REJ09B0015-0200Z

RENESAS

32

# 32171 Group

## User's Manual

RENESAS 32-BIT RISC SINGLE-CHIP MICROCOMPUTER

M32R FAMILY / M32R/ECU SERIES

User's Manual

Before using this material, please visit our website to confirm that this is the most current document available.

Rev. 2.00
Revision date: Sep 19, 2003

RenesasTechnology
www.renesas.com

## Keep safety first in your circuit designs!

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

## Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
- Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.

  The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.

  Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.

  Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Renesas Technology Corporation for further details on these materials or t he products contained therein.

## REVISION HISTORY

## 32171 Group User’s Manual

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 0.1 | Apr 8, 2000 | – | First edition issued |
| 1.0 | Nov 1, 2002 | all | Explanation of the M32171F2 added |
| | | all | Designation of M32R/E changed to M32R/ECU |
| | | P1-6 | Description in Section 1.1.6, Built-in Full-CAN Function, corrected<br>Incorrect: Compliant with CAN Specification V2.0B<br>Correct: Compliant with CAN Specification V2.0B active |
| | | P1-7 | M32171F2 added to the internal flash memory in Figure 1.2.1 |
| | | P1-8 | M32171F2 added to the internal flash memory in Table 1.2.2 |
| | | P1-10 | Table 1.2.4, List of Type Name added |
| | | P1-11 | Note 1 in Figure 1.3.1 corrected<br>Incorrect: Operates with a 5 V power supply<br>Correct: Operates with a 3.3 V or 5 V power supply |
| | | P1-12 | Functional description of pin names VCCE and OSC-VCC in Table 1.3.1corrected |
| | | | Explanation of $\overline{WR}$ added to the functional description of clock in Table 1.3.1 |
| | | P1-13 | Explanation of the A-D converter in Table 1.3.1 corrected |
| | | P1-17 | Figure 1.4.1 corrected |
| | | P3-5 | Figure 3.1.3, "M32171F2 address space," added |
| | | P3-6 | Table 3.2.1 corrected |
| | | | Note 1 in Table 3.2.1 corrected |
| | | P3-7 | Figure 3.2.3 "M32171F2 operation mode and internal ROM/external extended areas," added |
| | | P3-8 | M32171F2 added to Table 3.3.1 |
| | | P4-25 | Section 4.13, "Precautions on EIT," added |
| | | P5-13 | Relevant names of causes added to Table 5.4.1 |
| | | P5-17 | Relevant names of causes added to Table 5.5.1 |
| | | P5-19 | Explanation added to (4) "Enabling multiple interrupts" in Section 5.5.2, "Processing of Internal Peripheral I/O Interrupts by Handler" |
| | | P6-2 | Description in Section 6.1, "Outline of the Internal Memory," corrected |
| | | | Precautions added to Table 6.2.1 |
| | | P6-3 | M32171F2 added to Table 6.3.1 |
| | | P6-5 | Precautions added |
| | | P6-7 | Precautions (Note 2) added |
| | | P6-8 | Precautions added |
| | | P6-13 | Figure 6.4.4, “FCNT4 Register Usage Example 2,” added |
| | | P6-22 | Table 6.5.1 corrected |
| | | P6-25 | Precautions (Note 2, 3, 4) added to Table 6.5.2 |
| | | | |

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.0 | Nov 1, 2002 | P6-27 | Table 6.5.5, “M32171F2’s relevant block and specificaion address,” added |
| | | P6-30 | Table 6.5.9, “Block configuration of M32171F2 flash memory,” added |
| | | P6-38 | Figure 6.5.15, Figure 6.5.16 and Figure 6.5.17 corrected |
| | | P6-40 | (3) M32171F2 added to Section 6.5.4, “Flash Programming Time (Reference Value)” |
| | | P6-43 | Precautios (Notes 2, 3, 4) added |
| | | P6-46 | Figure 6.7.6, “Virtual-flash emulation area of the M32171F2 divided in 8 Kbyte units,” added |
| | | | Figure 6.7.7, “Virtual-flash emulation area of the M32171F2 divided in 4 Kbyte units,” added |
| | | P6-47,<br>P6-48 | Incorrect register names in Figures 6.7.8 through 6.7.11 corrected<br>Incorrect: LBAKNKAD<br>Correct: LBANKAD |
| | | P6-49 | Figure 6.7.12, “Virtual-flash bank register setup values for the M32171F2 when divided in 8 Kbyte units,” added |
| | | | Figure 6.7.13, “Virtual-flash bank register setup values for the M32171F2 when divided in 4 Kbyte units,” added |
| | | P6-55 | Section 6.9, “Internal Flash Memory Protect Functions,” added |
| | | P6-56 | Explanation in Section 6.10, ”Precautions to Be Taken when Reprogramming Flash Memory,” changed |
| | | P7-3 | Table 7.3.1 corrected |
| | | P7-4 to P7-7 | Tables 7.3.2 to 7.3.5, “ Pin Status When Reset,” added or corrected |
| | | P8-4 | Table 8.2.1 corrected |
| | | | Precautions in Table 8.2.1 corrected |
| | | P8-22 to P8-25 | Figures 8.4.1 to 8.4.4 corrected |
| | | P8-26 | Section 8.5, “Precautions on Input/output Ports,” added |
| | | P9-4 | Figure 9.1.2, “Causes of DMAC Requests Connection Diagram,” added |
| | | P10-1 to P10-142 | Chapter 10 overall, designation of the prescaler unified to PRS |
| | | P10-4 | Port numbers added to Figure 10.1.1 |
| | | P10-5 | Port numbers added to Figure 10.1.2 |
| | | P10-12 | Port numbers added to Figure 10.2.2 |
| | | P10-31 | Figure 10.2.5 changed |
| | | P10-47 | Port numbers added to Figure 10.3.1 |
| | | P10-55 | Port number added to Figure 10.3.5 |
| | | P10-66 | Figure 10.3.8 corrected |

| REVISION HISTORY | 32171 Group User's Manual |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.0 | Nov 1, 2002 | P10-84 | Port numbers added to Figure 10.4.1 |
| | | P10-93 | Port numbers added to Figure 10.4.5 |
| | | P10-96 | Port numbers added to Figure 10.4.6 |
| | | P10-124 | Port numbers added to Figure 10.5.1 |
| | | P10-130 | Figure 10.5.3 corrected |
| | | P10-133 | Port numbers added to Figure 10.6.1 |
| | | P10-141 | Note 1 in Figure 10.6.3 corrected |
| | | P11-3 | Table 11.1.1 corrected |
| | | | Precautions in Table 11.1.1 corrected |
| | | P11-4 | Register names in Figure 11.1.1 corrected |
| | | P11-35 | Method for calculating the conversion time during A-D conversion mode and that for conversion time during comparate mode explained separately |
| | | | Table 11.3.1 and precausions corrected |
| | | | Figure 11.3.4, "Conceptual Diagram of Conversion Time in Comparate Mode," added |
| | | | Table 11.3.2, "Conversion Clock Cycles in Comparate Mode," added |
| | | P11-37 to P11-38 | Explanation in Section 11.3.5, "Definition of the A-D Conversion Accuracy," changed |
| | | P11-40 to P11-42 | A section "Regarding the analog input pins" added to Section 11.4, "Precautions on Using A-D Converters" |
| | | P12-12 | Figure 12.2.4 corrected |
| | | P12-24 | Description of the last line in Section 12.2.8, "SIO Baud Rate Register," corrected<br>Incorrect: 7 or less<br>Correct: greater than 7 |
| | | P12-58 | Figure 12.7.5, "Detecting the Start Bit, added |
| | | | Figure 12.7.6, "Example of an Invalid Start Bit (Not Received)," added |
| | | | Figure 12.7.7, "Delay when Receiving," added |
| | | P13-2 | Description in Section 13.1, "Outline of the  CAN Module," corrected<br>Incorrect: Compliant with CAN (Controller Area Network) Specification V2.0B<br>Correct: Compliant with CAN (Controller Area Network) Specification V2.0B active |
| | | | Protocol explanation in Table 13.1.1 corrected<br>Incorrect: CAN Specification V2.0B<br>Correct:  CAN Specification V2.0B active |
| | | | Explanation of acceptance filters in Table 13.1.1 changed |
| | | | Precautions in Table 13.1.1 changed |
| | | P13-3 | Figure 13.1.1 corrected |
| | | P13-19 | Table 13.2.2, "Example for Setting Bit Timing when CPU Clock: 32 MHz," added |
| | | P13-20 | Note 3 added |
| | | | |

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.0 | Nov 1, 2002 | P13-28 | Figure 13.2.5 corrected |
| | | P13-29 | Figure 13.2.6 corrected |
| | | P13-30 | Figure 13.2.7 corrected |
| | | P13-35 | Figure 13.2.8, “Relationship between Mask Registers and the Controlled Slots,” added |
| | | | Figure 13.2.9, “ Operation of the Acceptance Filter,” added |
| | | P13-61 | Explanation in (2) Confirming that transmission is idle corrected |
| | | P13-64 | Figure 13.5.2 corrected |
| | | P13-65 | Explanation in (2) Confirming that reception is idle corrected |
| | | P13-68 | Figure 13.6.2 corrected |
| | | P13-71 | Explanation in (2) Confirming that transmission is idle corrected |
| | | P13-75 | Figure 13.7.2 corrected |
| | | P13-78 | Explanation in (2) Confirming that reception is idle corrected |
| | | P13-82 | Figure 13.8.2 corrected |
| | | P15-6 to P15-11 | Figures 15.2.1 to 15.2.6 corrected (Address signals A12 to A30 and chip select signals $\overline{CS0}$, $\overline{CS1}$ separately shown) |
| | | P15-12 to P15-13 | Figures 15.3.1 and 15.3.2 corrected (Address signals A12 to A30 and chip select signals $\overline{CS0}$, $\overline{CS1}$ separately shown) |
| | | P16-6 to P16-19 | Figures 16.3.1 to 16.3.14 corrected (Address signals A12 to A30 and chip select signals $\overline{CS0}$, $\overline{CS1}$ separately shown) |
| | | P18-2 | Precautions added to Figure 18.1.1 |
| | | P19-7 | Figure 19.4.2 corrected |
| | | P19-14 | Precautions added to Section 19.5, “Boundary Scan Description Language” |
| | | P19-14 | BSDL description language for the 32171 (Figures 19.5.1 to 19.5.14) deleted |
| | | P19-15 | Precautions added to Figure 19.6.1 |
| | | P19-16 | Precautions added to Section 19.7, “Processing Pins when Not Using JTAG” |
| | | | Figure 19.7.1, “Processing Pins when Not Using JTAG,” added |
| | | P20-1 to P20-16 | In Chapter 20, explanation of power supply turn-on/turn-off sequences during VCCE=3.3V added |
| | | | Chapter 20 overall, designations of “5V system” and “3.3V system” changed to “external I/O” and “internal,” respectively |
| | | P20-12 | Figure 20.3.6 corrected |
| | | P20-13 | Figure 20.3.8, “CPU Reset State” deleted |
| | | P20-15 | Figure 20.3.12, “SRAM Data Backup State” deleted |
| | | P21-3, P21-4 | Recommended operating conditions corrected (minimum value of analog reference voltage added) |
| | | P21-5 | (1) Electrical characteristics when f(XIN) = 10 MHz corrected |
| | | | |

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.0 | Nov 1, 2002 | P21-7 | (3) Electrical characteristics when f(XIN) = 8 MHz corrected |
| | | P21-10 | Section 21.1.4, "A/D Conversion Characteristics," corrected |
| | | P21-11 to P21-18 | Section 21.2, "Electrical Characteristics (when VCCE = 3.3V)," added |
| | | P21-19 | Explanation in Section 21.3.1, "Timing Requirements," corrected |
| | | P21-22 | (9) Table of rated RTD timings corrected |
| | | P21-32 | Figure 21.3.12 corrected |
| | | Appendix 3 | Appendix 3, "Processing Unused Pins," added |
| | | Appendix 4 | Appendix 4, "Summary of Precautions," added |
| | | | "Precautions about Noise" in Appendix 3 moved to Appendix 4, "Summary of Precautions" |
| 2.00 | Sep 19, 2003 | all | The word "Mitsubishi" deleted or replaced by "Renesas" |
| | | P1-4 | Figure 1.1.1 and Table 1.1.1 newly added |
| | | P2-14 | Section 2.7, "Precautions on CPU" added |
| | | P3-8 | Addresses in the third line of Section 3.3 corrected<br>Incorrect: H'0000 0000 to H'0003 FFFF<br>Correct: H'0000 0000 to H'003F FFFF |
| | | P3-9 | Addresses in Section 3.4.1 corrected<br>Incorrect: H'0080 4000 through H'0080 3FFF<br>Correct: H'0080 4000 through H'0080 7FFF |
| | | P4-20 | Designation in (2), "Updating SM, IE and C bits" in the Section [EIT processing] corrected<br>Incorrect: SM ← 0<br>Correct: SM ← Unchanged |
| | | P5-end | Section 5.2 and 5.3 placed in reversed |
| | | | Title of Section 5.3 (former 5.2) changed<br>Before: Interrupt Sources of Internal Peripheral<br>After: Interrupt Request Sources in Internal Peripheral |
| | | P5-2 | Description in the fourth line of Section 5.1 corrected<br>Incorrect: total of 31<br>Correct: total of 22 |
| | | | Note added to Table 5.1.1 |
| | | P5-3 | Figure 5.1.1 altered |
| | | P5-5,<br>P5-6 | Note (former CAUTION) altered |
| | | P5-7 | Description in Section 5.2.3 altered |
| | | P5-9 | Description in (1), "IREQ (Interrupt Request) bit (D3 or D11)", altered |
| | | P5-10 | Figures 5.2.2, "Configuration of the Interrupt Control Register (Edge-recognized Type)", and 5.2.3, "Configuration of the Interrupt Control Register (Level-recognized Type)", changed |

| REVISION HISTORY | 32171 Group User's Manual |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 2.00 | Sep 19, 2003 | P5-17 | Table 5.5.1 corrected |
| | | P5-19, P5-20 | Description in (2) to (4), Section 5.5.2 changed |
| | | P5-21 | Figure 5.5.2 changed |
| | | P6-43 | Note 3 for Section 6.7.1 corrected |
| | | P6-44 | Notes in Figures 6.7.2 and 6.7.3 corrected |
| | | P6-50 | Figure, "Virtual-Flash Emulation Mode to Normal Mode Return Sequence" deleted |
| | | P7-1 to P7-7 | Chapter 7 overall,<br>The phrase "reset release" changed to " exiting reset" |
| | | P7-3 | Registers R0-R15 added to Table 7.3.1 |
| | | P10-end | Sections 10.7 to 10.9 deleted |
| | | P10-19, P10-20 | Note added |
| | | P10-49 | Figure 10.3.2, "Count Clock Dependent Delay", newly added |
| | | P10-72 | Figure (former 10.3.13), "Prescaler Delay", deleted |
| | | P10-82 | Figure 10.3.22 deleted |
| | | P10-83 to P10-122 | Section 10.4 overall,<br>Description of DMA transfer request generation (for only the TIO8) newly added |
| | | P10-87 | Description of "Count clock-dependent delay" along with Figure 10.4.2 newly added |
| | | P10-96 | Figure 10.4.7, "Outline Diagram of TIO5-9 Clock/Enable Inputs", altered |
| | | P10-103 | Description of W=△ corrected |
| | | P10-115 | (3), "Precautions on using TIO PWM output mode", newly added |
| | | P10-119 | Last item of Section 10.4.13. (2) added |
| | | P10-124 | Figure 10.5.1 corrected |
| | | | Description of "Count clock-dependent delay" along with Figure 10.5.2 newly added |
| | | P10-141 | Second paragraph of Section 10.6.7. (1) corrected |
| | | P11-6 | Description added to Section 11.1.2 |
| | | P11-16 | Note 1 added |
| | | P11-36 | Conversion time for Comparator mode in Table 11.3.3 corrected<br>Incorrect: 27<br>Correct: 29 |
| | | P11-39 | "AD1CSTP" is deleted from the explanation of "Forcible termination during scan operation" in Section 11.14 |
| | | P11-41, P11-42 | Equations altered |
| | | | |

| REVISION HISTORY | 32171 Group User’s Manual |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 2.00 | Sep 19, 2003 | P12-3 | Baud rate for UART mode in Table 12.1.1 changed<br>Before: 156K bits/sec<br>After: 1.25M bits/sec |
| | | P12-14 | Note in Section 12.2.3. (1) corrected |
| | | P12-24 | Last paragraph of Section 12.2.8 changed |
| | | P12-34 | Figure 12.4.1 corrected |
| | | P12-42 | Note deleted |
| | | P12-46 | Figure 12.6.3 corrected |
| | | | Note deleted |
| | | P12-53 | Figure 12.7.1 corrected |
| | | | Note deleted |
| | | P12-60 | Description in “Setting of Baud Rate (BRG) Regiser” partly deleted |
| | | P13-9 | Notes and Explanation added for 13.2.1. (4), “RFST (Forcible Reset) bit” |
| | | P13-77 | Figure 13.7.3 altered |
| | | P15-16 | Figure 5.4.3 corrected |
| | | P17-4 | Note 4 for Figure 17.2.3 corrected |
| | | P17-6 | Note 2 for Figure 17.3.2 altered |
| | | P18-5 | Figure 18.2.1 altered |
| | | P19-13 | TAP states for (2) continuous access to the same datagister in Figure 19.4.5 corrected |
| | | P19-14 | Note in Section 19.5 altered |
| | | P21-5,<br>P21-7 | Note 3 changed |
| | | P21-9 | Figures of ICCI-3V temperature characteristics newly added |
| | | P21-11 | Descriptions of IIAN in the tables, Section 21.1.4 addedd |
| | | P21-18 | (2) Electrical characteristics of each power supply pin when f(XIN)=10 MHz corrected to (4) Electrical characteristics of each power supply pin when f(XIN)= 8 MHz |
| | | P21-19 | “A-D conversion characteristics (Referenced to AVCC=VREF=VCCE=3.3V, Ta=25°C, f(XIN) = 8.0 MHz Unless Otherwise Noted)” corrected to “A-D conversion characteristics (Referenced to AVCC=VREF=VCCE=3.3V, Ta = -40 to 85°C, f(XIN) = 8.0 MHz Unless Otherwise Noted)” |
| | | | Descriptions of IIAN in the tables, Section 21.2.4 added |
| | | P21-23 | Maximum rated value for td(RTDCLKH-RTDRXD) corrected |
| | | P21-25 | “tv(BCLKL-BHWL)” corrected to “td(BCLKL-D)” |
| | | P21-26 | Parameter, “Byte enable delay time after write” corrected to “Valid Byte enable timer after write” |
| | | P21-27 | Figure 21.3.1 altered |
| | | P22-2 | Normal mode added to (1) Test conditions |
| | | | |

| REVISION HISTORY | 32171 Group User’s Manual |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 2.00 | Sep 19, 2003 | Appendix 3-2, 3-3 | Processing for Input/output ports in Table A3.1.1 alterd<br>Note 3 altered |
| | | Appendix 4-11 | Last item of Appendix 4.8.6 added |
| | | Appendix 4-24 | Last line of the 1st paragraph deleted |
| | | Appendix 4-25 | Description in (2), “Wiring of clock input/output pins”, altered |
| | | | Figure A4.13.2 changed |
| | | Appendix 4-26 | (3), “Wiring of the VCNT pin”, and Figure A4.13.3, “Example Wiring of the VCNT Pin”, newly added |
| | | | Figure A4.13.7, “Exmple Wiring of the MOD0 and MOD1 Pins”, altered |
| | | Appendix 4-29 | Description in (1), “Avoidance from large-current signal lines”, altered |
| | | | Figure A4.13.7, “Example Wiring of Large-current Signal Lines”, changed |
| | | Appendix 4-30 | Figure A.4.13.8, “Example Wiring of Rapidly Level-changing Signal Lines”, changed |
| | | Appendix 4-31,32 | (3), “Protection against signal lines that are the source of strong noise”, and Figures A4.13.9, “Example Processing of a Noise-laden Pin”, and A4.13.10, “Example Processing of Pins Adjacent to the Oscillator and VCNT Pins”, newly added |

## How to read internal I/O register tables

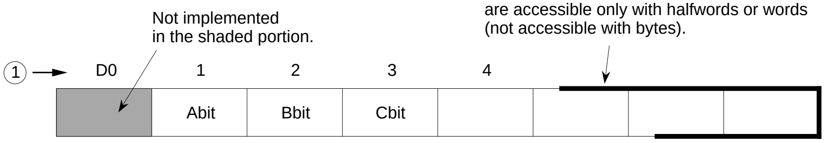
① Bit Numbers: Each register is connected with an internal bus of 16-bit wide, so the bit numbers of the registers located at even addresses are D0-D7, and those at odd addresses are D8-D15.

② State of Register at Reset: Represents the initial state of each register immediately after reset with hexadecimal numbers (undefined bits after reset are indicated each in column ③.)

③ At read:
○ ... read enabled
? ... read disabled (read value invalid)
0 ... Read always as 0
1 ... Read always as 1

④ At write:
○ : Write enabled
: Write enable conditionally
(include some conditions at write)
- : Write disabled (Written value invalid)

<Example of representation>



Not implemented in the shaded portion.

Registers represented with thick rectangles are accessible only with halfwords or words (not accessible with bytes).

② → <at reset: H'04>

| D | Bit name | Function | R | W |
|---|---|---|---|---|
| 0 | Not assigned. | | 0 | – |
| 1 | Abit<br>(...................) | 0: -----<br>1: ----- | ○ | ○ |
| 2 | Bbit<br>(...................) | 0: -----<br>1: ----- | ○ | ○ |
| 3 | Cbit<br>(...................) | 0: -----<br>1: ----- | ○ | ○ |

③ ④

# Table of contents

## CHAPTER 1 OVERVIEW

**1.1 Outline of the 32171 .......... 1-2**

1.1.1 M32R Family CPU Core .......... 1-2

1.1.2 Built-in Multiply-Accumulate Operation Function .......... 1-3

1.1.3 Built-in Flash Memory and RAM .......... 1-3

1.1.4 Built-in Clock Frequency Multiplier .......... 1-4

1.1.5 Built-in Powerful Peripheral Functions .......... 1-4

1.1.6 Built-in Full-CAN Function .......... 1-6

1.1.7 Built-in Debug Function .......... 1-6

**1.2 Block Diagram .......... 1-7**

**1.3 Pin Function .......... 1-11**

**1.4 Pin Layout .......... 1-17**

## CHAPTER 2 CPU

**2.1 CPU Registers .......... 2-2**

**2.2 General-purpose Registers .......... 2-2**

**2.3 Control Registers .......... 2-3**

2.3.1 Processor Status Word Register: PSW (CR0) .......... 2-4

2.3.2 Condition Bit Register: CBR (CR1) .......... 2-5

2.3.3 Interrupt Stack Pointer: SPI (CR2) .......... 2-5
User Stack Pointer: SPU (CR3)

2.3.4 Backup PC: BPC (CR6) .......... 2-5

**2.4 Accumulator .......... 2-6**

**2.5 Program Counter .......... 2-6**

**2.6 Data Formats .......... 2-7**

2.6.1 Data Types .......... 2-7

2.6.2 Data Formats .......... 2-8

**2.7 Precautions on CPU .......... 2-14**

## CHAPTER 3 ADDRESS SPACE

**3.1 Outline of Address Space ..................... 3-2**

**3.2 Operation Modes ..................... 3-6**

**3.3 Internal ROM Area and External Extension Area ..................... 3-8**

3.3.1 Internal ROM Area ..................... 3-8

3.3.2 External Extension Area ..................... 3-8

**3.4 Internal RAM Area and SFR Area ..................... 3-9**

3.4.1 Internal RAM Area ..................... 3-9

3.4.2 Special Function Register (SFR) Area ..................... 3-9

**3.5 EIT Vector Entry ..................... 3-23**

**3.6 ICU Vector Table ..................... 3-24**

**3.7 Notes on Address Space ..................... 3-26**

## CHAPTER 4 EIT

**4.1 Outline of EIT ..................... 4-2**

**4.2 EIT Events ..................... 4-3**

4.2.1 Exception ..................... 4-3

4.2.2 Interrupt ..................... 4-3

4.2.3 Trap ..................... 4-3

**4.3 EIT Processing Procedure ..................... 4-4**

**4.4 EIT Processing Mechanism ..................... 4-6**

**4.5 Acceptance of EIT Events ..................... 4-7**

**4.6 Saving and Restoring the PC and PSW ..................... 4-8**

**4.7 EIT Vector Entry ..................... 4-10**

**4.8 Exception Processing ..................... 4-11**

4.8.1 Reserved Instruction Exception (RIE) ..................... 4-11

4.8.2 Address Exception (AE) ..................... 4-13

**4.9 Interrupt Processing ..................... 4-15**

4.9.1 Reset Interrupt (RI) ..................... 4-15

4.9.2 System Break Interrupt (SBI) ..................... 4-16

4.9.3 External Interrupt (EI) ..................... 4-18

**4.10 Trap Processing ........................................ 4-20**

4.10.1 Trap (TRAP) ........................................ 4-20

**4.11 EIT Priority Levels ........................................ 4-22**

**4.12 Example of EIT Processing ........................................ 4-23**

**4.13 Precautions on EIT ........................................ 4-25**

## CHAPTER 5 INTERRUPT CONTROLLER (ICU)

**5.1 Outline of the Interrupt Controller (ICU) ........................................ 5-2**

**5.2 ICU Related Registers ........................................ 5-4**

5.2.1 Interrupt Vector Register ........................................ 5-5

5.2.2 Interrupt Mask Register ........................................ 5-6

5.2.3 SBI (System Break Interrupt) Control Register ........................................ 5-7

5.2.4 Interrupt Control Registers ........................................ 5-8

**5.3 Interrupt Sources in Internal Peripheral I/O ........................................ 5-12**

**5.4 ICU Vector Table ........................................ 5-13**

**5.5 Description of Interrupt Operation ........................................ 5-16**

5.5.1 Acceptance of Internal Peripheral I/O Interrupts ........................................ 5-16

5.5.2 Processing of Internal Peripheral I/O Interrupts by Handlers ........ 5-19

**5.6 Description of System Break Interrupt (SBI) Operation ........................................ 5-22**

5.6.1 Acceptance of SBI ........................................ 5-22

5.6.2 SBI Processing by Handler ........................................ 5-22

## CHAPTER 6 INTERNAL MEMORY

**6.1 Outline of the Internal Memory ........................................ 6-2**

**6.2 Internal RAM ........................................ 6-2**

**6.3 Internal Flash Memory ........................................ 6-3**

**6.4 Registers Associated with the Internal Flash Memory ........................................ 6-3**

6.4.1 Flash Mode Register ........................................ 6-4

6.4.2 Flash Status Registers ........................................ 6-5

6.4.3 Flash Control Registers ........................................ 6-8

6.4.4 Virtual Flash L Bank Register ........................................ 6-14

6.4.5 Virtual Flash S Bank Registers ........................................ 6-15

**6.5 Programming of the Internal Flash Memory ................................................ 6-16**

6.5.1 Outline of Programming Flash Memory ....................................... 6-16

6.5.2 Controlling Operation Mode during Programming Flash ............... 6-22

6.5.3 Programming Procedure to the Internal Flash Memory ................. 6-25

6.5.4 Flash Program Time (for Reference) ........................................... 6-39

**6.6 Boot ROM ......................................................................................... 6-41**

**6.7 Virtual Flash Emulation Function ........................................................ 6-42**

6.7.1 Virtual Flash Emulation Area ...................................................... 6-43

6.7.2 Entering Virtual Flash Emulation Mode ....................................... 6-50

6.7.3 Application Example of Virtual Flash Emulation Mode .................. 6-51

**6.8 Connecting to A Serial Programmer ..................................................... 6-53**

**6.9 Internal Flash Memory Protect Functions ............................................. 6-55**

**6.10 Precautions to Be Taken When Reprogramming Flash Memory ............. 6-56**

## CHAPTER 7 RESET

**7.1 Outline of Reset ................................................................................ 7-2**

**7.2 Reset Operation ................................................................................ 7-2**

7.2.1 Reset at Power-on ........................................................................ 7-2

7.2.2 Reset during Operation .................................................................. 7-2

7.2.3 Reset Vector Relocation during Flash Reprogramming ................. 7-2

**7.3 Internal State after Exiting Reset ........................................................ 7-3**

**7.4 Things To Be Considered after Exiting Reset ......................................... 7-8**

## CHAPTER 8 INPUT/OUTPUT PORTS AND PIN FUNCTIONS

**8.1 Outline of Input/Output Ports ............................................................ 8-2**

**8.2 Selecting Pin Functions ...................................................................... 8-4**

**8.3 Input/Output Port Related Registers .................................................... 8-6**

8.3.1 Port Data Registers ...................................................................... 8-8

8.3.2 Port Direction Registers .................................................................. 8-9

8.3.3 Port Operation Mode Registers .................................................... 8-10

**8.4 Port Peripheral Circuits ...................................................................... 8-22**

**8.5 Precautions on Input/output Ports ........................................................ 8-26**

## CHAPTER 9 DMAC

**9.1 Outline of the DMAC ........ 9-2**

**9.2 DMAC Related Registers ........ 9-5**

9.2.1 DMA Channel Control Register ........ 9-7

9.2.2 DMA Software Request Generation Registers ........ 9-18

9.2.3 DMA Source Address Registers ........ 9-19

9.2.4 DMA Destination Address Registers ........ 9-20

9.2.5 DMA Transfer Count Registers ........ 9-21

9.2.6 DMA Interrupt Request Status Registers ........ 9-22

9.2.7 DMA Interrupt Mask Registers ........ 9-24

**9.3 Functional Description of the DMAC ........ 9-28**

9.3.1 Cause of DMA Request ........ 9-28

9.3.2 DMA Transfer Processing Procedure ........ 9-32

9.3.3 Starting DMA ........ 9-33

9.3.4 Channel Priority ........ 9-33

9.3.5 Gaining and Releasing Control of the Internal Bus ........ 9-33

9.3.6 Transfer Units ........ 9-34

9.3.7 Transfer Counts ........ 9-34

9.3.8 Address Space ........ 9-34

9.3.9 Transfer Operation ........ 9-34

9.3.10 End of DMA and Interrupt ........ 9-38

9.3.11 Status of Each Register after Completion of DMA Transfer ........ 9-38

**9.4 Precautions about the DMAC ........ 9-39**

## CHAPTER 10 MULTIJUNCTION TIMERS

**10.1 Outline of Multijunction Timers ........ 10-2**

**10.2 Common Units of Multijunction Timer ........ 10-7**

10.2.1 Timer Common Register Map ........ 10-7

10.2.2 Prescaler Unit ........ 10-9

10.2.3 Clock Bus/Input-Output Event Bus Control Unit ........ 10-10

10.2.4 Input Processing Control Unit ........ 10-15

10.2.5 Output Flip-Flop Control Unit ........ 10-21

10.2.6 Interrupt Control Unit ........ 10-29

**10.3 TOP (Output-related 16-bit Timer) ........................................................ 10-46**

10.3.1 Outline of TOP ........................................................................ 10-46

10.3.2 Outline of Each Mode of TOP ................................................... 10-48

10.3.3 TOP Related Register Map ...................................................... 10-50

10.3.4 TOP Control Registers ............................................................ 10-53

10.3.5 TOP Counters (TOP0CT-TOP10CT) ....................................... 10-60

10.3.6 TOP Reload Registers (TOP0RL-TOP10RL) ........................... 10-61

10.3.7 TOP Correction Registers (TOP0CC-TOP10CC) ..................... 10-62

10.3.8 TOP Enable Control Register ................................................... 10-63

10.3.9 Operation in TOP Single-shot Output Mode (with Correction Function) ......... 10-67

10.3.10 Operation in TOP Delayed Single-shot Output Mode (With Correction Function) ...... 10-74

10.3.11 Operation in TOP Continuous Output Mode (Without Correction Function) ............. 10-79

**10.4 TIO (Input/Output-related 16-bit Timer) ................................................. 10-83**

10.4.1 Outline of TIO ........................................................................ 10-83

10.4.2 Outline of Each Mode of TIO ................................................... 10-85

10.4.3 TIO Related Register Map ....................................................... 10-88

10.4.4 TIO Control Registers ............................................................. 10-91

10.4.5 TIO Counter (TIO0CT-TIO9CT) ............................................. 10-102

10.4.6 TIO Reload 0/ Measure Register (TIO0RL0-TIO9RL0) ........... 10-103

10.4.7 TIO Reload 1 Registers (TIO0RL1-TIO9RL1) ........................ 10-104

10.4.8 TIO Enable Control Registers ................................................ 10-105

10.4.9 Operation in TIO Measure Free-run/Clear Input Modes .......... 10-108

10.4.10 Operation in TIO Noise Processing Input Mode ..................... 10-112

10.4.11 Operation in TIO PWM Output Mode ..................................... 10-113

10.4.12 Operation in TIO Single-shot Output Mode (without Correction Function) .. 10-117

10.4.13 Operation in TIO Delayed Single-shot Output Mode (without Correction Function) .. 10-119

10.4.14 Operation in TIO Continuous Output Mode (Without Correction Function) 10-121

**10.5 TMS (Input-related 16-bit Timer) ........................................................ 10-123**

10.5.1 Outline of TMS ..................................................................... 10-123

10.5.2 Outline of TMS Operation ...................................................... 10-123

10.5.3 TMS Related Register Map .................................................... 10-125

10.5.4 TMS Control Registers ......................................................... 10-126

10.5.5 TMS Counter (TMS0CT, TMS1CT) ....................................... 10-128

10.5.6 TMS Measure Registers (TMS0MR3-0, TMS1MR3-0) ........... 10-129

10.5.7 Operation of TMS Measure Input ........................................... 10-130

**10.6 TML (Input-related 32-bit Timer) .......... 10-132**

10.6.1 Outline of TML .......... 10-132

10.6.2 Outline of TML Operation .......... 10-133

10.6.3 TML Related Register Map .......... 10-134

10.6.4 TML Control Registers .......... 10-135

10.6.5 TML Counters .......... 10-137

10.6.6 TML Measure Registers .......... 10-139

10.6.7 Operation of TML Measure Input .......... 10-141

## CHAPTER 11 A-D CONVERTER

**11.1 Outline of A-D Converter .......... 11-2**

11.1.1 Conversion Modes .......... 11-5

11.1.2 Operation Modes .......... 11-6

11.1.3 Special Operation Modes .......... 11-10

11.1.4 A-D Converter Interrupt and DMA Transfer Requests .......... 11-13

**11.2 A-D Converter Related Registers .......... 11-14**

11.2.1 A-D Single Mode Register 0 .......... 11-16

11.2.2 A-D Single Mode Register 1 .......... 11-19

11.2.3 A-D Scan Mode Register 0 .......... 11-21

11.2.4 A-D Scan Mode Register 1 .......... 11-24

11.2.5 A-D Successive Approximation Register .......... 11-26

11.2.6 A-D0 Comparate Data Register .......... 11-27

11.2.7 10-bit A-D Data Registers .......... 11-28

11.2.8 8-bit A-D Data Registers .......... 11-29

**11.3 Functional Description of A-D Converter .......... 11-30**

11.3.1 How to Find Along Input Voltages .......... 11-30

11.3.2 A-D Conversion by Successive Approximation Method .......... 11-31

11.3.3 Comparator Operation .......... 11-33

11.3.4 Calculation of the A-D Conversion Time .......... 11-34

11.3.5 Definition of the A-D Conversion Accuracy .......... 11-37

**11.4 Precautions on Using A-D Converter .......... 11-39**

## CHAPTER 12 SERIAL I/O

**12.1 Outline of Serial I/O .......... 12-2**

**12.2 Serial I/O Related Registers ........................................................................ 12-6**

12.2.1 SIO Interrupt Related Registers ................................................. 12-7

12.2.2 SIO Interrupt Control Registers ................................................. 12-9

12.2.3 SIO Transmit Control Registers ............................................... 12-13

12.2.4 SIO Transmit/Receive Mode Registers .................................... 12-15

12.2.5 SIO Transmit Buffer Registers ................................................. 12-18

12.2.6 SIO Receive Buffer Registers .................................................. 12-19

12.2.7 SIO Receive Control Registers ................................................ 12-20

12.2.8 SIO Baud Rate Registers ........................................................ 12-23

**12.3 Transmit Operation in CSIO Mode ......................................................... 12-25**

12.3.1 Setting the CSIO Baud Rate .................................................... 12-25

12.3.2 Initial Settings for CSIO Transmission ...................................... 12-26

12.3.3 Starting CSIO Transmission .................................................... 12-28

12.3.4 Successive CSIO Transmission .............................................. 12-28

12.3.5 Processing at End of CSIO Transmission ................................ 12-29

12.3.6 Transmit Interrupt .................................................................... 12-29

12.3.7 Transmit DMA Transfer Request ............................................. 12-29

12.3.8 Typical CSIO Transmit Operation ............................................ 12-31

**12.4 Receive Operation in CSIO Mode .......................................................... 12-33**

12.4.1 Initial Settings for CSIO Reception .......................................... 12-33

12.4.2 Starting CSIO Reception ......................................................... 12-35

12.4.3 Processing at End of CSIO Reception ..................................... 12-35

12.4.4 About Successive Reception ................................................... 12-36

12.4.5 Flags Indicating the Status of CSIO Receive Operation ........... 12-37

12.4.6 Typical CSIO Receive Operation ............................................. 12-38

**12.5 Precautions on Using CSIO Mode ......................................................... 12-40**

**12.6 Transmit Operation in UART Mode ........................................................ 12-42**

12.6.1 Setting the UART Baud Rate ................................................... 12-42

12.6.2 UART Transmit/Receive Data Formats ................................... 12-43

12.6.3 Initial Settings for UART Transmission .................................... 12-45

12.6.4 Starting UART Transmission ................................................... 12-47

12.6.5 Successive UART Transmission ............................................. 12-47

12.6.6 Processing at End of UART Transmission .............................. 12-48

12.6.7 Transmit Interrupt .................................................................... 12-48

12.6.8 Transmit DMA Transfer Request ............................................. 12-48

12.6.9 Typical UART Transmit Operation ........................................... 12-50

**12.7 Receive Operation in UART Mode ........................................................ 12-52**

12.7.1 Initial Settings for UART Reception ......................................... 12-52

12.7.2 Starting UART Reception ....................................................... 12-54

12.7.3 Processing at End of UART Reception .................................... 12-54

12.7.4 Typical UART Receive Operation ............................................ 12-56

12.7.5 Detecting the Start Bit during UART Reception ........................ 12-58

**12.8 Fixed Period Clock Output Function ...................................................... 12-59**

**12.9 Precautions on Using UART Mode ........................................................ 12-60**

## CHAPTER 13 CAN MODULE

**13.1 Outline of the CAN Module ........................................................................ 13-2**

**13.2 CAN Module Related Registers ................................................................. 13-4**

13.2.1 CAN Control Register ............................................................... 13-8

13.2.2 CAN Status Register ............................................................... 13-11

13.2.3 CAN Extended ID Register ...................................................... 13-15

13.2.4 CAN Configuration Register .................................................... 13-16

13.2.5 CAN Time Stamp Count Register ............................................ 13-20

13.2.6 CAN Error Count Registers ..................................................... 13-21

13.2.7 CAN Baud Rate Prescaler ....................................................... 13-22

13.2.8 CAN Interrupt Related Registers .............................................. 13-23

13.2.9 CAN Mask Registers ............................................................... 13-31

13.2.10 CAN Message Slot Control Registers .................................... 13-36

13.2.11 CAN Message Slots ............................................................... 13-40

**13.3 CAN Protocol ............................................................................................ 13-55**

13.3.1 CAN Protocol Frame ............................................................... 13-55

**13.4 Initializing the CAN Module ...................................................................... 13-58**

13.4.1 Initialization of the CAN Module ............................................... 13-58

**13.5 Transmitting Data Frames ......................................................................... 13-61**

13.5.1 Data Frame Transmit Procedure ............................................. 13-61

13.5.2 Data Frame Transmit Operation .............................................. 13-63

13.5.3 Transmit Abort Function .......................................................... 13-64

**13.6 Receiving Data Frames ........................................................................ 13-65**

13.6.1 Data Frame Receive Procedure ............................................... 13-65

13.6.2 Data Frame Receive Operation ................................................ 13-67

13.6.3 Reading Out Received Data Frames ........................................ 13-69

**13.7 Transmitting Remote Frames .................................................................. 13-71**

13.7.1 Remote Frame Transmit Procedure ......................................... 13-71

13.7.2 Remote Frame Transmit Operation .......................................... 13-73

13.7.3 Reading Out Received Data Frames when Set for Remote Frame Transmission ...... 13-76

**13.8 Receiving Remote Frames ....................................................................... 13-78**

13.8.1 Remote Frame Receive Procedure .......................................... 13-78

13.8.2 Remote Frame Receive Operation ........................................... 13-80

## CHAPTER 14 REAL-TIME DEBUGGER (RTD)

**14.1 Outline of the Real-Time Debugger (RTD) ................................................. 14-2**

**14.2 Pin Function of the RTD ............................................................................ 14-3**

**14.3 Functional Description of the RTD ............................................................ 14-4**

14.3.1 Outline of RTD Operation ........................................................... 14-4

14.3.2 Operation of RDR (Real-time RAM Content Output) .................. 14-5

14.3.3 Operation of WRR (RAM Content Forcible Rewrite) .................. 14-7

14.3.4 Operation of VER (Continuous Monitor) ..................................... 14-9

14.3.5 Operation of VEI (Interrupt Request) ........................................ 14-10

14.3.6 Operation of RCV (Recover from Runaway) ............................ 14-11

14.3.7 Method to Set a Specified Address when Using the RTD ......... 14-12

14.3.8 Resetting the RTD .................................................................... 14-13

**14.4 Typical Connection with the Host ............................................................ 14-14**

## CHAPTER 15 EXTERNAL BUS INTERFACE

**15.1 External Bus Interface Related Signals ..................................................... 15-2**

**15.2 Read/Write Operations ............................................................................... 15-6**

**15.3 Bus Arbitration ......................................................................................... 15-12**

**15.4 Typical Connection of External Extension Memory ................................ 15-14**

## CHAPTER 16 WAIT CONTROLLER

**16.1 Outline of the Wait Controller ........................................................................ 16-2**

**16.2 Wait Controller Related Registers ................................................................ 16-4**

16.2.1 Wait Cycles Control Register (WTCCR) ..................................... 16-5

**16.3 Typical Operation of the Wait Controller ................................................... 16-6**

## CHAPTER 17 RAM BACKUP MODE

**17.1 Outline of RAM Backup Mode ......................................................................... 17-2**

**17.2 Example of RAM Backup when Power is Down ........................................ 17-2**

17.2.1 Normal Operating State ............................................................ 17-3

17.2.2 RAM Backup State ................................................................... 17-4

**17.3 Example of RAM Backup for Saving Power Consumption ....................... 17-5**

17.3.1 Normal Operating State ............................................................ 17-6

17.3.2 RAM Backup State ................................................................... 17-7

17.3.3 Precautions to Be Observed at Power-on .................................. 17-8

**17.4 Exiting RAM Backup Mode (Wakeup) ........................................................ 17-9**

## CHAPTER 18 OSCILLATION CIRCUIT

**18.1 Oscillator Circuit .............................................................................................. 18-2**

18.1.1 Example of an Oscillator Circuit ................................................. 18-2

18.1.2 System Clock Output Function .................................................. 18-3

18.1.3 Oscillation Stabilization Time at Power-on ................................. 18-4

**18.2 Clock Generator Circuit .................................................................................. 18-5**

## CHAPTER 19 JTAG

**19.1 Outline of JTAG ................................................................................................ 19-2**

**19.2 Configuration of the JTAG Circuit ................................................................ 19-3**

**19.3 JTAG Registers ................................................................................................ 19-4**

19.3.1 Instruction Register (JTAGIR) .................................................... 19-4

19.3.2 Data Registers ........................................................................... 19-5

**19.4 Basic Operation of JTAG ........................................................................... 19-6**

19.4.1 Outline of JTAG Operation ........................................................ 19-6

19.4.2 IR Path Sequence ..................................................................... 19-8

19.4.3 DR Path Sequence .................................................................. 19-10

19.4.4 Examining and Setting Data Registers ..................................... 19-12

**19.5 Boundary Scan Description Language ................................................... 19-14**

**19.6 Precautions on Board Design when Using JTAG ................................... 19-15**

**19.7 Processing Pins when Not Using JTAG ................................................. 19-16**

## CHAPTER 20 POWER-ON/POWER-OFF SEQUENCE

**20.1 Configuration of the Power Supply Circuit ............................................... 20-2**

**20.2 Power-On Sequence .................................................................................. 20-4**

20.2.1 Power-On Sequence When Not Using RAM Backup .................. 20-4

20.2.2 Power-On Sequence When Using RAM Backup ........................ 20-6

**20.3 Power-off Sequence .................................................................................. 20-8**

20.3.1 Power-off Sequence When Not Using RAM Backup ................... 20-8

20.3.2 Power-off Sequence When Using RAM Backup ....................... 20-10

## CHAPTER 21 ELECTRICAL CHARACTERISTICS

**21.1 Electrical Characteristics (VCCE = 5 V) .................................................... 21-2**

21.1.1 Absolute Maximum Ratings ....................................................... 21-2

21.1.2 Recommended Operating Conditions ......................................... 21-3

21.1.3 DC Characteristics ..................................................................... 21-5

21.1.4 A-D Conversion Characteristics ............................................... 21-11

**21.2 Electrical Characteristics (VCCE = 3.3 V) ............................................... 21-12**

21.2.1 Absolute Maximum Ratings ..................................................... 21-12

21.2.2 Recommended Operating Conditions ....................................... 21-13

21.2.3 DC Characteristics ................................................................... 21-15

21.2.4 A-D Conversion Characteristics ............................................... 21-19

**21.3 AC Characteristics ................................................................................... 21-20**

21.3.1 Timing Requirements ............................................................... 21-20

21.3.2 Switching Characteristics ......................................................... 21-24

21.3.3 AC Characteristics ................................................................... 21-27

## CHAPTER 22 TYPICAL CHARACTERISTICS

**22.1 A-D Conversion Characteristics ................................................................ 22-2**

## APPENDIX 1 MECHANICAL SPECIFICATIONS

**Appendix 1.1 Dimensional Outline Drawing ...................................... Appendix 1-2**

## APPENDIX 2 INSTRUCTION PROCESSING TIME

**Appendix 2.1 M32R/ECU Instruction Processing Time ..................... Appendix 2-2**

## APPENDIX 3 PROCESSING OF UNUSED PINS

**Appendix 3.1 Example for Processing Unused Pins ......................... Appendix 3-2**

## APPENDIX 4 SUMMARY OF PRECAUTIONS

**Appendix 4.1 Precautions Regarding the CPU ................................. Appendix 4-2**

Appendix 4.1.1 Things to be noted for data transfer ................ Appendix 4-2

**Appendix 4.2 Precautions on Address Space ................................... Appendix 4-2**

Appendix 4.2.1 Virtual flash emulation function ........................ Appendix 4-2

**Appendix 4.3 Precautions on EIT ....................................................... Appendix 4-3**

**Appendix 4.4 Precautions to Be Taken When Reprogramming Flash Memory ............................................................. Appendix 4-3**

**Appendix 4.5 Things to Be Considered after Exiting Reset ............. Appendix 4-4**

Appendix 4.5.1 Input/output Ports .......................................... Appendix 4-4

**Appendix 4.6 Precautions on Input/output Ports .............................. Appendix 4-4**

Appendix 4.6.1 When using the ports in output mode ............. Appendix 4-4

**Appendix 4.7 Precautions about the DMAC ....................................... Appendix 4-5**

Appendix 4.7.1 About writing to DMAC related registers .......... Appendix 4-5

Appendix 4.7.2 Manipulating DMAC related registers by DMA transfer .................................................. Appendix 4-6

Appendix 4.7.3 About the DMA Intrrupt Reqest Status Register ........................................................ Appendix 4-6

Appendix 4.7.4 About the stable operation of DMA transfer ..... Appendix 4-6

**Appendix 4.8 Precautions on Multijunction Timers .......................... Appendix 4-7**

Appendix 4.8.1 Precautions to be observed when using TOP single-shot output mode .......................... Appendix 4-7

Appendix 4.8.2 Precautions to be observed when using TOP delayed single-shot output mode ........... Appendix 4-9

Appendix 4.8.3 Precautions to be observed when using TOP continuous output mode ....................... Appendix 4-10

Appendix 4.8.4 Precautions to be observed when using TIO measure free-run/clear input modes ...... Appendix 4-11

Appendix 4.8.5 Precautions to be observed when using TIO single-shot output mode ......................... Appendix 4-11

Appendix 4.8.6 Precautions to be observed when using TIO delayed single-shot output mode ........... Appendix 4-11

Appendix 4.8.7 Precautions to be observed when using TIO continuous output mode ......................... Appendix 4-12

Appendix 4.8.8 Precautions to be observed when using TMS measure input ...................................... Appendix 4-12

Appendix 4.8.9 Precautions to be observed when using TML measure input ...................................... Appendix 4-13

**Appendix 4.9 Precautions on Using A-D Converters ...................... Appendix 4-14**

**Appendix 4.10 Precautions on Serial I/O ......................................... Appendix 4-18**

Appendix 4.10.1 Precautions on Using CSIO mode ............... Appendix 4-18

Appendix 4.10.2 Precautions on Using UART mode .............. Appendix 4-20

**Appendix 4.11 Precautions on RAM Backup Mode ......................... Appendix 4-21**

Appendix 4.11.1 Precautions to be observed at Power-on ..... Appendix 4-21

**Appendix 4.12 Precautions on Processing JTAG Pins ................... Appendix 4-22**

Appendix 4.12.1 Precautions on Board Design when Using JTAG ................................................ Appendix 4-22

Appendix 4.12.2 Processing Pins when Not Using JTAG ...... Appendix 4-23

**Appendix 4.13 Precautions about Noise ......................................... Appendix 4-24**

Appendix 4.13.1 Reduction of Wiring Length ......................... Appendix 4-24

Appendix 4.13.2 Inserting a Bypass Capacitor between VSS and VCC Lines ................................... Appendix 4-27

Appendix 4.13.3 Processing Analog Input Pin Wiring ............ Appendix 4-28
Appendix 4.13.4 Consideration about the Oscillator
and VCNT Pin ............................................ Appendix 4-29
Appendix 4.13.5 Processing Input/Output Ports ..................... Appendix 4-33

# CHAPTER 1
# OVERVIEW

1.1 Outline of the 32171
1.2 Block Diagram
1.3 Pin Function
1.4 Pin Layout

# 1.1 Outline of the 32171

## 1.1.1 M32R Family CPU Core

### (1) Based on RISC architecture

- The 32171 is a 32-bit RISC single-chip microcomputer which is built around the M32R family CPU core (hereafter referred to as the M32R) and incorporates flash memory, RAM, and various other peripheral functions-all integrated into a single chip.
- The M32R is based on RISC architecture. Memory access is performed using load and store instructions, and various arithmetic operations are executed using register-to-register operation instructions. The M32R internally contains sixteen 32-bit general-purpose registers and has 83 distinct instructions.
- The M32R supports compound instructions such as Load & Address Update and Store & Address Update, in addition to ordinary load and store instructions. These compound instructions help to speed up data transfers.

### (2) 5-stage pipelined processing

- The M32R uses 5-stage pipelined instruction processing consisting of Instruction Fetch, Decode, Execute, Memory Access, and Write Back. Not just load and store instructions or register-to-register operation instructions, compound instructions such as Load & Address Update and Store & Address Update also are executed in one cycle.
- Instructions are entered into the execution stage in the order they are fetched, but this does not always mean that the first instruction entered is executed first. If the execution of a load or store instruction entered earlier is delayed by one or more wait cycles inserted in memory access, a register-to-register operation instruction entered later may be executed before said load or store instruction. By using "out-of-order-completion" like this, the M32R controls instruction execution without wasting clock cycles.

### (3) Compact instruction code

- The M32R instructions come in two types: one consisting of 16 bits in length, and the other consisting of 32 bits in length. Use of the 16-bit length instruction format especially helps to suppress the program code size.
- Some 32-bit long instructions can branch directly to a location 32 Mbytes forward or backward from the instruction address being executed. Compared to architectures where address space is segmented, this direct jump allows for easy programming.

### 1.1.2 Built-in Multiply-Accumulate Operation Function

**(1) Built-in high-speed multiplier**

- The M32R incorporates a 32-bit × 16-bit high-speed multiplier which enables it to execute a 32-bit × 32-bit integral multiplication instruction in three cycles (1 cycle = 25 ns when using a 40 MHz internal CPU clock).

**(2) Supports Multiply-Accumulate operation instructions comparable to DSP**

- The M32R supports the following four modes of Multiply-Accumulate operation instructions (or multiplication instructions) using a 56-bit accumulator. Any of these operations can be executed in one cycle.

  (a) 16 high-order register bits × 16 high-order register bits
  (b) 16 low-order register bits × 16 low-order register bits
  (c) Entire 32 register bits × 16 high-order register bits
  (d) Entire 32 register bits × 16 low-order register bits

- The M32R has instructions to round off the value stored in the accumulator to 16 or 32 bits, as well as instructions to shift the accumulator value to adjust digits and store the digit-adjusted value in a register. These instructions also can be executed in one cycle, so that when combined with high-speed data transfer instructions such as Load & Address Update and Store & Address Update, they enable the M32R to exhibit high data processing capability comparable to that of DSP.

### 1.1.3 Built-in Flash Memory and RAM

- The 32171 contains flash memory and RAM which can be accessed with no wait states, allowing you to build a high-speed embedded system.
- The internal flash memory allows for on-board programming (you can write to it while being mounted on the printed circuit board). Use of flash memory means the chip engineered at the development phase can be used directly in mass-production, so that you can smoothly migrate from prototype to mass-production without changing the printed circuit board.
- The internal flash memory can be rewritten 100 times.
- The internal flash memory has a virtual-flash emulation function, allowing the internal RAM to be artificially mapped into part of the internal flash memory. This function, when combined with the internal Real-Time Debugger (RTD), facilitates data tuning on ROM tables.
- The internal RAM can be accessed for read or rewrite from an external device independently of the M32R by using RTD (real-time debugger). It is communicated with external devices by RTD's exclusive clock-synchronized serial I/O.

### 1.1.4 Built-in Clock Frequency Multiplier

- The 32171 internally multiplies the input clock signal frequency by 4 and the internal peripheral clock by 2. If the input clock frequency is 10.0 MHz, the CPU clock frequency will be 40 MHz and the internal clock frequency 20 MHz.

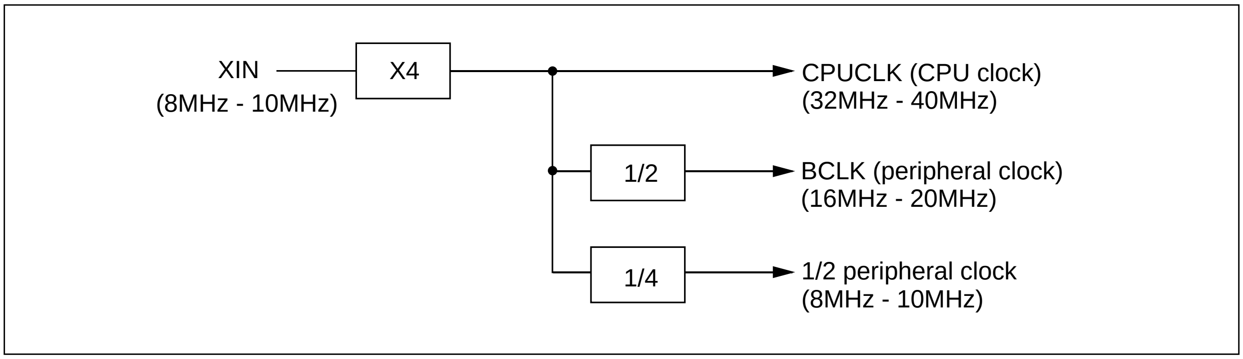


**Figure 1.1.1 Conceptual Diagram of the Clock Frequency Multiplier**

**Table 1.1.1 Clock**

| Functional Block | Features |
|---|---|
| CPUCLK | • CPU clock: Defined as f(CPUCLK) when it indicates the operating clock frequency for the M32R core, internal flah memory and inernal RAM. |
| BCLK | • Peripheral clock: Defined as f(BCLK) when it indicates the operating clock frequency for the internal peripheral I/O and external data bus. |
| Clock output (BCLK pin output) | • A clock with the same frequency as f(BCLK) is output from this pin. |
| 1/2 peripheral clock | • Count-source clock of MJT. Sampling clock of TCLK, TIN. |

### 1.1.5 Built-in Powerful Peripheral Functions

**(1) Built-in multijunction timer (MJT)**

- The multijunction timer is configured with the following 37 channels timers:
  - (a) 16-bit output-related timer × 11 channels
  - (b) 16-bit input/output-related timer × 10 channels
  - (c) 16-bit input-related timer × 8 channels
  - (d) 32-bit input-related timer × 8 channels

  Each timer has multiple modes of operation, which can be selected according of the purpose of use.
- The multijunction timer has internal clock bus, input event bus, and output event bus, allowing multiple timers to be combined for use internally. This provides a flexible way to make use of timer functions.
- The output-related timers (TOP) have a correction function. This function allows the timer's count value in progress to be increased or reduced as desired, thus materializing real-time output control.

### (2) Built-in 10-channel DMA

- The 10-channel DMA is built-in, supporting data transfers between internal peripheral I/Os or between internal peripheral I/O and internal RAM. Not only can DMA transfer requests be generated in software, but can also be triggered by a signal generated by an internal peripheral I/O (e.g., A-D converter, MJT, or serial I/O).
- Cascaded connection between DMA channels (DMA transfer in a channel is started by completion of transfer in another) is also supported, allowing for high-speed transfer processing without imposing any extra load on the CPU.

### (3) Built-in 16-channel A-D converter

- The 32171 contains one 16-channel A-D converter which can convert data in 10-bit resolution. In addition to single A-D conversion in each channel, successive A-D conversion in four, eight, or 16 channels combined into one unit is possible.
- In addition to ordinary A-D conversion, a comparator mode is supported in which the A-D conversion result is compared with a given set value to determine the relative magnitudes of two quantities.
- When A-D conversion is completed, the 32171 can generate not only an interrupt, but can also generate a DMA transfer request.
- The 32171 supports two readout modes, so that A-D conversion results can be read out in 8 bits or 10 bits.

### (4) High-speed serial I/O

- The 32171 incorporates 3 channels of serial I/O, which can be set for clock-synchronized serial I/O or UART.
- When set for clock-synchronized serial I/O, the data transfer rate is a high 2 Mbits per second.
- When data reception is completed or the transmit buffer becomes empty, the serial I/O can generate a DMA transfer request signal.

### (5) Built-in Real-Time Debugger (RTD)

- The Real-Time Debugger (RTD) provides a function for the M32R/ECU's internal RAM to be accessed directly from an external device. The debugger communicates with external devices through its exclusive clock-synchronized serial I/O.
- By using the RTD, you can read the contents of the internal RAM or rewrite its data from an external device independently of the M32R.
- The debugger can generate an RTD interrupt to notify that RTD-based data transmission or reception is completed.

### (6) Eight-level interrupt controller

- The interrupt controller manages interrupt requests from each internal peripheral I/O by resolving interrupt priority in eight levels including an interrupt-disabled state. Also, it can accept external interrupt requests due to power-down detection or generated by a watchdog timer as a System Break Interrupt (SBI).

### (7) Three operation modes

- The M32R/ECU has three operation modes: single-chip mode, external extension mode, and processor mode. The address space and external pin functions of the M32R/ECU are switched over according to a mode in which it operates. The MOD0 and MOD1 pins are used to set a mode.

### (8) Wait controller

- The wait controller supports access to external devices by the M32R. In all but single-chip mode, the external extension area provides 4 Mbytes of space.

## 1.1.6 Built-in Full-CAN Function

- The 32171 contains CAN Specification V2.0B active-compliant CAN module, thereby providing 16 message slots.

## 1.1.7 Built-in Debug Function

- The 32171 supports JTAG interface. Boundary scan test can be performed using this JTAG interface.

## 1.2 Block Diagram

Figure 1.2.1 shows a block diagram of the 32171. Features of each block are shown in Tables 1.2.1 through 1.2.3.

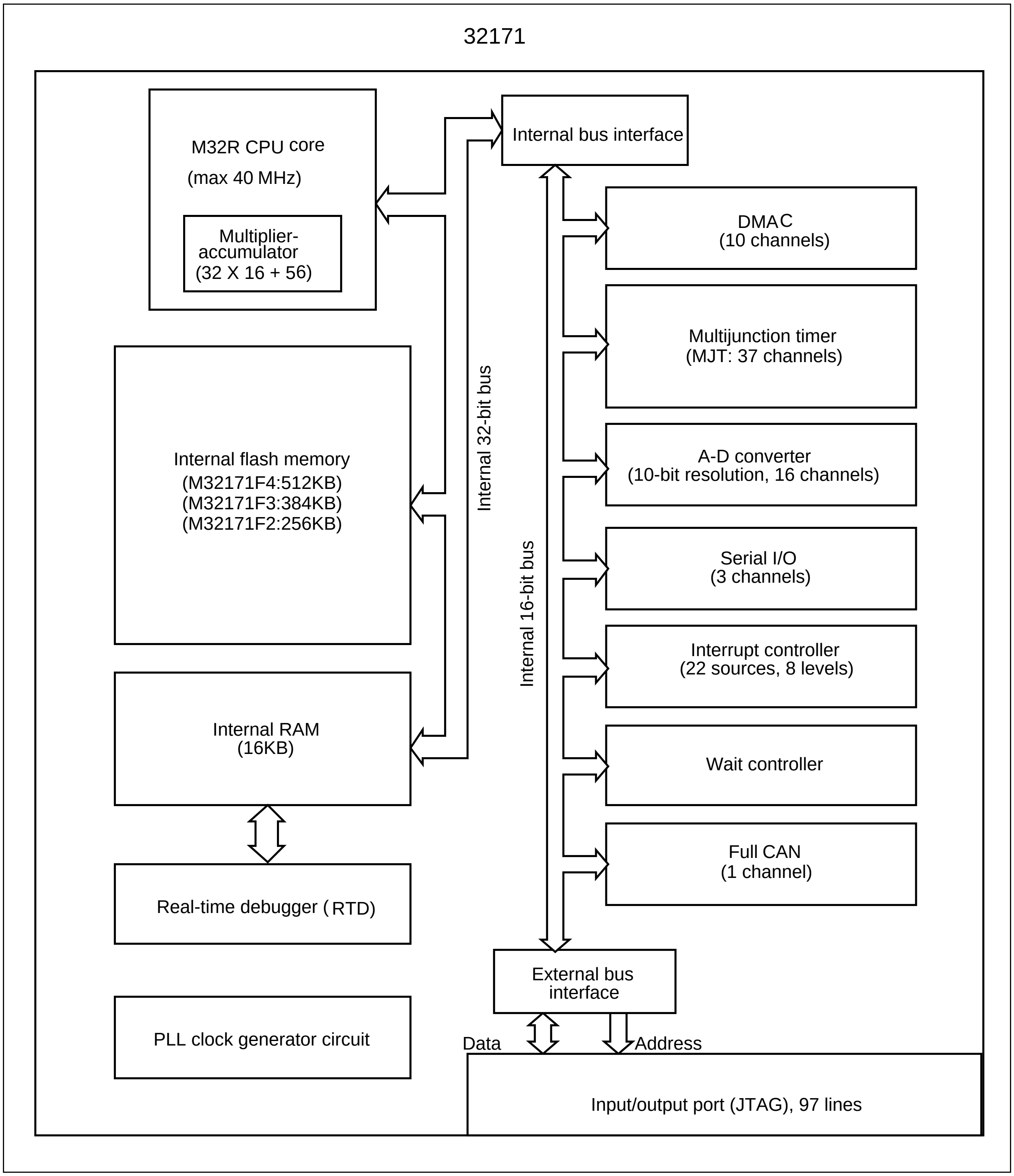


**Figure 1.2.1 Block Diagram of the 32171**

### Table 1.2.1 Features of the M32R Family CPU Core

| Functional Block | Features |
|---|---|
| M32R family CPU core | • Bus specifications<br>Basic bus cycle: 25 ns (when operating with 40 MHz CPU clock)<br>Logical address space: 4Gbytes, linear<br>External extension area: Maximum 4 Mbytes<br>External data bus: 16 bits<br>• Implementation: Five-stage pipeline<br>• Internal 32-bit architecture for the core<br>• Register configuration<br>General-purpose register: 32 bits × 16 registers<br>Control register: 32 bits × 5 registers<br>• Instruction set<br>16-bit and 32-bit instruction formats<br>83 distinct instructions and 6 addressing modes<br>• Built-in multiplier/accumulator (32 × 16 + 56) |

### Table 1.2.2 Features of Internal Memory

| Functional Block | Features |
|---|---|
| RAM | • Capacity : 16 Kbytes<br>• No-wait access<br>• By using RTD (real-time debugger), the internal RAM can be accessed for read or rewrite from external devices independently of the M32R. |
| Flash memory | • Capacity<br>M32171F4 : 512 Kbytes<br>M32171F3 : 384 Kbytes<br>M32171F2 : 256 Kbytes<br>• No-wait access<br>• Durability: Can be rewritten 100 times |

### Table 1.2.3 Features of Internal Peripheral I/O

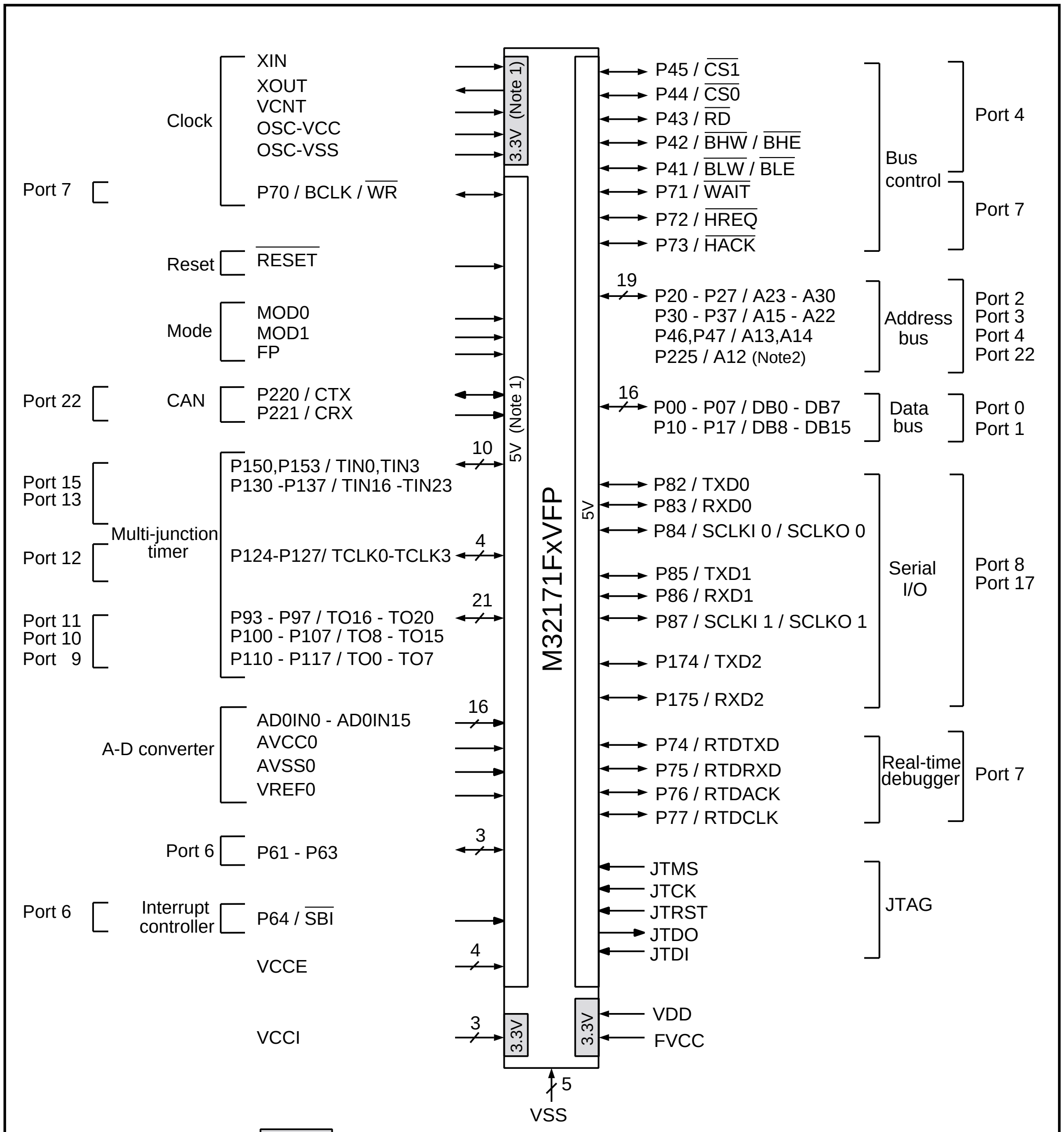
| Functional Block | Features |
|---|---|
| DMA | • 10-channel DMA<br>• Supports transfer between internal peripheral I/Os, between internal RAMs, and between internal peripheral I/O and internal RAM.<br>• Capable of advanced DMA transfer when operating in combination with internal peripheral I/O<br>• Capable of cascaded connection between DMA channels (DMA transfer in a channel is started by completion of transfer in another) |
| Multijunction | • 37-channel multifunction timer<br>• Contains output-related timer × 11 channels, input/output-related timer × 10 channels, 16-bit input-related timer × 8 channels, and 32-bit input-related timer × 8 channels.<br>• Capable of flexible timer configuration by mutual connection between each channel. |
| A-D converter | • 16-channel, 10-bit resolution A-D converter<br>• Incorporates comparator mode<br>• Can generate interrupt or start DMA transfer upon completion of A-D conversion.<br>• Can read out conversion results in 8 or 10 bits. |
| Serial I/O | • 3-channel serial I/O<br>• Can be set for clock-synchronized serial I/O or UART.<br>• Capable of high-speed data transfer at 2 Mbits per second when clock synchronized or 156 Kbits per second during UART. |
| Real-time debugger | • Can rewrite or monitor the internal RAM independently of the CPU by command input from an external source.<br>• Has its exclusive clock-synchronized serial port. |
| Interrupt controller | • Accepts and manages interrupt requests from internal peripheral I/O.<br>• Resolves interrupt priority in 8 levels including interrupt-disabled state. |
| Wait controller | • Controls wait state for access to external extension areas.<br>• Can insert 1 to 4 wait cycles by setting in software and extend wait period by external $\overline{\text{WAIT}}$ signal. |
| Clock PLL | • Multiply-by-4 clock generator circuit<br>• Maximum 40 MHz of CPU clock (CPU, internal ROM, internal RAM access)<br>• Maximum 20 MHz of internal peripheral clock (peripheral module access)<br>• Maximum external input clock frequency=10 MHz |
| CAN | • Sixteen message slots |
| JTAG | • Capable of boundary scan |

Table 1.2.4 List of Type Name

| Type Name | RAM Size (K bytes) | ROM Size (K bytes) | Package | Number of Pins |
|---|---|---|---|---|
| M32171F2VFP | 16 | 256 | 144LQFP | 144 |
| M32171F3VFP | 16 | 384 | 144LQFP | 144 |
| M32171F4VFP | 16 | 512 | 144LQFP | 144 |

## 1.3 Pin Function

Figure 1.3.1 shows pin functions of the M32171FxVFP. Table 1.3.1 explains the pin functions.



Note 1: 3.3V : denotes blocks operating with a 3.3 V power supply.
5V : denotes blocks operating with a 5 V or 3.3 V power supply.

Note 2: Use caution when using this port because it has a debug event function.

**Figure 1.3.1 Pin Function Diagram of 144LQFP**

### Table 1.3.1 Description of the 32171 Pin Function (1/5)

<table>
<tr><th>Type</th><th>Pin Name</th><th>Signal Name</th><th>Input/Output</th><th>Function</th></tr>
<tr><td>Power supply</td><td>VCCE</td><td>Power supply</td><td>—</td><td>Power supply to external I/O ports (5 V or 3.3 V).</td></tr>
<tr><td></td><td>VCCI</td><td>Power supply</td><td>—</td><td>Power supply to internal logic (3.3 V).</td></tr>
<tr><td></td><td>VDD</td><td>RAM power supply</td><td>—</td><td>Power supply for internal RAM backup (3.3 V).</td></tr>
<tr><td></td><td>FVCC</td><td>Flash power supply</td><td>—</td><td>Power supply for internal flash memory (3.3 V).</td></tr>
<tr><td></td><td>VSS</td><td>Ground</td><td>—</td><td>Connect all VSS to ground (GND).</td></tr>
<tr><td>Clock</td><td>XIN,<br>XOUT</td><td>Clock</td><td>Input<br>Output</td><td>Clock input/output pins. These pins contains a PLL-based frequency multiplier circuit. Apply a clock whose frequency is 1/4 the operating frequency. (When using 40 MHz CPU clock, XIN input = 10.0 MHz)</td></tr>
<tr><td></td><td>BCLK/$\overline{\text{WR}}$</td><td>System clock/write</td><td>Output</td><td>This pin outputs a clock whose frequency is twice that of external input clock. (When using 10 MHz external input clock, BCLK output = 20 MHz). Use this output when external operation needs to be synchronized.<br>If $\overline{\text{WR}}$ is selected, it indicates the byte position to which valid data is transferred when writing to an external device.</td></tr>
<tr><td></td><td>OSC-VCC</td><td>Power supply</td><td>—</td><td>Power supply for PLL circuit. Connect OSC-VCC to the power supply rail (3.3 V).</td></tr>
<tr><td></td><td>OSC-VSS</td><td>Ground</td><td>—</td><td>Connect OSC-VSS to ground.</td></tr>
<tr><td></td><td>VCNT</td><td>PLL control</td><td>Input</td><td>This pin controls the PLL circuit. Connect a resistor and capacitor to it. (For external circuits, refer to Section 18.1.1, "Example of an Oscillator Circuit.")</td></tr>
<tr><td>Reset</td><td>$\overline{\text{RESET}}$</td><td>Reset</td><td>Input</td><td>This pin resets the internal circuit.</td></tr>
<tr><td>Mode</td><td>MOD0<br>MOD1</td><td>Mode</td><td>Input</td><td>These pins set operation mode.
<table>
<tr><th>MOD0</th><th>MOD1</th><th>Mode</th></tr>
<tr><td>0</td><td>0</td><td>Single-chip mode</td></tr>
<tr><td>0</td><td>1</td><td>External extension mode</td></tr>
<tr><td>1</td><td>0</td><td>Processor mode<br>(Boot mode) (Note1)</td></tr>
<tr><td>1</td><td>1</td><td>(Reserved)</td></tr>
</table></td></tr>
<tr><td>Address Bus</td><td>A12 – A30</td><td>Address Bus</td><td>Output</td><td>The device has 19 address lines (A12-A30) to allow two channels of up to 1 MB of memory space to be added external to the chip. A31 is not output.</td></tr>
</table>

Note 1: For boot mode, refer to Chapter 6, "Internal Memory."

Table 1.3.1 Description of the 32171 Pin Function (2/5)

| Type | Pin Name | Signal Name | Input/Output | Function |
|---|---|---|---|---|
| Data bus | DB0-DB15 | Data bus | Input/output | These pins comprise 16-bit data bus to connect external devices. In write cycles, the valid byte positions to be written on the 16-bit data bus are output as $\overline{\text{BHW}}$/$\overline{\text{BHE}}$ and $\overline{\text{BLW}}$/$\overline{\text{BLE}}$. In read cycles, data is always read from the 16-bit data bus. However, when transferring to the internal circuit of the M32R, only data at the valid byte positions are transferred. |
| Bus control | $\overline{\text{CS0}}$,<br>$\overline{\text{CS1}}$ | Chip select | Output | These pins comprise external device chip select signal. For areas for which a chip select signal is output, refer to Chapter 3, "Address Space." |
| | $\overline{\text{RD}}$ | Read | Output | This signal is output when reading an external device. |
| | $\overline{\text{BHW}}$/$\overline{\text{BHE}}$ | Byte high write/enable | Output | Indicates the byte position to which valid data is transferred when writing to an external device. $\overline{\text{BHW}}$/$\overline{\text{BHE}}$ corresponds to the upper address (D0-D7 is valid); $\overline{\text{BLW}}$/$\overline{\text{BLE}}$ corresponds to the lower address (D8-D15 is valid). |
| | $\overline{\text{BLW}}$/$\overline{\text{BLE}}$ | Byte low write/enable | Output | |
| | $\overline{\text{WAIT}}$ | Wait | Input | When the M32R accesses an external device, a low on this $\overline{\text{WAIT}}$ input extends the wait cycle. |
| | $\overline{\text{HREQ}}$ | Hold request | Input | This pin is used by an external device to request control of the external bus. A low on this $\overline{\text{HREQ}}$ input causes the M32R to enter a hold state. |
| | $\overline{\text{HACK}}$ | Hold acknowledge | Output | This signal is used to notify that the M32R has entered a hold state and relinquished control of the external bus. |
| Multi-junction timer | TIN 0,TIN 3<br>TIN 16–TIN 23 | Timer input | Input | Input pins for multijunction timer. |
| | TO 0– TO 20 | Timer output | Output | Output pins for the multijunction timer. |
| | TCLK 0– TCLK 3 | Timer clock | Input | Clock input pins for the multijunction timer. |
| A-D converter | AVCC0 | Analog power supply | — | AVCC0 is the power supply for the A-D0 converter. Connect AVCC0 to the power supply rail (5 V or 3.3 V). |
| | AVSS0 | Analog ground | — | AVSS0 is analog ground for the A-D0 converter. Connect AVSS0 to the ground. |

### Table 1.3.1 Description of the 32171 Pin Function (3/5)

| Type | Pin Name | Signal Name | Input/Output | Function |
|---|---|---|---|---|
| A-D converter | AD0IN0 – AD0IN15 | Analog input | Input | 16-channel analog input pins for the A-D0 converter. |
| | VREF0 | voltage input | Input | VREF0 is the reference voltage input pin for the A-D0 converter. |
| Interrupt controller | $\overline{\text{SBI}}$ | System break interrupt | Input | System break interrupt (SBI) input pin for the interrupt controller |
| Serial I/O | SCLKI0 / SCLKO0 | UART transmit/ receive clock output<br>or<br>CSIO transmit/ receive clock imput/output | Input/output | When channel 0 is in UART mode:<br>This pin outputs a clock derived from BRG output by halving it.<br>When channel 0 is in CSIO mode:<br>This pin accepts as its input a transmit/receive clock when external clock source is selected or outputs a transmit/receive clock when internal clock source is selected. |
| | SCLKI1 / SCLKO1 | UART transmit/ receive clock output<br>or<br>CSIO transmit/ receive clock input/output | Input/output | When channel 1 is in UART mode:<br>This pin outputs a clock derived from BRG output by halving it.<br>When channel 1 is in CSIO mode:<br>This pin accepts as its input a transmit/receive clock when external clock source is selected or outputs a transmit/receive clock when internal clock source is selected. |
| | TXD0 | Transmit data | output | Transmit data output pin for serial I/O channel 0 |
| | RXD0 | Receive data | Input | Receive data input pin for serial I/O channel 0 |
| | TXD1 | Transmit data | Output | Transmit data output pin for serial I/O channel 1. |
| | RXD1 | Receive data | Input | Receive data input pin for serial I/O channel 1. |
| | TXD2 | Transmit data | Output | Transmit data output pin for serial I/O channel 2. |
| | RXD2 | Receive data | Input | Receive data input pin for serial I/O channel 2. |
| Real-time debugger | RTDTXD | Transmit data | Output | Serial data output pin for the real-time debugger. |
| | RTDRXD | Receive data | Input | Serial data input pin for the real-time debugger. |
| | RTDCLK | Clock input | Input | Serial data transmit/receive clock input pin for the real-time debugger. |
| | RTDACK | Acknowledge | Output | This pin outputs a low pulse synchronously with the beginning clock of the real-time debugger's serial data output word. The duration of this low pulse indicates the type of command/data that the real-time debugger has received. |
| Flash -only | FP | Flash Protect | Input | This mode pin has a function to protect the flash memory against E/W in hardware. |

Table 1.3.1 Description of the 32171 Pin Function (4/5)

| Type | Pin Name | Signal Name | Input/Output | Function |
|---|---|---|---|---|
| CAN | CTX | Data output | Output | This pin outputs data from the CAN module. |
| | CRX | Data input | Input | This pin is used to input data to the CAN module. |
| JTAG | JTMS | Test mode | Input | Test mode select input to control state transition of the test circuit. |
| | JTCK | clock | Input | Clock input for the debug module and test circuit. |
| | JTRST | Test reset | Input | Test reset input to initialize the test circuit asynchronously. |
| | JTDI | Serial input | Input | This pin is used to input test instruction code or test data serially. |
| | JTDO | Serial output | Output | This pin outputs test instruction code or test data serially. |
| Input/ output port (Note 1) | P00 – P07 | Input/output port 0 | Input/output | Programmable input/output port. |
| | P10 – P17 | Input/output port 1 | Input/output | Programmable input/output port. |
| | P20 – P27 | Input/output port 2 | Input/output | Programmable input/output port. |
| | P30 – P37 | Input/output port 3 | Input/output | Programmable input/output port. |
| | P41 – P47 | Input/output port 4 | Input/output | Programmable input/output port. |
| | P61 – P64 | Input/output port 6 | Input/output | Programmable input/output port.<br>(However, P64 is a $\overline{\text{SBI}}$ input-only port.) |
| | P70 – P77 | Input/output port 7 | Input/output | Programmable input/output port. |
| | P82 – P87 | Input/output port 8 | Input/output | Programmable input/output port. |
| | P93 – P97 | Input/output port 9 | Input/output | Programmable input/output port. |
| | P100 – P107 | Input/output port 10 | Input/output | Programmable input/output port. |
| | P110 – P117 | Input/output port 11 | Input/output | Programmable input/output port. |

Note 1: Input/output port 5 is reserved for future use.

### Table 1.3.1 Description of the 32171 Pin Function (5/5)

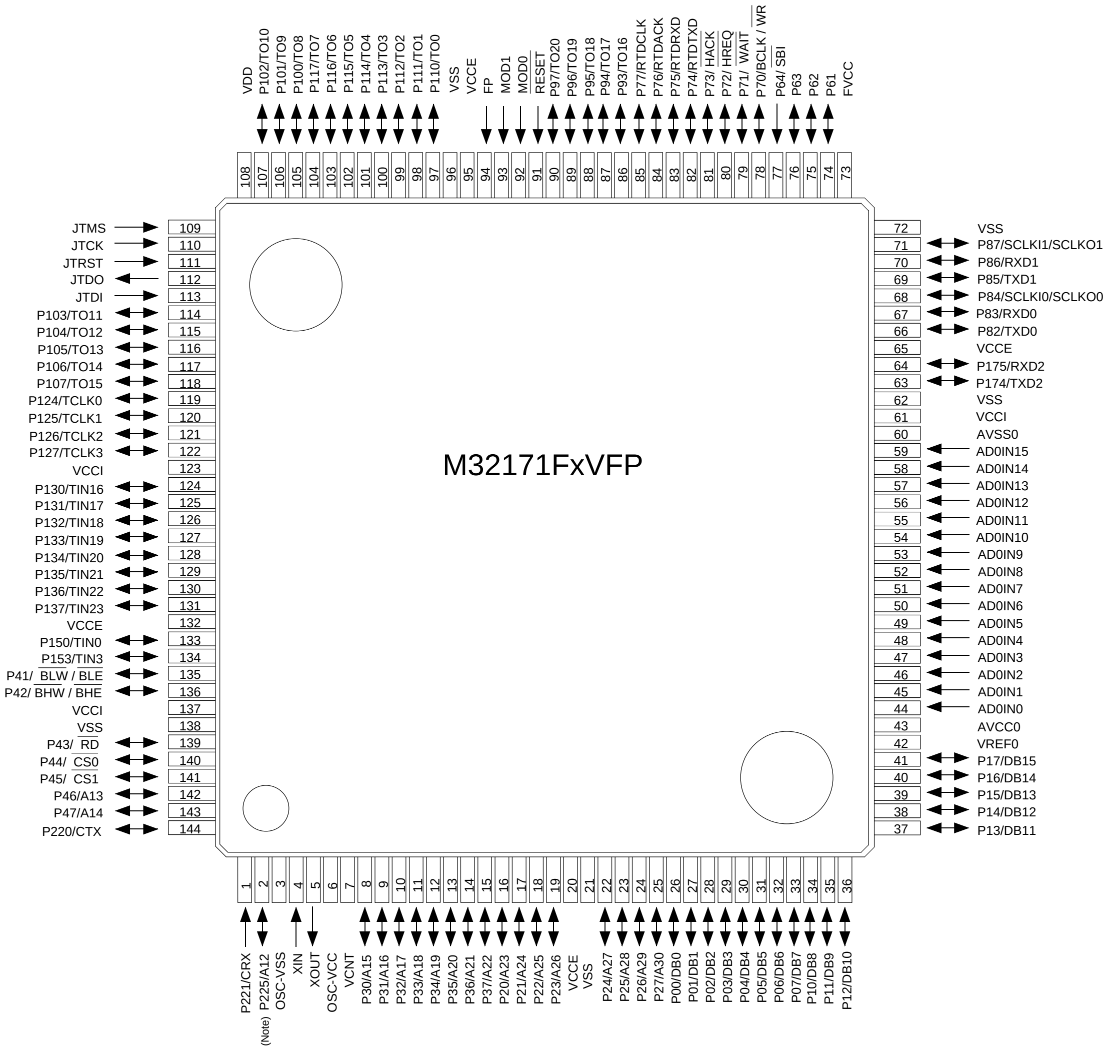
| Type | Pin Name | Signal Name | Input/Output | Function |
|---|---|---|---|---|
| Input/ output port (Note 1) | P124 – P127 | Input/output port 12 | Input/output | Programmable input/output port. |
| | P130 – P137 | Input/output port 13 | Input/output | Programmable input/output port. |
| | P150, P153 | Input/output port 15 | Input/output | Programmable input/output port. |
| | P174, P175 | Input/output port 17 | Input/output | Programmable input/output port. |
| | P220,P221 P225(Note 2) | Input/output port 22 | Input/output | Programmable input/output port.<br>(However, P221 is a CAN input only port.) |

Note 1: For the 32171, input/output ports 14, 16, 18, 19, 20, and 21 are nonexistent.

Note 2: Use caution when using P225 because they have a debug event function.

## 1.4 Pin Layout

Figure 1.4.1 shows pin assignments on the M32171FxVFP. Table 1.4.1 lists the pin assignments.



Package: 144P6Q (0.5 mm pitch)

Note: • Use caution when using these pins because they have a debug event function.

**Figure 1.4.1 Pin Layout Diagram of the M32171FxVFP (Top View)**

Table 1.4.1 Pin Assignments of the M32171FxVFP

| No. | Pin Name | No. | Pin Name | No. | Pin Name | No. | Pin Name |
|---|---|---|---|---|---|---|---|
| 1 | P221/CRX | 41 | P17 / DB15 | 81 | P73/ $\overline{\text{HACK}}$ | 121 | P126 / TCLK2 |
| 2 | P225/A12 | 42 | VREF0 | 82 | P74 / RTDTXD | 122 | P127 / TCLK3 |
| 3 | OSC-VSS | 43 | AVCC0 | 83 | P75 / RTDRXD | 123 | VCCI |
| 4 | XIN | 44 | AD0IN0 | 84 | P76 / RTDACK | 124 | P130 / TIN16 |
| 5 | XOUT | 45 | AD0IN1 | 85 | P77 / RTDCLK | 125 | P131 / TIN17 |
| 6 | OSC-VCC | 46 | AD0IN2 | 86 | P93 / TO16 | 126 | P132 / TIN18 |
| 7 | VCNT | 47 | AD0IN3 | 87 | P94 / TO17 | 127 | P133 / TIN19 |
| 8 | P30 / A15 | 48 | AD0IN4 | 88 | P95 / TO18 | 128 | P134 / TIN20 |
| 9 | P31 / A16 | 49 | AD0IN5 | 89 | P96 / TO19 | 129 | P135 / TIN21 |
| 10 | P32 / A17 | 50 | AD0IN6 | 90 | P97 / TO20 | 130 | P136 / TIN22 |
| 11 | P33 / A18 | 51 | AD0IN7 | 91 | $\overline{\text{RESET}}$ | 131 | P137 / TIN23 |
| 12 | P34 / A19 | 52 | AD0IN8 | 92 | MOD0 | 132 | VCCE |
| 13 | P35 / A20 | 53 | AD0IN9 | 93 | MOD1 | 133 | P150 / TIN0 |
| 14 | P36 / A21 | 54 | AD0IN10 | 94 | FP | 134 | P153 / TIN3 |
| 15 | P37 / A22 | 55 | AD0IN11 | 95 | VCCE | 135 | P41 / $\overline{\text{BLW}}$ / $\overline{\text{BLE}}$ |
| 16 | P20 / A23 | 56 | AD0IN12 | 96 | VSS | 136 | P42 / $\overline{\text{BHW}}$ / $\overline{\text{BHE}}$ |
| 17 | P21 / A24 | 57 | AD0IN13 | 97 | P110 / TO0 | 137 | VCCI |
| 18 | P22 / A25 | 58 | AD0IN14 | 98 | P111 / TO1 | 138 | VSS |
| 19 | P23 / A26 | 59 | AD0IN15 | 99 | P112 / TO2 | 139 | P43 / $\overline{\text{RD}}$ |
| 20 | VCCE | 60 | AVSS0 | 100 | P113 / TO3 | 140 | P44 / $\overline{\text{CS0}}$ |
| 21 | VSS | 61 | VCCI | 101 | P114 / TO4 | 141 | P45 / $\overline{\text{CS1}}$ |
| 22 | P24 / A27 | 62 | VSS | 102 | P115 / TO5 | 142 | P46 / A13 |
| 23 | P25 / A28 | 63 | P174 / TXD2 | 103 | P116 / TO6 | 143 | P47 / A14 |
| 24 | P26 / A29 | 64 | P175 / RXD2 | 104 | P117 / TO7 | 144 | P220 / CTX |
| 25 | P27 / A30 | 65 | VCCE | 105 | P100 / TO8 | | |
| 26 | P00 / DB0 | 66 | P82 / TXD0 | 106 | P101 / TO9 | | |
| 27 | P01 / DB1 | 67 | P83 / RXD0 | 107 | P102 / TO10 | | |
| 28 | P02 / DB2 | 68 | P84 / SCLKI0 / SCLKO0 | 108 | VDD | | |
| 29 | P03 / DB3 | 69 | P85 / TXD1 | 109 | JTMS | | |
| 30 | P04 / DB4 | 70 | P86 / RXD1 | 110 | JTCK | | |
| 31 | P05 / DB5 | 71 | P87 / SCLKI1 / SCLKO1 | 111 | JTRST | | |
| 32 | P06 / DB6 | 72 | VSS | 112 | JTDO | | |
| 33 | P07 / DB7 | 73 | FVCC | 113 | JTDI | | |
| 34 | P10 / DB8 | 74 | P61 | 114 | P103 / TO11 | | |
| 35 | P11 / DB9 | 75 | P62 | 115 | P104 / TO12 | | |
| 36 | P12 / DB10 | 76 | P63 | 116 | P105 / TO13 | | |
| 37 | P13 / DB11 | 77 | P64 / $\overline{\text{SBI}}$ | 117 | P106 / TO14 | | |
| 38 | P14 / DB12 | 78 | P70/ BCLK / $\overline{\text{WR}}$ | 118 | P107 / TO15 | | |
| 39 | P15 / DB13 | 79 | P71 / $\overline{\text{WAIT}}$ | 119 | P124 / TCLK0 | | |
| 40 | P16 / DB14 | 80 | P72 / $\overline{\text{HREQ}}$ | 120 | P125 / TCLK1 | | |

# CHAPTER 2
# CPU

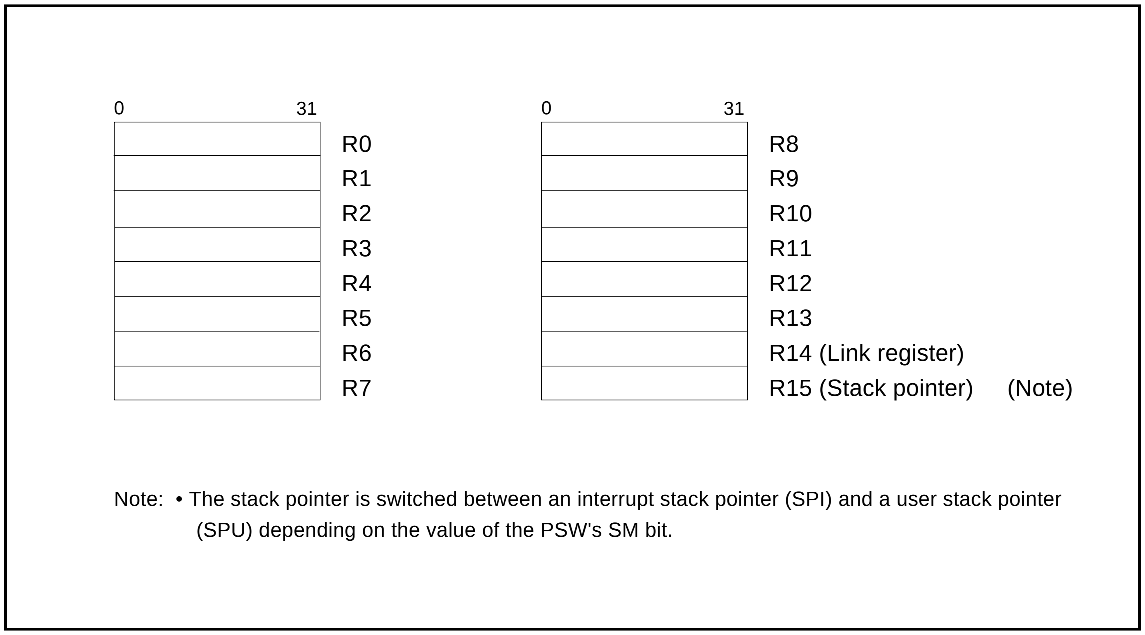
2.1 CPU Registers
2.2 General-purpose Registers
2.3 Control Registers
2.4 Accumulator
2.5 Program Counter
2.6 Data Formats
2.7 Precautions on CPU

## 2.1 CPU Registers

The M32R has sixteen general-purpose registers, five control registers, an accumulator, and a program counter. The accumulator is a 56-bit configuration, and all other registers are a 32-bit configuration.

## 2.2 General-purpose Registers

General-purpose registers are 32 bits in width and there are sixteen of them (R0 to R15), which are used to hold data and base addresses. Especially, R14 is used as a link register, and R15 is used as a stack pointer. The link register is used to store the return address when executing a subroutine call instruction. The stack pointer is switched between an interrupt stack pointer (SPI) and a user stack pointer (SPU) depending on the value of the Processor Status Word register (PSW)'s stack mode (SM) bit.

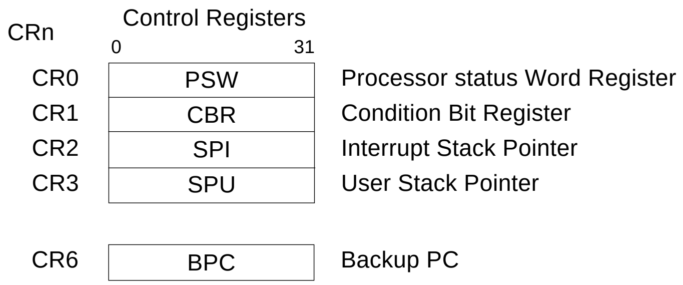


Note: • The stack pointer is switched between an interrupt stack pointer (SPI) and a user stack pointer (SPU) depending on the value of the PSW's SM bit.

Figure 2.2.1 General-purpose Registers

## 2.3 Control Registers

There are five control registers-Processor Status Word Register (PSW), Condition Bit Register (CBR), Interrupt Stack Pointer (SPI), User Stack Pointer (SPU), and Backup PC (BPC). Dedicated "MVTC" and "MVFC" instructions are used to set and read these control registers.



Notes: • CRn (n = 0-3, 6) denotes control register numbers.

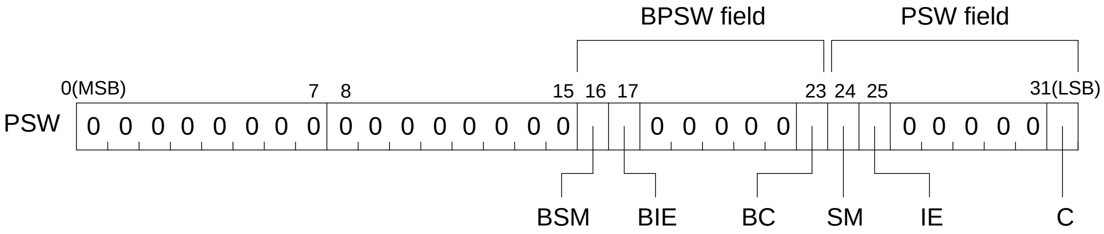
: • Dedicated "MVTC" and "MVFC" instructions are used to set and read the control registers.

**Figure 2.3.1 Control Registers**

### 2.3.1 Processor Status Word Register: PSW (CR0)

The Processor Status Word Register (PSW) is used to indicate the status of the M32R. It consists of a regularly used PSW field and a special BPSW field which is used to save the PSW field when an EIT occurs.

The PSW field consists of several bits labeled Stack Mode (SM), Interrupt Enable (IE), and Condition bit (C). The BPSW field consists of backup bits of the foregoing, i.e., Backup SM bit (BSM), Backup IE bit (BIE), and Backup C bit (BC).



(Note 1)

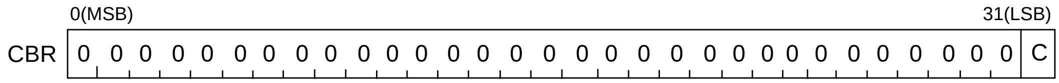
| D | Bit Name | Function | Initial | R | W |
|---|---|---|---|---|---|
| 16 | BSM (Backup SM) | Holds the value of SM bit when EIT is accepted. | Indeterminate | ○ | ○ |
| 17 | BIE (Backup IE) | Holds the value of IE bit when EIT is accepted. | Indeterminate | ○ | ○ |
| 23 | BC (Backup C) | Holds the value of C bit when EIT is accepted. | Indeterminate | ○ | ○ |
| 24 | SM (Stack Mode) | 0: Interrupt stack pointer is used.<br>1: User stack pointer is used. | 0 | ○ | ○ |
| 25 | IE (Interrupt Enable) | 0: No interrupt is accepted.<br>1: Interrupt is accepted. | 0 | ○ | ○ |
| 31 | C (Condition bit) | Depending on instruction execution, it indicates whether operation resulted in a carry, borrow, or overflow. | 0 | ○ | ○ |

Note 1: "Initial" shows the state immediately after reset, R = O means the register is readable, W = O means the register is writable.

Note: • For changes of the state of each bit when an EIT event occurs, refer to Chapter 4, "EIT."

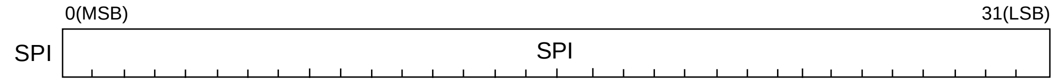
### 2.3.2 Condition Bit Register: CBR (CR1)

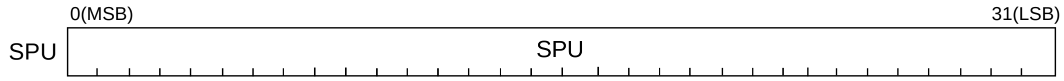
The Condition Bit Register (CBR) is created as a separate register from the PSW by extracting the Condition bit (C) from it. The value written to the PSW C bit is reflected in this register. This register is a read-only register (writes to this register by "MVTC" instruction are ignored).



### 2.3.3 Interrupt Stack Pointer: SPI (CR2)<br>User Stack Pointer: SPU (CR3)

The Interrupt Stack Pointer (SPI) and User Stack Pointer (SPU) hold the current address of the stack pointer. These registers can be accessed as general-purpose register R15. In this case, whether R15 is used as SPI or as SPU depends on the PSW's Stack Mode (SM) bit.

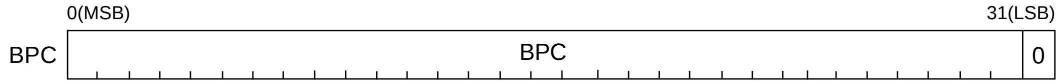




### 2.3.4 Backup PC: BPC (CR6)

The Backup PC (BPC) is a register used to save the value of the Program Counter (PC) when an EIT occurs. Bit 31 is fixed to 0.

When an EIT occurs, the value held in the PC immediately before the EIT occurred or the value of the next instruction is set in this register. When the "RTE" instruction is executed, the saved value is returned from the BPC to the PC. However, the two low-order bits of the PC when thus returned are always fixed to "00" (control always returns to word boundaries.)

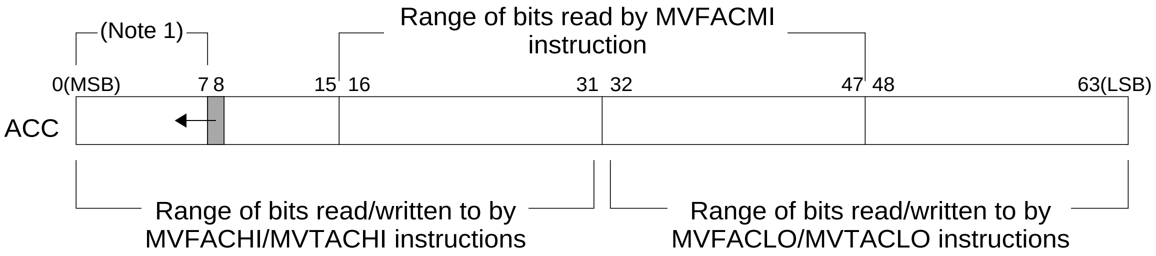


## 2.4 Accumulator

The accumulator (ACC) is a 56-bit register used by DSP function instructions. When read out or written to, it is handled as a 64-bit register. When reading, the value of bit 8 is sign-extended. When writing, bits 0--7 are ignored. Also, the accumulator is used by the multiplication instruction "MUL." Note that when executing this instruction, the value of the accumulator is destroyed.

The "MVTACHI" and "MVTACLO" instructions are used to write to the accumulator. The "MVTACHI" instruction writes data to the 32 high-order bits (bits 0-31), and the "MVTACLO" instruction writes data to the 32 low-order bits (bits 32-63).

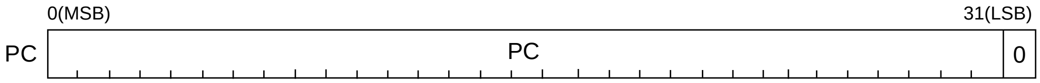
The "MVFACHI," "MVFACLO," and "MVFACMI" instructions are used to read data from the accumulator. The "MVFACHI" instruction reads data from the 32 high-order bits (bits 0-31), the "MVFACLO" instruction reads data from the 32 low-order bits (bits 32-63), and the "MVFACHI" instruction reads data from the 32 middle bits (bits 16-47).


(Note 1)
Range of bits read by MVFACMI instruction
0(MSB)
7
8
15
16
31
32
47
48
63(LSB)
ACC
Range of bits read/written to by MVFACHI/MVTACHI instructions
Range of bits read/written to by MVFACLO/MVTACLO instructions

Note 1: Bits 0-7 always show the sign-extended value of bit 8. Writes to this bit field are ignored.

## 2.5 Program Counter

The Program Counter (PC) is a 32-bit counter used to hold the address of the currently executed instruction. Because M32R instructions each start from an even address, the LSB (bit 31) is always 0.



## 2.6 Data Formats

### 2.6.1 Data Types

There are several data types that can be handled by the M32R's instruction set. These include signed and unsigned 8, 16, and 32-bit integers. Values of signed integers are represented by 2's complements.

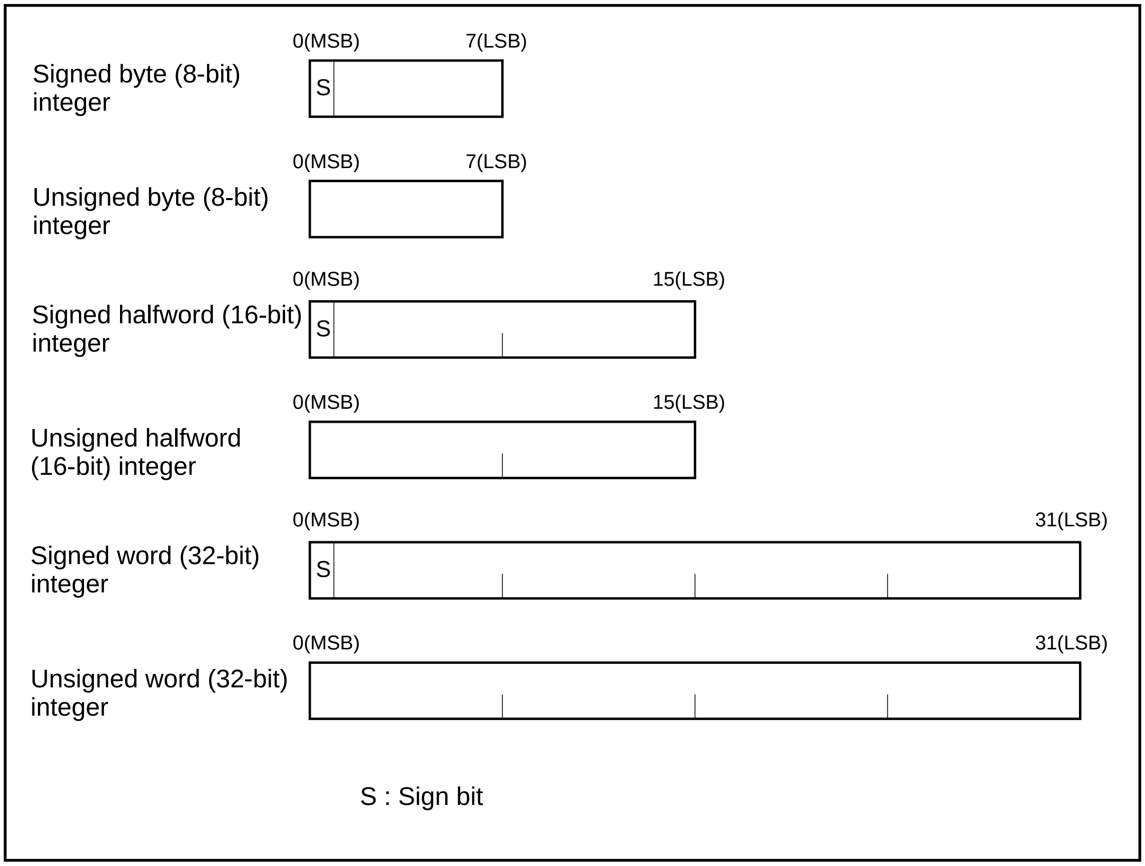

Signed byte (8-bit) integer
0(MSB)
7(LSB)
S
Unsigned byte (8-bit) integer
0(MSB)
7(LSB)
Signed halfword (16-bit) integer
0(MSB)
15(LSB)
S
Unsigned halfword (16-bit) integer
0(MSB)
15(LSB)
Signed word (32-bit) integer
0(MSB)
31(LSB)
S
Unsigned word (32-bit) integer
0(MSB)
31(LSB)
S : Sign bit

**Figure 2.6.1 Data Types**

### 2.6.2 Data Formats

**(1) Data formats in register**

Data sizes in M32R registers are always words (32 bits).

When loading byte (8-bit) or halfword (16-bit) data from memory into a register, the data is sign-extended (LDB, LDH instructions) or zero-extended (LDUB, LDUH instructions) into word (32-bit) data before being stored in the register. When storing data from M32R register into memory, the register data is stored in memory in different sizes depending on the instructions used. The ST instruction stores the entire 32-bit data of the register, the STH instruction stores the least significant 16-bit data, and the STB instruction stores the least significant 8-bit data.

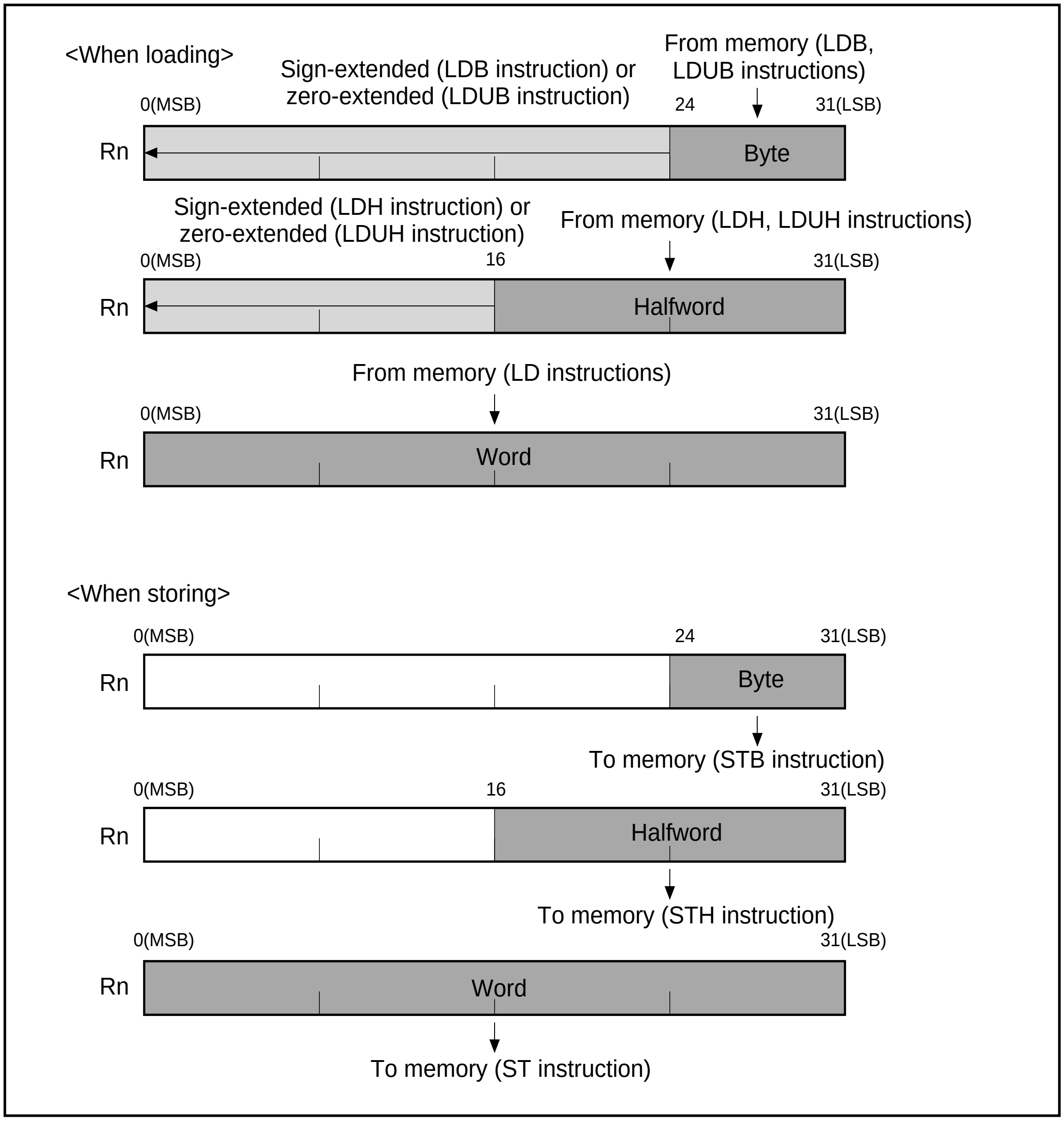

<When loading>
From memory (LDB, LDUB instructions)
Sign-extended (LDB instruction) or zero-extended (LDUB instruction)
0(MSB)
24
31(LSB)
Rn
Byte
Sign-extended (LDH instruction) or zero-extended (LDUH instruction)
From memory (LDH, LDUH instructions)
0(MSB)
16
31(LSB)
Rn
Halfword
From memory (LD instructions)
0(MSB)
31(LSB)
Rn
Word
<When storing>
0(MSB)
24
31(LSB)
Rn
Byte
To memory (STB instruction)
0(MSB)
16
31(LSB)
Rn
Halfword
To memory (STH instruction)
0(MSB)
31(LSB)
Rn
Word
To memory (ST instruction)

**Figure 2.6.2 Data Formats in Register**

### (2) Data formats in memory

Data sizes in memory are either byte (8 bits), halfword (16 bits), or word (32 bits). Byte data can be located at any address. However, halfword data must be located at halfword boundaries (where the LSB address bit = "0"), and word data must be located at word boundaries (where two LSB address bits = "00"). If an attempt is made to access memory data across these halfword or word boundaries, an address exception is generated.

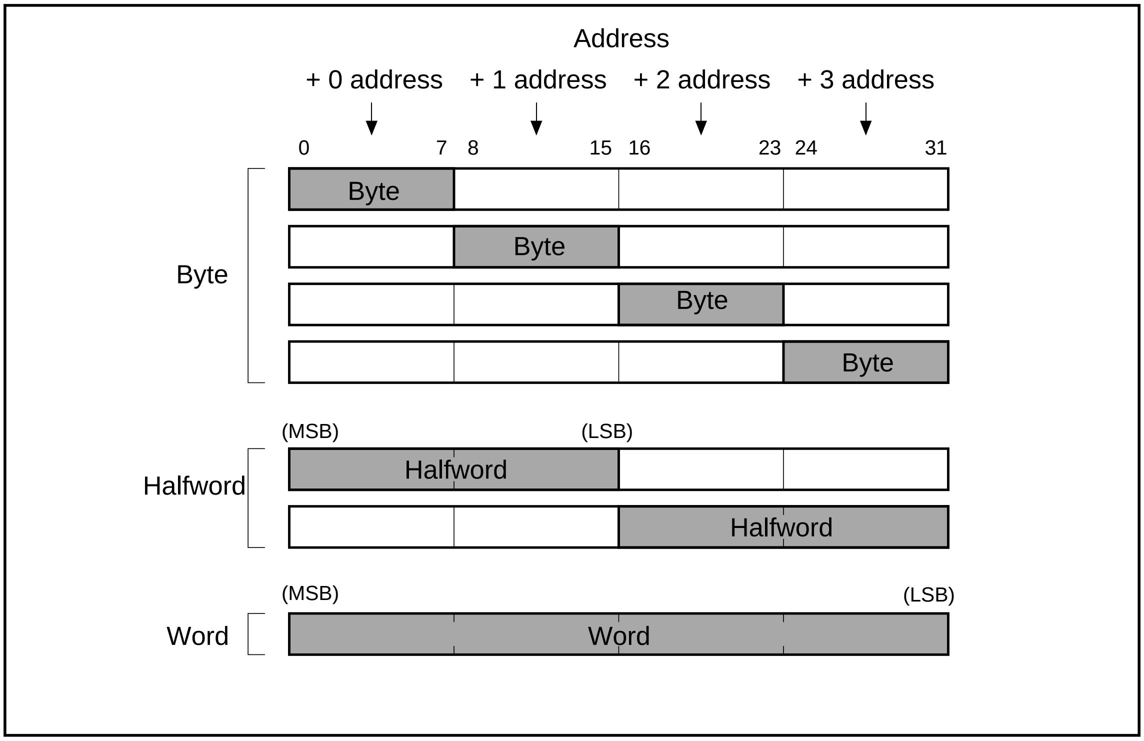


Figure 2.6.3 Data Formats in Memory

### (3) Endian

The following shows the generally used endian methods and the M32R family endian.

| | Bit endian<br>(H'01) | Byte endian<br>(H'01234567) |
|---|---|---|
| Big endian | MSB LSB<br>B'0000001<br>D0 D7 | MSB LSB<br>H'01 H'23 H'45 H'67<br>HH HL LH LL |
| Little endian | MSB LSB<br>B'0000001<br>D7 D0 | MSB LSB<br>H'67 H'45 H'23 H'01<br>LL LH HL HH |

Note: • Even for bit big endian, H'01 is not B'10000000.

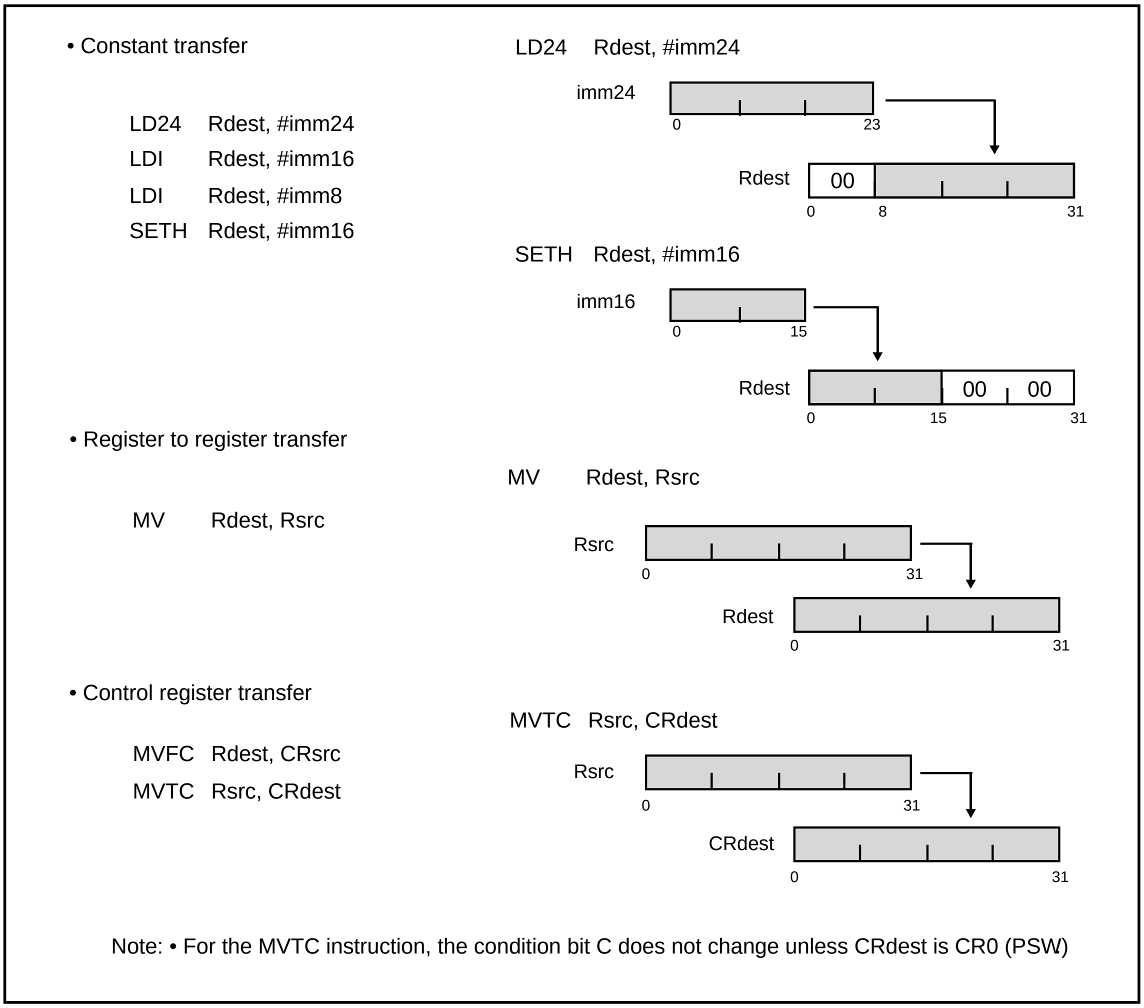
**Figure 2.6.4 Endian Methods**

| MPU name | 7700 family<br>M16C family | Competition | M32R family<br>M16 family |
|---|---|---|---|
| Endian<br>(Bit/Byte) | Little/Little | Little/Big | Big/Big |
| Address | +0 +1 +2 +3 | +0 +1 +2 +3 | +0 +1 +2 +3 |
| Data<br>arrangement | MSB L SB<br>LL LH HL HH | MSB L SB<br>HH HL LH LL | MSB L SB<br>HH HL LH LL |
| Bit number | 31-24 23-16 15-8 7-0 | 31-24 23-16 15-8 7-0 | -7 8-15 16-23 24-31 |
| Ex:0x01234567 | .byte 67,45,23,01 | .byte 01,23,45,67 | .byte 01,23,45,67 |

Note: • The M32R's endian method is big endian for both bit and byte

**Figure 2.6.5 M32R Family Endian**

### (4) Transfer instructions



- Constant transfer

| | |
|---|---|
| LD24 | Rdest, #imm24 |
| LDI | Rdest, #imm16 |
| LDI | Rdest, #imm8 |
| SETH | Rdest, #imm16 |

- Register to register transfer

| | |
|---|---|
| MV | Rdest, Rsrc |

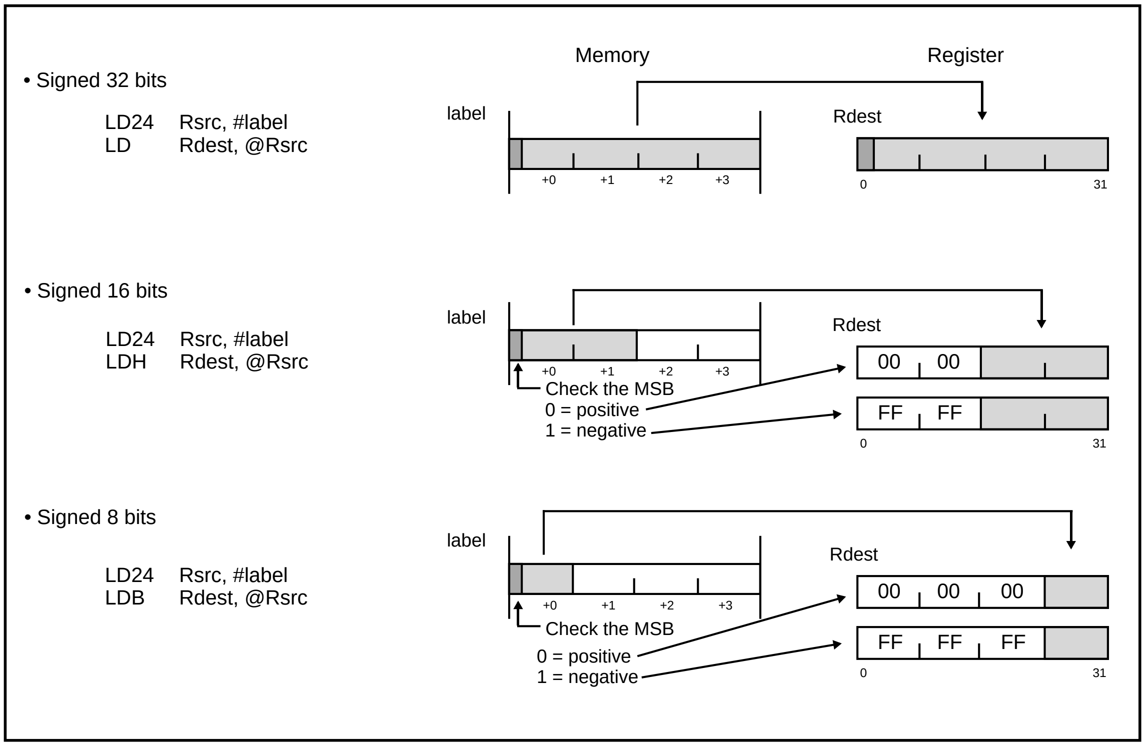
- Control register transfer

| | |
|---|---|
| MVFC | Rdest, CRsrc |
| MVTC | Rsrc, CRdest |

Note: • For the MVTC instruction, the condition bit C does not change unless CRdest is CR0 (PSW)

**Figure 2.6.6 Transfer instructions**

### (5) Memory (signed) to register transfer



- Signed 32 bits

```
LD24   Rsrc, #label
LD     Rdest, @Rsrc
```

- Signed 16 bits

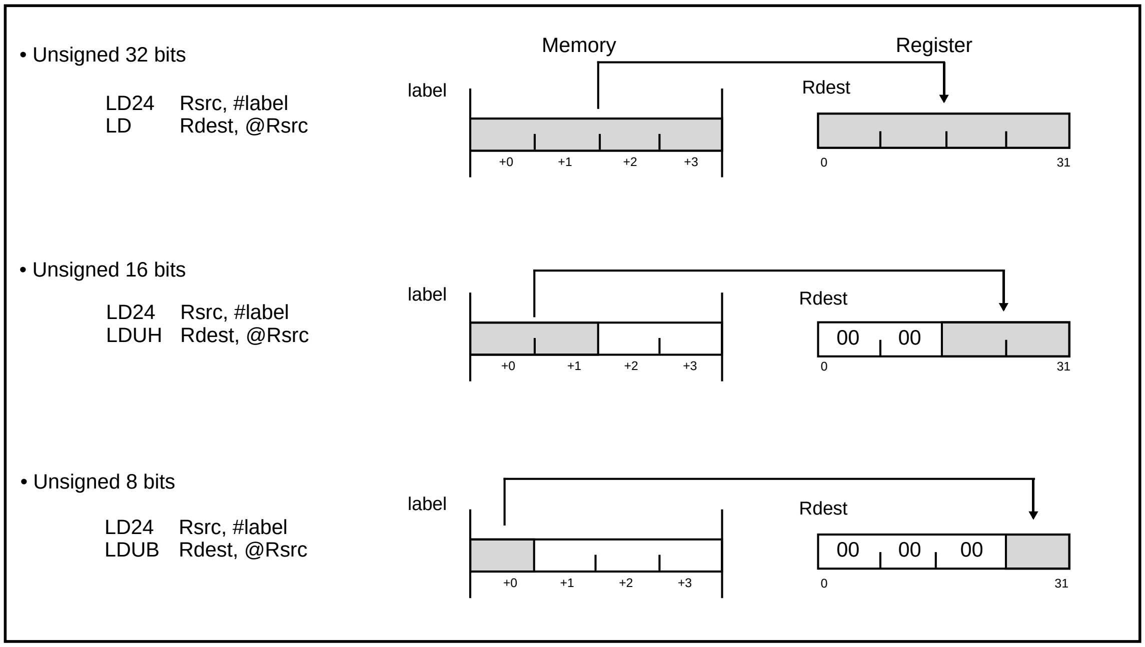
```
LD24   Rsrc, #label
LDH    Rdest, @Rsrc
```

- Signed 8 bits

```
LD24   Rsrc, #label
LDB    Rdest, @Rsrc
```

**Figure 2.6.7 Memory (signed) to register transfer**

### (6) Memory (unsigned) to register transfer



- Unsigned 32 bits

```
LD24   Rsrc, #label
LD     Rdest, @Rsrc
```

- Unsigned 16 bits

```
LD24   Rsrc, #label
LDUH   Rdest, @Rsrc
```

- Unsigned 8 bits

```
LD24   Rsrc, #label
LDUB   Rdest, @Rsrc
```

**Figure 2.6.8 Memory (unsigned) to register transfer**

### (7) Things to be noted for data transfer

Note that in data transfer, data arrangements in registers and those in memory are different.

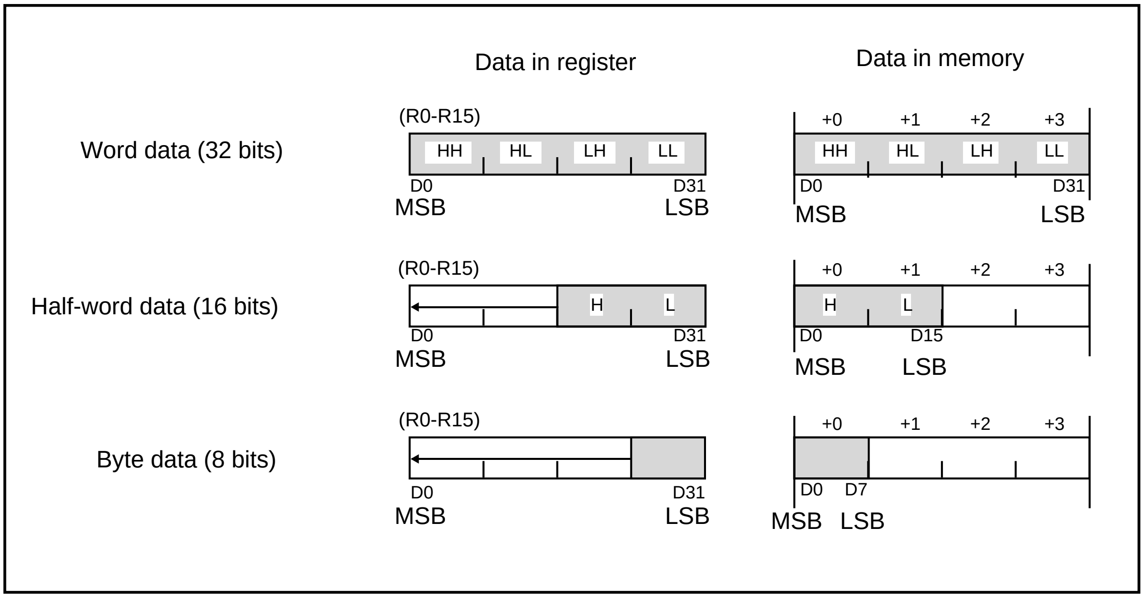


**Figure 2.6.9 Difference in Data Arrangements**

## 2.7 Precautions on CPU

### • Usage Notes for 0 Division Instruction

Problem and Conditions

Inaccurate calculations for the instructions listed in (2) will result from execution of the 0 division instruction under the conditions described in (1).

(1) If 0 division calculation is executed when the divisor = 0 for instructions DIV, DIVU, REM and REMU,

(2) the result will be inaccurate calculations for any of the following instructions that are executed immediately after 0 division:

ADDV, ADDX, ADD, ADDI, ADDV3, ADD3,
CMP, CMPU, CMPI, CMPUI,
SUBV, SUBX, SUB,
DIV, DIVU,
REM, REMU.

Countermeasure

Assuming that the 0 division occurrence itself is not expected by the system and therefore is the cause of miscalculations, before executing division or remainder instructions, do a 0 check on the divisor to make sure 0 division does not occur.

# CHAPTER 3
# ADDRESS SPACE

3.1 Outline of Address Space
3.2 Operation Modes
3.3 Internal ROM Area and External Extension Area
3.4 Internal RAM Area and SFR Area
3.5 EIT Vector Entry
3.6 ICU Vector Table
3.7 Notes on Address Space

## 3.1 Outline of Address Space

The M32R's logical addresses are always handled in 32 bits, providing 4 Gbytes of linear address space. The M32R/E's address space consists of the following:

(1) User space

- Internal ROM area
- External extension area
- Internal RAM area
- Special Function Register (SFR) area

(2) Boot program space

(3) System space (areas not open to the user)

### (1) User space

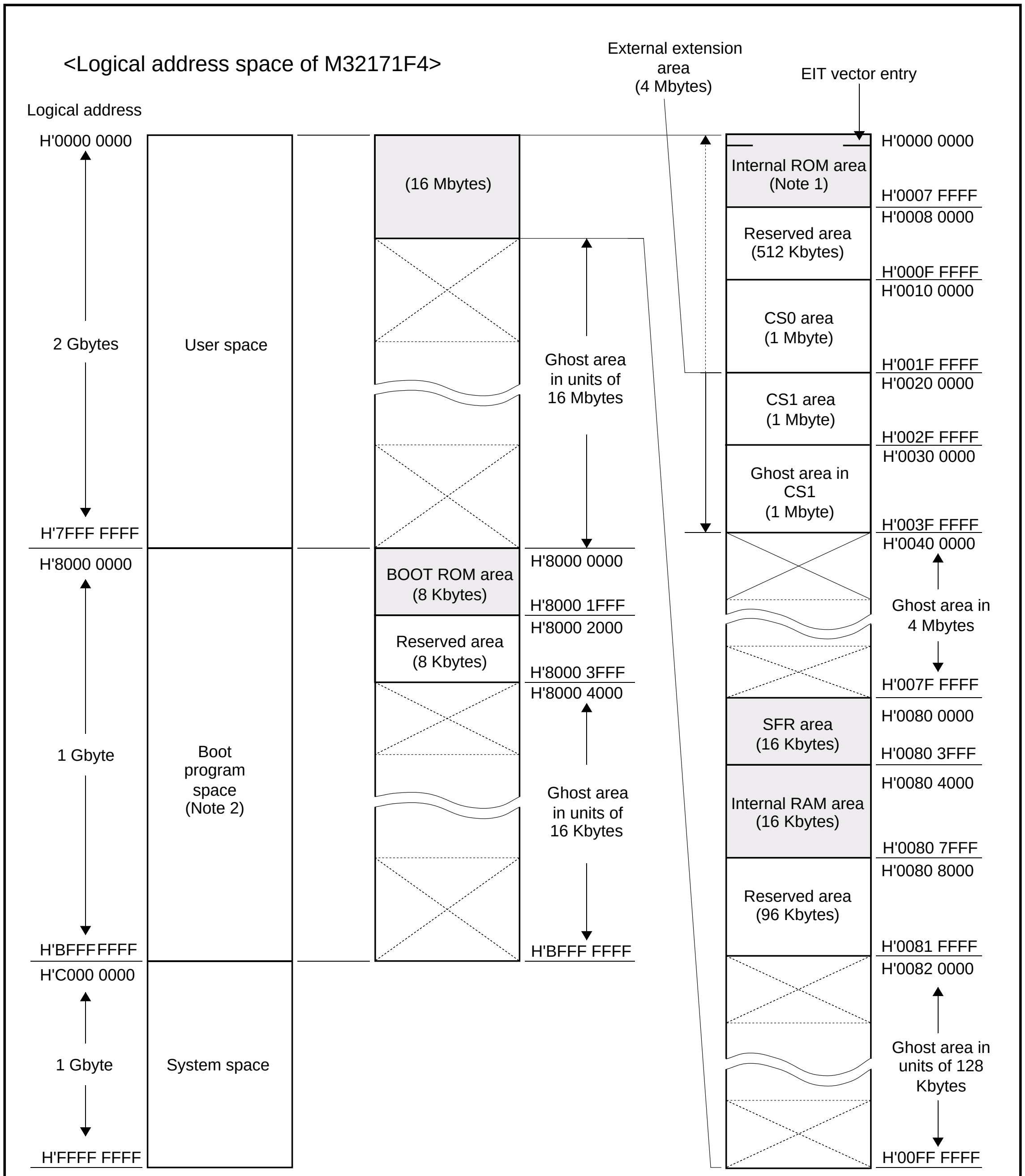
A 2 Gbytes of address space from H'0000 0000 to H'7FFF FFFF is the user space. Located in this space are the internal ROM area, external extension area, internal RAM area, and Special Function Register (SFR) area, an area containing a group of internal peripheral I/O registers. Of these, the internal ROM and external extension areas are located differently depending on mode settings which will be described later.

### (2) Boot program space

A 1 Gbyte of address space from H'8000 0000 to H'BFFF FFFF is the boot program space. This space stores a program (boot program) which enables on-board programming when the internal flash area is blank.

### (3) System space

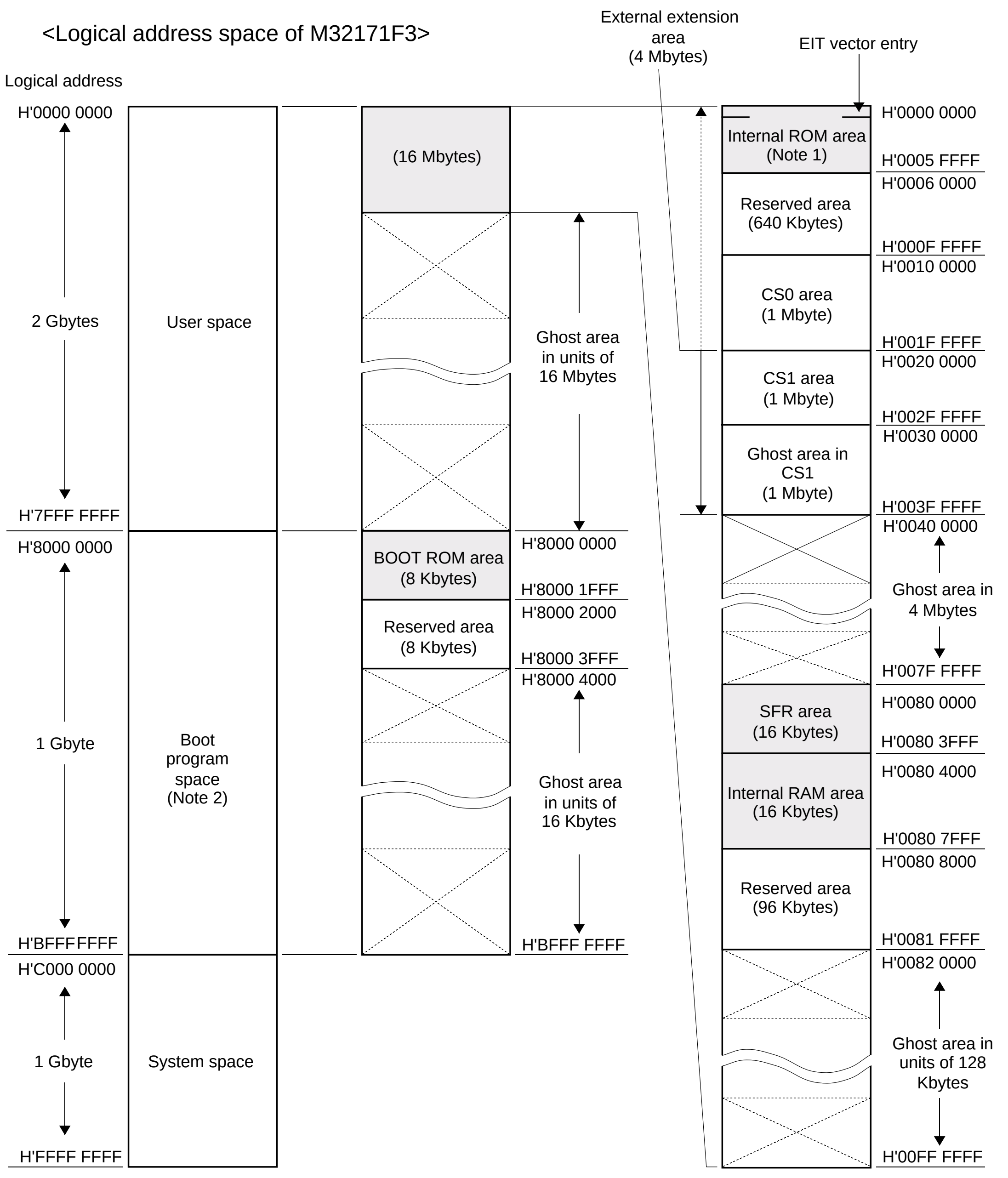
A 1 Gbyte of address space from H'C000 0000 to H'FFFF FFFF is the system space. This space is reserved for use by development tools such as an in-circuit emulator or a debug monitor, and cannot be used by the user.



Note 1: This location varies with chip mode settings.

Note 2: The boot program space can read out only when FP = 1, MOD0 = 1, and MOD1 = 0.

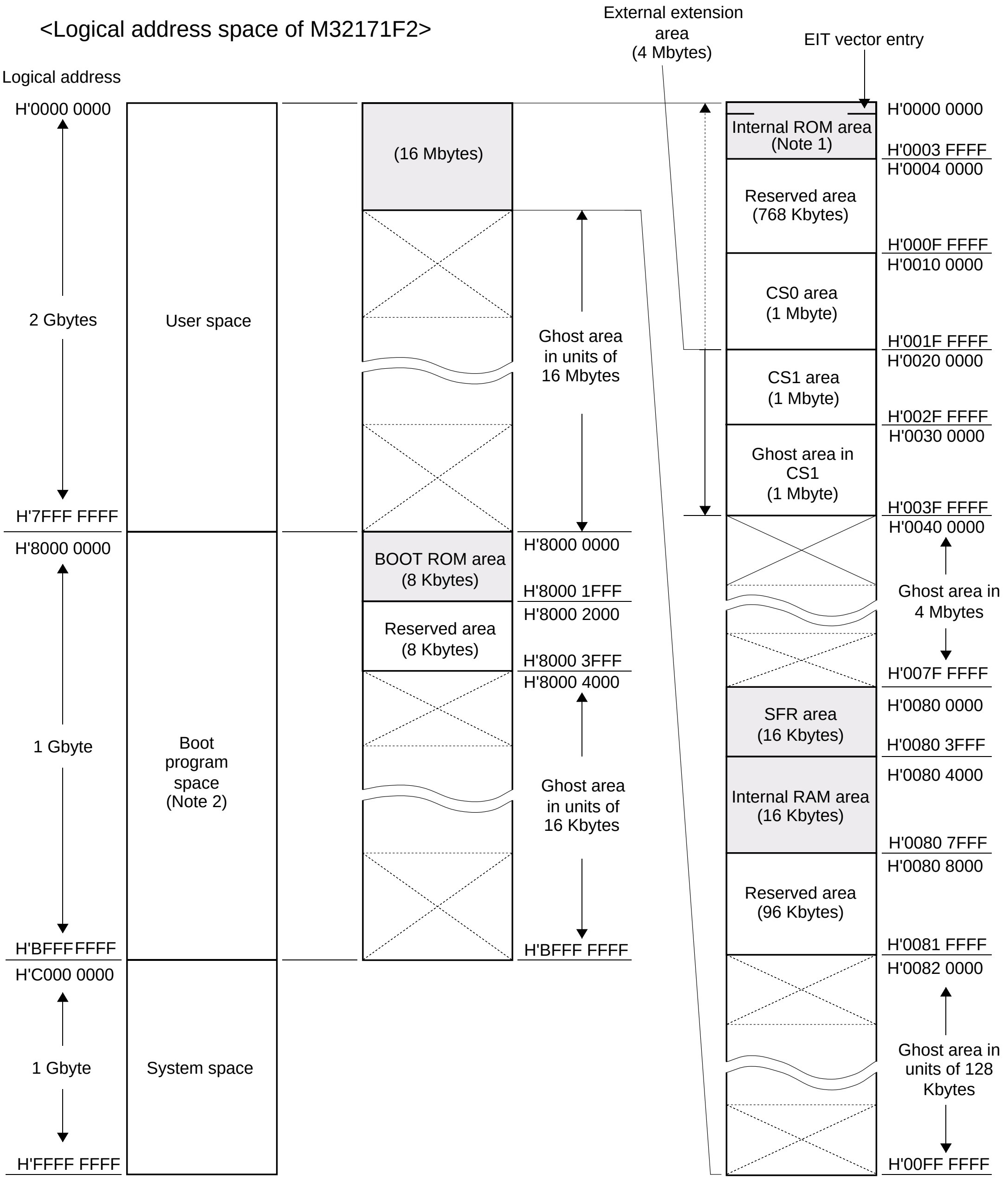
**Figure 3.1.1 Address Space of the M32171F4**



Note 1: This location varies with chip mode settings.

Note 2: The boot program space can read out only when FP = 1, MOD0 = 1, and MOD1 = 0.

**Figure 3.1.2 Address Space of the M32171F3**



Note 1: This location varies with chip mode settings.

Note 2: The boot program space can read out only when FP = 1, MOD0 = 1, and MOD1 = 0.

**Figure 3.1.3 Address Space of the M32171F2**

## 3.2 Operation Modes

The 32171 is placed in one of the following modes by setting its operation mode (using MOD0 and MOD1 pins). For details about the mode used to rewrite the internal flash memory, refer to Section 6.5, "Programming of Internal Flash Memory."

**Table 3.2.1 Setting Operation Modes**

| MOD0 | MOD1 (Note 1) | Operation Mode (Note 2) |
|---|---|---|
| VSS | VSS | Single-chip mode |
| VSS | VCCE | External extension mode |
| VCCE | VSS | Processor mode (FP = VSS) |
| VCCE | VCCE | Reserved (cannot be used) |

Note 1: VCCE connects to +5 V or +3.3 V, and VSS connects to GND.

Note 2: For flash rewrite mode (FP = VCCE) not listed in the above table, refer to Section 6.5, "Programming of Internal Flash Memory."

The internal ROM and external extension areas are located differently depending on the 32171's operation mode. (All other areas in address space are located the same way.) The address maps of internal ROM and external extension areas in each mode are shown below. (For flash rewrite mode (FP = VCCE) not listed in the above table, refer to Section 6.5, "Programming of Internal Flash Memory.")

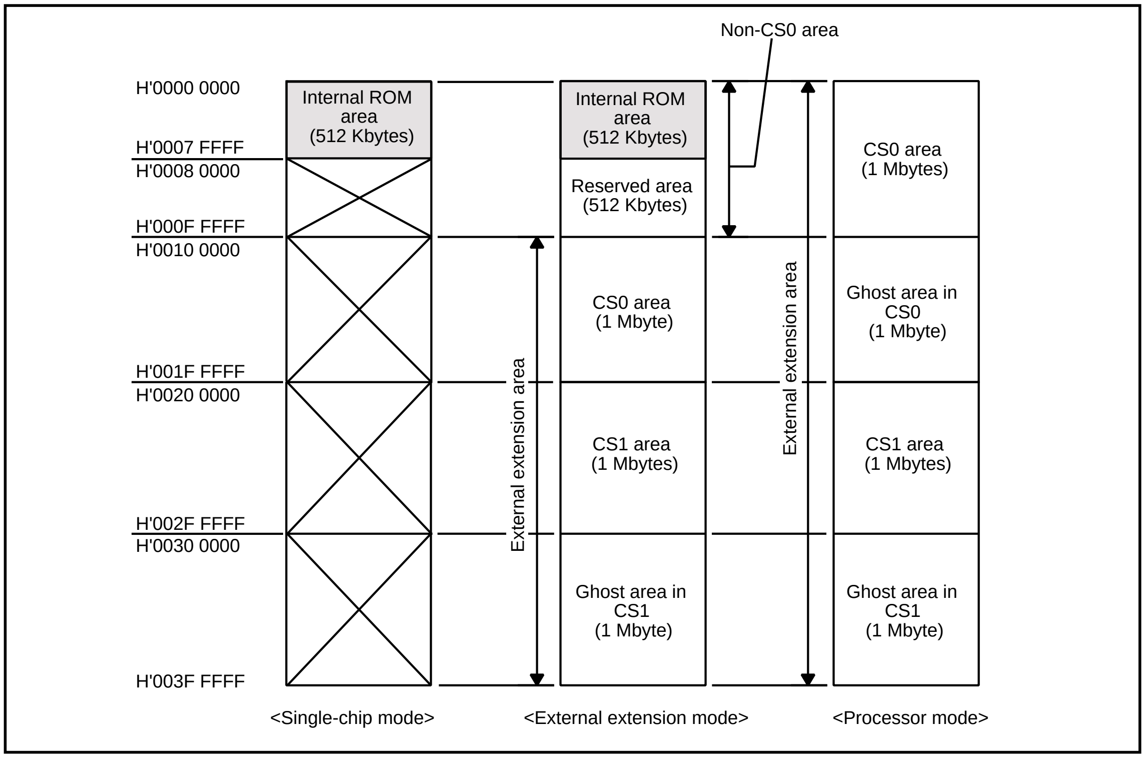


**Figure 3.2.1 M32171F4 Operation Mode and Internal ROM/External Extension Areas**

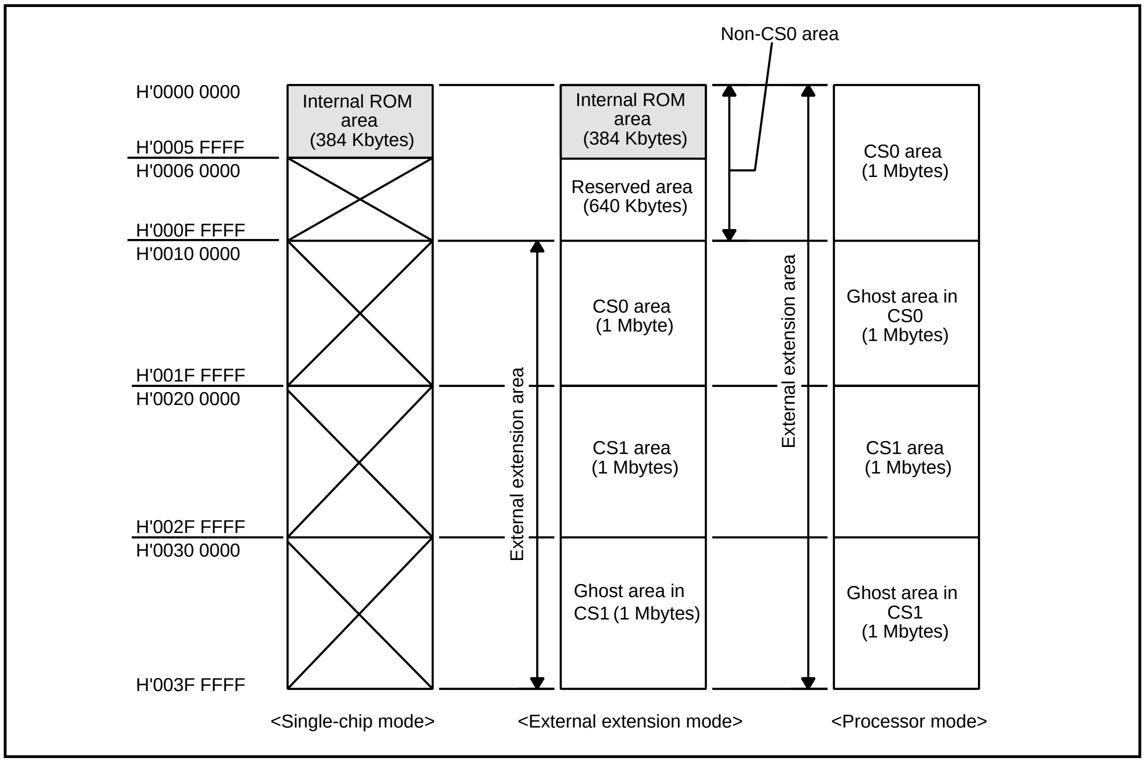


**Figure 3.2.2 M32171F3 Operation Mode and Internal ROM/External extension Areas**

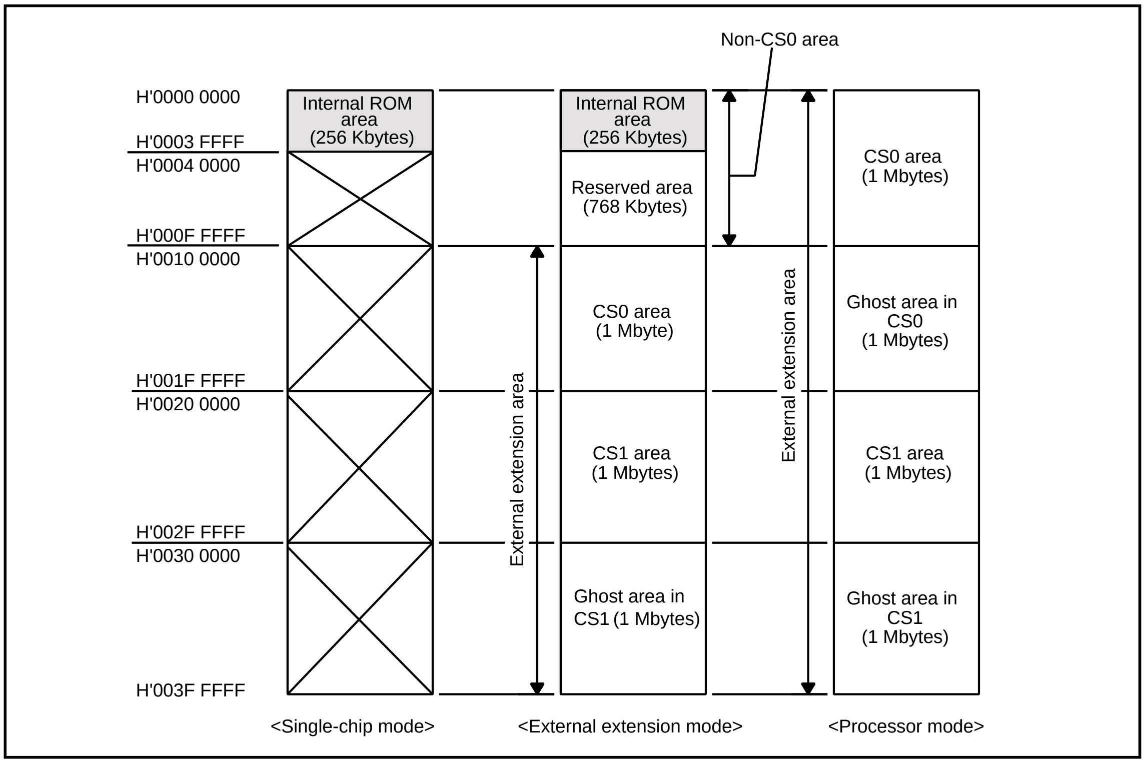


**Figure 3.2.3 M32171F2 Operation Mode and Internal ROM/External extension Areas**

## 3.3 Internal ROM Area and External Extension Area

The 8 Mbyte area at addresses H'0000 0000 to H'007F FFFF in the user space accommodates the internal ROM and external extension areas. Of this, a 4 Mbytes of address space from H'0000 0000 to H'003F FFFF is the area that the user can actually use. All other areas here comprise a 4 Mbytes of ghost area. (When programming, do not use this ghost area intentionally.)
For details on how the internal ROM and external extension areas are located differently depending on the 32171's operation modes set, refer to Section 3.2, "Operation Modes."

### 3.3.1 Internal ROM Area

The internal ROM is located in the area shown below. Also, this area has an EIT vector entry (and ICU vector table) located in it at the beginning.

**Table 3.3.1 Addresses at Which the 32171's Internal ROM is Located**

| Type Name | Size | Located address |
|---|---|---|
| M32171F4 | 512 Kbytes | H'0000 0000 - H'0007 FFFF |
| M32171F3 | 384 Kbytes | H'0000 0000 - H'0005 FFFF |
| M32171F2 | 256 Kbytes | H'0000 0000 - H'0003 FFFF |

### 3.3.2 External Extension Area

An external extension area is provided only when external extension mode or processor mode has been selected when setting the 32171's operation mode. For access to this external extension area, the 32171 outputs the control signals necessary to access external devices.
The 32171's $\overline{\text{CS0}}$ and $\overline{\text{CS1}}$ signals are output corresponding to the address mapping of the external extension area. The $\overline{\text{CS0}}$ signal is output for the CS0 area, and the $\overline{\text{CS1}}$ signal is output for the CS1 area.

**Table 3.3.2 Address Mapping of the External Extension Area in Each Operation Mode of the 32171**

| Operation Mode | Address mapping of the external extension area |
|---|---|
| Single-chip mode | None |
| External extension mode | Addresses H'0010 0000 to H'001F FFFF (CS0 area: 1 Mbytes)<br>Addresses H'0020 0000 to H'002F FFFF (CS1 area: 1 Mbytes) (Note 1) |
| Processor mode | Addresses H'0000 0000 to H'000F FFFF (CS0 area: 1 Mbytes) (Note 2)<br>Addresses H'0020 0000 to H'002F FFFF (CS1 area: 1 Mbytes) (Note 2) |

Note 1: During external extension mode, a ghost (1 Mbyte) of the CS1 area appears in an area of H’0030 0000 through H’003F FFFF.

Note 2: During processor mode, a ghost (1 Mbyte) of the CS0 area appears in an area of H’0010 0000 through H’001F FFFF and a ghost (1 Mbyte) of the CS1 area appears in an area of H’0030 0000 through H’003F FFFF.

## 3.4 Internal RAM Area and SFR Area

The 8 Mbyte area at addresses H'0080 0000 to H'00FF FFFF in the user space accommodates the internal RAM area and Special Function Register (SFR) area. Of this, a 128 Kbytes of address space from H'0080 0000 to H'0081 FFFF is the area that the user can actually use. All other areas here comprise a ghost area in units of 128 Kbytes. (When programming, do not use this ghost area intentionally.)

### 3.4.1 Internal RAM Area

The internal RAM (16-Kbyte) is allocated to the addresses H’0080 4000 through H’0080 7FFF.

### 3.4.2 Special Function Register (SFR) Area

Addresses H'0080 0000 to H'0080 3FFFF are the Special Function Register (SFR) area. This area has registers for internal peripheral I/O located in it.

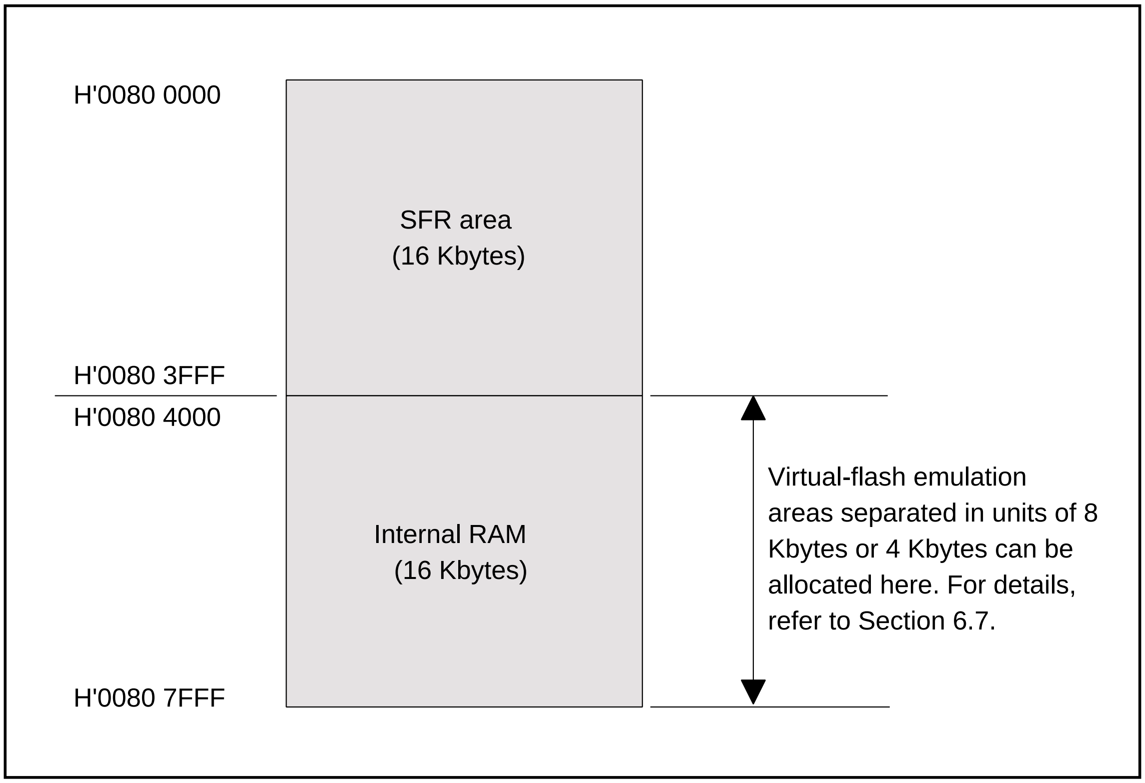
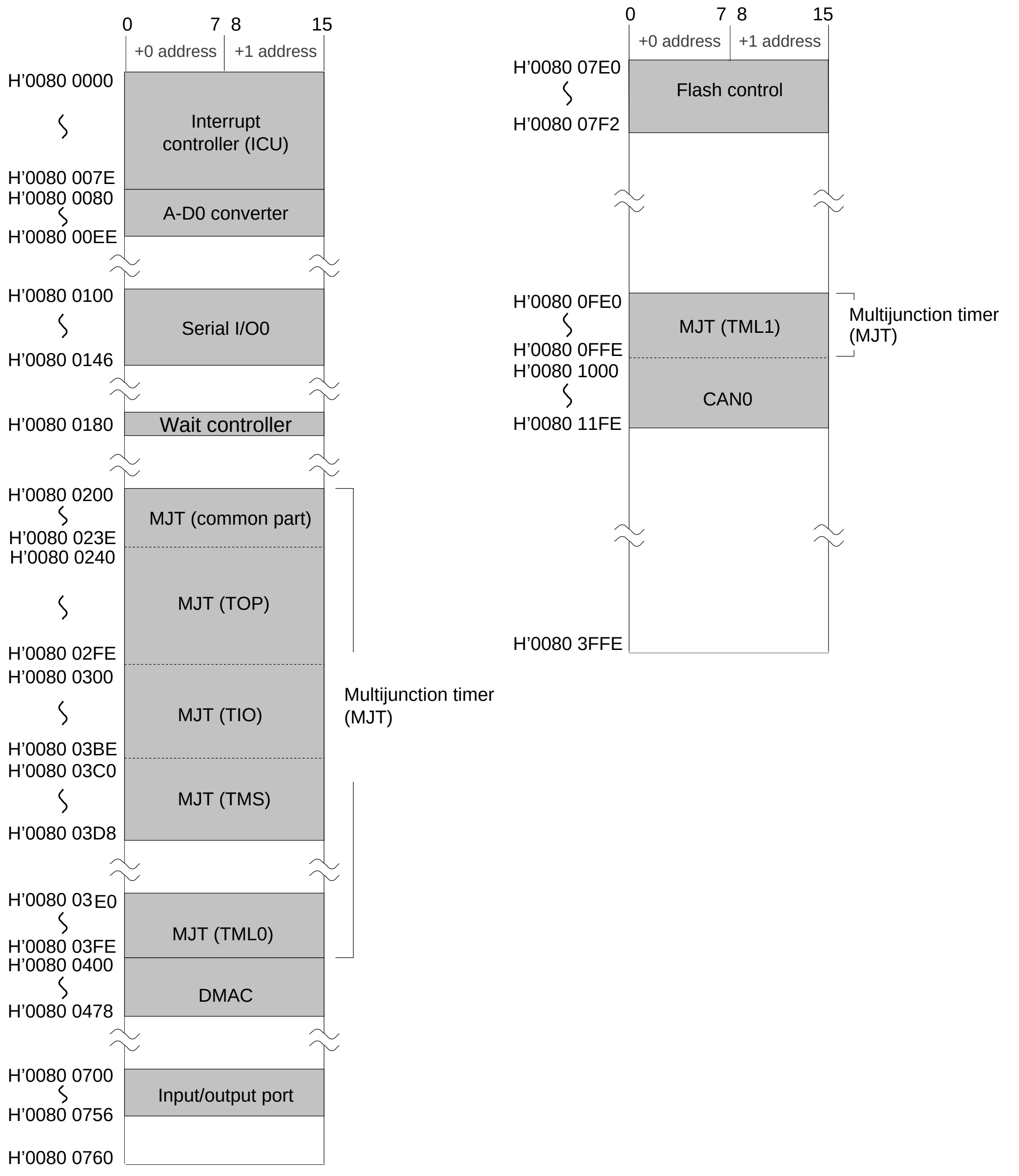

H'0080 0000
SFR area
(16 Kbytes)
H'0080 3FFF
H'0080 4000
Internal RAM
(16 Kbytes)
H'0080 7FFF
Virtual-flash emulation areas separated in units of 8 Kbytes or 4 Kbytes can be allocated here. For details, refer to Section 6.7.

**Figure 3.4.1 Internal RAM Area and Special Function Register (SFR) Area**



Note: • The Real-time Debugger (RTD) is designed to be an independent module operated from an external source, and is transparent to the CPU.

**Figure 3.4.2 Outline Address Mapping of the SFR Area**

| Address | +0 Address<br>D0 … D7 | +1 Address<br>D8 … D15 |
|---|---|---|
| H'0080 0000 | Interrupt Vector Register (IVECT) | |
| H'0080 0002 | | |
| H'0080 0004 | Interrupt Mask Register (IMASK) | |
| H'0080 0006 | SBI Control Register (SBICR) | |
| ≈ | | |
| H'0080 0060 | CAN0 Transmit/Receive & Error Interrupt Control Register (ICAN0CR) | |
| H'0080 0062 | | |
| H'0080 0064 | | |
| H'0080 0066 | | RTD Interrupt Control Register (IRTDCR) |
| H'0080 0068 | SIO2,3 Transmit/Receive Interrupt Control Register (ISO23CR) | DMA5-9 Interrupt Control Register (IDMA59CR) |
| H'0080 006A | | |
| H'0080 006C | A-D0 Conversion Interrupt Control Register (IAD0CCR) | SIO0 Transmit Interrupt Control Register (ISIO0TXCR) |
| H'0080 006E | SIO0 Receive Interrupt Control Register (ISIO0RXCR) | SIO1 Transmit Interrupt Control Register (ISIO1TXCR) |
| H'0080 0070 | SIO1 Receive Interrupt Control Register (ISIO1RXCR) | DMA0-4 Interrupt Control Register (IDMA04CR) |
| H'0080 0072 | MJT Output Interrupt Control Register 0 (IMJTOCR0) | MJT Output Interrupt Control Register 1 (IMJTOCR1) |
| H'0080 0074 | MJT Output Interrupt Control Register 2 (IMJTOCR2) | MJT Output Interrupt Control Register 3 (IMJTOCR3) |
| H'0080 0076 | MJT Output Interrupt Control Register 4 (IMJTOCR4) | MJT Output Interrupt Control Register 5 (IMJTOCR5) |
| H'0080 0078 | MJT Output Interrupt Control Register 6 (IMJTOCR6) | MJT Output Interrupt Control Register 7 (IMJTOCR7) |
| H'0080 007A | | MJT Input Interrupt Control Register 1 (IMJTICR1) |
| H'0080 007C | MJT Input Interrupt Control Register 2 (IMJTICR2) | MJT Input Interrupt Control Register 3 (IMJTICR3) |
| H'0080 007E | MJT Input Interrupt Control Register 4 (IMJTICR4) | |
| H'0080 0080 | A-D0 Single Mode Register 0 (AD0SIM0) | A-D0 Single Mode Register 1 (AD0SIM1) |
| H'0080 0082 | | |
| H'0080 0084 | A-D0 Scan Mode Register 0 (AD0SCM0) | A-D0 Scan Mode Register 1 (AD0SCM1) |
| H'0080 0086 | | |
| H'0080 0088 | A-D0 Successive Approximation Register (AD0SAR) | |
| H'0080 008A | | |
| H'0080 008C | A-D0 Comparate Data Register (AD0CMP) | |
| ≈ | | |
| H'0080 0090 | 10-bit A-D0 Data Register 0 (AD0DT0) | |
| H'0080 0092 | 10-bit A-D0 Data Register 1 (AD0DT1) | |
| H'0080 0094 | 10-bit A-D0 Data Register 2 (AD0DT2) | |
| H'0080 0096 | 10-bit A-D0 Data Register 3 (AD0DT3) | |
| H'0080 0098 | 10-bit A-D0 Data Register 4 (AD0DT4) | |
| H'0080 009A | 10-bit A-D0 Data Register 5 (AD0DT5) | |
| H'0080 009C | 10-bit A-D0 Data Register 6 (AD0DT6) | |
| H'0080 009E | 10-bit A-D0 Data Register 7 (AD0DT7) | |
| H'0080 00A0 | 10-bit A-D0 Data Register 8 (AD0DT8) | |
| H'0080 00A2 | 10-bit A-D0 Data Register 9 (AD0DT9) | |
| H'0080 00A4 | 10-bit A-D0 Data Register 10 (AD0DT10) | |
| H'0080 00A6 | 10-bit A-D0 Data Register 11 (AD0DT11) | |
| H'0080 00A8 | 10-bit A-D0 Data Register 12 (AD0DT12) | |
| H'0080 00AA | 10-bit A-D0 Data Register 13 (AD0DT13) | |
| H'0080 00AC | 10-bit A-D0 Data Register 14 (AD0DT14) | |
| H'0080 00AE | 10-bit A-D0 Data Register 15 (AD0DT15) | |
| ≈ | | |
| H'0080 00D0 | | 8-bit A-D0 Data Register 0 (AD08DT0) |

Blank addresses are reserved areas.

**Figure 3.4.3 Register Mapping of the SFR Area (1)**

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 00D2 | | 8-bit A-D0 Data Register 1 (AD08DT1) |
| H'0080 00D4 | | 8-bit A-D0 Data Register 2 (AD08DT2) |
| H'0080 00D6 | | 8-bit A-D0 Data Register 3 (AD08DT3) |
| H'0080 00D8 | | 8-bit A-D0 Data Register 4 (AD08DT4) |
| H'0080 00DA | | 8-bit A-D0 Data Register 5 (AD08DT5) |
| H'0080 00DC | | 8-bit A-D0 Data Register 6 (AD08DT6) |
| H'0080 00DE | | 8-bit A-D0 Data Register 7 (AD08DT7) |
| H'0080 00E0 | | 8-bit A-D0 Data Register 8 (AD08DT8) |
| H'0080 00E2 | | 8-bit A-D0 Data Register 9 (AD08DT9) |
| H'0080 00E4 | | 8-bit A-D0 Data Register 10 (AD08DT10) |
| H'0080 00E6 | | 8-bit A-D0 Data Register 11 (AD08DT11) |
| H'0080 00E8 | | 8-bit A-D0 Data Register 12 (AD08DT12) |
| H'0080 00EA | | 8-bit A-D0 Data Register 13 (AD08DT13) |
| H'0080 00EC | | 8-bit A-D0 Data Register 14 (AD08DT14) |
| H'0080 00EE | | 8-bit A-D0 Data Register 15 (AD08DT15) |
| ≈ | | |
| H'0080 0100 | SIO23 Interrupt Status Register (SI23STAT) | SIO03 Interrupt Mask Register (SI03MASK) |
| H'0080 0102 | SIO03 Receive Interrupt Cause Select Register (SI03SEL) | |
| ≈ | | |
| H'0080 0110 | SIO0 Transmit Control Register (S0TCNT) | SIO0 Transmit/Receive Mode Register (S0MOD) |
| H'0080 0112 | SIO0 Transmit Buffer Register (S0TXB) | |
| H'0080 0114 | SIO0 Receive Buffer Register (S0RXB) | |
| H'0080 0116 | SIO0 Receive Control Register (S0RCNT) | SIO1 Baud Rate Register (S1BAUR) |
| ≈ | | |
| H'0080 0120 | SIO1 Transmit Control Register (S1TCNT) | SIO0 Transmit/Receive Mode Register (S1MOD) |
| H'0080 0122 | SIO1 Transmit Buffer Register (S1TXB) | |
| H'0080 0124 | SIO1 Receive Buffer Register (S1RXB) | |
| H'0080 0126 | SIO1 Receive Control Register (S1RCNT) | SIO1 Baud Rate Register (S1BAUR) |
| ≈ | | |
| H'0080 0130 | SIO2 Transmit Control Register (S2TCNT) | SIO2 Transmit/Receive Mode Register (S2MOD) |
| H'0080 0132 | SIO2 Transmit Buffer Register (S2TXB) | |
| H'0080 0134 | SIO2 Receive Buffer Register (S2RXB) | |
| H'0080 0136 | SIO2 Receive Control Register (S2RCNT) | SIO2 Baud Rate Register (S2BAUR) |
| ≈ | | |
| H'0080 0180 | Wait Cycles Control Register (WTCCR) | |
| ≈ | | |
| H'0080 0200 | | Clock Bus & Input Event Bus Control Register (CKIEBCR) |
| H'0080 0202 | Prescaler Register 0 (PRS0) | Prescaler Register 1 (PRS1) |
| H'0080 0204 | Prescaler Register 2 (PRS2) | Output Event Bus Control Register (OEBCR) |
| ≈ | | |
| H'0080 0210 | TCLK Input Processing Control Register (TCLKCR) | |
| H'0080 0212 | TIN Input Processing Control Register 0 (TINCR0) | |
| H'0080 0214 | | |

Blank addresses are reserved areas.

**Figure 3.4.4 Register Mapping of the SFR Area (2)**

<table>
<tr><th>Address</th><th>+0 Address</th><th>+1 Address</th></tr>
<tr><th></th><th>D0 D7</th><th>D8 D15</th></tr>
<tr><td>H'0080 0216</td><td></td><td></td></tr>
<tr><td>H'0080 0218</td><td colspan="2">TIN Input Processing Control Register 3 (TINCR3)</td></tr>
<tr><td>H'0080 021A</td><td colspan="2">TIN Input Processing Control Register 4 (TINCR4)</td></tr>
<tr><td>H'0080 021C</td><td></td><td></td></tr>
<tr><td>H'0080 021E</td><td></td><td></td></tr>
<tr><td>H'0080 0220</td><td colspan="2">F/F Source Select Register 0 (FFS0)</td></tr>
<tr><td>H'0080 0222</td><td></td><td>F/F Source Select Register 1 (FFS1)</td></tr>
<tr><td>H'0080 0224</td><td colspan="2">F/F Protect Register 0 (FFP0)</td></tr>
<tr><td>H'0080 0226</td><td colspan="2">F/F Data Register 0 (FFD0)</td></tr>
<tr><td>H'0080 0228</td><td></td><td>F/F Protect Register 1 (FFP1)</td></tr>
<tr><td>H'0080 022A</td><td></td><td>F/F Data Register 1 (FFD1)</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>H'0080 0230</td><td>TOP Interrupt Control Register 0 (TOPIR0)</td><td>TOP Interrupt Control Register 1 (TOPIR1)</td></tr>
<tr><td>H'0080 0232</td><td>TOP Interrupt Control Register 2 (TOPIR2)</td><td>TOP Interrupt Control Register 3 (TOPIR3)</td></tr>
<tr><td>H'0080 0234</td><td>TIO Interrupt Control Register 0 (TIOIR0)</td><td>TIO Interrupt Control Register 1 (TIOIR1)</td></tr>
<tr><td>H'0080 0236</td><td>TIO Interrupt Control Register 2 (TIOIR2)</td><td>TMS Interrupt Control Register (TMSIR)</td></tr>
<tr><td>H'0080 0238</td><td>TIN Interrupt Control Register 0 (TINIR0)</td><td>TIN Interrupt Control Register 1 (TINIR1)</td></tr>
<tr><td>H'0080 023A</td><td></td><td></td></tr>
<tr><td>H'0080 023C</td><td>TIN Interrupt Control Register 4 (TINIR4)</td><td>TIN Interrupt Control Register 5 (TINIR5)</td></tr>
<tr><td>H'0080 023E</td><td>TIN Interrupt Control Register 6 (TINIR6)</td><td></td></tr>
<tr><td>H'0080 0240</td><td colspan="2">TOP0 Counter (TOP0CT)</td></tr>
<tr><td>H'0080 0242</td><td colspan="2">TOP0 Reload Register (TOP0RL)</td></tr>
<tr><td>H'0080 0244</td><td></td><td></td></tr>
<tr><td>H'0080 0246</td><td colspan="2">TOP0 Correction Register (TOP0CC)</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>H'0080 0250</td><td colspan="2">TOP1 Counter (TOP1CT)</td></tr>
<tr><td>H'0080 0252</td><td colspan="2">TOP1 Reload Register (TOP1RL)</td></tr>
<tr><td>H'0080 0254</td><td></td><td></td></tr>
<tr><td>H'0080 0256</td><td colspan="2">TOP1 Correction Register (TOP1CC)</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>H'0080 0260</td><td colspan="2">TOP2 Counter (TOP2CT)</td></tr>
<tr><td>H'0080 0262</td><td colspan="2">TOP2 Reload Register (TOP2RL)</td></tr>
<tr><td>H'0080 0264</td><td></td><td></td></tr>
<tr><td>H'0080 0266</td><td colspan="2">TOP2 Correction Register (TOP2CC)</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>H'0080 0270</td><td colspan="2">TOP3 Counter (TOP3CT)</td></tr>
<tr><td>H'0080 0272</td><td colspan="2">TOP3 Reload Register (TOP3RL)</td></tr>
<tr><td>H'0080 0274</td><td></td><td></td></tr>
<tr><td>H'0080 0276</td><td colspan="2">TOP3 Correction Register (TOP3CC)</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>H'0080 0280</td><td colspan="2">TOP4 Counter (TOP4CT)</td></tr>
<tr><td>H'0080 0282</td><td colspan="2">TOP4 Reload Register (TOP4RL)</td></tr>
<tr><td>H'0080 0284</td><td></td><td></td></tr>
<tr><td>H'0080 0286</td><td colspan="2">TOP4 Correction Register (TOP4CC)</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>H'0080 0290</td><td colspan="2">TOP5 Counter (TOP5CT)</td></tr>
<tr><td>H'0080 0292</td><td colspan="2">TOP5 Reload Register (TOP5RL)</td></tr>
<tr><td>H'0080 0294</td><td></td><td></td></tr>
</table>

Blank addresses are reserved areas.

**Figure 3.4.5 Register Mapping of the SFR Area (3)**

<table>
<tr><th>Address</th><th>D0 +0 Address D7</th><th>D8 +1 Addres D15</th></tr>
<tr><td>H'0080 0296</td><td colspan="2">TOP5 Correction Register (TOP5CC)</td></tr>
<tr><td>H'0080 0298</td><td></td><td></td></tr>
<tr><td>H'0080 029A</td><td colspan="2">TOP0-5 Control Register 0 (TOP05CR0)</td></tr>
<tr><td>H'0080 029C</td><td></td><td>TOP0-5 Control Register 1 (TOP05CR1)</td></tr>
<tr><td>H'0080 029E</td><td></td><td></td></tr>
<tr><td>H'0080 02A0</td><td colspan="2">TOP6 Counter (TOP6CT)</td></tr>
<tr><td>H'0080 02A2</td><td colspan="2">TOP6 Reload Register (TOP6RL)</td></tr>
<tr><td>H'0080 02A4</td><td colspan="2"></td></tr>
<tr><td>H'0080 02A6</td><td colspan="2">TOP6 Correction Register (TOP6CC)</td></tr>
<tr><td>H'0080 02A8</td><td></td><td></td></tr>
<tr><td>H'0080 02AA</td><td colspan="2">TOP6, 7 Control Register (TOP67CR)</td></tr>
<tr><td>≈</td><td></td><td></td></tr>
<tr><td>H'0080 02B0</td><td colspan="2">TOP7 Counter (TOP7CT)</td></tr>
<tr><td>H'0080 02B2</td><td colspan="2">TOP7 Reload Register (TOP7RL)</td></tr>
<tr><td>H'0080 02B4</td><td></td><td></td></tr>
<tr><td>H'0080 02B6</td><td colspan="2">TOP7 Correction Register (TOP7CC)</td></tr>
<tr><td>≈</td><td></td><td></td></tr>
<tr><td>H'0080 02C0</td><td colspan="2">TOP8Counter (TOP8CT)</td></tr>
<tr><td>H'0080 02C2</td><td colspan="2">TOP8 Reload Register (TOP8RL)</td></tr>
<tr><td>H'0080 02C4</td><td></td><td></td></tr>
<tr><td>H'0080 02C6</td><td colspan="2">TOP8 Correction Register (TOP8CC)</td></tr>
<tr><td>≈</td><td></td><td></td></tr>
<tr><td>H'0080 02D0</td><td colspan="2">TOP9 Counter (TOP9CT)</td></tr>
<tr><td>H'0080 02D2</td><td colspan="2">TOP9 Reload Register (TOP9RL)</td></tr>
<tr><td>H'0080 02D4</td><td></td><td></td></tr>
<tr><td>H'0080 02D6</td><td colspan="2">TOP9 Correction Register (TOP9CC)</td></tr>
<tr><td>≈</td><td></td><td></td></tr>
<tr><td>H'0080 02E0</td><td colspan="2">TOP10 Counter (TOP10CT)</td></tr>
<tr><td>H'0080 02E2</td><td colspan="2">TOP10 Reload Register (TOP10RL)</td></tr>
<tr><td>H'0080 02E4</td><td colspan="2"></td></tr>
<tr><td>H'0080 02E6</td><td colspan="2">TOP10 Correction Register (TOP10CC)</td></tr>
<tr><td>H'0080 02E8</td><td></td><td></td></tr>
<tr><td>H'0080 02EA</td><td colspan="2">TOP8-10 Control Register (TOP810CR)</td></tr>
<tr><td>≈</td><td></td><td></td></tr>
<tr><td>H'0080 02FA</td><td colspan="2">TOP0-10 External Enable Register (TOPEEN)</td></tr>
<tr><td>H'0080 02FC</td><td colspan="2">TOP0-10 Enable Protect Register (TOPPRO)</td></tr>
<tr><td>H'0080 02FE</td><td colspan="2">TOP0-10 Count Enable Register (TOPCEN)</td></tr>
<tr><td>H'0080 0300</td><td colspan="2">TIO0 Counter (TIO0CT)</td></tr>
<tr><td>H'0080 0302</td><td></td><td></td></tr>
<tr><td>H'0080 0304</td><td colspan="2">TIO0 Reload Register (TIO0RL1)</td></tr>
<tr><td>H'0080 0306</td><td colspan="2">TIO0 Reload 0/Measure Register (TIO0RL0)</td></tr>
<tr><td>≈</td><td></td><td></td></tr>
<tr><td>H'0080 0310</td><td colspan="2">TIO1 Counter (TIO1CT)</td></tr>
<tr><td>H'0080 0312</td><td colspan="2"></td></tr>
<tr><td>H'0080 0314</td><td colspan="2">TIO1 Reload Register (TIO1RL1)</td></tr>
<tr><td>H'0080 0316</td><td colspan="2">TIO1 Reload 0/Measure Register (TIO1RL0)</td></tr>
<tr><td>H'0080 0318</td><td></td><td></td></tr>
<tr><td>H'0080 031A</td><td colspan="2">TIO0-3 Control Register 0 (TIO03CR0)</td></tr>
</table>

Blank addresses are reserved areas.

**Figure 3.4.6 Register Mapping of the SFR Area (4)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 031C | | TIO0-3 Control Register 1 (TIO03CR1) |
| | | |
| H'0080 0320 | TIO2 Counter (TIO2CT) | |
| H'0080 0322 | | |
| H'0080 0324 | TIO2 Reload 1 Register (TIO2RL1) | |
| H'0080 0326 | TIO2 Reload 0/Measure Register (TIO2RL0) | |
| | | |
| H'0080 0330 | TIO3 Counter (TIO3CT) | |
| H'0080 0332 | | |
| H'0080 0334 | TIO3 Reload 1 Register (TIO3RL1) | |
| H'0080 0336 | TIO3 Reload 0/Measure Register (TIO3RL0) | |
| | | |
| H'0080 0340 | TIO4 Counter (TIO4CT) | |
| H'0080 0342 | | |
| H'0080 0344 | TIO4 Reload 1 Register (TIO4RL1) | |
| H'0080 0346 | TIO4 Reload 0/Measure Register (TIO4RL0) | |
| H'0080 0348 | | |
| H'0080 034A | TIO4 Control Register (TIO4CR) | TIO5 Control Register (TIO5CR) |
| | | |
| H'0080 0350 | TIO5 Counter (TIO5CT) | |
| H'0080 0352 | | |
| H'0080 0354 | TIO5 Reload 1 Register (TIO5RL1) | |
| H'0080 0356 | TIO5 Reload 0/Measure Register (TIO5RL0) | |
| | | |
| H'0080 0360 | TIO6 Counter (TIO6CT) | |
| H'0080 0362 | | |
| H'0080 0364 | TIO6 Reload 1 Register (TIO6RL1) | |
| H'0080 0366 | TIO6 Reload 0/Measure Register (TIO6RL0) | |
| H'0080 0368 | | |
| H'0080 036A | TIO6 Control Register (TIO6CR) | TIO7 Control Register (TIO7CR) |
| | | |
| H'0080 0370 | TIO7 Counter (TIO7CT) | |
| H'0080 0372 | | |
| H'0080 0374 | TIO7 Reload 1 Register (TIO7RL1) | |
| H'0080 0376 | TIO7 Reload 0/Measure Register (TIO7RL0) | |
| | | |
| H'0080 0380 | TIO8 Counter (TIO8CT) | |
| H'0080 0382 | | |
| H'0080 0384 | TIO8 Reload 1 Register (TIO8RL1) | |
| H'0080 0386 | TIO8 Reload 0/Measure Register (TIO8RL0) | |
| H'0080 0388 | | |
| H'0080 038A | TIO8 Control Register (TIO8CR) | TIO9 Control Register (TIO9CR) |
| | | |
| H'0080 0390 | TIO9 Counter (TIO9CT) | |
| H'0080 0392 | | |
| H'0080 0394 | TIO9 Reload 1 Register (TIO9RL1) | |
| H'0080 0396 | TIO9 Reload 0/Measure Register (TIO9RL0) | |
| | | |

Blank addresses are reserved areas.

**Figure 3.4.7 Register Mapping of the SFR Area (5)**

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 03BC | TIO0-9 Enable Protect Register (TIOPRO) | |
| H'0080 03BE | TIO0-9 Count Enable Register (TIOCEN) | |
| H'0080 03C0 | TMS0 Counter (TMS0CT) | |
| H'0080 03C2 | TMS0 Measure 3 Register (TMS0MR3) | |
| H'0080 03C4 | TMS0 Measure 2 Register (TMS0MR2) | |
| H'0080 03C6 | TMS0 Measure 1 Register (TMS0MR1) | |
| H'0080 03C8 | TMS0 Measure 0 Register (TMS0MR0) | |
| H'0080 03CA | TMS0 Control Register (TMS0CR) | TMS1 Control Register (TMS1CR) |
| ≈ | | |
| H'0080 03D0 | TMS1 Counter (TMS1CT) | |
| H'0080 03D2 | TMS1 Measure 3 Register (TMS1MR3) | |
| H'0080 03D4 | TMS1 Measure 2 Register (TMS1MR2) | |
| H'0080 03D6 | TMS1 Measure 1 Register (TMS1MR1) | |
| H'0080 03D8 | TMS1 Measure 0 Register (TMS1MR0) | |
| ≈ | | |
| H'0080 03E0 | TML0 Counter, High (TML0CTH) | |
| H'0080 03E2 | TML0 Counter, Low (TML0CTL) | |
| ≈ | | |
| H'0080 03EA | | TML0 Control Register (TML0CR) |
| ≈ | | |
| H'0080 03F0 | TML0 Measure 3 Register, High (TML0MR3H) | |
| H'0080 03F2 | TML0 Measure 3 Register, Low (TML0MR3L) | |
| H'0080 03F4 | TML0 Measure 2 Register, High (TML0MR2H) | |
| H'0080 03F6 | TML0 Measure 2 Register, Low (TML0MR2L) | |
| H'0080 03F8 | TML0 Measure 1 Register, High (TML0MR1H) | |
| H'0080 03FA | TML0 Measure 1 Register, Low (TML0MR1L) | |
| H'0080 03FC | TML0 Measure 0 Register, High (TML0MR0H) | |
| H'0080 03FE | TML0 Measure 0 Register, Low (TML0MR0L) | |
| H'0080 0400 | DMA0-4 Interrupt Request Status Register (DM04ITST) | DMA0-4 Interrupt Mask Register (DM04ITMK) |
| ≈ | | |
| H'0080 0408 | DMA5-9 Interrupt Request Status Register (DM59ITST) | DMA5-9 Interrupt Mask Register (DM59ITMK) |
| ≈ | | |
| H'0080 0410 | DMA0 Channel Control Register (DM0CNT) | DMA0 Transfer Count Register (DM0TCT) |
| H'0080 0412 | DMA0 Source Address Register (DM0SA) | |
| H'0080 0414 | DMA0 Destination Address Register (DM0DA) | |
| H'0080 0416 | | |
| H'0080 0418 | DMA5 Channel Control Register (DM5CNT) | DMA5 Transfer Count Register (DM5TCT) |
| H'0080 041A | DMA5 Source Address Register (DM5SA) | |
| H'0080 041C | DMA5 Destination Address Register (DM5DA) | |
| H'0080 041E | | |
| H'0080 0420 | DMA1 Channel Control Register (DM1CNT) | DMA1 Transfer Count Register (DM1TCT) |
| H'0080 0422 | DMA1 Source Address Register (DM1SA) | |
| H'0080 0424 | DMA1 Destination Address Register (DM1DA | |
| H'0080 0426 | | |
| H'0080 0428 | DMA6 Channel Control Register (DM6CNT) | DMA6 Transfer Count Register (DM6TCT) |
| H'0080 042A | DMA6 Source Address Register (DM6SA) | |
| H'0080 042C | DMA6 Destination Address Register (DM6DA) | |
| H'0080 042E | | |

Blank addresses are reserved areas.

**Figure 3.4.8 Register Mapping of the SFR Area (6)**

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 0430 | DMA2 Channel Control Register (DM2CNT) | DMA2 Transfer Count Register (DM2TCT) |
| H'0080 0432 | DMA2 Source Address Register (DM2SA) | |
| H'0080 0434 | DMA2 Destination Address Register (DM2DA) | |
| H'0080 0436 | | |
| H'0080 0438 | DMA7 Channel Control Register (DM7CNT) | DMA7 Transfer Count Register (DM7TCT) |
| H'0080 043A | DMA7 Source Address Register (DM7SA) | |
| H'0080 043C | DMA7 Destination Address Register (DM7DA) | |
| H'0080 043E | | |
| H'0080 0440 | DMA3 Channel Control Register (DM3CNT) | DMA3 Transfer Count Register (DM3TCT) |
| H'0080 0442 | DMA3 Source Address Register (DM3SA) | |
| H'0080 0444 | DMA3 Destination Address Register (DM3DA) | |
| H'0080 0446 | | |
| H'0080 0448 | DMA8 Channel Control Register (DM8CNT) | DMA8 Transfer Count Register (DM8TCT) |
| H'0080 044A | DMA8 Source Address Register (DM8SA) | |
| H'0080 044C | DMA8 Destination Address Register (DM8DA) | |
| H'0080 044E | | |
| H'0080 0450 | DMA4 Channel Control Register (DM4CNT) | DMA4 Transfer Count Register (DM4TCT) |
| H'0080 0452 | DMA4 Source Address Register (DM4SA) | |
| H'0080 0454 | DMA4 Destination Address Register (DM4DA) | |
| H'0080 0456 | | |
| H'0080 0458 | DMA9 Channel Control Register (DM9CNT) | DMA9 Transfer Count Register (DM9TCT) |
| H'0080 045A | DMA9 Source Address Register (DM9SA) | |
| H'0080 045C | DMA9 Destination Address Register (DM9DA) | |
| H'0080 045E | | |
| H'0080 0460 | DMA0 Software Request Generation Register (DM0SRI) | |
| H'0080 0462 | DMA1 Software Request Generation Register (DM1SRI) | |
| H'0080 0464 | DMA2 Software Request Generation Register (DM2SRI) | |
| H'0080 0466 | DMA3 Software Request Generation Register (DM3SRI) | |
| H'0080 0468 | DMA4 Software Request Generation Register (DM4SRI) | |
| | | |
| H'0080 0470 | DMA5 Software Request Generation Register (DM5SRI) | |
| H'0080 0472 | DMA6 Software Request Generation Register (DM6SRI) | |
| H'0080 0474 | DMA7 Software Request Generation Register (DM7SRI) | |
| H'0080 0476 | DMA8 Software Request Generation Register (DM8SRI) | |
| H'0080 0478 | DMA9 Software Request Generation Register (DM9SRI) | |
| | | |
| H'0080 0700 | P0 Data Register (P0DATA) | P1 Data Register (P1DATA) |
| H'0080 0702 | P2 Data Register (P2DATA) | P3 Data Register (P3DATA) |
| H'0080 0704 | P4 Data Register (P4DATA) | |
| H'0080 0706 | P6 Data Register (P6DATA) | P7 Data Register (P7DATA) |
| H'0080 0708 | P8 Data Register (P8DATA) | P9 Data Register (P9DATA) |
| H'0080 070A | P10 Data Register (P10DATA) | P11Data Register (P11DATA) |
| H'0080 070C | P12 Data Register (P12DATA) | P13 Data Register (P13DATA) |
| H'0080 070E | | P15 Data Register (P15DATA) |
| H'0080 0710 | | P17 Data Register (P17DATA) |
| H'0080 0712 | | |
| H'0080 0714 | | |

Blank addresses are reserved areas.

**Figure 3.4.9 Register Mapping of the SFR Area (7)**

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 0716 | P22 Data Register (P22DATA) | |
| ≈ | | |
| H'0080 0720 | P0 Direction Register (P0DIR) | P1 Direction Register (P1DIR) |
| H'0080 0722 | P2 Direction Register (P2DIR) | P3 Direction Register (P3DIR) |
| H'0080 0724 | P4 Direction Register (P4DIR) | |
| H'0080 0726 | P6 Direction Register (P6DIR) | P7 Direction Register (P7DIR) |
| H'0080 0728 | P8 Direction Register (P8DIR) | P9 Direction Register (P9DIR) |
| H'0080 072A | P10 Direction Register (P10DIR) | P11 Direction Register (P11DIR) |
| H'0080 072C | P12 Direction Register (P12DIR) | P13 Direction Register (P13DIR) |
| H'0080 072E | | P15 Direction Register (P15DIR) |
| H'0080 0730 | | P17 Direction Register (P17DIR) |
| H'0080 0732 | | |
| H'0080 0734 | | |
| H'0080 0736 | P22 Direction Register (P22DIR) | |
| ≈ | | |
| H'0080 0744 | | Port Input Function Enable Register (PIEN) |
| H'0080 0746 | | P7 Operation Mode Register (P7MOD) |
| H'0080 0748 | P8 Operation Mode Register (P8MOD) | P9 Operation Mode Register (P9MOD) |
| H'0080 074A | P10 Operation Mode Register (P10MOD) | P11 Operation Mode Register (P11MOD) |
| H'0080 074C | P12 Operation Mode Register (P12MOD) | P13 Operation Mode Register (P13MOD) |
| H'0080 074E | | P15 Operation Mode Register (P15MOD) |
| H'0080 0750 | | P17 Operation Mode Register (P17MOD) |
| H'0080 0752 | | |
| H'0080 0754 | | |
| H'0080 0756 | P22 Operation Mode Register (P22MOD) | |
| ≈ | | |
| H'0080 077E | | Bus Mode Control Register (BUSMODC) |
| ≈ | | |
| H'0080 07E0 | Flash Mode Register (FMOD) | Flash Status Register 1 (FSTAT1) |
| H'0080 07E2 | Flash Control Register 1 (FCNT1) | Flash Control Register 2 (FCNT2) |
| H'0080 07E4 | Flash Control Register 3 (FCNT3) | Flash Control Register 4 (FCNT4) |
| H'0080 07E6 | | |
| H'0080 07E8 | Virtual-flash L Bank Register 0 (FELBANK0) | |
| ≈ | | |
| H'0080 07F0 | Virtual-flash S Bank Register 0 (FESBANK0) | |
| H'0080 07F2 | Virtual-flash S Bank Register 1 (FESBANK1) | |
| ≈ | | |
| H'0080 0FE0 | TML1 Counter, High (TML1CTH) | |
| H'0080 0FE2 | TML1 Counter, Low (TML1CTL) | |
| ≈ | | |
| H'0080 0FEA | | TML1 Control Register (TML1CR) |
| ≈ | | |
| H'0080 0FF0 | TML1 Measure 3 Register, High (TML1MR3H) | |
| H'0080 0FF2 | TML1 Measure 3 Register, Low (TML1MR3L) | |

Blank addresses are reserved areas.

**Figure 3.4.10 Register Mapping of the SFR Area (8)**

| Address | +0 Address<br>D0 … D7 | +1 Address<br>D8 … D15 |
|---|---|---|
| H'0080 0FF4 | TML1 Measure 2 Register, High(TML1MR2H) | |
| H'0080 0FF6 | TML1 Measure 2 Register, Low(TML1MR2L) | |
| H'0080 0FF8 | TML1 Measure 1 Register, High(TML1MR1H) | |
| H'0080 0FFA | TML1 Measure 1 Register, Low(TML1MR1L) | |
| H'0080 0FFC | TML1 Measure 0 Register, High(TML1MR0H) | |
| H'0080 0FFE | TML1 Measure 0 Register, Low(TML1MR0L) | |
| H'0080 1000 | CAN0 Control Register (CAN0CNT) | |
| H'0080 1002 | CAN0 Status Register (CAN0STAT) | |
| H'0080 1004 | CAN0 Extension ID Register (CAN0EXTID) | |
| H'0080 1006 | CAN0 Configuration Register (CAN0CONF) | |
| H'0080 1008 | CAN0 Time Stamp Count Register (CAN0TSTMP) | |
| H'0080 100A | CAN0 Receive Error Count Register (CAN0REC) | CAN0 Transmit Error Count Register (CAN0TEC) |
| H'0080 100C | CAN0 Slot Interrupt Status Register (CAN0SLIST) | |
| H'0080 100E | | |
| H'0080 1010 | CAN0 Slot Interrupt Mask Register (CAN0SLIMK) | |
| H'0080 1012 | | |
| H'0080 1014 | CAN0 Error Interrupt Status Register (CAN0ERIST) | CAN0 Error Interrupt Mask Register (CAN0ERIMK) |
| H'0080 1016 | CAN0 Baut Rate Prescaler (CAN0BRP) | |
| H'0080 1028 | CAN0 Global Mask Register Standard ID0(C0GMSKS0) | CAN0 Global Mask Register Standard ID1(C0GMSKS1) |
| H'0080 102A | CAN0 Global Mask Register Extended ID0(C0GMSKE0) | CAN0 Global Mask Register Extended ID1(C0GMSKE1) |
| H'0080 102C | CAN0 Global Mask Register Extended ID2(C0GMSKE2) | |
| H'0080 102E | | |
| H'0080 1030 | CAN0 Local Mask Register A Standard ID0(C0LMSKAS0) | CAN0 Local Mask Register A Standard ID1(C0LMSKAS1) |
| H'0080 1032 | CAN0 Local Mask Register A Extended ID0(C0LMSKAE0) | CAN0 Local Mask Register A Extended ID1(C0LMSKAE1) |
| H'0080 1034 | CAN0 Local Mask Register A Extended ID2(C0LMSKAE2) | |
| H'0080 1036 | | |
| H'0080 1038 | CAN0 Local Mask Register B Standard ID0(C0LMSKBS0) | CAN0 Local Mask Register B Standard ID1(C0LMSKBS1) |
| H'0080 103A | CAN0 Local Mask Register B Extended ID0(C0LMSKBE0) | CAN0 Local Mask Register B Extended ID1(C0LMSKBE1) |
| H'0080 103C | CAN0 Local Mask Register B Extended ID2(C0LMSKBE2) | |
| H'0080 1050 | CAN0 Message Slot 0 Control Register (C0MSL0CNT) | CAN0 Message Slot 1 Control Register (C0MSL1CNT) |
| H'0080 1052 | CAN0 Message Slot 2 Control Register (C0MSL2CNT) | CAN0 Message Slot 3 Control Register (C0MSL3CNT) |
| H'0080 1054 | CAN0 Message Slot 4 Control Register (C0MSL4CNT) | CAN0 Message Slot 5 Control Register (C0MSL5CNT) |
| H'0080 1056 | CAN0 Message Slot 6 Control Register (C0MSL6CNT) | CAN0 Message Slot 7 Control Register (C0MSL7CNT) |
| H'0080 1058 | CAN0 Message Slot 8 Control Register (C0MSL8CNT) | CAN0 Message Slot 9 Control Register (C0MSL9CNT) |
| H'0080 105A | CAN0 Message Slot 10 Control Register (C0MSL10CNT) | CAN0 Message Slot 11 Control Register (C0MSL11CNT) |
| H'0080 105C | CAN0 Message Slot 12 Control Register (C0MSL12CNT) | CAN0 Message Slot 13 Control Register (C0MSL13CNT) |
| H'0080 105E | CAN0 Message Slot 14 Control Register (C0MSL14CNT) | CAN0 Message Slot 15 Control Register (C0MSL15CNT) |

Blank addresses are reserved areas.

**Figure 3.4.11 Register Mapping of the SFR Area (9)**

| Address | +0 Addres | +1 Addres |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 1100 | CAN0 Message Slot 0 Standard ID0 (C0MSL0SID0) | CAN0 Message Slot 0 Standard ID1 (C0MSL0SID1) |
| H'0080 1102 | CAN0 Message Slot 0 Extended ID0 (C0MSL0EID0) | CAN0 Message Slot 0 Extended ID1 (C0MSL0EID1) |
| H'0080 1104 | CAN0 Message Slot 0 Extended ID2 (C0MSL0EID2) | CAN0 Message Slot 0 Data Length Register (C0MSL0DLC) |
| H'0080 1106 | CAN0 Message Slot 0 Data 0 (C0MSL0DT0) | CAN0 Message Slot 0 Data 1 (C0MSL0DT1) |
| H'0080 1108 | CAN0 Message Slot 0 Data 2 (C0MSL0DT2) | CAN0 Message Slot 0 Data 3 (C0MSL0DT3) |
| H'0080 110A | CAN0 Message Slot 0 Data 4 (C0MSL0DT4) | CAN0 Message Slot 0 Data 5 (C0MSL0DT5) |
| H'0080 110C | CAN0 Message Slot 0 Data 6 (C0MSL0DT6) | CAN0 Message Slot 0 Data 7 (C0MSL0DT7) |
| H'0080 110E | CAN0 Message Slot 0 Time Stamp (C0MSL0TSP) | |
| H'0080 1110 | CAN0 Message Slot 1 Standard ID0 (C0MSL1SID0) | CAN0 Message Slot 1 Standard ID1 (C0MSL1SID1) |
| H'0080 1112 | CAN0 Message Slot 1 Extended ID0 (C0MSL1EID0) | CAN0 Message Slot 1 Extended ID1 (C0MSL1EID1) |
| H'0080 1114 | CAN0 Message Slot 1 Extended ID2 (C0MSL1EID2) | CAN0 Message Slot 1 Data Length Register (C0MSL1DLC) |
| H'0080 1116 | CAN0 Message Slot 1 Data 0 (C0MSL1DT0) | CAN0 Message Slot 1 Data 1 (C0MSL1DT1) |
| H'0080 1118 | CAN0 Message Slot 1 Data 2 (C0MSL1DT2) | CAN0 Message Slot 1 Data 3 (C0MSL1DT3) |
| H'0080 111A | CAN0 Message Slot 1 Data 4 (C0MSL1DT4) | CAN0 Message Slot 1 Data 5 (C0MSL1DT5) |
| H'0080 111C | CAN0 Message Slot 1 Data 6 (C0MSL1DT6) | CAN0 Message Slot 1 Data 7 (C0MSL1DT7) |
| H'0080 111E | CAN0 Message Slot 1 Time Stamp (C0MSL1TSP) | |
| H'0080 1120 | CAN0 Message Slot 2 Standard ID0 (C0MSL2SID0) | CAN0 Message Slot 2 Standard ID1 (C0MSL2SID1) |
| H'0080 1122 | CAN0 Message Slot 2 Extended ID0 (C0MSL2EID0) | CAN0 Message Slot 2 Extended ID1 (C0MSL2EID1) |
| H'0080 1124 | CAN0 Message Slot 2 Extended ID2 (C0MSL2EID2) | CAN0 Message Slot 2 Data Length Register (C0MSL2DLC) |
| H'0080 1126 | CAN0 Message Slot 2 Data 0 (C0MSL2DT0) | CAN0 Message Slot 2 Data 1 (C0MSL2DT1) |
| H'0080 1128 | CAN0 Message Slot 2 Data 2 (C0MSL2DT2) | CAN0 Message Slot 2 Data 3 (C0MSL2DT3) |
| H'0080 112A | CAN0 Message Slot 2 Data 4 (C0MSL2DT4) | CAN0 Message Slot 2 Data 5 (C0MSL2DT5) |
| H'0080 112C | CAN0 Message Slot 2 Data 6 (C0MSL2DT6) | CAN0 Message Slot 2 Data 7 (C0MSL2DT7) |
| H'0080 112E | CAN0 Message Slot 2 Time Stamp (C0MSL2TSP) | |
| H'0080 1130 | CAN0 Message Slot 3 Standard ID0 (C0MSL3SID0) | CAN0 Message Slot 3 Standard ID1 (C0MSL3SID1) |
| H'0080 1132 | CAN0 Message Slot 3 Extended ID0 (C0MSL3EID0) | CAN0 Message Slot 3 Extended ID1 (C0MSL3EID1) |
| H'0080 1134 | CAN0 Message Slot 3 Extended ID2 (C0MSL3EID2) | CAN0 Message Slot 3 Data Length Register (C0MSL3DLC) |
| H'0080 1136 | CAN0 Message Slot 3 Data 0 (C0MSL3DT0) | CAN0 Message Slot 3 Data 1 (C0MSL3DT1) |
| H'0080 1138 | CAN0 Message Slot 3 Data 2 (C0MSL3DT2) | CAN0 Message Slot 3 Data 3 (C0MSL3DT3) |
| H'0080 113A | CAN0 Message Slot 3 Data 4 (C0MSL3DT4) | CAN0 Message Slot 3 Data 5 (C0MSL3DT5) |
| H'0080 113C | CAN0 Message Slot 3 Data 6 (C0MSL3DT6) | CAN0 Message Slot 3 Data 7 (C0MSL3DT7) |
| H'0080 113E | CAN0 Message Slot 3 Time Stamp (C0MSL3TSP) | |
| H'0080 1140 | CAN0 Message Slot 4 Standard ID0 (C0MSL4SID0) | CAN0 Message Slot 4 Standard ID1 (C0MSL4SID1) |
| H'0080 1142 | CAN0 Message Slot 4 Extended ID0 (C0MSL4EID0) | CAN0 Message Slot 4 Extended ID1 (C0MSL4EID1) |
| H'0080 1144 | CAN0 Message Slot 4 Extended ID2 (C0MSL4EID2) | CAN0 Message Slot 4 Data Length Register (C0MSL4DLC) |
| H'0080 1146 | CAN0 Message Slot 4 Data 0 (C0MSL4DT0) | CAN0 Message Slot 4 Data 1 (C0MSL4DT1) |
| H'0080 1148 | CAN0 Message Slot 4 Data 2 (C0MSL4DT2) | CAN0 Message Slot 4 Data 3 (C0MSL4DT3) |
| H'0080 114A | CAN0 Message Slot 4 Data 4 (C0MSL4DT4) | CAN0 Message Slot 4 Data 5 (C0MSL4DT5) |
| H'0080 114C | CAN0 Message Slot 4 Data 6 (C0MSL4DT6) | CAN0 Message Slot 4 Data 7 (C0MSL4DT7) |
| H'0080 114E | CAN0 Message Slot 4 Time Stamp (C0MSL4TSP) | |
| H'0080 1150 | CAN0 Message Slot 5 Standard ID0 (C0MSL5SID0) | CAN0 Message Slot 5 Standard ID1 (C0MSL5SID1) |
| H'0080 1152 | CAN0 Message Slot 5 Extended ID0 (C0MSL5EID0) | CAN0 Message Slot 5 Extended ID1 (C0MSL5EID1) |

Blank addresses are reserved areas.

**Figure 3.4.12 Register Mapping of the SFR Area (10)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 1154 | CAN0 Message Slot 5 Extended ID2 (C0MSL5EID2) | CAN0 Message Slot 5 Data Length Register (C0MSL5DLC) |
| H'0080 1156 | CAN0 Message Slot 5 Data 0 (C0MSL5DT0) | CAN0 Message Slot 5 Data 1 (C0MSL5DT1) |
| H'0080 1158 | CAN0 Message Slot 5 Data 2 (C0MSL5DT2) | CAN0 Message Slot 5 Data 3 (C0MSL5DT3) |
| H'0080 115A | CAN0 Message Slot 5 Data 4 (C0MSL5DT4) | CAN0 Message Slot 5 Data 5 (C0MSL5DT5) |
| H'0080 115C | CAN0 Message Slot 5 Data 6 (C0MSL5DT6) | CAN0 Message Slot 5 Data 7 (C0MSL5DT7) |
| H'0080 115E | CAN0 Message Slot 5 Time Stamp (C0MSL5TSP) | |
| H'0080 1160 | CAN0 Message Slot 6 Standard ID0 (C0MSL6SID0) | CAN0 Message Slot 6 Standard ID1 (C0MSL6SID1) |
| H'0080 1162 | CAN0 Message Slot 6 Extended ID0 (C0MSL6EID0) | CAN0 Message Slot 6 Extended ID1 (C0MSL6EID1) |
| H'0080 1164 | CAN0 Message Slot 6 Extended ID2 (C0MSL6EID2) | CAN0 Message Slot 6 Data Length Register (C0MSL6DLC) |
| H'0080 1166 | CAN0 Message Slot 6 Data 0 (C0MSL6DT0) | CAN0 Message Slot 6 Data 1 (C0MSL6DT1) |
| H'0080 1168 | CAN0 Message Slot 6 Data 2 (C0MSL6DT2) | CAN0 Message Slot 6 Data 3 (C0MSL6DT3) |
| H'0080 116A | CAN0 Message Slot 6 Data 4 (C0MSL6DT4) | CAN0 Message Slot 6 Data 5 (C0MSL6DT5) |
| H'0080 116C | CAN0 Message Slot 6 Data 6 (C0MSL6DT6) | CAN0 Message Slot 6 Data 7 (C0MSL6DT7) |
| H'0080 116E | CAN0 Message Slot 6 Time Stamp (C0MSL6TSP) | |
| H'0080 1170 | CAN0 Message Slot 7 Standard ID0 (C0MSL7SID0) | CAN0 Message Slot 7 Standard ID1 (C0MSL7SID1) |
| H'0080 1172 | CAN0 Message Slot 7 Extended ID0 (C0MSL7EID0) | CAN0 Message Slot 7 Extended ID1 (C0MSL7EID1) |
| H'0080 1174 | CAN0 Message Slot 7 Extended ID2 (C0MSL7EID2) | CAN0 Message Slot 7 Data Length Register (C0MSL7DLC) |
| H'0080 1176 | CAN0 Message Slot 7 Data 0 (C0MSL7DT0) | CAN0 Message Slot 7 Data 1 (C0MSL7DT1) |
| H'0080 1178 | CAN0 Message Slot 7 Data 2 (C0MSL7DT2) | CAN0 Message Slot 7 Data 3 (C0MSL7DT3) |
| H'0080 117A | CAN0 Message Slot 7 Data 4 (C0MSL7DT4) | CAN0 Message Slot 7 Data 5 (C0MSL7DT5) |
| H'0080 117C | CAN0 Message Slot 7 Data 6 (C0MSL7DT6) | CAN0 Message Slot 7 Data 7 (C0MSL7DT7) |
| H'0080 117E | CAN0 Message Slot 7 Time Stamp (C0MSL7TSP) | |
| H'0080 1180 | CAN0 Message Slot 8 Standard ID0 (C0MSL8SID0) | CAN0 Message Slot 8 Standard ID1 (C0MSL8SID1) |
| H'0080 1182 | CAN0 Message Slot 8 Extended ID0 (C0MSL8EID0) | CAN0 Message Slot 8 Extended ID1 (C0MSL8EID1) |
| H'0080 1184 | CAN0 Message Slot 8 Extended ID2 (C0MSL8EID2) | CAN0 Message Slot 8 Data Length Register (C0MSL8DLC) |
| H'0080 1186 | CAN0 Message Slot 8 Data 0 (C0MSL8DT0) | CAN0 Message Slot 8 Data 1 (C0MSL8DT1) |
| H'0080 1188 | CAN0 Message Slot 8 Data 2 (C0MSL8DT2) | CAN0 Message Slot 8 Data 3 (C0MSL8DT3) |
| H'0080 118A | CAN0 Message Slot 8 Data 4 (C0MSL8DT4) | CAN0 Message Slot 8 Data 5 (C0MSL8DT5) |
| H'0080 118C | CAN0 Message Slot 8 Data 6 (C0MSL8DT6) | CAN0 Message Slot 8 Data 7 (C0MSL8DT7) |
| H'0080 118E | CAN0 Message Slot 8 Time Stamp (C0MSL8TSP) | |
| H'0080 1190 | CAN0 Message Slot 9 Standard ID0 (C0MSL9SID0) | CAN0 Message Slot 9 Standard ID1 (C0MSL9SID1) |
| H'0080 1192 | CAN0 Message Slot 9 Extended ID0 (C0MSL9EID0) | CAN0 Message Slot 9 Extended ID1 (C0MSL9EID1) |
| H'0080 1194 | CAN0 Message Slot 9 Extended ID2 (C0MSL9EID2) | CAN0 Message Slot 9 Data Length Register (C0MSL9DLC) |
| H'0080 1196 | CAN0 Message Slot 9 Data 0 (C0MSL9DT0) | CAN0 Message Slot 9 Data 1 (C0MSL9DT1) |
| H'0080 1198 | CAN0 Message Slot 9 Data 2 (C0MSL9DT2) | CAN0 Message Slot 9 Data 3 (C0MSL9DT3) |
| H'0080 119A | CAN0 Message Slot 9 Data 4 (C0MSL9DT4) | CAN0 Message Slot 9 Data 5 (C0MSL9DT5) |
| H'0080 119C | CAN0 Message Slot 9 Data 6 (C0MSL9DT6) | CAN0 Message Slot 9 Data 7 (C0MSL9DT7) |
| H'0080 119E | CAN0 Message Slot 9 Time Stamp (C0MSL9TSP) | |
| H'0080 11A0 | CAN0 Message Slot 10 Standard ID0 (C0MSL10SID0) | CAN0 Message Slot 10 Standard ID1 (C0MSL10SID1) |
| H'0080 11A2 | CAN0 Message Slot 10 Extended ID0 (C0MSL10EID0) | CAN0 Message Slot 10 Extended ID1 (C0MSL10EID1) |
| H'0080 11A4 | CAN0 Message Slot 10 Extended ID2 (C0MSL10EID2) | CAN0 Message Slot 10 Data Length Register (C0MSL10DLC) |
| H'0080 11A6 | CAN0 Message Slot 10 Data 0 (C0MSL10DT0) | CAN0 Message Slot 10 Data 1 (C0MSL10DT1) |

Blank addresses are reserved areas.

**Figure 3.4.13 Register Mapping of the SFR Area (11)**

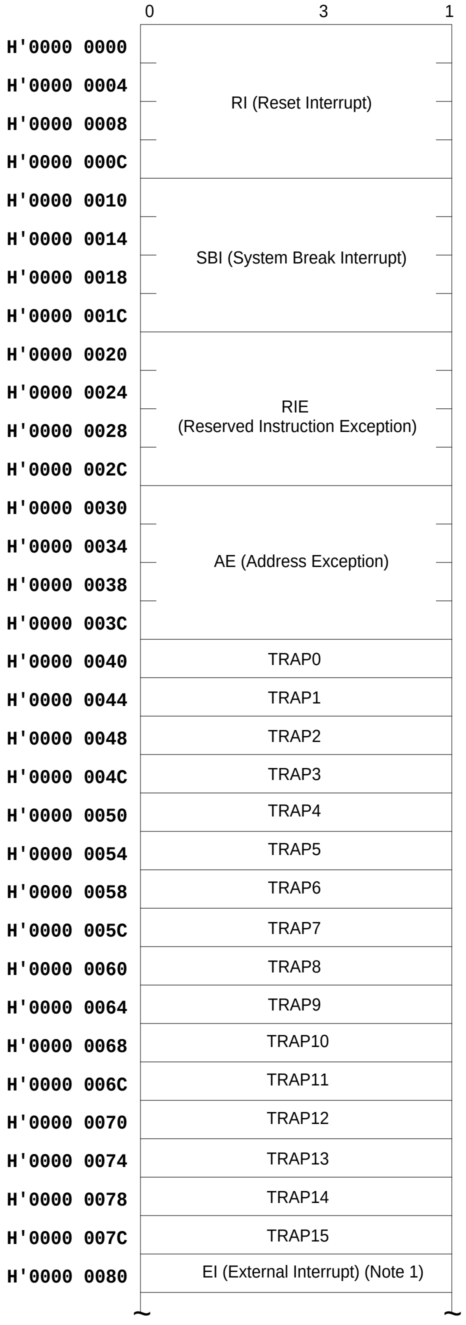
| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 11A8 | CAN0 Message Slot 10 Data 2 (C0MSL10DT2) | CAN0 Message Slot 10 Data 3 (C0MSL10DT3) |
| H'0080 11AA | CAN0 Message Slot 10 Data 4 (C0MSL10DT4) | CAN0 Message Slot 10 Data 5 (C0MSL10DT5) |
| H'0080 11AC | CAN0 Message Slot 10 Data 6 (C0MSL10DT6) | CAN0 Message Slot 10 Data 7 (C0MSL10DT7) |
| H'0080 11AE | CAN0 Message Slot 10 Time Stamp (C0MSL10TSP) | |
| H'0080 11B0 | CAN0 Message Slot 11 Standard ID0 (C0MSL11SID0) | CAN0 Message Slot 11 Standard ID1 (C0MSL11SID1) |
| H'0080 11B2 | CAN0 Message Slot 11 Extended ID0 (C0MSL11EID0) | CAN0 Message Slot 11 Extended ID1 (C0MSL11EID1) |
| H'0080 11B4 | CAN0 Message Slot 11 Extended ID2 (C0MSL11EID2) | CAN0 Message Slot 11 Data Length Register (C0MSL11DLC) |
| H'0080 11B6 | CAN0 Message Slot 11 Data 0 (C0MSL11DT0) | CAN0 Message Slot 11 Data 1 (C0MSL11DT1) |
| H'0080 11B8 | CAN0 Message Slot 11 Data 2 (C0MSL11DT2) | CAN0 Message Slot 11 Data 3 (C0MSL11DT3) |
| H'0080 11BA | CAN0 Message Slot 11 Data 4 (C0MSL11DT4) | CAN0 Message Slot 11 Data 5 (C0MSL11DT5) |
| H'0080 11BC | CAN0 Message Slot 11 Data 6 (C0MSL11DT6) | CAN0 Message Slot 11 Data 7 (C0MSL11DT7) |
| H'0080 11BE | CAN0 Message Slot 11 Time Stamp (C0MSL11TSP) | |
| H'0080 11C0 | CAN0 Message Slot 12 Standard ID0 (C0MSL12SID0) | CAN0 Message Slot 12 Standard ID1 (C0MSL12SID1) |
| H'0080 11C2 | CAN0 Message Slot 12 Extended ID0 (C0MSL12EID0) | CAN0 Message Slot 12 Extended ID1 (C0MSL12EID1) |
| H'0080 11C4 | CAN0 Message Slot 12 Extended ID2 (C0MSL12EID2) | CAN0 Message Slot 12 Data Length Register (C0MSL12DLC) |
| H'0080 11C6 | CAN0 Message Slot 12 Data 0 (C0MSL12DT0) | CAN0 Message Slot 12 Data 1 (C0MSL12DT1) |
| H'0080 11C8 | CAN0 Message Slot 12 Data 2 (C0MSL12DT2) | CAN0 Message Slot 12 Data 3 (C0MSL12DT3) |
| H'0080 11CA | CAN0 Message Slot 12 Data 4 (C0MSL12DT4) | CAN0 Message Slot 12 Data 5 (C0MSL12DT5) |
| H'0080 11CC | CAN0 Message Slot 12 Data 6 (C0MSL12DT6) | CAN0 Message Slot 12 Data 7 (C0MSL12DT7) |
| H'0080 11CE | CAN0 Message Slot 12 Time Stamp (C0MSL12TSP) | |
| H'0080 11D0 | CAN0 Message Slot 13 Standard ID0 (C0MSL13SID0) | CAN0 Message Slot 13 Standard ID1 (C0MSL13SID1) |
| H'0080 11D2 | CAN0 Message Slot 13 Extended ID0 (C0MSL13EID0) | CAN0 Message Slot 13 Extended ID1 (C0MSL13EID1) |
| H'0080 11D4 | CAN0 Message Slot 13 Extended ID2 (C0MSL13EID2) | CAN0 Message Slot 13 Data Length Register (C0MSL13DLC) |
| H'0080 11D6 | CAN0 Message Slot 13 Data 0 (C0MSL13DT0) | CAN0 Message Slot 13 Data 1 (C0MSL13DT1) |
| H'0080 11D8 | CAN0 Message Slot 13 Data 2 (C0MSL13DT2) | CAN0 Message Slot 13 Data 3 (C0MSL13DT3) |
| H'0080 11DA | CAN0 Message Slot 13 Data 4 (C0MSL13DT4) | CAN0 Message Slot 13 Data 5 (C0MSL13DT5) |
| H'0080 11DC | CAN0 Message Slot 13 Data 6 (C0MSL13DT6) | CAN0 Message Slot 13 Data 7 (C0MSL13DT7) |
| H'0080 11DE | CAN0 Message Slot 13 Time Stamp (C0MSL13TSP) | |
| H'0080 11E0 | CAN0 Message Slot 14 Standard ID0 (C0MSL14SID0 | CAN0 Message Slot 14 Standard ID1 (C0MSL14SID1) |
| H'0080 11E2 | CAN0 Message Slot 14 Extended ID0 (C0MSL14EID0) | CAN0 Message Slot 14 Extended ID1 (C0MSL14EID1) |
| H'0080 11E4 | CAN0 Message Slot 14 Extended ID2 (C0MSL14EID2) | CAN0 Message Slot 14 Data Length Register (C0MSL14DLC) |
| H'0080 11E6 | CAN0 Message Slot 14 Data 0 (C0MSL14DT0) | CAN0 Message Slot 14 Data 1 (C0MSL14DT1) |
| H'0080 11E8 | CAN0 Message Slot 14 Data 2 (C0MSL14DT2) | CAN0 Message Slot 14 Data 3 (C0MSL14DT3) |
| H'0080 11EA | CAN0 Message Slot 14 Data 4 (C0MSL14DT4) | CAN0 Message Slot 14 Data 5 (C0MSL14DT5) |
| H'0080 11EC | CAN0 Message Slot 14 Data 6 (C0MSL14DT6) | CAN0 Message Slot 14 Data 7 (C0MSL14DT7) |
| H'0080 11EE | CAN0 Message Slot 14 Time Stamp (C0MSL14TSP) | |
| H'0080 11F0 | CAN0 Message Slot 15 Standard ID0 (C0MSL15SID0) | CAN0 Message Slot 15 Standard ID1 (C0MSL15SID1) |
| H'0080 11F2 | CAN0 Message Slot 15 Extended ID0 (C0MSL15EID0) | CAN0 Message Slot 15 Extended ID1 (C0MSL15EID1) |
| H'0080 11F4 | CAN0 Message Slot 15 Extended ID2 (C0MSL15EID2) | CAN0 Message Slot 15 Data Length Register (C0MSL15DLC) |
| H'0080 11F6 | CAN0 Message Slot 15 Data 0 (C0MSL15DT0) | CAN0 Message Slot 15 Data 1 (C0MSL15DT1) |
| H'0080 11F8 | CAN0 Message Slot 15 Data 2 (C0MSL15DT2) | CAN0 Message Slot 15 Data 3 (C0MSL15DT3) |
| H'0080 11FA | CAN0 Message Slot 15 Data 4 (C0MSL15DT4) | CAN0 Message Slot 15 Data 5 (C0MSL15DT5) |
| H'0080 11FC | CAN0 Message Slot 15 Data 6 (C0MSL15DT6) | CAN0 Message Slot 15 Data 7 (C0MSL15DT7) |
| H'0080 11FE | CAN0 Message Slot 15 Time Stamp (C0MSL11TSP) | |
| | | |
| H'0080 3FFE | | |

Blank addresses are reserved areas.

**Figure 3.4.14 Register Mapping of the SFR Area (12)**

## 3.5 EIT Vector Entry

The EIT vector entry is located at the beginning of the internal ROM/external extension areas. Instructions for branching to the start addresses of respective EIT event handlers are written here. Note that it is branch instructions and not the jump addresses that are written here. For details, refer to Chapter 4, "EIT."



Note 1: When flash entry bit = 1 (i.e., flash enable mode), the EI vector entry is at H'0080 4000.

**Figure 3.5.1 EIT Vector Entry**

## 3.6 ICU Vector Table

The ICU vector table is used by the internal interrupt controller. The start addresses of interrupt handlers for the interrupt requests from respective internal peripheral I/Os are set at the addresses shown below. For details, refer to Chapter 5, "Interrupt Controller."

The 32171's ICU vector table is shown in Figures 3.6.1 and 3.6.2.

| Address | D0 +0 Address D7 | D8 +1 Address D15 |
|---|---|---|
| H'0000 0094 | MJT Input Interrupt 4 Handler Start Address (A0-A15) | |
| H'0000 0096 | MJT Input Interrupt 4 Handler Start Address (A16-A31) | |
| H'0000 0098 | MJT Input Interrupt 3 Handler Start Address (A0-A15) | |
| H'0000 009A | MJT Input Interrupt 3 Handler Start Address (A16-A31) | |
| H'0000 009C | MJT Input Interrupt 2 Handler Start Address (A0-A15) | |
| H'0000 009E | MJT Input Interrupt 2 Handler Start Address (A16-A31) | |
| H'0000 00A0 | MJT Input Interrupt 1 Handler Start Address (A0-A15) | |
| H'0000 00A2 | MJT Input Interrupt 1 Handler Start Address (A16-A31) | |
| H'0000 00A4 | | |
| H'0000 00A6 | | |
| H'0000 00A8 | MJT Output Interrupt 7 Handler Start Address (A0-A15) | |
| H'0000 00AA | MJT Output Interrupt 7 Handler Start Address (A16-A31) | |
| H'0000 00AC | MJT Output Interrupt 6 Handler Start Address (A0-A15) | |
| H'0000 00AE | MJT Output Interrupt 6 Handler Start Address (A16-A31) | |
| H'0000 00B0 | MJT Output Interrupt 5 Handler Start Address (A0-A15) | |
| H'0000 00B2 | MJT Output Interrupt 5 Handler Start Address (A16-A31) | |
| H'0000 00B4 | MJT Output Interrupt 4 Handler Start Address (A0-A15) | |
| H'0000 00B6 | MJT Output Interrupt 4 Handler Start Address (A16-A31) | |
| H'0000 00B8 | MJT Output Interrupt 3 Handler Start Address (A0-A15) | |
| H'0000 00BA | MJT Output Interrupt 3 Handler Start Address (A16-A31) | |
| H'0000 00BC | MJT Output Interrupt 2 Handler Start Address (A0-A15) | |
| H'0000 00BE | MJT Output Interrupt 2 Handler Start Address (A16-A31) | |
| H'0000 00C0 | MJT Output Interrupt 1 Handler Start Address (A0-A15) | |
| H'0000 00C2 | MJT Output Interrupt 1 Handler Start Address (A16-A31) | |
| H'0000 00C4 | MJT Output Interrupt 0 Handler Start Address (A0-A15) | |
| H'0000 00C6 | MJT Output Interrupt 0 Handler Start Address (A16-A31) | |

Blank addresses are reserved areas.

**Figure 3.6.1 ICU Vector Table of the 32171 (1/2)**

| Address | D0 +0 Address D7 D8 +1 Address D15 |
|---|---|
| H'0000 00C8 | DMA0-4 Interrupt Handler Start Address (A0-A15) |
| H'0000 00CA | DMA0-4 Interrupt Handler Start Address (A16-A31) |
| H'0000 00CC | SIO1 Receive Interrupt Handler Start Address (A0-A15) |
| H'0000 00CE | SIO1 Receive Interrupt Handler Start Address (A16-A31) |
| H'0000 00D0 | SIO1 Transmit Interrupt Handler Start Address (A0-A15) |
| H'0000 00D2 | SIO1 Transmit Interrupt Handler Start Address (A16-A31) |
| H'0000 00D4 | SIO0 Receive Interrupt Handler Start Address (A0-A15) |
| H'0000 00D6 | SIO0 Receive Interrupt Handler Start Address (A16-A31) |
| H'0000 00D8 | SIO0 Transmit Interrupt Handler Start Address (A0-A15) |
| H'0000 00DA | SIO0 Transmit Interrupt Handler Start Address (A16-A31) |
| H'0000 00DC | A-D0 Conversion Interrupt Handler Start Address (A0-A15) |
| H'0000 00DE | A-D0 Conversion Interrupt Handler Start Address (A16-A31) |
| H'0000 00E0 | |
| H'0000 00E2 | |
| H'0000 00E4 | |
| H'0000 00E6 | |
| H'0000 00E8 | DMA5-9 Interrupt Handler Start Address (A0-A15) |
| H'0000 00EA | DMA5-9 Interrupt Handler Start Address (A16-A31) |
| H'0000 00EC | SIO2,3 Transmit/Receive Interrupt Handler Start Address (A0-A15) |
| H'0000 00EE | SIO2,3 Transmit/Receive Interrupt Handler Start Address (A16-A31) |
| H'0000 00F0 | RTD Interrupt Handler Start Address (A0-A15) |
| H'0000 00F2 | RTD Interrupt Handler Start Address (A16-A31) |
| H'0000 00F4 | |
| H'0000 00F6 | |
| H'0000 00F8 | |
| H'0000 00FA | |
| H'0000 00FC | |
| H'0000 00FE | |
| H'0000 0100 | |
| H'0000 0102 | |
| H'0000 0104 | |
| H'0000 0106 | |
| H'0000 0108 | |
| H'0000 010A | |
| H'0000 010C | CAN0 Transmit/Receive & Error Interrupt Handler Start Address (A0-A15) |
| H'0000 010E | CAN0 Transmit/Receive & Error Interrupt Handler Start Address (A16-A31) |

Blank addresses are reserved areas.

**Figure 3.6.2 ICU Vector Table of the 32171 (2/2)**

## 3.7 Notes on Address Space

- **Virtual flash emulation function**

The 32171 can map one 8-Kbyte block of internal RAM beginning with the start address into one of 8-Kbyte areas (L banks) of the internal flash memory and can map up to two 4-Kbyte blocks of internal RAM beginning with address H’0080 6000 into one of 4-Kbyte areas (S banks) of the internal flash memory. This capability is referred to as the “virtual-flash emulation” function. For details about this function, refer to Section 6.7, “Virtual-Flash Emulation Function.”

# CHAPTER 4
# EIT

4.1 Outline of EIT
4.2 EIT Events
4.3 EIT Processing Procedure
4.4 EIT Processing Mechanism
4.5 Acceptance of EIT Events
4.6 Saving and Restoring the PC and PSW
4.7 EIT Vector Entry
4.8 Exception Processing
4.9 Interrupt Processing
4.10 Trap Processing
4.11 EIT Priority Levels
4.12 Example of EIT Processing
4.13 Precautions on EIT

## 4.1 Outline of EIT

If some event occurs when the CPU is executing an ordinary program, it may become necessary to suspend the program being executed and execute another program. Events like this one are referred to by a generic name as EIT (Exception, Interrupt, and Trap).

### (1) Exception

This is an event related to the context being executed. It is generated by an error or violation during instruction execution. In the M32R/ECU, this type of event includes Address Exception (AE) and Reserved Instruction Exception (RIE).

### (2) Interrupt

This is an event generated irrespective of the context being executed. It is generated in hardware by a signal from an external source. In the M32R/ECU, this type of event includes External Interrupt (EI), System Break Interrupt (SBI), and Reset Interrupt (RI).

### (3) Trap

This refers to a software interrupt generated by executing a TRAP instruction. This type of event is intentionally generated in a program as in the OS's system call by the programmer.

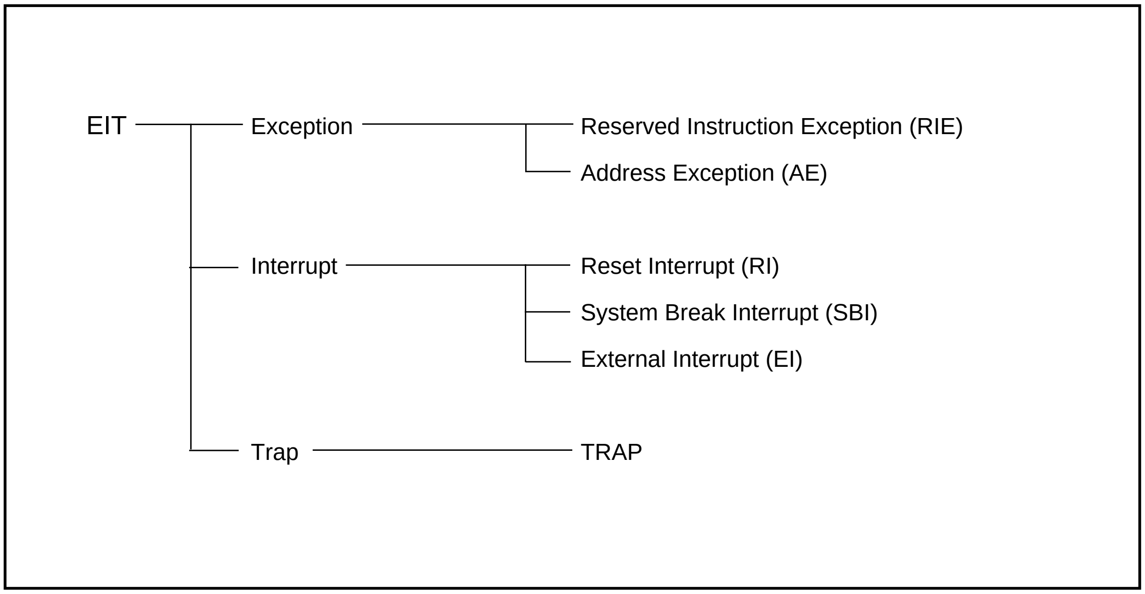


**Figure 4.1.1 Classification of EITs**

## 4.2 EIT Events

### 4.2.1 Exception

**(1) Reserved Instruction Exception (RIE)**

Reserved Instruction Exception (RIE) is generated when execution of a reserved instruction (unimplemented instruction) is detected.

**(2) Address Exception (AE)**

Address Exception (AE) is generated when an attempt is made to access a misaligned address in Load or Store instructions.

### 4.2.2 Interrupt

**(1) Reset Interrupt (RI)**

Reset Interrupt (RI) is always accepted by entering the $\overline{\text{RESET}}$ signal. The reset interrupt is assigned the highest priority.

**(2) System Break Interrupt (SBI)**

System Break Interrupt (SBI) is an emergency interrupt which is used when power outage is detected or a fault condition is notified by an external watchdog timer. This interrupt can only be used in cases when after interrupt processing, control will not return to the program that was being executed when the interrupt occurred.

**(3) External Interrupt (EI)**

External Interrupt (EI) is requested from internal peripheral I/Os managed by the interrupt controller. The 32171's internal interrupt controller manages these interrupts by assigning each one of eight priority levels including an interrupt-disabled state.

### 4.2.3 Trap

Traps are software interrupts which are generated by executing the TRAP instruction. Sixteen distinct vector addresses are provided corresponding to TRAP instruction operands 0-15.

## 4.3 EIT Processing Procedure

EIT processing consists of two parts, one in which they are handled automatically by hardware, and one in which they are handled by user-created programs (EIT handlers). The procedure for processing EITs when accepted, except for a rest interrupt, is shown below.

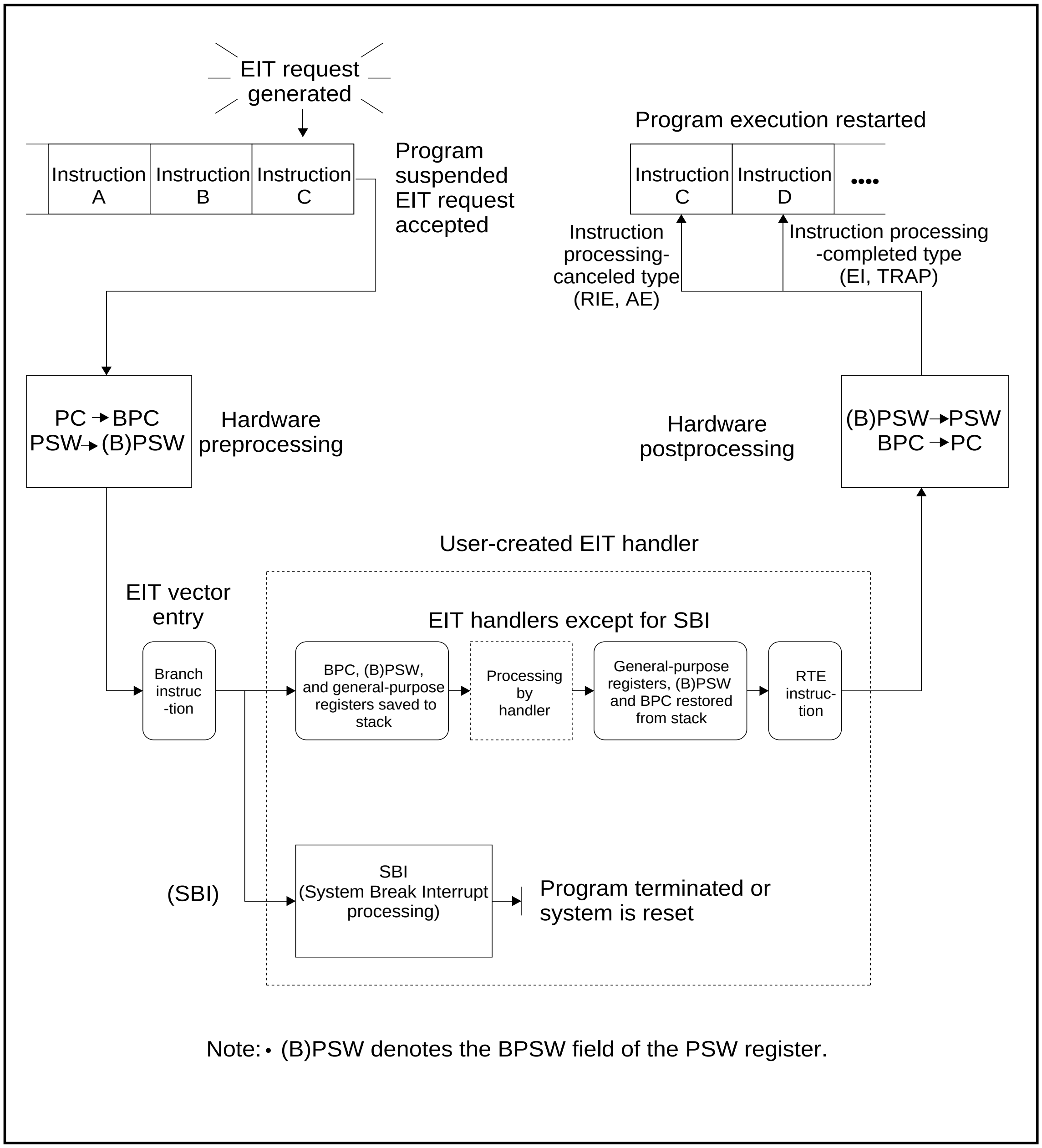

EIT request generated
Instruction A
Instruction B
Instruction C
Program suspended
EIT request accepted
Program execution restarted
Instruction C
Instruction D
....
Instruction processing-canceled type (RIE, AE)
Instruction processing -completed type (EI, TRAP)
PC → BPC
PSW → (B)PSW
Hardware preprocessing
Hardware postprocessing
(B)PSW → PSW
BPC → PC
User-created EIT handler
EIT vector entry
EIT handlers except for SBI
Branch instruc -tion
BPC, (B)PSW, and general-purpose registers saved to stack
Processing by handler
General-purpose registers, (B)PSW and BPC restored from stack
RTE instruc-tion
(SBI)
SBI (System Break Interrupt processing)
Program terminated or system is reset
Note: • (B)PSW denotes the BPSW field of the PSW register.

**Figure 4.3.1 Outline of EIT Processing Procedure**

When an EIT is accepted, the M32R/ECU saves the PC and PSW (as will be described later) and branches to the EIT vector. The EIT vector has an entry address assigned for each EIT. This is where the BRA (branch) instruction (note that these are not branch address) for the EIT handler is written.

In the M32R/ECU's hardware preprocessing, only the contents of the PC and PSW registers are transferred to the backup registers (BPC register and the BPSW field of the PSW register), and no other operations are performed. Therefore, please make sure the BPC register, the PSW register (including the BPSW field), and the general-purpose registers to be used in the EIT handler are saved to the stack by the EIT handler you write. (Remember that these registers must be saved to the stack in a program by the user.)

When processing by the EIT handler is completed, restore the saved registers from the stack and finally execute the "RTE" instruction. Control is thereby returned from EIT processing to the program that was being executed when the EIT occurred. (This does not apply to the System Break Interrupt, however.)
In the M32R/ECU's hardware postprocessing, the contents of the backup registers (BPC register and the BPSW field of the PSW register) are moved back to the PC and PSW registers.

## 4.4 EIT Processing Mechanism

The M32R/ECU's EIT processing mechanism consists of the M32R CPU core and the interrupt controller for internal peripheral I/Os. It also has the backup registers for the PC and PSW (BPC register and the BPSW field of the PSW register). The M32R/ECU's internal EIT processing mechanism is shown below.

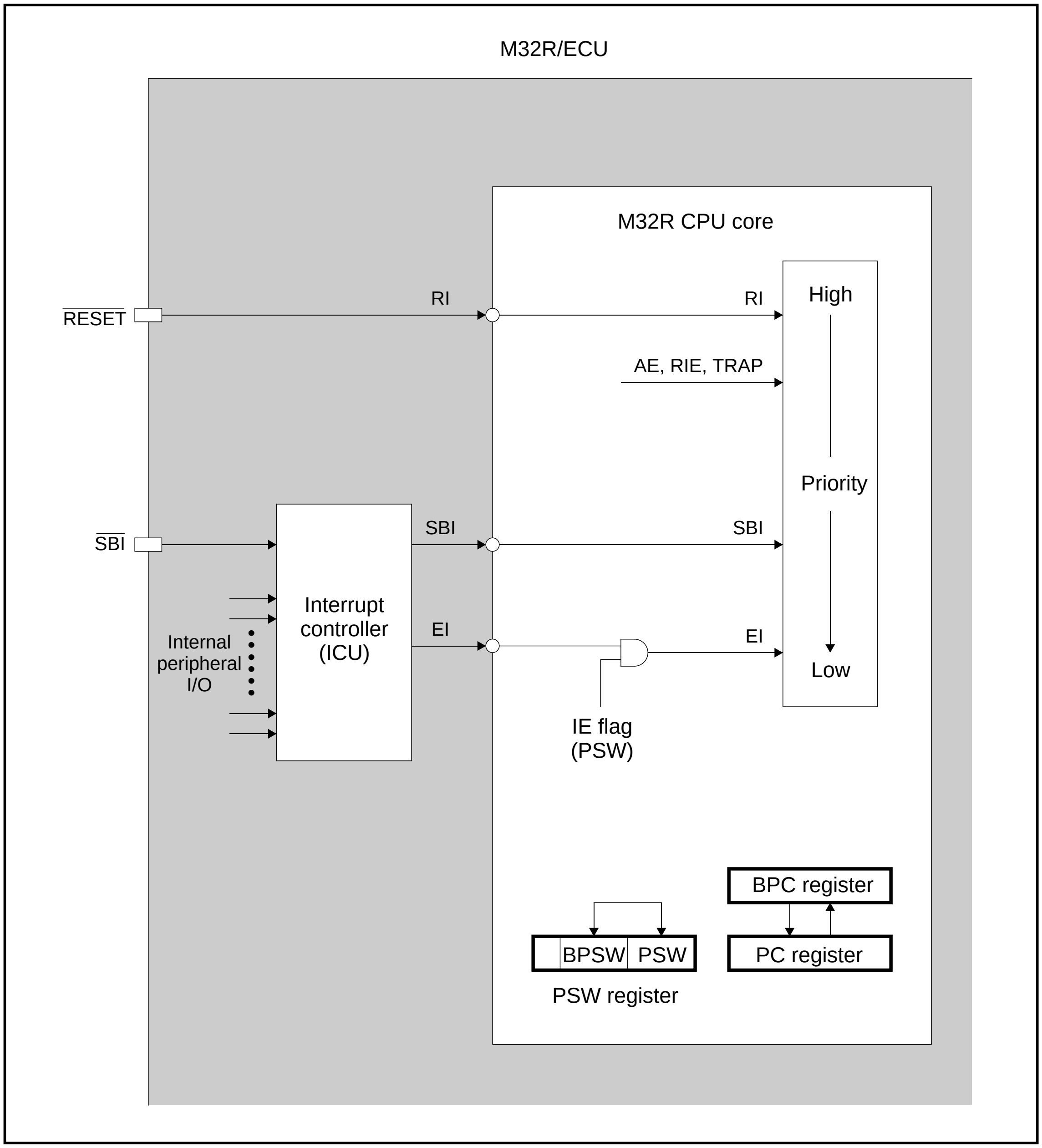


**Figure 4.4.1 The M32R/ECU's EIT Processing Mechanism**

## 4.5 Acceptance of EIT Events

When an EIT event occurs, the M32R/ECU suspends the program it has hitherto been executing and branches to EIT processing by the relevant handler. Conditions under which each EIT event occurs and the timing at which they are accepted are shown below.

**Table 4.5.1 Acceptance of EIT Events**

| EIT Event | Type of Processing | Acceptance Timing | Values Set in BPC Register |
|---|---|---|---|
| Reserved Instruction Exception (RIE) | Instruction processing-canceled type | During instruction execution | PC value of the instruction which generated RIE |
| Address Exception (AE) | Instruction processing-canceled type | During instruction execution | PC value of the instruction which generated AE |
| Reset Interrupt (RI) | Instruction processing-aborted type | Each machine cycle | Indeterminate value |
| System Break Interrupt (SBI) | Instruction processing-completed type | Break in instructions (only word boundaries) | PC value of the next instruction |
| External Interrupt (EI) | Instruction processing-completed type | Break in instructions (only word boundaries) | PC value of the next instruction |
| Trap (TRAP) | Instruction processing-completed type | Break in instructions | PC value of TRAP instruction + 4 |

## 4.6 Saving and Restoring the PC and PSW

The following describes operation of the M32R at the time when it accepts an EIT and when it executes the "RTE" instruction.

### (1) Hardware preprocessing when an EIT is accepted

(a) Save the SM, IE, and C bits of the PSW register

BSM ← SM<br>
BIE ← IE<br>
BC ← C

(b) Update the SM, IE, and C bits of the PSW register

SM ← Remains unchanged (RIE, AE, TRAP)
or set to 0 (SBI, EI, RI)<br>
IE ← Set to 0<br>
C ← Set to 0

(c) Save the PC register

BPC ← PC

(d) Set the vector address in the PC register

Branches to the EIT vector and executes the branch instruction ("BRA" instruction) written in it, thereby transferring control to the user-created EIT handler.

### (2) Hardware postprocessing when the "RTE" instruction is executed

(e) Restore the SM, IE, and C bits of the PSW register from their backup bits.

SM ← BSM<br>
IE ← BIE<br>
C ← BC

(f) Restore the value of the PC register from the BPC register

PC ← BPC

Note: • The value of the BPC register and those of the BSM, BIE, and BC bits of the PSW register after execution of the "RTE" instruction are indeterminate.

(a) Save SM, IE, and C bits

BSM ← SM
BIE ← IE
BC ← C

(b) Update SM, IE, and C bits

SM ← Unchanged/0
IE ← 0
C ← 0

(c) Save PC

BPC ← PC

(d) Set vector address in PC

PC ← Vector address

(e) Restore BSM, BIE, and BC bits from backup bits

SM ← BSM
IE ← BIE
C ← BC

The values of BSM, BIE, and BC bits after execution of the "RTE" instruction are indeterminate.

(f) Restore PC value from BPC

The value of BPC after execution of the "RTE" instruction is indeterminate.

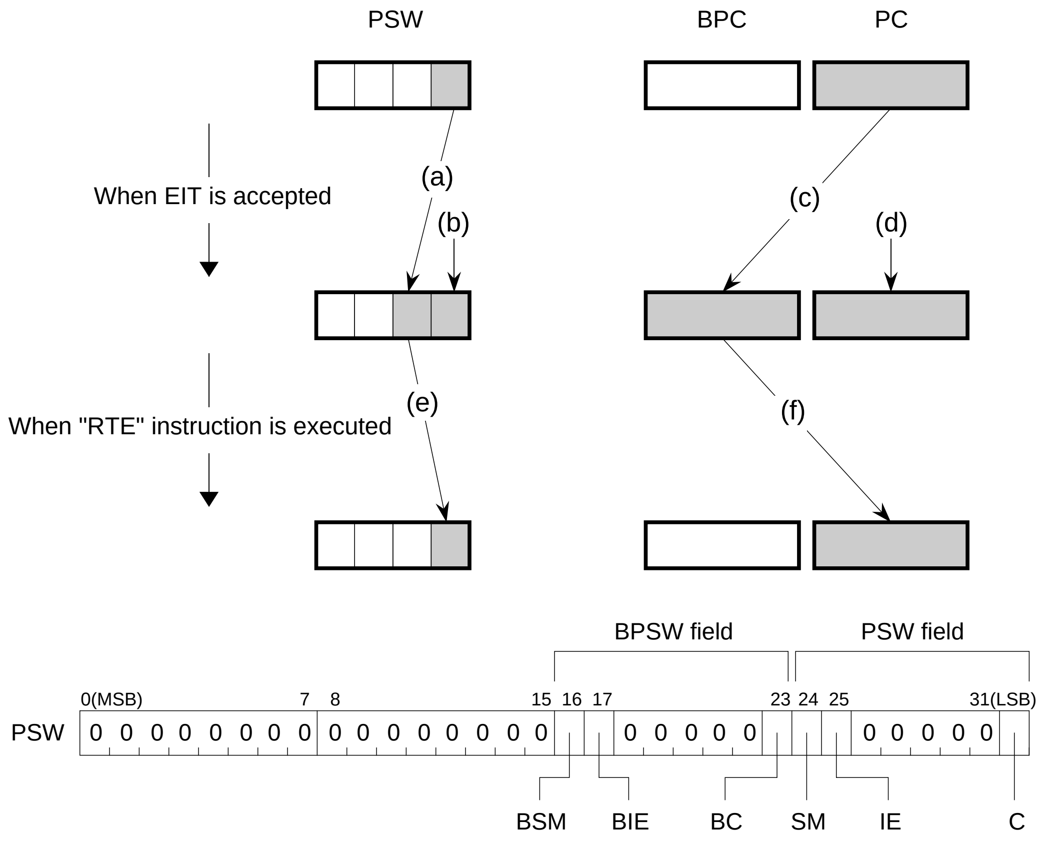


**Figure 4.6.1 Saving and Restoring the PC and PSW**

## 4.7 EIT Vector Entry

The EIT vector entry is located in the user space starting from address H'0000 0000. The table below lists the EIT vector entry.

**Table 4.7.1 EIT Vector Entry**

| Name | Abbreviation | Vector Address | SM | IE | BPC |
|---|---|---|---|---|---|
| Reset Interrupt | RI | H'0000 0000 (Note 1) | 0 | 0 | Indeterminate |
| System Break Interrupt | SBI | H'0000 0010 | 0 | 0 | PC of the next instruction |
| Reserved Instruction Exception | RIE | H'0000 0020 | Indeterminate | 0 | PC of the instruction that generated EIT |
| Address Exception | AE | H'0000 0030 | Indeterminate | 0 | PC of the instruction that generated RIE |
| Trap | TRAP0 | H'0000 0040 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP1 | H'0000 0044 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP2 | H'0000 0048 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP3 | H'0000 004C | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP4 | H'0000 0050 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP5 | H'0000 0054 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP6 | H'0000 0058 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP7 | H'0000 005C | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP8 | H'0000 0060 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP9 | H'0000 0064 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP10 | H'0000 0068 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP11 | H'0000 006C | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP12 | H'0000 0070 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP13 | H'0000 0074 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP14 | H'0000 0078 | Indeterminate | 0 | PC of TRAP instruction + 4 |
| | TRAP15 | H'0000 007C | Indeterminate | 0 | PC of TRAP instruction + 4 |
| External Interrupt | EI | H'0000 0080 (Note 2) | 0 | 0 | PC of the next instruction |

Note 1: During boot mode, this vector address is moved to the beginning of the boot ROM (address H'8000 0000). For details, refer to Section 6.5, "Programming of Internal Flash Memory."

Note 2: During flash E/W enable mode, this vector address is moved to the beginning of the internal RAM (address H'0080 4000). For details, refer to Section 6.5, "Programming of Internal Flash Memory."

## 4.8 Exception Processing

### 4.8.1 Reserved Instruction Exception (RIE)

**[Occurrence Conditions]**

Reserved Instruction Exception (RIE) is generated when execution of a reserved instruction (unimplemented instruction) is detected. Instruction check is performed on the op-code part of the instruction.
When a reserved instruction exception occurs, the instruction which generated it is not executed.
If an external interrupt is requested at the same time a reserved instruction exception is detected, it is the reserved instruction exception that is accepted.

**[EIT Processing]**

(1) Saving SM, IE, and C bits

The SM, IE, and C bits of the PSW register are saved to their backup bits – the BSM, BIE, and BC bits.

| | | |
|---|---|---|
| BSM | ← | SM |
| BIE | ← | IE |
| BC | ← | C |

(2) Updating SM, IE, and C bits

The SM, IE, and C bits of the PSW register are updated as shown below.

| | | |
|---|---|---|
| SM | ← | Unchanged |
| BIE | ← | 0 |
| BC | ← | 0 |

(3) Saving PC

The PC value of the instruction that generated the reserved instruction exception is set in the BPC register. For example, if the instruction that generated the reserved instruction exception is at address 4, the value 4 is set in the BPC register. Similarly, if the instruction is at address 6, the value 6 is set in the BPC register. In this case, the value of the BPC register bit 30 indicates whether the instruction that generated the reserved instruction exception resides on a word boundary (BPC[30] = 0) or not on a word boundary (BPC[30] = 1).
However, in either case of the above, the address to which the "RTE" instruction returns after completion of processing by the EIT handler is address 4. (This is because the two low-order bits are cleared to "00" when returning to the PC.)

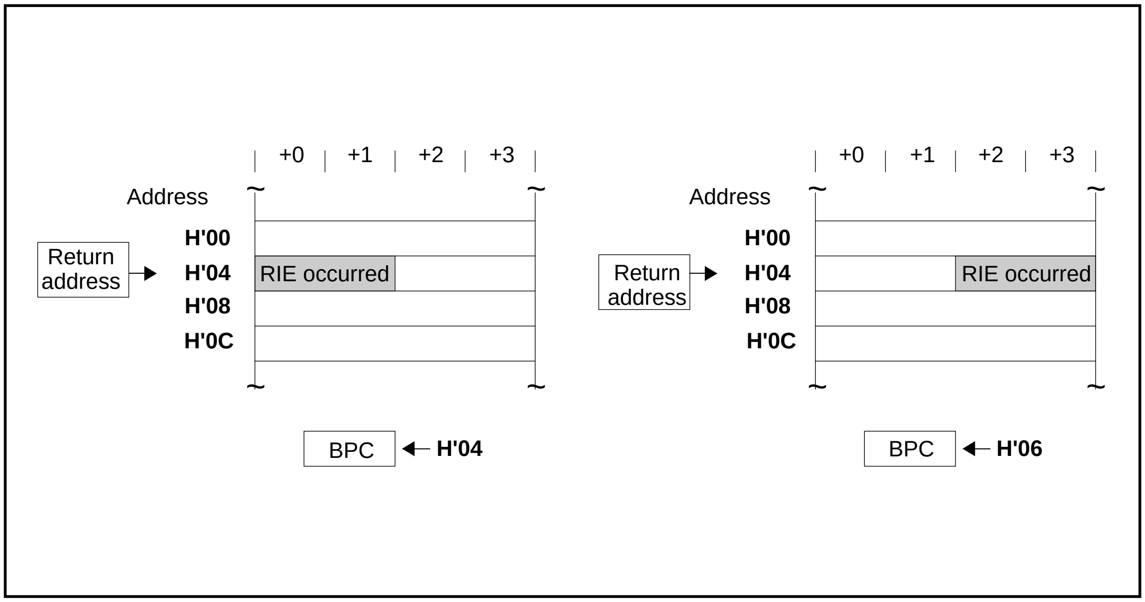


**Figure 4.8.1 Example of a Return Address for Reserved Instruction Exception (RIE)**

(4) Branching to the EIT vector entry

Control branches to the address H'0000 0020 in the user space. This is the last operation performed in hardware preprocessing by the M32R/ECU.

(5) Jumping from the EIT vector entry to the user-created handler

The M32R/ECU executes the "BRA" instruction written at address H'0000 0020 of the EIT vector entry by the user to jump to the start address of the user-created handler. At the beginning of the EIT handler you created, first save the BPC and PSW registers and the necessary general-purpose registers to the stack.

(6) Returning from the EIT handler

At the end of the EIT handler, restore the general-purpose registers and the BPC and PSW registers from the stack and then execute the "RTE" instruction. As you execute the "RTE" instruction, hardware postprocessing is automatically performed by the M32R/ECU.

### 4.8.2 Address Exception (AE)

**[Occurrence Conditions]**

Address Exception (AE) is generated when an attempt is made to access a misaligned address in Load or Store instructions. The following lists the combination of instructions and accessed addresses that may cause address exceptions to occur:

- When the LDH, LDUH, or STH instruction accesssed an address whose two low-order bits are "01" or "11"
- When the LD, ST, LOCK, or UNLOCK instruction accessed an address whose two low-order bits are "01," "10," or "11"

When an address exception occurs, memory access by the instruction that generated the exception is not performed. If an external interrupt is requested at the same time an address exception is detected, it is the address exception that is accepted.

**[EIT Processing]**

(1) Saving SM, IE, and C bits

The SM, IE, and C bits of the PSW register are saved to their backup bits – the BSM, BIE, and BC bits.

| | | |
|---|---|---|
| BSM | ← | SM |
| BIE | ← | IE |
| BC | ← | C |

(2) Updating SM, IE, and C bits

The SM, IE, and C bits of the PSW register are updated as shown below.

| | | |
|---|---|---|
| SM | ← | Unchanged |
| IE | ← | 0 |
| C | ← | 0 |

(3) Saving PC

The PC value of the instruction that generated the address exception is set in the BPC register. For example, if the instruction that generated the address exception is at address 4, the value 4 is set in the BPC register. Similarly, if the instruction is at address 6, the value 6 is set in the BPC register. In this case, the value of the BPC register bit 30 indicates whether the instruction that generated the address exception resides on a word boundary (BPC[30] = 0) or not on a word boundary (BPC[30] = 1).
However, in either case of the above, the address to which the "RTE" instruction returns after completion of processing by the EIT handler is address 4. (This is because the two low-order bits are cleared to "00" when returning to the PC.)

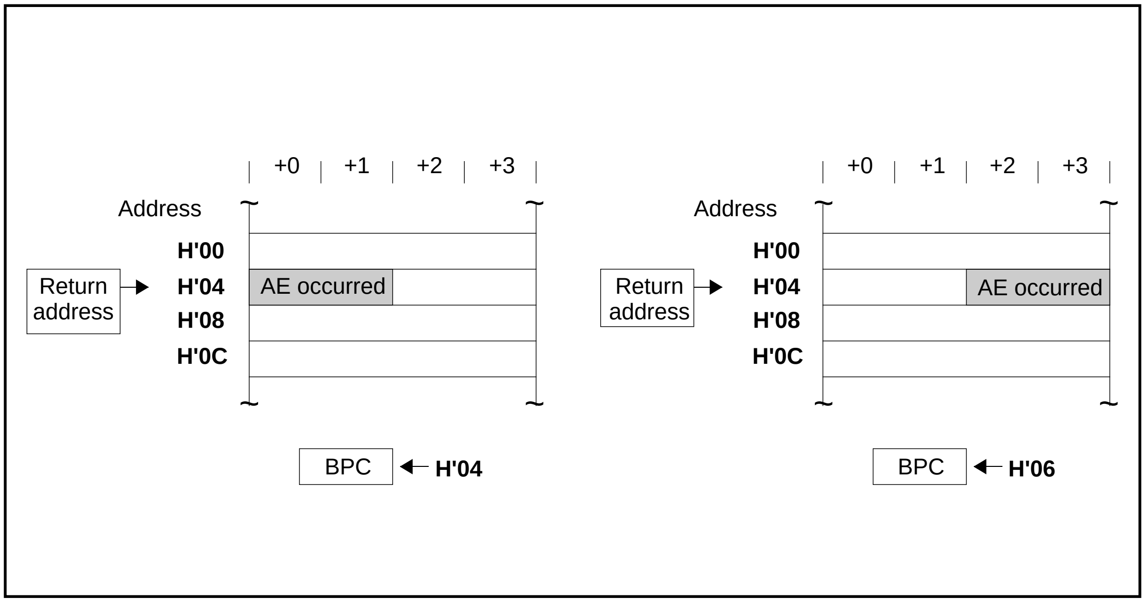


**Figure 4.8.2 Example of a Return Address for Address Exception (AE)**

(4) Branching to the EIT vector entry

Control branches to the address H'0000 0030 in the user space. This is the last operation performed in hardware preprocessing by the M32R/ECU.

(5) Jumping from the EIT vector entry to the user-created handler

The M32R/ECU executes the "BRA" instruction written at address H'0000 0030 of the EIT vector entry by the user to jump to the start address of the user-created handler. At the beginning of the EIT handler you created, first save the BPC and PSW registers and the necessary general-purpose registers to the stack.

(6) Returning from the EIT handler

At the end of the EIT handler, restore the general-purpose registers and the BPC and PSW registers from the stack and then execute the "RTE" instruction. As you execute the "RTE" instruction, hardware postprocessing is automatically performed by the M32R/ECU.

# 4.9 Interrupt Processing

## 4.9.1 Reset Interrupt (RI)

**[Occurrence Conditions]**

Reset Interrupt (RI) is unconditionally accepted in any machine cycle by pulling the $\overline{\text{RESET}}$ input signal low. The reset interrupt is assigned the highest priority among all EITs.

**[EIT Processing]**

(1) Initializing SM, IE, and C bits

The SM, IE, and C bits of the PSW register are initialized in the manner shown below.

| | | |
|---|---|---|
| SM | ← | 0 |
| IE | ← | 0 |
| C | ← | 0 |

For the reset interrupt, the values of BSM, BIE, and BC bits are indeterminate.

(2) Branching to the EIT vector entry

Control branches to the address H'0000 0000 in the user space. However, when operating in boot mode, control goes to the beginning of the boot ROM (address H'8000 0000). For details, refer to Section 6.5, "Programming of Internal Flash Memory."

(3) Jumping from the EIT vector entry to the user program

The M32R/ECU executes the instruction written at address H'0000 0000 of the EIT vector entry by the user. In the reset vector entry, be sure to initialize the PSW and SPI registers before jumping to the start address of the program you created.

### 4.9.2 System Break Interrupt (SBI)

System Break Interrupt (SBI) is an emergency interrupt which is used when power outage is detected or a fault condition is notified by an external watchdog timer. The system break interrupt cannot be masked by the PSW register IE bit. Therefore, the system break interrupt can only be used when some fatal event has already occurred to the system when the interrupt is detected. Also, this interrupt must be used on condition that after processing by the SBI handler, control will not return to the program that was being executed when the system break interrupt occurred.

**[Occurrence Conditions]**

A system break interrupt is accepted by a falling edge on $\overline{\text{SBI}}$ input pin. (The system break interrupt cannot be masked by the PSW register IE bit.)

In no case will a system break interrupt be activated immediately after executing a 16-bit instruction that starts from a word boundary. (For 16-bit branch instructions, however, the interrupt may be accepted immediately after branching.)

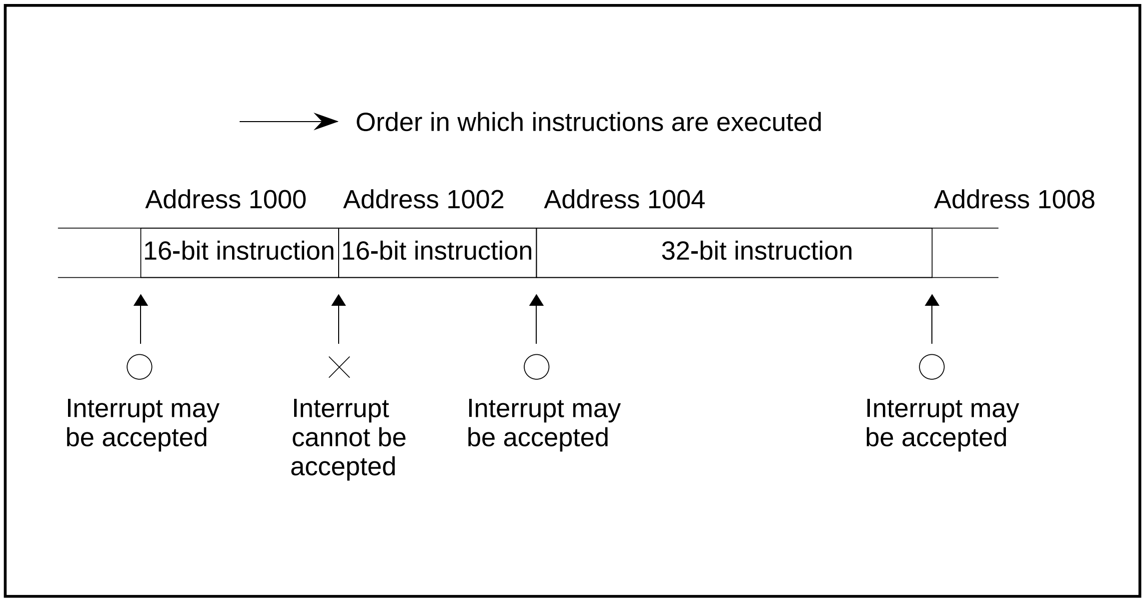

Order in which instructions are executed
Address 1000
Address 1002
Address 1004
Address 1008
16-bit instruction
16-bit instruction
32-bit instruction
○
×
○
○
Interrupt may be accepted
Interrupt cannot be accepted
Interrupt may be accepted
Interrupt may be accepted

**Figure 4.9.1 Timing at Which System Break Interrupt (SBI) is Accepted**

**[EIT Processing]**

(1) Saving SM, IE, and C bits

The SM, IE, and C bits of the PSW register are saved to their backup bits-the BSM, BIE, and BC bits.

BSM ← SM
BIE ← IE
BC ← C

(2) Updating SM, IE, and C bits

The SM, IE, and C bits of the PSW register are updated as shown below.

SM ← 0
IE ← 0
C ← 0

(3) Saving PC

The content (always word boundary) of the PC register is saved to the BPC register.

(4) Branching to the EIT vector entry

Control branches to the address H'0000 0010 in the user space. This is the last operation performed in hardware preprocessing by the M32R/ECU.

(5) Jumping from the EIT vector entry to the user-created handler

The M32R/ECU executes the "BRA" instruction written at address H'0000 0010 of the EIT vector entry by the user to jump to the start address of the user-created handler. The system break interrupt can only be used when some fatal event has occurred to the system. Also, this interrupt must be used on condition that after processing by the SBI handler, control will not return to the program that was being executed when the system break interrupt occurred.

### 4.9.3 External Interrupt (EI)

An external interrupt is generated upon an interrupt request which is output by the 32171's internal interrupt controller. The interrupt controller manages interrupt requests by assigning each one of seven priority levels. For details, refer to Chapter 5, "Interrupt Controller." For details about the interrupt sources, refer to each section in which the relevant internal peripheral I/O is described.

**[Occurrence Conditions]**

External interrupts are managed based on interrupt requests from each internal peripheral I/O by the 32171's internal interrupt controller. These interrupt requests are notified to the M32R CPU by the interrupt controller. The M32R/ECU checks these interrupt requests at a break in instructions residing on word boundaries, and when an interrupt request is detected and the PSW register IE flag = 1, accepts it as an external interrupt.
In no case will an external interrupt be activated immediately after executing a 16-bit instruction that starts from a word boundary. (For 16-bit branch instructions, however, the interrupt may be accepted immediately after branching.)

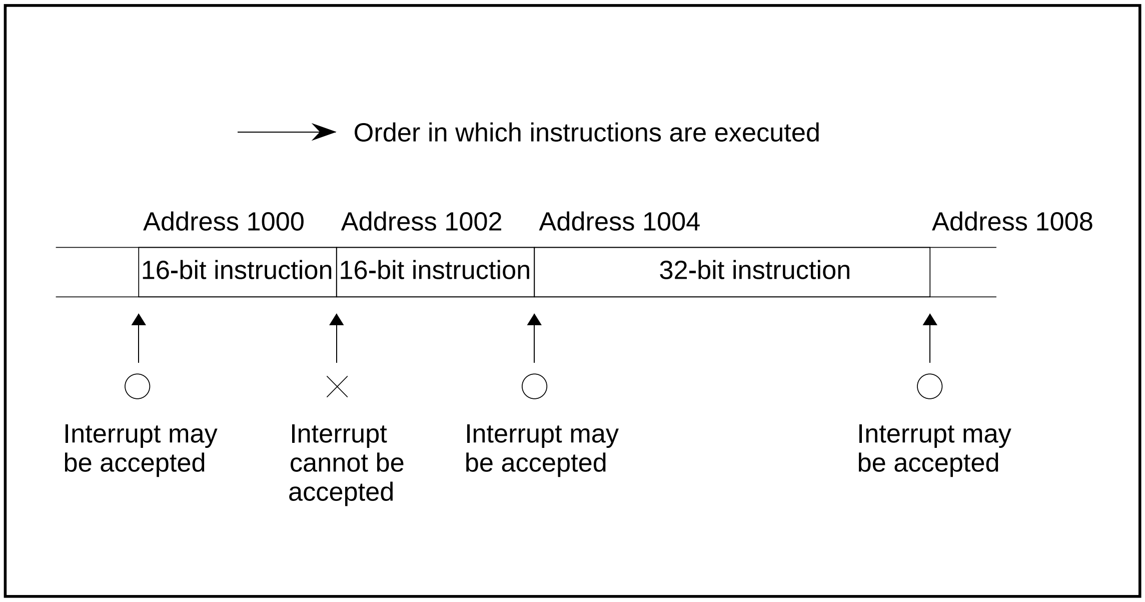

Order in which instructions are executed
Address 1000
Address 1002
Address 1004
Address 1008
16-bit instruction
16-bit instruction
32-bit instruction
○
Interrupt may be accepted
×
Interrupt cannot be accepted
○
Interrupt may be accepted
○
Interrupt may be accepted

**Figure 4.9.2 Timing at Which External Interrupt (EI) is Accepted**

**[EIT Processing]**

(1) Saving SM, IE, and C bits

The SM, IE, and C bits of the PSW register are saved to their backup bits – the BSM, BIE, and BC bits.

| | | |
|---|---|---|
| BSM | ← | SM |
| BIE | ← | IE |
| BC | ← | C |

(2) Updating SM, IE, and C bits

The SM, IE, and C bits of the PSW register are updated as shown below.

| | | |
|---|---|---|
| SM | ← | 0 |
| IE | ← | 0 |
| C | ← | 0 |

(3) Saving PC

The content (always word boundary) of the PC register is saved to the BPC register.

(4) Branching to the EIT vector entry

Control branches to the address H'0000 0080 in the user space. However, when operating in flash E/W enable mode, control goes to the beginning of the internal RAM (address H'0080 4000). (For details, refer to Section 6.5, "Writing to Internal Flash Memory.") This is the last operation performed in hardware preprocessing by the M32R/ECU.

(5) Jumping from the EIT vector entry to the user-created handler

The M32R/ECU executes the "BRA" instruction written at address H'0000 0080 of the EIT vector entry by the user to jump to the start address of the user-created handler. At the beginning of the EIT handler you created, first save the BPC and PSW registers and the necessary general-purpose registers to the stack.

(6) Returning from the EIT handler

At the end of the EIT handler, restore the general-purpose registers and the BPC and PSW registers from the stack and then execute the "RTE" instruction. As you execute the "RTE" instruction, hardware postprocessing is automatically performed by the M32R/ECU.

# 4.10 Trap Processing

## 4.10.1 Trap (TRAP)

### [Occurrence Conditions]

Traps refer to software interrupts which are generated by executing the "TRAP" instruction. Sixteen distinct traps are generated, each corresponding to one of "TRAP" instruction operands 0-15. Accordingly, sixteen vector entries are provided.

### [EIT Processing]

(1) Saving SM, IE, and C bits

The SM, IE, and C bits of the PSW register are saved to their backup bits – the BSM, BIE, and BC bits.

| | | |
|---|---|---|
| BSM | ← | SM |
| BIE | ← | IE |
| BC | ← | C |

(2) Updating SM, IE, and C bits

The SM, IE, and C bits of the PSW register are updated as shown below.

| | | |
|---|---|---|
| SM | ← | Unchanged |
| IE | ← | 0 |
| C | ← | 0 |

(3) Saving PC

When the trap instruction is executed, the "PC value of the TRAP instruction + 4" is set in the BPC register. For example, if the "TRAP" instruction is located at address 4, the value H'08 is set in the BPC register. Similarly, if the instruction is located at address 6, the value H'0A is set in the BPC register. In this case, the value of the BPC register bit 30 indicates whether the trap instruction resides on a word boundary (BPC[30] = 0) or not on a word boundary (BPC[30] = 1).
However, in either case of the above, the address to which the "RTE" instruction returns after completion of processing by the EIT handler is address 8. (This is because the two low-order bits are cleared to "00" when returning to the PC.)

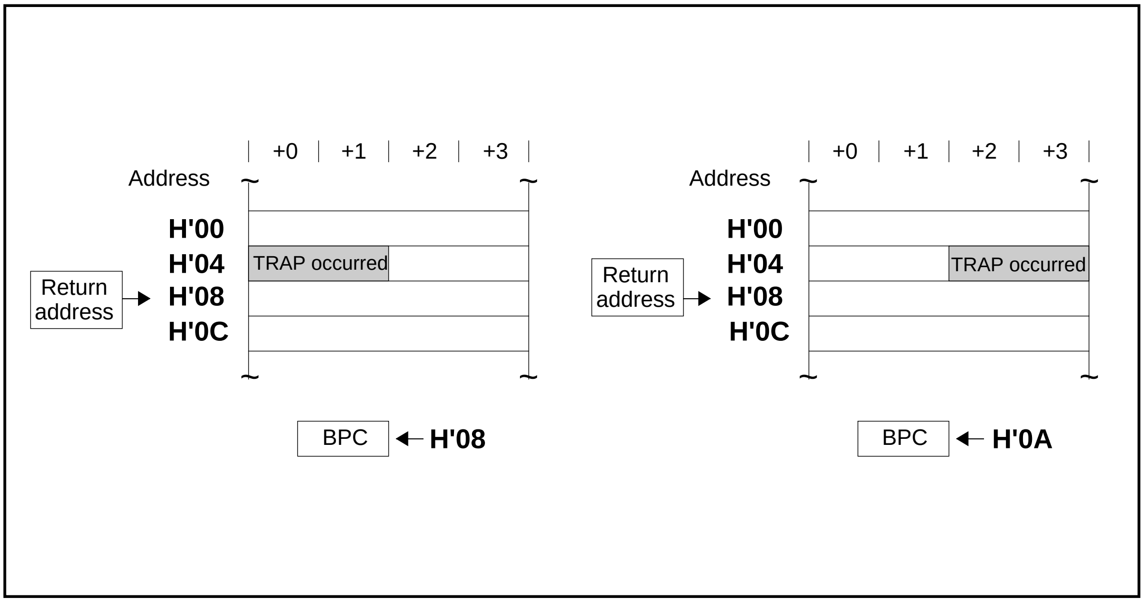


**Figure 4.10.1 Example of a Return Address for Trap (TRAP)**

(4) Branching to the EIT vector entry

Control branches to the addresses H'0000 0040 through H'0000 007C in the user space. This is the last operation performed in hardware preprocessing by the M32R/ECU.

(5) Jumping from the EIT vector entry to the user-created handler

The M32R/ECU executes the "BRA" instruction written at addresses H'0000 0040 through H'0000 007C of the EIT vector entry by the user to jump to the start address of the user-created handler. At the beginning of the EIT handler you created, first save the BPC and PSW registers and the necessary general-purpose registers to the stack.

(6) Returning from the EIT handler

At the end of the EIT handler, restore the general-purpose registers and the BPC and PSW registers from the stack and then execute the "RTE" instruction. As you execute the "RTE" instruction, hardware postprocessing is automatically performed by the M32R/ECU.

## 4.11 EIT Priority Levels

The table below lists the priority levels of EIT events. When multiple EITs occur simultaneously, the event with the highest priority is accepted first.

Table 4.11.1 Priority of EIT Events and How Returned from EIT

| Priority | EIT Event | Type of Processing | Values Set in BPC Register |
|---|---|---|---|
| 1(Highest) | Reset Interrupt (RI) | Instruction processing -aborted type | Indeterminate |
| 2 | Address Exception (AE) | Instruction processing-canceled type | PC of the instruction that generated AE |
|  | Reserved Instruction Exception (RIE) | Instruction processing-canceled type | PC of the instruction that generated AE |
|  | Trap (TRAP) | Instruction processing-completed type | TRAP instruction + 4 |
| 3 | System Break Interrupt (SBI) | Instruction processing-completed type | PC of the next instruction |
| 4 | External Interrupt (EI) | Instruction processing-completed type | PC of the next instruction |

Note that for External Interrupt (EI), the priority levels of interrupt requests from each peripheral I/O are set by the 32171's internal interrupt controller. For details, refer to Chapter 5, "Interrupt Controller."

## 4.12 Example of EIT Processing

### (1) When RIE, AE, SBI, EI, or TRAP occurs singly

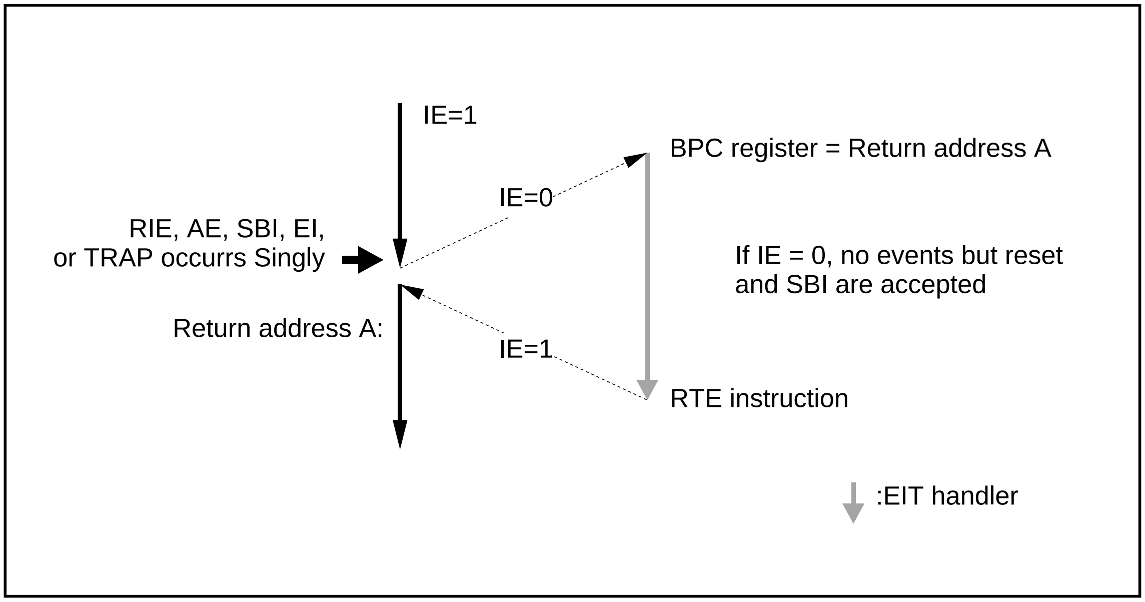


**Figure 4.12.1 Processing of Events When RIE, AE, SBI, EI, or TRAP Occurs Singly**

### (2) When RIE, AE, or TRAP and EI occurs simultaneously

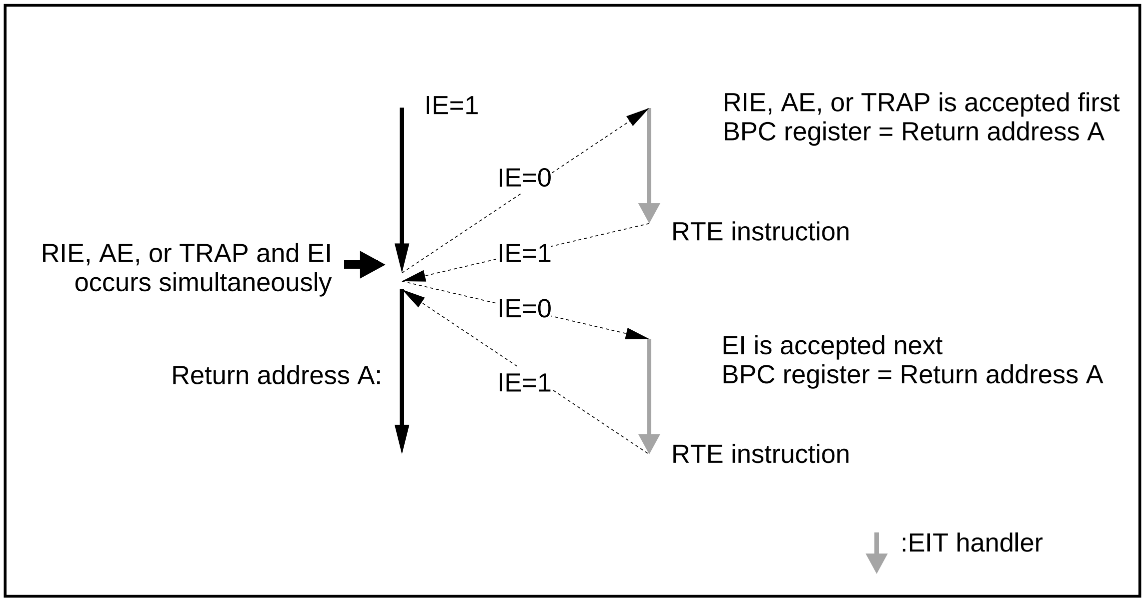


**Figure 4.12.2 Processing of Events when RIE, AE, or TRAP and EI Occurs Simultaneously**

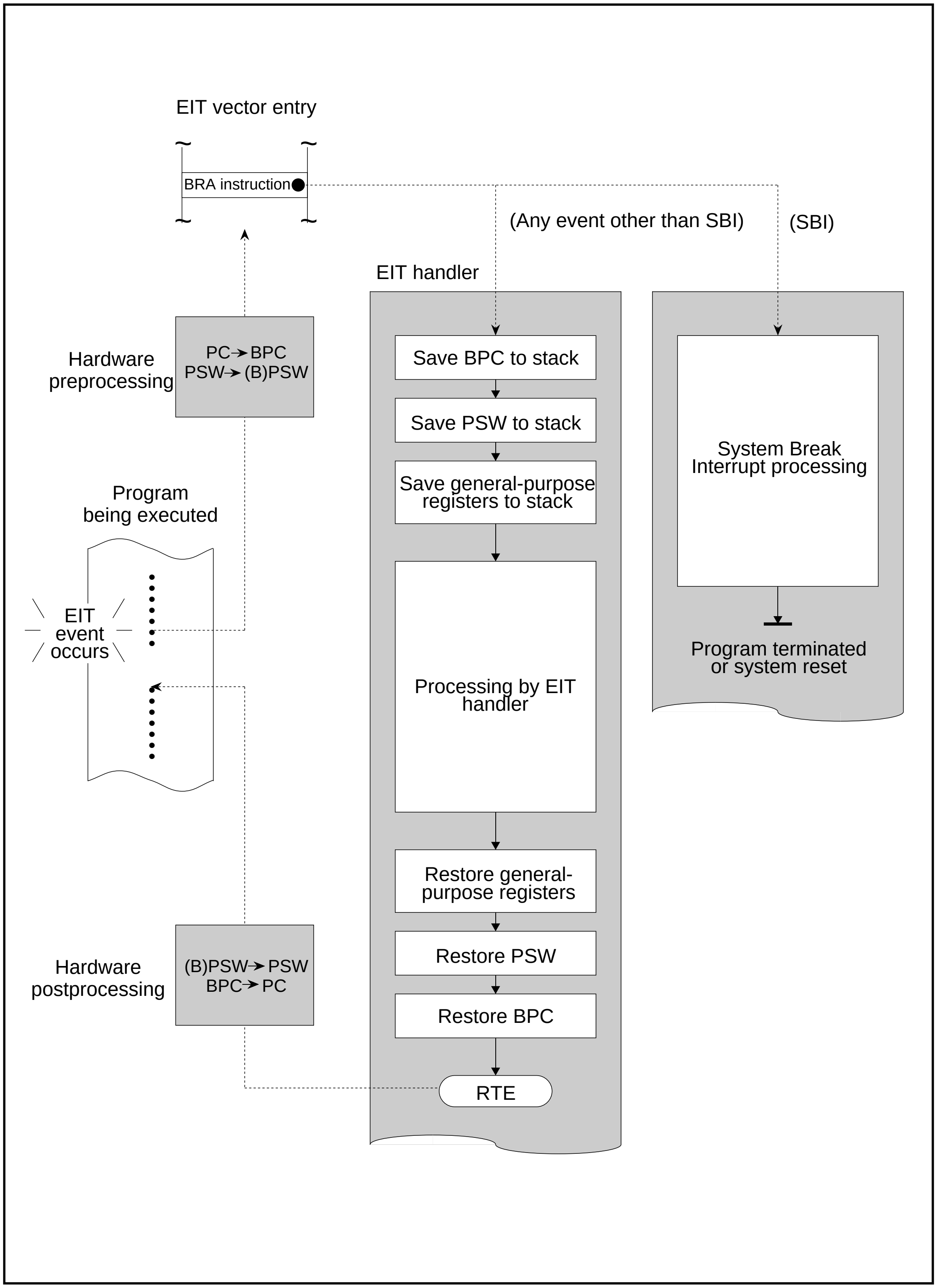


**Figure 4.12.3 Example of EIT Processing**

## 4.13 Precautions on EIT

Address Exception requires caution because when an address exception occurs pursuant to execution of an instruction (one of the following three) that uses the “register indirect + register update” addressing mode, the value of the automatically updated register (Rsrc or Rsrc2) becomes indeterminate.

Except that the values of Rsrc and Rsrc2 are indeterminate, the behavior is the same as when using other addressing modes.

- **Applicable instructions**

```
LD      Rdest,  @Rsrc+
ST      Rsrc1,  @-Rsrc2
ST      Rsrc1,  @+Rsrc2
```

If the above applies, because the register value becomes indeterminate as explained, consideration must be taken before continuing with system processing. (If an address exception occurs, it means that some fatal fault already occurred in the system at that point in time. Therefore, use EIT on condition that after processing by the address exception handler, the CPU will not return to the program it was executing when the exception occurred.)

* This is a blank page. *

# CHAPTER 5

# INTERRUPT CONTROLLER (ICU)

5.1 Outline of the Interrupt Controller (ICU)
5.2 ICU Related Registers
5.3 Interrupt Request Sources in Internal Peripheral I/O
5.4 ICU Vector Table
5.5 Description of Interrupt Operation
5.6 Description of System Break Interrupt (SBI) Operation

## 5.1 Outline of the Interrupt Controller (ICU)

The Interrupt Controller (ICU) manages maskable interrupts from internal peripheral I/Os and a system break interrupt (SBI). The maskable interrupts from internal peripheral I/Os are notified to the M32R CPU as external interrupts (EI).

There are a total of 22 interrupt sources for the maskable interrupts from internal peripheral I/Os, which are managed by assigning them one of eight priority levels including an interrupt-disabled state. When multiple interrupt requests of the same priority level occur simultaneously, their priorities are resolved by predetermined hardware priority. The source of an interrupt request generated in internal peripheral I/Os is identified by reading the relevant interrupt status register provided for internal peripheral I/Os.

On the other hand, the system break interrupt (SBI) is an interrupt request generated by a falling edge on the $\overline{\text{SBI}}$ signal input pin. This interrupt is used for emergency purposes such as when power outage is detected or a fault condition is notified by an external watchdog timer, so that it is always accepted irrespective of the PSW register IE bit status. When the ICU has finished servicing an SBI, terminate or reset the system without returning to the program that was being executed when the interrupt occurred.

Specifications of the interrupt controller are outlined in the table below.

**Table 5.1.1 Outline of Interrupt Controller (ICU)**

| Item | Specification |
|---|---|
| Interrupt source | Maskable interrupt from internal peripheral I/O : 22 sources (Note 1)<br>System break interrupt : 1 source (entered from $\overline{\text{SBI}}$ pin) |
| Level management | Eight levels including an interrupt-disabled state<br>(However, interrupts of the same level have their priorities resolved by fixed hardware priority.) |

Note 1: This is the number of interrupt requests divided into groups. There are actually a total of 70 interrupt request sources when counted individually.

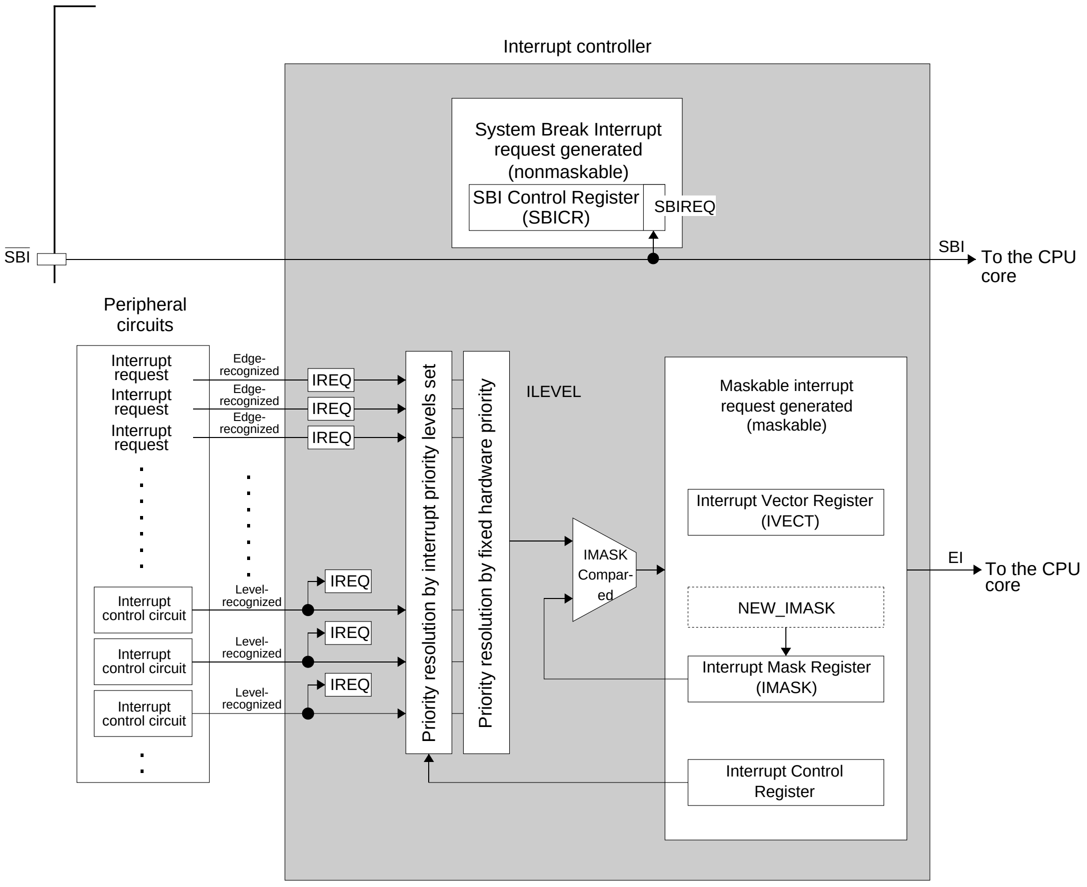

Interrupt controller
System Break Interrupt request generated (nonmaskable)
SBI Control Register (SBICR)
SBIREQ
SBI
SBI
To the CPU core
Peripheral circuits
Interrupt request
Interrupt request
Interrupt request
Edge-recognized
Edge-recognized
Edge-recognized
IREQ
IREQ
IREQ
Interrupt control circuit
Interrupt control circuit
Interrupt control circuit
Level-recognized
Level-recognized
Level-recognized
IREQ
IREQ
IREQ
Priority resolution by interrupt priority levels set
Priority resolution by fixed hardware priority
ILEVEL
IMASK Compar-ed
Maskable interrupt request generated (maskable)
Interrupt Vector Register (IVECT)
NEW_IMASK
Interrupt Mask Register (IMASK)
Interrupt Control Register
EI
To the CPU core

Figure 5.1.1 Block Diagram of the Interrupt Controller

## 5.2 ICU Related Registers

The diagram below shows a register map associated with the Interrupt Controller (ICU).

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 0000 | Interrupt Vector Register (IVECT) | |
| H'0080 0002 | | |
| H'0080 0004 | Interrupt Request Mask Register (IMASK) | |
| H'0080 0006 | SBI Control Register (SBICR) | |
| | | |
| H'0080 0060 | CAN0 Transmit/Receive & Error Interrupt Control Register (ICAN0CR) | |
| H'0080 0062 | | |
| H'0080 0064 | | |
| H'0080 0066 | | RTD Interrupt Control Register (IRTDCR) |
| H'0080 0068 | SIO2,3 Transmit/Receive Interrupt Control Register (ISIO23CR) | DMA5-9 Interrupt Control Register (IDMA59CR) |
| H'0080 006A | | |
| H'0080 006C | A-D0 Conversion Interrupt Control Register (IAD0CCR) | SIO0 Transmit Interrupt Control Register (ISIO0TXCR) |
| H'0080 006E | SIO0 Receive Interrupt Control Register (ISIO0RXCR) | SIO1 Transmit Interrupt Control Register (ISIO1TXCR) |
| H'0080 0070 | SIO1 Receive Interrupt Control Register (ISIO1RXCR) | DMA0-4 Interrupt Control Register (IDMA04CR) |
| H'0080 0072 | MJT Output Interrupt Control Register 0 (IMJTOCR0) | MJT Output Interrupt Control Register 1 (IMJTOCR1) |
| H'0080 0074 | MJT Output Interrupt Control Register 2 (IMJTOCR2) | MJT Output Interrupt Control Register 3 (IMJTOCR3) |
| H'0080 0076 | MJT Output Interrupt Control Register 4 (IMJTOCR4) | MJT Output Interrupt Control Register 5 (IMJTOCR5) |
| H'0080 0078 | MJT Output Interrupt Control Register 6 (IMJTOCR6) | MJT Output Interrupt Control Register 7 (IMJTOCR7) |
| H'0080 007A | | MJT Input Interrupt Control Register 1 (IMJTICR1) |
| H'0080 007C | MJT Input Interrupt Control Register 2 (IMJTICR2) | MJT Input Interrupt Control Register 3 (IMJTICR3) |
| H'0080 007E | MJT Input Interrupt Control Register 4 (IMJTICR4) | |

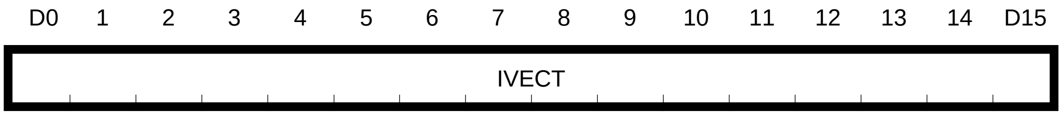
Blank addresses are reserved for future use.

Note: • The registers in the thick frames must always be accessed in halfwords.

**Figure 5.2.1 Interrupt Controller (ICU) Related Register Map**

### 5.2.1 Interrupt Vector Register

**■ Interrupt Vector Register (IVECT)** **<Address:H'0080 0000>**



<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 – 15 | IVECT (16 low-order bits of ICU vector table address) | When an interrupt request is accepted, the 16 low-order bits in ICU vector table address for the accepted interrupt source is stored in this register. | ○ | – |

Note: • This register must always be accessed in halfwords. (This is a read-only register).

The Interrupt Vector Register (IVECT) is used when an interrupt is accepted to store the 16 low-order bits of ICU vector table address for the accepted interrupt source.

Before this function can work, the ICU vector table (addresses H'0000 0094 through H'0000 010F) must have set in it the start addresses of interrupt handlers for each internal peripheral I/O. When an interrupt request is accepted, the 16 low-order bits of ICU vector table address for the accepted interrupt request source is stored in this IVECT register. In the EIT handler, read the content of this IVECT register using "LDH" instruction to get the ICU vector table address.

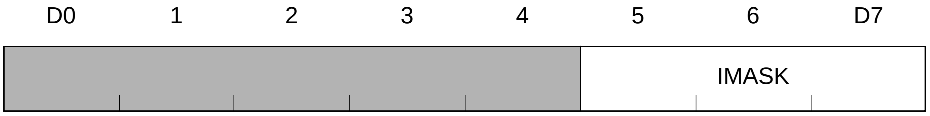
When the IVECT register is read, operations (1) to (4) below are automatically performed in hardware:

(1) The interrupt priority level (ILEVEL) of the accepted interrupt request source is set in the IMASK register as a new IMASK value. (Interrupts with lower priority levels than that of the accepted interrupt request source are masked.).
(2) The interrupt request bit for the accepted interrupt request source is cleared (not cleared for level-recognized interrupt request sources).
(3) The interrupt request (EI) to the CPU core is deasserted.
(4) The ICU's internal sequencer is activated to start internal processing (interrupt priority resolution).

Notes: • Do not read the Interrupt Vector Register (IVECT) in the EIT handler unless interrupts are disabled (PSW register IE bit = "0") . In the EIT handler, furthermore, read the Interrupt Request Mask Register (IMASK) first before reading the IVECT register.
• To reenable interrupts (by setting the IE bit to "1") after reading the Interrupt Vector Register (IVECT), perform a dummy access to the internal memory, etc. before reenabling interrupts, (The ICU vector table readout in the EI handler processing example in Figure 5.5.2 Typical Handler Operation for Interrupts from Internal Peripheral I/O is an access to the internal ROM and, therefore, does not require adding a dummy access).

### 5.2.2 Interrupt Request Mask Register

**■ Interrupt Request Mask Register (IMASK)** **<Address:H'0080 0004>**



<When reset: H"07>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 – 4 | No functions assigned | | 0 | – |
| 5– 7 | IMASK (Interrupt request mask bit) | 000 : Maskable interrupts are disabled<br>001 : Level 0 interrupts can be accepted<br>010 : Level 0-1 interrupts can be accepted<br>011 : Level 0-2 interrupts can be accepted<br>100 : Level 0-3 interrupts can be accepted<br>101 : Level 0-4 interrupts can be accepted<br>110 : Level 0-5 interrupts can be accepted<br>111 : Level 0-6 interrupts can be accepted | ○ | ○ |

The Interrupt Request Mask Register (IMASK) is used to finally determine whether or not to accept an interrupt request after comparing its priority levels (Interrupt Control Register ILEVEL bits) that have been set for each interrupt source.
When the Interrupt Vector Register (IVECT) described above is read, the interrupt priority level of the accepted interrupt request source is set in this IMASK register as a new mask value.

When any value is written to the IMASK register, operations (1) to (2) below are automatically performed in hardware:

(1) The interrupt request (EI) to the CPU core is deasserted.
(2) The ICU's internal sequencer is activated to start internal processing (interrupt priority resolution).

Notes: • Do not write to the Interrupt Request Mask Register (IMASK) in the EIT handler unless interrupts are disabled (PSW register IE bit = "0").
• To reenable interrupts (by setting the IE bit to "1") after writing to the Interrupt Request Mask Register (IMASK), perform a dummy access to the internal memory, etc. before reenabling interrupts.

### 5.2.3 SBI (System Break Interrupt) Control Register

■ **SBI (System Break Interrupt) Control Register** **<Address:H'0080 0006>**

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | SBIREQ |

<When reset: H"00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 – 6 | No functions assigned | | 0 | – |
| 7 | SBI REQ (SBI request bit) | 0 : SBI is not requested<br>1 : SBI is requested | ○ | △ (Note 1) |

Note 1: This bit can only be cleared (see below).

The System Break Interrupt (SBI) is an interrupt request generated by a falling edge on the $\overline{\text{SBI}}$ signal input pin. When a falling edge on the $\overline{\text{SBI}}$ signal input pin is detected and this bit is set to "1", a system break interrupt (SBI) request is generated to the CPU.
This bit cannot be set to "1" in software, it can only be cleared.
To clear this bit to "0", follow the procedure described below.

1. Write "1" to the SBI request bit.
2. Write "0" to the SBI request bit.

Note: • Unless this bit is set to "1", do not perform the above clearing operation.

### 5.2.4 Interrupt Control Registers

| | |
|---|---|
| ■ CAN0 Transmit/Receive & Error Interrupt Control Register (ICAN0CR) | <Address:H'0080 0060> |
| ■ RTD Interrupt Control Register (IRTDCR) | <Address:H'0080 0067> |
| ■ SIO2,3 Transmit/Receive Interrupt Control Register (ISIO23CR) | <Address:H'0080 0068> |
| ■ DMA5-9 Interrupt Control Register (IDMA59CR) | <Address:H'0080 0069> |
| ■ A-D0 Converter Interrupt Control Register (IAD0CCR) | <Address:H'0080 006C> |
| ■ SIO0 Transmit Interrupt Control Register (ISIO0TXCR) | <Address:H'0080 006D> |
| ■ SIO0 Receive Interrupt Control Register (ISIO0RXCR) | <Address:H'0080 006E> |
| ■ SIO1 Transmit Interrupt Control Register (ISIO1TXCR) | <Address:H'0080 006F> |
| ■ SIO1 Receive Interrupt Control Register (ISIO1RXCR) | <Address:H'0080 0070> |
| ■ DMA0-4 Interrupt Control Register (IDMA04CR) | <Address:H'0080 0071> |
| ■ MJT Output Interrupt Control Register 0 (IMJTOCR0) | <Address:H'0080 0072> |
| ■ MJT Output Interrupt Control Register 1 (IMJTOCR1) | <Address:H'0080 0073> |
| ■ MJT Output Interrupt Control Register 2 (IMJTOCR2) | <Address:H'0080 0074> |
| ■ MJT Output Interrupt Control Register 3 (IMJTOCR3) | <Address:H'0080 0075> |
| ■ MJT Output Interrupt Control Register 4 (IMJTOCR4) | <Address:H'0080 0076> |
| ■ MJT Output Interrupt Control Register 5 (IMJTOCR5) | <Address:H'0080 0077> |
| ■ MJT Output Interrupt Control Register 6 (IMJTOCR6) | <Address:H'0080 0078> |
| ■ MJT Output Interrupt Control Register 7 (IMJTOCR7) | <Address:H'0080 0079> |
| ■ MJT Input Interrupt Control Register 1 (IMJTICR1) | <Address:H'0080 007B> |
| ■ MJT Input Interrupt Control Register 2 (IMJTICR2) | <Address:H'0080 007C> |
| ■ MJT Input Interrupt Control Register 3 (IMJTICR3) | <Address:H'0080 007D> |
| ■ MJT Input Interrupt Control Register 4 (IMJTICR4) | <Address:H'0080 007E> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| (D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15) |
| ▨ | ▨ | ▨ | IREQ | ▨ |  | ILEVEL |  |

<When reset: H"07>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 – 2<br>(8-10) | No functions assigned |  | 0 | – |
| 3<br>(11) | IREQ<br>Interrupt request bit | <When edge recognized><br>At read<br>0: Interrupt not requested<br>1: Interrupt requested<br>At write<br>0: Clear interrupt request<br>1: Generate interrupt request | R | W |
|  |  | <When level-recognized><br>At read<br>0: Interrupt not requested<br>1: Interrupt requested | R | – |
| 4<br>(12) | No functions assigned |  | 0 | – |
| 5-7<br>(13-15) | ILEVEL<br>Interrupt priority level bits | 000 : Interrupt priority level 0<br>001 : Interrupt priority level 1<br>010 : Interrupt priority level 2<br>011 : Interrupt priority level 3<br>100 : Interrupt priority level 4<br>101 : Interrupt priority level 5<br>110 : Interrupt priority level 6<br>111 : Interrupt priority level 7 (Interrupt disabled) | R | W |

### (1) IREQ (Interrupt Request) bit (D3 or D11)

When an interrupt request from some internal peripheral I/O occurs, the corresponding IREQ (Interrupt Request) bit is set to "1".
This bit can be set and cleared in software for only edge-recognized interrupt request sources (and not for level-recognized interrupt request sources). Also, when this bit is set by an edge-recognized interrupt request generated, it is automatically cleared to "0" by reading the Interrupt Vector Register (IVECT) (not cleared in the case of level-recognized interrupt request).

If the IREQ bit is cleared in software at the same time it is set by an interrupt request generated, clearing in software has priority. Also, if the IREQ bit is cleared by reading the Interrupt Vector Register (IVECT) at the same time it is set by an interrupt request generated, clearing by a read of the IVECT register has priority.

Note: • Exernal Inerrupt (EI) to the CPU core is not deasserted by clearing the IREQ bit. External Interrupt (EI) to the CPU core can only be deasserted by the following operation:
(1) Reset
(2) IVECT register read
(3) Write to the IMASK regiser

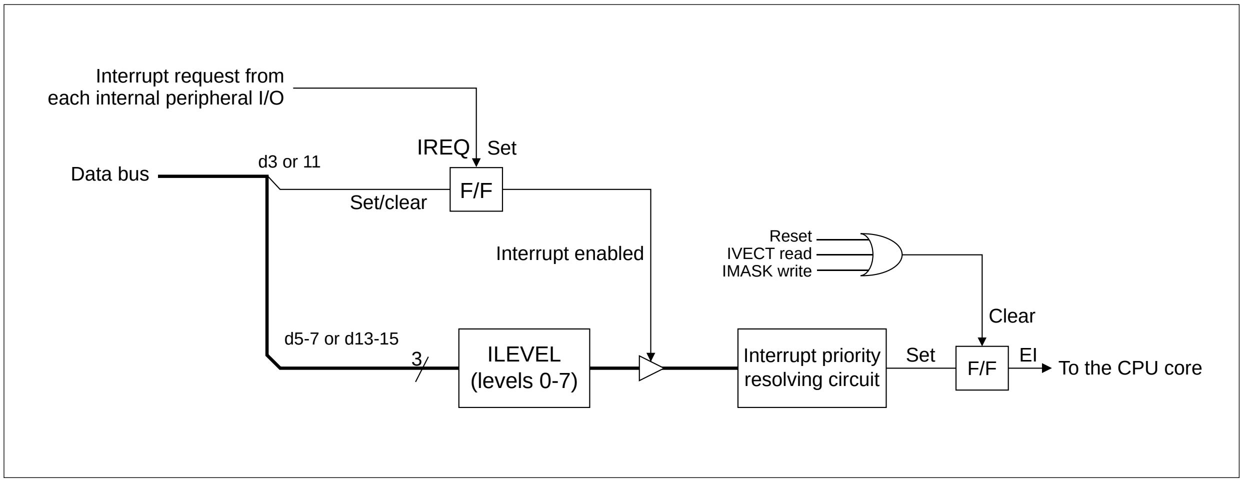


**Figure 5.2.2 Configuration of the Interrupt Control Register (Edge-recognized Type)**

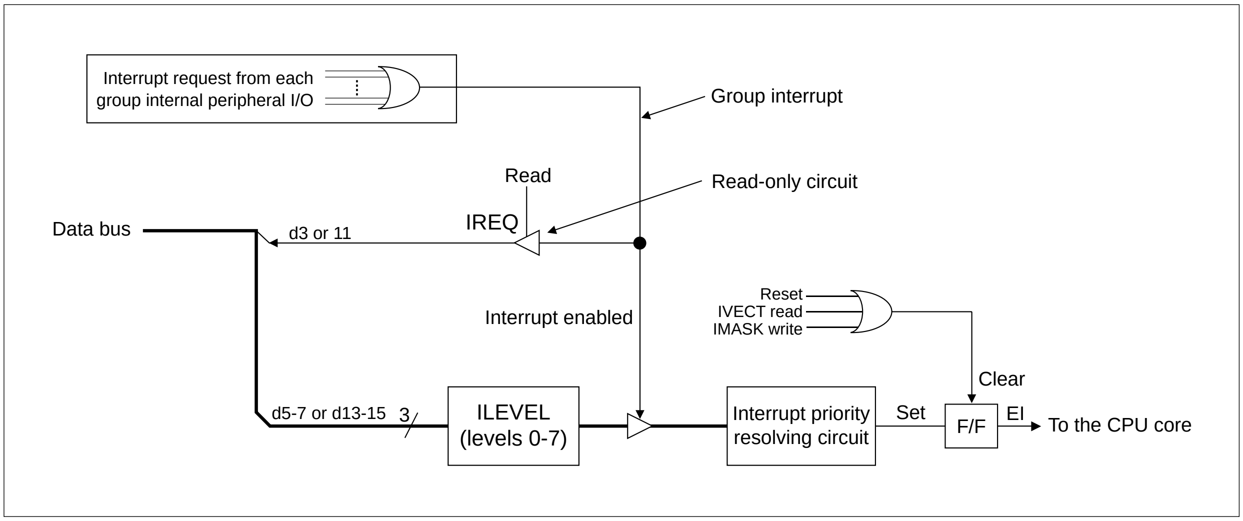


**Figure 5.2.3 Configuration of the Interrupt Control Register (Level-recognized Type)**

### (2) ILEVEL (Interrupt Priority Level) (D5-D7 or D13-D15)

These bits set the priority levels of interrupt requests from each internal peripheral I/O. Set priority level 7 to disable interrupts from some internal peripheral I/O or priority levels 0-6 to enable interrupts.
When an interrupt occurs, the interrupt controller resolves priority between this interrupt and other interrupt sources based on ILEVEL settings and finally compares its priority with the IMASK value to determine whether to forward an EI request to the CPU or keep it pending.

The table below shows the relationship between ILEVEL settings and the IMASK values at which interrupts are accepted.

**Table 5.2.1 ILEVEL Settings and Accepted IMASK Values**

| ILEVEL values set | IMASK values at which interrupts are accepted |
|---|---|
| 0 (ILEVEL="000") | Accepted when IMASK is 1-7 |
| 1 (ILEVEL="001") | Accepted when IMASK is 2-7 |
| 2 (ILEVEL="010") | Accepted when IMASK is 3-7 |
| 3 (ILEVEL="011") | Accepted when IMASK is 4-7 |
| 4 (ILEVEL="100") | Accepted when IMASK is 5-7 |
| 5 (ILEVEL="101") | Accepted when IMASK is 6-7 |
| 6 (ILEVEL="110") | Accepted when IMASK is 7 |
| 7 (ILEVEL="111") | Not accepted (interrupts disabled) |

## 5.3 Interrupt Request Sources in Internal Peripheral I/O

The interrupt controller receives as its inputs the interrupt requests from MJT (multijunction timer), DMAC, serial I/O, A-D converter, RTD, and CAN. For details about these interrupts, refer to each section in which the relevant internal peripheral I/O is described.

**Table 5.3.1 Interrupt Request Sources in Internal Peripheral I/O**

| Interrupt Request Sources | Contents | Number of Input Sources | ICU Type of Input Source(Note 1) |
|---|---|---|---|
| A-D0 conversion interrupt | Single-shot conversion in A-D0 converter scan mode completed, single mode completed, or comparator mode completed | 1 | Edge-recognized |
| SIO0 transmit interrupt | SIO0 transmit buffer empty interrupt | 1 | Edge-recognized |
| SIO0 receive interrupt | SIO0 reception completed or receive error interrupt | 1 | Edge-recognized |
| SIO1transmit interrupt | SIO1 transmit buffer empty interrupt | 1 | Edge-recognized |
| SIO1 receive interrupt | SIO1 reception completed or receive error interrupt | 1 | Edge-recognized |
| SIO2,3 transmit/receive interrupt | SIO2 reception completed or receive error interrupt, transmit buffer empty interrupt | 2 | Level-recognized |
| RTD interrupt | RTD interrupt generation command | 1 | Edge-recognized |
| DMA transfer interrupt 0 | DMA0-4 transfer completed | 5 | Level-recognized |
| DMA transfer interrupt 1 | DMA5-9 transfer completed | 5 | Level-recognized |
| CAN0 transmit/receive & error interrupt | CAN0 transmission completed, CAN0 reception completed, CAN0 error passive, CAN0 error bus-off, CAN0 bus error | 19 | Level-recognized |
| MJT output interrupt 7 | MJT output interrupt group 7 (TMS0, TMS1 output) | 2 | Level-recognized |
| MJT output interrupt 6 | MJT output interrupt group 6 (TOP8, TOP9 output) | 2 | Level-recognized |
| MJT output interrupt 5 | MJT output interrupt group 5 (TOP10 output) | 1 | Edge-recognized |
| MJT output interrupt 4 | MJT output interrupt group 4 (TIO4 - TIO7 output) | 4 | Level-recognized |
| MJT output interrupt 3 | MJT output interrupt group 3 (TIO8, TIO9 output) | 2 | Level-recognized |
| MJT output interrupt 2 | MJT output interrupt group 2 (TOP0 - TOP5 output) | 6 | Level-recognized |
| MJT output interrupt 1 | MJT output interrupt group 1 (TOP6, TOP7 output) | 2 | Level-recognized |
| MJT output interrupt 0 | MJT output interrupt group 0 (TIO0 - TIO3 output) | 4 | Level-recognized |
| MJT input interrupt 4 | MJT input interrupt group 4 (TIN3 input) | 1 | Level-recognized |
| MJT input interrupt 3 | MJT input interrupt group 3 (TIN20-TIN23 input) | 4 | Level-recognized |
| MJT input interrupt 2 | MJT input interrupt group 2 (TIN16-TIN19 input) | 4 | Level-recognized |
| MJT input interrupt 1 | MJT input interrupt group 1 (TIN0 input) | 1 | Level-recognized |

Note 1: ICU type of input source

- Edge-recognized: Interrupt requests are generated on a falling edge of the interrupt signal applied to the ICU.
- Level-recognized: Interrupt requests are generated when the interrupt signal applied to the ICU is held low. For these level-recognized interrupts, the ICU's Interrupt Control register IRQ bit cannot be set or cleared in software.

## 5.4 ICU Vector Table

The ICU vector table is used to set the start addresses of interrupt handlers for each internal peripheral I/O. The 22-source interrupts are assigned the following addresses:

**Table 5.4.1 ICU Vector Table Addresses**

| Interrupt Source | ICU Vector Table Address |
|---|---|
| MJT Input Interrupt request 4 (TIN3 input) | H'0000 0094-H'0000 0097 |
| MJT Input Interrupt request 3 (TIN20-TIN23 input) | H'0000 0098-H'0000 009B |
| MJT Input Interrupt request 2 (TIN16-TIN19 input) | H'0000 009C-H'0000 009F |
| MJT Input Interrupt request 1 (TIN0 input) | H'0000 00A0-H'0000 00A3 |
| MJT Output Interrupt request 7 (TMS0, TMS1 output) | H'0000 00A8-H'0000 00AB |
| MJT Output Interrupt request 6 (TOP8, TOP9 output) | H'0000 00AC-H'0000 00AF |
| MJT Output Interrupt request 5 (TOP10 output) | H'0000 00B0-H'0000 00B3 |
| MJT Output Interrupt request 4 (TIO4 - TIO7 output) | H'0000 00B4-H'0000 00B7 |
| MJT Output Interrupt request 3 (TIO8, TIO9 output) | H'0000 00B8-H'0000 00BB |
| MJT Output Interrupt request 2 (TOP0 - TOP5 output) | H'0000 00BC-H'0000 00BF |
| MJT Output Interrupt request 1 (TOP6, TOP7 output) | H'0000 00C0-H'0000 00C3 |
| MJT Output Interrupt request 0 (TIO0 - TIO3 output) | H'0000 00C4-H'0000 00C7 |
| DMA0-4 Interrupt request | H'0000 00C8-H'0000 00CB |
| SIO1 Receive Interrupt request | H'0000 00CC-H'0000 00CF |
| SIO1 Transmit Interrupt request | H'0000 00D0-H'0000 00D3 |
| SIO0 Receive Interrupt request | H'0000 00D4-H'0000 00D7 |
| SIO0 Transmit Interrupt request | H'0000 00D8-H'0000 00DB |
| A-D0 Converter Interrupt request | H'0000 00DC-H'0000 00DF |
| DMA5-9 Interrupt request | H'0000 00E8-H'0000 00EB |
| SIO2,3 Transmit/Receive Interrupt request | H'0000 00EC-H'0000 00EF |
| RTD Interrupt request | H'0000 00F0-H'0000 00F3 |
| CAN0 Transmit/Receive & Error Interrupt request | H'0000 010C-H'0000 010F |

| Address | D0 +0 Address D7 D8 +1 Address D15 |
|---|---|
| H'0000 0094 | MJT Input Interrupt 4 Handler Start Address (A0-A15) |
| H'0000 0096 | MJT Input Interrupt 4 Handler Start Address (A16-A31) |
| H'0000 0098 | MJT Input Interrupt 3 Handler Start Address (A0-A15) |
| H'0000 009A | MJT Input Interrupt 3 Handler Start Address (A16-A31) |
| H'0000 009C | MJT Input Interrupt 2 Handler Start Address (A0-A15) |
| H'0000 009E | MJT Input Interrupt 2 Handler Start Address (A16-A31) |
| H'0000 00A0 | MJT Input Interrupt 1 Handler Start Address (A0-A15) |
| H'0000 00A2 | MJT Input Interrupt 1 Handler Start Address (A16-A31) |
| H'0000 00A4 | |
| H'0000 00A6 | |
| H'0000 00A8 | MJT Output Interrupt 7 Handler Start Address (A0-A15) |
| H'0000 00AA | MJT Output Interrupt 7 Handler Start Address (A16-A31) |
| H'0000 00AC | MJT Output Interrupt 6 Handler Start Address (A0-A15) |
| H'0000 00AE | MJT Output Interrupt 6 Handler Start Address (A16-A31) |
| H'0000 00B0 | MJT Output Interrupt 5 Handler Start Address (A0-A15) |
| H'0000 00B2 | MJT Output Interrupt 5 Handler Start Address (A16-A31) |
| H'0000 00B4 | MJT Output Interrupt 4 Handler Start Address (A0-A15) |
| H'0000 00B6 | MJT Output Interrupt 4 Handler Start Address (A16-A31) |
| H'0000 00B8 | MJT Output Interrupt 3 Handler Start Address (A0-A15) |
| H'0000 00BA | MJT Output Interrupt 3 Handler Start Address (A16-A31) |
| H'0000 00BC | MJT Output Interrupt 2 Handler Start Address (A0-A15) |
| H'0000 00BE | MJT Output Interrupt 2 Handler Start Address (A16-A31) |
| H'0000 00C0 | MJT Output Interrupt 1 Handler Start Address (A0-A15) |
| H'0000 00C2 | MJT Output Interrupt 1 Handler Start Address (A16-A31) |
| H'0000 00C4 | MJT Output Interrupt 0 Handler Start Address (A0-A15) |
| H'0000 00C6 | MJT Output Interrupt 0 Handler Start Address (A16-A31) |

Blank addresses are reserved for future use.

**Figure 5.4.1 ICU Vector Table Memory Map (1/2)**

| Address | D0 +0 Address D7 D8 +1 Address D15 |
|---|---|
| H'0000 00C8 | DMA0-4 Interrupt Handler Start Address (A0-A15) |
| H'0000 00CA | DMA0-4 Interrupt Handler Start Address (A16-A31) |
| H'0000 00CC | SIO1 Receive Interrupt Handler Start Address (A0-A15) |
| H'0000 00CE | SIO1 Receive Interrupt Handler Start Address (A16-A31) |
| H'0000 00D0 | SIO1 Transmit Interrupt Handler Start Address (A0-A15) |
| H'0000 00D2 | SIO1 Transmit Interrupt Handler Start Address (A16-A31) |
| H'0000 00D4 | SIO0 Receive Interrupt Handler Start Address (A0-A15) |
| H'0000 00D6 | SIO0 Receive Interrupt Handler Start Address (A16-A31) |
| H'0000 00D8 | SIO0 Transmit Interrupt Handler Start Address (A0-A15) |
| H'0000 00DA | SIO0 Transmit Interrupt Handler Start Address (A16-A31) |
| H'0000 00DC | A-D0 Converter Interrupt Handler Start Address (A0-A15) |
| H'0000 00DE | A-D0 Converter Interrupt Handler Start Address (A16-A31) |
| H'0000 00E0 | |
| H'0000 00E2 | |
| H'0000 00E4 | |
| H'0000 00E6 | |
| H'0000 00E8 | DMA5-9 Interrupt Handler Start Address (A0-A15) |
| H'0000 00EA | DMA5-9 Interrupt Handler Start Address (A16-A31) |
| H'0000 00EC | SIO2 Transmit/Receive Interrupt Handler Start Address (A0-A15) |
| H'0000 00EE | SIO2 Transmit/Receive Interrupt Handler Start Address (A16-A31) |
| H'0000 00F0 | RTD Interrupt Handler Start Address (A0-A15) |
| H'0000 00F2 | RTD Interrupt Handler Start Address (A16-A31) |
| H'0000 00F4 | |
| H'0000 00F6 | |
| H'0000 00F8 | |
| H'0000 00FA | |
| H'0000 00FC | |
| H'0000 00FE | |
| H'0000 0100 | |
| H'0000 0102 | |
| H'0000 0104 | |
| H'0000 0106 | |
| H'0000 0108 | |
| H'0000 010A | |
| H'0000 010C | CAN0 Transmit/Receive & Error Interrupt Handler Start Address (A0-A15) |
| H'0000 010E | CAN0 Transmit/Receive & Error Interrupt Handler Start Address (A16-A31) |

Blank addresses are reserved for future use.

**Figure 5.4.2 ICU Vector Table Memory Map (2/2)**

# 5.5 Description of Interrupt Operation

## 5.5.1 Acceptance of Internal Peripheral I/O Interrupts

An interrupt from any internal peripheral I/O is checked to see whether or not to accept by comparing its ILEVEL value set in the Interrupt Control Register and the IMASK value of the Interrupt Request Mask Register. If its priority is higher than the IMASK value, the interrupt request is accepted. However, when multiple interrupt requests occur simultaneously, the interrupt controller resolves priority between these interrupt requests following the procedure described below.

(a) The ILEVEL values set in the Interrupt Control Register for each interrupt peripheral I/Os are compared with each other.
(b) If the ILEVEL values are the same, they are resolved according to the predetermined hardware priority.
(c) The ILEVEL value is compared with IMASK value.

When multiple interrupt requests occur simultaneously, the interrupt controller first compares their priority levels set in each Interrupt Control Register's ILEVEL bit to select an interrupt request which has the highest priority. If the interrupt requests have the same LEVEL value, they are resolved according to the hardware-fixed priority.
The interrupt request thus selected has its ILEVEL value compared with IMASK value and if its priority is higher than the IMASK value, the interrupt controller sends an EI request to the CPU.

Interrupt requests may be masked by setting the Interrupt Mask Register and the Interrupt Control Register's ILEVEL bit (level 7 = disabled) provided for each internal peripheral I/O and the PSW register IE bit.

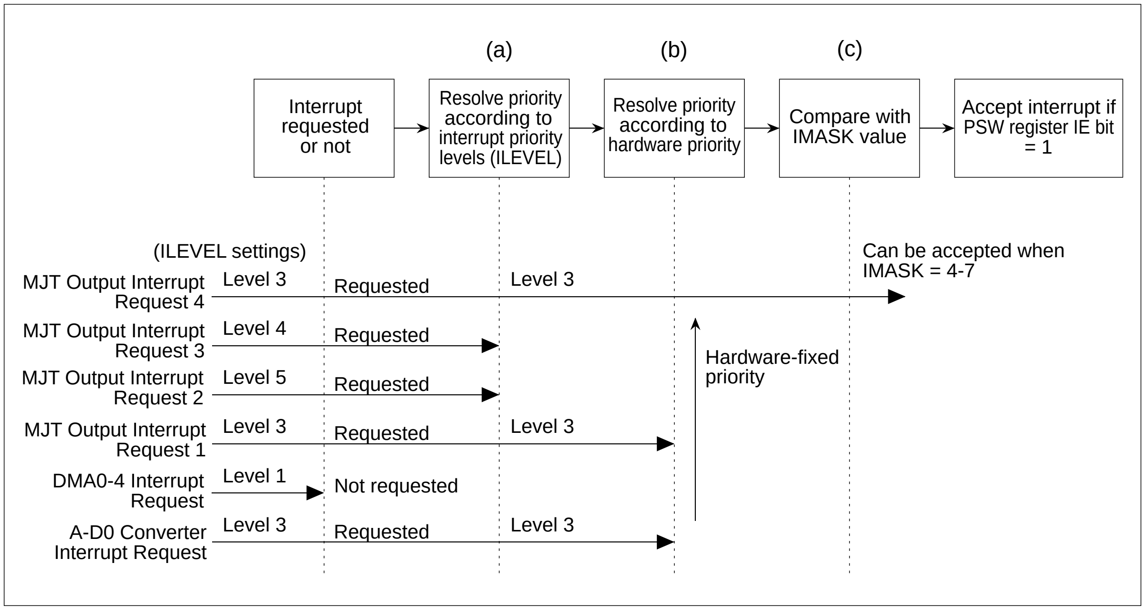

(a)
(b)
(c)
Interrupt requested or not
Resolve priority according to interrupt priority levels (ILEVEL)
Resolve priority according to hardware priority
Compare with IMASK value
Accept interrupt if PSW register IE bit = 1
(ILEVEL settings)
MJT Output Interrupt Request 4
Level 3
Requested
Level 3
Can be accepted when IMASK = 4-7
MJT Output Interrupt Request 3
Level 4
Requested
MJT Output Interrupt Request 2
Level 5
Requested
Hardware-fixed priority
MJT Output Interrupt Request 1
Level 3
Requested
Level 3
DMA0-4 Interrupt Request
Level 1
Not requested
A-D0 Converter Interrupt Request
Level 3
Requested
Level 3

**Figure 5.5.1 Example of Priority Resolution When Accepting Interrupt Requests**

Table 5.5.1 Hardware-fixed Priority Levels

| Priority | Interrupt Request Source | ICU Vector Table Address | Type of Input Source |
|---|---|---|---|
| High | MJT Input Interrupt Request 4 (TIN3 input) | H'0000 0094-H'0000 0097 | Level-recognized |
| | MJT Input Interrupt Request 3 (TIN20-TIN23 input) | H'0000 0098-H'0000 009B | Level-recognized |
| | MJT Input Interrupt Request 2 (TIN16-TIN19 input) | H'0000 009C-H'0000 009F | Level-recognized |
| | MJT Input Interrupt Request 1 (TIN0 input) | H'0000 00A0-H'0000 00A3 | Level-recognized |
| | MJT Output Interrupt Request 7 (TMS0,TMS1 output) | H'0000 00A8-H'0000 00AB | Level-recognized |
| | MJT Output Interrupt Request 6 (TOP8,TOP9 output) | H'0000 00AC-H'0000 00AF | Level-recognized |
| | MJT Output Interrupt Request 5 (TOP10 output) | H'0000 00B0-H'0000 00B3 | Edge-recognized |
| | MJT Output Interrupt Request 4 (TIO4-TIO7 output) | H'0000 00B4-H'0000 00B7 | Level-recognized |
| | MJT Output Interrupt Request 3 (TIO8, TIO9 output) | H'0000 00B8-H'0000 00BB | Level-recognized |
| | MJT Output Interrupt Request 2 (TOP0-TOP5 output) | H'0000 00BC-H'0000 00BF | Level-recognized |
| | MJT Output Interrupt Request 1 (TOP6, TOP7 output) | H'0000 00C0-H'0000 00C3 | Level-recognized |
| | MJT Output Interrupt Request 0 (TIO0-TIO3 output) | H'0000 00C4-H'0000 00C7 | Level-recognized |
| | DMA0-4 Interrupt Request | H'0000 00C8-H'0000 00CB | Level-recognized |
| | SIO1 Receive Interrupt Request | H'0000 00CC-H'0000 00CF | Edge-recognized |
| | SIO1 Transmit Interrupt Request | H'0000 00D0-H'0000 00D3 | Edge-recognized |
| | SIO0 Receive Interrupt Request | H'0000 00D4-H'0000 00D7 | Edge-recognized |
| | SIO0 Transmit Interrupt Request | H'0000 00D8-H'0000 00DB | Edge-recognized |
| | A-D0 Converter Interrupt Request | H'0000 00DC-H'0000 00DF | Edge-recognized |
| | DMA5-9 Interrupt Request | H'0000 00E8-H'0000 00EB | Level-recognized |
| | SIO2,3 Transmit/Receive Interrupt Request | H'0000 00EC-H'0000 00EF | Level-recognized |
| | RTD Interrupt Request | H'0000 00F0-H'0000 00F3 | Edge-recognized |
| Low | CAN0 Transmit/Receive & Error Interrupt Request | H'0000 010C-H'0000 010F | Level-recognized |

Table 5.5.2 ILEVEL Settings and Accepted IMASK Values

| ILEVEL values set | IMASK values at which interrupts are accepted |
|---|---|
| 0 (ILEVEL="000") | Accepted when IMASK is 1-7 |
| 1 (ILEVEL="001") | Accepted when IMASK is 2-7 |
| 2 (ILEVEL="010") | Accepted when IMASK is 3-7 |
| 3 (ILEVEL="011") | Accepted when IMASK is 4-7 |
| 4 (ILEVEL="100") | Accepted when IMASK is 5-7 |
| 5 (ILEVEL="101") | Accepted when IMASK is 6-7 |
| 6 (ILEVEL="110") | Accepted when IMASK is 7 |
| 7 (ILEVEL="111") | Not accepted (interrupts disabled) |

### 5.5.2 Processing of Internal Peripheral I/O Interrupts by Handler

**(1) Branching to the interrupt handler**

When the CPU accepts an interrupt, control branches to the EIT vector entry after hardware preprocessing as described in Section 4.3, "EIT Processing Procedure." The EIT vector entry for External Interrupt (EI) is located at address H'0000 0080. This address is where the instruction (not the jump address) for branching to the beginning of the interrupt processing routine for External Interrupt (EI) is written.

**(2) Processing in the External Interrupt (EI) handler**

A typical operation of the External Interrupt (EI) handler (for interrupts from internal peripheral I/O) is shown in Figure 5.5.2.

**[1] Saving each register to the stack**

Save the BPC, PSW and general-purpose registers to the stack. Also, save tthe accumulator as necessary.

**[2] Reading the Interrupt Request Mask Register (IMASK) and saving to the stack**

Read the Interrupt Request Mask Register and save its content to the stack.

**[3] Reading the Interrupt Vector Register (IVECT)**

Read the Interrupt Vector Register. This register holds the 16 low-order address bits of the ICU vector table for the accepted interrupt request source that was stored in it when accepting an interrupt request. When the Interrupt Vector Register is read, the following processing is automatically performed in hardware:

- The interrupt priority level of the accepted interrupt request (ILEVEL) is set in the IMASK register as a new IMASK value. (Interrupts with lower priority levels than that of the accepted interrupt request source are masked.)
- The accepted interrupt request source is cleared (not cleared for level-recognized interrupt request sources).
- The interrupt request (EI) to the CPU core is dropped.
- The ICU’s internal sequencer is activated to start internal processing (interrupt priority resolution).

**[4] Reading and overwriting the Interrupt Request Mask Register (IMASK)**

Read the Interrupt Request Mask Register and overwrite it with the read value. This write to the IMASK register causes the following processing to be automatically performed in hardware:

- The interrupt request (EI) to the CPU core is dropped.
- The ICU’s internal sequencer is activated to start internal processing (interrupt priority resolution).

Note: • Processing in [4] here is unnecessary when multiple interrupts are to be enabled in [6] below.

**[5] Reading the ICU vector table**

Read the ICU vector table for the accepted interrupt request source. The relevant ICU vector table address can be obtained by zero-extending the content of the Interrupt Vector Register that was read in [3] (i.e., the 16 low-order address bits of the ICU vector table for the accepted interrupt request source). The ICU vector table must have set in it the start address of the interrupt handler for the interrupt request source concerned.)

**[6] Enabling multiple interrupts**

To enable another higher priority interrupt while processing the accepted interrupt (i.e., enabling multiple interrupts), set the PSW register IE bit to "1".

**[7] Branching to the internal peripheral I/O interrupt handler**

Branch to the start address of the interrupt handler that was read out in [5].

**[8] Processing in the internal peripheral I/O interrupt handler**

**[9] Disabling interrupts**

Clear the PSW register IE bit to "0" to disable interrupts.

**[10] Restoring the Interrupt Request Mask Register (IMASK)**

Restore the Interrupt Request Mask Register that was saved to the stack in [2].

**[11] Restoring registers from the stack**

Restore the registers that were saved to the stack in [1].

**[12] Completion of external interrupt processing**

Execute the RTE instruction to complete the external interrupt processing. The program returns to the state in which it was before the currently processed interrupt request was accepted.

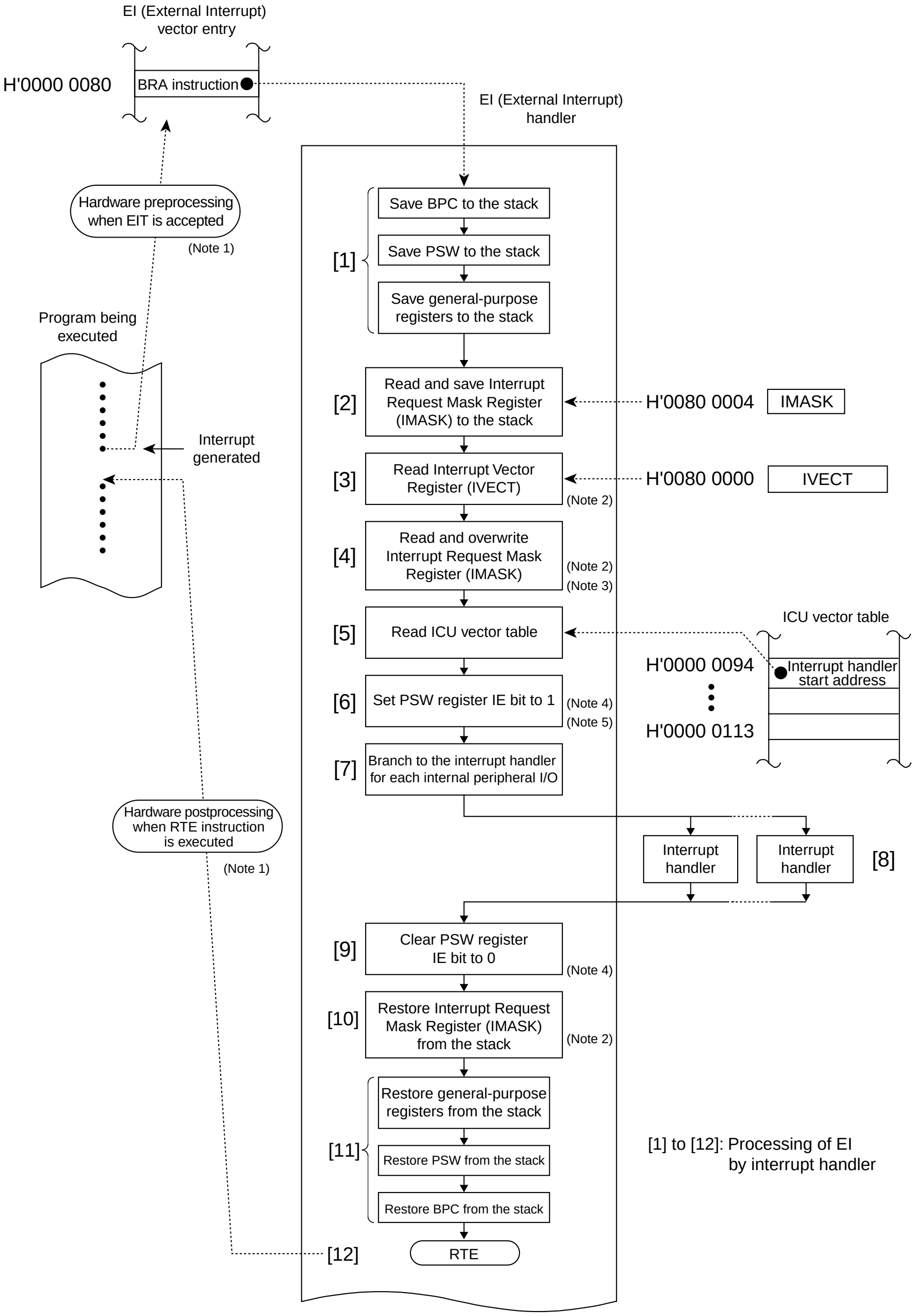
### (3) Identifying the source of the interrupt request generated

If any internal peripheral I/O has two or more interrupt request sources, check the Interrupt Request Status Register provided for each internal peripheral I/O to identify the source of the interrupt request generated.

### (4) Enabling multiple interrupts

To enable multiple interrupts in the interrupt handler, set the PSW register IE (Interrupt Enable) bit to enable interrupt requests to be accepted. However, before writing "1" to the IE bit, be sure to save each register (BPC, PSW, general-purpose registers and IMASK) to the stack.

Note: • Before enabling multiple interrupts, read the Interrupt Vector Register (IVECT) and then the ICU vector table, as shown in Figure 5.5.2, “Typical Handler Operation for Interrupts from Internal Peripheral I/O.”


EI (External Interrupt) vector entry
H'0000 0080
BRA instruction
EI (External Interrupt) handler
Hardware preprocessing when EIT is accepted
(Note 1)
Program being executed
Interrupt generated
[1]
Save BPC to the stack
Save PSW to the stack
Save general-purpose registers to the stack
[2]
Read and save Interrupt Request Mask Register (IMASK) to the stack
H'0080 0004
IMASK
[3]
Read Interrupt Vector Register (IVECT)
H'0080 0000
IVECT
(Note 2)
[4]
Read and overwrite Interrupt Request Mask Register (IMASK)
(Note 2)
(Note 3)
[5]
Read ICU vector table
ICU vector table
H'0000 0094
Interrupt handler start address
H'0000 0113
[6]
Set PSW register IE bit to 1
(Note 4)
(Note 5)
[7]
Branch to the interrupt handler for each internal peripheral I/O
Interrupt handler
Interrupt handler
[8]
Hardware postprocessing when RTE instruction is executed
(Note 1)
[9]
Clear PSW register IE bit to 0
(Note 4)
[10]
Restore Interrupt Request Mask Register (IMASK) from the stack
(Note 2)
[11]
Restore general-purpose registers from the stack
Restore PSW from the stack
Restore BPC from the stack
[12]
RTE
[1] to [12]: Processing of EI by interrupt handler

Note 1: For operations at EIT acceptance and return from EIT, also see Section 4.3, "EIT Processing Procedure."

Note 2: Do not read the Interrupt Vector Register (IVECT) or write to the Interrupt Request Mask Register (IMASK) in the EIT handler unless interrupts are disabled (PSW register IE bit = 0).

Note 3: When multiple interrupts are disabled, execute processing in [4]. Processing in [4] is unnecessary if multiple interrupts are enabled by executing processing in [6] and [9].

Note 4: To enable multiple interrupts, execute processing in [6] and [9].

Note 5: To reenable interrupts (by setting the IE bit to 1) after reading the Interrupt Vector Register (IVECT), perform a dummy access to the internal memory, etc. before reenabling interrupts. In the example here, there is no need to add a dummy access because the ICU vector table is read after reading the IVECT register. Similarly, to reenable interrupts (by setting the IE bit to 1) after writing to the Interrupt Request Mask Register (IMASK), perform a dummy access to the internal memory, etc. before reenabling interrupts.

**Figure 5.5.2 Typical Operation for Interrupts from Internal Peripheral I/O**

## 5.6 Description of System Break Interrupt (SBI) Operation

### 5.6.1 Acceptance of SBI

System Break Interrupt (SBI) is an emergency interrupt which is used when power failure is detected or a fault condition is notified by an external watchdog timer. The system break interrupt is accepted anytime upon detection of a falling edge on the $\overline{\text{SBI}}$ signal input pin regardless of how the PSW register IE bit is set, and cannot be masked.

### 5.6.2 SBI Processing by Handler

When the system break interrupt generated has been serviced, always be sure to terminate or reset the system without returning to the program that was being executed when the interrupt occurred.

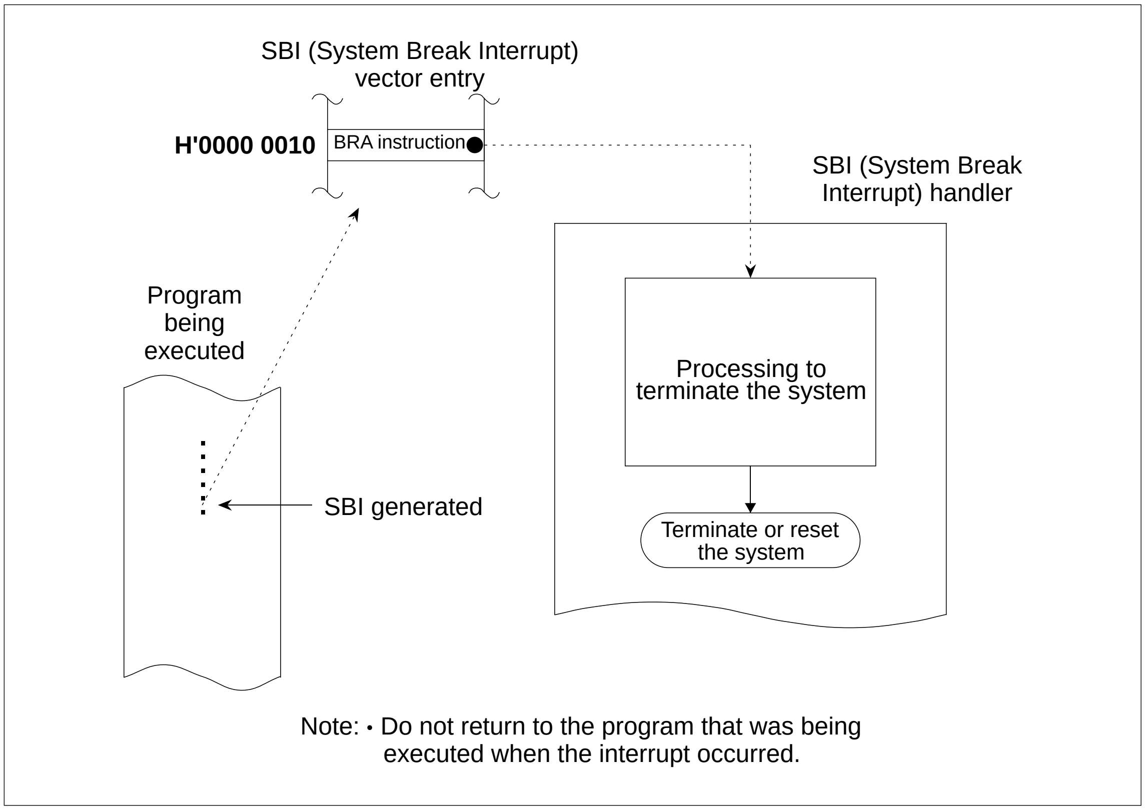


**Figure 5.6.1 Typical SBI Operation**

# CHAPTER 6
# INTERNAL MEMORY

6.1 Outline of the Internal Memory
6.2 Internal RAM
6.3 Internal Flash Memory
6.4 Registers Associated with the Internal Flash Memory
6.5 Programming of the Internal Flash Memory
6.6 Boot ROM
6.7 Virtual Flash Emulation Function
6.8 Connecting to A Serial Programmer
6.9 Internal Flash Memory Protect Functions
6.10 Precautions to Be Taken When Reprogramming Flash Memory

## 6.1 Outline of the Internal Memory

The 32171 internally contains the following types of memory:

- 16 Kbyte RAM
- 512 Kbyte, 384 Kbyte, or 256 Kbyte flash memory

## 6.2 Internal RAM

Specifications of the 32171's internal RAM are shown below.

**Table 6.2.1 Specifications of the Internal RAM**

| Item | Specification |
|---|---|
| Capacity | 16 Kbytes |
| Location address | H'0080 4000 - H'0080 7FFF |
| Wait insertion | Operates with no wait states (when using 40 MHz CPU clock) |
| Internal bus connection | Connected by 32-bit bus |
| Dual port | By using the Real-Time Debugger (RTD), data can be read (monitored) or written to any area of the internal RAM via serial communication from external devices independently of the CPU. (Refer to Chapter 14, "Real-Time Debugger.") |

Note: • At power-on reset, the internal RAM value is indeterminate. (However, if the device is reset and placed out of reset while the VDD pin has 2.0 V to 3.6 V being applied to it, the RAM content before a reset is retained.)

## 6.3 Internal Flash Memory

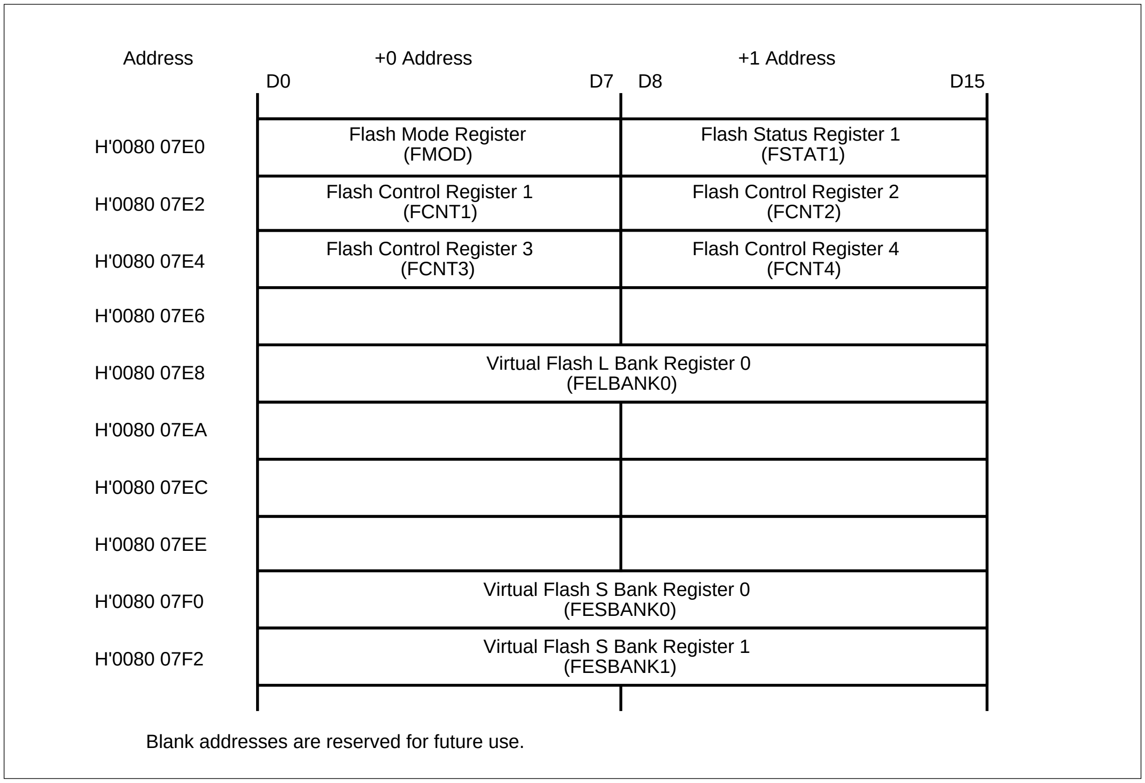
Specifications of the 32171's internal flash memory are shown below.

**Table 6.3.1 Specifications of the Internal Flash Memory**

| Item | Specification |
|---|---|
| Capacity | M32171F4 : 512 Kbytes M32171F3 : 384Kbytes M32171F2 : 256 Kbytes |
| Location address | M32171F4 : H'0000 0000 - H'0007 FFFF<br>M32171F3 : H'0000 0000 - H'0005 FFFF<br>M32171F2 : H'0000 0000 - H'0003 FFFF |
| Wait insertion | Operates with no wait states (when using 40 MHz CPU clock) |
| Durability | Can be rewritten 100 times |
| Internal bus connection | Connected by 32-bit bus |
| Other | Virtual flash emulation function is included. (Refer to Section 6.7, "Virtual Flash Emulation Function.") |

## 6.4 Registers Associated with the Internal Flash Memory

The diagram below shows a register map associated with the internal flash memory.



<table>
<tr><th>Address</th><th>+0 Address<br>D0 D7</th><th>+1 Address<br>D8 D15</th></tr>
<tr><td>H'0080 07E0</td><td>Flash Mode Register<br>(FMOD)</td><td>Flash Status Register 1<br>(FSTAT1)</td></tr>
<tr><td>H'0080 07E2</td><td>Flash Control Register 1<br>(FCNT1)</td><td>Flash Control Register 2<br>(FCNT2)</td></tr>
<tr><td>H'0080 07E4</td><td>Flash Control Register 3<br>(FCNT3)</td><td>Flash Control Register 4<br>(FCNT4)</td></tr>
<tr><td>H'0080 07E6</td><td></td><td></td></tr>
<tr><td>H'0080 07E8</td><td colspan="2">Virtual Flash L Bank Register 0<br>(FELBANK0)</td></tr>
<tr><td>H'0080 07EA</td><td></td><td></td></tr>
<tr><td>H'0080 07EC</td><td></td><td></td></tr>
<tr><td>H'0080 07EE</td><td></td><td></td></tr>
<tr><td>H'0080 07F0</td><td colspan="2">Virtual Flash S Bank Register 0<br>(FESBANK0)</td></tr>
<tr><td>H'0080 07F2</td><td colspan="2">Virtual Flash S Bank Register 1<br>(FESBANK1)</td></tr>
</table>

Blank addresses are reserved for future use.

**Figure 6.4.1 Register Map Associated with the Internal Flash Memory**

### 6.4.1 Flash Mode Register

**■ Flash Mode Register (FMOD)** <Address: H'0080 07E0>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | | | | | | FPMOD |

<When reset : H'0?>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 6 | No functions assigned | | 0 | — |
| 7 | FPMOD<br>(External FP pin status) | 0 : FP pin = low<br>1 : FP pin = high | ○ | — |

The Flash Mode Register (FMOD) is a read-only status register, with its FPMOD bit indicating the status of the FP (Flash Protect) pin. Write to the flash memory is enabled only when FPMOD = 1. Writing to the flash memory when FPMOD = 0 has no effect.

### 6.4.2 Flash Status Registers

The 32171 has two registers to indicate the flash memory status, one of which is Flash Status Register 1 (FSTAT1) located in the SFR area (address: H'0080 07E1), and the other is Flash Status Register 2 (FSTAT2) included in the flash memory itself. When programming or erasing the flash memory, use these two status registers (FSTAT1, FSTAT2) to control the program/erase operations.

**■ Flash Status Register 1 (FSTAT1)** <Address: H'0080 07E1>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | | | | | | FSTAT |

<When reset : H'01>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 14 | No functions assigned | | 0 | — |
| 15 | FSTAT<br>(Ready/Busy status) | 0 : Busy<br>1 : Ready | ○ | — |

The Flash Status Register 1 (FSTAT1) is a read-only status register used to know the execution status of whether the flash memory is being programmed or erased.

Note: • While FSTAT bit = 0 (Busy), do not manipulate Flash Control Register 4 (FCNT4)'s FRESET bit.

## ■ Flash Status Register 2 (FSTAT2)

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| FBUSY |  | ERASE | WRERR1 | WRERR2 |  |  |  |

<When reset : H'80>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | FBUSY<br>(Flash busy) | 0 : Program or erase under way<br>1 : Ready state | ○ | — |
| 9 | No functions assigned |  | 0 | — |
| 10 | ERASE<br>(Auto Erase operating condition) | 0 : Erase normally operating/terminated<br>1 : Erase error occurred | ○ | — |
| 11 | WRERR1<br>(Program operating condition) | 0 : Program normally operating/terminated<br>1 : Program error occurred | ○ | — |
| 12 | WRERR2<br>(Program operating condition) | 0 : Program normally operating/terminated<br>1 : Over-programming occurred | ○ | — |
| 13 - 15 | No functions assigned |  | 0 | — |

The Flash Status Register 2 (FSTAT2) consists of the following four read-only status bits which indicate the operating condition of the flash memory.

### (1) FBUSY (Flash Busy) bit (D8)

The FBUSY bit is used to determine whether the operation is terminated when programming or erasing the flash memory. When FBUSY = 0, it means the program or erase operation is being executed; when FBUSY = 1, the operation is terminated.

### (2) ERASE (Auto Erase operating condition) bit (D10)

The ERASE bit is used to determine whether execution of the flash memory erase operation has resulted in an error. When ERASE = 0, it means the erase operation terminated normally; when ERASE = 1, the operation terminated in an error.

### (3) WRERR1 (Program operating condition) bit (D11)

The WRERR1 bit is used to determine after completion of execution whether the flash memory program operation resulted in an error. When WRERR1 = 0, it means the program operation terminated normally; when WRERR1 = 1, the operation terminated in an error. The condition under which WRERR1 is set to 1 is when any bit other than those that must be 0 is found to be a 0 by comparison between the write data and the data in the flash memory.

### (4) WRERR2 (Program operating condition) bit (D12)

The WRERR2 bit is used to determine after execution whether the flash memory program operation resulted in an error. When WRERR2 = 0, it means the program operation terminated normally; when WRERR2 = 1, the operation terminated in an error. The condition under which WRERR2 is set to 1 is when the flash memory could not be programmed to by repeating the program operation a specified number of times.

Notes: • This status register is included in the internal flash memory itself, and can be read out by writing the Read Status Command (H'7070) to any address of the flash memory. For details, refer to Section 6.5, "Programming of Internal Flash Memory."

• While FBUSY bit = 0 (program/erase in progress), do not manipulate Flash Control Register 4 (FCNT4)’s FRESET bit.

### 6.4.3 Flash Control Registers

#### ■ Flash Control Register 1 (FCNT1)

<Address: H'0080 07E2>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  | FENTRY |  |  |  | FEMMOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 2 | No functions assigned |  | 0 | — |
| 3 | FENTRY<br>(Flash mode entry) | 0 : Normal read<br>1 : Erase/program enable | ○ | ○ |
| 4 - 6 | No functions assigned |  | 0 | — |
| 7 | FEMMOD<br>(Virtual flash emulation mode) | 0 : Normal mode<br>1 : Virtual Flash emulation mode | ○ | ○ |

The Flash Control Register 1 (FCNT1) consists of the following two bits to control the internal flash memory.

**(1) FENTRY (Flash Mode Entry) bit (D3)**

The FENTRY bit controls entry to flash E/W enable mode. Flash E/W enable mode can be entered only when FENTRY = 1.

To set the FENTRY bit to 1, write a 0 and then a 1 to the FENTRY bit in succession while the FP pin = high.

The FENTRY bit is cleared in the following cases:

- When a 0 is written to the FENTRY bit
- When the device is reset
- When the FP pin changes state from high to low

Note: • If while programming or erasing the flash memory, Flash Status Register 1 (FSTAT1)'s FSTAT bit = 0 (Busy) or Flash Status Register 2 (FSTAT2)'s FBUSY bit = 0 (program/erase in progress), do not clear the FENTRY bit.

When using a program in the flash memory while the FENTRY bit = 0, the EI vector entry is located at address H'0000 0080 of the flash memory. When running a flash write/erase program in RAM while the FENTRY bit = 1, the EI vector entry is located at address H'0080 4000 of the RAM, allowing for flash reprogram operation to be controlled using interrupts.

**Table 6.4.1 Changes of EI Vector Entry by FENTRY**

| FENTRY | EI Vector Entry | Address |
|---|---|---|
| 0 | Flash memory area | H'0000 0080 |
| 1 | Internal RAM area | H'0080 4000 |

### (2) FEMMOD (Virtual Flash Emulation Mode) bit (D7)

The FEMMOD bit controls entry to Virtual flash emulation mode. Virtual flash emulation mode is entered by setting the FEMMOD bit to 1 while the FENTRY bit = 0. (For details, refer to Section 6.7, "Virtual Flash Emulation Function.")

### ■ Flash Control Register 2 (FCNT2) <Address: H'0080 07E3>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  | FPROT |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 14 | No functions assigned |  | 0 | — |
| 15 | FPROT<br>(Unlock) | 0 : Protection by lock bit effective<br>1 : Protection by lock bit not effective | ○ | ○ |

The Flash Control Register 2 (FCNT2) controls invalidation of the internal flash memory protection by a lock bit (to disable erasing or programming of the flash memory). The flash memory protection becomes invalid (unlocked) by setting the FPROT bit to 1, so that any blocks protected by the lock bit can be erased or programmed.

To set the FPROT bit to 1, write a 0 and then a 1 to the FPROT bit in succession while the FENTRY bit = 1.

Also, the FPROT bit is cleared to 0 in one of the following cases:

- A low-level signal entered to the $\overline{\text{RESET}}$ pin
- FPROT bit reset by writing 0
- FP pin = low
- FENTRY bit cleared to 0

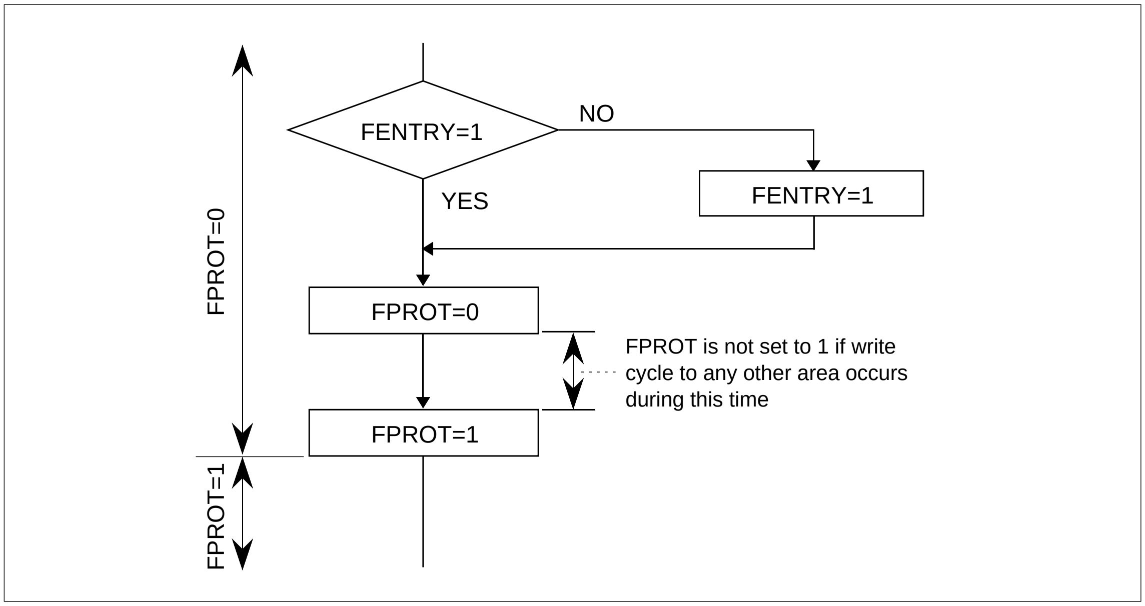

FENTRY=1
NO
YES
FENTRY=1
FPROT=0
FPROT=1
FPROT=0
FPROT=1
FPROT is not set to 1 if write cycle to any other area occurs during this time

**Figure 6.4.2 Protection Unlocking Flow**

### ■ Flash Control Register 3 (FCNT3)

<Address: H'0080 07E4>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  | FELEVEL |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 6 | No functions assigned |  | 0 | — |
| 7 | FELEVEL<br>(Raise erase margin) | 0 : Normal level<br>1 : Raise erase margin | ○ | ○ |

The Flash Control Register 3 (FCNT3) controls the depth of erase levels when erasing the internal flash memory with one of erase commands. By setting the FELEVEL bit to 1, the flash memory erase level can be deepened, which will result in an increased reliability margin.

### ■ Flash Control Register 4 (FCNT4)

<Address: H'0080 07E5>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  | FRESET |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 14 | No functions assigned |  | 0 | — |
| 15 | FRESET<br>(Reset flash) | 0 : No operation performed<br>1 : Reset the flash memory | ○ | ○ |

The Flash Control Register 4 (FCNT4) controls canceling program/erase operation in the middle and initializing each status bit of Flash Status Register 2 (FSTAT2). When the FRESET bit is set to 1, program/erase operation is canceled in the middle and each status bit of FSTAT2 is initialized (H'80).

The FRESET bit is effective only when the FENTRY bit = 1. Information on FRESET bit is ignored unless the FENTRY bit = 1.

Make sure that when programming or erasing the flash memory, the FRESET bit remains 0.

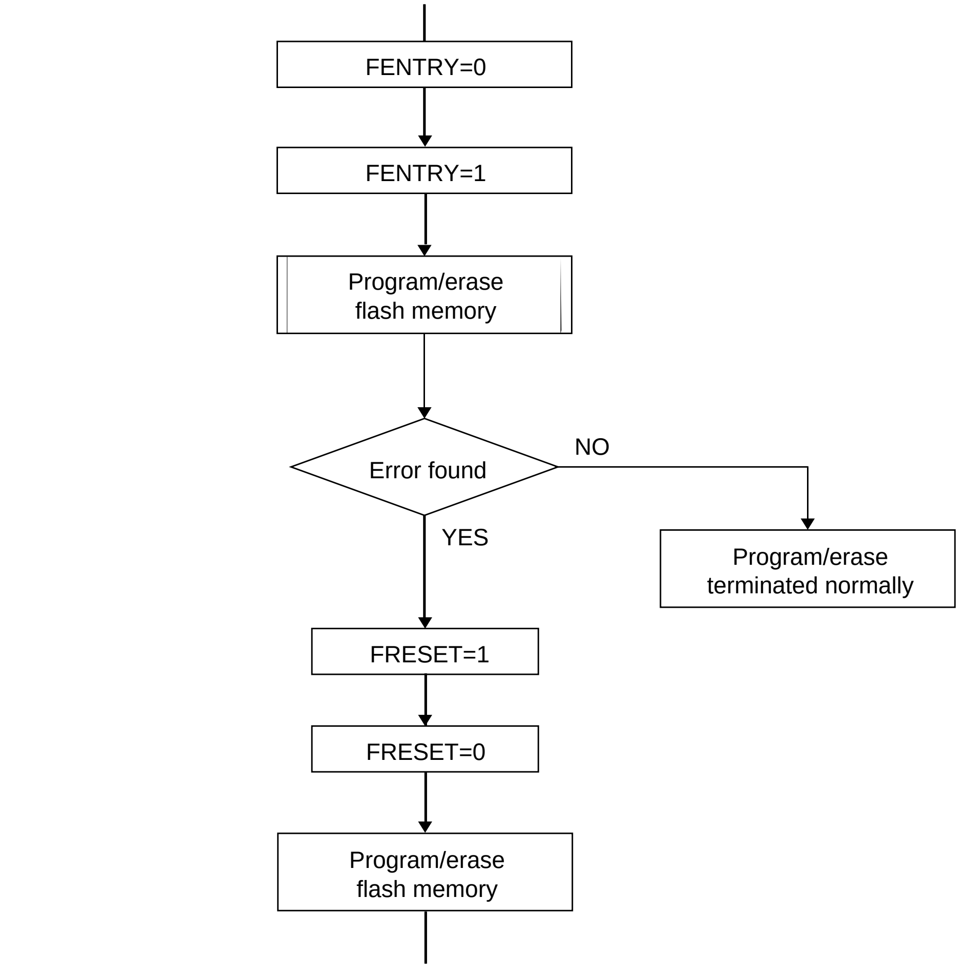

FENTRY=0
FENTRY=1
Program/erase flash memory
Error found
NO
YES
Program/erase terminated normally
FRESET=1
FRESET=0
Program/erase flash memory

**Figure 6.4.3 FCNT4 Register Usage Example 1 (Initializing Flash Status Register 2)**

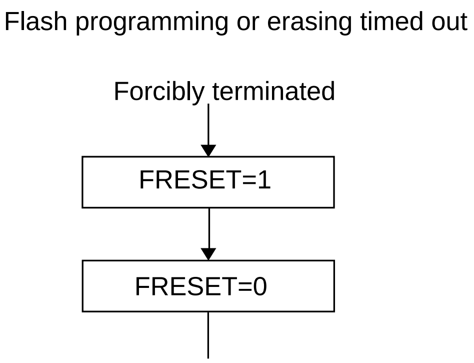
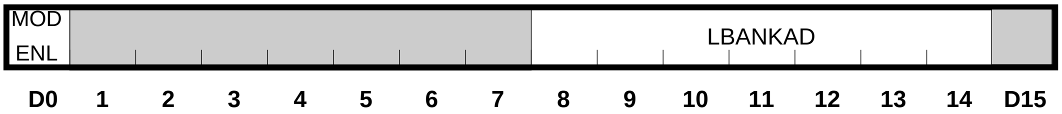


**Figure 6.4.4 FCNT4 Register Usage Example 2 (Forcibly terminating flash memory programming/erasing)**

### 6.4.4 Virtual Flash L Bank Register

**■ Virtual Flash L Bank Register 0 (FELBANK0)** <Address: H'0080 07E8>



<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MODENL<br>(Virtual flash emulation enable) | 0 : Disable virtual flash function<br>1 : Enable virtual flash function | ○ | ○ |
| 1 - 7 | No functions assigned | | 0 | — |
| 8 - 14 | LBANKAD<br>(L bank address) | A12 - A18 of start address of the L bank<br>to be selected | ○ | ○ |
| 15 | No functions assigned | | 0 | — |

Note: • This register must always be accessed in halfword.

**(1) MODENL (Virtual Flash Emulation Enable) bit (D0)**

The MODENL bit can be set to 1 after entering virtual flash emulation mode (by setting the FEMMOD bit to 1 while the FENTRY bit = 0). This causes the virtual flash emulation function to become effective for the L bank area selected by the LBANKAD bits.

**(2) LBANKAD (L Bank Address) bits (D8-D14)**

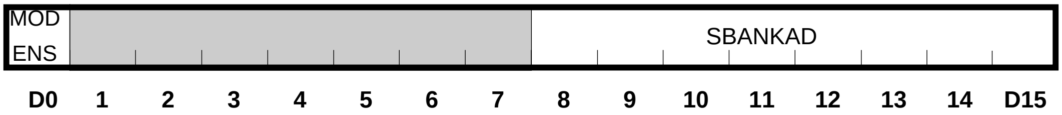
The LBANKAD bits are provided for selecting one L bank from L banks separated every 8 KB. Use these LBANKAD bits to set the seven bits, A12-A18, of the 32-bit start address of the L bank you want to select.

(For details, refer to Section 6.7, "Virtual Flash Emulation Function.")

### 6.4.5 Virtual Flash S Bank Registers

**■ Virtual Flash S Bank Register 0 (FESBANK0)** <Address: H'0080 07F0>

**■ Virtual Flash S Bank Register 1 (FESBANK1)** <Address: H'0080 07F2>



<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MODENS<br>(Virtual flash emulation enable) | 0 : Disable virtual flash function<br>1 : Enable virtual flash function | ○ | ○ |
| 1 - 7 | No functions assigned | | 0 | — |
| 8 - 15 | SBANKAD<br>(S bank address) | A12 - A19 of start address of the S bank<br>to be selected | ○ | ○ |

Note: • This register must always be accessed in halfword.

#### (1) MODENS (Virtual Flash Emulation Enable) bit (D0)

The MODENS bit can be set to 1 after entering virtual flash emulation mode (by setting the FEMMOD bit to 1 while the FENTRY bit = 0). This causes the virtual flash emulation function to become effective for the S bank area selected by the SBANKAD bits.

#### (2) SBANKAD (S Bank Address) bits (D8-D15)

The SBANKAD bits are provided for selecting one S bank from S banks separated every 4 KB. Use these SBANKAD bits to set the eight bits, A12-A19, of the 32-bit start address of the S bank you want to select.

(For details, refer to Section 6.7, "Virtual Flash Emulation Function.")

## 6.5 Programming of the Internal Flash Memory

### 6.5.1 Outline of Programming Flash Memory

When programming to the internal flash memory, there are following two methods to use depending on situation:

(1) When the write program does not exist in the internal flash memory
(2) When the write program already exists in the internal flash memory

For (1), set the FP pin = high, MOD0 = high, and MOD1 = low to enter boot mode. In this case, the reset vector entry is located at the beginning of the boot program area (H'8000 0000). (Normally, the reset vector entry is located at the start address of the internal flash memory.) Transfer the "flash write/erase program" from the boot area into the internal RAM using a boot program. After this transfer, jump to the RAM and set the Flash Control Register 1 FENTRY bit to 1 to make the flash memory ready for program(flash E/W enable mode). You now can program to the internal flash memory using the "flash write/erase program" that has been transferred into the internal RAM.

For (2), set the FP pin = high, MOD0 = low, and MOD1 = low to enter single-chip mode. Transfer the "flash write/erase program" from the internal flash memory in which it has been prepared beforehand into the internal RAM. After this transfer, jump to the RAM and set the Flash Control Register 1 (FCNT1) FENTRY bit to 1 using a program in the RAM to make the flash memory ready for program(flash E/W enable mode). You now can program to the internal flash memory using the "flash write/erase program" that has been transferred into the internal RAM. Or you can set the FP pin = high, MOD0 = low, and MOD1 = high to enter flash E/W enable mode in external extension mode.

When in flash E/W enable mode (FP pin = 1, FENTRY bit = 1), the EIT vector entry for External Interrupt (EI) is moved to the beginning of the internal RAM (H'0080 4000). During normal mode, the EIT vector entry exists in the flash area (H'0000 0080).

When using external interrupts (EI) in flash E/W enable mode, write at the beginning of the internal RAM the instruction for branching to the external interrupt (EI) handler that has been transferred into the internal RAM. Also, because the IVECT register which is read out in the external interrupt (EI) handler has stored in it the flash memory address of the ICU vector table, prepare the ICU vector table to be used during flash E/W enable mode in the internal RAM and convert its address from the IVECT register value to the internal RAM address (by, for example, adding an offset) when jumping to the handler.

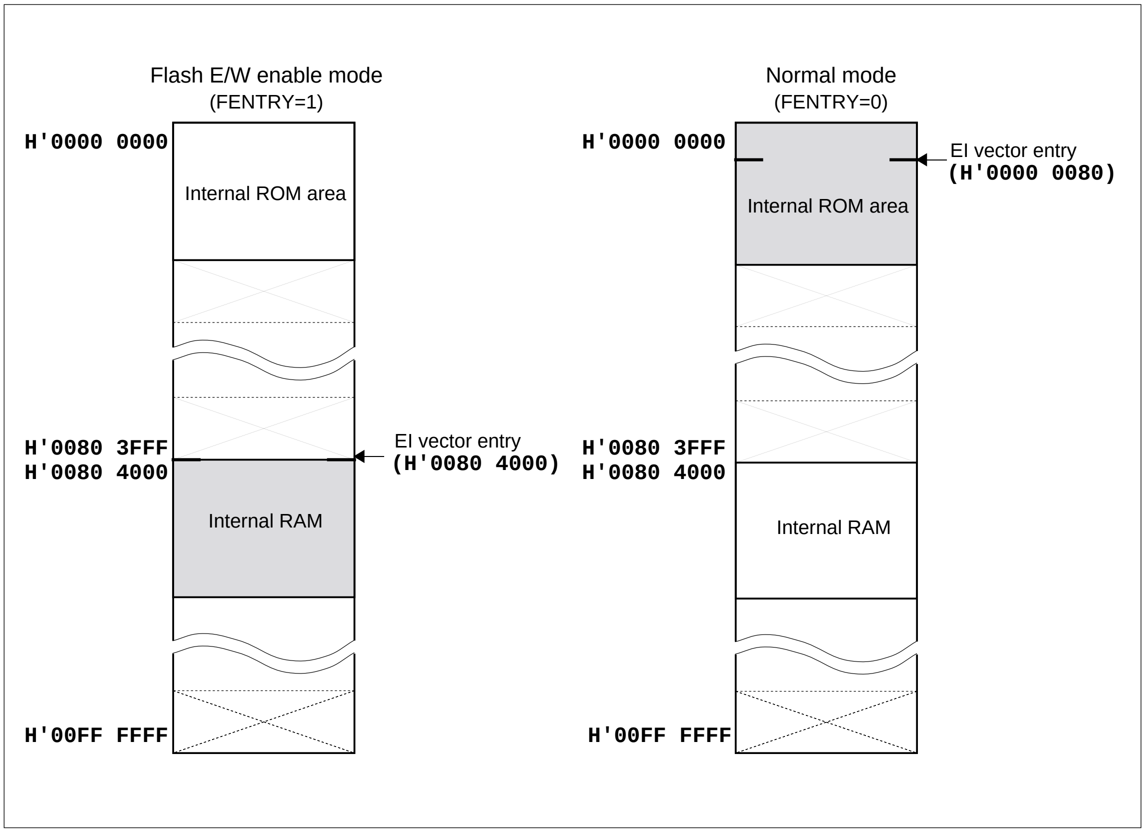
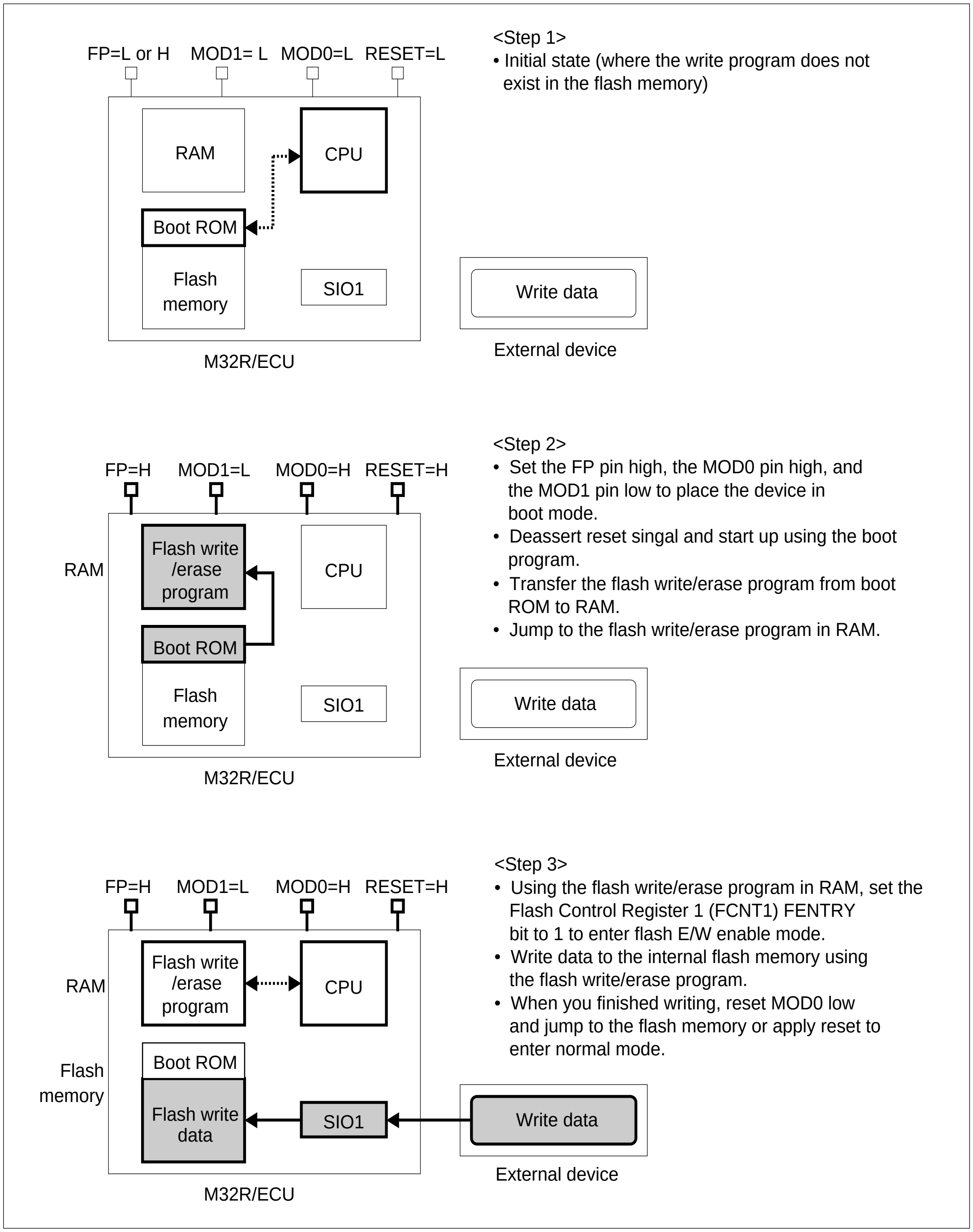


Figure 6.5.1 EI Vector Entry When in Flash E/W Enable Mode

### (1) When the write program does not exist in the internal flash memory

Use a program in the boot ROM located on memory map to program to the flash memory. To transfer the write data, use serial I/O1 in clock-synchronized serial mode. Use this serial transfer when writing to the flash memory using a flash programmer.



<Step 1>

- Initial state (where the write program does not exist in the flash memory)

<Step 2>

- Set the FP pin high, the MOD0 pin high, and the MOD1 pin low to place the device in boot mode.
- Deassert reset singal and start up using the boot program.
- Transfer the flash write/erase program from boot ROM to RAM.
- Jump to the flash write/erase program in RAM.

<Step 3>

- Using the flash write/erase program in RAM, set the Flash Control Register 1 (FCNT1) FENTRY bit to 1 to enter flash E/W enable mode.
- Write data to the internal flash memory using the flash write/erase program.
- When you finished writing, reset MOD0 low and jump to the flash memory or apply reset to enter normal mode.

**Figure 6.5.2 Procedure for Writing to Internal Flash Memory** (when the write program does not exist in the flash memory)

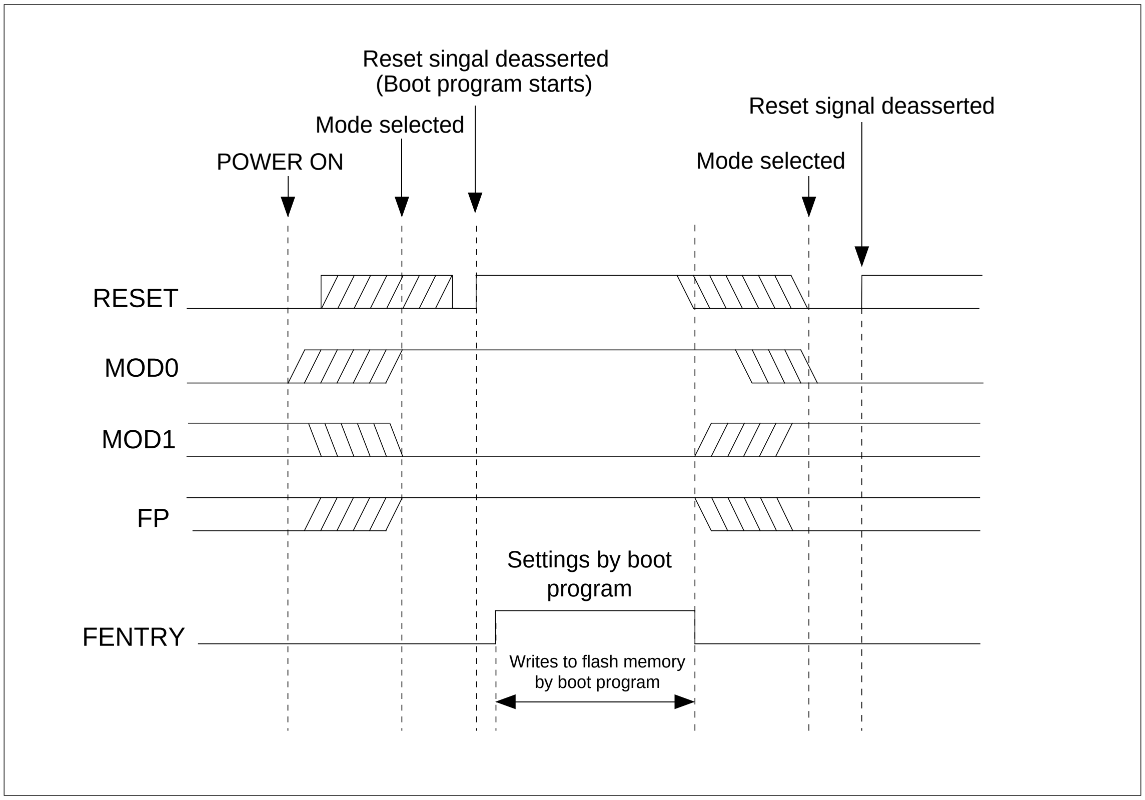
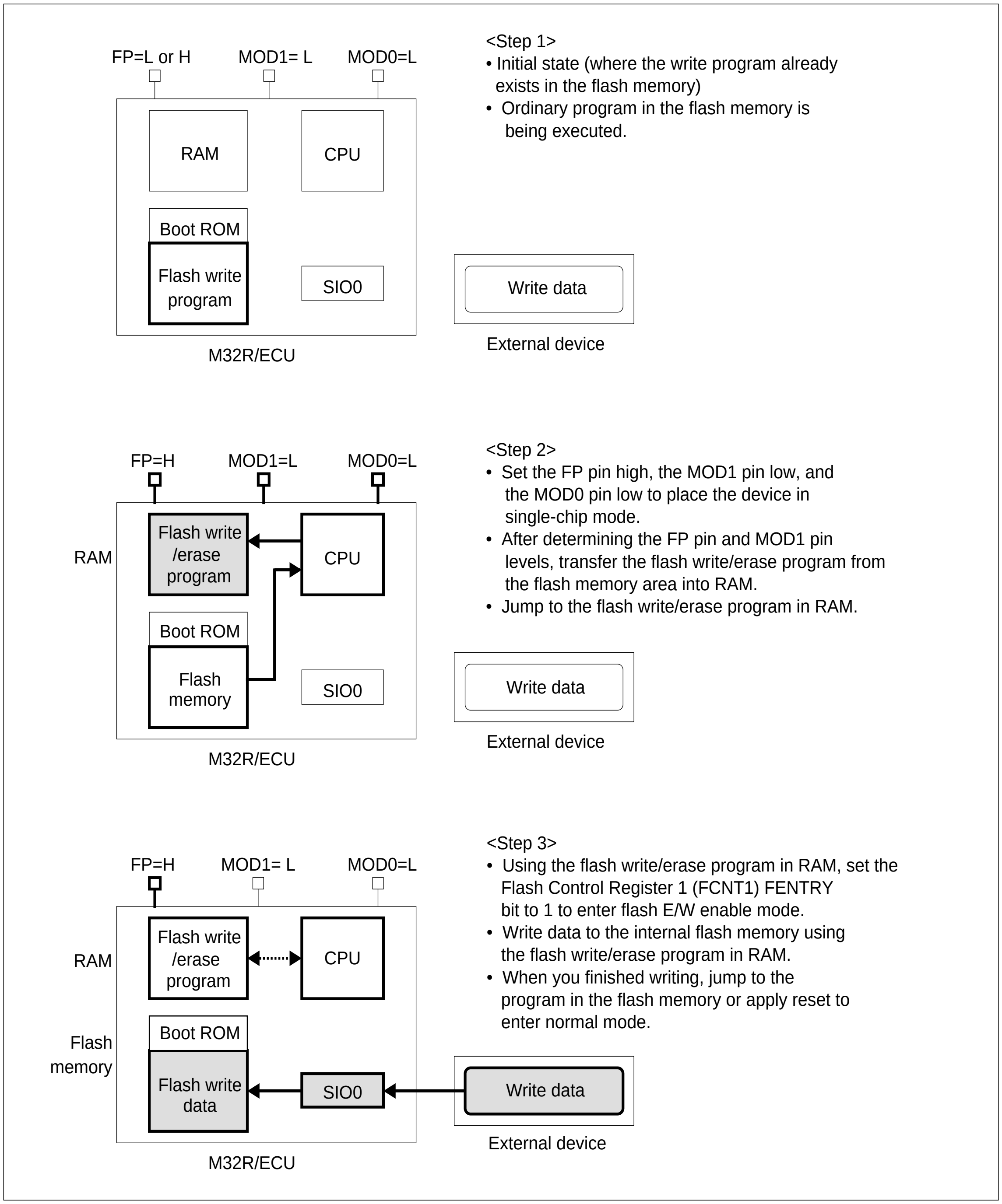


**Figure 6.5.3 Internal Flash Memory Write Timings** (when the write program does not exist in the flash memory)

### (2) When the write program already exists in the internal flash memory

Use the flash write/erase program already stored in the internal flash memory to program to the flash memory. For program to the flash memory, use the internal peripheral circuits according to your programming system. (The data bus, serial I/O, and ports can be used.)
The following shows an example for writing to the flash memory by using serial I/O0 in single-chip mode.



<Step 1>

- Initial state (where the write program already exists in the flash memory)
- Ordinary program in the flash memory is being executed.

<Step 2>

- Set the FP pin high, the MOD1 pin low, and the MOD0 pin low to place the device in single-chip mode.
- After determining the FP pin and MOD1 pin levels, transfer the flash write/erase program from the flash memory area into RAM.
- Jump to the flash write/erase program in RAM.

<Step 3>

- Using the flash write/erase program in RAM, set the Flash Control Register 1 (FCNT1) FENTRY bit to 1 to enter flash E/W enable mode.
- Write data to the internal flash memory using the flash write/erase program in RAM.
- When you finished writing, jump to the program in the flash memory or apply reset to enter normal mode.

**Figure 6.5.4 Procedure for Writing to Internal Flash Memory** (when the write program already exists in the flash memory)

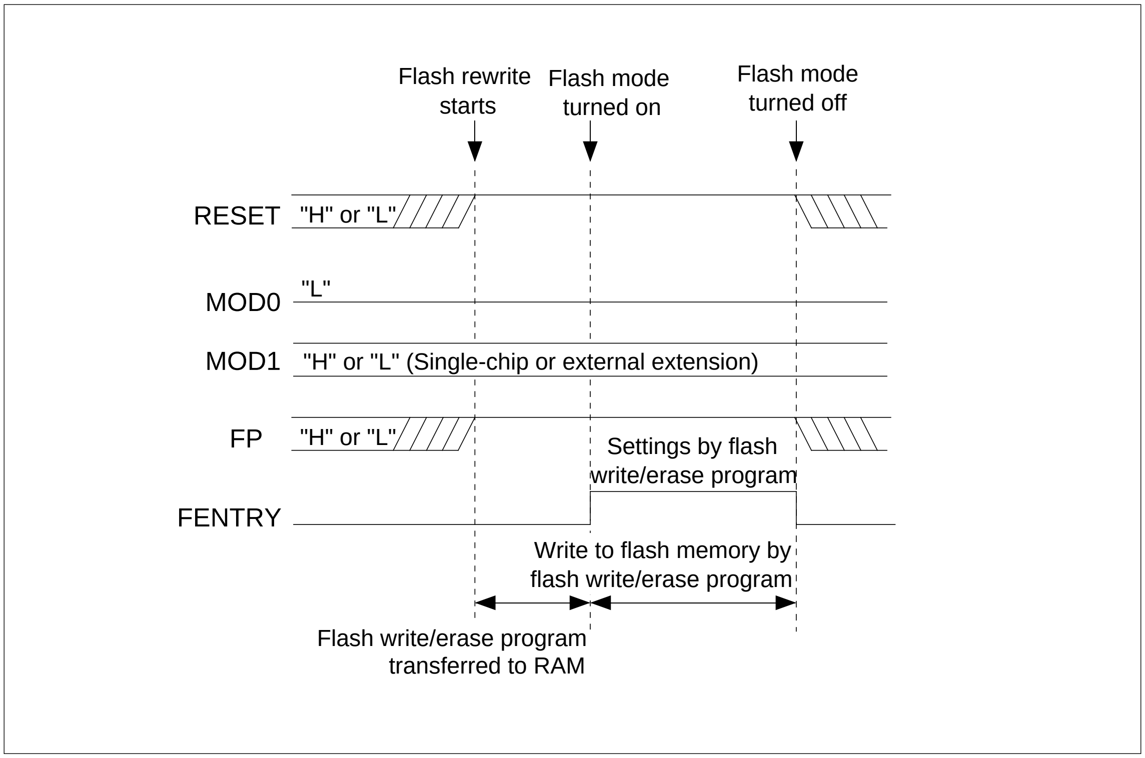

Flash rewrite starts
Flash mode turned on
Flash mode turned off
RESET
"H" or "L"
MOD0
"L"
MOD1
"H" or "L" (Single-chip or external extension)
FP
"H" or "L"
Settings by flash write/erase program
FENTRY
Write to flash memory by flash write/erase program
Flash write/erase program transferred to RAM

**Figure 6.5.5 Internal Flash Memory Write Timings** (when the write program already exists in the flash memory)

## 6.5.2 Controlling Operation Mode during Programming Flash

The device's operation modes are set by MOD0, MOD1, and Flash Control Register 1 (FCNT1) FENTRY bit. The table below lists operation modes that may be set during flash program.

**Table 6.5.1 Operation Modes Set during Flash Program**

| FP | MOD0 | MOD1 | FENTRY | Operation Mode | Reset Vector Entry | EI Vector Entry |
|---|---|---|---|---|---|---|
| L | L | L | 0 | Single-chip mode | Start address of flash memory (H'0000 0000) | Flash area (H'0000 0080) |
| H | L | L | 0 | | | |
| L | H | L | 0 | Processor mode | Start address of external area (H'0000 0000) | External area (H'0000 0080) |
| L | L | H | 0 | External extension mode | Start address of flash memory (H'0000 0000) | Flash area (H'0000 0080) |
| H | L | H | 0 | | | |
| H | L | L | 1 | Single-chip mode + flash E/W enable | Start address of flash memory (H'0000 0000) | Beginning of internal RAM (H'0080 4000) |
| H | H | L | 0 | Boot mode | Start address of boot program area (H'8000 0000) | Flash area (H'0000 0080) |
| H | H | L | 1 | Boot mode + flash E/W enable | Start address of boot program area (H'8000 0000) | Beginning of internal RAM (H'0080 4000) |
| H | L | H | 1 | External extension mode + flash E/W enable | Start address of flash memory (H'0000 0000) | Beginning of internal RAM (H'0080 4000) |
| — (Note 1) | H | H | — (Note 1) | reserved (use inhibited) | | |

Note 1: The bar "—" denotes "Don't Care."

### (1) Flash E/W enable mode

Flash E/W enable mode is a mode in which the internal flash memory can be programmed or erased. In flash E/W enable mode, no programs can be executed in the internal flash memory. Therefore, before entering flash E/W enable mode, you need to transfer the necessary program into the internal RAM and run the program in RAM.

### (2) Entering flash E/W enable mode

Flash E/W enable mode can be entered only when the device is operating in single-chip mode or external extension mode. Namely, you can enter flash E/W enable mode only when the FP pin = high and the Flash Control Register 1 (FCNT1) FENTRY bit = 1. You cannot enter flash E/W enable mode when the device is operating in processor mode or the FP pin = low.

### (3) Detecting the MOD0 and MOD1 pin levels

The MOD0 and MOD1 pin levels (high or low) can be verified using the P8 Data Register (Port Data Register, H'00800 0708) MOD0DT and MOD1DT bits.

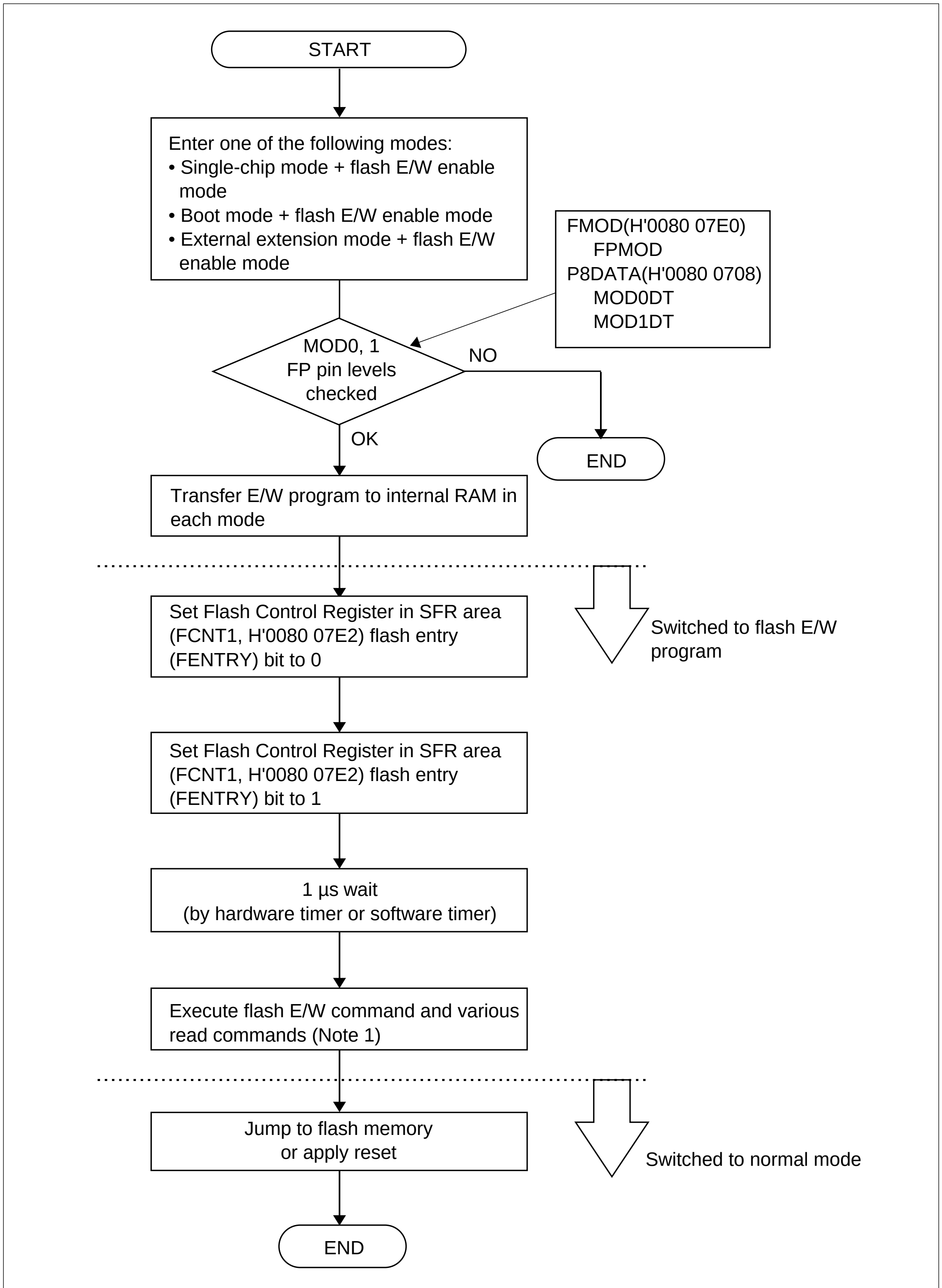
### ■ P8 Data Register (P8DATA)

<Address: H'0080 0708>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MOD0DT | MOD1DT | P82DT | P83DT | P84DT | P85DT | P86DT | P87DT |

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MOD0DT<br>(MOD0 data) | 0 : MOD0 pin = low<br>1 : MOD0 pin = high | ○ | — |
| 1 | MOD1DT<br>(MOD1 data) | 0 : MOD1 pin = low<br>1 : MOD1 pin = high | ○ | — |
| 2 | P82DT<br>(Port P82 data) | Depending on how the Port Direction Register is set<br>• When direction bit = 0 (input mode) | ○ | ○ |
| 3 | P83DT<br>(Port P83 data) | 0: Port input pin = low<br>1: Port input pin = high | ○ | ○ |
| 4 | P84DT<br>(Port P84 data) | • When direction bit = 1 (output mode)<br>0: Port output latch = low | ○ | ○ |
| 5 | P85DT<br>(Port P85 data) | 1: Port output latch = high | ○ | ○ |
| 6 | P86DT<br>(Port P86 data) | | ○ | ○ |
| 7 | P87DT<br>(Port P87 data) | | ○ | ○ |


START
Enter one of the following modes:
• Single-chip mode + flash E/W enable mode
• Boot mode + flash E/W enable mode
• External extension mode + flash E/W enable mode
FMOD(H'0080 07E0)
FPMOD
P8DATA(H'0080 0708)
MOD0DT
MOD1DT
MOD0, 1
FP pin levels
checked
NO
END
OK
Transfer E/W program to internal RAM in each mode
Set Flash Control Register in SFR area (FCNT1, H'0080 07E2) flash entry (FENTRY) bit to 0
Switched to flash E/W program
Set Flash Control Register in SFR area (FCNT1, H'0080 07E2) flash entry (FENTRY) bit to 1
1 µs wait
(by hardware timer or software timer)
Execute flash E/W command and various read commands (Note 1)
Jump to flash memory
or apply reset
Switched to normal mode
END

Note 1: For details about each command, refer to Section 6.5.3, "Programming Procedure to Internal Flash Memory."

**Figure 6.5.6 Procedure for Entering Flash E/W Enable Mode**

### 6.5.3 Programming Procedure to the Internal Flash Memory

To program to the internal flash memory, set the device's operation mode to enter flash E/W enable mode first and then use the flash write/erase program that has already been transferred from the flash memory into the internal RAM.
In flash E/W enable mode, no data can be read out from the internal flash memory as in normal mode, so you cannot execute a program that exists in the internal flash memory. Therefore, the flash write/erase program must be prepared in the internal RAM before entering flash E/W enable mode. (Once you've entered flash E/W enable mode, you cannot use any command except flash commands to access the flash memory.)

To access the internal flash memory in flash memory E/W enable mode, issue commands for the internal flash memory address to be operated on. The table below lists the commands that can be issued in flash memory E/W enable mode.

Note: • During flash E/W enable mode, the flash memory cannot be accessed for read or write wordwise.

**Table 6.5.2 Commands in Flash Memory E/W Enable Mode**

| Command Name | Issued Command Data |
|---|---|
| Read Array command | H'FFFF |
| Page Program command | H'4141 |
| Lock Bit Program command | H'7777 |
| Block Erase command | H'2020 |
| Erase All Unlock Block command | H'A7A7 |
| Read Status Register command | H'7070 |
| Clear Status Register command | H'5050 |
| Read Lock Bit Status command | H'7171 |
| Verify command (Note1 - 4) | H'D0D0 |

Note 1: This command is used in conjunction with Lock Bit Program, Block Erase, and Erase All Unlock Block operations.

Note 2: Always issue this command successively after the Lock Bit Program, Block Erase, or Erase All Unlock Block command.

Note 3: If the Read Array command (H’FFFF) is issued after the Lock Bit Program, Block Erase, or Erase All Unlock Block command, each of those preceding commands is canceled.

Note 4: If other than the Verify command (H’D0D0) and Read Array command (H’FFFF) are issued after the Lock Bit Program, Block Erase, or Erase All Unlock Block command, each of those preceding commands terminates in an error without ever being executed.

### (1) Read Array command

Read mode is entered by writing command data H'FFFF to any address of the internal flash memory. Then read the flash memory address you want to read out, and the content of that address will be read out.

Before exiting flash E/W enable mode, always be sure to execute the Read Array command.

### (2) Page Program command

Flash memory is programmed one page at a time, each page consisting of 256 bytes (lower addresses H'00 to H'FF). To write data to the flash memory (i.e., to program the flash memory), write the program command H'4141 to any address of the internal flash memory and then the program data to the address to which you want to write.

With the Page Program command, you cannot program to the protected blocks.

Page Program is automatically performed by the internal control circuit, and the completion of programming can be verified by checking the Flash Status Register 1 (FSTAT1) FSTAT bit. (Refer to Section 6.4.2, "Flash Status Registers.") While the FSTAT bit = 0, the next programming can not be performed.

### (3) Lock Bit Program command

Flash memory can be protected against program/erase one block at a time. The Lock Bit Program command is provided for protecting memory blocks.
Write the Lock Bit Program command data H'7777 to any address of the internal flash memory. Next, write the Verify command data H'D0D0 to the last even address of the block you want to protect, and this memory block is protected against program/erase. To remove protection, disable lock bit-effectuated protection using the Flash Control Register 2 (FCNT2) FPROT bit (see Section 6.4.3, "Flash Control Registers") and erase the block whose protection you want to remove. (The content of this memory block is also erased.)
The tables 6.5.3 to 6.5.5 list the target blocks and their specified addresses when writing the Verify command data.

**Table 6.5.3 M32171F4 Target Blocks and Specified Addresses**

| Target Block | Specified Address |
|---|---|
| 0 | H'0000 3FFE |
| 1 | H'0000 5FFE |
| 2 | H'0000 7FFE |
| 3 | H'0000 FFFE |
| 4 | H'0001 FFFE |
| 5 | H'0002 FFFE |
| 6 | H'0003 FFFE |
| 7 | H'0004 FFFE |
| 8 | H'0005 FFFE |
| 9 | H'0006 FFFE |
| 10 | H'0007 FFFE |

**Table 6.5.4 M32171F3 Target Blocks and Specified Addresses**

| Target Block | Specified Address |
|---|---|
| 0 | H'0000 3FFE |
| 1 | H'0000 5FFE |
| 2 | H'0000 7FFE |
| 3 | H'0000 FFFE |
| 4 | H'0001 FFFE |
| 5 | H'0002 FFFE |
| 6 | H'0003 FFFE |
| 7 | H'0004 FFFE |
| 8 | H'0005 FFFE |

**Table 6.5.5 M32171F2 Target Blocks and Specified Addresses**

| Target Block | Specified Address |
|---|---|
| 0 | H'0000 3FFE |
| 1 | H'0000 5FFE |
| 2 | H'0000 7FFE |
| 3 | H'0000 FFFE |
| 4 | H'0001 FFFE |
| 5 | H'0002 FFFE |
| 6 | H'0003 FFFE |

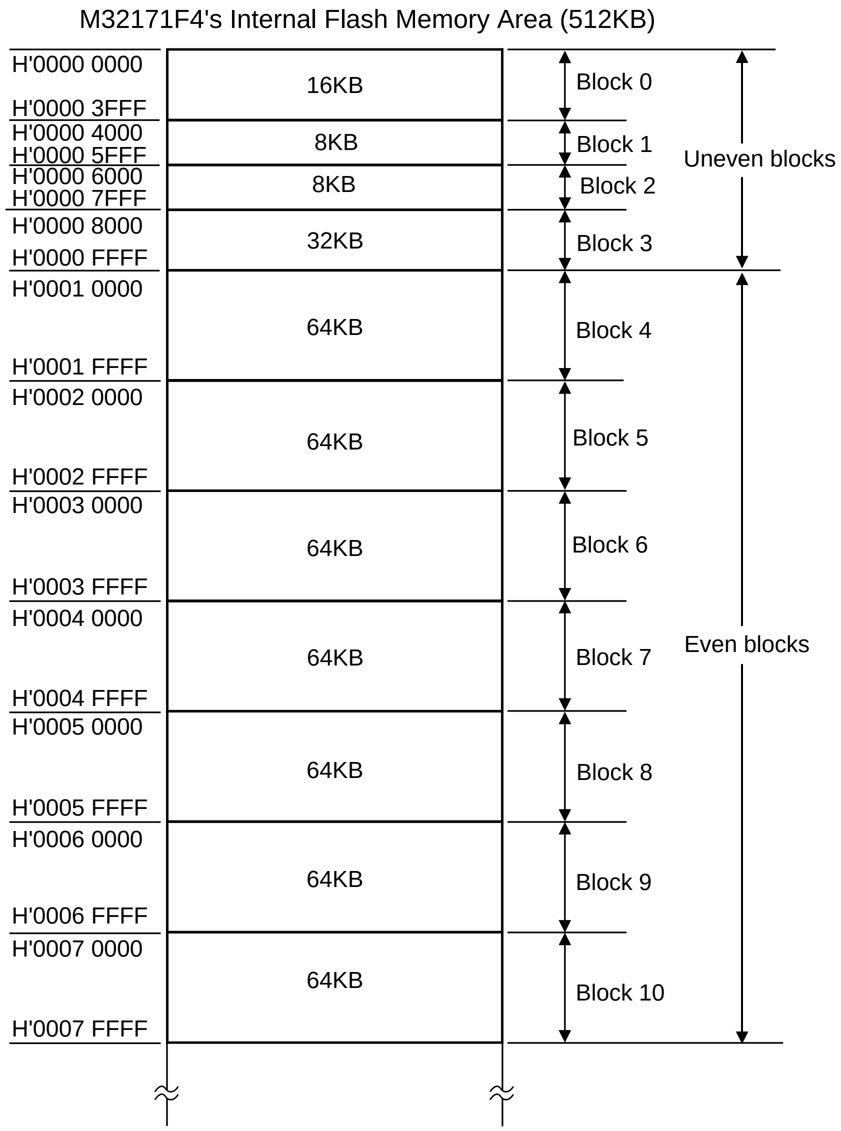


**Figure 6.5.7 Block Configuration of the M32171F4 Flash Memory**

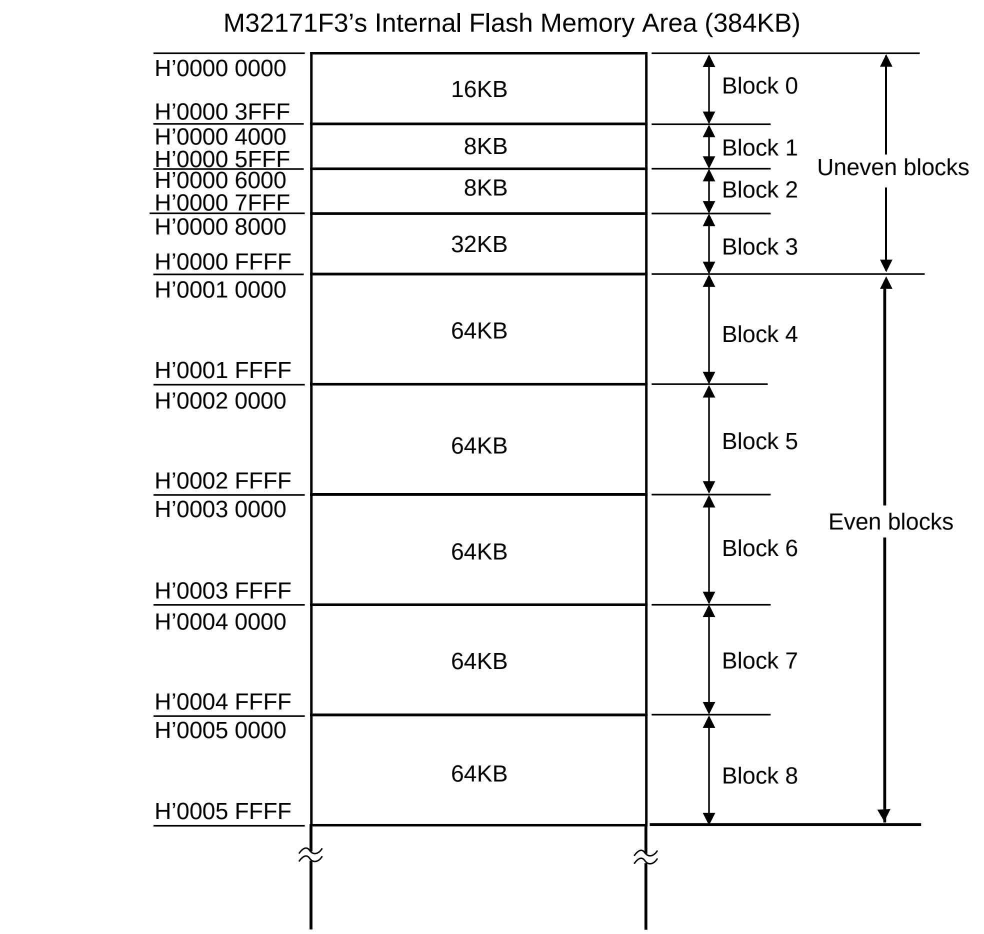


**Figure 6.5.8 Block Configuration of the M32171F3 Flash Memory**

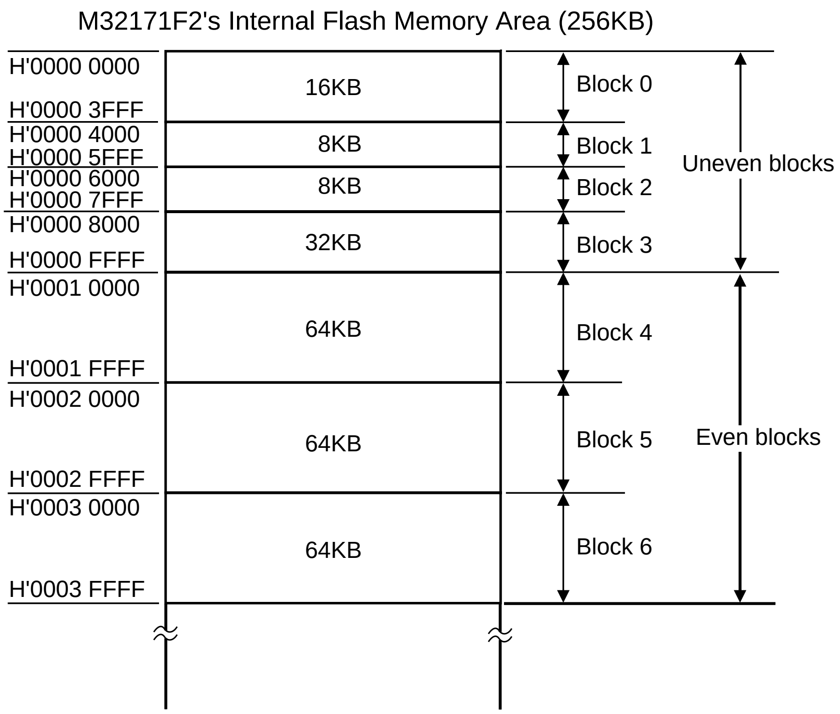


Figure 6.5.9 Block Configuration of the M32171F2 Flash Memory

### (4) Block Erase command

The Block Erase command erases the contents of internal flash memory one block at a time. For Block Erase, write the command data H'2020 to any address of the internal flash memory. Next, write the Verify command data H'D0D0 to the last even address of the memory block you want to erase (see Table 6.5.3, Table 6.5.4, and Table 6.5.5, "Target Blocks and Specified Addresses"). The content of this memory block is erased.

With the Block Erase command, you cannot erase the protected blocks.

Block Erase is automatically performed by the internal control circuit, and the completion of Block Erase can be verified by checking the Flash Status Register 1 (FSTAT1) FSTAT bit. (Refer to Section 6.4.2, "Flash Status Registers.") While the FSTAT bit = 0, you cannot erase the next block.

### (5) Erase All Unlock Block command

The Erase All Unlock Block command erases all memory blocks that are not protected. To erase all unlock blocks, write the command data H'A7A7 to any address of the internal flash memory. Next, write the command data H'D0D0 to any address of the internal flash memory, and all of unprotected memory blocks are erased.

### (6) Read Status Register command

The Read Status Register command reads out the content of Flash Status Register 2 (FSTAT2) that indicates whether flash memory write or erase operation has terminated normally or not. To read Flash Status Register 2, write the command data H'7070 to any address of the internal flash memory. Next, read any address of the internal flash memory, and the content of Flash Status Register 2 (FSTAT2) is read out.

### (7) Clear Status Register command

The Clear Status Register command clears the Flash Status Register 2 (FSTAT2) ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1), and WRERR2 (Program operating condition 2) bits to 0. Write the command data H'5050 to any address of the internal flash memory, and Flash Status Register 2 is cleared to 0.

If an error occurs when programming or erasing the flash memory and the Flash Status Register 2 (FSTAT2) ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1) or WRERR2 (Program operating condition 2) bit is set to 1, you cannot perform the next program or erase operation unless ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1) or WRERR2 (Program operating condition 2) is cleared to 0.

### (8) Read Lock Bit Status command

The Read Lock Bit Status command allows you to check whether or not a memory block is protected against program/erase. Write the command data H'7171 to any address of the internal flash memory. Next, read the last even address of the block you want to check (see Table 6.5.3, Table 6.5.4, and Table 6.5.5, "Target Blocks and Specified Addresses"), and the data you read shows whether or not the target block is protected. If the FLBST0 (lock bit 0) bit and FLBST1 (lock bit 1) bit of the data you read are 0s, it means that the target memory block is protected. If the FLBST0 (lock bit 0) bit and FLBST1 (lock bit 1) bit are 1s, it means that the target memory block is not protected.

## ■ Lock Bit Status Register (FLBST)

|  | FLBST0 |  |  |  |  |  |  |  | FLBST1 |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | No functions assigned |  | ? | — |
| 1 | FLBST0<br>(Lock bit 0) | 0 : Protected<br>1 :  Not protected | ○ | — |
| 2 - 8 | No functions assigned |  | ? | — |
| 9 | FLBST1<br>(Lock bit 1) | 0 : Protected<br>1 :  Not protected<br>(Same content as FLBST0 is output.) | ○ | — |
| 10 - 15 | No functions assigned |  | ? | — |

The Lock Bit Status Register is a read-only register, which contains said lock bits independently for each block.

Follow the procedure described below to write to the lock bits.

a) Setting the lock bit to 0 (protect the block)

Issue the Lock Bit Program command (H'7777) to the memory block you want to protect.

b) Setting the lock bit to 1 (unprotect the block)

After setting the Flash Control Register 2 FPROT bit to invalidate lock bit-effectuated protection, use the Block Erase command (H'2020) or Erase All Unprotect Block command (H'A7A7) to erase the memory block you want to unprotect. This is the only way to unprotect a memory block. You cannot set the lock bit alone to 1.

c) Status when the lock bit is reset

The lock bit is unaffected by a reset or power outage because it is a nonvolatile bit.

### (9) Execution flow of each command

The diagrams below show an execution flow of each command.

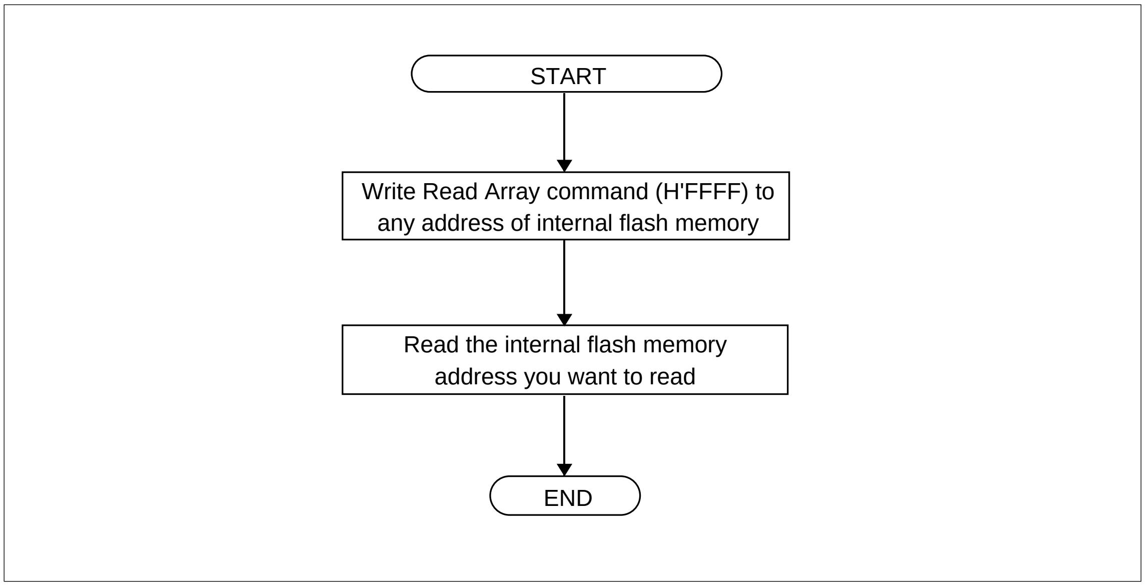
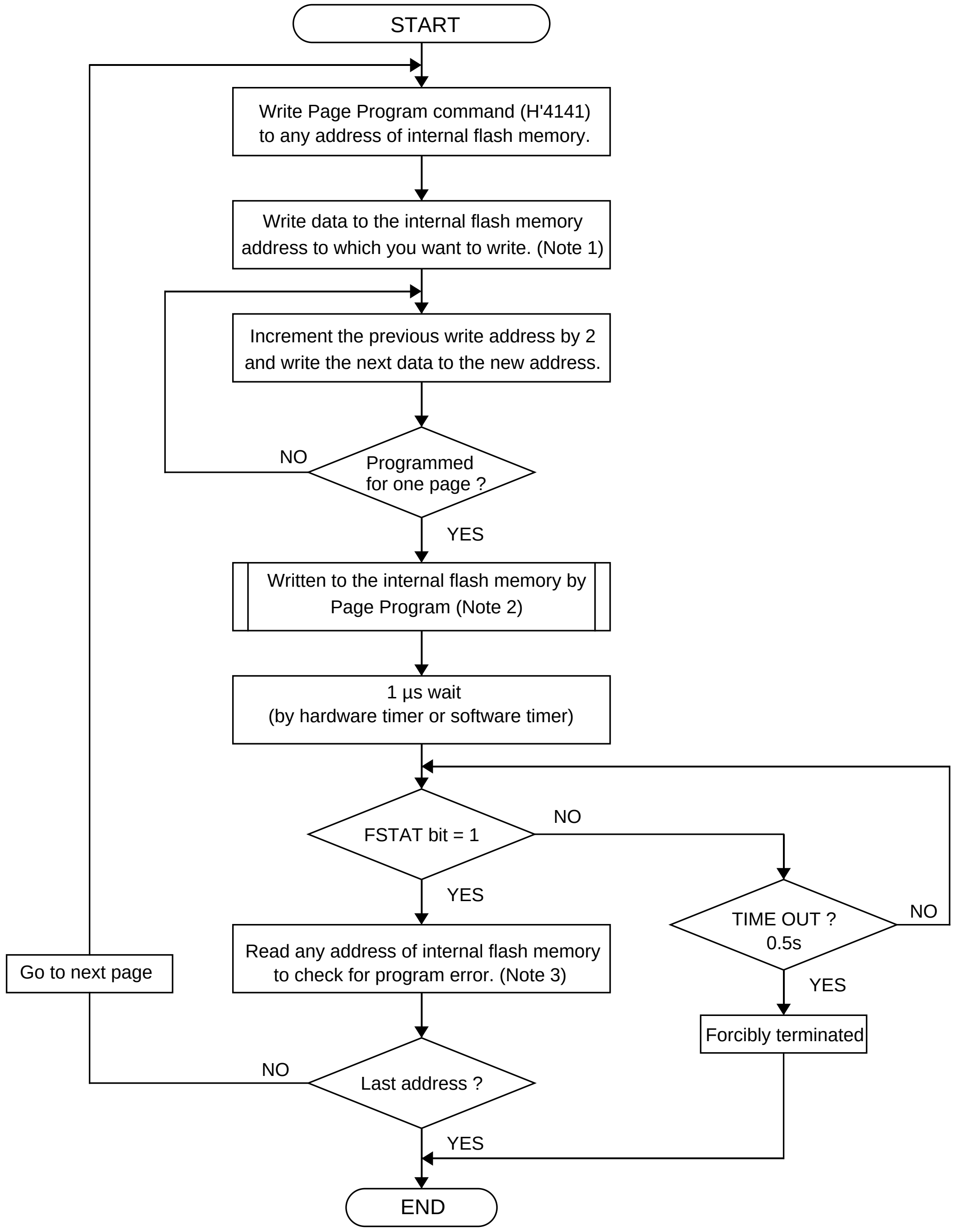

START
Write Read Array command (H'FFFF) to any address of internal flash memory
Read the internal flash memory address you want to read
END

**Figure 6.5.10 Read Array**

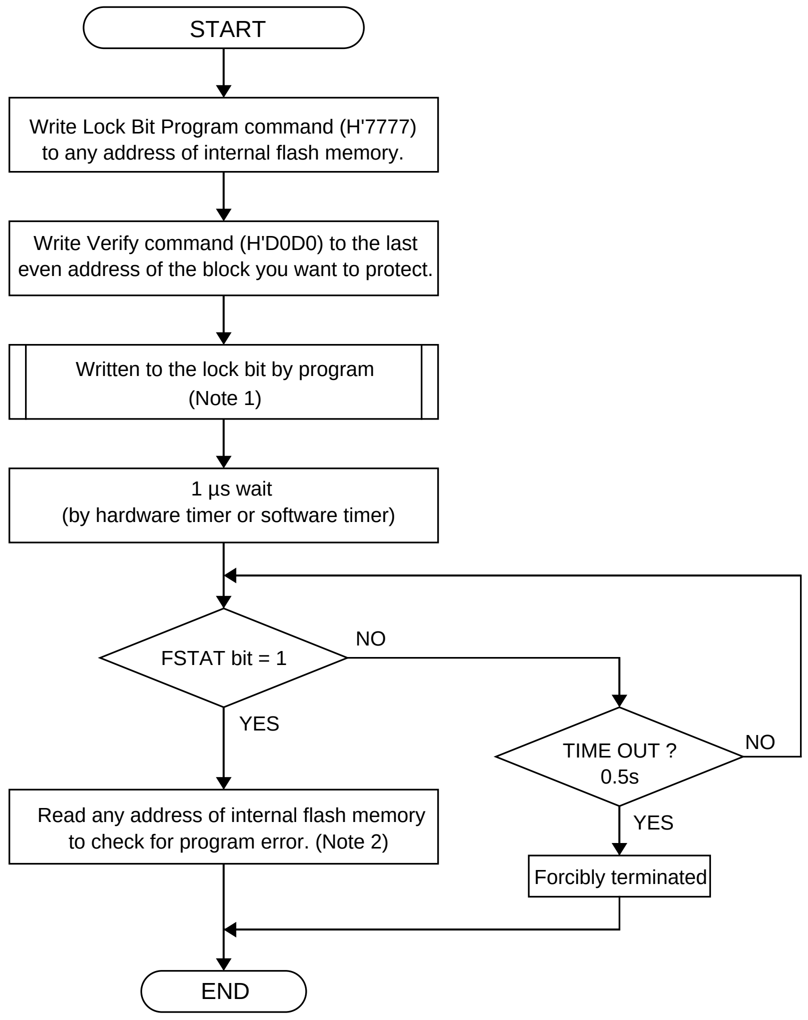

START
Write Page Program command (H'4141) to any address of internal flash memory.
Write data to the internal flash memory address to which you want to write. (Note 1)
Increment the previous write address by 2 and write the next data to the new address.
Programmed for one page ?
NO
YES
Written to the internal flash memory by Page Program (Note 2)
1 µs wait (by hardware timer or software timer)
FSTAT bit = 1
NO
YES
TIME OUT ? 0.5s
NO
YES
Forcibly terminated
Read any address of internal flash memory to check for program error. (Note 3)
Go to next page
Last address ?
NO
YES
END

Note 1: Start writing from the beginning of a 256-byte boundary of the flash memory (lower address H'00).

Note 2: When Program operation starts, you have the Read Status Register command automatically entered. (You do not need to enter the Read Status Register command until you issue another command.)

Note 3: Examine the Flash Status Register 2 ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1), and WRERR2 (Program operating condition 2) bits to check for program error.

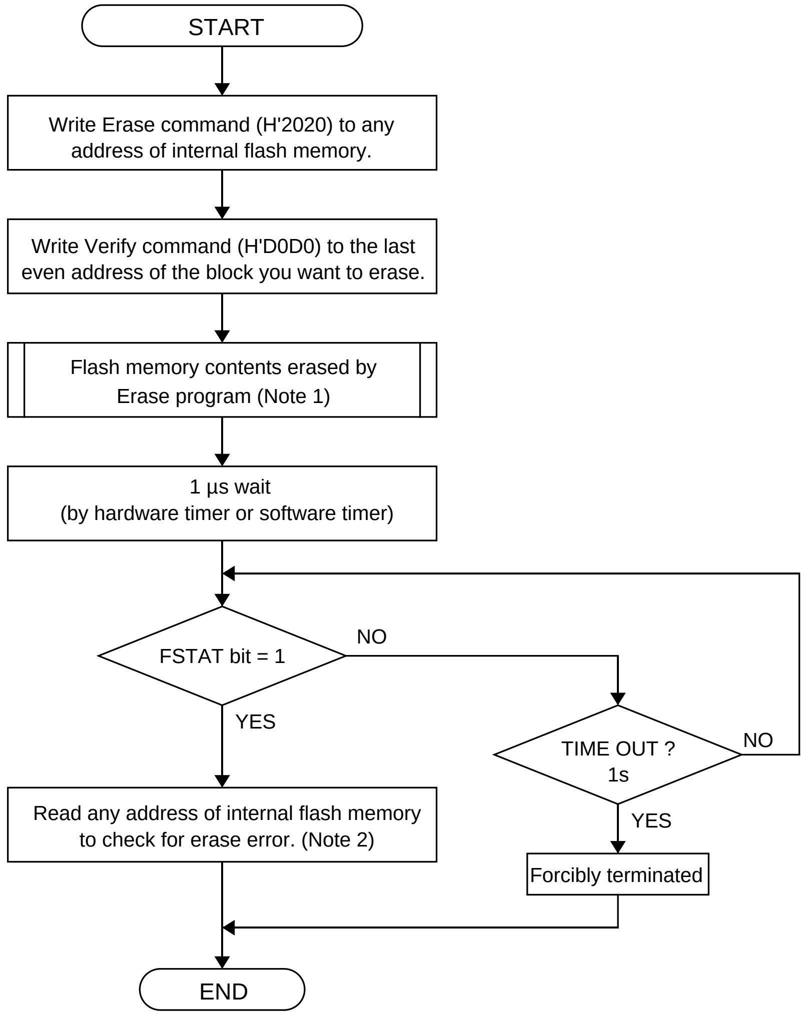
**Figure 6.5.11 Page Program**


START
Write Lock Bit Program command (H'7777) to any address of internal flash memory.
Write Verify command (H'D0D0) to the last even address of the block you want to protect.
Written to the lock bit by program (Note 1)
1 µs wait (by hardware timer or software timer)
FSTAT bit = 1
NO
YES
TIME OUT ? 0.5s
NO
YES
Read any address of internal flash memory to check for program error. (Note 2)
Forcibly terminated
END

Note 1: When Program operation starts, you have the Read Status Register command automatically entered. (You do not need to enter the Read Status Register command until you issue another command.)

Note 2: Examine the Flash Status Register 2 ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1), and WRERR2 (Program operating condition 2) bits to check for program error.

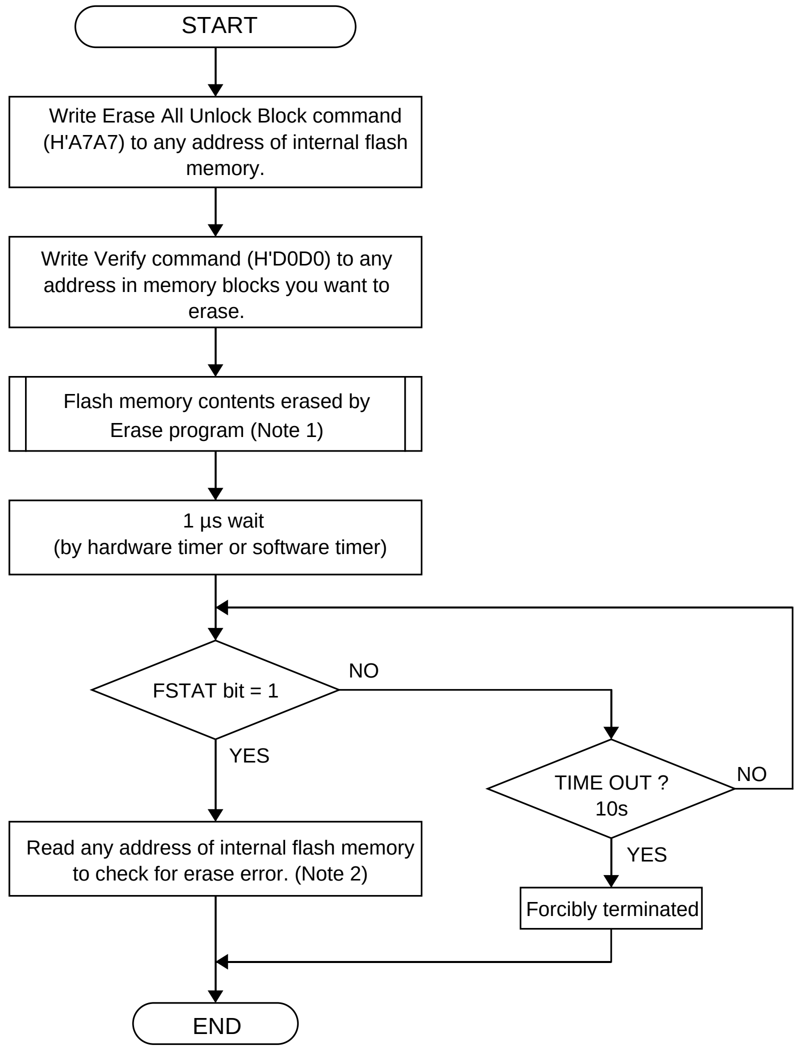
**Figure 6.5.12 Lock Bit Program**


START
Write Erase command (H'2020) to any address of internal flash memory.
Write Verify command (H'D0D0) to the last even address of the block you want to erase.
Flash memory contents erased by Erase program (Note 1)
1 µs wait (by hardware timer or software timer)
FSTAT bit = 1
NO
YES
TIME OUT ? 1s
NO
YES
Read any address of internal flash memory to check for erase error. (Note 2)
Forcibly terminated
END

Note 1: When Erase operation starts, you have the Read Status Register command automatically entered. (You do not need to enter the Read Status Register command until you issue another command.)

Note 2: Examine the Flash Status Register 2 ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1), and WRERR2 (Program operating condition 2) bits to check for erase error.

**Figure 6.5.13 Block Erase**


START
Write Erase All Unlock Block command (H'A7A7) to any address of internal flash memory.
Write Verify command (H'D0D0) to any address in memory blocks you want to erase.
Flash memory contents erased by Erase program (Note 1)
1 µs wait (by hardware timer or software timer)
FSTAT bit = 1
NO
YES
TIME OUT ? 10s
NO
YES
Read any address of internal flash memory to check for erase error. (Note 2)
Forcibly terminated
END

Note 1: When Erase operation starts, you have the Read Status Register command automatically entered. (You do not need to enter the Read Status Register command until you issue another command.)

Note 2: Examine the Flash Status Register 2 ERASE (Auto Erase operating condition), WRERR1 (Program operating condition 1), and WRERR2 (Program operating condition 2) bits to check for erase error.

**Figure 6.5.14 Erase All Unlock Block**

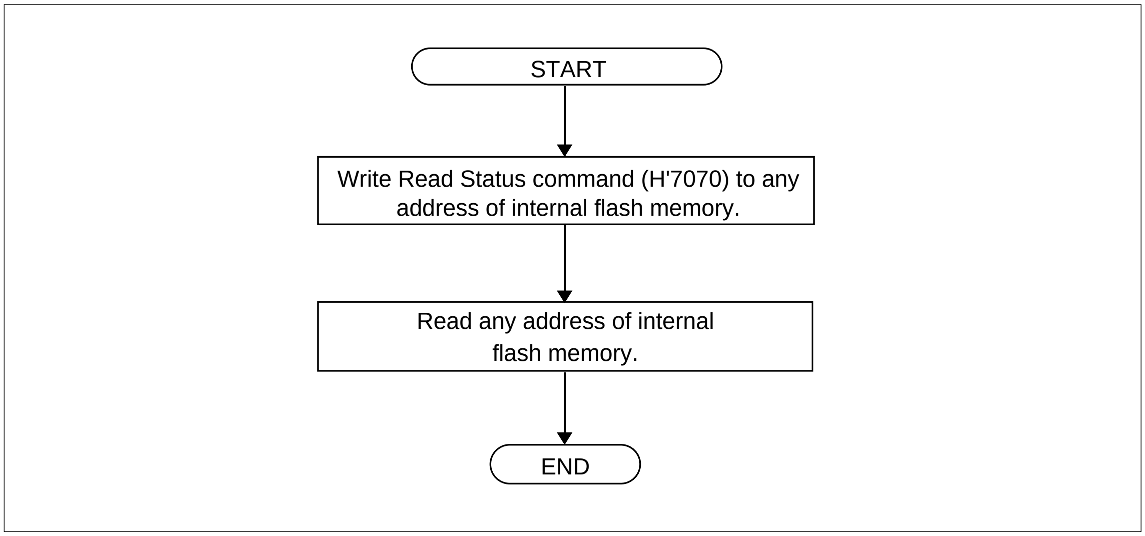

START
Write Read Status command (H'7070) to any address of internal flash memory.
Read any address of internal flash memory.
END

**Figure 6.5.15 Read Status Register**

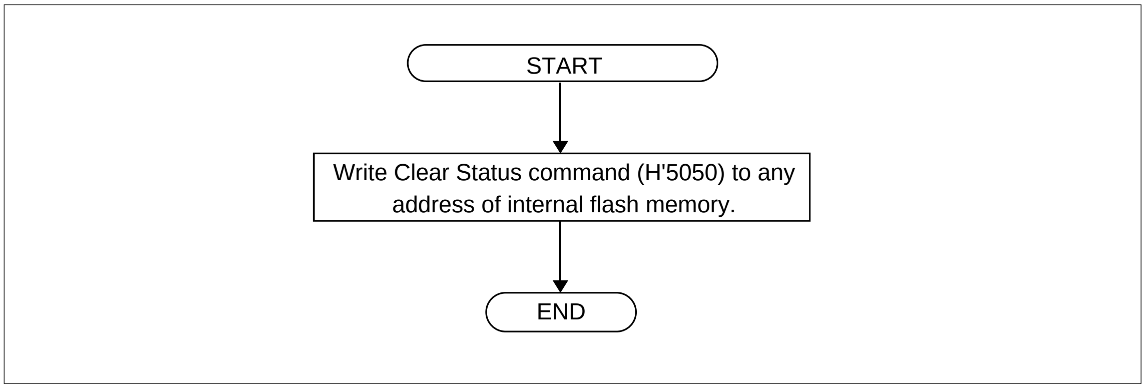

START
Write Clear Status command (H'5050) to any address of internal flash memory.
END

**Figure 6.5.16 Clear Status Register**

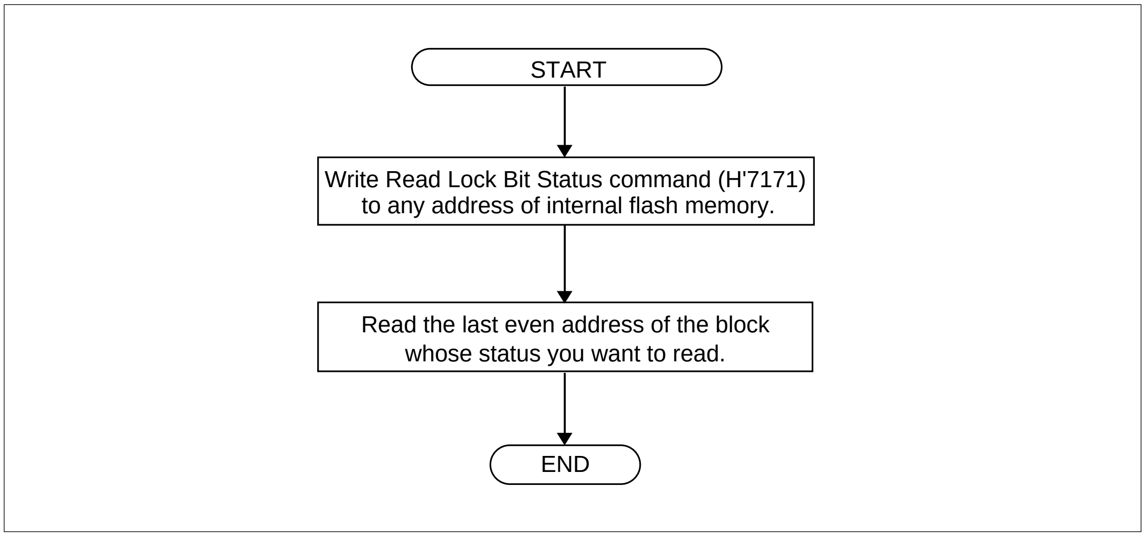

START
Write Read Lock Bit Status command (H'7171) to any address of internal flash memory.
Read the last even address of the block whose status you want to read.
END

**Figure 6.5.17 Read Lock Bit Status Register**

### 6.5.4 Flash Program Time (for Reference)

The time required for programming to the internal flash memory is shown below for your reference.

**(1) M32171F4**

a) Transfer time by SIO (for a transfer data size of 512 KB)

1/57600 bps × 1 (frame) × 11 (number of transfer bits) × 512 KB ≒ 100.1 [s]

b) Flash program time

512 KB/256-byte block × 8 ms ≒ 16.4 [s]

c) Erase time (entire area)

50 ms × number of blocks ≒ 550 [ms]

d) Total flash program time (entire 512 KB area)

- When communicating at 57600 bps using UART, the flash program time can be ignored because it is very short compared to the serial communication time. Therefore, the flash program time can be calculated using the equation below:

  a + c ≒ 101 [s]

  When programming data to flash memory at high speed by speeding up the serial communication or by other means, the fastest program time possible is as follows:

  b + c ≒ 17 [s]

**(2) M32171F3**

a) Transfer time by SIO (for a transfer data size of 384 KB)

1/57600 bps × 1 (frame) × 11 (number of transfer bits) × 384 KB ≒ 75.1 [s]

b) Flash program time

384 KB/256-byte block × 8 ms ≒ 12.3 [s]

c) Erase time (entire area)

50 ms × number of blocks ≒ 450 [ms]

d) Total flash program time (entire 384 KB area)

- When communicating at 57600 bps using UART, the flash program time can be ignored because it is very short compared to the serial communication time. Therefore, the flash program time can be calculated using the equation below:

  a + c ≒ 76 [s]

  When programming data to flash memory at high speed by speeding up the serial communication or by other means, the fastest program time possible is as follows:

  b + c ≒ 13 [s]

**(3) M32171F2**

a) Transfer time by SIO (for a transfer data size of 256 KB)

1/57600 bps ¥ 1 (frame) ¥ 11 (number of transfer bits) ¥ 256 KB ≒ 50.1 [s]

b) Flash program time

256 KB/256-byte block ¥ 8 ms ≒ 8.2 [s]

c) Erase time (entire area)

50 ms ¥ number of blocks ≒ 350 [ms]

d) Total flash program time (entire 256 KB area)

- When communicating at 57600 bps using UART, the flash program time can be ignored because it is very short compared to the serial communication time. Therefore, the flash program time can be calculated using the equation below:

  a + c ≒ 50.5 [s]

  When programming data to flash memory at high speed by speeding up the serial communication or by other means, the fastest program time possible is as follows:

  b + c ≒ 8.6 [s]

## 6.6 Boot ROM

The table below shows boot memory specifications of the 32171.

Table 6.6.1 Boot Memory Specifications

| Item | Specification |
|---|---|
| Capacity | 8 Kbytes |
| Location address | H'8000 0000 - H'8000 1FFF |
| Wait insertion | Operates with no wait states (with 40 MHz internal CPU memory clock) |
| Internal bus connection | Connected by 32-bit bus |
| Read | Can only be read when FP = 1, MOD0 = 1, and MOD1 = 0. When read in other modes, indeterminate values are read out. Cannot be accessed for write. |
| Other | Because the boot ROM area is a reserved area that can only be used in boot mode, the program cannot be modified. |

## 6.7 Virtual-Flash Emulation Function

The 32171 can map one 8-Kbyte block of internal RAM beginning with the start address into one of 8-Kbyte areas (L banks) of the internal flash memory and can map up to two 4-Kbyte blocks of internal RAM beginning with address H’0080 6000 into one of 4-Kbyte areas (S banks) of the internal flash memory. This capability is referred to as the “virtual-flash emulation” function.

This function allows the data located in an 8-Kbyte block or one or two 4-Kbyte blocks of the internal RAM to be switched for use to or from the L or S bank of flash memory specified by the Virtual-Flash Bank Register. Therefore, applications that require changes of data during program operation can have data dynamically changed using 8 or 4 Kbytes of RAM area. The RAM used for virtual-flash emulation can be accessed for read and write from both the internal RAM and the internal flash memory areas.

When this function is used in combination with the internal Real Time Debugger (RTD), the data tables created in the internal flash memory can be referenced or rewritten from outside, thus facilitating data table tuning.

**Before programming to the internal flash memory, always be sure to terminate this virtual-flash emulation mode.**

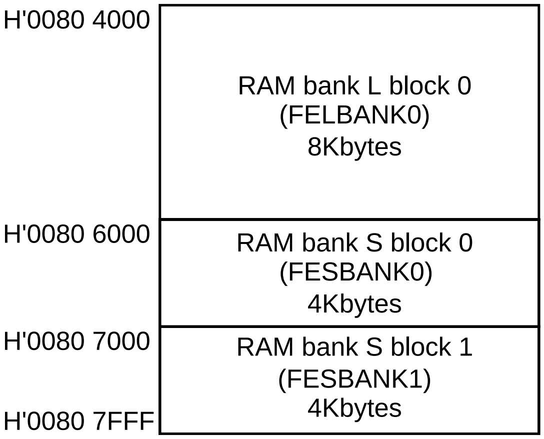
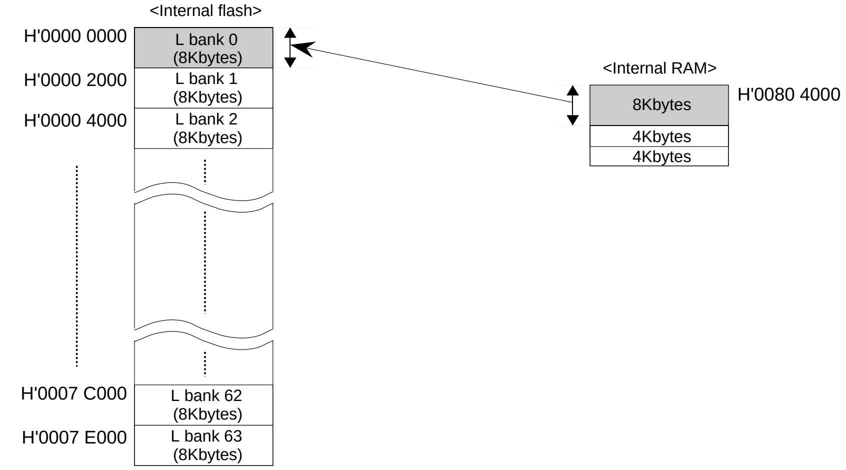


**Figure 6.7.1 Internal RAM Bank Configuration of the 32171**

### 6.7.1 Virtual-Flash Emulation Areas

The following shows the areas effective for the virtual-flash emulation function.
Select one of 8-Kbyte blocks or L banks of flash memory using the Virtual-Flash L Bank Register (FELBANK0) (by setting the seven address bits A12–A18 of the start address of the desired L bank in the Virtual-Flash L Bank Register LBANKAD bits). Then set the Virtual-Flash L Bank Register MODENL bit (MODENL0 bit) to 1. The selected L bank area can be rewritten with the 8-Kbyte content of the internal RAM beginning with its start address.
Also, select one or two of 4-Kbyte blocks or S banks of flash memory using the Virtual-Flash S Bank Registers (FESBANK0 and FESBANK1) (by setting the eight address bits A12–A19 of the start address of each desired S bank in the Virtual-Flash S Bank Register SBANKAD bits). Then set the Virtual-Flash S Bank Register MODENS0 and MODENS1 bits to 1. The selected S bank areas can be replaced with 4 Kbytes of the internal RAM, for up to two blocks, beginning with the address H’0080 6000.
In this way, one 8-Kbyte block or L bank and two 4-Kbyte blocks or S banks for up to a total of three banks can be selected.

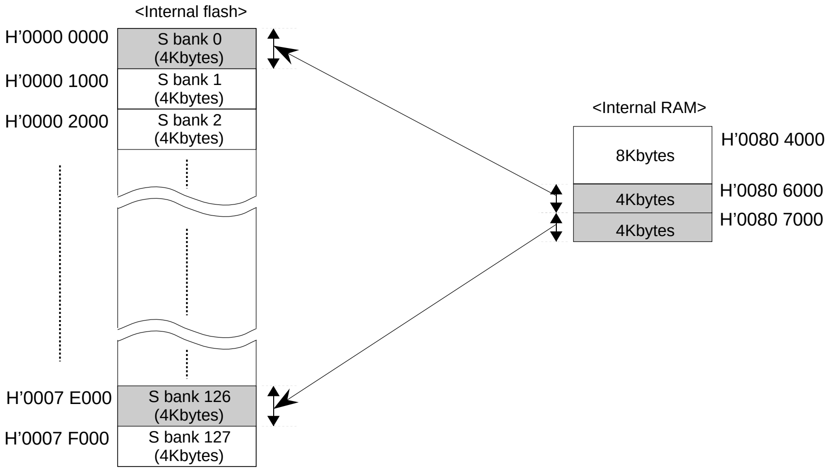
Notes:
- If the virtual-flash emulation enable bit is enabled after setting the same bank area in multiple virtual-flash bank registers, the corresponding internal RAM area (8 or 4 Kbytes) is allocated in order of priority FELBANK0 > FESBANK0 > FESBANK1.
- During virtual-flash emulation mode, RAM can be accessed for read and write from the internal RAM area and virtual-flash setup area.
- When performing virtual-flash read after setting Flash Control Register 1's Virtual-Flash Emmulation Mode bit to 1, be sure to wait for three CPU clock periods or more before performing virtual-flash read after setting the said bit to 1.
- Before performing virtual-flash read after setting the Virtual-flash Bank Register(L Bank and S Bank Registers)’s virtual-flash emulation enable and bank address bits, be sure to insert wait states equal to or greater than three CPU clock periods**.**



Notes: • If the Virtual-Flash Emulation Enable bit is enabled while the same bank area is set in multiple Virtual-Flash Bank Registers, the internal RAM area to be allocated is selected by priority: FELBANK0 > FESBANK0 > FESBANK 1.

• When you access the 8-Kbyte area (L bank) selected by Virtual-Flash L Bank Register 0, you actually are accessing the internal RAM area. During virtual-flash emulation mode, the RAM can be accessed for read and write from both the internal RAM and the selected virtual-flash memory areas.

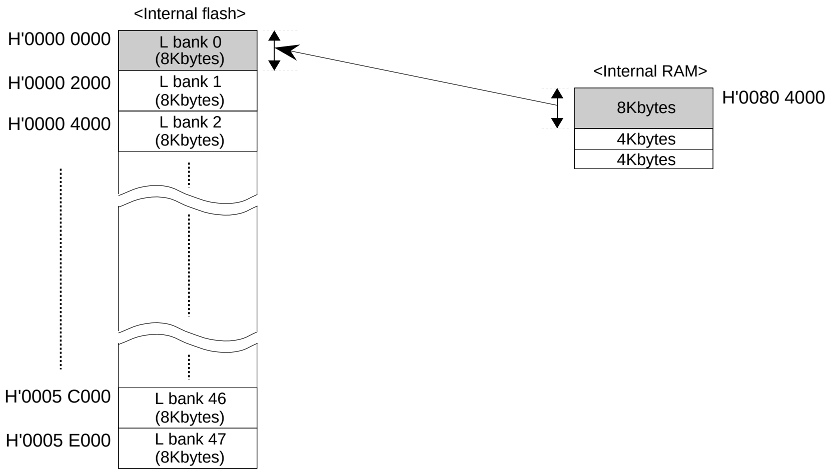
**Figure 6.7.2 Virtual-Flash Emulation Areas of the M32171F4 Divided in Units of 8 Kbytes**



Notes: • If the Virtual-Flash Emulation Enable bit is enabled while the same bank area is set in multiple Virtual-Flash Bank Registers, the internal RAM area (8 or 4 Kbytes) to be allocated is selected by priority: FELBANK0 > FESBANK0 > FESBANK 1.

• When you access the 4-Kbyte area (S bank) selected by Virtual-Flash S Bank Register 0,1, you actually are accessing the internal RAM area. During virtual-flash emulation mode, the RAM can be accessed for read and write from both the internal RAM and the selected virtual-flash memory areas.

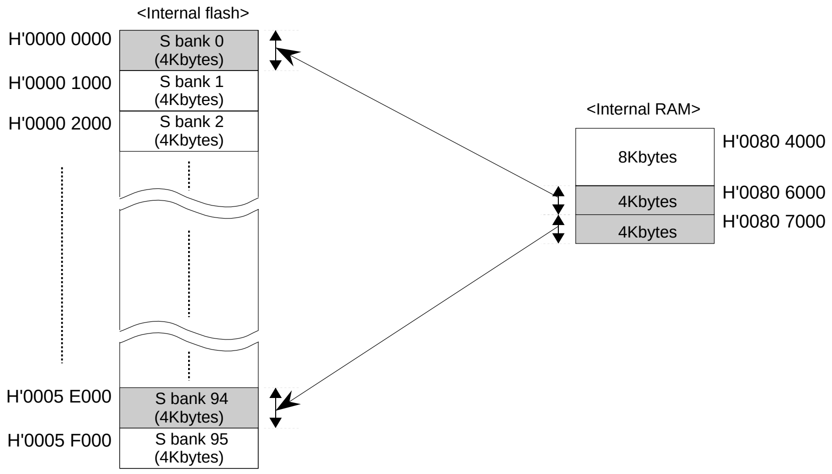
**Figure 6.7.3 Virtual-Flash Emulation Areas of the M32171F4 Divided in Units of 4 Kbytes**



Notes: • If the Virtual-Flash Emulation Enable bit is enabled while the same bank area is set in multiple Virtual-Flash Bank Registers, the internal RAM area (8 or 4 Kbytes) to be allocated is selected by priority: FELBANK0 > FESBANK0 > FESBANK 1.

• When you access the 8-Kbyte area (L bank) selected by Virtual-Flash L Bank Register 0, you actually are accessing the internal RAM area. During virtual-flash emulation mode, the RAM can be accessed for read and write from both the internal RAM and the selected virtual-flash memory areas.

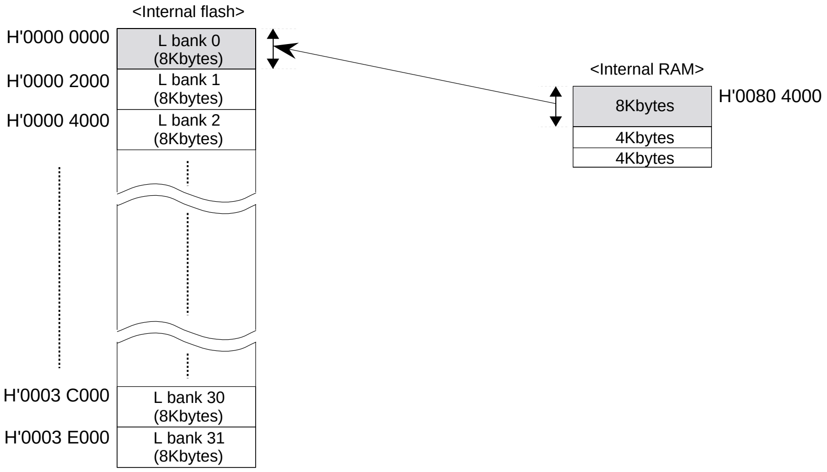
**Figure 6.7.4 Virtual-Flash Emulation Areas of the M32171F3 Divided in Units of 8 Kbytes**



Notea: • If the Virtual-Flash Emulation Enable bit is enabled while the same bank area is set in multiple Virtual-Flash Bank Registers, the internal RAM area (8 or 4 Kbytes) to be allocated is selected by priority: FELBANK0 > FESBANK0 > FESBANK 1.

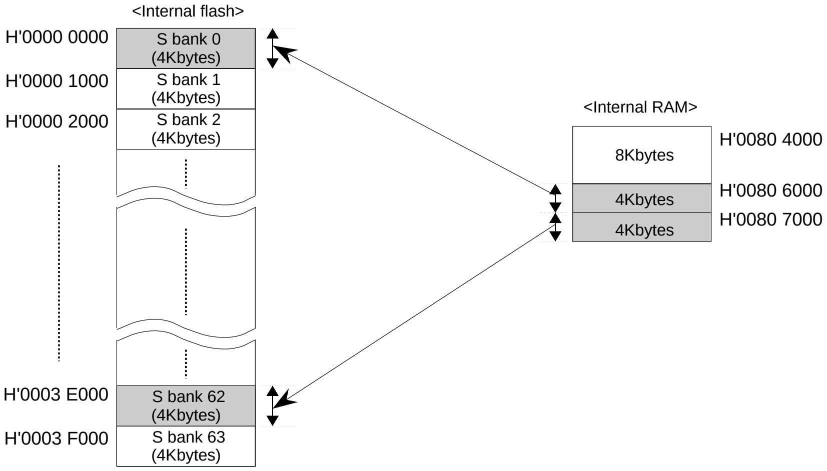
• When you access the 4-Kbyte area (S bank) selected by Virtual-Flash S Bank Register 0,1, you actually are accessing the internal RAM area. During virtual-flash emulation mode, the RAM can be accessed for read and write from both the internal RAM and the selected virtual-flash memory areas.

**Figure 6.7.5 Virtual-Flash Emulation Areas of the M32171F3 Divided in Units of 4 Kbytes**



Notes:
- If the Virtual-Flash Emulation Enable bit is enabled while the same bank area is set in multiple Virtual-Flash Bank Registers, the internal RAM area (8 or 4 Kbytes) to be allocated is selected by priority: FELBANK0 > FESBANK0 > FESBANK 1.
- When you access the 8-Kbyte area (L bank) selected by Virtual-Flash L Bank Register 0, you actually are accessing the internal RAM area. During virtual-flash emulation mode, the RAM can be accessed for read and write from both the internal RAM and the selected virtual-flash memory areas.

Figure 6.7.6 Virtual-Flash Emulation Areas of the M32171F2 Divided in Units of 8 Kbytes



Notes:
- If the Virtual-Flash Emulation Enable bit is enabled while the same bank area is set in multiple Virtual-Flash Bank Registers, the internal RAM area (8 or 4 Kbytes) to be allocated is selected by priority: FELBANK0 > FESBANK0 > FESBANK 1.
- When you access the 4-Kbyte area (S bank) selected by Virtual-Flash S Bank Register 0, 1, you actually are accessing the internal RAM area. During virtual-flash emulation mode, the RAM can be accessed for read and write from both the internal RAM and the selected virtual-flash memory areas.

Figure 6.7.7 Virtual-Flash Emulation Areas of the M32171F2 Divided in Units of 4 Kbytes

| L bank | Start address of bank in flash memory | L bank address (LBANKAD) bit set value |
|---|---|---|
| L bank 0 | H'0000 0000<br>(Note 1) | H'00 |
| L bank 1 | H'0000 2000 | H'02 |
| L bank 2 | H'0000 4000 | H'04 |
|  | ⋮ |  |
| L bank 62 | H'0007 C000 | H'7C |
| L bank 63 | H'0007 E000 | H'7E |

Note 1: Set the seven bits A12-A18 of the start address (32-bit) of each L bank of flash memory divided every 8 Kbytes in the Virtual Flash L Bank Register's L bank address (LBANKAD) bits.

**Figure 6.7.8 Values Set in the M32171F4's Virtual Flash Bank Register when Divided in Units of 8 Kbytes**

| S bank | Start address of bank in flash memory | S bank address (SBANKAD) bit set value |
|---|---|---|
| S bank 0 | H'0000 0000<br>(Note 1) | H'00 |
| S bank 1 | H'0000 1000 | H'01 |
| S bank 2 | H'0000 2000 | H'02 |
|  | ⋮ |  |
| S bank 126 | H'0007 E000 | H'7E |
| S bank 127 | H'0007 F000 | H'7F |

Note 1: Set the eight bits A12-A19 of the start address (32-bit) of each S bank of flash memory divided every 4 Kbytes in the Virtual Flash S Bank Register's S bank address (SBANKAD) bits.

**Figure 6.7.9 Values Set in the M32171F4's Virtual Flash Bank Register when Divided in Units of 4 Kbytes**

| L bank | Start address of bank in flash memory | L bank address (LBANKAD) bit set value |
|---|---|---|
| L bank 0 | H'0000 0000<br>(Note 1) | H'00 |
| L bank 1 | H'0000 2000 | H'02 |
| L bank 2 | H'0000 4000 | H'04 |
| ⋮ | ⋮ | ⋮ |
| L bank 46 | H'0005 C000 | H'5C |
| L bank 47 | H'0005 E000 | H'5E |

Note 1: Set the seven bits A12-A18 of the start address (32-bit) of each L bank of flash memory divided every 8 Kbytes in the Virtual Flash L Bank Register's L bank address (LBANKAD) bits.

**Figure 6.7.10 Values Set in the M32171F3's Virtual Flash Bank Register when Divided in Units of 8 Kbytes**

| S bank | Start address of bank in flash memory | S bank address (SBANKAD) bit set value |
|---|---|---|
| S bank 0 | H'0000 0000<br>(Note 1) | H'00 |
| S bank 1 | H'0000 1000 | H'01 |
| S bank 2 | H'0000 2000 | H'02 |
| ⋮ | ⋮ | ⋮ |
| S bank 94 | H'0005 E000 | H'5E |
| S bank 95 | H'0005 F000 | H'5F |

Note 1: Set the eight bits A12-A19 of the start address (32-bit) of each S bank of flash memory divided every 4 Kbytes in the Virtual Flash S Bank Register's S bank address (SBANKAD) bits.

**Figure 6.7.11 Values Set in the M32171F3's Virtual Flash Bank Register when Divided in Units of 4 Kbytes**

| L bank | Start address of bank in flash memory | L bank address (LBANKAD) bit set value |
|---|---|---|
| L bank 0 | H'0000 0000<br>(Note 1) | H'00 |
| L bank 1 | H'0000 2000 | H'02 |
| L bank 2 | H'0000 4000 | H'04 |
|  | ⋮ |  |
| L bank 30 | H'0003 C000 | H'3C |
| L bank 31 | H'0003 E000 | H'3E |

Note 1: Set the seven bits A12-A18 of the start address (32-bit) of each L bank of flash memory divided every 8 Kbytes in the Virtual Flash L Bank Register's L bank address (LBANKAD) bits.

**Figure 6.7.12 Values Set in the M32171F2's Virtual Flash Bank Register when Divided in Units of 8 Kbytes**

| S bank | Start address of bank in flash memory | S bank address (SBANKAD) bit set value |
|---|---|---|
| S bank 0 | H'0000 0000<br>(Note 1) | H'00 |
| S bank 1 | H'0000 1000 | H'01 |
| S bank 2 | H'0000 2000 | H'02 |
|  | ⋮ |  |
| S bank 62 | H'0003 E000 | H'3E |
| S bank 63 | H'0003 F000 | H'3F |

Note 1: Set the eight bits A12-A19 of the start address (32-bit) of each S bank of flash memory divided every 4 Kbytes in the Virtual Flash S Bank Register's S bank address (SBANKAD) bits.

**Figure 6.7.13 Values Set in the M32171F2's Virtual Flash Bank Register when Divided in Units of 4 Kbytes**

### 6.7.2 Entering Virtual Flash Emulation Mode

To enter Virtual Flash Emulation Mode, set the Flash Control Register 1 (FCNT1) FEMMOD bit to 1. After entering Virtual Flash Emulation Mode, set the Virtual Flash Bank Register MODEN bit to 1 to enable the Virtual Flash Emulation Function.

Even during virtual-flash emulation mode, the internal RAM area (H’0080 4000 through H’0080 7FFF) can be accessed as internal RAM.

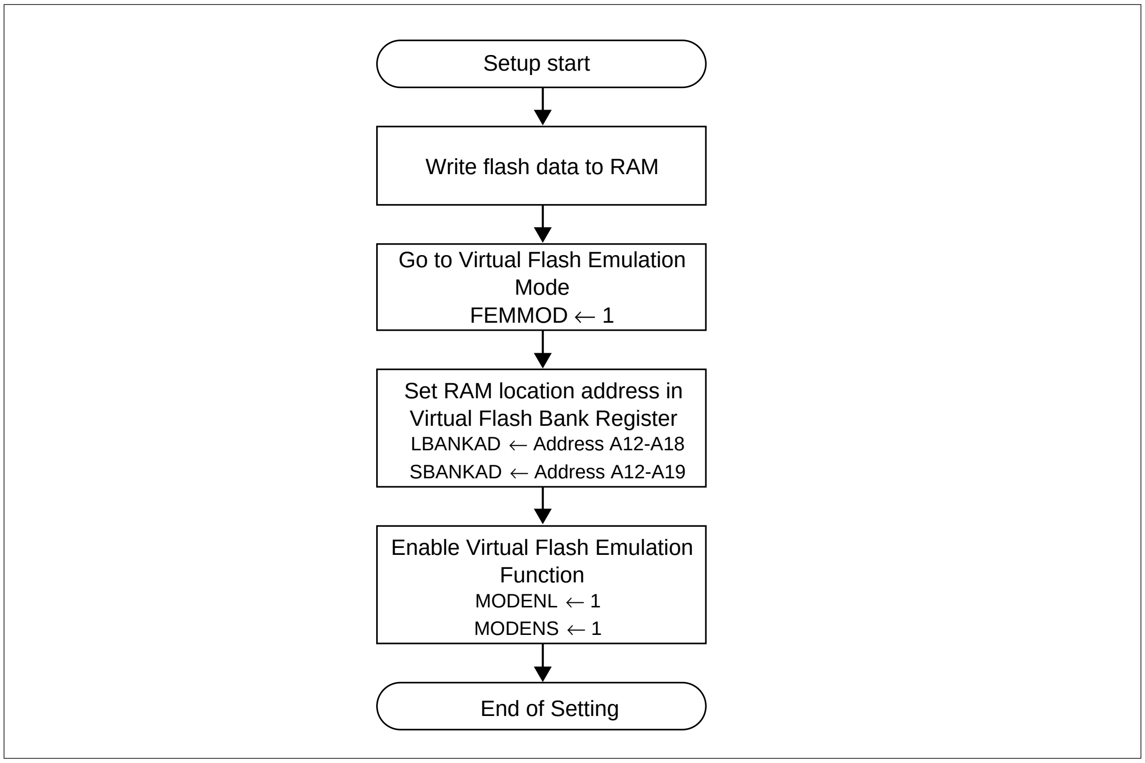

Setup start
Write flash data to RAM
Go to Virtual Flash Emulation Mode
FEMMOD ← 1
Set RAM location address in Virtual Flash Bank Register
LBANKAD ← Address A12-A18
SBANKAD ← Address A12-A19
Enable Virtual Flash Emulation Function
MODENL ← 1
MODENS ← 1
End of Setting

**Figure 6.7.14 Virtual-flash Emulation Mode Sequence**

### 6.7.3 Application Example of Virtual Flash Emulation Mode

By locating two RAM areas in the same virtual flash area using the Virtual Flash Emulation Function, you can rewrite data in the flash memory successively.

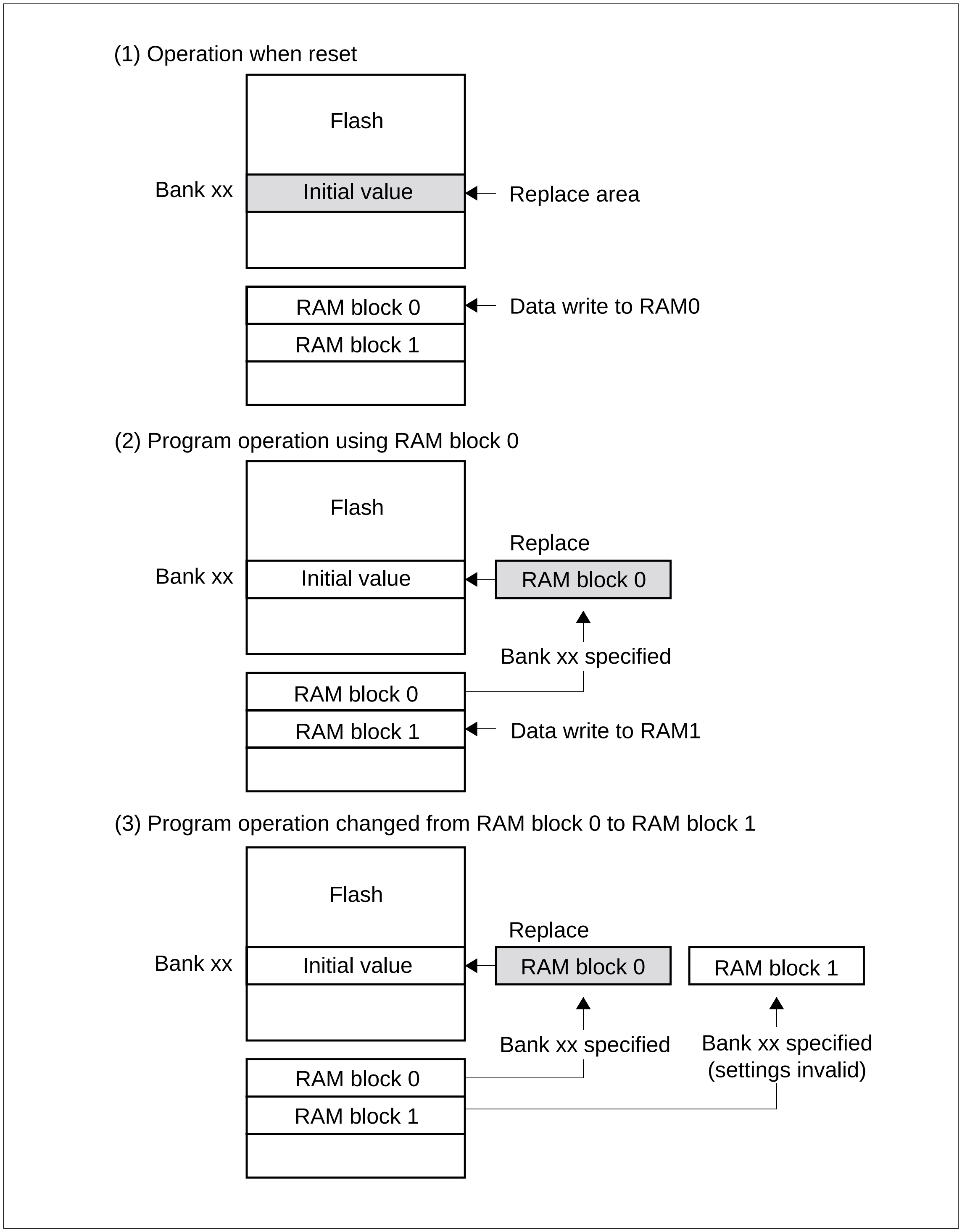


**Figure 6.7.15 Application Example of Virtual Flash Emulation (1/2)**

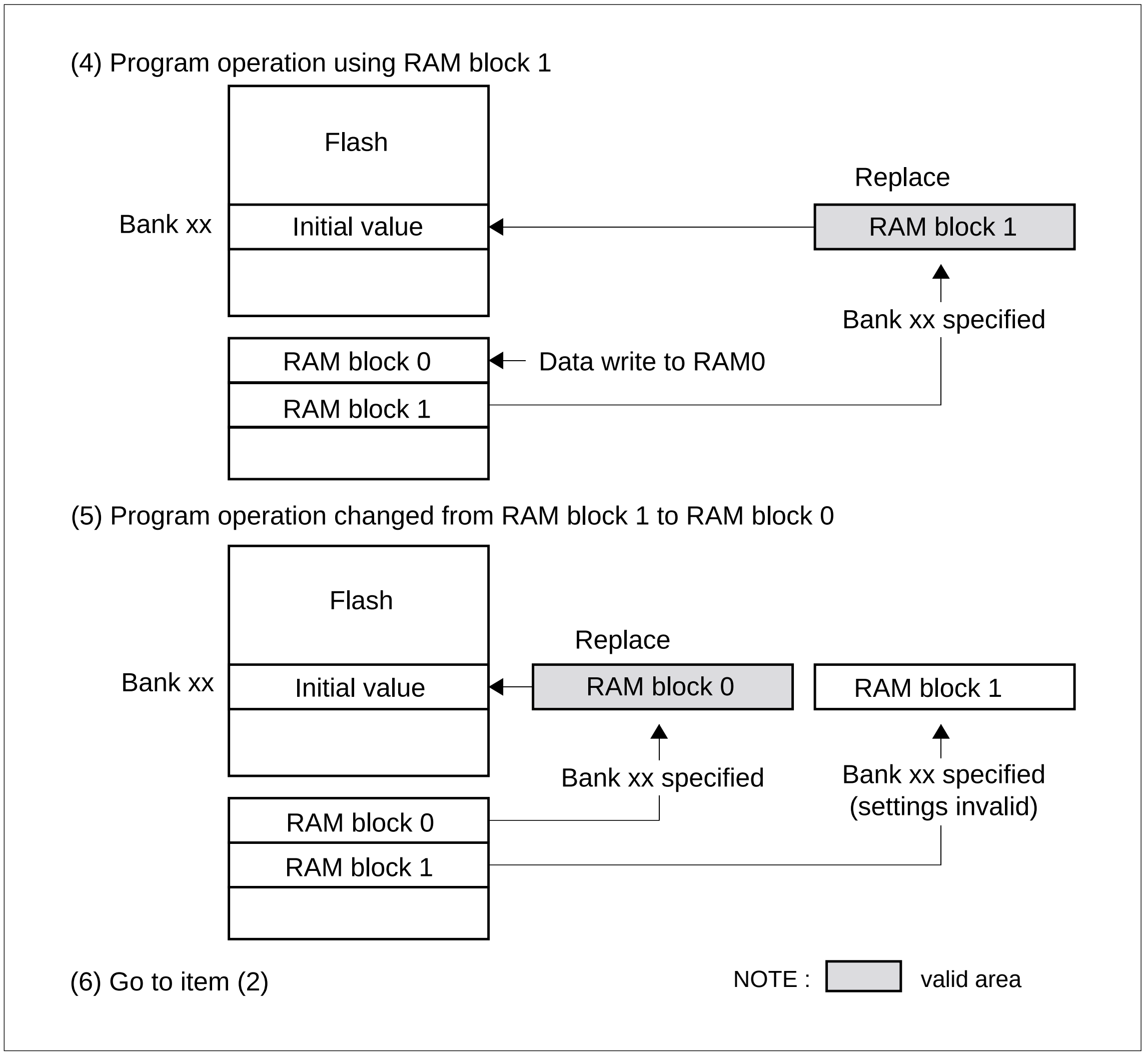


**Figure 6.7.16 Application Example of Virtual Flash Emulation (2/2)**

## 6.8 Connecting to A Serial Programmer

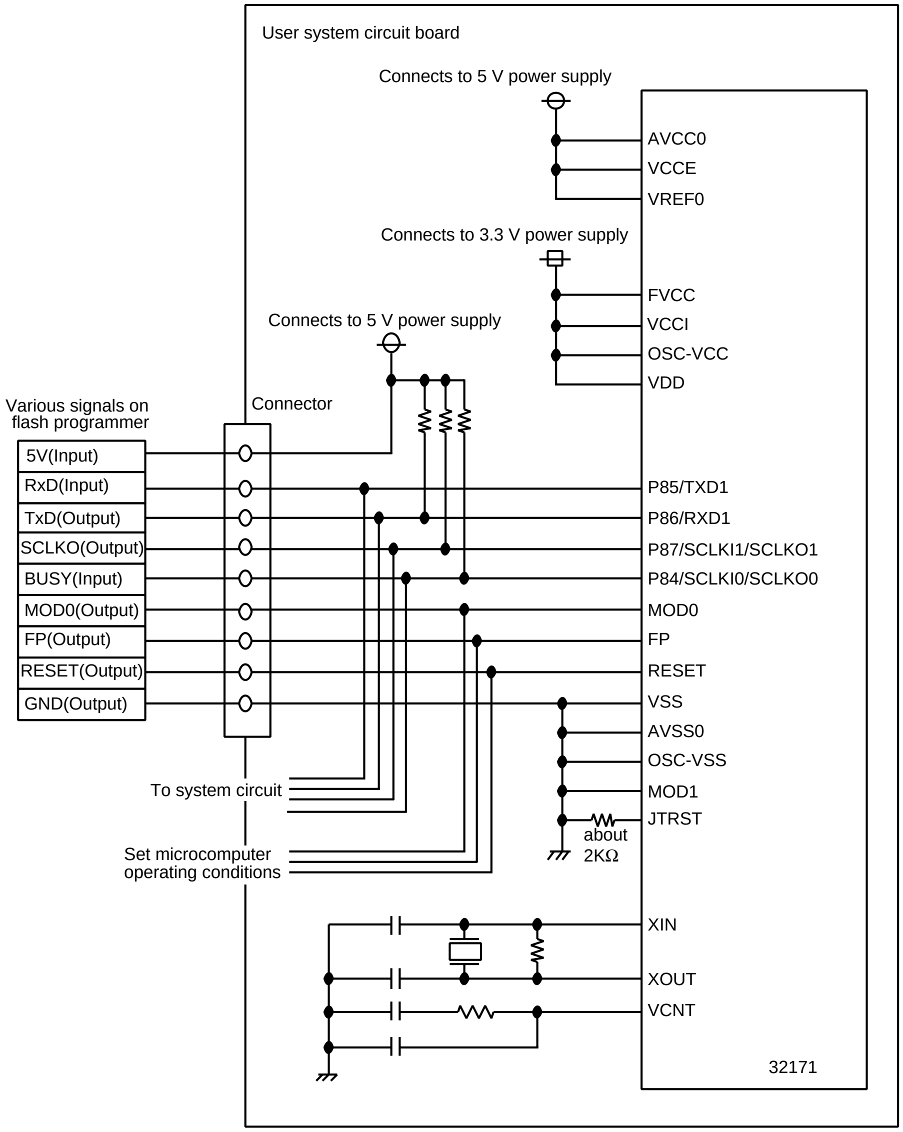
When you reprogram the internal flash memory using a general-purpose serial programmer in Boot Flash E/W Enable mode, you need to process the pins on the 32171 shown below to make them suitable for the serial programmer.

**Table 6.8.1 Processing the 32171 Pins when Using a Serial Programmer**

| Pin Name | Pin Number | Function | Remark |
|---|---|---|---|
| SCLKI1 | 71 | Transfer clock input | Need to be pulled high |
| RXD1 | 70 | Serial data input<br>(receive data) | Need to be pulled high |
| TXD1 | 69 | Serial data output<br>(transmit data) | |
| P84 | 68 | Transmit/receive enable output | Need to be pulled high |
| FP | 94 | Flash memory protect | |
| MOD0 | 92 | Operation mode 0 | |
| MOD1 | 93 | Operation mode 1 | Connect to ground |
| $\overline{\text{RESET}}$ | 91 | Reset | |
| XIN | 4 | Clock input | |
| XOUT | 5 | Clock output | |
| VCNT | 7 | PLL circuit control input | |
| OSC-VCC | 6 | PLL circuit power supply | Connect to 3.3 V power supply |
| OSC-VSS | 3 | PLL circuit ground | Connect to ground |
| VREF0 | 42 | A-D converter reference voltage input | Connect to 5 V power supply |
| AVCC0 | 43 | Analog power supply | Connect to 5 V power supply |
| AVSS0 | 60 | Analog ground | Connect to ground |
| FVCC | 73 | Flash memory power supply | Connect to 3.3 V power supply |
| VDD | 108 | RAM backup power supply | Connect to 3.3 V power supply |
| VCCE | 20, 65, 95, 132 | 5 V power supply | |
| VCCI | 61, 123, 137 | 3.3 V power supply | |
| VSS | 21, 62, 72, 96, 138 | Ground | |

Note: • All other pins do not need to be processed.

The diagram below shows an example of user system configuration which has had a serial programmer connected. After the user system is powered on, the serial programmer programs to the flash memory in clock-synchronized serial mode. No communication problems associated with the oscillation frequency may occur. If the system uses any 32171 pins which will connect to a serial programmer, care must be taken to prevent adverse effects on the system when a serial programmer is connected. Note that the serial programmer uses the addresses H'0000 0084 through H'0000 0093 as an area to check ID for flash memory protection.



Notes: • Turn on the power to the user system before you program to the flash memory.
- If the system circuit uses P84-P87, consideration must be taken for connection of a serial programmer.
- P64/SBI must be fixed high or low to ensure that interrupts will not be generated.
- The pullup resistances of P84, P86, and P87 must be set to suit system design conditions.
- The typical pullup resistances of P84, P86, and P87 are 4.7 to 10 kΩ.
- All other ports, whether high or low, do not affect flash memory programming.

**Figure 6.8.1 Pin Connection Diagram**

## 6.9 Internal Flash Memory Protect Functions

The 32171's internal flash memory has the following four protect functions to prevent unintended reprogramming by an erratic operation or unauthorized copying or reprogramming of its contents.

### (1) Flash memory protect ID

When using flash memory reprogramming tools such as a general-purpose serial programmer or an emulator, the ID entered from the keyboard is checked against the flash memory's internal ID. In no case can reprogramming be executed unless the correct ID is entered. (For some tools, erasing of the entire area only can be executed.)

### (2) Protection by FP pin

The flash memory is protected in hardware against E/W by pulling the FP (Flash Protect) pin low. Furthermore, because the FP pin level can be known by reading the Flash Mode Register (FMOD)'s FPMOD (external FP pin status) bit in a flash write program, the flash memory can also be protected in software. For systems that do not require protection by external pin settings, holding the FP pin high will help to simplify operation while reprogramming the flash memory.

### (3) Protection by FENTRY bit

Flash E/W enable mode cannot be entered unless Flash Control Register 1 (FCNT1)'s FENTRY (flash mode entry) bit is set to 1. Furthermore, the FENTRY bit can only be set to 1 by writing 0 and 1 in succession while the FP pin is high.

### (4) Protection by a lock bit

Each block of flash memory has a lock bit, so that any memory block can be protected against E/W by setting this bit to 0.

## 6.10 Precautions to Be Taken When Reprogramming Flash Memory

The following describes precautions to be taken when you reprogram the flash memory using a general-purpose serial programmer in Boot Flash E/W Enable mode.

- When reprogramming the flash memory, a high voltage is generated inside the chip. Because this high voltage could cause the chip to break down, be careful about mode pin and power supply management not to move from one mode to another while reprogramming.
- If the system uses any pin that is to be used by a general-purpose reprogramming tool, take appropriate measures to prevent adverse effects when connecting the tool.
- If flash memory protection is needed when using a general-purpose reprogramming tool, set any ID in the flash memory protect ID check area (H’0000 0084–H’0000 0093).
- If flash memory protection is not needed when using a general-purpose reprogramming tool, set H’FF in the entire flash memory protect ID check area (H’0000 0084–H’0000 0093).
- Before using a reset by Flash Control Register 4 (FCNT4)’s FRESET bit to clear each error status in Flash Status Register 2 (FSTAT2) (initialized to H’80), check to see that Flash Status Register 1 (FSTAT1)’s FSTAT bit = 1 (Ready).
- Before changing Flash Control Register 1 (FCNT1)’s FENTRY bit from 1 to 0, check to see that Flash Status Register 1 (FSTAT1)’s FSTAT bit = 1 (Ready) or Flash Status Register 2 (FSTAT2)’s FBUSY bit = 1 (Ready).
- If Flash Control Register 1 (FCNT1)’s FENTRY bit = 1 and Flash Status Register 1 (FSTAT1)’s FSTAT bit = 0 (Busy) or Flash Status Register 2 (FSTAT2)’s FBUSY bit = 0 (program/erase in progress), do not clear the FENTRY bit.

# CHAPTER 7
# RESET

7.1 Outline of Reset
7.2 Reset Operation
7.3 Internal State after Exiting Reset
7.4 Things To Be Considered after Exiting Reset

## 7.1 Outline of Reset

The device is reset by applying a low-level signal to the $\overline{\text{RESET}}$ input pin. The device is gotten out of a reset state by releasing the $\overline{\text{RESET}}$ input back high, upon which the reset vector entry address is set in the Program Counter (PC) and the program starts executing from the reset vector entry.

## 7.2 Reset Operation

### 7.2.1 Reset at Power-on

When powering on the device, hold the $\overline{\text{RESET}}$ input low until its internal multiply-by-4 clock generator becomes oscillating stably.

### 7.2.2 Reset during Operation

To reset the device during operation, hold the $\overline{\text{RESET}}$ input low for more than four clock periods of XIN signal.

### 7.2.3 Reset Vector Relocation during Flash Reprogramming

When placed in boot mode, the reset vector entry address is moved to the start address of the boot program space (address H'8000 0000). For details, refer to Section 6.5, "Programming of Internal Flash Memory."

## 7.3 Internal State after Exiting Reset

The table below lists the register state of the device after it has gotten out of reset. For details about the initial register state of each internal peripheral I/O, refer to each section in this manual where the relevant internal peripheral I/O is described.

Table 7.3.1 Internal State after Exiting Reset

| Register | State after Exiting Reset |
|---|---|
| PSW (CR0) | B'0000 0000 0000 0000 ??00 000? 0000 0000 (BSM, BIE, BC bits = indeterminate) |
| CBR (CR1) | H'0000 0000 (C bit = 0) |
| SPI (CR2) | Indeterminate |
| SPU (CR3) | Indeterminate |
| BPC (CR6) | Indeterminate |
| PC | H'0000 0000 (Executed beginning with address H'0000 0000) (Note 1) |
| R0–R15 | Indeterminate |
| ACC (accumulator) | Indeterminate |
| RAM | Indeterminate at power-on reset (However, if the device is reset and placed out of reset while the VDD pin has 2.0 V to 3.6 V being applied to it, the RAM content before a reset is retained.) |

Note 1: When in boot mode, this changes to the start address of the boot program space (H'8000 0000).

The pins that were set for input when reset go to a high-impedance state (Hi-Z). Here, “when reset” means that the RESET# pin input is held low (the device being reset) and is released back high (the device being placed out of reset).

### Table 7.3.2 Pin Status When Reset (1/4)

| PIN NO | Pin Name | Function | | | Input/output | Condition | Pin status when reset | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Port | Other than port | Other than port | | | function | Input/output | Status during reset | Status after exiting reset |
| 1 | P221/CRX (Note 1) | P221 | CRX | - | Input | | P221 | Input | Hi-z | Hi-z |
| 2 | P225/A12 | P225 | A12 | - | Input/output | During single-chip mode | P225 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A12 | Output | Hi-z | Indeterminate |
| 3 | OSC-VSS | - | OSC-VSS | - | - | | OSC-VSS | - | - | - |
| 4 | XIN | - | XIN | - | Input | | XIN | Input | - | - |
| 5 | XOUT | - | XOUT | - | Output | | XOUT | Output | XOUT | XOUT |
| 6 | OSC-VCC | - | OSC-VCC | - | - | | OSC-VCC | - | - | - |
| 7 | VCNT | - | VCNT | - | - | | VCNT | - | - | - |
| 8 | P30/A15 | P30 | A15 | - | Input/output | During single-chip mode | P30 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A15 | Output | Hi-z | Indeterminate |
| 9 | P31/A16 | P31 | A16 | - | Input/output | During single-chip mode | P31 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A16 | Output | Hi-z | Indeterminate |
| 10 | P32/A17 | P32 | A17 | - | Input/output | During single-chip mode | P32 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A17 | Output | Hi-z | Indeterminate |
| 11 | P33/A18 | P33 | A18 | - | Input/output | During single-chip mode | P33 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A18 | Output | Hi-z | Indeterminate |
| 12 | P34/A19 | P34 | A19 | - | Input/output | During single-chip mode | P34 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A19 | Output | Hi-z | Indeterminate |
| 13 | P35/A20 | P35 | A20 | - | Input/output | During single-chip mode | P35 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A20 | Output | Hi-z | Indeterminate |
| 14 | P36/A21 | P36 | A21 | - | Input/output | During single-chip mode | P36 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A21 | Output | Hi-z | Indeterminate |
| 15 | P37/A22 | P37 | A22 | - | Input/output | During single-chip mode | P37 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A22 | Output | Hi-z | Indeterminate |
| 16 | P20/A23 | P20 | A23 | - | Input/output | During single-chip mode | P20 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A23 | Output | Hi-z | Indeterminate |
| 17 | P21/A24 | P21 | A24 | - | Input/output | During single-chip mode | P21 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A24 | Output | Hi-z | Indeterminate |
| 18 | P22/A25 | P22 | A25 | - | Input/output | During single-chip mode | P22 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A25 | Output | Hi-z | Indeterminate |
| 19 | P23/A26 | P23 | A26 | - | Input/output | During single-chip mode | P23 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A26 | Output | Hi-z | Indeterminate |
| 20 | VCCE | - | VCCE | - | - | | VCCE | - | - | - |
| 21 | VSS | - | VSS | - | - | | VSS | - | - | - |
| 22 | P24/A27 | P24 | A27 | - | Input/output | During single-chip mode | P24 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A27 | Output | Hi-z | Indeterminate |
| 23 | P25/A28 | P25 | A28 | - | Input/output | During single-chip mode | P25 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A28 | Output | Hi-z | Indeterminate |
| 24 | P26/A29 | P26 | A29 | - | Input/output | During single-chip mode | P26 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A29 | Output | Hi-z | Indeterminate |
| 25 | P27/A30 | P27 | A30 | - | Input/output | During single-chip mode | P27 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A30 | Output | Hi-z | Indeterminate |
| 26 | P00/DB0 | P00 | DB0 | - | Input/output | During single-chip mode | P00 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB0 | Input | Hi-z | Hi-z |

Note 1: P221 is used exclusively for CAN input

### Table 7.3.3 Pin Status When Reset (2/4)

| Pin NO. | Pin Name | Function | | | Input/output | Condition | Pin status when reset | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Port | Other than port | Other than port | | | Function | Input/output | Status during reset | Status after exiting reset |
| 27 | P01/DB1 | P01 | DB1 | - | Input/output | During single-chip mode | P01 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB1 | Input | Hi-z | Hi-z |
| 28 | P02/DB2 | P02 | DB2 | - | Input/output | During single-chip mode | P02 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB2 | Input | Hi-z | Hi-z |
| 29 | P03/DB3 | P03 | DB3 | - | Input/output | During single-chip mode | P03 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB3 | Input | Hi-z | Hi-z |
| 30 | P04/DB4 | P04 | DB4 | - | Input/output | During single-chip mode | P04 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB4 | Input | Hi-z | Hi-z |
| 31 | P05/DB5 | P05 | DB5 | - | Input/output | During single-chip mode | P05 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB5 | Input | Hi-z | Hi-z |
| 32 | P06/DB6 | P06 | DB6 | - | Input/output | During single-chip mode | P06 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB6 | Input | Hi-z | Hi-z |
| 33 | P07/DB7 | P07 | DB7 | - | Input/output | During single-chip mode | P07 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB7 | Input | Hi-z | Hi-z |
| 34 | P10/DB8 | P10 | DB8 | - | Input/output | During single-chip mode | P10 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB8 | Input | Hi-z | Hi-z |
| 35 | P11/DB9 | P11 | DB9 | - | Input/output | During single-chip mode | P11 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB9 | Input | Hi-z | Hi-z |
| 36 | P12/DB10 | P12 | DB10 | - | Input/output | During single-chip mode | P12 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB10 | Input | Hi-z | Hi-z |
| 37 | P13/DB11 | P13 | DB11 | - | Input/output | During single-chip mode | P13 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB11 | Input | Hi-z | Hi-z |
| 38 | P14/DB12 | P14 | DB12 | - | Input/output | During single-chip mode | P14 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB12 | Input | Hi-z | Hi-z |
| 39 | P15/DB13 | P15 | DB13 | - | Input/output | During single-chip mode | P15 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB13 | Input | Hi-z | Hi-z |
| 40 | P16/DB14 | P16 | DB14 | - | Input/output | During single-chip mode | P16 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB14 | Input | Hi-z | Hi-z |
| 41 | P17/DB15 | P17 | DB15 | - | Input/output | During single-chip mode | P17 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | DB15 | Input | Hi-z | Hi-z |
| 42 | VREF0 | - | VREF0 | - | - | | VREF0 | - | - | - |
| 43 | AVCC0 | - | AVCC0 | - | - | | AVCC0 | - | - | - |
| 44 | AD0IN0 | - | AD0IN0 | - | Input | | AD0IN0 | Input | Hi-z | Hi-z |
| 45 | AD0IN1 | - | AD0IN1 | - | Input | | AD0IN1 | Input | Hi-z | Hi-z |
| 46 | AD0IN2 | - | AD0IN2 | - | Input | | AD0IN2 | Input | Hi-z | Hi-z |
| 47 | AD0IN3 | - | AD0IN3 | - | Input | | AD0IN3 | Input | Hi-z | Hi-z |
| 48 | AD0IN4 | - | AD0IN4 | - | Input | | AD0IN4 | Input | Hi-z | Hi-z |
| 49 | AD0IN5 | - | AD0IN5 | - | Input | | AD0IN5 | Input | Hi-z | Hi-z |
| 50 | AD0IN6 | - | AD0IN6 | - | Input | | AD0IN6 | Input | Hi-z | Hi-z |
| 51 | AD0IN7 | - | AD0IN7 | - | Input | | AD0IN7 | Input | Hi-z | Hi-z |
| 52 | AD0IN8 | - | AD0IN8 | - | Input | | AD0IN8 | Input | Hi-z | Hi-z |
| 53 | AD0IN9 | - | AD0IN9 | - | Input | | AD0IN9 | Input | Hi-z | Hi-z |
| 54 | AD0IN10 | - | AD0IN10 | - | Input | | AD0IN10 | Input | Hi-z | Hi-z |
| 55 | AD0IN11 | - | AD0IN11 | - | Input | | AD0IN11 | Input | Hi-z | Hi-z |
| 56 | AD0IN12 | - | AD0IN12 | - | Input | | AD0IN12 | Input | Hi-z | Hi-z |
| 57 | AD0IN13 | - | AD0IN13 | - | Input | | AD0IN13 | Input | Hi-z | Hi-z |
| 58 | AD0IN14 | - | AD0IN14 | - | Input | | AD0IN14 | Input | Hi-z | Hi-z |
| 59 | AD0IN15 | - | AD0IN15 | - | Input | | AD0IN15 | Input | Hi-z | Hi-z |
| 60 | AVSS0 | - | AVSS0 | - | - | | AVSS0 | - | - | - |
| 61 | VCCI | - | VCCI | - | - | | VCCI | - | - | - |

### Table 7.3.4 Pin Status When Reset (3/4)

| Pin NO. | Pin Name | Function | | | Input/output | Condition | Pin status when reset | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Port | Other than port | Other than port | | | Function | Input/output | Status during reset | Status after exiting reset |
| 62 | VSS | - | VSS | - | - | | VSS | - | - | - |
| 63 | P174/TXD2 | P174 | TXD2 | - | Input/output | | P174 | input | Hi-z | Hi-z |
| 64 | P175/RXD2 | P175 | RXD2 | - | Input/output | | P175 | input | Hi-z | Hi-z |
| 65 | VCCE | - | VCCE | - | - | | VCCE | - | - | - |
| 66 | P82/TXD0 | P82 | TXD0 | - | Input/output | | P82 | input | Hi-z | Hi-z |
| 67 | P83/RXD0 | P83 | RXD0 | - | Input/output | | P83 | input | Hi-z | Hi-z |
| 68 | P84/SCLKI0/SCLKO0 | P84 | SCLKI0 | SCLKO0 | Input/output | | P84 | input | Hi-z | Hi-z |
| 69 | P85/TXD1 | P85 | TXD1 | - | Input/output | | P85 | input | Hi-z | Hi-z |
| 70 | P86/RXD1 | P86 | RXD1 | - | Input/output | | P86 | input | Hi-z | Hi-z |
| 71 | P87/SCLKI1/SCLKO1 | P87 | SCLKI1 | SCLKO1 | Input/output | | P87 | input | Hi-z | Hi-z |
| 72 | VSS | - | VSS | - | - | | VSS | - | - | - |
| 73 | FVCC | - | FVCC | - | - | | FVCC | - | - | - |
| 74 | P61 | P61 | - | - | Input/output | | P61 | input | Hi-z | Hi-z |
| 75 | P62 | P62 | - | - | Input/output | | P62 | input | Hi-z | Hi-z |
| 76 | P63 | P63 | - | - | Input/output | | P63 | input | Hi-z | Hi-z |
| 77 | P64/$\overline{\text{SBI}}$ (Note 1) | P64 | $\overline{\text{SBI}}$ | - | Input | | $\overline{\text{SBI}}$ | input | Hi-z | Hi-z |
| 78 | P70/BCLK/$\overline{\text{WR}}$ | P70 | BCLK | $\overline{\text{WR}}$ | Input/output | | P70 | input | Hi-z | Hi-z |
| 79 | P71/$\overline{\text{WAIT}}$ | P71 | $\overline{\text{WAIT}}$ | - | Input/output | | P71 | input | Hi-z | Hi-z |
| 80 | P72/$\overline{\text{HREQ}}$ | P72 | $\overline{\text{HREQ}}$ | - | Input/output | | P72 | input | Hi-z | Hi-z |
| 81 | P73/$\overline{\text{HACK}}$ | P73 | $\overline{\text{HACK}}$ | - | Input/output | | P73 | input | Hi-z | Hi-z |
| 82 | P74/RTDTXD | P74 | RTDTXD | - | Input/output | | P74 | input | Hi-z | Hi-z |
| 83 | P75/RTDRXD | P75 | RTDRXD | - | Input/output | | P75 | input | Hi-z | Hi-z |
| 84 | P76/RTDACK | P76 | RTDACK | - | Input/output | | P76 | input | Hi-z | Hi-z |
| 85 | P77/RTDCLK | P77 | RTDCLK | - | Input/output | | P77 | input | Hi-z | Hi-z |
| 86 | P93/TO16 | P93 | TO16 | - | Input/output | | P93 | input | Hi-z | Hi-z |
| 87 | P94/TO17 | P94 | TO17 | - | Input/output | | P94 | input | Hi-z | Hi-z |
| 88 | P95/TO18 | P95 | TO18 | - | Input/output | | P95 | input | Hi-z | Hi-z |
| 89 | P96/TO19 | P96 | TO19 | - | Input/output | | P96 | input | Hi-z | Hi-z |
| 90 | P97/TO20 | P97 | TO20 | - | Input/output | | P97 | input | Hi-z | Hi-z |
| 91 | $\overline{\text{RESET}}$ | - | $\overline{\text{RESET}}$ | - | Input | | $\overline{\text{RESET}}$ | input | Hi-z | Hi-z |
| 92 | MOD0 | - | MOD0 | - | Input | | MOD0 | input | Hi-z | Hi-z |
| 93 | MOD1 | - | MOD1 | - | Input | | MOD1 | input | Hi-z | Hi-z |
| 94 | FP | - | FP | - | Input | | FP | input | Hi-z | Hi-z |
| 95 | VCCE | - | VCCE | - | - | | VCCE | - | - | - |
| 96 | VSS | - | VSS | - | - | | VSS | - | - | - |
| 97 | P110/TO0 | P110 | TO0 | - | Input/output | | P110 | input | Hi-z | Hi-z |
| 98 | P111/TO1 | P111 | TO1 | - | Input/output | | P111 | input | Hi-z | Hi-z |
| 99 | P112/TO2 | P112 | TO2 | - | Input/output | | P112 | input | Hi-z | Hi-z |
| 100 | P113/TO3 | P113 | TO3 | - | Input/output | | P113 | input | Hi-z | Hi-z |
| 101 | P114/TO4 | P114 | TO4 | - | Input/output | | P114 | input | Hi-z | Hi-z |
| 102 | P115/TO5 | P115 | TO5 | - | Input/output | | P115 | input | Hi-z | Hi-z |
| 103 | P116/TO6 | P116 | TO6 | - | Input/output | | P116 | input | Hi-z | Hi-z |
| 104 | P117/TO7 | P117 | TO7 | - | Input/output | | P117 | input | Hi-z | Hi-z |
| 105 | P100/TO8 | P100 | TO8 | - | Input/output | | P100 | input | Hi-z | Hi-z |
| 106 | P101/TO9 | P101 | TO9 | - | Input/output | | P101 | input | Hi-z | Hi-z |
| 107 | P102/TO10 | P102 | TO10 | - | Input/output | | P102 | input | Hi-z | Hi-z |
| 108 | VDD | - | VDD | - | - | | VDD | - | - | - |
| 109 | JTMS (Note 2) | - | JTMS | - | Input | | JTMS | input | Hi-z | Hi-z |
| 110 | JTCK (Note 2) | - | JTCK | - | Input | | JTCK | input | Hi-z | Hi-z |
| 111 | JTRST (Note 2) | - | JTRST | - | Input | | JTRST | input | Hi-z | Hi-z |
| 112 | JTDO (Note 2) | - | JTDO | - | Output | | JTDO | Output | Hi-z | Hi-z |
| 113 | JTDI (Note 2) | - | JTDI | - | Input | | JTDI | input | Hi-z | Hi-z |
| 114 | P103/TO11 | P103 | TO11 | - | Input/output | | P103 | input | Hi-z | Hi-z |
| 115 | P104/TO12 | P104 | TO12 | - | Input/output | | P104 | input | Hi-z | Hi-z |
| 116 | P105/TO13 | P105 | TO13 | - | Input/output | | P105 | input | Hi-z | Hi-z |
| 117 | P106/TO14 | P106 | TO14 | - | Input/output | | P106 | input | Hi-z | Hi-z |
| 118 | P107/TO15 | P107 | TO15 | - | Input/output | | P107 | input | Hi-z | Hi-z |
| 119 | P124/TCLK0 | P124 | TCLK0 | - | Input/output | | P124 | input | Hi-z | Hi-z |
| 120 | P125/TCLK1 | P125 | TCLK1 | - | Input/output | | P125 | input | Hi-z | Hi-z |

Note 1: P64 is used exclusively for $\overline{\text{SBI}}$ input.

Note 2: The JTCK, JTDI, JTDO, and JTMS pins are reset by the JTRST pin, and not by the $\overline{\text{RESET}}$ pin.
All of these pins are placed in the high-impedance state while the JTRST pin input is held low.

### Table 7.3.5 Pin Status When Reset (4/4)

| Pin NO. | Pin Name | Function | | | Input/output | Condition | Pin status when reset | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Port | Other than port | Other than Port | | | Function | Input/output | Status during reset | Status after exiting reset |
| 121 | P126/TCLK2 | P126 | TCLK2 | - | Input/output | | P126 | Input | Hi-z | Hi-z |
| 122 | P127/TCLK3 | P127 | TCLK3 | - | Input/output | | P127 | Input | Hi-z | Hi-z |
| 123 | VCCI | - | VCCI | - | - | | VCCI | - | - | - |
| 124 | P130/TIN16 | P130 | TIN16 | - | Input/output | | P130 | Input | Hi-z | Hi-z |
| 125 | P131/TIN17 | P131 | TIN17 | - | Input/output | | P131 | Input | Hi-z | Hi-z |
| 126 | P132/TIN18 | P132 | TIN18 | - | Input/output | | P132 | Input | Hi-z | Hi-z |
| 127 | P133/TIN19 | P133 | TIN19 | - | Input/output | | P133 | Input | Hi-z | Hi-z |
| 128 | P134/TIN20 | P134 | TIN20 | - | Input/output | | P134 | Input | Hi-z | Hi-z |
| 129 | P135/TIN21 | P135 | TIN21 | - | Input/output | | P135 | Input | Hi-z | Hi-z |
| 130 | P136/TIN22 | P136 | TIN22 | - | Input/output | | P136 | Input | Hi-z | Hi-z |
| 131 | P137/TIN23 | P137 | TIN23 | - | Input/output | | P137 | Input | Hi-z | Hi-z |
| 132 | VCCE | - | VCCE | - | - | | VCCE | - | - | - |
| 133 | P150/TIN0 | P150 | TIN0 | - | Input/output | | P150 | Input | Hi-z | Hi-z |
| 134 | P153/TIN3 | P153 | TIN3 | - | Input/output | | P153 | Input | Hi-z | Hi-z |
| 135 | P41/$\overline{\text{BLW}}$/$\overline{\text{BLE}}$ | P41 | $\overline{\text{BLW}}$ | $\overline{\text{BLE}}$ | Input/output | During single-chip mode | P41 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | $\overline{\text{BLW}}$ | Output | Hi-z | "H" level |
| 136 | P42/$\overline{\text{BHW}}$/$\overline{\text{BHE}}$ | P42 | $\overline{\text{BHW}}$ | $\overline{\text{BHE}}$ | Input/output | During single-chip mode | P42 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | $\overline{\text{BHW}}$ | Output | Hi-z | "H" level |
| 137 | VCCI | - | VCCI | - | - | | VCCI | - | - | - |
| 138 | VSS | - | VSS | - | - | | VSS | - | - | - |
| 139 | P43/$\overline{\text{RD}}$ | P43 | $\overline{\text{RD}}$ | - | Input/output | During single-chip mode | P43 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | $\overline{\text{RD}}$ | Output | Hi-z | "H" level |
| 140 | P44/$\overline{\text{CS0}}$ | P44 | $\overline{\text{CS0}}$ | - | Input/output | During single-chip mode | P44 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | $\overline{\text{CS0}}$ | Output | Hi-z | "H" level |
| 141 | P45/$\overline{\text{CS1}}$ | P45 | $\overline{\text{CS1}}$ | - | Input/output | During single-chip mode | P45 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | $\overline{\text{CS1}}$ | Output | Hi-z | "H" level |
| 142 | P46/A13 | P46 | A13 | - | Input/output | During single-chip mode | P46 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A13 | Output | Hi-z | Indeterminate |
| 143 | P47/A14 | P47 | A14 | - | Input/output | During single-chip mode | P47 | Input | Hi-z | Hi-z |
| | | | | | | During external extension or processor mode | A14 | Output | Hi-z | Indeterminate |
| 144 | P220/CTX | P220 | CTX | - | Input/output | | P220 | Input | Hi-z | Hi-z |

## 7.4 Things To Be Considered after Exiting Reset

- **Input/output ports**

After exiting reset, the 32171's input/output ports are disabled against input in order to prevent current from flowing through the port. To use any ports in input mode, enable them for input using the Port Input Function Enable Register (PIEN) PIEN0 bit. For details, refer to Section 8.3, "Input/ Output Port Related Registers."

# CHAPTER 8
# INPUT/OUTPUT PORTS AND PIN FUNCTIONS

8.1 Outline of Input/Output Ports
8.2 Selecting Pin Functions
8.3 Input/Output Port Related Registers
8.4 Port Peripheral Circuits
8.5 Precautions on Input/output Ports

## 8.1 Outline of Input/Output Ports

The 32171 has a total of 97 input/output ports consisting of P0–P13, P15, P17, and P22 (with P5 reserved for future use, however). These input/output ports can be used as input ports or output ports by setting up the direction registers.

Each input/output port serves as a dual-function or triple-function pin, sharing the pin with other internal peripheral I/O or external extension bus signal line. Pin functions are selected depending on the device's operation mode you choose or by setting the input/output port's Operation Mode Register. (If any internal peripheral I/O has still another function, you need to set the register provided for that peripheral I/O.)

As a new function, the 32171 internally contains a Port Input Function Enable bit that can be used to prevent current from flowing into the input ports. This helps to simplify the software and hardware processing to be performed immediately after reset or during flash rewrite. To use any ports in input mode, you need to set the Port Input Function Enable bit accordingly.

The input/output ports are outlined in the next pages.

Table 8.1.1 Outline of Input/Output Ports

| Item | Specification |
|---|---|
| Number of ports | Total 97 lines |
| | P0 : P00 - P07 (8 lines) |
| | P1 : P10 - P17 (8 lines) |
| | P2 : P20 - P27 (8 lines) |
| | P3 : P30 - P37 (8 lines) |
| | P4 : P41 - P47 (7 lines) |
| | P6 : P61 - P64 (4 lines) |
| | P7 : P70 - P77 (8 lines) |
| | P8 : P82 - P87 (6 lines) |
| | P9 : P93 - P97 (5 lines) |
| | P10 : P100 - P107 (8 lines) |
| | P11 : P110 - P117 (8 lines) |
| | P12 : P124 - P127 (4 lines) |
| | P13 : P130 - P137 (8 lines) |
| | P15 : P150 , P153 (2 lines) |
| | P17 : P174, P175 (2 lines) |
| | P22 : P220, P221, P225 (3 lines) |
| Port function | The input/output ports can individually be set for input or output mode using the Direction Control Register provided for each input/output port. (However, P64 is a $\overline{\text{SBI}}$ input-only port and P221 is a CAN input-only port.) |
| Pin function | Shared with peripheral I/O or external extension signals to serve dual functions (or with two or more peripheral I/O functions to serve multiple functions) |
| Pin function switchover | P0 - P4, P225 : Depends on CPU operation mode (determined by setting MOD0 and MOD1 pins) |
| | P6 - P22 : As set by each input/output port's Operation Mode Register (However, peripheral I/O pin functions are selected by peripheral I/O registers.) |

Note: • P14, P16, and P18–P21 are nonexistent.

## 8.2 Selecting Pin Functions

Each input/output port serves dual purposes along with other internal peripheral I/Os or external extension bus signal lines (or triple purposes along with multiple functions of peripheral I/O). Pin functions are selected according to the operation modes set or using the input/output port operation mode registers.
When the selected CPU operation mode is external extension mode or processor mode, P0–P4 and P225 all are switched to signal pins for external access. The operation mode is determined depending on how MOD0 and MOD1 pins are set. (See the table below.)

**Table 8.2.1 CPU Operation Modes and P0–P4 and P225 Pin Functions**

| MOD0 | MOD1 | Operation Mode | Pin Functions of P0-P4, P225 |
|---|---|---|---|
| VSS | VSS | Single-chip mode | input/output port pin |
| VSS | VCCE | External extension mode | External extension signal pin |
| VCCE | VSS | Processor mode | |
| VCCE | VCC | Reserved (Use inhibited) | — |

Note: • VCCE = 5 V or 3.3 V and VSS = GND.

Ports P6–P13, P15, P17, and P22 (except for P64, P221, P225) have their pin functions switched between input/output ports and internal peripheral I/Os by setting up the input/output port operation mode registers. If any internal peripheral I/O has multiple functions, select the desired pin function using the relevant internal peripheral I/O register.

Operation on FP and MOD1 pins during write to the internal flash memory does not affect the pin functions.

| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| Settings of CPU operation mode (Note 1) | P0 | DB0 | DB1 | DB2 | DB3 | DB4 | DB5 | DB6 | DB7 |
| | P1 | DB8 | DB9 | DB10 | DB11 | DB12 | DB13 | DB14 | DB15 |
| | P2 | A23 | A24 | A25 | A26 | A27 | A28 | A29 | A30 |
| | P3 | A15 | A16 | A17 | A18 | A19 | A20 | A21 | A22 |
| | P4 | | $\overline{BLW}$/<br>$\overline{BLE}$ | $\overline{BHW}$/<br>$\overline{BHE}$ | $\overline{RD}$ | $\overline{CS0}$ | $\overline{CS1}$ | A13 | A14 |
| (Reserved) | P5 | | | | | | | | |
| Settings of input/ output port Operation Mode Register | P6 | | (P61) | (P62) | (P63) | $\overline{SBI}$ | | | |
| | P7 | BCLK/<br>$\overline{WR}$ | $\overline{WAIT}$ | $\overline{HREQ}$ | $\overline{HACK}$ | RTDTXD | RTDRXD | RTDACK | RTDCLK |
| | P8 | | | TXD0 | RXD0 | SCLKI0/<br>SCLKO0 | TXD1 | RXD1 | SCLKI1/<br>SCLKO1 |
| | P9 | | | | TO16 | TO17 | TO18 | TO19 | TO20 |
| | P10 | TO8 | TO9 | TO10 | TO11 | TO12 | TO13 | TO14 | TO15 |
| | P11 | TO0 | TO1 | TO2 | TO3 | TO4 | TO5 | TO6 | TO7 |
| | P12 | | | | | TCLK0 | TCLK1 | TCLK2 | TCLK3 |
| | P13 | TIN16 | TIN17 | TIN18 | TIN19 | TIN20 | TIN21 | TIN22 | TIN23 |
| | P14 | | | | | | | | |
| | P15 | TIN0 | | | TIN3 | | | | |
| | P16 | | | | | | | | |
| | P17 | | | | | TXD2 | RXD2 | | |
| | P18 | | | | | | | | |
| | P19 | | | | | | | | |
| | P20 | | | | | | | | |
| | P21 | | | | | | | | |
| | P22 | CTX | CRX | | | | A12<br>(Note 2) | | |

Note 1: Pin functions are switched over by setting MOD0 and MOD1 pins.

Note 2: Pin functions are switched over by setting MOD0 and MOD1 pins. Also, use of this pin requires caution because it has a debug event function.

Note: • P14, P16, P18, P19, P20 and P21 have no functions assigned in M32171.

**Figure 8.2.1 Input/Output Ports and Pin Function Assignments**

## 8.3 Input/Output Port Related Registers

The input/output port related registers consist of the Port Data Register, Port Direction Register, and Port Operation Mode Register. Of these, the Port Operation Mode Register is available for only P7–P22. Ports P0–P4 and P225 have their pin functions determined depending on CPU operation mode (selected by FP, MOD0, and MOD1 pins).
Port P5 is reserved for future use. An input/output port related register map is shown below.

| Address | +0 Address<br>D0 … D7 | +1 Address<br>D8 … D15 |
|---|---|---|
| H'0080 0700 | P0 Data Register (P0DATA) | P1 Data Register (P1DATA) |
| H'0080 0702 | P2 Data Register (P2DATA) | P3 Data Register (P3DATA) |
| H'0080 0704 | P4 Data Register (P4DATA) | |
| H'0080 0706 | P6 Data Register (P6DATA) | P7 Data Register (P7DATA) |
| H'0080 0708 | P8 Data Register (P8DATA) | P9 Data Register (P9DATA) |
| H'0080 070A | P10 Data Register (P10DATA) | P11 Data Register (P11DATA) |
| H'0080 070C | P12 Data Register (P12DATA) | P13 Data Register (P13DATA) |
| H'0080 070E | | P15 Data Register (P15DATA) |
| H'0080 0710 | | P17 Data Register (P17DATA) |
| H'0080 0712 | | |
| H'0080 0714 | | |
| H'0080 0716 | P22 Data Register (P22DATA) | |
| | | |
| H'0080 0720 | P0 Direction Register (P0DIR) | P1 Direction Register (P1DIR) |
| H'0080 0722 | P2 Direction Register (P2DIR) | P3 Direction Register (P3DIR) |
| H'0080 0724 | P4 Direction Register (P4DIR) | |
| H'0080 0726 | P6 Direction Register (P6DIR) | P7 Direction Register (P7DIR) |
| H'0080 0728 | P8 Direction Register (P8DIR) | P9 Direction Register (P9DIR) |
| H'0080 072A | P10 Direction Register (P10DIR) | P11 Direction Register (P11DIR) |
| H'0080 072C | P12 Direction Register (P12DIR) | P13 Direction Register (P13DIR) |
| H'0080 072E | | P15 Direction Register (P15DIR) |
| H'0080 0730 | | P17 Direction Register (P17DIR) |
| H'0080 0732 | | |
| H'0080 0734 | | |
| H'0080 0736 | P22 Direction Register (P22DIR) | |

Blank addresses are reserved.

Note : • The Data Register, Direction Register, and Operation Mode Register for P14, P16, and P18-P21 are not included.

**Figure 8.3.1 Input/Output Port Related Register Map (1/2)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 | D8 D15 |
| H'0080 0744 | | Port Input Function Enable Register (PIEN) |
| H'0080 0746 | | P7 Operation Mode Register (P7MOD) |
| H'0080 0748 | P8 Operation Mode Register (P8MOD) | P9 Operation Mode Register (P9MOD) |
| H'0080 074A | P10 Operation Mode Register (P10MOD) | P11 Operation Mode Register (P11MOD) |
| H'0080 074C | P12 Operation Mode Register (P12MOD) | P13 Operation Mode Register (P13MOD) |
| H'0080 074E | | P15 Operation Mode Register (P15MOD) |
| H'0080 0750 | | P17 Operation Mode Register (P17MOD) |
| H'0080 0752 | | |
| H'0080 0754 | | |
| H'0080 0756 | P22 Operation Mode Register (P22MOD) | |

Blank addresses are reserved.

**8.3.2 Input/Output Port Related Register Map (2/2)**

### 8.3.1 Port Data Registers

■ **P0 Data Register (P0DATA)** <Address: H'0080 0700>

■ **P1 Data Register (P1DATA)** <Address: H'0080 0701>

■ **P2 Data Register (P2DATA)** <Address: H'0080 0702>

■ **P3 Data Register (P3DATA)** <Address: H'0080 0703>

■ **P4 Data Register (P4DATA)** <Address: H'0080 0704>

■ **P6 Data Register (P6DATA)** <Address: H'0080 0706>

■ **P7 Data Register (P7DATA)** <Address: H'0080 0707>

■ **P8 Data Register (P8DATA)** <Address: H'0080 0708>

■ **P9 Data Register (P9DATA)** <Address: H'0080 0709>

■ **P10 Data Register (P10DATA)** <Address: H'0080 070A>

■ **P11 Data Register (P11DATA)** <Address: H'0080 070B>

■ **P12 Data Register (P12DATA)** <Address: H'0080 070C>

■ **P13 Data Register (P13DATA)** <Address: H'0080 070D>

■ **P15 Data Register (P15DATA)** <Address: H'0080 070F>

■ **P17 Data Register (P17DATA)** <Address: H'0080 0711>

■ **P22 Data Register (P22DATA)** <Address: H'0080 0716>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ( D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 ) |
| Pn0DT | Pn1DT | Pn2DT | Pn3DT | Pn4DT | Pn5DT | Pn6DT | Pn7DT |

Note: • n = 0-13, 15, 17, and 22 (not including P5).

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 (8) | Pn0DT (Port Pn0 data) | Depending on how the Port Direction Register is set | ○ | ○ |
| 1 (9) | Pn1DT (Port Pn1 data) | • When direction bit = 0 (input mode) | ○ | ○ |
| 2 (10) | Pn2DT (Port Pn2 data) | 0: Port input pin = low | ○ | ○ |
| 3 (11) | Pn3DT (Port Pn3 data) | 1: Port input pin = high | ○ | ○ |
| 4 (12) | Pn4DT (Port Pn4 data) | • When direction bit = 1 (output mode) | ○ | ○ |
| 5 (13) | Pn5DT (Port Pn5 data) | 0: Port output latch = low | ○ | ○ |
| 6 (14) | Pn6DT (Port Pn6 data) | 1: Port output latch = high | ○ | ○ |
| 7 (15) | Pn7DT (Port Pn7 data) |  | ○ | ○ |

Notes: • The bits listed below have no functions assigned. (They show a 0 when read; writing to these bits has no effect.)
P40, P60, P65-P67, P90-P92, P120-P123, P151, P152, P154-P157, P170-P173, P176, P177, P222-P224, P226, P227

: • Port P64 is available for only input mode. Writing to P64DT bit has no effect.

: • Ports P80 and P81 are available for only input mode. Writing to P80DT and P81DT bits has no effect. When read, P80 and P81 show the MOD0 and MOD1 pin levels, respectively.

: • Port P221 is available for only input mode. Writing to P221DT bit has no effect.

: • P14, P16, and P18-P21 do not have data registers.

### 8.3.2 Port Direction Registers

| | |
|---|---|
| ■ **P0 Direction Register (P0DIR)** | <Address: H'0080 0720> |
| ■ **P1 Direction Register (P1DIR)** | <Address: H'0080 0721> |
| ■ **P2 Direction Register (P2DIR)** | <Address: H'0080 0722> |
| ■ **P3 Direction Register (P3DIR)** | <Address: H'0080 0723> |
| ■ **P4 Direction Register (P4DIR)** | <Address: H'0080 0724> |
| ■ **P6 Direction Register (P6DIR)** | <Address: H'0080 0726> |
| ■ **P7 Direction Register (P7DIR)** | <Address: H'0080 0727> |
| ■ **P8 Direction Register (P8DIR)** | <Address: H'0080 0728> |
| ■ **P9 Direction Register (P9DIR)** | <Address: H'0080 0729> |
| ■ **P10 Direction Register (P10DIR)** | <Address: H'0080 072A> |
| ■ **P11 Direction Register (P11DIR)** | <Address: H'0080 072B> |
| ■ **P12 Direction Register (P12DIR)** | <Address: H'0080 072C> |
| ■ **P13 Direction Register (P13DIR)** | <Address: H'0080 072D> |
| ■ **P15 Direction Register (P15DIR)** | <Address: H'0080 072F> |
| ■ **P17 Direction Register (P17DIR)** | <Address: H'0080 0731> |
| ■ **P22 Direction Register (P22DIR)** | <Address: H'0080 0736> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ( D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 ) |
| Pn0DIR | Pn1DIR | Pn2DIR | Pn3DIR | Pn4DIR | Pn5DIR | Pn6DIR | Pn7DIR |

Note: • n = 0-13, 15, 17, and 22 (not including P5).

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 (8) | Pn0DIR (Port Pn0 direction bit) | 0: Input mode (when reset) | ○ | ○ |
| 1 (9) | Pn1DIR (Port Pn1 direction bit) | 1: Output mode | ○ | ○ |
| 2 (10) | Pn2DIR (Port Pn2 direction bit) | | ○ | ○ |
| 3 (11) | Pn3DIR (Port Pn3 direction bit) | | ○ | ○ |
| 4 (12) | Pn4DIR (Port Pn4 direction bit) | | ○ | ○ |
| 5 (13) | Pn5DIR (Port Pn5 direction bit) | | ○ | ○ |
| 6 (14) | Pn6DIR (Port Pn6 direction bit) | | ○ | ○ |
| 7 (15) | Pn7DIR (Port Pn7 direction bit) | | ○ | ○ |

Notes: • he bits listed below have no functions assigned. (They show a 0 when read; writing to these bits has no effect.)
P40, P60, P65-P67, P90-P92, P120-P123, P151, P152, P154-P157,
P170-P173, P176, P177, P222-P224, P226, P227
: • When reset, all ports are placed in input mode.
: • Port P64 is input mode-only. The register does not have a P64DIR bit.
: • Port P221 is input mode-only. The register does not have a P221DIR bit.
: • Ports P80 and P81 are input mode-only. The register does not have P80DIR and P81DIR bits.
: • P14, P16, and P18-P21 do not have data registers.

### 8.3.3 Port Operation Mode Registers

**■ P7 Operation Mode Register (P7MOD)** <Address: H'0080 0747>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| P70MOD | P71MOD | P72MOD | P73MOD | P74MOD | P75MOD | P76MOD | P77MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | P70MOD<br>(Port P70 operation mode) | 0 : P70<br>1 : BCLK / $\overline{\text{WR}}$ | ○ | ○ |
| 9 | P71MOD<br>(Port P71 operation mode) | 0 : P71<br>1 : $\overline{\text{WAIT}}$ | ○ | ○ |
| 10 | P72MOD<br>(Port P72 operation mode) | 0 : P72<br>1 : $\overline{\text{HREQ}}$ | ○ | ○ |
| 11 | P73MOD<br>(Port P73 operation mode) | 0 : P73<br>1 : $\overline{\text{HACK}}$ | ○ | ○ |
| 12 | P74MOD<br>(Port P74 operation mode) | 0 : P74<br>1 : RTDTXD | ○ | ○ |
| 13 | P75MOD<br>(Port P75 operation mode) | 0 : P75<br>1 : RTDRXD | ○ | ○ |
| 14 | P76MOD<br>(Port P76 operation mode) | 0 : P76<br>1 : RTDACK | ○ | ○ |
| 15 | P77MOD<br>(Port P77 operation mode) | 0 : P77<br>1 : RTDCLK | ○ | ○ |

## ■ P8 Operation Mode Register (P8MOD)

<Address: H'0080 0748>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | P82MOD | P83MOD | P84MOD | P85MOD | P86MOD | P87MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | No functions assigned | | 0 | — |
| 2 | P82MOD<br>(Port P82 operation mode) | 0 : P82<br>1 : TXD0 | ○ | ○ |
| 3 | P83MOD<br>(Port P83 operation mode) | 0 : P83<br>1 : RXD0 | ○ | ○ |
| 4 | P84MOD<br>(Port P84 operation mode) | 0 : P84<br>1 : SCLKI0 / SCLKO0 | ○ | ○ |
| 5 | P85MOD<br>(Port P85 operation mode) | 0 : P85<br>1 : TXD1 | ○ | ○ |
| 6 | P86MOD<br>(Port P86 operation mode) | 0 : P86<br>1 : RXD1 | ○ | ○ |
| 7 | P87MOD<br>(Port P87 operation mode) | 0 : P87<br>1 : SCLKI1 / SCLKO1 | ○ | ○ |

Note : • Ports P80 and P81 are nonexistent.

### ■ P9 Operation Mode Register (P9MOD)

<Address: H'0080 0749>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | | P93MOD | P94MOD | P95MOD | P96MOD | P97MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | No functions assigned | | 0 | — |
| 11 | P93MOD<br>(Port P93 operation mode) | 0 : P93<br>1 : TO16 | ○ | ○ |
| 12 | P94MOD<br>(Port P94 operation mode) | 0 : P94<br>1 : TO17 | ○ | ○ |
| 13 | P95MOD<br>(Port P95 operation mode) | 0 : P95<br>1 : TO18 | ○ | ○ |
| 14 | P96MOD<br>(Port P96 operation mode) | 0 : P96<br>1 : TO19 | ○ | ○ |
| 15 | P97MOD<br>(Port P97 operation mode) | 0 : P97<br>1 : TO20 | ○ | ○ |

Note : • Ports P90 - P92 are nonexistent.

## ■ P10 Operation Mode Register (P10MOD)

<Address: H'0080 074A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| P100MOD | P101MOD | P102MOD | P103MOD | P104MOD | P105MOD | P106MOD | P107MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | P100MOD<br>(Port P100 operation mode) | 0 : P100<br>1 : TO8 | ○ | ○ |
| 1 | P101MOD<br>(Port P101 operation mode) | 0 : P101<br>1 : TO9 | ○ | ○ |
| 2 | P102MOD<br>(Port P102 operation mode) | 0 : P102<br>1 : TO10 | ○ | ○ |
| 3 | P103MOD<br>(Port P103 operation mode) | 0 : P103<br>1 : TO11 | ○ | ○ |
| 4 | P104MOD<br>(Port P104 operation mode) | 0 : P104<br>1 : TO12 | ○ | ○ |
| 5 | P105MOD<br>(Port P105 operation mode) | 0 : P105<br>1 : TO13 | ○ | ○ |
| 6 | P106MOD<br>(Port P106 operation mode) | 0 : P106<br>1 : TO14 | ○ | ○ |
| 7 | P107MOD<br>(Port P107 operation mode) | 0 : P107<br>1 : TO15 | ○ | ○ |

## ■ P11 Operation Mode Register (P11MOD)

<Address: H'0080 074B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| P110MOD | P111MOD | P112MOD | P113MOD | P114MOD | P115MOD | P116MOD | P117MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | P110MOD<br>(Port P110 operation mode) | 0 : P110<br>1 : TO0 | ○ | ○ |
| 9 | P111MOD<br>(Port P111 operation mode) | 0 : P111<br>1 : TO1 | ○ | ○ |
| 10 | P112MOD<br>(Port P112 operation mode) | 0 : P112<br>1 : TO2 | ○ | ○ |
| 11 | P113MOD<br>(Port P113 operation mode) | 0 : P113<br>1 : TO3 | ○ | ○ |
| 12 | P114MOD<br>(Port P114 operation mode) | 0 : P114<br>1 : TO4 | ○ | ○ |
| 13 | P115MOD<br>(Port P115 operation mode) | 0 : P115<br>1 : TO5 | ○ | ○ |
| 14 | P116MOD<br>(Port P116 operation mode) | 0 : P116<br>1 : TO6 | ○ | ○ |
| 15 | P117MOD<br>(Port P117 operation mode) | 0 : P117<br>1 : TO7 | ○ | ○ |

## ■ P12 Operation Mode Register (P12MOD)

<Address: H'0080 074C>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  |  | P124MOD | P125MOD | P126MOD | P127MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 3 | No functions assigned |  | 0 | — |
| 4 | P124MOD<br>(Port P124 operation mode) | 0 : P124<br>1 : TCLK0 | ○ | ○ |
| 5 | P125MOD<br>(Port P125 operation mode) | 0 : P125<br>1 : TCLK1 | ○ | ○ |
| 6 | P126MOD<br>(Port P126 operation mode) | 0 : P126<br>1 : TCLK2 | ○ | ○ |
| 7 | P127MOD<br>(Port P127 operation mode) | 0 : P127<br>1 : TCLK3 | ○ | ○ |

Note : • Ports P120 - P123 are nonexistent.

## ■ P13 Operation Mode Register (P13MOD)

<Address: H'0080 074D>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| P130MOD | P131MOD | P132MOD | P133MOD | P134MOD | P135MOD | P136MOD | P137MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | P130MOD<br>(Port P130 operation mode) | 0 : P130<br>1 : TIN16 | ○ | ○ |
| 9 | P131MOD<br>(Port P131 operation mode) | 0 : P131<br>1 : TIN17 | ○ | ○ |
| 10 | P132MOD<br>(Port P132 operation mode) | 0 : P132<br>1 : TIN18 | ○ | ○ |
| 11 | P133MOD<br>(Port P133 operation mode) | 0 : P133<br>1 : TIN19 | ○ | ○ |
| 12 | P134MOD<br>(Port P134 operation mode) | 0 : P134<br>1 : TIN20 | ○ | ○ |
| 13 | P135MOD<br>(Port P135 operation mode) | 0 : P135<br>1 : TIN21 | ○ | ○ |
| 14 | P136MOD<br>(Port P136 operation mode) | 0 : P136<br>1 : TIN22 | ○ | ○ |
| 15 | P137MOD<br>(Port P137 operation mode) | 0 : P137<br>1 : TIN23 | ○ | ○ |

## ■ P15 Operation Mode Register (P15MOD)

<Address: H'0080 074F>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| P150MOD | | | P153MOD | | | | |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | P150MOD<br>(Port P150 operation mode) | 0 : P150<br>1 : TIN0 | ○ | ○ |
| 9, 10 | No functions assigned | | 0 | – |
| 11 | P153MOD<br>(Port P153 operation mode) | 0 : P153<br>1 : TIN3 | ○ | ○ |
| 12 - 15 | No functions assigned | | 0 | – |

Note: • Ports P151, P152, and P154-157 are nonexistent.

### ■ P17 Operation Mode Register (P17MOD)

<Address: H'0080 0751>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  |  | P174MOD | P175MOD |  |  |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 11 | No functions assigned |  | 0 | — |
| 12 | P174MOD<br>(Port P174 operation mode) | 0 : P174<br>1 : TXD2 | ○ | ○ |
| 13 | P175MOD<br>(Port P175 operation mode) | 0 : P175<br>1 : RXD2 | ○ | ○ |
| 14, 15 | No functions assigned |  | 0 | — |

Note : • Ports P170-P173, and P176, P177 are nonexistent.

### ■ P22 Operation Mode Register (P22MOD)

<Address: H'0080 0756>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| P220MOD | | | | | P225MOD | | |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | P220MOD<br>(Port P220 operation mode) | 0 : P220<br>1 : CTX | ○ | ○ |
| 1 - 4 | No functions assigned | | 0 | — |
| 5 | P225MOD<br>(Port P225 operation mode) | 0 : P225<br>1 : Use inhibited | ○ | ○ |
| 6 - 7 | No functions assigned | | 0 | — |

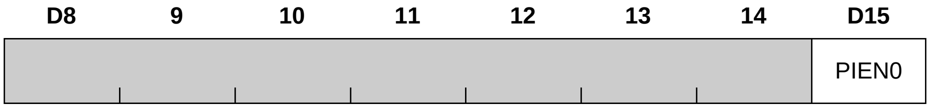
Notes: • P221 is a CAN input-only pin.

: • The pin function of P225 changes depending on how MOD0 and MOD1 pins are set. Also, because it has a debug event function, be careful when using this port.

: • P222-224, P226, and P227 are nonexistent.

## ■ Port Input Function Enable Register (PIEN)

<Address: H'0080 0745>



<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 14 | No functions assigned | | 0 | — |
| 15 | PIEN0<br>(Port input function enable bit) | 0 : Disables input (to prevent current from flowing in)<br>1 : Enables input | ○ | ○ |

This register is provided to prevent current from flowing into the port input pin. Because after reset this register is set to disable input, it must be set to 1 before input can be processed.
During boot mode, all pins shared with serial I/O function are enabled for input, so that when rewriting the flash memory via serial communication, you can set this register to 0 to prevent current from flowing in from any pins other than serial I/O function.

The next page lists the pins that can be controlled by the Port Input Function Enable Register in each mode.

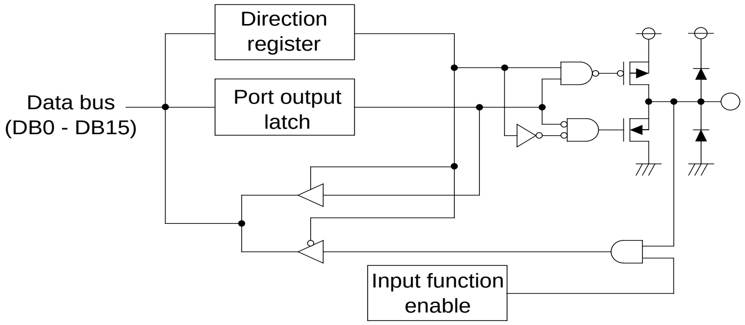
Table 8.3.1 Controllable Pins by Port Function Enable Bit

| Mode Name | Controllable Pins | Noncontrollable Pins |
|---|---|---|
| Single chip | P00 - P07, P10 - P17, P20 - P27<br>P30 -P37 , P41 - P47, P61 - P63<br>P70 - P77, P82 - P87, P93 - P97<br>P100 - P107, P110 - P117, P124 - P127<br>P130 - P137, P150, P153, P174, P175<br>P220, P225 | P64, P221, FP |
| External extension<br>Microprocessor | P61 - P63, P70 - P77, P82 - P87<br>P93 - P97, P100 - P107, P110 - P117<br>P124 - P127, P130 - P137<br>P150, P153, P174, P175, P220 | P00 - P07, P10 - P17<br>P20 - P27, P30 - P37<br>P41 - P47, P64, P221<br>P225, FP |
| Boot (single chip) | P00 - P07, P10 - P17, P20 - P27<br>P30 -P37 , P41 - P47, P61 - P63<br>P67, P70 - P77, P93 - P97<br>P100 - P107, P110 - P117, P124 - P127<br>P130 - P137, P150, P153, P220, P225 | P64, P82 - P87<br>P174, P175, P221, FP |

## 8.4 Port Peripheral Circuits

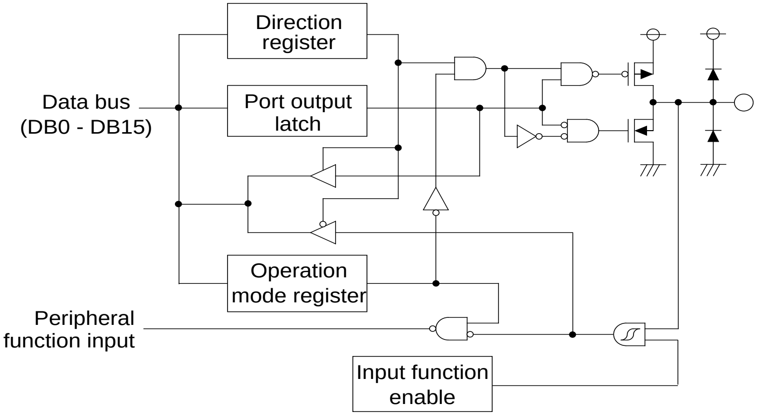
Figures 8.4.1 through 8.4.4 show the peripheral circuit diagrams of the input/output ports described in the preceding pages.

**P00 - P07 (DB0-DB7)**
**P10 - P17 (DB8-DB15)**
**P20 - P27 (A23-A30)**
**P30 - P37 (A15-A22)**
**P41 ($\overline{\text{BLW}}$ / $\overline{\text{BLE}}$)**
**P42 ($\overline{\text{BHW}}$ / $\overline{\text{BHE}}$)**
**P43 ($\overline{\text{RD}}$)**
**P44 ($\overline{\text{CS0}}$)**
**P45 ($\overline{\text{CS1}}$)**
**P46 - P47 (A13-A14)**
**P61 - P63**
**P225(A12)**



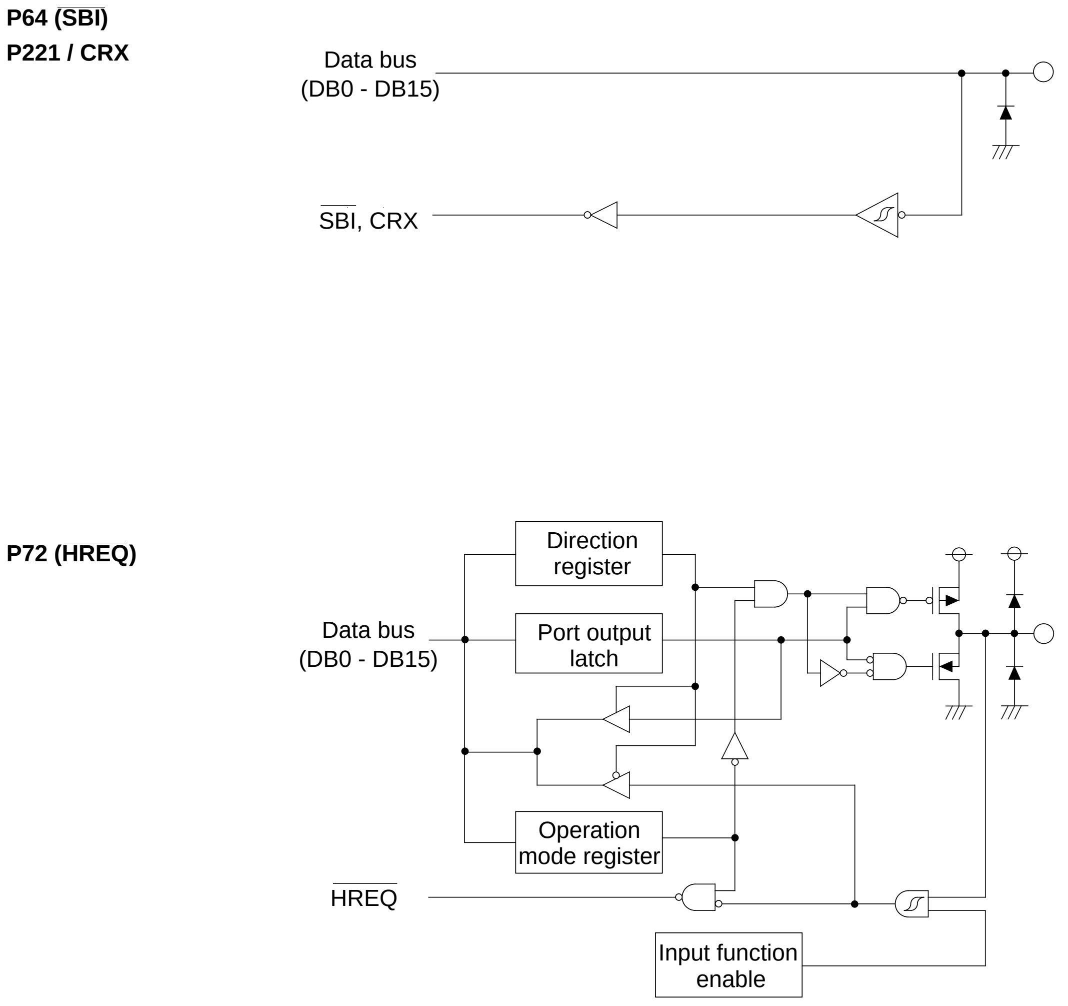
Note: • Although P00-07, P10-17, P20-27, P30-37, P41-47, and P225 serve as external bus interface control signal pins during external extension mode and processor mode, functional description is eliminated in this block diagram.

**P75 (RTDRXD)**
**P77 (RTDCLK)**
**P83 (RXD0)**
**P86 (RXD1)**
**P124 - P127 (TCLK0-TCLK3)**
**P130 - P137 (TIN16-TIN23)**
**P150, P153 (TIN0, TIN3)**
**P175 (RXD2)**



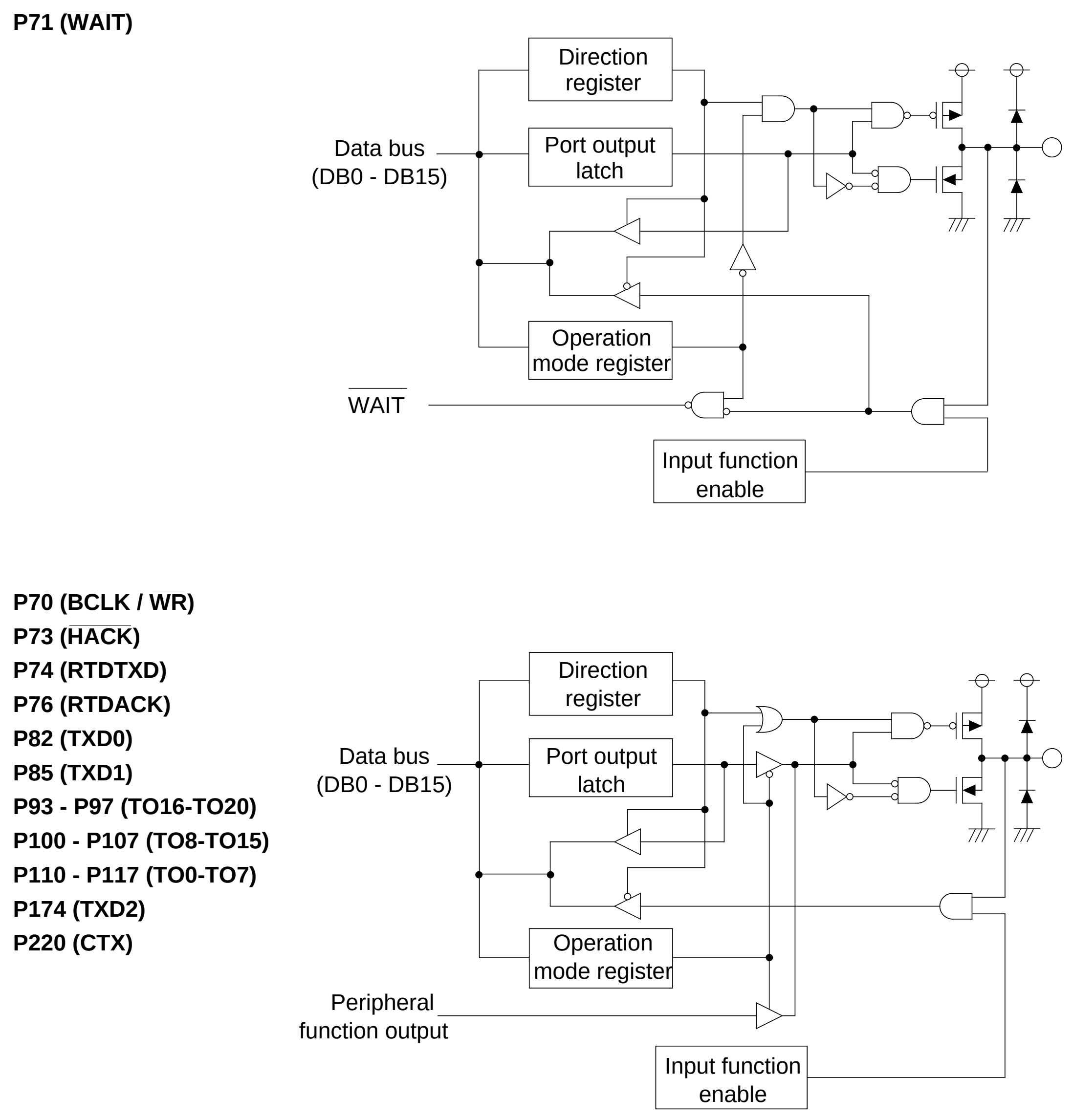
Notes: • ○ denotes pins.
: • ─|◀─ indicates a parasitic diode. Make sure the voltages applied to each port do not exceed VCCE.
: • The input capacitance of each pin is approximately 10 pF.

**Figure 8.4.1 Port Peripheral Circuit Diagram (1)**



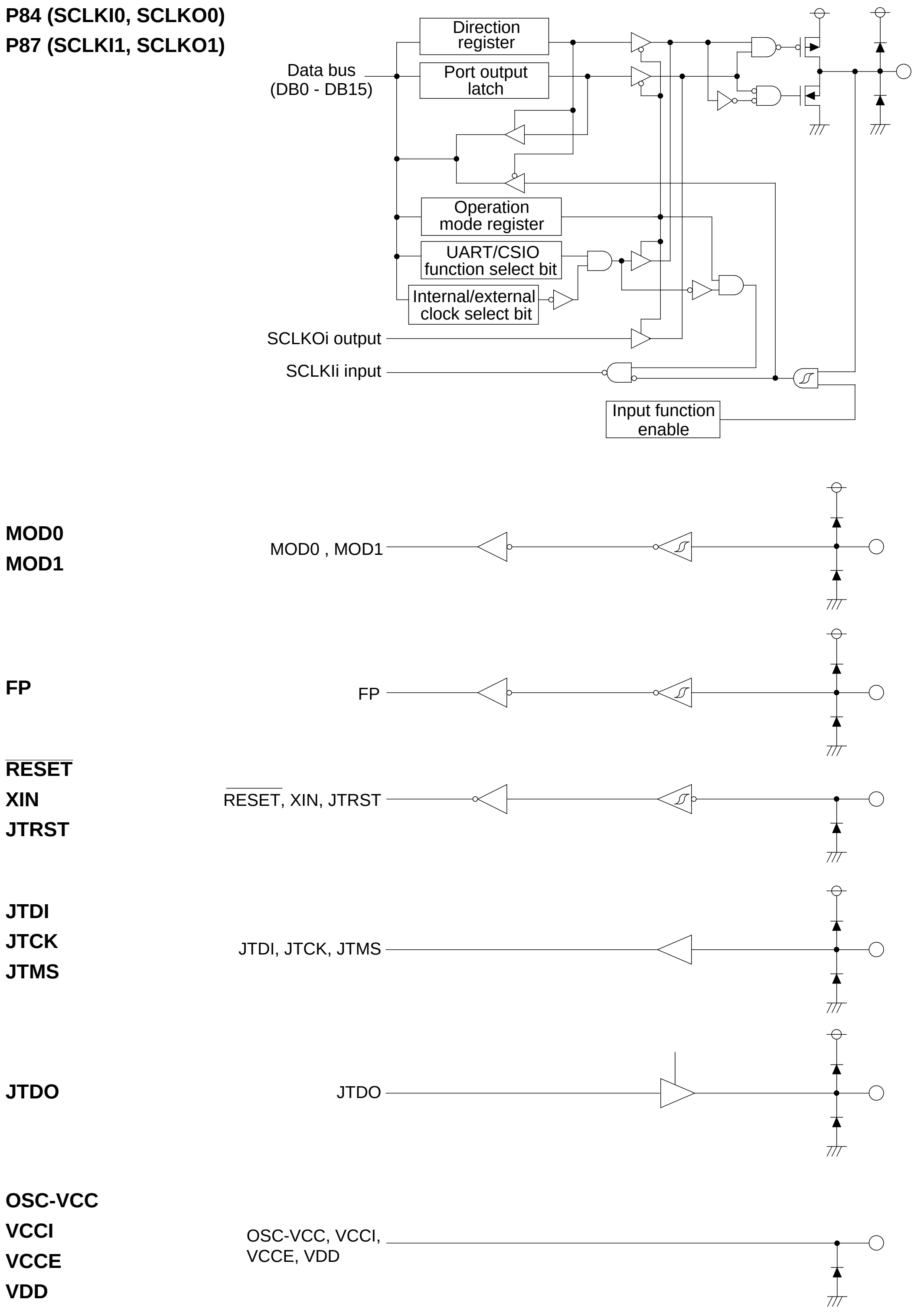
Notes: • ○ denotes pins.
: • —|◀— indicates a parasitic diode. Make sure the voltages applied to each port do not exceed VCCE.
: • The input capacitance of each pin is approximately 10 pF.

**Figure 8.4.2 Port Peripheral Circuit Diagram (2)**



Notes: • ○ denotes pins.
: • —|◀— indicates a parasitic diode. Make sure the voltages applied to each port do not exceed VCCE.
: • The input capacitance of each pin is approximately 10 pF.

**Figure 8.4.3 Port Peripheral Circuit Diagram (3)**



Notes: • ○ denotes pins.

: • —|◀— indicates a parasitic diode. Make sure the voltages applied to each port do not exceed VCCE.

**Figure 8.4.4 Port Peripheral Circuit Diagram (4)**

## 8.5 Precautions on Input/output Ports

### • When using the ports in output mode

Because the Port Data Register values immediately after a reset are indeterminate, it is necessary that the initial value be written to the Port Data Register before setting the Port Direction Register for output. Conversely, if the Port Direction Register is set for output before writing to the Port Data Register, indeterminate values will be output for a while until the initial value is set in the Port Data Register.

# CHAPTER 9
# DMAC

9.1 Outline of the DMAC
9.2 DMAC Related Registers
9.3 Functional Description of the DMAC
9.4 Precautions about the DMAC

## 9.1 Outline of the DMAC

The 32171 contains a 10 channel-DMA (Direct Memory Access) Controller. It allows you to transfer data at high speed between internal peripheral I/Os, between internal RAM and internal peripheral I/O, and between internal RAMs, as requested by a software trigger or from an internal peripheral I/O.

**Table 9.1.1 Outline of the DMAC**

| Item | Description |
|---|---|
| Number of channel | 10 channels |
| Transfer request | • Software trigger<br>• Request from internal peripheral I/Os: A-D converter, multijunction timer, or serial I/O (reception completed, transmit buffer empty)<br>• Transfer operation can be cascaded between DMA channels (Note) |
| Maximum number of times transferred | 256 times |
| Transferable address space | • 64 Kbytes (address space from H'0080 0000 to H'0080 FFFF)<br>• Transfers between internal peripheral I/Os, between internal RAM and internal peripheral I/O, between internal RAMs are supported |
| Transfer data size | 16 or 8 bits |
| Transfer method | Single transfer DMA (control of the internal bus is relinquished for each transfer performed), dual-address transfer |
| Transfer mode | Single transfer mode |
| Direction of transfer | One of three modes can be selected for the source and destination:<br>• Address fixed<br>• Address incremental<br>• Ring buffered |
| Channel priority | Channel 0 > channel 1 > channel 2 > channel 3 > channel 4 > channel 5 > channel 6 > channel 7 > channel 8 > channel 9 (Priority is fixed) |
| Maximum transfer rate | 13.3 Mbytes per second (with 20 MHz internal peripheral clock) |
| Interrupt request | Group interrupt request can be generated when each transfer count register underflows. |
| Transfer area | 64 Kbytes from H'0080 0000 to H'0080 FFFF<br>(Transferable in the entire internal RAM/SFR area) |

Note: • Transfer operation can be cascaded between DMA channels as shown below.
Completion of one transfer in channel 0 starts DMA transfer in channel 1
Completion of one transfer in channel 1 starts DMA transfer in channel 2
Completion of one transfer in channel 2 starts DMA transfer in channel 0
Completion of one transfer in channel 3 starts DMA transfer in channel 4
Completion of one transfer in channel 5 starts DMA transfer in channel 6
Completion of one transfer in channel 6 starts DMA transfer in channel 7
Completion of one transfer in channel 7 starts DMA transfer in channel 5
Completion of one transfer in channel 8 starts DMA transfer in channel 9
Completion of all DMA transfers in channel 0 (transfer count register underflow) starts DMA transfer in channel 5

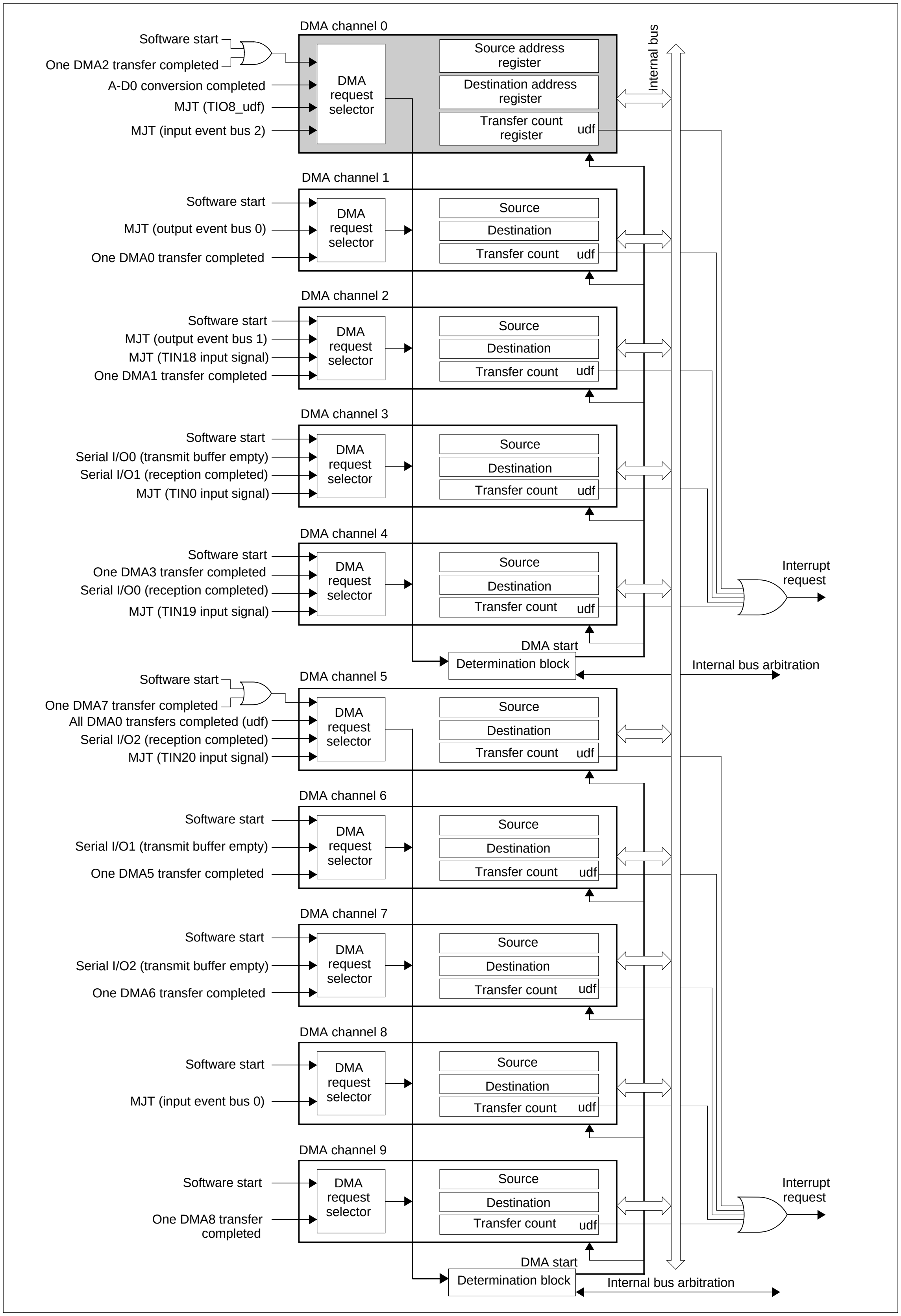


**Figure 9.1.1 Block Diagram of the DMAC**

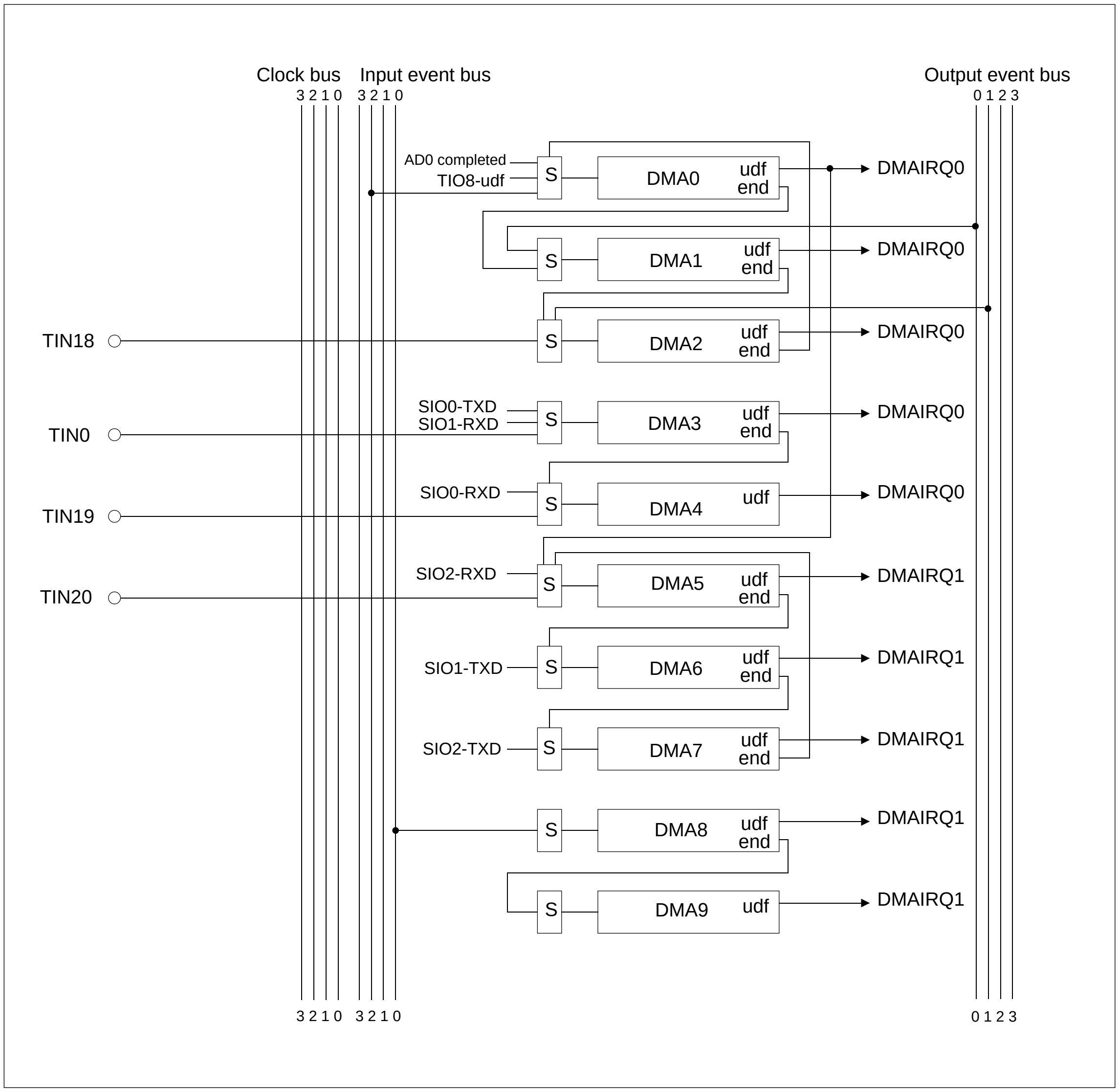


**Figure 9.1.2 Causes of DMAC Requests Connection Diagram**

## 9.2 DMAC Related Registers

The diagram below shows a memory map of DMAC related registers.

| | +0 Address | +1 Address |
|---|---|---|
| Address | D0 D7 | D8 D15 |
| H'0080 0400 | DMA0-4 Interrupt Request Status Register (DM04ITST) | DMA0-4 Interrupt Mask Register (DM04ITMK) |
| | | |
| H'0080 0408 | DMA5-9 Interrupt Request Status Register (DM59ITST) | DMA5-9 Interrupt Mask Register (DM59ITMK) |
| | | |
| H'0080 0410 | DMA0 Channel Control Register (DM0CNT) | DMA0 Transfer Count Register (DM0TCT) |
| H'0080 0412 | DMA0 Source Address Register (DM0SA) | |
| H'0080 0414 | DMA0 Destination Address Register (DM0DA) | |
| H'0080 0416 | | |
| H'0080 0418 | DMA5 Channel Control Register (DM5CNT) | DMA5 Transfer Count Register (DM5TCT) |
| H'0080 041A | DMA5 Source Address Register (DM5SA) | |
| H'0080 041C | DMA5 Destination Address Register (DM5DA) | |
| H'0080 041E | | |
| H'0080 0420 | DMA1 Channel Control Register (DM1CNT) | DMA1 Transfer Count Register (DM1TCT) |
| H'0080 0422 | DMA1 Source Address Register (DM1SA) | |
| H'0080 0424 | DMA1 Destination Address Register (DM1DA) | |
| H'0080 0426 | | |
| H'0080 0428 | DMA6 Channel Control Register (DM6CNT) | DMA6 Transfer Count Register (DM6TCT) |
| H'0080 042A | DMA6 Source Address Register (DM6SA) | |
| H'0080 042C | DMA6 Destination Address Register (DM6DA) | |
| H'0080 042E | | |
| H'0080 0430 | DMA2 Channel Control Register (DM2CNT) | DMA2 Transfer Count Register (DM2TCT) |
| H'0080 0432 | DMA2 Source Address Register (DM2SA) | |
| H'0080 0434 | DMA2 Destination Address Register (DM2DA) | |
| H'0080 0436 | | |
| H'0080 0438 | DMA7 Channel Control Register (DM7CNT) | DMA7 Transfer Count Register (DM7TCT) |
| H'0080 043A | DMA7 Source Address Register (DM7SA) | |
| H'0080 043C | DMA7 Destination Address Register (DM7DA) | |
| H'0080 043E | | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames can only be accessed in halfwords.

**Figure 9.2.1 DMAC Related Register Map (1/2)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 0440 | DMA3 Channel Control Register (DM3CNT) | DMA3 Transfer Count Register (DM3TCT) |
| H'0080 0442 | DMA3 Source Address Register (DM3SA) | |
| H'0080 0444 | DMA3 Destination Address Register (DM3DA) | |
| H'0080 0446 | | |
| H'0080 0448 | DMA8 Channel Control Register (DM8CNT) | DMA8 Transfer Count Register (DM8TCT) |
| H'0080 044A | DMA8 Source Address Register (DM8SA) | |
| H'0080 044C | DMA8 Destination Address Register (DM8DA) | |
| H'0080 044E | | |
| H'0080 0450 | DMA4 Channel Control Register (DM4CNT) | DMA4 Transfer Count Register (DM4TCT) |
| H'0080 0452 | DMA4 Source Address Register (DM4SA) | |
| H'0080 0454 | DMA4 Destination Address Register (DM4DA) | |
| H'0080 0456 | | |
| H'0080 0458 | DMA9 Channel Control Register (DM9CNT) | DMA9 Transfer Count Register (DM9TCT) |
| H'0080 045A | DMA9 Source Address Register (DM9SA) | |
| H'0080 045C | DMA9 Destination Address Register (DM9DA) | |
| H'0080 045E | | |
| H'0080 0460 | DMA0 Software Request Generation Register (DM0SRI) | |
| H'0080 0462 | DMA1 Software Request Generation Register (DM1SRI) | |
| H'0080 0464 | DMA2 Software Request Generation Register (DM2SRI) | |
| H'0080 0466 | DMA3 Software Request Generation Register (DM3SRI) | |
| H'0080 0468 | DMA4 Software Request Generation Register (DM4SRI) | |
| | | |
| H'0080 0470 | DMA5 Software Request Generation Register (DM5SRI) | |
| H'0080 0472 | DMA6 Software Request Generation Register (DM6SRI) | |
| H'0080 0474 | DMA7 Software Request Generation Register (DM7SRI) | |
| H'0080 0476 | DMA8 Software Request Generation Register (DM8SRI) | |
| H'0080 0478 | DMA9 Software Request Generation Register (DM9SRI) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames can only be accessed in halfwords.

**Figure 9.2.2 DMAC Related Register Map (2/2)**

### 9.2.1 DMA Channel Control Register

#### ■ DMA0 Channel Control Register (DM0CNT)

<Address: H'0080 0410>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL0 | TREQF0 | REQSL0 | | TENL0 | TSZSL0 | SADSL0 | DADSL0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL0<br>(Selects DMA0 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF0<br>(DMA0 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL0<br>(Selects cause of DMA0 request) | 00 : Software start or one DMA2 transfer completed<br>01 : A-D0 conversion completed<br>10 : MJT (TIO8_udf)<br>11 : MJT (input event bus 2) | ○ | ○ |
| 4 | TENL0<br>(Enables DMA0 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL0<br>(Selects DMA0 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL0<br>(Selects DMA0 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL0<br>(Selects DMA0 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

### ■ DMA1 Channel Control Register (DM1CNT)

<Address: H'0080 0420>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL1 | TREQF1 | REQSL1 | | TENL1 | TSZSL1 | SADSL1 | DADSL1 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL1<br>(Selects DMA1 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF1<br>(DMA1 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL1<br>(Selects cause of DMA1 request) | 00 : Software start<br>01 : MJT (output event bus 0)<br>10 : Use inhibited<br>11 : One DMA0 transfer completed | ○ | ○ |
| 4 | TENL1<br>(Enables DMA1 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL1<br>(Selects DMA1 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL1<br>(Selects DMA1 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL1<br>(Selects DMA1 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

## ■ DMA2 Channel Control Register (DM2CNT)

<Address: H'0080 0430>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL2 | TREQF2 | REQSL2 | | TENL2 | TSZSL2 | SADSL2 | DADSL2 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL2<br>(Selects DMA2 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF2<br>(DMA2 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL2<br>(Selects cause of DMA2 request) | 00 : Software start<br>01 : MJT (output event bus 1)<br>10 : MJT (TIN18 input signal)<br>11 : One DMA1 transfer completed | ○ | ○ |
| 4 | TENL2<br>(Enables DMA2 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL2<br>(Selects DMA2 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL2<br>(Selects DMA2 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL2<br>(Selects DMA2 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

## ■ DMA3 Channel Control Register (DM3CNT)

<Address: H'0080 0440>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL3 | TREQF3 | REQSL3 | | TENL3 | TSZSL3 | SADSL3 | DADSL3 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL3<br>(Selects DMA3 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF3<br>(DMA3 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL3<br>(Selects cause of DMA3 request) | 00 : Software start<br>01 : Serial I/O0 (transmit buffer empty)<br>10 : Serial I/O1 (reception completed)<br>11 : MJT (TIN0 input signal) | ○ | ○ |
| 4 | TENL3<br>(Enables DMA3 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL3<br>(Selects DMA3 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL3<br>(Selects DMA3 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL3<br>(Selects DMA3 destination<br>address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

## ■ DMA4 Channel Control Register (DM4CNT)

<Address: H'0080 0450>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| **MDSEL4** | **TREQF4** | **REQSL4** | | **TENL4** | **TSZSL4** | **SADSL4** | **DADSL4** |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL4<br>(Selects DMA4 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF4<br>(DMA4 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL4<br>(Selects cause of DMA4 request) | 00 : Software start<br>01 : One DMA3 transfer completed<br>10 : Serial I/O0 (reception completed)<br>11 : MJT (TIN19 input signal) | ○ | ○ |
| 4 | TENL4<br>(Enables DMA4 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL4<br>(Selects DMA4 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL4<br>(Selects DMA4 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL4<br>(Selects DMA4 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

### ■ DMA5 Channel Control Register (DM5CNT)

<Address: H'0080 0418>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL5 | TREQF5 | REQSL5 | | TENL5 | TSZSL5 | SADSL5 | DADSL5 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL5<br>(Selects DMA5 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF5<br>(DMA5 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL5<br>(Selects cause of DMA5 request) | 00 : Software start or one DMA7 transfer completed<br>01 : All DMA0 transfers completed<br>10 : Serial I/O2 (reception completed)<br>11 : MJT (TIN20 input signal) | ○ | ○ |
| 4 | TENL5<br>(Enables DMA5 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL5<br>(Selects DMA5 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL5<br>(Selects DMA5 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL5<br>(Selects DMA5 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

## ■ DMA6 Channel Control Register (DM6CNT)

<Address: H'0080 0428>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL6 | TREQF6 | REQSL6 | | TENL6 | TSZSL6 | SADSL6 | DADSL6 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL6<br>(Selects DMA6 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF6<br>(DMA6 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL6<br>(Selects cause of DMA6 request) | 00 : Software start<br>01 : Serial I/O1 (transmit buffer empty)<br>10 : Use inhibited<br>11 : One DMA5 transfer completed | ○ | ○ |
| 4 | TENL6<br>(Enables DMA6 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL6<br>(Selects DMA6 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL6<br>(Selects DMA6 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL6<br>(Selects DMA6 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

## ■ DMA7 Channel Control Register (DM7CNT)

<Address: H'0080 0438>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL7 | TREQF7 | REQSL7 | | TENL7 | TSZSL7 | SADSL7 | DADSL7 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL7<br>(Selects DMA7 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF7<br>(DMA7 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL7<br>(Selects cause of DMA7 request) | 00 : Software start<br>01 : Serial I/O2 (transmit buffer empty)<br>10 : Use inhibited<br>11 : One DMA6 transfer completed | ○ | ○ |
| 4 | TENL7<br>(Enables DMA7 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL7<br>(Selects DMA7 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL7<br>(Selects DMA7 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL7<br>(Selects DMA7 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

### ■ DMA8 Channel Control Register (DM8CNT)

<Address: H'0080 0448>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL8 | TREQF8 | REQSL8 | | TENL8 | TSZSL8 | SADSL8 | DADSL8 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL8<br>(Selects DMA8 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF8<br>(DMA8 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL8<br>(Selects cause of DMA8 request) | 00 : Software start<br>01 : MJT (input event bus 0)<br>10 : Use inhibited<br>11 : Use inhibited | ○ | ○ |
| 4 | TENL8<br>(Enables DMA8 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL8<br>(Selects DMA8 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL8<br>(Selects DMA8 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL8<br>(Selects DMA8 destination address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

### ■ DMA9 Channel Control Register (DM9CNT)

<Address: H'0080 0458>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| MDSEL9 | TREQF9 | REQSL9 | | TENL9 | TSZSL9 | SADSL9 | DADSL9 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | MDSEL9<br>(Selects DMA9 transfer mode) | 0 : Normal mode<br>1 : Ring buffer mode | ○ | ○ |
| 1 | TREQF9<br>(DMA9 transfer request flag) | 0 : Not requested<br>1 : Requested | ○ | △ |
| 2, 3 | REQSL9<br>(Selects cause of DMA9 request) | 00 : Software start<br>01 : Use inhibited<br>10 : Use inhibited<br>11 : One DMA8 transfer completed | ○ | ○ |
| 4 | TENL9<br>(Enables DMA9 transfer) | 0 : Disables transfer<br>1 : Enables transfer | ○ | ○ |
| 5 | TSZSL9<br>(Selects DMA7 transfer size) | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| 6 | SADSL9<br>(Selects DMA9 source address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |
| 7 | DADSL9<br>(Selects DMA9 destination<br>address direction) | 0 : Fixed<br>1 : Incremental | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

The DMA Channel Control Register consists of bits to select DMA transfer mode in each channel, set DMA transfer request flag, and the bits to select the cause of DMA request, enable DMA transfer, and set the transfer size and the source/destination address directions.

### (1) MDSELn (DMAn transfer mode select) bit (D0)

This bit when in single transfer mode selects normal mode or ring buffer mode. Normal mode is selected by setting this bit to 0 or ring buffer mode is selected by setting it to 1.

In ring buffer mode, transfer begins from the transfer start address and after performing transfers 32 times, control is recycled back to the transfer start address, from which transfer operation is repeated. In this case, the Transfer Count Register counts in free-run mode during which time transfer operation is continued until the transfer enable bit is reset to 0 (to disable transfer). No interrupt is generated at completion of DMA transfer.

### (2) TREQFn (DMAn transfer request flag) bit (D1)

This flag is set to 1 when a DMA transfer request occurs. Reading this flag helps to know DMA transfer requests in each channel.

The generated DMA request is cleared by writing a 0 to this bit. If you write a 1, the value you wrote is ignored and the bit retains its previous value. If a new DMA transfer request is generated for a channel whose DMA transfer request flag has already been set to 1, the next DMA transfer request is not accepted until the transfer under way in that channel is completed.

### (3) REQSLn (cause of DMAn request select) bits (D2, D3)

These bits select the cause of DMA request in each DMA channel.

### (4) TENLn (DMAn transfer enable) bit (D4)

Transfer is enabled by setting this bit to 1, so that the channel is ready for DMA transfer. Conversely, transfer is disabled by setting this bit to 0. However, if a transfer request has already been accepted, transfer in that channel is not disabled until after the requested transfer is completed.

### (5) TSZSLn (DMAn transfer size select) bit (D5)

This bit selects the number of bits to be transferred in one DMA transfer operation (unit of one transfer). The unit of one transfer is 16 bits when TSZSL = 0 or 8 bits when TSZSL = 1.

### (6) SADSLn (DMAn source address direction select) bit (D6)

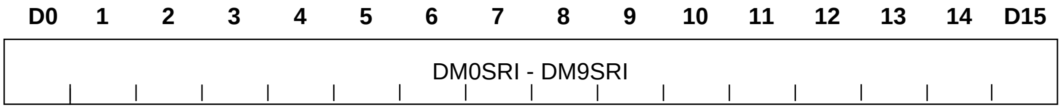
This bit selects the direction in which the source address changes as transfer proceeds. This mode can be selected from two choices: address fixed or address incremental.

### (7) DADSLn (DAMn destination address direction select) bit (D7)

This bit selects the direction in which the destination address changes as transfer proceeds. This mode can be selected from two choices: address fixed or address incremental.

### 9.2.2 DMA Software Request Generation Registers

| | |
|---|---|
| ■ **DMA0 Software Request Generation Register (DM0SRI)** | <Address: H'0080 0460> |
| ■ **DMA1 Software Request Generation Register (DM1SRI)** | <Address: H'0080 0462> |
| ■ **DMA2 Software Request Generation Register (DM2SRI)** | <Address: H'0080 0464> |
| ■ **DMA3 Software Request Generation Register (DM3SRI)** | <Address: H'0080 0466> |
| ■ **DMA4 Software Request Generation Register (DM4SRI)** | <Address: H'0080 0468> |
| ■ **DMA5 Software Request Generation Register (DM5SRI)** | <Address: H'0080 0470> |
| ■ **DMA6 Software Request Generation Register (DM6SRI)** | <Address: H'0080 0472> |
| ■ **DMA7 Software Request Generation Register (DM7SRI)** | <Address: H'0080 0474> |
| ■ **DMA8 Software Request Generation Register (DM8SRI)** | <Address: H'0080 0476> |
| ■ **DMA9 Software Request Generation Register (DM9SRI)** | <Address: H'0080 0478> |



<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 15 | DM0SRI - DM9SRI<br>(Generates DMA software request) | DMA transfer request is generated<br>by writing any data | ? | ○ |

Note: • This register can be accessed in either bytes or halfwords.

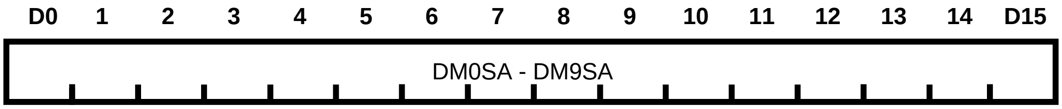
The DMA Software Request Generation Register is used to generate DMA transfer requests in software. A DMA transfer request can be generated by writing any data to this register when "Software start" has been selected for the cause of DMA request.

**DM0SRI - DM9SRI (DMA software request generate) bit**

A software DMA transfer request is generated by writing any data to this register in halfword (16 bits) or in byte (8 bits) beginning with an even or odd address when "Software" is selected as the cause of DMA transfer request (by setting the DMA Channel Control Register D2, D3 bits to "00").

### 9.2.3 DMA Source Address Registers

| | |
|---|---|
| ■ **DMA0 Source Address Register (DM0SA)** | <Address: H'0080 0412> |
| ■ **DMA1 Source Address Register (DM1SA)** | <Address: H'0080 0422> |
| ■ **DMA2 Source Address Register (DM2SA)** | <Address: H'0080 0432> |
| ■ **DMA3 Source Address Register (DM3SA)** | <Address: H'0080 0442> |
| ■ **DMA4 Source Address Register (DM4SA)** | <Address: H'0080 0452> |
| ■ **DMA5 Source Address Register (DM5SA)** | <Address: H'0080 041A> |
| ■ **DMA6 Source Address Register (DM6SA)** | <Address: H'0080 042A> |
| ■ **DMA7 Source Address Register (DM7SA)** | <Address: H'0080 043A> |
| ■ **DMA8 Source Address Register (DM8SA)** | <Address: H'0080 044A> |
| ■ **DMA9 Source Address Register (DM9SA)** | <Address: H'0080 045A> |



<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 15 | DM0SA - DM9SA<br>(DMA source address) | A16-A31 of the source address<br>(A0-A15 are fixed to H'0080) | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The DMA Source Address Register is used to set the source address of DMA transfer in such a way that D0 corresponds to A16, and D15 corresponds to A31. Because this register is comprised of a current register, the value you get by reading this register is always the current value.
When DMA transfer finishes (at which the Transfer Count Register underflows), the value in this register if "Address fixed" is selected, is the same source address that was set in it before DMA transfer began; if "Address incremental" is selected, the value in this register is the last transfer address + 1 (for 8-bit transfer) or the last transfer address + 2 (for 16-bit transfer).

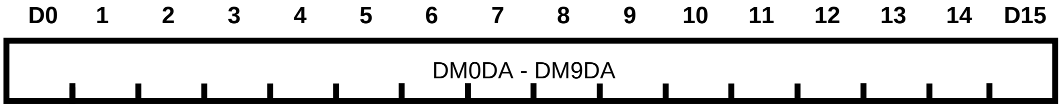
Make sure the DMA Source Address Register is always accessed in halfwords (16 bits) beginning with an even address. If accessed in bytes, the value read from this register is indeterminate.

**DM0SA-DM9SA (A16-A31 of the source address)**

By setting this register, specify the source address of DMA transfer in internal I/O space ranging from H'0080 0000 to H'0080 FFFF or in the RAM space.
The 16 high-order bits of the source address (A0-A15) are always fixed to H'0080. Use this register to set the 16 low-order bits of the source address (with D0 corresponding to A16, and D15 corresponding to A31).

### 9.2.4 DMA Destination Address Registers

| | |
|---|---|
| ■ **DMA0 Destination Address Register (DM0DA)** | <Address: H'0080 0414> |
| ■ **DMA1 Destination Address Register (DM1DA)** | <Address: H'0080 0424> |
| ■ **DMA2 Destination Address Register (DM2DA)** | <Address: H'0080 0434> |
| ■ **DMA3 Destination Address Register (DM3DA)** | <Address: H'0080 0444> |
| ■ **DMA4 Destination Address Register (DM4DA)** | <Address: H'0080 0454> |
| ■ **DMA5 Destination Address Register (DM5DA)** | <Address: H'0080 041C> |
| ■ **DMA6 Destination Address Register (DM6DA)** | <Address: H'0080 042C> |
| ■ **DMA7 Destination Address Register (DM7DA)** | <Address: H'0080 043C> |
| ■ **DMA8 Destination Address Register (DM8DA)** | <Address: H'0080 044C> |
| ■ **DMA9 Destination Address Register (DM9DA)** | <Address: H'0080 045C> |



<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 15 | DM0DA - DM9DA<br>(DMA destination address) | A16-A31 of the destination address<br>(A0-A15 are fixed to H'0080) | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The DMA Destination Address Register is used to set the destination address of DMA transfer in such a way that D0 corresponds to A16, and D15 corresponds to A31. Because access to this register is comprised of a current register, the value you get by reading this register is always the current value.

When DMA transfer finishes (at which the Transfer Count Register underflows), the value in this register if "Address fixed" is selected, is the same destination address that was set in it before DMA transfer began; if "Address incremental" is selected, the value in this register is the last transfer address + 1 (for 8-bit transfer) or the last transfer address + 2 (for 16-bit transfer).

Make sure the DMA Destination Address Register is always accessed in halfwords (16 bits) beginning with an even address. If accessed in bytes, the value read from this register is indeterminate.

**DM0DA-DM9DA (A16-A31 of the destination address)**

By setting this register, specify the destination address of DMA transfer in internal I/O space ranging from H'0080 0000 to H'0080 FFFF or in the RAM space.

The 16 high-order bits of the destination address (A0-A15) are always fixed to H'0080. Use this register to set the 16 low-order bits of the destination address (with D0 corresponding to A16, and D15 corresponding to A31).

### 9.2.5 DMA Transfer Count Registers

| | |
|---|---|
| ■ **DMA0 Transfer Count Register (DM0TCT)** | <Address: H'0080 0411> |
| ■ **DMA1 Transfer Count Register (DM1TCT)** | <Address: H'0080 0421> |
| ■ **DMA2 Transfer Count Register (DM2TCT)** | <Address: H'0080 0431> |
| ■ **DMA3 Transfer Count Register (DM3TCT)** | <Address: H'0080 0441> |
| ■ **DMA4 Transfer Count Register (DM4TCT)** | <Address: H'0080 0451> |
| ■ **DMA5 Transfer Count Register (DM5TCT)** | <Address: H'0080 0419> |
| ■ **DMA6 Transfer Count Register (DM6TCT)** | <Address: H'0080 0429> |
| ■ **DMA7 Transfer Count Register (DM7TCT)** | <Address: H'0080 0439> |
| ■ **DMA8 Transfer Count Register (DM8TCT)** | <Address: H'0080 0449> |
| ■ **DMA9 Transfer Count Register (DM9TCT)** | <Address: H'0080 0459> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| DM0TCT - DM9TCT | | | | | | | |

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 15 | DM0TCT - DM9TCT<br>(DMA transfer count) | DMA transfer count<br>(ignored during 32-channel ring buffer mode) | ○ | ○ |

The DMA Transfer Count Register is used to set the number of times data is transferred in each channel. However, the value in this register is ignored during ring buffer mode.

The transfer count is the (value set in the transfer count register + 1). Because the DMA Transfer Count Register is comprised of a current register, the value you get by reading this register is always the current value. (However, if you read this register in a cycle immediately after transfer, the value you get is the value that was in the count register before the transfer began.) When transfer finishes, this count register underflows, so that the read value you get is H'FF.

If any cascaded channel exists, each time one DMA transfer (byte or halfword) is completed or when all transfers are completed (at which the transfer count register underflows), transfer in the cascaded channel starts.

### 9.2.6 DMA Interrupt Request Status Registers

■ **DMA0-4 Interrupt Request Status Register (DM04ITST)** <Address: H'0080 0400>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | | DMITST4 | DMITST3 | DMITST2 | DMITST1 | DMITST0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 2 | No functions assigned | | 0 | — |
| 3 | DMITST4 (DMA4 interrupt request status) | 0 : No interrupt request<br>1 : Interrupt requested | ○ | △ |
| 4 | DMITST3 (DMA3 interrupt request status) | | | |
| 5 | DMITST2 (DMA2 interrupt request status) | | | |
| 6 | DMITST1 (DMA1 interrupt request status) | | | |
| 7 | DMITST0 (DMA0 interrupt request status) | | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

The DMA0-4 Interrupt Request Status Register lets you know the status of interrupt requests in channels 0-4. If the DMAn interrupt request status bit (n = 0 to 4) is set to 1, it means that a DMAn interrupt request in the corresponding channel has been generated.

**DMITSTn (DMAn interrupt request status) bit (n = 0 to 4)**

[Setting the DMAn interrupt request status bit]

This bit can only be set in hardware, and cannot be set in software.

[Clearing the DMAn interrupt request status bit]

This bit is cleared by writing a 0 in software.

Note: • The DMAn interrupt request status bit cannot be cleared by writing a 0 to the "IREQ bit" of the DMA Interrupt Control Register(IDMA04CR) that the interrupt controller has.

When writing to the DMA0-4 Interrupt Request Status Register, be sure to set the bits you want to clear to 0 and all other bits to 1. The bits which are thus set to 1 are unaffected by writing in software, and retain the value they had before you wrote.

## ■ DMA5-9 Interrupt Request Status Register (DM59ITST) <Address: H'0080 0408>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | | DMITST9 | DMITST8 | DMITST7 | DMITST6 | DMITST5 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 2 | No functions assigned | | 0 | — |
| 3 | DMITST9 (DMA9 interrupt request status) | 0 : No interrupt request | ○ | △ |
| 4 | DMITST8 (DMA8 interrupt request status) | 1 : Interrupt requested | | |
| 5 | DMITST7 (DMA7 interrupt request status) | | | |
| 6 | DMITST6 (DMA6 interrupt request status) | | | |
| 7 | DMITST5 (DMA5 interrupt request status) | | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

The DMA5-9 Interrupt Request Status Register lets you know the status of interrupt requests in channels 5-9. If the DMAn interrupt request status bit (n = 5 to 9) is set to 1, it means that a DMAn interrupt request in the corresponding channel has been generated.

### DMITSTn (DMAn interrupt request status) bit (n = 5 to 9)

[Setting the DMAn interrupt request status bit]

This bit can only be set in hardware, and cannot be set in software.

[Clearing the DMAn interrupt request status bit]

This bit is cleared by writing a 0 in software.

Note: • The DMAn interrupt request status bit cannot be cleared by writing a 0 to the "IREQ bit" of the DMA Interrupt Control Register(IDMA59CR) that the interrupt controller has.

When writing to the DMA5-9 Interrupt Request Status Register, be sure to set the bits you want to clear to 0 and all other bits to 1. The bits which are thus set to 1 are unaffected by writing in software, and retain the value they had before you wrote.

### 9.2.7 DMA Interrupt Mask Registers

#### ■ DMA0-4 Interrupt Mask Register (DM04ITMK)

<Address: H'0080 0401>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| ▨ | ▨ | ▨ | DMITMK4 | DMITMK3 | DMITMK2 | DMITMK1 | DMITMK0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | No functions assigned | | 0 | — |
| 11 | DMITMK4 (DMA4 interrupt request mask) | 0 : Enables interrupt request<br>1 : Masks (disables) interrupt request | ○ | ○ |
| 12 | DMITMK3 (DMA3 interrupt request mask) | | | |
| 13 | DMITMK2 (DMA2 interrupt request mask) | | | |
| 14 | DMITMK1 (DMA1 interrupt request mask) | | | |
| 15 | DMITMK0 (DMA0 interrupt request mask) | | | |

The DMA0-4 Interrupt Mask Register is used to mask interrupt requests in DMA channels 0-4.

**DMITMKn (DMAn interrupt request mask) bit (n = 0 to 4)**

DMAn interrupt request is masked by setting the DMAn interrupt request mask bit to 1. However, when an interrupt request is generated, the DMAn interrupt request status bit is always set to 1 irrespective of the contents of this register.

## ■ DMA5-9 Interrupt Mask Register (DM59ITMK)

<Address: H'0080 0409>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  | DMITMK9 | DMITMK8 | DMITMK7 | DMITMK6 | DMITMK5 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | No functions assigned |  | 0 | — |
| 11 | DMITMK9 (DMA9 interrupt request mask) | 0 : Enables interrupt request<br>1 : Masks (disables) interrupt request | ○ | ○ |
| 12 | DMITMK8 (DMA8 interrupt request mask) |  |  |  |
| 13 | DMITMK7 (DMA7 interrupt request mask) |  |  |  |
| 14 | DMITMK6 (DMA6 interrupt request mask) |  |  |  |
| 15 | DMITMK5 (DMA5 interrupt request mask) |  |  |  |

The DMA5-9 Interrupt Mask Register is used to mask interrupt requests in DMA channels 5-9.

### DMITMKn (DMAn interrupt request mask) bit (n = 5 to 9)

DMAn interrupt request is masked by setting the DMAn interrupt request mask bit to 1. However, when an interrupt request is generated, the DMAn interrupt request status bit is always set to 1 irrespective of the contents of this register.

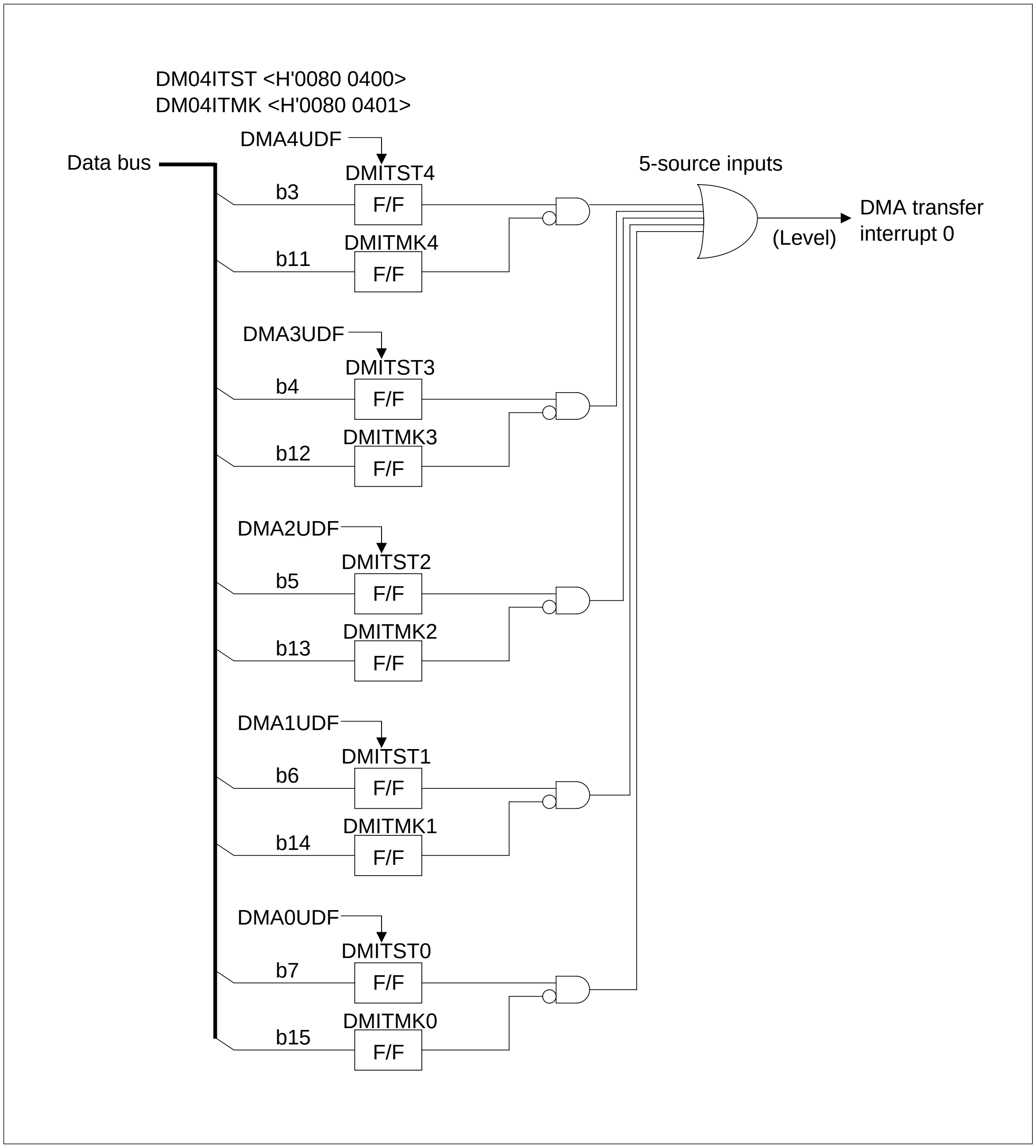


**Figure 9.2.3 Block Diagram of DMA Transfer Interrupt 0**

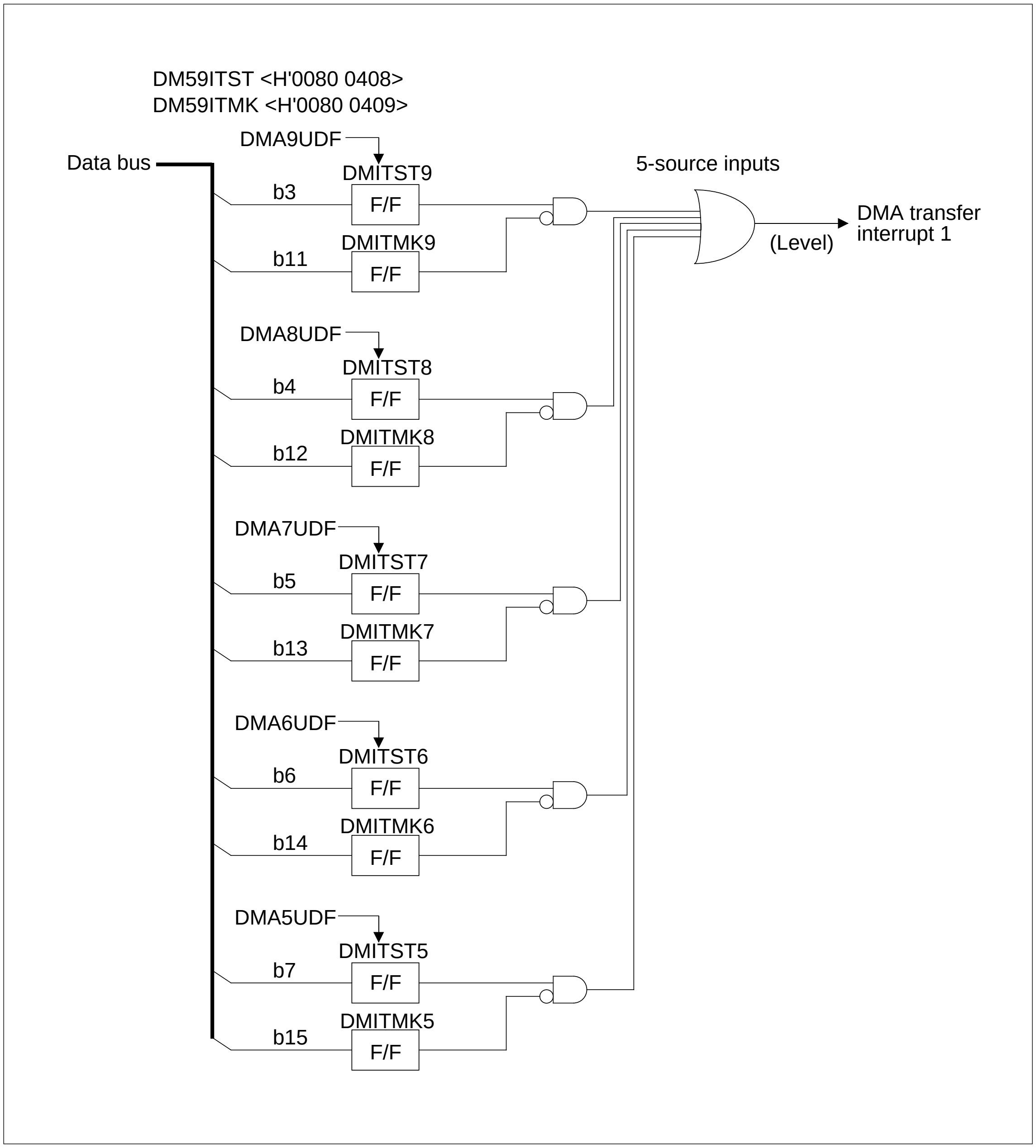


**Figure 9.2.4 Block Diagram of DMA Transfer Interrupt 1**

## 9.3 Functional Description of the DMAC

### 9.3.1 Cause of DMA Request

For each DMA channel (channels 0 to 9), DMA transfer can be requested from multiple sources. There are various causes of DMA transfer, so that DMA transfer can be started by a request from internal peripheral I/O, started in software by a program, or can be started upon completion of one transfer or all transfers in a DMA channel (cascade mode).

The cause of DMA request is selected using the cause of request select bit provided for each channel, REQSLn (DMAn Channel Control Register bits D2, D3). The table below lists the causes of DMA requests in each channel.

**Table 9.3.1 Causes of DMA Requests in DMA0 and Generation Timings**

| REQSL0 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start<br>or one DMA2 transfer completed | When any data is written to DMA0 Software Request Generation Register (software start)<br>or one DMA2 transfer is completed (cascade mode) |
| 0 | 1 | A-D0 conversion completed | When A-D0 conversion is completed |
| 1 | 0 | MJT (TIO8_udf) | When MJT TIO8 underflow occurs |
| 1 | 1 | MJT (input event bus 2) | When MJT's input event bus 2 signal is generated |

**Table 9.3.2 Causes of DMA Requests in DMA1 and Generation Timings**

| REQSL1 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA1 Software Request Generation Register |
| 0 | 1 | MJT (output event bus 0) | When MJT's output event bus 0 signal is generated |
| 1 | 0 | None (Use inhibited) | – |
| 1 | 1 | One DMA0 transfer completed | When one DMA0 transfer is completed (cascade mode) |

Table 9.3.3 Causes of DMA Requests in DMA2 and Generation Timings

| REQSL2 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA2 Software Request Generation Register |
| 0 | 1 | MJT (output event bus 1) | When MJT's output event bus 1 signal is generated |
| 1 | 0 | MJT (TIN18 input signal) | When MJT's TIN18 input signal is generated |
| 1 | 1 | One DMA1 transfer completed | When one DMA1 transfer is completed (cascade mode) |

Table 9.3.4 Causes of DMA Requests in DMA3 and Generation Timings

| REQSL3 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA3 Software Request Generation Register |
| 0 | 1 | Serial I/O0 (transmit buffer empty) | When serial I/O0 transmit buffer is emptied |
| 1 | 0 | Serial I/O1 (reception completed) | When serial I/O1 reception is completed |
| 1 | 1 | MJT (TIN0 input signal) | When MJT's TIN0 input signal is generated |

Table 9.3.5 Causes of DMA Requests in DMA4 and Generation Timings

| REQSL4 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA4 Software Request Generation Register |
| 0 | 1 | One DMA3 transfer completed | When one DMA3 transfer is completed (cascade mode) |
| 1 | 0 | Serial I/O0 (reception completed) | When serial I/O0 reception is completed |
| 1 | 1 | MJT (TIN19 input signal) | When MJT's TIN19 input signal is generated |

Table 9.3.6 Causes of DMA Requests in DMA5 and Generation Timings

| REQSL5 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start<br>or one DMA7 transfer completed | When any data is written to DMA5 Software Request Generation Register or one DMA7 transfer is completed (cascade mode) |
| 0 | 1 | All DMA0 transfers completed | When all DMA0 transfers are completed (cascade mode) |
| 1 | 0 | Serial I/O2 (reception completed) | When serial I/O2 reception is completed |
| 1 | 1 | MJT (TIN20 input signal) | When MJT's TIN20 input signal is generated |

Table 9.3.7 Causes of DMA Requests in DMA6 and Generation Timings

| REQSL6 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA6 Software Request Generation Register |
| 0 | 1 | Serial I/O1 (transmit buffer empty) | When serial I/O1 transmit buffer is emptied |
| 1 | 0 | None (Use inhibited) | – |
| 1 | 1 | One DMA5 transfer completed | When one DMA5 transfer is completed (cascade mode) |

Table 9.3.8 Causes of DMA Requests in DMA7 and Generation Timings

| REQSL7 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA7 Software Request Generation Register |
| 0 | 1 | Serial I/O2 (transmit buffer empty) | When serial I/O2 transmit buffer is emptied |
| 1 | 0 | None (Use inhibited) | – |
| 1 | 1 | One DMA6 transfer completed | When one DMA6 transfer is completed (cascade mode) |

Table 9.3.9 Causes of DMA Requests in DMA8 and Generation Timings

| REQSL8 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA8 Software Request Generation Register |
| 0 | 1 | MJT (input event bus 0) | When MJT's input event bus 0 signal is generated |
| 1 | 0 | None (Use inhibited) | – |
| 1 | 1 | None (Use inhibited) | – |

Table 9.3.10 Causes of DMA Requests in DMA9 and Generation Timings

| REQSL9 | | Cause of DMA Request | DMA Request Generation Timing |
|---|---|---|---|
| 0 | 0 | Software start | When any data is written to DMA9 Software Request Generation Register |
| 0 | 1 | None (Use inhibited) | – |
| 1 | 0 | None (Use inhibited) | – |
| 1 | 1 | One DMA8 transfer completed | When one DMA8 transfer is completed (cascade mode) |

### 9.3.2 DMA Transfer Processing Procedure

Shown below is an example of how to control DMA transfer in cases when performing transfer in DMA channel 0.

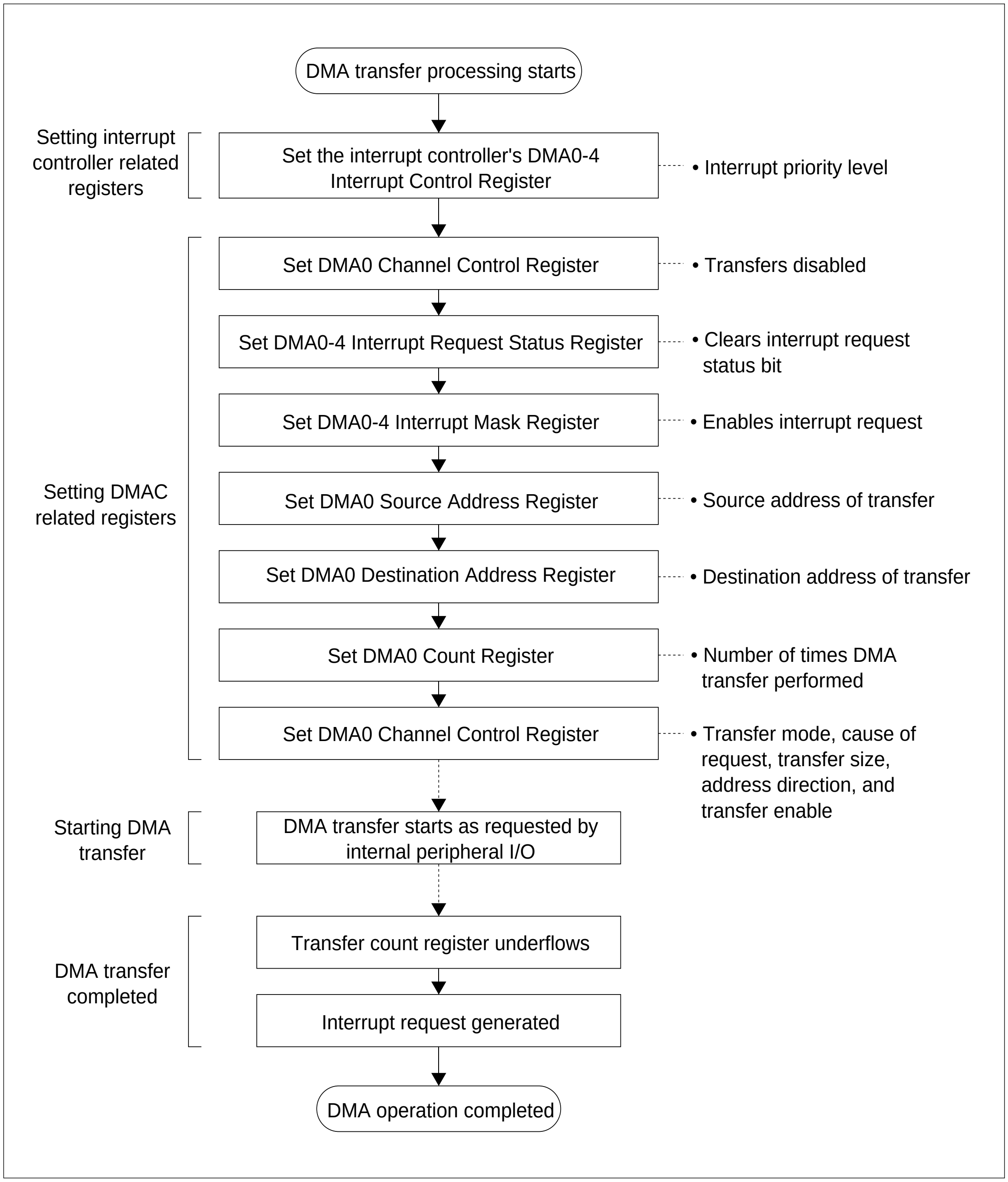

DMA transfer processing starts
Setting interrupt controller related registers
Set the interrupt controller's DMA0-4 Interrupt Control Register
• Interrupt priority level
Set DMA0 Channel Control Register
• Transfers disabled
Set DMA0-4 Interrupt Request Status Register
• Clears interrupt request status bit
Set DMA0-4 Interrupt Mask Register
• Enables interrupt request
Setting DMAC related registers
Set DMA0 Source Address Register
• Source address of transfer
Set DMA0 Destination Address Register
• Destination address of transfer
Set DMA0 Count Register
• Number of times DMA transfer performed
Set DMA0 Channel Control Register
• Transfer mode, cause of request, transfer size, address direction, and transfer enable
Starting DMA transfer
DMA transfer starts as requested by internal peripheral I/O
DMA transfer completed
Transfer count register underflows
Interrupt request generated
DMA operation completed

**Figure 9.3.1 Example of a DMA Transfer Processing Procedure**

### 9.3.3 Starting DMA

Use the REQSL (cause of DMA request select) bit to set the cause of DMA request. To enable DMA, set the TENL (DMA transfer enable) bit to 1. DMA transfer begins when the specified cause of DMA request becomes effective after setting the TENL (DMA transfer enable) bit to 1.

### 9.3.4 Channel Priority

Channel 0 has the highest priority. The priority of this and other channels is shown below.

Channel 0 > channel 1 > channel 2 > channel 3 > channel 4 > channel 5 > channel 6 > channel 7 > channel 8 > channel 9

This order of priority is fixed and cannot be changed. Among channels for which DMA transfers are requested, the channel that has the highest priority is selected. Channel selection is made every transfer cycle (one DMA bus cycle consisting of three machine cycles).

### 9.3.5 Gaining and Releasing Control of the Internal Bus

For any channel, control of the internal bus is gained and released in "single transfer DMA" mode. In single transfer DMA, the DMA gains control of the internal bus when DMA transfer request is accepted and after executing one DMA transfer (consisting of one read cycle + one write cycle of internal peripheral clock), returns bus control to the CPU. The diagram below shows DMA operation in single transfer DMA.

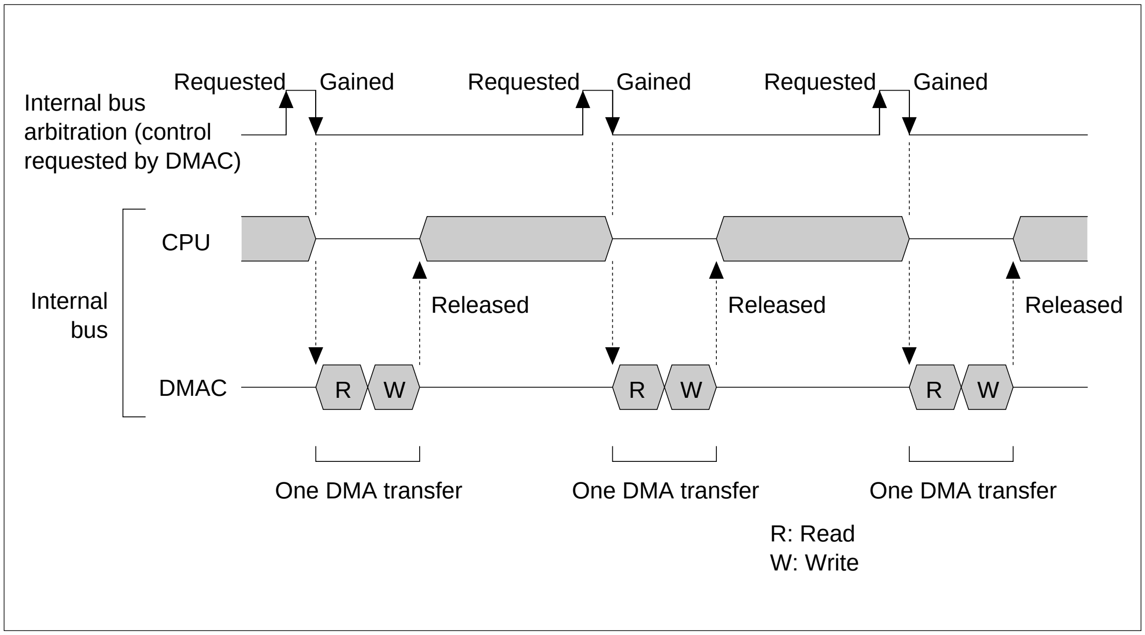


**Figure 9.3.2 Gaining and Releasing Control of the Internal Bus**

### 9.3.6 Transfer Units

Use the TSZSL (DMA transfer size select) bit to set for each channel the number of bits (8 or 16 bits) to be transferred in one DMA transfer.

### 9.3.7 Transfer Counts

Use the DMA Transfer Count Register to set transfer counts for each channel. Transfer can be performed up to 256 times. The value of the DMA Transfer Count Register is decremented by one each time one transfer unit is transferred. In ring buffer mode, the DMA Transfer Count Register operates in free-run mode, with the value set in it ignored.

### 9.3.8 Address Space

The address space in which data can be transferred by DMA is the internal peripheral I/O or 64 Kbytes of RAM space (H'0080 0000 through H'0080 FFFF) for either source or destination. To set the source and destination addresses in each channel, use the DMA Source Address Register and DMA Destination Address Register.

### 9.3.9 Transfer Operation

**(1) Dual-address transfer**

Irrespective of the size of transfer unit, data is transferred in two bus cycles, one for source read access and one for destination write access. (The transfer data is temporarily taken into the DMA's internal temporary register.)

**(2) Bus protocol and bus timing**

Because the bus interface is shared with the CPU, the same applies to both bus protocol and bus timing as in peripheral module access from the CPU.

**(3) Transfer rate**

The maximum transfer rate is calculated using the equation below:

$$\text{Maximum transfer rate [bytes/second]} = 2 \text{ bytes} \times \frac{1}{1 / f\,(\text{BCLK}) \times 3 \text{ cycles}}$$

### (4) Address count direction and address changes

The direction in which the source and destination addresses are counted as transfer proceeds ("Address fixed" or "Address incremental") is set for each channel using the SADSL (source address direction select) and DADSL (destination address select) bits.

When the transfer size is 16 bits, the address is incremented by two for each DMA transfer performed; when the transfer size is 8 bits, the address is incremented by one.

**Table 9.3.11 Address Count Direction and Address Changes**

| Address Count Direction | Transfer Unit | Address Change for One DMA |
|---|---|---|
| Address fixed | 8 bits | 0 |
| | 16 bits | 0 |
| Address incremental | 8 bits | +1 |
| | 16 bits | +2 |

### (5) Transfer count value

The transfer count value is decremented by one at a time irrespective of the size of transfer unit (8 or 16 bits).

### (6) Transfer byte positions

When the transfer unit = 8 bits, the LSB of the address register is effective for both source and destination. (Therefore, in addition to data transfers between even addresses or between odd addresses, data may be transferred from even address to odd address, or from odd address to even address.)

When the transfer unit = 16 bits, the LSB of the address register (D15 of the address register) is ignored, and data are always transferred in two bytes aligned to the 16-bit bus.

The diagram below shows the valid transfer byte positions.

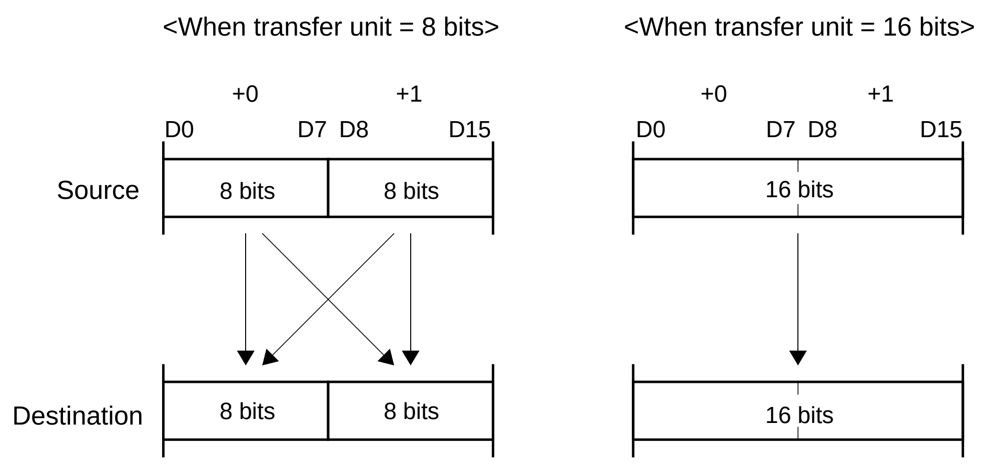


Figure 9.3.3 Transfer Byte Positions

### (7) Ring buffer mode

When ring buffer mode is selected, transfer begins from the transfer start address and after performing transfers 32 times, control is recycled back to the transfer start address, from which transfer operation is repeated. In this case, however, the five low-order bits of the ring buffer start address must always be B'00000. The address increment operation in ring buffer mode is described below.

① When the transfer unit = 8 bits

The 27 high-order bits of the transfer start address are fixed, and the five low-order bits are incremented by one at a time. When as transfer proceeds the five low-order bits reach B'11111, they are recycled to B'00000 by the next increment operation, thus returning to the start address again.

② When the transfer unit = 16 bits

The 26 high-order bits of the transfer start address are fixed, and the six low-order bits are incremented by two at a time. When as transfer proceeds the six low-order bits reach B'111110, they are recycled to B'000000 by the next increment operation, thus returning to the start address again.

When the source address has been set to be incremented, it is the source address that recycles to the start address; when the destination address has been set to be incremented, it is the destination address that recycles to the start address. If both source and destination addresses have been set to be incremented, both addresses recycle to the start address. However, the start address on either side must have their five low-order bits initially being B'00000.

During ring buffer mode, the transfer count register is ignored. Also, once DMA operation starts, the counter operates in free-run mode, and the transfer continues until the transfer enable bit is cleared to (to disable transfer).

| <When transfer unit = 8 bits> | | <When transfer unit = 16 bits> | |
|---|---|---|---|
| Transfer count | Transfer address | Transfer count | Transfer address |
| 1 | H'0080 1000 | 1 | H'0080 1000 |
| 2 | H'0080 1001 | 2 | H'0080 1002 |
| 3 | H'0080 1002 | 3 | H'0080 1004 |
| \| | \| | \| | \| |
| 31 | H'0080 101E | 31 | H'0080 103C |
| 32 | H'0080 101F | 32 | H'0080 103E |
| ↓ | ↓ | ↓ | ↓ |
| 1 | H'0080 1000 | 1 | H'0080 1000 |
| 2 | H'0080 1001 | 2 | H'0080 1002 |
| \| | \| | \| | \| |

**Figure 9.3.4 Example of Address Increment Operation in 32-Channel Ring Buffer Mode**

### 9.3.10 End of DMA and Interrupt

In normal mode, DMA transfer is terminated when the transfer count register underflows. When transfer finishes, the transfer enable bit is cleared to 0 and transfers are thereby disabled. Also, an interrupt request is generated at completion of transfer. However, this interrupt is not generated for channels where interrupt requests have been masked by the DMA Interrupt Mask Register.

During ring buffer mode, the transfer count register operates in free-run mode, and transfer continues until the transfer enable bit is cleared to 0 (to disable transfer). In this case, therefore, the DMA transfer-completed interrupt request is not generated. Nor is this interrupt request generated even when transfer in ring buffer mode is terminated by clearing the transfer enable bit.

### 9.3.11 Status of Each Register after Completion of DMA Transfer

When DMA transfer is completed, the status of the source address and destination address registers becomes as follows:

**(1) Address fixed**

- The value set in the address register before DMA transfer started remains intact (fixed).

**(2) Address incremental**

- For 8-bit transfer, the value of the address register is the last transfer address + 1.
- For 16-bit transfer, the value of the address register is the last transfer address + 2.

The transfer count register when DMA transfer completed is in an underflow state (H'FF). Therefore, to perform another DMA transfer, set the transfer count register newly again, except when you are performing transfers 256 times (H'FF).

## 9.4 Precautions about the DMAC

### • About writing to DMAC related registers

Because DMA transfer involves exchanging data via the internal bus, basically you only can write to the DMAC related registers immediately after reset or when transfer is disabled (transfer enable bit = 0). When transfer is enabled, do not write to the DMAC related registers because write operation to those registers, except the DMA transfer enable bit, transfer request flag, and the DMA Transfer Count Register which is protected in hardware, is instable.

The table below shows the registers that can or cannot be accessed for write.

**Table 9.4.1 DMAC Related Registers That Can or Cannot Be Accessed for Write**

| Status | Transfer enable bit | Transfer request flag | Other DMAC related registers |
|---|---|---|---|
| When transfer is enabled | ○ | ○ | × |
| When transfer is disabled | ○ | ○ | ○ |

○ : Can be accessed ; × : Cannot be accessed

For even registers that can exceptionally be written to while transfer is enabled, the following requirements must be met.

① DMA Channel Control Register's transfer enable bit and transfer request flag

For all other bits of the channel control register, be sure to write the same data that those bits had before you wrote to the transfer enable bit or transfer request flag. Note that you only can write a 0 to the transfer request flag as valid data.

② DMA Transfer Count Register

When transfer is enabled, this register is protected in hardware, so that any data you write to this register is ignored.

③ Rewriting the DMA source and DMA destination addresses on different channels by DMA transfer

In this case, you are writing to the DMAC related registers while DMA is enabled, but this practically does not present any problem. However, you cannot DMA-transfer to the DMAC related registers on the local channel itself in which you are currently operating.

### • Manipulating DMAC related registers by DMA transfer

When manipulating DMAC related registers by means of DMA transfer (e.g., reloading the DMAC related registers' initial values by DMA transfer), do not write to the DMAC related registers on the local channel itself through that channel. (If this precaution is neglected, device operation cannot be guaranteed.)

Only if residing on other channels, you can write to the DMAC related registers by means of DMA transfer. (For example, you can rewrite the DMAn Source Address and DMAn Destination Address Registers on channel 1 by DMA transfer through channel 0.)

### • About the DMA Interrupt Request Status Register

When clearing the DMA Interrupt Request Status Register, be sure to write 1s to all bits but the one you want to clear. The bits to which you wrote 1s retain the previous data they had before the write.

### • About the stable operation of DMA transfer

To ensure the stable operation of DMA transfer, never rewrite the DMAC related registers, except the DMA Channel Control Register's transfer enable bit, unless transfer is disabled. One exception is that even when transfer is enabled, you can rewrite the DMA Source Address and DMA Destination Address Registers by DMA transfer from one channel to another.

# CHAPTER 10
# MULTIJUNCTION TIMERS

10.1 Outline of Multijunction Timers
10.2 Common Units of Multijunction Timer
10.3 TOP (Output-related 16-bit Timer)
10.4 TIO (Input/Output-related 16-bit Timer)
10.5 TMS (Input-related 16-bit Timer)
10.6 TML (Input-related 32-bit Timer)

## 10.1 Outline of Multijunction Timers

The multijunction timers (abbreviated MJT) have input event and output event buses. Therefore, in addition to being used as a single unit, the timers can be internally connected to each other. This capability allows for highly flexible timer configuration, making it possible to meet various application needs. It is because the timers are connected to the internal event bus at multiple points that they are called the "multijunction" timers.

The 32171 has four types of multijunction timers as listed in the table below, providing a total of 37 channels of timers.

**Table 10.1.1 Outline of Multijunction Timers**

| Name | Type | Number of Channels | Description |
|---|---|---|---|
| TOP<br>(Timer Output) | Output-related<br>16-bit timer<br>(down-counter) | 11 | One of three output modes can be selected by software.<br><With correction function><br>• Single-shot output mode<br>• Delayed single-shot output mode<br><Without correction function><br>• Continuous output mode |
| TIO<br>(Timer Input Output) | Input/output-related<br>16-bit timer<br>(down-counter) | 10 | One of three input modes or four output modes can be selected by software.<br><Input modes><br>• Measure clear input mode<br>• Measure free-run input mode<br>• Noise processing input mode<br><Output mode without correction function><br>• PWM output mode<br>• Single-shot output mode<br>• Delayed single-shot output mode<br>• Continuous output mode |
| TMS<br>(Timer Measure Small) | Input-related<br>16-bit timer<br>(up-counter) | 8 | 16-bit input measure timer |
| TML<br>(Timer Measure Large) | Input-related<br>32-bit timer<br>(up-counter) | 8 | 32-bit input measure timer |

**Table 10.1.2 Interrupt Generation Functions of MJT**

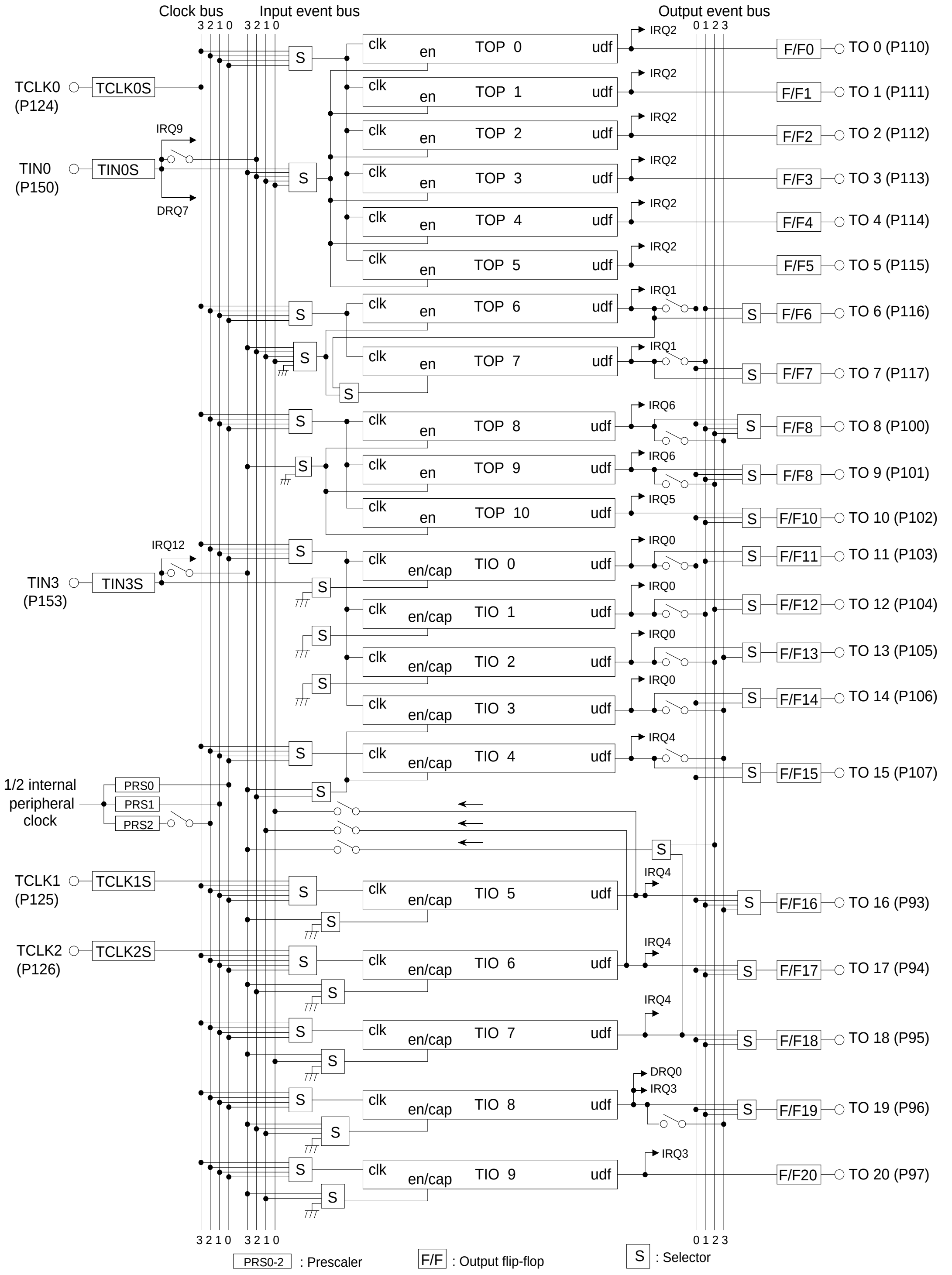
| Signal Name | MJT Interrupt Request Source | Source of Interrupt Request | No. of ICU Input Source |
|---|---|---|---|
| IRQ12 | TIN3 input | MJT input interrupt 4 | 1 |
| IRQ11 | TIN20 - TIN23 input | MJT input interrupt 3 | 4 |
| IRQ10 | TIN16 - TIN19 input | MJT input interrupt 2 | 4 |
| IRQ9 | TIN0 input | MJT input interrupt 1 | 1 |
| IRQ7 | TMS0, TMS1 output | MJT output interrupt 7 | 2 |
| IRQ6 | TOP8, TOP9 output | MJT output interrupt 6 | 2 |
| IRQ5 | TOP10 output | MJT output interrupt 5 | 1 |
| IRQ4 | TIO4 - 7 output | MJT output interrupt 4 | 4 |
| IRQ3 | TIO8, TIO9 output | MJT output interrupt 3 | 2 |
| IRQ2 | TOP0 - 5 output | MJT output interrupt 2 | 6 |
| IRQ1 | TOP6, TOP7 output | MJT output interrupt 1 | 2 |
| IRQ0 | TIO0 - 3 output | MJT output interrupt 0 | 4 |

**Table 10.1.3 DMA Transfer Request Generation by MJT**

| Signal Name | DMA Transfer Request Source | DMAC Input Channel |
|---|---|---|
| DRQ0 | TIO8 underflow | Channel 0 |
| DRQ1 | Input event bus 2 | Channel 0 |
| DRQ2 | Output event bus 0 | Channel 1 |
| DRQ4 | Output event bus 1 | Channel 2 |
| DRQ5 | TIN18 input | Channel 2 |
| DRQ6 | TIN19 input | Channel 4 |
| DRQ7 | TIN0 input | Channel 3 |
| DRQ12 | TIN20 input | Channel 5 |
| DRQ13 | Input event bus 0 | Channel 8 |

**Table 10.1.4 A-D Conversion Start Request by MJT**

| Signal Name | A-D Conversion Start Request Source | A-D Converter |
|---|---|---|
| AD0TRG | Output event bus 3 | Can be input to A-D0 conversion start trigger |



Note 1: IRQ0-7 and IRQ9-12 are interrupt signals, with the same number representing interrupts of the same group (see Table 10.1.2). DRQ 0-2, DRQ4-7, DRQ12, and DRQ13 are DMA request signals to the DMAC (see Table 10.1.3). AD0TRG is a trigger signal to the A-D converter (see Table 10.1.4).

Note 2: Indicates timer input pin edge selection output.

Note 3: Indicates input signals from peripheral circuits (AD and SIO).

**Figure 10.1.1 Block Diagram of MJT (1/3)**

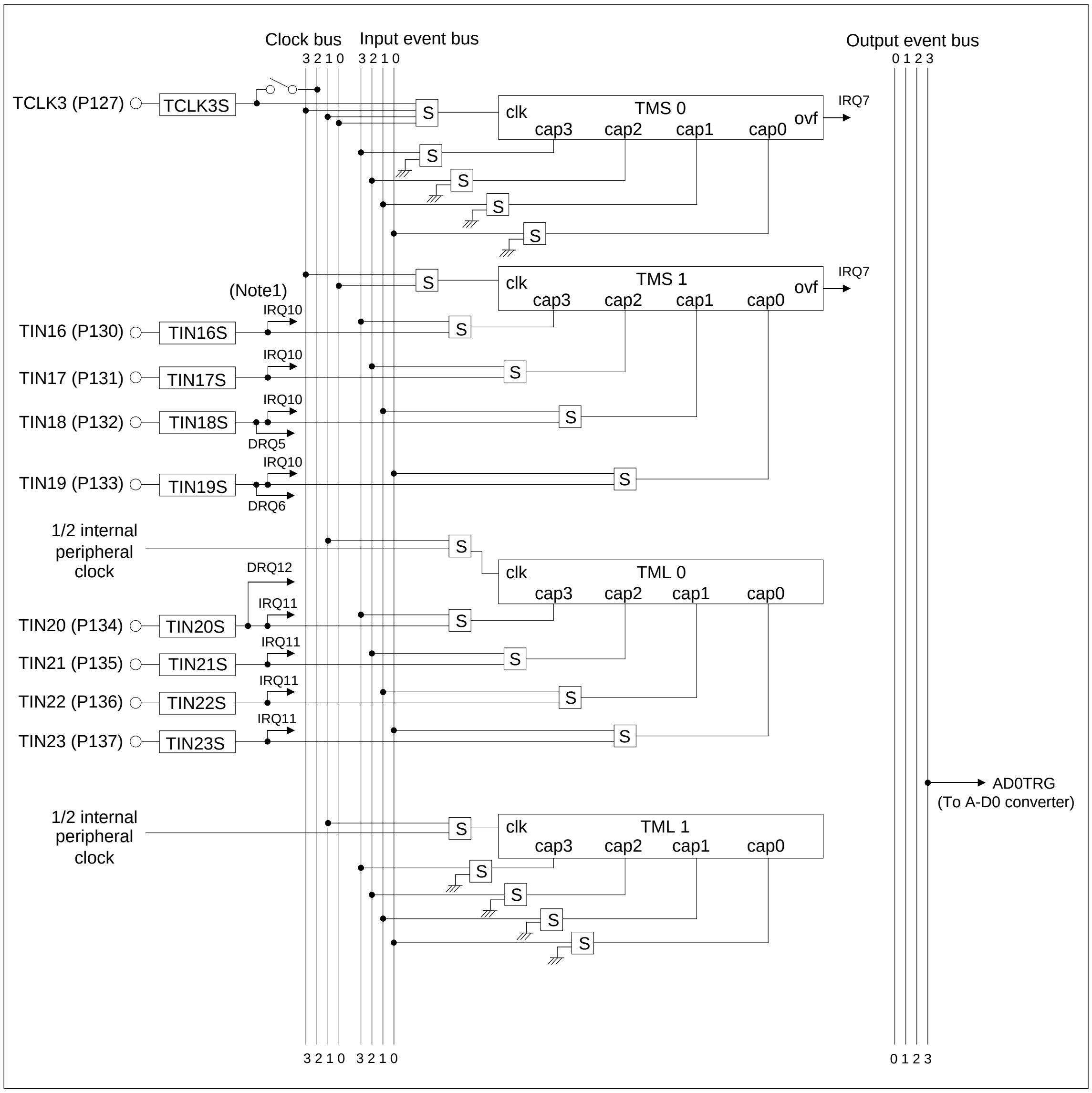


**Figure 10.1.2 Block Diagram of MJT (2/3)**

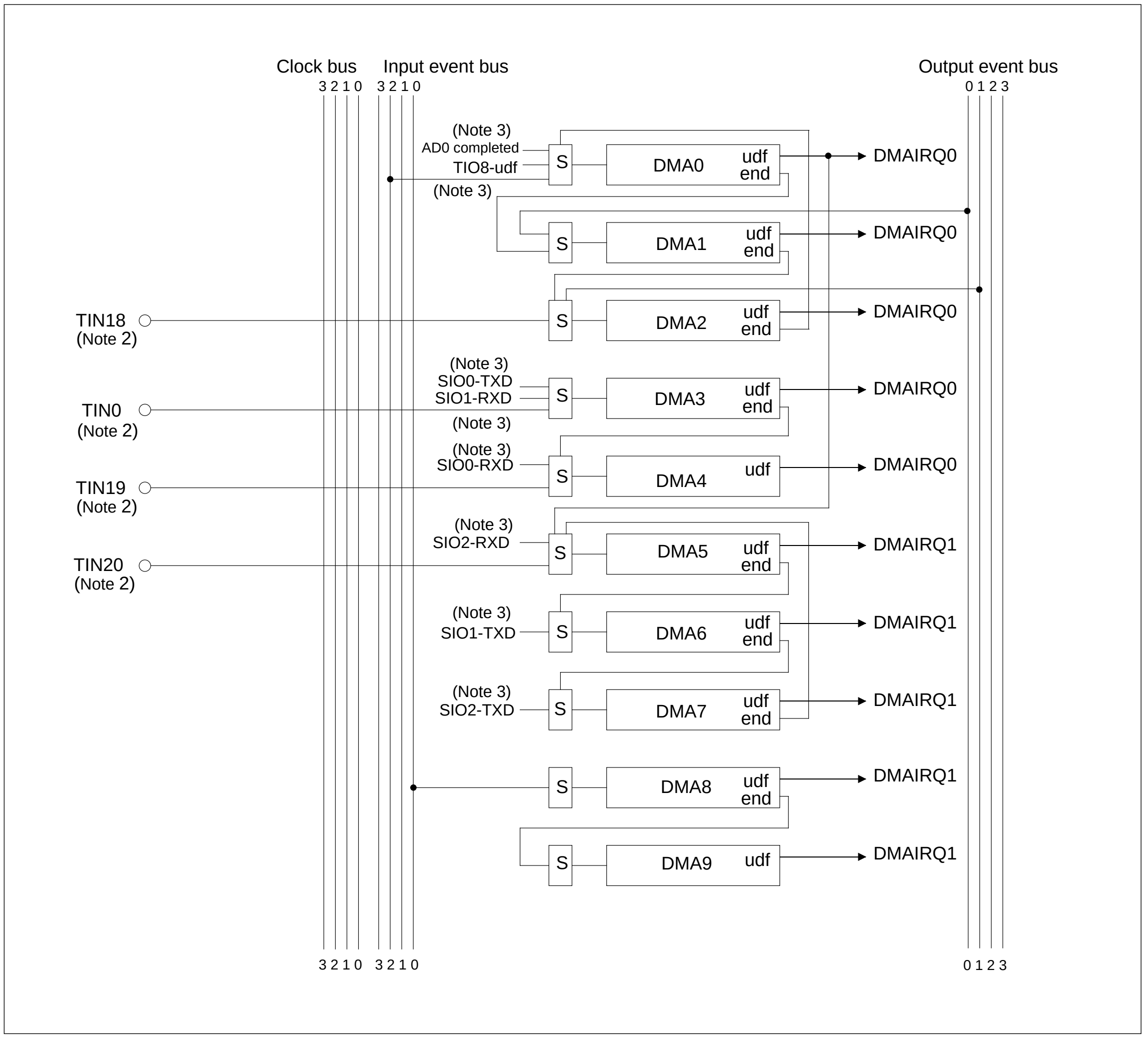


**Figure 10.1.3 Block Diagram of MJT (3/3)**

## 10.2 Common Units of Multijunction Timer

The common units of the multijunction timer include the following:

- Prescaler unit
- Clock bus/input-output event bus control unit
- Input processing control unit
- Output flip-flop control unit
- Interrupt control unit

### 10.2.1 Timer Common Register Map

The diagrams in the next pages show a map of registers in the common units of the multijunction timer.

| Address | D0 +0 Address D7 | D8 +1 Address D15 |
|---|---|---|
| H'0080 0200 | | Clock Bus & Input Event Bus Control Register (CKIEBCR) |
| H'0080 0202 | Prescaler Register 0 (PRS0) | Prescaler Register 1 (PRS1) |
| H'0080 0204 | Prescaler Register 2 (PRS2) | Output Event Bus Control Register (OEBCR) |
| H'0080 0210 | TCLK Input Processing Control Register (TCLKCR) | |
| H'0080 0212 | TIN Input Processing Control Register 0 (TINCR0) | |
| H'0080 0214 | | |
| H'0080 0216 | | |
| H'0080 0218 | TIN Input Processing Control Register 3 (TINCR3) | |
| H'0080 021A | TIN Input Processing Control Register 4 (TINCR4) | |
| H'0080 0220 | F/F Source Select Register 0 (FFS0) | |
| H'0080 0222 | | F/F Source Select Register 1 (FFS1) |
| H'0080 0224 | F/F Protect Register 0 (FFP0) | |
| H'0080 0226 | F/F Data Register 0 (FFD0) | |
| H'0080 0228 | | F/F Protect Register 1 (FFP1) |
| H'0080 022A | | F/F Data Register 1 (FFD1) |
| H'0080 0230 | TOP Interrupt Control Register 0 (TOPIR0) | TOP Interrupt Control Register 1 (TOPIR1) |
| H'0080 0232 | TOP Interrupt Control Register 2 (TOPIR2) | TOP Interrupt Control Register 3 (TOPIR3) |
| H'0080 0234 | TIO Interrupt Control Register 0 (TIOIR0) | TIO Interrupt Control Register 1 (TIOIR1) |
| H'0080 0236 | TIO Interrupt Control Register 2 (TIOIR2) | TMS Interrupt Control Register (TMSIR) |
| H'0080 0238 | TIN Interrupt Control Register 0 (TINIR0) | TIN Interrupt Control Register 1 (TINIR1) |
| H'0080 023A | | |
| H'0080 023C | TIN Interrupt Control Register 4 (TINIR4) | TIN Interrupt Control Register 5 (TINIR5) |
| H'0080 023E | TIN Interrupt Control Register 6 (TINIR6) | |

Blank addresses are reserved.

Note: • The registers included in thick frames must always be accessed in halfwords.

**Figure 10.2.1 Timer Common Register Map**

### 10.2.2 Prescaler Unit

The prescalers PRS0-2 are an 8-bit counter, which generates clocks supplied to each timer (TOP, TIO, TMS, and TML) from the divide-by-2 frequency of the internal peripheral clock (10.0 MHz when the internal peripheral clock = 20 MHz).
The values of prescaler registers are initialized to H'00 immediately after reset. Also, when you rewrite the set value of any prescaler register, the device starts operating with the new value simultaneously when the prescaler underflows.

Values H'00 to H'FF can be set in the counter registers of prescalers. The prescalers' divide-by ratios are given by the equation below:

$$\text{Prescaler divide-by ratio} = \frac{1}{\text{Prescaler set value} + 1}$$

**■ Prescaler Register 0 (PRS0)** <Address: H'0080 0202>

**■ Prescaler Register 1 (PRS1)** <Address: H'0080 0203>

**■ Prescaler Register 2 (PRS2)** <Address: H'0080 0204>

| **D0** | **1** | **2** | **3** | **4** | **5** | **6** | **D7** |
|---|---|---|---|---|---|---|---|
| **( D8** | **9** | **10** | **11** | **12** | **13** | **14** | **D15 )** |
| PRS0 - PRS2 | | | | | | | |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 7 | PRS0, 2 | Sets the prescaler's divide-by value | ○ | ○ |
| 8 - 15 | PRS1 | | | |

Prescaler Registers 0-2 start counting after exiting reset.

### 10.2.3 Clock Bus/Input-Output Event Bus Control Unit

#### (1) Clock bus

The clock bus is provided for supplying clock to each timer, and is comprised of four lines of clock bus 0-3. Each timer can use this clock bus signal as clock input signal. The table below lists the signals that can be fed to the clock bus.

**Table 10.2.1 Signals That Can Be Fed to Each Clock Bus Line**

| Clock Bus | Acceptable Signal |
|---|---|
| 3 | TCLK0 input |
| 2 | Internal prescaler (PRS2) or TCLK3 input |
| 1 | Internal prescaler (PRS1) |
| 0 | Internal prescaler (PRS0) |

#### (2) Input event bus

The input event bus is provided for supplying a count enable signal or measure capture signal to each timer, and is comprised of four lines of input event bus 0-3. Each timer can use this input event bus signal as enable (or capture) signal input. The table below lists the signals that can be fed to the input event bus.

**Table 10.2.2 Signals That Can Be Fed to Each Input Event Bus Line**

| Input Event Bus | Acceptable Signal |
|---|---|
| 3 | TIN3 input, output event bus 2 or TIO7 underflow signal |
| 2 | TIN0 input |
| 1 | TIO6 underflow signal |
| 0 | TIO5 underflow signal |

### (3) Output event bus

The output event bus has the underflow signal from each timer connected to it, and is comprised of four lines of output event bus 0-3. Output event bus signals are connected to output flip-flops, and can also be connected to other peripheral circuits-output event bus 3 to A-D0 converter, output event bus 0 to DMA channel 1, and output event bus 1 to DMA channel 2. Furthermore, output event bus 2 can be connected to input event bus 3.

The table below lists the signals that can be connected to the output event bus.

**Table 10.2.3 Signals That Can Be Connected (Fed) to Each Output Event Bus Line**

| Output Event Bus | Connectable (Acceptable) Signal (Note 1) |
|---|---|
| 3 | TOP8, TIO3, TIO4, or TIO8 underflow signal |
| 2 | TOP9 or TIO2 underflow signal |
| 1 | TOP7 or TIO1 underflow signal |
| 0 | TOP6 or TIO0 underflow signal |

Note 1: For details about the output destinations of output event bus signals, refer to Figure 10.1.1, "Block Diagram of MJT."

Timings at which signals are generated to the output event bus by each timer (and those generated to the input event bus by TIO5, 6) are shown below. (Note that they are generated at different timings than those forwarded to output flip-flops by timers.)

**Table 10.2.4 Timings at Which Signals Are Generated to the Output Event Bus by Each Timer**

| Timer | Mode | Timings at which signals are generated to the output event bus |
|---|---|---|
| TOP | Single-shot output mode<br>Delayed single-shot output mode<br>Continuous output mode | When the counter underflows<br>When the counter underflows<br>When the counter underflows |
| TIO (Note 1) | Measure clear input mode<br>Measure free-run input mode<br>Noise processing input mode | When the counter underflows<br>When the counter underflows<br>When the counter underflows |
| | PWM output mode<br>Single-shot output mode<br>Delayed single-shot output mode<br>Continuous output mode | When the counter underflows<br>When the counter underflows<br>When the counter underflows<br>When the counter underflows |
| TMS | (16-bit measure input) | No signal generation function |
| TML | (32-bit measure input) | No signal generation function |

Note 1: TIO5, 6 output underflow signals to the input event bus.

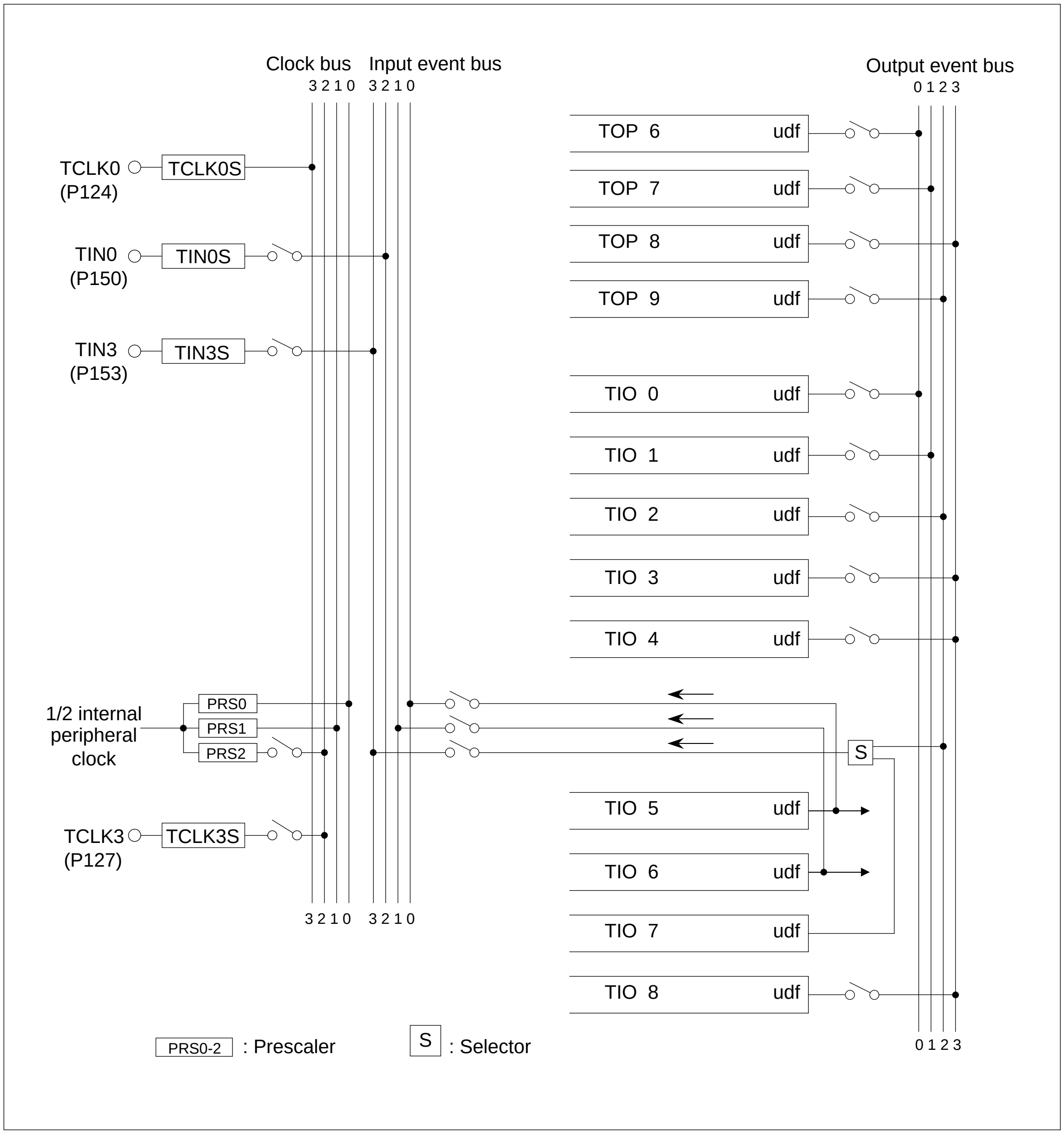


**Figure 10.2.2 Conceptual Diagram of the Clock Bus and Input/Output Event Bus**

The clock bus/input-output bus control unit has the following registers:

- Clock Bus & Input Event Bus Control Register (CKIEBCR)
- Output Event Bus Control Register (OEBCR)

### ■ Clock Bus & Input Event Bus Control Register (CKIEBCR) <Address: H'0080 0201>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| IEB3S | | IEB2S | | IEB1S | IEB0S | ▨ | CKB2S |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8, 9 | IEB3S<br>(input event bus 3 input selection) | 0X : Selects external input 3 (TIN3)<br>10 : Selects output event bus 2<br>11 : Selects TIO7 output | ○ | ○ |
| 10, 11 | IEB2S<br>(input event bus 2 input selection) | 00 : Selects external input 0 (TIN0)<br>01 : No selection<br>1X : No selection | ○ | ○ |
| 12 | IEB1S<br>(input event bus 1 input selection) | 0 : No selection<br>1 : Selects TIO6 output | ○ | ○ |
| 13 | IEB0S<br>(input event bus 0 input selection) | 0 : No selection<br>1 : Selects TIO5 output | ○ | ○ |
| 14 | No functions assigned | | 0 | — |
| 15 | CKB2S<br>(Clock Bus 2 input selection) | 0 : Selects prescaler 2<br>1 : Selects external clock 3 (TCLK3) | ○ | ○ |

The register CKIEBCR is used to select the clock source (external input or prescaler) supplied to the clock bus and the count enable/capture signal (external input or output event bus) supplied to the input event bus.

### ■ Output Event Bus Control Register (OEBCR) <Address: H'0080 0205>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| OEB3S | | ▨ | OEB2S | ▨ | OEB1S | ▨ | OEB0S |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8, 9 | OEB3S<br>(output event bus 3 input selection) | 00 : Selects TOP8 output<br>01 : Selects TIO3 output<br>10 : Selects TIO4 output<br>11 : Selects TIO8 output | ○ | ○ |
| 10 | No functions assigned | | 0 | — |
| 11 | OEB2S<br>(output event bus 2 input selection) | 0 : Selects TOP9 output<br>1 : Selects TIO2 output | ○ | ○ |
| 12 | No functions assigned | | 0 | — |
| 13 | OEB1S<br>(output event bus 1 input selection) | 0 : Selects TOP7 output<br>1 : Selects TIO1 output | ○ | ○ |
| 14 | No functions assigned | | 0 | — |
| 15 | OEB0S<br>(output event bus 0 input selection) | 0 : Selects TOP6 output<br>1 : Selects TIO0 output | ○ | ○ |

The register OEBCR is used to select the timer (TOP or TIO) whose underflow signal is supplied to the output event bus.

### 10.2.4 Input Processing Control Unit

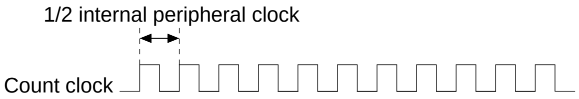
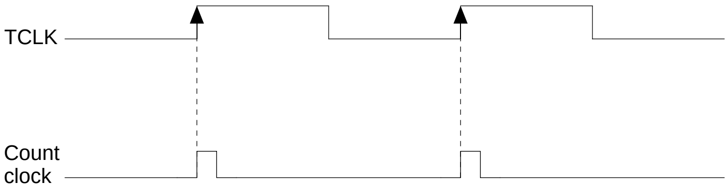
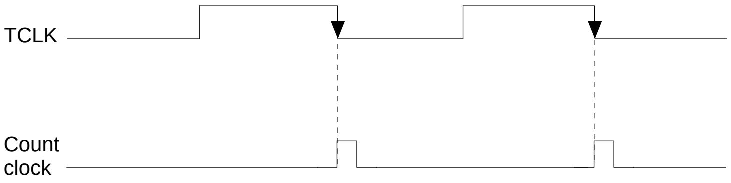
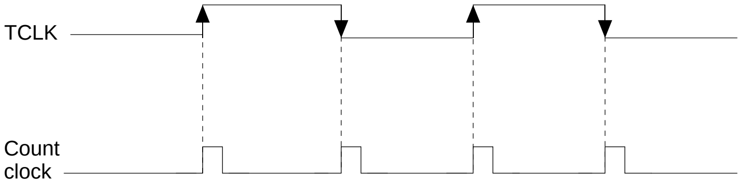
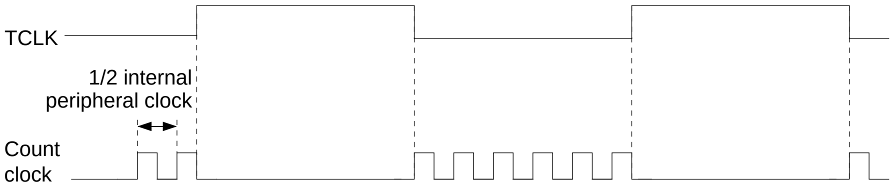
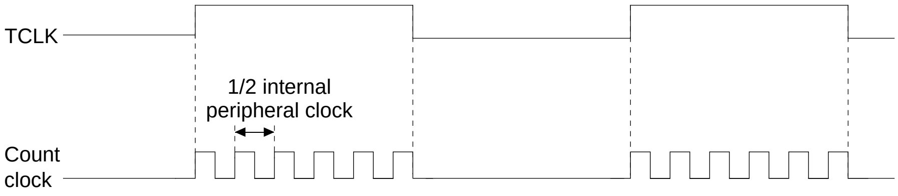
The input processing control unit processes the TCLK and TIN signals fed into the MJT. In the TCLK input processing unit, selection is made of the source of TCLK signal, or for external input, the active edge (rising or falling or both) or level (high or low) of the signal, with or at which to generate the clock signal fed to the clock bus.

In the TIN input processing unit, selection is made of the active edge (rising or falling or both) or level (high or low) of the signal at which to generate the enable, measure or count source signal for each timer or the signal fed to each event bus.

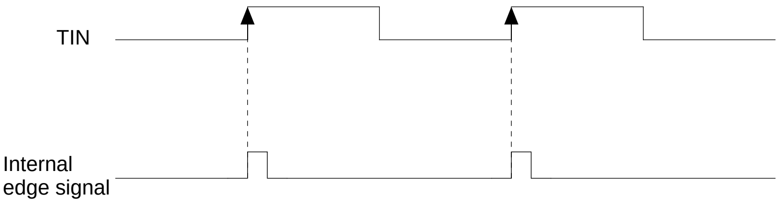
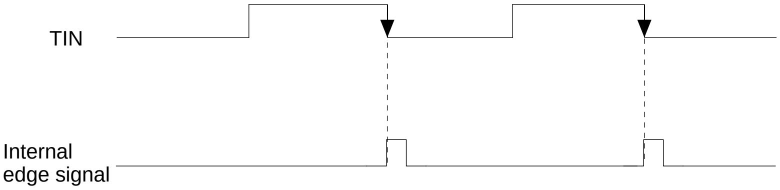
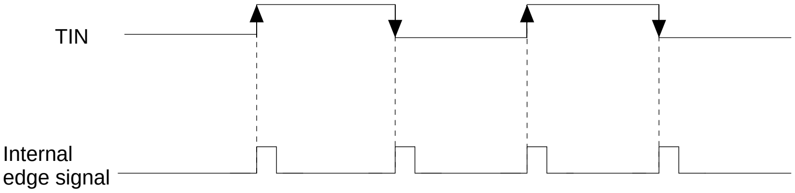
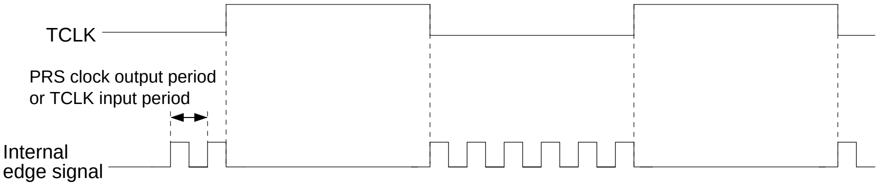
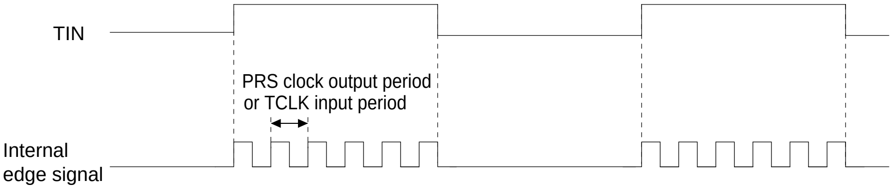
Following input processing control registers are included:

- TCLK Input Processing Control Register (TCLKCR)
- TIN Input Processing Control Register 0 (TINCR0)
- TIN Input Processing Control Register 3 (TINCR3)
- TIN Input Processing Control Register 4 (TINCR4)

### (1) Functions of TCLK input processing control registers

| Item | Function |
|---|---|
| 1/2 internal peripheral clock |  |
| Rising clock edge |  |
| Falling clock edge |  |
| Both edges |  |
| Low level |  |
| High level |  |

### (2) Functions of TIN input processing control registers

| Item | Function |
|---|---|
| Rising edge |  |
| Falling edge |  |
| Both edges |  |
| Low level |  TCLK<br>PRS clock output period or TCLK input period<br>Internal edge signal |
| High level |  TIN<br>PRS clock output period or TCLK input period<br>Internal edge signal |

### ■ TLCK Input Processing Control Register (TCLKCR)

<Address: H'0080 0210>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  | TCLK3S |  |  | TCLK2S |  |  |  | TCLK1S |  |  |  |  | TCLK0S |  |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | No functions assigned |  | 0 | — |
| 2, 3 | TCLK3S<br>(TCLK3 input<br>processing selection) | 00 : 1/2 internal peripheral clock<br>01 : Rising edge<br>10 : Falling edge<br>11 : Both edges | ○ | ○ |
| 4 | No functions assigned |  | 0 | — |
| 5 - 7 | TCLK2S<br>(TCLK2 input<br>processing selection) | 000 : Invalidates input<br>001 : Rising edge<br>010 : Falling edge<br>011 : Both edges<br>10X : Low level<br>11X : High level | ○ | ○ |
| 8 | No functions assigned |  | 0 | — |
| 9 - 11 | TCLK1S<br>(TCLK1 input<br>processing selection) | 000 : Invalidates input<br>001 : Rising edge<br>010 : Falling edge<br>011 : Both edges<br>10X : Low level<br>11X : High level | ○ | ○ |
| 12, 13 | No functions assigned |  | 0 | — |
| 14, 15 | TCLK0S<br>(TCLK0 input<br>processing selection) | 00 : 1/2 internal peripheral clock<br>01 : Rising edge<br>10 : Falling edge<br>11 : Both edges | ○ | ○ |

Note: • This register must always be accessed in halfwords.

### ■ TIN Input Processing Control Register 0 (TINCR0)

<Address: H'0080 0212>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | TIN4S |  |  |  | TIN3S |  |  |  |  | TIN2S |  | TIN1S |  | TIN0S |  |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | No functions assigned |  | 0 | — |
| 1 - 3 | TIN4S (reserved) | Set these bits to '000' (Note 1) | ○ | ○ |
| 4 | No functions assigned |  | 0 | — |
| 5 - 7 | TIN3S<br>(TIN3 input<br>processing selection) | 000 : Invalidates input<br>001 : Rising edge<br>010 : Falling edge<br>011 : Both edges<br>10X : Low level<br>11X : High level | ○ | ○ |
| 8, 9 | No functions assigned |  | 0 | — |
| 10, 11 | TIN2S (reserved) | Set these bits to '00' (Note 2) | ○ | ○ |
| 12, 13 | TIN1S (reserved) | Set these bits to '00' (Note 2) | ○ | ○ |
| 14, 15 | TIN0S<br>(TIN0 input<br>processing selection) | 00 : Invalidates input<br>01 : Rising edge<br>10 : Falling edge<br>11 : Both edges | ○ | ○ |

Note 1: Always set the TIN4S bits to '000.'

Note 2: Always set the TIN2S bits and TIN1S bits to '00.'

Note: • This register must always be accessed in halfwords.

### ■ TIN Input Processing Control Register 3 (TINCR3)

<Address: H'0080 0218>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TIN19S | | TIN18S | | TIN17S | | TIN16S | | TIN15S | | TIN14S | | TIN13S | | TIN12S | |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | TIN19S (TIN19 input processing selection) | 00 : Invalidates input | ○ | ○ |
| 2, 3 | TIN18S (TIN18 input processing selection) | 01 : Rising edge | | |
| 4, 5 | TIN17S (TIN17 input processing selection) | 10 : Falling edge | | |
| 6, 7 | TIN16S (TIN16 input processing selection) | 11 : Both edges | | |
| 8, 9 | TIN15S (reserved) | Set these bits to '00' (Note) | ○ | ○ |
| 10, 11 | TIN14S (reserved) | | | |
| 12, 13 | TIN13S (reserved) | | | |
| 14, 15 | TIN12S (reserved) | | | |

Notes: • Always set the TIN15S bits, TIN14S bits, TIN13S bits, and TIN12S bits to '00' .

• This register must always be accessed in halfwords.

### ■ TIN Input Processing Control Register 4 (TINCR4)

<Address: H'0080 021A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TIN33S | | TIN32S | | TIN31S | | TIN30S | | TIN23S | | TIN22S | | TIN21S | | TIN20S | |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | TIN33S (reserved) | Set these bits to '00' (Note) | ○ | ○ |
| 2, 3 | TIN32S (reserved) | | | |
| 4, 5 | TIN31S (reserved) | | | |
| 6, 7 | TIN30S (reserved) | | | |
| 8, 9 | TIN23S (TIN23 input processing selection) | 00 : Invalidates input | ○ | ○ |
| 10, 11 | TIN22S (TIN22 input processing selection) | 01 : Rising edge | | |
| 12, 13 | TIN21S (TIN21 input processing selection) | 10 : Falling edge | | |
| 14, 15 | TIN20S (TIN20 input processing selection) | 11 : Both edges | | |

**Notes:** • Always set the TIN33S bits, TIN32S bits, TIN31S bits, and TIN30S bits to '00' .

• This register must always be accessed in halfwords.

### 10.2.5 Output Flip-Flop Control Unit

The output flip-flop control unit controls the flip-flop (F/F) provided for each timer output. Following flip-flop control registers are included:

- F/F Source Select Register 0 (FFS0)
- F/F Source Select Register 1 (FFS1)
- F/F Protect Register 0 (FFP0)
- F/F Protect Register 1 (FFP1)
- F/F Data Register 0 (FFD0)
- F/F Data Register 1 (FFD1)

Timings at which signals are generated to the output flip-flop by each timer are shown in Table 10.2.5 below. (Note that signals are generated at different timings than those fed to the output event bus.)

Table 10.2.5 Timings at Which Signals Are Generated to the Output Flip-Flop by Each Timer

| Timer | Mode | Timings at which signals are generated to the output flip-flop |
|---|---|---|
| TOP | Single-shot output mode | When counter is enabled and when underflows |
| | Delayed single-shot output mode | When counter underflows |
| | Continuous output mode | When counter is enabled and when underflows |
| TIO | Measure clear input mode | When counter underflows |
| | Measure free-run input mode | When counter underflows |
| | Noise processing input mode | When counter underflows |
| | PWM output mode | When counter is enabled and when underflows |
| | Single-shot output mode | When counter is enabled and when underflows |
| | Delayed single-shot output mode | When counter underflows |
| | Continuous output mode | When counter is enabled and when underflows |
| TMS | (16-bit measure input) | No signal generation function |
| TML | (32-bit measure input) | No signal generation function |

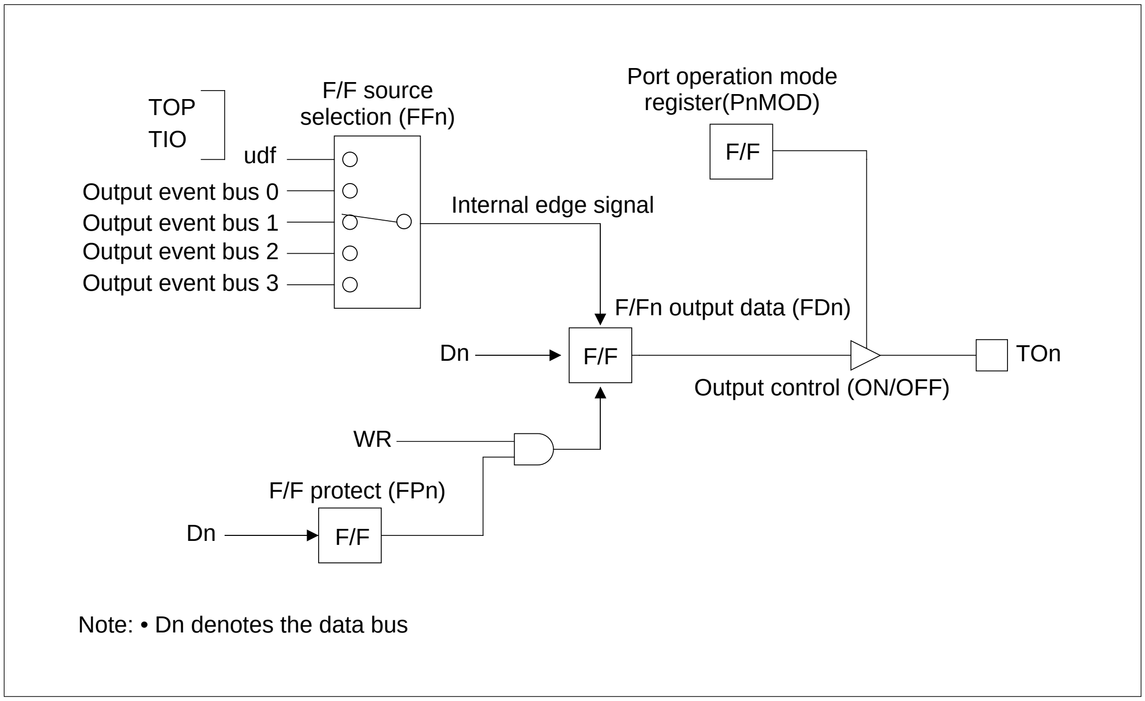


Figure 10.2.3 Configuration of the F/F Output Circuit

### ■ F/F Source Select Register 0 (FFS0)

<Address: H'0080 0220>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  | FF15 | FF14 | FF13 | FF12 | FF11 | FF10 |  | FF9 |  | FF8 |  | FF7 | FF6 |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 2 | No functions assigned |  | 0 | — |
| 3 | FF15 (F/F15 source selection) | 0 : TIO4 output<br>1 : Output event bus 0 | ○ | ○ |
| 4 | FF14 (F/F14 source selection) | 0 : TIO3 output<br>1 : Output event bus 0 | ○ | ○ |
| 5 | FF13 (F/F13 source selection) | 0 : TIO2 output<br>1 : Output event bus 3 | ○ | ○ |
| 6 | FF12 (F/F12 source selection) | 0 : TIO1 output<br>1 : Output event bus 2 | ○ | ○ |
| 7 | FF11 (F/F11 source selection) | 0 : TIO0 output<br>1 : Output event bus 1 | ○ | ○ |
| 8, 9 | FF10 (F/F10 source selection) | 0X : TOP10 output<br>10 : Output event bus 0<br>11 : Output event bus 1 | ○ | ○ |
| 10, 11 | FF9 (F/F9 source selection) | 0X : TOP9 output<br>10 : Output event bus 0<br>11 : Output event bus 1 | ○ | ○ |
| 12, 13 | FF8 (F/F8 source selection) | 00 : TOP8 output<br>01 : Output event bus 0<br>10 : Output event bus 1<br>11 : Output event bus 2 | ○ | ○ |
| 14 | FF7 (F/F7 source selection) | 0 : TOP7 output<br>1 : Output event bus 0 | ○ | ○ |
| 15 | FF6 (F/F6 source selection) | 0 : TOP6 output<br>1 : Output event bus 1 | ○ | ○ |

Note: • This register must always be accessed in halfwords.

### ■ F/F Source Select Register 1 (FFS1)

<Address: H'0080 0223>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| FF19 | | FF18 | | FF17 | | FF16 | |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8, 9 | FF19 (F/F19 source selection) | 0X : TIO8 output<br>10 : Output event bus 0<br>11 : Output event bus 1 | ○ | ○ |
| 10, 11 | FF18 (F/F18 source selection) | 0X : TIO7 output<br>10 : Output event bus 0<br>11 : Output event bus 1 | ○ | ○ |
| 12, 13 | FF17 (F/F17 source selection) | 0X : TIO6 output<br>10 : Output event bus 0<br>11 : Output event bus 1 | ○ | ○ |
| 14, 15 | FF16 (F/F16 source selection) | 00 : TIO5 output<br>01 : Output event bus 0<br>10 : Output event bus 1<br>11 : Output event bus 3 | ○ | ○ |

The registers FFS0 and FFS1 are used to select the signal sources fed to each output F/F (flip-flop). For these signal sources, you can choose signals from the internal output bus or underflow output from each timer.

## ■ F/F Protect Register 0 (FFP0)

<Address: H'0080 0224>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FP15 | FP14 | FP13 | FP12 | FP11 | FP10 | FP9 | FP8 | FP7 | FP6 | FP5 | FP4 | FP3 | FP2 | FP1 | FP0 |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | FP15 (F/F15 protect) | 0 : Enables write to F/F output bit | ○ | ○ |
| 1 | FP14 (F/F14 protect) | 1 : Disables write to F/F output bit | | |
| 2 | FP13 (F/F13 protect) | | | |
| 3 | FP12 (F/F12 protect) | | | |
| 4 | FP11 (F/F11 protect) | | | |
| 5 | FP10 (F/F10 protect) | | | |
| 6 | FP9 (F/F9 protect) | | | |
| 7 | FP8 (F/F8 protect) | | | |
| 8 | FP7 (F/F7 protect) | | | |
| 9 | FP6 (F/F6 protect) | | | |
| 10 | FP5 (F/F5 protect) | | | |
| 11 | FP4 (F/F4 protect) | | | |
| 12 | FP3 (F/F3 protect) | | | |
| 13 | FP2 (F/F2 protect) | | | |
| 14 | FP1 (F/F1 protect) | | | |
| 15 | FP0 (F/F0 protect) | | | |

Note: • This register must always be accessed in halfwords.

This register controls write to each output F/F (flip-flop) by enabling or disabling it. When this register is set to disable write to any output F/F, writing to the F/F Data Register has no effect.

### ■ F/F Protect Register 1 (FFP1)

<Address: H'0080 0229>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  | FP20 | FP19 | FP18 | FP17 | FP16 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | No functions assigned |  | 0 | — |
| 11 | FP20 (F/F20 protect) | 0 : Enables write to F/F output bit<br>1 : Disables write to F/F output bit | ○ | ○ |
| 12 | FP19 (F/F19 protect) |  |  |  |
| 13 | FP18 (F/F18 protect) |  |  |  |
| 14 | FP17 (F/F17 protect) |  |  |  |
| 15 | FP16 (F/F16 protect) |  |  |  |

This register controls write to each output F/F (flip-flop) by enabling or disabling it. When this register is set to disable write to any output F/F, writing to the F/F Data Register has no effect.

## ■ F/F Data Register 0 (FFD0)

<Address: H'0080 0226>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FD15 | FD14 | FD13 | FD12 | FD11 | FD10 | FD9 | FD8 | FD7 | FD6 | FD5 | FD4 | FD3 | FD2 | FD1 | FD0 |

<When reset : H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | FD15 (F/F15 output data) | 0 : F/F output data = 0 | ○ | ○ |
| 1 | FD14 (F/F14 output data) | 1 : F/F output data = 1 | | |
| 2 | FD13 (F/F13 output data) | | | |
| 3 | FD12 (F/F12 output data) | | | |
| 4 | FD11 (F/F11 output data) | | | |
| 5 | FD10 (F/F10 output data) | | | |
| 6 | FD9 (F/F9 output data) | | | |
| 7 | FD8 (F/F8 output data) | | | |
| 8 | FD7 (F/F7 output data) | | | |
| 9 | FD6 (F/F6 output data) | | | |
| 10 | FD5 (F/F5 output data) | | | |
| 11 | FD4 (F/F4 output data) | | | |
| 12 | FD3 (F/F3 output data) | | | |
| 13 | FD2 (F/F2 output data) | | | |
| 14 | FD1 (F/F1 output data) | | | |
| 15 | FD0 (F/F0 output data) | | | |

Note: • This register must always be accessed in halfwords.

This register is used to set data in each output F/F (flip-flop). Normally, the data output from F/F changes with timer output, but by setting data 0 or 1 in this register you can produce the desired output from any F/F. The F/F Data Register can only be accessed for write when the F/F Protect Register described above is enabled for write.

## ■ F/F Data Register 1 (FFD1)

<Address: H'0080 022B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  | FD20 | FD19 | FD18 | FD17 | FD16 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | No functions assigned |  | 0 | — |
| 11 | FD20 (F/F20 output data) | 0 : F/F output data = 0<br>1 : F/F output data = 1 | ○ | ○ |
| 12 | FD19 (F/F19 output data) |  |  |  |
| 13 | FD18 (F/F18 output data) |  |  |  |
| 14 | FD17 (F/F17 output data) |  |  |  |
| 15 | FD16 (F/F16 output data) |  |  |  |

This register is used to set data in each output F/F (flip-flop). Normally, the data output from F/F changes with timer output, but by setting data 0 or 1 in this register you can produce the desired output from any F/F. The F/F Data Register can only be accessed for write when the F/F Protect Register described above is enabled for write.

### 10.2.6 Interrupt Control Unit

The interrupt control unit controls the interrupt signals sent to the interrupt controller by each timer. Following timer interrupt control registers are provided for each timer.

- TOP Interrupt Control Register 0 (TOPIR0)
- TOP Interrupt Control Register 1 (TOPIR1)
- TOP Interrupt Control Register 2 (TOPIR2)
- TOP Interrupt Control Register 3 (TOPIR3)
- TIO Interrupt Control Register 0 (TIOIR0)
- TIO Interrupt Control Register 1 (TIOIR1)
- TIO Interrupt Control Register 2 (TIOIR2)

- TMS Interrupt Control Register (TMSIR)

- TIN Interrupt Control Register 0 (TINIR0)
- TIN Interrupt Control Register 1 (TINIR1)
- TIN Interrupt Control Register 4 (TINIR4)
- TIN Interrupt Control Register 5 (TINIR5)
- TIN Interrupt Control Register 6 (TINIR6)

For interrupts which have only one source of interrupt in one interrupt table, no interrupt control registers are provided in the timer, and the interrupt status flags are automatically managed within the interrupt controller. For details, refer to Chapter 5, "Interrupt Controller."

- TOP10 MJT Output Interrupt 5 (IRQ5)

For interrupts which have two or more sources of interrupt in one interrupt table, interrupt control registers are provided, with which to control interrupt requests and determine interrupt input. Therefore, the status flags in the interrupt controller function only as a bit to show whether an interrupt-enabled interrupt request occurred and cannot be written to.

#### (1) Interrupt request status bit

This status bit shows whether an interrupt request occurred. When an interrupt request is generated, this bit is set in hardware (but cannot be set in software). The status bit is cleared by writing a 0, but not affected by writing a 1, in which case the bit holds the status intact. Because the status bit is unaffected by interrupt mask bits, it can also be used to check the operation of peripheral function. In interrupt processing, make sure that among grouped interrupt flags, only the flag for the serviced interrupt is cleared. Clearing flags for unserviced interrupts results in the pending interrupt requests also being cleared.

### (2) Interrupt mask bit

This bit is used to disable unnecessary interrupts among grouped interrupt requests. Set this bit to 0 to enable interrupts or 1 to disable interrupts.

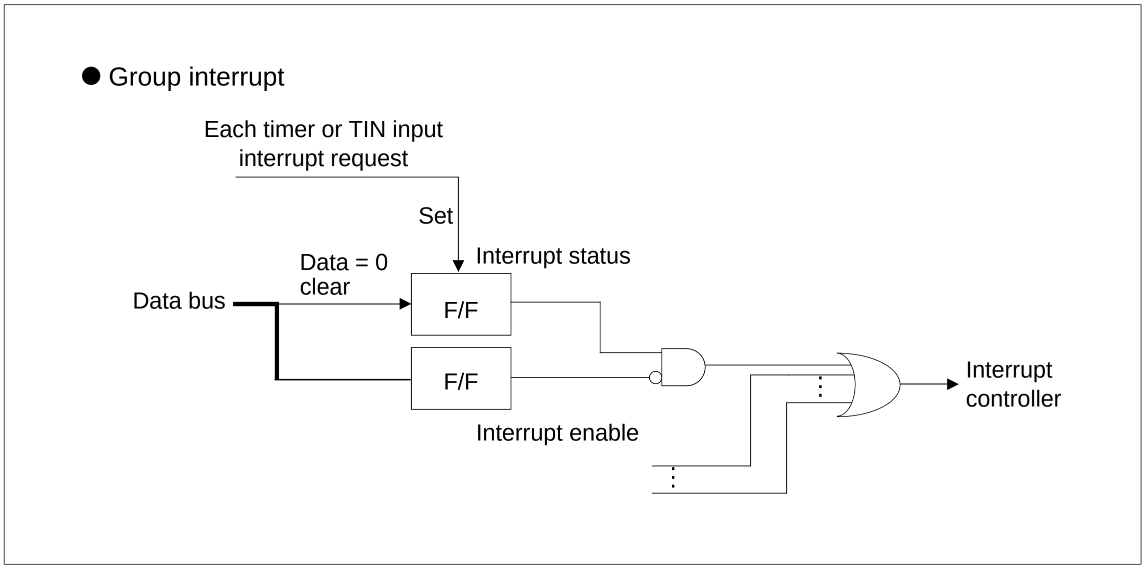
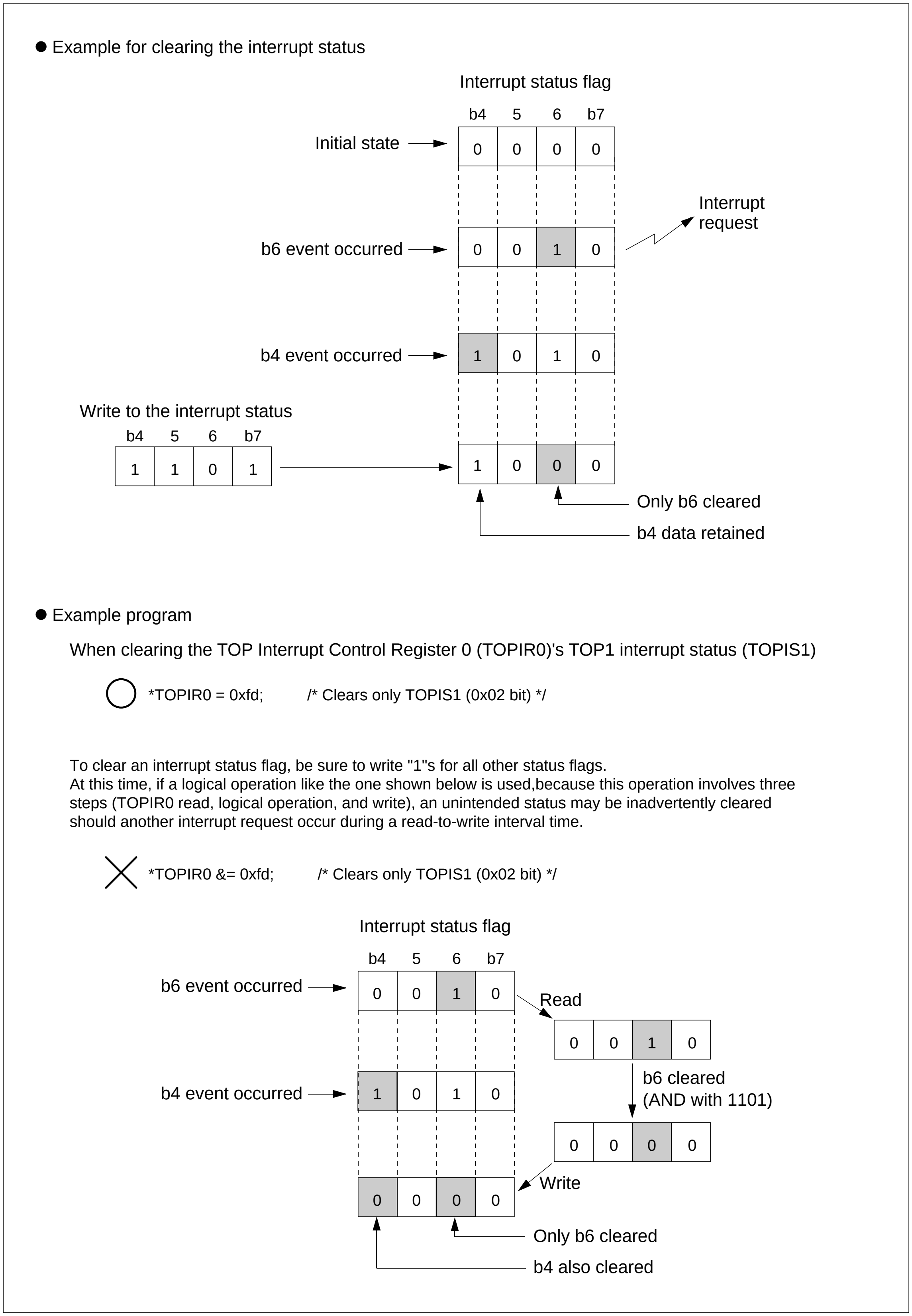


**Figure 10.2.4 Interrupt Status Register and Mask Register**



● Example program

When clearing the TOP Interrupt Control Register 0 (TOPIR0)'s TOP1 interrupt status (TOPIS1)

```
○ *TOPIR0 = 0xfd;        /* Clears only TOPIS1 (0x02 bit) */
```

To clear an interrupt status flag, be sure to write "1"s for all other status flags.
At this time, if a logical operation like the one shown below is used,because this operation involves three steps (TOPIR0 read, logical operation, and write), an unintended status may be inadvertently cleared should another interrupt request occur during a read-to-write interval time.

```
× *TOPIR0 &= 0xfd;        /* Clears only TOPIS1 (0x02 bit) */
```

**Figure 10.2.5 Example for Clearing the Interrupt Status**

The table below shows the relationship between the interrupt signals generated by multijunction timers and the interrupt sources input to the interrupt controller.

**Table 10.2.6 Interrupt Signals Generated by MJT**

| Signal Name | Source of Interrupt Generated | Interrupt Sources Input to ICU (Note 1) | Number of Input Sources |
|---|---|---|---|
| IRQ0 | TIO0, TIO1, TIO2, TIO3 | MJT output interrupt 0 | 4 |
| IRQ1 | TOP6, TOP7 | MJT output interrupt 1 | 2 |
| IRQ2 | TOP0, TOP1, TOP2, TOP3, TOP4, TOP5 | MJT output interrupt 2 | 6 |
| IRQ3 | TIO8, TIO9 | MJT output interrupt 3 | 2 |
| IRQ4 | TIO4, TIO5, TIO6, TIO7 | MJT output interrupt 4 | 4 |
| IRQ6 | TOP8, TOP9 | MJT output interrupt 6 | 2 |
| IRQ7 | TMS0, TMS1 | MJT output interrupt 7 | 2 |
| IRQ9 | TIN0 | MJT input interrupt 1 | 1 |
| IRQ10 | TIN16, TIN17, TIN18, TIN19 | MJT input interrupt 2 | 4 |
| IRQ11 | TIN20, TIN21, TIN22, TIN23 | MJT input interrupt 3 | 4 |
| IRQ12 | TIN3 | MJT input interrupt 4 | 1 |

Note 1: Refer to Chapter 5, "Interrupt Controller (ICU)."

Note: • For TOP10, there are no interrupt status and mask bits in MJT interrupt control register because it only has one source of interrupt in the group. (It is controlled directly by the interrupt controller.)

## ■ TOP Interrupt Control Register 0 (TOPIR0)

<Address: H'0080 0230>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  | TOPIS5 | TOPIS4 | TOPIS3 | TOPIS2 | TOPIS1 | TOPIS0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | No functions assigned |  | 0 | — |
| 2 | TOPIS5 (TOP5 interrupt status) | 0 : No interrupt request<br>1 : Interrupt request generated | ○ | △ |
| 3 | TOPIS4 (TOP4 interrupt status) |  |  |  |
| 4 | TOPIS3 (TOP3 interrupt status) |  |  |  |
| 5 | TOPIS2 (TOP2 interrupt status) |  |  |  |
| 6 | TOPIS1 (TOP1 interrupt status) |  |  |  |
| 7 | TOPIS0 (TOP0 interrupt status) |  |  |  |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

## ■ TOP Interrupt Control Register 1 (TOPIR1)

<Address: H'0080 0231>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  | TOPIM5 | TOPIM4 | TOPIM3 | TOPIM2 | TOPIM1 | TOPIM0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8, 9 | No functions assigned. |  | 0 | — |
| 10 | TOPIM5 (TOP5 interrupt mask) | 0 : Enables interrupt request<br>1 : Masks (disables) interrupt request | ○ | ○ |
| 11 | TOPIM4 (TOP4 interrupt mask) |  |  |  |
| 12 | TOPIM3 (TOP3 interrupt mask) |  |  |  |
| 13 | TOPIM2 (TOP2 interrupt mask) |  |  |  |
| 14 | TOPIM1 (TOP1 interrupt mask) |  |  |  |
| 15 | TOPIM0 (TOP0 interrupt mask) |  |  |  |

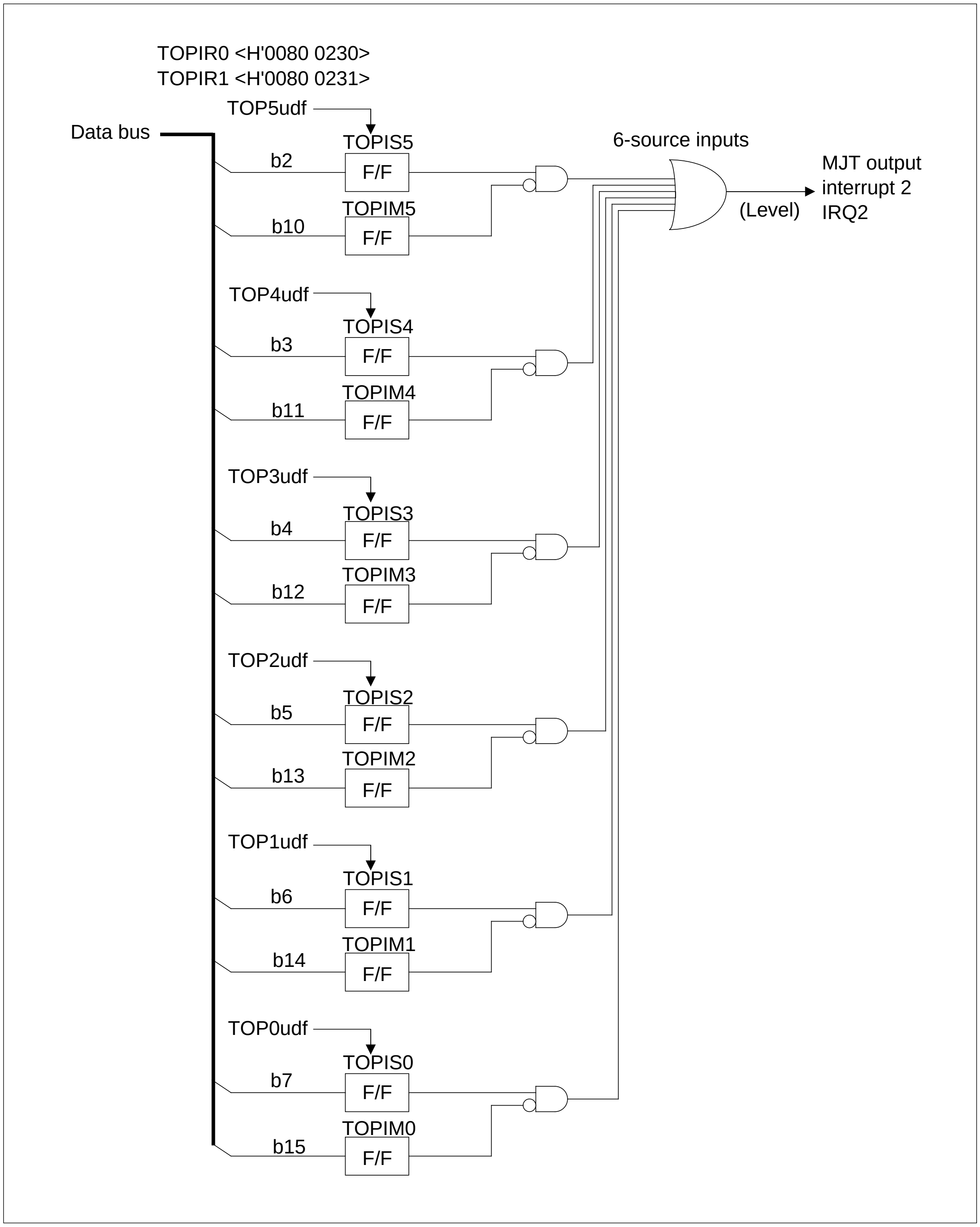


**Figure 10.2.6 Block Diagram of MJT Output Interrupt 2**

### ■ TOP Interrupt Control Register 2 (TOPIR2)

<Address: H'0080 0232>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  | TOPIS7 | TOPIS6 |  |  | TOPIM7 | TOPIM6 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | No functions assigned |  | 0 | — |
| 2 | TOPIS7 (TOP7 interrupt status) | 0 : No interrupt request | ○ | △ |
| 3 | TOPIS6 (TOP6 interrupt status) | 1 : Interrupt request generated |  |  |
| 4, 5 | No functions assigned |  | 0 | — |
| 6 | TOPIM7 (TOP7 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 7 | TOPIM6 (TOP6 interrupt mask) | 1 : Masks (disables) interrupt request |  |  |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

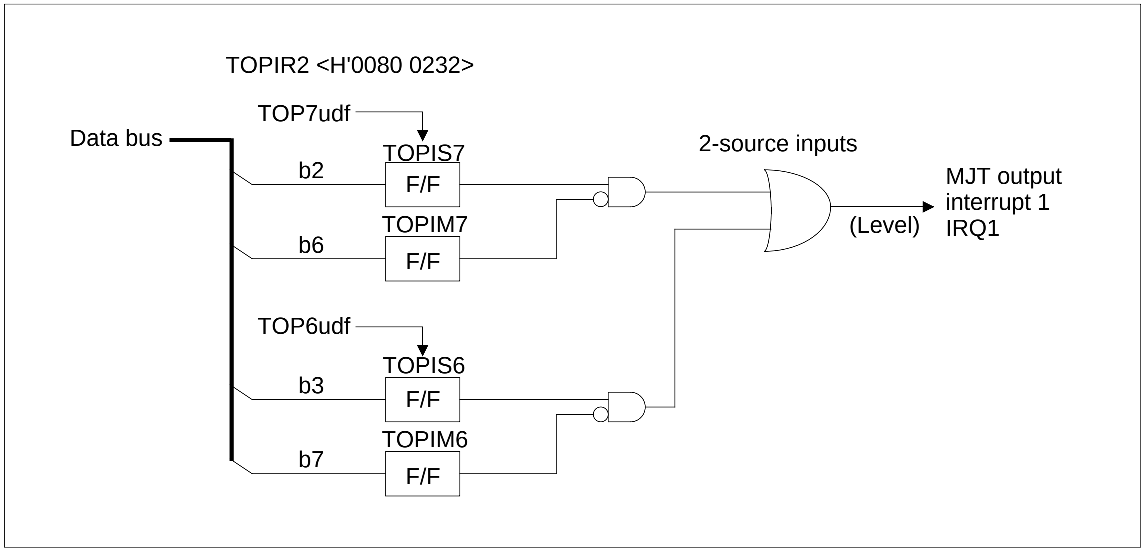


Figure 10.2.7 Block Diagram of MJT Output Interrupt 1

### ■ TOP Interrupt Control Register 3 (TOPIR3)

<Address: H'0080 0233>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | TOPIS9 | TOPIS8 | | | TOPIM9 | TOPIM8 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8, 9 | No functions assigned | | 0 | — |
| 10 | TOPIS9 (TOP9 interrupt status) | 0 : No interrupt request | ○ | △ |
| 11 | TOPIS8 (TOP8 interrupt status) | 1 : Interrupt request generated | | |
| 12, 13 | No functions assigned | | 0 | — |
| 14 | TOPIM9 (TOP9 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 15 | TOPIM8 (TOP8 interrupt mask) | 1 : Masks (disables) interrupt request | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

Note: • For TOP10, there are no interrupt status and mask bits in MJT interrupt control registers because it only has one source of interrupt in the group. (It is controlled directly by the interrupt controller.)

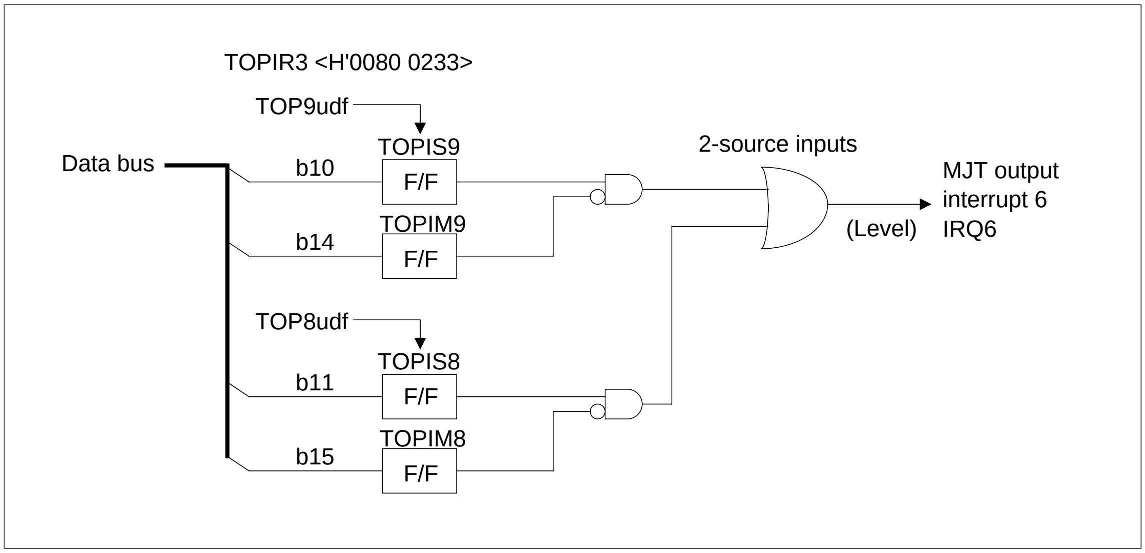


**Figure 10.2.8 Block Diagram of MJT Output Interrupt 6**

## ■ TIO Interrupt Control Register 0 (TIOIR0)

<Address: H'0080 0234>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| TIOIS3 | TIOIS2 | TIOIS1 | TIOIS0 | TIOIM3 | TIOIM2 | TIOIM1 | TIOIM0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | TIOIS3 (TIO3 interrupt status) | 0 : No interrupt request | ○ | △ |
| 1 | TIOIS2 (TIO2 interrupt status) | 1 : Interrupt request generated | | |
| 2 | TIOIS1 (TIO1 interrupt status) | | | |
| 3 | TIOIS0 (TIO0 interrupt status) | | | |
| 4 | TIOIM3 (TIO3 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 5 | TIOIM2 (TIO2 interrupt mask) | 1 : Masks (disables) interrupt request | | |
| 6 | TIOIM1 (TIO1 interrupt mask) | | | |
| 7 | TIOIM0 (TIO0 interrupt mask) | | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

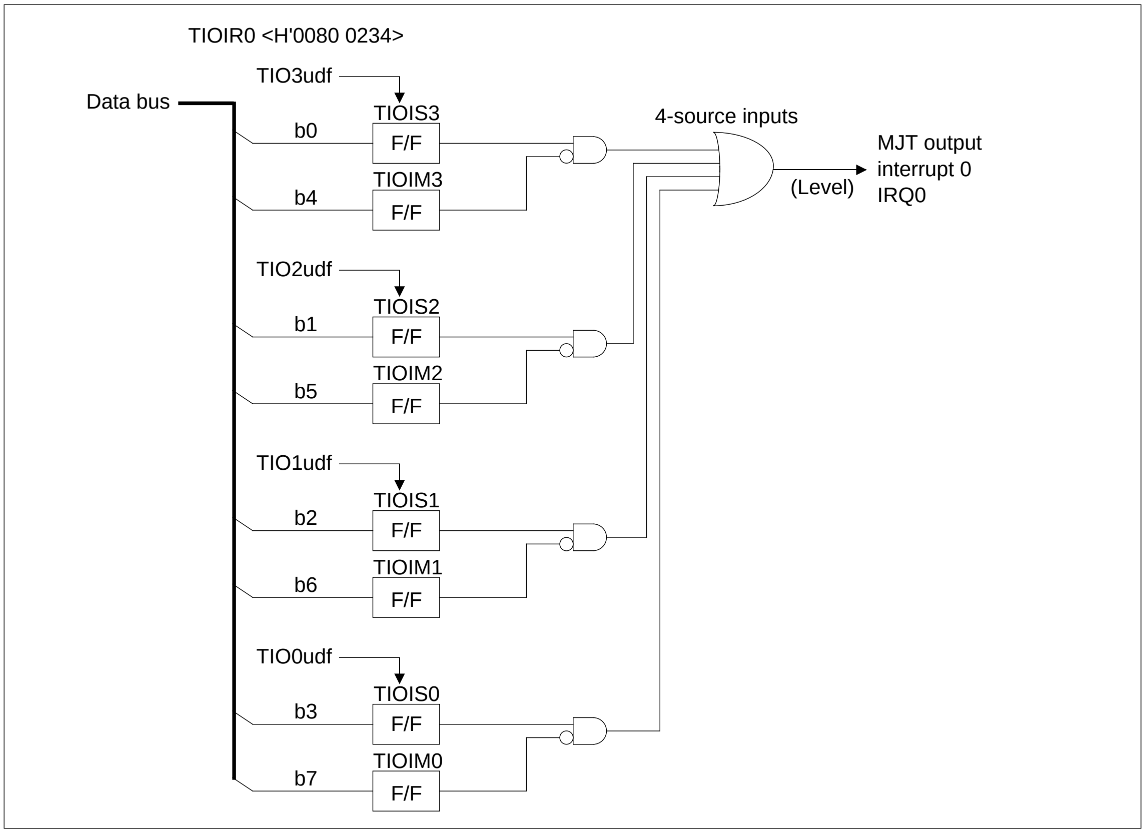


**Figure 10.2.9 Block Diagram of MJT Output Interrupt 0**

### ■ TIO Interrupt Control Register 1 (TIOIR1) <Address: H'0080 0235>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TIOIS7 | TIOIS6 | TIOIS5 | TIOIS4 | TIOIM7 | TIOIM6 | TIOIM5 | TIOIM4 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | TIOIS7 (TIO7 interrupt status) | 0 : No interrupt request<br>1 : Interrupt request generated | ○ | △ |
| 9 | TIOIS6 (TIO6 interrupt status) | | | |
| 10 | TIOIS5 (TIO5 interrupt status) | | | |
| 11 | TIOIS4 (TIO4 interrupt status) | | | |
| 12 | TIOIM7 (TIO7 interrupt mask) | 0 : Enables interrupt request<br>1 : Masks (disables) interrupt request | ○ | ○ |
| 13 | TIOIM6 (TIO6 interrupt mask) | | | |
| 14 | TIOIM5 (TIO5 interrupt mask) | | | |
| 15 | TIOIM4 (TIO4 interrupt mask) | | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

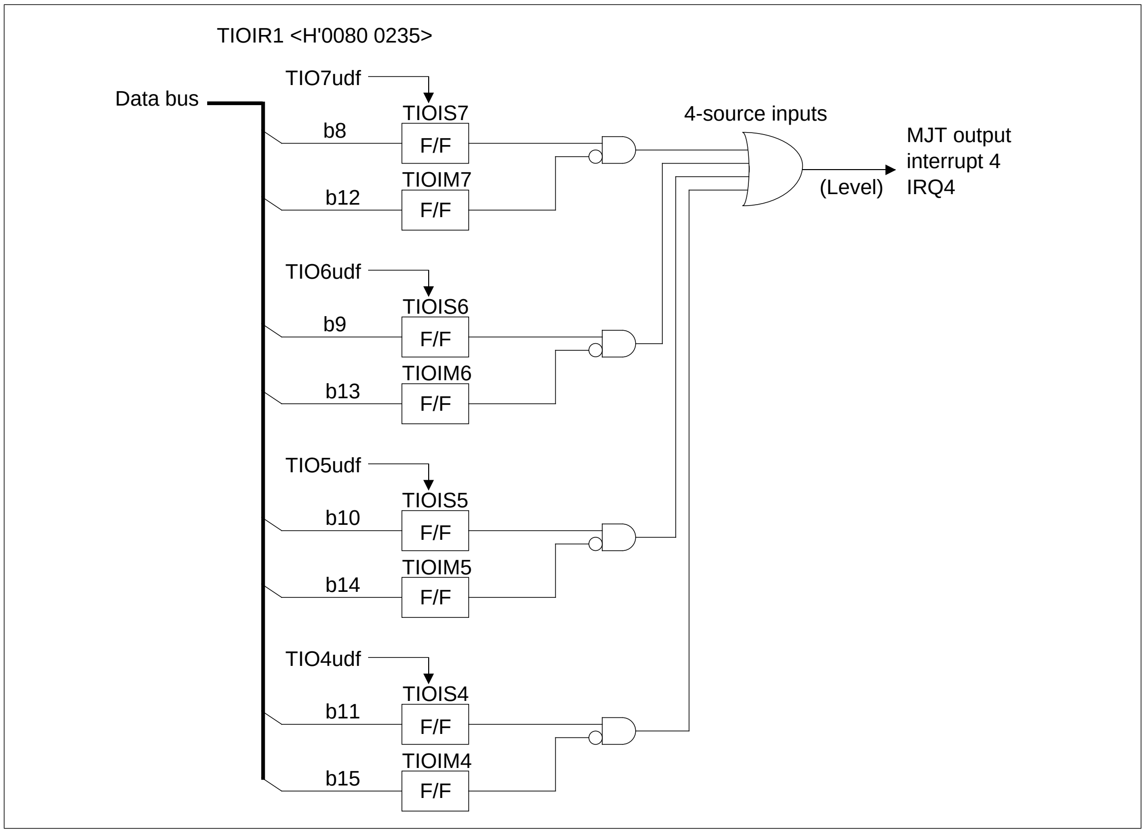


**Figure 10.2.10 Block Diagram of MJT Output Interrupt 4**

## ■ TIO Interrupt Control Register 2 (TIOIR2)

<Address: H'0080 0236>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  | TIOIS9 | TIOIS8 |  |  | TIOIM9 | TIOIM8 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | No functions assigned |  | 0 | — |
| 2 | TIOIS9 (TIO9 interrupt status) | 0 : No interrupt request | ○ | △ |
| 3 | TIOIS8 (TIO8 interrupt status) | 1 : Interrupt request generated |  |  |
| 4, 5 | No functions assigned |  | 0 | — |
| 6 | TIOIM9 (TIO9 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 7 | TIOIM8 (TIO8 interrupt mask) | 1 : Masks (disables) interrupt request |  |  |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

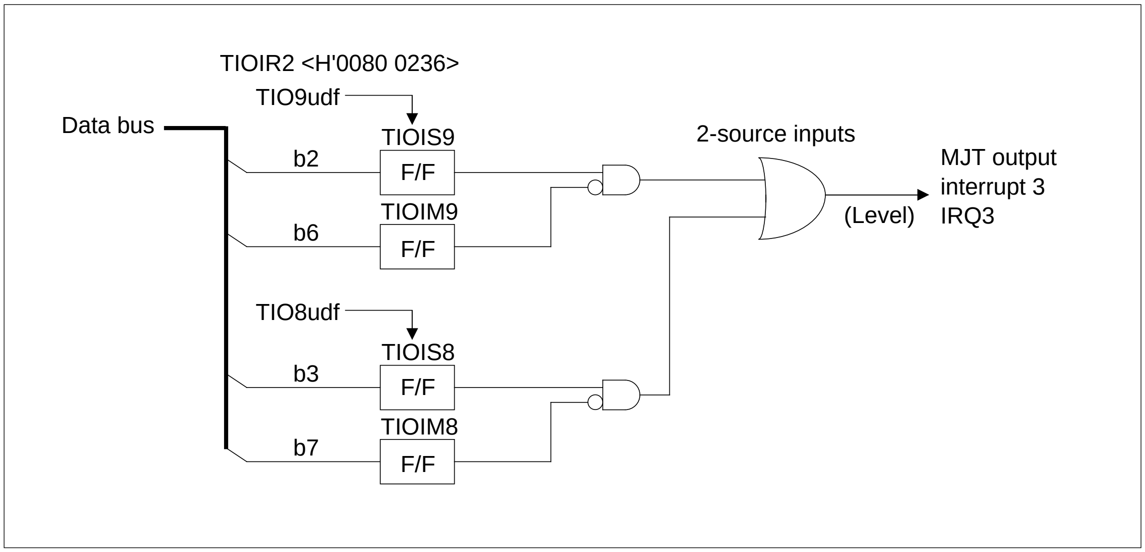


**Figure 10.2.11 Block Diagram of MJT Output Interrupt 3**

## ■ TMS Interrupt Control Register (TMSIR)

<Address: H'0080 0237>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | TMSIS1 | TMSIS0 | | | TMSIM1 | TMSIM0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8, 9 | No functions assigned | | 0 | — |
| 10 | TMSIS1 (TMS1 interrupt status) | 0 : No interrupt request | ○ | △ |
| 11 | TMSIS0 (TMS0 interrupt status) | 1 : Interrupt request generated | | |
| 12, 13 | No functions assigned | | 0 | — |
| 14 | TMSIM1 (TMS1 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 15 | TMSIM0 (TMS0 interrupt mask) | 1 : Masks (disables) interrupt request | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

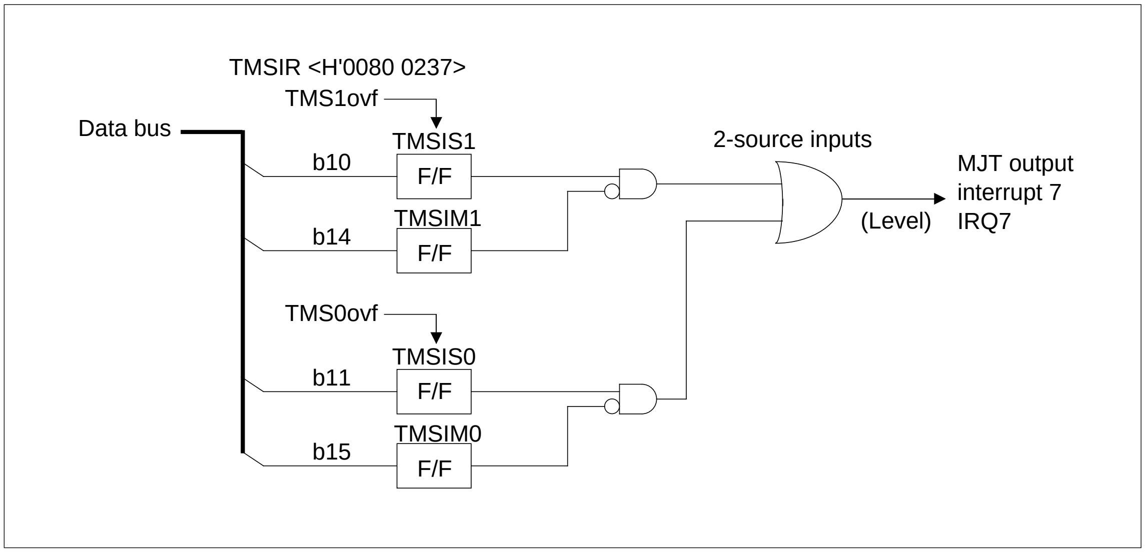


**Figure 10.2.12 Block Diagram of MJT Output Interrupt 7**

## ■ TIN Interrupt Control Register 0 (TINIR0)

<Address: H'0080 0238>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  | TINIS0 |  | TINIM2 | TINIM1 | TINIM0 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 2 | No functions assigned |  | 0 | — |
| 3 | TINIS0 (TIN0 interrupt status) | 0 : No interrupt request<br>1 : Interrupt request generated | ○ | △ |
| 4 | No functions assigned |  | 0 | — |
| 5 | TINIM2 (reserved) | Setting this bit has no effect | ○ | ○ |
| 6 | TINIM1 (reserved) | Setting this bit has no effect | ○ | ○ |
| 7 | TINIM0 (TIN0 interrupt mask) | 0 : Enables interrupt request<br>1 : Masks (disables) interrupt request | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

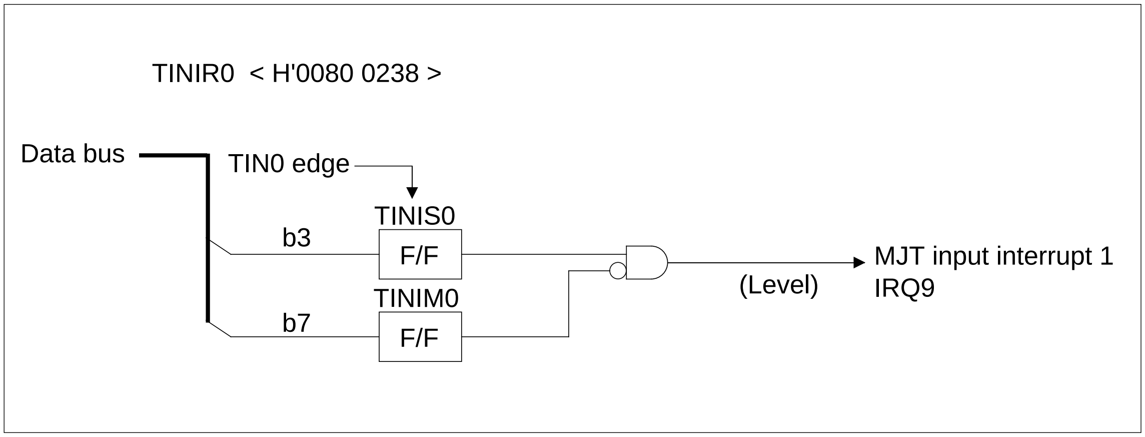


**Figure 10.2.13 Block Diagram of MJT Input Interrupt 1**

### ■ TIN Interrupt Control Register 1 (TINIR1)

<Address: H'0080 0239>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  | TINIS3 | TINIM6 | TINIM5 | TINIM4 | TINIM3 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | No functions assigned |  | 0 | — |
| 11 | TINIS3 (TIN3 interrupt status) | 0 : No interrupt request<br>1 : Interrupt request generated | ○ | △ |
| 12 | TINIM6 (reserved) | Setting this bit has no effect | ○ | ○ |
| 13 | TINIM5 (reserved) |  |  |  |
| 14 | TINIM4 (reserved) |  |  |  |
| 15 | TINIM3 (TIN3 interrupt mask) | 0 : Enables interrupt request<br>1 : Masks (disables) interrupt request | ○ | ○ |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

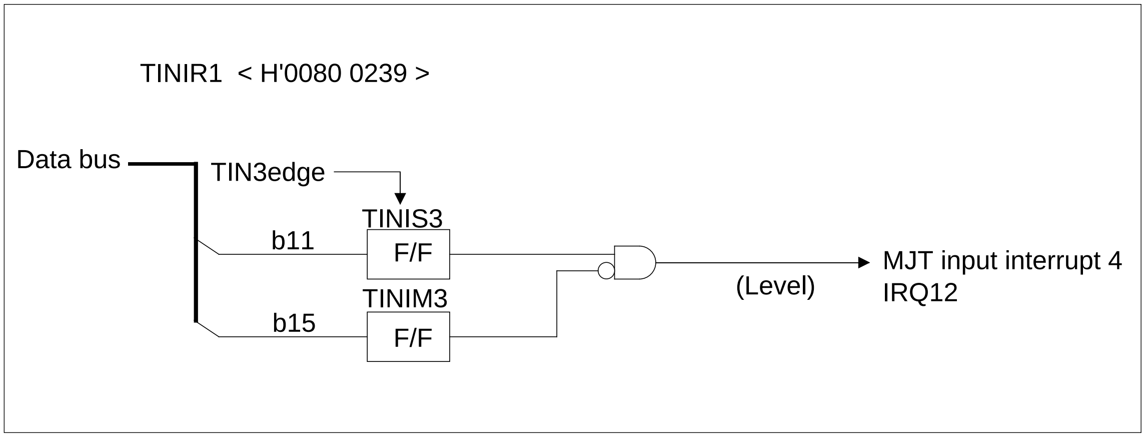


**Figure 10.2.14 Block Diagram of MJT Input Interrupt 4**

### ■ TIN Interrupt Control Register 4 (TINIR4)

<Address: H'0080 023C>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| TINIS19 | TINIS18 | TINIS17 | TINIS16 |  |  |  |  |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | TINIS19 (TIN19 interrupt status) | 0 : No interrupt request | ○ | △ |
| 1 | TINIS18 (TIN18 interrupt status) | 1 : Interrupt request generated |  |  |
| 2 | TINIS17 (TIN17 interrupt status) |  |  |  |
| 3 | TINIS16 (TIN16 interrupt status) |  |  |  |
| 4 - 7 | No functions assigned |  | 0 | — |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

### ■ TIN Interrupt Control Register 5 (TINIR5)

<Address: H'0080 023D>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TINIM19 | TINIM18 | TINIM17 | TINIM16 | TINIM15 | TINIM14 | TINIM13 | TINIM12 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | TINIM19 (TIN19 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 9 | TINIM18 (TIN18 interrupt mask) | 1 : Masks (disables) interrupt request |  |  |
| 10 | TINIM17 (TIN17 interrupt mask) |  |  |  |
| 11 | TINIM16 (TIN16 interrupt mask) |  |  |  |
| 12 | TINIM15 (reserved) | Setting this bit has no effect | ○ | ○ |
| 13 | TINIM14 (reserved) |  |  |  |
| 14 | TINIM13 (reserved) |  |  |  |
| 15 | TINIM12 (reserved) |  |  |  |

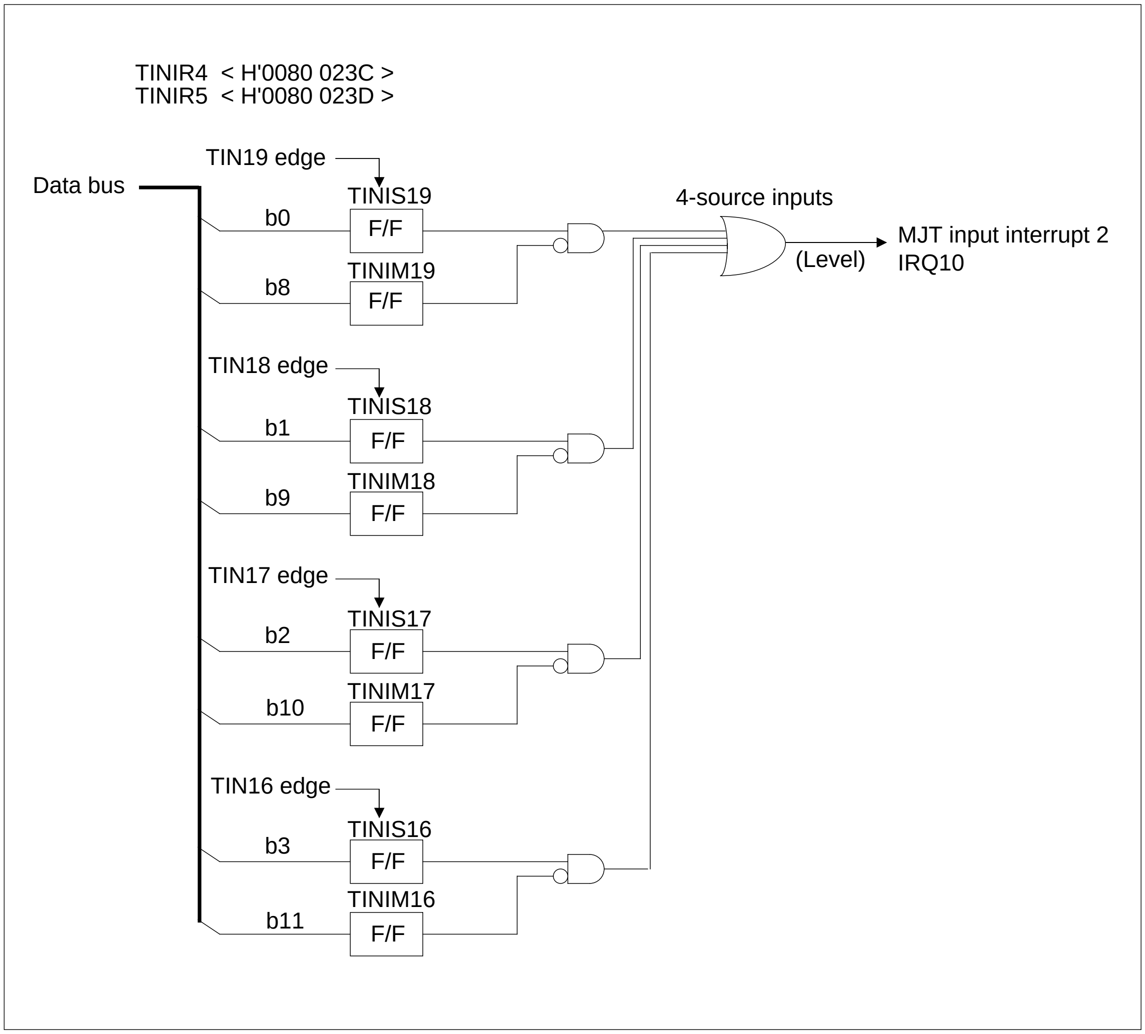


**Figure 10.2.15 Block Diagram of MJT Input Interrupt 2**

### ■ TIN Interrupt Control Register 6 (TINIR6) <Address: H'0080 023E>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| TINIS23 | TINIS22 | TINIS21 | TINIS20 | TINIM23 | TINIM22 | TINIM21 | TINIM20 |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | TINIS23 (TIN23 interrupt status) | 0 : No interrupt request | ○ | △ |
| 1 | TINIS22 (TIN22 interrupt status) | 1 : Interrupt request generated | | |
| 2 | TINIS21 (TIN21 interrupt status) | | | |
| 3 | TINIS20 (TIN20 interrupt status) | | | |
| 4 | TINIM23 (TIN23 interrupt mask) | 0 : Enables interrupt request | ○ | ○ |
| 5 | TINIM22 (TIN22 interrupt mask) | 1 : Masks (disables) interrupt request | | |
| 6 | TINIM21 (TIN21 interrupt mask) | | | |
| 7 | TINIM20 (TIN20 interrupt mask) | | | |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

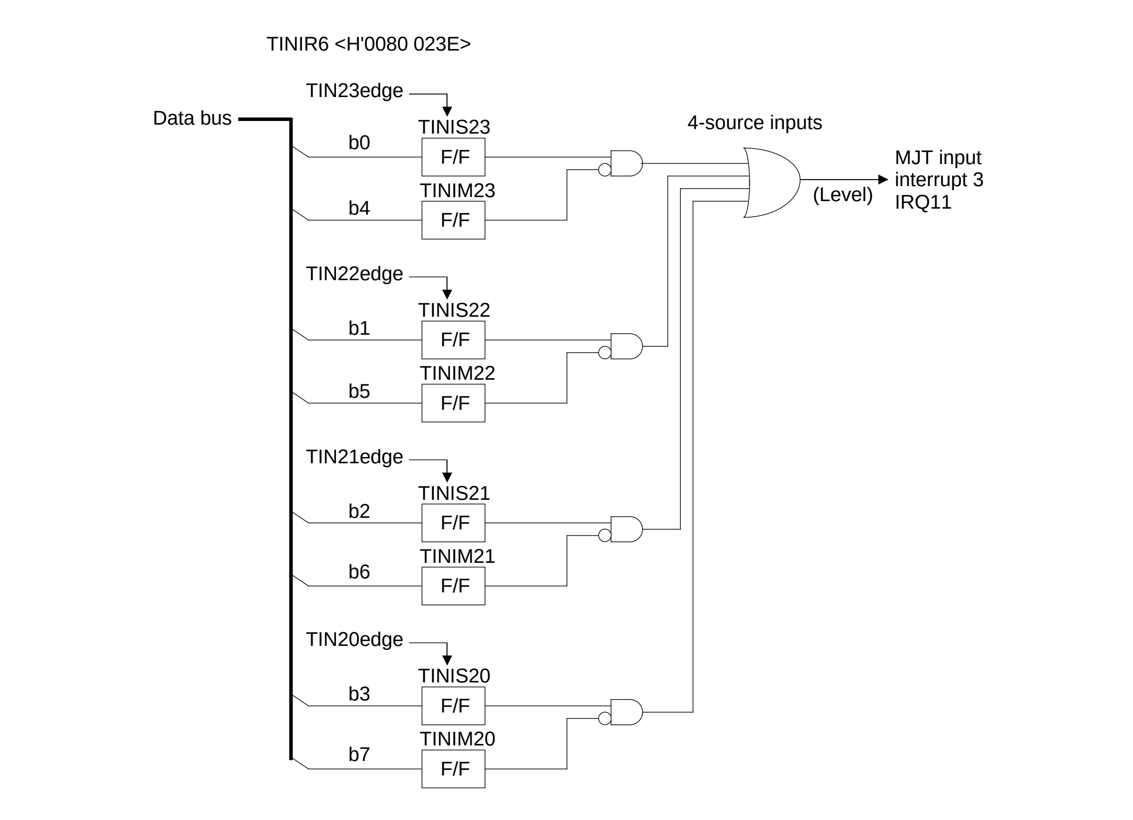


**Figure 10.2.16 Block Diagram of MJT Input Interrupt 3**

## 10.3 TOP (Output-related 16-bit Timer)

### 10.3.1 Outline of TOP

TOP (Timer Output) is an output-related 16-bit timer, whose operation mode can be selected from the following by mode switching in software:

- Single-shot output mode
- Delayed single-shot output mode
- Continuous output mode

The table below shows specifications of TOP. The diagram in the next page shows a block diagram of TOP.

**Table 10.3.1 Specifications of TOP (Output-related 16-bit Timer)**

| Item | Specification |
|---|---|
| Number of channels | 11 channels |
| Counter | 16-bit down-counter |
| Reload register | 16-bit reload register |
| Correction register | 16-bit correction register |
| Timer startup | Started by writing to enable bit in software or by enabling with external input (rising or falling edge or both) |
| Mode selection | <With correction function><br>• Single-shot output mode<br>• Delayed single-shot output mode<br><Without correction function><br>• Continuous output mode |
| Interrupt generation | Can be generated by a counter underflow |

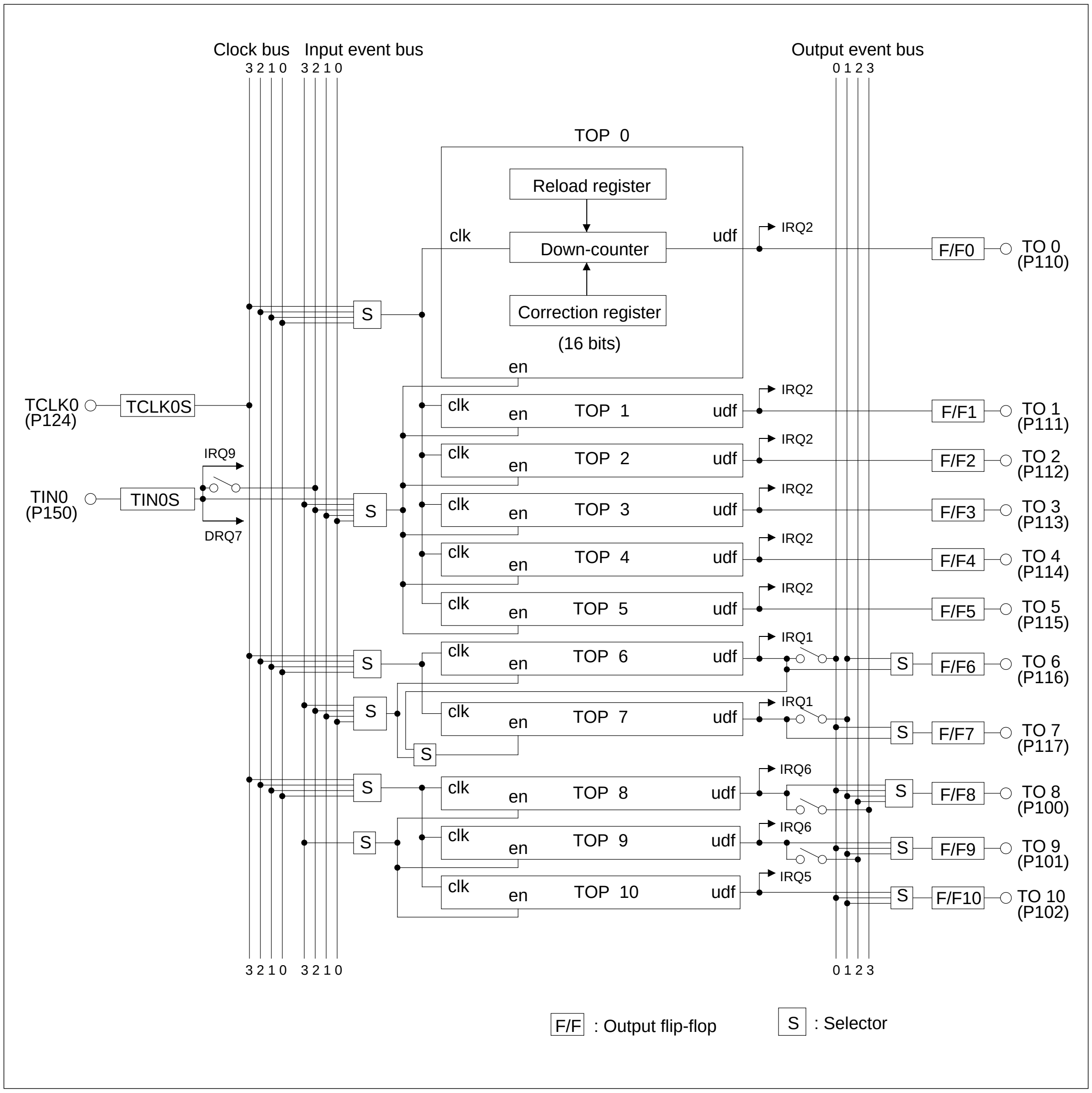


**Figure 10.3.1 Block Diagram of TOP (Output-related 16-bit Timer)**

### 10.3.2 Outline of Each Mode of TOP

Each mode of TOP is outlined below. For each TOP channel, only one of the following modes can be selected.

#### (1) Single-shot output mode

In single-shot output mode, the timer generates a pulse in width of (reload register set value + 1) only once and then stops.

When after setting the reload register, the timer is enabled (by writing to the enable bit in software or by external input), the content of the reload register is loaded into the counter synchronously with the count clock, letting the counter start counting. The counter counts down clock pulses and stops when it underflows after reaching the minimum count.

The F/F output waveform in single-shot output mode is inverted at startup and upon underflow, generating a single-shot pulse waveform in width of (reload register set value + 1) only once. Also, an interrupt can be generated when the counter underflows.

#### (2) Delayed single-shot output mode

In delayed single-shot output mode, the timer generates a pulse in width of (reload register set value + 1) only once, with the output delayed by an amount of time equal to (counter set value + 1) and then stops.

When after setting the counter and reload register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock. The first time the counter underflows, the reload register value is loaded into the counter causing it to continue counting down, and the counter stops when it underflows next time.

The F/F output waveform in delayed single-shot output mode is inverted when the counter underflows first time and next, generating a single-shot pulse waveform in width of (reload register set value + 1) only once, with the output delayed by an amount of time equal to (first set value of counter + 1) . Also, an interrupt can be generated when the counter underflows first time and next.

#### (3) Continuous output mode

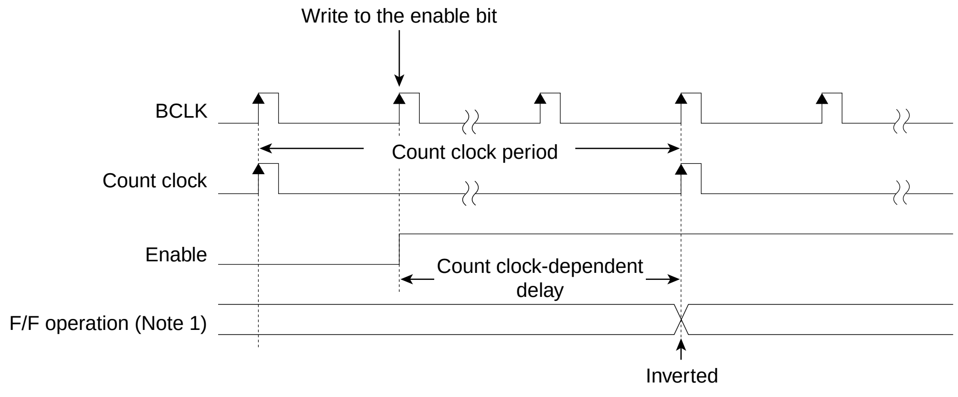
In continuous output mode, the timer counts down clock pulses starting from the set value of the counter and when the counter underflows, reloads it with the reload register value. Thereafter, this operation is repeated each time the counter underflows, thus generating consecutive pulses whose waveform is inverted in width of (reload register set value + 1).

When after setting the counter and reload register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock and when the minimum count is reached, generates an underflow. This underflow causes the counter to be reloaded with the content of the reload register and start counting over again. Thereafter, this operation is repeated each time an underflow occurs. To stop the counter, disable count by writing to the enable bit in software.

The F/F output waveform in continuous output mode is inverted at startup and upon underflow, generating consecutive pulses until the timer stops counting. Also, an interrupt can be generated each time the counter underflows.

### <Count clock-dependent delay>

- Because the timer operates synchronously witth the count clock, there is a count clock-dependent delay from when the timer is enabled till it actually starts operating. In operation mode where the F/F output is inverted when the timer is enabled, the F/F output is inverted synchronously with the count clock.



Note 1: This applies to the case where F/F output is inverted when the timer is enabled.

**Figure 10.3.2 Count clock Dependent Delay**

### 10.3.3 TOP Related Register Map

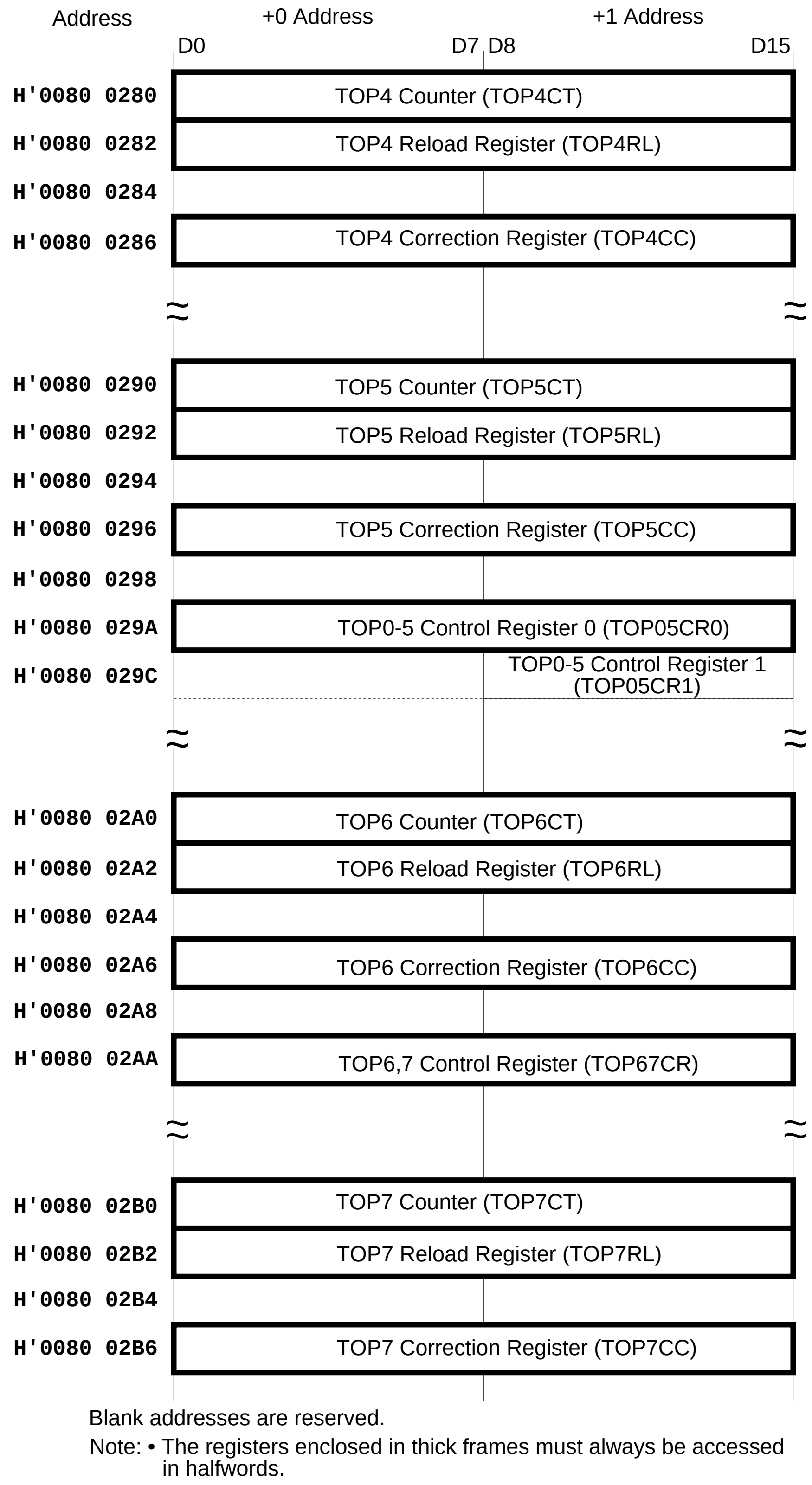
The diagram below shows a TOP-related register map.

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 0240 | TOP0 Counter (TOP0CT) | |
| H'0080 0242 | TOP0 Reload Register (TOP0RL) | |
| H'0080 0244 | | |
| H'0080 0246 | TOP0 Correction Register (TOP0CC) | |
| ≈ | | ≈ |
| H'0080 0250 | TOP1 Counter (TOP1CT) | |
| H'0080 0252 | TOP1 Reload Register (TOP1RL) | |
| H'0080 0254 | | |
| H'0080 0256 | TOP1 Correction Register (TOP1CC) | |
| ≈ | | ≈ |
| H'0080 0260 | TOP2 Counter (TOP2CT) | |
| H'0080 0262 | TOP2 Reload Register (TOP2RL) | |
| H'0080 0264 | | |
| H'0080 0266 | TOP2 Correction Register (TOP2CC) | |
| ≈ | | ≈ |
| H'0080 0270 | TOP3 Counter (TOP3CT) | |
| H'0080 0272 | TOP3 Reload Register (TOP3RL) | |
| H'0080 0274 | | |
| H'0080 0276 | TOP3 Correction Register (TOP3CC) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 10.3.3 TOP Related Register Map (1/3)**

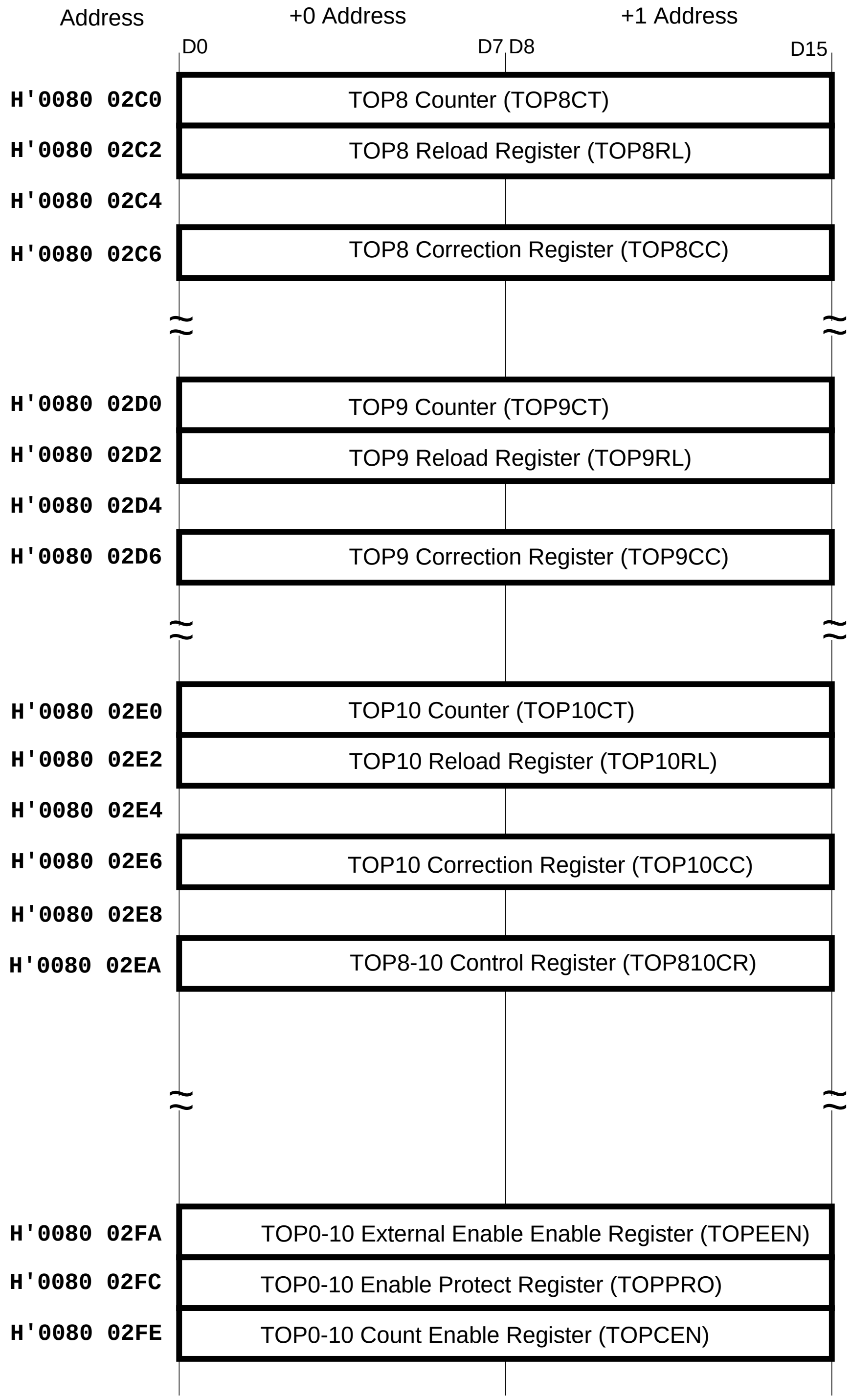


| Address | +0 Address<br>D0 … D7 | +1 Address<br>D8 … D15 |
|---|---|---|
| H'0080 0280 | TOP4 Counter (TOP4CT) | |
| H'0080 0282 | TOP4 Reload Register (TOP4RL) | |
| H'0080 0284 | | |
| H'0080 0286 | TOP4 Correction Register (TOP4CC) | |
| ≈ | | |
| H'0080 0290 | TOP5 Counter (TOP5CT) | |
| H'0080 0292 | TOP5 Reload Register (TOP5RL) | |
| H'0080 0294 | | |
| H'0080 0296 | TOP5 Correction Register (TOP5CC) | |
| H'0080 0298 | | |
| H'0080 029A | TOP0-5 Control Register 0 (TOP05CR0) | |
| H'0080 029C | | TOP0-5 Control Register 1 (TOP05CR1) |
| ≈ | | |
| H'0080 02A0 | TOP6 Counter (TOP6CT) | |
| H'0080 02A2 | TOP6 Reload Register (TOP6RL) | |
| H'0080 02A4 | | |
| H'0080 02A6 | TOP6 Correction Register (TOP6CC) | |
| H'0080 02A8 | | |
| H'0080 02AA | TOP6,7 Control Register (TOP67CR) | |
| ≈ | | |
| H'0080 02B0 | TOP7 Counter (TOP7CT) | |
| H'0080 02B2 | TOP7 Reload Register (TOP7RL) | |
| H'0080 02B4 | | |
| H'0080 02B6 | TOP7 Correction Register (TOP7CC) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 10.3.4 TOP Related Register Map (2/3)**



| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 02C0 | TOP8 Counter (TOP8CT) | |
| H'0080 02C2 | TOP8 Reload Register (TOP8RL) | |
| H'0080 02C4 | | |
| H'0080 02C6 | TOP8 Correction Register (TOP8CC) | |
| ≈ | | ≈ |
| H'0080 02D0 | TOP9 Counter (TOP9CT) | |
| H'0080 02D2 | TOP9 Reload Register (TOP9RL) | |
| H'0080 02D4 | | |
| H'0080 02D6 | TOP9 Correction Register (TOP9CC) | |
| ≈ | | ≈ |
| H'0080 02E0 | TOP10 Counter (TOP10CT) | |
| H'0080 02E2 | TOP10 Reload Register (TOP10RL) | |
| H'0080 02E4 | | |
| H'0080 02E6 | TOP10 Correction Register (TOP10CC) | |
| H'0080 02E8 | | |
| H'0080 02EA | TOP8-10 Control Register (TOP810CR) | |
| ≈ | | ≈ |
| H'0080 02FA | TOP0-10 External Enable Enable Register (TOPEEN) | |
| H'0080 02FC | TOP0-10 Enable Protect Register (TOPPRO) | |
| H'0080 02FE | TOP0-10 Count Enable Register (TOPCEN) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 10.3.5 TOP Related Register Map (3/3)**

### 10.3.4 TOP Control Registers

The TOP control registers are used to select operation modes of TOP0-10 (single-shot, delayed single-shot, or continuous mode), as well as select the counter enable and counter clock sources. Following four TOP control registers are provided for each timer group.

- TOP0-5 Control Register 0 (TOP05CR0)
- TOP0-5 Control Register 1 (TOP05CR1)
- TOP6, 7 Control Register (TOP67CR)
- TOP8-10 Control Register (TOP810CR)

### ■ TOP0-5 Control Register 0 (TOP05CR0)

<Address:H'0080 029A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOP3M | | TOP2M | | TOP1M | | TOP0M | | | TOP05ENS | | | | | TOP05CKS | |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0,1 | TOP3M (TOP3 operation mode selection) | 00: Single-shot output mode<br>01: Delayed single-shot output mode<br>1X: Continuous output mode | ○ | ○ |
| 2,3 | TOP2M (TOP2 operation mode selection) | | | |
| 4,5 | TOP1M (TOP1 operation mode selection) | | | |
| 6,7 | TOP0M (TOP0 operation mode selection) | | | |
| 8 | No functions assigned | | 0 | – |
| 9-10 | TOP05ENS<br>(TOP0-5 enable source selection) | 0XX: External TIN0 input<br>100: Input event bus 0<br>101: Input event bus 1<br>110: Input event bus 2<br>111: Input event bus 3 | ○ | ○ |
| 12,13 | No functions assigned | | 0 | – |
| 14,15 | TOP05CKS<br>(TOP0-5 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |

Notes: • This register must always be accessed in halfwords.

• Always make sure the counter has stopped and is idle before setting or changing operation modes.

## ■ TOP0-5 Control Register 1 (TOP05CR1)

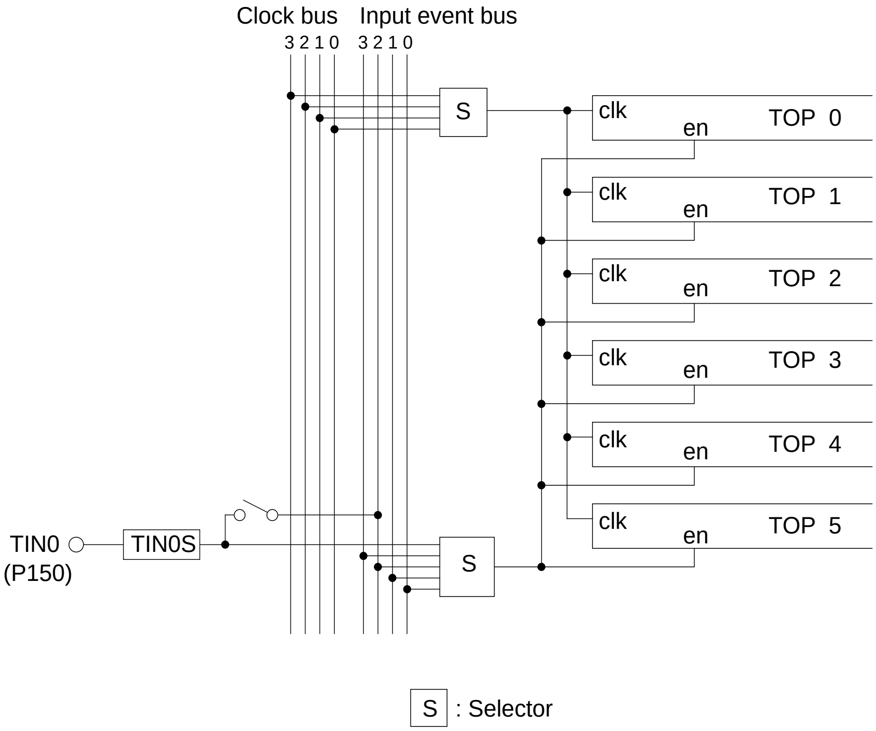
<Address:H'0080 029D>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | | | TOP5M | | TOP4M | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-11 | No functions assigned | | 0 | – |
| 12,13 | TOP5M (TOP5 operation mode selection) | 00: Single-shot output mode | ○ | ○ |
| 14,15 | TOP4M (TOP4 operation mode selection) | 01: Delayed single-shot output mode | | |
| | | 1X: Continuous output mode | | |

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.



Note: • This diagram is shown for the explanation of TOP control registers, and is partly omitted.

**Figure 10.3.6 Outline Diagram of TOP0-5 Clock/Enable Inputs**

### ■ TOP6,7 Control Register (TOP67CR)

<Address:H'0080 02AA>

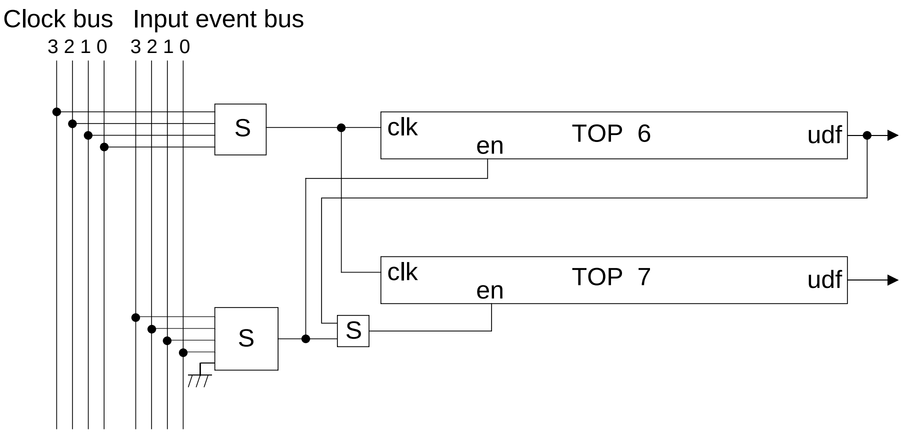
| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ▨ | TOP7<br>ENS | TOP7M | | ▨ | | TOP6M | | ▨ | TOP67ENS | | | ▨ | | TOP67CKS | |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | No functions assigned | | 0 | – |
| 1 | TOP7ENS<br>(TOP7 enable source selection) | 0: Result selected by TOP67ENS bit<br>1: TOP6 output | ○ | ○ |
| 2,3 | TOP7M (TOP7 operation mode selection) | 00: Single-shot output mode<br>01: Delayed single-shot output mode<br>1X: Continuous output mode | ○ | ○ |
| 4,5 | No functions assigned | | 0 | – |
| 6,7 | TOP6M (TOP6 operation mode selection) | 00: Single-shot output mode<br>01: Delayed single-shot output mode<br>1X: Continuous output mode | ○ | ○ |
| 8 | No functions assigned | | 0 | – |
| 9-11 | TOP67ENS<br>(TOP6, TOP7 enable source selection) | 0XX: No selection<br>100: Input event bus 0<br>101: Input event bus 1<br>110: Input event bus 2<br>111: Input event bus 3 | ○ | ○ |
| 12,13 | No functions assigned | | 0 | – |
| 14,15 | TOP67CKS<br>(TOP6, TOP7 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |

Notes: • This register must always be accessed in halfwords.

• Always make sure the counter has stopped and is idle before setting or changing operation modes.



S : Selector

Note: • This diagram is shown for the explanation of TOP control registers, and is partly omitted.

**Figure 10.3.7 Outline Diagram of TOP6, TOP7 Clock/Enable Inputs**

### ■ TOP8-10 Control Register (TOP810CR)

<Address:H'0080 02EA>

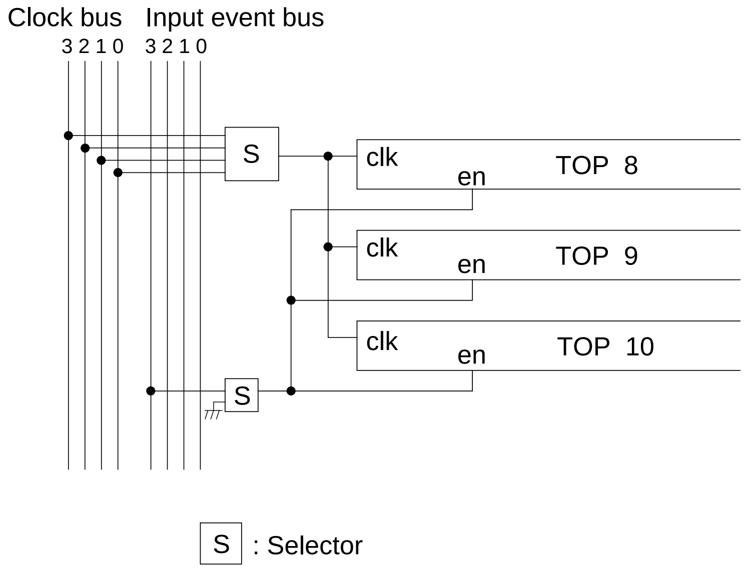
| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  | TOP10M |  | TOP9M |  | TOP8M |  |  |  |  | TOP<br>810<br>ENS |  |  | TOP810CKS |  |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0,1 | No functions assigned |  | 0 | – |
| 2,3 | TOP10M (TOP10 operation mode selection) | 00: Single-shot output mode<br>01: Delayed single-shot output mode<br>1X: Continuous output mode | ○ | ○ |
| 4,5 | TOP9M (TOP9 operation mode selection) |  |  |  |
| 6,7 | TOP8M (TOP8 operation mode selection) |  |  |  |
| 8-10 | No functions assigned |  | 0 | – |
| 11 | TOP810ENS<br>(TOP8-10 enable source selection) | 0: No selection<br>1: Input event bus 3 | ○ | ○ |
| 12,13 | No functions assigned |  | 0 | – |
| 14,15 | TOP810CKS<br>(TOP8-10 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>01: Clock bus 3 | ○ | ○ |

Notes: • This register must always be accessed in halfwords.

• Always make sure the counter has stopped and is idle before setting or changing operation modes.



Note: • This diagram is shown for the explanation of TOP control registers, and is partly omitted.

**Figure 10.3.8 Outline Diagram of TOP8-10 Clock/Enable Inputs**

### 10.3.5 TOP Counters (TOP0CT-TOP10CT)

| | |
|---|---|
| ■ **TOP0 Counter (TOP0CT)** | <Address:H'0080 0240> |
| ■ **TOP1 Counter (TOP1CT)** | <Address:H'0080 0250> |
| ■ **TOP2 Counter (TOP2CT)** | <Address:H'0080 0260> |
| ■ **TOP3 Counter (TOP3CT)** | <Address:H'0080 0270> |
| ■ **TOP4 Counter (TOP4CT)** | <Address:H'0080 0280> |
| ■ **TOP5 Counter (TOP5CT)** | <Address:H'0080 0290> |
| ■ **TOP6 Counter (TOP6CT)** | <Address:H'0080 02A0> |
| ■ **TOP7 Counter (TOP7CT)** | <Address:H'0080 02B0> |
| ■ **TOP8 Counter (TOP8CT)** | <Address:H'0080 02C0> |
| ■ **TOP9 Counter (TOP9CT)** | <Address:H'0080 02D0> |
| ■ **TOP10 Counter (TOP10CT)** | <Address:H'0080 02E0> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOP0CT-TOP10CT | | | | | | | | | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TOP0CT-TOP10CT | 16-bit counter value | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The TOP counters are a 16-bit down-counter. After the timer is enabled (by writing to the enable bit in software or by external input), the counter starts counting synchronously with the count clock.

### 10.3.6 TOP Reload Registers (TOP0RL-TOP10RL)

| | |
|---|---|
| ■ **TOP0 Reload Register (TOP0RL)** | <Address:H'0080 0242> |
| ■ **TOP1 Reload Register (TOP1RL)** | <Address:H'0080 0252> |
| ■ **TOP2 Reload Register (TOP2RL)** | <Address:H'0080 0262> |
| ■ **TOP3 Reload Register (TOP3RL)** | <Address:H'0080 0272> |
| ■ **TOP4 Reload Register (TOP4RL)** | <Address:H'0080 0282> |
| ■ **TOP5 Reload Register (TOP5RL)** | <Address:H'0080 0292> |
| ■ **TOP6 Reload Register (TOP6RL)** | <Address:H'0080 02A2> |
| ■ **TOP7 Reload Register (TOP7RL)** | <Address:H'0080 02B2> |
| ■ **TOP8 Reload Register (TOP8RL)** | <Address:H'0080 02C2> |
| ■ **TOP9 Reload Register (TOP9RL)** | <Address:H'0080 02D2> |
| ■ **TOP10 Reload Register (TOP10RL)** | <Address:H'0080 02E2> |

<table>
<tr><th>D0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>D15</th></tr>
<tr><td colspan="16">TOP0RL-TOP10RL</td></tr>
</table>

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TOP0RL-TOP10RL | 16-bit reload register value | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The TOP reload registers are used to load data into the TOP counter registers (TOP0CT-TOP10CT). It is in the following cases that the content of the reload register is loaded in the counter:

- When the counter is enabled in single-shot mode
- When the counter underflowed in delayed single-shot or continuous mode

Writing data to the reload register does not mean that the data is loaded into the counter simultaneously.

Note that data reloading after an underflow is performed synchronously with the clock period in which the counter underflowed.

### 10.3.7 TOP Correction Registers (TOP0CC-TOP10CC)

| | |
|---|---|
| ■ **TOP0 Correction Register (TOP0CC)** | <Address:H'0080 0246> |
| ■ **TOP1 Correction Register (TOP1CC)** | <Address:H'0080 0256> |
| ■ **TOP2 Correction Register (TOP2CC)** | <Address:H'0080 0266> |
| ■ **TOP3 Correction Register (TOP3CC)** | <Address:H'0080 0276> |
| ■ **TOP4 Correction Register (TOP4CC)** | <Address:H'0080 0286> |
| ■ **TOP5 Correction Register (TOP5CC)** | <Address:H'0080 0296> |
| ■ **TOP6 Correction Register (TOP6CC)** | <Address:H'0080 02A6> |
| ■ **TOP7 Correction Register (TOP7CC)** | <Address:H'0080 02B6> |
| ■ **TOP8 Correction Register (TOP8CC)** | <Address:H'0080 02C6> |
| ■ **TOP9 Correction Register (TOP9CC)** | <Address:H'0080 02D6> |
| ■ **TOP10 Correction Register (TOP10CC)** | <Address:H'0080 02E6> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOP0CC-TOP10CC | | | | | | | | | | | | | | | |

(Acceptable set values +32767- –32768)

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TOP0CC-TOP10CC | 16-bit correction register value | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The TOP correction registers are used to correct the TOP counter value by adding or subtracting it in the middle of operation. To increase or reduce the counter value, write a value to this correction register, the value by which you want to be increased or reduced from the initial count set in the counter. To add, write the value you want to add to the correction register directly as is; to subtract, write the two's complement of the value you want to subtract to the correction register.

Correction of the counter is performed synchronously with a clock period next to the one in which the correction value was written to the TOP correction register. In this case, one down-count in the clock period during which the correction was performed is canceled. Therefore, note that the counter value actually is corrected by (correction register value + 1). For example, if the initial counter value is 10 and you write a value 3 to the correction register when the counter has counted down to 5, then the counter underflows after a total of 15 counts.

### 10.3.8 TOP Enable Control Register

**■ TOP0-10 External Enable Permit Register (TOPEEN)** <Address:H'0080 02FA>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TOP10<br>EEN | TOP9<br>EEN | TOP8<br>EEN | TOP7<br>EEN | TOP6<br>EEN | TOP5<br>EEN | TOP4<br>EEN | TOP3<br>EEN | TOP2<br>EEN | TOP1<br>EEN | TOP0<br>EEN |

<When reset: H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-4 | No functions assigned | | 0 | – |
| 5 | TOP10EEN (TOP10 external enable permit) | 0: Disables external enable | ○ | ○ |
| 6 | TOP9EEN (TOP9 external enable permit) | 1: Enables external enable | | |
| 7 | TOP8EEN (TOP8 external enable permit) | | | |
| 8 | TOP7EEN (TOP7 external enable permit) | | | |
| 9 | TOP6EEN (TOP6 external enable permit) | | | |
| 10 | TOP5EEN (TOP5 external enable permit) | | | |
| 11 | TOP4EEN (TOP4 external enable permit) | | | |
| 12 | TOP3EEN (TOP3 external enable permit) | | | |
| 13 | TOP2EEN (TOP2 external enable permit) | | | |
| 14 | TOP1EEN (TOP1 external enable permit) | | | |
| 15 | TOP0EEN (TOP0 external enable permit) | | | |

Note: • This register must always be accessed in halfwords.

The TOP0-10 External Enable Permit Register controls enable operation from sources external to the TOP counter by enabling or disabling it.

## ■ TOP0-10 Enable Protect Register (TOPPRO) <Address:H'0080 02FC>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  | TOP10<br>PRO | TOP9<br>PRO | TOP8<br>PRO | TOP7<br>PRO | TOP6<br>PRO | TOP5<br>PRO | TOP4<br>PRO | TOP3<br>PRO | TOP2<br>PRO | TOP1<br>PRO | TOP0<br>PRO |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-4 | No functions assigned |  | 0 | – |
| 5 | TOP10PRO (TOP10 enable protect) | 0: Enables rewrite<br>1: Disables rewrite | ○ | ○ |
| 6 | TOP9PRO (TOP9 enable protect) |  |  |  |
| 7 | TOP8PRO (TOP8 enable protect) |  |  |  |
| 8 | TOP7PRO (TOP7 enable protect) |  |  |  |
| 9 | TOP6PRO (TOP6 enable protect) |  |  |  |
| 10 | TOP5PRO (TOP5 enable protect) |  |  |  |
| 11 | TOP4PRO (TOP4 enable protect) |  |  |  |
| 12 | TOP3PRO (TOP3 enable protect) |  |  |  |
| 13 | TOP2PRO (TOP2 enable protect) |  |  |  |
| 14 | TOP1PRO (TOP1 enable protect) |  |  |  |
| 15 | TOP0PRO (TOP0 enable protect) |  |  |  |

Note: • This register must always be accessed in halfwords.

The TOP0-10 Enable Protect Register controls rewriting of the TOP0-10 count enable bits shown in the next page by enabling or disabling rewrite.

### ■ TOP0-10 Count Enable Register (TOPCEN)

<Address:H'0080 02FE>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | TOP10<br>CEN | TOP9<br>CEN | TOP8<br>CEN | TOP7<br>CEN | TOP6<br>CEN | TOP5<br>CEN | TOP4<br>CEN | TOP3<br>CEN | TOP2<br>CEN | TOP1<br>CEN | TOP0<br>CEN |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-4 | No functions assigned | | 0 | – |
| 5 | TOP10CEN (TOP10 count enable) | 0: Stops count<br>1: Enables count | ○ | ○ |
| 6 | TOP9CEN (TOP9 count enable) | | | |
| 7 | TOP8CEN (TOP8 count enable) | | | |
| 8 | TOP7CEN (TOP7 count enable) | | | |
| 9 | TOP6CEN (TOP6 count enable) | | | |
| 10 | TOP5CEN (TOP5 count enable) | | | |
| 11 | TOP4CEN (TOP4 count enable) | | | |
| 12 | TOP3CEN (TOP3 count enable) | | | |
| 13 | TOP2CEN (TOP2 count enable) | | | |
| 14 | TOP1CEN (TOP1 count enable) | | | |
| 15 | TOP0CEN (TOP0 count enable) | | | |

Note: • This register must always be accessed in halfwords.

The TOP0-10 Count Enable Register controls the operation of TOP counter. To enable the counter in software, enable the relevant TOP0-10 Enable Protect Register for write and set the count enable bit by writing a 1. To stop the counter, enable the TOP0-10 Enable Protect Register for write and reset the count enable bit by writing a 0.

In all but continuous mode, when the counter stops due to an occurrence of underflow, the count enable bit is automatically reset to 0. Therefore, what you get by reading the TOP0-10 Count Enable Register is the status that indicates the counter's operating status (active or idle).

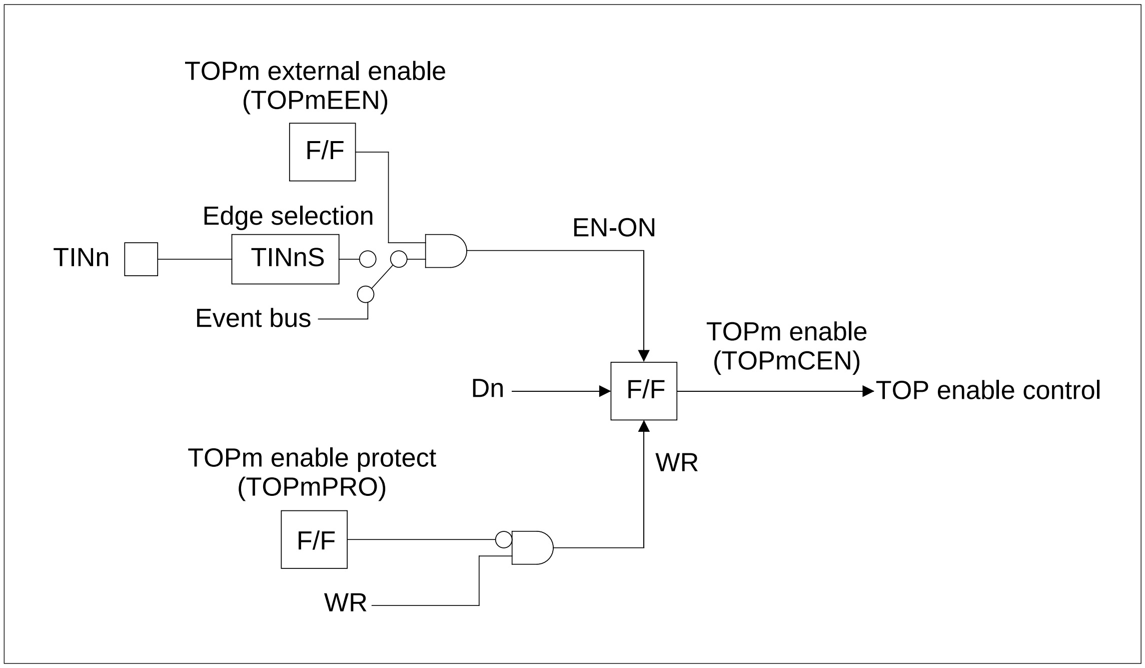


**Figure 10.3.9 Configuration of the TOP Enable Circuit**

### 10.3.9 Operation in TOP Single-shot Output Mode (with Correction Function)

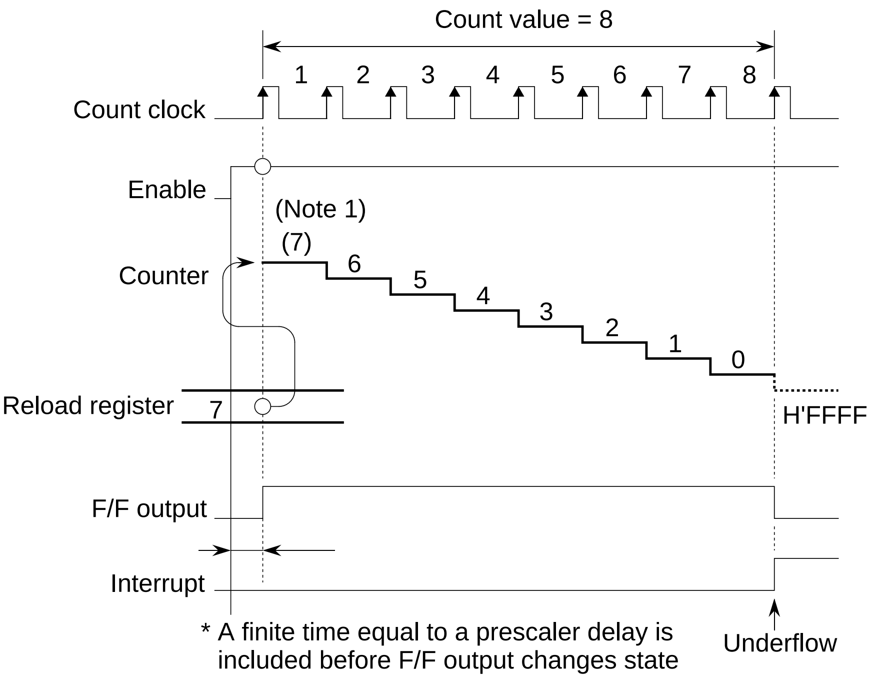
#### (1) Outline of TOP single-shot output mode

In single-shot output mode, the timer generates a pulse in width of (reload register value + 1) only once and stops.

When after setting the reload register, the timer is enabled (by writing to the enable bit in software or by external input), it loads the content of the reload register into the counter synchronously with the count clock, letting the counter start counting. The counter counts down clock pulses and stops when it underflows after reaching the minimum count.

The F/F output waveform in single-shot output mode is inverted (F/F output levels change from low to high, or vice versa) at startup and upon underflow, generating a single-shot pulse waveform in width of (reload register set value + 1) only once. Also, an interrupt can be generated when the counter underflows.

The count value is (reload register set value + 1). In the case shown below, for example, if the reload register value = 7, then the count value = 8.
Because all internal circuits operate synchronously with the count clock, a finite time equal to a prescaler delay is included before F/F output changes state after the timer is enabled.



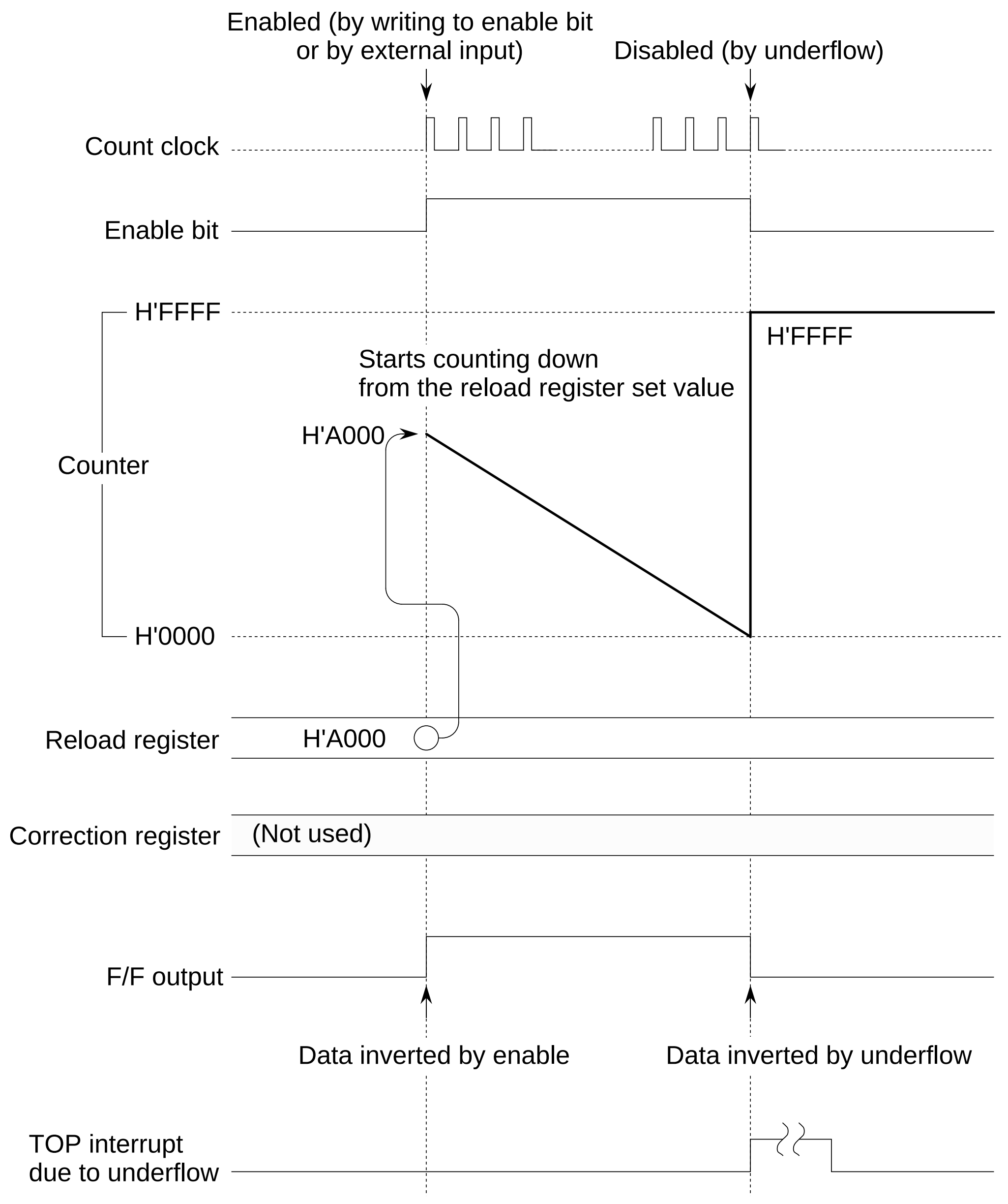
* A finite time equal to a prescaler delay is included before F/F output changes state after the timer is enabled.

Note 1: What you actually see in the cycle immediately after reload is the previous counter value, and not 7.

Note: • This diagram does not show detail timing information.

**Figure 10.3.10 Example of Counting in TOP Single-shot Output Mode**

In the example below, the reload register has the initial value H'A000 set in it. (The initial value of the counter can be indeterminate, and does not have to be specific.) When the timer starts, the reload register value is loaded into the counter causing it to start counting. Thereafter, it continues counting down clock pulses until it underflows after reaching the minimum count.



Note: • This diagram does not show detail timing information.

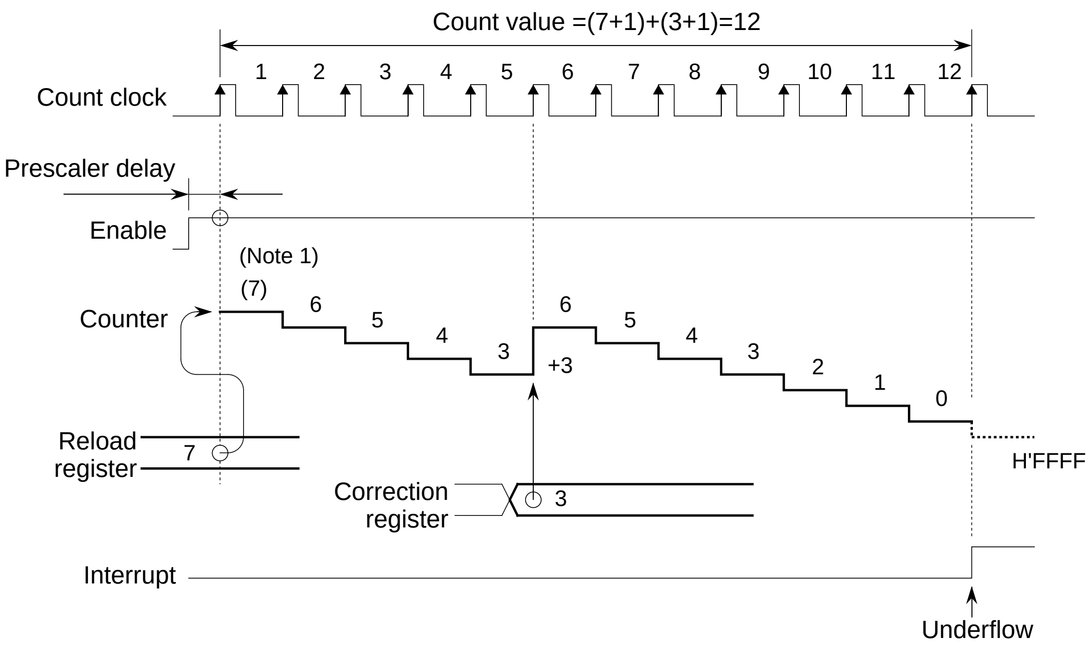
**Figure 10.3.11 Typical Operation in TOP Single-shot Output Mode**

### (2) Correction function of TOP single-shot output mode

If you want to change the counter value during operation, write a value to the TOP correction register, the value by which you want to be increased or reduced from the initial count set in the counter. To add, write the value you want to add to the correction register directly as is; to subtract, write the two's complement of the value you want to subtract to the correction register.

Correction of the counter is performed synchronously with a clock period next to the one in which the correction value was written to the TOP correction register. In this case, one down-count in the clock period during which the correction was performed is canceled. Therefore, note that the counter value actually is corrected by (correction register value + 1).

For example, if the initial counter value is 7 and you write a value 3 to the correction register when the counter has counted down to 3, then the counter underflows after a total of 12 counts.

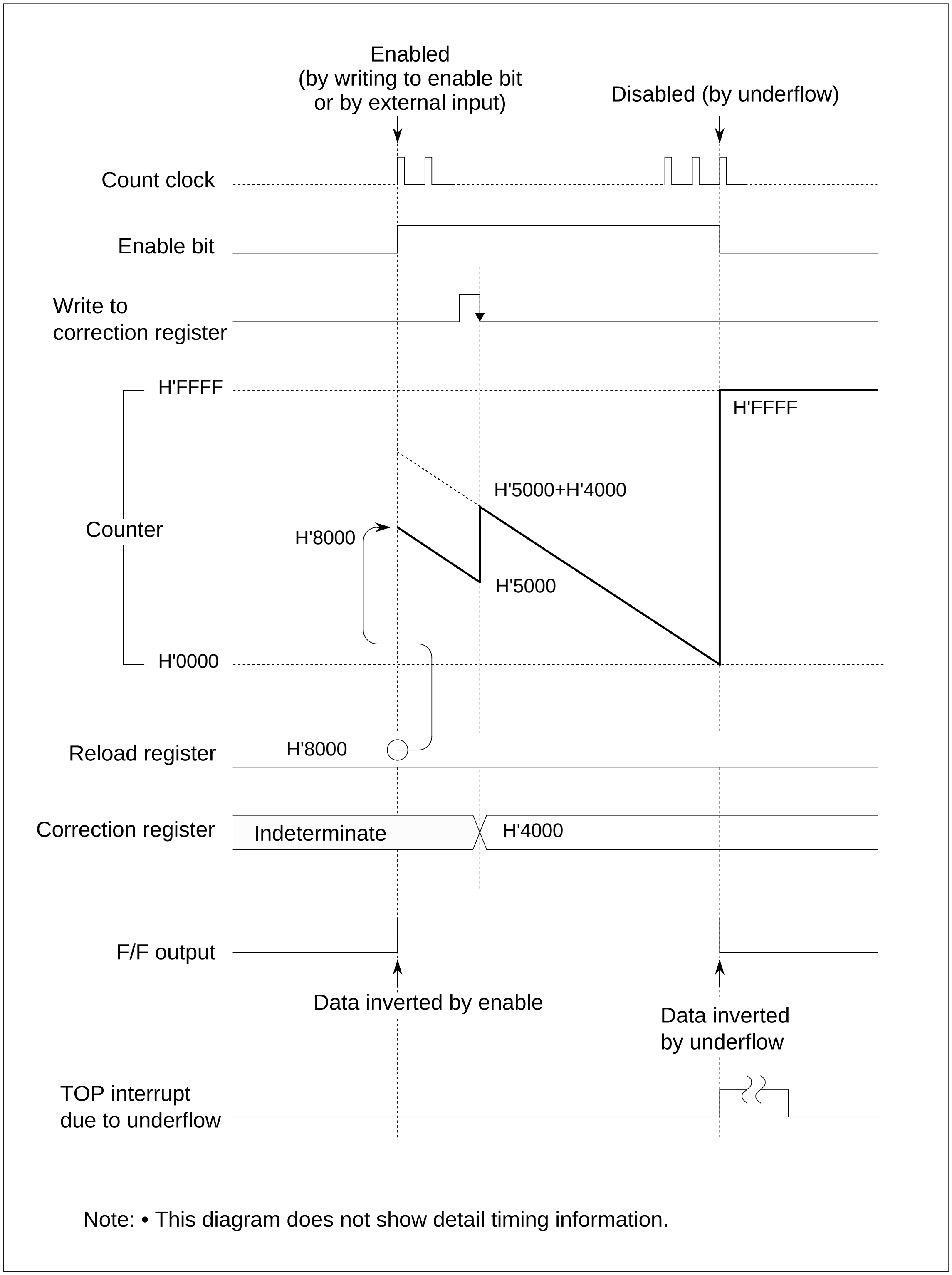


Note 1: What you actually see in the cycle immediately after reload is the previous counter value, and not 7.
Note: • This diagram does not show detail timing information.

Figure 10.3.12 Example of Counting in TOP Single-shot Output Mode When Count is Corrected

When writing to the correction register, be careful not to cause the counter to overflow. Even when the counter overflows due to correction of counts, no interrupt is generated for the occurrence of overflow.

In the example next page, the reload register has the initial value H'8000 set in it. When the timer starts, the reload register value is loaded into the counter causing it to start counting down. In the example diagram here, H'4000 is written to the correction register when the counter has counted down to H'5000. As a result of this correction, the count has been increased to H'9000, so that the counter stops after counting a total of (H'8000 + 1 + H'4000 + 1) counts.



Note: • This diagram does not show detail timing information.

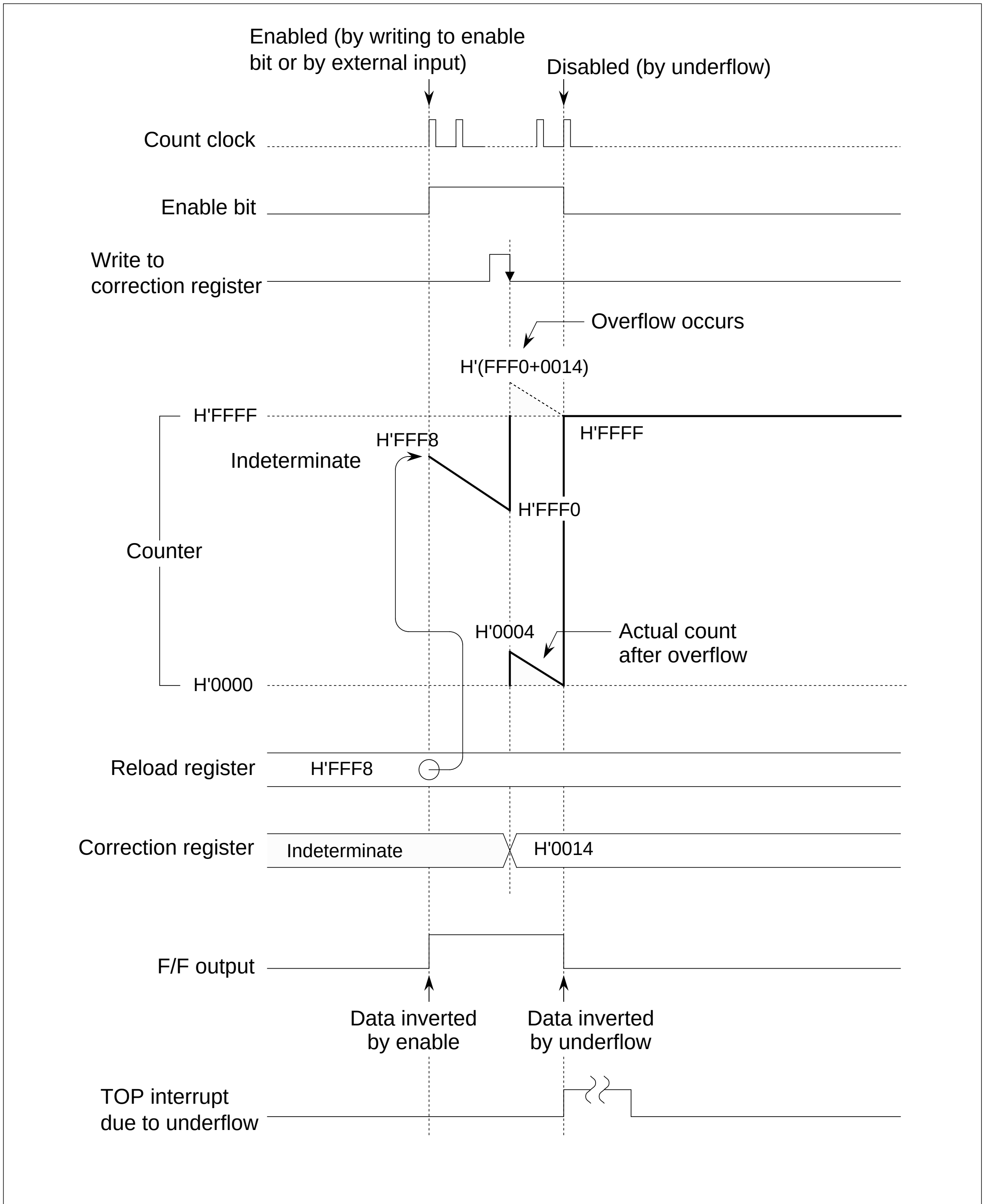
**Figure 10.3.13 Example of Counting in TOP Single-shot Output Mode When Count is Corrected**

### (3) Precautions to be observed when using TOP single-shot output mode

The following describes precautions to be observed when using TOP single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).
- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).
- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.
- When writing to the correction register, be careful not to cause the counter to overflow. Even when the counter overflows due to correction of counts, no interrupt is generated for the occurrence of overflow. When the counter underflows in the subsequent down-count after overflow, a false underflow interrupt is generated due to overcounting.

In the example below, the reload register has the initial value H'FFF8 set in it. When the timer starts, the reload register value is loaded into the counter causing it to start counting down. In the example diagram here, H'0014 is written to the correction register when the counter has counted down to H'FFF0. As a result of this correction, the count overflows to H'0004 and fails to count correctly. Also, an interrupt is generated for an erroneous overcount.



Note: • This diagram does not show detail timing information.

**Figure 10.3.14 Example of Operation in TOP Single-shot Output Mode Where Count Overflows due to Correction**

## 10.3.10 Operation in TOP Delayed Single-shot Output Mode (With Correction Function)

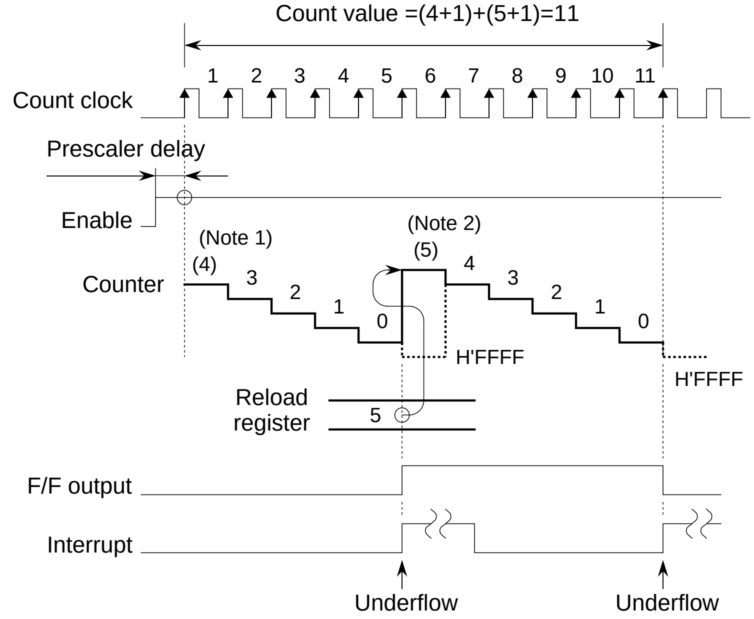
### (1) Outline of TOP delayed single-shot output mode

In delayed single-shot output mode, the timer generates a pulse in width of (reload register set value + 1) only once, with the output delayed by an amount of time equal to (counter set value + 1) and then stops.

When after setting the counter and reload register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock. The first time the counter underflows, the reload register value is loaded into the counter causing it to continue counting down, and the counter stops when it underflows next time.

The F/F output waveform in delayed single-shot output mode is inverted (F/F output levels change from low to high, or vice versa) when the counter underflows first time and next, generating a single-shot pulse waveform in width of (reload register set value + 1) only once, with the output delayed by an amount of time equal to (first set value of counter + 1). Also, an interrupt can be generated when the counter underflows first time and next.

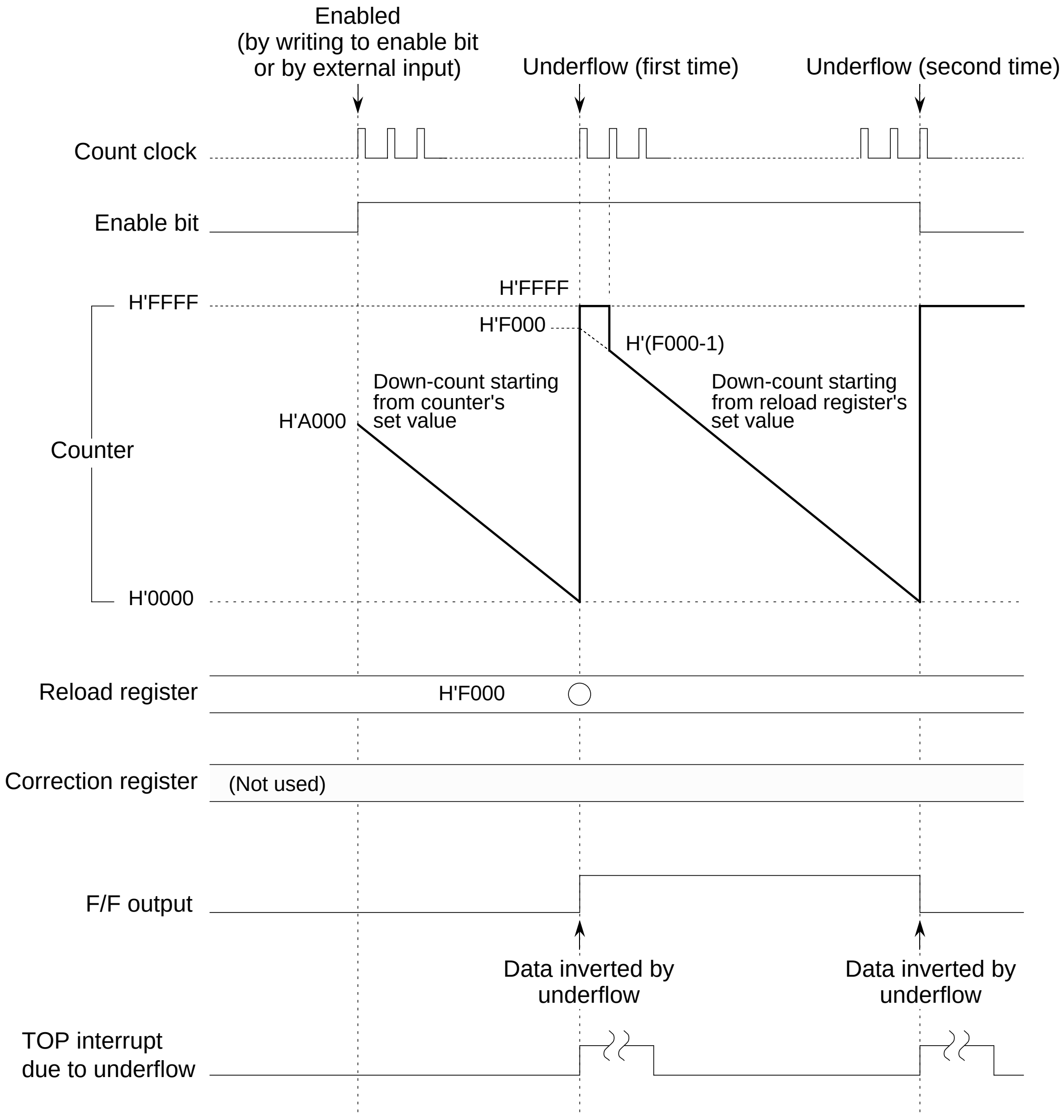
The valid count values are the (counter set value + 1) and (reload register set value + 1). The diagram below shows timer operation as an example when the initial counter value = 4 and the initial reload register value = 5.



Note 1: What you actually see in the cycle immediately after enable is the previous counter value, and not 4.
Note 2: What you actually see in the cycle immediately after reload is H'FFFF (underflow value), and not 5.
Note: • This diagram does not show detail timing information.

**Figure 10.3.15 Example of Counting in TOP Delayed Single-shot Output Mode**

In the example below, the counter has the initial value H'A000 set in it and the reload register has the initial value H'F000 set in it. When the timer starts, the counter starts counting down clock pulses and when it underflows after reaching the minimum count, the counter is reloaded with the content of the reload register. Then when the counter underflows next time while continuing down-count, it stops.



Note: • This diagram does not show detail timing information.

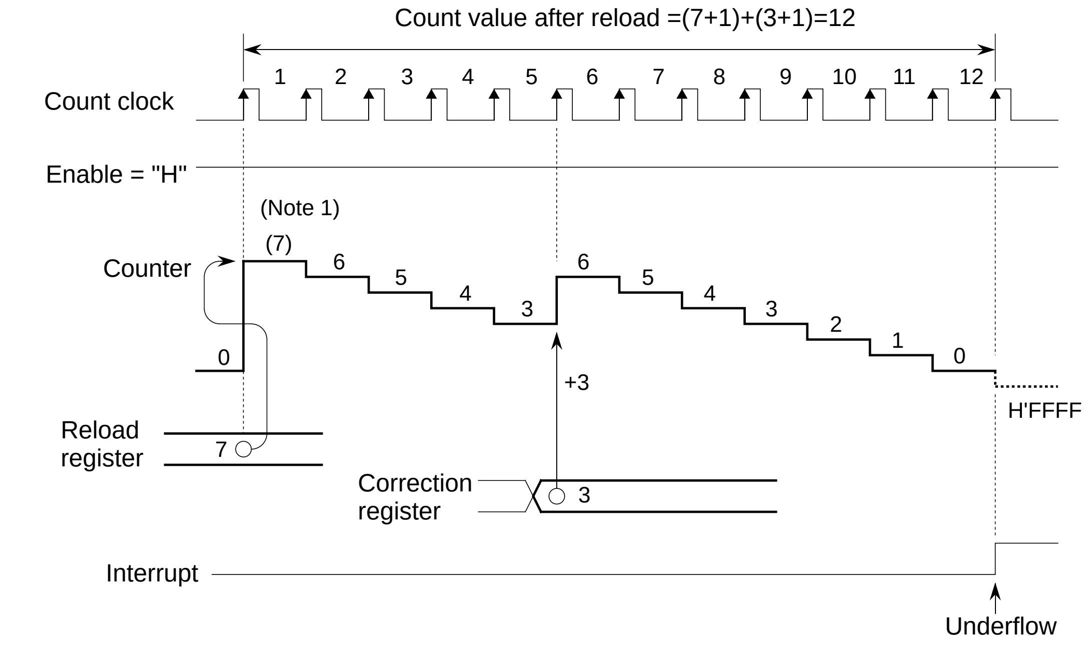
**Figure 10.3.16 Typical Operation in TOP Delayed Single-shot Output**

### (2) Correction function of TOP delayed single-shot output mode

If you want to change the counter value during operation, write a value to the TOP correction register, the value by which you want to be increased or reduced from the initial count set in the counter. To add, write the value you want to add to the correction register directly as is; to subtract, write the two's complement of the value you want to subtract to the correction register.

Correction of the counter is performed synchronously with a clock period next to the one in which the correction value was written to the TOP correction register. In this case, one down-count in the clock period during which the correction was performed is canceled. Therefore, note that the counter value actually is corrected by (correction register value + 1).

For example, if the initial counter value is 7 and you write a value 3 to the correction register when the counter has counted down to 3, then the counter underflows after a total of 12 counts after reload.

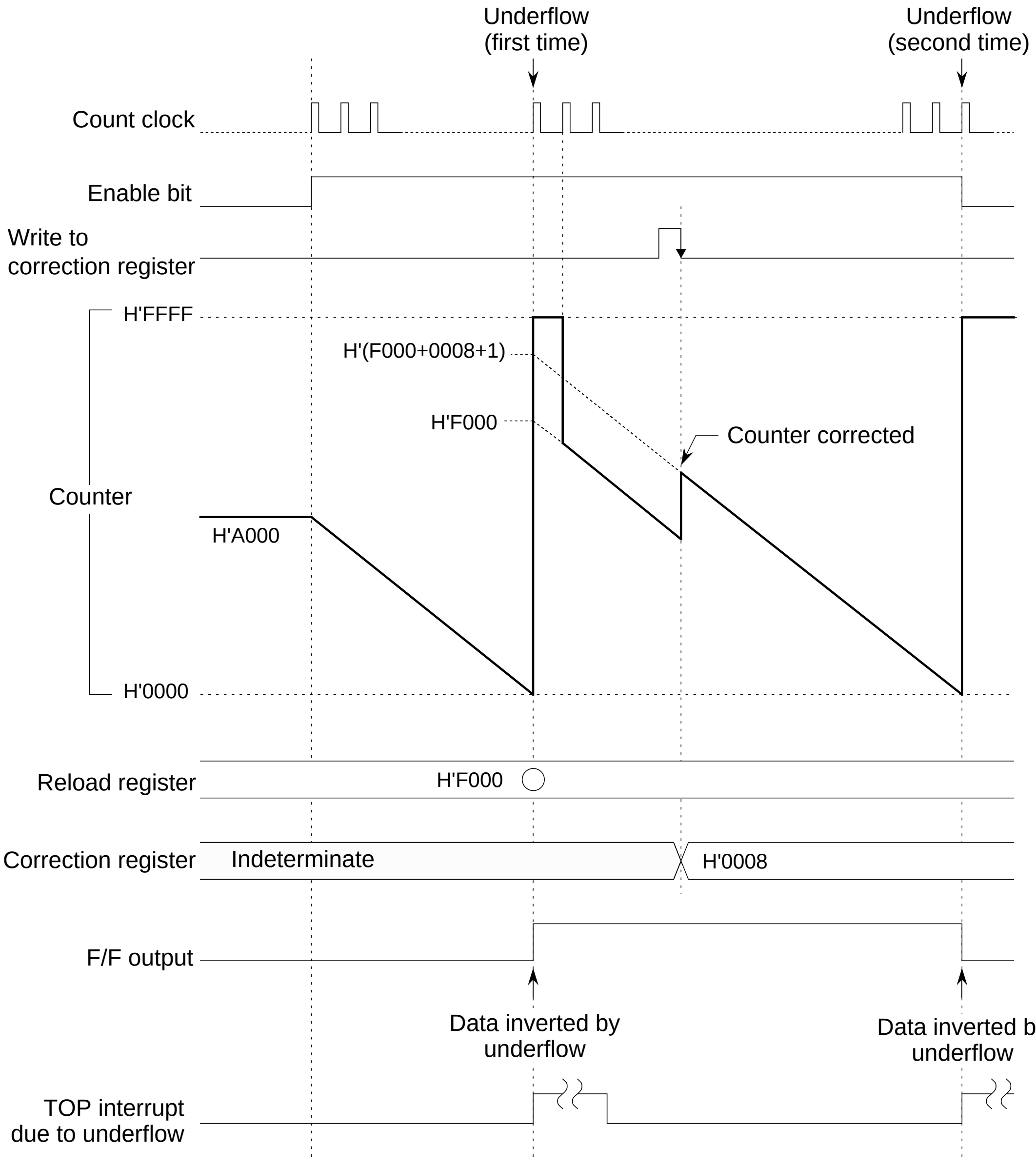


Note 1: What you actually see in the cycle immediately after reload is the previous counter value, and not 7.
Note: • This diagram does not show detail timing information.

**Figure 10.3.17 Example of Counting in TOP Delayed Single-shot Output Mode When Count is Corrected**

When writing to the correction register, be careful not to cause the counter to overflow. Even when the counter overflows due to correction of counts, no interrupt is generated for the occurrence of overflow.

In the example below, the counter and the reload register are initially set to H'A000 and H'F000, respectively. When the timer is enabled, the counter starts counting down and when it underflows first time after reaching the minimum count, the counter is loaded with the content of the reload register and continues counting down. In the diagram below, the value H'0008 is written to the correction register when the counter has counted down to H'9000. As a result of this correction, the counter has its count value increased to H'9008 and counts (H'F000 + 1 + H'0008 +1) after the first underflow before it stops.



Note: • This diagram does not show detail timing information.

**Figure 10.3.18 Typical Operation in TOP Delayed Single-shot Output Mode when Correction Applied**

### (3) Precautions to be observed when using TOP delayed single-shot output mode

The following describes precautions to be observed when using TOP delayed single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).
- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).
- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- Even when the counter overflows due to correction of counts, no interrupt is generated for the occurrence of overflow. When the counter underflows in the subsequent down-count after overflow, a false underflow interrupt is generated due to overcounting.
- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.

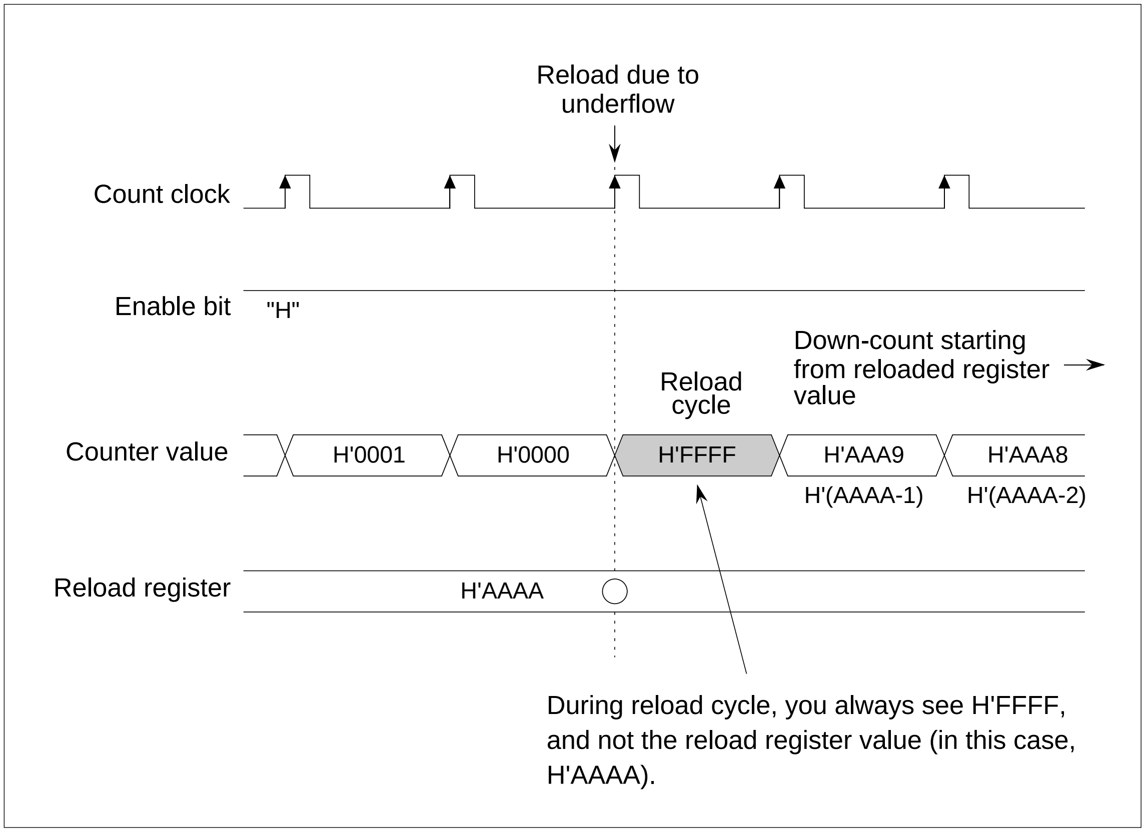

Reload due to underflow
Count clock
Enable bit
"H"
Down-count starting from reloaded register value
Reload cycle
Counter value
H'0001
H'0000
H'FFFF
H'AAA9
H'AAA8
H'(AAAA-1)
H'(AAAA-2)
Reload register
H'AAAA
During reload cycle, you always see H'FFFF, and not the reload register value (in this case, H'AAAA).

**Figure 10.3.19 Counter Value Immediately after Underflow**

### 10.3.11 Operation in TOP Continuous Output Mode (Without Correction Function)

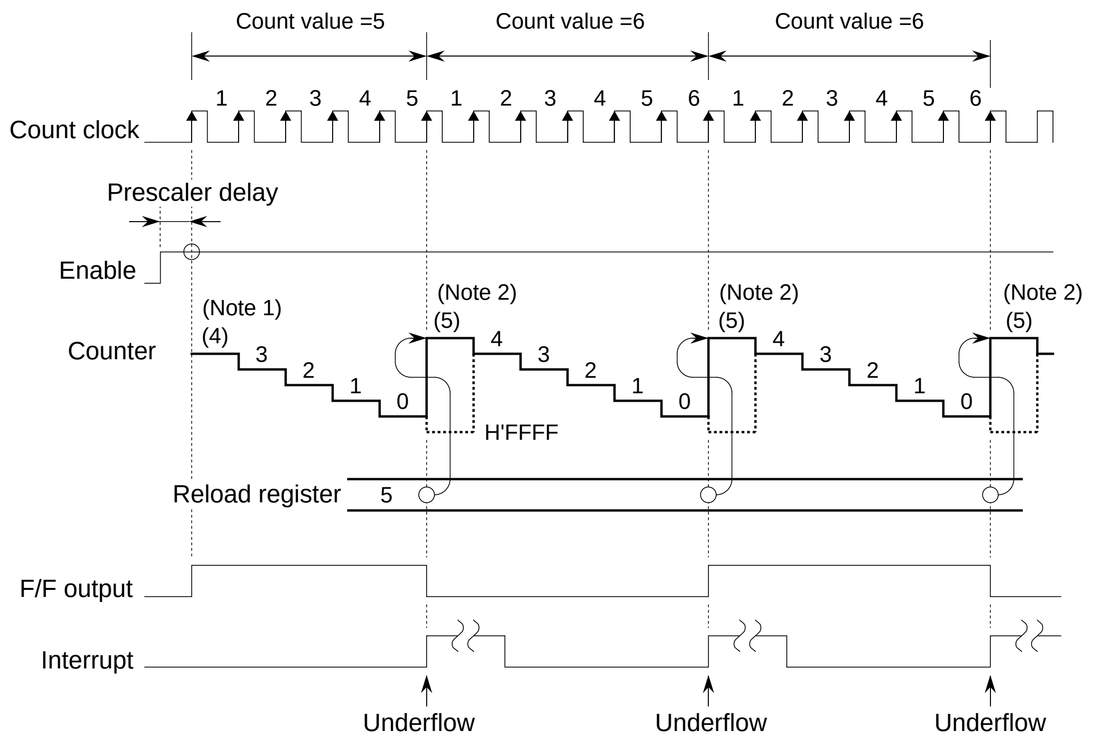
#### (1) Outline of TOP continuous output mode

In continuous output mode, the timer counts down clock pulses starting from the set value of the counter and when the counter underflows, reloads it with the reload register value. Thereafter, this operation is repeated each time the counter underflows, thus generating consecutive pulses whose waveform is inverted in width of (reload register set value + 1).

When after setting the counter and reload register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock and when the minimum count is reached, generates an underflow. This underflow causes the counter to be reloaded with the content of the reload register and start counting over again. Thereafter, this operation is repeated each time an underflow occurs. To stop the counter, disable count by writing to the enable bit in software.

The F/F output waveform in continuous output mode is inverted (F/F output levels change from low to high, or vice versa) at startup and upon underflow, generating consecutive pulses until the timer stops counting. Also, an interrupt can be generated each time the counter underflows.

The valid count values are the (counter set value + 1) and (reload register set value + 1). The diagram below shows timer operation as an example when the initial counter value = 4 and the initial reload register value = 5.



Note 1: What you actually see in the cycle immediately after enable is the previous counter value, and not 4.
Note 2: What you actually see in the cycle immediately after reload is H'FFFF (underflow value), and not 5.
Note: • This diagram does not show detail timing information.

**Figure 10.3.20 Example of Counting in TOP Continuous Output Mode**

In the example below, the counter has the initial value H'A000 set in it and the reload register has the initial value H'E000 set in it. When the timer starts, the counter starts counting down clock pulses and when it underflows after reaching the minimum count, the counter is reloaded with the content of the reload register and continues counting down.

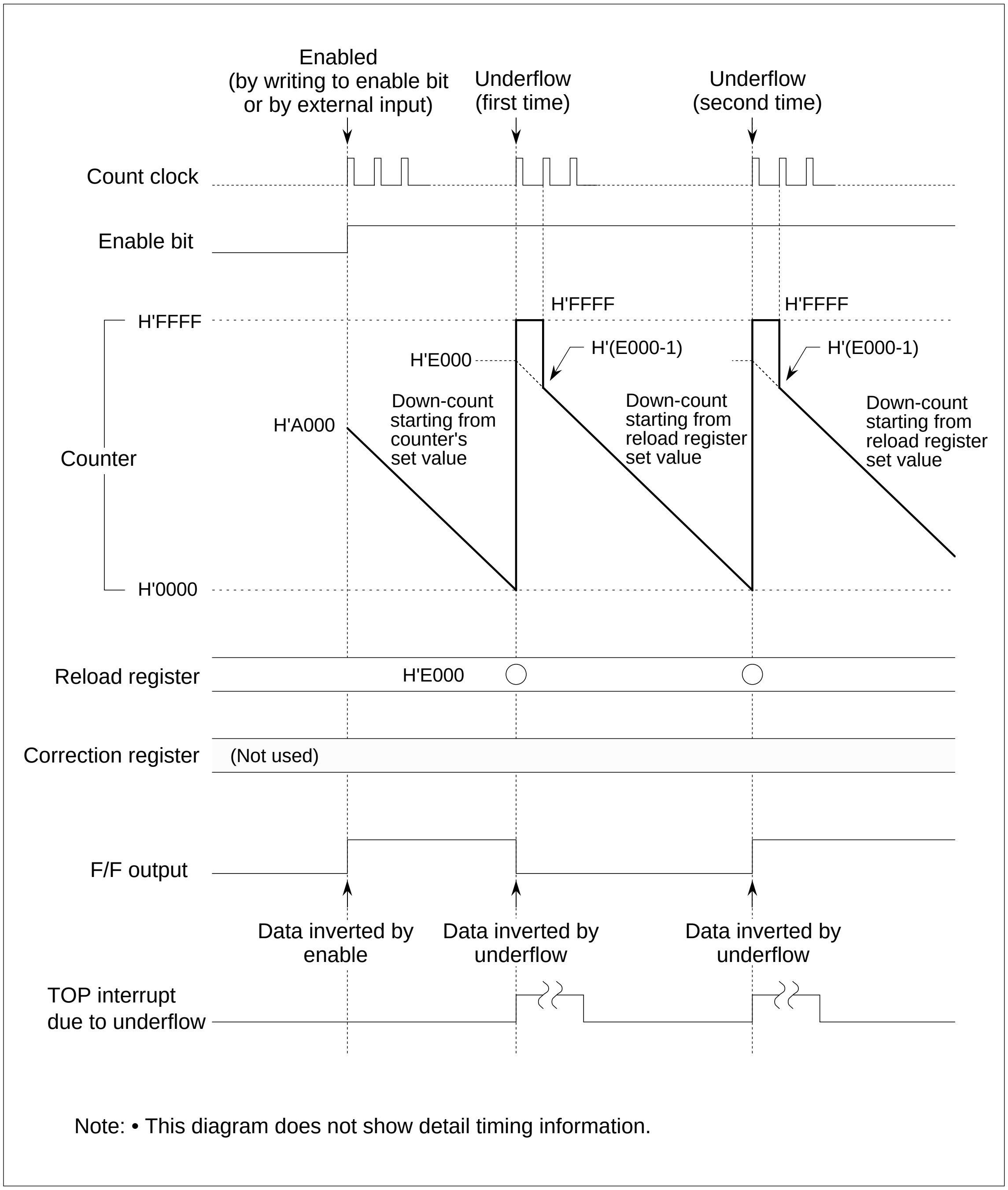

Enabled
(by writing to enable bit or by external input)
Underflow (first time)
Underflow (second time)
Count clock
Enable bit
H'FFFF
H'FFFF
H'FFFF
H'(E000-1)
H'(E000-1)
H'E000
H'A000
Down-count starting from counter's set value
Down-count starting from reload register set value
Down-count starting from reload register set value
Counter
H'0000
Reload register
H'E000
Correction register
(Not used)
F/F output
Data inverted by enable
Data inverted by underflow
Data inverted by underflow
TOP interrupt due to underflow
Note: • This diagram does not show detail timing information.

**Figure 10.3.21 Typical Operation in TOP Continuous Output Mode**

### (2) Precautions to be observed when using TOP continuous output mode

The following describes precautions to be observed when using TOP continuous output mode.

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.
- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.

## 10.4 TIO (Input/Output-related 16-bit Timer)

### 10.4.1 Outline of TIO

TIO (Timer Input/Output) is an input/output-related 16-bit timer, whose operation mode can be selected from the following by mode switching in software:

<Input mode>

- Measure clear input mode
- Measure free-run input mode
- Noise processing input mode

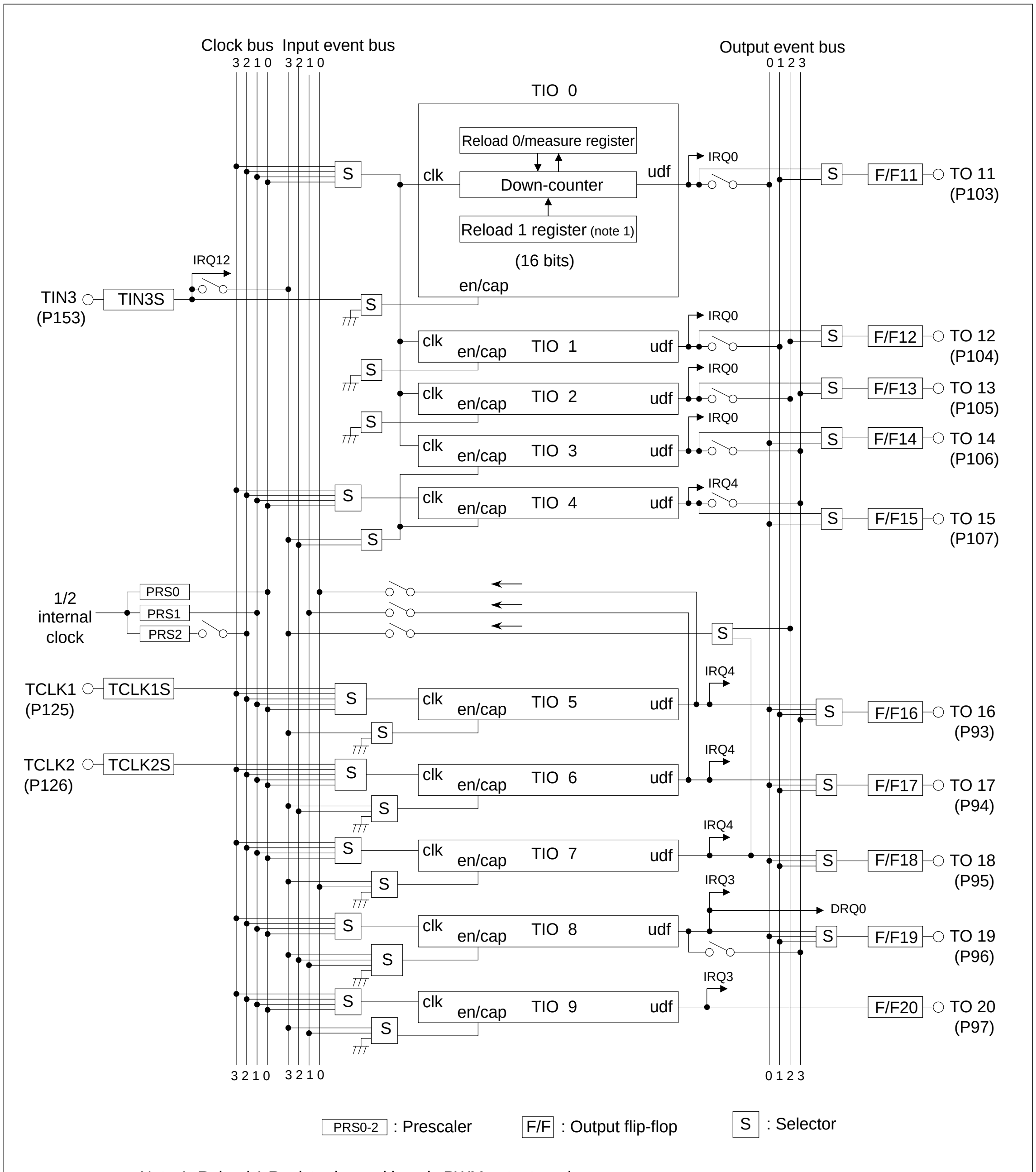
<Output mode without correction function>

- PWM output mode
- Single-shot output mode
- Delayed single-shot output mode
- Continuous output mode

The following shows TIO specifications. Figure 10.4.1 shows a TIO block diagram.

**Table 10.4.1 Specifications of TIO (Input/Output-related 16-bit Timer)**

| Item | Specification |
|---|---|
| Number of channels | 10 channels |
| Counter | 16-bit down-counter |
| Reload register | 16-bit reload register |
| Measure register | 16-bit capture register |
| Timer startup | Started by writing to enable bit in software or by enabling with external input (rising/falling edge or both or high/low level) |
| Mode selection | <Input mode><br>• Measure clear input mode<br>• Measure free-run input mode<br>• Noise processing input mode<br><Output mode without correction function><br>• PWM output mode<br>• Single-shot output mode<br>• Delayed single-shot output mode<br>• Continuous output mode |
| Interrupt generation | Can be generated by a counter underflow |
| DMA transfer request generation | Can be generated by a counter underflow (for only the TIO8) |



Note 1: Reload 1 Register is used in only PWM output mode.

**Figure 10.4.1 Block Diagram of TIO (Input/Output-related 16-bit Timer)**

### 10.4.2 Outline of Each Mode of TIO

Each mode of TIO is outlined below. For each TIO channel, only one of the following modes can be selected.

#### (1) Measure clear/free-run input modes

In measure clear/free-run input modes, the timer measures a duration of time from when it starts counting till when an external capture signal is entered.

After the timer is enabled (by writing to the enable bit in software), the counter starts counting down synchronously with the count clock. When a capture signal is entered from an external device, the counter value at that point in time is written to a register called the "measure register."

Especially in measure clear input mode, the counter value is initialized to H'FFFF upon capture, from which the counter starts counting down again. In measure free-run mode, the counter continues counting down even after capture and upon underflow, recycles to H'FFFF, from which it starts counting down again.

To stop the counter, disable count by writing to the enable bit in software. An interrupt can be generated by a counter underflow or execution of measure operation. Also, a DMA transfer request (for only the TIO8) can be generated when the counter underflows.

#### (2) Noise processing input mode

In noise processing input mode, the timer detects the status of an input signal that it remained in the same state for over a predetermined time.

In noise processing input mode, the counter is started by entering a high or low-level signal from an external device and if the signal remains in the same state for over a predetermined time before the counter underflows, the counter stops after generating an interrupt. If the valid-level signal being applied turns to an invalid level before the counter underflows, the counter temporarily stops counting and when a valid-level signal is entered again, it is reloaded with the initial count and restarts counting.

The timer stops at the same time the counter underflows or count is disabled by writing to the enable bit. An interrupt as well as a DMA transfer request (for only the TIO8) can be generated by a counter underflow.

#### (3) PWM output mode (without correction function)

In PWM output mode, the timer uses two reload registers to generate a waveform with a given duty cycle.

When after setting the initial values in reload 0 and reload 1 registers, the timer is enabled (by writing to the enable bit in software or by external input), it loads the reload 0 register value into the counter synchronously with the count clock letting the counter start counting down. The first time the counter underflows, the reload 1 register value is loaded into the counter letting it continue counting. Thereafter, the counter is reloaded with the reload 0 and reload 1 register values alternately each time an underflow occurs.

The F/F output waveform in PWM output mode is inverted at count startup and upon each underflow. The timer stops at the same time count is disabled by writing to the enable bit (and not in synchronism with PWM output period). An interrupt can be generated when the counter underflows every even time (second time, fourth time, and so on) after being enabled. Also, a DMA ttransfer request (for only the TIO8) can be generated every time the counter underflows.

### (4) Single-shot output mode (without correction function)

In single-shot output mode, the timer generates a pulse in width of (reload 0 register set value + 1) only once and stops.

When after setting the reload 0 register, the timer is enabled (by writing to the enable bit in software or by external input), it loads the content of reload 0 register into the counter synchronously with the count clock, letting the counter start counting. The counter counts down clock pulses and stops when it underflows after reaching the minimum count.

The F/F output waveform in single-shot output mode is inverted at startup and upon underflow, generating a single-shot pulse waveform in width of (reload 0 register set value + 1) only once. Also, an interrupt as well as a DMA transfer request (for only the TIO8) can be generated when the counter underflows.

### (5) Delayed single-shot output mode (without correction function)

In delayed single-shot output mode, the timer generates a pulse in width of (reload 0 register set value + 1) only once, with the output delayed by an amount of time equal to (counter set value + 1) and then stops.

When after setting the counter and reload 0 register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock. The first time the counter underflows, the reload 0 register value is loaded into the counter causing it to continue counting down, and the counter stops when it underflows next time.

The F/F output waveform in delayed single-shot output mode is inverted when the counter underflows first time and next, generating a single-shot pulse waveform in width of (reload 0 register set value + 1) only once, with the output delayed by an amount of time equal to (first set value of counter + 1). Also, an interrupt and a DMA transfer request (for only the TIO8) can be generated when the counter underflows first time and next.

### (6) Continuous output mode (without correction function)

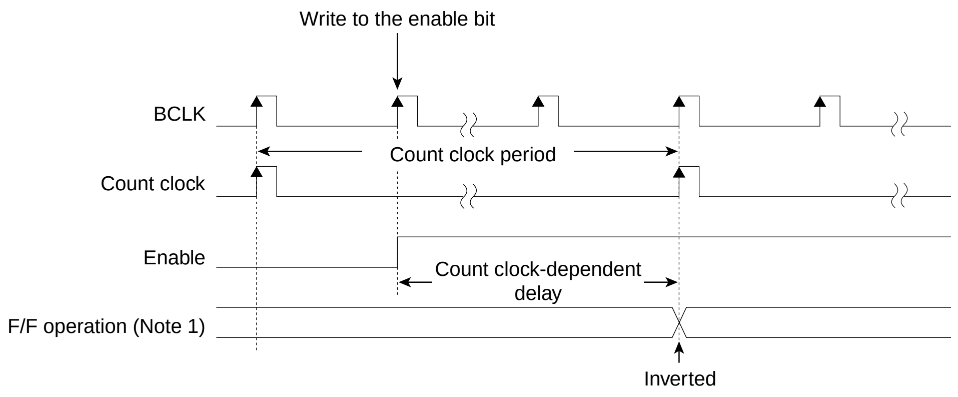
In continuous output mode, the timer counts down clock pulses starting from the set value of the counter and when the counter underflows, reloads it with the reload 0 register value. Thereafter, this operation is repeated each time the counter underflows, thus generating consecutive pulses in width of (reload 0 register set value + 1).

When after setting the counter and reload 0 register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock and when the minimum count is reached, generates an underflow. This underflow causes the counter to be reloaded with the content of reload 0 register and start counting over again. Thereafter, this operation is repeated each time an underflow occurs. To stop the counter, disable count by writing to the enable bit in software.

The F/F output waveform in continuous output mode is inverted at startup and upon underflow, generating consecutive pulses until the timer stops counting. Also, an interrupt as well as a DMA transfer request (for only the TIO8) can be generated each time the counter underflows.

### <Count clock-dependent delay>

- Because the timer operates synchronously with the count clock, there is a count clock-dependent delay from when the timer is enabled till it actually starts operating. In operation mode where the F/F output is inverted when the timer is enabled, the F/F output is inverted synchronously with the count clock.

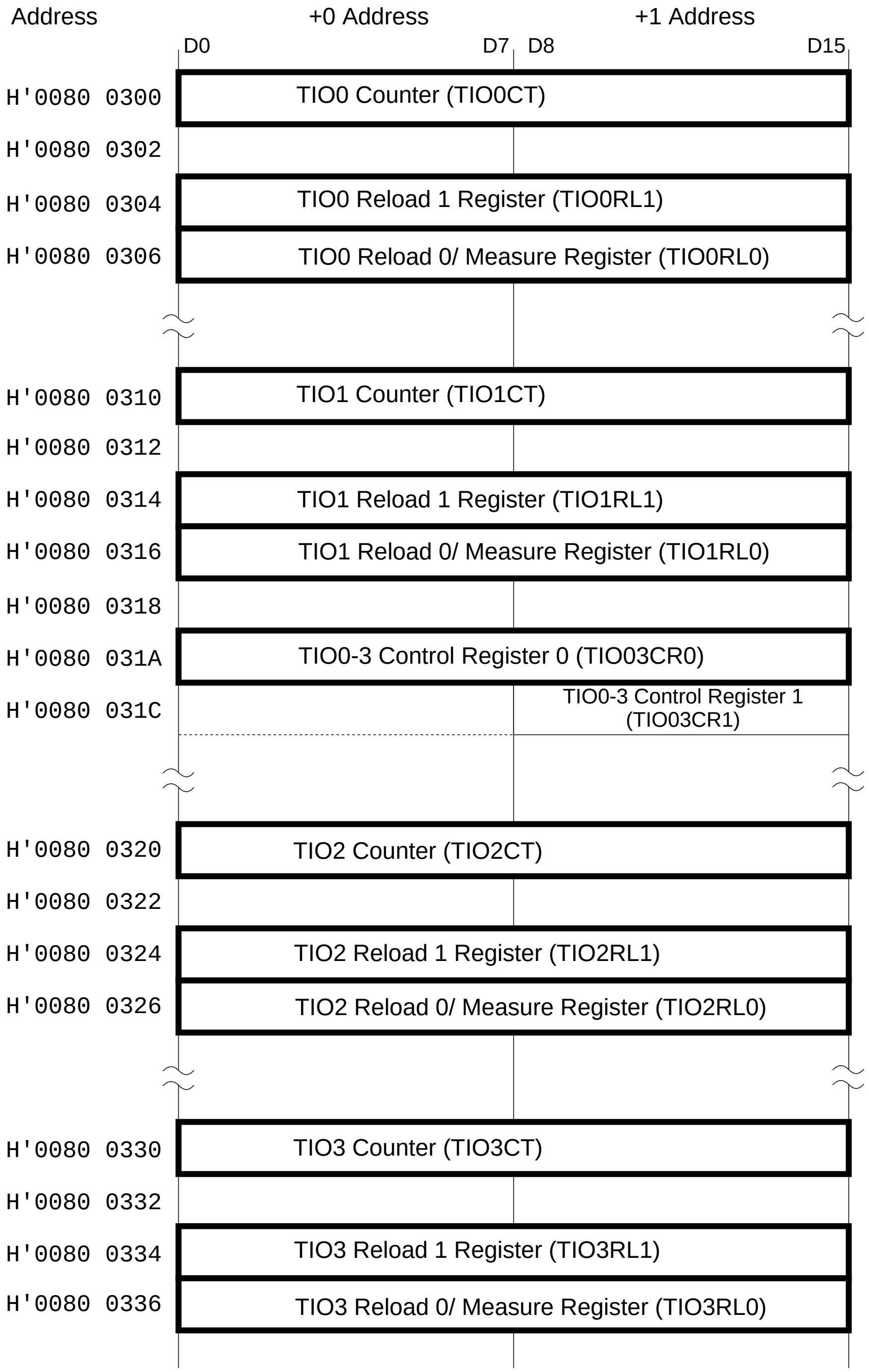


Note 1: This applies to the case where F/F output is inverted when the timer is enabled.

**Figure 10.4.2 Count Clock Dependent Delay**

### 10.4.3 TIO Related Register Map

The diagram below shows a TIO related register map.



| Address | +0 Address<br>D0 … D7 | +1 Address<br>D8 … D15 |
|---|---|---|
| H'0080 0300 | TIO0 Counter (TIO0CT) | |
| H'0080 0302 | | |
| H'0080 0304 | TIO0 Reload 1 Register (TIO0RL1) | |
| H'0080 0306 | TIO0 Reload 0/ Measure Register (TIO0RL0) | |
| H'0080 0310 | TIO1 Counter (TIO1CT) | |
| H'0080 0312 | | |
| H'0080 0314 | TIO1 Reload 1 Register (TIO1RL1) | |
| H'0080 0316 | TIO1 Reload 0/ Measure Register (TIO1RL0) | |
| H'0080 0318 | | |
| H'0080 031A | TIO0-3 Control Register 0 (TIO03CR0) | |
| H'0080 031C | | TIO0-3 Control Register 1 (TIO03CR1) |
| H'0080 0320 | TIO2 Counter (TIO2CT) | |
| H'0080 0322 | | |
| H'0080 0324 | TIO2 Reload 1 Register (TIO2RL1) | |
| H'0080 0326 | TIO2 Reload 0/ Measure Register (TIO2RL0) | |
| H'0080 0330 | TIO3 Counter (TIO3CT) | |
| H'0080 0332 | | |
| H'0080 0334 | TIO3 Reload 1 Register (TIO3RL1) | |
| H'0080 0336 | TIO3 Reload 0/ Measure Register (TIO3RL0) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 10.4.3 TIO Related Register Map (1/3)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 0340 | TIO4 Counter (TIO4CT) | |
| H'0080 0342 | | |
| H'0080 0344 | TIO4 Reload 1 Register (TIO4RL1) | |
| H'0080 0346 | TIO4 Reload 0/ Measure Register (TIO4RL0) | |
| H'0080 0348 | | |
| H'0080 034A | TIO4 Control Register (TIO4CR) | TIO5 Control Register (TIO5CR) |
| H'0080 0350 | TIO5 Counter (TIO5CT) | |
| H'0080 0352 | | |
| H'0080 0354 | TIO5 Reload 1 Register (TIO5RL1) | |
| H'0080 0356 | TIO5 Reload 0/ Measure Register (TIO5RL0) | |
| H'0080 0360 | TIO6 Counter (TIO6CT) | |
| H'0080 0362 | | |
| H'0080 0364 | TIO6 Reload 1 Register (TIO6RL1) | |
| H'0080 0366 | TIO6 Reload 0/ Measure Register (TIO6RL0) | |
| H'0080 0368 | | |
| H'0080 036A | TIO6 Control Register (TIO6CR) | TIO7 Control Register (TIO7CR) |
| H'0080 0370 | TIO7 Counter (TIO7CT) | |
| H'0080 0372 | | |
| H'0080 0374 | TIO7 Reload 1 Register (TIO7RL1) | |
| H'0080 0376 | TIO7 Reload 0/ Measure Register (TIO7RL0) | |

Blank addresses are reserved.

**Figure 10.4.4 TIO Related Register Map (2/3)**

<table>
<tr><th>Address</th><th>+0 Address<br>D0 D7</th><th>+1 Address<br>D8 D15</th></tr>
<tr><td>H'0080 0380</td><td colspan="2">TIO8 Counter (TIO8CT)</td></tr>
<tr><td>H'0080 0382</td><td></td><td></td></tr>
<tr><td>H'0080 0384</td><td colspan="2">TIO8 Reload 1 Register (TIO8RL1)</td></tr>
<tr><td>H'0080 0386</td><td colspan="2">TIO8 Reload 0/ Measure Register (TIO8RL0)</td></tr>
<tr><td>H'0080 0388</td><td></td><td></td></tr>
<tr><td>H'0080 038A</td><td>TIO8 Control Register<br>(TIO8CR)</td><td>TIO9 Control Register<br>(TIO9CR)</td></tr>
<tr><td>H'0080 0390</td><td colspan="2">TIO9 Counter (TIO9CT)</td></tr>
<tr><td>H'0080 0392</td><td></td><td></td></tr>
<tr><td>H'0080 0394</td><td colspan="2">TIO9 Reload 1 Register (TIO9RL1)</td></tr>
<tr><td>H'0080 0396</td><td colspan="2">TIO9 Reload 0/ Measure Register (TIO9RL0)</td></tr>
<tr><td>H'0080 03BC</td><td colspan="2">TIO0-9 Enable Protect Register (TIOPRO)</td></tr>
<tr><td>H'0080 03BE</td><td colspan="2">TIO0-9 Count Enable Register (TIOCEN)</td></tr>
</table>

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 10.4.5 TIO Related Register Map (3/3)**

### 10.4.4 TIO Control Registers

The TIO control registers are used to select TIO0-9 operation modes (measure input, noise processing input, PWM output, single-shot output, delayed single-shot output, or continuous output mode), as well as select the counter enable and counter clock sources. Following eight TIO control registers are provided for each timer group.

- TIO0-3 Control Register 0 (TIO03CR0)
- TIO0-3 Control Register 1 (TIO03CR1)
- TIO4 Control Register (TIO4CR)

- TIO5 Control Register (TIO5CR)
- TIO6 Control Register (TIO6CR)
- TIO7 Control Register (TIO7CR)
- TIO8 Control Register (TIO8CR)
- TIO9 Control Register (TIO9CR)

### ■ TIO0-3 Control Register 0 (TIO3CR0)

<Address:H'0080 031A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TIO3<br>EEN | TIO3M | | | TIO2<br>ENS | TIO2M | | | TIO1<br>ENS | TIO1M | | | TIO0<br>ENS | TIO0M | | |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | TIO3EEN (TIO3 external input enable)<br>(Note 1) | 0: Disables external input<br>1: Enables external input | ○ | ○ |
| 1-3 | TIO3M (TIO3 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |
| 4 | TIO2ENS (reserved) | Setting this bit has no effect | ○ | ○ |
| 5-7 | TIO2M<br>(TIO2 operation mode selection)<br>(Note 2) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Use inhibited | ○ | ○ |
| 8 | TIO1ENS (reserved) | Setting this bit has no effect | ○ | ○ |

(Continues to the next page)

Note 1: To select TIO3 enable/measurement input source, use the TIO4 Control Register TIO34ENS (TIO3,4 enable/measurement input source select) bits.

Note 2: Even when this bit is 0 (external input disabled) during measurement (free-run/clear) input mode, if a capture signal is entered from an external device, the counter value at that point in time is written to the measurement register. However, because if this bit is 0 (external input disabled) during measurement clear input mode, the counter value may not be initialized (H'FFFF) upon capturing, make sure this bit = 1 (external input enabled) before using the measurement clear function.

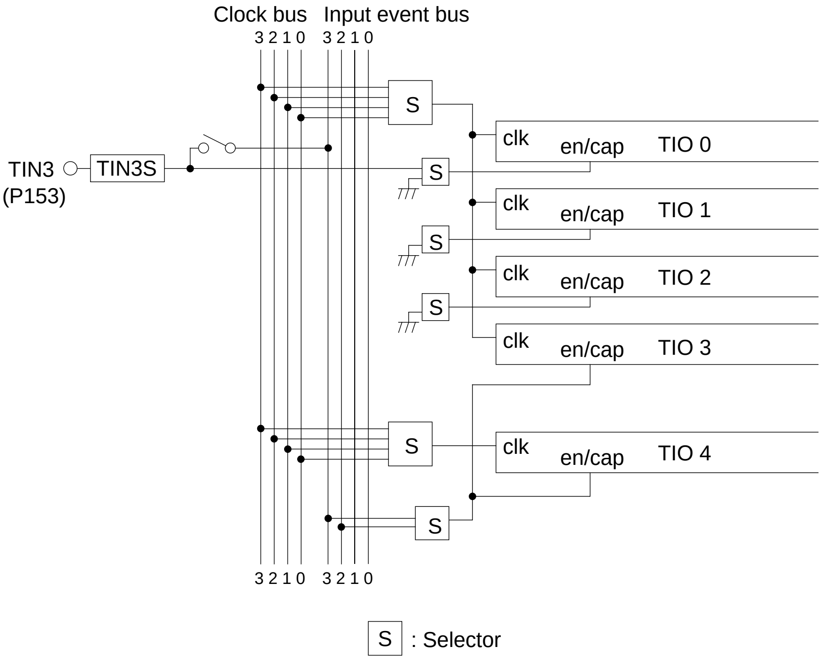
Notes:
- During measurement (free-run/clear) input mode, the TIO1 and TIO2 timers do not have the capture function.
- This register must always be accessed in half word.
- Always make sure the counter has stopped and is idle before setting or changing operation modes.

(Continued from the preceding page)

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 9-11 | TIO1M<br>(TIO1 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Use inhibited | ○ | ○ |
| 12 | TIO0ENS (TIO0 enable/<br>measure input source selection) | 0: No selection<br>1: External input TIN3 | ○ | ○ |
| 13-15 | TIO0M<br>(TIO0 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Notes: • This register must always be accessed in halfwords.

• Always make sure the counter has stopped and is idle before setting or changing operation modes.



Note: • This diagram is shown for the explanation of TIO control registers, and is partly omitted.

**Figure 10.4.6 Outline Diagram of TIO0-4 Clock/Enable Inputs**

### ■ TIO0-3 Control Register 1 (TIO03CR1)

<Address:H'0080 031D>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | | | | | TIO03CKS | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-13 | No functions assigned | | 0 | – |
| 14,15 | TIO03CKS<br>(TIO0-3 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |

### ■ TIO4 Control Register (TIO4CR)

<Address:H'0080 034A>

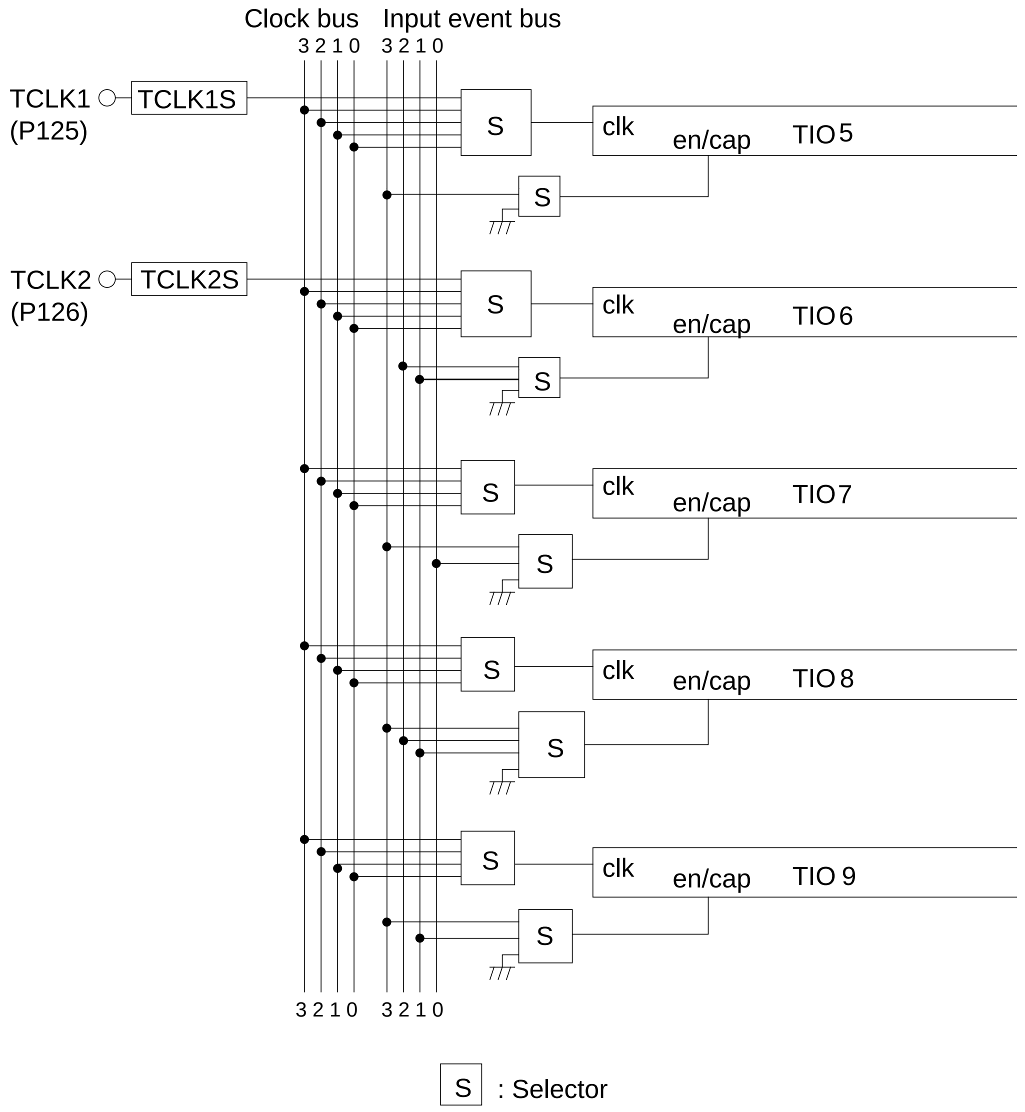
<table>
<tr><th>D0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>D7</th></tr>
<tr><td colspan="2">TIO4CKS</td><td>TIO4EEN</td><td colspan="2">TIO34ENS</td><td colspan="3">TIO4M</td></tr>
</table>

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0, 1 | TIO4CKS<br>(TIO4 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |
| 2 | TIO4EEN (Note 1)<br>(TIO4 external input enable) | 0: Disables external input<br>1: Enables external input | ○ | ○ |
| 3,4 | TIO34ENS<br>(TIO3,4 enable/measure<br>input source selection) | 0X: No selection<br>10: Input event bus 2<br>11: Input event bus 3 | ○ | ○ |
| 5-7 | TIO4M<br>(TIO4 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Note 1: During measure free-run/clear input mode, even if this bit is set to 0 (external input disabled), when a capture signal is entered from an external device, the counter value at that point in time is written to the measure register. However, because in measure clear input mode, if this bit = 0 (external input disabled), the counter value is not initialized (H'FFFF) upon capture, we recommend that this bit be set to 1 (external input enabled) when using measure clear input mode.

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.



Note: • This is an outline diagram shown for the explanation of TIO Control Register.

**Figure 10.4.7 Outline Diagram of TIO5-9 Clock/Enable Inputs**

### ■ TIO5 Control Register (TIO5CR)

<Address:H'0080 034B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TIO5CKS | | | TIO5ENS | | TIO5M | | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-10 | TIO5CKS<br>(TIO5 clock source selection) | 0XX: External input TCLK1<br>100: Clock bus 0<br>101: Clock bus 1<br>110: Clock bus 2<br>111: Clock bus 3 | ○ | ○ |
| 11,12 | TIO5ENS<br>(TIO5 enable/measure input source selection) | 0X: No selection<br>10: No selection<br>11: Input event bus 3 | ○ | ○ |
| 13-15 | TIO5M<br>(TIO5 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.

## ■ TIO6 Control Register (TIO6CR)

<Address:H'0080 036A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| TIO6CKS | | | TIO6ENS | | TIO6M | | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-2 | TIO6CKS<br>(TIO6 clock source selection) | 0XX: External input TCLK2<br>100: Clock bus 0<br>101: Clock bus 1<br>110: Clock bus 2<br>111: Clock bus 3 | ○ | ○ |
| 3,4 | TIO6ENS<br>(TIO6 enable/measure input source selection) | 0X: No selection<br>10: Input event bus 2<br>11: Input event bus 3 | ○ | ○ |
| 5-7 | TIO6M<br>(TIO6 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.

## ■ TIO7 Control Register (TIO7CR)

<Address:H'0080 036B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  | TIO7CKS |  | TIO7ENS |  | TIO7M |  |  |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | No functions assigned |  | 0 | – |
| 9,10 | TIO7CKS<br>(TIO7 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |
| 11,12 | TIO7ENS<br>(TIO7 enable/measure input source selection) | 0X: No selection<br>10: Input event bus 0<br>11: Input event bus 3 | ○ | ○ |
| 13-15 | TIO7M<br>(TIO7 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.

### ■ TIO8 Control Register (TIO8CR)

<Address:H'0080 038A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| TIO8CKS | | TIO8ENS | | | TIO8M | | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0,1 | TIO8CKS<br>(TIO8 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |
| 2-4 | TIO8ENS<br>(TIO8 enable/measure<br>input source selection) | 0XX: No selection<br>100: No selection<br>101: Input event bus 1<br>110: Input event bus 2<br>111: Input event bus 3 | ○ | ○ |
| 5-7 | TIO8M<br>(TIO8 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.

## ■ TIO9 Control Register (TIO9CR)

<Address:H'0080 038B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  | TIO9CKS |  | TIO9ENS |  | TIO9M |  |  |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | No functions assigned |  | 0 | – |
| 9,10 | TIO9CKS<br>(TIO9 clock source selection) | 00: Clock bus 0<br>01: Clock bus 1<br>10: Clock bus 2<br>11: Clock bus 3 | ○ | ○ |
| 11,12 | TIO9ENS<br>(TIO9 enable/measure<br>input source selection) | 0X: No selection<br>10: Input event bus 1<br>11: Input event bus 3 | ○ | ○ |
| 13-15 | TIO9M<br>(TIO9 operation mode selection) | 000: Single-shot output mode<br>001: Delayed single-shot output mode<br>010: Continuous output mode<br>011: PWM output mode<br>100: Measure clear input mode<br>101: Measure free-run input mode<br>11X: Noise processing input mode | ○ | ○ |

Note: • Always make sure the counter has stopped and is idle before setting or changing operation modes.

### 10.4.5 TIO Counter (TIO0CT-TIO9CT)

| | |
|---|---|
| ■ **TIO0 Counter (TIO0CT)** | <Address:H'0080 0300> |
| ■ **TIO1 Counter (TIO1CT)** | <Address:H'0080 0310> |
| ■ **TIO2 Counter (TIO2CT)** | <Address:H'0080 0320> |
| ■ **TIO3 Counter (TIO3CT)** | <Address:H'0080 0330> |
| ■ **TIO4 Counter (TIO4CT)** | <Address:H'0080 0340> |
| ■ **TIO5 Counter (TIO5CT)** | <Address:H'0080 0350> |
| ■ **TIO6 Counter (TIO6CT)** | <Address:H'0080 0360> |
| ■ **TIO7 Counter (TIO7CT)** | <Address:H'0080 0370> |
| ■ **TIO8 Counter (TIO8CT)** | <Address:H'0080 0380> |
| ■ **TIO9 Counter (TIO9CT)** | <Address:H'0080 0390> |

<table>
<tr><th>D0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>D15</th></tr>
<tr><td colspan="16">TIO0CT-TIO9CT</td></tr>
</table>

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TIO0CT-TIO9CT | 16-bit counter value | ○ | △ |

W=△ : Write to this register is not accepted in PWM output mode.

Note: • This register must always be accessed in halfwords.

The TIO Counters are a 16-bit down-counter. After the timer is enabled (by writing to the enable bit in software or by external input), the counter starts counting synchronously with the count clock. The counter cannot be written to during PWM output mode.

### 10.4.6 TIO Reload 0/ Measure Register (TIO0RL0-TIO9RL0)

| | |
|---|---|
| **■ TIO0 Reload 0/ Measure Register (TIO0RL0)** | <Address:H'0080 0306> |
| **■ TIO1 Reload 0/ Measure Register (TIO1RL0)** | <Address:H'0080 0316> |
| **■ TIO2 Reload 0/ Measure Register (TIO2RL0)** | <Address:H'0080 0326> |
| **■ TIO3 Reload 0/ Measure Register (TIO3RL0)** | <Address:H'0080 0336> |
| **■ TIO4 Reload 0/ Measure Register (TIO4RL0)** | <Address:H'0080 0346> |
| **■ TIO5 Reload 0/ Measure Register (TIO5RL0)** | <Address:H'0080 0356> |
| **■ TIO6 Reload 0/ Measure Register (TIO6RL0)** | <Address:H'0080 0366> |
| **■ TIO7 Reload 0/ Measure Register (TIO7RL0)** | <Address:H'0080 0376> |
| **■ TIO8 Reload 0/ Measure Register (TIO8RL0)** | <Address:H'0080 0386> |
| **■ TIO9 Reload 0/ Measure Register (TIO9RL0)** | <Address:H'0080 0396> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TIO0RL0-TIO9RL0 | | | | | | | | | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TIO0RL0-TIO9RL0 | 16-bit reload register value | ○ | △ |

W=△ : Write to this register is not accepted in measure input mode.

Note: • This register must always be accessed in halfwords.

The TIO Reload 0/ Measure Registers serve dual purposes as a register for reloading TIO Count Registers (TIO0CT-TIO9CT) with data, and as a measure register during measure input mode. These registers are disabled against write during measure input mode.

It is in the following cases that the content of reload 0 register is loaded into the counter:

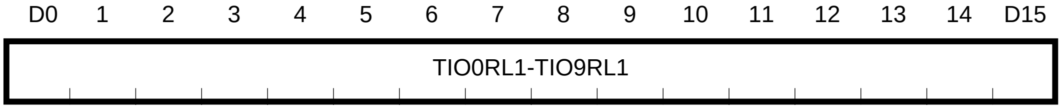
- When after the counter started counting in noise processing input mode, the input signal is inverted and a valid-level signal is entered again before the counter underflows
- When the counter is enabled in single-shot mode
- When the counter underflowed in delayed single-shot or continuous mode
- When the counter is enabled in PWM mode and when the counter value set by reload 1 register underflowed

Writing data to the reload 0 register does not mean that the data is loaded into the counter simultaneously.

When used as a measure register, the counter value is latched into the measure register by an event input.

### 10.4.7 TIO Reload 1 Registers (TIO0RL1-TIO9RL1)

| | |
|---|---|
| ■ **TIO0 Reload 1 Register (TIO0RL1)** | <Address:H'0080 0304> |
| ■ **TIO1 Reload 1 Register (TIO1RL1)** | <Address:H'0080 0314> |
| ■ **TIO2 Reload 1 Register (TIO2RL1)** | <Address:H'0080 0324> |
| ■ **TIO3 Reload 1 Register (TIO3RL1)** | <Address:H'0080 0334> |
| ■ **TIO4 Reload 1 Register (TIO4RL1)** | <Address:H'0080 0344> |
| ■ **TIO5 Reload 1 Register (TIO5RL1)** | <Address:H'0080 0354> |
| ■ **TIO6 Reload 1 Register (TIO6RL1)** | <Address:H'0080 0364> |
| ■ **TIO7 Reload 1 Register (TIO7RL1)** | <Address:H'0080 0374> |
| ■ **TIO8 Reload 1 Register (TIO8RL1)** | <Address:H'0080 0384> |
| ■ **TIO9 Reload 1 Register (TIO9RL1)** | <Address:H'0080 0394> |



<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TIO0RL1-TIO9RL1 | 16-bit reload register value | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The TIO Reload 1 Registers are used to reload data into the TIO Counter Registers (TIO0CT-TIO9CT).
The content of reload 1 register is loaded into the counter in the following cases:

- When the count value set by reload 0 register underflowed in PWM output mode

Writing data to the reload 1 register does not mean that the data is loaded into the counter simultaneously.

## 10.4.8 TIO Enable Control Registers

### ■ TIO0-9 Enable Protect Register (TIOPRO)

<Address:H'0080 03BC>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | TIO9<br>PRO | TIO8<br>PRO | TIO7<br>PRO | TIO6<br>PRO | TIO5<br>PRO | TIO4<br>PRO | TIO3<br>PRO | TIO2<br>PRO | TIO1<br>PRO | TIO0<br>PRO |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-5 | No functions assigned | | 0 | – |
| 6 | TIO9PRO (TIO9 Enable Protect) | 0: Enables rewrite<br>1: Disables rewrite | ○ | ○ |
| 7 | TIO8PRO (TIO8 Enable Protect) | | | |
| 8 | TIO7PRO (TIO7 Enable Protect) | | | |
| 9 | TIO6PRO (TIO6 Enable Protect) | | | |
| 10 | TIO5PRO (TIO5 Enable Protect) | | | |
| 11 | TIO4PRO (TIO4 Enable Protect) | | | |
| 12 | TIO3PRO (TIO3 Enable Protect) | | | |
| 13 | TIO2PRO (TIO2 Enable Protect) | | | |
| 14 | TIO1PRO (TIO1 Enable Protect) | | | |
| 15 | TIO0PRO (TIO0 Enable Protect) | | | |

Note: • This register must always be accessed in halfwords.

The TIO0-9 Enable Protect Register controls rewriting of the TIO count enable bit described in the next page by enabling or disabling rewrite.

## ■ TIO0-9 Count Enable Register (TIOCEN)

<Address:H'0080 03BE>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | TIO9<br>CEN | TIO8<br>CEN | TIO7<br>CEN | TIO6<br>CEN | TIO5<br>CEN | TIO4<br>CEN | TIO3<br>CEN | TIO2<br>CEN | TIO1<br>CEN | TIO0<br>CEN |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-5 | No functions assigned | | 0 | – |
| 6 | TIO9CEN (TIO9 count enable) | 0: Stops count<br>1: Enables count | ○ | ○ |
| 7 | TIO8CEN (TIO8 count enable) | | | |
| 8 | TIO7CEN (TIO7 count enable) | | | |
| 9 | TIO6CEN (TIO6 count enable) | | | |
| 10 | TIO5CEN (TIO5 count enable) | | | |
| 11 | TIO4CEN (TIO4 count enable) | | | |
| 12 | TIO3CEN (TIO3 count enable) | | | |
| 13 | TIO2CEN (TIO2 count enable) | | | |
| 14 | TIO1CEN (TIO1 count enable) | | | |
| 15 | TIO0CEN (TIO0 count enable) | | | |

Note: • This register must always be accessed in halfwords.

The TIO0-9 Count Enable Register controls operation of TIO counters. To enable the counter in software, enable the relevant TIO0-9 Enable Protect Register for write and set the count enable bit by writing a 1. To stop the counter, enable the TIO0-9 Enable Protect Register for write and reset the count enable bit by writing a 0.

In all but continuous mode, when the counter stops due to an occurrence of underflow, the count enable bit is automatically reset to 0. Therefore, what you get by reading the TIO0-9 Count Enable Register is the status that indicates the counter's operating status (active or idle).

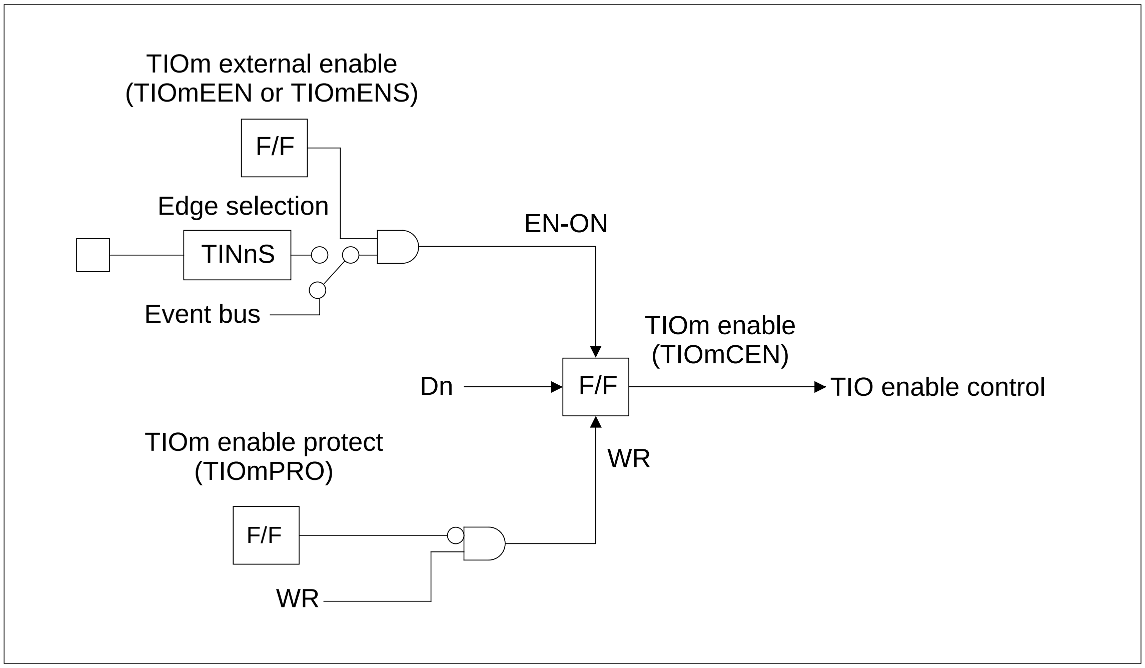


**Figure 10.4.8 Configuration of the TIO Enable Circuit**

## 10.4.9 Operation in TIO Measure Free-run/Clear Input Modes

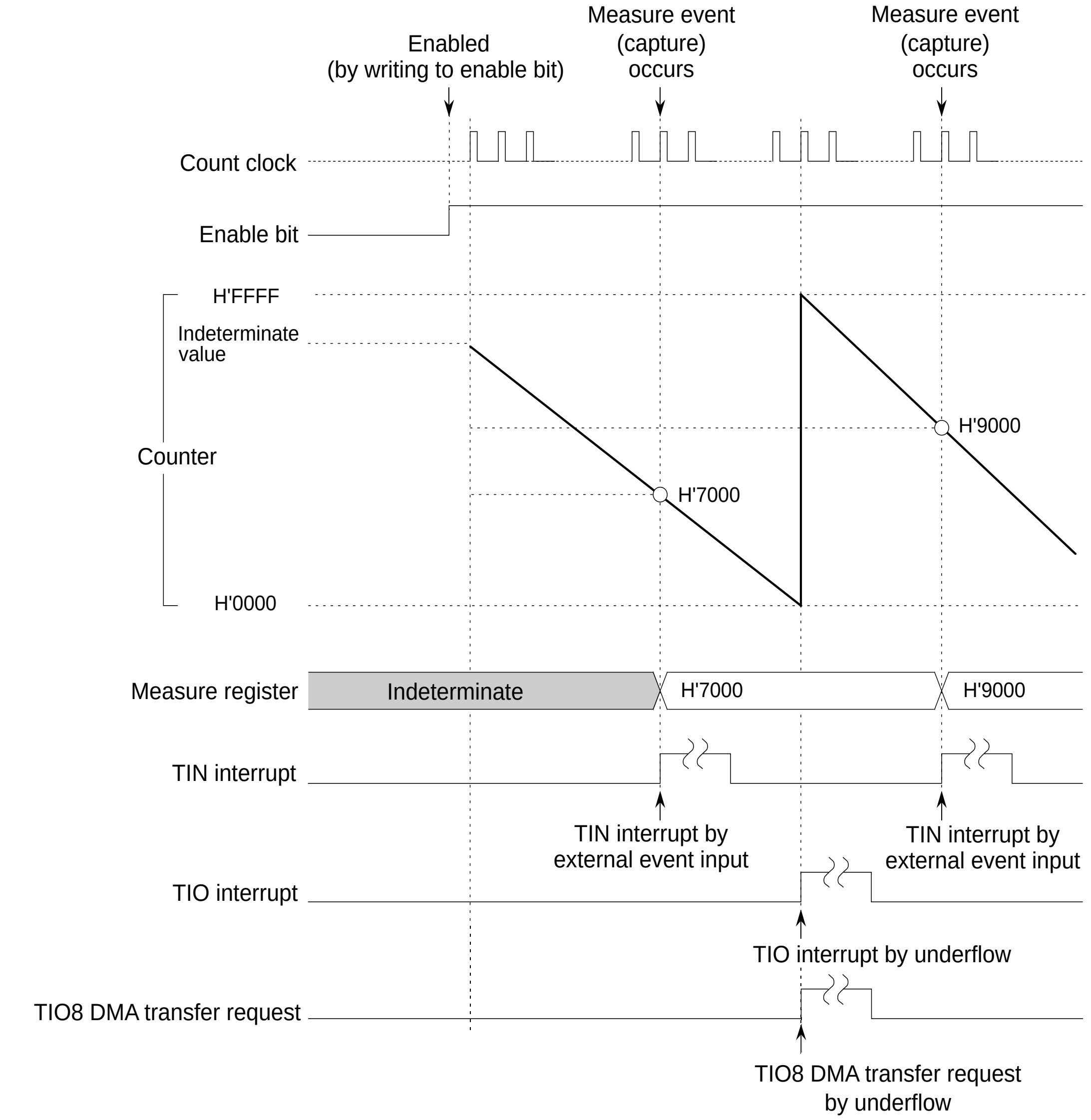
### (1) Outline of TIO measure free-run/clear input modes

In TIO measure free-run/clear input modes, the timer measures a duration of time from when it starts counting till when an external capture signal is entered. An interrupt can be generated by a counter underflow or execution of measure operation. Also, a DMA transfer request (for only the TIO8) can be generated when the counter underflows.

After the timer is enabled (by writing to the enable bit in software), the counter starts counting down synchronously with the count clock. When a capture signal is entered from an external device, the counter value at that point in time is written to the measure register.

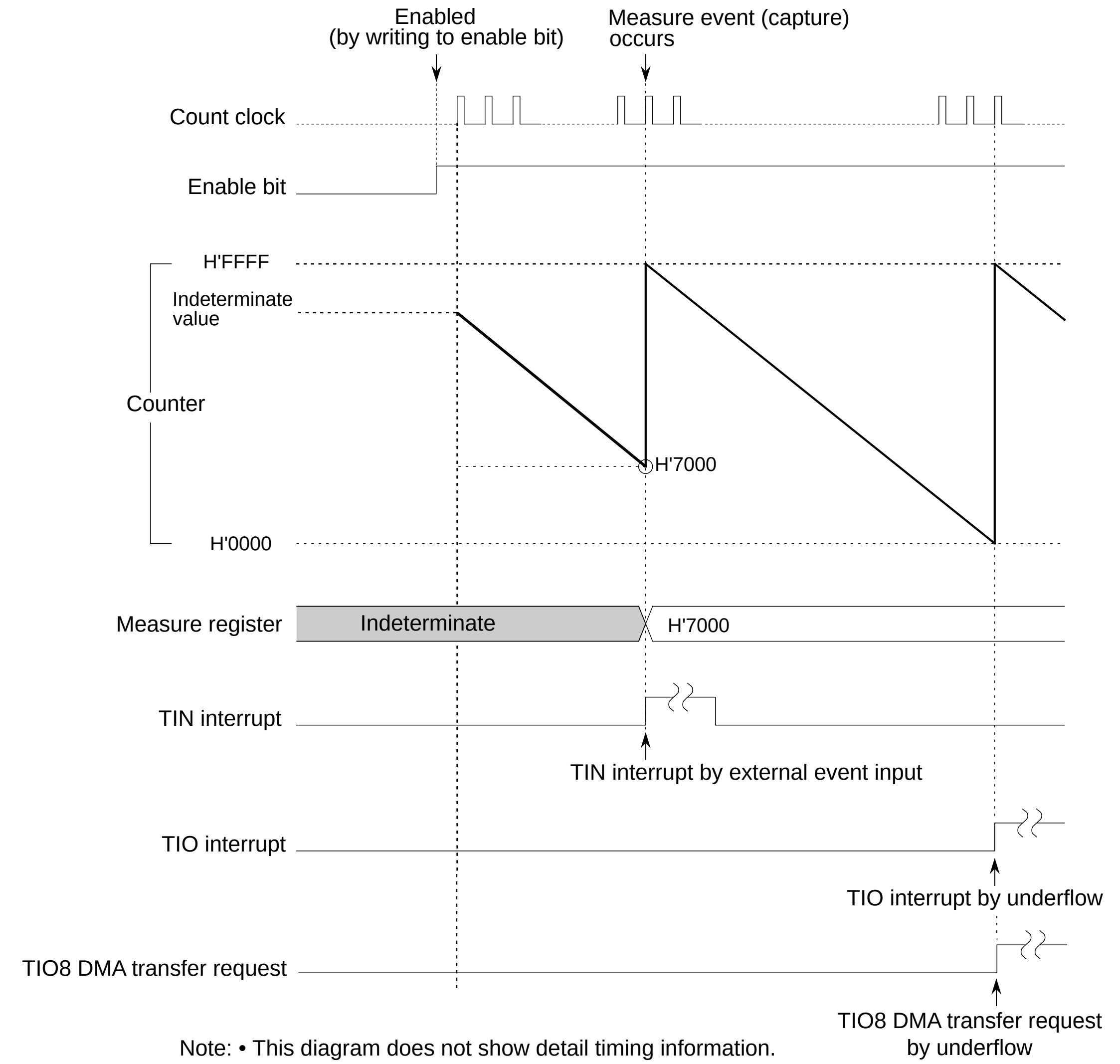
Especially in measure clear input mode, the counter value is initialized to H'FFFF upon capture, from which the counter starts counting down again. When the counter underflows after reaching the minimum count, it starts counting down from H'FFFF again. In measure free-run input mode, the counter continues counting down even after capture and upon underflow, recycles to H'FFFF, from which it starts counting down again.

To stop the counter, disable count by writing to the enable bit in software.



Note: • This diagram does not show detail timing information.

**Figure 10.4.9 Typical Operation in Measure Free-run Input Mode**



Note: • This diagram does not show detail timing information.

**Figure 10.4.10 Typical Operation in Measure Clear Input Mode**

### (2) Precautions to be observed when using TIO measure free-run/clear input modes

The following describes precautions to be observed when using TIO measure free-run/clear input modes.

- If measure event input and write to the counter occur simultaneously in the same clock period, the write value is set in the counter while at the same time latched into the measure register.

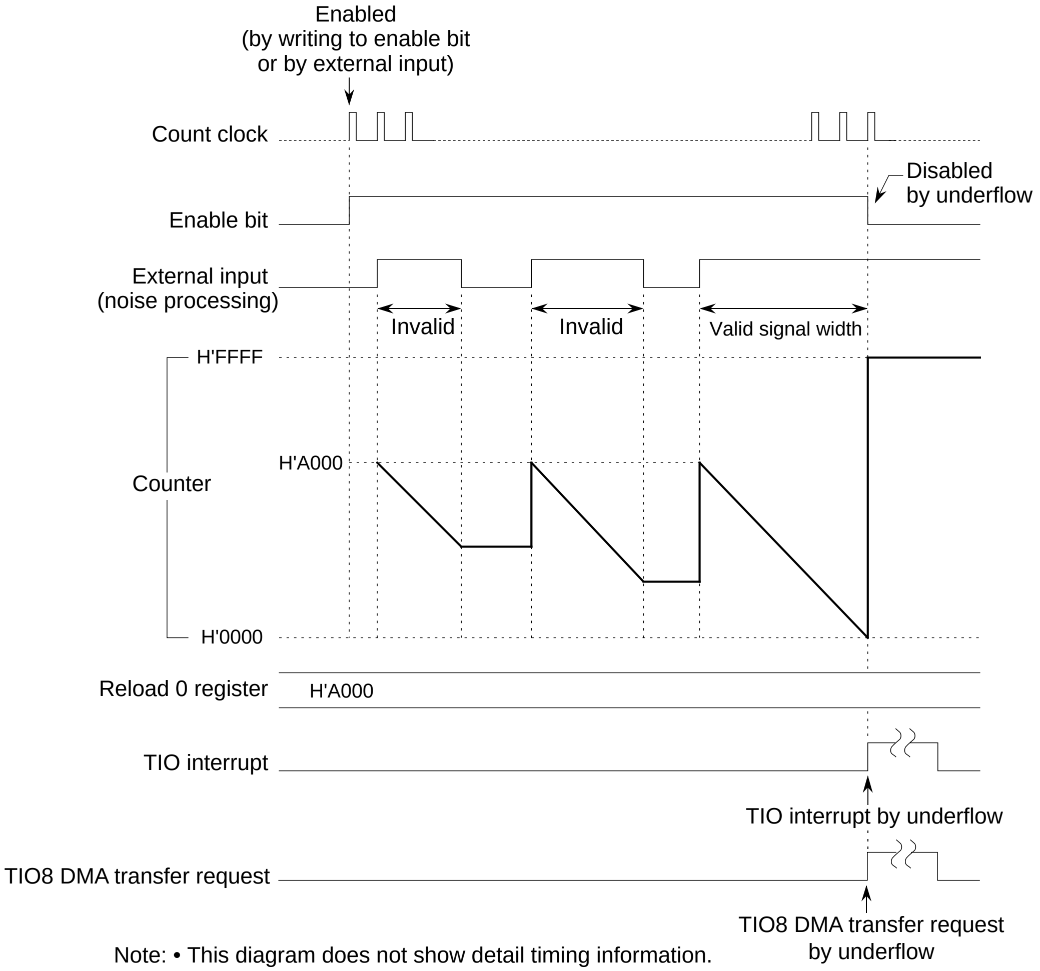
### 10.4.10 Operation in TIO Noise Processing Input Mode

In noise processing input mode, the timer detects the status of an input signal that it remained in the same state for over a predetermined time.

In noise processing input mode, the counter is started by entering a high or low-level signal from an external device and if the signal remains in the same state for over a predetermined time before the counter underflows, the counter stops after generating an interrupt. If the valid-level signal being applied turns to an invalid level before the counter underflows, the counter temporarily stops counting and when a valid-level signal is entered again, it is reloaded with the initial count and restarts counting. The valid count value is (reload 0 register set value + 1).

The timer stops at the same time the counter underflows or count is disabled by writing to the enable bit.

Also, an interrupt as well as a DMA transfer request (for only the TIO8) can be generated by a counter underflow.



Note: • This diagram does not show detail timing information.

**Figure 10.4.11 Typical Operation in Noise Processing Input Mode**

### 10.4.11 Operation in TIO PWM Output Mode

**(1) Outline of TIO PWM output mode**

In PWM output mode, the timer uses two reload registers to generate a waveform with a given duty cycle.

When after setting the initial values in reload 0 and reload 1 registers, the timer is enabled (by writing to the enable bit in software or by external input), it loads the reload 0 register value into the counter synchronously with the count clock letting the counter start counting down. The first time the counter underflows, the reload 1 register value is loaded into the counter letting it continue counting. Thereafter, the counter is reloaded with the reload 0 and reload 1 register values alternately each time an underflow occurs. The valid count values are (reload 0 register set value + 1) and (reload 1 register set value + 1). The timer stops at the same time count is disabled by writing to the enable bit (and not in synchronism with PWM output period).

The F/F output waveform in PWM output mode is inverted (F/F output levels change from low to high, or vice versa) at count startup and upon each underflow. An interrupt can be generated when the counter underflows every even time (second time, fourth time, and so on) after being enabled. Also, a DMA transfer request (for only the TIO8) can be generated every time the counter underflows.

Note that TIO's PWM output mode does not have the correction function.

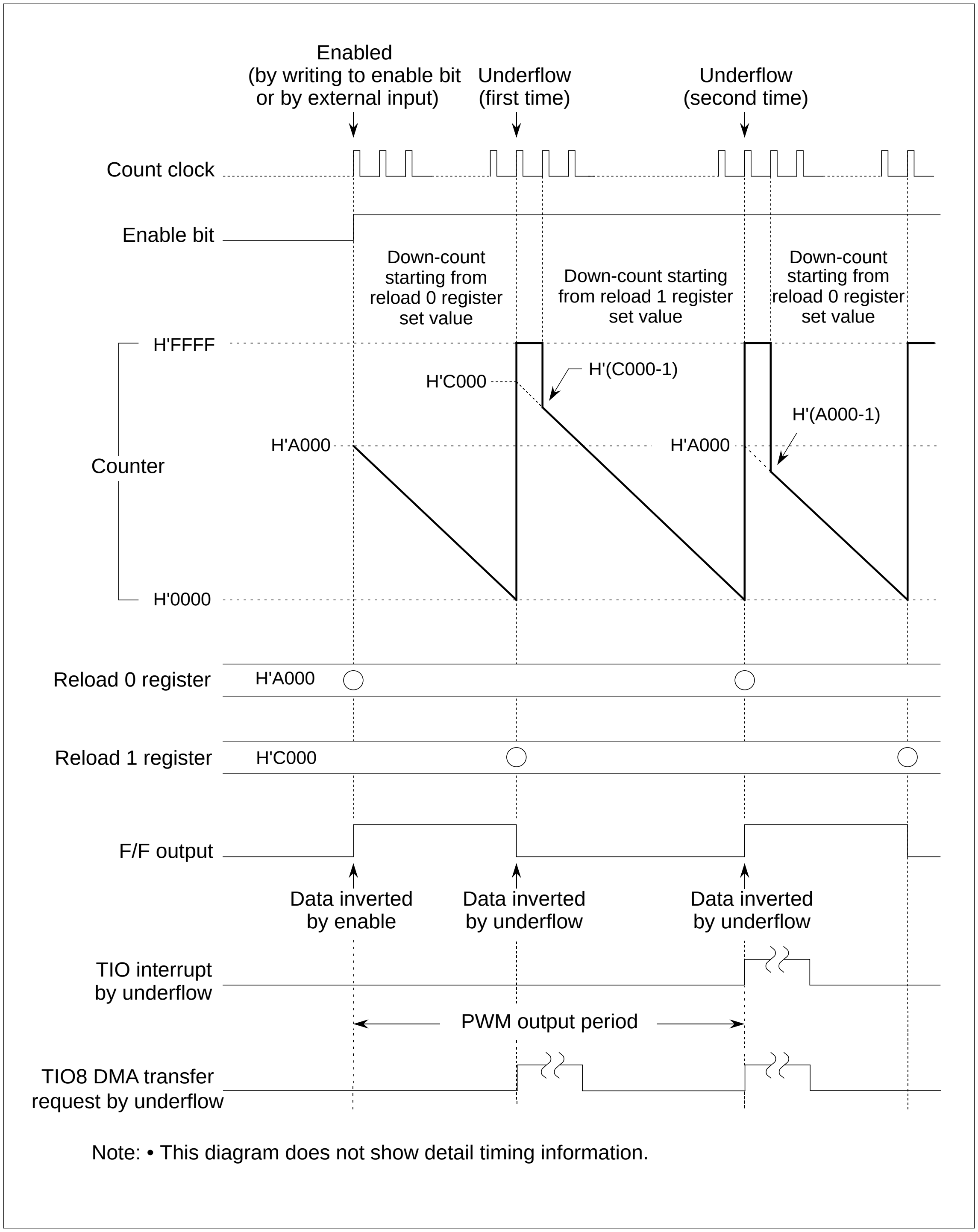


**Figure 10.4.12 Typical Operation in PWM Output Mode**

### (2) Reload register updates in TIO PWM output mode

In PWM output mode, when the timer remains idle, reload 0 and reload 1 registers are updated at the same time data are written to the registers. But when the timer is active, reload 1 register is updated by updating reload 0 register. However, when you read reload 0 and reload 1 registers, the values you get are always the data written to the registers.

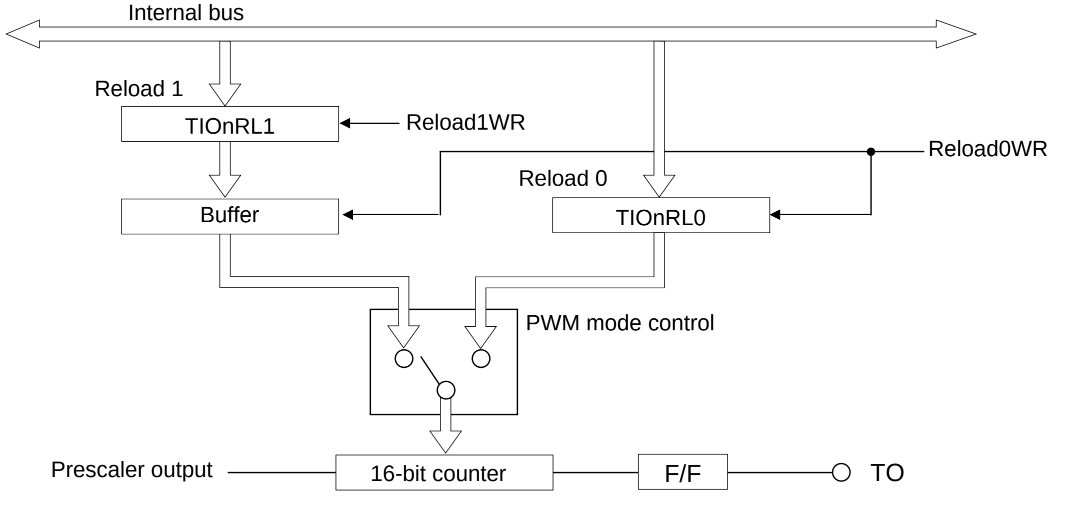


**Figure 10.4.13 PWM Circuit Diagram**

If you want to rewrite reload 0 and reload 1 registers while the timer is operating, rewrite reload 1 register first and then reload 0 register. In this way, reload 0 and reload 1 registers both are updated synchronously with PWM periods, from which the timer starts operating again. This operation can normally be performed collectively by accessing register addresses wordwise (in 32 bits) beginning with that of reload 1 register. (Data are automatically written to reload 1 and then reload 0 registers in succession.)

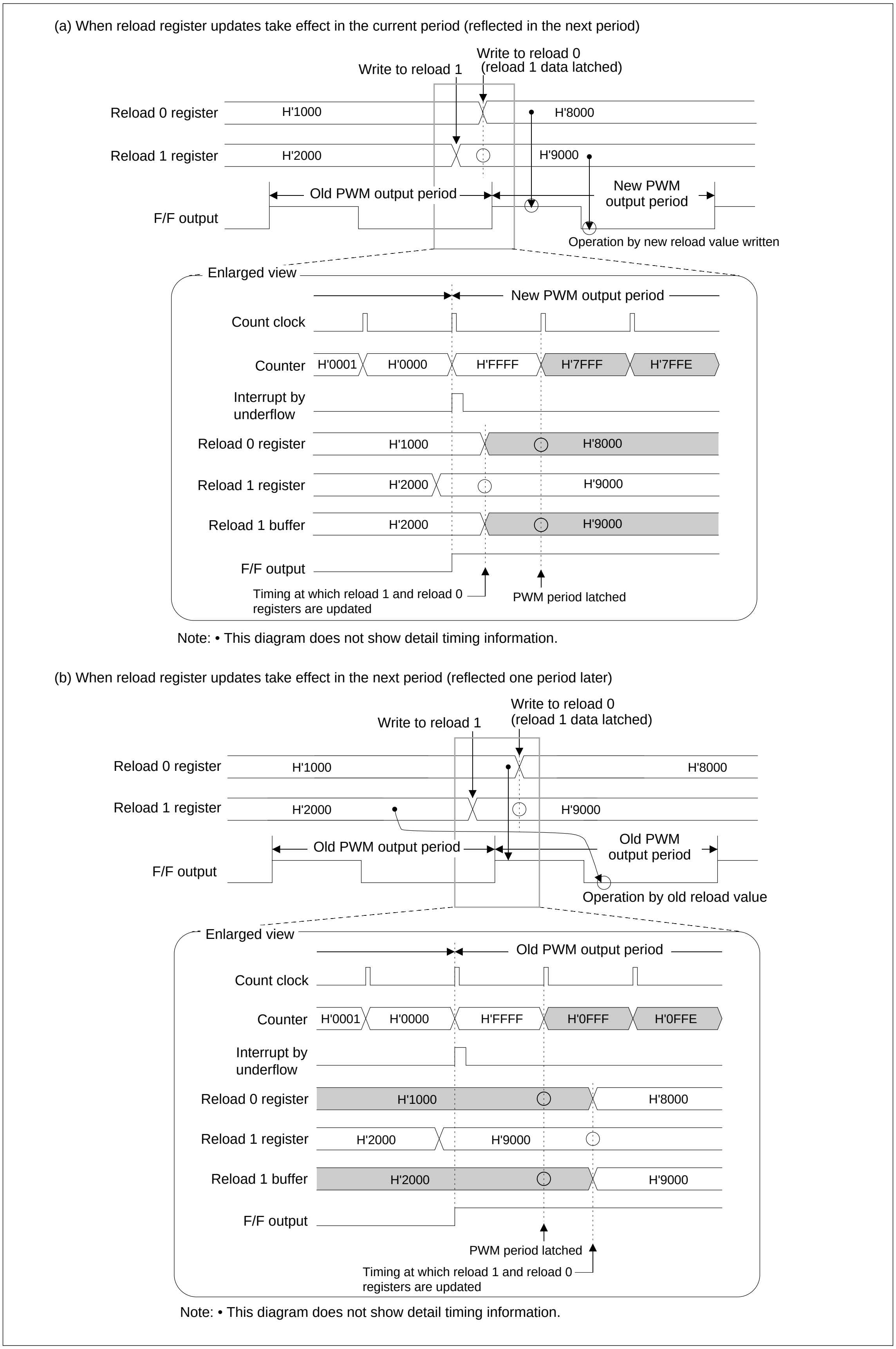
If you update the reload registers in reverse by updating reload 0 register first and then reload 1 register, only reload 0 register is updated. when you read reload 0 and reload 1 registers, the values you get are always the data written to the registers, and not the reload values being actually used.

Note that when updating the PWM period, if the PWM period is terminated before you finished writing to reload 0, the PWM period is not updated in the current period and what you've set is reflected in the next period.

### (3) Precautions on using TIO PWM output mode

The following describes precautions to be observed when using TIO PWM output mode.

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority so that count is disabled.
- If the counter is accessed for read immediately after being reloaded pursuant to an underflow, the counter value temporarily reads as H’FFFF but immediately changes to (reload value – 1) at the next clock edge.
- Because the timer operates synchronously with the count clock, a count clock-dependent delay is included before F/F output is inverted after the timer is enabled.



(a) When reload register updates take effect in the current period (reflected in the next period)

Note: • This diagram does not show detail timing information.

(b) When reload register updates take effect in the next period (reflected one period later)

Note: • This diagram does not show detail timing information.

**Figure 10.4.14 Reload 0 and Reload 1 Register Updates in PWM Output Mode**

### 10.4.12 Operation in TIO Single-shot Output Mode (without Correction Function)

#### (1) Outline of TIO single-shot output mode

In single-shot output mode, the timer generates a pulse in width of (reload 0 register set value + 1) only once and stops.

When after setting the reload 0 register, the timer is enabled (by writing to the enable bit in software or by external input), it loads the content of reload 0 register into the counter synchronously with the count clock, letting the counter start counting. The counter counts down clock pulses and stops when it underflows after reaching the minimum count.

The F/F output waveform in single-shot output mode is inverted (F/F output levels change from low to high, or vice versa) at startup and upon underflow, generating a single-shot pulse waveform in width of (reload 0 register set value + 1) only once. Also, an interrupt as well as a DMA transfer request (for only  the TIO8) can be generated when the counter underflows.

The count value is (reload 0 register set value + 1). For details about count operation, also refer to Section 10.3.9, "Operation in TOP Single-shot Output Mode (with Correction Function)."

#### (2) Precautions to be observed when using TIO single-shot output mode

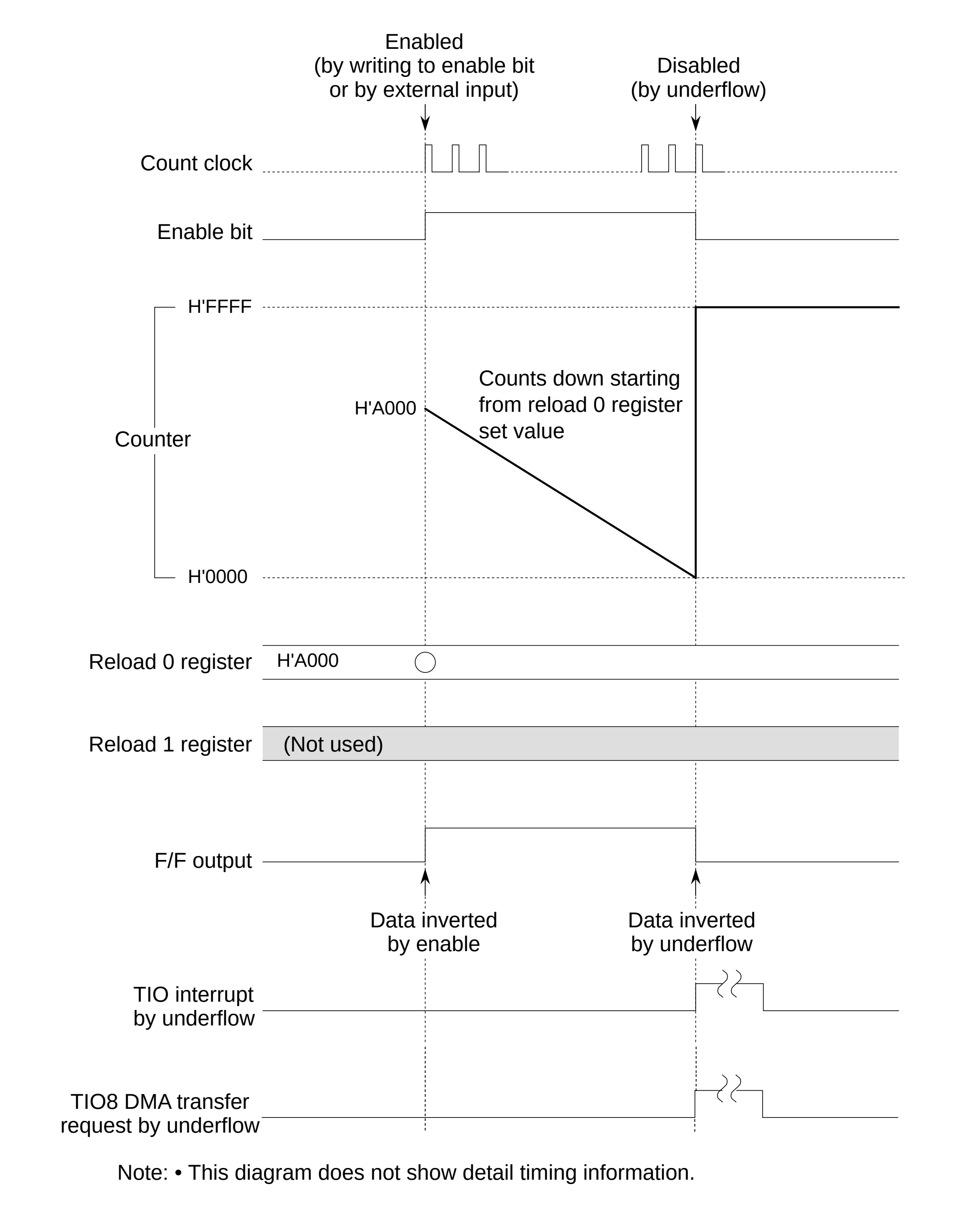
The following describes precautions to be observed when using TIO single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).

- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).

- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.



Note: • This diagram does not show detail timing information.

**Figure 10.4.15 Typical Operation in TIO Single-shot Output Mode (without Correction Function)**

### 10.4.13 Operation in TIO Delayed Single-shot Output Mode (without Correction Function)

#### (1) Outline of TIO delayed single-shot output mode

In delayed single-shot output mode, the timer generates a pulse in width of (reload 0 register set value + 1) only once, with the output delayed by an amount of time equal to (counter set value + 1) and then stops without performing any operation.

When after setting the counter and reload 0 register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock. The first time the counter underflows, the reload 0 register value is loaded into the counter causing it to continue counting down, and the counter stops when it underflows next time.

The F/F output waveform in delayed single-shot output mode is inverted (F/F output levels change from low to high, or vice versa) when the counter underflows first time and next, generating a single-shot pulse waveform in width of (reload 0 register set value + 1) only once, with the output delayed by an amount of time equal to (first set value of counter + 1). Also, an interrupt and a DMA transfer request (for only  the TIO8) can be generated when the counter underflows first time and next .

The valid count values are the (counter set value + 1) and (reload 0 register set value + 1). For details about count operation, also see Section 10.3.10, "Operation in TOP Delayed Single-shot Output Mode."

#### (2) Precautions to be observed when using TIO delayed single-shot output mode

The following describes precautions to be observed when using TIO delayed single-shot output mode.

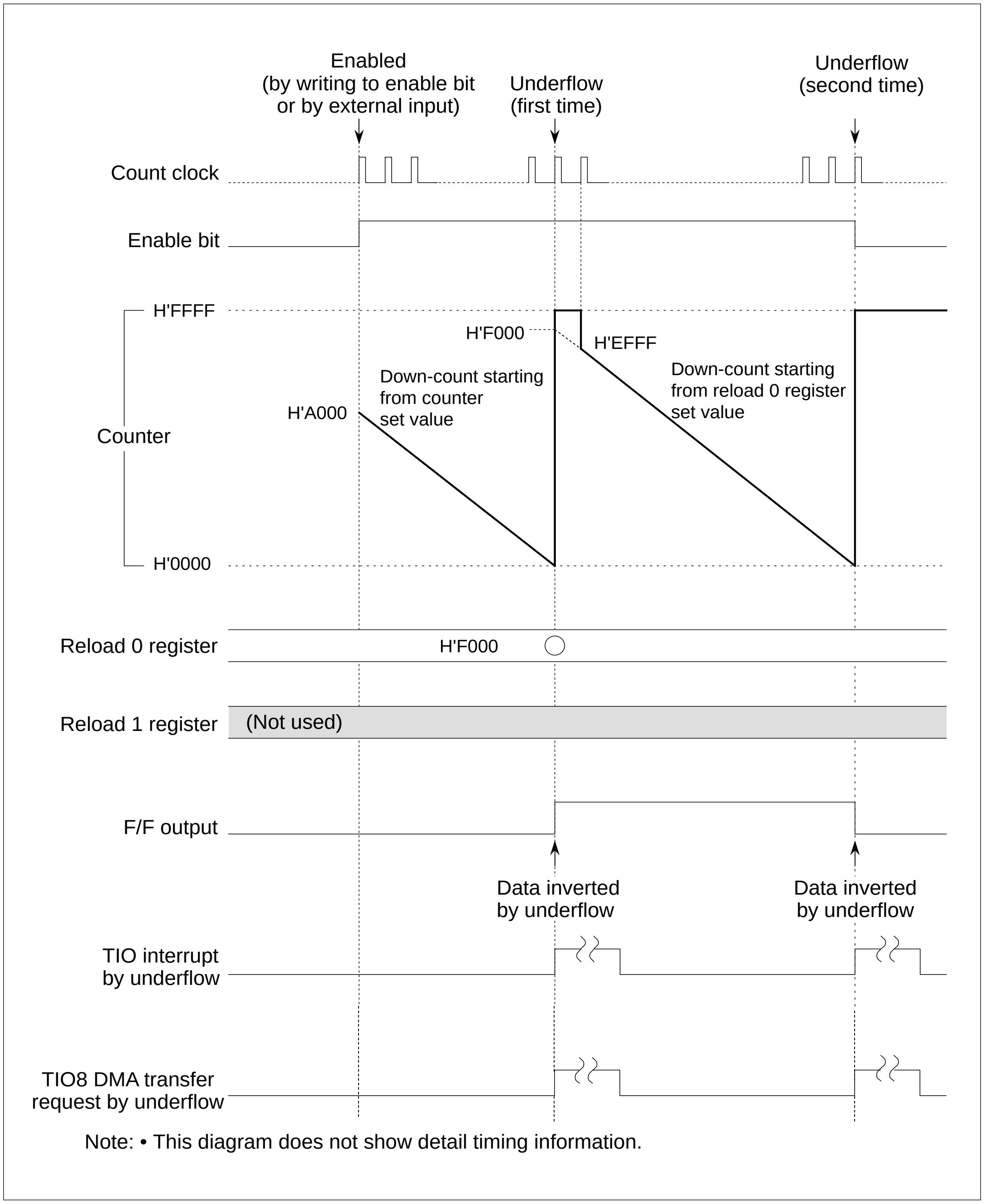
- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).

- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).

- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.

- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.



Note: • This diagram does not show detail timing information.

**Figure 10.4.16 Typical Operation in TIO Delayed Single-shot Output Mode (without Correction Function)**

### 10.4.14 Operation in TIO Continuous Output Mode (Without Correction Function)

#### (1) Outline of TIO continuous output mode

In continuous output mode, the timer counts down clock pulses starting from the set value of the counter and when the counter underflows, reloads it with reload 0 register value. Thereafter, this operation is repeated each time the counter underflows, thus generating consecutive pulses whose waveform is inverted in width of (reload 0 register set value + 1).

When after setting the counter and reload 0 register, the timer is enabled (by writing to the enable bit in software or by external input), it starts counting down from the counter's set value synchronously with the count clock and when the minimum count is reached, generates an underflow. This underflow causes the counter to be reloaded with the content of reload 0 register and start counting over again. Thereafter, this operation is repeated each time an underflow occurs. To stop the counter, disable count by writing to the enable bit in software.

The F/F output waveform in continuous output mode is inverted (F/F output levels change from low to high, or vice versa) at startup and upon underflow, generating consecutive pulses until the timer stops counting. Also, an interrupt as well as a DMA transfer request (for only the TIO8) can be generated each time the counter underflows.

The valid count values are the (counter set value + 1) and (reload 0 register set value + 1). For details about count operation, also see Section 10.3.11, "Operation in TOP Continuous Output Mode (Without Correction Function) ."

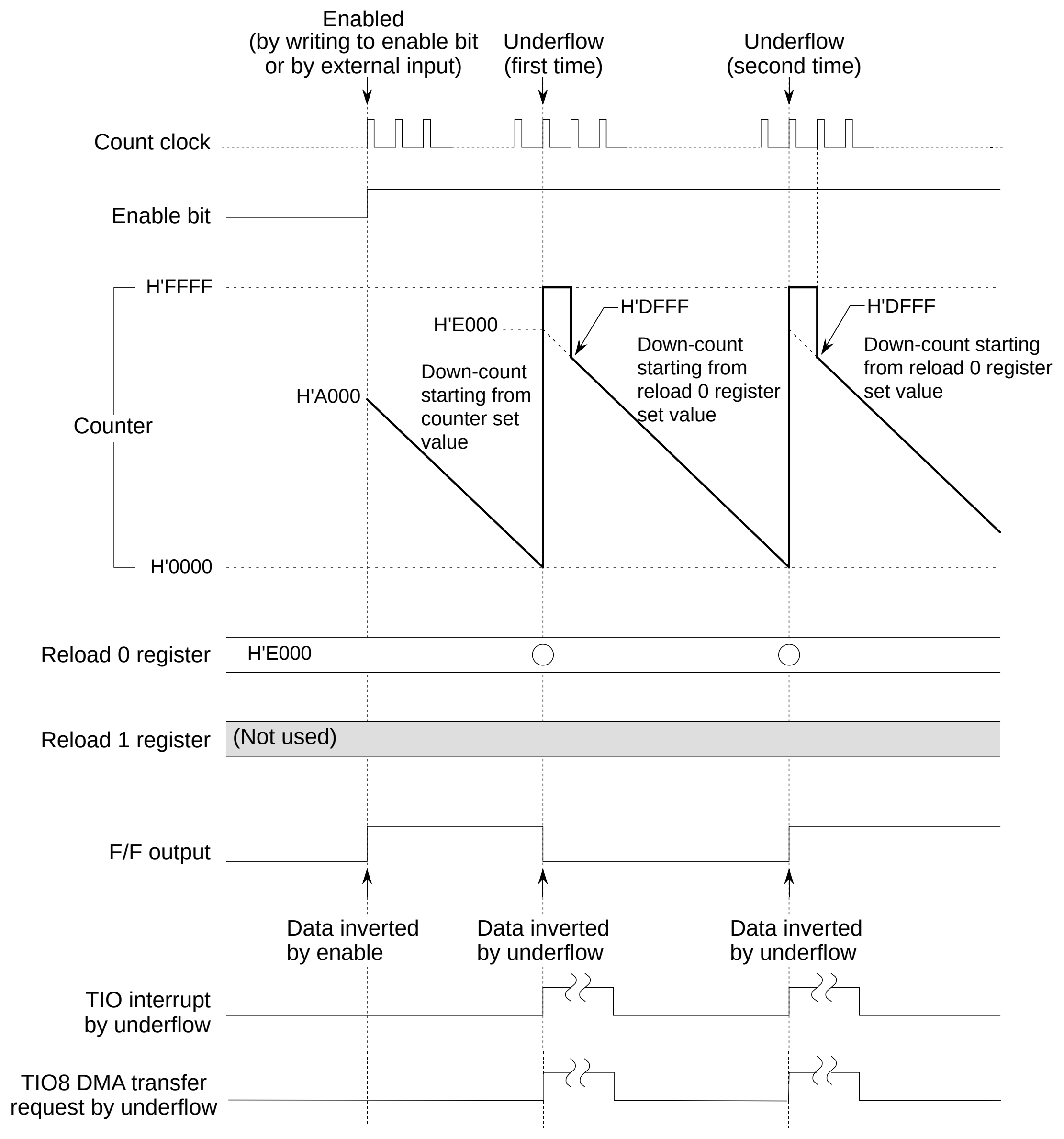
#### (2) Precautions to be observed when using TIO continuous output mode

The following describes precautions to be observed when using TIO continuous output mode.

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).

- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.

- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.



Note: • This diagram does not show detail timing information.

**Figure 10.4.17 Typical Operation in TIO Continuous Output Mode (without Correction Function)**

# 10.5 TMS (Input-related 16-bit Timer)

## 10.5.1 Outline of TMS

TMS (Timer Measure Small) is an input-related 16-bit timer capable of measuring input pulses in two circuit blocks comprising a total eight channels.
The table below shows specifications of TMS. The diagram in the next page shows a block diagram of TMS.

**Table 10.5.1 Specifications of TMS (Input-related 16-bit Timer)**

| Item | Specification |
|---|---|
| Number of channels | 8 channels (2 circuit blocks consisting of 4 channels each, 8 channels in total) |
| Counter | 16-bit up-counter x 2 |
| Measure register | 16-bit measure register x 8 |
| Timer startup | Started by writing to enable bit in software |
| Interrupt generation | Can be generated by a counter overflow |

## 10.5.2 Outline of TMS Operation

In TMS, when the timer is started by writing to the enable bit in software, the counter starts operating. The counter is a 16-bit up-counter, where when a measure signal is entered from an external device, the counter value is latched into each measure register.
The counter stops counting at the same time count is disabled by writing to the enable bit in software.
TIN interrupts can be generated by entering an external measurement signal (no TIN interrupts available for TMS0), and TMS interrupts can be generated by an overflow signal from the counter.

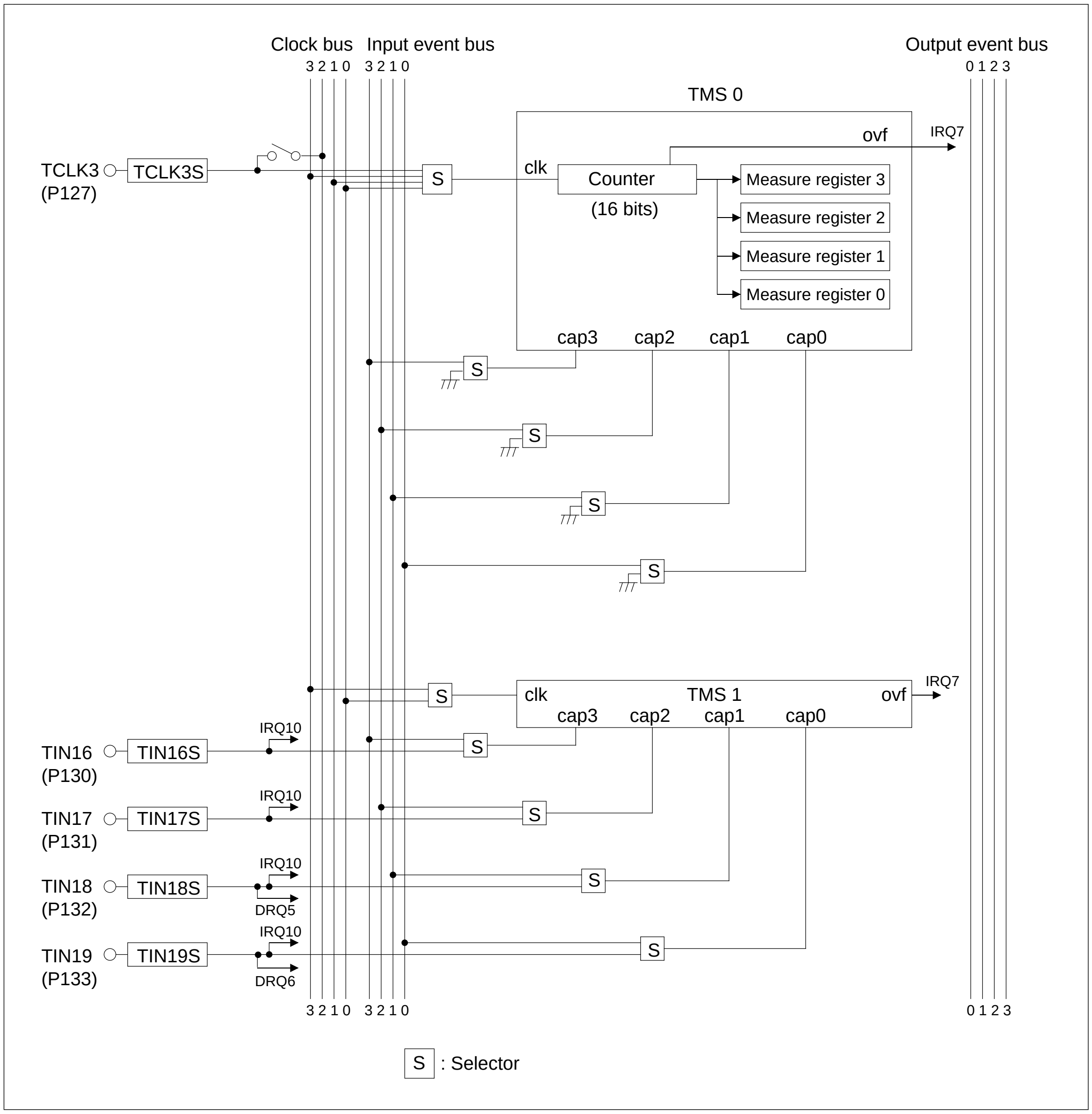


**Figure 10.5.1 Block Diagram of TMS (Input-related 16-bit Timer)**

### <Count clock-dependent delay>

- Because the timer operates synchronously with the count clock, there is a count clock-dependent delay from when the timer is enabled tilll it actuually starts operating.

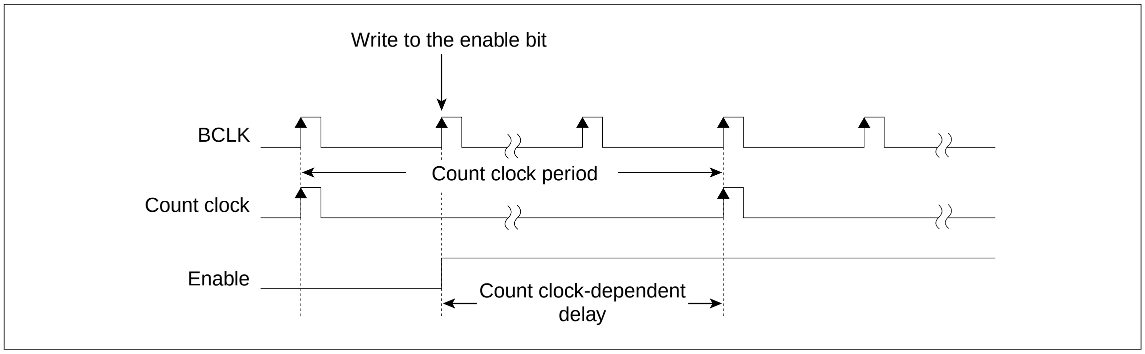


**Figure 10.5.2 Count Clock-Dependent Delay**

### 10.5.3 TMS Related Register Map

The diagram below shows a TMS related register map.

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 03C0 | TMS0 Counter (TMS0CT) | |
| H'0080 03C2 | TMS0 Measure 3 Register (TMS0MR3) | |
| H'0080 03C4 | TMS0 Measure 2 Register (TMS0MR2) | |
| H'0080 03C6 | TMS0 Measure 1 Register (TMS0MR1) | |
| H'0080 03C8 | TMS0 Measure 0 Register (TMS0MR0) | |
| H'0080 03CA | TMS0 Control Register (TMS0CR) | TMS1 Control Register (TMS1CR) |
| | ≈ | ≈ |
| H'0080 03D0 | TMS1 Counter (TMS1CT) | |
| H'0080 03D2 | TMS1 Measure 3 Register (TMS1MR3) | |
| H'0080 03D4 | TMS1 Measure 2 Register (TMS1MR2) | |
| H'0080 03D6 | TMS1 Measure 1 Register (TMS1MR1) | |
| H'0080 03D8 | TMS1 Measure 0 Register (TMS1MR0) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 10.5.3 TMS Related Register Map**

### 10.5.4 TMS Control Registers

The TMS control registers are used to select TMS0/1 input events and the counter clock source, as well as control counter startup. Following two TMS control registers are included:

- TMS0 Control Register (TMS0CR)
- TMS1 Control Register (TMS1CR)

### ■ TMS0 Control Register (TMS0CR)

<Address: H'0080 03CA>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| TMS0<br>SS0 | TMS0<br>SS1 | TMS0<br>SS2 | TMS0<br>SS3 | TMS0CKS | | | TMS0CEN |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | TMS0SS0<br>(TMS0 measure 0 source selection) | 0: No selection<br>1: Input event bus 0 | ○ | ○ |
| 1 | TMS0SS1<br>(TMS0 measure 1 source selection) | 0: No selection<br>1: Input event bus 1 | ○ | ○ |
| 2 | TMS0SS2<br>(TMS0 measure 2 source selection) | 0: No selection<br>1: Input event bus 2 | ○ | ○ |
| 3 | TMS0SS3<br>(TMS0 measure 3 source selection) | 0: No selection<br>1: Input event bus 3 | ○ | ○ |
| 4,5 | TMS0CKS<br>(TMS0 clock source selection) | 00: External input TCLK3<br>01: Clock bus 0<br>10: Clock bus 1<br>11: Clock bus 3 | ○ | ○ |
| 6 | No functions assigned | | 0 | – |
| 7 | TMS0CEN<br>(TMS0 count enable) | 0: Count stops<br>1: Count starts | ○ | ○ |

## ■ TMS1 Control Register (TMS1CR)

<Address: H'0080 03CB>

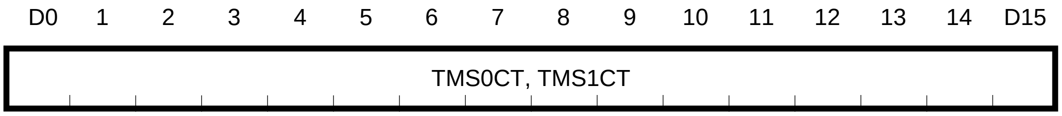
| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TMS1<br>SS0 | TMS1<br>SS1 | TMS1<br>SS2 | TMS1<br>SS3 | ▨ | TMS1CKS | ▨ | TMS1CEN |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | TMS1SS0<br>(TMS1 measure 0 source selection) | 0: External input TIN19<br>1: Input event bus 0 | ○ | ○ |
| 9 | TMS1SS1<br>(TMS1 measure 1 source selection) | 0: External input TIN18<br>1: Input event bus 1 | ○ | ○ |
| 10 | TMS1SS2<br>(TMS1 measure 2 source selection) | 0: External input TIN17<br>1: Input event bus 2 | ○ | ○ |
| 11 | TMS1SS3<br>(TMS1 measure 3 source selection) | 0: External input TIN16<br>1: Input event bus 3 | ○ | ○ |
| 12 | No functions assigned | | 0 | – |
| 13 | TMS1CKS<br>(TMS1 clock source selection) | 0: Clock bus 0<br>1: Clock bus 3 | ○ | ○ |
| 14 | No functions assigned | | 0 | – |
| 15 | TMS1CEN<br>(TMS1 count enable) | 0: Count stops<br>1: Count starts | ○ | ○ |

### 10.5.5 TMS Counter (TMS0CT, TMS1CT)

**■ TMS0 Counter (TMS0CT)** <Address: H'0080 03C0>
**■ TMS1 Counter (TMS1CT)** <Address: H'0080 03D0>



<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TMS0CT, TMS1CT | 16-bit counter value | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The TMS counters are a 16-bit up-counter, which starts counting when the timer is enabled (by writing to the enable bit in software). The counter can be read during operation.

### 10.5.6 TMS Measure Registers (TMS0MR3-0, TMS1MR3-0)

| | |
|---|---|
| ■ **TMS0 Measure 3 Register (TMS0MR3)** | <Address: H'0080 03C2> |
| ■ **TMS0 Measure 2 Register (TMS0MR2)** | <Address: H'0080 03C4> |
| ■ **TMS0 Measure 1 Register (TMS0MR1)** | <Address: H'0080 03C6> |
| ■ **TMS0 Measure 0 Register (TMS0MR0)** | <Address: H'0080 03C8> |
| ■ **TMS1 Measure 3 Register (TMS1MR3)** | <Address: H'0080 03D2> |
| ■ **TMS1 Measure 2 Register (TMS1MR2)** | <Address: H'0080 03D4> |
| ■ **TMS1 Measure 1 Register (TMS1MR1)** | <Address: H'0080 03D6> |
| ■ **TMS1 Measure 0 Register (TMS1MR0)** | <Address: H'0080 03D8> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMS0MR3-0, TMS1MR3-0 | | | | | | | | | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TMS0MR3-TMS0MR0<br>TMS1MR3-TMS1MR0 | 16-bit counter value | ○ | – |

Notes: • This register is a read-only register.
• This register can be accessed in either byte or halfword.

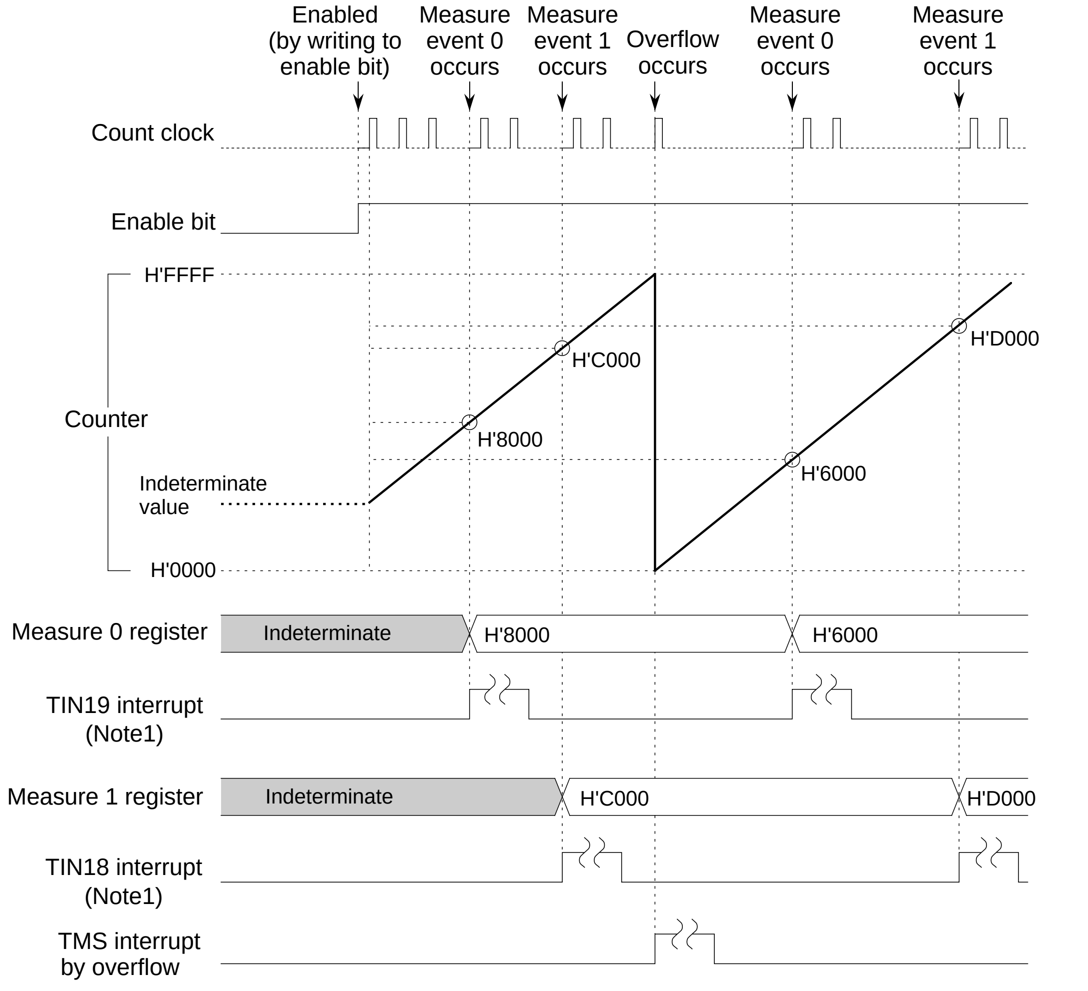
The TMS measure registers are used to latch counter contents upon event input. The TMS measure registers are a read-only register.

### 10.5.7 Operation of TMS Measure Input

#### (1) Outline of TMS measure input

In TMS measure input, the counter starts counting up clock pulses when the timer is actuated by writing to the enable bit in software. When event input is entered to TMS while the timer is operating, the counter value is latched into measure registers 0-3. The timer stops at the same time count is disabled by writing to the enable bit.

A TIN interrupt can be generated by entering a measure signal from an external device (for TMS1 only; no TIN interrupts available for TMS0). Also, when the counter overflows, a TMS interrupt can be generated.



Note1: TIN interrupts can be generated by entering an external measurement signal for TMS1 only (No TIN interrupts available for TMS0).

Note: • This diagram does not show detail timing information.

**Figure 10.5.4 Typical Operation in TMS Measure Input**

### (2) Precautions to be observed when using TMS measure input

The following describes precautions to be observed when using TMS measure input.

- If measure event input and write to the counter occur simultaneously in the same clock period, the write value is set in the counter while at the same time latched to the measure register.

## 10.6 TML (Input-related 32-bit Timer)

### 10.6.1 Outline of TML

TML (Timer Measure Large) is an input-related 32-bit timer capable of measuring input pulses in two circuit blocks comprising a total of eight channels.
The table below shows specifications of TML. The diagram in the next page shows a block diagram of TML.

**Table 10.6.1 Specifications of TML (Input-related 32-bit Timer)**

| Item | Specification |
|---|---|
| Number of channels | 8 channels (2 circuit blocks consisting of 4 channels each, 8 channels in total) |
| Input clock | Divided-by-2 frequency of the internal peripheral operating clock (e.g., 10.0 MHz when using 20 MHz internal peripheral operating clock) or clock bus 1 input |
| Counter | 32-bit up-counter × 2 |
| Measure register | 32-bit measure register × 8 |
| Timer startup | Starts counting after exiting reset |

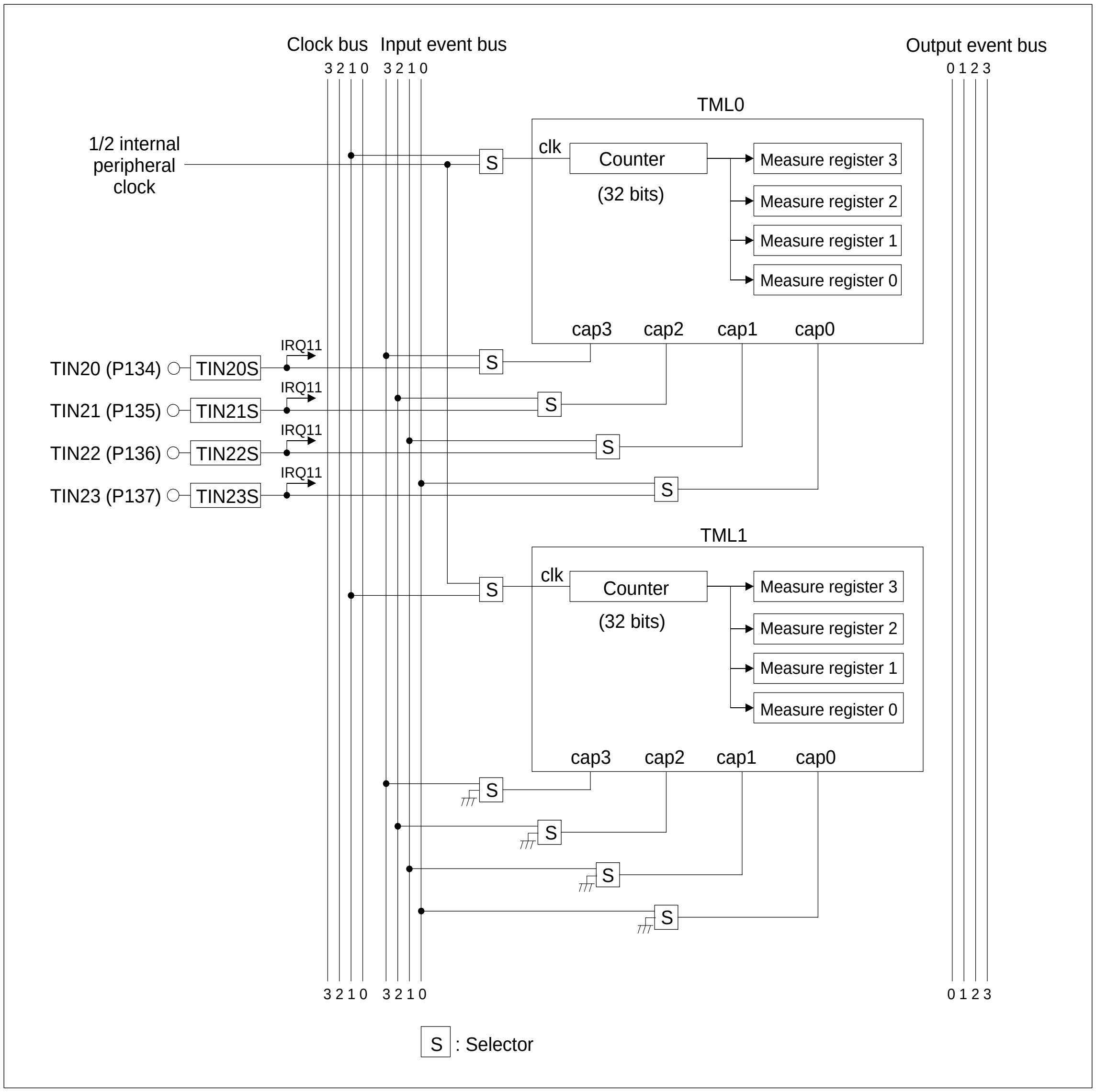


Figure 10.6.1 Block Diagram of TML (Input-related 32-bit Timer)

## 10.6.2 Outline of TML Operation

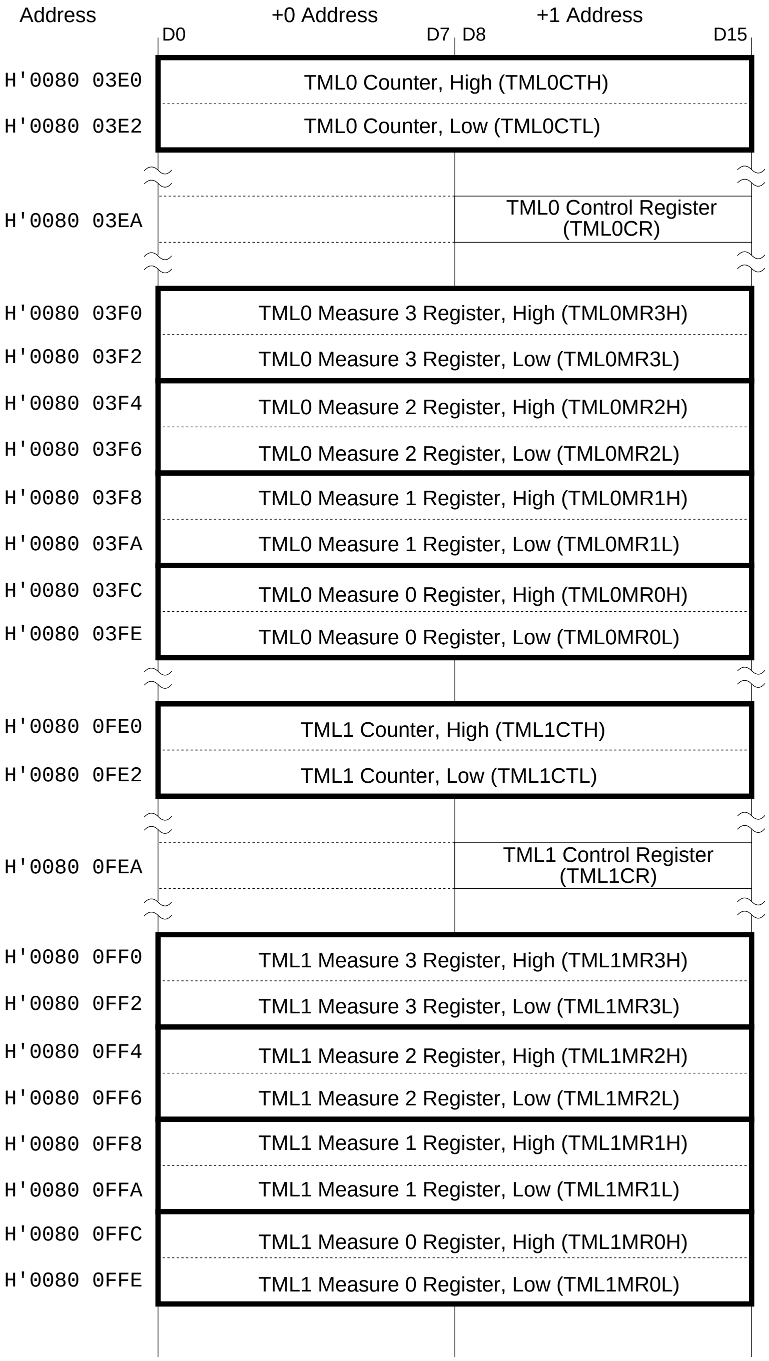
In TML, the counter starts counting upon deassertion of the reset input signal. The counter is a 32-bit up-counter, where when a measure event signal is entered from an external device, the counter value at that point in time is stored in each 32-bit measure register.

When the reset input signal is deasserted, the counter starts operating with a divided-by-2 frequency of the internal peripheral clock, and cannot be stopped once it has started. The counter is idle only when the device remains reset.

TIN interrupts can be generated by entering an external measurement signal (for TML0 only; no TIN interrupts available for TML1). However, the TML does not have counter overflow interrupts.

### 10.6.3 TML Related Register Map

The diagram below shows a TML related register map.



| Address | +0 Address (D0–D7) | +1 Address (D8–D15) |
|---|---|---|
| H'0080 03E0 | TML0 Counter, High (TML0CTH) | |
| H'0080 03E2 | TML0 Counter, Low (TML0CTL) | |
| H'0080 03EA | | TML0 Control Register (TML0CR) |
| H'0080 03F0 | TML0 Measure 3 Register, High (TML0MR3H) | |
| H'0080 03F2 | TML0 Measure 3 Register, Low (TML0MR3L) | |
| H'0080 03F4 | TML0 Measure 2 Register, High (TML0MR2H) | |
| H'0080 03F6 | TML0 Measure 2 Register, Low (TML0MR2L) | |
| H'0080 03F8 | TML0 Measure 1 Register, High (TML0MR1H) | |
| H'0080 03FA | TML0 Measure 1 Register, Low (TML0MR1L) | |
| H'0080 03FC | TML0 Measure 0 Register, High (TML0MR0H) | |
| H'0080 03FE | TML0 Measure 0 Register, Low (TML0MR0L) | |
| H'0080 0FE0 | TML1 Counter, High (TML1CTH) | |
| H'0080 0FE2 | TML1 Counter, Low (TML1CTL) | |
| H'0080 0FEA | | TML1 Control Register (TML1CR) |
| H'0080 0FF0 | TML1 Measure 3 Register, High (TML1MR3H) | |
| H'0080 0FF2 | TML1 Measure 3 Register, Low (TML1MR3L) | |
| H'0080 0FF4 | TML1 Measure 2 Register, High (TML1MR2H) | |
| H'0080 0FF6 | TML1 Measure 2 Register, Low (TML1MR2L) | |
| H'0080 0FF8 | TML1 Measure 1 Register, High (TML1MR1H) | |
| H'0080 0FFA | TML1 Measure 1 Register, Low (TML1MR1L) | |
| H'0080 0FFC | TML1 Measure 0 Register, High (TML1MR0H) | |
| H'0080 0FFE | TML1 Measure 0 Register, Low (TML1MR0L) | |

Blank addresses are reserved.

Note: • The registers enclosed in thick frames must always be accessed in words.

**Figure 10.6.2 TML Related Register Map**

### 10.6.4 TML Control Registers

■ **TML0 Control Register (TML0CR)** <Address: H'0080 03EB>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TML0SS0 | TML0SS1 | TML0SS2 | TML0SS3 | | | | TML0CKS |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | TML0SS0<br>(TML0 measure 0 source selection) | 0: External input TIN23<br>1: Input event bus 0 | ○ | ○ |
| 9 | TML0SS1<br>(TML0 measure 1 source selection) | 0: External input TIN22<br>1: Input event bus 1 | ○ | ○ |
| 10 | TML0SS2<br>(TML0 measure 2 source selection) | 0: External input TIN21<br>1: Input event bus 2 | ○ | ○ |
| 11 | TML0SS3<br>(TML0 measure 3 source selection) | 0: External input TIN20<br>1: Input event bus 3 | ○ | ○ |
| 12-14 | No functions assigned | | 0 | – |
| 15 | TML0CKS<br>(TML0 clock source selection) | 0: 1/2 internal peripheral clock<br>1: Clock bus 1 | ○ | ○ |

The TML0 Control Register is used to select TML0 input event and the counter clock source.

Note: • The counter can be written normally only when the selected clock source is a 1/2 internal peripheral clock. When using any other clock source, you cannot write to the counter normally. Under this condition, do not write to the counter.

### ■ TML1 Control Register (TML1CR)

<Address: H'0080 0FEB>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TML1SS0 | TML1SS1 | TML1SS2 | TML1SS3 | | | | TML1CKS |

<When reset:H'00>

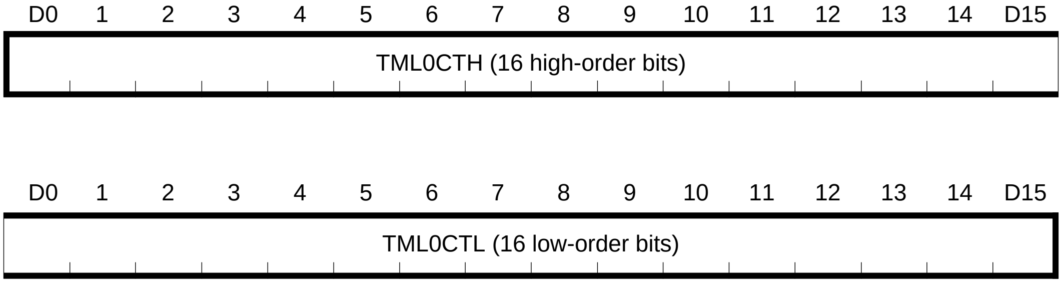
| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | TML1SS0<br>(TML1 measure 0 source selection) | 0: No selection<br>1: Input event bus 0 | ○ | ○ |
| 9 | TML1SS1<br>(TML1 measure 1 source selection) | 0: No selection<br>1: Input event bus 1 | ○ | ○ |
| 10 | TML1SS2<br>(TML1 measure 2 source selection) | 0: No selection<br>1: Input event bus 2 | ○ | ○ |
| 11 | TML1SS3<br>(TML1 measure 3 source selection) | 0: No selection<br>1: Input event bus 3 | ○ | ○ |
| 12-14 | No functions assigned | | 0 | – |
| 15 | TML1CKS<br>(TML1 clock source selection) | 0: 1/2 internal peripheral clock<br>1: Clock bus 1 | ○ | ○ |

The TML1 Control Register is used to select TML1 input event and the counter clock source.

Note: • The counter can be written normally only when the selected clock source is a 1/2 internal peripheral clock. When using any other clock source, you cannot write to the counter normally. Under this condition, do not write to the counter.

### 10.6.5 TML Counters

**■ TML0 Counter, High (TML0CTH)** <Address: H'0080 03E0>
**■ TML0 Counter, Low (TML0CTL)** <Address: H'0080 03E2>



<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TML0CTH | 32-bit counter value (16 high-order bits) | ○ | ○ |
|  | TML0CTL | 32-bit counter value (16 low-order bits) |  |  |

Note: • This register must always be accessed in words (32 bits) beginning with the address of TML0CTH.

The TML0 Counter is a 32-bit up-counter, which starts counting upon deassertion of the reset input signal. The TML0CTH register accommodates the 16 high-order bits, and the TML0CTL register accommodates the 16 low-order bits of the 32-bit counter.
The counter can be read duaring operation.

### ■ TML1 Counter, High (TML1CTH) <Address: H'0080 0FE0>
### ■ TML1 Counter, Low (TML1CTL) <Address: H'0080 0FE2>

<table>
<tr><th>D0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>D15</th></tr>
<tr><td colspan="16">TML1CTH (16 high-order bits)</td></tr>
</table>

<table>
<tr><th>D0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>D15</th></tr>
<tr><td colspan="16">TML1CTL (16 low-order bits)</td></tr>
</table>

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TML1CTH | 32-bit counter value (16 high-order bits) | ○ | ○ |
| | TML1CTL | 32-bit counter value (16 low-order bits) | | |

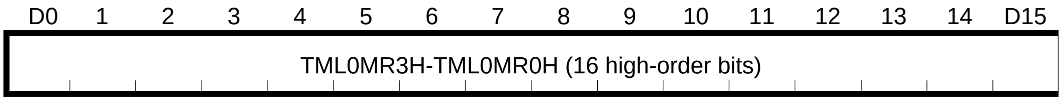
Note: • This register must always be accessed in words (32 bits) beginning with the address of TML1CTH.

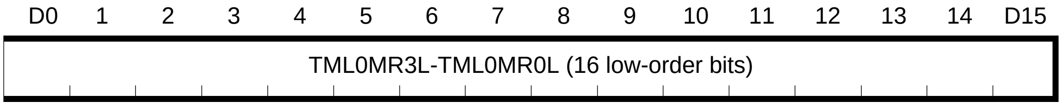
The TML1 Counter is a 32-bit up-counter, which starts counting upon deassertion of the reset input signal. The TML1CTH register accommodates the 16 high-order bits, and the TML1CTL register accommodates the 16 low-order bits of the 32-bit counter.

The counter can be read during operation.

### 10.6.6 TML Measure Registers

| | |
|---|---|
| ■ **TML0 Measure 3 Register (TML0MR3H)** | <Address: H'0080 03F0> |
| ■ **TML0 Measure 3 Register (TML0MR3L)** | <Address: H'0080 03F2> |
| ■ **TML0 Measure 2 Register (TML0MR2H)** | <Address: H'0080 03F4> |
| ■ **TML0 Measure 2 Register (TML0MR2L)** | <Address: H'0080 03F6> |
| ■ **TML0 Measure 1 Register (TML0MR1H)** | <Address: H'0080 03F8> |
| ■ **TML0 Measure 1 Register (TML0MR1L)** | <Address: H'0080 03FA> |
| ■ **TML0 Measure 0 Register (TML0MR0H)** | <Address: H'0080 03FC> |
| ■ **TML0 Measure 0 Register (TML0MR0L)** | <Address: H'0080 03FE> |





<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TML0MR3H-0H | 32-bit counter value (16 high-order bits) | ○ | – |
| | TML0MR3L-0L | 32-bit counter value (16 low-order bits) | | |

Notes: • These registers are a read-only register.
• These registers must always be accessed in words (32 bits) beginning with a word boundary.

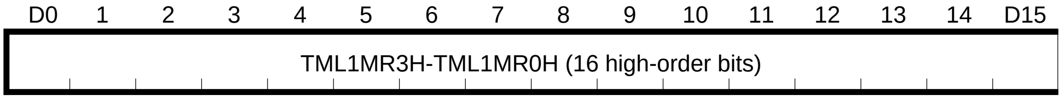
The TML0 Measure Registers are used to latch counter contents upon event input. The TML0 Measure Registers are configured with 32 bits, the TML0MR3H-0H accommodating the 16 high-order bits, and the TML0MR3L-0L accommodating the 16 low-order bits. The TML0 Measure Registers are a read-only register. These registers must always be accessed in words (32 bits) beginning with a word boundary.

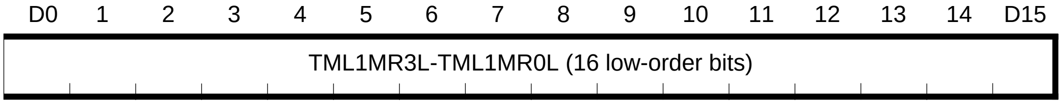
**■ TML1 Measure 3 Register (TML1MR3H)** <Address: H'0080 0FF0>
**■ TML1 Measure 3 Register (TML1MR3L)** <Address: H'0080 0FF2>

**■ TML1 Measure 2 Register (TML1MR2H)** <Address: H'0080 0FF4>
**■ TML1 Measure 2 Register (TML1MR2L)** <Address: H'0080 0FF6>

**■ TML1 Measure 1 Register (TML1MR1H)** <Address: H'0080 0FF8>
**■ TML1 Measure 1 Register (TML1MR1L)** <Address: H'0080 0FFA>

**■ TML1 Measure 0 Register (TML1MR0H)** <Address: H'0080 0FFC>
**■ TML1 Measure 0 Register (TML1MR0L)** <Address: H'0080 0FFE>





<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | TML1MR3H-0H | 32-bit counter value (16 high-order bits) | ○ | – |
| | TML1MR3L-0L | 32-bit counter value (16 low-order bits) | | |

Notes: • These registers are a read-only register.
• These registers must always be accessed in words (32 bits) beginning with a word boundary.

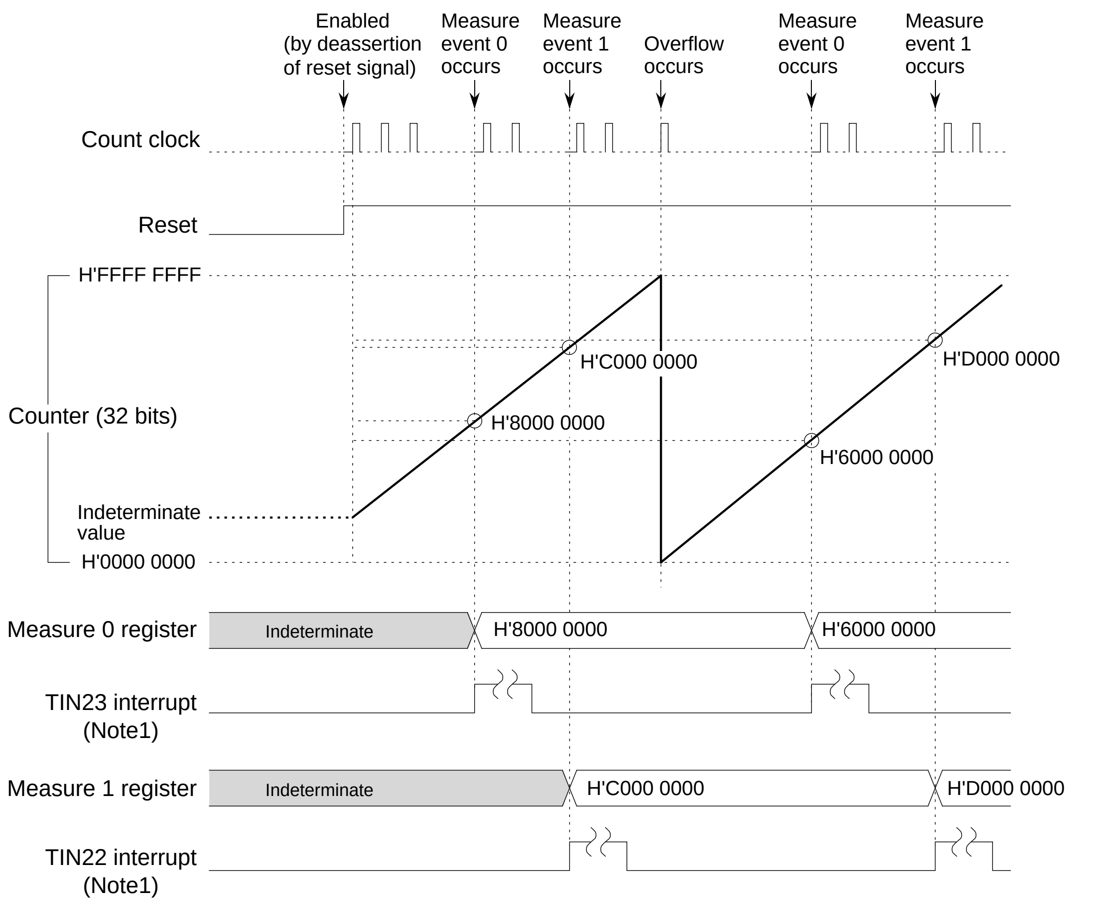
The TML1 Measure Registers are used to latch counter contents upon event input. The TML1 Measure Registers are configured with 32 bits, the TML1MR3H-0H accommodating the 16 high-order bits, and the TML1MR3L-0L accommodating the 16 low-order bits. The TML1 Measure Registers are a read-only register. These registers must always be accessed in words (32 bits) beginning with a word boundary.

### 10.6.7 Operation of TML Measure Input

#### (1) Outline of TML measure input

In TML measure input, the counter starts counting up clock pulses upon deassertion of the reset input signal. When event input is entered to measure registers 0-3, the counter value is latched into the measure registers.

A TIN interrupt can be generated by entering an external measure signal. (For TML0 only; No TIN interrupts are available for TML1.) However, no counter overflow interrupts are available.



Note1: TIN interrupts can be generated by entering an external measurement signal for TML0 only (No TIN interrupts available for TML1).

Note: • This diagram does not show detail timing information.

**Figure 10.6.3 Typical Operation in TML Measure Input**

### (2) Precautions to be observed when using TML measure input

The following describes precautions to be observed when using TML measure input.

- If measure event input and write to the counter occur simultaneously in the same clock period, the write value is set in the counter, whereas the up-count value (before being rewritten) is latched to the measure register.

- If the timer operates with any clock other than the 1/2 internal peripheral clock while clock bus 1 is selected for the count clock, the counter cannot be written normally. Therefore, when operating with any clock other than the 1/2 internal peripheral clock, do not write to the counter.

- If the timer operates with any clock other than the 1/2 internal peripheral clock while clock bus 1 is selected for the count clock, the captured value is one that leads the actual counter value by one clock period. However, during the 1/2 internal peripheral clock interval from the count clock, this problem does not occur and the counter value is captured at exact timing.

The diagram below shows the relationship between counter operation and the valid data that can be captured.

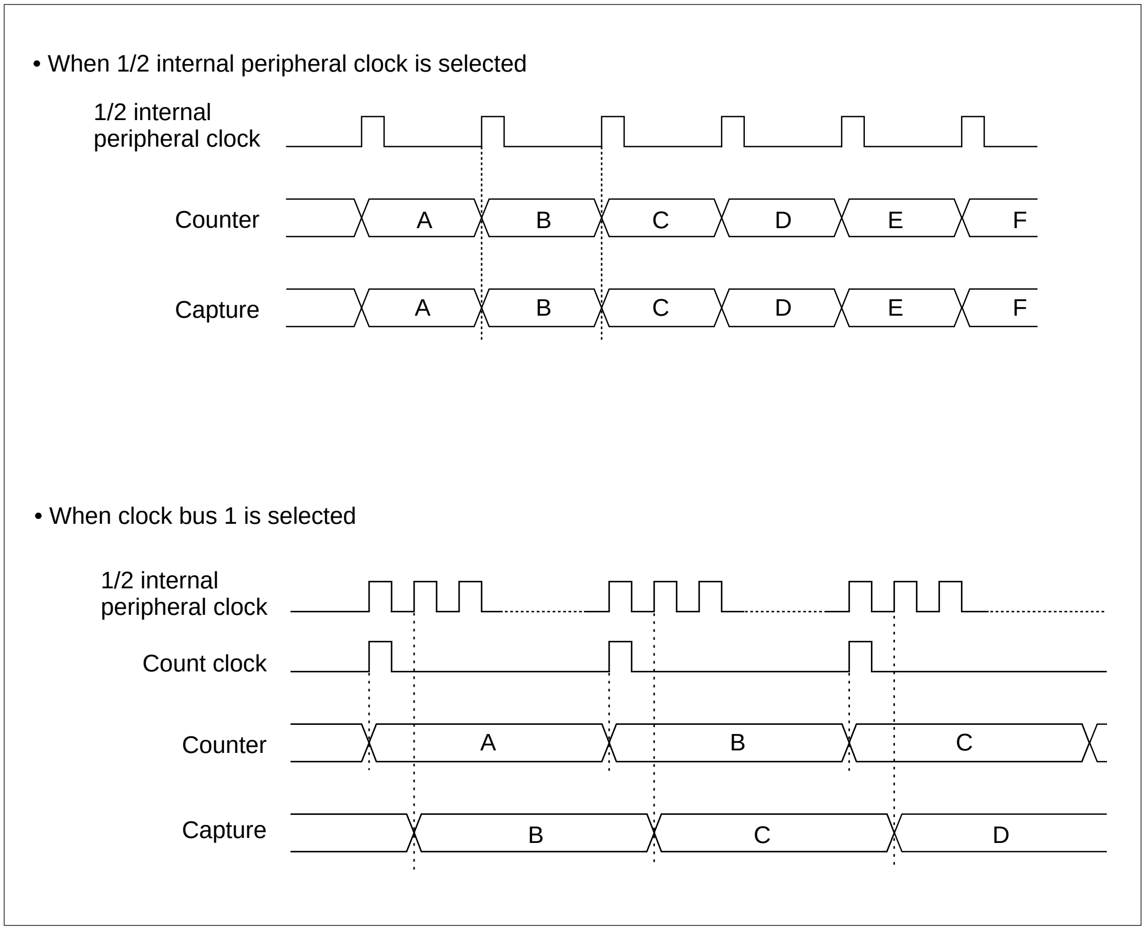

• When 1/2 internal peripheral clock is selected
1/2 internal peripheral clock
Counter
A B C D E F
Capture
A B C D E F
• When clock bus 1 is selected
1/2 internal peripheral clock
Count clock
Counter
A B C
Capture
B C D

**Figure 10.6.4 Mistimed Counter Value and Captured Value**

# CHAPTER 11
# A-D CONVERTER

11.1 Outline of A-D Converter

11.2 A-D Converter Related Registers

11.3 Functional Description of A-D Converter

11.4 Precautions on Using A-D Converter

## 11.1 Outline of A-D Converter

The 32171 contains a 10-bit resolution A-D converter based on successive approximation method.
A total of 16 analog input pins (channels) from AD0IN0 to AD0IN15 are available.
The A-D conversion results can be read out in either 8 bits or 10 bits.

For A-D conversion, there are following conversion modes and operation modes:

### (1) Conversion mode

- A-D conversion mode: Ordinary mode in which analog input voltages are converted into digital quantities.
- Comparator mode (Note 1): A mode in which analog input voltage is compared with a preset comparison voltage to only find the relative magnitude of two quantities. (Single mode only)

### (2) Operation mode

- Single mode: Analog input voltage in one channel is A-D converted once or comparated (Note 1) with a given quantity.
- Scan mode: Analog input voltages in multiple selected channels (4, 8, or 16 channels) are sequentially A-D converted.

### (3) Types of scan modes

- Single-shot scan mode: Scan operation is performed for one machine cycle.
- Continuous scan mode: Scan operation is performed repeatedly until stopped.

### (4) Special operation mode

- Forcible single mode execution during scan mode: Conversion is forcibly executed in single mode during scan operation.
- Scan mode start after single mode execution: Scan operation is started subsequently after executing conversion in single mode.
- Conversion restart: A-D conversion being executed in single or scan mode is restarted.

The A-D conversion and comparate rates can be selected between normal and double rate. An A-D conversion interrupt request or a DMA transfer request can be generated at completion of A-D conversion, comparate operation, single-shot scan operation, or one cycle of continuous scan operation.

Note 1: To discriminate between the comparison operation performed internally by the successive approximation-type A-D converter and the operation in comparator mode performed using the A-D converter as a comparator, the comparison operation in comparator mode in this manual is referred to as "comparate."

Table 11.1.1 outlines the A-D converter. Figure 11.1.1 shows a block diagram of the A-D converter.

**Table 11.1.1 Outline of A-D Converter**

| Item | Content | | |
|---|---|---|---|
| Analog input | 16 channels | | |
| A-D conversion method | Successive approximation method | | |
| Resolution | 10 bits (Conversion results can be read out in either 8 bits or 10 bits) | | |
| Absolute accuracy (Note1) (Conditions : Ta = -40 to 125°C, AVCC0=VREF0=5.12V) | Normal mode | ±2LSB | |
| | Double speed mode | ±2LSB | |
| Conversion mode | A-D conversion mode, comparator mode | | |
| Operation mode | Single mode, scan mode | | |
| Scan mode | Single-shot scan mode, continuous scan mode | | |
| Conversion start trigger | Software start | Started by setting A-D converter start bit to 1 | |
| | Hardware start | Starts A-D0 converter by MJT output event bus 3. (Note 2) | |
| Conversion rate f(BCLK): Internal peripheral clock operating frequency (Note 3) | During single mode (shortest time) | Normal rate | 299 × 1/f(BCLK) (Note 3) |
| | | Double rate | 173 × 1/f(BCLK) |
| | During comparator mode (shortest time) | Normal rate | 47 × 1/f(BCLK) |
| | | Double rate | 29 × 1/f(BCLK) |
| Interrupt request generation function | Generated at completion of A-D conversion, comparate operation, single-shot scan operation, or one cycle of continuous scan operation | | |
| DMA transfer request generation function | Generated at completion of A-D conversion, comparate operation, single-shot scan operation, or one cycle of continuous scan operation | | |

Note 1: The rated value (accuracy) is that of the microcomputer alone, premised on an assumption that power supply wiring on the board where the microcomputer is mounted is stable and unaffected by noise.

Note 2: Refer to Chapter 10, "Multijunction Timers."

Note 3: Note 3: f(BCLK) = 20 MHz when the input clock (XIN) = 10 MHz.

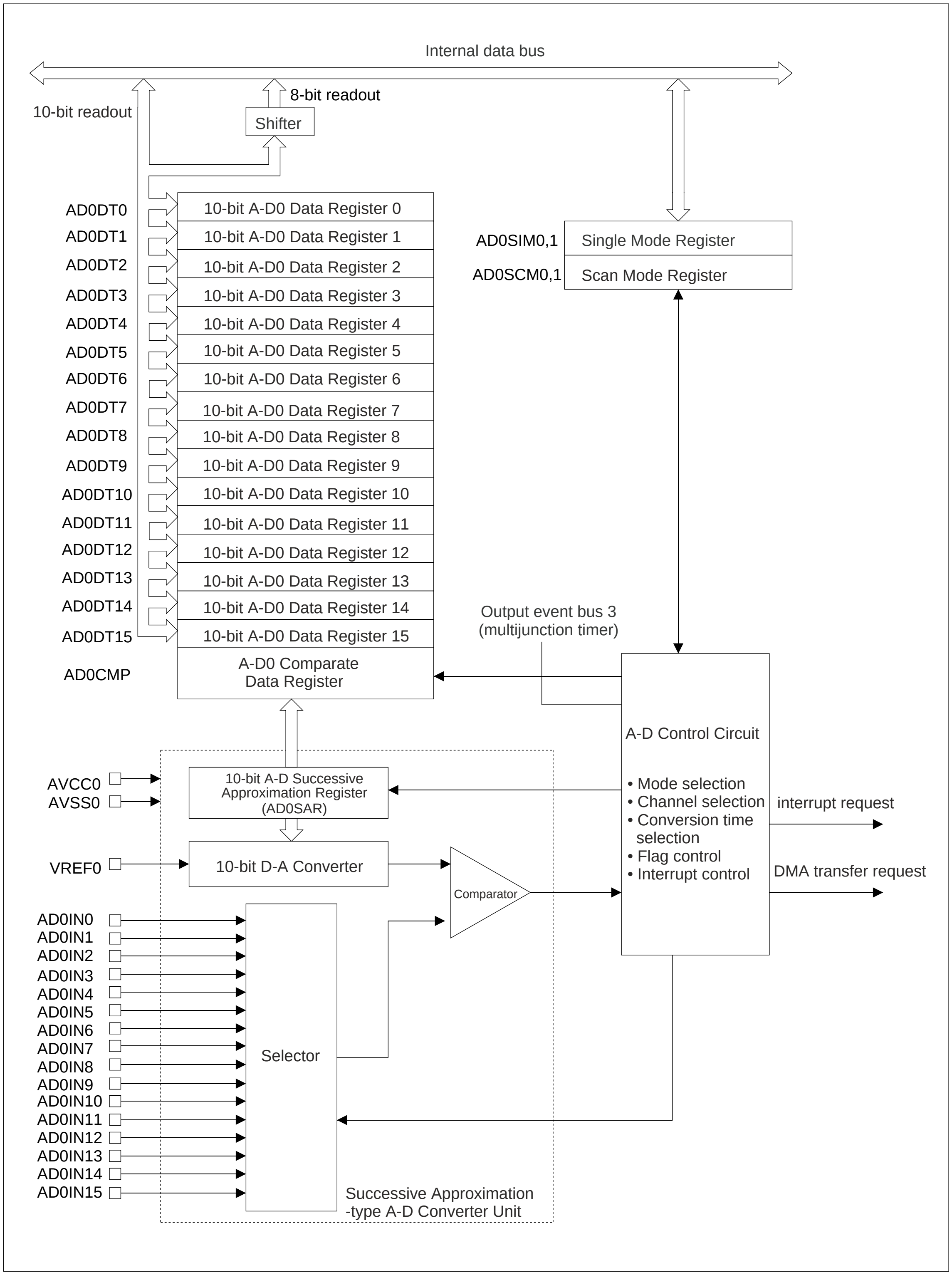


**Figure 11.1.1 Block Diagram of A-D0 Converter**

### 11.1.1 Conversion Modes

The A-D converter has two conversion modes: "A-D conversion mode" and "Comparator mode."

**(1) A-D conversion mode**

In A-D conversion mode, the analog input voltage in a specified channel is converted into digital quantity.

In single mode, A-D conversion is performed on a channel selected by the Single Mode Register 1 analog input pin select bit. In scan mode, A-D conversion is performed on channels selected by Scan Mode Register 1 according to settings of Scan Mode Register 0. The conversion result is stored in each channel's corresponding 10-bit A-D Data Register. Also, 8-bit A-D conversion results can be read from each 8-bit A-D Data Register.

An A-D conversion interrupt request or a DMA transfer request can be generated at completion of A-D conversion when in single mode, or when operating in scan mode, at completion of one cycle of scan loop.

**(2) Comparator mode**

In comparator mode, the analog input voltage in a specified channel is "comparated" (compared) with the Successive Approximation Register value, and the result (relative magnitude of two values) is returned to a flag.

The channel to be comparated is selected using the Single Mode Register 1 analog input pin select bit. The result of comparate operation is flagged (1 or 0) by setting the A-D Comparate Data Register bit that corresponds to the selected channel.

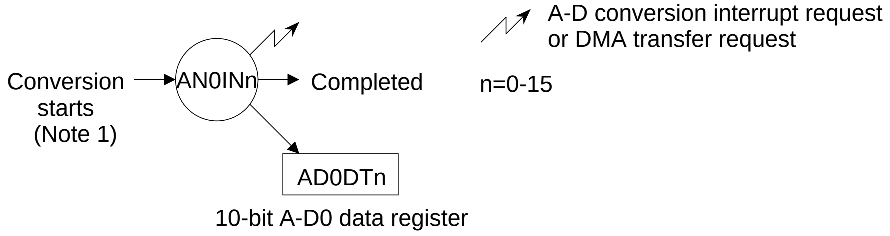
An A-D conversion interrupt request or a DMA transfer request can be generated at completion of comparate operation.

### 11.1.2 Operation Modes

The A-D converter operates in two modes: "Single mode" and "Scan mode." When comparator mode is selected as A-D conversion mode, only single mode can be used.

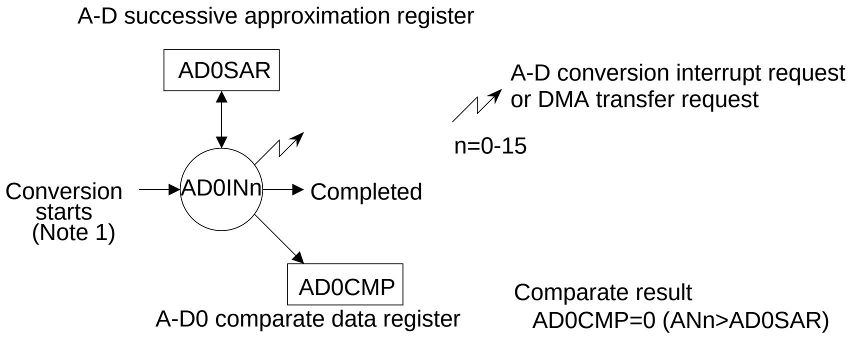
**(1) Single mode**

In single mode, the analog input voltage in one selected channel is A-D converted once or comparated with a given quantity. An A-D conversion interrupt request or a DMA transfer request can be generated at completion of A-D conversion.


A-D conversion interrupt request
or DMA transfer request
Conversion starts (Note 1)
AN0INn
Completed
n=0-15
AD0DTn
10-bit A-D0 data register

Note 1: A-D0 conversion start: Software trigger → Started by setting A-D0 conversion start bit to 1
Hardware trigger → Started by output event bus 3

**Figure 11.1.2 Operation in Single Mode (A-D Conversion)**


A-D successive approximation register
AD0SAR
A-D conversion interrupt request
or DMA transfer request
n=0-15
Conversion starts (Note 1)
AD0INn
Completed
AD0CMP
A-D0 comparate data register
Comparate result
AD0CMP=0 (ANn>AD0SAR)

Note 1: Comparate start: Started by writing a comparison value to the successive approximation register (AD0SAR)

**Figure 11.1.3 Operation in Single Mode (Comparate)**

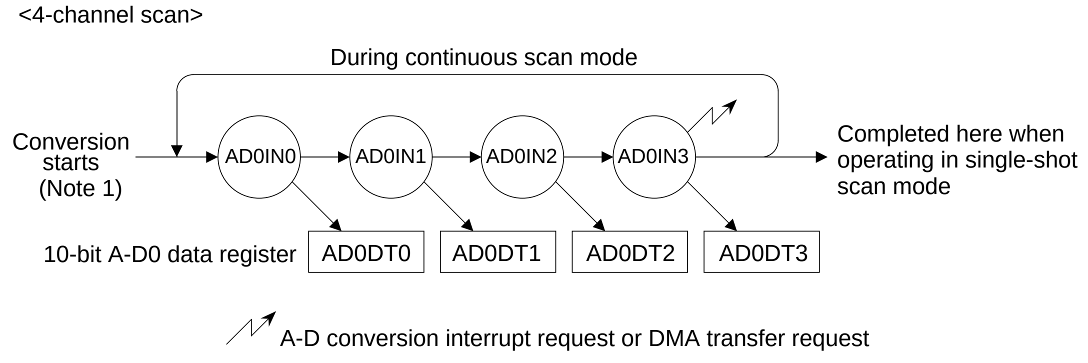
### (2) Scan mode

In scan mode, analog input voltages in multiple selected channels (4, 8, or 16 channels) are sequentially A-D converted.

There are two types of scan modes: "Single-shot scan mode" in which A-D conversion is completed by performing one cycle of scan operation, and "Continuous scan mode" in which scan operation is continued until halted by setting the Scan Mode Register A-D conversion stop bit to 1.

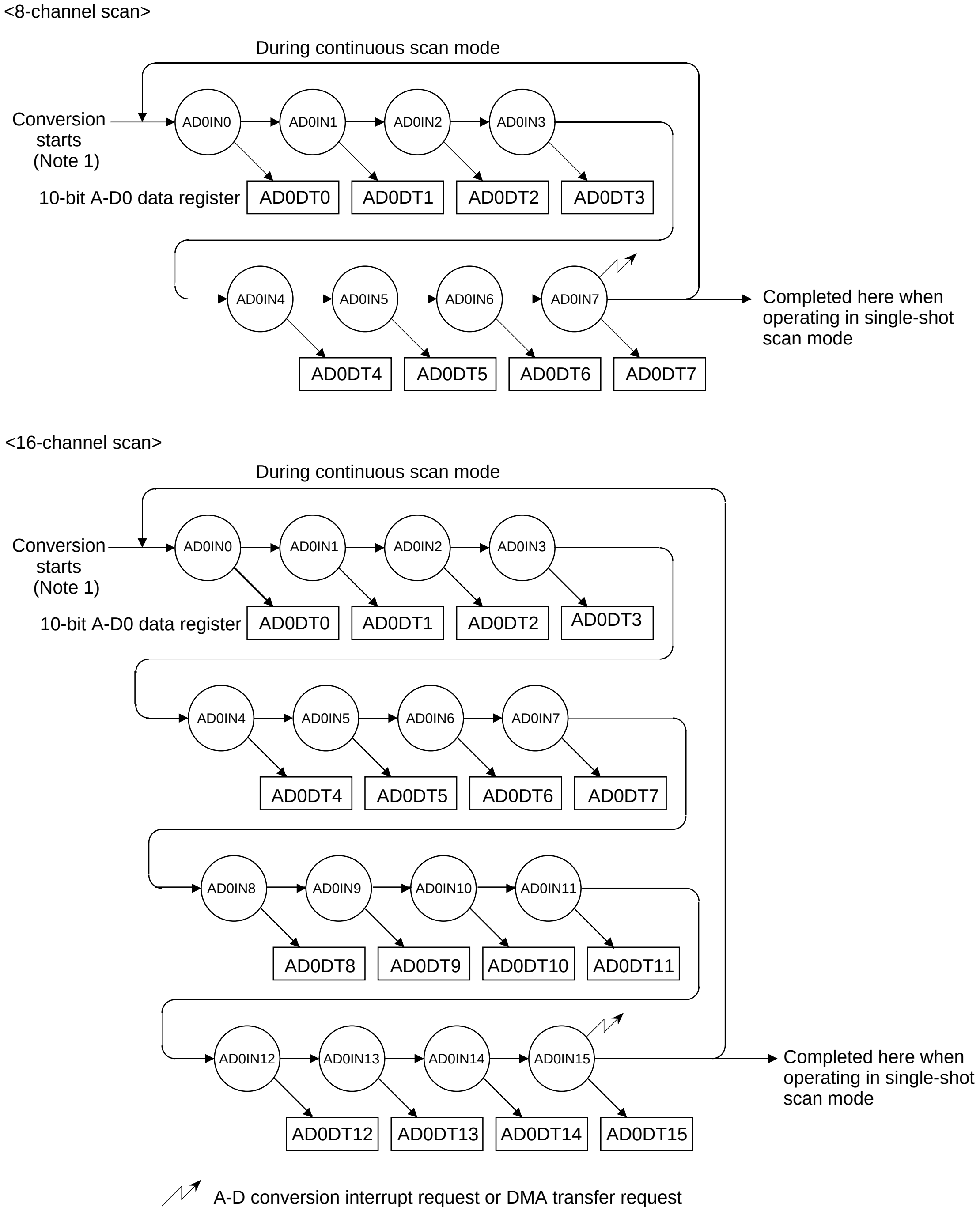
These types of scan modes are selected using Scan Mode Register 0. The channels to be scanned are selected using Scan Mode Register 1. The number of channels and the sequence to be scanned can be selected from three combinations available: 4, 8, or 16 channels. Channels AD0IN0 to AD0IN3 are used for 4-channel scan. Similarly, channels AD0IN0 to AD0IN7 and channels AD0IN0 to AD0IN15 are used for 8-channel scan and 16-channel scan, respectively.

An A-D conversion interrupt request or a DMA transfer request can be generated at completion of one cycle of scan operation.


<4-channel scan>
During continuous scan mode
Conversion starts (Note 1)
AD0IN0
AD0IN1
AD0IN2
AD0IN3
Completed here when operating in single-shot scan mode
10-bit A-D0 data register
AD0DT0
AD0DT1
AD0DT2
AD0DT3
A-D conversion interrupt request or DMA transfer request

Note 1: A-D0 conversion start: Software trigger → Started by setting A-D0 conversion start bit to 1
Hardware trigger → Started by output event bus 3

**Figure 11.1.4 Operation of A-D Conversion in Scan Mode (for 4-channel Scan)**



Note 1 : A-D0 conversion start: Software trigger → Started by setting A-D0 conversion start bit to 1
Hardware trigger → Started by output event bus 3

**Figure 11.1.5 Operation of A-D Conversion in Scan Mode (for 8-channel/16-channel Scan)**

### Table 11.1.2 Registers in Which Scan Mode A-D Conversion Results Are Stored

| Scan loop selection | Selected channels for single-shot scan | Selected channels for continue scan | A-D Conversion result storage Register |
|---|---|---|---|
| 4-channel scan | AD0IN0 | AD0IN0 | 10-bit A-D0 Data Register 0 |
| | AD0IN1 | AD0IN1 | 10-bit A-D0 Data Register 1 |
| | AD0IN2 | AD0IN2 | 10-bit A-D0 Data Register 2 |
| | AD0IN3 | AD0IN3 | 10-bit A-D0 Data Register 3 |
| | Completed | AD0IN0 | 10-bit A-D0 Data Register 0 |
| | | ⋮ (Repeated until forcibly halted) | ⋮ |
| 8-channel scan | AD0IN0 | AD0IN0 | 10-bit A-D0 Data Register 0 |
| | AD0IN1 | AD0IN1 | 10-bit A-D0 Data Register 1 |
| | AD0IN2 | AD0IN2 | 10-bit A-D0 Data Register 2 |
| | AD0IN3 | AD0IN3 | 10-bit A-D0 Data Register 3 |
| | AD0IN4 | AD0IN4 | 10-bit A-D0 Data Register 4 |
| | AD0IN5 | AD0IN5 | 10-bit A-D0 Data Register 5 |
| | AD0IN6 | AD0IN6 | 10-bit A-D0 Data Register 6 |
| | AD0IN7 | AD0IN7 | 10-bit A-D0 Data Register 7 |
| | Completed | AD0IN0 | 10-bit A-D0 Data Register 0 |
| | | ⋮ (Repeated until forcibly halted) | ⋮ |
| 16-channel scan | AD0IN0 | AD0IN0 | 10-bit A-D0 Data Register 0 |
| | AD0IN1 | AD0IN1 | 10-bit A-D0 Data Register 1 |
| | AD0IN2 | AD0IN2 | 10-bit A-D0 Data Register 2 |
| | AD0IN3 | AD0IN3 | 10-bit A-D0 Data Register 3 |
| | AD0IN4 | AD0IN4 | 10-bit A-D0 Data Register 4 |
| | AD0IN5 | AD0IN5 | 10-bit A-D0 Data Register 5 |
| | AD0IN6 | AD0IN6 | 10-bit A-D0 Data Register 6 |
| | AD0IN7 | AD0IN7 | 10-bit A-D0 Data Register 7 |
| | AD0IN8 | AD0IN8 | 10-bit A-D0 Data Register 8 |
| | AD0IN9 | AD0IN9 | 10-bit A-D0 Data Register 9 |
| | AD0IN10 | AD0IN10 | 10-bit A-D0 Data Register 10 |
| | AD0IN11 | AD0IN11 | 10-bit A-D0 Data Register 11 |
| | AD0IN12 | AD0IN12 | 10-bit A-D0 Data Register 12 |
| | AD0IN13 | AD0IN13 | 10-bit A-D0 Data Register 13 |
| | AD0IN14 | AD0IN14 | 10-bit A-D0 Data Register 14 |
| | AD0IN15 | AD0IN15 | 10-bit A-D0 Data Register 15 |
| | Completed | AD0IN0 | 10-bit A-D Data Register 0 |
| | | ⋮ (Repeated until forcibly halted) | ⋮ |

### 11.1.3 Special Operation Modes

#### (1) Forcible single mode execution during scan mode

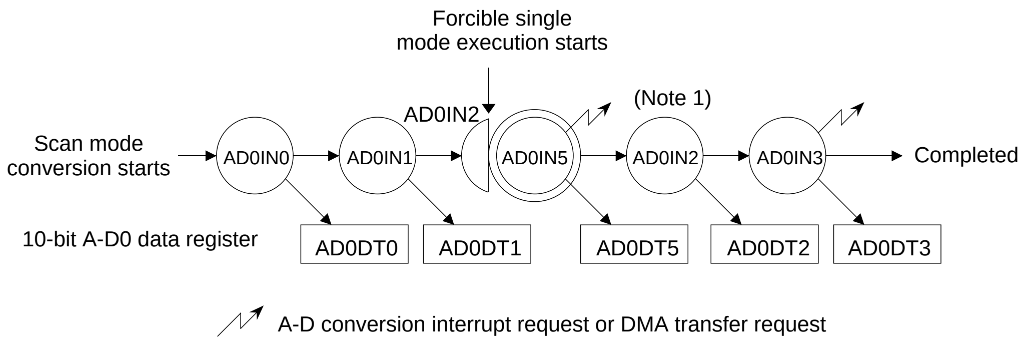
This special operation mode forcibly executes single mode conversion (A-D conversion or comparate) in a specified channel during scan mode operation. For A-D conversion mode, the conversion result is stored in the 10-bit A-D Data Register corresponding to the specified channel. For comparate mode, the conversion result is stored in the 10-bit A-D Comparate Data Register. When the A-D conversion or comparate operation in the specified channel is completed, scan mode A-D conversion is restarted from where it was canceled during scan operation.

To start single mode conversion during scan mode operation in software, select software trigger using the Single Mode Register 0’s A-D conversion start trigger select bit and for A-D conversion, set the said register’s A-D conversion start bit to 1. For comparate mode, write the value to be compared into the A-D Successive Approximation Register (AD0SAR) during scan mode operation.

To start single mode conversion during scan mode operation in hardware, select hardware trigger using Single Mode Register 0’s A-D conversion start trigger select bit and enter the hardware trigger (output event bus 3) specified by the said register.

An A-D conversion interrupt request or a DMA transfer request can be generated at completion of conversion in the specified channel, or at completion of one cycle of scan operation.

<To perform single mode conversion on AD0IN5 during AD0IN2 conversion in 4-channel single-shot scan mode>



Note 1: The canceled convert operation in channel 2 is reexecuted from the beginning.

**Figure 11.1.6 Forcible Single Mode Execution during Scan Mode**

### (2) Scan mode start after single mode execution

This special operation mode starts scan operation subsequently after executing conversion in single mode (A-D conversion or comparate).

To start this mode in software, choose a software trigger using the Scan Mode Register 0 A-D conversion start trigger select bit. Then set the said register's A-D conversion start bit to 1 during single mode conversion operation.

To start in hardware, select hardware trigger using the Scan Mode Register 0’s A-D conversion start trigger select bit and enter the hardware trigger (output event bus 3) specified by the said register while single mode conversion is in operation.

When a hardware trigger (output event bus 3) is entered after selecting hardware trigger with the A-D conversion start trigger select bits of both Single Mode Register 0 and Scan Mode Register 0, conversion is first performed in single mode and then after execution of it, conversion is performed in scan mode.

An A-D conversion interrupt request or a DMA transfer request can be generated at completion of single mode conversion in the specified channel, or at completion of one cycle of scan operation.

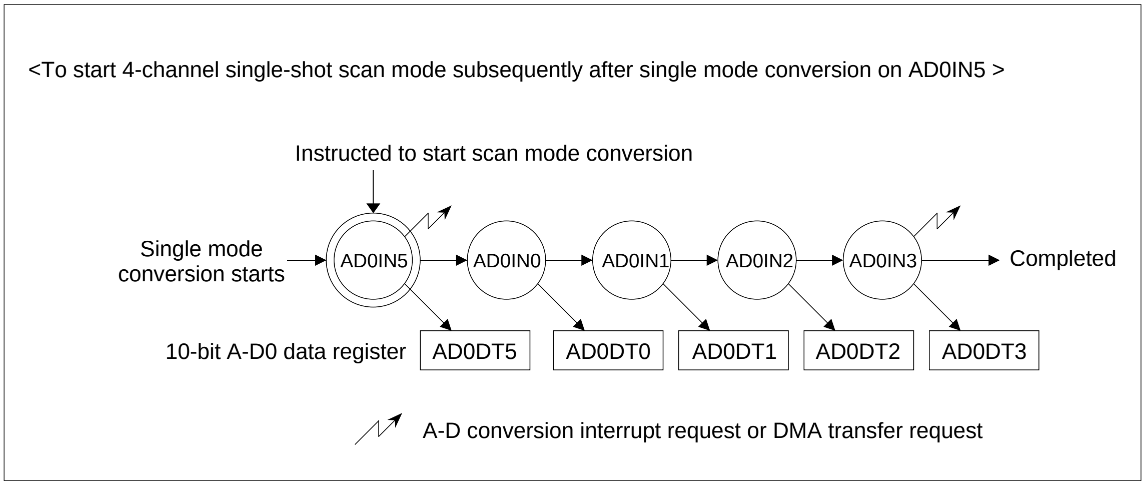

<To start 4-channel single-shot scan mode subsequently after single mode conversion on AD0IN5 >
Instructed to start scan mode conversion
Single mode conversion starts
AD0IN5
AD0IN0
AD0IN1
AD0IN2
AD0IN3
Completed
10-bit A-D0 data register
AD0DT5
AD0DT0
AD0DT1
AD0DT2
AD0DT3
A-D conversion interrupt request or DMA transfer request

**Figure 11.1.7 Scan Mode Start after Single Mode Execution**

### (3) Conversion restart

This special operation mode stops operation being executed in single mode or scan mode and reexecutes the operation from the beginning.

In the case of single mode, the operation being executed is redone by setting Single Mode Register 0's A-D conversion start bit to 1 again during A-D conversion or comparate operation or by entering a hardware trigger (output event bus 3).

For scan mode, the channel being converted is canceled and A-D conversion is restarted from channel 0 by setting Scan Mode Register 0's A-D conversion start bit to 1 again during scan operation or by entering a hardware trigger (output event bus 3).

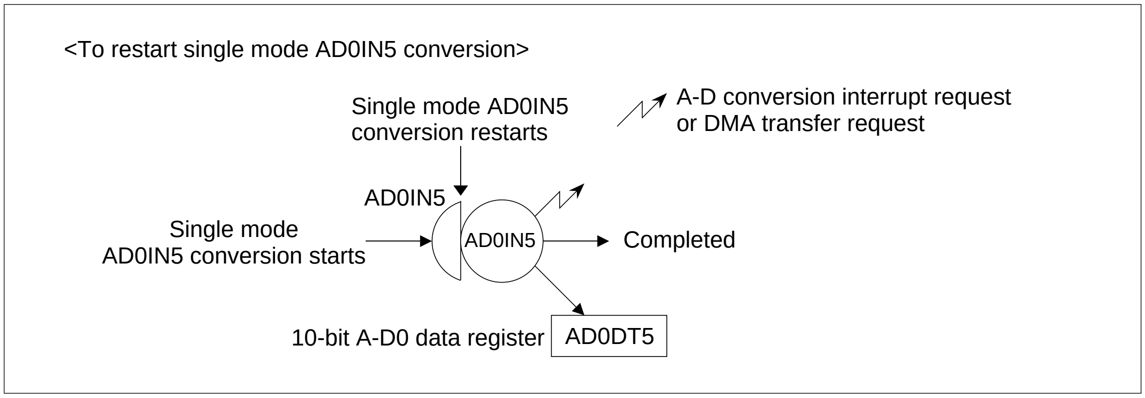


**Figure 11.1.8 Restarting Conversion during Single Mode Operation**

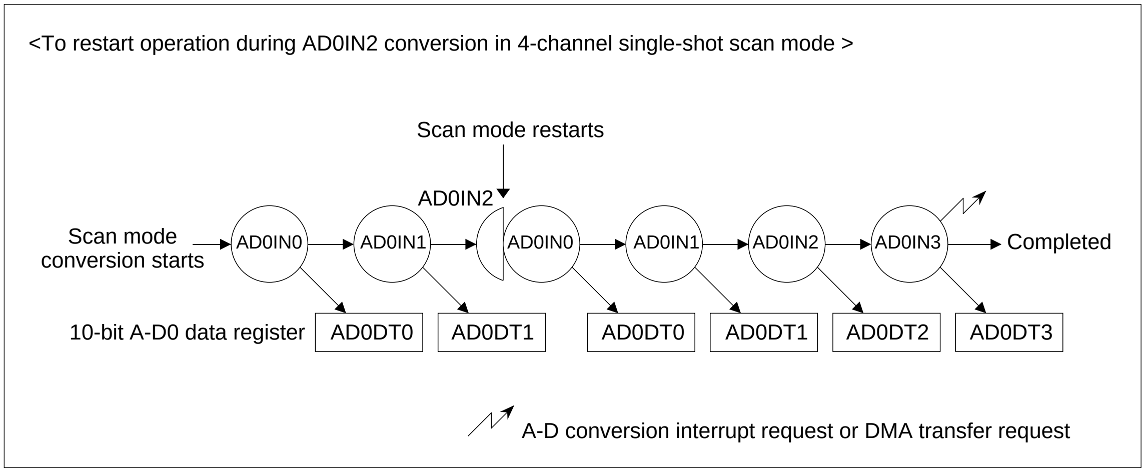

<To restart operation during AD0IN2 conversion in 4-channel single-shot scan mode >
Scan mode restarts
AD0IN2
Scan mode
conversion starts
AD0IN0
AD0IN1
AD0IN0
AD0IN1
AD0IN2
AD0IN3
Completed
10-bit A-D0 data register
AD0DT0
AD0DT1
AD0DT0
AD0DT1
AD0DT2
AD0DT3
A-D conversion interrupt request or DMA transfer request

**Figure 11.1.9 Restarting Conversion during Scan Operation**

### 11.1.4 A-D Converter Interrupt and DMA Transfer Requests

The A-D converter can generate an A-D conversion interrupt request or DMA transfer request at completion of A-D conversion, comparate operation, or one-shot scan or when each cycle of continuous scan mode is completed.

To select between A-D conversion interrupt or DMA transfer requests to generate, use Single Mode Register 0 and Scan Mode Register 0.

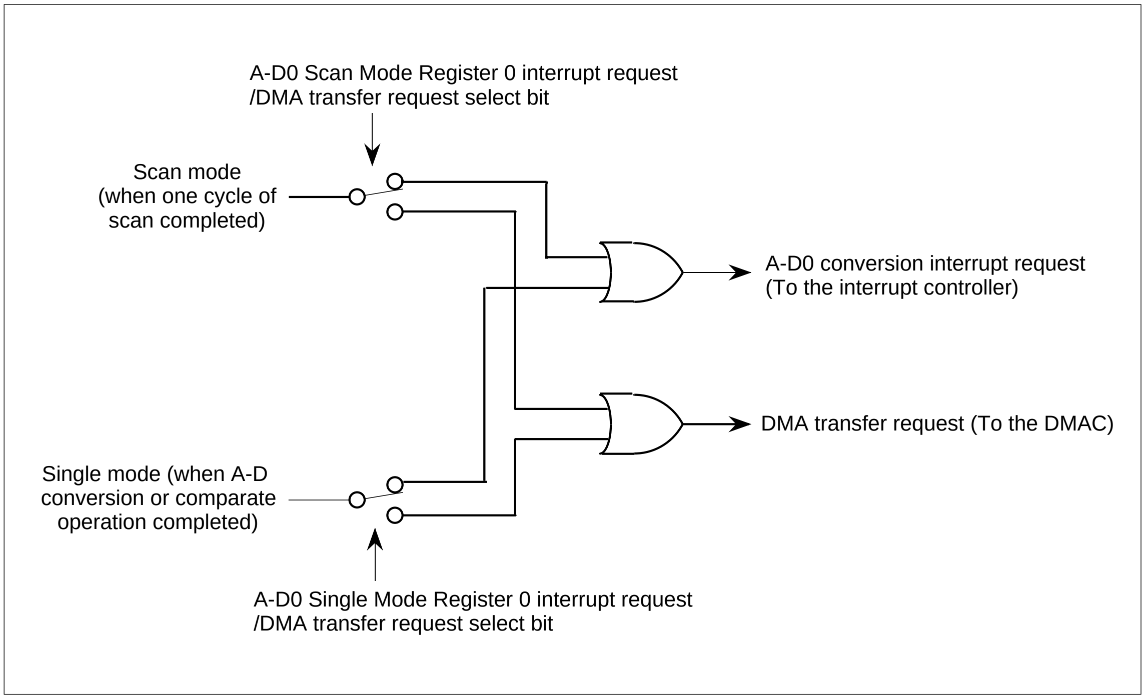

A-D0 Scan Mode Register 0 interrupt request
/DMA transfer request select bit
Scan mode
(when one cycle of
scan completed)
A-D0 conversion interrupt request
(To the interrupt controller)
DMA transfer request (To the DMAC)
Single mode (when A-D
conversion or comparate
operation completed)
A-D0 Single Mode Register 0 interrupt request
/DMA transfer request select bit

**Figure 11.1.10 Selecting between Interrupt Request and DMA Transfer Request**

## 11.2 A-D Converter Related Registers

The diagrams below show an A-D converter related register map.

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 0080 | A-D0 Single Mode Register 0<br>(AD0SIM0) | A-D0 Single Mode Register 1<br>(AD0SIM1) |
| H'0080 0082 | | |
| H'0080 0084 | A-D0 Scan Mode Register 0<br>(AD0SCM0) | A-D0 Scan Mode Register 1<br>(AD0SCM1) |
| H'0080 0086 | | |
| H'0080 0088 | A-D0 Successive Approximation Register (AD0SAR) | |
| H'0080 008A | | |
| H'0080 008C | A-D0 Comparate Data Register (AD0CMP) | |
| H'0080 0090 | 10-bit A-D0 Data Register 0 (AD0DT0) | |
| H'0080 0092 | 10-bit A-D0 Data Register 1 (AD0DT1) | |
| H'0080 0094 | 10-bit A-D0 Data Register 2 (AD0DT2) | |
| H'0080 0096 | 10-bit A-D0 Data Register 3 (AD0DT3) | |
| H'0080 0098 | 10-bit A-D0 Data Register 4 (AD0DT4) | |
| H'0080 009A | 10-bit A-D0 Data Register 5 (AD0DT5) | |
| H'0080 009C | 10-bit A-D0 Data Register 6 (AD0DT6) | |
| H'0080 009E | 10-bit A-D0 Data Register 7 (AD0DT7) | |
| H'0080 00A0 | 10-bit A-D0 Data Register 8 (AD0DT8) | |
| H'0080 00A2 | 10-bit A-D0 Data Register 9 (AD0DT9) | |
| H'0080 00A4 | 10-bit A-D0 Data Register 10 (AD0DT10) | |
| H'0080 00A6 | 10-bit A-D0 Data Register 11 (AD0DT11) | |
| H'0080 00A8 | 10-bit A-D0 Data Register 12 (AD0DT12) | |
| H'0080 00AA | 10-bit A-D0 Data Register 13 (AD0DT13) | |
| H'0080 00AC | 10-bit A-D0 Data Register 14 (AD0DT14) | |
| H'0080 00AE | 10-bit A-D0 Data Register 15 (AD0DT15) | |

Blank addresses are reserved.
Note: • The registers enclosed in thick frames must always be accessed in halfwords.

**Figure 11.2.1 A-D Converter Related Register Map (1/2)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 00D0 | | 8-bit A-D0 Data Register 0<br>(AD08DT0) |
| H'0080 00D2 | | 8-bit A-D0 Data Register 1<br>(AD08DT1) |
| H'0080 00D4 | | 8-bit A-D0 Data Register 2<br>(AD08DT2) |
| H'0080 00D6 | | 8-bit A-D0 Data Register 3<br>(AD08DT3) |
| H'0080 00D8 | | 8-bit A-D0 Data Register 4<br>(AD08DT4) |
| H'0080 00DA | | 8-bit A-D0 Data Register 5<br>(AD08DT5) |
| H'0080 00DC | | 8-bit A-D0 Data Register 6<br>(AD08DT6) |
| H'0080 00DE | | 8-bit A-D0 Data Register 7<br>(AD08DT7) |
| H'0080 00E0 | | 8-bit A-D0 Data Register 8<br>(AD08DT8) |
| H'0080 00E2 | | 8-bit A-D0 Data Register 9<br>(AD08DT9) |
| H'0080 00E4 | | 8-bit A-D0 Data Register 10<br>(AD08DT10) |
| H'0080 00E6 | | 8-bit A-D0 Data Register 11<br>(AD08DT11) |
| H'0080 00E8 | | 8-bit A-D0 Data Register 12<br>(AD08DT12) |
| H'0080 00EA | | 8-bit A-D0 Data Register 13<br>(AD08DT13) |
| H'0080 00EC | | 8-bit A-D0 Data Register 14<br>(AD08DT14) |
| H'0080 00EE | | 8-bit A-D0 Data Register 15<br>(AD08DT15) |

Blank addresses are reserved.

**Figure 11.2.2 A-D Converter Related Register Map (2/2)**

### 11.2.1 A-D Single Mode Register 0

■ **A-D0 Single Mode Register 0 (AD0SIM0)** <Address: H'0080 0080>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | AD0STRG | AD0SSEL | AD0SREQ | AD0SCMP | AD0SSTP | AD0SSTT |

<When reset:H'04>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0,1 | No functions assigned | | 0 | – |
| 2 | AD0STRG<br>(A-D0 hardware trigger selection) | 0: Use inhibited<br>1: Output event bus 3 | ○ | ○ |
| 3 | AD0SSEL<br>(A-D0 conversion start trigger selection) | 0: Software trigger<br>1: Hardware trigger (Note 1) | ○ | ○ |
| 4 | AD0SREQ<br>(Interrupt request/DMA transfer request selection) | 0: A-D0 interrupt request<br>1: DMA transfer request | ○ | ○ |
| 5 | AD0SCMP<br>(A-D0 conversion/comparate completed) | 0: A-D0 conversion/comparate in progress<br>1: A-D0 conversion/comparate completed | ○ | – |
| 6 | AD0SSTP<br>(A-D0 conversion stop) | 0: Performs no operation<br>1: Stops A-D0 conversion | 0 | ○ |
| 7 | AD0SSTT<br>(A-D0 conversion start) | 0: Performs no operation<br>1: Starts A-D0 conversion | 0 | ○ |

Note 1: During comparator mode, hardware triggers, if any selected, are ignored and operation is started by a software trigger.

A-D0 Single Mode Register 0 is used to control operation of the A-D0 converter during single mode (including special mode "Forcible single mode execution during scan mode").

### (1) AD0STRG (A-D0 hardware trigger select) bit (D2)

When starting A-D conversion of the A-D0 converter in hardware, this bit specifies the conversion to be started by MJT output (output event bus 3). If software trigger is selected with the AD0SSEL (A-D0 conversion start trigger select) bit, the content of this bit is ignored.

### (2) AD0SSEL (A-D0 conversion start trigger select) bit (D3)

This bit selects whether to apply the A-D0 conversion start trigger in software or in hardware during single mode. When software trigger is selected, A-D conversion is started by setting the AD0SSTT (A-D0 conversion start) bit to 1. When hardware trigger is selected, set the AD0STRG (hardware trigger select) bit to 1 and specify conversion to be started by MJT output.

### (3) AD0SREQ (A-D0 interrupt request/DMA transfer request select) bit (D4)

This bit selects whether to generate an A-D0 conversion interrupt request or a DMA transfer request at completion of single mode (A-D conversion or comparate).

### (4) AD0SCMP (A-D0 conversion/comparate complete) bit (D5)

This is a read-only bit, and is 1 when reset. This bit is 0 when the A-D0 converter in single mode (A-D conversion or comparate) is operating and set to 1 when the operation is completed. It also is set to 1 when A-D conversion or comparate operation is forcibly terminated by setting the AD0SSTT (A-D0 conversion stop) bit to 1 during A-D conversion or comparate operation.

### (5) AD0SSTP (A-D0 conversion stop) bit (D6)

The A-D0 converter in single mode (A-D conversion or comparate) can be stopped by setting this bit to 1 while the converter is operating. Manipulation of this bit is ignored while the converter in single mode remains idle or is operating in scan mode. Operation is stopped immediately after writing to this bit and the content of the A-D0 Successive Approximation Register when read after being stopped shows an intermediate value that was in the middle of conversion. (No transfers to the A-D0 Data Register are performed.)

If the A-D0 conversion start and A-D0 conversion stop bits are set to 1 simultaneously, the A-D0 conversion stop bit is effective.

If this bit is set to 1 while single mode operation of special mode is under way (forcible execution of single mode during scan mode operation), only single mode conversion stops and scan mode operation restarts.

### (6) AD0SSTT (A-D0 conversion start) bit (D7)

A-D conversion of the A-D0 converter is started by setting this bit to 1 while software trigger has been selected with the AD0SSEL (A-D0 conversion start trigger select) bit.

If the A-D0 conversion start and A-D0 conversion stop bits are set to 1 simultaneously, the A-D0 conversion stop bit is effective.

When this bit is set to 1 during single mode conversion, special operation mode “Conversion restart” is assumed, so that conversion in single mode restarts.

When this bit is set to 1 during A-D conversion in scan mode, special operation mode “Forcible execution of single mode during scan mode operation” is assumed, so that the channel being converted in scan mode is canceled and single mode conversion is performed. When single mode conversion finishes, A-D conversion in scan mode restarts from the canceled channel.

### 11.2.2 A-D Single Mode Register 1

■ **A-D0 Single Mode Register 1 (AD0SIM1)** <Address: H'0080 0081>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| AD0SMSL | AD0SSPD | | | AN0SEL | | | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | AD0SMSL<br>(A-D0 conversion mode selection) | 0: A-D0 conversion mode<br>1: Comparator mode | ○ | ○ |
| 9 | AD0SSPD<br>(A-D0 conversion rate selection) | 0: Normal rate<br>1: Double rate | ○ | ○ |
| 10,11 | No functions assigned | | 0 | △ |
| 12-15 | AN0SEL<br>(Analog input pin selection) | 0000: Selects AD0IN0<br>0001: Selects AD0IN1<br>0010: Selects AD0IN2<br>0011: Selects AD0IN3<br>0100: Selects AD0IN4<br>0101: Selects AD0IN5<br>0110: Selects AD0IN6<br>0111: Selects AD0IN7<br>1000: Selects AD0IN8<br>1001: Selects AD0IN9<br>1010: Selects AD0IN10<br>1011: Selects AD0IN11<br>1100: Selects AD0IN12<br>1101: Selects AD0IN13<br>1110: Selects AD0IN14<br>1111: Selects AD0IN15 | ○ | ○ |

W=△ : Only writing a 0 is effective; when you write a 1, device operation cannot be guaranteed.

A-D0 Single Mode Register 1 is used to control operation of the A-D0 converter during single mode (including special mode "Forcible single mode execution during scan mode").

### (1) AD0SMSL (A-D0 conversion mode selection) bit (D8)

This bit selects A-D conversion mode for the A-D0 converter during single mode. Setting this bit to 0 selects A-D conversion mode, and setting this bit to 1 selects comparator mode.

### (2) AD0SSPD (A-D0 conversion rate selection) bit (D9)

This bit selects an A-D conversion rate for the A-D0 converter during single mode. Setting this bit to 0 selects a normal speed, and setting this bit to 1 selects a x2 speed.

### (3) AN0SEL (analog input pin selection) bits (D12-D15)

These bits select analog input pins for the A-D0 converter during single mode. It is the channels selected by these bits that are operated on for A-D conversion or comparate operation. When you read these bits, they show the values written to them.

### 11.2.3 A-D Scan Mode Register 0

**■ A-D0 Scan Mode Register 0 (AD0SCM0)** <Address: H'0080 0084>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  | AD0CMSL | AD0CTRG | AD0CSEL | AD0CREQ | AD0CCMP | AD0CSTP | AD0CSTT |

<When reset:H'04>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | No functions assigned |  | 0 | – |
| 1 | AD0CMSL<br>(A-D0 scan mode selection) | 0: Single-shot mode<br>1: Continuous mode | ○ | ○ |
| 2 | AD0CTRG<br>(A-D0 hardware trigger selection) | 0: Use inhibited<br>1: Output event bus 3 | ○ | ○ |
| 3 | AD0CSEL<br>(A-D0 conversion start trigger selection) | 0: Software trigger<br>1: Hardware trigger | ○ | ○ |
| 4 | AD0CREQ<br>(Interrupt request/DMA request selection) | 0: Requests A-D0 interrupt<br>1: Requests DMA transfer | ○ | ○ |
| 5 | AD0CCMP<br>(A-D0 conversion completed) | 0: A-D0 conversion in progress<br>1: A-D0 conversion completed | ○ | – |
| 6 | AD0CSTP<br>(A-D0 conversion stop) | 0: Performs no operation<br>1: Stops A-D0 conversion | 0 | ○ |
| 7 | AD0CSTT<br>(A-D0 conversion start) | 0: Performs no operation<br>1: Starts A-D0 conversion | 0 | ○ |

A-D0 Scan Mode Register 0 is used to control operation of the A-D0 converter during scan mode.

### (1) AD0CMSL (A-D0 scan mode select) bit (D1)

This bit selects the A-D0 converter scan mode between one-shot scan and continuous scan modes.

Setting this bit to 0 selects one-shot scan mode, so that A-D conversion of channels selected with the AN0SCAN (scan loop select) bit are performed sequentially. When A-D conversion on all selected channels is completed, the convert operation stops.

Setting this bit to 1 selects continuous scan mode, so that when operation in one-shot mode finishes, A-D conversion is performed from the first channel again. This is repeated until stopped by setting the AD0CSTP (A-D0 conversion stop) bit to 1.

### (2) AD0CTRG (A-D0 hardware trigger select) bit (D2)

When starting A-D conversion of the A-D0 converter in hardware, this bit specifies the conversion to be started by MJT output (output event bus 3). If software trigger is selected with the AD0CSEL (A-D conversion start trigger select) bit, the content of this bit is ignored.

### (3) AD0CSEL (A-D0 conversion start trigger select) bit (D3)

This bit selects whether to apply the A-D conversion start trigger in software or in hardware during scan mode of the A-D0 converter. When software trigger is selected, A-D conversion is started by setting the AD0CSTT (A-D0 conversion start) bit to 1. When hardware trigger is selected, set the AD0CTRG (hardware trigger select) bit to 1 and specify conversion to be started by MJT output.

### (4) AD0CREQ (A-D0 interrupt/DMA transfer request select) bit (D4)

This bit selects whether to generate an A-D0 conversion interrupt request or a DMA transfer request at completion of one cycle of scan mode operation.

### (5) AD0CCMP (A-D0 conversion complete) bit (D5)

This is a read-only bit, and is 1 when reset. This bit is 0 when scan mode conversion of the A-D0 converter is in progress and set to 1 when one-shot scan mode operation is completed or when continuous scan mode is stopped by setting the AD0CSTT (A-D0 conversion stop) bit to 1.

### (6) AD0CSTP (A-D0 conversion stop) bit (D6)

Scan mode operation of the A-D0 converter can be stopped by setting this bit to 1 while scan mode A-D conversion is under way. This bit is effective for only scan mode operation, and does not affect single mode operation when both single and scan modes of special operation mode are active.

Operation is stopped immediately after writing to this bit and A-D conversion on the channel which is in the middle of conversion is aborted, with no data transferred to the A-D Data Register.

If the A-D0 conversion start and A-D0 conversion stop bits are set to 1 simultaneously, the A-D0 conversion stop bit is effective.

### (7) AD0CSTT (A-D0 conversion start) bit (D7)

This bit is used to start scan mode operation of the A-D0 converter in software. Only when software trigger has been selected with the AD0CSEL (A-D0 conversion start trigger select) bit, A-D conversion can be started by setting this bit to 1.

If the A-D0 conversion start and A-D0 conversion stop bits are set to 1 simultaneously, the A-D0 conversion stop bit is effective.

When this bit is set to 1 during scan mode conversion again, special operation mode “Conversion restart” is assumed, so that scan operation restarts according to the contents set by Scan Mode Register 0 and Scan Mode Register 1.

When this bit is set to 1 during A-D conversion in single mode, special operation mode “Start scan mode after executing single mode” is assumed, so that scan mode operation starts on successive channels after single mode finishes.

### 11.2.4 A-D Scan Mode Register 1

#### ■ A-D0 Scan Mode Register 1 (AD0SCM1)

<Address: H'0080 0085>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  | AD0CSPD |  |  | AN0SCAN |  |  |  |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | No functions assigned |  | 0 | – |
| 9 | AD0CSPD<br>(A-D0 conversion rate selection) | 0: Normal rate<br>1: Double rate | ○ | ○ |
| 10,11 | No functions assigned |  | 0 | – |
| 12-15 | AN0SCAN<br>(A-D0 scan loop selection) | <For wirte><br>01XX: 4-channel scan<br>10XX: 8-channel scan<br>11XX: 16-channel scan<br>00XX: 16-channel scan<br><br><For read during conversion><br>0000: Converting AD0IN0<br>0001: Converting AD0IN1<br>0010: Converting AD0IN2<br>0011: Converting AD0IN3<br>0100: Converting AD0IN4<br>0101: Converting AD0IN5<br>0110: Converting AD0IN6<br>0111: Converting AD0IN7<br>1000: Converting AD0IN8<br>1001: Converting AD0IN9<br>1010: Converting AD0IN10<br>1011: Converting AD0IN11<br>1100: Converting AD0IN12<br>1101: Converting AD0IN13<br>1110: Converting AD0IN14<br>1111: Converting AD0IN15 | ○ | ○ |

A-D0 Scan Mode Register 1 is used to control operation of the A-D0 converter during scan mode.

### (1) AD0CSPD (A-D0 conversion rate selection) bit (D9)

This bit selects an A-D conversion rate for the A-D0 converter during scan mode. Setting this bit to 0 selects a normal speed, and setting this bit to 1 selects a x2 speed.

### (2) AN0SCAN (A-D0 scan loop selection) bits (D12-D15)

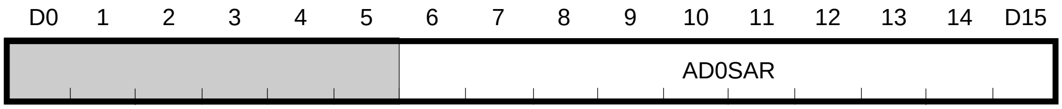
The AN0SCAN (A-D0 scan loop selection) bits set the channels to be scanned during scan mode of the A-D0 converter. In this case, writes to D14 and D15 have no effect.

The AN0SCAN (A-D0 scan loop selection) bits when read during scan operation show the status of the A-D0 converter, indicating the channel it is converting.

The value read from these bits during single mode are always "B'0000." If A-D conversion is halted by setting Scan Mode Register 0 AD0CSTP (A-D0 conversion stop) bit to 1 during scan mode execution, the bits when read at this time show the value of the channel in which the A-D conversion has been canceled. Also, if halted during single mode conversion in special operation mode "Forcible single mode execution during scan mode," the bits when read at this time show the value of the channel in which the A-D conversion has been canceled in the middle of scan.

### 11.2.5 A-D Successive Approximation Register

**■ A-D0 Successive Approximation Register (AD0SAR)** <Address: H'0080 0088>



<When reset:Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-5 | No functions assigned | | 0 | – |
| 6-15 | AD0SAR<br>(A-D0 successive approximation<br>value/comparison value) | • A-D successive approximation value<br>(A-D conversion mode)<br>• Comparison value (comparator mode) | ○ | ○ |

Note: • This register must always be accessed in halfwords.

The A-D0 Successive Approximation Register (AD0SAR), when in A-D conversion mode, is used to read out the conversion result of the A-D0 converter, and when in comparator mode, it is used to write a comparison value.

In A-D conversion mode, the successive approximation method is used to perform A-D conversion. With this method, the reference voltage VREF0 and analog input voltages are sequentially compared bitwise beginning with the high-order side, and the comparison result is set in the A-D0 Successive Approximation Register (AD0SAR) bits (D6-D15). After the A-D conversion is completed, the value of this register is transferred to the 10-bit A-D0 Data Register (AD0DTn) corresponding to the converted channel. When you read this register in the middle of A-D conversion, you see the result in the middle of conversion.

In comparator mode, write a comparison value (the value to be compared in comparate operation) to this register. Simultaneously with a write to this register, comparate operation with the analog input pin that has been set by Single Mode Register 1 starts. After comparate operation, the result is stored in the A-D0 Comparate Data Register (AD0CMP).

Use the calculation formula shown below to find the comparison value to be written to the A-D0 Successive Approximation Register (AD0SAR) during comparator mode.

$$\text{Comparison value} = \text{H'3FF} \times \frac{\text{Comparate comparison voltage [V]}}{\text{VREF0 input voltage [V]}}$$

### 11.2.6 A-D Comparate Data Register

#### ■ A-D0 Comparate Data Register (AD0CMP)

<Address: H'0080 008C>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AD0<br>CMP0 | AD0<br>CMP1 | AD0<br>CMP2 | AD0<br>CMP3 | AD0<br>CMP4 | AD0<br>CMP5 | AD0<br>CMP6 | AD0<br>CMP7 | AD0<br>CMP8 | AD0<br>CMP9 | AD0<br>CMP10 | AD0<br>CMP11 | AD0<br>CMP12 | AD0<br>CMP13 | AD0<br>CMP14 | AD0<br>CMP15 |

<When reset:Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | AD0CMP0-AD0CMP15 (Note 2) | 0: Analog input voltage > comparison voltage | ○ | – |
| | (A-D0 comparate result flag) | 1: Analog input voltage < comparison voltage | | |

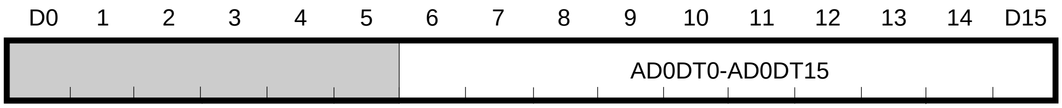
Notes : • This register must always be accessed in halfwords.
• During comparator mode, each bit corresponds to channels 0 through 15.

When comparator mode is selected by setting the A-D0 Single Mode Register 1 AD0SMSL (A-D0 conversion mode selection) bit, the selected analog input value is compared with the value written to the A-D0 Successive Approximation Register, with the result stored in the corresponding bit of this comparate data register.

The bit is 0 when the analog input voltage > comparison voltage, and is 1 when the analog input voltage < comparison voltage.

### 11.2.7 10-bit A-D Data Registers

| | |
|---|---|
| ■ **10-bit A-D0 Data Register 0 (AD0DT0)** | <Address: H'0080 0090> |
| ■ **10-bit A-D0 Data Register 1 (AD0DT1)** | <Address: H'0080 0092> |
| ■ **10-bit A-D0 Data Register 2 (AD0DT2)** | <Address: H'0080 0094> |
| ■ **10-bit A-D0 Data Register 3 (AD0DT3)** | <Address: H'0080 0096> |
| ■ **10-bit A-D0 Data Register 4 (AD0DT4)** | <Address: H'0080 0098> |
| ■ **10-bit A-D0 Data Register 5 (AD0DT5)** | <Address: H'0080 009A> |
| ■ **10-bit A-D0 Data Register 6 (AD0DT6)** | <Address: H'0080 009C> |
| ■ **10-bit A-D0 Data Register 7 (AD0DT7)** | <Address: H'0080 009E> |
| ■ **10-bit A-D0 Data Register 8 (AD0DT8)** | <Address: H'0080 00A0> |
| ■ **10-bit A-D0 Data Register 9 (AD0DT9)** | <Address: H'0080 00A2> |
| ■ **10-bit A-D0 Data Register 10 (AD0DT10)** | <Address: H'0080 00A4> |
| ■ **10-bit A-D0 Data Register 11 (AD0DT11)** | <Address: H'0080 00A6> |
| ■ **10-bit A-D0 Data Register 12 (AD0DT12)** | <Address: H'0080 00A8> |
| ■ **10-bit A-D0 Data Register 13 (AD0DT13)** | <Address: H'0080 00AA> |
| ■ **10-bit A-D0 Data Register 14 (AD0DT14)** | <Address: H'0080 00AC> |
| ■ **10-bit A-D0 Data Register 15 (AD0DT15)** | <Address: H'0080 00AE> |



<When reset:Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-5 | No functions assigned | | 0 | – |
| 6-15 | AD0DT0-AD0DT15<br>(10-bit A-D0 data) | A-D conversion result | ○ | – |

Note: • This register must always be accessed in halfwords.

In single mode of the A-D0 converter, the result of A-D conversion is stored in the 10-bit A-D0 Data Register for each corresponding channel. In single-shot and continuous scan modes, the content of the A-D0 Successive Approximation Register is transferred to the 10-bit A-D Data Register for the corresponding channel every time the A-D conversion in each channel is completed. Each 10-bit A-D Data Register retains the last conversion result until they receive the next conversion result transferred, allowing the content to be read out at any time.

### 11.2.8 8-bit A-D Data Registers

| | |
|---|---|
| ■ **8-bit A-D0 Data Register 0 (AD08DT0)** | <Address: H'0080 00D1> |
| ■ **8-bit A-D0 Data Register 1 (AD08DT1)** | <Address: H'0080 00D3> |
| ■ **8-bit A-D0 Data Register 2 (AD08DT2)** | <Address: H'0080 00D5> |
| ■ **8-bit A-D0 Data Register 3 (AD08DT3)** | <Address: H'0080 00D7> |
| ■ **8-bit A-D0 Data Register 4 (AD08DT4)** | <Address: H'0080 00D9> |
| ■ **8-bit A-D0 Data Register 5 (AD08DT5)** | <Address: H'0080 00DB> |
| ■ **8-bit A-D0 Data Register 6 (AD08DT6)** | <Address: H'0080 00DD> |
| ■ **8-bit A-D0 Data Register 7 (AD08DT7)** | <Address: H'0080 00DF> |
| ■ **8-bit A-D0 Data Register 8 (AD08DT8)** | <Address: H'0080 00E1> |
| ■ **8-bit A-D0 Data Register 9 (AD08DT9)** | <Address: H'0080 00E3> |
| ■ **8-bit A-D0 Data Register 10 (AD08DT10)** | <Address: H'0080 00E5> |
| ■ **8-bit A-D0 Data Register 11 (AD08DT11)** | <Address: H'0080 00E7> |
| ■ **8-bit A-D0 Data Register 12 (AD08DT12)** | <Address: H'0080 00E9> |
| ■ **8-bit A-D0 Data Register 13 (AD08DT13)** | <Address: H'0080 00EB> |
| ■ **8-bit A-D0 Data Register 14 (AD08DT14)** | <Address: H'0080 00ED> |
| ■ **8-bit A-D0 Data Register 15 (AD08DT15)** | <Address: H'0080 00EF> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| AD08DT0-AD08DT15 | | | | | | | |

<When reset:Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | AD08DT0-AD08DT15<br>(8-bit A-D0 data) | 8-bit A-D conversion result | ○ | – |

This A-D data register stores the 8-bit conversion data from the A-D0 converter.

In single mode of the A-D0 converter, the result of A-D conversion is stored in the 8-bit A-D0 Data Register for each corresponding channel. In single-shot and continuous scan modes, the content of the A-D0 Successive Approximation Register is transferred to the 8-bit A-D Data Register for the corresponding channel every time the A-D conversion in each channel is completed. Each 8-bit A-D Data Register retains the last conversion result until they receive the next conversion result transferred, allowing the content to be read out at any time.

## 11.3 Functional Description of A-D Converter

### 11.3.1 How to Find Along Input Voltages

The A-D converter uses a 10-bit successive approximation method, and finds the actual analog input voltage from the value (digital quantity) obtained through execution of A-D conversion by performing the following calculation.

$$\text{Analog input voltage [V]} = \frac{\text{A-D conversion result x VREF0 input voltage [V]}}{1024}$$

The A-D converter is a 10-bit converter, providing a resolution of 1,024 discrete voltage levels. Because the reference voltage for the A-D converter is the voltage applied to the VREF0 pin, make sure an exact and stable constant-voltage power supply is connected to VREF0. Also, make sure the analog circuit power supply and ground (AVCC0, AVSS0) are separated from those of the digital circuit, with sufficient noise prevention measures incorporated.
For details about the conversion accuracy, refer to Section 11.3.5, "Accuracy of A-D Conversion."

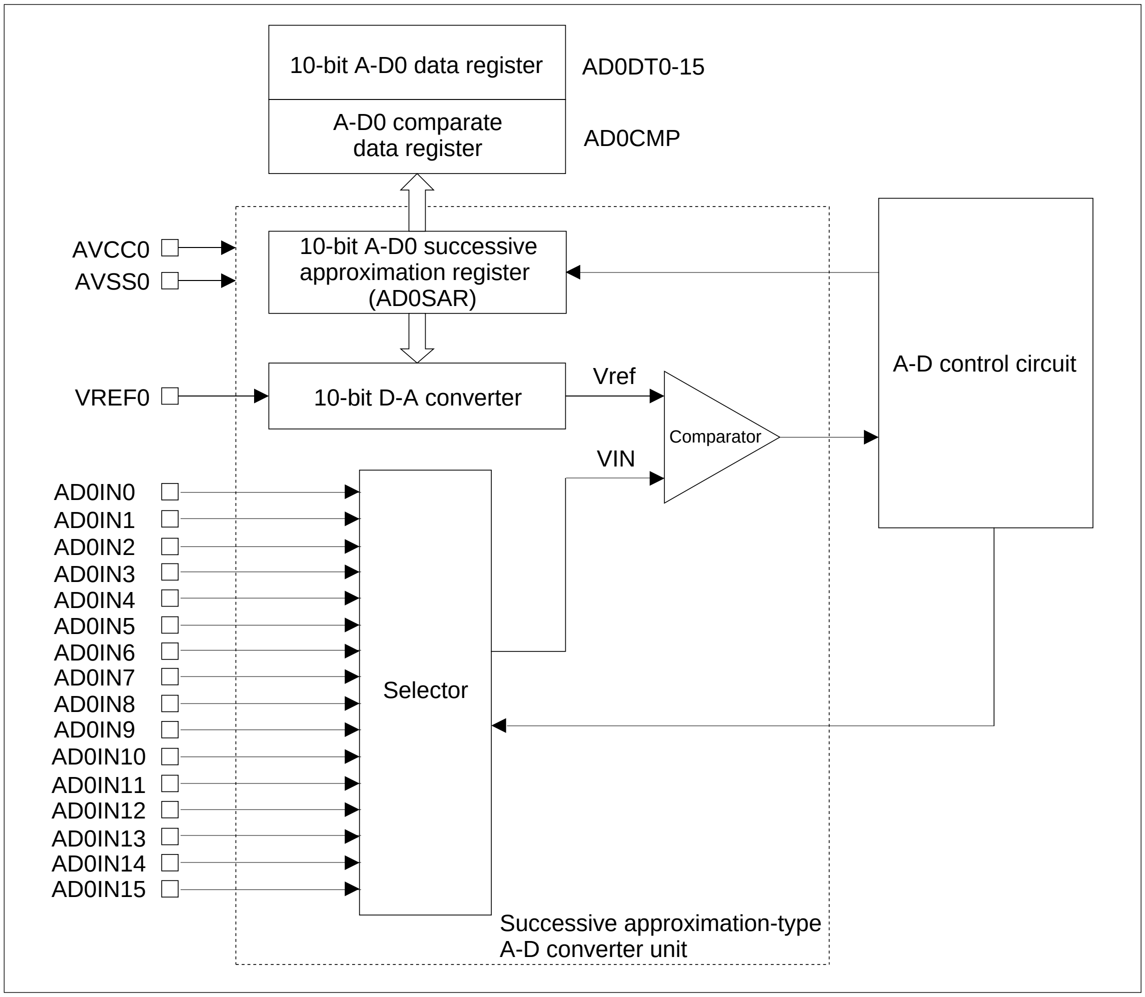


**Figure 11.3.1 Outline Block Diagram of the Successive Approximation-type A-D Converter Unit**

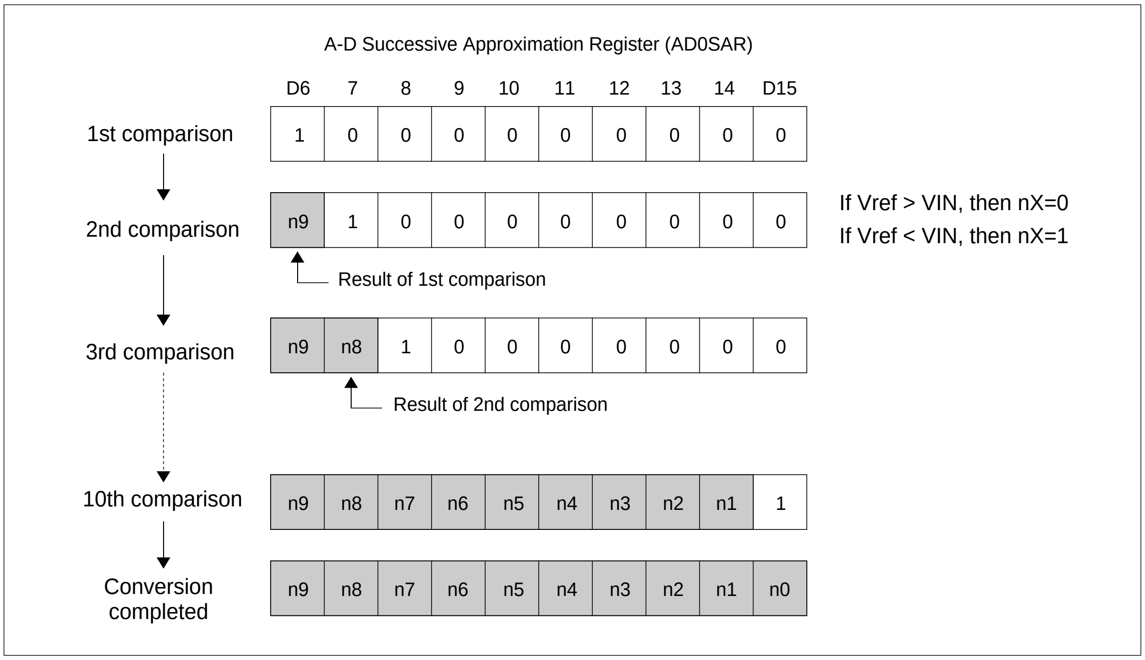
### 11.3.2 A-D Conversion by Successive Approximation Method

The A-D converter has A-D convert operation started by an A-D conversion start trigger (in software or hardware). Once A-D conversion begins, the following operation is automatically executed.

1. During single mode, Single Mode Register 0's A-D conversion/comparate completion bit is cleared to 0. During scan mode, Scan Mode Register 0's A-D conversion completion bit is cleared to 0.
2. The content of the A-D Successive Approximation Register is cleared to "H'0000."
3. The A-D Successive Approximation Register's most significant bit (D6) is set to 1.
4. The comparison voltage, Vref (Note 1), is fed from the D-A converter into the comparator.
5. The comparison voltage, Vref, and the analog input voltage, VIN, are compared, with the comparison result stored in D6.

   If Vref < VIN, then D6 = 1

   If Vref > VIN, then D6 = 0
6. Operations in steps 3 through 5 above are executed for all other bits from D7 to D15.
7. The value stored in the A-D Successive Approximation Register at completion of the comparison of D15 is the final A-D conversion result.



A-D Successive Approximation Register (AD0SAR)

| | D6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1st comparison | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2nd comparison | n9 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3rd comparison | n9 | n8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10th comparison | n9 | n8 | n7 | n6 | n5 | n4 | n3 | n2 | n1 | 1 |
| Conversion completed | n9 | n8 | n7 | n6 | n5 | n4 | n3 | n2 | n1 | n0 |

**Figure 11.3.2 Changes of the A-D Successive Approximation Register during A-D Convert Operation**

Note 1: The comparison voltage, Vref (the voltage fed from the D-A converter into the comparator), is determined according to changes of the content of the A-D Successive Approximation Register. Shown below are the equations used to calculate the comparison voltage, Vref.

- When the content of the A-D Successive Approximation Register = 0

  Vref [V] = 0

- When the content of the A-D Successive Approximation Register = 1 to 1,023

  Vref [V] = (reference voltage VREF0 / 1,024) x (content of the A-D Successive Approximation Register - 0.5)

The comparison result is stored in the 10-bit A-D Data Register (AD0DTn) corresponding to each converted channel. Also, the 8 high-order bits of the 10-bit A-D conversion result can be read out from the 8-bit A-D Data Register (AD08DTn).

The following shows the procedure for A-D conversion by successive approximation in each operation mode.

### (1) Single mode

The convert operation stops when comparison of the A-D Successive Approximation Register's D15 bit is completed. The content (A-D conversion result) of the A-D Successive Approximation Register is transferred to the 10-bit A-D Data Registers 0-15 for the converted channel.

### (2) Single-shot scan mode

When comparison of the A-D Successive Approximation Register's D15 bit in a specified channel is completed, the content of the A-D Successive Approximation Register is transferred to the corresponding 10-bit A-D Data Registers 0-15, and convert operations in steps 2 to 7 above are reexecuted for the next channel to be converted.
In single-shot scan mode, the convert operation stops when A-D conversion for one specified scan loop is completed.

### (3) Continuous scan mode

When comparison of the A-D Successive Approximation Register's D15 bit in a specified channel is completed, the content of the A-D Successive Approximation Register is transferred to the corresponding 10-bit A-D Data Registers 0-15, and convert operations in steps 2 to 7 above are reexecuted for the next channel to be converted.
During continuous scan mode, the convert operation is executed continuously until scan operation is forcibly halted by setting the A-D conversion stop bit (Scan Mode Register 0's D6 bit) to 1.

### 11.3.3 Comparator Operation

When comparator mode (single mode only) is selected, the A-D converter functions as a comparator which compares analog input voltages with the comparison voltage that is set by software.

When a comparison value is written to the successive approximation register, the A-D converter starts 'comparating' the analog input voltage selected by the Single Mode Register 1 analog input selection bit with the value written to the successive approximation register. Once comparate begins, the following operation is automatically executed.

1. The Single Mode Register 0 or Scan Mode Register 0's A-D conversion/comparate completion flag is cleared to 0.
2. The comparison voltage, Vref (Note 1), is fed from the D-A converter into the comparator.
3. The comparison voltage, Vref, and the analog input voltage, VIN, are compared, with the comparison result stored in the comparate result flag (A-D Comparate Data Register's D15).
   If Vref < VIN, then the comparate result flag = 0
   If Vref > VIN, then the comparate result flag = 1
4. The comparate operation stops after storing the comparison result.

The comparison result is stored in the A-D Comparate Data Register (AD0CMP)'s corresponding bit.

Note 1: The comparison voltage, Vref (the voltage fed from the D-A converter into the comparator), is determined according to changes of the content of the A-D Successive Approximation Register. Shown below are the equations used to calculate the comparison voltage, Vref.

- When the content of the A-D Successive Approximation Register = 0
  Vref [V] = 0

- When the content of the A-D Successive Approximation Register = 1 to 1,023
  Vref [V] = (reference voltage VREF0 / 1,024) x (content of the A-D Successive Approximation Register - 0.5)

### 11.3.4 Calculation of the A-D Conversion Time

The A-D conversion time is expressed by the sum of dummy cycle time and the actual execution cycle time. The following shows each time factor necessary to calculate the conversion time.

1. **Start dummy time**

   A time from when the CPU executed the A-D conversion start instruction to when the A-D converter starts A-D conversion

2. **A-D conversion execution cycle time**
3. **Comparate execution cycle time**
4. **End dummy time**

   A time from when the A-D converter finished A-D conversion to when the CPU can stably read out this conversion result from the A-D data register

5. **Scan to scan dummy time**

   A time during single-shot or continuous scan mode from when the A-D converter finished A-D conversion in a channel to when it starts A-D conversion in the next channel

The equation to calculate the A-D conversion time is as follows:

**A-D conversion time = Start dummy time + Execution cycle time**
**(+ Scan to scan dummy time + Execution cycle time**
**+ Scan to scan dummy time + Execution cycle time**
**+ Scan to scan dummy time .... + Execution cycle time)**
**+ End dummy time**

Note: • Shown in ( ) are the conversion time required for the second and subsequent channels to be converted in scan mode.

(1) Calculating the conversion time during A-D conversion mode

The following shows how to calculate the conversion time during A-D conversion mode.

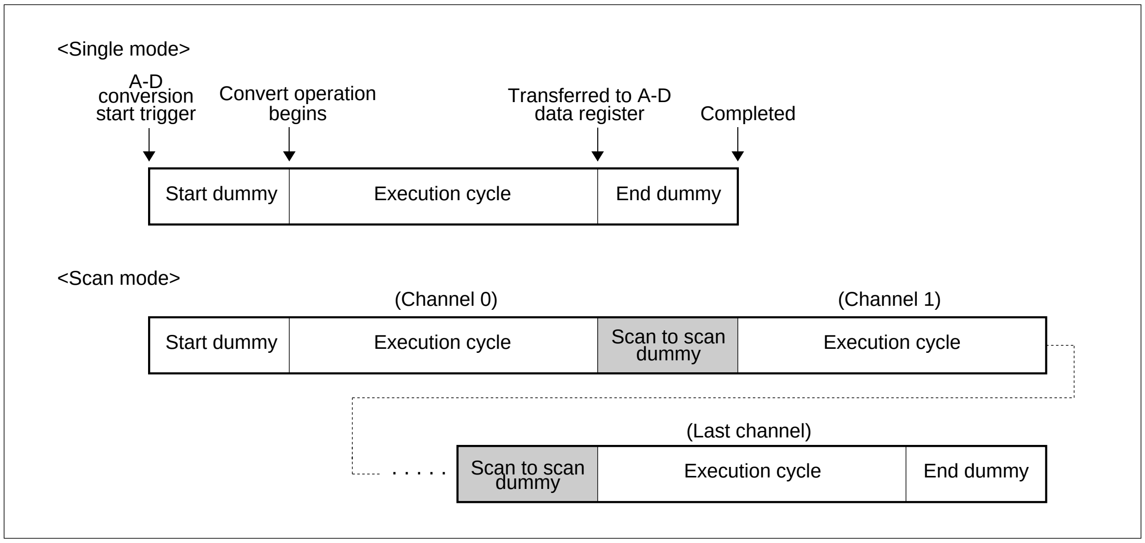


**Figure 11.3.3 Conceptual Diagram of Conversion Time in A-D Conversion mode**

**Table 11.3.1 Conversion Clock Cycles in A-D Conversion Mode**

Unit: BCLK

| **Conversion rate** | **Start dummy** | **A-D conversion execution cycle** | **End dummy** | **Scan to scan dummy** (Note 1) |
|---|---|---|---|---|
| Normal rate | 4 | 294 | 1 | 4 |
| Double rate | 4 | 168 | 1 | 4 |

Note 1: This applies to only scan mode, and is added to the execution time for each channel.

(2) Calculating the conversion time during comparate mode

The following shows how to calculate the conversion time during comparate mode.

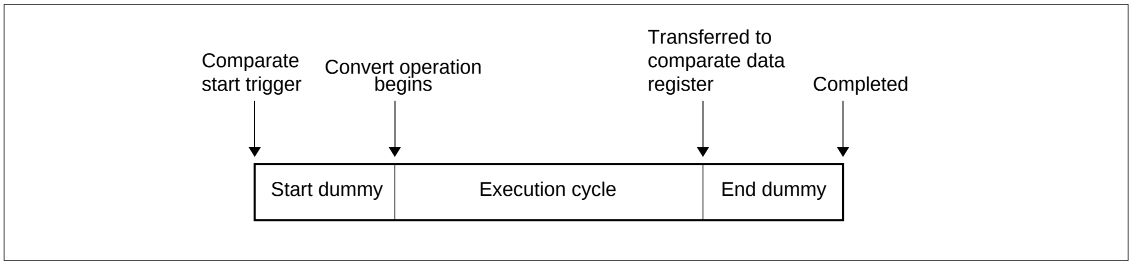


**Figure 11.3.4 Conceptual Diagram of Conversion Time in Comparate mode**

**Table 11.3.2 Conversion Clock Cycles in Comparate Mode**

Unit: BCLK

| **Conversion rate** | **Start dummy** | **Comparate execution cycle** | **End dummy** |
|---|---|---|---|
| Normal rate | 4 | 42 | 1 |
| Double rate | 4 | 24 | 1 |

(2) A-D conversion time

The table below lists A-D conversion times.

**Table 11.3.3 Total A-D Conversion Time**

| Conversion started by | Conversion rate | Conversion mode (Note 1) | | Conversion time [BCLK] |
|---|---|---|---|---|
| Software trigger<br>(Note 2) | Normal rate | Single mode | | 299 |
| | | Single-shot scan<br>/Continuous | 4-channel scan | 1193 |
| | | | 8-channel scan | 2385 |
| | | | 16-channel scan | 4769 |
| | | Comparator mode | | 47 |
| | Double rate | Single mode | | 173 |
| | | Single-shot scan<br>/Continuous | 4-channel scan | 689 |
| | | | 8-channel scan | 1377 |
| | | | 16-channel scan | 2753 |
| | | Comparator mode | | 29 |
| Hardware trigger<br>(Note 3) | Normal | Single mode | | 299 |
| | | Single-shot scan<br>/Continuous | 4-channel scan | 1193 |
| | | | 8-channel scan | 2385 |
| | | | 16-channel scan | 4769 |
| | | Comparator mode | | 47 |
| | Double speed | Single mode | | 173 |
| | | Single-shot scan<br>/Continuous | 4-channel scan | 689 |
| | | | 8-channel scan | 1377 |
| | | | 16-channel scan | 2753 |
| | | Comparator mode | | 29 |

Note 1: For single and comparator modes, this shows the time for A-D conversion in one channel or for comparate operation. For single-shot and continuous scan modes, this shows the time for A-D conversion in one scan loop.

Note 2: This shows the time from when a write-to-register cycle is completed to when an A-D conversion interrupt request is generated.

Note 3: This shows the time from when output event bus 3 is actuated to when an A-D conversion interrupt request is generated.

### 11.3.5 Definition of the A-D Conversion Accuracy

The accuracy of the A-D Converter is expressed by absolute accuracy. Absolute accuracy refers to the difference, expressed in terms of LSB, between the output code actually obtained by converting analog input voltages into digital quantities and the output code that can be expected from an A-D converter with ideal characteristics.

The analog input voltages used during accuracy measurement are chosen to be the midpoint values of voltage width at which an A-D converter with ideal characteristics will produce the same output code. For example, when VREF0 = 5.12 V, the width of 1 LSB of a 10-bit A-D converter is 5 mV, so that the middle points of analog input voltages are chosen to be 0 mV, 5 mV, 10 mV, 15 mV, 20 mV, 25 mV, and so on.

If the absolute accuracy of an A-D converter is said to be ±2 LSB, it means that if the input voltage is 25 mV, for example, then the actual A-D conversion result is in the range of H’003 to H’007, whereas the output code that can be expected from an ideal A-D converter is H’005. Note that absolute accuracy includes a zero error and full-scale error.

Although when actually using the A-D Converter, the analog input voltages are in the range of AVSS0 to VREF0, excessively lowering the VREF0 voltage requires caution because resolution may be degraded. Note also that output codes for analog input voltages from VREF0 to AVCC0 are always H’3FF.

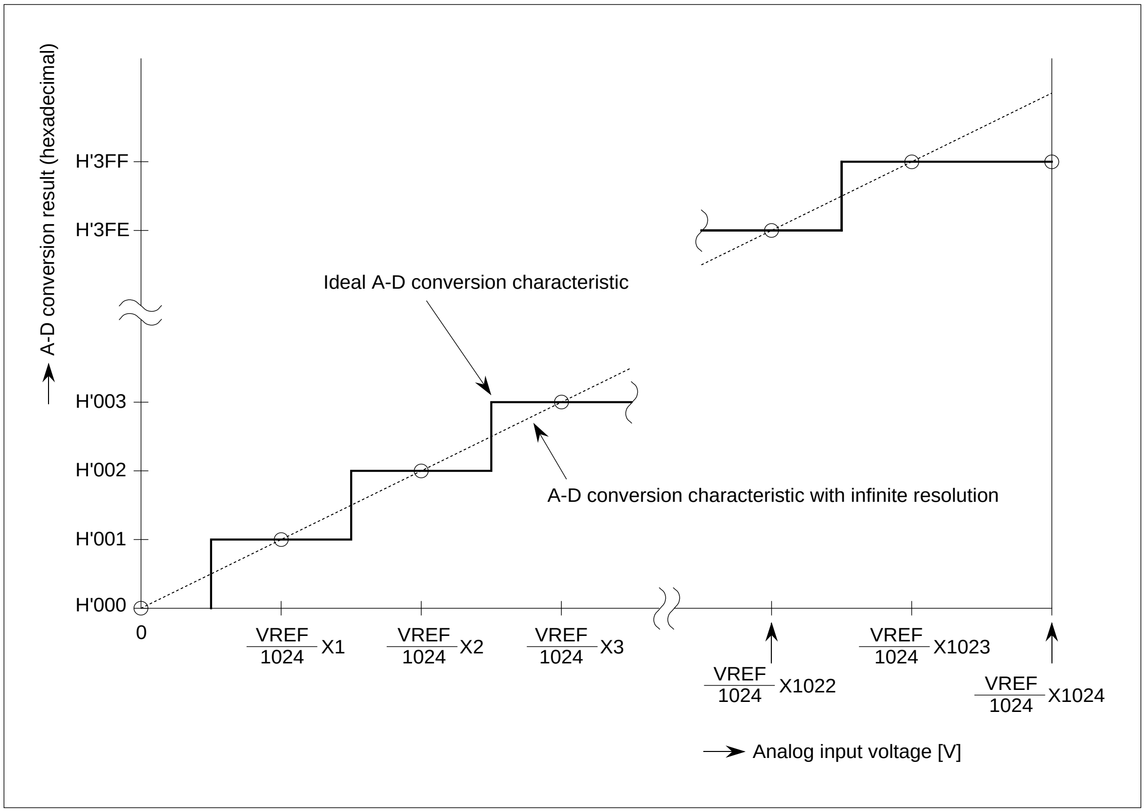


**Figure 11.3.5 Ideal A-D Conversion Characteristics Relative to the 10-bit A-D Converter's Analog Input Voltages**

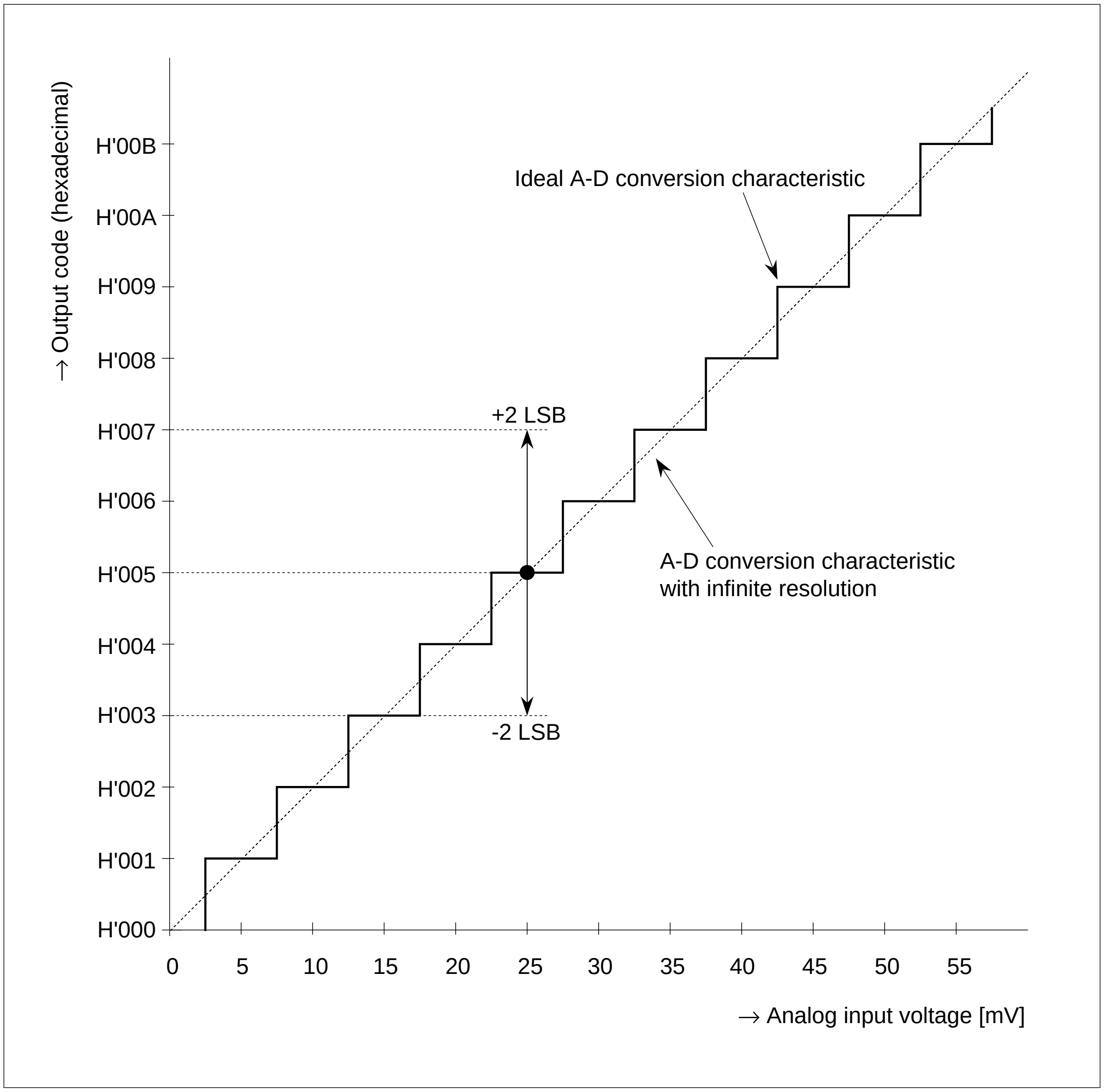


**Figure 11.3.6 Absolute Accuracy of an A-D Converter**

## 11.4 Precautions on Using A-D Converter

### • Forcible termination during scan operation

If A-D conversion is forcibly terminated by setting the A-D conversion stop bit (AD0CSTP) to 1 during scan mode operation and you read the content of the A-D data register for the channel in which conversion was in progress, it shows the last conversion result that had been transferred to the A-D data register before the conversion was forcibly terminated.

### • Modification of A-D converter related registers

If you want to change the contents of the A-D Conversion Interrupt Control Register, each Single and Scan Mode Register, or A-D Successive Approximation Register, except for the A-D conversion stop bit, do your change while A-D conversion is inactive, or be sure to restart A-D conversion after you changed the register contents. If the contents of these registers are changed in the middle of A-D conversion, the conversion results cannot be guaranteed.

### • Handling of analog input signals

The A-D converter included in the 32171 does not have a sample-and-hold circuit. Therefore, make sure the analog input levels are fixed during A-D conversion.

### • A-D conversion completion bit readout timing

If you want to read the A-D conversion completion bit (Single Mode Register 0's D5 bit or Scan Mode Register 0's D5 bit) immediately after A-D conversion has started, be sure to adjust the timing one clock cycle by, for example, inserting a NOP instruction before you read.

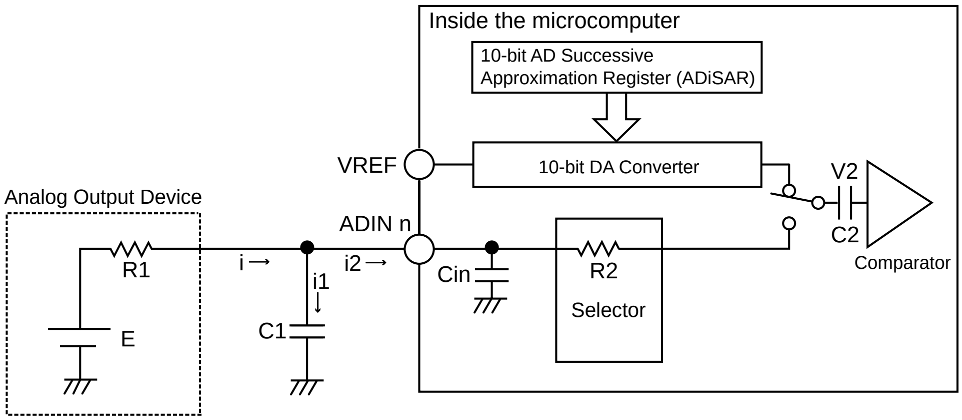
### • Rated value of absolute accuracy

The rated value of absolute accuracy is that of the microcomputer alone, premised on an assumption that power supply wiring on the board where the microcomputer is mounted is stable and unaffected by noise. When designing the board, pay careful attention to its layout by, for example, separating AVCC0, AVSS0, and VREF0 from other digital power supplies or protecting the analog input pins against noise from other digital signals.

## • Regarding the analog input pins

Figure 11.4.1 shows an internal equivalent circuit of the analog input unit. To obtain exact A-D conversion results, it is necessary that the A-D conversion circuit finishes charging its internal capacitor C2 within a designated time (sampling time). To meet this sampling time requirement, we recommend connecting a stabilizing capacitor, C1, external to the chip.

The following shows the analog output device's output impedance and how to determine the value of the external stabilizing capacitor to meet this timing requirement. Also shown below is the case where the analog output device's output impedance is low and the external stabilizing capacitor C1 is unnecessary.



C1 : Board's parasitic capacitance + stabilizing C
VREF : Analog reference voltage
C2 : Comparator capacitance (approx. 2.9 pF)
Cin : Input pin capacitance (approx. 10 pF)
R2 : Selector's parasitic resistance (1- 2 kΩ)
R1 : Analog output device's resistance
V2 : Voltage across C2
E : Analog output device's voltage

**Figure 11.4.1 Internal Equivalent Circuit of the Analog Input Unit**

### (a) Example for calculating the value of an external stabilizing capacitor C1 (recommended)

In Figure 11.4.1, as we calculate the capacitance of C1, we assume R1 is infinitely large, that the current needed to charge the internal capacitor C2 is sourced from C1, and that the voltage fluctuation due to C1 and C2 capacitance divisions, Vp, is 0.1 LSB or less. For the10-bit A-D converter where VREF is 5.12 V, the 1 LSB determination voltage = 5.12 V / 1024 = 5 mV. With up to 0.1 LSB voltage fluctuations considered, this equals 0.5 mV fluctuation.

The relationship between C1 and C2 capacitance divisions and Vp is obtained by the equation:

$$Vp = \frac{C2}{C1 + C2} \times (E - V2) \quad \text{------------------------------------------} \quad \text{Eq. (A-1)}$$

Also, Vp is obtained by the equation:

$$Vp = Vp1 \times \sum_{i=0}^{x-1} \frac{1}{2^i} < \frac{VREF}{10 \times 2^x} \quad \text{----------------------------------} \quad \text{Eq. (A-2)}$$

Notes: • Where Vp1 = voltage fluctuation in first A-D conversion.
• The exponent *x* is 10 because of a 10-bit resolution A-D converter.

When Eqs. (A-1) and (A-2) are solved,

$$C1 = C2 \left\{ \frac{E - V2}{Vp1} - 1 \right\} \quad \text{------------------------------------------} \quad \text{Eq. (A-3)}$$

$$\therefore \; C1 > C2 \left\{10 \times 2^x \times \sum_{i=0}^{x-1} \frac{1}{2^i} - 1 \right\} \quad \text{------------------------------} \quad \text{Eq. (A-4)}$$

Thus, for 10-bit resolution A-D converter where C2 = 2.9 pF, C1 is 0.06 µF or greater.

Use this for reference when determining the value of C1.

### (b) Maximum value of the output impedance R1 when not adding C1

In Figure 11.4.1, if the external capacitor C1 is not used, examination must be made of whether C2 can be fully charged. First, the following shows the equation to find i2 when C1 is nonexistent in Figure 11.4.1.

$$i2 = \frac{C2(E - V2)}{Cin \times R1 + C2(R1 + R2)} \times \exp\left\{\frac{-t}{Cin \times R1 + C2(R1 + R2)}\right\} \cdots\cdots\cdots \quad \text{Eq. (B-1)}$$

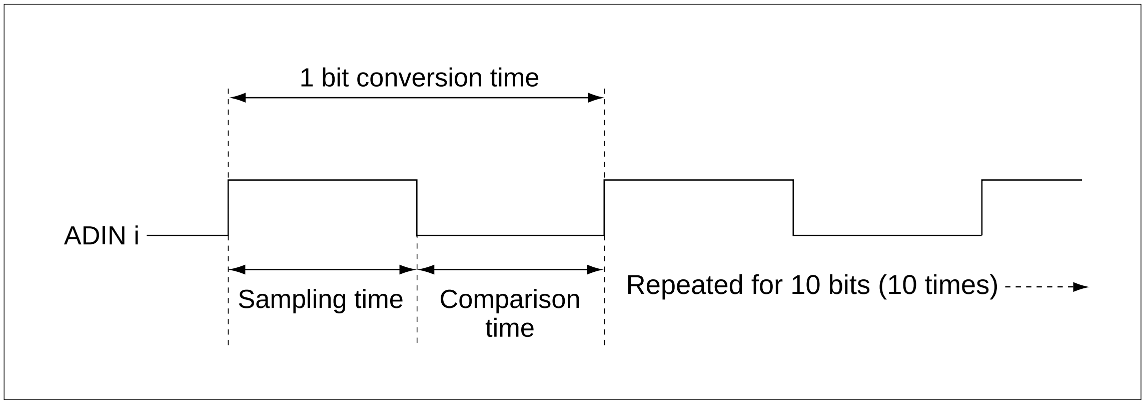


**Figure 11.4.2 A-D Conversion Timing Diagram**

The time needed for charging C2 must be within the sampling time (in Figure 11.4.2, A-D Conversion Timing Diagram) divided by 2.
Assuming t = T (time needed for charging C2)

$$T = \frac{\text{Sampling time}}{2} = \frac{\text{A-D conversion time}}{10 \times 4}$$

Therefore, from Eq. (B-1), the time needed for charging C2 is

$$T= \text{(time needed for charging C2)} > \text{Cin} \times \text{R1} + \text{C2(R1} + \text{R2)} \cdots\cdots\cdots\cdots \text{Eq. (B-2)}$$

Thus, the maximum value of R1 as an approximate guide can be obtained by the equation:

$$\text{R1} < \frac{\dfrac{\text{A-D conversion time}}{10 \times 4} - \text{C2} \times \text{R2}}{\text{Cin} + \text{C2}} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \text{Eq. (B-3)}$$

The table below shows an example of how to calculate the maximum value of R1 during A-D conversion mode when Xin = 10 and 8 MHz.

| Xin | BCLK period | Conversion mode | Speed mode | Conversion cycles | T (C2 charging time) in ns | Maximum value of R1 (Ω) |
|---|---|---|---|---|---|---|
| 10MHz | 50ns | A-D conversion | Normal | 294 | 367 | 28,225 |
| | | mode/Single | Double speed | 168 | 210 | 16,054 |
| 8MHz | 62.5ns | A-D conversion | Normal | 294 | 459 | 35,357 |
| | | mode/Single | Double speed | 168 | 262 | 20,085 |

Note: • The above conversion cycles do not include dummy cycles at the start and end of conversion.

In comparate mode, because sampling and comparison each are performed only once, the maximum value of R1 can be derived from the equation

$$\text{R1} < \frac{\dfrac{\text{A-D conversion time}}{4} - \text{C2} \times \text{R2}}{\text{Cin} + \text{C2}} \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \text{Eq. (B-4)}$$

The table below shows an example of how to calculate the maximum value of R1 during comparate mode when Xin = 10 and 8 MHz.

| Xin | BCLK period | Conversion mode | Speed mode | Conversion cycles | T (C2 charging time) in ns | Maximum value of R1 (Ω) |
|---|---|---|---|---|---|---|
| 10MHz | 50ns | comparate mode | Normal | 42 | 525 | 40,473 |
| | | /Single | Double speed | 24 | 300 | 23,031 |
| 8MHz | 62.5ns | comparate mode | Normal | 42 | 656 | 50,628 |
| | | /Single | Double speed | 24 | 375 | 28,845 |

Note: • The above conversion cycles do not include dummy cycles at the start and end of conversion.

# CHAPTER 12
# SERIAL I/O

12.1 Outline of Serial I/O
12.2 Serial I/O Related Registers
12.3 Transmit Operation in CSIO Mode
12.4 Receive Operation in CSIO Mode
12.5 Precautions on Using CSIO Mode
12.6 Transmit Operation in UART Mode
12.7 Receive Operation in UART Mode
12.8 Fixed Period Clock Output Function
12.9 Precautions on Using UART Mode

## 12.1 Outline of Serial I/O

The 32171 contains a total of three serial I/O channels: SIO0, SIO1, and SIO2. Serial channels SIO0 and SIO1 can be selected between CSIO mode (clock-synchronous serial I/O) and UART mode (asynchronous serial I/O). SIO2 is UART mode only.

- **CSIO mode (clock-synchronous serial I/O)**

  Communication is performed synchronously with transfer clock, using the same clock on both transmit and receive sides. The transfer data is 8 bits long (fixed).

- **UART mode (asynchronous serial I/O)**

  Communication is performed asynchronously. The transfer data length can be selected from 7 bits, 8 bits, and 9 bits.

Serial I/Os 0-2 each have transmit DMA and receive DMA transfer requests. Through a combined use with the internal DMAC, they allow for fast serial communication, and help to reduce the data communication load on the CPU.

Serial I/O is outlined in the pages to follow.

Table 12.1.1 Outline of Serial I/O

| Item | Content | |
|---|---|---|
| Number of channels | CSIO/UART : 2 channels (SIO0, SIO1)<br>UART only : 1 channels (SIO2) | |
| Clock | During CSIO mode : Internal clock or external clock as selected (Note 1)<br>During UART mode : Internal clock only | |
| Transfer mode | Transmit half-duplex, receive half-duplex, transmit/receive full-duplex | |
| BRG count source | f(BCLK), f(BCLK)/8, f(BCLK)/32, f(BCLK)/256 (when internal peripheral clock selected) (Note 2)<br>f(BCLK) : Internal peripheral clock operating frequency | |
| Data format | CSIO mode : | Data length = 8 bits (fixed)<br>Order of transfer = LSB first (fixed) |
| | UART mode : | Start bit = 1 bit<br>Character length = 7, 8, or 9 bits<br>Parity bit = Added or not added (when added, selectable between odd and even parity)<br>Stop bit = 1 or 2 bits<br>Order of transfer = LSB first (fixed) |
| Baud rate | CSIO mode : | 152 bits/sec to 2M bits/sec (at f(BCLK) = 20 MHz) |
| | UART mode : | 19 bits/sec to 1.25M bits/sec (at f(BCLK) = 20 MHz) |
| Error detection | CSIO mode : | Overrun error only |
| | UART mode : | Overrun error, parity error, framing error (Occurrence of any of these errors is indicated by an error sum bit) |
| Fixed period clock function | When using SIO0 and SIO1 as UART, this function outputs a divided-by-2 BRG clock from the SCLK pin. | |

Note 1 : The maximum input frequency of external clock during CSIO mode is 1/16 of f(BCLK).

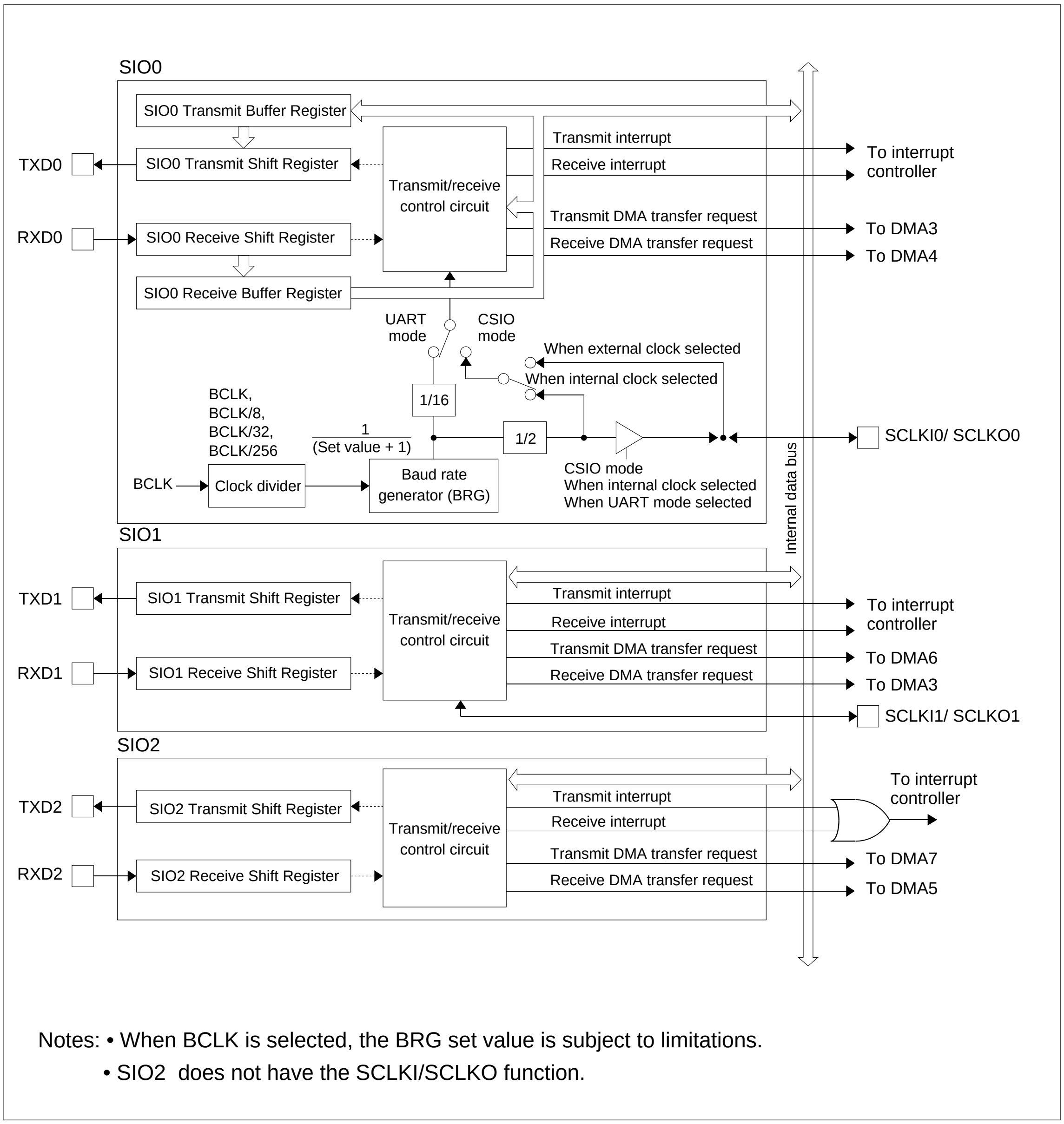
Note 2 : When f(BCLK) is selected as the BRG count source, the BRG set value is subject to limitations.

Table 12.1.2 Serial I/O Interrupt Request Generation Function

| Serial I/O Interrupt Request | ICU Interrupt Cause |
|---|---|
| SIO0 transmit buffer empty interrupt | SIO0 transmit interrupt |
| SIO0 receive-finished<br>or receive error interrupt (selectable) | SIO0 receive interrupt |
| SIO1 transmit buffer empty interrupt | SIO1 transmit interrupt |
| SIO1 receive-finished<br>or receive error interrupt (selectable) | SIO1 receive interrupt |
| SIO2 transmit buffer empty interrupt | SIO2 transmit/receive interrupt (group interrupt) |
| SIO2 receive-finished<br>or receive error interrupt (selectable) | SIO2 transmit/receive interrupt (group interrupt) |

Table 12.1.3 Serial I/O DMA Transfer Request Generation Function

| Serial I/O DMA Transfer Request | DMAC Input Channel |
|---|---|
| SIO0 transmit buffer empty | Channel 3 |
| SIO0 receive-finished | Channel 4 |
| SIO1 transmit buffer empty | Channel 6 |
| SIO1 receive-finished | Channel 3 |
| SIO2 transmit buffer empty | Channel 7 |
| SIO2 receive-finished | Channel 5 |


SIO0
SIO0 Transmit Buffer Register
TXD0
SIO0 Transmit Shift Register
Transmit/receive control circuit
RXD0
SIO0 Receive Shift Register
SIO0 Receive Buffer Register
Transmit interrupt
Receive interrupt
To interrupt controller
Transmit DMA transfer request
To DMA3
Receive DMA transfer request
To DMA4
UART mode
CSIO mode
When external clock selected
When internal clock selected
BCLK,
BCLK/8,
BCLK/32,
BCLK/256
1/16
1
(Set value + 1)
1/2
SCLKI0/ SCLKO0
BCLK
Clock divider
Baud rate generator (BRG)
CSIO mode
When internal clock selected
When UART mode selected
Internal data bus
SIO1
TXD1
SIO1 Transmit Shift Register
Transmit/receive control circuit
RXD1
SIO1 Receive Shift Register
Transmit interrupt
Receive interrupt
To interrupt controller
Transmit DMA transfer request
To DMA6
Receive DMA transfer request
To DMA3
SCLKI1/ SCLKO1
SIO2
TXD2
SIO2 Transmit Shift Register
Transmit/receive control circuit
RXD2
SIO2 Receive Shift Register
Transmit interrupt
Receive interrupt
To interrupt controller
Transmit DMA transfer request
To DMA7
Receive DMA transfer request
To DMA5

Notes: • When BCLK is selected, the BRG set value is subject to limitations.
• SIO2 does not have the SCLKI/SCLKO function.

**Figure 12.1.1 Block Diagram of SIO0-SIO2**

## 12.2 Serial I/O Related Registers

The diagram below shows a serial I/O related register map.

| Address | +0 Address | | +1 Address | |
|---|---|---|---|---|
| | D0 | D7 | D8 | D15 |
| H'0080 0100 | SIO23 Interrupt Status Register (SI23STAT) | | SIO03 Interrupt Mask Register (SI03MASK) | |
| H'0080 0102 | SIO03 Cause of Receive Interrupt Select Register (SI03SEL) | | | |
| | | | | |
| H'0080 0110 | SIO0 Transmit Control Register (S0TCNT) | | SIO0 Transmit/Receive Mode Register (S0MOD) | |
| H'0080 0112 | SIO0 Transmit Buffer Register (S0TXB) | | | |
| H'0080 0114 | SIO0 Receive Buffer Register (S0RXB) | | | |
| H'0080 0116 | SIO0 Receive Control Register (S0RCNT) | | SIO0 Baud Rate Register (S0BAUR) | |
| | | | | |
| H'0080 0120 | SIO1 Transmit Control Register (S1TCNT) | | SIO1 Transmit/Receive Mode Register (S1MOD) | |
| H'0080 0122 | SIO1 Transmit Buffer Register (S1TXB) | | | |
| H'0080 0124 | SIO1 Receive Buffer Register (S1RXB) | | | |
| H'0080 0126 | SIO1 Receive Control Register (S1RCNT) | | SIO1 Baud Rate Register (S1BAUR) | |
| | | | | |
| H'0080 0130 | SIO2 Transmit Control Register (S2TCNT) | | SIO2 Transmit/Receive Mode Register (S2MOD) | |
| H'0080 0132 | SIO2 Transmit Buffer Register (S2TXB) | | | |
| H'0080 0134 | SIO2 Receive Buffer Register (S2RXB) | | | |
| H'0080 0136 | SIO2 Receive Control Register (S2RCNT) | | SIO2 Baud Rate Register (S2BAUR) | |

Blank addresses are reserved.

**Figure 12.2.1 Serial I/O Related Register Map**

### 12.2.1 SIO Interrupt Related Registers

#### (1) Selecting the cause of interrupt

Interrupt signals sent from each SIO to the ICU (Interrupt Controller) are broadly classified into transmit interrupts and receive interrupts. Transmit interrupts are generated when the transmit buffer is empty. Receive interrupts are either receive-finished interrupts or receive error interrupts as selected by the Cause of Receive Interrupt Select Register (SI03SEL).

Note: • No interrupt signals are generated unless interrupts are enabled by the SIO Interrupt Mask Register after enabling the TEN (transmit enable) bit or REN (receive enable) bit for the corresponding SIO.

#### (2) Precautions on using transmit interrupts

Transmit interrupts are generated when the corresponding TEN (transmit enable) bit is enabled while the SIO Interrupt Mask Register is set to enable interrupts.

#### (3) About DMA transfer requests from SIO

Each SIO can generate a transmit DMA transfer and a receive-finished DMA transfer request. These DMA transfer requests can be generated by enabling each SIO's corresponding TEN (transmit enable) bit or REN (receive enable) bit. When using DMA transfers to communicate with external devices, be sure to set the DMAC before enabling the TEN or REN bits. When a receive error occurs, no receive-finished DMA transfer requests are generated.

- Transmit DMA transfer request

  Generated when the transmit buffer is empty and the TEN bit is enabled.

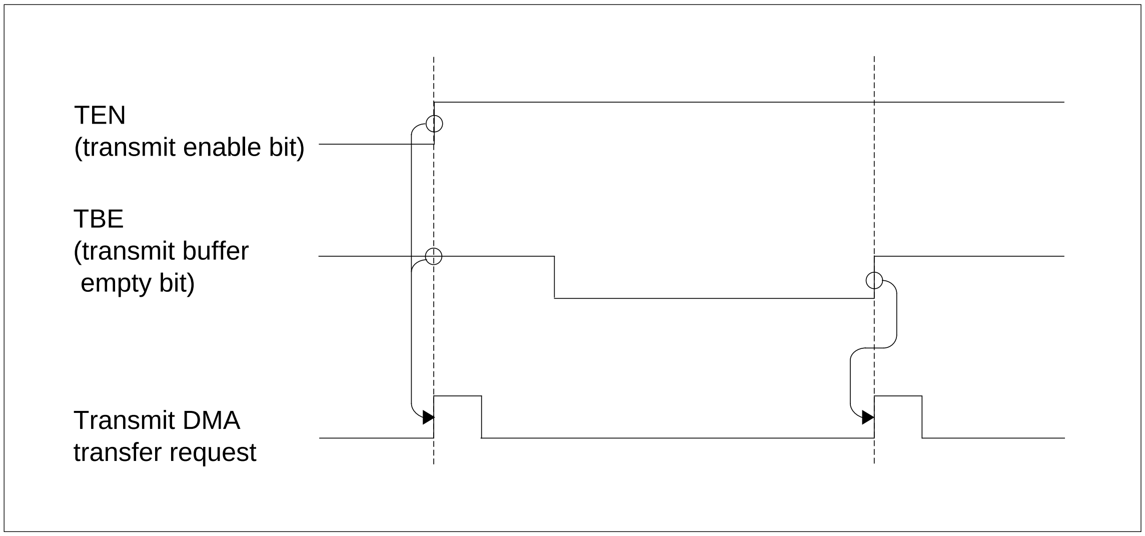
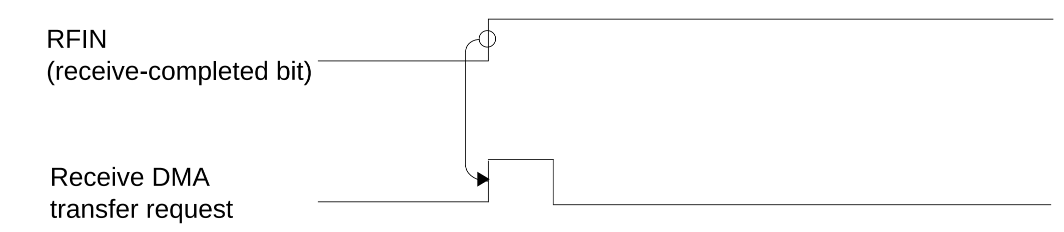


**Figure 12.2.2 Transmit DMA Transfer Request**

- Receive-finished DMA transfer request

  DMA transfer request is generated when the receive buffer is filled.



Note: • When a receive error occurs, no receive-finished DMA transfer requests are generated.

**Figure 12.2.3 Receive-finished DMA Transfer Request**

### 12.2.2 SIO Interrupt Control Registers

#### ■ SIO23 Interrupt Status Register (SI23STAT)

<Address: H'0080 0100>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  |  | IRQT2 | IRQR2 |  |  |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 3 | No functions assigned |  | 0 | — |
| 4 | IRQT2 (SIO2 transmit-finished interrupt request status bit) | 0 : Interrupt not requested<br>1 : Interrupt requested | ○ | △ |
| 5 | IRQR2 (SIO2 receive interrupt request status bit) | 0 : Interrupt not requested<br>1 : Interrupt requested | ○ | △ |
| 6 - 7 | These bits have no functions assigned. |  | 0 | — |

W = △ : Only writing a 0 is effective; when you write a 1, the previous value is retained.

Transmit/receive interrupt requests from SIO2 are described below.

[Setting the interrupt request status bit]

This bit can only be set in hardware, and cannot be set in software.

[Clearing the interrupt request status bit]

This bit is cleared by writing a 0 in software.

Note: • If the status bit is set in hardware at the same time it is cleared in software, the former has priority and the status bit is set.

When writing to the SIO Interrupt Status Register, make sure the bits you want to clear are set to 0 and all other bits are set to 1. The bits which are thus set to 1 are unaffected by writing in software and retain the value they had before you write.

## ■ SIO03 Interrupt Mask Register (SI03MASK)

<Address: H'0080 0101>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| T0MASK | R0MASK | T1MASK | R1MASK | T2MASK | R2MASK | | |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | T0MASK (SIO0 transmit interrupt mask bit) | 0 : Masks (disables) interrupt request<br>1 : Enables interrupt request | ○ | ○ |
| 9 | R0MASK (SIO0 receive interrupt mask bit) | 0 : Masks (disables) interrupt request<br>1 : Enables interrupt request | ○ | ○ |
| 10 | T1MASK (SIO1 transmit interrupt mask bit) | 0 : Masks (disables) interrupt request<br>1 : Enables interrupt request | ○ | ○ |
| 11 | R1MASK (SIO1 receive interrupt mask bit) | 0 : Masks (disables) interrupt request<br>1 : Enables interrupt request | ○ | ○ |
| 12 | T2MASK (SIO2 transmit interrupt mask bit) | 0 : Masks (disables) interrupt request<br>1 : Enables interrupt request | ○ | ○ |
| 13 | R2MASK (SIO2 receive interrupt mask bit) | 0 : Masks (disables) interrupt request<br>1 : Enables interrupt request | ○ | ○ |
| 14 - 15 | No functions assigned. | | 0 | — |

This register enables or disables interrupt requests generated by each SIO. Interrupt requests from an SIO are enabled by setting its corresponding interrupt mask bit to 1.

### ■ SIO03 Cause of Receive Interrupt Select Register (SI03SEL) <Address: H'0080 0102>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  |  | ISR0 | ISR1 | ISR2 |  |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 3 | No functions assigned |  | 0 | — |
| 4 | ISR0 (SIO0 receive interrupt cause select bit) | 0 : Receive-finished interrupt<br>1 : Receive error interrupt | ○ | ○ |
| 5 | ISR1 (SIO1 receive interrupt cause select bit) | 0 : Receive-finished interrupt<br>1 : Receive error interrupt | ○ | ○ |
| 6 | ISR2 (SIO2 receive interrupt cause select bit) | 0 : Receive-finished interrupt<br>1 : Receive error interrupt | ○ | ○ |
| 7 | No functions assigned. |  | 0 | — |

This register selects the cause of an interrupt generated at completion of receive operation.

[When set to 0]

Receive-finished interrupt (receive buffer full) is selected. Receive-finished interrupts occur for receive errors (except an overrun error), as well as for completion of receive operation.

[When set to 1]

Receive error interrupt is selected. The following lists the types of errors detected for reception errors.

- CSIO mode : Overrun error
- UART mode : Overrun error, parity error, and framing error

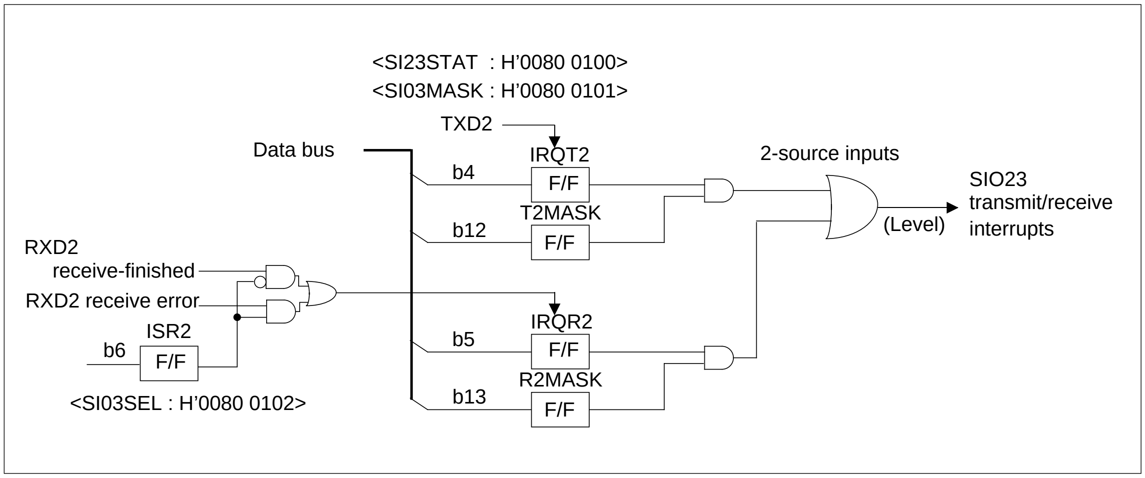


**Figure 12.2.4 Block Diagram of SIO23 Transmit Interrupts**

### 12.2.3 SIO Transmit Control Registers

**■ SIO0 Transmit Control Register (S0TCNT)** <Address: H'0080 0110>

**■ SIO1 Transmit Control Register (S1TCNT)** <Address: H'0080 0120>

**■ SIO2 Transmit Control Register (S2TCNT)** <Address: H'0080 0130>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  | CDIV |  |  | TSTAT | TBE | TEN |

<When reset : H'12>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 , 1 | No functions assigned |  | 0 | — |
| 2 , 3 | CDIV<br>(BRG count source select bit) | 00 : Selects f(BCLK)<br>01 : Selects divided-by-8 f(BCLK)<br>10 : Selects divided-by-32 f(BCLK)<br>11 : Selects divided-by-256 f(BCLK) | ○ | ○ |
| 4 | No functions assigned |  | 0 | — |
| 5 | TSTAT<br>(Transmit status bit) | 0 : Transmit halted & no data<br>in transmit buffer register<br>1 : Transmit in progress or data exists<br>in transmit buffer register | ○ | — |
| 6 | TBE<br>(Transmit buffer empty bit) | 0 : Data exists in transmit buffer register<br>1 : No data in transmit buffer register | ○ | — |
| 7 | TEN<br>(Transmit enable bit) | 0 : Disables transmit<br>1 : Enables transmit | ○ | ○ |

### (1) CDIV (baud rate generator count source select) bits (D2, D3)

These bits select the count source for the baud rate generator (BRG).

Note : • If f(BCLK) is selected as the count source for the BRG, make sure when you set BRG that the baud rate will not exceed the maximum transfer rate. For details, refer to the section of this manual where the SIO baud rate register is described.

### (2) TSTAT (transmit status) bit (D5)

[Set condition]

This bit is set to 1 by a write to the Transmit Buffer Register when transmit is enabled.

[Clear condition]

This bit is cleared to 0 when transmit is idle (no data in the Transmit Shift Register) and no data exists in the Transmit Buffer Register. This bit also is cleared by clearing the transmit enable bit.

### (3) TBE (transmit buffer empty) bit (D6)

[Set condition]

This bit is set to 1 when data is transferred from the Transmit Buffer Register to the Transmit Shift Register and the Transmit Buffer Register becomes empty. This bit also is set by clearing the transmit enable bit.

[Clear condition]

This bit is cleared to 0 by writing data to the lower byte of the Transmit Buffer Register when transmit is enabled (TEN = 1).

### (4) TEN (transmit enable) bit (D7)

Transmit is enabled by setting this bit to 1 and disabled by clearing this bit to 0. If this bit is cleared to 0 while transmitting data, the transmit operation stops.

### 12.2.4 SIO Transmit/Receive Mode Registers

| | |
|---|---|
| ■ **SIO0 Mode Register (S0MOD)** | <Address: H'0080 0111> |
| ■ **SIO1 Mode Register (S1MOD)** | <Address: H'0080 0121> |
| ■ **SIO2 Mode Register (S2MOD)** | <Address: H'0080 0131> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| SMOD | | | CKS | STB | PSEL | PEN | SEN |

<When reset : 00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 10 | SMOD | 000 : 7-bit UART | ○ | ○ |
| | (Serial I/O mode select bit) | 001 : 8-bit UART | | |
| | (Note 1) | 01X : 9-bit UART | | |
| | | 1XX : 8-bit clock-synchronized serial I/O | | |
| 11 | CKS | 0 : Internal clock | ○ | ○ |
| | (Internal/external clock select bit) | 1 : External clock | | (Note 2) |
| 12 | STB (Stop bit length select bit, | 0 : One stop bit | ○ | ○ |
| | UART mode only) | 1 : Two stop bits | | (Note 3) |
| 13 | PSEL (Parity odd/even select bit, | 0 : Odd parity | ○ | ○ |
| | UART mode only) | 1 : Even parity | | (Note 3) |
| 14 | PEN (Parity enable bit, | 0 : Disables parity | ○ | ○ |
| | UART mode only) | 1 : Enables parity | | (Note 3) |
| 15 | SEN (Sleep select bit, | 0 : Disables sleep function | ○ | ○ |
| | UART mode only) | 1 : Enables sleep function | | (Note 3) |

Note 1 : For SIO2, the D8 bit is fixed to 0 in hardware. You cannot set the D8 bit to 1 (to choose clock-synchronous serial I/O).

Note 2 : Has no effect when UART mode is selected.

Note 3 : D12 to D15 have no effect during clock-synchronous mode.

The SIO Mode Register consists of bits to set the serial I/O operation mode, data format, and the functions used during communication.

The SIO Transmit/Receive Mode Register must always be set before serial I/O starts operating. If you want to change settings of this register after the serial I/O started transmitting or receiving data, be sure to confirm that transmit and receive operations have been completed and disable transmit/ receive operations (by clearing the SIO Transmit Control Register transmit enable bit and SIO Receive Control Register receive enable bit to 0) before you change.

### (1) SMOD (serial I/O mode select) bits (D8 to D10)

These bits select the operation mode of serial I/O.

### (2) CKS (internal/external clock select) bit (D11)

This bit is effective when CSIO mode is selected. Setting this bit has no effect when UART mode is selected, in which case the serial I/O is clocked by an internal clock.

### (3) STB (stop bit length select) bit (D12)

This bit is effective when UART mode is selected. Use this bit to select the stop bit length that indicates the end of data to transmit. Setting this bit to 0 selects one stop bit, and setting this bit to 1 selects two stop bits. During clock-synchronous mode, the content of this bit has no effect.

### (4) PSEL (parity odd/even select) bit (D13)

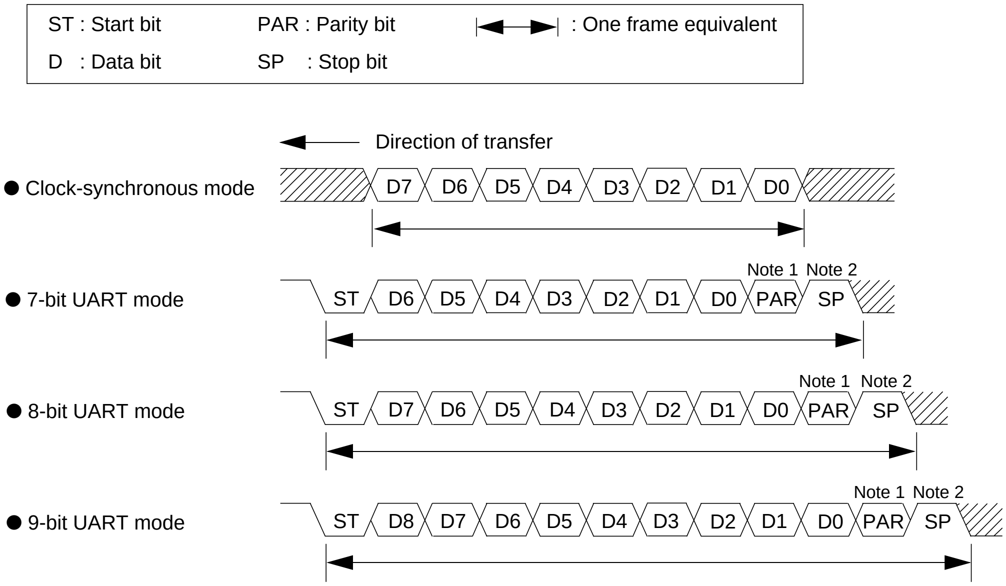
This bit is effective during UART mode. When parity is enabled (D14 = 1), use this bit to select the parity attribute (whether odd or even). Setting this bit to 0 selects an odd parity, and setting this bit to 1 selects an even parity. When parity is disabled (D14 = 0) and during clock-synchronous mode, the content of this bit has no effect.

### (5) PEN (parity enable) bit (D14)

This bit is effective during UART mode. When this bit is set to 1, a parity bit is added immediately after the data bits of transmit data, and for receive data, the parity in it is checked. The parity bit added to the transmit data is automatically determined to be a 1 or a 0 in such a way that the attribute (odd/even) of the sum of the number of 1's in data bits and the content of the parity bit agrees with one selected by the parity odd/even select bit (D13). Figure 12.2.5 shows an example of data format when parity is enabled.

### (6) SEN (sleep select) bit (D15)

This bit is effective during UART mode. If the sleep function is enabled by setting this bit to 1, data is latched into the UART Receive Buffer Register only when the most significant bit (MSB) of the received data is 1.



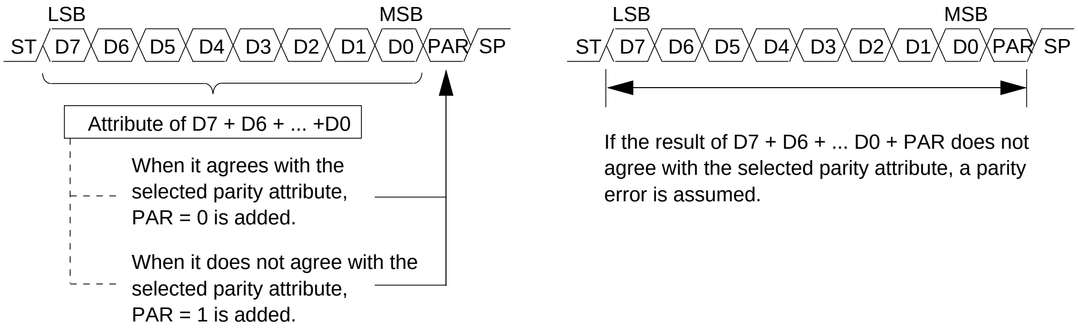
Note 1: Whether or not to add a parity bit is selectable.
Note 2: The stop bit can be one bit or two bits long as selected.

- When transmitting

If the attribute (odd/even) of the number of 1's included in data bits agrees with the selected parity attribute, a 0 is added as parity bit. If the attribute (odd/even) of the number of 1's included in data bits does not agree with the selected parity attribute, a 1 is added as parity bit.

- When receiving

Received data is checked to see if the number of 1's included in its data bits and the parity bit agrees with the parity attribute (known as parity check).


LSB
MSB
ST D7 D6 D5 D4 D3 D2 D1 D0 PAR SP
Attribute of D7 + D6 + ... +D0
LSB
MSB
ST D7 D6 D5 D4 D3 D2 D1 D0 PAR SP

When it agrees with the selected parity attribute, PAR = 0 is added.

When it does not agree with the selected parity attribute, PAR = 1 is added.

If the result of D7 + D6 + ... D0 + PAR does not agree with the selected parity attribute, a parity error is assumed.

Notes: • Shown above is an example of data format in 8-bit UART mode.
• The data bit numbers (Dn) above indicate bit numbers in a data list, and not the register bit numbers (Dn).

**Figure 12.2.5 Data Format when Parity is Enabled**

### 12.2.5 SIO Transmit Buffer Registers

■ **SIO0 Transmit Buffer Register (S0TXB)** <Address: H'0080 0112>

■ **SIO1 Transmit Buffer Register (S1TXB)** <Address: H'0080 0122>

■ **SIO2 Transmit Buffer Register (S2TXB)** <Address: H'0080 0132>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | TDATA | | | | | | | | |

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 6 | No functions assigned | | ? | ○ |
| 7 - 15 | TDATA<br>(Transmit data) | Sets transmit data. | ? | ○ |

R = ? : Indeterminate when read

The SIOn Transmit Buffer Register is used to set transmit data. This register is a write-only register, so you cannot read out the content of this register. Set data LSB-aligned, and write transmit data to bits D9-D15 for 7-bit data (UART mode only), D8-D15 for 8-bit data, or D7-D15 for 9-bit data (UART mode only).

Before you set data in this register, enable the Transmit Control Register TEN (transmit enable) bit by setting it to 1. Writing data to this register while the TEN bit is disabled (cleared to 0) has no effect. When data is written to the Transmit Buffer Register while transmit is enabled, the data is transferred from the SIO Transmit Buffer Register to the SIO Transmit Shift Register, upon which the serial I/O starts transmitting the data.

Note: • For 7-bit and 8-bit data, the register can be accessed bytewise.

### 12.2.6 SIO Receive Buffer Registers

**■ SIO0 Receive Buffer Register (S0RXB)** <Address: H'0080 0114>

**■ SIO1 Receive Buffer Register (S1RXB)** <Address: H'0080 0124>

**■ SIO2 Receive Buffer Register (S2RXB)** <Address: H'0080 0134>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | RDATA | | | | | | | | |

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 - 6 | No functions assigned | | 0 | — |
| 8 - 15 | RDATA<br>(Receive data) | Stores receive data. | ○ | — |

The SIOn Receive Buffer Register is used to store the receive data. When the serial I/O finishes receiving data, the content of the SIO Receive Shift Register is transferred to the SIO Receive Buffer Register. This register is a read-only register.

For 7-bit data (UART mode only), data is set in bits D9-D15, with D8 and D7 always set to 0. For 8-bit data, data is set in bits D8-D15, with D7 always set to 0.

After reception is completed, you may read out the content of the SIO Receive Buffer Register, but if the serial I/O finishes receiving the next data before you read the previous data, an overrun error occurs. In this case, the data received thereafter is not transferred to the Receive Buffer Register. To restart reception normally, clear the Receive Control Register's REN (receive enable) bit to 0.

Note: • For 7-bit and 8-bit data, the register can be accessed bytewise.

### 12.2.7 SIO Receive Control Registers

| | |
|---|---|
| ■ **SIO0 Receive Control Register (S0RCNT)** | <Address: H'0080 0116> |
| ■ **SIO1 Receive Control Register (S1RCNT)** | <Address: H'0080 0126> |
| ■ **SIO2 Receive Control Register (S2RCNT)** | <Address: H'0080 0136> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ▨ | RSTAT | RFIN | REN | OVR | PTY | FLM | ERS |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | No functions assigned | | 0 | — |
| 1 | RSTAT<br>(Receive status bit) | 0 : Reception stopped<br>1 : Reception in progress | ○ | — |
| 2 | RFIN<br>(Receive completed bit) | 0 : No data in receive buffer register<br>1 : Data exists in receive buffer register | ○ | — |
| 3 | REN<br>(Receive enable bit) | 0 : Disables reception<br>1 : Enables reception | ○ | ○ |
| 4 | OVR<br>(Overrun error bit) | 0 : No overrun error<br>1 : Overrun error occurred | ○ | — |
| 5 | PTY<br>(Parity error bit, UART mode only) | 0 : No parity error<br>1 : Parity error occurred | ○ | — |
| 6 | FLM<br>(Framing error bit, UART mode only) | 0 : No framing error<br>1 : Framing error occurred | ○ | — |
| 7 | ERS<br>(Error sum bit) | 0 : No error<br>1 : Error occurred | ○ | — |

**(1) RSTAT (receive status) bit (D1)**

[Set condition]

This bit is set to 1 by a start of receive operation. When this bit = 1, it means that the serial I/O is receiving data.

[Clear condition]

This bit is cleared to 0 upon completion of receive operation or by clearing the REN (receive enable) bit.

**(2) RFIN (receive completed) bit (D2)**

[Set condition]

This bit is set to 1 when all data bits have been received in the Receive Shift Register and whose content is transferred to the Receive Buffer Register.

[Clear condition]

This bit is cleared to 0 by reading the lower byte from the Receive Buffer Register or by clearing the REN (receive enable) bit. However, if an overrun error occurs, this bit cannot be cleared by reading the lower byte from the Receive Buffer Register. In this case, clear the REN (receive enable) bit.

**(3) REN (receive enable) bit (D3)**

Receive is enabled by setting this bit to 1, and is disabled by clearing this bit to 0, at which time the receive unit is initialized. Accordingly, the receive status flag, receive-completed flag bit, overrun error flag, framing error flag, parity error flag, and error sum flag all are cleared. The receive operation stops when the receive enable bit is cleared to 0 while receiving data.

**(4) OVR (overrun error) bit (D4)**

[Set condition]

This bit is set to 1 when all bits of the next receive data have been received in the Receive Shift Register while the Receive Buffer Register still contains the previous receive data. In this case, the receive data is not stored in the Receive Buffer Register. Although receive operation is continued when the overrun error flag = 1, the receive data is not stored in the Receive Buffer Register. To start reception normally, you need to clear this bit.

[Clear condition]

This bit is cleared by clearing the REN (receive enable) bit to 0.

**(5) PTY (parity error) bit (D5)**

This bit is effective in only UART mode. During CSIO mode, this bit is fixed to 0.

[Set condition]

The PTY (parity error) bit is set to 1 when the SIO Transmit/Receive Mode Register's PEN (parity enable/disable) bit is enabled and the parity (even/odd) of the receive data does not agree with the value that has been set by the said register's PSEL bit (parity select) bit.

[Clear condition]

The PTY bit is cleared by reading the lower byte from the SIO Receive Buffer Register or by clearing the SIO Receive Control Register's REN (receive enable) bit. However, if an overrun error occurs, this bit cannot be cleared by reading the lower byte from the Receive Buffer Register. In this case, clear the REN (receive enable) bit.

**(6) FLM (framing error) bit (D6)**

This bit is effective in only UART mode. During CSIO mode, this bit is fixed to 0.

[Set condition]

The FLM (framing error) bit is set to 1 when the number of received bits does not agree with one that has been selected by the SIO Transmit/Receive Mode Register.

[Clear condition]

The FLM bit is cleared by reading the lower byte from the SIO Receive Buffer Register or by clearing the SIO Receive Control Register's REN (receive enable) bit.
However, if an overrun error occurs, this bit cannot be cleared by reading the lower byte from the Receive Buffer Register. In this case, clear the REN (receive enable) bit.

**(7) ERS (Error sum) bit (D7)**

[Set condition]

This flag is set to 1 when any one of overrun, framing, or parity errors is detected at completion of reception.

[Clear condition]

If an overrun has occurred, this flag is cleared by clearing the REN (receive enable) bit.
Otherwise, this flag is cleared by reading the lower byte from the Receive Buffer Register or clearing the SIO Receive Control Register's REN (receive enable) bit.

### 12.2.8 SIO Baud Rate Registers

■ **SIO0 Baud Rate Register (S0BAUR)** <Address: H'0080 0117>

■ **SIO1 Baud Rate Register (S1BAUR)** <Address: H'0080 0127>

■ **SIO2 Baud Rate Register (S2BAUR)** <Address: H'0080 0137>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| BRG | | | | | | | |

<When reset : Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 15 | BRG<br>(Baud rate divide value) | Divides the baud rate count source selected by SIO Mode Register by (n + 1) according to the BRG set value 'n.' | ○ | ○ |

**BRG (baud rate divide value) (D8-D15)**

The SIO Baud Rate Register divides the baud rate count source selected by SIO Mode Register by (BRG set value + 1) according to the BRG set value.
In the initial state, the BRG value is indeterminate, so be sure to set the divide value before serial I/O starts operating. The value written to the BRG during transmit/receive operation takes effect in the next cycle after the BRG counter finished counting.

When using the internal clock (to output the SCLKO signal) in CSIO mode, the serial I/O divides the internal BCLK using the clock divider. Next, it divides the resulting clock by (BRG set value + 1) according to the BRG set value and then by 2, which results in generating a transmit/receive shift clock.

When using an external clock in CSIO mode, the serial I/O does not use the BRG. (Transmit/ receive operations are synchronized to the externally supplied clock.)

In UART mode, the serial I/O divides the internal BCLK using the clock divider. Next, it divides the resulting clock by (BRG set value + 1) according to the BRG set value and then by 16, which results in generating a transmit/receive shift clock.

When using SIO0 or SIO1 in UART mode, you can choose the relevant port (P84 or P87) to function as the SCLKO pin, so that a divided-by-2 BRG output clock can be output from the SCLKO pin.

When using the internal clock (internally clocked CSIO), with f(BCLK) selected as the BRG count source, make sure that during CSIO mode, the transfer rate does not exceed 2 Mbits per second.

# 12.3 Transmit Operation in CSIO Mode

## 12.3.1 Setting the CSIO Baud Rate

The baud rate (data transfer rate) in CSIO mode is determined by a transmit/receive shift clock. The clock source from which to generate the transmit/receive shift clock is selected from the internal clock f(BCLK) or external clock. The CKS (internal/external clock select) bit (SIO Transmit/Receive Mode Register D11 bit) is used to select the clock source. The equation by which to calculate the transmit/receive baud rate values differs with the selected clock source, whether internal or external.

### (1) When internal clock is selected in CSIO mode

When the internal clock is selected, f(BCLK) is divided by the clock divider before being fed into the baud rate generator (BRG).

The clock divider's divide-by value is selected from 1, 8, 32, or 256 by using the CDIV (baud rate generator count source select) bits (Transmit Control Register D2, D3 bits). The baud rate generator divides the clock divider output by (baud rate register set value + 1) and then by 2, which results in generating a transmit/receive shift clock.

When the internal clock is selected in CSIO mode, the baud rate is calculated using the equation below.

$$\underset{\text{[bps]}}{\text{Baud rate}} = \frac{\text{f (BCLK)}}{\text{Clock divider's divide-by value} \times \text{(baud rate register set value + 1)} \times 2}$$

f(BCLK):Internal peripheral clock operating frequency
Baud rate register set value = H'00 to H'FF (Note 1)
Clock divider's divide-by value = 1, 8, 32, or 256

Note 1: If the divide-by value selected for the baud rate generator count source is "1" (i.e., f(BCLK) itself), make sure the baud rate register value you set does not exceed 2 Mbps.

### (2) When external clock is selected in CSIO mode

In this case, the baud rate generator is not used; instead, the input clock from the SCLKI pin serves directly as CSIO transmit/receive shift clock.
<u>The maximum frequency of the SCLKI pin input clock is 1/16 of f(BCLK).</u>

$$\underset{\text{[bps]}}{\text{Baud rate}} = \text{SCLKI pin input clock}$$

### 12.3.2 Initial Settings for CSIO Transmission

To transmit data in CSIO mode, initialize the serial I/O following the procedure described below.

**(1) Setting SIO Transmit/Receive Mode Register**

- Set the register to CSIO mode
- Select the internal or an external clock

**(2) Setting SIO Transmit Control Register**

- Select the clock divider's divide-by ratio (when internal clock selected)

**(3) Setting SIO Baud Rate Register**

When the internal clock is selected, set a baud rate generator value. (Refer to Section 12.3.1, "Setting the CSIO Baud Rate.")

**(4) Setting SIO Interrupt Mask Register**

- Enable or disable the transmit buffer empty interrupt (SIO Interrupt Mask Register)

**(5) Setting the Interrupt Controller (SIO Transmit Interrupt Control Register)**

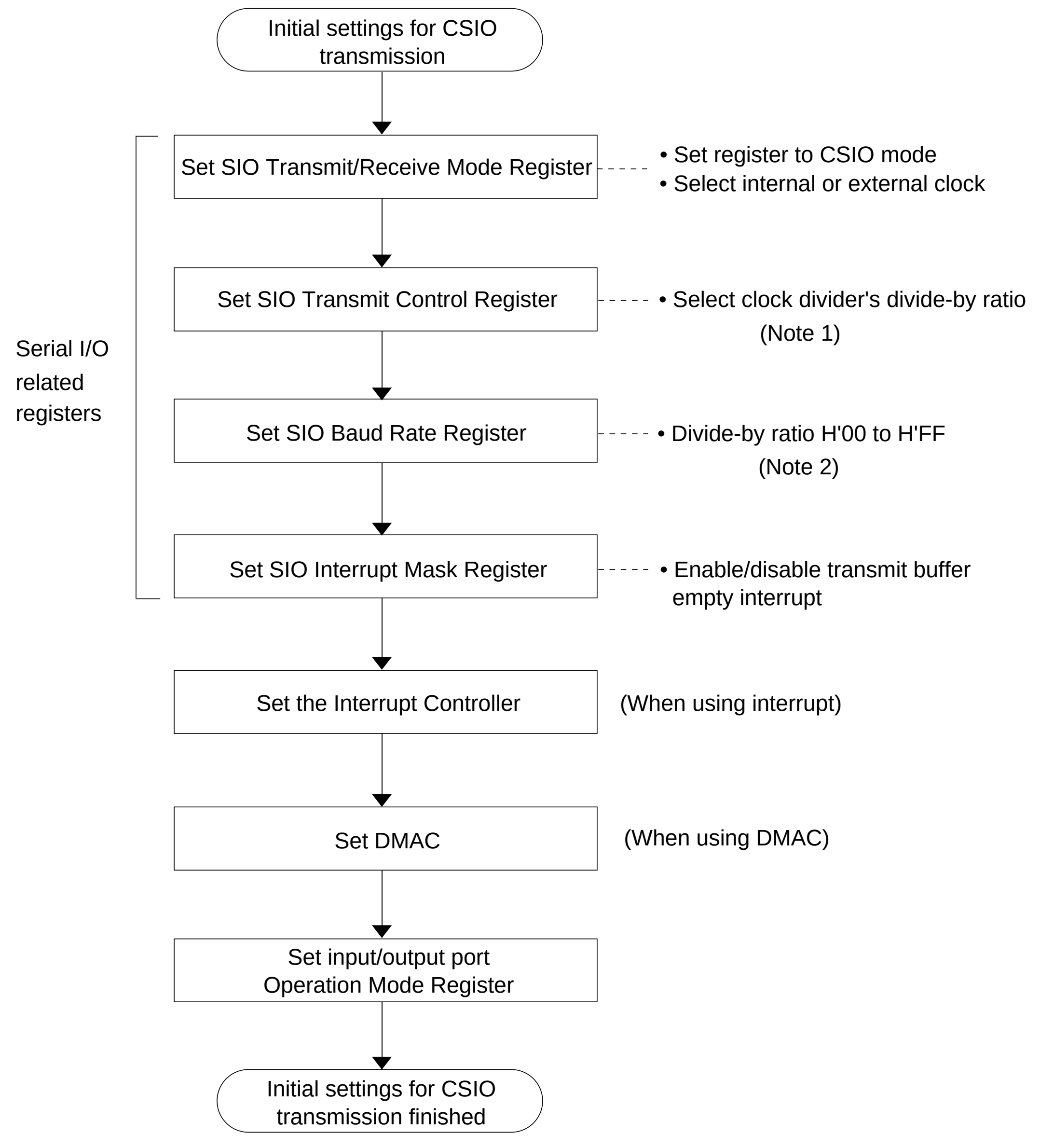
When you use a transmit buffer empty interrupt during transmission, set its priority level.

**(6) Setting DMAC**

When you issue DMA transfer requests to the internal DMAC when the transmit buffer is empty, set the DMAC. (Refer to Chapter 9, "DMAC.")

**(7) Selecting pin functions**

Because the serial I/O related pins serve dual purposes (shared with input/output ports), set pin functions. (Refer to Chapter 8, "Input/Output Ports and Pin Functions.")


Initial settings for CSIO transmission
Set SIO Transmit/Receive Mode Register
• Set register to CSIO mode
• Select internal or external clock
Set SIO Transmit Control Register
• Select clock divider's divide-by ratio (Note 1)
Serial I/O related registers
Set SIO Baud Rate Register
• Divide-by ratio H'00 to H'FF (Note 2)
Set SIO Interrupt Mask Register
• Enable/disable transmit buffer empty interrupt
Set the Interrupt Controller
(When using interrupt)
Set DMAC
(When using DMAC)
Set input/output port Operation Mode Register
Initial settings for CSIO transmission finished

Note 1 : This is necessary when you use the internal clock.

Note 2 : When you selected the internal clock and a divide-by ratio = 1, you are subject to limitations that the baud rate generator must be set not to exceed 2 Mbps.

**Figure 12.3.1 Procedure for CSIO Transmit Initialization**

### 12.3.3 Starting CSIO Transmission

When all of the following transmit conditions are met after you finished initialization, the serial I/O starts transmit operation.

**(1) Transmit conditions when CSIO mode internal clock is selected**

- The SIO Transmit Control Register's transmit enable bit is set to 1.
- Transmit data (8 bits) is written to the lower byte of the SIO Transmit Buffer Register (transmit buffer empty bit = 0).

**(2) Transmit conditions when CSIO mode external clock is selected**

- The SIO Transmit Control Register's transmit enable bit is set to 1.
- Transmit data is written to the lower byte of the SIO Transmit Buffer Register (transmit buffer empty bit = 0).
- A falling edge of transmit clock on the SCLKI pin is detected.

Notes: • While the transmit enable bit is cleared to 0, writes to the transmit buffer register are ignored. Always be sure to set the transmit enable bit to 1 before you write to the transmit buffer register.
• When the internal clock is selected, a write to the lower byte of the transmit buffer register in the note above triggers a start of transmission.
• The transmit status bit is set to 1 at the time data is set in the lower byte of the SIO Transmit Buffer Register.

When transmission starts, the serial I/O transmits data following the procedure below.

- Transfer the content of the SIO Transmit Buffer Register to the SIO Transmit Shift Register.
- Set the transmit buffer empty bit to 1. (Note 1)
- Start sending data synchronously with the shift clock beginning with the LSB.

Note 1: A transmit buffer empty interrupt request and/or a DMA transfer request can be generated when the transmit buffer is emptied.

### 12.3.4 Successive CSIO Transmission

Once data is transferred from the transmit buffer register to the transmit shift register, the next data can be written to the transmit buffer register even when transmission of the preceding data is not completed. When the next data is written to the transmit buffer before completion of the preceding data transmission, the preceding and the next data are successively transmitted.

To see if data has been transferred from the transmit buffer register to the transmit shift register, check the SIO Status Register's transmit buffer empty flag.

### 12.3.5 Processing at End of CSIO Transmission

When data transmission is completed, the following operation is automatically performed in hardware.

**(1) When not transmitting successively**

- The transmit status bit is set to 0.

**(2) When transmitting successively**

- When transmission of the last data in a consecutive data train is completed, the transmit status bit is set to 0.

### 12.3.6 Transmit Interrupt

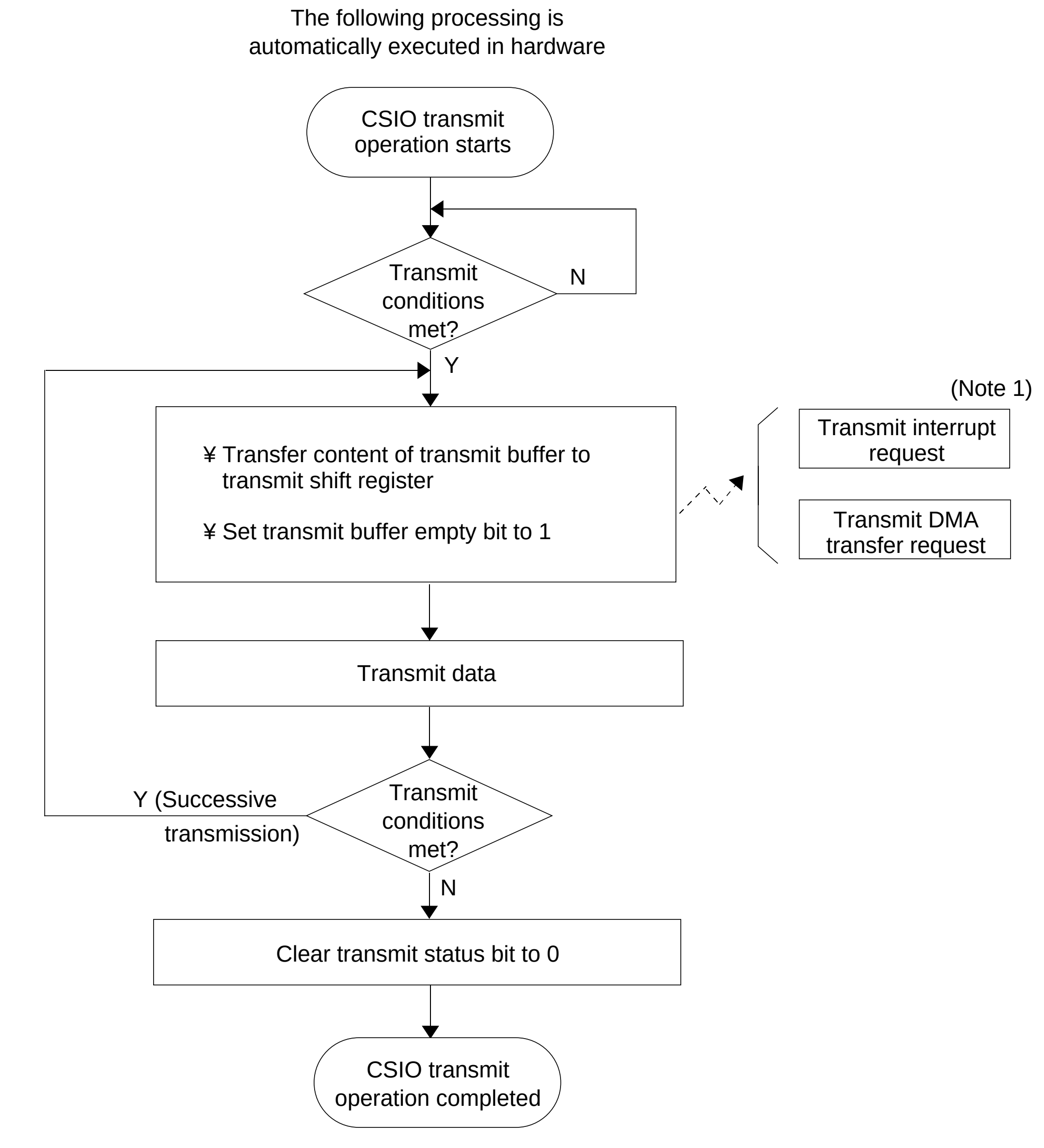
If a transmit buffer empty interrupt has been enabled by the SIO Interrupt Mask Register, a transmit buffer empty interrupt is generated at the time data is transferred from the transmit buffer register to the transmit shift register. Also, a transmit buffer empty interrupt is generated when the TEN (transmit enable) bit is set to 1 (enabled after being disabled) while a transmit buffer empty interrupt has been enabled.

You must set the Interrupt Controller (ICU) before you can use transmit interrupts.

### 12.3.7 Transmit DMA Transfer Request

When data has been transferred from the transmit buffer register to the transmit shift register, a transmit DMA transfer request for the corresponding SIO channel is ouput to the DMAC. This transfer request is also output when the TEN (transmit enable) bit is set to 1 (enabled after being disabled).

You must set the Interrupt Controller (ICU) before you can transmit data using DMA transfers.

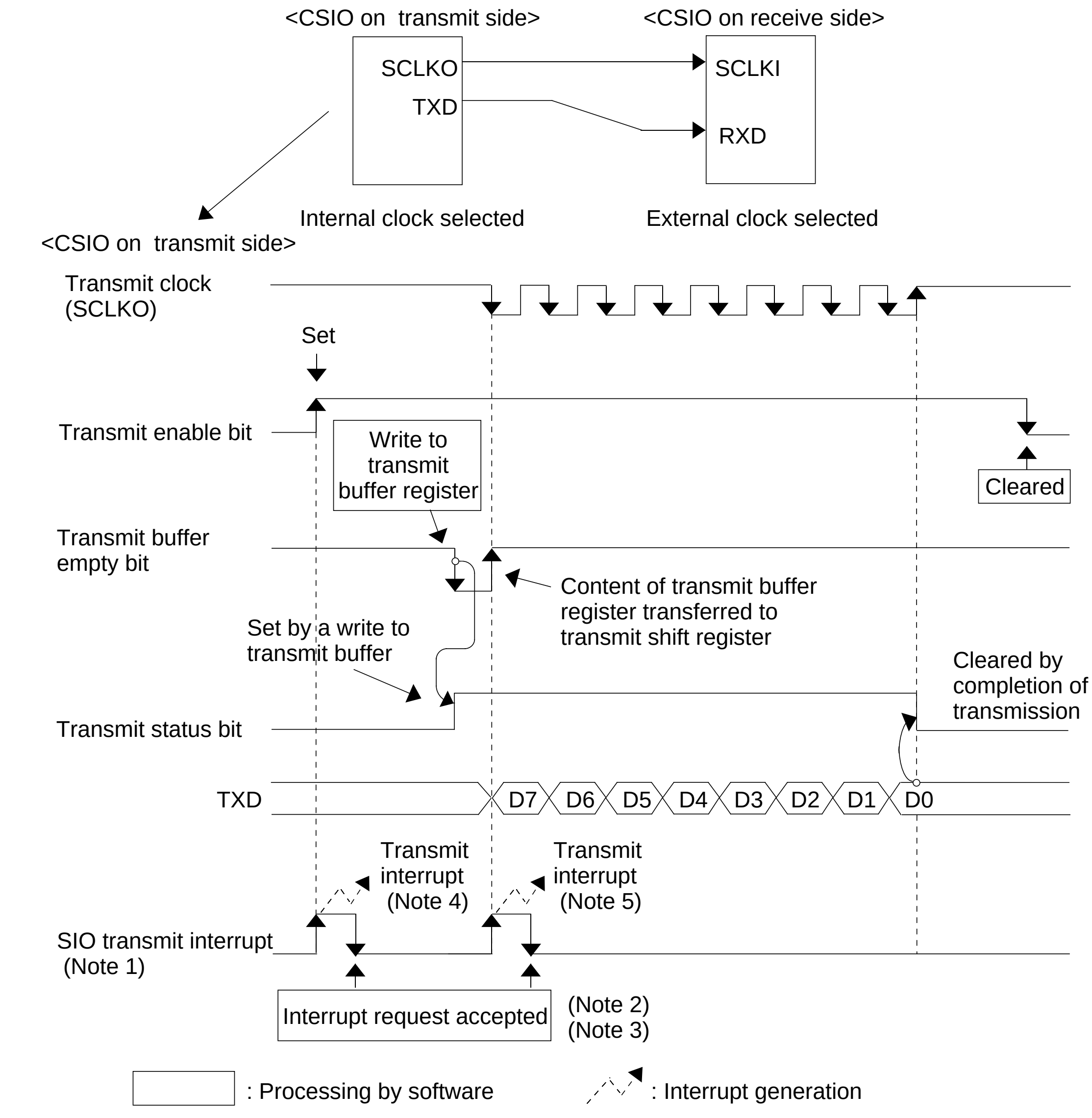

The following processing is automatically executed in hardware
CSIO transmit operation starts
Transmit conditions met?
N
Y
(Note 1)
Transmit interrupt request
Transmit DMA transfer request
¥ Transfer content of transmit buffer to transmit shift register
¥ Set transmit buffer empty bit to 1
Transmit data
Transmit conditions met?
Y (Successive transmission)
N
Clear transmit status bit to 0
CSIO transmit operation completed

Note 1: This applies when transmit interrupt has been enabled by SIO Interrupt Mask Register.

**Figure 12.3.2 Transmit Operation during CSIO Mode (Hardware Processing)**

### 12.3.8 Typical CSIO Transmit Operation

The following shows a typical transmit operation in CSIO mode.



Note 1 : Change of the Interrupt Controller "SIO Transmit Interrupt Control Register" interrupt request bit

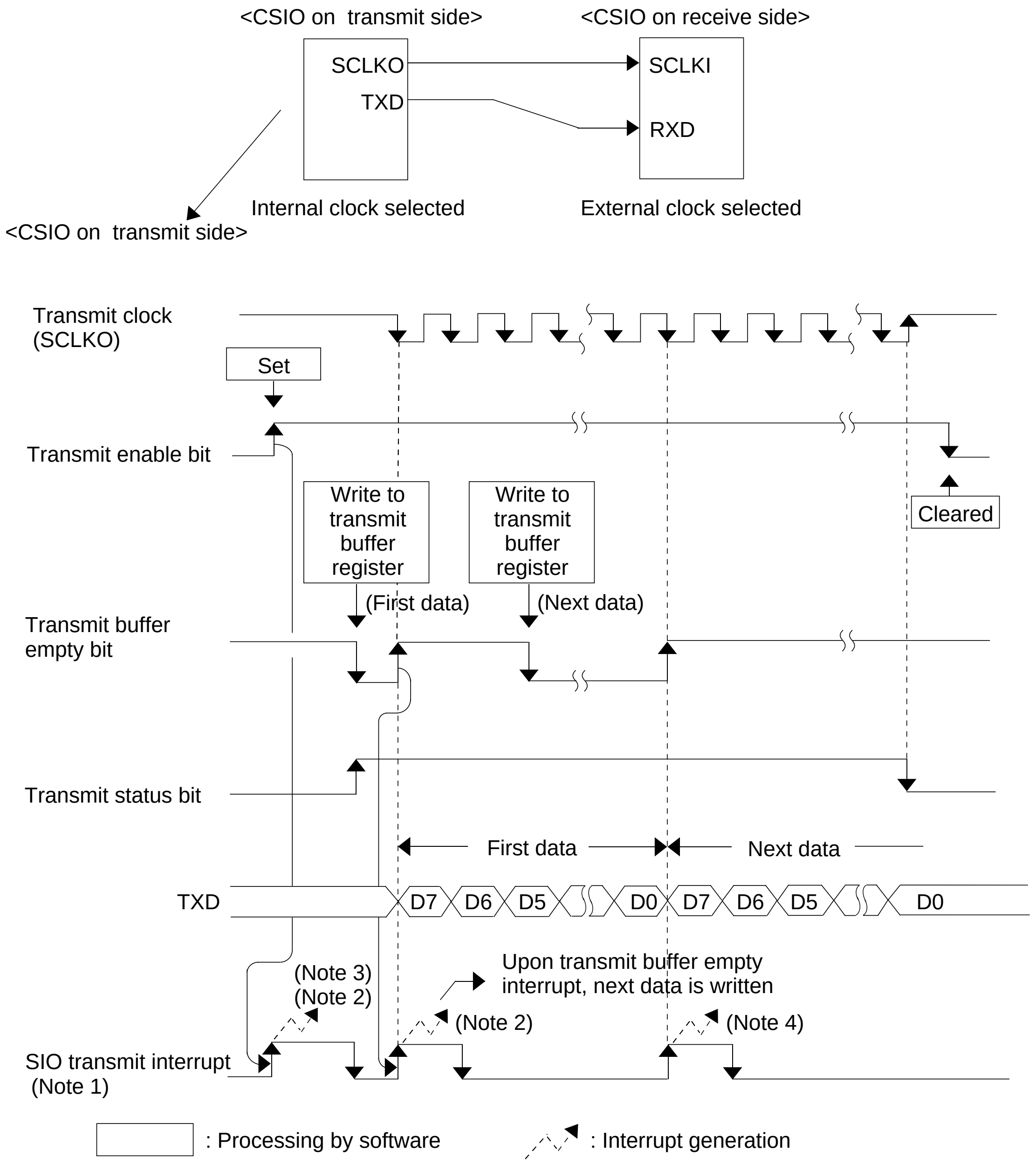
Note 2 : When transmit interrupt is enabled (DMA transfer can also be requested at the same timing)

Note 3 : The Interrupt Controller IVECT register is read or "SIO Transmit Interrupt Control Register" interrupt request bit cleared

Note 4 : Transmit interrupt request is generated when transmission is enabled.

Note 5 : Even after transmit data is written to the transmit buffer, a transmit interrupt request is generated when the data is transferred from the transmit buffer to the transmit shift register and the transmit buffer is thereby emptied.

**Figure 12.3.3 Example of CSIO Transmission (Transmitted Only Once, with Transmit Interrupt Used)**



Note 1 : Change of the Interrupt Controller "SIO Transmit Interrupt Control Register" interrupt request bit

Note 2 : When transmit interrupt is enabled (DMA transfer can also be requested at the same timing)

Note 3 : Transmit interrupt request is generated when transmission is enabled.

Note 4 : Even after transmit data is written to the transmit buffer, a transmit interrupt request is generated when the data is transferred from the transmit buffer to the transmit shift register and the transmit buffer is thereby emptied.

**Figure 12.3.4 Example of CSIO Transmission (Successive Transmission, with Transmit Buffer Empty and Transmit Finished Interrupts Used)**

# 12.4 Receive Operation in CSIO Mode

## 12.4.1 Initial Settings for CSIO Reception

To receive data in CSIO mode, initialize the serial I/O following the procedure described below. Note, however, that because the receive shift clock is derived from operation of the transmit circuit, you need to execute transmit operation even when you only want to receive data.

### (1) Setting SIO Transmit/Receive Mode Register

- Set the register to CSIO mode
- Select the internal or an external clock

### (2) Setting SIO Transmit Control Register

- Select the clock divider's divide-by ratio (when internal clock selected)

### (3) Setting SIO Baud Rate Register

When the internal clock is selected, set a baud rate generator value. (Refer to Section 12.3.1, "Setting the CSIO Baud Rate.")

### (4) Setting SIO Interrupt Mask Register

- Enable or disable the transmit buffer empty interrupt (SIO Interrupt Mask Register)
- Select the cause of receive interrupt (receive finished/error) (Cause of Receive Interrupt Select Register)

### (5) Setting SIO Receive Control Register

Set the receive enable bit

### (6) Setting the Interrupt Controller (SIO Transmit Interrupt Control Register)

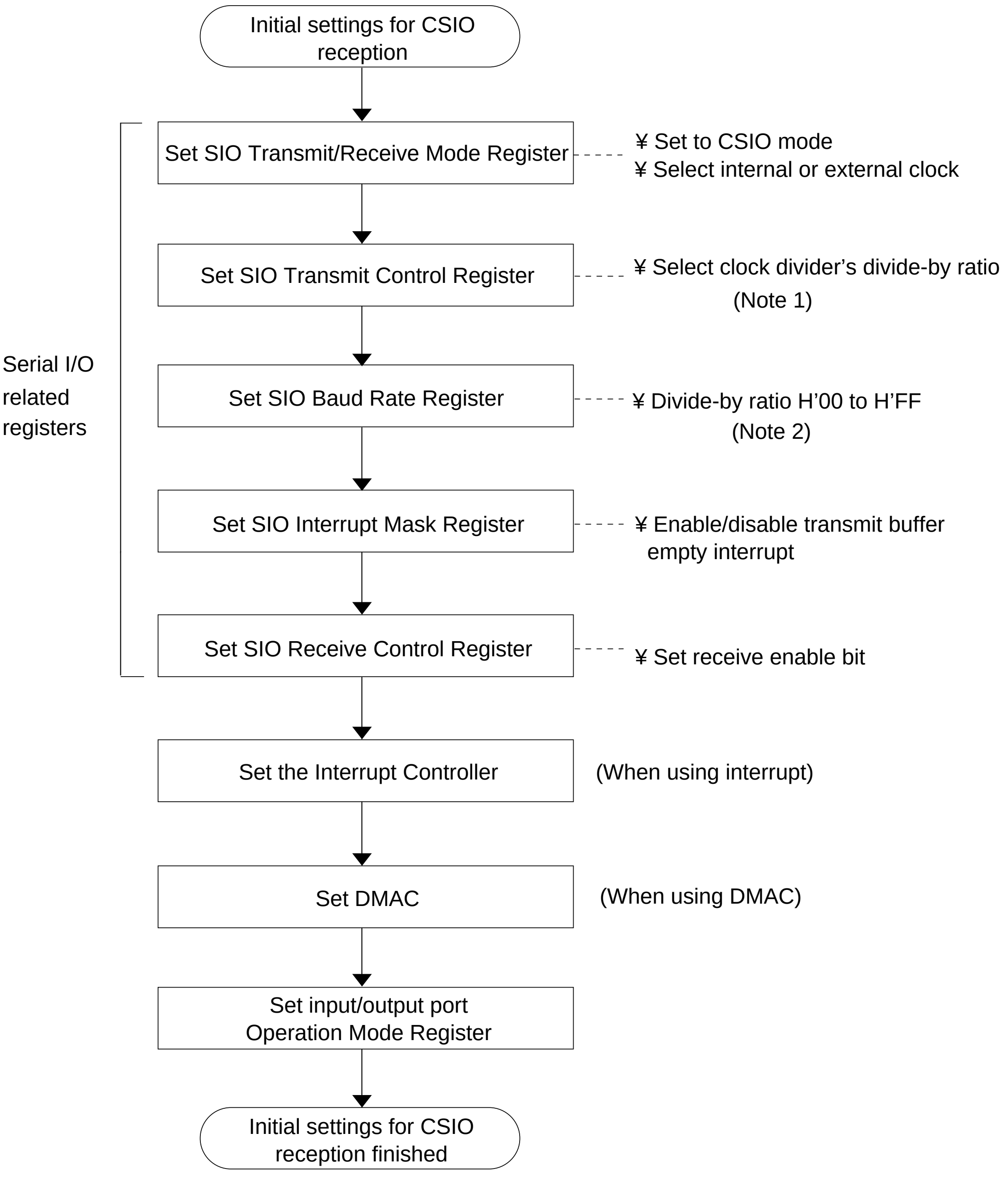
When you use a transmit interrupt or receive interrupt during transmission/reception, set its priority level.

### (7) Setting DMAC

When you generate a DMA transfer request to the internal DMAC when the transmit buffer is empty or transmission is completed, set the DMAC. (Refer to Chapter 9, "DMAC.")

### (8) Selecting pin functions

Because the serial I/O related pins serve dual purposes (shared with input/output ports), set pin functions. (Refer to Chapter 8, "Input/Output Ports and Pin Functions.")


Initial settings for CSIO reception
Set SIO Transmit/Receive Mode Register
¥ Set to CSIO mode
¥ Select internal or external clock
Set SIO Transmit Control Register
¥ Select clock divider’s divide-by ratio (Note 1)
Serial I/O related registers
Set SIO Baud Rate Register
¥ Divide-by ratio H’00 to H’FF (Note 2)
Set SIO Interrupt Mask Register
¥ Enable/disable transmit buffer empty interrupt
Set SIO Receive Control Register
¥ Set receive enable bit
Set the Interrupt Controller
(When using interrupt)
Set DMAC
(When using DMAC)
Set input/output port Operation Mode Register
Initial settings for CSIO reception finished

Note 1 : This is necessary when you use the internal clock.

Note 2 : When you selected the internal clock and a divide-by ratio = 1, you are subject to limitations that the baud rate generator must be set not to exceed 2 Mbps.

**Figure 12.4.1 Procedure for CSIO Receive Initialization**

### 12.4.2 Starting CSIO Reception

When all of the following receive conditions are met after you finished initialization, the serial I/O starts receive operation.

**(1) Receive conditions when CSIO mode internal clock is selected**

- The SIO Receive Control Register's receive enable bit is set to 1.
- Transmit conditions are met. (Refer to Section 12.3.3, "Starting CSIO Transmission.")

**(2) Receive conditions when CSIO mode external clock is selected**

- The SIO Receive Control Register's receive enable bit is set to 1.
- Transmit conditions are met. (Refer to Section 12.3.3, "Starting CSIO Transmission.")

Note: • The receive status bit is set to 1 at the time dummy data is set in the lower byte of the SIO Transmit Buffer Register.

When the above conditions are met, the serial I/O starts receiving 8-bit serial data (LSB first) synchronously with the receive shift clock.

### 12.4.3 Processing at End of CSIO Reception

When data reception is completed, the following operation is automatically performed in hardware.

**(1) When reception is completed normally**

The receive-finished (receive buffer full) bit is set to 1.

Notes: • If a receive-finished (receive buffer full) interrupt has been enabled, an interrupt request is generated.
• A DMA transfer request is generated.

**(2) When error occurs during reception**

When an error (only overrun error in CSIO mode) occurs during reception, the overrun error bit and receive sum bit are set to 1.

Notes: • If a receive-finished interrupt has been selected (by SIO Cause of Receive Interrupt Select Register), neither a receive-finished interrupt request nor a DMA transfer request is generated.
• If a receive error interrupt has been selected (by SIO Cause of Receive Interrupt Select Register), a receive error interrupt request is generated when interrupt requests are enabled. No DMA transfer requests are generated.

### 12.4.4 About Successive Reception

When the following conditions are met at completion of data reception, data may be received successively.

- The receive enable bit is set to 1.
- Transmit conditions are met.
- No overrun error has occurred.

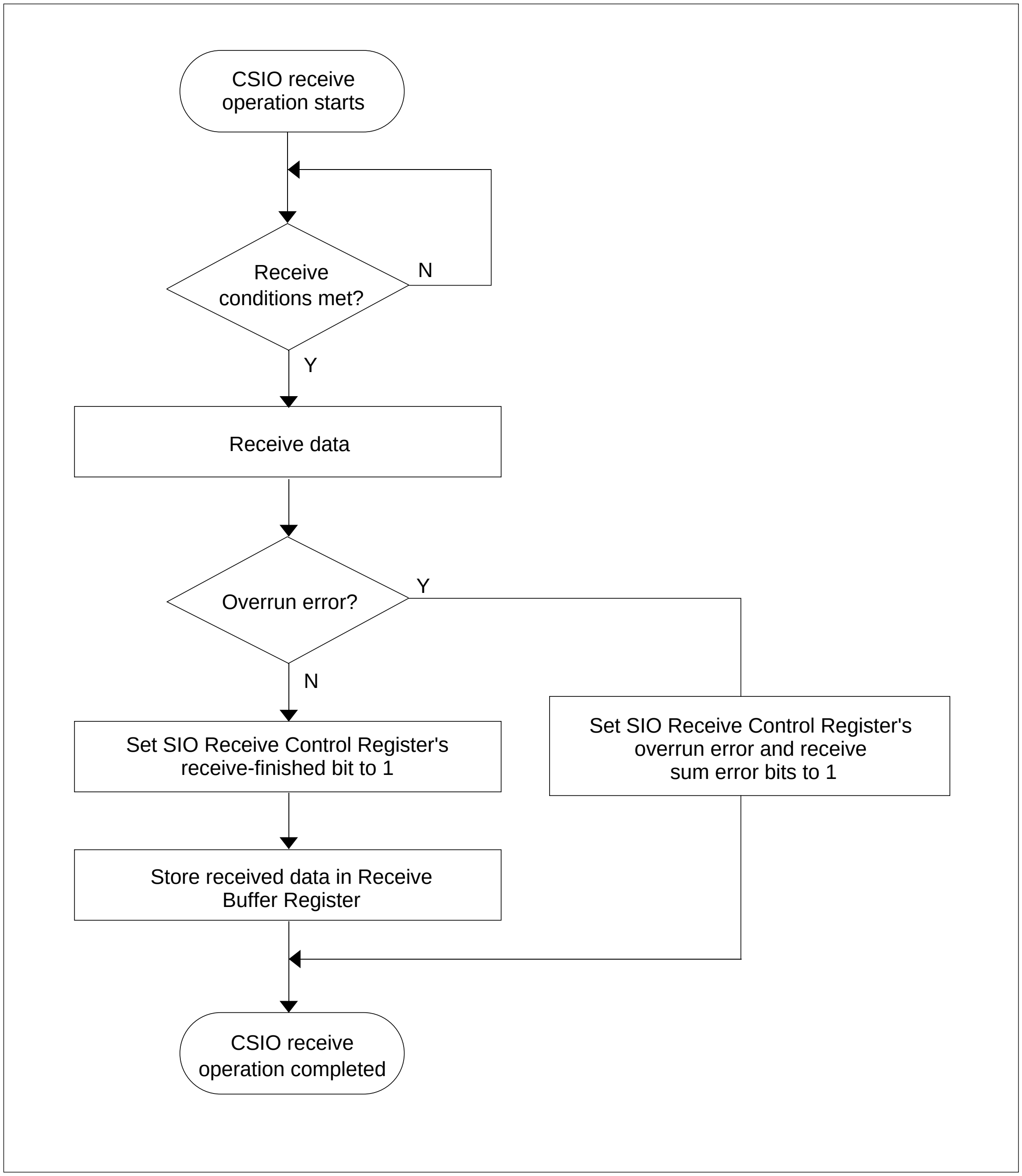

CSIO receive operation starts
Receive conditions met?
N
Y
Receive data
Overrun error?
Y
N
Set SIO Receive Control Register's receive-finished bit to 1
Set SIO Receive Control Register's overrun error and receive sum error bits to 1
Store received data in Receive Buffer Register
CSIO receive operation completed

**Figure 12.4.2 Receive Operation during CSIO Mode (Hardware Processing)**

### 12.4.5 Flags Indicating the Status of CSIO Receive Operation

Following flags are available that indicate the status of receive operation in CSIO mode.

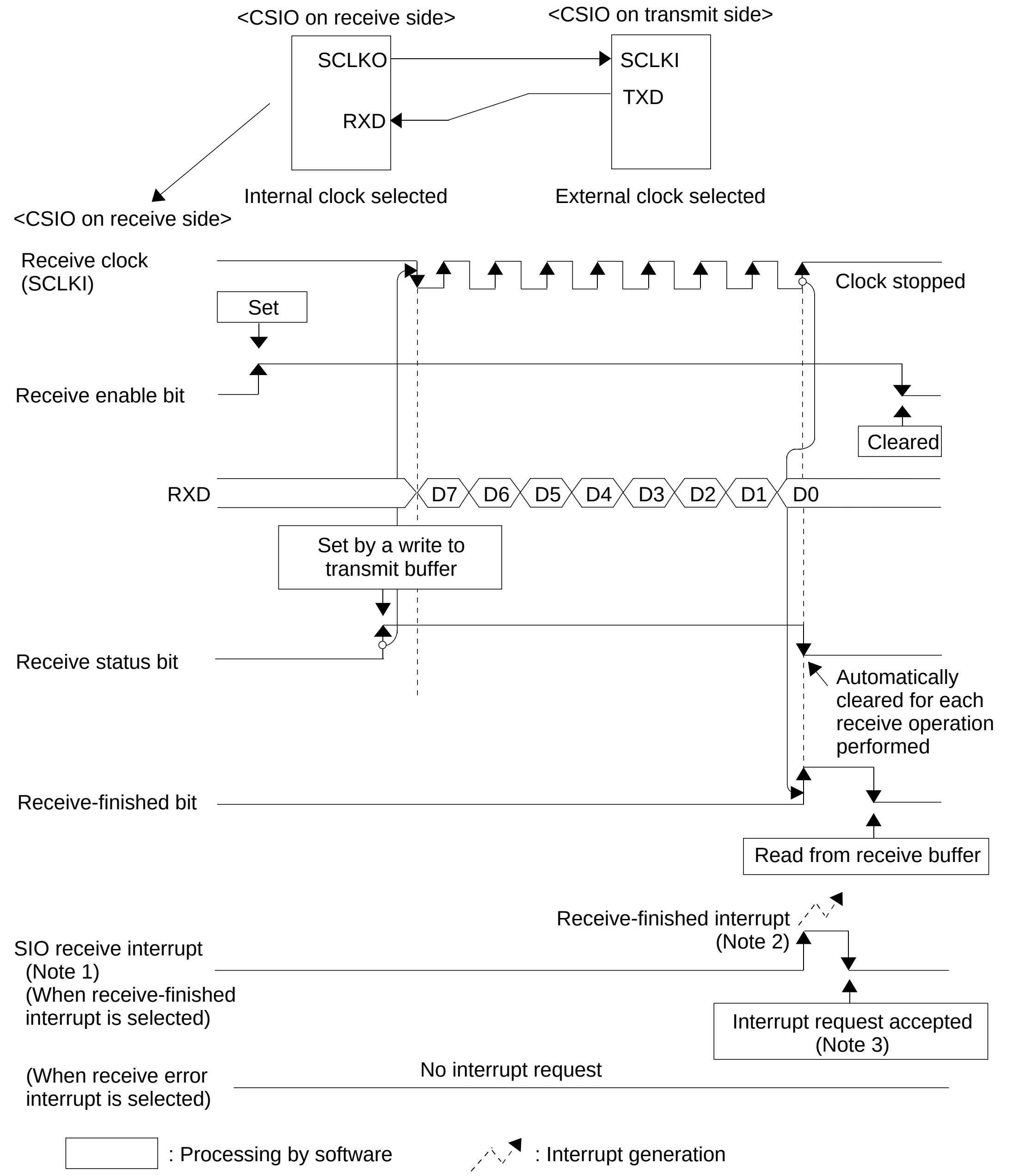
- SIO Receive Control Register receive status bit
- SIO Receive Control Register receive-finished bit
- SIO Receive Control Register receive error sum bit
- SIO Receive Control Register overrun error bit

After reception is completed, you may read out the content of the SIO Receive Buffer Register, but if the serial I/O finishes receiving the next data before you read, an overrun error occurs. In this case, the data received thereafter is not transferred to the SIO Receive Buffer Register. To restart reception, temporarily clear the receive enable bit to 0 and initialize the receive control block before you restart.
The said receive enable bit can be cleared, when there are no receive errors (Note 1) encountered, by reading the lower byte from the SIO Receive Buffer Register or clearing the REN (receive enable) bit. If any receive error has occurred, it can only be cleared by clearing the REN (receive enable) bit, and cannot be cleared by reading the lower byte from the SIO Receive Buffer Register.

Note 1: Overrun error is the only error that can be detected during reception in CSIO mode.

### 12.4.6 Typical CSIO Receive Operation

The following shows a typical receive operation in CSIO mode.

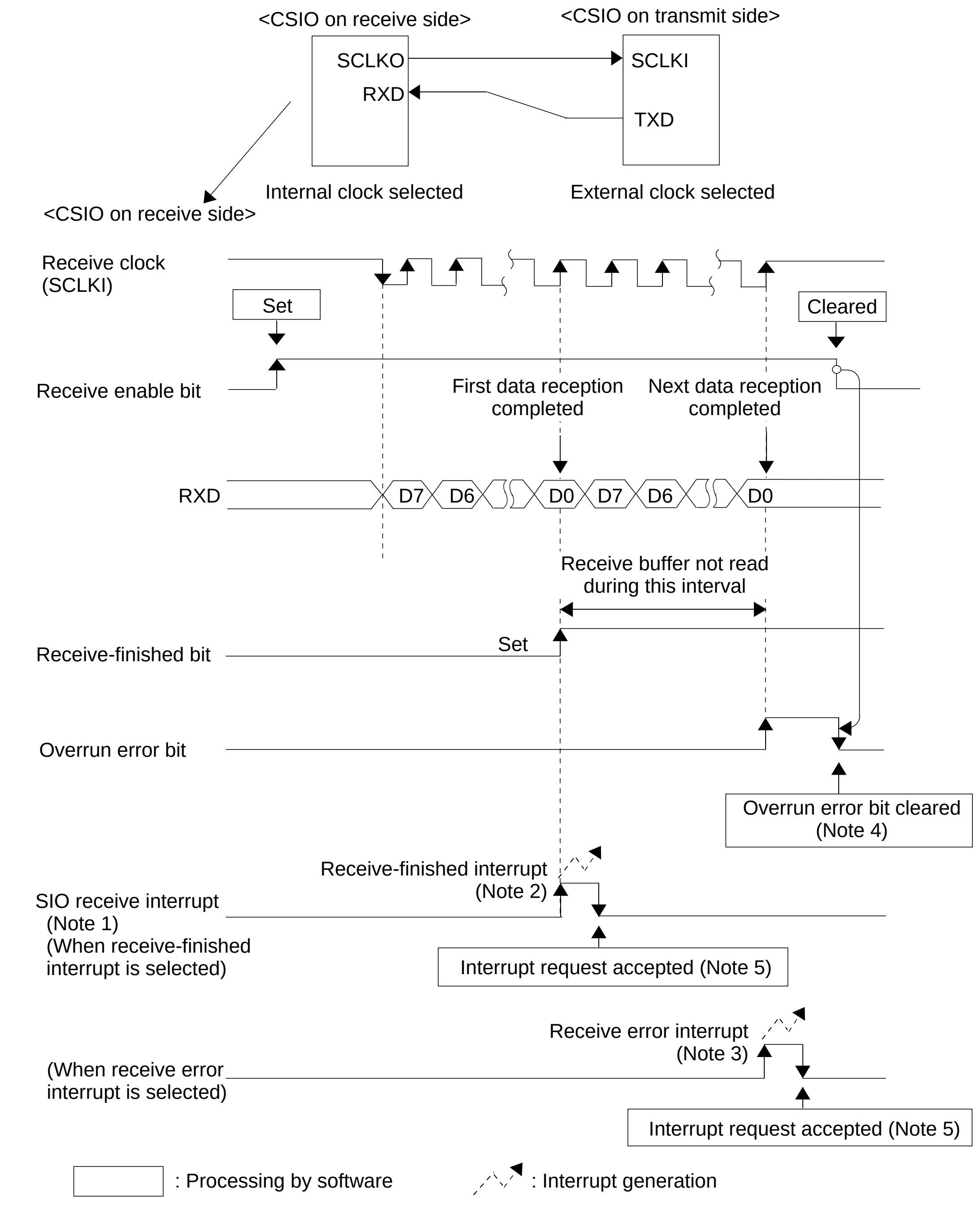


Note 1 : Change of the Interrupt Controller "SIO Receive Interrupt Control Register" interrupt request bit

Note 2 : When receive-finished interrupt is enabled (DMA transfer can also be requested at the same timing)

Note 3 : The Interrupt Controller IVECT register is read or "SIO Receive Interrupt Control Register" interrupt request bit cleared

**Figure 12.4.3 Example of CSIO Reception (When Received Normally)**



Note 1 : Change of the Interrupt Controller "SIO Receive Interrupt Control Register" interrupt request bit

Note 2 : When receive-finished interrupt is enabled

Note 3 : When receive error interrupt is enabled

Note 4 : Receive enable bit cleared

Note 5 : The Interrupt Controller IVECT register is read or "SIO Receive Interrupt Control Register" interrupt request bit cleared

**Figure 12.4.4 Example of CSIO Reception (When Overrun Error Occurred)**

## 12.5 Precautions on Using CSIO Mode

- **Settings of SIO Transmit/Receive Mode Register and SIO Baud Rate Register**

  The SIO Transmit/Receive Mode Register and SIO Baud Rate Register and the Transmit Control Register's BRG count source select bit must always be set when not operating. When transmitting or receiving data, be sure to check that transmission and/or reception under way has been completed and clear the transmit and receive enable bits before you set the registers.

- **Settings of Baud Rate (BRG) Register**

  If you selected f(BCLK) with the BRG clock source select bit, make sure the BRG register value you set does not exceed 2 Mbps.

- **About successive transmission**

  To transmit multiple data successively, set the next transmit data in the SIO Transmit Buffer Register before transmission of the preceding data is completed.

- **About reception**

  Because during CSIO mode the receive shift clock is derived from operation of the transmit circuit, you need to execute transmit operation (by sending dummy data) even when you only want to receive data. In this case, note that if the port function is set for TXD pin (by setting the operation mode register to 1), dummy data is actually output from the pin.

- **About successive reception**

  To receive multiple data successively, set data (dummy data) in the SIO Transmit Buffer Register before the transmitter starts sending data.

- **Transmit/receive operations using DMA**

  To transmit/receive data in DMA request mode, enable the DMAC to accept transfer requests (by setting the DMA Mode Register) before you start serial communication.

- **About the receive-finished bit**

  If a receive error (overrun error) occurs, the receive-finished bit cannot be cleared by reading out the receive buffer register. In this case, it can only be cleared by clearing the receive enable bit.

- **About overrun error**

If all bits of the next receive data are received in the SIO Receive Shift Register before you read out the SIO Receive Buffer Register (an overrun error occurs), the receive data is not stored in the Receive Buffer Register and the Receive Buffer Register retains the previously received data. Thereafter, although receive operation is continued, no receive data is stored in the Receive Buffer Register (the receive status bit = 1). To restart reception normally, you need to temporarily clear the receive enable bit before you restart. This is the only way you can clear the overrun error flag.

- **About DMA transfer request generation during SIO transmission**

If the Transmit Buffer Register becomes empty (the transmit buffer empty flag = 1) while the transmit enable bit is set to 1 (transmit enabled), an SIO transmit buffer empty DMA transfer request is generated.

- **About DMA transfer request generation during SIO reception**

When the receive-finished bit is set to 1 (the receive buffer register full), a receive-finished DMA transfer request is generated. However, if an overrun error has occurred, this DMA transfer request is not generated.

## 12.6 Transmit Operation in UART Mode

### 12.6.1 Setting the UART Baud Rate

The baud rate (data transfer rate) during UART mode is determined by a transmit/receive shift clock. In UART mode, the source for this transmit/receive shift clock is always the internal clock regardless of how the internal/external clock select bit (SIO Transmit/Receive Mode Register bit D11) is set.

**(1) Calculating the UART mode baud rate**

After being divided by the clock divider, f(BCLK) is fed into the Baud Rate Generator (BRG), after which it is further divided by 16 to produce a transmit/receive shift clock. The clock divider's divide-by value is selected from 1, 8, 32, or 256 using the SIO Transmit Control Register's CDIV (baud rate generator count source select) bits (D2, D3). The Baud Rate Generator divides the clock it received from the clock divider by (baud rate register set value + 1) and further divides the resulting clock by 16 to produce a transmit/receive shift clock.

During UART mode (in which the internal clock is always used), the baud rate is calculated using the equation below.

$$\underset{\text{[bps]}}{\text{Baud rate}} = \frac{\text{f (BCLK)}}{\text{Clock divider's divide-by value} \times \text{(baud rate register set value + 1)} \times 16}$$

Baud rate register set value = H'00 to H'FF
Clock divider's divide-by value = 1, 8, 32, or 256

### 12.6.2 UART Transmit/Receive Data Formats

The transmit/receive data format during UART mode is determined by setting the SIO Transmit/ Receive Mode Register. Shown below is the transmit/receive data format that can be used in UART mode.

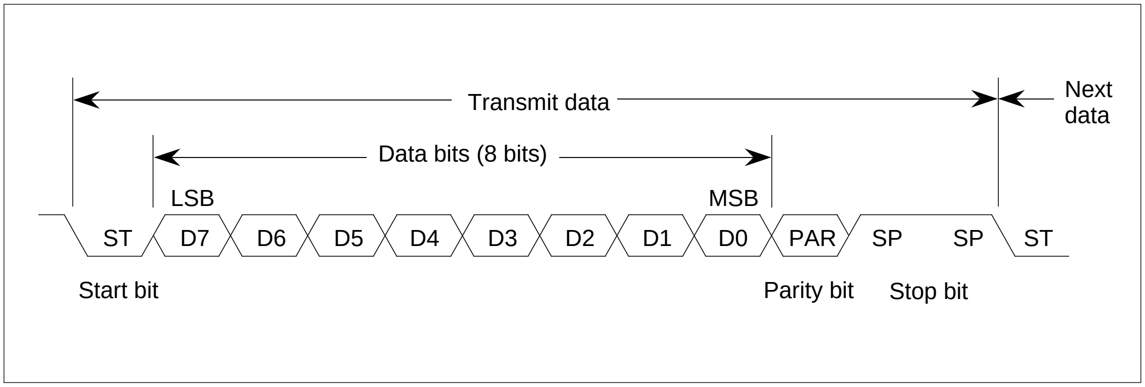
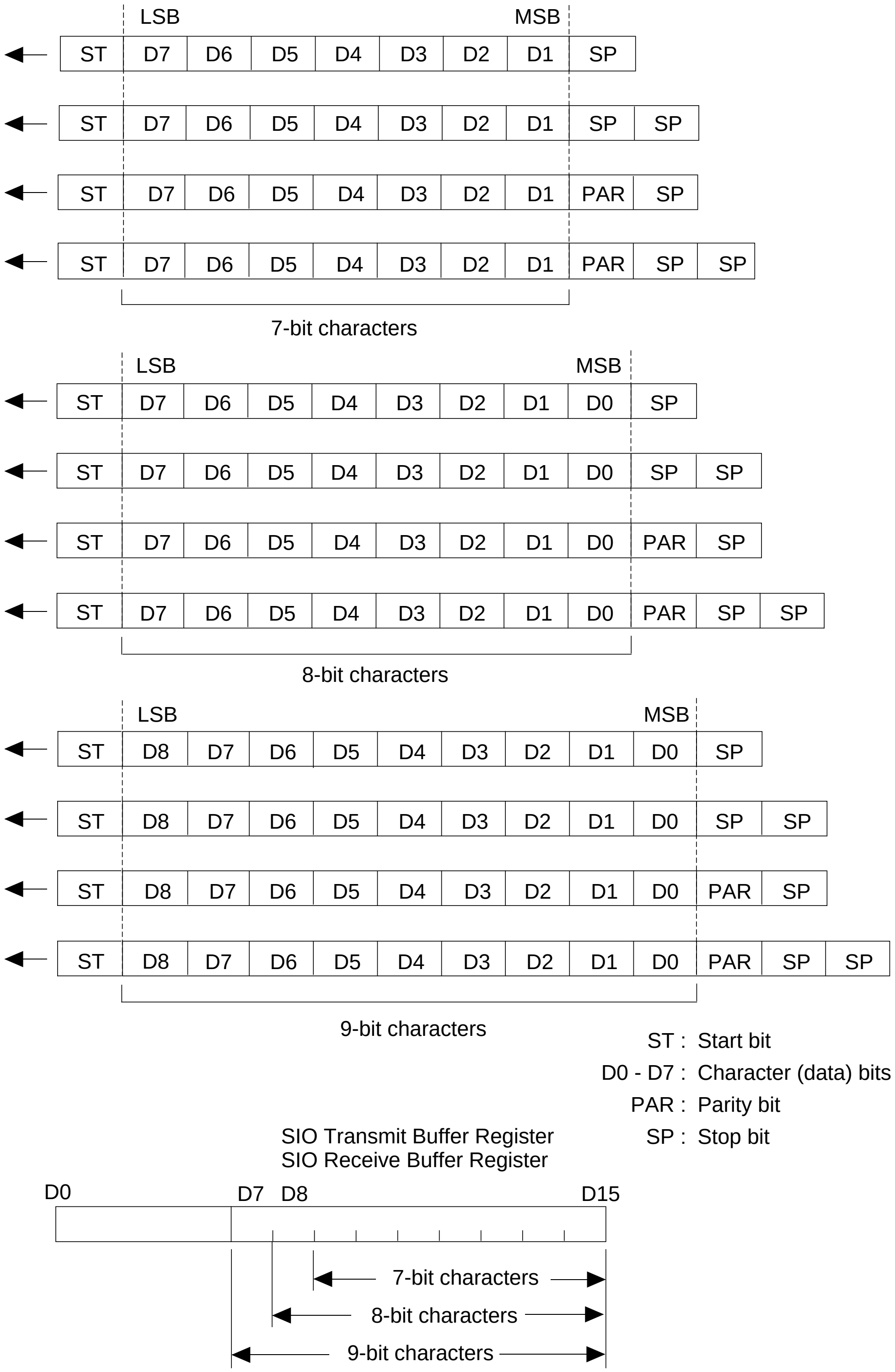


**Figure 12.6.1 Example of Transmit/Receive Data Format in UART Mode**

**Table 12.6.1 Transfer Data in UART Mode**

| Bit Name | Content |
|---|---|
| ST (start bit) | Indicates the beginning of data transmission. This is a low signal of a one bit duration, which is added immediately before the transmit data. |
| D0-D8 (character bits) | Transmit/receive data transferred via serial I/O. In UART mode, data in 7, 8, or 9 bits can be transmitted/received. |
| PAR (parity bit) | Added to the transmit/receive characters. When parity is enabled, parity is automatically set in such a way that the number of 1's in characters including the parity bit itself is always even or odd as selected by the even/odd parity select bit. |
| SP (stop bit) | Indicates the end of data transmission, and is added immediately after characters (or if parity enabled, immediately after the parity bit). The stop bit can be chosen to be one bit or two bits long. |


LSB
MSB
ST D7 D6 D5 D4 D3 D2 D1 SP
ST D7 D6 D5 D4 D3 D2 D1 SP SP
ST D7 D6 D5 D4 D3 D2 D1 PAR SP
ST D7 D6 D5 D4 D3 D2 D1 PAR SP SP
7-bit characters
LSB
MSB
ST D7 D6 D5 D4 D3 D2 D1 D0 SP
ST D7 D6 D5 D4 D3 D2 D1 D0 SP SP
ST D7 D6 D5 D4 D3 D2 D1 D0 PAR SP
ST D7 D6 D5 D4 D3 D2 D1 D0 PAR SP SP
8-bit characters
LSB
MSB
ST D8 D7 D6 D5 D4 D3 D2 D1 D0 SP
ST D8 D7 D6 D5 D4 D3 D2 D1 D0 SP SP
ST D8 D7 D6 D5 D4 D3 D2 D1 D0 PAR SP
ST D8 D7 D6 D5 D4 D3 D2 D1 D0 PAR SP SP
9-bit characters
ST : Start bit
D0 - D7 : Character (data) bits
PAR : Parity bit
SP : Stop bit
SIO Transmit Buffer Register
SIO Receive Buffer Register
D0
D7
D8
D15
7-bit characters
8-bit characters
9-bit characters

Notes : • The high-order bits of the SIO Receive Buffer Register's selected character bits are fixed to 0.

• The data bit numbers (Dn) above indicate bit numbers in a data list, and not the register bit numbers (Dn).

**Figure 12.6.2 Selectable Data Formats during UART Mode**

### 12.6.3 Initial Settings for UART Transmission

To transmit data in UART mode, initialize the serial I/O following the procedure described below.

**(1) Setting SIO Transmit/Receive Mode Register**

- Set the register to UART mode
- Set parity (when enabled, select odd/even)
- Set stop bit length
- Set character length

Note : • During UART mode, settings of the internal/external clock select bit have no effect (only the internal clock is useful).

**(2) Setting SIO Transmit Control Register**

Select the clock divider's divide-by ratio.

**(3) Setting SIO Baud Rate Register**

Set a baud rate generator value. (Refer to Section 12.6.1, "Setting the UART Baud Rate.")

**(4) Setting SIO Interrupt Mask Register**

- Enable or disable SIO transmit interrupt

**(5) Setting the Interrupt Controller (SIO Transmit Interrupt Control Register)**

When you use a transmit interrupt, set its priority level.

**(6) Setting DMAC**

When you issue DMA transfer requests to the internal DMAC when the transmit buffer is empty, set the DMAC. (Refer to Chapter 9, "DMAC.")

**(7) Selecting pin functions**

Because the serial I/O related pins serve dual purposes (shared with input/output ports), set pin functions. (Refer to Chapter 8, "Input/Output Ports and Pin Functions.")

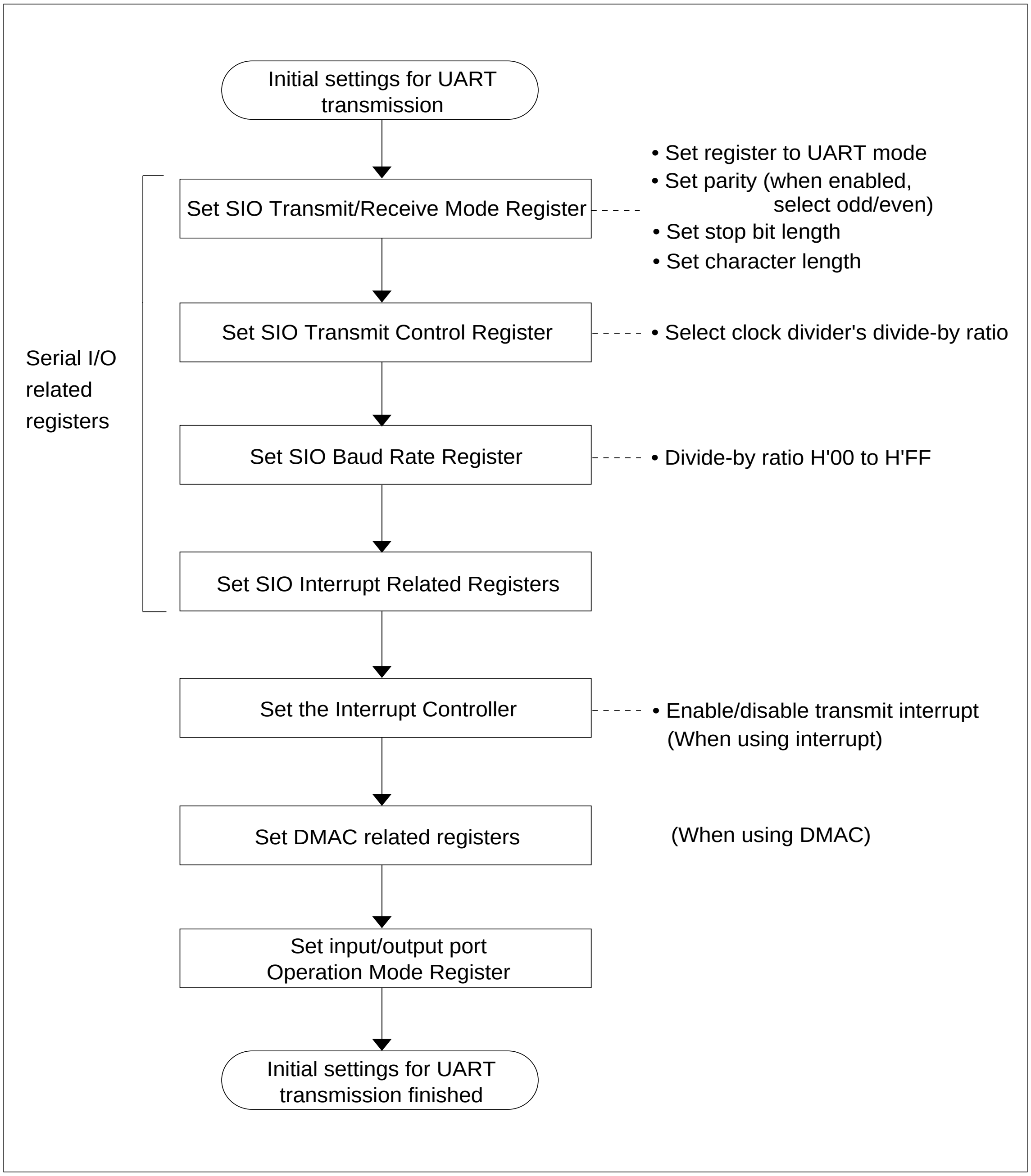

Initial settings for UART transmission
Set SIO Transmit/Receive Mode Register
• Set register to UART mode
• Set parity (when enabled, select odd/even)
• Set stop bit length
• Set character length
Set SIO Transmit Control Register
• Select clock divider's divide-by ratio
Serial I/O related registers
Set SIO Baud Rate Register
• Divide-by ratio H'00 to H'FF
Set SIO Interrupt Related Registers
Set the Interrupt Controller
• Enable/disable transmit interrupt (When using interrupt)
Set DMAC related registers
(When using DMAC)
Set input/output port Operation Mode Register
Initial settings for UART transmission finished

**Figure 12.6.3 Procedure for UART Transmit Initialization**

### 12.6.4 Starting UART Transmission

When all of the following transmit conditions are met after you finished initialization, the serial I/O starts transmit operation.

- The SIO Transmit Control Register's TEN (transmit enable) bit is set to 1. (Note 1)
- Transmit data is written to the SIO Transmit Buffer Register (transmit buffer empty bit = 0).

Note 1: While the transmit enable bit is cleared to 0, writes to the transmit buffer are ignored. Always be sure to set the transmit enable bit to 1 before you write to the transmit buffer register.

When transmission starts, the serial I/O transmits data following the procedure below.

- Transfer the content of the SIO Transmit Buffer Register to the SIO Transmit Shift Register.
- Set the transmit buffer empty bit to 1. (Note 2)
- Start sending data synchronously with the shift clock beginning with the LSB.

Note 2: A transmit buffer empty interrupt request and/or a DMA transfer request can be generated when the transmit buffer is emptied.

### 12.6.5 Successive UART Transmission

Once data is transferred from the transmit buffer register to the transmit shift register, the next data can be written to the transmit buffer register even when transmission of the preceding data is not completed. When the next data is written to the transmit buffer before completion of the preceding data transmission, the preceding and the next data are successively transmitted.

To see if data has been transferred from the transmit buffer register to the transmit shift register, check the SIO Transmit Control Register's transmit buffer empty flag.

### 12.6.6 Processing at End of UART Transmission

When data transmission is completed, the following operation is automatically performed in hardware.

**(1) When not transmitting successively**

- The transmit status bit is set to 0.

**(2) When transmitting successively**

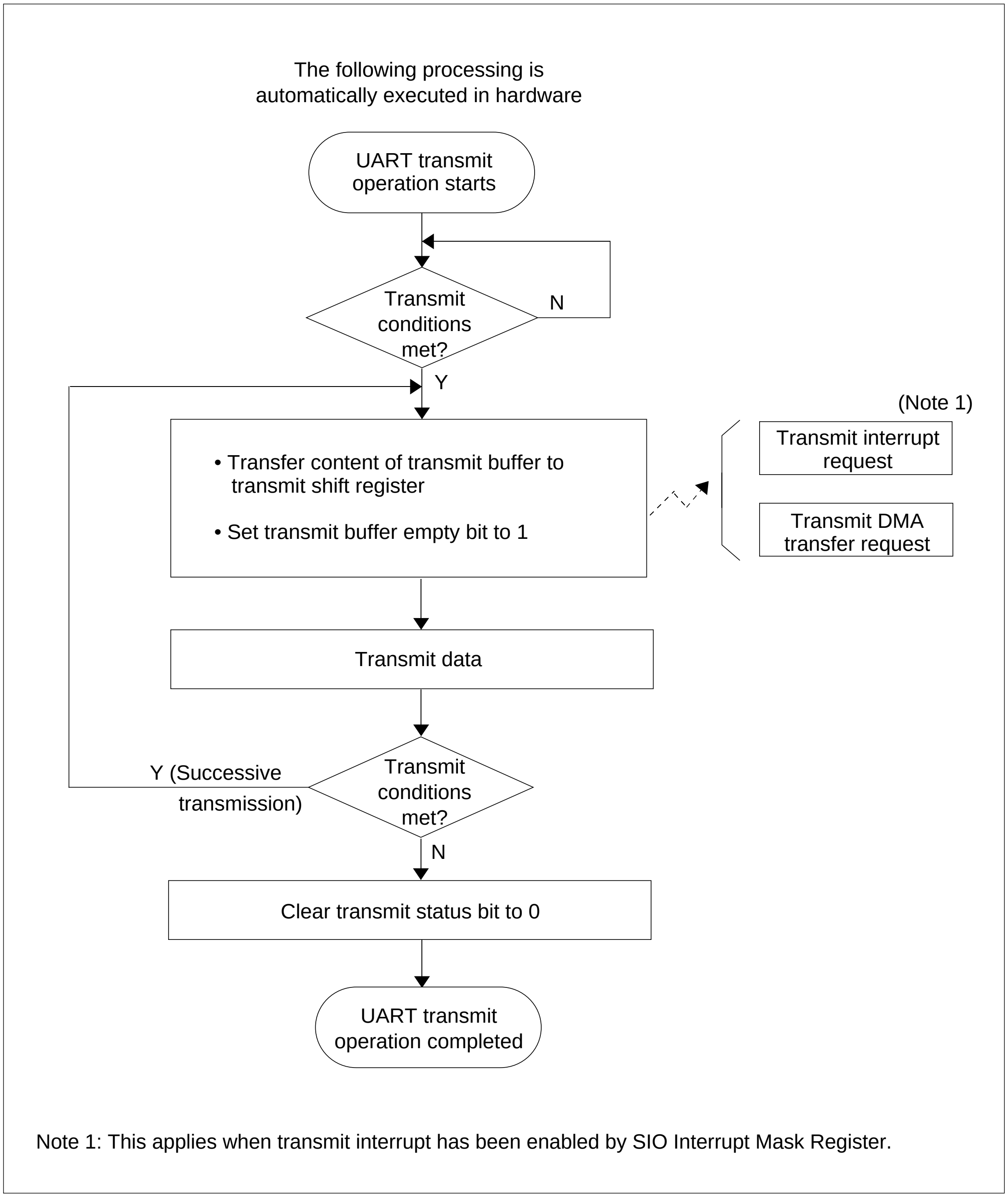
- When transmission of the last data in a consecutive data train is completed, the transmit status bit is set to 0.

### 12.6.7 Transmit Interrupt

If a transmit buffer empty interrupt has been enabled by the SIO Interrupt Mask Register, a transmit buffer empty interrupt is generated at the time data is transferred from the transmit buffer register to the transmit shift register. Also, a transmit buffer empty interrupt is generated when the TEN (transmit enable) bit is set to 1 (enabled after being disabled) while a transmit buffer empty interrupt has been enabled.
You must set the Interrupt Controller (ICU) before you can use transmit interrupts.

### 12.6.8 Transmit DMA Transfer Request

When data has been transferred from the transmit buffer register to the transmit shift register, a transmit DMA transfer request for the corresponding SIO channel is ouput to the DMAC. This transfer request is also output when the TEN (transmit enable) bit is set to 1 (enabled after being disabled).
You must set the DMAC before you can transmit data using DMA transfers.

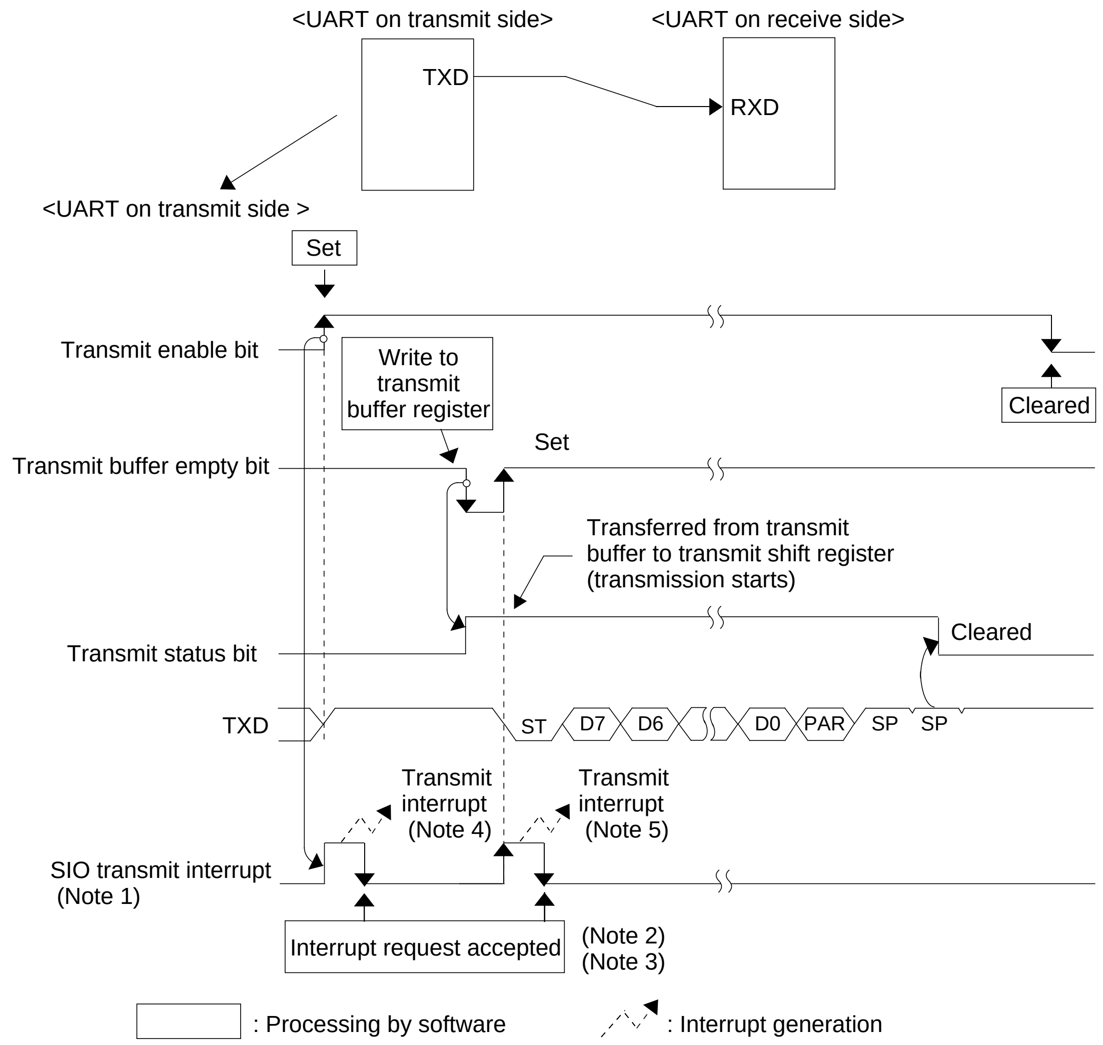


Note 1: This applies when transmit interrupt has been enabled by SIO Interrupt Mask Register.

**Figure 12.6.4 Transmit Operation during UART Mode (Hardware Processing)**

### 12.6.9 Typical UART Transmit Operation

The following shows a typical transmit operation in CSIO mode.


<UART on transmit side>
<UART on receive side>
TXD
RXD
<UART on transmit side >
Set
Transmit enable bit
Write to transmit buffer register
Cleared
Transmit buffer empty bit
Set
Transferred from transmit buffer to transmit shift register (transmission starts)
Transmit status bit
Cleared
TXD
ST
D7
D6
D0
PAR
SP
SP
Transmit interrupt (Note 4)
Transmit interrupt (Note 5)
SIO transmit interrupt (Note 1)
Interrupt request accepted
(Note 2)
(Note 3)
: Processing by software
: Interrupt generation

Note 1 : Change of the Interrupt Controller "SIO Transmit Interrupt Control Register" interrupt request bit

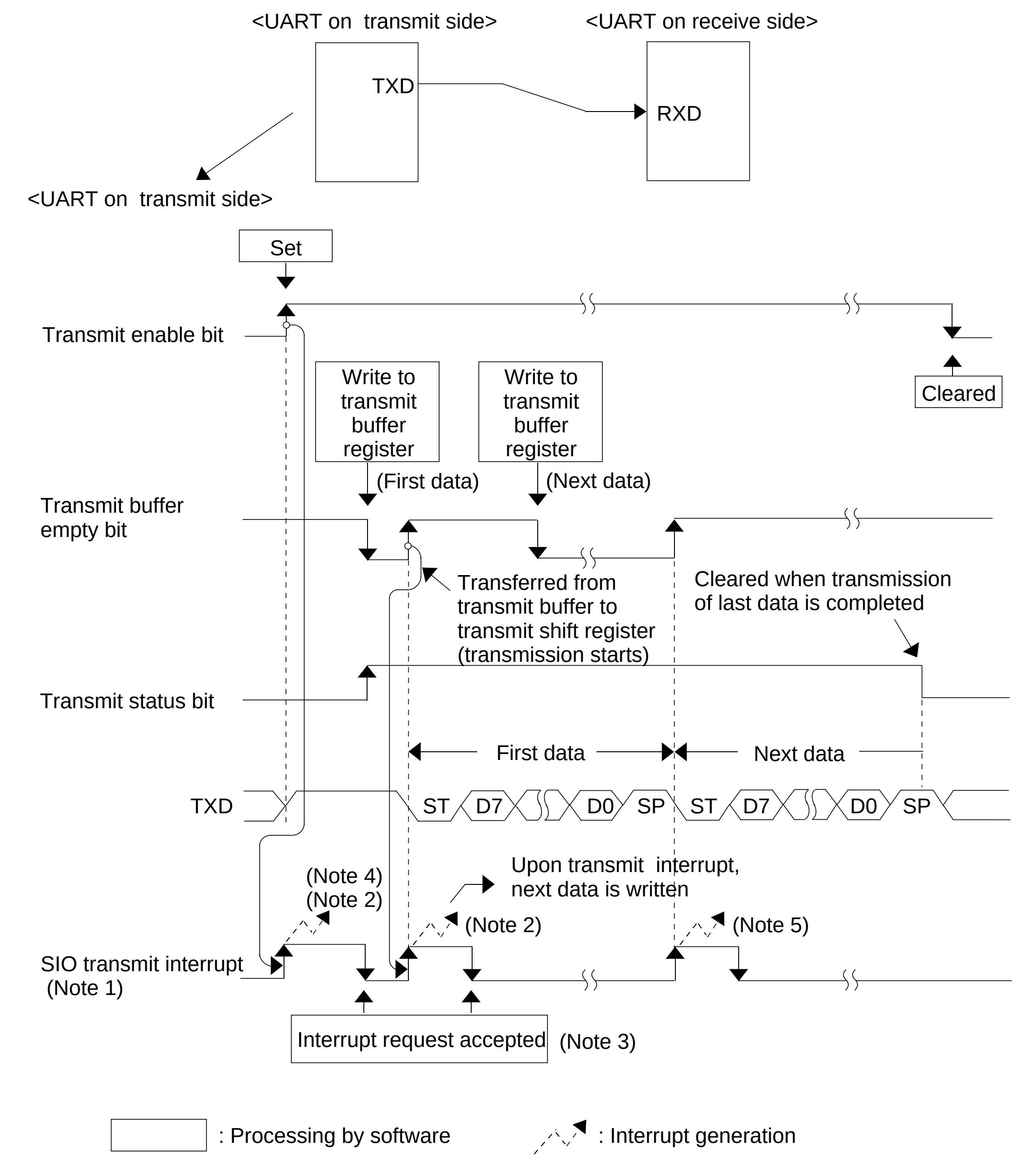
Note 2 : When transmit-finished interrupt is enabled (DMA transfer can also be requested at the same timing)

Note 3 : The Interrupt Controller IVECT register is read or "SIO Transmit Interrupt Control Register" interrupt request bit cleared

Note 4 : Transmit interrupt request is generated when transmission is enabled.

Note 5 : Even after transmit data is written to the transmit buffer, a transmit interrupt request is generated when the data is transferred from the transmit buffer to the transmit shift register and the transmit buffer is thereby emptied.

**Figure 12.6.5 Example of UART Transmission (Transmitted Only Once, with Transmit Interrupt Used)**


<UART on transmit side>
<UART on receive side>
TXD
RXD
<UART on transmit side>
Set
Transmit enable bit
Write to transmit buffer register
Write to transmit buffer register
Cleared
(First data)
(Next data)
Transmit buffer empty bit
Transferred from transmit buffer to transmit shift register (transmission starts)
Cleared when transmission of last data is completed
Transmit status bit
First data
Next data
TXD
ST
D7
D0
SP
ST
D7
D0
SP
Upon transmit interrupt, next data is written
(Note 4)
(Note 2)
(Note 2)
(Note 5)
SIO transmit interrupt (Note 1)
Interrupt request accepted
(Note 3)
: Processing by software
: Interrupt generation

Note 1 : Change of the Interrupt Controller "SIO Transmit Interrupt Control Register" interrupt request bit

Note 2 : When transmit buffer empty interrupt is enabled (DMA transfer can also be requested at the same timing)

Note 3 : The Interrupt Controller IVECT register is read or "SIO Transmit Interrupt Control Register" interrupt request bit cleared

Note 4 : Transmit interrupt request is generated when transmission is enabled.

Note 5 : Even after transmit data is written to the transmit buffer, a transmit interrupt request is generated when the data is transferred from the transmit buffer to the transmit shift register and the transmit buffer is thereby emptied.

**Figure 12.6.6 Example of UART Transmission (Successive Transmission, with Transmit Interrupt Used)**

## 12.7 Receive Operation in UART Mode

### 12.7.1 Initial Settings for UART Reception

To receive data in UART mode, initialize the serial I/O following the procedure described below.

**(1) Setting SIO Transmit/Receive Mode Register**

- Set the register to UART mode
- Set parity (when enabled, select odd/even)
- Set stop bit length
- Set character length

Note : • During UART mode, settings of the internal/external clock select bit have no effect (only the internal clock is useful).

**(2) Setting SIO Transmit Control Register**

Select the clock divider's divide-by ratio.

**(3) Setting SIO Baud Rate Register**

Set a baud rate generator value. (Refer to Section 12.6.1, "Setting the UART Baud Rate.")

**(4) Setting SIO interrupt related registers**

- Cause of Receive Interrupt Select Register
  Select the cause of receive interrupt (receive finished/receive error)
- Interrupt Mask Register
  Enable/disable receive interrupts

**(5) Setting the Interrupt Controller**

When you use interrupts during reception, set its priority level.

**(6) Setting DMAC**

When you issue DMA transfer requests to the internal DMAC when reception is completed, set the DMAC. (Refer to Chapter 9, "DMAC.")

**(7) Selecting pin functions**

Because the serial I/O related pins serve dual purposes (shared with input/output ports), set pin functions. (Refer to Chapter 8, "Input/Output Ports and Pin Functions.")

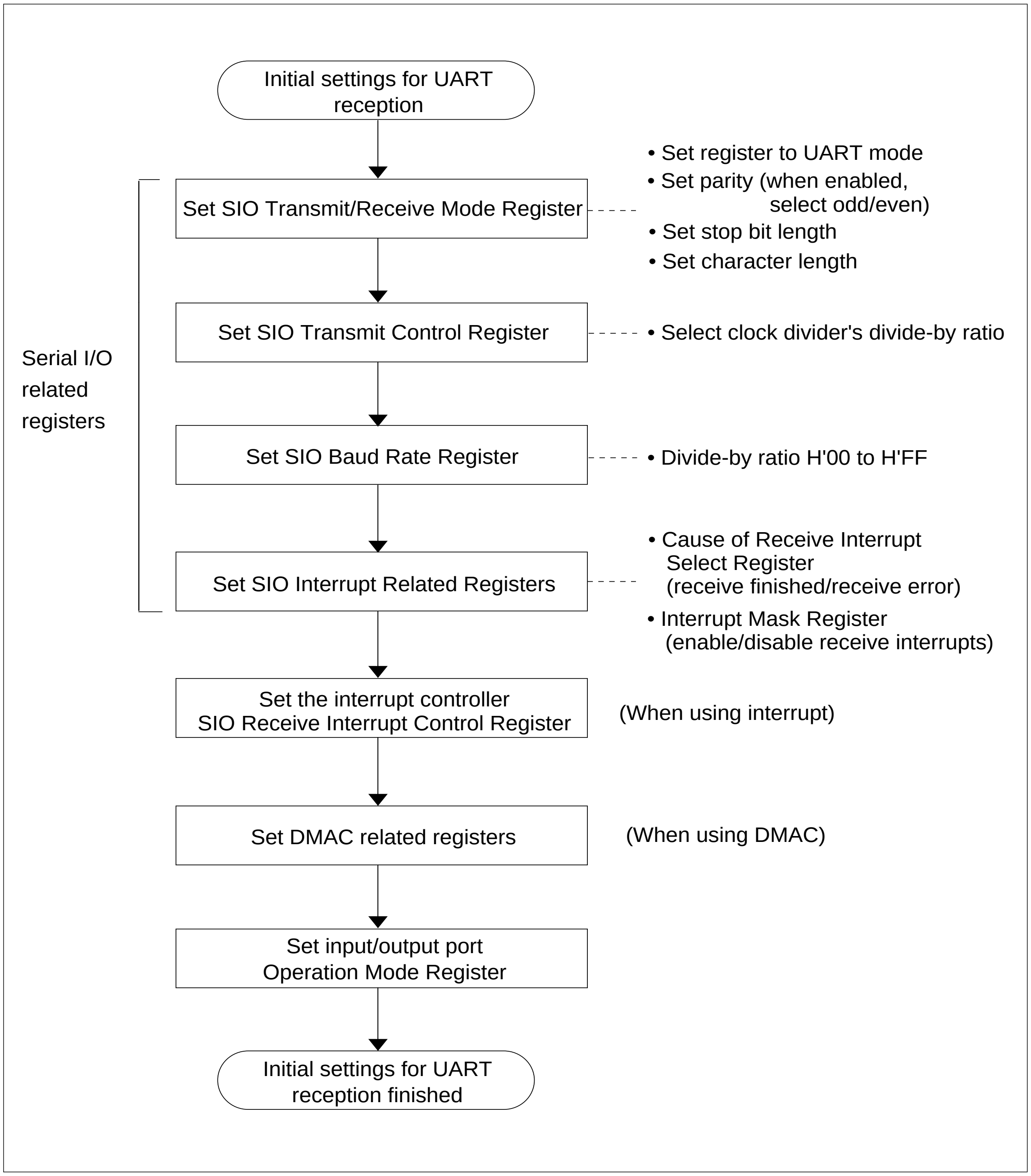

Initial settings for UART reception
Set SIO Transmit/Receive Mode Register
• Set register to UART mode
• Set parity (when enabled, select odd/even)
• Set stop bit length
• Set character length
Set SIO Transmit Control Register
• Select clock divider's divide-by ratio
Serial I/O related registers
Set SIO Baud Rate Register
• Divide-by ratio H'00 to H'FF
Set SIO Interrupt Related Registers
• Cause of Receive Interrupt Select Register (receive finished/receive error)
• Interrupt Mask Register (enable/disable receive interrupts)
Set the interrupt controller SIO Receive Interrupt Control Register
(When using interrupt)
Set DMAC related registers
(When using DMAC)
Set input/output port Operation Mode Register
Initial settings for UART reception finished

**Figure 12.7.1 Procedure for UART Receive Initialization**

### 12.7.2 Starting UART Reception

When all of the following receive conditions are met after you finished initialization, the serial I/O starts receive operation.

- The SIO Receive Control Register's receive enable bit is set to 1
- Start bit (falling edge signal) is applied to the RXD pin

When the above conditions are met, the serial I/O enters UART receive operation. However, if the start bit when checked again at the first rise of the internal receive shift clock is detected high for reason of noise, etc., the serial I/O stops receive operation and waits for the start bit again.

### 12.7.3 Processing at End of UART Reception

When data reception is completed, the following operation is automatically performed in hardware.

**(1) When reception is completed normally**

The receive-finished (receive buffer full) bit is set to 1.

Notes: • If a receive-finished (receive buffer full) interrupt has been enabled, an interrupt request is generated.
• A DMA transfer request is generated.

**(2) When error occurs during reception**

When an error occurs during reception, the corresponding error bit (OE, FE, or PE) and the receive sum bit are set to 1.

Notes: • If a receive-finished interrupt has been selected (by SIO Cause of Receive Interrupt Select Register), a receive-finished interrupt request is generated when interrupt requests are enabled. However, if an overrun error has occurred, this interrupt is not generated.
• If a receive error interrupt has been selected (by SIO Cause of Receive Interrupt Select Register), a receive error interrupt request is generated when interrupt requests are enabled.
• No DMA transfer requests are generated.

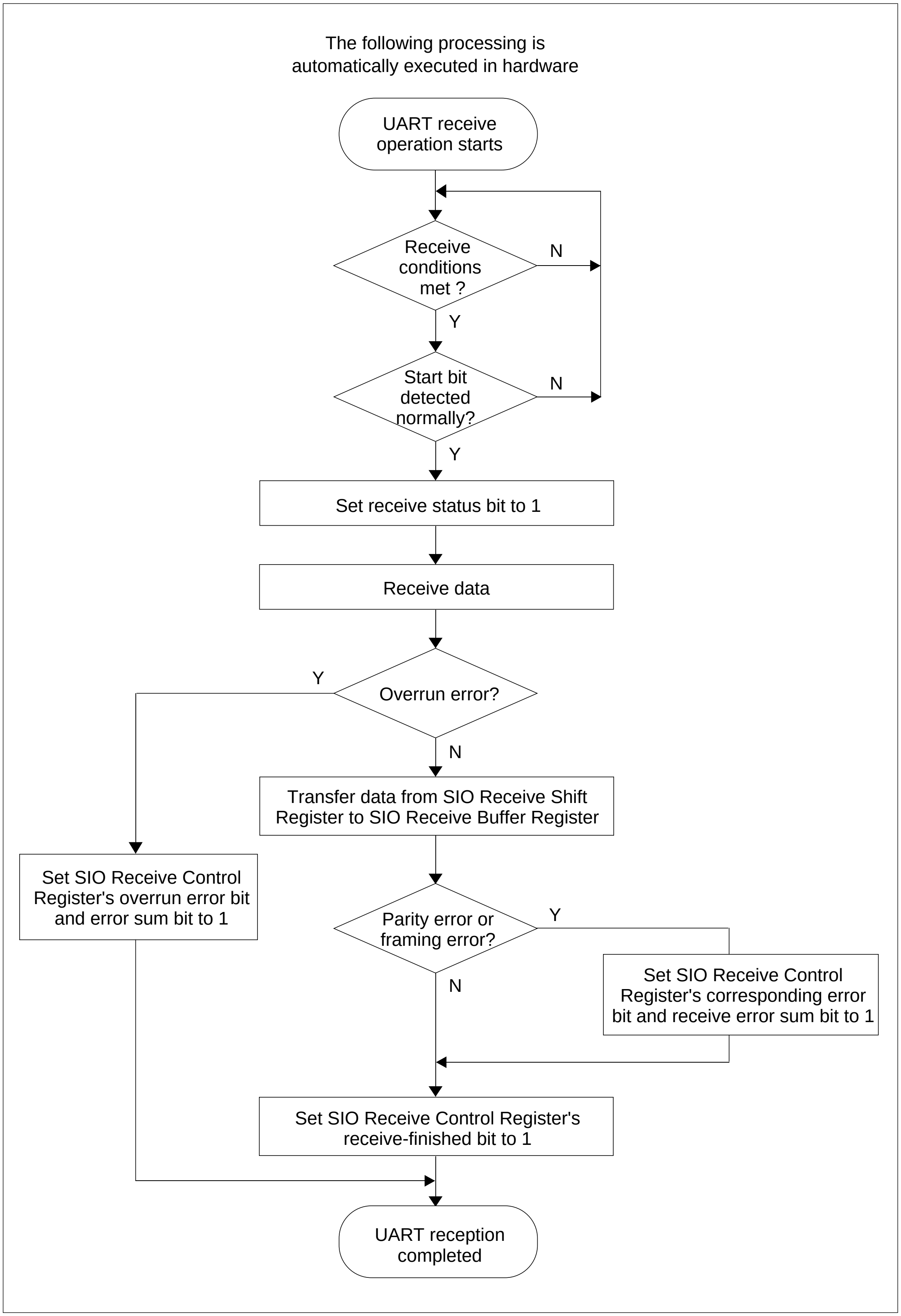
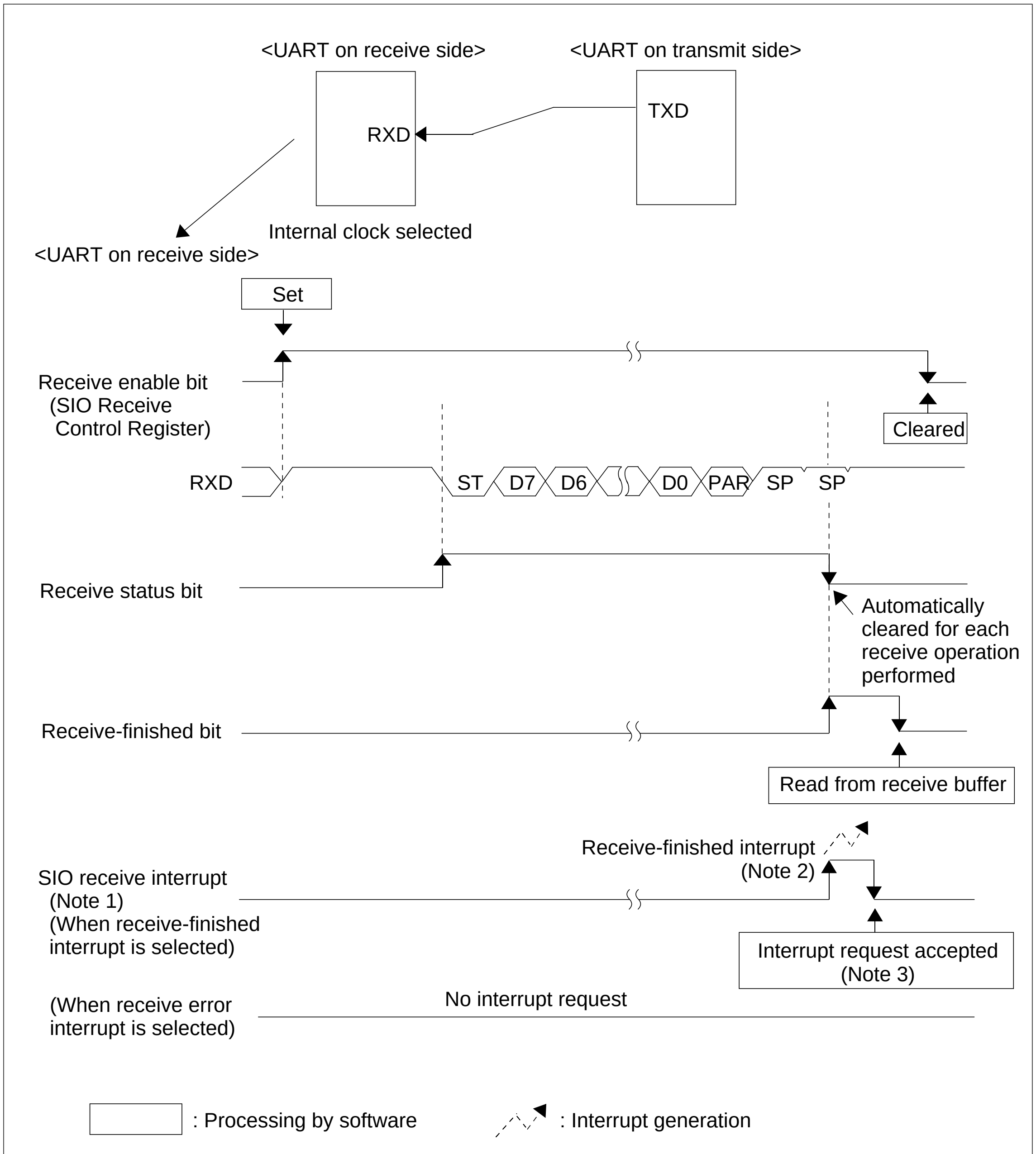

The following processing is
automatically executed in hardware
UART receive
operation starts
Receive
conditions
met ?
N
Y
Start bit
detected
normally?
N
Y
Set receive status bit to 1
Receive data
Y
Overrun error?
N
Transfer data from SIO Receive Shift
Register to SIO Receive Buffer Register
Set SIO Receive Control
Register's overrun error bit
and error sum bit to 1
Parity error or
framing error?
Y
N
Set SIO Receive Control
Register's corresponding error
bit and receive error sum bit to 1
Set SIO Receive Control Register's
receive-finished bit to 1
UART reception
completed

**Figure 12.7.2 Receive Operation during UART Mode (Hardware Processing)**

### 12.7.4 Typical UART Receive Operation

The following shows a typical receive operation in UART mode.

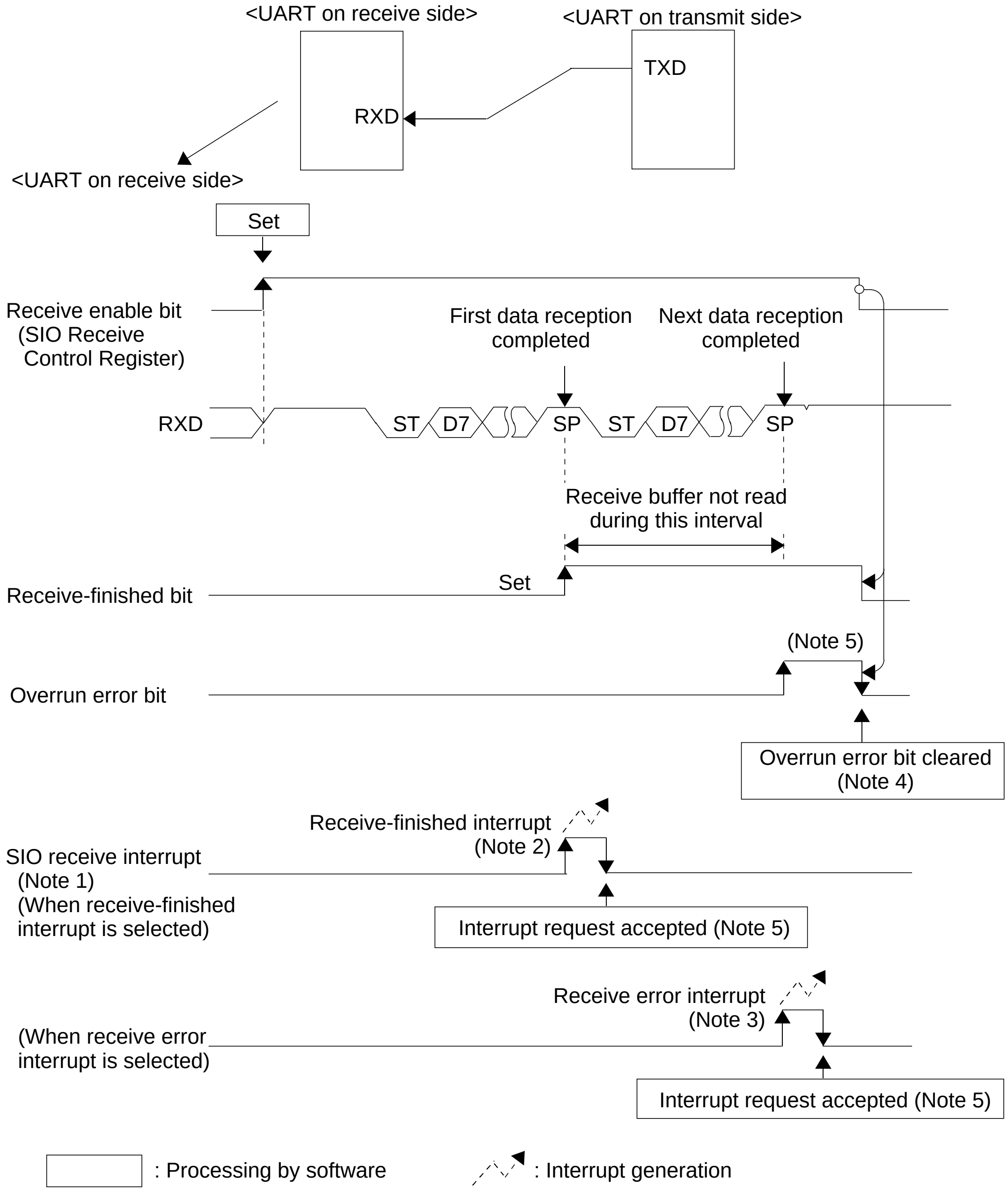


Note 1 : Change of the Interrupt Controller "SIO Receive Interrupt Control Register" interrupt request bit

Note 2 : When receive-finished interrupt is enabled (DMA transfer can also be requested at the same timing)

Note 3 : The Interrupt Controller IVECT register is read or "SIO Receive Interrupt Control Register" interrupt request bit cleared

**Figure 12.7.3 Example of UART Reception (When Received Normally)**



Note 1 : Change of the Interrupt Controller "SIO Receive Interrupt Control Register" interrupt request bit

Note 2 : When receive-finished interrupt is enabled

Note 3 : When receive error interrupt is enabled

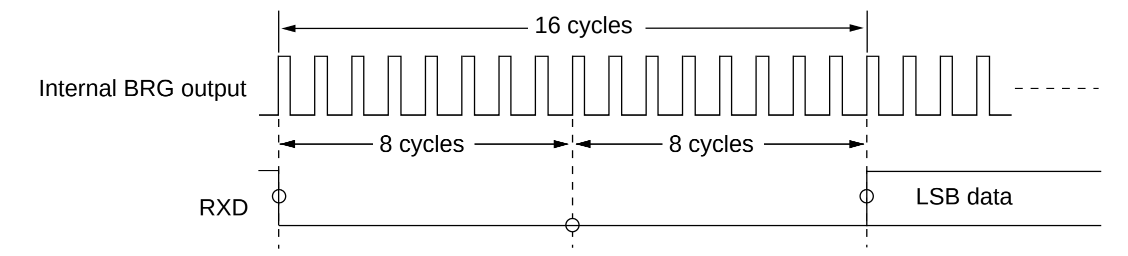
Note 4 : This is done by clearing the receive enable bit to 0.

Note 5 : The Interrupt Controller IVECT register is read or "SIO Receive Interrupt Control Register" interrupt request bit cleared

**Figure 12.7.4 Example of UART Reception (When Overrun Error Occurred)**

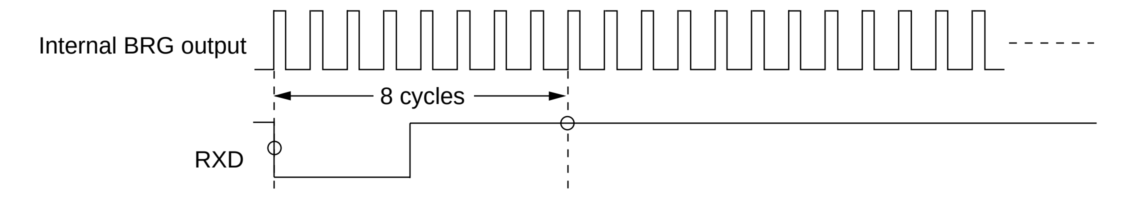
### 12.7.5 Detecting the Start Bit during UART Reception

The start bit is sampled synchronously with the internal BRG output timing. The start bit is detected as valid when RXD is sampled low eight internal BRG output cycles after detecting a falling edge of the start bit, and another eight cycles later the CPU starts latching RXD as the LSB data (first bit data). If RXD is sampled high at the 8th cycle, the CPU again starts detecting a low-going transition of the start bit. Because RXD is sampled synchronously with the internal BRG, there is a delay equal to a BRG output at maximum. Thereafter, RXD is received with the delayed timing.



Note: • This diagram does not include detailed timing information.

Figure 12.7.5 Detecting the Start Bit



Note: • This diagram does not include detailed timing information.

Figure 12.7.6 Example of an Invalid Start Bit (Not Received)

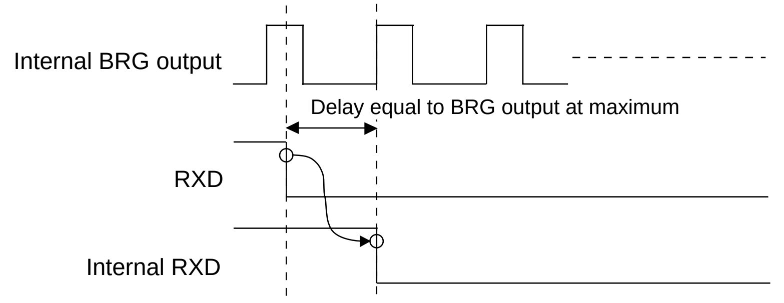


Figure 12.7.7 Delay when Receiving

## 12.8 Fixed Period Clock Output Function

When using SIO0 or SIO1 in UART mode, you can choose the relevant port (P84 or P87) to function as the SCLKO0 or SCLKO1 pin. In this way, a clock derived from BRG output by dividing it by 2 can be output from the SCLKO pin.

Note : • This clock is output not just during data transfer.

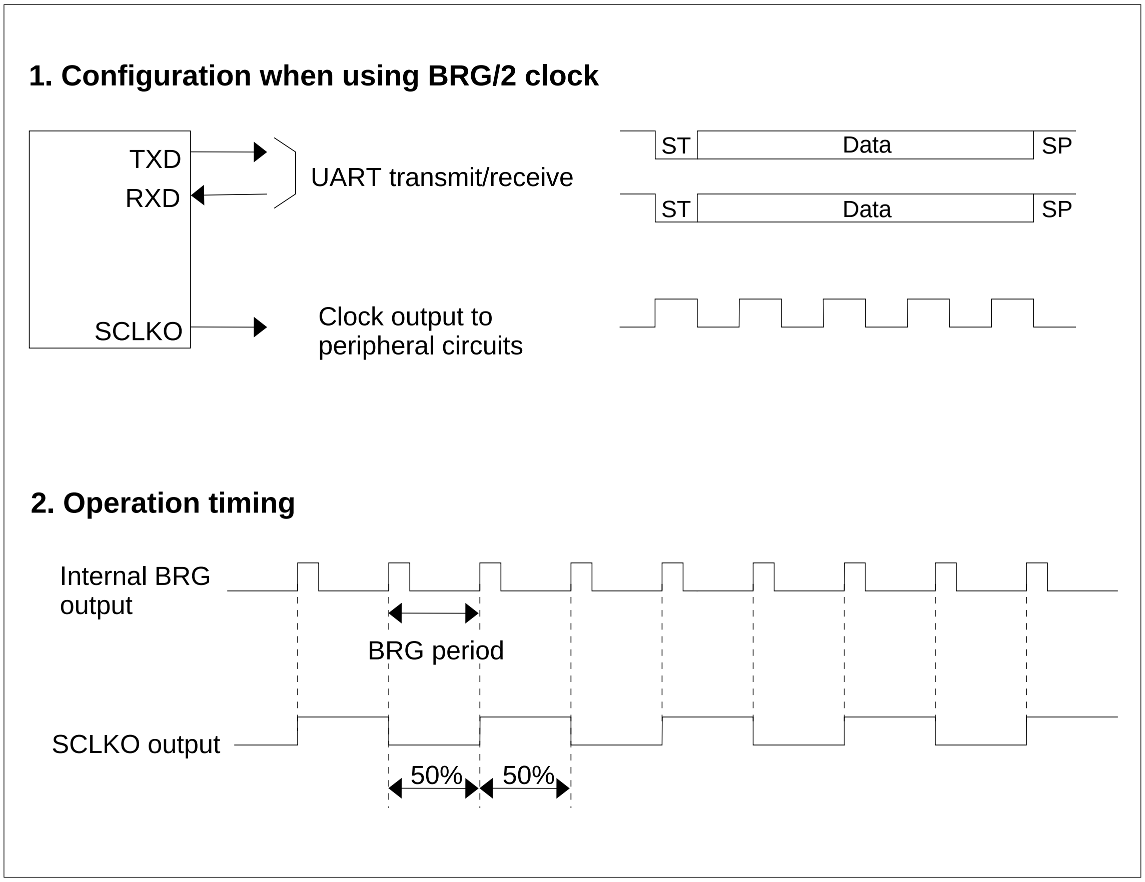


**Figure 12.8.1 Example of Fixed Period Clock Output**

## 12.9 Precautions on Using UART Mode

- **Settings of SIO Transmit/Receive Mode Register and SIO Baud Rate Register**

  The SIO Transmit/Receive Mode Register and SIO Baud Rate Register and the Transmit Control Register's BRG count source select bit must always be set when not operating. When transmitting or receiving data, be sure to check that transmission and/or reception under way has been completed and clear the transmit and receive enable bits before you set the registers.

- **Settings of Baud Rate (BRG) Register**

  The value written to the SIO Baud Rate Register becomes effective beginning with the next period after the BRG counter finished counting. However, when transmit and receive operations are disabled, the register value can be changed at the same time you write to the register.

- **Transmit/receive operations using DMA**

  To transmit/receive data in DMA request mode, enable the DMAC to accept transfer requests (by setting the DMA Mode Register) before you start serial communication.

- **About overrun error**

  If all bits of the next receive data are received in the SIO Receive Shift Register before you read out the SIO Receive Buffer Register (an overrun error occurs), the receive data is not stored in the Receive Buffer Register and the Receive Buffer Register retains the previously received data. Once an overrun error occurs, no receive data is stored in the Receive Buffer Register although receive operation is continued. To restart reception normally, you need to temporarily clear the receive enable bit before you restart. This is the only way you can clear the overrun error flag.

## • Flags indicating the status of UART receive operation

Following flags are available that indicate the status of receive operation during UART mode.

- SIO Receive Control Register receive status bit
- SIO Receive Control Register receive-finished bit
- SIO Receive Control Register receive error sum bit
- SIO Receive Control Register overrun error bit
- SIO Receive Control Register parity error bit
- SIO Receive Control Register framing error bit

The manner in which the receive-finished bit and various error bit flags are cleared varies depending on whether an overrun error has occurred or not, as described below.

**[When no overrun error has occurred]**

Said bits can be cleared by reading the lower byte from the receive buffer register or clearing the receive enable bit to 0.

**[When an overrun error has occurred]**

Said bits can only be cleared by clearing the receive enable bit to 0.

* This is a blank page. *

# CHAPTER 13
# CAN MODULE

13.1 Outline of the CAN Module
13.2 CAN Module Related Registers
13.3 CAN Protocol
13.4 Initializing the CAN Module
13.5 Transmitting Data Frames
13.6 Receiving Data Frames
13.7 Transmitting Remote Frames
13.8 Receiving Remote Frames

## 13.1 Outline of the CAN Module

The 32171 contains CAN (Controller Area Network) Specification 2.0B active-compliant Full CAN module. This module has 16 message slots and three mask registers, effective use of which helps to reduce the CPU load for data processing.

The following outlines the Full CAN module.

**Table 13.1.1 Outline of the CAN Module**

| Item | Content |
|---|---|
| Protocol | CAN Specification 2.0B acvtive |
| Number of message slots | Total 16 slots (14 global slots, two local slots) |
| Polarity | 0: Dominant<br>1: Recessive |
| Acceptance filter | One global mask (Function to receive ID in only a specified range by using receive ID filter)<br>Two local masks |
| Baud rate | 1 Time quantum (Tq) = (BRP + 1)/CPU clock<br>(BRP: Baud rate prescaler set value)<br>$\text{Baud rate} = \dfrac{1}{\text{Tq period x number of Tq's for one bit}}$ ··· Max 1 Mbps (Note 1)<br>BRP :1-255 (0: Inhibited)<br>Number of Tq's for one bit = Synchronization Segment +<br>Propagation Segment +<br>Phase Segment 1 +<br>Phase Segment 2 +<br>Progagation Segment : 1-8Tq<br>Phase Segment 1 : 1-8Tq<br>Phase Segment 2 : 2-8Tq (IPT = 2) |
| Remote frame automatic response function | A slot which received a remote frame automatically sends a data frame. |
| Time stamp function | Time stamp function implemented by a 16-bit counter. Using CAN bus bit period as the fundamental period, a count period can be set to 1/1 through 1/4 of it. |
| BasicCAN mode | BasicCAN function is materialized using two local slots. |
| Transmit abort function | Transmit request can be canceled. |
| Loopback function | The data transmitted by CAN module itself is received. |
| Return bus off function | Forcibly placed into error active mode after clearing error counter. |

Note 1: The maximum baud rate depends on the system configuration (e.g., bus length, clock error, CAN bus transceiver, sampling position, and bit configuration).

Table 13.1.2 CAN Module Interrupt Generation Function

| CAN module interrupt source | ICU interrupt source |
|---|---|
| CAN0 transmit complete interrupt | CAN0 Transmit/Receive & Error interrupt |
| CAN0 receive complete interrupt | CAN0 Transmit/Receive & Error interrupt |
| CAN0 bus error interrupt | CAN0 Transmit/Receive & Error interrupt |
| CAN0 error passive interrupt | CAN0 Transmit/Receive & Error interrupt |
| CAN0 bus off interrupt | CAN0 Transmit/Receive & Error interrupt |

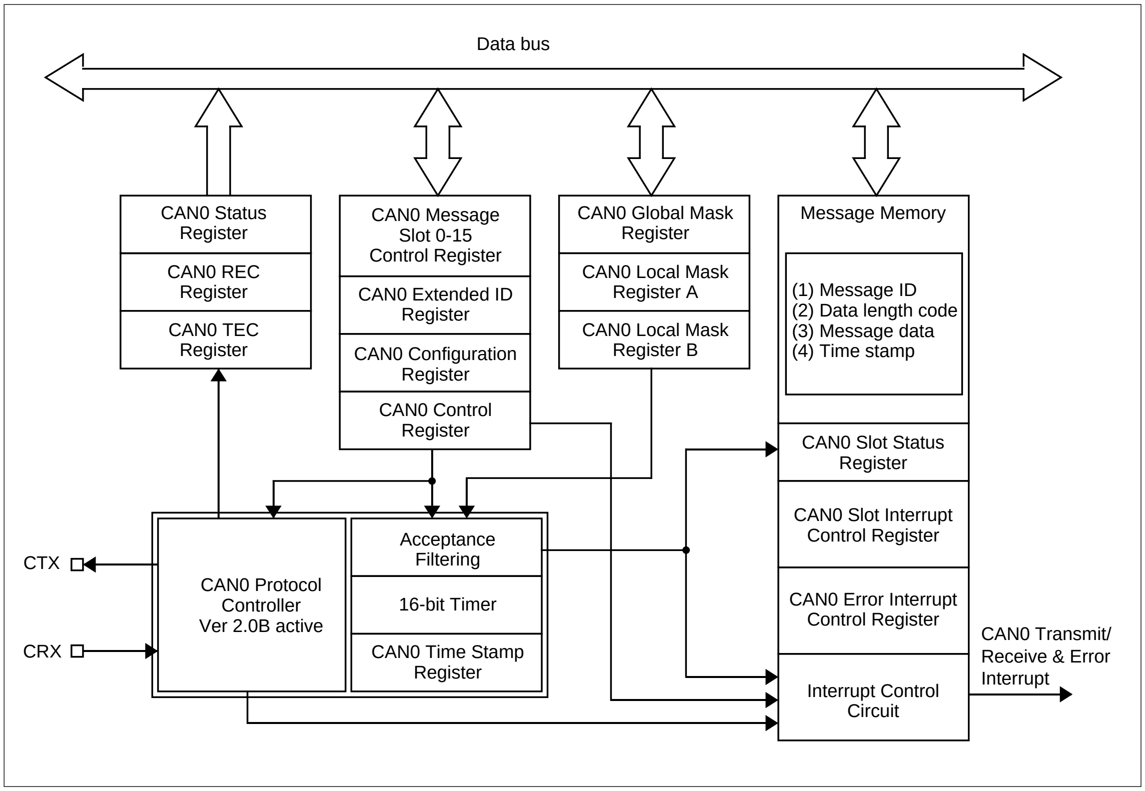

Data bus
CAN0 Status Register
CAN0 REC Register
CAN0 TEC Register
CAN0 Message Slot 0-15 Control Register
CAN0 Extended ID Register
CAN0 Configuration Register
CAN0 Control Register
CAN0 Global Mask Register
CAN0 Local Mask Register A
CAN0 Local Mask Register B
Message Memory
(1) Message ID
(2) Data length code
(3) Message data
(4) Time stamp
CAN0 Slot Status Register
CAN0 Slot Interrupt Control Register
CAN0 Error Interrupt Control Register
Interrupt Control Circuit
CAN0 Transmit/ Receive & Error Interrupt
CTX
CRX
CAN0 Protocol Controller Ver 2.0B active
Acceptance Filtering
16-bit Timer
CAN0 Time Stamp Register

Figure 13.1.1 Block Diagram of the CAN Module

## 13.2 CAN Module Related Registers

The diagram below shows a CAN module related register map.

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 1000 | CAN0 Control Register (CAN0CNT) | |
| H'0080 1002 | CAN0 Status Register (CAN0STAT) | |
| H'0080 1004 | CAN0 Extended ID Register (CAN0EXTID) | |
| H'0080 1006 | CAN0 Configuration Register (CAN0CONF) | |
| H'0080 1008 | CAN0 Time Stamp Count Register (CAN0TSTMP) | |
| H'0080 100A | CAN0 Receive Error Count Register (CAN0REC) | CAN0 Transmit Error Count Register (CAN0TEC) |
| H'0080 100C | CAN0 Slot Interrupt Status Register (CAN0SLIST) | |
| H'0080 100E | | |
| H'0080 1010 | CAN0 Slot Interrupt Mask Register (CAN0SLIMK) | |
| H'0080 1012 | | |
| H'0080 1014 | CAN0 Error Interrupt Status Register (CAN0ERIST) | CAN0 Error Interrupt Mask Register (CAN0ERIMK) |
| H'0080 1016 | CAN0 Baud Rate Prescaler (CAN0BRP) | |
| ≈ | | |
| H'0080 1028 | CAN0 Global Mask Register Standard ID0 (C0GMSKS0) | CAN0 Global Mask Register Standard ID1 (C0GMSKS1) |
| H'0080 102A | CAN0 Global Mask Register Extended ID0 (C0GMSKE0) | CAN0 Global Mask Register Extended ID1 (C0GMSKE1) |
| H'0080 102C | CAN0 Global Mask Register Extended ID2 (C0GMSKE2) | |
| H'0080 102E | | |
| H'0080 1030 | CAN0 Local Mask Register A Standard ID0 (C0LMSKAS0) | CAN0 Local Mask Register A Standard ID1 (C0LMSKAS1) |
| H'0080 1032 | CAN0 Local Mask Register A Extended ID0 (C0LMSKAE0) | CAN0 Local Mask Register A Extended ID1 (C0LMSKAE1) |
| H'0080 1034 | CAN0 Local Mask Register A Extended ID2 (C0LMSKAE2) | |
| H'0080 1036 | | |
| H'0080 1038 | CAN0 Local Mask Register B Standard ID0 (C0LMSKBS0) | CAN0 Local Mask Register B Standard ID1 (C0LMSKBS1) |
| H'0080 103A | CAN0 Local Mask Register B Extended ID0 (C0LMSKBE0) | CAN0 Local Mask Register B Extended ID1 (C0LMSKBE1) |
| H'0080 103C | CAN0 Local Mask Register B Extended ID2 (C0LMSKBE2) | |
| ≈ | | |
| H'0080 1050 | CAN0 Message Slot 0 Control Register (C0MSL0CNT) | CAN0 Message Slot 1 Control Register (C0MSL1CNT) |
| H'0080 1052 | CAN0 Message Slot 2 Control Register (C0MSL2CNT) | CAN0 Message Slot 3 Control Register (C0MSL3CNT) |
| H'0080 1054 | CAN0 Message Slot 4 Control Register (C0MSL4CNT) | CAN0 Message Slot 5 Control Register (C0MSL5CNT) |
| H'0080 1056 | CAN0 Message Slot 6 Control Register (C0MSL6CNT) | CAN0 Message Slot 7 Control Register (C0MSL7CNT) |
| H'0080 1058 | CAN0 Message Slot 8 Control Register (C0MSL8CNT) | CAN0 Message Slot 9 Control Register (C0MSL9CNT) |
| H'0080 105A | CAN0 Message Slot 10 Control Register (C0MSL10CNT) | CAN0 Message Slot 11 Control Register (C0MSL11CNT) |
| H'0080 105C | CAN0 Message Slot 12 Control Register (C0MSL12CNT) | CAN0 Message Slot 13 Control Register (C0MSL13CNT) |
| H'0080 105E | CAN0 Message Slot 14 Control Register (C0MSL14CNT) | CAN0 Message Slot 15 Control Register (C0MSL15CNT) |
| ≈ | | |

Blank addresses are reserved.

**Figure 13.2.1 CAN Module Related Register Map (1/4)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 1100 | CAN0 Message Slot 0 Standard ID0 (C0MSL0SID0) | CAN0 Message Slot 0 Standard ID1 (C0MSL0SID1) |
| H'0080 1102 | CAN0 Message Slot 0 Extended ID0 (C0MSL0EID0) | CAN0 Message Slot 0 Extended ID1 (C0MSL0EID1) |
| H'0080 1104 | CAN0 Message Slot 0 Extended ID2 (C0MSL0EID2) | CAN0 Message Slot 0 Data Length Register (C0MSL0DLC) |
| H'0080 1106 | CAN0 Message Slot 0 Data 0 (C0MSL0DT0) | CAN0 Message Slot 0 Data 1 (C0MSL0DT1) |
| H'0080 1108 | CAN0 Message Slot 0 Data 2 (C0MSL0DT2) | CAN0 Message Slot 0 Data 3 (C0MSL0DT3) |
| H'0080 110A | CAN0 Message Slot 0 Data 4 (C0MSL0DT4) | CAN0 Message Slot 0 Data 5 (C0MSL0DT5) |
| H'0080 110C | CAN0 Message Slot 0 Data 6 (C0MSL0DT6) | CAN0 Message Slot 0 Data 7 (C0MSL0DT7) |
| H'0080 110E | CAN0 Message Slot 0 Time Stamp (C0MSL0TSP) | |
| H'0080 1110 | CAN0 Message Slot 1 Standard ID0 (C0MSL1SID0) | CAN0 Message Slot 1 Standard ID1 (C0MSL1SID1) |
| H'0080 1112 | CAN0 Message Slot 1 Extended ID0 (C0MSL1EID0) | CAN0 Message Slot 1 Extended ID1 (C0MSL1EID1) |
| H'0080 1114 | CAN0 Message Slot 1 Extended ID2 (C0MSL1EID2) | CAN0 Message Slot 1 Data Length Register (C0MSL1DLC) |
| H'0080 1116 | CAN0 Message Slot 1 Data 0 (C0MSL1DT0) | CAN0 Message Slot 1 Data 1 (C0MSL1DT1) |
| H'0080 1118 | CAN0 Message Slot 1 Data 2 (C0MSL1DT2) | CAN0 Message Slot 1 Data 3 (C0MSL1DT3) |
| H'0080 111A | CAN0 Message Slot 1 Data 4 (C0MSL1DT4) | CAN0 Message Slot 1 Data 5 (C0MSL1DT5) |
| H'0080 111C | CAN0 Message Slot 1 Data 6 (C0MSL1DT6) | CAN0 Message Slot 1 Data 7 (C0MSL1DT7) |
| H'0080 111E | CAN0 Message Slot 1 Time Stamp (C0MSL1TSP) | |
| H'0080 1120 | CAN0 Message Slot 2 Standard ID0 (C0MSL2SID0) | CAN0 Message Slot 2 Standard ID1 (C0MSL2SID1) |
| H'0080 1122 | CAN0 Message Slot 2 Extended ID0 (C0MSL2EID0) | CAN0 Message Slot 2 Extended ID1 (C0MSL2EID1) |
| H'0080 1124 | CAN0 Message Slot 2 Extended ID2 (C0MSL2EID2) | CAN0 Message Slot 2 Data Length Register (C0MSL2DLC) |
| H'0080 1126 | CAN0 Message Slot 2 Data 0 (C0MSL2DT0) | CAN0 Message Slot 2 Data 1 (C0MSL2DT1) |
| H'0080 1128 | CAN0 Message Slot 2 Data 2 (C0MSL2DT2) | CAN0 Message Slot 2 Data 3 (C0MSL2DT3) |
| H'0080 112A | CAN0 Message Slot 2 Data 4 (C0MSL2DT4) | CAN0 Message Slot 2 Data 5 (C0MSL2DT5) |
| H'0080 112C | CAN0 Message Slot 2 Data 6 (C0MSL2DT6) | CAN0 Message Slot 2 Data 7 (C0MSL2DT7) |
| H'0080 112E | CAN0 Message Slot 2 Time Stamp (C0MSL2TSP) | |
| H'0080 1130 | CAN0 Message Slot 3 Standard ID0 (C0MSL3SID0) | CAN0 Message Slot 3 Standard ID1 (C0MSL3SID1) |
| H'0080 1132 | CAN0 Message Slot 3 Extended ID0 (C0MSL3EID0) | CAN0 Message Slot 3 Extended ID1 (C0MSL3EID1) |
| H'0080 1134 | CAN0 Message Slot 3 Extended ID2 (C0MSL3EID2) | CAN0 Message Slot 3 Data Length Register (C0MSL3DLC) |
| H'0080 1136 | CAN0 Message Slot 3 Data 0 (C0MSL3DT0) | CAN0 Message Slot 3 Data 1 (C0MSL3DT1) |
| H'0080 1138 | CAN0 Message Slot 3 Data 2 (C0MSL3DT2) | CAN0 Message Slot 3 Data 3 (C0MSL3DT3) |
| H'0080 113A | CAN0 Message Slot 3 Data 4 (C0MSL3DT4) | CAN0 Message Slot 3 Data 5 (C0MSL3DT5) |
| H'0080 113C | CAN0 Message Slot 3 Data 6 (C0MSL3DT6) | CAN0 Message Slot 3 Data 7 (C0MSL3DT7) |
| H'0080 113E | CAN0 Message Slot 3 Time Stamp (C0MSL3TSP) | |
| H'0080 1140 | CAN0 Message Slot 4 Standard ID0 (C0MSL4SID0) | CAN0 Message Slot 4 Standard ID1 (C0MSL4SID1) |
| H'0080 1142 | CAN0 Message Slot 4 Extended ID0 (C0MSL4EID0) | CAN0 Message Slot 4 Extended ID1 (C0MSL4EID1) |
| H'0080 1144 | CAN0 Message Slot 4 Extended ID2 (C0MSL4EID2) | CAN0 Message Slot 4 Data Length Register (C0MSL4DLC) |
| H'0080 1146 | CAN0 Message Slot 4 Data 0 (C0MSL4DT0) | CAN0 Message Slot 4 Data 1 (C0MSL4DT1) |
| H'0080 1148 | CAN0 Message Slot 4 Data 2 (C0MSL4DT2) | CAN0 Message Slot 4 Data 3 (C0MSL4DT3) |
| H'0080 114A | CAN0 Message Slot 4 Data 4 (C0MSL4DT4) | CAN0 Message Slot 4 Data 5 (C0MSL4DT5) |
| H'0080 114C | CAN0 Message Slot 4 Data 6 (C0MSL4DT6) | CAN0 Message Slot 4 Data 7 (C0MSL4DT7) |
| H'0080 114E | CAN0 Message Slot 4 Time Stamp (C0MSL4TSP) | |
| H'0080 1150 | CAN0 Message Slot 5 Standard ID0 (C0MSL5SID0) | CAN0 Message Slot 5 Standard ID1 (C0MSL5SID1) |
| H'0080 1152 | CAN0 Message Slot 5 Extended ID0 (C0MSL5EID0) | CAN0 Message Slot 5 Extended ID1 (C0MSL5EID1) |

Blank addresses are reserved.

**Figure 13.2.2 CAN Module Related Register Map (2/4)**

| Address | +0 Address<br>D0 D7 | +1 Address<br>D8 D15 |
|---|---|---|
| H'0080 1154 | CAN0 Message Slot 5 Extended ID2 (C0MSL5EID2) | CAN0 Message Slot 5 Data Length Register (C0MSL5DLC) |
| H'0080 1156 | CAN0 Message Slot 5 Data 0 (C0MSL5DT0) | CAN0 Message Slot 5 Data 1 (C0MSL5DT1) |
| H'0080 1158 | CAN0 Message Slot 5 Data 2 (C0MSL5DT2) | CAN0 Message Slot 5 Data 3 (C0MSL5DT3) |
| H'0080 115A | CAN0 Message Slot 5 Data 4 (C0MSL5DT4) | CAN0 Message Slot 5 Data 5 (C0MSL5DT5) |
| H'0080 115C | CAN0 Message Slot 5 Data 6 (C0MSL5DT6) | CAN0 Message Slot 5 Data 7 (C0MSL5DT7) |
| H'0080 115E | CAN0 Message Slot 5 Time Stamp (C0MSL5TSP) | |
| H'0080 1160 | CAN0 Message Slot 6 Standard ID0 (C0MSL6SID0) | CAN0 Message Slot 6 Standard ID1 (C0MSL6SID1) |
| H'0080 1162 | CAN0 Message Slot 6 Extended ID0 (C0MSL6EID0) | CAN0 Message Slot 6 Extended ID1 (C0MSL6EID1) |
| H'0080 1164 | CAN0 Message Slot 6 Extended ID2 (C0MSL6EID2) | CAN0 Message Slot 6 Data Length Register (C0MSL6DLC) |
| H'0080 1166 | CAN0 Message Slot 6 Data 0 (C0MSL6DT0) | CAN0 Message Slot 6 Data 1 (C0MSL6DT1) |
| H'0080 1168 | CAN0 Message Slot 6 Data 2 (C0MSL6DT2) | CAN0 Message Slot 6 Data 3 (C0MSL6DT3) |
| H'0080 116A | CAN0 Message Slot 6 Data 4 (C0MSL6DT4) | CAN0 Message Slot 6 Data 5 (C0MSL6DT5) |
| H'0080 116C | CAN0 Message Slot 6 Data 6 (C0MSL6DT6) | CAN0 Message Slot 6 Data 7 (C0MSL6DT7) |
| H'0080 116E | CAN0 Message Slot 6 Time Stamp (C0MSL6TSP) | |
| H'0080 1170 | CAN0 Message Slot 7 Standard ID0 (C0MSL7SID0) | CAN0 Message Slot 7 Standard ID1 (C0MSL7SID1) |
| H'0080 1172 | CAN0 Message Slot 7 Extended ID0 (C0MSL7EID0) | CAN0 Message Slot 7 Extended ID1 (C0MSL7EID1) |
| H'0080 1174 | CAN0 Message Slot 7 Extended ID2 (C0MSL7EID2) | CAN0 Message Slot 7 Data Length Register (C0MSL7DLC) |
| H'0080 1176 | CAN0 Message Slot 7 Data 0 (C0MSL7DT0) | CAN0 Message Slot 7 Data 1 (C0MSL7DT1) |
| H'0080 1178 | CAN0 Message Slot 7 Data 2 (C0MSL7DT2) | CAN0 Message Slot 7 Data 3 (C0MSL7DT3) |
| H'0080 117A | CAN0 Message Slot 7 Data 4 (C0MSL7DT4) | CAN0 Message Slot 7 Data 5 (C0MSL7DT5) |
| H'0080 117C | CAN0 Message Slot 7 Data 6 (C0MSL7DT6) | CAN0 Message Slot 7 Data 7 (C0MSL7DT7) |
| H'0080 117E | CAN0 Message Slot 7 Time Stamp (C0MSL7TSP) | |
| H'0080 1180 | CAN0 Message Slot 8 Standard ID0 (C0MSL8SID0) | CAN0 Message Slot 8 Standard ID1 (C0MSL8SID1) |
| H'0080 1182 | CAN0 Message Slot 8 Extended ID0 (C0MSL8EID0) | CAN0 Message Slot 8 Extended ID1 (C0MSL8EID1) |
| H'0080 1184 | CAN0 Message Slot 8 Extended ID2 (C0MSL8EID2) | CAN0 Message Slot 8 Data Length Register (C0MSL8DLC) |
| H'0080 1186 | CAN0 Message Slot 8 Data 0 (C0MSL8DT0) | CAN0 Message Slot 8 Data 1 (C0MSL8DT1) |
| H'0080 1188 | CAN0 Message Slot 8 Data 2 (C0MSL8DT2) | CAN0 Message Slot 8 Data 3 (C0MSL8DT3) |
| H'0080 118A | CAN0 Message Slot 8 Data 4 (C0MSL8DT4) | CAN0 Message Slot 8 Data 5 (C0MSL8DT5) |
| H'0080 118C | CAN0 Message Slot 8 Data 6 (C0MSL8DT6) | CAN0 Message Slot 8 Data 7 (C0MSL8DT7) |
| H'0080 118E | CAN0 Message Slot 8 Time Stamp (C0MSL8TSP) | |
| H'0080 1190 | CAN0 Message Slot 9 Standard ID0 (C0MSL9SID0) | CAN0 Message Slot 9 Standard ID1 (C0MSL9SID1) |
| H'0080 1192 | CAN0 Message Slot 9 Extended ID0 (C0MSL9EID0) | CAN0 Message Slot 9 Extended ID1 (C0MSL9EID1) |
| H'0080 1194 | CAN0 Message Slot 9 Extended ID2 (C0MSL9EID2) | CAN0 Message Slot 9 Data Length Register (C0MSL9DLC) |
| H'0080 1196 | CAN0 Message Slot 9 Data 0 (C0MSL9DT0) | CAN0 Message Slot 9 Data 1 (C0MSL9DT1) |
| H'0080 1198 | CAN0 Message Slot 9 Data 2 (C0MSL9DT2) | CAN0 Message Slot 9 Data 3 (C0MSL9DT3) |
| H'0080 119A | CAN0 Message Slot 9 Data 4 (C0MSL9DT4) | CAN0 Message Slot 9 Data 5 (C0MSL9DT5) |
| H'0080 119C | CAN0 Message Slot 9 Data 6 (C0MSL9DT6) | CAN0 Message Slot 9 Data 7 (C0MSL9DT7) |
| H'0080 119E | CAN0 Message Slot 9 Time Stamp (C0MSL9TSP) | |
| H'0080 11A0 | CAN0 Message Slot 10 Standard ID0 (C0MSL10SID0) | CAN0 Message Slot 10 Standard ID1 (C0MSL10SID1) |
| H'0080 11A2 | CAN0 Message Slot 10 Extended ID0 (C0MSL10EID0) | CAN0 Message Slot 10 Extended ID1 (C0MSL10EID1) |
| H'0080 11A4 | CAN0 Message Slot 10 Extended ID2 (C0MSL10EID2) | CAN0 Message Slot 10 Data Length Register (C0MSL10DLC) |
| H'0080 11A6 | CAN0 Message Slot 10 Data 0 (C0MSL10DT0) | CAN0 Message Slot 10 Data 1 (C0MSL10DT1) |

Blank addresses are reserved.

**Figure 13.2.3 CAN Module Related Register Map (3/4)**

| Address | +0 Address | +1 Address |
|---|---|---|
| | D0 D7 | D8 D15 |
| H'0080 11A8 | CAN0 Message Slot 10 Data 2 (C0MSL10DT2) | CAN0 Message Slot 10 Data 3 (C0MSL10DT3) |
| H'0080 11AA | CAN0 Message Slot 10 Data 4 (C0MSL10DT4) | CAN0 Message Slot 10 Data 5 (C0MSL10DT5) |
| H'0080 11AC | CAN0 Message Slot 10 Data 6 (C0MSL10DT6) | CAN0 Message Slot 10 Data 7 (C0MSL10DT7) |
| H'0080 11AE | CAN0 Message Slot 10 Time Stamp (C0MSL10TSP) | |
| H'0080 11B0 | CAN0 Message Slot 11 Standard ID0 (C0MSL11SID0) | CAN0 Message Slot 11 Standard ID1 (C0MSL11SID1) |
| H'0080 11B2 | CAN0 Message Slot 11 Extended ID0 (C0MSL11EID0) | CAN0 Message Slot 11 Extended ID1 (C0MSL11EID1) |
| H'0080 11B4 | CAN0 Message Slot 11 Extended ID2 (C0MSL11EID2) | CAN0 Message Slot 11 Data Length Register (C0MSL11DLC) |
| H'0080 11B6 | CAN0 Message Slot 11 Data 0 (C0MSL11DT0) | CAN0 Message Slot 11 Data 1 (C0MSL11DT1) |
| H'0080 11B8 | CAN0 Message Slot 11 Data 2 (C0MSL11DT2) | CAN0 Message Slot 11 Data 3 (C0MSL11DT3) |
| H'0080 11BA | CAN0 Message Slot 11 Data 4 (C0MSL11DT4) | CAN0 Message Slot 11 Data 5 (C0MSL11DT5) |
| H'0080 11BC | CAN0 Message Slot 11 Data 6 (C0MSL11DT6) | CAN0 Message Slot 11 Data 7 (C0MSL11DT7) |
| H'0080 11BE | CAN0 Message Slot 11 Time Stamp (C0MSL11TSP) | |
| H'0080 11C0 | CAN0 Message Slot 12 Standard ID0 (C0MSL12SID0) | CAN0 Message Slot 12 Standard ID1 (C0MSL12SID1) |
| H'0080 11C2 | CAN0 Message Slot 12 Extended ID0 (C0MSL12EID0) | CAN0 Message Slot 12 Extended ID1 (C0MSL12EID1) |
| H'0080 11C4 | CAN0 Message Slot 12 Extended ID2 (C0MSL12EID2) | CAN0 Message Slot 12 Data Length Register (C0MSL12DLC) |
| H'0080 11C6 | CAN0 Message Slot 12 Data 0 (C0MSL12DT0) | CAN0 Message Slot 12 Data 1 (C0MSL12DT1) |
| H'0080 11C8 | CAN0 Message Slot 12 Data 2 (C0MSL12DT2) | CAN0 Message Slot 12 Data 3 (C0MSL12DT3) |
| H'0080 11CA | CAN0 Message Slot 12 Data 4 (C0MSL12DT4) | CAN0 Message Slot 12 Data 5 (C0MSL12DT5) |
| H'0080 11CC | CAN0 Message Slot 12 Data 6 (C0MSL12DT6) | CAN0 Message Slot 12 Data 7 (C0MSL12DT7) |
| H'0080 11CE | CAN0 Message Slot 12 Time Stamp (C0MSL12TSP) | |
| H'0080 11D0 | CAN0 Message Slot 13 Standard ID0 (C0MSL13SID0) | CAN0 Message Slot 13 Standard ID1 (C0MSL13SID1) |
| H'0080 11D2 | CAN0 Message Slot 13 Extended ID0 (C0MSL13EID0) | CAN0 Message Slot 13 Extended ID1 (C0MSL13EID1) |
| H'0080 11D4 | CAN0 Message Slot 13 Extended ID2 (C0MSL13EID2) | CAN0 Message Slot 13 Data Length Register (C0MSL13DLC) |
| H'0080 11D6 | CAN0 Message Slot 13 Data 0 (C0MSL13DT0) | CAN0 Message Slot 13 Data 1 (C0MSL13DT1) |
| H'0080 11D8 | CAN0 Message Slot 13 Data 2 (C0MSL13DT2) | CAN0 Message Slot 13 Data 3 (C0MSL13DT3) |
| H'0080 11DA | CAN0 Message Slot 13 Data 4 (C0MSL13DT4) | CAN0 Message Slot 13 Data 5 (C0MSL13DT5) |
| H'0080 11DC | CAN0 Message Slot 13 Data 6 (C0MSL13DT6) | CAN0 Message Slot 13 Data 7 (C0MSL13DT7) |
| H'0080 11DE | CAN0 Message Slot 13 Time Stamp (C0MSL13TSP) | |
| H'0080 11E0 | CAN0 Message Slot 14 Standard ID0 (C0MSL14SID0) | CAN0 Message Slot 14 Standard ID1 (C0MSL14SID1) |
| H'0080 11E2 | CAN0 Message Slot 14 Extended ID0 (C0MSL14EID0) | CAN0 Message Slot 14 Extended ID1 (C0MSL14EID1) |
| H'0080 11E4 | CAN0 Message Slot 14 Extended ID2 (C0MSL14EID2) | CAN0 Message Slot 14 Data Length Register (C0MSL14DLC) |
| H'0080 11E6 | CAN0 Message Slot 14 Data 0 (C0MSL14DT0) | CAN0 Message Slot 14 Data 1 (C0MSL14DT1) |
| H'0080 11E8 | CAN0 Message Slot 14 Data 2 (C0MSL14DT2) | CAN0 Message Slot 14 Data 3 (C0MSL14DT3) |
| H'0080 11EA | CAN0 Message Slot 14 Data 4 (C0MSL14DT4) | CAN0 Message Slot 14 Data 5 (C0MSL14DT5) |
| H'0080 11EC | CAN0 Message Slot 14 Data 6 (C0MSL14DT6) | CAN0 Message Slot 14 Data 7 (C0MSL14DT7) |
| H'0080 11EE | CAN0 Message Slot 14 Time Stamp (C0MSL14TSP) | |
| H'0080 11F0 | CAN0 Message Slot 15 Standard ID0 (C0MSL15SID0) | CAN0 Message Slot 15 Standard ID1 (C0MSL15SID1) |
| H'0080 11F2 | CAN0 Message Slot 15 Extended ID0 (C0MSL15EID0) | CAN0 Message Slot 15 Extended ID1 (C0MSL15EID1) |
| H'0080 11F4 | CAN0 Message Slot 15 Extended ID2 (C0MSL15EID2) | CAN0 Message Slot 15 Data Length Register (C0MSL15DLC) |
| H'0080 11F6 | CAN0 Message Slot 15 Data 0 (C0MSL15DT0) | CAN0 Message Slot 15 Data 1 (C0MSL15DT1) |
| H'0080 11F8 | CAN0 Message Slot 15 Data 2 (C0MSL15DT2) | CAN0 Message Slot 15 Data 3 (C0MSL15DT3) |
| H'0080 11FA | CAN0 Message Slot 15 Data 4 (C0MSL15DT4) | CAN0 Message Slot 15 Data 5 (C0MSL15DT5) |
| H'0080 11FC | CAN0 Message Slot 15 Data 6 (C0MSL15DT6) | CAN0 Message Slot 15 Data 7 (C0MSL15DT7) |
| H'0080 11FE | CAN0 Message Slot 15 Time Stamp (C0MSL15TSP) | |
| | | |
| H'0080 3FFE | | |

Blank addresses are reserved.

**Figure 13.2.4 CAN Module Related Register Map (4/4)**

### 13.2.1 CAN Control Register

#### ■ CAN0 Control Register (CAN0CNT)

<Address:H'0080 1000>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | RBO | TSR | TSP | | | | | FRST | BCM | | LBM | RST |

<When reset:H'0011>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-3 | No functions assigned | | 0 | – |
| 4 | RBO<br>(Return bus off) | 0: Enables normal operation<br>1: Requests clearing of error counter | ○ | △ |
| 5 | TSR<br>(Time stamp counter reset) | 0: Enables count operation<br>1: Initializes count (by setting H'0000) | ○ | △ |
| 6-7 | TSP<br>(Time stamp prescaler) | D6 D7<br>0 0 : Selects CAN bus bit clock<br>0 1 : Selects CAN bus bit clock divided by 2<br>1 0 : Selects CAN bus bit clock divided by 3<br>1 1 : Selects CAN bus bit clock divided by 4 | ○ | ○ |
| 8-9 | No functions assigned | | 0 | – |
| 10 | No functions assigned (Always set this bit to 0) | | 0 | – |
| 11 | FRST<br>(Forcible reset) | 0: Negates rest<br>1: Forcibly resets | ○ | ○ |
| 12 | BCM<br>(BasicCAN mode) | 0: Disables BasicCAN function<br>1: BasicCAN mode | ○ | ○ |
| 13 | No functions assigned | | 0 | – |
| 14 | LBM<br>(Loopback mode) | 0: Disables loopback function<br>1: Enables loopback function | ○ | ○ |
| 15 | RST<br>(CAN reset) | 0: Negates reset<br>1: Requests reset | ○ | ○ |

W = △ : Only writing a 1 is effective. Automatically cleared to 0 in hardware.

### (1) RBO (Return Bus Off) bit (D4)

Setting this bit to 1 clears the Receive Error Counter (CAN0REC) and Transmit Error Counter (CAN0TEC) and forcibly places the CAN module into an error active state. This bit is cleared when an error active state is entered.

Note: • After clearing the error counter, transmission becomes possible when 11 consecutive recessive bits are detected on the CAN bus.

### (2) TSR (Time Stamp Counter Reset) bit (D5)

Setting this bit to 1 clears the value of the CAN Time Stamp Counter Register (CAN0TSTMP) to H'0000. This bit is cleared when the value of the CAN Time Stamp Counter Register (CAN0TSTMP) is cleared to H'0000.

### (3) TSP (Time Stamp Prescaler) bits (D6, D7)

These bits select the count clock source for the time stamp counter.

Note: • Do not change settings of TSP bits while CAN is operating (CAN Status Register CRS bit = 0).

### (4) FRST (Forcible Reset) bit (D11)

When the FRST bit is set to 1, the CAN module is separated from the CAN bus regardless of whether or not the CAN module is communicating and the protocol control unit is reset. Up to 5 BCLK periods are required before the protocol control unit is reset after setting the FRST bit.

Notes: • If the FRST bit is set to 1 during communication, the CTX pin output goes high immediately after that. Therefore, setting the FRST bit to 1 while transmitting CAN frame may cause a CAN bus error.
- When the protocol control unit is reset by setting the RST bit to 1, the CAN Time Stamp Count Register and CAN Transmit/Receive Error Count Registers are initialized to 0.
- To restart CAN communication, the FRST and RST bits must be cleared to 0.
- The CAN Message Slot Control Register's transmit/receive request are not cleared for reasons that the FRST or RST bits are set.

### (5) BCM (BasicCAN Mode) bit (D12)

By setting this bit to 1, the CAN module can be operated in BasicCAN mode.

- **Operation during BasicCAN mode**

  In BasicCAN mode, two local slots-slots 14 and 15-are used as double buffers, and receive frames that are found matching to the ID by acceptance filtering are stored alternately in slots 14 and 15. Used for this acceptance filtering when slot 14 is active (next receive frame to be stored in slot 14) are the ID set for slot 14 and local mask A, and those used when slot 15 is active are the ID set for slot 15 and local mask B. Two types of frames-data frame and remote frame-can be received in this mode.

By using the same ID and setting the same value in mask registers for the two slots, the possibility of a message-lost trouble when, for example, receiving frames which have many IDs can be reduced.

- **Procedure for entering BasicCAN mode**

Follow the procedure below during initialization:

1. Set the IDs for slots 14 and 15 and local mask registers A and B. (We recommend setting the same value.)
2. Set the frame types handled by slots 14 and 15 (standard or extended) in the CAN Extended ID Register. (We recommend setting the same type.)
3. Set the Message Slot Control Register for slots 14 and 15 to for data frame reception.
4. Set the BCM bit to 1.

Notes: • Do not change settings of BCM bit when CAN is operating (CAN Status Register CRS bit = 0).
• The first slot that is active after clearing the RST bit is slot 14.
• Even during BasicCAN mode, slots 0 to 13 can be used as in normal operation.

### (6) LBM (Loopback Mode) bit (D14)

When the LBM bit is set to 1, if a receive slot exists whose ID matches that of the frame sent by the CAN module itself, then the frame can be received.

Notes: • No ACK is returned for the transmit frame.
• Do not change settings of LBM bit when CAN is operating (CAN Status Register CRS bit = 0).

### (7) RST (CAN Reset) bit (D15)

When the RST bit is cleared to 0, the CAN module is connected to the CAN bus and becomes possible to communicate after detecting 11 consecutive recessive bits. Also, the CAN Time Stamp Count Register thereby starts counting.
When the RST bit is set to 1, the CAN module is reset so that after sending a frame from the slot which has had a transmit request set, the protocol control unit is reset and the CAN module is disconnected from the CAN bus. Frames received during this time are processed normally.

Notes: • It is inhibited to set a new transmit request for a while from when the CAN Status Register CRS bit is set to 1 after setting the RST bit to 1 till when the protocol control unit is reset.
• When the protocol control unit is reset by setting the RST bit to 1, the CAN Time Stamp Count Register and CAN Transmit/Receive Error Count Registers are initialized to 0.
• To restart CAN communication, the FRST and RST bits must be cleared to 0.

### 13.2.2 CAN Status Register

**■ CAN0 Status Register (CAN0STAT)** <Address:H'0080 1002>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | BOS | EPS | CBS | BCS |  | LBS | CRS | RSB | TSB | RSC | TSC | MSN |  |  |  |

<When reset:H'0100>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | No functions assigned |  | 0 | – |
| 1 | BOS<br>(Bus off status) | 0: Not Bus off<br>1: Bus off state | ○ | – |
| 2 | EPS<br>(Error passive status) | 0: Not error passive<br>1: Error passive state | ○ | – |
| 3 | CBS<br>(CAN bus error) | 0: No error occurred<br>1: Error occurred | ○ | – |
| 4 | BCS<br>(BasicCAN status) | 0: Normal mode<br>1: BasicCAN mode | ○ | – |
| 5 | No functions assigned |  | 0 | – |
| 6 | LBS<br>(Loopback status) | 0: Normal mode<br>1: Loopback mode | ○ | – |
| 7 | CRS<br>(CAN reset status) | 0: Operating<br>1: Reset | ○ | – |
| 8 | RSB<br>(Receive status) | 0: Not receiving<br>1: Receiving | ○ | – |
| 9 | TSB<br>(Transmit status) | 0: Not transmitting<br>1: Transmitting | ○ | – |
| 10 | RSC<br>(Receive complete status) | 0: Reception not completed yet<br>1: Reception completed | ○ | – |
| 11 | TSC<br>(Transmit complete status) | 0: Transmission not completed yet<br>1: Transmission completed | ○ | – |

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 12-15 | MSN | Number of message slot which has finished sending or receiving | | |
| | (Message slot number) | 0000 : Slot0 | ○ | – |
| | | 0001 : Slot1 | | |
| | | 0010 : Slot2 | | |
| | | 0011 : Slot3 | | |
| | | 0100 : Slot4 | | |
| | | 0101 : Slot5 | | |
| | | 0110 : Slot6 | | |
| | | 0111 : Slot7 | | |
| | | 1000 : Slot8 | | |
| | | 1001 : Slot9 | | |
| | | 1010 : Slot10 | | |
| | | 1011 : Slot11 | | |
| | | 1100 : Slot12 | | |
| | | 1101 : Slot13 | | |
| | | 1110 : Slot14 | | |
| | | 1111 : Slot15 | | |

### (1) BOS (Bus Off Status) bit (D1)

When BOS bit = 1, it means that the CAN module is in a bus-off state.

[Set condition]

This bit is set to 1 when the transmit error counter value exceeded 255 and a bus-off state is entered.

[Clear condition]

This bit is cleared when returned from the bus-off state.

### (2) EPS (Error Passive Status) bit (D2)

When EPS bit = 1, it means that the CAN module is in an error passive state.

[Set condition]

This bit is set to 1 when the transmit or receive error counter value exceeded 127 and an error passive state is entered.

[Clear condition]

This bit is cleared when switched from the error passive state.

### (3) CBS (CAN Bus Error) bit (D3)

[Set condition]

This bit is set to 1 when an error on the CAN bus is detected.

[Clear condition]

This bit is cleared when normally transmitted or received.

### (4) BCS (BasicCAN Status) bit (D4)

When BCS bit = 1, it means that the CAN module is operating in BasicCAN mode.

[Set condition]

This bit is set to 1 when operating in BasicCAN mode.

BasicCAN mode operates under the following conditions:

- The CAN Control Register BCM bit must be set to 1.
- Slots 14 and 15 both must be set for data frame reception.

[Clear condition]

This bit is cleared by clearing the BCM bit to 0.

### (5) LBS (Loopback Status) bit (D6)

When LBS bit = 1, it means that the CAN module is operating in loopback mode.

[Set condition]

This bit is set to 1 by setting the CAN Control Register LBM (loopback mode) bit to 1.

[Clear condition]

This bit is cleared by clearing the LBM bit to 0.

### (6) CRS (CAN Reset Status) bit (D7)

When CRS bit = 1, it means that the protocol control unit is in a reset state.

[Set condition]

This bit is set to 1 when the CAN module's protocol control unit is in a reset state.

[Clear condition]

This bit is cleared by clearing the CAN Control Register RST (CAN reset) bit to 0.

**(7) RSB (Receive Status) bit (D8)**

[Set condition]

This bit is set to 1 when the CAN module is operating as a receive node.

[Clear condition]

This bit is cleared when the CAN module started operating as a transmit node or entered a bus idle state.

**(8) TSB (Transmit Status) bit (D9)**

[Set condition]

This bit is set to 1 when the CAN module is operating as a transmit node.

[Clear condition]

This bit is cleared when the CAN module started operating as a receive node or entered a bus idle state.

**(9) RSC (Receive Complete Status) bit (D10)**

[Set condition]

This bit is set to 1 when the CAN module finished receiving normally (regardless of whether any slot exists that meets receive conditions).

[Clear condition]

This bit is cleared when the CAN module finished transmitting normally.

**(10) TSC (Transmit Complete Status) bit (D11)**

[Set condition]

This bit is set to 1 when the CAN module finished transmitting normally.

[Clear condition]

This bit is cleared when the CAN module finished receiving normally.

**(11) MSN (Message Slot Number) bits (D12-D15)**

These bits show the relevant slot number when the CAN module finished transmitting or finished storing received data. This bit cannot be cleared to 0 in software.

### 13.2.3 CAN Extended ID Register

#### ■ CAN0 Extended ID Register (CAN0EXTID) <Address:H'0080 1004>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IDE0 | IDE1 | IDE2 | IDE3 | IDE4 | IDE5 | IDE6 | IDE7 | IDE8 | IDE9 | IDE10 | IDE11 | IDE12 | IDE13 | IDE14 | IDE15 |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | IDE0 (Extended ID0) | 0: Standard ID format | ○ | ○ |
| 1 | IDE1 (Extended ID1) | 1: Extended ID format | | |
| 2 | IDE2 (Extended ID2) | | | |
| 3 | IDE3 (Extended ID3) | | | |
| 4 | IDE4 (Extended ID4) | | | |
| 5 | IDE5 (Extended ID5) | | | |
| 6 | IDE6 (Extended ID6) | | | |
| 7 | IDE7 (Extended ID7) | | | |
| 8 | IDE8 (Extended ID8) | | | |
| 9 | IDE9 (Extended ID9) | | | |
| 10 | IDE10 (Extended ID10) | | | |
| 11 | IDE11 (Extended ID11) | | | |
| 12 | IDE12 (Extended ID12) | | | |
| 13 | IDE13 (Extended ID13) | | | |
| 14 | IDE14 (Extended ID14) | | | |
| 15 | IDE15 (Extended ID15) | | | |

This register selects the format of frames handled in message slots corresponding to each bit. The standard ID format is selected when a message slot's corresponding bit is set to 0, or the extended ID format is selected when the bit is set to 1.

Note: • Settings of each bit of this register can only be changed when the corresponding slot does not have transmit or receive requests set.

### 13.2.4 CAN Configuration Register

#### ■ CAN0 Configuration Register (CAN0CONF)

<Address:H'0080 1006>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SJW | | PH2 | | | PH1 | | | PRB | | | SAM | | | | |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-1 | SJW<br>(reSynchronization Jump Width) | Sets reSynchronization Jump Width<br>00: SJW = 1Tq<br>01: SJW = 2Tq<br>10: SJW = 3Tq<br>11: SJW = 4Tq | ○ | ○ |
| 2-4 | PH2<br>(Phase Segment2) | Sets Phase Segment2<br>000: Settings inhibited<br>001: Phase Segment2 = 2Tq<br>010: Phase Segment2 = 3Tq<br>011: Phase Segment2 = 4Tq<br>100: Phase Segment2 = 5Tq<br>101: Phase Segment2 = 6Tq<br>110: Phase Segment2 = 7Tq<br>111: Phase Segment2 = 8Tq | ○ | ○ |
| 5-7 | PH1<br>(Phase Segment1) | Sets Phase Segment1<br>000: Phase Segment1 = 1Tq<br>001: Phase Segment1 = 2Tq<br>010: Phase Segment1 = 3Tq<br>011: Phase Segment1 = 4Tq<br>100: Phase Segment1 = 5Tq<br>101: Phase Segment1 = 6Tq<br>110: Phase Segment1 = 7Tq<br>111: Phase Segment1 = 8Tq | ○ | ○ |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-10 | PRB<br>(Propagation Segment) | Sets Propagation Segment<br>000: Propagation Seqment =1Tq<br>001: Propagation Seqment = 2Tq<br>010: Propagation Seqment = 3Tq<br>011: Propagation Seqment = 4Tq<br>100: Propagation Seqment = 5Tq<br>101: Propagation Seqment = 6Tq<br>110: Propagation Seqment = 7Tq<br>111: Propagation Seqment = 8Tq | ○ | ○ |
| 11 | SAM<br>(Number of times sampled) | 0: Samples once<br>1: Samples three times | ○ | ○ |
| 12-15 | No functions assigned | | 0 | – |

Notes: • During CAN operation (CNA Status Register CRS bit = 0), do not alter settings of the CAN Configuration Registers (CAN0CONF and CAN1CONF).

- The bit configuration in this register must be set so as to meet the conditions below.
  - Number of Tq's in one bit: 8 to 25 Tq's
  - SJW ≤ min (Phase Segment 1, Phase Segment 2)
  - Phase Segment 2 = max (Phase Segment 1, IPT) However, IPT = 2 for the M32R/ECU's internal CAN modules.

    Note that min() is the function that returns a smaller value, whereas max() is the function that returns the maximum value.

### (1) SJW bits (D0-D1)

These bits set reSynchronization Jump Width.

### (2) PH2 bits (D2-D4)

These bits set the width of Phase Segment2.

Note: • The internal CAN module of the 32171 has IPT (Information Processing Time) = 2. Because PH2 bits = 0 after reset, be sure to change it to a value equal to or greater than 2 before you use the CAN module.

### (3) PH1 bits (D5-D7)

These bits set the width of Phase Segment1.

### (4) PRB bits (D8-D10)

These bits set the width of Propagation Segment.

### (5) SAM bit (D11)

This bit sets the number of times each bit is sampled. When SAM = 0, the value sampled at the end of Phase Segment1 is assumed to be the value of the bit. When SAM = 1, the value of the bit is determined by a majority circuit from values sampled at three points-one sampled at the end of Phase Segment1, one sampled before 1Tq, and one sampled before 2Tq.

**Table 13.2.1 Typical Settings of Bit Timing when CPU Clock = 40 MHz**

| Baud Rate | BRP Set Value | Tq Period (ns) | Tq's for 1 Bit | PROP+PH1 | PH2 | Sampling Point |
|---|---|---|---|---|---|---|
| 1M bps | 3 | 100 | 10 | 7 | 2 | 80% |
| | 3 | 100 | 10 | 6 | 3 | 70% |
| | 3 | 100 | 10 | 5 | 4 | 60% |
| | 4 | 125 | 8 | 5 | 2 | 75% |
| | 4 | 125 | 8 | 4 | 3 | 63% |
| 500K bps | 4 | 125 | 16 | 13 | 2 | 88% |
| | 4 | 125 | 16 | 12 | 3 | 81% |
| | 4 | 125 | 16 | 11 | 4 | 75% |
| | 7 | 200 | 10 | 7 | 2 | 80% |
| | 7 | 200 | 10 | 6 | 3 | 70% |
| | 7 | 200 | 10 | 5 | 4 | 60% |
| | 9 | 250 | 8 | 5 | 2 | 75% |
| | 9 | 250 | 8 | 4 | 3 | 63% |

Table 13.2.2 Typical Settings of Bit Timing when CPU Clock = 32 MHz

| Baud Rate | BRP Set Value | Tq Period (ns) | Tq's for 1 Bit | PROP+PH1 | PH2 | Sampling Point |
|---|---|---|---|---|---|---|
| 1M bps | 1 | 62.5 | 16 | 10 | 5 | 69% |
| | 3 | 125 | 8 | 5 | 2 | 75% |
| | 3 | 125 | 8 | 4 | 3 | 63% |
| 500K bps | 3 | 125 | 16 | 13 | 2 | 88% |
| | 3 | 125 | 16 | 11 | 4 | 75% |
| | 7 | 250 | 8 | 5 | 2 | 75% |
| | 7 | 250 | 8 | 4 | 3 | 63% |

### 13.2.5 CAN Time Stamp Count Register

#### ■ CAN0 Time Stamp Count Register (CAN0TSTMP) <Address:H'0080 1008>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CANTSTMP | | | | | | | | | | | | | | | |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | CANSTMP | 16-bit counter value | ○ | – |

The CAN module contains a 16-bit counter. The count period can be chosen to be the CAN bus bit period divided by 1, 2, 3, or 4 by setting the CAN Control Register (CAN0CNT)'s TSP (Time Stamp Prescaler) bits.

When the CAN module finishes transmitting or receiving, it captures the counter value and stores it in a message slot. The counter is made to start counting by clearing the CAN Control Register (CAN0CNT)'s RST bit to 0.

Notes:
- The protocol control unit is reset and the counter is initialized to H'0000 by setting the CAN Control Register (CAN0CNT)'s RST (CAN Reset) bit to 1. Also, the counter can be initialized to H'0000 while the CAN module is operating by setting TSR (Time Stamp Counter Reset) bit to 1.
- During loopback mode, if an ID-matching slot exists, the CAN module stores the time stamp value in the corresponding slot when it finished receiving. (No time stamp value is stored this way when the CAN module finished transmitting.)
- The CAN Timestamp Count Register’s count period varies with the CAN resynchronizing function.

### 13.2.6 CAN Error Count Registers

■ **CAN0 Receive Error Count Register (CAN0REC)** <Address:H'0080 100A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| REC | | | | | | | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | REC<br>(Receive error counter) | Receive error count value | ○ | – |

In an error-active/error-passive state, a receive error count is stored in this register. When received normally, the counter counts down; when an error occurs, the counter counts up.

When received normally while REC ≧ 128 (error-passive), REC is set to 127.

In a bus-off state, an indeterminate value is stored in this register. The count is reset to H'00 upon returning to an error-active state.

■ **CAN0 Transmit Error Count Register (CAN0TEC)** <Address:H'0080 100B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| TEC | | | | | | | |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | TEC<br>(Transmit error counter) | Transmit error count value | ○ | – |

In an error-active/error-passive state, a transmit error count is stored in this register. When transmitted normally, the counter counts down; when an error occurs, the counter counts up.

In a bus-off state, an indeterminate value is stored in this register. The count is reset to H'00 upon returning to an error-active state.

### 13.2.7 CAN Baud Rate Prescaler

#### ■ CAN0 Baud Rate Prescaler (CAN0BRP)

<Address:H'0080 1016>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| BRP | | | | | | | |

<When reset:H'01>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | BRP | Selects baud rate prescaler value | ○ | ○ |

This register sets the Tq period of CAN. The CAN baud rate is determined by (Tq period x number of Tq's for 1 bit).

Tq period = (CANBRP + 1)/ CPU clock

$$\text{CAN transfer baud rate} = \frac{1}{\text{Tq period} \times \text{number of Tq's for 1 bit}}$$

Number of Tq's for 1 bit = Synchronization Segment +
Progagation Segment +
Phase Segment 1 +
Phase Segment 2

Note: • Setting H'00 (divided by 1) is inhibited.

### 13.2.8 CAN Interrupt Related Registers

#### ■ CAN0 Slot Interrupt Status Register (CAN0SLIST)

<Address:H'0080 100C>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSB0 | SSB1 | SSB2 | SSB3 | SSB4 | SSB5 | SSB6 | SSB7 | SSB8 | SSB9 | SSB10 | SSB11 | SSB12 | SSB13 | SSB14 | SSB15 |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | SSB0 (Slot 0 interrupt request status) | 0: No interrupt request | ○ | △ |
| 1 | SSB1 (Slot 1 interrupt request status) | 1: Interrupt requested | | |
| 2 | SSB2 (Slot 2 interrupt request status) | | | |
| 3 | SSB3 (Slot 3 interrupt request status) | | | |
| 4 | SSB4 (Slot 4 interrupt request status) | | | |
| 5 | SSB5 (Slot 5 interrupt request status) | | | |
| 6 | SSB6 (Slot 6 interrupt request status) | | | |
| 7 | SSB7 (Slot 7 interrupt request status) | | | |
| 8 | SSB8 (Slot 8 interrupt request status) | | | |
| 9 | SSB9 (Slot 9 interrupt request status) | | | |
| 10 | SSB10 (Slot 10 interrupt request status) | | | |
| 11 | SSB11 (Slot 11 interrupt request status) | | | |
| 12 | SSB12 (Slot 12 interrupt request status) | | | |
| 13 | SSB13 (Slot 13 interrupt request status) | | | |
| 14 | SSB14 (Slot 14 interrupt request status) | | | |
| 15 | SSB15 (Slot 15 interrupt request status) | | | |

W = △: Only writing a 0 is effective; when you write a 1, the previous value is retained.

When using CAN interrupts, this register lets you know which slot requested an interrupt.

- **Slots set for transmission**

  The bit is set to 1 when the CAN module finished transmitting. The bit is cleared by writing a 0 in software.

- **Slots set for reception**

  The bit is set to 1 when the CAN module finished receiving and finished storing the received message in the message slot. The bit is cleared by writing a 0 in software.

When writing to the CAN slot interrupt status, make sure the bits you want to clear are set to 0 and all other bits are set to 1. The bits thus set to 1 are unaffected by writing in software and retain the value they had before you write.

Notes: • If the automatic response function is enabled for remote frame receive slots, the status is set after the CAN module received a remote frame and when it transmitted a data frame.

• For remote frame transmit slots, the status is set after the CAN module transmitted a remote frame and when it received a data frame.

• If the status is set by an interrupt request at the same time it is cleared in software, the former has priority so that the status is set.

## ■ CAN0 Slot Interrupt Mask Register (CAN0SLIMK) <Address:H'0080 1010>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IRB0 | IRB1 | IRB2 | IRB3 | IRB4 | IRB5 | IRB6 | IRB7 | IRB8 | IRB9 | IRB10 | IRB11 | IRB12 | IRB13 | IRB14 | IRB15 |

<When reset:H'0000>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 | IRB0 (Slot 0 interrupt request mask) | 0: Masks (disables) interrupt request | ○ | ○ |
| 1 | IRB1 (Slot 1 interrupt request mask) | 1: Enables interrupt request | | |
| 2 | IRB2 (Slot 2 interrupt request mask) | | | |
| 3 | IRB3 (Slot 3 interrupt request mask) | | | |
| 4 | IRB4 (Slot 4 interrupt request mask) | | | |
| 5 | IRB5 (Slot 5 interrupt request mask) | | | |
| 6 | IRB6 (Slot 6 interrupt request mask) | | | |
| 7 | IRB7 (Slot 7 interrupt request mask) | | | |
| 8 | IRB8 (Slot 8 interrupt request mask) | | | |
| 9 | IRB9 (Slot 9 interrupt request mask) | | | |
| 10 | IRB10 (Slot 10 interrupt request mask) | | | |
| 11 | IRB11 (Slot 11 interrupt request mask) | | | |
| 12 | IRB12 (Slot 12 interrupt request mask) | | | |
| 13 | IRB13 (Slot 13 interrupt request mask) | | | |
| 14 | IRB14 (Slot 14 interrupt request mask) | | | |
| 15 | IRB15 (Slot 15 interrupt request mask) | | | |

This register controls interrupt requests generated at completion of data transmission or reception in each corresponding slot by enabling or disabling them. When IRBn (n = 0-15) is set to 1, interrupt requests to be generated at completion of transmission or reception in the corresponding slot are enabled.

The CAN Slot Interrupt Status Register (CAN0SLIST) shows you which slot has requested the interrupt.

### ■ CAN0 Error Interrupt Status Register (CAN0ERIST) <Address:H'0080 1014>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  | EIS | PIS | OIS |

<When reset:H00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-4 | No functions assigned |  | 0 | – |
| 5 | EIS<br>(CAN bus error interrupt status) | 0: No interrupt request<br>1: Interrupt requested | ○ | △ |
| 6 | PIS<br>(Error passive interrupt status) |  |  |  |
| 7 | OIS<br>(Bus off interrupt status) |  |  |  |

W = △: Only writing a 0 is effective; when you write a 1, the previous value is retained.

When using CAN interrupts and the interrupt sources are associated with errors, this register lets you know which source generated the interrupt.

#### (1) EIS (CAN Bus Error Interrupt Status) bit (D5)

This bit is set to 1 when a communication error is detected. This bit is cleared by writing a 0 in software.

#### (2) PIS (Error Passive Interrupt Status) bit (D6)

This bit is set to 1 when the CAN module goes to an error passive state. This bit is cleared by writing a 0 in software.

#### (3) OIS (Bus Off Interrupt Status) bit (D7)

This bit is set to 1 when the CAN module goes to a bus-off state. This bit is cleared by writing a 0 in software.

When writing to the CAN error interrupt status, make sure the bits you want to clear are set to 0 and all other bits are set to 1. The bits thus set to 1 are unaffected by writing in software and retain the value they had before you write.

## ■ CAN0 Error Interrupt Mask Register (CAN0ERIMK) <Address:H'0080 1015>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  | EIM | PIM | OIM |

<When reset:H00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-12 | No functions assigned |  | 0 | – |
| 13 | EIM<br>(CAN bus error interrupt mask) | 0: Masks (disables) interrupt request<br>1: Enables interrupt request | ○ | ○ |
| 14 | PIM<br>(Error passive interrupt mask) |  |  |  |
| 15 | OIM<br>(Bus off interrupt mask) |  |  |  |

### (1) EIM (CAN Bus Error Interrupt Mask) bit (D13)

This bit controls interrupt requests generated for occurrence of CAN bus errors by enabling or disabling them. CAN bus error interrupt requests are enabled by setting this bit to 1.

### (2) PIM (Error Passive Interrupt Mask) bit (D14)

This bit controls interrupt requests generated when the CAN module enters an error passive state by enabling or disabling them. Error passive interrupt requests are enabled by setting this bit to 1.

### (3) OIM (Bus Off Interrupt Mask) bit (D15)

This bit controls interrupt requests generated when the CAN module enters a bus-off state by enabling or disabling them. Bus-off interrupt requests are enabled by setting this bit to 1.

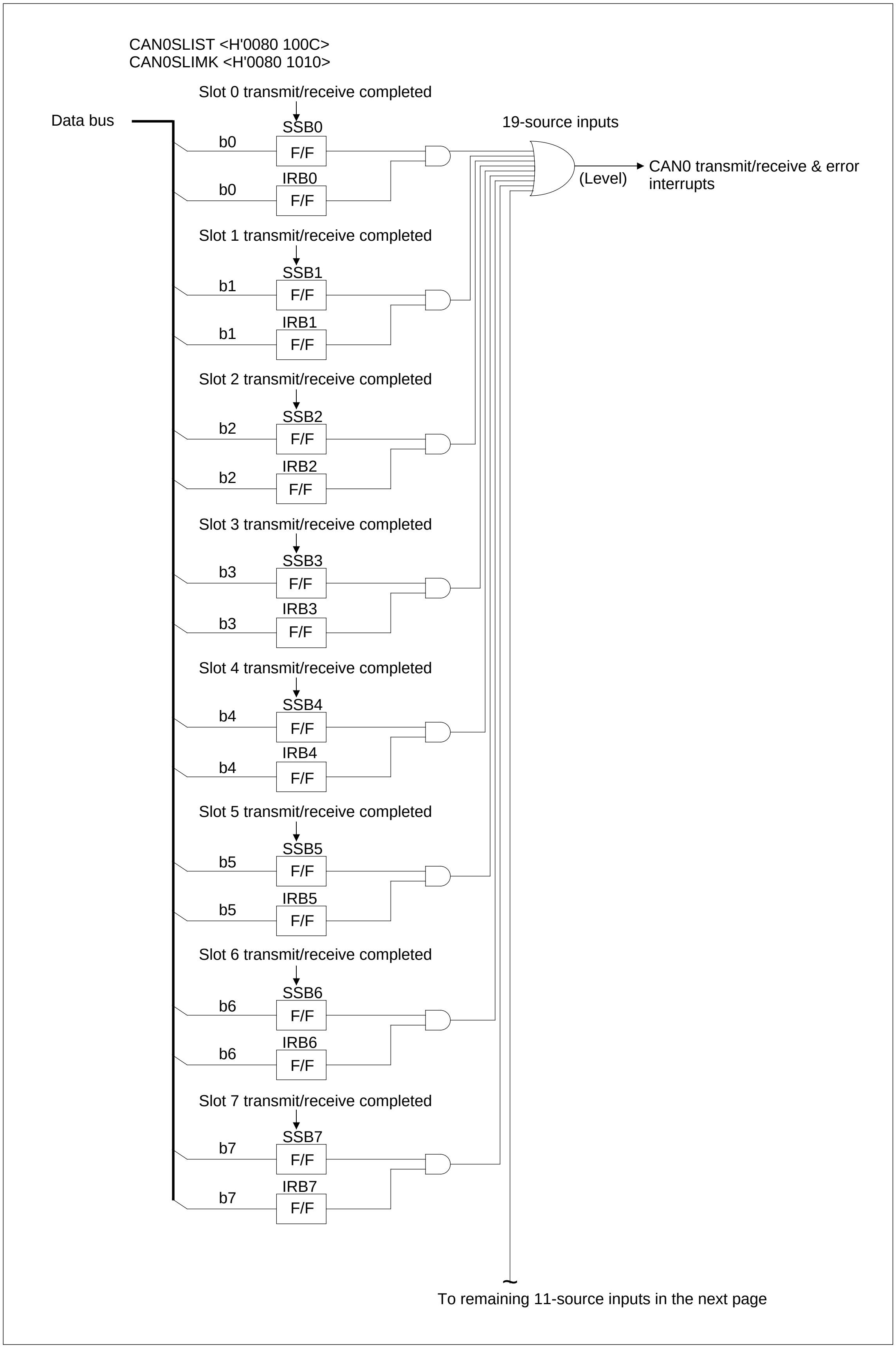

CAN0SLIST <H'0080 100C>
CAN0SLIMK <H'0080 1010>
Slot 0 transmit/receive completed
Data bus
SSB0
19-source inputs
b0
F/F
IRB0
b0
F/F
CAN0 transmit/receive & error interrupts
(Level)
Slot 1 transmit/receive completed
SSB1
b1
F/F
IRB1
b1
F/F
Slot 2 transmit/receive completed
SSB2
b2
F/F
IRB2
b2
F/F
Slot 3 transmit/receive completed
SSB3
b3
F/F
IRB3
b3
F/F
Slot 4 transmit/receive completed
SSB4
b4
F/F
IRB4
b4
F/F
Slot 5 transmit/receive completed
SSB5
b5
F/F
IRB5
b5
F/F
Slot 6 transmit/receive completed
SSB6
b6
F/F
IRB6
b6
F/F
Slot 7 transmit/receive completed
SSB7
b7
F/F
IRB7
b7
F/F
To remaining 11-source inputs in the next page

**Figure 13.2.5 Block Diagram of CAN0 Group Interrupts (1/3)**

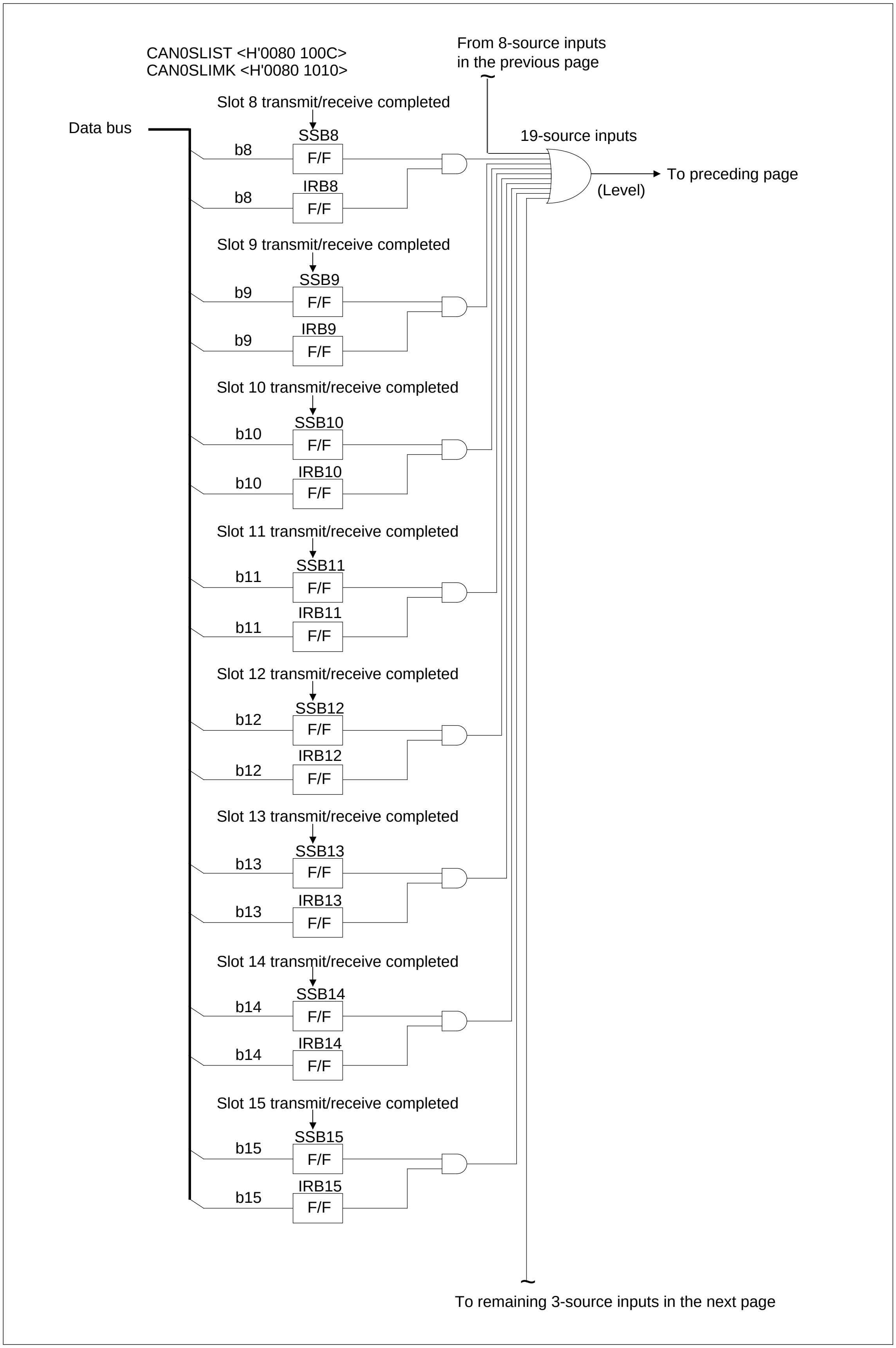

CAN0SLIST <H'0080 100C>
CAN0SLIMK <H'0080 1010>
From 8-source inputs
in the previous page
Data bus
Slot 8 transmit/receive completed
SSB8
b8
F/F
IRB8
b8
F/F
19-source inputs
To preceding page
(Level)
Slot 9 transmit/receive completed
SSB9
b9
F/F
IRB9
b9
F/F
Slot 10 transmit/receive completed
SSB10
b10
F/F
IRB10
b10
F/F
Slot 11 transmit/receive completed
SSB11
b11
F/F
IRB11
b11
F/F
Slot 12 transmit/receive completed
SSB12
b12
F/F
IRB12
b12
F/F
Slot 13 transmit/receive completed
SSB13
b13
F/F
IRB13
b13
F/F
Slot 14 transmit/receive completed
SSB14
b14
F/F
IRB14
b14
F/F
Slot 15 transmit/receive completed
SSB15
b15
F/F
IRB15
b15
F/F
To remaining 3-source inputs in the next page

**Figure 13.2.6 Block Diagram of CAN0 Group Interrupts (2/3)**

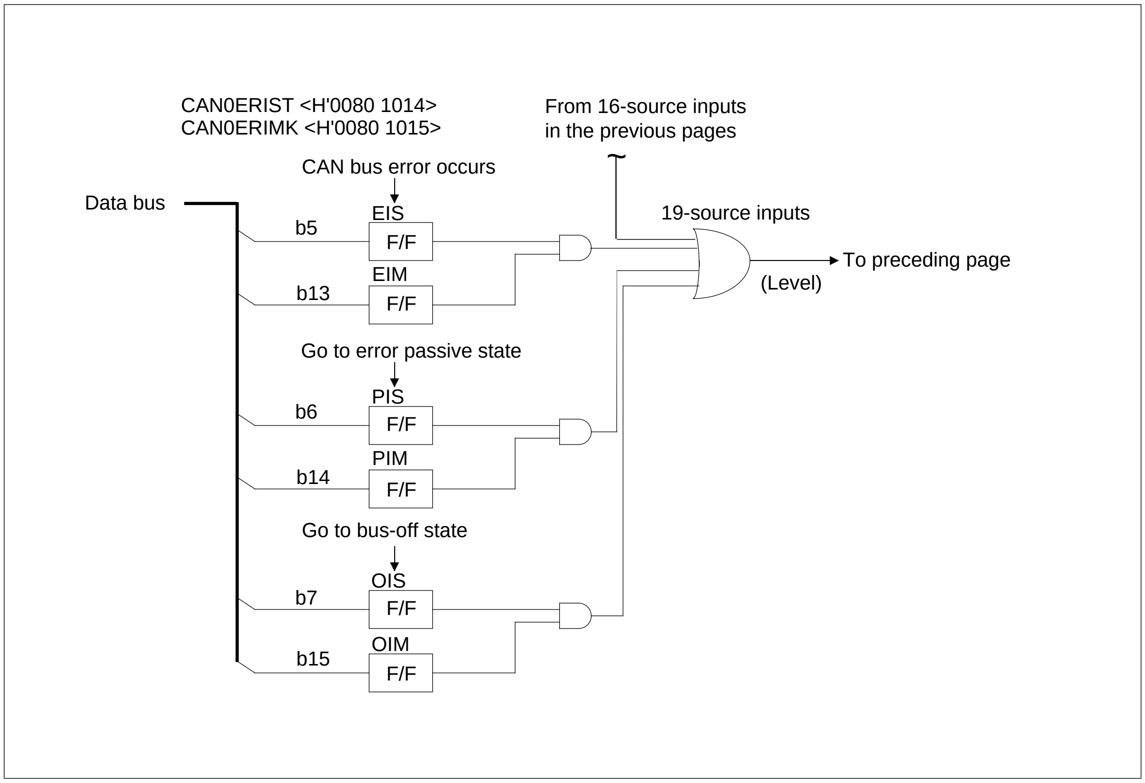


**Figure 13.2.7 Block Diagram of CAN0 Group Interrupts (3/3)**

### 13.2.9 CAN Mask Registers

■ **CAN0 Global Mask Register Standard ID0 (C0GMSKS0)** <Address:H'0080 1028>
■ **CAN0 Local Mask Register A Standard ID0 (C0LMSKAS0)** <Address:H'0080 1030>
■ **CAN0 Local Mask Register B Standard ID0 (C0LMSKBS0)** <Address:H'0080 1038>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  |  | SID0M | SID1M | SID2M | SID3M | SID4M |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-2 | No functions assigned |  | 0 | – |
| 3-7 | SID0M-SID4M<br>(Standard ID0 to standard ID4) | 0: ID not checked<br>1: ID checked | ○ | ○ |

■ **CAN0 Global Mask Register Standard ID1 (C0GMSKS1)** <Address:H'0080 1029>
■ **CAN0 Local Mask Register A Standard ID1 (C0LMSKAS1)** <Address:H'0080 1031>
■ **CAN0 Local Mask Register B Standard ID1 (C0LMSKBS1)** <Address:H'0080 1039>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  | SID5M | SID6M | SID7M | SID8M | SID9M | SID10M |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-9 | No functions assigned |  | 0 | – |
| 10-15 | SID5M-SID10M<br>(Standard ID5 to standard ID10) | 0: ID not checked<br>1: ID checked | ○ | ○ |

Three registers are used in acceptance filtering: Global Mask Register, Local Mask Register A, and Local Mask Register B. The Global Mask Register is used for message slots 0-13, while Local Mask Registers A and B are used for message slots 14 and 15, respectively.

- When a bit in this register is set to 0, its corresponding ID bit is masked (assumed to have matched) during acceptance filtering.
- When a bit in this register is set to 1, its corresponding ID bit is compared with the receive ID during acceptance filtering and when it matches the ID set for the message slot, the received data is stored in it.

Notes: • SID0M corresponds to the MSB of standard ID.
• The Global Mask Register can only be changed when none of slots 0-13 have receive requests set.
• The Local Mask Register A can only be changed when slot 14 does not have a receive request set.
• The Local Mask Register B can only be changed when slot 15 does not have a receive request set.

### ■ CAN0 Global Mask Register Extended ID0 (C0GMSKE0) <Address:H'0080 102A>
### ■ CAN0 Local Mask Register A Extended ID0 (C0LMSKAE0) <Address:H'0080 1032>
### ■ CAN0 Local Mask Register B Extended ID0 (C0LMSKBE0) <Address:H'0080 103A>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | | | EID0M | EID1M | EID2M | EID3M |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-3 | No functions assigned | | 0 | – |
| 4-7 | EID0M-EID3M<br>(Extended ID0 to extended ID3) | 0: ID not checked<br>1: ID checked | ○ | ○ |

### ■ CAN0 Global Mask Register Extended ID1 (C0GMSKE1) <Address:H'0080 102B>
### ■ CAN0 Local Mask Register A Extended ID1 (C0LMSKAE1) <Address:H'0080 1033>
### ■ CAN0 Local Mask Register B Extended ID1 (C0LMSKBE1) <Address:H'0080 103B>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| EID4M | EID5M | EID6M | EID7M | EID8M | EID9M | EID10M | EID11M |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | EID4M-EID11M<br>(Extended ID4 to extended ID11) | 0: ID not checked<br>1: ID checked | ○ | ○ |

### ■ CAN0 Global Mask Register Extended ID2 (C0GMSKE2) <Address:H'0080 102C>
### ■ CAN0 Local Mask Register A Extended ID2 (C0LMSKAE2) <Address:H'0080 1034>
### ■ CAN0 Local Mask Register B Extended ID2 (C0LMSKBE2) <Address:H'0080 103C>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
|  |  | EID12M | EID13M | EID14M | EID15M | EID16M | EID17M |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0,1 | No functions assigned |  | 0 | – |
| 2-7 | EID12M-EID17M<br>(Extended ID12 to extended ID17) | 0: ID not checked<br>1: ID checked | ○ | ○ |

Three registers are used in acceptance filtering: Global Mask Register, Local Mask Register A, and Local Mask Register B. The Global Mask Register is used for message slots 0-13, while Local Mask Registers A and B are used for message slots 14 and 15, respectively.

- When a bit in this register is set to 0, its corresponding ID bit is masked (assumed to have matched) during acceptance filtering.
- When a bit in this register is set to 1, its corresponding ID bit is compared with the receive ID during acceptance filtering and when it matches the ID set for the message slot, the received data is stored in it.

Notes: • EID0M corresponds to the MSB of extended ID.
•The Global Mask Register can only be changed when none of slots 0-13 have receive requests set.
• The Local Mask Register A can only be changed when slot 14 does not have a receive request set.
• The Local Mask Register B can only be changed when slot 15 does not have a receive request set.

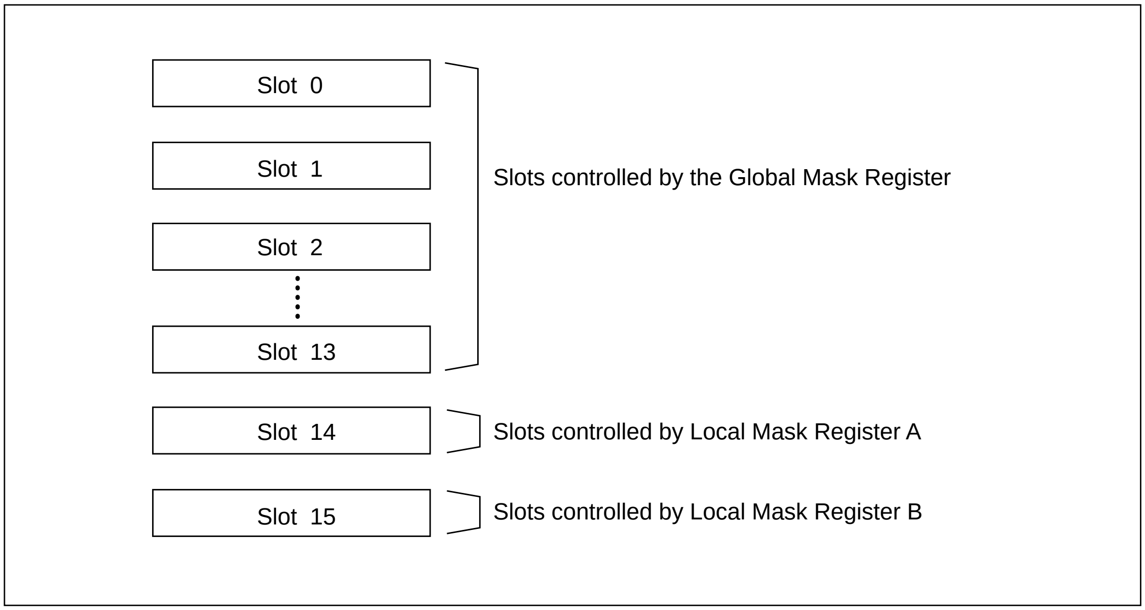

Slot 0
Slot 1
Slot 2
Slot 13
Slots controlled by the Global Mask Register
Slot 14
Slots controlled by Local Mask Register A
Slot 15
Slots controlled by Local Mask Register B

**Figure 13.2.8 Relationship between Mask Registers and the Controlled Slots**

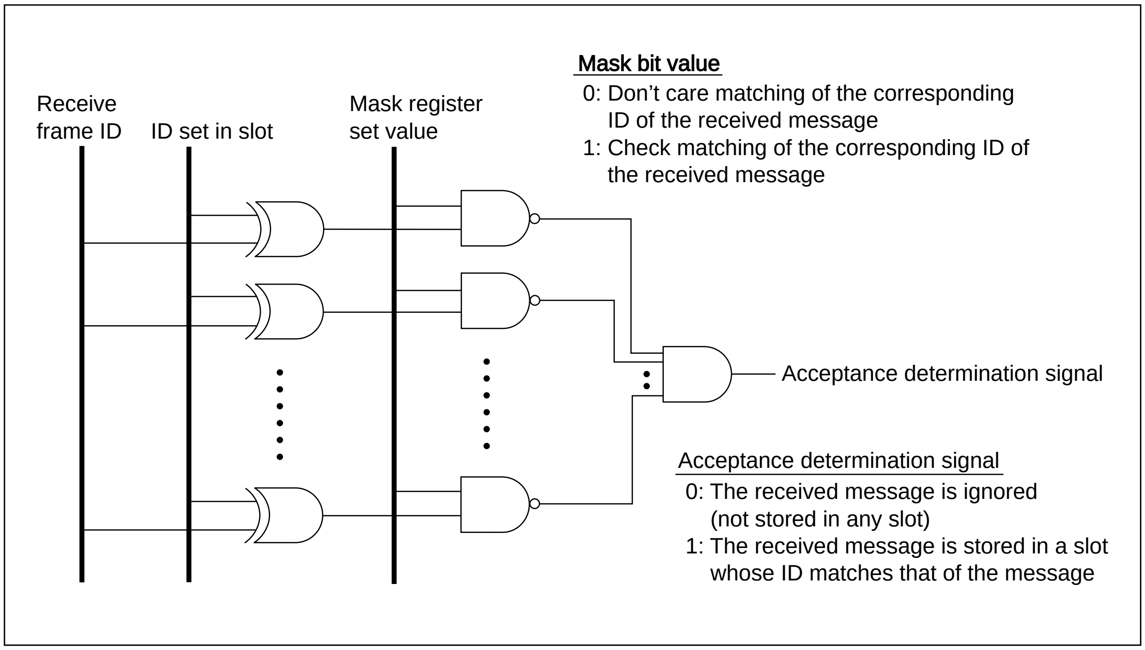

Receive frame ID
ID set in slot
Mask register set value
Mask bit value
0: Don’t care matching of the corresponding ID of the received message
1: Check matching of the corresponding ID of the received message
Acceptance determination signal
Acceptance determination signal
0: The received message is ignored (not stored in any slot)
1: The received message is stored in a slot whose ID matches that of the message

**Figure 13.2.9 Operation of the Acceptance Filter**

### 13.2.10 CAN Message Slot Control Registers

| | |
|---|---|
| **■ CAN0 Message Slot0 Control Registers (COMSL0CNT)** | <Address:H'0080 1050> |
| **■ CAN0 Message Slot1 Control Registers (COMSL1CNT)** | <Address:H'0080 1051> |
| **■ CAN0 Message Slot2 Control Registers (COMSL2CNT)** | <Address:H'0080 1052> |
| **■ CAN0 Message Slot3 Control Registers (COMSL3CNT)** | <Address:H'0080 1053> |
| **■ CAN0 Message Slot4 Control Registers (COMSL4CNT)** | <Address:H'0080 1054> |
| **■ CAN0 Message Slot5 Control Registers (COMSL5CNT)** | <Address:H'0080 1055> |
| **■ CAN0 Message Slot6 Control Registers (COMSL6CNT)** | <Address:H'0080 1056> |
| **■ CAN0 Message Slot7 Control Registers (COMSL7CNT)** | <Address:H'0080 1057> |
| **■ CAN0 Message Slot8 Control Registers (COMSL8CNT)** | <Address:H'0080 1058> |
| **■ CAN0 Message Slot9 Control Registers (COMSL9CNT)** | <Address:H'0080 1059> |
| **■ CAN0 Message Slot10 Control Registers (COMSL10CNT)** | <Address:H'0080 105A> |
| **■ CAN0 Message Slot11 Control Registers (COMSL11CNT)** | <Address:H'0080 105B> |
| **■ CAN0 Message Slot12 Control Registers (COMSL12CNT)** | <Address:H'0080 105C> |
| **■ CAN0 Message Slot13 Control Registers (COMSL13CNT)** | <Address:H'0080 105D> |
| **■ CAN0 Message Slot14 Control Registers (COMSL14CNT)** | <Address:H'0080 105E> |
| **■ CAN0 Message Slot15 Control Registers (COMSL15CNT)** | <Address:H'0080 105F> |

| D0(D8) | 1 | 2 | 3 | 4 | 5 | 6 | D7(D15) |
|---|---|---|---|---|---|---|---|
| TR | RR | RM | RL | RA | ML | TRSTAT | TRFIN |

<When reset:H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0<br>(8) | TR<br>(Transmit request) | 0: Does not use message slot as transmit slot<br>1: Uses message slot as transmit slot | ○ | ○ |
| 1<br>(9) | RR<br>(Receive request) | 0: Does not use message slot as receive slot<br>1: Uses message slot as receive slot | ○ | ○ |
| 2<br>(10) | RM<br>(Remote) | 0: Transmits/receives data frame<br>1: Transmits/receives remote frame | ○ | ○ |
| 3<br>(11) | RL<br>(Automatic response inhibit) | 0: Enables automatic response for remote frame<br>1: Disables automatic response for remote frame | ○ | ○ |
| 4<br>(12) | RA<br>(Remote active) | BasicCAN mode<br>0: Receives data frame (status)<br>1: Receives remote frame (status)<br>Normal mode<br>0: Data frame<br>1: Remote frame | ○ | – |

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 5<br>(13) | ML<br>(Message lost) | 0: Message-lost not occurred<br>1: Message-lost occurred | ○ | △ |
| 6<br>(14) | TRSTAT<br>(Transmit/receive status) | For transmit slots<br>0: Transmission idle<br>1: Transmit request accepted<br>For receive slots<br>0: Reception idle<br>1: Storing received data | ○ | – |
| 7<br>(15) | TRFIN<br>(Transmit/receive complete) | For transmit slots<br>0: Not transmitted yet<br>1: Finished transmitting<br>For receive slots<br>0: Not received yet<br>1: Finished receiving | ○ | △ |

W = △: Only writing a 0 is effective; when you write a 1, the previous value is retained.

### (1) TR (Transmit Request) bit (D0) (D8)

To use the message slot as a transmit slot, set this bit to 1. To use the message slot as a data frame or remote frame receive slot, set this bit to 0.

### (2) RR (Receive Request) bit (D1) (D9)

To use the message slot as a receive slot, set this bit to 1. To use the message slot as a data frame or remote frame transmit slot, set this bit to 0.
If both TR (Transmit Request) and RR (Receive Request) bits are set to 1, device operation is indeterminate.

### (3) RM (Remote) bit (D2) (D10)

To handle remote frames in the message slot, set this bit to 1. The message slot may be set to handle remote frames in following two ways:

- **Set for remote frame transmission**
  The data set in the message slot is transmitted as a remote frame. When the CAN module finished transmitting, the slot is automatically changed to a data frame receive slot. However, if a data frame is received before the CAN module finished sending a remote frame, the data is stored in the message slot and the remote frame is not transmitted.
- **Set for remote frame reception**
  Remote frames are received. The processing to be performed after receiving a remote frame is selected by RL (automatic response inhibit) bit.

### (4) RL (Automatic Response Inhibit) bit (D3) (D11)

This bit is effective when the message slot has been set as a remote frame receive slot. It selects the processing to be performed after receiving a remote frame. When this bit is set to 0, the message slot automatically changes to a transmit slot after receiving a remote frame and transmits the data set in it as a data frame. When this bit is set to 1, the message slot stops operating after receiving a remote frame.

**Note:** Always set this bit to 0 unless the message slot is set for remote frame reception.

### (5) RA (Remote Active) bit (D4) (D12)

This bit functions differently for slots 0-13 and slots 14 and 15.

- **Slots 0-13**
  This bit is set to 1 when the message slot is set for remote frame transmission (reception). Then it is cleared to 0 when remote frame transmission (reception) is completed.
- **Slots 14, 15**
  The function of this bit differs depending on how the CAN Control Register's BCM (BasicCAN mode) bit is set. If BCM = 0 (normal operation), this bit is set to 1 when the message slot is set for remote frame transmission (reception). If BCM = 1 (BasicCAN), this bit shows which type of frame is received. In BasicCAN mode, the received data is stored in slots 14 and 15 for both data frame and remote frame. If RA = 0, it means that the frame stored in the slot is a data frame; if RA = 1, it means that the frame stored in the slot is a remote frame.

### (6) ML (Message Lost) bit (D5) (D13)

This bit is effective for receive slots. It is set to 1 when the message slot contains unread receive data which is overwritten by reception. This bit is cleared by writing a 0 in software.

### (7) TRSTAT (Transmit/Receive Status) bit (D6) (D14)

This bit indicates that the CAN module is transmitting or receiving and is accessing the message slot. This bit is set to 1 when the CAN module is accessing, and set to 0 when not accessing.

- **For transmit slots**
  This bit is set to 1 when a transmit request for the message slot is accepted. It is cleared to 0 when the CAN module lost bus arbitration, when a CAN bus error occurs, or when transmission is completed.
- **For receive slots**
  This bit is set to 1 when during data reception, the received data is being stored in the message slot. Note that the value read from message slot while TRSTAT bit remains set is indeterminate.

### (8) TRFIN (Transmit/Receive Finished) bit (D7) (D15)

This bit indicates that the CAN module finished transmitting or receiving.

- **When set for transmit slots**
  This bit is set to 1 when the CAN module finished transmitting the data stored in the message slot. This bit is cleared by writing a 0 in software. However, it cannot be cleared when TRSTAT (Transmit/Receive Status) bit = 1.
- **When set for receive slots**
  This bit is set to 1 when the CAN module finished receiving normally the data to be stored in the message slot. This bit is cleared by writing a 0 in software. However, it cannot be cleared when TRSTAT (Transmit/Receive Status) bit = 1.

Notes: • Before you can read received data from the message slot, you must clear the TRFIN (Transmit/Receive Finished) bit. Note also that if the TRFIN (Transmit/Receive Finished) bit is set to 1 after you read data, it means that new receive data was stored while you were reading and the data you read contains an indeterminate value. In this case, discard the read data, clear the TRFIN (Transmit/Receive Finished) bit, and read out data again.

• The TRFIN (Transmit/Receive Finished) bit has no effect for remote frames, so that it is not set when remote frame transmission or reception is completed.

### 13.2.11 CAN Message Slots

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Standard ID0 (C0MSL0SID0)** | <Address:H'0080 1100> |
| ■ **CAN0 Message Slot 1 Standard ID0 (C0MSL1SID0)** | <Address:H'0080 1110> |
| ■ **CAN0 Message Slot 2 Standard ID0 (C0MSL2SID0)** | <Address:H'0080 1120> |
| ■ **CAN0 Message Slot 3 Standard ID0 (C0MSL3SID0)** | <Address:H'0080 1130> |
| ■ **CAN0 Message Slot 4 Standard ID0 (C0MSL4SID0)** | <Address:H'0080 1140> |
| ■ **CAN0 Message Slot 5 Standard ID0 (C0MSL5SID0)** | <Address:H'0080 1150> |
| ■ **CAN0 Message Slot 6 Standard ID0 (C0MSL6SID0)** | <Address:H'0080 1160> |
| ■ **CAN0 Message Slot 7 Standard ID0 (C0MSL7SID0)** | <Address:H'0080 1170> |
| ■ **CAN0 Message Slot 8 Standard ID0 (C0MSL8SID0)** | <Address:H'0080 1180> |
| ■ **CAN0 Message Slot 9 Standard ID0 (C0MSL9SID0)** | <Address:H'0080 1190> |
| ■ **CAN0 Message Slot 10 Standard ID0 (C0MSL10SID0)** | <Address:H'0080 11A0> |
| ■ **CAN0 Message Slot 11 Standard ID0 (C0MSL11SID0)** | <Address:H'0080 11B0> |
| ■ **CAN0 Message Slot 12 Standard ID0 (C0MSL12SID0)** | <Address:H'0080 11C0> |
| ■ **CAN0 Message Slot 13 Standard ID0 (C0MSL13SID0)** | <Address:H'0080 11D0> |
| ■ **CAN0 Message Slot 14 Standard ID0 (C0MSL14SID0)** | <Address:H'0080 11E0> |
| ■ **CAN0 Message Slot 15 Standard ID0 (C0MSL15SID0)** | <Address:H'0080 11F0> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ▨ | ▨ | ▨ | SID0 | SID1 | SID2 | SID3 | SID4 |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-2 | No functions assigned (Always set these bits to 0) | | 0 | – |
| 3-7 | SID0-SID4<br>(Standard ID0 to standard ID4) | Standard ID0 to standard ID4 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

■ **CAN0 Message Slot 0 Standard ID1 (C0MSL0SID1)** <Address:H'0080 1101>
■ **CAN0 Message Slot 1 Standard ID1 (C0MSL1SID1)** <Address:H'0080 1111>
■ **CAN0 Message Slot 2 Standard ID1 (C0MSL2SID1)** <Address:H'0080 1121>
■ **CAN0 Message Slot 3 Standard ID1 (C0MSL3SID1)** <Address:H'0080 1131>
■ **CAN0 Message Slot 4 Standard ID1 (C0MSL4SID1)** <Address:H'0080 1141>
■ **CAN0 Message Slot 5 Standard ID1 (C0MSL5SID1)** <Address:H'0080 1151>
■ **CAN0 Message Slot 6 Standard ID1 (C0MSL6SID1)** <Address:H'0080 1161>
■ **CAN0 Message Slot 7 Standard ID1 (C0MSL7SID1)** <Address:H'0080 1171>
■ **CAN0 Message Slot 8 Standard ID1 (C0MSL8SID1)** <Address:H'0080 1181>
■ **CAN0 Message Slot 9 Standard ID1 (C0MSL9SID1)** <Address:H'0080 1191>
■ **CAN0 Message Slot 10 Standard ID1 (C0MSL10SID1)** <Address:H'0080 11A1>
■ **CAN0 Message Slot 11 Standard ID1 (C0MSL11SID1)** <Address:H'0080 11B1>
■ **CAN0 Message Slot 12 Standard ID1 (C0MSL12SID1)** <Address:H'0080 11C1>
■ **CAN0 Message Slot 13 Standard ID1 (C0MSL13SID1)** <Address:H'0080 11D1>
■ **CAN0 Message Slot 14 Standard ID1 (C0MSL14SID1)** <Address:H'0080 11E1>
■ **CAN0 Message Slot 15 Standard ID1 (C0MSL15SID1)** <Address:H'0080 11F1>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  | SID5 | SID6 | SID7 | SID8 | SID9 | SID10 |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8,9 | No functions assigned (Always set these bits to 0) |  | 0 | – |
| 10-15 | SID5-SID10<br>(Standard ID5 to standard ID10) | Standard ID5 to standard ID10 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

**■ CAN0 Message Slot 0 Extended ID0 (C0MSL0EID0)** <Address:H'0080 1102>
**■ CAN0 Message Slot 1 Extended ID0 (C0MSL1EID0)** <Address:H'0080 1112>
**■ CAN0 Message Slot 2 Extended ID0 (C0MSL2EID0)** <Address:H'0080 1122>
**■ CAN0 Message Slot 3 Extended ID0 (C0MSL3EID0)** <Address:H'0080 1132>
**■ CAN0 Message Slot 4 Extended ID0 (C0MSL4EID0)** <Address:H'0080 1142>
**■ CAN0 Message Slot 5 Extended ID0 (C0MSL5EID0)** <Address:H'0080 1152>
**■ CAN0 Message Slot 6 Extended ID0 (C0MSL6EID0)** <Address:H'0080 1162>
**■ CAN0 Message Slot 7 Extended ID0 (C0MSL7EID0)** <Address:H'0080 1172>
**■ CAN0 Message Slot 8 Extended ID0 (C0MSL8EID0)** <Address:H'0080 1182>
**■ CAN0 Message Slot 9 Extended ID0 (C0MSL9EID0)** <Address:H'0080 1192>
**■ CAN0 Message Slot 10 Extended ID0 (C0MSL10EID0)** <Address:H'0080 11A2>
**■ CAN0 Message Slot 11 Extended ID0 (C0MSL11EID0)** <Address:H'0080 11B2>
**■ CAN0 Message Slot 12 Extended ID0 (C0MSL12EID0)** <Address:H'0080 11C2>
**■ CAN0 Message Slot 13 Extended ID0 (C0MSL13EID0)** <Address:H'0080 11D2>
**■ CAN0 Message Slot 14 Extended ID0 (C0MSL14EID0)** <Address:H'0080 11E2>
**■ CAN0 Message Slot 15 Extended ID0 (C0MSL15EID0)** <Address:H'0080 11F2>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ▨ | ▨ | ▨ | ▨ | EID0 | EID1 | EID2 | EID3 |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-3 | No functions assigned (Always set these bits to 0) | | 0 | – |
| 4-7 | EID0-EID3<br>(Extended ID0 to extended ID3) | Extended ID0 to extended ID3 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • When set for the receive slot standard ID format, values written to EID bits when storing received data in the slot are indeterminate.

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Extended ID1 (C0MSL0EID1)** | <Address:H'0080 1103> |
| ■ **CAN0 Message Slot 1 Extended ID1 (C0MSL1EID1)** | <Address:H'0080 1113> |
| ■ **CAN0 Message Slot 2 Extended ID1 (C0MSL2EID1)** | <Address:H'0080 1123> |
| ■ **CAN0 Message Slot 3 Extended ID1 (C0MSL3EID1)** | <Address:H'0080 1133> |
| ■ **CAN0 Message Slot 4 Extended ID1 (C0MSL4EID1)** | <Address:H'0080 1143> |
| ■ **CAN0 Message Slot 5 Extended ID1 (C0MSL5EID1)** | <Address:H'0080 1153> |
| ■ **CAN0 Message Slot 6 Extended ID1 (C0MSL6EID1)** | <Address:H'0080 1163> |
| ■ **CAN0 Message Slot 7 Extended ID1 (C0MSL7EID1)** | <Address:H'0080 1173> |
| ■ **CAN0 Message Slot 8 Extended ID1 (C0MSL8EID1)** | <Address:H'0080 1183> |
| ■ **CAN0 Message Slot 9 Extended ID1 (C0MSL9EID1)** | <Address:H'0080 1193> |
| ■ **CAN0 Message Slot 10 Extended ID1 (C0MSL10EID1)** | <Address:H'0080 11A3> |
| ■ **CAN0 Message Slot 11 Extended ID1 (C0MSL11EID1)** | <Address:H'0080 11B3> |
| ■ **CAN0 Message Slot 12 Extended ID1 (C0MSL12EID1)** | <Address:H'0080 11C3> |
| ■ **CAN0 Message Slot 13 Extended ID1 (C0MSL13EID1)** | <Address:H'0080 11D3> |
| ■ **CAN0 Message Slot 14 Extended ID1 (C0MSL14EID1)** | <Address:H'0080 11E3> |
| ■ **CAN0 Message Slot 15 Extended ID1 (C0MSL15EID1)** | <Address:H'0080 11F3> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| EID4 | EID5 | EID6 | EID7 | EID8 | EID9 | EID10 | EID11 |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | EID4-EID11<br>(Extended ID4 to extended ID11) | Extended ID4 to extended ID11 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • When set for the receive slot standard ID format, values written to EID bits when storing received data in the slot are indeterminate.

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Extended ID2 (C0MSL0EID2)** | <Address:H'0080 1104> |
| ■ **CAN0 Message Slot 1 Extended ID2 (C0MSL1EID2)** | <Address:H'0080 1114> |
| ■ **CAN0 Message Slot 2 Extended ID2 (C0MSL2EID2)** | <Address:H'0080 1124> |
| ■ **CAN0 Message Slot 3 Extended ID2 (C0MSL3EID2)** | <Address:H'0080 1134> |
| ■ **CAN0 Message Slot 4 Extended ID2 (C0MSL4EID2)** | <Address:H'0080 1144> |
| ■ **CAN0 Message Slot 5 Extended ID2 (C0MSL5EID2)** | <Address:H'0080 1154> |
| ■ **CAN0 Message Slot 6 Extended ID2 (C0MSL6EID2)** | <Address:H'0080 1164> |
| ■ **CAN0 Message Slot 7 Extended ID2 (C0MSL7EID2)** | <Address:H'0080 1174> |
| ■ **CAN0 Message Slot 8 Extended ID2 (C0MSL8EID2)** | <Address:H'0080 1184> |
| ■ **CAN0 Message Slot 9 Extended ID2 (C0MSL9EID2)** | <Address:H'0080 1194> |
| ■ **CAN0 Message Slot 10 Extended ID2 (C0MSL10EID2)** | <Address:H'0080 11A4> |
| ■ **CAN0 Message Slot 11 Extended ID2 (C0MSL11EID2)** | <Address:H'0080 11B4> |
| ■ **CAN0 Message Slot 12 Extended ID2 (C0MSL12EID2)** | <Address:H'0080 11C4> |
| ■ **CAN0 Message Slot 13 Extended ID2 (C0MSL13EID2)** | <Address:H'0080 11D4> |
| ■ **CAN0 Message Slot 14 Extended ID2 (C0MSL14EID2)** | <Address:H'0080 11E4> |
| ■ **CAN0 Message Slot 15 Extended ID2 (C0MSL15EID2)** | <Address:H'0080 11F4> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| ▨ | ▨ | EID12 | EID13 | EID14 | EID15 | EID16 | EID17 |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0,1 | No functions assigned (Always set these bits to 0) | | 0 | – |
| 2-7 | EID12-EID17<br>(Extended ID12 to extended ID17) | Extended ID12 to extended ID17 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • When set for the receive slot standard ID format, values written to EID bits when storing received data in the slot are indeterminate.

- **CAN0 Message Slot 0 Data Length Register (C0MSL0DLC)** <Address:H'0080 1105>
- **CAN0 Message Slot 1 Data Length Register (C0MSL1DLC)** <Address:H'0080 1115>
- **CAN0 Message Slot 2 Data Length Register (C0MSL2DLC)** <Address:H'0080 1125>
- **CAN0 Message Slot 3 Data Length Register (C0MSL3DLC)** <Address:H'0080 1135>
- **CAN0 Message Slot 4 Data Length Register (C0MSL4DLC)** <Address:H'0080 1145>
- **CAN0 Message Slot 5 Data Length Register (C0MSL5DLC)** <Address:H'0080 1155>
- **CAN0 Message Slot 6 Data Length Register (C0MSL6DLC)** <Address:H'0080 1165>
- **CAN0 Message Slot 7 Data Length Register (C0MSL7DLC)** <Address:H'0080 1175>
- **CAN0 Message Slot 8 Data Length Register (C0MSL8DLC)** <Address:H'0080 1185>
- **CAN0 Message Slot 9 Data Length Register (C0MSL9DLC)** <Address:H'0080 1195>
- **CAN0 Message Slot 10 Data Length Register (C0MSL10DLC)** <Address:H'0080 11A5>
- **CAN0 Message Slot 11 Data Length Register (C0MSL11DLC)** <Address:H'0080 11B5>
- **CAN0 Message Slot 12 Data Length Register (C0MSL12DLC)** <Address:H'0080 11C5>
- **CAN0 Message Slot 13 Data Length Register (C0MSL13DLC)** <Address:H'0080 11D5>
- **CAN0 Message Slot 14 Data Length Register (C0MSL14DLC)** <Address:H'0080 11E5>
- **CAN0 Message Slot 15 Data Length Register (C0MSL15DLC)** <Address:H'0080 11F5>

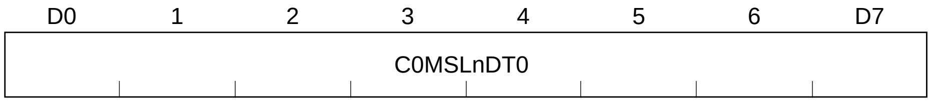
| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| | | | | DLC0 | DLC1 | DLC2 | DLC3 |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-11 | No functions assigned (Always set these bits to 0) | | 0 | – |
| 12-15 | DLC0-DLC3 | 0 0 0 0 : 0 byte | ○ | ○ |
| | (Sets data length) | 0 0 0 1 : 1 byte | | |
| | | 0 0 1 0 : 2 byte | | |
| | | 0 0 1 1 : 3 byte | | |
| | | 0 1 0 0 : 4 byte | | |
| | | 0 1 0 1 : 5 byte | | |
| | | 0 1 1 0 : 6 byte | | |
| | | 0 1 1 1 : 7 byte | | |
| | | 1 X X X : 8 byte | | |

These registers are the transmit frame/receive frame memory space. When transmitting, the register sets the length of transmit data. When receiving, the register stores the received DLC.

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Data 0 (C0MSL0DT0)** | <Address:H'0080 1106> |
| ■ **CAN0 Message Slot 1 Data 0 (C0MSL1DT0)** | <Address:H'0080 1116> |
| ■ **CAN0 Message Slot 2 Data 0 (C0MSL2DT0)** | <Address:H'0080 1126> |
| ■ **CAN0 Message Slot 3 Data 0 (C0MSL3DT0)** | <Address:H'0080 1136> |
| ■ **CAN0 Message Slot 4 Data 0 (C0MSL4DT0)** | <Address:H'0080 1146> |
| ■ **CAN0 Message Slot 5 Data 0 (C0MSL5DT0)** | <Address:H'0080 1156> |
| ■ **CAN0 Message Slot 6 Data 0 (C0MSL6DT0)** | <Address:H'0080 1166> |
| ■ **CAN0 Message Slot 7 Data 0 (C0MSL7DT0)** | <Address:H'0080 1176> |
| ■ **CAN0 Message Slot 8 Data 0 (C0MSL8DT0)** | <Address:H'0080 1186> |
| ■ **CAN0 Message Slot 9 Data 0 (C0MSL9DT0)** | <Address:H'0080 1196> |
| ■ **CAN0 Message Slot 10 Data 0 (C0MSL10DT0)** | <Address:H'0080 11A6> |
| ■ **CAN0 Message Slot 11 Data 0 (C0MSL11DT0)** | <Address:H'0080 11B6> |
| ■ **CAN0 Message Slot 12 Data 0 (C0MSL12DT0)** | <Address:H'0080 11C6> |
| ■ **CAN0 Message Slot 13 Data 0 (C0MSL13DT0)** | <Address:H'0080 11D6> |
| ■ **CAN0 Message Slot 14 Data 0 (C0MSL14DT0)** | <Address:H'0080 11E6> |
| ■ **CAN0 Message Slot 15 Data 0 (C0MSL15DT0)** | <Address:H'0080 11F6> |



<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | COMSLnDT0 | Message slot n data 0 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 0, an indeterminate value is written to this register.

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Data 1 (C0MSL0DT1)** | <Address:H'0080 1107> |
| ■ **CAN0 Message Slot 1 Data 1 (C0MSL1DT1)** | <Address:H'0080 1117> |
| ■ **CAN0 Message Slot 2 Data 1 (C0MSL2DT1)** | <Address:H'0080 1127> |
| ■ **CAN0 Message Slot 3 Data 1 (C0MSL3DT1)** | <Address:H'0080 1137> |
| ■ **CAN0 Message Slot 4 Data 1 (C0MSL4DT1)** | <Address:H'0080 1147> |
| ■ **CAN0 Message Slot 5 Data 1 (C0MSL5DT1)** | <Address:H'0080 1157> |
| ■ **CAN0 Message Slot 6 Data 1 (C0MSL6DT1)** | <Address:H'0080 1167> |
| ■ **CAN0 Message Slot 7 Data 1 (C0MSL7DT1)** | <Address:H'0080 1177> |
| ■ **CAN0 Message Slot 8 Data 1 (C0MSL8DT1)** | <Address:H'0080 1187> |
| ■ **CAN0 Message Slot 9 Data 1 (C0MSL9DT1)** | <Address:H'0080 1197> |
| ■ **CAN0 Message Slot 10 Data 1 (C0MSL10DT1)** | <Address:H'0080 11A7> |
| ■ **CAN0 Message Slot 11 Data 1 (C0MSL11DT1)** | <Address:H'0080 11B7> |
| ■ **CAN0 Message Slot 12 Data 1 (C0MSL12DT1)** | <Address:H'0080 11C7> |
| ■ **CAN0 Message Slot 13 Data 1 (C0MSL13DT1)** | <Address:H'0080 11D7> |
| ■ **CAN0 Message Slot 14 Data 1 (C0MSL14DT1)** | <Address:H'0080 11E7> |
| ■ **CAN0 Message Slot 15 Data 1 (C0MSL15DT1)** | <Address:H'0080 11F7> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT1 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | COMSLnDT1 | Message slot n data 1 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 1 or less , an indeterminate value is written to this register.

### ■ CAN0 Message Slot 0 Data 2 (C0MSL0DT2) <Address:H'0080 1108>
### ■ CAN0 Message Slot 1 Data 2 (C0MSL1DT2) <Address:H'0080 1118>
### ■ CAN0 Message Slot 2 Data 2 (C0MSL2DT2) <Address:H'0080 1128>
### ■ CAN0 Message Slot 3 Data 2 (C0MSL3DT2) <Address:H'0080 1138>
### ■ CAN0 Message Slot 4 Data 2 (C0MSL4DT2) <Address:H'0080 1148>
### ■ CAN0 Message Slot 5 Data 2 (C0MSL5DT2) <Address:H'0080 1158>
### ■ CAN0 Message Slot 6 Data 2 (C0MSL6DT2) <Address:H'0080 1168>
### ■ CAN0 Message Slot 7 Data 2 (C0MSL7DT2) <Address:H'0080 1178>
### ■ CAN0 Message Slot 8 Data 2 (C0MSL8DT2) <Address:H'0080 1188>
### ■ CAN0 Message Slot 9 Data 2 (C0MSL9DT2) <Address:H'0080 1198>
### ■ CAN0 Message Slot 10 Data 2 (C0MSL10DT2) <Address:H'0080 11A8>
### ■ CAN0 Message Slot 11 Data 2 (C0MSL11DT2) <Address:H'0080 11B8>
### ■ CAN0 Message Slot 12 Data 2 (C0MSL12DT2) <Address:H'0080 11C8>
### ■ CAN0 Message Slot 13 Data 2 (C0MSL13DT2) <Address:H'0080 11D8>
### ■ CAN0 Message Slot 14 Data 2 (C0MSL14DT2) <Address:H'0080 11E8>
### ■ CAN0 Message Slot 15 Data 2 (C0MSL15DT2) <Address:H'0080 11F8>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT2 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | COMSLnDT2 | Message slot n data 2 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 2 or less, an indeterminate value is written to this register.

| | |
|---|---|
| **■ CAN0 Message Slot 0 Data 3 (C0MSL0DT3)** | <Address:H'0080 1109> |
| **■ CAN0 Message Slot 1 Data 3 (C0MSL1DT3)** | <Address:H'0080 1119> |
| **■ CAN0 Message Slot 2 Data 3 (C0MSL2DT3)** | <Address:H'0080 1129> |
| **■ CAN0 Message Slot 3 Data 3 (C0MSL3DT3)** | <Address:H'0080 1139> |
| **■ CAN0 Message Slot 4 Data 3 (C0MSL4DT3)** | <Address:H'0080 1149> |
| **■ CAN0 Message Slot 5 Data 3 (C0MSL5DT3)** | <Address:H'0080 1159> |
| **■ CAN0 Message Slot 6 Data 3 (C0MSL6DT3)** | <Address:H'0080 1169> |
| **■ CAN0 Message Slot 7 Data 3 (C0MSL7DT3)** | <Address:H'0080 1179> |
| **■ CAN0 Message Slot 8 Data 3 (C0MSL8DT3)** | <Address:H'0080 1189> |
| **■ CAN0 Message Slot 9 Data 3 (C0MSL9DT3)** | <Address:H'0080 1199> |
| **■ CAN0 Message Slot 10 Data 3 (C0MSL10DT3)** | <Address:H'0080 11A9> |
| **■ CAN0 Message Slot 11 Data 3 (C0MSL11DT3)** | <Address:H'0080 11B9> |
| **■ CAN0 Message Slot 12 Data 3 (C0MSL12DT3)** | <Address:H'0080 11C9> |
| **■ CAN0 Message Slot 13 Data 3 (C0MSL13DT3)** | <Address:H'0080 11D9> |
| **■ CAN0 Message Slot 14 Data 3 (C0MSL14DT3)** | <Address:H'0080 11E9> |
| **■ CAN0 Message Slot 15 Data 3 (C0MSL15DT3)** | <Address:H'0080 11F9> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT3 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | COMSLnDT3 | Message slot n data 3 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 3 or less, an indeterminate value is written to this register.

**■ CAN0 Message Slot 0 Data 4 (C0MSL0DT4)** <Address:H'0080 110A>
**■ CAN0 Message Slot 1 Data 4 (C0MSL1DT4)** <Address:H'0080 111A>
**■ CAN0 Message Slot 2 Data 4 (C0MSL2DT4)** <Address:H'0080 112A>
**■ CAN0 Message Slot 3 Data 4 (C0MSL3DT4)** <Address:H'0080 113A>
**■ CAN0 Message Slot 4 Data 4 (C0MSL4DT4)** <Address:H'0080 114A>
**■ CAN0 Message Slot 5 Data 4 (C0MSL5DT4)** <Address:H'0080 115A>
**■ CAN0 Message Slot 6 Data 4 (C0MSL6DT4)** <Address:H'0080 116A>
**■ CAN0 Message Slot 7 Data 4 (C0MSL7DT4)** <Address:H'0080 117A>
**■ CAN0 Message Slot 8 Data 4 (C0MSL8DT4)** <Address:H'0080 118A>
**■ CAN0 Message Slot 9 Data 4 (C0MSL9DT4)** <Address:H'0080 119A>
**■ CAN0 Message Slot 10 Data 4 (C0MSL10DT4)** <Address:H'0080 11AA>
**■ CAN0 Message Slot 11 Data 4 (C0MSL11DT4)** <Address:H'0080 11BA>
**■ CAN0 Message Slot 12 Data 4 (C0MSL12DT4)** <Address:H'0080 11CA>
**■ CAN0 Message Slot 13 Data 4 (C0MSL13DT4)** <Address:H'0080 11DA>
**■ CAN0 Message Slot 14 Data 4 (C0MSL14DT4)** <Address:H'0080 11EA>
**■ CAN0 Message Slot 15 Data 4 (C0MSL15DT4)** <Address:H'0080 11FA>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT4 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | COMSLnDT4 | Message slot n data 4 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 4 or less, an indeterminate value is written to this register.

| | |
|---|---|
| **■ CAN0 Message Slot 0 Data 5 (C0MSL0DT5)** | <Address:H'0080 110B> |
| **■ CAN0 Message Slot 1 Data 5 (C0MSL1DT5)** | <Address:H'0080 111B> |
| **■ CAN0 Message Slot 2 Data 5 (C0MSL2DT5)** | <Address:H'0080 112B> |
| **■ CAN0 Message Slot 3 Data 5 (C0MSL3DT5)** | <Address:H'0080 113B> |
| **■ CAN0 Message Slot 4 Data 5 (C0MSL4DT5)** | <Address:H'0080 114B> |
| **■ CAN0 Message Slot 5 Data 5 (C0MSL5DT5)** | <Address:H'0080 115B> |
| **■ CAN0 Message Slot 6 Data 5 (C0MSL6DT5)** | <Address:H'0080 116B> |
| **■ CAN0 Message Slot 7 Data 5 (C0MSL7DT5)** | <Address:H'0080 117B> |
| **■ CAN0 Message Slot 8 Data 5 (C0MSL8DT5)** | <Address:H'0080 118B> |
| **■ CAN0 Message Slot 9 Data 5 (C0MSL9DT5)** | <Address:H'0080 119B> |
| **■ CAN0 Message Slot 10 Data 5 (C0MSL10DT5)** | <Address:H'0080 11AB> |
| **■ CAN0 Message Slot 11 Data 5 (C0MSL11DT5)** | <Address:H'0080 11BB> |
| **■ CAN0 Message Slot 12 Data 5 (C0MSL12DT5)** | <Address:H'0080 11CB> |
| **■ CAN0 Message Slot 13 Data 5 (C0MSL13DT5)** | <Address:H'0080 11DB> |
| **■ CAN0 Message Slot 14 Data 5 (C0MSL14DT5)** | <Address:H'0080 11EB> |
| **■ CAN0 Message Slot 15 Data 5 (C0MSL15DT5)** | <Address:H'0080 11FB> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT5 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8-15 | COMSLnDT5 | Message slot n data 5 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 5 or less, an indeterminate value is written to this register.

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Data 6 (C0MSL0DT6)** | <Address:H'0080 110C> |
| ■ **CAN0 Message Slot 1 Data 6 (C0MSL1DT6)** | <Address:H'0080 111C> |
| ■ **CAN0 Message Slot 2 Data 6 (C0MSL2DT6)** | <Address:H'0080 112C> |
| ■ **CAN0 Message Slot 3 Data 6 (C0MSL3DT6)** | <Address:H'0080 113C> |
| ■ **CAN0 Message Slot 4 Data 6 (C0MSL4DT6)** | <Address:H'0080 114C> |
| ■ **CAN0 Message Slot 5 Data 6 (C0MSL5DT6)** | <Address:H'0080 115C> |
| ■ **CAN0 Message Slot 6 Data 6 (C0MSL6DT6)** | <Address:H'0080 116C> |
| ■ **CAN0 Message Slot 7 Data 6 (C0MSL7DT6)** | <Address:H'0080 117C> |
| ■ **CAN0 Message Slot 8 Data 6 (C0MSL8DT6)** | <Address:H'0080 118C> |
| ■ **CAN0 Message Slot 9 Data 6 (C0MSL9DT6)** | <Address:H'0080 119C> |
| ■ **CAN0 Message Slot 10 Data 6 (C0MSL10DT6)** | <Address:H'0080 11AC> |
| ■ **CAN0 Message Slot 11 Data 6 (C0MSL11DT6)** | <Address:H'0080 11BC> |
| ■ **CAN0 Message Slot 12 Data 6 (C0MSL12DT6)** | <Address:H'0080 11CC> |
| ■ **CAN0 Message Slot 13 Data 6 (C0MSL13DT6)** | <Address:H'0080 11DC> |
| ■ **CAN0 Message Slot 14 Data 6 (C0MSL14DT6)** | <Address:H'0080 11EC> |
| ■ **CAN0 Message Slot 15 Data 6 (C0MSL15DT6)** | <Address:H'0080 11FC> |

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT6 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | COMSLnDT6 | Message slot n data 6 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 6 or less, an indeterminate value is written to this register.

| | |
|---|---|
| ■ **CAN0 Message Slot 0 Data 7 (C0MSL0DT7)** | <Address:H'0080 110D> |
| ■ **CAN0 Message Slot 1 Data 7 (C0MSL1DT7)** | <Address:H'0080 111D> |
| ■ **CAN0 Message Slot 2 Data 7 (C0MSL2DT7)** | <Address:H'0080 112D> |
| ■ **CAN0 Message Slot 3 Data 7 (C0MSL3DT7)** | <Address:H'0080 113D> |
| ■ **CAN0 Message Slot 4 Data 7 (C0MSL4DT7)** | <Address:H'0080 114D> |
| ■ **CAN0 Message Slot 5 Data 7 (C0MSL5DT7)** | <Address:H'0080 115D> |
| ■ **CAN0 Message Slot 6 Data 7 (C0MSL6DT7)** | <Address:H'0080 116D> |
| ■ **CAN0 Message Slot 7 Data 7 (C0MSL7DT7)** | <Address:H'0080 117D> |
| ■ **CAN0 Message Slot 8 Data 7 (C0MSL8DT7)** | <Address:H'0080 118D> |
| ■ **CAN0 Message Slot 9 Data 7 (C0MSL9DT7)** | <Address:H'0080 119D> |
| ■ **CAN0 Message Slot 10 Data 7 (C0MSL10DT7)** | <Address:H'0080 11AD> |
| ■ **CAN0 Message Slot 11 Data 7 (C0MSL11DT7)** | <Address:H'0080 11BD> |
| ■ **CAN0 Message Slot 12 Data 7 (C0MSL12DT7)** | <Address:H'0080 11CD> |
| ■ **CAN0 Message Slot 13 Data 7 (C0MSL13DT7)** | <Address:H'0080 11DD> |
| ■ **CAN0 Message Slot 14 Data 7 (C0MSL14DT7)** | <Address:H'0080 11ED> |
| ■ **CAN0 Message Slot 15 Data 7 (C0MSL15DT7)** | <Address:H'0080 11FD> |

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| C0MSLnDT7 | | | | | | | |

<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-7 | COMSLnDT7 | Message slot n data 7 | ○ | ○ |

These registers are the transmit frame/receive frame memory space.

Note: • For receive slots, if when storing a data frame the data length (DLC value) = 7 or less, an indeterminate value is written to this register.

■ **CAN0 Message Slot 0 Time Stamp (C0MSL0TSP)** <Address:H'0080 110E>

■ **CAN0 Message Slot 1 Time Stamp (C0MSL1TSP)** <Address:H'0080 111E>

■ **CAN0 Message Slot 2 Time Stamp (C0MSL2TSP)** <Address:H'0080 112E>

■ **CAN0 Message Slot 3 Time Stamp (C0MSL3TSP)** <Address:H'0080 113E>

■ **CAN0 Message Slot 4 Time Stamp (C0MSL4TSP)** <Address:H'0080 114E>

■ **CAN0 Message Slot 5 Time Stamp (C0MSL5TSP)** <Address:H'0080 115E>

■ **CAN0 Message Slot 6 Time Stamp (C0MSL6TSP)** <Address:H'0080 116E>

■ **CAN0 Message Slot 7 Time Stamp (C0MSL7TSP)** <Address:H'0080 117E>

■ **CAN0 Message Slot 8 Time Stamp (C0MSL8TSP)** <Address:H'0080 118E>

■ **CAN0 Message Slot 9 Time Stamp (C0MSL9TSP)** <Address:H'0080 119E>

■ **CAN0 Message Slot 10 Time Stamp (C0MSL10TSP)** <Address:H'0080 11AE>

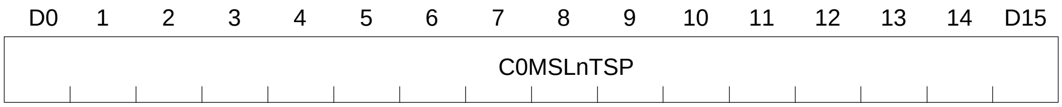
■ **CAN0 Message Slot 11 Time Stamp (C0MSL11TSP)** <Address:H'0080 11BE>

■ **CAN0 Message Slot 12 Time Stamp (C0MSL12TSP)** <Address:H'0080 11CE>

■ **CAN0 Message Slot 13 Time Stamp (C0MSL13TSP)** <Address:H'0080 11DE>

■ **CAN0 Message Slot 14 Time Stamp (C0MSL14TSP)** <Address:H'0080 11EE>

■ **CAN0 Message Slot 15 Time Stamp (C0MSL15TSP)** <Address:H'0080 11FE>



<When reset: Indeterminate>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0-15 | COMSLnTSP | Message slot n time stamp | ○ | ○ |

These registers are the transmit frame/receive frame memory space. When the CAN module finishes transmitting or receiving, the CAN0 Time Stamp Count Register value is set in this register.

# 13.3 CAN Protocol

## 13.3.1 CAN Protocol Frame

There are four types of frames which are handled by CAN protocol:

(1) Data frame
(2) Remote frame
(3) Error frame
(4) Overload frame

Frames are separated from each other by an interframe space.

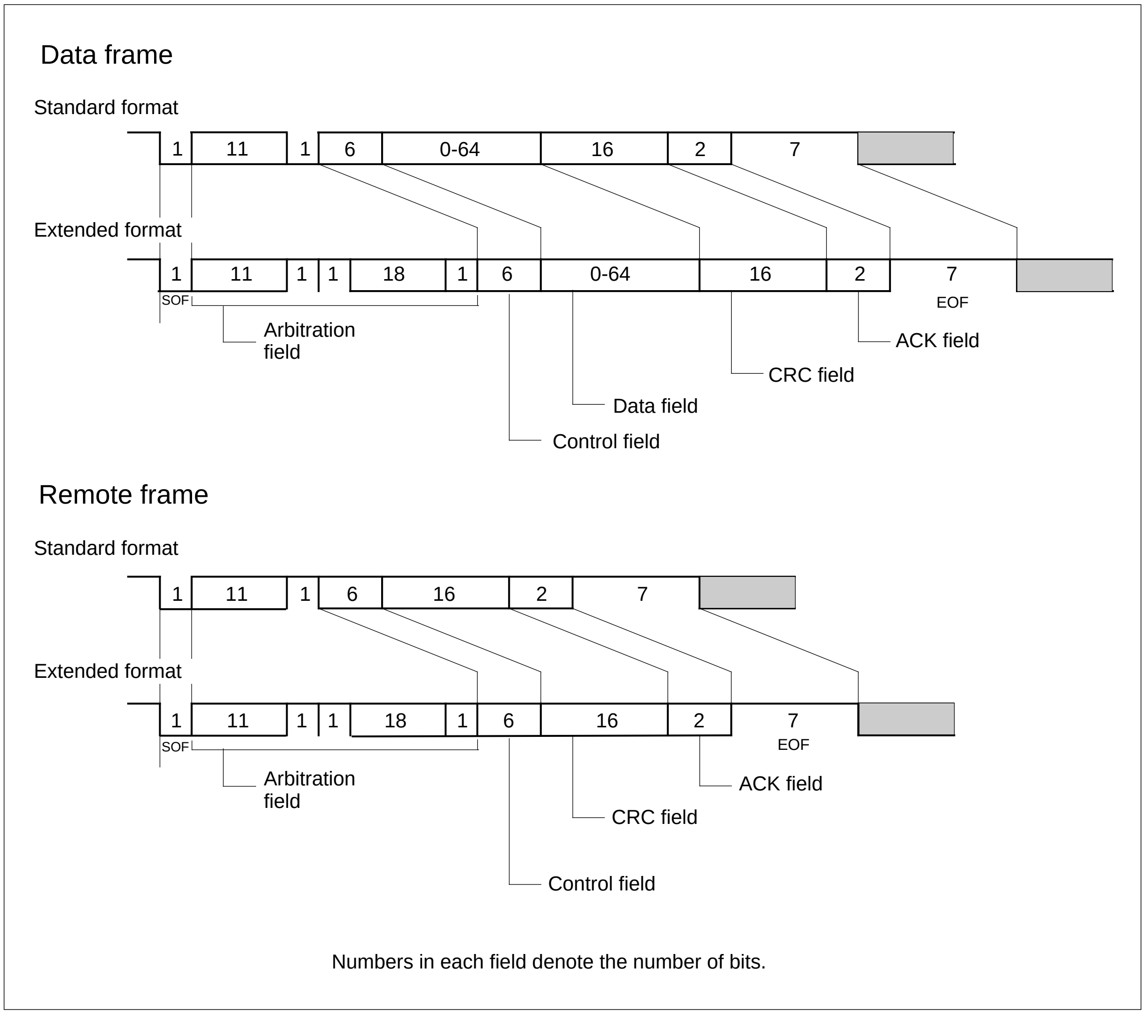


**Figure 13.3.1 CAN Protocol Frames (1)**

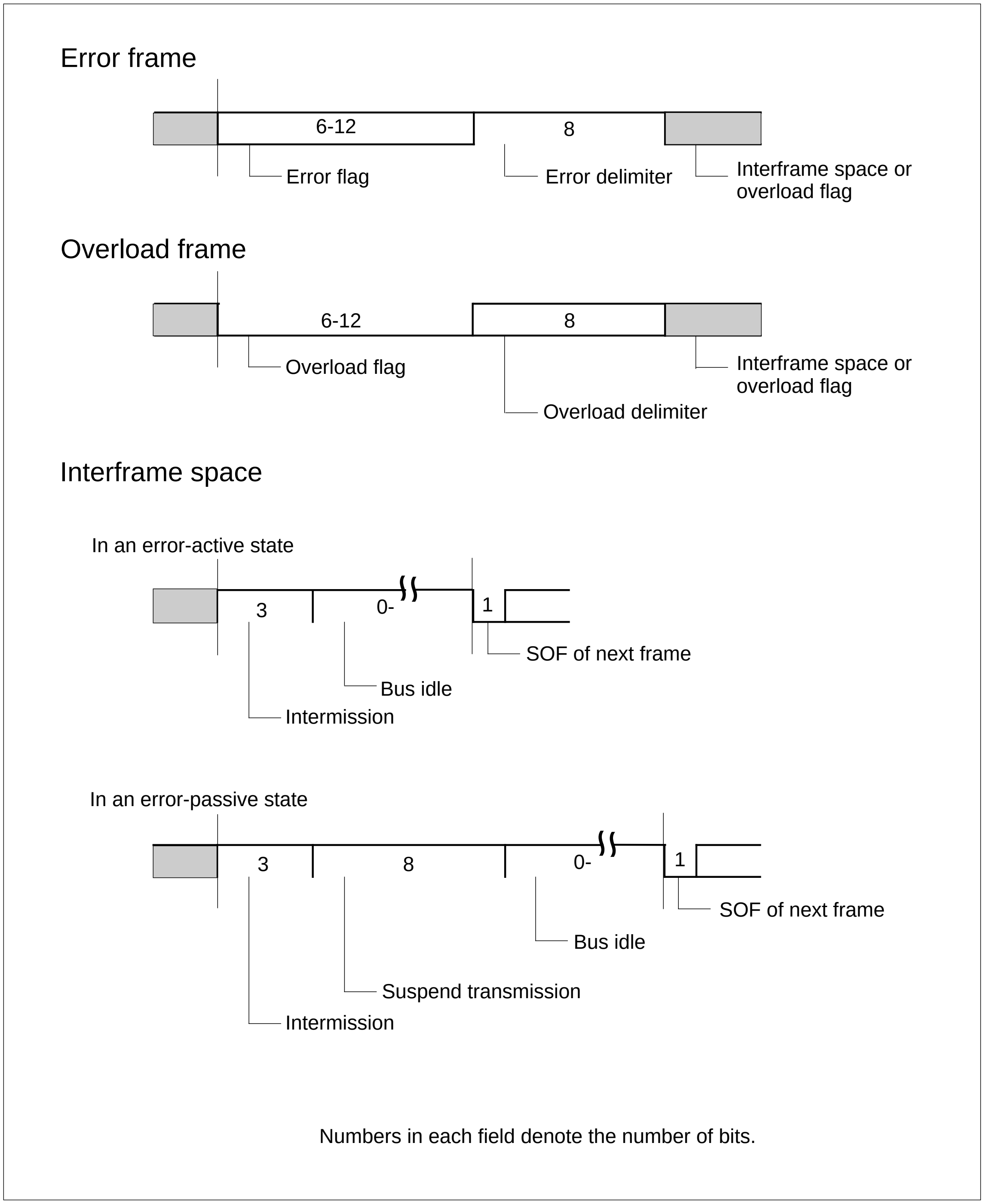


**Figure 13.3.2 CAN Protocol Frames (2)**

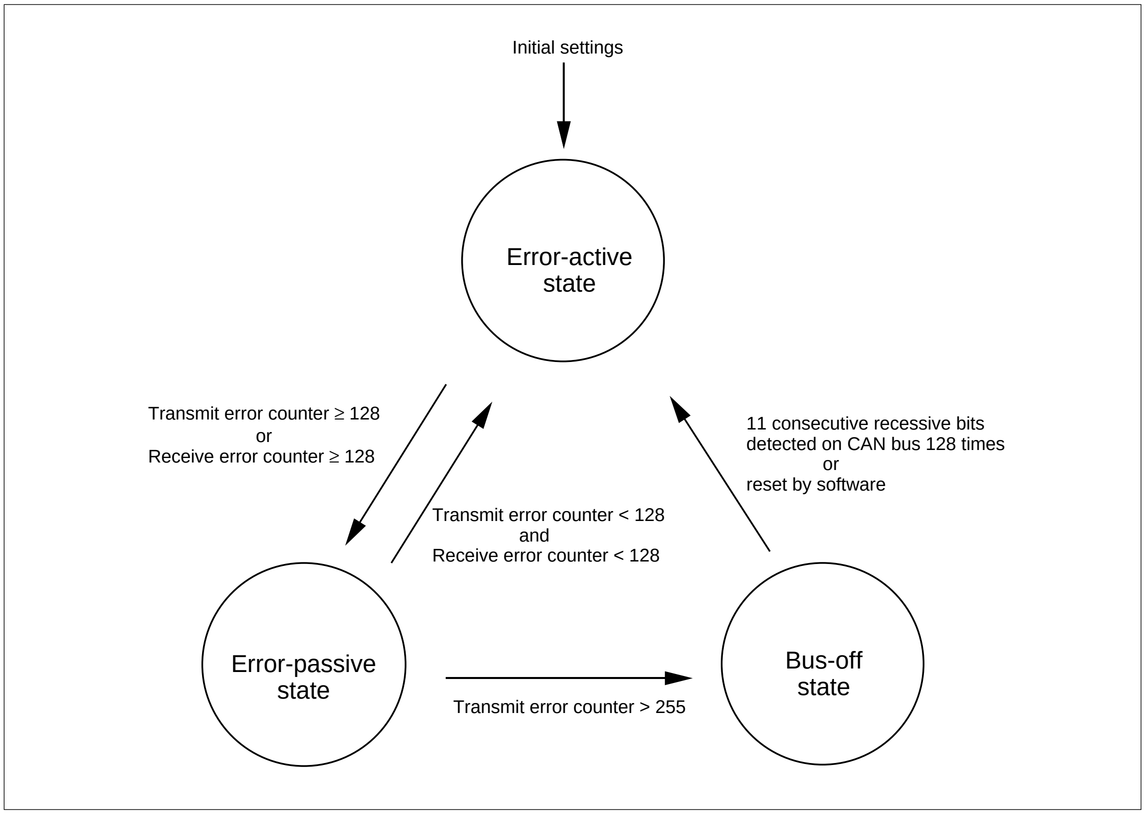

Initial settings
Error-active state
Transmit error counter ≥ 128
or
Receive error counter ≥ 128
Transmit error counter < 128
and
Receive error counter < 128
11 consecutive recessive bits
detected on CAN bus 128 times
or
reset by software
Error-passive state
Bus-off state
Transmit error counter > 255

**Figure 13.3.3 CAN Control Error States**

The CAN controller assumes one of the following three error states depending on the transmit error and receive error counter values.

(1) Error-active state
- This is a state where almost no errors have occurred.
- When an error is detected, an active error flag is transmitted.
- Immediately after being initialized, the CAN controller is in this state.

(2) Error-passive state
- This is a state where many errors have occurred.
- When an error is detected, a passive error flag is transmitted.

(3) Bus-off state
- This is a state where a large number of errors have occurred.
- CAN communication with other nodes cannot be performed until the CAN module returns to an error-active state.

| Error status of the unit | Transmit error counter | | Receive error counter |
|---|---|---|---|
| Error-active state | 0 -127 | and | 0 - 127 |
| Error-passive state | 128 - 255 | or | 128 - |
| Bus-off state | 256 - | | – |

## 13.4 Initializing the CAN Module

### 13.4.1 Initialization of the CAN Module

Before you perform communication, set up the CAN module as described below.

(1) Selecting pin functions

The CAN transmit data output pin (CTX) and CAN data receive input pin (CRX) are shared with input/output ports, so be sure to select the functions of these pins. (Refer to Chapter 8, "Input/ Output Ports and Pin Functions."

(2) Setting the interrupt controller (ICU)

When you use CAN module interrupts, set the interrupt priority.

(3) Setting CAN Error Interrupt Mask and CAN Slot Interrupt Mask Registers

When you use CAN bus error interrupts, CAN error passive interrupts, CAN error bus-off interrupts, or CAN slot interrupts, set each corresponding bit to 1 to enable interrupt requests.

(4) Setting bit timing and the number of times sampled

Using the CAN Configuration Register and CAN Baud Rate Prescaler, set the bit timing and the number of times the CAN bus is sampled.

1) Setting the bit timing

Determine the period Tq that is the base of bit timing, the configuration of Propagation Segment, Phase Segment1, and Phase Segment2, and reSynchronization Jump Width. The equation to calculate Tq is shown below.

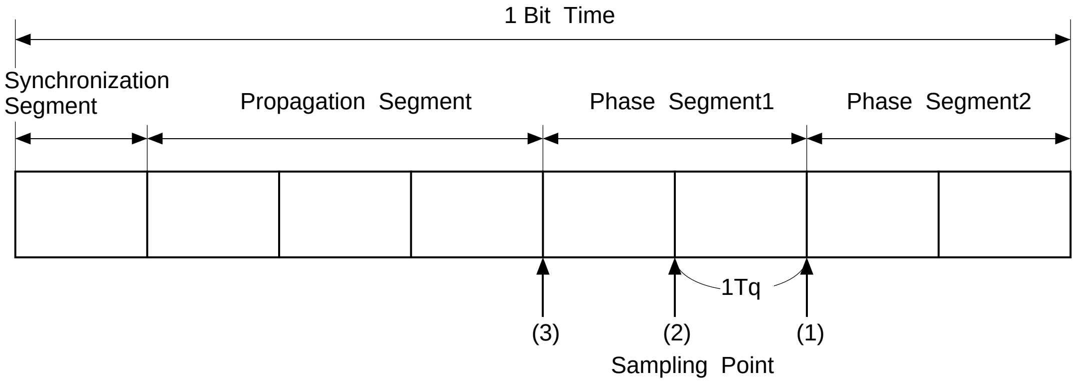
$$\text{Tq} = (\text{BRP}+1) / \text{CPU clock}$$

The baud rate is determined by the number of Tq's that comprise one bit. The equation to calculate the baud rate is shown below.

$$\text{Baud rate (bps)} = \frac{1}{\text{Tq period} \times \text{number of Tq's for 1 bit}}$$

$$\begin{aligned}\text{Number of Tq's for 1 bit} = {} & \text{Synchronization Segment} + \\ & \text{Propagation Segment} + \\ & \text{Phase Segment 1} + \\ & \text{Phase Segment 2}\end{aligned}$$

Note: • The maximum communicatable baud rate depends on the system configuration (e.g., bus length, clock error, CAN bus transceiver, sampling position, and bit configuration). Please consider the system configuration when setting the baud rate and the number of Tq’s.



- Shown in this diagram is the bit timing for cases where one bit consists of 8 Tq's.
- When one-time sampling is selected, the value sampled at Sampling Point (1) is assumed to be the value of the bit.
- When three-time sampling is selected, the value of the bit is determined by majority from CAN bus values sampled at Sampling Points (1), (2), and (3).

**Figure 13.4.1 Example of Bit Timing**

2) Setting the number of times sampled

Select the number of times the CAN bus is sampled from "one time" and "three times."

- When you select one-time sampling, the value sampled at the end of Phase Segment1 is assumed to be the value of the bit.
- When you select three-time sampling, the value of the bit is determined by majority from values sampled at three points, i.e., the value sampled at the first point and those sampled one Tq before and two Tq's before that.

(5) Setting ID Mask Registers

Set the values of ID Mask Registers (Global Mask Register, Local Mask Register A, and Local Mask Register B) which are used in acceptance filtering of received messages.

(6) Settings when running in BasicCAN mode

- Set the CAN Extended ID Register IDE14 and IDE15 bits. (We recommend setting the same value in these bits.)
- Set IDs for message slots 14 and 15.
- Set the Message Control Registers 14 and 15 for data frame reception (H'40).

(7) Setting CAN module operation mode

Using the CAN Control Register (CAN0CNT), select the CAN module's operation mode (BasicCAN or loopback mode) and the clock source for the time stamp counter.

(8) Releasing the CAN module from reset

After you finished settings (1) through (7) above, clear the CAN Control Register (CAN0CNT)'s forcible reset bit (FRST) and reset bit (RST) to 0. Then, after detecting 11 consecutive "recessive" bits on the CAN bus, the CAN module becomes ready to communicate.

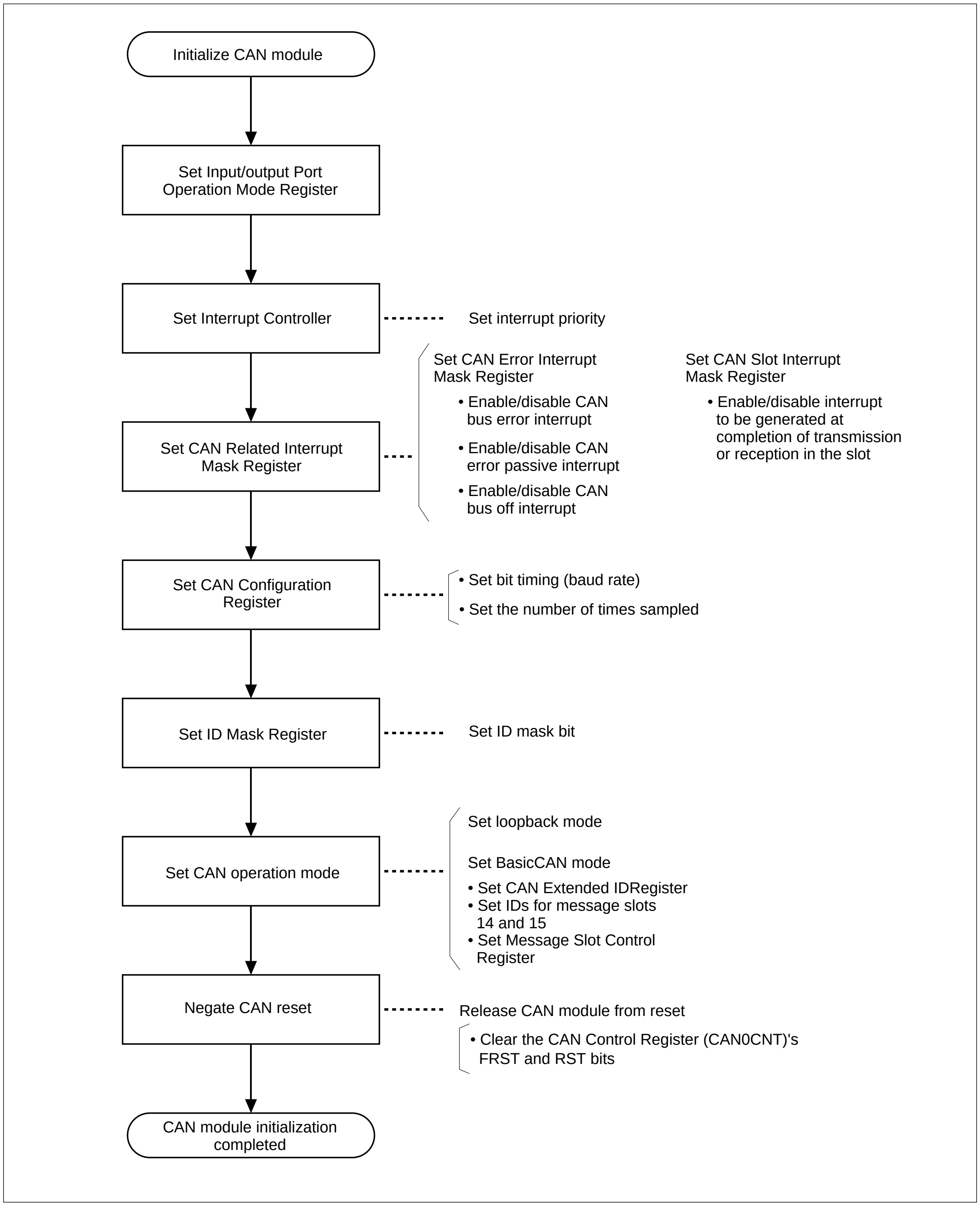

Initialize CAN module
Set Input/output Port Operation Mode Register
Set Interrupt Controller
Set interrupt priority
Set CAN Related Interrupt Mask Register
Set CAN Error Interrupt Mask Register
• Enable/disable CAN bus error interrupt
• Enable/disable CAN error passive interrupt
• Enable/disable CAN bus off interrupt
Set CAN Slot Interrupt Mask Register
• Enable/disable interrupt to be generated at completion of transmission or reception in the slot
Set CAN Configuration Register
• Set bit timing (baud rate)
• Set the number of times sampled
Set ID Mask Register
Set ID mask bit
Set CAN operation mode
Set loopback mode
Set BasicCAN mode
• Set CAN Extended IDRegister
• Set IDs for message slots 14 and 15
• Set Message Slot Control Register
Negate CAN reset
Release CAN module from reset
• Clear the CAN Control Register (CAN0CNT)'s FRST and RST bits
CAN module initialization completed

**Figure 13.4.2 Initializing the CAN Module**

## 13.5 Transmitting Data Frames

### 13.5.1 Data Frame Transmit Procedure

The following describes the procedure for transmitting data frames.

(1) Initializing the CAN Message Slot Control Register

Initialize the CAN Message Slot Control Register for the slot in which you want to transmit by writing H'00 to the register.

(2) Confirming that transmission is idle

Read the initialized CAN Message Slot Control Register and check the TRSTAT (transmit/ receive status) bit to see that CAN has stopped sending or receiving. If this bit = 1, it means that the CAN module is accessing the message slot, so you need to wait until the bit is cleared.

(3) Setting transmit data

Set the transmit ID and transmit data in the message slot.

(4) Setting the Extended ID Register

Set the corresponding bit of the Extended ID Register to 0 when you want to transmit the data as a standard frame or 1 when you want to transmit the data as an extended frame.

(5) Setting the CAN Message Slot Control Register

Write H'80 (Note 1) to the CAN Message Slot Control Register to set the TR (Transmit Request) bit to 1.

Note 1: When you are transmitting a data frame, always write H'80 to this register.

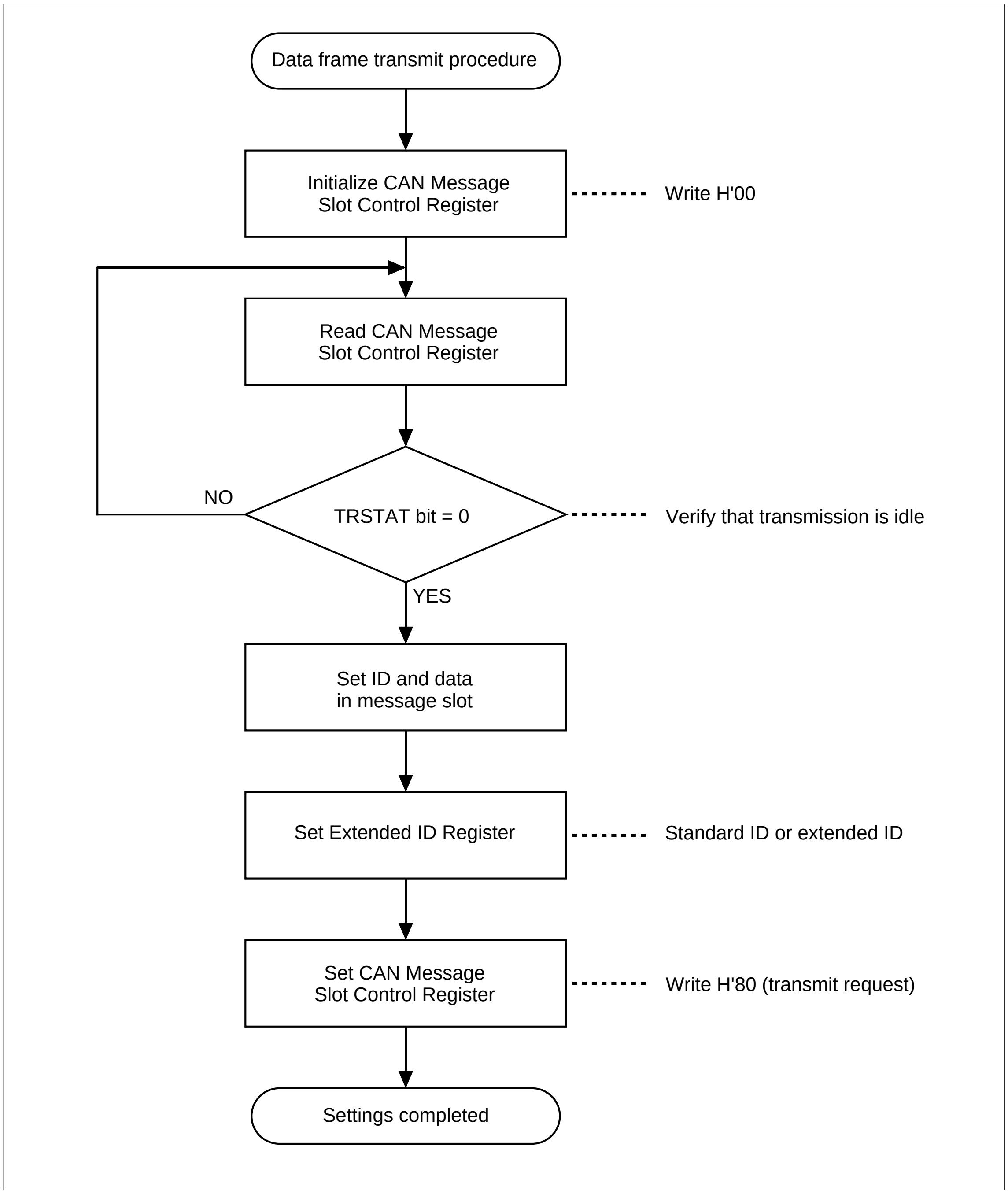

Data frame transmit procedure
Initialize CAN Message Slot Control Register
Write H'00
Read CAN Message Slot Control Register
NO
TRSTAT bit = 0
Verify that transmission is idle
YES
Set ID and data in message slot
Set Extended ID Register
Standard ID or extended ID
Set CAN Message Slot Control Register
Write H'80 (transmit request)
Settings completed

**Figure 13.5.1 Data Frame Transmit Procedure**

### 13.5.2 Data Frame Transmit Operation

The following describes data frame transmit operation. The operations described below are automatically performed in hardware.

(1) Selecting a transmit frame

The CAN module checks slots which have transmit requests (including remote frame transmit slots) every intermission to determine the frame to transmit. If there are multiple transmit slots, frames are transmitted in order of slot numbers beginning with the smallest.

(2) Transmitting a data frame

After determining the transmit slot, the CAN module sets the corresponding CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 1, thereby starting transmission.

(3) If the CAN module lost bus arbitration or a CAN bus error occurs

If the CAN module lost bus arbitration or a CAN bus error occurs while transmitting, the CAN module clears the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 0. If the CAN module requested a transmit abort, the transmit abort is accepted and writing to the message slot is enabled.

(4) Completion of data frame transmission

When data frame transmission is completed, the CAN Message Slot Control Register's TRFIN (Transmit/Receive Finished) bit and the CAN Slot Interrupt Status Register are set to 1. Also, a time stamp count value at the time transmission was completed is written to the CAN Message Slot Time Stamp (C0MSLnTSP), and the transmit operation is thereby completed.
If the CAN slot interrupt has been enabled, an interrupt request is generated at completion of transmit operation. The slot which has had transmission completed goes to an inactive state and remains inactive (neither transmit nor receive) until it is newly set in software.

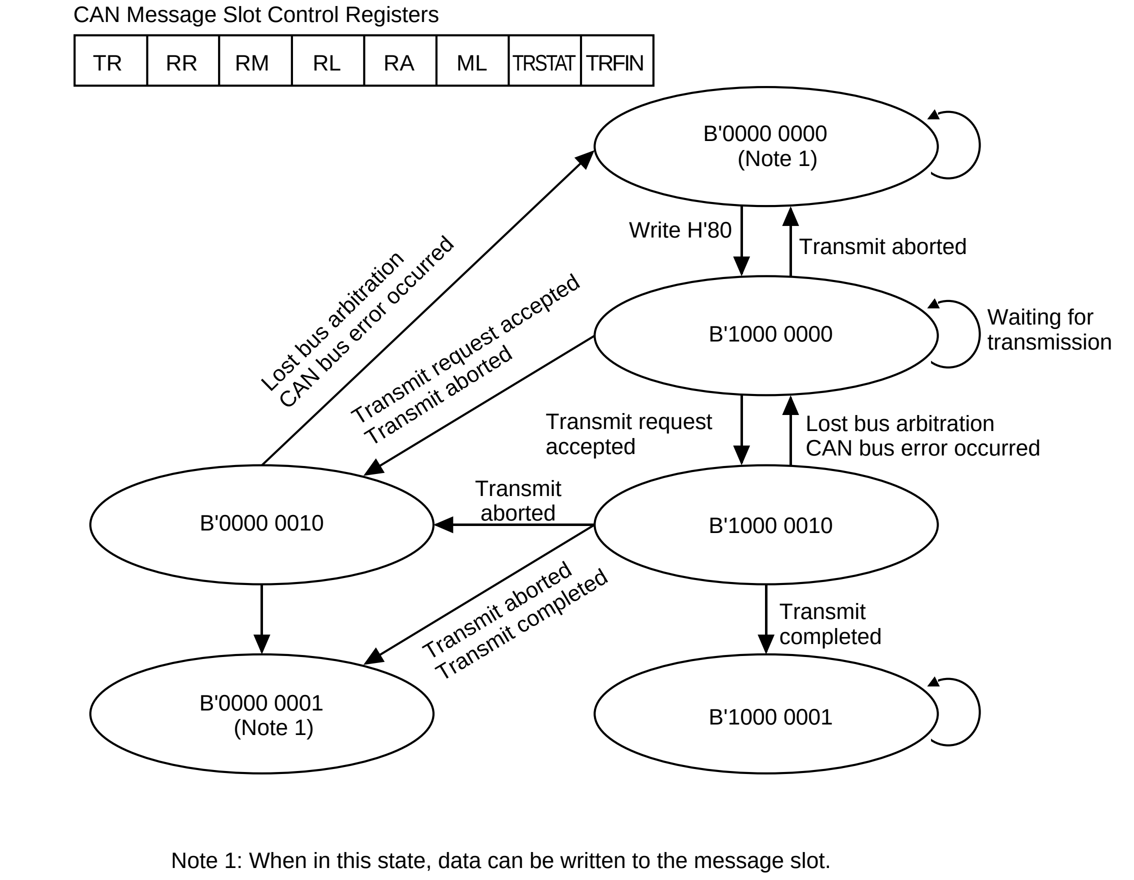


**Figure 13.5.2 Operation of the CAN Message Slot Control Register when Transmitting Data Frames**

### 13.5.3 Transmit Abort Function

The transmit abort function is used to cancel a transmit request that has once been set. This is accomplished by writing H'0F to the CAN Message Slot Control Register for the slot concerned. When transmit abort is accepted, the CAN module clears the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 0, allowing for data to be written to the message slot. The following shows conditions under which transmit abort is accepted:

**[Conditions]**

- When the target message is waiting for transmission
- When a CAN bus error occurs during transmission
- When the CAN module lost bus arbitration

## 13.6 Receiving Data Frames

### 13.6.1 Data Frame Receive Procedure

The following describes the procedure for receiving data frames.

(1) Initializing the CAN Message Slot Control Register

Initialize the CAN Message Slot Control Register for the slot in which you want to receive by writing H'00 to the register.

(2) Confirming that reception is idle

Read the CAN Message Slot Control Register after being initialized and check the TRSTAT (Transmit/Receive Status) bit to see that reception has stopped and remains idle. If this bit = 1, it means that the CAN module is accessing the message slot, so you need to wait until the bit is cleared.

(3) Setting the receive ID

Set the ID you want to receive in the message slot.

(4) Setting the Extended ID Register

Set the corresponding bit of the Extended ID Register to 0 when you want to receive a standard frame or 1 when you want to receive an extended frame.

(5) Setting the CAN Message Slot Control Register

Write H'40 to the CAN Message Slot Control Register to set the RR (Receive Request) bit to 1.

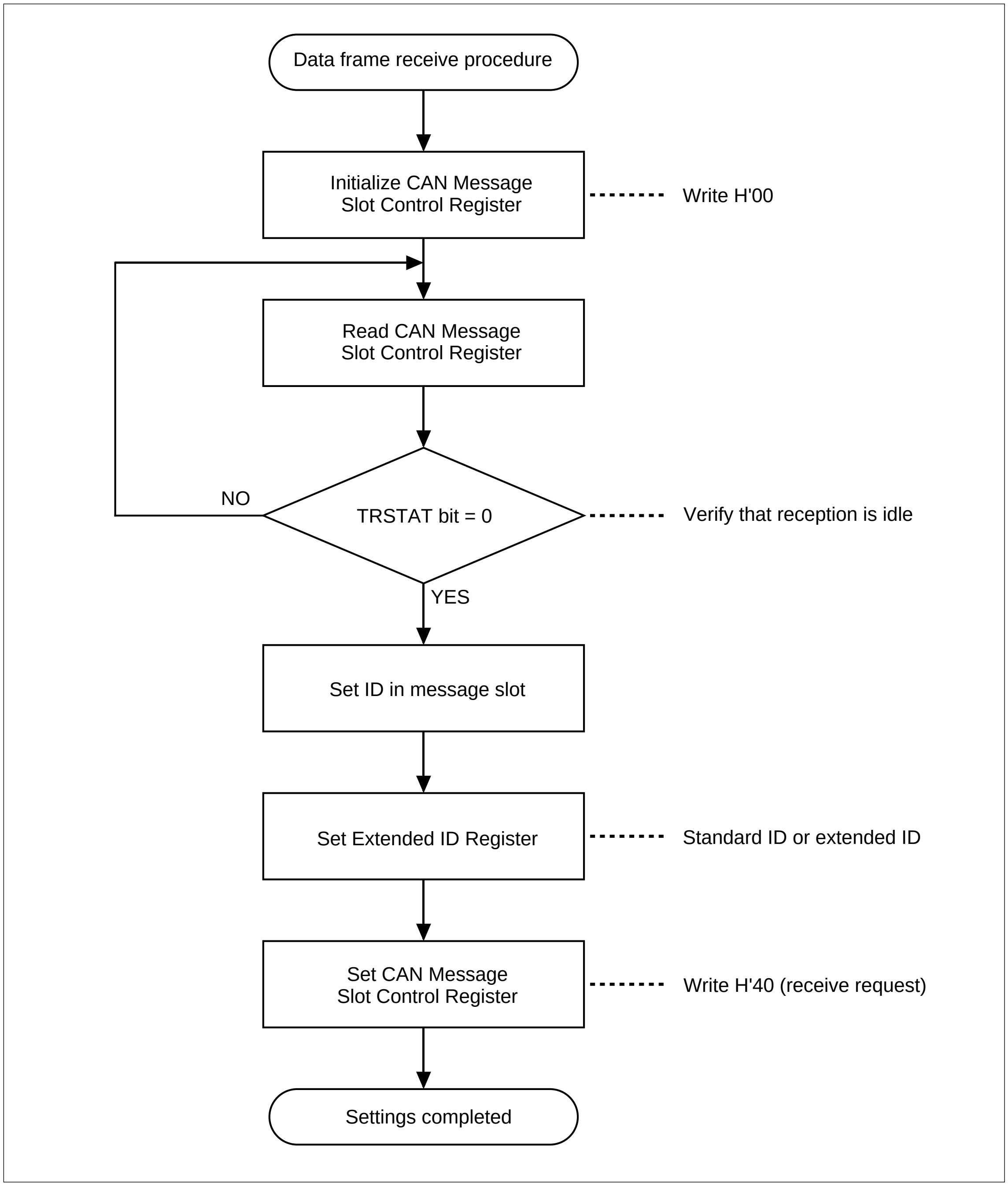

Data frame receive procedure
Initialize CAN Message Slot Control Register
Write H'00
Read CAN Message Slot Control Register
TRSTAT bit = 0
NO
YES
Verify that reception is idle
Set ID in message slot
Set Extended ID Register
Standard ID or extended ID
Set CAN Message Slot Control Register
Write H'40 (receive request)
Settings completed

**Figure 13.6.1 Data Frame Receive Procedure**

### 13.6.2 Data Frame Receive Operation

The following describes data frame receive operation. The operations described below are automatically performed in hardware.

(1) Acceptance filtering

When the CAN module finished receiving data, it starts searching for the slot that satisfies conditions for receiving the received message sequentially from slot 0 (up to slot 15). The following shows receive conditions for slots that have been set for data frame reception.

**[Conditions]**

- The receive frame is a data frame.
- The receive ID and the slot ID are identical, assuming the ID Mask Register bits set to 0 are "Don't care bits."
- The standard and extended frame types are the same.

Note: • In BasicCAN mode, slots 14 and 15 while being set for data frame reception can also receive remote frames.

(2) When receive conditions are met

When receive conditions in (1) above are met, the CAN module sets the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) and TRFIN (Transmit/Receive Finished) bits to 1 while at the same time writing the received data to the message slot. If the TRFIN (Transmit/Receive Finished) bit is already 1, the CAN module also sets the ML (Message Lost) bit to 1, indicating that the message slot has been overwritten. The message slot has its ID field and DLC field both overwritten and an indeterminate value written in its unused area (e.g., extended ID field for standard frame reception and an unused data field).
Furthermore, a time stamp count value at the time the message was received is written to the CAN Message Slot Time Stamp (C0MSLnTSP) along with the received data. When the CAN module finished writing to the message slot, it sets the CAN Slot Interrupt Status bit to 1. If the interrupt for the slot has been enabled, an interrupt request is generated, and the slot goes to a wait state for the next reception.

(3) When receive conditions are not met

The received frame is discarded, and the CAN module goes to the next transmit/receive operation without writing to the message slot.

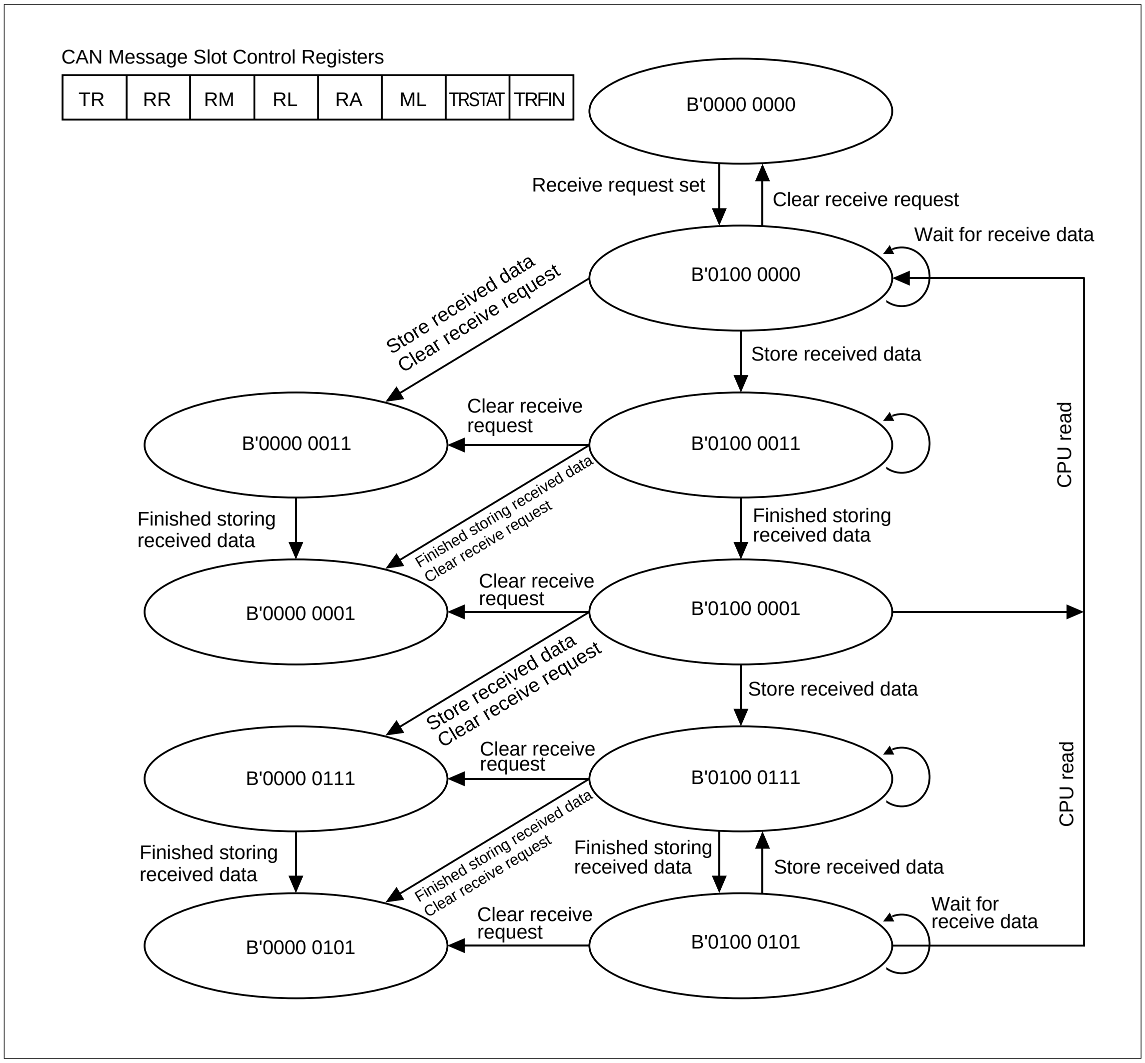


**Figure 13.6.2 Operation of the CAN Message Slot Control Register when Receiving Data Frames**

### 13.6.3 Reading Out Received Data Frames

The following describes the procedure for reading out received data frames from the slot.

(1) Clearing the TRFIN (Transmit/Receive Finished) bit

Write H'4E, H'40 or H'00 to the CAN Message Control Register (C0MSLnCNT) to clear the TRFIN bit to 0. After this write, the slot operates as follows:

| Value written to C0MSLnCNT | Slot operation after write |
|---|---|
| H'4E | Operates as a data frame receive slot.<br>Overwrite can be verified by ML bit. |
| H'40 | Operates as a data frame receive slot.<br>Overwrite cannot be verified by ML bit. |
| H'00 | The slot stops transmit/receive operation. |

Notes: • If message-lost check by the ML bit is needed, write H'4E to the C0MSLnCNT register as you clear the TRFIN bit.

• If you clear the TRFIN bit by writing H'4E, H'40 or H'00, it is possible that new data will be stored in the slot while still reading a message from the slot.

(2) Reading out from the message slot

Read out a message from the message slot.

(3) Checking the TRFIN (Transmit/Receive Finished) bit

Read the CAN Message Control Register to check the TRFIN (Transmit/Receive Finished) bit.

1) When TRFIN (Transmit/Receive Finished) bit = 1
It means that new data was stored in the slot while still reading out from the slot in (2). In this case, the data read out in (2) may contain an indeterminate value. Therefore, reexecute beginning with clearing of the TRFIN (Transmit/Receive Finished) bit in (1).

2) When TRFIN (Transmit/Receive Finished) bit = 0
It means that the CAN module finished reading out from the slot normally.

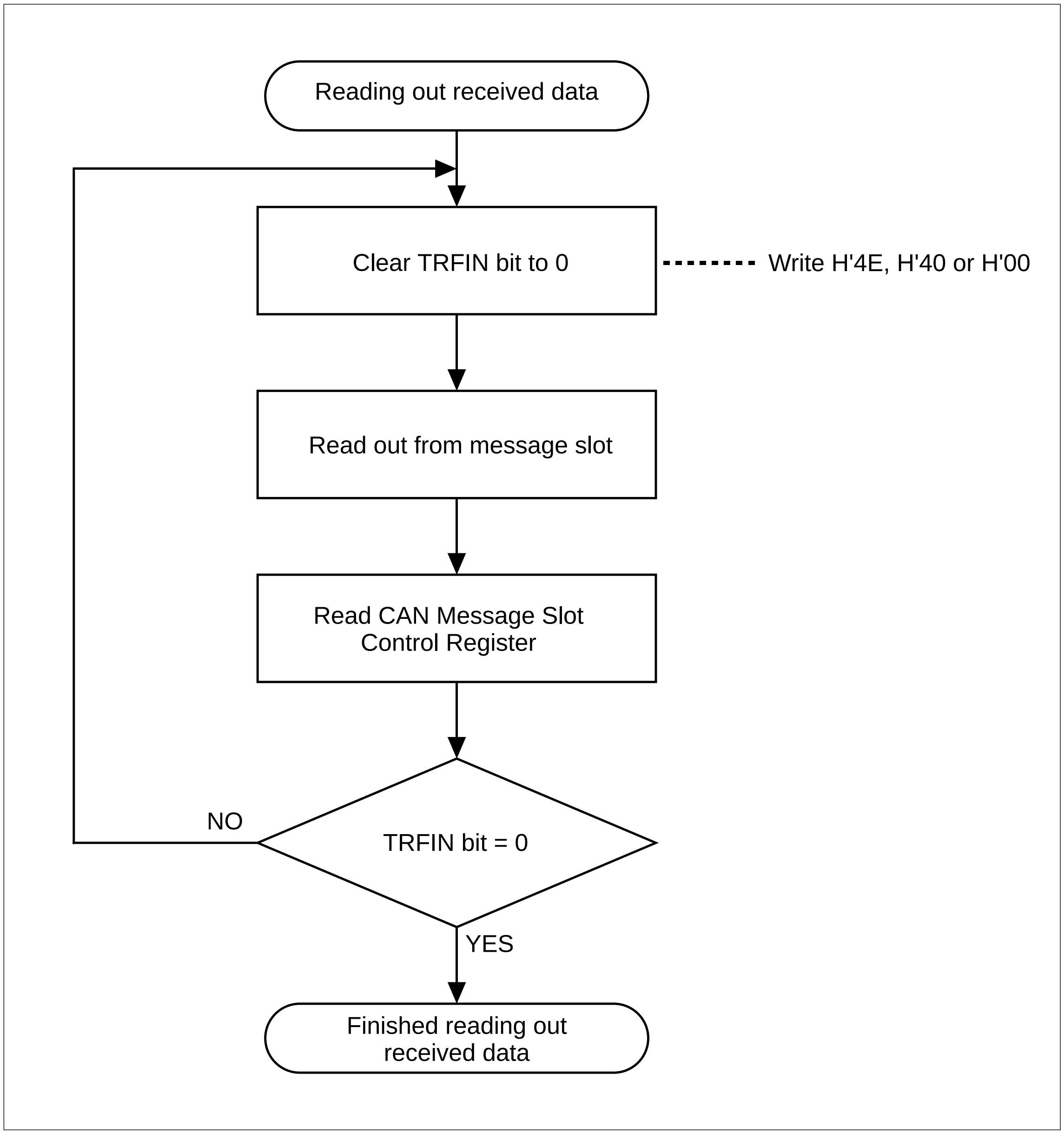

Reading out received data
Clear TRFIN bit to 0
Write H'4E, H'40 or H'00
Read out from message slot
Read CAN Message Slot Control Register
TRFIN bit = 0
NO
YES
Finished reading out received data

**Figure 13.6.3 Procedure for Reading Out Received Data**

## 13.7 Transmitting Remote Frames

### 13.7.1 Remote Frame Transmit Procedure

The following describes the procedure for transmitting remote frames.

(1) Initializing the CAN Message Slot Control Register

Initialize the CAN Message Slot Control Register for the slot in which you want to transmit by writing H'00 to the register.

(2) Confirming that transmission is idle

Read the CAN Message Slot Control Register after being initialized and check the TRSTAT (Transmit/Receive Status) bit to see that transmission has stopped and remains idle. If this bit = 1, it means that the CAN module is accessing the message slot, so you need to wait until the bit is cleared.

(3) Setting transmit ID

Set the ID to be transmitted in the message slot.

(4) Setting the Extended ID Register

Set the corresponding bit of the Extended ID Register to 0 when you want to transmit the frame as a standard frame or 1 when you want to transmit the frame as an extended frame.

(5) Setting the CAN Message Slot Control Register

Write H'A0 to the CAN Message Slot Control Register to set the TR (Transmit Request) and RM (Remote) bits to 1.

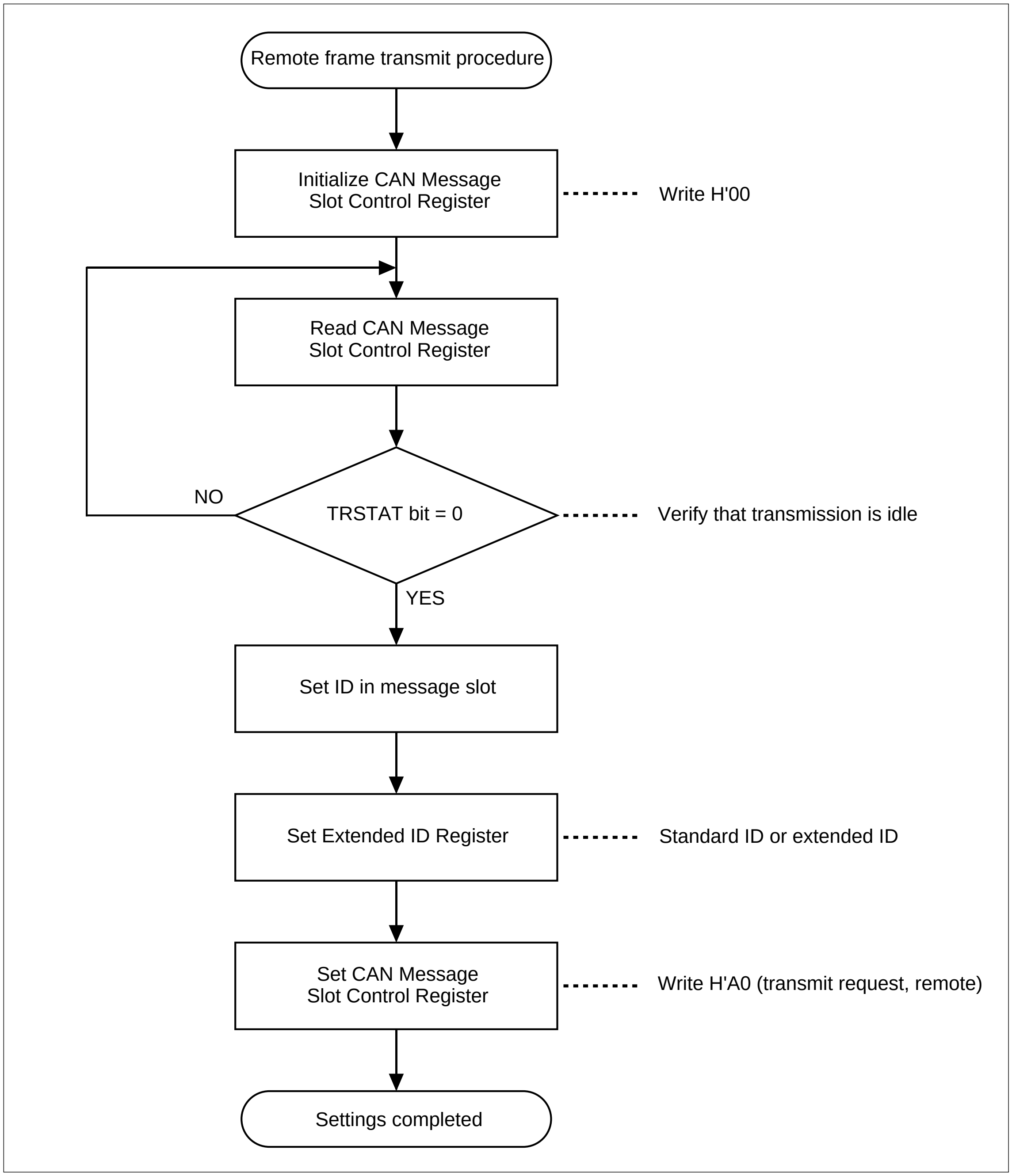

Remote frame transmit procedure
Initialize CAN Message Slot Control Register
Write H'00
Read CAN Message Slot Control Register
NO
TRSTAT bit = 0
Verify that transmission is idle
YES
Set ID in message slot
Set Extended ID Register
Standard ID or extended ID
Set CAN Message Slot Control Register
Write H'A0 (transmit request, remote)
Settings completed

**Figure 13.7.1 Remote Frame Transmit Procedure**

### 13.7.2 Remote Frame Transmit Operation

The following describes remote frame transmit operation. The operations described below are automatically performed in hardware.

(1) Setting the RA (Remote Active) bit

At the same time H'A0 (Transmit Request, Remote) is written to the CAN Message Slot Control Register, the RA (Remote Active) bit is set to 1, indicating that the corresponding slot is to handle remote frames.

(2) Selecting a transmit frame

The CAN module checks slots which have transmit requests (including data frame transmit slots) every intermission to determine the frame to transmit. If there are multiple transmit slots, frames are transmitted in order of slot numbers beginning with the smallest.

(3) Transmitting a remote frame

After determining the transmit slot, the CAN module sets the corresponding CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 1, thereby starting transmission.

(4) If the CAN module lost bus arbitration or a CAN bus error occurs

If the CAN module lost bus arbitration or a CAN bus error occurs while transmitting, the CAN module clears the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 0. If the CAN module requested a transmit abort, the transmit abort is accepted and writing to the message slot is enabled.

(5) Completion of remote frame transmission

When remote frame transmission is completed, a time stamp count value at the time transmission was completed is written to the CAN Message Slot Time Stamp (C0MSLnTSP) and the CAN Message Slot Control Register's RA (Remote Active) bit is cleared to 0. Also, the CAN Slot Interrupt Status bit is set to 1 by completion of transmission, but the CAN Message Slot Control Register's TRFIN (Transmit/Receive Finished) bit is not set to 1. If the CAN slot interrupt has been enabled, an interrupt request is generated upon completion of transmission.

(6) Receiving a data frame

When remote frame transmission is completed, the slot automatically starts functioning as a data frame receive slot.

(7) Acceptance filtering

When the CAN module finished receiving data, it starts searching for the slot that satisfies conditions for receiving the received message sequentially from slot 0 (up to slot 15).

The following shows receive conditions for slots that have been set for data frame reception.

**[Conditions]**

- The receive frame is a data frame.
- The receive ID and the slot ID are identical, assuming the ID Mask Register bits set to 0 are "Don't care bit."
- The standard and extended frame types are the same.

Note: • In BasicCAN mode, slots 14 and 15 cannot be used as transmit slots.

(8) When receive conditions are met

When receive conditions in (7) above are met, the CAN module sets the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) and TRFIN (Transmit/Receive Finished) bits to 1 while at the same time writing the received data to the message slot. If the TRFIN (Transmit/Receive Finished) bit is already 1, the CAN module also sets the ML (Message Lost) bit to 1, indicating that the message slot has been overwritten. The message slot has its ID field and DLC field both overwritten and an indeterminate value written in its unused area (e.g., extended ID field for standard frame reception and an unused data field).
Furthermore, a time stamp count value at the time the message was received is written to the CAN Message Slot Time Stamp (C0MSLnTSP) along with the received data. When the CAN module finished writing to the message slot, it sets the CAN Slot Interrupt Status bit to 1. If the interrupt for the slot has been enabled, an interrupt request is generated, and the slot goes to a wait state for the next reception.

Note: • If the CAN module received a data frame before transmitting a remote frame, it stores the data frame in the slot and does not transmit the data frame.

(9) When receive conditions are not met

The received frame is discarded, and the CAN module goes to the next transmit/receive operation without writing to the message slot.

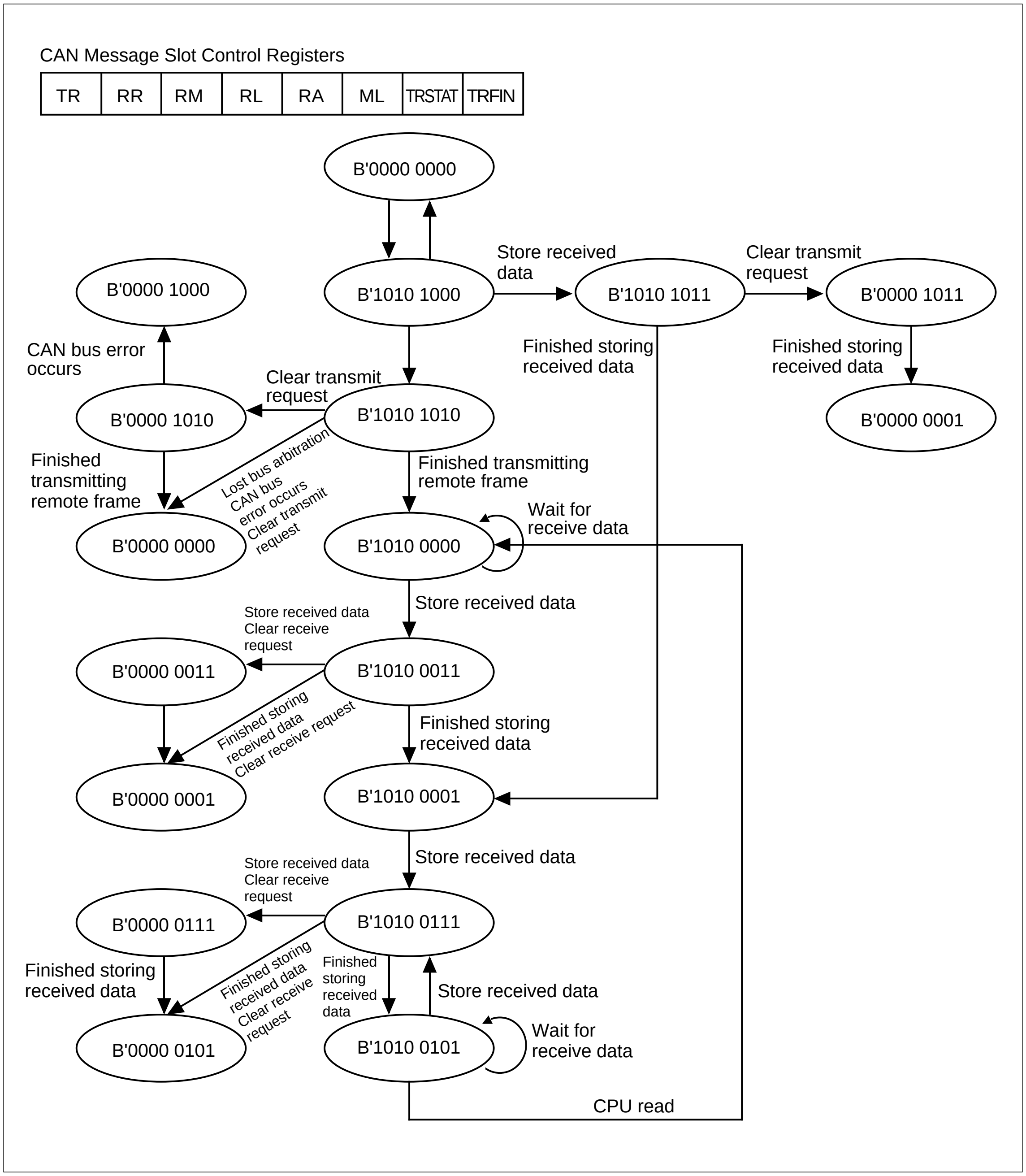


**Figure 13.7.2 Operation of the CAN Message Slot Control Register when Transmitting Remote Frames**

### 13.7.3 Reading Out Received Data Frames when Set for Remote Frame Transmission

The following describes the procedure for reading out received data frames from the slot when it is set for remote frame transmission.

(1) Clearing the TRFIN (Transmit/Receive Finished) bit

Write H'AE or H'00 to the CAN Message Control Register (C0MSLnCNT) to clear the TRFIN bit to 0. After this write, the slot operates as follows:

| Value written to C0MSLnCNT | Slot operation after write |
|---|---|
| H'AE | Operates as a data frame receive slot.<br>Overwrite can be verified by ML bit. |
| H'00 | The slot stops transmit/receive operation. |

Notes: • If message-lost check by the ML bit is needed, write H'AE to the C0MSLnCNT register as you clear the TRFIN bit.

- If you clear the TRFIN bit by writing H'AE or H'00, it is possible that new data will be stored in the slot while still reading a message from the slot.
- The received data frame cannot be read out by writing H'A0 to the register. If you clear the TRFIN bit by writing H'A0, the slot performs remote frame transmit operation.

(2) Reading out from the message slot

Read out a message from the message slot.

(3) Checking the TRFIN (Transmit/Receive Finished) bit

Read the CAN Message Control Register to check the TRFIN (Transmit/Receive Finished) bit.

1) When TRFIN (Transmit/Receive Finished) bit = 1

It means that new data was stored in the slot while still reading out from the slot in (2). In this case, the data read out in (2) may contain an indeterminate value. Therefore, reexecute beginning with clearing of the TRFIN (Transmit/Receive Finished) bit in (1).

2) When TRFIN (Transmit/Receive Finished) bit = 0

It means that the CAN module finished reading out from the slot normally.

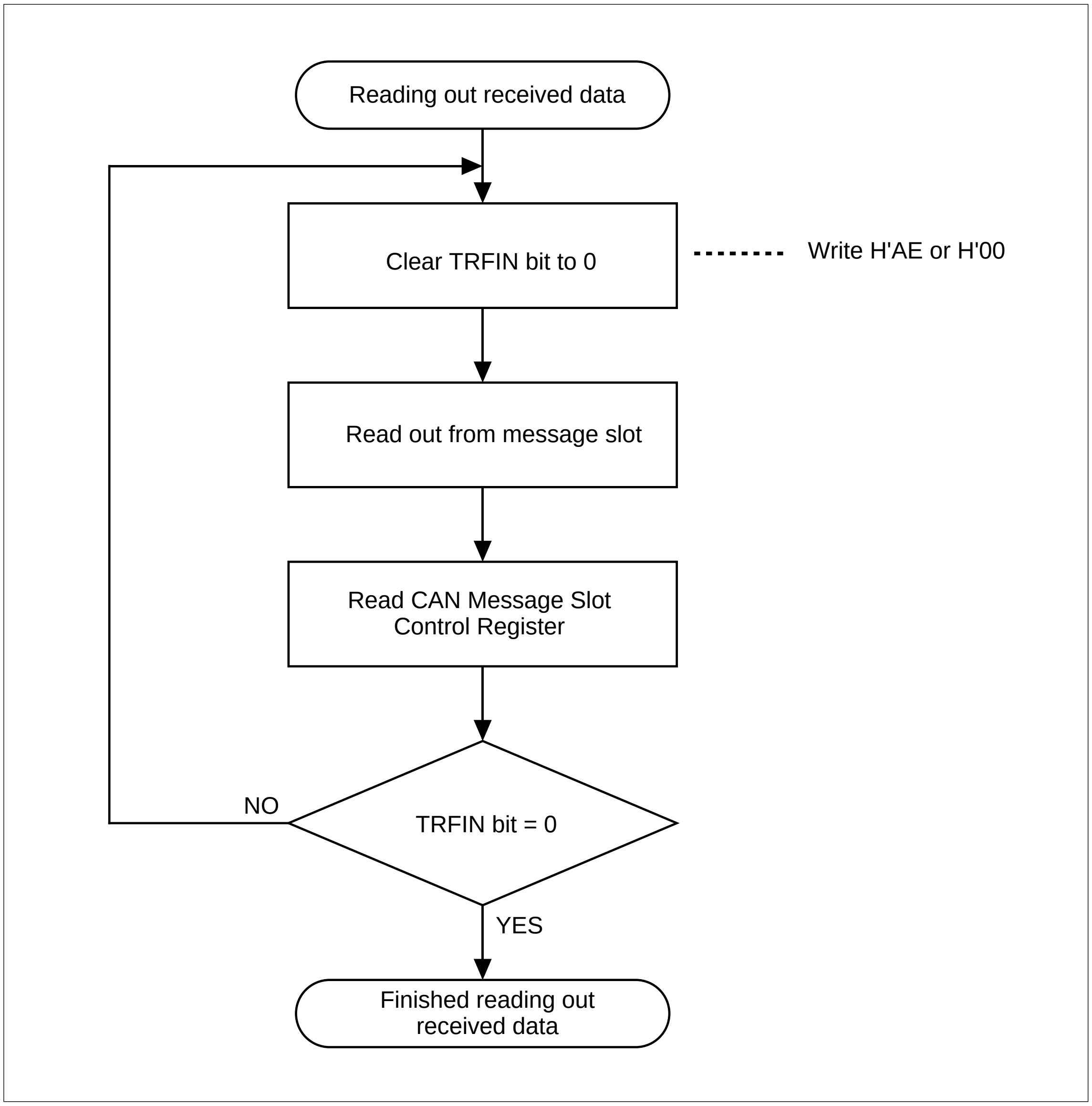

Reading out received data
Clear TRFIN bit to 0
Write H'AE or H'00
Read out from message slot
Read CAN Message Slot Control Register
TRFIN bit = 0
NO
YES
Finished reading out received data

Figure 13.7.3 Procedure for Reading Out Received Data when Set for Remote Frame Transmission

## 13.8 Receiving Remote Frames

### 13.8.1 Remote Frame Receive Procedure

The following describes the procedure for receiving remote frames.

(1) Initializing the CAN Message Slot Control Register

Initialize the CAN Message Slot Control Register for the slot in which you want to receive by writing H'00 to the register.

(2) Confirming that reception is idle

Read the CAN Message Slot Control Register after being initialized and check the TRSTAT (Transmit/Receive Status) bit to see that reception has stopped and remains idle. If this bit = 1, it means that the CAN module is accessing the message slot, so you need to wait until the bit is cleared.

(3) Setting the receive ID

Set the ID you want to receive in the message slot.

(4) Setting the Extended ID Register

Set the corresponding bit of the Extended ID Register to 0 when you want to receive a standard frame or 1 when you want to receive an extended frame.

(5) Setting the CAN Message Slot Control Register

1) When automatic response (data frame transmission) for remote frame reception is desired
Write H'60 to the CAN Message Slot Control Register to set the RR (Receive Request) and RM (Remote) bits to 1.

2) When automatic response (data frame transmission) for remote frame reception is not needed
Write H'70 to the CAN Message Slot Control Register to set the RR (Receive Request), RM (Remote), and RL (Automatic Response Inhibit) bits to 1.

Note: • In BasicCAN mode, slots 14 and 15, although capable of receiving remote frames, cannot automatically respond to remote frame reception.

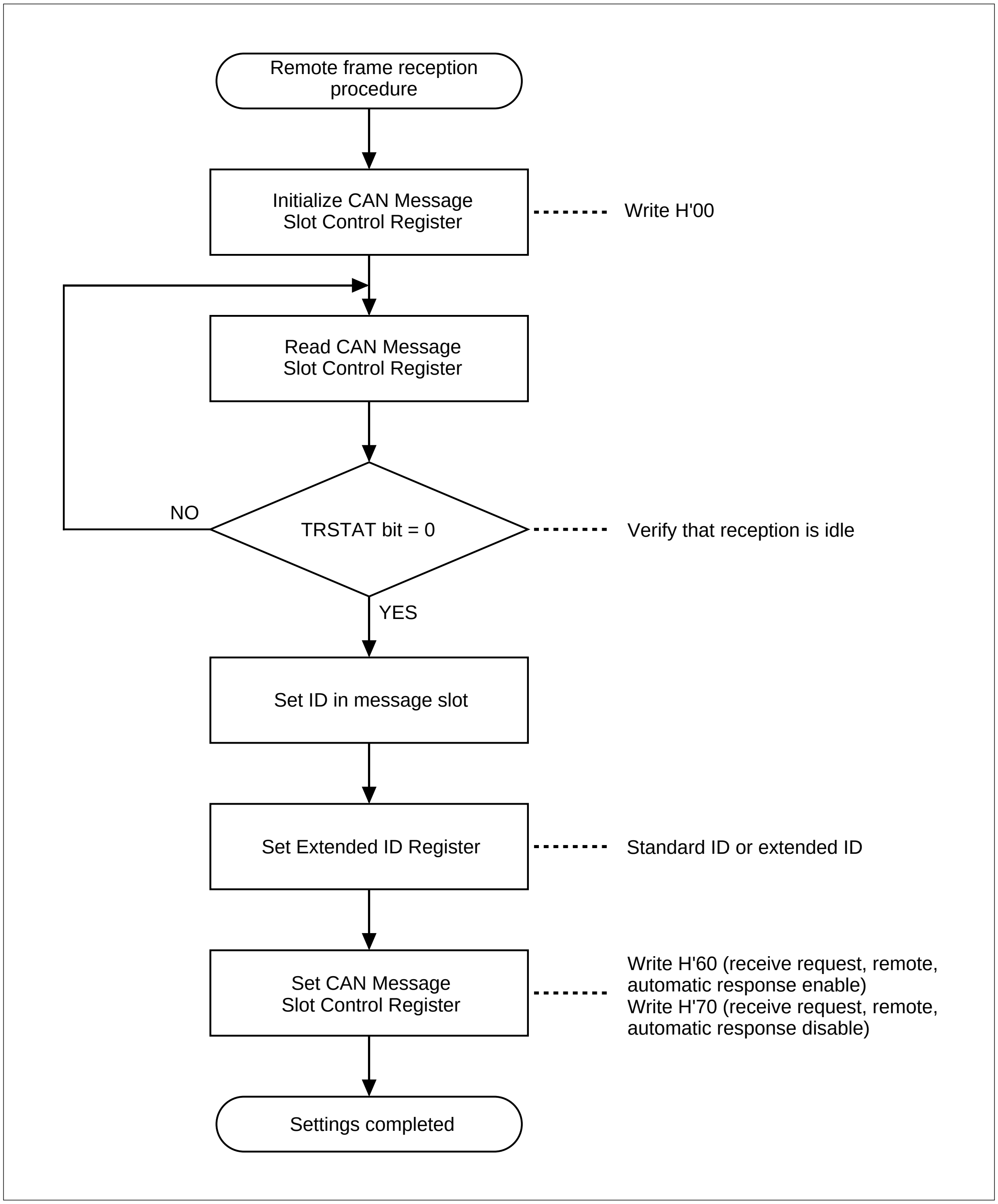

Remote frame reception procedure
Initialize CAN Message Slot Control Register
Write H'00
Read CAN Message Slot Control Register
NO
TRSTAT bit = 0
Verify that reception is idle
YES
Set ID in message slot
Set Extended ID Register
Standard ID or extended ID
Set CAN Message Slot Control Register
Write H'60 (receive request, remote, automatic response enable)
Write H'70 (receive request, remote, automatic response disable)
Settings completed

**Figure 13.8.1 Remote Frame Receive Procedure**

### 13.8.2 Remote Frame Receive Operation

The following describes remote frame receive operation. The operations described below are automatically performed in hardware.

(1) Setting the RA (Remote Active) bit

When H'60 (Receive Request, Remote) or H'70 (Receive Request, Remote, Automatic Response Disable) is written to the CAN Message Slot Control Register, the RA (Remote Active) bit is set to 1, indicating that the corresponding slot is to handle remote frames.

(2) Acceptance filtering

When the CAN module finished receiving data, it starts searching for the slot that satisfies conditions for receiving the received message sequentially from slot 0 (up to slot 15). The following shows receive conditions for slots that have been set for data frame reception.

**[Conditions]**

- The receive frame is a remote frame.
- The receive ID and the slot ID are identical, assuming the ID Mask Register bits set to 0 are "Don't care bit."
- The standard and extended frame types are the same.

(3) When receive conditions are met

When receive conditions in (2) above are met, the CAN module sets the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) and TRFIN (Transmit/Receive Finished) bits to 1 while at the same time writing the received data to the message slot. Furthermore, a time stamp count value at the time the message was received is written to the CAN Message Slot Time Stamp (C0MSLnTSP) along with the received data. When the CAN module finished writing to the message slot, it sets the CAN Slot Interrupt Status bit to 1. If the interrupt for the slot has been enabled, an interrupt request is generated.

Notes:
- The ID field and DLC value are written to the message slot.
- When receiving standard format frames, an indeterminate value is written to the extended ID area.
- The data field is not accessed for write.
- The RA and TRFIN bits are cleared to 0 after writing the remote frame received data.

(4) When receive conditions are not met

The received frame is discarded, and the CAN module waits for the next receive frame. No data is written to the message slot.

(5) Operation after receiving a remote frame

The operation performed after receiving a remote frame differs depending on how automatic response is set.

1) When automatic response is disabled

The slot which finished receiving goes to an inactive state and remains inactive (neither transmit nor receive) until it is newly set in software.

2) When automatic response is enabled

After receiving a remote frame, the slot automatically changes to a data frame transmit slot and performs the transmit operation described below. In this case, the transmitted data conforms to the ID and DLC of the received remote frame.

- Selecting a transmit frame

  The CAN module checks slots which have transmit requests (including remote frame transmit slots) every intermission to determine the frame to transmit. If there are multiple transmit slots, frames are transmitted in order of slot numbers beginning with the smallest.

- Transmitting a data frame

  After determining the transmit slot, the CAN module sets the corresponding CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 1, thereby starting transmission.

- If the CAN module lost bus arbitation or a CAN bus error occurs

  If the CAN module lost bus arbitation or a CAN bus error occurs while transmitting, the CAN module clears the CAN Message Slot Control Register's TRSTAT (Transmit/Receive Status) bit to 0. If the CAN module requested a transmit abort, the transmit abort is accepted and writing to the message slot is enabled.

- Completion of data frame transmission

  When data frame transmission is completed, the CAN Message Slot Control Register's TRFIN (Transmit/Receive Finished) bit and the CAN Slot Interrupt Status Register are set to 1. Also, a time stamp count value at the time transmission was completed is written to the CAN Message Slot Time Stamp (C0MSLnTSP), and the transmit operation is thereby completed.

  If the CAN slot interrupt has been enabled, an interrupt request is generated at completion of transmit operation. The slot which has had transmission completed goes to an inactive state and remains inactive (neither transmit nor receive) until it is newly set in software.

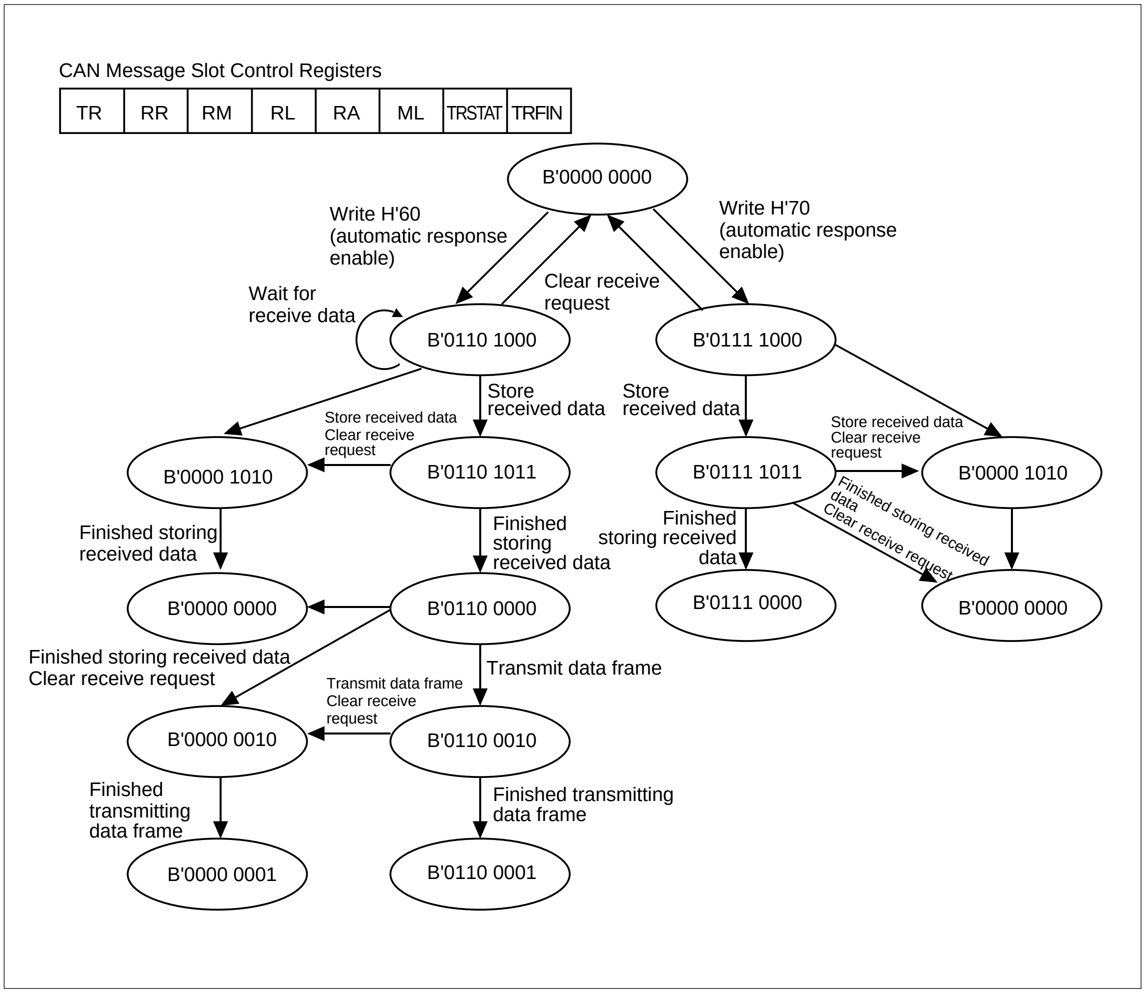


**Figure 13.8.2 Operation of the CAN Message Slot Control Register when Receiving Remote Frames**

# CHAPTER 14
# REAL-TIME DEBUGGER (RTD)

14.1 Outline of the Real-Time Debugger (RTD)
14.2 Pin Function of the RTD
14.3 Functional Description of the RTD
14.4 Typical Connection with the Host

## 14.1 Outline of the Real-Time Debugger (RTD)

The Real-Time Debugger (RTD) is a serial I/O through which to read or write to the internal RAM's entire area using commands from outside the microprocessor. Because data transfers between the RTD and internal RAM are performed using an internal dedicated bus independently of the M32R CPU, operation can be controlled without having the stop the M32R CPU.

**Table 14.1.1 Outline of the Real-Time Debugger (RTD)**

| Item | Content |
|---|---|
| Transfer method | Clock-synchronized serial I/O |
| Generation of transfer clock | Generated by external host |
| RAM access area | Entire area of internal RAM (controlled by A16-A29) |
| Transmit/receive data length | 32 bits (fixed) |
| Bit transfer sequence | LSB first |
| Maximum transfer rate | 2 Mbits/second |
| Input/output pins | 4 lines (RTDTXD, RTDRXD, RTDACK, RTDCLK) |
| Number of commands | Following five functions<br>• Monitors continuously<br>• Outputs real-time RAM contents<br>• Forcibly rewrites RAM contents (with verify)<br>• Recovers from runaway<br>• Requests RTD interrupt |

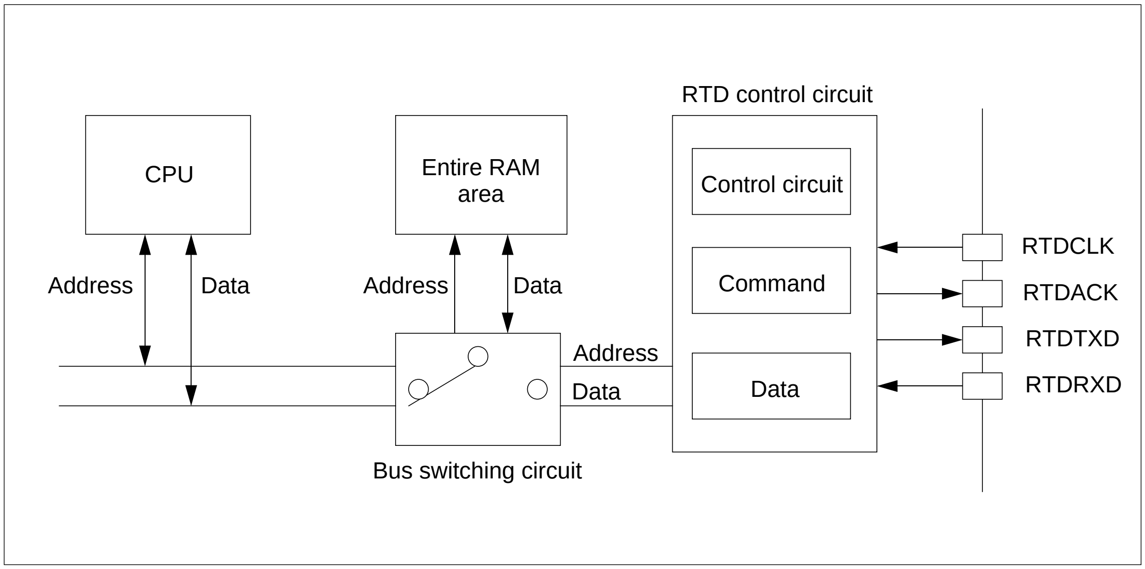


**Figure 14.1.1 Block Diagram of the Real-Time Debugger (RTD)**

## 14.2 Pin Function of the RTD

Pin functions of the RTD are shown below.

**Table 14.2.1 Pin Function of the RTD**

| Pin Name | Type | Function |
|---|---|---|
| RTDTXD | Output | RTD serial data output |
| RTDRXD | Input | RTD serial data input |
| RTDACK | Output | Outputs a low-level pulse synchronously with the beginning clock edge of the output data word. The width of the low-level pulse thus output indicates the type of instruction/data that the RTD received.<br>1 clock period : VER (continuous monitor) command<br>1 clock period : VEI (RTD interrupt request) command<br>2 clock periods : RDR (real-time RAM content output) command<br>3 clock periods : WRR (RAM content forcible rewrite) command or the data to rewrite<br>4 clock periods or more : RCV (recover from runaway) command |
| RTDCLK | Input | RTD transfer clock input |

# 14.3 Functional Description of the RTD

## 14.3.1 Outline of RTD Operation

Operation of the RTD is specified by a command entered from devices external to the chip. A command is specified in bits 16-19 (Note 1) of the RTD receive data.

**Table 14.3.1 RTD Commands**

| RTD Receive Data | | | | Command Mnemonic | RTD Function |
|---|---|---|---|---|---|
| **b19** | **b18** | **b17** | **b16** | | |
| 0 | 0 | 0 | 0 | VER (VERify) | Continuous monitor |
| 0 | 1 | 0 | 0 | | |
| 0 | 1 | 0 | 1 | | |
| 0 | 1 | 1 | 0 | VEI (VErify Interrupt request) | RTD interrupt request |
| 0 | 0 | 1 | 0 | RDR (ReaD RAM) | Real-time RAM content output |
| 0 | 0 | 1 | 1 | WRR (WRite RAM) | RAM content forcibly rewrite (with verify) |
| 1 | 1 | 1 | 1 | RCV (ReCoVer) | Recover from runaway (Note 2, Note 3) |
| 0 | 0 | 0 | 1 | System reserved (use inhibited) | |

↑ (Note 1)

Note 1 : Bit 19 of RTD receive data is not actually stored in the command register and except for the RCV command, is handled as "Don't Care" bit. (Bits 16-18 are effective for the command specified.)

Note 2 : The RCV command must always be transmitted twice in succession.

Note 3 : For the RCV command, all bits, not just bits 16-19, (i.e., bits 0-15 and bits 20-31) must be set to 1.

### 14.3.2 Operation of RDR (Real-time RAM Content Output)

When the RDR (real-time RAM content output) command is issued, the RTD is made possible to transfer the contents of the internal RAM to external devices without causing the CPU's internal bus to stop. Because the RTD reads data from the internal RAM while no transfers are being performed between the CPU and internal RAM, the CPUinno extra load.

The address to be read from the internal RAM can only be specified on 32-bit word boundaries. (The two low-order address bits specified by a command are ignored.) Note also that data are read out in units of 32 bits as transferred from the internal RAM to an external device.

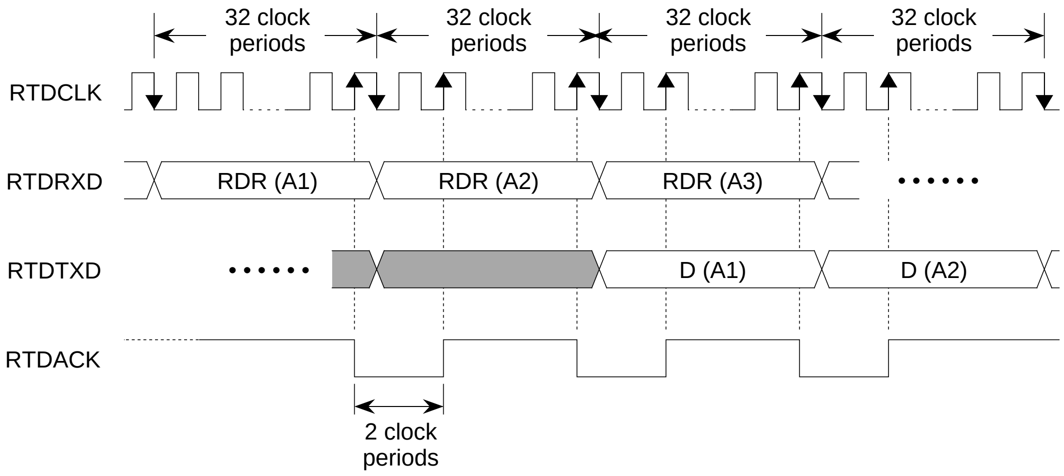
(LSB side) (MSB side)

| | 31 | •••••• | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | •••••• | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RTDRXD ← | X | •••••• | X | 0 | 0 | 1 | 0 | X | X | A29 | A28 | •••••• | A17 | A16 |

Command (RDR) (bits 19–16) Specified address (bits 13–0)

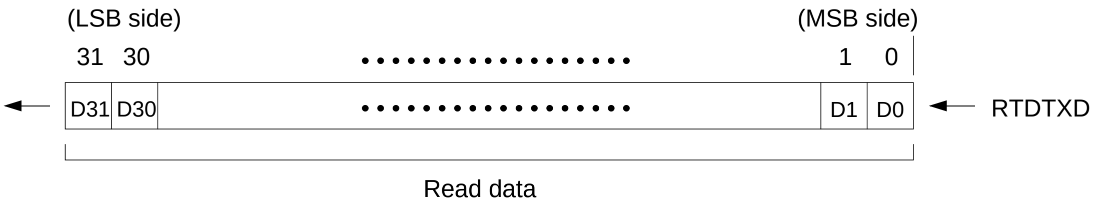
Note: • X = Don't Care (However, if issued immediately after the RCV command, bits 20-31 must all be set to 1.)

**Figure 14.3.1 RDR Command Data Format**



Note: • (An) = Specified address
D (An) = Data at specified address (An)

**Figure 14.3.2 Operation of the RDR Command**

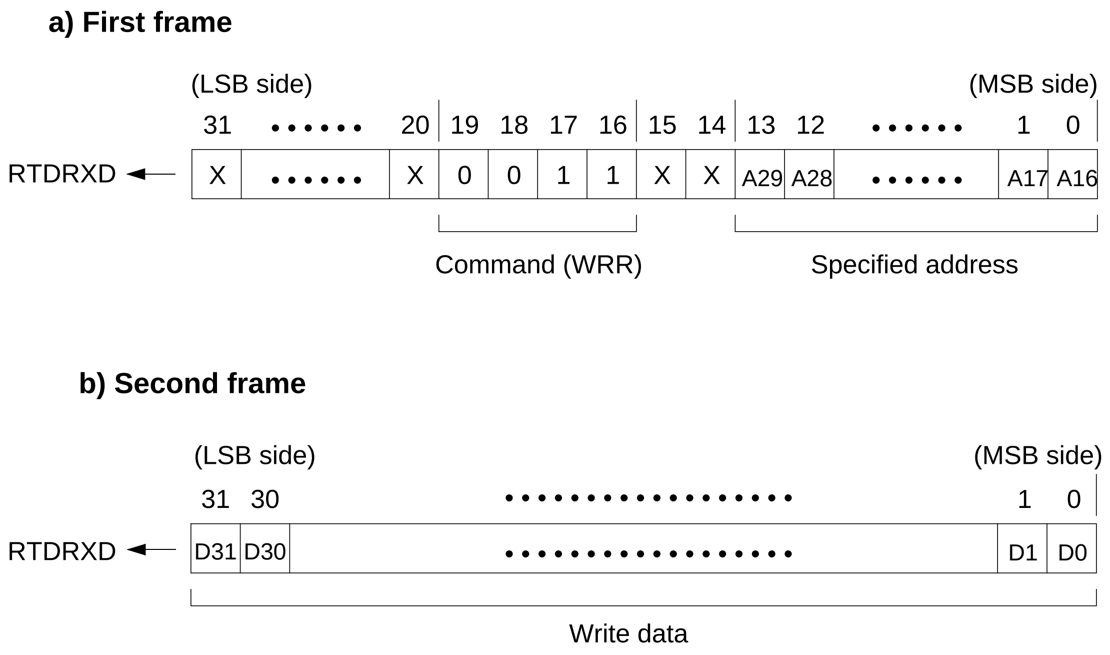


Note: • The read data is transferred LSB-first.

**Figure 14.3.3 Read Data Transfer Format**

### 14.3.3 Operation of WRR (RAM Content Forcible Rewrite)

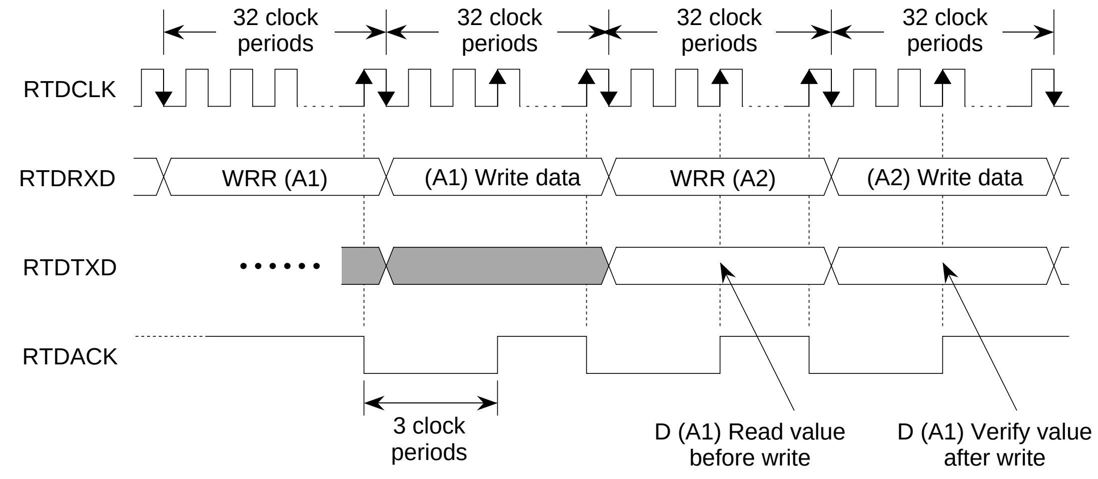
When the WRR (RAM content forcible rewrite) command is issued, the RTD forcibly rewrites the contents of the internal RAM without causing the CPU's internal bus to stop. Because the RTD writes data to the internal RAM while no transfers are being performed between the CPU and internal RAM, the CPU incurs no extra load.
The address to be read from the internal RAM can only be specified on 32-bit word boundaries. (The two low-order address bits specified by a command are ignored.) Note also that data are written to the internal RAM in units of 32 bits.
The external host should transmit the command and address in the first frame and then the write data in the second frame. The timing at which the RTD writes to the internal RAM occurs in the third frame after receiving the write data.



Notes: • X = Don't Care (However, if issued immediately after the RCV command, bits 20-31 must all be set to 1.)
• The specified address and write data are transferred LSB-first.

**Figure 14.3.4 WRR Command Data Format**

The RTD reads out data from the specified address before writing to the internal RAM and again reads out from the same address immediately after writing to the internal RAM (this helps to verify the data written to the internal RAM). The read data is output at the timing shown below.

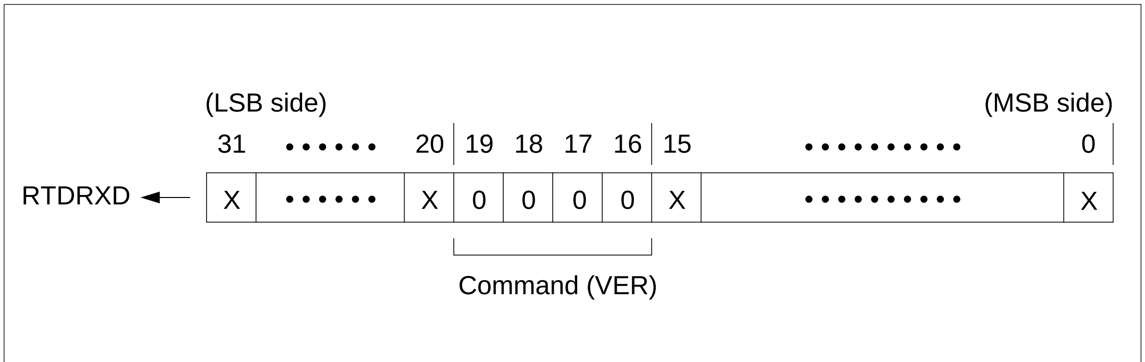


Note: • (An) = Specified address
D (An) = Data at specified address (An)

**Figure 14.3.5 Operation of the WRR Command**

### 14.3.4 Operation of VER (Continuous Monitor)

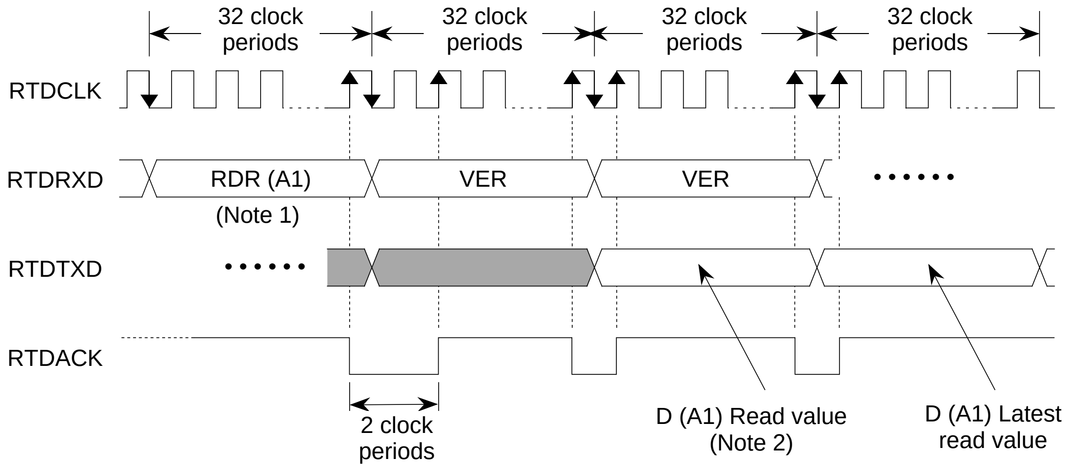
When the VER (continuous monitor) command is issued, the RTD outputs data from the address that has been accessed by the instruction (either read or write) immediately before receiving the VER command.



| 31 | •••••• | 20 | 19 | 18 | 17 | 16 | 15 | •••••••••• | 0 |
|---|---|---|---|---|---|---|---|---|---|
| X | •••••• | X | 0 | 0 | 0 | 0 | X | •••••••••• | X |

Note: • X = Don't Care (However, if issued immediately after the RCV command, bits 20-31 must all be set to 1.)

**Figure 14.3.6 VER (Continuous Monitor) Command Data Format**

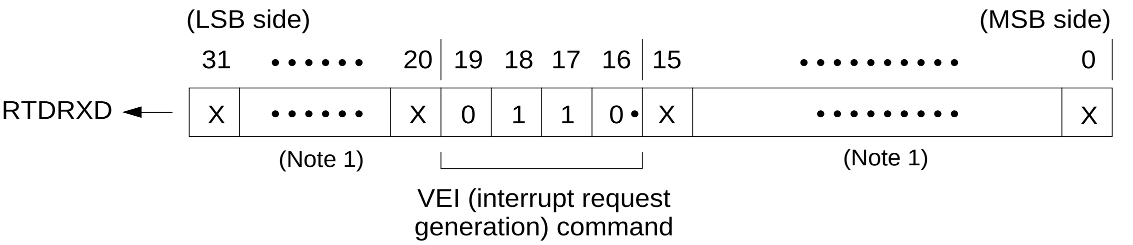


Note 1 : WRR command can also be used.
Note 2 : (An) = Specified address
D (An) = Data at specified address (An)

**Figure 14.3.7 Operation of the VER (Continuous Monitor) Command**

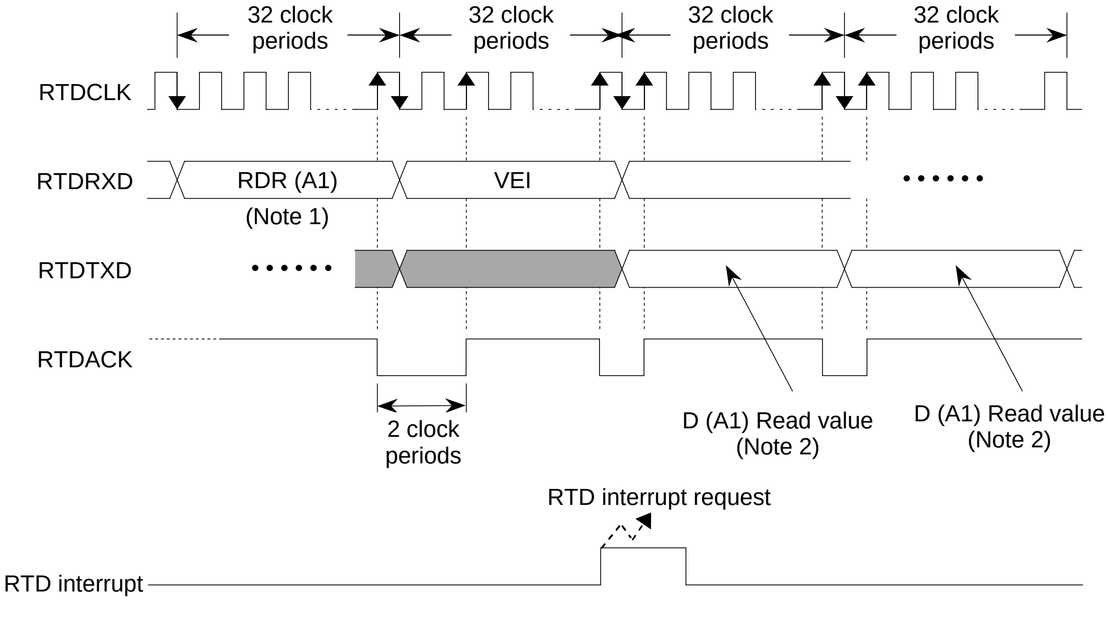
### 14.3.5 Operation of VEI (Interrupt Request)

When the VEI (interrupt request) command is issued, the RTD outputs data from the address that has been accessed by the instruction (either read or write) immediately before receiving the VEI command.



Note 1: X = Don't Care (However, if issued immediately after the RCV command, bits 20-31 must all be set to 1.)

**Figure 14.3.8 VEI (Interrupt Request) Command Data Format**



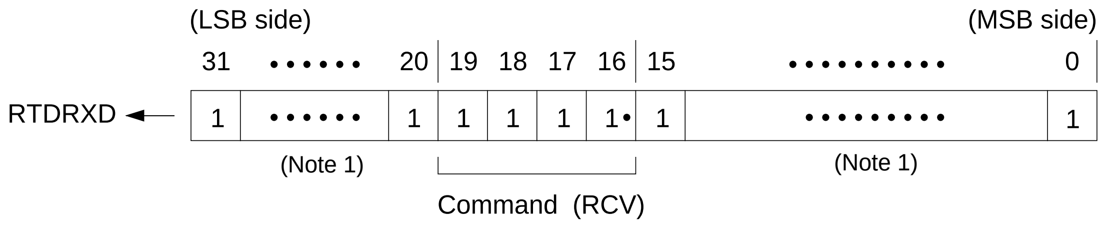
Note 1 : WRR command can also be used.

Note 2 : (An) = Specified address
D (An) = Data at specified address (An)

**Figure 14.3.9 Operation of the VEI (Interrupt Request) Command**

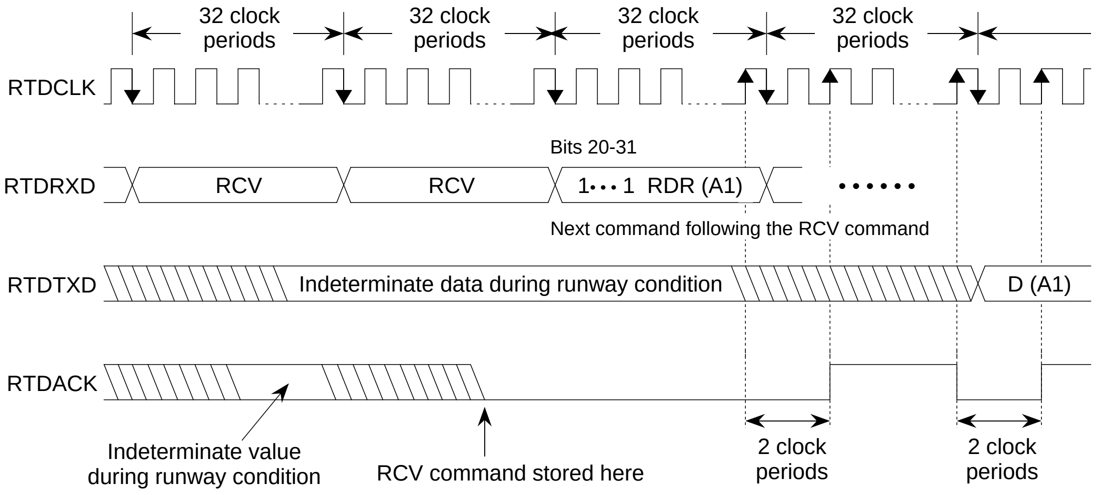
### 14.3.6 Operation of RCV (Recover from Runaway)

When the RTD runs out of control, the RCV (recover from runway) command can be issued to forcibly recover from the runaway condition without having to reset the system. The RCV command must always be issued twice in succession. Also, any command issued subsequently after the RCV command must have its bits 20-31 all set to 1.



Note 1: All of 32 data bits are 1's. The RCV command must always be issued twice in succession.

**Figure 14.3.10 RCV Command Data Format**



Note: • The next command following the RCV command must have its bits 20-31 all set to 1.

**Figure 14.3.11 Operation of the RCV Command**

### 14.3.7 Method to Set a Specified Address when Using the RTD

When using the Real-Time Debugger (RTD), you can set low-order 16-bit addresses of the internal RAM area. Because the internal RAM area is located in a 48 KB area ranging from H'0080 4000 to H'0080 FFFF, you can set low-order 16-bit addresses (H'4000 to H'FFFF) of that area. However, access to any locations other than the area where the RAM resides is inhibited. Note also that two least significant address bits, A31 and A30, are always 0's because data are read and written to the internal RAM in a fixed length of 32 bits.

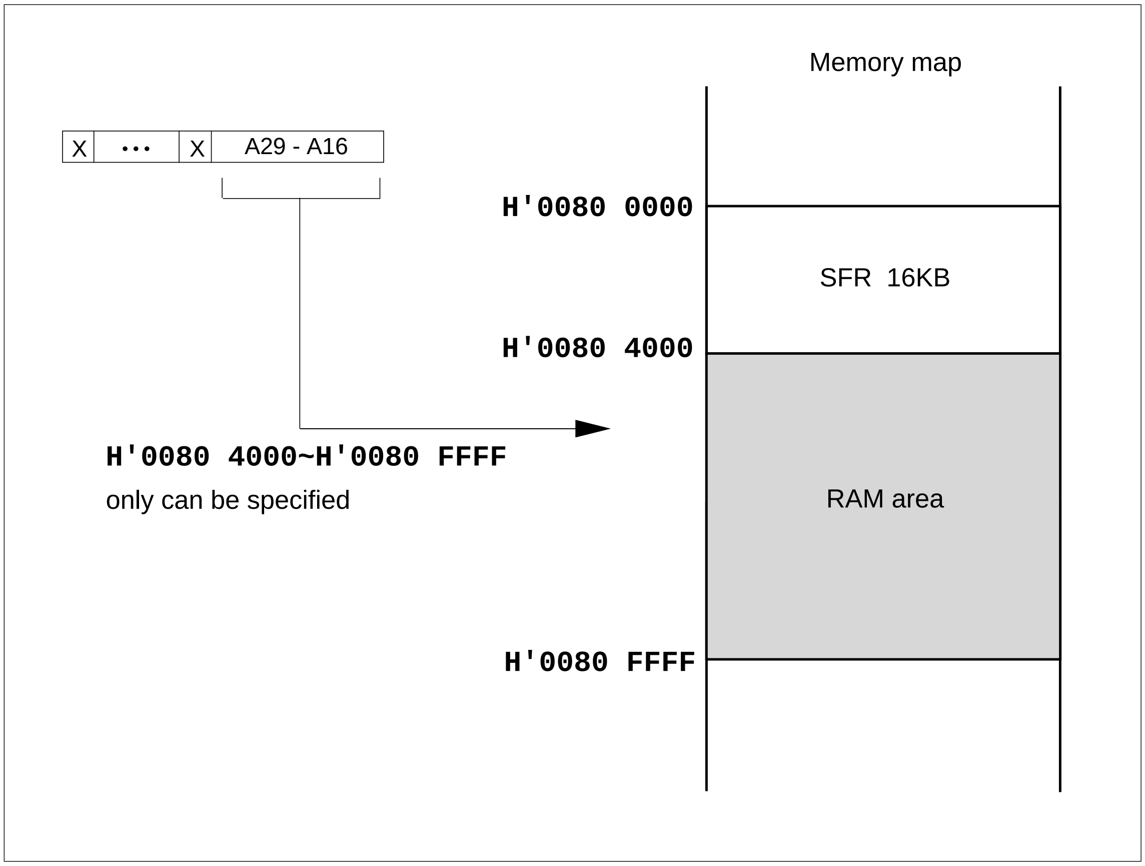


Figure 14.3.12 Method for Setting Addresses in Real-Time Debugger

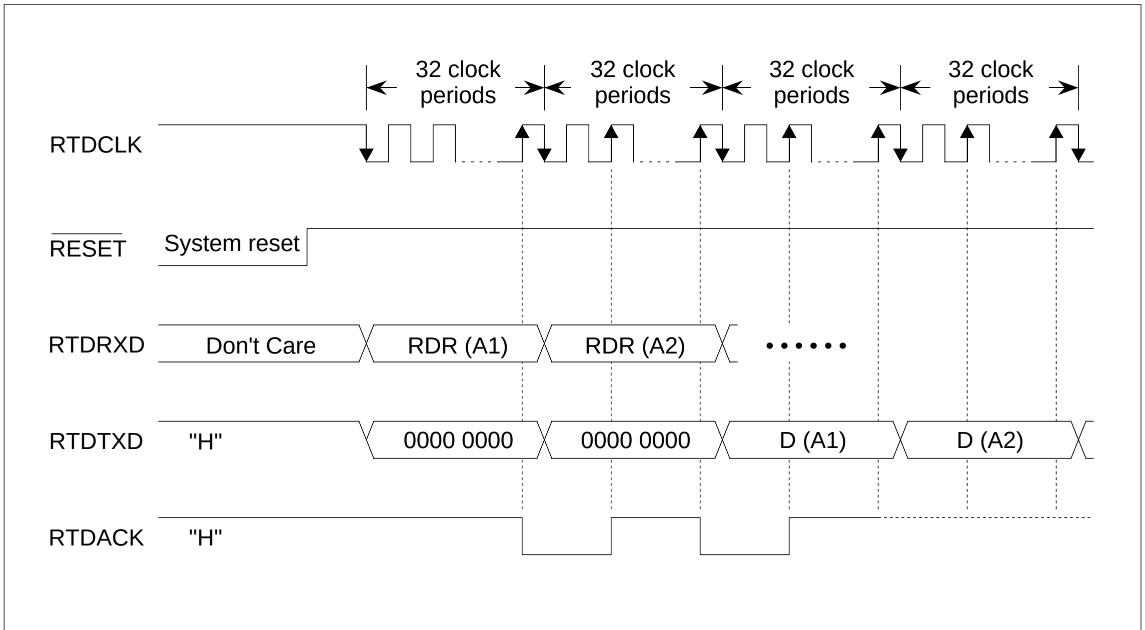
### 14.3.8 Resetting the RTD

The RTD is reset by applying a system rest (i.e., by entering the $\overline{\text{RESET}}$ signal). The status of the RTD related output pins after a system reset are shown below.

**Table 14.3.2 RTD Pin State after Releasing System from Reset**

| Pin Name | State |
|---|---|
| RTDACK | High-level output |
| RTDTXD | High-level output |

The first command transfer to the RTD after it was reset is initiated by transferring data to the RTDRXD pin synchronously with falling edges of RTDCLK.

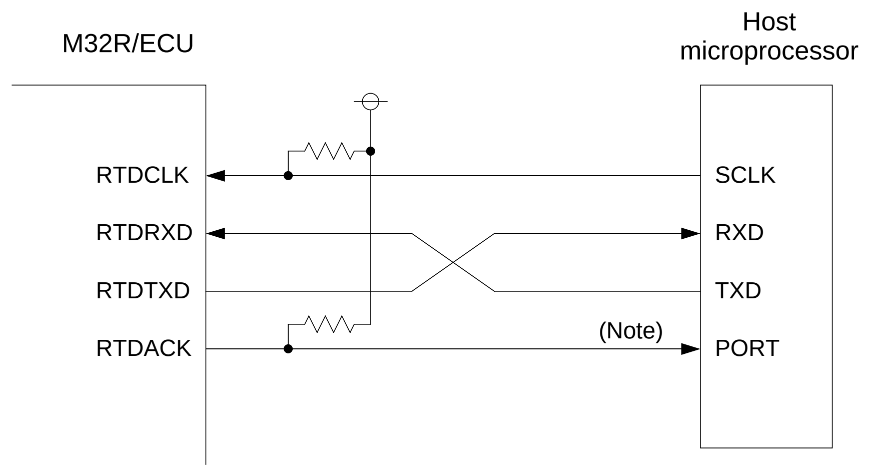


Note : • (An) = Specified address
D (An) = Data at specified address (An)

**Figure 14.3.13 Command Transfer to the RTD after System Reset**

## 14.4 Typical Connection with the Host

The host uses a serial synchronous interface to transfer data. The clock for synchronous is generated by the host. An example for connecting the RTD and host is shown below.



Note: • In this example, the RTDACK level is checked between transfer frames.

**Figure 14.4.1 Connecting the RTD and Host**

The RTD communication for a fixed length of 32 bits per frame generally is performed in four operations sending 8 bits at a time, because most serial interfaces transfer data in units of 8 bits. The RTDACK signal is used to verify that communication is performed normally.

After transmitting a command, the RTDACK signal is pulled low, making it possible to verify the communication status. When issuing the VER command, the RTDACK signal goes low for only one clock period. Therefore, after sending 32 bits in one frame, turn off RTDCLK output and check whether RTDACK is low. If RTDACK is low, you know that the RTD is communicating normally.

If you want to identify the type of transmitted command by the width of RTDACK, use the 32171's internal measurement timer (to count RTDCLK pulses while RTDACK is low) or create a dedicated circuit.

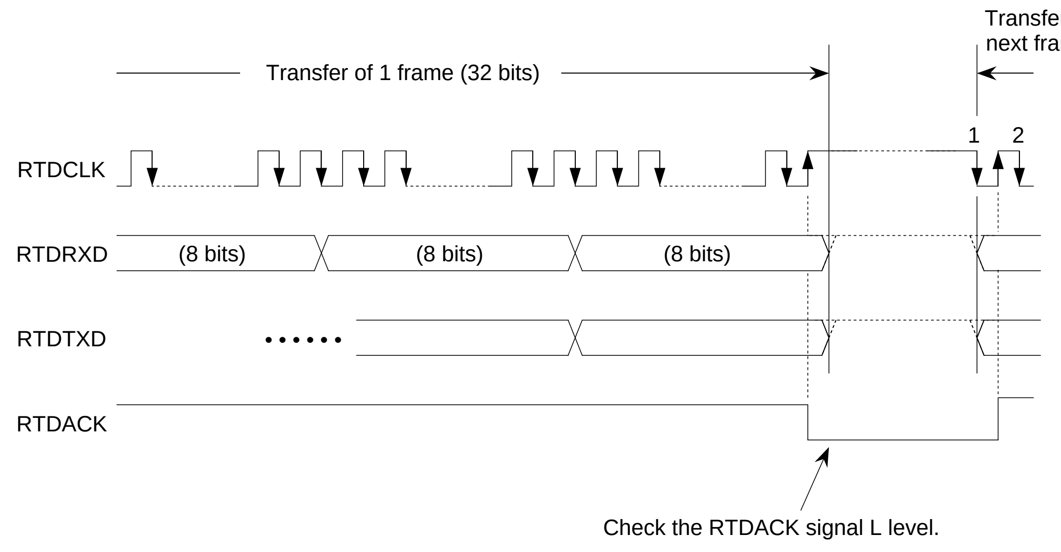


**Figure 14.4.2 Typical Operation for Communication with the Host (when Issuing VER Command)**

* This is a blank page.*

# CHAPTER 15
# EXTERNAL BUS INTERFACE

15.1 External Bus Interface Related Signals
15.2 Read/Write Operations
15.3 Bus Arbitration
15.4 Typical Connection of External Extension Memory

## 15.1 External Bus Interface Related Signals

The 32171 comes with external bus interface related signals shown below. These signals can be used in external extension mode or processor mode.

### (1) Address

The 32171 outputs a 19-bit address (A12-A30) for addressing any location in 1 Mbytes of space. The least significant A31 is not output, and in external write cycles, the 32171 outputs $\overline{\text{BHW}}$ and $\overline{\text{BLW}}$ signals to indicate the valid byte position at which to write on the 16-bit data bus. In read cycles, the 32171 reads data always in 16 bits, transferring only the data read from the valid byte position of the bus.

### (2) Chip select ($\overline{\text{CS0}}$, $\overline{\text{CS1}}$)

These signals are output in external extension mode or processor mode, with $\overline{\text{CS0}}$ and $\overline{\text{CS1}}$ specifying an external extension area of 2 Mbytes each. The $\overline{\text{CS0}}$ signal points to a 2-Mbyte area in processor mode or a 1-Mbyte area in external extension mode. (For details, refer to Chapter 3, "Address Space.")

### (3) Read strobe ($\overline{\text{RD}}$)

Output during external read cycle, this signal indicates the timing at which to read data from the bus. This signal is driven high when writing to the bus or accessing the internal function.

### (4) Byte High Write/Byte High Enable ($\overline{\text{BHW}}$ / $\overline{\text{BHE}}$)

The pin function changes depending on the Bus Mode Control Register (BUSMODC).

When BUSMOD = 0 and this signal is Byte High Write ($\overline{\text{BHW}}$), during external write access it indicates that the upper byte (DB0-DB7) of the data bus is the valid data to transfer. During external read and when accessing the internal function it outputs a high.

When BUSMOD = 1 and this signal is Byte High Enable ($\overline{\text{BHE}}$), during external access it indicates that the upper byte (DB0-DB7) of the data bus is the valid data to transfer. When accessing the internal function, it outputs a high.

### (5) Byte Low Write/Byte Low Enable ($\overline{\text{BLW}}$ / $\overline{\text{BLE}}$)

The pin function changes depending on the Bus Mode Control Register (BUSMODC).

When BUSMOD = 0 and this signal is Byte Low Write ($\overline{\text{BLW}}$), during external write access it indicates that the lower byte (DB8-DB15) of the data bus is the valid data to transfer. During external read cycle, it outputs a high.

When BUSMOD = 1 and this signal is Byte Low Enable ($\overline{\text{BLE}}$), during external access it indicates that the lower byte (DB8-DB15) of the data bus is the valid data to transfer. When accessing the internal function, it outputs a high.

### (6) Data bus (DB0 - DB15)

This is the 16-bit data bus used to access external devices.

### (7) System clock/write (BCLK / $\overline{\text{WR}}$)

The pin function changes depending on the Bus Mode Control Register (BUSMODC).

When BUSMOD = 0 and this signal is System Clock (BCLK), it outputs the system clock necessary to synchronize operations in an external system. When the CPU clock = 40 MHz, a 20 MHz clock is output from BCLK. When not using the BCLK/WR function, this pin can be used as P70 by setting the P7 Operation Mode Register P70MOD bit to 0.

When BUSMOD = 1 and this signal is Write ($\overline{\text{WR}}$), during external write access it indicates the valid data on the data bus to transfer. During external read cycle and when accessing the internal function, it outputs a high.

### (8) Wait ($\overline{\text{WAIT}}$)

When the 32171 started an external bus cycle, it automatically inserts wait cycles while the $\overline{\text{WAIT}}$ signal is asserted. For details, refer to Chapter 16, "Wait Controller." When not using the WAIT function, this pin can be used as P71 by setting the P7 Operation Mode Register P71MOD bit to 0. Note that the 32171 always inserts one or more wait cycles for external access. Therefore, the shortest time in which an external device can be accessed is one wait cycle (2 BCLK periods).

### (9) Hold control ($\overline{\text{HREQ}}$, $\overline{\text{HACK}}$)

The hold state refers to a state in which the 32171 has stopped bus access and bus interface related pins are tristated (high impedance). While the 32171 is in a hold state, any bus master external to the chip can use the system bus to transfer data.
The 32171 is placed in a hold state by pulling the $\overline{\text{HREQ}}$ pin input low. While the 32171 remains in a hold state after accepting the hold request and during a transition to the hold state, the $\overline{\text{HACK}}$ pin outputs a low-level signal. To exit from the hold state and return to normal operating state, release the $\overline{\text{HREQ}}$ signal back high. When not using the HREQ and HACK functions, these pins can be used as P72 and P73 by setting the P7 Operation Mode Register P72MOD and P73MOD bits to 0. The status of each 32171 pin during hold are shown below.

**Table 15.1.1 Pin State during Hold Period**

| Pin Name | Pin State or Operation |
|---|---|
| A12-A30, DB0-DB15, $\overline{\text{CS0}}$, $\overline{\text{CS1}}$, $\overline{\text{RD}}$, $\overline{\text{BHW}}$, $\overline{\text{BLW}}$, $\overline{\text{BHE}}$, $\overline{\text{BLE}}$, $\overline{\text{WR}}$ | High impedance |
| $\overline{\text{HACK}}$ | Outputs a low |
| Other pins (e.g., ports and timer output) | Normal operation |

### (10) Port P7 Operation Mode Register (P7MOD)

The BCLK/$\overline{\text{WR}}$, $\overline{\text{WAIT}}$, $\overline{\text{HREQ}}$, and $\overline{\text{HACK}}$ pins respectively are shared with P70, P71, P72, and P73. The Port P7 Operation Mode Register is used to select the function of port P7. Configuration of this register is shown below.

### ■ P7 Operation Mode Register

<Address: H'0080 0747>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| P70MOD | P71MOD | P72MOD | P73MOD | P74MOD | P75MOD | P76MOD | P77MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | P70MOD<br>(Port P70 operation mode) | 0 : P70<br>1 : BCLK / $\overline{\text{WR}}$ | ○ | ○ |
| 9 | P71MOD<br>(Port P71 operation mode) | 0 : P71<br>1 : $\overline{\text{WAIT}}$ | ○ | ○ |
| 10 | P72MOD<br>(Port P72 operation mode) | 0 : P72<br>1 : $\overline{\text{HREQ}}$ | ○ | ○ |
| 11 | P73MOD<br>(Port P73 operation mode) | 0 : P73<br>1 : $\overline{\text{HACK}}$ | ○ | ○ |
| 12 | P74MOD<br>(Port P74 operation mode) | 0 : P74<br>1 : RTDTXD | ○ | ○ |
| 13 | P75MOD<br>(Port P75 operation mode) | 0 : P75<br>1 : RTDRXD | ○ | ○ |
| 14 | P76MOD<br>(Port P76 operation mode) | 0 : P76<br>1 : RTDACK | ○ | ○ |
| 15 | P77MOD<br>(Port P77 operation mode) | 0 : P77<br>1 : RTDCLK | ○ | ○ |

### (11) Bus Mode Control Register (BUSMODC)

The 32171 contains a function to switch between two external bus modes.

### ■ Bus Mode Control Register (BUSMODC)

<Address: H'0080 077F>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  | BUSMOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 - 15 | No functions assigned |  | 0 | — |
| 15 | BUSMOD<br>(Bus mode control) | 0: WR signal separate mode<br>1: Byte enable separate mode | ○ | ○ |

This register is used to facilitate memory connection in processor mode and external extension mode.

When Bus Mode Control Register (BUSMOD) = 0, the WR signal is output separately for each byte area. Signals $\overline{\text{RD}}$, $\overline{\text{BHW}}$, $\overline{\text{BLW}}$, $\overline{\text{BCLK}}$, and $\overline{\text{WAIT}}$ can be used. For memory connection in boot mode, the Bus Mode Control Register has no effect and the interface operates under conditions where Bus Mode Control Register (BUSMOD) = 0.

When Bus Mode Control Register (BUSMOD) = 1, the byte enable signal is output separately for each byte area. Signals $\overline{\text{RD}}$, $\overline{\text{BHW}}$, $\overline{\text{BLE}}$, $\overline{\text{WR}}$, and $\overline{\text{WAIT}}$ can be used. For WAIT control circuit configuration, because BCLK is not output, external timing control is required.

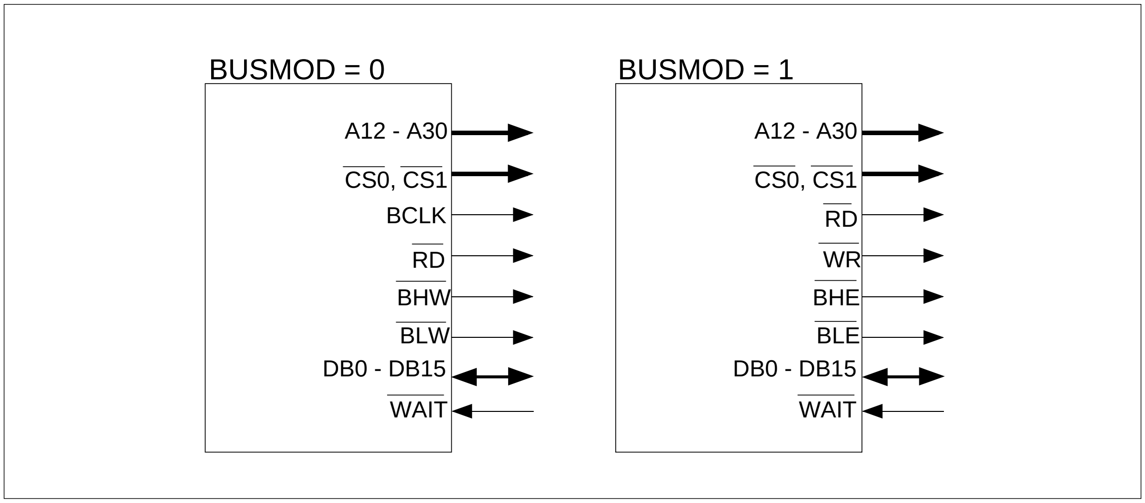


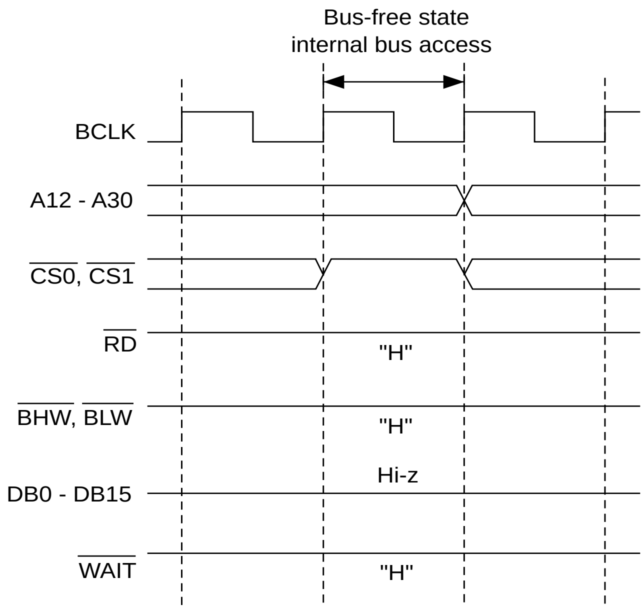
**Figure 15.1.1 Pin Function when Bus Modes are Changed**

## 15.2 Read/Write Operations

### (1) When Bus Mode Control Register = 0

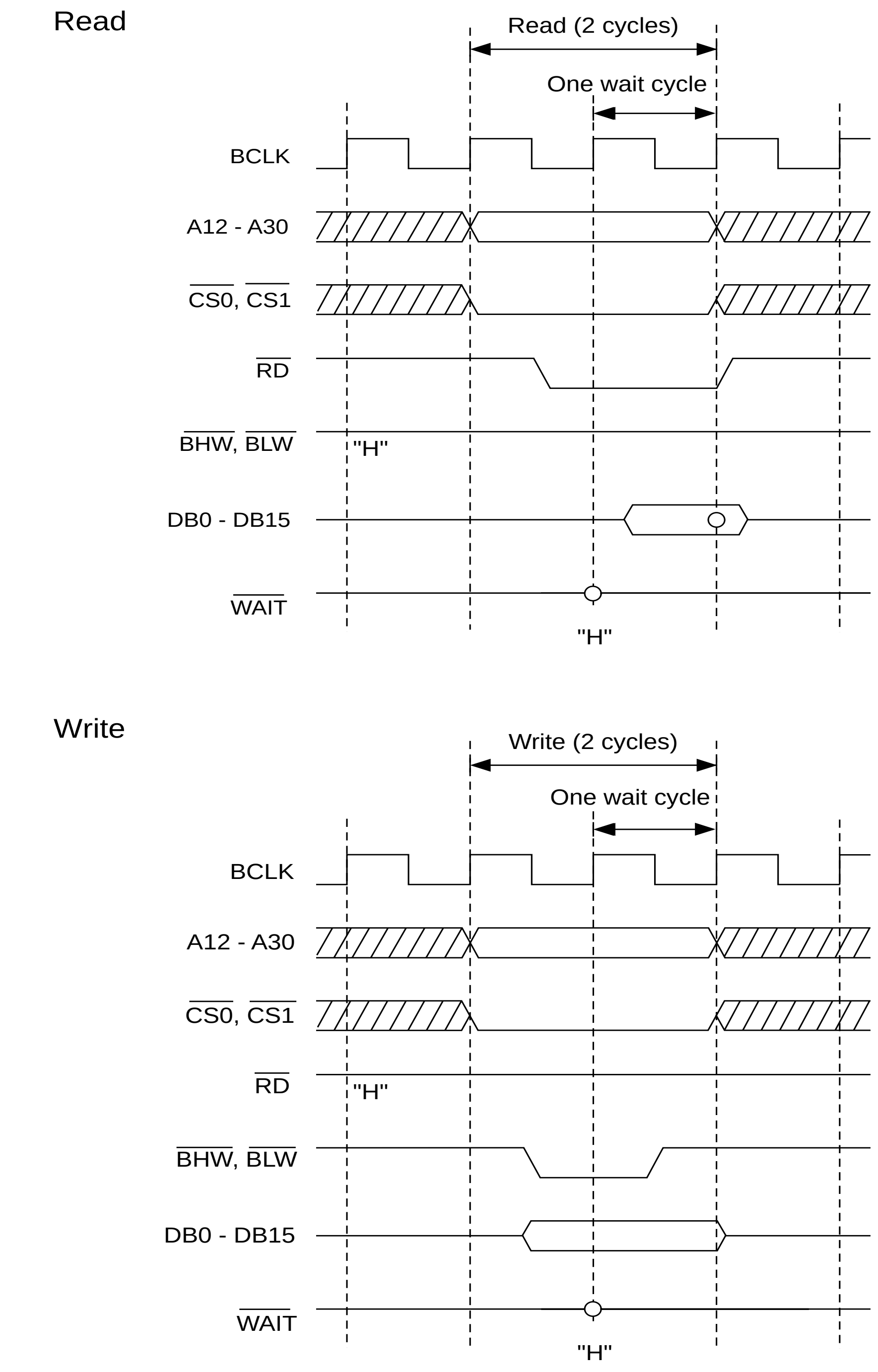
External read/write operations are performed using the address bus, data bus, and signals CS0, CS1, RD, BHW, BLW, WAIT, and BCLK. In external read cycle, the RD signal is low while BHW and BLW both are high, reading data from only the valid byte position of the bus. In external write cycle, BHW or BLW output for the byte position to which to write is pulled low as data is written to the bus.

When an external bus cycle starts, wait cycles are inserted as long as the WAIT signal is low. Unless the WAIT signal is needed, leave it held high. During external bus cycles, at least one wait cycle is inserted even for the shortest-case access. (The shortest bus cycle is 2 BCLK periods.)



Note: • THi-Z denotes a high-impedance state.

**Figure 15.2.1 Internal Bus Access during Bus Free State**



Note: • Circles ○ above indicate points at which signals are sampled.

**Figure 15.2.2 Read/Write Timing (for Shortest-case External Access)**

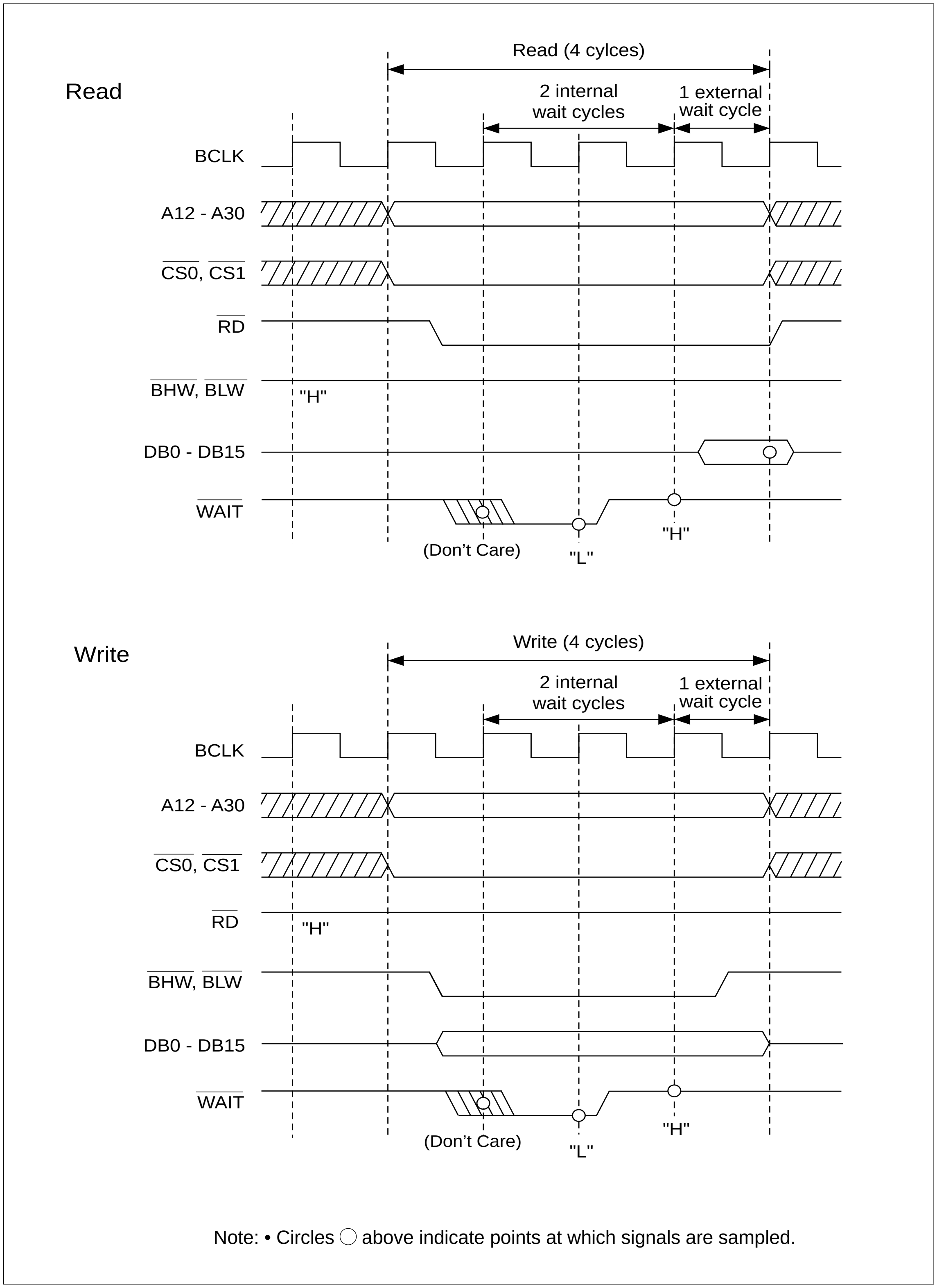
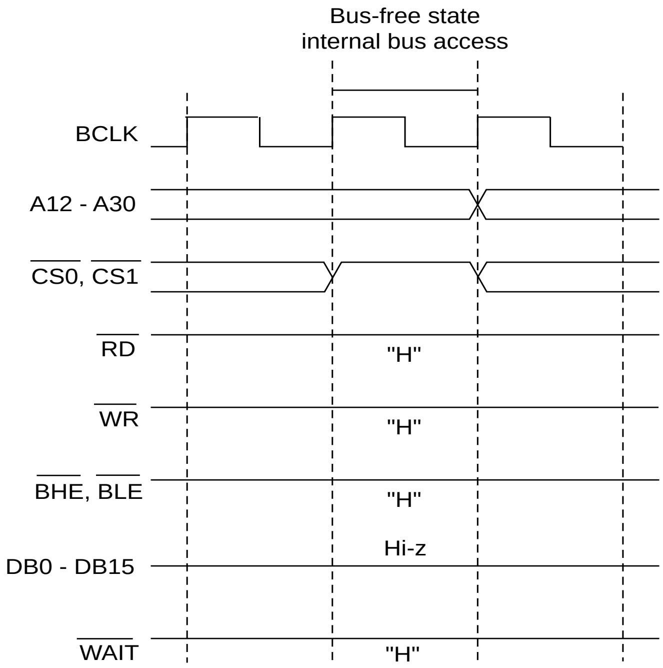


**Figure 15.2.3 Read/Write Timing (for Access with 2 Internal and 1 External Wait Cycles)**

### (2) When Bus Mode Control Register = 1

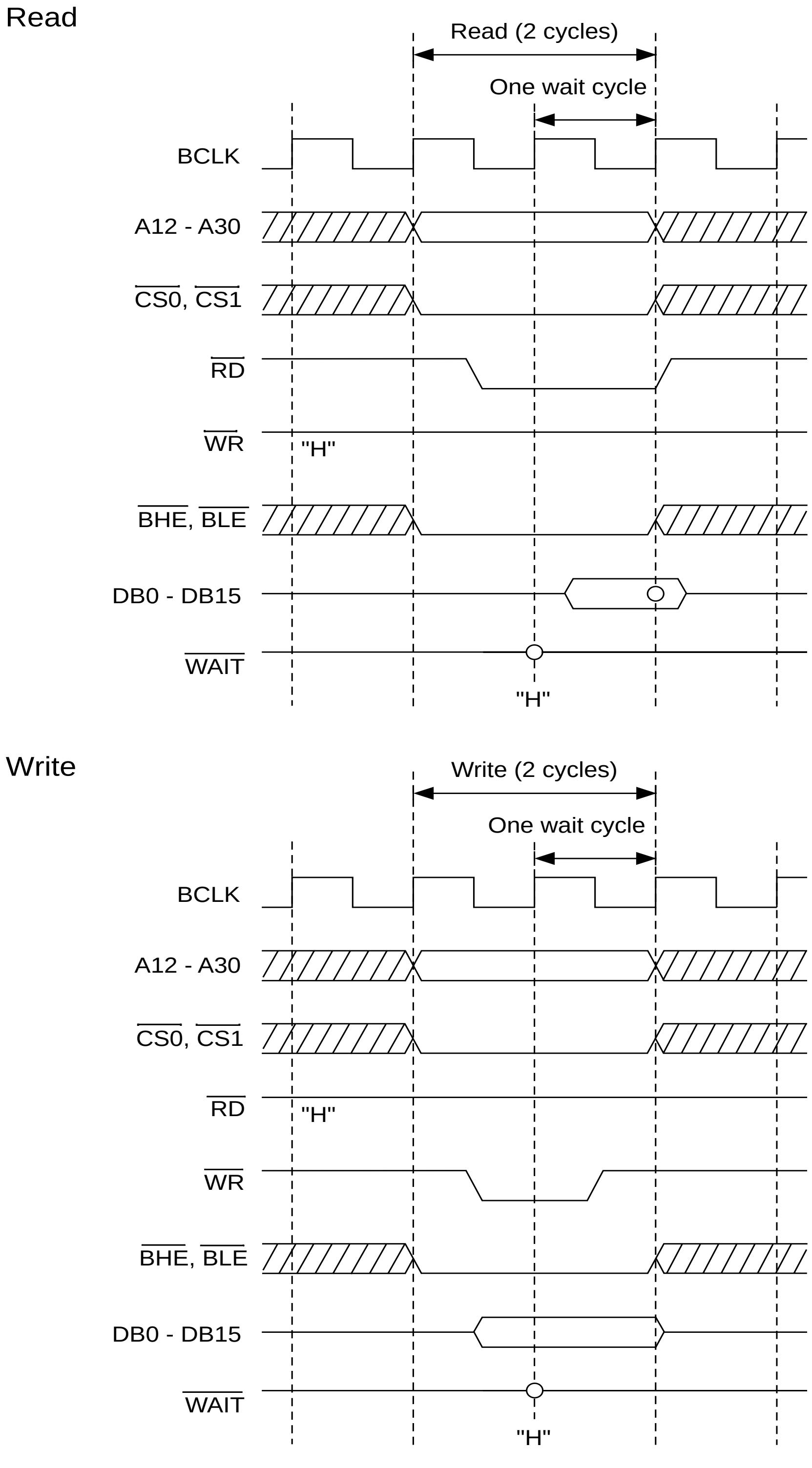
External read/write operations are performed using the address bus, data bus, and signals CS0, CS1, RD, BHE, BLE, WAIT, and WR. In external read cycle, the RD signal goes low and BHE or BLE output for the byte position from which to read is pulled low, reading data from only the byte position of the bus. In external write cycle, the WR signal goes low and BHE or BLE output for the byte position to which to write is pulled low, writing data to the necessary byte position.

When an external bus cycle starts, wait cycles are inserted as long as the WAIT signal is low. Unless the WAIT signal is needed, leave it held high. During external bus cycle, at least one wait cycle is inserted even for the shortest-case access. (The shortest bus cycle is 2 BCLK periods.) When not using the WAIT function, the pin can be used as P71 by setting the P7 Operation Mode Register P71MOD bit to 0.



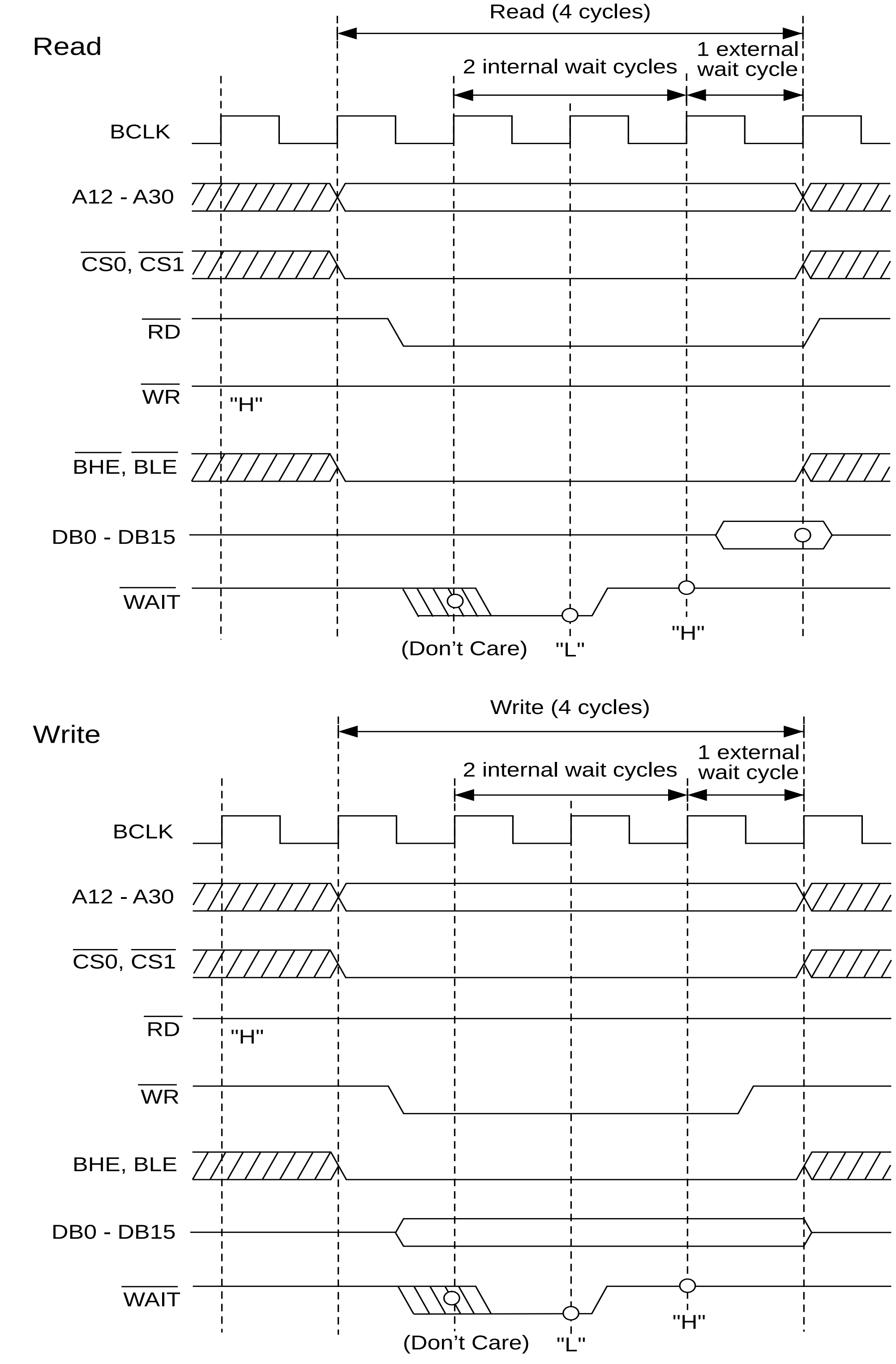
Notes: • Hi-Z denotes a high-impedance state.
• BCLK is not output.

**Figure 15.2.4 Internal Bus Access during Bus Free State**



Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 15.2.5 Read/Write Timing (for Shortest-case External Access)**



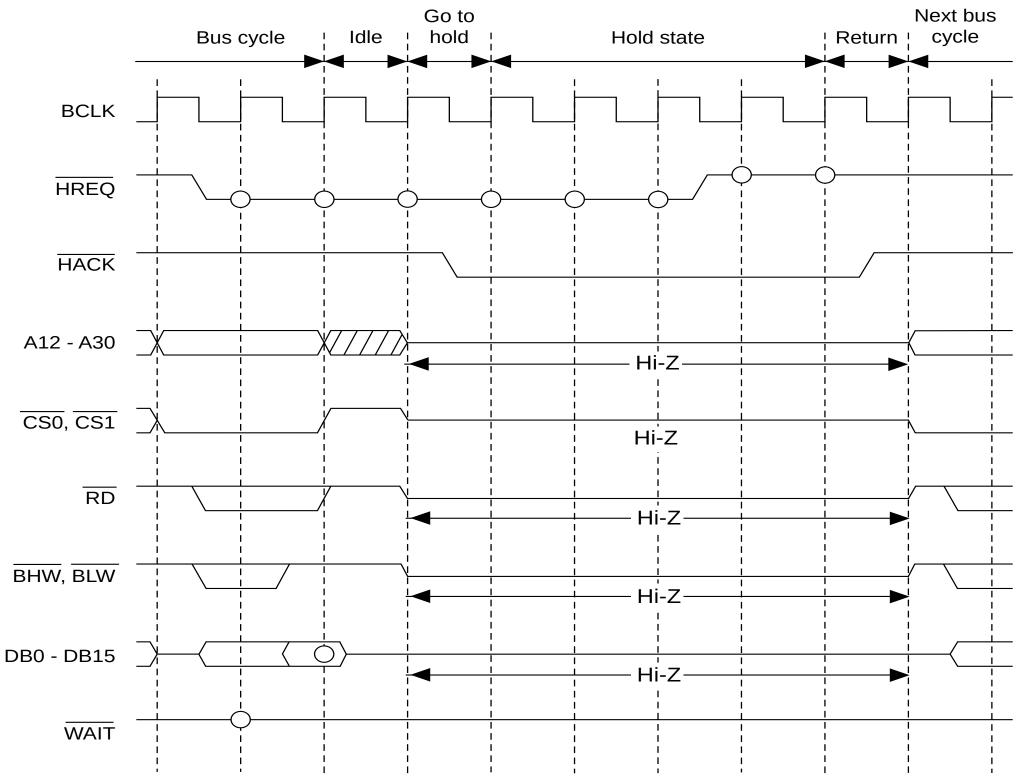
Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 15.2.6 Read/Write Timing (for Access with 2 Internal and 1 External Wait Cycles)**

## 15.3 Bus Arbitration

### (1) When Bus Mode Control Register = 0

When HREQ pin input is pulled low and the hold request is accepted, the 32171 goes to a hold state and outputs a low from the HACK pin. During hold state, all bus related pins are placed in the high-impedance state, allowing data to be transferred on the system bus. To exit the hold state and return to normal operating state, release the HREQ signal back high.

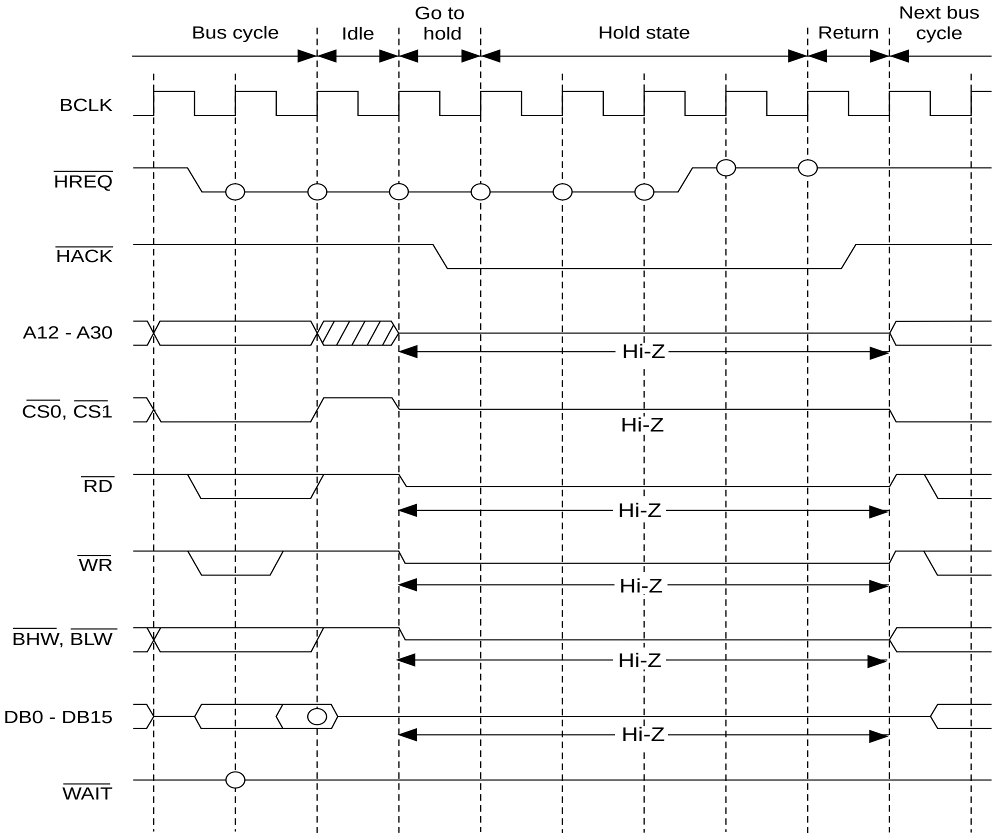


Notes: • Circles ○ above indicate points at which signals are sampled.
• Hi-z indicate the high-impedance state.
• Idle cycles are inserted only when the hold state is assumed after external read access.

**Figure 15.3.1 Bus Arbitration Timing**

### (2) When Bus Mode Control Register = 1

When $\overline{\text{HREQ}}$ pin input is pulled low and the hold request is accepted, the 32171 goes to a hold state and outputs a low from the $\overline{\text{HACK}}$ pin. During hold state, all bus related pins are placed in the high-impedance state, allowing data to be transferred on the system bus. To exit the hold state and return to normal operating state, release the $\overline{\text{HREQ}}$ signal back high.



Notes: • Circles ○ above indicate points at which signals are sampled.
• Hi-z indicate the high-impedance state.
• Idle cycles are inserted only when the hold state is assumed after external read access.

**Figure 15.3.2 Bus Arbitration Timing**

## 15.4 Typical Connection of External Extension Memory

### (1) When Bus Mode Control Register = 0

A typical connection when using external extension memory is shown in Figure 15.4.1. (External extension memory can only be used in external extension mode and processor mode.)

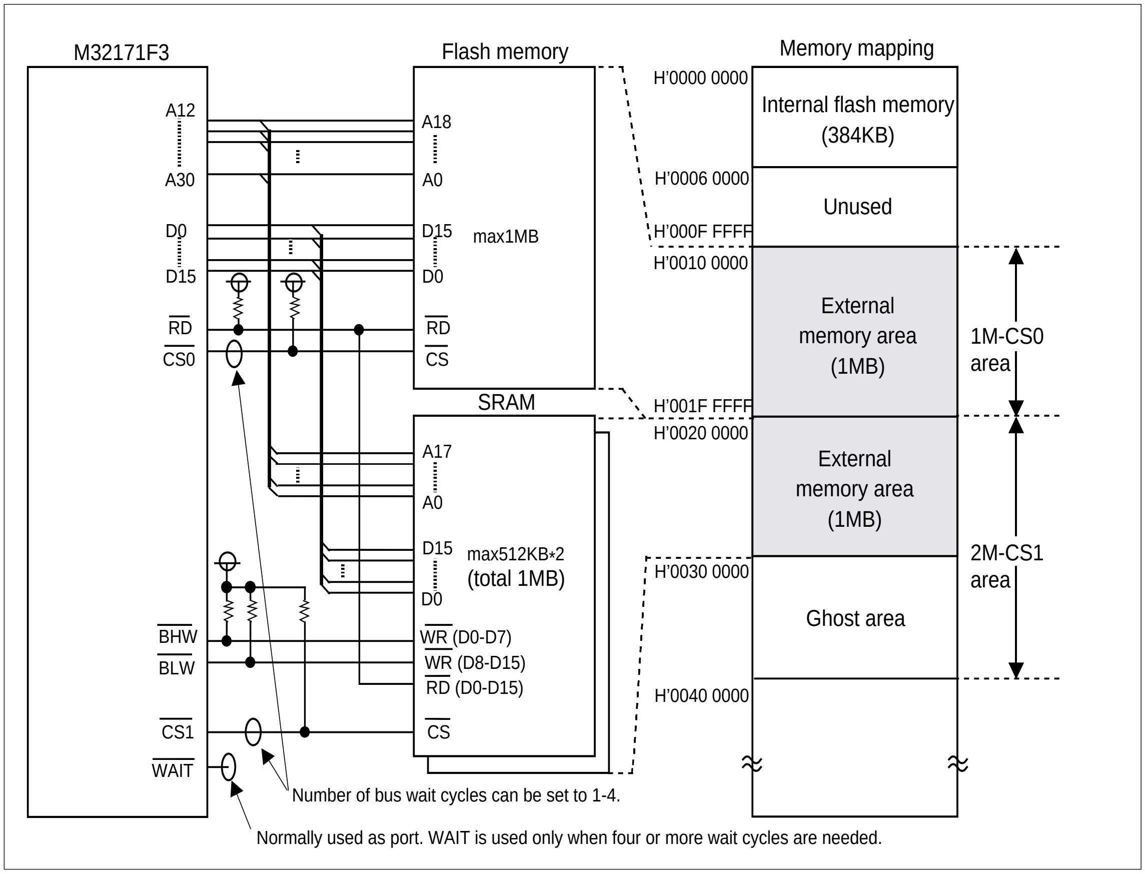


**Figure 15.4.1 Typical Connection of External Extension Memory (When BUSMOD = 0)**

Note: • The 32171 addresses and data are arranged in such a way that bit 0 = MSB, and bit 15 = LSB. Therefore, the MSB and LSB sides must be reversed when connecting external extension memory.

### (2) When Bus Mode Control Register = 1

A typical connection when using external extension memory is shown in Figure 15.4.2. (External extension memory can only be used in external extension mode and processor mode.)

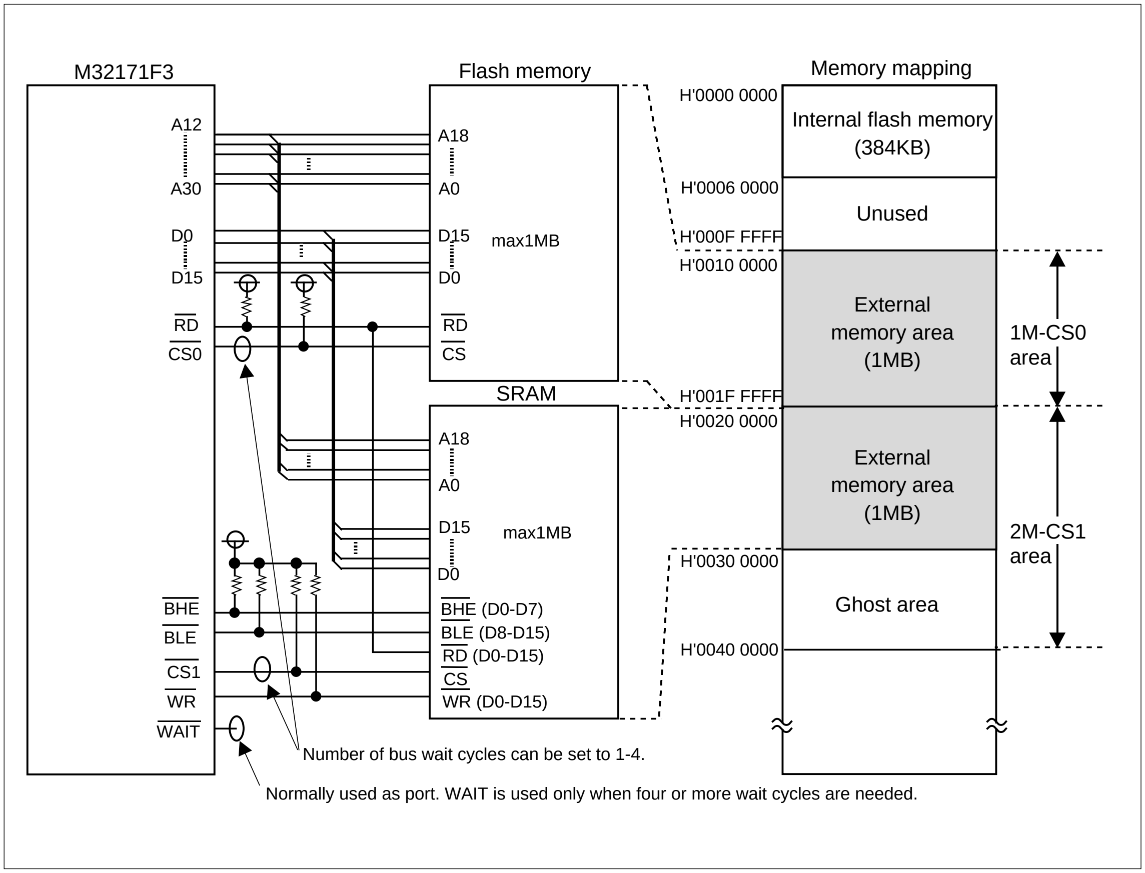
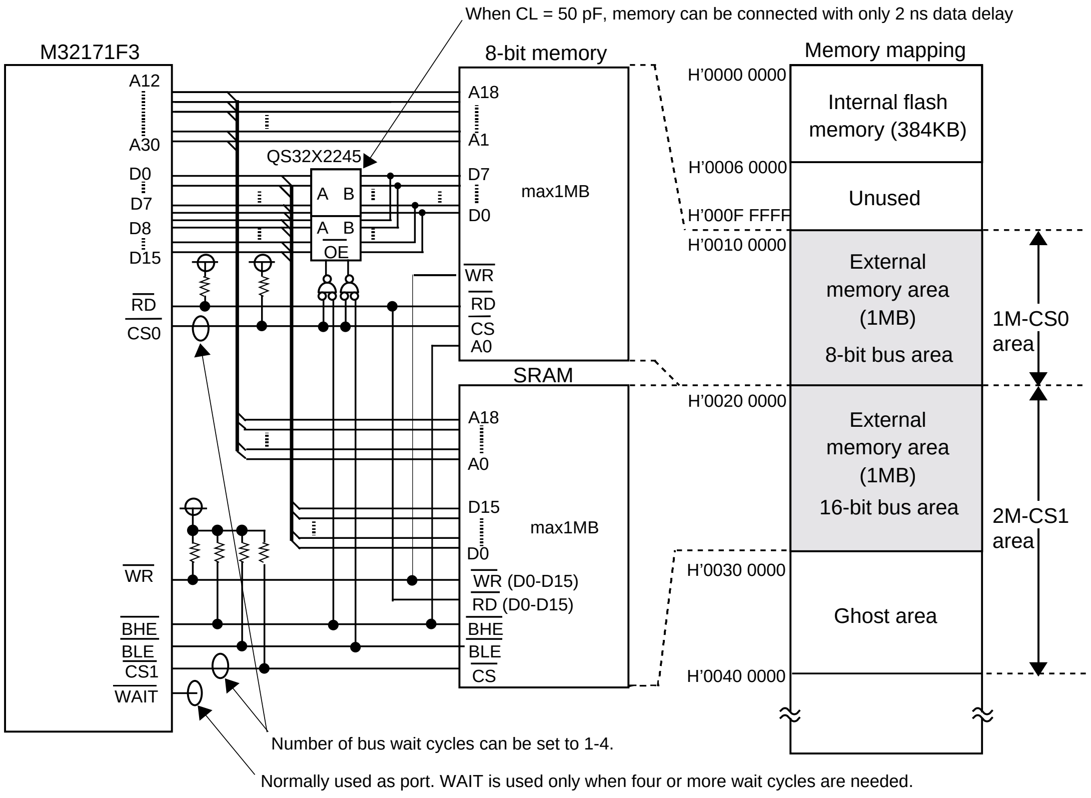


**Figure 15.4.2 Typical Connection of External Extension Memory (When BUSMOD = 1)**

Note: • The 32171 addresses and data are arranged in such a way that bit 0 = MSB, and bit 15 = LSB. Therefore, the MSB and LSB sides must be reversed when connecting external extension memory.

### (3) Using 8/16-bit data bus memories in combination when Bus Mode Control Register = 1

The diagram below shows a typical connection of external extension memory, with 8-bit data bus memory located in the CS0 area, and 16-bit data bus memory located in the CS1 area. (External extension memory can only be used in external extension mode and processor mode.)



When CL = 50 pF, memory can be connected with only 2 ns data delay

Number of bus wait cycles can be set to 1-4.

Normally used as port. WAIT is used only when four or more wait cycles are needed.

Note: • The QS32X2245 is a product made by IDT Company.

**Figure 15.4.3 Typical Connection of External Extension Memory (Using 8/16-bit Mixed Memories when BUSMOD = 1)**

Note: • The 32171 addresses and data are arranged in such a way that bit 0 = MSB, and bit 15 = LSB. Therefore, the MSB and LSB sides must be reversed when connecting external extension memory.

CHAPTER 16

# WAIT CONTROLLER

16.1 Outline of the Wait Controller

16.2 Wait Controller Related Registers

16.3 Typical Operation of the Wait Controller

## 16.1 Outline of the Wait Controller

The wait controller controls the number of wait cycles inserted in bus cycles during access to an external extension area. The following outlines the wait controller.

**Table 16.1.1 Outline of the Wait Controller**

| Item | Specification |
|---|---|
| Target space | Wait cycles in following memory spaces are controlled depending on operation mode<br>Single-chip mode : No target space (Wait controller settings have no effect)<br>External extension mode : CS0 area (1 Mbytes), CS1 area (1 Mbytes)<br>Processor mode : CS0 area (1 Mbytes), CS1 area (1 Mbytes) |
| Number of wait cycles that can be inserted | 1 to 4 wait cycles inserted by software + any number of wait cycles inserted from $\overline{\text{WAIT}}$ pin (Bus cycles with 1 wait cycle are the shortest bus cycle for external access.) |

In external extension mode and processor mode, two chip select signals ($\overline{\text{CS0}}$, $\overline{\text{CS1}}$) are output to an external extension area. Two areas in it corresponding to $\overline{\text{CS0}}$ and $\overline{\text{CS1}}$ signals are called the CS0 and the CS1 areas, respectively.

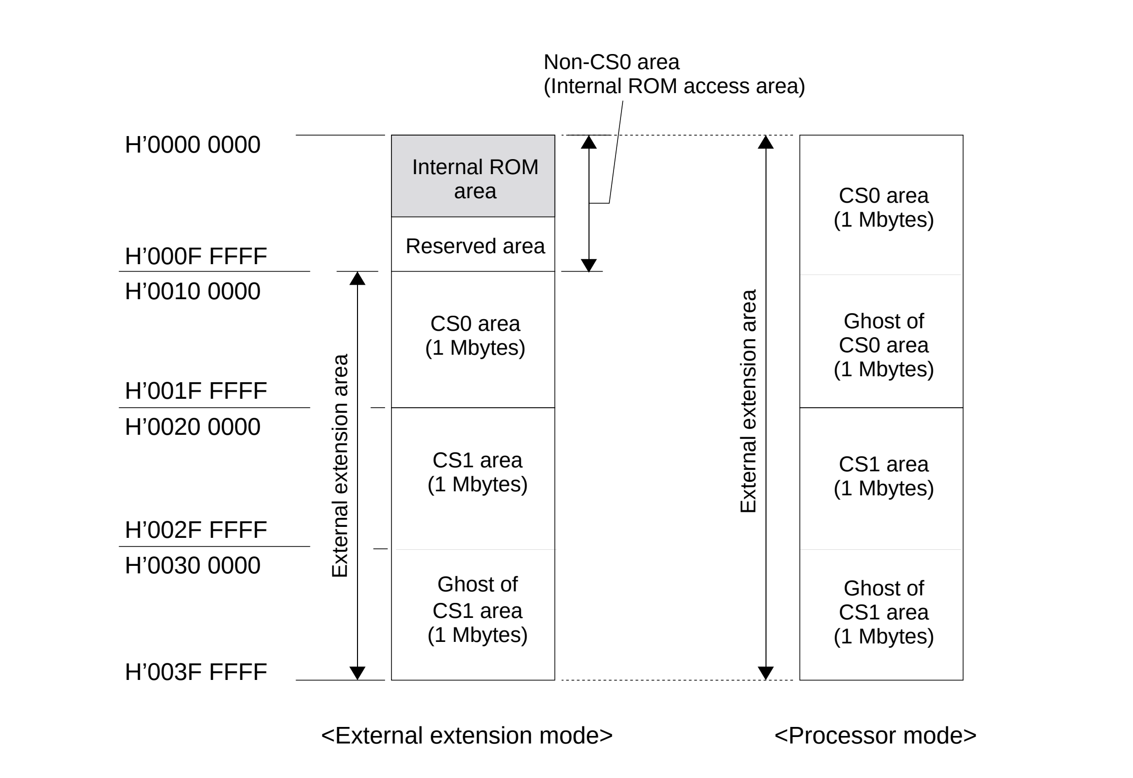


**Figure 16.1.1 CS0 and CS1 Area Address Map**

When accessing an external extension area, the wait controller controls the number of wait cycles to be inserted in bus cycles based on the number of wait cycles set by software and those entered from the $\overline{\text{WAIT}}$ pin.

The number of wait cycles that can controlled in software is 1 to 4. (For external access, bus cycles with 1 wait cycle are the shortest bus cycle.)

When the $\overline{\text{WAIT}}$ pin input is sampled low in the last cycle of internal wait cycles set by software, the wait cycle is extended as long as the $\overline{\text{WAIT}}$ signal is held low. Then when the $\overline{\text{WAIT}}$ signal is released back high, the wait cycle is terminated and the next new bus cycle is entered into.

**Table 16.1.2 Number of Wait Cycles that Can be Set by the Wait Controller**

| External Extension Area | Address | Number of Wait Cycles Inserted |
|---|---|---|
| CS0 area | H'0010 0000 - H'001F FFFF<br>(External extension mode)<br>H'0000 0000 - H'000F FFFF<br>(Processor mode) (Note 1) | One to 4 wait cycles set by software + any number of wait cycles entered from $\overline{\text{WAIT}}$ pin<br>(However, wait cycles set by software have priority.) |
| CS1 area | H'0020 0000 - H'002F FFFF<br>(External extension mode and processor mode) (Note 2) | One to 4 wait cycles set by software + any number of wait cycles entered from $\overline{\text{WAIT}}$ pin<br>(However, wait cycles set by software have priority.) |

Note 1: During processor mode, a ghost (1 Mbyte) of the CS0 area appears in an area of H’0010 0000 through H’001F FFFF.

Note 2: A ghost (1 Mbyte) of the CS1 area appears in an area of H’0030 0000 through H’003F FFFF.

## 16.2 Wait Controller Related Registers

The following shows a wait controller related register map.

| Address | D0 +0 Address D7 | D8 +1 Address D15 |
|---|---|---|
| H'0080 0180 | Wait Cycles Control Register (WTCCR) | |

Blank addresses are reserved area.

**Figure 16.2.1 Wait Controller Related Register Map**

### 16.2.1 Wait Cycles Control Register

**■ Wait Cycles Control Register (WTCCR)** <Address: H'0080 0180>

| D0 | 1 | 2 | 3 | 4 | 5 | 6 | D7 |
|---|---|---|---|---|---|---|---|
| | | CS0WTC | | | | CS1WTC | |

<When reset : H'00>

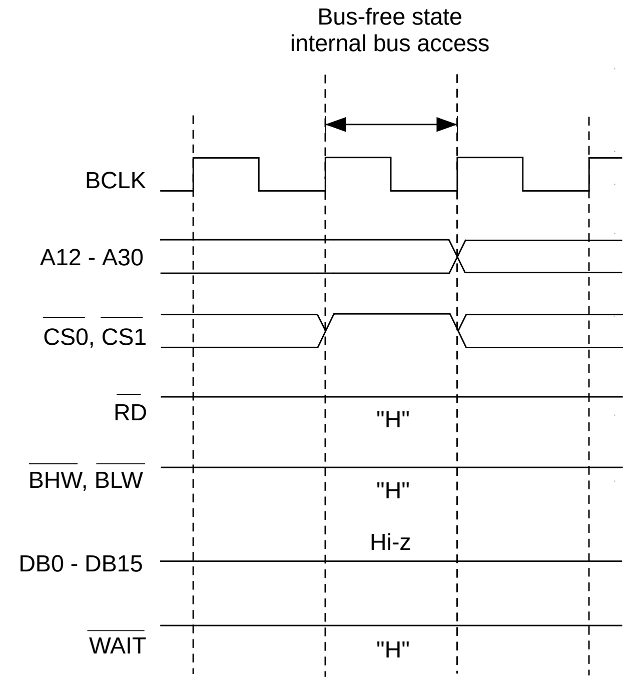
| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 0 , 1 | No functions assigned | | 0 | — |
| 2 , 3 | CS0WTC<br>(CS0 wait cycles control) | 00 : 4 wait cycles (when reset)<br>01 : 3 wait cycles<br>10 : 2 wait cycles<br>11 : 1 wait cycle | ○ | ○ |
| 4 , 5 | No functions assigned | | 0 | — |
| 6 , 7 | CS1WTC<br>(CS1 wait cycles control) | 00 : 4 wait cycles (when reset)<br>01 : 3 wait cycles<br>10 : 2 wait cycles<br>11 : 1 wait cycle | ○ | ○ |

## 16.3 Typical Operation of the Wait Controller

The following shows a typical operation of the wait controller. The wait controller can control bus access in the range of 2 to 5 cycles. If more access cycles than that are needed, use the WAIT function in combination with the wait controller.

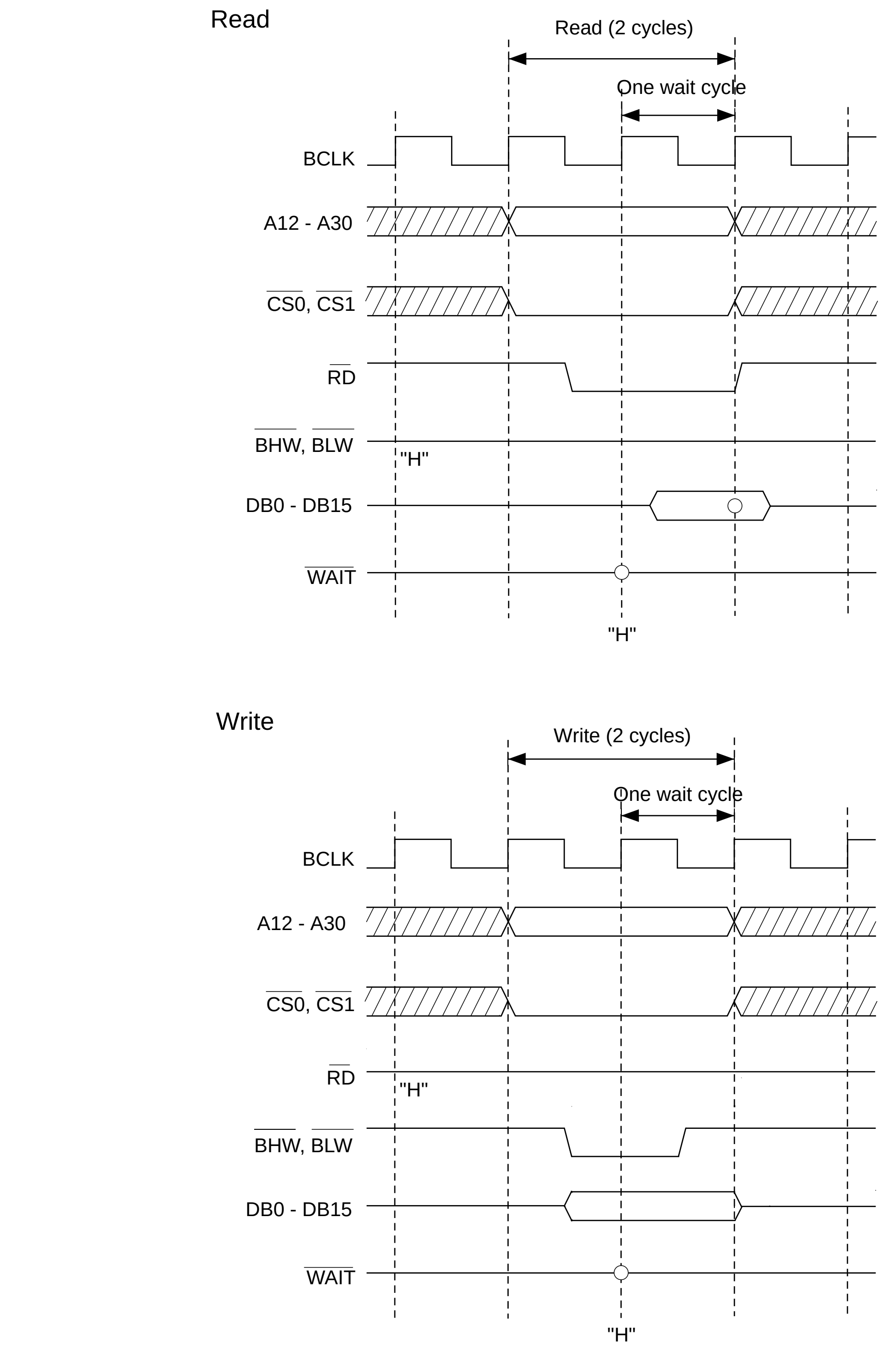
### (1) When Bus Mode Control Register = 0

External read/write operations are performed using the address bus, data bus, and signals $\overline{\text{CS0}}$, $\overline{\text{CS1}}$, $\overline{\text{RD}}$, $\overline{\text{BHW}}$, $\overline{\text{BLW}}$, $\overline{\text{WAIT}}$, and BCLK.



Note: • Hi-Z denotes a high-impedance state.

**Figure 16.3.1 Internal Bus Access during Bus Free State**



Note: • Circles ○ above indicate points at which signals are sampled.

**Figure 16.3.2 Read/Write Timing (for Access with 1 Internal Wait Cycle)**

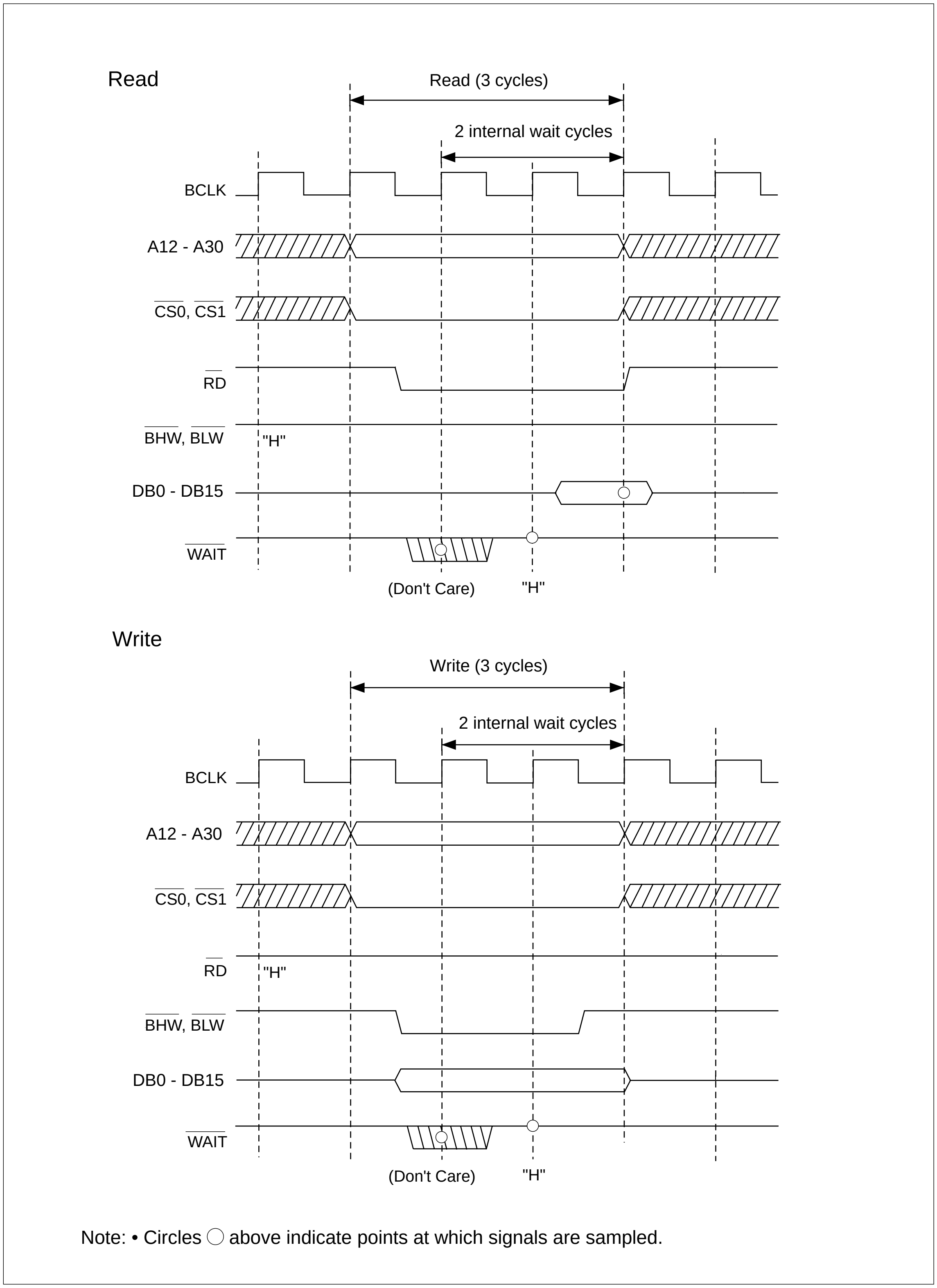


**Figure 16.3.3 Read/Write Timing (for Access with 2 Internal Wait Cycles)**

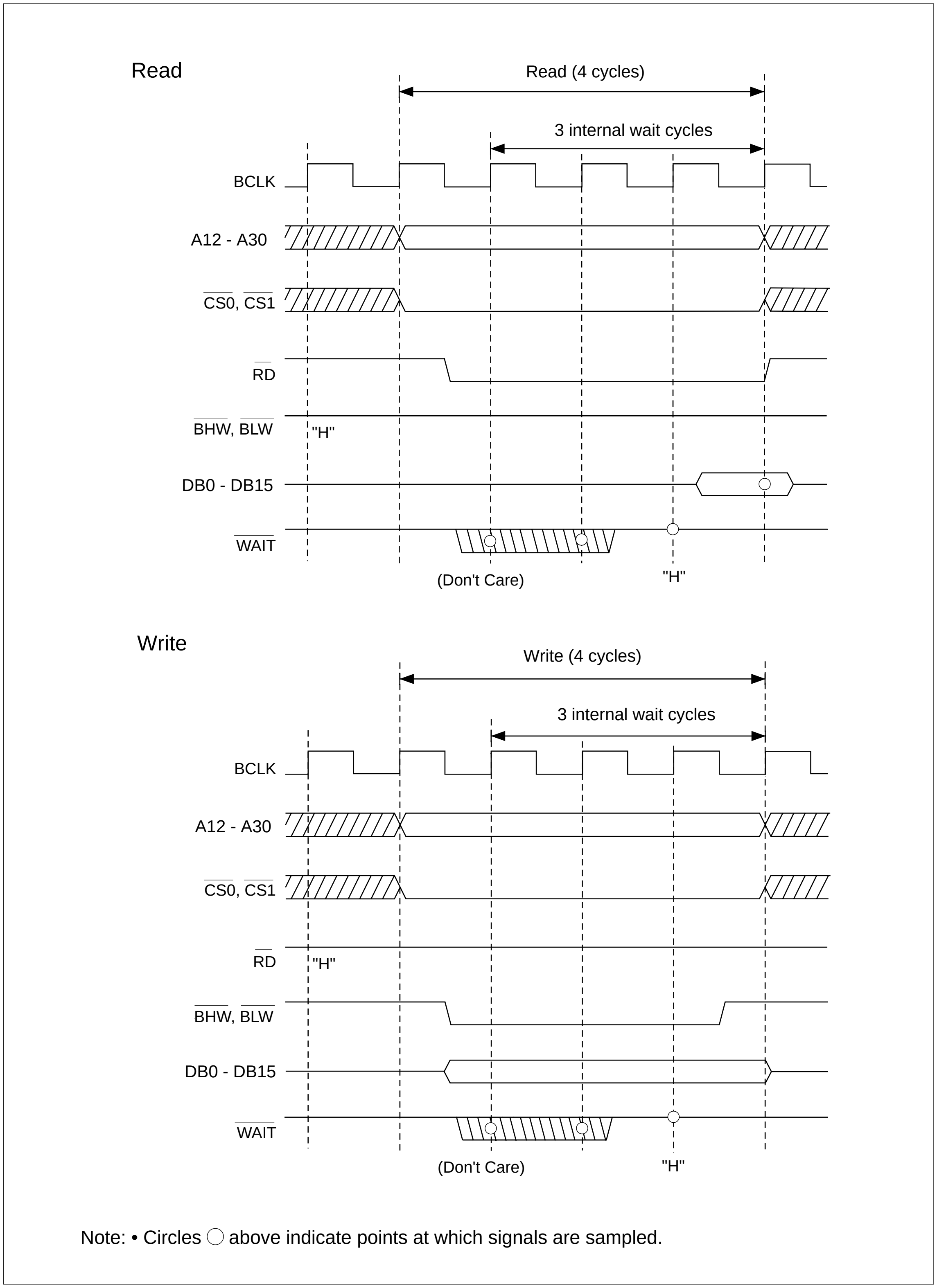
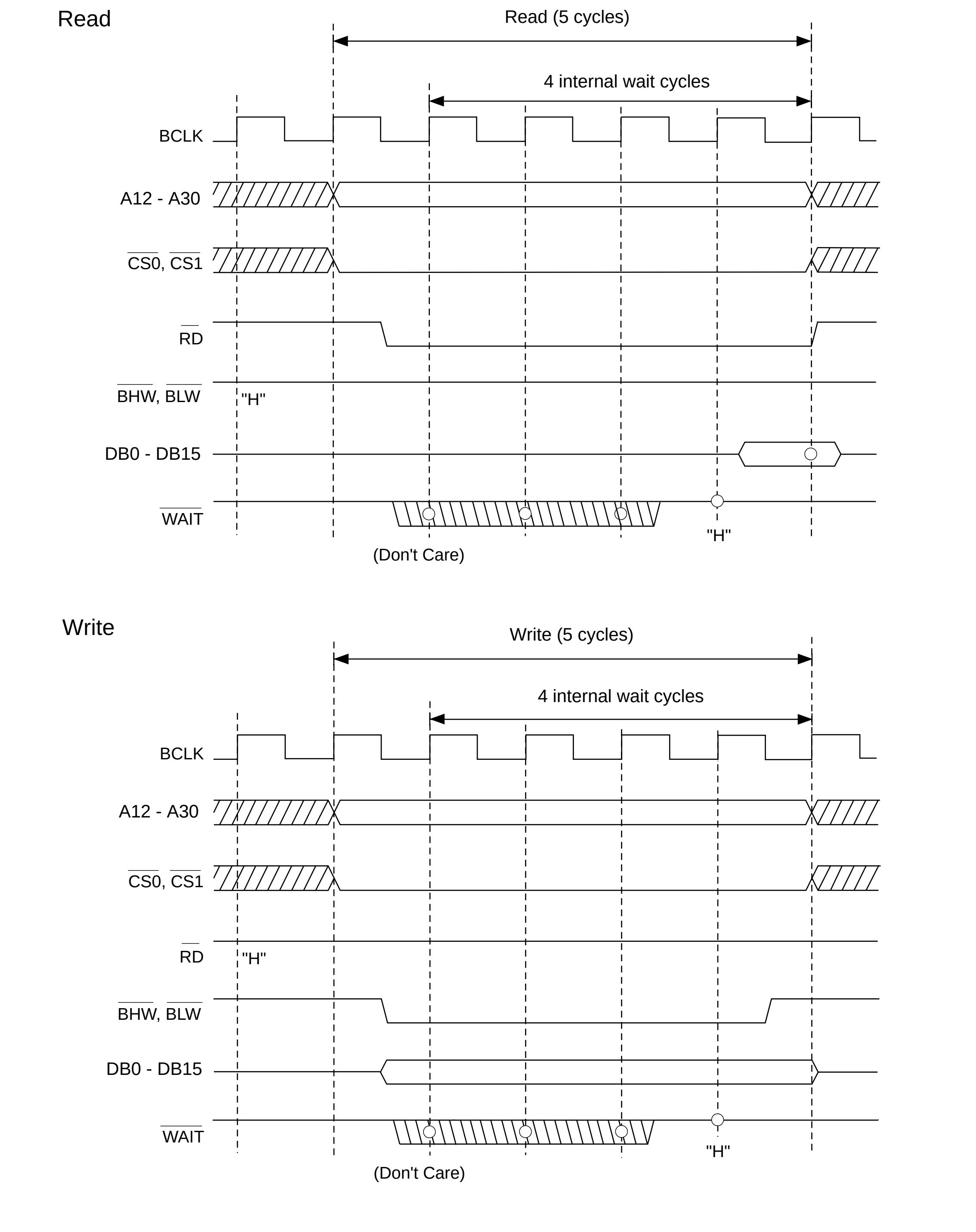
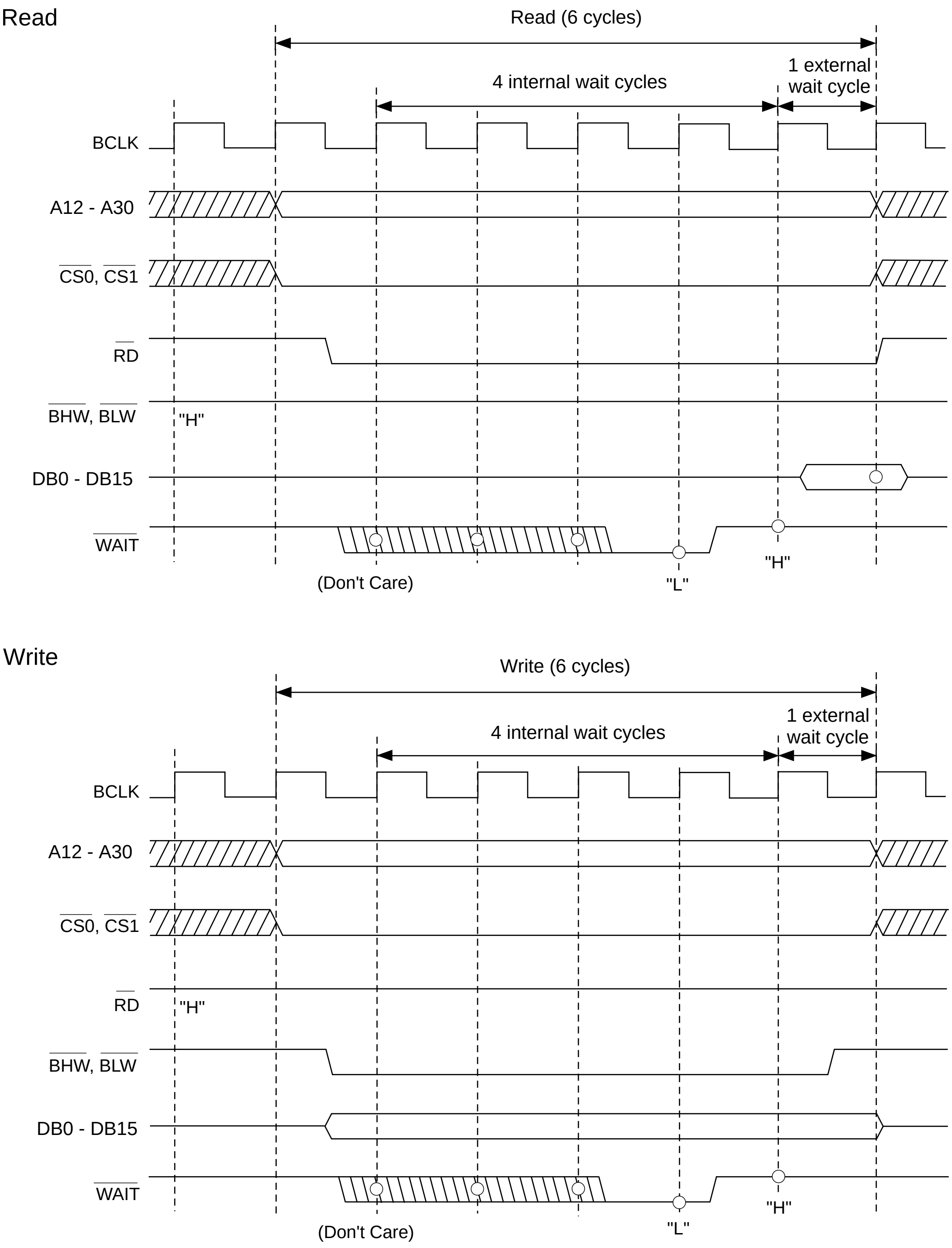


**Figure 16.3.4 Read/Write Timing (for Access with 3 Internal Wait Cycles)**



Note: • Circles ○ above indicate points at which signals are sampled.

**Figure 16.3.5 Read/Write Timing (for Access with 4 Internal Wait Cycles)**



Note: • Circles ○ above indicate points at which signals are sampled.

**Figure 16.3.6 Read/Write Timing (for Access with 4 Internal and 1 External Wait Cycles)**

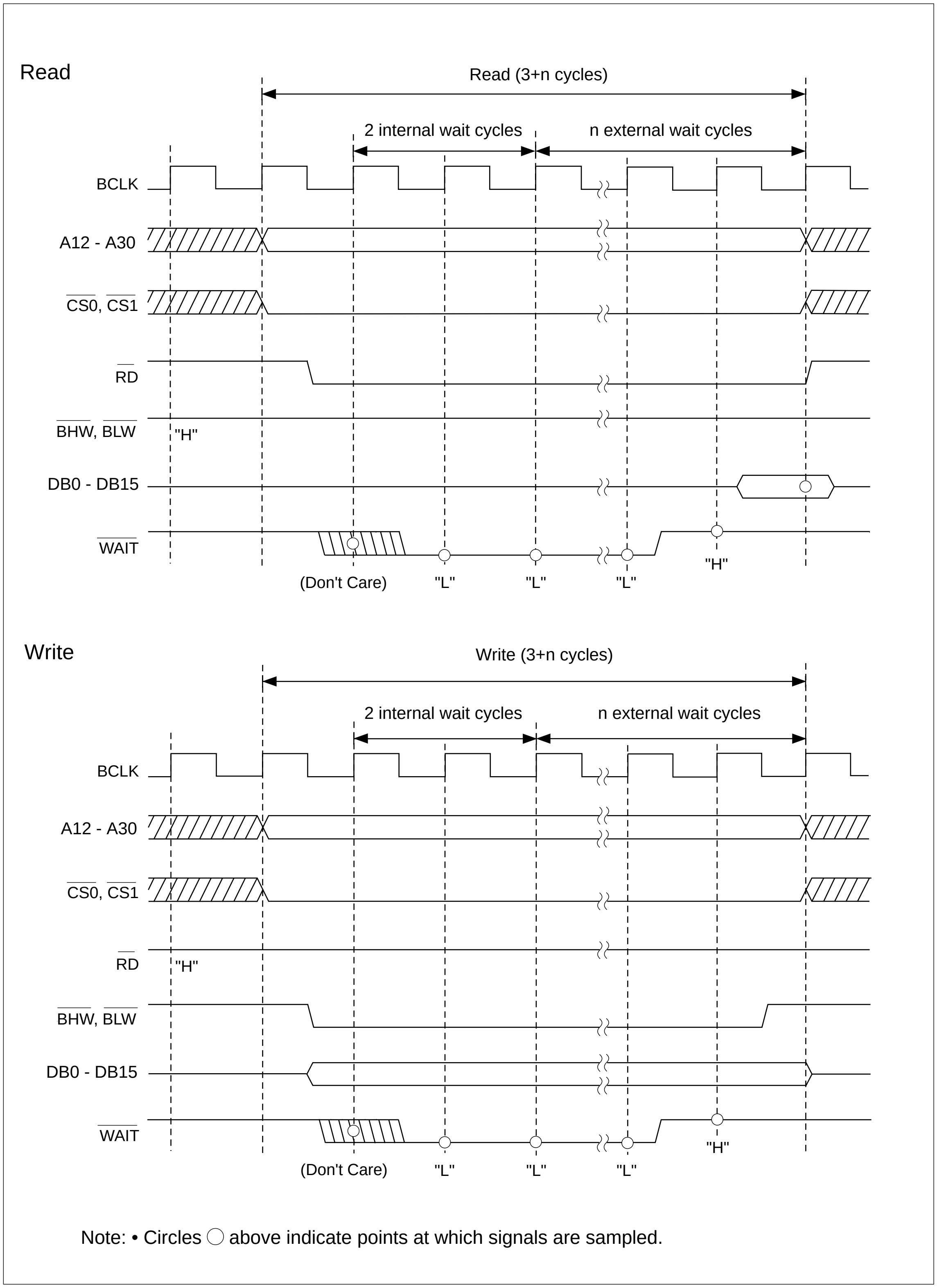
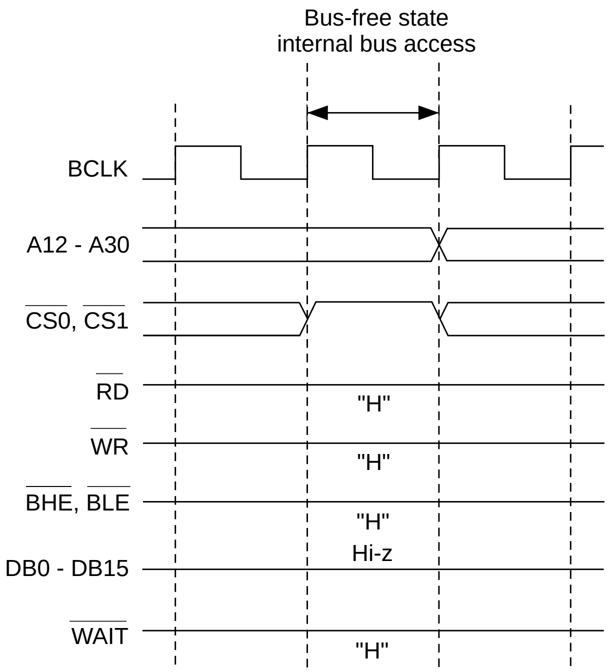


**Figure 16.3.7 Read/Write Timing (for Access with 2 Internal and n External Wait Cycles)**

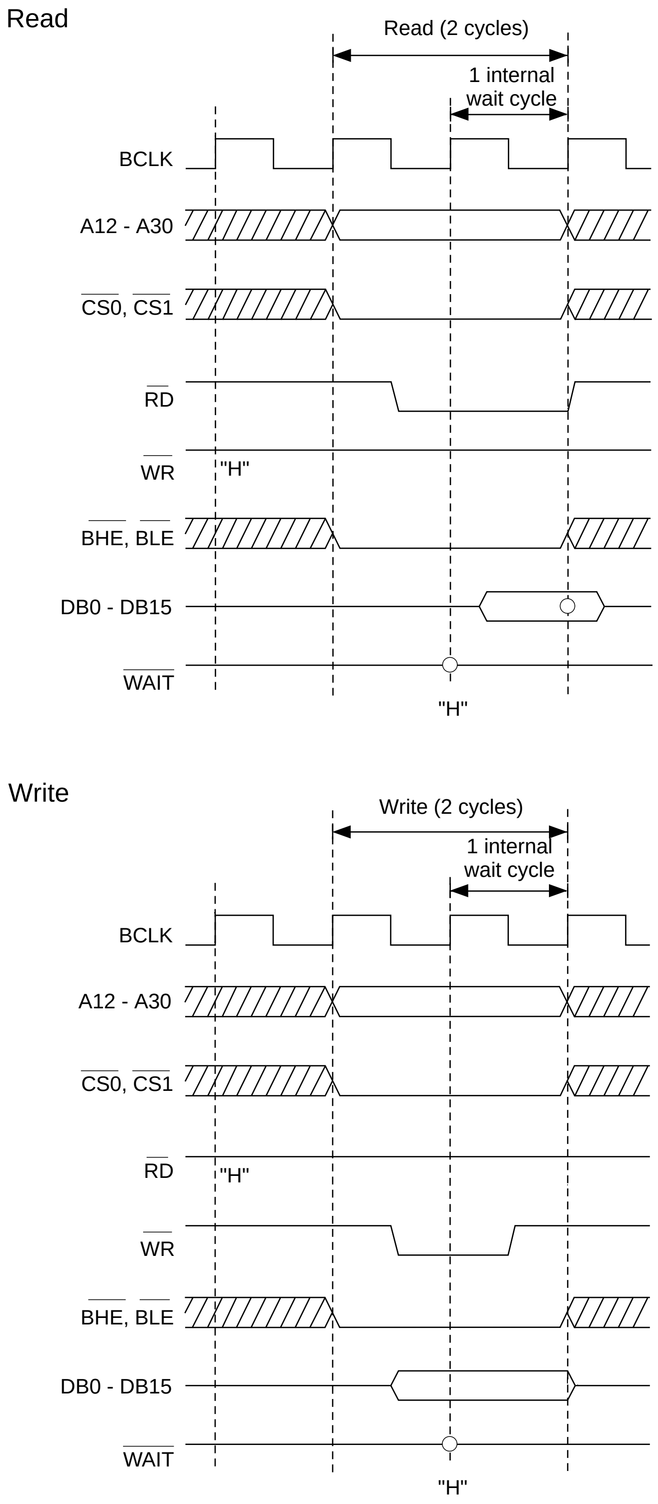
### (2) When Bus Mode Control Register = 1

External read/write operations are performed using the address bus, data bus, and signals $\overline{\text{CS0}}$, $\overline{\text{CS1}}$, $\overline{\text{RD}}$, $\overline{\text{BHE}}$, $\overline{\text{BLE}}$, $\overline{\text{WAIT}}$, and $\overline{\text{WR}}$.



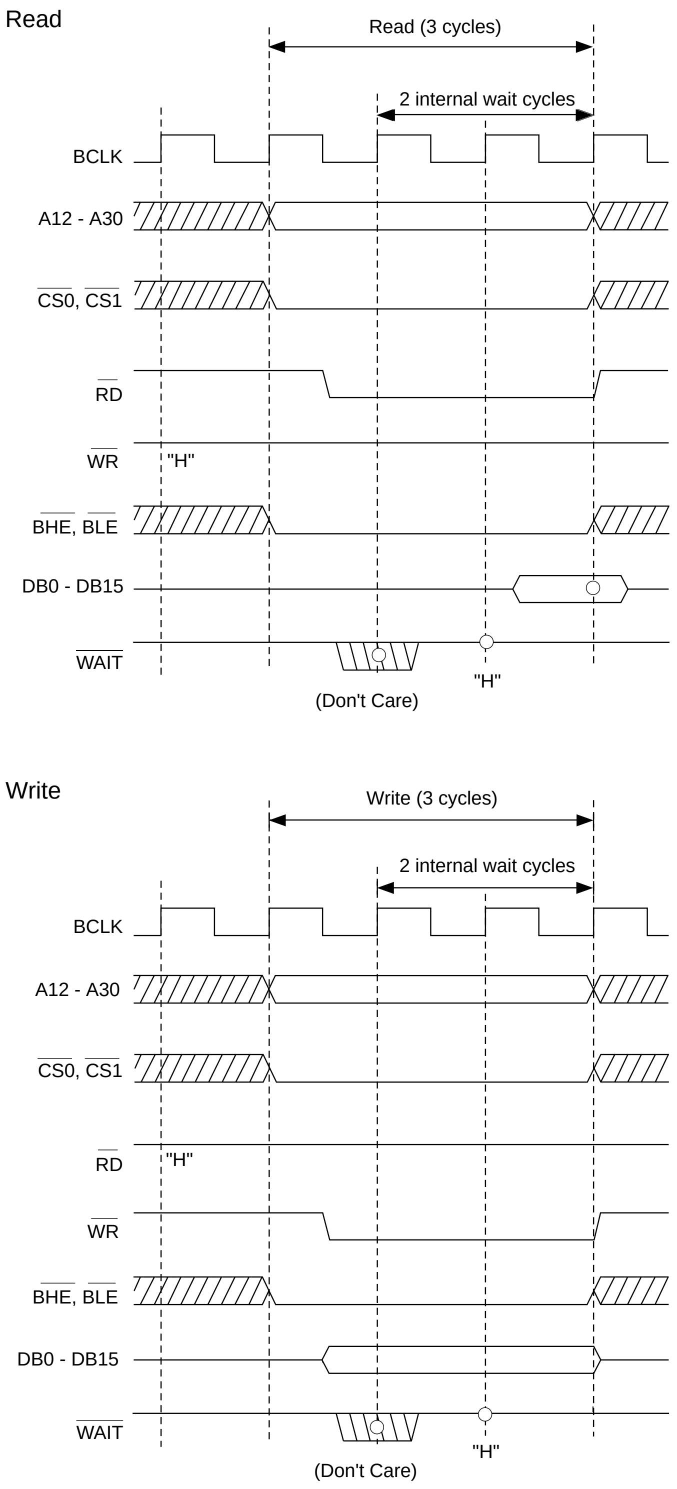
Notes: • Hi-Z denotes a high-impedance state.
• BCLK is not output.

**Figure 16.3.8 Internal Bus Access during Bus Free State**



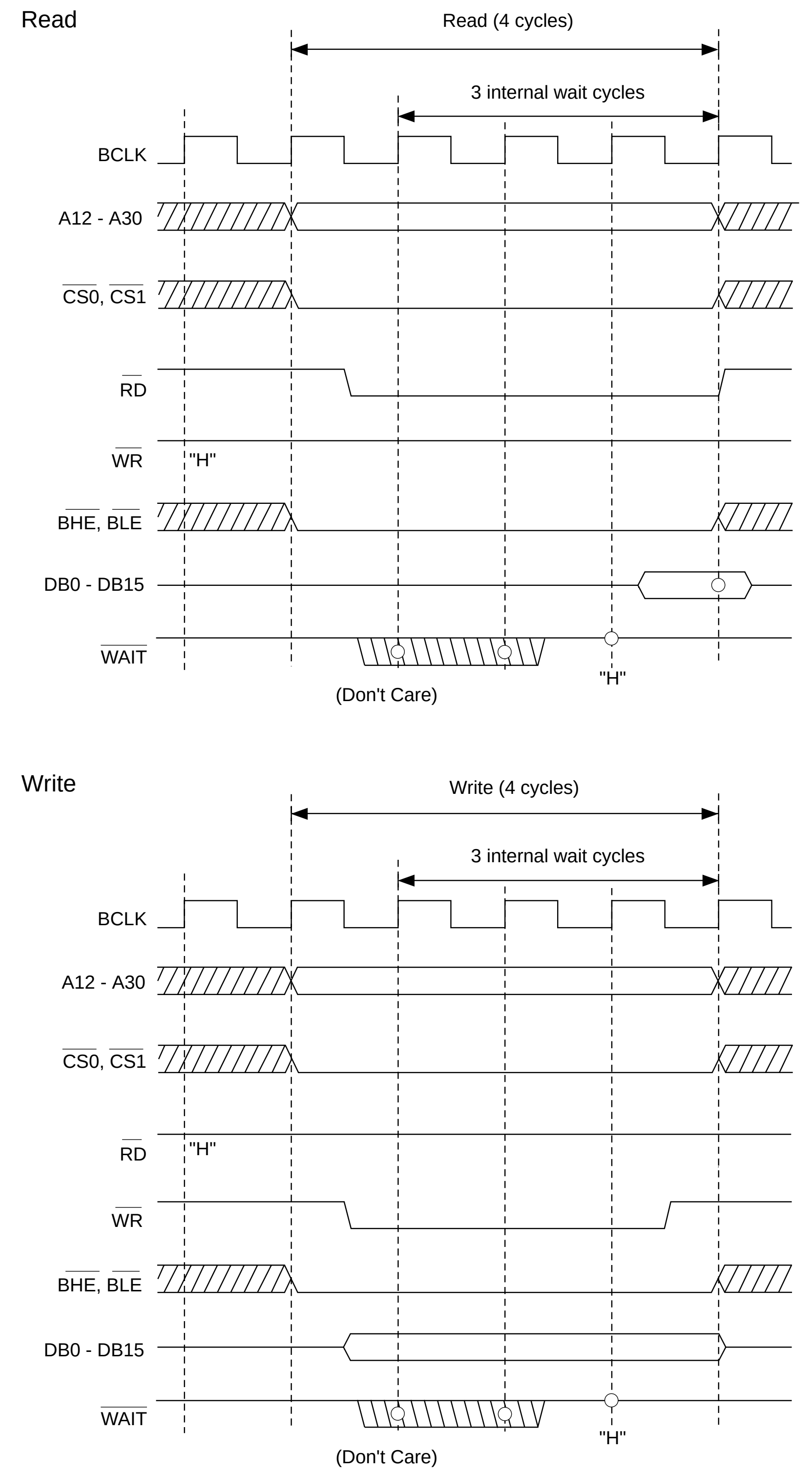
Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 16.3.9 Read/Write Timing (for Access with 1 Internal Wait Cycle)**



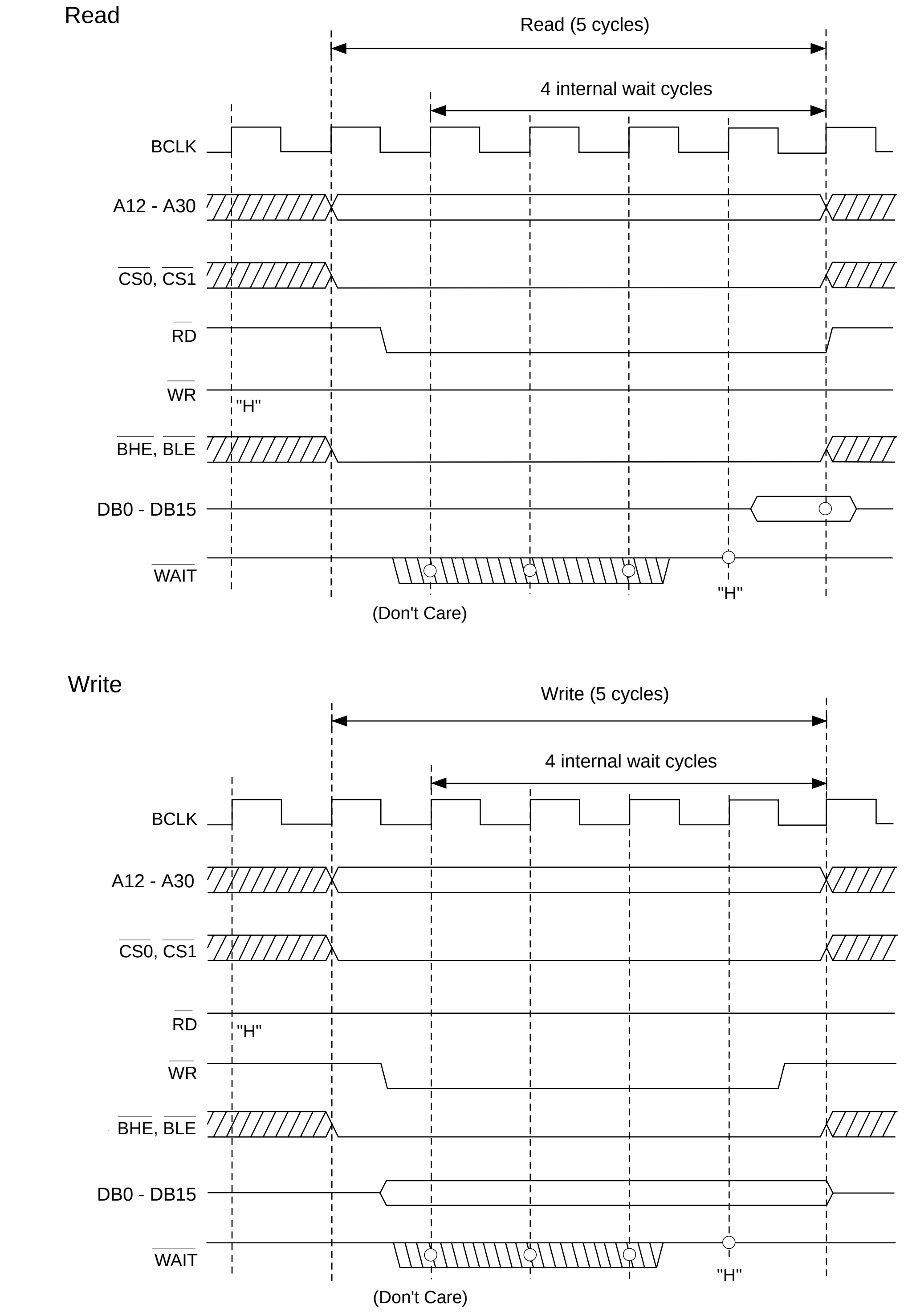
Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 16.3.10 Read/Write Timing (for Access with 2 Internal Wait Cycles)**



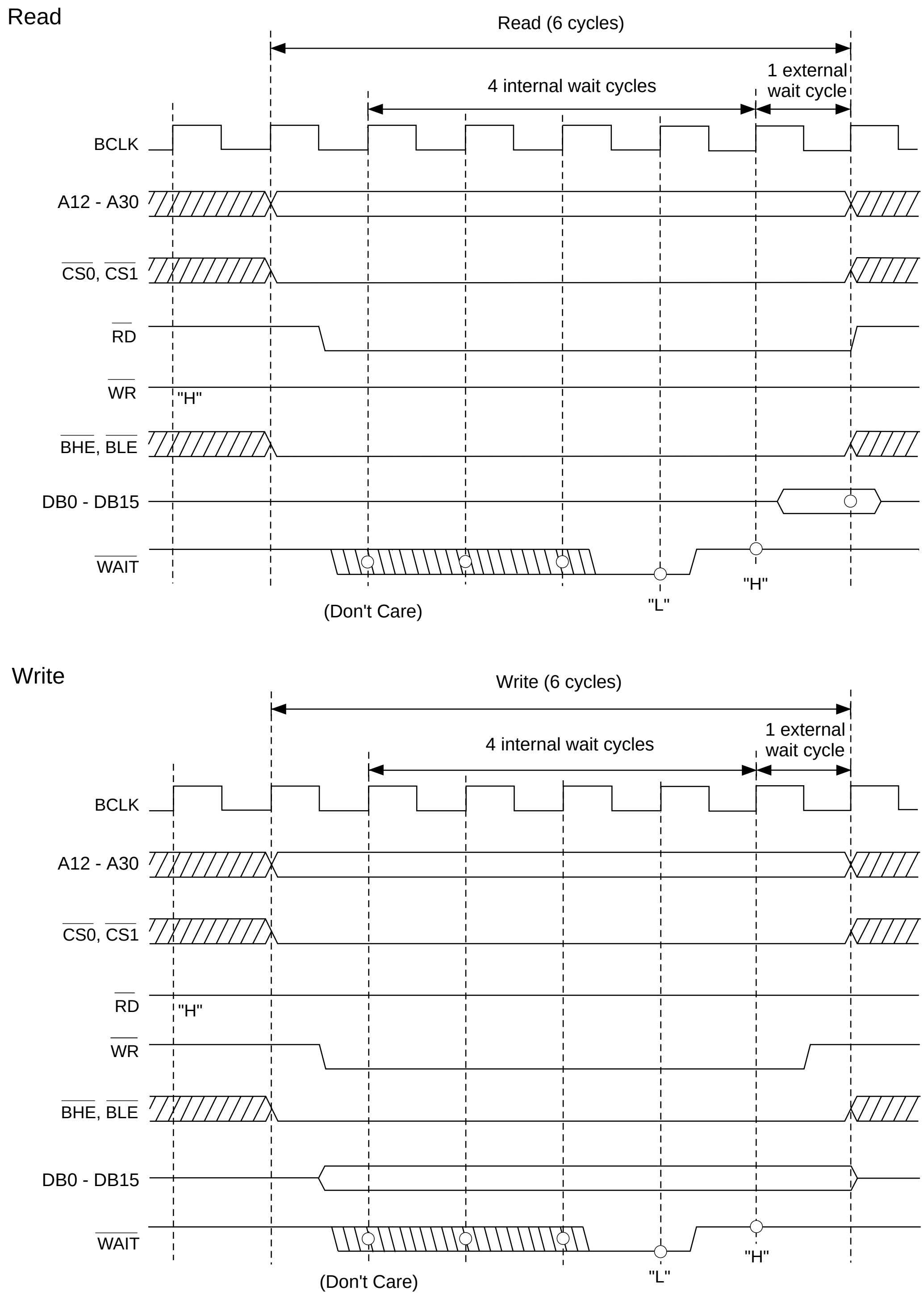
Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 16.3.11 Read/Write Timing (for Access with 3 Internal Wait Cycles)**



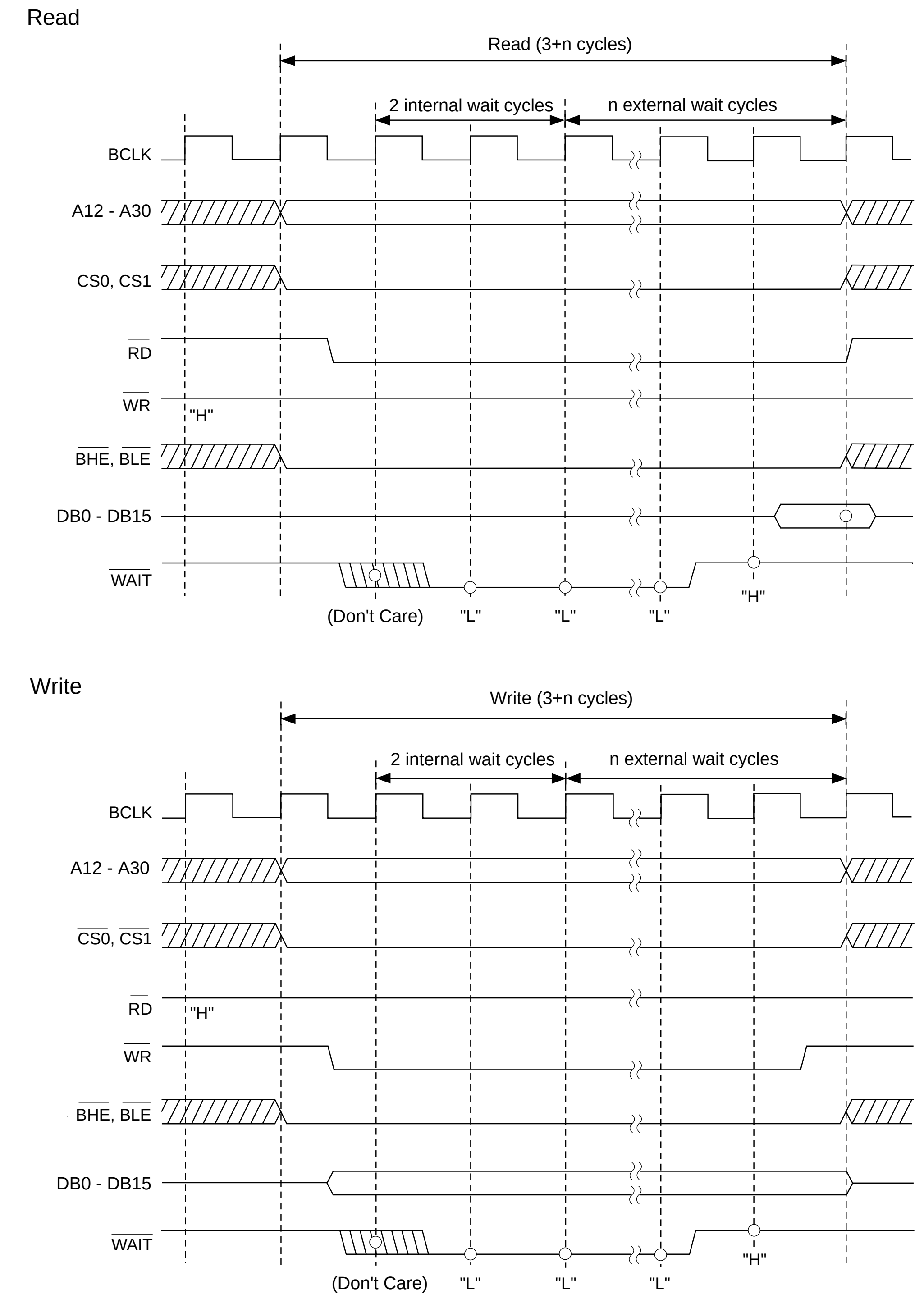
Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 16.3.12 Read/Write Timing (for Access with 4 Internal Wait Cycles)**



Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 16.3.13 Read/Write Timing (for Access with 4 Internal and 1 External Wait Cycles)**



Notes: • Circles ○ above indicate points at which signals are sampled.
• BCLK is not output.

**Figure 16.3.14 Read/Write Timing (for Access with 2 Internal and n External Wait Cycles)**

* This is a blank page.*

# CHAPTER 17
# RAM BACKUP MODE

17.1 Outline of RAM Backup Mode
17.2 Example of RAM Backup when Power is Down
17.3 Example of RAM Backup for Saving Power Consumption
17.4 Exiting RAM Backup Mode (Wakeup)

## 17.1 Outline of RAM Backup Mode

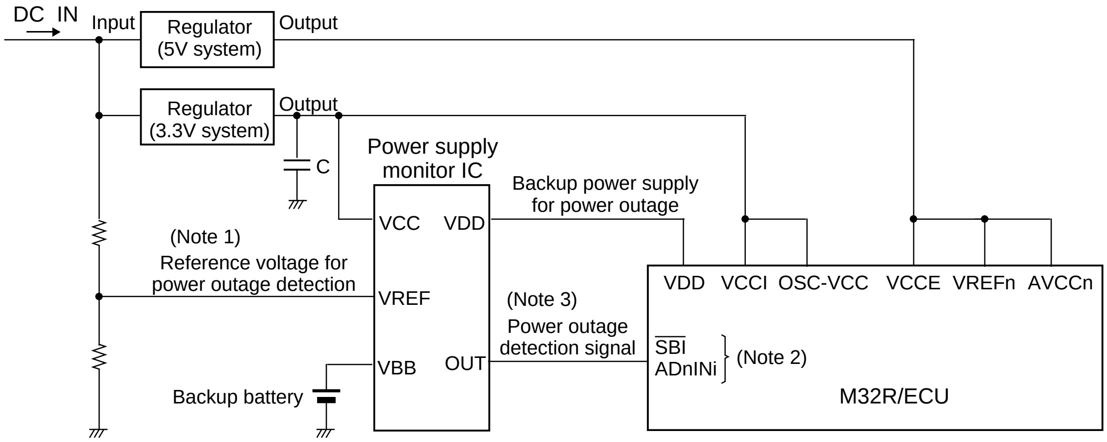
In RAM backup mode, the contents of the internal RAM are retained while the power is turned off. RAM backup mode is used for the following two purposes:

- Back up the internal RAM data when the power is down
- Turn off the power to the CPU whenever necessary to save on the system's power consumption

The 32R/E CPU is placed in RAM backup mode by applying a voltage of 2.0-3.3 V to the VDD pin (provided for RAM backup) and 0 V to all other pins. During RAM backup mode, the contents of the internal RAM are retained, while the CPU and internal peripheral I/O remain idle. Also, because all pins except VDD are held low during RAM backup mode, power consumption in the system can effectively reduced.

## 17.2 Example of RAM Backup when Power is Down

A typical circuit for RAM backup at power outage is shown in Figure 17.2.1. The following explains how the RAM can be backed up by using this circuit as an example.

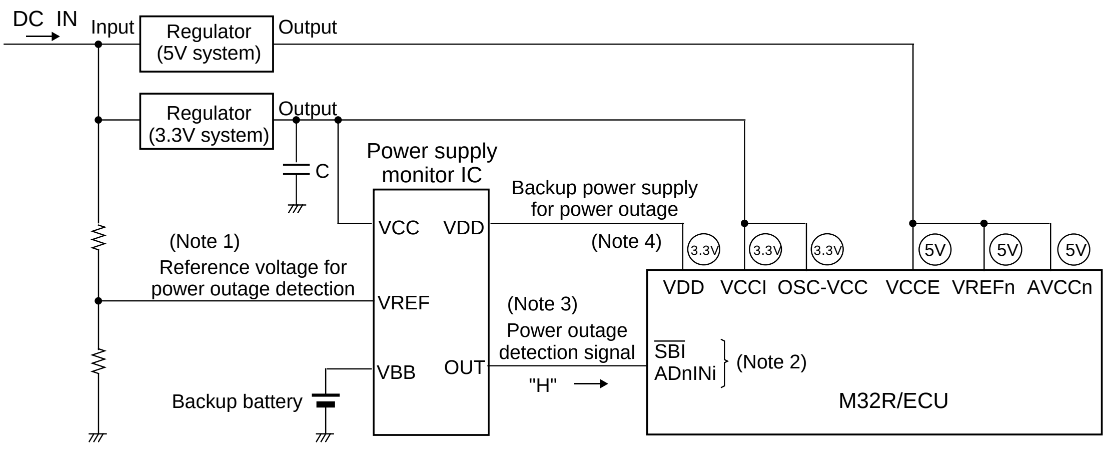


Note 1 : Power outage is detected by the DC IN (regulator input) voltage.
Note 2 : These pins are used to detect a RAM backup signal.
Note 3 : This pin outputs a high when the power is on and outputs a low when the power is down.

**Figure 17.2.1 Typical Circuit for RAM Backup at Power Outage**

### 17.2.1 Normal Operating State

Figure 17.2.2 shows the normal operating state of the M32R/ECU. During normal operation, input on the $\overline{\text{SBI}}$ pin or ADnINi (i = 0-15) pin used for RAM backup signal detection remains high.



Note 1 : Power outage is detected by the DC IN (regulator input) voltage.
Note 2 : These pins are used to detect a RAM backup signal.
Note 3 : This pin outputs a high when the power is on and outputs a low when the power is down.
Note 4 : Backup power supply = 2.0 to 3.3 V

**Figure 17.2.2 Normal Operating State**

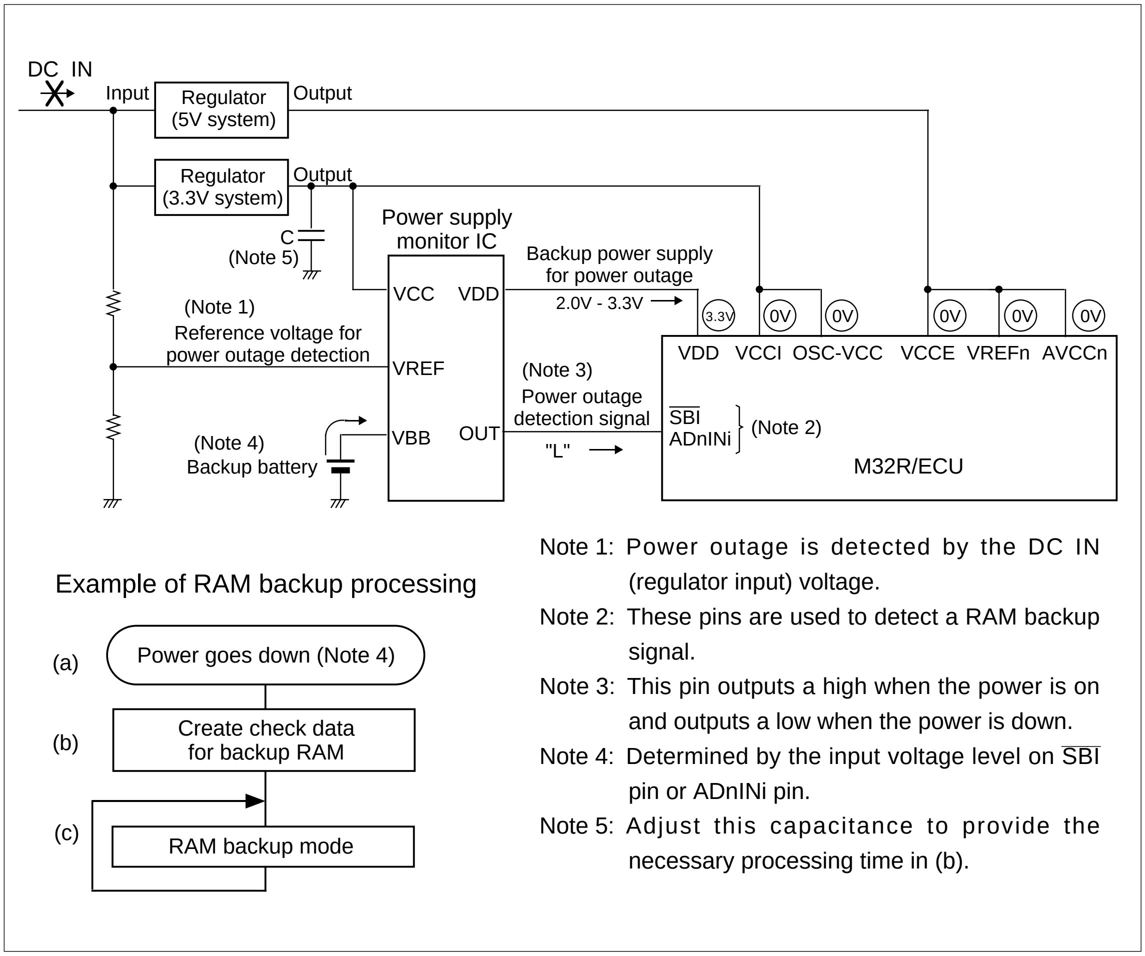
### 17.2.2 RAM Backup State

Shown in Figure 17.2.3 is the power outage RAM backup state of the M32R/ECU. When the power supply goes down, the power supply monitor IC starts feeding current from the backup battery to the M32R/ECU. Also, the power supply monitor IC's power outage detection pin outputs a low, causing the $\overline{\text{SBI}}$ pin or ADnINi pin input to go low, which generates a RAM backup signal ((a) in Figure 17.2.3). Whether the power is down or not must be determined with respect to the DC IN (regulator input) voltage in order to allow for a software processing time at power outage.

To enable RAM backup mode, make the following settings.

(1) Create check data to verify after returning from RAM backup to normal mode whether the RAM data has been retained normally ((b) in Figure 17.2.3).

When the power supply to VCC goes down after settings in (1), the voltage applied to the VDD pin becomes 2.0-3.3 V and voltages applied to all other pins drop to 0 V, and the M32R/ECU thereby enters RAM backup mode ((c) in Figure 17.2.3).



Note 1: Power outage is detected by the DC IN (regulator input) voltage.

Note 2: These pins are used to detect a RAM backup signal.

Note 3: This pin outputs a high when the power is on and outputs a low when the power is down.

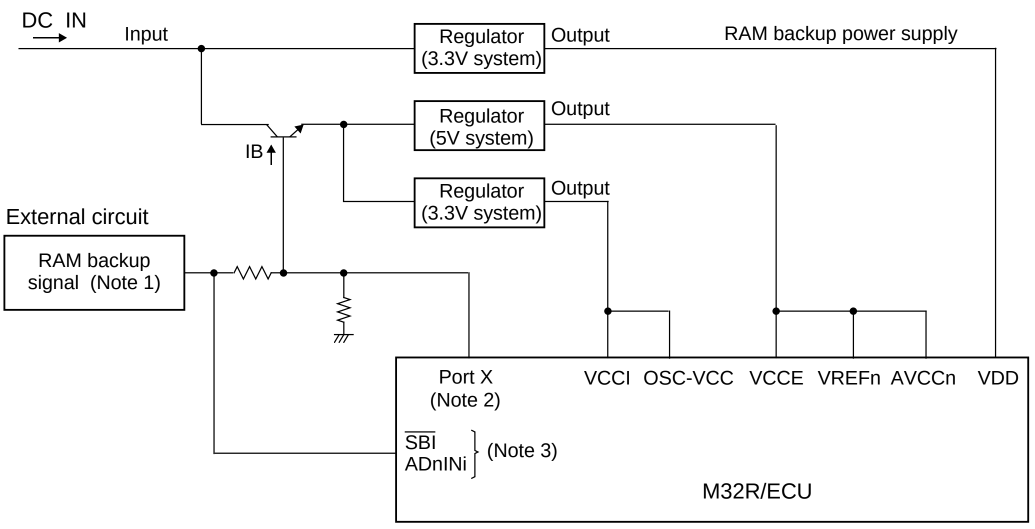
Note 4: Determined by the input voltage level on $\overline{\text{SBI}}$ pin or ADnINi pin.

Note 5: Adjust this capacitance to provide the necessary processing time in (b).

**Figure 17.2.3 RAM Backup State at Power Outage**

## 17.3 Example of RAM Backup for Saving Power Consumption

Figure 17.3.1 shows a typical circuit for RAM backup to save on power consumption. The following explains how the RAM is backed up for the purpose of low-power operation by using this circuit as an example.



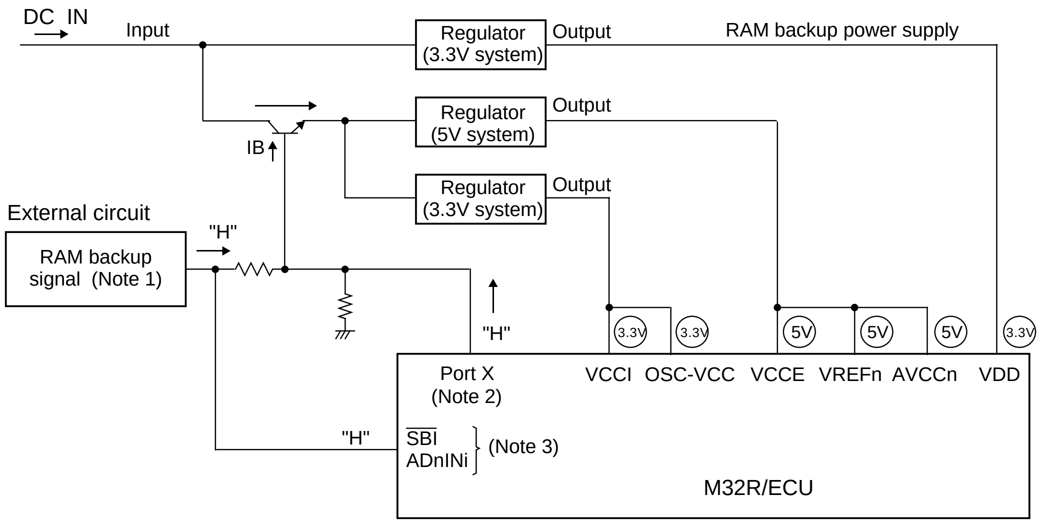
Note 1 : This signal outputs a low for RAM backup.
Note 2 : This pin outputs a high when the power is on, and is set for input mode when in RAM backup mode.
Note 3 : These pins are used to detect a RAM backup signal.

**Figure 17.3.1 Typical Circuit for RAM Backup to Save on Power Consumption**

### 17.3.1 Normal Operating State

Figure 17.3.2 shows the normal operating state of the M32R/ECU. During normal operation, the RAM backup signal output by the external signal is high. Also, input on the SBI pin or ADnINi (i = 0-15) pin used for RAM backup signal detection remains high.

Port X, which is the transistor's base connecting pin, should output a high. This causes the transistor's base voltage, IB, to go high, so that current is fed from the power supply to the VCC pin via the transistor.



Note 1 : This signal outputs a low for RAM backup.

Note 2 : This pin outputs a high when the power is on, and is set for input mode when in RAM backup mode (One of the port pins selected).

Note 3 : These pins are used to detect a RAM backup signal.

**Figure 17.3.2 Normal Operating State**

### 17.3.2 RAM Backup State

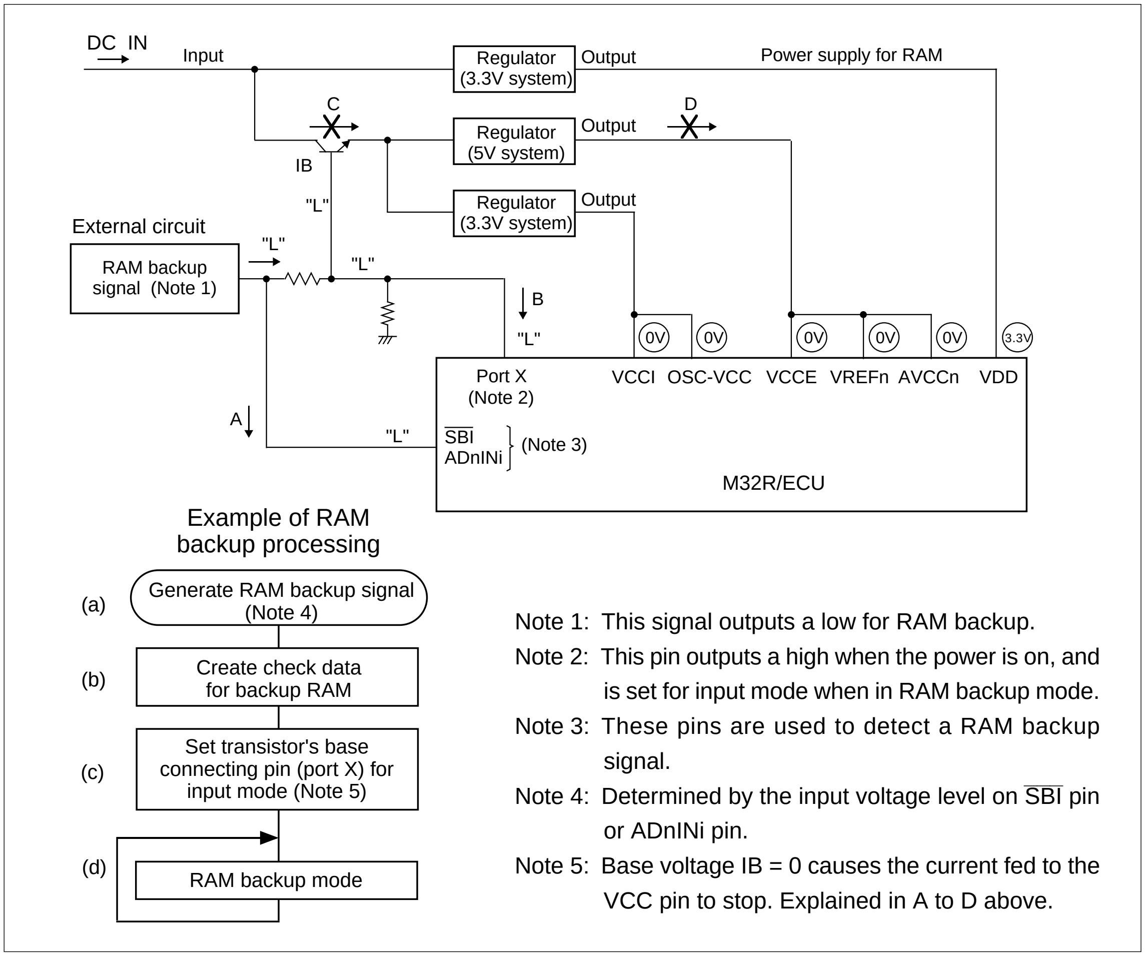
Figure 17.3.3 shows the RAM backup state of the M32R/ECU. Figure 17.3.4 shows a RAM backup sequence. When the external circuit outputs a low, input on the $\overline{\text{SBI}}$ pin or ADnINi pin goes low. A low on these input pins generates a RAM backup signal (A and (a) in Figure 17.3.3). To enable RAM backup mode, make the following settings.

(1) Create check data to verify after returning from RAM backup to normal mode whether the RAM data has been retained normally ((b) in Figure 17.3.3).

(2) To materialize low-power operation, set all programmable input/output pins except port X for input mode (or for output mode, with pins outputting a low) ((c) in Figure 17.3.3).

(3) Set port X for input mode (B and (d) in Figure 17.3.3). This causes the transistor's base voltage, IB, to go low, so that no current flows from the power supply to the VCC pin via the transistor (C in Figure 17.3.3). Consequently, the power to the VCC pin is shut off (D in Figure 17.3.3).

Due to settings in (1) to (3), the voltage applied to the VDD pin becomes 3.3 V ± 10% and voltages applied to all other pins drop to 0 V, thus placing the M32R/ECU in RAM backup mode ((d) in Figure 17.3.3).



Note 1: This signal outputs a low for RAM backup.

Note 2: This pin outputs a high when the power is on, and is set for input mode when in RAM backup mode.

Note 3: These pins are used to detect a RAM backup signal.

Note 4: Determined by the input voltage level on $\overline{\text{SBI}}$ pin or ADnINi pin.

Note 5: Base voltage IB = 0 causes the current fed to the VCC pin to stop. Explained in A to D above.

**Figure 17.3.3 RAM Backup State for Low-Power Operation**

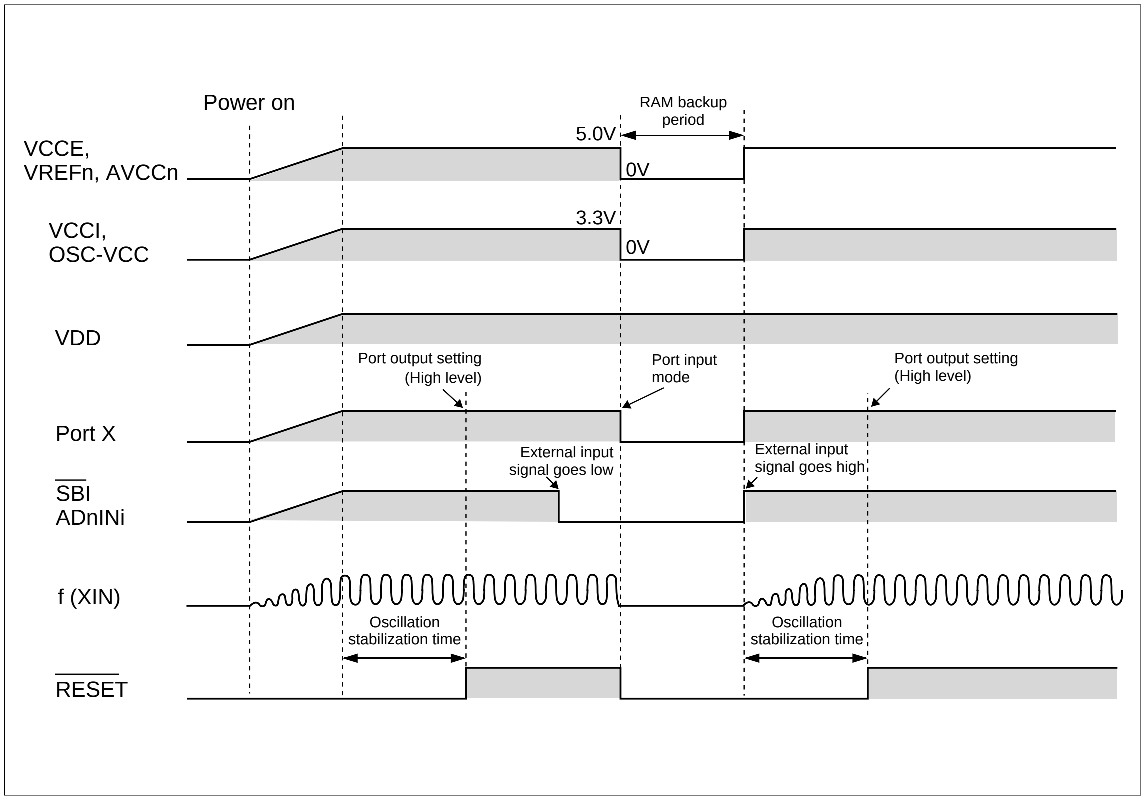

Power on
RAM backup period
5.0V
0V
VCCE, VREFn, AVCCn
3.3V
0V
VCCI, OSC-VCC
VDD
Port output setting (High level)
Port input mode
Port output setting (High level)
Port X
External input signal goes low
External input signal goes high
SBI
ADnINi
f (XIN)
Oscillation stabilization time
Oscillation stabilization time
RESET

**Figure 17.3.4 Example of RAM Backup Sequence for Low-Power Operation**

### 17.3.3 Precautions to Be Observed at Power-on

When changing port X from input mode to output mode after power-on, pay attention to the following.

If port X is set for output mode while no data is set in the Port X Data Register, the port's initial output level is indeterminate. Therefore, be sure to set the output high level in the Port X Data Register before you set port X for output mode. Unless this method is followed, port output may go low at the same time port output is set after the clock oscillation has stabilized, causing the device to enter RAM backup mode.

## 17.4 Exiting RAM Backup Mode (Wakeup)

Processing to exit RAM backup mode and return to normal operation is referred to as "wakeup processing." Figure 17.4.1 shows an example of wakeup processing. Wakeup processing is initiated by reset input. The following shows how to execute wakeup processing.

(1) Reset the device ((a) in Figure 17.4.1). For details about reset, refer to Chapter 7, "Reset."

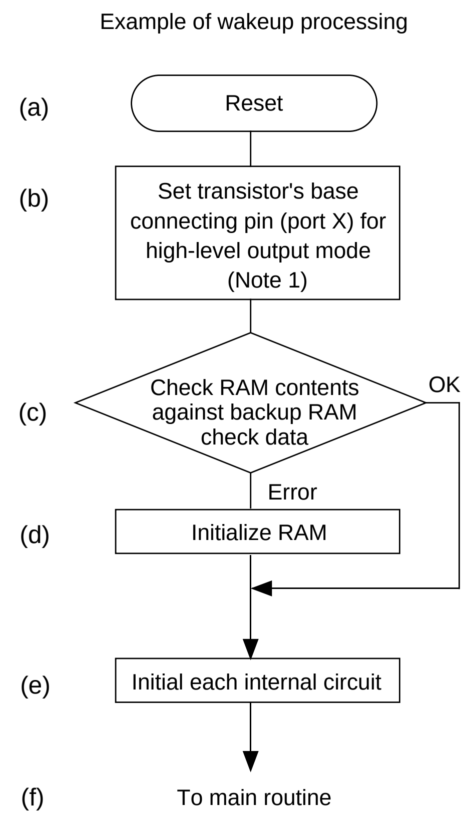
(2) Set port X for output mode and output a high from the port ((b) in Figure 17.4.1). (Note 1)

(3) Check the RAM contents against the check data created before entering RAM backup mode ((c) in Figure 17.4.1).

(4) If the RAM contents and check data did not match when checked in (3), initialize the RAM ((d) in Figure 17.4.1). If the RAM contents and check data matched, use the retained data in the program.

(5) After initializing each internal circuit ((e) in Figure 17.4.1), return the main routine ((f) in Figure 17.4.1).

Note 1: For wakeup from power outage RAM backup mode, settings for port X are unnecessary.


Example of wakeup processing
(a)
Reset
(b)
Set transistor's base connecting pin (port X) for high-level output mode (Note 1)
(c)
Check RAM contents against backup RAM check data
OK
Error
(d)
Initialize RAM
(e)
Initial each internal circuit
(f)
To main routine

Note 1: For wakeup from power outage RAM backup mode, settings for port X are unnecessary.

**Figure 17.4.1 Wakeup Processing**

* This is a blank page.*

# CHAPTER 18
# OSCILLATION CIRCUIT

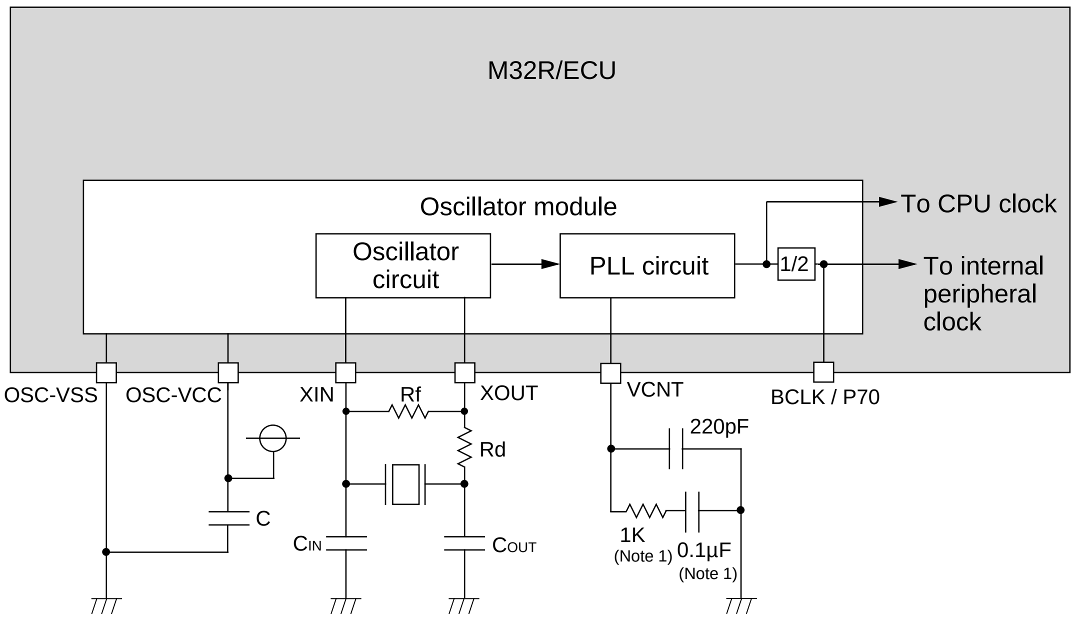
18.1 Oscillator Circuit
18.2 Clock Generator Circuit

## 18.1 Oscillator Circuit

The M32R/ECU contains an oscillator circuit that supplies operating clocks for the CPU core, internal peripheral I/O, and internal memory. The frequency fed to the clock input pin (XIN) is multiplied by 4 by the internal PLL circuit to produce the CPU clock, which is the operating clock for the CPU core and internal memory. The frequency of this clock is divided by 2 in the subsequent circuit to produce the internal peripheral clock, which is the operating clock for the internal peripheral I/O.

### 18.1.1 Example of an Oscillator Circuit

A clock generating circuit can be configured by connecting a ceramic (or crystal) resonator between the XIN and XOUT pins external to the chip. Figure 18.1.1 below shows an example of a system clock generating circuit using a resonator connected external to the chip and an RC network connected to the PLL circuit control pin (VCNT). For constants Rf, CIN, COUT, and Rd, consult your resonator manufacturer to determine the appropriate values.
When you use an externally sourced clock signal without using the internal oscillator circuit, connect the external clock signal to the XIN pin and leave the XOUT pin open.



OSCVCC : 3.3 V power supply
Note 1: allowable error ±10%

**Figure 18.1.1 Example of a System Clock Generating Circuit**

### 18.1.2 System Clock Output Function

A clock whose frequency is twice the input frequency can be output from the BCLK pin. The BCLK pin is shared with port P70. When you use this pin to output the system clock, set the P7 Operation Mode Register (P7MOD)'s D8 bit to 1. Configuration of the P7 Operation Mode Register is shown below.

■ **P7 Operation Mode Register (P7MOD)** <Address: H'0080 0747>

| D8 | 9 | 10 | 11 | 12 | 13 | 14 | D15 |
|---|---|---|---|---|---|---|---|
| P70MOD | P71MOD | P72MOD | P73MOD | P74MOD | P75MOD | P76MOD | P77MOD |

<When reset : H'00>

| D | Bit Name | Function | R | W |
|---|---|---|---|---|
| 8 | P70MOD<br>(Port P70 operation mode) | 0 : P70<br>1 : BCLK | ○ | ○ |
| 9 | P71MOD<br>(Port P71 operation mode) | 0 : P71<br>1 : $\overline{\text{WAIT}}$ | ○ | ○ |
| 10 | P72MOD<br>(Port P72 operation mode) | 0 : P72<br>1 : $\overline{\text{HREQ}}$ | ○ | ○ |
| 11 | P73MOD<br>(Port P73 operation mode) | 0 : P73<br>1 : $\overline{\text{HACK}}$ | ○ | ○ |
| 12 | P74MOD<br>(Port P74 operation mode) | 0 : P74<br>1 : RTDTXD | ○ | ○ |
| 13 | P75MOD<br>(Port P75 operation mode) | 0 : P75<br>1 : RTDRXD | ○ | ○ |
| 14 | P76MOD<br>(Port P76 operation mode) | 0 : P76<br>1 : RTDACK | ○ | ○ |
| 15 | P77MOD<br>(Port P77 operation mode) | 0 : P77<br>1 : RTDCLK | ○ | ○ |

### 18.1.3 Oscillation Stabilization Time at Power-on

The oscillator circuit comprised of a ceramic (or crystal) resonator has a finite time after power-on at which its oscillation is instable. Therefore, create a certain amount of oscillation stabilization time that suits the oscillator circuit used. Figure 18.1.2 shows an oscillation stabilization time at power-on.

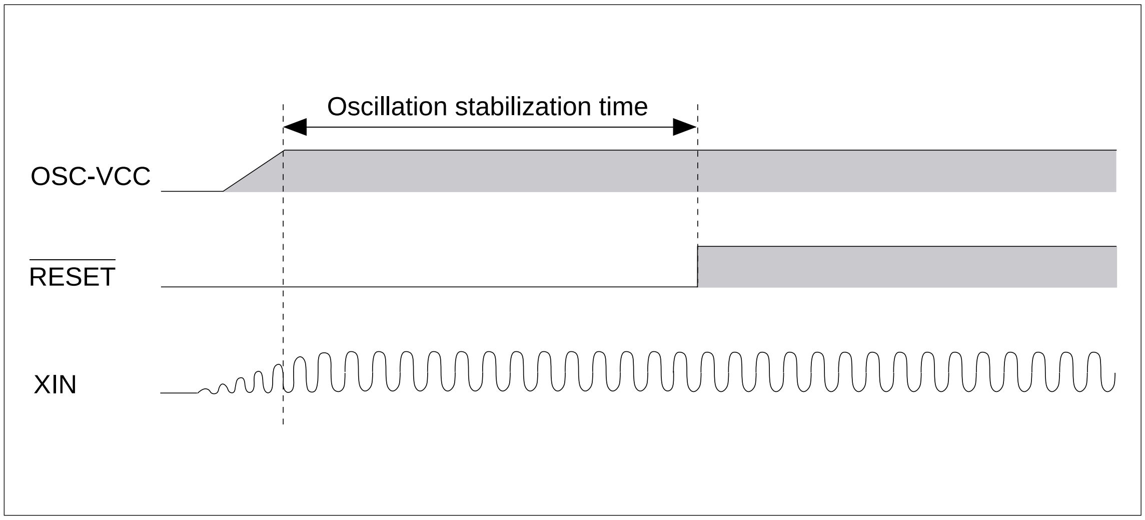


**Figure 18.1.2 Oscillation Stabilization Time at Power-on**

## 18.2 Clock Generator Circuit

The clock generator supplies independent clocks to the CPU and internal peripheral circuits.

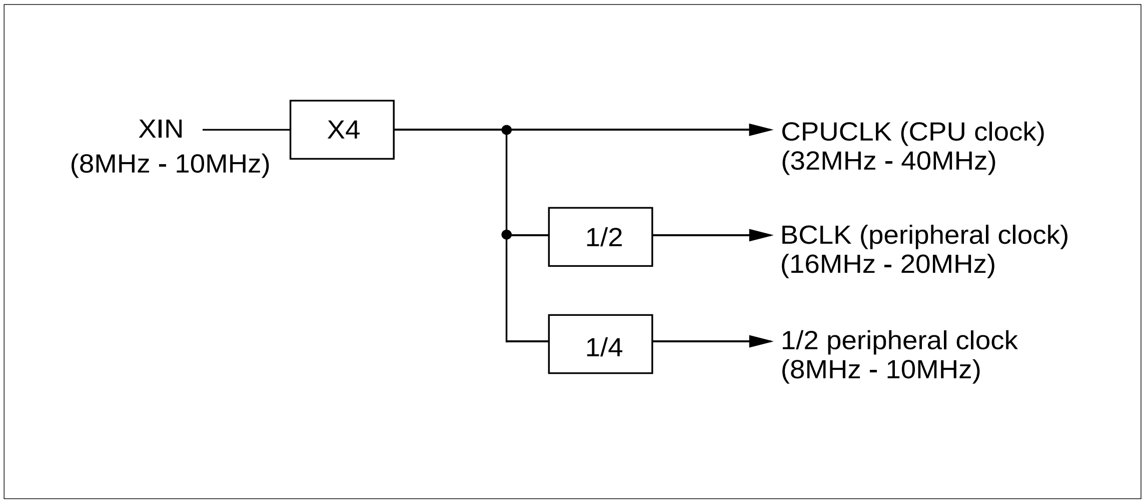


**Figure 18.2.1 Configuration of the Clock Generator Circuit**

* This is a blank page.*

# CHAPTER 19
# JTAG

19.1 Outline of JTAG
19.2 Configuration of the JTAG Circuit
19.3 JTAG Registers
19.4 Basic Operation of JTAG
19.5 Boundary Scan Description Language
19.6 Precautions on Board Design when Using JTAG
19.7 Processing Pins when Not Using JTAG

## 19.1 Outline of JTAG

The 32171 contains a JTAG (Joint Test Action Group) interface based on IEEE Standard Test Access Port and Boundary-Scan Architecture (IEEE Std. 1149.1a-1993). This JTAG interface can be used as an input/output path for boundary-scan test (boundary-scan path). For details about IEEE 1149.1 JTAG test access ports, refer to the IEEE Std. 1149.1a-1993 documentation.

The functions of JTAG interface related pins mounted on the 32171 are shown below.

**Table 19.1.1 JTAG Pin Functions**

| Type | Symbol | Pin Name | I/O | Function |
|---|---|---|---|---|
| TAP (Note1) | JTCK | Test clock | Input | Clock input to the test circuit. |
| | JTDI | Test data input | input | Synchronous serial data input pin used to enter test instruction code and test data. This input is sampled on rising edges of JTCK. |
| | JTDO | Test data output | output | Synchronous serial data output pin used to output test instruction code and test data. This signal changes state on falling edges of JTCK, and is output only in Shift-IR or Shift-DR state. |
| | JTMS | Test mode select | Input | Test mode select input to control the test circuit's state transitions. This input is sampled on rising edges of JTCK. |
| | JTRST | Test reset | Input | Active-low test reset input to initialize the test circuit asynchronously. To ensure that the test circuit is reset without fail, JTMS signal input must be held high while this signal changes state from low to high. |

Note 1: TAP = Test Access Port, a JTAG interface stipulated in IEEE 1149.1.

## 19.2 Configuration of the JTAG Circuit

The 32171's JTAG circuit consists of the following blocks:

- Instruction register to hold instruction codes which are fetched through the boundary-scan path
- A set of data registers which are accessed through the boundary-scan path
- Test access port (abbreviated TAP) controller to control the JTAG unit's state transitions
- Control logic to select input, output, etc.

A configuration of the JTAG circuit is shown below.

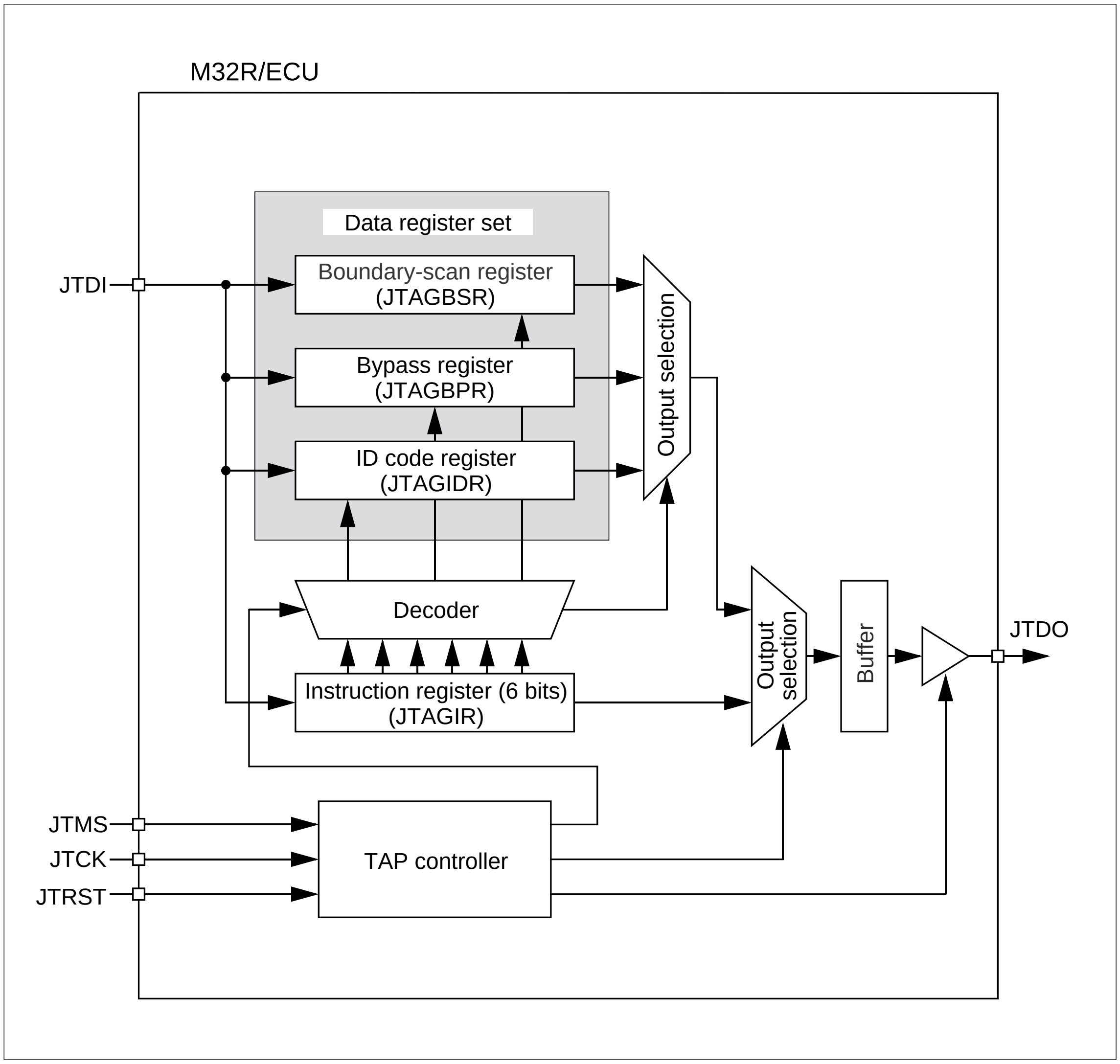

M32R/ECU
Data register set
JTDI
Boundary-scan register (JTAGBSR)
Bypass register (JTAGBPR)
ID code register (JTAGIDR)
Output selection
Decoder
Instruction register (6 bits) (JTAGIR)
Output selection
Buffer
JTDO
JTMS
JTCK
JTRST
TAP controller

**Figure 19.2.1 Configuration of the JTAG Circuit**

## 19.3 JTAG Registers

### 19.3.1 Instruction Register (JTAGIR)

The Instruction Register (JTAGIR) is a 6-bit register to hold instruction code. This register is set in IR path sequence. The instructions set in this register determine the data register to be selected in the subsequent DR path sequence.

When test is reset (to initialize the test circuit), the initial value of this register is b'000010 (IDCODE instruction). After a test reset, the IDCODE Register is selected as the data register until an instruction code is set by an external device. In "Capture-IR" state, this register always has b'110001 (fixed value) loaded into it. Therefore, when in "Shift-IR" state, no matter what value was set in this register, b'110001 is always output from the JTDO pin (sequentially beginning with LSB). However, this value normally is not handled as instruction code.

Shown below is outside the scope of guaranteed operations. Note that if this operation is performed, the device may inadvertently handle b'110001 as instruction code, which makes it unable to operate normally.

[Capture-IR] → [Exit1-IR] → [Update-IR]

The 32171's JTAG interface supports the following instructions:

- Three instructions stipulated as essential in IEEE 1149.1 (EXTEST, SAMPLE/PRELOAD, BYPASS)
- Device ID register access instruction (IDCODE)

**Table 19.3.1 JTAG Instruction List**

| Instruction Code | Abbreviation | Operation |
|---|---|---|
| b'000000 | EXTEST | Tests circuit/board-level connections outside the chip. |
| b'000001 | SAMPLE/PRELOAD | Samples operating circuit status and outputs the sampled status from JTDO pin, while at the same time entering the data used for boundary-scan test from the JTDI pin and presets it in Boundary Scan Register. |
| b'000010 | IDCODE | Selects ID Code Register and outputs device and manufacturer identification data from JTDO pin. |
| b'111111 | BYPASS | Selects Bypass Register and inspects or sets data. |

Notes: • Do not set any other instruction code.

• For details about "IR path sequence," "DR path sequence," "Test reset," "Capture-IR" state, "Shift-IR" state, "Exit1-IR" state, and "Update-IR" state, refer to Section 19.4.

### 19.3.2 Data Registers

#### (1) Boundary Scan Register (JTAGBSR)

The Boundary Scan Register is a 471-bit register used to perform boundary-scan test. Bits in this register are assigned to each pin on the 32171.

Connected between the JTDI and JTDO pins, this register is selected when issuing EXTEST or SAMPLE/PRELOAD instruction. In "Capture-DR" state, this register captures the status of input pins or internal logic output values. In "Shift-DR" state, while outputting the sampled value, it is used to set pin functions (input/output pin and tristate output pin direction) and output values by entering data for boundary-scan test.

#### (2) Bypass Register (JTAGBPR)

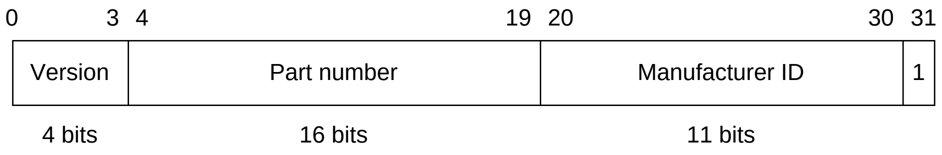
The Bypass Register is a 1-bit register used to bypass boundary-scan passes when the 32171 is not the target of boundary-scan test. Connected between the JTDI and JTDO pins, this register is selected when issuing BYPASS instruction. This register when in "Capture-DR" state has b'0 (fixed value) loaded into it.

#### (3) ID Code Register (JTAGIDR)

The ID Code Register is a 32-bit register used to identify the device and manufacturer. It holds the following information:

| | | |
|---|---|---|
| • Version information (4 bits) | : | b'0000 |
| • Part number (16 bits) | : | b'0011 0010 0010 0000 |
| • Manufacturer ID (11 bits) | : | b'000 0001 1100 |

This register is connected between the JTDI and JTDO pins, and is selected when issuing IDCODE instruction. When in "Capture-DR" state, this register has the said IDCODE data loaded into it, which is output from the JTDO pin in "Shift_DR" state.
This register is a read-only register, so that the data written from the JTDI pin during DR pass sequence is ignored. Therefore, make sure JTDI input = low during "Shift-DR" state.



Note: • For details about "Capture-DR" and "Shift-DR" states, refer to Section 19.4.

## 19.4 Basic Operation of JTAG

### 19.4.1 Outline of JTAG Operation

The instruction and data registers basically are accessed in the following three operations, which are performed based on state transitions of the TAP controller. The TAP controller changes state according to JTMS input, and generates control signals required for operation in each state.

- **Capture operation**

  The result of boundary-scan test or the fixed data defined for each register is sampled. As register operation, the input data is loaded into the shift register stage.

- **Shift operation**

  The register is accessed from outside through the boundary-scan path. The sampled value is output to an external device at the same time data is set from outside. As register operation, bits are shifted right between each shift register stage.

- **Update operation**

  The data set from outside during shift is driven. As register operation, the value set in the shift register stage is transferred to the parallel output stage.

The JTAG interface undergoes transitions of internal state depending on JTMS input as it performs the following two operations. In either case, the operation basically is performed in order of Capture → Shift → Update.

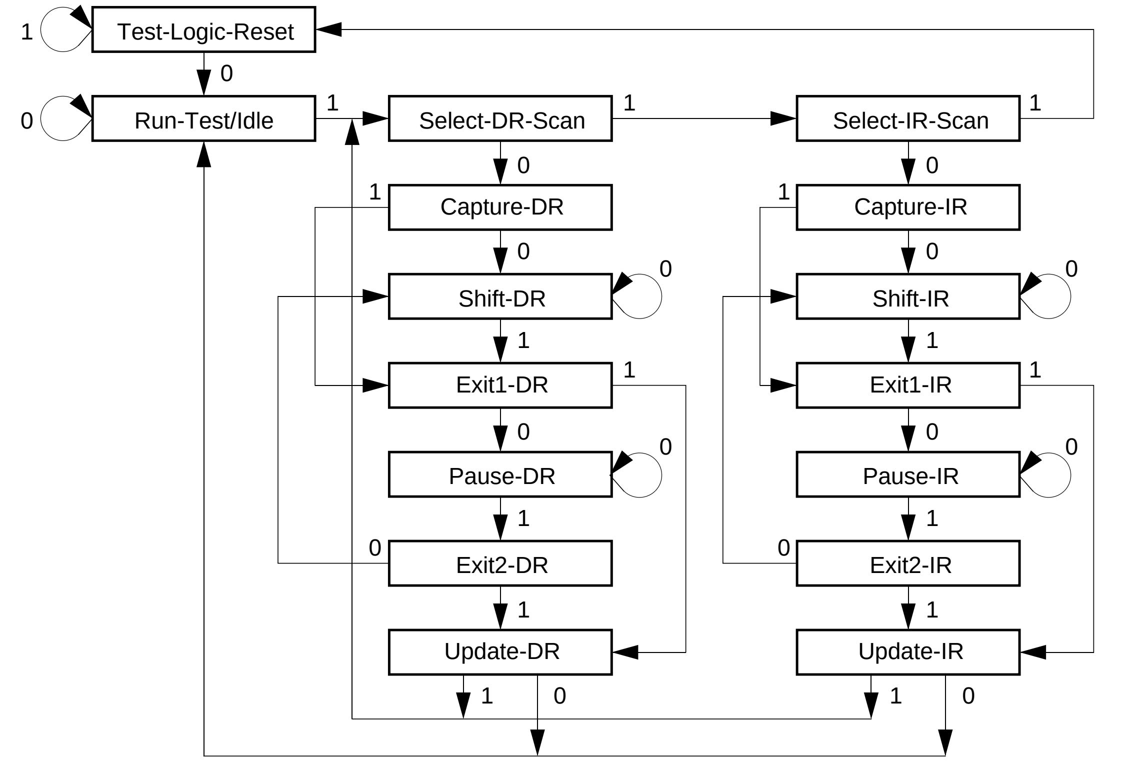
- **IR path sequence**

  Instruction code is set in the instruction register to select the data register to be operated on in the subsequent DR path sequence.

- **DR path sequence**

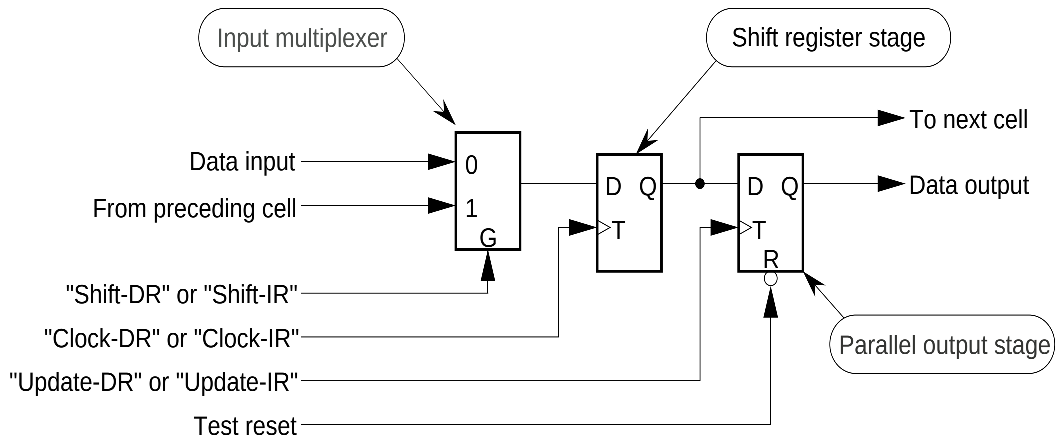
  The selected data register is operated on to inspect or set data.

The state transitions of the TAP controller and the basic configuration of the 32171's JTAG related registers are shown below.



Note: • Values (0 and 1) in this diagram denote the state of JTMS input signal.

**Figure 19.4.1 TAP Controller State Transition**



Note: • Shown here is the basic configuration, and the configuration of DR and IR does not all have to be like this.

**Figure 19.4.2 Basic Configuration of JTAG Related Registers**

### 19.4.2 IR Path Sequence

Instruction code is set in the Instruction Register (JTAGIR) to select the data register to be accessed in the subsequent DR path sequence. The IR path sequence is performed following the procedure described below.

(1) Enter JTMS = high for a period of two JTCK cycles from "Run-Test/Idle" state to go to "Select-IR-Scan" state.

(2) Set JTMS = low to go to "Capture-IR" state. At this time, b'110001 (fixed value) is set in the instruction register's shift register stage.

(3) Subsequently, enter JTMS = low to go to "Shift-IR" state. In "Shift-IR" state, the value of the shift register stage is shifted right one bit every cycle, and the data b'110001 (fixed value) that was set in (2) is serially output from the JTDO pin. At the same time, the instruction code serially entered from the JTDI pin is set in the shift register stage bit by bit. Because instruction code is set in the instruction register which is comprised of 6 bits, the "Shift-IR" state continues for a period of 6 JTCK cycles. To stop the shift operation in the middle, go to "Pause-IR" state via temporarily "Exit1-IR" state (by setting JTMS input from high to low). Also, to return from "Pause-IR" state, go to "Shift-IR" state via temporarily "Exit2-IR" state (by setting JTMS input from high to low).

(4) By setting JTMS = high, go from "Shift-IR" state to "Exit1-IR" state. This completes the shift operation.

(5) Subsequently, enter JTMS = high to go to "Update-IR" state. In "Update-IR" state, the instruction code that was set in the instruction register's shift register stage is transferred to the instruction register's parallel output stage and, thus, JTAG instruction decoding begins.

(6) Subsequently, enter JTMS = high to go to "Select-DR-Scan" state or JTMS = low to go to "Run-Test/Idle" state.

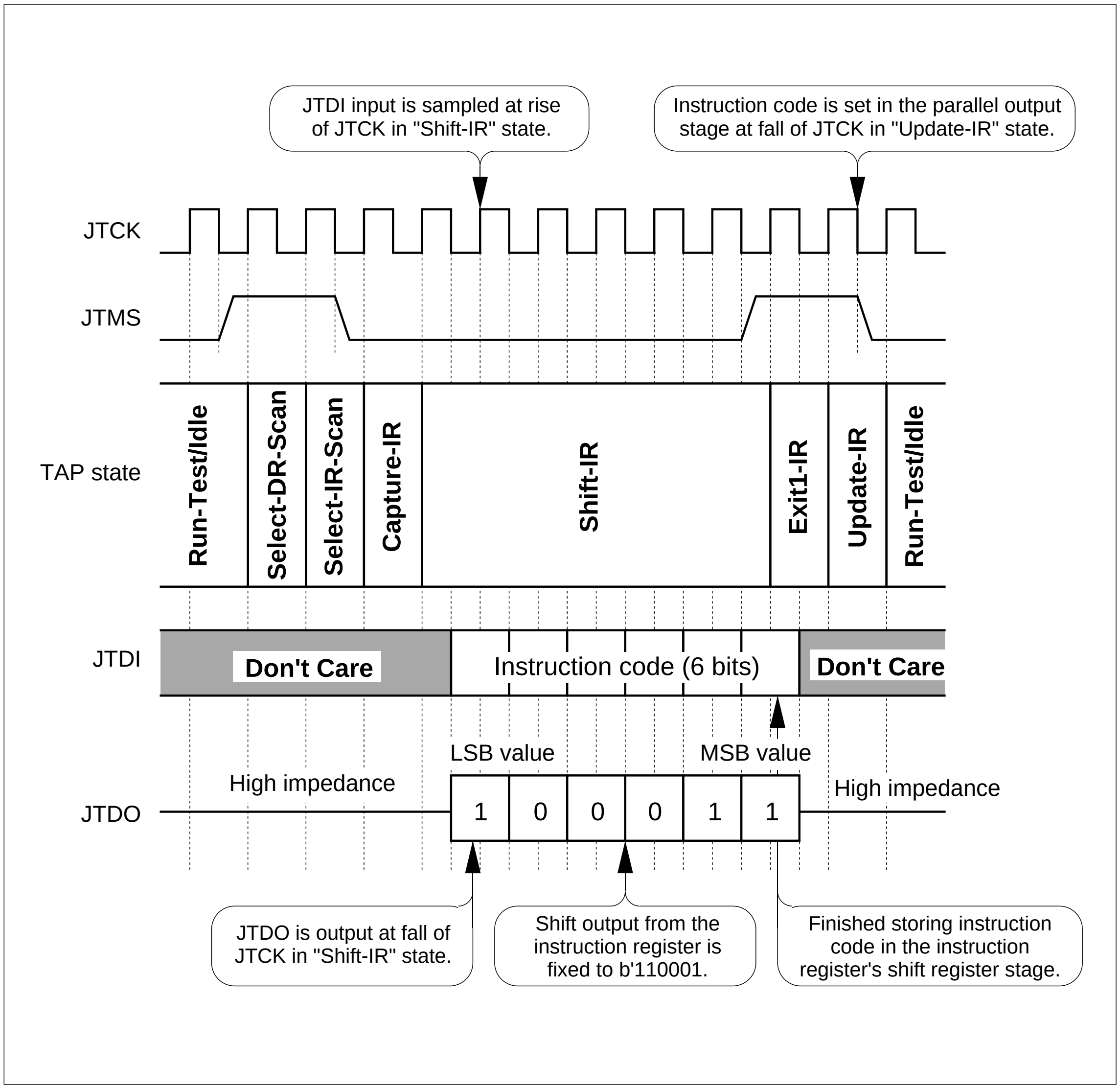

JTDI input is sampled at rise of JTCK in "Shift-IR" state.
Instruction code is set in the parallel output stage at fall of JTCK in "Update-IR" state.
JTCK
JTMS
TAP state
Run-Test/Idle
Select-DR-Scan
Select-IR-Scan
Capture-IR
Shift-IR
Exit1-IR
Update-IR
Run-Test/Idle
JTDI
Don't Care
Instruction code (6 bits)
Don't Care
LSB value
MSB value
High impedance
High impedance
JTDO
1
0
0
0
1
1
JTDO is output at fall of JTCK in "Shift-IR" state.
Shift output from the instruction register is fixed to b'110001.
Finished storing instruction code in the instruction register's shift register stage.

**Figure 19.4.3 IR Path Sequence**

### 19.4.3 DR Path Sequence

The data register that was selected during the IR path sequence prior to the DR path sequence is operated on to inspect or set data in it. The DR path sequence is performed following the procedure described below.

(1) Enter JTMS = high for a period of one JTCK cycle from "Run-Test/Idle" state to go to "Select-DR-Scan" state. Which data register will be selected at this time depends on the instruction that was set during the IR path sequence performed prior to the DR path sequence.

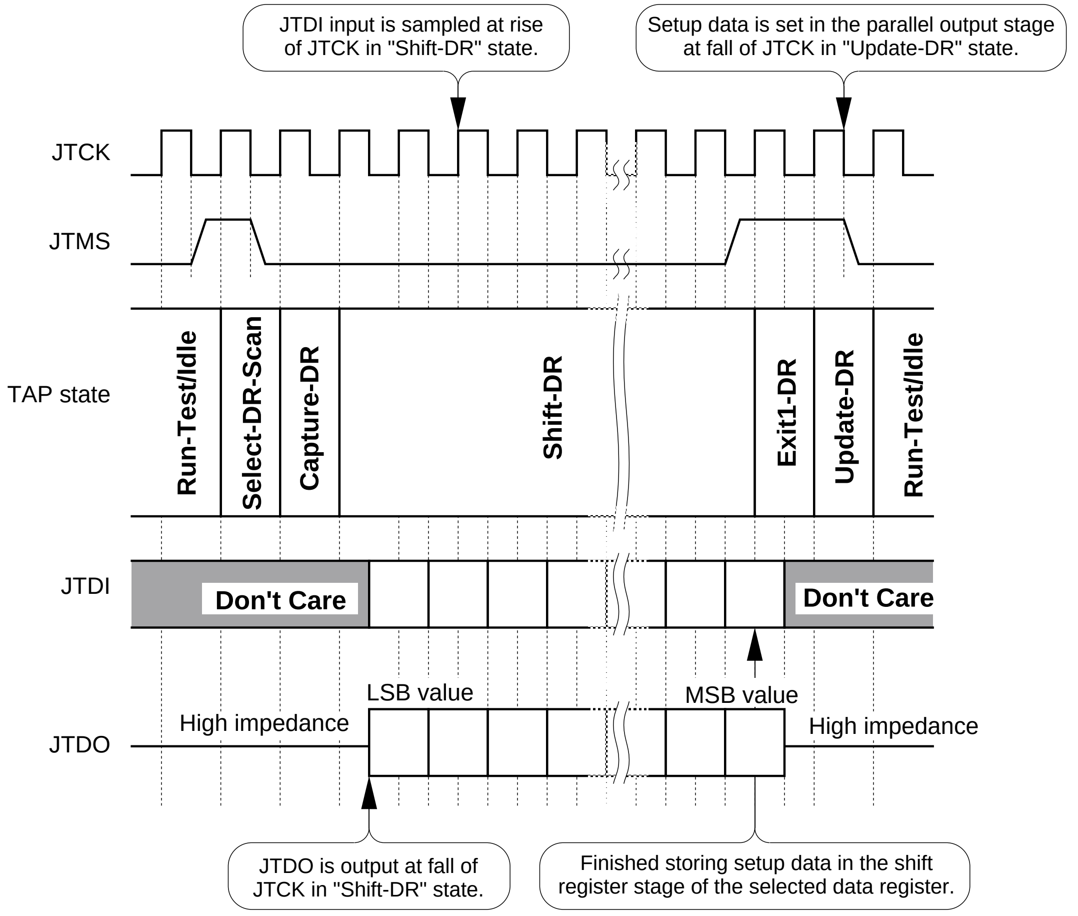
(2) Set JTMS = low to go to "Capture-DR" state. At this time, the result of boundary-scan test or the fixed data defined for each register is set in the data register's shift register stage.

(3) Subsequently, enter JTMS = low to go to "Shift-DR" state. In "Shift-DR" state, the DR value is shifted right one bit every cycle, and the data that was set in (2) is serially output from the JTDO pin. At the same time, the setup data serially entered from the JTDI pin is set in the data register's shift register stage bit by bit. By continuing the "Shift-DR" state as long as the number of bits of the selected data register (by entering JTMS = low), all bits of data can be set in and read out from the shift register stage. To stop the shift operation in the middle, go to "Pause-DR" state via temporarily "Exit1-DR" state (by setting JTMS input from high to low). Also, to return from "Pause-DR" state, go to "Shift-DR" state via temporarily "Exit2-DR" state (by setting JTMS input from high to low).

(4) Set JTMS = high to go from "Shift-DR" state to "Exit1-DR" state. This completes the shift operation.

(5) Subsequently, enter JTMS = high to go to "Update-DR" state. In "Update-DR" state, the data that was set in the data register's shift register stage is transferred to the parallel output stage and, thus, the setup data becomes ready for use.

(6) Subsequently, enter JTMS = high to go to "Select-DR-Scan" state or JTMS = low to go to "Run-Test/Idle" state.


JTDI input is sampled at rise of JTCK in "Shift-DR" state.
Setup data is set in the parallel output stage at fall of JTCK in "Update-DR" state.
JTCK
JTMS
TAP state
Run-Test/Idle
Select-DR-Scan
Capture-DR
Shift-DR
Exit1-DR
Update-DR
Run-Test/Idle
JTDI
Don't Care
Don't Care
LSB value
MSB value
High impedance
High impedance
JTDO
JTDO is output at fall of JTCK in "Shift-DR" state.
Finished storing setup data in the shift register stage of the selected data register.

Note: • The shift operation of the data register for the shift register stage is right-shifted, therefore, the output from JTDO is from the LSB side. Input to JTDI starts from the value to be set in LSB side.

**Figure 19.4.4 DR Path Sequence**

### 19.4.4 Examining and Setting Data Registers

To inspect or set the data register, follow the procedure described below.

(1) To access the test access port (JTAG) for the first time, enter test reset (to initialize the test circuit). Test reset can be entered by one of the following two methods:

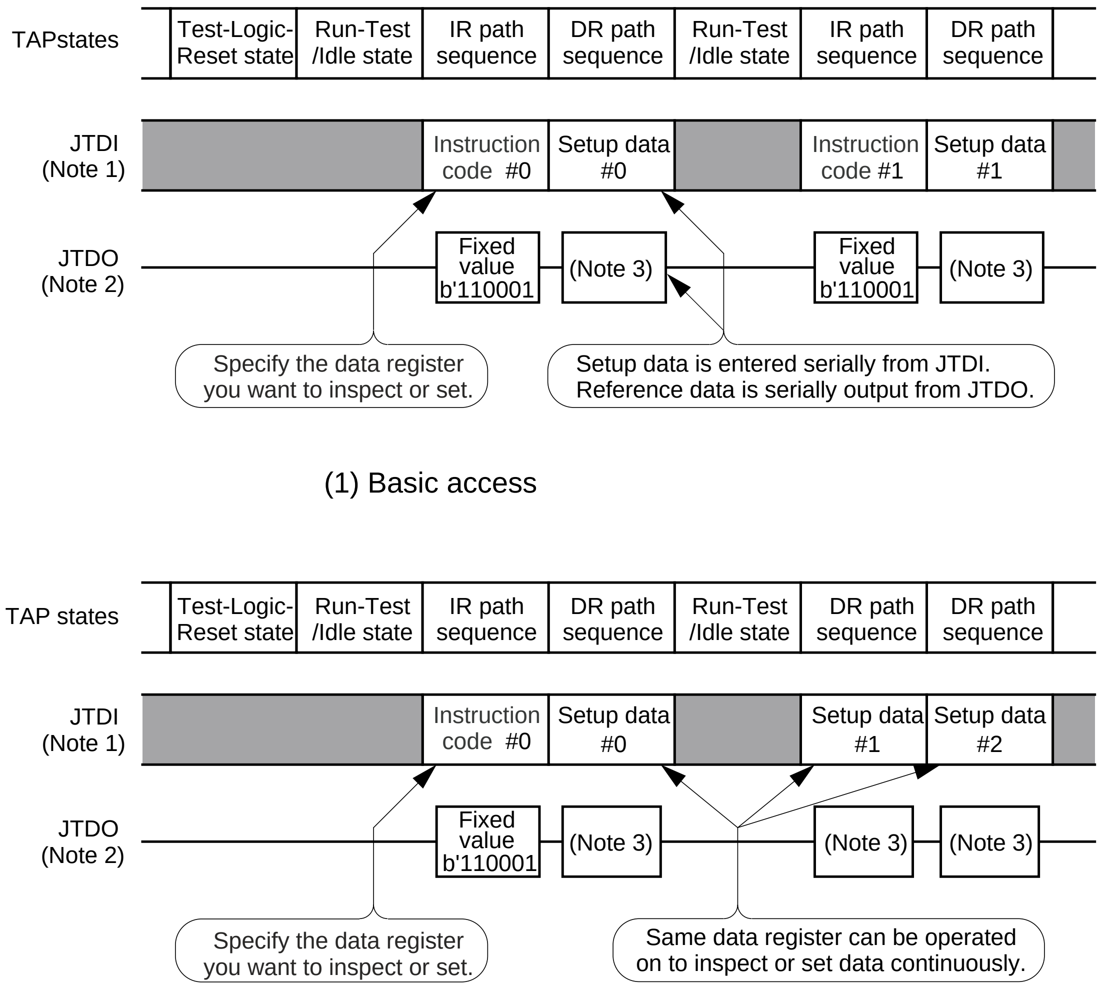
- Pull JTRST pin input low
- Drive JTMS pin input high and enter JTCK for 5 cycles or more

(2) Set JTMS = low to go to "Run-Test/Idle" state. To continue the idle state, hold JTMS input low.

(3) Set JTMS = high to exit "Run-Test/Idle" state and perform IR path sequence. In IR path sequence, specify the data register you want to inspect or set.

(4) Subsequently, perform DR path sequence. For the data register specified in IR path sequence, enter setup data from the JTDI pin and read out reference data from the JTDO pin.

(5) If you want to proceed and perform IR path sequence or DR path sequence after DR path sequence is completed, enter JTMS = high to return to "Select-DR-Scan" state. If you want to wait for the next processing after a series of IR and DR path sequence processing is completed, enter JTMS = low to go to "Run-Test/Idle" state and retain the state.



(1) Basic access

(2) Continuous access to the same data register

Note 1 : The setup value for each register must be entered from the JTDI pin beginning with the LSB.

Note 2 : The value of each register is output from the JTDO pin beginning with the LSB. The JTDO pin outputs valid data in only "Shift-IR" state of IR path sequence and "Shift-DR" state of DR path sequence. In all other states, the JTDO pin is tristated (high impedance).

Note 3 : Data can only be read out from the data register which is selected by the instruction that was set in the immediately preceding IR path sequence. Output in the selected data register's shift register stage is the value that was sampled during "Capture-DR" state.

**Figure 19.4.5 Continuous JTAG Access**

## 19.5 Boundary Scan Description Language

The Boundary Scan Description Language (abbreviated BSDL) is stipulated in supplements to "Standard Test Access Port and Boundary-Scan Architecture" of IEEE 1149.1-1990 and IEEE 1149.1a-1993. BSDL is a subset of IEEE 1076-1993 Standard VHSIC Hardware Description Language (VHDL). BSDL helps to precisely describe the functions of standard-compliant components to be tested. For package connection test, this language is used by Automated Test Pattern Generation tools, and for synthesized test logic and verification, it is used by Electronic Design Automation tools. BSDL provides powerful extended functions usable in internal test generation and necessary to write hardware debug and diagnostics software.

The primary section of BSDL contains statements of logical port description, physical pin map, instruction set, and boundary register description.

- **Logical port description**

  The logical port description assigns meaningful symbol names to each pin on the chip. This determines the logic type of input, output, input/output, buffer, or link of each pin that defines the logical direction of signal flow.

- **Physical pin map**

  The physical pin map correlates the chip's logical ports to the physical pins on each package. Use of separate names for each map makes it possible to define multiple physical pin maps in one BSDL description.

- **Instruction set statement**

  The instruction set statement writes bit patterns to be shifted in into the chip's instruction register. This bit pattern is necessary to place the chip into each test mode defined in standards. It is also possible to write instructions exclusive to the chip.

- **Boundary register description**

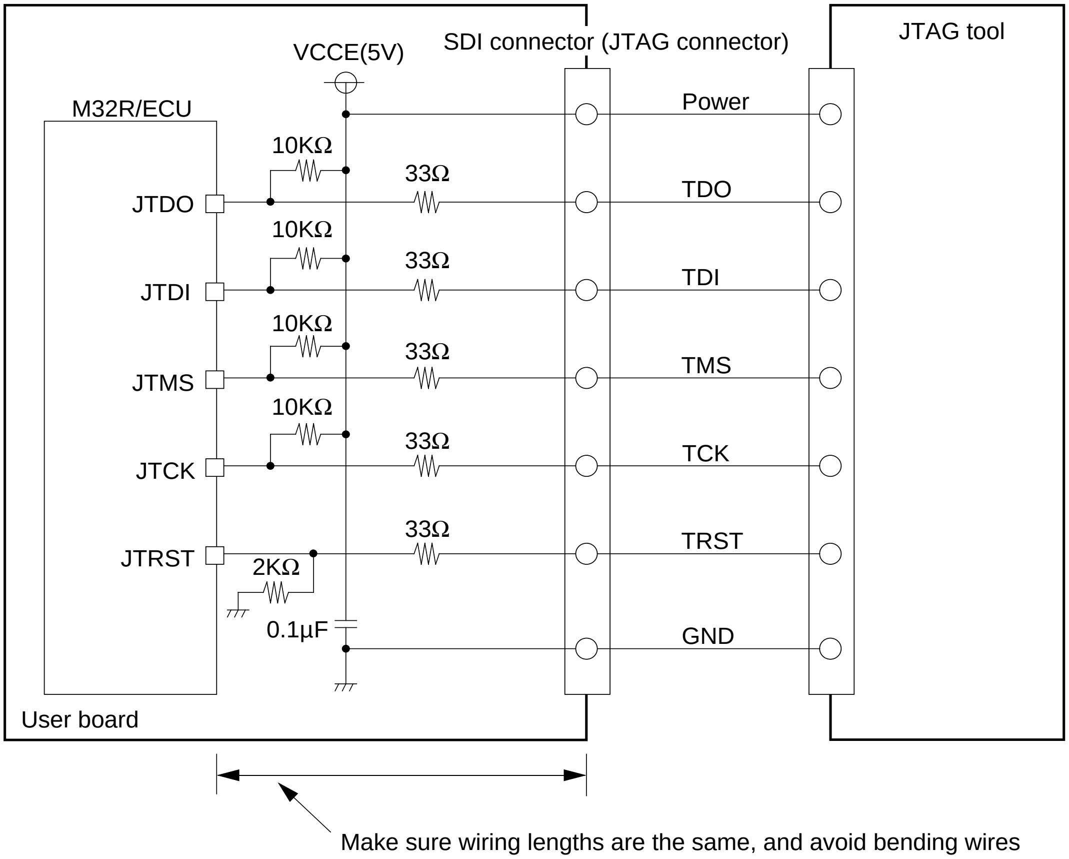
  The boundary register description is a list of boundary register cells or shift stages. Each cell is assigned a separate number. The cell with number 0 is located closest to the test data output (JTDO) pin, and the cell with the largest number is located closest to the test data input (JTDI) pin. Cells also contain related other information which includes cell type, logical port corresponding to cell, logical function of cell, safety value, control cell number, disable value, and result value.

Note: • Information on the Boundary Scan Description Language (BSDL) can be downloaded from the M32R family application engineering data in “Renesas Home Page.”
The URL address of this home page is shown below.

• **http: //www.renesas.com/**

## 19.6 Precautions on Board Design when Using JTAG

The JTAG pins require that wiring lengths be matched during board design in order to accomplish fast, highly reliable communication with JTAG tools.
An example of how to process pins when using JTAG tools is shown below.



Make sure wiring lengths are the same, and avoid bending wires as much as possible. Also, do not use through-holes within wiring.

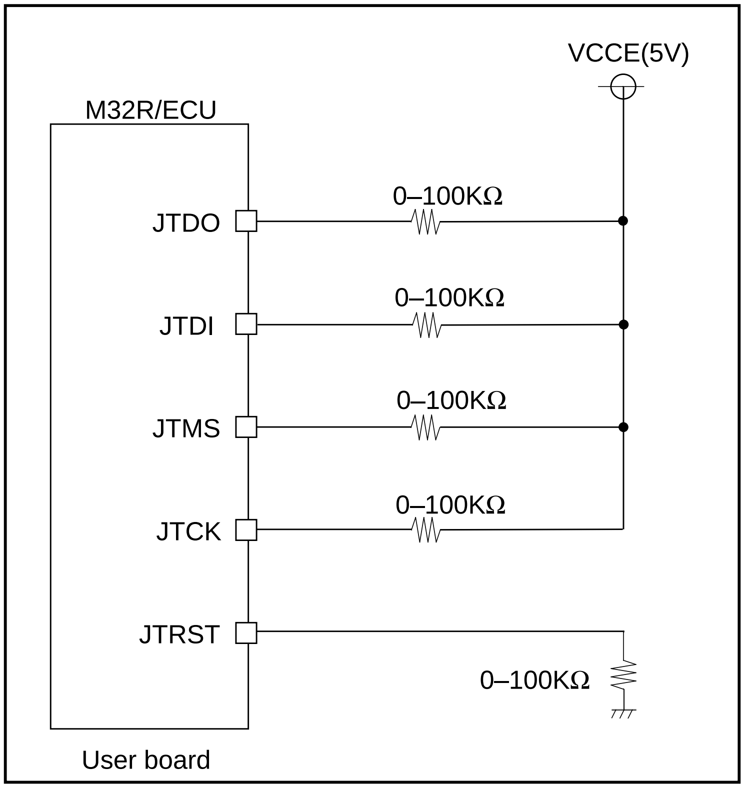
Notes: •Only if the JTRST pin is firmly tied to ground, it dosn’t matter whether the JTDO, JTDI, JTMS, and JTCK pins are pulled high or pulled low.

- Even when not using JTAG tools, always be sure to process each pin. The same pulldown/pullup resistance values as when using JTAG tools may be used without causing any problem.

**Figure 19.6.1 Example for Processing Pins when Using JTAG Tools**

## 19.7 Processing Pins when Not Using JTAG

The diagram below shows how to process JTAG pins when not using these pins (i.e. for boards that do not have pins/connectors connecting to JTAG tools).



Note: • Only if the JTRST pin is firmly tied to ground, it dosn’t matter whether the JTDO, JTDI,JTMS, and JTCK pins are pulled high or pulled low.

**Figure 19.7.1 Processing Pins when Not Using JTAG**

CHAPTER 20

# POWER-ON/POWER-OFF SEQUENCE

20.1 Configuration of the Power Supply Circuit
20.2 Power-on Sequence
20.3 Power-off Sequence

## 20.1 Configuration of the Power Supply Circuit

To allow for high-speed operation with low power consumption, the M32/ECU is designed in such a way that the external interface circuits operate with a 5 V or 3.3 V external I/O power supply, while all other circuits operate with the 3.3 V internal power supply.
This requires that control timing of both 5 V and 3.3 V power supplies be considered when designing your circuit.

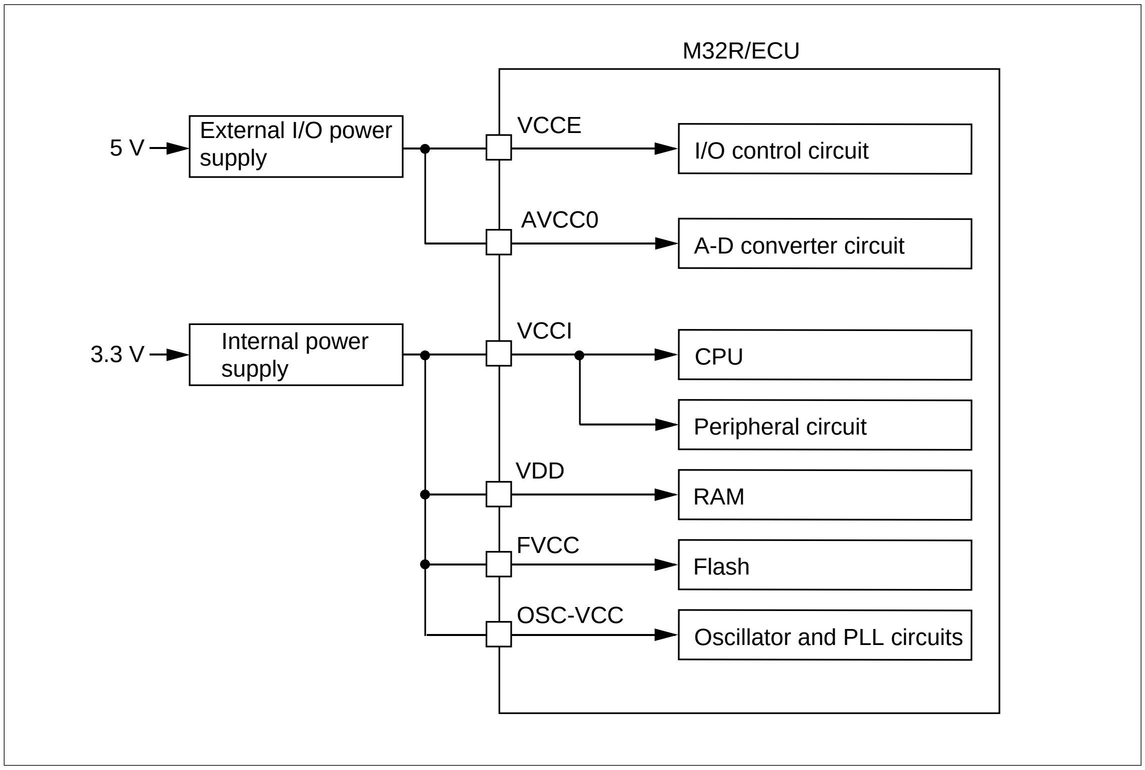


**Figure 20.1.1 Configuration of the Power Supply Circuit (when external I/O power supply = 5V )**

**Table 20.1.1 List of Power Supply Functions**

| Type of Power Supply | Pin Name | Function |
|---|---|---|
| External I/O<br>Power Supply | VCCE | Supplies power to external I/O ports |
| | AVCC0 | Power supply for A-D converter |
| | VREF0 | Reference voltage for A-D converter |
| Internal<br>Power Supply | VCCI | Supplies power to internal logic |
| | FVCC | Power supply for internal flash memory |
| | VDD | Power supply for internal RAM backup |
| | OSC-VCC | Power supply for oscillator and PLL circuits |

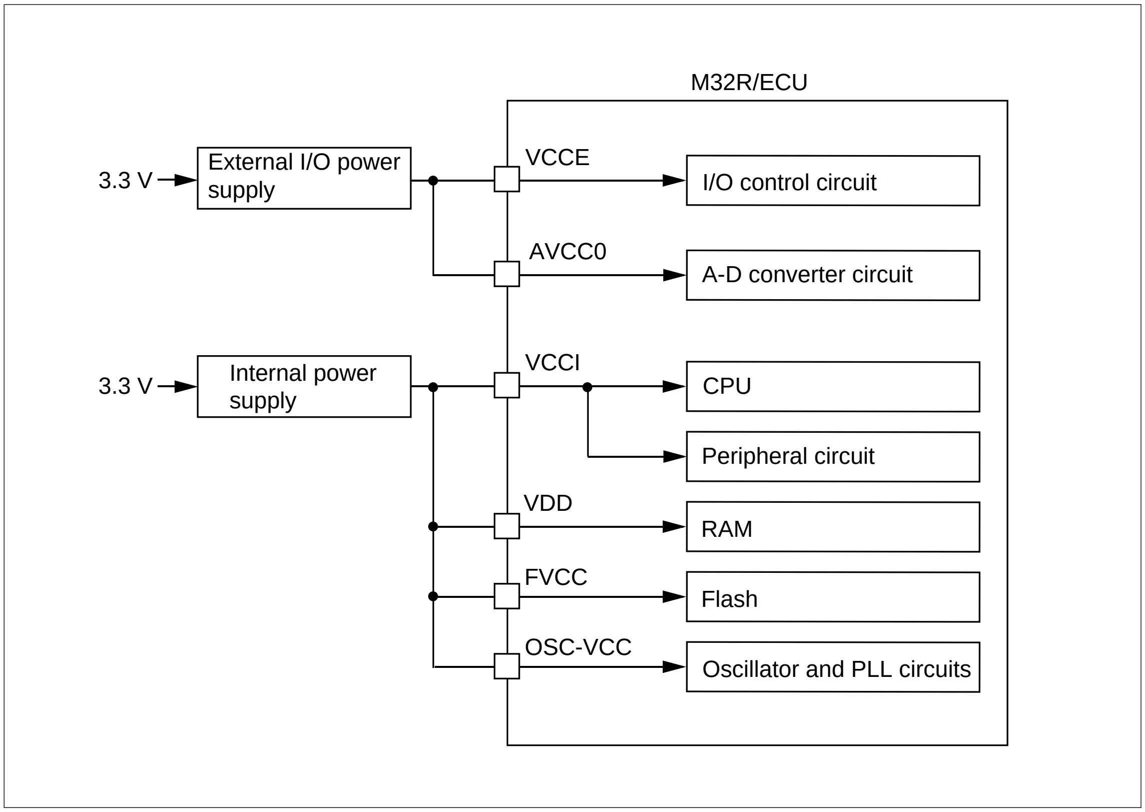
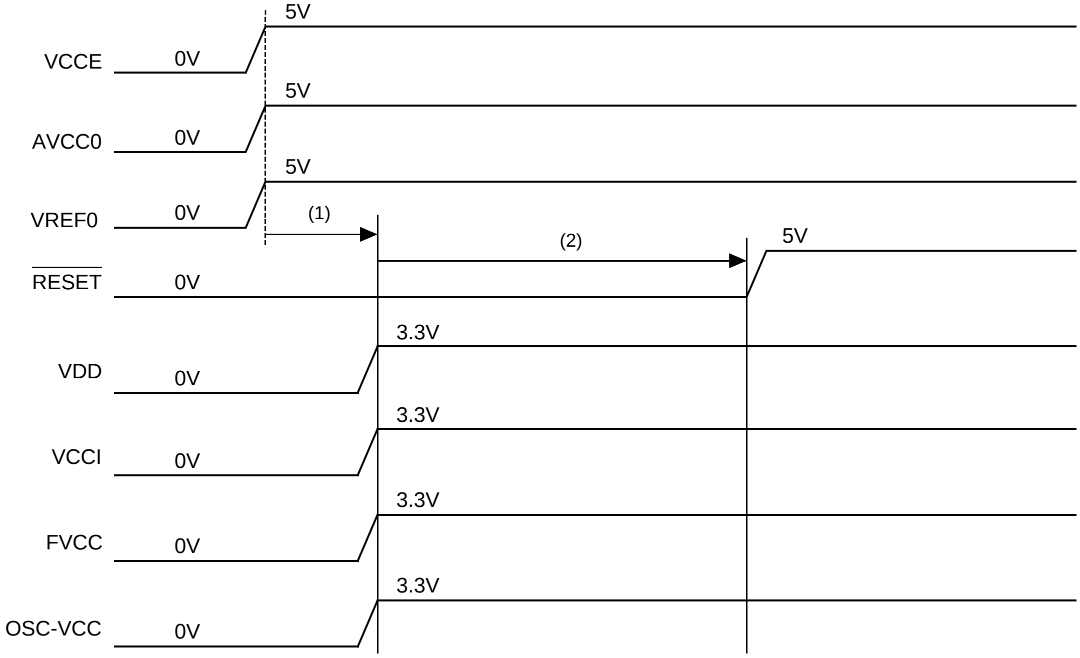


**Figure 20.1.2 Configuration of the Power Supply Circuit (when external I/O power supply = 3.3V )**

## 20.2 Power-On Sequence

### 20.2.1 Power-On Sequence When Not Using RAM Backup

The diagram below shows the M32/ECU's power supply (external I/O and internal) turn-on sequence when not using RAM backup.



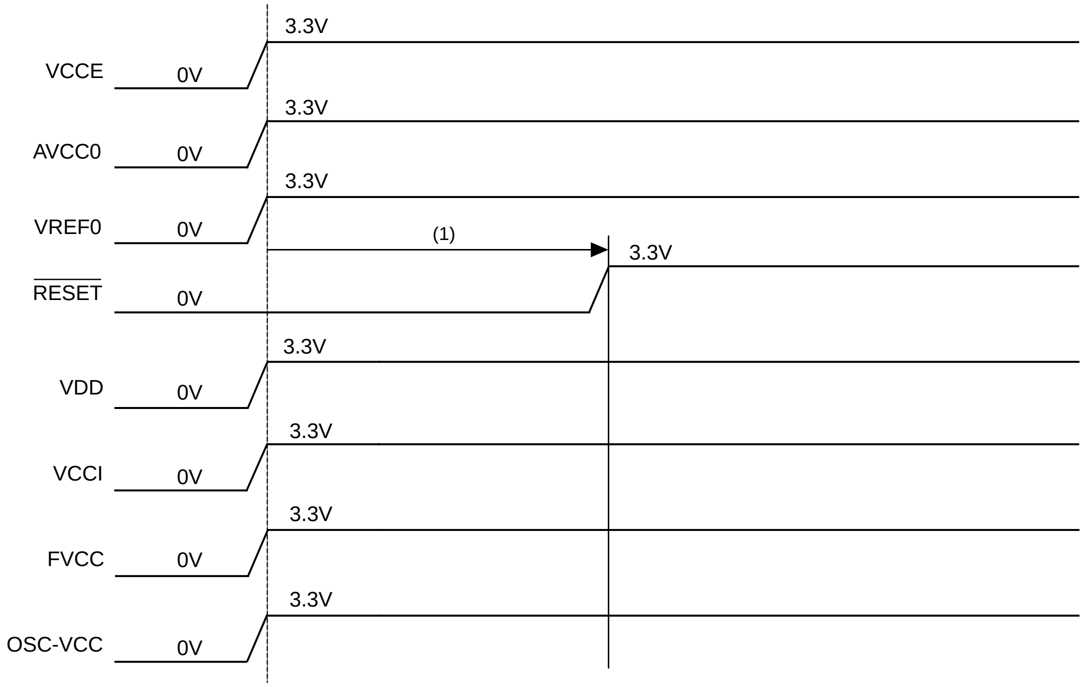
(1): Turn on the external I/O power supply before turning on the internal power supply.

(2): After turning on all power supplies and holding the $\overline{\text{RESET}}$ pin low for an oscillation stabilization time, release the $\overline{\text{RESET}}$ pin input back high (to exit the reset state).

Note: • Power-on limitations
- VDD ≧ OSC-VCC ≧ VCCI ≧ FVCC
- VCCE ≧ VCCI, FVCC, OSC-VCC

Figure 20.2.1 Power-On Sequence When Not Using RAM Backup (when external I/O power supply = 5V )

Note: • Providing the difference in voltage levels is within a range (about 0.1–0.2 V in a transient state) where no current in-flow due to diode characteristics will occur, inversion of phases ay not present a problem. To ensure stable operation, however, make sure the circuit you design satisfies the recommended operating conditions.



(1): After turning on all power supplies and holding the $\overline{\text{RESET}}$ pin low for an oscillation stabilization time, release the $\overline{\text{RESET}}$ pin input back high (to exit the reset state).

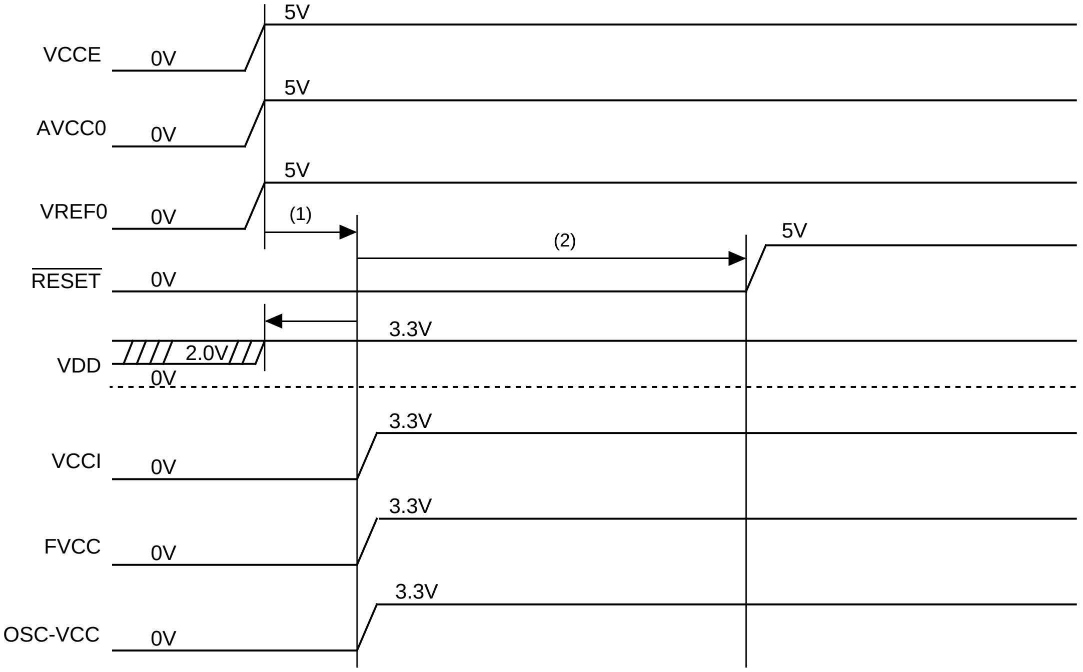
Note: • Power-on limitations

- VDD ≧ OSC-VCC ≧ VCCI ≧ FVCC
- VCCE ≧ VCCI, FVCC, OSC-VCC

**Figure 20.2.2 Power-On Sequence When Not Using RAM Backup (when external I/O power supply = 3.3V )**

### 20.2.2 Power-On Sequence When Using RAM Backup

The diagram below shows a power-on sequence(external I/O and internal power supply) of the M32R/ECU when using RAM backup.



(1): Turn on the internal power supply after turning on the external I/O power supply.

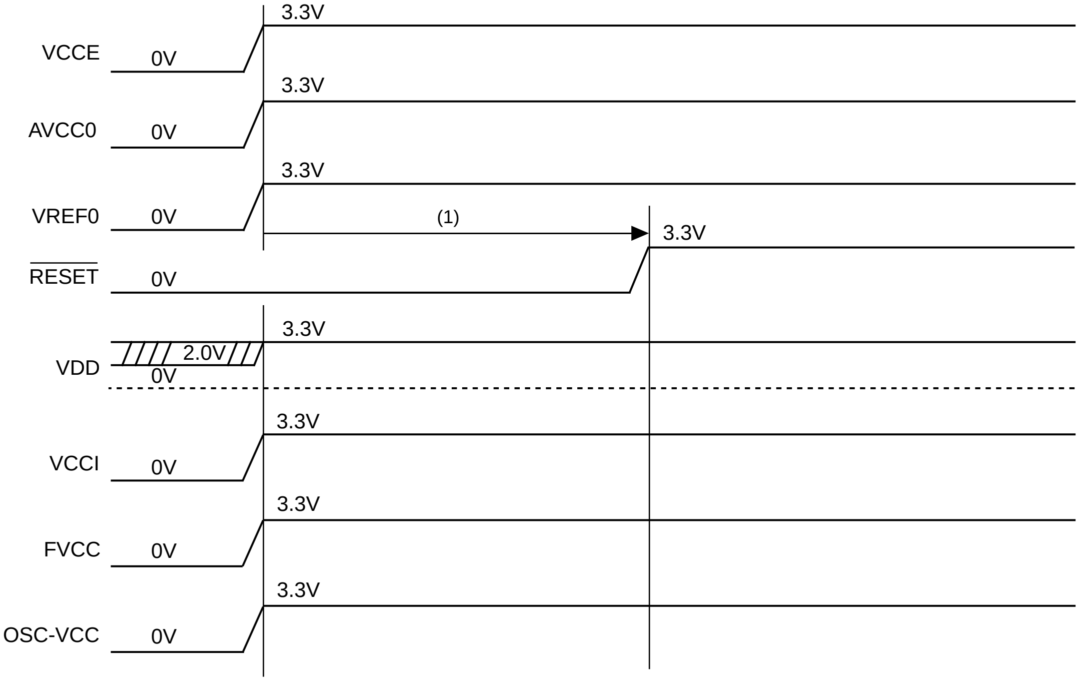
(2): After turning on all power supplies and holding the $\overline{\text{RESET}}$ pin low for an oscillation stabilization time, release the $\overline{\text{RESET}}$ pin input back high (to exit the reset state).

Note: • Power-on limitations

- VDD ≧ OSC-VCC ≧ VCCI ≧ FVCC
- VCCE ≧ VCCI, FVCC, OSC-VCC

**Figure 20.2.3 Power-On Sequence When Using RAM Backup(when external I/O power supply = 5 V )**

Note: • Providing the difference in voltage levels is within a range (about 0.1–0.2 V in a transient state) where no current in-flow due to diode characteristics will occur, inversion of phases may not present a problem. To ensure stable operation, however, make sure the circuit you design satisfies the recommended operating conditions.



(1): After turning on all power supplies and holding the $\overline{\text{RESET}}$ pin low for an oscillation stabilization time, release the $\overline{\text{RESET}}$ pin input back high (to exit the reset state).

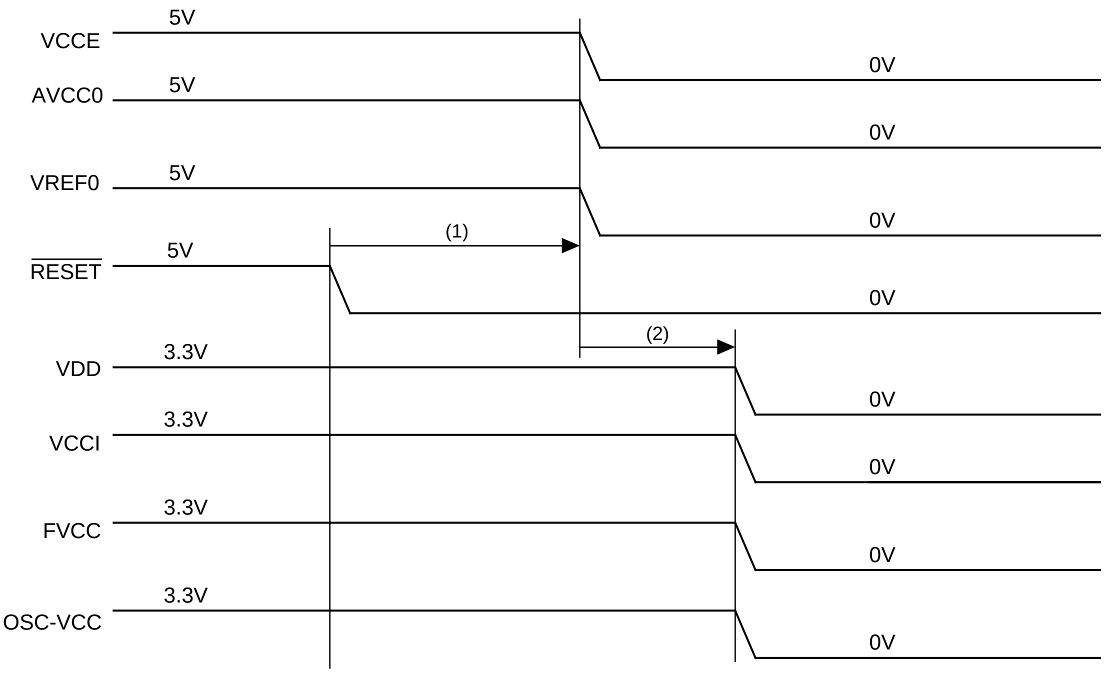
Note: • Power-on limitations
- VDD ≧ OSC-VCC ≧ VCCI ≧ FVCC
- VCCE ≧ VCCI, FVCC, OSC-VCC

**Figure 20.2.4 Power-On Sequence When Using RAM Backup(when external I/O power supply = 3.3 V )**

## 20.3 Power-off Sequence

### 20.3.1 Power-off Sequence When Not Using RAM Backup

The diagram below shows a power-off sequence (external I/O and internal power supply) of the M32R/ECU when not using RAM backup.

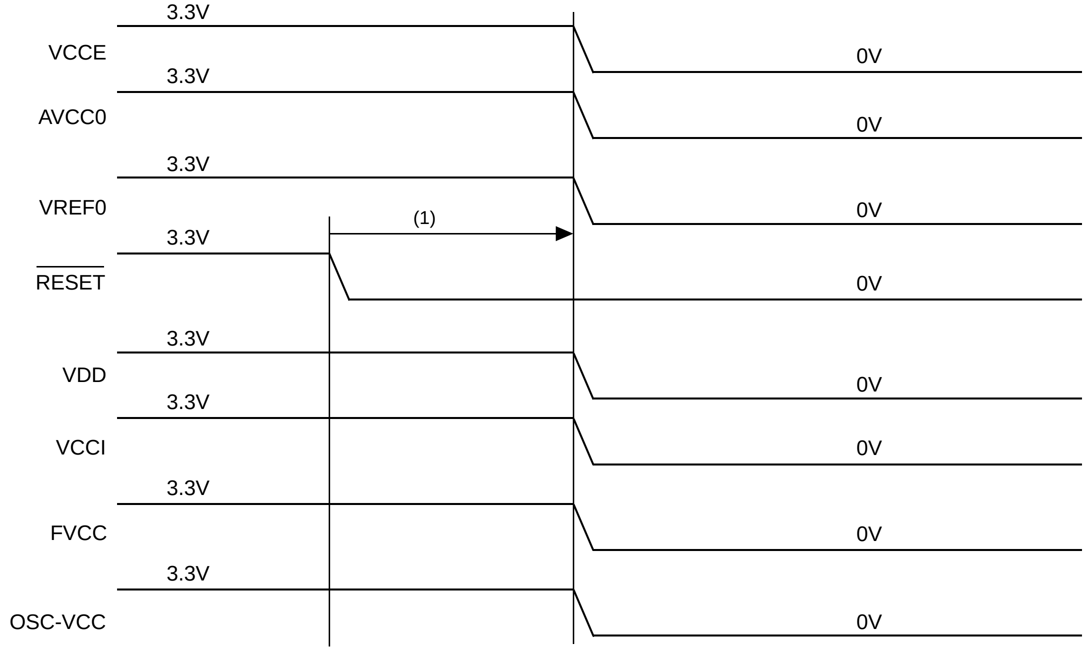


(1): Pull the $\overline{\text{RESET}}$ pin input low.
(2): Turn off the external I/O and the internal power supply after the $\overline{\text{RESET}}$ pin goes low.

Note: • Power-off requirements
- VDD ≧ VCCI ≧ FVCC
- OSC-VCC ≧ VCCI

**Figure 20.3.1 Power-off Sequence When Not Using RAM Backup(when external I/O power supply = 5 V )**

Note: • Providing the difference in voltage levels is within a range (about 0.1–0.2 V in a transient state) where no current in-flow due to diode characteristics will occur, inversion of phases may not present a problem. To ensure stable operation, however, make sure the circuit you design satisfies the recommended operating conditions.



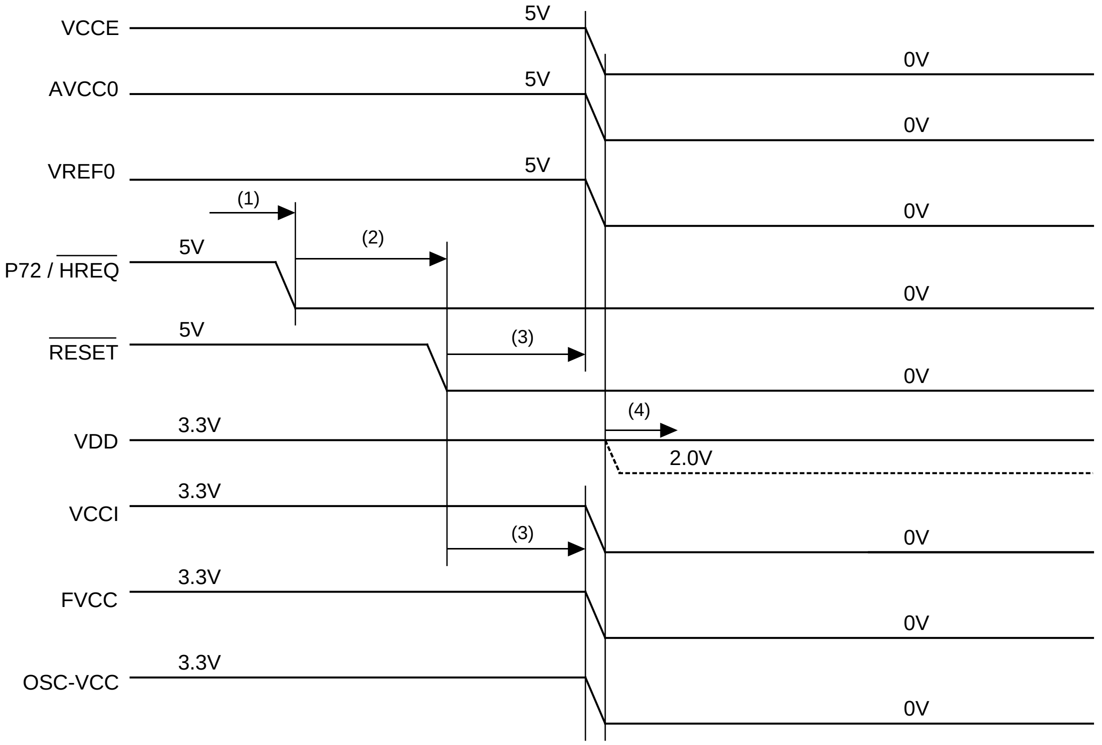
(1): Turn off all power supplies after the $\overline{\text{RESET}}$ pin goes low.

Note: • Power-off requirements
- VDD ≧ VCCI ≧ FVCC
- OSC-VCC ≧ VCCI

**Figure 20.3.2 Power-off Sequence When Not Using RAM Backup(when external I/O power supply = 3.3 V )**

### 20.3.2 Power-off Sequence When Using RAM Backup

The diagram below shows a power-off sequence (external I/O and internal power supply) of the M32R/ECU when using RAM backup.



(1): Pull the HREQ pin input low to halt the CPU at end of bus cycle. Or disable RAM access in software. The M32R/ECU allows P72 to be used as HREQ irrespective of its operation mode.

(2): With the CPU halted, pull the RESET pin input low. Or while RAM access is disabled, pull the RESET pin input low.

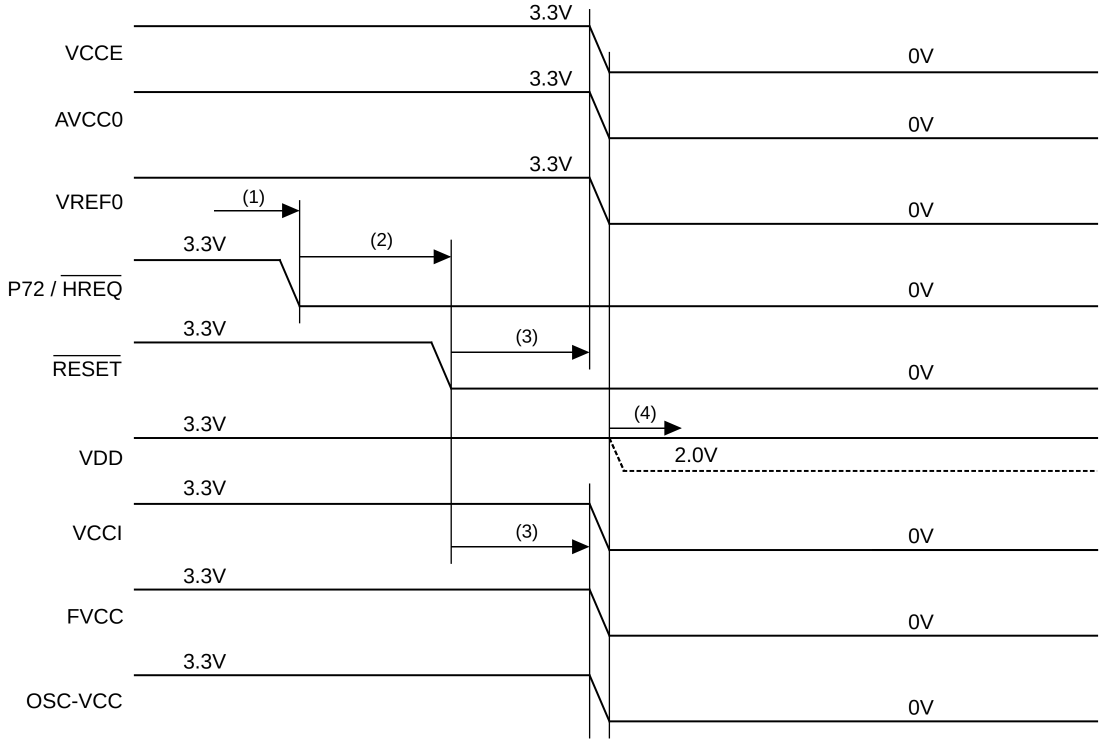
(3): Turn off the external I/O and the internal power supply after the RESET pin goes low.

(4): Reduce the VDD voltage from 3.3 V to 2.0 V as necessary.

Note: • Power-off requirements
- VDD ≧ VCCI ≧ FVCC
- OSC-VCC ≧ VCCI

**Figure 20.3.3 Power-off Sequence When Using RAM Backup(when external I/O power supply = 5 V)**

Note: • Providing the difference in voltage levels is within a range (about 0.1–0.2 V in a transient state) where no current in-flow due to diode characteristics will occur, inversion of phases may not present a problem. To ensure stable operation, however, make sure the circuit you design satisfies the recommended operating conditions.



(1): Pull the HREQ pin input low to halt the CPU at end of bus cycle. Or disable RAM access in software. The M32R/ECU allows P72 to be used as HREQ irrespective of its operation mode.

(2): With the CPU halted, pull the RESET pin input low. Or while RAM access is disabled, pull the RESET pin input low.

(3): Turn off all power supply after the RESET pin goes low.

(4): Reduce the VDD voltage from 3.3 V to 2.0 V as necessary.

Note: • Power-off requirements
- VDD ≧ VCCI ≧ FVCC
- OSC-VCC ≧ VCCI

**Figure 20.3.4 Power-off Sequence When Using RAM Backup(when external I/O power supply = 3.3 V)**

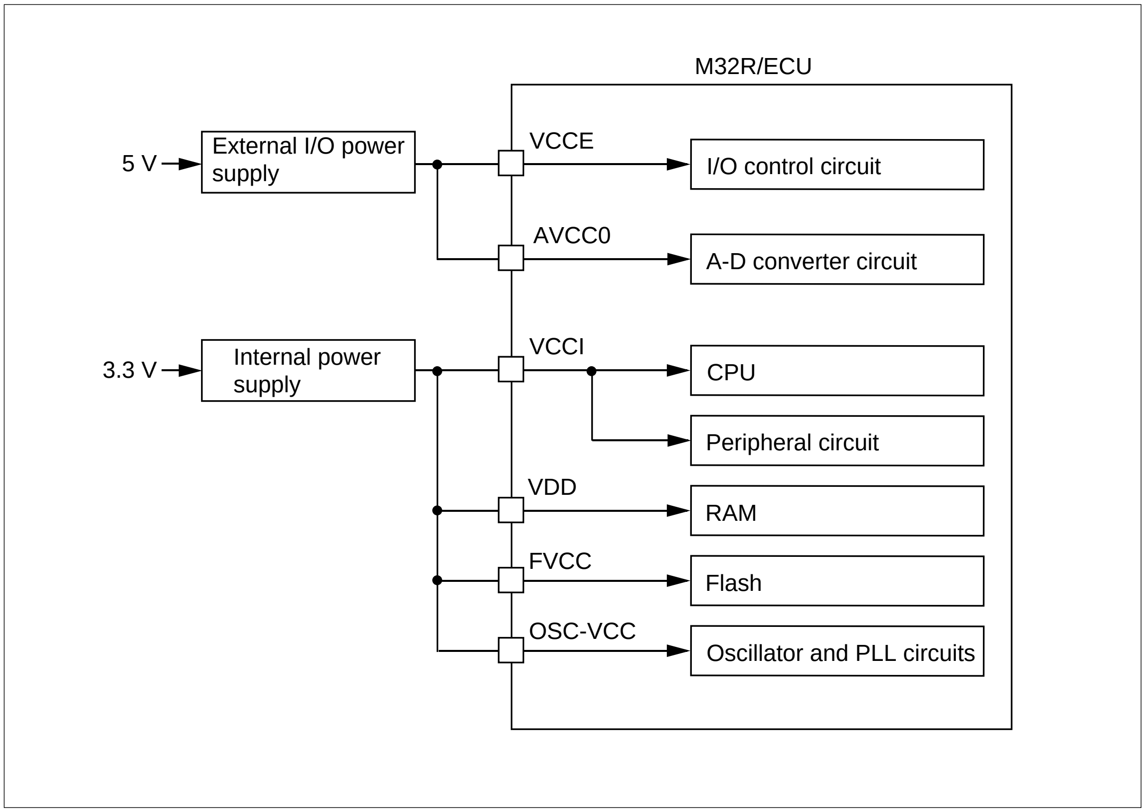


Figure 20.3.5 Microcomputer Ready to Run State 1

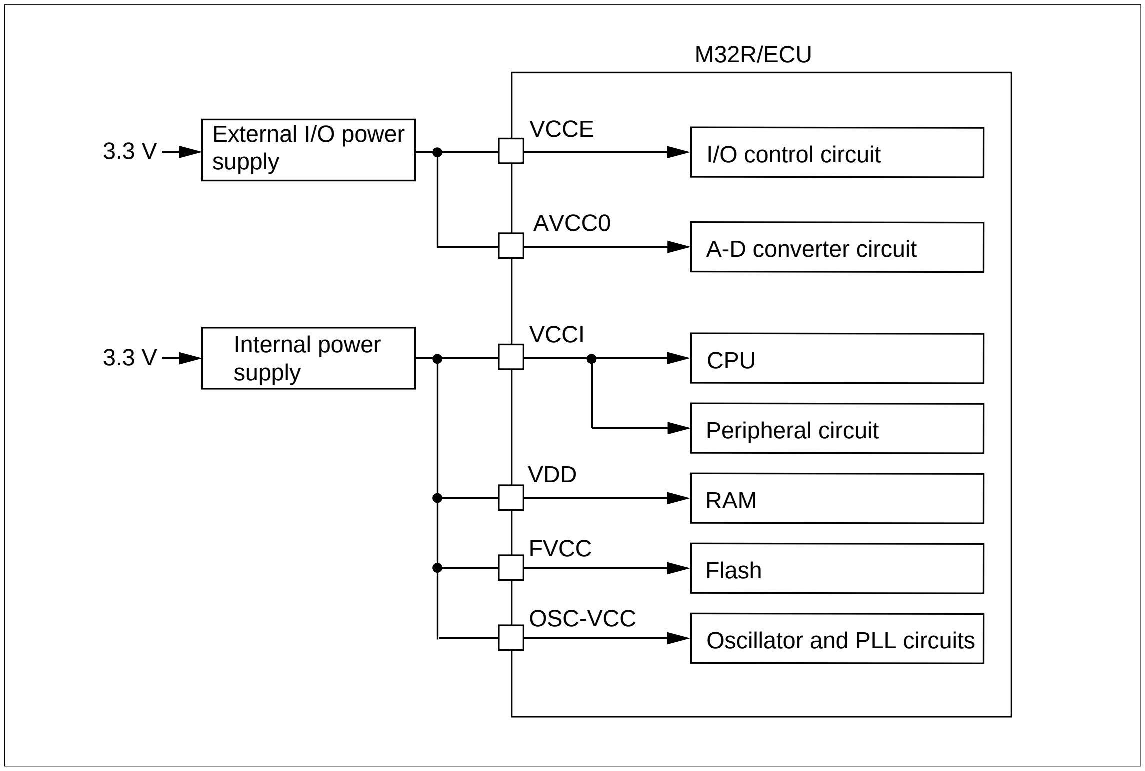


Figure 20.3.6 Microcomputer Ready to Run State 2

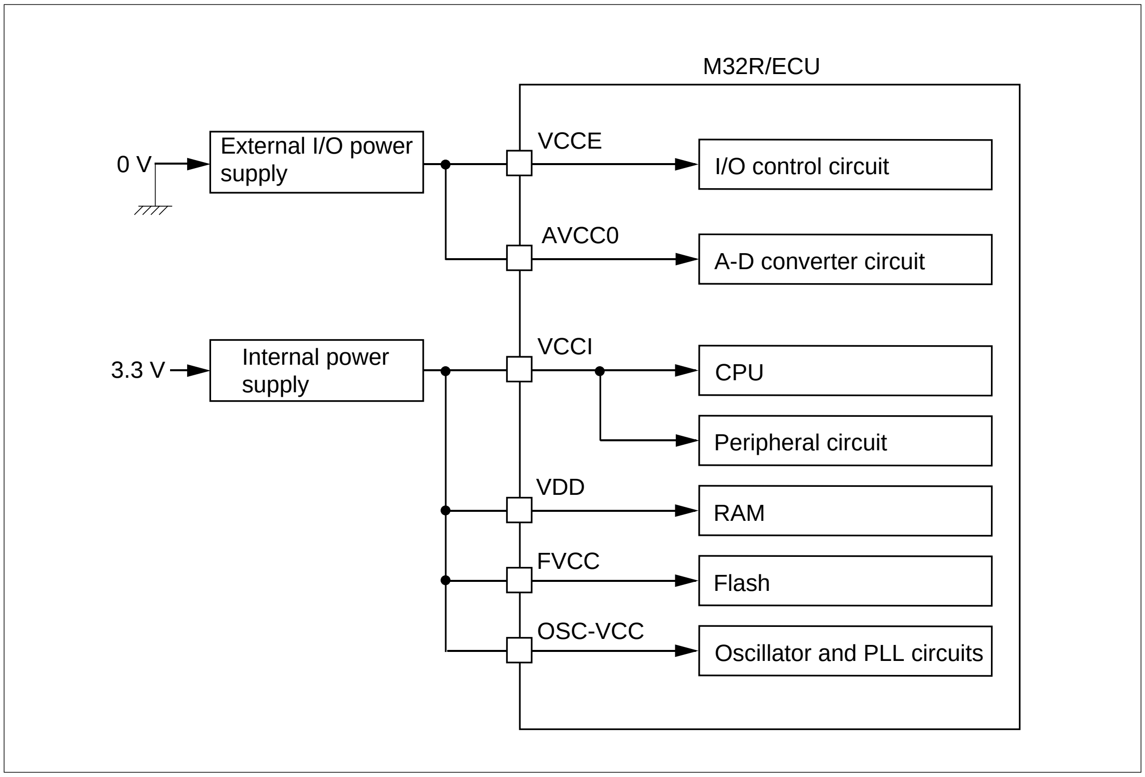


Figure 20.3.7 CPU Reset State

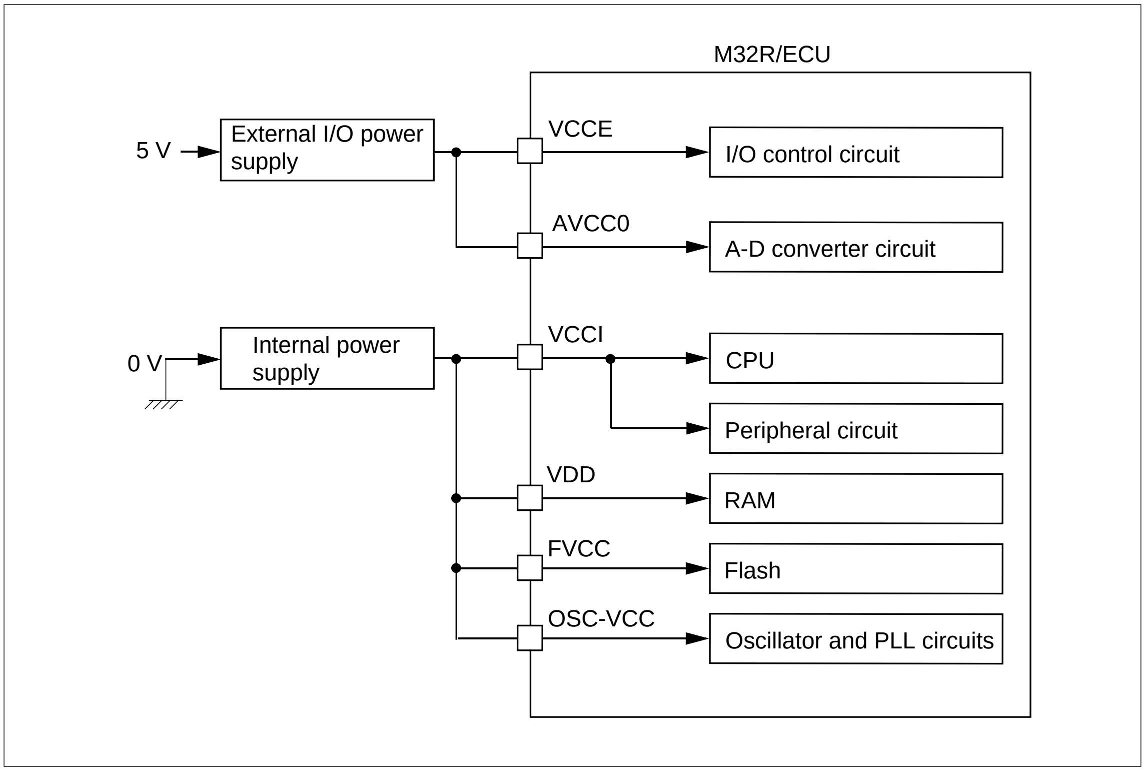


**Figure 20.3.8 CPU Stop State 1**

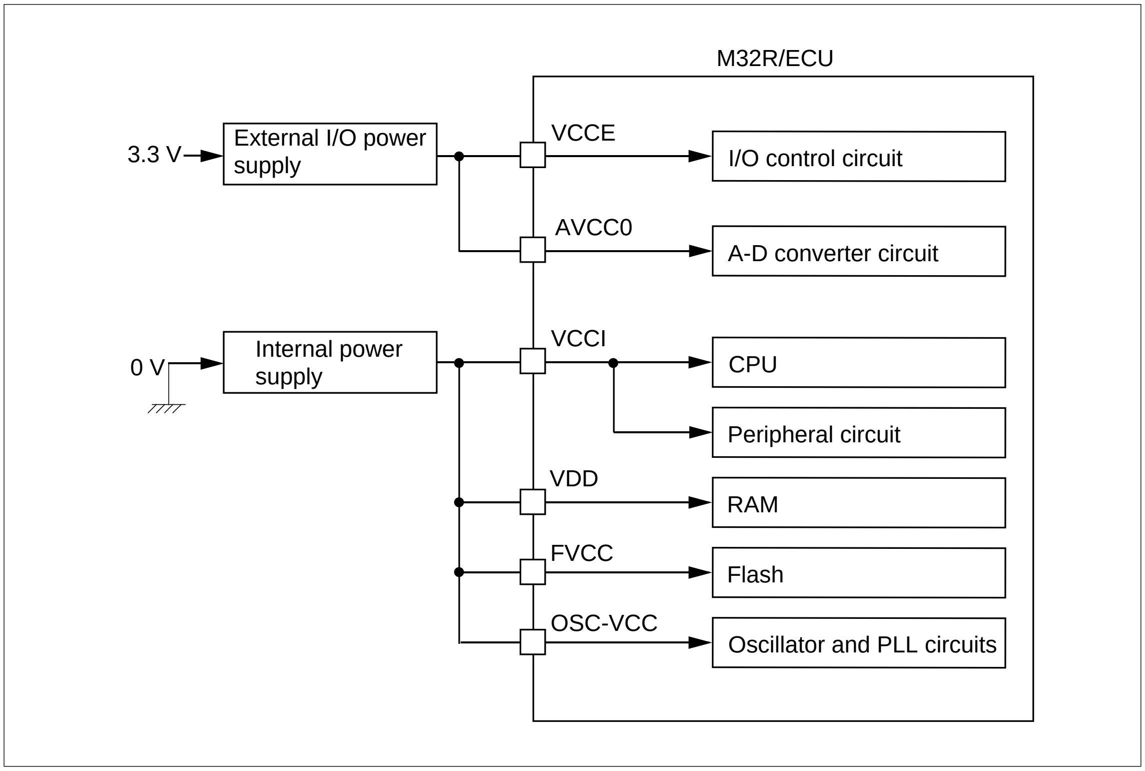


**Figure 20.3.9 CPU Stop State 2**

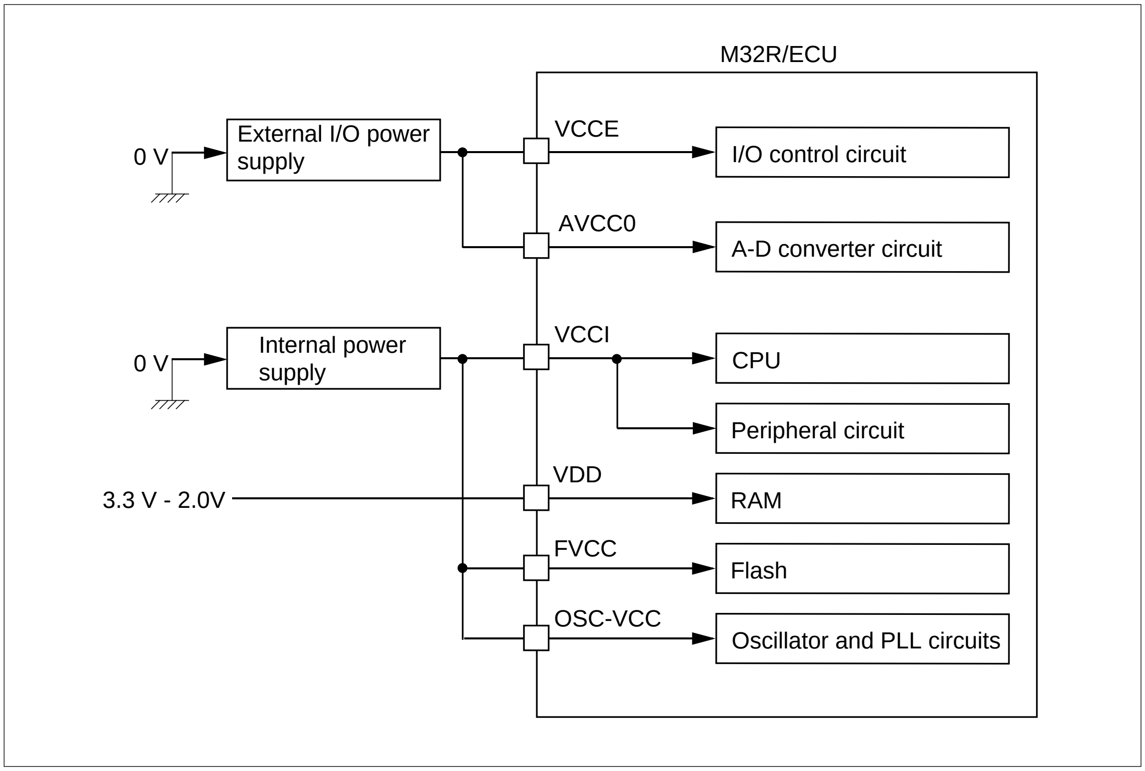


**Figure 20.3.10 SRAM Data Backup State**

* This is a blank page. *

CHAPTER 21

# ELECTRICAL CHARACTERISTICS

21.1 Electrical Characteristics (VCCE = 5V)

21.2 Electrical Characteristics (VCCE = 3.3V)

21.3 AC Characteristics

## 21.1 Electrical Characteristics (VCCE = 5V)

### 21.1.1 Absolute Maximum Ratings

Absolute Maximum Ratings (Guaranteed for Operation at -40 to 125°C)

| Symbol | Parameter | Condition | Rated Value | Unit |
|---|---|---|---|---|
| VCCI | Internal Logic Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| VDD | RAM Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| OSC-VCC | PLL Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| FVCC | Flash Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| VCCE | External I/O Buffer Voltage | VCCE≧AVCC≧VREF | -0.3 to 6.5 | V |
| AVCC | Analog Power Supply Voltage | VCCE≧AVCC≧VREF | -0.3 to 6.5 | V |
| VREF | Analog Reference Voltage | VCCE≧AVCC≧VREF | -0.3 to 6.5 | V |
| VI | Xin, VCNT | | -0.3 to OSC-VCC+0.3 | V |
| | Other | | -0.3 to VCCE+0.3 | |
| VO | Xout | | -0.3 to OSC-VCC+0.3 | V |
| | Other | | -0.3 to VCCE+0.3 | |
| Pd | Power Dissipation | Ta=-40 to 85°C | 600 | mW |
| | | Ta=-40 to 125°C | 500 | mW |
| TOPR | Operating Ambient Temperature (Note 1) | | -40 to 125 | °C |
| Tstg | Storage Temperature | | -65 to 150 | °C |

Note 1: This does not guarantee that the device can operate continuously at 125°C. If you are considering the use of this product in 125°C application, please consult Renesas.

### 21.1.2 Recommended Operating Conditions

Recommended Operating Conditions (Referenced to VCCE = 5 V ± 0.5 V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 85°C Unless Otherwise Noted)

| Symbol | Parameter | | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VCCE | External I/O Buffer Power Supply Voltage (Note 1) | | 4.5 | 5.0 | 5.5 | V |
| VCCI | Internal Logic Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| VDD | RAM Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| FVCC | Flash Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| AVCC | Analog Power Supply Voltage (Note1) | | 4.5 | 5.0 | 5.5 | V |
| OSC-VCC | PLL Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| VREF | Analog Reference Voltage (Note1) | | 4.5 | 5.0 | 5.5 | V |
| VIH | Input High Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0.8VCCE | | VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0.43VCCE | | VCCE | V |
| VIL | Input Low Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0 | | 0.2VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0 | | 0.16VCCE | V |
| IOH(peak) | High State Peak Output Current P0-P22 (Note 3) | | | | -10 | mA |
| IOH(avg) | High State Average Output Current P0-P22 (Note 4) | | | | -5 | mA |
| IOL(peak) | Low State Peak Output Current P0-P22 (Note 3) | | | | 10 | mA |
| IOL(avg) | Low State Average Output Current P0-P22 (Note 4) | | | | 5 | mA |
| CL | Output Load Capacitance | JTCK,JTDI,JTMS, JTDO,JTRST | | | 80 | PF |
| | | Other than above | 15 | | 50 | PF |
| f(XIN) | External Clock Input Frequency | | 5 | | 10 | MHz |

Note 1: Subject to conditions VCCE ≧ AVCC ≧ VREF.

Note 2: Subject to conditions VDD ≧ VCCI ≧ FVCC ≧ OSC-VCC

Note 3: The total amount of output current (peak) on ports must satisfy the conditions below.

| Ports P0 + P1 + P2 | ≦ 80 mA
| Ports P3 + P4 + P13 + P15 + P22 | ≦ 80 mA
| Ports P6 + P7 + P8 + P9 + P17 | ≦ 80 mA
| Ports P10 + P11 + P12 | ≦ 80 mA

Note 4: The average output current is a value averaged during a 100 ms period.

Recommended Operating Conditions (Referenced to VCCE = 5 V ± 0.5 V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C Unless Otherwise Noted)

| Symbol | Parameter | | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VCCE | External I/O Buffer Power Supply Voltage (Note 1) | | 4.5 | 5.0 | 5.5 | V |
| VCCI | Internal Logic Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| VDD | RAM Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| FVCC | Flash Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| AVCC | Analog Power Supply Voltage (Note 1) | | 4.5 | 5.0 | 5.5 | V |
| OSC-VCC | PLL Power Supply Voltage (Note 2) | | 3.0 | 3.3 | 3.6 | V |
| VREF | Analog Reference Voltage (Note 1) | | 4.5 | 5.0 | 5.5 | V |
| VIH | Input High Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0.8VCCE | | VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0.43VCCE | | VCCE | V |
| VIL | Input Low Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0 | | 0.2VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0 | | 0.16VCCE | V |
| IOH(peak) | High State Peak Output Current P0-P22 (Note 3) | | | | -10 | mA |
| IOH(avg) | High State Average Output Current P0-P22 (Note 4) | | | | -5 | mA |
| IOL(peak) | Low State Peak Output Current P0-P22 (Note 3) | | | | 10 | mA |
| IOL(avg) | Low State Average Output Current P0-P22 (Note 4) | | | | 5 | mA |
| CL | Output Load Capacitance | JTCK,JTDI,JTMS, JTDO,JTRST | | | 80 | PF |
| | | Other than above | 15 | | 50 | PF |
| f(XIN) | External Clock Input Frequency | | 5 | | 8 | MHz |

Note 1: Subject to conditions VCCE ≧ AVCC ≧ VREF.

Note 2: Subject to conditions VDD ≧ VCCI ≧ FVCC ≧ OSC-VCC

Note 3: The total amount of output current (peak) on ports must satisfy the conditions below.

| Ports P0 + P1 + P2 | ≦ 80 mA
| Ports P3 + P4 + P13 + P15 + P22 | ≦ 80 mA
| Ports P6 + P7 + P8 + P9 + P17 | ≦ 80 mA
| Ports P10 + P11 + P12 | ≦ 80 mA

Note 4: The average output current is a value averaged during a 100 ms period.

### 21.1.3 DC Characteristics

#### 21.1.3.1 Electrical Characteristics

(1) Electrical characteristics when f(XIN) = 10 MHz
(Referenced to VCCE = 5 V ± 0.5V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 85°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VOH | Output High Voltage | IOH≧-5mA | VCCE+0.165 × IOH(mA) | | VCCE | V |
| VOL | Output Low Voltage | IOL≦5mA | 0 | | 0.15 × IOL (mA) | V |
| VDD | RAM Retention Power Supply Voltage | When operating | 3.0 | | VCCI | V |
| | | When back-up | 2.0 | | 3.6 | |
| IIH | High State Input Current | VI=VCCE | -5 | | 5 | µA |
| IIL | Low State Input Current | VI=0V | -5 | | 5 | µA |
| ICC-5V | 5 V power supply (Note 1) | f(XIN)=10.0MHz, When reset | | | 1 | mA |
| | | f(XIN)=10.0MHz, When operating | | 1 | 10 | |
| ICCI-3V | 3.3 V power supply (Note 2) | f(XIN)=10.0MHz, When reset | | | 75 | mA |
| | | f(XIN)=10.0MHz, When operating | | 75 | 125 | |
| IDDhold | RAM Retention Power Supply Current | Ta=25°C | | See RAM retention power supply current characteristic graph | 50 | µA |
| | | Ta=85°C | | | 1500 | |
| VT+ — VT- | Hysteresis (Note 3)<br>RTDCLK, RTDRXD, SCLKI0,1, RXD0,1,2, TCLK3-0, TIN0,3,16-23, $\overline{\text{RESET}}$, FP, MOD0,1, JTMS, JTRST, JTDI | VCCE=5V | 1.0 | | | V |
| VT+ — VT- | Hysteresis (Note 4)<br>$\overline{\text{SBI}}$, $\overline{\text{HREQ}}$ | VCCE=5V | 0.3 | | | V |

Note 1: Total current when VCCE = AVCC = VREF in single-chip mode. See the next page for the rated values of power supply current on each power supply pin.

Note 2: Total current when VCCI = VDD = FVCC = OSC-VCC in single-chip mode. See the next page for the rated values of power supply current on each power supply pin.

Note 3: All these pins except $\overline{\text{RESET}}$, FP, MOD0, 1, JTMS, JTRST, and JTDI serve dual-functions.

Note 4: The $\overline{\text{HREQ}}$ pin serves dual-functions.

(2) Electrical characteristics of each power supply pin when f(XIN) = 10 MHz

(Referenced to VCCE = 5 V ± 0.5V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 85°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| ICCE | VCCE power supply current when operating | f(XIN)=10.0MHz | | | 10 | mA |
| ICCI | VCCI power supply current when operating | f(XIN)=10.0MHz | | | 120 | |
| IOSC-VCC | OSC-VCC power supply current when operating | f(XIN)=10.0MHz | | | 20 | mA |
| FICC | FVCC power supply current when operating (Note 1) | f(XIN)=10.0MHz | | | 50 | mA |
| IDD | VDD power supply current when operating (Note 2) | f(XIN)=10.0MHz | | | 35 | mA |
| IAVCC | AVCC power supply current when operating | f(XIN)=10.0MHz | | | 3 | mA |
| IVREF | VREF power supply current | f(XIN)=10.0MHz | | | 1 | mA |

Note 1: Maximum value including currents during program/erase operation.

Note 2: Maximum value including cases where the program is executed in RAM.

(3) Electrical characteristics when f(XIN) = 8 MHz

(Referenced to VCCE = 5 V ± 0.5V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VOH | Output High Voltage | IOH≧-5mA | VCCE+0.165 × IOH(mA) | | VCCE | V |
| VOL | Output Low Voltage | IOL≦5mA | 0 | | 0.15 × IOL (mA) | V |
| VDD | RAM Retention Power Supply Voltage | When operating | 3.0 | | VCCI | V |
| | | When back-up | 2.0 | | 3.6 | |
| IIH | High State Input Current | VI=VCCE | -5 | | 5 | μA |
| IIL | Low State Input Current | VI=0V | -5 | | 5 | μA |
| ICC-5V | 5 V power supply (Note 1) | f(XIN)=8.0MHz, When reset | | | 1 | mA |
| | | f(XIN)=8.0MHz, When operating | | 1 | 10 | |
| ICCI-3V | 3.3 V power supply (Note 2) | f(XIN)=8.0MHz, When reset | | | 70 | mA |
| | | f(XIN)=8.0MHz, When operating | | 60 | 110 | |
| IDDhold | RAM Retention Power Supply Current | Ta=25°C | | See RAM retention power supply current characteristic graph | 50 | μA |
| | | Ta=125°C | | | 4000 | |
| VT+ — VT- | Hysteresis (Note 3) RTDCLK, RTDRXD, SCLKI0,1, RXD0,1,2, TCLK3-0, TIN0,3,16-23, RESET, FP, MOD0,1, JTMS, JTRST, JTDI | VCCE=5V | 1.0 | | | V |
| VT+ — VT- | Hysteresis (Note 4) SBI, HREQ | VCCE=5V | 0.3 | | | V |

Note 1: Total current when VCCE = AVCC = VREF in single-chip mode. See the next page for the rated values of power supply current on each power supply pin.

Note 2: Total current when VCCI = VDD = FVCC = OSC-VCC in single-chip mode. See the next page for the rated values of power supply current on each power supply pin.

Note 3: All these pins except RESET, FP, MOD0, 1, JTMS, JTRST, and JTDI serve dual-functions.

Note 4: The HREQ pin serves dual-functions.

(4) Electrical characteristics of each power supply pin when f(XIN) = 8 MHz

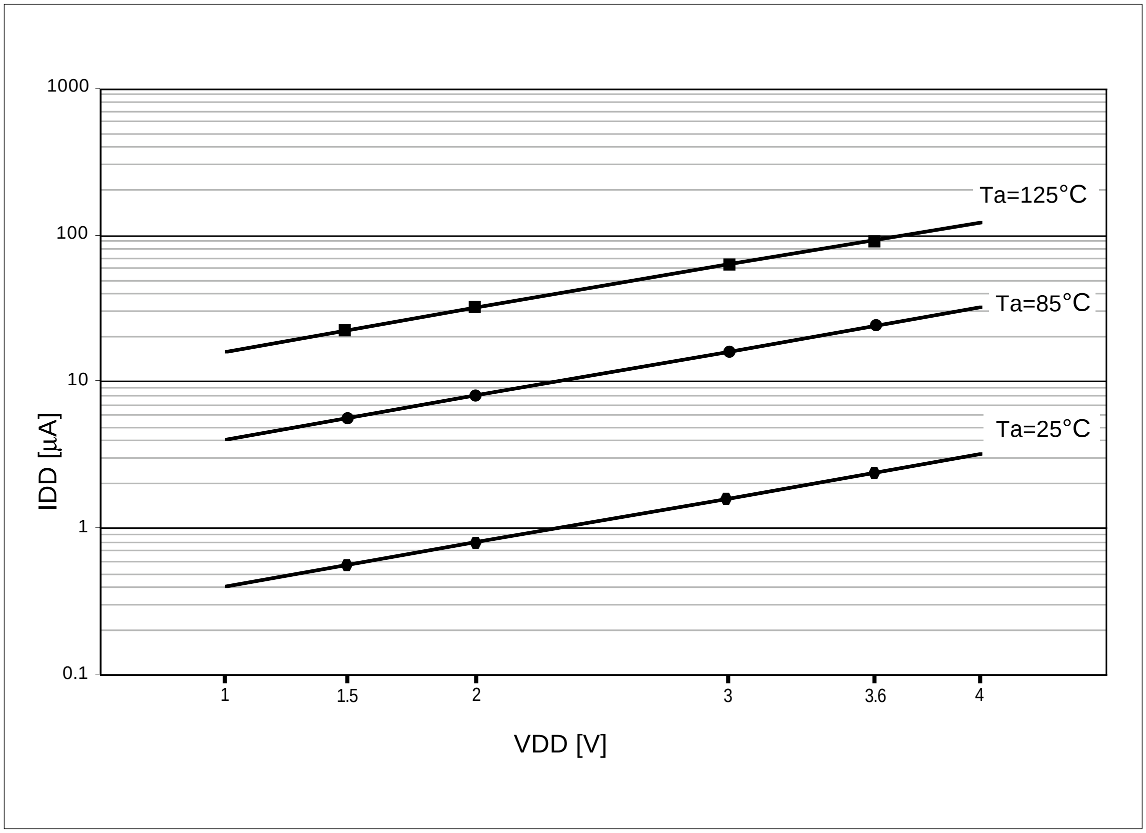
(Referenced to VCCE = 5 V ± 0.5V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| ICCE | VCCE power supply current when operating | f(XIN)=8.0MHz | | | 10 | mA |
| ICCI | VCCI power supply current when operating | f(XIN)=8.0MHz | | | 105 | |
| IOSC-VCC | OSCVCC power supply current when operating | f(XIN)=8.0MHz | | | 16 | mA |
| FICC | FVCC power supply current when operating (Note 1) | f(XIN)=8.0MHz | | | 50 | mA |
| IDD | VDD power supply current when operating (Note 2) | f(XIN)=8.0MHz | | | 30 | mA |
| IAVCC | AVCC power supply current when operating | f(XIN)=8.0MHz | | | 3 | mA |
| IVREF | VREF power supply current | f(XIN)=8.0MHz | | | 1 | mA |

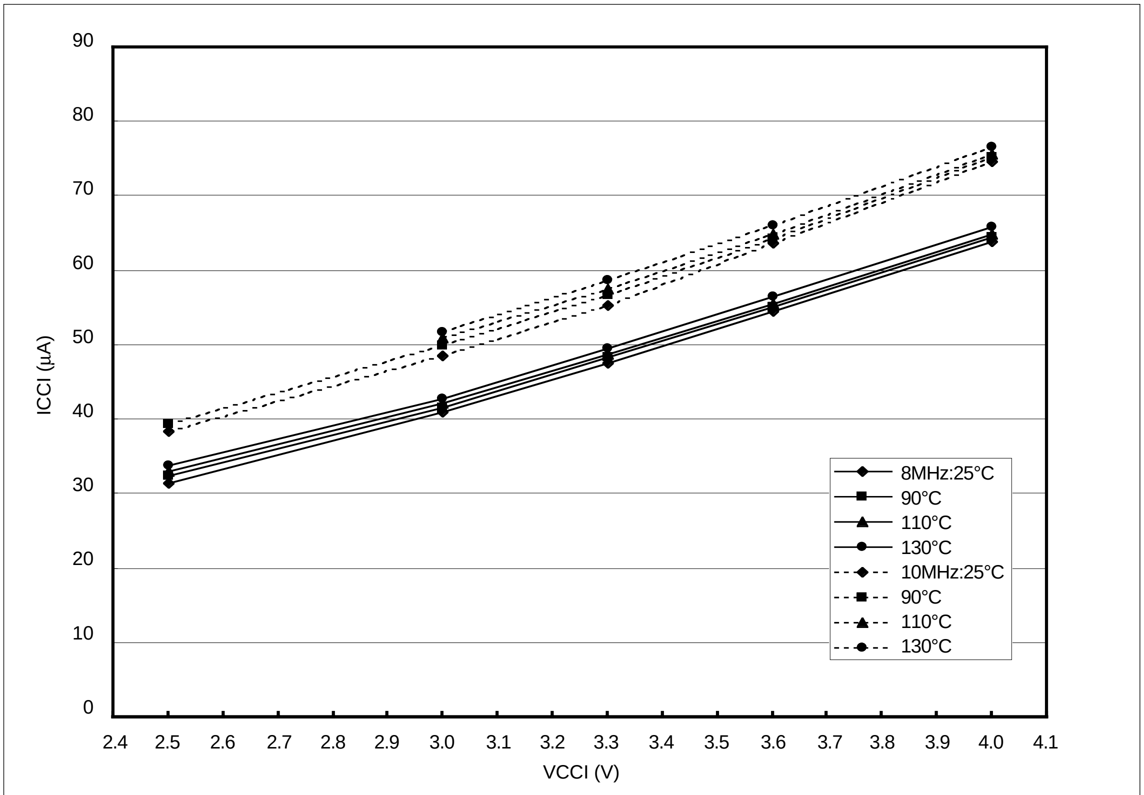
Note 1: Maximum value including currents during program/erase operation.

Note 2: Maximum value including cases where the program is executed in RAM.

**RAM retention power supply current in a standard sample (reference value)**

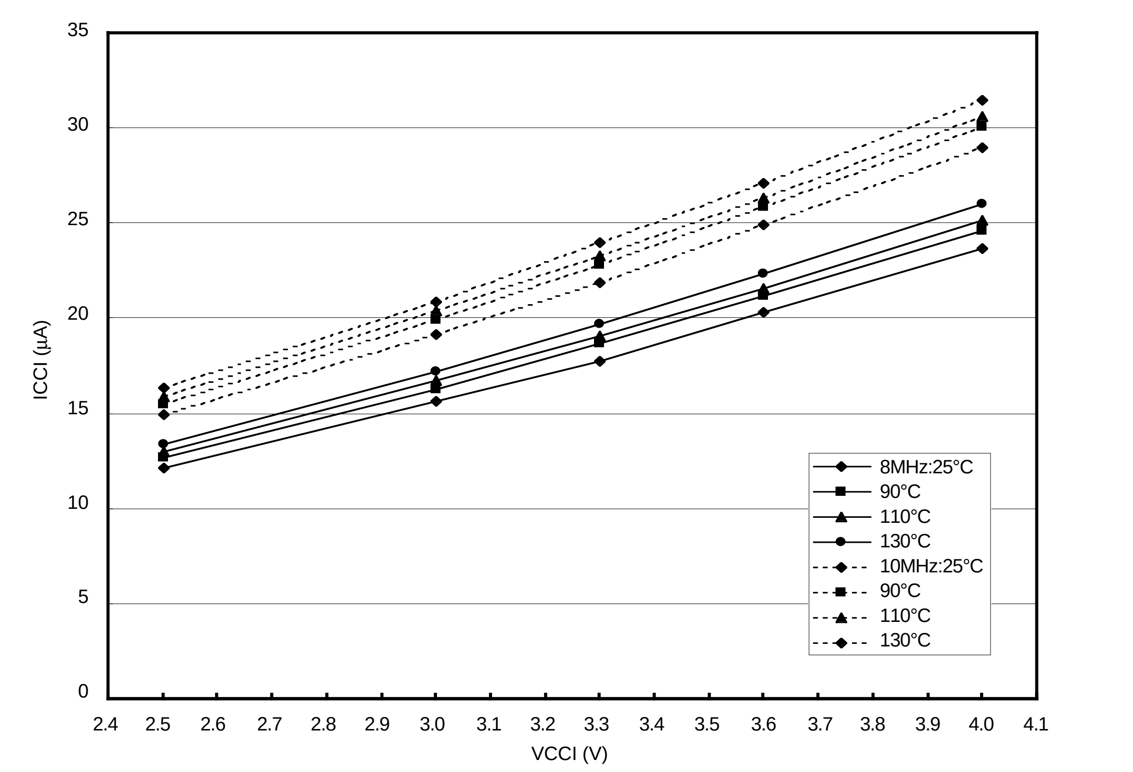


### Standard sample's ICCI-3V temperature characteristics (when operating: f = 8 MHz, 10 MHz)



Note: • VCCI = VDD = FVCC = OSCVCC, VCCE = AVCC = 5.0V

### Standard sample's ICCI-3V temperature characteristics (when reset: f = 8 MHz, 10 MHz)



Note: • VCCI = VDD = FVCC = OSCVCC, VCCE = AVCC = 5.0V

### 21.1.3.2 Flash Related Electrical Characteristics

Flash Related Electrical Characteristics (Referenced to VCCE = 5 V ± 0.5 V, VCCI = 3.3 V ± 0.3 V Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| Ifvcc1 | FVCC Power Supply Current (when Programming) | | | | 50 | mA |
| Ifvcc2 | FVCC Power Supply Current (when Erasing) | | | | 40 | mA |
| Topr | Flash Rewrite Ambient Temperature | | 0 | | 70 | °C |
| cycle | Rewrite Durability | | | | 100 | times |
| tPRG | Program Time | 1 Page | | 8 | 120 | ms |
| tBERS | Block Erase Time | 1 Block | | 50 | 600 | ms |

### 21.1.4 A-D Conversion Characteristics

A-D Conversion Characteristics (Referenced to AVCC = VREF = VCCE = 5.12 V, Ta = -40 to 85°C, f(XIN) = 10.0 MHz Unless Otherwise Noted)

| Symbol | Parameter | | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | MIN | TYP | MAX | |
| — | Resolution | | VREF=VCCE | | | 10 | Bits |
| — | Absolute Accuracy (Note 1) | | | | | ±2 | LSB |
| TCONV | Conversion Time | During nomal mode | | 14950 | | | ns |
| | | During double-speed mode | | 8650 | | | |
| IIAN | Analog Input Leakage Current | | (Note 2) | -5 | | 5 | µA |

Note 1: The absolute accuracy represents the accuracy of output code including all error sources (including quantization error) of the A-D converter relative to the analog input, and is obtained by the equation below:

Absolute accuracy = output code – (analog input voltage ANi/ 1 LSB)
When AVCC = VREF = 5.12 V, 1 LSB = 5 mV.

Note 2: This referes to input leakage current on AN0-AN15 when the A-D converter remains idle. Input voltage condition: 0 ≦ ANi≦ AVCC. Temperature condition: -40 to 85°C.

A-D Conversion Characteristics (Referenced to AVCC = VREF = VCCE = 5.12 V, Ta = -40 to 125°C, f(XIN) = 8.0 MHz Unless Otherwise Noted)

| Symbol | Parameter | | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | MIN | TYP | MAX | |
| — | Resolution | | VREF=VCCE | | | 10 | Bits |
| — | Absolute Accuracy (Note 1) | | | | | ±2 | LSB |
| TCONV | Conversion Time | During nomal mode | | 18687.5 | | | ns |
| | | During double-speed mode | | 10812.5 | | | |
| IIAN | Analog Input Leakage Current | | (Note 2) | -5 | | 5 | µA |

Note1: The absolute accuracy represents the accuracy of output code including all error sources (including quantization error) of the A-D converter relative to the analog input, and is obtained by the equation below:

Absolute accuracy = output code – (analog input voltage ANi/ 1 LSB)
When AVCC = VREF = 5.12 V, 1 LSB = 5 mV.

Note 2: This referes to input leakage current on AN0-AN15 when the A-D converter remains idle. Input voltage condition: 0≦ANi≦ AVCC. Temperature condition: -40 to 85°C.

## 21.2 ELECTRICAL CHARACTERISTICS (VCCE = 3.3V)

### 21.2.1 Absolute Maximum Ratings

Absolute Maximum Ratings (Guaranteed for Operation at -40 to 125°C)

| Symbol | Parameter | Condition | Rated Value | Unit |
|---|---|---|---|---|
| VCCI | Internal Logic Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| VDD | RAM Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| OSC-VCC | PLL Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| FVCC | Flash Power Supply Voltage | VDD≧VCCI≧FVCC=OSC-VCC | -0.3 to 4.2 | V |
| VCCE | External I/O Buffer Voltage | VCCE≧AVCC≧VREF | -0.3 to 6.5 | V |
| AVCC | Analog Power Supply Voltage | VCCE≧AVCC≧VREF | -0.3 to 6.5 | V |
| VREF | Analog Reference Voltage | VCCE≧AVCC≧VREF | -0.3 to 6.5 | V |
| VI | Xin, VCNT | | -0.3 to OSC-VCC+0.3 | V |
| | Other | | -0.3 to VCCE+0.3 | |
| VO | Xout | | -0.3 to OSC-VCC+0.3 | V |
| | Other | | -0.3 to VCCE+0.3 | |
| Pd | Power Dissipation | Ta=-40 to 85°C | 600 | mW |
| | | Ta=-40 to 125°C | 500 | mW |
| TOPR | Operating Ambient Temperature (Note 1) | | -40 to 125 | °C |
| Tstg | Storage Temperature | | -65 to 150 | °C |

Note 1: This does not guarantee that the device can operate continuously at 125°C. If you are considering the use of this product in 125°C application, please consult Renesas.

### 21.2.2 Recommended Operating Conditions

Recommended Operating Conditions (Referenced to VCCE = VCCI = 3.3 V ± 0.3 V, Ta = -40 to 85°C Unless Otherwise Noted)

| Symbol | Parameter | | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VCCE | External I/O Buffer Power Supply Voltage | | 3.0 | 3.3 | 3.6 | V |
| VCCI | Internal Logic Power Supply Voltage | | 3.0 | 3.3 | 3.6 | V |
| VDD | RAM Power Supply Voltage | | 3.0≦ VCCI-0.3 | VCCI | VCCI+0.3≦3.6 | V |
| FVCC | Flash Power Supply Voltage | | 3.0≦ VCCI-0.3 | VCCI | VCCI+0.3≦3.6 | V |
| AVCC | Analog Power Supply Voltage | | 3.0≦VCCE-0.3 | VCCE | VCCE+0.3≦3.6 | V |
| OSC-VCC | PLL Power Supply Voltage | | 3.0≦ VCCI-0.3 | VCCI | VCCI+0.3≦3.6 | V |
| VREF | Analog Reference Voltage | | 3.0≦VCCE-0.3 | VCCE | VCCE+0.3≦3.6 | V |
| VIH | Input High Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0.8VCCE | | VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0.43VCCE | | VCCE | V |
| VIL | Input Low Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0 | | 0.2VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0 | | 0.16VCCE | V |
| IOH(peak) | High State Peak Output Current P0-P22 (Note 1) | | | | -10 | mA |
| IOH(avg) | High State Average Output Current P0-P22 (Note 2) | | | | -5 | mA |
| IOL(peak) | Low State Peak Output Current P0-P22 (Note 1) | | | | 10 | mA |
| IOL(avg) | Low State Average Output Current P0-P22 (Note 2) | | | | 5 | mA |
| CL | Output Load Capacitance | JTCK,JTDI,JTMS, JTDO,JTRST | | | 80 | PF |
| | | Other than above | 15 | | 50 | PF |
| f(XIN) | External Clock Input Frequency | | 5 | | 10 | MHz |

Note 1: The total amount of output current (peak) on ports must satisfy the conditions below.
| Ports P0 + P1 + P2 | ≦ 80 mA
| Ports P3 + P4 + P13 + P15 + P22 | ≦ 80 mA
| Ports P6 + P7 + P8 + P9 + P17 | ≦ 80 mA
| Ports P10 + P11 + P12 | ≦ 80 mA

Note 2: The average output current is a value averaged during a 100 ms period.

Recommended Operating Conditions (Referenced to VCCE = VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C Unless Otherwise Noted)

| Symbol | Parameter | | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VCCE | External I/O Buffer Power Supply Voltage | | 3.0 | 3.3 | 3.6 | V |
| VCCI | Internal Logic Power Supply Voltage | | 3.0 | 3.3 | 3.6 | V |
| VDD | RAM Power Supply Voltage | | 3.0≦VCCI-0.3 | VCCI | VCCI+0.3≦3.6 | V |
| FVCC | Flash Power Supply Voltage | | 3.0≦VCCI-0.3 | VCCI | VCCI+0.3≦3.6 | V |
| AVCC | Analog Power Supply Voltage | | 3.0≦VCCE-0.3 | VCCE | VCCE+0.3≦3.6 | V |
| OSC-VCC | PLL Power Supply Voltage | | 3.0≦VCCI-0.3 | VCCI | VCCI+0.3≦3.6 | V |
| VREF | Analog Reference Voltage | | 3.0≦VCCE-0.3 | VCCE | VCCE+0.3≦3.6 | V |
| VIH | Input High Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0.8VCCE | | VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0.43VCCE | | VCCE | V |
| VIL | Input Low Voltage | Ports P0-P22, $\overline{\text{RESET}}$, MOD0, MOD1, FP | 0 | | 0.2VCCE | V |
| | | Ports P0, P1 (external extension/ processor mode only), $\overline{\text{WAIT}}$ | 0 | | 0.16VCCE | V |
| IOH(peak) | High State Peak Output Current P0-P22 (Note 1) | | | | -10 | mA |
| IOH(avg) | High State Average Output Current P0-P22 (Note 2) | | | | -5 | mA |
| IOL(peak) | Low State Peak Output Current P0-P22 (Note 1) | | | | 10 | mA |
| IOL(avg) | Low State Average Output Current P0-P22 (Note 2) | | | | 5 | mA |
| CL | Output Load Capacitance | JTCK,JTDI,JTMS, JTDO,JTRST | | | 80 | PF |
| | | Other than above | 15 | | 50 | PF |
| f(XIN) | External Clock Input Frequency | | 5 | | 8 | MHz |

Note 1: The total amount of output current (peak) on ports must satisfy the conditions below.

| Ports P0 + P1 + P2 | ≦ 80 mA

| Ports P3 + P4 + P13 + P15 + P22 | ≦ 80 mA

| Ports P6 + P7 + P8 + P9 + P17 | ≦ 80 mA

| Ports P10 + P11 + P12 | ≦ 80 mA

Note 2: The average output current is a value averaged during a 100 ms period.

## 21.2.3 DC Characteristics

### 21.2.3.1 Electrical Characteristics

(1) Electrical characteristics when f(XIN) = 10 MHz

(Referenced to VCCE = VCCI = 3.3 V ± 0.3 V, Ta = -40 to 85°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VOH | Output High Voltage | IOH≧-2mA | VCCE+0.5<br>×IOH(mA) | | VCCE | V |
| VOL | Output Low Voltage | IOL≦2mA | 0 | | 0.225×<br>IOL (mA) | V |
| VDD | RAM Retention Power Supply Voltage | When operating | 3.0 | | VCCI | V |
| | | When back-up | 2.0 | | 3.6 | |
| IIH | High State Input Current | VI=VCCE | -5 | | 5 | µA |
| IIL | Low State Input Current | VI=0V | -5 | | 5 | µA |
| ICCres | Power supply current when reset (Note 1) | f(XIN)=10.0MHz, When reset | | | 76 | mA |
| ICC | Power supply current when operating (Note 1) | f(XIN)=10.0MHz, When operating | | 76 | 132 | |
| IDDhold | RAM Retention Power Supply Current | Ta=25°C | | See RAM retention power supply current characteristic graph | 50 | µA |
| | | Ta=85°C | | | 1500 | |
| $V_{T+}$ — $V_{T-}$ | Hysteresis (Note 2)<br>RTDCLK, RTDRXD, SCLKI0,1, RXD0,1,2, TCLK3-0, TIN0,3,16-23, $\overline{\mathrm{RESET}}$, FP, MOD0,1, JTMS, JTRST, JTDI | VCCE=3.3V | 0.65 | | | V |
| $V_{T+}$ — $V_{T-}$ | Hysteresis (Note 3)<br>$\overline{\mathrm{SBI}}$, $\overline{\mathrm{HREQ}}$ | VCCE=3.3V | 0.2 | | | V |

Note 1: Total current when VCCE = AVCC = VREF= VCCI = VDD = FVCC = OSC-VCC in single-chip mode. See the next page for the rated values of power supply current on each power supply pin.

Note 2: All these pins except $\overline{\mathrm{RESET}}$ serve dual-functions.

Note 3: The $\overline{\mathrm{HREQ}}$ pin serves dual-functions.

(2) Electrical characteristics of each power supply pin when f(XIN) = 10 MHz

(Referenced to VCCE = VCCI = 3.3 V ± 0.3 V, Ta = -40 to 85°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| ICCE | VCCE power supply current when operating | f(XIN)=10.0MHz | | | 7 | mA |
| ICCI | VCCI power supply current when operating | f(XIN)=10.0MHz | | | 120 | |
| OSC-ICC | OSC-VCC power supply current when operating | f(XIN)=10.0MHz | | | 20 | mA |
| FICC | FVCC power supply current when operating (Note 1) | f(XIN)=10.0MHz | | | 50 | mA |
| IDD | VDD power supply current when operating (Note 2) | f(XIN)=10.0MHz | | | 35 | mA |
| IAVCC | AVCC power supply current when operating | f(XIN)=10.0MHz | | | 2 | mA |
| IVREF | VREF power supply current | f(XIN)=10.0MHz | | | 1 | mA |

Note 1: Maximum value including currents during program/erase operation.

Note 2: Maximum value including cases where the program is executed in RAM.

(3) Electrical characteristics when f(XIN) = 8 MHz
(Referenced to VCCE = VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| VOH | Output High Voltage | IOH≧ -2mA | VCCE+0.5<br>×IOH(mA) | | VCCE | V |
| VOL | Output Low Voltage | IOL≦ 2mA | 0 | | 0.225×<br>IOL (ma) | V |
| VDD | RAM Retention Power Supply Voltage | When operating | 3.0 | | VCCI | V |
| | | When back-up | 2.0 | | 3.6 | |
| IIH | High State Input Current | VI=VCCE | -5 | | 5 | μA |
| IIL | Low State Input Current | VI=0V | -5 | | 5 | μA |
| ICCres | Power supply current when reset (Note 1) | f(XIN)=8.0MHz, When reset | | | 71 | mA |
| ICC | Power supply current when operating (Note 1) | f(XIN)=8.0MHz, When operating | | 61 | 117 | |
| IDDhold | RAM Retention Power Supply Current | Ta=25°C | | See RAM retention power supply current characteristic graph | 50 | μA |
| | | Ta=125°C | | | 4000 | |
| $V_{T+}$ —$V_{T-}$ | Hysteresis (Note 2)<br>RTDCLK, RTDRXD, SCLKI0,1, RXD0,1,2, TCLK3-0, TIN0,3,16-23, $\overline{\text{RESET}}$, FP, MOD0,1, JTMS, JTRST, JTDI | VCCE=3.3V | 0.65 | | | V |
| $V_{T+}$ —$V_{T-}$ | Hysteresis (Note 3)<br>$\overline{\text{SBI}}$, $\overline{\text{HREQ}}$ | VCCE=3.3V | 0.2 | | | V |

Note 1: Total current when VCCE = AVCC = VREF= VCCI = VDD = FVCC = OSC-VCC in single-chip mode. See the next page for the rated values of power supply current on each power supply pin.

Note 2: All these pins except $\overline{\text{RESET}}$ serve dual-functions.

Note 3: The $\overline{\text{HREQ}}$ pin serves dual-functions.

(4) Electrical characteristics of each power supply pin when f(XIN) = 8 MHz
(Referenced to VCCE = VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| ICCE | VCCE power supply current when operating | f(XIN)=8.0MHz | | | 7 | mA |
| ICCI | VCCI power supply current when operating | f(XIN)=8.0MHz | | | 105 | |
| OSC-ICC | OSC-VCC power supply current when operating | f(XIN)=8.0MHz | | | 16 | mA |
| FICC | FVCC power supply current when operating (Note 1) | f(XIN)=8.0MHz | | | 50 | mA |
| IDD | VDD power supply current when operating (Note 2) | f(XIN)=8.0MHz | | | 30 | mA |
| IAVCC | AVCC power supply current when operating | f(XIN)=8.0MHz | | | 2 | mA |
| IVREF | VREF power supply current | f(XIN)=8.0MHz | | | 1 | mA |

Note 1: Maximum value including currents during program/erase operation.

Note 2: Maximum value including cases where the program is executed in RAM.

### 21.2.3.2 Flash Related Electrical Characteristics

Flash Related Electrical Characteristics (Referenced to VCCE = VCCI = 3.3 V ± 0.3 V Unless Otherwise Noted)

| Symbol | Parameter | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|
| | | | MIN | TYP | MAX | |
| Ifvcc1 | FVCC Power Supply Current (when Programming) | | | | 50 | mA |
| Ifvcc2 | FVCC Power Supply Current (when Erasing) | | | | 40 | mA |
| Topr | Flash Rewrite Ambient Temperature | | 0 | | 70 | °C |
| cycle | Rewrite Durability | | | | 100 | times |
| tPRG | Program Time | 1 Page | | 8 | 120 | ms |
| tBERS | Block Erase Time | 1 Block | | 50 | 600 | ms |

### 21.2.4 A-D Conversion Characteristics

A-D Conversion Characteristics (Referenced to AVCC = VREF = VCCE = 3.3 V, Ta = -40 to 85°C, f(XIN) = 10.0 MHz Unless Otherwise Noted)

| Symbol | Parameter | | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | MIN | TYP | MAX | |
| — | Resolution | | VREF=VCCE | | | 10 | Bits |
| — | Absolute Accuracy (Note 1) | | | | | ±4 | LSB |
| TCONV | Conversion Time | During nomal mode | | 14950 | | | ns |
| | | During double-speed mode | | 8650 | | | |
| IIAN | Analog Input Leakage Current | | (Note 2) | -5 | | 5 | µA |

Note 1: The absolute accuracy represents the accuracy of output code including all error sources (including quantization error) of the A-D converter relative to the analog input, and is obtained by the equation below:

Absolute accuracy = output code – (analog input voltage ANi/ 1 LSB)
When AVCC = VREF = 3.072 V, 1 LSB = 3 mV.

Note 2: This referes to input leakage current on AN0-AN15 when the A-D converter remains idle. Input voltage condition: 0≦ ANi≦ AVCC. Temperature condition: -40 to 85°C.

A-D Conversion Characteristics (Referenced to AVCC = VREF = VCCE = 3.3 V, Ta = -40 to 125°C, f(XIN) = 8.0 MHz Unless Otherwise Noted)

| Symbol | Parameter | | Condition | Rated Value | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | MIN | TYP | MAX | |
| — | Resolution | | VREF=VCCE | | | 10 | Bits |
| — | Absolute Accuracy (Note 1) | | | | | ±4 | LSB |
| TCONV | Conversion Time | During nomal mode | | 18687.5 | | | ns |
| | | During double-speed mode | | 10812.5 | | | |
| IIAN | Analog Input Leakage Current | | (Note 2) | -5 | | 5 | µA |

Note 1: The absolute accuracy represents the accuracy of output code including all error sources (including quantization error) of the A-D converter relative to the analog input, and is obtained by the equation below:

Absolute accuracy = output code – (analog input voltage ANi/ 1 LSB)
When AVCC = VREF = 3.072 V, 1 LSB = 3 mV.

Note 2: This referes to input leakage current on AN0-AN15 when the A-D converter remains idle. Input voltage condition: 0 ≦ANi≦ AVCC. Temperature condition: -40 to 85°C.

## 21.3 AC Characteristics

### 21.3.1 Timing Requirements

- Unless otherwise noted, timing conditions are VCCE = 5 V ± 0.5 V or VCCE = 3.3 V ± 0.3 V, VCCI = 3.3 V ± 0.3 V, Ta = -40 to 125°C
- The characteristic values apply to the case of concentrated capacitance with an output load capacitance of 15 to 50 pF (however, 80 pF for JTAG-related).

**(1) Input/output ports**

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.1 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{su}$(P-E) | Port Input Setup Time | | 100 | | ns | ① |
| $t_{h}$(E-P) | Port Input Hold Time | | 0 | | ns | ② |

**(2) Serial I/O**

**a) CSIO mode, with internal clock selected**

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.2 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| tsu(D-CLK) | RxD Input Setup Time | | 150 | | ns | ④ |
| th(CLK-D) | RxD Input Hold Time | | 50 | | ns | ⑤ |

**b) CSIO mode, with external clock selected**

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.2 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| tc(CLK) | CLK Input Cycle Time | | 640 | | ns | ⑦ |
| tw(CLKH) | CLK Input High Pulse Width | | 300 | | ns | ⑧ |
| tw(CLKL) | CLK Input Low Pulse Width | | 300 | | ns | ⑨ |
| tsu(D-CLK) | RxD Input Setup Time | | 60 | | ns | ⑩ |
| th(CLK-D) | RxD Input Hold Time | | 100 | | ns | ⑪ |

**(3) SBI**

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.3 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| tw(SBIL) | SBI Input Pulse Width | | $\frac{5}{2}$tc(BCLK) | | ns | ⑬ |

### (4) TIN

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.5 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{w(TIN)}$ | TIN Input Pulse Width | | $\frac{7}{2}$ tc(BCLK) | | ns | 14 |

### (5) TCLK

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.6 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{w(TCLKH)}$ | TCLK Input High Pulse Width | | $\frac{7}{2}$ tc(BCLK) | | ns | 99 |
| $t_{w(TCLKL)}$ | TCLK Input Low Pulse Width | | $\frac{7}{2}$ tc(BCLK) | | ns | 100 |

### (6) Read and write timing

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.7<br>21.3.8<br>21.3.9 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{su(D-BCLKH)}$ | Data Input Setup Time before BCLK | | 26 | | ns | 31 |
| $t_{h(BCLKH-D)}$ | Data Input Hold Time after BCLK | | 0 | | ns | 32 |
| $t_{su(WAITL-BCLKH)}$ | $\overline{\text{WAIT}}$ Input Setup Time before BCLK | | 26 | | ns | 33 |
| $t_{h(BCLKH-WAITL)}$ | $\overline{\text{WAIT}}$ Input Hold Time after BCLK | | 0 | | ns | 34 |
| $t_{su(WAITH-BCLKH)}$ | $\overline{\text{WAIT}}$ Input Setup Time before BCLK | | 26 | | ns | 78 |
| $t_{h(BCLKH-WAITH)}$ | $\overline{\text{WAIT}}$ Input Hold Time after BCLK | | 0 | | ns | 79 |
| $t_{w(RDL)}$ | Read Low Pulse Width | | $\frac{3}{2}$ tc(BCLK)-23 | | ns | 43 |
| $t_{su(D-RDH)}$ | Data Input Setup Time before Read | | 30 | | ns | 44 |
| $t_{h(RDH-D)}$ | Data Input Hold Time after Read | | 0 | | ns | 45 |
| $t_{w(BLWL)}$<br>$t_{w(BHWL)}$ | Write Low Pulse Width (Byte write mode) | | tc(BCLK) -25 | | ns | 51 |
| $t_{d(RDH-BLWL)}$<br>$t_{d(RDH-BHWL)}$ | Write Delay Time after Read | | $\frac{\text{tc(BCLK)}}{2}$ -10 | | ns | 56 |
| $t_{d(BLWH-RDL)}$<br>$t_{d(BHWH-RDL)}$ | Read Delay Time after Write | | $\frac{\text{tc(BCLK)}}{2}$ -10 | | ns | 57 |
| $t_{w(WRL)}$ | Write Low Pulse Width (Byte enable mode) | | tc(BCLK) -25 | | ns | 68 |
| $t_{d(RDH-BLEL)}$<br>$t_{d(RDH-BHEL)}$ | Write Delay Time after Read (Byte enable mode) | | $\frac{\text{tc(BCLK)}}{2}$ -10 | | ns | 80 |
| $t_{d(BLEH-RDL)}$<br>$t_{d(BHEH-RDL)}$ | Read Delay Time after Write (Byte enable mode) | | $\frac{\text{tc(BCLK)}}{2}$ -10 | | ns | 81 |

### (7) Bus arbitration timing

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.10 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{su(HREQL-BCLKH)}$ | $\overline{\text{HREQ}}$ Input Setup Time before BCLK | | 27 | | ns | 35 |
| $t_{h(BCLKH-HREQL)}$ | $\overline{\text{HREQ}}$ Input Hold Time after BCLK | | 0 | | ns | 36 |

### (8) Input transition time on JTAG pin

| Symbol | Condition | | | Rated Value | | Unit | See Figure 21.3.11 |
|---|---|---|---|---|---|---|---|
| | | | | MIN | MAX | | |
| $t_r$ | Input Rising Transition Time | Other than JTRST pin<br>(JTCK,JTDI,JTMS,JTDO) | | | 10 | ns | 58 |
| | | JTRST pin | When using TAP | | 10 | ns | |
| | | | When not using TAP | | 2 | ms | |
| $t_f$ | Input Falling Transition Time | Other than JTRST pin<br>(JTCK,JTDI,JTMS,JTDO) | | | 10 | ns | 59 |
| | | JTRST pin | When using TAP | | 10 | ns | |
| | | | When not using TAP | | 2 | ms | |

Note: • Stipulated values are guaranteed values when the test pin load capacitance CL=80pF.

### (9) JTAG interface timing

| Symbol | Condition | Rated Value | | Unit | See Figure 21.3.12 |
|---|---|---|---|---|---|
| | | MIN | MAX | | |
| $t_{c(JTCK)}$ | JTCK Input Cycle Time | 100 | | ns | 60 |
| $t_{w(JTCKH)}$ | JTCK Input High Pulse Width | 40 | | ns | 61 |
| $t_{w(JTCKL)}$ | JTCK Input Low Pulse Width | 40 | | ns | 62 |
| $t_{su(JTDI-JTCK)}$ | JTDI, JTMS Input Setup Time | 15 | | ns | 63 |
| $t_{h(JTCK-JTDI)}$ | JTDI, JTMS Input Hold Time | 20 | | ns | 64 |
| $t_{d(JTCK-JTDOV)}$ | JTDO Output Delay Time after JTCK Fall | | 40 | ns | 65 |
| $t_{d(JTCK-JTDOX)}$ | JTDO Output Hi-Z Delay Time after JTCK Fall | | 40 | ns | 66 |
| $t_{W(JTRST)}$ | TRST Input Low Pulse Width | $t_{c(JTCK)}$ | | ns | 67 |

Note: • Stipulated values are guaranteed values when the test pin load capacitance CL=80pF.

### (10) RTD timing

| Symbol | Parameter | Rated Value | | Unit | See Figure 21.3.13 |
|---|---|---|---|---|---|
| | | MIN | MAX | | |
| tc(RTDCLK) | RTDCLK Input Cycle Time | 500 | | ns | 90 |
| tw(RTDCLKH) | RTDCLK Input High Pulse Width | 230 | | ns | 83 |
| tw(RTDCLKL) | RTDCLK Input Low Pulse Width | 230 | | ns | 84 |
| td(RTDCLKH-RTDACK) | RTDACK Delay Time after RTDCLK Input | | 160 | ns | 85 |
| tv(RTDCLKL-RTDACK) | Valid RTDACK Time after RTDCLK input | | 160 | ns | 86 |
| td(RTDCLKH-RTDTXD) | RTDTXD Delay Time after RTDCLK Input | | tw(RTDCLKH)+160 | ns | 87 |
| th(RTDCLKH-RTDRXD) | RTDRXD Input Hold Time | 50 | | ns | 88 |
| tv(RTDRXD-RTDCLKL) | RTDRXD Input Setup Time | 60 | | ns | 89 |

### 21.3.2 Switching Characteristics

#### (1) Input/output ports

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.1 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{d(E-P)}$ | Port Data Output Delay Time | | | 100 | ns | 3 |

#### (2) Serial I/O

**a) CSIO mode, with internal clock selected**

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.2 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{d(CLK-D)}$ | TxD Output Delay Time | | | 60 | ns | 6 |
| $t_{h(CLK-D)}$ | TxD Hold Time | | 0 | | ns | 82 |

**b) CSIO mode, with external clock selected**

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.2 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{d(CLK-D)}$ | TxD Output Delay Time | | | 160 | ns | 12 |

#### (3) TO

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.4 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{d(BCLK-TO)}$ | TO Output Delay Time | | | 100 | ns | 15 |

### (4) Read and write timing

| Symbol | Parameter | Condition | Rated Value | | Unit | See<br>Figure<br>21.3.7<br>21.3.8<br>21.3.9 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{c(BCLK)}$ | BCLK Output Cycle Time | | | $\frac{t_{c(Xin)}}{2}$ | ns | ⑯ |
| $t_{w(BCLKH)}$ | BCLK Output High Pulse Width | | $\frac{t_{c(BCLK)}}{2} - 5$ | | ns | ⑰ |
| $t_{w(BCLKL)}$ | BCLK Output Low Pulse Width | | $\frac{t_{c(BCLK)}}{2} - 5$ | | ns | ⑱ |
| $t_{d(BCLKH-A)}$ | Address Delay Time after BCLK | | | 24 | ns | ⑲ |
| $t_{d(BCLKH-CS)}$ | Chip Select Delay Time after BCLK | | | 24 | ns | ⑳ |
| $t_{v(BCLKH-A)}$ | Valid Address Time after BCLK | | -11 | | ns | ㉑ |
| $t_{v(BCLKH-CS)}$ | Valid Chip Select Time after BCLK | | -11 | | ns | ㉒ |
| $t_{d(BCLKL-RDL)}$ | Read Delay Time after BCLK | | | 10 | ns | ㉓ |
| $t_{v(BCLKH-RDL)}$ | Valid Read Time after BCLK | | -12 | | ns | ㉔ |
| $t_{d(BCLKL-BLWL)}$<br>$t_{d(BCLKL-BHWL)}$ | Write Delay Time after BCLK | | | 11 | ns | ㉕ |
| $t_{v(BCLKL-BLWL)}$<br>$t_{d(BCLKL-D)}$ | Valid Write Time after BCLK | | -12 | | ns | ㉖ |
| $t_{d(BCLKL-D)}$ | Data Output Delay Time after BCLK | | | 18 | ns | ㉗ |
| $t_{v(BCLKH-D)}$ | Valid Data Output Time after BCLK | | -16 | | ns | ㉘ |
| $t_{pzx(BCLKL-DZ)}$ | Data Output Enable Time after BCLK | | -19 | | ns | ㉙ |
| $t_{pxz(BCLKH-DZ)}$ | Data Output Disable Time after BCLK | | | 5 | ns | ㉚ |
| $t_{d(A-RDL)}$ | Address Delay Time before Read | | $\frac{t_{c(BCLK)}}{2} -15$ | | ns | ㊴ |
| $t_{d(CS-RDL)}$ | Chip Select Delay Time before Read | | $\frac{t_{c(BCLK)}}{2} -15$ | | ns | ㊵ |
| $t_{v(RDH-A)}$ | Valid Address Time after Read | | 0 | | ns | ㊶ |
| $t_{v(RDH-CS)}$ | Valid Chip Select Time after Read | | 0 | | ns | ㊷ |
| $t_{pzx(RDH-DZ)}$ | Data Output Enable Time after Read | | $\frac{t_{c(BCLK)}}{2}$ | | ns | ㊻ |
| $t_{d(A-BLWL)}$<br>$t_{d(A-BHWL)}$ | Address Delay Time before Write (Byte write mode) | | $\frac{t_{c(BCLK)}}{2} -15$ | | ns | ㊼ |
| $t_{d(CS-BLWL)}$<br>$t_{d(CS-BHWL)}$ | Chip Select Delay Time before Write (Byte write mode) | | $\frac{t_{c(BCLK)}}{2} -15$ | | ns | ㊽ |
| $t_{v(BLWH-A)}$<br>$t_{v(BHWH-A)}$ | Valid Address Time after Write (Byte write mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | ㊾ |
| $t_{v(BLWH-CS)}$<br>$t_{v(BHWH-CS)}$ | Valid Chip Select Time after Write (Byte write mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | ㊿ |

### Read and write timing (continued from the preceding page)

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.7<br>21.3.8<br>21.3.9 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{d(BLWL-D)}$<br>$t_{d(BHWL-D)}$ | Data Output Delay Time after Write (Byte write mode) | | | 15 | ns | 52 |
| $t_{v(BLWH-D)}$<br>$t_{v(BHWH-D)}$ | Valid Data Output Time after Write (Byte write mode) | | $\frac{t_{c(BCLK)}}{2}-13$ | | ns | 53 |
| $t_{pxz(BLWH-DZ)}$<br>$t_{pxz(BHWH-DZ)}$ | Data Output Disable Time after Write (Byte write mode) | | | $\frac{t_{c(BCLK)}}{2}+5$ | ns | 54 |
| $t_{d(A-WRL)}$ | Address Delay Time before Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | 69 |
| $t_{d(CS-WRL)}$ | Chip Select Delay Time before Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | 70 |
| $t_{v(WRH-A)}$ | Valid Address Time after Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | 71 |
| $t_{v(WRH-CS)}$ | Valid Chip Select Time after Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | 72 |
| $t_{d(BLE-WRL)}$<br>$t_{d(BHE-WRL)}$ | Byte Enable Delay Time before Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | 73 |
| $t_{v(WRH-BLE)}$<br>$t_{v(WRH-BHE)}$ | Valid Byte Enable Time after Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-15$ | | ns | 74 |
| $t_{d(WRL-D)}$ | Data Output Delay Time after Write (Byte enable mode) | | | 15 | ns | 75 |
| $t_{v(WRH-D)}$ | Valid Data Output Time after Write (Byte enable mode) | | $\frac{t_{c(BCLK)}}{2}-13$ | | ns | 76 |
| $t_{pxz(WRH-DZ)}$ | Data Output Disable Time after Write (Byte enable mode) | | | $\frac{t_{c(BCLK)}}{2}+5$ | ns | 77 |
| $t_{w(RDH)}$ | Read High-level Pulse Width | | $\frac{t_{c(BCLK)}}{2}-3$ | | ns | 55 |

### (5) Bus arbitration

| Symbol | Parameter | Condition | Rated Value | | Unit | See Figure 21.3.10 |
|---|---|---|---|---|---|---|
| | | | MIN | MAX | | |
| $t_{d(BCLKL-HACKL)}$ | $\overline{HACK}$ Delay Time after BCLK | | | 29 | ns | 37 |
| $t_{v(BCLKL-HACKL)}$ | Valid $\overline{HACK}$ Time after BCLK | | -11 | | ns | 38 |

### 21.3.3 AC Characteristics

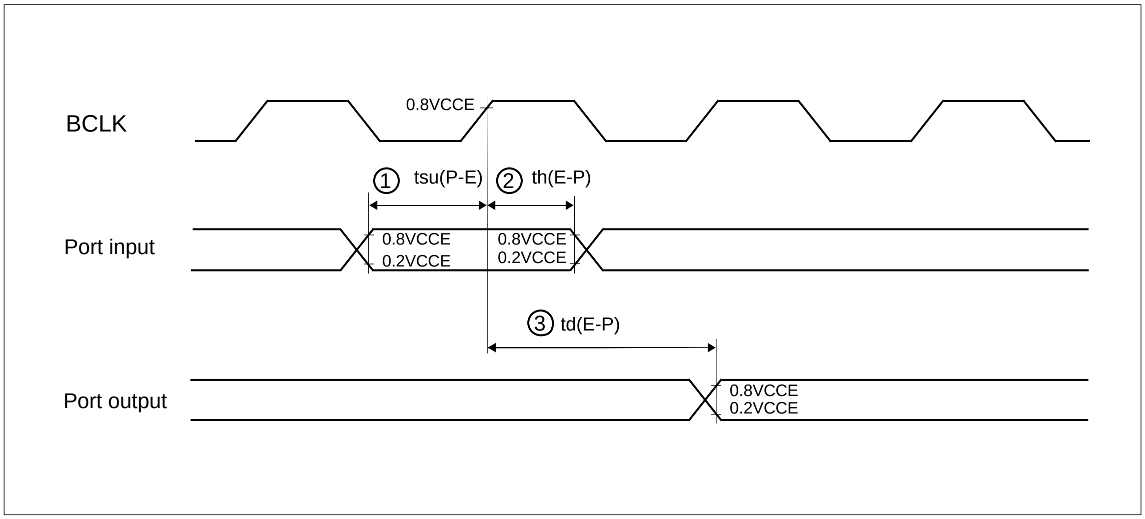


**Figure 21.3.1 Input/Output Port Timing**

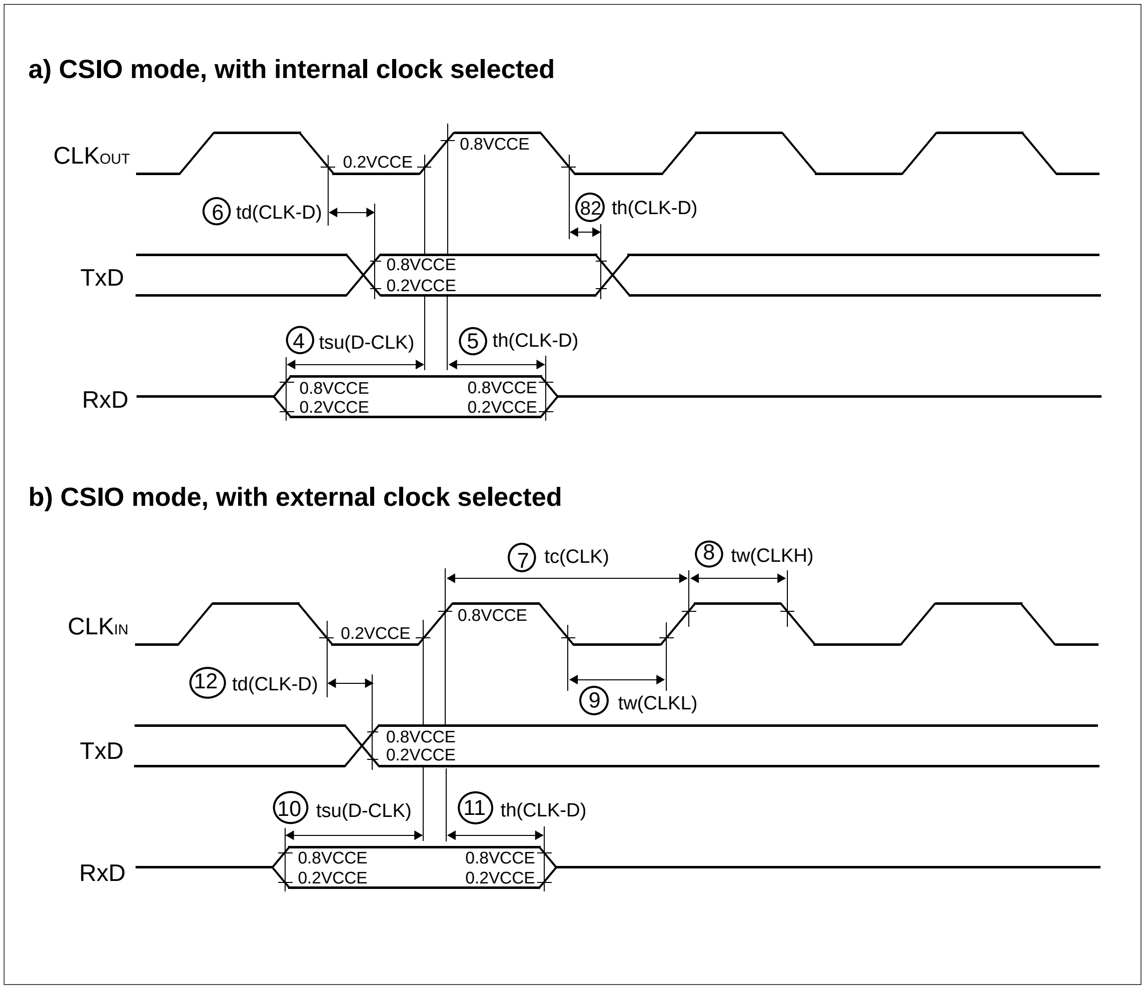


**Figure 21.3.2 Serial I/O Timing**

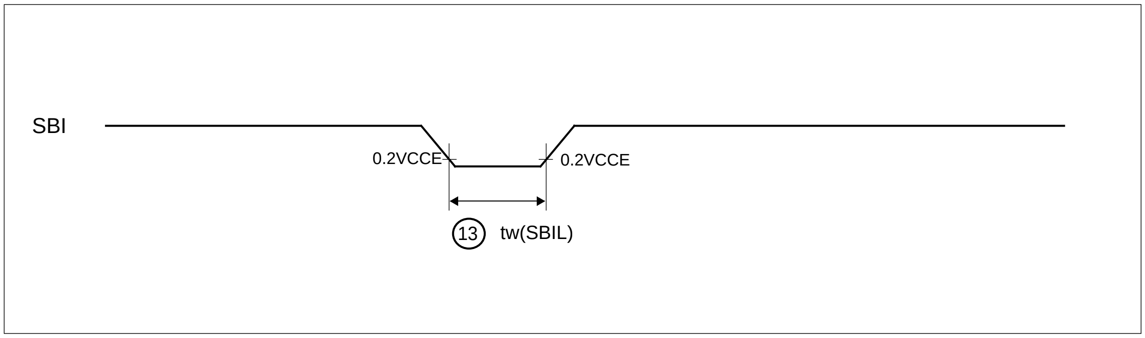


Figure 21.3.3 SBI Timing

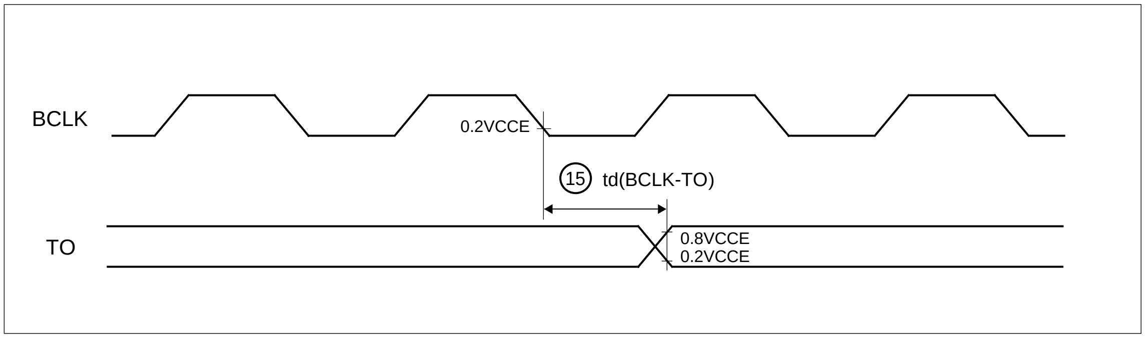


Figure 21.3.4 TO Timing

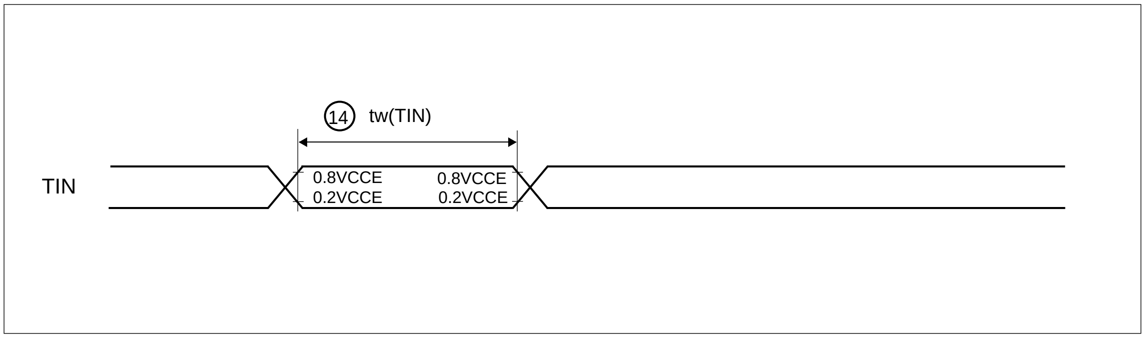


Figure 21.3.5 TIN Timing

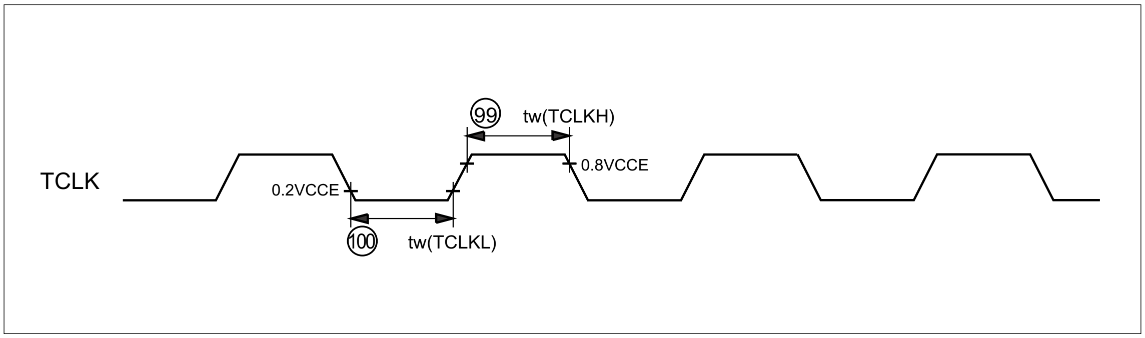
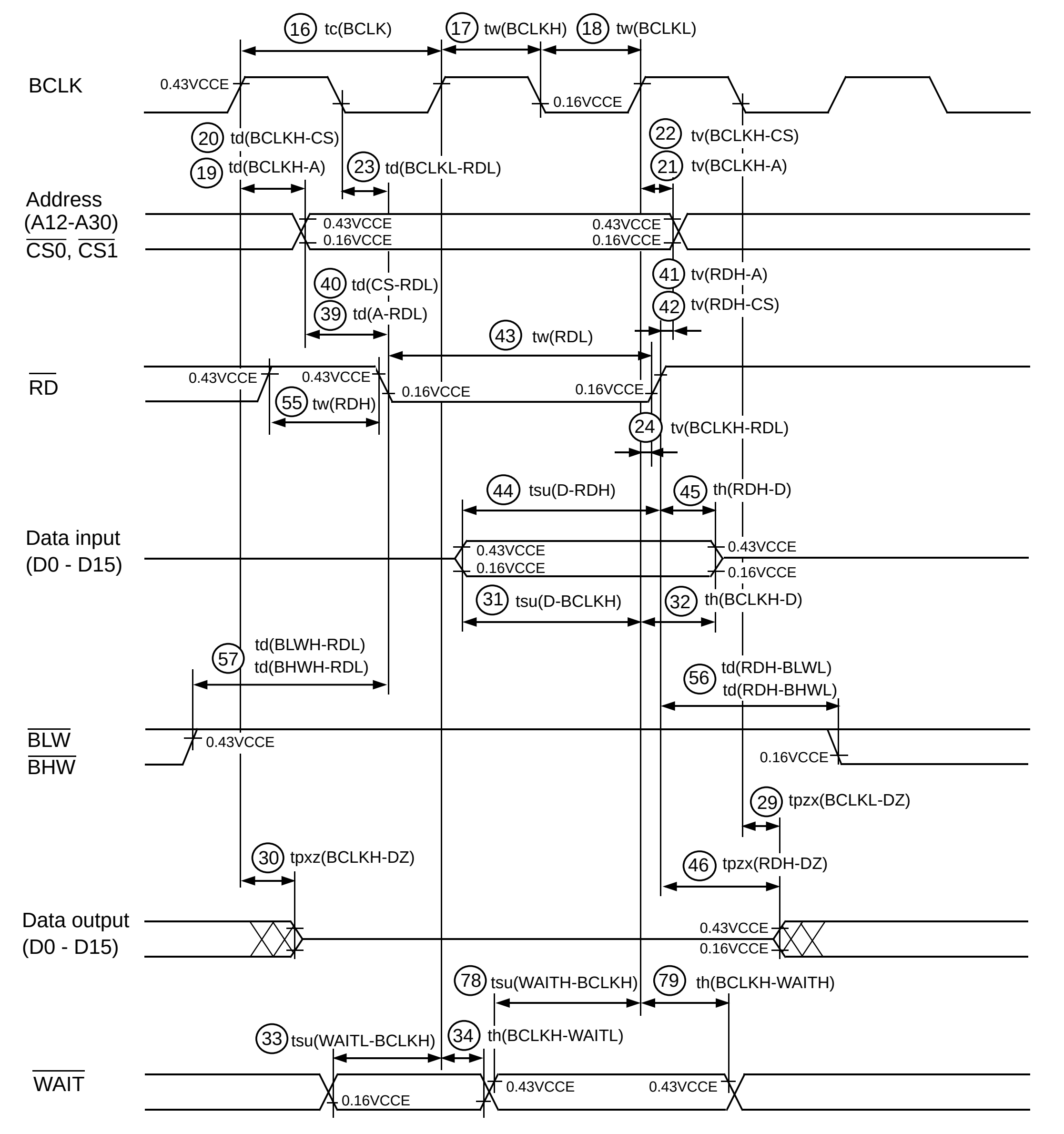


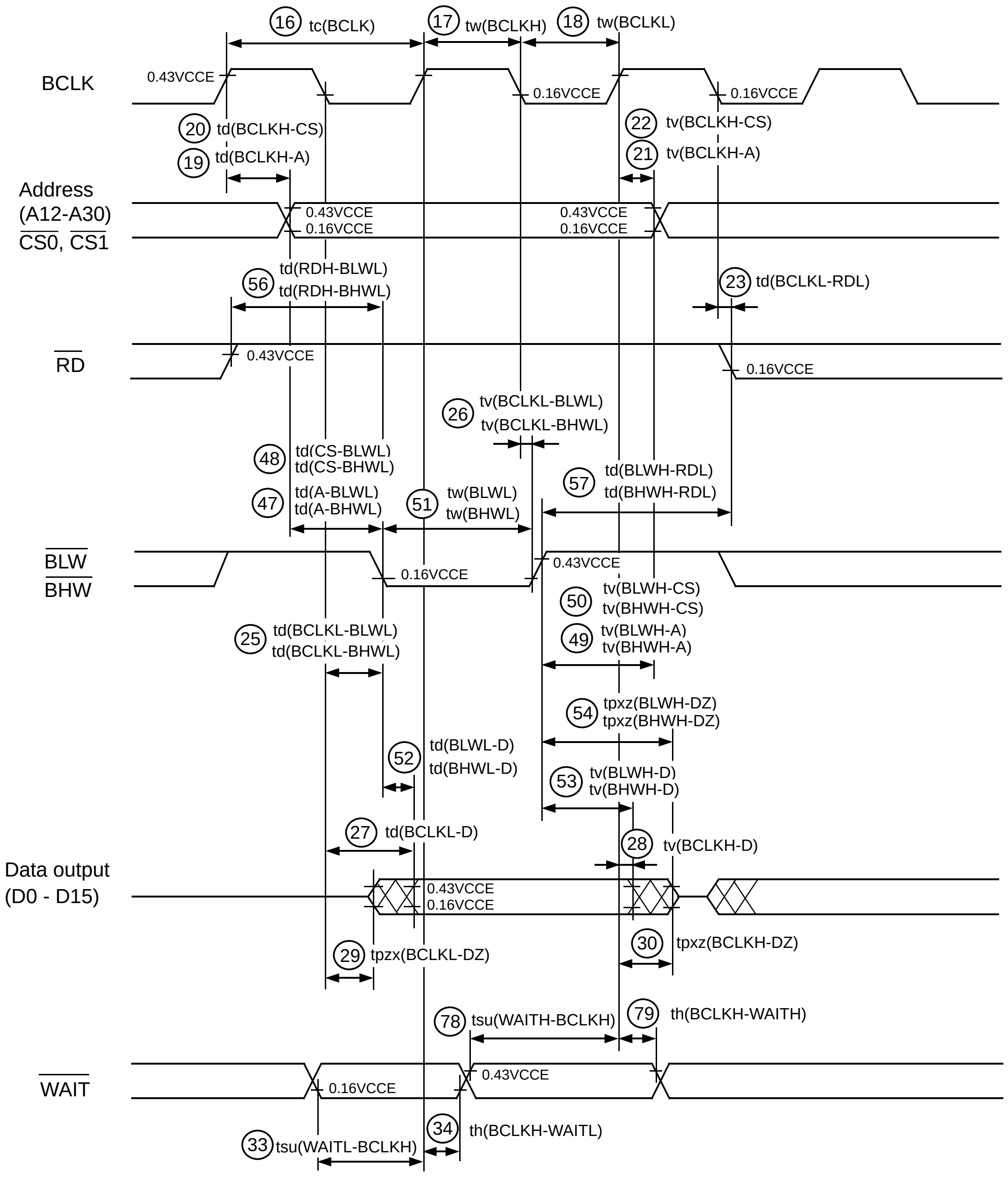
Figure 21.3.6 TCLK Timing



Notes: • Stipulated values are guaranteed values when the test pin load capacitance CL = 15 to 50 pF.

• Input and output signals are determined high or low with respect to TTL level.

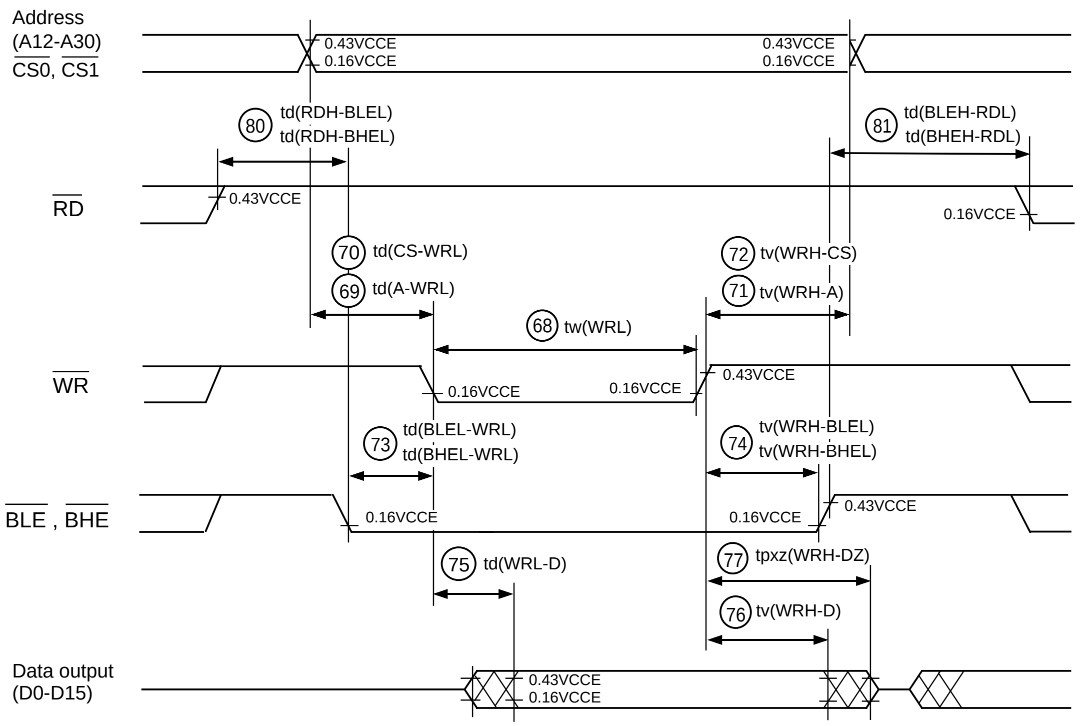
**Figure 21.3.7 Read Timing**



Notes: • Stipulated values are guaranteed values when the test pin load capacitance CL = 15 to 50 pF.

• Input and output signals are determined high or low with respect to TTL level.

**Figure 21.3.8 Write Timing**



Notes: • Stipulated values are guaranteed values when the test pin load capacitance CL = 15 to 50 pF.

• Input and output signals are determined high or low with respect to TTL level.

**Figure 21.3.9 Write Timing (Byte enable mode)**

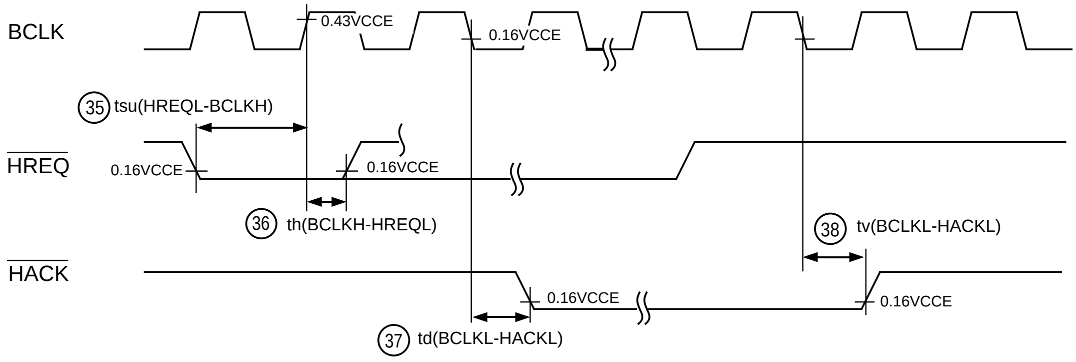
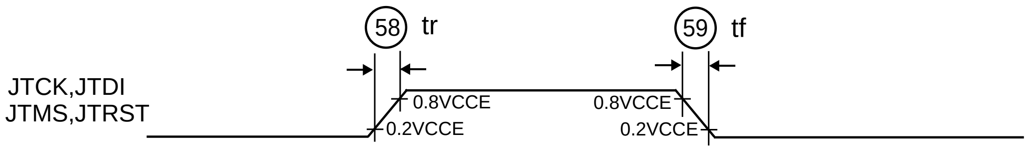
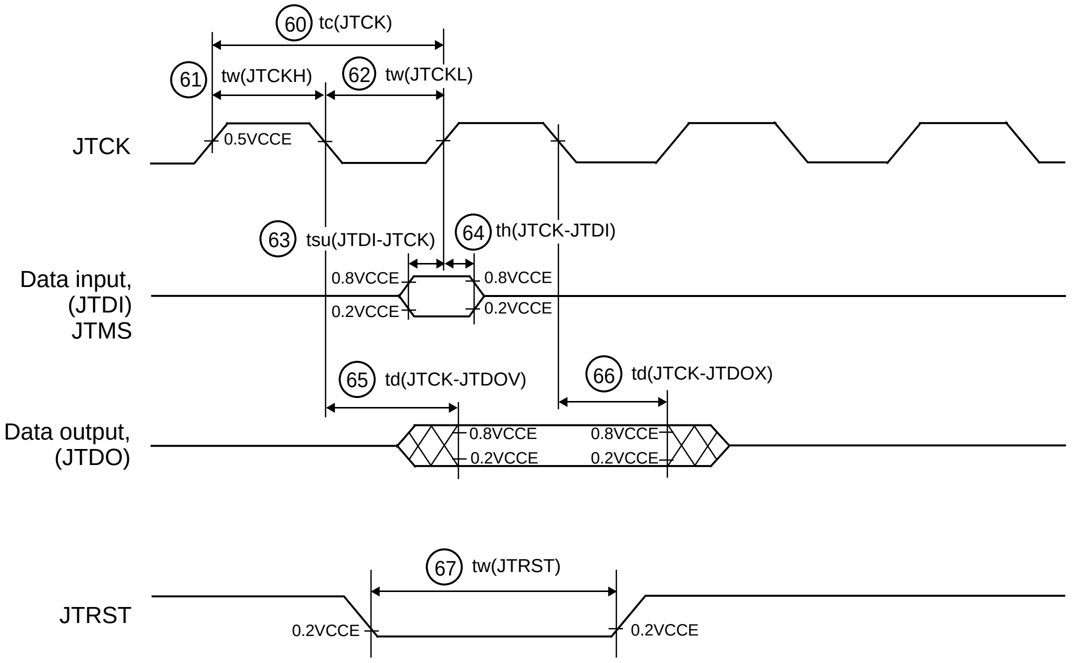


**Figure 21.3.10 Bus Arbitration Timing**



Note: • Stipulated values are guaranteed values when the test pin load capacitance CL = 80 pF.

**Figure 21.3.11 Input Transition Time on JTAG pins**



Note: • Stipulated values are guaranteed values when the test pin load capacitance CL = 80 pF.

**Figure 21.3.12 JTAG Interface Timing**

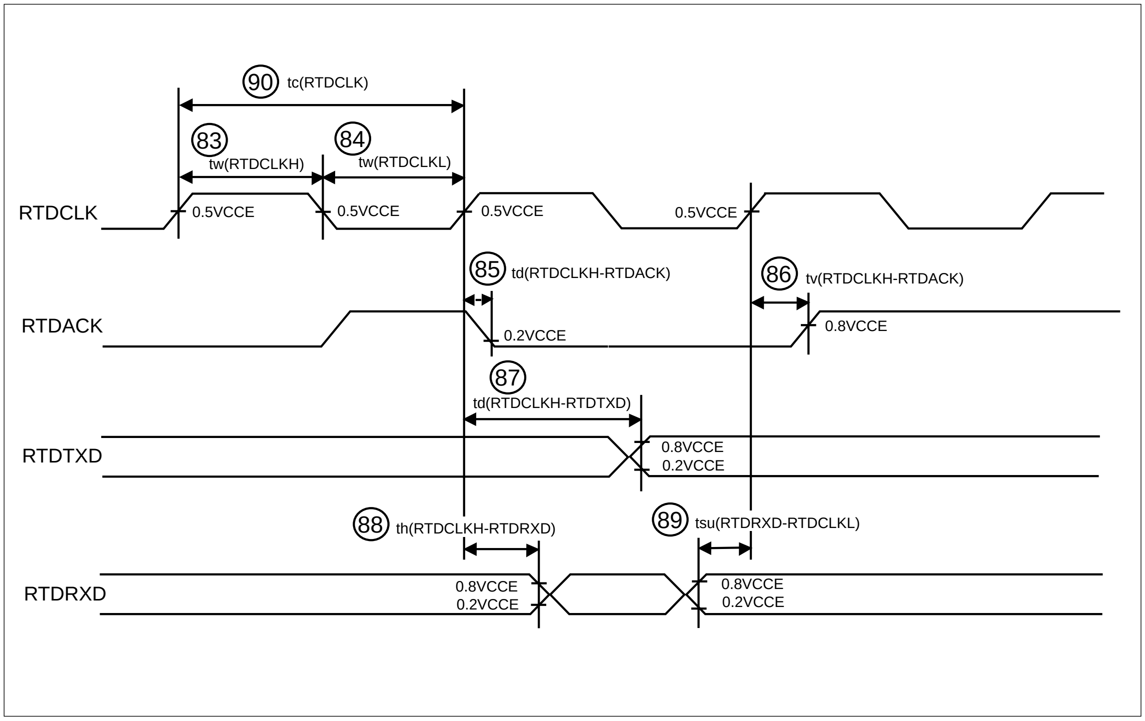


**Figure 21.3.13 RTD Timing**

* This is a blank page.*

# CHAPTER 22

# TYPICAL CHARACTERISTICS

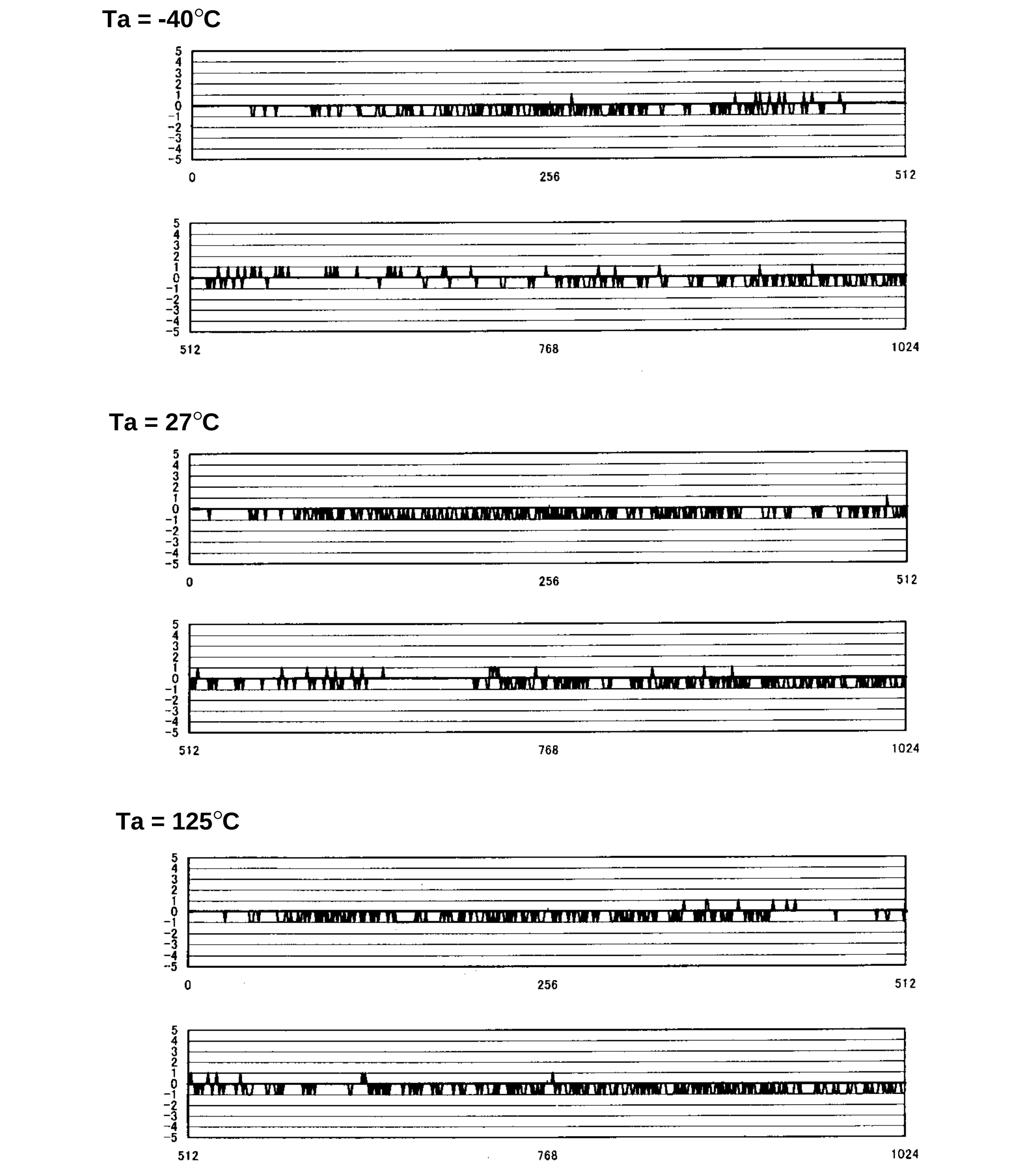
22.1 A-D Conversion Characteristics

## 22.1 A-D Conversion Characteristics

### (1) Test conditions

- Ta = -40°C, 27°C, 125°C
- Test voltage (VCC) = 5.12 V
- Normal mode, Double-speed mode

### (2) Measured value (Reference value)



Vertical axis : Conversion error (LSB)

Horizontal axis : Analog input voltage ( 5.12 × N/1024 [V] )

# APPENDIX 1

# MECHANICAL SPECIFICATIONS

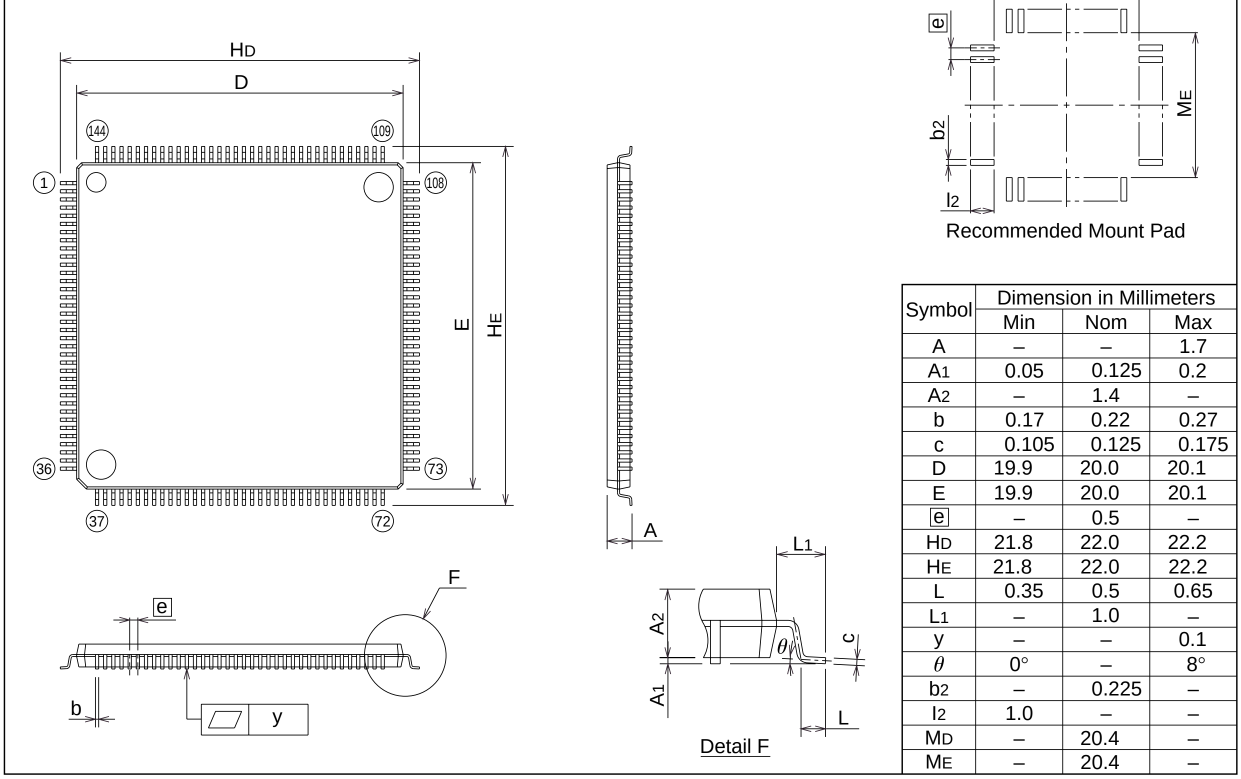
Appendix 1.1 Dimensional Outline Drawing

## Appendix 1.1 Dimensional Outline Drawing

### (1) 144 pin LQFP

**144P6Q-A** **Plastic 144pin 20×20mm body LQFP**

| EIAJ Package Code | JEDEC Code | Weight(g) | Lead Material |
|---|---|---|---|
| LQFP144-P-2020-0.50 | – | | Cu Alloy |



| Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| A | – | – | 1.7 |
| $A_1$ | 0.05 | 0.125 | 0.2 |
| $A_2$ | – | 1.4 | – |
| b | 0.17 | 0.22 | 0.27 |
| c | 0.105 | 0.125 | 0.175 |
| D | 19.9 | 20.0 | 20.1 |
| E | 19.9 | 20.0 | 20.1 |
| $\boxed{e}$ | – | 0.5 | – |
| $H_D$ | 21.8 | 22.0 | 22.2 |
| $H_E$ | 21.8 | 22.0 | 22.2 |
| L | 0.35 | 0.5 | 0.65 |
| $L_1$ | – | 1.0 | – |
| y | – | – | 0.1 |
| $\theta$ | 0° | – | 8° |
| $b_2$ | – | 0.225 | – |
| $l_2$ | 1.0 | – | – |
| $M_D$ | – | 20.4 | – |
| $M_E$ | – | 20.4 | – |

# APPENDIX 2

# INSTRUCTION PROCESSING TIME

Appendix 2.1 M32R/ECU Instruction Processing Time

## Appendix 2.1 M32R/ECU Instruction Processing Time

For the M32R/ECU, the number of instruction execution cycles in E stage normally represents its instruction processing time. However, depending on pipeline operation, other stages may affect the instruction processing time. Especially when a branch instruction is executed, the processing time in the IF (instruction fetch), D (decode) and E (execution) stages of the next instruction must also be taken into account.
The table below shows the instruction processing time in each pipelined stage of the M32R/ECU.

**Table 2.1.1 Instruction Processing Time of Each Pipeline Stage**

| | **Number of execution cycles in each stage** (Note 1) | | | | |
|---|---|---|---|---|---|
| **Instruction** | **IF** | **D** | **E** | **MEM** | **WB** |
| Load instructions (LD, LDB, LDUB, LDH, LDUH, LOCK) | R | 1 | 1 | R | 1 |
| Store instructions (ST,STB,STH,UNLOCK) | R | 1 | 1 | W | - |
| Multiply instruction (MUL) | R | 1 | 3 | - | 1 |
| Divide/remainder instructions (DIV, DIVU,REM,REMU) | R | 1 | 37 | - | 1 |
| Other instructions (including those for DSP function) | R | 1 | 1 | - | 1 |

Note 1: For R and W, refer to the calculation methods described in the next page.

The following shows the number of memory access cycles in IF and MEM stages. Shown here are the minimum number of cycles required for memory access. Therefore, these values do not always reflect the number of cycles required for actual memory or bus access.

In write access, for example, although the CPU finishes the MEM stage by only writing to the write buffer, this operation actually is followed by a write to memory. Depending on the memory or bus state before or after the CPU requested a memory access, the instruction processing may take more time than the calculated value.

| ■ **R (read cycle)** | **Cycles** |
|---|---|
| When existing in instruction queue | 1 |
| When reading internal resource (ROM, RAM) | 1 |
| When reading internal resource (SFR)(byte, halfword) | 2 |
| When reading internal resource (SFR)(word) | 4 |
| When reading external memory (byte, halfword) | 5 (Note 1) |
| When reading external memory (word) | 9 (Note 1) |
| When successively fetching instructions from external memory | 8 (Note 1) |

| ■ **W (write cycle)** | **Cycles** |
|---|---|
| When writing to internal resource (RAM) | 1 |
| When writing to internal resource (SFR)(byte, halfword) | 2 |
| When writing to internal resource (SFR)(word) | 4 |
| When writing to external memory (byte, halfword) | 4 (Note 1) |
| When writing to external memory (word) | 8 (Note 1) |

Note 1: This applies for external access with one wait cycle. (When the M32R/ECU accesses external circuits, it requires at least one wait cycle inserted.)

❊ This is a blank page. ❊

# APPENDIX 3

# PROCESSING OF UNUSED PINS

Appendix 3.1 Example for Processing Unused Pins

## Appendix 3.1 Example for Processing Unused Pins

An example for processing unused pins is shown below.

### (1) When operating in single-chip mode

**Table A3.1.1 Example for Processing Unused Pins when Operating in Single-chip Mode**

| Pin name | Processing |
|---|---|
| Input/output ports (Note 1)<br>P00-P07, P10-P17, P20-P27,<br>P30-P37, P41-P47, P61-P63,<br>P70-P77, P82-P87, P93-P97,<br>P100-P107, P110-P117, P124-P127,<br>P130-P137, P150, P153, P174, P175,<br>P220, P221, P225 (Note 2) | Set these pins for input mode and connect them to VSS via 1 kΩ to 10 kΩ resistors (pulldown). |
| P64 / $\overline{\text{SBI}}$ (Note 3) | Connect this pin to VSS (pulldown) via a 1 to 10 kΩ resistor. |
| XOUT (Note 4) | Leave these pins open. |
| A-D converter | |
| AD0IN0-AD0IN15, AVREF0, AVSS0 | Connect these pins to VSS. |
| AVCC0 | Connect this pin to VCCE. |
| JTAG | |
| JTDO, JTMS, JTDI, JTCK | Connect these pins to VCCE (pullup) or VSS (pulldown) via 0 to 100 kΩ resistors. |
| JTRST | Connect this pin to VSS (pulldown) via a 0 to 100 kΩ resistor. |

Note 1: After exiting reset, the input/output ports are set for input by default.

Note 2: P221 is used exclusively for CAN input.

Note 3: P64 is used exclusively for $\overline{\text{SBI}}$ input. Make sure that unintended falling edges due to noise, etc. will not be applied. (A falling edge at P64/$\overline{\text{SBI}}$ pin causes a system break interrupt to occur).

Note 4: This applies when an external clock is fed to XIN.

### (2) When operating in external extension mode or processor mode

**Table A3.1.2 Example for Processing Unused Pins when Operating in External Extension or Processor Mode**

| Pin name | | Processing |
|---|---|---|
| Input/output ports (Note 1)<br>P61-P63, P70-P77, P82-P87,<br>P93-P97, P100-P107, P110-P117,<br>P124-P127, P130-P137, P150, P153,<br>P174, P175, P220, P221, P225 (Note 2) | | Set these pins for input mode and connect them to VSS via 1 kΩ to 10 kΩ resistors (pulldown). |
| P64 / $\overline{\text{SBI}}$ (Note 3) | | Connect this pin to VSS (pulldown) via a 1 to 10 kΩ resistor. |
| $\overline{\text{BLW}}/\overline{\text{BLE}}$, $\overline{\text{BHW}}/\overline{\text{BHE}}$, $\overline{\text{CS1}}$ | | Leave these pins open. |
| XOUT (Note 4) | | Leave these pins open. |
| A-D converter | | |
| | AD0IN0-AD0IN15, AVREF0, AVSS0 | Connect these pins to VSS. |
| | AVCC0 | Connect these pins to VCCE. |
| JTAG | | |
| | JTDO, JTMS, JTDI, JTCK | Connect these pins to VCCE (pullup) or VSS (pulldown) via 0 to 100 kΩ resistors. |
| | JTRST | Connect this pin to VSS (pulldown) via a 0 to 100 kΩ resistor. |

Note 1: After exiting reset, the input/output ports are set for input by default.

Note 2: P221 is used exclusively for CAN input.

Note 3: P64 is used exclusively for $\overline{\text{SBI}}$ input. Make sure that unintended falling edges due to noise, etc. will not be applied. (A falling edge at P64/$\overline{\text{SBI}}$ pin causes a system break interrupt to occur).

Note 4: This applies when an external clock is fed to XIN.

* This is a blank page. *

# APPENDIX 4

# SUMMARY OF PRECAUTIONS

Appendix 4.1 Precautions Regarding the CPU

Appendix 4.2 Precautions on Address Space

Appendix 4.3 Precautions on EIT

Appendix 4.4 Precautions to Be Taken When Reprogramming Flash Memory

Appendix 4.5 Things To Be Considered after Exiting Reset

Appendix 4.6 Precautions on Input/output Ports

Appendix 4.7 Precautions about the DMAC

Appendix 4.8 Precautions on Multijunction Timers

Appendix 4.9 Precautions on Using A-D Converters

Appendix 4.10 Precautions on Serial I/O

Appendix 4.11 Precautions on RAM Backup Mode

Appendix 4.12 Precautions on Processing JTAG Pins

Appendix 4.13 Precautions about Noise

## Appendix 4.1 Precautions Regarding the CPU

### Appendix 4.1.1 Things to be noted for data transfer

Note that in data transfer, data arrangements in registers and those in memory are different.

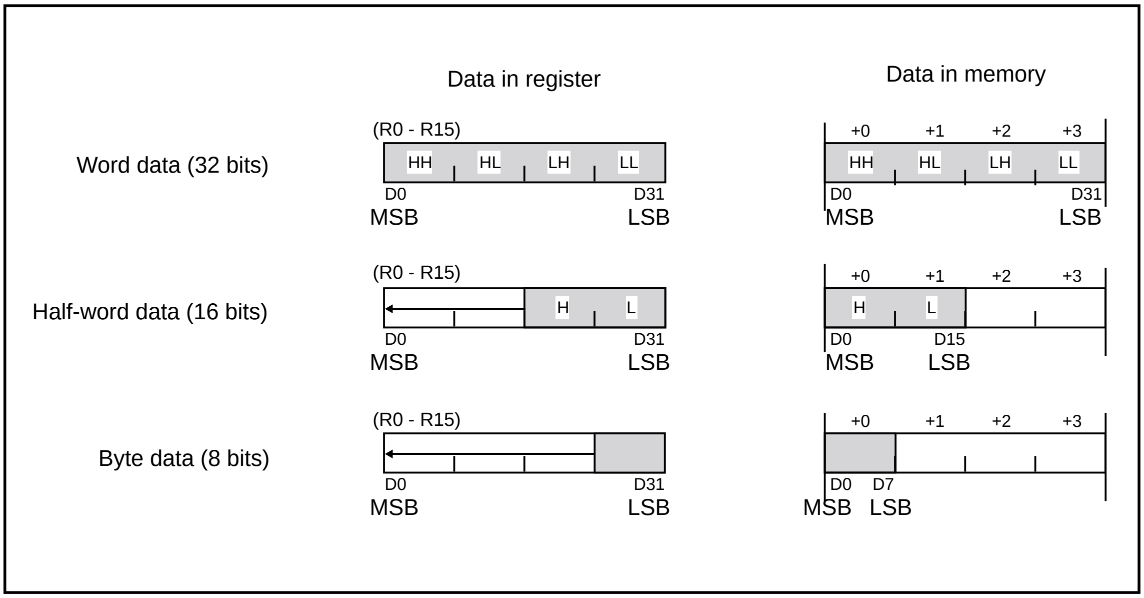


**Figure A4.1.1 Difference in Data Arrangements**

## Appendix 4.2 Precautions on Address Space

### Appendix 4.2.1 Virtual flash emulation function

The 32171 can map one 8-Kbyte block of internal RAM beginning with the start address into one of 8-Kbyte areas (L banks) of the internal flash memory and can map up to two 4-Kbyte blocks of internal RAM beginning with address H’0080 6000 into one of 4-Kbyte areas (S banks) of the internal flash memory. This capability is referred to as the “virtual-flash emulation” function. For details about this function, refer to Section 6.7, “Virtual-Flash Emulation Function.”

## Appendix 4.3 Precautions on EIT

Address Exception requires caution because when an address exception occurs pursuant to execution of an instruction (one of the following three) that uses the “register indirect + register update” addressing mode, the value of the automatically updated register (Rsrc or Rsrc2) becomes indeterminate.
Except that the values of Rsrc and Rsrc2 are indeterminate, the behavior is the same as when using other addressing modes.

- **Applicable instructions**

```
LD      Rdest,  @Rsrc+
ST      Rsrc1,  @-Rsrc2
ST      Rsrc1,  @+Rsrc2
```

If the above applies, because the register value becomes indeterminate as explained, consideration must be taken before continuing with system processing. (If an address exception occurs, it means that some fatal fault already occurred in the system at that point in time. Therefore, use EIT on condition that after processing by the address exception handler, the CPU will not return to the program it was executing when the exception occurred.)

## Appendix 4.4 Precautions to Be Taken When Reprogramming Flash Memory

The following describes precautions to be taken when you reprogram the flash memory using a general-purpose serial programmer in Boot Flash E/W Enable mode.

- When reprogramming the flash memory, a high voltage is generated inside the chip. Because this high voltage could cause the chip to break down, be careful about mode pin and power supply management not to move from one mode to another while reprogramming.
- If the system uses any pin that is to be used by a general-purpose reprogramming tool, take appropriate measures to prevent adverse effects when connecting the tool.
- If flash memory protection is needed when using a general-purpose reprogramming tool, set any ID in the flash memory protect ID check area (H’0000 0084–H’0000 0093).
- If flash memory protection is not needed when using a general-purpose reprogramming tool, set H’FF in the entire flash memory protect ID check area (H’0000 0084–H’0000 0093).
- Before using a reset by Flash Control Register 4 (FCNT4)’s FRESET bit to clear each error status in Flash Status Register 2 (FSTAT2) (initialized to H’80), check to see that Flash Status Register 1 (FSTAT1)’s FSTAT bit = 1 (Ready).

- Before changing Flash Control Register 1 (FCNT1)’s FENTRY bit from 1 to 0, check to see that Flash Status Register 1 (FSTAT1)’s FSTAT bit = 1 (Ready) or Flash Status Register 2 (FSTAT2)’s FBUSY bit = 1 (Ready).
- If Flash Control Register 1 (FCNT1)’s FENTRY bit = 1 and Flash Status Register 1 (FSTAT1)’s FSTAT bit = 0 (Busy) or Flash Status Register 2 (FSTAT2)’s FBUSY bit = 0 (Busy), do not clear the FENTRY bit.

## Appendix 4.5 Things To Be Considered after Exiting Reset

### Appendix 4.5.1 Input/output ports

After exiting reset, the 32171's input/output ports are disabled against input in order to prevent current from flowing through the port. To use any ports in input mode, enable them for input using the Port Input Function Enable Register (PIEN) PIEN0 bit. For details, refer to Section 8.3, "Input/ Output Port Related Registers."

## Appendix 4.6 Precautions on Input/output Ports

### Appendix 4.6.1 When using the ports in output mode

Because the Port Data Register values immediately after reset are indeterminate, it is necessary that the initial value be written to the Port Data Register before setting the Port Direction Register for output. Conversely, if the Port Direction Register is set for output before writing to the Port Data Register, indeterminate values will be output for a while until the initial value is set in the Port Data Register.

## Appendix 4.7 Precautions about the DMAC

### Appendix 4.7.1 About writing to DMAC related registers

Because DMA transfer involves exchanging data via the internal bus, basically you only can write to the DMAC related registers immediately after reset or when transfer is disabled (transfer enable bit = 0). When transfer is enabled, do not write to the DMAC related registers because write operation to those registers, except the DMA transfer enable bit, transfer request flag, and the DMA Transfer Count Register which is protected in hardware, is instable.

The table below shows the registers that can or cannot be accessed for write.

**Table A4.7.1 DMAC Related Registers That Can or Cannot Be Accessed for Write**

| Status | Transfer enable bit | Transfer request flag | Other DMAC related registers |
|---|---|---|---|
| When transfer is enabled | ○ | ○ | ✕ |
| When transfer is disabled | ○ | ○ | ○ |

○ : Can be accessed ; ✕ : Cannot be accessed

For even registers that can exceptionally be written to while transfer is enabled, the following requirements must be met.

(1) DMA Channel Control Register's transfer enable bit and transfer request flag

For all other bits of the channel control register, be sure to write the same data that those bits had before you wrote to the transfer enable bit or transfer request flag. Note that you only can write a 0 to the transfer request flag as valid data.

(2) DMA Transfer Count Register

When transfer is enabled, this register is protected in hardware, so that any data you write to this register is ignored.

(3) Rewriting the DMA source and DMA destination addresses on different channels by DMA transfer

In this case, you are writing to the DMAC related registers while DMA is enabled, but this practically does not present any problem. However, you cannot DMA-transfer to the DMAC related registers on the local channel itself in which you are currently operating.

## Appendix 4.7.2 Manipulating DMAC related registers by DMA transfer

When manipulating DMAC related registers by means of DMA transfer (e.g., reloading the DMAC related registers' initial values by DMA transfer), do not write to the DMAC related registers on the local channel itself through that channel. (If this precaution is neglected, device operation cannot be guaranteed.)

Only if residing on other channels, you can write to the DMAC related registers by means of DMA transfer. (For example, you can rewrite the DMAn Source Address and DMAn Destination Address Registers on channel 1 by DMA transfer through channel 0.)

## Appendix 4.7.3 About the DMA Interrupt Request Status Register

When clearing the DMA Interrupt Request Status Register, be sure to write 1s to all bits but the one you want to clear. The bits to which you wrote 1s retain the previous data they had before the write.

## Appendix 4.7.4 About the stable operation of DMA transfer

To ensure the stable operation of DMA transfer, never rewrite the DMAC related registers, except the DMA Channel Control Register's transfer enable bit, unless transfer is disabled. One exception is that even when transfer is enabled, you can rewrite the DMA Source Address and DMA Destination Address Registers by DMA transfer from one channel to another.

## Appendix 4.8 Precautions on Multijunction Timers

### Appendix 4.8.1 Precautions to be observed when using TOP single-shot output mode

The following describes precautions to be observed when using TOP single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).
- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).
- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.

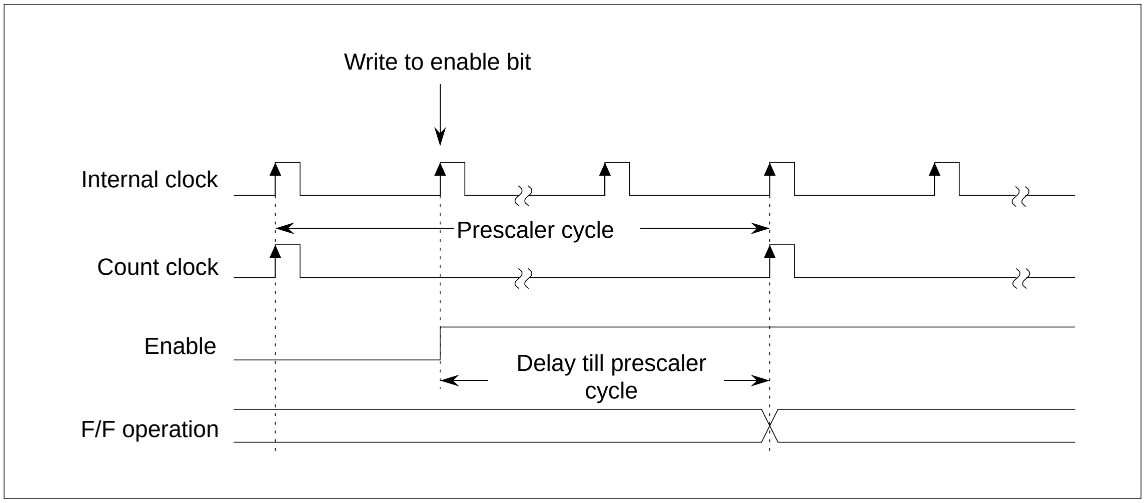
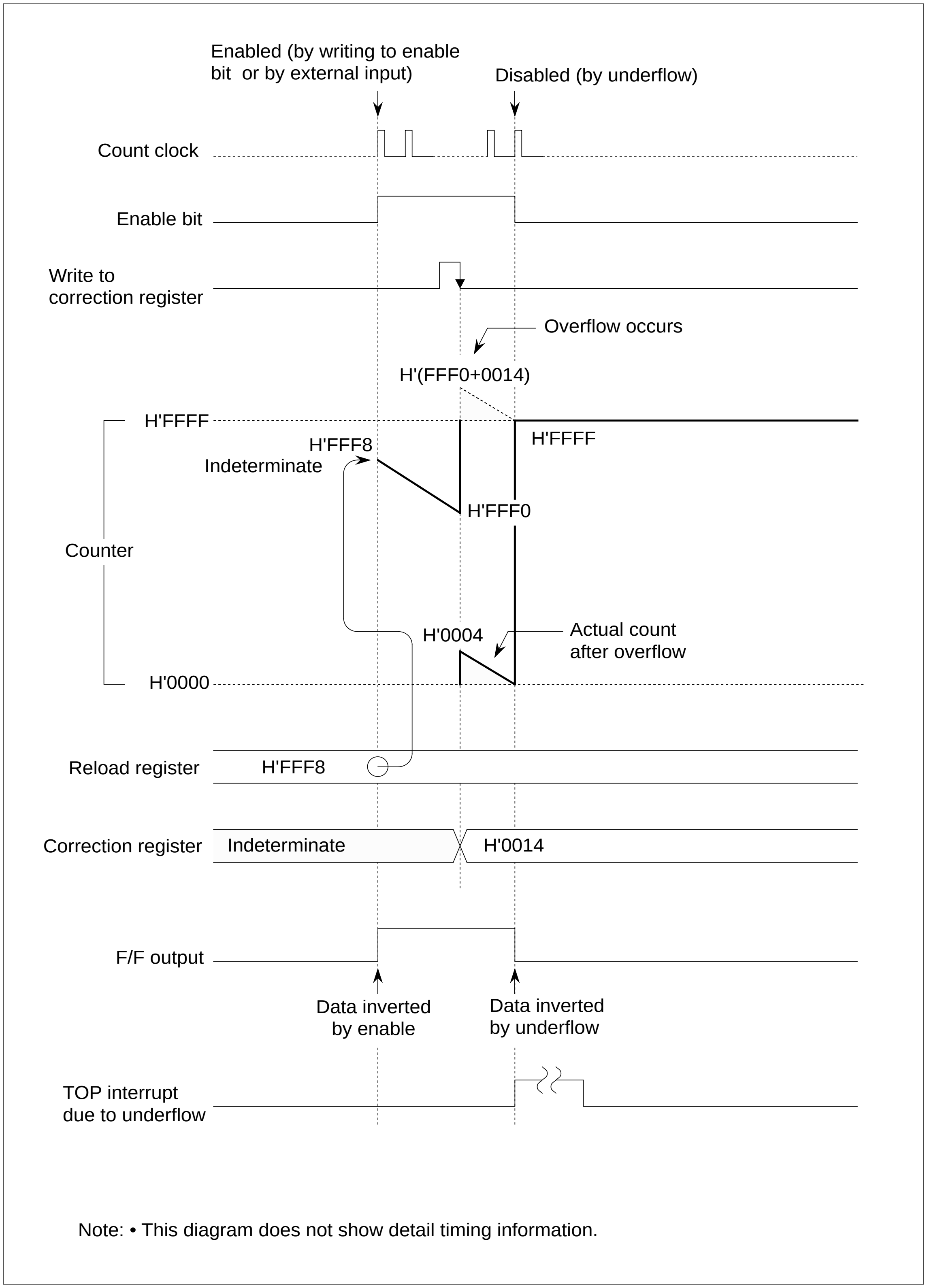


**Figure A4.8.1 Prescaler Delay**

- When writing to the correction register, be careful not to cause the counter to overflow. Even when the counter overflows due to correction of counts, no interrupt is generated for the occurrence of overflow. When the counter underflows in the subsequent down-count after overflow, a false underflow interrupt is generated due to overcounting.

In the example below, the reload register has the initial value H'FFF8 set in it. When the timer starts, the reload register value is loaded into the counter causing it to start counting down. In the example diagram here, H'0014 is written to the correction register when the counter has counted down to H'FFF0. As a result of this correction, the count overflows to H'0004 and fails to count correctly. Also, an interrupt is generated for an erroneous overcount.



Note: • This diagram does not show detail timing information.

**Figure A4.8.2 Example of Operation in TOP Single-shot Output Mode Where Count Overflows due to Correction**

## Appendix 4.8.2 Precautions to be observed when using TOP delayed single-shot output mode

The following describes precautions to be observed when using TOP delayed single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).
- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).
- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- Even when the counter overflows due to correction of counts, no interrupt is generated for the occurrence of overflow. When the counter underflows in the subsequent down-count after overflow, a false underflow interrupt is generated due to overcounting.
- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.

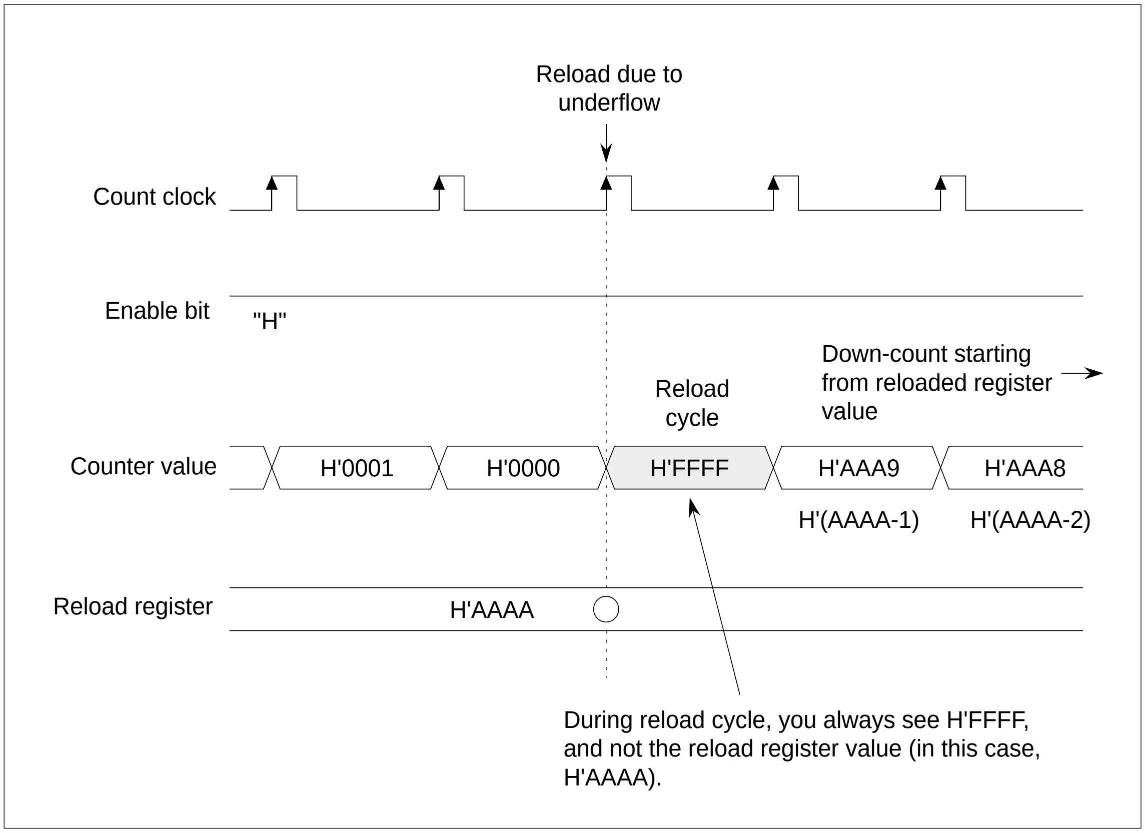

Reload due to underflow
Count clock
Enable bit
"H"
Reload cycle
Down-count starting from reloaded register value
Counter value
H'0001
H'0000
H'FFFF
H'AAA9
H'AAA8
H'(AAAA-1)
H'(AAAA-2)
Reload register
H'AAAA
During reload cycle, you always see H'FFFF, and not the reload register value (in this case, H'AAAA).

**Figure A4.8.3 Counter Value Immediately after Underflow**

## Appendix 4.8.3 Precautions to be observed when using TOP continuous output mode

The following describes precautions to be observed when using TOP continuous output mode.

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.
- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.

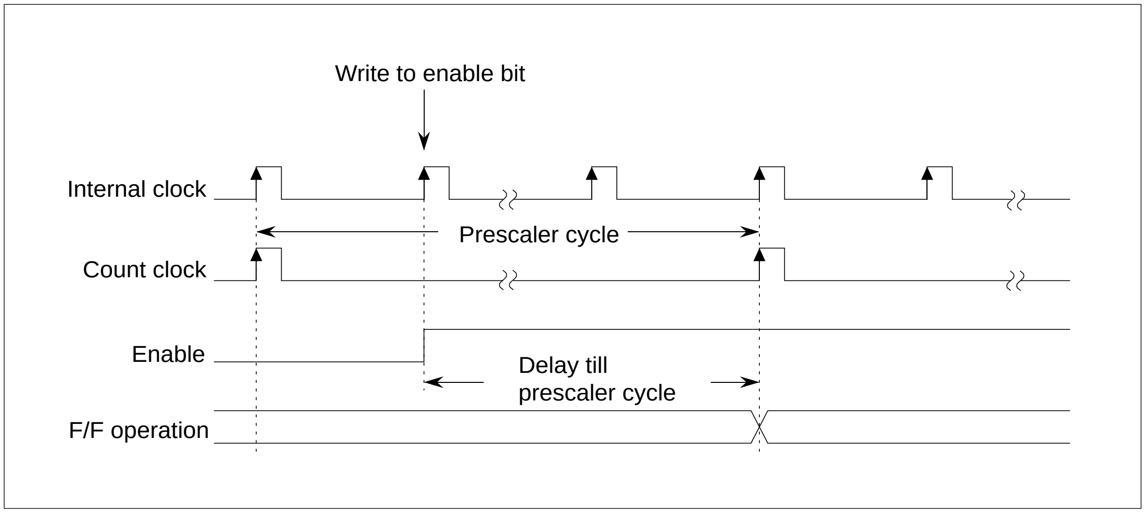


**Figure A4.8.4 Prescaler Delay**

### Appendix 4.8.4 Precautions to be observed when using TIO measure free-run/clear input modes

The following describes precautions to be observed when using TIO measure free-run/clear input modes.

- If measure event input and write to the counter occur simultaneously in the same clock period, the write value is set in the counter while at the same time latched into the measure register.

### Appendix 4.8.5 Precautions to be observed when using TIO single-shot output mode

The following describes precautions to be observed when using TIO single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).

- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).

- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.

### Appendix 4.8.6 Precautions to be observed when using TIO delayed single-shot output mode

The following describes precautions to be observed when using TIO delayed single-shot output mode.

- If the counter stops due to underflow in the same clock period as the timer is enabled by external input, the former has priority (so that the counter stops).

- If the counter stops due to underflow in the same clock period as count is enabled by writing to the enable bit, the latter has priority (so that count is enabled).

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).

- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.

- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite equal to a prescaler delay is included before F/F starts operating after the timer is enabled.

## Appendix 4.8.7 Precautions to be observed when using TIO continuous output mode

The following describes precautions to be observed when using TIO continuous output mode.

- If the timer is enabled by external input in the same clock period as count is disabled by writing to the enable bit, the latter has priority (so that count is disabled).
- When you read the counter immediately after reloading it pursuant to underflow, the value you get is temporarily H'FFFF. But this counter value immediately changes to (reload value - 1) at the next clock edge.
- Because the internal circuit operation is synchronized to the count clock (prescaler output), a finite time equal to a prescaler delay is included before F/F starts operating after the timer is enabled.

## Appendix 4.8.8 Precautions to be observed when using TMS measure input

The following describes precautions to be observed when using TMS measure input.

- If measure event input and write to the counter occur simultaneously in the same clock period, the write value is set in the counter while at the same time latched to the measure register.

### Appendix 4.8.9 Precautions to be observed when using TML measure input

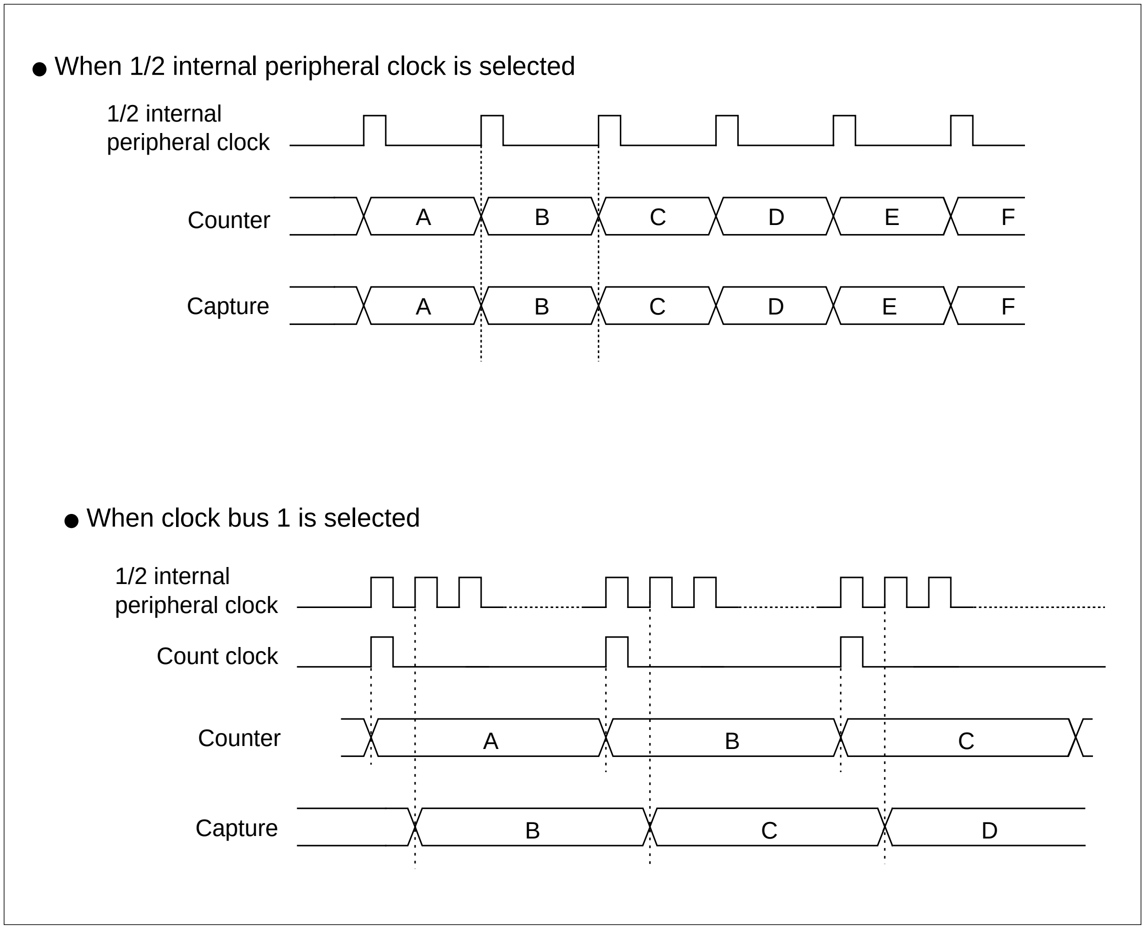
The following describes precautions to be observed when using TML measure input.

- If measure event input and write to the counter occur simultaneously in the same clock period, the write value is set in the counter, whereas the up-count value (before being rewritten) is latched to the measure register.

- If the timer operates with any clock other than the 1/2 internal peripheral clock while clock bus 1 is selected for the count clock, the counter cannot be written normally. Therefore, when operating with any clock other than the 1/2 internal peripheral clock, do not write to the counter.

- If the timer operates with any clock other than the 1/2 internal peripheral clock while clock bus 1 is selected for the count clock, the captured value is one that leads the actual counter value by one clock period. However, during the 1/2 internal peripheral clock interval from the count clock, this problem does not occur and the counter value is captured at exact timing.

The diagram below shows the relationship between counter operation and the valid data that can be captured.



● When 1/2 internal peripheral clock is selected

● When clock bus 1 is selected

**Figure A4.8.5 Mistimed Counter Value and Captured Value**

## Appendix 4.9 Precautions on Using A-D Converters

- Forcible termination during scan operation

  If A-D conversion is forcibly terminated by setting the A-D conversion stop bit (AD0CSTP) to 1 during scan mode operation and you read the content of the A-D data register for the channel in which conversion was in progress, it shows the last conversion result that had been transferred to the A-D data register before the conversion was forcibly terminated.

- Modification of A-D converter related registers

  If you want to change the contents of the A-D Conversion Interrupt Control Register, each Single and Scan Mode Register, or A-D Successive Approximation Register, except for the A-D conversion stop bit, do your change while A-D conversion is inactive, or be sure to restart A-D conversion after you changed the register contents. If the contents of these registers are changed in the middle of A-D conversion, the conversion results cannot be guaranteed.

- Handling of analog input signals

  The A-D converters included in the 32171 do not have a sample-and-hold circuit. Therefore, make sure the analog input levels are fixed during A-D conversion.

- A-D conversion completion bit readout timing

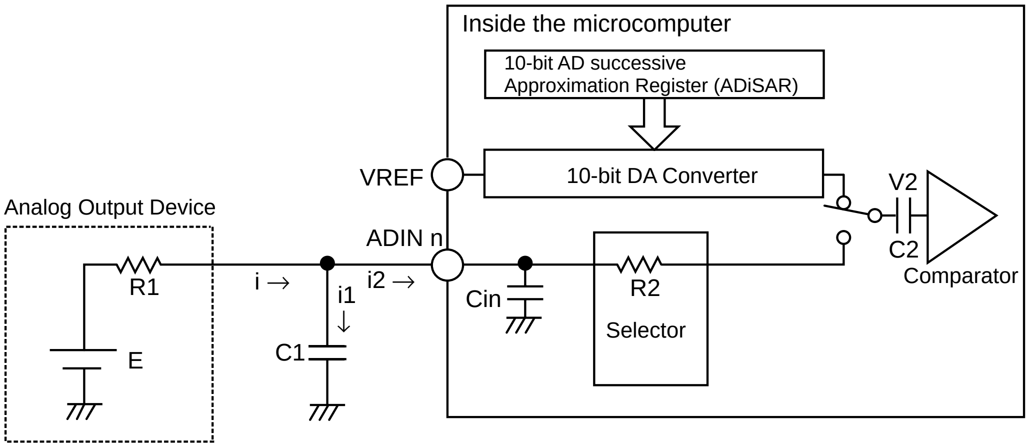
  If you want to read the A-D conversion completion bit (Single Mode Register 0's D5 bit or Scan Mode Register 0's D5 bit) immediately after A-D conversion has started, be sure to adjust the timing one clock cycle by, for example, inserting a NOP instruction before you read.

- Rated value of absolute accuracy

  The rated value of absolute accuracy is that of the microcomputer alone, premised on an assumption that power supply wiring on the board where the microcomputer is mounted is stable and unaffected by noise. When designing the board, pay careful attention to its layout by, for example, separating AVCC0, AVSS0, and VREF0 from other digital power supplies or protecting the analog input pins against noise from other digital signals.

- Regarding the analog input pins

Figure A4.9.1 shows an internal equivalent circuit of the analog input unit. To obtain exact A-D conversion results, it is necessary that the A-D conversion circuit finishes charging its internal capacitor C2 within a designated time (sampling time). To meet this sampling time requirement, we recommend connecting a stabilizing capacitor, C1, external to the chip.
The following shows the analog output device’s output impedance and how to determine the value of the external stabilizing capacitor to meet this timing requirement. Also shown below is the case where the analog output device’s output impedance is low and the external stabilizing capacitor C1 is unnecessary.



C1: Board's parasitic capacitance + stabilizing C
VREF: Analog reference voltage
C2: Comparator capacitance (approx. 2.9 pF)
Cin: Input pin capacitance (approx. 10pF)

R2: Selector's parastic resistance (1 - 2 kΩ)
R1: Analog output device's resistance
V2: Voltage across C2
E: Analog output device's voltage

**Figure A4.9.1 Internal Equivalent Circuit of the Analog Input Unit**

(a) Example for calculating the value of an external stabilizing capacitor C1 (recommended)

In Figure A4.9.1, as we calculate the capacitance of C1, we assume R1 is infinitely large, that the current needed to charge the internal capacitor C2 is sourced from C1, and that the voltage fluctuation due to C1 and C2 capacitance divisions, Vp, is 0.1 LSB or less. For the10-bit A-D converter where VREF is 5.12 V, the 1 LSB determination voltage = 5.12 V / 1024 = 5 mV. With up to 0.1 LSB voltage fluctuations considered, this equals 0.5 mV fluctuation.

The relationship between C1 and C2 capacitance divisions and Vp is obtained by the equation:

$$\mathrm{Vp} = \frac{\mathrm{C2}}{\mathrm{C1} + \mathrm{C2}} \times (\mathrm{E} - \mathrm{V2}) \cdots\cdots \text{Eq. (A-1)}$$

Also, Vp is obtained by the equation:

$$\mathrm{Vp} = \mathrm{Vp1} \times \sum_{i=0}^{x-1} \frac{1}{2^{i}} < \frac{\mathrm{VREF}}{10 \text{ x } 2^{x}} \cdots\cdots \text{Eq. (A-2)}$$

Notes: • Where Vp1 = voltage fluctuation in first A-D conversion.

• The exponent *x* is 10 because of a 10-bit resolution A-D converter.

When Eqs. (A-1) and (A-2) are solved,

$$\mathrm{C1} = \mathrm{C2}\left\{ \frac{\mathrm{E} - \mathrm{V2}}{\mathrm{Vp1}} - 1 \right\} \cdots\cdots \text{Eq. (A-3)}$$

$$\therefore\ \mathrm{C1} > \mathrm{C2}\left\{10 \times 2^{x} \times \sum_{i=0}^{x-1} \frac{1}{2^{i}} - 1 \right\} \cdots\cdots \text{Eq. (A-4)}$$

Thus, for 10-bit resolution A-D converters where C2 = 2.9 pF, C1 is 0.06 µF or greater.

Use this for reference when determining the value of C1.

(b) Maximum value of the output impedance R1 when not adding C1

In Figure A4.9.1, if the external capacitor C1 is not used, examination must be made of whether C2 can be fully charged. First, the following shows the equation to find i2 when C1 is nonexistent in Figure A4.9.1.

$$\mathrm{i2} = \frac{\mathrm{C2}\,(\mathrm{E} - \mathrm{V2})}{\mathrm{Cin} \times \mathrm{R1} + \mathrm{C2}\,(\mathrm{R1} + \mathrm{R2})} \times \exp\left\{ \frac{-t}{\mathrm{Cin} \text{ x } \mathrm{R1} \times \mathrm{C2}\,(\mathrm{R1} + \mathrm{R2})} \right\} \cdots\cdots \text{Eq. (B-1)}$$

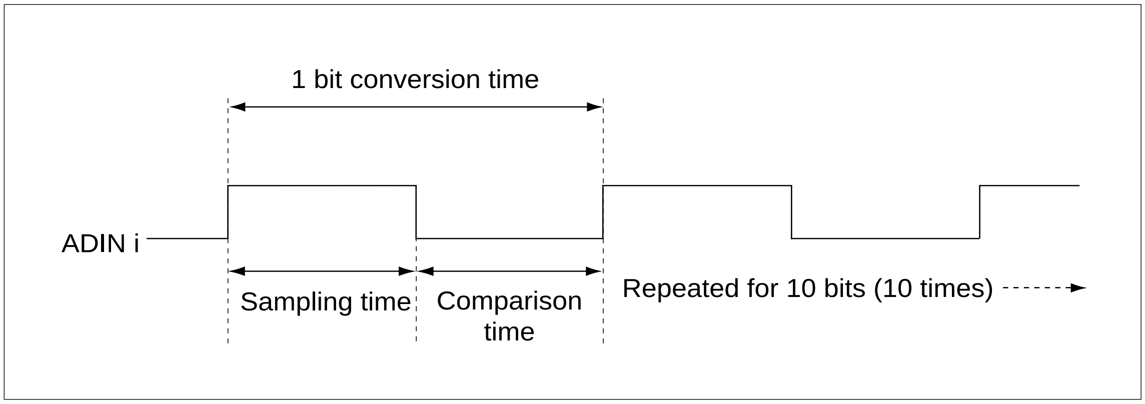


**Figure A4.9.2 A-D Conversion Timing Diagram**

The time needed for charging C2 must be within the sampling time (in Figure A4.9.2, A-D Conversion Timing Diagram) divided by 2.

Assuming t = T (time needed for charging C2)

$$T = \frac{\text{Sampling time}}{2} = \frac{\text{A-D conversion time}}{10 \times 4}$$

Therefore, from Eq. (B-1), the time needed for charging C2 is

$$T = (\text{time needed for charging C2}) > \text{Cin} \times \text{R1} + \text{C2}\,(\text{R1} + \text{R2}) \text{ -------------- Eq. (B-2)}$$

Thus, the maximum value of R1 as an approximate guide can be obtained by the equation:

$$\text{R1} < \frac{\dfrac{\text{A-D conversion time}}{10 \times 4} - \text{C2} \times \text{R2}}{\text{Cin} + \text{C2}} \text{ -------------------------------- Eq. (B-3)}$$

The table below shows an example of how to calculate the maximum value of R1 during A-D conversion mode when Xin = 10 and 8 MHz.

| Xin | BCLK period | Conversion mode | Speed mode | Conversion cycles | T (C2 charging time) in ns | Maximum value of R1 (Ω) |
|---|---|---|---|---|---|---|
| 10MHz | 50ns | A-D conversion | Normal | 294 | 367 | 28,225 |
| | | mode/Single | Double speed | 168 | 210 | 16,054 |
| 8MHz | 62.5ns | A-D conversion | Normal | 294 | 459 | 35,357 |
| | | mode/Single | Double speed | 168 | 262 | 20,085 |

Note: • The above conversion cycles do not include dummy cycles at the start and end of conversion.

In comparate mode, because sampling and comparison each are performed only once, the maximum value of R1 can be derived from the equation

$$\text{R1} > \frac{\dfrac{\text{A-D conversion time}}{4} - \text{C2} \times \text{R2}}{\text{Cin} + \text{C2}} \text{ -------------------------------- Eq. (B-4)}$$

The table below shows an example of how to calculate the maximum value of R1 during comparate mode when Xin = 10 and 8 MHz.

| Xin | BCLK period | Conversion mode | Speed mode | Conversion cycles | T (C2 charging time) in ns | Maximum value of R1 (Ω) |
|---|---|---|---|---|---|---|
| 10MHz | 50ns | comparate mode | Normal | 42 | 525 | 40,473 |
| | | /Single | Double speed | 24 | 300 | 23,031 |
| 8MHz | 62.5ns | comparate mode | Normal | 42 | 656 | 50,628 |
| | | /Single | Double speed | 24 | 375 | 28,845 |

Note: • The above conversion cycles do not include dummy cycles at the start and end of conversion.

## Appendix 4.10 Precautions on Serial I/O

### Appendix 4.10.1 Precautions on Using CSIO Mode

- **Settings of SIO Transmit/Receive Mode Register and SIO Baud Rate Register**

  The SIO Transmit/Receive Mode Register and SIO Baud Rate Register and the Transmit Control Register's BRG count source select bit must always be set when not operating. When transmitting or receiving data, be sure to check that transmission and/or reception under way has been completed and clear the transmit and receive enable bits before you set the registers.

- **Settings of Baud Rate (BRG) Register**

  If you selected f(BCLK) with the BRG clock source select bit, make sure the BRG register value you set does not exceed 2 Mbps.

- **About successive transmission**

  To transmit multiple data successively, set the next transmit data in the SIO Transmit Buffer Register before transmission of the preceding data is completed.

- **About reception**

  Because during CSIO mode the receive shift clock is derived from operation of the transmit circuit, you need to execute transmit operation (by sending dummy data) even when you only want to receive data. In this case, note that if the port function is set for TXD pin (by setting the operation mode register to 1), dummy data is actually output from the pin.

- **About successive reception**

  To receive multiple data successively, set data (dummy data) in the SIO Transmit Buffer Register before the transmitter starts sending data.

- **Transmit/receive operations using DMA**

  To transmit/receive data in DMA request mode, enable the DMAC to accept transfer requests (by setting the DMA Mode Register) before you start serial communication.

- **About the receive-finished bit**

  If a receive error (overrun error) occurs, the receive-finished bit cannot be cleared by reading out the receive buffer register. In this case, it can only be cleared by clearing the receive enable bit.

- **About overrun error**

  If all bits of the next receive data are received in the SIO Receive Shift Register before you read out the SIO Receive Buffer Register (an overrun error occurs), the receive data is not stored in the Receive Buffer Register and the Receive Buffer Register retains the previously received data. Thereafter, although receive operation is continued, no receive data is stored in the Receive Buffer Register (the receive status bit = 1). To restart reception normally, you need to temporarily clear the receive enable bit before you restart. This is the only way you can clear the overrun error flag.

- **About DMA transfer request generation during SIO transmission**

  If the Transmit Buffer Register becomes empty (the transmit buffer empty flag = 1) while the transmit enable bit is set to 1 (transmit enabled), an SIO transmit buffer empty DMA transfer request is generated.

- **About DMA transfer request generation during SIO reception**

  When the receive-finished bit is set to 1 (the receive buffer register full), a receive-finished DMA transfer request is generated. However, if an overrun error has occurred, this DMA transfer request is not generated.

## Appendix 4.10.2 Precautions on Using UART Mode

- **Settings of SIO Transmit/Receive Mode Register and SIO Baud Rate Register**

  The SIO Transmit/Receive Mode Register and SIO Baud Rate Register and the Transmit Control Register's BRG count source select bit must always be set when not operating. When transmitting or receiving data, be sure to check that transmission and/or reception under way has been completed and clear the transmit and receive enable bits before you set the registers.

- **Settings of Baud Rate (BRG) Register**

  If you selected f(BCLK) with the BRG clock source select bit, make sure the BRG register value you set is equal to or greater than 7.

  The value written to the SIO Baud Rate Register becomes effective beginning with the next period after the BRG counter finished counting. However, when transmit and receive operations are disabled, the register value can be changed at the same time you write to the register.

- **Transmit/receive operations using DMA**

  To transmit/receive data in DMA request mode, enable the DMAC to accept transfer requests (by setting the DMA Mode Register) before you start serial communication.

- **About overrun error**

  If all bits of the next receive data are received in the SIO Receive Shift Register before you read out the SIO Receive Buffer Register (an overrun error occurs), the receive data is not stored in the Receive Buffer Register and the Receive Buffer Register retains the previously received data. Once an overrun error occurs, no receive data is stored in the Receive Buffer Register although receive operation is continued. To restart reception normally, you need to temporarily clear the receive enable bit before you restart. This is the only way you can clear the overrun error flag.

- **Flags indicating the status of UART receive operation**

  Following flags are available that indicate the status of receive operation during UART mode.

  - SIO Receive Control Register receive status bit
  - SIO Receive Control Register receive-finished bit
  - SIO Receive Control Register receive error sum bit
  - SIO Receive Control Register overrun error bit
  - SIO Receive Control Register parity error bit
  - SIO Receive Control Register framing error bit

  The manner in which the receive-finished bit and various error bit flags are cleared varies depending on whether an overrun error has occurred or not, as described below.

  [When no overrun error has occurred]

  Said bits can be cleared by reading the lower byte from the receive buffer register or clearing the receive enable bit to 0.

  [When an overrun error has occurred]

  Said bits can only be cleared by clearing the receive enable bit to 0.

## Appendix 4.11 Precautions on RAM Backup Mode

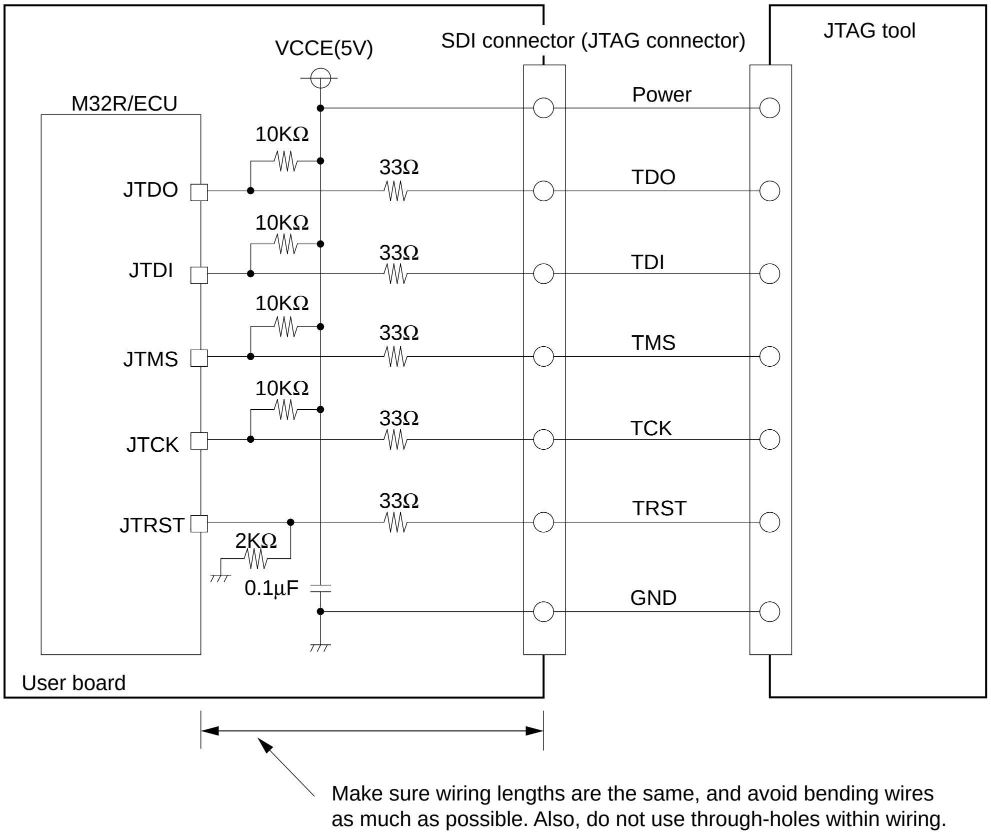
### Appendix 4.11.1 Precautions to Be Observed at Power-on

When changing port X from input mode to output mode after power-on, pay attention to the following.
If port X is set for output mode while no data is set in the Port X Data Register, the port's initial output level is indeterminate. Therefore, be sure to set the output high level in the Port X Data Register before you set port X for output mode. Unless this method is followed, port output may go low at the same time port output is set after the clock oscillation has stabilized, causing the device to enter RAM backup mode.

# Appendix 4.12 Precautions on Processing JTAG Pins

## Appendix 4.12.1 Precautions on Board Design when Using JTAG

The JTAG pins require that wiring lengths be matched during board design in order to accomplish fast, highly reliable communication with JTAG tools.
An example of how to process pins when using JTAG tools is shown below.



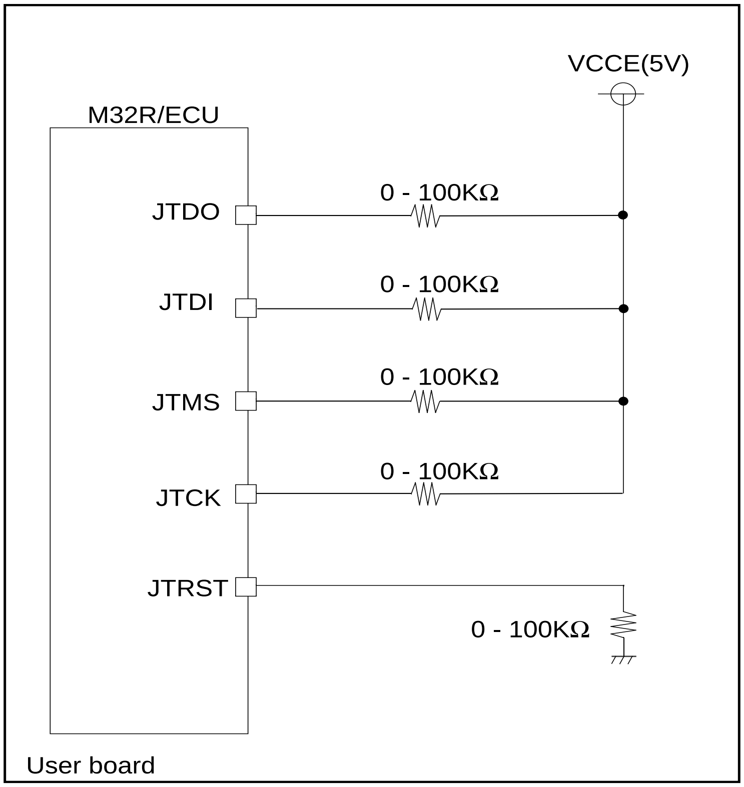
Make sure wiring lengths are the same, and avoid bending wires as much as possible. Also, do not use through-holes within wiring.

Notes: • Only if the JTRST pin is firmly tied to ground, it dosn't matter whether the JTDO, JTDI, JTMS, and JTCK pins are pulled high or pulled low.
• Even when not using JTAG tools, always be sure to process each pin. The same pulldown/ pullup resistance values as when using JTAG tools may be used without causing any problem.

**Figure A4.12.1 Example for Processing Pins when Using JTAG Tools**

## Appendix 4.12.2 Processing Pins when Not Using JTAG

The diagram below shows how to process JTAG pins when not using these pins (i.e. for boards that do not have pins/connectors connecting to JTAG tools).



Note: • Only if the JTRST pin is firmly tied to ground, it dosn’t matter whether the JTDO, JTDI,JTMS, and JTCK pins are pulled high or pulled low.

**Figure A4.12.2 Example for Processing Pins when Not Using JTAG**

# Appendix 4.13 Precautions about Noise

The following describes precautions to be taken about noise and corrective measures against noise. The corrective measures described here are theoretically effective for noise, but require that the application system incorporating these measures be fully evaluated before it can actually be put to use.

## Appendix 4.13.1 Reduction of Wiring Length

Wiring on the board may serve as an antenna to draws noise into the microcomputer. Shorter the total wiring length, the smaller the possibility of drawing noise into the microcomputer.

### (1) Wiring of the $\overline{\text{RESET}}$ pin

Reduce the length of wiring connecting to the $\overline{\text{RESET}}$ pin. Especially when connecting a capacitor between the $\overline{\text{RESET}}$ and VSS pins, make sure it is connected to each pin in the shortest distance possible (within 20 mm).

**<Reasons>**

Reset is a function to initialize the internal logic of the microcomputer. The width of a pulse applied to the $\overline{\text{RESET}}$ pin is important and is therefore stipulated as part of timing requirements. If a pulse in width shorter than the stipulated duration (i.e., noise) is applied to the $\overline{\text{RESET}}$ pin, the microcomputer will not be reset for a sufficient duration of time and exit the reset state before its internal logic is fully initialized, causing the program to go malfunction.

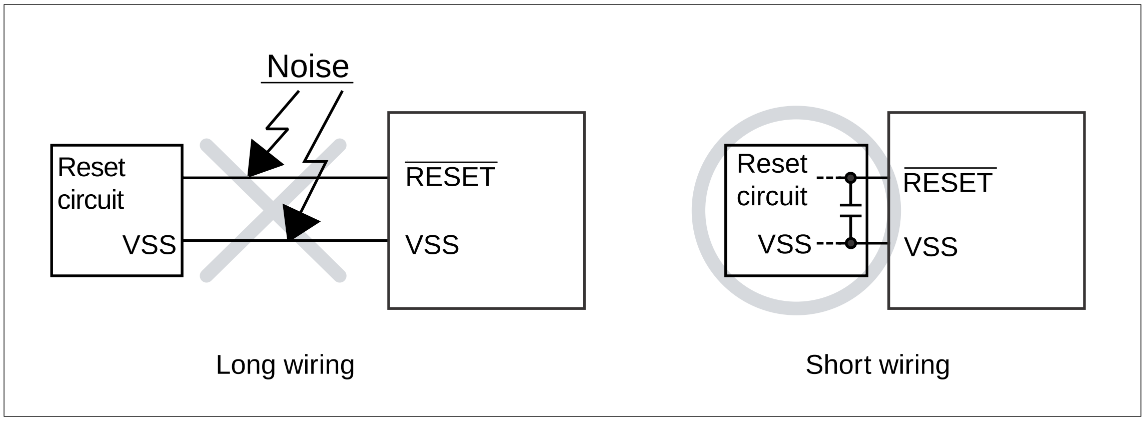


**Figure A4.13.1 Example Wiring of the $\overline{\text{RESET}}$ Pin**

### (2) Wiring of clock input/output pins

Use as much thick and short wiring as possible for connections to the clock input/output pins. When connecting a capacitor to the oscillator, make sure its grounding lead wire and the OSC-VSS pin on the microcomputer are connected in the shortest distance possible (within 20 mm). Also, make sure the VSS pattern used for clock oscillation is a large ground plane and is connected to GND.

**<Reasons>**

The microcomputer operates synchronously with the clock generated by the oscillator circuit. Inclusion of noise on the clock input/output pins causes the clock waveform to become distorted, which may result in the microcomputer operating erratically or getting out of control. Also, if a noise-induced potential difference exists between the microcomputer's VSS level and that of the oscillator, the clock fed into the microcomputer may not be an exact clock.

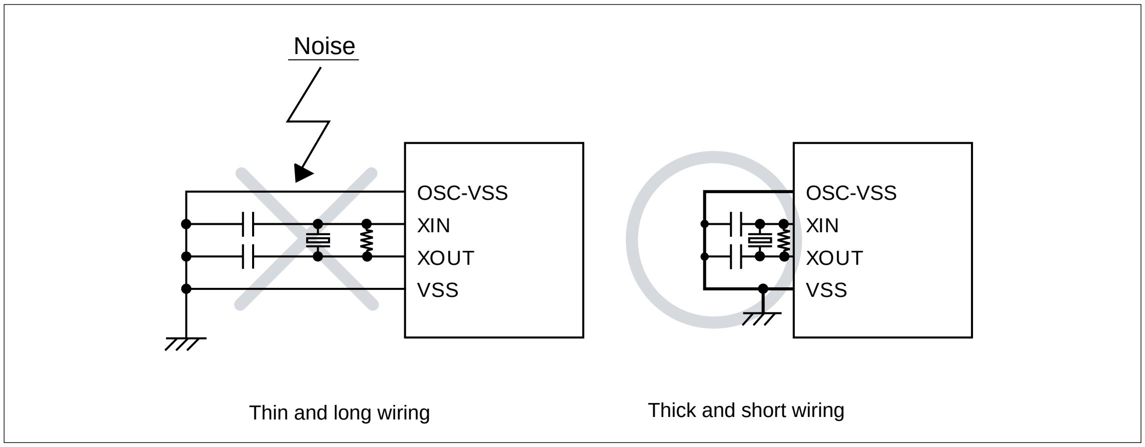


**Figure A4.13.2 Example Wiring of Clock Input/Output Pins**

### (3) Wiring of the VCNT pin

Use as much thick and short wiring as possible for connections to the VCNT pin.
When connecting a capacitor to VCNT, make sure its grounding lead wire and the OSC-VSS pin on the microcomputer are connected in the shortest distance possible.
Also, make sure the VSS pattern used for VCNT is a large ground plane and is connected to GND.

**<Reasons>**

The external circuit inserted for the VCNT pin plays the role of a low-pass filter that stabilizes the PLL's internal voltage and eliminates noise. If noise exceeding the limit of the low-pass filter penetrates into the wiring, the internal circuit may be disturbed by that noise and become unable to produce a precise clock, causing the microcomputer to operate erratically or get out of control.

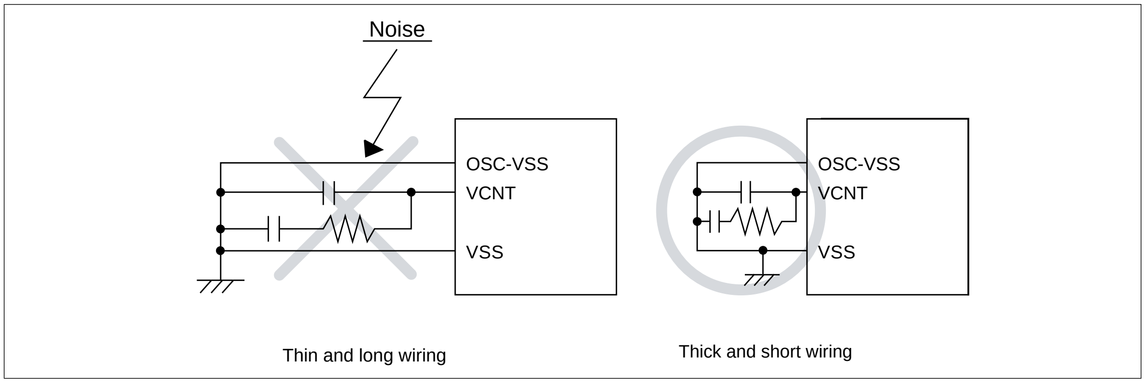


**Figure A4.13.3 Example Wiring of the VCNT Pin**

### (4) Wiring of operation mode setup pins

When connecting operation mode setup pins and the VCC or VSS pin, make sure they are connected in the shortest distance possible.

**<Reasons>**

The levels of operation mode setup pins affect the microcomputer's operation mode. When connecting the operation mode setup pins and the VCC or VSS pin, be careful that no noise-induced potential difference will exist between the operation mode setup pins and the VCC or VSS pin. This is because the presence of such a potential difference makes operation mode instable, which may result in the microcomputer operating erratically or getting out of control.

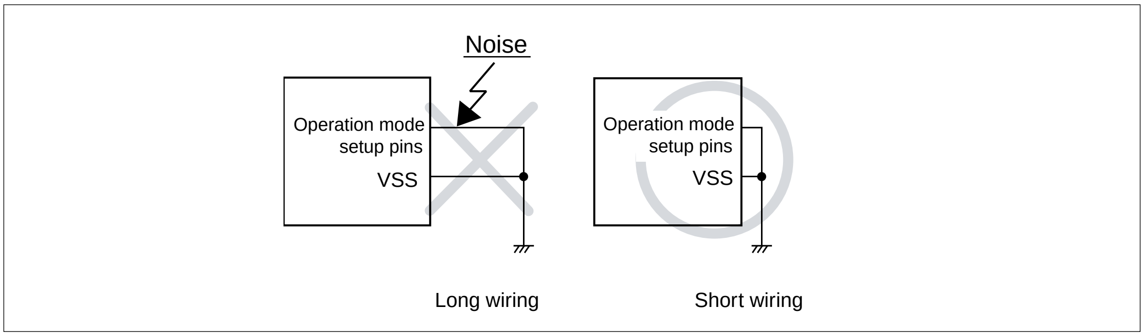


**Figure A4.13.4 Example Wiring of the MOD0 and MOD1 Pins**

## Appendix 4.13.2 Inserting a Bypass Capacitor between VSS and VCC Lines

Insert a bypass capacitor of about 0.1 μF between VSS and VCC lines in such a way as to meet the requirements described below.

- The wiring length between the VSS pin and bypass capacitor and that between the VCC pin and bypass capacitor are equal.
- The wiring length between the VSS pin and bypass capacitor and that between the VCC pin and bypass capacitor are the shortest distance possible.
- The VSS and VCC lines have a greater wiring width than that of other signal lines.

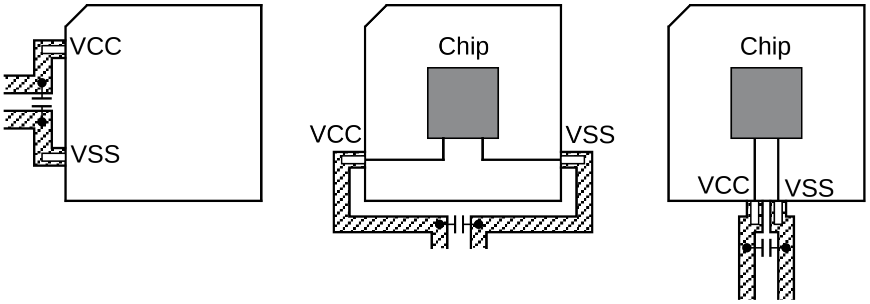


**Figure A4.13.5 Example of a Bypass Capacitor Inserted between VSS and VCC Lines**

## Appendix 4.13.3 Processing Analog Input Pin Wiring

Insert a resistor of about 100 to 500 Ω in series to the analog signal line connecting to the analog input pin at a position as close to the microcomputer as possible. Also, insert a capacitor of about 100 pF between the analog input pin and AVSS pin at a position as close to the AVSS pin as possible.

**<Reasons>**

The signal fed into the analog input pin (e.g., A-D converter input pin) normally is an output signal from a sensor. In many cases, a sensor to detect changes of event is located apart from the board on which the microcomputer is mounted, so that wiring to the analog input pin is inevitably long. Because a long wiring serves as an antenna which draws noise into the microcomputer, the signal fed into the analog input pin tends to be noise-ridden. Furthermore, if the capacitor connected between the analog input pin and AVSS pin is grounded at a position apart from the AVSS pin, noise ridding on the ground line may penetrate into the microcomputer via the capacitor.

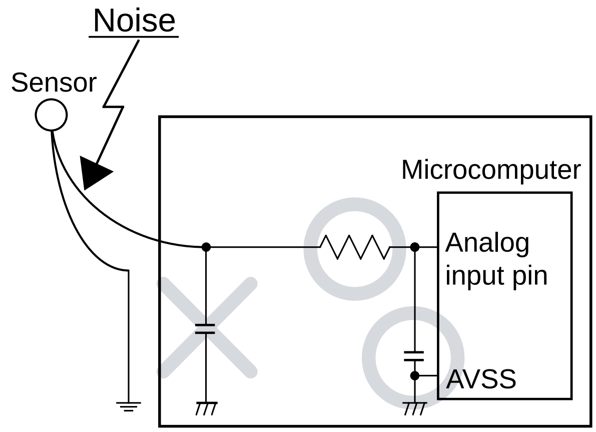


**Figure A4.13.6 Example of a Resistor and Capacitor Inserted for the Analog Signal Line**

### Appendix 4.13.4 Consideration about the Oscillator and VCNT Pin

The oscillator that generates the fundamental clock for microcomputer operation requires consideration to make it less susceptible to influences from other signals.

#### (1) Avoidance from large-current signal lines

Signal lines in which a large current flows exceeding the range of current values that the microcomputer can handle must be routed as far away from the microcomputer (especially the oscillator and VCNT pin) as possible. Also, make sure the circuit is protected with a GND pattern.

**<Reasons>**

Systems using the microcomputer contain signal lines to control, for example, a motor, LED, and thermal head. When a large current flows in these signal lines, it generates noise due to mutual inductance (M).

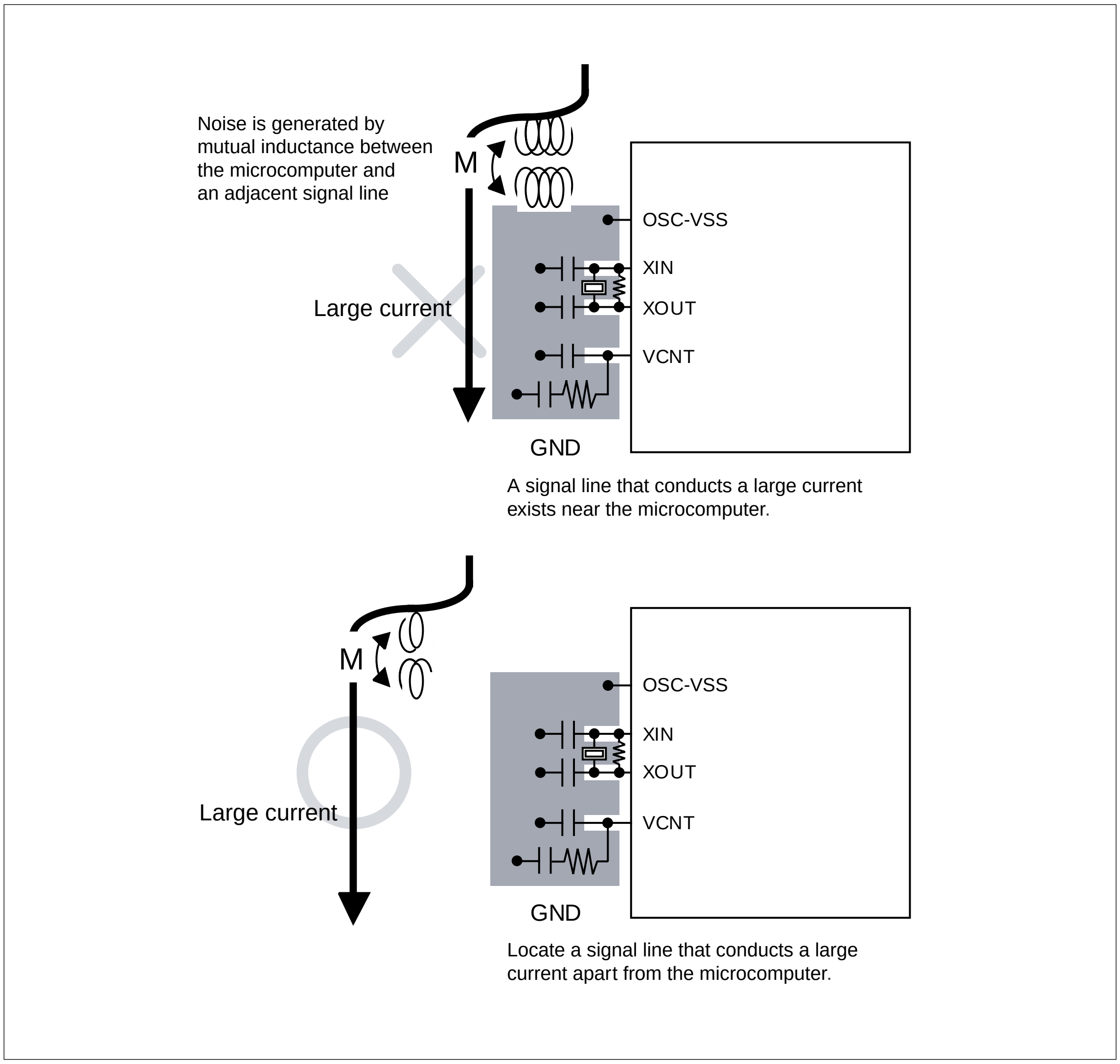

Noise is generated by mutual inductance between the microcomputer and an adjacent signal line
M
Large current
OSC-VSS
XIN
XOUT
VCNT
GND
A signal line that conducts a large current exists near the microcomputer.
M
Large current
OSC-VSS
XIN
XOUT
VCNT
GND
Locate a signal line that conducts a large current apart from the microcomputer.

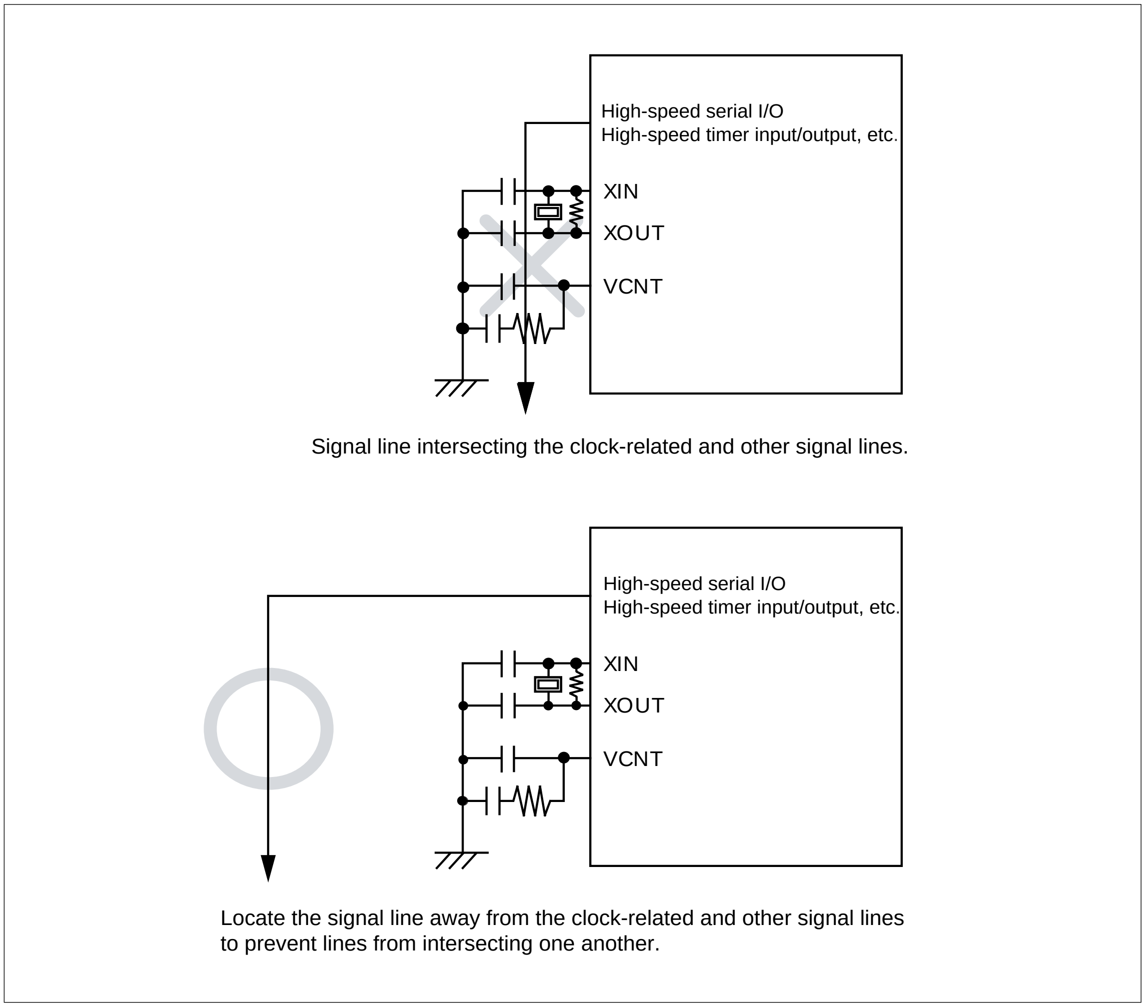
**Figure A4.13.7 Example Wiring of Large-current Signal Lines**

### (2) Avoiding effects of rapidly level-changing signal lines

Locate signal lines whose levels change rapidly as far away from the oscillator as possible. Also, make sure the rapidly level-changing signal lines will not intersect the clock-related signal lines and other noise-sensitive signal lines.

**<Reasons>**

Rapidly level-changing signal lines tend to affect other signal lines as their voltage level frequently rises and falls. Especially if these signal lines intersect the clock-related signal lines, they will cause the clock waveform to become distorted, which may result in the microcomputer operating erratically or getting out of control.


High-speed serial I/O
High-speed timer input/output, etc.
XIN
XOUT
VCNT
High-speed serial I/O
High-speed timer input/output, etc.
XIN
XOUT
VCNT

Signal line intersecting the clock-related and other signal lines.

Locate the signal line away from the clock-related and other signal lines to prevent lines from intersecting one another.

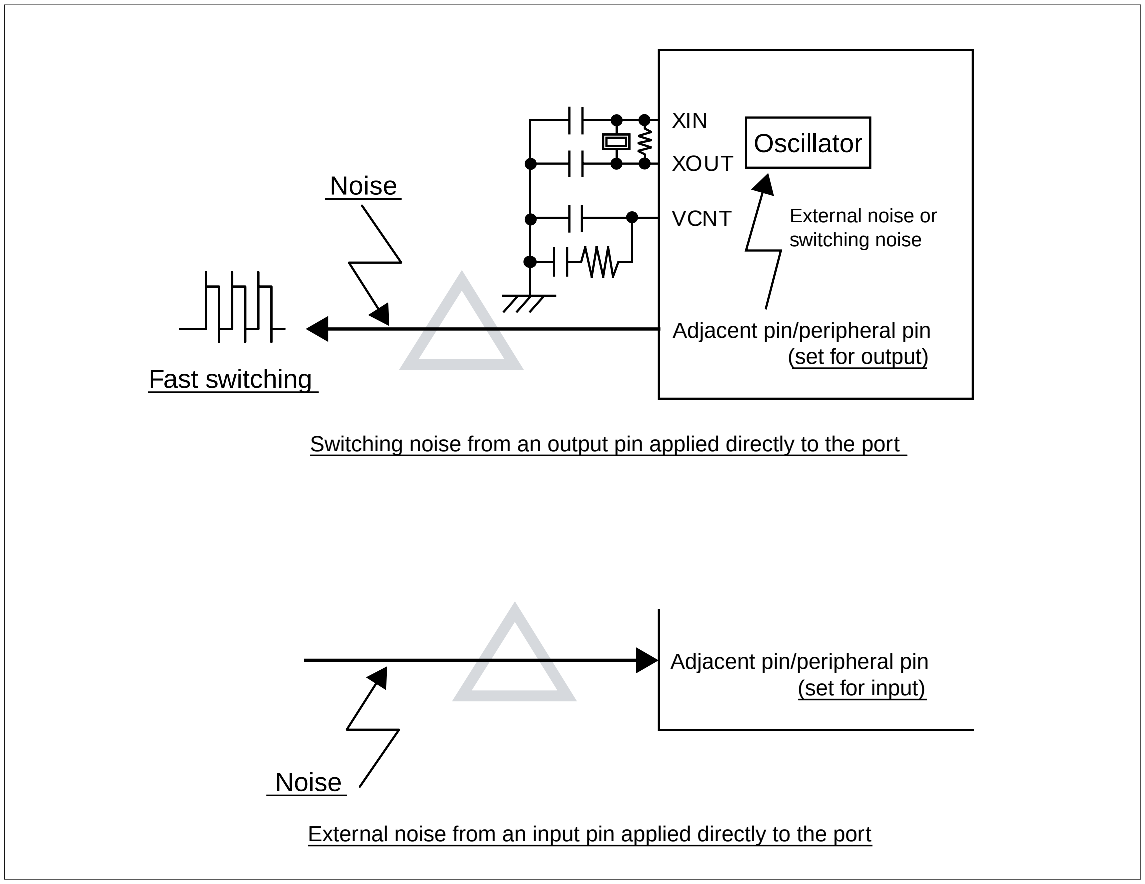
**Figure A4.13.8 Example Wiring of Rapidly Level-changing Signal Lines**

### (3) Protection against signal lines that are the source of strong noise

Do not use any pin that will probably be subject to strong noise for an adjacent port near the oscillator and VCNT pins. If the pin can be left unused, set it for input and connect to GND via a resistor, or fix it to output and leave open. If the pin needs to be used, it is recommended that it be used for input-only.
For protectioon against a still stronger noise source, set the adjacent port for input and connect to GND via a resistor, and use those that belong to the same port group as much for input-only as possible. If greater stability is required, do not use those that belong to the same port group and set them for input and connect to GND via a resistor. If they need to be used, insert a limiting resistor for protection against noise.

**<Reasons>**

If the ports or pins adjacent to the oscillator and VCNT pins operate at high speed or are exposed to strong noise from an external source, noise may affect the oscillator circuit, causing its oscillation to become instable.

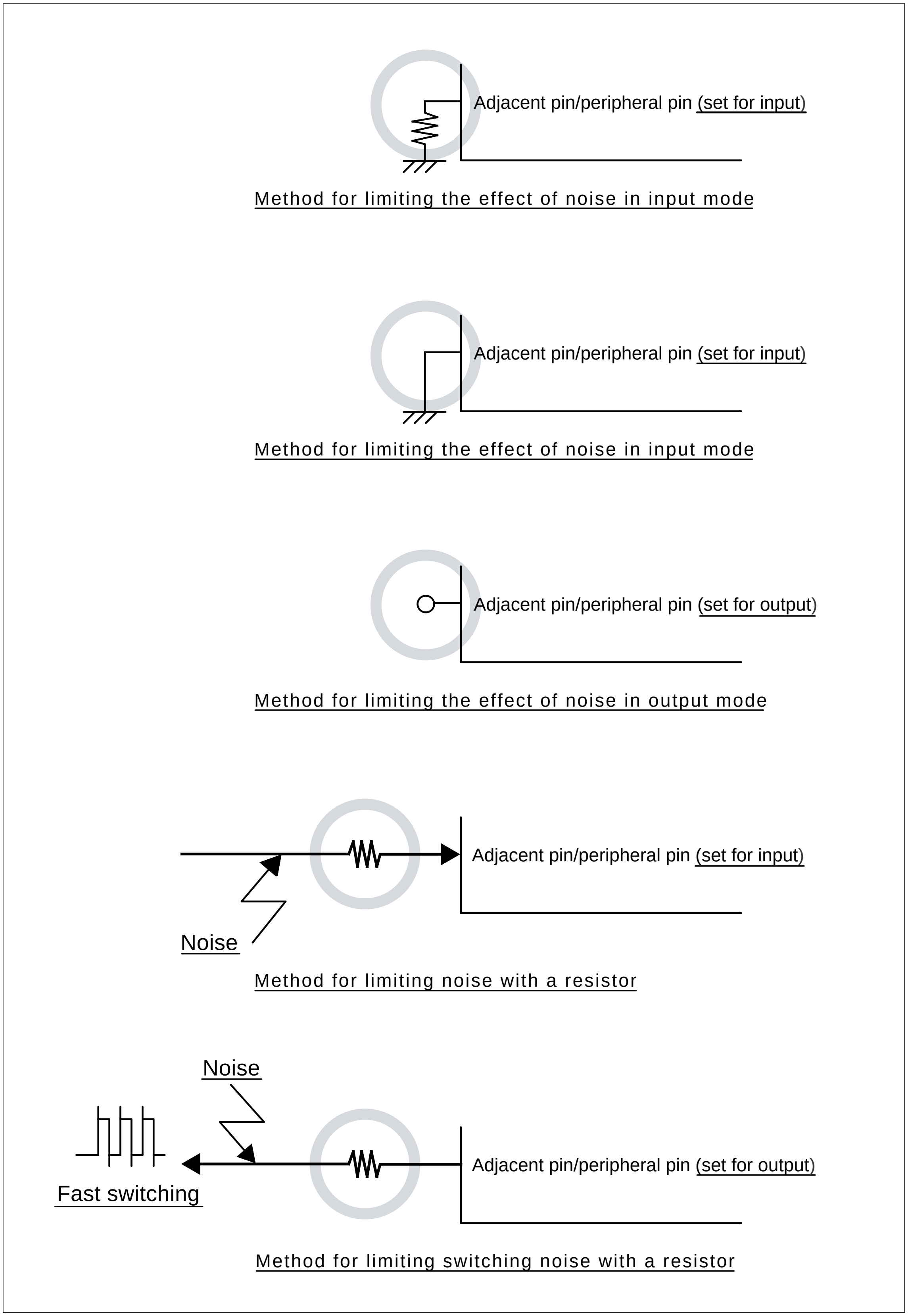

XIN
XOUT
VCNT
Oscillator
Noise
External noise or switching noise
Fast switching
Adjacent pin/peripheral pin (set for output)

Switching noise from an output pin applied directly to the port

Adjacent pin/peripheral pin (set for input)
Noise

External noise from an input pin applied directly to the port

**Figure A4.13.9 Example Processing of a Noise-laden Pin**


Adjacent pin/peripheral pin (set for input)
Adjacent pin/peripheral pin (set for input)
Adjacent pin/peripheral pin (set for output)
Adjacent pin/peripheral pin (set for input)
Noise
Noise
Fast switching
Adjacent pin/peripheral pin (set for output)

Method for limiting the effect of noise in input mode

Method for limiting the effect of noise in input mode

Method for limiting the effect of noise in output mode

Method for limiting noise with a resistor

Method for limiting switching noise with a resistor

**Figure A4.13.10 Example Processing of Pins Adjacent to the Oscillator and VCNT Pins**

### Appendix 4.13.5 Processing Input/Output Ports

For input/output ports, take the appropriate measures in both hardware and software following the procedure described below.

#### Hardware measures

- Insert resistors of 100 Ω (or more) in series to input/output ports.

#### Software measures

- For input ports, read out data in a program two or more times to verify that levels match.
- For output ports, rewrite the data register at certain intervals, because there is a possibility of the output data being inverted by noise.
- Rewrite the direction register at certain intervals.

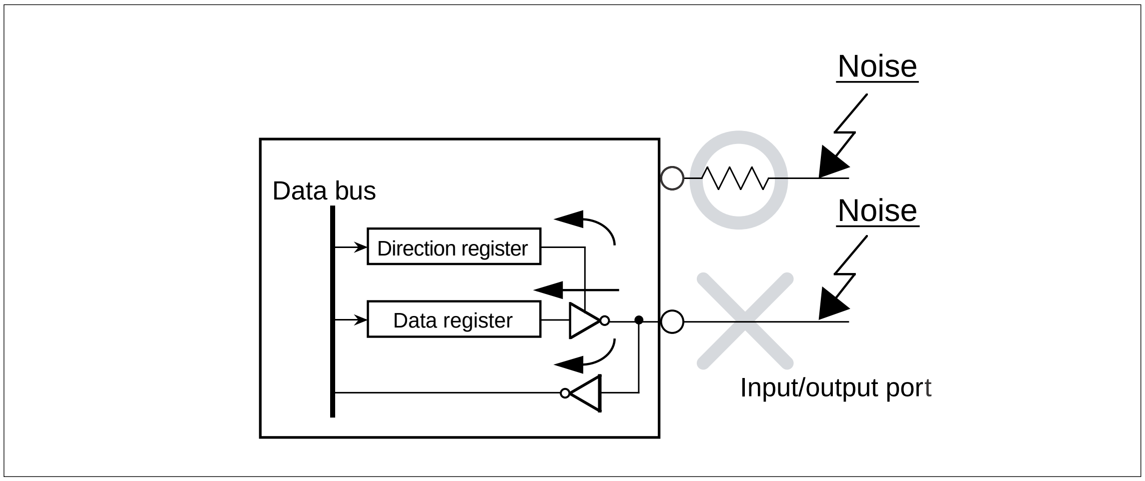


**Figure A4.13.11 Example Processing of Input/Output Ports**

* This is a blank page. *

**RENESAS 32-BIT RISC SINGLE-CHIP MICROCOMPUTER**
**USER'S MANUAL**
**32171 Group**

| | |
|---|---|
| **Publication Data :** | **Rev.0.10 Apr 08, 2000** |
| | **Rev.2.00 Sep 19, 2003** |
| **Published by :** | **Sales Strategic Planning Div.** |
| | **Renesas Technology Corp.** |

# 32171 Group
# User's Manual

RENESAS

Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan