR ..... 298, 635, 642, 651
UCTLR ..... 468, 632, 640, 649
UCVR ..... 488, 633, 641, 649
UDMAR ..... 471, 632, 640, 649
UDRR ..... 472, 632, 640, 649
UDSR ..... 487, 633, 641, 649
UEDR0i ..... 477, 632, 640, 649
UEDR0o ..... 478, 632, 640, 649
UEDR0s ..... 477, 632, 640, 649
UESTL0 ..... 476, 632, 640, 649
UESTL1 ..... 477, 632, 640, 649
UESZ0o ..... 479, 633, 640, 649
UESZ2 ..... 479, 633, 640, 649
UFCLR0 ..... 475, 632, 640, 649
UIER0 ..... 484, 633, 640, 649
UIER1 ..... 485, 633, 640, 649
UIER3 ..... 485, 633, 640, 649
UIFR0 ..... 480, 633, 640, 649
UIFR1 ..... 482, 633, 640, 649
UIFR3 ..... 483, 633, 640, 649
UISR0 ..... 486, 633, 640, 649
UISR1 ..... 486, 633, 641, 649
UISR3 ..... 487, 633, 641, 649
UTRG0 ..... 473, 632, 640, 649
UTSTR0 ..... 489, 633, 641, 650
UTSTR1 ..... 490, 633, 641, 650
WCRH ..... 120, 635, 643, 651
WCRL ..... 120, 635, 643, 651
Register Direct ..... 58
Register Field ..... 57
Register Indirect ..... 58
Register Indirect with Displacement ..... 59
Register Indirect with Post-Increment ..... 59
Register indirect with pre-decrement ..... 59
Reset ..... 83
Reset exception handling ..... 84

Scan Mode ..... 543

Serial Communication Interface..... 363
Shift Instructions..... 51
Single Mode..... 542
Software Protection..... 583
Stack pointer (SP)..... 40
Stack Status..... 89
Stall Operations..... 517
Suspend and Resume..... 502
Synchronous Operation..... 307
System Control Instruction..... 55

TCI0V..... 322
TCI1U..... 322
TCI1V..... 322
TCI2U..... 322
TCI2V..... 322
TGI0A..... 322
TGI0B..... 322
TGI0C..... 322
TGI0D..... 322
TGI1A..... 322
TGI1B..... 322
TGI2A..... 322
TGI2B..... 322
Toggle output..... 304
Trace Bit..... 41
Traces..... 87
Trap Instruction..... 88
TRAPA..... 60, 88

Universal Serial Bus..... 465
USB Cable Connection/Disconnection..... 498

Watchdog timer..... 339
Waveform Output by Compare Match..... 303
WOVI..... 345

**Renesas 16-Bit Single-Chip Microcomputer**
**Hardware Manual**
**H8S/2218 Group, H8S/2212 Group**


| | |
|---|---|
| Publication Date: | 1st Edition, February 2003 |
| | Rev.7.00, December 24, 2008 |
| Published by: | Sales Strategic Planning Div. |
| | Renesas Technology Corp. |
| Edited by: | Customer Support Department |
| | Global Strategic Communication Div. |
| | Renesas Solutions Corp. |


© 2008. Renesas Technology Corp., All rights reserved. Printed in Japan.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

## RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

**Renesas Technology America, Inc.**
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

**Renesas Technology Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

**Renesas Technology (Shanghai) Co., Ltd.**
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7858/7898

**Renesas Technology Hong Kong Ltd.**
7th Floor, North Tower, World Finance Centre, Harbour City, Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2377-3473

**Renesas Technology Taiwan Co., Ltd.**
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 3518-3399

**Renesas Technology Singapore Pte. Ltd.**
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

**Renesas Technology Korea Co., Ltd.**
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

**Renesas Technology Malaysia Sdn. Bhd**
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

# H8S/2218 Group, H8S/2212 Group
# Hardware Manual

Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan